PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

FACULTAD DE INGENIERÍA

MAESTRÍA EN REDES DE COMUNICACIÓN

PERFIL DEL TRABAJO PREVIO LA OBTENCIÓN DEL TÍTULO DE:

MASTER EN REDES DE COMUNICACIÓN

TEMA:

"DISEÑO E IMPLEMENTACIÓN DE UN WEB SERVICE EN EL GAD MUNICIPAL DE CHORDELEG, PARA PROVEER DE UNA APP ANDROID QUE PERMITA CONSULTAR LOS PRINCIPALES TRIBUTOS MUNICIPALES COMO PREDIO URBANO, PREDIO RÚSTICO, AGUA POTABLE Y PATENTE MUNICIPAL"

JHOVANY MAURICIO BUELE VILLA

Quito - 2017

AUTORÍA

Yo, Jhovany Mauricio Buele Villa, portador de la cédula de ciudadanía Nro. 0104129390, declaro bajo juramento que la presente investigación es de total responsabilidad del autor, y que se ha respetado las diferentes fuentes de información realizando las citas correspondientes. Esta investigación no contiene plagio alguno y es resultado de un trabajo serio desarrollado en su totalidad por mi persona.

_____

Jhovany Mauricio Buele Villa

DEDICATORIA

A Dios, por haberme dado hasta ahora una vida llena de salud para lograr mis objetivos, además de su infinita bondad y amor.

A mis padres Gilberto y Alejandrina, por sus valores y sabios concejos que me forman día a día como un mejor ser humano, a más de su sacrificio y apoyo incondicional durante mi vida profesional.

A mi esposa Natalia e hijos Camila e Israel, por ser la fuente de inspiración para plasmar mis sueños y cumplir con mis metas y objetivos en el transcurso de la vida.

A mis familiares, por sus palabras de apoyo y estímulos de superación durante mi formación profesional, sin dejar de lado el gran cariño y amor demostrado.

AGRADECIMIENTO

En primer lugar agradezco a Dios por darme la vida, a mis padres por su sacrificio y apoyo incondicional en todo momento, a mi esposa e hijos por impulsarme en el cumplimiento de mis metas, a los docentes de la “Maestría de Redes de Comunicaciones” de la “Pontificia Universidad Católica del Ecuador” por compartir sus conocimientos en el transcurso de la maestría, al Dr. Gustavo Chafla PhD. por su guía como director de tesis, al Gobierno Autónomo Descentralizado Municipal de Chordeleg por permitir implementar mi proyecto de tesis en su institución y a todos mis familiares y amigos por sus palabras de apoyo para salir adelante.

TABLA DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Las Tecnologías de la Información y Comunicación TIC´s, están sufriendo un gran desarrollo, lo cual está afectando prácticamente a todos los campos de una sociedad. Hablando específicamente en el presente caso de estudio, el campo gubernamental; el Gobierno Autónomo Descentralizado Municipal de Chordeleg está enfocado en tomar interés en su desarrollo para emplear las TIC´s como una herramienta de soporte en los servicios que ofrece la institución.

Este trabajo presenta el desarrollo de un sistema de consulta de los principales tributos municipales del GAD Municipal de Chordeleg, tales como, predio urbano, predio rústico, agua potable y patente municipal. Por tal motivo la razón principal de elaborar este proyecto, es diseñar y dimensionar un sistema escalable y efectivo en base a un análisis de tráfico de datos, que permita a la ciudadanía del cantón Chordeleg en general, consultar dichos tributos municipales desde cualquier lugar donde disponga de internet, a través de una Aplicación Android instalada en sus dispositivos móviles.

El sistema consiste en un Web Service que será implementado en un servidor dedicado dentro del GAD Municipal de Chordeleg, el cual se encargará de gestionar con la base de datos de los tributos municipales de la institución, las peticiones solicitadas por los contribuyentes (predio urbano, predio rústico, agua potable y patente municipal) a través de una App Móvil desarrollada para sistemas Android.

Actualmente existen en otras áreas, sistemas que permiten realizar consultas a través de una App Android, sin embargo muchos de ellos no son diseñados en base a un estudio que garantice un ancho de banda adecuado según el volumen masivo de datos que se manejará en diferentes escenarios y mucho menos con miras al futuro, volviéndolo así un sistema no escalable e inestable con caídas y congelamientos repentinos del servicio.

Es así que mediante este trabajo se plantea diseñar e implementar un sistema que a más de brindar un servicio de consulta a través de una App Android, sea escalable, con el fin de proyectar en un futuro la implementación de otros servicios, tales como, la consulta a través de una página web, y sobre todo que sea capaz de responder ante situaciones en las cuales existirán saturaciones por las consultas masivas de datos; todo esto sin la necesidad de implementar nuevos equipos y mucho menos modificar el sistema desarrollado. A más de ello se pretende implementar calidad de servicio (QoS) para priorizar el tráfico y mejorar los tiempos de respuesta del servicio brindado.

JUSTIFICACIÓN

Actualmente el GAD Municipal de Chordeleg cuenta con una ventanilla de recaudación ubicada en su edificio principal, donde los contribuyentes pueden consultar y/o a su vez cancelar los tributos municipales pendientes de pago, entre los principales están el impuesto al predio urbano, el impuesto al predio rústico, el consumo de agua potable y la patente municipal; lamentablemente a más de contar con una sola ventanilla, se dispone de una sola empleada como responsable del área de recaudación, lo cual ha venido generando demoras en la atención al público y molestias a los contribuyentes en general, provocando que el valor mensual recaudado no cumpla con el objetivo planteado por la institución; a continuación menciono las principales razones por la cual surgen estos problemas:

Los contribuyentes que desean simplemente consultar sus valores pendientes de pago, necesariamente tienen que acercarse a la única ventanilla de recaudación, que en su mayoría tiene que llegar a hacer fila, generando así más trabajo de consulta para la recaudadora.

En horas pico las filas en la ventanilla de recaudación son extensas, razón por la cual, muchos de los contribuyentes que desean consultar el valor pendiente de pago, optan por regresar en otro momento; generando pérdidas de tiempo para el contribuyente.

Por el impuesto al predio urbano, de acuerdo a la ley se tienen descuentos durante los 6 primeros meses del año, y en gran demanda los contribuyentes desean consultar el valor a cancelar incluido el descuento, para lo cual necesariamente tienen que acercarse a la ventanilla de recaudación para consultar dicho descuento en ese momento dado, generando más trabajo de consulta para el recaudador.

Por el impuesto al predio rural, se tienen los contribuyentes de las parroquias que se encuentran a una distancia lejana del centro cantonal, que en el caso de las consultas necesariamente tienen que acercarse a la ventanilla de recaudación y en su gran mayoría

cuando se acercan a cancelar el valor pendiente de pago, no les alcanza el dinero, y por ende tienen que regresar a sus parroquias y volver al siguiente día con el valor completo.

El valor por consumo de agua potable se emite una vez al mes, sin embargo no se tiene una fecha exacta de la emisión, provocando molestias en los contribuyentes, ya que muchas de las veces se acercan en vano a ventanilla para consultar su valor a cancelar, lo cual también genera mayor trabajo de consulta para la recaudadora.

El valor por la patente municipal se genera de manera anual en el mes de marzo, sin embargo no se tiene una fecha exacta de la emisión, y al igual que la emisión por el consumo de agua potable, se generan molestias en los contribuyentes, ya que muchas de la veces se acercan a consultar en vano en la ventanilla de recaudación, generando mayor trabajo de consulta para la recaudadora.

Es así que hoy en día al estar en una era de gran avance de la tecnología y del internet, se la puede utilizar y aplicarla dentro del campo gubernamental. Motivado por los experiencias de la ventanilla de recaudación del GAD Municipal de Chordeleg y de los contribuyentes en general, puedo manifestar que de acuerdo a las razones mencionadas anteriormente, el principal motivo de la demora en la atención al público y la baja recaudación de los tributos municipales, es debido a que se cuenta con un solo punto como ventanilla de atención al público, donde se realizan tanto las consultas como los pagos en general. Es ahí donde surge la idea de diseñar e implementar un Web Service en el GAD Municipal de Chordeleg, para proveer de una App Android de consulta de los principales tributos municipales, con el fin de brindar un servicio alternativo de consulta para los contribuyentes en general, a más de ello dicha App les permitirá consultar desde cualquier lugar donde se cuente con un dispositivo Android que disponga de internet. También se priorizará el tráfico de datos implementando calidad de servicio (QoS) dentro de la red del GAD Municipal de Chordeleg, con el fin de mejorar los tiempos de respuesta.

Cabe indicar que actualmente el GAD Municipal de Chordeleg no dispone de ningún tipo de Web Service dentro de su red LAN, para desplegar servicios web a través de internet, y mucho menos de alguna App Android; así mismo menciono que se dispone de un router administrable pero no tiene implementado ningún modelo de calidad de servicio (QoS).

Por otra parte, es importante mencionar que en el mercado actual existen sistemas de consulta a través de una App Android, sin embargo muchos de ellos no cuentan con un estudio de diseño y análisis del tráfico de datos, lo cual es la razón y el objetivo principal de este proyecto, permitiendo obtener un sistema de consulta con proyecciones al futuro y que garantice un ancho de banda adecuado, basado en el volumen masivo de datos que se generará a partir de un número determinado de contribuyentes del GAD Municipal de Chordeleg en general; volviéndolo así un sistema escalable y sobre todo estable, frente a situaciones como saturaciones en las horas pico, caídas y congelamientos repentinos del servicio.

ANTECEDENTES

El edificio principal del GAD Municipal de Chordeleg, donde se propone desarrollar el proyecto, se encuentra ubicado en la calle 23 de Enero 4-81 y Juan Bautista Cobos, con un total de 30 empleados distribuidos en las áreas Financiera, Jurídica, Administrativa, Planificación, Movilidad y Desarrollo.

Actualmente el GAD Municipal de Chordeleg, según la base de datos que reposa en la institución, se tiene hasta la presente fecha 11454 contribuyentes, 2551 títulos por predio urbano, 11205 títulos por predio rural, 1874 títulos por consumo de agua potable y 1021 títulos por patente municipal; de lo cual indica la Analista de Determinación y Rentas del GAD Municipal de Chordeleg, que los títulos mencionados están en constante crecimiento anual.

En cuanto a hardware el GAD Municipal de Chordeleg dispone tanto de un Servidor de Producción como de un router principal administrable de la empresa mikrotik, los cuales no tienen implementado ningún tipo de web service ni calidad de servicio (QoS).

Por otro parte el GAD Municipal de Chordeleg cuenta con una sola ventanilla y un solo empleado de recaudación, donde se realizan todos los cobros y consultas de los tributos municipales; lo cual ha venido generando demoras en la atención al público y molestias a los contribuyentes en general.

Esto ha llevado en lo referente al tema, a proponer alternativas para consultar los principales tributos municipales, las cuales deben ser acompañadas de recursos tecnológicos y un análisis apegado a la realidad del lugar a cubrir la necesidad, como la implementación de un Web Service en el GAD Municipal de Chordeleg que permita proveer a sus contribuyentes de una App Android de consulta de los tributos municipales como el predio urbano, el predio rústico, el agua potable y la patente municipal. Hasta el momento el GAD Municipal de Chordeleg no dispone de ningún tipo de Web Service dentro de su red LAN, para desplegar

servicios web a través de internet, y mucho menos de alguna App Android; así mismo no se ha implementado ningún modelo de calidad de servicio (QoS) dentro de su red.

Si bien es cierto hoy en día el continuo avance de la tecnología pone a disposición de los usuarios una gran variedad de dispositivos móviles con diferentes sistemas operativos; pues el principal sistema operativo a nivel mundial es Android y por otro lado es gratuito; razón por la cual el número de usuarios está creciendo exponencialmente.

Haciendo un enfoque al sistema que se desarrollará en la presente investigación, es importante mencionar que previo al desarrollo tanto del Web Service como de la App Android, se realizará un levantamiento de información en base a un análisis del tráfico de datos apegado a la realidad del GAD Municipal de Chordeleg y el cantón en general, el cual permitirá calcular y dimensionar el ancho de banda total requerido, para garantizar la estabilidad y escalabilidad del sistema, sobre todo en diferentes escenarios como consultas masivas de datos que tenderán a saturar el sistema en diferentes horas del día. También se priorizará el tráfico de la red implementando calidad de servicio (QoS), con el fin de optimizar los tiempos de respuesta del Web Service y sobre todo de las consultas que se realicen a la base de datos.

OBJETIVOS

Objetivo General

Investigar y estudiar los mecanismos y métodos eficaces con enfoque en el análisis de tráfico de datos, para diseñar y dimensionar un sistema escalable y efectivo, que garantice un ancho de banda adecuado, para implementar un Web Service en un servidor dedicado del GAD Municipal de Chordeleg, con el objeto de proveer a los contribuyentes en general, de una App Android, para consultar los principales tributos municipales que recauda la institución, tales como: predio urbano, predio rústico, agua potable y patente municipal.

Objetivos Específicos

1. Dotar de un Web Service al GAD Municipal de Chordeleg, y desarrollar una App Android de consulta de los principales Tributos Municipales como: predio urbano, predio rústico, agua potable y patente municipal.
2. Levantamiento de información de la Red LAN actual del GAD Municipal de Chordeleg, para analizar e investigar los requerimientos necesarios que permitan desplegar un Web Service y una App Android.
3. Obtener, analizar y proyectar el número de usuarios que usarán la App Android y por ende el Web Service, basándose en la realidad del Cantón Chordeleg y en el continuo crecimiento anual de los títulos generados por los principales tributos municipales del GAD Municipal de Chordeleg.
4. Calcular, analizar y determinar la cantidad de datos totales requeridos, por cada consulta de los principales tributos municipales como: predio urbano, predio rústico, agua potable y patente municipal.
5. Estudiar y analizar los diferentes escenarios de saturación del sistema, en base al número de usuarios, horas pico, consultas y el ancho de banda disponible en el GAD Municipal de Chordeleg.

6. Calcular, analizar y determinar el ancho de banda total requerido por el Web Service, para garantizar un tiempo de respuesta adecuado a las diferentes consultas generadas por los usuarios a través de la App Android.
7. Diseñar e implementar un modelo de Calidad de Servicio (QoS) en el GAD Municipal de Chordeleg, para priorizar el tráfico y mejorar el tiempo de respuesta del Web Service y las consultas a la Base de Datos, a más de mejorar el rendimiento de la red.
8. Realizar pruebas de consulta desde la App Android instalada en un dispositivo móvil hacia el Web Service, para analizar en tiempo real el tráfico de datos en el servidor de producción.
9. Generar un script que permita simular los diferentes escenarios de saturación del sistema, para analizar en tiempo real el tráfico de datos del Web Service, conjuntamente con su estabilidad y robustez frente a estas situaciones.
10. Calcular, evaluar y comparar los diferentes costos que generarían las principales operadoras de nuestro país como Movistar, Claro y CNT, por el consumo de datos de la App en un dispositivo móvil.

CAPITULO 1

1 Levantamiento de Información del GAD Municipal de Chordeleg

1.1 Red de Área Local (LAN)

El GAD Municipal de Chordeleg viene funcionando como Cantón desde el año 1992 en un edificio de 3 plantas ubicado en la calle 23 de Enero 4-21 y Carlos Serrano del cantón Chordeleg, con el pasar del tiempo la demanda y crecimiento del cantón en general ha obligado a extender su personal con el fin de prestar un mejor servicio a la ciudadanía, sin embargo por el mismo hecho de ser un cantón pequeño y en crecimiento, aún existen departamentos que trabajan con un solo funcionario, tal es el caso del Departamento de Sistemas que apenas lleva 7 años de haberse creado en la municipalidad y existe un solo técnico a cargo; actualmente cuenta con un personal de 30 funcionarios con un horario de atención desde las 08:00 hasta las 17:00, quienes hacen uso de 30 equipos de cómputo, 5 impresoras en red, 2 lectores biométricos y 3 servidores de datos, distribuidos en 13 departamentos que se encuentran ubicados en las diferentes plantas del edificio, lo cual daría un total de 40 clientes, a más de ello se tiene una red inalámbrica para clientes ocasionales en la sala de concejales; en la siguiente tabla se detalla los clientes que operan por cada departamento en la red LAN del GAD Municipal de Chordeleg:

Tabla 1. Clientes de la Red LAN del GAD Municipal de Chordeleg

Planta	Departamento	Cliente	# Clientes
Baja	Tesorería	Tesorería Municipal	2
		Recaudación	
	Pasillo	Lector Biométrico	1
	Compras Públicas	Analista de Compras Públicas	1
	Turismo	Planificación Estratégica	3
		Analista de Turismo	
		Analista de Comunicación Social	
1	Alcaldía	Alcalde	1
	Secretaría	Secretaria del Concejo	3
		Asistente de Secretaría	
		Impresora en red	

	Talento Humano	Especialista de Talento Humano	4
		Asistente de Talento Humano	
		Analista de Determinación y Rentas	
		Impresora en red	
	Pasillo	Lector Biométrico	1
	Sistemas	Analista de Sistemas Informáticos	1
	Servidores	Servidor SIMM	3
		Servidor SIGAME	
		Servidor SIM	
	Financiero	Dirección Financiera	4
		Analista de Contabilidad	
		Asistente de Contabilidad	
		Impresora en red	
2	Avalúos y Catastros	Analista de Avalúos y Catastros	4
		Analista de Planificación Urbana	
		Analista de Diseño Arquitectónico	
		Analista de Gestión de Territorio	
	Hábitat y Ordenamiento Territorial	Dirección Hábitat y O.T.	4
		Secretaria Hábitat y O.T.	
		Analista de Fiscalización	
		Impresora en red	
	Movilidad, Energía y Conectividad	Dirección MEC	4
		Secretaria MEC	
		Topografía	
		Dirección JASMAM	
	Auditoría	Auditora Interna	1
	Jurídico	Procurador Síndico	3
		Asistente de Procuraduría Síndica	
		Impresora en red	
	Concejales	Ocasionales	0 a 13

Fuente: Autor

Por otra parte al ser un cantón turístico, el GAD Municipal de Chordeleg brinda un servicio de internet gratuito al Parque Central ubicado frente al edificio del GAD Municipal de Chordeleg, el cual está habilitado de lunes a viernes a partir desde las 17:00 hasta las 00:00, y el sábado y domingo todo el día. En esta red se encuentra habilitado un servidor HOSPOT para dar el servicio de internet durante 60 minutos diarios a 50 clientes.

En la siguiente sección se entra en detalle para conocer la configuración de cada equipo, incluido los clientes que intervienen dentro de la red LAN del GAD Municipal de Chordeleg.

1.1.1 Equipos que permiten el despliegue de la red LAN

De acuerdo a la información levantada, el GAD Municipal de Chordeleg, no ha tenido mayor crecimiento tecnológico desde sus inicios, sobre todo en la infraestructura de su red interna, siendo así que hasta el año 2015 venía operando en su red de core con un router casero modelo D-Link Dir 600 y en su red de distribución con otro router del mismo modelo, en ninguna etapa de la red existía un equipo administrable, sin embargo a inicios del año 2016 se adquieren en parte equipos administrables con el fin de mejorar el rendimiento de la red. A continuación se describe cada uno de los equipos que intervienen en la red LAN conjuntamente con su configuración.

1.1.1.1 Red de Core

El proveedor del servicio de internet del GAD Municipal de Chordeleg es la empresa pública CNT (Corporación Nacional de Telecomunicaciones), razón por la cual en la red de core se encuentran tanto los equipos prestados por la CNT como el router principal del GAD Municipal de Chordeleg. La CNT entrega una hoja técnica con los datos de la red a la cual tiene que conectarse el router principal del GAD Municipal de Chordeleg, los mismos que son:

- Red: 181.211.253.96 /29
- Broadcast: 181.211.253.103
- Rango de IP´s disponibles: 181.211.253.97 - 181.211.253.102
- Gateway: 181.211.253.97
- DNS´s: 208.67.220.220, 208.67.222.222
- Tipo: IP pública - Clase B

Por lo tanto, el router principal del GAD Municipal de Chordeleg para conectarse a la red de la CNT, tiene configurado en el puerto que conecta al router de la CNT, lo siguiente:

- Dirección IP: 181.211.253.98
- Máscara: 255.255.255.248
- Gateway: 181.211.253.97

Dentro del mismo router principal del GAD Municipal de Chordeleg, se tiene dos puertos adicionales configurados para enlazar punto a punto con la red del parque central y la red LAN del GAD Municipal de Chordeleg. Los puertos están configurados de la siguiente manera:

- Puerto para la red inalámbrica del Parque Central del Cantón Chordeleg
  - Red: 10.10.1.0 /30
  - Broadcast: 10.10.1.3
  - Rango de IP´s disponibles: 10.10.1.1 – 10.10.1.2
  - Gateway: 10.10.1.1
  - Tipo: IP privada - Clase A
- Puerto para la red LAN del GAD Municipal de Chordeleg
  - Red: 10.10.2.0 /30
  - Broadcast: 10.10.2.3
  - Rango de IP´s disponibles: 10.10.2.1 – 10.10.2.2
  - Gateway: 10.10.2.1
  - Tipo: IP privada - Clase A

En la siguiente figura se puede observar un esquema general de la red de core del GAD Municipal de Chordeleg:

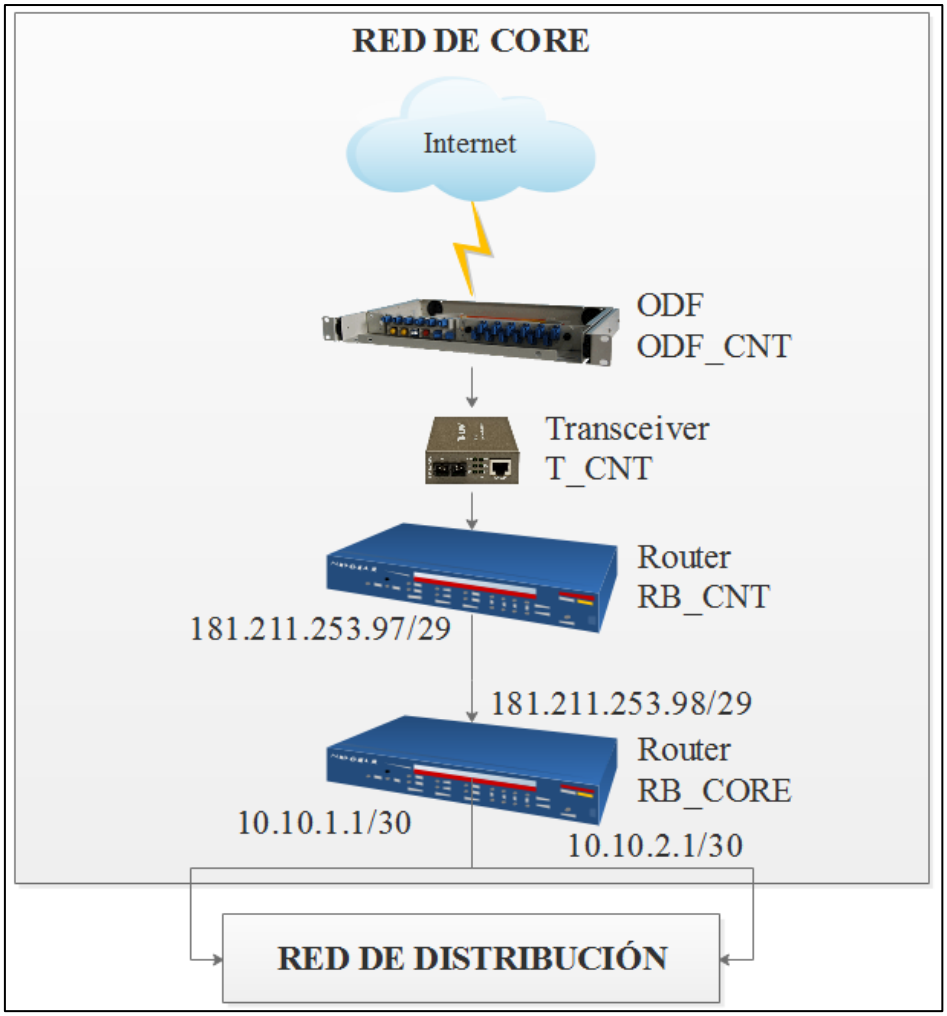


Figura 1. Red de Core del GAD Municipal de Chordeleg
Fuente: Autor

En la siguiente tabla se detalla cada uno de los equipos que pertenecen a la Red de Core del GAD Municipal de Chordeleg:

Tabla 2. Equipos de la Red de Core del GAD Municipal de Chordeleg

Equipo	Etiqueta	Marca	Modelo	Ubicación	Propietario
ODF	ODF_CNT	Cisco	532	Dep-Sistemas	CNT
Transceiver	T_CNT	TP-Link	MC110L	Dep-Sistemas	CNT
Router	RB_CNT	Cisco	Serie 803	Dep-Sistemas	CNT
Router	RB_CORE	Mikrotik	CCR1036-12G-4S-EM	Dep-Sistemas	Municipio

Fuente: Autor

1.1.1.2 Red de Distribución

En esta etapa se encuentran dos router´s, una para la administración de la red del Parque Central y otro para la red LAN del GAD Municipal de Chordeleg, los cuales están enlazados punto a punto al router principal (RB_CORE) que se encuentra en la red de core, la configuración de los puertos de cada router que se conecta punto a punto con el router principal, son las siguientes:

- Puerto del router que enlaza punto a punto la red del Parque Central con el router de Core:
  - Dirección ip: 10.10.1.2
  - Máscara: 255.255.255.252
  - Gateway: 10.10.1.1
- Puerto del router que enlaza punto a punto la red LAN del GAD Municipal de Chordeleg con el router de Core:
  - Dirección ip: 10.10.2.2
  - Máscara: 255.255.255.252
  - Gateway: 10.10.2.1

El router del Parque Central, tiene configurada la siguiente red para los clientes del parque:

- Red: 192.168.1.0/24
- Gateway: 192.168.1.1
- Broadcast: 192.168.1.255
- Rango de IP´s disponibles: 192.168.1.1 – 192.168.1.254
- Tipo: IP privada - Clase B

De las cuales el rango de ip´s es limitado en el router a 50 clientes.

El router de la red LAN del GAD Municipal de Chordeleg, parte de la siguiente dirección ip física:

- Red: 192.168.2.0/24
- Gateway: 192.168.2.1
- Broadcast: 192.168.2.255
- Rango de IP´s disponibles: 192.168.2.1 – 192.168.2.254
- Tipo: IP privada - Clase B

La cual es dividida, para asignar una subred lógica a cada departamento; revisando la tabla 1, se observa un total de 17 departamentos y por ende se tendría 17 subredes, sin embargo los departamentos que tienen un solo cliente a excepción del departamento de sistemas, se incluyen en los departamentos más cercanos para no desperdiciar subredes, es decir, el cliente "Compras Públicas" y el "Lector Biométrico" están incluidos en el Departamento de Tesorería, el cliente "Alcalde" está incluido en el Departamento de Secretaría, el cliente "Lector Biométrico" está incluido en el Departamento de Talento Humano y el cliente "Auditoría Interna" está incluido en el Departamento Jurídico, por lo tanto se eliminan 5 departamentos quedando un total de 12, en la siguiente tabla se detalla la configuración de cada subred por departamento:

Tabla 3. Subredes de la Red LAN del GAD Municipal de Chordeleg

Departamento	SubRed	Broadcast	Ip´s disponibles	Gateway
Tesorería / Compras Públicas	192.168.2.8/29	192.168.2.15	192.168.2.9 - 192.168.2.14	192.168.2.9
Turismo	192.168.2.16/29	192.168.2.23	192.168.2.17 - 192.168.2.22	192.168.2.17
Alcaldía / Secretaría	192.168.2.24/29	192.168.2.31	192.168.2.25 - 192.168.2.30	192.168.2.25
Talento Humano	192.168.2.32/29	192.168.2.39	192.168.2.33 - 192.1682.38	192.168.2.33
Sistemas	192.168.2.40/29	192.168.2.47	192.168.2.41 - 192.168.2.46	192.168.2.41

Servidores	192.168.2.48/29	192.168.2.55	192.168.2.49 - 192.168.2.54	192.168.2.49
Financiero	192.168.2.56/29	192.168.2.63	192.168.2.57 - 192.168.2.62	192.168.2.57
Avalúos y Catastros	192.168.2.64/29	192.168.2.71	192.168.2.65 - 192.168.2.70	192.168.2.65
Hábitat y Ordenamiento Territorial	192.168.2.72/29	192.168.2.79	192.168.2.73 - 192.168.2.78	192.168.2.73
Movilidad, Energía y Conectividad	192.168.2.80/29	192.168.2.87	192.168.2.81 - 192.168.2.88	192.168.2.81
Jurídico	192.168.2.88/29	192.168.2.95	192.168.2.89 - 192.168.2.94	192.168.2.89
Concejales	192.168.2.96/28	192.168.2.111	192.168.2.97 - 192.168.2.110	192.168.2.97

Fuente: Autor

En la siguiente figura se puede apreciar un esquema general de la red de distribución del GAD Municipal de Chordeleg:

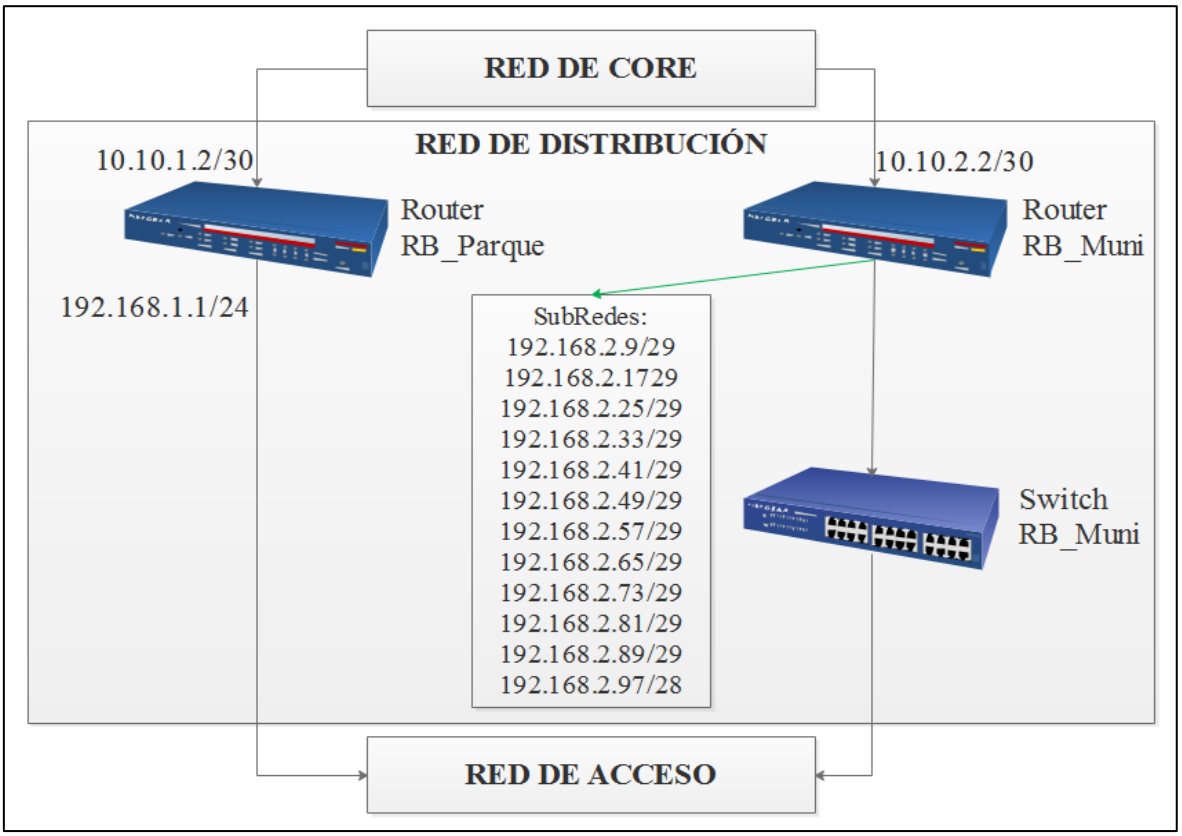


Figura 2. Red de Distribución del GAD Municipal de Chordeleg
Fuente: Autor

Todos los equipos que se encuentran dentro de la red de distribución del GAD Municipal de Chordeleg son de propiedad del Municipio, en la siguiente tabla se detalla cada uno de ellos:

Tabla 4. Equipos de la Red de Distribución del GAD Municipal de Chordeleg

Equipo	Etiqueta	Marca	Modelo	Ubicación
Router	RB_Parque	Mikrotik	RB2011UiAS-RM	Dep-Sistemas
Router	RB_Municipio	Mikrotik	RB2011UiAS-RM	Dep-Sistemas
Switch	SW_Municipio	D-Link	DES-1024D	Dep-Sistemas

Fuente: Autor

1.1.1.3 Red de Acceso

En la red de acceso se tiene un Access Point, conectado al router de la red del Parque Central que se encuentra en la red de distribución, el cual permite de manera inalámbrica conectar a la red los 50 clientes ocasionales del Parque Central.

En cuanto a la red LAN del GAD Municipal de Chordeleg, se tiene conectado al switch del GAD Municipal de Chordeleg de la red de distribución, un total de 10 Switch´s y un Access Point, distribuidos en 12 departamentos ubicados en las diferentes plantas del edificio, los cuales permiten la conexión de los 40 clientes estáticos y 13 ocasionales que se detallaron en la tabla 1. En la siguiente tabla se tiene un detalle de las ip´s asignadas a cada cliente de acuerdo al departamento y a la subred que pertenecen, y el rango de ip´s disponibles para la red inalámbrica de la sala de concejales:

Tabla 5. Ip´s de cada cliente de la Red LAN del GAD Municipal de Chordeleg

Departamento	SubRed	Gateway	Ip_Cliente	Cliente
Tesorería / Compras Públicas	192.168.2.8/29	192.168.2.9	192.168.2.10	Tesorera
			192.168.2.11	Recaudadora
			192.168.2.12	ComprasP
			192.168.2.13	LectorFacial
Turismo	192.168.2.16/29	192.168.2.17	192.168.2.18	PlanificadorE
			192.168.2.19	Turismo
			192.168.2.20	ComunicadoraS
Alcaldía / Secretaría	192.168.2.24/29	192.168.2.25	192.168.2.26	Alcalde
			192.168.2.27	SecreConcejo

			192.168.2.28	AsisSecre
			192.168.2.29	ImpreRed
Talento Humano	192.168.2.32/29	192.168.2.33	192.168.2.34	EspecialistaTH
			192.168.2.35	AsistenteTH
			192.168.2.36	Rentas
			192.168.2.37	ImpreRed
			192.168.2.38	LectorHuella
Sistemas	192.168.2.40/29	192.168.2.41	192.168.2.42	Sistemas
Servidores	192.168.2.48/29	192.168.2.49	192.168.2.50	ServerSIMM
			192.168.2.51	ServerSIGAME
			192.168.2.52	ServerSIM
Financiero	192.168.2.56/29	192.168.2.57	192.168.2.58	DirFinanciera
			192.168.2.59	Contabilidad
			192.168.2.60	AsisContabilidad
			192.168.2.61	ImpreRed
Avalúos y Catastros	192.168.2.64/29	192.168.2.65	192.168.2.66	AvalúosC
			192.168.2.67	DiseñoArq
			192.168.2.68	Proyectos
			192.168.2.69	PlanificaciónU
Hábitat y Ordenamiento Territorial	192.168.2.72/29	192.168.2.73	192.168.2.74	DirHábitat
			192.168.2.75	SecreHábitat
			192.168.2.76	Fiscalización
			192.168.2.77	ImpreRed
Movilidad, Energía y Conectividad	192.168.2.80/29	192.168.2.81	192.168.2.82	DirMovilidad
			192.168.2.83	SecreMovilidad
			192.168.2.84	Topografía
			192.168.2.85	DirAgua
Jurídico / Auditoría Interna	192.168.2.88/29	192.168.2.89	192.168.2.90	Auditora
			192.168.2.91	Síndico
			192.168.2.92	AsisSíndico
			192.168.2.93	ImpreRed
Concejales	192.168.2.96/28	192.168.2.97	192.168.2.98 - 192.168.2.110	Ocasionales

Fuente: Autor

En la siguiente figura se presenta un esquema general de la red de acceso del GAD Municipal de Chordeleg:

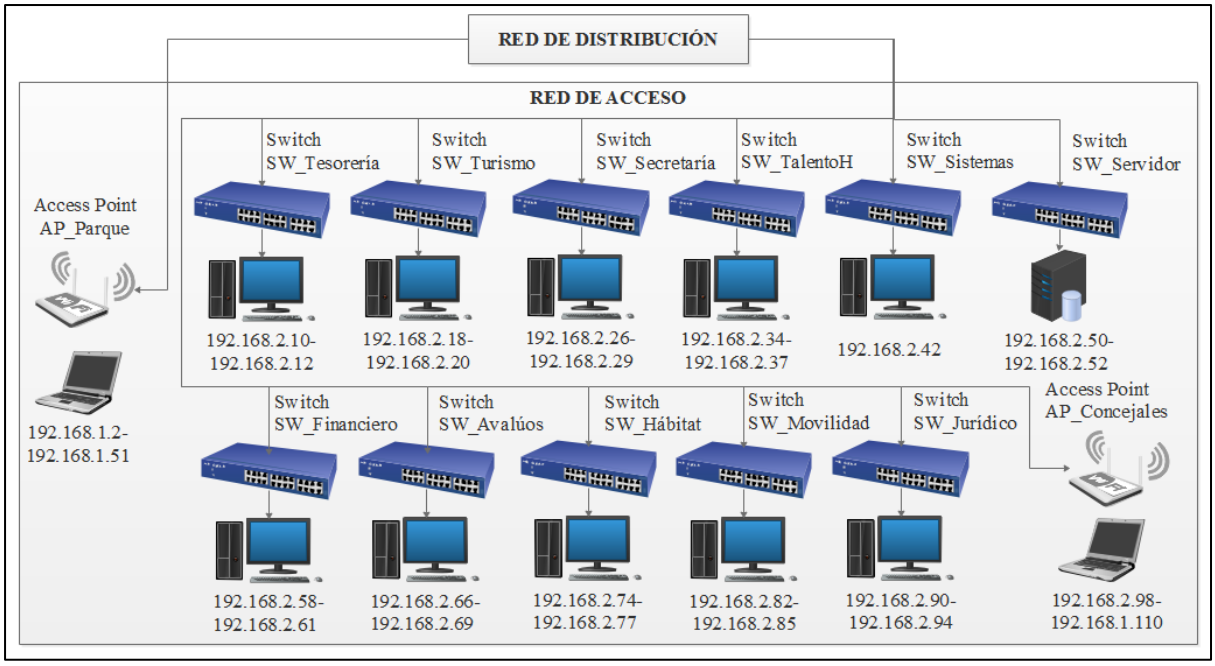


Figura 3. Red de Acceso del GAD Municipal de Chordeleg
Fuente: Autor

Todos los equipos que se encuentran dentro de la red de acceso del GAD Municipal de Chordeleg son de propiedad del Municipio, en la siguiente tabla se detalla cada uno de ellos, excepto los equipos clientes:

Tabla 6. Equipos de la Red de Acceso del GAD Municipal de Chordeleg

Equipo	Etiqueta	Marca	Modelo	Ubicación
Access Point	AP_Parque	D-Link	DWL-2100AP	Dep-Alcaldía
Switch	SW_Tesorería	D-Link	DES-1024D	Dep-Tesorería
Switch	SW_Turismo	D-Link	DES-1024D	Dep-Turismo
Switch	SW_Secretaría	D-Link	DES-1024D	Dep-Secretaría
Switch	SW_TalentoH	D-Link	DES-1024D	Dep-TalentoH
Switch	SW_Sistemas	D-Link	DES-1024D	Dep-Sistemas
Switch	SW_Servidores	D-Link	DES-1024D	Dep-Servidores
Switch	SW_Financiero	D-Link	DES-1024D	Dep-Financiero
Switch	SW_Avalúos	D-Link	DES-1024D	Dep-Avalúos
Switch	SW_Hábitat	D-Link	DES-1024D	Dep-Hábitat
Switch	SW_Movilidad	D-Link	DES-1024D	Dep-Movilidad
Switch	SW_Jurídico	D-Link	DES-1024D	Dep-Jurídico
Access Point	AP_Concejales	D-Link	DWL-2100AP	Dep-Concejales

Fuente: Autor

1.1.2 Rango de ip´s públicas disponibles

Anteriormente se mencionó en la sección de la red de core, que el GAD Municipal de Chordeleg tiene un contrato con la CNT para el servicio de internet, quienes a su vez otorgan 6 ip´s públicas al GAD Municipal de Chordeleg, a continuación se detalla cada una de ellas:

Tabla 7. Ip´s públicas del GAD Municipal de Chordeleg

Red	Broadcast	Ip	Tipo	Proveedor
181.211253.96/29	181.211.253.103	181.211.253.97	Pública	CNT
		181.211.253.98	Pública	CNT
		181.211.253.99	Pública	CNT
		181.211.253.100	Pública	CNT
		181.211.253.101	Pública	CNT
		181.211.253.102	Pública	CNT

Fuente: Autor

Donde, la dirección IP 181.211.253.97 está asignada al puerto del router de la CNT como Gateway y la dirección IP 181.211.253.98 está asignada al puerto del router principal de la red del GAD Municipal de Chordeleg, diríjase a la figura 1 para ver el esquema general de conexión entre el router de la CNT y el router principal del GAD Municipal de Chordeleg.

1.1.3 Ancho de banda contratado

El GAD Municipal de Chordeleg, tiene un contrato con la CNT para dotar del servicio de internet con las siguientes características:

- Plan corporativo de 10Mbps.
- Simétrico, velocidad de descarga igual a la velocidad de subida.
- Compartimiento 1:1.

1.1.4 Servicios por internet

Actualmente el GAD Municipal de Chordeleg cuenta únicamente con una página web, la misma que lleva el dominio http://www.chordeleg.gob.ec, que está alojada en un servidor externo, en la que se puede consultar toda la información gubernamental y lo más destacado del turismo del cantón Chordeleg. A más de la página web que se encuentra en un servidor

externo, el GAD Municipal de Chordeleg en su red LAN no cuenta con ningún otro tipo de servicio web que permita acceder a la ciudadanía del cantón en general desde internet y consultar diversos servicios que ofrece la municipalidad; por este motivo se ha propuesto desarrollar el presente proyecto para implementar un Web Service en un servidor local dentro de la red LAN del GAD Municipal de Chordeleg, con el fin de permitir a la ciudadanía y cantón en general, consultar sus principales tributos municipales, tales como el predio urbano, predio rústico, agua potable y patente municipal, a través de una aplicación móvil desarrollada en la plataforma con más demanda en la actualidad, la misma que es Android.

1.1.5 Servicios locales

La red LAN del GAD Municipal de Chordeleg, actualmente cuenta con los siguientes servicios locales:

- **Navegación por internet**, el router principal que se encuentra en la red de core del GAD Municipal de Chordeleg, provee el servicio de internet a través del puerto 80 a todos los clientes de la red LAN del GAD Municipal de Chordeleg.
- **Base de Datos MySQL**, en la red LAN del GAD Municipal de Chordeleg existe un servidor trabajando bajo la distribución Linux - Ubuntu Server que provee el servicio de base de datos MySQL en el puerto 3306, a un software llamado SIMM, el cual permite la administración del Predio Urbano, Predio Rústico, Agua Potable, Patente, Mejoras, Mercado, Multas, Permisos, Alquileres y Recaudación; los clientes que hacen uso del software SIMM y a la vez interactúan con la base de datos MySQL, son los siguientes: Tesorera, Recaudadora, Rentas y Avalúos.
- **Base de Datos Postgres**, en la red LAN del GAD Municipal de Chordeleg existe un servidor trabajando bajo la distribución Linux – CentOS, que provee el servicio de base de datos Postgres en el puerto 5432, a un software llamado SIM, el cual ha sido implementado en el año 2016 de manera temporal, en base a un proyecto de

actualización catastral, este software administra únicamente el Predio Urbano desde el año 2016; los clientes de la red LAN que hacen uso del software mencionado y a la vez interactúan con la base de datos Postgres, son los siguientes: Tesorera, Recaudadora, Rentas y Avalúos.

- **Base de Datos SQL Server**, en la red LAN del GAD Municipal de Chordeleg existe un servidor que trabaja bajo el sistema operativo Windows Server, que provee el servicio de base de datos SQL Server en el puerto 1433, a un software llamado SIGAME, el cual permite la administración del área contable; los clientes que hacen uso del software SIGAME y a la vez interactúan con la base de datos SQL Server, son los siguientes: Contadora, Directora Financiera y Tesorera.

1.1.6 Calidad de servicio

En la red LAN del GAD Municipal de Chordeleg, actualmente no tiene implemtado ningún tipo de calidad de servicio; recordando lo mencionado en la sección de equipos que permiten el despliegue de la red LAN, apenas a inicios del año 2016 se realizó la adquisición de tres equipos administrables, uno de ellos está operando en la red de core como nodo principal, por lo tanto en el presente proyecto se aplicará calidad de servicio en el router mencionado, para mejorar el rendimiento de la red y sobre todo obtener mejores resultados del web service que se implementará más adelante.

1.1.6.1 Tipo de router

En la sección 1.1.1.1 de la red de core, se detalla en la tabla 2 que el router principal es de marca Mikrotik y modelo CCR1036-12G-4S-EM, el cual es de tipo administrable.

Figura 4. Router Mikrotik CCR1036-12G-4S-EM
Fuente: https://routerboard.com/CCR1036-12G-4S

La implementación de equipos mikrotik ha crecido significativamente en los últimos años dentro de nuestro país, tal es el caso que se ofertan continuamente cursos a nivel nacional sobre el manejo y administración de estos equipos, por defecto tienen instalado un sistema operativo basado en linux, llamado RouterOS, que permite una administración vía terminal o de manera gráfica con una aplicación llamada WinBox, otra alternativa y de gran importancia que ha implemtado mikrotik en sus últimas versiones, es la administración gráfica vía web.

Figura 5. Ingreso a RouterOS vía terminal
Fuente: Autor

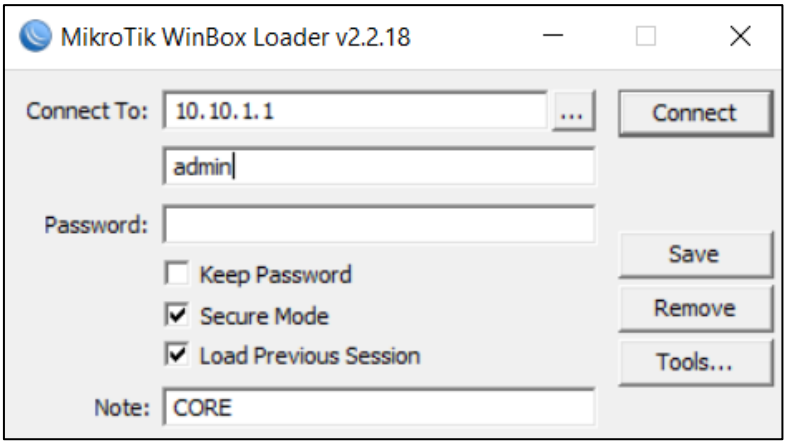


Figura 6. Ingreso a RouterOS vía software WinBox
Fuente: Autor

Figura 7. Ingreso a RouterOS vía Web
Fuente: Autor

1.1.6.2 Manejo y administración del router principal

Para el manejo y administración del router principal, el GAD Municipal de Chordeleg utiliza la administración gráfica a través del software WinBox en su versión más reciente la v2.2.18. Se realizó una prueba de ingreso a la administración del router en el departamento de sistemas del GAD Municipal de Chordeleg con el usuario y password correspondientes, a continuación se presenta una captura de pantalla:

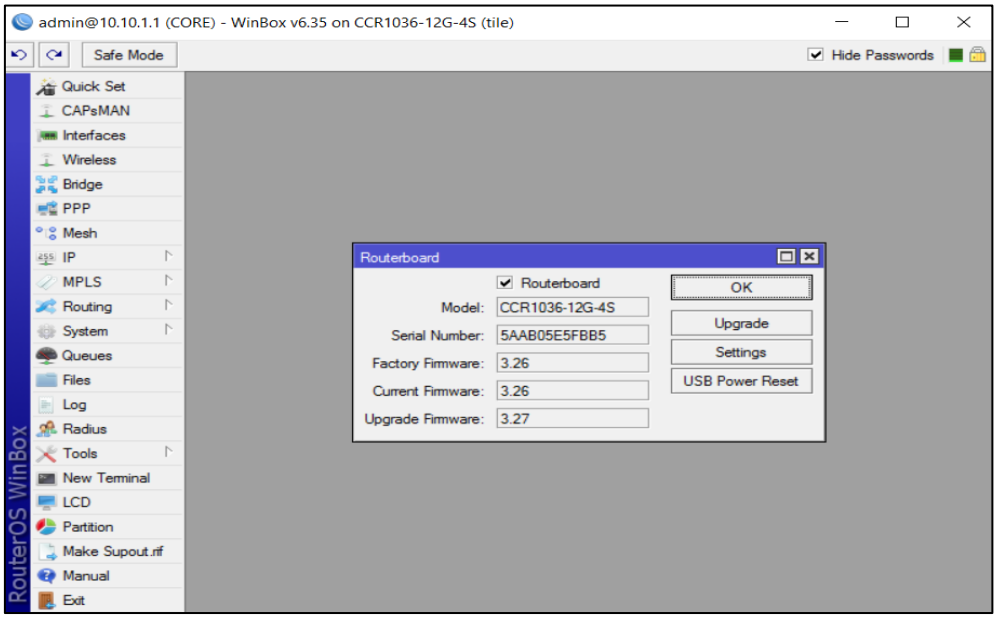


Figura 8. Pantalla de ingreso a RouterOS vía software WinBox v2.2.18
Fuente: Autor

1.1.7 Control de ancho de banda

El GAD Municipal de Chordeleg, realiza el control de ancho de banda de toda su red a través del router principal que se encuentra en la red de core, donde se limita a 2Mbps por cada subred, a excepción de las subredes correspondientes al departamento de sistemas y servidores que tienen asignado un ancho de banda de 5Mbps. A continuación se detalla en una tabla, el control de ancho asignado a cada subred de la red LAN del GAD Municipal de Chordeleg:

Tabla 8. Ancho de banda asignado a cada subred del GAD Municipal de Chordeleg

Departamento	SubRed	Ancho de Banda	Clientes
Tesorería / Compras Públicas	192.168.2.8/29	2Mbps	Tesorera
			Recaudadora
			ComprasP
			LectorFacial
Turismo	192.168.2.16/29	2Mbps	PlanificadorE
			Turismo
			ComunicadoraS
Alcaldía / Secretaría	192.168.2.24/29	2Mbps	Alcalde
			SecreConcejo
			AsisSecre
			ImpreRed
Talento Humano	192.168.2.32	2Mbps	EspecialistaTH
			AsistenteTH
			Rentas
			LectorHuella
			ImpreRed
Sistemas	192.168.2.40/29	5Mbps	Sistemas
Servidores	192.168.2.48/29	5Mbps	ServerSIMM
			ServerSIGAME
			ServerSIM
Financiero	192.168.2.56/29	2Mbps	DirFinanciera
			Contabilidad
			AsisContabilidad
			ImpreRed
Avalúos y Catastros	192.168.2.64/29	2Mbps	AvalúosC
			DiseñoArq
			Proyectos
			PlanificaciónU

Hábitat y Ordenamiento Territorial	192.168.2.72/29	2Mbps	DirHábitat
			SecreHábitat
			Fiscalización
			ImpreRed
Movilidad, Energía y Conectividad	192.168.2.80/29	2Mbps	DirMovilidad
			SecreMovilidad
			Topografía
			DirAgua
Jurídico / Auditoría Interna	192.168.2.88/29	2Mbps	Auditora
			Síndico
			AsisSíndico
			ImpreRed
Concejales	192.168.2.96/28	2Mbps	Ocasionales

Fuente: Autor

1.1.8 Número de usuarios internos

En esta sección se debe tomar en cuenta ciertos aspectos antes de obtener el número total de usuarios internos, por lo tanto acorde a lo visto en la tabla 1, donde se tiene un total de 40 clientes estáticos y 13 clientes ocasionales, de tal manera que dentro de la red LAN del GAD Municipal de Chordeleg, se tendrán como mínimo 40 clientes y como máximo 51 clientes.

Por otra parte está la red inalámbrica del Parque Central del Cantón Chordeleg, sin embargo esta red no interfiere en absoluto con el rendimiento de la red LAN del GAD Municipal de Chordeleg, en vista que opera de lunes a viernes desde las 17:00 hasta las 00:00 y los fines de semana todo el día, siendo un horario distinto al de trabajo del GAD Municipal de Chordeleg y por ende al de la red LAN del GAD Municipal de Chordeleg.

1.2 Base de datos de los principales tributos municipales

El GAD Municipal de Chordeleg, hasta el año 2009 venía trabajando con más de un software para la administración de sus principales tributos municipales, como son: el predio urbano, predio rústico, agua potable y patente municipal, razón por la cual disponían de más de una base de datos dentro de la red LAN; sin embargo en el año 2010 implementan el software SIMM (Sistema Integral Municipal Multifinalitario) desarrollado en Visual Basic

6.0, por la empresa PLANERP de la ciudad de Cuenca, su propósito primordial fue el unificar todas las bases de datos existentes, para administrar los tributos municipales a modo de ventanilla única, a través de una sola base de datos consolidada, la cual lleva el mismo nombre que el software, que es SIMM.

La base de datos SIMM fue creada en MySQL 5.1.69, y actualmente está instalada y en funcionamiento, en un servidor bajo la distribución linux “Ubuntu Server 9.0”, el cual es parte de la red LAN del GAD Municipal de Chordeleg; a continuación se presenta un esquema general de la conexión entre los equipos del departamento de tesorería y el servidor que provee la base de datos SIMM.

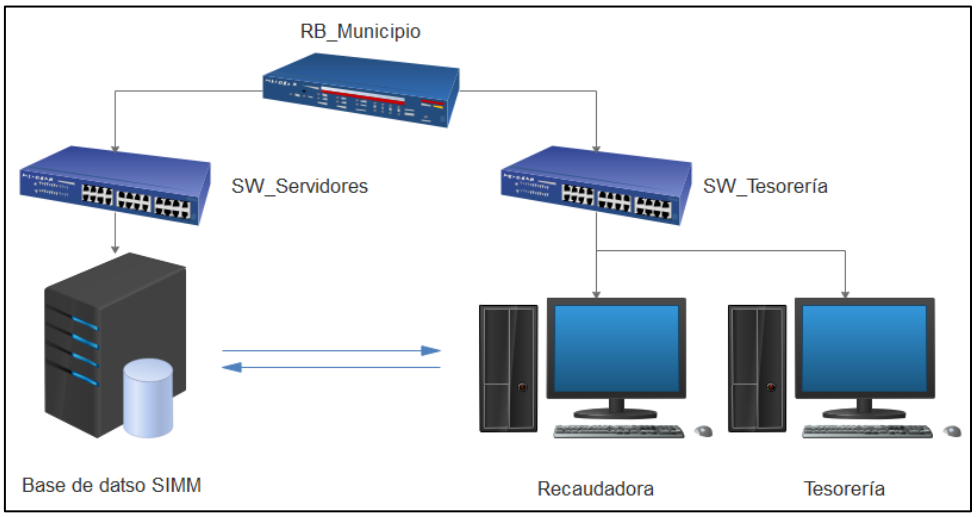


Figura 9. Departamento de tesorería en conexión con la base de datos SIMM
Fuente: Autor

1.2.1 Tipo de base de datos

MySQL es un gestor de base de datos relacional, por lo tanto la base de datos SIMM del GAD Municipal de Chordeleg es de tipo relacional. Una base de datos relacional es una colección de elementos de datos organizados en un conjunto de tablas formalmente descritas desde la que se puede acceder a los datos o volver a montarlos de muchas maneras diferentes sin tener que reorganizar las tablas de la base (Rouse, 2015).

1.2.2 Herramienta para la gestión de la base de datos

En internet se puede encontrar viarias herramientas gratuitas para la administración de una base de datos MySQL, entre ellas está una de las más importantes con la cual trabaja el departamento de sistemas del GAD Municipal de Chordeleg, la misma que se trata de MySQL Workbench del fabricante Oracle. En el desarrollo del presente proyecto se utilizará la misma herramienta, ya que se necesita únicamente generar las consultas de los principales tributos municipales, como son el predio urbano, predio rural, agua potable y patente municipal, y esta herramienta dispone de lo necesario. En la siguiente figura se puede observar la pantalla principal de la herramienta MySQL Workbench conectada a la base de datos SIMM:

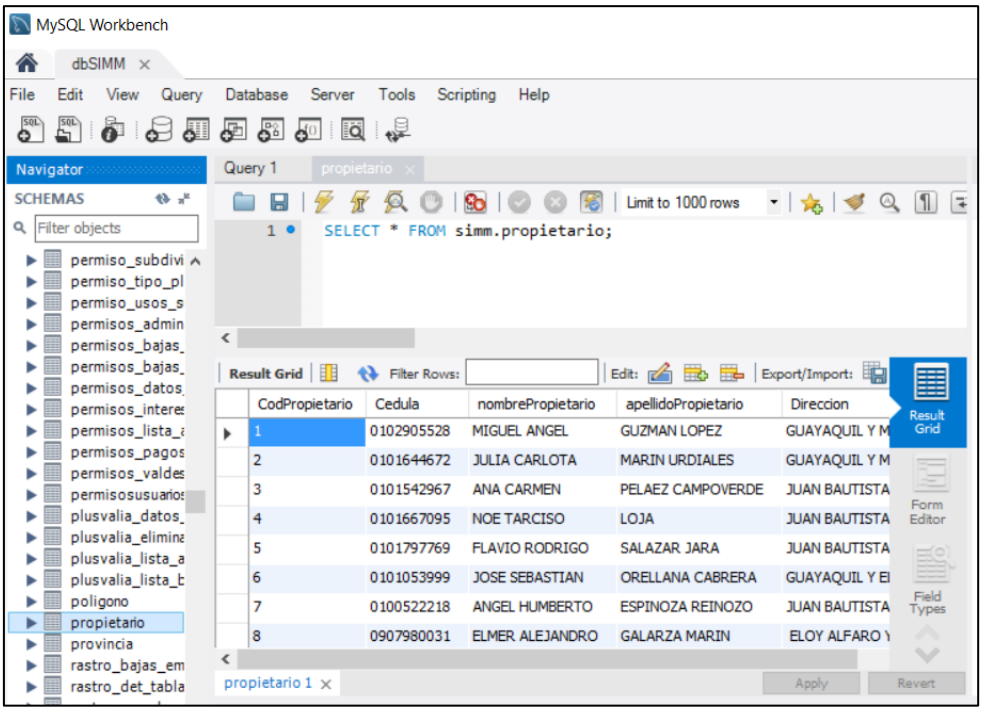


Figura 10. Software MySQL Workbench para gestionar una base de datos MySQL
Fuente: Autor

1.2.3 Estructura de la base de datos

En la estructura de la base de datos SIMM del GAD Municipal de Chordeleg, únicamente se levantará información acerca de las tablas que tengan relación con los títulos de crédito de los principales tributos municipales, por otra parte también se levantará

información acerca de la tabla de contribuyentes e intereses, ya que se relacionan directamente con la información correspondiente a las consultas de los principales tributos municipales, a continuación se detalla cada una de ellas:

- **Clientes:** Se registran únicamente en una tabla llamada "propietario", a continuación se detalla su estructura:

  Tabla 9. Estructura de la tabla propietario de la base de datos SIMM

Field	Type	Null	Key	Default
CodPropietario	int(11)	NO	PRI	0
Cedula	varchar(15)	YES		
nombrePropietario	varchar(60)	YES		
apellidoPropietario	varchar(60)	YES		
Direccion	varchar(60)	YES		
telefono	varchar(12)	YES		
TerceraEdad	int(10) unsigned	NO		0
EsDiscapacitado	int(11)	NO		0
institucionPublica	int(11)	YES		
entidadReligiosaExonerada	int(11)	YES		
porcentajeExoneracion	varchar(5)	YES		

Fuente: Autor

- **Intereses:** Se registran únicamente en una tabla llamada "interes", a continuación se detalla su estructura:

  Tabla 10. Estructura de la tabla intereses de la base de datos SIMM

Field	Type	Null	Key	Default
ano	int(4)	NO	PRI	0
mes	char(3)	NO	PRI	
porcentajeinteres	float	YES		
fecha	int(6)	NO		

Fuente: Autor

- **Predio Urbano:** Se registran en una tabla llamada "scum_parcela", la cual se presenta en la siguiente tabla:

  Tabla 11. Estructura de la tabla scum_parcela de la base de datos SIMM

Campo	Tipo	Null	Key	Default
CodProv	char(2)	NO	PRI	

CodCant	char(2)	NO	PRI	
CodParroq	char(2)	NO	PRI	
CodZona	char(3)	NO	PRI	
CodSect	char(3)	NO	PRI	
CodPolg	char(3)	NO	PRI	
CodParc	char(3)	NO	PRI	
CodPropHoriz	varchar(5)	NO	PRI	
CodPropietario	int(11)	NO		0
Calle	varchar(50)	YES		
Entre	varchar(65)	YES		
AbcisaX	varchar(15)	YES		
OrdenadaY	varchar(15)	YES		
Personeria_Jur	char(2)	YES		
Responsable	varchar(60)	YES		
sector_sensal	varchar(12)	YES		
sector_planeamiento	varchar(12)	YES		
tipo_avaluo	char(2)	YES		0
exonerado	int(11)	YES		0
bloqueado	int(11)	YES		0
observaciones	varchar(255)	YES		
propietario_Anterior	varchar(140)	YES		
Imagen	mediumblob	YES		
FACActual	varchar(45)	YES		
FACAnterior	varchar(45)	YES		
riesgo	varchar(4)	NO		01

Fuente: Autor

En cuanto al registro de los títulos de crédito, se registran en una tabla llamada "scum_liquidacion_avaluo", la cual se puede observar a continuación:

Tabla 12. Estructura de la tabla simm.scum_liquidacion_avaluo

Campo	Tipo	Null	Key	Default
CodProv	char(2)	NO	PRI	
CodCant	char(2)	NO	PRI	
CodParroq	char(2)	NO	PRI	
CodZona	char(3)	NO	PRI	
CodSect	char(3)	NO	PRI	
CodPolg	char(3)	NO	PRI	
CodParc	char(3)	NO	PRI	
CodPropHoriz	varchar(5)	NO	PRI	
CodAvaluo	varchar(12)	NO	PRI	
TipoAvaluo	int(11)	YES		0

TotalAvaluo	float	YES		0
BaseImponible	float	YES		0
TotalImpuestos	float	YES		0
TotalRebajas	float	YES		0
Descuento_Recargo	float	YES		0
Intereses	float	YES		0
exoneracion	float	YES		0
Total_Pagar	float	YES		0
Ano	float	YES		0
Pagado	int(11)	YES		0
fecha	date	YES		
conjunto	int(11)	YES		0
quien_cobro	int(11)	YES		
CodPropietario	int(11)	NO		0
DireccionPredio	varchar(120)	NO		
observaciones	varchar(60)	NO		
areaTerreno	float	NO		0
areaConstruccion	float	NO		0
valorTerreno	float	NO		0
valorConstruccion	float	NO		0
impPredial	float	NO		0
solarNoEdificado	float	NO		0
bomberos	float	NO		0
antiguos	float	NO		0
otros	float	NO		0
computacion	float	NO		0
fecha_convenio	date	YES		2012-08-14
convenio_pago	int(4)	NO		0

Fuente: Autor

- **Predio Rural:** Se registran en una tabla llamada “scr_parcela”, la cual se presenta en la siguiente tabla:

Tabla 13. Estructura de la tabla scr_parcela de la base de datos SIMM

Field	Type	Null	Key	Default
CodProv	char(2)	NO	PRI	
CodCant	char(2)	NO	PRI	
CodParroq	char(2)	NO	PRI	
CodZona	char(3)	NO	PRI	
CodSect	char(3)	NO	PRI	
CodPolg	char(3)	NO	PRI	
CodParc	char(3)	NO	PRI	
CodPropietario	int(11)	NO		0

NombParcela	varchar(100)	YES		
Localizacion	varchar(150)	YES		
AbcisaX	varchar(15)	YES		
OrdenadaY	varchar(15)	YES		
Personeria_Jur	char(2)	YES		
CodParcAntigua	varchar(15)	YES		
exonerado	int(11)	YES		0
bloqueado	int(11)	YES		0
observaciones	varchar(255)	YES		
propietario_anterior	varchar(140)	YES		
Imagen	mediumblob	YES		
FACActual	varchar(45)	YES		
FACAnterior	varchar(45)	YES		
responsable	varchar(60)	NO		
fechaRegistro	date	NO		

Fuente: Autor

En cuanto al registro de los títulos de crédito, se registran en una tabla llamada "scr_liquidacion_avaluo", la cual se puede observar a continuación:

Tabla 14. Estructura de la tabla scr_liquidacion_avaluo de la base de datos SIMM

Field	Type	Null	Key	Default
CodProv	char(2)	NO	PRI	
CodCant	char(2)	NO	PRI	
CodParroq	char(2)	NO	PRI	
CodZona	char(3)	NO	PRI	
CodSect	char(3)	NO	PRI	
CodPolg	char(3)	NO	PRI	
CodParc	char(3)	NO	PRI	
CodParcAntiguo	varchar(10)	YES		
CodAvaluo	varchar(12)	NO	PRI	
TipoAvaluo	int(11)	YES		1
TotalAvaluo	float	YES		0
BaseImponible	float	YES		0
TotalImpuestos	float	YES		0
TotalRebajas	float	YES		0
Descuento_Recargo	float	YES		0
Intereses	float	YES		0
exoneracion	float	YES		0
Total_Pagar	float	YES		0
Ano	float	YES		
pagado	int(11)	YES		0
Fecha	date	YES		

quien_cobro	int(11)	YES		0
CodPropietario	int(11)	NO		0
direccionPredio	varchar(100)	NO		
observaciones	varchar(60)	NO		
areaTerreno	float	NO		0
areaConstruccion	float	NO		0
valorTerreno	float	NO		0
valorConstruccion	float	NO		0
impPredial	float	NO		0
bomberos	float	NO		0
antiguos	float	NO		0
otros	float	NO		0
computacion	float	NO		0
convenio_pago	int(11)	NO		0
fecha_convenio	date	NO		2012-08-14

Fuente: Autor

- **Agua Potable:** Tiene una tabla llamada “agua_medidor” donde se registran todos los títulos de crédito, en la siguiente tabla se puede observar su estructura:

Tabla 15. Estructura de la tabla agua_medidor de la base de datos SIMM

Field	Type	Null	Key	Default
Codmedidor	varchar(20)	NO	PRI	
numMedidor	varchar(15)	NO		
Codprov	char(2)	NO		
CodCant	char(2)	NO		
CodParrq	char(2)	NO		
CodZona	char(3)	NO		
CodSect	char(3)	NO		
CodManzana	char(3)	NO		
estadomed	varchar(20)	YES		
Fechainstalacion	date	YES		0000-00-00
Marca	varchar(15)	YES		
CodParcela	char(3)	NO		
CodHorizontal	varchar(6)	YES		
tipo	int(10) unsigned	NO		0
Direccion	varchar(200)	YES		
MedAnterior	varchar(150)	NO		

Fuente: Autor

En cuanto al registro de los títulos de crédito, se registran en una tabla llamada "agua_factura", la cual se puede observar a continuación:

Tabla 16. Estructura de la tabla agua_factura de la base de datos SIMM

Field	Type	Null	Key	Default
id	bigint(20)	NO	PRI	
codMedidor	varchar(20)	NO		
codPropietario	bigint(20)	NO		
codTarifa	varchar(10)	NO		
id_clase_rubro	int(11)	NO		
codemision	varchar(10)	NO		
nro_factura	varchar(60)	YES		
estado	int(11)	NO		1
pagado	int(11)	NO		0
quien_cobro	int(11)	NO		0
fecha_cobro	datetime	YES		
fecha_factura	date	YES		
lectura_anterior	float	NO		0
lectura_actual	float	NO		0
valor_principal	float	NO		0
total_tarifa	float	NO		0
intereses	float	NO		0
descuentos	float	NO		0
meses_multa	int(11)	NO		0
multas	float	NO		0
iva	float	NO		0
total_pagado	float	NO		0
convenio_pago	int(11)	NO		0
fecha_convenio	date	NO		2012-08-14
usuario_creacion	int(11)	NO		
usuario_modificacion	int(11)	YES		
fecha_creacion	datetime	NO		
fecha_modificacion	datetime	YES		
lectura_digitada	float	NO		0
forma_pago	int(10) unsigned	NO		1
alcantarillado	float	NO		0
basura	float	NO		0

Fuente: Autor

- **Patente Municipal:** Tiene una tabla llamada "patentes_patente" donde se registran todos los títulos de crédito, en la siguiente tabla se puede observar su estructura:

Tabla 17. Estructura de la tabla patentes_patente de la base de datos SIMM

Field	Type	Null	Key	Default
numPatente	int(11)	NO	PRI	
codPropietario	int(11)	NO		0
codActividad	int(11)	NO		0
codDetActividad	int(11)	NO		0
fecha_registro	date	NO		0000-00-00
razon_social	varchar(100)	YES		
capital	float	NO		0
ubicacion	varchar(200)	YES		
entre	varchar(200)	YES		
referencia	varchar(200)	YES		
estado	int(11)	NO		0
CodTipoPatente	int(11)	NO		0
codProv	char(2)	NO		
codCant	char(2)	NO		
codParroq	char(2)	NO		
codZona	char(3)	NO		
codSect	char(3)	NO		
codPolg	char(3)	NO		
observaciones	varchar(255)	YES		
codParc	char(3)	NO		
codPropHoriz	char(3)	NO		
tipo_ubicacion	int(11)	NO		0
tercera_edad	int(11)	NO		0
Lleva_Contabilidad	int(11)	NO		0
activoTotal	double	NO		0

Fuente: Autor

En cuanto al registro de los títulos de crédito, se registran en una tabla llamada "patente_emision_patente", la cual se puede observar a continuación:

Tabla 18. Estructura de la tabla simm.patente_emision_patente

Field	Type	Null	Key	Default
numPatente	int(11)	NO	PRI	0
ano	int(4)	NO	PRI	0
fecha_emision	date	NO		0000-00-00
valor_impuesto	float	NO		0
impuesto_capital	float	NO		0
serv_administrativos	float	NO		0
interes	float	NO		0
otro	float	NO		0

Valor_Especie	float	NO		0
pagado	int(11)	NO		0
quien_cobro	int(11)	NO		0
codTablaPatente	int(11)	NO		0
FechaPago	date	NO		0000-00-00
exoneracion	float	NO		0
observaciones	varchar(90)	NO		
codPropietario	int(11)	NO		0
actividad	varchar(100)	NO		
razon_social	varchar(100)	NO		
capital	float	NO		0
direccion	varchar(255)	NO		
tipoPatente	varchar(100)	NO		
clave	varchar(20)	NO		
activoTotal	float	NO		0
quien_emitio	int(11)	NO		0

Fuente: Autor

1.2.4 Modelo de consulta de los contribuyentes

Los contribuyentes se registran en una sola tabla llamada “propietario”, donde tiene un campo llamado codPropietario que se utiliza como referencia del contribuyente dentro de otras tablas, tales como los títulos de crédito generados a nombre del contribuyente; en el presente proyecto, únicamente se necesita consultar la cédula y el nombre del propietario, filtrándolo por su código de referencia; para ello está la siguiente consulta:

- SELECT codPropietario, cedula, apellidoPropietario, nombrePropietario FROM propietario WHERE codPropietario=’’.

1.2.4.1 Total de contribuyentes

La tabla de registro de contribuyentes es la tabla “propietario”, en la siguiente figura se puede ver la consulta correspondiente para obtener el número total de contribuyentes:

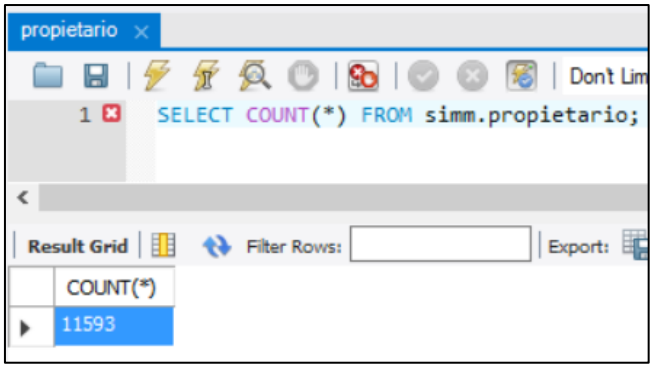


Figura 11. Consulta del total de contribuyentes en la base de datos SIMM
Fuente: Autor

Donde claramente se observa un total de 11593 contribuyentes hasta la presente fecha.

1.2.4.2 Tasa de crecimiento anual

Basados en la consulta de la figura 11, se consulta el total de contribuyentes ingresados durante los últimos 5 años, seguidamente se analiza el crecimiento anual por periodo y se obtiene el promedio de crecimiento anual de los contribuyentes, en la siguiente tabla se puede observar detalladamente cada uno de los datos:

Tabla 19. Detalle de crecimiento anual de contribuyentes

Año	Contribuyentes	Periodo	Tasa de Crecimiento	Total	Promedio
2013	9015	2013-2014	713	2578	644.5
2014	9728				
2014	9728	2014-2015	506		
2015	10234				
2015	10234	2015-2016	571		
2016	10805				
2016	10805	2016-2017	788		
2017	11593				

Fuente: Autor

Donde se puede observar que el promedio de crecimiento anual analizado entre el año 2013 y 2017 es de 644.5 contribuyentes, lo que da un aproximado de 645.

1.2.5 Modelo de consulta del predio urbano

Antes de obtener el modelo final de consulta del predio urbano, se explicará por partes cada consulta. En primer lugar, se debe consultar si el predio buscado existe, para lo cual se tiene que contar el número de filas (COUNT(*)) que resulte de consultar la tabla de registro

de predios (scum_parcela), filtrada por la clave catastral (CodProv, CodCant, CodParroq, CodZona, CodSect, CodPolg, CodParc, CodPropHoriz), también se extraerá el código del contribuyente (CodPropietario) que servirá mas adelante para la consulta de los datos personales, para ello se tiene la siguiente consulta:

- SELECT COUNT(), codPropietario FROM scum_parcela WHERE CodProv='' AND CodCant='' AND CodParroq='' AND CodZona='' AND CodSect='' AND CodPolg='' AND CodParc='' AND CodPropHoriz=''.

Si el resultado es igual a 1 se entiende que el predio existe, caso contrario no; como se menciona anteriormente, se tiene que consultar los datos personales del propietario del predio, esto se realiza ingresando el código de contribuyente de la consulta anterior como filtro en la tabla de contribuyentes (propietario), para obtener la cedula, el apellido y el nombre del contribuyente; para ello se tiene la siguiente consulta:

- SELECT cedula, apellidoPropietario, nombrePropietario FROM propietario WHERE CodPropietario="al CodPropietario de la consulta anterior".

Luego de obtener los datos del contribuyente se procede a consultar si tiene predios pendientes de pago, para ello se consulta la tabla "scum_liquidacion_avaluo", filtrando por la clave del predio de la primera consulta e indicando solo las emisiones pendientes de pago (pagado), en esta consulta se extrae los siguientes campos de interés: fecha de emisión (CodAvaluo), año (Ano), impuesto predial (impPredial), solar no edificado (solarNoEdificado), bomberos (), servicios administrativos (computacion) y descuentos (exoneracion), para lo cual se tiene la siguiente consulta:

- SELECT CodAvaluo, Ano, impPredial, solarNoEdificado, bomberos, exoneracion FROM scum_liquidacion_avaluo WHERE CodProv='' AND CodCant='' AND CodParroq='' AND CodZona='' AND CodSect='' AND CodPolg='' AND CodParc='' AND CodPropHoriz='' AND pagado='0'.

Ahora queda por consultar el porcentaje de intereses de los predios pendientes de pago, siempre y cuando existan, para ello se tiene que consultar la tabla de intereses (intereses), filtrando un rango de fechas según la fecha de emisión del predio que se obtiene de la consulta anterior y la fecha actual en la que se consulta, para lo cual se realiza lo siguiente:

- SELECT sum(porcentajeInteres) FROM intereses WHERE fecha BETWEEN ‘fecha de emisión del predio’ AND ‘fecha actual del predio’.

Finalmente para optimizar los tiempos de respuesta, se combinan todas las consultas anidando y haciendo uso del comando UNION de MySQL, para simplemente remplazar la clave catastral correspondiente, la consulta quedaría de la siguiente manera:

- (SELECT IF(COUNT(*)>0, b.cedula,"N.E.") AS flag_ced_period, CONCAT(b.apellidoPropietario, " ", b.nombrePropietario) AS prop_fecha, 0 as impPredial, 0 as solarNoEdificado, 0 as bomberos, 0 as computacion, 0 as exoneracion, 0 as interés FROM scum_parcela a, propietario b WHERE (a.CodProv='01' AND a.CodCant='11' AND a.CodParroq='50' AND a.CodZona='02' AND a.CodSect='03' AND a.CodPolg='05' AND a.CodParc='012' AND a.CodPropHoriz='001') AND (b.codPropietario=a.codPropietario)) UNION (SELECT (CAST(a.Ano AS char)) AS flag_ced_period, (DATE_FORMAT(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-", SUBSTRING(a.CodAvaluo,5,2)), "%d/%m/%Y")) AS prop_fecha, a.impPredial, a.solarNoEdificado, a.bomberos, a.computacion, a.exoneracion, sum(b.porcentajeinteres) as interés FROM scum_liquidacion_avaluo a, intereses b WHERE (a.CodProv='01' AND a.CodCant='11' AND a.CodParroq='50' AND a.CodZona='02' AND a.CodSect='03' AND a.CodPolg='05' AND a.CodParc='012' AND a.CodPropHoriz='001' AND a.pagado=0) AND (b.fecha BETWEEN IF(

date_format(NOW(), "%y-%m") >= date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1 MONTH, "%y-%m"), date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1 MONTH, "%Y%m"), "100001") AND IF(date_format(NOW(), "%y-%m") >= date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1 MONTH, "%y-%m"), date_format(NOW( ), "%Y%m"), "100001")) GROUP BY a.Ano)

1.2.5.1 Total de títulos emitidos anualmente

La tabla de registro de emisiones del predio urbano, es la tabla "scum_liquidacion_avaluo", en la siguiente figura se puede ver la consulta correspondiente para obtener el número total de predios registrados en el año 2017:

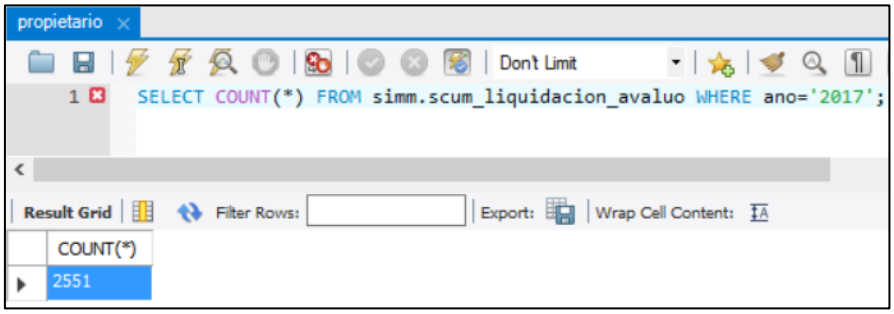


Figura 12. Consulta del total de predios urbanos 2017 en la base de datos SIMM
Fuente: Autor

Donde claramente se observa un total de 2551 predios urbanos emitidos en el año 2017.

1.2.5.2 Tasa de crecimiento anual

De igual manera, haciendo uso de la consulta anterior, se obtiene el total de predios emitidos durante los últimos 5 años, seguidamente se analiza el crecimiento anual por periodo y se obtiene el promedio de crecimiento anual del predio urbano emitido anualmente, en la siguiente tabla se puede apreciar detalladamente cada uno de los datos:

Tabla 20. Detalle de crecimiento anual de la emisión de predios urbanos

Año	Predios Emitidos	Periodo	Tasa de Crecimiento	Total	Promedio
2013	2269	2013-2014	93	282	70.5
2014	2362				
2014	2362	2014-2015	36		
2015	2398				
2015	2398	2015-2016	71		
2016	2469				
2016	2469	2016-2017	82		
2017	2551				

Fuente: Autor

Donde se observa que el promedio de crecimiento anual analizado entre el año 2013 y 2017 es de 70.5 emisiones por predio urbano, lo que da un aproximado de 71.

1.2.6 Modelo de consulta del predio rural

De igual manera que el predio urbano, antes de obtener el modelo final de consulta del predio rural, se explicará por partes cada consulta. En primer lugar, se debe consultar si el predio buscado existe, para lo cual se tiene que contar el número de filas (COUNT(*)) que resulte de consultar la tabla de registro de predios (scr_parcela), filtrada por la clave catastral (CodProv, CodCant, CodParroq, CodZona, CodSect, CodPolg, CodParc), también se extraerá el código del contribuyente (CodPropietario) que servirá más adelante para la consulta de los datos personales, para ello se tiene la siguiente consulta:

- SELECT COUNT(), codPropietario FROM scr_parcela WHERE CodProv=’’ AND CodCant=’’ AND CodParroq=’’ AND CodZona=’’ AND CodSect=’’ AND CodPolg=’’ AND CodParc=’’.

Si el resultado es igual a 1 se entiende que el predio existe, caso contrario no; como se menciona anteriormente, se tiene que consultar los datos personales del propietario del predio, esto se realiza ingresando el código de contribuyente de la consulta anterior como filtro en la tabla de contribuyentes (propietario), para obtener la cedula, el apellido y el nombre del contribuyente, para ello se tiene la siguiente consulta:

- SELECT cedula, apellidoPropietario, nombrePropietario FROM propietario WHERE CodPropietario=”al CodPropietario de la consulta anterior”.

Luego de obtener los datos del contribuyente se procede a consultar si tiene predios pendientes de pago, para ello se debe consultar la tabla “scr_liquidacion_avaluo”, filtrando por la clave del predio de la primera consulta e indicando solo las emisiones pendientes de pago (pagado), en esta consulta se extrae los siguientes campos de interés: fecha de emisión (CodAvaluo), año (Ano), impuesto predial (impPredial), bomberos (), servicios administrativos (computacion) y descuentos (exoneracion), para lo cual se tiene la siguiente consulta:

- SELECT CodAvaluo, Ano, impPredial, bomberos, exoneracion FROM scr_liquidacion_avaluo WHERE CodProv=’’ AND CodCant=’’ AND CodParroq=’’ AND CodZona=’’ AND CodSect=’’ AND CodPolg=’’ AND CodParc=’’ AND pagado=’0’.

Ahora queda por consultar el porcentaje de intereses de los predios pendientes de pago, siempre y cuando existan, para ello se tiene que consultar la tabla de intereses (intereses), filtrando un rango de fechas según la fecha de emisión del predio que se obtiene de la consulta anterior y la fecha actual en la que se consulta, para lo cual se realiza lo siguiente:

- SELECT sum(porcentajeInteres) FROM intereses WHERE fecha BETWEEN ‘fecha de emision del predio’ AND ‘fecha actual del predio’.

Finalmente para optimizar los tiempos de respuesta, se combinan todas las consultas anidando y haciendo uso del comando UNION de MySQL, para simplemente remplazar la clave catastral correspondiente, la consulta quedaría de la siguiente manera:

- (SELECT IF(CONUT(*)>0, b.cedula,"N.E.") AS flag_ced_period, concat(b.apellidoPropietario, " ", b.nombrePropietario) AS prop_fecha, 0 AS impPredial, 0 AS bomberos, 0 AS computacion, 0 AS exoneracion, 0 AS interés FROM scr_parcela a, propietario b WHERE (a.CodProv='01' AND a.CodCant='11' AND a.CodParroq='51' AND a.CodZona='51' AND a.CodSect='03' AND a.CodPolg='23' AND a.CodParc='025') AND (b.codPropietario=a.codPropietario)) UNION (SELECT (CAST(a.Ano AS char)) AS flag_ced_period, (DATE_FORMAT(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-", SUBSTRING(a.CodAvaluo,5,2)), "%d/%m/%Y")) AS prop_fecha, a.impPredial, a.bomberos, a.computacion, a.exoneracion, sum(b.porcentajeinteres) as interes FROM scr_liquidacion_avaluo a, intereses b WHERE (a.CodProv='01' AND a.CodCant='11' AND a.CodParroq='51' AND a.CodZona='51' AND a.CodSect='03' AND a.CodPolg='23' AND a.CodParc='025' AND a.pagado=0) AND (b.fecha BETWEEN IF(date_format(NOW(), "%y-%m") >= date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1 MONTH, "%y-%m"), date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1 MONTH, "%Y%m"), "100001") AND IF(date_format(NOW(), "%y-%m") >= date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1 MONTH, "%y-%m"), date_format(NOW( ), "%Y%m"), "100001")) GROUP BY a.Ano)

1.2.6.1 Total de títulos emitidos anualmente

La tabla de registro de emisiones del predio rural, es la tabla “scr_liquidacion_avaluo”, en la siguiente figura se puede ver la consulta correspondiente para obtener el número total de predios rurales registrados en el año 2017:

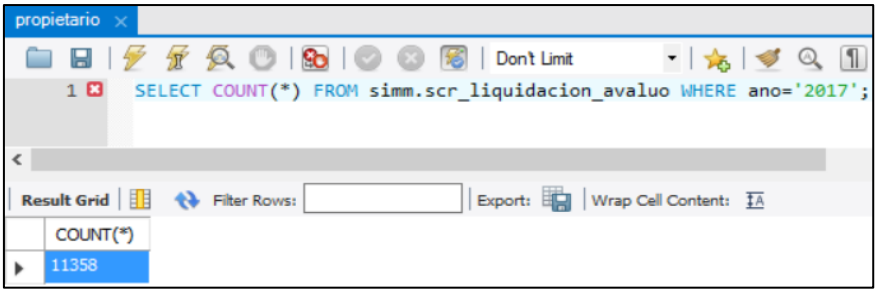


Figura 13. Consulta del total de predios rurales 2017 en la base de datos SIMM
Fuente: Autor

Donde claramente se observa un total de 11358 predios rurales emitidos en el año 2017.

1.2.6.2 Tasa de crecimiento anual

De igual manera que la consulta anterior, se consulta el total de predios emitidos durante los últimos 5 años, seguidamente se analiza el crecimiento anual por periodo y se obtiene el promedio de crecimiento anual de los predios rurales, en la siguiente tabla se puede ver detalladamente cada uno de los datos:

Tabla 21. Detalle de crecimiento anual de la emisión de predios rurales

Año	Predios Emitidos	Periodo	Tasa de Crecimiento	Total	Promedio
2013	10802	2013-2014	183	556	139
2014	10985				
2014	10985	2014-2015	65		
2015	11050				
2015	11050	2015-2016	211		
2016	11261				
2016	11261	2016-2017	97		
2017	11358				

Fuente: Autor

Donde se observa que el promedio de crecimiento anual analizado entre el año 2013 y 2017 es de 139 emisiones por predio rural.

1.2.7 Modelo de consulta del consumo de agua potable

De igual manera que los modelos anteriores, antes de obtener el modelo final de consulta del agua potable, se explicará por partes cada consulta. En primer lugar, se debe consultar si el medidor de consumo de agua potable existe, para lo cual se va a contar el

número de filas (COUNT(*)) que resulte de consultar la tabla de registro de medidores (agua_medidor), filtrada por el código de medidor(codmedidor), también extrae el código del contribuyente (CodPropietario) que servirá más adelante para la consulta de los datos personales, para ello se tiene la siguiente consulta:

- SELECT COUNT(), codPropietario FROM agua_medidor WHERE codmedidor=’’.

Si el resultado es igual a 1 se entiende que el medidor existe, caso contrario no; como se había mencionado anteriormente, se tiene que consultar los datos personales del propietario del medidor, esto se realiza ingresando el código de contribuyente de la consulta anterior como filtro en la tabla de contribuyentes (propietario), para obtener la cedula, el apellido y el nombre del contribuyente, para ello se tiene la siguiente consulta:

- SELECT cedula, apellidoPropietario, nombrePropietario FROM propietario WHERE CodPropietario=”al CodPropietario de la consulta anterior”.

Luego de obtener los datos del contribuyente se procede a consultar si tiene medidores pendientes de pago, para ello se consulta la tabla “agua_factura”, filtrando por el medidor de la primera consulta e indicando solo las emisiones pendientes de pago (pagado), en esta consulta se extrae los siguientes campos de interés: código de emisión (codemision), fecha de emisión (fecha_factura), consumo de agua (valor_principal), alcantarillado (alcantarillado), basura (basura), total de la tarifa (total_tarifa), para lo cual se tiene la siguiente consulta:

- SELECT codemision, fecha_factura, valor_principal, alcantarillado, basura, total_tarifa WHERE codmedidor=’’ AND pagado=’0’.

Ahora queda por consultar el porcentaje de intereses de los medidores pendientes de pago, siempre y cuando existan, para ello se tiene que consultar la tabla de intereses (intereses), filtrando un rango de fechas según la fecha de emisión del medidor que se obtiene

de la consulta anterior y la fecha actual en la que se consulta, para lo cual se realiza lo siguiente:

- SELECT sum(porcentajeInteres) FROM intereses WHERE fecha BETWEEN ‘fecha de emisión del consumo del medidor’ AND ‘fecha actual’.

Finalmente para optimizar los tiempos de respuesta, se combinan todas las consultas anidando y haciendo uso del comando UNION de MySQL, para simplemente remplazar el código de medidor correspondiente, la consulta quedaría de la siguiente manera:

- (SELECT IF(COUNT(*)>0, b.cedula,'N.E.') as flag_ced_period, CONCAT(b.apellidoPropietario, " ", b.nombrePropietario) AS prop_fecha, 0 AS valor_principal, 0 AS alcantarillado, 0 AS basura, 0 AS total_tarifa, 0 AS interes FROM agua_medidor_abonado a, propietario b WHERE a.codmedidor='1405' AND b.codPropietario=a.codPropietario) UNION (SELECT (a.codemision) as flag_ced_period, (DATE_FORMAT(a.fecha_factura, "%d/%m/%Y")) as prop_fecha, a.valor_principal, a.alcantarillado, a.basura, a.total_tarifa, sum(b.porcentajeinteres) as interes FROM agua_factura a, intereses b WHERE (a.codMedidor='1405' AND a.pagado=0 AND estado=1 AND id_clase_rubro=1) AND (b.fecha BETWEEN IF(date_format(NOW(), "%Y-%m-%d") >= DATE_ADD(a.fecha_factura, INTERVAL 1 MONTH), date_format(DATE_ADD(a.fecha_factura, INTERVAL 1 MONTH), "%Y%m"), 100001) AND IF(date_format(NOW(), "%Y-%m-%d") >= DATE_ADD(a.fecha_factura, INTERVAL 1 MONTH), IF(DAY(NOW()) >= DAY(DATE_ADD(a.fecha_factura, INTERVAL period_diff(date_format(NOW(), "%Y%m"), date_format(a.fecha_factura, "%Y%m")) MONTH)), date_format(NOW( ), "%Y%m"), date_format(DATE_SUB(NOW(), INTERVAL 1 MONTH), "%Y%m")), 100001)) GROUP BY a.codemision)

1.2.7.1 Total de títulos emitidos anualmente

La tabla de registro de emisiones del consumo de agua potable, es la tabla "agua_factura", en la siguiente figura se puede ver la consulta correspondiente para obtener el número total de emisiones del consumo de agua potable:

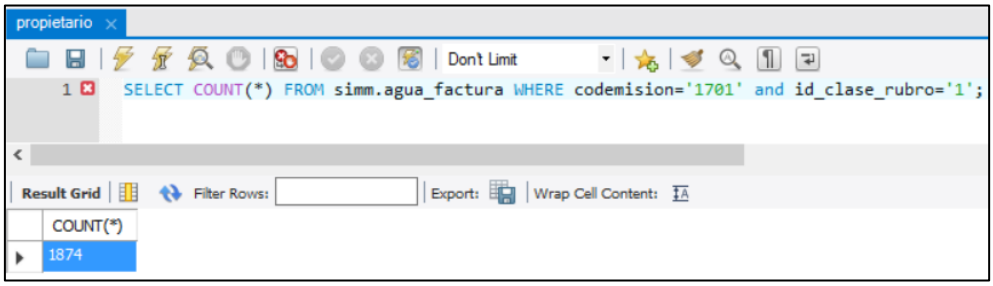


Figura 14. Consulta del total de emisiones de agua potable en el año 2017
Fuente: Autor

Donde claramente se observa un total de 1874 emisiones por consumo de agua en el en el año 2017.

1.2.7.2 Tasa de crecimiento anual

Con la ayuda de la consulta anterior, se consulta el total de emisiones por consumo de agua potable durante los últimos 5 años, seguidamente se analiza el crecimiento anual por periodo y se obtiene el promedio de crecimiento anual de las emisiones por consumo de agua potable, en la siguiente tabla se puede ver detalladamente cada uno de los datos:

Tabla 22. Detalle de crecimiento mensual de la emisión del consumo de agua potable

Año	Medidores Emitidos	Periodo	Tasa de Crecimiento	Total	Promedio
2013	1750	2013-2014	24	124	31
2014	1774				
2014	1774	2014-2015	44		
2015	1818				
2015	1818	2015-2016	41		
2016	1859				
2016	1859	2016-2017	15		
2017	1874				

Fuente: Autor

Donde se observa que el promedio de crecimiento anual analizado entre el año 2013 y 2017 es de 31 emisiones por consumo de agua potable.

1.2.8 Modelo de consulta de la patente municipal

Finalmente se obtiene el modelo de patente municipal, antes de obtener el modelo correspondiente, se explicará al igual que los modelos anteriores, realizándolo por partes cada consulta. En primer lugar, se debe consultar si la patente existe, para lo cual se tiene que contar el número de filas (COUNT(*)) que resulte de consultar la tabla de registro de patentes (patentes_patente), filtrada por el número de patente (numPatente), también se extrae el código del contribuyente (CodPropietario) que servirá más adelante para la consulta de los datos personales, para ello se tiene la siguiente consulta:

- SELECT COUNT(), codPropietario FROM patentes_patente WHERE numPatente=’’.

Si el resultado es igual a 1 se entiende que la patente existe, caso contrario no; como se menciona anteriormente, se tiene que consultar los datos personales del propietario de la patente, esto se realiza ingresando el código de contribuyente de la consulta anterior como filtro en la tabla de contribuyentes (propietario), para obtener la cedula, el apellido y el nombre del contribuyente, para ello se tiene la siguiente consulta:

- SELECT cedula, apellidoPropietario, nombrePropietario FROM propietario WHERE CodPropietario=”al CodPropietario de la consulta anterior”.

Luego de obtener los datos del contribuyente se procede a consultar si tiene patentes pendientes de pago, para ello se consulta la tabla “patentes_emision_patente”, filtrando por el número de patente de la primera consulta e indicando solo las emisiones pendientes de pago (pagado), en esta consulta se extrae los siguientes campos de interés: año (ano), fecha de emisión (fecha_emision), impuesto de la patente (valor_impuesto), impuesto al capital (impuesto_capital), servicios administrativos (serv_administrativos), otros (otros), valor de la especie (valor_especie), para lo cual se tiene la siguiente consulta:

- SELECT ano, fecha_emision, valor_impuesto, impuesto_capital, serv_administrativos, otros, valor_especie WHERE numPatente=’’ AND pagado=’0’.

Ahora queda por consultar el porcentaje de intereses de las patentes pendientes de pago, siempre y cuando existan, para ello se tiene que consultar la tabla de intereses (intereses), filtrando un rango de fechas según la fecha de emisión de la patente que se obtiene de la consulta anterior y la fecha actual en la que se consulta, para lo cual se realiza lo siguiente:

- SELECT sum(porcentajeInteres) FROM intereses WHERE fecha BETWEEN ‘fecha de emisión de la patente’ AND ‘fecha actual’.

Finalmente para optimizar los tiempos de respuesta, se combinan todas las consultas anidando y haciendo uso del comando UNION de MySQL, para simplemente remplazar el número de patente correspondiente, la consulta quedaría de la siguiente manera:

- (SELECT (COUNT(*)) as anio, concat(b.Cedula, " ", b.apellidoPropietario, " ", b.nombrePropietario) AS fecha, (0) AS valorImpuesto, (0) AS impuestoCapital, (0) AS servAdmin, (0) AS otro, (0) AS valorEspe, (0) AS interes FROM patentes_patente a, propietario b WHERE a.numPatente='100' AND b.codPropietario=a.codPropietario) UNION (SELECT (a.ano) as anio, (DATE_FORMAT(a.fecha_emision, "%d/%m/%Y")) as fecha, a.valor_impuesto, a.impuesto_capital, a.serv_administrativos, a.otro, a.Valor_Especie, sum(b.porcentajeinteres) AS interes FROM patentes_emision_patente a, intereses b WHERE (a.numpatente='100' AND a.pagado=0) AND (b.fecha BETWEEN IF(date_format(NOW(), "%Y-%m-%d") >= DATE_ADD(a.fecha_emision, INTERVAL 1 MONTH), date_format(DATE_ADD(a.fecha_emision, INTERVAL 1 MONTH), "%Y%m"), 100001) AND IF(date_format(NOW(),

"%Y-%m-%d") >= DATE_ADD(a.fecha_emision, INTERVAL 1 MONTH), IF(DAY(NOW( )) >= DAY(DATE_ADD(a.fecha_emision, INTERVAL period_diff(date_format(NOW(), "%Y%m"), date_format(a.fecha_emision, "%Y%m")) MONTH)), date_format(NOW( ), "%Y%m"), date_format(DATE_SUB(NOW( ), INTERVAL 1 MONTH), "%Y%m")), 100001)) GROUP BY a.ano).

1.2.8.1 Total de títulos emitidos anualmente

La tabla de registro de emisiones de la patente municipal, es la tabla "patentes_emision_patente", en la siguiente figura se puede ver la consulta correspondiente para obtener el número total de emisiones de patente registradas en el año 2017:

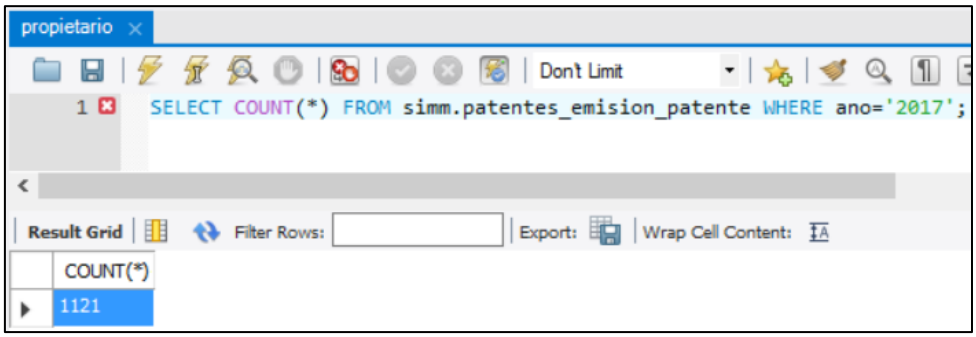


Figura 15. Consulta del total de títulos emitidos por patente 2017
Fuente: Autor

Donde claramente se observa un total de 1121 títulos emitidos por patente en el año 2017.

1.2.8.2 Tasa de crecimiento anual

Con la consulta de la figura anterior, se consulta el total de títulos por patente municipal emitidos durante los últimos 5 años, seguidamente se analiza el crecimiento anual por periodo y se obtiene el promedio de crecimiento anual de los títulos por patente municipal, en la siguiente tabla se puede ver detalladamente cada uno de los datos:

Tabla 23. Detalle de crecimiento anual de la emisión de patentes municipales

Año	Patentes Emitidas	Periodo	Tasa Crecimiento	Total	Promedio
2013	883	2013-2014	128	274	68.5

2014	1011				
2014	1011	2014-2015	-18		
2015	993				
2015	993	2015-2016	53		
2016	1046				
2016	1046	2016-2017	75		
2017	1121				

Fuente: Autor

Donde se observa que el promedio de crecimiento anual analizado entre el año 2013 y 2017 es de 68.5 títulos de crédito por patente municipal, que aproximadamente es 69.

1.3 Cantón Chordeleg

El cantón Chordeleg pertenece a la provincia del Azuay ubicado aproximadamente a 42km de la ciudad de Cuenca, es elevado a categoría de cantón el 15 de abril de 1992, actualmente tiene un total de 5 parroquias, la parroquia urbana de Chordeleg y cuatro parroquias rurales que son: Principal, Delegsol, San Martín de Puzhio y La Unión. Su temperatura promedio es de 16oC (GAD Municipal de Chordeleg, 2016).

Sus límites son al norte con el cantón Gualaceo cabecera cantonal parroquia Remigio Crespo Toral; al sur con el río Burroplaya en toda su extensión; al este con la parroquias Remigio Crespo y Daniel Córdova y la Provincia de Morona Santiago; y, al oeste, con la parroquia Guel del cantón Sigsig y la parroquia San Juan del cantón Gualaceo (GAD Municipal de Chordeleg, 2016).

1.3.1 Número de habitantes

El número de habitantes del cantón Chordeleg, se obtuvo desde la página web oficial del INEC (Instituto Nacional de Estadísticas y Censos), y según el censo más actual realizado en el año 2010, se tienen 6756 mujeres y 5821 hombres, dando un total de 12577 habitantes en el cantón Chordeleg (INEC, s.f.).

1.3.2 Tasa de crecimiento anual

Al igual que el número de habitantes, se obtuvo desde la página web oficial del INEC (Instituto Nacional de Estadísticas y Censos), y según el crecimiento entre el censo más actual realizado en el año 2010 y el anterior en el año 2001, se tiene una tasa de crecimiento anual promedio de 1.48%, lo cual es 187 habitantes por año (INEC, s.f.).

CAPITULO II

2 Calculo y Análisis del tráfico de datos

2.1 Introducción

En el presente capítulo se estudiará los diferentes escenarios que permitan dimensionar un sistema, proyectando el número total de clientes y el ancho de banda total necesario para garantizar la trasferencia de datos entre un cliente y el web service; lo cual dependerá del número de usuarios y el ancho de banda total del Municipio de Chordeleg.

2.2 Obtener, analizar y proyectar el número de usuarios que soportará el sistema

2.2.1 Análisis en base a la tasa de crecimiento de los habitantes del cantón

De igual manera, recordando los datos del capítulo anterior, se tiene un total de 12577 habitantes hasta el año 2010, con una tasa de crecimiento anual de 187 habitantes, lo cual proyectándolo 5 años después a partir del 2017, se obtiene la siguiente tabla:

Tabla 24. Proyección del número de habitantes del cantón Chordeleg en 5 años

Año	Contribuyentes
2010	12577
2011	12764
2012	12951
2013	13138
2014	13325
2015	13512
2016	13699
2017	13886
2018	14073
2019	14260
2020	14447
2021	14634

Fuente: Autor

Donde se puede observar que en 5 años posteriores al 2017, se tendrá un estimado de 14634 habitantes, lo cual sería el número de clientes a soportar el sistema si dependería únicamente de los habitantes del cantón Chordeleg.

2.2.2 Análisis en base a la tasa de crecimiento de contribuyentes

De acuerdo al levantamiento de información del capítulo 1, se tiene un total de 11593 contribuyentes, con una tasa de crecimiento anual de 645 contribuyentes; las políticas internas del GAD Municipal de Chordeleg establecen que un bien intangible como es el caso de desarrollo de software, tiene que cumplir al menos con 5 años de vida útil, por lo tanto se debe proyectar y garantizar el servicio como mínimo por un lapso de 5 años. Con estos datos conocidos, se proyecta el número de contribuyentes que se tendrán en el año 2021, en la siguiente tabla se puede observar la proyección de contribuyentes:

Tabla 25. Proyección del número de contribuyentes en 5 años

Año	Contribuyentes
2017	11593
2018	11608
2019	11623
2020	11638
2021	11653

Fuente: Autor

Donde se puede observar que en 5 años se tendrá un estimado de 11653 contribuyentes registrados, lo cual sería el número de clientes a que soportar el sistema si dependería únicamente de los contribuyentes del ´GAD Municipal de Chordeleg.

2.2.3 Análisis en base a la tasa de crecimiento de los títulos emitidos anualmente por los principales tributos municipales.

De acuerdo a los datos recopilados en el capítulo anterior, se tiene la siguiente tabla:

Tabla 26. Datos de los principales tributos del GAD Municipal de Chordeleg

Tributo Municipal	Títulos Emitidos	Tasa de Crecimiento Anual
Predio Urbano	2551	71
Predio Rural	11358	139
Agua Potable	1874	31
Patente Municipal	1121	69

Fuente: Autor

Sin embargo, existen contribuyentes que poseen más de un predio, medidor de agua potable o negocio, razón por la cual se realizará una consulta adicional para agrupar los títulos por contribuyente, ya que los clientes del sistema dependerán del número de usuarios que lo utilicen mas no del número de títulos emitidos. En las siguientes figuras se puede observar las consultas de cada tributo municipal agrupado por contribuyente:

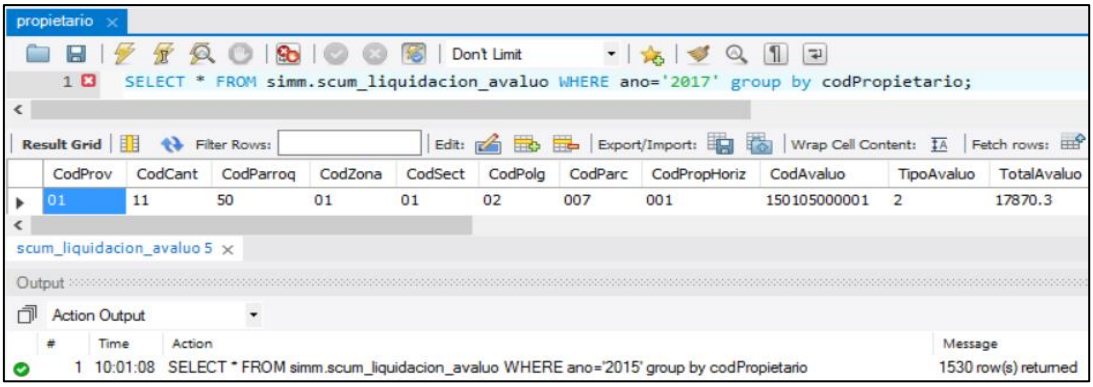


Figura 16. Total de predios urbanos agrupados por contribuyente
Fuente: Autor

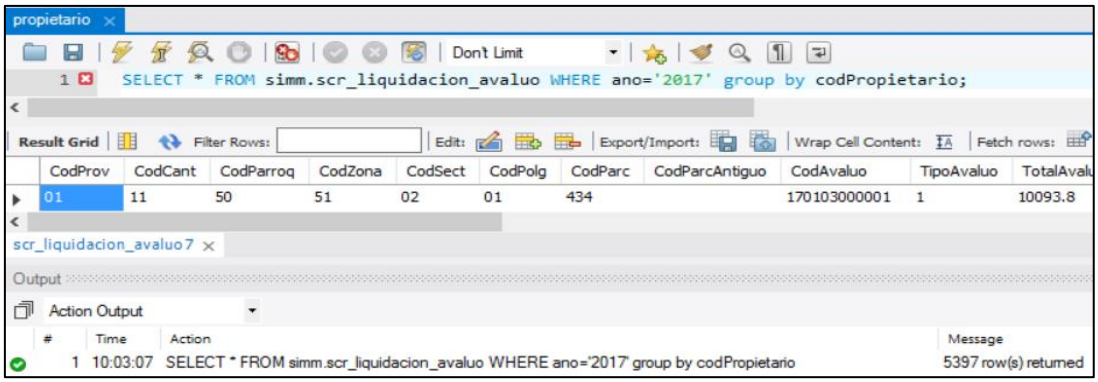


Figura 17. Total de predios rurales agrupados por contribuyente
Fuente: Autor

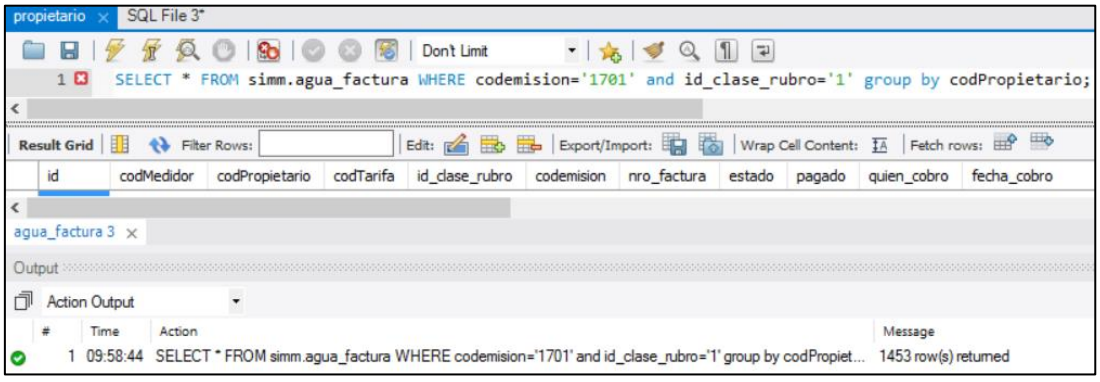


Figura 18. Total de consumos de agua potable agrupados por contribuyente
Fuente: Autor

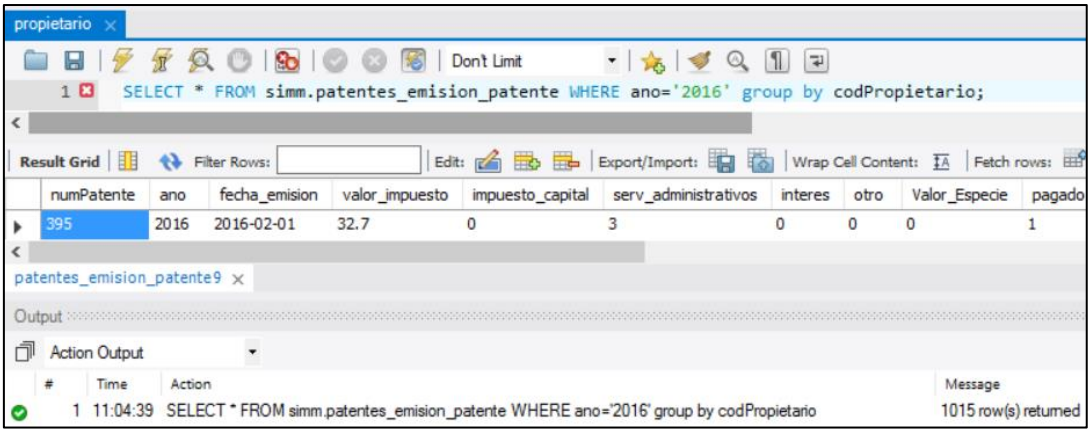


Figura 19. Total de patentes municipales agrupadas por contribuyente
Fuente: Autor

Donde se obtiene la siguiente tabla:

Tabla 27. Datos de los principales tributos agrupados por contribuyentes del GAD Municipal de Chordeleg

Tributo Municipal	Contribuyentes	Tasa de Crecimiento Anual
Predio Urbano	1530	71
Predio Rural	5397	139
Agua Potable	1453	31
Patente Municipal	1015	69

Fuente: Autor

Continuando con el análisis, los contribuyentes del agua ya están incluidos en el predio urbano, de acuerdo a la información obtenida en el GAD Municipal de Chordeleg los derechos de agua se dan paso únicamente a quienes poseen un predio, por lo tanto en contribuyentes del agua se tendría 0; en cuanto a las patentes, existen algunos contribuyentes que ya están incluidos en los predios, en la siguiente figura se puede observar la consulta realizada para sumar los contribuyentes del predio más los contribuyentes de patente que no se encuentren en el predio:

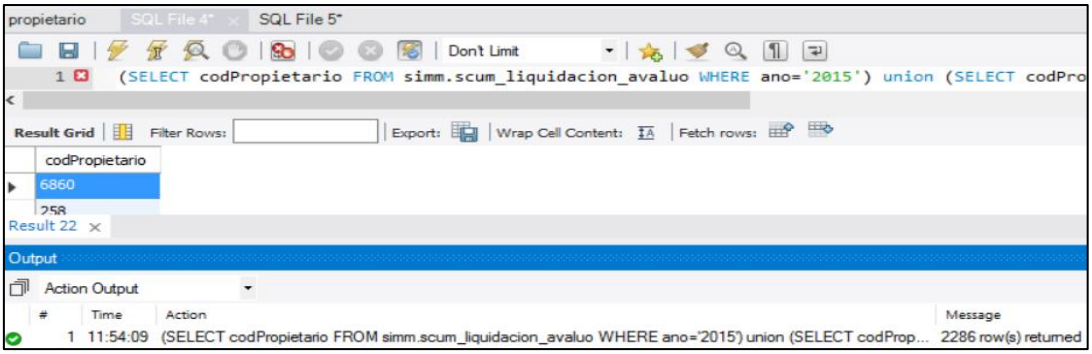


Figura 20. Total de predios más patentes agrupados por contribuyente
Fuente: Autor

Observando la figura anterior, se obtiene 2286 contribuyentes filtrados entre el predio urbano y la patente municipal, es decir, en la tabla anterior se tiene un total de 1530 contribuyentes en el predio urbano, lo cual al descontar de los 2286 contribuyentes daría un total de 756 contribuyentes de la patente municipal que no están incluidos en los predios. En resumen hasta aquí se tiene la siguiente:

Tabla 28. Datos de los principales tributos municipales realizando un análisis de los contribuyentes del GAD Municipal de Chordeleg.

Tributo Municipal	Contribuyentes	Tasa de Crecimiento Anual
Predio Urbano	1530	71
Predio Rural	5397	139
Agua Potable	0	31
Patente Municipal	756	69

Fuente: Autor

Con lo cual al proyectar un crecimiento de contribuyentes a 5 años, se obtiene lo siguiente:

Tabla 29. Proyección del número de los principales tributos del GAD Municipal de Chordeleg en 5 años.

Año	Títulos Emitidos				
	Predio Urbano	Predio Rural	Agua Potable	Patente Municipal	Total
2017	1530	5397	0	756	7683
2018	1601	5536	0	825	7962
2019	1672	5675	0	894	8241
2020	1743	5814	0	963	8520
2021	1814	5953	0	1032	8799

Fuente: Autor

Donde se puede observar que en 5 años posteriores al 2017, se tendría un total de 8799 contribuyentes, lo que es igual a los clientes a soportar el sistema si dependería únicamente de los títulos emitidos por los principales tributos municipales.

2.2.4 Tabla comparativa

2.2.4.1 Tabla

A fin de realizar un análisis con motivo de obtener el número de clientes que soportará el sistema, se resumen los datos obtenidos anteriores en una sola tabla.

Tabla 30. Resumen del total de contribuyentes según el análisis realizado.

Tipo de Análisis	Contribuyentes
Habitantes del cantón Chordeleg	14634
Contribuyentes del GAD de Chordeleg	11653
Principales tributos municipales del GAD de Chordeleg	8799

Fuente: Autor

2.2.4.2 Análisis general

Una vez obtenido el total de contribuyentes en los diferentes análisis realizados, lo cual se refleja en la tabla anterior, es claro notar que se parte de los habitantes de todo el cantón Chordeleg, para luego filtrar los contribuyentes del GAD Municipal de Chordeleg y por último llegar a obtener específicamente los contribuyentes de los principales tributos municipales, en vista que son ellos quienes harán uso del sistema en desarrollo para la consulta de los principales tributos municipales del GAD Municipal de Chordeleg; obteniendo al final un total 8799 clientes que pueden hacer uso del sistema.

2.3 Analizar el ancho de banda disponible en el GAD Municipal de Chordeleg

2.3.1 Consumo del ancho de banda de bajada

En el GAD Municipal de Chordeleg, existe un límite general de ancho de banda de bajada máximo de 2Mbps por departamento, forzando a limitar cualquier servicio que exista, por otra parte de acuerdo al monitoreo realizado se nota claramente por ejemplo que el ancho

de banda sobrante en departamentos que no estaban haciendo uso del internet, no se reaprovechaba para reasignar a quienes si lo hacían.

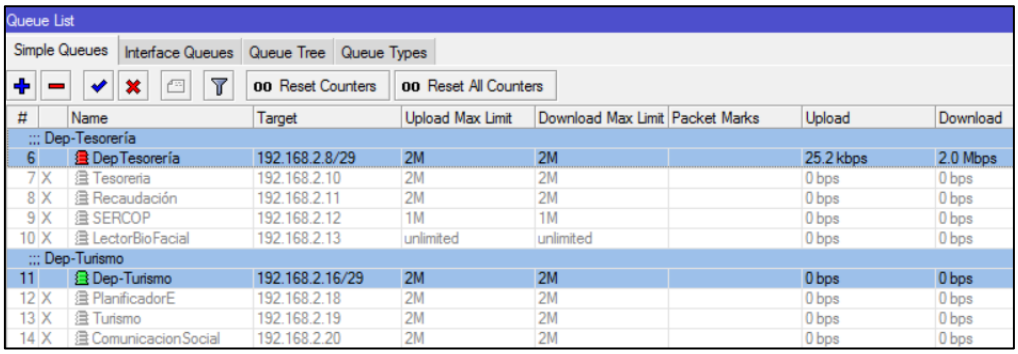


Figura 21. Control de ancho de banda de bajada del GAD Municipal de Chordeleg
Fuente: Autor

Tal es el caso que se puede observar en la figura anterior, que el ancho de banda de bajada del departamento de tesorería se encuentra totalmente saturado, mientras que en el departamento de turismo no existe ningún tipo de actividad, y como se había mencionado en el capítulo 1, el GAD Municipal de Chordeleg tiene contratado 10Mbps simétricos sin compartimiento, es decir estaría desperdiciado 7Mbps en este caso en específico. Sin bien es cierto el ancho de banda de bajada no afecta en gran parte al presente proyecto en vista que está destinado a servir la mayor parte de datos desde la red LAN hacia un cliente en internet, se aplicará calidad de servicio con motivo de mejorar en su mayoría la red interna.

También se realizó un monitoreo de la red LAN durante una semana para obtener el consumo promedio del ancho de banda de bajada, haciendo uso del software de administración del router mikrotik, el cual se había mencionado en el capítulo 1 y lleva como nombre RouterOS, en la siguiente figura se puede observar el consumo de ancho de banda en un momento dado haciendo uso del software RouterOS:

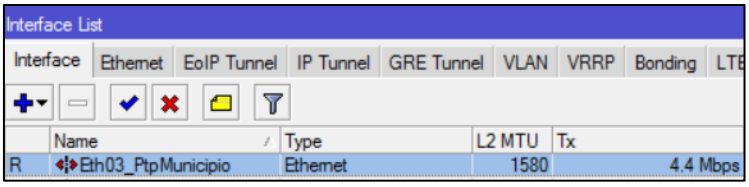


Figura 22. Ancho de banda de bajada del GAD Municipal de Chordeleg
Fuente: Autor

Con lo cual se logró obtener los siguientes datos:

Tabla 31. Datos de un monitoreo del consumo de ancho de banda de bajada en el municipio.

Hora	Ancho de banda de bajada (Mbps)				
	Día 1	Día 2	Día 3	Día 4	Día 5
8:30	4.5	4.1	4.3	4.7	4.1
9:30	5.2	5.4	5.3	5.5	5
10:30	6.7	6.5	4.2	6.7	4.2
11:30	6.5	6.3	6.2	6.3	6.9
12:30	8.1	8.1	8.0	7.8	8.0
13:30	0.1	1.2	1.1	0.1	0.2
14:30	6.2	6.5	4.1	6.2	6.2
15:30	6.5	6	4.9	6.3	6.6
16:30	8.0	7.9	8.1	7.8	8.2

Fuente: Autor

En los datos obtenidos se puede notar que existen horas en las que el ancho de banda no llega ni a siquiera a la mitad de lo disponible que es los 10Mbps, esto se debe por el mismo hecho de estar limitado los departamentos a 2Mbps y no es reasignado el ancho de banda sobrante; así mismo existen horas en las que el ancho de banda es saturado, sobre todo al medio día, y quizás este ancho de banda sea únicamente consumido por tres departamentos, ya que no se tiene una distribución uniforme del ancho de banda total. Realizando un promedio de los datos obtenidos en la tabla 31 se tiene un consumo del ancho de banda de bajada de 5.57Mbps.

2.3.2 Consumo del ancho de banda de subida

En el GAD Municipal de Chordeleg, al igual que el ancho de banda de bajada, existe un límite general de ancho de banda de subida máximo de 2Mbps por departamento, forzando a limitar cualquier servicio que exista, sin embargo por el mismo hecho de que el GAD Municipal de Chordeleg no provee de servicios por internet, el consumo es bajo, teniendo únicamente como servicio que consume el ancho de banda de subida, el correo institucional

que se encuentra en un hosting externo a la institución. En la siguiente figura se puede ver el consumo de ancho de banda de subida de un departamento en específico:

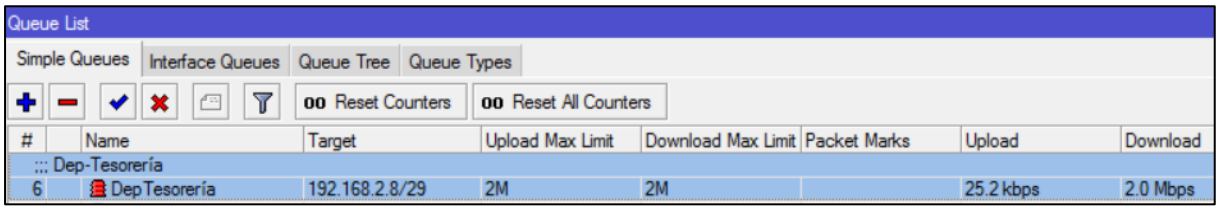


Figura 23. Control de ancho de banda de bajada del Departamento de Tesorería
Fuente: Autor

Donde se puede observar que mientras el ancho de banda de bajada está saturado, el ancho de banda de subida apenas está en 25.2kbps que es igual al 1.26% de lo disponible.

Es así que se procedió a realizar un monitoreo de la red LAN durante una semana para obtener el consumo promedio del ancho de banda de subida, de igual manera haciendo uso del software RouterOS, se puede observar en la siguiente figura el consumo de ancho de banda en un momento dado:

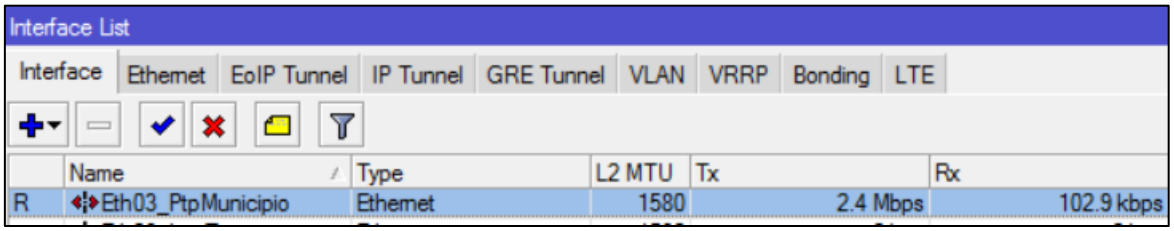


Figura 24. Ancho de banda de subida del GAD Municipal de Chordeleg
Fuente: Autor

Realizando esta misma tarea, se logró obtener los siguientes datos:

Tabla 32. Datos de un monitoreo del consumo de ancho de banda de subida en el municipio

Hora	Ancho de banda de subida (kbps)				
	Día 1	**Día 2**	**Día 3**	**Día 4**	**Día 5**
8:30	100.5	110.2	109.9	117.5	119.2
9:30	103.2	100.2	101.7	115.2	120.5
10:30	120.5	106.5	103.2	116.2	101.5
11:30	101.1	107.7	109.3	101.2	100.5
12:30	106.6	106.8	107.4	100.2	100.1
13:30	0.1	0.1	0.1	0.1	0.1
14:30	111.9	110.1	104.4	108.4	108.8
15:30	110.2	107.8	10.4.2	109.6	108.6

16:30	115.2	100.5	107.1	107.3	105.7

Fuente: Autor

En los datos obtenidos se puede notar claramente que el consumo de ancho de banda de bajada es muy bajo, estando por debajo de los 120.5kbps, realizando un promedio se tiene un consumo del ancho de banda de subida del 93.4kbps, que representa el 0.93% del total de ancho de banda disponible que es de 10Mbps.

2.3.3 Realizar un test de velocidad de subida y comparar frente a lo contratado

El test de velocidad, se realizó directamente en el nodo principal situado en la red de core, para lo cual se ha configurado una computadora con los datos técnicos de la red de CNT y se conectó directamente al router de la CNT, en la siguiente figura se puede observar un esquema general de la conexión realizada para el test de velocidad:

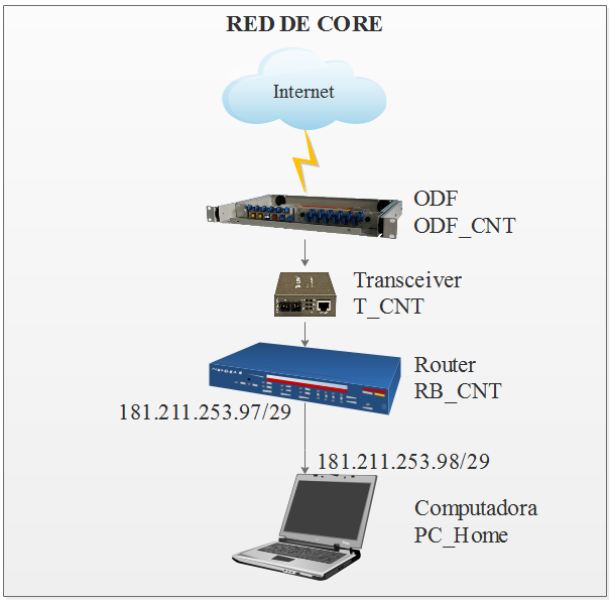


Figura 25. Esquema de conexión para realizar un test de velocidad de internet
Fuente: Autor

Entrando en detalle la configuración de la Computadora es la siguiente:

- Dirección IP: 181.211.253.98
- Máscara: 255.255.255.248
- Gateway: 181.211.253.97
- DNS Primario: 208.67.220.220

- DNS Secundario: 208.67.222.222

Una vez conectada y configurada la computadora, se realizaron las pruebas de velocidad del servicio, haciendo uso de la página web http://speedtest.cnt-grms.com.ec/, la cual es de la misma empresa CNT, que brinda sin costo alguno el servicio de pruebas de velocidad de internet, a continuación se presenta los resultados obtenidos.

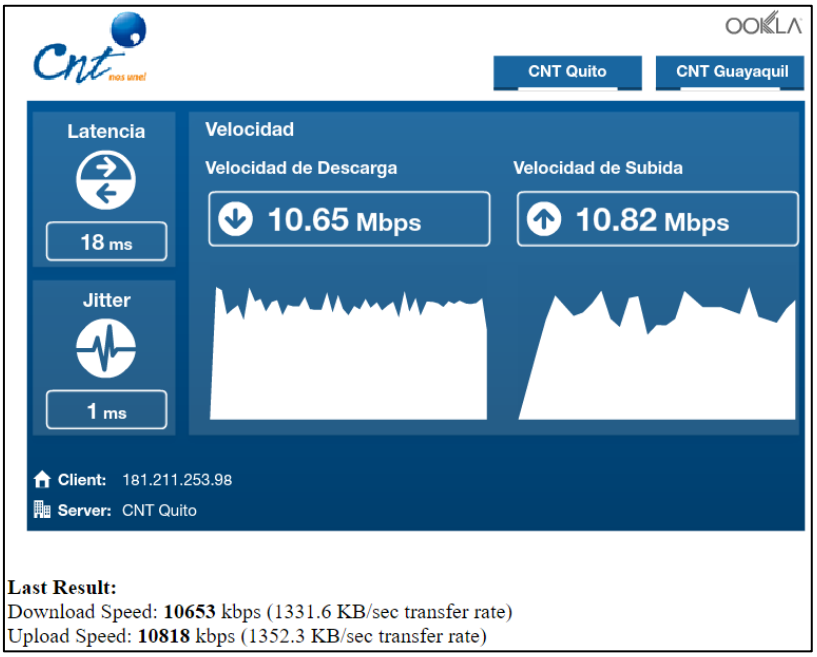


Figura 26. Test de velocidad de internet en la página de la CNT
Fuente: http://speedtest.cnt-grms.com.ec/

De acuerdo a los datos obtenidos en la figura anterior y comparando con los datos contratados a la empresa CNT, se puede ver que cumple con los 10Mbps contratados, siendo el resultado obtenido de 10.65Mbps.

Con el fin de corroborar la información anterior, se dirige a la página http://www.speedtest.net/es/ que es muy popular en la red para medir la velocidad de internet sin costo alguno, y se obtiene los siguientes resultados:

Figura 27. Test de velocidad de internet en la página de SpeedTest
Fuente: http://www.speedtest.net/es/

De igual manera se observa que el servicio de internet de la CNT cumple con los 10Mbps contratados, obteniendo como resultado 10.51Mbps.

2.3.4 Realizar un test de velocidad de bajada y comparar frente a lo contratado

Para el test de velocidad sirven los resultados obtenidos en el test de velocidad de subida, lo cuales se puede observar en la figura 26 con un resultado de 10.82Mbps y en la figura 27 un resultado de 10.69Mbps, por lo tanto al igual que el ancho de banda de bajada se cumple el ancho de banda de subida contratado a la empresa CNT, el cual es de 10Mbps.

2.3.5 Tabla Comparativa

2.3.5.1 Tabla

A fin de realizar un análisis de los datos obtenidos tanto en el ancho de banda de bajada como el de subida, se tiene la siguiente tabla.

Tabla 33. Ancho de banda en el GAD Municipal de Chordeleg.

Tipo	Ancho de banda		
	Contratado	Medido	Utilizado en el GAD de Chordeleg (promedio)
Bajada	10Mbps	10.65Mbps	5.57Mbps
Subida	10Mbps	10.82Mbps	93.4kbps

Fuente: Autor

2.3.5.2 Análisis General

Haciendo un análisis de la tabla anterior, se puede ver que el ancho de banda de bajada no es aprovechado al máximo, lo cual se debe por el mismo hecho de tener limitado de manera forzada el ancho de banda por departamento y no permitir reasignar el ancho de banda sobrante. En cuanto al ancho de banda de bajada, se refleja claramente los resultados, ya que al no tener ningún servicio web, la mayor parte del tiempo está en desuso, estando disponible únicamente para enviar correos.

Estos datos obtenidos servirán de mucho para realizar el cálculo respectivo, de cuanto ancho de banda se dispone para implementar el web service y brindar el servicio de consulta de los principales tributos municipales del cantón Chordeleg.

2.4 Calcular, analizar y determinar la cantidad de datos requeridos para realizar la consulta de los principales tributos municipales.

Antes de entrar en detalle en el cálculo de la cantidad de datos por cada tributo municipal, es necesario conocer el formato y encabezado de los datos que tendrá el web service, para sumarle al total de datos de cada consulta.

Si bien es cierto que en el capítulo 3 se estudiará todo lo relacionado con el web service, es necesario por ahora indicar que el formato de datos utilizado en el web service es “JSON”, el mismo que hoy en día es una excelente alternativa al formato XML para identificar y gestionar los datos transferidos entre el cliente y el servidor. A continuación se tiene la estructura que debe llevar un formato JSON:

- [{"etiqueta1":"dato1", "etiqueta2":"dato2",…}]

Donde se define una etiqueta para identificar cada dato que se desee transferir.

En cuanto al encabezado de datos que lleva el web service, se tienen los que recibe el web service y los que envía, de igual manera se estudiará en el siguiente capítulo, sin embargo por ahora se analizará de manera general para obtener el tamaño del mismo que será

sumado al tamaño de consulta por cada tributo municipal; el web service tendrá los siguientes datos en el encabezado cuando responde a una consulta:

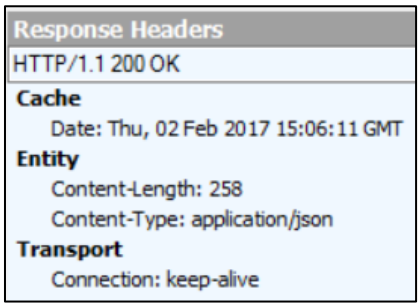


Figura 28. Encabezado de datos por respuesta del Web Service
Fuente: Autor

De lo cual sumando los caracteres correspondientes a la Response Headers, Date, Content-Lenght, Content-Type y Connection; se tiene un total de 133 caracteres incluidos los espacios correspondientes que es igual a un total de 133Bytes, los mismos que tienen que ser sumados a cada consulta que se estudiará en la siguiente sección.

En cuanto a los datos de encabezado que tiene el web service por cada consulta que recibe se puede observar en la siguiente figura:

Figura 29. Encabezado de datos por consulta al Web Service
Fuente: Autor

De lo cual sumando los caracteres correspondientes a la Request Headers, Client, y Transport; se tiene un total de 60 caracteres incluidos los espacios correspondientes que es igual a un total de 60Bytes.

2.4.1 Total de datos requeridos para consultar el predio urbano

Anteriormente se había visto en la sección 1.2.3, la estructura de las tablas relacionadas con el predio urbano, donde cada carácter consultado a la base de datos SIMM, representará 1Byte, por lo tanto se consultará el promedio de caracteres de todos los campos

de interés, a continuación se tiene un ejemplo para consultar el promedio del nombre y apellido de un propietario:

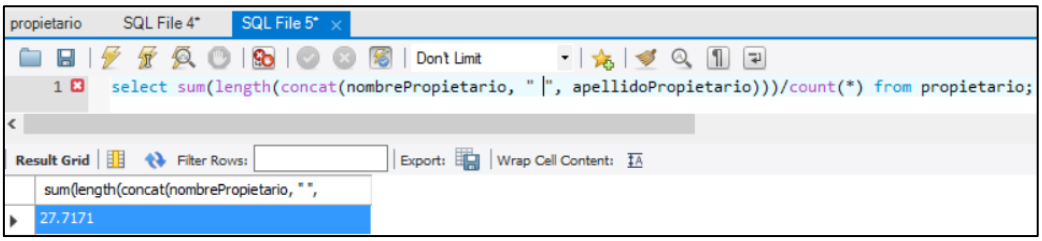


Figura 30. Consulta del tamaño promedio de datos del contribuyente
Fuente: Autor

Los contribuyentes en promedio tienen 27.72 caracteres por campo, que es igual a un aproximado de 28 caracteres, y al tener un 1 Byte por carácter, se tiene en este campo un total de 28Bytes.

A más del tamaño de los campos a consultar, se tiene que definir la etiqueta para identificar cada campo y colocarlo en formato JSON, es así que haciendo uso de la consulta anterior, donde se remplaza únicamente el campo a consultar y el nombre de la tabla que es "scum_liquidacion_avaluo", se obtiene los siguientes datos:

Tabla 34. Datos de consulta del predio urbano.

Datos	Campo	Campo (Bytes)	Etiqueta JSON	Etiqueta (Bytes)	SubTotal (Bytes)	Total (Bytes)
Generales	Propietario	28	[{"p1":"",	10	38	183
	Cédula	10	"c1":"",	8	18	
Deuda Actual	Fecha de Emisión	10	"f2":"",	8	18	
	Período	4	"p2":"",	8	12	
	Valor	7	"v2":"",	8	15	
	Interés	6	"i2":"",	8	14	
Deuda Anterior	Planillas	1	"t3":"",	8	9	
	Periodo	4	"p3":"",	8	12	
	Valor	7	"v3":"",	8	15	
	Interés	6	"i3":"",	8	14	
Deuda Total	Valor	8	"d4":""]}	10	17	

Fuente: Autor

En total se tiene 183Bytes, a esto se tiene que sumar los 133Bytes de la cabecera de datos de la respuesta del web service, llegando a un promedio total de 316Bytes que el web service tiene servir por cada consulta de predio urbano.

En cuanto a los datos que recibe el web service por cada consulta es únicamente el tamaño de encabezado de la figura 29, el cual es de 60Bytes.

2.4.2 Total de datos requeridos para consultar el predio rural

Para el cálculo de datos del predio rural, básicamente es el mismo esquema que el predio urbano, se tiene la estructura de las tablas relacionadas con el predio rural en la sección 1.2.3, y haciendo uso de la misma estructura de consulta de la figura 30, remplazando únicamente el campo a consultar y el nombre de la tabla que es “scr_liquidacion_avaluo”, se obtiene los siguientes datos:

Tabla 35. Datos de consulta del predio urbano.

Datos	Campo	Campo (Bytes)	Etiqueta (JSON)	Etiqueta (Bytes)	SubTotal (Bytes)	Total (Bytes)
Generales	Propietario	28	[{"p1":"",	10	38	183
	Cédula	10	"c1":"",	8	18	
Deuda Actual	Fecha de Emisión	10	"f2":"",	8	18	
	Período	4	"p2":"",	8	12	
	Valor	7	"v2":"",	8	15	
	Interés	6	"i2":"",	8	14	
Deuda Anterior	Planillas	1	"t3":"",	8	9	
	Periodo	4	"p3":"",	8	12	
	Valor	7	"v3":"",	8	15	
	Interés	6	"i3":"",	8	14	
Deuda Total	Valor	8	"d4":""]}	10	17	

Fuente: Autor

En total se tiene 183Bytes, a esto se tiene que sumar los 133Bytes de la cabecera de datos la respuesta del web service, llegando a un promedio total de 316Bytes por cada consulta de predio rural.

De igual manera que en el predio urbano, los datos que recibe el web service por cada consulta es únicamente el tamaño de encabezado de la figura 29, el cual es de 60Bytes.

2.4.3 Total de datos requeridos para consultar el agua potable.

Para el cálculo de datos del agua potable, una vez más se tiene que dirigir a la sección 1.2.3, donde se encuentra la estructura de las tablas relacionadas con el agua potable, luego con la consulta de la figura 30, remplazando únicamente el campo a consultar y el nombre de la tabla que es “agua_factura”, se obtiene los siguientes datos:

Tabla 36. Datos de consulta del agua potable.

Datos	Campo	Campo (Bytes)	Etiqueta (JSON)	Etiqueta (Bytes)	SubTotal (Bytes)	Total (Bytes)
Generales	Propietario	28	[{"p1":"",	10	38	177
	Cédula	10	"c1":"",	8	18	
Deuda Actual	Fecha de Emisión	10	"f2":"",	8	18	
	Período	4	"p2":"",	8	12	
	Valor	6	"v2":"",	8	14	
	Interés	5	"i2":"",	8	13	
Deuda Anterior	Planillas	1	"t3":"",	8	9	
	Periodo	4	"p3":"",	8	12	
	Valor	6	"v3":"",	8	14	
	Interés	5	"i3":"",	8	13	
Deuda Total	Valor	6	"d4":""]}	10	17	

Fuente: Autor

En total se tiene 177Bytes, a esto se tiene que sumar los 133Bytes de la cabecera de datos de la respuesta del web service, llegando a un promedio total de 310Bytes por cada consulta del agua potable.

De igual manera que los anteriores, los datos que recibe el web service por cada consulta es únicamente el tamaño de encabezado de la figura 29, el cual es de 60Bytes.

2.4.4 Total de datos requeridos para consultar la patente municipal

Para finalizar la consulta de los tributos principales del GAD Municipal de Chordeleg, se tiene los cálculo de datos de la patente municipal, que al igual que los tributos anteriores,

en la sección 1.2.3 se encuentra la estructura de las tablas relacionadas con el patente municipal, luego con la consulta de la figura 30, remplazando únicamente el campo a consultar y el nombre de la tabla que es “patentes_emision_patente”, se obtiene los siguientes datos:

Tabla 37. Datos de consulta de la patente municipal.

Datos	Campo	Campo (Bytes)	Etiqueta (JSON)	Etiqueta (Bytes)	SubTotal (Bytes)	Total (Bytes)
Generales	Propietario	28	[{"p1":"",	10	38	182
	Cédula	10	"c1":"",	8	18	
Deuda Actual	Fecha de Emisión	10	"f2":"",	8	18	
	Período	4	"p2":"",	8	12	
	Valor	7	"v2":"",	8	15	
	Interés	6	"i2":"",	8	14	
Deuda Anterior	Planillas	1	"t3":"",	8	9	
	Periodo	4	"p3":"",	8	12	
	Valor	7	"v3":"",	8	15	
	Interés	6	"i3":"",	8	14	
Deuda Total	Valor	7	"d4":""]}	10	17	

Fuente: Autor

En total se tiene 182Bytes, a esto se tiene que sumar los 133Bytes de la cabecera del web service, llegando a un promedio total de 315Bytes por cada consulta de patente municipal.

De igual manera que los anteriores, los datos que recibe el web service por cada consulta es únicamente el tamaño de encabezado de la figura 29, el cual es de 60Bytes.

2.4.5 Tabla Comparativa

2.4.5.1 Tabla

A fin de realizar un análisis de los datos obtenidos por cada tributo municipal del GAD de Chordeleg, se tiene la siguiente tabla obtenida en base a los cálculos anteriores realizados para cada tributo municipal:

Tabla 38. Datos por cada consulta de los tributos principales del municipio de Chordeleg.

Tributo Municipal	Datos (Bytes)			
	Campos	Cabecera Web Service	Total	Promedio
Predio Urbano	183	133	316	314.25
Predio Rural	183	133	316	
Agua Potable	177	133	310	
Patente Municipal	182	133	315	

Fuente: Autor

En cuanto a los datos que recibe el web service, no es necesario realizar una tabla ya que el tamaño de datos depende únicamente del encabezado que es el mismo en todos los casos, dando un total de 60Bytes

2.4.5.2 Análisis General

Observando la tabla anterior, se tiene un promedio de 314.25Bytes que responde el web service por cada consulta y a su vez recibe 60Bytes. Con el tamaño de datos por consulta, la cantidad de usuarios y el ancho de banda disponible, se realizará en la siguiente sección, un análisis global de la carga que tendrá que soportar el sistema, así mismo tomando en cuenta el ancho de banda disponible en el GAD Municipal de Chordeleg.

2.5 Estudiar y analizar el ancho de banda en los diferentes escenarios de saturación del sistema

2.5.1 Análisis en base al número de usuarios

Al llegar a este punto, se conoce de antemano el número total de clientes que pueden realizar las consultas al sistema y el tamaño de datos por consulta tanto enviados como recibidos, por lo tanto, se definirá 4 escenarios donde se proyecta una concurrencia baja, media, alta y hasta saturar completamente de clientes el sistema; y seguidamente se realiza el cálculo respectivo para obtener la cantidad de datos que tendría que soportar el sistema acorde al escenario presentado. En primer lugar se detalla en base a los datos que tiene que responder el sistema.

Tabla 39. Total de datos a responder el web service

Escenario	Clientes Simultáneos	Tamaño x consulta (Bytes)	Total = clientes*tamaño/1000 (KBytes)	Ancho de banda requerido = (total * 8)/1000 (Mbps)
Bajo	1000	314.25	314.25	2.51
Medio	4500	314.25	1414.13	11.31
Alto	7000	314.25	2199.75	17.60
Saturado	8799	314.25	2765.09	22.12

Fuente: Autor

Donde se puede observar que el web service para atender a 1000 clientes simultáneamente es necesario un ancho de banda de subida de 2.51Mbps, para 4500 clientes simultáneamente se necesitaría un ancho de banda de subida de 11.31Mbps, para atender a 7000 clientes se necesitaría un ancho de banda de subida de 17.60Mbps y finalmente cuando el sistema es saturado completamente para atender a 8799 clientes es necesario disponer de un ancho de banda de subida de 22.12Mbps.

En cuanto a los datos que recibe el sistema, se detalla en la siguiente tabla:

Tabla 40. Total de datos a recibir el web service

Escenario	Clientes Simultáneos	Tamaño x consulta (Bytes)	Total = clientes*tamaño/ 1000 (KBytes)	Ancho de banda requerido = (total * 8)/1000 (Mbps)
Bajo	1000	60	60	0.48
Medio	4500	60	270	2.16
Alto	7000	60	420	3.36
Saturado	8799	60	527.94	4.22

Fuente: Autor

Así mismo se puede observar que el web service para atender 1000 clientes simultáneamente se necesitaría un ancho de banda de bajada de 0.48Mbps, para atender 4500 clientes simultáneamente se necesitaría un ancho de banda de bajada de 2.16Mbps, para atender a 7000 clientes se necesitaría un ancho de banda de bajada de 3.36Mbps y finalmente cuando el sistema es saturado completamente para atender a 8799 clientes es necesario disponer de un ancho de banda de bajada de 4.22Mbps

En todos los cálculos anteriores se analiza para el caso de atender todas las peticiones en un segundo, evitando colas en la red.

2.5.2 Análisis en base a las horas pico

En la tabla 31 se tiene un monitoreo del ancho de banda de bajada, donde se puede notar que existen dos horas específicas consideradas como horas pico en cuanto al consumo de ancho de banda de bajada del GAD Municipal de Chordeleg, de 12:00 a 13:00 y de 16:00 a 17:00, donde el promedio de ancho de banda utilizado en cualquiera de las dos horas es de 8Mbps.

Tabla 41. Ancho de banda de bajada requerido durante la hora pico

Escenario	Clientes Simultáneos	Ancho de banda del web service (Mbps)	Ancho de banda de la red (Mbps)	Total (Mbps)
Bajo	1000	0.48	8.00	8.48
Medio	4500	2.16	8.00	10.16
Alto	7000	3.36	8.00	11.36
Saturado	8799	4.22	8.00	12.22

Fuente: Autor

Donde se puede observar que el ancho de banda de bajada requerido para recibir peticiones de 8799 clientes en un segundo es de 12.22Mbps; cabe indicar que más adelante aún quedan algunos aspectos por analizar para obtener el total del ancho de nada de bajada mínimo y recomendado para el sistema.

De la misma forma que el análisis en la hora pico del ancho de banda de bajada se analizará el de subida. En tabla 32 se tiene un monitoreo del ancho de banda de subida y no existe en ningún momento un excesivo consumo del ancho de banda disponible; en el proyecto que se está desarrollando, es importante mencionar que es de interés analizar la demanda que existe de la red interna para servir datos a través de internet en vista que en esta vía se encuentra gran cantidad de datos, pero al no tener mayor consumo da igual analizar en cualquier hora del día. A continuación se analizará el comportamiento de la red en los

mismos escenarios de la tabla anterior, tomando el mayor consumo de ancho de banda de subida de la tabla 32, que es de 120.5kbps:

Tabla 42. Total de datos a soportar el sistema en una hora determinada.

Escenario	Clientes Simultáneos	Ancho de banda del web service (Mbps)	Ancho de banda de la red (kbps)	Total (Mbps)
Bajo	1000	2.51	120.50	2.63
Medio	4500	11.31	120.50	11.43
Alto	7000	17.60	120.50	17.72
Saturado	8799	22.12	120.50	22.24

Fuente: Autor

Donde se puede observar que el ancho de banda requerido en cada escenario, incrementa apenas en 120.50kbps, lo cual no es muy significativo pero necesario para dimensionar el sistema y garantizar el ancho de banda de subida requerido.

2.5.3 Análisis en base a las consultas

En primer lugar se necesita conocer la cantidad de conexiones simultáneas que soporta el servidor de la base de datos SIMM, como se había mencionado en el capítulo 1, el servidor fue instalado y configurado por la empresa PLANERP de la ciudad de Cuenca; para consultar dicho valor se realizó la siguiente consulta:

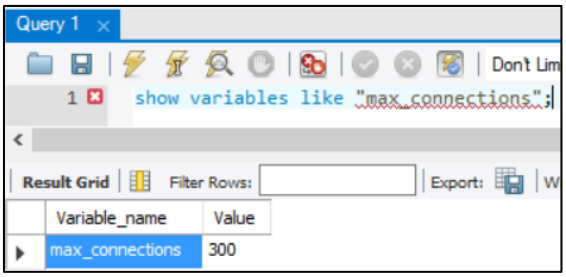


Figura 31. Máximas conexiones simultaneas del servidor de la base de datos SIMM
Fuente: Autor

Por lo tanto el número máximo permitido de conexiones simultáneas es de 300, seguidamente se necesita conocer el tiempo de respuesta que toma realizar las consultas de cada tributo municipal obtenidas en las secciones 1.2.5, 1.2.6, 1.2.7 y 1.2.8, en las siguientes figuras se puede observar el tiempo que toma obtener la consulta del predio urbano, predio rural, agua potable y patente municipal respectivamente:

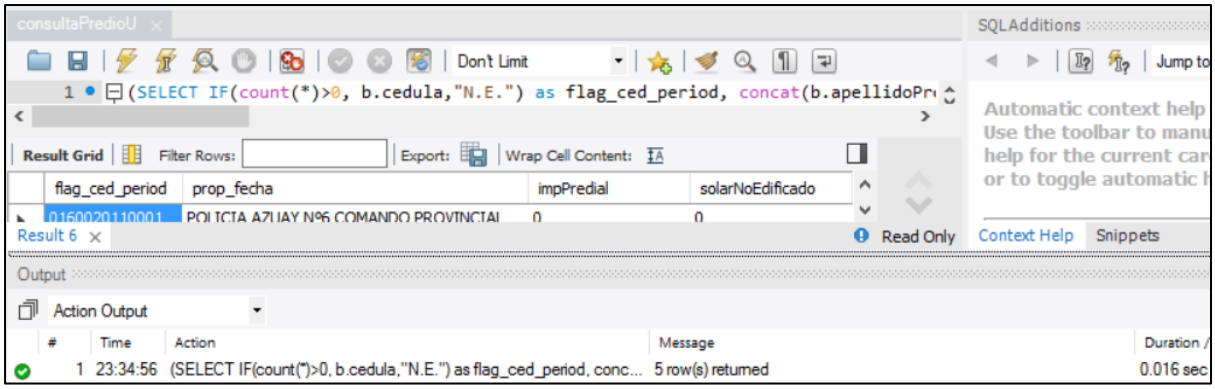


Figura 32. Tiempo de respuesta de MySQL por consultar el predio urbano
Fuente: Autor

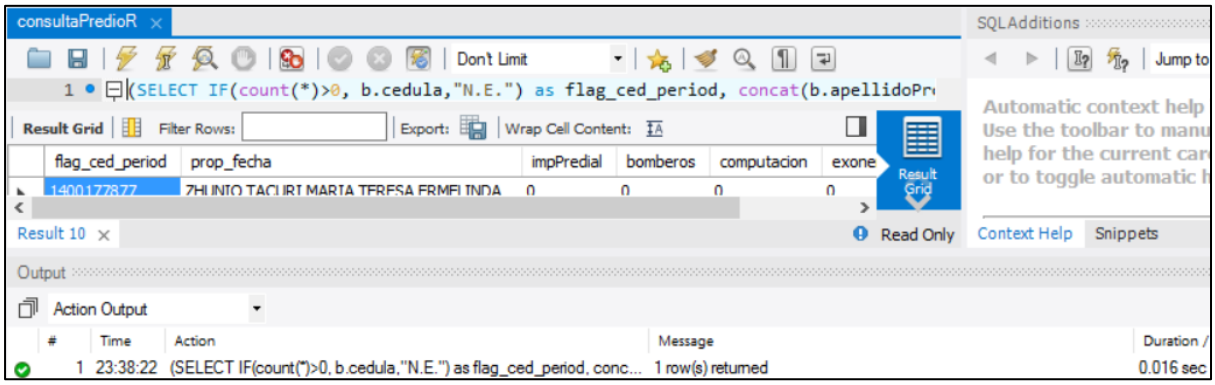


Figura 33. Tiempo de respuesta de MySQL por consultar el predio rural
Fuente: Autor

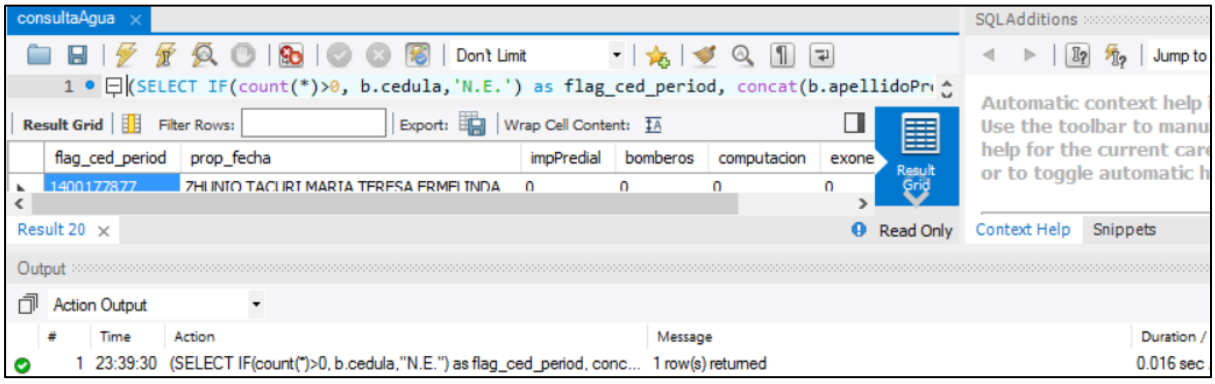


Figura 34. Tiempo de respuesta de MySQL por consultar el consumo de agua potable
Fuente: Autor

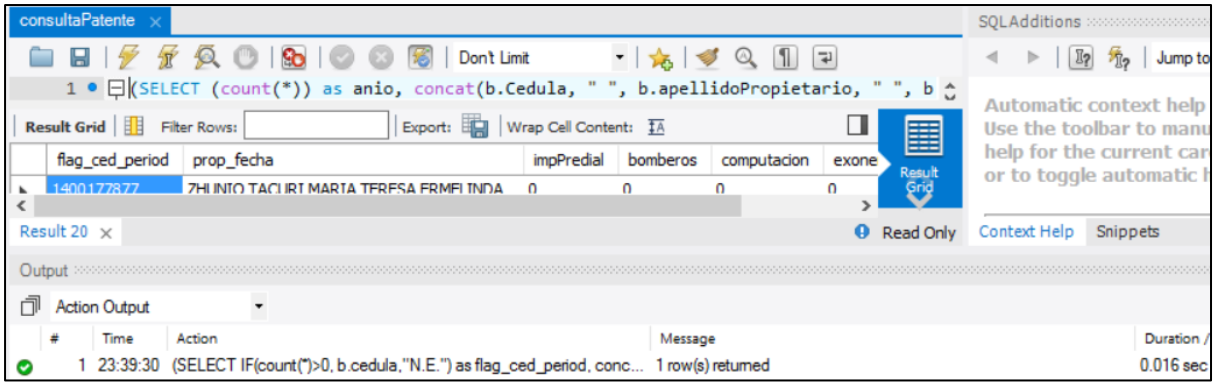


Figura 35. Tiempo de respuesta de MySQL por consultar la patente municipal
Fuente: Autor

Observando cada consulta se nota que el tiempo de respuesta es de 0.016seg en cualquiera de los 4 tributos municipales, esto se debe a que el modelo de consulta es el mismo para todos los tributos, únicamente cambia el nombre de los campos y las tablas consultadas.

Una vez conocidos los datos anteriores, se procede a realizar un análisis con en los mismos escenarios que se viene trabajando desde la tabla 39, a continuación se tiene una tabla detallada en base a las consultas:

Tabla 43. Tiempo de respuesta para atender los clientes de la base de datos SIMM

Escenario	Clientes Simultáneos	Tiempo x consulta (seg)	Conexiones soportadas x MySQL	Tiempo total requerido (seg)
Bajo	1000	0.016	300	0.05
Medio	4500	0.016	300	0.24
Alto	7000	0.016	300	0.37
Saturado	8799	0.016	300	0.47

Fuente: Autor

De lo cual se puede ver que el tiempo de respuesta del servidor es sumamente rápido, inclusive si el sistema está saturado atendimiento a los 8799 clientes simultáneamente, es capaz de responder en 0.47seg a todas las peticiones haciendo colas de espera cada 300 clientes.

2.5.4 Análisis en base al ancho de banda disponible

De acuerdo a los datos obtenidos en la sección 2.3.5.1, el ancho de banda disponible de bajada del GAD Municipal de Chordeleg es de 10.65Mbps, de lo cual utilizan un promedio de 5.57Mbps, por lo tanto se tiene disponible 10.65Mbps-5.57=5.08Mbps.

Se convierte los Mbps a KB/s para obtener el ancho de banda disponible en kilobytes por segundo, para ello se realiza lo siguiente 5.08Mbps*1000/8=635KB/s, es decir se puede recibir de internet 635KBytes en 1 segundo, con estos datos de por medio, se tiene la siguiente tabla:

Tabla 44. Clientes soportados según el ancho de banda de bajada disponible

Ancho de banda disponible (KBytes)	Tamaño x consulta (Bytes)	Clientes simultáneos que soporta = (ancho de banda/ tamaño)
635	60	10583.33

Fuente: Autor

Donde se puede observar que de acuerdo al ancho de banda disponible en el GAD Municipal de Chordeleg, el sistema puede soportar hasta 10583 clientes en 1 segundo.

De igual manera para el caso del ancho de banda de subida, se acudió a los datos obtenidos en la sección 2.3.5.1, donde el ancho de banda disponible de subida del GAD Municipal de Chordeleg es de 10.83Mbps, de lo cual utilizan un promedio de 93.4kbps, por lo tanto se tiene disponible 10.83Mbps-(93.4kbps/1000)=10.74Mbps.

Se convierte los Mbps a KB/s para obtener el ancho de banda disponible en kilobytes por segundo, para ello se realiza lo siguiente 10.74Mbps*1000/8=1324.5KB/s, es decir se puede transferir a internet 1324.5KBytes en 1 segundo, con estos datos de por medio se tiene la siguiente tabla:

Tabla 45. Clientes soportados según el ancho de banda de subida disponible

Ancho de banda disponible (KBytes)	Tamaño x consulta (Bytes)	Clientes simultáneos que soporta = (ancho de banda/ tamaño)
1324.5	314.25	4214.79

Fuente: Autor

Donde se puede observar que de acuerdo al ancho de banda disponible en el GAD Municipal de Chordeleg, el sistema puede soportar hasta 4214 clientes en 1 segundo.

2.5.5 Tabla Comparativa

2.5.5.1 Tabla

A fin de realizar un análisis del ancho de banda en diferentes escenarios, se tiene las siguientes tablas obtenidas en base a los datos estudiados dentro de la sección 2.5:

Tabla 46. Datos a recibir el web service por consultas de los tributos principales del municipio de Chordeleg

Escenario	Clientes Simultáneos	Cantidad de datos (KB)	Ancho de banda disponible (KB/s)	Tiempo de respuesta = cantidad/ancho (seg)
Bajo	1000	60.00	635	0.09
Medio	4500	270.00	635	0.43
Alto	7000	420.00	635	0.66
Saturado	8799	527.94	635	0.83

Fuente: Autor

Tabla 47. Datos a enviar el web service por consultas de los tributos principales del municipio de Chordeleg

Escenario	Clientes Simultáneos	Cantidad de datos (KB)	Ancho de banda disponible (KB/s)	Tiempo de respuesta = cantidad/ancho (seg)
Bajo	1000	310.00	1324.5	0.23
Medio	4500	1395.00	1324.5	1.05
Alto	7000	2170.00	1324.5	1.64
Saturado	8799	2727.69	1324.5	2.06

Fuente: Autor

2.5.5.2 Análisis General

Observando la tabla 46, el tiempo de respuesta del sistema depende directamente del ancho de banda de bajada disponible, tal es el caso del escenario de saturación donde se puede ver que se recibiría 527.94Kbytes en 0.83 segundos para atender a 8799 clientes, conociendo que en 1 segundo puede transferir máximo 635KBytes.

De igual manera en la tabla 47, el tiempo de respuesta del sistema depende directamente del ancho de banda de subida disponible, tal es el caso del escenario de

saturación donde se puede ver que se enviaría 2727.69KBytes en 2.06 segundos para atender a 8799 clientes, conociendo que en 1 segundo puede transferir máximo 1324.5KBytes.

2.6 Calcular, analizar y determinar el ancho de banda total requerido

2.6.1 Ancho de banda total mínimo

Para el ancho de banda de bajada, conociendo de antemano que el web service puede recibir 527.94KBytes en 0.83 segundos, resultado de disponer de un ancho de banda de bajada de 5.08Mbps, se debe tomar en cuenta que en la práctica se tendrá una afluencia simultánea del 50% de clientes, sustentado en que apenas 2551 clientes pertenecen al sector urbano donde existe el servicio de internet fijo en el cantón Chordeleg, por lo tanto se tendría un consumo de 5.08Mbps/2=2.54Mbps; a esto sumando el ancho de banda de bajada consumido por el GAD Municipal de Chordeleg que es de 5.57Mbps, se tiene un total de 8.11Mbps de ancho de banda de bajada mínimo.

Para el ancho de banda de subida, se considera que un cliente debe tener un tiempo de espera de 3 segundos por consulta cuando mucho para tener una experiencia satisfactoria al interactuar con el servicio, por lo tanto para obtener el ancho de banda mínimo que permita atender a 8799 clientes con una carga total de 2727.69KB en 3 segundos, se realiza el siguiente cálculo: (1seg*2727.69KB)/3seg=909.23KB/seg, donde al convertir el resultado en Mbps, se determina que se necesita como mínimo un ancho de banda de subida de 7.27Mbps. Así mismo se considera que en la práctica exista una afluencia simultánea del 50% de clientes, sustentado de igual forma que apenas 2551 clientes pertenecen al sector urbano donde existe el servicio de internet fijo en el cantón Chordeleg, por lo tanto se tendría un consumo de 7.27Mbps/2=3.63Mbps; a esto sumando el ancho de banda de subida consumido por el GAD Municipal de Chordeleg que es de 93.4kbps, se tiene un total de 3.72Mbps de ancho de banda de subida mínimo.

2.6.2 Ancho de banda total recomendable

Finalmente, como se había estudiado en todo el capítulo, el GAD Municipal de Chordeleg cuenta con un ancho de banda de 10Mbps simétrico; por lo tanto en el tráfico de bajada, aun así se reste el ancho de banda de bajada mínimo requerido del total contratado queda 1.89Mbps disponibles; y en el tráfico de subida aun así se reste el ancho de banda de subida mínimo requerido del total contratado queda 6.28Mbps disponibles.

Lo cual lleva a concluir que los 10Mbps contratados por el GAD Municipal de Chordeleg, es más que recomendable para implementar el proyecto.

CAPITULO III

3 Estudio, diseño y desarrollo del web service

3.1 Concepto

Un web service es un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como internet (Culturacion, s.f.).

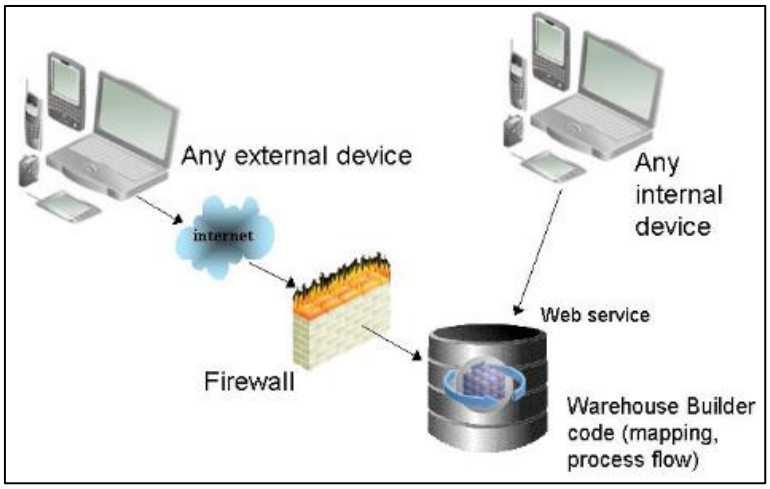


Figura 36. Esquema general de un web service
Fuente: http://culturacion.com/que-es-y-para-que-sirve-un-web-service/

En la figura anterior se puede observar claramente que al realizar peticiones al web service, tanto desde la red externa como la red interna, tomando en cuenta que para la red externa es necesario tener un firewall ya que se expone la red interna a posibles ataques desde internet.

3.2 Arquitectura

Para el desarrollo del web service, se tomará en cuenta las siguientes dos técnicas de arquitecturas:

- **SOAP (Simple Object Access Protocol):** Es un protocolo estándar que define cómo dos objetos en diferentes procesos que pueden comunicarse por medio de

intercambios de datos XML, el punto identificativo de SOAP es que las operaciones son definidas como puertos WSDL (Web Services Description Language). Es por esto que será aconsejable utilizar este protocolo en entornos donde se establecerá un contrato formal y donde se describirán todas las funciones de la interfaz así como el tipo de datos utilizados tanto de entrada como de salida. El lenguaje WSDL permitirá definir claramente cualquier detalle de las funciones del web service (QODE, 2013).

- **REST (Representational State Transfer):** Es un estilo de arquitectura de software para sistemas distribuidos tales como la web, a diferencia de SOAP, se centra en el uso de los estándares HTTP y XML para la transmisión de datos sin la necesidad de contar con una capa adicional. Las operaciones o funciones se solicitarán mediante los métodos GET, POST, PUT y DELETE, por lo que no requiere de implementaciones especiales para consumir estos servicios. Además se podrá utilizar JSON en vez de XML como contenedor de la información, por lo que será aconsejable utilizar este protocolo cuando se busque mejorar el rendimiento, o cuando se disponga de escasos recursos, como sería el caso de los dispositivos móviles (QODE, 2013).

En el presente proyecto se hará uso de la técnica de arquitectura REST, tomando en cuenta que el servicio no requiere mayor complejidad en vista que será únicamente de consulta y que está destinada para dispositivos móviles.

3.3 Protocolos

Luego de analizar la sección 3.2.1, donde se menciona que se hará uso de la técnica de arquitectura REST, se tiene el protocolo HTTP.

HTTP (Protocolo de transferencia de hipertexto) es el protocolo más utilizado en Internet, inicialmente la versión 0.9 transfería solo los datos a través de Internet, sin embargo

la versión 1.0 permite la transferencia de mensajes con encabezados que describen el contenido de los mensajes mediante la codificación MIME (CCM, s.f.).

El protocolo HTTP permite la transferencia de archivos principalmente en formato HTML, entre un cliente y un web service localizado mediante una cadena de caracteres denominada dirección URL (CCM, s.f.).

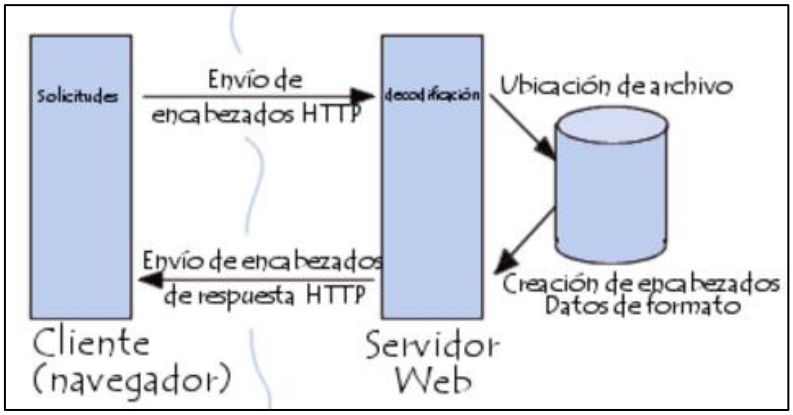


Figura 37. Comunicación entre el cliente y el servidor
Fuente: http://es.ccm.net/contents/264-el-protocolo-http

En la figura anterior se puede observar que el cliente envía su solicitud HTTP, la cual es recibida por el web service, quien a su vez la decodifica, busca la información y seguidamente envía la respuesta HTTP al cliente. A continuación se detalla tanto la solicitud como la respuesta HTTP:

- Una solicitud HTTP es un conjunto de líneas que el cliente envía al servidor, la cual incluye lo siguiente (CCM, s.f.):
  - **Una línea de solicitud:** es una línea que especifica el tipo de documento solicitado, el método que se aplicará y la versión del protocolo utilizada. La línea está formada por tres elementos que deben estar separados por un espacio: el método, la dirección URL y la versión del protocolo utilizada por el cliente (CCM, s.f.).

    En este punto se menciona a continuación los cuatro métodos HTTP comunes (CCM, s.f.):

- GET, solicita el recurso ubicado en la URL especificada.
- POST, envía datos al recurso ubicado en la URL especificada.
- PUT, envía datos a la URL especificada.
- DELETE, borra el recurso ubicado en la URL especificada.

- **Los campos del encabezado de solicitud:** es un conjunto de líneas opcionales que permiten aportar información adicional sobre la solicitud y/o el cliente. Cada una de estas líneas está formada por un nombre que describe el tipo de encabezado, seguido de dos puntos (:) y el valor del encabezado (CCM, s.f.).
- **El cuerpo de la solicitud:** es un conjunto de líneas opcionales que deben estar separadas de las líneas precedentes por una línea en blanco (CCM, s.f.).

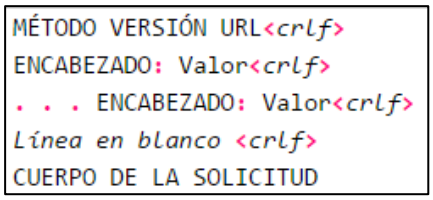


Figura 38. Sintaxis de una solicitud HTTP
Fuente: http://es.ccm.net/contents/264-el-protocolo-http

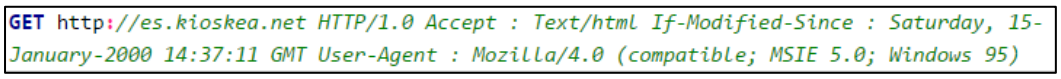


Figura 39. Ejemplo de una solicitud HTTP
Fuente: http://es.ccm.net/contents/264-el-protocolo-http

- Una respuesta HTTP es un conjunto de líneas que el servidor envía al cliente, la cual incluye lo siguiente (CCM, s.f.):
  - **Una línea de estado:** es una línea que especifica la versión del protocolo utilizada y el estado de la solicitud en proceso mediante un texto explicativo y un código. La línea está formada por tres elementos que deben estar separados por un espacio: la versión del protocolo utilizado, el código de estado y el significado del código (CCM, s.f.).

    En este punto se menciona a continuación los códigos de respuesta HTTP (ZEOKAT, 2013):

- Mensajes de información:
  - 100 – 101 Conexión rechazada.
- Mensajes de operación exitosa:
  - 200 OK
  - 201-203 Información no oficial
  - 204 Sin Contenido
  - 205 Contenido para recargar
  - 206 Contenido parcial
- Código de redirección:
  - 301 Mudado permanentemente
  - 302 Encontrado
  - 303 Vea otros
  - 304 No modificado
  - 305 Utilice un proxy
  - 307 Redirección temporal
- Error por parte del cliente
  - 400 Solicitud incorrecta
  - 401 No autorizado
  - 402 Pago requerido
  - 403 Prohibido
  - 404 No encontrado
  - 409 Conflicto
  - 410 Ya no disponible

- 412 Falló precondición

▪ Error del servidor

- 500 Error interno
- 501 No implementado
- 502 Pasarela incorrecta
- 503 Servicio no disponible
- 504 Tiempo de espera de la pasarela agotado
- 505 Versión de HTTP no soportada

o **Los campos del encabezado de respuesta:** es un conjunto de líneas opcionales que permiten aportar información adicional sobre la respuesta y/o el servidor. Cada una de estas líneas está compuesta por un nombre que califica el tipo de encabezado, seguido por dos puntos (:) y por el valor del encabezado Cada una de estas líneas está formada por un nombre que describe el tipo de encabezado, seguido de dos puntos (:) y el valor del encabezado (CCM, s.f.).

o **El cuerpo de la respuesta:** contiene los datos solicitados por el cliente (CCM, s.f.).

```
VERSIÓN-HTTP CÓDIGO EXPLICACIÓN <crlf>
ENCABEZADO: Valor<crlf>
. . . ENCABEZADO: Valor<crlf>
Línea en blanco <crlf>
CUERPO DE LA RESPUESTA
```

Figura 40. Sintaxis de una respuesta HTTP
Fuente: http://es.ccm.net/contents/264-el-protocolo-http

```
HTTP/1.0 200 OK Date: Sat, 15 Jan 2000 14:37:12 GMT Server : Microsoft-IIS/2.0 Content-Type :
text/HTML Content-Length : 1245 Last-Modified : Fri, 14 Jan 2000 08:25:13 GMT
```

Figura 41. Ejemplo de una respuesta HTTP
Fuente: http://es.ccm.net/contents/264-el-protocolo-http

3.4 Ventajas

Entre las ventajas principales investigadas acerca de una arquitectura REST, se tiene las siguientes:

- Es muy ligero, las respuestas del web service enviadas a un cliente contienen exactamente la información que se necesita (datos planos), razón por la cual utiliza pocos recursos (Mejia, 2015).
- Es una arquitectura sin estado, lo cual significa que cada petición al servidor es tratada de manera totalmente independiente (Mejia, 2015).
- Utiliza mucho más la cache en el cliente que en el servidor, pero proporcionan una buena infraestructura de almacenamiento en caché a través de HTTP método GET (para la mayoría de los servidores) (Mejia, 2015).
- Es sencillo de desarrollar y no se necesita mucho código extra, lo cual se observará en el desarrollo del código fuente del web service (Mejia, 2015).
- Es flexible para sus respuestas. estas pueden estar en formato XML o JSON, siendo este último el más popular como una alternativa excelente a XML, por lo tanto en el presente proyecto se hará uso del formato JSON (Mejia, 2015),
- Es fácil y simple de interpretar, no hay herramientas costosas que se requiera para utilizar con los webservices, dentro del mercado se puede encontrar varias alternativas de software libre (Mejia, 2015).

3.5 Estructura

En el presente proyecto se implementará el web service en un servidor independiente de la base de datos SIMM del GAD Municipal de Chordeleg, en la siguiente figura se puede observar la estructura correspondiente:

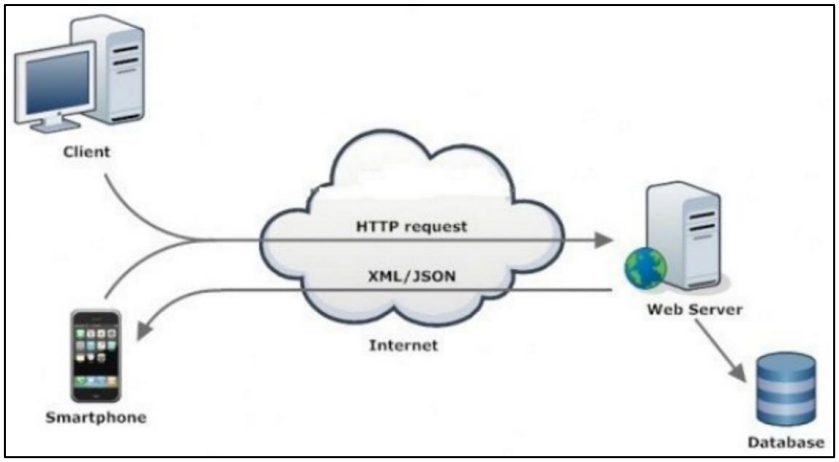


Figura 42. Estructura de un web service conectado a una base de datos
Fuente: http://es.slideshare.net/josebernam/android-con-word-press-rest-api-y-retrofit

3.6 Preparación e instalación de software en la PC de desarrollo del web service

De antemano se conoce que el GAD Municipal de Chordeleg es una institución pública, la cual se acoge al decreto presidencial Nro. 1014, donde se indica en el artículo 1 "*Establecer como política pública para las Entidades de la Administración Pública Central la utilización de Software Libre en sus sistemas y equipamientos informáticos*"; razón por la cual es necesario regirse estrictamente al uso de software libre.

3.6.1 Características de la PC de desarrollo

Con el fin de continuar a futuro mejorando y/o agregando el código fuente en el GAD Municipal de Chordeleg, se instalará en la PC de desarrollo todo el software dependiente desde cero, incluido su sistema operativo. Es así que se dispone de una PC portátil con las siguientes características:

Tabla 48. Características principales de la PC en la cual se desarrollará el proyecto

Ítem	Características
Marca	Toshiba
Modelo	Satellite P55t-B
Procesador	Core I7-4710HQ 2.5GHz, 64bits
Disco Duro	1TB
Pantalla	17"
Memoria Ram	16GB

Fuente: Autor

3.6.2 Sistema operativo

El sistema operativo que se instalará en la PC de desarrollo es Ubuntu Desktop 16.10, como la versión estable más actual al momento de desarrollar el presente proyecto. A continuación se detalla paso a paso su instalación:

- En primer lugar, se descarga Ubuntu desktop 16.10 de la página oficial de Ubuntu https://www.ubuntu.com/download, donde se encuentra las versiones para 32 y 64 bits, la PC es de 64 bits, por lo tanto se descarga la versión correspondiente.

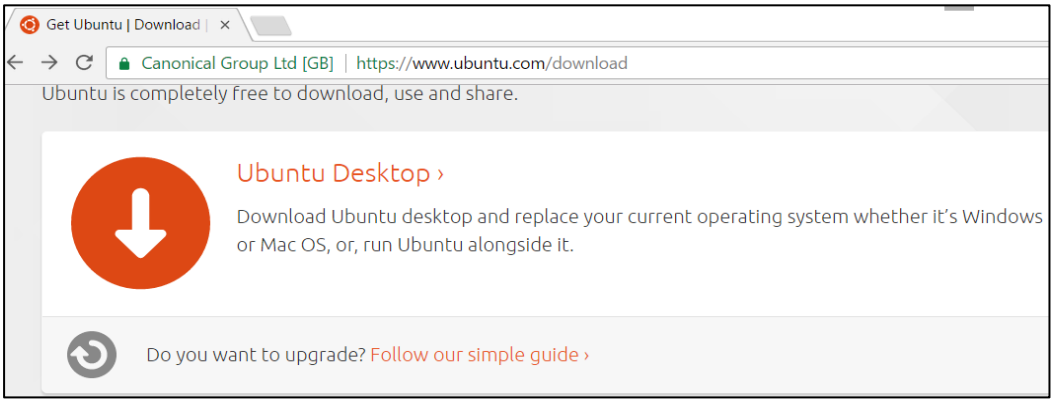


Figura 43. Página oficial de descarga de Ubuntu desktop
Fuente: Autor

- Una vez descargado el sistema operativo se graba en un DVD, se inserta el DVD en la PC y se arranca desde la unidad de CD/DVD, una vez que inicia el asistente, se da un clic en "Instalar Ubuntu":

Figura 44. Inicio del asistente de instalación de Ubuntu Desktop 16.10
Fuente: Autor

- Luego se selecciona la descarga de actualizaciones y la instalación de software de terceros como librerías extras, seguidamente se da un clic en “Continuar”.

Figura 45. Selección de actualizaciones y software de terceros para la instalación de Ubuntu Desktop 16.10
Fuente: Autor

- Se escoge el tipo de instalación, como es una instalación limpia, se selecciona la primera opción y se da un clic en “Instalar ahora”.

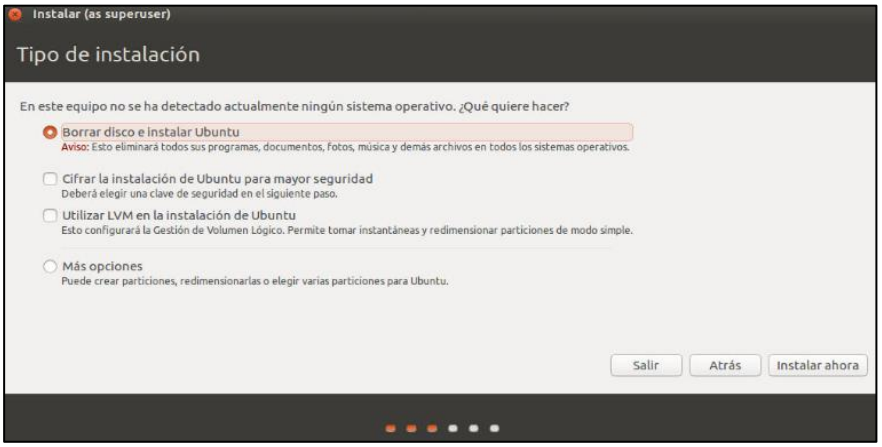


Figura 46. Selección de instalación limpia de Ubuntu Desktop 16.10
Fuente: Autor

- Se confirma la escritura en los discos dando un clic en “Continuar”.

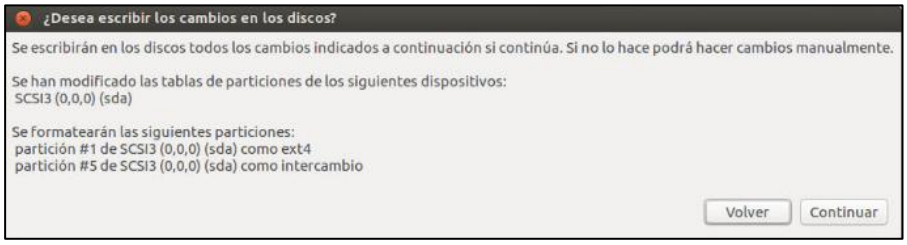


Figura 47. Mensaje de confinación de escritura de discos para instalar de Ubuntu Desktop 16.10
Fuente: Autor

- Se cambia la zona horaria que viene por defecto “Madrid” a la local que es “América/Guayaquil”.

Figura 48. Configurar zona horaria en la instalación de Ubuntu Desktop 16.10
Fuente: Autor

- Se configura el teclado en español y se da un clic en “Continuar”.

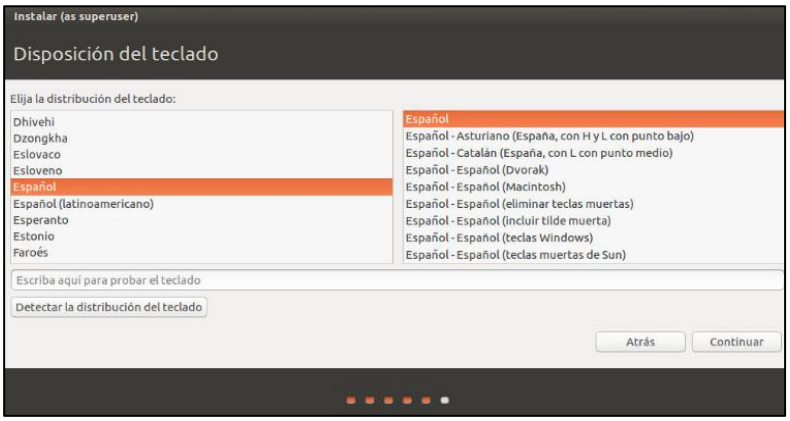


Figura 49. Configurar el teclado en español en la instalación de Ubuntu Desktop 16.10
Fuente: Autor

- Se ingresa los datos del usuario y se da un clic en “Continuar”.

Figura 50. Configurar usuario en la instalación de Ubuntu Desktop 16.10
Fuente: Autor

- Se debe esperar que termine la instalación.

Figura 51. Instalando Ubuntu Desktop 16.10
Fuente: Autor

- Una vez que termine de instalar, se da un clic en “Reiniciar ahora”.

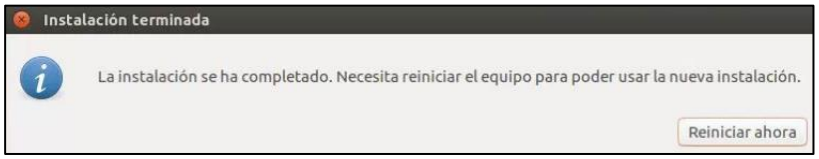


Figura 52. Instalación terminada de Ubuntu Desktop 16.10
Fuente: Autor

- Finalmente luego de reiniciar, se ingresa la contraseña, se da un enter y ya inicia la pantalla de principal de Ubuntu Desktop 16.10.

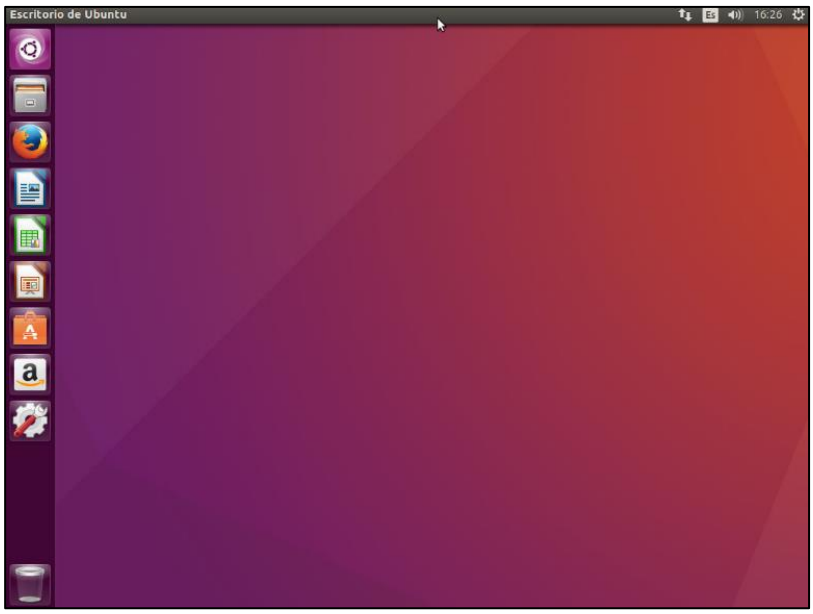


Figura 53. Instalación terminada de Ubuntu Desktop 16.10
Fuente: Autor

3.6.3 Servidor de base de datos

Si bien es cierto, el servidor de la base de datos ya está implementado en el GAD Municipal de Chordeleg, el cual se había mencionado en el capítulo 1 que es MySQL Server en su versión 5.1.69; sin embargo es necesario instalarlo en la PC de desarrollo, con el fin de realizar todas las pruebas a nivel local y sobre todo que no sea necesario estar conectado a la red LAN del GAD Municipal de Chordeleg. A continuación se detalla su instalación:

- Por temas de compatibilidad, se instalará la misma versión del servidor de base de datos del GAD Municipal de Chordeleg, para ello se dirige a la página oficial de MySQL https://dev.mysql.com/downloads/ y se descarga MySQL Server 5.1.69; seguidamente se copia el archivo descargado en la dirección /usr/local de la PC de desarrollo. Posterior a ello abrir la terminal de Ubuntu, ubicarse en la ruta donde se copió el archivo y se porcede a descomprimirlo.

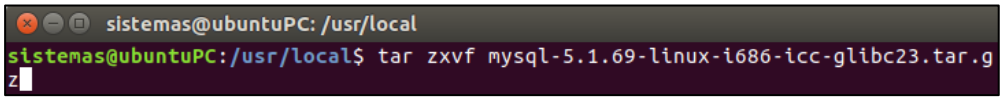


Figura 54. Descomprimir MySQL Server 5.1.69
Fuente: Autor

- Se procede a crear el enlace simbólico para poder iniciar desde una ruta conocida.

Figura 55. Creación de un enlace simbólico para MySQL Server
Fuente: Autor

- Ahora se crea el usuario y grupo para MySQL.

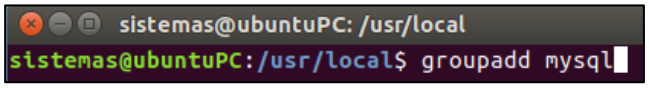


Figura 56. Creación del grupo de MySQL
Fuente: Autor

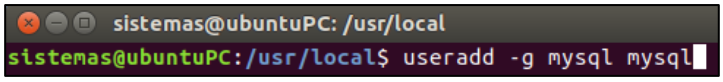


Figura 57. Creación del usuario MySQL
Fuente: Autor

- Ubicarse en la ruta /usr/local/mysql y proceder con la instalación de MySQL Server ejecutando el siguiente comando.

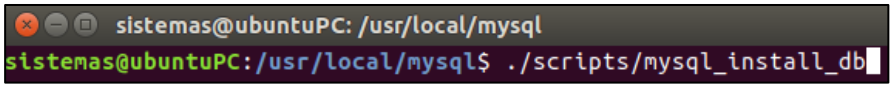


Figura 58. Instalación de MySQL Server en Ubuntu
Fuente: Autor

- Se configura la contraseña de MySQL.

Figura 59. Establecer una contraseña para MySQL Server
Fuente: Autor

- Finalmente se inicia el servicio con el siguiente comando.

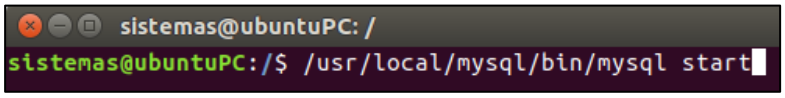


Figura 60. Iniciar MySQL Server en Ubuntu
Fuente: Autor

3.6.4 Administrador de la base de datos

Se conoce que la base de datos es MySQL Server, por lo tanto como administrador de la base de datos se tiene la herramienta MySQL Workbench 6.3.6, que es la versión más actual al momento de realizar este proyecto, a continuación se detalla su instalación paso a paso en Ubuntu:

- Se procede a descargar e instala las dependencias de Ubuntu para instalar MySQL Workbench.

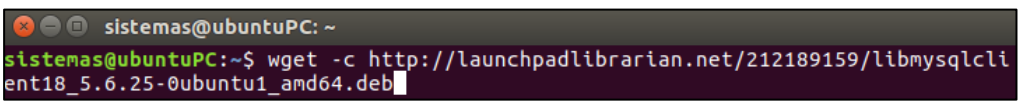


Figura 61. Descargar la librería libmysqlclient
Fuente: Autor

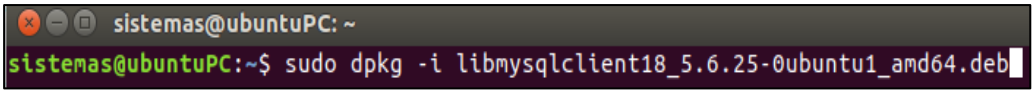


Figura 62. Instalar la librería libmysqlclient
Fuente: Autor

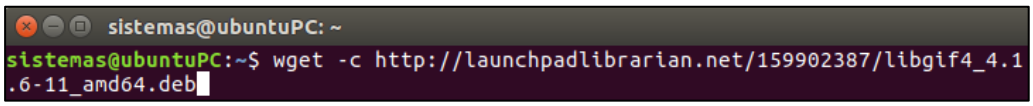


Figura 63. Descargar la librería libgif4
Fuente: Autor

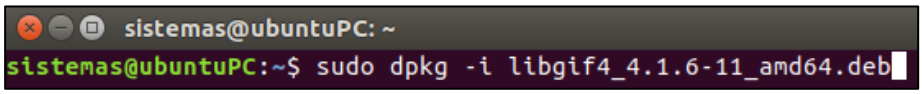


Figura 64. Instalar la librería libgif4
Fuente: Autor

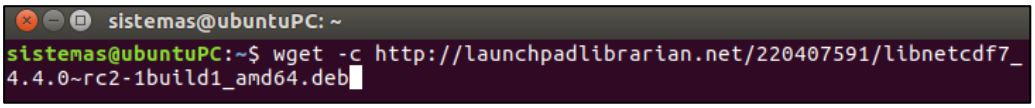


Figura 65. Descargar la librería libnetcdf7
Fuente: Autor

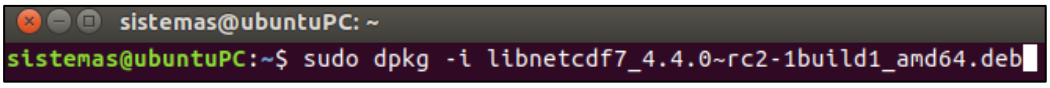


Figura 66. Instalar la librería libnetcdf7
Fuente: Autor

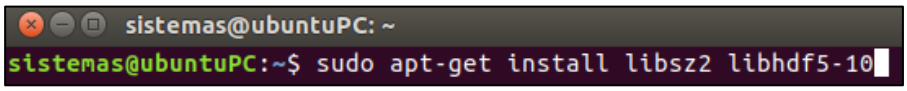


Figura 67. Instalación de librerías libsz2 y libhdf5-1 del repositorio de Ubuntu
Fuente: Autor

- Una vez instaladas las dependencias, se procede a descargar MySQL Workbench desde su página oficial https://dev.mysql.com/downloads/workbench/ y posterior a ello se descomprime el paquete.

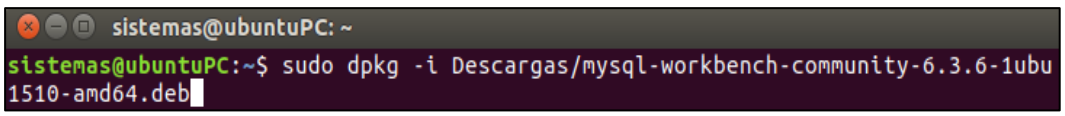


Figura 68. Descomprimir MySQL Workbench 6.3.6
Fuente: Autor

- Finalmente se procede con la instalación de MySQL Workbench.

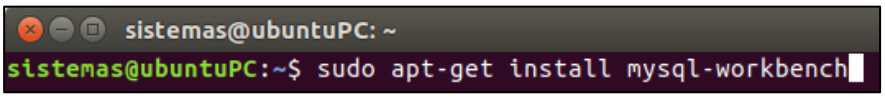


Figura 69. Instalación de MySQL Workbench 6.3.6
Fuente: Autor

- Para acceder a MySQL Workbench, se presiona Alt+F2 y se digita mysql-workbench, luego se da un clic en la aplicación y listo.

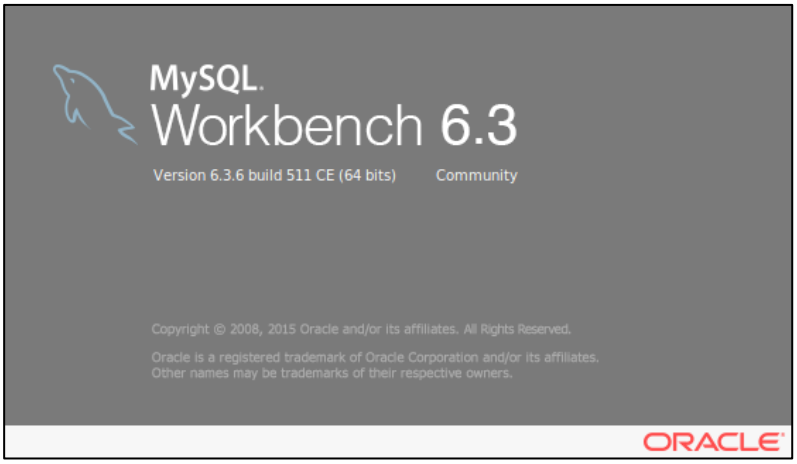


Figura 70. Acerca de MySQL Workbench
Fuente: Autor

3.6.5 Lenguajes de programación

Antes que nada se debe conocer que cada lenguaje de programación, tiene sus pros y sus contras, por lo tanto se tiene que enfocar a utilizar el que más se ajuste a cada necesidad. Entre los principales lenguajes de programación libre que permiten implementar un web service se tienen los siguientes:

3.6.5.1 PHP

Figura 71. Logo del lenguaje de programación PHP
Fuente: https://www.quora.com/What-font-is-the-php-logo

PHP es uno de los lenguajes más populares de programación web, es conocido como un lenguaje basado en servidores. Esto es porque el PHP no se ejecuta en el cliente, sino en el lado del servidor. Las peticiones la realizan los clientes, luego la solicitud es procesada en el servidor y finalmente se envía la respuesta al cliente (Luiggi, 2014).

Entre las características principales se tiene las siguientes:

- **Código abierto**: esto significa que está disponible completamente gratis (Luiggi, 2014).
- **Multiplataforma:** es decir permite operar en varios sistemas operativos tales como LINUX, UNIX y Windows (Luiggi, 2014).
- **Soporte:** al ser un programa muy popular, existe una comunidad muy grande, con referencias, guías, foros, entre otros disponibles en la web (Luiggi, 2014).
- **Altos retornos:** Permite crear páginas web dinámicas. Esto asegura mayor participación de los visitantes y por lo tanto mayores retornos (Luiggi, 2014).
- **Extensiones:** tiene múltiples extensiones y es extremadamente escalable (Luiggi, 2014).
- **Confiable y Seguro:** El código fuente escrito en PHP es invisible al navegador y al cliente ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador (Martinez, 2011).

PHP se puede ejecutar en el conocido servidor web Apache, en el cual se tienen que instalar las dependencias correspondientes.

3.6.5.2 Python

Figura 72. Logo del lenguaje de programación Python
Fuente: https://underc0de.org/foro/python/introduccion-a-python/

Python es un lenguaje de programación interpretado de alto nivel creado por Guido Van Rossum a fines de los años ochenta (WhiZ, 2014).

Entre sus características principales, se puede nombrar las siguientes:

- **Simple:** Python fue diseñado con esta filosofía: legibilidad y transparencia (Abernethy, 2014).
- **Interpretado:** Todo lo escrito en python será ejecutado por medio de un intérprete, a diferencia del lenguajes compilados (C, C++, C#, Visual Basic, Delphi, etc.) que son ejecutados por medio de un compilador (Abernethy, 2014).
- **Multiplataforma:** Se puede ejecutar los scripts en cualquier plataforma, tales como GNU/Linux, Microsoft Windows o Mac OSX (Abernethy, 2014).
- **Interactivo:** Se puede ejecutar sentencias desde la línea de comandos y ver qué responde el intérprete (es de gran ayuda para comprender mejor el lenguaje y, principalmente, al momento de programar, permitiendo probar segmentos específicos del código en desarrollo, en busca de errores) (Abernethy, 2014).
- **Extensiones:** Implementa una gran cantidad de bibliotecas disponibles en la web (Abernethy, 2014).

Python se puede ejecutar al igual que PHP, en un servidor web Apache, para ello se instalan las dependencias correspondientes.

3.6.5.3 JavaScript

Figura 73. Logo del lenguaje de programación JavaScript
Fuente: http://www.microsiervos.com/archivo/ordenadores/javascript-de-hoy-en-dia.html

JavaScript es uno de los lenguajes de programación más utilizados y conocidos, ya que permite crear páginas dinámicas y llamativas en las que se puede interactuar más con los usuarios. Cabe resaltar que JavaScript inicialmente se ejecutaba en el cliente, sin embargo gracias al proyecto Node.js es posible ejecutarlo en el servidor (Pao, s.f).

A continuación se puede observar las ventajas más destacadas de JavaScript:

- Es un lenguaje muy sencillo (Pao, s.f).
- Es rápido, por lo tanto tiende a ejecutar las funciones inmediatamente (Pao, s.f).
- Cuenta con múltiples opciones de efectos visuales (Pao, s.f).
- Es soportado por los navegadores más populares y es compatible con los más modernos, incluyendo iPhone, móviles y PS3 (Pao, s.f).
- Es muy versátil, puesto que es muy útil para desarrollar páginas dinámicas y aplicaciones web (Pao, s.f).
- Es una buena solución para poner en práctica la validación de datos en un formulario (Pao, s.f).
- Es multiplataforma, puede ser ejecutado de manera híbrida en cualquier sistema operativo móvil (Pao, s.f).

3.6.5.4 Tabla comparativa y selección del lenguaje de programación

En la siguiente tabla se menciona las características principales de interés:

Tabla 49. Lenguaje de programación PHP, Python y JavaScript

Item	PHP	Python	JavaScript
Creador	Rasmus Lerdorf	Guido Van Rossum	Brendan Eich
Año de lanzamiento	1994	1991	1995
Multiplataforma	Si	Si	Si
Código Libre	Si	Si	Si
Soporte en internet	Si	Si	Si
Escalable, flexible y rápido	Si	Si	Si
Programación de lado del servidor	Si	Si	Si

Fuente: Autor

Como se puede observar en la tabla anterior, para este caso es posible hacer uso de cualquiera de los 3 lenguajes de programación, sin embargo se resalta que hoy en día JavaScript ha logrado un gran crecimiento desde que es posible trabajar del lado del servidor gracias al proyecto Node.js creado en el año 2009 por Ryan Dahl.

Node.js un intérprete JavaScript del lado del servidor que cambia la noción de cómo debería trabajar un servidor. Su meta es permitir a un programador construir aplicaciones altamente escalables y escribir código que maneje decenas de miles de conexiones simultáneas en una sólo una máquina física (Abernethy, 2011).

Utilizando un servidor web tradicional, como pueda ser Apache, cada vez que se solicita un recurso web, se crea un hilo separado, o invoca un nuevo proceso, para atender dicha solicitud, a pesar de que Apache responde rápidamente a las solicitudes, y limpia el proceso una vez que ha terminado, este sistema puede necesitar varios recursos, por decir una aplicación web con muchas visitas, puede tener serios problemas de rendimiento. Node.js, por el contrario, utiliza un sistema de E/S asíncrono y basado en eventos y callbacks de funciones. Es decir, todo lo realiza en un único hilo: se queda escuchando a ciertos eventos que ocurran en la aplicación, y cuando ocurren, responde en consecuencia, de modo asíncrono, es decir, un evento no bloquea a otro, dicho de otro modo, crea miles de subprocesos y trata a sus outputs como streams, esto supone una mayor velocidad de respuesta (Martín de Pozuelo, 2013).

Por lo tanto de acuerdo a lo investigado, se incluna por desarrollar el código del web service en JavaScript y como intérprete del mismo s tendría a Node.js por las grandes bondades que presta para realizar consultas simultáneas al servidor.

3.6.5.5 Editor de código

En la sección anterior se definió trabajar con JavaSript como lenguaje de programación, por lo tanto para el desarrollo del código fuente del web service, se utilizará el

software Atom, el cual es libre, gratuito y de gran popularidad en la red, su instalación se realiza de la siguiente manera.

- Antes de iniciar la instalación, se agrega el repositorio de instalación de Atom.

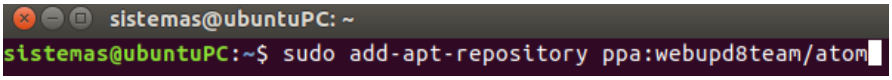


Figura 74. Agregar el repositorio para instalar Atom en Ubuntu
Fuente: Autor

- Luego se actualiza la lista de paquetes de Ubuntu.

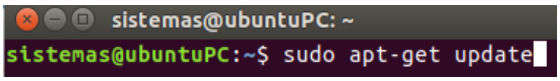


Figura 75. Actualizar la lista de paquetes de Ubuntu para instalar Atom
Fuente: Autor

- Por último se procede a instalar Atom en Ubuntu.

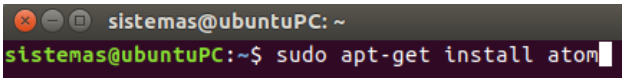


Figura 76. Instalación de Atom en Ubuntu
Fuente: Autor

- Para acceder a Atom, se presiona Alt+F2 y se digita Atom, luego se da un clic en la aplicación y listo.

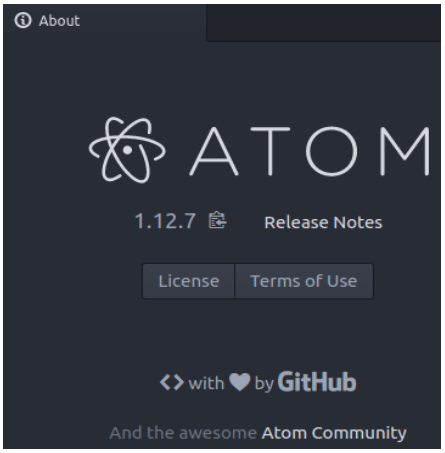


Figura 77. Acerca de Atom en Ubuntu
Fuente: Autor

3.6.6 Servidor de aplicaciones

En vista que el web service se desarrollará en JavaSript como lenguaje de programación, se había mencionado el uso de Node.js como intérprete del mismo, por lo

tanto se procede a instalar paso a paso el Node.js como servidor de aplicaciones en Ubuntu Desktop 16.10.

- Se abre la terminal de Ubuntu presionando Ctrl+Alt+T.

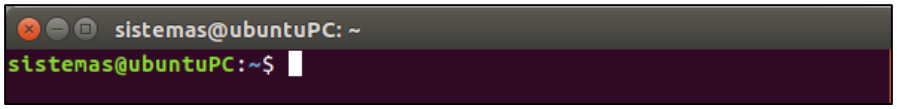


Figura 78. Terminal de Ubuntu
Fuente: Autor

- Se actualiza el repositorio de Ubuntu.

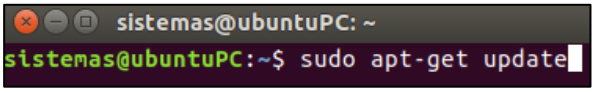


Figura 79. Actualizar el repositorio de Ubuntu
Fuente: Autor

- Se instala las librerías dependientes que requiere Node.js.

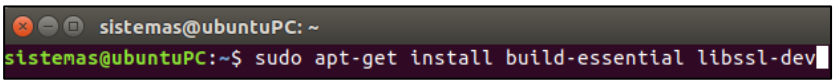


Figura 80. Librerías para instalar Node.js en Ubuntu
Fuente: Autor

- Se instala NVM, el cual sirve como gestor para instalar Node.js.

Figura 81. Instalar NVM como gestor de la instalación de Node.js
Fuente: Autor

- Se procede a instalar Node.js en su versión estable 6.9.1, que es la más actual al momento de realizar el presente proyecto.

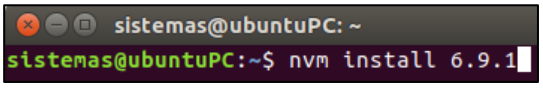


Figura 82. Instalar Node.js en Ubuntu
Fuente: Autor

- Finalmente se puede comprobar la versión instalada realizando lo siguiente:

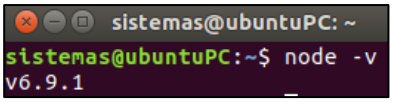


Figura 83. Comprobar la versión de Node.js instalada en Ubuntu
Fuente: Autor

En este punto es necesario indicar que Node.js se instala conjuntamente con el gestor de paquetes "NPM", el cual sirve para instalar y agregar módulos adicionales a Node.js, de igual manera se puede comprobar su versión instalada realizando lo siguiente:

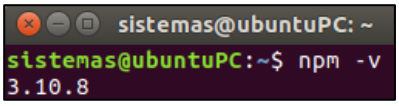


Figura 84. Comprobar la versión de NPM, gestor de paquetes de Node.js
Fuente: Autor

3.7 Desarrollo del web service

3.7.1 Crear la base de datos

Para la creación de la base de datos, simplemente se realizó un respaldo de la base de datos SIMM del GAD Municipal de Chordeleg y se restauró en el servidor local de la PC de desarrollo. Parar ello se realizó los siguientes pasos:

- Haciendo uso del administrador de base de datos MySQL WorkBench, se crea la conexión a la base de datos del GAD Municipal de Chordeleg.

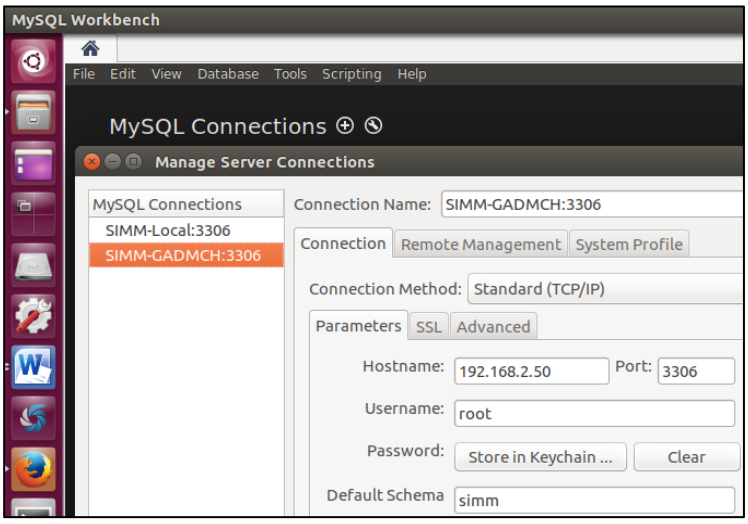


Figura 85. Crear la conexión al servidor de la base de datos SIMM remota
Fuente: Autor

- Luego de crear la conexión al servidor de base de datos SIMM remota, se inicia la conexión y se dirige a “Server/Data Export”; se selecciona la base de datos SIMM y se da un clic en “Start Export” para iniciar el respaldo.

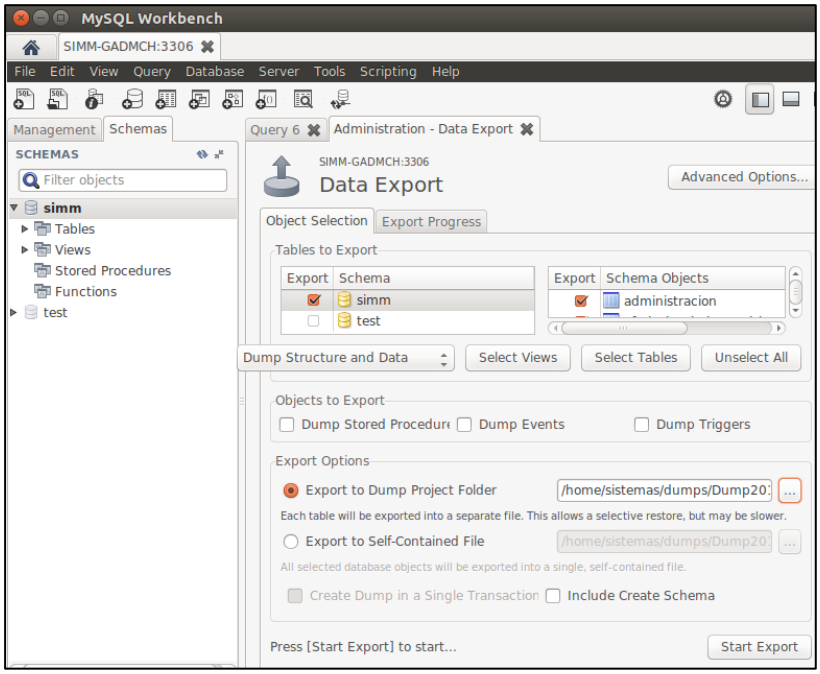


Figura 86. Respaldo de la base de datos SIMM remota
Fuente: Autor

- De igual manera haciendo uso del administrador de base de datos MySQL Workbench, se crea la conexión a la base de datos de la PC de desarrollo.

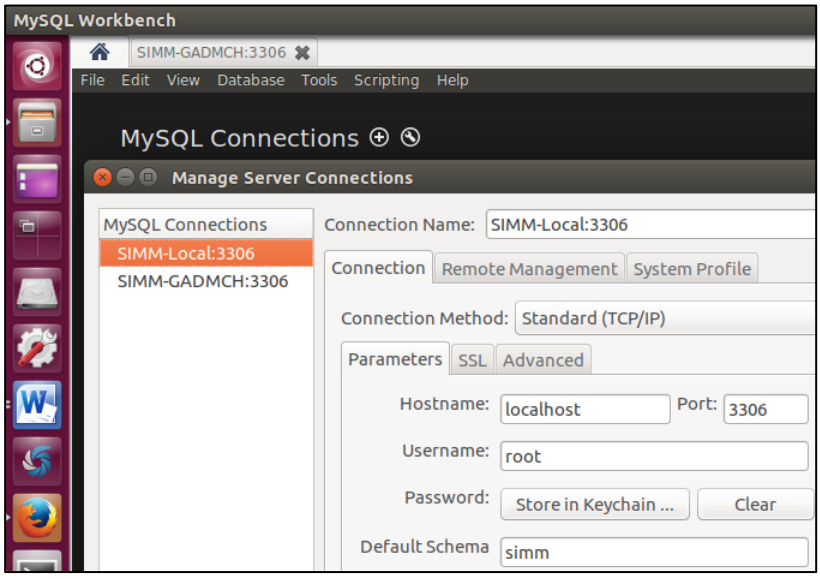


Figura 87. Crear la conexión al servidor de la base de datos local
Fuente: Autor

- Luego de crear la conexión al servidor de base de datos SIMM local, se inicia la conexión y se dirige a “Server/Data Import”; se selecciona la base de datos del archivo respaldado y se da un clic en “Start Import” para iniciar la restauración.

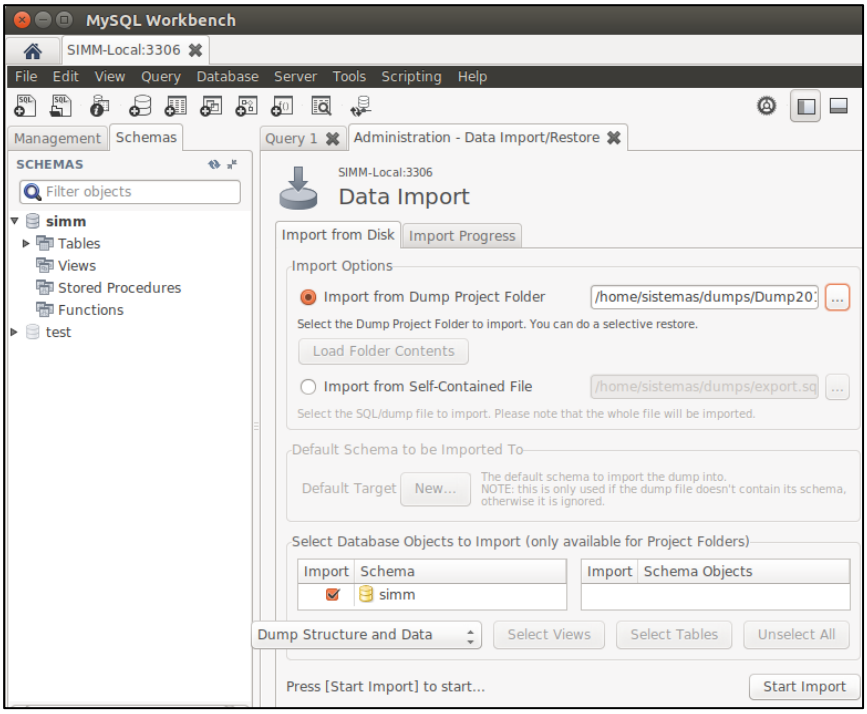


Figura 88. Restauración de la base de datos SIMM local
Fuente: Autor

3.7.2 Crear el proyecto

Para la creación del proyecto, es necesaria una carpeta principal, en la que se incluye los archivos dependientes para la ejecución del web service, a continuación detallo lo mencionado:

- Crear una carpeta principal en la cual se va a guardar todos los archivos correspondientes al código fuente en desarrollo del web service; a dicha carpeta se la llamará “codigoApp” y se ubicará en la ruta “Escritorio/TESIS/NodeJs/” de la PC de desarrollo.

Figura 89. Creación de la carpeta principal del código fuente del web service
Fuente: Autor

- Haciendo uso del editor de código Atom, se crea un archivo de extensión .js, en el cual se desarrollará el código principal JavaScript para el web service y a su vez será el archivo que se ejecutará al final para iniciar el servicio, a este archivo se nombra como “app.js”.

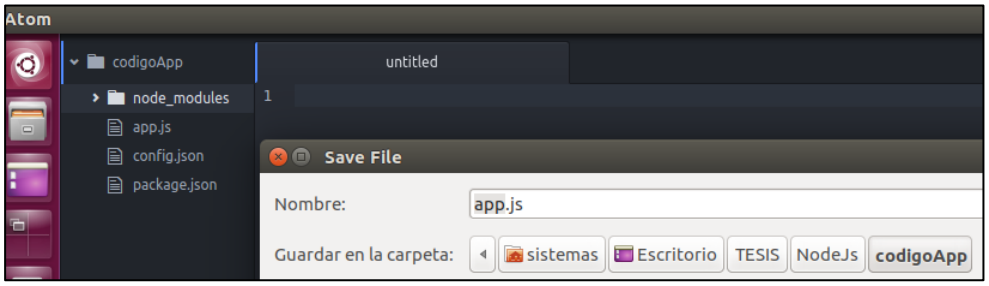


Figura 90. Creación del archivo principal del web service
Fuente: Autor

- De igual manera haciendo uso del editor de código Atom, se crea un archivo de extensión .json, en el cual se define los módulos necesarios para el desarrollo del código principal del web service, este archivo se llamará package.json.

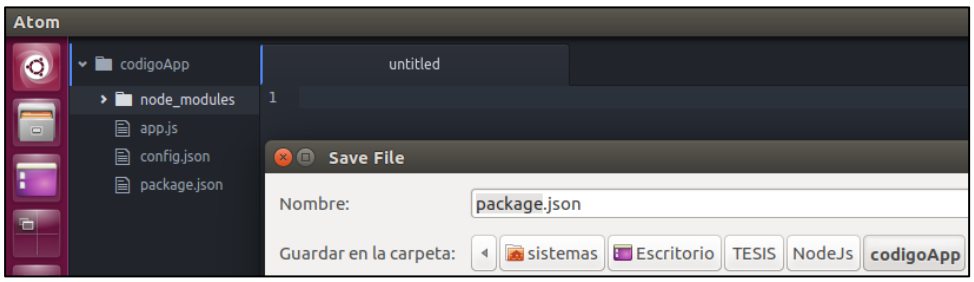


Figura 91. Creación del archivo de módulos a instalar en el web service
Fuente: Autor

- Por último haciendo uso del mismo editor de código Atom, se crea otro archivo de extensión .json, en el cual se configura los datos de conexión del web service y de la base de datos, este archivo se llamará config.json.

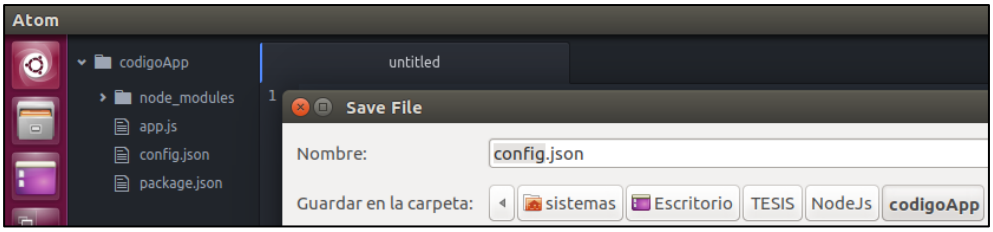


Figura 92. Creación del archivo de configuración del web service
Fuente: Autor

3.7.3 Instalar módulos para la arquitectura REST y la base de datos MySQL.

En el servidor de aplicaciones Node.js, se puede agregar una diversidad de módulos disponibles en internet y que se ajusten a cada una de las necesidades respectivas; por lo tanto en el desarrollo del servidor se hará uso del módulo “restify” para crear el web service en base a la arquitectura REST, lo cual se había definido en la sección 3.2; y el módulo “mysql” para la conexión a la base de datos SIMM. A continuación se define los pasos necesarios para agregar estos módulos al presente proyecto:

- Primeramente se tiene que configurar el archivo package.json con los módulos que se desea instalar, para ello se debe editar el archivo con Atom y asignar el nombre respectivo:

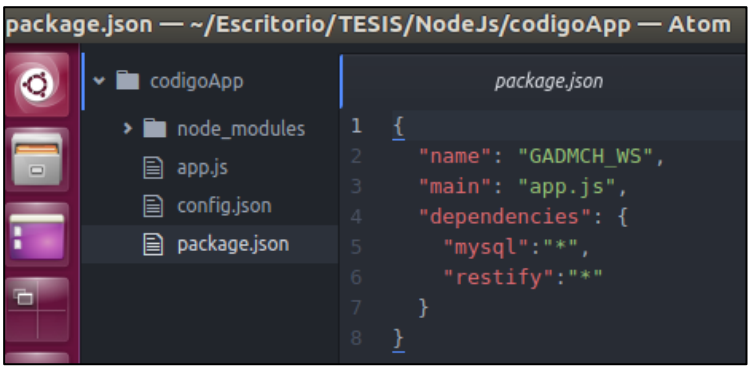


Figura 93. Configuración del archivo package.json
Fuente: Autor

- Una vez configurado el archivo package.json, se abre la terminal de Ubuntu, se ubica en la ruta correspondiente y mediante NPM que es el gestor de paquetes de Node.js, se instala las dependencias.

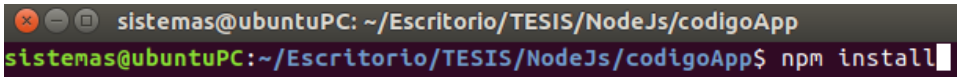


Figura 94. Instalación de dependencias en Node.js
Fuente: Autor

- Luego de instalar las dependencias, se puede observar que en la carpeta principal del proyecto, se ha creado una carpeta llamada "node_modules", dentro de ella se encuentra una carpeta de nombre restify la cual contiene el módulo que permite crear el web service con la arquitectura Rest, y otra carpeta de nombre mysql que así mismo contiene el módulo que permite la conexión del web service con la base de datos MySQL.

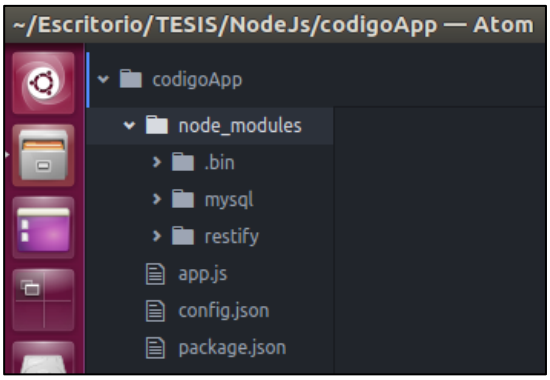


Figura 95. Módulos instalados en Node.js
Fuente: Autor

3.7.4 Conexión al web service y el enlace a la base de datos

Para la ejecución del web service es necesario definir una ip y un puerto de operación, por lo tanto al estar en un entorno local la ip será "localhost" y el para el puerto de operación se tiene el 3001 que a su vez está libre en la red del GAD Municipal de Chordeleg. En cuanto al enlace de la base de datos, recordando que se instaló un servidor local, se tiene la ip configurada como "localhost", el usuario por defecto "root" y la base de datos "simm".

Una vez definido los parámetros de configuración, se procede a editar el archivo config.json del proyecto, mediante Atom, donde se tiene lo siguiente:

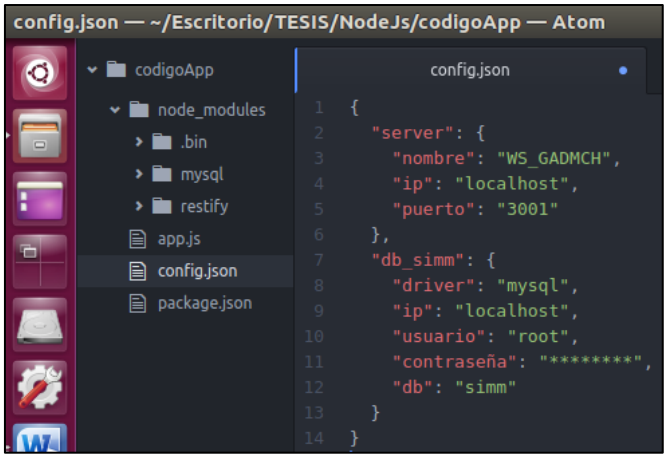


Figura 96. Configuración del archivo config.json
Fuente: Autor

3.7.5 Código fuente desarrollado

El código fuente completamente desarrollado se puede revisar en la sección de Apéndices, sin embargo a continuación se describe de manera general el código desarrollado:

- En la sección 3.7.2 se crea el archivo app.js, para el desarrollo del código fuente, por lo tanto haciendo uso de Atom, se abre el archivo en mención y lo primero que se realizó fue incluir las dependencias para crear el servidor rest y la conexión a la base de datos mysql, a más de ello se incluyó las configuraciones del archivo config.json donde se encuentran los datos del web server y la base de datos.

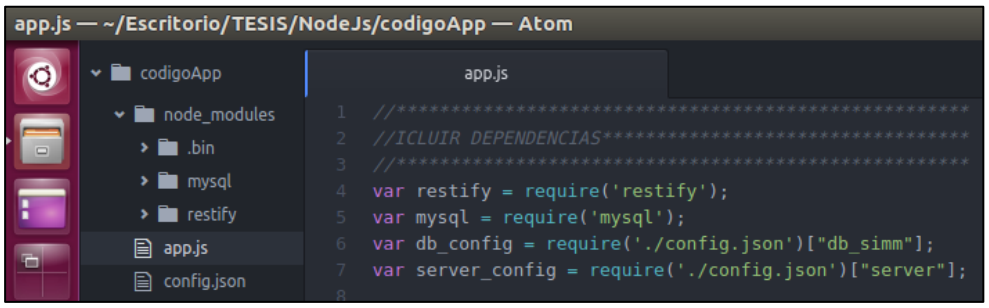


Figura 97. Incluir dependencias en app.js
Fuente: Autor

- Posteriormente, con los datos de configuración y dependencias obtenidos, se configuró la conexión a la base de datos.

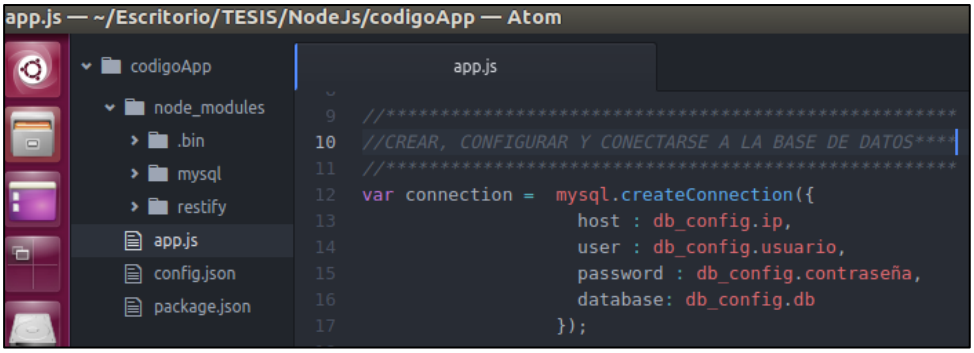


Figura 98. Conexión a la base de datos desde el web service
Fuente: Autor

- Así mismo con los datos de configuración y dependencias obtenidos, se creó el arranque del web server.

Figura 99. Código que arranca el web service
Fuente: Autor

- Luego de configurar el enlace a la base de datos y el web server, se crean las rutas a las cuales tiene que acceder el cliente para realizar las peticiones de la información y es aquí donde se incluye el comando GET del protocolo HTTP para consultar al web service, en este caso se define las rutas para consultar los principales tributos municipales, los cuales son predio urbano, predio rural, agua potable y patente municipal.

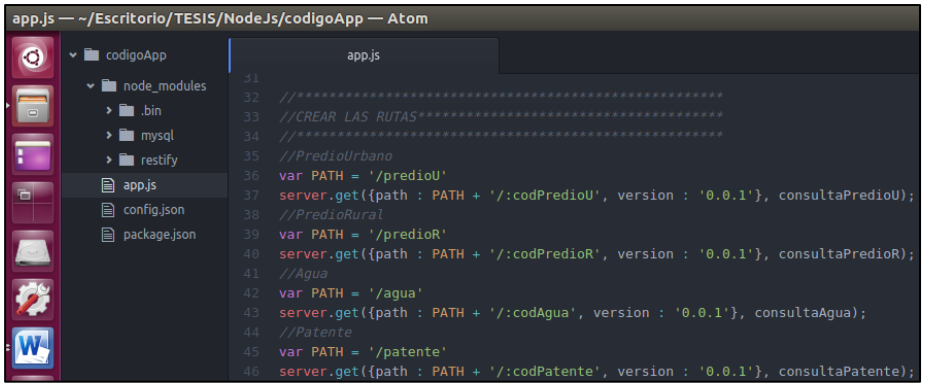


Figura 100. Rutas de consulta del web service
Fuente: Autor

- Finalmente en la figura anterior se puede observar que para el predio urbano se tiene la función "consultaPreioU", para el predio rural se tiene la función "consultaPreioR", para el agua potable se tiene la función "consultaAgua" y para la patente municipal se tiene la función "consultaPatente"; cada función recibe como parámetro de entrada el código correspondiente dependiendo del tributo que se esté consultando, y esta a su vez procesa la información solicitada, consulta a la base de datos SIMM y devuelve la respuesta correspondiente. En las siguientes figuras se puede observar parte del código de cada función, recordando nuevamente que el código completo se encuentra en la sección de Apéndices.

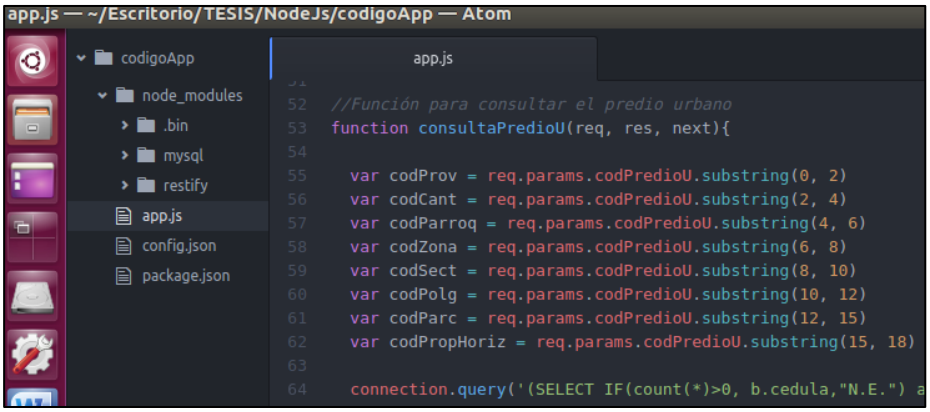


Figura 101. Función de consulta del predio urbano del web service
Fuente: Autor

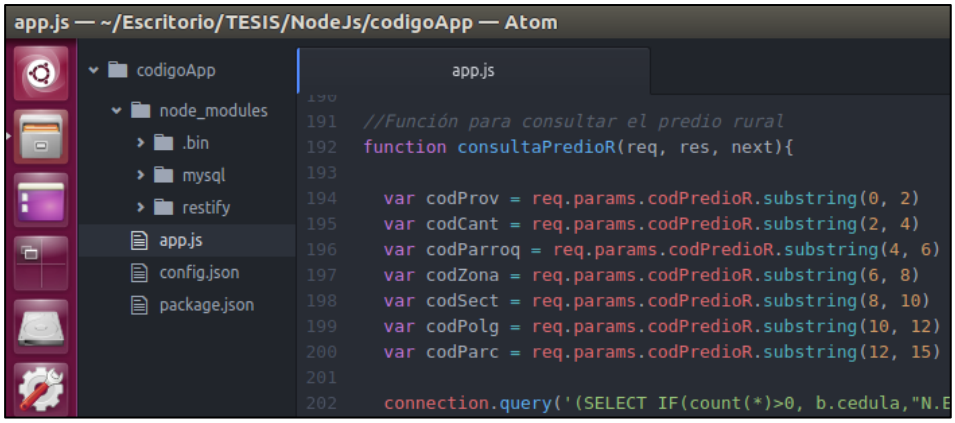


Figura 102. Función de consulta del predio rural del web service
Fuente: Autor

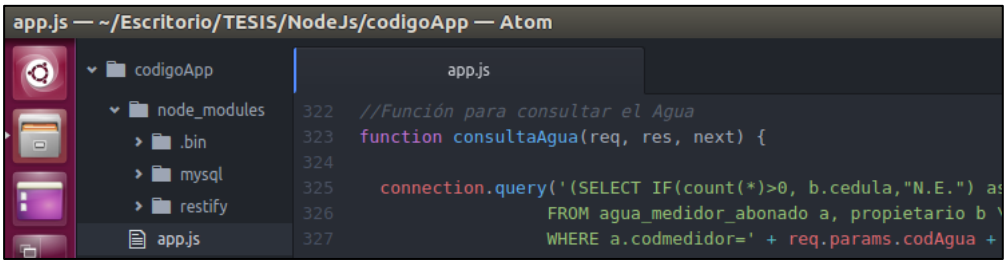


Figura 103. Función de consulta del agua potable del web service
Fuente: Autor

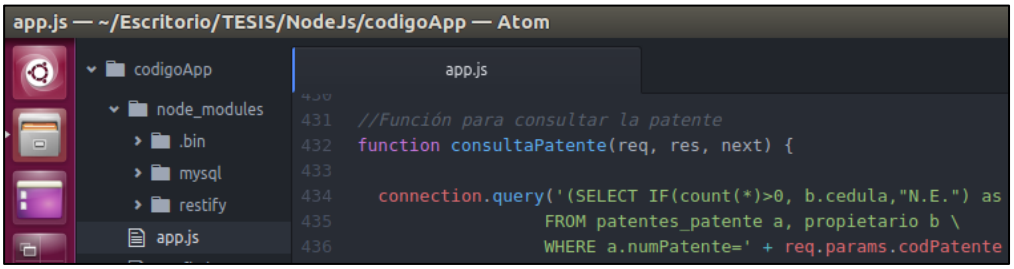


Figura 104. Función de consulta de la patente municipal del web service
Fuente: Autor

3.7.6 Pruebas del web service

Se realizaron pruebas de inicialización del web service y las consultas por cada tributo municipal, a continuación se detalla lo mencionado:

- La primera prueba realizada fue la de iniciar el web service, sabiendo de antemano que Node.js es el intérprete del código desarrollado para el web service, se abre la terminal de Ubuntu e inicia la aplicación con el comando “node + la ruta del archivo app.js del proyecto”; donde se obtuvo lo siguiente:

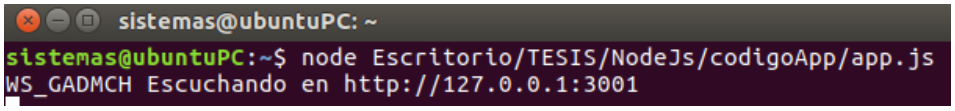


Figura 105. Arranque del web service
Fuente: Autor

- Una vez iniciado el web service, se comienza por realizar la consulta del predio urbano, debido a que aún no se tiene la app android desarrollada, se hará uso de la librería CURL que viene por defecto instalada en Ubuntu, dicha librería permitirá realizar peticiones url al servidor a través de la terminal de Ubuntu; a continuación en las siguientes figuras, se tiene una consulta por cada tributo principal del GAD Municipal de Chordeleg:

Figura 106. Consulta al predio urbano mediante CURL
Fuente: Autor

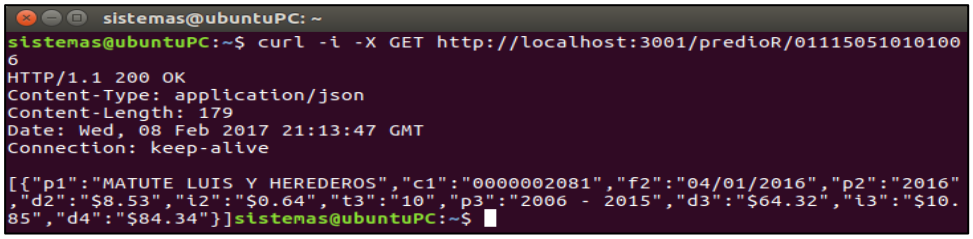


Figura 107. Consulta al predio rural mediante CURL
Fuente: Autor

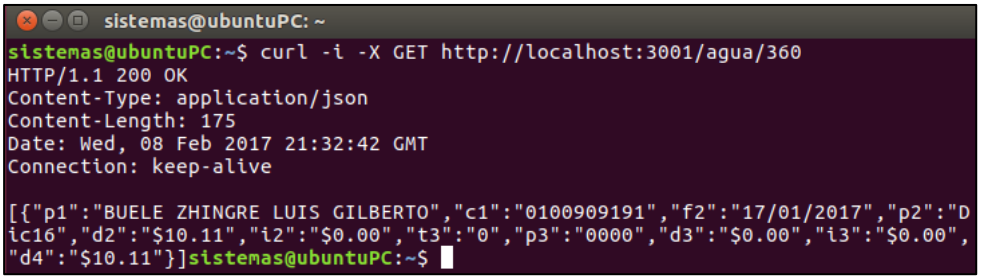


Figura 108. Consulta del agua potable mediante CURL
Fuente: Autor

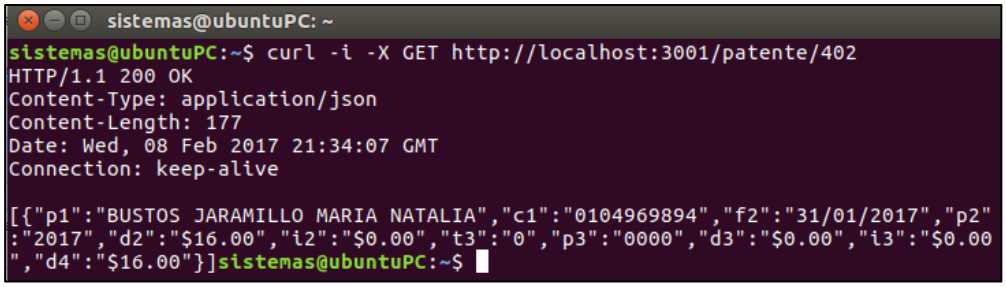


Figura 109. Consulta de la patente municipal mediante CURL
Fuente: Autor

Las pruebas presentadas en las figuras anteriores, se ejecutaron de manera exitosa, luego de revisar, analizar y realizar varias correcciones en el código fuente desarrollado.

3.7.7 Generar el ejecutable del proyecto

El ejecutable del web service no es más que la carpeta principal del proyecto "codigoApp" incluida todas las subcarpetas y archivos; para luego al momento de implementar el servidor de producción se procede a copiar la carpeta en cualquier ruta que se desee.

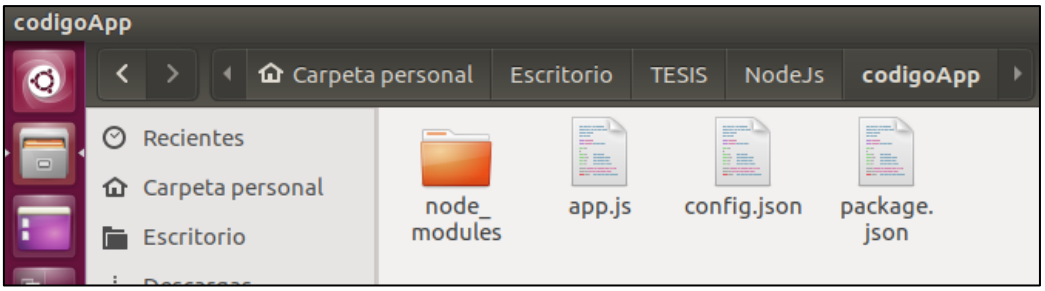


Figura 110. Carpeta de los archivos desarrollados para implementar el web service
Fuente: Autor

CAPITULO IV

4 Estudio, diseño y desarrollo de la app android.

4.1 Concepto

App, es una aplicación informática diseñada para ser ejecutada en dispositivos móviles y/o tabletas, y permiten al usuario efectuar una tarea concreta de cualquier tipo, por ejemplo aplicaciones móviles de ocio, profesional, educativas, de acceso a servicios, etc, facilitando las gestiones o actividades que pretende realizar el usuario (Wikipedia, 2017).

Android en un Sistema Operativo además de ser una plataforma de Software basada en el núcleo de Linux, por lo tanto es de código abierto; fue diseñada en un principio para dispositivos móviles que permite controlarlos a través de bibliotecas desarrolladlas o adaptados por Google mediante el lenguaje de programación Java (MUNDOMANUALES, 2011).

Figura 111. Dispositivo móvil con sistema operativo Android
Fuente: http://celulares.com.uy/data/f/c/blog/6245/0x0/celulares-android-gamamedia.jpg

Recurriendo a los conceptos mencionados se puede decir que una app android es una aplicación que se puede instalar en dispositivos móviles y/o tabletas que tengan como sistema operativo Android, la cual sirve para facilitar diversas actividades al usuario.

4.2 Arquitectura

Android está construido con una arquitectura de 4 capas o niveles relacionados entre sí, en la siguiente figura se puede observar un diagrama general (Revelo, 2014):

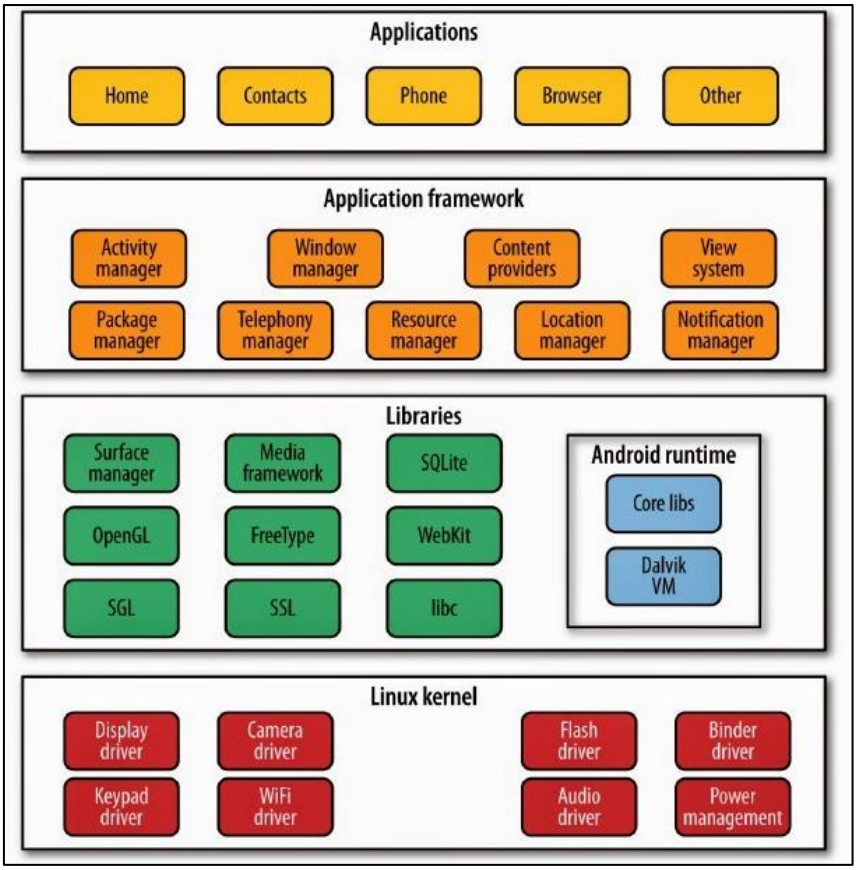


Figura 112. Arquitectura de Android por capas
Fuente: http://www.hermosaprogramacion.com/2014/08/aprendiendo-la-arquitectura-de-android/

En la figura 112 se puede observar que la estructura de Android se encuentra construida sobre el Kernel de Linux, luego sigue la capa de librerías relacionadas con una estructura administradora en tiempo de ejecución; en la siguiente capa se encuentra un Framework de apoyo para construcción de aplicaciones, aquí es donde se desarrollará la app android para el GAD Municipal de Chordeleg y al final será ejecutada en la última capa, que es la de Aplicaciones (Revelo, 2014).

4.3 Ventajas

Entre las ventajas más importantes de una app android, se tiene las siguientes.

- Android es compatible con varios dispositivos y actualmente es el más utilizado por dispositivos móviles, por lo tanto la app se podría instalar en la mayoría de dispositivos móviles (Porras, 2016).
- Android es un software libre gracias a la licencia Apache, lo cual le convierte en un sistema operativo totalmente abierto, donde cualquier desarrollador pueda crear, modificar y mejorar el código de las apps (Porras, 2016).
- El desarrollo de una app android no requiere de permisos a terceros, existe total libertad (Porras, 2016).
- Existe una amplia comunidad de desarrolladores en la red (Porras, 2016).
- Android permite la opción multitarea inteligente, lo que hace que los dispositivos sean capaces de gestionar varias app abiertas al mismo tiempo (Porras, 2016).

4.4 Estructura

Las aplicaciones desarrolladas para el sistema operativo Android, siguen la misma estructura básica, la cual se compone del código fuente como tal, archivos de recursos y vistas, librerías de código y el archivo android manifest de configuración global (Gomez, 2015).

En la siguiente figura se puede observar las carpetas y archivos que conforman la estructura de una app android:

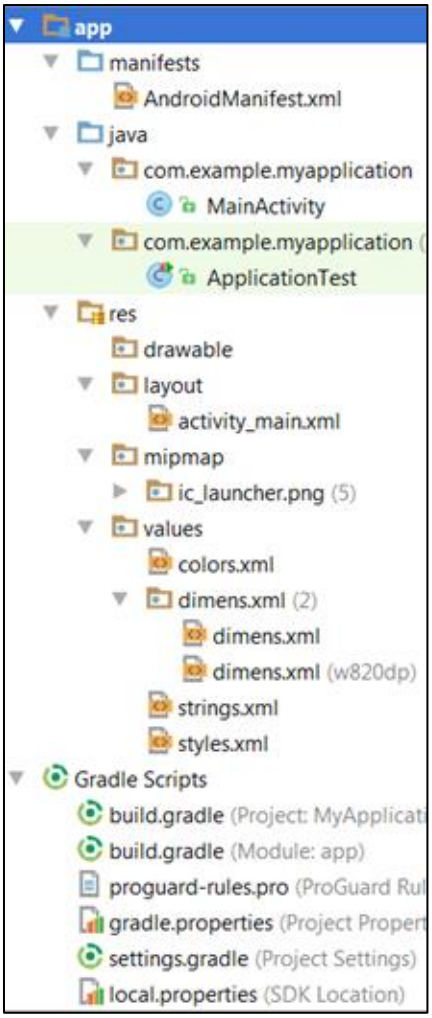


Figura 113. Estructura de una App Android
Fuente: http://www.androidcurso.com/index.php/tutoriales-android/31-unidad-1-vision-general-y-entorno-de-desarrollo/148-elementos-de-un-proyecto-android

A continuación se describe cada uno de los ficheros observados en la figura 113 (Tomás, 2015):

- AndroidManifest.xml: es el archivo básico de configuración general de la app Android, donde se definen todas las activities de la app, sus permisos, entre otros.
- java: es la carpeta que contiene el código fuente de la aplicación.
- MainActivity: Clase Java principal del código de la actividad inicial.
- res: Carpeta que contiene los recursos usados por la aplicación.
  - drawable: carpeta donde se almacenan los ficheros de imágenes.
  - layout: carpeta que contiene los ficheros XML con las vistas de la aplicación.

- values: se encuentran los ficheros XML donde se puede indicar valores que serán usados en la aplicación.

- Gradle Scripts: carpeta en la cual se almacenan una serie de ficheros Gradle que permiten compilar y construir la aplicación.

4.5 Preparación e instalación de software en la PC de desarrollo

Al igual que en el desarrollo del web service, se debe regir estrictamente a utilizar software libre para el desarrollo de la app Android, por las razones expuestas anteriormente en la sección 3.6.

En cuanto al equipo de cómputo, se utilizará la misma PC de desarrollo descrita en la sección 3.6.1, recordando que en la sección 3.6.2 se instaló como sistema operativo, Ubuntu Desktop 16.10.

4.5.1 IDE de programación

Un IDE es una herramienta que ayuda a desarrollar de una manera amigable las aplicaciones, brindando ayudas visuales en la sintaxis, plantillas, wizards, plugins y sencillas opciones para probar y hacer un debug (Global Mentoring, 2012).

Android se programa de forma nativa en el lenguaje de programación “Java”, razón por la cual se desarrollará la aplicación en el mismo lenguaje nativo de android; a continuación se describe los IDEs más populares para su desarrollo:

4.5.1.1 Eclipse

Figura 114. Logo del IDE Eclipse
Fuente: http://www.pythondiario.com/2013/06/eclipse-y-pydev-configuracion-del-ide.html

Eclipse es una plataforma de desarrollo de código abierto basada en Java; por sí mismo, es simplemente un marco de trabajo y un conjunto de servicios para la construcción del entorno de desarrollo de los componentes de entrada. Eclipse es multiplataforma, puede ser instalado en sistemas operativos como Windows y Linux, además tiene un conjunto de complementos, incluidas las herramientas de desarrollo de Java (JDT) (Gallardo, 2012).

Entre las principales ventajas se tiene las siguientes:

- Emplea varios plug-in para proporcionar toda su funcionalidad, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no (ECURED, s.f.).
- Eclipse se puede extender mediante plugin's a otros lenguajes de programación como son C/C++, Python, entre otros (ECURED, s.f.).
- La definición que da el proyecto Eclipse acerca de su Software es: "una especie de herramienta universal, un IDE abierto y extensible para todo y nada en particular" (ECURED, s.f.).
- El SDK de Eclipse incluye las herramientas de desarrollo de Java, ofreciendo un IDE con un compilador de Java interno y un modelo completo de los archivos fuente de Java (ECURED, s.f.).
- Tiene una gran comunidad en la red, donde existen foros, chats, entre otros.
- Se puede programar Android haciendo integrando los plugins correspondientes.

4.5.1.2 NetBeans

Figura 115. Logo del IDE NetBeans
Fuente: https://cwiki.apache.org/confluence/display/NETBEANS/NetBeans+Logo

NetBeans es un entorno de desarrollo integrado libre, principalmente trabaja para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDE2 es un producto libre y gratuito sin restricciones de uso (Wikipedia, 2016).

A continuación se describe las características principales (Gimenez, 2012):

- Es un IDE multiplataforma, puede ser instalado en windows, linux, entre otros.
- NetBeans genera código con tan solo arrastrar y soltar los componentes.
- La plataforma Netbeans puede ser usada para desarrollar cualquier tipo de aplicación.
- Se puede extender mediante módulos.
- Incluye Templates y Wizards.
- Existe un gran soporte en la red de internet.
- Se puede programar Android haciendo integrando los módulos correspondientes.

4.5.1.3 Android Studio

Figura 116. Logo del IDE Android Studio
Fuente: https://www.taringa.net/post/ebooks-tutoriales/19729951/Megapost-Aprende-crear-Aplicaciones-Android-con-este-post.html

Android Studio es el entorno de desarrollo integrado oficial para la plataforma Android de código libre. Fue anunciado el 16 de mayo de 2013 en la conferencia Google I/O, y reemplazó a Eclipse como el IDE oficial para el desarrollo de aplicaciones para Android. (Wikipedia, 2016).

Entre las características principales de Android Studio, se tiene las siguientes (Ardións, 2016):

- Es puramente Android, creado para programar en Android (Ardións, 2016).
- Más intuitivo, más fácil de usar (Ardións, 2016).
- Código más ordenado, estructurado y mejores sugerencias (Ardións, 2016).
- Es mejor para diseñar Interfaces (Ardións, 2016).
- Compilador Gradle (Ardións, 2016).
- Olvídate de los problemas con las dependencias (Ardións, 2016).
- Es un IDE multiplataforma (Ardións, 2016).
- Existe un gran soporte en la red (Ardións, 2016).

4.5.1.4 Tabla comparativa y selección del IDE de programación

En la siguiente tabla se menciona las características principales de interés:

Tabla 50. IDE de programación Eclipse, NetBeans y Android Studio

Item	Eclipse	NetBeans	Android Studio
Multiplataforma	Si	Si	Si
Código Libre	Si	Si	Si
Soporte en internet	Si	Si	Si
Programación visual	Si	Si	Si
Desarrollo de App Android	Si, agregando módulos	Si, agregando módulos	Si, directo

Fuente: Autor

En la tabla anterior se puede observar, que en este caso es posible hacer uso de cualquiera de los 3 IDE's de programación, sin embargo se resalta que Android Studio es el IDE oficial para programar en Android y no requiere de módulos adicionales para iniciar con el desarrollo de una aplicación; por lo tanto este último se elegirá como el IDE de programación de la app android.

4.5.1.5 Instalación del IDE

En la sección anterior se decidió desarrollar la app android con el IDE de programación Android Studio; razón por la cual se ha realizado los siguientes pasos para instalar el IDE en la PC de desarrollo:

- Antes de iniciar la instalación, se descarga el software de la página oficial de Android Studio https://developer.android.com/studio/index.html, la versión actual al momento que se realiza el presente proyecto es la 2.2.2. Posterior a ello se debe descomprimir el archivo descargado y se copia en la ruta "/usr/local".

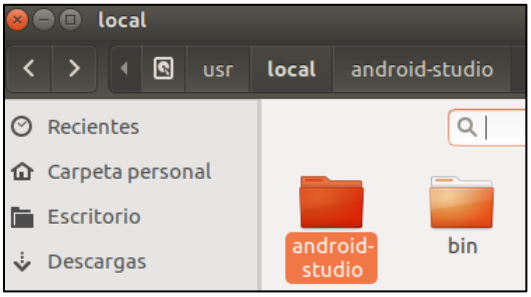


Figura 117. Carpeta de instalación de Android Studio
Fuente: Autor

- Luego se tiene que abrir la terminal de Ubuntu ejecutando Crtl+ALT+T, para instalar las dependencias de una PC de 64bits.

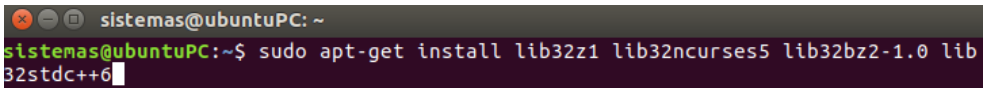


Figura 118. Librerías para instalar Android Studio en una PC de 64bits
Fuente: Autor

- Dirigirse a la carpeta "bin" que se encuentra dentro de la carpeta de la figura 117 y seguidamente se ejecuta el script de instalación de Android Studio.

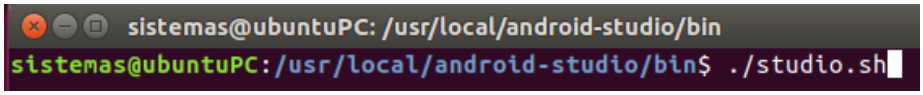


Figura 119. Iniciar la instalación de Android Studio
Fuente: Autor

- Una vez que se inicie el asistente de instalación de Android Studio, se continúa con un clic en next.

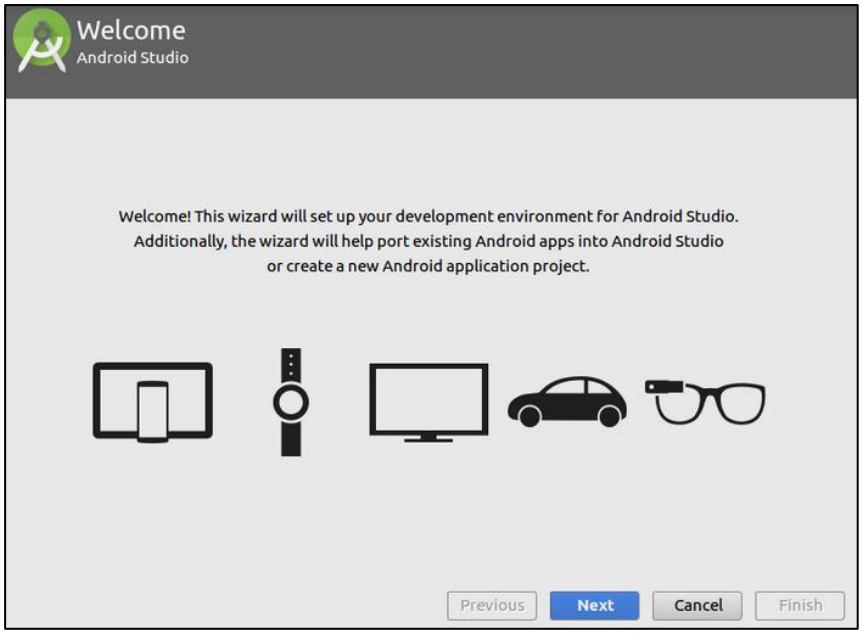


Figura 120. Asistente de instalación de Android Studio
Fuente: Autor

- Luego debe seleccionar el tipo de instalación estándar y se continúa con un clic en next.

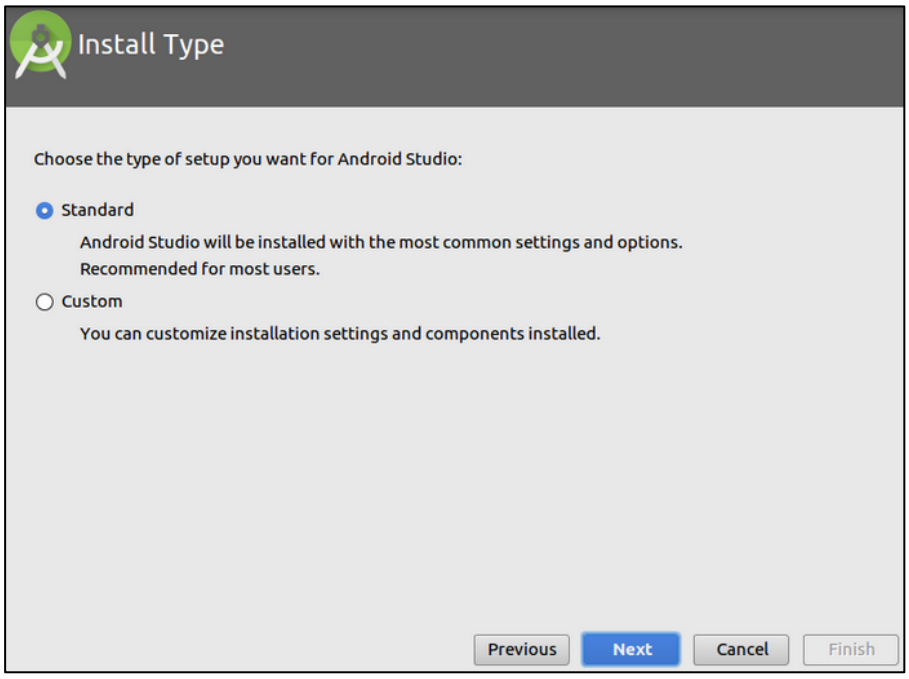


Figura 121. Instalación estándar de Android Studio
Fuente: Autor

- Verificar la configuración previa a la instalación de Android Studio y continuar con un clic en next.

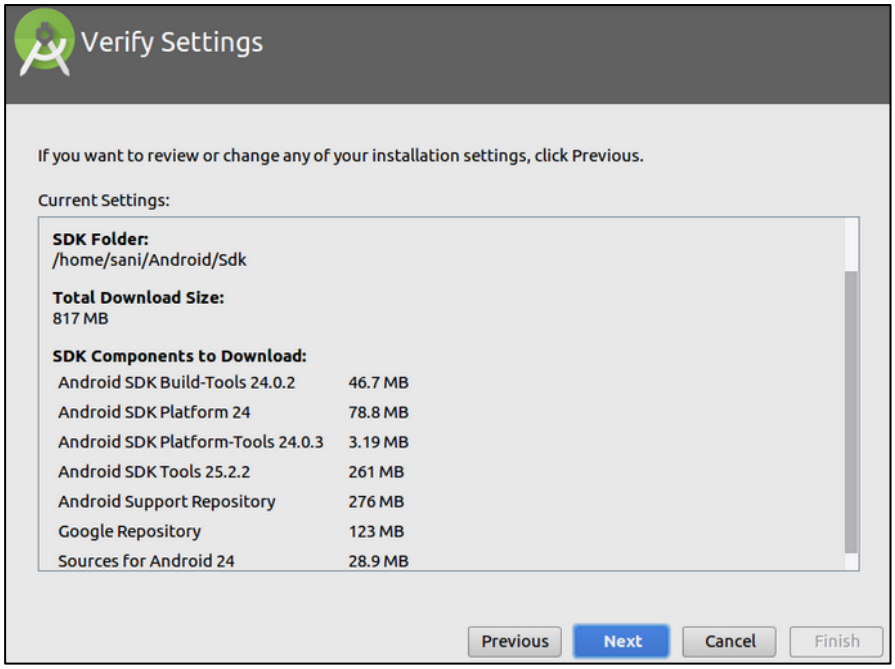


Figura 122. Verificación de la instalación de Android Studio
Fuente: Autor

- Se presenta una notificación sobre una mejor configuración para el emulador de Android, únicamente se da un clic en Finish y la instalación comienza inmediatamente.

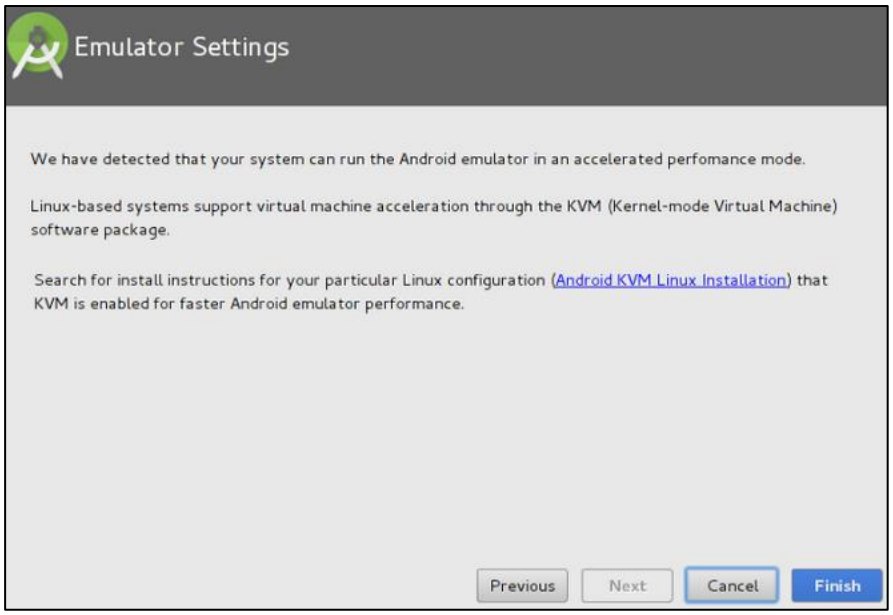


Figura 123. Notificación para mejorar el emulador de Android
Fuente: Autor

- Una vez que termine la instalación se da un clic en Finish.

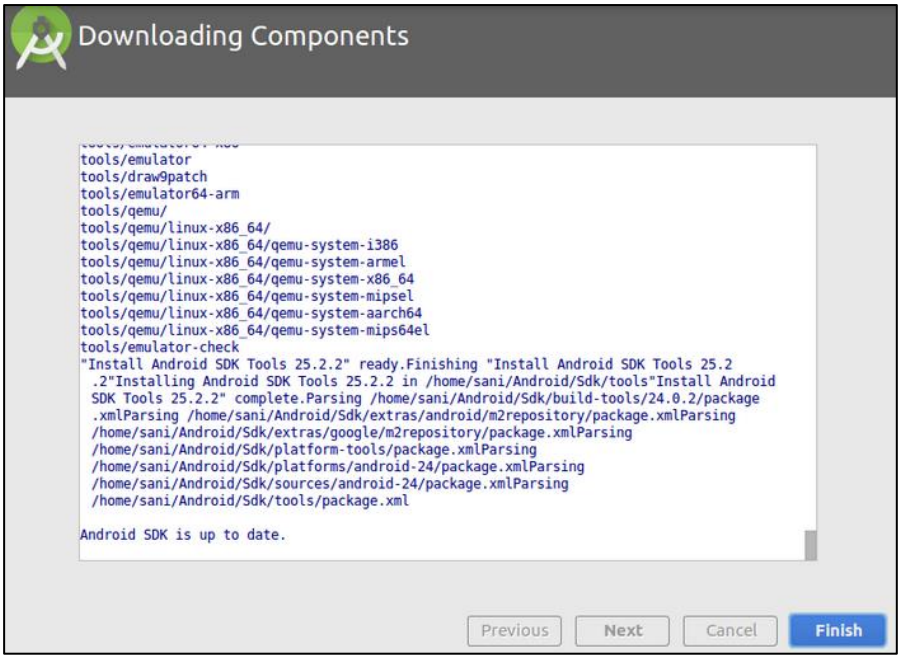


Figura 124. Instalación de Android Studio terminada
Fuente: Autor

- Finalmente se abre la pantalla de inicio de Android Studio.

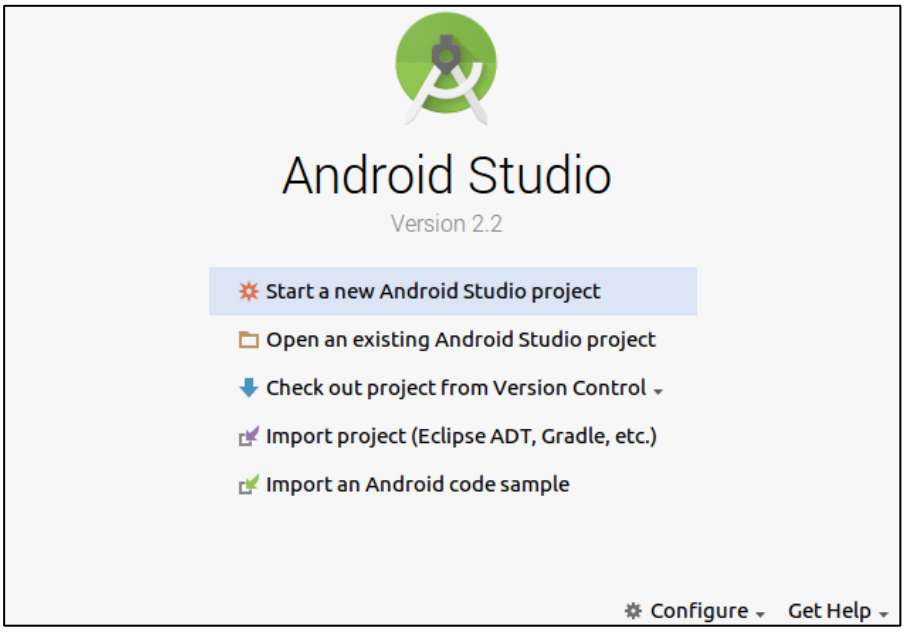


Figura 125. Pantalla de inicio de Android Studio
Fuente: Autor

4.6 Desarrollo de la app Android

4.6.1 Crear el proyecto

A continuación se describe una serie de pasos, que se ejecutaron para crear el proyecto de la aplicación en Android Studio:

- Al igual que en el web service, se crea una carpeta principal donde va a guardar todos los archivos correspondientes al código fuente en desarrollo de la aplicación; la carpeta llevará el nombre de "codigoApp" y se ubica en la ruta "Escritorio/TESIS/Android/" de la PC de desarrollo.

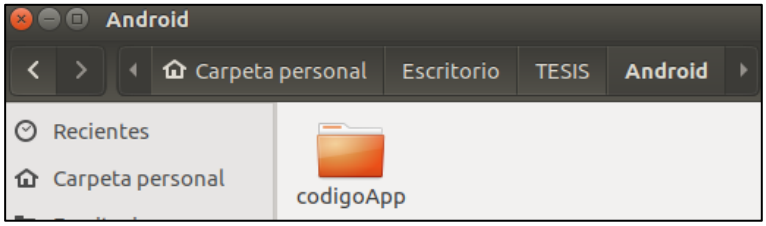


Figura 126. Creación de la carpeta principal del código fuente de la app android
Fuente: Autor

- Iniciar el IDE Android Studio, para crear el nuevo proyecto se tiene que dar un clic en "File>New>New Project", luego se ingresa el nombre del proyecto, el dominio, la ruta de la carpeta y se da un clic en "Next".

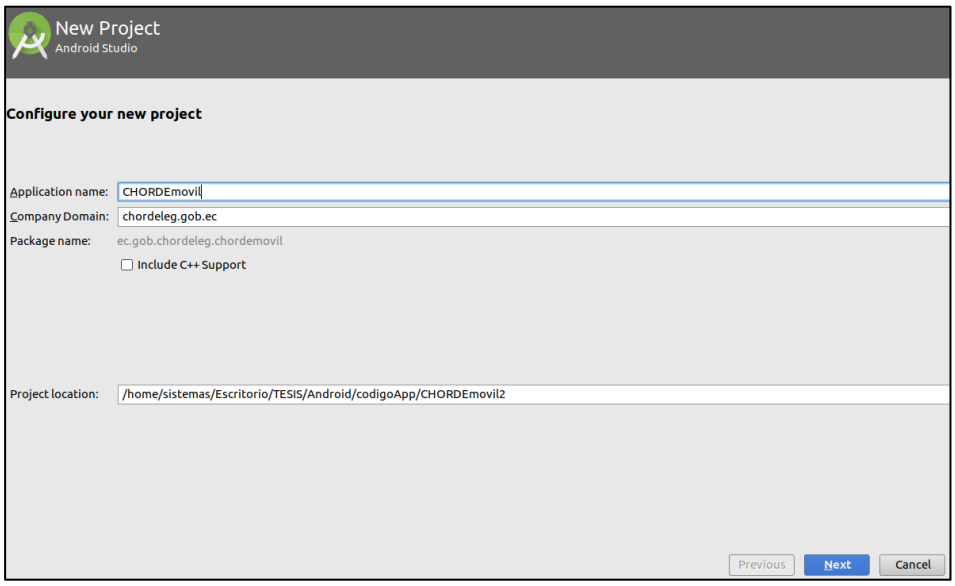


Figura 127. Creación del nuevo proyecto para la app android
Fuente: Autor

- Seleccionar el SDK mínimo con el cual trabajará la aplicación, se deja el recomendado que es la API 15: Android 4.0.3 y se da un clic en "Next" para continuar.

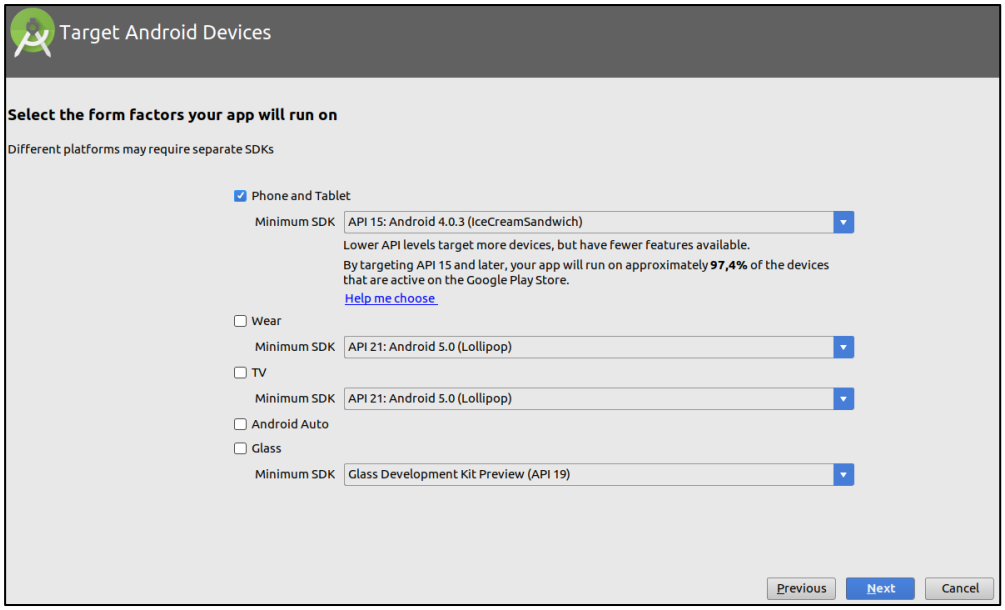


Figura 128. Selección del SDK mínimo para la app android
Fuente: Autor

- Seleccionar la plantilla “Empty Activity” que viene por defecto y se debe dar un clic en “Next”.

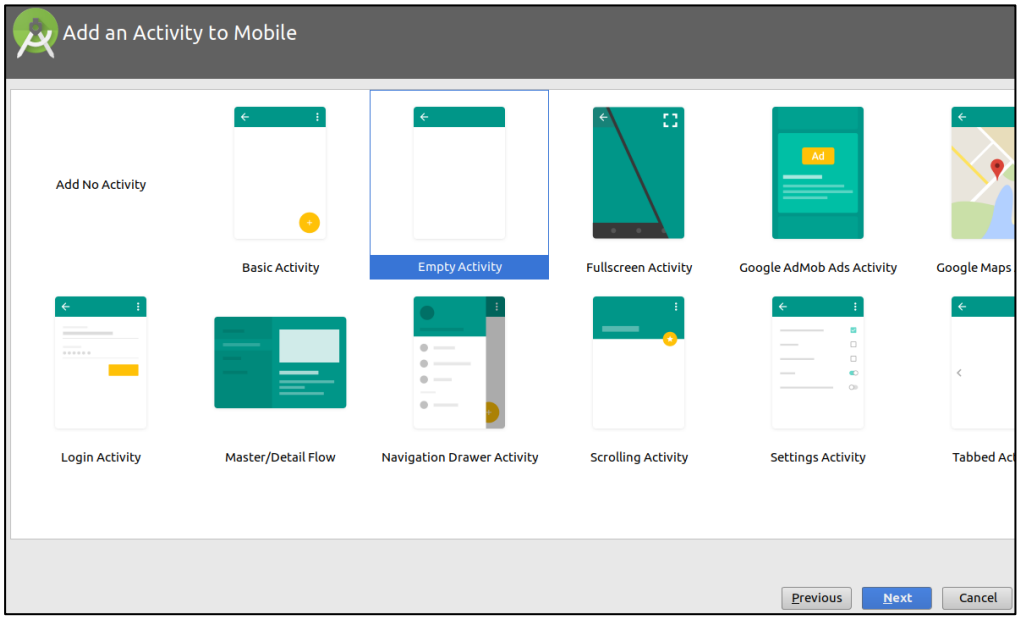


Figura 129. Selección del SDK mínimo para la app android
Fuente: Autor

- A continuación se asigna un nombre para el Activity y Layout principal del proyecto, y se da un clic en “Finish”.

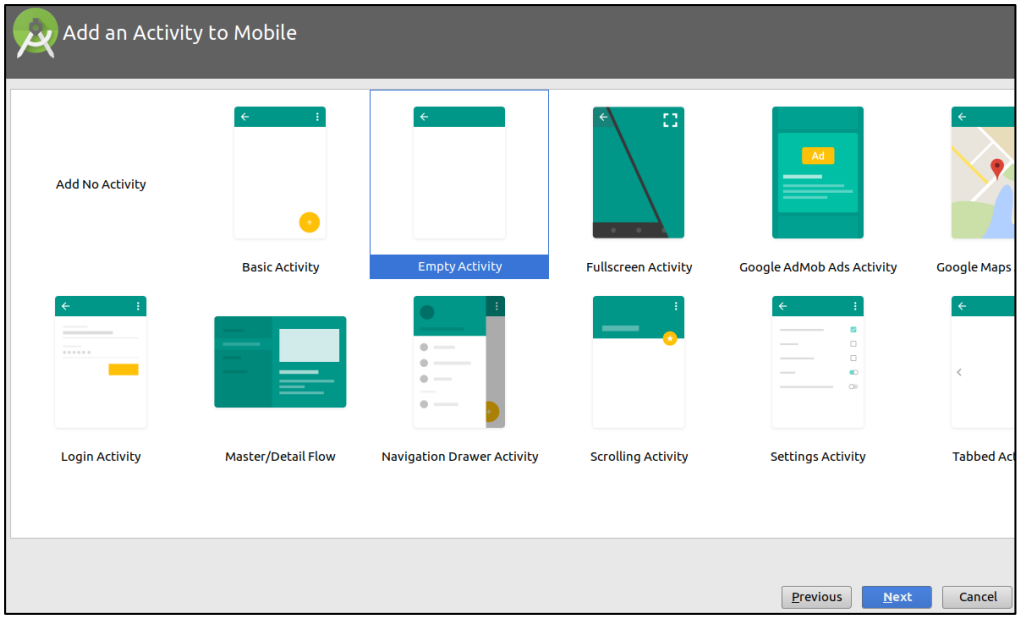


Figura 130. Selección del SDK mínimo para la app android
Fuente: Autor

- Finalmente se abre el nuevo proyecto con las configuraciones que se realizó anteriormente.

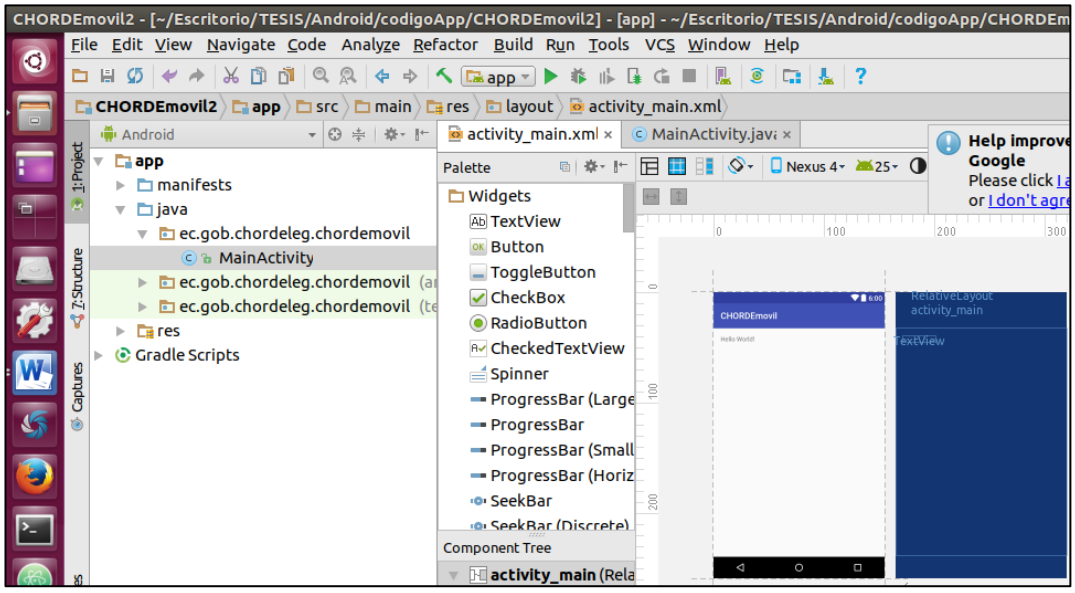


Figura 131. Selección del SDK mínimo para la app android
Fuente: Autor

4.6.2 Permisos para usar el internet y la red del dispositivo móvil

Luego de tener creado el proyecto, se tiene que asignar los permisos a la aplicación para hacer uso del internet y la red del dispositivo móvil en el cual se instalará la aplicación,

lo cual es necesario para realizar las consultas de los tributos municipales desde la red de internet. Para ello se realiza los siguientes pasos:

- Dentro del proyecto, se abre el archivo de configuración global, el cual es "AndoridManifest.xml" que se encuentra en la ruta "app>manifests".

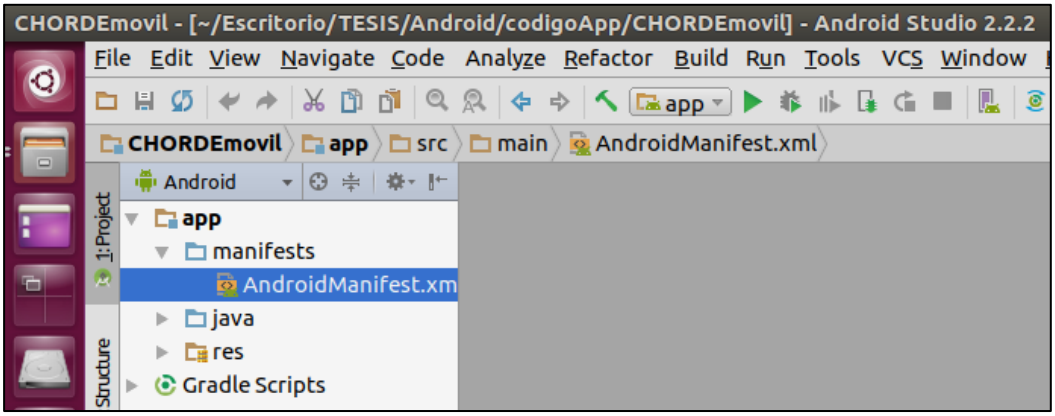


Figura 132. Ubicación del archivo AndroidManifest.xml en la app android
Fuente: Autor

- Luego se ingresa los permisos respectivos, al inicio del archivo "AndoridManifest.xml".

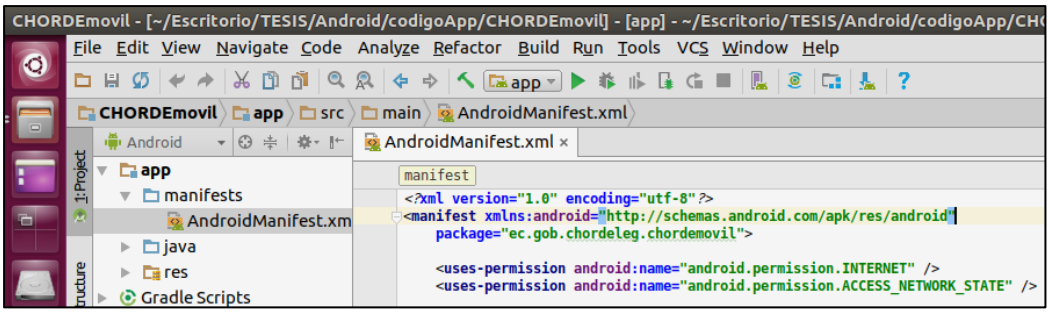


Figura 133. Permisos de internet y red del dispositivo móvil con la app android
Fuente: Autor

4.6.3 Instalar dependencias para las peticiones HTTP

Para las peticiones HTTP de la app en desarrollo, se utilizará la librería "volley", para su instalación se realiza los siguientes pasos:

- Dentro del proyecto, se abre el archivo que contiene las dependencias del proyecto, el cual es "build.gradle" que se encuentra en la carpeta "Gradle Scripts".

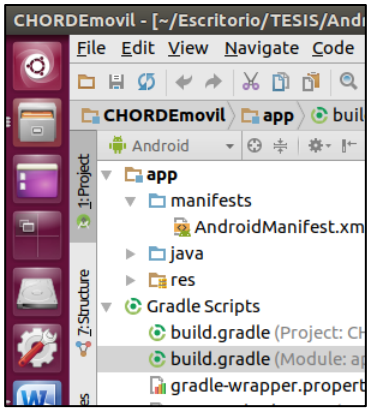


Figura 134. Archivo de instalación de dependencias de la app android
Fuente: Autor

- Luego se incluye la librería correspondiente, en la sección de dependencias dentro del archivo "bulid.gradle".

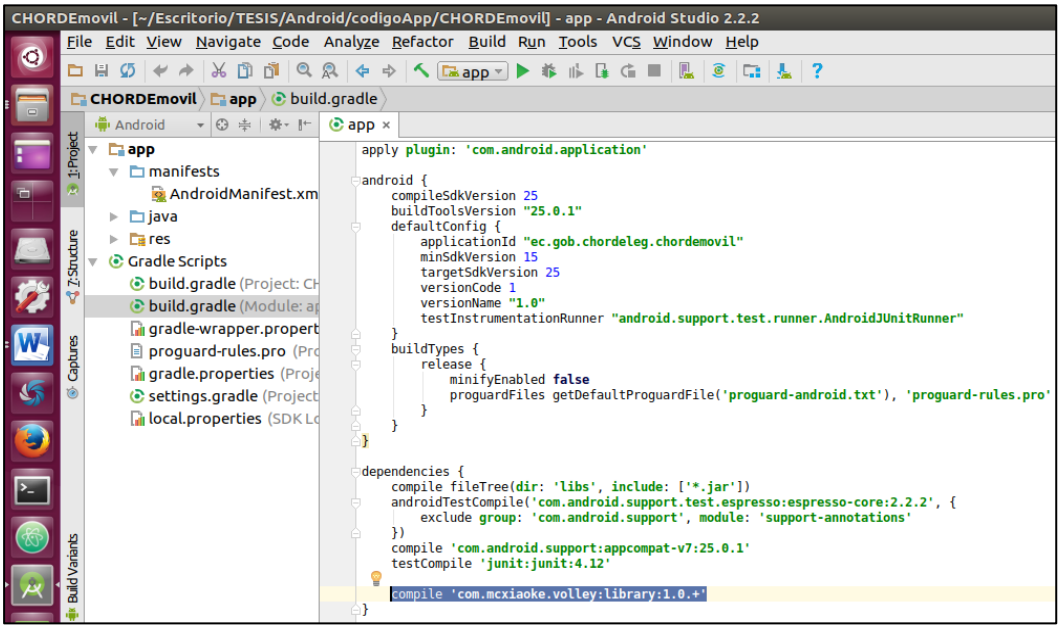


Figura 135. Agregar la librería para peticiones HTTP en la app android
Fuente: Autor

- Por último es necesario sincronizar el proyecto con el archivo "build.gradle" para instalar la librería "volley".

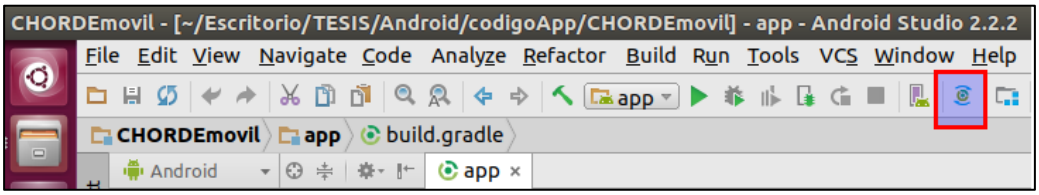


Figura 136. Sincronizar el proyecto con la librería "volley"
Fuente: Autor

4.6.4 Emulador del dispositivo móvil

Para las pruebas de la app en desarrollo, es necesario instalar un emulador del dispositivo móvil, para ello se sigue los siguientes pasos:

- Dentro de del proyecto, dirigirse a “Tools>Android>AVD Manager” y dar un clic en “Create Virtual Device”.

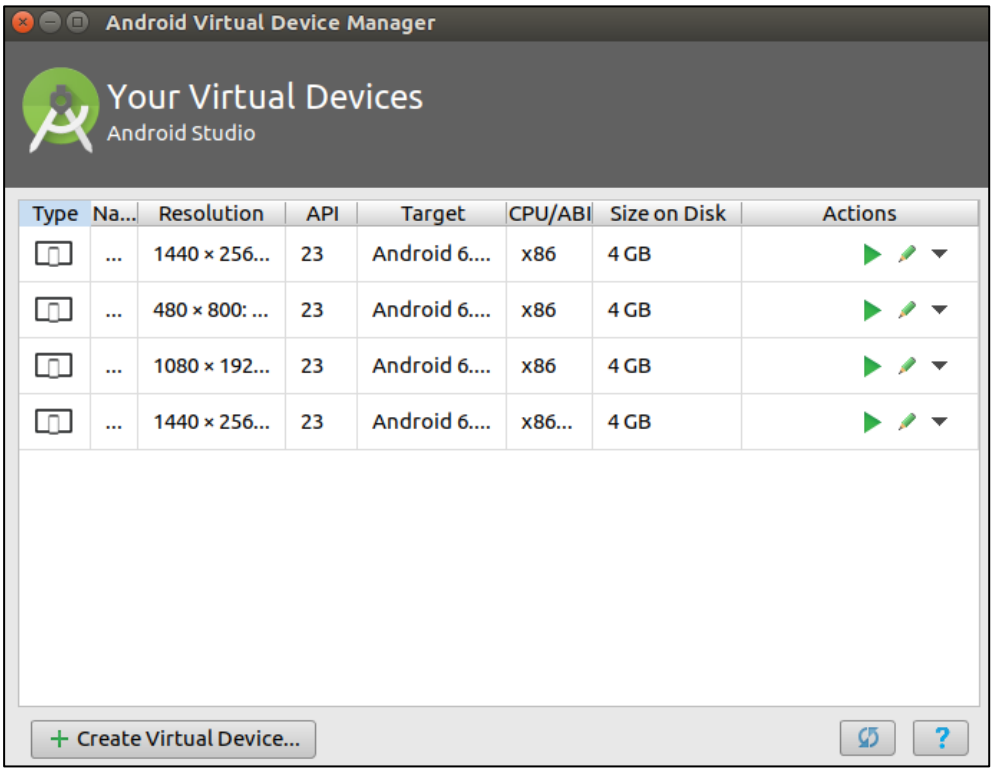


Figura 137. Asistente para instalar un dispositivo móvil virtual en Android Studio
Fuente: Autor

- Luego seleccionar uno de los dispositivos móviles virtuales, y dar un clic en “Next”.

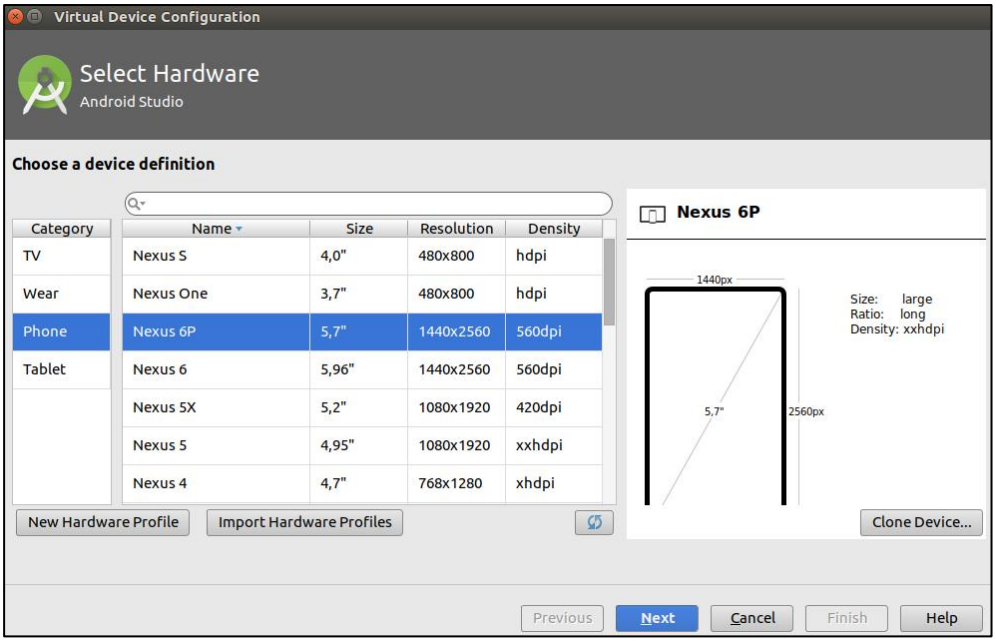


Figura 138. Selección del dispositivo móvil virtual en Android Studio
Fuente: Autor

- Seguidamente seleccionar la versión de Android que tendrá el dispositivo móvil virtual, y se da un clic en “Next”.

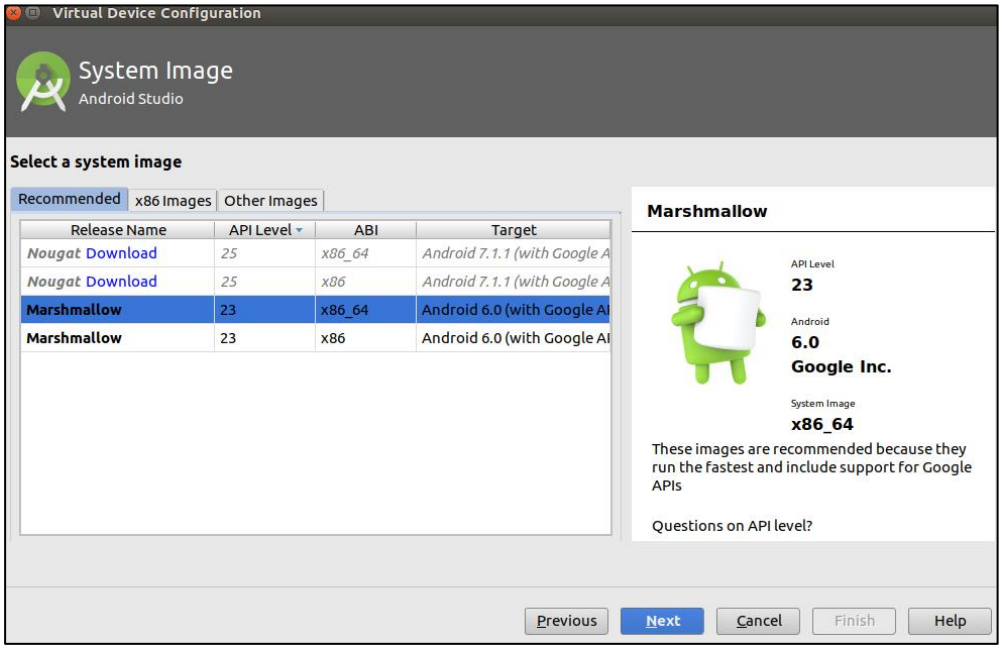


Figura 139. Selección de Android para el dispositivo móvil virtual en Android Studio
Fuente: Autor

- Finalmente se asigna un nombre al dispositivo móvil virtual, y se da un clic en “Finish”.

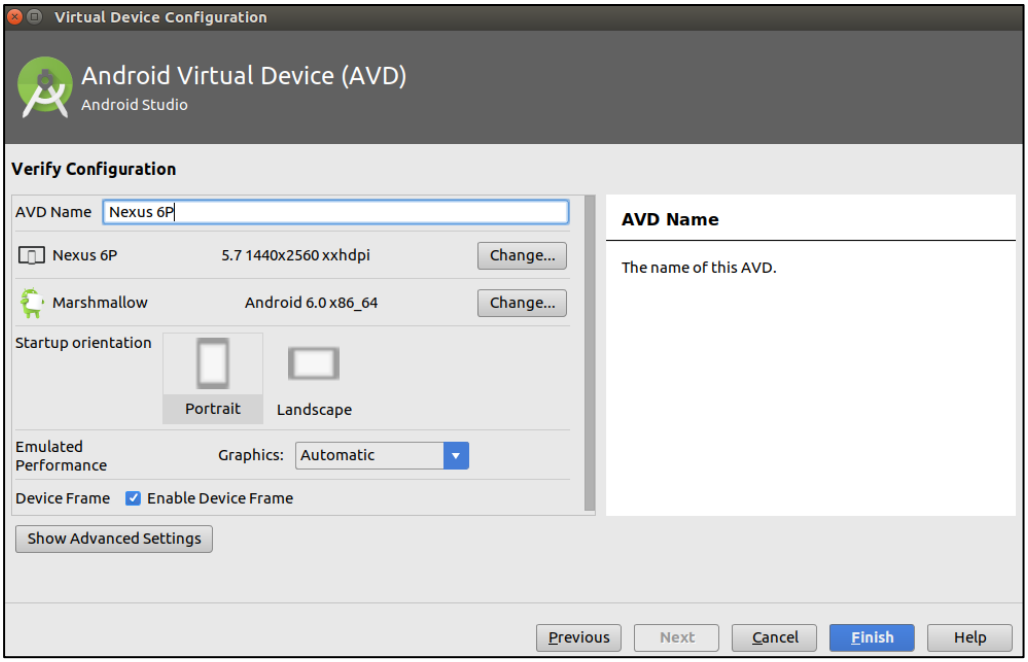


Figura 140. Configurar el nombre del dispositivo móvil virtual de Android Studio
Fuente: Autor

4.6.5 Código fuente desarrollado

El código fuente completamente desarrollado se puede revisar en la sección de Apéndices, sin embargo a continuación se describe de manera general lo más destacado del código fuente:

- Para cada ventana de la aplicación se tiene dos archivos relacionados entre sí, el primero es de extensión .xml que contiene toda la parte gráfica en la cual se define los componentes, tales como cajas de texto, botones, entre otros; y el segundo es de extensión .java, en el cual se define las operaciones a realizar de acuerdo a los propósitos planteados. La ventana principal de la app android es la de "Bienvenido", para ello se tiene los dos archivos correspondientes, que en este caso se nombró "activity_bienvenido.xml" y "Bienvenido.java", en el archivo Bienvenido.java se crea una función que espera 2 segundos al iniciar la aplicación y luego automáticamente pasa al menú principal de la aplicación.

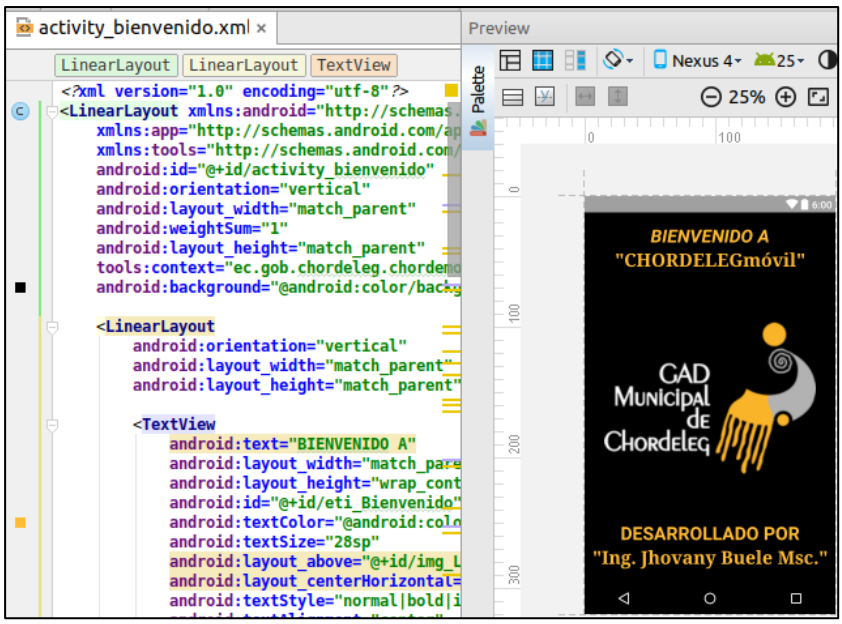


Figura 141. Ventana principal de la aplicación (activity_bienvenido.xml)
Fuente: Autor

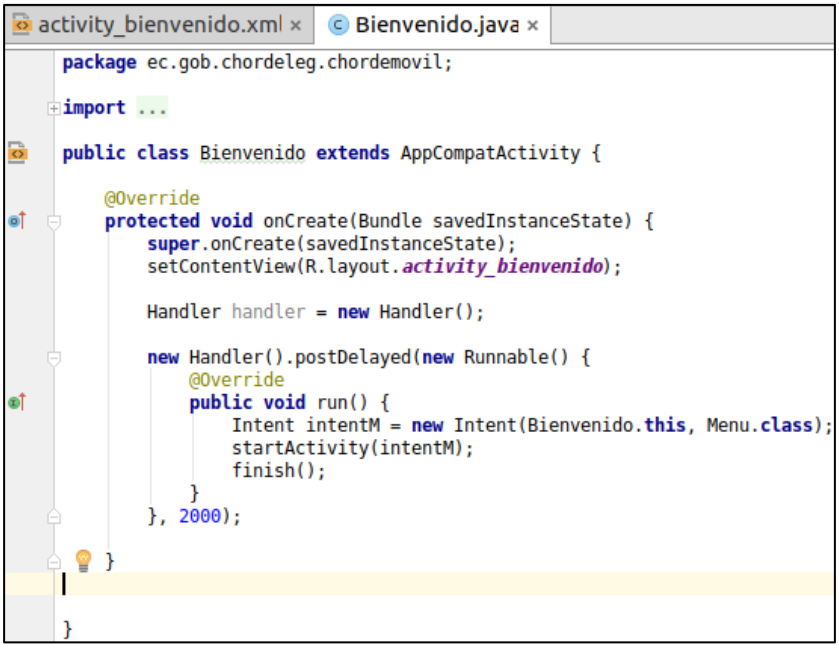


Figura 142. Código de la ventana principal de la aplicación (Bienvenido.java)
Fuente: Autor

- Luego de ejecutarse la función de la figura 142, donde espera 2 segundos, inmediatamente se abre el menú principal, para el cual se tiene los archivos “activity_menu.xml” y “menu.java”; dentro del menú principal se tiene un botón para consultar cada tributo principal del GAD Municipal de Chordeleg, en las

siguientes figuras se puede observar el diseño del menú principal y el código que recibe las acciones ejecutadas desde el menú:

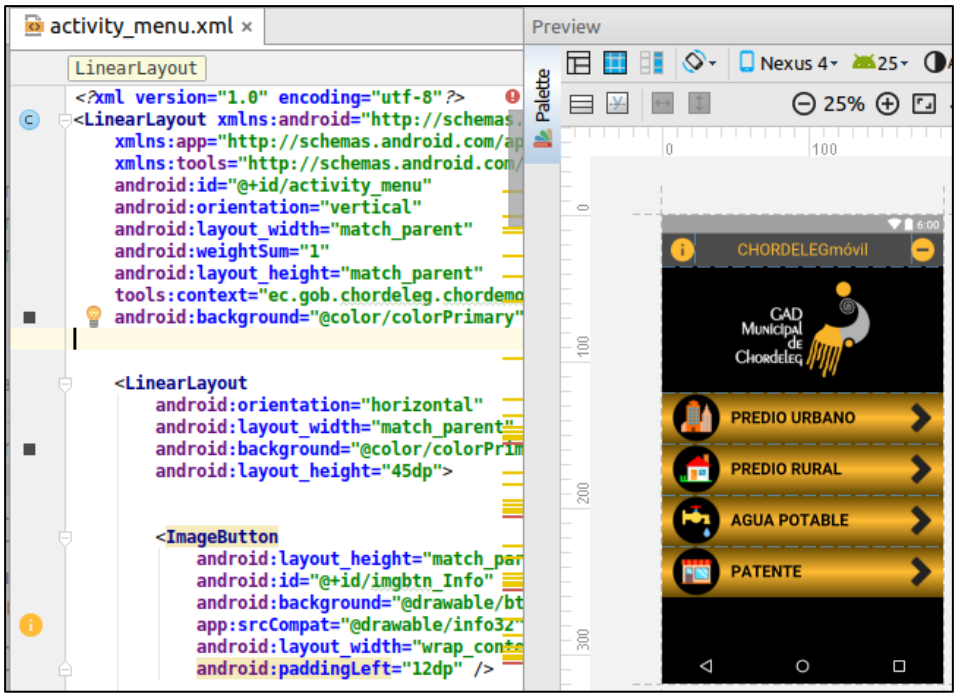


Figura 143. Menú principal de la aplicación (activity_menu.xml)
Fuente: Autor

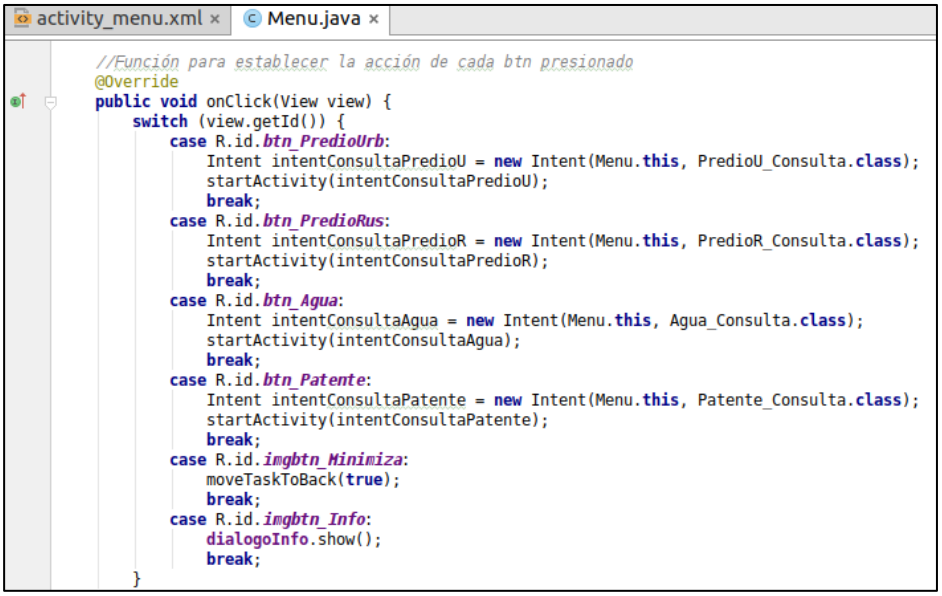


Figura 144. Código del menú principal de la aplicación (Menu.java)
Fuente: Autor

- En la figura 143 se tiene 4 botones para la consulta de los principales tributos municipales de Chordeleg; cada botón lleva a una nueva ventana de consulta según

sea el caso, para el predio urbano se tiene dos archivos: el primero "activity_prediou_consulta.xml" que corresponde al diseñó de la ventana de consulta y el segundo "PredioU_Consulta.java" que contiene el código a ejecutarse por cada consulta al predio urbano. Para el predio rural, agua potable y patente municipal, se tiene el mismo esquema, dos archivos por cada tributo, el primero con el diseño de la ventana de consulta y el segundo con el código a ejecutarse por cada consulta; en la siguientes figuras se puede observar cada una de las ventanas de consulta y parte del código fuente a ejecutarse por cada consulta de los tributos municipales:

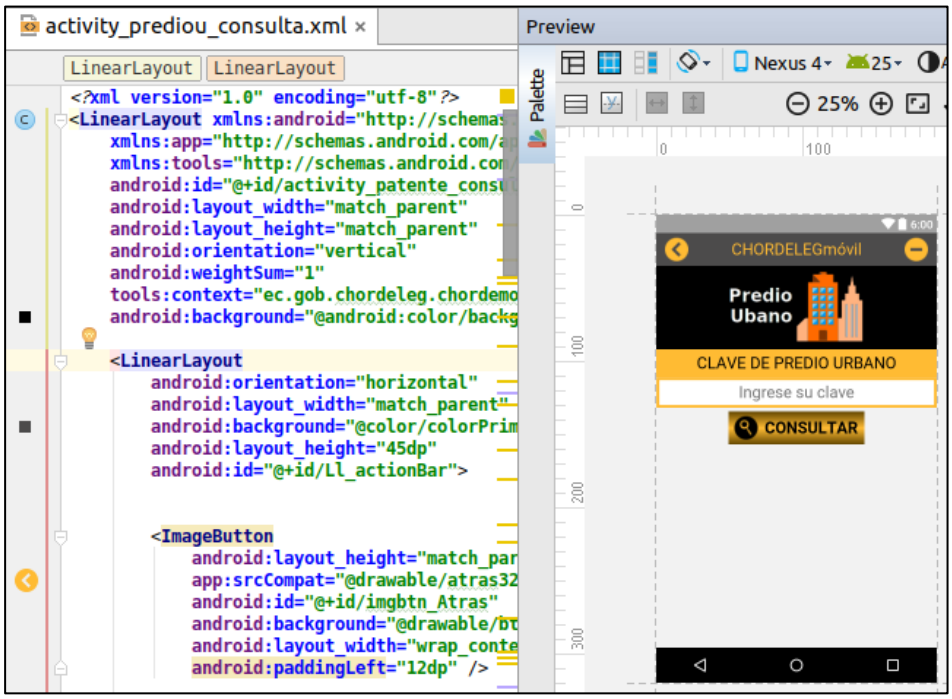


Figura 145. Consulta de predio urbano (activity_prediou_consulta.xml)
Fuente: Autor

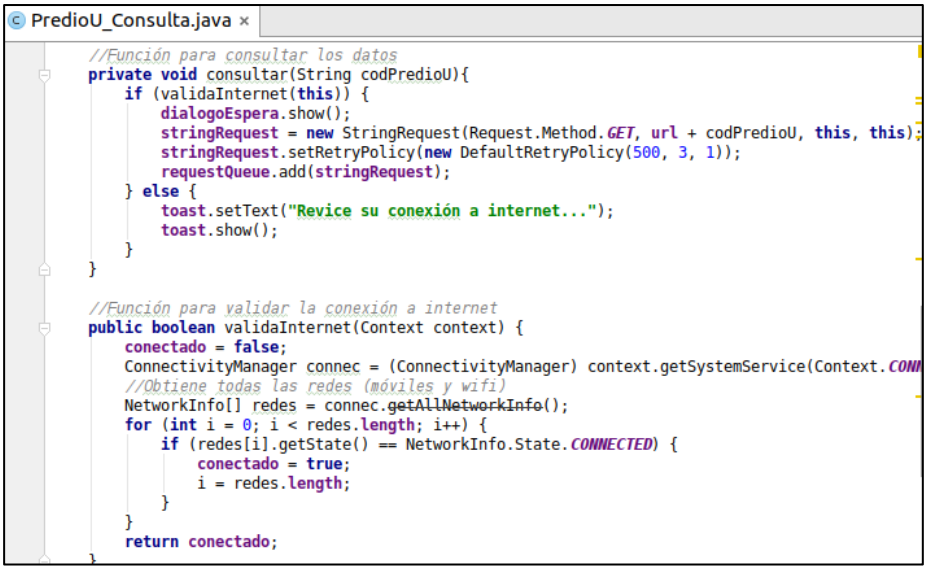


Figura 146. Código de consulta de predio urbano (PredioU_Consulta.java)
Fuente: Autor

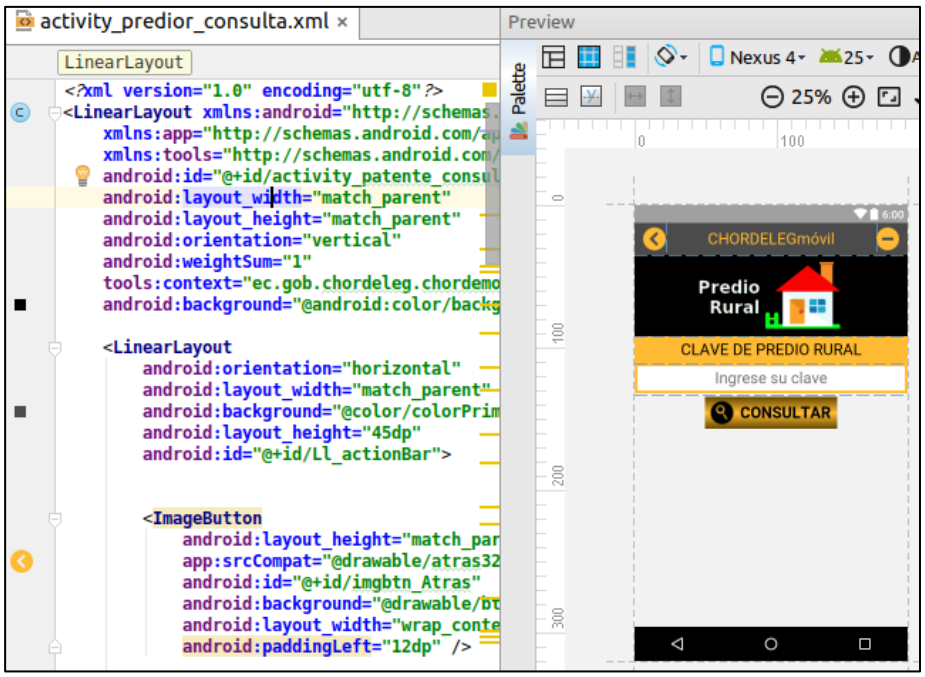


Figura 147. Consulta de predio rural (activity_predior_consulta.xml)
Fuente: Autor

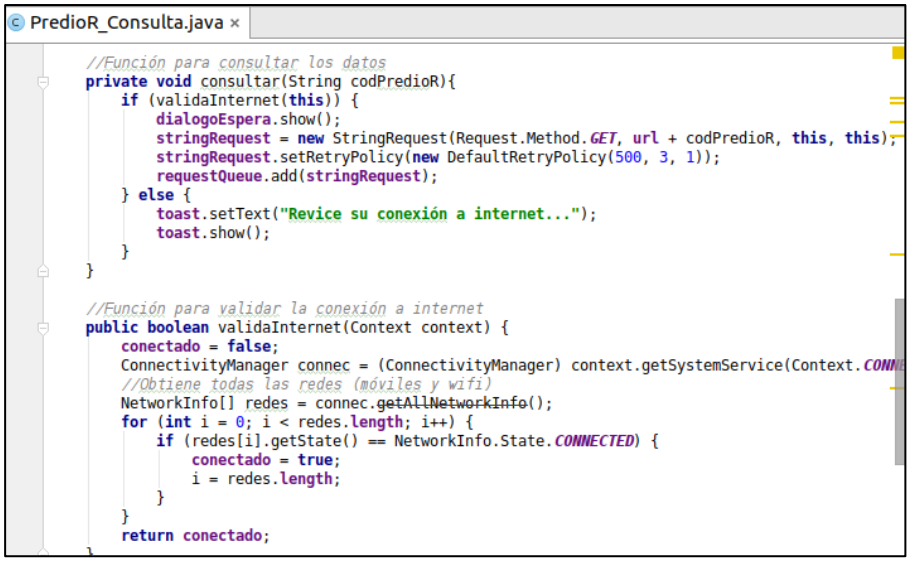


Figura 148. Código de consulta de predio rural (PredioR_Consulta.java)
Fuente: Autor

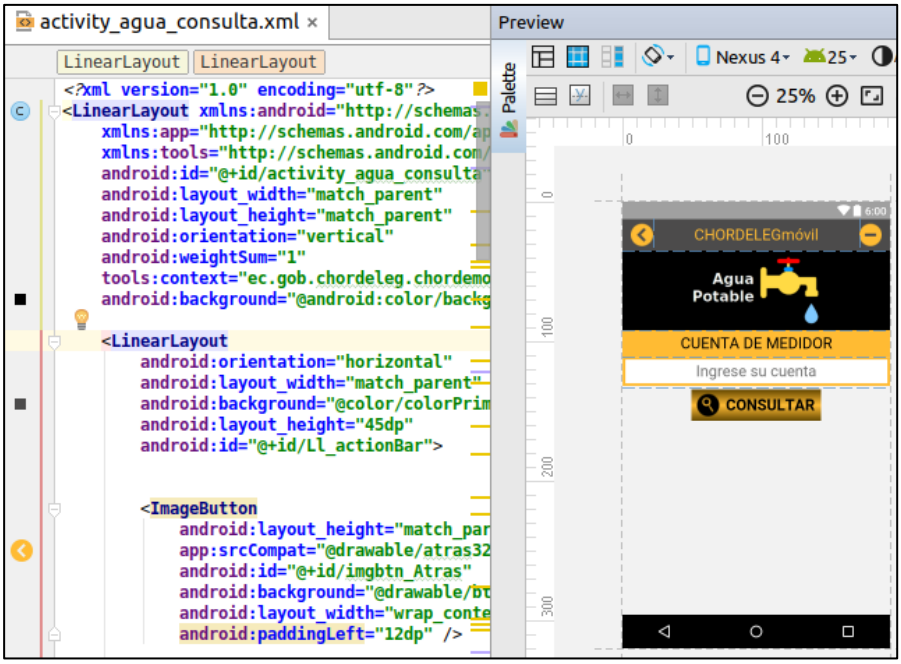


Figura 149. Consulta de agua potable (activity_agua_consulta.xml)
Fuente: Autor

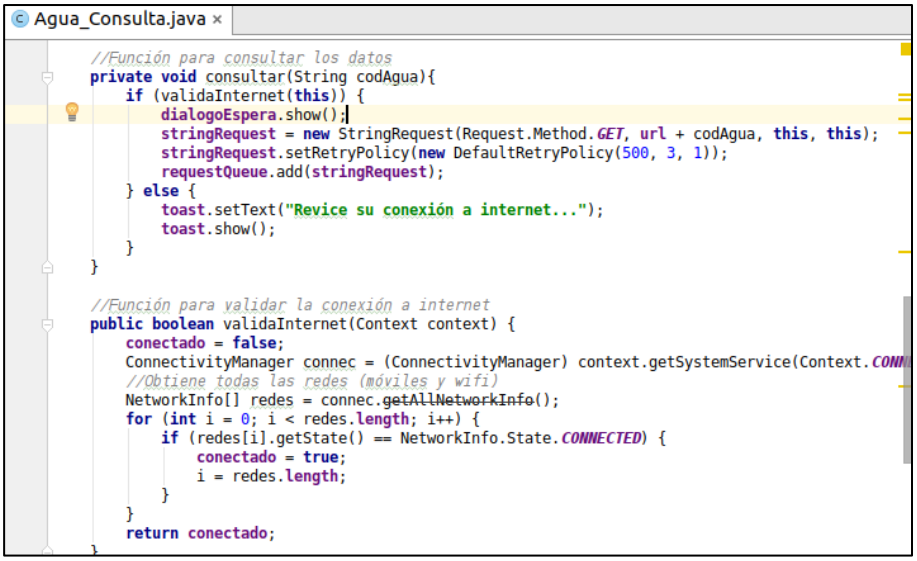


Figura 150. Código de consulta de agua potable (Agua_Consulta.java)
Fuente: Autor

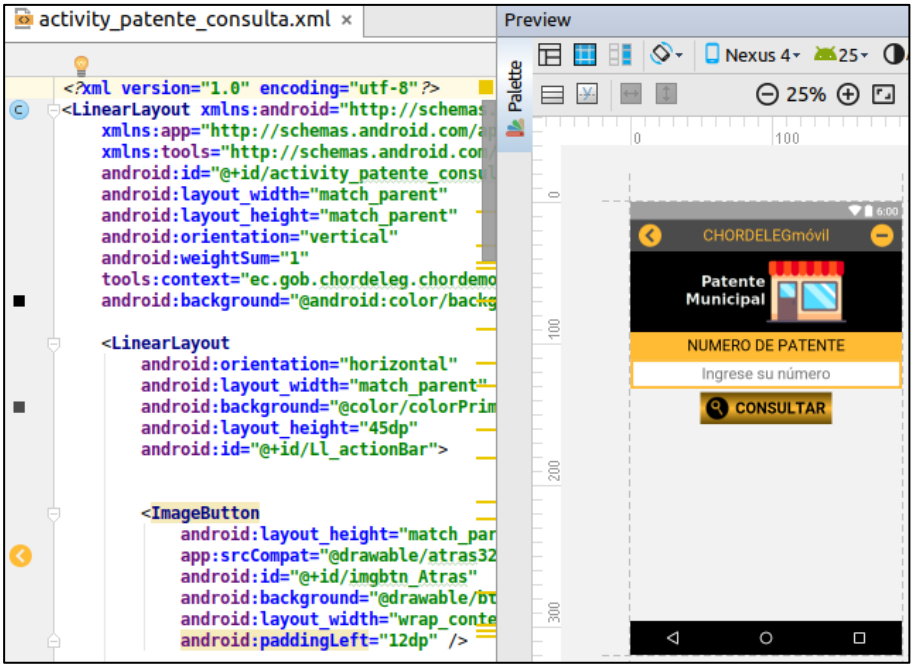


Figura 151. Consulta de patente municipal (activity_patente_consulta.xml)
Fuente: Autor

Patente_Consulta.java ×

```
//Función para consultar los datos
private void consultar(String codPatente){
    if (validaInternet(this)) {
        dialogoEspera.show();
        stringRequest = new StringRequest(Request.Method.GET, url + codPatente, this, this);
        stringRequest.setRetryPolicy(new DefaultRetryPolicy(500, 3, 1));
        requestQueue.add(stringRequest);
    } else {
        toast.setText("Revice su conexión a internet...");
        toast.show();
    }
}

//Función para validar la conexión a internet
public boolean validaInternet(Context context) {
    conectado = false;
    ConnectivityManager connec = (ConnectivityManager) context.getSystemService(Context.CONN
    //Obtiene todas las redes (móviles y wifi)
    NetworkInfo[] redes = connec.getAllNetworkInfo();
    for (int i = 0; i < redes.length; i++) {
        if (redes[i].getState() == NetworkInfo.State.CONNECTED) {
            conectado = true;
            i = redes.length;
        }
    }
    return conectado;
}
```

Figura 152. Código de consulta de patente municipal (Patente_Consulta.java)
Fuente: Autor

- Finalmente se tiene las ventanas que presentan los datos obtenidos por cada consulta de los principales tributos municipales del GAD Municipal de Chordeleg; regresando a la figura 145 se puede ver que en la ventana de consulta del predio urbano se dispone de un botón consultar, que al presionarlo despliega una nueva ventana que presenta un reporte de los datos consultados por predio urbano, para lo cual se tienen así mismo 2 archivos, el primero “activity_prediou_reporte.xml” que corresponde al diseñó de la ventana de reporte y el segundo “PredioU_Reporte.java” que contiene el código a ejecutarse al obtener el reporte del predio urbano. Para el predio rural, agua potable y patente municipal, se tiene el mismo esquema, dos archivos por cada tributo, el primero con el diseño de la ventana de reporte y el segundo con el código a ejecutarse al obtener cada reporte; en la siguientes figuras se puede observar cada una de las ventanas de reporte y parte del código fuente que se ejecuta por cada reporte obtenido:

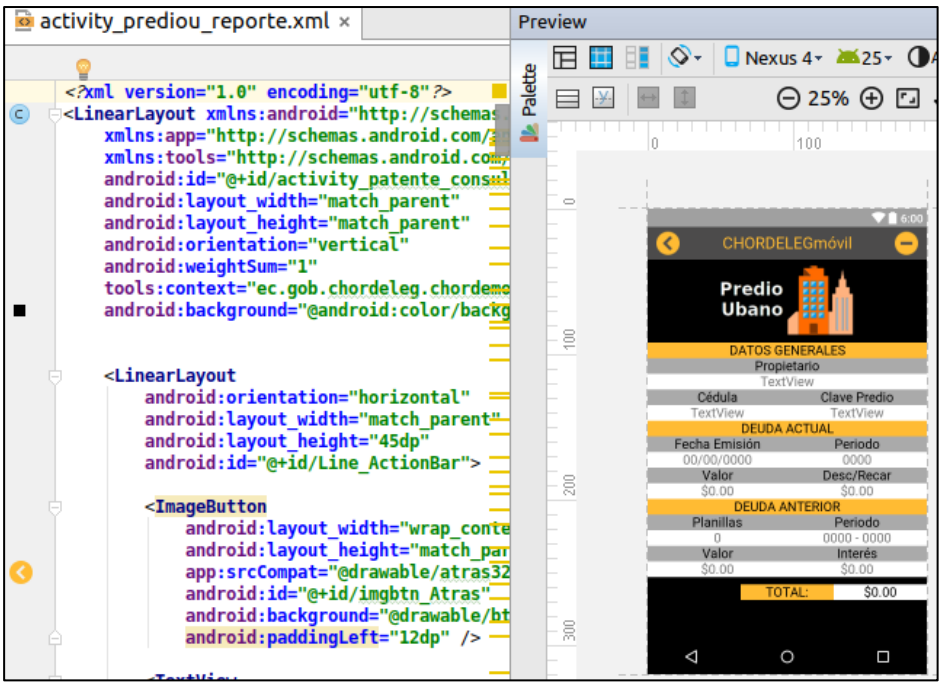


Figura 153. Reporte de predio urbano (activity_prediou_reporte.xml)
Fuente: Autor

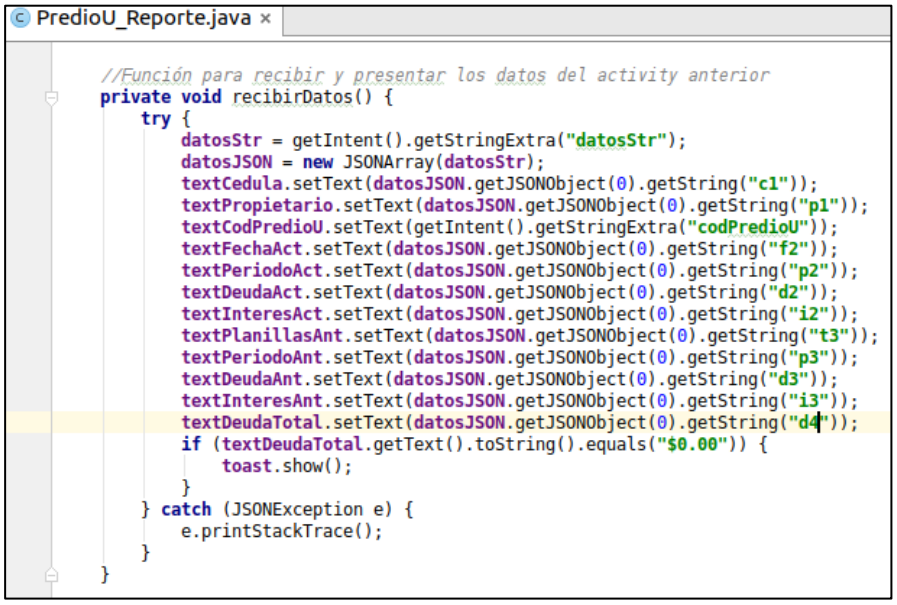


Figura 154. Código del reporte de predio urbano (PredioU_Reporte.java)
Fuente: Autor

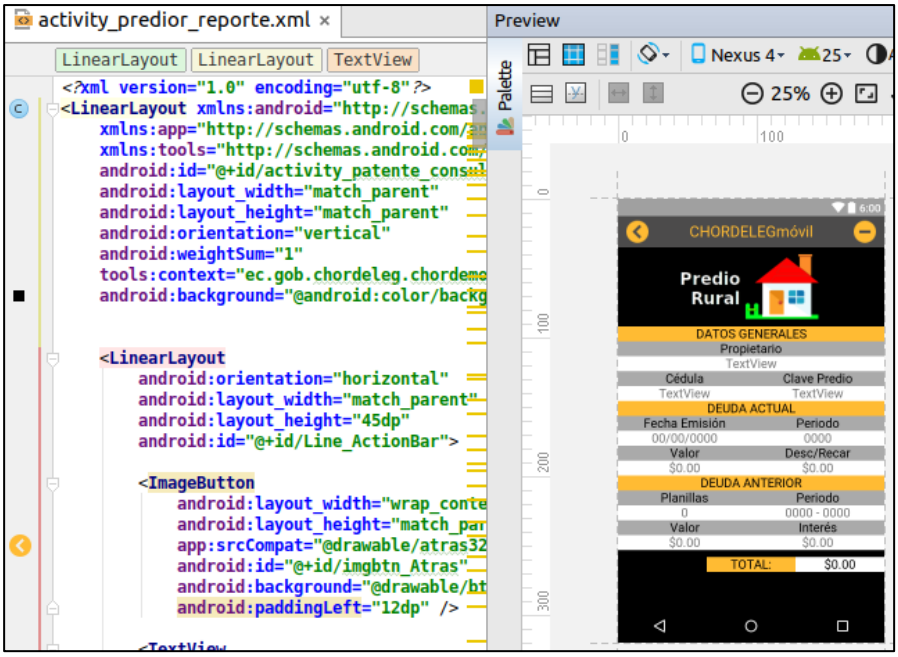


Figura 155. Reporte de predio rural (activity_predior_reporte.xml)
Fuente: Autor

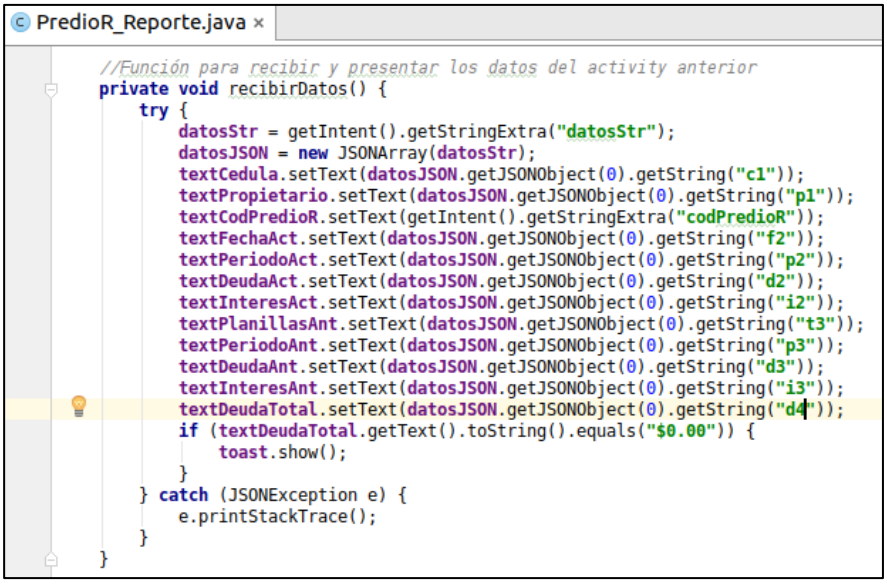


Figura 156. Código del reporte de predio rural (PredioR_Reporte.java)
Fuente: Autor

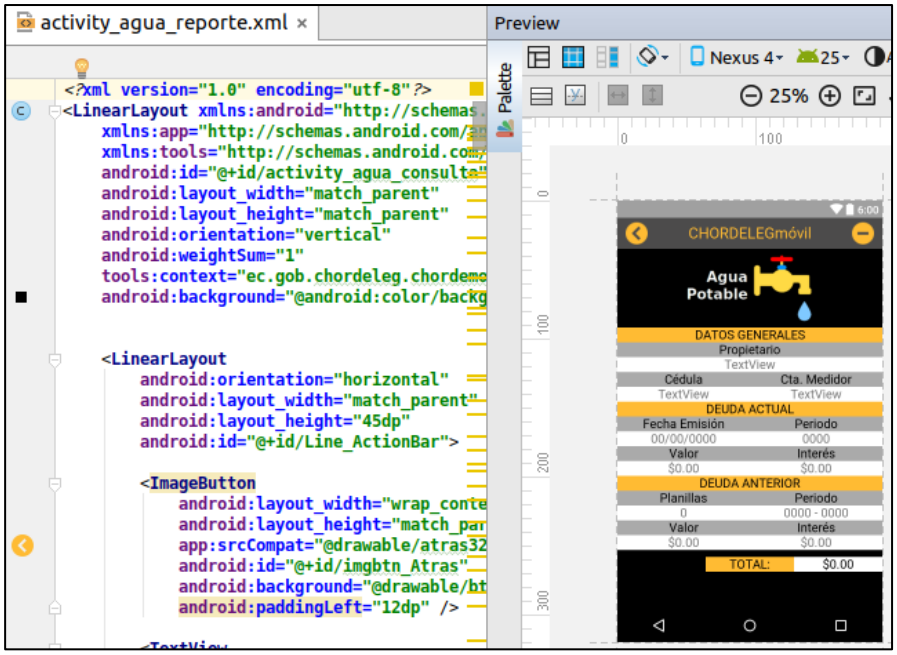


Figura 157. Reporte de agua potable (activity_agua_reporte.xml)
Fuente: Autor

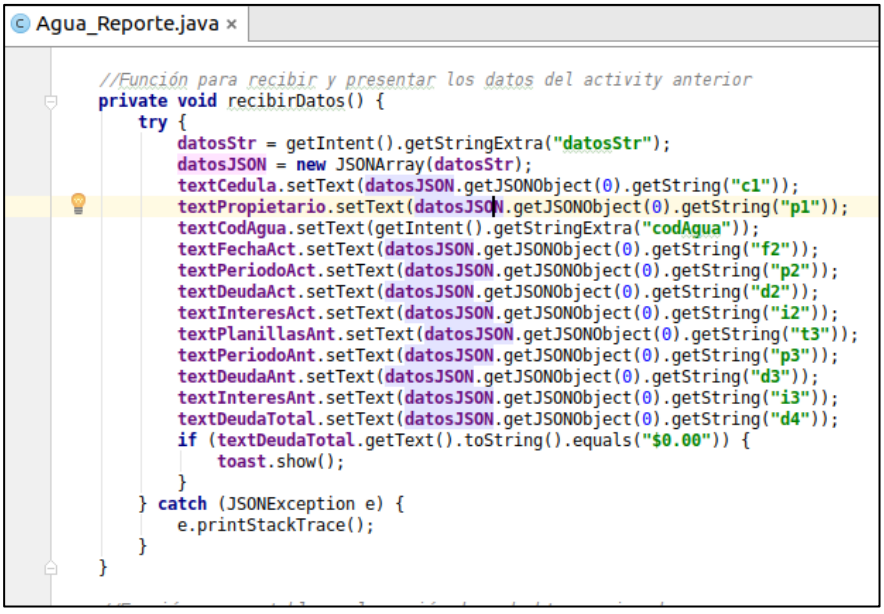


Figura 158. Código del reporte de agua potable (Agua_Reporte.java)
Fuente: Autor

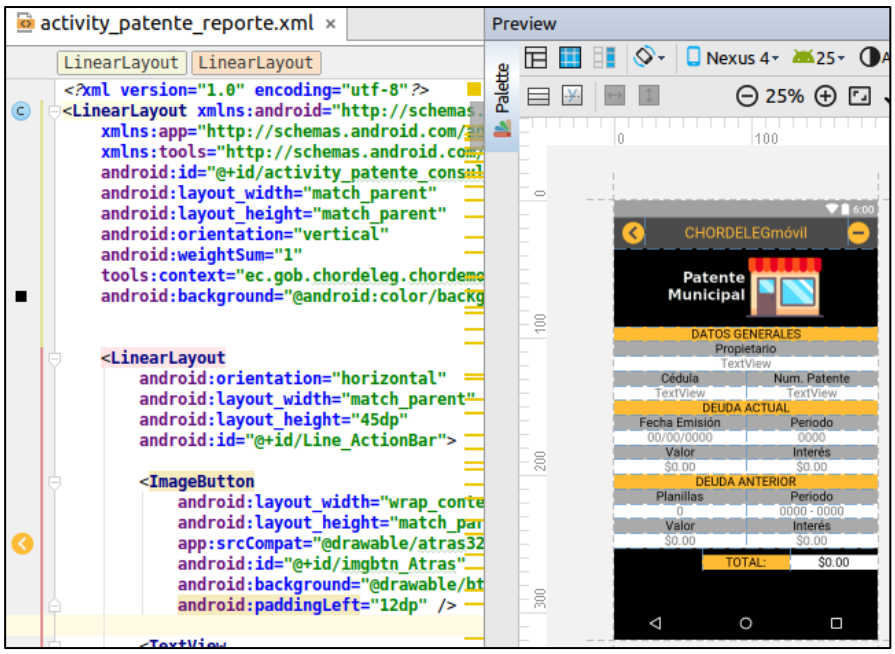


Figura 159. Reporte de patente municipal (activity_patente_reporte.xml)
Fuente: Autor

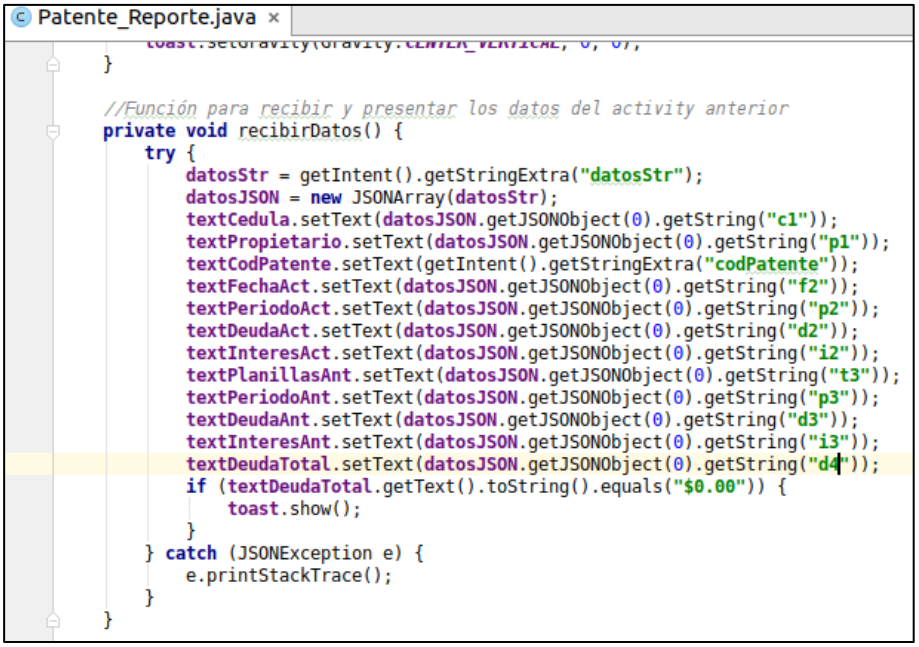


Figura 160. Código del reporte de patente municipal (Patente_Reporte.java)
Fuente: Autor

4.6.6 Pruebas de la app android

Se realizaron pruebas de consulta por cada tributo municipal, a continuación se detalla lo mencionado:

- Antes de iniciar con las pruebas respectivas, se ejecuta la aplicación seleccionando el dispositivo móvil virtual que se instaló en la sección 4.6.4. Para ello una vez que esté abierto el proyecto en Android Studio, dar un clic en el ícono “play”, seleccionar el dispositivo móvil y se da un clic en “OK”.

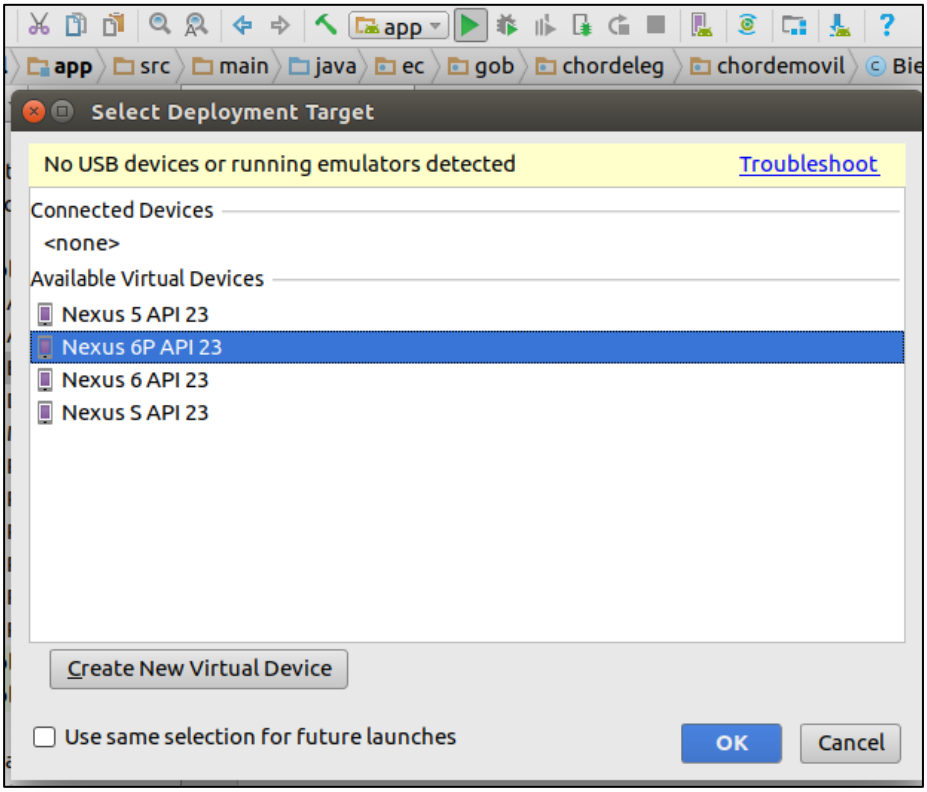


Figura 161. Iniciar la aplicación desarrollada en Android Studio
Fuente: Autor

- Una vez iniciada la aplicación, se presenta el emulador del dispositivo móvil con la ventana de bienvenido:

Figura 162. Pantalla de bienvenida de la aplicación android
Fuente: Autor

- Luego de 2 segundos, automáticamente pasa al menú principal:

Figura 163. Menú principal de la aplicación android
Fuente: Autor

- En la parte superior izquierda del menú principal se tiene el ícono de información acerca de la aplicación.

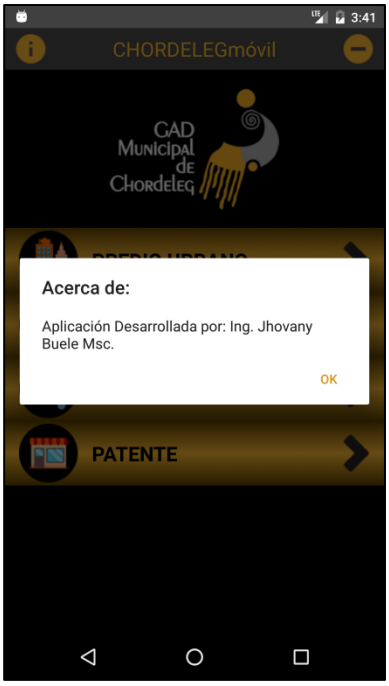


Figura 164. Información acerca de la aplicación android
Fuente: Autor

- En la parte superior derecha del menú principal, se tiene el ícono para minimizar la aplicación.

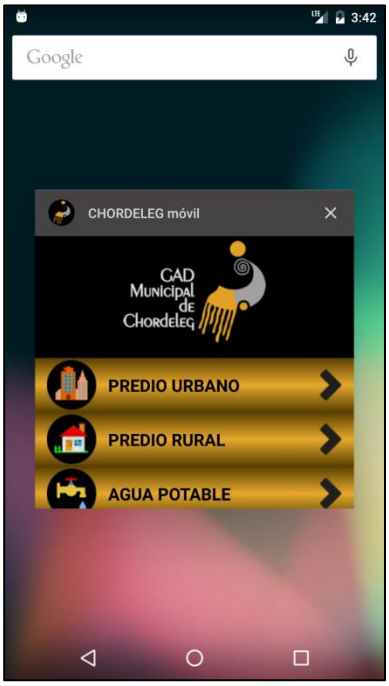


Figura 165. Minimizar la aplicación android
Fuente: Autor

- En el menú principal se da un clic en el botón “PREDIO URBANO”, y se entra a consultar un predio urbano de ejemplo.

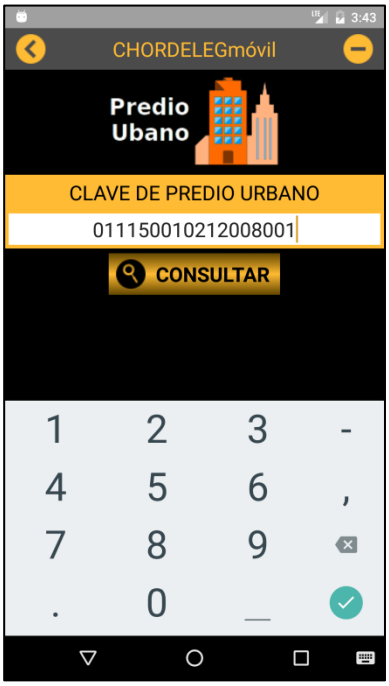


Figura 166. Consultar el predio urbano en la aplicación android
Fuente: Autor

- En la figura anterior al presionar en el botón “CONSULTAR”, se obtiene los datos de consulta del predio urbano.

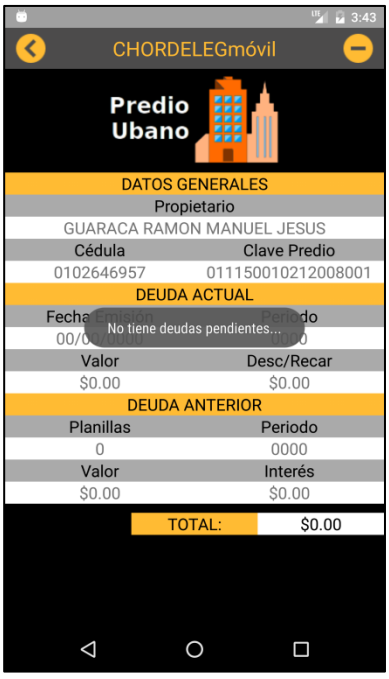


Figura 167. Reporte de consulta del predio urbano en la aplicación android
Fuente: Autor

- En el menú principal se da un clic en el botón “PREDIO RURAL”, y se ingresa en la ventana para consultar un predio rural de ejemplo.

Figura 168. Consultar el predio rural en la aplicación android
Fuente: Autor

- En la figura anterior al presionar en el botón “CONSULTAR”, se obtiene los datos de consulta del predio rural.

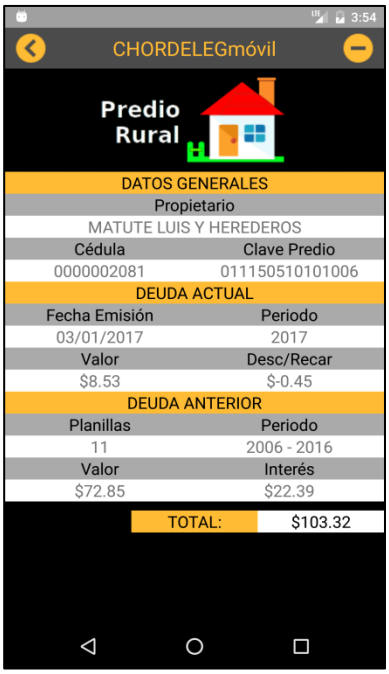


Figura 169. Reporte de consulta del predio rural en la aplicación android
Fuente: Autor

- En el menú principal se da un clic en el botón “AGUA POTABLE”, y entra a consultar un medidor de agua potable como ejemplo.

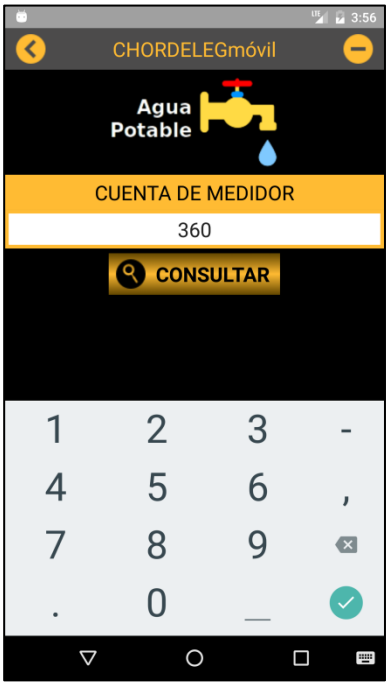


Figura 170. Consultar el agua potable en la aplicación android
Fuente: Autor

- En la figura anterior al presionar el botón “CONSULTAR”, se obtiene los datos de consulta del medidor de agua potable.

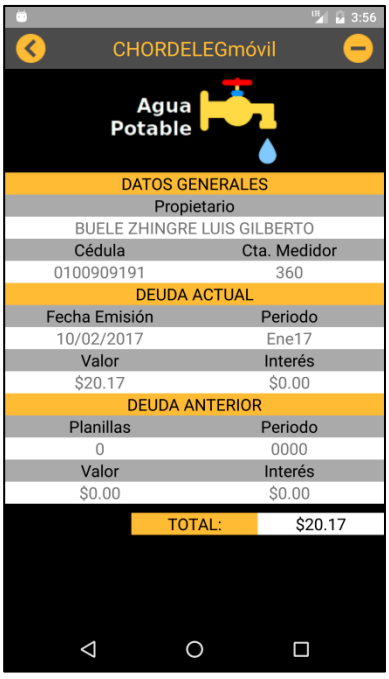


Figura 171. Reporte de consulta del agua potable en la aplicación android
Fuente: Autor

- En el menú principal se da un clic en el botón “PATENTE MUNICIPAL”, y se ingresa a consultar una patente de ejemplo.

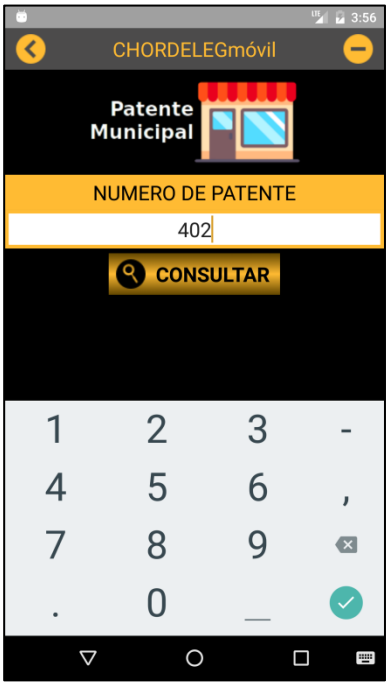


Figura 172. Consultar la patente municipal en la aplicación android
Fuente: Autor

- En la figura anterior al presionar el botón “CONSULTAR”, se obtiene los datos de consulta de la patente municipal.

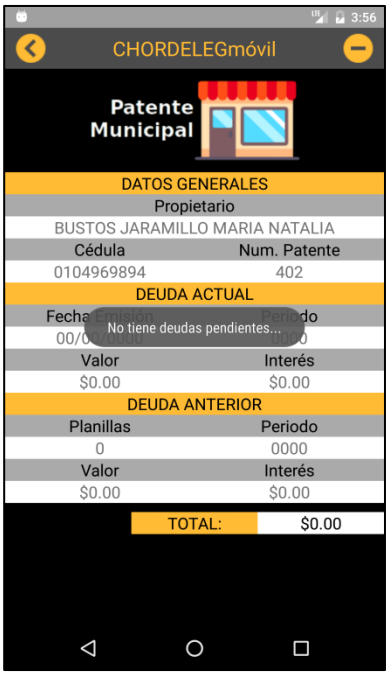


Figura 173. Reporte de consulta de la patente municipal en la aplicación android
Fuente: Autor

- Además de realizar pruebas en las consultas de los principales tributos municipales, se realizaron pruebas que permitan observar ciertas notificaciones que la aplicación android solicita para permitir una acción determinada, en las siguientes figuras se puede observar las notificaciones mencionadas.

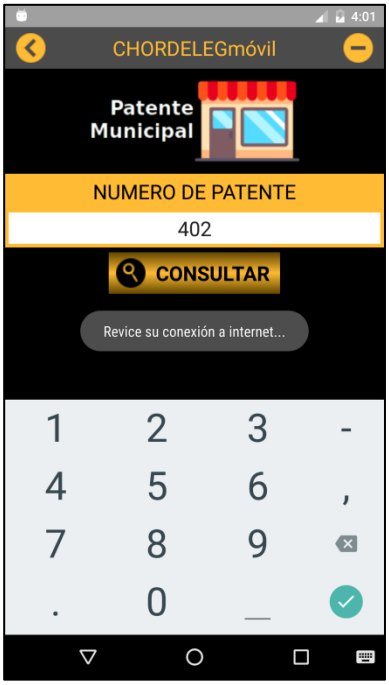


Figura 174. Notificación por falta de internet en el dispositivo móvil
Fuente: Autor

Figura 175. Notificación cuando el web service no está disponible
Fuente: Autor

Figura 176. Notificación cuando no ingresa el valor a consultar
Fuente: Autor

Figura 177. Notificación cuando la consulta ingresada no existe
Fuente: Autor

4.6.7 Generar el ejecutable del proyecto

El ejecutable de la aplicación android desarrollada tiene el formato de extensión .APK, para generarlo se tiene que dirigir al IDE de programación Android Studio y realizar los siguientes pasos:

- En primer lugar se tiene que generar un certificado que permita firmar digitalmente la aplicación, lo cual es un requisito a futuro para subir la aplicación a google play; para ello se tiene que ir a la siguiente ruta “Build>Generated Signed APK”.

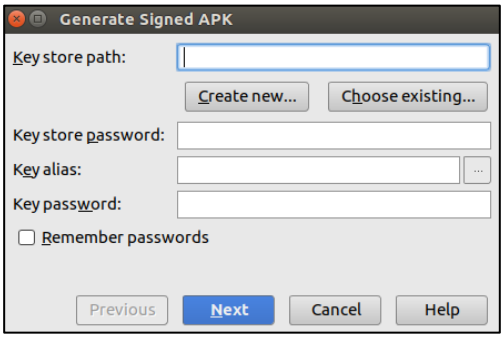


Figura 178. Generador de un APK certificado
Fuente: Autor

- Luego al dar un clic en “Create new…”, se abre una nueva ventana, se llena los datos correspondientes y se da un clic en “OK”.

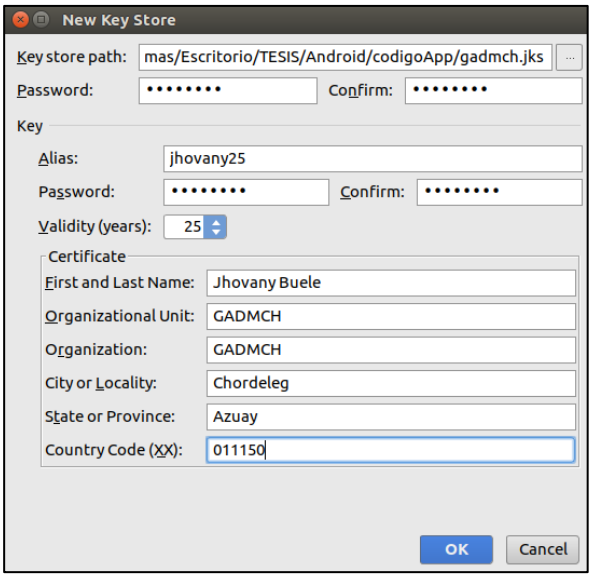


Figura 179. Crear un certificado para firmar el APK
Fuente: Autor

- Una vez generado el certificado, se carga en la ventana que genera el APK, la cual se observó anteriormente que se encuentra en la ruta “Build>Generated Signed APK”, seguidamente se da un clic en “Next”.

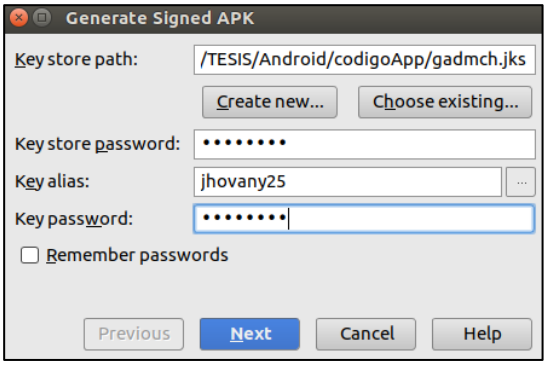


Figura 180. Cargar el certificado previo a la generación del APK
Fuente: Autor

- Por último se da un clic un “Finish” y el ejecutable de la aplicación queda generado con la extensión .APK.

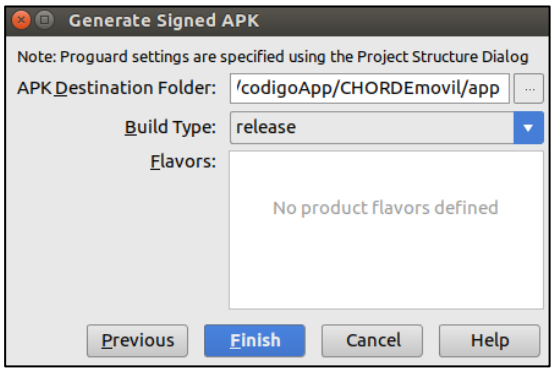


Figura 181. Terminar la creación del ejecutable de la aplicación android
Fuente: Autor

- El ejecutable generado se guarda en la misma carpeta del proyecto con el nombre “app-release.apk”.

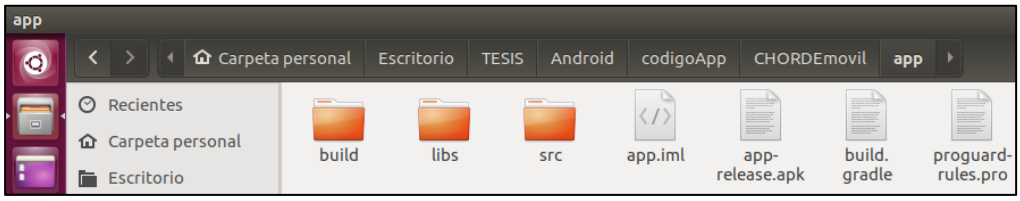


Figura 182. Carpeta donde se guarda el ejecutable de la aplicación android
Fuente: Autor

CAPITULO V

5 Estudio y diseño de calidad de servicio (QOS)

5.1 Concepto

Calidad de servicio se entiende por las tecnologías existentes que garantizan una cierta calidad para diferentes servicios de la red. Por ejemplo en una red se puede garantizar un nivel de ancho de banda, un tiempo de espera reducido, una reducción de perdida de paquetes, entre otros; para garantizar un rendimiento óptimo en servicios como voip, bases de datos, web service, etc (Martinez, s.f.).

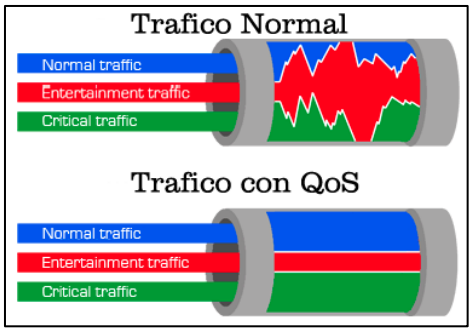


Figura 183. Tráfico de datos con y sin Calidad de Servicio
Fuente: http://www.blazar.mx/hardware.html

5.2 Convergencia de red y QoS

Una red convergente soporta diferentes tipos de aplicaciones a la vez dentro de una misma red, tales como video, voz, datos, etc.; dichas aplicaciones se caracterizan por sus diferentes requerimientos específicos de ciertos factores, entre ellos se tiene los siguientes:

- **Ancho de banda (Bandwidth):** Es una medida de la capacidad de transmisión de datos y se refiere al número de bits por segundo que pueden viajar a través de un medio. Dicha capacidad se ve reducida por factores tales como el retardo. Si se aumenta el ancho de banda se puede transmitir más datos, pero a la vez implica un incremento en el costo (Molina, s.f.).
- **Retardo (Delay):** Es la variación temporal y/o retraso al llegar los flujos de datos a su destino. Una característica que se puede observar es en aplicaciones como la

videoconferencia. En aplicaciones donde se tiene voz sobre IP, es necesario que en las políticas de QoS de la red, garanticen el retardo al mínimo requerido (María Molina, s.f.).

- **Variación del retardo (Jitter):** Es da cuando los paquetes transmitidos en una red no llegan a su destino en un debido orden o en la base de tiempo determinada, es decir, varían en latencia. Una solución ante el jitter es la utilización de buffers en el receptor (Molina, s.f.).
- **Pérdida de paquetes (Packet Loss):** Indica el número de paquetes perdidos durante la transmisión. Normalmente se mide en tanto por ciento (Molina, s.f.).

5.3 Modelos de QoS

Actualmente se puede encontrar 3 modelos de aplicación de calidad de servicios para redes de datos:

- **Best-Effort:** No se discrimina ningún tipo de tráfico y se brinda el mejor soporte posible desde la infraestructura (Gerometta, 2010).

  Es aplicado en Internet y por defecto lo tiene aplicado toda red que no posee políticas explícitamente definidas. No garantiza ningún tratamiento o recurso específico a ningún flujo de información. Todo paquete es tratado de igual forma, no existe un tratamiento preferencial (Gerometta, 2010).

  Entre sus características principales se tiene las siguientes:

  - Altamente escalable (Gerometta, 2010).
  - No requiere mecanismos o configuraciones especiales (Gerometta, 2010).
  - No garantiza recursos ni diferencia ningún tipo de servicio (Gerometta, 2010).

- **IntServ:** En aplicaciones donde el tráfico requiere un tratamiento diferencial, señalizan la red para requerir y garantizar los recursos necesarios para el adecuado funcionamiento de la aplicación (Gerometta, 2010).

  Garantiza las condiciones de operación de cada una de las sesiones que se establecen. Su objetivo es garantizar recursos disponibles a lo largo de una ruta para una aplicación específica. Antes de iniciarse propiamente la sesión de la aplicación se marca la ruta para verificar la disponibilidad de los recursos necesarios que permitan garantizar el flujo de datos, una vez que la aplicación realiza la reserva de recursos, esta se mantiene hasta que se levante la reserva de recursos (Gerometta, 2010).

  Entre sus características principales se tiene las siguientes:

  - Negocia condiciones específicas de calidad de servicio antes de que se inicie la comunicación (Gerometta, 2010).
  - Una vez hecha la reserva, la aplicación cuenta con los recursos reservados indistintamente de la situación de tráfico de la red (Gerometta, 2010).
  - Puede adecuarse a demandas específicas y diferentes de cada tipo de tráfico o aplicación (Gerometta, 2010).
  - La reserva de recursos se realiza para cada flujo de información en particular (Gerometta, 2010).
  - Utiliza los servicios de RSVP (Resource Reservation Protocol) (Gerometta, 2010).
  - No es escalable en grandes redes o implementaciones muy complejas (Gerometta, 2010).

- **DiffServ**: La infraestructura de la red es la que reconoce los diferentes tipos de tráfico y aplica políticas diferenciadas para cada clase de tráfico (Gerometta, 2010). Es más escalable y flexible en su implementación, con recursos garantizados de modo genérico y no por flujos o sesiones. Permite garantizar diferentes condiciones de servicio para diferentes tipos de tráfico a través de toda la red (Gerometta, 2010).

  Entre sus características principales se tiene las siguientes:

  - No requiere señalización previa (Gerometta, 2010).
  - No permite garantizar condiciones de tráfico extremo a extremo (Gerometta, 2010).
  - Es muy flexible y escalable (Gerometta, 2010).
  - Divide el tráfico en clases en función de los requerimientos (Gerometta, 2010).
  - Cada paquete recibe el tratamiento que se ha definido para la clase a la cual pertenece el paquete (Gerometta, 2010).
  - A cada clase se le puede asignar un diferente nivel de servicio y con ello diferentes recursos (Gerometta, 2010).
  - La asignación de recursos se hace salto por salto en cada dispositivo de la red y no para una ruta específica (Gerometta, 2010).
  - El mecanismo de implementación es relativamente complejo (Gerometta, 2010).

5.4 Diseño de QoS con enfoque al web service desarrollado

Revisada la teoría relacionada con calidad de servicio QoS, se conoce que el router principal del GAD Municipal de Chordeleg es un Mikrotik, por lo cual se describe a continuación el método que usa para implementar calidad de servicio.

5.4.1 Método de implementación de QoS en mikrotik

Mikrotik está relacionado con el método diffserv, razón por la cual de manera general realiza los siguientes pasos:

- Marcación de conexiones.
- Marcación de paquetes.
- Priorización de tráfico.

Estos pasos son aplicados en base a políticas determinadas, cabe indicar que QoS se implementa y configura según las necesidades de cada caso de estudio. A continuación describe los pasos mencionados anteriormente:

- **Marcación de conexiones:** Dentro del sistema operativo RouterOS de Mikrotik, se tiene un "Firewall", al cual se puede crear reglas "Mangle" las mismas que permiten realizar el marcado de conexiones por http, ssh, icmp, etc., según las políticas determinadas, las cuales pueden ser por puerto, dirección ip, interface, etc.

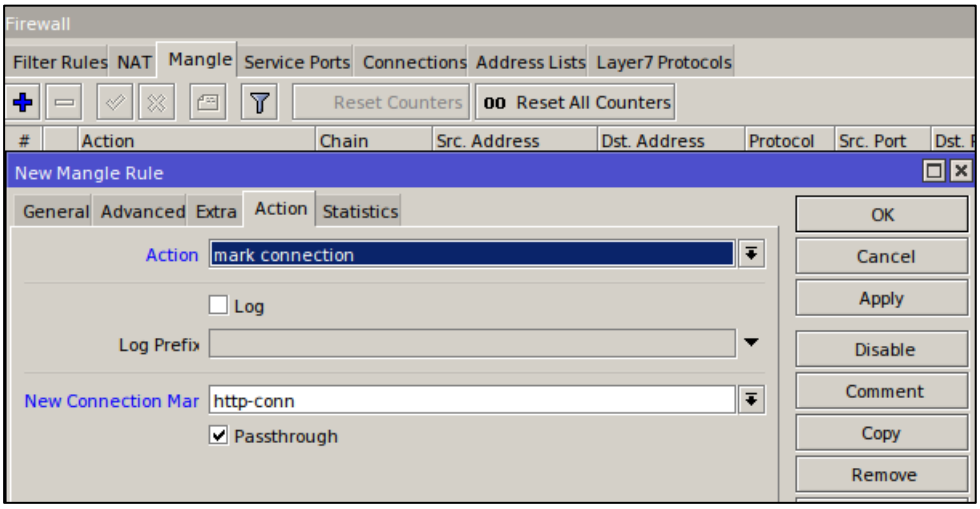


Figura 184. Regla mangle de mikrotik para marcar conexiones
Fuente: Autor

- **Marcación de paquetes:** De igual forma, en el "Firewall" de mikrotik, se puede crear reglas "Mangle" a diferencia que en este caso, se crea el marcado de paquetes según la conexión marcada.

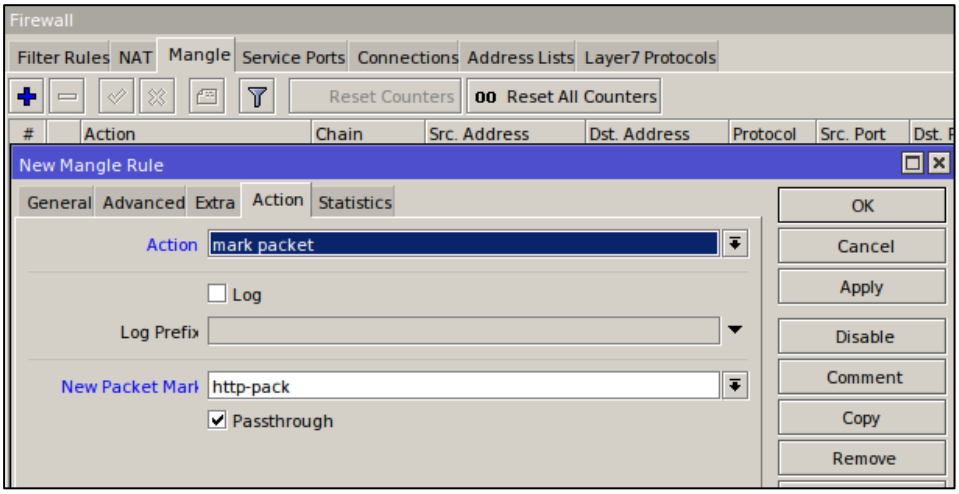


Figura 185. Regla mangle de mikrotik para marcar paquetes
Fuente: Autor

- **Priorización de tráfico:** La gestión de ancho de banda de mikrotik está basada en HTB (Hierarchical Token Bucket), lo cual permite crear colas jerárquicas y priorizar cada paquete marcado. Para ello se tiene dentro mikrotik el "Queue Tree", donde se crea las colas que permiten asignar un ancho de banda y priorizar el los paquetes marcados en las reglas mangle como se observó en la figura anterior.

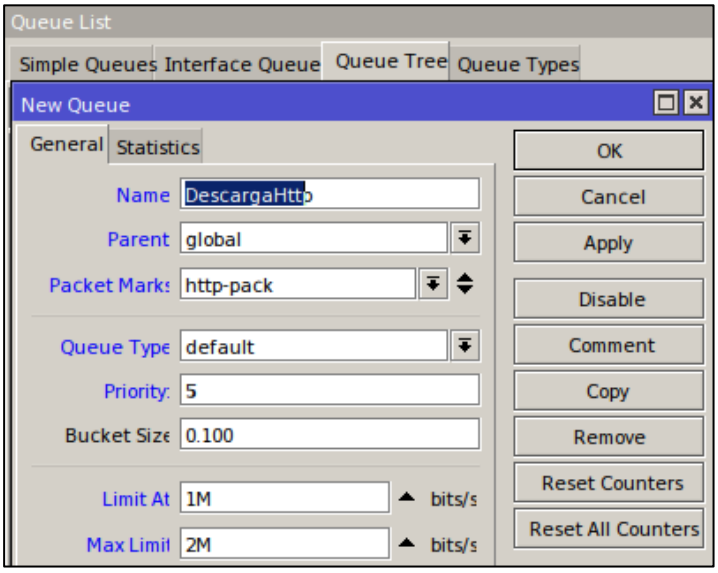


Figura 186. Cola de mikrotik para gestionar el tráfico de paquetes marcados
Fuente: Autor

5.4.2 Determinar el tráfico del GAD Municipal de Chordeleg

En la sección 1.1.4 y 1.1.5, de acuerdo al levantamiento de información del GAD Municipal de Chordeleg se obtuvieron los servicios que dispone la institución, lo cual ha servido de mucho para determinar el tráfico que fluye a través de su red. En la siguiente tabla se describe el tráfico conocido por los servicios que se dispone en la red del GAD Municipal de Chordeleg.

Tabla 51. Trafico por servicios en el GAD Municipal de Chordeleg

Descripción	Servicio	Protocolo	Puertos
Correo Institucional	POP, IMAP, SMTP	TCP	110, 143, 995, 993, 22
Sistema SIMM	MySQL	TCP	3306
Sistema SIM	PostgreSQL	TCP	5432
Sistema SIGAME	SQL server	TCP	1433
Internet	HTTP, HTTPs	TCP	80,443
Web Service	Node	TCP	3001

Fuente: Autor

A más del tráfico anterior se tienen otros servicios conocidos como las consultas a servidores para resoluciones de nombres en internet, herramientas ping para determinar conexiones de un host en la red, administración remota de servidores, administración gráfica de equipos mikrotik y descarga de archivos con programas P2P. De igual manera a continuación se tiene una tabla detallada con estos servicios:

Tabla 52. Trafico por servicios adicionales en el GAD Municipal de Chordeleg

Descripción	Servicio	Protocolo	Puertos
Resolución de nombres	DNS	UDP	53
Herramientas de ping	ICMP	no hace falta describir en mikrotik	
Conexión remota a servidores	SSH	TCP	22
Conexión remota a mikrotik	WinBox	TCP	8229
Descargas P2P	P2P	no hace falta describir en mikrotik	

Fuente: Autor

5.4.3 Establecer niveles de prioridad por cada servicio

Una vez conocido el tráfico que se desea priorizar, se clasifica según las necesidades de la red del GAD Municipal de Chordeleg; a más de ello es necesario conocer que el router mikrotik dispone de 8 niveles para priorizar el tráfico, razón por la cual se crea 8 grupos para tal priorización. En la siguiente tabla se describe con mayor detalle la agrupación de cada tráfico y la priorización elegida.

Tabla 53. Priorización del tráfico en el GAD Municipal de Chordeleg

Priorización	Servicio	Protocolo	Puertos
P1 - ADMINISTRACIÓN RED Y DNS	ICMP, SSH, WinBox	TCP	22, 8291
	DNS	UDP	53
P2 - BASES DE DATOS	MySQL, PostgreSQL, SQLserver	TCP	3306, 5432, 1433
P3 - WEB SERVICE	Node	TCP	3001
P4 - NAVEGACION WEB	HTTP, HTTPs	TCP	80, 443
P5 - CORREO	POP, IMAP, SMTP	TCP	110, 143, 995, 993, 25, 587
P6 - OTROS	Servicios Restantes	TCP	Puertos Restantes
P7 - DESCARGAS>50M	todos los conexiones > 50M		
P8 - P2P	no hace falta describir en mikrotik		

Fuente: Autor

En la tabla anterior se puede ver que dentro del nivel 1 están las herramientas de administración de la red con el fin de que el departamento de sistemas no tenga problemas de conexión para reparar problemas urgentes en los servidores y el router principal, a más de ello se tiene el servicio DNS, el cual está en este grupo para acelerar las consultas en internet.

En el nivel 2 se da una priorización a las bases de datos que trabajan con los sistemas de recaudación y administración de los tributos municipales, también se encuentra el sistema financiero; los cuales son de gran importancia para la institución.

En el nivel 3 se tiene el web service al cual se asigna esta priorización en vista que se desea dar un servicio de calidad a la ciudadanía para las consultas móviles de los principales tributos municipales, y al ser conexiones sumamente pequeñas no afecta en absoluto a los siguientes niveles.

En el nivel 4 se tiene todo tipo de navegaciones a páginas web, en este grupo están todos los clientes que hacer uso del servicio de internet.

En el nivel 5 se ha considerado priorizar al correo interno de la institución que si bien es cierto se usa en menor cantidad que la navegación web, pero no deja de ser importante para comunicaciones externas de la institución.

En el nivel 6 de manera general se ha clasificado al resto de servicios que no requieran de una priorización.

Antes de seguir con el siguiente nivel, se podría decir que hasta aquí termina la configuración de calidad de servicio en la red del GAD de Chordeleg, sin embargo se ha considerado dos niveles más que ayudaran de mucho a mejorar la red.

En el nivel 7, se ha considerado tener aquellas descargas que superen los 50Megas, con esto se limita aquellos usuarios que se cuelgan en la red para descargar sobre todo archivos de entretenimiento.

En el nivel 8, que es el último, se tiene todos los programas P2P, de lo que se conoce solo el departamento de sistemas tiene este tipo de programas para descargar herramientas libres para mantenimiento de la red.

CAPITULO VI

6 Implementación del sistema

6.1 Implementación del web service

6.1.1 Características del servidor de producción

El GAD Municipal de Chordeleg dispone de un CPU con las siguientes características:

Tabla 54. Características del servidor de producción para implementar el web service

Ítem	Características
Marca	HP
Modelo	ProLiant ML110 G5
Procesador	Intel Xeon 2.33GHz, 64bits
Disco Duro	500GB
Interface	NetXtreme Gigabit ethernet
Memoria Ram	2GB

Fuente: Autor

El cual será el servidor de producción para la implementación del web service desarrollado en el capítulo 3, dentro de la red LAN del GAD Municipal de Chordeleg.

Figura 187. Servidor HP ProLiant Ml110 G5 del GAD Municipal de Chordeleg
Fuente: Autor

6.1.2 Instalación del sistema operativo

El sistema operativo a instalar, será el mismo que la pc de desarrollo, a diferencia que se descarga la versión para servidores, es decir Ubuntu Server 16.10. A continuación se detalla los pasos para instalar:

- En primer lugar, se descarga Ubuntu Server 16.10 de la página oficial de Ubuntu https://www.ubuntu.com/download, donde se encuentra las versiones para 32 y 64 bits, el servidor de producción soporta 64 bits, por lo tanto se descarga aquella versión.

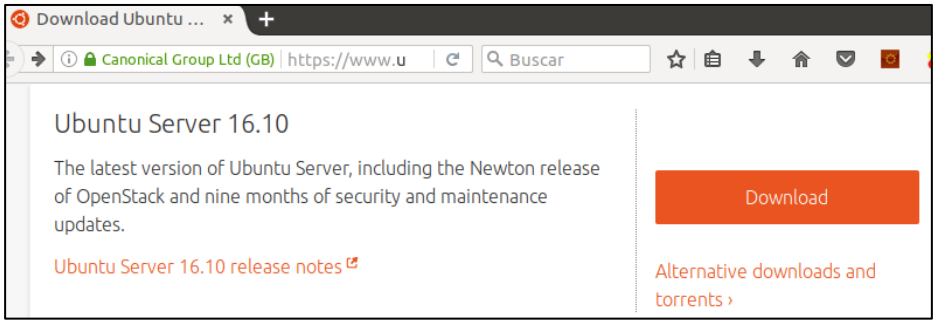


Figura 188. Página oficial de descarga de Ubuntu Server
Fuente: Autor

- Una vez descargado el sistema operativo se graba en un DVD, luego se inserta en el DVD del servidor de producción y se arranca desde la unidad de CD/DVD, una vez que inicia el asistente, se da un clic en “Español”:

Figura 189. Iniciar el asistente de instalación de Ubuntu Server 16.10 en español
Fuente: Autor

- Luego se selecciona la primera opción, que es “Instalar Ubuntu Server”

Figura 190. Instalación de Ubuntu Server 16.10
Fuente: Autor

- Buscar y seleccionar la ubicación “Ecuador”, y se da un enter en el teclado.

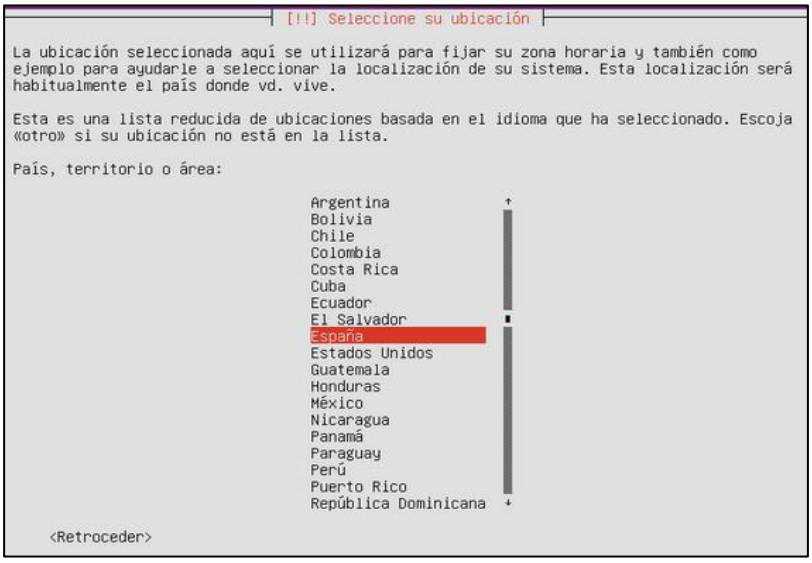


Figura 191. Selección de ubicación de Ubuntu Server 16.10
Fuente: Autor

- Continuar con la configuración del teclado, para no detectar la disposición del teclado se tiene que seleccionar “No” y dar un enter en el teclado.

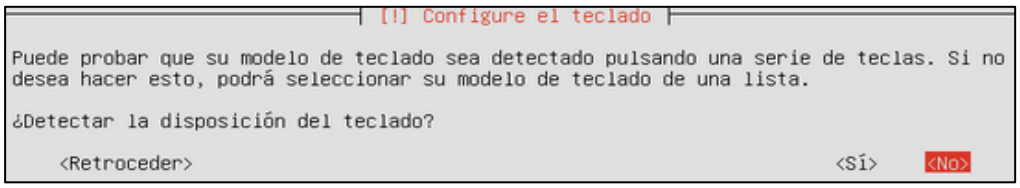


Figura 192. No detectar disposición de teclado de Ubuntu Server 16.10
Fuente: Autor

- Luego se selecciona el país de origen del teclado que es español.

Figura 193. País de origen del teclado en Ubuntu Server 16.10
Fuente: Autor

- Seleccionar la distribución del teclado en español.

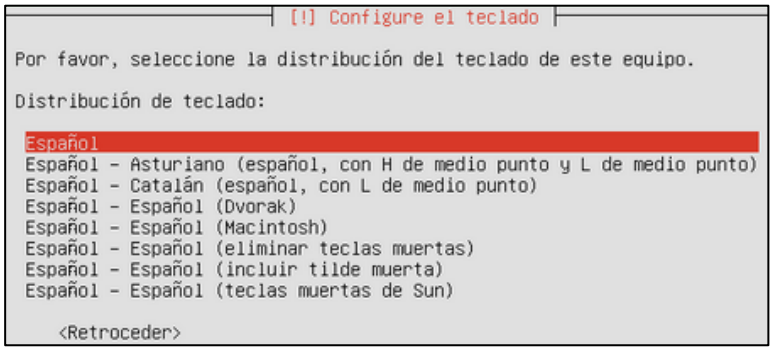


Figura 194. Distribución del teclado en español en Ubuntu Server 16.10
Fuente: Autor

- Ingresar el nombre de la máquina, la cual se definió como “ubuntuWS”, luego se selecciona “Continuar” y dar un enter en el teclado.

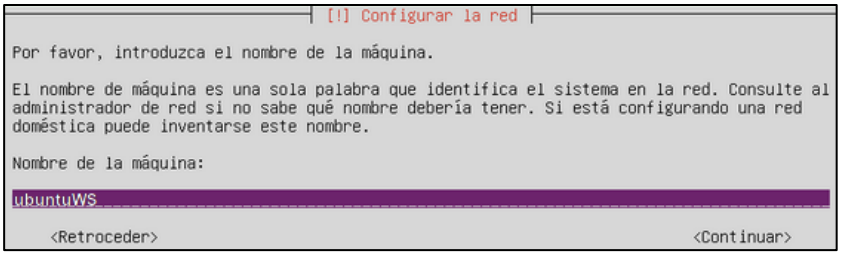


Figura 195. Nombre de la máquina con Ubuntu Server 16.10
Fuente: Autor

- Ingresar el nombre completo del usuario “Analista de Sistemas”, seleccionar “Continuar” y dar un enter en el teclado.

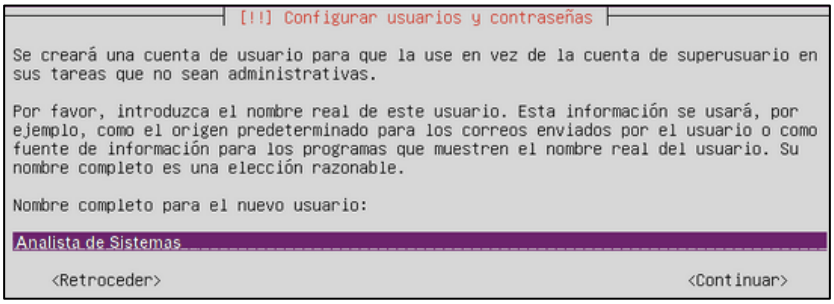


Figura 196. Nombre completo del usuario de Ubuntu Server 16.10
Fuente: Autor

- Ingresar el usuario “sistemas” para la cuenta, seleccionar “Compartir” y dar un enter con el teclado.

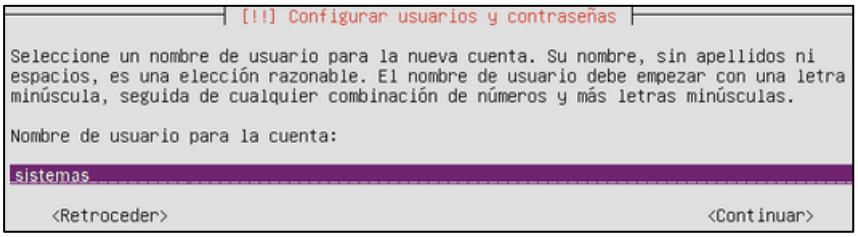


Figura 197. Nombre del usuario para ingresar a Ubuntu Server 16.10
Fuente: Autor

- Ingresar una contraseña para el usuario creado en la figura anterior, seleccionar “Compartir” y dar un enter en el teclado.

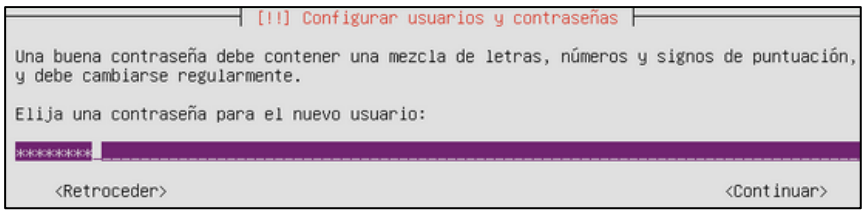


Figura 198. Contraseña para el usuario de Ubuntu Server 16.10
Fuente: Autor

- Nuevamente ingresar la contraseña para su verificación, seleccionar “Compartir” y dar un enter en el teclado.

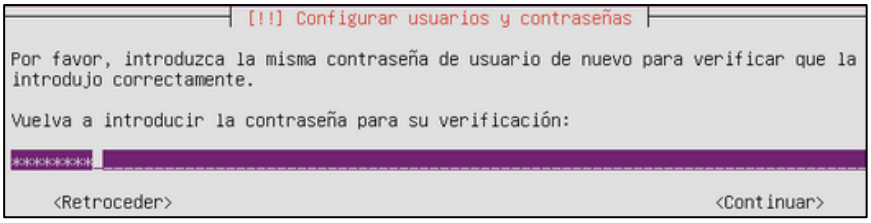


Figura 199. Verificación de la contraseña del usuario de Ubuntu Server 16.10
Fuente: Autor

- Para el cifrado de la carpeta personal, seleccionar que “No” y dar un enter en el teclado.

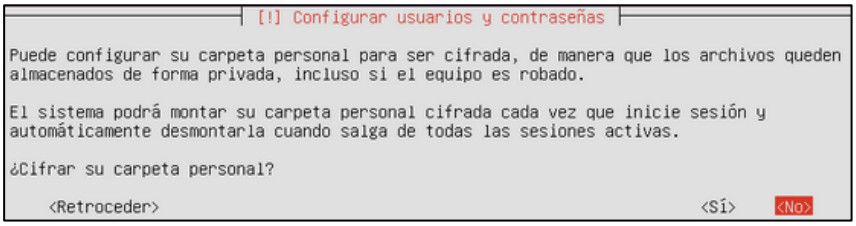


Figura 200. Cifrado de carpeta personal Ubuntu Server 16.10
Fuente: Autor

- Para la configuración del reloj seleccinar “Si” por la zona horaria detectada y dar un enter en el teclado.

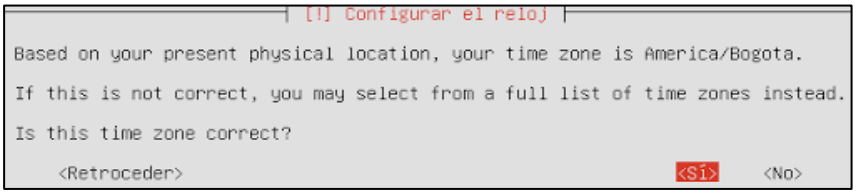


Figura 201. Zona horaria en Ubuntu Server 16.10
Fuente: Autor

- Utilizar todo el disco, en vista que el servidor está destinado únicamente para el web service, para ello seleccionar “Guiado – utilizar todo el disco”.

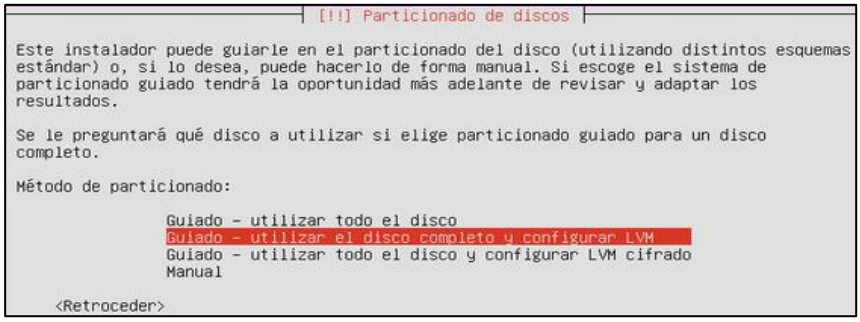


Figura 202. Partición de discos para la instalación de Ubuntu Server 16.10
Fuente: Autor

- Aceptar la advertencia de instalación de discos seleccionando “Si”, luego se da un enter en el teclado y continúa.

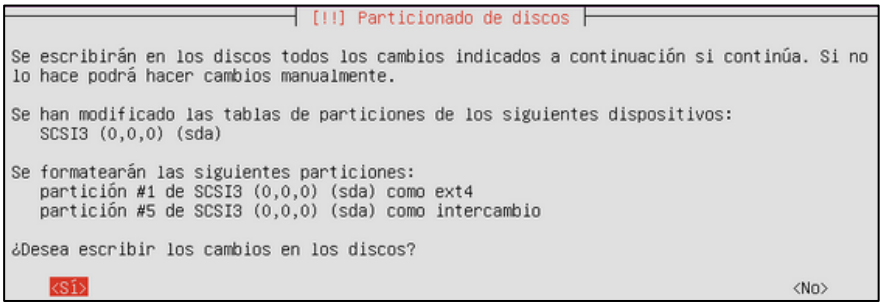


Figura 203. Aceptar la partición de discos seleccionada para Ubuntu Server 16.10
Fuente: Autor

- Esperar a que termine la instalación del sistema.

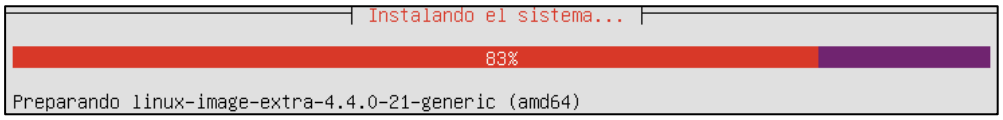


Figura 204. Instalación del sistema Ubuntu Server 16.10
Fuente: Autor

- Luego que termine la instalación se tiene la configuración del proxy, dejar en blanco ya que el GAD Municipal de Chordeleg no tiene configurado nada al respecto, seleccionar “Continuar” y dar un enter en el teclado

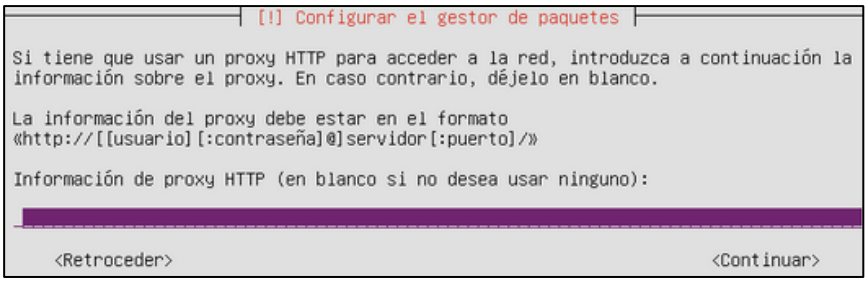


Figura 205. Configuración del proxy para instalar Ubuntu Server 16.10
Fuente: Autor

- Por defecto se deja la instalación sin actualizaciones automáticas y se da un enter en el teclado.

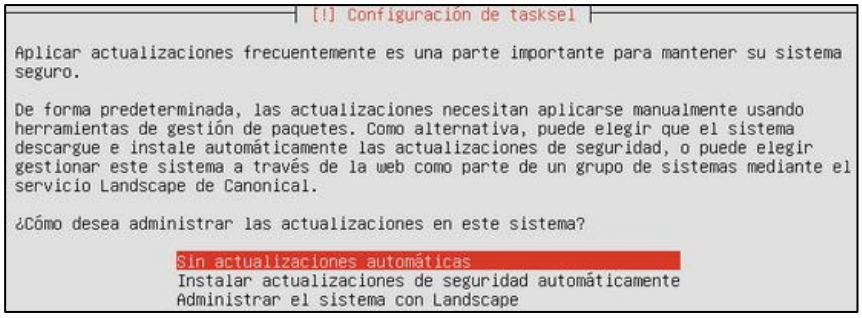


Figura 206. Instalación sin actualizaciones automáticas en Ubuntu Server 16.10
Fuente: Autor

- Seguidamente se pasa a la instalación de programas con el sistema básico, aquí únicamente se ha seleccionado OpenSSH server para conectarse de manera remota al servidor; luego de ello seleccionar “Continuar” y dar un enter en el teclado.

Figura 207. Selección de programas para instalar con en Ubuntu Server 16.10
Fuente: Autor

- Esperar que termine la instalación de los programas seleccionados.

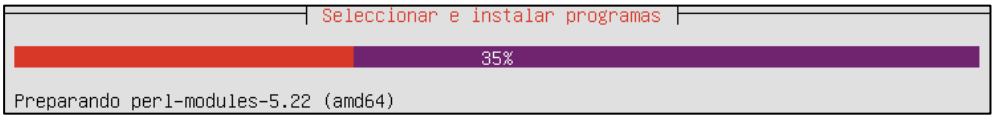


Figura 208. Instalación de programas con en Ubuntu Server 16.10
Fuente: Autor

- Luego tiene que instalar el cargador de arranque GRUB, seleccionando “Si” y dando un enter en el teclado.

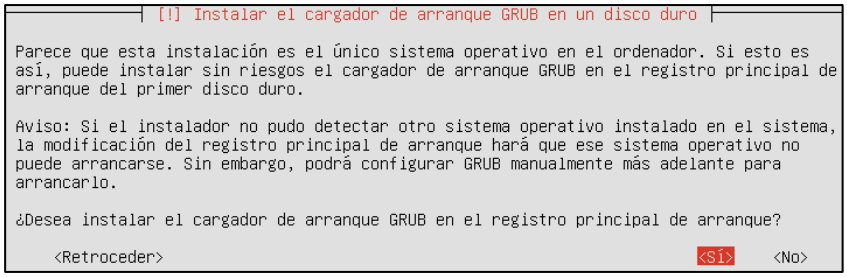


Figura 209. Instalación del GRUB de Ubuntu Server 16.10
Fuente: Autor

- Finalmente al terminar la instalación, se retira el DVD de la unidad de CD/DVD del servidor de producción, se da un enter en el teclado y el equipo se reinicia.

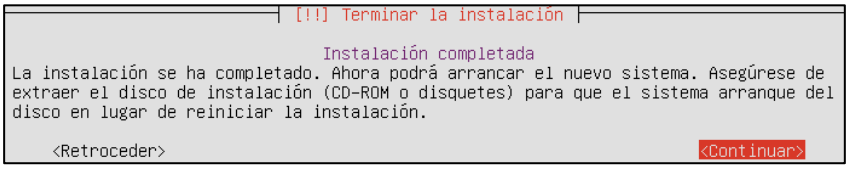


Figura 210. Finalizar la instalación de Ubuntu Server 16.10
Fuente: Autor

- Luego de reiniciar el equipo, se ingresa los datos de la cuenta configurada anteriormente y ya se inicia Ubuntu Server 16.10.

Figura 211. Ubuntu Server 16.10
Fuente: Autor

6.1.3 Configuración de la dirección ip

Recordando la tabla 5, donde el GAD Municipal tiene asignada una subred para los servidores, se busca una ip disponible, la misma que es 192.168.2.53 y se configura en el servidor de producción para el web service; la configuración de la ip en el servidor de producción se realizó de la siguiente manera:

- El archivo que contiene toda la configuración de la red es el “interfaces” y se encuentra en la ruta “/etc/network”, por lo tanto se edita con el comando “nano” que viene instalado en Ubuntu Server.

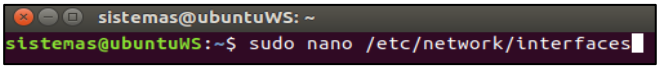


Figura 212. Editar la red en Ubuntu Server
Fuente: Autor

- Una vez que se tiene abierto el archivo, se edita con los siguientes datos:

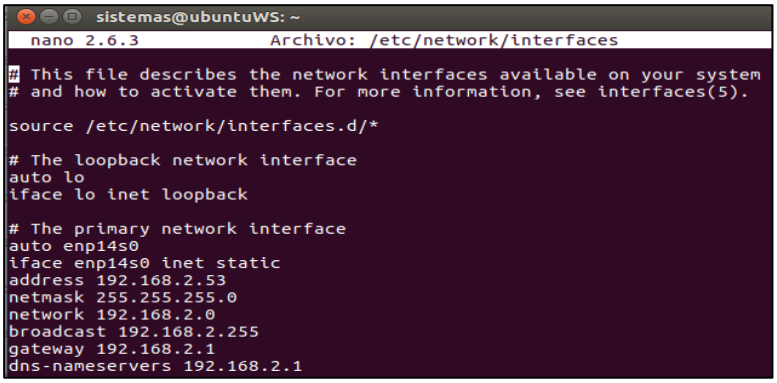


Figura 213. Configurar la red en Ubuntu Server
Fuente: Autor

- Por último se reinicia la red para que los cambios tengan efecto:

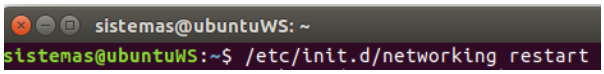


Figura 214. Reiniciar el servicio de red en Ubuntu Server
Fuente: Autor

6.1.4 Instalación del servidor de aplicaciones

Se conoce que el servidor de aplicaciones es Node.js, por lo tanto se realiza los siguientes pasos para su instalación en Ubuntu Server 16.10.

- Actualizar el repositorio de Ubuntu.

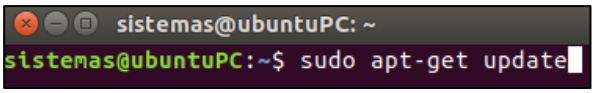


Figura 215. Actualizar el repositorio de Ubuntu Server
Fuente: Autor

- Instalar las librerías dependientes que requiere Node.js.

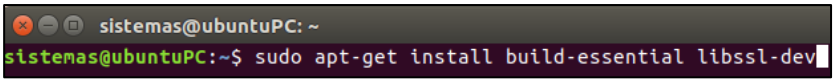


Figura 216. Librerías para instalar Node.js en Ubuntu Server
Fuente: Autor

- Instalar NVM, el cual sirve como gestor de instalación de Node.js.

Figura 217. Instalar NVM gestor de la instalación de Node.js en Ubuntu Server
Fuente: Autor

- Se procede a instalar Node.js en su versión estable 6.9.1, que es la más actual al momento de realizar el presente proyecto.

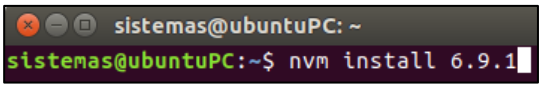


Figura 218. Instalar Node.js en Ubuntu Server
Fuente: Autor

- Finalmente se puede comprobar la versión instalada realizando lo siguiente:

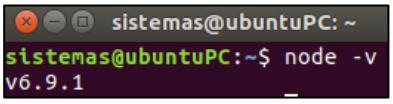


Figura 219. Comprobar la versión de Node.js instalada en Ubuntu Server
Fuente: Autor

En este punto es necesario indicar que Node.js se instala conjuntamente con el gestor de paquetes "NPM", el cual sirve para instalar y agregar módulos adicionales a Node.js, de igual manera se puede comprobar su versión instalada realizando lo siguiente:

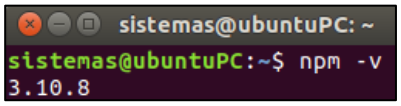


Figura 220. Comprobar la versión de NPM, gestor de paquetes de Node.js
Fuente: Autor

6.1.5 Instalar el web service desarrollado

En la sección 3.7.7, se había indicado que el ejecutable del web service desarrollado no es más que la carpeta principal que contiene todos los archivos con el código fuente

desarrollado y que para su implementación simplemente se copia dicha carpeta en alguna dirección del servidor de producción. Por lo tanto se realizaron los siguientes pasos:

- Desde la PC de desarrollo, copiar toda la carpeta del proyecto del web service al servidor de producción; para ello haciendo uso del protocolo "ssh" se ejecutó el siguiente comando:

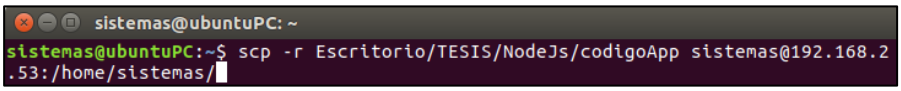


Figura 221. Copiar el web service desarrollado desde la PC de desarrollo hacia el servidor de producción
Fuente: Autor

- Luego de copiar todo el proyecto, se recurrió a la tabla 5, donde se indica que el servidor de producción MySQL de la base de datos SIMM del GAD Municipal de Chordeleg tiene la dirección ip 192.168.2.50, y anteriormente ya se había asignado al servidor de producción del web service la dirección ip 192.168.2.53; por lo tanto se tiene que editar el archivo de configuración "config.json" del web service desarrollado en vista que ya estaba configurado para trabajar a nivel local, razón por la cual se edita el archivo "config.json" haciendo uso del siguiente comando:

Figura 222. Editar el archivo config.json en el servidor de producción
Fuente: Autor

- Una vez que se abra el archivo "config.json", finalmente se configura los siguientes datos:

Figura 223. Configurar los datos del archivo config.json para producción
Fuente: Autor

6.1.6 Configurar el web service como servicio

Es necesario que el web service esté configurado como servicio, con el fin de que inicie o se restablezca el servicio cuando exista cortes de energía o congelamientos del mismo; para ello se realizó los siguientes pasos:

- En Node.js se tiene el módulo forever, que permitirá arrancar las aplicaciones de extensión .js como servicios, para instalar forever se ejecuta lo siguiente:

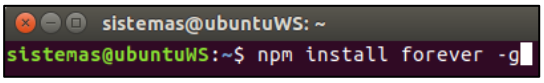


Figura 224. Instalar forever en Node.js
Fuente: Autor

- En un inicio Node.js se instaló bajo el usuario “sistemas”, sin embargo para ejecutarlo como servicio, es necesario darle permisos a root para ejecutar aplicaciones de Node.js, esto se realiza con el siguiente comando:

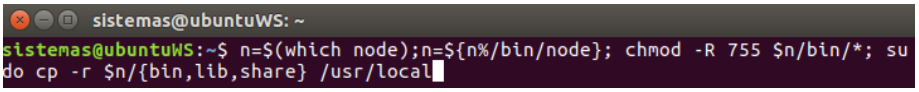


Figura 225. Permisos a root para ejecutar aplicaciones de Node.js
Fuente: Autor

- Luego se necesita editar el crontab, el cual es un archivo que se puede ingresar el comando necesario para que arranque automáticamente la aplicación con forever; se edita el crontab con el siguiente comando:

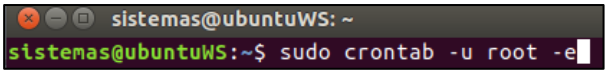


Figura 226. Editar el contrab en Ubuntu Server
Fuente: Autor

- Una vez abierto el contrab, se agrega al final del archivo, la línea que no está comentada en la siguiente figura, cabe recordar que la aplicación se encuentra en la ruta "/home/sistemas/codigoApp/app.js".

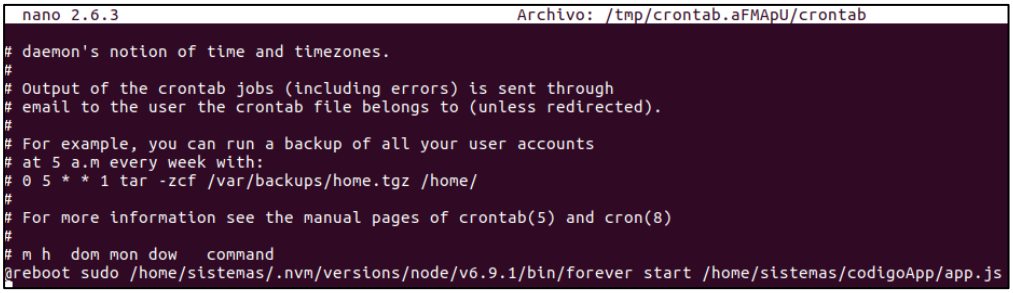


Figura 227. Ingresando el comando para iniciar el web service como servicio
Fuente: Autor

- Finalmente reiniciar el servidor de producción con el siguiente comando:

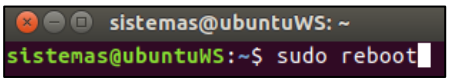


Figura 228. Reiniciando el servidor de producción del web service
Fuente: Autor

Hasta este punto el web service es accesible únicamente desde la red LAN del GAD Municipal de Chordeleg; en la siguiente sección se configurarán los equipos respectivos para tener acceso desde la red externa como internet.

6.1.7 Acceder al web service desde internet

Para desplegar el web service en internet, se tiene que configurar el router principal para que todas las peticiones que se hagan desde internet a través de la dirección pública 181.211.253.98:3001, se redirijan el servidor de producción del web service que tiene la dirección ip 192.168.2.53:3001, para ello se realiza las siguientes actividades:

- Ingresar al router principal del GAD Municipal de Chordeleg, mediante la aplicación winbox.

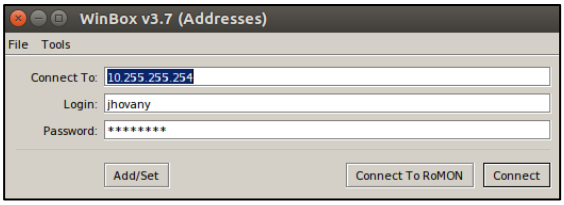


Figura 229. Ingresando al router principal a través de winbox
Fuente: Autor

- Dirigirse a “IP>Firewall>NAT”, un clic en +, y en la pestaña “General” configurar el puerto con la dirección ip externa que se desea redirigir:

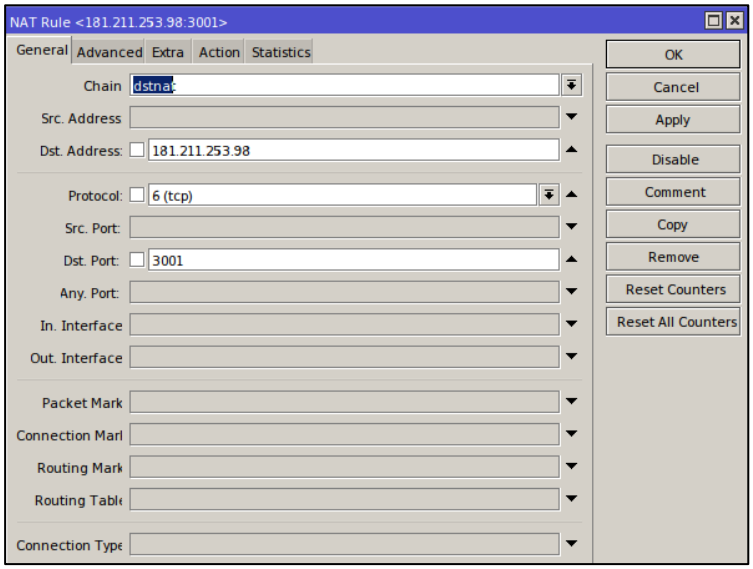


Figura 230. Redirección de puertos desde una ip externa
Fuente: Autor

- En la misma ventana anterior, ubicarse en la pestaña “Action” y configurar el puerto con la dirección ip interna a la cual se va a redirigir las peticiones externas:

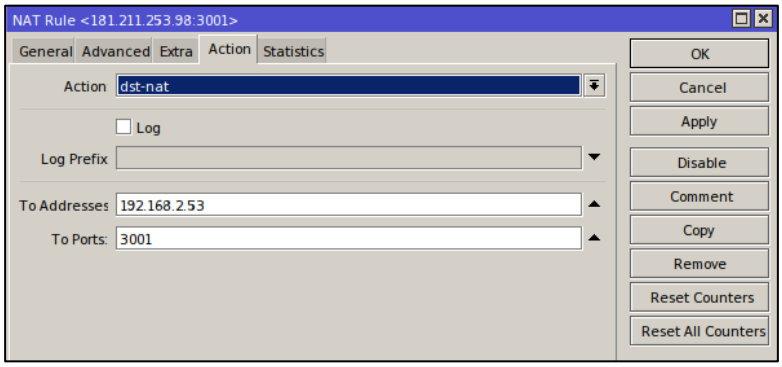


Figura 231. Redirección de puertos hacia una ip interna
Fuente: Autor

- Finalmente quedaría la regla de la siguiente manera:

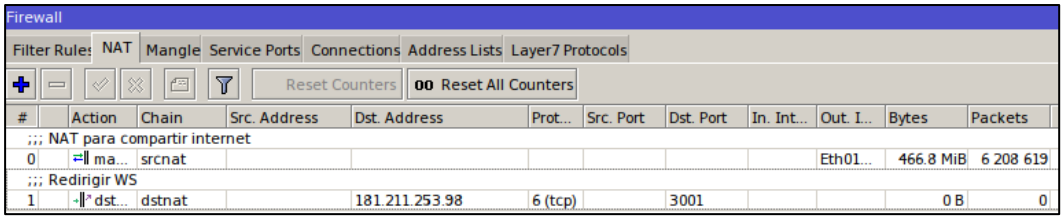


Figura 232. Redirección de puertos hacia una ip interna
Fuente: Autor

Con estas configuraciones realizadas en el router principal, ya es posible acceder al web service desde internet.

6.1.8 Asignar un dominio al web service

Si bien es cierto con las configuraciones de la sección anterior ya es posible hacer uso del web service desde internet, esto se lo hace únicamente conociendo la dirección ip pública; por lo tanto se va asignarle un dominio exclusivo para el web service, para acceder al servicio a través de un nombre, el cual llevará el nombre a partir del dominio principal llegando a ser un subdominio de la página web http://www.chordeleg.gob.ec del GAD Municipal de Chordeleg, a continuación se describe los pasos realizados:

- La página del GAD Municipal de Chordeleg. está alojada en un servidor externo, y su administración se realiza a través de la plataforma “CPanel”, para ello se tiene que dirigir a la página de administración http://www.chordeleg.gob.ec/cpanel, en la cual se ingresa con el usuario y contraseña facilitados por el GAD Municipal de Chordeleg:

Figura 233. CPanel para administrar la página http://www.chordeleg.gob.ec
Fuente: Autor

- Dirigirse a la sección “Dominios” y dar un clic en “Subdominios”:

Figura 234. Ingresar al administrador de subdominios en CPanel
Fuente: Autor

- Ingresar el subdominio correspondiente, el cual se definió como “ws.chordeleg.gob.ec” y para continuar se da un clic en “Crear”.

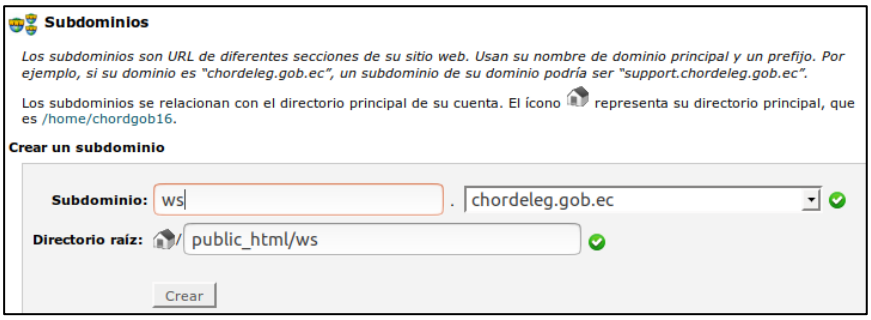


Figura 235. Crear un subdominio en CPanel
Fuente: Autor

- Luego en la sección “Dominios”, se da un clic en “Editor de zona DNS avanzado”, para redirigir las peticiones del subdominio hacia la dirección ip pública del GAD Municipal de Chordeleg:

Figura 236. Editar zona DNS en CPanel
Fuente: Autor

- Una vez que esté dentro del editor de zona DNS avanzado, finalmente se remplaza todas las ip registradas del subdominio ws.chordeleg.gob.ec, con la ip pública del GAD Municipal de Chordeleg, que es la 181.211.253.98.

ws.chordeleg.gob.ec.	14400	IN	A	181.211.253.98	Editar	Borrar
www.ws.chordeleg.gob.ec.	14400	IN	A	181.211.253.98	Editar	Borrar
webdisk.ws.chordeleg.gob.ec.	14400	IN	A	181.211.253.98	Editar	Borrar
autoconfig.ws.chordeleg.gob.ec.	14400	IN	A	181.211.253.98	Editar	Borrar
autodiscover.ws.chordeleg.gob.ec.	14400	IN	A	181.211.253.98	Editar	Borrar

Figura 237. Redirección del subdominio ws.chordeleg.gob.ec en CPanel
Fuente: Autor

6.1.9 Seguridad

En vista que el web service estará expuesto a la red externa, es necesario implementar ciertas seguridades; actualmente el GAD Municipal de Chordeleg cuenta con algunas reglas de seguridad en el firewall del router principal ante posibles ataques desde internet, en las siguientes figuras se puede observar cada una de ellas:

;;; Bloquear Webproxy Externo										
0		drop	input			6 (tcp)		3128	Eth01...		24.8 KiB	435

Figura 238. Regla para evitar ataques al Web Proxy
Fuente: Autor

;;; Bloquear DNS Caché Externo										
1		drop	input			17 (udp)		53	Eth01...		3599 B	54

Figura 239. Regla para evitar ataques al DNS Caché
Fuente: Autor

;;; Bloquear ping										
2		drop	input			1 (icmp)			Eth01...		0 B	0

Figura 240. Regla para evitar ataques de Ping
Fuente: Autor

;;; Bloquear FTP										
3		drop	input			6 (tcp)		21			0 B	0

Figura 241. Regla para evitar ataques por FTP
Fuente: Autor

;;; Proteccion VSC contra ataques via SSH										
6		drop	input			6 (tcp)		22			0 B	0

Figura 242. Regla para evitar ataques por SSH
Fuente: Autor

;;; Bloquea Lista Telnet										
11		drop	input	!192.168.2....		6 (tcp)		23			320 B	7

Figura 243. Regla para evitar ataques por Telnet
Fuente: Autor

Por lo tanto se observa que el router principal posee cierta protección ante posibles ataques desde la red externa, sin embargo para complementar esta seguridad primeramente se debe cambiar el puerto ssh de conexión remota al servidor.

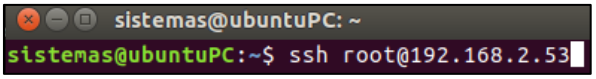


Figura 244. Ingresar de manera remota al servidor de producción
Fuente: Autor

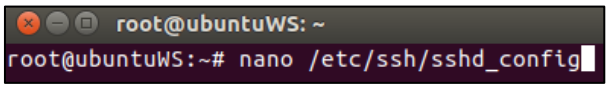


Figura 245. Comando para editar el archivo de configuración ssh
Fuente: Autor

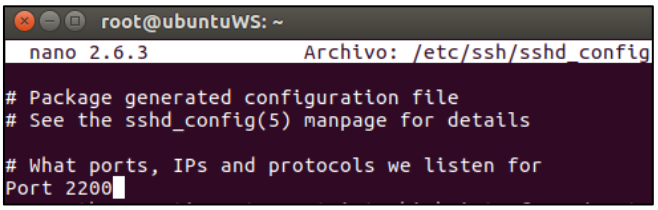


Figura 246. Cambiar el puerto 22 que viene por defecto
Fuente: Autor

Luego se habilita el script UFW que viene instalado por defecto en Ubuntu server, este script permitirá administrar el cortafuegos del servidor, con el cual se deja abierto únicamente los puertos correspondientes al web service y a la conexión remota ssh, lo demás queda bloqueado.

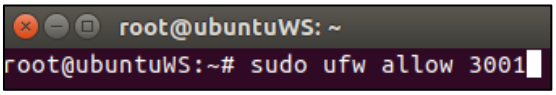


Figura 247. Agregar los puertos del web service (3001)
Fuente: Autor

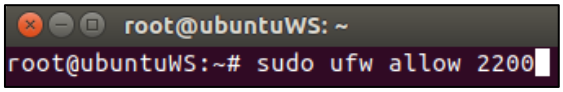


Figura 248. Agregar los puertos del ssh (2200)
Fuente: Autor

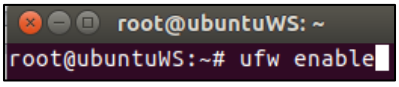


Figura 249. Iniciar UFW
Fuente: Autor

```
root@ubuntuWS:~# ufw status numbered
Estado: activo

     Hasta                      Acción      Desde
     -----                      ------      -----
[ 1] 2200                       ALLOW IN    Anywhere
[ 2] 3001                       ALLOW IN    Anywhere
```

Figura 250. Lista de puertos habilitados por UFW
Fuente: Autor

6.2 Implementación de la app android

6.2.1 Incluir el subdominio del web service en la app

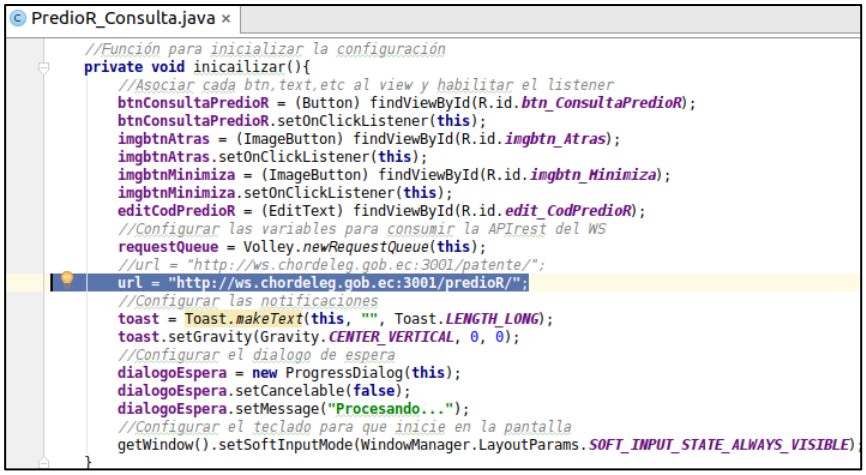
Luego de configurar un subdominio para el web service, es necesario editar el código fuente y generar un nuevo ejecutable de la app android desarrollada, para que realice las peticiones al dominio http://ws.chordeleg.gob.ec:3001 en vez de la dirección ip pública http://181.211.253.98:3001; para ello se realizan los siguientes pasos.

- El código fuente se tiene que editar únicamente la parte donde se define la variable de la url a la cual se va a conectar, esto se realizada para cada tributo municipal, en las siguientes figuras se puede observar el código editado para cada caso:

PredioR_Consulta.java ×

```
//Función para inicializar la configuración
private void inicailizar(){
    //Asociar cada btn,text,etc al view y habilitar el listener
    btnConsultaPredioR = (Button) findViewById(R.id.btn_ConsultaPredioR);
    btnConsultaPredioR.setOnClickListener(this);
    imgbtnAtras = (ImageButton) findViewById(R.id.imgbtn_Atras);
    imgbtnAtras.setOnClickListener(this);
    imgbtnMinimiza = (ImageButton) findViewById(R.id.imgbtn_Minimiza);
    imgbtnMinimiza.setOnClickListener(this);
    editCodPredioR = (EditText) findViewById(R.id.edit_CodPredioR);
    //Configurar las variables para consumir la APIrest del WS
    requestQueue = Volley.newRequestQueue(this);
    //url = "http://ws.chordeleg.gob.ec:3001/patente/";
    url = "http://ws.chordeleg.gob.ec:3001/predioR/";
    //Configurar las notificaciones
    toast = Toast.makeText(this, "", Toast.LENGTH_LONG);
    toast.setGravity(Gravity.CENTER_VERTICAL, 0, 0);
    //Configurar el dialogo de espera
    dialogoEspera = new ProgressDialog(this);
    dialogoEspera.setCancelable(false);
    dialogoEspera.setMessage("Procesando...");
    //Configurar el teclado para que inicie en la pantalla
    getWindow().setSoftInputMode(WindowManager.LayoutParams.SOFT_INPUT_STATE_ALWAYS_VISIBLE);
}
```

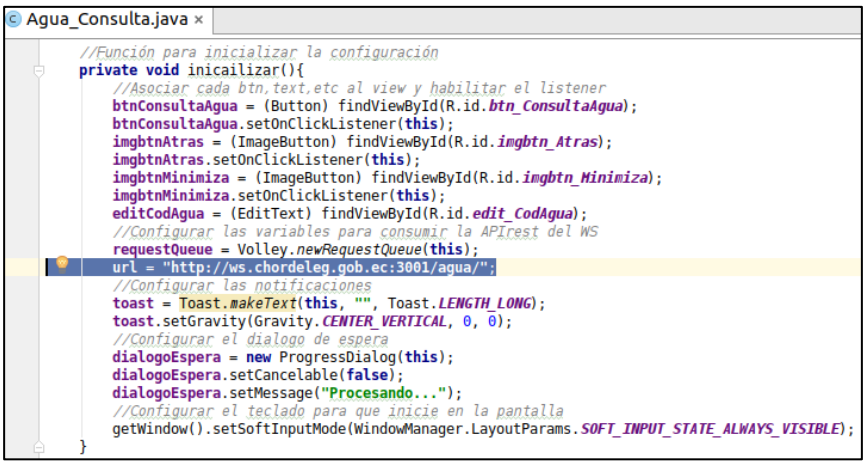
Figura 251. Url para el predio urbano con el dominio http://ws.chordeleg.gob.ec
Fuente: Autor



```
PredioR_Consulta.java ×

    //Función para inicializar la configuración
    private void inicailizar(){
        //Asociar cada btn,text,etc al view y habilitar el listener
        btnConsultaPredioR = (Button) findViewById(R.id.btn_ConsultaPredioR);
        btnConsultaPredioR.setOnClickListener(this);
        imgbtnAtras = (ImageButton) findViewById(R.id.imgbtn_Atras);
        imgbtnAtras.setOnClickListener(this);
        imgbtnMinimiza = (ImageButton) findViewById(R.id.imgbtn_Minimiza);
        imgbtnMinimiza.setOnClickListener(this);
        editCodPredioR = (EditText) findViewById(R.id.edit_CodPredioR);
        //Configurar las variables para consumir la APIrest del WS
        requestQueue = Volley.newRequestQueue(this);
        //url = "http://ws.chordeleg.gob.ec:3001/patente/";
        url = "http://ws.chordeleg.gob.ec:3001/predioR/";
        //Configurar las notificaciones
        toast = Toast.makeText(this, "", Toast.LENGTH_LONG);
        toast.setGravity(Gravity.CENTER_VERTICAL, 0, 0);
        //Configurar el dialogo de espera
        dialogoEspera = new ProgressDialog(this);
        dialogoEspera.setCancelable(false);
        dialogoEspera.setMessage("Procesando...");
        //Configurar el teclado para que inicie en la pantalla
        getWindow().setSoftInputMode(WindowManager.LayoutParams.SOFT_INPUT_STATE_ALWAYS_VISIBLE);
    }
```

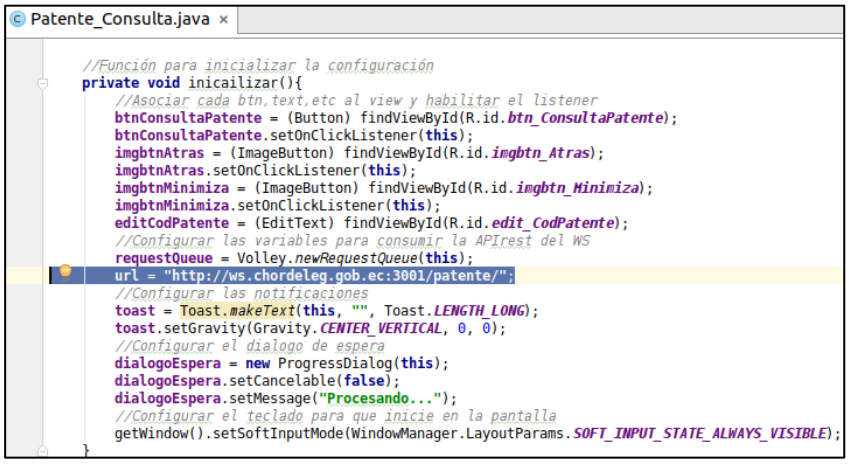
Figura 252. Url para el predio rural con el dominio http://ws.chordeleg.gob.ec
Fuente: Autor



```
Agua_Consulta.java ×

    //Función para inicializar la configuración
    private void inicailizar(){
        //Asociar cada btn,text,etc al view y habilitar el listener
        btnConsultaAgua = (Button) findViewById(R.id.btn_ConsultaAgua);
        btnConsultaAgua.setOnClickListener(this);
        imgbtnAtras = (ImageButton) findViewById(R.id.imgbtn_Atras);
        imgbtnAtras.setOnClickListener(this);
        imgbtnMinimiza = (ImageButton) findViewById(R.id.imgbtn_Minimiza);
        imgbtnMinimiza.setOnClickListener(this);
        editCodAgua = (EditText) findViewById(R.id.edit_CodAgua);
        //Configurar las variables para consumir la APIrest del WS
        requestQueue = Volley.newRequestQueue(this);
        url = "http://ws.chordeleg.gob.ec:3001/agua/";
        //Configurar las notificaciones
        toast = Toast.makeText(this, "", Toast.LENGTH_LONG);
        toast.setGravity(Gravity.CENTER_VERTICAL, 0, 0);
        //Configurar el dialogo de espera
        dialogoEspera = new ProgressDialog(this);
        dialogoEspera.setCancelable(false);
        dialogoEspera.setMessage("Procesando...");
        //Configurar el teclado para que inicie en la pantalla
        getWindow().setSoftInputMode(WindowManager.LayoutParams.SOFT_INPUT_STATE_ALWAYS_VISIBLE);
    }
```

Figura 253. Url para el agua potable con el dominio http://ws.chordeleg.gob.ec
Fuente: Autor



```
Patente_Consulta.java ×

    //Función para inicializar la configuración
    private void inicailizar(){
        //Asociar cada btn,text,etc al view y habilitar el listener
        btnConsultaPatente = (Button) findViewById(R.id.btn_ConsultaPatente);
        btnConsultaPatente.setOnClickListener(this);
        imgbtnAtras = (ImageButton) findViewById(R.id.imgbtn_Atras);
        imgbtnAtras.setOnClickListener(this);
        imgbtnMinimiza = (ImageButton) findViewById(R.id.imgbtn_Minimiza);
        imgbtnMinimiza.setOnClickListener(this);
        editCodPatente = (EditText) findViewById(R.id.edit_CodPatente);
        //Configurar las variables para consumir la APIrest del WS
        requestQueue = Volley.newRequestQueue(this);
        url = "http://ws.chordeleg.gob.ec:3001/patente/";
        //Configurar las notificaciones
        toast = Toast.makeText(this, "", Toast.LENGTH_LONG);
        toast.setGravity(Gravity.CENTER_VERTICAL, 0, 0);
        //Configurar el dialogo de espera
        dialogoEspera = new ProgressDialog(this);
        dialogoEspera.setCancelable(false);
        dialogoEspera.setMessage("Procesando...");
        //Configurar el teclado para que inicie en la pantalla
        getWindow().setSoftInputMode(WindowManager.LayoutParams.SOFT_INPUT_STATE_ALWAYS_VISIBLE);
    }
```

Figura 254. Url para la patente con el dominio http://ws.chordeleg.gob.ec
Fuente: Autor

- Finalmente se vuelve a generar el ejecutable, siguiendo los mismos pasos de la sección 4.6.7.

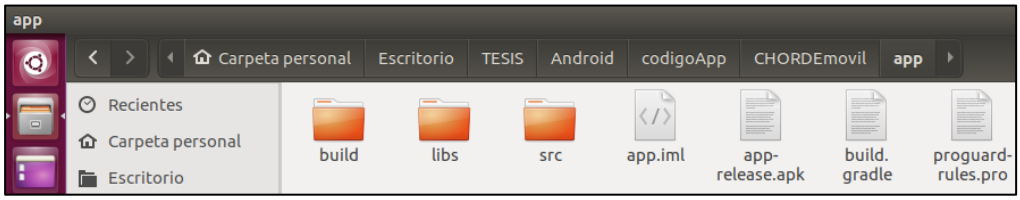


Figura 255. Carpeta donde se remplaza el ejecutable de la aplicación android
Fuente: Autor

6.2.2 Subir la app a google play

Para subir una app a Google Play es necesario disponer de una cuenta para desarrolladores, la cual se adquirió realizando los siguientes pasos:

- Antes de iniciar se necesita disponer de una cuenta en gmail, luego se ingresa a la página de Google Play Developer Console, se loguea con la cuenta de gmail y se abre el asistente para adquirir la cuenta de desarrollador; el costo de una cuenta para desarrolladores es de $25.00 por única vez, tomando en cuenta su costo se acepta la licencia y se da un clic en “Continuar para completar el pago”.

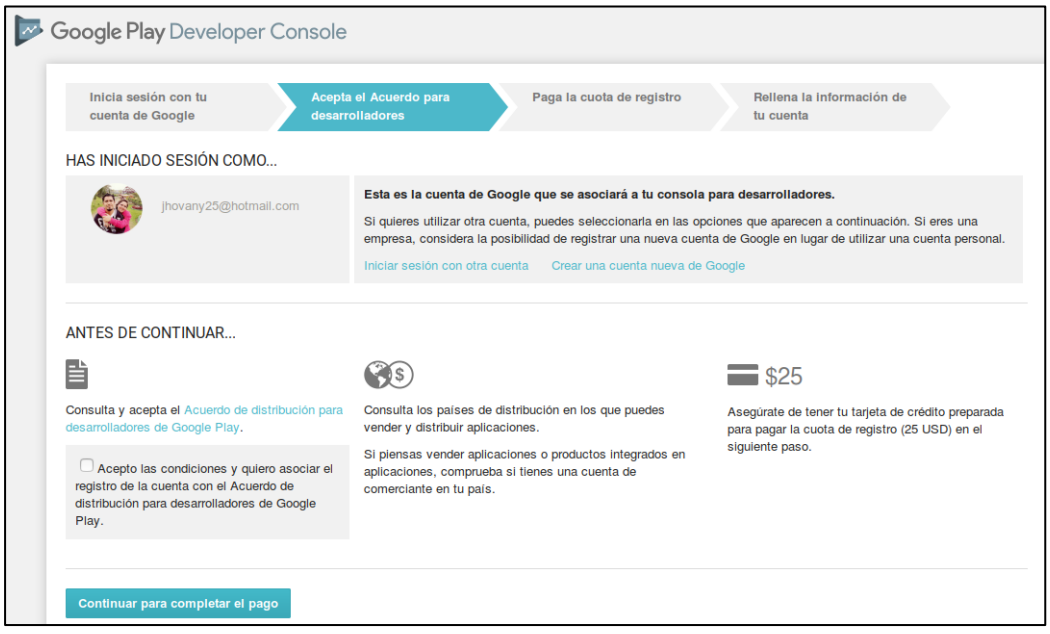


Figura 256. Asistente para crear una cuenta Google Play Developer
Fuente: Autor

- Seguidamente, ingresar los datos de la tarjeta de crédito para realizar el pago, la dirección de facturación y dar un clic en “PAGAR”.

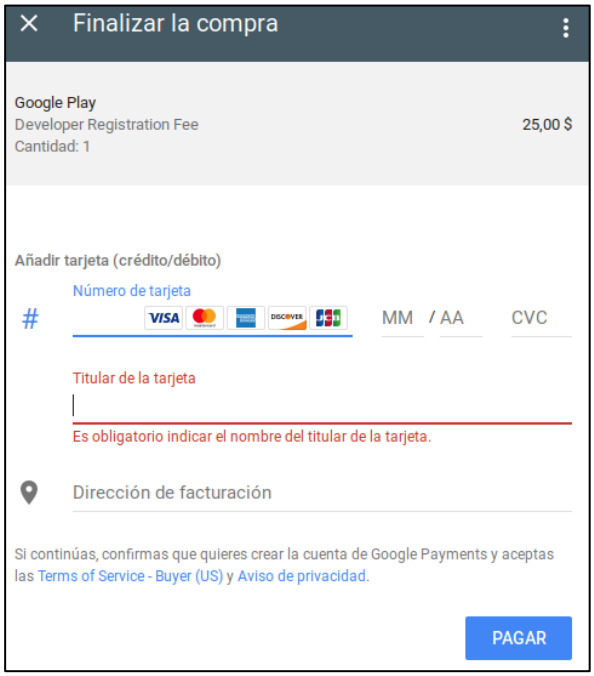


Figura 257. Pago de la cuenta Google Play Developer
Fuente: Autor

- Por último se llena los datos de perfil de la cuenta de desarrollador, los cuales pueden ser editados más adelante, y se da un clic “Completar registro”, con eso ya se tiene creada la cuenta de desarrollador.

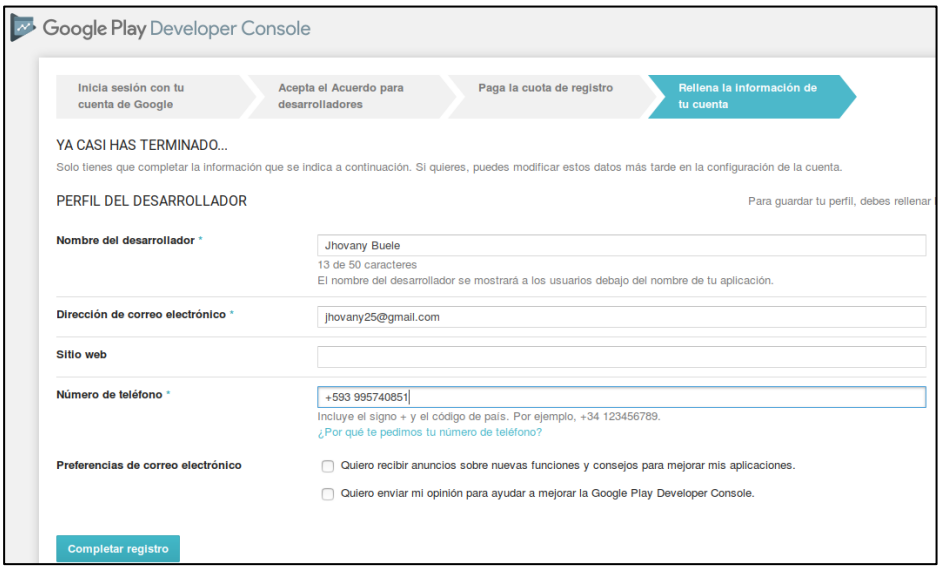


Figura 258. Pago de la cuenta Google Play Developer
Fuente: Autor

Luego de adquirir y crear la cuenta de desarrollador de Google Play, se procede a subir la aplicación desarrollada para consultar los principales tributos del GAD Municipal de Chordeleg, para lo cual se realizaron los siguientes pasos:

- Ingresar en la cuenta de desarrollador de Google Play y dar un clic en “Publicar una aplicación de Android en Google Play”.

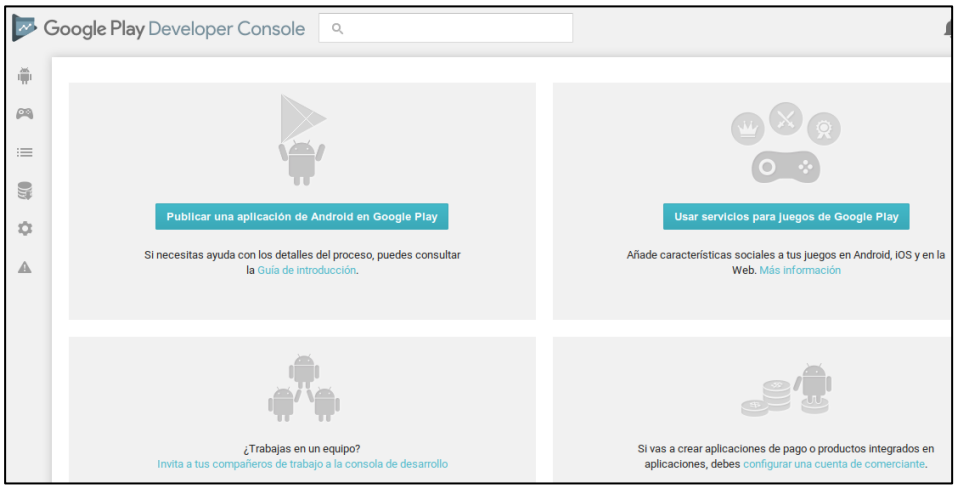


Figura 259. Asistente para publicar una app en Google Play
Fuente: Autor

- En la siguiente ventana seleccionar el idioma predeterminado de la aplicación, ingresar el nombre de la aplicación y dar un clic en “Crear”.

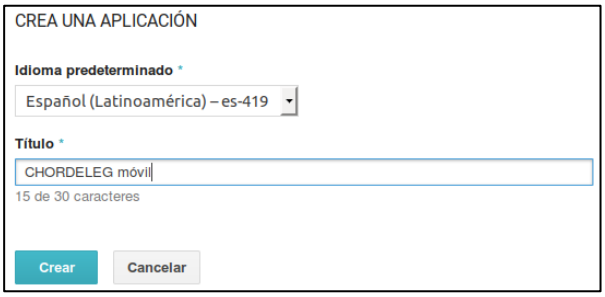


Figura 260. Ingresar el nombre y el idioma de la app de Google Play
Fuente: Autor

- A continuación se llena todos los datos del formulario, y la aplicación queda lista para ser publicada.

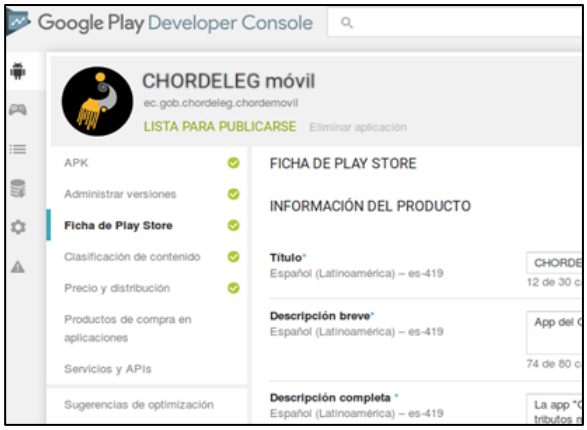


Figura 261. Llenar los datos de la app previa a su publicación en Google Play
Fuente: Autor

- Para iniciar la publicación en Google Play, dirigirse a “Administrar versiones”, y dar un clic en “Iniciar Lanzamiento”.

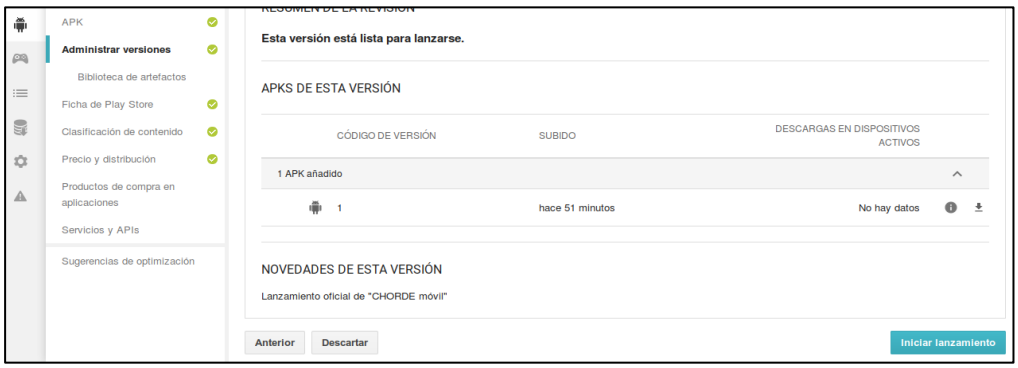


Figura 262. Iniciar el lanzamiento de la app en Google Play
Fuente: Autor

- Se regresa al menú principal de la cuenta de desarrolladores y se observa la app pendiente de publicación.

Figura 263. App pendiente de publicación en Google Play
Fuente: Autor

- Finalmente, luego de una espera de 3 horas aproximadamente, la app fue validada por Google Play y se publicó exitosamente.

Figura 264. App publicada exitosamente en Google Play
Fuente: Autor

6.2.3 Instalar la app en un dispositivo móvil

La instalación de la aplicación se realizó sobre un dispositivo móvil Samsung Note 5, para ello se realizaron los siguientes pasos:

- Dirigirse al Play Store de Google, digitar “CHORDELEG móvil” y presionar en “Buscar”.

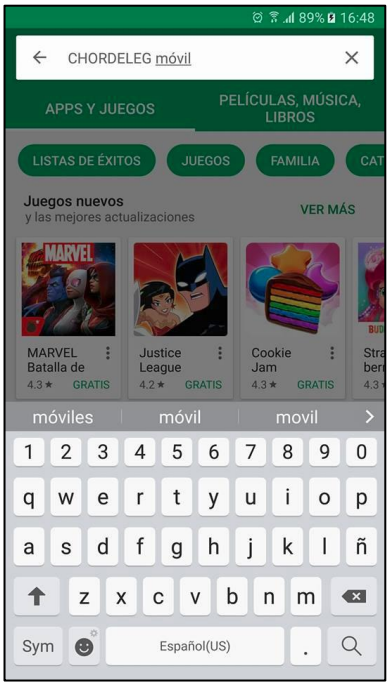


Figura 265. Buscando la App en Play Store
Fuente: Autor

- Se presiona en el nombre de la aplicación buscada, se abre la descripción de la app y seguidamente se debe presionar en “INSTALAR”.

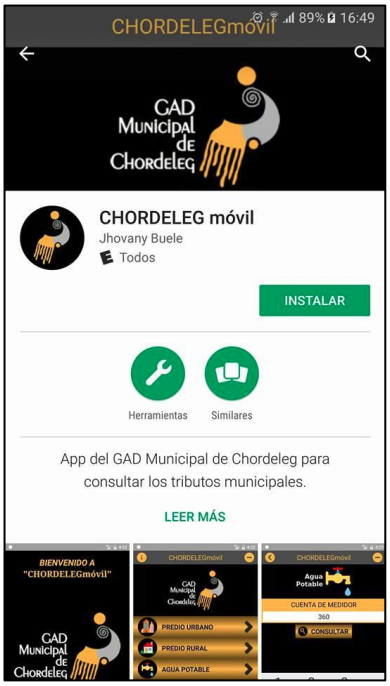


Figura 266. Descripción de la app “CHORDELEG móvil”
Fuente: Autor

- Inicia la descarga, se instala, y para ejecutarla se presiona en “ABRIR”.

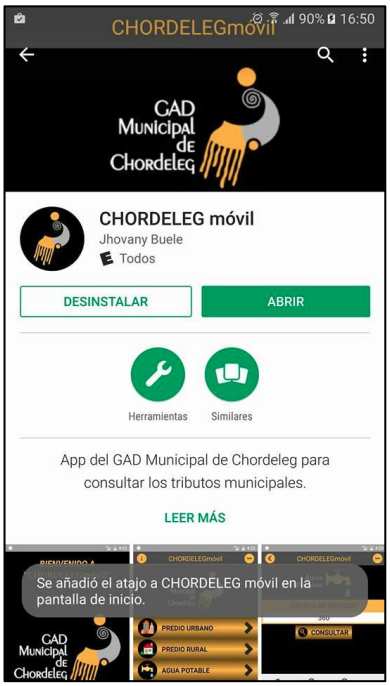


Figura 267. App “CHORDELEG móvil” instalada en el Samsung Note 5
Fuente: Autor

- Por último se puede ver el ícono creado para ingresar a la app en cualquier momento desde la sección de aplicaciones del dispositivo móvil.

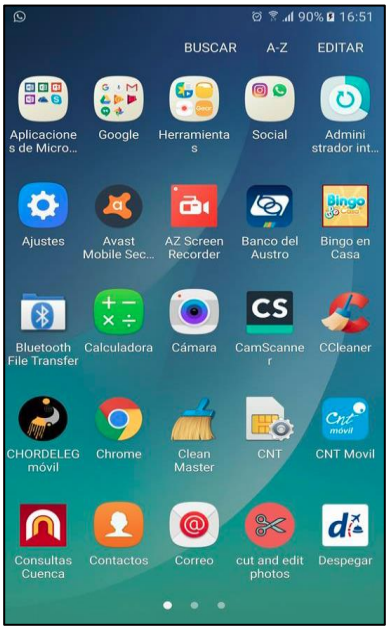


Figura 268. Ícono creado de la app "CHORDELEG móvil" en el dispositivo móvil
Fuente: Autor

6.3 Implementación de calidad de servicio (QoS)

6.3.1 Marcación de conexiones y paquetes

Recordando la sección 5.4.3 específicamente la tabla 53, se agrupó el tráfico de la red acorde a las necesidades del GAD Municipal de Chordeleg, es así que manteniendo el mismo esquema de la tabla en mención, a continuación se marcará las conexiones y paquetes de cada grupo, dentro del router principal haciendo uso del software WinBox:

6.3.1.1 Grupo de administración de red y dns

Siguiendo el método mencionado en la sección 5.4, se realizaron los siguientes pasos:

- Marcar la conexión para el protocolo ICMP y la nombrarlo "PRIO 1 - conn".

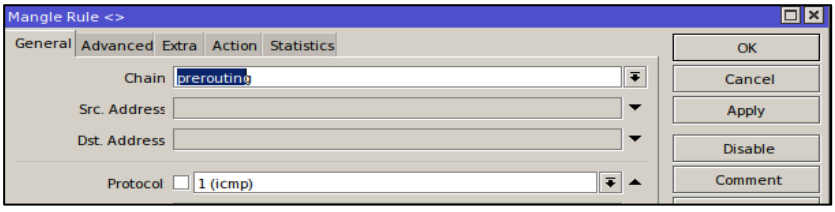


Figura 269. Configuración de la conexión ICMP
Fuente: Autor

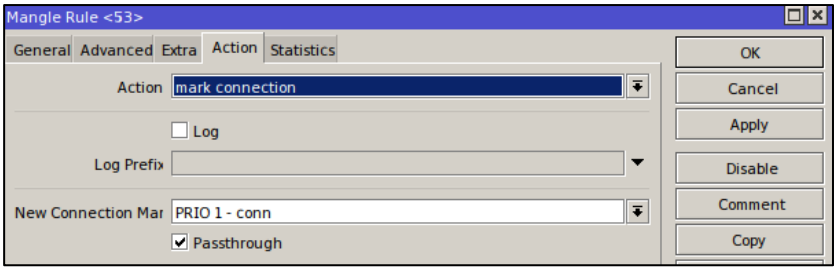


Figura 270. Marcación de la conexión ICMP
Fuente: Autor

- Marcar la conexión para SSH (puerto 21) y WinBox (puerto 8291) en el protocolo TCP; así mismo nombrarlo “PRIO 1 - conn” con el fin de que pertenezca al mismo grupo.

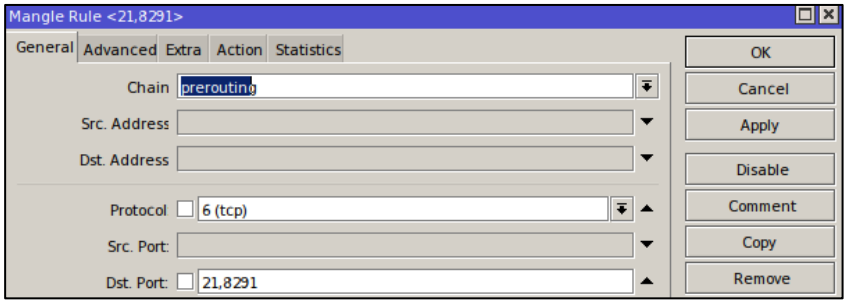


Figura 271. Configuración de las conexiones SSH y WinBox
Fuente: Autor

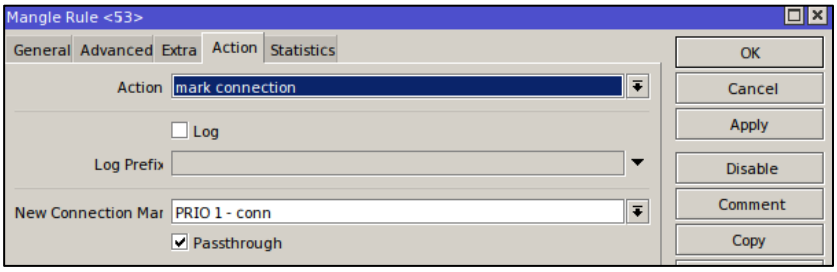


Figura 272. Marcación de las conexiones SSH y WinBox
Fuente: Autor

- Marcar la conexión DNS (puerto 53) en el protocolo UDP y nombrarlo “PRIO 1 - conn” con el fin de que pertenezca al mismo grupo.

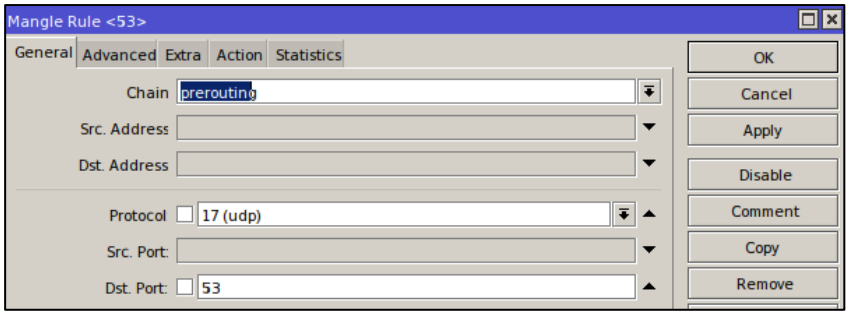


Figura 273. Configuración de la conexión DNS
Fuente: Autor

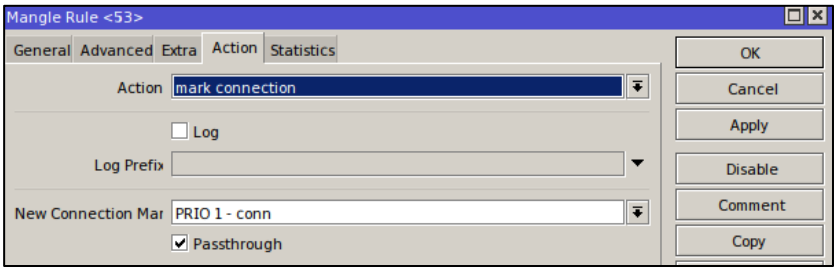


Figura 274. Marcación de la conexión DNS
Fuente: Autor

- Por último se realiza la marcación de paquetes de las conexiones agrupadas como "PRIO 1 - conn", a la cual se nombra "PRIO 1".

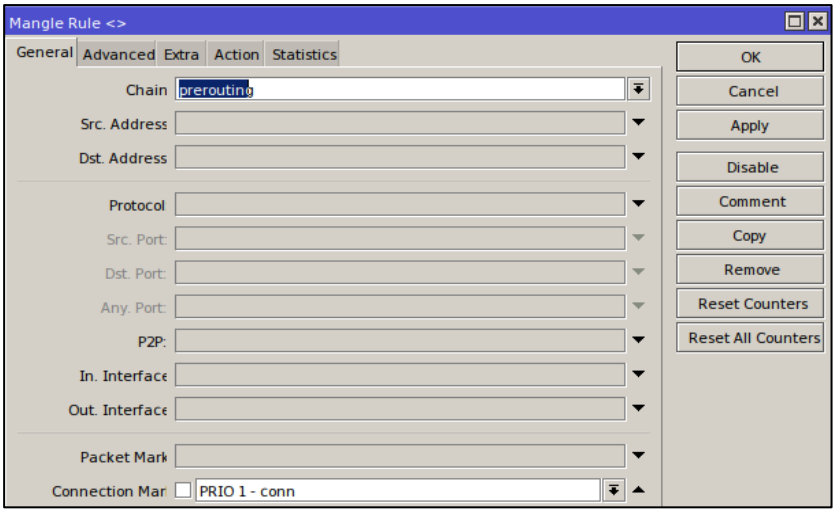


Figura 275. Configuración del grupo de conexiones "PRIO 1 - conn"
Fuente: Autor

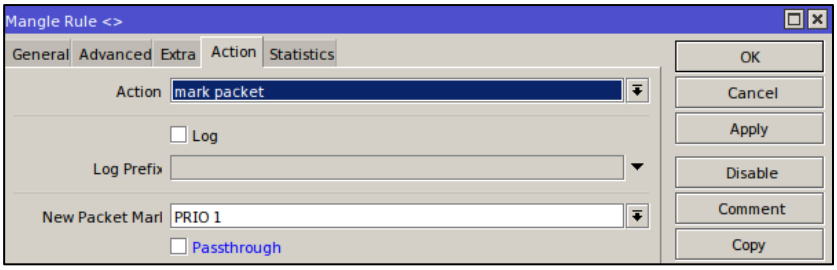


Figura 276. Marcación de paquetes del grupo de conexiones "PRIO 1 - conn"
Fuente: Autor

6.3.1.2 Grupo de base de datos

Siguiendo el método anterior, se realizaron los siguientes pasos:

- Marcar la conexión para MySQL (puerto 3306), PostgreSQL (puerto 5432) y SQL server (puerto 1433) en el protocolo TCP; y nombrarla "PRIO 2 - conn".

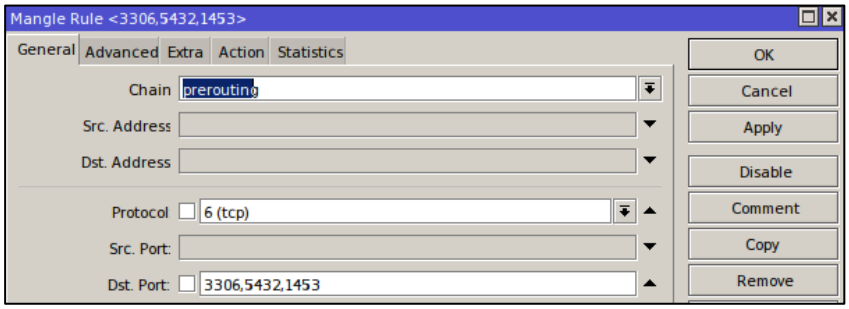


Figura 277. Configuración de las conexiones MySQL, PostgreSQL y SQL server
Fuente: Autor

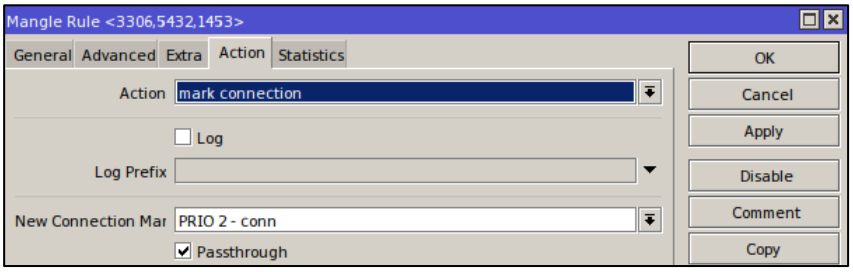


Figura 278. Marcación de las conexiones MySQL, PostgreSQL y SQL server
Fuente: Autor

- Por último se realiza la marcación de paquetes de las conexiones agrupadas como "PRIO 2 - conn", a la cual se nombra "PRIO 2".

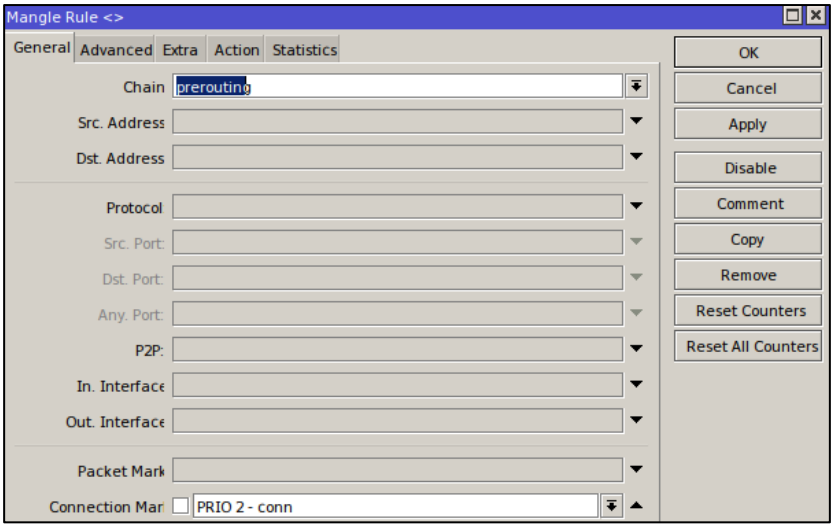


Figura 279. Configuración del grupo de conexiones "PRIO 2 - conn"
Fuente: Autor

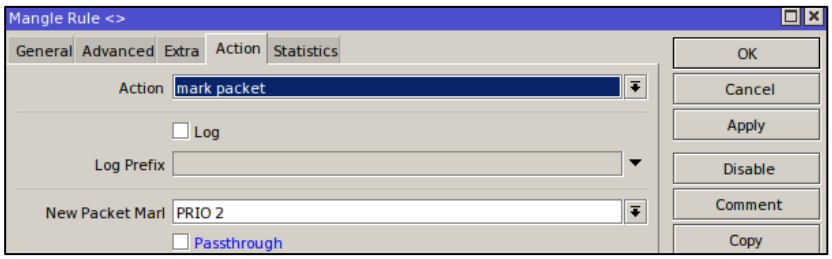


Figura 280. Marcación de paquetes del grupo de conexiones "PRIO 2 - conn"
Fuente: Autor

6.3.1.3 Grupo de web service

De igual forma siguiendo el método anterior, se realizaron los siguientes pasos:

- Marcar la conexión para Node.Js (puerto 3001) y nombrarla como "PRIO 3 - conn".

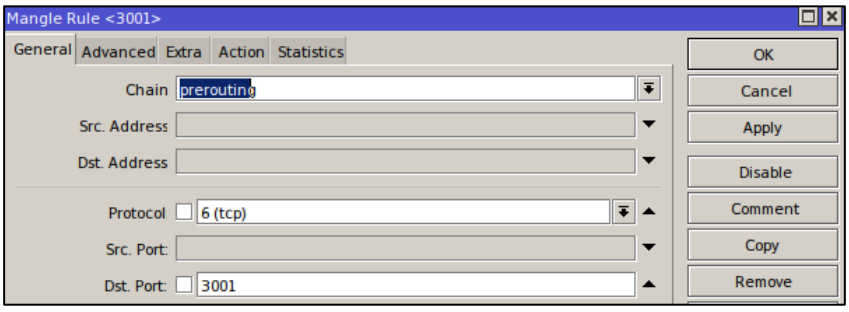


Figura 281. Configuración de la conexión Node.Js
Fuente: Autor

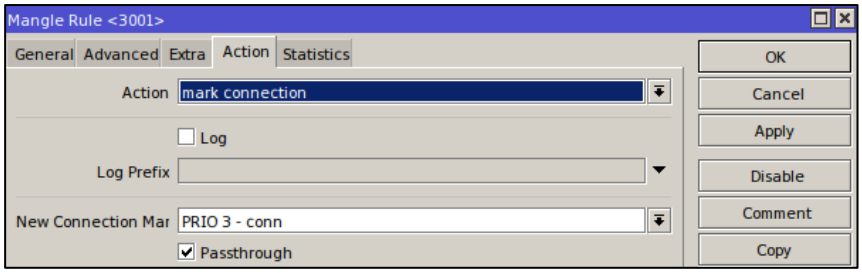


Figura 282. Marcación de la conexión Node.Js
Fuente: Autor

- Por último se realiza la marcación de paquetes de las conexiones agrupadas como "PRIO 3 - conn", a la cual se nombra "PRIO 3".

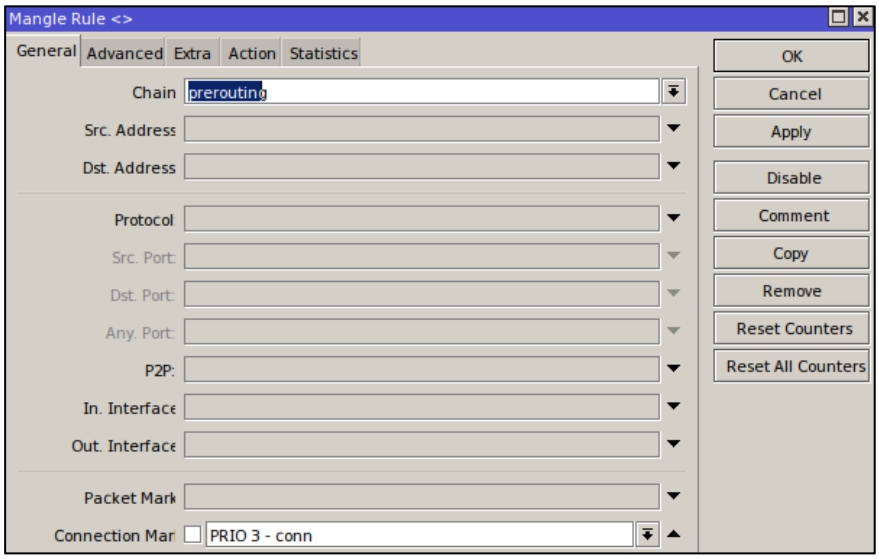


Figura 283. Configuración del grupo de conexiones "PRIO 3 - conn"
Fuente: Autor

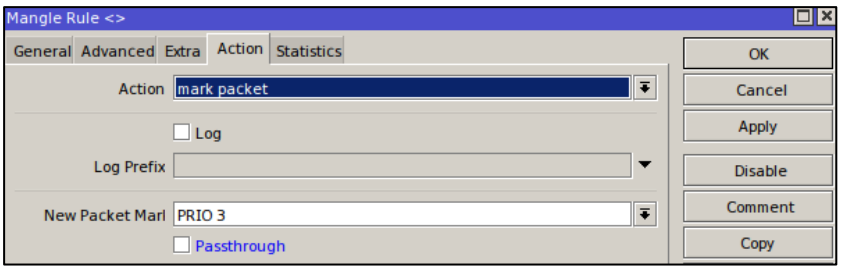


Figura 284. Marcación de paquetes del grupo de conexiones "PRIO 3 - conn"
Fuente: Autor

6.3.1.4 Grupo de navegación web

Continuando con el método anterior, se realizaron los siguientes pasos:

- Marcar las conexiones para HTTP (puerto 80) y HTTPs (puerto 443), a la cual se nombra como "PRIO 4 - conn".

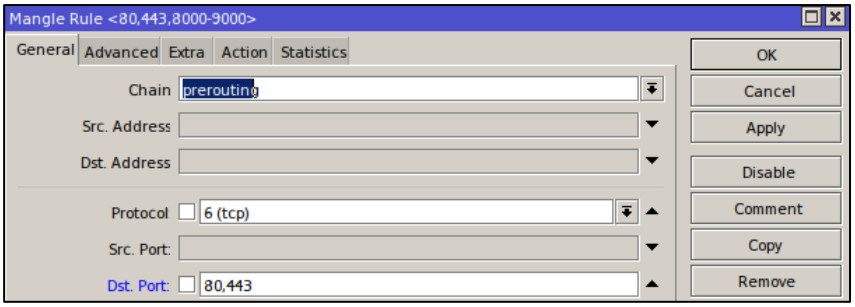


Figura 285. Configuración de las conexiones HTTP y HTTPs
Fuente: Autor

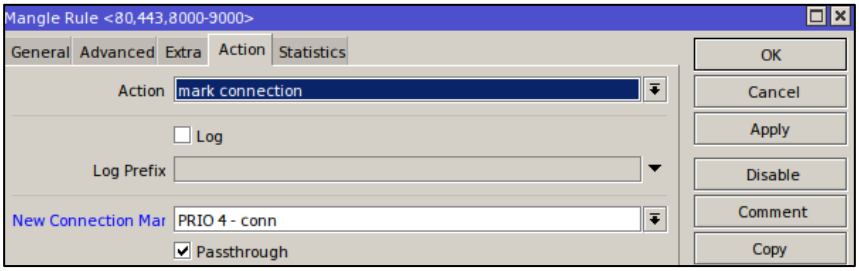


Figura 286. Marcación de las conexiones HTTP y HTTPs
Fuente: Autor

- Por último realizar la marcación de paquetes de las conexiones agrupadas como "PRIO 4 - conn", a la cual se nombra "PRIO 4".

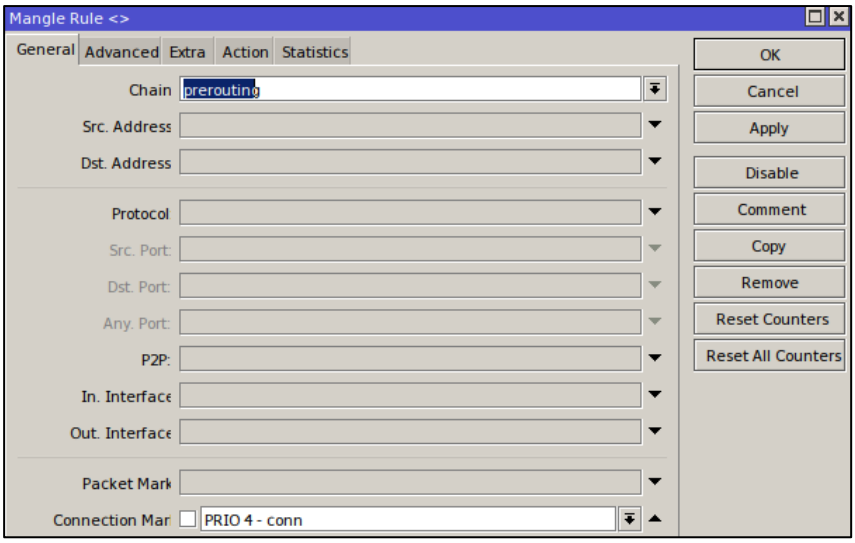


Figura 287. Configuración del grupo de conexiones "PRIO 4 - conn"
Fuente: Autor

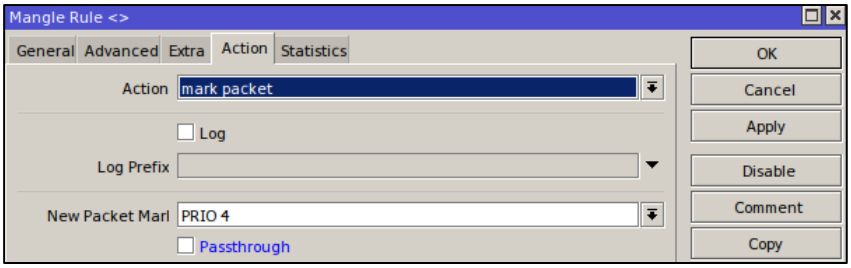


Figura 288. Marcación de paquetes del grupo de conexiones "PRIO 4 - conn"
Fuente: Autor

6.3.1.5 Grupo de correo

Continuando con el método anterior, se realizaron los siguientes pasos:

- Marcar las conexiones para IMAP, POP y SMTP, a la cual se nombra como "PRIO 5 - conn".

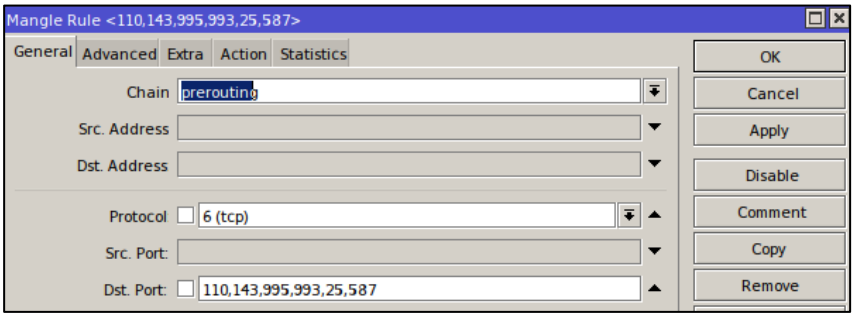


Figura 289. Configuración de las conexiones IMAP, POP y SMTP
Fuente: Autor

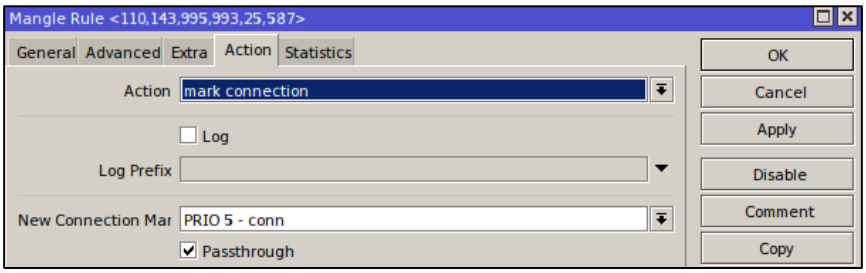


Figura 290. Marcación de las conexiones IMAP, POP y SMTP
Fuente: Autor

- Por último se realiza la marcación de paquetes de las conexiones agrupadas como "PRIO 5 - conn", a la cual se nombra "PRIO 5".

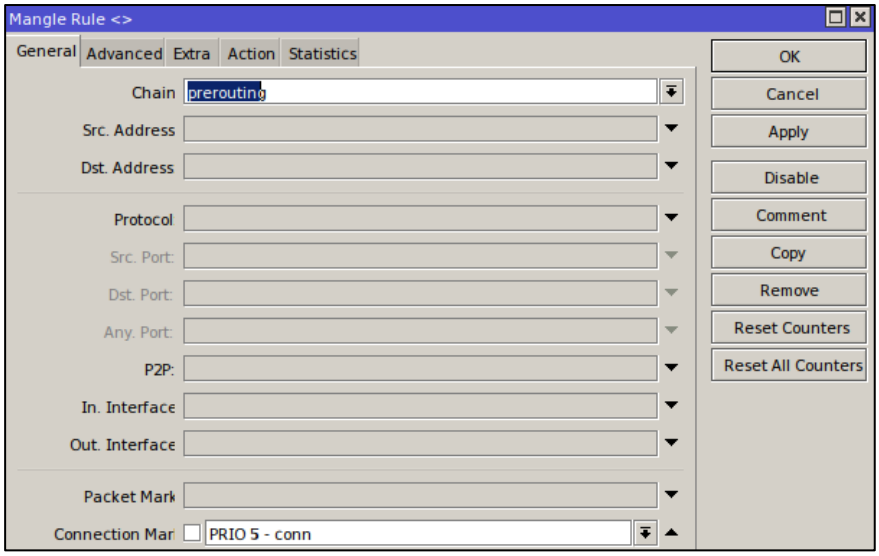


Figura 291. Configuración del grupo de conexiones "PRIO 5 - conn"
Fuente: Autor

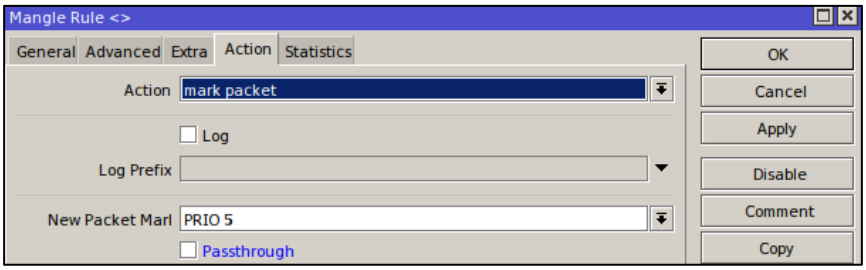


Figura 292. Marcación de paquetes del grupo de conexiones "PRIO 5 - conn"
Fuente: Autor

6.3.1.6 Grupo de otros

Siguiendo el método anterior, se realizaron los siguientes pasos:

- Marcar las conexiones restantes que son menos importantes que las anteriores, a la cual se nombra "PRIO 6 - conn".

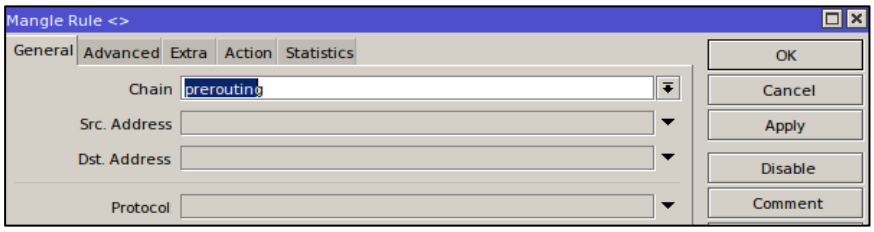


Figura 293. Configuración de las conexiones OTROS
Fuente: Autor

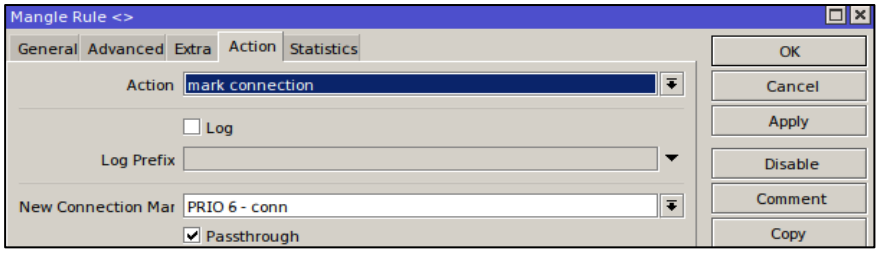


Figura 294. Marcación de las conexiones OTROS
Fuente: Autor

- Por último realizar la marcación de paquetes de las conexiones agrupadas como "PRIO 6 - conn", a la cual se nombra "PRIO 6".

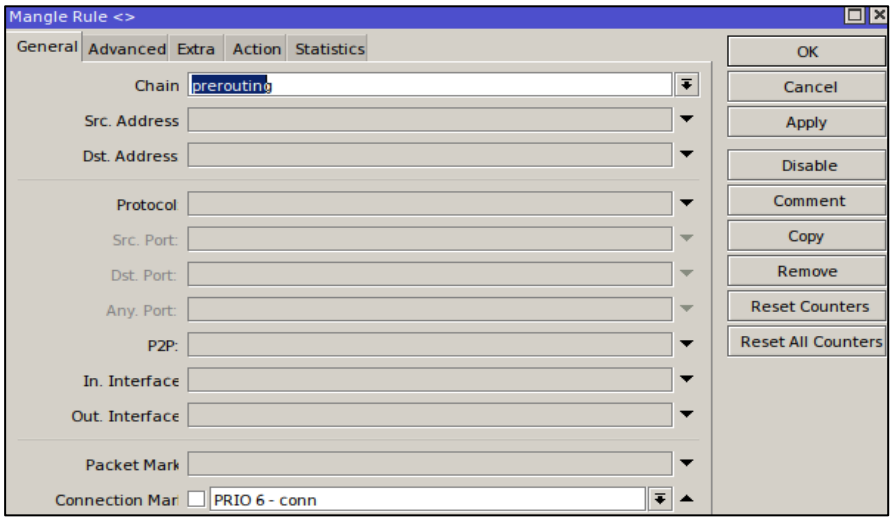


Figura 295. Configuración del grupo de conexiones "PRIO 6 - conn"
Fuente: Autor

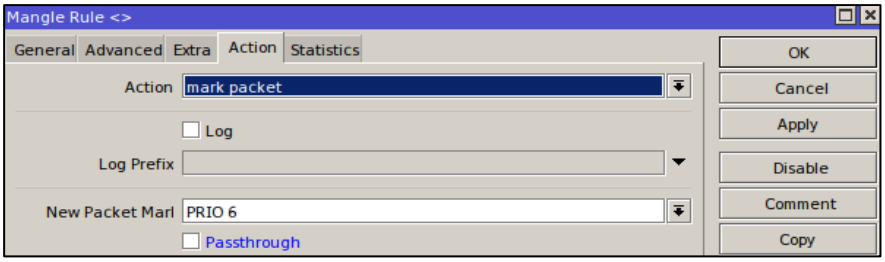


Figura 296. Marcación de paquetes del grupo de conexiones "PRIO 6 - conn"
Fuente: Autor

6.3.1.7 Grupo de descargas mayores a 50 megas

Con el mismo método anterior, se realizaron los siguientes pasos:

- Marcar todas las conexiones del protocolo TCP y filtrar las que tienen un tamaño mayor a 50Megas; luego se nombra como "PRIO 7 - conn".

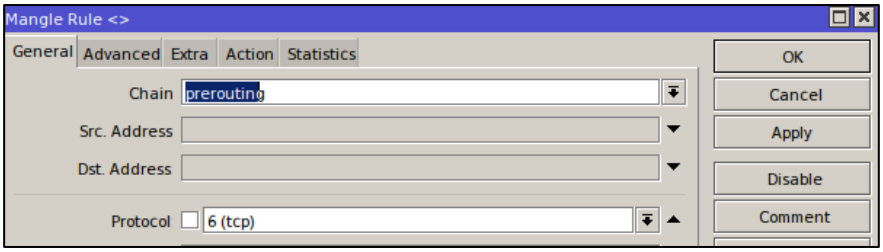


Figura 297. Configuración de todas las conexiones TCP
Fuente: Autor

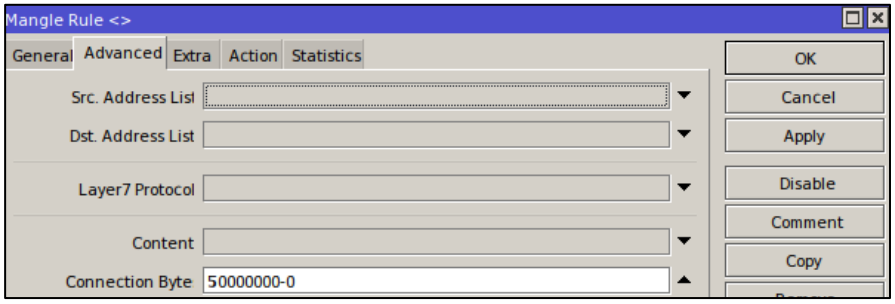


Figura 298. Configuración de las conexiones mayores a 50Megas
Fuente: Autor

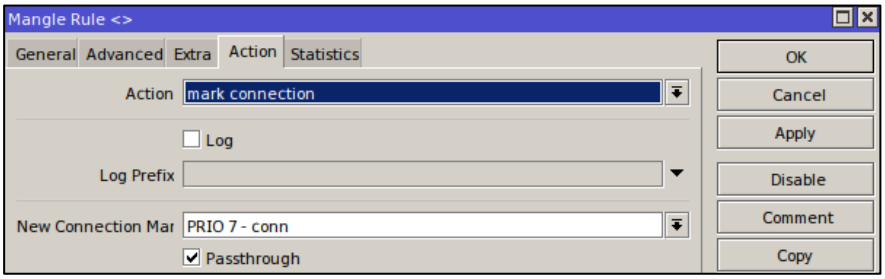


Figura 299. Marcación de las conexiones mayores a 50Megas
Fuente: Autor

- Por último realizar la marcación de paquetes de las conexiones agrupadas como "PRIO 7 - conn", a la cual se llama como "PRIO 7".

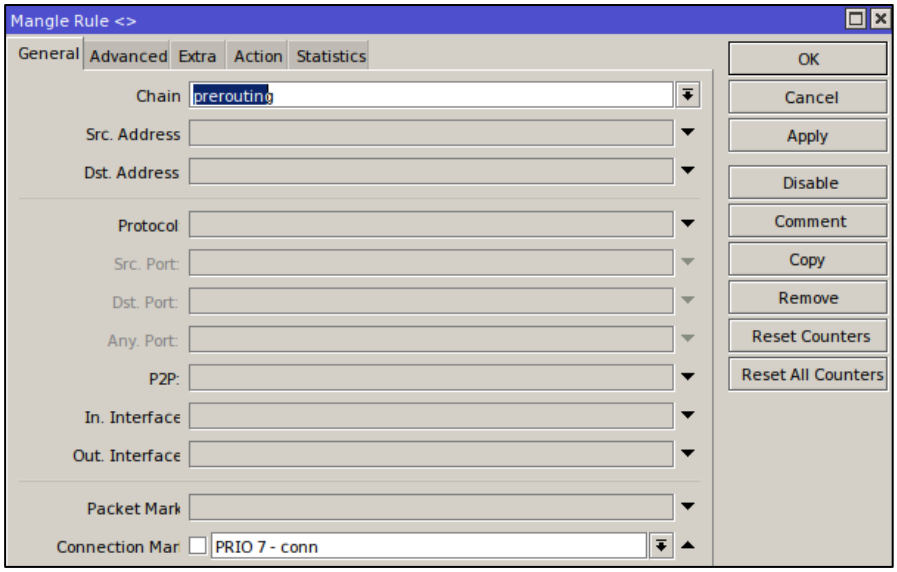


Figura 300. Configuración del grupo de conexiones "PRIO 7 - conn"
Fuente: Autor

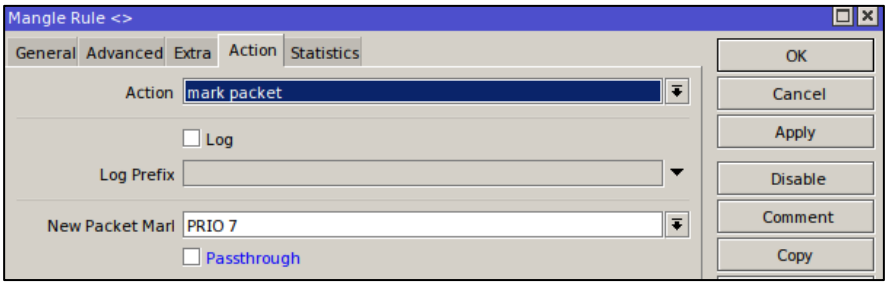


Figura 301. Marcación de paquetes del grupo de conexiones "PRIO 7 - conn"
Fuente: Autor

6.3.1.8 Grupo de programas p2p

Continuar con el último grupo, para ello se realizaron los siguientes pasos:

- Marcar todas las conexiones de P2P y nombrarla "PRIO 8 - conn".

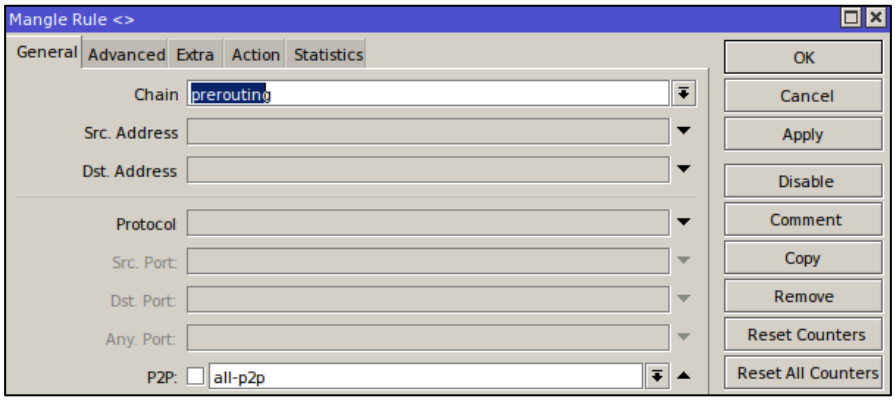


Figura 302. Configuración de todas las conexiones P2P
Fuente: Autor

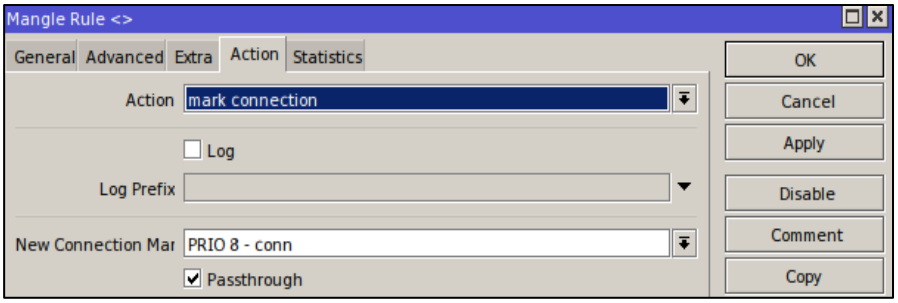


Figura 303. Marcación de todas las conexiones P2P
Fuente: Autor

- Por último realizar la marcación de paquetes de las conexiones agrupadas como "PRIO 8 - conn", a la cual se nombra como "PRIO 8".

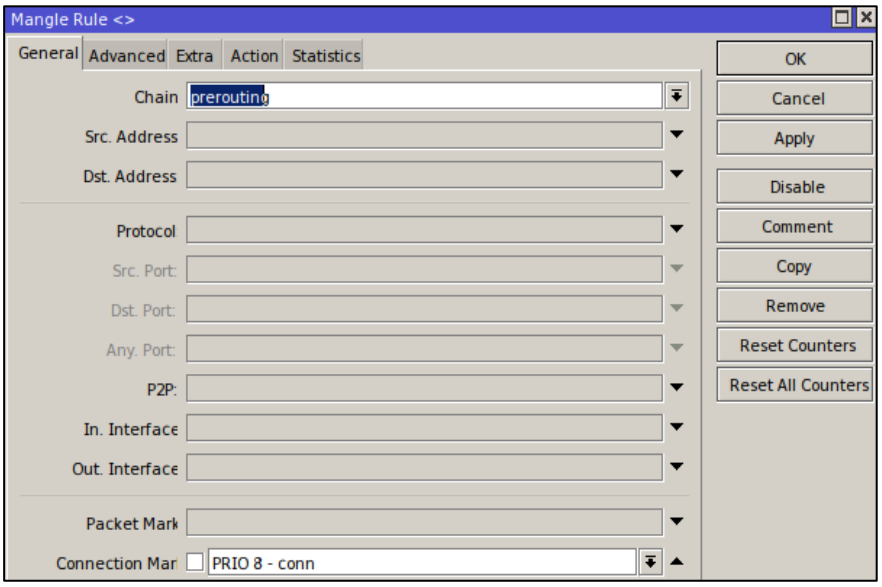


Figura 304. Configuración del grupo de conexiones "PRIO 8 - conn"
Fuente: Autor

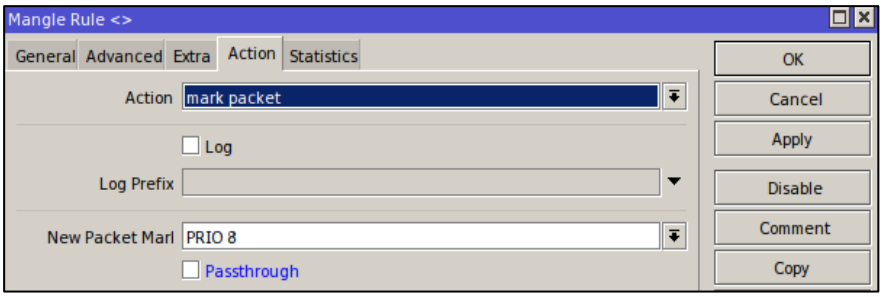
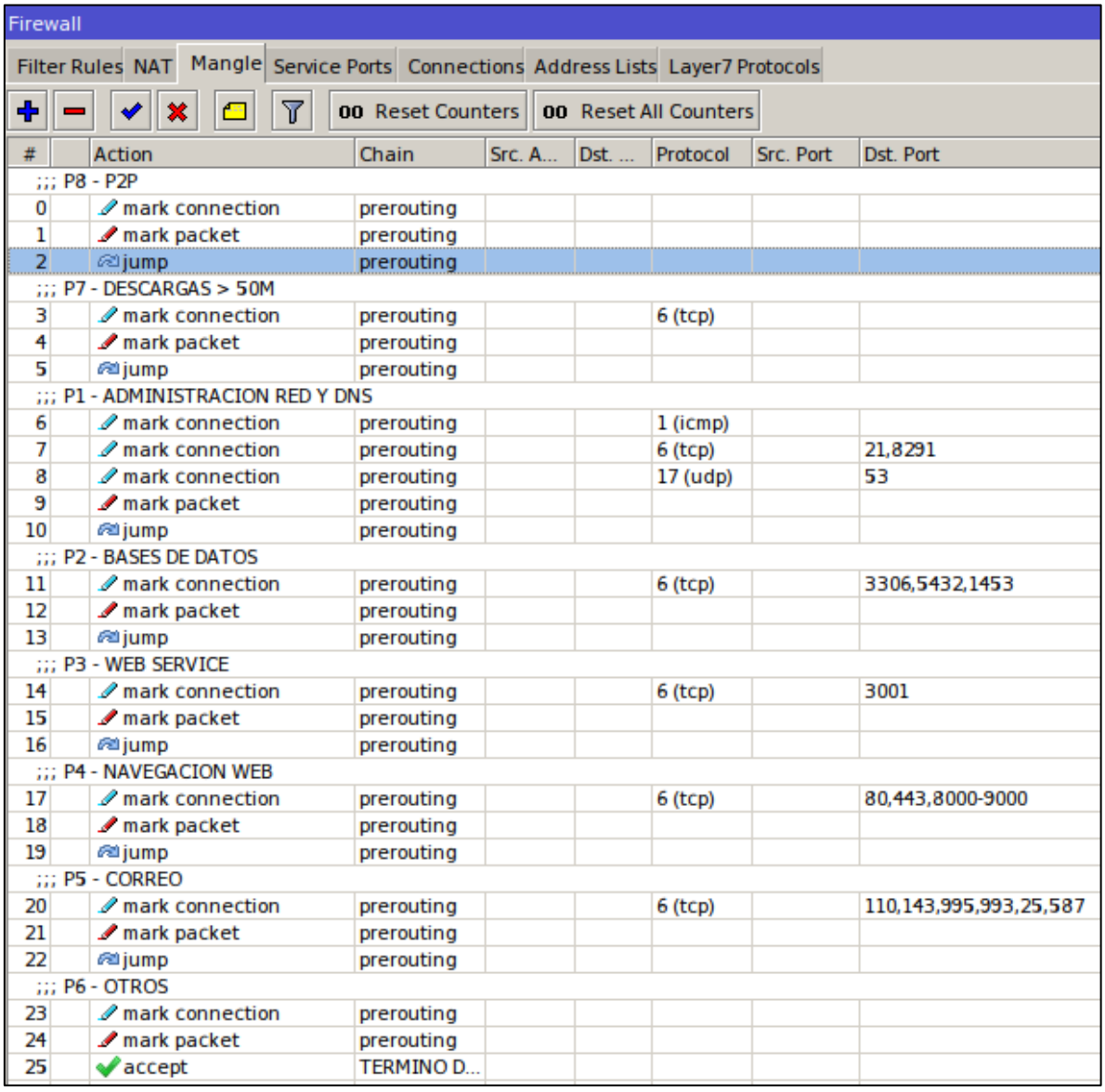


Figura 305. Marcación de paquetes del grupo de conexiones "PRIO 8 - conn"
Fuente: Autor

6.3.1.9 Resumen de marcación de conexiones y paquetes

Finalmente en la siguiente figura se puede observar, un resumen de todas las reglas que se crearon anteriormente, tanto para la marcación de conexiones como para la marcación de paquetes:



#		Action	Chain	Src. A...	Dst. ...	Protocol	Src. Port	Dst. Port
;;; P8 - P2P								
0		mark connection	prerouting					
1		mark packet	prerouting					
2		jump	prerouting					
;;; P7 - DESCARGAS > 50M								
3		mark connection	prerouting			6 (tcp)		
4		mark packet	prerouting					
5		jump	prerouting					
;;; P1 - ADMINISTRACION RED Y DNS								
6		mark connection	prerouting			1 (icmp)		
7		mark connection	prerouting			6 (tcp)		21,8291
8		mark connection	prerouting			17 (udp)		53
9		mark packet	prerouting					
10		jump	prerouting					
;;; P2 - BASES DE DATOS								
11		mark connection	prerouting			6 (tcp)		3306,5432,1453
12		mark packet	prerouting					
13		jump	prerouting					
;;; P3 - WEB SERVICE								
14		mark connection	prerouting			6 (tcp)		3001
15		mark packet	prerouting					
16		jump	prerouting					
;;; P4 - NAVEGACION WEB								
17		mark connection	prerouting			6 (tcp)		80,443,8000-9000
18		mark packet	prerouting					
19		jump	prerouting					
;;; P5 - CORREO								
20		mark connection	prerouting			6 (tcp)		110,143,995,993,25,587
21		mark packet	prerouting					
22		jump	prerouting					
;;; P6 - OTROS								
23		mark connection	prerouting					
24		mark packet	prerouting					
25		accept	TERMINO D...					

Figura 306. Resumen de marcación de conexiones y paquetes en mikrotik
Fuente: Autor

6.3.2 Priorización de tráfico

Una vez que tenga marcadas las conexiones y paquetes de cada grupo de la tabla 53, se tiene que priorizar el tráfico, para ello en la misma tabla mencionada se puede ver la priorización que se otorga a cada grupo.

Antes de priorizar el tráfico, es necesario crear dos colas padres en el router principal de la red LAN del GAD Municipal de Chordeleg, que permitan clasificar el tráfico de bajada y subida por separado, para ello a través del software de administración WinBox, realizar los siguientes pasos en el router:

- Una vez que se esté dentro de la configuración del router mikrotik, dirigirse a la sección de "Queue Tree" y crear una cola con el nombre de "Bajada", luego seleccionar el puerto de la red LAN, que en este caso es el "Eth05_LANmuni" lo demás queda por defecto.

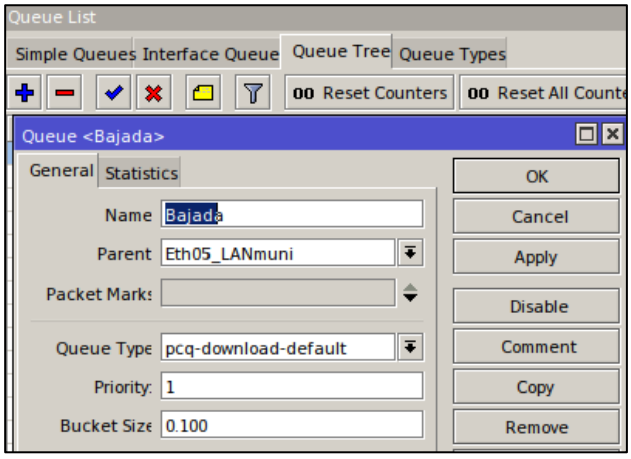


Figura 307. Cola padrea para priorizar el tráfico de bajada
Fuente: Autor

- Para los datos de subida únicamente seleccionar el puerto de internet que es el Eth01_WanCNT y lo nombrarlo como "Subida".

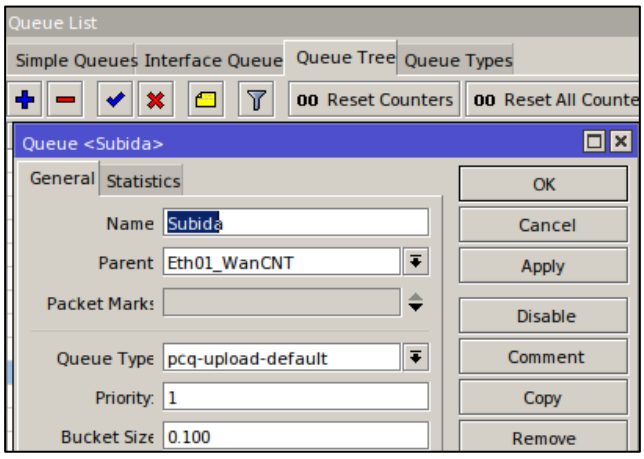


Figura 308. Cola padrea para priorizar el tráfico de subida
Fuente: Autor

6.3.2.1 Priorización del grupo de administración de red y dns

En base al método mencionado en la sección 5.4.1, se realizaron los siguientes pasos:

- Crear una cola en la sección “Queue Tree” con el nombre “P1 - ADMIN, DNS_down”, definir la cola padre “Bajada”, seleccionar los paquetes marcados en la sección 6.3.1.1 que en este caso es “PRIO 1” y por último le se asigna la prioridad “1”.

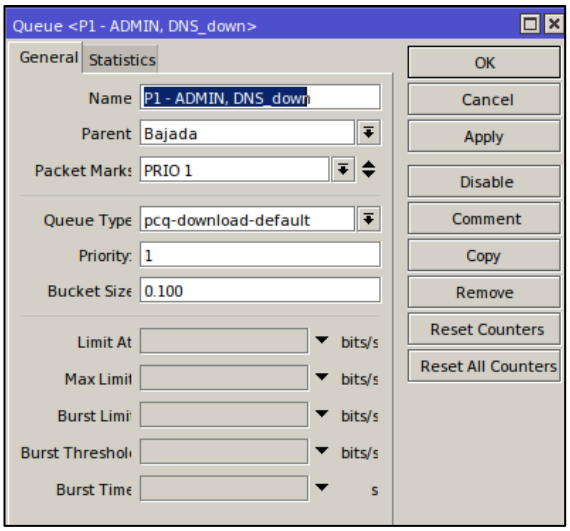


Figura 309. Priorizar el tráfico de bajada del grupo de administración de red y dns
Fuente: Autor

- Para el tráfico de subida únicamente cambia el nombre a “P1 - ADMIN, DNS_up” y la cola padre “Subida”.

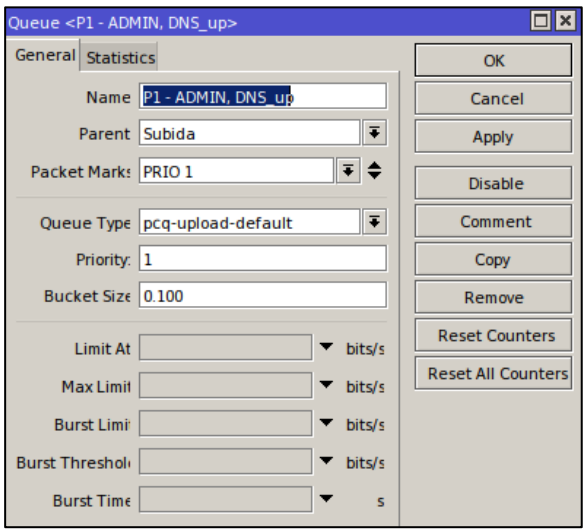


Figura 310. Priorizar el tráfico de bajada del grupo de administración de red y dns
Fuente: Autor

6.3.2.2 Priorización del grupo de base de datos

Siguiendo con el método anterior, se realizaron los siguientes pasos:

- Crear una cola en la sección “Queue Tree” con el nombre “P2 - BASES DE DATOS_down”, definir la cola padre “Bajada”, seleccionar los paquetes marcados en la sección 6.3.1.2 que en este caso es “PRIO 2” y por último asignar la prioridad “2”.

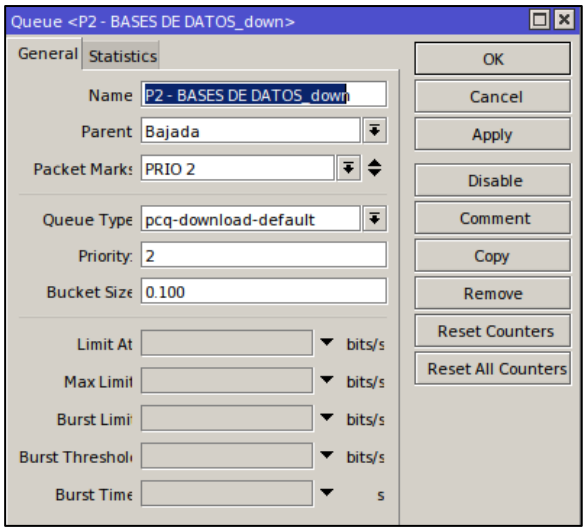


Figura 311. Priorizar el tráfico de bajada del grupo de base de datos
Fuente: Autor

- Para el tráfico de subida únicamente cambia el nombre a “P2 - BASES DE DATOS_up” y la cola padre “Subida”.

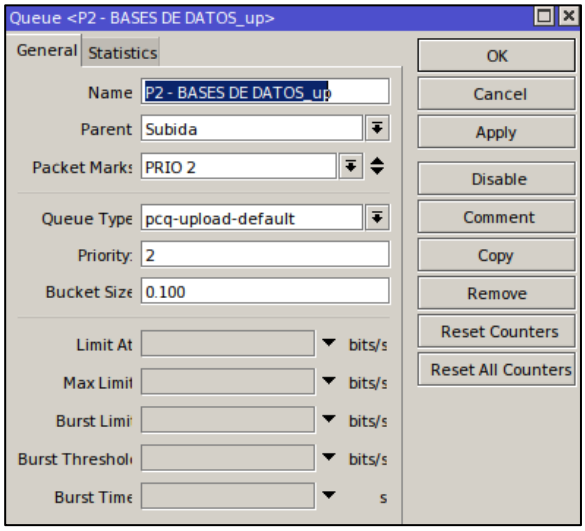


Figura 312. Priorizar el tráfico de bajada del grupo de base de datos
Fuente: Autor

6.3.2.3 Priorización del grupo de web service

Siguiendo con el método anterior, se realizaron los siguientes pasos:

- Crear una cola en la sección “Queue Tree” con el nombre “P3 – WEB SERVICE_down”, definir la cola padre “Bajada”, seleccionar los paquetes marcados en la sección 6.3.1.3 que en este caso es “PRIO 3”, se asigna la prioridad “3” y los límites del ancho de banda de bajada que resulta del capítulo 2.

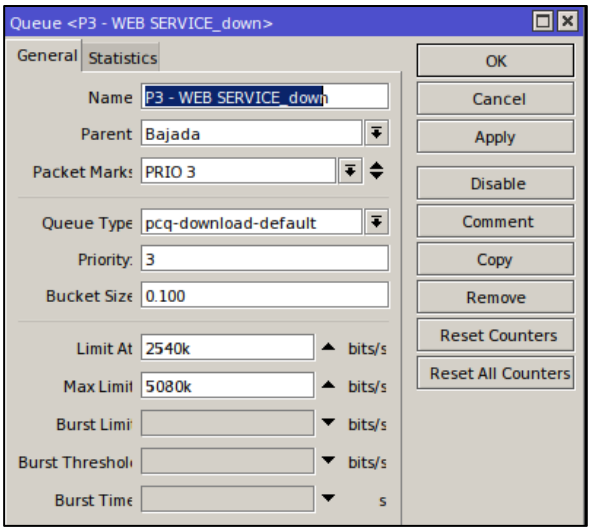


Figura 313. Priorizar el tráfico de bajada del grupo de web service
Fuente: Autor

- Para el tráfico de subida cambia el nombre a “P3 – WEB SERVICE_up”, la cola padre “Subida” y el ancho de banda de subida que resulta del capítulo 2.

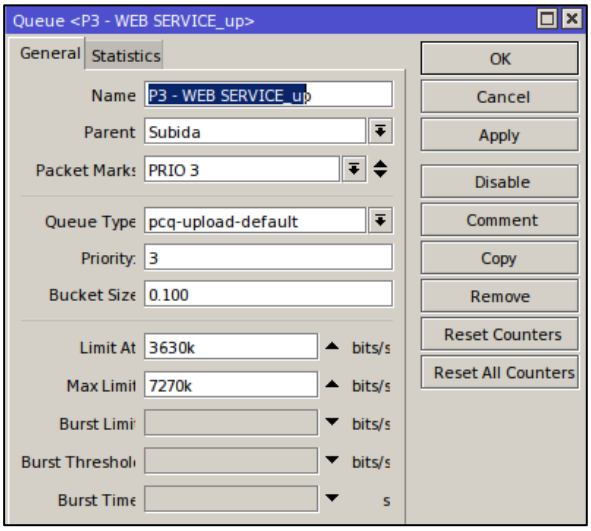


Figura 314. Priorizar el tráfico de bajada del grupo de web service
Fuente: Autor

6.3.2.4 Priorización del grupo de navegación web

Continuando con el mismo método, se realizaron los siguientes pasos:

- Crear una cola en la sección “Queue Tree” con el nombre “P4 – NAVEGACION WEB_down”, definir la cola padre “Bajada”, seleccionar los paquetes marcados en la sección 6.3.1.4 que en este caso es “PRIO 4” y se asigna la prioridad “4”.

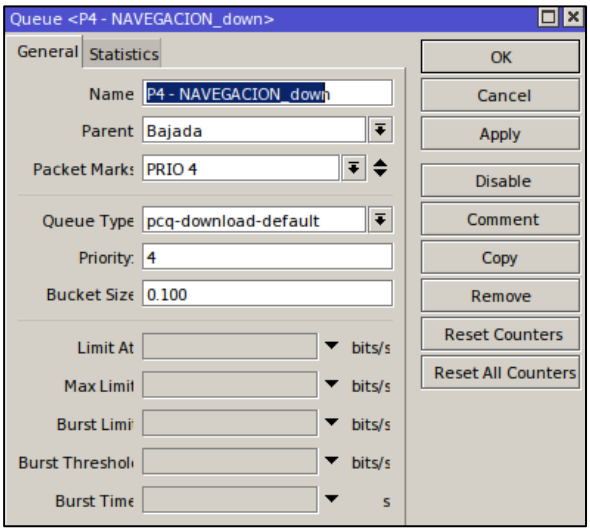


Figura 315. Priorizar el tráfico de bajada del grupo de navegación web
Fuente: Autor

- Para el tráfico de subida cambia el nombre a “P4 – NAVEGACION WEB_up” y la cola padre “Subida”.

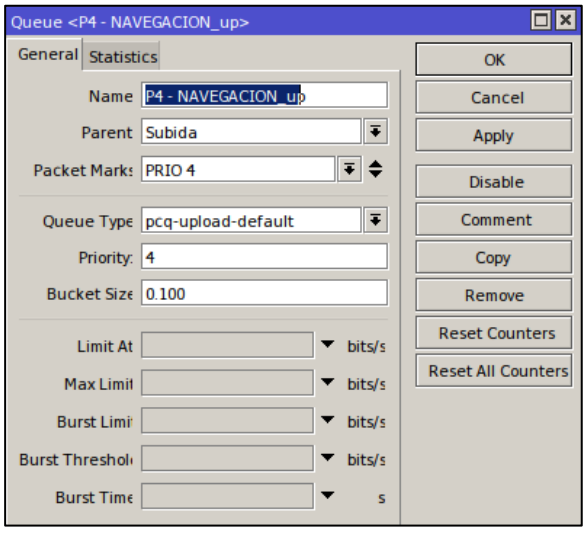


Figura 316. Priorizar el tráfico de bajada del grupo de navegación web
Fuente: Autor

6.3.2.5 Priorización del grupo de correo

De igual forma con el método anterior, se realizaron los siguientes pasos:

- Crear una cola en la sección “Queue Tree” con el nombre “P5 – CORREO_down”, definir la cola padre “Bajada”, seleccionar los paquetes marcados en la sección 6.3.1.5 que en este caso es “PRIO 5” y asignar la prioridad “5”.

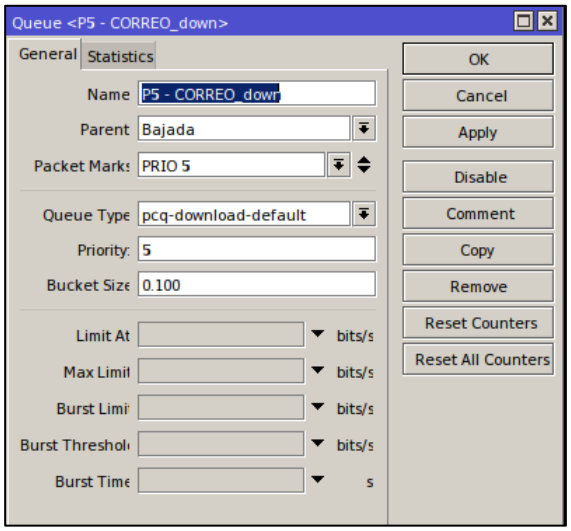


Figura 317. Priorizar el tráfico de bajada del grupo de correo
Fuente: Autor

- Para el tráfico de subida cambia el nombre a “P5 – CORREO_up” y la cola padre “Subida”.

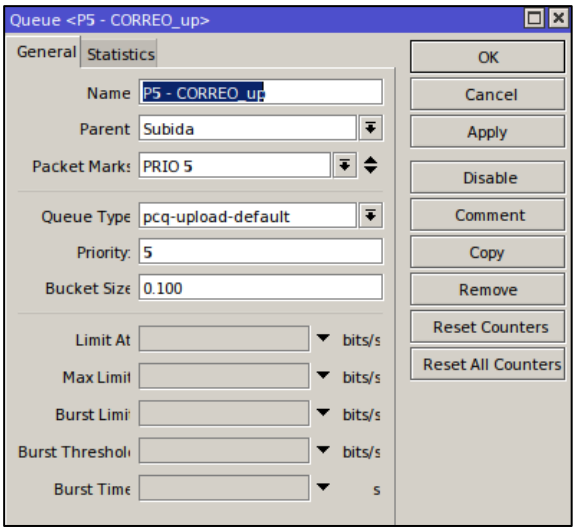


Figura 318. Priorizar el tráfico de bajada del grupo de correo
Fuente: Autor

6.3.2.6 Priorización del grupo de otros

Con el método anterior, se realizaron los siguientes pasos:

- Crear una cola en la sección “Queue Tree” con el nombre “P6 – OTROS_down”, definir la cola padre “Bajada”, seleccionar los paquetes marcados en la sección 6.3.1.6 que en este caso es “PRIO 6” y asignarle la prioridad “6”.

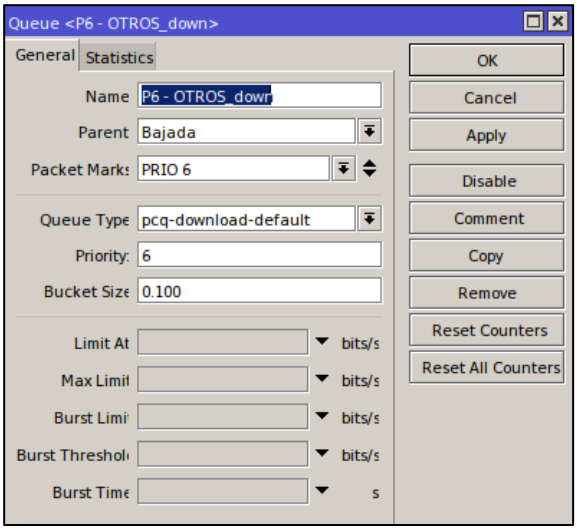


Figura 319. Priorizar el tráfico de bajada del grupo de otros
Fuente: Autor

- Para el tráfico de subida cambia el nombre a “P6 – OTROS_up” y la cola padre “Subida”.

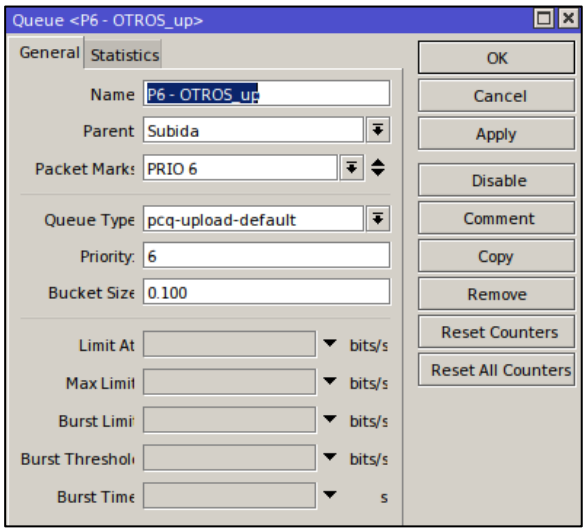


Figura 320. Priorizar el tráfico de bajada del grupo de otros
Fuente: Autor

6.3.2.7 Priorización del grupo de descargas mayores a 50 megas

Con el método anterior, se realizaron los siguientes pasos:

- Crear una cola en la sección “Queue Tree” con el nombre “P7 – DESCARGAS>50_down”, definir la cola padre “Bajada”, seleccionar los paquetes marcados en la sección 6.3.1.7 que en este caso es “PRIO 7” y asignar la prioridad “7”.

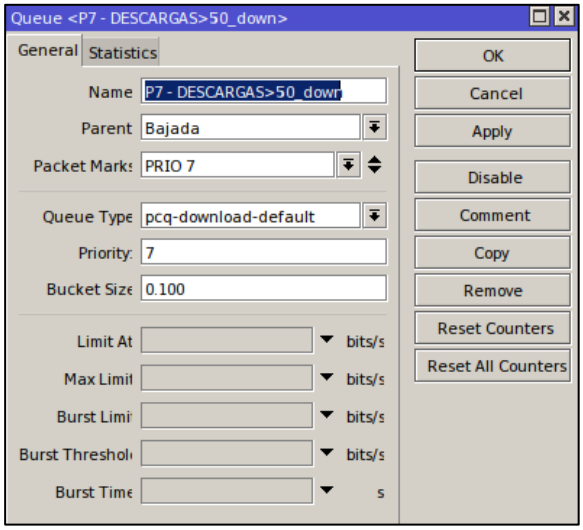


Figura 321. Priorizar el tráfico de bajada del grupo de descargas > 50 Megas
Fuente: Autor

- Para el tráfico de subida cambia el nombre a “P7 – DESCARGAS>50_up” y la cola padre “Subida”.

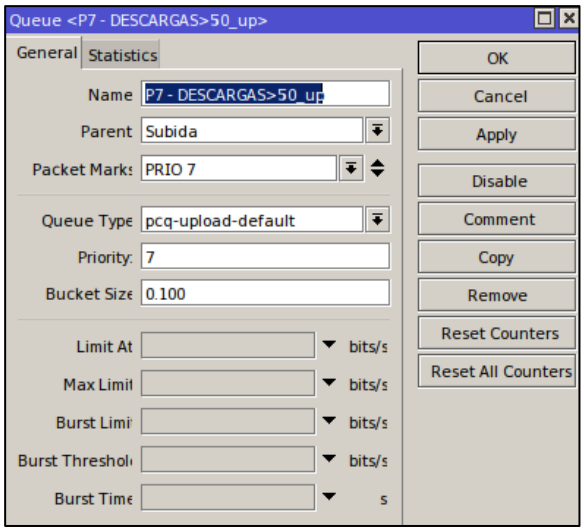


Figura 322. Priorizar el tráfico de bajada del grupo de descargas > 50 Megas
Fuente: Autor

6.3.2.8 Priorización del grupo de programas p2p

Se tiene la última priorización de tráfico, donde se realizaron los siguientes pasos:

- Crear una cola en la sección “Queue Tree” con el nombre “P8 – P2P_down”, definir la cola padre “Bajada”, seleccionar los paquetes marcados en la sección 6.3.1.8 que en este caso es “PRIO 8” y se asigna la prioridad “8”.

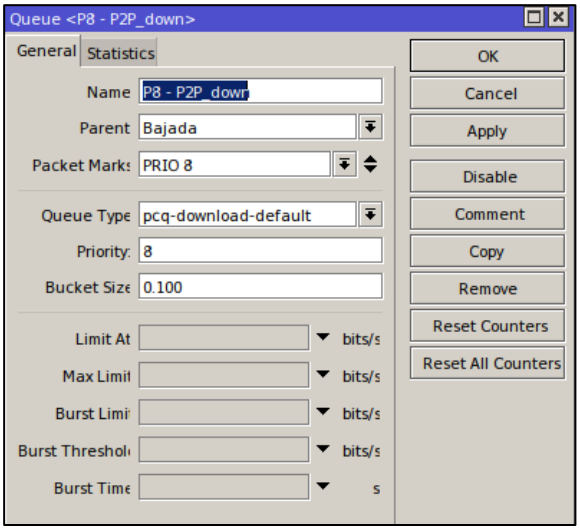


Figura 323. Priorizar el tráfico de bajada del grupo de programas P2P
Fuente: Autor

- Para el tráfico de subida cambia el nombre a “P8 – P2P_up” y la cola padre “Subida”.

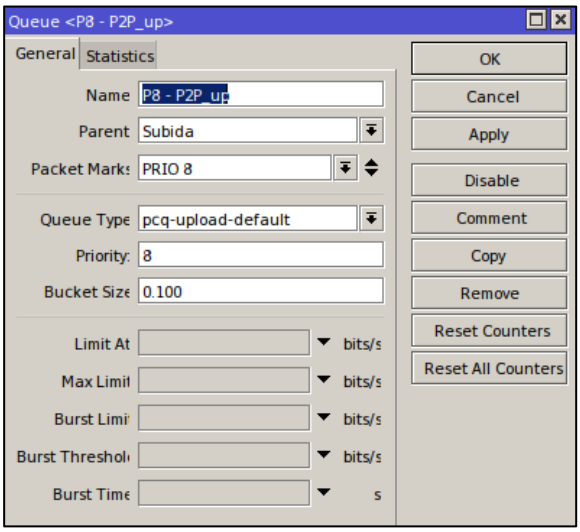
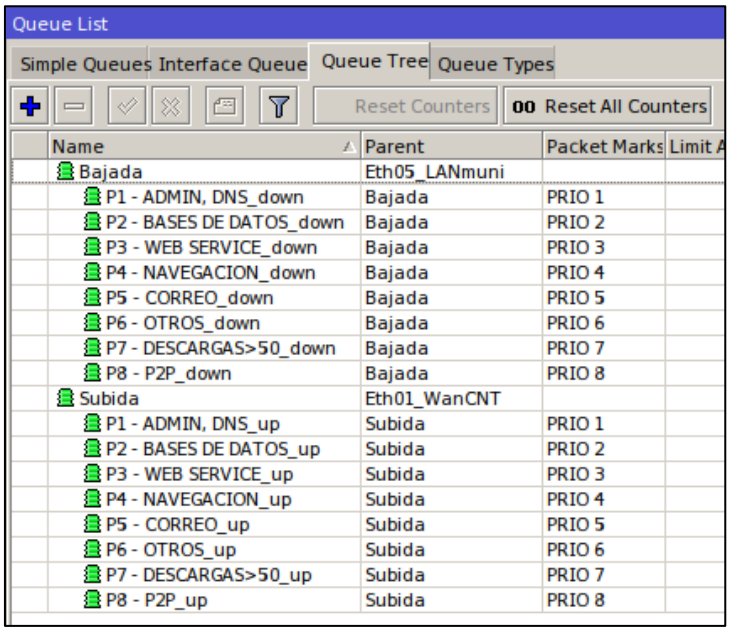


Figura 324. Priorizar el tráfico de bajada del grupo de programas P2P
Fuente: Autor

6.3.2.9 Resumen de priorización de tráfico

Finalmente en la siguiente figura se puede observar un resumen de todas las colas que se crearon anteriormente, para priorizar el tráfico de cada grupo de servicios:



Name	Parent	Packet Marks	Limit A
Bajada	Eth05_LANmuni		
P1 - ADMIN, DNS_down	Bajada	PRIO 1	
P2 - BASES DE DATOS_down	Bajada	PRIO 2	
P3 - WEB SERVICE_down	Bajada	PRIO 3	
P4 - NAVEGACION_down	Bajada	PRIO 4	
P5 - CORREO_down	Bajada	PRIO 5	
P6 - OTROS_down	Bajada	PRIO 6	
P7 - DESCARGAS>50_down	Bajada	PRIO 7	
P8 - P2P_down	Bajada	PRIO 8	
Subida	Eth01_WanCNT		
P1 - ADMIN, DNS_up	Subida	PRIO 1	
P2 - BASES DE DATOS_up	Subida	PRIO 2	
P3 - WEB SERVICE_up	Subida	PRIO 3	
P4 - NAVEGACION_up	Subida	PRIO 4	
P5 - CORREO_up	Subida	PRIO 5	
P6 - OTROS_up	Subida	PRIO 6	
P7 - DESCARGAS>50_up	Subida	PRIO 7	
P8 - P2P_up	Subida	PRIO 8	

Figura 325. Resumen de colas implementadas para priorizar el tráfico en mikrotik
Fuente: Autor

CAPITULO VII

7 Pruebas de campo

7.1 Calidad de servicio en el router principal

7.1.1 Comprobar la marcación de conexiones

Es necesario verificar que las conexiones de la red se están marcando, acorde a las etiquetas implementadas en la sección 6.3.1, donde se define 8 grupos con prioridades desde el nivel 1 hasta el 8. Haciendo uso de la herramienta “WinBox” para la administración del router principal mikrotik del GAD Municipal de Chordeleg, se realizan varias capturas de los diferentes grupos de conexiones; para ello se siguieron los siguientes pasos:

- Se ingresa a la administración del router principal, a través de la herramienta “WinBox”, luego dirigirse al “Firewall”, dar un clic en la pestaña “Connections” e iniciar con la captura de imágenes.

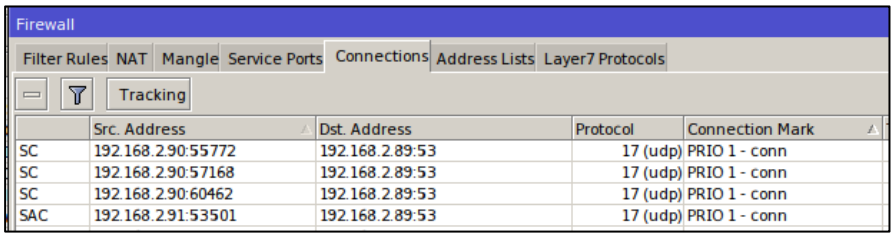


Figura 326. Conexiones marcadas para el grupo con etiqueta “PRIO 1 - conn”
Fuente: Autor

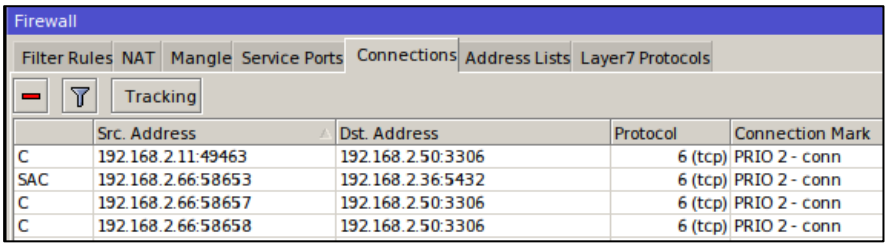


Figura 327. Conexiones marcadas para el grupo con etiqueta “PRIO 2 - conn”
Fuente: Autor

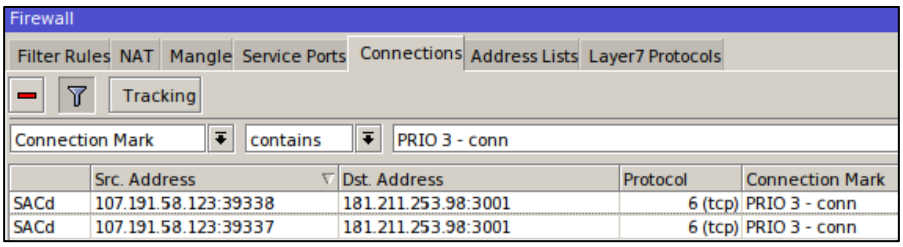


Figura 328. Conexiones marcadas para el grupo con etiqueta “PRIO 3 - conn”
Fuente: Autor

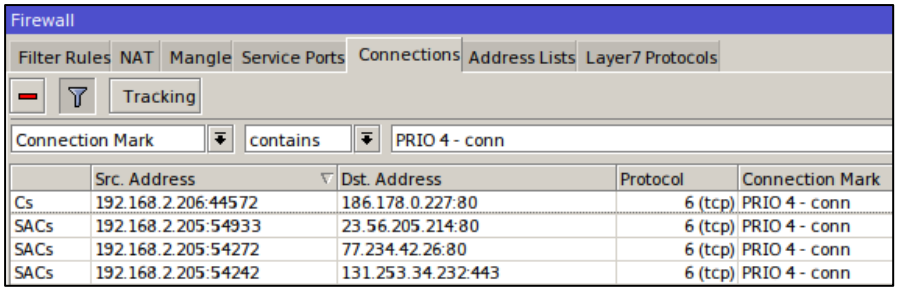


Figura 329. Conexiones marcadas para el grupo con etiqueta "PRIO 4 - conn"
Fuente: Autor

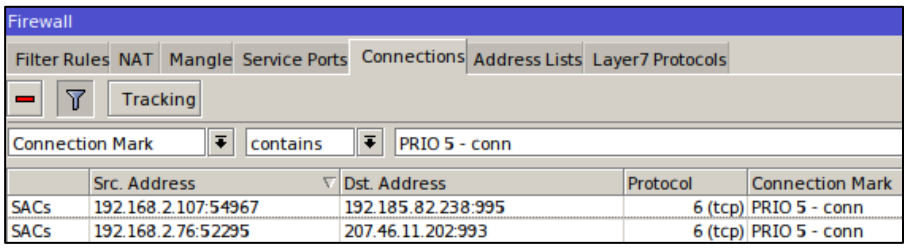


Figura 330. Conexiones marcadas para el grupo con etiqueta "PRIO 5 - conn"
Fuente: Autor

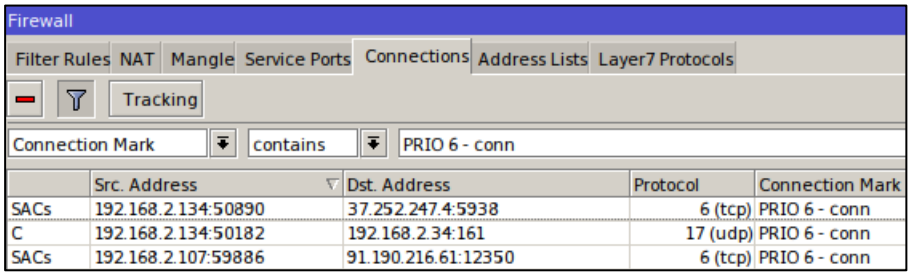


Figura 331. Conexiones marcadas para el grupo con etiqueta "PRIO 6 - conn"
Fuente: Autor

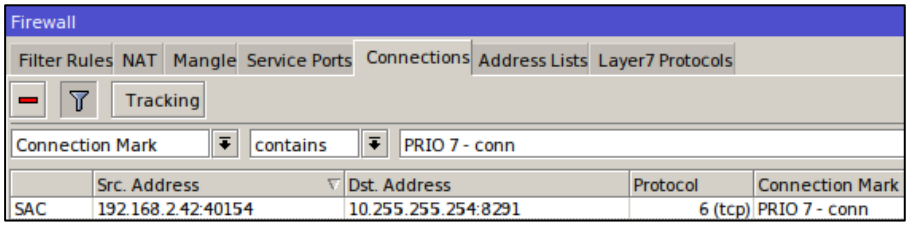


Figura 332. Conexiones marcadas para el grupo con etiqueta "PRIO 7 - conn"
Fuente: Autor

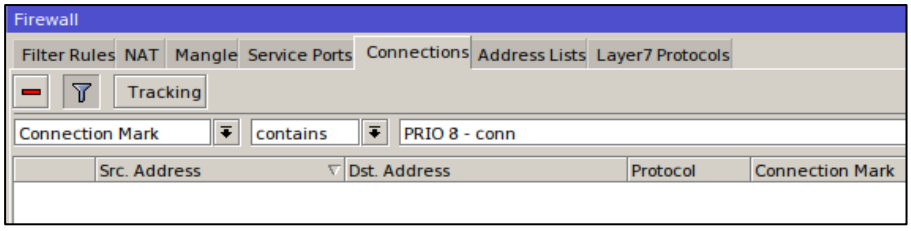


Figura 333. Conexiones marcadas para el grupo con etiqueta "PRIO 8 - conn"
Fuente: Autor

En todas las figuras anteriores se puede notar claramente que se están marcando las conexiones de manera satisfactoria, salvo los que pertenecen al grupo 8 ya que en el momento de la captura de datos no se tiene ningún software p2p descargando o compartiendo archivos, cabe recordar que solo el departamento de sistemas tiene instalado software p2p.

7.1.2 Comprobar la priorización del tráfico

Para comprobar que efectivamente están haciendo efecto las reglas configuradas para la calidad de servicio, se utilizó la herramienta "ping", para ello se realizaron las siguientes actividades:

- Partiendo de que actualmente están habilitadas las reglas para priorizar el tráfico de la red, abrir una terminal de la PC de desarrollo presionando "Ctrl + ALT + T" y se ejecuta el siguiente comando:

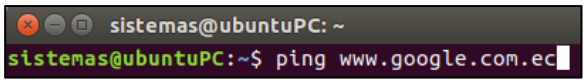


Figura 334. Realizar un ping a la www.google.com.ec
Fuente: Autor

- Paralelamente se inicia la descarga de un archivo y se obtiene el siguiente resultado:

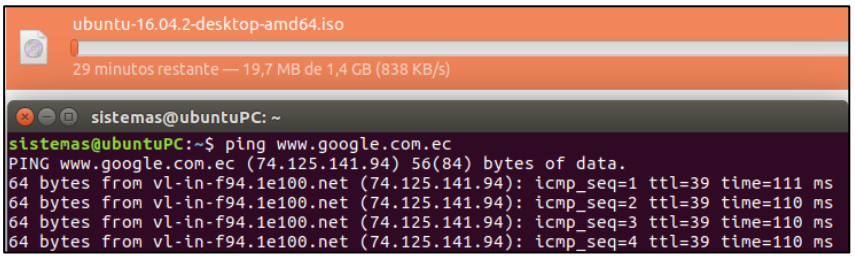


Figura 335. Realizando un ping con una descarga paralela con QoS
Fuente: Autor

- Luego se deshabilita las reglas de priorización de tráfico, haciendo uso de la herramienta "WinBox" para la administración del router principal, con la cual se ingresa al router y se dirige a "Queue Tree", se selecciona todas las reglas y se procede a deshabilitarlas momentáneamente.

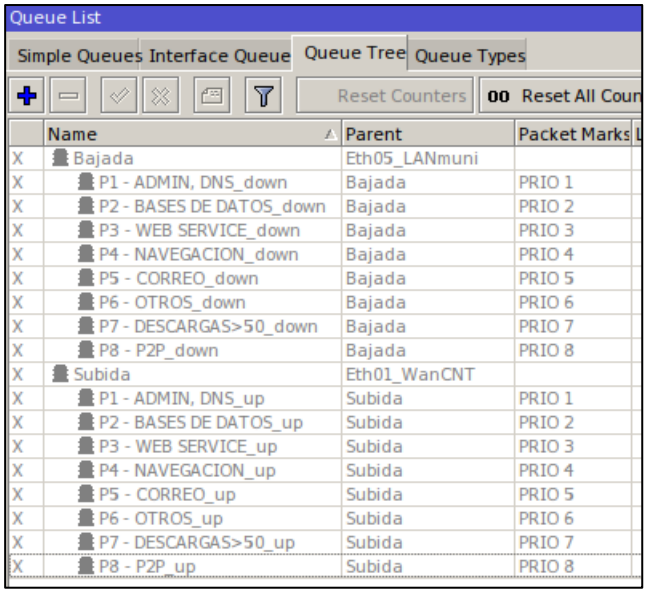


Figura 336. Deshabilitar las reglas de priorización de tráfico
Fuente: Autor

- Luego al regresar a observar el ping con la descarga paralela, se tiene los siguientes resultados:

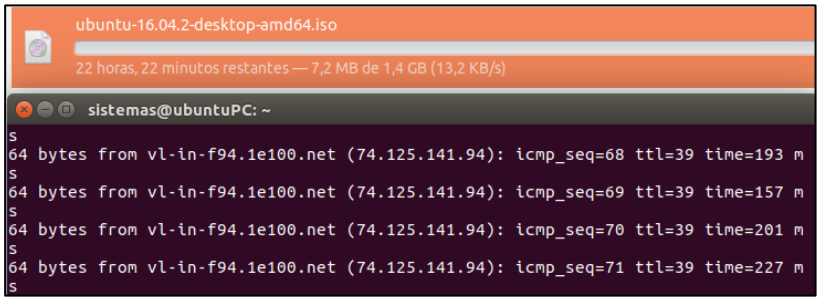


Figura 337. Realizando un ping con una descarga paralela sin QoS
Fuente: Autor

7.1.3 Análisis general

En la sección 7.1.1 se puede ver que las reglas implementadas en la sección 6.3.1, trabajan de manera efectiva marcando el tráfico de la red de acuerdo a las políticas establecidas; luego en la sección 7.1.2 se realiza una prueba para verificar la priorización de tráfico, es así que en la figura 324 se observa que se aplica calidad de servicio a una descarga y un ping simultáneamente, y se obtiene un tiempo de respuesta del ping de 110ms; sin embargo en la figura 326, se observa que al deshabilitar las reglas de calidad de servicio el tiempo de ping incrementa notablemente llegando hasta 227ms, esto se debe a que en las

reglas implementadas se da un nivel de priorización 1 al “ping” y al ser deshabilitadas no tiene tal priorización, lo cual repercute en el tiempo de respuesta.

En resumen las reglas implementadas para la calidad de servicio están trabajando de manera satisfactoria; más adelante en la sección 7.4, se observará que las reglas de calidad de servicio garantizan el ancho de banda para el web service.

7.2 Tráfico generado por cada consulta de la app “CHORDELEG móvil”

7.2.1 Instalación de la herramienta para capturar los datos transferidos

Para analizar los datos transferidos por cada consulta, se utilizará la herramienta “Wireshark”, la cual permitirá capturar, filtrar y observar detalladamente los bytes de cada consulta; la herramienta mencionada se instalará en la PC de desarrollo, realizando los siguientes pasos:

- Ingresar al terminal de la PC de desarrollo presionando “Ctrl + ALT + T” y ejecutar el siguiente comando:

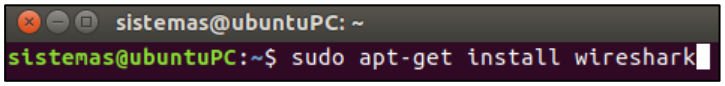


Figura 338. Instalando “Wireshark” en la pc de desarrollo
Fuente: Autor

- Para iniciar la herramienta simplemente se ejecuta lo siguiente:

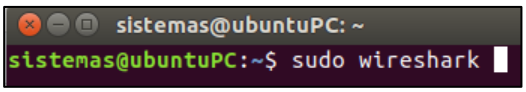


Figura 339. Comando para ejecutar “Wireshark” en la pc de desarrollo
Fuente: Autor

De esta manera queda lista la herramienta “Wireshark” para capturar los datos de cada consulta al web service.

7.2.2 Herramienta para observar el ancho de banda consumido

Para observar el consumo de ancho de banda por cada consulta que realice la app “CHORDELEG móvil”, se utilizará el router principal mikrotik de la red LAN del GAD

Municipal de Chordeleg, conociendo de antemano que se dispone de la herramienta de administración gráfica llamada “WinBox”.

7.2.3 Tráfico de datos de los principales tributos municipales

Recordando el capítulo 2, específicamente la sección 2.4.5; se había determinado que en promedio se tiene 60Bytes en cada consulta de los principales tributos municipales del GAD Municipal de Chordeleg, y de igual manera como respuesta se determinó que se recibe un promedio de 314.5Bytes por cada consulta. En la sección 6.2.3 se instaló la app “CHORDELEG móvil” en un dispositivo móvil Samsung note 5, por lo tanto haciendo uso de este equipo para realizar las consultas respectivas, se realizó una prueba por cada tributo municipal, donde a continuación se observará los datos transferidos y el ancho de banda consumido:

7.2.3.1 Consultas al predio urbano

Se realizaron las siguientes actividades:

- **Configurar la herramienta de captura de datos,** abrir la herramienta “Wireshark” en la pc de desarrollo, y seleccionar dando un doble clic en la tarjeta de red que se encuentra conectada a la red LAN del GAD Municipal de Chordeleg.

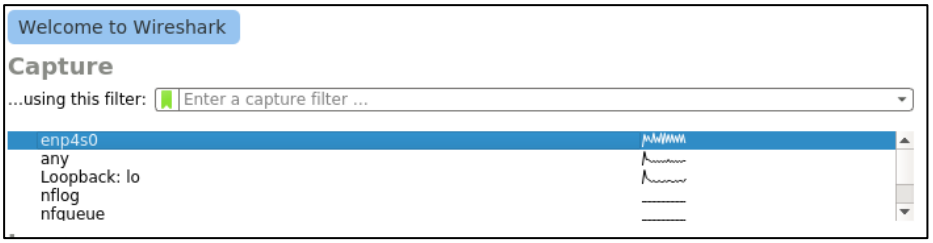


Figura 340. Seleccionando la tarjeta de red en “Wireshark”
Fuente: Autor

Luego crear un filtro para las conexiones http (la ip del web service es 192.168.2.53) y dar un clic en “Start Capturing Packets”, con lo cual queda lista la herramienta para capturar los Bytes transmitidos.

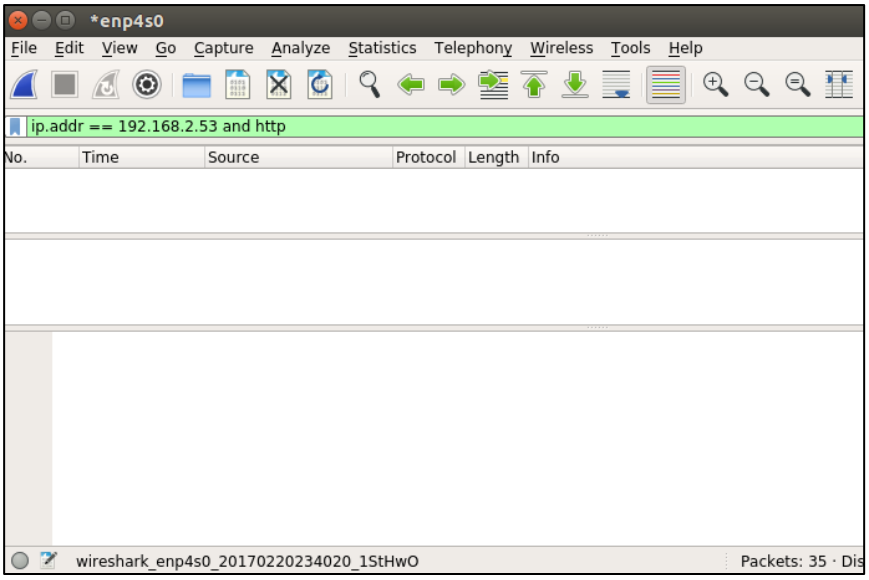


Figura 341. Iniciar la captura de paquetes en "Wireshark"
Fuente: Autor

- **Configurar la herramienta para observar el ancho de banda consumido,** ingresar al router principal mikrotik mediante la herramienta "WinBox", dirigirse a la sección de "Simple Queue", ubicar la regla creada en el capítulo 6 para ver el tráfico del servidor de producción definida en base a la dirección ip 192.168.2.53 y se da doble clic sobre ella.

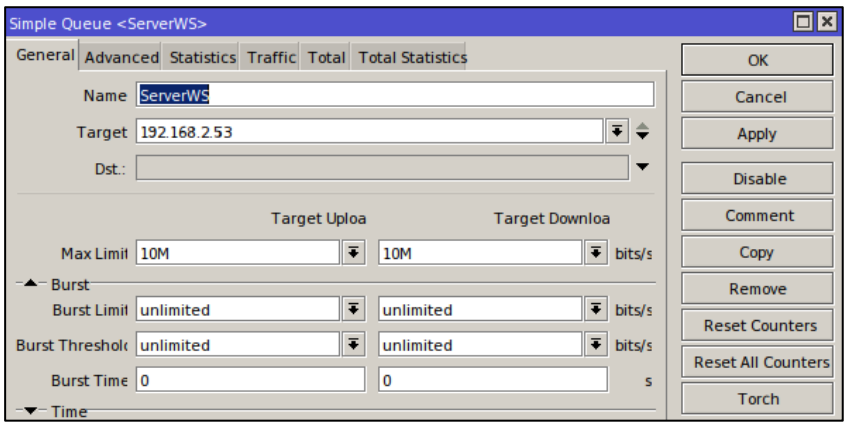


Figura 342. Cola simple para gestionar el ancho de banda del web service
Fuente: Autor

Luego dirigirse a la pestaña "Traffic" y queda lista la herramienta para observar el ancho de banda consumido.

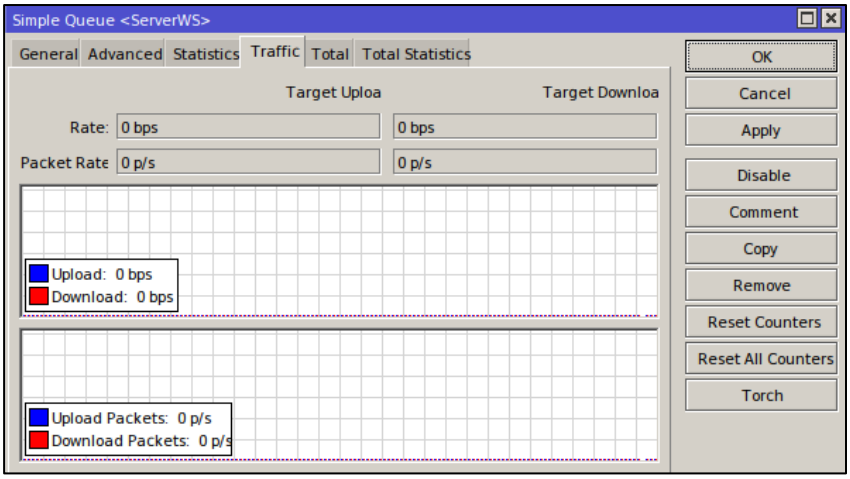


Figura 343. Ventana para observar el ancho de banda consumido en el web service
Fuente: Autor

- **Realizar las consultas del predio urbano,** abrir la app "CHORDELEG móvil" en el dispositivo móvil y de manera casi inmediata se abre automáticamente el menú principal de consultas, presionar en "PREDIO URBANO".

Figura 344. Menú principal de la app "CHORDELEG móvil"
Fuente: Autor

En la siguiente ventana, ingresar una clave catastral conocida del predio urbano y presionar en "CONSULTAR"

Figura 345. Consultar el predio urbano en la app "CHORDELEG móvil"
Fuente: Autor

Los datos consultados son presentados en la siguiente ventana.

Figura 346. Reporte del predio urbano en la app "CHORDELEG móvil"
Fuente: Autor

- **Resultados obtenidos en wireshark,** regresando a la herramienta que se preparó para la captura de datos, se tiene en primera instancia la captura de los bytes transmitidos desde la app hacia el web service.

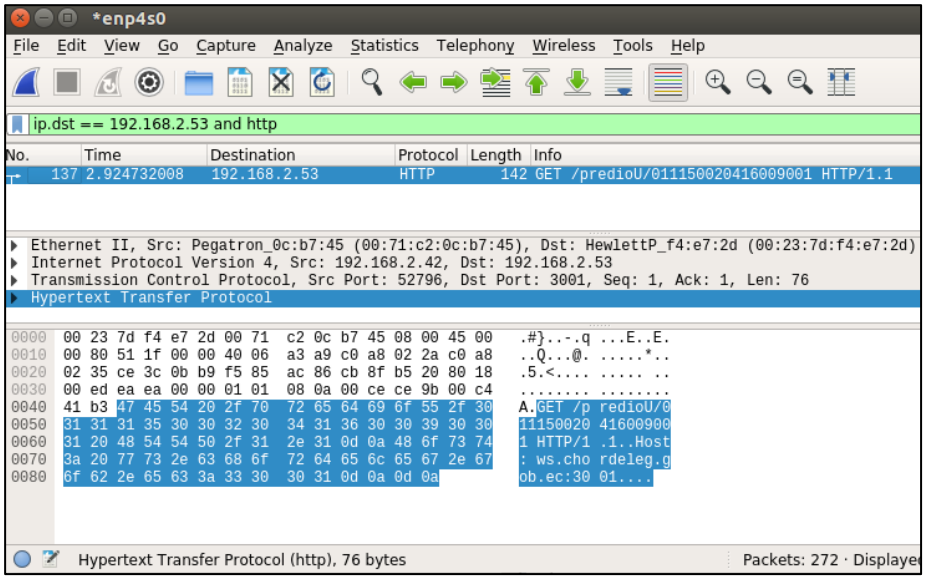


Figura 347. Bytes transmitidos desde la app por consulta al predio urbano
Fuente: Autor

Donde se observa que la app ha transmitido únicamente 76Bytes en su header; a continuación se tiene los datos transmitidos desde el web service hacia la app.

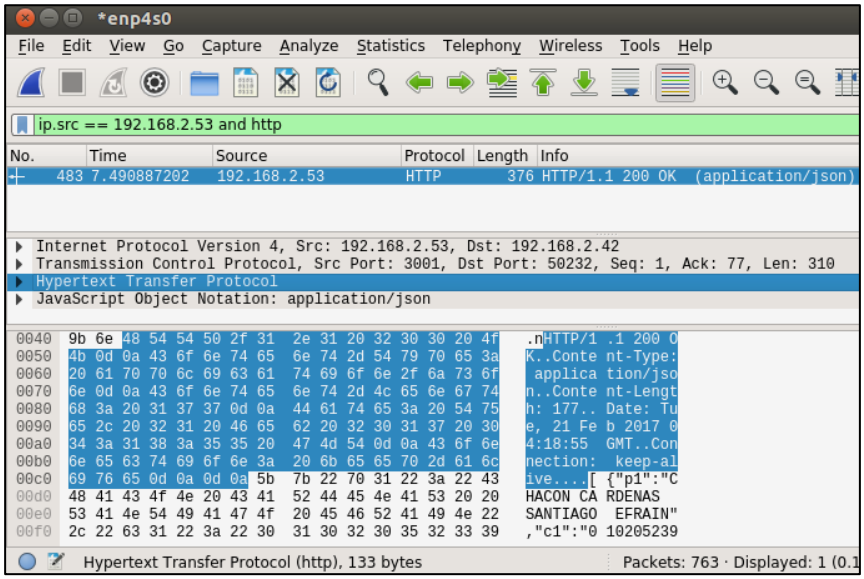


Figura 348. Bytes del header desde el web service por consulta al predio urbano
Fuente: Autor

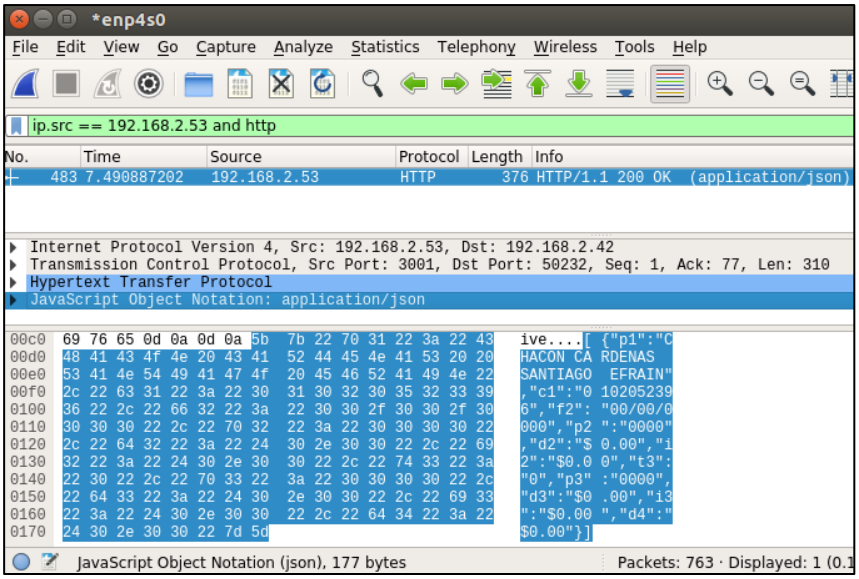


Figura 349. Bytes de info desde el web service por consulta al predio urbano
Fuente: Autor

De lo cual se puede ver que el web service respondió con 133Bytes del header y 177Bytes de la información solicitada, dando un total de 310Bytes transmitidos.

- **Resultados obtenidos en RouterOS,** regresando a la herramienta RouterOS que se preparó para observar el ancho de banda consumido por el web service, se captura el ancho de banda consumido.

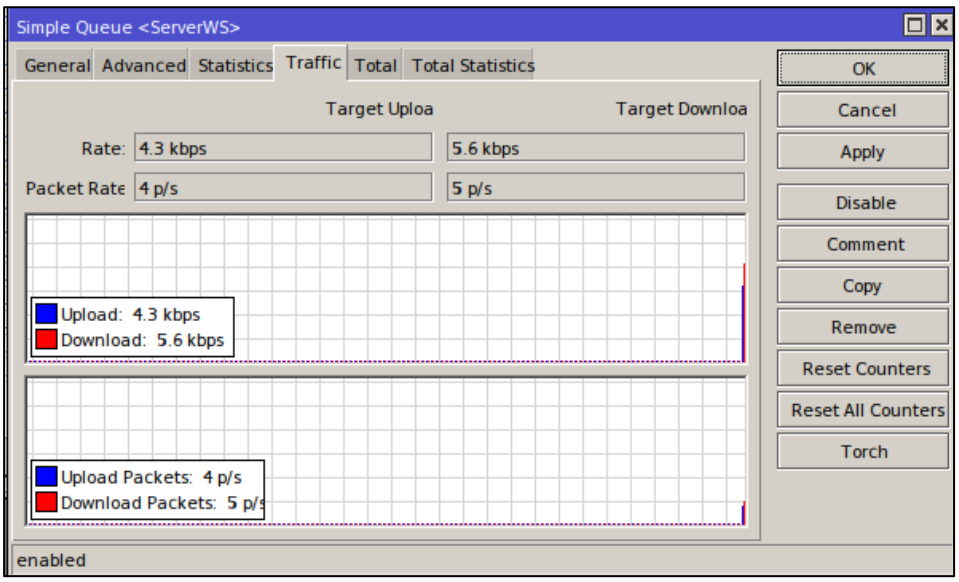


Figura 350. Ancho de banda consumido por consulta al predio urbano
Fuente: Autor

En la figura anterior se puede observar que la consulta consume un ancho de banda de bajada de 5.6kbps y de subida 4.3kbps.

7.2.3.2 Consulta del predio rural

Se realizaron las siguientes actividades:

- **Configurar la herramienta de captura de datos,** no hace falta porque se configuró en la sección 7.1.3.1, simplemente se reinicia la captura.
- **Configurar la herramienta para observar el ancho de banda consumido,** no hace falta porque se configuró en la sección 7.1.3.1.
- **Realizar las consultas del predio rural,** en la sección 7.1.3.1 se abrió la app "CHORDELEG móvil" en el dispositivo móvil, por lo tanto se regresa al menú principal y esta vez se presiona en "PREDIO RURAL", luego se ingresa una clave catastral conocida del predio rural y se presiona en "CONSULTAR".

Figura 351. Consultar el predio rural en la app "CHORDELEG móvil"
Fuente: Autor

Los datos consultados son presentados en la siguiente ventana.

Figura 352. Reporte del predio rural en la app "CHORDELEG móvil"
Fuente: Autor

- **Resultados obtenidos en Wireshark,** en la captura de datos se tiene en primera instancia los bytes transmitidos desde la app hacia el web service.

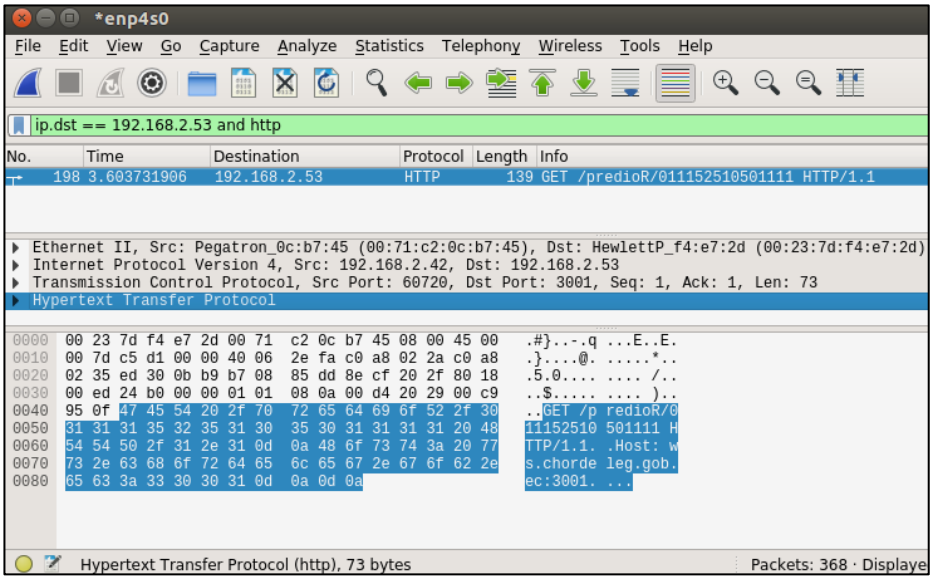


Figura 353. Bytes transmitidos desde la app por consulta al predio rural
Fuente: Autor

Donde se observa que la app ha transmitido únicamente 73Bytes en su header; a continuación se tiene los datos transmitidos desde el web service hacia la app.

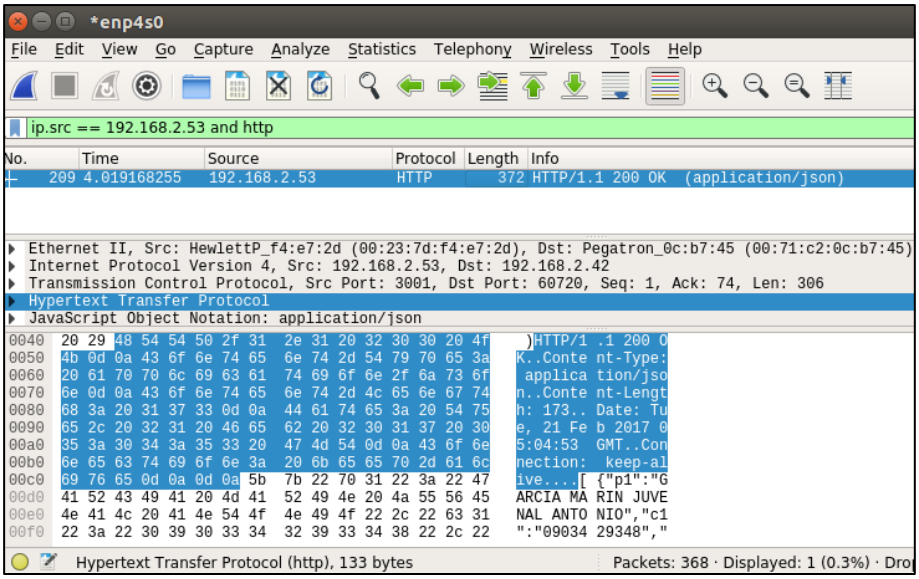


Figura 354. Bytes del header desde el web service por consulta al predio rural
Fuente: Autor

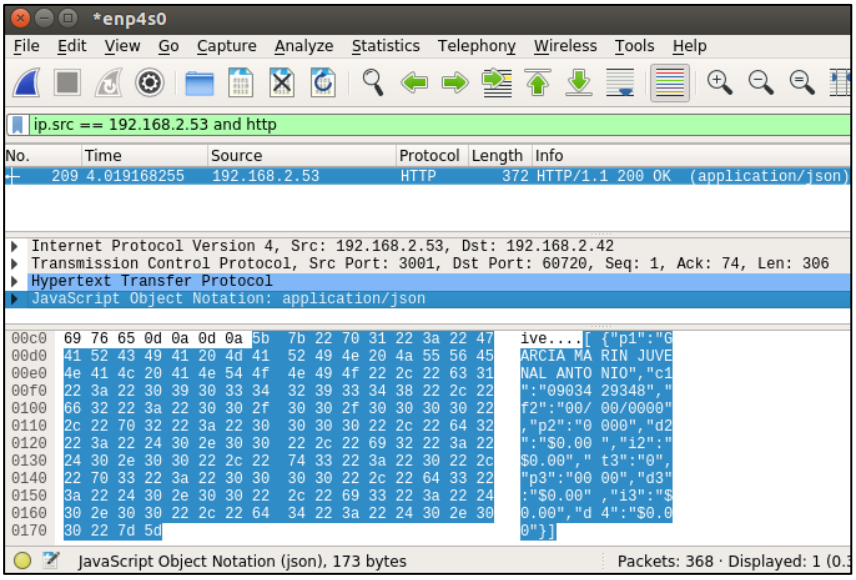


Figura 355. Bytes de info desde el web service por consulta al predio rural
Fuente: Autor

De lo cual se puede ver que el web service respondió con 133Bytes del header y 173Bytes de la información solicitada, dando un total de 306Bytes transmitidos.

- **Resultados obtenidos en RouterOS,** regresando a la ventana de routerOS para observar el ancho de banda consumido por el web service, se captura el ancho de banda consumido.

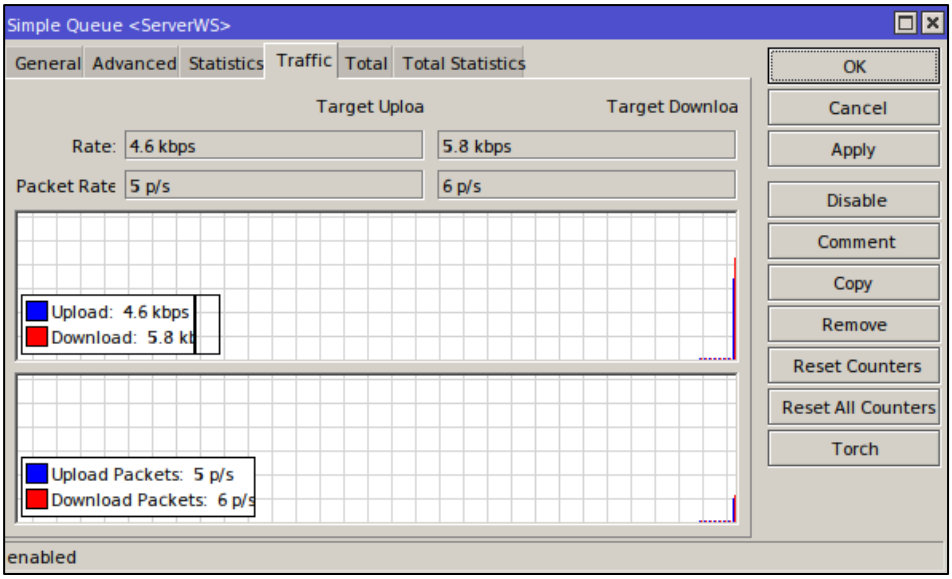


Figura 356. Ancho de banda consumido por consulta al predio rural
Fuente: Autor

En la figura anterior se observa que la consulta consume un ancho de banda de bajada de 5.8kbps y de subida 4.6kbps.

7.2.3.3 Consultas del agua potable

Se realizaron las siguientes actividades:

- **Configurar la herramienta de captura de datos,** no hace falta porque se configuró en la sección 7.1.3.1, simplemente se reinicia la captura.
- **Configurar la herramienta para observar el ancho de banda consumido,** no hace falta porque se configuró en la sección 7.1.3.1.
- **Realizar las consultas del agua potable,** en la sección 7.1.3.1 se abrió la app "CHORDELEG móvil" en el dispositivo móvil, por lo cual se regresa al menú principal y se presiona en "AGUA POTABLE", luego se ingresa una cuenta de medidor conocida y se presiona en "CONSULTAR".

Figura 357. Consultar el agua potable en la app "CHORDELEG móvil"
Fuente: Autor

Los datos consultados son presentados en la siguiente ventana.

Figura 358. Reporte del agua potable en la app "CHORDELEG móvil"
Fuente: Autor

- **Resultados obtenidos en Wireshark,** en la captura de datos se tiene en primera instancia los bytes transmitidos desde la app hacia el web service.

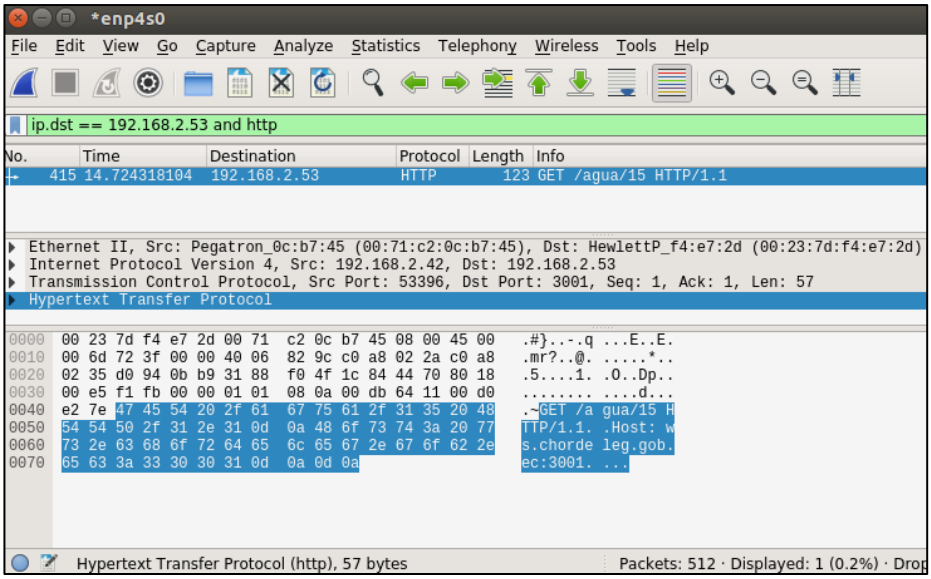


Figura 359. Bytes transmitidos desde la app por consulta al agua potable
Fuente: Autor

Donde se observa que la app ha transmitido únicamente 57Bytes en su header; a continuación se tiene los datos transmitidos desde el web service hacia la app.

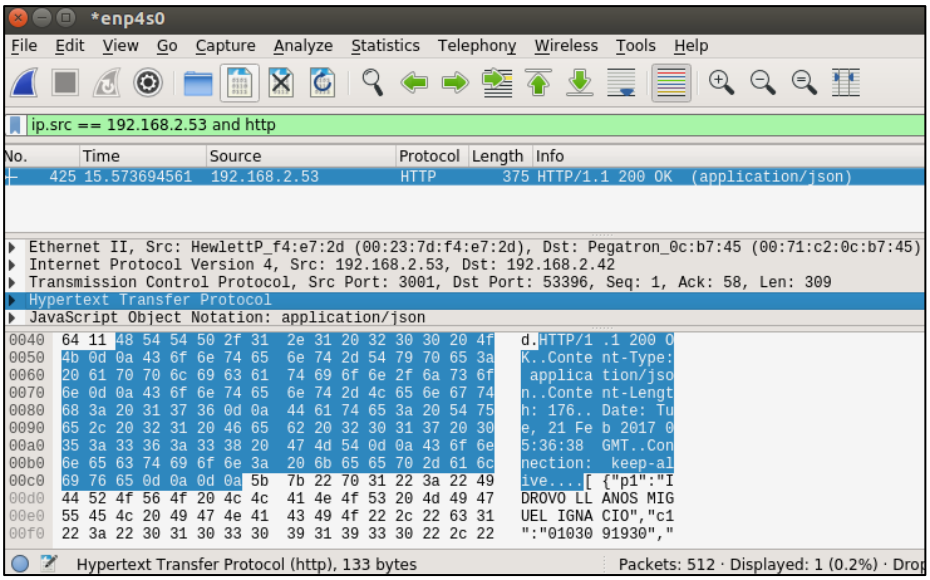


Figura 360. Bytes del header desde el web service por consulta al agua potable
Fuente: Autor

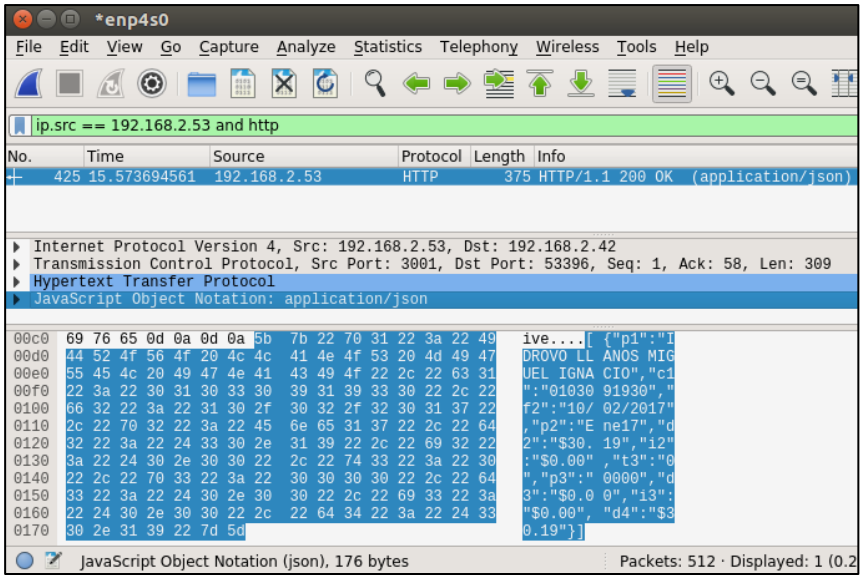


Figura 361. Bytes de info desde el web service por consulta al agua potable
Fuente: Autor

De lo cual se puede ver que el web service respondió con 133Bytes del header y 176Bytes de la información solicitada, dando un total de 309Bytes transmitidos.

- **Resultados obtenidos en RouterOS,** regresando a la ventana de routerOS para observar el ancho de banda consumido por el web service, se captura el ancho de banda consumido.

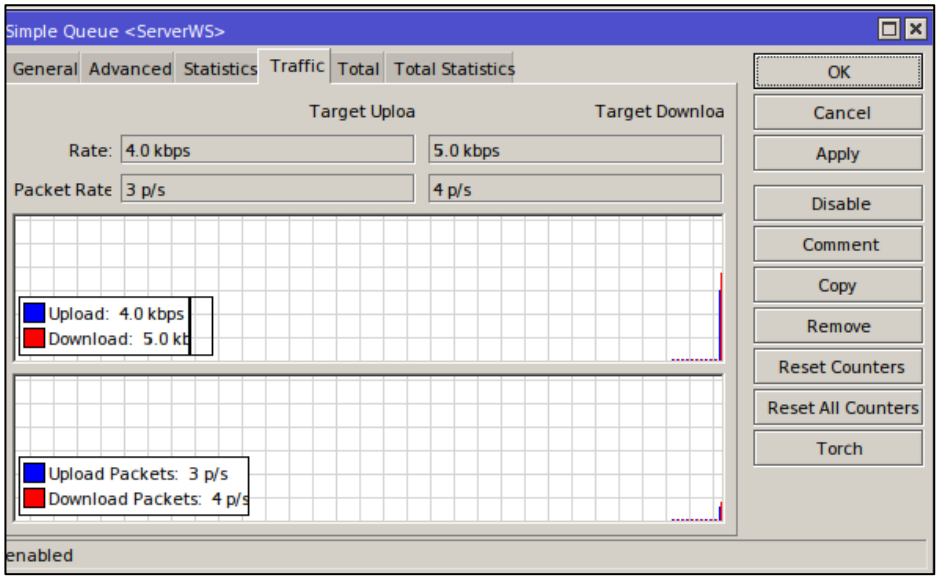


Figura 362. Ancho de banda consumido por consulta al agua potable
Fuente: Autor

En la figura anterior se observa que la consulta consume un ancho de banda de bajada de 5kbps y de subida 4kbps.

7.2.3.4 Consultas de la patente municipal

Se realizaron las siguientes actividades:

- **Configurar la herramienta de captura de datos,** no hace falta porque se configuró en la sección 7.1.3.1, simplemente se reinicia la captura.
- **Configurar la herramienta para observar el ancho de banda consumido,** no hace falta porque se configuró en la sección 7.1.3.1.
- **Realizar las consultas del agua potable,** en la sección 7.1.3.1 se observó como abrir la app “CHORDELEG móvil” en el dispositivo móvil, con lo cual se dirige al menú principal y se presiona en “PATENTE”, luego se ingresa un número de patente conocido y se presiona en “CONSULTAR”.

Figura 363. Consultar la patente en la app “CHORDELEG móvil”
Fuente: Autor

Los datos consultados son presentados en la siguiente ventana.

Figura 364. Reporte de la patente en la app "CHORDELEG móvil"
Fuente: Autor

- **Resultados obtenidos en Wireshark,** en la captura de datos se tiene en primera instancia los bytes transmitidos desde la app hacia el web service.

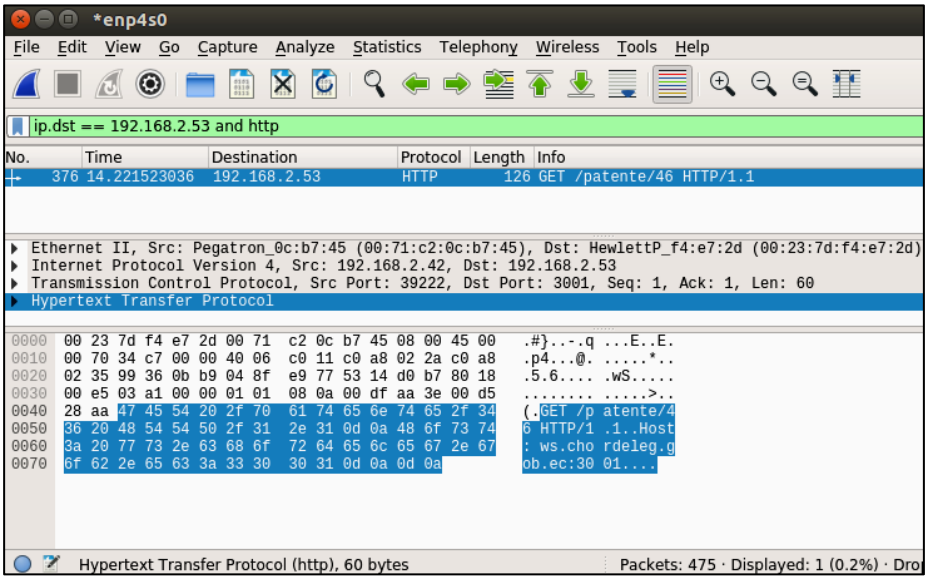


Figura 365. Bytes transmitidos desde la app por consulta de la patente
Fuente: Autor

Donde se observa que la app ha transmitido únicamente 60Bytes en su header; a continuación se tiene los datos transmitidos desde el web service hacia la app.

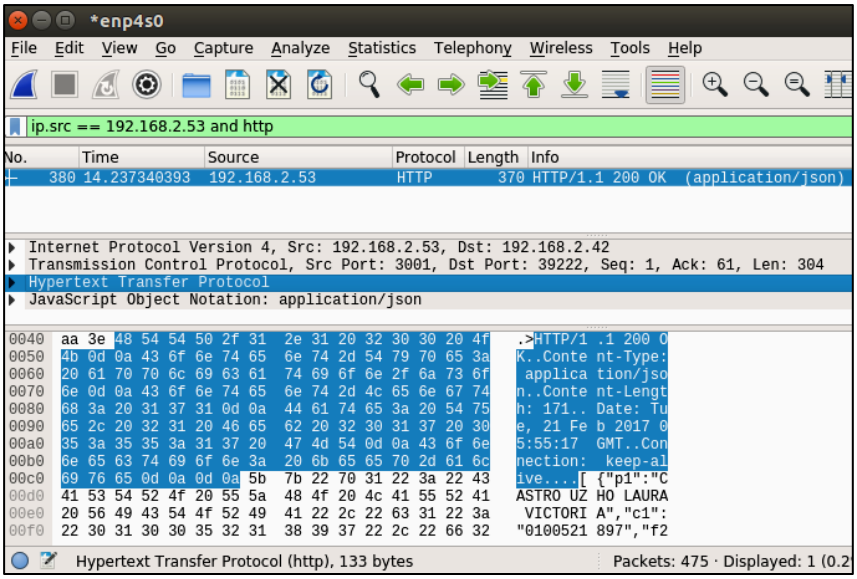


Figura 366. Bytes del header desde el web service por consulta a la patente
Fuente: Autor

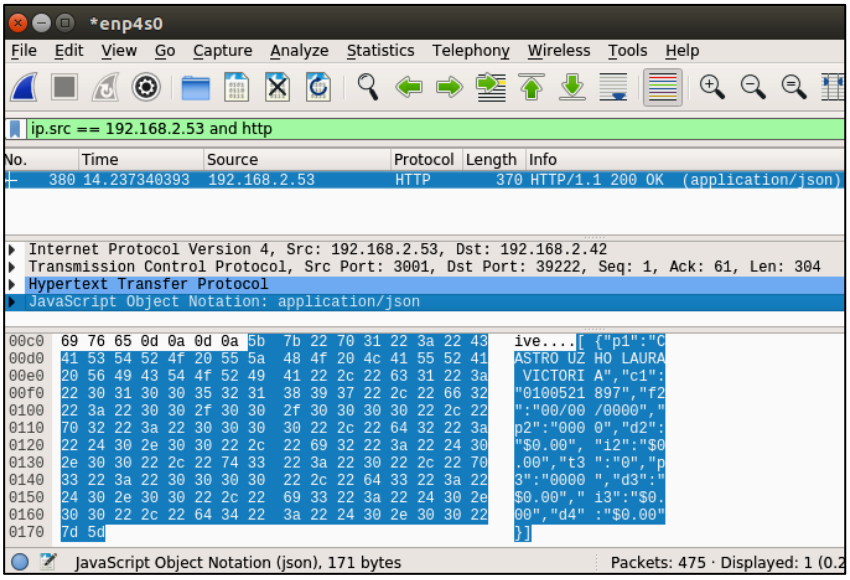


Figura 367. Bytes de info desde el web service por consulta a la patente
Fuente: Autor

De lo cual se puede ver que el web service respondió con 133Bytes del header y 171Bytes de la información solicitada, dando un total de 304Bytes transmitidos.

- **Resultados obtenidos en RouterOS,** regresando a la ventana de routerOS para observar el ancho de banda consumido por el web service, se captura el ancho de banda consumido.

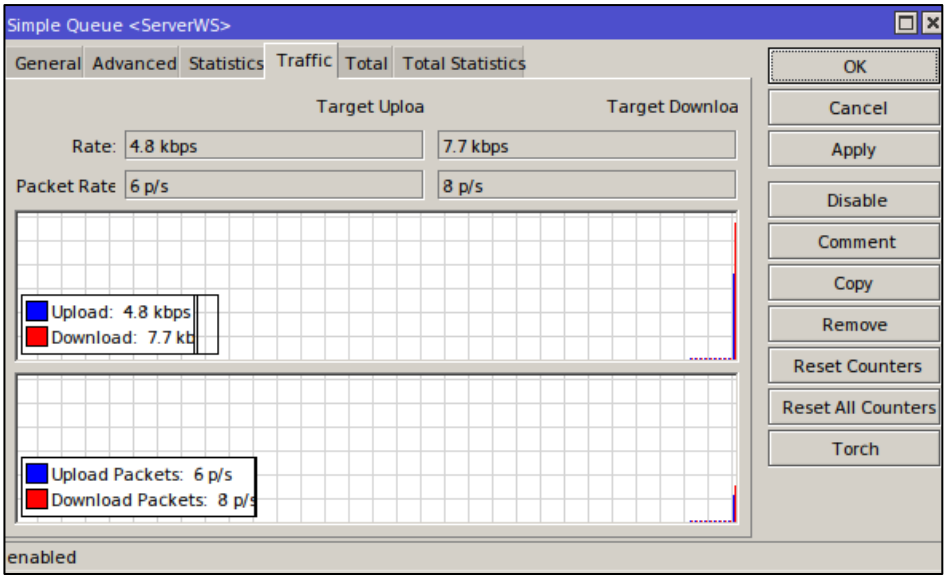


Figura 368. Ancho de banda consumido por consulta a la patente
Fuente: Autor

En la figura anterior se observa que la consulta consume un ancho de banda de bajada de 7.7kbps y de subida 4.8kbps.

7.2.3.5 Análisis general

Antes de proceder con un análisis general, se detallará en la siguiente tabla los resultados obtenidos a lo largo de la sección 7.1.3.

Tabla 55. Resumen de pruebas de campo obtenidas con la app "CHORDELEG móvil".

Consultas	Web Service			
	Datos Transmitidos		Ancho de banda consumido	
	Bajada	Subida	Bajada	Subida
Predio Urbano	76Bytes	310Bytes	4,3kbps	5,6kbps
Predio Rural	73Bytes	306Bytes	4,6kbps	5,8kbps
Agua Potable	57Bytes	309Bytes	4,0kbps	5,0kbps
Patente Municipal	60Bytes	304Bytes	4,8kbps	7,7kbps

Fuente: Autor

En el capítulo 2, se realizó un análisis de datos para calcular el tamaño promedio de datos a enviar y recibir por cada consulta, donde se determinó un promedio de 60Bytes de bajada y 314.5Bytes de subida por consulta. Por lo tanto al compararlo con los valores de la tabla anterior que fueron obtenidos en cada prueba de campo del proyecto implementado en el GAD Municipal de Chordeleg, se observa que tanto los datos de subida como los de bajada

en cada consulta están cerca del promedio calculado en el capítulo 2, es decir se cumple lo esperado por cada consulta realizada.

En cuanto al ancho de banda consumido se observa que el valor máximo se da en el sentido de subida, el mismo que llega a los 7.7kbps, que es igual a 0.96KB/s; siendo un valor relativamente pequeño que no afecta en absoluto al rendimiento de la red del GAD Municipal de Chordeleg, sin embargo cabe recordar que estos valores fueron tomados al realizar una consulta a la vez, en la siguiente sección se analizará el comportamiento del web service realizando consultas concurrentes.

7.3 Consultas concurrentes al web service

7.3.1 Instalación del script para realizar pruebas al web service

Recordando que el web service está desarrollado en el lenguaje de programación "JavaScript" y corre bajo el servidor de aplicaciones "Node.js", se recurre al script "loadtest" el cual es un módulo gratuito que está disponible en la red para realizar diferentes tipos de pruebas en servidores que trabajan con "Node.js" acorde a las políticas que se configuren; a continuación se instalará el script en la PC de desarrollo como equipo cliente para realizar las peticiones al servidor de producción:

- Ingresar al terminal de la PC de desarrollo presionando "Ctrl + ALT + T", y haciendo uso de "NPM" como gestor de módulos de Node.js, se ejecuta el siguiente comando:

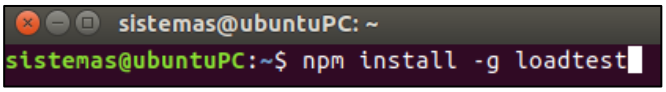


Figura 369. Instalando el módulo "loadtest" en Node.js
Fuente: Autor

De esta manera queda instalado y listo el script para realizar las pruebas en el web service.

7.3.2 Tráfico de datos en diferentes escenarios

En el capítulo 2, específicamente en la sección 2.5.1, se definieron 4 escenarios para realizar el cálculo del ancho de banda requerido por el sistema, los cuales fueron “Baja” para 1000 usuarios, “Media” para 4500 usuarios, “Alta” para 7000 usuarios y “Saturado” para el total de los usuarios proyectados que son 8799; es así que a continuación se analizará el comportamiento del web service en cada uno de los escenarios mencionados:

7.3.2.1 Escenario de concurrencia “Baja” de clientes

Para el escenario de concurrencia baja se definió que 1000 clientes realizaran peticiones al web service a través de la app de consulta, por lo tanto haciendo uso del módulo “loadtest” en la PC de desarrollo, se ejecutaron las pruebas de rendimiento por cada uno de los tributos principales del GAD Municipal de Chordeleg; a continuación se detalla paso a paso las actividades realizadas y los resultados obtenidos:

- **Predio Urbano,** en el desarrollo del web service se definió la ruta de consulta del predio urbano como http://ws.chordeleg.gob.ec/predioU/#codigo, donde #codigo se remplaza por una clave catastral conocida, por lo tanto se preparó el comando de “loadtest” para realizar 1000 peticiones a la ruta mencionada, quedando de la siguiente manera:

Figura 370. Baja concurrencia en el web service por consultar el predio urbano
Fuente: Autor

Se ejecutó el comando anterior, y una vez terminada la evaluación se obtienen los siguientes resultados:

```
INFO Max requests:        1000
INFO Concurrency level:   100
INFO Agent:               keepalive
INFO
INFO Completed requests:  1000
INFO Total errors:        0
INFO Total time:          0.755200591 s
INFO Requests per second: 1324
INFO Mean latency:        69.3 ms
INFO
INFO Percentage of the requests served within a certain time
INFO   50%      67 ms
INFO   90%      80 ms
INFO   95%      91 ms
INFO   99%      105 ms
INFO  100%      113 ms (longest request)
```

Figura 371. Resultados de baja concurrencia por consultas del predio urbano
Fuente: Autor

En la figura anterior se puede observar que el web service es capaz de responder 1000 peticiones del predio urbano en 0.76s.

- **Predio Rural,** así mismo en el desarrollo del web service se definió la ruta de consulta del predio rural como http://ws.chordeleg.gob.ec/predioR/#codigo, donde #codigo se remplaza por una clave catastral conocida, por lo tanto se preparó el comando de "loadtest" para realizar 1000 peticiones a la ruta mencionada, quedando de la siguiente manera:

```
sistemas@ubuntuPC: ~
sistemas@ubuntuPC:~$ loadtest --keepalive -c 100 -n 1000 http://ws.chordeleg.gob.e
c:3001/predioR/011151510323025
```

Figura 372. Baja concurrencia en el web service por consultar el predio rural
Fuente: Autor

Se ejecutó el comando anterior y una vez terminada la evaluación se obtienen los siguientes resultados:

```
INFO Max requests:        1000
INFO Concurrency level:   100
INFO Agent:               keepalive
INFO
INFO Completed requests:  1000
INFO Total errors:        0
INFO Total time:          0.718129275 s
INFO Requests per second: 1393
INFO Mean latency:        66.1 ms
INFO
INFO Percentage of the requests served within a certain time
INFO   50%      58 ms
INFO   90%      98 ms
INFO   95%      112 ms
INFO   99%      145 ms
INFO  100%      152 ms (longest request)
```

Figura 373. Resultados de baja concurrencia por consultas del predio urbano rural
Fuente: Autor

En la figura anterior se puede observar que el web service es capaz de responder 1000 peticiones del predio rural en 0.72s.

- **Agua Potable,** para el desarrollo del web service se definió la ruta de consulta del agua potable como http://ws.chordeleg.gob.ec/agua/#codigo, donde #codigo se remplaza por una código de un medidor conocido, por lo tanto se preparó el comando de "loadtest" para realizar 1000 peticiones a la ruta mencionada, quedando de la siguiente manera:

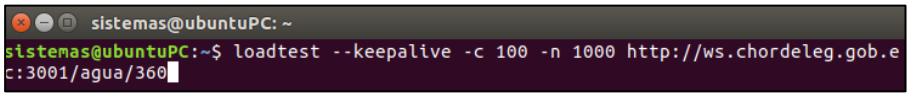


Figura 374. Baja concurrencia en el web service por consultar el agua potable
Fuente: Autor

Se ejecutó el comando anterior, y una vez terminada la evaluación se obtienen los siguientes resultados:

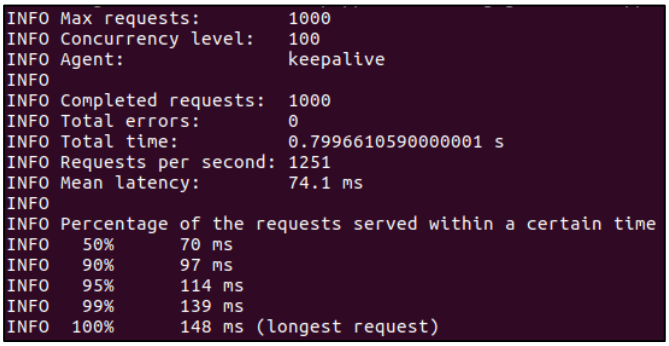


Figura 375. Resultados de baja concurrencia por consultas del agua potable
Fuente: Autor

En la figura anterior se puede observar que el web service es capaz de responder 1000 peticiones del agua potable en 0.80s.

- **Patente Municipal,** para el desarrollo del web service se definió la ruta de consulta de la patente municipal como http://ws.chordeleg.gob.ec/patente/#codigo, donde #codigo se remplaza por un número de patente conocido, por lo tanto se preparó el comando de "loadtest" para realizar 1000 peticiones a la ruta mencionada, quedando de la siguiente manera:

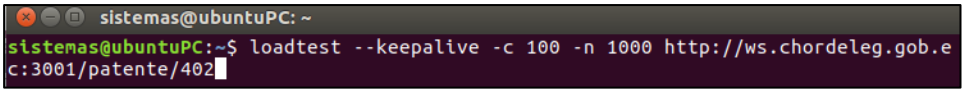


Figura 376. Baja concurrencia en el web service por consultar la patente
Fuente: Autor

Se ejecutó el comando anterior y una vez terminada la evaluación se obtienen los siguientes resultados:

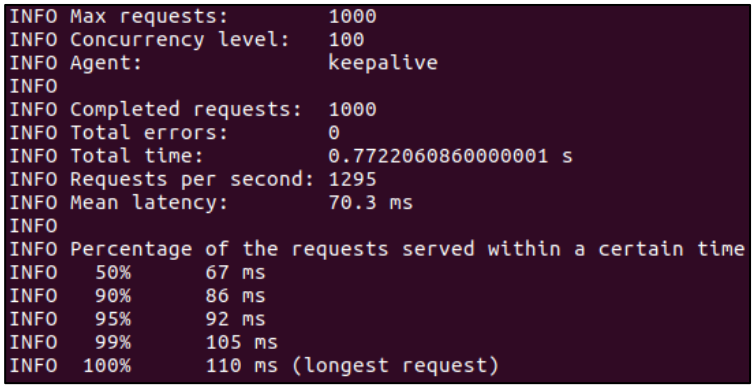


Figura 377. Resultados de baja concurrencia por consultas de la patente
Fuente: Autor

En la figura anterior se puede observar que el web service es capaz de responder 1000 peticiones de la patente municipal en 0.77s.

7.3.2.2 Escenario de concurrencia “Media” de clientes

Para el escenario de concurrencia “Media” se definió que 4500 clientes realizaran peticiones al web service a través de la app de consulta, por lo tanto haciendo uso del módulo “loadtest” en la PC de desarrollo, se ejecutaron las pruebas de rendimiento por cada uno de los tributos principales del GAD Municipal de Chordeleg. A diferencia que la sección anterior únicamente se cambia el número de clientes de los comandos de prueba de “loadtest”, que en este caso son 4500 clientes, a continuación se presenta los comandos y resultados obtenidos:

- **Predio Urbano:**

Figura 378. Media concurrencia en el web service por consultar el predio urbano
Fuente: Autor

```
INFO Max requests:        4500
INFO Concurrency level:   100
INFO Agent:               keepalive
INFO
INFO Completed requests:  4500
INFO Total errors:        0
INFO Total time:          3.28977978 s
INFO Requests per second: 1368
INFO Mean latency:        71.8 ms
INFO
INFO Percentage of the requests served within a certain time
INFO   50%      71 ms
INFO   90%      79 ms
INFO   95%      88 ms
INFO   99%      96 ms
INFO  100%      112 ms (longest request)
```

Figura 379. Resultados de media concurrencia por consultas de predio urbano
Fuente: Autor

Donde se puede observar que el web service es capaz de responder 4500 peticiones del predio urbano en 3.29s.

- **Predio Rural:**

```
sistemas@ubuntuPC: ~
sistemas@ubuntuPC:~$ loadtest --keepalive -c 100 -n 4500 http://ws.chordeleg.gob.e
c:3001/predioR/011151510323025
```

Figura 380. Media concurrencia en el web service por consultar el predio rural
Fuente: Autor

```
INFO Max requests:        4500
INFO Concurrency level:   100
INFO Agent:               keepalive
INFO
INFO Completed requests:  4500
INFO Total errors:        0
INFO Total time:          3.1173904539999997 s
INFO Requests per second: 1444
INFO Mean latency:        67.8 ms
INFO
INFO Percentage of the requests served within a certain time
INFO   50%      67 ms
INFO   90%      72 ms
INFO   95%      74 ms
INFO   99%      98 ms
INFO  100%      125 ms (longest request)
```

Figura 381. Resultados de media concurrencia por consultas de predio rural
Fuente: Autor

Donde se puede observar que el web service es capaz de responder 4500 peticiones del predio rural en 3.12s.

- **Agua Potable:**

```
sistemas@ubuntuPC: ~
sistemas@ubuntuPC:~$ loadtest --keepalive -c 100 -n 4500 http://ws.chordeleg.gob.e
c:3001/aqua/360
```

Figura 382. Media concurrencia en el web service por consultar el agua potable
Fuente: Autor

```
INFO Max requests:        4500
INFO Concurrency level:   100
INFO Agent:               keepalive
INFO
INFO Completed requests:  4500
INFO Total errors:        0
INFO Total time:          3.214469201 s
INFO Requests per second: 1400
INFO Mean latency:        70.1 ms
INFO
INFO Percentage of the requests served within a certain time
INFO   50%      68 ms
INFO   90%      75 ms
INFO   95%      78 ms
INFO   99%      105 ms
INFO  100%      140 ms (longest request)
```

Figura 383. Resultados de media concurrencia por consultas de agua potable
Fuente: Autor

Donde se puede observar que el web service es capaz de responder 4500 peticiones del agua potable en 3.21s.

- **Patente Municipal:**

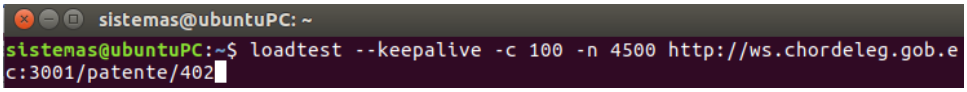


Figura 384. Media concurrencia en el web service por consultar la patente
Fuente: Autor

```
INFO Max requests:        4500
INFO Concurrency level:   100
INFO Agent:               keepalive
INFO
INFO Completed requests:  4500
INFO Total errors:        0
INFO Total time:          3.227407498 s
INFO Requests per second: 1394
INFO Mean latency:        70.4 ms
INFO
INFO Percentage of the requests served within a certain time
INFO   50%      69 ms
INFO   90%      79 ms
INFO   95%      81 ms
INFO   99%      102 ms
INFO  100%      139 ms (longest request)
```

Figura 385. Resultados de media concurrencia por consultas de la patente
Fuente: Autor

Donde se puede observar que el web service es capaz de responder 4500 peticiones de la patente municipal en 3.23s.

7.3.2.3 Escenario de concurrencia “Alta” de clientes

Para el escenario de concurrencia “Alta” se definió que 7000 clientes realizaran peticiones al web service a través de la app de consulta, por lo tanto haciendo uso del módulo

"loadtest" en la PC de desarrollo, se ejecutaron las pruebas de rendimiento por cada uno de los tributos principales del GAD Municipal de Chordeleg. A diferencia que la sección anterior únicamente se cambió el número de clientes de los comandos de prueba de "loadtest", que en este caso son 7000 clientes, a continuación se presenta los comandos y resultados obtenidos:

- **Predio Urbano:**

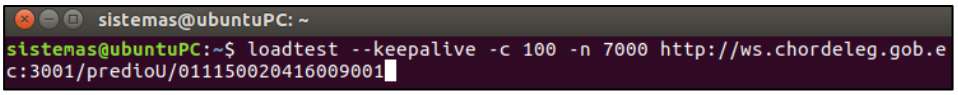


Figura 386. Alta concurrencia en el web service por consultar el predio urbano
Fuente: Autor

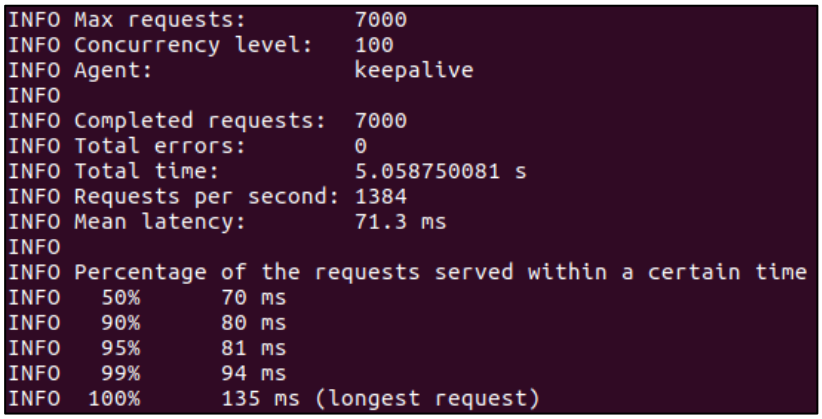


Figura 387. Resultados de alta concurrencia por consultas del predio urbano
Fuente: Autor

Donde se puede observar que el web service es capaz de responder 7000 peticiones del predio urbano en 5.06s.

- **Predio Rural:**

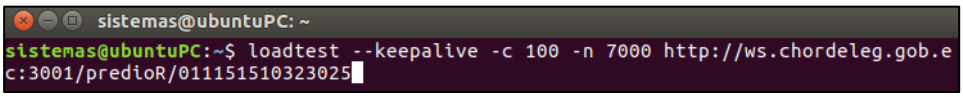


Figura 388. Alta concurrencia en el web service por consultar el predio rural
Fuente: Autor

```
INFO Max requests:        7000
INFO Concurrency level:   100
INFO Agent:               keepalive
INFO
INFO Completed requests:  7000
INFO Total errors:        0
INFO Total time:          5.012621672 s
INFO Requests per second: 1396
INFO Mean latency:        70.7 ms
INFO
INFO Percentage of the requests served within a certain time
INFO   50%      69 ms
INFO   90%      77 ms
INFO   95%      84 ms
INFO   99%      91 ms
INFO  100%      122 ms (longest request)
```

Figura 389. Resultados de alta concurrencia por consultas del predio rural
Fuente: Autor

Donde se puede observar que el web service es capaz de responder 7000 peticiones del predio rural en 5.01s.

- **Agua Potable:**

```
sistemas@ubuntuPC: ~
sistemas@ubuntuPC:~$ loadtest --keepalive -c 100 -n 7000 http://ws.chordeleg.gob.e
c:3001/agua/360
```

Figura 390. Alta concurrencia en el web service por consultar el agua potable
Fuente: Autor

```
INFO Max requests:        7000
INFO Concurrency level:   100
INFO Agent:               keepalive
INFO
INFO Completed requests:  7000
INFO Total errors:        0
INFO Total time:          5.188004763 s
INFO Requests per second: 1349
INFO Mean latency:        73.2 ms
INFO
INFO Percentage of the requests served within a certain time
INFO   50%      72 ms
INFO   90%      81 ms
INFO   95%      84 ms
INFO   99%      92 ms
INFO  100%      126 ms (longest request)
```

Figura 391. Resultados de alta concurrencia por consultas del agua potable
Fuente: Autor

Donde se puede observar que el web service es capaz de responder 7000 peticiones del agua potable en 5.19s.

- **Patente Municipal:**

```
sistemas@ubuntuPC: ~
sistemas@ubuntuPC:~$ loadtest --keepalive -c 100 -n 7000 http://ws.chordeleg.gob.e
c:3001/patente/402
```

Figura 392. Alta concurrencia en el web service por consultar la patente
Fuente: Autor

```
INFO Max requests:        7000
INFO Concurrency level:   100
INFO Agent:               keepalive
INFO
INFO Completed requests:  7000
INFO Total errors:        0
INFO Total time:          5.113857035 s
INFO Requests per second: 1369
INFO Mean latency:        72.1 ms
INFO
INFO Percentage of the requests served within a certain time
INFO   50%      70 ms
INFO   90%      80 ms
INFO   95%      82 ms
INFO   99%      97 ms
INFO  100%      130 ms (longest request)
```

Figura 393. Resultados de alta concurrencia por consultar la potable
Fuente: Autor

Donde se puede observar que el web service es capaz de responder 7000 peticiones de la patente municipal en 5.11s.

7.3.2.4 Escenario de concurrencia “Saturado” de clientes

Para el escenario de concurrencia “Saturado” se definió que todos los 8799 clientes proyectados realizaran peticiones al web service a través de la app de consulta, por lo tanto haciendo uso del módulo “loadtest” en la PC de desarrollo, se ejecutaron las pruebas de rendimiento por cada uno de los tributos principales del GAD Municipal de Chordeleg. A diferencia que la sección anterior únicamente se cambió el número de clientes de los comandos de prueba de “loadtest”, que en este caso son 8799 clientes, a continuación se presenta los comandos y resultados obtenidos:

- **Predio Urbano:**

```
sistemas@ubuntuPC: ~
sistemas@ubuntuPC:~$ loadtest --keepalive -c 100 -n 8799 http://ws.chordeleg.gob.e
c:3001/predioU/0111500204160090001
```

Figura 394. Alta concurrencia en el web service por consultar el predio urbano
Fuente: Autor

```
INFO Max requests:        8799
INFO Concurrency level:   100
INFO Agent:               keepalive
INFO
INFO Completed requests:  8799
INFO Total errors:        0
INFO Total time:          6.2526506820000005 s
INFO Requests per second: 1407
INFO Mean latency:        70.3 ms
INFO
INFO Percentage of the requests served within a certain time
INFO   50%      68 ms
INFO   90%      77 ms
INFO   95%      81 ms
INFO   99%      98 ms
INFO  100%      150 ms (longest request)
```

Figura 395. Resultados de alta concurrencia por consultas del predio urbano
Fuente: Autor

Donde se puede observar que el web service es capaz de responder 8799 peticiones del predio urbano en 6.25s.

- **Predio Rural:**

```
sistemas@ubuntuPC: ~
sistemas@ubuntuPC:~$ loadtest --keepalive -c 100 -n 8799 http://ws.chordeleg.gob.e
c:3001/predioR/011151510323025
```

Figura 396. Alta concurrencia en el web service por consultar el predio rural
Fuente: Autor

```
Max requests:        8799
Concurrency level:   100
Agent:               keepalive

Completed requests:  8799
Total errors:        0
Total time:          6.199428297000001 s
Requests per second: 1419
Mean latency:        69.7 ms

Percentage of the requests served within a certain time
  50%      69 ms
  90%      74 ms
  95%      76 ms
  99%      99 ms
 100%      125 ms (longest request)
```

Figura 397. Resultados de alta concurrencia por consultas del predio rural
Fuente: Autor

Donde se puede observar que el web service es capaz de responder 8799 peticiones del predio rural en 6.20s.

- **Agua Potable:**

Figura 398. Alta concurrencia en el web service por consultar el agua potable
Fuente: Autor

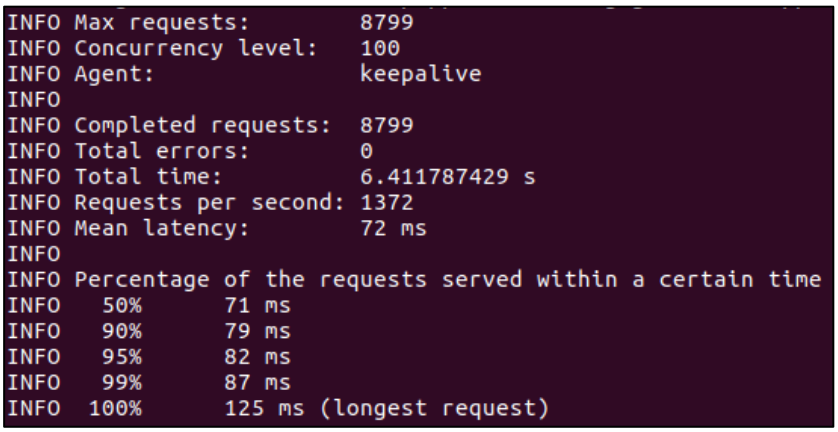


Figura 399. Resultados de alta concurrencia por consultas del agua potable
Fuente: Autor

Donde se puede observar que el web service es capaz de responder 8799 peticiones del agua potable en 6.41s.

- **Patente Municipal:**

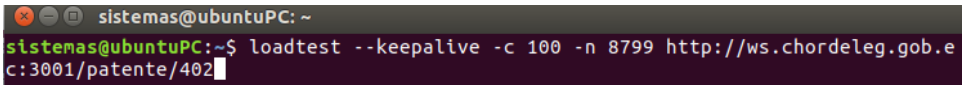


Figura 400. Alta concurrencia en el web service por consultar la patente
Fuente: Autor

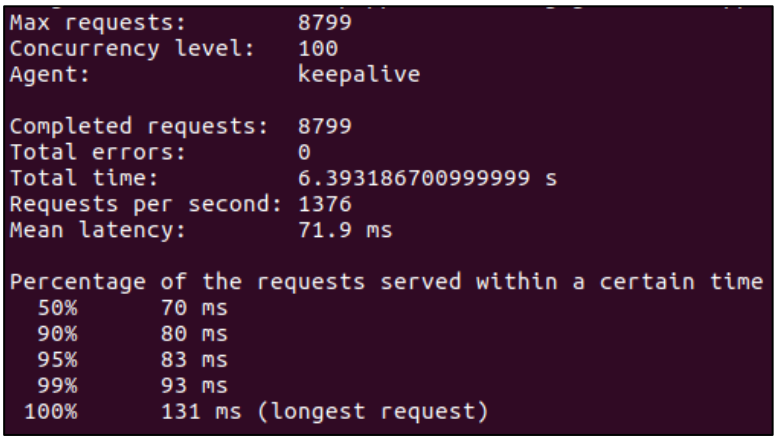


Figura 401. Resultados de alta concurrencia por consultar la potable
Fuente: Autor

Donde se puede observar que el web service es capaz de responder 8799 peticiones de la patente municipal en 6.39s.

7.3.2.5 Análisis general

Antes de proceder con un análisis general, se detalla en la siguiente tabla los resultados obtenidos a lo largo de la sección 7.2.2.

Tabla 56. Resumen de pruebas de campo obtenidas en el web service.

Consultas	Escenarios							
	Baja (1000 clientes)		Media (4500 clientes)		Alta (7000 clientes)		Saturado (8799 clientes)	
	Velo-cidad	Tiempo Total	Velo-cidad	Tiempo Total	Velo-cidad	Tiempo Total	Velo-cidad	Tiempo Total
Predio Urbano	1324res/ seg	0,76seg	1368res/ seg	3,29seg	1384res/ seg	5,06seg	1407res/ seg	6,25seg
Predio Rural	1393res/ seg	0,72seg	1444res/ seg	3,12seg	1396res/ seg	5,01seg	1419res/ seg	6,20seg
Agua Potable	1251res/ seg	0,80seg	1400res/ seg	3,21seg	1349res/ seg	5,19seg	1372res/ seg	6,41seg
Patente Municipal	1295res/ seg	0,77seg	1394res/ seg	3,23seg	1369res/ seg	5,11seg	1376res/ seg	6,39seg

Fuente: Autor

En el capítulo 2, se realizó un análisis de datos para calcular el ancho de banda necesario que permita responder a 8799 clientes en 3seg con una carga total de 2727.69KB proyectados en 5 años, y se determinó que el ancho de banda necesario es de 7.27Mbps, el cual se tiene disponible en la red del GAD Municipal de Chordeleg. Por lo tanto al compararlo con los valores de la tabla anterior que fueron obtenidos en cada prueba de campo del proyecto implementado en la red del GAD Municipal de Chordeleg, se observó que en un escenario Alta-Saturado los tiempos de respuesta están entre 5,01seg y 6.41seg, y lo esperado era que lleguen a un máximo de 3seg, esto se debe a que el servidor de producción apenas cuenta con 2GB de memoria ram de los 8Gb que soporta, a más de ello el servidor de producción de la base de datos es de las mismas características; sin embargo al recordar que en el capítulo 2 se había analizado que se tendrá una concurrencia del 50% de clientes ya que

se dispone del servicio de internet fijo únicamente en el sector urbano del cantón Chordeleg y parte del sector urbano de sus parroquias, por lo tanto los resultados obtenidos para un escenario Baja-Media que están entre 1000 y 4500 clientes, dan un tiempo de respuesta que oscila entre 0.72seg y 3.23seg, lo cual es sumamente considerable para que el usuario tenga una experiencia satisfactoria en la app “CHORDELEG móvil”.

7.4 Ancho de banda disponible en el web service

7.4.1 Instalación del script para realizar pruebas del ancho de banda en el web service

Para realizar las pruebas de rendimiento del ancho de banda del web service, se optó por instalar el script “Speedtest-Cli” que se encuentra disponible de manera gratuita en internet, el cual permitirá realizar las pruebas de conexión de internet del servidor de producción donde esta implementado el web service; a continuación se detalla paso a paso la instalación del script en el servidor de producción:

- Ingresar al terminal de la PC de desarrollo presionando “Ctrl + ALT + T” y entrar al servidor de producción de manera remota con el siguiente comando:

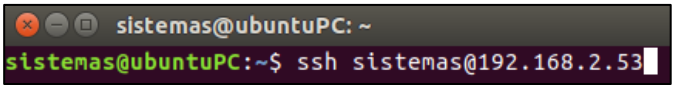


Figura 402. Ingresando al servidor de producción por ssh
Fuente: Autor

- Una vez que se esté en el servidor de producción se procede a instalar las dependencias del script “Speedtest-Cli”, para ello se debe realizar lo siguiente:

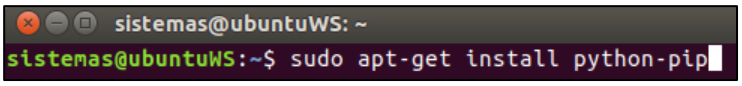


Figura 403. Instalación de “python-pip” en el servidor de producción
Fuente: Autor

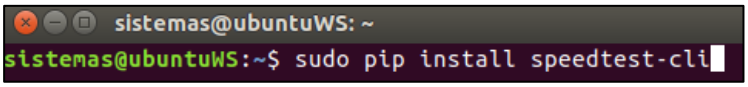


Figura 404. Instalación de “pip” de “speedtest-cli” en el servidor de producción
Fuente: Autor

- Luego de instalar las dependencias correspondientes, se procede a instalar "Speedtest-Cli" realizando lo siguiente:

Figura 405. Descargar "speedtest_cli.py" en el servidor de producción
Fuente: Autor

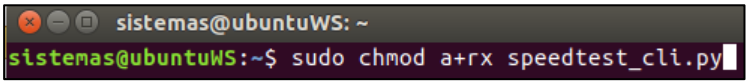


Figura 406. Ejecutar los permisos para "speedtest_cli-py"
Fuente: Autor

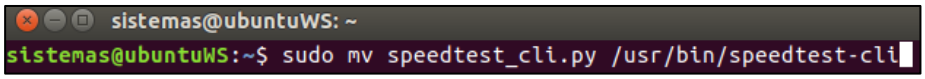


Figura 407. Mover el "speedtest" a la ruta "/usr/bin"
Fuente: Autor

- Finalmente para iniciar un test simplemente se ejecuta el siguiente comando:

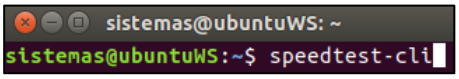


Figura 408. Comando para realizar un test de internet en el servidor de producción
Fuente: Autor

7.4.2 Consumo de Ancho de banda del web service

En la sección 6.3.2.3 se configuró el router principal del GAD Municipal de Chordeleg con los límites de ancho de banda de 5.08Mbps de bajada y 7.27Mbps de subida, para que garanticen la calidad de servicio del web service, razón por la cual se realizará las pruebas correspondientes para observar el comportamiento en distintos horarios de trabajo del GAD Municipal de Chordeleg, tales como el horario de mañana, horario de la tarde y la hora pico.

7.4.2.1 Horario de la mañana

En vista que es un horario de la mañana, la prueba se fija entre las 9::30am y 10:30am, a continuación se detalla las siguientes actividades:

- **Configurar la herramienta para realizar el test de conexión de internet,** abrir la terminal del servidor de producción presionando “Ctrl + ALT + T” y cuando sea necesario se ejecuta el test con el comando de la figura 408.

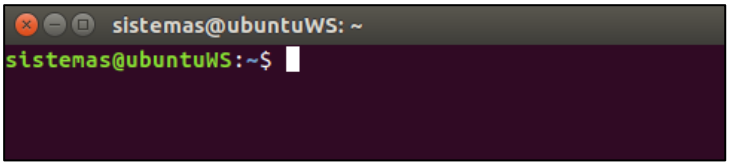


Figura 409. Ventana de la terminal en el servidor de producción
Fuente: Autor

- **Configurar la herramienta para observar el ancho de banda consumido,** ingresar al router principal mikrotik mediante la herramienta “WinBox”, dirigirse a la sección de “Simple Queue”, se ubica la regla creada en el capítulo 6, para observar el tráfico del servidor de producción definida en base a la dirección ip 192.168.2.53, se da doble clic sobre ella y se dirige a la pestaña “Traffic”.

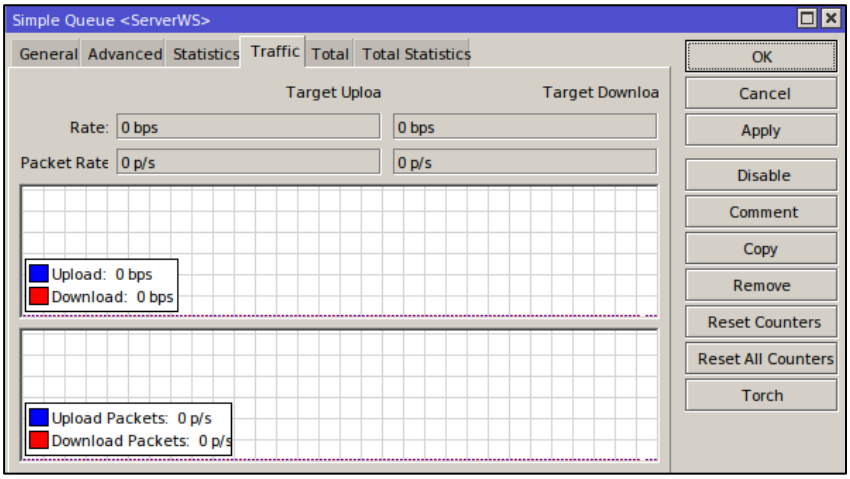


Figura 410. Regla para observar el ancho de banda consumido en el web service
Fuente: Autor

- **Realizar la prueba de conexión,** a las 10:14am se ejecutó el siguiente comando.

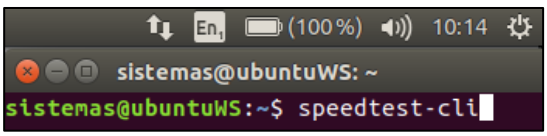


Figura 411. Pruebas con “speedtest” en horario de la mañana
Fuente: Autor

- **Resultados obtenidos con Speedtest y RouterOS,** en las siguientes figuras se puede observar al ancho de banda de subida y bajada disponible a las 10:14am:

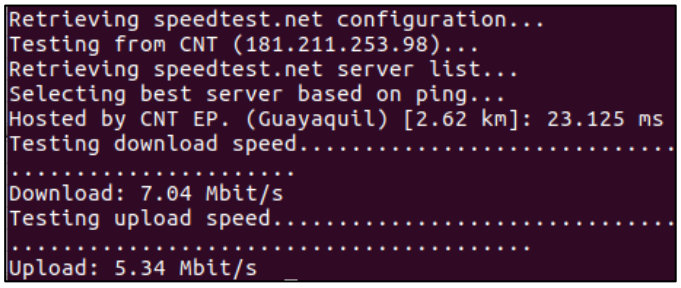


Figura 412. Resultados de "speedtest" en horario de la mañana
Fuente: Autor

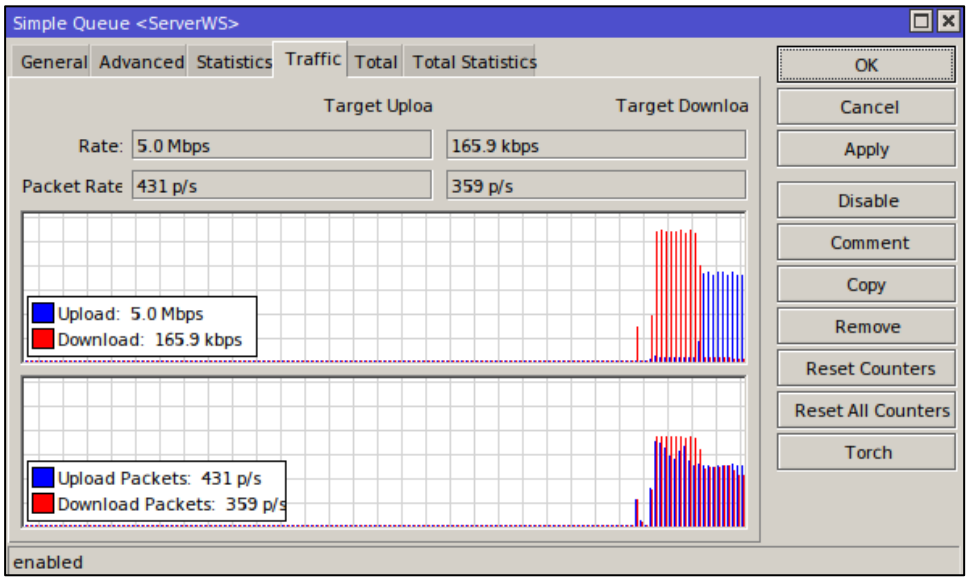


Figura 413. Gráfica del resultado de "speedtest" en horario de la mañana
Fuente: Autor

7.4.2.2 Horario de la tarde

Para el horario de la tarde, la prueba se fija entre las 15:00pm y 16:00pm, a continuación se detalla las siguientes actividades:

- **Configurar la herramienta para realizar el test de conexión de internet,** ya se encuentra lista desde la sección anterior, se puede dirigir a la figura 409.
- **Configurar la herramienta para observar el ancho de banda consumido,** ya se encuentra lista desde la sección anterior, se puede dirigir a la figura 410.
- **Realizar la prueba de conexión,** a las 15:18pm se ejecutó el siguiente comando.

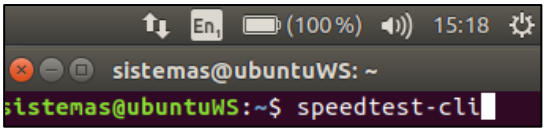


Figura 414. Pruebas con "speedtest" en horario de la tarde
Fuente: Autor

- **Resultados obtenidos con Speedtest y RouterOS,** en las siguientes figuras se apreciará al ancho de banda de subida y bajada disponible a las 15:18pm:

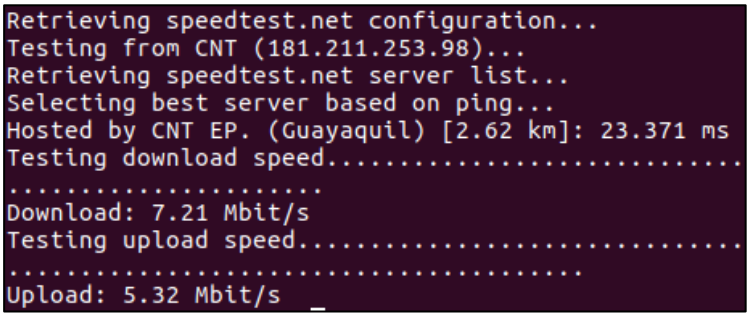


Figura 415. Resultados de "speedtest" en horario de la tarde
Fuente: Autor

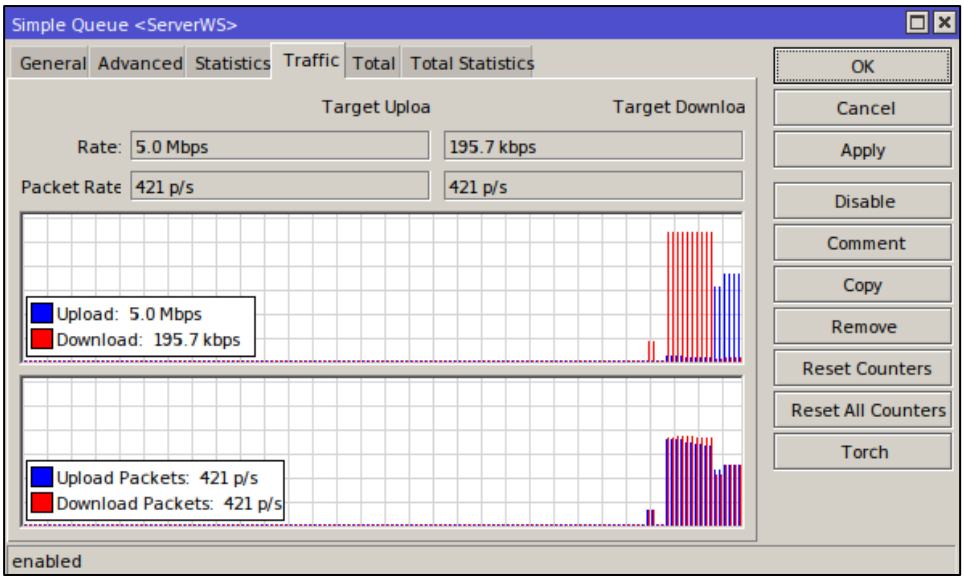


Figura 416. Gráfica del resultado de "speedtest" en horario de la tarde
Fuente: Autor

7.4.2.3 Hora pico

La hora pico de acuerdo a la tabla 31, está entre las 12:00 y 13:00pm, a continuación se detalla las siguientes actividades:

- **Configurar la herramienta para realizar el test de conexión de internet,** ya se encuentra lista desde la sección 7.4.2.1, se puede dirigir a la figura 409.
- **Configurar la herramienta para observar el ancho de banda consumido,** ya se encuentra lista desde la sección 7.4.2.1, se puede dirigir a la figura 410.
- **Realizar la prueba de conexión,** a las 12:31pm se ejecutó el siguiente comando.

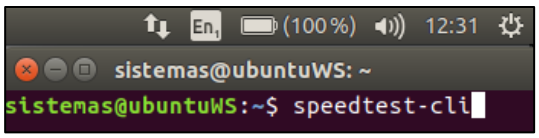


Figura 417. Pruebas con "speedtest" en la hora pico
Fuente: Autor

- **Resultados obtenidos con Speedtest y RouterOS,** en las siguientes figuras se puede observar el ancho de banda de subida y bajada disponible a las 15:18pm:

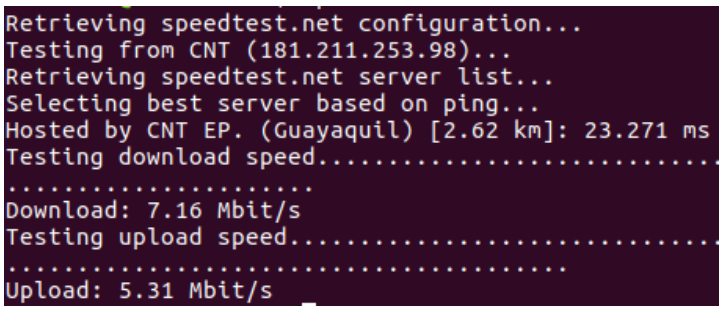


Figura 418. Resultados de "speedtest" en la hora pico
Fuente: Autor

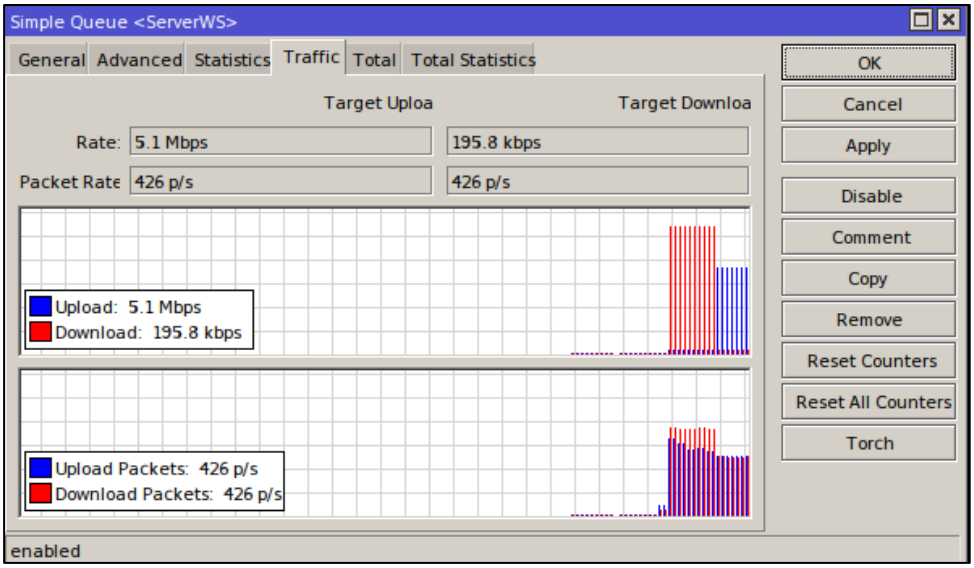


Figura 419. Gráfica del resultado de "speedtest" en la hora pico
Fuente: Autor

7.4.2.4 Análisis general

Antes de proceder con un análisis general, se detallará en la siguiente tabla los resultados obtenidos a lo largo de la sección 7.4.2.

Tabla 57. Resumen de pruebas de campo del ancho de banda obtenidas en el web service.

Horario	Ancho de Banda Consumido	
	Bajada	Subida
Mañana	5,34Mbps	7,04Mbps
Tarde	5,32Mbps	7,21Mbps
Hora Pico	5,31Mbps	7,16Mbps

Fuente: Autor

En la tabla anterior se puede notar claramente que para los tres casos, el ancho de banda es el esperado, lo cual quiere decir que las reglas configuradas en el router mikrotik para garantizar la calidad de servicio, se cumplen sin ningún inconveniente. Cabe indicar que este ancho de banda asignado al web service, se reasigna de manera automática a los otros servicios del GAD Municipal de Chordeleg cuando está en desuso.

7.5 Operación de la App "CHORDELEG móvil" en las principales operadoras de telefonía móvil de nuestro país.

Actualmente en nuestro país, las principales operadoras que brindan el servicio de telefonía móvil son: Movistar, Claro y CNT; antes de entrar en detalle con cada una de las operadoras mencionadas, se presenta los datos consumidos por consultas desde la app "CHORDELEG móvil".

7.5.1 Consumo de datos de la app "CHORDELEG móvil"

Para generar y observar el tráfico generado desde los datos móviles, se realizaron los siguientes pasos:

- **Activar datos en el dispositivo móvil,** los pasos para activar dependerá del dispositivo que se esté usando, en esta caso se está trabajando con un Samsung

Note 5; para ello se desliza el menú superior del dispositivo y se presiona en "Móvil y datos".

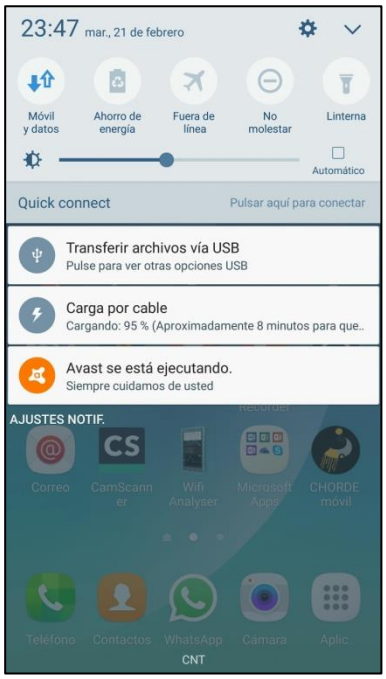


Figura 420. Activando datos en el dispositivo móvil Samsung Note 5
Fuente: Autor

- **Realizar consultas,** se generara dos consultas aleatorias de la siguiente manera:

Figura 421. Consulta de patente con datos móviles
Fuente: Autor

Figura 422. Reporte de consulta de patente con datos móviles
Fuente: Autor

Figura 423. Consulta del agua potable con datos móviles
Fuente: Autor

Figura 424. Reporte de consulta del agua potable con datos móviles
Fuente: Autor

- **Consultar los datos consumidos,** para ello dentro del dispositivo móvil, dirigirse a "Ajustes", luego a "Usos de datos" y se presiona sobre la aplicación respectiva para ver el detalle de los datos consumidos.

Figura 425. Consumo de datos de la app "CHORDELEG móvil"
Fuente: Autor

Finalmente se puede observar en la figura anterior, que el consumo de datos por las dos consultas realizadas es de 871Bytes; con este valor obtenido se analizará los costos que se generarían en las diferentes operadoras.

7.5.2 Datos de la app con Movistar

7.5.2.1 Paquete de datos

Como referencia se toma un paquete de datos de 1000MB, el cual se obtiene de la página oficial de Movistar (http://www.movistar.com.ec).

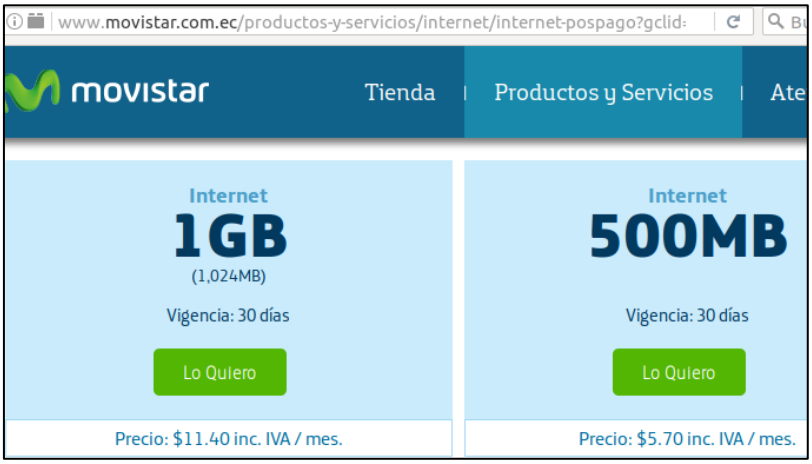


Figura 426. Paquete de datos de 1000MB en Movistar
Fuente: http://www.movistar.com.ec/productos-y-servicios/internet

Donde se puede observar que el costo de 1000MB es de $11.40 incluido impuestos, por lo tanto realizando el cálculo correspondiente se tiene que el costo por cada MB es de (1MB * $11.40)/1000MB = $0.0114.

7.5.2.2 Costo de datos de la app

En la sección 7.5.1 se obtuvo un consumo de datos de 871Bytes por dos consultas realizadas con la app “CHORDELEG móvil”, por lo tanto realizando el cálculo correspondiente se tiene que Movistar cobraría (0.000871MB * $0.0114)/1MB = $0.0000099294, por el servicio prestado.

7.5.3 Datos de la app con Claro

7.5.3.1 Paquete de datos

De igual manera como referencia se toma un paquete de datos de 1000MB, el cual se obtuvo de la página oficial de Claro (http://www.claro.com.ec).

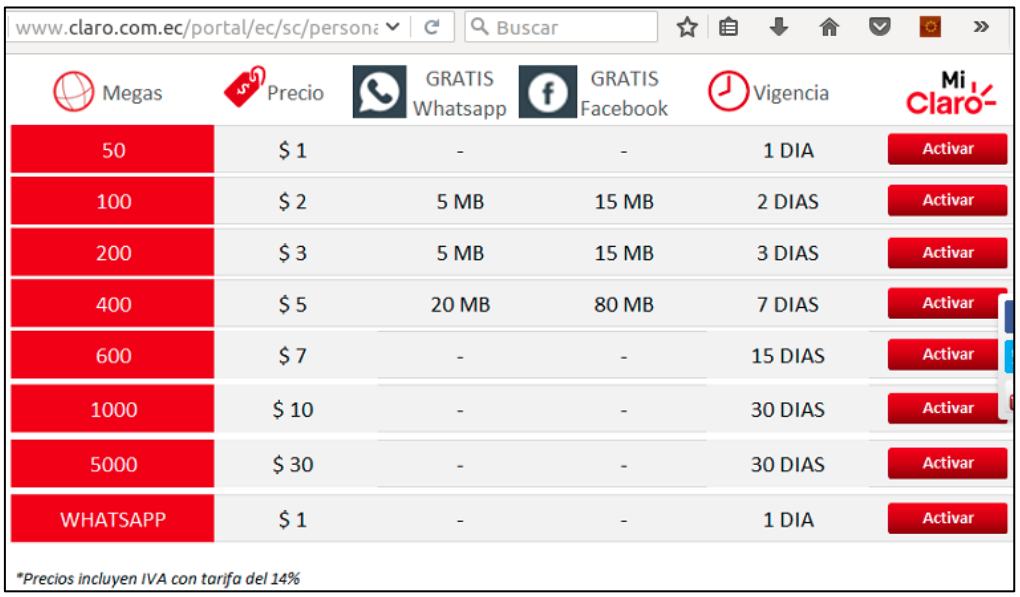


Figura 427. Paquete de datos de 1000MB en Claro
Fuente: http://www.claro.com.ec/portal/ec/sc/personas/movil

Donde se puede observar que el costo de 1000MB es de $10.00 incluido impuestos, por lo tanto realizando el cálculo correspondiente se tiene que el costo por 1MB es de (1MB * $10.00)/1000MB = $0.01.

7.5.3.2 Costo de datos de la app

De igual forma tomando como referencia la sección 7.5.1, se obtiene un consumo de datos de 871Bytes por dos consultas realizadas con la app “CHORDELEG móvil”, por lo tanto realizando el cálculo correspondiente se tiene que Claro cobraría (0.000871MB * $0.01)/1MB = $0.00000871, por el servicio prestado.

7.5.4 Datos de la app con CNT

7.5.4.1 Paquete de datos

Al igual que en las dos operadoras anteriores se toma como referencia un paquete de datos de 1000MB, el cual se obtuvo de la página oficial de CNT (http://www.cnt.gob.ec).

Figura 428. Paquete de datos de 1000MB en CNT
Fuente: https://www.cnt.gob.ec/movil/plan/

Donde se puede observar que el costo de 1000MB es de $11.46 incluido impuestos, por lo tanto realizando el cálculo correspondiente se tiene que el costo por 1MB es de (1MB * $11.46)/1000MB = $0.01146.

7.5.4.2 Costo de datos de la app

Al igual que los dos casos anteriores, tomando como referencia la sección 7.5.1, se obtiene un consumo de datos de 871Bytes por dos consultas realizadas con la app "CHORDELEG móvil", por lo tanto realizando el cálculo correspondiente se tiene que CNT cobraría (0.000871MB * $0.01146)/1MB = $0.0000099817, por el servicio prestado.

7.5.5 Análisis general

Antes de proceder con el análisis, se detalla en la siguiente tabla los valores obtenidos en la sección 7.5.

Tabla 58. Costos de datos generados por la app "CHORDELEG móvil".

Operadoras	Costo por 1MB (Paquete de datos de 1000MB)	Datos consumidos por dos consultas de la App	Total Consumido
Movistar	$ 0,0114	871Bytes	$ 0,0000099294
Claro	$ 0,01	871Bytes	$ 0,00000871
CNT	$ 0,01146	871Bytes	$ 0,0000099817

Fuente: Autor

Observando la tabla anterior, se puede ver que el total consumido por el servicio no llega ni siquiera 1 centavo de dólar, en cualquiera de las 3 operadoras; lo cual es evidente ya que el consumo de datos de la app “CHORDELEG móvil” es mínimo, y en este caso en específico apenas consume 871Bytes por dos consultas.

Así mismo comparando el costo por mega de cada operadora, se puede notar que no existe mayor diferencia entre sí y se podría optar por hacer uso de cualquiera de las operadoras, sin embargo en el cantón Chordeleg únicamente dan servicio las operadoras de Movistar y CNT.

CAPITULO VIII

8 Conclusiones y Recomendaciones

8.1 Conclusiones

- Hoy en día existen una gran variedad de herramientas libres en internet, para ser descargadas de manera gratuita; por tal motivo este proyecto fue desarrollado haciendo uso de software libre en su totalidad; de lo cual se puede decir que el hecho de utilizar software libre y gratuito no significa que es un producto de mala calidad, todo lo contrario, existen grandes comunidades que dan soporte al software libre que inclusive se han desarrollado herramientas muy superiores a las de pago.
- De la experiencia que se obtuvo en el desarrollo del proyecto, se determina que los equipos Mikrotik a más de ser mucho más económicos que los equipos CISCO, tienen todo lo necesario para ser implementados y garantizar el funcionamiento efectivo de cualquier tipo de red; como se pudo observar en el transcurso del proyecto, disponen de 3 tipos de administración (web, software de escritorio y por consola), control de ancho de banda, calidad de servicio, firewall, proxy, entre otros.
- El web service actualmente se encuentra limitado por las capacidades de hardware del servidor de producción facilitado por el GAD Municipal de Chordeleg, sin embargo en las pruebas realizadas mediante el script “loadtest”, se observó que soporta la cantidad de usuarios proyectados en un tiempo de respuesta satisfactorio; no obstante se puede mejorar el rendimiento actualizando su hardware, tales como el aumento de la memoria ram.
- El web service desarrollado, además de ser consumido por la app “CHORDELEG móvil”, se puede desarrollar a futuro otras aplicaciones como páginas web, software de escritorio, entre otros, y consumir el mismo servicio sin la necesidad de realizar cambios en su código fuente; debido a que el servicio es desarrollado

independientemente de la aplicación que utiliza el cliente. El cliente para su desarrollo únicamente debe tomar en cuenta que las peticiones se realizan con el comando GET del protocolo HTTP, donde se debe incluir la ruta del servicio a consultar; posteriormente las respuestas recibidas deben ser interpretadas bajo el formato JSON.

- El análisis de tráfico realizado en la hora pico, presentó resultados positivos, donde la priorización y el ancho de banda fijado para el web service garantizó completamente lo esperado; si bien es cierto que se establecieron límites del ancho de banda para el web service, no quiere decir que si el ancho de banda disponible está en desuso se desaprovecha, todo lo contrario, en el momento que el web service no reciba peticiones, el ancho de banda es reasignado automáticamente a la red LAN del GAD Municipal de Chordeleg; la diferencia está en que el servicio prioriza ese ancho de banda asignado cuando lo necesita.
- No siempre la contratación de mayor ancho de banda es la solución para mejorar el rendimiento de una red, en el presente proyecto bastó con implementar calidad de servicio y priorizar el tráfico acorde a las necesidades del GAD Municipal de Chordeleg para mejorar cada uno de los servicios que dispone la institución; tales como sistemas de recaudación tributaria, correo institucional, navegación web, web service, descargas, entro otros.
- El consumo de datos de la app desarrollada “CHORDELEG móvil” es mínimo, tal es el caso que al realizar el análisis de costos en cualquiera de las principales operadoras como Movistar, Claro y CNT, no llega a costar ni un centavo de dólar por cada consulta realizada.

8.2 Recomendaciones

- Se recomienda al GAD Municipal de Chordeleg, aumentar la memoria ram de los servidores de producción del web service y de la base de datos respectivamente, al

menos duplicar su memoria que actualmente es de 2GB en cada servidor, esto con el fin de mejorar el rendimiento y la velocidad de procesamiento del hardware, lo cual se reflejará disminuyendo el tiempo de respuesta del servicio y aumentando el número de peticiones simultáneas.

- Teniendo como guía la implementación de calidad de servicio (QoS) en el router principal de la red LAN del GAD Municipal de Chordeleg, se recomienda al departamento de sistemas del GAD Municipal de Chordeleg, aplicar el mismo concepto en la red de internet del parque central del cantón Chordeleg, la cual opera dentro del mismo router principal, pero en un horario diferente; con ello se mejoraría notablemente el rendimiento de la red, obteniendo como resultado una mejorar experiencia en la navegación web para el cliente.
- Si bien es cierto, la app "CHORDELEG móvil" se desarrolló para consultar los principales tributos municipales como el predio urbano, predio rural, agua potable y patente municipal, pues también se detalló en el desarrollo del proyecto, desde la instalación del sistema operativo de la pc de desarrollo hasta el código fuente de la aplicación, con el fin de que se continúe incluyendo los demás tributos municipales del GAD de Chordeleg. Razón por la cual se recomienda al departamento de sistemas revisar detenidamente la información de interés, para continuar con el desarrollo de la app que permita incluir los demás tributos municipales.
- Para que la app "CHORDELEG móvil" sea difundida y tenga una gran acogida, se recomienda al departamento de comunicaciones del GAD Municipal de Chordeleg incluir un enlace en la página web oficial del Municipio con la app que se encuentra en "Google Play", y difundirla a través de las redes sociales oficiales del Municipio de Chordeleg.

BIBLIOGRAFÍA

Abernethy, M. (14 de Junio de 2011). ¿Simplemente qué es Node.js?. Obtenido de https://www.ibm.com/developerworks/ssa/opensource/library/os-nodejs/

Ardións, A. (01 de Febrero de 2016). Android Studio vs Eclipse. Obtenido de http://androidstudiofaqs.com/conceptos/android-studio-vs-eclipse

CCM. (s.f.). El protocolo HTTP. Obtenido de http://es.ccm.net/contents/264-el-protocolo-http

Culturacion. (s.f.). ¿Qué es y para qué sirve un web service?. Obtenido de http://culturacion.com/que-es-y-para-que-sirve-un-web-service/

ECURED. (s.f.). Eclipse, entorno de desarrollo integrado. Obtenido de https://www.ecured.cu/Eclipse,_entorno_de_desarrollo_integrado

GAD Municipal de Chordeleg. (Agosto del 2016). Obtenido de http://chordeleg.gob.ec/datos-generales/

Gallardo, D. (26 de Noviembre de 2012). Iniciándose en la plataforma Eclipse. Obtenido de https://www.ibm.com/developerworks/ssa/library/os-ecov/

Gerometta, O. (30 de Agosto de 2010). Modelos de implementación de QoS. Obtenido de http://librosnetworking.blogspot.com/2010/08/modelos-de-implementacion-de-qos.html

Gimenez, M. A. (07 de Julio de 2012). Información Básica sobre netbeans. Obtenido de http://netbeansaccesible.blogspot.com/

Global Mentoring. (27 de Enero de 2012). ¿Qué es un IDE?. Obtenido de http://globalmentoring.com.mx/cursos-java/java-fundamentos/que-es-un-ide/

Gomez, I. A. (10 de Septiembre de 2015). Estructura de una aplicación móvil. Obtenido de https://prezi.com/dimql0fshegp/estructura-de-una-aplicacion-movil/

INEC. (s.f.). Resultados Censo de Población. Obtenido de http://www.inec.gob.ec/cpv/

Luiggi, S. M. (24 de Julio de 2014). 12 Ventajas de la Programación PHP que Debes Saber. Obtenido de http://www.staffcreativa.pe/blog/ventajas-programacion-php/

Martin de Pozuelo, A. (07 de Julio de 2013). Node.js: qué es y cuándo utilizarlo. Obtenido de http://moreovercoder.blogspot.com/2013/07/nodejs-que-es-y-cuando-utilizarlo.html

Martinez, J. (s.f.). CALIDAD DE SERVICIO (QoS). Obtenido de http://cic.puj.edu.co/wiki/lib/exe/fetch.php?media=materias:daysenr:daysenr_-_calidad_de_servicio_qos_.pdf

Martinez, K. (04 de Abril de 2011). TODO SOBRE PHP. Obtenido de http://klarimartinezbenjumea.blogspot.com/2011/04/ventajas-y-desventajas.html

Mejia, D. (26 de Octubre de 2015). REST VS SOAP. Obtenido de http://restysoap.blogspot.com/2015/10/normal-0-21-false-false-false-es-pe-x.html

Molina, M. R. (s.f.). Introducción a la calidad de servicios (QoS). Obtenido de https://sites.google.com/site/redesconvergentessitioweb/unidad-ii-calidad-de-servicio-qos/1---introduccion-a-la-calidad-de-servicios-qos

MUNDOMANUALES. (23 de Enero de 2011). Que es Android: Características y Aplicaciones. Obtenido de http://www.mundomanuales.com/telefonia/telefonos-moviles/que-es-android-caracteristicas-y-aplicaciones-4110.html

Pao. (s.f.). CONOCE LAS VENTAJAS Y DESVENTAJAS DE JAVASCRIPT. Obtenido de https://www.nextu.com/blog/conoce-las-ventajas-y-desventajas-de-javascript/

Porras, M. (17 de Noviembre de 2016). 15 ventajas de las aplicaciones Android para tu negocio. Obtenido de http://gananci.com/ventajas-aplicaciones-android/

QODE. (28 de Noviembre de 2013). Web Services – REST vs SOAP. Obtenido de http://qode.pro/blog/web-services-rest-vs-soap/

Revelo, J. (07 de Agosto de 2014). Aprendiendo Sobre La Arquitectura De Android. Obtenido de http://www.hermosaprogramacion.com/2014/08/aprendiendo-la-arquitectura-de-android/

Rouse, M. (Enero de 2015). Base de datos relacional. Obtenido de http://searchdatacenter.techtarget.com/es/definicion/Base-de-datos-relacional

Tomás, J. (2015). Ficheros y carpetas de un proyecto Android. Obtenido de http://www.androidcurso.com/index.php/tutoriales-android/31-unidad-1-vision-general-y-entorno-de-desarrollo/148-elementos-de-un-proyecto-android

WhiZ. (29 de Septiembre de 2014). Introducción a Python. Obtenido de https://underc0de.org/foro/python/introduccion-a-python/

Wikipedia. (07 de diciembre de 2016). Netbeans. Obtenido de https://es.wikipedia.org/wiki/NetBeans

Wikipedia. (17 de Febrero de 2017). Aplicación móvil. Obtenido de https://es.wikipedia.org/wiki/Aplicaci%C3%B3n_m%C3%B3vil

ZEOKAT. (21 de Noviembre de 2013). Protocolo HTTP para desarrolladores (parte II). Obtenido de http://www.vozidea.com/protocolo-http-desarrolladores-dos

ANEXOS

Anexo 1: Código de consultas SQL de los principales tributos municipales

- Consulta del predio urbano

```
(SELECT IF(count(*)>0, b.cedula,"N.E.") as flag_ced_period, concat(b.apellidoPropietario, " ", b.nombrePropietario) as
prop_fecha, 0 as impPredial, 0 as solarNoEdificado, 0 as bomberos, 0 as computacion, 0 as exoneracion, 0 as interes
            FROM scum_parcela a, propietario b
            WHERE (a.CodProv='01' AND a.CodCant='11' AND a.CodParroq='50' AND a.CodZona='02' AND a.CodSect='03'
AND a.CodPolg='05' AND a.CodParc='012' AND a.CodPropHoriz='001') AND (b.codPropietario=a.codPropietario))
            UNION
            (SELECT (cast(a.Ano as char)) as flag_ced_period, (DATE_FORMAT(CONCAT(SUBSTRING(a.CodAvaluo,1,2),
"-", SUBSTRING(a.CodAvaluo,3,2), "-", SUBSTRING(a.CodAvaluo,5,2)), "%d/%m/%Y")) as prop_fecha, a.impPredial,
a.solarNoEdificado, a.bomberos, a.computacion, a.exoneracion, sum(b.porcentajeinteres) as interes
             FROM scum_liquidacion_avaluo a, intereses b
             WHERE (a.CodProv='01' AND a.CodCant='11' AND a.CodParroq='50' AND a.CodZona='02' AND a.CodSect='03'
AND a.CodPolg='05' AND a.CodParc='012' AND a.CodPropHoriz='001' AND a.pagado=0)
                          AND
                    (b.fecha BETWEEN
                                                                    IF(date_format(NOW(), "%y-%m") >=
date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1 MONTH,
"%y-%m"),

              date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1
MONTH, "%Y%m"),
                                                                        "100001")
                                AND
                                                                    IF(date_format(NOW(), "%y-%m") >=
date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1 MONTH,
"%y-%m"),
                                                                        date_format(NOW( ), "%Y%m"),
                                                                            "100001")
                    )
            GROUP BY a.Ano)
```

- Consula del predio rural

```
(SELECT IF(count(*)>0, b.cedula,"N.E.") as flag_ced_period, concat(b.apellidoPropietario, " ", b.nombrePropietario) as
prop_fecha, 0 as impPredial, 0 as bomberos, 0 as computacion, 0 as exoneracion, 0 as interes
            FROM scr_parcela a, propietario b
            WHERE (a.CodProv='01' AND a.CodCant='11' AND a.CodParroq='51' AND a.CodZona='51' AND a.CodSect='03'
AND a.CodPolg='23' AND a.CodParc='025') AND (b.codPropietario=a.codPropietario))
            UNION
            (SELECT (cast(a.Ano as char)) as flag_ced_period, (DATE_FORMAT(CONCAT(SUBSTRING(a.CodAvaluo,1,2),
"-", SUBSTRING(a.CodAvaluo,3,2), "-", SUBSTRING(a.CodAvaluo,5,2)), "%d/%m/%Y")) as prop_fecha, a.impPredial,
a.bomberos, a.computacion, a.exoneracion, sum(b.porcentajeinteres) as interes
             FROM scr_liquidacion_avaluo a, intereses b
             WHERE (a.CodProv='01' AND a.CodCant='11' AND a.CodParroq='51' AND a.CodZona='51' AND a.CodSect='03'
AND a.CodPolg='23' AND a.CodParc='025' AND a.pagado=0)
                          AND
                    (b.fecha BETWEEN
                                                                    IF(date_format(NOW(), "%y-%m") >=
date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1 MONTH,
"%y-%m"),

              date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1
MONTH, "%Y%m"),
                                                                        "100001")
                                AND
                                                                    IF(date_format(NOW(), "%y-%m") >=
date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1 MONTH,
"%y-%m"),
                                                                        date_format(NOW( ), "%Y%m"),
                                                                            "100001")
                    )
            GROUP BY a.Ano)
```

- Consulta del agua potable

```
(SELECT IF(count(*)>0, b.cedula,'N.E.') as flag_ced_period, concat(b.apellidoPropietario, " ", b.nombrePropietario) as
prop_fecha, 0 as valor_principal, 0 as alcantarillado, 0 as basura, 0 as total_tarifa, 0 as interes
            FROM agua_medidor_abonado a, propietario b
            WHERE a.codmedidor='1405' AND b.codPropietario=a.codPropietario)
            UNION
            (SELECT (a.codemision) as flag_ced_period, (DATE_FORMAT(a.fecha_factura, "%d/%m/%Y")) as prop_fecha,
a.valor_principal, a.alcantarillado, a.basura, a.total_tarifa, sum(b.porcentajeinteres) as interes
             FROM agua_factura a, intereses b
             WHERE (a.codMedidor='1405' AND a.pagado=0 AND estado=1 AND id_clase_rubro=1)
                                                                AND
                (b.fecha BETWEEN
                                                                    IF( date_format(NOW(), "%Y-%m-%d") >=
DATE_ADD(a.fecha_factura, INTERVAL 1 MONTH),
                                                    date_format(DATE_ADD(a.fecha_factura, INTERVAL 1 MONTH), "%Y%m"),
                                                    100001)
                                                                    AND
                                                                    IF( date_format(NOW(), "%Y-%m-%d") >=
DATE_ADD(a.fecha_factura, INTERVAL 1 MONTH),
                                                    IF( DAY(NOW( )) >= DAY(DATE_ADD(a.fecha_factura, INTERVAL
period_diff(date_format(NOW(), "%Y%m"), date_format(a.fecha_factura, "%Y%m")) MONTH)),
                                                                date_format(NOW( ), "%Y%m"),
                                                                date_format(DATE_SUB(NOW( ), INTERVAL 1 MONTH),
"%Y%m")),
                                                    100001)
                )
            GROUP BY a.codemision)
```

- Consulta de la patente municipal

```
(SELECT (count(*)) as anio, concat(b.Cedula, " ", b.apellidoPropietario, " ", b.nombrePropietario) as fecha, (0) as
valorImpuesto, (0) as impuestoCapital, (0) as servAdmin, (0) as otro, (0) as valorEspe, (0) as interes
            FROM patentes_patente a, propietario b
            WHERE a.numPatente='100' AND b.codPropietario=a.codPropietario)
            UNION
            (SELECT (a.ano) as anio, (DATE_FORMAT(a.fecha_emision, "%d/%m/%Y")) as fecha, a.valor_impuesto,
a.impuesto_capital, a.serv_administrativos, a.otro, a.Valor_Especie, sum(b.porcentajeinteres) as interes
             FROM patentes_emision_patente a, intereses b
             WHERE (a.numpatente='100' AND a.pagado=0)
                        AND
                (b.fecha BETWEEN
                    IF( date_format(NOW(), "%Y-%m-%d") >= DATE_ADD(a.fecha_emision, INTERVAL 1 MONTH),
                                                    date_format(DATE_ADD(a.fecha_emision, INTERVAL 1 MONTH), "%Y%m"),
                                                    100001)
                                AND
                                        IF( date_format(NOW(), "%Y-%m-%d") >= DATE_ADD(a.fecha_emision, INTERVAL 1
MONTH),
                                                    IF( DAY(NOW( )) >= DAY(DATE_ADD(a.fecha_emision, INTERVAL
period_diff(date_format(NOW(), "%Y%m"), date_format(a.fecha_emision, "%Y%m")) MONTH)),
                                                                date_format(NOW( ), "%Y%m"),
                                                                date_format(DATE_SUB(NOW( ), INTERVAL 1 MONTH),
"%Y%m")),
                                                    100001)
                )
            GROUP BY a.ano)
```

Anexo 2: Código fuente del web service

- Archivo “package.json”

```
{
        "name": "GADMCH_WS",
        "main": "app.js",
        "dependencies": {
                "mysql":"*",
                "restify":"*"
        }
}
```

- Archivo “config.json”

```
{
```

```
        "server": {
                "nombre": "WS_GADMCH",
                "ip": "192.168.2.53",
                "puerto": "3001"
        },
        "db_simm": {
                "driver": "mysql",
                "ip": "192.168.2.50",
                "usuario": "root",
                "contraseña": "*********",
                "db": "simm"
        }
}
```

- Archivo app.js

```
//*****************************************************
//ICLUIR DEPENDENCIAS*********************************
//*****************************************************
var restify = require('restify');
var mysql = require('mysql');
var db_config = require('./config.json')["db_simm"];
var server_config = require('./config.json')["server"];
//*****************************************************
//CREAR, CONFIGURAR Y CONECTARSE A LA BASE DE DATOS****
//*****************************************************
var connection =  mysql.createConnection({
                host : db_config.ip,
                user : db_config.usuario,
                password : db_config.contraseña,
                database: db_config.db
              });
//*****************************************************
//CREAR, CONFIGURAR E INCIAR EL SERVIDOR REST**********
//*****************************************************
var server = restify.createServer({
     name : server_config.nombre
   });
server.use(restify.queryParser());
server.use(restify.bodyParser());
server.use(restify.CORS());
server.listen(server_config.puerto, server_config.ip, function(){
   console.log('%s listening at %s ', server.name , server.url);
});
//*****************************************************
//CREAR LAS RUTAS*************************************
//*****************************************************
//Datos de PredioUrbano
var PATH = '/predioU'
server.get({path : PATH + '/:codPredioU', version : '0.0.1'}, consultaPredioU);
//Datos de PredioRural
var PATH = '/predioR'
server.get({path : PATH + '/:codPredioR', version : '0.0.1'}, consultaPredioR);
//Datos de Agua
var PATH = '/agua'
server.get({path : PATH + '/:codAgua', version : '0.0.1'}, consultaAgua);
//Datos de Patente
var PATH = '/patente'
server.get({path : PATH + '/:codPatente', version : '0.0.1'}, consultaPatente);
//*****************************************************
//FUNCIONES*******************************************
//*****************************************************
//Función para consultar el predio urbano
function consultaPredioU(req, res, next){
  var codProv = req.params.codPredioU.substring(0, 2)
  var codCant = req.params.codPredioU.substring(2, 4)
  var codParroq = req.params.codPredioU.substring(4, 6)
  var codZona = req.params.codPredioU.substring(6, 8)
  var codSect = req.params.codPredioU.substring(8, 10)
  var codPolg = req.params.codPredioU.substring(10, 12)
  var codParc = req.params.codPredioU.substring(12, 15)
  var codPropHoriz = req.params.codPredioU.substring(15, 18)
```

```
 connection.query('(SELECT IF(count(*)>0, b.cedula,"N.E.") as flag_ced_period, concat(b.apellidoPropietario, " ",
b.nombrePropietario) as prop_fecha, 0 as impPredial, 0 as solarNoEdificado, 0 as bomberos, 0 as computacion, 0 as
exoneracion, 0 as interes \
                FROM scum_parcela a, propietario b \
                WHERE (a.CodProv='+ codProv +' AND a.CodCant='+ codCant +' AND a.CodParroq='+ codParroq +' AND
a.CodZona='+ codZona +' AND a.CodSect='+ codSect +' AND a.CodPolg='+ codPolg +' AND a.CodParc='+ codParc +' AND
a.CodPropHoriz='+ codPropHoriz +') AND (b.codPropietario=a.codPropietario)) \
                UNION \
                (SELECT (cast(a.Ano as char)) as flag_ced_period, (DATE_FORMAT(CONCAT(SUBSTRING(a.CodAvaluo,1,2),
"-", SUBSTRING(a.CodAvaluo,3,2), "-", SUBSTRING(a.CodAvaluo,5,2)), "%d/%m/%Y")) as prop_fecha, a.impPredial,
a.solarNoEdificado, a.bomberos, a.computacion, a.exoneracion, sum(b.porcentajeinteres) as interes \
                 FROM scum_liquidacion_avaluo a, intereses b \
                 WHERE (a.CodProv='+ codProv +' AND a.CodCant='+ codCant +' AND a.CodParroq='+ codParroq +' AND
a.CodZona='+ codZona +' AND a.CodSect='+ codSect +' AND a.CodPolg='+ codPolg +' AND a.CodParc='+ codParc +' AND
a.CodPropHoriz='+ codPropHoriz +' AND a.pagado=0) \
                                AND \
                        (b.fecha BETWEEN \
                                                                        IF(date_format(NOW(), "%y-%m") >=
date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1 MONTH,
"%y-%m"), \

                date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1
MONTH, "%Y%m"), \
                                                                                "100001") \
                                        AND \
                                                                        IF(date_format(NOW(), "%y-%m") >=
date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1 MONTH,
"%y-%m"), \
                                                                                date_format(NOW( ), "%Y%m"), \
                                                                                "100001") \
                        ) \
                GROUP BY a.Ano) \
                ', function (error, results){
                  if(error) {
                    throw error;
                  } else {
                    //Compruebo que la patente exista, la posición results[0].anio corresponde al contador de filas y si es 1 existe la
patente caso contrario no existe
                    if (results[0].flag_ced_period != "N.E.") {
                      //obtener DATOS GENERALES
                      var cedula = results[0].flag_ced_period;
                      var propietario = results[0].prop_fecha;
                      var fechaEmiAct = "00/00/0000";
                      var periodoEmiAct = "0000";
                      var deudaEmiAct = 0;
                      var inteDescRecarAct = 0;
                      var interesEmiAct = 0;
                      var planillasEmiAnt = 0;
                      var periodoEmiAntIni = "";
                      var periodoEmiAntFin = "";
                      var periodoEmiAnt = "0000";
                      var deudaEmiAnt = 0;
                      var interesEmiAnt = 0;
                      var deudaTotal = 0;
                      //*****************************************************
                      //calcular el valor total de las deudas si existen
                      var planillasTot = results.length - 1; //obtengo el numero de planillas totales y a su vez es el index de la planilla
de emisiòn actual
                      if (planillasTot > 0) {
                        //*****************************************************
                        //calcular DEUDA EMISIÓN ACTUAL
                        periodoEmiAct = (results[planillasTot].flag_ced_period).toString();
                        fechaEmiAct = results[planillasTot].prop_fecha;
                        deudaEmiAct = results[planillasTot].impPredial + results[planillasTot].solarNoEdificado +
results[planillasTot].bomberos + results[planillasTot].computacion - results[planillasTot].exoneracion;
                        deudaEmiAct = Math.round(deudaEmiAct * 100) / 100;
                        //calculo del interes, descuento o recargo
                        var fechaHoy = new Date();
                        var anioHoy = (fechaHoy.getFullYear()).toString();
                        var mesHoy = fechaHoy.getMonth()+1;
                        var diaHoy = fechaHoy.getDate();
                        if (periodoEmiAct != anioHoy) {
                          //interes
                          inteDescRecarAct = results[planillasTot].interes;
                          interesEmiAct = results[planillasTot].impPredial * (inteDescRecarAct / 100);
```

```
                    } else {
                      switch(true) {
                        //descuento
                        case (mesHoy=1 && diaHoy<=15): inteDescRecarAct=-10; break;
                        case (mesHoy=1 && diaHoy>=16): inteDescRecarAct=-9; break;
                        case (mesHoy=2 && diaHoy<=15): inteDescRecarAct=-8; break;
                        case (mesHoy=2 && diaHoy>=16): inteDescRecarAct=-7; break;
                        case (mesHoy=3 && diaHoy<=15): inteDescRecarAct=-6; break;
                        case (mesHoy=3 && diaHoy>=16): inteDescRecarAct=-5; break;
                        case (mesHoy=4 && diaHoy<=15): inteDescRecarAct=-4; break;
                        case (mesHoy=4 && diaHoy>=16): case (mesHoy=5 && diaHoy<=15): inteDescRecarAct=-3; break;
                        case (mesHoy=5 && diaHoy>=16): case (mesHoy=6 && diaHoy<=15): inteDescRecarAct=-2; break;
                        case (mesHoy=6 && diaHoy>=16): inteDescRecarAct=-1; break;
                        //recargo
                        case mesHoy=7: case mesHoy=8: case mesHoy=9: case mesHoy=10: case mesHoy=11: case
mesHoy=12: inteDescRecarAct=10; break;
                      }
                      interesEmiAct = (results[planillasTot].impPredial + results[planillasTot].solarNoEdificado -
results[planillasTot].exoneracion) * (inteDescRecarAct / 100);
                    }
                    interesEmiAct = Math.round(interesEmiAct * 100) / 100;
                    //************************************************************
                    //calcular DEUDA EMISIONES ANTERIORES
                    if ((planillasTot-1) > 0) {
                      for (var cont=1; cont < planillasTot; cont++) {
                        deudaEmiAnt = deudaEmiAnt + results[cont].impPredial + results[cont].solarNoEdificado +
results[cont].bomberos + results[cont].computacion - results[cont].exoneracion;
                        interesEmiAnt = interesEmiAnt + (results[cont].impPredial * (results[cont].interes / 100));
                      }
                      if ((planillasTot-1) > 1) {
                        periodoEmiAntFin = " - " + (results[planillasTot-1].flag_ced_period).toString();
                      }
                      periodoEmiAntIni = (results[1].flag_ced_period).toString();
                      planillasEmiAnt = planillasTot-1;
                      periodoEmiAnt = periodoEmiAntIni + periodoEmiAntFin;
                      deudaEmiAnt = Math.round(deudaEmiAnt * 100) / 100;
                      interesEmiAnt = Math.round(interesEmiAnt * 100) / 100;
                    }

                    //************************************************************
                    //calcular DEUDA TOTAL
                    deudaTotal = deudaEmiAct + interesEmiAct + deudaEmiAnt + interesEmiAnt;
                  }
                  //************************************************************
                  //cargar y enviar los DATOS
                  var datosEnv=[{p1 : propietario,
                          c1 : cedula,
                          f2 : fechaEmiAct,
                          p2 : periodoEmiAct,
                          d2 : '$'+deudaEmiAct.toFixed(2),
                          i2 : '$'+interesEmiAct.toFixed(2),
                          t3 : planillasEmiAnt.toString(),
                          p3 : periodoEmiAnt,
                          d3 : '$'+deudaEmiAnt.toFixed(2),
                          i3 : '$'+interesEmiAnt.toFixed(2),
                          d4 : '$'+deudaTotal.toFixed(2)}];
                  res.send(200, datosEnv);
                } else {
                  //Si la patente no existe entro en este bucle
                  res.send(404);
                }
                return next();
              }
            }
  );
}
//Función para consultar el predio rural
function consultaPredioR(req, res, next){
  var codProv = req.params.codPredioR.substring(0, 2)
  var codCant = req.params.codPredioR.substring(2, 4)
  var codParroq = req.params.codPredioR.substring(4, 6)
  var codZona = req.params.codPredioR.substring(6, 8)
  var codSect = req.params.codPredioR.substring(8, 10)
  var codPolg = req.params.codPredioR.substring(10, 12)
  var codParc = req.params.codPredioR.substring(12, 15)
```

```
 connection.query('(SELECT IF(count(*)>0, b.cedula,"N.E.") as flag_ced_period, concat(b.apellidoPropietario, " ",
b.nombrePropietario) as prop_fecha, 0 as impPredial, 0 as bomberos, 0 as computacion, 0 as exoneracion, 0 as interes \
                FROM scr_parcela a, propietario b \
                WHERE (a.CodProv='+ codProv +' AND a.CodCant='+ codCant +' AND a.CodParroq='+ codParroq +' AND
a.CodZona='+ codZona +' AND a.CodSect='+ codSect +' AND a.CodPolg='+ codPolg +' AND a.CodParc='+ codParc +') AND
(b.codPropietario=a.codPropietario)) \
                UNION \
                (SELECT (cast(a.Ano as char)) as flag_ced_period, (DATE_FORMAT(CONCAT(SUBSTRING(a.CodAvaluo,1,2),
"-", SUBSTRING(a.CodAvaluo,3,2), "-", SUBSTRING(a.CodAvaluo,5,2)), "%d/%m/%Y")) as prop_fecha, a.impPredial,
a.bomberos, a.computacion, a.exoneracion, sum(b.porcentajeinteres) as interes \
                 FROM scr_liquidacion_avaluo a, intereses b \
                 WHERE (a.CodProv='+ codProv +' AND a.CodCant='+ codCant +' AND a.CodParroq='+ codParroq +' AND
a.CodZona='+ codZona +' AND a.CodSect='+ codSect +' AND a.CodPolg='+ codPolg +' AND a.CodParc='+ codParc +' AND
a.pagado=0) \
                                AND \
                        (b.fecha BETWEEN \
                                                                        IF(date_format(NOW(), "%y-%m") >=
date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1 MONTH,
"%y-%m"), \

                date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1
MONTH, "%Y%m"), \
                                                                                "100001") \
                                        AND \
                                                                        IF(date_format(NOW(), "%y-%m") >=
date_format(CONCAT(SUBSTRING(a.CodAvaluo,1,2), "-", SUBSTRING(a.CodAvaluo,3,2), "-01") + INTERVAL 1 MONTH,
"%y-%m"), \
                                                                                date_format(NOW( ), "%Y%m"), \
                                                                                "100001") \
                        ) \
                GROUP BY a.Ano) \
              ', function (error, results){
                if(error) {
                  throw error;
                } else {
                  //Compruebo que la patente exista, la posición results[0].anio corresponde al contador de filas y si es 1 existe la
patente caso contrario no existe
                  if (results[0].flag_ced_period != "N.E.") {
                    //obtener DATOS GENERALES
                    var cedula = results[0].flag_ced_period;
                    var propietario = results[0].prop_fecha;
                    var fechaEmiAct = "00/00/0000";
                    var periodoEmiAct = "0000";
                    var deudaEmiAct = 0;
                    var inteDescRecarAct = 0;
                    var interesEmiAct = 0;
                    var planillasEmiAnt = 0;
                    var periodoEmiAntIni = "";
                    var periodoEmiAntFin = "";
                    var periodoEmiAnt = "0000";
                    var deudaEmiAnt = 0;
                    var interesEmiAnt = 0;
                    var deudaTotal = 0;
                    //*******************************************************
                    //calcular el valor total de las deudas si existen
                    var planillasTot = results.length - 1; //obtengo el numero de planillas totales y a su vez es el index de la planilla
de emisiòn actual
                    if (planillasTot > 0) {
                      //*******************************************************
                      //calcular DEUDA EMISIÓN ACTUAL
                      periodoEmiAct = (results[planillasTot].flag_ced_period).toString();
                      fechaEmiAct = results[planillasTot].prop_fecha;
                      deudaEmiAct = results[planillasTot].impPredial + results[planillasTot].bomberos +
results[planillasTot].computacion - results[planillasTot].exoneracion;
                      deudaEmiAct = Math.round(deudaEmiAct * 100) / 100;
                      //calculo del interes, descuento o recargo
                      var fechaHoy = new Date();
                      var anioHoy = (fechaHoy.getFullYear()).toString();
                      var mesHoy = fechaHoy.getMonth()+1;
                      var diaHoy = fechaHoy.getDate();

                      if (periodoEmiAct != anioHoy) {
                        //interes
                        inteDescRecarAct = results[planillasTot].interes;
                                                interesEmiAct = results[planillasTot].impPredial * (inteDescRecarAct / 100);
```

```
                    } else {
                      switch(mesHoy) {
                        //descuento
                        case 1: case 2: case 3: case 4: case 5: case 6: case 7: case 8: case 9: inteDescRecarAct=-10; break;
                        //recargo
                        case 10: case 11: case 12: inteDescRecarAct=0; break;
                      }
                                                    interesEmiAct = (results[planillasTot].impPredial -
results[planillasTot].exoneracion) * (inteDescRecarAct / 100);
                    }
                    interesEmiAct = Math.round(interesEmiAct * 100) / 100;
                    //************************************************************
                    //calcular DEUDA EMISIONES ANTERIORES
                    if ((planillasTot-1) > 0) {
                      for (var cont=1; cont < planillasTot; cont++) {
                        deudaEmiAnt = deudaEmiAnt + results[cont].impPredial + results[cont].bomberos +
results[cont].computacion - results[cont].exoneracion;
                        interesEmiAnt = interesEmiAnt + (results[cont].impPredial * (results[cont].interes / 100));
                      }
                      if ((planillasTot-1) > 1) {
                        periodoEmiAntFin = " - " + (results[planillasTot-1].flag_ced_period).toString();
                      }
                      periodoEmiAntIni = (results[1].flag_ced_period).toString();
                      planillasEmiAnt = planillasTot-1;
                      periodoEmiAnt = periodoEmiAntIni + periodoEmiAntFin;
                      deudaEmiAnt = Math.round(deudaEmiAnt * 100) / 100;
                      interesEmiAnt = Math.round(interesEmiAnt * 100) / 100;
                    }
                    //************************************************************
                    //calcular DEUDA TOTAL
                    deudaTotal = deudaEmiAct + interesEmiAct + deudaEmiAnt + interesEmiAnt;
                  }

                  //************************************************************
                  //cargar y enviar los DATOS
                  var datosEnv=[{p1 : propietario,
                          c1 : cedula,
                          f2 : fechaEmiAct,
                          p2 : periodoEmiAct,
                          d2 : '$'+deudaEmiAct.toFixed(2),
                          i2 : '$'+interesEmiAct.toFixed(2),
                          t3 : planillasEmiAnt.toString(),
                          p3 : periodoEmiAnt,
                          d3 : '$'+deudaEmiAnt.toFixed(2),
                          i3 : '$'+interesEmiAct.toFixed(2),
                          d4 : '$'+deudaTotal.toFixed(2)}];
                  res.send(200, datosEnv);
                } else {
                  //Si la patente no existe entro en este bucle
                  res.send(404);
                }
                return next();
              }
            }
  );

}


//Función para consultar el Agua
function consultaAgua(req, res, next) {
  connection.query('(SELECT IF(count(*)>0, b.cedula,"N.E.") as flag_ced_period, concat(b.apellidoPropietario, " ",
b.nombrePropietario) as prop_fecha, 0 as valor_principal, 0 as alcantarillado, 0 as basura, 0 as total_tarifa, 0 as interes \
            FROM agua_medidor_abonado a, propietario b \
            WHERE a.codmedidor=' + req.params.codAgua + ' AND b.codPropietario=a.codPropietario) \
            UNION \
            (SELECT (a.codemision) as flag_ced_period, (DATE_FORMAT(a.fecha_factura, "%d/%m/%Y")) as prop_fecha,
a.valor_principal, a.alcantarillado, a.basura, a.total_tarifa, sum(b.porcentajeinteres) as interes \
            FROM agua_factura a, intereses b \
            WHERE (a.codMedidor=' + req.params.codAgua + ' AND a.pagado=0 AND estado=1 AND id_clase_rubro=1) \
              AND \
                (b.fecha BETWEEN \
                  IF( date_format(NOW(), "%Y-%m-%d") >= DATE_ADD(a.fecha_factura, INTERVAL 1 MONTH), \
                                        date_format(DATE_ADD(a.fecha_factura, INTERVAL 1 MONTH), "%Y%m"), \
                                        100001) \
```

```
                    AND \
                        IF( date_format(NOW(), "%Y-%m-%d") >= DATE_ADD(a.fecha_factura, INTERVAL 1 MONTH),
\
                            IF( DAY(NOW( )) >= DAY(DATE_ADD(a.fecha_factura, INTERVAL
period_diff(date_format(NOW(), "%Y%m"), date_format(a.fecha_factura, "%Y%m")) MONTH)), \
                                date_format(NOW( ), "%Y%m"), \
                                date_format(DATE_SUB(NOW( ), INTERVAL 1 MONTH), "%Y%m")), \
                            100001) \
                    ) \
            GROUP BY a.codemision) \
            ', function (error, results){
              if(error) {
                throw error;
              } else {
                //Compruebo que la patente exista, la posición results[0].anio corresponde al contador de filas y si es 1 existe la
patente caso contrario no existe
                if (results[0].flag_ced_period != "N.E.") {
                  //obtener DATOS GENERALES
                  var cedula = results[0].flag_ced_period;
                  var propietario = results[0].prop_fecha;
                  var fechaEmiAct = "00/00/0000";
                  var periodoMes = "00";
                  var descAnio = 0;
                  var periodoEmiAct = "0000";
                  var deudaEmiAct = 0;
                  var interesEmiAct = 0;
                  var planillasEmiAnt = 0;
                  var periodoEmiAntIni = "";
                  var periodoEmiAntFin = "";
                  var periodoEmiAnt = "0000";
                  var deudaEmiAnt = 0;
                  var interesEmiAnt = 0;
                  var deudaTotal = 0;
                  //*********************************************************
                  //calcular el valor total de las deudas si existen
                  var planillasTot = results.length - 1; //obtengo el numero de planillas totales y a su vez es el index de la planilla
de emisiòn actual
                  if (planillasTot > 0) {
                    //*********************************************************
                    //calcular DEUDA EMISIÓN ACTUAL
                    var meses = new Array ("Dic","Ene","Feb","Mar","Abr","May","Jun","Jul","Ago","Sep","Oct","Nov");
                    if ((results[planillasTot].flag_ced_period).substring(2, 4) == "01") { descAnio = 1; };
                    periodoEmiAct = meses[parseInt((results[planillasTot].flag_ced_period).substring(2, 4))-1] +
(parseInt((results[planillasTot].flag_ced_period).substring(0, 2))-descAnio);
                    fechaEmiAct = results[planillasTot].prop_fecha;
                    deudaEmiAct = results[planillasTot].total_tarifa;
                    deudaEmiAct = Math.round(deudaEmiAct * 100) / 100;
                    interesEmiAct = (results[planillasTot].valor_principal + results[planillasTot].alcantarillado +
results[planillasTot].basura) * (results[planillasTot].interes / 100);
                    interesEmiAct = Math.round(interesEmiAct * 100) / 100;
                    //*********************************************************
                    //calcular DEUDA EMISIONES ANTERIORES
                    if ((planillasTot-1) > 0) {
                      for (var cont=1; cont < planillasTot; cont++) {
                        deudaEmiAnt = deudaEmiAnt + results[cont].total_tarifa;
                        interesEmiAnt = interesEmiAnt + ((results[cont].valor_principal + results[cont].alcantarillado +
results[cont].basura) * (results[cont].interes / 100));
                      }
                      if ((planillasTot-1) > 1) {
                        periodoEmiAntFin = " - " + meses[parseInt((results[planillasTot-1].flag_ced_period).substring(2, 4))-1] +
(results[planillasTot-1].flag_ced_period).substring(0, 2);
                      }
                      if ((results[1].flag_ced_period).substring(2, 4) == "01") { descAnio = 1;} else {descAnio = 0;}
                      periodoEmiAntIni = meses[parseInt((results[1].flag_ced_period).substring(2, 4))-1] +
(parseInt((results[1].flag_ced_period).substring(0, 2))-descAnio);
                      planillasEmiAnt = planillasTot-1;
                      periodoEmiAnt =  periodoEmiAntIni + periodoEmiAntFin;
                      deudaEmiAnt = Math.round(deudaEmiAnt * 100) / 100;
                      interesEmiAnt = Math.round(interesEmiAnt * 100) / 100;
                    }
                    //*********************************************************
                    //calcular DEUDA TOTAL
                    deudaTotal = deudaEmiAct + interesEmiAct + deudaEmiAnt + interesEmiAnt;
                  }
                  //*********************************************************
```

```
                    //cargar y enviar los DATOS
                    var datosEnv=[{p1 : propietario,
                            c1 : cedula,
                            f2 : fechaEmiAct,
                            p2 : periodoEmiAct,
                            d2 : '$'+deudaEmiAct.toFixed(2),
                            i2 : '$'+interesEmiAct.toFixed(2),
                            t3 : planillasEmiAnt.toString(),
                            p3 : periodoEmiAnt,
                            d3 : '$'+deudaEmiAnt.toFixed(2),
                            i3 : '$'+interesEmiAnt.toFixed(2),
                            d4 : '$'+deudaTotal.toFixed(2)}];
                    res.send(200, datosEnv);
                  } else {
                    //Si la patente no existe entro en este bucle
                    res.send(404);
                  }
                  return next();
                }
              }
  );
}

//Función para consultar la patente
function consultaPatente(req, res, next) {
  connection.query('(SELECT IF(count(*)>0, b.cedula,"N.E.") as flag_ced_period, concat(b.apellidoPropietario, " ",
b.nombrePropietario) as prop_fecha, 0 as valorImpuesto, 0 as impuestoCapital, 0 as servAdmin, 0 as otro, 0 as valorEspe, 0 as
interes \
              FROM patentes_patente a, propietario b \
              WHERE a.numPatente=' + req.params.codPatente + ' AND b.codPropietario=a.codPropietario) \
              UNION \
              (SELECT a.ano as flag_ced_period, (DATE_FORMAT(a.fecha_emision, "%d/%m/%Y")) as prop_fecha,
a.valor_impuesto, a.impuesto_capital, a.serv_administrativos, a.otro, a.Valor_Especie, sum(b.porcentajeinteres) as interes \
               FROM patentes_emision_patente a, intereses b \
               WHERE (a.numpatente=' + req.params.codPatente + ' AND a.pagado=0) \
                           AND \
                    (b.fecha BETWEEN \
                        IF( date_format(NOW(), "%Y-%m-%d") >= DATE_ADD(a.fecha_emision, INTERVAL 1 MONTH), \
                                                date_format(DATE_ADD(a.fecha_emision, INTERVAL 1 MONTH), "%Y%m"), \
                                                100001) \
                                AND \
                                        IF( date_format(NOW(), "%Y-%m-%d") >= DATE_ADD(a.fecha_emision, INTERVAL 1
MONTH), \
                                                IF( DAY(NOW( )) >= DAY(DATE_ADD(a.fecha_emision, INTERVAL
period_diff(date_format(NOW(), "%Y%m"), date_format(a.fecha_emision, "%Y%m")) MONTH)), \
                                                        date_format(NOW( ), "%Y%m"), \
                                                        date_format(DATE_SUB(NOW( ), INTERVAL 1 MONTH), "%Y%m")),
\
                                                100001) \
                    ) \
              GROUP BY a.ano) \
              ', function (error, results){
                if(error) {
                  throw error;
                } else {
                  //Compruebo que la patente exista, la posición results[0].anio corresponde al contador de filas y si es 1 existe la
patente caso contrario no existe
                  if (results[0].flag_ced_period != "N.E.") {
                    //obtener DATOS GENERALES
                    var cedula = (results[0].flag_ced_period).toString();
                    var propietario = results[0].prop_fecha;
                    var fechaEmiAct = "00/00/0000";
                    var periodoEmiAct = "0000";
                    var deudaEmiAct = 0;
                    var interesEmiAct = 0;
                    var planillasEmiAnt = 0;
                    var periodoEmiAntIni = "";
                    var periodoEmiAntFin = "";
                    var periodoEmiAnt = "0000";
                    var deudaEmiAnt = 0;
                    var interesEmiAnt = 0;
                    var deudaTotal = 0;
                    //*****************************************************
                    //calcular el valor total de las deudas si existen
```

```
                var planillasTot = results.length - 1; //obtengo el numero de planillas totales y a su vez es el index de la planilla
de emisiòn actual
                if (planillasTot > 0) {
                  //************************************************************
                  //calcular DEUDA EMISIÓN ACTUAL
                  periodoEmiAct = (results[planillasTot].flag_ced_period).toString();
                  fechaEmiAct = results[planillasTot].prop_fecha;
                  deudaEmiAct = results[planillasTot].valorImpuesto + results[planillasTot].impuestoCapital +
results[planillasTot].servAdmin + results[planillasTot].otro + results[planillasTot].valorEspe;
                  deudaEmiAct = Math.round(deudaEmiAct * 100) / 100;
                  interesEmiAct = results[planillasTot].valorImpuesto * (results[planillasTot].interes / 100);
                  interesEmiAct = Math.round(interesEmiAct * 100) / 100;
                  //************************************************************
                  //calcular DEUDA EMISIONES ANTERIORES
                  if ((planillasTot-1) > 0) {
                    for (var cont=1; cont < planillasTot; cont++) {
                      deudaEmiAnt = deudaEmiAnt + results[cont].valorImpuesto + results[cont].impuestoCapital +
results[cont].servAdmin + results[cont].otro + results[cont].valorEspe;
                      interesEmiAnt = interesEmiAnt + (results[cont].valorImpuesto * (results[cont].interes / 100));
                    }
                    if ((planillasTot-1) > 1) {
                      periodoEmiAntFin = " - " + (results[planillasTot-1].flag_ced_period).toString();
                    }
                    periodoEmiAntIni = (results[1].flag_ced_period).toString();
                    planillasEmiAnt = planillasTot-1;
                    periodoEmiAnt = periodoEmiAntIni + periodoEmiAntFin;
                    deudaEmiAnt = Math.round(deudaEmiAnt * 100) / 100;
                    interesEmiAnt = Math.round(interesEmiAnt * 100) / 100;
                  }
                  //************************************************************
                  //calcular DEUDA TOTAL
                  deudaTotal = deudaEmiAct + interesEmiAct + deudaEmiAnt + interesEmiAnt;
                }

                //************************************************************
                //cargar y enviar los DATOS
                var datosEnv=[{p1 : propietario,
                          c1 : cedula,
                          f2 : fechaEmiAct,
                          p2 : periodoEmiAct,
                          d2 : '$'+deudaEmiAct.toFixed(2),
                          i2 : '$'+interesEmiAct.toFixed(2),
                          t3 : planillasEmiAnt.toString(),
                          p3 : periodoEmiAnt,
                          d3 : '$'+deudaEmiAnt.toFixed(2),
                          i3 : '$'+interesEmiAnt.toFixed(2),
                          d4 : '$'+deudaTotal.toFixed(2)}];
                res.send(200, datosEnv);
              } else {
                //Si la patente no existe entro en este bucle
                res.send(404);
              }
              return next();
            }
          }
  );
}
```

Anexo 3: Código fuente de la app “CHORDELEG móvil”

- Archivo “AndroidManifest.xml”

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="ec.gob.chordeleg.chordemovil">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
```

```
        android:theme="@style/AppTheme">
        <activity android:name=".Bienvenido">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".Menu" />
        <activity android:name=".Patente_Consulta" />
        <activity android:name=".Patente_Reporte" />
        <activity android:name=".Agua_Consulta" />
        <activity android:name=".Agua_Reporte" />
        <activity android:name=".PredioR_Consulta" />
        <activity android:name=".PredioR_Reporte" />
        <activity android:name=".PredioU_Consulta" />
        <activity android:name=".PredioU_Reporte"></activity>
    </application>
</manifest>
```

- Archivo “Bienvenido.java”

```
package ec.gob.chordeleg.chordemovil;
import android.content.Intent;
import android.os.Handler;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
public class Bienvenido extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_bienvenido);
        Handler handler = new Handler();
        new Handler().postDelayed(new Runnable() {
            @Override
            public void run() {
                Intent intentM = new Intent(Bienvenido.this, Menu.class);
                startActivity(intentM);
                finish();
            }
        }, 2000);

    }
}
```

- Archivo “activity_bienvenido.xml”

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_bienvenido"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:weightSum="1"
    android:layout_height="match_parent"
    tools:context="ec.gob.chordeleg.chordemovil.Bienvenido"
    android:background="@android:color/background_dark">
    <LinearLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent">
        <TextView
            android:text="BIENVENIDO A"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/eti_Bienvenido"
            android:textColor="@android:color/holo_orange_light"
            android:textSize="28sp"
            android:layout_above="@+id/img_LogoCandonga"
            android:layout_centerHorizontal="true"
            android:textStyle="normal|bold|italic"
            android:textAlignment="center"
            android:layout_marginTop="20dp" />
```

```
        <TextView
            android:text="'CHORDELEGmóvil"'
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/eti_CHM"
            android:textColor="@android:color/holo_orange_light"
            android:textSize="28sp"
            android:layout_above="@+id/img_LogoCandonga"
            android:layout_centerHorizontal="true"
            android:textStyle="normal|bold"
            android:textAlignment="center"
            android:fontFamily="serif" />
        <ImageView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            app:srcCompat="@drawable/candongaini"
            android:id="@+id/img_LogoCandonga"
            android:adjustViewBounds="true"
            android:layout_centerVertical="true"
            android:layout_alignRight="@+id/eti_Bienvenido"
            android:layout_alignEnd="@+id/eti_Bienvenido"
            android:layout_gravity="center_horizontal"
            android:layout_weight="0.54"
            android:paddingRight="30dp"
            android:paddingLeft="30dp"
            android:paddingBottom="10dp"
            android:paddingTop="10dp" />
        <TextView
            android:text="DESARROLLADO POR"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/eti_Desarrollo"
            android:textColor="@android:color/holo_orange_light"
            android:textSize="28sp"
            android:textAlignment="center"
            android:fontFamily="sans-serif"
            android:textStyle="normal|bold" />
        <TextView
            android:text="'Ing. Jhovany Buele Msc."'
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/eti_JB"
            android:textColor="@android:color/holo_orange_light"
            android:textSize="28sp"
            android:textAlignment="center"
            android:fontFamily="serif"
            android:textStyle="normal|bold"
            android:layout_marginBottom="20dp" />
    </LinearLayout>
</LinearLayout>
```

- Archivo “Menu.java”

```
package ec.gob.chordeleg.chordemovil;
import android.content.DialogInterface;
import android.content.Intent;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.KeyEvent;
import android.view.View;
import android.widget.Button;
import android.widget.ImageButton;
import android.widget.Toast;

public class Menu extends AppCompatActivity implements View.OnClickListener {
    //Variables globales
    private Button btnPredioUrb, btnPredioRus, btnAgua, btnPatente;
    private ImageButton imgbtnMinimiza, imgbtnInfo;
    private AlertDialog.Builder dialogoInfo;
    private Toast toast;
    //Función de inicio del activity
    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_menu);
        inicializar();
    }
    //Función para inicializar la configuración
    private void inicializar() {
        //Asociar cada btn,text,etc al view y habilitar el listener
        btnPredioUrb = (Button) findViewById(R.id.btn_PredioUrb);
        btnPredioUrb.setOnClickListener(this);
        btnPredioRus = (Button) findViewById(R.id.btn_PredioRus);
        btnPredioRus.setOnClickListener(this);
        btnAgua = (Button) findViewById(R.id.btn_Agua);
        btnAgua.setOnClickListener(this);
        btnPatente = (Button) findViewById(R.id.btn_Patente);
        btnPatente.setOnClickListener(this);
        imgbtnMinimiza = (ImageButton) findViewById(R.id.imgbtn_Minimiza);
        imgbtnMinimiza.setOnClickListener(this);
        imgbtnInfo = (ImageButton) findViewById(R.id.imgbtn_Info);
        imgbtnInfo.setOnClickListener(this);
        //Configurar el mensaje de INFO
        dialogoInfo = new AlertDialog.Builder(this);
        dialogoInfo.setTitle("Acerca de:");
        dialogoInfo.setCancelable(false);
        dialogoInfo.setMessage("Aplicación Desarrollada por: Msc. Ing. Jhovany Buele V.");
        dialogoInfo.setPositiveButton("OK", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialogoInfo, int id) {
                dialogoInfo.cancel();
            }
        });
    }
    //Función para establecer la acción del btn atras presionado
    @Override
    public boolean onKeyDown(int keyCode, KeyEvent event) {
        if (keyCode == KeyEvent.KEYCODE_BACK && event.getRepeatCount() == 0) {
            moveTaskToBack(true);   //minimizar la aplicación
        }
        return super.onKeyDown(keyCode, event);
    }
    //Función para establecer la acción de cada btn presionado
    @Override
    public void onClick(View view) {
        switch (view.getId()) {
            case R.id.btn_PredioUrb:
                Intent intentConsultaPredioU = new Intent(Menu.this, PredioU_Consulta.class);
                startActivity(intentConsultaPredioU);
                break;
            case R.id.btn_PredioRus:
                Intent intentConsultaPredioR = new Intent(Menu.this, PredioR_Consulta.class);
                startActivity(intentConsultaPredioR);
                break;
            case R.id.btn_Agua:
                Intent intentConsultaAgua = new Intent(Menu.this, Agua_Consulta.class);
                startActivity(intentConsultaAgua);
                break;
            case R.id.btn_Patente:
                Intent intentConsultaPatente = new Intent(Menu.this, Patente_Consulta.class);
                startActivity(intentConsultaPatente);
                break;
            case R.id.imgbtn_Minimiza:
                moveTaskToBack(true);
                break;
            case R.id.imgbtn_Info:
                dialogoInfo.show();
                break;
        }
    }
}
```

- Archivo “activity_menu.xml”

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
```

```
android:id="@+id/activity_menu"
android:orientation="vertical"
android:layout_width="match_parent"
android:weightSum="1"
android:layout_height="match_parent"
tools:context="ec.gob.chordeleg.chordemovil.Menu"
android:background="@color/colorPrimary">
<LinearLayout
   android:orientation="horizontal"
   android:layout_width="match_parent"
   android:background="@color/colorPrimary"
   android:layout_height="45dp">
   <ImageButton
      android:layout_height="match_parent"
      android:id="@+id/imgbtn_Info"
      android:background="@drawable/btn_cafe"
      app:srcCompat="@drawable/info32"
      android:layout_width="wrap_content"
      android:paddingLeft="12dp" />
   <TextView
      android:text="CHORDELEGmóvil"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/eti_NombreApp"
      android:background="@color/colorPrimary"
      android:textColor="@android:color/holo_orange_light"
      android:textAlignment="center"
      android:fontFamily="sans-serif"
      android:textSize="22sp"
      android:gravity="center_vertical"
      android:layout_weight="1" />
   <ImageButton
      android:layout_height="match_parent"
      app:srcCompat="@drawable/minimizar32"
      android:id="@+id/imgbtn_Minimiza"
      android:background="@drawable/btn_cafe"
      android:layout_width="wrap_content"
      android:paddingRight="12dp" />
</LinearLayout>
<ScrollView
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:background="@android:color/background_dark">
   <LinearLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:orientation="vertical" >
      <ImageView
         android:layout_width="match_parent"
         app:srcCompat="@drawable/candongaini"
         android:id="@+id/img_LogoCandonga"
         android:background="@android:color/background_dark"
         android:layout_height="172dp"
         android:paddingBottom="20dp"
         android:paddingTop="20dp" />
      <Button
         android:text="PREDIO URBANO"
         android:layout_width="match_parent"
         android:id="@+id/btn_PredioUrb"
         android:background="@drawable/btn_amarillo"
         android:fontFamily="sans-serif"
         android:textSize="22sp"
         android:textColor="@android:color/background_dark"
         android:drawableLeft="@drawable/urbano64"
         android:drawableRight="@drawable/adelante40"
         android:paddingLeft="15dp"
         android:paddingRight="10dp"
         android:textAlignment="textStart"
         android:drawablePadding="15dp"
         android:textAllCaps="false"
         android:layout_height="70dp"
         android:textStyle="normal|bold" />
      <Button
         android:text="PREDIO RURAL"
         android:layout_width="match_parent"
```

```
            android:id="@+id/btn_PredioRus"
            android:fontFamily="sans-serif"
            android:textSize="22sp"
            android:textColor="@android:color/background_dark"
            android:background="@drawable/btn_amarillo"
            android:drawableLeft="@drawable/rural64"
            android:drawableRight="@drawable/adelante40"
            android:paddingLeft="15dp"
            android:paddingRight="10dp"
            android:textAlignment="textStart"
            android:drawablePadding="15dp"
            android:layout_height="70dp"
            android:textAllCaps="false"
            android:textStyle="normal|bold" />
        <Button
            android:text="AGUA POTABLE"
            android:layout_width="match_parent"
            android:id="@+id/btn_Agua"
            android:fontFamily="sans-serif"
            android:textSize="22sp"
            android:textColor="@android:color/background_dark"
            android:background="@drawable/btn_amarillo"
            android:drawableLeft="@drawable/agua64"
            android:drawableRight="@drawable/adelante40"
            android:paddingLeft="15dp"
            android:paddingRight="10dp"
            android:textAlignment="textStart"
            android:drawablePadding="15dp"
            android:layout_height="70dp"
            android:textAllCaps="false"
            android:textStyle="normal|bold" />
        <Button
            android:text="PATENTE"
            android:layout_width="match_parent"
            android:id="@+id/btn_Patente"
            android:textColor="@android:color/background_dark"
            android:fontFamily="sans-serif"
            android:textSize="22sp"
            android:background="@drawable/btn_amarillo"
            android:drawableLeft="@drawable/patente64"
            android:drawableRight="@drawable/adelante40"
            android:paddingLeft="15dp"
            android:paddingRight="10dp"
            android:textAlignment="textStart"
            android:drawablePadding="15dp"
            android:layout_height="70dp"
            android:textAllCaps="false"
            android:textStyle="normal|bold" />
    </LinearLayout>
  </ScrollView>
</LinearLayout>
```

- Archivo “PredioU_Consulta.java”

```
package ec.gob.chordeleg.chordemovil;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Gravity;
import android.view.View;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.Toast;
import com.android.volley.DefaultRetryPolicy;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.ServerError;
```

```
import com.android.volley.TimeoutError;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;
import java.io.UnsupportedEncodingException;
public class PredioU_Consulta extends AppCompatActivity implements Response.Listener<String>, Response.ErrorListener,
View.OnClickListener{
    //Variables globales
    private ImageButton imgbtnAtras, imgbtnMinimiza;
    private Button btnConsultaPredioU;
    private EditText editCodPredioU;
    private RequestQueue requestQueue;
    private String url;
    private StringRequest stringRequest;
    private ProgressDialog dialogoEspera;
    private Toast toast;
    private Boolean conectado;
    //Función de inicio del activity
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_prediou_consulta);
        inicailizar();
    }
    //Función para inicializar la configuración
    private void inicailizar(){
        //Asociar cada btn,text,etc al view y habilitar el listener
        btnConsultaPredioU = (Button) findViewById(R.id.btn_ConsultaPredioU);
        btnConsultaPredioU.setOnClickListener(this);
        imgbtnAtras = (ImageButton) findViewById(R.id.imgbtn_Atras);
        imgbtnAtras.setOnClickListener(this);
        imgbtnMinimiza = (ImageButton) findViewById(R.id.imgbtn_Minimiza);
        imgbtnMinimiza.setOnClickListener(this);
        editCodPredioU = (EditText) findViewById(R.id.edit_CodPredioU);
        //Configurar las variables para consumir la APIrest del WS
        requestQueue = Volley.newRequestQueue(this);
        url = "http://ws.chordeleg.gob.ec:3001/predioU/";
        //Configurar las notificaciones
        toast = Toast.makeText(this, "", Toast.LENGTH_LONG);
        toast.setGravity(Gravity.CENTER_VERTICAL, 0, 0);
        //Configurar el dialogo de espera
        dialogoEspera = new ProgressDialog(this);
        dialogoEspera.setCancelable(false);
        dialogoEspera.setMessage("Procesando...");
        //Configurar el teclado para que inicie en la pantalla
     getWindow().setSoftInputMode(WindowManager.LayoutParams.SOFT_INPUT_STATE_ALWAYS_VISIBLE);
    }
    //Función para consultar los datos
    private void consultar(String codPredioU){
        if (validaInternet(this)) {
            dialogoEspera.show();
            stringRequest = new StringRequest(Request.Method.GET, url + codPredioU, this, this);
            stringRequest.setRetryPolicy(new DefaultRetryPolicy(500, 3, 1));
            requestQueue.add(stringRequest);
        } else {
            toast.setText("Revice su conexión a internet...");
            toast.show();
        }
    }

    //Función para validar la conexión a internet
    public boolean validaInternet(Context context) {
        conectado = false;
        ConnectivityManager connec = (ConnectivityManager) context.getSystemService(Context.CONNECTIVITY_SERVICE);
        //Obtiene todas las redes (móviles y wifi)
        NetworkInfo[] redes = connec.getAllNetworkInfo();
        for (int i = 0; i < redes.length; i++) {
            if (redes[i].getState() == NetworkInfo.State.CONNECTED) {
                conectado = true;
                i = redes.length;
            }
        }
        return conectado;
    }
    //Función para establecer la acción de cada btn presionado
```

```
    @Override
    public void onClick(View view) {
        switch (view.getId()) {
            case R.id.btn_ConsultaPredioU:
                if (!editCodPredioU.getText().toString().equals("")) {
                    consultar(editCodPredioU.getText().toString());
                } else {
                    toast.setText("Ingrese su clave de Predio Urbano...");
                    toast.show();
                }
                break;
            case R.id.imgbtn_Atras:
                onBackPressed();
                break;
            case R.id.imgbtn_Minimiza:
                moveTaskToBack(true);
                break;
        }
    }
    //Función para establecer la acción si se obtiene respuesta del WS
    @Override
    public void onResponse(String response) {
        try {
            response = new String(response.getBytes("ISO-8859-1"), "UTF-8");
            Intent intentReportePat = new Intent(this, PredioU_Reporte.class);
            intentReportePat.putExtra("codPredioU", editCodPredioU.getText().toString());
            intentReportePat.putExtra("datosStr", response);
            dialogoEspera.dismiss();
            startActivity(intentReportePat);
        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        }
    }
    //Función para establecer la acción si se obtiene un error del WS
    @Override
    public void onErrorResponse(VolleyError error) {
        dialogoEspera.dismiss();
        if (error instanceof ServerError) {
            toast.setText("No existe el Predio ingresado...");
            toast.show();
        } else if (error instanceof TimeoutError) {
            toast.setText("Tiempo de espera agotado, vuelva a intentarlo...");
            toast.show();
        } else {
            toast.setText("Lo sentimos, servicio no disponible. Inténtelo más tarde...");
            toast.show();
        }
    }
}
```

- Archivo “activity_prediou_consulta.xml”

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_patente_consulta"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:weightSum="1"
    tools:context="ec.gob.chordeleg.chordemovil.PredioU_Consulta"
    android:background="@android:color/background_dark">
    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:background="@color/colorPrimary"
        android:layout_height="45dp"
        android:id="@+id/Ll_actionBar">
        <ImageButton
            android:layout_height="match_parent"
            app:srcCompat="@drawable/atras32"
            android:id="@+id/imgbtn_Atras"
            android:background="@drawable/btn_cafe"
```

```
            android:layout_width="wrap_content"
            android:paddingLeft="12dp" />
        <TextView
            android:text="CHORDELEGmóvil"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:id="@+id/eti_NombreApp"
            android:background="@color/colorPrimary"
            android:textColor="@android:color/holo_orange_light"
            android:textAlignment="center"
            android:fontFamily="sans-serif"
            android:textSize="22sp"
            android:gravity="center_vertical"
            android:layout_weight="1" />
        <ImageButton
            android:layout_height="match_parent"
            app:srcCompat="@drawable/minimizar32"
            android:id="@+id/imgbtn_Minimiza"
            android:background="@drawable/btn_cafe"
            android:layout_width="wrap_content"
            android:paddingRight="12dp" />
    </LinearLayout>
    <ImageView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:srcCompat="@drawable/urbano114"
        android:id="@+id/img_LogoPredioU"
        android:background="@android:color/background_dark" />
    <TextView
        android:text="CLAVE DE PREDIO URBANO"
        android:layout_width="match_parent"
        android:id="@+id/eti_CodPredioU"
        android:gravity="center_vertical|center_horizontal"
        android:background="@drawable/btn_amarillopress"
        android:fontFamily="sans-serif"
        android:textSize="22sp"
        android:textColor="@android:color/background_dark"
        android:layout_height="38dp" />
    <EditText
        android:layout_width="match_parent"
        android:id="@+id/edit_CodPredioU"
        android:hint="Ingrese su clave"
        android:textSize="22sp"
        android:textAlignment="center"
        android:fontFamily="sans-serif"
        android:textAllCaps="false"
        android:inputType="number"
        android:background="@drawable/edit_blanco"
        android:layout_gravity="center_horizontal"
        android:ems="10"
        android:maxLength="18"
        android:layout_height="42dp" />
    <Button
        android:layout_width="186dp"
        android:layout_height="45dp"
        android:id="@+id/btn_ConsultaPredioU"
        android:fontFamily="sans-serif"
        android:background="@drawable/btn_amarillo"
        android:textColor="@android:color/background_dark"
        android:textSize="22sp"
        android:textAllCaps="false"
        android:layout_gravity="center_horizontal"
        android:text="CONSULTAR"
        android:drawableLeft="@drawable/lupa32"
        android:paddingLeft="8dp"
        android:textStyle="normal|bold"
        android:layout_marginTop="5dp" />
</LinearLayout>
```

- Archivo “PredioU_Reporte.java”

```
package ec.gob.chordeleg.chordemovil;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
```

```
import android.view.Gravity;
import android.view.View;
import android.widget.ImageButton;
import android.widget.TextView;
import android.widget.Toast;
import org.json.JSONArray;
import org.json.JSONException;
public class PredioU_Reporte extends AppCompatActivity implements View.OnClickListener {
    //Variables globales
    private ImageButton imgbtnAtras, imgbtnMinimiza;
    private TextView textCedula, textPropietario, textCodPredioU,
            textFechaAct, textPeriodoAct, textDeudaAct, textInteresAct,
            textPlanillasAnt, textPeriodoAnt, textDeudaAnt, textInteresAnt,
            textDeudaTotal;
    private String datosStr;
    private JSONArray datosJSON;
    private Toast toast;
    //Función de inicio del activity
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_prediou_reporte);
        inicializar();
        recibirDatos();
    }
    //Función para inicializar la configuración
    private void inicializar() {
        //Asociar cada btn,text,etc al view y habilitar el listener
        imgbtnAtras = (ImageButton) findViewById(R.id.imgbtn_Atras);
        imgbtnAtras.setOnClickListener(this);
        imgbtnMinimiza = (ImageButton) findViewById(R.id.imgbtn_Minimiza);
        imgbtnMinimiza.setOnClickListener(this);
        textCedula = (TextView) findViewById(R.id.text_Cedula);
        textPropietario = (TextView) findViewById(R.id.text_Propietario);
        textCodPredioU = (TextView) findViewById(R.id.text_CodPredioU);
        textFechaAct = (TextView) findViewById(R.id.text_FechaAct);
        textPeriodoAct = (TextView) findViewById(R.id.text_PeriodoAct);
        textDeudaAct = (TextView) findViewById(R.id.text_DeudaAct);
        textInteresAct = (TextView) findViewById(R.id.text_InteresAct);
        textPlanillasAnt = (TextView) findViewById(R.id.text_PlanillasAnt);
        textPeriodoAnt = (TextView) findViewById(R.id.text_PeriodoAnt);
        textDeudaAnt = (TextView) findViewById(R.id.text_DeudaAnt);
        textInteresAnt = (TextView) findViewById(R.id.text_InteresAnt);
        textDeudaTotal = (TextView) findViewById(R.id.text_DeudaTotal);
        //Configurar la notificación de deuda pendiente
        toast = Toast.makeText(this, "No tiene deudas pendientes...", Toast.LENGTH_LONG);
        toast.setGravity(Gravity.CENTER_VERTICAL, 0, 0);
    }
    //Función para recibir y presentar los datos del activity anterior
    private void recibirDatos() {
        try {
            datosStr = getIntent().getStringExtra("datosStr");
            datosJSON = new JSONArray(datosStr);
            textCedula.setText(datosJSON.getJSONObject(0).getString("c1"));
            textPropietario.setText(datosJSON.getJSONObject(0).getString("p1"));
            textCodPredioU.setText(getIntent().getStringExtra("codPredioU"));
            textFechaAct.setText(datosJSON.getJSONObject(0).getString("f2"));
            textPeriodoAct.setText(datosJSON.getJSONObject(0).getString("p2"));
            textDeudaAct.setText(datosJSON.getJSONObject(0).getString("d2"));
            textInteresAct.setText(datosJSON.getJSONObject(0).getString("i2"));
            textPlanillasAnt.setText(datosJSON.getJSONObject(0).getString("t3"));
            textPeriodoAnt.setText(datosJSON.getJSONObject(0).getString("p3"));
            textDeudaAnt.setText(datosJSON.getJSONObject(0).getString("d3"));
            textInteresAnt.setText(datosJSON.getJSONObject(0).getString("i3"));
            textDeudaTotal.setText(datosJSON.getJSONObject(0).getString("d4"));
            if (textDeudaTotal.getText().toString().equals("$0.00")) {
                toast.show();
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }
    //Función para establecer la acción de cada btn presionado
    @Override
    public void onClick(View view) {
```

```
        switch (view.getId()) {
            case R.id.imgbtn_Atras:
                onBackPressed();
                break;
            case R.id.imgbtn_Minimiza:
                moveTaskToBack(true);
                break;
        }
    }
}
```

- Archivo “activity_prediou_reporte.xml”

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_patente_consulta"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:weightSum="1"
    tools:context="ec.gob.chordeleg.chordemovil.PredioU_Reporte"
    android:background="@android:color/background_dark">
    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="45dp"
        android:id="@+id/Line_ActionBar">
        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            app:srcCompat="@drawable/atras32"
            android:id="@+id/imgbtn_Atras"
            android:background="@drawable/btn_cafe"
            android:paddingLeft="12dp" />
        <TextView
            android:text="CHORDELEGmóvil"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:id="@+id/text_NombreApp"
            android:background="@color/colorPrimary"
            android:textColor="@android:color/holo_orange_light"
            android:textAlignment="center"
            android:fontFamily="sans-serif"
            android:textSize="22sp"
            android:gravity="center_vertical"
            android:layout_weight="1" />
        <ImageButton
            android:layout_height="match_parent"
            app:srcCompat="@drawable/minimizar32"
            android:id="@+id/imgbtn_Minimiza"
            android:background="@drawable/btn_cafe"
            android:layout_width="wrap_content"
            android:paddingRight="12dp" />
    </LinearLayout>
    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:orientation="vertical" >
            <ImageView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                app:srcCompat="@drawable/urbano114"
                android:id="@+id/img_LogoPredioU" />
            <TextView
                android:text="DATOS GENERALES"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:id="@+id/eti_DatosGenerales"
                android:textAlignment="center"
                android:textSize="18sp"
```

```
    android:textColor="@android:color/background_dark"
    android:background="@drawable/btn_amarillopress"
    android:gravity="center_vertical" />
<TextView
    android:text="Propietario"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/eti_Propietario"
    android:layout_weight="1"
    android:textAlignment="center"
    android:textSize="18sp"
    android:gravity="center_vertical"
    android:textColor="@android:color/background_dark"
    android:background="@android:color/darker_gray" />
<TextView
    android:text="TextView"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/text_Propietario"
    android:textAlignment="center"
    android:textSize="18sp"
    android:layout_weight="1"
    android:gravity="center_vertical"
    android:background="@android:color/background_light" />
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="Cédula"
        android:layout_height="match_parent"
        android:id="@+id/eti_Cedula"
        android:layout_weight="1"
        android:textAlignment="center"
        android:background="@android:color/darker_gray"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:layout_width="match_parent"
        android:gravity="center_vertical" />
    <TextView
        android:text="Clave Predio"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_CodPredioU"
        android:layout_weight="1"
        android:textAlignment="center"
        android:background="@android:color/darker_gray"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:gravity="center_vertical" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="TextView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_Cedula"
        android:layout_weight="1"
        android:textAlignment="center"
        android:background="@android:color/background_light"
        android:textSize="18sp"
        android:gravity="center_vertical" />
    <TextView
        android:text="TextView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_CodPredioU"
        android:layout_weight="1"
        android:textAlignment="center"
        android:background="@android:color/background_light"
        android:textSize="18sp"
        android:gravity="center_vertical" />
```

```
</LinearLayout>

<TextView
   android:text="DEUDA ACTUAL"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:id="@+id/eti_DeudaActual"
   android:textAlignment="center"
   android:fontFamily="sans-serif"
   android:textColor="@android:color/background_dark"
   android:textSize="18sp"
   android:background="@drawable/btn_amarillopress"
   android:gravity="center_vertical" />
<LinearLayout
   android:orientation="horizontal"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
   <TextView
      android:text="Fecha Emisión"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/eti_FechaAct"
      android:layout_weight="1"
      android:textAlignment="center"
      android:textSize="18sp"
      android:textColor="@android:color/background_dark"
      android:background="@android:color/darker_gray" />
   <TextView
      android:text="Periodo"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/eti_PeriodoAct"
      android:layout_weight="1"
      android:textAlignment="center"
      android:textSize="18sp"
      android:textColor="@android:color/background_dark"
      android:background="@android:color/darker_gray" />
</LinearLayout>
<LinearLayout
   android:orientation="horizontal"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
   <TextView
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/text_FechaAct"
      android:layout_weight="1"
      android:textAlignment="center"
      android:textSize="18sp"
      android:background="@android:color/background_light"
      android:text="00/00/0000" />
   <TextView
      android:text="0000"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/text_PeriodoAct"
      android:layout_weight="1"
      android:textAlignment="center"
      android:textSize="18sp"
      android:background="@android:color/background_light" />
</LinearLayout>
<LinearLayout
   android:orientation="horizontal"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
   <TextView
      android:text="Valor"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/eti_DeudaAct"
      android:layout_weight="1"
      android:textAlignment="center"
      android:textSize="18sp"
      android:textColor="@android:color/background_dark"
      android:background="@android:color/darker_gray" />
```

```
    <TextView
        android:text="Desc/Recar"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_InteresAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_DeudaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
    <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_InteresAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
</LinearLayout>
<TextView
    android:text="DEUDA ANTERIOR"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/eti_DeudaAnterior"
    android:textAlignment="center"
    android:fontFamily="sans-serif"
    android:textColor="@android:color/background_dark"
    android:textSize="18sp"
    android:background="@drawable/btn_amarillopress"
    android:gravity="center_vertical" />
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="Planillas"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_PlanillasAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />

    <TextView
        android:text="Periodo"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_periodoAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
```

```
    <TextView
        android:text="0"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_PlanillasAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
    <TextView
        android:text="0000 - 0000"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_PeriodoAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="Valor"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_DeudaAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
    <TextView
        android:text="Interés"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_InteresAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_DeudaAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
    <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_InteresAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingTop="10dp">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
```

```
            android:id="@+id/eti_Espacio"
            android:layout_weight="1"
            android:textAlignment="center"
            android:textSize="18sp"
            android:background="@android:color/background_dark"
            android:textColor="@android:color/background_light" />
        <TextView
            android:text="TOTAL:"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:id="@+id/eti_DeudaTotal"
            android:layout_weight="1"
            android:textAlignment="center"
            android:textSize="18sp"
            android:background="@drawable/btn_amarillopress"
            android:textColor="@android:color/background_dark" />
        <TextView
            android:text="$0.00"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:id="@+id/text_DeudaTotal"
            android:layout_weight="1"
            android:textAlignment="center"
            android:textSize="18sp"
            android:background="@android:color/background_light"
            android:textColor="@android:color/background_dark" />
      </LinearLayout>
    </LinearLayout>
  </ScrollView>
</LinearLayout>
```

- Archivo “PredioR_Consulta.java”

```
package ec.gob.chordeleg.chordemovil;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Gravity;
import android.view.View;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.Toast;
import com.android.volley.DefaultRetryPolicy;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.ServerError;
import com.android.volley.TimeoutError;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;
import java.io.UnsupportedEncodingException;
public class PredioR_Consulta extends AppCompatActivity implements Response.Listener<String>,
Response.ErrorListener, View.OnClickListener{
  //Variables globales
  private ImageButton imgbtnAtras, imgbtnMinimiza;
  private Button btnConsultaPredioR;
  private EditText editCodPredioR;
  private RequestQueue requestQueue;
  private String url;
  private StringRequest stringRequest;
  private ProgressDialog dialogoEspera;
  private Toast toast;
  private Boolean conectado;
  //Función de inicio del activity
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
```

```
    setContentView(R.layout.activity_predior_consulta);
    inicailizar();
}
//Función para inicializar la configuración
private void inicailizar(){
    //Asociar cada btn,text,etc al view y habilitar el listener
    btnConsultaPredioR = (Button) findViewById(R.id.btn_ConsultaPredioR);
    btnConsultaPredioR.setOnClickListener(this);
    imgbtnAtras = (ImageButton) findViewById(R.id.imgbtn_Atras);
    imgbtnAtras.setOnClickListener(this);
    imgbtnMinimiza = (ImageButton) findViewById(R.id.imgbtn_Minimiza);
    imgbtnMinimiza.setOnClickListener(this);
    editCodPredioR = (EditText) findViewById(R.id.edit_CodPredioR);
    //Configurar las variables para consumir la APIrest del WS
    requestQueue = Volley.newRequestQueue(this);
    url = "http://ws.chordeleg.gob.ec:3001/predioR/";
    //Configurar las notificaciones
    toast = Toast.makeText(this, "", Toast.LENGTH_LONG);
    toast.setGravity(Gravity.CENTER_VERTICAL, 0, 0);
    //Configurar el dialogo de espera
    dialogoEspera = new ProgressDialog(this);
    dialogoEspera.setCancelable(false);
    dialogoEspera.setMessage("Procesando...");
    //Configurar el teclado para que inicie en la pantalla
   getWindow().setSoftInputMode(WindowManager.LayoutParams.SOFT_INPUT_STATE_ALWAYS_VISIBLE);
}
//Función para consultar los datos
private void consultar(String codPredioR){
    if (validaInternet(this)) {
        dialogoEspera.show();
        stringRequest = new StringRequest(Request.Method.GET, url + codPredioR, this, this);
        stringRequest.setRetryPolicy(new DefaultRetryPolicy(500, 3, 1));
        requestQueue.add(stringRequest);
    } else {
        toast.setText("Revice su conexión a internet...");
        toast.show();
    }
}
//Función para validar la conexión a internet
public boolean validaInternet(Context context) {
    conectado = false;
    ConnectivityManager connec = (ConnectivityManager) context.getSystemService(Context.CONNECTIVITY_SERVICE);
    //Obtiene todas las redes (móviles y wifi)
    NetworkInfo[] redes = connec.getAllNetworkInfo();
    for (int i = 0; i < redes.length; i++) {
        if (redes[i].getState() == NetworkInfo.State.CONNECTED) {
            conectado = true;
            i = redes.length;
        }
    }
    return conectado;
}
//Función para establecer la acción de cada btn presionado
@Override
public void onClick(View view) {
    switch (view.getId()) {
        case R.id.btn_ConsultaPredioR:
            if (!editCodPredioR.getText().toString().equals("")) {
                consultar(editCodPredioR.getText().toString());
            } else {
                toast.setText("Ingrese su clave de Predio Rural...");
                toast.show();
            }
            break;
        case R.id.imgbtn_Atras:
            onBackPressed();
            break;
        case R.id.imgbtn_Minimiza:
            moveTaskToBack(true);
            break;
    }
}
//Función para establecer la acción si se obtiene respuesta del WS
@Override
public void onResponse(String response) {
```

```
            try {
                response = new String(response.getBytes("ISO-8859-1"), "UTF-8");
                Intent intentReportePat = new Intent(this, PredioR_Reporte.class);
                intentReportePat.putExtra("codPredioR", editCodPredioR.getText().toString());
                intentReportePat.putExtra("datosStr", response);
                dialogoEspera.dismiss();
                startActivity(intentReportePat);
            } catch (UnsupportedEncodingException e) {
                e.printStackTrace();
            }
        }
        //Función para establecer la acción si se obtiene un error del WS
        @Override
        public void onErrorResponse(VolleyError error) {
            dialogoEspera.dismiss();
            if (error instanceof ServerError) {
                toast.setText("No existe el Predio ingresado...");
                toast.show();
            } else if (error instanceof TimeoutError) {
                toast.setText("Tiempo de espera agotado, vuelva a intentarlo...");
                toast.show();
            } else {
                toast.setText("Lo sentimos, servicio no disponible. Inténtelo más tarde...");
                toast.show();
            }
        }
    }
```

- Archivo “activity_predior_consulta.xml”

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_patente_consulta"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:weightSum="1"
    tools:context="ec.gob.chordeleg.chordemovil.PredioR_Reporte"
    android:background="@android:color/background_dark">


    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="45dp"
        android:id="@+id/Line_ActionBar">
        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            app:srcCompat="@drawable/atras32"
            android:id="@+id/imgbtn_Atras"
            android:background="@drawable/btn_cafe"
            android:paddingLeft="12dp" />
        <TextView
            android:text="CHORDELEGmóvil"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:id="@+id/text_NombreApp"
            android:background="@color/colorPrimary"
            android:textColor="@android:color/holo_orange_light"
            android:textAlignment="center"
            android:fontFamily="sans-serif"
            android:textSize="22sp"
            android:gravity="center_vertical"
            android:layout_weight="1" />
        <ImageButton
            android:layout_height="match_parent"
            app:srcCompat="@drawable/minimizar32"
            android:id="@+id/imgbtn_Minimiza"
            android:background="@drawable/btn_cafe"
            android:layout_width="wrap_content"
            android:paddingRight="12dp" />
```

```
</LinearLayout>
<ScrollView
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical" >
        <ImageView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            app:srcCompat="@drawable/rural114"
            android:id="@+id/img_LogoPredioR" />
        <TextView
            android:text="DATOS GENERALES"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/eti_DatosGenerales"
            android:textAlignment="center"
            android:textSize="18sp"
            android:textColor="@android:color/background_dark"
            android:background="@drawable/btn_amarillopress"
            android:gravity="center_vertical" />
        <TextView
            android:text="Propietario"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/eti_Propietario"
            android:layout_weight="1"
            android:textAlignment="center"
            android:textSize="18sp"
            android:gravity="center_vertical"
            android:textColor="@android:color/background_dark"
            android:background="@android:color/darker_gray" />
        <TextView
            android:text="TextView"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:id="@+id/text_Propietario"
            android:textAlignment="center"
            android:textSize="18sp"
            android:layout_weight="1"
            android:gravity="center_vertical"
            android:background="@android:color/background_light" />
        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="match_parent">
            <TextView
                android:text="Cédula"
                android:layout_height="match_parent"
                android:id="@+id/eti_Cedula"
                android:layout_weight="1"
                android:textAlignment="center"
                android:background="@android:color/darker_gray"
                android:textSize="18sp"
                android:textColor="@android:color/background_dark"
                android:layout_width="match_parent"
                android:gravity="center_vertical" />
            <TextView
                android:text="Clave Predio"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/eti_CodPredioR"
                android:layout_weight="1"
                android:textAlignment="center"
                android:background="@android:color/darker_gray"
                android:textSize="18sp"
                android:textColor="@android:color/background_dark"
                android:gravity="center_vertical" />
        </LinearLayout>
        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="match_parent">
```

```
    <TextView
        android:text="TextView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_Cedula"
        android:layout_weight="1"
        android:textAlignment="center"
        android:background="@android:color/background_light"
        android:textSize="18sp"
        android:gravity="center_vertical" />
    <TextView
        android:text="TextView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_CodPredioR"
        android:layout_weight="1"
        android:textAlignment="center"
        android:background="@android:color/background_light"
        android:textSize="18sp"
        android:gravity="center_vertical" />
</LinearLayout>
<TextView
    android:text="DEUDA ACTUAL"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/eti_DeudaActual"
    android:textAlignment="center"
    android:fontFamily="sans-serif"
    android:textColor="@android:color/background_dark"
    android:textSize="18sp"
    android:background="@drawable/btn_amarillopress"
    android:gravity="center_vertical" />
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="Fecha Emisión"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_FechaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
    <TextView
        android:text="Periodo"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_PeriodoAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_FechaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light"
        android:text="00/00/0000" />
    <TextView
        android:text="0000"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_PeriodoAct"
```

```
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
    </LinearLayout>
    <LinearLayout
      android:orientation="horizontal"
      android:layout_width="match_parent"
      android:layout_height="match_parent">
      <TextView
        android:text="Valor"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_DeudaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
      <TextView
        android:text="Desc/Recar"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_InteresAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
    </LinearLayout>
    <LinearLayout
      android:orientation="horizontal"
      android:layout_width="match_parent"
      android:layout_height="match_parent">
      <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_DeudaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
      <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_InteresAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
    </LinearLayout>
    <TextView
      android:text="DEUDA ANTERIOR"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:id="@+id/eti_DeudaAnterior"
      android:textAlignment="center"
      android:fontFamily="sans-serif"
      android:textColor="@android:color/background_dark"
      android:textSize="18sp"
      android:background="@drawable/btn_amarillopress"
      android:gravity="center_vertical" />
    <LinearLayout
      android:orientation="horizontal"
      android:layout_width="match_parent"
      android:layout_height="match_parent">
      <TextView
        android:text="Planillas"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_PlanillasAnt"
        android:layout_weight="1"
        android:textAlignment="center"
```

```
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
    <TextView
        android:text="Periodo"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_periodoAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="0"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_PlanillasAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
    <TextView
        android:text="0000 - 0000"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_PeriodoAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="Valor"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_DeudaAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
    <TextView
        android:text="Interés"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_InteresAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_DeudaAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
```

```
            <TextView
                android:text="$0.00"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/text_InteresAnt"
                android:layout_weight="1"
                android:textAlignment="center"
                android:textSize="18sp"
                android:background="@android:color/background_light" />
        </LinearLayout>
        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:paddingTop="10dp">
            <TextView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:id="@+id/eti_Espacio"
                android:layout_weight="1"
                android:textAlignment="center"
                android:textSize="18sp"
                android:background="@android:color/background_dark"
                android:textColor="@android:color/background_light" />
            <TextView
                android:text="TOTAL:"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/eti_DeudaTotal"
                android:layout_weight="1"
                android:textAlignment="center"
                android:textSize="18sp"
                android:background="@drawable/btn_amarillopress"
                android:textColor="@android:color/background_dark" />
            <TextView
                android:text="$0.00"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/text_DeudaTotal"
                android:layout_weight="1"
                android:textAlignment="center"
                android:textSize="18sp"
                android:background="@android:color/background_light"
                android:textColor="@android:color/background_dark" />
        </LinearLayout>
    </LinearLayout>
  </ScrollView>
</LinearLayout>
```

- Archivo “PredioR_Reporte.java”

```
package ec.gob.chordeleg.chordemovil;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Gravity;
import android.view.View;
import android.widget.ImageButton;
import android.widget.TextView;
import android.widget.Toast;
import org.json.JSONArray;
import org.json.JSONException;
public class PredioR_Reporte extends AppCompatActivity implements View.OnClickListener {
    //Variables globales
    private ImageButton imgbtnAtras, imgbtnMinimiza;
    private TextView textCedula, textPropietario, textCodPredioR,
            textFechaAct, textPeriodoAct, textDeudaAct, textInteresAct,
            textPlanillasAnt, textPeriodoAnt, textDeudaAnt, textInteresAnt,
            textDeudaTotal;
    private String datosStr;
    private JSONArray datosJSON;
    private Toast toast;
    //Función de inicio del activity
    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_predior_reporte);
        inicializar();
        recibirDatos();
    }
    //Función para inicializar la configuración
    private void inicializar() {
        //Asociar cada btn,text,etc al view y habilitar el listener
        imgbtnAtras = (ImageButton) findViewById(R.id.imgbtn_Atras);
        imgbtnAtras.setOnClickListener(this);
        imgbtnMinimiza = (ImageButton) findViewById(R.id.imgbtn_Minimiza);
        imgbtnMinimiza.setOnClickListener(this);
        textCedula = (TextView) findViewById(R.id.text_Cedula);
        textPropietario = (TextView) findViewById(R.id.text_Propietario);
        textCodPredioR = (TextView) findViewById(R.id.text_CodPredioR);
        textFechaAct = (TextView) findViewById(R.id.text_FechaAct);
        textPeriodoAct = (TextView) findViewById(R.id.text_PeriodoAct);
        textDeudaAct = (TextView) findViewById(R.id.text_DeudaAct);
        textInteresAct = (TextView) findViewById(R.id.text_InteresAct);
        textPlanillasAnt = (TextView) findViewById(R.id.text_PlanillasAnt);
        textPeriodoAnt = (TextView) findViewById(R.id.text_PeriodoAnt);
        textDeudaAnt = (TextView) findViewById(R.id.text_DeudaAnt);
        textInteresAnt = (TextView) findViewById(R.id.text_InteresAnt);
        textDeudaTotal = (TextView) findViewById(R.id.text_DeudaTotal);
        //Configurar la notificación de deuda pendiente
        toast = Toast.makeText(this, "No tiene deudas pendientes...", Toast.LENGTH_LONG);
        toast.setGravity(Gravity.CENTER_VERTICAL, 0, 0);
    }
    //Función para recibir y presentar los datos del activity anterior
    private void recibirDatos() {
        try {
            datosStr = getIntent().getStringExtra("datosStr");
            datosJSON = new JSONArray(datosStr);
            textCedula.setText(datosJSON.getJSONObject(0).getString("c1"));
            textPropietario.setText(datosJSON.getJSONObject(0).getString("p1"));
            textCodPredioR.setText(getIntent().getStringExtra("codPredioR"));
            textFechaAct.setText(datosJSON.getJSONObject(0).getString("f2"));
            textPeriodoAct.setText(datosJSON.getJSONObject(0).getString("p2"));
            textDeudaAct.setText(datosJSON.getJSONObject(0).getString("d2"));
            textInteresAct.setText(datosJSON.getJSONObject(0).getString("i2"));
            textPlanillasAnt.setText(datosJSON.getJSONObject(0).getString("t3"));
            textPeriodoAnt.setText(datosJSON.getJSONObject(0).getString("p3"));
            textDeudaAnt.setText(datosJSON.getJSONObject(0).getString("d3"));
            textInteresAnt.setText(datosJSON.getJSONObject(0).getString("i3"));
            textDeudaTotal.setText(datosJSON.getJSONObject(0).getString("d4"));
            if (textDeudaTotal.getText().toString().equals("$0.00")) {
                toast.show();
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }
    //Función para establecer la acción de cada btn presionado
    @Override
    public void onClick(View view) {
        switch (view.getId()) {
            case R.id.imgbtn_Atras:
                onBackPressed();
                break;
            case R.id.imgbtn_Minimiza:
                moveTaskToBack(true);
                break;
        }
    }
}
```

- Archivo “activity_predior_reporte.xml”

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_patente_consulta"
    android:layout_width="match_parent"
```

```
android:layout_height="match_parent"
android:orientation="vertical"
android:weightSum="1"
tools:context="ec.gob.chordeleg.chordemovil.PredioR_Reporte"
android:background="@android:color/background_dark">
<LinearLayout
   android:orientation="horizontal"
   android:layout_width="match_parent"
   android:layout_height="45dp"
   android:id="@+id/Line_ActionBar">
   <ImageButton
      android:layout_width="wrap_content"
      android:layout_height="match_parent"
      app:srcCompat="@drawable/atras32"
      android:id="@+id/imgbtn_Atras"
      android:background="@drawable/btn_cafe"
      android:paddingLeft="12dp" />
   <TextView
      android:text="CHORDELEGmóvil"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/text_NombreApp"
      android:background="@color/colorPrimary"
      android:textColor="@android:color/holo_orange_light"
      android:textAlignment="center"
      android:fontFamily="sans-serif"
      android:textSize="22sp"
      android:gravity="center_vertical"
      android:layout_weight="1" />
   <ImageButton
      android:layout_height="match_parent"
      app:srcCompat="@drawable/minimizar32"
      android:id="@+id/imgbtn_Minimiza"
      android:background="@drawable/btn_cafe"
      android:layout_width="wrap_content"
      android:paddingRight="12dp" />
</LinearLayout>

<ScrollView
   android:layout_width="match_parent"
   android:layout_height="match_parent">
   <LinearLayout
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:orientation="vertical" >
      <ImageView
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         app:srcCompat="@drawable/rural114"
         android:id="@+id/img_LogoPredioR" />
      <TextView
         android:text="DATOS GENERALES"
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:id="@+id/eti_DatosGenerales"
         android:textAlignment="center"
         android:textSize="18sp"
         android:textColor="@android:color/background_dark"
         android:background="@drawable/btn_amarillopress"
         android:gravity="center_vertical" />
      <TextView
         android:text="Propietario"
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:id="@+id/eti_Propietario"
         android:layout_weight="1"
         android:textAlignment="center"
         android:textSize="18sp"
         android:gravity="center_vertical"
         android:textColor="@android:color/background_dark"
         android:background="@android:color/darker_gray" />
      <TextView
         android:text="TextView"
         android:layout_width="match_parent"
         android:layout_height="match_parent"
```

```
    android:id="@+id/text_Propietario"
    android:textAlignment="center"
    android:textSize="18sp"
    android:layout_weight="1"
    android:gravity="center_vertical"
    android:background="@android:color/background_light" />
  <LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
      android:text="Cédula"
      android:layout_height="match_parent"
      android:id="@+id/eti_Cedula"
      android:layout_weight="1"
      android:textAlignment="center"
      android:background="@android:color/darker_gray"
      android:textSize="18sp"
      android:textColor="@android:color/background_dark"
      android:layout_width="match_parent"
      android:gravity="center_vertical" />
    <TextView
      android:text="Clave Predio"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/eti_CodPredioR"
      android:layout_weight="1"
      android:textAlignment="center"
      android:background="@android:color/darker_gray"
      android:textSize="18sp"
      android:textColor="@android:color/background_dark"
      android:gravity="center_vertical" />
  </LinearLayout>
  <LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
      android:text="TextView"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/text_Cedula"
      android:layout_weight="1"
      android:textAlignment="center"
      android:background="@android:color/background_light"
      android:textSize="18sp"
      android:gravity="center_vertical" />
    <TextView
      android:text="TextView"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/text_CodPredioR"
      android:layout_weight="1"
      android:textAlignment="center"
      android:background="@android:color/background_light"
      android:textSize="18sp"
      android:gravity="center_vertical" />
  </LinearLayout>
  <TextView
    android:text="DEUDA ACTUAL"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/eti_DeudaActual"
    android:textAlignment="center"
    android:fontFamily="sans-serif"
    android:textColor="@android:color/background_dark"
    android:textSize="18sp"
    android:background="@drawable/btn_amarillopress"
    android:gravity="center_vertical" />
  <LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
      android:text="Fecha Emisión"
```

```
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_FechaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
    <TextView
        android:text="Periodo"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_PeriodoAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_FechaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light"
        android:text="00/00/0000" />
    <TextView
        android:text="0000"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_PeriodoAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="Valor"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_DeudaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
    <TextView
        android:text="Desc/Recar"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_InteresAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
```

```
        android:id="@+id/text_DeudaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
    <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_InteresAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
</LinearLayout>
<TextView
    android:text="DEUDA ANTERIOR"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/eti_DeudaAnterior"
    android:textAlignment="center"
    android:fontFamily="sans-serif"
    android:textColor="@android:color/background_dark"
    android:textSize="18sp"
    android:background="@drawable/btn_amarillopress"
    android:gravity="center_vertical" />
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="Planillas"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_PlanillasAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
    <TextView
        android:text="Periodo"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_periodoAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="0"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_PlanillasAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
    <TextView
        android:text="0000 - 0000"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_PeriodoAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
</LinearLayout>
<LinearLayout
```

```
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="Valor"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_DeudaAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
    <TextView
        android:text="Interés"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_InteresAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_DeudaAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
    <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_InteresAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingTop="10dp">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/eti_Espacio"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_dark"
        android:textColor="@android:color/background_light" />
    <TextView
        android:text="TOTAL:"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_DeudaTotal"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@drawable/btn_amarillopress"
        android:textColor="@android:color/background_dark" />
    <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
```

```
            android:id="@+id/text_DeudaTotal"
            android:layout_weight="1"
            android:textAlignment="center"
            android:textSize="18sp"
            android:background="@android:color/background_light"
            android:textColor="@android:color/background_dark" />
        </LinearLayout>
    </LinearLayout>
  </ScrollView>
</LinearLayout>
```

- Archivo “Agua_Consulta.java”

```
package ec.gob.chordeleg.chordemovil;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Gravity;
import android.view.View;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.Toast;
import com.android.volley.DefaultRetryPolicy;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.ServerError;
import com.android.volley.TimeoutError;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;
import java.io.UnsupportedEncodingException;
public class Agua_Consulta extends AppCompatActivity implements Response.Listener<String>, Response.ErrorListener,
View.OnClickListener{
    //Variables globales
    private ImageButton imgbtnAtras, imgbtnMinimiza;
    private Button btnConsultaAgua;
    private EditText editCodAgua;
    private RequestQueue requestQueue;
    private String url;
    private StringRequest stringRequest;
    private ProgressDialog dialogoEspera;
    private Toast toast;
    private Boolean conectado;
    //Función de inicio del activity
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_agua_consulta);
        inicailizar();
    }
    //Función para inicializar la configuración
    private void inicailizar(){
        //Asociar cada btn,text,etc al view y habilitar el listener
        btnConsultaAgua = (Button) findViewById(R.id.btn_ConsultaAgua);
        btnConsultaAgua.setOnClickListener(this);
        imgbtnAtras = (ImageButton) findViewById(R.id.imgbtn_Atras);
        imgbtnAtras.setOnClickListener(this);
        imgbtnMinimiza = (ImageButton) findViewById(R.id.imgbtn_Minimiza);
        imgbtnMinimiza.setOnClickListener(this);
        editCodAgua = (EditText) findViewById(R.id.edit_CodAgua);
        //Configurar las variables para consumir la APIrest del WS
        requestQueue = Volley.newRequestQueue(this);
        url = "http://ws.chordeleg.gob.ec:3001/agua/";
        //Configurar las notificaciones
        toast = Toast.makeText(this, "", Toast.LENGTH_LONG);
        toast.setGravity(Gravity.CENTER_VERTICAL, 0, 0);
        //Configurar el dialogo de espera
```

```
    dialogoEspera = new ProgressDialog(this);
    dialogoEspera.setCancelable(false);
    dialogoEspera.setMessage("Procesando...");
    //Configurar el teclado para que inicie en la pantalla
    getWindow().setSoftInputMode(WindowManager.LayoutParams.SOFT_INPUT_STATE_ALWAYS_VISIBLE);
}
//Función para consultar los datos
private void consultar(String codAgua){
    if (validaInternet(this)) {
        dialogoEspera.show();
        stringRequest = new StringRequest(Request.Method.GET, url + codAgua, this, this);
        stringRequest.setRetryPolicy(new DefaultRetryPolicy(500, 3, 1));
        requestQueue.add(stringRequest);
    } else {
        toast.setText("Revice su conexión a internet...");
        toast.show();
    }
}
//Función para validar la conexión a internet
public boolean validaInternet(Context context) {
    conectado = false;
    ConnectivityManager connec = (ConnectivityManager) context.getSystemService(Context.CONNECTIVITY_SERVICE);
    //Obtiene todas las redes (móviles y wifi)
    NetworkInfo[] redes = connec.getAllNetworkInfo();
    for (int i = 0; i < redes.length; i++) {
        if (redes[i].getState() == NetworkInfo.State.CONNECTED) {
            conectado = true;
            i = redes.length;
        }
    }
    return conectado;
}
//Función para establecer la acción de cada btn presionado
@Override
public void onClick(View view) {
    switch (view.getId()) {
        case R.id.btn_ConsultaAgua:
            if (!editCodAgua.getText().toString().equals("")) {
                consultar(editCodAgua.getText().toString());
            } else {
                toast.setText("Ingrese su cuenta de Medidor...");
                toast.show();
            }
            break;
        case R.id.imgbtn_Atras:
            onBackPressed();
            break;
        case R.id.imgbtn_Minimiza:
            moveTaskToBack(true);
            break;
    }
}
//Función para establecer la acción si se obtiene respuesta del WS
@Override
public void onResponse(String response) {
    try {
        response = new String(response.getBytes("ISO-8859-1"), "UTF-8");
        Intent intentReporteAgua = new Intent(this, Agua_Reporte.class);
        intentReporteAgua.putExtra("codAgua", editCodAgua.getText().toString());
        intentReporteAgua.putExtra("datosStr", response);
        dialogoEspera.dismiss();
        startActivity(intentReporteAgua);
    } catch (UnsupportedEncodingException e) {
        e.printStackTrace();
    }
}
//Función para establecer la acción si se obtiene un error del WS
@Override
public void onErrorResponse(VolleyError error) {
    dialogoEspera.dismiss();
    if (error instanceof ServerError) {
        toast.setText("No existe el Medidor ingresado...");
        toast.show();
    } else if (error instanceof TimeoutError) {
        toast.setText("Tiempo de espera agotado, vuelva a intentarlo...");
```

```
            toast.show();
        } else {
            toast.setText("Lo sentimos, servicio no disponible. Inténtelo más tarde...");
            toast.show();
        }
    }
}
```

- Archivo “activity_agua_consulta.xml”

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_agua_consulta"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:weightSum="1"
    tools:context="ec.gob.chordeleg.chordemovil.Agua_Consulta"
    android:background="@android:color/background_dark">
    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:background="@color/colorPrimary"
        android:layout_height="45dp"
        android:id="@+id/LI_actionBar">

        <ImageButton
            android:layout_height="match_parent"
            app:srcCompat="@drawable/atras32"
            android:id="@+id/imgbtn_Atras"
            android:background="@drawable/btn_cafe"
            android:layout_width="wrap_content"
            android:paddingLeft="12dp" />
        <TextView
            android:text="CHORDELEGmóvil"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:id="@+id/eti_NombreApp"
            android:background="@color/colorPrimary"
            android:textColor="@android:color/holo_orange_light"
            android:textAlignment="center"
            android:fontFamily="sans-serif"
            android:textSize="22sp"
            android:gravity="center_vertical"
            android:layout_weight="1" />
        <ImageButton
            android:layout_height="match_parent"
            app:srcCompat="@drawable/minimizar32"
            android:id="@+id/imgbtn_Minimiza"
            android:background="@drawable/btn_cafe"
            android:layout_width="wrap_content"
            android:paddingRight="12dp" />
    </LinearLayout>
    <ImageView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:srcCompat="@drawable/agua114"
        android:id="@+id/img_LogoAgua"
        android:background="@android:color/background_dark" />
    <TextView
        android:text="CUENTA DE MEDIDOR"
        android:layout_width="match_parent"
        android:id="@+id/eti_CodAgua"
        android:gravity="center_vertical|center_horizontal"
        android:background="@drawable/btn_amarillopress"
        android:fontFamily="sans-serif"
        android:textSize="22sp"
        android:textColor="@android:color/background_dark"
        android:layout_height="38dp" />
    <EditText
        android:layout_width="match_parent"
        android:id="@+id/edit_CodAgua"
```

```
        android:hint="Ingrese su cuenta"
        android:textSize="22sp"
        android:textAlignment="center"
        android:fontFamily="sans-serif"
        android:textAllCaps="false"
        android:inputType="number"
        android:background="@drawable/edit_blanco"
        android:layout_gravity="center_horizontal"
        android:ems="10"
        android:maxLength="5"
        android:layout_height="42dp" />
    <Button
        android:layout_width="186dp"
        android:layout_height="45dp"
        android:id="@+id/btn_ConsultaAgua"
        android:fontFamily="sans-serif"
        android:background="@drawable/btn_amarillo"
        android:textColor="@android:color/background_dark"
        android:textSize="22sp"
        android:textAllCaps="false"
        android:layout_gravity="center_horizontal"
        android:text="CONSULTAR"
        android:drawableLeft="@drawable/lupa32"
        android:paddingLeft="8dp"
        android:textStyle="normal|bold"
        android:layout_marginTop="5dp" />
</LinearLayout>
```

- Archivo “Agua_Reporte.java”

```
package ec.gob.chordeleg.chordemovil;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Gravity;
import android.view.View;
import android.widget.ImageButton;
import android.widget.TextView;
import android.widget.Toast;
import org.json.JSONArray;
import org.json.JSONException;
public class Agua_Reporte extends AppCompatActivity implements View.OnClickListener {
    //Variables globales
    private ImageButton imgbtnAtras, imgbtnMinimiza;
    private TextView textCedula, textPropietario, textCodAgua,
            textFechaAct, textPeriodoAct, textDeudaAct, textInteresAct,
            textPlanillasAnt, textPeriodoAnt, textDeudaAnt, textInteresAnt,
            textDeudaTotal;
    private String datosStr;
    private JSONArray datosJSON;
    private Toast toast;
    //Función de inicio del activity
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_agua_reporte);
        inicializar();
        recibirDatos();
    }
    //Función para inicializar la configuración
    private void inicializar() {
        //Asociar cada btn,text,etc al view y habilitar el listener
        imgbtnAtras = (ImageButton) findViewById(R.id.imgbtn_Atras);
        imgbtnAtras.setOnClickListener(this);
        imgbtnMinimiza = (ImageButton) findViewById(R.id.imgbtn_Minimiza);
        imgbtnMinimiza.setOnClickListener(this);
        textCedula = (TextView) findViewById(R.id.text_Cedula);
        textPropietario = (TextView) findViewById(R.id.text_Propietario);
        textCodAgua = (TextView) findViewById(R.id.text_CodAgua);
        textFechaAct = (TextView) findViewById(R.id.text_FechaAct);
        textPeriodoAct = (TextView) findViewById(R.id.text_PeriodoAct);
        textDeudaAct = (TextView) findViewById(R.id.text_DeudaAct);
        textInteresAct = (TextView) findViewById(R.id.text_InteresAct);
        textPlanillasAnt = (TextView) findViewById(R.id.text_PlanillasAnt);
        textPeriodoAnt = (TextView) findViewById(R.id.text_PeriodoAnt);
```

```
        textDeudaAnt = (TextView) findViewById(R.id.text_DeudaAnt);
        textInteresAnt = (TextView) findViewById(R.id.text_InteresAnt);
        textDeudaTotal = (TextView) findViewById(R.id.text_DeudaTotal);
        //Configurar la notificación de deuda pendiente
        toast = Toast.makeText(Agua_Reporte.this, "No tiene deudas pendientes...", Toast.LENGTH_LONG);
        toast.setGravity(Gravity.CENTER_VERTICAL, 0, 0);
    }
    //Función para recibir y presentar los datos del activity anterior
    private void recibirDatos() {
        try {
            datosStr = getIntent().getStringExtra("datosStr");
            datosJSON = new JSONArray(datosStr);
            textCedula.setText(datosJSON.getJSONObject(0).getString("c1"));
            textPropietario.setText(datosJSON.getJSONObject(0).getString("p1"));
            textCodAgua.setText(getIntent().getStringExtra("codAgua"));
            textFechaAct.setText(datosJSON.getJSONObject(0).getString("f2"));
            textPeriodoAct.setText(datosJSON.getJSONObject(0).getString("p2"));
            textDeudaAct.setText(datosJSON.getJSONObject(0).getString("d2"));
            textInteresAct.setText(datosJSON.getJSONObject(0).getString("i2"));
            textPlanillasAnt.setText(datosJSON.getJSONObject(0).getString("t3"));
            textPeriodoAnt.setText(datosJSON.getJSONObject(0).getString("p3"));
            textDeudaAnt.setText(datosJSON.getJSONObject(0).getString("d3"));
            textInteresAnt.setText(datosJSON.getJSONObject(0).getString("i3"));
            textDeudaTotal.setText(datosJSON.getJSONObject(0).getString("d4"));
            if (textDeudaTotal.getText().toString().equals("$0.00")) {
                toast.show();
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }
    //Función para establecer la acción de cada btn presionado
    @Override
    public void onClick(View view) {
        switch (view.getId()) {
            case R.id.imgbtn_Atras:
                onBackPressed();
                break;
            case R.id.imgbtn_Minimiza:
                moveTaskToBack(true);
                break;
        }
    }
}
```

- Archivo “activity_agua_reporte.xml”

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_agua_consulta"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:weightSum="1"
    tools:context="ec.gob.chordeleg.chordemovil.Agua_Reporte"
    android:background="@android:color/background_dark">
    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="45dp"
        android:id="@+id/Line_ActionBar">
        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            app:srcCompat="@drawable/atras32"
            android:id="@+id/imgbtn_Atras"
            android:background="@drawable/btn_cafe"
            android:paddingLeft="12dp" />
        <TextView
            android:text="CHORDELEGmóvil"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
```

```
        android:id="@+id/text_NombreApp"
        android:background="@color/colorPrimary"
        android:textColor="@android:color/holo_orange_light"
        android:textAlignment="center"
        android:fontFamily="sans-serif"
        android:textSize="22sp"
        android:gravity="center_vertical"
        android:layout_weight="1" />
    <ImageButton
        android:layout_height="match_parent"
        app:srcCompat="@drawable/minimizar32"
        android:id="@+id/imgbtn_Minimiza"
        android:background="@drawable/btn_cafe"
        android:layout_width="wrap_content"
        android:paddingRight="12dp" />
</LinearLayout>
<ScrollView
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical" >
        <ImageView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            app:srcCompat="@drawable/agua114"
            android:id="@+id/img_LogoAgua" />
        <TextView
            android:text="DATOS GENERALES"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/eti_DatosGenerales"
            android:textAlignment="center"
            android:textSize="18sp"
            android:textColor="@android:color/background_dark"
            android:background="@drawable/btn_amarillopress"
            android:gravity="center_vertical" />
        <TextView
            android:text="Propietario"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/eti_Propietario"
            android:layout_weight="1"
            android:textAlignment="center"
            android:textSize="18sp"
            android:gravity="center_vertical"
            android:textColor="@android:color/background_dark"
            android:background="@android:color/darker_gray" />
        <TextView
            android:text="TextView"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:id="@+id/text_Propietario"
            android:textAlignment="center"
            android:textSize="18sp"
            android:layout_weight="1"
            android:gravity="center_vertical"
            android:background="@android:color/background_light" />
        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="match_parent">
            <TextView
                android:text="Cédula"
                android:layout_height="match_parent"
                android:id="@+id/eti_Cedula"
                android:layout_weight="1"
                android:textAlignment="center"
                android:background="@android:color/darker_gray"
                android:textSize="18sp"
                android:textColor="@android:color/background_dark"
                android:layout_width="match_parent"
                android:gravity="center_vertical" />
            <TextView
```

```
        android:text="Cta. Medidor"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_CodAgua"
        android:layout_weight="1"
        android:textAlignment="center"
        android:background="@android:color/darker_gray"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:gravity="center_vertical" />
    </LinearLayout>
    <LinearLayout
      android:orientation="horizontal"
      android:layout_width="match_parent"
      android:layout_height="match_parent">
      <TextView
        android:text="TextView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_Cedula"
        android:layout_weight="1"
        android:textAlignment="center"
        android:background="@android:color/background_light"
        android:textSize="18sp"
        android:gravity="center_vertical" />
      <TextView
        android:text="TextView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_CodAgua"
        android:layout_weight="1"
        android:textAlignment="center"
        android:background="@android:color/background_light"
        android:textSize="18sp"
        android:gravity="center_vertical" />
    </LinearLayout>
    <TextView
      android:text="DEUDA ACTUAL"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:id="@+id/eti_DeudaActual"
      android:textAlignment="center"
      android:fontFamily="sans-serif"
      android:textColor="@android:color/background_dark"
      android:textSize="18sp"
      android:background="@drawable/btn_amarillopress"
      android:gravity="center_vertical" />
    <LinearLayout
      android:orientation="horizontal"
      android:layout_width="match_parent"
      android:layout_height="match_parent">
      <TextView
        android:text="Fecha Emisión"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_FechaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
      <TextView
        android:text="Periodo"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_PeriodoAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
    </LinearLayout>
    <LinearLayout
      android:orientation="horizontal"
      android:layout_width="match_parent"
```

```
    android:layout_height="match_parent">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_FechaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light"
        android:text="00/00/0000" />
    <TextView
        android:text="0000"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_PeriodoAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="Valor"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_DeudaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
    <TextView
        android:text="Interés"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_InteresAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_DeudaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
    <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_InteresAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
</LinearLayout>
<TextView
    android:text="DEUDA ANTERIOR"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/eti_DeudaAnterior"
    android:textAlignment="center"
    android:fontFamily="sans-serif"
```

```
    android:textColor="@android:color/background_dark"
    android:textSize="18sp"
    android:background="@drawable/btn_amarillopress"
    android:gravity="center_vertical" />
  <LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
      android:text="Planillas"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/eti_PlanillasAnt"
      android:layout_weight="1"
      android:textAlignment="center"
      android:textSize="18sp"
      android:textColor="@android:color/background_dark"
      android:background="@android:color/darker_gray" />
    <TextView
      android:text="Periodo"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/eti_periodoAnt"
      android:layout_weight="1"
      android:textAlignment="center"
      android:textSize="18sp"
      android:textColor="@android:color/background_dark"
      android:background="@android:color/darker_gray" />
  </LinearLayout>
  <LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
      android:text="0"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/text_PlanillasAnt"
      android:layout_weight="1"
      android:textAlignment="center"
      android:textSize="18sp"
      android:background="@android:color/background_light" />
    <TextView
      android:text="0000 - 0000"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/text_PeriodoAnt"
      android:layout_weight="1"
      android:textAlignment="center"
      android:textSize="18sp"
      android:background="@android:color/background_light" />
  </LinearLayout>
  <LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
      android:text="Valor"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/eti_DeudaAnt"
      android:layout_weight="1"
      android:textAlignment="center"
      android:textSize="18sp"
      android:textColor="@android:color/background_dark"
      android:background="@android:color/darker_gray" />
    <TextView
      android:text="Interés"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/eti_InteresAnt"
      android:layout_weight="1"
      android:textAlignment="center"
      android:textSize="18sp"
      android:textColor="@android:color/background_dark"
```

```
            android:background="@android:color/darker_gray" />
        </LinearLayout>
        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="match_parent">
            <TextView
                android:text="$0.00"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/text_DeudaAnt"
                android:layout_weight="1"
                android:textAlignment="center"
                android:textSize="18sp"
                android:background="@android:color/background_light" />
            <TextView
                android:text="$0.00"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/text_InteresAnt"
                android:layout_weight="1"
                android:textAlignment="center"
                android:textSize="18sp"
                android:background="@android:color/background_light" />
        </LinearLayout>
        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:paddingTop="10dp">
            <TextView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:id="@+id/eti_Espacio"
                android:layout_weight="1"
                android:textAlignment="center"
                android:textSize="18sp"
                android:background="@android:color/background_dark"
                android:textColor="@android:color/background_light" />
            <TextView
                android:text="TOTAL:"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/eti_DeudaTotal"
                android:layout_weight="1"
                android:textAlignment="center"
                android:textSize="18sp"
                android:background="@drawable/btn_amarillopress"
                android:textColor="@android:color/background_dark" />
            <TextView
                android:text="$0.00"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/text_DeudaTotal"
                android:layout_weight="1"
                android:textAlignment="center"
                android:textSize="18sp"
                android:background="@android:color/background_light"
                android:textColor="@android:color/background_dark" />
        </LinearLayout>
    </LinearLayout>
  </ScrollView>
</LinearLayout>
```

- Archivo “Patente_Consulta.java”

```
package ec.gob.chordeleg.chordemovil;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
```

```
import android.view.Gravity;
import android.view.View;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.Toast;
import com.android.volley.DefaultRetryPolicy;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.ServerError;
import com.android.volley.TimeoutError;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;
import java.io.UnsupportedEncodingException;

public class Patente_Consulta extends AppCompatActivity implements Response.Listener<String>, Response.ErrorListener,
View.OnClickListener{
    //Variables globales
    private ImageButton imgbtnAtras, imgbtnMinimiza;
    private Button btnConsultaPatente;
    private EditText editCodPatente;
    private RequestQueue requestQueue;
    private String url;
    private StringRequest stringRequest;
    private ProgressDialog dialogoEspera;
    private Toast toast;
    private Boolean conectado;
    //Función de inicio del activity
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_patente_consulta);
        inicailizar();
    }
    //Función para inicializar la configuración
    private void inicailizar(){
        //Asociar cada btn,text,etc al view y habilitar el listener
        btnConsultaPatente = (Button) findViewById(R.id.btn_ConsultaPatente);
        btnConsultaPatente.setOnClickListener(this);
        imgbtnAtras = (ImageButton) findViewById(R.id.imgbtn_Atras);
        imgbtnAtras.setOnClickListener(this);
        imgbtnMinimiza = (ImageButton) findViewById(R.id.imgbtn_Minimiza);
        imgbtnMinimiza.setOnClickListener(this);
        editCodPatente = (EditText) findViewById(R.id.edit_CodPatente);
        //Configurar las variables para consumir la APIrest del WS
        requestQueue = Volley.newRequestQueue(this);
        url = "http://ws.chordeleg.gob.ec:3001/patente/";
        //Configurar las notificaciones
        toast = Toast.makeText(this, "", Toast.LENGTH_LONG);
        toast.setGravity(Gravity.CENTER_VERTICAL, 0, 0);
        //Configurar el dialogo de espera
        dialogoEspera = new ProgressDialog(this);
        dialogoEspera.setCancelable(false);
        dialogoEspera.setMessage("Procesando...");
        //Configurar el teclado para que inicie en la pantalla
        getWindow().setSoftInputMode(WindowManager.LayoutParams.SOFT_INPUT_STATE_ALWAYS_VISIBLE);
    }
    //Función para consultar los datos
    private void consultar(String codPatente){
        if (validaInternet(this)) {
            dialogoEspera.show();
            stringRequest = new StringRequest(Request.Method.GET, url + codPatente, this, this);
            stringRequest.setRetryPolicy(new DefaultRetryPolicy(500, 3, 1));
            requestQueue.add(stringRequest);
        } else {
            toast.setText("Revice su conexión a internet...");
            toast.show();
        }
    }
    //Función para validar la conexión a internet
    public boolean validaInternet(Context context) {
        conectado = false;
```

```
        ConnectivityManager connec = (ConnectivityManager) context.getSystemService(Context.CONNECTIVITY_SERVICE);
        //Obtiene todas las redes (móviles y wifi)
        NetworkInfo[] redes = connec.getAllNetworkInfo();
        for (int i = 0; i < redes.length; i++) {
            if (redes[i].getState() == NetworkInfo.State.CONNECTED) {
                conectado = true;
                i = redes.length;
            }
        }
        return conectado;
    }
    //Función para establecer la acción de cada btn presionado
    @Override
    public void onClick(View view) {
        switch (view.getId()) {
            case R.id.btn_ConsultaPatente:
                if (!editCodPatente.getText().toString().equals("")) {
                    consultar(editCodPatente.getText().toString());
                } else {
                    toast.setText("Ingrese su número de Patente...");
                    toast.show();
                }
                break;
            case R.id.imgbtn_Atras:
                onBackPressed();
                break;
            case R.id.imgbtn_Minimiza:
                moveTaskToBack(true);
                break;
        }
    }
    //Función para establecer la acción si se obtiene respuesta del WS
    @Override
    public void onResponse(String response) {
        try {
            response = new String(response.getBytes("ISO-8859-1"), "UTF-8");
            Intent intentReportePat = new Intent(this, Patente_Reporte.class);
            intentReportePat.putExtra("codPatente", editCodPatente.getText().toString());
            intentReportePat.putExtra("datosStr", response);
            dialogoEspera.dismiss();
            startActivity(intentReportePat);
        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        }
    }
    //Función para establecer la acción si se obtiene un error del WS
    @Override
    public void onErrorResponse(VolleyError error) {
        dialogoEspera.dismiss();
        if (error instanceof ServerError) {
            toast.setText("No existe la Patente ingresada...");
            toast.show();
        } else if (error instanceof TimeoutError) {
            toast.setText("Tiempo de espera agotado, vuelva a intentarlo...");
            toast.show();
        } else {
            toast.setText("Lo sentimos, servicio no disponible. Inténtelo más tarde...");
            toast.show();
        }
    }
}
```

- Archivo “activity_patente_consulta.xml”

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_patente_consulta"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:weightSum="1"
    tools:context="ec.gob.chordeleg.chordemovil.Patente_Consulta"
```

```
android:background="@android:color/background_dark">
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:background="@color/colorPrimary"
    android:layout_height="45dp"
    android:id="@+id/LI_actionBar">
    <ImageButton
        android:layout_height="match_parent"
        app:srcCompat="@drawable/atras32"
        android:id="@+id/imgbtn_Atras"
        android:background="@drawable/btn_cafe"
        android:layout_width="wrap_content"
        android:paddingLeft="12dp" />
    <TextView
        android:text="CHORDELEGmóvil"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_NombreApp"
        android:background="@color/colorPrimary"
        android:textColor="@android:color/holo_orange_light"
        android:textAlignment="center"
        android:fontFamily="sans-serif"
        android:textSize="22sp"
        android:gravity="center_vertical"
        android:layout_weight="1" />
    <ImageButton
        android:layout_height="match_parent"
        app:srcCompat="@drawable/minimizar32"
        android:id="@+id/imgbtn_Minimiza"
        android:background="@drawable/btn_cafe"
        android:layout_width="wrap_content"
        android:paddingRight="12dp" />
</LinearLayout>
<ImageView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    app:srcCompat="@drawable/pate114"
    android:id="@+id/img_LogoPatente"
    android:background="@android:color/background_dark" />
<TextView
    android:text="NUMERO DE PATENTE"
    android:layout_width="match_parent"
    android:id="@+id/eti_CodPatente"
    android:gravity="center_vertical|center_horizontal"
    android:background="@drawable/btn_amarillopress"
    android:fontFamily="sans-serif"
    android:textSize="22sp"
    android:textColor="@android:color/background_dark"
    android:layout_height="38dp" />
<EditText
    android:layout_width="match_parent"
    android:id="@+id/edit_CodPatente"
    android:hint="Ingrese su número"
    android:textSize="22sp"
    android:textAlignment="center"
    android:fontFamily="sans-serif"
    android:textAllCaps="false"
    android:inputType="number"
    android:background="@drawable/edit_blanco"
    android:layout_gravity="center_horizontal"
    android:ems="10"
    android:maxLength="5"
    android:layout_height="42dp" />
<Button
    android:layout_width="186dp"
    android:layout_height="45dp"
    android:id="@+id/btn_ConsultaPatente"
    android:fontFamily="sans-serif"
    android:background="@drawable/btn_amarillo"
    android:textColor="@android:color/background_dark"
    android:textSize="22sp"
    android:textAllCaps="false"
    android:layout_gravity="center_horizontal"
    android:text="CONSULTAR"
```

```
        android:drawableLeft="@drawable/lupa32"
        android:paddingLeft="8dp"
        android:textStyle="normal|bold"
        android:layout_marginTop="5dp" />
</LinearLayout>
```

- Archivo “Patente_Reporte.java”

```
package ec.gob.chordeleg.chordemovil;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Gravity;
import android.view.View;
import android.widget.ImageButton;
import android.widget.TextView;
import android.widget.Toast;
import org.json.JSONArray;
import org.json.JSONException;
public class Patente_Reporte extends AppCompatActivity implements View.OnClickListener {
    //Variables globales
    private ImageButton imgbtnAtras, imgbtnMinimiza;
    private TextView textCedula, textPropietario, textCodPatente,
            textFechaAct, textPeriodoAct, textDeudaAct, textInteresAct,
            textPlanillasAnt, textPeriodoAnt, textDeudaAnt, textInteresAnt,
            textDeudaTotal;
    private String datosStr;
    private JSONArray datosJSON;
    private Toast toast;
    //Función de inicio del activity
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_patente_reporte);
        inicializar();
        recibirDatos();
    }
    //Función para inicializar la configuración
    private void inicializar() {
        //Asociar cada btn,text,etc al view y habilitar el listener
        imgbtnAtras = (ImageButton) findViewById(R.id.imgbtn_Atras);
        imgbtnAtras.setOnClickListener(this);
        imgbtnMinimiza = (ImageButton) findViewById(R.id.imgbtn_Minimiza);
        imgbtnMinimiza.setOnClickListener(this);
        textCedula = (TextView) findViewById(R.id.text_Cedula);
        textPropietario = (TextView) findViewById(R.id.text_Propietario);
        textCodPatente = (TextView) findViewById(R.id.text_CodPatente);
        textFechaAct = (TextView) findViewById(R.id.text_FechaAct);
        textPeriodoAct = (TextView) findViewById(R.id.text_PeriodoAct);
        textDeudaAct = (TextView) findViewById(R.id.text_DeudaAct);
        textInteresAct = (TextView) findViewById(R.id.text_InteresAct);
        textPlanillasAnt = (TextView) findViewById(R.id.text_PlanillasAnt);
        textPeriodoAnt = (TextView) findViewById(R.id.text_PeriodoAnt);
        textDeudaAnt = (TextView) findViewById(R.id.text_DeudaAnt);
        textInteresAnt = (TextView) findViewById(R.id.text_InteresAnt);
        textDeudaTotal = (TextView) findViewById(R.id.text_DeudaTotal);
        //Configurar la notificación de deuda pendiente
        toast = Toast.makeText(this, "No tiene deudas pendientes...", Toast.LENGTH_LONG);
        toast.setGravity(Gravity.CENTER_VERTICAL, 0, 0);
    }
    //Función para recibir y presentar los datos del activity anterior
    private void recibirDatos() {
        try {
            datosStr = getIntent().getStringExtra("datosStr");
            datosJSON = new JSONArray(datosStr);
            textCedula.setText(datosJSON.getJSONObject(0).getString("c1"));
            textPropietario.setText(datosJSON.getJSONObject(0).getString("p1"));
            textCodPatente.setText(getIntent().getStringExtra("codPatente"));
            textFechaAct.setText(datosJSON.getJSONObject(0).getString("f2"));
            textPeriodoAct.setText(datosJSON.getJSONObject(0).getString("p2"));
            textDeudaAct.setText(datosJSON.getJSONObject(0).getString("d2"));
            textInteresAct.setText(datosJSON.getJSONObject(0).getString("i2"));
            textPlanillasAnt.setText(datosJSON.getJSONObject(0).getString("t3"));
            textPeriodoAnt.setText(datosJSON.getJSONObject(0).getString("p3"));
            textDeudaAnt.setText(datosJSON.getJSONObject(0).getString("d3"));
```

```
            textInteresAnt.setText(datosJSON.getJSONObject(0).getString("i3"));
            textDeudaTotal.setText(datosJSON.getJSONObject(0).getString("d4"));
            if (textDeudaTotal.getText().toString().equals("$0.00")) {
                toast.show();
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }

    //Función para establecer la acción de cada btn presionado
    @Override
    public void onClick(View view) {
        switch (view.getId()) {
            case R.id.imgbtn_Atras:
                onBackPressed();
                break;
            case R.id.imgbtn_Minimiza:
                moveTaskToBack(true);
                break;
        }
    }
}
```

- Archivo “activity_patente_reporte.xml”

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_patente_consulta"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:weightSum="1"
    tools:context="ec.gob.chordeleg.chordemovil.Patente_Reporte"
    android:background="@android:color/background_dark">
    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="45dp"
        android:id="@+id/Line_ActionBar">
        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            app:srcCompat="@drawable/atras32"
            android:id="@+id/imgbtn_Atras"
            android:background="@drawable/btn_cafe"
            android:paddingLeft="12dp" />
        <TextView
            android:text="CHORDELEGmóvil"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:id="@+id/text_NombreApp"
            android:background="@color/colorPrimary"
            android:textColor="@android:color/holo_orange_light"
            android:textAlignment="center"
            android:fontFamily="sans-serif"
            android:textSize="22sp"
            android:gravity="center_vertical"
            android:layout_weight="1" />
        <ImageButton
            android:layout_height="match_parent"
            app:srcCompat="@drawable/minimizar32"
            android:id="@+id/imgbtn_Minimiza"
            android:background="@drawable/btn_cafe"
            android:layout_width="wrap_content"
            android:paddingRight="12dp" />
    </LinearLayout>
    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent">
        <LinearLayout
            android:layout_width="match_parent"
```

```
android:layout_height="match_parent"
android:orientation="vertical" >
<ImageView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    app:srcCompat="@drawable/pate114"
    android:id="@+id/img_LogoPatente" />
<TextView
    android:text="DATOS GENERALES"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/eti_DatosGenerales"
    android:textAlignment="center"
    android:textSize="18sp"
    android:textColor="@android:color/background_dark"
    android:background="@drawable/btn_amarillopress"
    android:gravity="center_vertical" />
<TextView
    android:text="Propietario"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/eti_Propietario"
    android:layout_weight="1"
    android:textAlignment="center"
    android:textSize="18sp"
    android:gravity="center_vertical"
    android:textColor="@android:color/background_dark"
    android:background="@android:color/darker_gray" />
<TextView
    android:text="TextView"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/text_Propietario"
    android:textAlignment="center"
    android:textSize="18sp"
    android:layout_weight="1"
    android:gravity="center_vertical"
    android:background="@android:color/background_light" />
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="Cédula"
        android:layout_height="match_parent"
        android:id="@+id/eti_Cedula"
        android:layout_weight="1"
        android:textAlignment="center"
        android:background="@android:color/darker_gray"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:layout_width="match_parent"
        android:gravity="center_vertical" />
    <TextView
        android:text="Num. Patente"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_CodPatente"
        android:layout_weight="1"
        android:textAlignment="center"
        android:background="@android:color/darker_gray"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:gravity="center_vertical" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="TextView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_Cedula"
        android:layout_weight="1"
```

```
        android:textAlignment="center"
        android:background="@android:color/background_light"
        android:textSize="18sp"
        android:gravity="center_vertical" />
    <TextView
        android:text="TextView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_CodPatente"
        android:layout_weight="1"
        android:textAlignment="center"
        android:background="@android:color/background_light"
        android:textSize="18sp"
        android:gravity="center_vertical" />
</LinearLayout>
<TextView
    android:text="DEUDA ACTUAL"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/eti_DeudaActual"
    android:textAlignment="center"
    android:fontFamily="sans-serif"
    android:textColor="@android:color/background_dark"
    android:textSize="18sp"
    android:background="@drawable/btn_amarillopress"
    android:gravity="center_vertical" />
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="Fecha Emisión"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_FechaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
    <TextView
        android:text="Periodo"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_PeriodoAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_FechaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light"
        android:text="00/00/0000" />
    <TextView
        android:text="0000"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_PeriodoAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
</LinearLayout>
<LinearLayout
```

```
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="Valor"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_DeudaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
    <TextView
        android:text="Interés"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_InteresAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_DeudaAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
    <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_InteresAct"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
</LinearLayout>
<TextView
    android:text="DEUDA ANTERIOR"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/eti_DeudaAnterior"
    android:textAlignment="center"
    android:fontFamily="sans-serif"
    android:textColor="@android:color/background_dark"
    android:textSize="18sp"
    android:background="@drawable/btn_amarillopress"
    android:gravity="center_vertical" />
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:text="Planillas"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_PlanillasAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
    <TextView
        android:text="Periodo"
        android:layout_width="match_parent"
```

```
        android:layout_height="match_parent"
        android:id="@+id/eti_periodoAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
    </LinearLayout>
    <LinearLayout
      android:orientation="horizontal"
      android:layout_width="match_parent"
      android:layout_height="match_parent">
      <TextView
        android:text="0"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_PlanillasAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
      <TextView
        android:text="0000 - 0000"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_PeriodoAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
    </LinearLayout>
    <LinearLayout
      android:orientation="horizontal"
      android:layout_width="match_parent"
      android:layout_height="match_parent">
      <TextView
        android:text="Valor"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_DeudaAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
      <TextView
        android:text="Interés"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/eti_InteresAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textColor="@android:color/background_dark"
        android:background="@android:color/darker_gray" />
    </LinearLayout>
    <LinearLayout
      android:orientation="horizontal"
      android:layout_width="match_parent"
      android:layout_height="match_parent">
      <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_DeudaAnt"
        android:layout_weight="1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:background="@android:color/background_light" />
      <TextView
        android:text="$0.00"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/text_InteresAnt"
        android:layout_weight="1"
```

```
            android:textAlignment="center"
            android:textSize="18sp"
            android:background="@android:color/background_light" />
        </LinearLayout>
        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:paddingTop="10dp">
            <TextView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:id="@+id/eti_Espacio"
                android:layout_weight="1"
                android:textAlignment="center"
                android:textSize="18sp"
                android:background="@android:color/background_dark"
                android:textColor="@android:color/background_light" />
            <TextView
                android:text="TOTAL:"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/eti_DeudaTotal"
                android:layout_weight="1"
                android:textAlignment="center"
                android:textSize="18sp"
                android:background="@drawable/btn_amarillopress"
                android:textColor="@android:color/background_dark" />
            <TextView
                android:text="$0.00"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/text_DeudaTotal"
                android:layout_weight="1"
                android:textAlignment="center"
                android:textSize="18sp"
                android:background="@android:color/background_light"
                android:textColor="@android:color/background_dark" />
        </LinearLayout>
    </LinearLayout>
  </ScrollView>
</LinearLayout>
```

Anexo 4: Código de configuración de QoS en el router mikrotik

- Marcación de conexiones y paquetes

```
/ip firewall mangle
add action=mark-connection chain=prerouting comment="P8 - P2P" new-connection-mark="PRIO 8 - conn" p2p=all-p2p
add action=mark-packet chain=prerouting connection-mark="PRIO 8 - conn" new-packet-mark="PRIO 8"
add action=jump chain=prerouting jump-target="TERMINO DE PROCESAR" packet-mark="PRIO 8"
add action=mark-connection chain=prerouting comment="P7 - DESCARGAS > 50M" connection-bytes=50000000-0 new-
connection-mark="PRIO 7 - conn" protocol=tcp
add action=mark-packet chain=prerouting connection-mark="PRIO 7 - conn" new-packet-mark="PRIO 7"
add action=jump chain=prerouting jump-target="TERMINO DE PROCESAR" packet-mark="PRIO 7"
add action=mark-connection chain=prerouting comment="P1 - ADMINISTRACION RED Y DNS" new-connection-mark="PRIO
1 - conn" protocol=icmp
add action=mark-connection chain=prerouting dst-port=21,8291 new-connection-mark="PRIO 1 - conn" protocol=tcp
add action=mark-connection chain=prerouting dst-port=53 new-connection-mark="PRIO 1 - conn" protocol=udp
add action=mark-connection chain=output dst-port=53 new-connection-mark="PRIO 1 - conn" protocol=udp
add action=mark-packet chain=prerouting connection-mark="PRIO 1 - conn" new-packet-mark="PRIO 1"
add action=jump chain=prerouting jump-target="TERMINO DE PROCESAR" packet-mark="PRIO 1"
add action=mark-connection chain=prerouting comment="P2 - BASES DE DATOS" dst-port=3306,5432,1453 new-connection-
mark="PRIO 2 - conn" protocol=tcp
add action=mark-packet chain=prerouting connection-mark="PRIO 2 - conn" new-packet-mark="PRIO 2"
add action=jump chain=prerouting jump-target="TERMINO DE PROCESAR" packet-mark="PRIO 2"
add action=mark-connection chain=prerouting comment="P3 - WEB SERVICE" dst-port=3001 new-connection-mark="PRIO 3 -
conn" protocol=tcp
add action=mark-packet chain=prerouting connection-mark="PRIO 3 - conn" new-packet-mark="PRIO 3"
add action=jump chain=prerouting jump-target="TERMINO DE PROCESAR" packet-mark="PRIO 3"
add action=mark-connection chain=prerouting comment="P4 - NAVEGACION WEB" dst-port=80,443,8000-9000 new-
connection-mark="PRIO 4 - conn" protocol=tcp
add action=mark-packet chain=prerouting connection-mark="PRIO 4 - conn" new-packet-mark="PRIO 4"
```

```
add action=jump chain=prerouting jump-target="TERMINO DE PROCESAR" packet-mark="PRIO 4"
add action=mark-connection chain=prerouting comment="P5 - CORREO" dst-port=110,143,995,993,25,587 new-connection-
mark="PRIO 5 - conn" protocol=tcp
add action=mark-packet chain=prerouting connection-mark="PRIO 5 - conn" new-packet-mark="PRIO 5"
add action=jump chain=prerouting jump-target="TERMINO DE PROCESAR" packet-mark="PRIO 5"
add action=mark-connection chain=prerouting comment="P6 - OTROS" new-connection-mark="PRIO 6 - conn"
add action=mark-packet chain=prerouting connection-mark="PRIO 6 - conn" new-packet-mark="PRIO 6"
add chain="TERMINO DE PROCESAR"
```

- Priorización de tráfico

```
/queue tree
add name=Bajada parent=Eth05_LANmuni priority=1 queue=pcq-download-default
add name="P1 - ADMIN, DNS_down" packet-mark="PRIO 1" parent=Bajada priority=1 queue=pcq-download-default
add name="P2 - BASES DE DATOS_down" packet-mark="PRIO 2" parent=Bajada priority=2 queue=pcq-download-default
add limit-at=2540k max-limit=5080k name="P3 - WEB SERVICE_down" packet-mark="PRIO 3" parent=Bajada priority=3
queue=pcq-download-default
add name="P4 - NAVEGACION_down" packet-mark="PRIO 4" parent=Bajada priority=4 queue=pcq-download-default
add name="P5 - CORREO_down" packet-mark="PRIO 5" parent=Bajada priority=5 queue=pcq-download-default
add name="P7 - DESCARGAS>50_down" packet-mark="PRIO 7" parent=Bajada priority=7 queue=pcq-download-default
add name="P8 - P2P_down" packet-mark="PRIO 8" parent=Bajada queue=pcq-download-default
add name=Subida parent=Eth01_WanCNT priority=1 queue=pcq-upload-default
add name="P1 - ADMIN, DNS_up" packet-mark="PRIO 1" parent=Subida priority=1 queue=pcq-upload-default
add name="P2 - BASES DE DATOS_up" packet-mark="PRIO 2" parent=Subida priority=2 queue=pcq-upload-default
add limit-at=3630k max-limit=7270k name="P3 - WEB SERVICE_up" packet-mark="PRIO 3" parent=Subida priority=3
queue=pcq-upload-default
add name="P4 - NAVEGACION_up" packet-mark="PRIO 4" parent=Subida priority=4 queue=pcq-upload-default
add name="P5 - CORREO_up" packet-mark="PRIO 5" parent=Subida priority=5 queue=pcq-upload-default
add name="P6 - OTROS_up" packet-mark="PRIO 6" parent=Subida priority=6 queue=pcq-upload-default
add name="P8 - P2P_up" packet-mark="PRIO 8" parent=Subida queue=pcq-upload-default
add name="P6 - OTROS_down" packet-mark="PRIO 6" parent=Bajada priority=6 queue=pcq-download-default
add name="P7 - DESCARGAS>50_up" packet-mark="PRIO 7" parent=Subida priority=7 queue=pcq-upload-default
```