PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

FACULTAD DE INGENIERÍA

ESCUELA DE SISTEMAS

DESARROLLO E IMPLEMENTACIÓN DE UNA PLATAFORMA DIGITAL PARA LA GESTIÓN DE AUTO COMPARTIDO COMO APOYO AL PROGRAMA DE CAMPUS SOSTENIBLE PUCE VIVA

EDUARDO ALBUJA

DIEGO DE LA TORRE

2020

ÍNDICE

TABLAS

ILUSTRACIONES

1. CAPITULO I

1.1. INTRODUCCION

Puesto que como causa del aumento del tráfico vehicular y por ende el incremento de la contaminación en la ciudad de Quito, gracias al análisis realizado por la REMMAQ (Red Metropolitana de Monitoreo Atmosférico Quito) en el año 2017, puede resaltar que las máximas emisiones de Óxidos de Nitrógeno (gases que afectan a la contaminación ambiental) se producen debido al incremento del parque vehicular, aumento del uso de automóviles a base de Diesel y también a las condiciones climáticas en la ciudad. (Secretaría de Ambiente Alcaldía de Quito, 2018). Por lo tanto, se ha visto oportuno desarrollar la idea de poder involucrar a la comunidad universitaria de la Pontificia Universidad Católica del Ecuador sede Quito, para lograr una sostenibilidad institucional enfocada en el cuidado del medioambiente para aportar así al cambio que la ciudad necesita para combatir con estos serios problemas que afectan a cada uno de los ciudadanos.

Por tal motivo, nace la idea de apoyar al proyecto PUCE VIVA mediante nuestros conocimientos tecnológicos poder realizar la construcción de un modelo organizacional eficiente y sostenible, con el cual iniciar desde la universidad un cambio; el cual podrá ser la base para la continuación de una estrategia colaborativa para el cuidado del medio ambiente.

1.2. JUSTIFICACION

En el 2015 más de 150 países aprobaron la agenda 2030 para el desarrollo sostenible, que involucra la implementación a nivel global de 16 objetivos conocidos como ODS (Objetivos de Desarrollo Sostenible), si bien estos objetivos no son obligatorios, se espera que todos los gobiernos incentiven su aplicación, ya que es una responsabilidad mundial asegurar el bienestar de las generaciones actuales y futuras.

El deseo de la ONU es que las personas, comunidades e individuos se apropien de estos objetivos, tomen un papel más protagónico y dejen de esperar que las soluciones vengan de las autoridades nacionales e internacionales; en el contexto de nuestro país el plan nacional de buen vivir también contempla al desarrollo sostenible dentro de sus objetivos, y en el plano institucional tanto la encíclica papal Laudato SI como el marco de las universidades AUSJAL (Asociación de Universidades Confiadas a la Compañía de Jesús en América Latina) prioriza la implementación de acciones que nos ayuden a alcanzar nuestro desarrollo y bienestar; sin comprometer la disponibilidad de recursos para el futuro y teniendo respeto y cuidado hacia nuestro medio ambiente; dentro de este contexto las universidades marcan un rol de vital importancia en la sociedad, ya que al ser la cuna de los futuros profesionales y gobernantes de los países, es importante que se formen con una visión y actitud responsables ante los graves problemas ambientales, sociales y económicos afrontados en la actualidad, además de tener la facilidad desde el campo de la investigación el desarrollar e implementar nuevas herramientas y estrategias que puedan ser replicadas por otras organizaciones.

El proyecto PUCE VIVA surge como una iniciativa interdisciplinaria, que busca responder a las necesidades institucionales y nacionales en el campo de la sostenibilidad, mediante la búsqueda de un mejor desempeño en todas las actividades ejecutadas por la comunidad universitaria, además esta iniciativa busca consolidarse como ente receptor de las diferentes iniciativas en el campo de la sostenibilidad dentro de la universidad, con el apoyo de las diferentes facultades desde sus fortalezas. Con el presente proyecto se busca también generar protocolos que puedan ser replicables en otras sedes de la universidad y en otras instituciones del país.

1.3. PLANTEAMIENTO DEL PROBLEMA

El proyecto PUCE VIVA nace con la finalidad de transformar a la PUCE en un campus sostenible y es la continuación del proyecto denominado en su primera fase ECOPUCE, el cual busca establecer un programa de campus sostenible usando el formato de auto certificación. Este programa establece un conjunto de acciones que se enmarcan en cinco áreas:

1. Transporte,
2. Energía,
3. Agua,
4. Comunicación,
5. Manejo de desechos.

Proyecto PUCE VIVA está bajo la responsabilidad de la facultad de Ciencias Exactas y Naturales, con el liderazgo de la Msc. Isabel Cipriani.

Desde el séptimo resultado del proyecto PUCE VIVA, que es la aplicación de las TICs se requiere establecer una plataforma digital como soporte para la operación del campus sostenible para la movilidad de las personas en la ciudad.

El proyecto PUCE VIVA tiene siete resultados finales, el séptimo de ellos corresponde a las TICs, con el desarrollo de una plataforma digital de soporte para la operación del campus sostenible, en lo que corresponde a movilidad de las personas.

El desarrollo de este sistema que busca una relación de la PUCE sede Quito con el medio ambiente para establecer el auto compartido, tiene como finalidad la disminución del uso de vehículos entre la comunidad universitaria y por otra parte brindar un servicio de transporte entre los miembros de la PUCE sede Quito, mediante esto se busca un ahorro de recursos naturales y económicos.

1.4. OBJETIVOS

1.4.1. General

Desarrollar e implementar una plataforma digital para la gestión de auto compartido como apoyo al programa de campus sostenible PUCE VIVA.

1.4.2. Específicos

- Extraer y analizar las necesidades que el problema presenta para transformarlas en requerimientos y plantear una solución ingenieril.
- Modelar y diseñar un sistema mediante el resultado del análisis de requerimientos para la gestión del servicio de auto compartido planteado en el proyecto PUCE VIVA, el cual está direccionado a miembros de la Comunidad Universitaria como son: Administrativos, Docentes y Estudiantes.
- Construir un modelo mediante la arquitectura de tres capas (modelo, vista y controlador) mediante el uso de herramientas tecnológicas basadas en la calidad de desarrollo de software para el manejo tanto de usuarios, vehículos, rutas, horarios y marcadores que permitan la fácil interacción entre usuario - plataforma.
- Gestionar las rutas de transporte mediante el uso de la API (Interfaz de Programación de Aplicaciones) de Google Maps para optimizar el servicio de la plataforma de auto compartido.
- Evaluar el funcionamiento del sistema, aplicando herramientas tecnológicas para corroborar el impacto del sistema en la reducción vehicular y amigabilidad con el medio ambiente en cumplimiento con los requisitos del sistema.
- Documentar correctamente el proceso de desarrollo de software del aplicativo, de igual manera el Manual de Usuario.
- Concluir mediante la determinación de las ventajas y desventajas del proyecto aplicativo desarrollado para la gestión del auto compartido en la comunidad universitaria.

2. CAPITULO II

2.1. MARCO REFERENCIAL

En la actualidad las personas se ven involucradas de manera directa frente al uso del vehículo, debido a que este objeto se ha convertido en muchos de los casos en una herramienta del diario vivir, el cual en el pasado se lo podía observar como un lujo; en cambio hoy en día se ha vuelto tan importante que se podría hablar de una necesidad.

El medio de transporte es uno de los servicios que a diario utiliza la población dentro de la ciudad de Quito, la gran cantidad de vehículos que se observan en las calles capitalinas ha puesto en marcha desde la Agencia Metropolitana de Tránsito que se implemente el plan de Pico y Placa, para disminuir tanto el número como el daño ambiental en horario de horas pico.

Actualmente se ha mejorado y optimizado este proceso extendiendo en horarios de restricciones reemplazando al anterior proyecto de la Alcaldía con el denominado: “Hoy no circula”, Ordenanza reformatoria 001 de la Ordenanza Metropolitana No. 305 art. I. 473 (4) Implementación de hoy no circula. (Agencia Metropolitana Tránsito, 2019).

En el caso particular del transporte de personas, se ha podido evidenciar que por varios factores como menciona (Garces & Naranjo, 2013) que el alto uso del vehículo privado se debe al deficiente servicio de transporte público de la ciudad. En este contexto la Pontificia Universidad Católica del Ecuador, busca disminuir el parque automotor que circula dentro de la ciudad de Quito al crear un mecanismo de servicio de auto compartido para beneficiar a la comunidad educativa y administrativa de la universidad, con el fin de alinearse a ser un campus sostenible.

La solidaridad en cuanto al uso del vehículo compartido, hace que se pueda plantear la problemática que se está enfocando en este proyecto o investigación, para evitar el uso del vehículo individual en el cual se busca solventar la solución del problema antes mencionado acerca del alto uso del vehículo privado, con opciones que los miembros de la Pontificia Universidad Católica del Ecuador puedan solventar su movilidad, así obteniendo un beneficio mutuo, tanto para el medio ambiente como para la movilidad de los miembros de la comunidad universitaria.

Actualmente esta iniciativa ha sido adoptada por otras instituciones de nivel superior en la ciudad de Quito, por lo que se pretende contribuir con el servicio de la movilidad de las personas de nuestra comunidad.

2.1.1. Historia de Servicios y aplicaciones de movilización en Quito

En la actualidad con el avance de la tecnología existen más de tres aplicaciones de movilidad que rigen en nuestra ciudad tal como: UBER, INDRIVER, CABIFY, considerando las más destacadas, las cuales ofrecen el servicio de transporte y movilidad a los usuarios en forma particular, cada empresa maneja su modalidad de trabajo, sin embargo, han ayudado de manera significativa en la movilidad de las personas que residen dentro de nuestra ciudad.

Uber comenzó a operar en el Ecuador en el año 2017, la base que maneja es la conexión entre usuario y socio conductores que facilitan el transporte de manera segura y sencilla.

Desde 2017 aparecieron más aplicaciones orientadas al servicio de movilidad. La necesidad de una innovación en el sistema de transporte es un proceso de alto índice de portabilidad en Quito que sigue en constante innovación tecnológica.

Las Universidades han optado por esta modalidad, existiendo Auto Compartido USF, Auto Compartido UIDE, etc., logrando innovar la movilidad de transporte de sus estudiantes, liberando el uso del taxi clásico, buses y/o busetas.

La Universidad San Francisco de Quito cuenta con la plataforma Auto compartido, la cual se basa que estudiantes, profesores y personal tengan una alternativa de transporte para movilizarse hacia la universidad. La cual está limitada hacia el uso de los miembros de esta universidad, la cual está centrada en la problemática del uso de parqueaderos externos y liberar el tráfico en la ciudad. (Universidad San Francisco de Quito, 2016).

2.1.2. Transporte PUCE

La Universidad Católica en los últimos años juntamente con las campañas de organizaciones estudiantiles, han impulsado el crecimiento del servicio a la comunidad universitaria, ofreciendo para el transporte estudiantil el uso del bus escolar, que posee distintas rutas de transporte para los estudiantes, esperando

que en los próximos semestres cuente con una nueva forma de este tipo de servicio de transporte.

2.2. MARCO TEÓRICO

La ingeniería de Software y Computación tiene como objetivo el análisis de problemas de la vida real, para poder realizar un modelamiento y diseño del mismo para poder desarrollar una solución que se adapte al problema, el presente estudio está enfocado en la resolución del problema planteado en la Pontificia Universidad Católica del Ecuador, el cual está basado en la disminución del uso del vehículo privado en la comunidad universitaria mediante el uso compartido de vehículos entre los miembros de la misma, buscando el beneficio mutuo tanto para los usuarios de este proyecto como para el medio ambiente.

Un Sistema Web en el Desarrollo de Software se hace referencia a la herramienta para que los usuarios puedan acceder y utilizar mediante un servidor de Internet con la ayuda de un navegador (Miguel, 2015). El cual está construido en un lenguaje de programación, basado en una arquitectura específica y construido mediante una metodología de desarrollo. El principal beneficio de dichos sistemas es la portabilidad, debido a que hoy en día se puede obtener acceso prácticamente en cualquier dispositivo electrónico.

Mediante una arquitectura de software se puede lograr construir un sistema web, según el Instituto de Ingeniería Eléctrica y Electrónica (IEEE), afirma que: “La Arquitectura del Software es la organización fundamental de un sistema formada por sus componentes, las relaciones entre ellos y el contexto en el que se implantaran, y los principios que orientan su diseño y evolución” (The Institute of Electrical and Electronics Engineers, Inc., 2000). La cual proporciona una guía sobre la construcción de un producto software, mediante la abstracción de alto nivel para el desarrollo del diseño del software.

De igual manera una Metodología de Desarrollo de Software es la que colabora en el proceso de construcción de un sistema web, por ende, una Metodología de Desarrollo de Software se define como **“**El conjunto de filosofías, fases, procedimientos, reglas, técnicas herramientas, documentos y aspectos de formación para los desarrolladores de sistemas informáticos” (Rozo Nader, 2014). Que brinda un conjunto de actividades que se deben seguir para alcanzar el objetivo de desarrollar un producto de software

correctamente documentado y con las bases específicas para transformar los requerimientos del usuario en un sistema software.

Por otra parte, para una mejor gestión e interacción de una plataforma web se puede realizar el uso de API'S. Cuya referencia a las siglas en inglés son Application Programming Interface, que integran un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software en aplicaciones (Red Hat, 2019). Tienen la facilidad de la interacción entre la comunicación de varios servicios, unos con otros mediante la interacción de datos o funciones establecidas previamente. Estas se caracterizan por brindar herramientas de análisis e interacción con software específico, la cual provee de funciones predefinidas para interactuar entre el sistema y la API, brindando soluciones a problemas con mayor facilidad, en menor tiempo y costo posible.

2.3. MARCO CONCEPTUAL

2.3.1. Ingeniería

Aplicación Web

Es todo software o aplicativo que posee una interacción directa con el navegador, es decir que se ejecuta mediante un servidor de aplicaciones web el cual hace el uso del Internet. (Zofío Jiménez, 2013).

Servidor Web

Es el medio capaz de ejecutar software del lado del servidor en donde agrupa protocolos como el http y https, estándares que permiten la conexión e interacción entre sistemas informáticos, generando respuestas a distintas peticiones que se las realiza por parte del lado del usuario. (Zofío Jiménez, 2013).

Programación Orientada a Objetos

Tipo de programación enfocada en la implementación y generación de objetos utilizados dentro de la producción de software para solventar y resolver problemáticas y algoritmos. (Moreno Pérez, 2015).

Se considera como la evolución de la programación clásica ya que se encuentra enfocada en la abstracción de objetos del mundo real por medio de la aplicación de propiedades reflejadas dentro del lenguaje computacional.

Ciclo de vida de un desarrollo de Software

Es la guía metodológica que se debe seguir para el desarrollo de un producto de software, en el cual se especifican una serie de pasos que son necesarios para cumplir los requerimientos de calidad que un sistema debe tener, así como todo el proceso que se debe proseguir para hacer una implementación con alta superioridad y excelencia. El ciclo de vida de desarrollo de software comienza cuando el usuario tiene un problema que lo especifica como requerimientos que posterior se lo deben solventar. El ciclo de vida en cascada consta de las siguientes etapas:

- Análisis
- Diseño
- Implementación

- Pruebas
- Documentación
- Mantenimiento

Arquitectura 3 Capas

(Maldonado Guerrero, 2016) afirma que: "La arquitectura tres capas es un diseño que introduce una capa intermedia en el proceso".

Con lo mencionado anteriormente se define como un modelo de desarrollo en el cual se caracteriza por dividir la estructura del sistema en tres capas que son:

- Capa de Presentación**:** Interacción usuario-sistema y viceversa, interfaz del software.
- Capa de Negocio: Procesos y peticiones por parte de la interacción del usuario en el sistema, esta capa se caracteriza por ejecutar y enviar información.
- Capa de Datos: En la cual se almacena los datos del sistema, respaldo general de la información.

Patrón MVC

Metodología de diseño de software encargado de desglosar la lógica del negocio y cliente en tres niveles de abstracción o bloques estructurales:

- Modelo: Relacionada con la capa de datos en la cual posee relación directa con la base de datos.
- Vista: Relacionada con el Front-End, interacción con el usuario.
- Controlador: Es el punto medio entre la vista y el controlador, envía datos a la vista, captura las peticiones del usuario.

Modelo Cascada

Es una metodología de desarrollo de software enfocada al cumplimento de actividades en cada una de sus etapas las cuales son:

- Análisis
- Diseño
- Implementación
- Pruebas

- Documentación
- Manteamiento

Tomado de (Maldonado Guerrero, 2016).

PHP

Se lo define como Preprocesador de Hipertexto, lenguaje de programación capaz de realizar el procesado de texto plano UTF-8 por el lado del servidor con el cual es factible desarrollar aplicaciones web. (Vaswani, 2010).

Base de Datos

Estructura o repositorio para el almacenamiento de datos de diferentes tipos enfocado en la contención de información con accesibilidad a gestionar de manera completa e integra su contenido. (Beynon-Davies, 2014).

2.3.2. Aplicativo

Auto Compartido

El proyecto nace con la finalidad de mejorar y facilitar el servicio de transporte a la comunidad universitaria, así mismo busca fomentar el cuidado del medio ambiente reduciendo el número de vehículos que circulan en nuestra ciudad. El proyecto PUCE VIVA pone énfasis en tener una mejora continua para lograr una sostenibilidad alcanzable. Como parte de esta certificación, la Universidad requiere tener una innovación en el área de Transporte. Brindando al personal docente y estudiantes un medio de transporte seguro, confiable y permitiendo mantener un traslado confortable entre sus pasajeros. Con el uso de las TICs y la evolución tecnológica muchas instituciones de educación superior han ido implementado estas plataformas de auto compartido para sus estudiantes. Este proyecto busca conseguir esta meta y beneficio para los usuarios de la PUCE.

Por lo tanto, se define como un tipo de servicio de transporte en el cual se realiza el desplazamiento desde un punto específico hacia la universidad o viceversa, juntamente con otros usuarios de la comunidad universitaria que residan en un lugar cercano a la ruta ofrecida por el oferente quien comparte su vehículo. (Maldonado Guerrero, 2016).

Oferente

Es un miembro de la comunidad universitaria que brinda el servicio de transporte mediante el compartimiento de su vehículo. (Maldonado Guerrero, 2016).

Pasajero

Es un miembro de la comunidad universitaria que utiliza el servicio de auto compartido brindado por un oferente. (Maldonado Guerrero, 2016).

Ruta

"Camino o dirección que se toma para un propósito**".** (Real Academia Española, 2019)**.**

Este término hace referencia al camino que recorre el vehículo del oferente brindando el servicio de auto compartido. (Maldonado Guerrero, 2016)

3. CAPITULO III

3.1. ANALÍSIS DE REQUERIMIENTOS

3.1.1. SRS

El SRS (Especificación de requerimientos de software) es un conjunto de especificaciones de los requerimientos para el desarrollo de software.

ESPECIFICACIÓN DE REQUISITOS DE SOFTWARE

Proyecto: PUCE VIVA – AUTO COMPARTIDO

Revisión 1.0

Ficha del documento

Fecha	Revisión	Autor	Verificado
02/03/2020	1	Eduardo Albuja Diego De La Torre	

Documento validado por las partes en la fecha:

Jefe de Desarrollo del proyecto Auto Compartido	Por el director del proyecto Auto Compartido
Eduardo Albuja	Ing. Beatriz Campos

3.1.1.1. Introducción

En la fase de requerimientos se refiere a la especificación de requerimientos del software a desarrollarse en donde se toma las necesidades de los usuarios, y se aclaran las dudas sobre dichas necesidades, con preguntas, llegando a clarificar todo lo que el usuario desea que el software efectúe.

El propósito de aclarar las necesidades del usuario es traducir esas necesidades en requerimientos del software que se transformara en software como resultado final y a su vez acompañado de la documentación necesaria para el usuario.

3.1.1.1.1. Propósito

El propósito del documento es obtener los requerimientos de funcionalidad y la forma en la que el software debe funcionar según las necesidades del usuario.

Definir las necesidades de los usuarios para convertirlos en los requerimientos y funcionalidades del producto.

El documento está dirigido a los usuarios, ingenieros de desarrollo, y al equipo de desarrollo del proyecto dando la trazabilidad necesaria para el desarrollo del proyecto; en este caso en específico del Auto Compartido, encargado de proporcionar una mejora del uso de los recursos que tienen un impacto ambiental en la Pontificia Universidad Católica del Ecuador, brindando un servicio de vehículos compartidos entre la comunidad universitaria de la PUCE, y este sea aceptado por el usuario.

3.1.1.1.2. Alcance

Generar un sistema web de uso privado para las personas que conforman la comunidad universitaria de la Pontificia Universidad Católica del Ecuador, mediante el cual brindar un servicio de transporte compartido en el cual los usuarios pueden ser oferentes o pasajeros de los autos compartidos.

El destino en común siempre será el campus de la PUCE Quito y los distintos puntos de origen dependerá del oferente de acuerdo con su ruta especificada y los pasajeros serán aquellos que tengan registrado su domicilio cerca de la ruta dada por el oferente.

Las rutas serán registradas mediante el servicio de la API de geolocalización de rutas de Google Maps.

El tiempo de desarrollo del sistema web dependerá de la definición de requerimientos en el presente documento.

El desarrollo del sistema tendrá que mantenerse alineado a los estándares dispuestos por la Dirección de Informática y el Área de Desarrollo de la PUCE, razón por la cual es importante el involucramiento de un Funcionario de la Dirección de Informática.

3.1.1.1.3. Personal Involucrado

Nombre	Dr. Henry Roa
Rol	Directora del proyecto de Auto Compartido
Categoría profesional	Docente de la Pontificia Universidad Católica del Ecuador
Responsabilidades	Facilitador con las autoridades del proyecto PUCE VIVA y coordinador con el equipo de desarrollo.
Información de contacto	bcampos@puce.edu.ec

Nombre	Eduardo Albuja
Rol	Líder del equipo de desarrollo y programador del proyecto Auto Compartido
Categoría profesional	Estudiante la carrera de Ingeniería de Sistemas y Computación
Responsabilidades	Facilitador entre el equipo de desarrollo y el director del proyecto, desarrollador en el proyecto.

Información de contacto	ealbuja276@puce.edu.ec
Nombre	Diego De La Torre
Rol	Programador del proyecto PUCE VIVA
Categoría profesional	Estudiante la carrera de Ingeniería de Sistemas y Computación
Responsabilidades	Desarrollador en el proyecto
Información de contacto	ddelatorre908@puce.edu.ec

Nombre	Pablo Almeida
Rol	Vinculador entre el área de sistemas de la facultad de Ingeniería con la dirección de Informática
Categoría profesional	Ingeniero
Responsabilidades	Facilitador entre la dirección de Informática de la PUCE y el equipo de desarrollo del proyecto Auto Compartido, soporte para el equipo de desarrollo.
Información de contacto	palmeida319@puce.edu.ec

3.1.1.2. Descripción general

3.1.1.2.1. Perspectiva del producto

El presente sistema web se desarrollará con el objetivo de brindar un servicio de transporte compartido, en el cual las personas de la comunidad universitaria de la PUCE se podrán identificar Oferentes o Pasajeros.

3.1.1.2.2. Funcionalidad del producto

Las funcionalidades principales del producto se detallan a continuación:

- F0: Ingreso al Sistema
- F1: Administración de Usuarios
- F2: Administración de Vehículos
- F3: Administración de Rutas
- F4: Administración de Puntos

3.1.1.2.3. Características de los usuarios

Tipo de usuario	Oferente
Formación	Estudiante, docente o administrativo de la PUCE
Habilidades	Tener un vehículo con parqueadero en la PUCE
Actividades	Ofrecer su vehículo para brindar el servicio de Auto Compartido, registrar rutas, aceptar Puntos para la Ruta y observar la información de su Vehículo. Uso de tecnología para el manejo de la plataforma web.

Tipo de usuario	Pasajero
Formación	Estudiante, docente o administrativo de la PUCE
Habilidades	Uso de tecnología para el manejo de la plataforma web
Actividades	Inscribirse y observar rutas.

Tipo de usuario	Administrador
Formación	Administrativo de la PUCE
Habilidades	Uso de tecnología para el manejo de la plataforma web
Actividades	Administrar parámetros generales y realización de reportes

3.1.1.2.4. Restricciones

El proyecto tendrá las siguientes restricciones a considerar:

- Framework de Back – End: CodeIgniter.
- Framework de Front – End: Bootstrap, jQuery, JQueryUI.
- Base de da datos: MySQL.
- Versionamiento: GitHub.
- Servidor Web: Apache.

3.1.1.2.5. Suposiciones y dependencias

Los requerimientos no fueron tomados con el usuario, estos fueron asumidos a partir de la propuesta de la realización de un prototipo vinculado al proyecto PUCE VIVA, enfocado a incentivar un mejor uso de los recursos que tienen un gran impacto en el medio ambiente mediante el director del proyecto.

3.1.1.3. Requisitos específicos

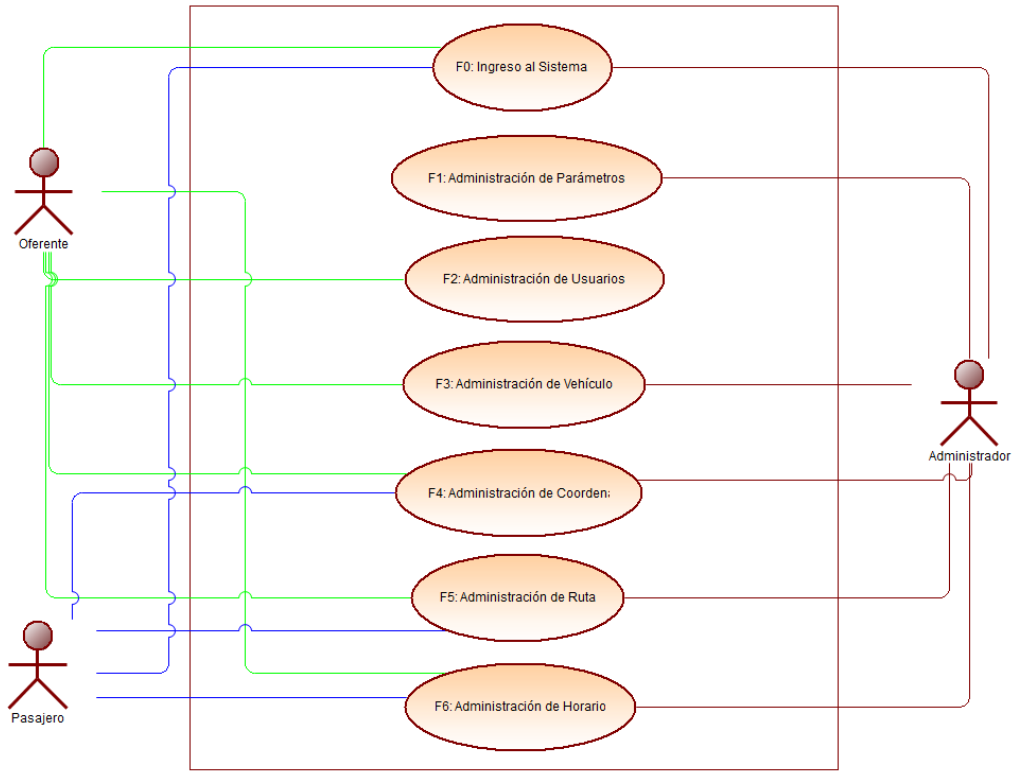

Ilustración 1.Requisitos específicos

Tabla 1. Ingreso al Sistema

Número de requisito	F0
Nombre de requisito	Ingreso al Sistema
Descripción	Permite el ingreso a la plataforma a los diversos usuarios, y a su vez distinguir si es Administrador, Oferente o Pasajero.
Prioridad del requisito	☒ Alta/Esencial ☐ Media/Deseado ☐ Baja/ Opcional
Actores	Administrador, Oferente y Pasajero

Tabla 2. Administración de Parámetros

Número de requisito	F1
Nombre de requisito	Administración de Parámetros
Descripción	Permite administrar los parámetros generales de la plataforma, en lo que se refiere a tiempo para la inscripción de rutas, rutas máximas por oferentes.
Prioridad del requisito	☒ Alta/Esencial ☐ Media/Deseado ☐ Baja/ Opcional
Actores	Administrador

Tabla 3. Administración de Usuario

Número de requisito	F2
Nombre de requisito	Administración de Usuario
Descripción	Permite administrar los usuarios, en lo que se refiere a la consulta de información.
Prioridad del requisito	☒ Alta/Esencial ☐ Media/Deseado ☐ Baja/ Opcional
Actores	Administrador, Oferente y Pasajero

Tabla 4. Administración de Vehículo

Número de requisito	F3
Nombre de requisito	Administración de Vehículo
Descripción	Permite administrar los vehículos, en lo que se refiere a la consulta de información.
Prioridad del requisito	☒ Alta/Esencial ☐ Media/Deseado ☐ Baja/ Opcional
Actores	Administrador y Oferente

Tabla 5. Administración de Coordenada

Número de requisito	F4
Nombre de requisito	Administración de Coordenada
Descripción	Permite administrar las coordenadas, en lo que se refiere a la creación y visualización.
Prioridad del requisito	☒ Alta/Esencial ☐ Media/Deseado ☐ Baja/ Opcional
Actores	Administrador, Oferente y Pasajero

Tabla 6. Administración de Ruta

Número de requisito	F5
Nombre de requisito	Administración de Ruta
Descripción	Permite administrar las rutas, en lo que se refiere a la creación, modificación y visualización.
Prioridad del requisito	☒ Alta/Esencial ☐ Media/Deseado ☐ Baja/ Opcional
Actores	Administrador, Oferente y Pasajero

Tabla 7. Administración de Horario

Número de requisito	F5
Nombre de requisito	Administración de Horario
Descripción	Permite administrar los horarios, en lo que se refiere a la creación, modificación y visualización.
Prioridad del requisito	☒ Alta/Esencial ☐ Media/Deseado ☐ Baja/ Opcional
Actores	Administrador, Oferente y Pasajero

3.1.1.3.1. Requisitos Funcionales

F1. Administración de Parámetros

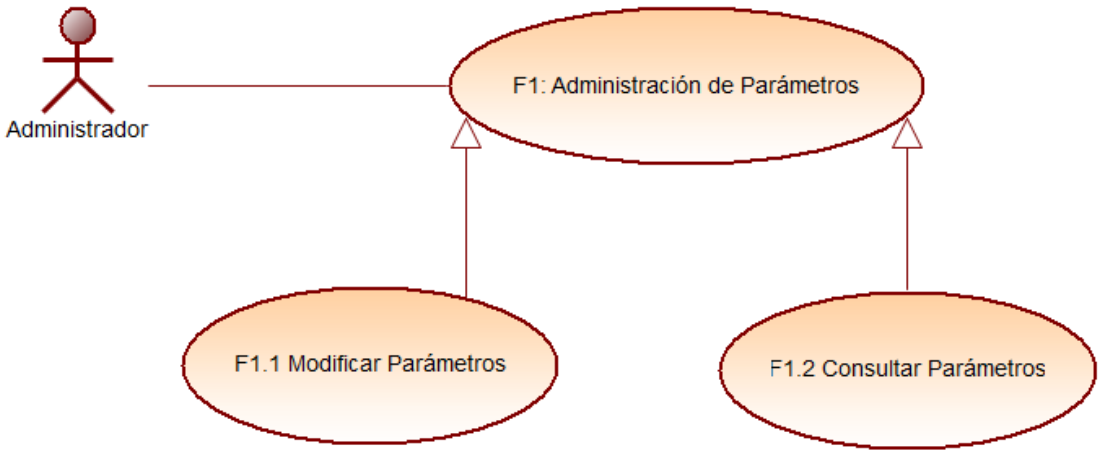

Ilustración 2. Administración de Parámetros

F2. Administración de Usuario

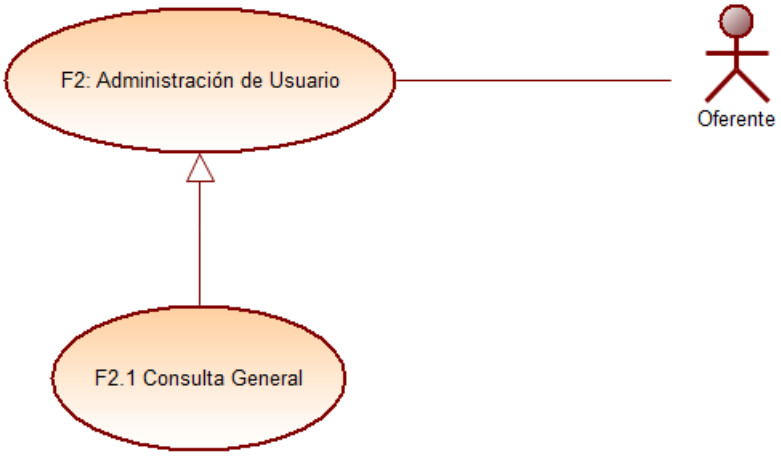

Ilustración 3. Administración de Usuario

F3. Administración de Vehículo

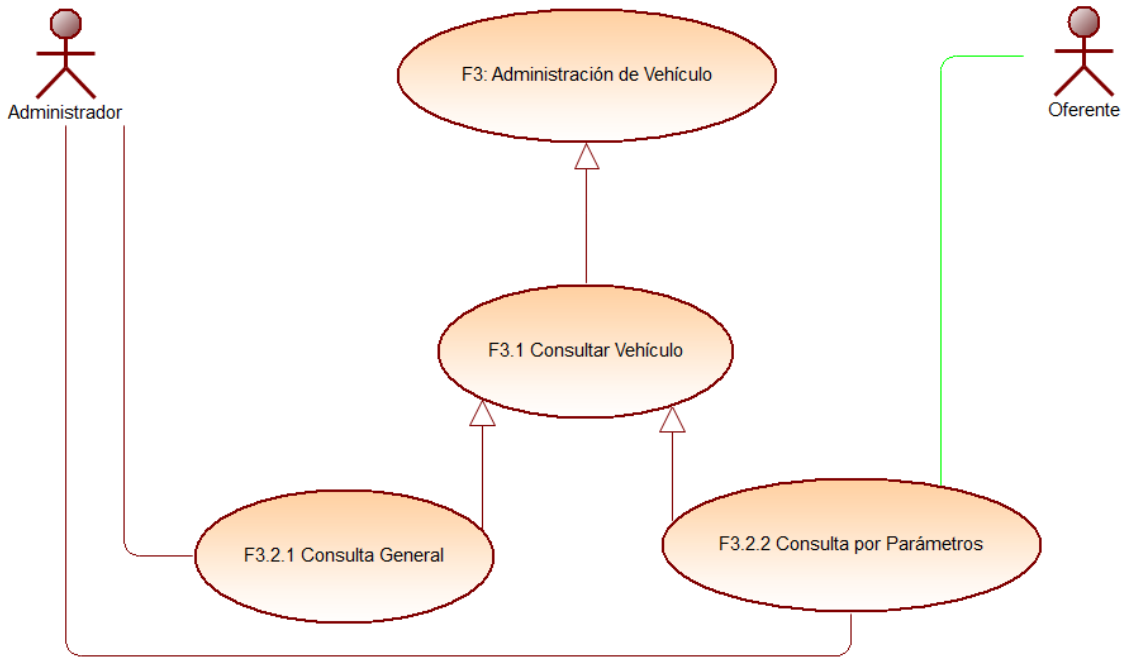

Ilustración 4. Administración de Vehículo

F4. Administración de Coordenada

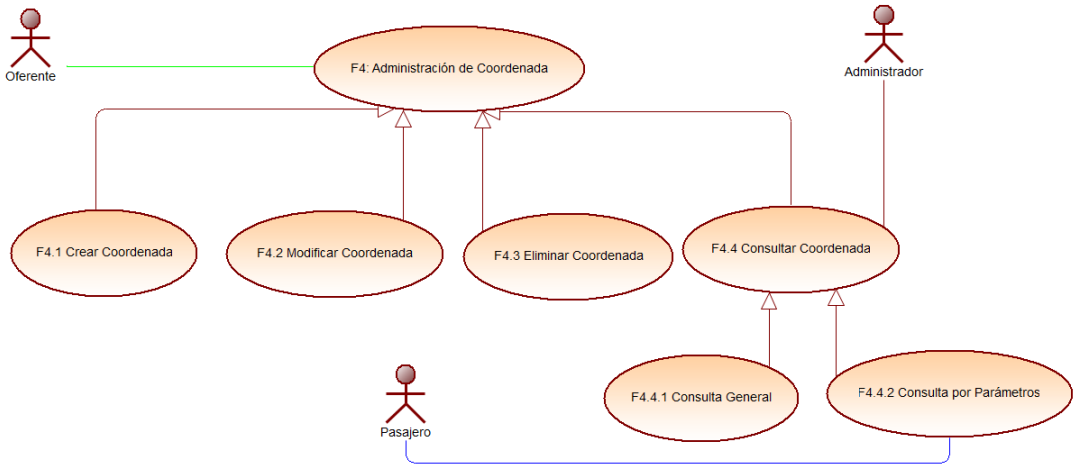

Ilustración 5. Administración de Coordenada

F5. Administración de Ruta

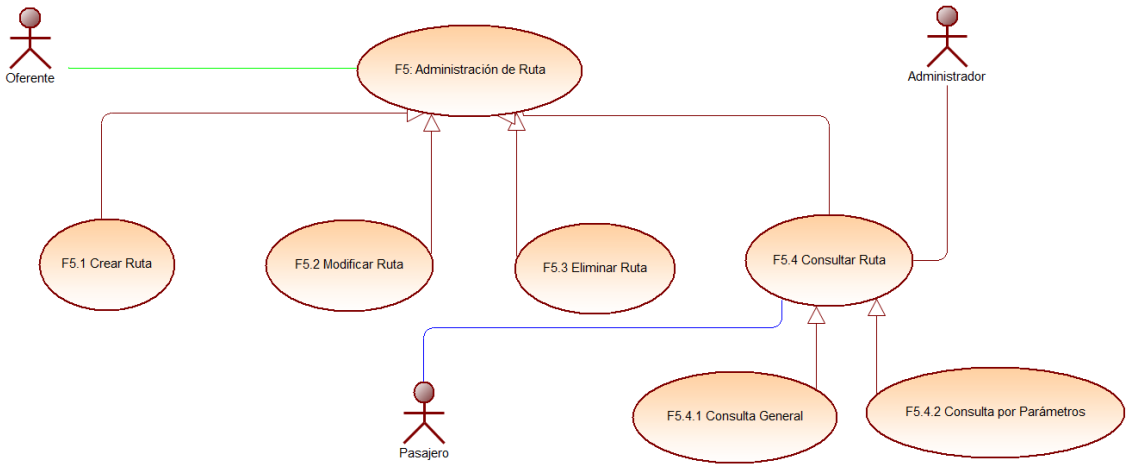

Ilustración 6. Administración de Ruta

F6. Administración de Horario

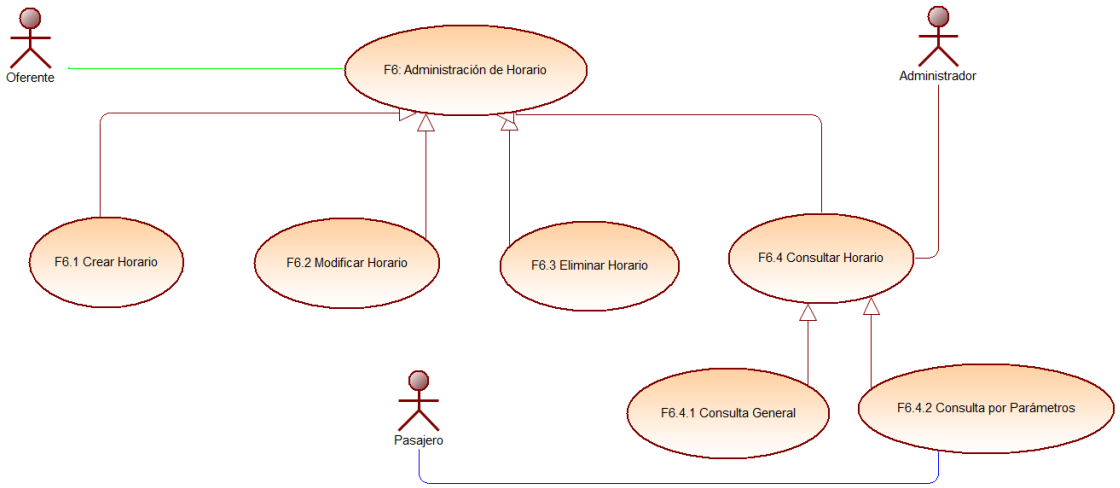

Ilustración 7. Administración de Horario

3.1.1.3.2. Requisitos No Funcionales

De Rendimiento

Los requisitos de rendimiento permiten una visión clara de los requerimientos del usuario, ya que especifican cuanto soporta un sistema antes de que se vuelva inutilizable. Por esta razón es necesario que se presenten de forma mesurable y no con palabras vagas.

Cada transacción debe realizarse con eficacia y eficiencia.

El despliegue de cualquier ventana, texto, gráfico o contenedor de información debe ser inmediato.

Seguridad

En la seguridad de un sistema se debe tomar las normas de uso del sistema, chequeos programados, copias de datos e información crítica, entre otras medidas.

El sistema implementará un inicio de sesión que brindará seguridad al acceso del usuario. No todos los actores pueden tener acceso a todas las partes de la plataforma.

Fiabilidad

La fiabilidad hace referencia a la tolerancia que tendrá frente a los fallos que puedan existir, como la conexión a internet, o a la base de datos y con esto manejar posibles pérdidas de información.

Disponibilidad

Similar a la fiabilidad, pero con un enfoque mayor hacia el tiempo. Se establece un tiempo estándar dentro del cual se garantiza que el sistema permanece en funcionamiento, incluso si se presentan fallos en él.

El sistema estará disponible siempre y cuando el servidor web este activo, ya que depende de este el funcionamiento de la plataforma.

Mantenibilidad

La mantenibilidad se refiere que tan preparado está el sistema para mejoras de funcionamiento, corrección de fallos o adaptación a un nuevo entorno. Además, se define los periodos de mantenimiento y los responsables de este.

Portabilidad

La portabilidad es la facilidad que tiene un sistema para ser trasladado y usado en otra plataforma o hardware. Considera el lenguaje en el que fue programado, el sistema operativo, la computadora, entre otras variables.

3.1.1.4. Diagrama de Clases

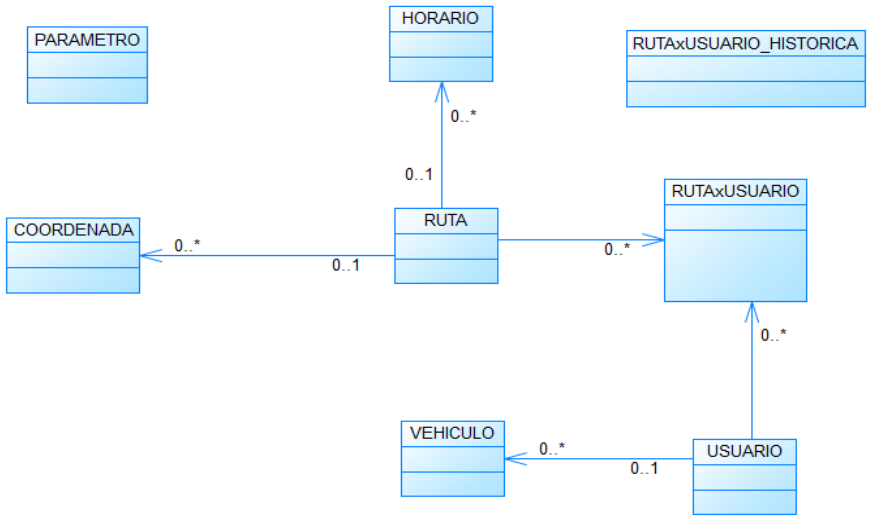

Ilustración 8. Diagrama de Clases

3.1.1.5. Diagrama Enditad Relación

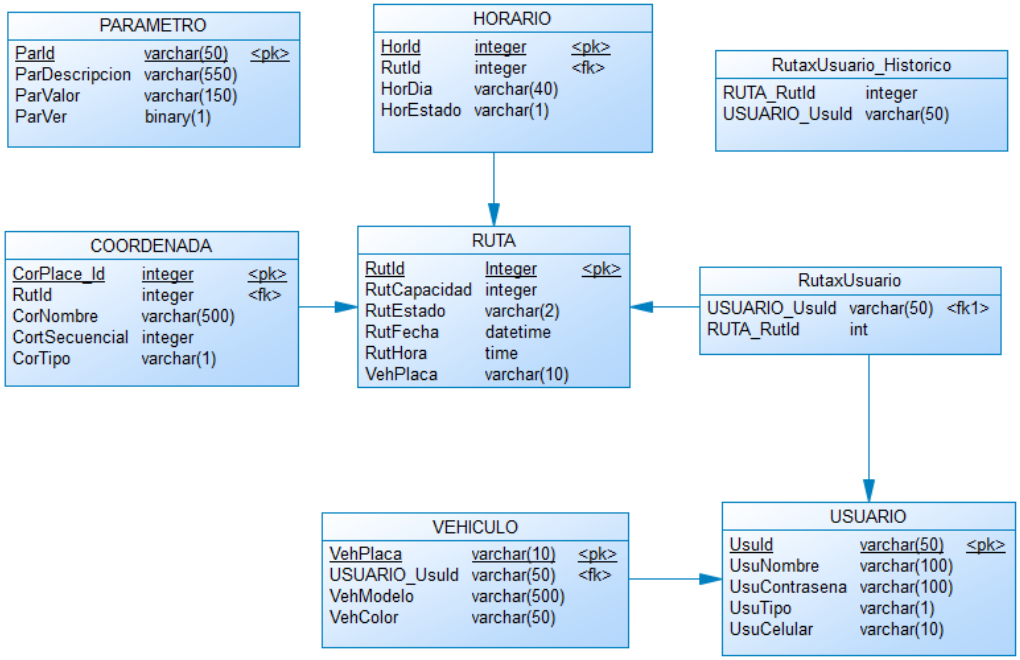

Ilustración 9. Diagrama Entidad Relación

3.2. HERRAMIENTAS

3.2.1. CODEIGNITER

Introducción

CodeIgniter es el framework en el cual se desarrolla la plataforma digital para gestión de auto compartido, se lo define como un marco de desarrollo para aplicaciones web que abarca como base el lenguaje de programación PHP. La característica principal es la rapidez con la que permite desarrollar proyectos y aplicativos siguiendo el modelamiento MVC con estructuras lógicas que permiten acceder a múltiples bibliotecas, lo que disminuye la cantidad de código a comparación de otros framework.

Estructura de la Aplicación

Un aplicativo desarrollado con este framework cuanta con 6 directorios por defecto los cuales son los siguientes:

- **/app:** Es el directorio en el cual se establece el código del aplicativo, además incluye las carpetas donde establece los controladores, modelos, vistas,

conexión a base de datos que son muy importantes para construir el producto de software.

- **/system:** Directorio en el cual se encuentra los archivos de configuración del marco del aplicativo, archivos que generalmente no se deben modificar.
- **/public:** Directorio en el que se establece la raíz web del aplicativo, contiene la accesibilidad del proyecto con la web, además es donde se puede agregar recursos multimedia de cualquier tipo.
- **/writable:** Directorio en el cual se pueden agregar archivos de cache, registros, archivos en general enviados por parte del usuario que se van añadiendo y creando conforme crezca y se construya el aplicativo.
- **/tests:** Directorio en el cual almacena archivos de prueba, a su vez contiene directorio support en el cual están clases, las cuales hacen como medio de prueba facilitando su modificación antes de ser ejecutadas y transferidas al servidor de producción del aplicativo.
- **/docs:** Directorio en el cual se encuentra la documentación oficial de CodeIgniter y guías de usuario para su desarrollo.

Modelos, vistas y controladores

Una aplicación web requiere de una estructura general en la cual se organice el código para lograr un buen funcionamiento, se necesita localizar y entender el código, es por esto que nace la necesidad de tener modelos estructurados con patrones de diseño que separan el proyecto en 3 partes, así como lo es MVC, el cual contiene modelos, vistas, controladores, pues así la arquitectura es mucho más eficaz y eficiente que modelos anteriores usados en la programación web.

Modelos

Gestionan los datos del aplicativo, contienen la lógica del negocio y reglas comerciales que cumplen con los requerimientos del usuario. Una de las funcionalidades del modelo es la interacción directa con la base de datos ya que establece e impone las reglas sobre los datos que se extraen de la misma, así como el guardado y recuperación de información.

Dentro de la estructura del proyecto los modelos se encuentran en la siguiente dirección: / app / Models.

Vistas

Gestiona la interacción con el usuario ya que es la interfaz del aplicativo, generalmente los archivos que pertenecen a este directorio son tipo HTML con muy poco código PHP, ayudan a la visualización de datos en tablas o estructuras para el despliegue de información la cual es obtenida por parte de los controladores, es decir existe un paso de información desde controlador a la vista.

Dentro de la estructura del proyecto las vistas se encuentran en la siguiente dirección: / app / Views

Controladores

Gestionan las peticiones por parte del usuario determinando acciones a ejecutar, es decir el paso de información al modelo permitiéndole guardar y solicitar data que el modelo envía a la vista. El controlador es el medio de interacción del aplicativo. Otra de las funcionalidades que ejerce es la administración de las solicitudes HTTP, el código relacionado con la autenticación, seguridad y redireccionamiento del proyecto.

Dentro de la estructura del proyecto los controladores se encuentran en la siguiente dirección: / app / Controllers.

3.2.2. GOOGLE MAPS JAVASCRIPT API

Esta herramienta permite la personalización de mapas visuales a través de distintos mecanismos de control y configuraciones. Se puede consumir este servicio en cualquier tipo de aplicativo web.

La carga del mapa general de Google Maps se lo realiza en un script, el cual genera una petición a la URL de la API.

```
<script
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyANQ4fOigvvkY9uxMGHCB_bh24-tlyFwgA&libraries=places&callback=initMap"
```

```
</script>
```

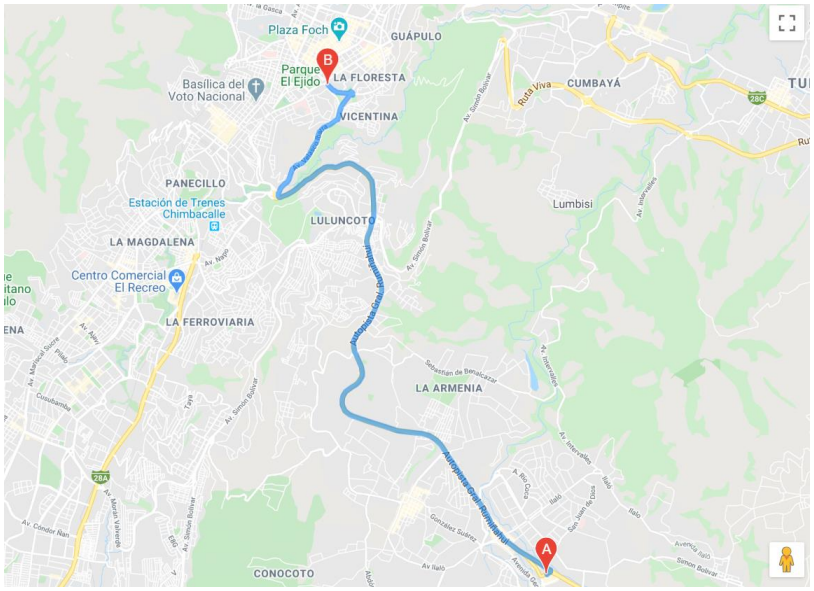

Ilustración 10. Google Maps JavaScript API

La personalización de un mapa requiere el tener una cuenta en la plataforma general de Google para poder utilizar dinámicamente el servicio en cualquier aplicativo.

Visualización

La forma de visualizar el mapa de Google maps es por medio del elemento “div” añadiendo el nombre map como atributo en su id.

Configuraciones Propósito

- center: Atributo en coordenadas para configurar la posición inicial del mapa, utiliza latitud y longitud.
- zoom: Atributo para aumentar o disminuir el acercamiento del mapa.

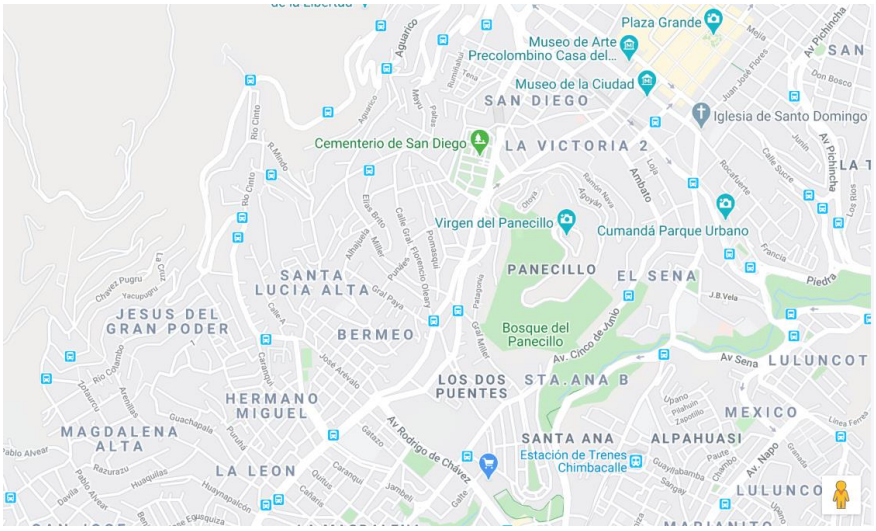

Ilustración 11. Ejemplo Google Maps JavaScript API

Personalización

La funcionalidad establecida en el mapa sirve para visualizar una ruta, la cual puede tener hasta tres puntos referenciales entre el punto de inicio y el punto final.

El ajuste que se realizó en el servicio de Google maps para el sistema, es el uso del estilo de mapa "Waypoints in Directions". Se implementó las siguientes funciones de servicio adicionales para que se pueda cumplir correctamente con el requerimiento de gestionar rutas por parte de los usuarios de la plataforma.

Directions Service

Es un servicio mediante el cual permite el cálculo y despliegue entre dos o más lugares. (Developers). Este servicio se implementa en la construcción de la ruta, permite capturar y obtener ubicaciones elegidas por parte de los usuarios del aplicativo para poder utilizarlas en la traza completa de la ruta final añadiendo puntos de referencias.

Directions Renderer

Es un servicio el cual complementa el Directions Service para representar gráficamente las ubicaciones elegidas por parte de los usuarios. (Developers)

AutocompleteDirectionsHandler

Este método se implementa en los campos de entrada del formulario del mapa para generar el auto completamiento de lugares en la previsualización de una ruta. (Developers)

3.2.3. MSQL

Es un servicio open source para la administración de base de datos relacionales, se caracteriza ya que es totalmente administrada y respaldada. Una de sus principales ventajas es tener la facilidad de integrar con entornos y lenguajes de programación como php, java, perl, etc., para el desarrollo de diferentes tipos de aplicaciones web. Entre los principales servicios que ofrece esta base de datos están los siguientes:

Protección de Datos

Implementa el cifrado de datos, seguridad total con firewall para el almacenamiento de información. Constantemente mantiene actualizaciones para evitar cualquier tipo de vulnerabilidad. (Corporation, 2020).

Aprovisionamiento instantáneo

Cuenta con la facilidad de conectar bases de datos preconfiguradas para la producción de aplicaciones web. (Corporation, 2020).

Integración

MSQL está implementado con infraestructura Oracle, por tanto, puede integrarse con cualquier tecnología de este medio. (Corporation, 2020).

3.3. DISEÑO

La plataforma web, estará basada en la arquitectura Modelo Vista Controlador (MVC), en cuanto al Modelo se lo administrará en la parte del Framework destinado al uso de este la cual está contenida en la carpeta “Models”, encargada de la interacción con la base de datos. En cuanto a la parte del Controlador será administrada por la carpeta “Controller”; y finalmente la parte de interacción entre el usuario y la plataforma estará destinada en la carpeta “Views”.

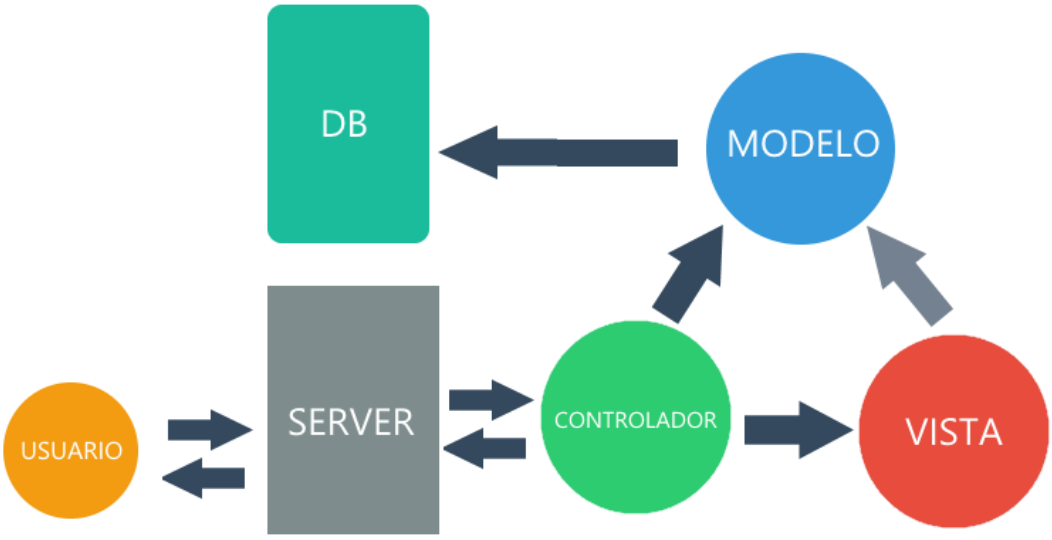

Ilustración 12. Diagrama Modelo Vista Controlador

Por otra parte, el diseño visual de la plataforma estará dispuesto a ser amigable con el usuario, conteniendo colores y formas distintivas para cada una de las funcionalidades de esta. Es así como, con cada tipo de usuario, se tendrá una barra lateral izquierda en la cual se contemplará la navegabilidad de cada una de las opciones a las cuales se podrá ingresar e interactuar.

Es así, que el sistema contara con una pantalla de inicio de sesión; de siguiente manera se encontrar con tres pantallas principales, las cuales harán referencia a los distintos tipos de usuarios: Administrador, Oferente y Pasajero.

3.4. IMPLEMENTACIÓN

Para iniciar con el desarrollo del sistema, primero se debe obtener las siguientes herramientas:

- CodeIgniter (Framework), obtenido mediante la página oficial. https://codeigniter.com/download
- XAMP (Servidor Local), obtenido mediante la página oficial. https://www.apachefriends.org/es/download.html
- Visual Studio Code (Editor), obtenido mediante la página oficial. https://code.visualstudio.com/download

Una vez instalado el servidor local, se debe colocar la carpeta del framework en la dirección del servidor local (..\xampp\htdocs). Posteriormente se cambia el nombre al directorio, en este caso auto-compartido.

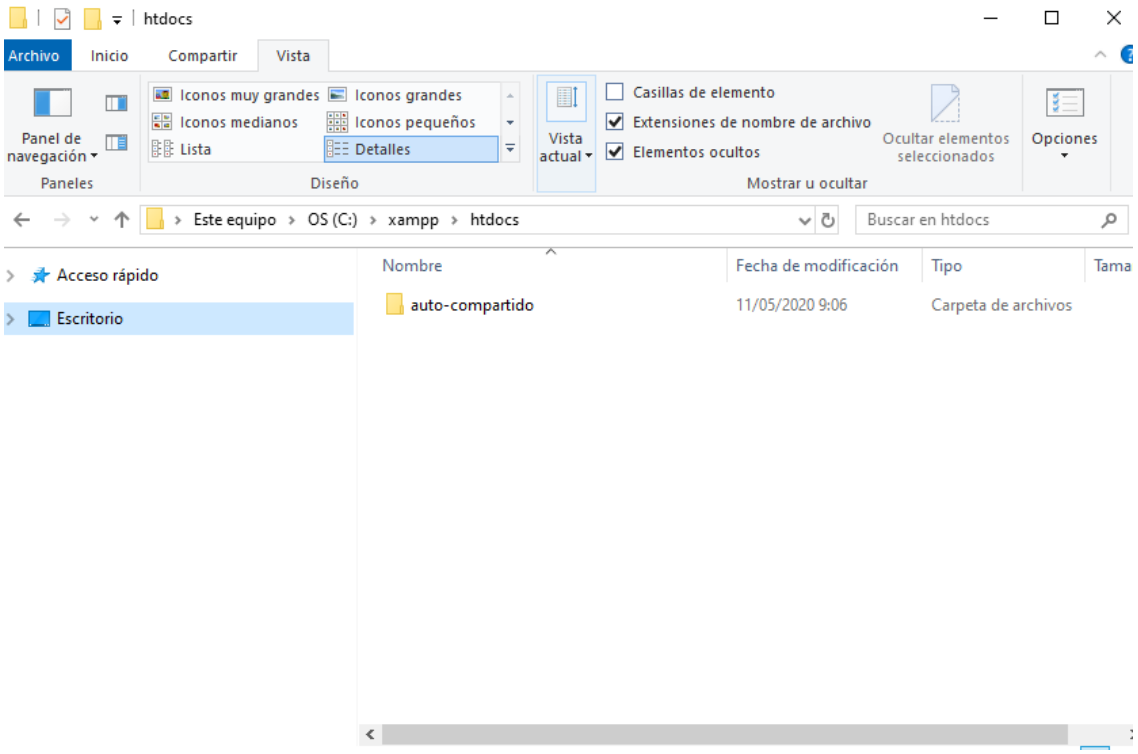

Ilustración 13. Carpeta del proyecto

Posteriormente, se debe realizar la configuración de la base de datos, creando una base de datos, y a su vez generar el script de generación mediante el modelo de base de datos presentado posteriormente.

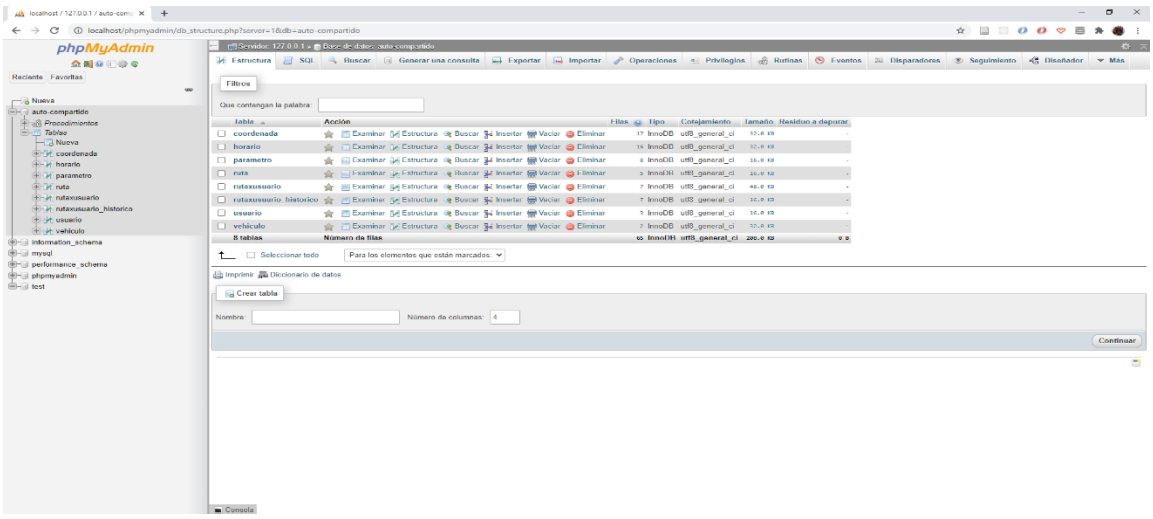

Ilustración 14. Base de Datos del proyecto

Para finalizar la configuración inicial, en el editor se debe abrir el proyecto y configurar el archivo de la base de datos (app/Config/Database.php), con las credenciales y nombre de la base de datos que se tiene en servidor phpMyAdmin del Servidor Local XAMPP.

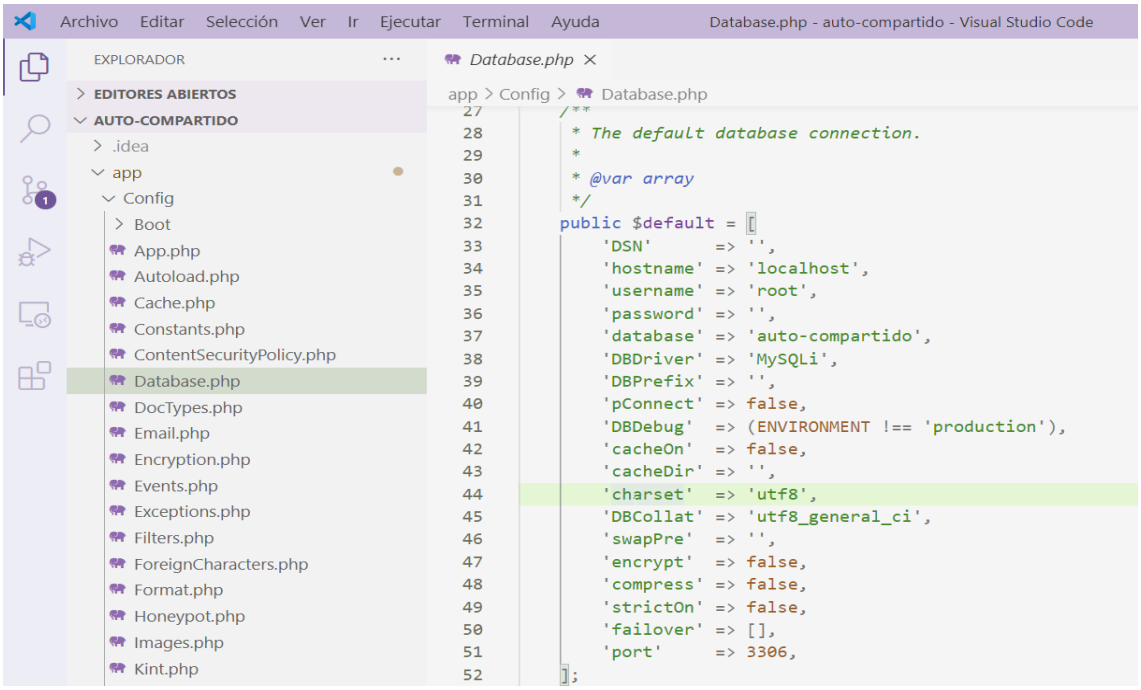

Ilustración 15. Configuración archivo Database.php

Posteriormente se procederá a la creación de modelos, controladores y vistas.

3.5. PRUEBAS

3.5.1. Ingreso al Sistema

Tabla 8. Pruebas - Ingreso al Sistema

Descripción	Estado
El sistema cuenta con un inicio de sesión.	Aprobado

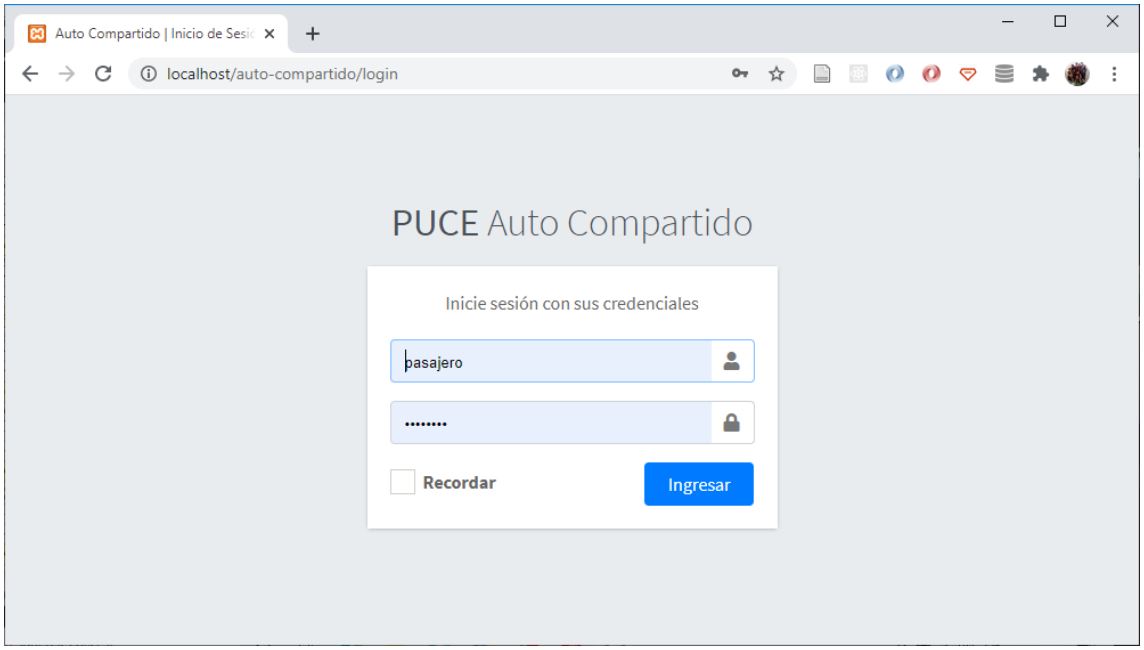

Ilustración 16. Inicio de Sesión

Descripción	Estado
El sistema autentifica los usuarios, en caso de credenciales incorrectas, muestra información al usuario.	Aprobado

Tabla 9. Pruebas - Validación Inicio de Sesión

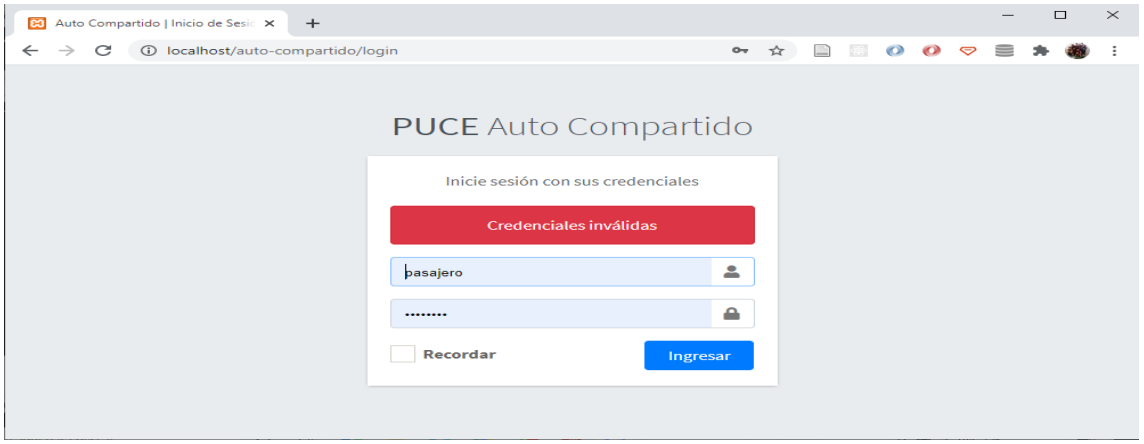

Ilustración 17. Validación de Inicio de Sesión

Tabla 10. Pruebas - Inicio de Sesión Administrador

Descripción	Estado
Inicio de sesión como administrador	Aprobado

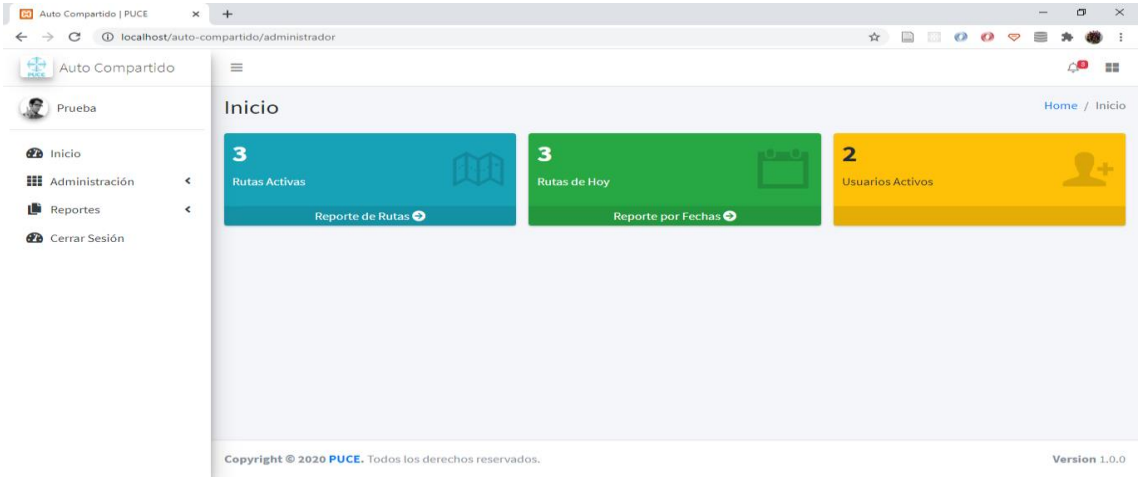

Ilustración 18. Inicio de Sesión Administrador.

Tabla 11. Pruebas - Inicio de Sesión Pasajeros

Descripción	Estado
Inicio de sesión como pasajero.	Aprobado

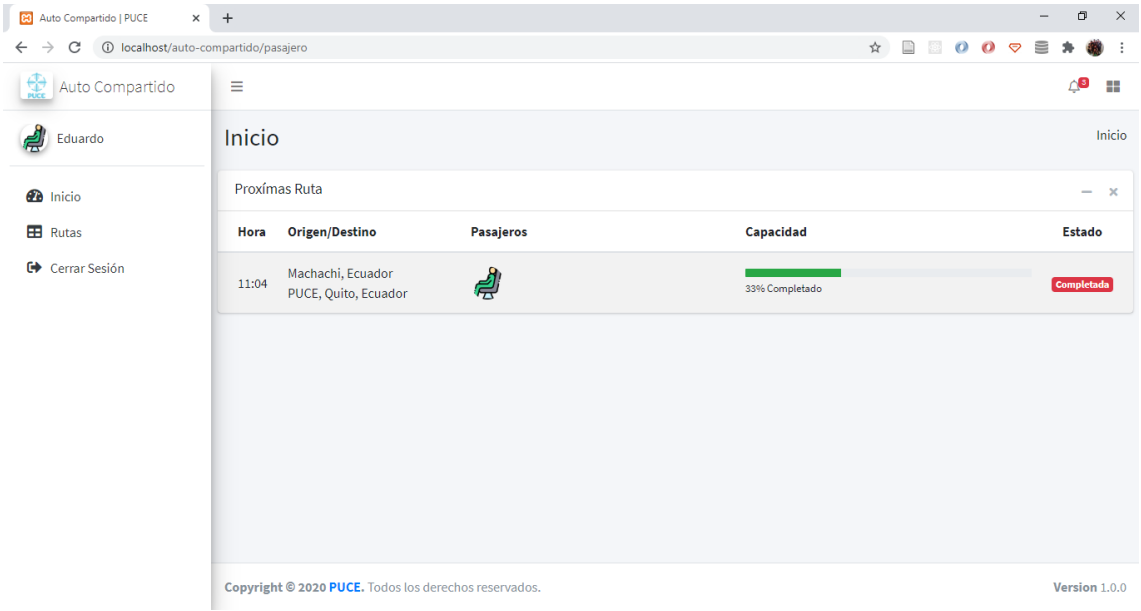

Ilustración 19. Inicio de Sesión Pasajeros

Tabla 12. Pruebas - Inicio de Sesión Oferente

Descripción	Estado
Inicio de sesión como oferente.	Aprobado

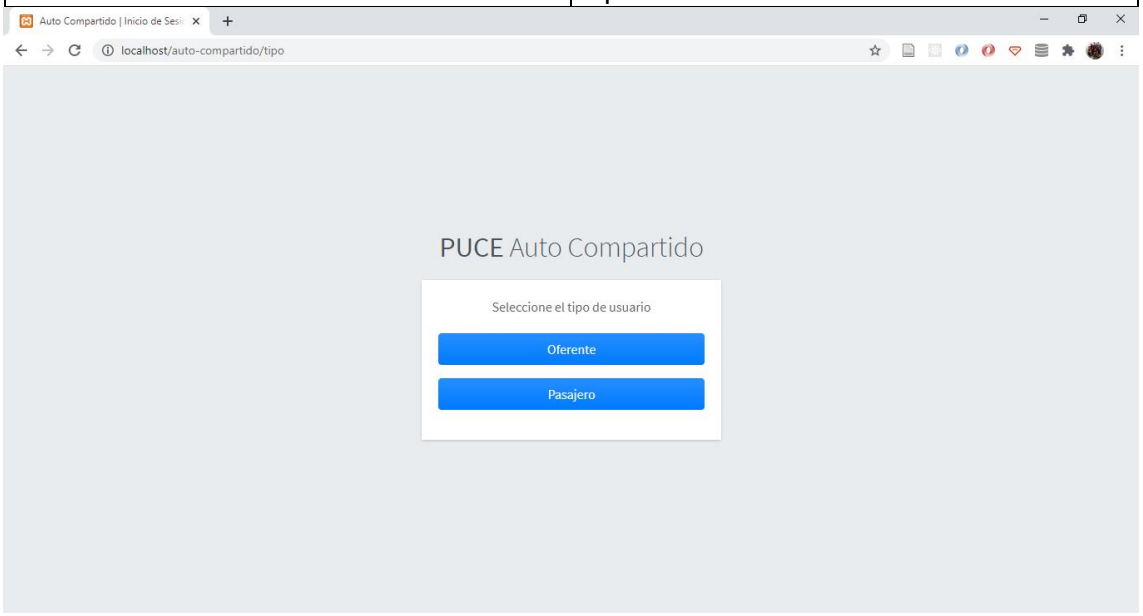

Ilustración 20. Inicio de Sesión Oferente

3.5.2. Administración de Parámetros

Tabla 13. Pruebas - Administración de Parámetros

Descripción	Estado
El sistema cuenta con administración de parámetros.	Aprobado

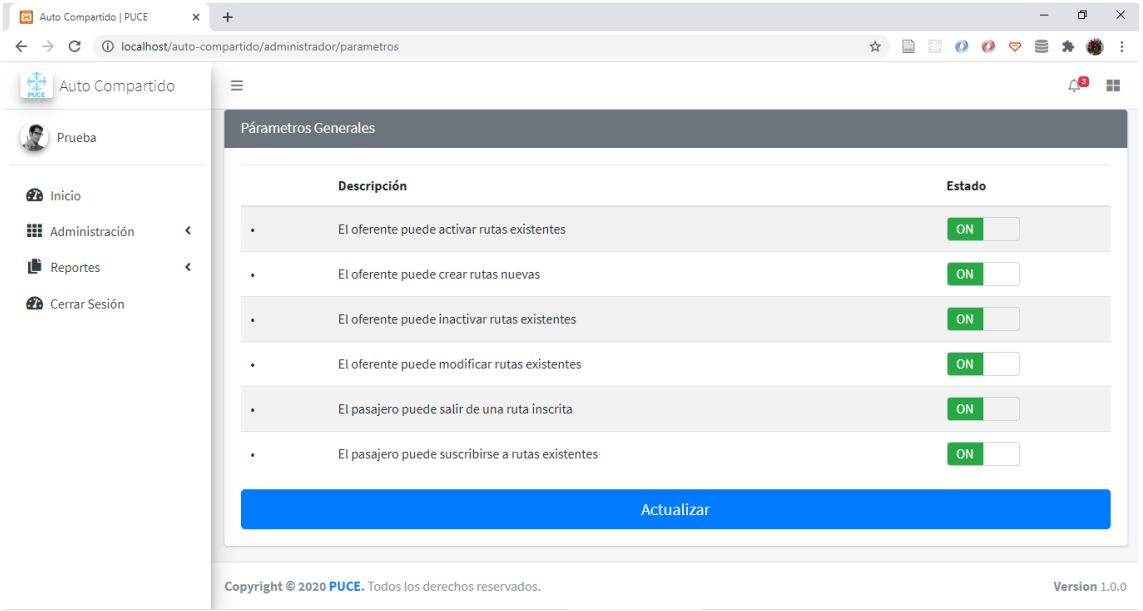

Ilustración 21. Administración de Parámetros

3.5.3. Administración de Usuarios

Tabla 14.Pruebas - Administración de Usuarios

Descripción	Estado
El sistema cuenta con administración de usuarios	Aprobado

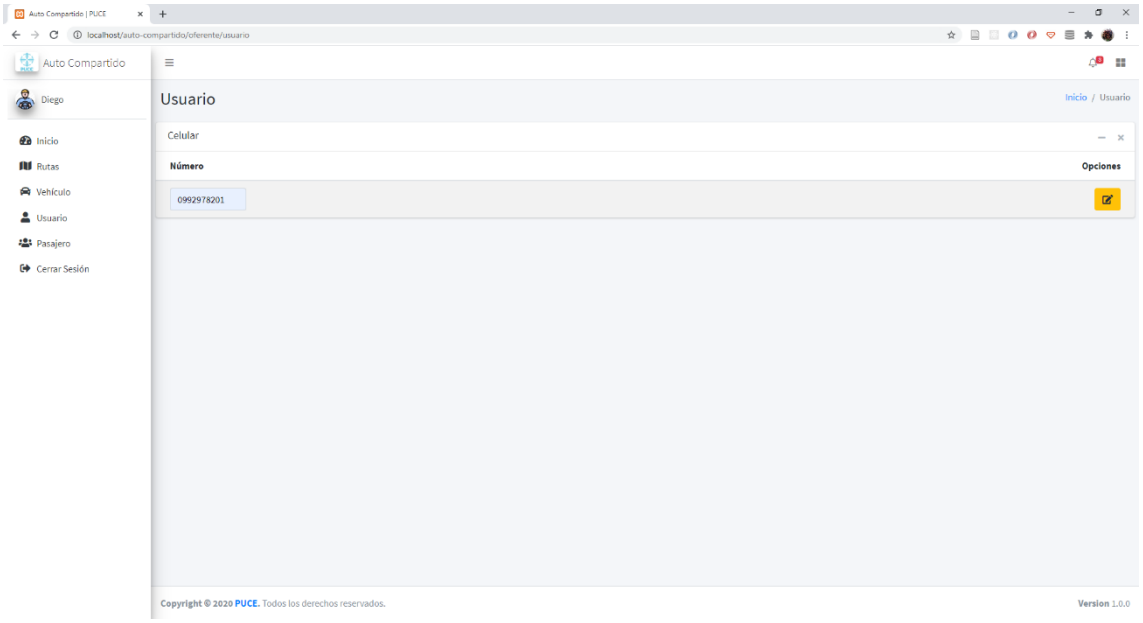

Ilustración 22. Administración de Usuarios

3.5.4. Administración de Vehículos

Tabla 15.Pruebas - Administración de Vehículos

Descripción	Estado
El sistema cuenta con administración de vehículos.	Aprobado

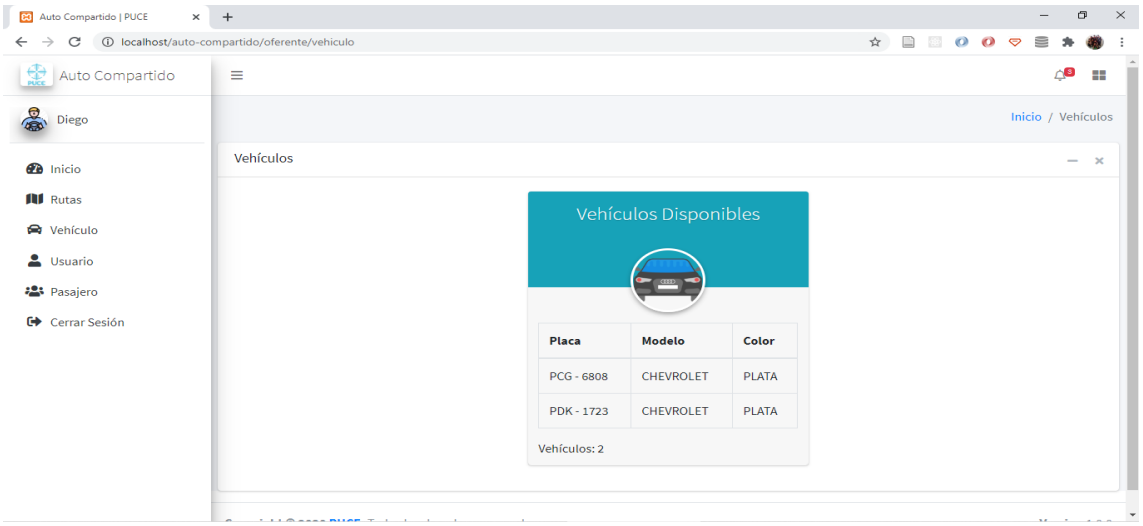

Ilustración 23. Administración de Vehículos

3.5.5. Administración de Ruta, Coordenadas y Horario

Tabla 16. Pruebas - Administración de Ruta, Coordenadas y Horario

Descripción	Estado
El sistema cuenta con administración de ruta, coordenadas y horario.	Aprobado

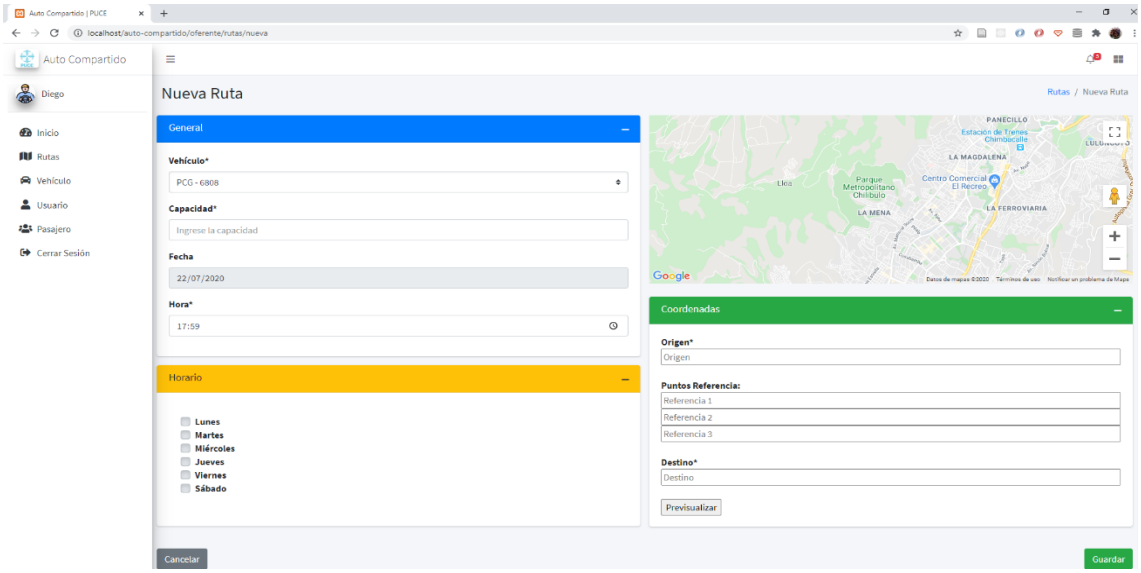

Ilustración 24. Administración de Ruta, Coordenadas y Horario

3.5.6. Usabilidad

Mediante la prueba del sistema realizada a usuarios seleccionados se puede tener en cuenta que, la plataforma es amigable para el usuario, principalmente en la fácil interacción entre la plataforma y el usuario final.

Por otra parte, también la flexibilidad en la planeación de las rutas mediando los Puntos de Referencia hacen que los usuarios tengan mayor facilidad de trazar el trayecto y elegir puntos relacionados con el Origen y Destino.

De igual manera, la parte visual es importante, ya que los usuarios se sienten cómodos con las interfaces de la plataforma, también el mapa es de gran ayuda para observar la ruta completa, lo que le da una mejor comprensión y facilidad de uso.

En cuanto a la seguridad de los usuarios, el administrador será capaz de bloquear a los usuarios, ya sea por mala utilización de la plataforma, o mal servicio que brinda o recibe; es así como a los usuarios les llamo la atención y se sienten mucho más seguro que la plataforma cuente con esta opción.

3.5.7. Requerimientos no Funcionales

Es importante destacar que la plataforma cuenta con requerimientos no funcionales que ya fueron mencionados, como el lenguaje de desarrollo, el framework y la base de datos, para que se pueda cumplir con uno de los requerimientos de la Dirección de Informática, que es la integración de la aplicación con los sistemas informáticos de la PUCE.

Por otra parte, cumple con estándares de calidad para el desarrollo de software, como es de rendimiento, seguridad, fiabilidad y mantenibilidad;

4. CAPITULO IV

4.1. CONCLUSIONES

Luego de haber culminado el proceso de diseño, desarrollo e implementación de la plataforma de Auto Compartido de la PUCE, se destacan las siguientes conclusiones:

1. Luego de observar los requerimientos acerca del programa de campus sostenible PUCE VIVA, se puede constatar el desarrollo de una plataforma digital para solventar las necesidades de movilización de los miembros de la Comunidad Universitaria; por otra parte, el beneficio que brinda el poder simplificar el uso de vehículos en la ciudad, dando importancia para la protección del medio ambiente y fomentando la concientización del bien común.

2. Como resultado del diseño y desarrollo de la plataforma, se concluyó que las herramientas empleadas en este proyecto como: Xampp, MySQL, CodeIgniter, API de Google Maps, Visual Studio Code, han sido de gran ayuda para la solución ingenieril a la problemática planteada. Es así como cada una de ellas respectivamente aporta con varios aspectos importantes como son el ambiente de desarrollo, servidor local, lenguaje de programación (PHP), librerías, servicios, clases que facilitan el desarrollo, plasmando mediante la arquitectura de tres capas, la plataforma para la gestión de rutas, coordenadas, horarios en el aplicativo amigable para los usuarios de la PUCE.

3. La aplicación permite la visualización de rutas por medio de la gestión de puntos referenciales colocados por parte de los usuarios en el mapa de Google Maps, de tal manera que cumple totalmente el requerimiento del trazado de una ruta con el fin de mejorar la navegabilidad y experiencia de usuario en la aplicación.

4. La metodología de desarrollo Cascada, empleada en este proyecto, está enfocada en el desarrollo secuencial, por lo tanto, luego del análisis de requerimiento se procede al diseño tanto arquitectónico como de interfaces para dar vida a la plataforma, posteriormente a someterse a pruebas en las

cuales constatamos el cumplimiento tanto de los requerimientos funcionales, como no funcionales que tenía como punto de partida el proyecto.

5. Es importante recalcar, el uso del framework CodeIgniter, el cual aportó compatibilidad, ligereza, rapidez y facilidad de adaptación a la plataforma; de igual manera una buena interacción con la base de datos MySQL.

6. Es indispensable mencionar, el desarrollo de la plataforma cuenta con estándares de desarrollo dadas por la Dirección de Informática de la PUCE, como es la nomenclatura de clases, controladores, modelos y vistas. Haciendo que el código fuente del aplicativo sea claro y entendible. Por otra parte, la plataforma cuenta con la parametrización necesaria, obteniendo buenas prácticas de programación y sobre todo pensando en la escalabilidad de esta.

7. Finalmente, se puede destacar que el uso de varios lenguajes en un entorno de desarrollo es posible, es así como la aplicación cuenta con el lenguaje de desarrollo principal PHP, y con la colaboración de JavaScript para el manejo de actividades complejas en la plataforma; a su vez estos dos lenguajes van de la mano con HTML, el cual es un lenguaje de etiquetas. Por último, el uso de clases de estilos de Bootstrap.

8. Por la aplicación de estándares y herramientas de desarrollo, la aplicación es compatible con los sistemas de información de la PUCE.

4.2. RECOMENDACIONES

Previo a finalizar, se debe tener en cuenta las siguientes recomendaciones:

1. El alcance del sistema está enfocado en contar con una plataforma web, sin embargo, la migración e integración con una aplicación móvil ayudaría a la portabilidad y accesibilidad del aplicativo, pese a esto la plataforma se adapta a los navegadores móviles teniendo una experiencia aceptable para el usuario.
2. Es importante tener el conocimiento necesario acerca de las clases y servicios que la Api de Google Maps brinda para la ayuda en el manejo de mapas y rutas; por otra parte, es fundamental tener conocimiento que luego de un cierto número de peticiones a los servicios que incorpora podrían generar costos.
3. Es recomendable implementar un control en tiempo real para saber exactamente el tiempo de duración de una ruta, esto con el fin de ayudar a visualizar e informar al usuario el inicio y fin del trayecto completo del recorrido.
4. Dado que el aplicativo es de uso netamente de la PUCE, es importante que se realice una integración a los sistemas propios de la universidad, haciendo que tenga durabilidad y sobre todo acogida en la Comunidad Universitaria.
5. Debido a la situación sobre el COVID-19, se debe realizar un estudio para lograr establecer un número máximo de pasajeros por ruta en el aplicativo; ya que es un tema delicado y que se debe tener en cuenta para el futuro.

4.3. ANEXOS

4.3.1. Definiciones, acrónimos y abreviaturas

6. PUCE: Pontificia Universidad Católica del Ecuador
7. Trazabilidad: Serie de procedimientos que permiten seguir el proceso de evolución de un producto en cada una de sus etapas.
8. Api: Conjunto de subrutinas, funciones y procedimientos que ofrece una biblioteca para ser utilizada por algún software.
9. Api de Google Maps: Servicios para la administración de geo – localización creado por Google
10. SRS: Documento de Especificación de Requerimientos de Software.
11. Oferente: Miembro de la comunidad universitaria de la PUCE que brinda el servicio de Auto Compartido a través de su vehículo. A su vez este también podría ser un pasajero.
12. Pasajero: Miembro de la comunidad universitaria que usara el servicio de Auto Compartido.
13. Administrador: Miembro de la comunidad universitaria que administra los parámetros generales de la aplicación y generación de reportes.
14. Rutas: Camino determinado que un Oferente determina.
15. Puntos: Lugares específicos por los cuales pasara una determinada ruta.
16. Framework: Es un entorno de trabajo, que utiliza conceptos, prácticas y criterios estandarizados para el desarrollo de software.
17. Back – End: Parte del desarrollo de software que se encarga de la lógica de la aplicación.
18. Front – End: Parte del desarrollo de software que se encarga de la interacción entre la aplicación y el usuario.
19. CodeIgniter: Framework de desarrollo de aplicaciones web, mediante el lenguaje de programación PHP, que utiliza MVC.
20. PHP: Lenguaje de código abierto para desarrollo web.
21. MVC: Metodología de diseño de software encargado de desglosar la lógica del negocio y cliente en tres niveles de abstracción o bloques estructurales.

22. Bootstrap: Framework multiplataforma enfocado en el diseño de aplicaciones web que permite cambiar la adaptabilidad de componentes y objetos utilizados en el desarrollo web. (Recio García, 2016).
23. JQuery: Biblioteca multiplataforma para añadir funcionalidades con técnica Ajax a componentes HTML facilitando la manera de trabajar con ellos e implementar animaciones con métodos y funcionalidades de JavaScript.
24. JQueryUI: Librería de componentes que comúnmente es usado en la parte del Front-End de un aplicativo web que permite añadir e implementar funcionalidades dinámicas y efectos visuales a diversos componentes. (Recio García, 2016).
25. Base de Datos: Estructura o repositorio para el almacenamiento de datos de diferentes tipos enfocado en la contención de información con accesibilidad a gestionar de manera completa e integra su contenido. (Beynon-Davies, 2014).
26. MySQL: Sistema de administración de base de datos relacional que trabaja con lenguaje de consultas estructurado o SQL.
27. Versionamiento: Proceso utilizado en el desarrollo de software para controlar y gestionar los cambios y versiones del código aplicado en toda la arquitectura del aplicativo.
28. GitHub: Plataforma que permite gestionar el versionamiento de código fuente.
29. Servidor Web: Es el medio capaz de ejecutar software del lado del servidor en donde agrupa protocolos como el http y https, estándares que permiten la conexión e interacción entre sistemas informáticos, generando respuestas a distintas peticiones que se las realiza por parte del lado del cliente. (Zofío Jiménez, 2013).
30. Apache: Servidor HTTP web para la ejecución de aplicativos webs.

4.4. BIBLIOGRAFÍA

Garces, P., & Naranjo, E. (2013). *Análisis de los estudios de impacto de tráfico vigentes en la cuidad de Quito bajo el enfoque del modelo Manhein (Disertación de grado).* Obtenido de http://repositorio.puce.edu.ec/bitstream/handle/22000/6039/T-PUCE-6293.pdf?sequence=1&isAllowed=y

Agencia Metropolitana Tránsito. (2019). *AMT*. Obtenido de Hoy no circula: http://www.amt.gob.ec/index.php/pico-placa-homepage.html

Beynon-Davies, P. (2014). *Sistemas de bases de datos.* Reverté.

Corporation, O. (2020). *MSQL*. Obtenido de Servicios: https://www.mysql.com/cloud/

Developers, G. (s.f.). *Google Maps Platform.* Obtenido de Directions: https://developers.google.com/maps/documentation/javascript/reference/directions#DirectionsService

M. A. (20 de Enero de 2015). *MIALTOWEB*. Obtenido de Definición de aplicación web: http://mialtoweb.es/definicion-de-aplicacion-web/

Maldonado Guerrero, J. (2016). *DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA WEB DE SEGUIMIENTO Y EVALUACIÓN DE LAS PRÁCTICAS PRE-PROFESIONALES PARA LA FACULTAD DE INGENIERÍA ESCUELA CIVIL DE LA PUCE (Disertación de Grado) Quito.* Obtenido de http://repositorio.puce.edu.ec/bitstream/handle/22000/12562/Tesis_Teoria.pdf?sequence=1&isAllowed=y

Moreno Pérez, J. (2015). *Programación orientada a objetos.* RA-MA Editorial.

Real Academia Española. (2019). *RAE*. Obtenido de https://dle.rae.es/srv/search?m=30&w=ruta

Recio García, J. (2016). *HTML5, CSS3 y JQuery.* España: RA-MA Editorial.

Red Hat. (2019). *INTEGRACIÓN*. Obtenido de ¿Qué es una API?: https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces

Rozo Nader, J. (15 de Febrero de 2014). *Universidad de la Rioja.* Obtenido de Metodología de Desarrollo de Software: MBM: https://dialnet.unirioja.es/descarga/articulo/5980502.pdf

Secretaría de Ambiente Alcaldía de Quito. (mayo de 2018). *Informes.* Obtenido de Informe Calidad del Aire de Quito 2017: http://www.quitoambiente.gob.ec/ambiente/images/Secretaria_Ambiente/red_monitoreo/informacion/ICA2017.rar

The Institute of Electrical and Electronics Engineers, Inc. (2000). *IEEE Recommended Practice for Architectural Description of Software-Intensive Systems- IEEE Std 1471-2000.* IEEE Computer Society, New York. Obtenido de IEEE Recommended Practice for Architectural Description of Software-Intensive Systems.

Universidad San Francisco de Quito. (2016). *Autocompartido*. Obtenido de http://www.usfq.edu.ec/sobre_la_usfq/oficinainnovacion/Paginas/autocompartido.aspx

Vaswani, V. (2010). *Fundamentos de PHP.* McGraw-Hill Interamericana.

Zofío Jiménez, J. (2013). *Aplicaciones web.* Macmillan Iberia, S.A.