PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

FACULTAD DE INGENIERIA

ESCUELA DE SISTEMAS Y COMPUTACIÓN

DISERTACIÓN PREVIA A LA OBTENCIÓN DEL TÍTULO DE INGENIERO DE SISTEMAS Y COMPUTACIÓN

ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE ADMINISTRACIÓN PARA UN CONSULTORIO MÉDICO. CASO DE ESTUDIO: CONSULTORIO DE CIRUGÍA ESTÉTICA Y RECONSTRUCTIVA

AUTOR: ALEXEI NICOLAS RAMOS RIVERA

DIRECTOR: ING. OSWALDO ESPINOSA

QUITO, ECUADOR

2019 - 2020

Agradecimientos

A mis padres, quienes, con su constante apoyo, compresión, soporte emocional y mucho esfuerzo han sabido guiarme en mis años de formación académica y personal. Les estaré infinitamente agradecido toda mi vida, por ser un gran ejemplo a seguir, deseo llegar a ser un profesional igual o más aclamado que ustedes, son el pilar fundamental de mi desarrollo como persona y el eje de mi mundo, espero poder cumplir con todas las expectativas que depositaron en mí, así como ustedes han sabido cumplir a cabalidad el rol de padre y madre, trabajadores, amorosos, responsables, personas capaces de luchar por sus ideales y cumplir las metas que han fijado en su vida.

Nunca seré capaz de expresar cuan agradecido estoy con ustedes, por todo lo que son, lo que han hecho y lo que significan para mí, por ahora diré gracias desde el fondo de mi alma. Gracias por ser mis confidentes, mis mentores, mi familia.

A mi hermana, quien ha tenido que soportar mis quejas, mis conflictos internos, por ser ese faro de luz en medio de la oscuridad, quien cuando me sentía triste, agobiado o agotado supo acercarse a mí, cuestionar mi estado actual, guiarme hacia una mejor toma de decisiones y apoyarme en mi trayecto.

En ti veo alguien con quien puedo conversar y a quien le puedo confiar mis pesares, compartir aquellas noticias con las cuales no se lidiar y saber que me brindaras el mejor consejo, una opinión asertiva que me guie a la mejor solución.

Aprecio con todo mi ser que no pierdas las casillas conmigo y aun si lo hicieras, entender que cuando te necesite estarás para mí, asi mismo, quiero expresar que siempre estaré para ti.

A mi abuelita quien supo escucharme y con su sonrisa tranquiliza mis temores, por estar presente en los momentos más importantes de mi vida y dar amor incondicional a sus hijos, hermanos, nietos y allegados.

A mi tía Amadita, quien fue una segunda madre para mí, por cuidarme cuando niño, protegerme al ser infante, recibirme los fines de semana en su hogar y compartir tan gratos recuerdos que me acompañan en mis peores días, por ser la representación de bondad, amor, dedicación y comprensión.

A mi pareja, quien me conoció en los momentos cuando más inseguridad sentí, cuando las dudas sobre la carrera y mi futuro me aquejaban, quien supo apoyarme y preocuparse por mí, quien me acompaño en desvelos, risas y con quien he pasado la transición de un estudiante a un profesional. Espero poder seguir compartiendo el resto de etapas por venir, crecer continuamente juntos.

A mis amigos quienes me ayudaron en este trayecto, con sus ocurrencias y soporte, compartimos momentos increíbles, con quienes se formaron lazos inquebrantables.

Dedicatoria

El presente trabajo de titulación está dirigido a todas y cada una de las personas que me acompañaron durante este trayecto.

A aquellas personas que ya no están junto a mí, por las lecciones brindadas.

A quienes se encuentran a mi lado, por compartir su vida conmigo.

A mi familia por ser el mejor regalo del universo a mí.

A mi pareja, por permitirme vivir a plenitud esta y las futuras etapas de la vida como un equipo.

A mí mismo, por haber superado contratiempos, dudas y temores, permitiéndome crecer y mejorar continuamente, de tal manera que hoy puedo estar aquí.

Resumen

Conforme el desarrollo de nuevas tecnologías ha avanzado de forma explosiva, la integración de las mismas a diversas áreas con necesidad de automatizar y simplificar procesos ha tomado tendencia. En la actualidad existen diversas herramientas para desarrollar aplicativos tanto móviles como de escritorio, en línea como fuera de línea, esto permite el escoger la herramienta más adecuada a las necesidades del cliente o parte interesada y aplicar metodologías de desarrollo acorde a el grado de participación e involucramiento del cliente.

La presente investigación, desarrollo e implementación de un aplicativo optimo y eficiente para la administración de un consultorio médico expone el proceso de selección de las herramientas que mejor se adapten a la naturaleza del proyecto y a los requisitos del cliente. Para manejar un entorno en línea con un panel administrativo se optó por Laravel debido a sus prestaciones, Java debido a su predilección por desarrolladores a nivel mundial para el manejo de sistemas de facturación e inventarios, la metodología aplicada fue Cascada debido al poco grado de involucramiento del cliente en el proceso de desarrollo y las reuniones limitadas por su disponibilidad de tiempo.

En el proyecto de disertación se expondrán los conceptos más relevantes tras cada herramienta utilizada, su integración con el proyecto y los beneficios que estas brindan, de igual manera se detallaran los esquemas, requerimientos y el debido proceso detrás de la implementación y el progreso de cada módulo de desarrollo con sus funcionalidades generadas y los resultados que estas presentan, así como la debida aceptación y aprobación de los mismos por el cliente.

Tabla de contenido

Tabla de Ilustraciones

Capítulo 1 – Aspectos Introductorios

Hoy en día conforme avanza la tecnología y automatización de procesos, empresas, entidades gubernamentales, financieras, casas de salud, consultorios entre muchos otros buscan por reducir gastos operativos y perseguir una mayor rentabilidad, como apoyo a esta necesidad se aplica el desarrollo de proyectos de software, pues los mismos por concepto responden a las diversas necesidades que se presenten, implementando soluciones eficaces y eficientes.

La medicina y sus ramificaciones presentan una facilidad para la aplicación de desarrollo de software debido a sus numerosos procesos simples, mismos que pueden ser abordados y automatizados al estar compuestos de protocolos y secuencias simples de definir, pero de suma importancia.

El manejo de procesos administrativos como reservas de citas, control de inventario de productos medicinales o indumentaria, registro de historiales clínicos y administración de empleados, sucursales.

Una oportunidad de distinción y mercado se encuentra en el sector de la salud, en este se puede apreciar una oferta principalmente extranjera, el objetivo de este proyecto de disertación es la creación e implementación de un sistema de administración para un consultorio médico. Se parte de una integración de distintas áreas vitales para el correcto funcionamiento del establecimiento, al realizar una consulta previa se observó un programa de manejo de historias clínicas, sin embargo, el mismo producto no incluía áreas de manejo de pedido de insumos, administración de empleados, procesos de reserva de citas o notificaciones automatizadas. Dando como resultado la destinación de recursos económicos entre otros para cubrir la administración de las áreas previamente mencionadas. Esto resulta en una perdida directa para el consultorio, debido a esta necesidad se presenta una propuesta de un sistema más completo capaz de administrar un mayor porcentaje de las áreas clave y automatizar de la mejor manera los procesos para agilizar los tiempos de respuesta entre cliente y usuario.

En el presente capítulo se aclararán los objetivos del mismo y las razones de su importancia respecto a la resolución de las necesidades previamente mencionadas y los siguientes capítulos:

- Aspectos Introductorios, donde se trata acerca de la motivación, los objetivos del proyecto y el alcance del mismo.
- Fundamento teórico, aquí se explican todos los conceptos involucrados durante el desarrollo del presente proyecto, los conceptos auxiliares para reforzar los conceptos principales, descripción de procesos y metodología.

- Caso de Estudio, se levantan los requerimientos y plantea la estrategia a seguir para el desarrollo del proyecto.
- Fases de diseño, planificación, codificación, pruebas y postmortem.
- Implementación y entrega del producto, una vez terminado el proceso de desarrollo, procedemos a registrar la implementación y prueba del sistema en el ambiente de producción real.
- Aceptación del producto y documentación de respuestas.
- Conclusiones y Recomendaciones, registramos los comentarios y consejos sobre el producto tras implementarlo y entregarlo.

1.1 Datos de la Organización

Nombre	Consultorio de cirugía estética y reconstructiva.
Actividad	Salud, belleza, atención a pacientes.
Ubicación	Eloy Alfaro e Italia, Edificio Fortune Plaza, Quito - Ecuador.
Características	Historias clínicas, manejo de inventarios, administración de personal y distribuidores, manejo de sucursales y cotización de servicios.

1.2 Justificación

Actualmente hay diferentes tipos de tecnologías las cuales continuaran en aumento, permitiendo la implementación de sistemas más dinámicos para la administración de negocios. Se espera una adaptación a las necesidades del consumidor creando sistemas completos y competitivos en un mercado que se expande constantemente. Es imprescindible el asegurar la información de la parte interesada, evitando la pérdida o robo de la misma, así como una documentación de la arquitectura, diseño e implementación del sistema que será integrado. Con el fin de proporcionarles a los usuarios, encargados y administradores, la facilidad de registro, consultas y manejo de historias clínicas, proveedores, pedidos de inventario, facturación y reserva de citas, dando como resultado la certeza de que se está asegurando lo mejor posible la calidad del producto.

Se debe innovar continuamente para poder ser competitivo, este proyecto permite la adaptación de un sistema más completo, con la finalidad de reducir costos en adquisiciones de diversas licencias de software y adaptación entre tecnologías, vinculando todo en un solo producto, permitiendo el desarrollo de una herramienta unificada que facilita el manejo de información y administración del consultorio de especialidad.

1.3 Planteamiento del Problema

Es necesario implementar productos de software para administración de hospitales, consultorios médicos, clínicas privadas, mismas que requieren de soluciones completas y no parciales como se aprecia en este tipo de establecimientos.

En muchas ocasiones se presentan casos de problemas de compatibilidad entre módulos, pues un programa maneja historias clínicas, pacientes y empleados, mas no permite el manejo y administración de insumos, proveedores o facturación.

Aquella situación genera desconformidad pues requiere de recursos sean monetarios o humanos para adaptar soluciones en el conflicto de tecnología.

1.4 Alcance

El presente trabajo de disertación desarrollará un programa para facilitar la administración de un consultorio médico, mismo que incluirá las funcionalidades de manejo de empleados, proveedores, inventario, integración con un módulo de facturación, registro y agendación de citas y consultas, manejo de pedidos, administración de historias clínicas.

Por este motivo el proyecto se dará por finalizado al entregar el programa funcional con la implementación de todos los módulos mencionados, reflejando el cumplimiento de los requisitos que fueron levantados y documentados respectivamente en el documento de especificación de requerimientos de software.

A esto se anexará cualquier sugerencia, observación o solicitud de implementación, modificación o eliminación de módulos, mismas que serán analizadas por el desarrollador y acorde a su viabilidad e impacto al proyecto se procederán a aprobar o rechazar para el ciclo o etapa correspondiente.

Una vez finalizado el proyecto la parte interesada el Doctor propietario del consultorio de cirugía estética y reconstructiva emitirá u extenderá una carta de aceptación y conformidad del proyecto, así como, un certificado de entrega y recepción del programa y la documentación respectiva del mismo.

1.5 Objetivos

1.5.1 Objetivo General

- Análisis, diseño e implementación de un sistema de administración de un consultorio médico. Caso de estudio: consultorio de cirugía estética y reconstructiva.

1.5.2 Objetivo Específico

En este trabajo, se tienen como objetivos:

- Analizar los requisitos y el entorno donde se implementará el proyecto
- Diseñar el sistema para la administración de un consultorio médico, permitiendo la automatización de procesos con una interfaz amigable para el usuario.
- Implementación del sistema capaz de permitir el correcto funcionamiento del establecimiento, agilizando procesos y mejorando el nivel de calidad de atención y actividad laboral.
- Diseñar buenas prácticas para la seguridad y manejo de un consultorio médico.
- Proporcionar buenas herramientas para facilitar el desempeño laboral.
- Promover el concepto de eficiencia.

Capítulo 2 – Marco Teórico

Dada la connotación del presente proyecto se delimitan algunos de los conceptos, criterios, modelos, metodologías compatibles con la adecuada elaboración del mismo.

Es imperativo el establecer de manera clara el tipo de proyecto a desarrollar, por ello se delimita específicamente la arquitectura de software, la metodología de programación, diseño, planificación y evaluación del proyecto de disertación, así como herramientas tecnológicas de las cuales dependerá la optimización y automatización de procesos acompañados de sus respectivos conceptos.

El presente capítulo, por ende, trata acerca de la metodología seleccionada, introducción y forma de aplicación e igualmente acerca de las herramientas a utilizar durante el desarrollo e implementación del mismo.

La ingeniería de software está formada por un proceso, un conjunto de métodos (prácticas) y un arreglo de herramientas que permite a los profesionales elaborar software de cómputo de alta calidad. (Pressman, 2010, pág. 28)

De esta cita se puede inferir la importancia de seleccionar una metodología adecuada al tipo de cliente, el grado de envolvimiento que tendrá con el proyecto según la disponibilidad horario de la parte interesada, así como los requisitos solicitados. El escoger una metodología nos brinda una ventaja muy clara, organizar el trabajo acorde a un marco de trabajo que nos ayuda a definir las etapas o fases a seguir durante el desarrollo, establecer fechas de entrega y tiempos claros de desarrollo y generar documentación respectiva, clara y concisa.

2.1 Metodología de Desarrollo

El desarrollo de software no es una tarea fácil. Prueba de ello es que existen numerosas propuestas metodológicas que inciden en distintas dimensiones del proceso de desarrollo. (JH Canós, 2012)

Se puede escoger entre metodologías orientas a la documentación extensa, definidas por ciclos de desarrollo sin mucha flexibilidad de cambio, que tienden a ser clasificadas como antiguas o metodologías agiles, orientadas a adaptar la forma de trabajar acorde a las condiciones del proyecto, esto nos brinda mayor flexibilidad a cambios.

Este proyecto tiene un enfoque más tradicional al aplicar la metodología en cascada, debido a la poca disponibilidad de tiempo de la parte interesada y la rigurosidad en el aspecto de documentación de la misma, así como en necesidad de finalización de cada etapa de desarrollo.

2.1.1 Análisis

Para este caso de estudio en particular se tiene un contexto en el que no existe la participación activa del cliente, esto se debe a una rutina compleja y poca o nula disponibilidad de tiempo, quien no tiene una idea firme sobre lo que el sistema debería hacer y los procesos que debería manejar, por lo cual se asume que los requerimientos podrían tener variaciones, mismas que serán receptadas y analizadas para definir su viabilidad, no obstante, el enfoque en la recopilación de requisitos nos permitirá el minimizar posibles variaciones.

El desarrollo deberá ser sometido a evaluación periódica durante las etapas de entrega de resultados con el cliente y por parte del tutor de desarrollo para verificar el cumplimiento de fechas de entrega presentadas a la parte interesada.

Se escogió la metodología en cascada porque tiene como objetivo generar eficiencia en el proceso de codificación y proporcionar a los clientes una documentación exhaustiva y clara acerca del desarrollo del proyecto.

Las ventajas de cascada, secuencial o ciclo de vida incluyen enfoque metodológico, robustez, resistencia, cumplimiento de fechas, planificación de pruebas y una documentación de cada etapa realizada; así mismo, genera buenas prácticas de desarrollo en el profesional y se adapta a la poca disponibilidad de reuniones del cliente, mostrando solo finalizaciones de ciclo y generando el informe del mismo.

2.1.2 Desarrollo en Cascada

Es el enfoque metodológico que ordena rigurosamente las etapas del proceso para el desarrollo de software, de tal forma que el inicio de cada etapa debe esperar a la finalización de la etapa anterior. Al final de cada etapa, el modelo está diseñado para llevar a cabo una revisión final, que se encarga de determinar si el proyecto está listo para avanzar a la siguiente fase. De este modo se tendrá un desarrollo ordenado y verificable del avance del proyecto de disertación.

Se plantea el siguiente avance de cada ciclo:

1. Análisis de requisitos.
2. Diseño del sistema.
3. Diseño del programa.
4. Codificación.
5. Pruebas.
6. Implementación o verificación del programa.
7. Mantenimiento.

De esta manera se podrá corregir errores, verificar avances y documentar el proceso de manera entendible y estructurada.

2.2 Herramientas

Se presentan las herramientas que se utilizarán para el desarrollo del proyecto, la propuesta inicial fue realizada en base a una integración de Laravel con Java, la cuasi totalidad de módulos serían desarrollados en Laravel mientras que los módulos de facturación se implementarán en Java, se utilizará MySQL y PostgreSQL como base de datos.

Tras una reunión con la parte interesada para mostrar la propuesta, esta fue aceptada con el único cambio en la base de datos, se solicitó se implemente en MySQL. Debido a la ventaja que tiene Laravel para adaptarse y trabajar con varias bases de datos sin cambiar la estructura del programa, utilizando eloquent, se procedió a aceptar el cambio, sin embargo, se deja como posibilidad la configuración de PostGreSQL lista para puesta en marcha.

Es necesario el aclarar el motivo de uso de cada herramienta de entre todas las disponibles en la actualidad, presentar sus ventajas, desventajas y como estas se integran de mejor manera a las necesidades del cliente.

2.2.1 Base de Datos

Se deberá ocupar una base de datos relacional, orientada a la web, en este caso el aplicativo que se va a realizar, misma que debe ser de licencia abierta, de modo que no genere un costo de renovación de licencia.

La base debe contar con soporte y actualizaciones de parte de su comunidad, así mismo, debe ser independiente de hardware dedicado, que su documentación sea extensa y clara, con facilidad para encontrarla o con un foro de ayuda perteneciente a una comunidad activa, como parte de sus requisitos funcionales, debe tener escalabilidad, adaptabilidad, su desempeño debe estar orientado a consultas y transaccionalidad.

Es necesario comparar entre las diversas opciones de sistemas gestores de bases de datos para evaluar el rendimiento, operabilidad y satisfacción de experiencia de usuarios.

341 systems in ranking, December 2018

Rank Dec 2018	Rank Nov 2018	Rank Dec 2017	DBMS	Database Model	Score Dec 2018	Score Nov 2018	Score Dec 2017
1.	1.	1.	Oracle	Relational DBMS	1283.22	-17.89	-58.32
2.	2.	2.	MySQL	Relational DBMS	1161.25	+1.36	-156.82
3.	3.	3.	Microsoft SQL Server	Relational DBMS	1040.34	-11.21	-132.14
4.	4.	4.	PostgreSQL	Relational DBMS	460.64	+20.39	+75.21
5.	5.	5.	MongoDB	Document store	378.62	+9.14	+47.85
6.	6.	6.	IBM Db2	Relational DBMS	180.75	+0.87	-8.83
7.	7.	↑ 8.	Redis	Key-value store	146.83	+2.66	+23.59
8.	8.	↑ 10.	Elasticsearch	Search engine	144.70	+1.24	+24.92
9.	9.	↓ 7.	Microsoft Access	Relational DBMS	139.51	+1.08	+13.63
10.	10.	↑ 11.	SQLite	Relational DBMS	123.02	+0.31	+7.82

Ilustración 1: Top 10 sistemas en el ranking 2017-2018

Java Hispano, Portada. (diciembre 2019).

2.2.1.2 MySQL

Es uno de los gestores de base de datos más populares, esto es debido a su extensa documentación e integración como paquete estándar de descarga cuando ocupamos o instalamos otras herramientas como XAMPP o WampServer.

MySQL implementa un modelo de cliente servidor, utiliza sintaxis SQL para su operabilidad, pese a ser desarrollado en C y C++.

SQL fue desarrollado a comienzos de los años 70, su modelo base fue relacional con gran influencia de la estructura que utilizaba IBM. Durante esa década logro sobresalir frente a VISAM e ISAM, que fueron declarados obsoletos, esto le permitió posicionarse de forma relativamente sencilla en el mercado.

Posteriormente paso a ser estándar de la organización internacional de normalización mejor conocida como ISO en 1987.

A través de SQL le damos instrucciones al servidor para realizar diversas operaciones tales como:

- Consulta de datos: solicitar información específica de la base de datos con la que trabajamos.
- Manipulación de datos: agregar, eliminar, modificar, ordenar y otras operaciones para manipular la data presente en la base de datos.
- Identidad de datos: nos permite definir el esquema de datos y las relaciones de tablas entre sí, de igual manera podemos realizar conversiones en los tipos de datos de campos presentes en las tablas de la base de datos.

- Control de acceso a los datos: se basa en el nivel de seguridad y acceso a los datos de la base, quien puede visualizar e interactuar con los datos, el rango de posibilidades de interacción y funciones de recuperación de información o anulación de cambios.

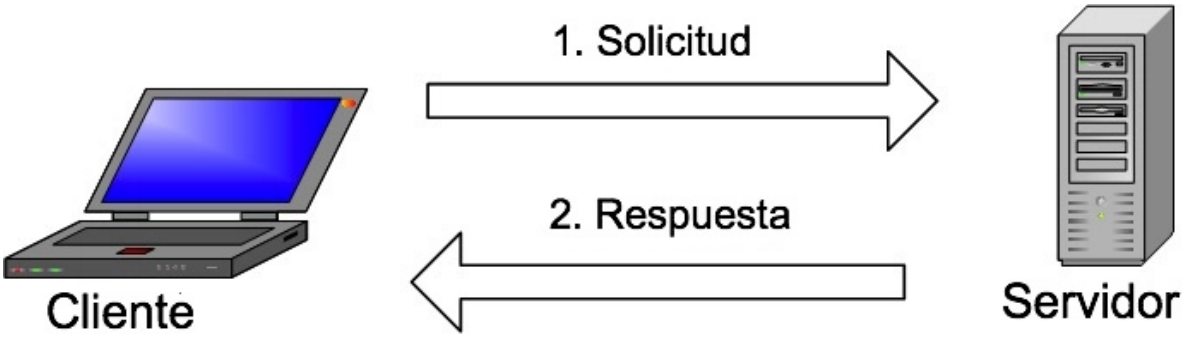


Ilustración 2: Estructura básica cliente-servidor

Ramos, A. (diciembre 2019).

El presente gráfico indica la estructura más básica de comunicación entre uno o más clientes y un servidor a través de una red específica.

Se envía una solicitud desde el lado del cliente y el servidor la receptara y devolverá un resultado a la misma siempre que ambas partes puedan procesar la instrucción.

En síntesis:

1. MySQL crea una base de datos que contiene diversas tablas con el fin de almacenar y manipular datos, definiendo la relación de cada tabla.
2. Los clientes pueden realizar solicitudes escribiendo instrucciones SQL específicas en MySQL.
3. La aplicación del servidor responderá con la información solicitada y esta aparecerá frente a los clientes.

MySQL se encuentra posicionado como el segundo motor de bases de datos más descargado del año 2018, esto nos demuestra no solo su popularidad sino su preferencia para implementación de proyectos a nivel organizacional.

Las características que lo posicionan en este lugar son:

1. Licencia libre, lo que permite su descarga, el instalador es intuitivo de ocupar y sencillo de descargar.
2. Gran cantidad de cursos, capacitaciones, material de autoaprendizaje y ejemplos operativos como entorno de aprendizaje.

2.2.1.3 PostgreSQL

Referenciado igualmente como Postgre, es un sistema de gestión de base de datos relacional orientado a objetos posicionado como uno de las opciones más populares llegando al cuarto lugar en el ranking comparativo, en parte debido a su capacidad de aceptar consultas no relacionales en el mismo por medio de JSON.

Al ser un sistema de código abierto y gratuito, se convierte en una opción cómoda y accesible tanto para estudios por las prestaciones que ofrece, como para negocios y empresas pequeñas, medianas y grandes.

El desarrollo de este sistema no es dirigido o manejado por una empresa o persona individual, sino por medio de una gran comunidad global de desarrolladores cuyo enfoque es mejorar constantemente las funcionalidades y operabilidad del presente sistema gestor de base de datos.

Dos puntos importantes a destacar son:

1. Es posible ejecutar optimizaciones de rendimiento avanzadas, funcionalidad presente principalmente en sistemas de bases de datos comerciales, como Oracle o SQL Server.
2. Posee tipos de datos o data types.

Su origen está directamente vinculado al Dr. Michael Stonebreaker y uno de sus tantos proyectos Ingres durante 1980, el nombren original del proyecto era Post Ingres, siendo combinado eventualmente para formar PostGre, la primera versión disponible para el público fue lanzada en el año 1989.

Su desarrollo prosperó hasta que en el año 1994 el equipo original del sistema se separó, al ser un proyecto de licencia libre el soporte para el mismo fue retomado para incluir soporte a SQL en 1995.

PostgreSQL obtuvo su denominación actual en 1997 en conjunto con el lanzamiento de sexta versión, a partir de este momento se comienza a formar la comunidad de desarrollo actual, que brinda soporte, respaldo y un continuo desarrollo.

Este sistema no cuenta con interfaz gráfica por lo que la misma debe ser seleccionada por el usuario, entre las diversas opciones el presente proyecto fue administrado con PgAdmin.

2.2.2 Lenguajes de Programación

Se puede describir un lenguaje de programación como una serie de instrucciones vinculadas a un lenguaje formal, estructurado con reglas y algoritmos, su función es permitir el desarrollo de acciones consecutivas u ordenes definidas por el programador. Estas acciones permiten controlar el comportamiento lógico del

ordenador.
Se puede decir que es el método de vinculación entre el desarrollador y el computador, delimitando de manera clara que datos manejara el programa, la forma de almacenamiento y uso de datos, entre otras funciones lógicas operativas.

Concluyendo es posible decir que un lenguaje de programación es sistema estructurado de comunicación.

2.2.2.1 Tipos de Lenguaje de Programación

Existen diversas clasificaciones para los lenguajes de programación, mismas que nos permiten agrupar y clasificarlos de mejor manera.

1. De bajo nivel: estos lenguajes se caracterizan por estar directamente relacionados u orientados a la máquina, lo cual crea un puente entre hardware y software, para manejar este tipo de lenguajes es necesario un nivel de conocimiento de hardware elevado. Se puede subdividir aún más este nivel en dos subgrupos:
    a. Lenguaje de máquina: Se puede considerar como la versión más primitiva de los lenguajes de programación, siendo manejado por secuencias de código binario, los cuales son leídos e interpretados por el ordenador.
    b. Lenguaje ensamblador: Es considerado como el primer enfoque o acercamiento de convertir el lenguaje de máquina a una sintaxis similar a la de los seres humanos, acercando ligeramente la forma de comunicarnos con el computador, se caracteriza por ser almacenado como texto conformando una serie de instrucciones u ordenes que están sujetas a un flujo establecido y ejecutables por el equipo.
    Es necesario convertir este lenguaje por medio de un ensamblador el cual convierte las instrucciones a secuencias binarias, generando códigos compactos y eficientes.
2. De alto nivel: Es un lenguaje basado en instrucciones más similares a la sintaxis de lenguajes humanos, llevándolo a un nivel más cercano que no es posible alcanzar por medio del lenguaje ensamblador, esto se puede comprobar al permitir el uso de instrucciones más intuitivas y sencillas de entender desde el elemento humano.
    Este nos permite escribir programas en idiomas conocidos como son el inglés, ruso, español etc. Se debe recalcar que el inglés es el idioma más utilizado en países occidentales. Estas instrucciones o líneas de código son traducidas a lenguaje de máquinas por medio de compiladores o traductores.
    a. Traductores: se encargan de traducir programas escritos en un lenguaje de programación, como ejemplo C#, al lenguaje de máquina,

su característica operativa es que ejecutan el código a medida que es traducido.

b. Compiladores: su función es similar a los traductores, con la diferencia clave que primero debe traducir todo el programa o líneas de código, permitiendo una ejecución rápida y es posible su almacenamiento para reusarse posteriormente sin repetir la traducción.

En base a los lenguajes de programación se han desarrollado diversos tipos de software de programación cuyas funciones cumplen un rol especifico como son depurar, empaquetar, mantener código, crear programas, editar líneas de código entre otros. Algunas agrupaciones de software son:

1. Editores de código o texto: son programas que nos permiten escribir líneas de código en muchos casos se auto completan a medida que vamos escribiendo y nos muestran errores sintácticos o bloques, funciones por cerrar.

   Un ejemplo de este tipo de software es Sublime o Atom que tienen compatibilidad para completar código y una versión más simple es Notepad++ o bloc de notas.
2. Compiladores: Carga todo el código y lo traduce para que pueda ser ejecutado por el computado, facilita el reusó del código leído ya que es almacenado y no debe repetir el proceso de traducción.
3. Depuradores: Nos ayudan a optimizar el tiempo de desarrollo y controlar la calidad de código creado gracias al monitoreo de la ejecución del programa, control de valores, variables, referencias a objetos en memoria, conexión a bases de datos, esto nos permite encontrar y corregir errores.
4. Interpretadores o traductores: Carga el código y lo traduce para que pueda ser ejecutado por el computador línea por línea.
5. IDE´s: Los entornos de desarrollo integrado, son aplicaciones informáticas que incluyen una serie de servicios para facilitar el proceso de desarrollo tales como:
   a. Autocompletado de texto
   b. Editor de código fuente
   c. Sistemas de control de versionamiento
   d. Gestión de conexiones a bases de datos
   e. Simuladores de equipos o dispositivos
   f. Depuradores
   g. Sistemas de integración de librerías externas

2.2.2.2 PHP

Es un lenguaje de programación de uso general con un mayor enfoque hacia la programación o desarrollo web, sus siglas representaban Personal Home Page sin embargo actualmente representa el inicialismo recursivo PHP: Hypertext Reprocessor.
Se suele procesar este lenguaje por medio de o en un servidor web gracias a un intérprete PHP que es implementado como un Daemon, módulo o CGI: common gateway interface cuya traducción es interfaz de entrada común.

Entre los usos de PHP apartados al desarrollo web, encontramos aplicaciones gráficas, control de drones entre aplicaciones en conjunto para estadísticas o cálculos matemáticos.

Cabe recalcar que este lenguaje es utilizado comúnmente por su conectividad junto con MySQL y PostgreSQL, siendo estas las opciones más populares de bases de datos para desarrollo web.

PHP es libre y se estima que más del 70% de páginas web han sido programados con la versión descontinuada de PHP cuyo soporte por parte de “The PHP Group”.

2.2.2.3 JavaScript

Es un lenguaje de programación interpretado y orientado a objetos, es utilizado principalmente como parte de un navegador web, siendo implementado para mejoras en páginas dinámicas e interfaces de usuario, no obstante cabe recalcar que puede ser ocupado en aplicaciones externas tales como documentos PDF y diversos widgets.

Su uso fue generalizado a tal punto que actualmente todo navegador web puede interpretar JavaScript.

Es importante destacar el envío y recepción de información a servidores por medio de tecnologías auxiliares como AJAX, esto previamente era imposible pues en su origen trabaja realizando operaciones en el lado del cliente.

2.2.2.3.1 JSON

Es el acrónimo de JavaScript Object Notation, cuya traducción es Notación de Objetos JavaScript, es decir de una derivación de la notación literal de objetos utilizados por JavaScript siendo un formato de texto simple utilizado en intercambio de datos.

En años recientes se ha comenzado a considerar a JSON como un formato independiente de JavaScript llegando a ser una alternativa para XML

2.2.2.4 HTML

Es el acrónimo de Hypertext Markup Language, cuya traducción es Lenguaje de marcas de hipertexto, siendo utilizado en el desarrollo de páginas web, nos permite definir una estructura base y el contenido de nuestras páginas, presentando diferentes elementos como imágenes, videos, áreas de texto, juegos entre otros.

Cabe destacar que todos los navegadores en sus versiones actuales utilizan HTML, siendo un estándar impuesto por el Consorcio World Wide Web, organización con el fin de estandarizar las tecnologías con una relación hacia la web.

En el presente trabajo se ha optado por la quinta versión de HTML, denominada HTML5.

2.2.2.5 CSS

Es el acrónimo de Cascading Style Sheets, cuya traducción es Hojas de estilo en cascada, siendo un lenguaje de diseño gráfico que nos facilita la creación y edición de la presentación de un documento estructurado. Debido a esto es ocupado en conjunto con HTML para establecer la estética a la estructura programada en HTML.

Es posible utilizar CSS en cualquier documento XML.

En el presente trabajo se ha optado por la tercera versión de CSS, denominada CSS3.

2.2.2.6 JAVA

Es un lenguaje de programación orientado a objetos, siendo uno de los más utilizados en la actualidad para el desarrollo de aplicaciones cliente-servidor web.

Es debido a esta cualidad que se ha optado por el presente lenguaje de programación.

Cabe respaldar dicha elección con la posibilidad de ejecutarse en cualquier arquitectura de computador, consola, dispositivos móviles desde su máquina virtual, esta brinda la capacidad de no tener que recompilarse si fue ejecutado en una plataforma y es pasada a otra.

La popularidad de Java es tal que muchas páginas y aplicaciones web no podrían funcionar sin Java.

2.2.3 Marcos de Trabajo

El presente trabajo será implementado bajo el marco de trabajo modelo vista controlador, MVC referente a su denominación Model View Controller, debido a su adaptabilidad a diversos entornos de desarrollo, mientras facilita la implementación

de medidas de seguridad correspondientes a las políticas de seguridad requeridas por el sistema.

Es importante destacar la función que cumple cada componente de este marco, siendo estos:

1. Modelo: se conecta a la base de datos del sistema y manejar las operaciones a nivel de envío y recepción de datos.
2. Controlador: se conecta con las vistas para recibir eventos y enviarlos al modelo para su envío y ejecución.
3. Vista: interfaz visual para que el usuario interactúe, envía eventos al controlador y muestra el resultado de la operación.

2.2.3.1 Laravel

Es uno de los marcos de trabajo o framework más utilizado para el desarrollo de aplicaciones web por su compatibilidad con PHP 5 y 7, se optó por este al ser de código abierto, gran parte de la funcionalidad de Laravel está vinculada a Symfony pues de este surgen muchas dependencias, ya que Laravel depende del desarrollo de las mismas para su correcta implementación.

La ideología de Laravel es crear una herramienta que permita escribir código de forma simple y elegante, un punto clave al momento del desarrollo del presente trabajo es la integración de puertas o Gate para manejar la seguridad de usuarios y accesos.

Laravel nos permite interactuar de forma sencilla con diversos motores de base de datos, siendo los principalmente utilizados MySQL y PostgreSQL, sin embargo cabe destacar de igual manera a SQL Server y SQLite, esto gracias al constructor de consultas integrado a la herramienta Object Relation Mapper, Mapeo de relaciones de objetos, ORM Eloquent.

2.2.3.2 Bootstrap

Se la puede definir como un conjunto de herramientas multiplataforma que permite mejorar la capacidad de diseño de una página o aplicación web, gracias a sus librerías y por ser de código abierto.

Dentro de las facilidades que esta herramienta nos facilita podemos encontrar diversas plantillas para manejar botones, formularios, menús de navegación, tipografía, animaciones, transiciones, entre muchas otras, sus elementos están basados en HTML y CSS, pero de igual manera podemos apreciar extensiones JavaScript para las animaciones y transiciones complejas. Esta herramienta se ve limitada a ser solo para desarrollo front-end, es decir, para diseñar interfaces de usuario.

Tanta es la popularidad que muchas páginas recopilan el trabajo de diversos desarrolladores y diseñadores en una amplia colección para poder descargar diversas plantillas, entre estos podemos apreciar diseños gratuitos, pagados y lite, este último tipo hace referencia a una muestra con funciones limitadas de plantillas pagadas para probar su implementación y funcionamiento con la finalidad de convencer a quienes lo prueban de comprar la versión completa o pagada.

Es debido a su popularidad y su integración con laravel que se optó por dicha herramienta en el presente proyecto.

2.3 Conceptos Relacionados al Desarrollo

2.3.1 Especificación de Requerimientos de Software

En la fase de requerimientos se realiza el recogimiento de las necesidades del cliente y el usuario. Que es lo que desea que el software realice. Hay que tener muy en cuenta que existen requerimientos funcionales y no funcionales.

Los requerimientos no funcionales hacen relación a las características del sistema que aplican de manera general como un todo, más que a rasgos particulares del mismo. Estos requerimientos son adicionales a los requerimientos funcionales que debe cumplir el sistema, y corresponden a aspectos tales como la disponibilidad, mantenibilidad, flexibilidad, seguridad, facilidad de uso, etc.

2.3.2 Especificación de Diseño de Software

Se procede con el diseño del producto software. Diseñar es el proceso creativo para decidir cómo construir el producto, en esta fase nos enfrentamos al reto de ir generando los artefactos o módulos correspondientes a las funcionalidades del sistema.

Cuando los requerimientos han sido recabados de manera eficiente, el diseño refleja de mejor manera las soluciones a las necesidades del cliente. En esta fase están completamente definidos los componentes a desarrollar, la arquitectura a emplear, la interacción de los módulos que dan forma al sistema, el mapeo de errores, entre otros.

En esta fase se definen los diversos paquetes que el programa tendrá como respaldo de documentación, siendo algunos de estos diagramas: Clases, Actividades, Paquetes y el diagrama de base de datos de Entidad-Relación. Finalmente definimos el plan de pruebas de integración.

2.3.3 Minuta

Es el documento que recoge puntos clave tratados en las reuniones establecidas al final de cada ciclo y en la entrevista para recopilar requerimientos, se estima el generar un total de 4 o 5 minutas, siendo estas referentes al inicio y cierre de ciclos, capacitaciones presentación de documentos.

2.3.4 Informe de pruebas

Es el documento que recopila los resultados de las pruebas que se realizan al sistema para verificar su correcto funcionamiento al final de cada ciclo de desarrollo.

2.3.5 Informe Postmortem

Es el documento que informa de los resultados generados al final de cada ciclo, complicaciones, posibles cambios de requerimientos, su factibilidad, impacto en el proyecto, así mismo permite destacar que se realizó con éxito, que se puede mejorar y que se debe corregir para los siguientes ciclos, es la base de desarrollo para ciclos posteriores al primero, pues nos sirve de guía para comprender como evolucionar el sistema y continuar con el proceso de desarrollo.

Capítulo 3 – Caso de Estudio

El presente capítulo está enfocado a describir de forma clara y detallada el proceso de recolección de requisitos funcionales del programa, acorde a la metodología escogida, siendo esta la metodología de desarrollo en cascada, para esto se han designado diversos ciclos de desarrollo en los cuales se mostraran implementaciones de módulos referentes a las diversas interfaces, controladores y modelos respectivos a cada historia de usuario y enfocados a resolver los múltiples requisitos funcionales del proyecto. Sin embargo se incluiría de igual manera, minutas de reuniones, las cuales se realizan al final de cada ciclo para presentar los resultados de desarrollo y recabar posibles cambios sean ligeros o complejos, evaluando el posible impacto de los mismos, observaciones y pruebas de funcionamiento.

En el presento proyecto se anexarán los documentos SDS, Software Design Specification, especificación de diseño de software, SRS, Software Requirements Specification, especificación de requerimientos de software, Informe postmortem e informe de pruebas, dado que el ciclo final incluye una integración de la documentación de ciclos previos para hacer un compilado completo se optó por tomar esta versión de los entregables y presentarlos en dicho estado.

Dado el tipo de metodología la cantidad de reuniones será mínima dentro de un margen conservador debido al tipo de sistema a desarrollar y el acuerdo con la parte beneficiaria.

3.1 Ciclo 1 y Ciclo 2 - Laravel

El presente ciclo marca el inicio del desarrollo del proyecto, para esta sección se ha designado el comienzo del programa en Laravel, se ha establecido avanzar 5 módulos de funcionalidad, así como sus respectivas pruebas de funcionalidad y aceptación.

Todo avance constará de una carta de conformidad con el avance y funcionalidades.

3.1.1 Especificación de Requerimientos de Software

3.1.1.1 Introducción

En la fase de requerimientos se realiza la recopilación de las necesidades del cliente y el usuario. Que es lo que desea que el software realice. Hay que tener muy en cuenta que existen requerimientos funcionales y no funcionales.

Los requerimientos no funcionales hacen relación a las características del sistema que aplican de manera general como un todo, más que a rasgos particulares del

mismo. Estos requerimientos son adicionales a los requerimientos funcionales que debe cumplir el sistema, y corresponden a aspectos tales como la disponibilidad, mantenibilidad, flexibilidad, seguridad, facilidad de uso, etc.

3.1.1.1.1 Propósito

El presente documento se realiza con la finalidad de presentar la recopilación de criterios, funcionalidades y requerimientos necesarios para el correcto desarrollo del producto. De este modo se tendrá una referencia presentada y sujeta a revisión por parte del beneficiario para verificar si se ha logrado extraer toda idea por parte de este y tomada en cuenta al momento del desarrollo. De ser un documento completo el cliente lo deberá firmar dando de esta manera su aprobación para empezar el desarrollo del mismo.

3.1.1.1.2 Alcance

Se identificará al producto con el nombre de: Sistema de Administración, sin embargo, para referencias futuras y comodidad de tiempo y espacio se referirá al presente con las siglas SA en base a las iniciales del nombre.
El producto se desarrolla como un sistema de apoyo para la gestión de un consultorio médico, dividido en módulos con pantallas para cada funcionalidad de casos de uso, en las cuales el usuario podrá llenar campos de información que generaran entre algunas funciones, administración de inventario, control de usuarios, historias clínicas, entre otras actividades.
Se presenta un programa capaz de interactuar entre interfaces de manera que garantice adaptabilidad y aumentabilidad del mismo.
Al ser un programa de gestión como se mencionó previamente evitamos la necesidad de instalación de programas secundarios para el control o implementación de funciones faltantes en el sistema, evitando problemas de conformidad y compatibilidad.
El programa busca ser lo más sencillo posible para la persona que deberá utilizarlo, siendo este enfoque una de nuestras prioridades.

3.1.1.1.3 Personal involucrado

Nombre	Alexei Ramos
Rol	Desarrollador
Categoría profesional	Egresado
Responsabilidades	Supervisa el cumplimiento de la planificación establecida, desarrollo del producto, desarrollo de pruebas, presentación de informes, avances y reuniones.
Información de contacto	e-mail: alexei.ramos.rivera@hotmail.com Teléfono: 0999817839

3.1.1.1.4 Definiciones, acrónimos y abreviaturas

- Configuration Change Request: CCR
- Configuration Status Report: CSR
- Inspection Report: INS
- Issue Tracking Log: ITL
- Defect Recording Log: LOGD
- Time Recording Form: LOGT
- Test Log: LOGTEST
- Process Improvement Proposal: PIP
- Schedule Planning Template: SCHEDULE
- Strategy Recording Form: STRAT
- Defects Injected Summary Form: SUMDI
- Defects Removed Summary Form: SUMDR
- Program Plan Summary: SUMP
- Quality Plan: SUMQ
- Size Summary: SUMS
- Time Summary Form: SUMT
- Task Summary Form: SUMTASK
- Task Planning Template: TASK
- Weekly Status Report: WEEK
- Sistema de Administración: SA
- Institute of Electrical and Electronics Engineers: IEEE
- Standard: STD/std
- DBMS: Data Base Management System.
- PowerDesigner: Herramienta de modelado empresarial y colaborativa desarrollada por Sybase.
- Visual Paradigm Online Diagrams: Herramienta para modelado en línea.
- PostgreSQL: Base de datos para el desarrollo.
- MySQL: Base de datos para el desarrollo.
- Windows: Sistema operativo privativo desarrollado por Microsoft.
- Javascript: Es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.
- React: Biblioteca Javascript de código abierto diseñada para crear interfaces de usuario con el objetivo de facilitar el desarrollo de aplicaciones en una sola página. Es mantenido por Facebook y la comunidad de software libre, han participado en el proyecto más de mil desarrolladores diferentes.

- IOS: Sistema operativo privativo para dispositivos móviles desarrollado por Apple.
- Android: Sistema operativo libre para dispositivos móviles desarrollado por Google.
- SDK: Software Development Kit.
- INTEL: Procesador con arquitectura x86/x64 desarrollada por Intel Corp.
- AMD: Procesador con arquitectura x86/x64 desarrollada por AMD Inc.
- CU: Casos de uso.
- CP: Casos de prueba.
- IDE: Integrated Development Environment.
- Laravel: Es un framework de código abierto para desarrollar aplicaciones y servicios web con PHP 5 y PHP 7. Su filosofía es desarrollar código PHP de forma elegante y simple, evitando el "código espagueti". Fue creado en 2011 y tiene una gran influencia de frameworks como Ruby on Rails, Sinatra y ASP.NET MVC.
- Framework: Entorno de trabajo o marco de trabajo es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

3.1.1.1.5 Referencias

Referencia	Titulo	Ruta	Fecha	Autor
1	IEEE std	IEEE Std 830-1998.	16-07-2019	Alexei Ramos
2	Definiciones, abreviaturas	Definiciones, acrónimos y abreviaturas	16-07-2019	Alexei Ramos

3.1.1.1.6 Resumen

El presente documento tendrá los diversos diagramas generales de funcionalidad, mismos que están sujetos a cambios, casos de uso y prueba, la recopilación de objetivos y especificará a detalle el servicio que se brindará al cliente, así como la estructura de desarrollo del producto. Se presentan dichos elementos con la esperanza de la aprobación del cliente para empezar el desarrollo y en caso de no conseguirla se recabarán el resto de los requerimientos e inquietudes de este para presentar un producto de acuerdo a un alto estándar de calidad.

Se estructura el producto en base al documento de requerimientos presentado por el instructor y se basa en el estándar IEEE Std 830-1998.

3.1.1.2 Descripción general

3.1.1.2.1 Perspectiva del producto

El presente sistema que se implementará para la creación de un Sistema de Administración se lo realizará de manera independiente, debido a que con el software que se desarrollará se realizará todas las funciones a optimizar dentro del mismo. El programa podría más adelante proveerse de sistemas más pequeños, a fin de la evolución de este.

3.1.1.2.2 Funcionalidad del producto, casos de uso general

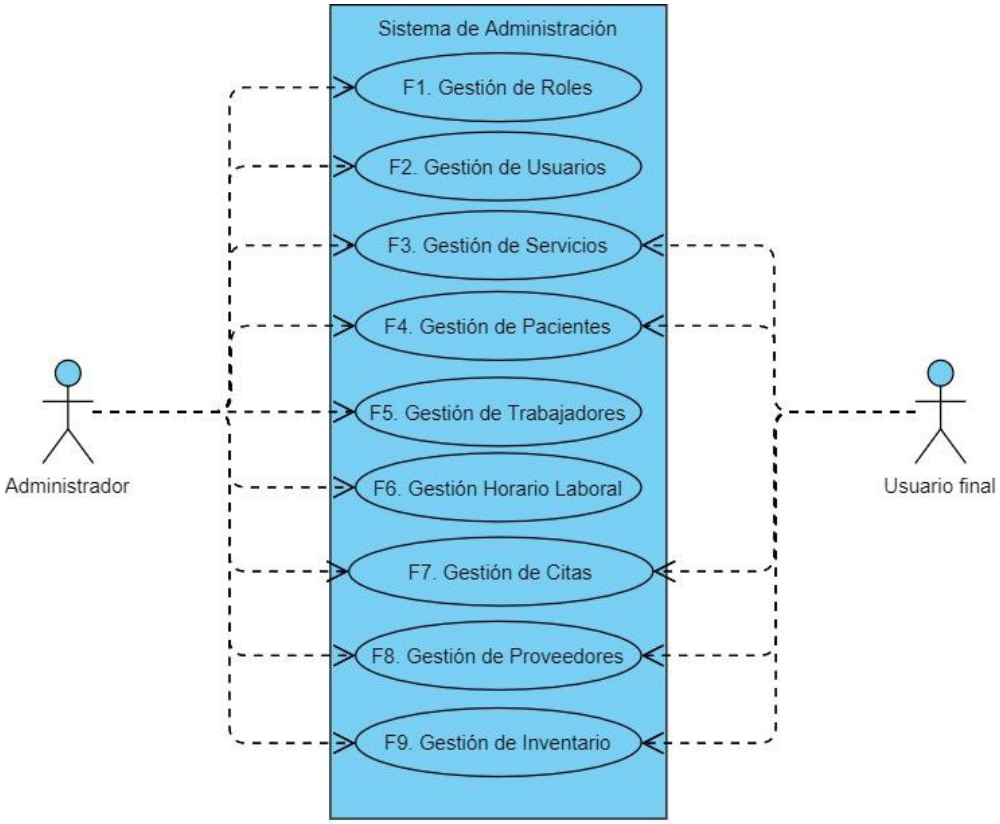


Ilustración 3: Funcionalidades del sistema

Ramos, A. (diciembre 2019).

Las funcionalidades que presentará el proyecto corresponden a diferentes gestiones dentro del cotizador, entre estos tenemos:

- F1: Gestión de Roles.
    - Controlar los roles que el sistema presentará, administrador y usuario simple de forma inicial. Se podrá ingresar, modificar, eliminar los roles presentes en este módulo. Además, se podrá

realizar consultas de tipo informativo. El presente módulo solo será visible para administradores.

- F2: Gestión de Usuarios.
    - Controlar los usuarios que el sistema presentará, estos estarán sujetos a un rol asignado respectivamente a sus funciones. Se podrá ingresar, modificar, eliminar los usuarios presentes en este módulo. Además, se podrá realizar consultas de tipo informativo. El presente módulo solo será visible para administradores.
- F3: Gestión de Servicios.
    - Controlar la creación de servicios que ofrece el consultorio médico, se podrá ingresar, modificar, eliminar y consultar la información acerca de los diferentes servicios como administrador y usuario simple.
- F4: Gestión de Pacientes.
    - Controlar el ingreso de pacientes que son atendidos en el consultorio médico, se podrá ingresar, modificar, eliminar y consultar la información acerca de cada paciente como administrador y usuario simple.
- F5: Gestión de Empleados o Trabajadores.
    - Controlar las personas que trabajan en el consultorio médico, estos estarán sujetos a un servicio asignado respectivamente a sus funciones. Se podrá ingresar, modificar, eliminar los usuarios presentes en este módulo. Además, se podrá realizar consultas de tipo informativo. El presente módulo solo será visible para administradores.
- F6: Gestión de Horas Laborales.
    - Controlar los horarios laborales que el sistema presentará, estos estarán sujetos a un empleado asignado respectivamente a sus funciones y contrato laboral. Dado que todos los empleados están sujetos a un horario de tiempo completo se contemplará un intervalo de 9 horas de jornada laboral, en estas ya se incluye una hora de almuerzo. Se podrá ingresar y modificar los horarios laborales. El presente módulo solo será visible para administradores.
- F7: Gestión de Citas.
    - Controlar el ingreso de citas de diversos servicios realizadas en el consultorio médico, se podrá ingresar, modificar, eliminar y consultar la información acerca de cada cita como administrador y usuario simple.

- F8: Gestión de Proveedores.
    - Controlar el ingreso de proveedores que distribuyen productos al consultorio médico, se podrá ingresar, modificar, eliminar y consultar la información acerca de cada proveedor como administrador y usuario simple.
- F9: Gestión de Inventario.
    - Controlar el inventario de objetos o elementos correspondientes al consultorio médico, se podrá ingresar, modificar, eliminar y consultar la información acerca de cada elemento u objeto como administrador y usuario simple.

3.1.1.2.3 Características de los usuarios

Tipo de usuario	Cliente: Administrador
Formación	Superior
Habilidades	Médicas.
Actividades	Control de Usuarios, Roles, Empleados, Pacientes, enfocado a la atención médica del consultorio.

Tipo de usuario	Empleado: Usuario Final
Formación	Superior
Habilidades	Capaz de utilizar un dispositivo móvil u ordenador.
Actividades	Gestión de ingreso, modificación, consulta y eliminación de procesos de gestión de inventario, proveedores, pacientes, citas y servicios.

Tipo de usuario	Empleado: Administrador
Formación	Superior
Habilidades	Manejo de bases de datos.
Actividades	Gestión de entidades, entendimiento de lenguajes de programación, control del software.

3.1.1.2.4 Restricciones

El presente sistema será desarrollado en php, JavaScript y HMTL5, para el manejo de estos, por tanto, es necesario poseer en la máquina a usar el sistema las versiones actualizadas de estas herramientas, así como un entorno que nos permita el desarrollo fluido de aplicaciones, debido a ello se escoge un equipo para ser el servidor del consultorio, al tener 3 usuarios la demanda de uso es fácilmente manejada por una torre o computador de escritorio, para cumplir con los requisitos y herramientas del sistema se usara Laravel, CSS3 y las librerías de iconos FontIsAwesome. Es importante señalar que el software utilizará como base de datos MySQL, así como para la codificación se usará Sublime Text Editor o Atom. Para el diseño de los diagramas se utiliza el software en línea Visual Paradigm en su edición gratuita o pagada, la diferencia radica en la marca de agua al exportar los gráficos, es permitido el uso de un programa diferente de ser debidamente documentado. El sistema está orientado a ser utilizado dispositivos móviles y ordenadores por medio de la web, para esto es necesario un equipo con conexión a internet.

3.1.1.2.5 Suposiciones y dependencias

El sistema funcionará correctamente en el sistema operativo Windows, OS, IOS, Android. Si se utiliza un sistema operativo basado en Linux, de soportar la versión de Java 1.7 como mínimo, no debería de presentar problemas al momento de su ejecución.
Todos los ordenadores y portátiles presentes en el consultorio médico utilizan Windows y los dispositivos móviles, dos ocupan IOS y uno Android

3.1.1.2.6 Evolución previsible del sistema

Se espera que el sistema a futuro se pueda enfocar a un entorno nativo de ordenador, es decir se convierta en un aplicativo de escritorio, con lo cual se restringa los dispositivos capaces de utilizarlo, además de poder implementarse a otros negocios que estén relacionados a la empresa, por cuestiones de las funcionalidades que presenta.

3.1.1.3 Requisitos específicos

Número de requisito	RF 1		
Nombre de requisito	Gestión de Roles.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El usuario debe ser capaz de poder gestionar roles en el sistema, pudiendo hacer un control de los componentes que se ingresen, modifiquen, consulten o eliminen.

Número de requisito	RF 2		
Nombre de requisito	Gestión de Usuarios.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El usuario debe ser capaz de poder gestionar usuarios en el sistema, pudiendo hacer un control de los componentes que se ingresen, modifiquen, consulten o eliminen.

Número de requisito	RF 3		
Nombre de requisito	Gestión de Servicios.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El usuario debe ser capaz de poder gestionar servicios en el sistema, pudiendo hacer un control de los componentes que se ingresen, modifiquen, consulten o eliminen.

Número de requisito	RF 4		
Nombre de requisito	Gestión de Pacientes.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El usuario debe ser capaz de poder manejar pacientes en el sistema, pudiendo hacer un control de los componentes que se ingresen, modifiquen, consulten o eliminen.

Número de requisito	RF 5		
Nombre de requisito	Gestión de Trabajadores.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El usuario debe ser capaz de poder manejar trabajadores en el sistema, pudiendo hacer un control de los componentes que se ingresen, modifiquen, consulten o eliminen.

Número de requisito	RF 6		
Nombre de requisito	Gestión de Horas Laborales.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El usuario debe ser capaz de poder crear y modificar horarios laborales en el sistema.

Número de requisito	RF 7		
Nombre de requisito	Gestión de Citas.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El usuario debe ser capaz de poder manejar citas en el sistema, pudiendo hacer un control de los componentes que se ingresen, modifiquen, consulten o eliminen.

Número de requisito	RF 8		
Nombre de requisito	Gestión de Proveedores.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El usuario debe ser capaz de poder manejar proveedores en el sistema, pudiendo hacer un control de los componentes que se ingresen, modifiquen, consulten o eliminen.

Número de requisito	RF 9		
Nombre de requisito	Gestión de Inventario.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El usuario debe ser capaz de poder manejar objetos de inventario en el sistema, pudiendo hacer un control de los componentes que se ingresen, modifiquen, consulten o eliminen.

Número de requisito	RF 10		
Nombre de requisito	Documentar o presentar los instaladores ocupados.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☐ Alta/Esencial	☒ Media/Deseado	☐ Baja/ Opcional

El producto necesita documentar o incluir instaladores utilizados en el proceso de desarrollo.

Número de requisito	RF 11		
Nombre de requisito	Cumplir con todos los requisitos.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☐ Alta/Esencial	☒ Media/Deseado	☐ Baja/ Opcional

El producto necesita cumplir con todos los requisitos esenciales.

Número de requisito	RF 12		
Nombre de requisito	Liberar el producto dentro del plazo establecido.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Equipo		
Prioridad del requisito	☐ Alta/Esencial	☒ Media/Deseado	☐ Baja/ Opcional

El producto necesita ser liberado dentro del plazo establecido.

Número de requisito	RF 13		
Nombre de requisito	Conectar los módulos que dependan de otros de forma eficiente y sin errores.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Equipo		
Prioridad del requisito	☐ Alta/Esencial	☒ Media/Deseado	☐ Baja/ Opcional

El producto necesita permitir la integración y comunicación entre módulos cuyos campos están vinculados a otros.

Número de requisito	RF 14		
Nombre de requisito	Responder a un proceso administrado.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☐ Alta/Esencial	☒ Media/Deseado	☐ Baja/ Opcional

El producto necesita seguir un proceso que asegure la trazabilidad.

Número de requisito	RF 15		
Nombre de requisito	Liberar un producto de calidad.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Equipo		
Prioridad del requisito	☐ Alta/Esencial	☒ Media/Deseado	☐ Baja/ Opcional

El producto necesita estar libre de defectos en mayor medida.

Número de requisito	RF 16		
Nombre de requisito	Fácil de configurar y mantenible.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☐ Alta/Esencial	☒ Media/Deseado	☐ Baja/ Opcional

El producto necesita ser fácil de configurar y realizar mantenimiento.

Número de requisito	RF 17		
Nombre de requisito	Trascender el cambio de tecnología.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☐ Alta/Esencial	☐ Media/Deseado	☒ Baja/ Opcional

El producto podría ejecutarse en tecnologías/arquitecturas futuras.

Número de requisito	RF 18		
Nombre de requisito	Manejar generación de contratos y reportes por orden.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☐ Alta/Esencial	☐ Media/Deseado	☒ Baja/ Opcional

El producto podría añadir funcionalidades en línea.

Número de requisito	RF 19		
Nombre de requisito	Lenguaje de programación php, JavaScript y html5		
Tipo	☐ Requisito	☒ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El producto necesita utilizar php, JavaScript, html5 y librerías React.

Número de requisito	RF 20		
Nombre de requisito	Base de Datos MySQL		
Tipo	☐ Requisito	☒ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El producto necesita utilizar el DBMS de *MySQL*.

3.1.1.3.1 Requisitos comunes de las interfaces

El programa cuenta con ingreso de datos mediante cuadros de ingreso de texto y botones; y la salida de datos mediante tablas. Se manejará todo el tiempo interfaz gráfica.

3.1.1.3.1.1 Interfaces de usuario

El programa cuenta con una interfaz principal, que contiene un login, con la finalidad de redireccionar y autentificar el usuario que accede a su debida interfaz, mostrando solo los módulos permitidos a su rol.
Tras la debida validación se generan las interfaces de menú principal por el respectivo usuario, estas son el menú principal de administrador y usuario final.
El interfaz administrador tendrá un menú con todos los módulos disponibles.
La interfaz de usuario final tendrá un menú orientado a la gestión del consultorio mas no sus recursos como empleados, usuarios o roles, mismos accesos fueron establecidos en los casos de uso de cada módulo.
Tanto los menús de usuario final y administrador que interactúen con la base tienen una interfaz para escoger con que funcionalidad o modulo desean interactuar.

3.1.1.3.1.2 Interfaces de hardware

Componentes de hardware que soporten arquitecturas INTEL o AMD.
Componentes de hardware móvil que soporten arquitecturas Android o IOS.

3.1.1.3.1.3 Interfaces de software

Interfaces web responsivas, es decir que se adapten a las pantallas de los dispositivos que los ocupen, el principal enfoque sin embargo se verá orientado al ordenador.

3.1.1.3.1.4 Interfaces de comunicación

No aplica, ya que no es un sistema que contemple chat en tiempo real.

3.1.1.3.2 Requisitos funcionales

En base al diagrama general de casos de uso previamente presentado, obtenemos la funcionalidad a detalle esperada de los módulos a desarrollar.

3.1.1.3.2.1 FG: Funciones Múltiples

Diagrama de Casos de Uso, siguiente nivel, para el presente documento se decidió unificar los casos de uso cuya funcionalidad, diseño y estructura debido a su semejanza. Las funciones que cumplen este criterio de agrupamiento son:

- **F1: Roles, F2: Usuarios, F3: Servicios, F4: Pacientes, F8: Proveedor y F9: Inventario.**

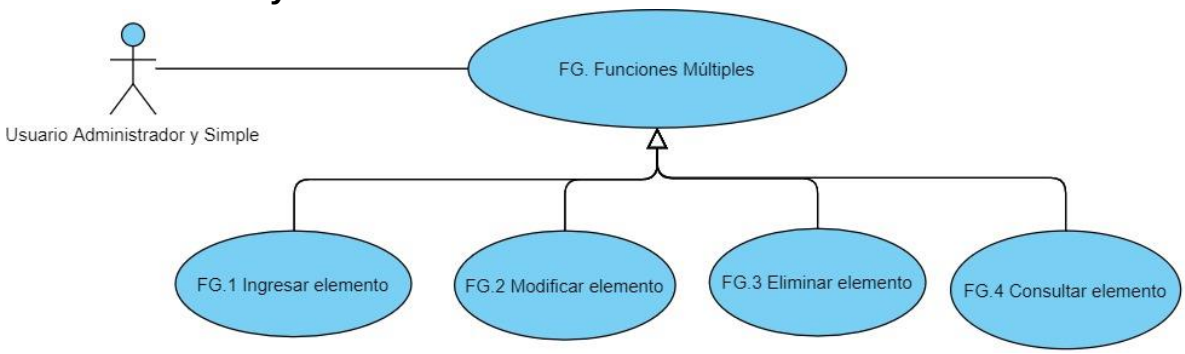


Ilustración 4: Funcionalidades Agrupadas Laravel

Ramos, A. (diciembre 2019).

3.1.1.3.2.1.1 FG.1: Funciones Múltiples: Diagrama a detalle, Ingresar Elemento, este caso de uso permite ingresar elementos nuevos a la tabla.

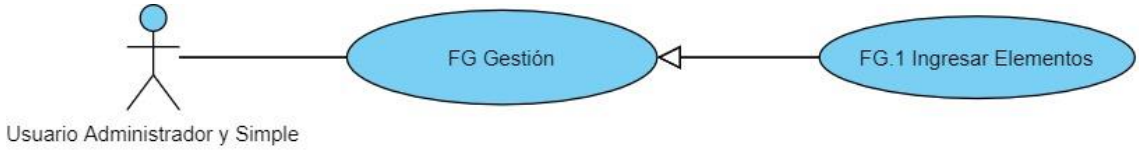


Ilustración 5: FG:F1 Ingresar

Ramos, A. (diciembre 2019).

Flujo principal:

1. El actor selecciona el módulo desde el menú principal.
2. El sistema presenta la ventana del módulo, desplegando los registros existentes.
3. El actor selecciona la opción de ingreso de elementos.
4. El sistema despliega la ventana de creación de elementos.
5. El actor ingresa los campos presentados y requeridos en el formulario desplegado, los campos obligatorios llevarán un “*” junto a su etiqueta, de igual manera los campos opcionales no constarán con esta distinción.
6. El actor presiona en el botón guardar.
7. El sistema verifica los campos obligatorios estén llenos.
8. El sistema almacena los datos.

Flujo alterno:

8. Si el elemento ya existe, el sistema no permite el ingreso.

9. Ir al caso de uso FG.2: Modificar o FG.3: Eliminar.

Excepciones:

Código	Causa	Solución
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.1.1.3.2.1.2 FG.2: Funciones Múltiples: Diagrama a detalle, Modificar Elementos, este caso de uso permite modificar elementos existentes en la tabla.

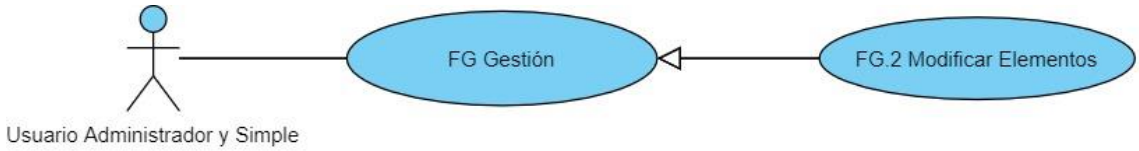


Ilustración 6: FG:F2 Modificar

Ramos, A. (diciembre 2019).

Flujo principal:

1. El actor selecciona el módulo desde el menú principal.
2. El sistema presenta la ventana del módulo, desplegando los registros existentes.
3. El actor selecciona la opción de modificar en la tabla de elementos disponibles.
4. El sistema despliega la ventana de edición de elementos.
5. El actor modifica los elementos que desee cambiar.
6. El actor presiona en el botón guardar.
7. El sistema almacena los datos.

Flujo alterno:

3. Si el producto no existe ir al caso de uso FG.1: Ingresar Elemento.

Excepciones:

Código	Causa	Solución
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.1.1.3.2.1.3 FG.3: Funciones Múltiples: Diagrama a detalle, Eliminar Elementos, este caso de uso permite al eliminar elementos presentes en la tabla.

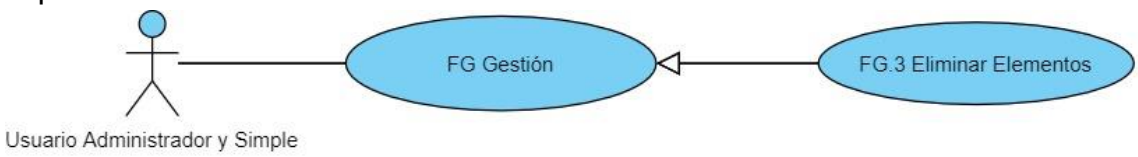


Ilustración 7: FG:F3 Eliminar

Ramos, A. (diciembre 2019).

Flujo principal:

1. El actor selecciona el módulo desde el menú principal.
2. El sistema presenta la ventana del módulo, desplegando los registros existentes.
3. El actor selecciona la opción de eliminar en la tabla de elementos disponibles.
4. El sistema presenta una alerta pop-up para confirmar la eliminación del elemento.
5. El sistema presenta la tabla sin el elemento eliminado.
6. El sistema realiza un soft delete desde la base.

Flujo alterno:

3. Si el producto no existe ir al caso de uso FG.1: Ingresar Elemento.

Excepciones:

Código	Causa	Solución
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.1.1.3.2.1.4 FG.4: Funciones Múltiples: Diagrama a detalle, Consultar Elementos, este caso de uso permite consultar a detalle los elementos de la tabla.

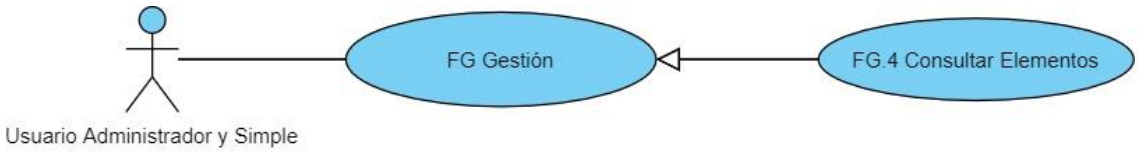


Ilustración 8: FG:F1 Consultar

Ramos, A. (diciembre 2019).

Flujo principal:

1. El actor selecciona el módulo desde el menú principal.
2. El sistema presenta la ventana del módulo, desplegando los registros existentes.
3. El actor selecciona la opción de consultar, mostrar en la tabla de elementos disponibles.
4. El sistema despliega la ventana de consulta de elementos.
5. El sistema presenta la información del elemento consultado.

Flujo alterno:

3. Si el producto no existe ir al caso de uso FG.1: Ingresar Elemento.

Excepciones:

Código	**Causa**	**Solución**
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.1.1.3.2.2 F5: Función Trabajadores

Diagrama de Casos de Uso, siguiente nivel, para la funcionalidad de trabajadores.

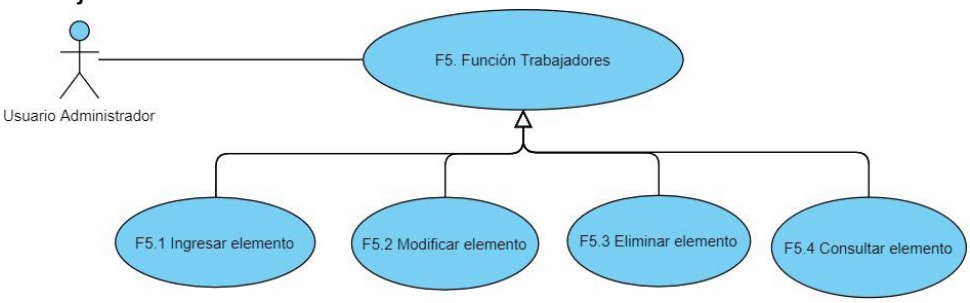


Ilustración 9: F5 Trabajadores

Ramos, A. (diciembre 2019).

3.1.1.3.2.2.1 F5.1: Función Trabajadores: Diagrama a detalle, Ingresar Elemento, este caso de uso permite ingresar elementos nuevos a la tabla de trabajadores.

Ilustración 10: F5:F1 Ingresar

Ramos, A. (diciembre 2019).

Flujo principal:

1. El actor selecciona el módulo Trabajadores desde el menú principal.
2. El sistema presenta la ventana del módulo Trabajadores, desplegando los registros existentes.
3. El actor selecciona la opción de ingreso de trabajadores.
4. El sistema despliega la ventana de creación de trabajadores.
5. El actor ingresa los campos presentados y requeridos en el formulario desplegado, los campos obligatorios llevarán un “*” junto a su etiqueta, de igual manera los campos opcionales no constarán con esta distinción, estos campos son propios de Trabajadores.
6. El actor debe escoger el servicio que deberá supervisar el trabajador como un campo desplegable del formulario.
7. El actor presiona en el botón guardar.
8. El sistema verifica los campos obligatorios estén llenos.
9. El sistema almacena los datos.

Flujo alterno:

9. Si el elemento ya existe, el sistema no permite el ingreso.

10. Ir al caso de uso Modificar o Eliminar.

6. Si no existen servicio, ir al caso de uso F3.1: Crear Servicio.

Excepciones:

Código	Causa	Solución
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.1.1.3.2.2.2 F5.2: Función Trabajadores: Diagrama a detalle, Modificar Elementos, este caso de uso permite modificar elementos existentes en la tabla de trabajadores.

Ilustración 11: F5:F2 Modificar

Ramos, A. (diciembre 2019).

Flujo principal:

1. El actor selecciona el módulo Trabajadores desde el menú principal.
2. El sistema presenta la ventana del módulo Trabajadores, desplegando los registros existentes.
3. El actor selecciona la opción de modificar en la tabla de trabajadores disponibles.
4. El sistema despliega la ventana de edición de trabajador.
5. El actor modifica los elementos que desee cambiar, el único campo que no puede ser modificado es el servicio asignado.
6. El actor presiona en el botón guardar.
7. El sistema almacena los datos.

Flujo alterno:

3. Si el producto no existe ir al caso de uso F5.1: Ingresar Trabajador.

Excepciones:

Código	Causa	Solución
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.1.1.3.2.2.3 F5.3: Función Trabajadores: Diagrama a detalle, Eliminar Elementos, este caso de uso permite al eliminar elementos presentes en la tabla de trabajadores.

Ilustración 12: F5:F3 Eliminar

Ramos, A. (diciembre 2019).

Flujo principal:

1. El actor selecciona el módulo desde el menú principal.
2. El sistema presenta la ventana del módulo, desplegando los registros existentes.
3. El actor selecciona la opción de eliminar en la tabla de trabajadores disponibles.
4. El sistema presenta una alerta pop-up para confirmar la eliminación del trabajador.
5. El sistema presenta la tabla sin el trabajador eliminado.
6. El sistema realiza un soft delete desde la base.

Flujo alterno:

3. Si el producto no existe ir al caso de uso F5.1: Ingresar Trabajador.

Excepciones:

Código	Causa	Solución
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.1.1.3.2.2.4 F5.4: Función Trabajadores: Diagrama a detalle, Consultar Elementos, este caso de uso permite consultar a detalle los elementos de la tabla de trabajadores.

Ilustración 13: F5:F4 Consultar

Ramos, A. (diciembre 2019).

Flujo principal:

1. El actor selecciona el módulo Trabajadores desde el menú principal.
2. El sistema presenta la ventana del módulo Trabajadores, desplegando los registros existentes.
3. El actor selecciona la opción de consultar en la tabla de trabajadores disponibles.
4. El sistema despliega la ventana de consulta de elementos.
5. El sistema presenta la información del trabajador consultado.

Flujo alterno:

3. Si el producto no existe ir al caso de uso F5.1: Ingresar Trabajador.

Excepciones:

Código	Causa	Solución
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.1.1.3.2.3 F6: Función Horarios Laborales

Diagrama de Casos de Uso, siguiente nivel, para la funcionalidad de horarios laborales.

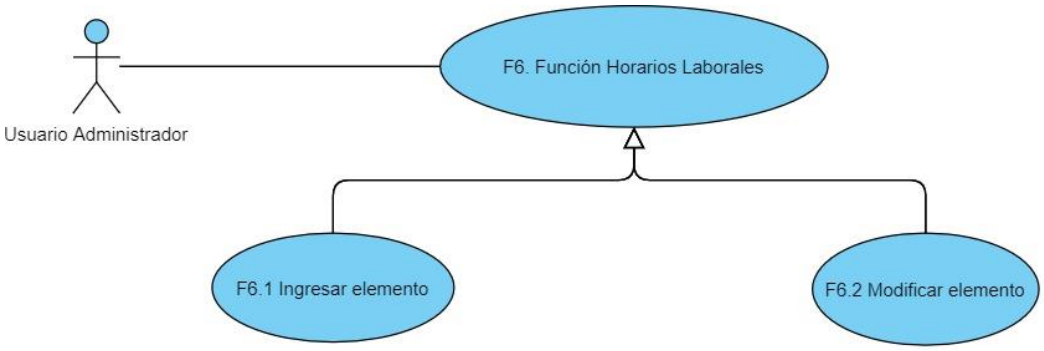


Ilustración 14: F6 Horarios Laborales

Ramos, A. (diciembre 2019).

3.1.1.3.2.3.1 F6.1: Función Horarios Laborales: Diagrama a detalle, Ingresar Horario Laboral, este caso de uso permite ingresar horarios laborales vinculados a un empleado.

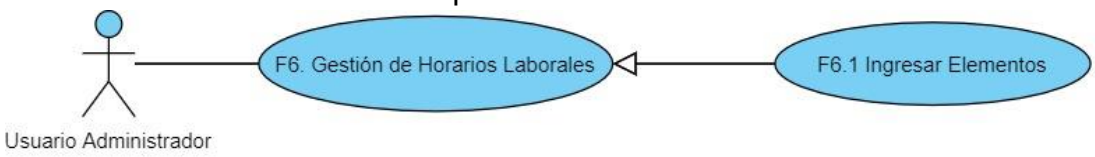


Ilustración 15: F6:F1 Ingresar

Ramos, A. (diciembre 2019).

Flujo principal:

1. El actor selecciona el módulo Horarios Laborales desde el menú principal.
2. El sistema presenta la ventana del módulo Horarios Laborales, desplegando los registros existentes en una vista calendario.
3. El actor selecciona la opción de ingreso de horarios laborales.
4. El sistema despliega la ventana de creación de horarios laborales.
5. El actor ingresa los campos presentados y requeridos en el formulario desplegado, los campos obligatorios llevarán un “*” junto a su etiqueta, de igual manera los campos opcionales no constarán con esta distinción, estos campos son propios de Horarios Laborales.
6. El actor debe escoger el empleado vinculado a ese horario laboral como un campo desplegable del formulario.

7. El actor presiona en el botón guardar.
8. El sistema verifica los campos obligatorios estén llenos.
9. El sistema almacena los datos.

Flujo alterno:

6. Si no existen empleados, ir al caso de uso F5.1: Crear Empleado.

Excepciones:

Código	Causa	Solución
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.1.1.3.2.3.2 F6.2: Función Horarios Laborales: Diagrama a detalle, Modificar Elemento, este caso de uso permite modificar horarios laborales vinculados a un empleado, siendo posible cambiar los empleados asignados a un espacio.

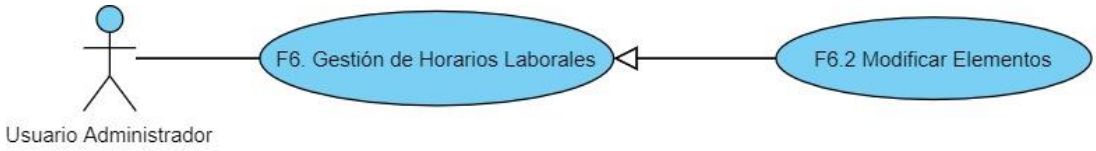


Ilustración 16: F6:F2 Modificar

Ramos, A. (diciembre 2019).

Flujo principal:

1. El actor selecciona el módulo Horarios Laborales desde el menú principal.
2. El sistema presenta la ventana del módulo Horarios Laborales, desplegando los registros existentes en una vista calendario.
3. El actor selecciona un horario laboral desplegado en el calendario.
4. El sistema despliega la ventana de edición de horarios laborales.
5. El actor modifica los campos presentados y requeridos en el formulario desplegado, los campos obligatorios llevarán un "*" junto a su etiqueta, de igual manera los campos opcionales no constarán con esta distinción, estos campos son propios de Horarios Laborales.
6. El actor puede escoger vincular otro empleado al horario laboral como un campo desplegable del formulario.

7. El actor presiona en el botón guardar.
8. El sistema verifica los campos obligatorios estén llenos.
9. El sistema almacena los datos.

Flujo alterno:

3. Si no existen empleados, ir al caso de uso F5.1: Crear Empleado.

Excepciones:

Código	Causa	Solución
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.1.1.3.2.4 F7: Función Citas

Diagrama de Casos de Uso, siguiente nivel, para la funcionalidad de Citas.

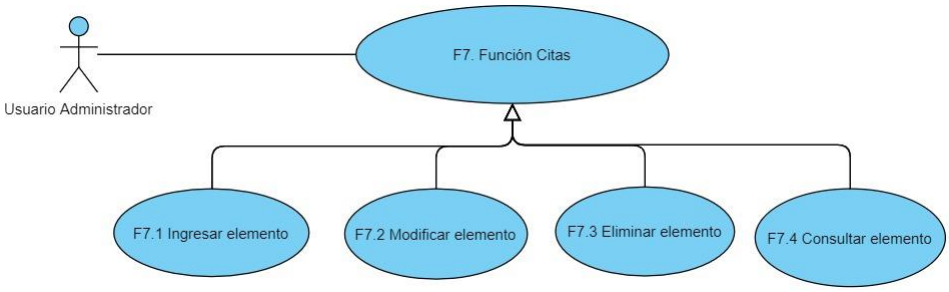


Ilustración 17: F7 Citas

Ramos, A. (diciembre 2019).

3.1.1.3.2.4.1 F7.1: Función Citas: Diagrama a detalle, Ingresar Citas, este caso de uso permite ingresar citas nuevas a la tabla y calendario de citas.

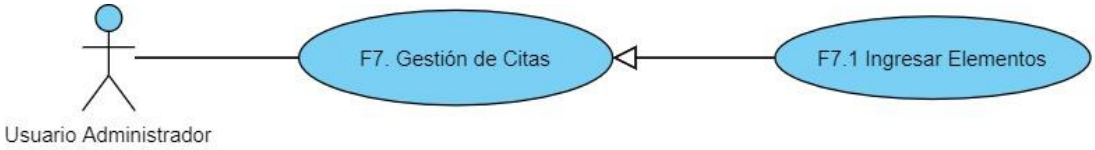


Ilustración 18: F7:F1 Ingresar

Ramos, A. (diciembre 2019).

Flujo principal:

1. El actor selecciona el módulo Citas desde el menú principal.
2. El sistema presenta la ventana del módulo Citas, desplegando los registros existentes en una vista calendario y una tabla abajo del calendario.
3. El actor selecciona la opción de ingreso de citas.
4. El sistema despliega la ventana de creación de citas.
5. El actor ingresa los campos presentados y requeridos en el formulario desplegado, los campos obligatorios llevarán un "*" junto a su etiqueta, de igual manera los campos opcionales no constarán con esta distinción, estos campos son propios de citas.
6. El actor debe escoger el empleado y el servicio vinculado al horario que desea la cita como campos desplegables del formulario.
7. El actor presiona en el botón guardar.
8. El sistema verifica los campos obligatorios estén llenos.
9. El sistema almacena los datos.

Flujo alterno:

9. Ir al caso de uso Modificar o Eliminar.

6. Si no existen servicios, ir al caso de uso F3.1: Crear Servicio

6. Si no existen empleados, ir al caso de uso F5.1: Crear Empleado

Excepciones:

Código	Causa	Solución
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.1.1.3.2.4.2 F7.2: Función Citas: Diagrama a detalle, Editar Citas, este caso de uso permite editar citas nuevas a la tabla y calendario de citas.

Ilustración 19: F7:F2 Modificar

Ramos, A. (diciembre 2019).

Flujo principal:

1. El actor selecciona el módulo Citas desde el menú principal.
2. El sistema presenta la ventana del módulo Citas, desplegando los registros existentes en una vista calendario y una tabla.
3. El actor selecciona una cita desplegada en el calendario o tabla.
4. El sistema despliega la ventana de edición de citas.
5. El actor modifica los campos presentados y requeridos en el formulario desplegado, los campos obligatorios llevarán un "*" junto a su etiqueta, de igual manera los campos opcionales no constarán con esta distinción, estos campos son propios de Citas.
6. El actor puede escoger vincular otro empleado a la cita como un campo desplegable del formulario.
7. El actor presiona en el botón guardar.
8. El sistema verifica los campos obligatorios estén llenos.
9. El sistema almacena los datos.

Flujo alterno:

3. Si el producto no existe ir al caso de uso F5.1: Ingresar Trabajador.

6. Si no existen empleados, ir al caso de uso F5.1: Crear Empleado

Excepciones:

Código	**Causa**	**Solución**
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.1.1.3.4.3 F7.3: Función Citas: Diagrama a detalle, Eliminar Citas, este caso de uso permite eliminar citas nuevas de la tabla Citas.

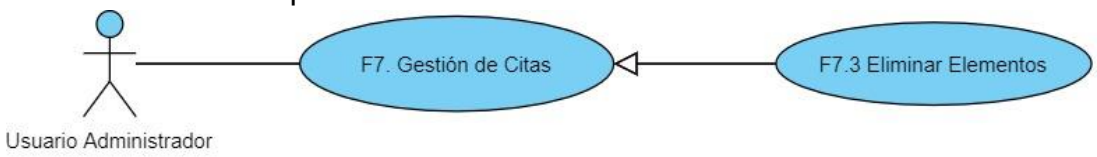


Ilustración 20: F7:F3 Eliminar

Ramos, A. (diciembre 2019).

Flujo principal:

1. El actor selecciona el módulo Citas desde el menú principal.
2. El sistema presenta la ventana del módulo Citas, desplegando los registros existentes.
3. El actor selecciona la opción de eliminar en la tabla de citas disponibles.
4. El sistema presenta una alerta pop-up para confirmar la eliminación de la cita.
5. El sistema presenta la tabla sin la cita eliminada.
6. El sistema realiza un soft delete desde la base.

Flujo alterno:

3. Si la cita no existe ir al caso de uso F7.1: Ingresar Citas.

Excepciones:

Código	Causa	Solución
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.1.1.3.4.4 F7.4: Función Citas: Diagrama a detalle, Consultar Citas, este caso de uso permite consultar a detalle los elementos de la tabla de citas.

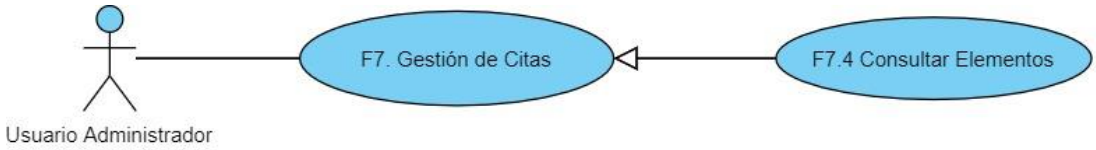


Ilustración 21: F7:F4 Consultar

Ramos, A. (diciembre 2019).

Flujo principal:

1. El actor selecciona el módulo Citas desde el menú principal.
2. El sistema presenta la ventana del módulo Citas, desplegando los registros existentes.
3. El actor selecciona la opción de consultar en la tabla de citas disponibles.
4. El sistema despliega la ventana de consulta de elementos.

5. El sistema presenta la información de la cita consultada.

Flujo alterno:

3. Si el producto no existe ir al caso de uso F7.1: Ingresar Trabajador.

Excepciones:

Código	Causa	Solución
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.1.1.3.3 Requisitos no funcionales

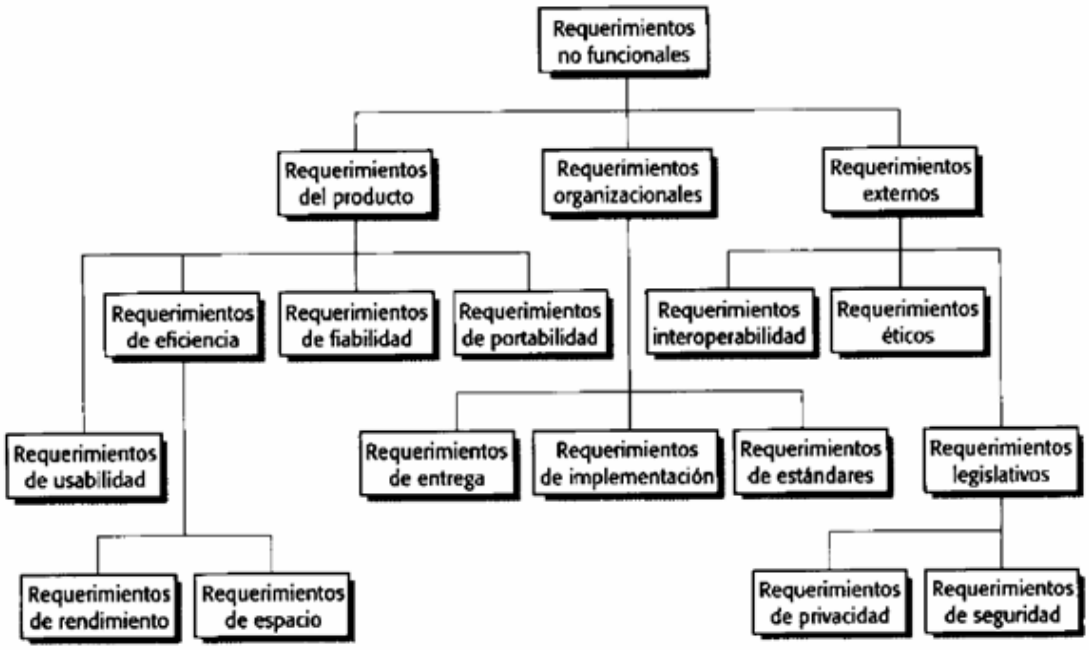


Ilustración 22: Requisitos de Software no Funcionales

Camacho, L. (diciembre 2016).

a. Del Producto

i. El programa debe ser utilizable de manera adecuada por el usuario sin presentar errores o bugs, con un tiempo de respuesta óptimo.
ii. El programa debe ser amigable para el usuario e intuitivo.
iii. El programa debe ejecutar sus consultas en un rango de tiempo de 3 a 7 segundos, este tiempo fluctuara en función de la

conexión de internet del dispositivo que realiza la consulta, su capacidad y la optimización de consultas.

iv. Se debe establecer el peso total al incorporar cada funcionalidad, especificando las mismas y generando un respaldo.
v. El programa debe poder ejecutarse sin fallas o caídas.
vi. El programa debe estar orientado a la web, permitiendo su uso desde varios dispositivos que soporten una conexión a internet, la base de datos y el repositorio del programa estará enfocada a un ordenador que simulara un servidor, cuyo sistema operativo es Windows 10.

b. Organizacionales

i. Se entrega el programa totalmente funcional.
ii. El primer ciclo implementara las funcionalidades críticas establecidas en la reunión que se llevó a cabo el día 15 de julio de 2019.
iii. El programa se orienta totalmente a la web.
iv. Se definen los estándares a ocupar, estos serán los mismos para los ciclos posteriores siempre y cuando no se requiera algún cambio, mismo que solo será válido al pasar por mesa de cambio.
v. Se debe precautelar el correcto uso de recursos.

c. Externos

i. La parte aplicativa debe poder relacionarse sin conflictos con la base de datos.
ii. Se informará al supervisor en caso de no poder evitar incumplimientos en fechas por conflictos de programas externos, como los entornos de desarrollo, sin embargo, el mitigar o hacer nulo este tipo de conflictos es prioridad durante el proceso de planificación.
iii. Se usarán solo los programas presentados en el respectivo informe y aprobados por el supervisor.

iv. Se mantendrán los requisitos de patentes y derechos de autor requeridos para legalizar el producto tras su entrega y finalización, en este caso se liga dicha responsabilidad al representante jurídico del beneficiario.
    1. Como responsabilidad del desarrollador se compromete a mantener la privacidad del código, la información proporcionada por el beneficiario y cualquier otro tipo de

información relacionada al desarrollo del proyecto a terceros.
2. El programa debe poder reaccionar a intentos de romper su seguridad acorde al caso.
3. Se debe generar informes de vulnerabilidad en la fase de pruebas para precautelar la integridad del proyecto y permitir un lanzamiento seguro del mismo.

v. El personal que utilizara el producto será el responsable de las acciones que se realicen con el mismo, fuera del marco permitido por la empresa y se tomaran sanciones por parte del organismo pertinente dentro de la misma.

3.1.1.3.3.1 Requisitos de rendimiento

El programa soportará los 3 usuarios miembros del equipo de trabajo del consultorio médico a la vez, el proceso se realizará por medio de la web, por lo que el rendimiento sobre la base de datos será óptimo.

3.1.1.3.3.2 Seguridad

El programa manejará encapsulación lo que defenderá al programa contra ataques maliciosos a nivel de código. Además, el programa defenderá información sensible. Al incorporar la interfaz web se protegerá al sistema con un límite de solicitudes de ping a la base determinados.

3.1.1.3.3.3 Fiabilidad

El programa se espera que sea fiable en un uso típico, inclusive, se espera que el sistema no falle debido al estrés u otros factores.

3.1.1.3.3.4 Disponibilidad

El programa tendrá siempre un 100% de disponibilidad mientras el equipo esté encendido y posea una conexión a internet.

3.1.1.3.3.5 Mantenibilidad

El proyecto será desarrollado acorde a diversos módulos y sus respectivos controladores, se definirán las rutas del mismo así como sus interfaces, al estar documentado a detalle, permite su mantenimiento adecuado.

3.1.1.3.3.6 Portabilidad

El programa está desarrollado en Laravel lo que asegura su portabilidad, pues al ser un producto orientado a la web, permite su uso desde múltiples dispositivos cuyo requisito es tener una conexión a internet.

3.1.1.3.4 Otros requisitos

- Documentar los códigos utilizados para etiquetas en la base de datos.
- Control de usuarios, tablas, base datos más restringidos.
- Sugerencias, advertencias, confirmaciones para las interfaces del cotizador por campos o botones clave.

3.1.1.3.5 Apéndices

Plan de Pruebas del Sistema

Manejo de Funcionalidades Básicas: Este caso de prueba busca efectuar pruebas en el manejo de todo el sistema, dada la similitud de módulos se han agrupado las primeras cuatro funcionalidades como FG: Funcionalidades múltiples, los módulos que se agruparon son: **F1: Roles, F2: Usuarios, F3: Servicios, F4: Pacientes, F8: Proveedor y F9: Inventario.**

De igual manera las pruebas para las funcionalidades: **F5: Empleados, F6: Horarios Laborales y F7: Citas** son similares a las que serán realizadas en módulos anteriores, con la diferenciación que se hará énfasis a la conexión entre servicios y empleados o trabajadores. Verificando la correcta vinculación de un servicio a un trabajador, de un trabajador a una cita y de un servicio a una cita.

Precondiciones:

Este caso de prueba depende de la terminación del ciclo de desarrollo, durante los informes de pruebas se explorará esta tabla a profundidad módulo por módulo.

Entradas	Resultados	Caso de Uso	Exitoso
1. El actor selecciona opción X del menú principal	Despliega menú	FG/F5/F6/F7	X
2. El sistema presenta la ventana de X	Despliega ventana	FG/F5/F6/F7	X

3. El actor ingresa datos del formulario	El sistema verifica datos del formulario	FG/F5/F6/F7	X
4. El actor presiona Ingresar	El sistema envía los datos del formulario	FG/F5/F6/F7	X
5. El actor presiona Ingresar	El sistema almacena los datos	FG/F5/F6/F7	X
6. El actor ingresa el código del elemento	El sistema verifica el código de elemento ingresado	FG/F5/F6/F7	X
7. El sistema presenta la ventana de consulta de X.	Despliega ventana	FG/F5/F6/F7	X
8. El sistema presenta las X existentes en una tabla	Despliega datos en tabla	FG/F5/F6/F7	X
9. El actor ingresa parámetro X de la función escogida	El sistema presenta los datos de la función escogida.	FG/F5/F6/F7	X
10. El actor ingresa el código de X	El sistema presenta los datos de la X	FG/F5/F6/F7	X

3.1.2 Especificación de Diseño de Software

3.1.2.1 Definición.

En la presente fase, se procede con el diseño del producto software. Diseñar es el proceso creativo de cómo se decide construir el producto, en esta fase nos enfrentamos al reto de ir generando los módulos correspondientes a las funcionalidades del sistema.

Es importante partir de los requisitos obtenidos en la especificación de requerimientos de software, pues brinda una guía sólida para la presente fase, es de vital importancia el relacionar de manera adecuada los diversos módulos entre sí delimitando como interactúan individualmente y entre ellos estableciendo una interacción encilla con el usuario.
En esta fase definimos la arquitectura de la aplicación, diagramas UML, tales como: Clases, Actividades, Paquetes y el diagrama de base de datos de Entidad-Relación. Finalmente definimos el plan de pruebas de integración.

3.1.2.2 Arquitectura.

a. Arquitecturas MVC.

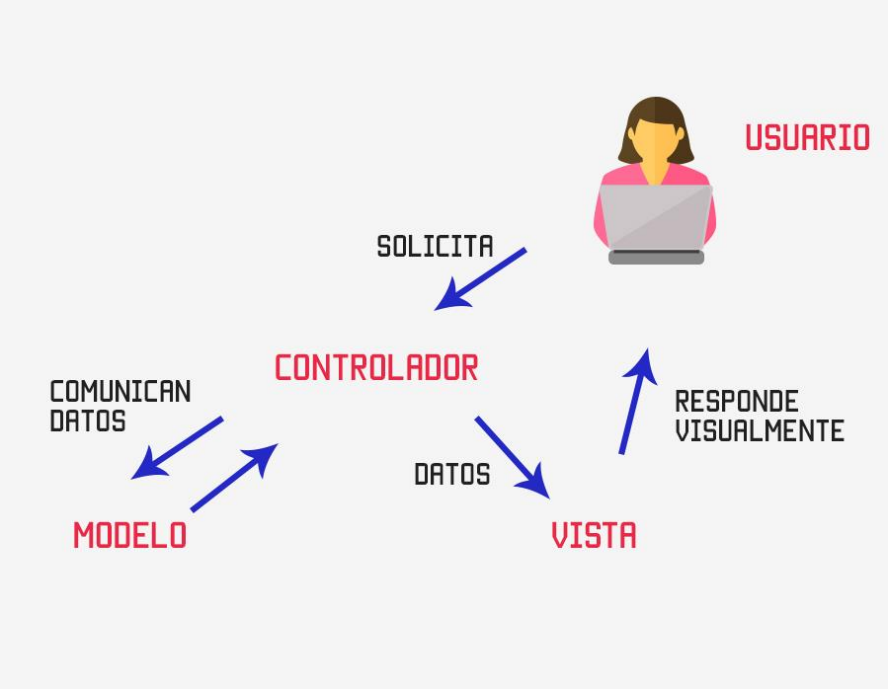


Ilustración 23: Modelo-Vista-Controlador

(Hernandez, 2015).

Como se ha explicado en capítulos anteriores, el presente proyecto parte de una estructura modelo-vista-controlador, esto se conoce como una aplicación 3 capas, la capa de modelo interactúa con la base de datos y con el controlador, generando consultas SQL y enviando el resultado, el controlador luego redirecciona los resultados de las consultas hacia las vistas, presentando los resultados en la capa visual del programa.

Sin embargo para precautelar la seguridad del sistema ante ataques y con la finalidad de cumplir los requisitos de alta protección se ha integrado internamente elementos tales como:

1. Puertas o Gates: una capa de autorización de las operaciones que el controlador envía al modelo, esta herramienta nos permite definir que operaciones puede realizar un usuario, si no se tiene la autorización para realizarlas, no permite la ejecución.
2. HTTP Request: Asemeja la interacción orientada a objetos con las solicitudes HTTP que se estén generando en la aplicación, podemos de esta manera recuperar inputs, cookies y archivos enviados por medio de la solicitud, el proyecto integra este método en el controlador, generando dos archivos por cada controlador, responsables de manejar la información enviada al crear y editar elementos de los diferentes módulos, de esta forma podemos proteger la información manejada por las operaciones post.
3. Middleware: Una herramienta de autentificación de operaciones de inicio de sesión y cambio de contraseñas, se encarga de filtrar las peticiones HTTP, en conjunto con el módulo de roles , permiten filtrar que menú se le mostrara al usuario tras iniciar sesión acorde a los permisos de cada rol, se integra esta herramienta en la operación de inicio de sesión al filtrar y encriptar la contraseña de cada usuario.

Ventajas que ofrece el presente proyecto:

1. Separación entre el cliente y el servidor: el protocolo REST separa totalmente la interfaz de usuario del servidor y el almacenamiento de datos esto permite mejorar la portabilidad de la interfaz a otro tipo de plataformas, aumenta la escalabilidad de los proyectos y permite que los distintos componentes de los desarrollos se puedan evolucionar de forma independiente. Gracias a esto podemos separar el front y back end en diferentes servidores, la escalabilidad y evolución del programa permite generar diferentes respuestas e interfaces mientras la conexión se realice de manera adecuada es sencillo por ejemplo alternar entre bases de datos.

3.1.2.3 Diagrama de Aplicación

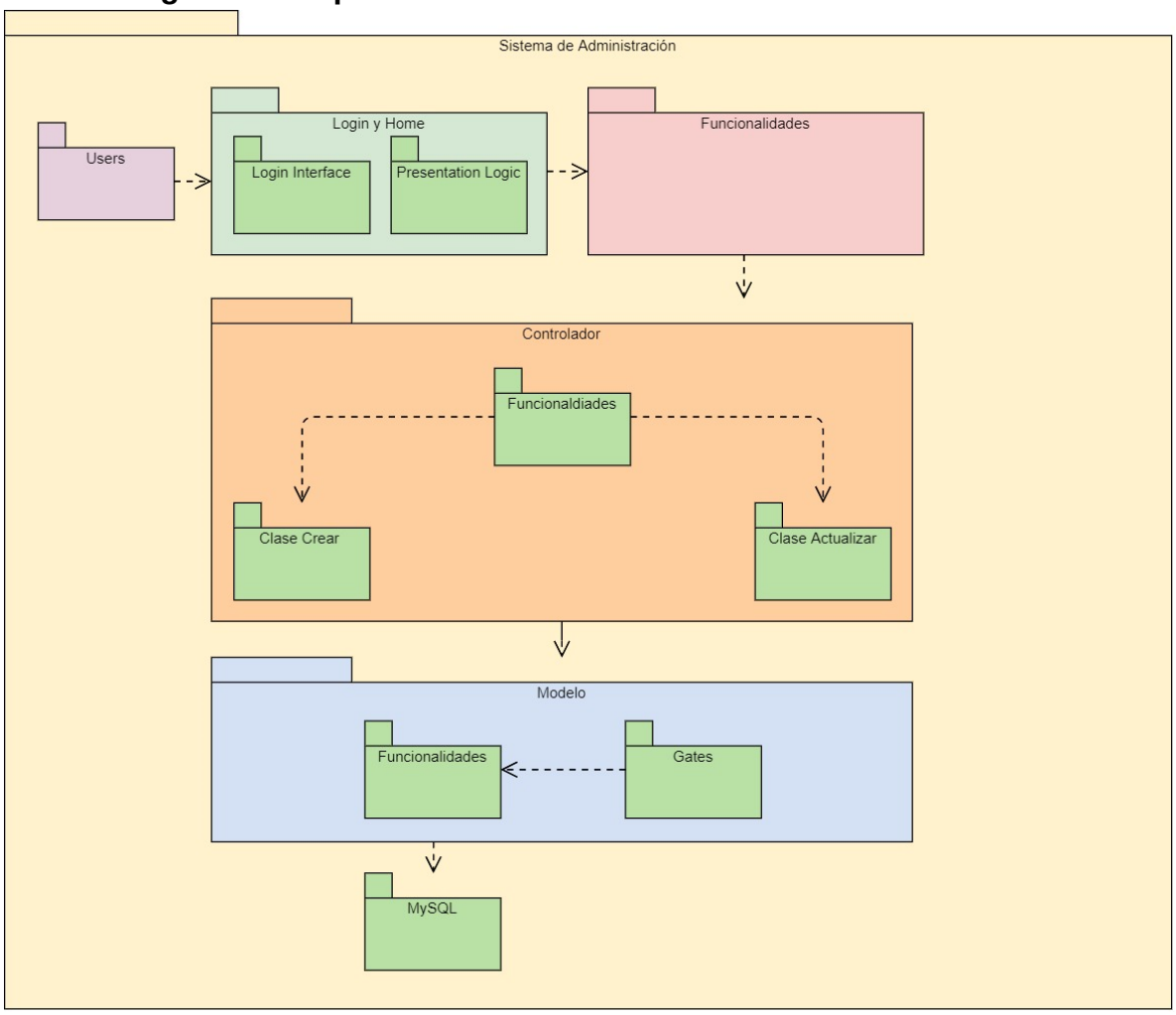


Ilustración 24: Diagrama de Aplicación

Ramos, A. (diciembre 2019).

3.1.2.4 Diagrama de Paquetes Ciclo 1

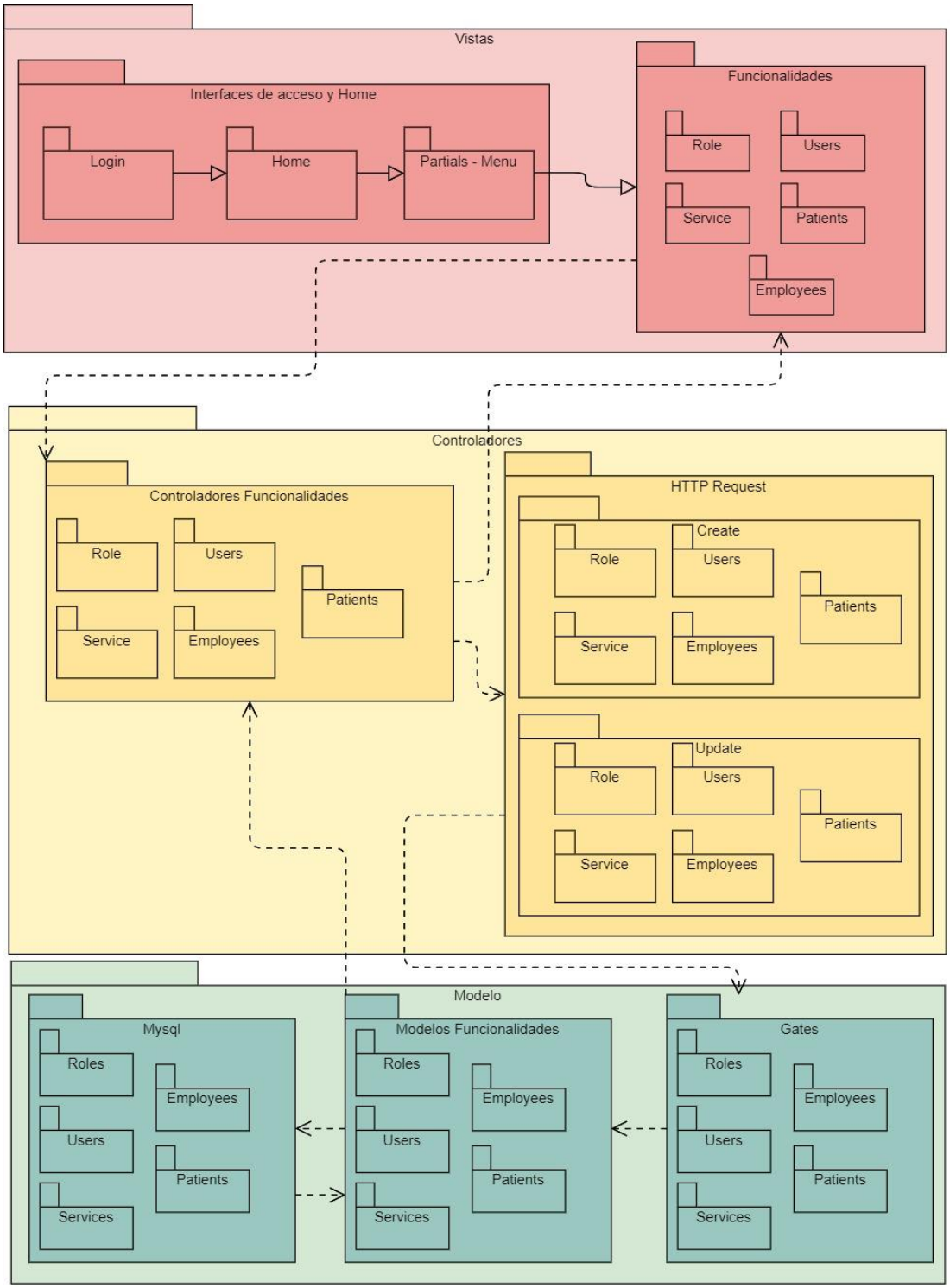


Ilustración 25: Diagrama de Paquetes C1

Ramos, A. (diciembre 2019).

Diagrama de Paquetes Ciclo 2

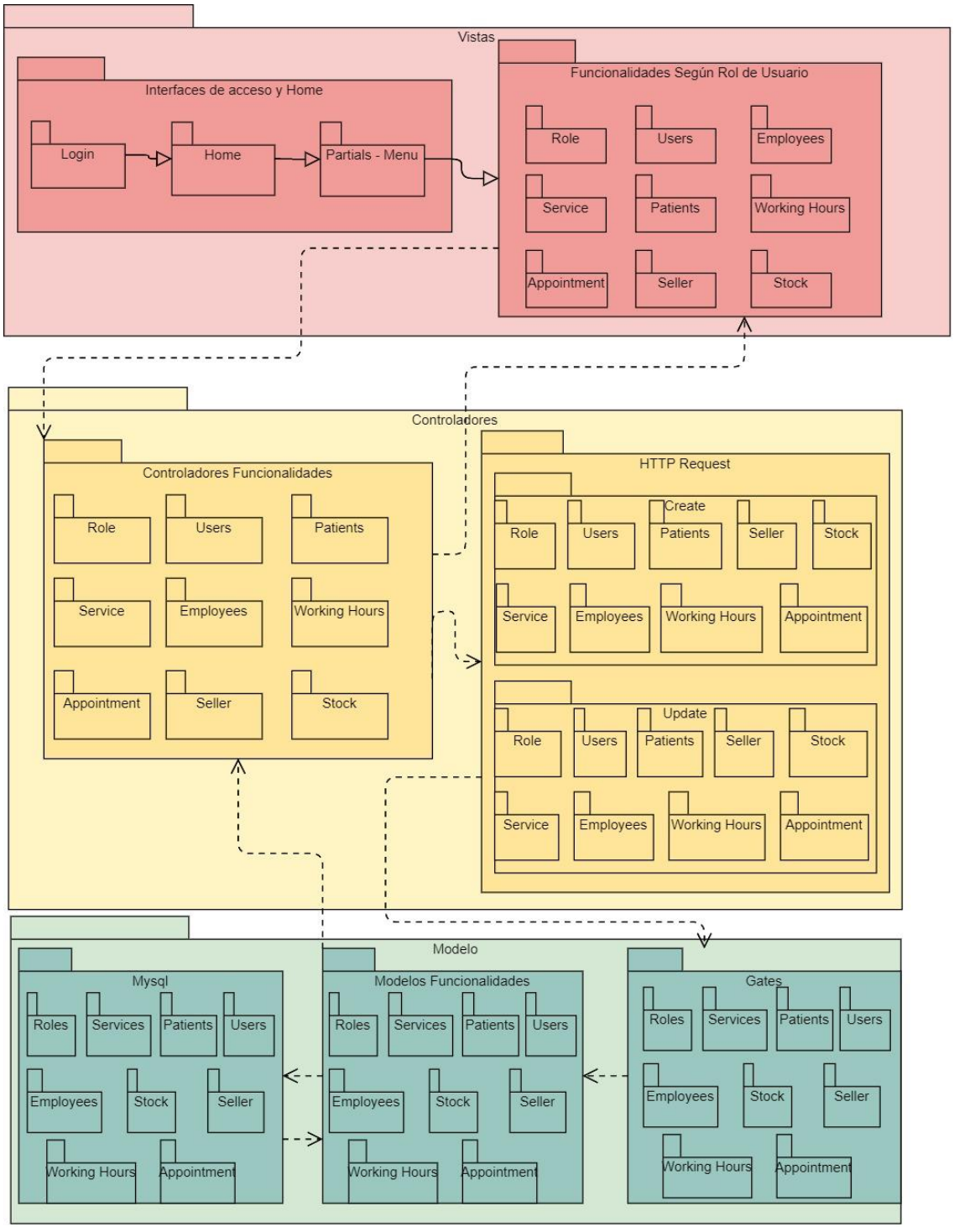


Ilustración 26: Diagrama de Paquetes C2

Ramos, A. (diciembre 2019).

3.1.2.5 Diagrama de Clases.

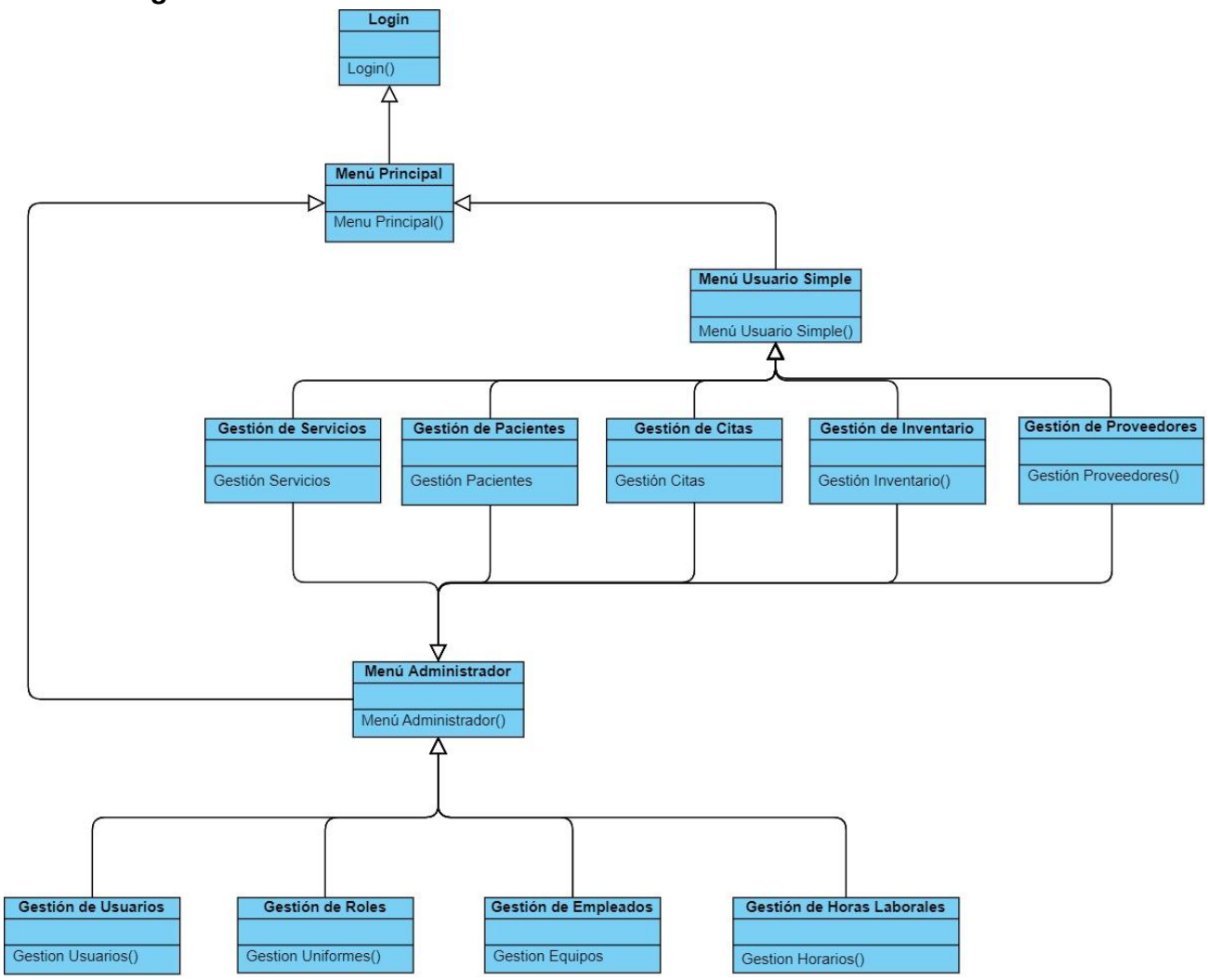


Ilustración 27: Diagrama de Clases

Ramos, A. (diciembre 2019).

3.1.2.6 Diagrama de Secuencia.

Ingreso al sistema.

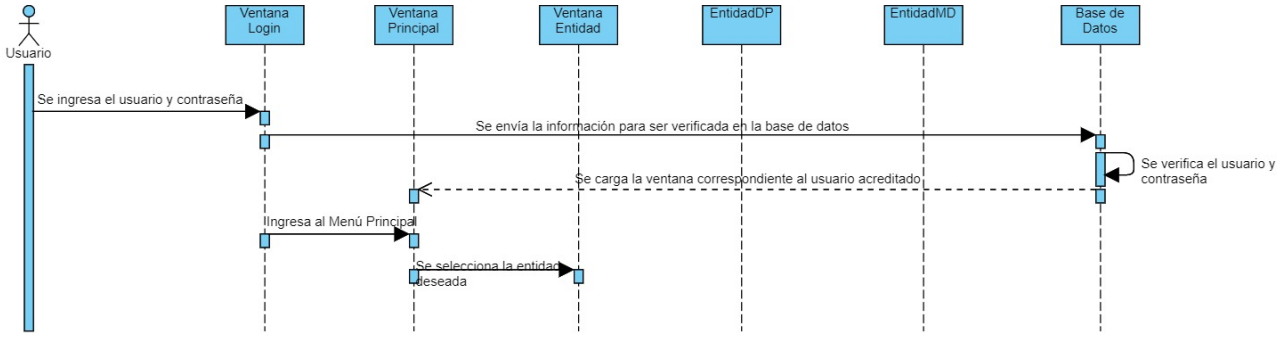


Ilustración 28: Diagrama de Secuencia Ingreso

Ramos, A. (diciembre 2019).

Procesos.

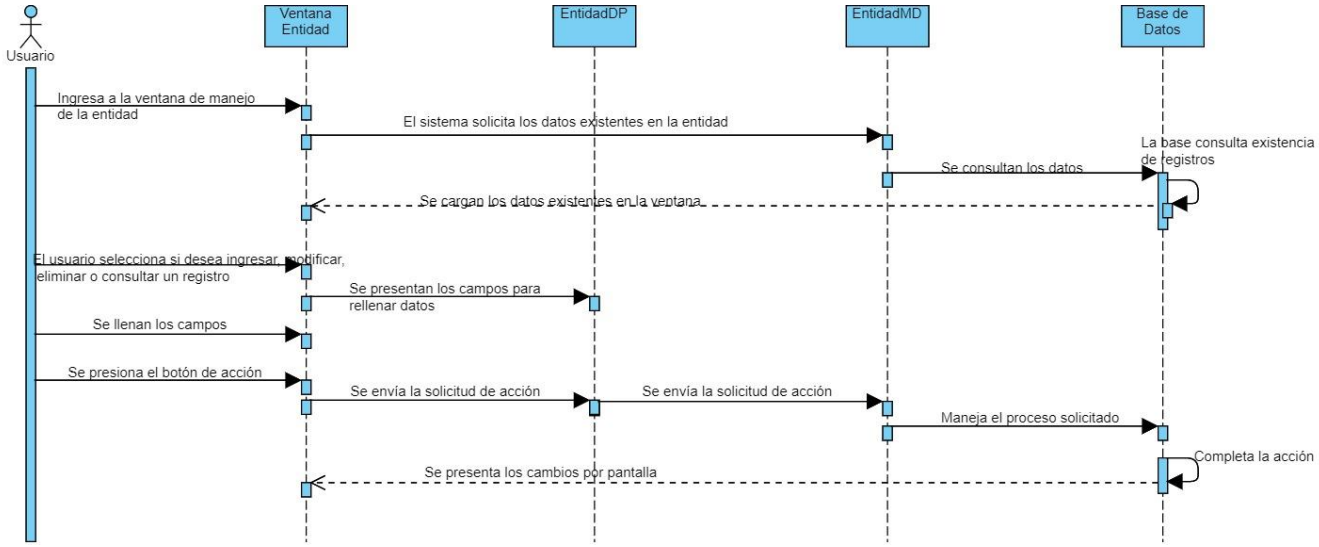


Ilustración 29: Diagrama de Secuencia Procesos

Ramos, A. (diciembre 2019).

3.1.2.7 Modelo Conceptual.

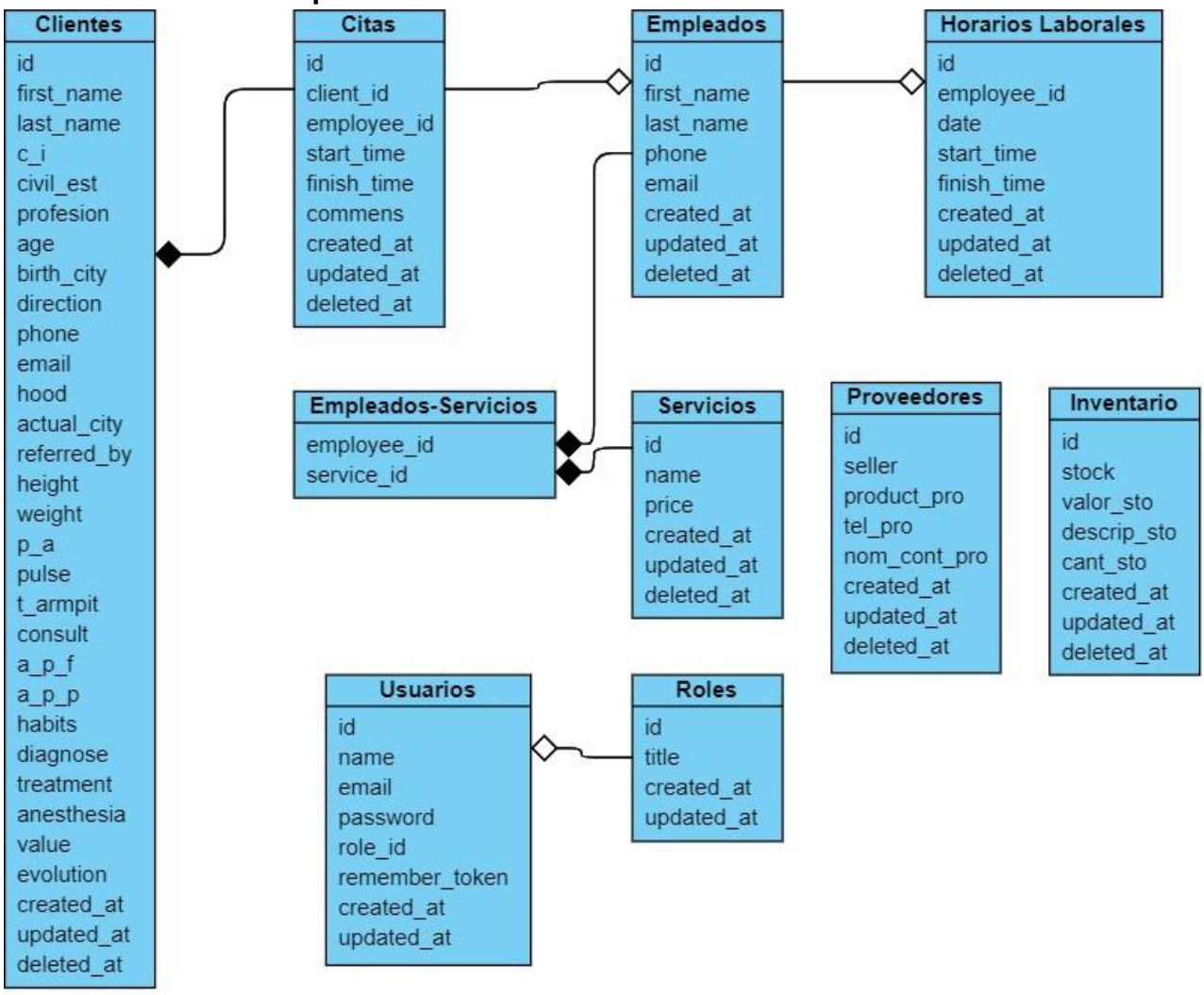


Ilustración 30: Diagrama de Modelo Conceptual

Ramos, A. (diciembre 2019).

3.1.2.8 Modelo E/R

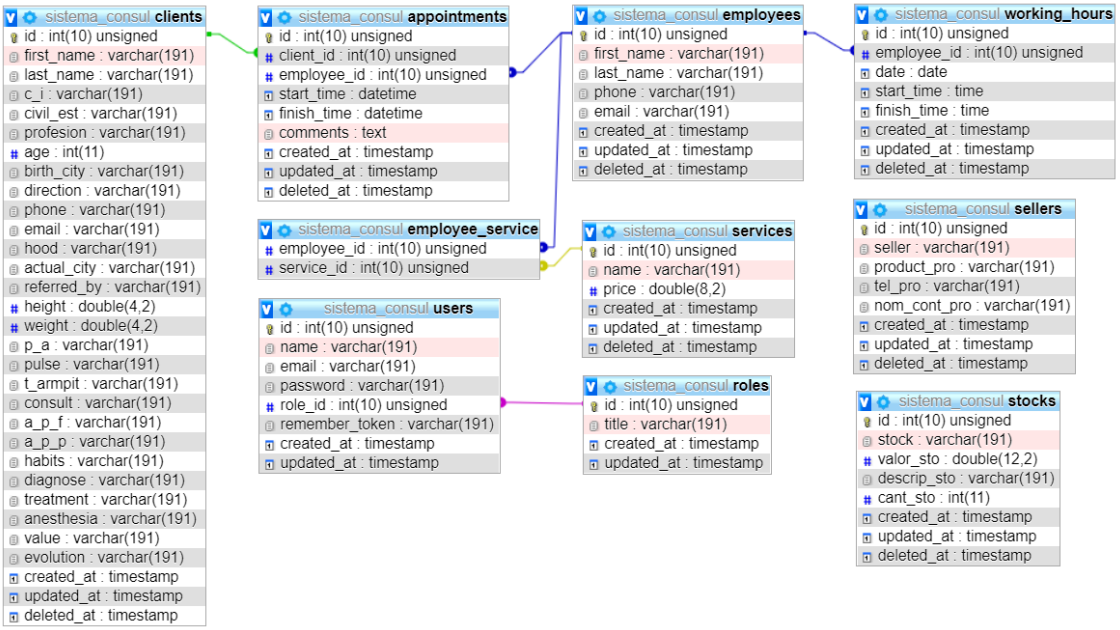


Ilustración 31: Diagrama de Modelo E/R

Ramos, A. (diciembre 2019).

Modelo E/R Completo

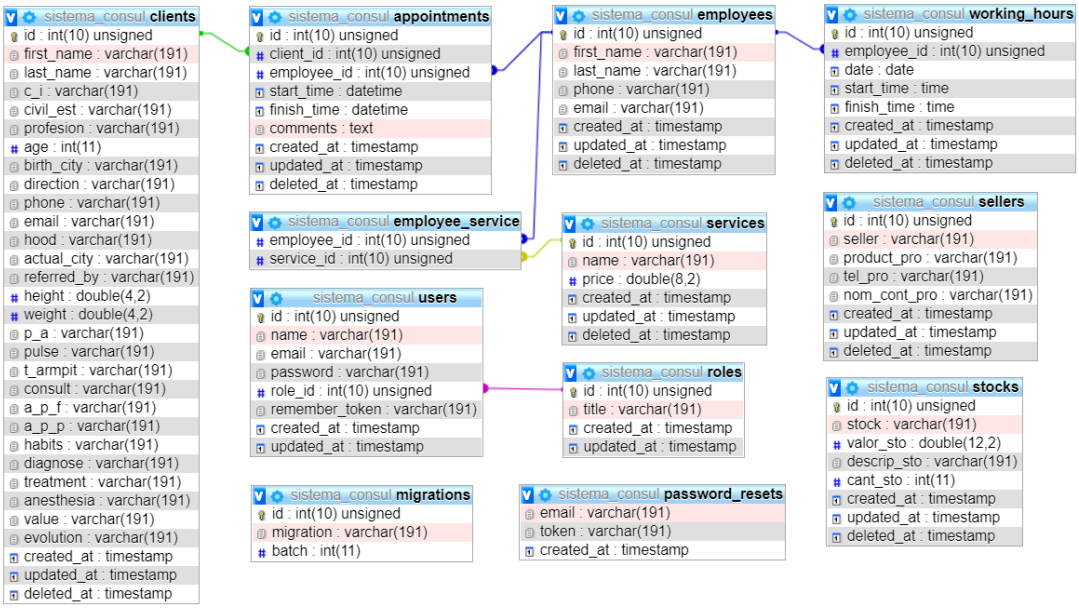


Ilustración 32: Diagrama de E/R Completo

Ramos, A. (diciembre 2019).

3.1.2.9 Plan de Pruebas de Integración.

El plan de pruebas de software se elabora para atender los objetivos de calidad en un desarrollo de sistemas, encargándose de definir aspectos como por ejemplo los módulos o funcionalidades sujetos a verificación, tipos de pruebas, entornos, recursos asignados, entre otros aspectos. Durante la fase de diseño esto es importante pues se comprueban las funcionalidades y su comportamiento esperado, así como un prototipo del desempeño esperado.

Integración por hilos: Se basa en probar las funciones de un programa siguiendo un proceso sencillo o solo partes específicas, sin embargo no una totalidad como la dependencia de clases.

Integración por dependencia de clases: Se basa en ir probando el sistema de forma jerárquica, se encargan de probar la interacción de los diversos componentes de sistemas y su interacción.

Las pruebas se realizaron con éxito.

Se determino que los componentes registran y trabajan con la información de manera adecuada, corregir errores de sintaxis en las vistas llevo el mayor tiempo, se entrega las funcionalidades del presente ciclo optimas y completas.

3.1.3 Informe de Pruebas

Semántica: Se basan en errores por denominaciones de variables, controladores, funciones o sintaxis del código generado, pude incluir cosas simples como una mayúsculas o minúscula as nombres de clases y rutas diferentes. De igual manera hace referencia a errores en la redacción de documentos.

Error 1: En la redacción del reporte de requerimientos los diagramas de la funcionalidad Trabajadores presentaban un error al leerse “Trabajdores”.

Estado: Corregido.

Código:

Error 1: El botón “Volver” presente en las vistas “index” de las funcionalidades del primer ciclo presentaba el problema de retornar a la última vista abierta sea cual sea esta, no redirigía al menú principal como se esperaba.

Fue corregido de la manera adecuada, cambiando el código presente en las vistas de este componente, se creó un componente del controlador para manejar el retorno a la vista que contiene el menú principal de administración, así como una ruta que nos dirige al controlador previamente mencionado.

Estado: Corregido.

Cambios:

Cambio 1: Debido al cambio del código del botón correspondiente al Error 1, se debió realizar cambios tanto en la línea de código del mismo, así como agregar una ruta y un componente en el controlador.

Cambio 2: Se debió modificar el nombre del controlador, modelo, rutas y vistas de “Inventory” a “Stock” con la finalidad de configurar de manera adecuada las rutas del controlador, permitiendo el funcionamiento completo de las vistas contenidas en la carpeta dotaciones, pues al crear las migraciones por la sintaxis de Laravel convertía en “Inventories” el plural, esto se repite con cualquier palabra que termine en “Y”.

Cambio 3: Se modifico el contenido del informe de implementación para visualizar los cambios previamente mencionados en este documento.

Pruebas de:

Caja Blanca: Se centran en los detalles procedimentales del software, por lo que su diseño está fuertemente ligado al código fuente. El ingeniero de pruebas escoge distintos valores de entrada para examinar cada uno de los posibles flujos de ejecución del programa y cerciorarse de que se devuelven los valores de salida adecuados.

Al estar basadas en una implementación concreta, si esta se modifica, por regla general las pruebas también deberán rediseñarse.

Resultado: Exitoso, los valores que se pueden ingresar se encuentran restringidos acordes al tipo de dato que cada campo espera recibir, de presentar un conflicto se informa el impedimento al usuario y debe de corregir los mismos antes de proseguir.

Caja Negra: Conociendo una función específica para la que fue diseñado el producto, se pueden diseñar pruebas que demuestren que dicha función está bien realizada. Dichas pruebas son llevadas a cabo sobre la interfaz del software, es decir, de la función, actuando sobre ella como una caja negra, proporcionando unas entradas y estudiando las salidas para ver si concuerdan con las esperadas.

Resultado: Exitoso, se probaron las funcionalidades por separado, antes de integrarse y ser probadas como módulos pertenecientes a un ciclo donde se les aisló acorde a cada funcionalidad y probo como una entidad aparte cada una.

Alfa/Alpha: Son pruebas de software realizadas cuando el sistema está en desarrollo y cuyo objetivo es asegurar que lo que estamos desarrollando es probablemente correcto y útil para el cliente. Por ejemplo podríamos desarrollar un prototipo de una parte del sistema que no estamos seguros de cómo implementar la interfaz. Si las pruebas alfa devuelven unos comentarios positivos entonces

podríamos seguir por esa vía. Si devolvieran resultados muy negativos tendríamos que replantear el problema para adaptarse mejor a los requerimientos.

Son normalmente realizadas contra un prototipo desarrollado rápidamente y con poco coste realizado para poder experimentar con él. De esta forma si las pruebas alfa fallan no hemos desperdiciado mucho tiempo ni dinero. Una vez que las pruebas alfa se completan, el prototipo debería ser descartado y aprovechar todo lo que se ha aprendido sobre el problema para el desarrollo de la versión final.

Resultado: Exitosa, se probaron diferentes maneras de programación de componentes y funcionalidades en Laravel, al final se descartaron aquellas prácticas poco óptimas o con muchos conflictos, aprovechando del aprendizaje la mejor manera de realizar rutas, vistas y poniendo en práctica estas al momento del desarrollo del sistema presente.

Beta: Son las pruebas de software que se realizan cuando el sistema está teóricamente correcto y pasa a ejecutarse en un entorno real. Es la fase siguiente a las pruebas Alpha. No importa lo bueno que sea nuestro proceso de desarrollo, siempre habrá fallos que no han sido descubiertos por los desarrolladores ni por el equipo de pruebas. Las pruebas beta son pruebas para localizar esos problemas que no han sido detectados y poder corregirlos antes de liberar una versión. Debería ser realizada por usuarios finales. Dependiendo de la naturaleza del software podría, por ejemplo, ser realizada por compañeros de trabajo, por algunos clientes reales, o por una combinación de ambos.

A los que realizan las pruebas beta se les suele llamar beta tester.

Resultado: Tras una reunión el día 27 de diciembre de 2019, se determinó el funcionamiento total de las funcionalidades previstas para el cierre del ciclo 2, dando por terminada la fase de pruebas, esto permite la entrega de la última etapa del presente ciclo de desarrollo, generando el aplicativo Laravel completo.

3.1.4 Informe Postmortem

Se basa en evaluar el cumplimiento de los objetivos del proyecto basado en el análisis de los datos que hemos ido recopilando en las diversas etapas de este. Esto nos permite encontrar aspectos que podamos mejorar en los ciclos o proyectos futuros y el proponer la mejor manera de mejor los aspectos mencionados.

Es importante verificar que los estándares de calidad fijados se hayan cumplido y en caso de no haberlo hecho definir el motivo y proponer soluciones para corregir estos de manera eficiente en el siguiente ciclo.

Todas estas observaciones y propuestas deben estar registradas en un plan de mejoramiento con la finalidad de llegar a un acuerdo con el equipo de trabajo en la búsqueda de la mejora.

¿Qué salió bien?

El primer y segundo ciclo lograron apegarse a los estándares impuestos, cumpliendo con la planificación de tiempo de desarrollo y eficiencia.

Se cumplió con el tiempo de entrega previsto, las pruebas fueron realizadas por cada funcionalidad y los resultados mostraron un perfecto funcionamiento de cada módulo.

Al integrar todas las funcionalidades y generar el aplicativo final, se determinó que dos funcionalidades no presentan conexión directa entre otros módulos como es el caso de las restantes funcionalidades.

¿Qué salió mal?

No se presentaron inconvenientes durante el proceso de desarrollo, por ende, no hubo incidencias de algo malo, se podría denotar un inconveniente como observación, el uso de herramientas como puertas, solicitudes y roles fue algo novedoso y desconocido al empezar, por lo que se debió designar mucho tiempo a la investigación de estas herramientas durante el primer ciclo, sin embargo en el desarrollo del segundo ciclo cabe recalcar esto no se repitió.

¿Qué actitudes debemos repetir?

- Aceptar críticas en base a opiniones constructivas y sinceras.
- Presentar cada fase acorde al tiempo planificado, de ser posible trabajar de forma eficiente y adelantar tiempos manteniendo la calidad.

¿Qué actitudes debemos evitar?

- Estrés y pánico general por el tiempo de entrega.
- Desgana o desapego al rol por motivos emocionales, académicos u otros.
- Atrasos en las fechas de presentación.
- Incumplimiento de actividades.
- Sucumbir a la comodidad de realizar un trabajo mediocre.

3.2 Ciclo 3 JAVA

El presente ciclo marca el desarrollo del proyecto en Java, para este ciclo se completará todas las funcionalidades relacionadas con facturación, estos módulos utilizarán como base de datos PostgreSQL.

Todo avance constará de una carta de conformidad con el avance y funcionalidades

3.2.1 Especificación de Requerimientos de Software

3.2.1.1 Introducción

El presente documento de especificación de requerimientos correspondiente al último ciclo de desarrollo, por esto los requerimientos cambian tanto en funcionales como en no funcionales, adaptándose a las solicitudes del beneficiario y el lenguaje de programación JAVA.

3.2.1.1.1 Propósito

El presente documento se realiza con la finalidad de presentar la recopilación de criterios, funcionalidades y requerimientos necesarios para el correcto desarrollo del producto.

3.2.1.1.2 Alcance

Se identificará al producto con el nombre de: Sistema de Facturación, sin embargo, para referencias futuras y comodidad de tiempo y espacio se referirá al presente con las siglas SF en base a sus iniciales.
El producto se desarrolla como un sistema de facturación para un consultorio médico, dividido en módulos con pantallas para cada funcionalidad de casos de uso, en las cuales el usuario podrá llenar campos de información que generaran entre algunas funciones, administración de servicios, control de usuarios, clientes y la correspondiente creación e impresión de facturas.
Se presenta un programa de escritorio por solicitud del beneficiario.
El programa busca ser lo más sencillo posible para la persona que deberá utilizarlo, siendo este enfoque una de nuestras prioridades.
Se manejará el modelo MVC como punto de partida para el desarrollo.

3.2.1.1.3 Personal involucrado

Nombre	Alexei Ramos
Rol	Desarrollador
Categoría profesional	Egresado
Responsabilidades	Supervisa el cumplimiento de la planificación establecida, desarrollo del producto, desarrollo de pruebas, presentación de informes, avances y reuniones.
Información de contacto	e-mail: alexei.ramos.rivera@hotmail.com Teléfono: 0999817839

3.2.1.1.4 Definiciones, acrónimos y abreviaturas

- Sistema de Facturación: SF
- Institute of Electrical and Electronics Engineers: IEEE
- Standard: STD/std
- DBMS: Data Base Management System.
- Visual Paradigm Online Diagrams: Herramienta para modelado en línea.
- PostgreSQL: Base de datos para el desarrollo.
- Windows: Sistema operativo privativo desarrollado por Microsoft.
- SDK: Software Development Kit.
- INTEL: Procesador con arquitectura x86/x64 desarrollada por Intel Corp.
- IDE: Integrated Development Environment.
- JAVA: Lenguaje de programación a utilizar.
- Framework: Entorno de trabajo o marco de trabajo es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

3.2.1.1.5 Referencias

Referencia	Titulo	Ruta	Fecha	Autor
1	IEEE std	IEEE Std 830-1998.	16-07-2019	Alexei Ramos
2	Definiciones, abreviaturas	Definiciones, acrónimos y abreviaturas	16-07-2019	Alexei Ramos

3.2.1.1.6 Resumen

El presente documento tendrá los diversos diagramas generales de funcionalidad, mismos que están sujetos a cambios, casos de uso y prueba, la recopilación de objetivos y especificará a detalle el servicio que se brindará al cliente, así como la estructura de desarrollo del producto. Se presentan dichos elementos con la esperanza de la aprobación del cliente para empezar el desarrollo y en caso de no conseguirla se recabarán el resto de los requerimientos e inquietudes de este para presentar un producto de acuerdo a un alto estándar de calidad.
Se estructura el producto en base al documento de requerimientos presentado por el instructor y se basa en el estándar IEEE Std 830-1998.

3.2.1.2 Descripción general

3.2.1.2.1 Perspectiva del producto

El presente sistema que se implementará para la creación de un Sistema de Facturación se lo realizará de manera independiente, debido a que con el software que se desarrollará se realizará todas las funciones a optimizar dentro del mismo.

3.2.1.2.2 Funcionalidad del producto, casos de uso general

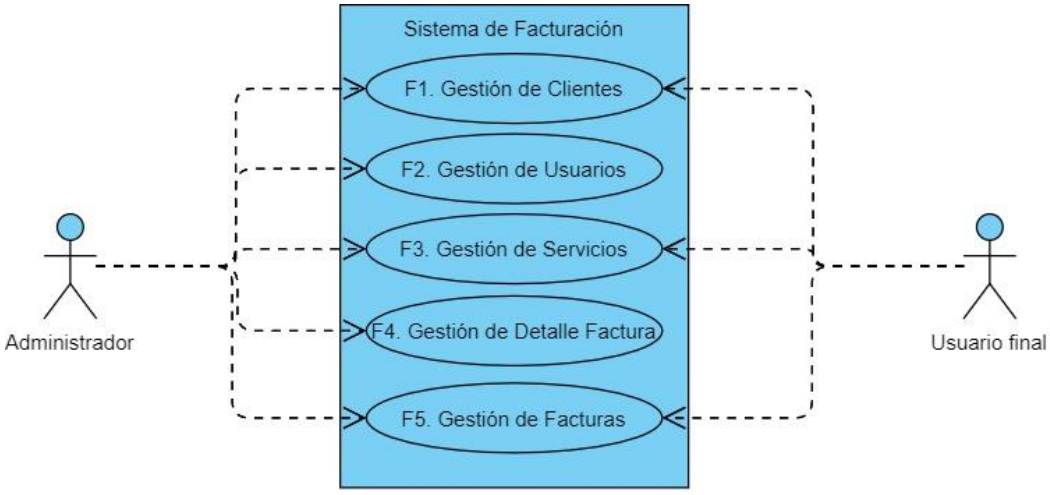


Ilustración 33: Diagrama de Casos Uso C3

Ramos, A. (diciembre 2020).

Las funcionalidades que presentará el proyecto corresponden a diferentes gestiones dentro de la facturación, entre estos tenemos:

- F1: Gestión de Clientes.
    - Controlar los roles que el sistema presentará, administrador y usuario simple de forma inicial. Se podrá ingresar, modificar,

eliminar los roles presentes en este módulo. Además, se podrá realizar consultas de tipo informativo. El presente módulo solo será visible para administradores.

- F2: Gestión de Usuarios.
    - Controlar los usuarios que el sistema presentará, estos estarán sujetos a un rol asignado respectivamente a sus funciones. Se podrá ingresar, modificar, eliminar los usuarios presentes en este módulo. Además, se podrá realizar consultas de tipo informativo. El presente módulo solo será visible para administradores.
- F3: Gestión de Servicios.
    - Controlar la creación de servicios que ofrece el consultorio médico, se podrá ingresar, modificar, eliminar y consultar la información acerca de los diferentes servicios como administrador y usuario simple.
- F4: Gestión de Detalle de Factura.
    - Controlar el ingreso de pacientes que son atendidos en el consultorio médico, se podrá ingresar, modificar, eliminar y consultar la información acerca de cada paciente como administrador y usuario simple.
- F5: Gestión de Factura.
    - Controlar las personas que trabajan en el consultorio médico, estos estarán sujetos a un servicio asignado respectivamente a sus funciones. Se podrá ingresar, modificar, eliminar los usuarios presentes en este módulo. Además, se podrá realizar consultas de tipo informativo. El presente módulo solo será visible para administradores.

3.2.1.2.3 Características de los usuarios

Tipo de usuario	Empleado: Usuario Final
Formación	Superior
Habilidades	Capaz de utilizar un dispositivo móvil u ordenador.
Actividades	Gestión de ingreso, modificación, consulta y eliminación de procesos de gestión de inventario, proveedores, pacientes, citas y servicios.

Tipo de usuario	Empleado: Administrador
Formación	Superior
Habilidades	Manejo de bases de datos.
Actividades	Gestión de entidades, entendimiento de lenguajes de programación, control del software.

3.2.1.2.4 Restricciones

El presente sistema será desarrollado en JAVA, por tanto, es necesario poseer en la máquina a usar el sistema las versiones actualizadas de esta herramienta asi como su IDE, siendo seleccionado NetBeans, debido a ello se escoge un equipo para ser el servidor del consultorio. Es importante señalar que el software utilizará como base de datos PostgreSQL. Para el diseño de los diagramas se utiliza el software en línea Visual Paradigm en su edición pagada.

El sistema está orientado a ser utilizado en un ordenador, para esto es necesario un equipo con conexión a internet por la conexión a base de datos.

3.2.1.2.5 Suposiciones y dependencias

El sistema funcionará correctamente en el sistema operativo Windows. Si se utiliza un sistema operativo basado en Linux, de soportar la versión de Java 1.7 como mínimo, no debería de presentar problemas al momento de su ejecución.

Todos los ordenadores y portátiles presentes en el consultorio médico utilizan el sistema operativo Windows.

3.2.1.3 Requisitos específicos

Número de requisito	RF 1		
Nombre de requisito	Gestión de Clientes.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El usuario debe ser capaz de poder gestionar clientes en el sistema, pudiendo hacer un control de los componentes que se ingresen, modifiquen, consulten o eliminen.

Número de requisito	RF 2		
Nombre de requisito	Gestión de Usuarios.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El usuario administrador debe ser capaz de poder gestionar usuarios en el sistema, pudiendo hacer un control de los componentes que se ingresen, modifiquen, consulten o eliminen.

Número de requisito	RF 3		
Nombre de requisito	Gestión de Servicios.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El usuario debe ser capaz de poder gestionar servicios en el sistema, pudiendo hacer un control de los componentes que se ingresen, modifiquen, consulten o eliminen.

Número de requisito	RF 4		
Nombre de requisito	Gestión de Detalle de Facturas.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El usuario debe ser capaz de poder consultar la información del detalle de factura en el reporte de factura.

Número de requisito	RF 5		
Nombre de requisito	Gestión de Facturas.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El usuario debe ser capaz de poder manejar trabajadores en el sistema, pudiendo hacer un control de los componentes que se ingresen, modifiquen, consulten o eliminen.

Número de requisito	RF 6		
Nombre de requisito	Documentar o presentar los instaladores ocupados.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☐ Alta/Esencial	☒ Media/Deseado	☐ Baja/ Opcional

El producto necesita documentar o incluir instaladores utilizados en el proceso de desarrollo.

Número de requisito	RF 7		
Nombre de requisito	Cumplir con todos los requisitos.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☐ Alta/Esencial	☒ Media/Deseado	☐ Baja/ Opcional

El producto necesita cumplir con todos los requisitos esenciales.

Número de requisito	RF 8		
Nombre de requisito	Liberar el producto dentro del plazo establecido.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Equipo		
Prioridad del requisito	☐ Alta/Esencial	☒ Media/Deseado	☐ Baja/ Opcional

El producto necesita ser liberado dentro del plazo establecido.

Número de requisito	RF 9		
Nombre de requisito	Conectar los módulos que dependan de otros de forma eficiente y sin errores.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Equipo		
Prioridad del requisito	☐ Alta/Esencial	☒ Media/Deseado	☐ Baja/ Opcional

El producto necesita permitir la integración y comunicación entre módulos cuyos campos están vinculados a otros.

Número de requisito	RF 10		
Nombre de requisito	Responder a un proceso administrado.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☐ Alta/Esencial	☒ Media/Deseado	☐ Baja/ Opcional

El producto necesita seguir un proceso que asegure la trazabilidad.

Número de requisito	RF 11		
Nombre de requisito	Liberar un producto de calidad.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Equipo		
Prioridad del requisito	☐ Alta/Esencial	☒ Media/Deseado	☐ Baja/ Opcional

El producto necesita estar libre de defectos en mayor medida.

Número de requisito	RF 12		
Nombre de requisito	Fácil de configurar y mantenible.		
Tipo	☒ Requisito	☐ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☐ Alta/Esencial	☒ Media/Deseado	☐ Baja/ Opcional

El producto necesita ser fácil de configurar y realizar mantenimiento.

Número de requisito	RF 13		
Nombre de requisito	Lenguaje de programación JAVA		
Tipo	☐ Requisito	☒ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El producto necesita utilizar JAVA.

Número de requisito	RF 14		
Nombre de requisito	Base de Datos PostgreSQL		
Tipo	☐ Requisito	☒ Restricción	
Fuente del requisito	Sistema		
Prioridad del requisito	☒ Alta/Esencial	☐ Media/Deseado	☐ Baja/ Opcional

El producto necesita utilizar el DBMS de *PostgreSQL*.

3.2.1.3.1 Requisitos comunes de las interfaces

El programa cuenta con ingreso de datos mediante cuadros de ingreso de texto y botones; y la salida de datos mediante tablas. Se manejará todo el tiempo interfaz gráfica.

3.2.1.3.1.1 Interfaces de usuario

El programa cuenta con una interfaz principal, que contiene un login, con la finalidad de redireccionar y autentificar el usuario que accede a su debida interfaz, mostrando solo los módulos permitidos a su rol.
Tras la debida validación se generan las interfaces de menú principal por el respectivo usuario, estas son el menú principal de administrador y usuario final.
El interfaz administrador tendrá un menú con todos los módulos disponibles.
La interfaz de usuario final no contará con el módulo de usuarios, el resto de módulos estarán disponibles.

3.2.1.3.1.2 Interfaces de hardware

Componentes de hardware que soporten arquitecturas INTEL o AMD.

3.2.1.3.1.3 Interfaces de software

Interfaces con un tamaño delimitado por el desarrollador que no abarquen toda la pantalla sino una parte de esta.

3.2.1.3.2 Requisitos funcionales

En base al diagrama general de casos de uso previamente presentado, obtenemos la funcionalidad a detalle esperada de los módulos a desarrollar.

3.2.1.3.2.1 FG: Funciones Múltiples

Diagrama de Casos de Uso, siguiente nivel, para el presente documento se decidió unificar los casos de uso cuya funcionalidad, diseño y estructura debido a su semejanza. Las funciones que cumplen este criterio de agrupamiento son:

- **F1: Clientes, F2: Usuarios y F3: Servicios.**

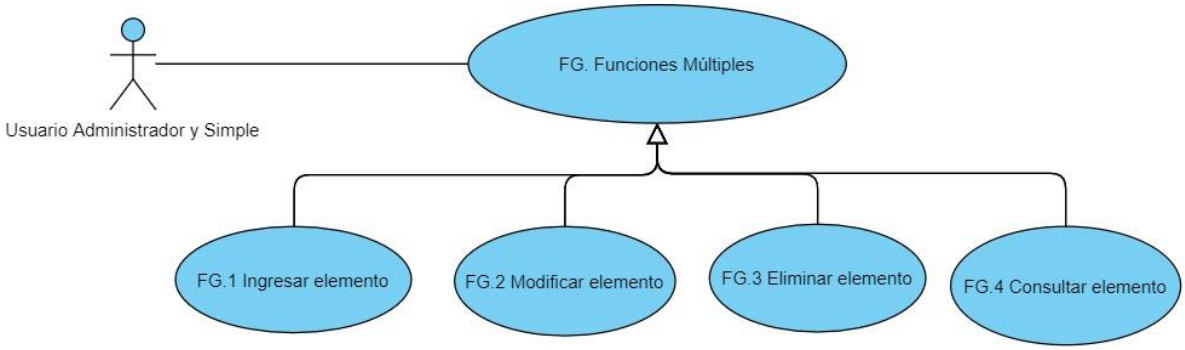


Ilustración 34: Funcionalidades Agrupadas JAVA

Ramos, A. (diciembre 2020).

3.2.1.3.2.1.1 FG.1: Funciones Múltiples: Diagrama a detalle, Ingresar Elemento, este caso de uso permite ingresar elementos nuevos a la tabla.

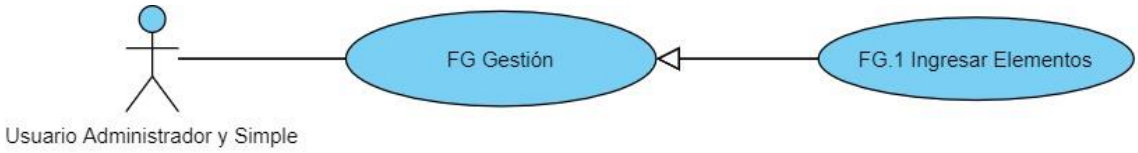


Ilustración 35: FG:F1 Ingresar

Ramos, A. (diciembre 2020).

Flujo principal:

1. El actor selecciona el módulo desde el menú principal.
2. El sistema presenta la ventana del módulo, desplegando los registros existentes.
3. El actor selecciona la opción de ingreso de elementos.
4. El sistema activa los campos de creación de elementos.
5. El actor ingresa los campos presentados y requeridos en el formulario.
6. El actor presiona en el botón guardar.
7. El sistema verifica los campos obligatorios estén llenos.
8. El sistema almacena los datos.

Flujo alterno:

8. Si el elemento ya existe, el sistema no permite el ingreso.

9. Ir al caso de uso FG.2: Modificar o FG.3: Eliminar.

Excepciones:

Código	**Causa**	**Solución**
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.2.1.3.2.1.2 FG.2: Funciones Múltiples: Diagrama a detalle, Modificar Elementos, este caso de uso permite modificar elementos existentes en la tabla.

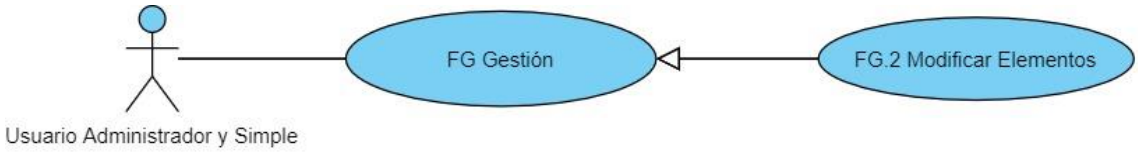


Ilustración 36: FG:F2 Modificar

Ramos, A. (diciembre 2020).

Flujo principal:

1. El actor selecciona el módulo desde el menú principal.
2. El sistema presenta la ventana del módulo, desplegando los registros existentes.
3. El actor selecciona un registro en la tabla de elementos disponibles.
4. El sistema carga los datos del elemento en campos para su edición.
5. El actor modifica los elementos que desee cambiar.
6. El actor presiona en el botón guardar.
7. El sistema almacena los datos.

Flujo alterno:

3. Si el producto no existe ir al caso de uso FG.1: Ingresar Elemento.

Excepciones:

Código	**Causa**	**Solución**
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.2.1.3.2.1.3 FG.3: Funciones Múltiples: Diagrama a detalle, Eliminar Elementos, este caso de uso permite al eliminar elementos presentes en la tabla.

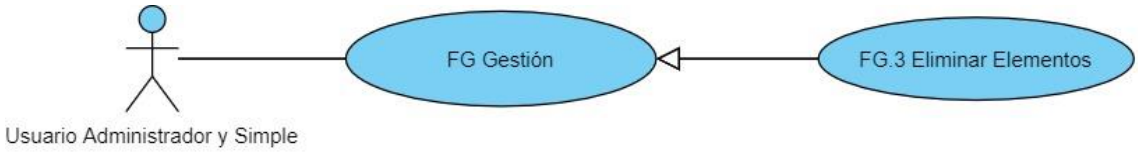


Ilustración 37: FG:F3 Eliminar

Ramos, A. (diciembre 2020).

Flujo principal:

1. El actor selecciona el módulo desde el menú principal.
2. El sistema presenta la ventana del módulo, desplegando los registros existentes.
3. El actor selecciona un registro en la tabla de elementos disponibles
4. El actor selecciona la opción de eliminar.
5. El sistema presenta una alerta pop-up para confirmar la eliminación del elemento.
6. El sistema presenta la tabla sin el elemento eliminado.
7. El sistema realiza un soft delete desde la base.

Flujo alterno:

3. Si el producto no existe ir al caso de uso FG.1: Ingresar Elemento.

Excepciones:

Código	**Causa**	**Solución**
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.2.1.3.2.1.4 FG.4: Funciones Múltiples: Diagrama a detalle, Consultar Elementos, este caso de uso permite consultar a detalle los elementos de la tabla.

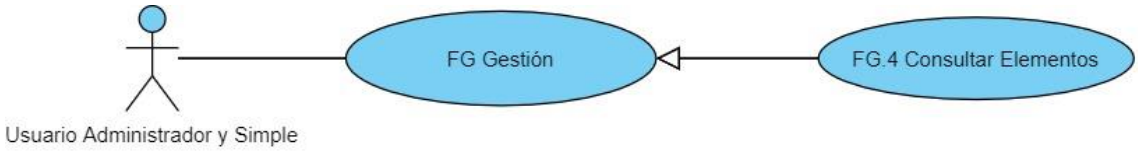


Ilustración 38: FG:F4 Consultar

Ramos, A. (diciembre 2020).

Flujo principal:

1. El actor selecciona el módulo desde el menú principal.
2. El sistema presenta la ventana del módulo, desplegando las opciones disponibles para consultar.
3. El sistema presenta la información del elemento consultado.

Excepciones:

Código	Causa	Solución
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.2.1.3.2.2 F4: Detalle Factura

Diagrama de Casos de Uso, siguiente nivel, para la funcionalidad de detalle de factura.

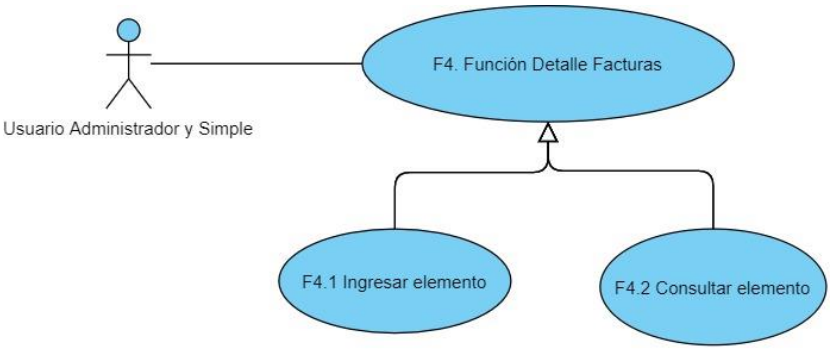


Ilustración 39: F4 Detalle Factura

Ramos, A. (diciembre 2020).

3.2.1.3.2.2.1 F4.1: Función Detalle Factura: Diagrama a detalle, Ingresar Elemento, este caso de uso permite ingresar elementos nuevos a la tabla de detalles de facturas.

Ilustración 40: F4:F1 Ingresar

Ramos, A. (diciembre 2020).

Flujo principal:

1. El actor selecciona el módulo Factura desde el menú principal.
2. El sistema presenta la ventana del módulo Factura, desplegando los registros existentes.
3. El actor selecciona la opción de ingreso de factura.
4. El sistema activa los campos de cabecera y cuerpo, este módulo maneja el cuerpo de servicios adquiridos.
5. El actor escoge los servicios a cobrar.
6. El actor presiona en el botón guardar.
7. El sistema verifica los campos obligatorios estén llenos.
8. El sistema almacena los datos.

Excepciones:

Código	Causa	Solución
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.2.1.3.2.2.2 F4.2: Función Detalle Factura: Diagrama a detalle, Consultar Elementos, este caso de uso permite consultar a detalle los elementos de la tabla de detalle de factura.

Ilustración 41: F4:F2 Consultar

Ramos, A. (diciembre 2020).

Flujo principal:

1. El actor selecciona el módulo Reporte cobros desde el menú principal.
2. El sistema presenta la ventana de reportes.
3. El actor selecciona la opción de consultar cobros.
4. El sistema despliega la ventana de consulta de facturas.
5. El sistema presenta la información de la factura consultada.

Excepciones:

Código	Causa	Solución
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.2.1.3.2.3 F5: Función Facturas

Diagrama de Casos de Uso, siguiente nivel, para la funcionalidad de Facturas.

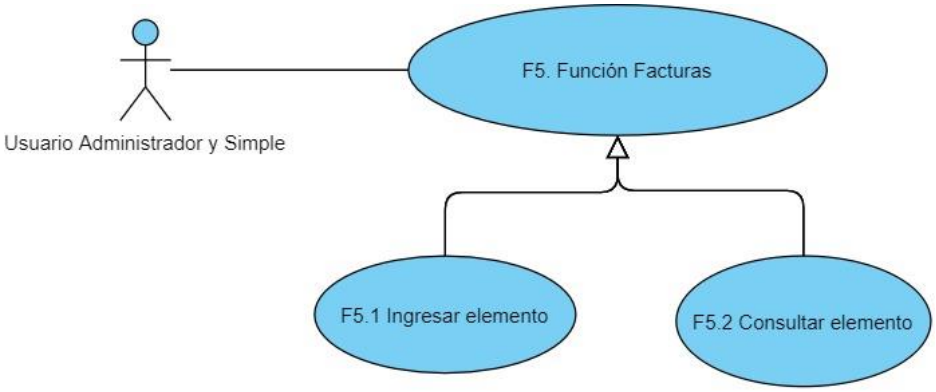


Ilustración 42: F5 Facturas

Ramos, A. (diciembre 2020).

3.2.1.3.2.3.1 F5.1: Función Factura: Diagrama a detalle, Ingresar Factura, este caso de uso permite ingresar facturas vinculados a un cliente y servicio.

Ilustración 43: F4:F2 Consultar

Ramos, A. (diciembre 2020).

Flujo principal:

1. El actor selecciona el módulo Factura desde el menú principal.

2. El sistema presenta la ventana del módulo Factura, desplegando los registros existentes.
3. El actor selecciona la opción de ingreso de factura.
4. El sistema activa los campos de cabecera y cuerpo, este módulo maneja el cuerpo de servicios adquiridos.
5. El actor escoge el cliente a facturar.
6. El actor escoge los servicios a cobrar.
7. El actor presiona en el botón guardar.
8. El sistema verifica los campos obligatorios estén llenos.
9. El sistema almacena los datos.

Flujo alterno:

9. Si no existen servicios o clientes, ir al caso de uso F1.1: Crear Cliente y F3.1: Crear Servicio.

Excepciones:

Código	**Causa**	**Solución**
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.2.1.3.2.3.2 F5.2: Función Facturas: Diagrama a detalle, Consultar Elemento, este caso de uso permite ver las facturas generadas.

Ilustración 44: F5:F1 Ingresar

Ramos, A. (diciembre 2020).

Flujo principal:

1. El actor selecciona el módulo Reporte cobros desde el menú principal.
2. El sistema presenta la ventana de reportes.
3. El actor selecciona la opción de consultar cobros.
4. El sistema despliega la ventana de consulta de facturas.
5. El sistema presenta la información de la factura consultada.

Excepciones:

Código	Causa	Solución
E_1	Error en el ingreso de datos	Verificar si el ingreso de datos es acorde con lo requerido.
E_2	Error en conexión	Comunicarse con programador.

3.2.1.3.3 Requisitos no funcionales

a. Del Producto

i. El programa debe ser utilizable de manera adecuada por el usuario sin presentar errores o bugs, con un tiempo de respuesta óptimo.
ii. El programa debe ser amigable para el usuario e intuitivo.
iii. El programa debe poder ejecutarse sin fallas o caídas.
iv. El programa debe ser un aplicativo de escritorio, cuyo sistema operativo es Windows 10.

b. Organizacionales

i. Se entrega el programa totalmente funcional.
ii. El tercer ciclo implementara las funcionalidades críticas establecidas en la reunión que se llevó a cabo el día 06 de noviembre de 2020.

c. Externos

i. La parte aplicativa debe poder relacionarse sin conflictos con la base de datos.
ii. Se usarán solo los programas presentados en el respectivo informe y aprobados por el supervisor.
iii. El personal que utilizara el producto será el responsable de las acciones que se realicen con el mismo, fuera del marco permitido por la empresa y se tomaran sanciones por parte del organismo pertinente dentro de la misma.

3.2.1.3.3.1 Seguridad

El programa manejará encapsulación lo que defenderá al programa contra ataques maliciosos a nivel de código. Además, el programa defenderá información sensible.

3.2.1.3.3.2 Disponibilidad

El programa tendrá siempre un 100% de disponibilidad mientras el equipo esté encendido y posea una conexión a internet para su base de datos.

3.2.1.3.3.3 Mantenibilidad

El proyecto será desarrollado acorde a diversos módulos y sus respectivos controladores, se definirán las rutas del mismo así como sus interfaces, al estar documentado a detalle, permite su mantenimiento adecuado.

3.2.1.3.4 Otros requisitos

- Documentar los códigos utilizados para etiquetas en la base de datos.
- Control de usuarios, tablas, base datos más restringidos.
- Sugerencias, advertencias, confirmaciones para las interfaces del cotizador por campos o botones clave.

3.2.1.4 Apéndices

Plan de Pruebas del Sistema

Manejo de Funcionalidades Básicas: Este caso de prueba busca efectuar pruebas en el manejo de todo el sistema, dada la similitud de módulos se han agrupado las primeras cuatro funcionalidades como FG: Funcionalidades múltiples, los módulos que se agruparon son: **F1: Roles, F2: Usuarios, F3: Servicios, F4: Pacientes, F8: Proveedor y F9: Inventario.**

De igual manera las pruebas para las funcionalidades: **F5: Empleados, F6: Horarios Laborales y F7: Citas** son similares a las que serán realizadas en módulos anteriores, con la diferenciación que se hará énfasis a la conexión entre servicios y empleados o trabajadores. Verificando la correcta vinculación de un servicio a un trabajador, de un trabajador a una cita y de un servicio a una cita.

Precondiciones:

Este caso de prueba depende de la terminación del ciclo de desarrollo, durante los informes de pruebas se explorará esta tabla a profundidad módulo por módulo.

Entradas	Resultados	Caso de Uso	Exitoso
1. El actor selecciona	Despliega menú	FG/F4/F5	X

opción X del menú principal			
2. El sistema presenta la ventana de X	Despliega ventana	FG/F4/F5	X
3. El actor ingresa datos del formulario	El sistema verifica datos del formulario	FG/F4/F5	X
4. El actor presiona Ingresar	El sistema envía los datos del formulario	FG/F4/F5	X
5. El actor presiona Ingresar	El sistema almacena los datos	FG/F4/F5	X
6. El actor ingresa el código del elemento	El sistema verifica el código de elemento ingresado	FG/F4/F5	X
7. El sistema presenta la ventana de consulta de X.	Despliega ventana	FG/F4/F5	X
8. El sistema presenta las X existentes en una tabla	Despliega datos en tabla	FG/F4/F5	X
9. El actor ingresa parámetro X de la función escogida	El sistema presenta los datos de la función escogida.	FG/F4/F5	X
10. El actor ingresa el código de X	El sistema presenta los datos de la X	FG/F4/F5	X

3.2.2 Especificación de Diseño de Software

3.2.2.1 Definición.

En esta fase definimos la arquitectura de la aplicación, diagramas UML, tales como: Clases, Actividades, Paquetes y el diagrama de base de datos de Entidad-Relación. Finalmente definimos el plan de pruebas de integración. Ya que el informe comparte la misma estructura de ciclos anteriores se mostrará aquellas partes que han cambiado en el presente ciclo contra los anteriores.

3.2.2.2 Arquitectura.

a. Arquitecturas MVC.

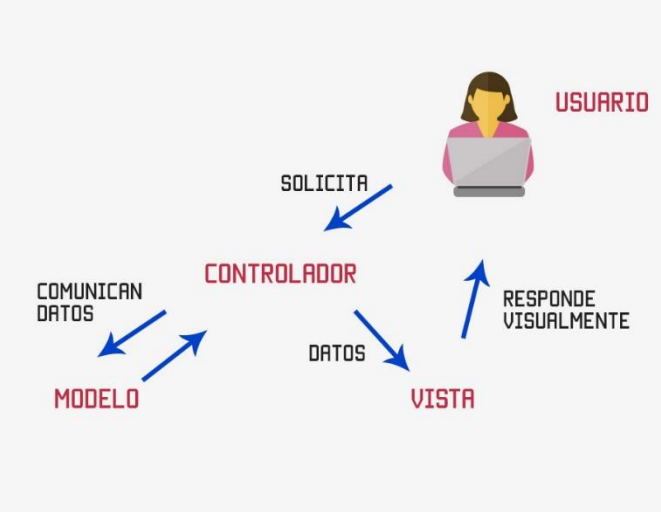


Ilustración 45: Modelo-Vista-Controlador C3

(Hernandez, 2015).

Al igual que nuestro aplicativo Laravel, el desarrollo del ciclo 3 con JAVA contara con una base MVC, con la diferenciación de la ceración de paquetes específicos para el manejo de conexión, generación de reportes PDF y un paquete de diseño, almacenamiento y manejo de imágenes para dar estilo a los botones del sistema.

3.2.2.3 Diagrama de Aplicación

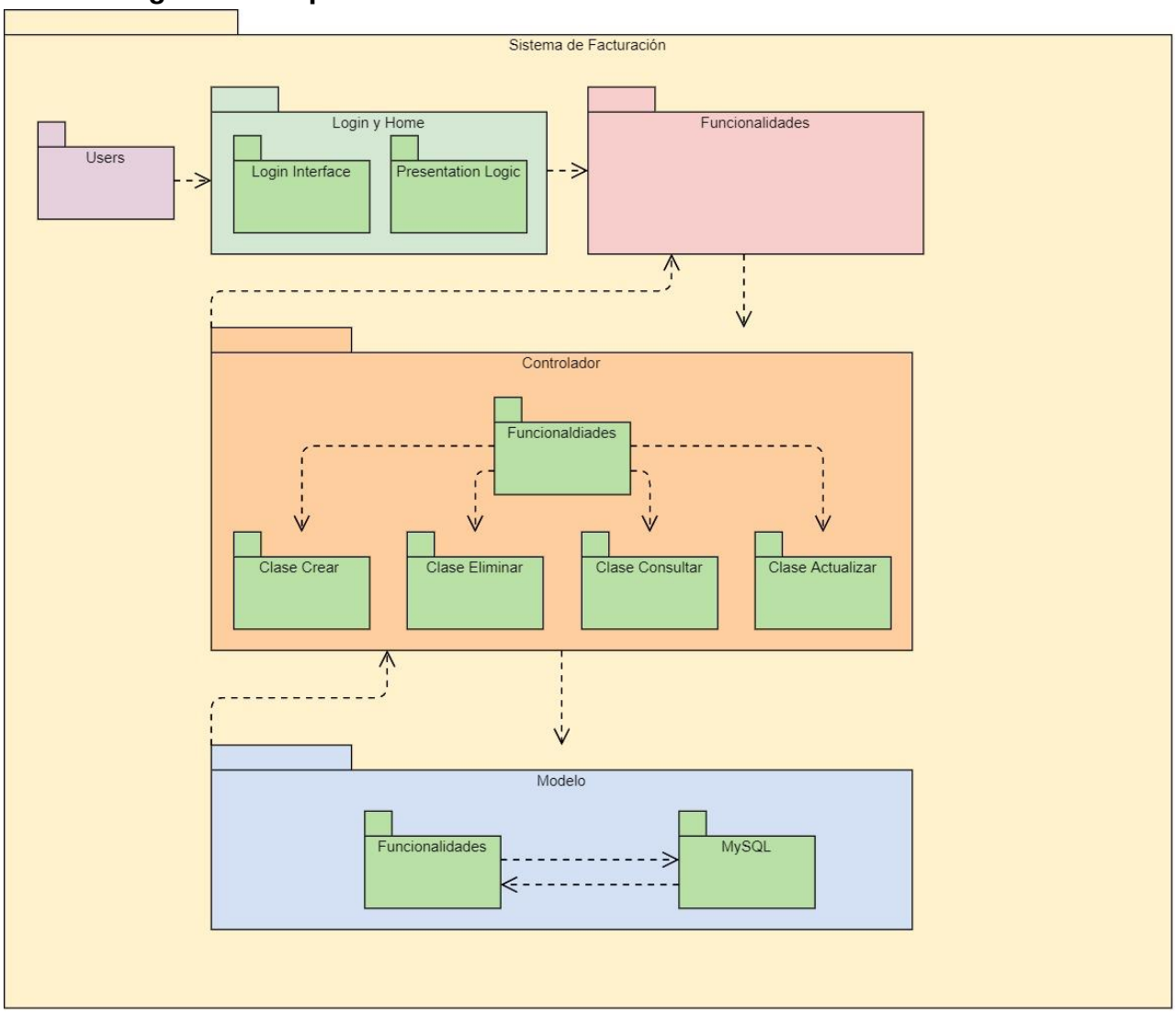


Ilustración 46: Diagrama de Aplicación

Ramos, A. (diciembre 2020).

3.2.2.4 Diagrama de Paquetes Ciclo 3

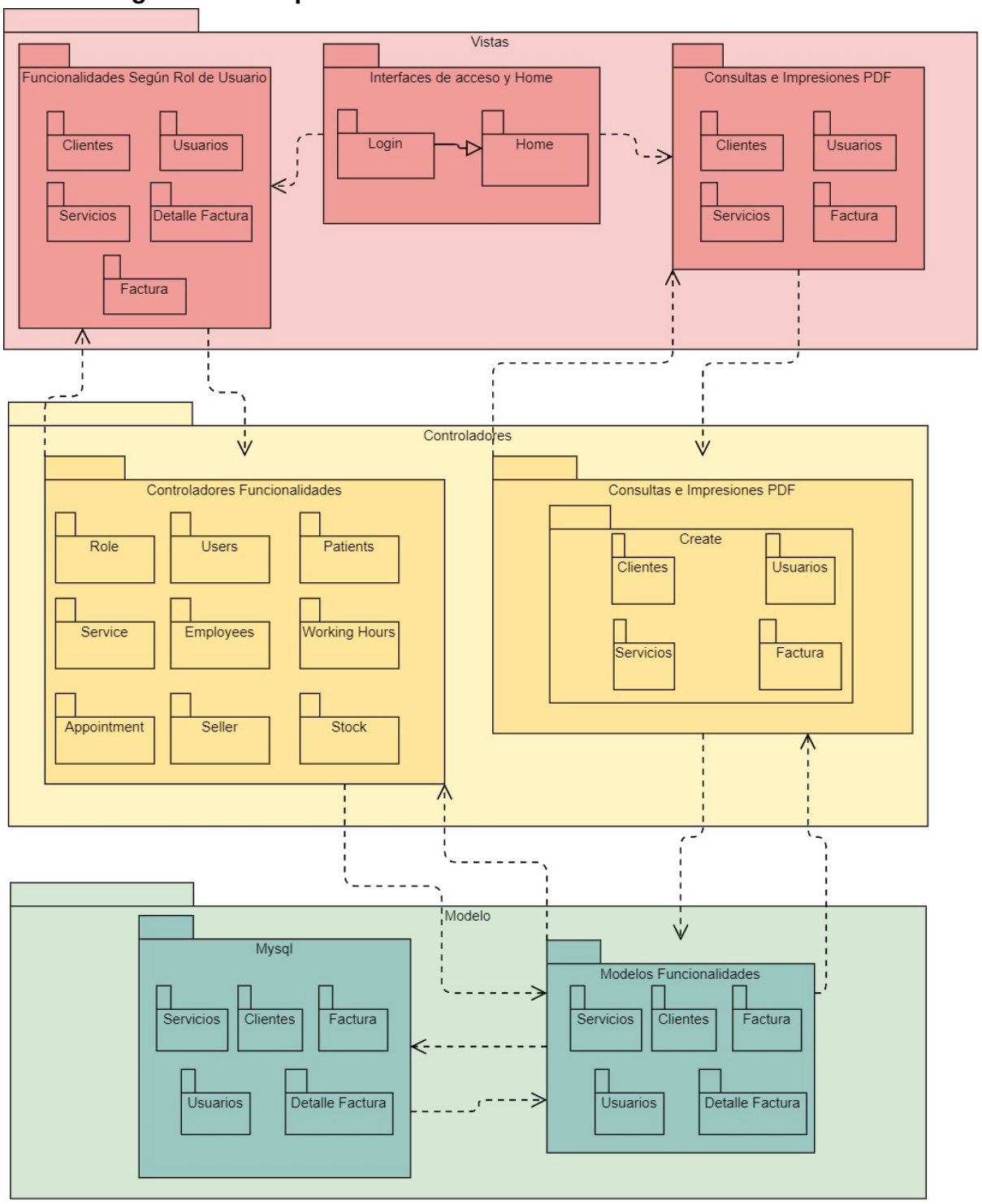


Ilustración 47: Diagrama de Paquetes C3

Ramos, A. (diciembre 2020).

3.2.2.5 Diagrama de Clases.

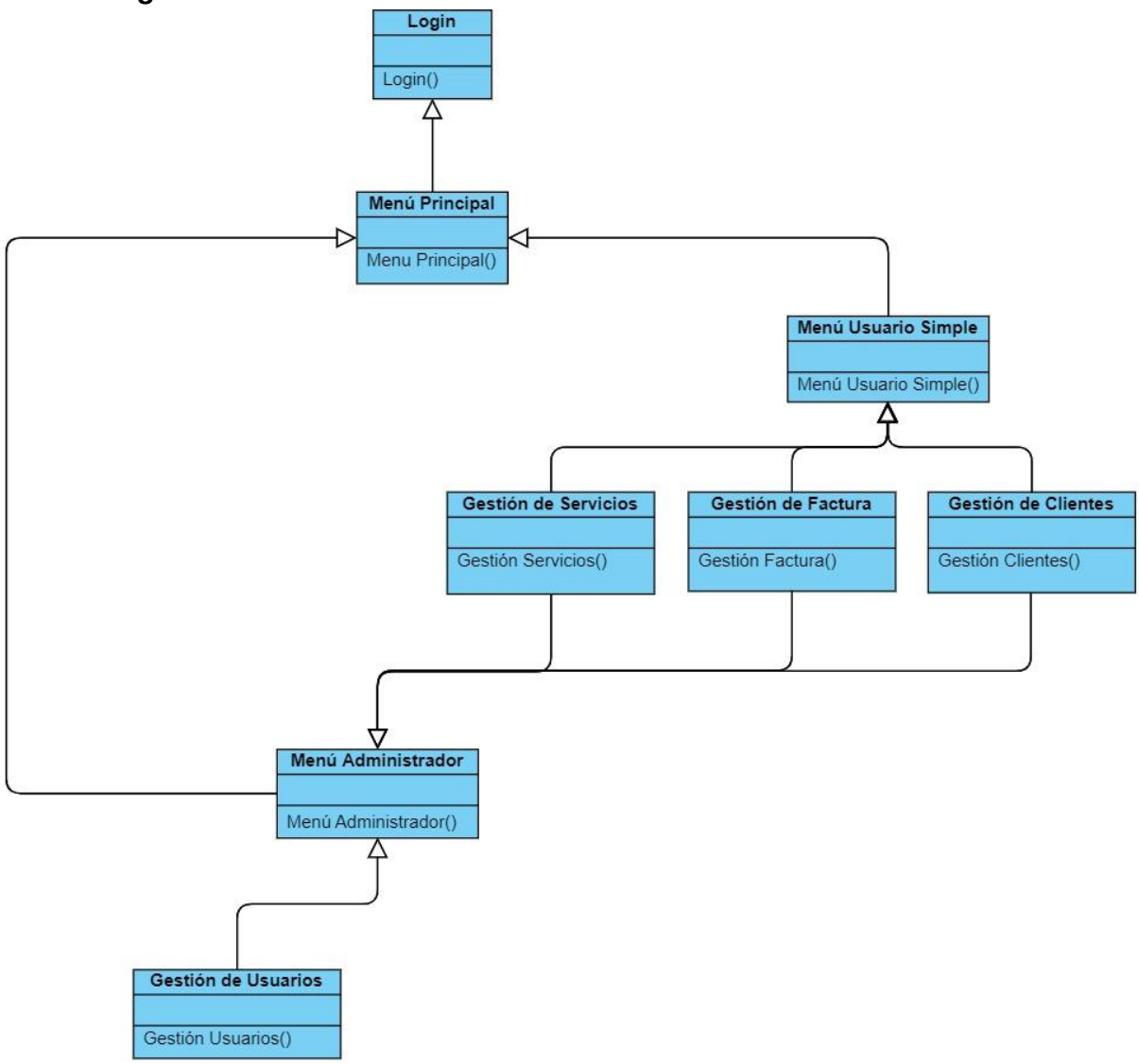


Ilustración 48: Diagrama de Clases C3

Ramos, A. (diciembre 2020).

3.2.2.6 Diagrama de Secuencia.

Ingreso al sistema.

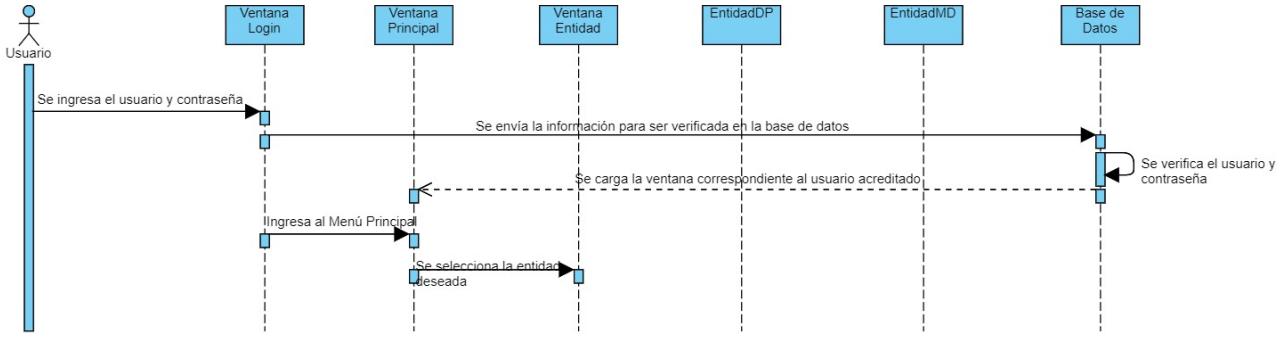


Ilustración 49: Diagrama de Secuencia C3 Ingreso

Ramos, A. (diciembre 2020).

Procesos.

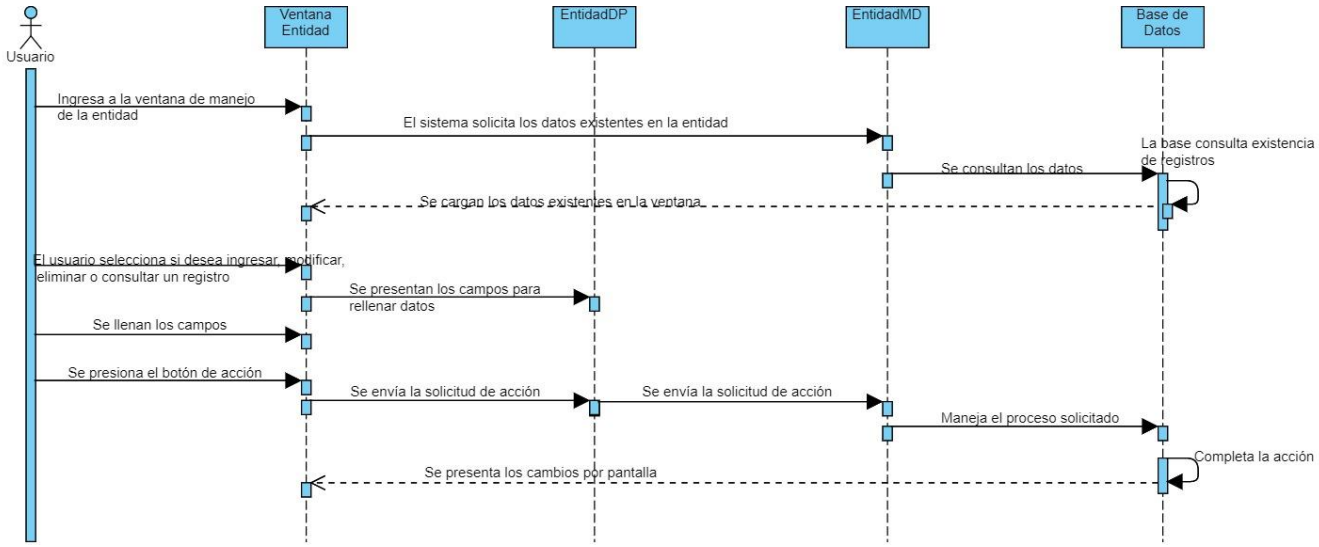


Ilustración 50: Diagrama de Secuencia C3 Procesos

Ramos, A. (diciembre 2020).

3.2.2.7 Modelo Conceptual.

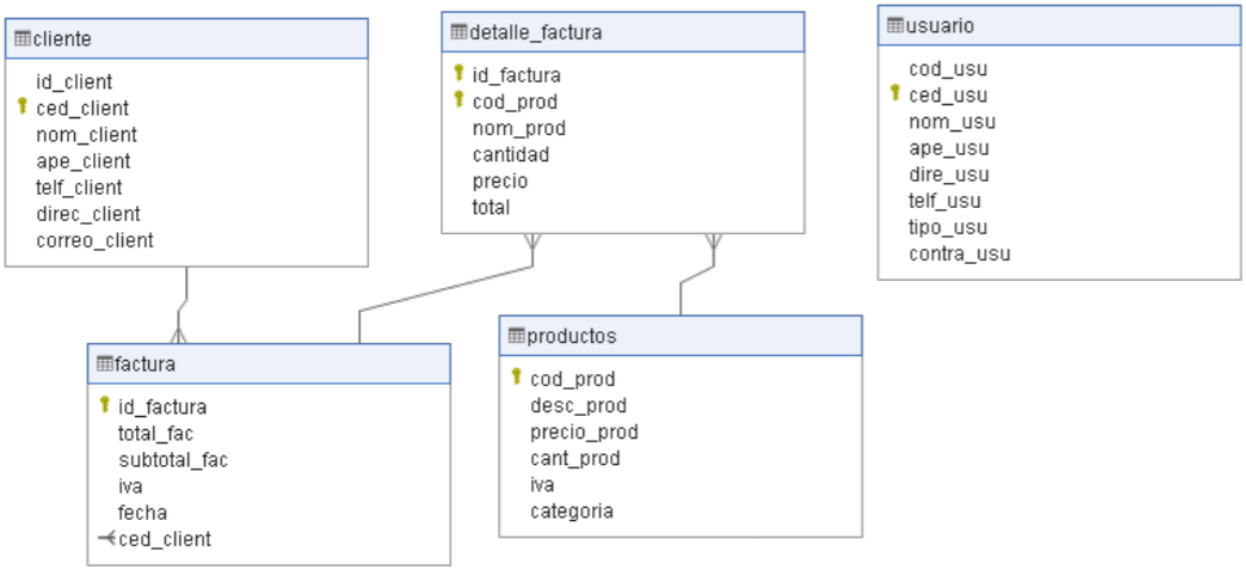


Ilustración 51: Diagrama de Modelo Conceptual C3

Ramos, A. (diciembre 2020).

Modelo E/R

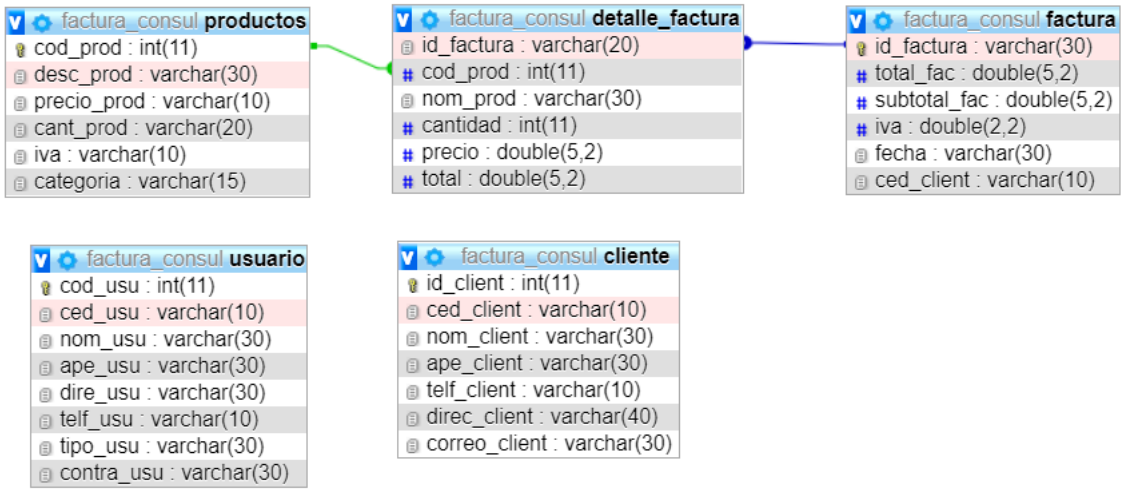


Ilustración 52: Diagrama de Modelo E/R C3

Ramos, A. (diciembre 2020).

3.2.2.8 Plan de Pruebas de Integración.

El plan de pruebas de software se elabora para atender los objetivos de calidad en un desarrollo de sistemas, encargándose de definir aspectos como por ejemplo los módulos o funcionalidades sujetos a verificación, tipos de pruebas, entornos, recursos asignados, entre otros aspectos. Durante la fase de diseño esto es importante pues se comprueban las funcionalidades y su comportamiento esperado, así como un prototipo del desempeño esperado.

Integración por hilos: Se basa en probar las funciones de un programa siguiendo un proceso sencillo o solo partes específicas, sin embargo no una totalidad como la dependencia de clases.

Integración por dependencia de clases: Se basa en ir probando el sistema de forma jerárquica, se encargan de probar la interacción de los diversos componentes de sistemas y su interacción.

Las pruebas se realizaron con éxito.

Se determino que los componentes registran y trabajan con la información de manera adecuada, corregir errores de sintaxis en las vistas llevo el mayor tiempo, se entrega las funcionalidades del presente ciclo optimas y completas.

3.2.3 Informe de Pruebas

Semántica

Error 1: En la ventana de consulta se lee “Etregué” en la documentación para la guía se notó este error y se lo resolvió .

Estado: Corregido.

Código:

Error 1: El botón “Agrega” presente en la vista “Registro usuario” de la funcionalidad Usuarios presentaba el problema de no registrar usuarios nuevos.

Fue corregido de la manera adecuada, cambiando el código presente en las vistas de este componente, es posible el ingresar usuarios nuevos mientras no hayan sido registrados previamente.

Estado: Corregido.

Cambios:

Cambio 1: Debido al cambio del código del botón correspondiente al Error 1, se debió realizar cambios tanto en la línea de código del mismo, así como agregar una ruta y un componente en el controlador.

Cambio 2: Se modifico el contenido del informe de implementación para visualizar los cambios previamente mencionados en este documento.

Pruebas de

Caja Blanca:

Resultado: Exitoso, los valores que se pueden ingresar se encuentran restringidos acordes al tipo de dato que cada campo espera recibir, de presentar un conflicto se informa el impedimento al usuario y debe de corregir los mismos antes de proseguir.

Caja Negra:

Resultado: Exitoso, se probaron las funcionalidades por separado, antes de integrarse y ser probadas como módulos separados y en forma de entidad completa.

Alfa:

Resultado: Exitosa, se probaron diferentes maneras de programación de componentes y funcionalidades en JAVA, al final se descartaron aquellas prácticas poco óptimas o con muchos conflictos, aprovechando del aprendizaje la mejor manera de realizar rutas, vistas y poniendo en práctica estas al momento del desarrollo del sistema presente, se manejaron diversas capaz de conexión y vinculación de elementos para tener un mejor control y visibilidad del proceso.

Beta:

Resultado: Tras una reunión el día 11 de diciembre de 2020, se determinó el funcionamiento total de las funcionalidades previstas para el cierre del ciclo 3, dando por terminada la fase de pruebas, esto permite la entrega de la última etapa del presente ciclo de desarrollo, generando el aplicativo JAVA completo.

3.2.4 Informe Postmortem

¿Qué salió bien?

Durante el tercer ciclo se logró apegarse a los estándares impuestos, cumpliendo con la planificación de tiempo de desarrollo y eficiencia, pese al cambio de herramienta de desarrollo y el lenguaje de programación.

Las pruebas fueron realizadas por cada funcionalidad y los resultados mostraron un correcto funcionamiento de cada módulo.

Tras generar el aplicativo completo se puede apreciar un flujo de ventanas adecuado, presentando cada vista en un tiempo aceptable y cumpliendo con todas las funcionalidades para cada módulo.

¿Qué salió mal?

Se presento un error al momento de visualizar las vistas o formularios, esto fue debido a una librería que debió ser actualizada encargada de controlar interfaces gráficas, el error no estaba relacionado con el código sino con un componente de las herramientas de NetBeans.

¿Qué actitudes debemos repetir?

- Aceptar críticas en base a opiniones constructivas y sinceras.
- Presentar cada fase acorde al tiempo planificado, de ser posible trabajar de forma eficiente y adelantar tiempos manteniendo la calidad.

¿Qué actitudes debemos evitar?

- Estrés y pánico general por el tiempo de entrega.
- Desgana o desapego al rol por motivos emocionales, académicos u otros.
- Atrasos en las fechas de presentación.
- Incumplimiento de actividades.
- Sucumbir a la comodidad de realizar un trabajo mediocre.

Capítulo 4 – Codificación

Este capítulo da la introducción necesaria para comprender el desarrollo del código del proyecto, las bases de las que se parte, comandos necesarios y aplicación de diseño.

4.1 Artisan Console

Laravel nos proporciona una interfaz basada en línea de comandos, que contienen herramientas para poder generar una aplicación de forma sencilla y cómoda, entre algunos de los comandos más destacables se encuentran, el crear controladores, migraciones, modelos, limpiar o borrar cache, crear seeders o pobladores de tablas, crear vistas.

4.2 Migraciones

Esta herramienta nos permite crear respaldos de la base de datos que presenta un sistema Laravel, es una forma de crear la estructura de las tablas de forma interna y luego exportada hacia el motor de base de datos seleccionado.

De esta manera se puede compartir los esquemas de base de datos de forma sencilla y editarlos directamente, eliminando la necesidad de realizar cambios primera en la base de datos y luego reflejarlos al aplicativo.

Podemos encontrar una carpeta llamada “migrations” dentro de la carpeta de componentes “database”.

Se pueden crear migraciones para tablas o una combinación de MVC y migración por medio de comandos en la consola Artisan.

La ruta de acceso a esta carpeta en el aplicativo Laravel es: C:\Laravel\Sistema Administracion\database\migrations.

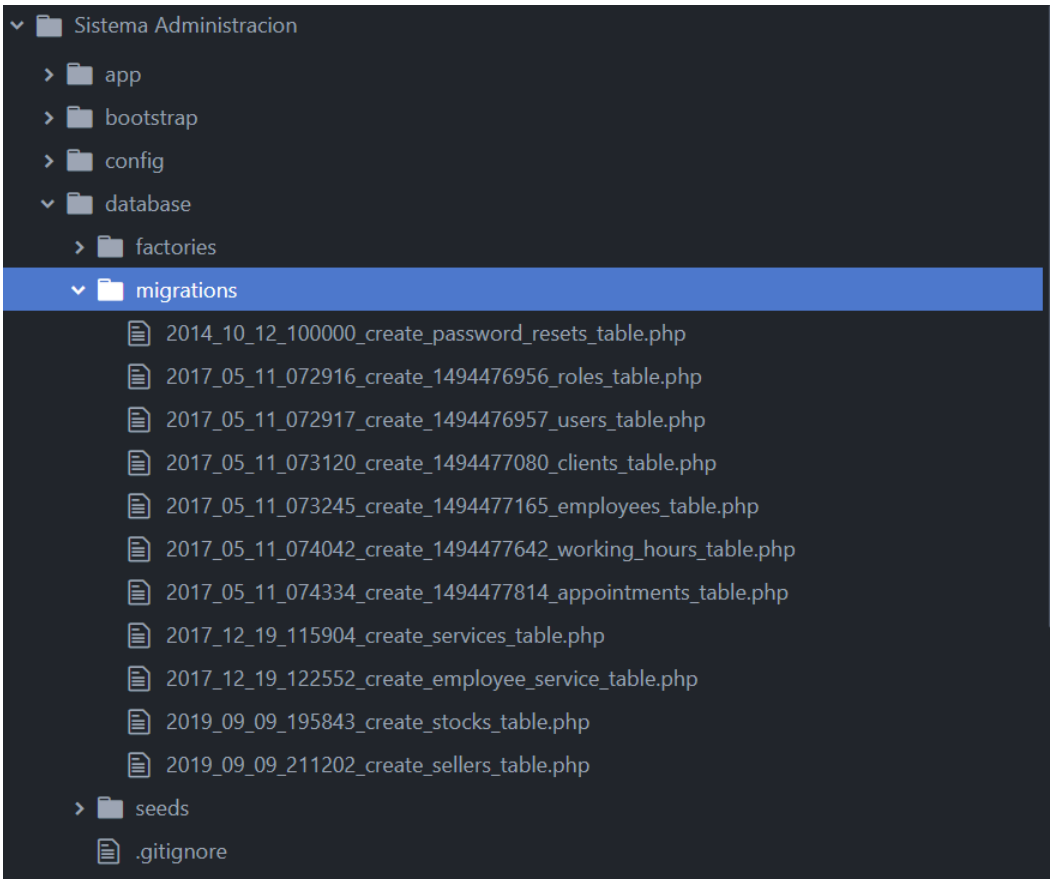


Ilustración 53: Migrations

Ramos, A. (diciembre 2020).

El presente proyecto presenta la carga de migraciones de tablas en el siguiente orden:

1. Restauración de contraseñas
2. Roles
3. Usuarios
4. Clientes
5. Trabajadores o Empleados
6. Horarios Laborales
7. Citas
8. Servicios
9. Trabajador-Servicio
10. Inventario
11. Proveedores

Tras ejecutar las migraciones creamos la estructura de base de datos del proyecto.

4.3 Pobladores

Una de las herramientas más útiles de Laravel, nos permite crear pobladores o seeders, capaces de llenar con información preconfigurada las tablas generadas en las migraciones. Esta información puede ser utiliza para poblar datos de prueba o poblar el aplicativo final con información válida para el uso del mismo.

Para el presente proyecto se ha designado el nombre de cada poblador o clase semilla acorde a las tablas creadas.

Los pobladores están almacenados en la carpeta “seeds”, la ruta de acceso a esta carpeta en el aplicativo es: C:\Laravel\Sistema Administracion\database\seeds.

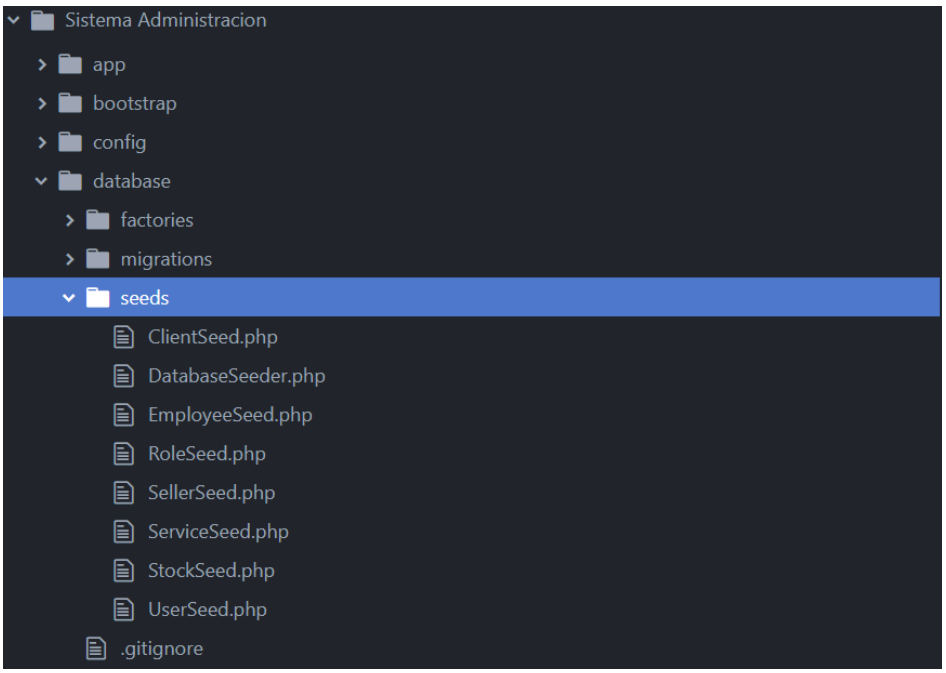


Ilustración 54: Seeds

Ramos, A. (diciembre 2020).

El presente proyecto presenta la carga de semillas en el siguiente orden:

1. RoleSeed
2. UserSeed
3. SellerSeed
4. StockSeed
5. ServiceSeed
6. ClientSeed
7. EmployeeSeed

Es importante recalcar que el orden de las semillas puede variar a especificación del desarrollador editando el mismo dentro del archivo DatabaseSeeder.

4.4 Listas Enlazadas

Estructura de datos basada en punteros, nos permite manejar datos de forma dinámica y permite concatenar un nodo tras otro, esto puede entenderse como una cadena formada elemento almacenado en un nodo tras otro, dándonos variabilidad al momento de crear estructuras e importar o llamar datos cuando el tamaño o cantidad de datos es variable.

Se utilizo este tipo de estructuras para invocar la información que será enviada a los reportes PDF en el aplicativo JAVA.

Los archivos referentes a cada lista utilizada están en el paquete Listas.

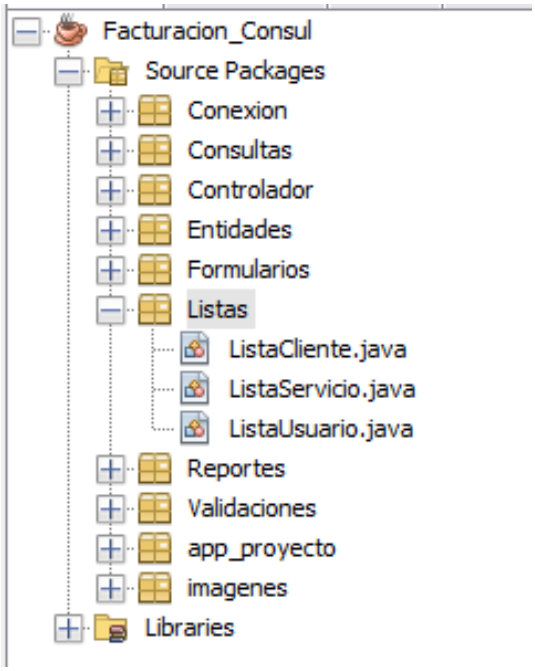


Ilustración 55: Listas

Ramos, A. (diciembre 2020).

4.5 JSP – JavaServer Pages

Es una herramienta que permite al desarrollador generar páginas web dinámicas basadas en XML y HTML, archivos PDF entre otros tipos de documentos.

Su estructura asemeja la funcionalidad de PHP pero dirigida al lenguaje de programación JAVA, se beneficia del compilador de la máquina virtual de JAVA al permitir una compilación dinámica, lo que nos permite tener un tiempo de respuesta corto al generar archivos PDF.

4.6 Esquema de Interfaces y Prototipos

Todas las interfaces gráficas del proyecto en Laravel fueron basadas en componentes de la herramienta QuickAdminPanel, la principal ventaja de esta herramienta radica en la incorporación de funcionalidades como exportación de registros tablas dinámicas a archivos CSV, PDF, HTML y un funcionamiento de activar y desactivar columnas en las tablas acorde a la voluntad del usuario.

A continuación se detalla el esquema general de esta herramienta:

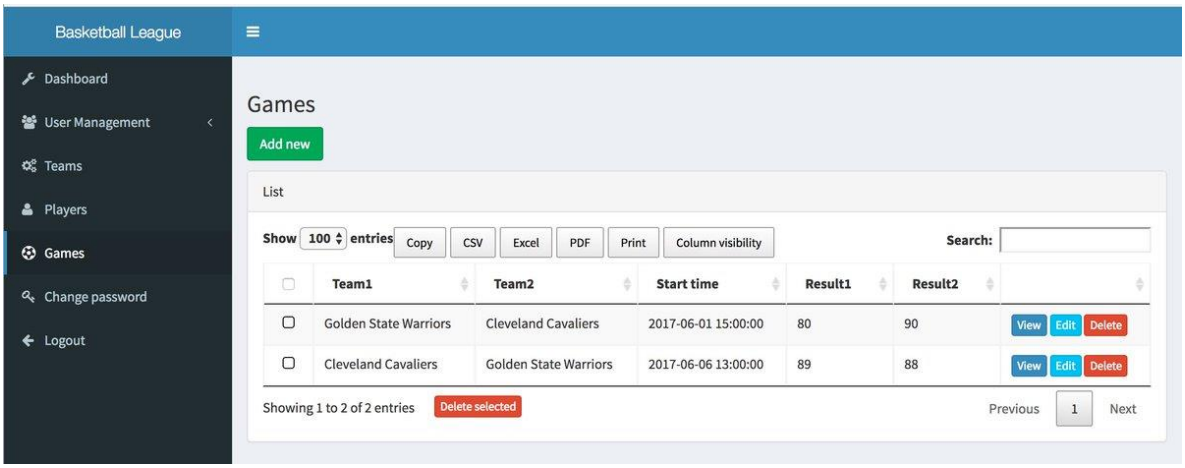


Ilustración 56: QuickAdmin Panel

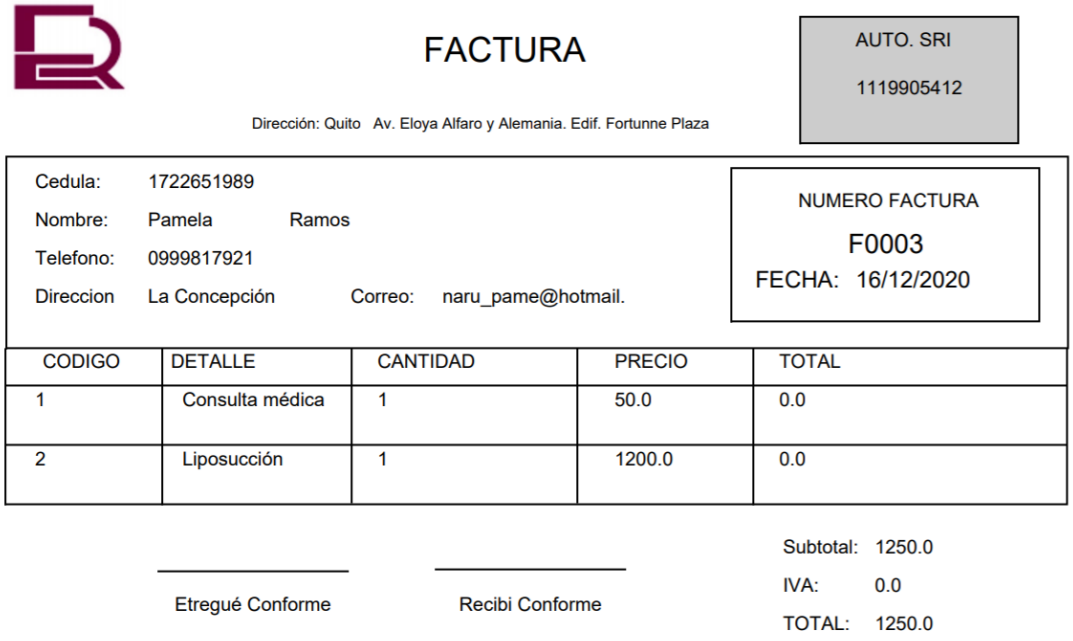
(Korop, 2020).

Debido a las prestaciones que nos permite se la ha definido como un apoyo crítico en el desarrollo de interfaces amigables para el usuario, siendo estas intuitivas en su uso y eficientes en la adopción de componentes a proyectos.

Para el prototipo de JSP generando archivos PDF se ha optado por un modelo tradicional, donde se incluirá un título, logotipo del consultorio médico y su representante legal directo, una tabla de muestra de registros y fecha del sistema.

El estilo de reportes será el mismo para servicios, clientes y cobros, mientras que el diseño de facturas será definido por el tipo de factura que se imprime en el consultorio médico.

Los prototipos son:



FACTURA

AUTO. SRI
1119905412

Dirección: Quito Av. Eloya Alfaro y Alemania. Edif. Fortunne Plaza

Cedula: 1722651989
Nombre: Pamela Ramos
Telefono: 0999817921
Direccion La Concepción Correo: naru_pame@hotmail.

NUMERO FACTURA
F0003
FECHA: 16/12/2020

CODIGO	DETALLE	CANTIDAD	PRECIO	TOTAL
1	Consulta médica	1	50.0	0.0
2	Liposucción	1	1200.0	0.0

Etregué Conforme

Recibi Conforme

Subtotal: 1250.0
IVA: 0.0
TOTAL: 1250.0

Ilustración 57: Prototipo Facturas

Ramos, A. (diciembre 2020).

REPORTE CLIENTES

miércoles 16

Cod	Cedula	Nombres		Telefono	Direccion	Correo
1	1722651989	Pamela	Ramos	0999817921	La	naru_pame@hotmail.

Ilustración 58: Prototipo Clientes

Ramos, A. (diciembre 2020).

Las interfaces de formulario del aplicativo JAVA corresponden a dos elementos, el formulario para crear, editar, eliminar y cargar información, vinculado a usuarios, clientes y servicios, en conjunto con el formulario de facturación vinculado a cargar información de clientes y servicios y la creación de factura.

Los prototipos son:

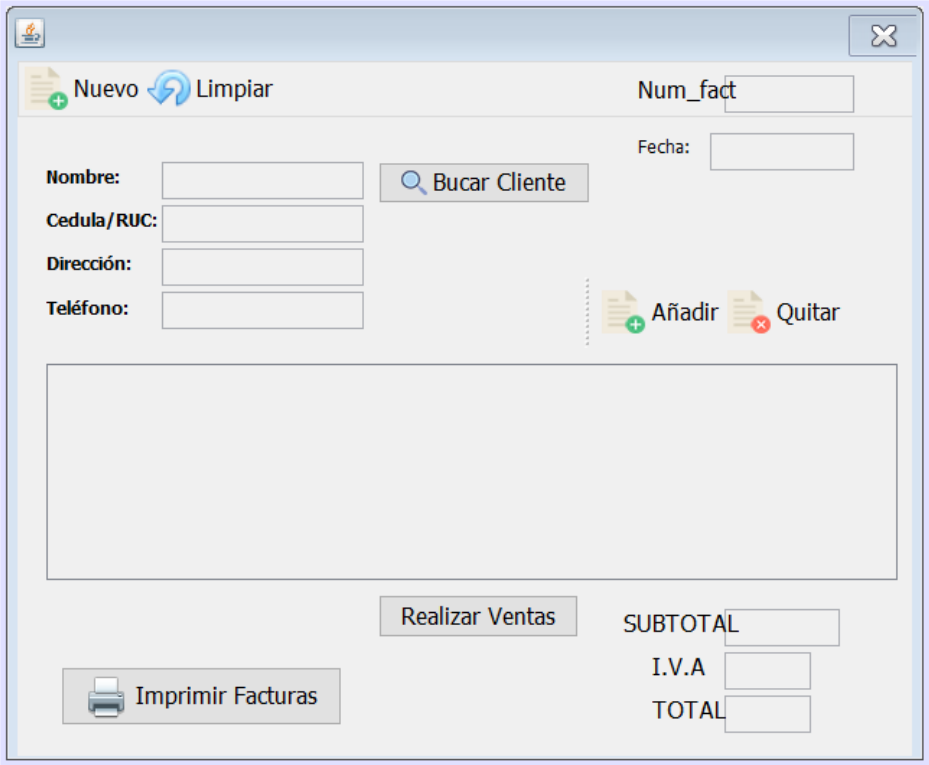


Ilustración 59: Prototipo Generar Facturas

Ramos, A. (diciembre 2020).

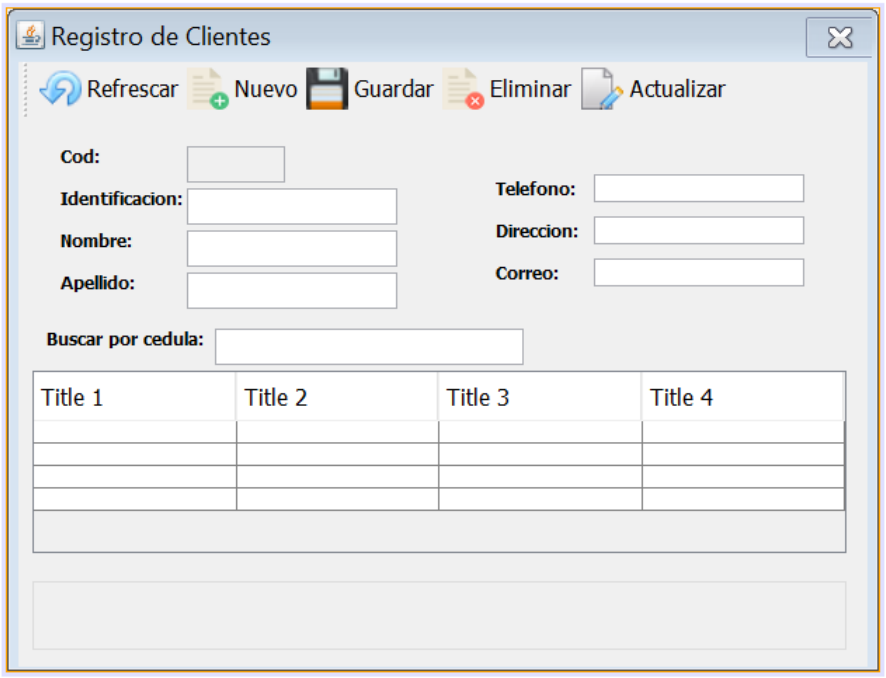


Ilustración 60: Prototipo Generar Clientes

Ramos, A. (diciembre 2020).

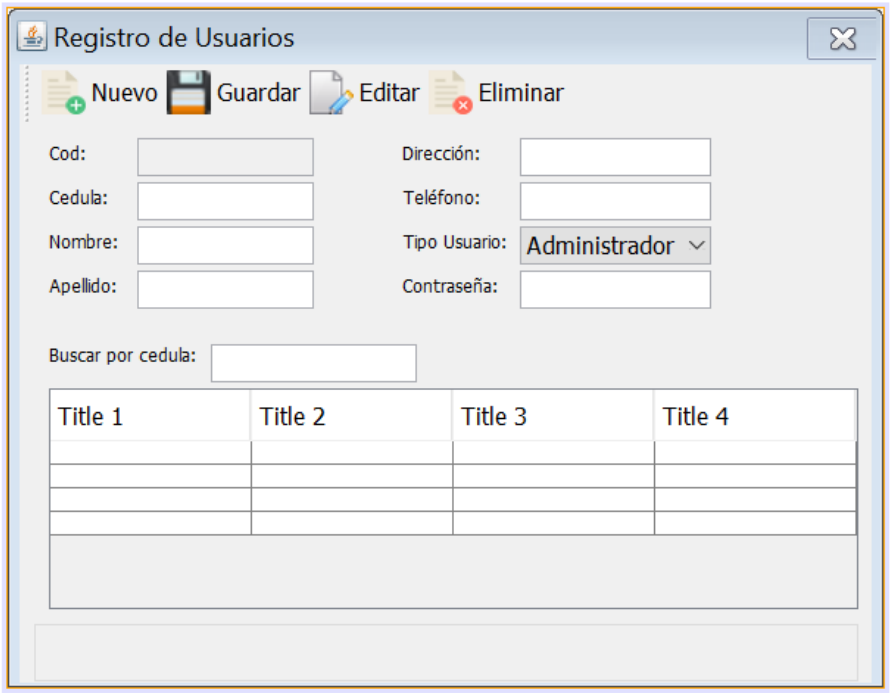


Ilustración 61: Prototipo Generar Usuarios

Ramos, A. (diciembre 2020).

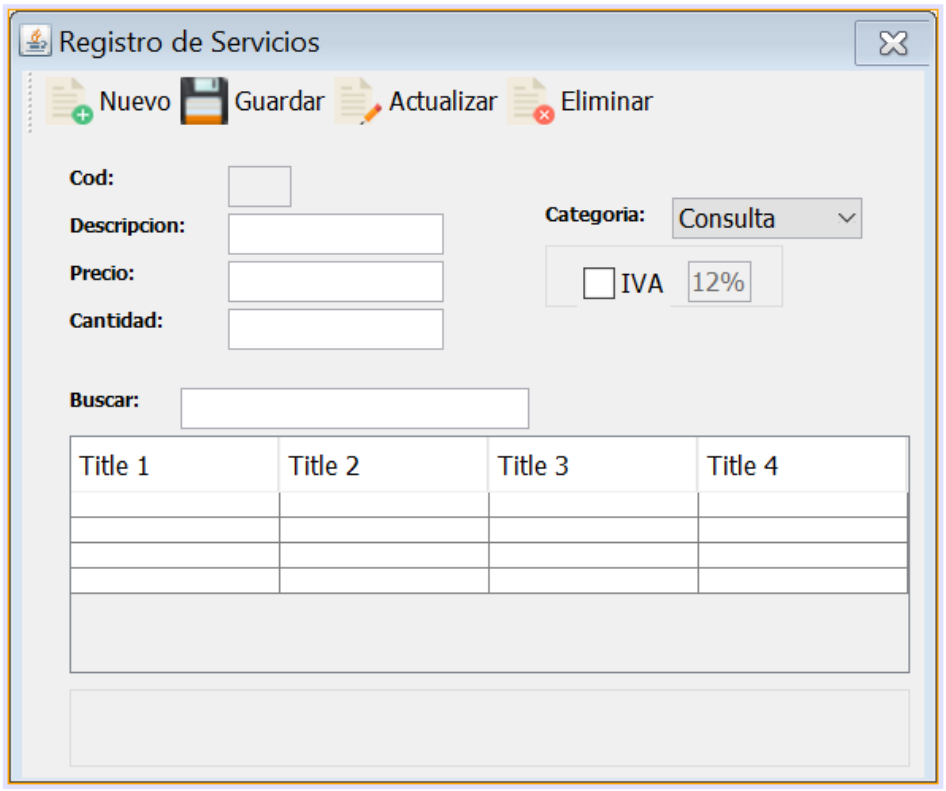


Ilustración 62: Prototipo Generar Servicios

Ramos, A. (diciembre 2020).

4.7 Rutas

Definen la manera de acceder a los componentes del aplicativo, recursos del sistema, vistas y servicios web.

Para el aplicativo Laravel, solo ha necesitado definir las rutas del sistema, de esta forma no se ha determinado la funcionalidad de servicios web, eventos de suscripción de usuarios o clientes, creación de comandos adicionales, por esto se ha mantenido en el estado por defecto al generar cualquier proyecto Laravel los archivos:

- api.php
- channels.php
- console.php

Las rutas utilizadas en el sistema se basan en solicitudes HTTP enviadas a clases dentro del controlador respectivo. De esta forma por medio de clases para crear y subir cambios, eliminar un registro, eliminar registros masivos, llevar al index o índice, mostrar registros, entre otras posibles operaciones, logramos manejar rutas simples y dinámicas.

Debido al uso de puertas de autentificación, la única ruta pública en el programa es aquella que carga y maneja el proceso de inicio de sesión.

La ubicación del archivo que contiene las rutas se encuentra en la carpeta routes.

El enlace en el aplicativo es: C:\Laravel\Sistema Administracion\routes\web.php

La distribución de las rutas puede ser apreciada en la siguiente imagen:

```php
Route::get('/', function () { return redirect('/admin/home'); });

// Rutas Autentificación
$this->get('login', 'Auth\LoginController@showLoginForm')->name('auth.login');
$this->post('login', 'Auth\LoginController@login')->name('auth.login');
$this->post('logout', 'Auth\LoginController@logout')->name('auth.logout');

// Rutas Cambio Contraseña
$this->get('change_password', 'Auth\ChangePasswordController@showChangePasswordForm')->name('auth.change_password');
$this->patch('change_password', 'Auth\ChangePasswordController@changePassword')->name('auth.change_password');

// Rutas Restablecer Contraseña
$this->get('password/reset', 'Auth\ForgotPasswordController@showLinkRequestForm')->name('auth.password.reset');
$this->post('password/email', 'Auth\ForgotPasswordController@sendResetLinkEmail')->name('auth.password.reset');
$this->get('password/reset/{token}', 'Auth\ResetPasswordController@showResetForm')->name('password.reset');
$this->post('password/reset', 'Auth\ResetPasswordController@reset')->name('auth.password.reset');

//Rutas filtros peticiones PHP
Route::group(['middleware' => ['auth'], 'prefix' => 'admin', 'as' => 'admin.'], function () {
    //Rutas HOME - Principal
    Route::get('/home', 'HomeController@index');
    //Rutas Roles
    Route::resource('roles', 'Admin\RolesController');
    Route::post('roles_mass_destroy', ['uses' => 'Admin\RolesController@massDestroy', 'as' => 'roles.mass_destroy']);
    //Rutas Usuarios
    Route::resource('users', 'Admin\UsersController');
    Route::post('users_mass_destroy', ['uses' => 'Admin\UsersController@massDestroy', 'as' => 'users.mass_destroy']);
    //Rutas Pacientes
    Route::resource('clients', 'Admin\ClientsController');
    Route::post('clients_mass_destroy', ['uses' => 'Admin\ClientsController@massDestroy', 'as' => 'clients.mass_destroy']);
    //Rutas Proveedores
    Route::resource('sellers', 'Admin\SellerController');
    Route::post('sellers_mass_destroy', ['uses' => 'Admin\SellerController@massDestroy', 'as' => 'sellers.mass_destroy']);
    //Rutas Inventario
    Route::resource('stocks', 'Admin\StockController');
    Route::post('stocks_mass_destroy', ['uses' => 'Admin\StockController@massDestroy', 'as' => 'stocks.mass_destroy']);
    //Rutas Empleados
    Route::get('get-employees', 'Admin\EmployeesController@GetEmployees');
    Route::resource('employees', 'Admin\EmployeesController');
    Route::post('employees_mass_destroy', ['uses' => 'Admin\EmployeesController@massDestroy', 'as' => 'employees.mass_destroy']);
    //Rutas Horas de Trabajo
    Route::resource('working_hours', 'Admin\WorkingHoursController');
    Route::post('working_hours_mass_destroy', ['uses' => 'Admin\WorkingHoursController@massDestroy', 'as' => 'working_hours.mass_destroy']);
    //Rutas Citas
    Route::resource('appointments', 'Admin\AppointmentsController');
    Route::post('appointments_mass_destroy', ['uses' => 'Admin\AppointmentsController@massDestroy', 'as' => 'appointments.mass_destroy']);
    //Rutas Servicios
    Route::resource('services', 'Admin\ServicesController');
    Route::post('services_mass_destroy', ['uses' => 'Admin\ServicesController@massDestroy', 'as' => 'services.mass_destroy']);
```

Ilustración 63: Rutas

Ramos, A. (diciembre 2020).

4.8 Controladores

Como se menciona previamente en el controlador se crean clases encargadas de procesar las solicitudes de diferentes operaciones. En el caso del aplicativo Laravel, los controladores se almacenan en el directorio App/Http/Controllers.

Siempre se almacenarán los diversos controladores de nuestro aplicativo en: C:\Laravel\Sistema Administracion\app\Http\Controllers, sin embargo es importante recalcar que dentro de los procesos de los controladores hay llamadas a solicitudes, manejadas como una capa adicional entre la operación y el modelo, estas serán almacenadas en: C:\Laravel\Sistema Administracion\app\Http\Requests\Admin, donde las solicitudes generadas son para almacenar y actualizar datos.

En el caso del aplicativo JAVA, los controladores generan igualmente clases enfocadas a las operaciones de creación, edición, eliminación y visualización de información, adicionalmente se crea una clase para poblar las tablas que muestran de forma amigable los registros existentes en la base de datos. Los controladores son almacenados en el paquete Controladores.

4.9 Vistas

Las vistas son el enlace con el usuario, muestran ventanas, paginas o pantallas de interfaz gráfica que permiten la interacción de un usuario con los elementos y operaciones del sistema, se encargan de enviar solicitudes al controlador para ejecutar las operaciones que este maneja. Es la capa de presentación del aplicativo.

En el aplicativo Laravel, las vistas están basadas en el motor de generación de plantillas Blade.

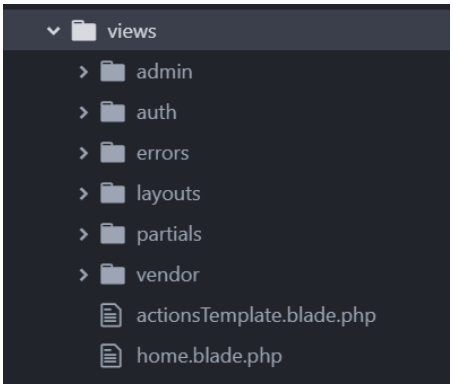


Ilustración 64: Vistas

Ramos, A. (diciembre 2020).

Podemos dividir las vistas en base a sus directorios:

1. Archivos en el directorio Views: vistas de manejo general.
    a. actionsTemplate, contiene una plantilla para los botones de edición, visualización y eliminación de elementos que estarán vinculados a las tablas dinámicas y formularios a utilizar en las demás vistas.
    b. home, contiene la primera pantalla que el usuario visualizará tras iniciar sesión, en esta podemos apreciar una nota hacia el usuario explicando puntos del sistema y un carrusel de fotos, es la pantalla de bienvenida al sistema.

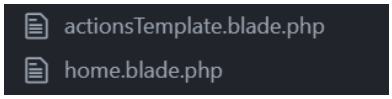


Ilustración 65: Directorio Vistas

Ramos, A. (diciembre 2020).

2. Partials: contiene elementos que conforman la distribución estética del sistema.
    a. head, contiene el nombre y diseño del icono que se visualizará en la pestaña del navegador.
    b. header, contiene la estructura del componente que se visualizará el parte superior del aplicativo en su página web.
    c. javascripts, importa los scripts que se utilizarán en el aplicativo.
    d. topbar, contiene un formato de plantilla para los títulos.
    e. sidebar, contiene el formato y distribución del menú de navegación del aplicativo, este archivo contiene los iconos de cada funcionalidad y los vincula a cada una de estas acorde el caso.

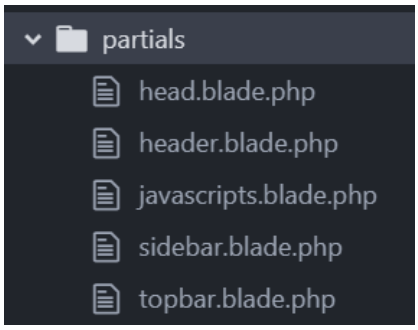


Ilustración 66: Vistas Partials

Ramos, A. (diciembre 2020).

3. errors: contiene una vista que se muestra en caso de error.
    a. 503blade, una vista que contiene un texto que cita “Volveremos Pronto”

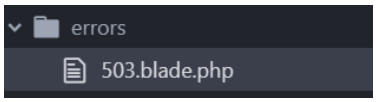


Ilustración 67: Vistas Errors

Ramos, A. (diciembre 2020).

4. auth: contiene dos directorios creados por Laravel, un directorio para restablecer contraseñas y correos, así como un directorio con una vista para redirigir a la vista de restablecer contraseña. De Igual forma contiene 3 vistas para el inicio de sesión, cambio de contraseña y registro en el sistema.
   En el sistema ocuparemos solo las vistas:
    a. login, la primera vista que carga el sistema, permite el inicio de sesión y redireccionamiento según el rol de cada usuario.
    b. change_password, una vista que se carga como parte del menú de funcionalidades tras iniciar sesión, permite cambiar la contraseña del usuario que entro al sistema, solo es posible cambiar la contraseña del usuario que inicio sesión, no de otros usuarios.

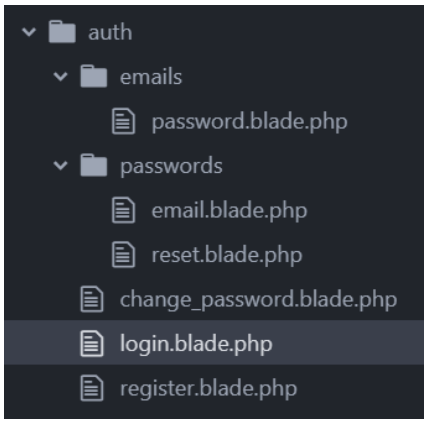


Ilustración 68: Vistas Auth

Ramos, A. (diciembre 2020).

5. admin: contiene los directorios appointments, clients, employees, roles, Sellers, services, stocks, users y working_hours, estas contienen las vistas personalizadas que permiten realizar las funciones de CRUD según la funcionalidad particular a la que hagan referencia.

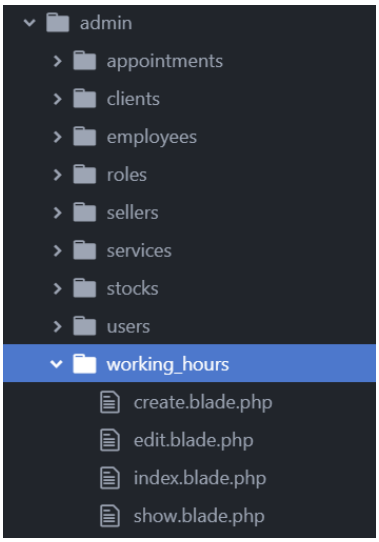


Ilustración 69: Vistas Admin

Ramos, A. (diciembre 2020).

En el aplicativo JAVA se han diseñado 2 paquetes de vistas, el primero está enfocado a las vistas típicas en un CRUD, con un formulario para gestionar un módulo o funcionalidad, mientras que el segundo paquete contiene los reportes PDF generados a través de JSF para la impresión de registros disponibles en la base de datos vinculados a clientes, servicios, cobros e impresión de facturas.

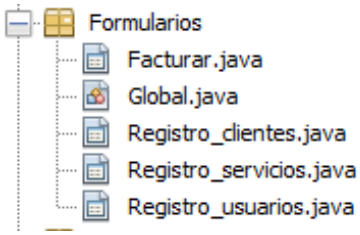


Ilustración 70: Vistas Formularios

Ramos, A. (diciembre 2020).

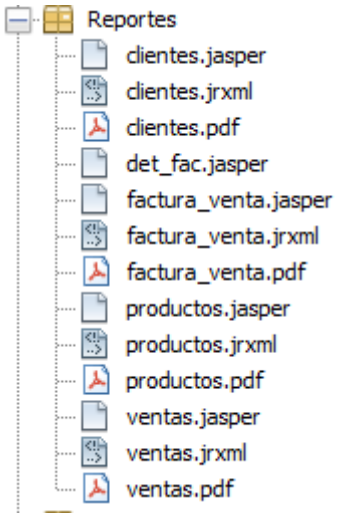


Ilustración 71: Vistas Reportes

Ramos, A. (diciembre 2020).

Capítulo 5 – Implementación y Aceptación

En este capítulo se detallan las pruebas unitarias que conducen a las pruebas de aceptación, se realizaron pruebas por cada módulo o funcionalidad en ambos aplicativos, al igual que una corrida simulando un uso cotidiano esperado por parte de los miembros del consultorio médico.

5.1 Informes Finales y Presentación del proyecto

Esta sección contiene el reporte final de entrega de cada aplicativo, presentado al momento de realizar la presentación de los mismos al beneficiario, el reporte de aceptación y la descripción de las guías de usuario y desarrollador.

5.1.1 Pruebas

Las tablas a continuación detallan las pruebas unitarias realizadas a las funcionalidades de Laravel.

Roles

Funcionalidades	Operación	Resultado	N° Pruebas
F1: Roles	Index	Carga exitosamente	1
F1: Roles	Create	Crea el registro exitosamente	2
F1: Roles	Store	Guarda los cambios con éxito	2
F1: Roles	Update	Abre el formulario con éxito	2
F1: Roles	Delete	Elimina el elemento	1
F1: Roles	Mass Delete	Borra varios registros con éxito	1

Usuarios

Funcionalidades	Operación	Resultado	N° Pruebas
F2: Usuarios	Index	Carga exitosamente	2
F2: Usuarios	Create	Crea el registro exitosamente	1
F2: Usuarios	Store	Guarda los cambios con éxito	1
F2: Usuarios	Update	Abre el formulario con éxito	1
F2: Usuarios	Delete	Elimina el elemento	1
F2: Usuarios	Mass Delete	Borra varios registros con éxito	1

Servicios

Funcionalidades	Operación	Resultado	N° Pruebas
F3: Servicios	Index	Carga exitosamente	2
F3: Servicios	Create	Crea el registro exitosamente	1
F3: Servicios	Store	Guarda los cambios con éxito	1
F3: Servicios	Update	Abre el formulario con éxito	1
F3: Servicios	Delete	Elimina el elemento	1
F3: Servicios	Mass Delete	Borra varios registros con éxito	1

Clientes-Pacientes

Funcionalidades	Operación	Resultado	N° Pruebas
F4: Pacientes	Index	Carga exitosamente	3
F4: Pacientes	Create	Crea el registro exitosamente	2
F4: Pacientes	Store	Guarda los cambios con éxito	1
F4: Pacientes	Update	Abre el formulario con éxito	1
F4: Pacientes	Delete	Elimina el elemento	2
F4: Pacientes	Mass Delete	Borra varios registros con éxito	1

Trabajadores-Empleados

Funcionalidades	Operación	Resultado	N° Pruebas
F5: Trabajadores	Index	Carga exitosamente	1
F5: Trabajadores	Create	Crea el registro exitosamente	2
F5: Trabajadores	Store	Guarda los cambios con éxito	2
F5: Trabajadores	Update	Abre el formulario con éxito	2
F5: Trabajadores	Delete	Elimina el elemento	1
F5: Trabajadores	Mass Delete	Borra varios registros con éxito	1

Horarios Laborales

Funcionalidades	Operación	Resultado	N° Pruebas
F6: Horarios Laborales	Index	Carga exitosamente	1
F6: Horarios Laborales	Create	Crea el registro exitosamente	2
F6: Horarios Laborales	Update	Abre el formulario con éxito	2

Citas

Funcionalidades	Operación	Resultado	N° Pruebas
F7: Citas	Index	Carga exitosamente	3
F7: Citas	Create	Crea el registro exitosamente	2
F7: Citas	Store	Guarda los cambios con éxito	1
F7: Citas	Update	Abre el formulario con éxito	1
F7: Citas	Delete	Elimina el elemento	2
F7: Citas	Mass Delete	Borra varios registros con éxito	1

Proveedores

Funcionalidades	Operación	Resultado	N° Pruebas
F8: Proveedores	Index	Carga exitosamente	1
F8: Proveedores	Create	Crea el registro exitosamente	1
F8: Proveedores	Store	Guarda los cambios con éxito	1
F8: Proveedores	Update	Abre el formulario con éxito	1
F8: Proveedores	Delete	Elimina el elemento	1
F8: Proveedores	Mass Delete	Borra varios registros con éxito	1

Inventario

Funcionalidades	Operación	Resultado	N° Pruebas
F9: Inventario	Index	Carga exitosamente	1
F9: Inventario	Create	Crea el registro exitosamente	2
F9: Inventario	Store	Guarda los cambios con éxito	2
F9: Inventario	Update	Abre el formulario con éxito	2
F9: Inventario	Delete	Elimina el elemento	1
F9: Inventario	Mass Delete	Borra varios registros con éxito	1

Las tablas a continuación detallan las pruebas unitarias realizadas a las funcionalidades de Java.

Clientes

Funcionalidades	Operación	Resultado	N° Pruebas
F1: Clientes	Index	Carga exitosamente	1
F1: Clientes	Create	Crea el registro exitosamente	1
F1: Clientes	Store	Guarda los cambios con éxito	1
F1: Clientes	Update	Abre el formulario con éxito	1
F1: Clientes	Delete	Elimina el elemento	1

Usuarios

Funcionalidades	Operación	Resultado	N° Pruebas
F2: Usuarios	Index	Carga exitosamente	1
F2: Usuarios	Create	Crea el registro exitosamente	1
F2: Usuarios	Store	Guarda los cambios con éxito	1
F2: Usuarios	Update	Abre el formulario con éxito	1
F2: Usuarios	Delete	Elimina el elemento	1

Servicios

Funcionalidades	Operación	Resultado	N° Pruebas
F3: Servicios	Index	Carga exitosamente	5
F3: Servicios	Create	Crea el registro exitosamente	5
F3: Servicios	Store	Guarda los cambios con éxito	3
F3: Servicios	Update	Abre el formulario con éxito	3
F3: Servicios	Delete	Elimina el elemento	2

Detalle Facturas

Funcionalidades	Operación	Resultado	N° Pruebas
F4: Detalle Facturas	Index	Carga exitosamente	2
F4: Detalle Facturas	Create	Crea el registro exitosamente	2

Facturas

Funcionalidades	Operación	Resultado	N° Pruebas
F5: Facturas	Index	Carga exitosamente	2
F5: Facturas	Create	Crea el registro exitosamente	2

5.1.2 Informe Entrega

Se presenta un informe detallando los diversos ciclos de desarrollo, siguiendo la estructura de la metodología en cascada, de esta forma se contó con una documentación que respalda ambos aplicativos, verifica se cumplan los requisitos funcionales y no funcionales de los sistemas, garantiza el cumplimiento de altos estándares de calidad y seguridad, el acta de entrega fue firmada y verificada por ambas partes involucradas, así mismo se hace entrega de los manuales de usuario y administrador.

Por último se acordó las fechas de capacitación de personal respecto a los aplicativos y su debido uso.

5.2 Aceptación

Para que el beneficiario pueda aceptar el proyecto, debe verificar los aplicativos cumplan con las funciones que el usuario deseaba recopiladas en los requerimientos no funcionales y funcionales, realizando asi pruebas de uso o aceptación por parte de su persona y los empleados o trabajadores de su consultorio médico, certificando y comprobando ambos aplicativos cumplan sus respectivos procesos de forma eficiente y sin presentar errores. Los documentos que definen la aceptación del producto se encuentran en la sección de anexos en orden ascendente conforme a la versión a la que corresponden, finalmente se tiene el documento de aceptación del proyecto, mismo que corresponde a la versión de lanzamiento de los aplicativos y cuyo propósito es certificar que todas las pruebas condujeron a resultados satisfactorios.

5.3 Manual de Usuario/ Guía de Usuario

Se debe presentar un manual de usuario simple o final, el cual detalla cómo utilizar los aplicativos desde el lado de un usuario simple asi como una guía de manejo simple para el tipo de usuario administrador. Estas guías contienen una serie de pasos e imágenes que brindan un soporte o apoyo a los usuarios en caso de que no recuerden como realizar alguna acción en el sistema o necesiten de guía para poder manejar funcionalidades.

Es asi, que se considera un manual de apoyo enfocado a descripciones de funcionamiento y no un manual técnico.

5.4 Manual de Desarrollador

Es un manual técnico a diferencia del manual de usuario, este documenta como realizar cambios en el sistema, su estructura la forma de dar mantenimiento para los aplicativos JAVA y Laravel, es asi pues un respaldo para futuros desarrolladores, capaz de describir de forma sencilla y útil como está estructurado el sistema que cambios no involucran modificaciones mayores y cuales si, como realizar estas y el funcionamiento esperado de cada funcionalidad.

Capítulo 6 – Conclusiones y Recomendaciones

En este capítulo se detalla el cumplimiento de los objetivos del proyecto reflejados en la documentación presentada, los aplicativos aprobados, la implementación de los sistemas y la capacitación al personal del consultorio médico.

Tanto las conclusiones como las recomendaciones deben de estar sujetas al alcance del proyecto, de esta forma todo el proyecto se verá alineado al esfuerzo de cumplir los requisitos solicitados por el beneficiario, así como

6.1 Conclusiones

- Se completó con éxito la especificación de los requerimientos para el desarrollo e implementación de los aplicativos solicitados por parte del consultorio de cirugía plástica del Dr. Edison Ramos.
- Se crearon satisfactoriamente los módulos para gestión de pacientes, clientes, trabajadores, usuarios, roles, citas, horarios de trabajo, proveedores, inventario y facturas que corresponden a los aplicativos Laravel y JAVA.
- Se realizaron las pruebas unitarias y de aceptación de manera exitosa.
- Se desarrollaron los aplicativos cumpliendo un alto estándar de seguridad, apegados al criterio de un uso sencillo e intuitivo por medio de sus interfaces.
- Se implemento de forma exitosa los aplicativos generados, manejados desde el servidor y ordenador principal ubicado en el consultorio médico.
- Se logró utilizar los conceptos MVC en el la construcción de los módulos o funcionalidades de los aplicativos.
- Se logró cubrir los objetivos del proyecto utilizando las herramientas detalladas en el marco teórico del proyecto de disertación.
- Debido al cambio de un aplicativo de facturación en línea a uno de escritorio no fue necesario el utilizar librerías faces.

6.2 Recomendaciones

- Mantener el diseño aplicado al aplicativo Laravel, esto garantiza una fuerte seguridad y dada su estructura nos permite una escalabilidad de funciones e implementación de nuevos módulos de forma sencilla y eficiente.
- Considerar migrar el sistema totalmente a la web, es decir incluir un módulo de facturación en el aplicativo Laravel, sustituyendo el aplicativo JAVA.
- Las herramientas utilizadas se pueden aplicadas en diferentes proyectos de desarrollo de aplicaciones o programas dada su adaptabilidad y escalabilidad.

Referencias

Alazraqui, M. (2006). Sistemas de Información en Salud: de sistemas cerrados a la ciudadanía social. Un desafío en la reducción de desigualdades en la gestión local. *Scielo Publi Health*.

Datos, F. d. (2002). *A. Silberschatz, H. Korth, S. Sudarshan.* Bombay: McGraw Hill.

Guérin, B.-A. (2015). *Gestión de proyectos informáticos.* Eni Ediciones.

Hernandez, U. (22 de 05 de 2015). MVC (Model, View, Controller) Explicado.

JH Canós, M. L. (2012). *Metodologías ágiles en el desarrollo de software.* Las Tunas, Cuba: Universidad de las Tunas.

K. Arnold, J. G. (2005). *THE Java™ Programming Language.* Addison Wesley Professional.

Korop, P. (7 de Marzo de 2020). Teams Multi-Tenancy: Add “Team Admin” to Manage Users.

Pressman, R. S. (2010). *Ingeniería del software Un enfoque práctico.* Connecticut: McGraw Hill.

Soriano, L. C. (Diciembre de 2016). Requerimientos del software. Objetivos Introducir los conceptos de requerimientos del usuario y sistema Describir los requerimientos funcionales y no.

Vasconcellos-Silva, C. &. (2005). Sociedad de la información, comunicación y salud: actualidades y proyecciones. *Questión; vol. 1, no. 20*.

Anexos

Esta sección contendrá los documentos enlazados que fueron presentados junto con el último ciclo de desarrollo, cartas de aceptación y manuales de usuario y desarrollador.

Anexo 1: Documento de Aceptación y Actas de Reuniones

https://drive.google.com/file/d/1e9rNR6ZfmKMKc2FGNQC_S4-DKO3DI1-w/view?usp=sharing

Anexo 2: Guía – Manual de Desarrollador o Administrador

https://drive.google.com/file/d/10mBAZEeEgr7buzmBnHAcya7nnnywCSeu/view?usp=sharing

Anexo 3: Guía – Manual de Usuario

https://drive.google.com/file/d/19dgKcaa74rzO9EGC9_LeSwNmtFCUTMgI/view?usp=sharing