UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL

CARRERA DE TEGNOLOGÍAS DE LA INFORMACIÓN

TEMA:

SISTEMA WEB MULTIPLATAFORMA BASADO EN LA ARQUITECTURA MODELO VISTA CONTROLADOR (MVC) Y CLOUD COMPUTING PARA LA GESTIÓN DE ACTIVIDADES DE NOVADENT-ESPECIALIDADES ODONTOLÓGICAS EN LA CIUDAD DE LATACUNGA

Trabajo de Integración Curricular, Modalidad: Proyecto de Investigación, presentado previo a la obtención del título de Ingeniero en Tecnologías de la Información.

ÁREA: Software

LÍNEA DE INVESTIGACIÓN: Desarrollo de software

AUTOR: Edwin Esteban Chanatasig Maigua

TUTOR: Ing. Oscar Fernando Ibarra Torres

Ambato - Ecuador

septiembre - 2022

APROBACIÓN DEL TUTOR

En calidad de tutor del Trabajo de Integración Curricular con el tema: SISTEMA WEB MULTIPLATAFORMA BASADO EN LA ARQUITECTURA MODELO VISTA CONTROLADOR (MVC) Y CLOUD COMPUTING PARA LA GESTIÓN DE ACTIVIDADES DE NOVADENT-ESPECIALIDADES ODONTOLÓGICAS EN LA CIUDAD DE LATACUNGA, desarrollado bajo la modalidad Proyecto de Investigación realizado por el señor Edwin Esteban Chanatasig Maigua estudiante de la Carrera de Tecnologías de la Información, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que el estudiante ha sido tutorado durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 17 del Reglamento para la ejecución de la Unidad de Integración Curricular y la obtención del título de tercer nivel, de grado en la Universidad Técnica de Ambato y sus reformas y el numeral 7.4 del respectivo instructivo.

Ambato, septiembre 2022

Firmado electrónicamente por:
OSCAR FERNANDO IBARRA TORRES

Ing. Oscar Fernando Ibarra Torres

TUTOR

AUTORÍA

El presente trabajo de Integración Curricular titulado: SISTEMA WEB MULTIPLATAFORMA BASADO EN LA ARQUITECTURA MODELO VISTA CONTROLADOR (MVC) Y CLOUD COMPUTING PARA LA GESTIÓN DE ACTIVIDADES DE NOVADENT-ESPECIALIDADES ODONTOLÓGICAS EN LA CIUDAD DE LATACUNGA es absolutamente original, auténtico y personal. En tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, septiembre 2022

Edwin Esteban Chanatasig Maigua

C.C 0503907065

AUTOR

APROBACIÓN TRIBUNAL DE GRADO

En calidad de par calificador del Informe Final del Trabajo de Integración Curricular presentado por el señor Edwin Esteban Chanatasig Maigua, estudiante de la Carrera de Tecnologías de la Información, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Proyecto de Investigación, titulado SISTEMA WEB MULTIPLATAFORMA BASADO EN LA ARQUITECTURA MODELO VISTA CONTROLADOR (MVC) Y CLOUD COMPUTING PARA LA GESTIÓN DE ACTIVIDADES DE NOVADENT-ESPECIALIDADES ODONTOLÓGICAS EN LA CIUDAD DE LATACUNGA, nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 19 del Reglamento para la ejecución de la Unidad de Integración Curricular y la obtención del título de tercer nivel, de grado en la Universidad Técnica de Ambato y sus reformas y al numeral 7.6 del respectivo instructivo. Para cuya constancia suscribimos, conjuntamente con la señora Presidente del Tribunal.

Ambato, septiembre 2022

Firmado electrónicamente por:
ELSA PILAR URRUTIA

Ing. Pilar Urrutia, Mg.
PRESIDENTE DEL TRIBUNAL

Firmado electrónicamente por:
EDISON HOMERO ALVAREZ MAYORGA

Ing. Edison Homero Alvarez Mayorga
PROFESOR CALIFICADOR

Firmado electrónicamente por:
FRANKLIN OSWALDO MAYORGA MAYORGA

Ing. Franklin Oswaldo Mayorga Mayorga
PROFESOR CALIFICADOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que haga uso de este Trabajo de Integración Curricular como un documento disponible para la lectura, consulta y procesos de investigación.

Cedo los derechos de mi Trabajo de Integración Curricular en favor de la Universidad Técnica de Ambato, con fines de difusión pública. Además, autorizo su reproducción total o parcial dentro de las regulaciones de la institución.

Ambato, septiembre 2022

Edwin Esteban Chanatasig Maigua

C.C 0503907065

AUTOR

DEDICATORIA

El presente proyecto está dedicado a mis padres María Dolores y José Ascencio por estar presentes en cada etapa de mi crecimiento como profesional y siempre brindame su apoyo en momentos complicados y ensenándome que en la vida la perseverancia es el camino del éxito.

A mis abuelitos que desde el cielo me protegen que siempre están presentes en mi mente y en cada decisión tomada a lo largo de la carrera. A mis hermanos que siempre estuvieron apoyándome y brindándome su apoyo incondicional.

A todos mis amigos quienes siempre estuvieron presentes en cada etapa de mi vida universitaria brindándome su apoyo; en especial a mi amigo Christopher Sánchez que desde el cielo brilla con luz propia.

Edwin Esteban Chanatasig Maigua

AGRADECIMIENTO

Agradezco a Dios por brindarme salud y sabiduría necesaria durante todo este proceso de crecimiento personal y profesional permitiéndome alcanzar mis objetivos.

A mis padres y hermanos que siempre brindaron su apoyo incondicional en cada paso que daba y nunca permitieron que desmayara.

A mis amigos que me acompañaron durante todos estos años y siempre me brindaron muy buenos momentos de alegría y excelentes recuerdos. Muchas gracias!

Al Centro Odontológico NovaDent-Especialidades Odontológicas por brindarme la confianza de poder desarrollar mi proyecto de investigación y además siempre mantener las puertas abiertas ante cualquier situación.

A mi tutor, Ing. Oscar Fernando Ibarra Torres por brindarme su apoyo y orientación durante todo el proceso del trabajo de investigación.

Edwin Esteban Chanatasig Maigua

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN EJECUTIVO

El impacto del desarrollo tecnológico ha sido innegable en estos últimos años teniendo grandes incidencias en distintos ámbitos ya sean empresariales o sociales; las instituciones que brindan servicios de salud no son una excepción debido que en su mayoría hacen uso de herramientas tecnológicas y sistemas informáticos que ayudan a la gestión de actividades que estas deben cumplir.

La implantación de un sistema web multiplataforma es el objetivo principal del presente proyecto de investigación, el aplicativo cuenta con una serie de funciones que ayudan a gestionar actividades como: registro de citas, filtro de horarios disponibles, envío de mensajes de WhatsApp de recordatorios de citas asignadas, además, de la administración de datos e historial clínico de los pacientes que anteriormente se las realizaba de forma manual.

El sistema web se desarrolló con un entorno de ejecución en tiempo real basado en JavaScript llamado node.js, el mismo que permitió implementar la arquitectura Modelo Vista Controlador (MVC) y que además provee un ambiente multiplataforma. El gestor de base de datos utilizado fue MariaDB que garantiza la seguridad y accesibilidad a los datos al mismo tiempo que protege la integridad de estos, la implantación se lo realizó en un servidor en la nube que cuenta con sistema operativo Linux específicamente la distro Alma Linux bajo el dominio novadentsalud.com. Para el control de cumplimientos de proceso del desarrollo de la aplicación se aplicó la metodología Extreme Programming (XP) para una mejor gestión de actividades.

Palabras clave: Sistema web, Node.js, Metodología XP, JavaScript, Arquitectura MVC.

ABSTRACT

The impact of technological development has been undeniable in recent years, having a great impact on different areas, whether business or social; institutions that provide health services are no exception, as most of them make use of technological tools and computer systems that help them to manage the activities they must carry out.

The implementation of a multi-platform web system is the main objective of this research project. The application has a series of functions that help to manage activities such as: registration of appointments, filtering of available schedules, sending WhatsApp' messages reminders of assigned appointments, as well as the administration of data and clinical history of patients that were previously done manually.

The web system was developed with a real-time execution environment based on JavaScript called node.js, which allowed the implementation of the Model View Controller (MVC) architecture and provides a cross-platform environment. The database manager used was MariaDB which guarantees security and accessibility to data while protecting the integrity of these, the implementation was done on a server in the cloud that has Linux operating system specifically the Alma Linux distro under the domain novadentsalud.com. The Extreme Programming (XP) methodology was applied to control the application development process for better management of activities.

Keywords: Web system, Node.js, XP methodology, JavaScript, MVC Architecture.

CAPÍTULO I.- MARCO TEÓRICO

1.1 Tema de Investigación

SISTEMA WEB MULTIPLATAFORMA BASADO EN LA ARQUITECTURA MODELO VISTA CONTROLADOR (MVC) Y CLOUD COMPUTING PARA LA GESTIÓN DE ACTIVIDADES DE NOVADENT-ESPECIALIDADES ODONTOLÓGICAS EN LA CIUDAD DE LATACUNGA.

1.1.1 Planteamiento del Problema

Las tecnologías de la información en Latinoamérica han tenido un significativo impacto en los últimos 30 años, estos han contribuido al desarrollo del sector industrial, económico y social, según datos obtenidos del análisis de inversión en tecnologías de la información se concluyó que un alto porcentaje de países latinos optan por seguir las vías de innovación y desarrollo en sus industrias [1]. Las tecnologías de la información es el protagonista del cambio de la forma de trabajo de diversas organizaciones, estas, en la actualidad optan por optimizar los procesos que se llevan a cabo en sus instalaciones mediante soluciones informáticas, sistemas web o aplicaciones, gracias a ello se minimiza el trabajo humano, aumentando la productividad, efectividad y estimulando la competitividad entre sus semejantes; el objetivo de estos cambios es llegar a un constante desarrollo y mejor control de las actividades que se realizan a diario. Cabe agregar que las soluciones anteriormente mencionadas deben otorgar usabilidad que es considerada como un atributo que mide la calidad de los sistemas y la facilidad de uso de estos [2]. Por antes mencionado se deduce que un sistema no solo debe cumplir con las funciones para la cual fue desarrollo sino que tenga facilidad de interacción con el usuario.

Dentro del entorno nacional varias empresas e industrias que ofrecen servicios en el Ecuador, el bajo índice de interacción con la tecnología o el desconocimiento de uso

es un causante de que sus actividades diarias se vuelvan obsoletas y anticuadas, muchas de las empresas eligen seguir operando de manera tradicional realizando su trabajo de forma manual o con softwares básicos que no representan una ayuda significativa, como consecuencia de esto existe un deficiente control en la gestión y control de las actividades dentro de las organizaciones. Esto en un futuro supone un estancamiento y pérdida de confiabilidad de la empresa ante los consumidores por falta de originalidad. La innovación que tiene base en principios científicos y se centra en la búsqueda de nuevas ideas y aplicaciones [3], es una salida óptima para evitar tal estancamiento.

En el caso de la ciudad de Latacunga, una ciudad que en la actualidad es la cuna de múltiples industrias en vías de desarrollo, los sistemas informáticos son una opción de mejora e innovación reduciendo los errores que en ocasiones son provocados de forma accidental por los trabajadores u operarios. En el centro de especialidades odontológicas existe una deficiente organización de trabajo puesto que únicamente hacen uso de herramientas básicas para el control de citas y otras actividades, para ejemplificar la problemática el odontólogo responsable de la revisión del paciente para encontrar el historial clínico utiliza apuntes plasmados en hojas de papel lo que genera una acumulación excesiva de documentos y como consecuencia de esto la pérdida de tiempo tanto del paciente como del dentista.

1.2 Antecedentes Investigativos

Basado en la revisión que se realizó en proyectos de investigación de distintas universidades del país se encontró los siguientes antecedentes:

Según José Bernabe Vera Marin [4], en su tesis “Plataforma Como Servicio (PaaS) Para La creación, desarrollo y despliegue de aplicaciones web en la Facultad de Ingeniería en Sistemas, Electrónica e Industrial”, trabajo realizado como tesis en la Universidad Técnica de Ambato en el año 2018, se determinó:

- La utilización del servicio PaaS permitió una reducción de costos en lo que se refiere a hardware y software además de optimizar el tiempo de configuración, los aplicativos desarrollados en este modelo de servicio tienen mayor facilidad de compilación, implementación y administración.

Según David André Villamil Carrillo [5], en su tesis "Diseño de un sistema de pedidos online a través de una infraestructura de Cloud Computing", trabajo realizado como tesis en la Universidad de las Américas en el año 2017, se determinó:

- Cloud Computing ofrece varios modelos de servicios Infraestructure as a Service (IaaS), Platform as a Service (PaaS) y Software as a Service (SaaS), dependiendo de la necesidad que se tiene se debe optar por uno de los servicios, cada modelo ofrece distintos servicios siendo el PaaS quien ofrece servidores web, base de datos y entornos de desarrollo de software que son recursos necesarios para el desarrollo de aplicativos.
- Todos los recursos generados para del sistema de pedidos se alojaron en los servidores de los proveedores de cloud computing haciendo que la implementación se haya realizado de manera sencilla.

Según Diego Armando Alvarado Napa y Jefferson Clíder Guillén Valenzuela [6], en su tesis "Desarrollo de una Aplicación Web de Gestión y Control del Historial Clínico en el Departamento Médico de la Espam Mfl", trabajo realizado como tesis en la Escuela Superior Politécnica Agropecuaria de Manabí "Manuel Félix López" en el año 2018, se determinó:

- El uso de la arquitectura MVC benefició a la estructuración del sistema web separando las responsabilidades (accesibilidad a los datos, interfaz de usuario y la lógica de control) haciendo que al aplicativo sea más eficiente y propenso a escalabilidad en un futuro.

- Las actividades que antes se las registraba en papel se la recopiló como información digital evitando riesgos de pérdida de datos en ambientes físicos, además el control de registro de las citas y visitas a los departamentos de atención se volvió ágil y segura derivando en la satisfacción de los pacientes.

Según Eduardo Javier Gonzales Tumbaco [7], en su tesis "Implementar un sistema web para gestión clínica dental, aplicando tecnologías open source: caso Consultorio Odontológico Navarro", trabajo realizado como tesis en la Universidad Estatal Península de Santa Elena en el año 2017, se determinó:

- Como instrumento para la recolección de datos se usó la Entrevista, misma que se formó con 5 interrogantes dirigida a los odontólogos especialistas que forman parte del cuerpo de trabajo de la clínica dental "Navarro" y al personal administrativo.
- Se obtuvo una reducción en el tiempo tanto en la recepción de datos de los pacientes que son nuevos en el centro odontológico como en la búsqueda de información del historial clínico de los pacientes que frecuentan la clínica dental.
- La administración de las actividades realizadas por los odontólogos se volvió más eficiente, además de disponer de información confiable y consistente.

Según Ricardo Alfonso Jota Quezada y Anthony Estuard Mosquera Romero [8], en su tesis "Desarrollo e implementación de aplicación web para gestión de historias clínicas de los pacientes del consultorio dental Odonto Candy", trabajo realizado como tesis en la Universidad Politécnica Salesiana sede Guayaquil en el año 2021, se determinó:

- Se obtuvo una mejora en el manejo de información de los pacientes a través de un sistema web que está desarrollado con tecnologías que proporcionan seguridad y eficiencia en sus funcionalidades agilitando los procesos de forma

idónea, cada campo del aplicativo cuenta con atributos que evitan el error de digitación obteniendo reportes con datos confiables.

- El uso de Eloquent que es un ORM (Object-Relational Mapping) perteneciente a Laravel ayudó a los procesos ejecutados sobre la base datos del proyecto, a través de la traducción de las consultas Structured Query Language (SQL) a un patrón MVC lo que facilita a la ejecución de peticiones SQL de formar directa evitando la inyección SQL.

1.3 Fundamentación Teórica

1.3.1 Ingeniería web

Es una especialización de la Ingeniería de Software (IS) pero que su objetivo está centrado en el uso de tecnologías web para el desarrollo de software de alta calidad, no es una ciencia que se la pueda catalogar como nueva, sino que sientas sus bases en las técnicas más útiles de IS direccionados para su uso en aplicativos web [9].

La ingeniería web se basa en el uso y establecimiento de principios científicos, ingeniería con enfoques ordenados y sistemáticos para el desarrollo de sistemas web que brinden usabilidad a los usuarios. Incorpora ciertos principios que son propios de la Ingeniería en Software pero con una adaptación flexible a los aplicativos web [10].

1.3.2 Tecnologías Web

A medida que se van creando más dispositivos para el intercambio y distribución de la información a través del acceso a internet para los usuarios finales pero, se presenta la problemática de como mostrar el contenido de los aplicativos a los consumidores de forma que este no afecte a la experiencia del usuario independientemente del dispositivo que esté usando. La experiencia de usuario no se basa únicamente a la

recepción de información del contenido sino también intervienen otros factores como la rapidez de la carga de información y la fluidez de navegación [11].

Con el desarrollo de nuevas tecnologías que se han ido incorporando en el desarrollo se han ido reemplazando las tecnologías clásicas ocasionando una continua mejora en los aplicativos web, muchas de estas tecnologías se introducen en dos aspectos del desarrollo web.

Tecnologías Frontend

Frontend es la representación visual de la interfaz de usuario, el diseño de la estructura, donde se muestra los medios audiovisuales, tipografía, banners teniendo como objetivo principal lograr un diseño atractivo e intuitivo, las tecnologías que este incorpora están básicamente orientadas al diseño de interfaces las cuales son capaces de estructurar el contenido de las plataformas, proporcionar estilos y programar el comportamiento de los elementos, a esto se suman otras tecnologías como frameworks y librerías, que ayudan a la compatibilidad con los navegadores y a tener un código más legible [12].

Tecnologías Backend

Es la parte lógica del aplicativo web y la cual el usuario no puede visualizar, encargado de procesar la información generada del lado del cliente a través del Frontend, implementa el comportamiento del aplicativo web en el servidor [12].

La tecnologías Backend son la parte funcional del aplicativo web pues ayudan a la operabilidad del sistema, realizan la conexión con la base de datos, permiten realizar operaciones sobre los archivos del servidor, básicamente es el lenguaje en el cual se desarrolla el sistema teniendo múltiples opciones a elección. Otras tecnologías del lado del servidor son los frameworks que favorecen al control de la seguridad de la plataforma adema de mantener el código de manera ordenada [12].

1.3.3 Cloud Computing

Es una arquitectura que permite el acceso remoto a tecnologías de la información y recursos informáticos configurables (servicios, aplicaciones, redes, almacenamiento, servidores) a través de la red, estos recursos son reservados de manera rápida y eficiente sin mucho esfuerzo por parte del proveedor. Su modelo de trabajo se resume en pago por uso [13].

1.3.3.1 Modelo de servicio de cloud computing

Software as a Service (SaaS)

Es el servicio más común, una organización adquiere este tipo de modelo de servicio cuando su objetivo es desarrollar sus propias aplicaciones y alojarlas en un servido evitando la relación con terceros, como ventajas principales se tiene la gestión y pago por las licencias de software, hosting , interfaces web, etc. Como las soluciones más comunes de este modelo de servicio tenemos alojamiento web, Comercio Electrónico, Colaboración Online, entre otros [14].

Platform as a Service (PaaS)

Este modelo se refiere a poseer una plataforma que es usada bajo demanda, es un servicio que por lo general posee base de datos, servicios que posibilita el desarrollo de aplicaciones probarlas e implementarlas [14].

En referencia al desarrollo d software , un consumidor de este servicio tiene la posibilidad de desarrollar aplicativos sin la necesidad de comprar un entorno de desarrollo o editores de texto y sin tener que configurar hardware [14].

Infraestructure as a Service (IaaS)

Es un modelo de servicio que posibilita a los usuarios el rentar hardware virtual o físico como pueden ser la conexión a la red, servidores e incluso almacenamiento, este modelo se caracteriza también en el escalado dinámico y virtualización de escritorio [14].

1.3.4 Arquitectura MVC

Es un patrón de diseño de software que separa la capa de lógica de la interfaz gráfica basándose en el concepto de 3 capas , modelo, vista y controlador [15].

Modelo

Contiene la lógica del negocio, es la capa que interactúa directamente con los datos, contiene los mecanismos de acceso a los datos [15].

Vista

Se refiere a la interfaz con que el usuario final interactúa, se puede tener múltiples vistas que represente un mismo modelo de datos [15].

Controlador

Es el intermediario entre la Vista y el Modelo, captura las peticiones desde la interfaz, procesas la información y realiza la solitud de los datos [15].

1.3.5 Gestión de calidad

Se refiere a la coordinación de actividades , una correcta dirección y gobierno, mediante la gestión de calidad se verifica la garantía de los servicios que son ofertados a los solicitantes a través de implantación de programas y controlando los resultados con proyección a una mejora continua [16].

La gestión de calidad tiene como objetivo mejorar la competitividad de la empresa haciendo uso de los recursos que esta posea ya sean humanos, informáticos, etc. Para la gestión de la calidad es imprescindible la información de todos los niveles con el fin de detectar anomalías o irregularidades que afecten al desempeño de los procesos, de las actividades y que deriven en la insatisfacción del cliente [16].

1.3.6 Sistemas de información

Los sistemas de información son factores clave para una empresa, esta, puede tener una estructura sólida y una directriz eficiente pero si carece de sistemas de información no se podrá constatar el desempeño de las actividades que se realizan en la empresa por tanto no será posible un cambio o mejoras en aquellas que no estén cumpliendo con el objetivo o una toma de decisiones [17].

Una empresa puede poseer páginas web, comprar dispositivos computacionales o incluso desenvolverse el ámbito de negocios electrónicos, pero por lo anterior no se concluye que cuenta con un sistema de información pues no se tienen en cuenta únicamente estas herramientas sino también la organización de estas para generar información que puede repercutir en la toma de medidas, decisiones para mejorar el rendimiento de la organización [17].

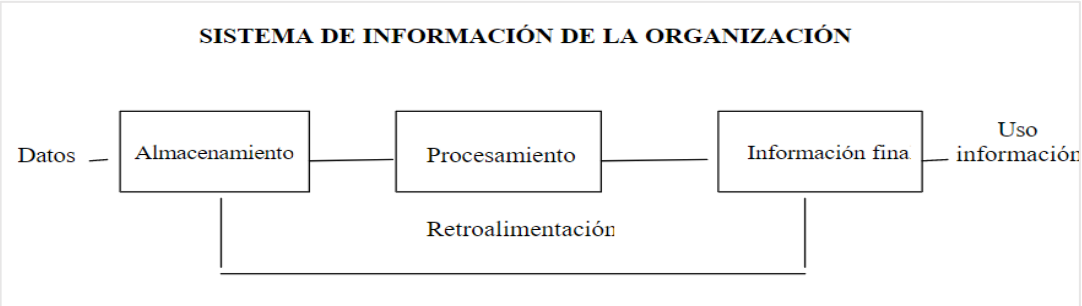


Figura 1.1: Sistema de Información de la Organización Empresarial

Fuente:[17]

1.3.7 Gestión de la Información

Se basa en la manipulación de datos, en su recopilación, seguridad, disponibilidad e integridad, además de su utilización de manera eficiente sin exceder el límite de las normas y políticas, para que se tomen decisiones y medidas que beneficien a la empresa [18].

Es el proceso en donde se hace uso de recursos básicos para el manejo de la información ya sean humanos, informáticos, económicos, etc. A dicha información se la denomina recursos de recursos puesto que al estar controlada por una herramienta hace que se optimice y mejore su desempeño [19].

1.3.8 Metodologías ágiles

Son metodologías ligeras o adaptativas que prestan mejor flexibilidad y agilidad en el desarrollo de proyectos adecuándose a las necesidades de los clientes sin perder el enfoque de obtener mejores resultados. Divide el proyecto en pequeñas partes que pueden resolverse en poco tiempo, pero manteniendo un control estricto sobre estas actividades de desarrollo de software, además, evitan la entrega de documentación excesiva y optan por mostrar mayor interacción con personas que con procesos [20].

1.3.8.1 Scrum

Esta metodología se caracteriza por estar direccionada al trabajo grupal en la cual no existe el término "Jefe de proyecto" y que está construida en base a tres roles:

1. **Product Owner. –** Conocido también como PO y es quien idealiza el producto. Encargado de elaborar las User Stories a través de la redacción de documentos [20].
2. **Scrum Master. –** Es quien facilita el trabajo a los miembros del equipo puesto que se encarga de la eliminación de obstáculos que se presenten durante el desarrollo del proyecto garantizando la efectividad de la aplicación de la metodología [20].
3. **Equipo de Desarrollo. –** Tienen funciones distintas como lo son el diseño, arquitectura, sistemas web, desarrollador, etc. El principal objetivo es el desarrollar las User Stories entregadas por el PO garantizando que los entregables sean de calidad [20].

Los tres pilares o estrategias que deben implementarse en la metodología para la obtención de mejores resultados son los siguientes:

1. **Transparencia.-** Cada miembro del equipo de trabajo debe estar en igual de condiciones con respecto a la información acerca del proyecto y conocer el estado actualmente, esto se consigue a través de la aplicación de un lenguaje común [20].
2. **Inspección.-** Continuamente el equipo de trabajo debe inspeccionar lo que produce y el estado en el que se encuentra, es decir, la finalidad es que la aplicación de la metodología fluya de manera organizada [20]
3. **Adaptación.-** Es un pilar muy importante puesto que el equipo debe ajustarse fácilmente a los nuevos lineamientos sin mayor contrariedad [20].

Eventos o ceremoniales

1. **Sprint.-** Es cada una de las iteraciones que tendrá el proyecto en un período de tiempo máximo de un mes cuyo objetivo es incrementar el valor del producto que se desarrolla [20].
2. **Reunión de planificación del Sprint.-** es el periodo de tiempo en una reunión de no más de 8 horas en la cual se planifica el Sprint [20].
3. **La Melé diaria.-** Es una reunión diaria con los miembros del equipo de no más de 15 minutos con el fin de sincronizar las tareas que se están desarrollando y realizar una planificación para las próximas 24 horas [20].
4. **Revisión del Sprint.-** Es una reunión en la cual se evalúa la calidad del entregable y que no dura más a allá de 4 horas para un Sprint de un mes y dos horas para uno de dos semanas [20].
5. **Retrospectiva del Sprint.-** Analiza el desempeño que está teniendo la metodología Scrum y evalúa si es necesario la implementación de un plan de mejora [20].

1.3.8.2 Kanban

Metodología cuyo objetivo es controlar el proceso de producción exitoso desde el inicio hasta la finalización del producto. Se encarga de la gestión del desarrollo incremental, historias de usuarios, tareas, etc [21]. Se utiliza un tablero Kanban en donde se registra el estado del proyecto y tiene una visión global a cada miembro del equipo, entre sus beneficios incluye:

- Estimulación al rendimiento con el fin de obtener mejores resultados [21].
- El tablero es compartido por cada miembro del equipo por lo que mejora la colaboración y organización [21].
- Cada miembro del equipo puede participar en las tareas, distribuir el trabajo y tener una comunicación efectiva gracias al tablero [21].

Principios básicos de Kanban

1. **Visualizar el trabajo.-** Se evidencia el avance de las fases a través del uso de una tarjeta Kanban [21].
2. **Limitar el trabajo en curso.-** Kanban tiene como principio la entrega temprana del producto limitando al equipo de trabajo en cumplir con una tarea a la vez hasta finalizarla y evitar el cumplimiento de tareas simultáneas [21].
3. **Gestionar el flujo de trabajo.-** Se centra en la reducción del tiempo utilizado por etapa pero a la vez cumpliendo con las historias de usuarios [21].
4. **Implementar ciclos de comentarios.-** Principio en el cual es participe el cliente pues es este quien brinda retroalimentación, además de analizar los comentarios de los miembros del equipo acerca de la ejecución del marco Kanban [21].

1.3.8.3 Extreme Programming (XP)

Fundamentada en la ideología de la sencillez, el coraje, comunicación y la retroalimentación cuyos principios se basan en la simplicidad, feedback, calidad y adaptación a los cambios incrementales. Sus objetivos están centrados en comprometer a los miembros del equipo mediante la asignación de roles y cumplimiento de fases para satisfacer las necesidades del cliente [21].

Los roles existentes en la metodología XP son los siguientes:

1. **Programmer.-** Encargado de la digitación del código funcional, se considera el miembro de mayor importancia en el equipo [21].
2. **Customer.-** Es quien elabora las historias de usuarios y asigna un grado de prioridad a cada una de estas, además de decide cual se implementa en cada iteración. Se encarga de aprobar y validar la funcionalidad [21].

3. **Tester.-** Encargado de ejecutar pruebas funcionales y difundir los resultados al equipo, responsable de las herramientas y software utilizados [21].
4. **Tracker.-** Proporciona un feedback al equipo, evalúa el grado de eficiencia y tiempo real consumido en las estimaciones hechas. Se encarga del seguimiento [21].
5. **Coach.-** Guía a los miembros del equipo para continuar con el proceso de manera correcta [21].
6. **Consultor.-** Es un elemento externo al proyecto pero que ayuda a resolver posibles conflictos aplicando sus conocimientos [21].
7. **Gestor.-** Encargado de la administración de tareas del equipo, planifica las reuniones, asiste a estas y lleva consigo información importante para el equipo [21].

La metodología incluye las fases:

1. **Exploración.-** Define el alcance del proyecto, los clientes generan las historias de usuarios y los desarrolladores en base a la documentación entregada se encargan de la evaluación de los tiempos de desarrollo [21].
2. **Planificación.-** Es de corta duración, en la cual se realizan reunión para elaborar la planificación en la cual se involucran el cliente, el equipo de trabajo y directivos en donde se establece la forma en que como se debe implementar las historias de usuarios (agrupando por entregas) [21].
3. **De iteraciones.-** Dentro del ciclo de XP es la fase que mayor importancia tiene debido a que en esta etapa se desarrollan las funcionalidades, al final de cada iteración se genera un entregable funcional en la que se encuentran las historias de usuarios asignadas a esta fase, se requiere la participación de los clientes para el análisis y correspondiente desarrollo [21].
4. **De puesta en producción.-** Se entregan módulos funcionales y sin errores. En caso de requerimientos adicionales se puede realizar un ajuste [21].
5. **Mantenimiento.-** Una vez finalizada la fase de puesta en producción entra en efecto esta nueva fase en la cual se analizan nuevas historias de usuarios y la

incorporación de nuevas funcionalidades dependiendo del valor del negocio; se pueden incorporar nuevos elementos al equipo de desarrollo [21].

6. **Finalización del proyecto.-** Esta fase se caracteriza por la inexistencia de nuevas historias de usuarios o cuando el valor de estos se hayan reducido unas ves implementadas [21].

1.3.9 Sistemas web

El desarrollo y ejecución de sistemas web por lo general requiere de código fuente del lado del servidor que genere archivos HTML (HyperText Markup Language) que son visualizados por el usuario a través de un navegador, la comunicación servidor-cliente se lo realiza a mediante peticiones HTTP; el servidor delega a distintos módulos la generación de vistas HTML y son enviadas a los clientes según su petición [22].

1.3.9.1 Sistemas web Multiplataforma

Son sistemas desarrollados en un único lenguaje de programación que permite su visualización en cualquier dispositivo sin importar su sistema operativo, al estar basados en un mismo lenguaje solo se necesita implementar cambios mínimos en su estructura para la adaptación a los dispositivos [22]. El desarrollo de sistemas web multiplataforma se puede hacer de dos maneras:

A través del uso de lenguajes como JavaScript, JQuery, HTML y CSS, en la cual se puede añadir las funcionalidades adaptativas a los atributos para que sean compatibles con cualquier dispositivos [22].

Por otro lado, se puede hacer uso de herramientas como: Flutter o React Native que son conocidos como de rendering nativo [22].

1.3.10 JavaScript

Lenguaje de programación del lado del cliente que es utilizado por los buscadores para la ejecución de código y controlar el comportamiento de aplicativos y páginas web; permite una comunicación asincrónica con el cliente es decir, que se pueden actualizar fragmentos de código o reemplazarlos de forma total. Actualmente, es utilizado por servidores como: Node.js, Meteor, CouchDB, etc [23].

1.3.11 HTML (HyperText Markup Language)

Es un lenguaje utilizado para estructurar sitios web, contenidos en forma de texto, agregar objetos estáticos o dinámicos como imágenes, videos, audios, etc. Se define como un conjunto de etiquetas que proveen múltiples funciones e interacción en los sistemas o páginas web, en otras palabras es el primer componente que forma parte de aplicativos que requieren acceso a través del uso de navegadores [24].

1.3.12 CSS (Cascading Style Sheets)

Lenguaje utilizado para proveer diseños a los elementos HTML para así mejorar la presentación al cliente, introduce la idea de separar el aspecto de la estructura, es decir; en una parte tenemos el sitio web escrito en un lenguaje de marcado y por otro lado se tiene al diseño gráfico del aplicativo [24].

1.3.13 Bootstrap

Es un enfoque orientado al diseño web optimizado para desarrollar una visualización óptima de los sitios web mejorando la experiencia de navegación de los usuarios, posee una capacidad de adaptación en todos los entornos (desde computadores hasta móviles) de todas las gamas de dispositivos. Un aplicativo web que posee las

dependencias de Bootstrap adapta su contenido a un formato de fluido, proporciones en cuadriculas propias de CSS3 lo que otorga fluidez en la carga de elementos [25].

1.3.14 Visual Studio Code

Editor de código desarrollado por Microsoft para Windows, Linux y MacOS, utilizado para editar, ejecutar y depurar aplicaciones que comúnmente son desarrolladas del lado del cliente (Frontend), pero que no presenta complicaciones para ejecutar código fuente de lado del servidor (Backend) [26]. Contiene características particulares como son:

- **Soporte para múltiples lenguajes de programación:** Cuenta con un soporte integrado para varios lenguajes, además de detectar la mala referencia entre los distintos lenguajes [26].
- **Intelli-Sense:** Tiene la capacidad de detectar líneas que generen errores o incompletas, las sintaxis de variables y declaraciones se las hace de forma automática [26].
- **Extensiones y soporte:** Existe la posibilidad de usar lenguajes de programación que no son compatibles únicamente al descargar la extensión sin afectar el rendimiento debido a que funciona como un proceso distinto [26].
- **Estructura de jerarquía:** Los archivos de código se encuentran organizadas en diversas carpetas, los directorios a su vez pueden contener subdirectorios y estos pueden gestionarse de acuerdo con la conveniencia del desarrollador [26].
- **Repositorio:** Permite la conexión con Git o plataformas similares para la almacenar o extraer código [26].
- **Soporte de Git:** El almacenamiento seguro y ágil es oportunamente importante por lo que la extracción de recursos desde Git se lo puede hacer en línea, además de la clonación de código [26].

1.3.15 MariaDB

Es un gestor de base de datos relacional Open Source desarrollado en el 2009 por los creadores de MySQL con la finalidad de usar las facilidades que este otorga pero con la característica de código abierto. Al poseer casi la misma estructura de MySQL existe la posibilidad de sustituirlo en proyectos sin afectar la operacionalidad. A pesar de que estos gestores son relativamente los mismos existe una ventaja por parte de MariaDB pues esta integra nuevos motores de almacenamiento más eficientes y mejoras en el rendimiento y seguridad [27].

1.3.15.1 Seguridad y disponibilidad

MariaDB en su seguridad proporciona autenticación y encriptación a nivel de red protegiendo de ataques de seguridad a través de soportes TLS/SSL en sus conectores, a nivel de servidor y aplicación lo que garantiza la protección de la información. Dispone de motores como: Aria, XtraDB; además, integra Clúster multi-master que permite mantener un clúster activo-activo garantizando una alta disponibilidad [27].

1.3.16 Node.js

Es un entorno de ejecución en tiempo real multiplataforma que se basa en el lenguaje de programación JavaScript y que es Open Source es decir; cualquier desarrollador puede hacer uso de sus paquetes de forma gratuita sin tener que adquirir una licencia. Su arquitectura está orientada específicamente a eventos y hace uso del motor de Google llamada V8 empleado para la interpretación ejecución de código JavaScript [28].

1.3.17 Frameworks y complementos para Node.js

Node.js posee una sintaxis que en muchas ocasiones puede considerarse de bajo nivel cuando se requiere usarse en ciertas tareas en el desarrollo de aplicaciones web por lo que generalmente cuando se usa esta plataforma se debe complementar con un framework JavaScript como: Sails, Express, Koa o Meteor que adicionan una capa de abstracción que provee servicios adicionales a los ya existentes de forma nativa [28].

1.3.18 Express

Framework que además de ser Open Source contiene módulos que facilitan el desarrollo de sitios y aplicaciones web. Fue desarrollado por la comunidad de Node.js con el fin de ofrecer una estructura de trabajo con múltiples funcionalidades como: la gestión de rutas carga de dependencias y la creación del servidor. Permite peticiones múltiples de forma concurrente que en teoría nos protege de los ataques de denegación de servicios [29].

1.3.19 Modelo Event-Loop

Node.js ejecuta operaciones sobre un hilo único de ejecución cuya finalidad es mejorar la escalabilidad y desempeño de aplicaciones web. El modelo Event-Loop está pensado para satisfacer la velocidad de respuesta de las peticiones haciendo que el desarrollo de las aplicaciones sean asíncronas reduciendo el consumo de recursos debido a que se evita el fork para atender las peticiones que hacen los clientes en procesos independientes [29].

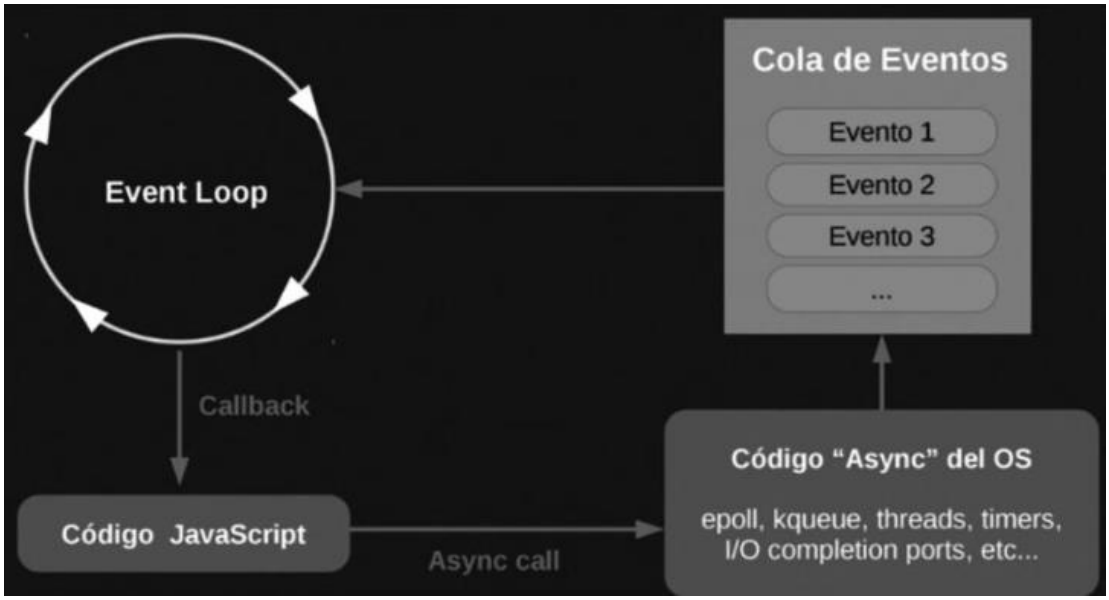


Figura 1.2: Ejecución asincrónica de JavaScript

Fuente: [29]

1.3.20 Gestión de paquetes

NPM (node package manager) es el principal gestor de paquetes por defecto que posee Node.js, administra módulos que se tienen instalados en el entorno de desarrollo, distribuye los paquetes y agrega nuevas dependencias a los proyectos haciendo que la programación sea modular, es decir; los desarrolladores crean diversos módulos diseñados para una tarea específica que después pueden reutilizarse e incluso compartirse [29].

Al crear un proyecto Node.js se crean también módulos de forma nativa (Módulos Nativos) por lo que es innecesario la descarga de paquetes extras para su utilización como lo son "http" usado para realizar peticiones, "fs" para acceder al sistema de archivos; "útil" un conjunto de utilidades o "domain" que permite el manejo de errores. Los paquetes que no están incluidos de forma nativa se los puede incluir mediante la ejecución de NPM [29].

1.4 Objetivos

1.4.1 Objetivo general

Implantar un sistema web multiplataforma desarrollado con arquitectura Modelo Vista Controlador (MVC) y Cloud Computing para la mejora de la gestión de actividades en NovaDent-Especialidades Odontológicas en la ciudad de Latacunga.

1.4.2 Objetivos específicos

- Analizar los procesos que intervienen en la gestión de actividades del centro odontológico.
- Investigar acerca de la arquitectura MVC y su aplicación en los sistemas web.
- Indagar sobre Cloud Computing, su importancia en las tecnologías de la información y los beneficios que este otorga en la implementación de un sistema web.
- Implementar un sistema web multiplataforma basado en arquitectura MVC y Cloud Computing en un servidor configurado en la nube.

CAPÍTULO II.- METODOLOGÍA

2.1 Materiales

Para el presente proyecto de investigación se utilizó una encuesta dirigida al gerente del centro odontológico y a los especialistas que forman parte de este, con el fin de conocer cuáles son los procesos que estos realizan para cumplir con sus tareas diarias y analizar los aspectos que generan inconvenientes; en complemento, se realizó una encuesta aplicada a los pacientes para descubrir datos relevantes acerca de la gestión de actividades del centro de atención.

Entrevista dirigida al gerente y especialistas de NovaDent-Especialidades Odontológicas.

Pregunta 1: ¿En la actualidad el centro odontológico NovaDent presenta inconvenientes o problemas que influyan de manera negativa en las actividades que se cumplen diariamente?

Pregunta 2: En el tiempo actual, ¿existe algún medio por el cual los pacientes pueden tomar sus turnos de atención con los distintos especialistas sin la necesidad de asistir personalmente al centro odontológico u optar por llamadas telefónicas que por lo general no suelen ser muy cómodas?

Pregunta 3: En caso de que un paciente requiera cancelar su turno o posponerlo, además de una llamada telefónica ¿Qué otros medios tienen a disposición para poder comunicarse con el centro odontológico?

Pregunta 4: Con respecto a la generación de historias clínicas o registro de información de los pacientes podría decirse que la manera en que lo realizan

actualmente es la forma óptima? En caso de tener una respuesta negativa, ¿cuál sería su punto de vista referente a dar una solución que beneficie al centro odontológico?

Pregunta 5: ¿Cuál es su opinión con respecto a que los procesos del centro odontológico puedan optimizarse haciendo más productiva las actividades, ganando tiempo y ahorrando recursos a través del uso de la tecnología que se encuentra en auge actualmente?

Pregunta 6: Con respecto a las actividades (registrar citas, crear historial clínico, almacenar imágenes de hallazgos potencialmente importante en algún paciente) realizados por los especialistas u otros profesionales odontólogos podría usted identificar la problemática que supone hacerlo de manera tradicional/manual utilizando materiales físicos (notas en papel, uso de folders y carpetas, etc)?

Pregunta 7: ¿En algún punto se han suscitado eventos por los cuales dieron cabida a la necesidad de buscar una solución óptima, por ejemplo: hubo una acumulación de pacientes en el centro odontológico por desorganización de los turnos de atención?

Pregunta 8: ¿Podría identificar las necesidades actuales del centro odontológico con respecto a innovación tecnológica se refiere?

Pregunta 9: ¿El centro odontológico estaría en posibilidad de adecuar una sala/cuarto con un ambiente e infraestructura adecuada exclusivamente para alojar un servidor?

Pregunta 10: ¿Cuál es su opinión con la idea de implementar un solución informática que ayude a la gestión de las actividades que se realizan en centro de atención y minimizar errores?

Encuesta aplicada a los pacientes para el análisis de actividades del centro odontológico.

1. ¿Considera usted que existe deficiencia en los aspectos mencionados a continuación?

- ☐ El tiempo de toma de información personal es excesivo
- ☐ El tiempo de acceso al historial clínico y médico es excesivo
- ☐ El registro de tratamientos se los hace en documentación física

2. ¿Ha tenido alguna vez algún tipo de inconveniente para poder registrar una cita para atenderse en el centro de atención odontológico?

- ☐ Siempre
- ☐ Casi siempre
- ☐ Con frecuencia
- ☐ En ocasiones
- ☐ Casi nunca
- ☐ Nunca

3. ¿Considera usted que el tiempo de espera es excesivo para poder reservar un turno y poder atenderse en el centro odontológico?

- ☐ Totalmente de acuerdo
- ☐ Parcialmente de acuerdo
- ☐ Indeciso
- ☐ Parcialmente en desacuerdo
- ☐ Totalmente en desacuerdo

4. ¿Considera usted que sería más factible reservar un turno para la atención médica desde su de residencia?

- ☐ Totalmente de acuerdo
- ☐ Parcialmente de acuerdo
- ☐ Indeciso
- ☐ Parcialmente en desacuerdo
- ☐ Totalmente en desacuerdo

5. ¿Considera usted que se podría agilizar el proceso de asignar horarios/turnos de atención al paciente del centro odontológico?

- ☐ Totalmente de acuerdo
- ☐ Parcialmente de acuerdo
- ☐ Indeciso
- ☐ Parcialmente en desacuerdo
- ☐ Totalmente en desacuerdo

6. ¿Qué tipo de aplicación le gustaría utilizar para la gestión de información en el centro odontológico?

- ☐ App
- ☐ Web
- ☐ Indeciso
- ☐ Aplicación de Escritorio
- ☐ Otro

7. Una vez que tiene su cita asignada, ¿considera que sería acertada la idea de recibir una notificación con anterioridad para que usted pueda presentarse a la cita de manera puntual?

- ☐ Totalmente de acuerdo
- ☐ Parcialmente de acuerdo
- ☐ Indeciso
- ☐ Parcialmente en desacuerdo
- ☐ Totalmente en desacuerdo

8. ¿Usted preferiría que su información personal e historial clínico (tratamientos realizados o en proceso) estén protegidos en soportes informáticos adecuados y que se pueda acceder a ellos únicamente mediante dispositivos inteligentes?

- ☐ Totalmente de acuerdo
- ☐ Parcialmente de acuerdo
- ☐ Indeciso
- ☐ Parcialmente en desacuerdo
- ☐ Totalmente en desacuerdo

2.2 Métodos

2.2.1 Modalidad de la Investigación

Investigación de campo

Se hizo uso de la modalidad de campo puesto que, el análisis de las causas y efectos además de la recolección de información se da en el lugar donde se origina el problema.

Investigación Bibliográfica-Documental

Se utilizó la modalidad bibliográfica-documental ya que estaba basada en la revisión de documentaciones similares al problema de investigación, donde su fuente de información sea más profunda, además de indagar acerca de cómo se llegó a la solución.

2.2.2 Población y Muestra

La población que se tomó en consideración son los especialistas que forman parte del cuerpo odontológico, el propietario de este, además de los pacientes que siguen un tratamiento en un período de tiempo extendido siendo estos los más relacionados con el centro de atención.

La siguiente tabla muestra el número de personas con las cuales se trabajó para el desarrollo del proyecto.

Población	Número	Porcentaje
Gerente del centro odontológico	1	5.26%
Especialistas del centro odontológico	3	15.79%
Pacientes	15	78.95%
Total	19	100%

Tabla 2.1: Población de estudio

Elaborado por: El Investigador

2.2.3 Recolección de Información

2.2.3.1 Resultados de la entrevista

Los resultados obtenidos de la entrevista aplicada al gerente y especialistas del centro odontológico fueron los siguientes:

Pregunta	Resultados	Comentarios	Conclusión General
1. ¿En la actualidad el centro odontológico NovaDent presenta inconvenientes o problemas que influyan de manera negativa en las	1. Impuntualidad de los pacientes. 2. Cruce de horarios. 3. Existe demora en la toma de datos personales cuando hay afluencia de pacientes.	Al existir desorganización por parte de los pacientes que en muchas ocasiones no asisten de acuerdo con su turno o no lo posponen/cancelan sin previo aviso, aumenta la posibilidad de un cruce de horarios haciendo que dentro de la sala	Actualmente, NovaDent-Especialidades Odontológicas presenta deficiencias en los procesos que se realizan a diario, esto es debido a múltiples causas que generan un retraso en

actividades que se cumplen diariamente?	4. En ocasiones, los pacientes se presentan al día siguiente de su turno.	de espera exista acumulación de personas lo que deriva en un retraso en las actividades que realiza la persona encargada de la toma de información.	actividades relacionadas específicamente con los pacientes ya sean nuevos o aquellos que cumplen con un plan de tratamiento a largo plazo, sin embargo, los especialistas quienes están directamente relacionados con los pacientes se han acoplado a la forma tradicional de trabajo que de cierta manera cumple con su objetivo pero no lo hace de forma eficiente y en el tiempo establecido. El trabajo dentro de las instalaciones se ha llevado acabo de esta manera desde sus inicios y no han optado el buscar una solución
2. En el tiempo actual, ¿existe algún medio por el cual los pacientes pueden tomar sus turnos de atención con los distintos especialistas sin la necesidad de asistir personalmente al centro odontológico u optar por llamadas telefónicas que por lo general no suelen ser muy cómodas?	1. Actualmente se usan medios de comunicación como llamadas telefónicas y mensajes directos por WhatsApp, y en ocasiones también se utilizan mensajes de texto, aunque no es muy común que se presente.	Los medios de comunicación entre paciente-odontólogo o paciente-recepcionista son tradicionales (llamadas telefónicas o mensajería por redes sociales).	

3. En caso de que un paciente requiera cancelar su turno o posponerlo, además de una llamada telefónica ¿Qué otros medios tienen a disposición para poder comunicarse con el centro odontológico?	1. Actualmente, los únicos medios de comunicación para cancelar un turno o posponerlo son únicamente los mensajes de WhatsApp que están dirigidos a la recepcionista. 2. No se registra ningún otro medio.	En la actualidad el centro odontológico no cuenta con algún medio además de mensajes por redes sociales (WhatsApp) que facilite una comunicación asertiva.	informática por falta de propuestas que optimicen los procesos.
4. Con respecto a la generación de historias clínicas o registro de información de los pacientes podría decirse que, ¿la manera en que lo realizan actualmente es la	1. No es la óptima, pero podría decirse que es funcional, además de ser la única forma de receptar la información. 2. La generación de números de historias clínicas se lo hace en base al número de cédula del paciente. Además, una solución	La manera en que se realizan las actividades no son las óptimas pero los especialistas del centro odontológico han tenido que acoplarse a estas debido a que es el único medio por cual realizar sus tareas.	

forma óptima? En caso de tener una respuesta negativa, ¿cuál sería su punto de vista referente a dar una solución que beneficie al centro odontológico?	informática podría ayudar a agilizar estos procesos.		
5. ¿Cuál es su opinión con respecto a que los procesos del centro odontológico puedan optimizarse haciendo más productiva las actividades, ganando tiempo y ahorrando recursos a través del uso de la tecnología que se	1. Es una propuesta viable debido a que las actividades serían mucho más sencillas de realizar, además de escatimar recursos materiales como: folders, resmas de papel, etc.	Existe un notorio interés por adicionar una herramienta tecnológica que permita gestionar de forma óptima los procesos realizados por los profesionales del centro odontológico.	

encuentra en auge actualmente?			
6. Con respecto a las actividades (registrar citas, crear historial clínico, almacenar imágenes de hallazgos potencialmente importante en algún paciente) realizados por los especialistas u otros profesionales odontólogos podría usted identificar la problemática que supone hacerlo de manera tradicional/manual utilizando materiales físicos	Existe acumulación y desorganización de la información así como también de la documentación de los pacientes. Las fotografías importantes se almacenan en teléfonos particulares, saturando el espacio de almacenamiento. Al existir muchos pacientes, los antiguos documentos tienden a extraviarse con frecuencia.	Las deficiencias en algunas actividades que realizan los odontólogos se presentan al momento de atender a los pacientes esto causa que el tiempo de atención se prolongue más de lo debido generando un cruce de horarios/turnos y otros inconvenientes.	

(notas en papel, uso de folders y carpetas, etc)?			
7. ¿En algún punto se han suscitado eventos por los cuales dieron cabida a la necesidad de buscar una solución óptima, por ejemplo: hubo una acumulación de pacientes en el centro odontológico por desorganización de los turnos de atención?	1. En cuanto a la impuntualidad de los pacientes, existen ocasiones en las cuales asisten en el turno que no les corresponde o hay pacientes que se presentan al día siguiente, pero no cancela las citas. 2. Cuando existen muchos pacientes, la toma de información se torna lenta, puesto que hay un único operario para receptar dicha información.	Los eventos que alteran el transcurso correcto de las actividades comúnmente son provocados ya sean por factores externos como los pacientes que acuden personalmente al centro odontológico o internos como una simple búsqueda de historial médico.	
8. ¿Podría identificar las necesidades actuales del centro odontológico con	1. La necesidad de generar un número de historial clínico de forma automática.	Los actividades que realizan los odontólogos se las cumple de forma tradicional, lo que en muchas de las	

respecto a innovación tecnológica se refiere?	2. La toma rápida de información de los pacientes. 3. La necesidad de tener un almacén de fotografías de los pacientes con hallazgos importantes y poderlos analizarlos a futuro. 4. Para consultar la información del paciente únicamente se requiere su cédula de identidad.	ocasiones genera malestar y demora en el cumplimiento de estas tareas.	
9. ¿El centro odontológico estaría en posibilidad de adecuar una sala/cuarto con un ambiente e infraestructura adecuada exclusivamente para alojar un servidor?	1. La respuesta a esta pregunta es negativa, puesto que llevaría demasiado tiempo y gasto de recursos, además de que el centro odontológico no cuenta con espacios para construir más salas que no sean destinadas directamente para atención a los	El centro odontológico no cuenta con un área que pueda destinarse a adecuaciones estructurales que no estén relacionados con atención al paciente es decir, nuevos consultorios o ampliar la sala de espera.	

	pacientes; el espacio muy reducido.		
10. ¿Cuál es su opinión con la idea de implementar un solución informática que ayude a la gestión de las actividades que se realizan en centro de atención y minimizar errores?	1. La solución es buena y óptima, sin embargo, el sistema se debe regir a las normas del centro odontológico, además de asegurar la confidencialidad de los datos de los pacientes. 2. El sistema debe ser capaz de agilizar los procesos, además de adaptarse a los cambios que se sucedan en el futuro.	Se necesita la implantación de una solución informática que cumpla correctamente su función y que esté apegado al marco legal del centro odontológico.	

Tabla 2.2: Resultados de la entrevista

Elaborado por: El Investigador

2.2.3.2 Resultados de la encuesta

Los resultados obtenidos de la encuesta aplicada en pacientes son los siguientes:

1. ¿Considera usted que existe deficiencia en los aspectos mencionados a continuación?

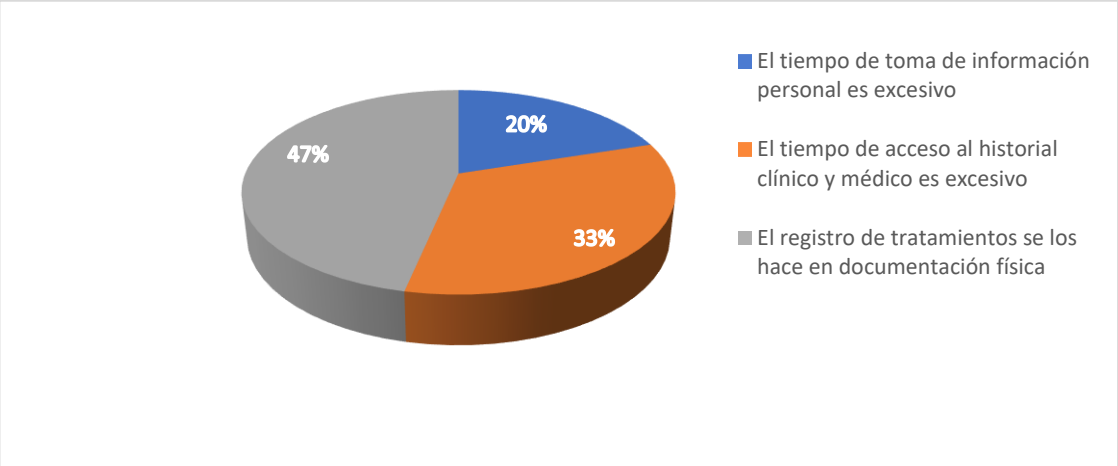


Figura 2.1: Pregunta de encuesta número 1

Elaborado por: El Investigador

Análisis e interpretación de resultados

Según los resultados representados en la figura 2.1, se evidencia que el 47% de los encuestados ha notado una deficiencia en el registro de información de sus tratamientos, el 33% indica que la deficiencia es notoria en el tiempo de acceso a las fuentes de información de su historial médico y finalmente el 20% indica que el tiempo de espera para la toma de su información personal es prolongado, por lo tanto, la mayoría de los encuestados tienen inconvenientes en la manera en que los especialistas llevan que el control de su información personal y planes de tratamientos debido a que esto se los realiza con la ayuda de documentación física, sin embargo, se debe tomar también en consideración que el tiempo para acceder al historial clínico y médico

también es un factor que ha provocado inconvenientes en los pacientes que están siendo atendidos.

2. ¿Ha tenido alguna vez algún tipo de inconveniente para poder registrar una cita para atenderse en el centro de atención odontológico?

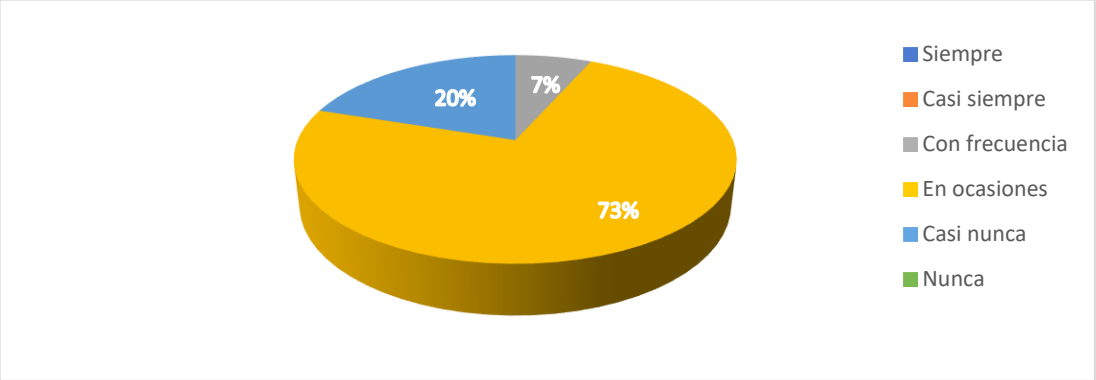


Figura 2.2: Pregunta de encuesta número 2

Elaborado por: El Investigador

Análisis e interpretación de resultados

De acuerdo con los resultados representados en la figura 2.2, el 73% de los encuestados ocasionalmente han tenido inconvenientes al momento de registrar una cita/turno para atenderse, seguido del 20% que casi nunca han tenido mayor problema para realizar la misma actividad y finalmente un 7% de los encuestados dicen que esta problemática se presenta con frecuencia, demostrando que la mayoría de los pacientes han tenido inconvenientes cuando tratan de registrar un cita/turno para atenderse en el centro odontológico.

3. ¿Considera usted que el tiempo de espera es excesivo para poder reservar un turno y poder atenderse en el centro odontológico?

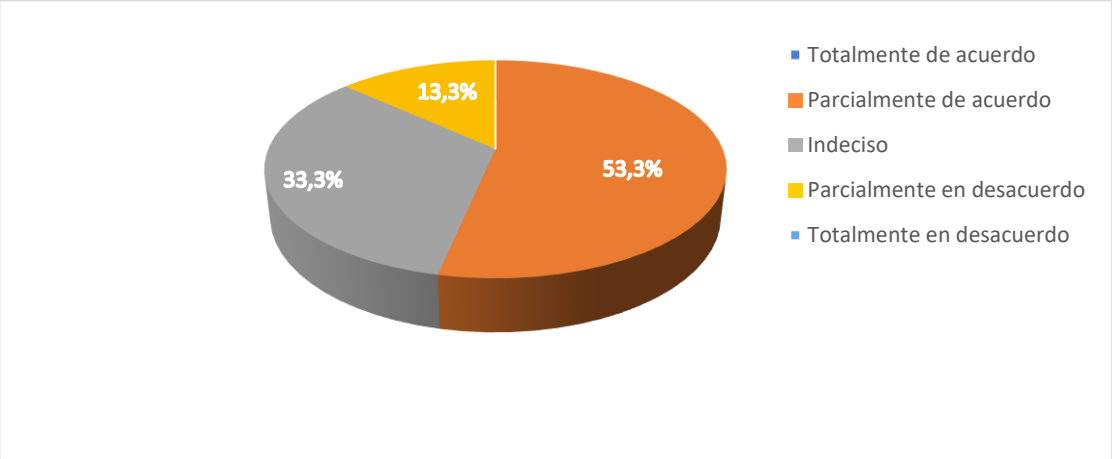


Figura 2.3: Pregunta de encuesta número 3

Elaborado por: El Investigador

Análisis e interpretación de resultados

Según los resultados representados en la figura 2.3, el 53.3% están parcialmente de acuerdo con el tiempo de espera para reservar un turno, seguido del 33.3% que no han notado diferencia alguna y finalmente un 13.3% de pacientes consideran que el tiempo empleado no han provocado serios inconvenientes, por lo tanto, es evidente que existe un excesivo tiempo de espera para poder reservar un turno de atención lo que en muchas ocasiones pueden causarles molestias, esto generalmente sucede en situaciones en donde existe acumulación de personas esperando atenderse.

4. ¿Considera usted que sería más factible reservar un turno para la atención médica desde su de residencia?

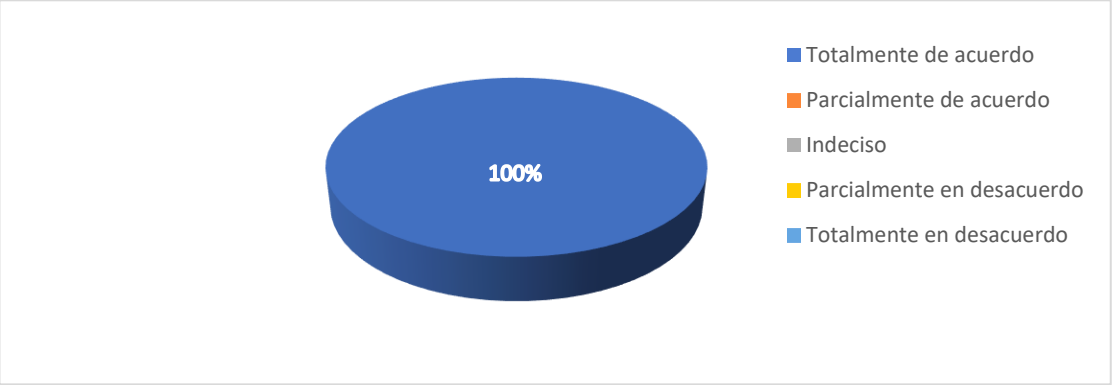


Figura 2.4: Pregunta de encuesta número 4

Elaborado por: El Investigador

Análisis e interpretación de resultados

De acuerdo con los resultados representados en la figura 2.4, se evidencia que el 100% de los encuestados están de acuerdo que es óptimo la posibilidad de realizar una reservación desde su lugar de residencia, demostrando que la totalidad de los encuestados consideran que es óptimo reservar turnos de atención desde su lugar de confort siendo mucho más cómodo y seguro, evitando el traslado innecesario hacia el centro odontológico o comunicación por medios tradicionales.

5. ¿Considera usted que se podría agilizar el proceso de asignar horarios/turnos de atención al paciente del centro odontológico?

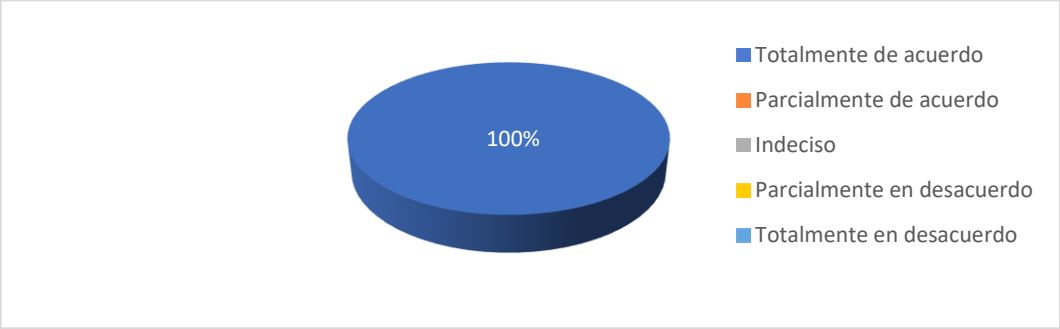


Figura 2.5: Pregunta de encuesta número 5

Elaborado por: El Investigador

Análisis e interpretación de resultados

De acuerdo con el resultado de la figura 2.5, el 100% de encuestados están totalmente de acuerdo con la idea de optimizar la asignación de turnos, indicando que los encuestados optan por la eficiencia en el cumplimiento de sus objetivos en cuanto a reserva de turnos se trata.

6. ¿Qué tipo de aplicación le gustaría utilizar para la gestión de información en el centro odontológico?

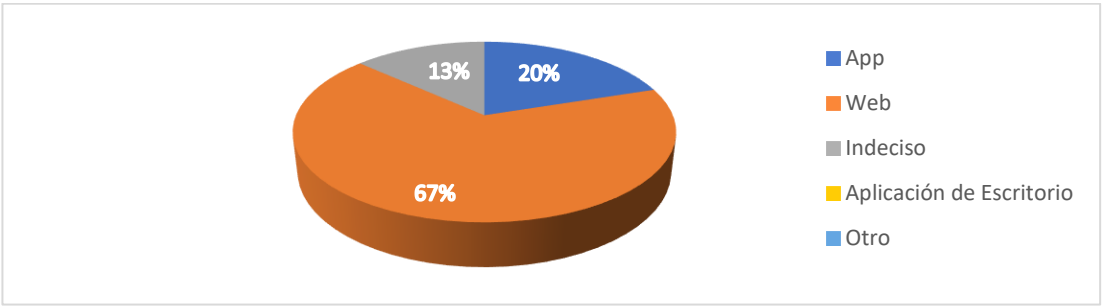


Figura 2.6: Pregunta de encuesta número 6

Elaborado por: El Investigador

Análisis e interpretación de resultados

Según los resultados representados en la figura 2.6, el 67% de los encuestados le gustaría utilizar un aplicativo web, seguido de un 20% que opta por una aplicación móvil y finalmente el 13% restante no encuentra interés para hacer uso de algún tipo de aplicación mencionadas en las opciones, por lo tanto, es evidente que la mayoría de los encuestados tienen cierto conocimiento e interés sobre el uso de un sistema web por lo que podrán adaptarse con facilidad a su uso.

7. Una vez que tiene su cita asignada, ¿considera que sería acertada la idea de recibir una notificación con anterioridad para que usted pueda presentarse a la cita de manera puntual?

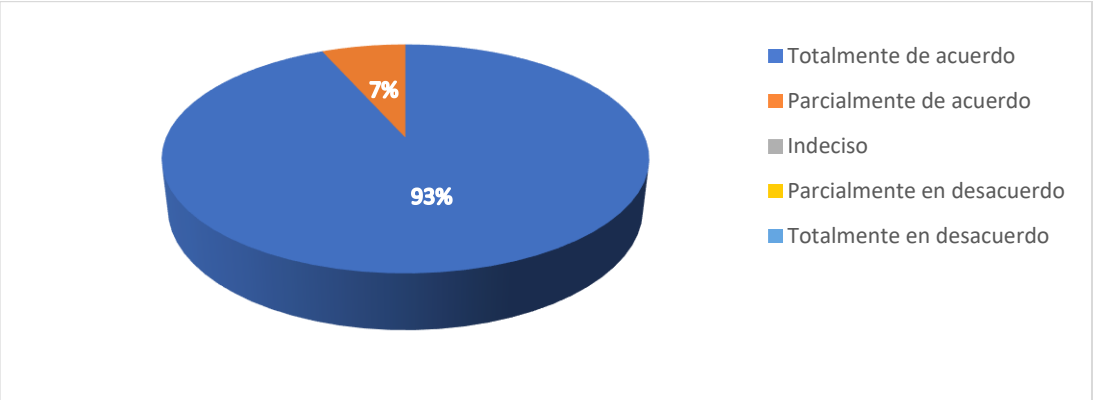


Figura 2.7: Pregunta de encuesta número 7

Elaborado por: El Investigador

Análisis e interpretación de resultados

De acuerdo con los resultados de la figura 2.8, es posible apreciar que el 93% de los encuestados están totalmente de acuerdo con recibir una notificación para asistir puntual a una revisión médica y un 7% están parcialmente de acuerdo con la idea,

indicando que la mayoría de los encuestados consideran que es acertada la idea de recibir una notificación tipo recordatorio para asistir a una revisión médica con puntualidad evitando inconvenientes por lo que se evidencia que existe cierta deficiencia en las notificaciones por parte del centro odontológico.

8. ¿Usted preferiría que su información personal e historial clínico (tratamientos realizados o en proceso) estén protegidos en soportes informáticos adecuados y que se pueda acceder a ellos únicamente mediante dispositivos inteligentes?

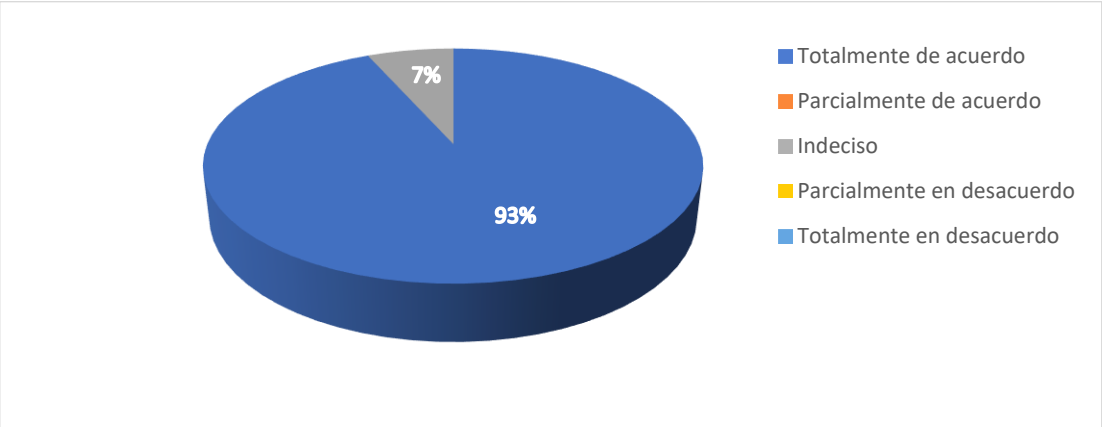


Figura 2.8: Pregunta de encuesta número 8

Elaborado por: El Investigador

Análisis e interpretación de resultados

Según los resultados representados en la figura 2.9, se puede observar que el 93% de los encuestados están de acuerdo en su totalidad en que su información personal esté debidamente protegida y un 10% no muestra interés en donde se encuentre almacenado su información, por lo tanto, la mayoría de los encuestados prefieren que su información personal y registro de sus tratamientos estén almacenados en soportes informáticos y que únicamente se acceda a ellos por medio de dispositivos tecnológicos lo que garantiza una mayor confidencialidad de dichos datos y también su tranquilidad como paciente.

2.2.4 Procesamiento y Análisis de Datos

De acuerdo con la entrevista dirigida al gerente y a los especialistas del centro odontológico, además de la encuesta aplicada a los pacientes, se demostró lo siguiente:

- La mayoría de los turnos de atención a los pacientes están registrados en documentos físicos (cuadernos) lo que provoca una acumulación de estos, motivo por el cual existe demora en el momento que se disponen a revisar el historial de atención.
- Existe una clara desorganización en la asignación de turnos y horarios de atención a los pacientes, lo que en muchas ocasiones pueden provocar molestias en los mismos.
- Es necesario llevar un control adecuado de los registros de tratamientos de los pacientes, es decir, que sea rápido acceder a estos en caso de ser necesario.
- En muchas ocasiones es conveniente evitar la generación manual del historial médico puesto que al existir muchos pacientes se pueden generar inconvenientes.
- Con frecuencia, los especialistas encuentran con hallazgos importantes durante la revisión a los pacientes por lo que es necesario documentarlas a través de fotografías que son almacenadas en sus dispositivos y estos tienden a saturar su espacio de almacenamiento.

CAPÍTULO III.- RESULTADOS Y DISCUSIÓN

3.1 Análisis y discusión de los resultados

3.1.1 Proceso de registro de turnos de atención y posterior revisión

Cuando una persona requiere un turno de atención odontológica debe seguir una serie de fases que termina en la valoración de su caso por parte del odontólogo.

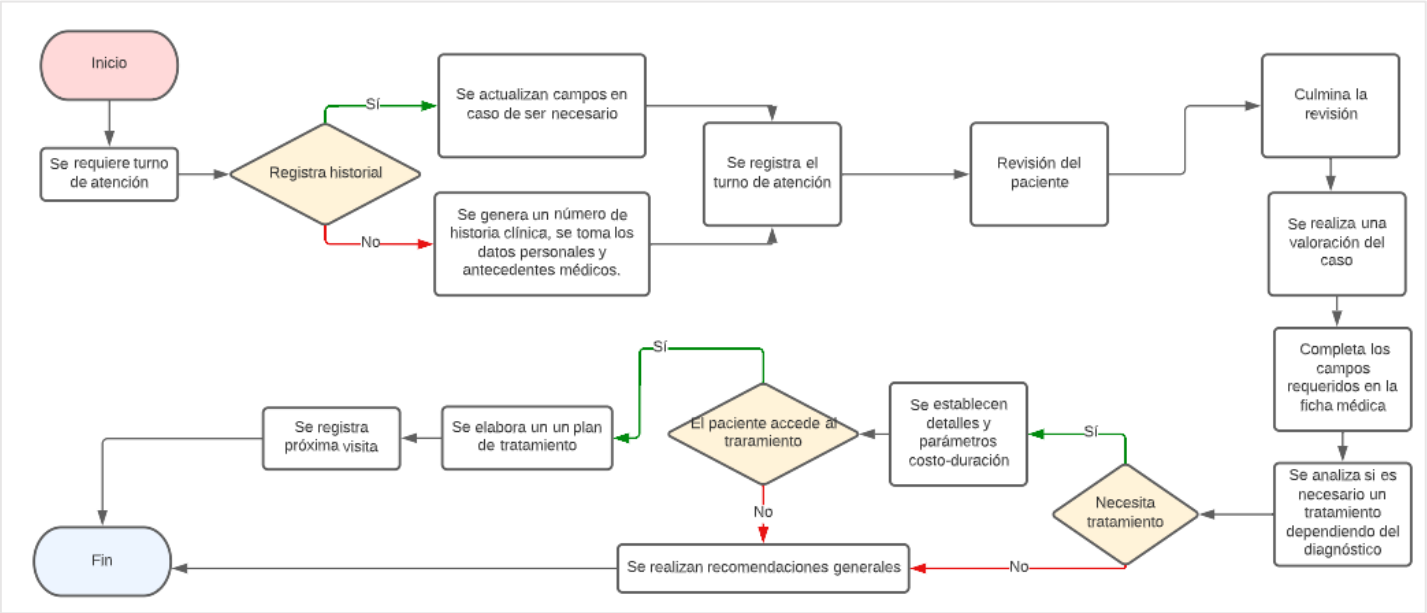


Figura 3.1: Proceso de registro de turno de atención

Elaborado por: El investigador

3.1.2 Descripción de las etapas de atención odontológica

Como se observa en la figura 3.1, el proceso para que un paciente pueda atenderse contempla una serie de etapas y fases que se describen a continuación:

1. Un cliente requiere de un turno para atención odontológica.

2. El recepcionista o personal a cargo verifica si ya cuenta con historial médico.
   2.1. Si el cliente ya es parte de los registros del centro odontológico (se pasa a llamar paciente) se agenda una cita en un horario que esté disponible y se ajuste a las necesidades del cliente.
   2.2. En caso de no contar con un historial médico, es decir; es un cliente nuevo, se procede a llenar un formulario con sus datos personales y ciertos antecedente médicos de mayor relevancia. A continuación, se agenda una cita según la disponibilidad de horarios.
3. Una vez que el paciente asiste a la cita, un odontólogo general procederá a hacer la revisión y evaluará el estado del caso, registrará en un formulario los hallazgos encontrado y se evaluara si es necesario un tratamiento.
4. Si es necesario un tratamiento se elaborará un plan de precio y duración, además de proporcionar información acerca del caso.
5. Si el paciente accede al plan de tratamiento se registrará una nueva cita, de lo contrario se termina el turno de atención.

3.1.3 Arquitectura MVC orientado a aplicaciones web

3.1.3.1 Características

- MVC permite una sencilla división de roles, cada capa puede ser trabajada por separado sin que afecte a la funcionalidad.
- Es fácilmente adaptable al desarrollo de aplicativo web permitiendo un desenvolvimiento más ágil.

3.1.3.2 Modelo

- Se encarga de la gestión de los datos, asegurando su persistencia y almacenamiento ya sean en base de datos o archivos planos.
- Accede a los almacenes de datos.

- Define como será la funcionalidad de la aplicación (negocio).
- Notifica los cambios en los datos a las vistas.

3.1.3.3 Vista

- Esta encargada de la interacción amigable con el cliente mostrando los datos que este requiera a través de elementos como formularios, cuadros de texto, etc.
- La vista proporciona una vista dinámica de las páginas que conforma el sistema según los requerimientos del cliente.

3.1.3.4 Controlador

- El controlador es el encargado de capturar los eventos generados por el usuario a través de la vista para posteriormente procesarlos en el modelo y finalmente regresar los datos nuevamente en la vista correspondiente.
- El controlador es quien decide que vista se utilizará.

3.1.4 Flujo de control

1. El usuario genera un evento a través de la interfaz.
2. El evento es receptado por uno de los métodos del controlador y es procesado.
3. El controlador hace una petición al modelo para modificarlos, obtenerlos o realizar la acción que se está requiriendo.
4. Estos datos obtenidos del modelo son nuevamente procesados por el controlador y son enviados a la vista.

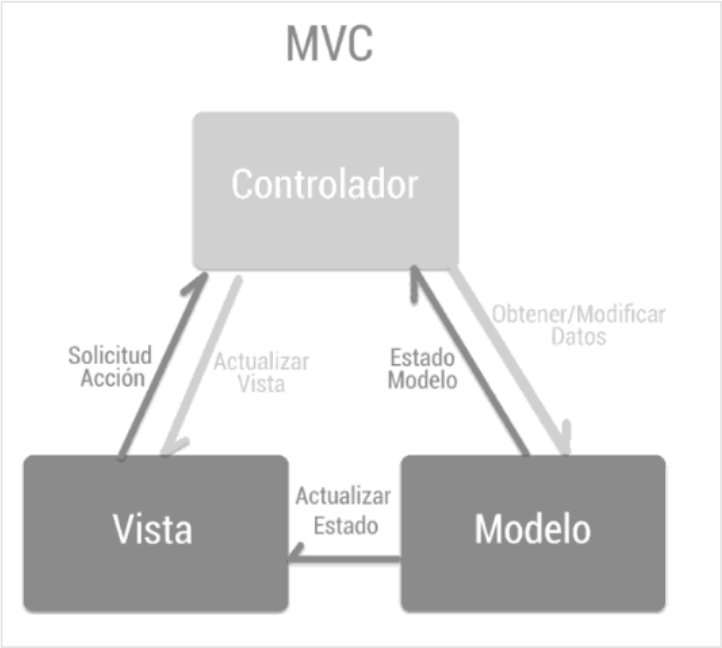


Figura 3.2: Flujo de control MVC

Fuente: [30]

3.1.5 Patrón MVC asociado a la tecnología Web

3.1.5.1 Vista

- Web
- Mobile
- Form

3.1.5.2 Modelo

- SGBD

3.1.5.3 Controlador

- Web server

- App server

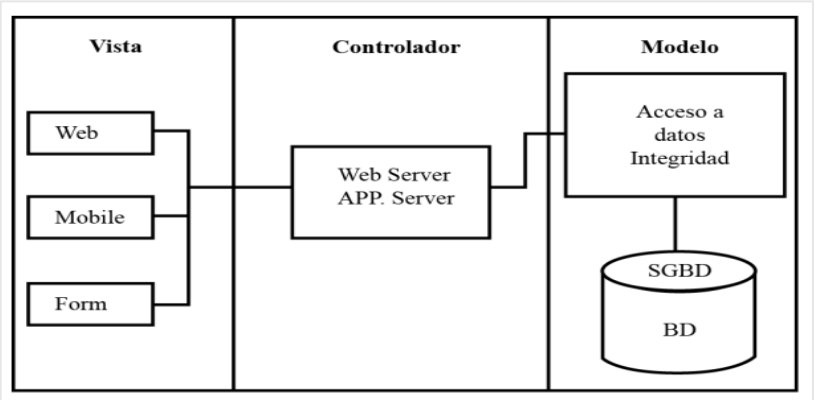


Figura 3.3: Patrón MVC asociado a la tecnología Web

Fuente:[30]

3.1.6 Cloud computing y su importancia en las tecnologías de la información

La computación en la nube ha tenido un gran impacto dentro del mundo tecnológico-empresarial. En el pasado, para que una organización pueda tener su información ordenada era necesario centralizarla en un centro de datos que suponía los siguientes aspectos y costos:

- Costo inicial
- Predecir la demanda
- Costo de mantenimiento
- Alcance limitado
- Latencia

Cloud computing permite distribuir de los recurso TI que demanda una organización a través del uso de internet introduciendo el término del “pago por uso” presentando las siguientes características:

Aspecto	Características
Cambio de costo inicial por costo variable	El costo inicial puede reducirse a cero o puede ir creciendo, dependiendo del uso de recursos.
Ahorro en mantenimiento técnico	El mantenimiento, refrigeración, seguridad, etc, se vuelven responsabilidad del proveedor de servicios de la nube.
Dejar de predecir demandas	El control y predicción de clientes que acceden a las plataformas dejan de ser un problema puesto que los recursos usados se pueden incrementar o disminuir de forma automática.
Accesibilidad	Se puede acceder a la nube desde cualquier sitio y cualquier dispositivo.
Agilidad	La adquisición de nuevos programas es mucho más fácil y rápido así también como la eliminación de estos.
Software está en la nube	Todo el software está en la nube y por esta razón no hay necesidad de instalar programas en cada dispositivo cliente.
Ahorro en hardware	No hay necesidad de adquirir nuevos implementos software como discos duros puesto que todo se almacena en la nube.
Ahorro en software	Un mismo aplicativo o programa puede ser compartido por los usuarios sin necesidad de adquirirlos individualmente.

Elasticidad	Los aplicativos cloud se adaptan a cualquier sistema en el que estén siendo consumidos.
Escalabilidad	Permite acceder a múltiples usuarios simultáneos.
Recuperación	Los proveedores cloud ofrecen un almacenamiento emergente o secundario para guardar información y recuperarla si se requiere.
Seguridad	Los proveedores de servicio cloud realizan controles estrictos de seguridad para proteger la información de los clientes.

Tabla 3.1: Características de Cloud Computing

Elaborado por: El investigador

Por lo expuesto en la tabla 3.1, se aprecia la clara ventaja que provee Cloud Computing al sector empresarial frente a un centro de datos tradicional siendo el ahorro económico uno de los aspectos más sobresalientes, puesto que únicamente por la contratación de un modelo de servicio en la nube el proveedor garantiza total seguridad, protección, transparencia y acceso a los datos de forma eficiente.

3.1.7 Metodologías de desarrollo

3.1.7.1 Determinación de metodología de desarrollo

A continuación, se muestra una tabla comparativa con tres metodologías ágiles como lo son: Scrum, Extreme Programming (XP) y Kanban. Se hizo elección de estas metodologías debido a que se encuentran orientadas al desarrollo de software y al tener mayor fuentes de información para su aplicación.

Características	Metodologías		
	XP	**Scrum**	**Kanban**
Objetivo principal	Comprometer a los miembros del equipo mediante la asignación de roles y cumplimiento de fases para satisfacer las necesidades del cliente.	Elaborar un marco de trabajo que agilite el desarrollo del proyecto dividiéndolo en iteraciones y tareas específicas.	Proporcionar una mejor visualización de las fases del proyecto y del flujo de trabajo usando el tablero Kanban.
Nivel de interacción con usuarios	Alto	Alto	Alto
Complejidad de uso	Media	Media	Media
Tamaño de proyectos	Pequeños, medianos	Grandes, adaptables a medianos y pequeños	Pequeños
Tiempo de duración	Medios, cortos	Largos, adaptable a otros tiempos de duración	Medios
Aumento de productividad	Alto	Alto	Alto
Integración de cambios	Cambios en cada iteración	Cambios continuos	Durante el plazo de entrega

Método de aplicación	El proyecto se basa en la simplicidad otorgando roles a cada miembro del equipo y aplicando un conjunto de fases.	El proyecto se divide en Sprints cada una con un conjunto de tares específicas y reglas.	La tareas se las exponen un tablero a la vista de todos los miembros garantizando un conocimiento proporcionado, el tablero debe mantenerse en continua actualización.
Entregables	Cuando se termina cada iteración	Entrega continua	En el plazo de entrega correspondiente
Documentación	Historias de usuarios	Historias de usuarios Definiciones de hecho	Tablero Kanban

Tabla 3.2: Cuadro comparativo de metodologías ágiles

Elaborado por: El investigador

Por lo expuesto en la tabla 3.2 se eligió la metodología XP debido a que presta sencillez, comunicación, tiene un tiempo de duración adaptable al proyecto de investigación, admite ajustes en las iteraciones; todo esto con el fin de generar la satisfacción del cliente a través de entregables de calidad. Adicionalmente se adapta al trabajo en equipo lo que facilita el desarrollo de las actividades.

3.1.8 Frameworks de desarrollo

Características	**Framework**		
	Node.js	**Laravel**	**Angular**
Lenguajes soportados	JavaScript	PHP	TypeScript
Licencia	MIT	MIT	MIT
ORM	Sequelize	Eloquent	Redux
Soporte y seguridad	Comunidad Node.js	Comunidad Laravel y Taylor Otwell	Comunidad Angular y Guards
Motor de plantillas	Template engines	Blade	Ivy engine
Tipo de código	Compilado	Interpretado	Compilado
Programado en	C++, JavaScript	PHP	TypeScript
Admite MVC	Admite	Modificaciones a directorios	Admite
Tiempo de aprendizaje	Rápido	Medio	Medio
Envíos de mensajes de WhatsApp	Uso de la librería Whatsapp-web.js	(Twilio, messagebird, nexmo)	(Twilio, messagebird, nexmo)

Tabla 3.3: Comparación de Frameworks de desarrollo

Elaborado por: El investigador

3.1.8.1 Determinación de Framework de desarrollo

Se ha seleccionado Node.js como Framework de desarrollo debido a que tanto el Frontend como Backend pueden desarrollarse usando la misma tecnología sin recurrir a otras, además de ser multiplataforma y permitir la arquitectura MVC como base del aplicativo; proporciona un ORM que permite la eficaz manipulación de datos y una facilidad de aprendizaje. Además, cuenta con la factibilidad de usar la librería Whatsapp-web.js como mensajería de WhatsApp sin el uso de plataformas intermediarias que suponen un costo mensual.

3.2 Desarrollo de la propuesta

3.2.1 Fase 1: Exploración

3.2.1.1 Requerimientos del sistema

En base a los resultados obtenidos a partir de la encuesta aplicada a los pacientes y la entrevista dirigida al gerente así como a los especialista del centro odontológico se ha llegado a la conclusión de la necesidad de implementar una sistema web y automatizar actividades que se llevan a cabo en la institución de salud como lo son: la asignación de turnos, registro de la información del paciente, postergación/cancelación de citas y llevar un control de los documentos importantes con la ayuda de los siguientes módulos:

- **Control de acceso.-** El acceso a los módulos se los hará mediante correo electrónico y su respectiva contraseña.
- **Gestión de pacientes.-** Se mostrará una lista de los pacientes con su respectiva información personal, esta podrá modificarse en caso de requerirse; nuevos pacientes podrán añadirse al sistema.

- **Gestión de tratamiento médico.-** Se encuentra la información de los pacientes que estén en proceso de tratamiento con sus respectivas observaciones y fechas de visitas.
- **Galería de imágenes.-** En esta sección se almacenarán las fotografías del proceso de atención de los pacientes, es decir que se mostrarán imágenes desde el inicio hasta el fin del tratamiento.
- **Gestión de citas.-** En esta sección el usuario podrá asignar turnos de atención según el requerimiento del paciente.
- **Notificaciones.-** Una vez que un paciente tiene una cita programada un día antes se enviará un mensaje a través de WhatsApp con un recordatorio.
- **Cerrar sesión.-** El usuario que tiene activa la conexión al sistema podrá cerrar sesión limpiando el caché y el token de autenticación evitando el retroceso a páginas anteriores.

3.2.1.2 Arquitectura de la aplicación

El presente proyecto se desarrolló con el entorno en tiempo de ejecución multiplataforma Node.js, el sistema de gestión de datos MariaDB y en base a la arquitectura de desarrollo MVC; el servidor en el cual se almacena el sistema web se encuentra configurado en el modelo de servicio en la nube PaaS haciendo uso del término “pago por uso”.

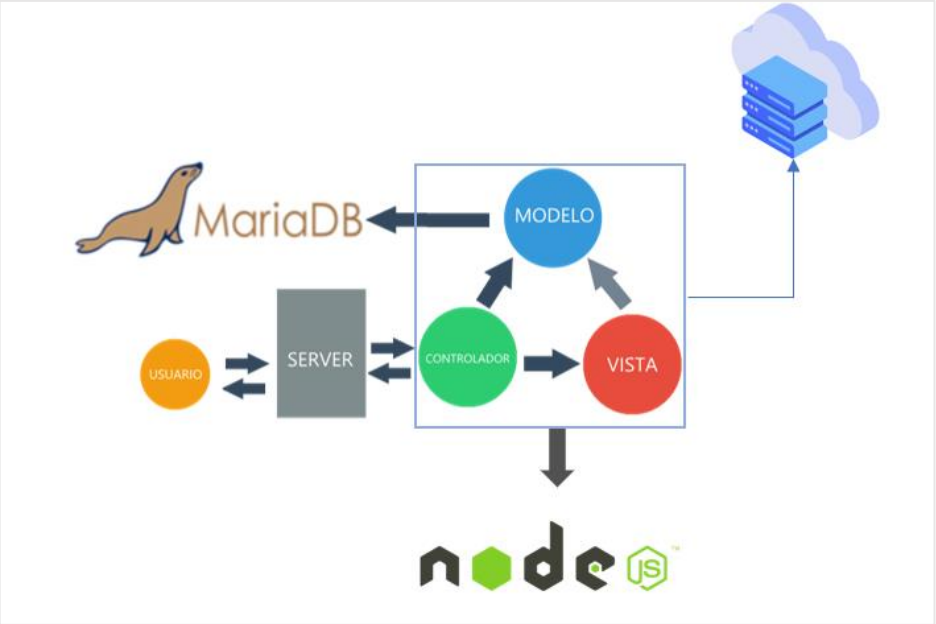


Figura 3.4: Arquitectura de la aplicación

Elaborado por: El investigador

3.2.1.3 Requerimiento de software

Se realizó un análisis acerca de las herramientas, software y de tecnologías necesarias para el desarrollo de aplicaciones web y que se describen a continuación:

- **Visual Studio Code.-** Es un editor de código fuente optimizado para el desarrollo y depuración de aplicaciones web desarrollado por Microsoft.
- **JavaScript.-** Lenguaje de programación de scripts que añade funcionalidades y eventos a los aplicativos web.
- **Node.js.-** Definido como un entorno de ejecución multiplataforma basado en JavaScript y de código abierto que se orienta a eventos asíncronos, está diseñado para la creación de aplicaciones del lado del servidor y aplicaciones tradicionales.
- **HTML (HyperText Markup).-** Es un lenguaje de marcado cuya función principal es definir los contenidos de los aplicativos y páginas web.

- **CSS (Cascading Style Sheets**).- Lenguaje de estilos que especifican la forma de presentación de los aplicativos a los usuarios a través de un enfoque visual otorgados por las interfaces.
- **Bootstrap.-** Conocida como una biblioteca multiplataforma y de código abierto que se utiliza para el diseños de los aplicativos web.
- **MariaDB.-** Sistema de gestión de base de datos con licencia GPL (General Public Licence) que se deriva de MySQL.

3.2.1.4 Roles del proyecto

La metodología XP menciona los roles que se deben otorgar a cada miembro del equipo de desarrollo, estos se describen a continuación:

Nombre	Rol	Función
Esteban Chanatasig	Programador	Realizar el análisis de los requerimientos, además de la estructura del aplicativo, diseño y pruebas funcionales del sistema web.
Dra. Nancy Rubio	Cliente -Tester	Realizar las peticiones y hacer pruebas del sistema web.
Ing. Fernando Ibarra	Entrenador (coach)	Realizar el seguimiento a las actividades del proyecto y revisión de los avances de forma constante.

Tabla 3.4: Roles del proyecto

Elaborado por: El investigador

3.2.2 Fase 2: Planificación

Para la planificación se ha optado por utilizar historias de usuarios que facilitan la transmisión de ideas proporcionadas por el cliente y la cual describe cada funcionalidad, estas se encuentran distribuidas en una cierta cantidad de iteraciones.

3.2.2.1 Historias de usuario

Las historias de usuarios son los requerimientos por parte del cliente y los cuales describen de forma sencilla las funcionalidades que se deben implementar en el sistema.
Para el desarrollo del presente proyecto se han definido las siguientes historias de usuarios que consta con la participación del cliente y el investigador.

- **Número:** Identificador de la historia de usuario.
- **Usuario:** Persona a cargo de la actividad.
- **Nombre de historia:** Nombre general de la historia de usuario.
- **Prioridad de negocio:** Es el nivel de importancia que provee el cliente (Alta, Media, Baja).
- **Riesgo de desarrollo:** Es el nivel de riesgo que representa el desarrollo de la actividad para el programador (Alta, Media, Baja).
- **Puntos estimados:** Cantidad de días estimados para el desarrollo de la historia de usuario.
- **Iteración asignada:** Iteración asignada a la historia de usuario.
- **Programador responsable:** Persona encargada del desarrollo de la actividad.
- **Descripción:** Detalles técnicos que provee un breve resumen acerca de la actividad.
- **Observación:** Actividades que se debe tener en consideración.

Historia de usuario	
Número: 1	**Usuario:** Cliente
Nombre de historia: Ingreso al sistema	
Prioridad en negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada:** 2
Programador responsable: El investigador	
Descripción: Ingreso al sistema a través de un correo electrónico y contraseña.	
Observaciones: El usuario que inicie sesión e ingresar al sistema deberá estar previamente registrado.	

Tabla 3.5: Historia de usuario - Inicio de sesión

Elaborado por: El investigador

Historia de usuario	
Número: 2	**Usuario:** Cliente
Nombre de historia: Reseteo de contraseña	
Prioridad en negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada:** 2
Programador responsable: El investigador	
Descripción: Cuando un usuario no recuerde su contraseña se le desplegará una interfaz en la cual se enviará un correo electrónico, una vez ingresado a la dirección especificada en el mensaje de email el usuario debe ingresar una nueva contraseña cumpliendo con el formato especificado.	
Observaciones: El usuario debe proporcionar un correo real.	

Tabla 3.6: Historia de usuario - Reseteo de contraseña

Elaborado por: El investigador

Historia de usuario	
Número: 3	**Usuario:** Cliente
Nombre de historia: Gestión de pacientes	

Prioridad en negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 3	**Iteración asignada:** 3
Programador responsable: El investigador	
Descripción: Se muestra el listado de todos los pacientes del centro odontológico con la opción de modificar sus datos o de ingresar uno nuevo al sistema.	
Observaciones: El ingreso de un nuevo paciente en el sistema se lo hará una sola vez.	

Tabla 3.7: Historia de usuario - Gestión de pacientes

Elaborado por: El investigador

	Historia de usuario
Número: 4	**Usuario:** Cliente, paciente
Nombre de historia: Galería de imágenes	
Prioridad en negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 3	**Iteración asignada:** 3
Programador responsable: El investigador	
Descripción: Se muestra el listado de todos los pacientes que posean fotografías referentes al plan de tratamientos.	
Observaciones: Las fotografías son referentes de acuerdo con el tratamiento en la que se encuentren , se las organizará en orden cronológico dependiendo de la fecha en la cual se subió al sistema.	

Tabla 3.8: Historia de usuario - Galería de imágenes

Elaborado por: El investigador

	Historia de usuario
Número: 5	**Usuario:** Cliente, paciente
Nombre de historia: Gestión de citas	
Prioridad en negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 3	**Iteración asignada:** 6
Programador responsable: El investigador	

Descripción: El recepcionista de turno deberá poder verificar si una persona registra un turno de atención haciendo uso de una consulta mediante el número de cédula.
Observaciones: Si la persona no registra un turno de atención se procederá a generar uno en caso de que la respuesta del paciente lo requiera.

Tabla 3.9: Historia de usuario - Gestión de citas

Elaborado por: El investigador

Historia de usuario	
Número: 6	**Usuario:** Cliente
Nombre de historia: Gestión de tratamiento médico	
Prioridad en negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 3	**Iteración asignada:** 4
Programador responsable: Esteban Chanatasig	
Descripción: En la lista de pacientes es posible acceder al panel en el cual se puede asignar un tratamiento nuevo y si el médico lo requiere se puede agregar observaciones de acuerdo con cada visita del paciente.	
Observaciones: Los pacientes pueden mantenerse en múltiples tratamientos.	

Tabla 3.10: Historia de usuario - Gestión de información médica

Elaborado por: El investigador

Historia de usuario	
Número: 7	**Usuario:** Cliente
Nombre de historia: Número de historia clínica	
Prioridad en negocio: Media	**Riesgo de desarrollo:** Media
Puntos estimados: 2	**Iteración asignada:** 8
Programador responsable: Esteban Chanatasig	
Descripción: El número de historia clínica correspondiente a cada paciente se lo debe generar de formar automática una vez que este ingrese al sistema, dicho número deberá estar basado en el número de cédula de identidad.	

Observaciones: Cada número de historia debe ser único.

Tabla 3.11: Historia de usuario - Número de historia clínica

Elaborado por: El investigador

	Historia de usuario
Número: 8	**Usuario:** Cliente
Nombre de historia: Cerrar sesión	
Prioridad en negocio: Media	**Riesgo de desarrollo:** Media
Puntos estimados: 2	**Iteración asignada:** 5
Programador responsable: Esteban Chanatasig	
Descripción: La sesión actual del usuario registrado procederá a destruirse evitando mantener iniciada la sesión.	
Observaciones: El usuario debe primero iniciar sesión.	

Tabla 3.12: Historia de usuario - Cerrar sesión

Elaborado por: El investigador

	Historia de usuario
Número: 9	**Usuario:** Cliente, paciente
Nombre de historia: Notificaciones WhatsApp	
Prioridad en negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 3	**Iteración asignada:** 5
Programador responsable: Esteban Chanatasig	
Descripción: Una vez que un cliente requiere de un turno de atención el cliente puede asignarlo de acuerdo con la disponibilidad de horarios, una vez establecido la cita se enviará un mensaje de WhatsApp con los detalles de la cita.	
Observaciones: El usuario debe tener una cuenta registrada en WhatsApp.	

Tabla 3.13: Historia de usuario - Notificaciones WhatsApp

Elaborado por: El investigador

Historia de usuario	
Número: 10	**Usuario:** Cliente, paciente
Nombre de historia: Notificaciones WhatsApp de recordatorio	
Prioridad en negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 3	**Iteración asignada:** 5
Programador responsable: Esteban Chanatasig	
Descripción: Un día antes de la cita registrada de cada paciente se le enviará un mensaje de WhatsApp con un recordatorio.	
Observaciones: El usuario debe tener una cuenta registrada en WhatsApp y debe tener un turno asignado.	

Tabla 3.14: Historia de usuario - Notificaciones WhatsApp de recordatorio

Elaborado por: El investigador

3.2.2.2 **Estimación de Historias de Usuario**

El tiempo estimado para las historias de usuarios es de 5 horas diarias es decir, 30 horas de trabajo a la semana, la distribución del tiempo de ejecución de cada actividad se la detalla a continuación:

No.	Historia de usuario	Tiempo estimado		
		Horas	**Días**	**Semanas**
1	Ingreso al sistema	5	1	0.2
2	Reseteo de contraseña	5	1	0.2
3	Gestión de pacientes	15	3	0.6
4	Galería de imágenes	10	2	0.4
5	Gestión de citas	5	1	0.2
6	Gestión de tratamiento médico	15	3	0.6
7	Número de historia clínica	15	3	0.6
8	Cerrar sesión	5	1	0.2
9	Notificaciones WhatsApp	5	1	0.2

10	Notificaciones WhatsApp de recordatorio	5	1	0.2
Total estimado		**85**	**17**	**3.4**

Tabla 3.15: Estimación de Historias de Usuario

Elaborado por: El investigador

3.2.2.3 Estimación de Actividades de Desarrollo de Backend y Frontend

Para el desarrollo del flujo de trabajo Backend y Frontend se ha determinado una serie de actividades con su tiempo estimado para su realización.

Actividad	**Tiempo estimado**		
	Horas	**Días**	**Semanas**
Diseño del modelo entidad-relación	5	1	0.2
Creación de controladores	5	1	0.2
Creación del modelo de datos	5	1	0.2
Definición de rutas en el archivo index.js	2	0.5	0.1
Definición de los templates engines	2	0.5	0.1
Diseño de interfaces	15	3	0.6
Total estimado	**34**	**7**	**1.4**

Tabla 3.16: Estimación de Actividades de Desarrollo de Backend y Frontend

Elaborado por: El investigador

3.2.2.4 Plan de Entrega

No.	**Actividades / Historia de Usuario**	**Iteración**					**Tiempo estimado**	
		1	**2**	**3**	**4**	**5**	**Horas**	**Días**

-	Diseño del modelo entidad-relación	**X**					5	1
-	Creación de controladores	**X**					5	1
-	Creación del modelo de datos	**X**					5	1
-	Definición de rutas en el archivo index.js	**X**					2	0.5
-	Definición de los templates engines	**X**					2	0.5
-	Diseño de interfaces	**X**					15	3
1	Inicio de sesión		**X**				5	1
2	Reseteo de contraseña		**X**				5	1
3	Gestión de pacientes			**X**			15	3
4	Galería de imágenes			**X**			10	2
5	Gestión de citas			**X**			5	1
6	Gestión de tratamiento médico			**X**			15	3
7	Número de historia clínica			**X**			15	3
8	Cerrar sesión				**X**		5	1
9	Notificaciones WhatsApp					**X**	5	1
10	Notificaciones WhatsApp de recordatorio					**X**	5	1

Tabla 3.17: Plan de Entrega

Elaborado por: El investigador

3.2.3 Fase 3: Iteraciones

3.2.3.1 Iteración 1

Diseño del diagrama entidad-relación

Para el diseño del modelo relacional se utilizó la herramienta de creación de diagramas que se encuentra en el gestor de base de datos SQL Server.

Tarea 1	
Nombre de tarea: Diseño de la base de datos	
Tipo de Tarea: Diseño	**Puntos Estimados:** 3
Programador responsable: Esteban Chanatasig	
Descripción: Diseño relacional de la base de datos.	

Tabla 3.18: Tarea - Diseño relacional de la base de datos

Elaborado por: El investigador

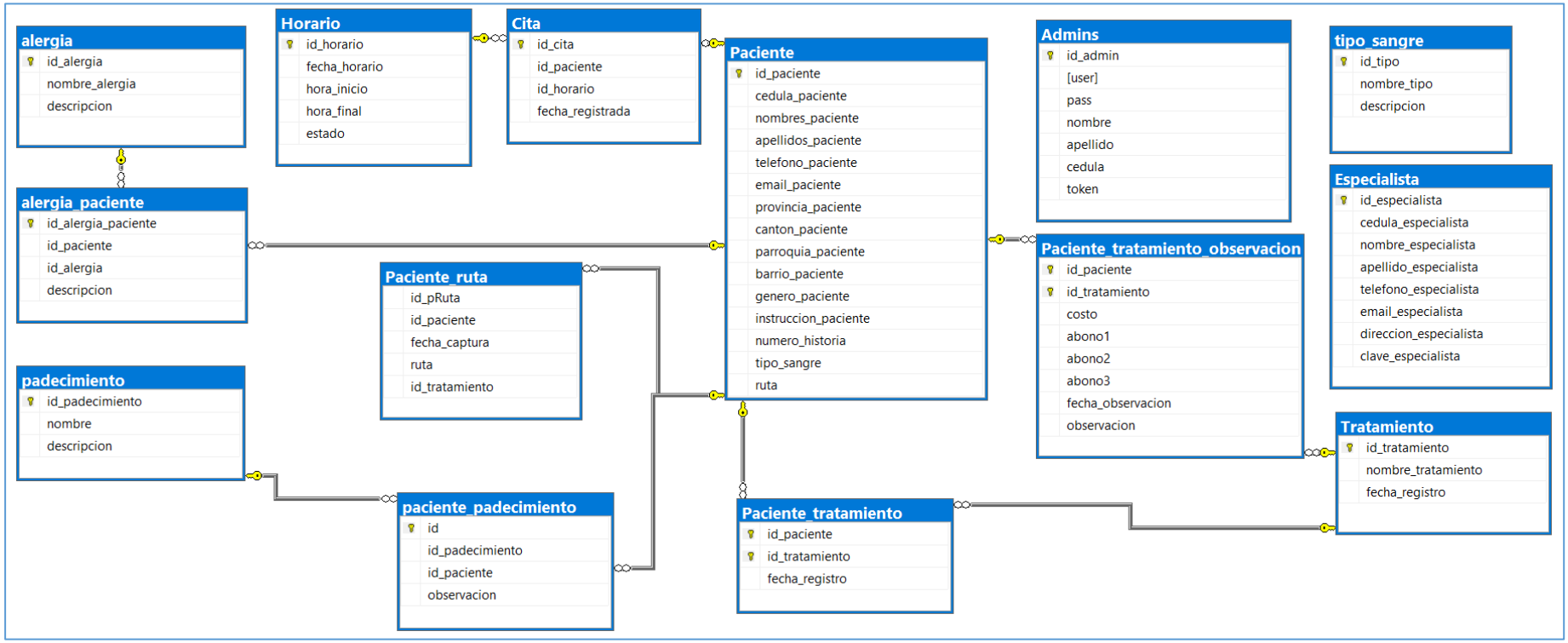


Figura 3.5: Diseño del diagrama entidad-relación

Elaborado por: El investigador

Desarrollo del Backend

Tarea 2	
Nombre de tarea: Desarrollo del Backend	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Programador responsable: Esteban Chanatasig	
Descripción: Desarrollo del Backend, instalación de las dependencias/módulos necesarias para el proyecto y gestión de archivos. Se muestra la estructura de como se ve el aplicativo de forma abstracta.	

Tabla 3.19: Tarea - Desarrollo del Backend

Elaborado por: El investigador

Para el desarrollo de Backend de hizo una conexión a la base de datos MariaDB que se encuentra configurada en la carpeta app/config/db.js y la cual permite el acceso a los datos a través de los modelos de las distintas entidades que participan en las transacciones SQL. Los controladores acceden a los modelos, y son estos quienes tienen relación directa con las vistas.

El archivo index.js se le considera el núcleo de la aplicación ya que en este se encuentran especificados las configuraciones, motores de plantillas (se hizo uso de la extensión .ejs), los archivos estáticos, el puerto en el que funciona así como distintos middlewares.

En la carpeta app/routes se encuentran especificadas las rutas que contienen la acción que se va a realizar de acuerdo con la petición que haga el cliente, en esta sección los controladores que contienen los diversos eventos son instanciados para la generación de acciones dependiendo de la vista en la cual son solicitadas. Manejan una sintaxis simple como: ***app.put/post/get('/nombreRuta/', instanciaControlador.evento).***

Arquitectura Backend

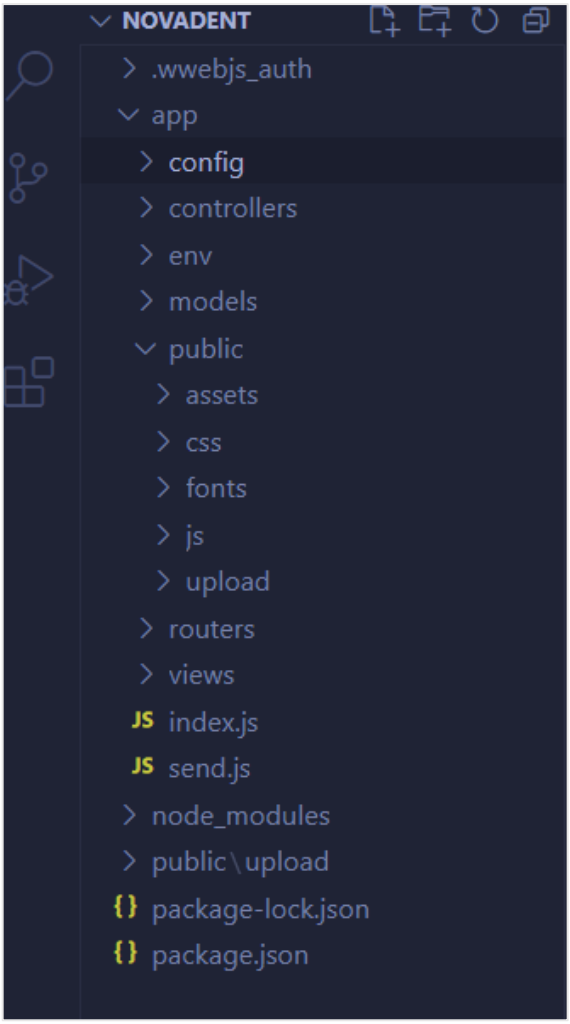


Figura 3.6: Arquitectura Backend

Elaborado por: El investigador

Packages Node.js

Los paquetes utilizados para el desarrollo de la aplicación son los siguientes:

- **Express.-** Framework web, desarrollado en JavaScript y se encuentra alojado en el entorno de Node.js, permite la conexión a la base de datos, además de gestionar eventos proveyendo un ambiente más organizado de desarrollo.

- **Motor de plantilla ejs.-** Effective JavaScript templating, usado para la generación de plantillas ejs solo con JavaScript, una vez ejecutado la aplicación la plantilla se renderiza en formato HTML de forma legible para los navegadores.
- **Bcryptjs.-** Permite la encriptación de datos.
- **mysql.-** Dependencia que cumple el rol de intermediario entre express y la base de datos, es decir permite realizar la conexión segura con MariaDB/MySQL.
- **whatsapp-web.js.-** Librería para Node.js que permite la utilización de WhatsApp Web de forma independiente al dispositivo móvil dentro del código fuente, permite la salida y recepción de mensajes.
- **qrcode-terminal.-** Permite la creación de códigos QR legibles en la terminal.
- **nodemailer.-** Es un módulo utilizado para el envío de correos.
- **multer.-** Es un middleware usado principalmente para la caga de archivos al servidor, está relacionado con el sistema de archivos pues accede a este para almacenar y leer archivos.
- **jsonwebtoken.-** Se considera un medio seguro basado en JSON para el manejo de autenticación de forma segura en las aplicaciones, garantizando que el envío de datos sea seguro.
- **body-parser.-** Middleware que analiza el cuerpo (body) de las solicitudes entrantes y los captura con un req.body para que sea más fácil interactuar con los elementos, los eventos capturados se identifican con el id de los elementos HTML.
- **node-cron.-** Librería utilizada para la creación de cron-job en Node.js, se establece como una tarea programada encargada de ejecutar una acción o evento de acuerdo con su programación y frecuencia.

Dependencias

- **Nodemon.-** Herramienta que ayuda a la gestión del sistema de archivos y la interacción con la línea de comando usando NPM, permite ejecutar sentencias

y añadir dependencias al proyecto. Además, reinicia el proceso de compilación de la app de forma automática una vez que detecta un cambio en los archivos de la aplicación.

Aplicación de la arquitectura MVC

Models.- En esta ruta se encuentran los distintos modelos de las entidades que están representados en el modelo relacional de la base de datos que se pueden apreciar en la figura 3.5, esta capa es quien tiene contacto directo con datos. Las sentencias SQL se hacen presentes y son ejecutados por los distintos métodos que se encuentran definidos, cada uno de estos representan una orden que se lo realiza desde las vistas.

Controllers.- Comprende como una de las rutas o carpetas de mayor importancia al igual que el modelo pues es esta quien decide la dirección del negocio. Para la ejecución de cada métodos se hace un llamado al modelo correspondiente, esta capa es quien actúa de intermediario entre las vistas y los modelos, además, es aquí donde se recuperan cada atributo de las vistas que representan a los campos de las distintas tablas de la base de datos, el acceso a estos se los hace a través de la palabra clave **req.body.campoSolicitidado.**

Rutas.- En esta sección se instancian cada de uno de los controladores que participan en las transacciones de datos, las salidas o entradas de estos de los hace a través de las rutas que pueden contener los siguientes métodos:

- PUT
- POST
- GET

Vistas.- En esta capa se hizo uso de la plantilla de renderización **ejs** que al momento de la ejecución de la aplicación en los navegadores automáticamente se lee como plantillas html.

Ejs, facilita la captura y recepción de los datos de que son procesados por las distintas rutas establecidas en la carpeta router.

Diseño de interfaz de usuario

Comprende cada una de las interfaces con las cuales el usuario podrá interactuar para cumplir con sus objetivos.

Tarea 3	
Nombre de tarea: Diseño de interfaz de usuario	
Tipo de Tarea: Diseño	**Puntos Estimados:** 3
Programador responsable: Esteban Chanatasig	
Descripción: Diseño de interfaz de usuario.	

Tabla 3.20: Tarea - Diseño de interfaz de usuario

Elaborado por: El investigador

Registro de administradores del aplicativo

Todos los usuarios a quienes la gerente considere factible otorgarles el rol de administrador se registrarán en la siguiente ventana, los campos son: nombres, apellidos, cédula de identidad, el correo electrónico que debe ser de uso único y un password de 10 dígitos.

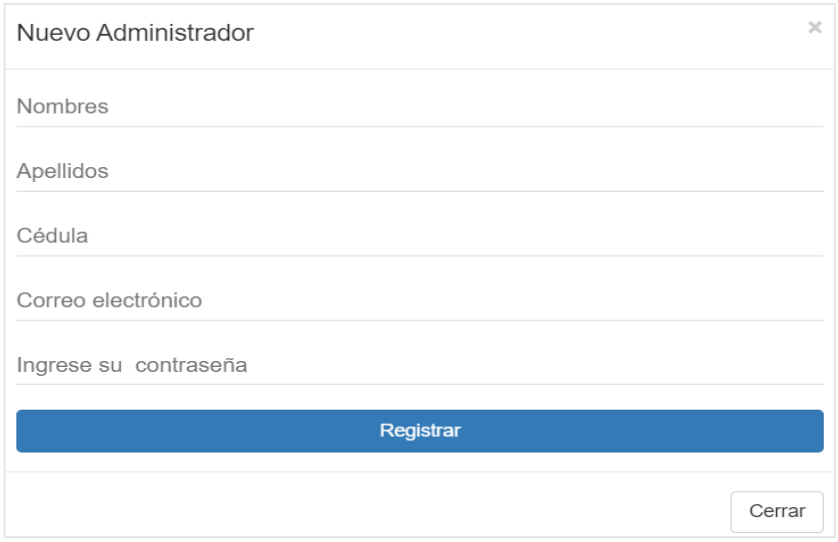


Figura 3.7: Interfaz - Registro de administradores del aplicativo

Elaborado por: El investigador

Inicio de sesión

Los administradores del aplicativo podrán iniciar sesión mediante un correo electrónico y una contraseña.

Figura 3.8: Interfaz - Inicio de sesión

Elaborado por: El investigador

Una vez iniciado sesión se mostrará el perfil con los datos básicos del administrador, se puede editar la información en caso de ser necesario.

Figura 3.9: Interfaz - Perfil de administrador

Elaborado por: El investigador

Panel de registro y consultas de turnos

Se visualizará un menú con dos opciones:

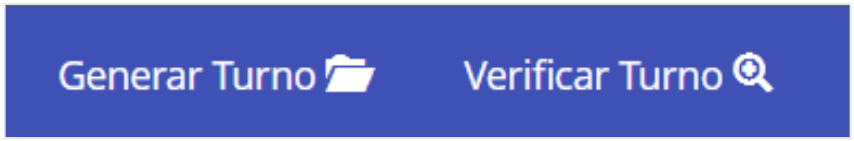


Figura 3.10: Interfaz - Panel de registro y consultas de turnos

Elaborado por: El investigador

Formulario para registro de atención

En este módulo se mostrará o registrará (según sea el caso) la información personal y médica de los pacientes que requieran un turno de atención.

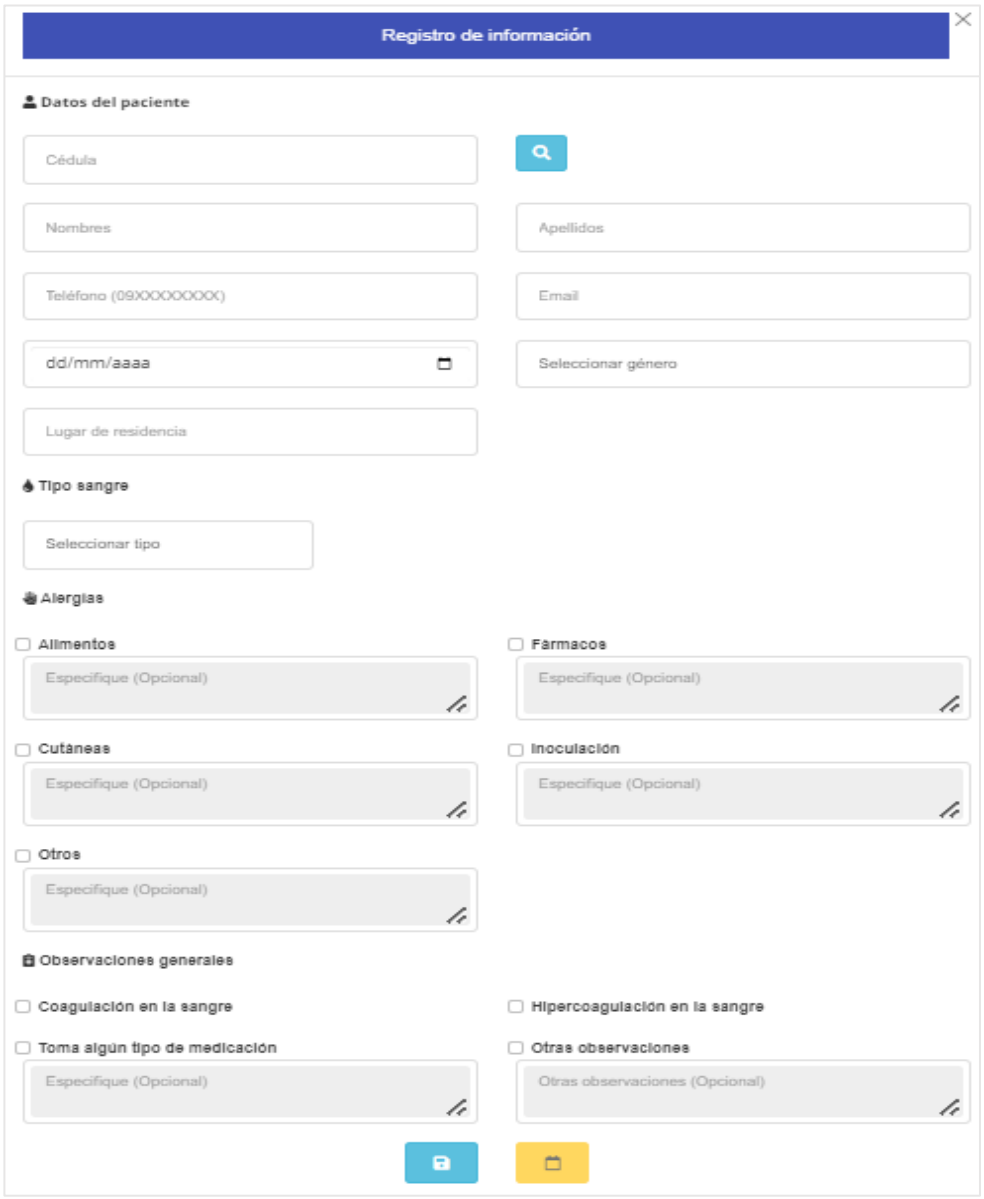


Figura 3.11:Formulario para registro de atención

Elaborado por: El investigador

Generar turno

Una vez terminado de rellenar el formulario de la figura 3.11 en se mostrará el siguiente módulo en la cual se deberá escoger las horarios disponibles dependiendo de la fecha establecida.

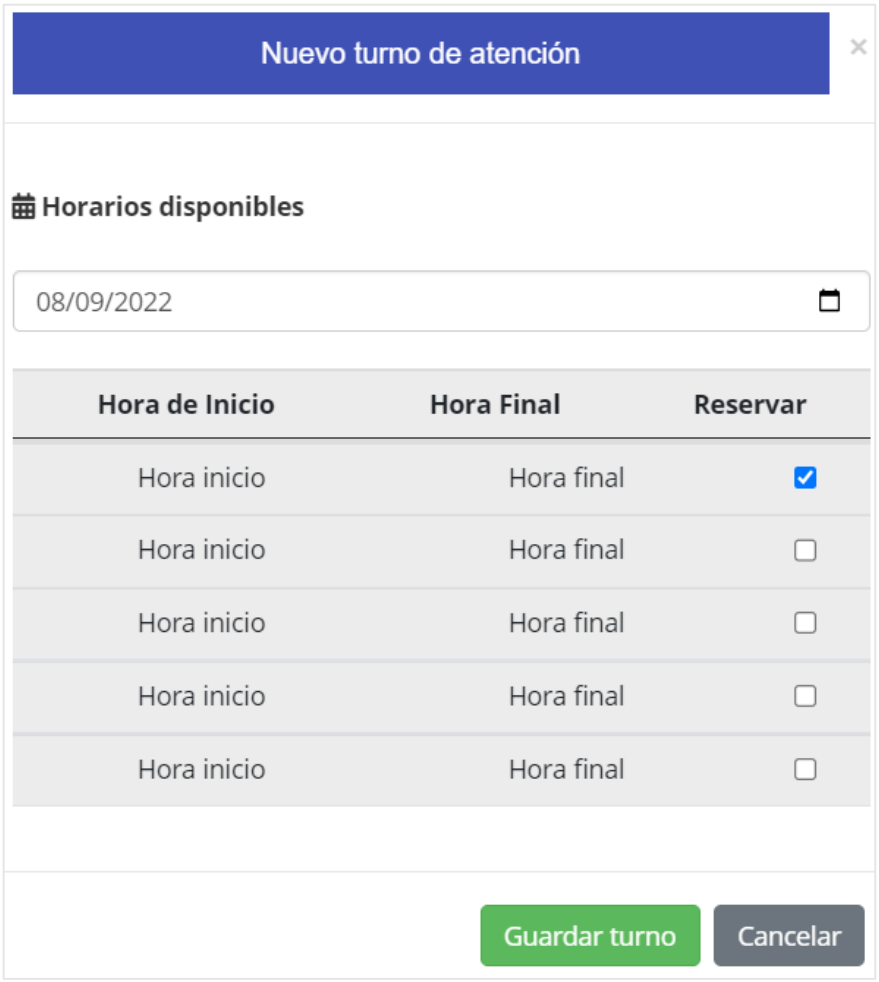


Figura 3.12: Interfaz - Generar Turno

Elaborado por: El investigador

Verificar Turno

Se podrá consultar si el paciente tiene un turno asignado para la revisión médica, en caso de requerir también se existe la posibilidad de cancelar el turno liberando el espacio de atención o reagendarlo a una fecha distinta.

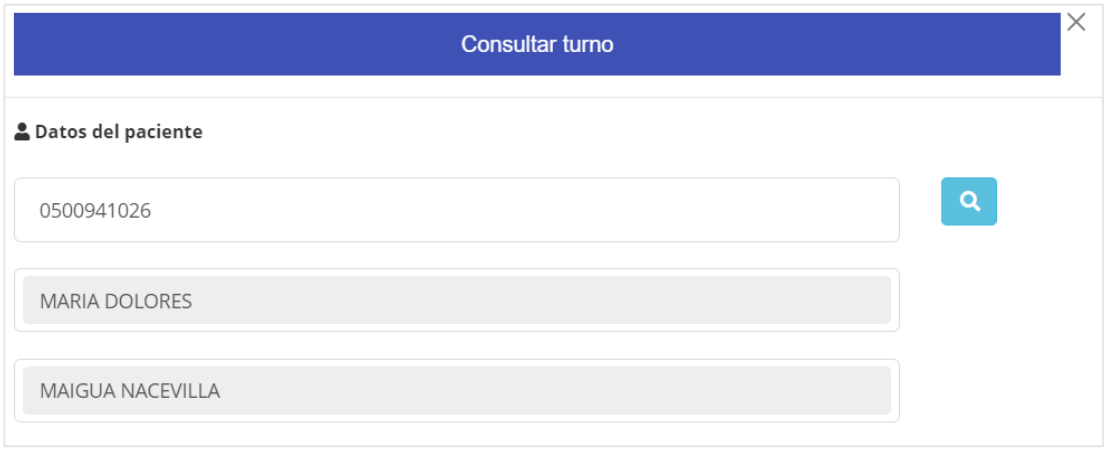


Figura 3.13: Interfaz - Verificar Turno

Elaborado por: El investigador

Figura 3.14: Interfaz - Turnos reservados

Elaborado por: El investigador

Citas de hoy

Figura 3.15: Interfaz - Modal citas de hoy

Elaborado por: El investigador

Dashboard administrativo

Una vez iniciado sesión se visualizará un panel de admistración que redirecciona a cada una de las pantallas de actividades que se pueden hacer en el aplicativo.

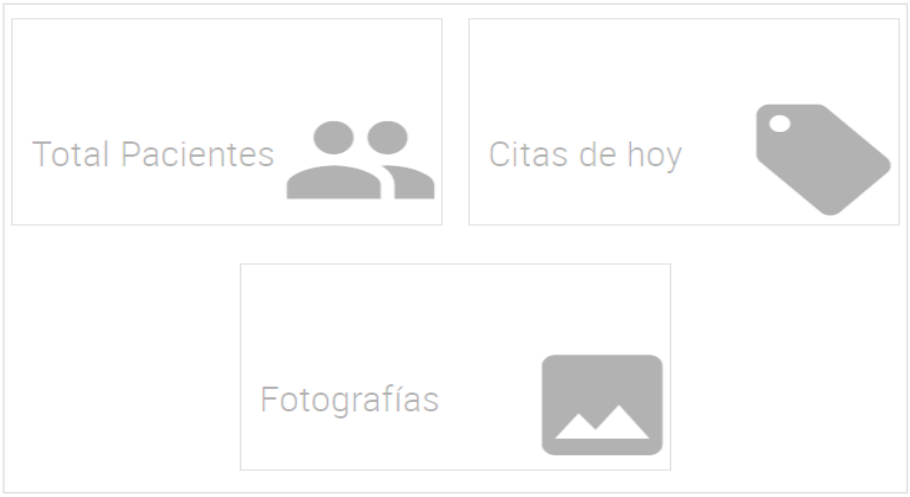


Figura 3.16: Interfaz - Dashboard administrativo

Elaborado por: El investigador

Menú navegacional lateral

Se mostrará cada una de las opciones para la admistración de las actividades como un menú lateral.

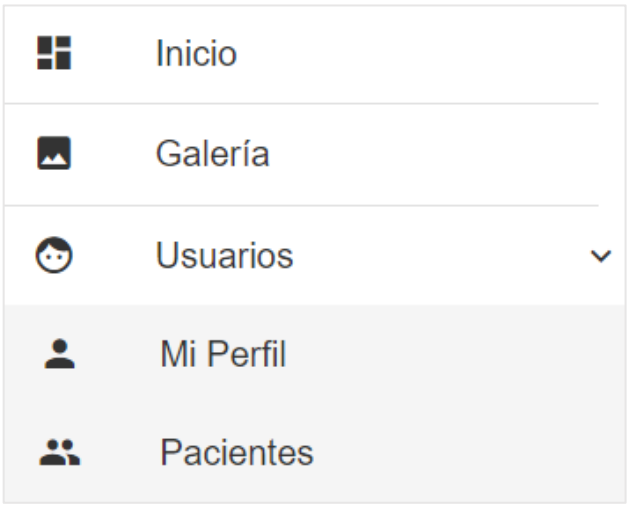


Figura 3.17: Interfaz - Menú navegacional lateral

Elaborado por: El investigador

Galería de imágenes

Se realizará un filtrado de los pacientes que se encuentran registrados en distintos tratamientos, en cada uno de ellos se muestra una opción (botón) que permitirá visualizar las imágenes de la galería de fotos.

Figura 3.18: Interfaz - Galería de imágenes

Elaborado por: El investigador

Gestión de pacientes

Listado de pacientes

En esta interfaz de usuario se podrá enlistar a todos los pacientes registrados en el centro odontológico, con la facilidad de un cuadro de búsqueda en caso de existir mucha carga de información.

De igual manera se integrarán dos botones tanto como para generar un documento PDF (Portable Document Format) que contenga toda la información del listado de pacientes, así como también un botón que generará un documento Excel.

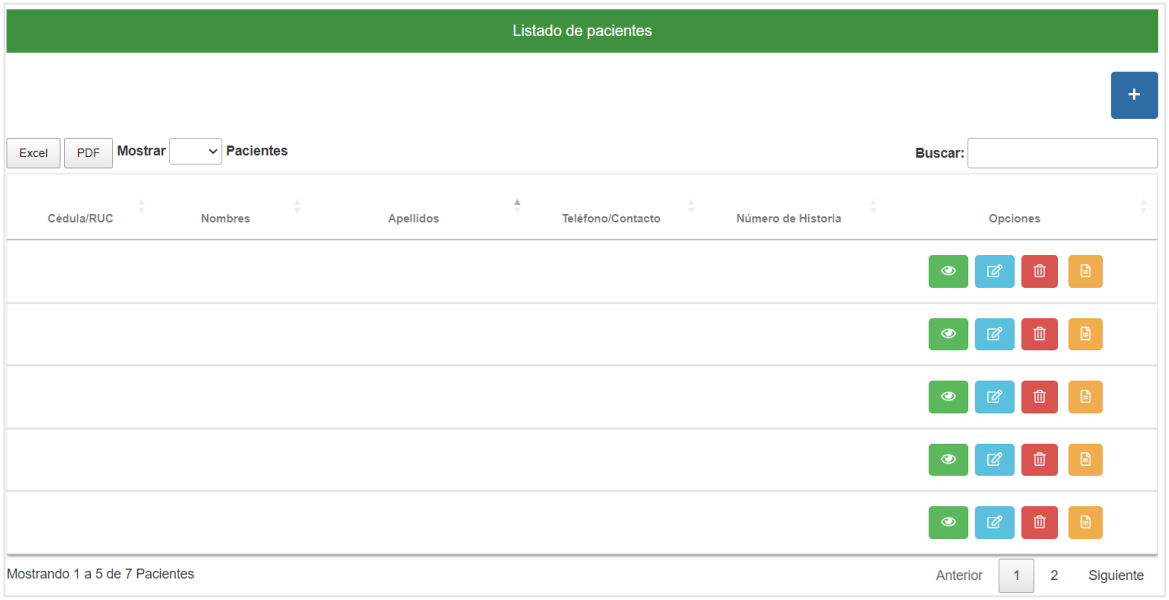


Figura 3.19: Interfaz - Listado de pacientes

Elaborado por: El investigador

Añadir nuevo paciente

Permitirá añadir nuevos pacientes con la información necesaria como se visualiza en la figura 3.20, todos los datos no son requeridos puesto que no siempre se puede contar con la información completa.

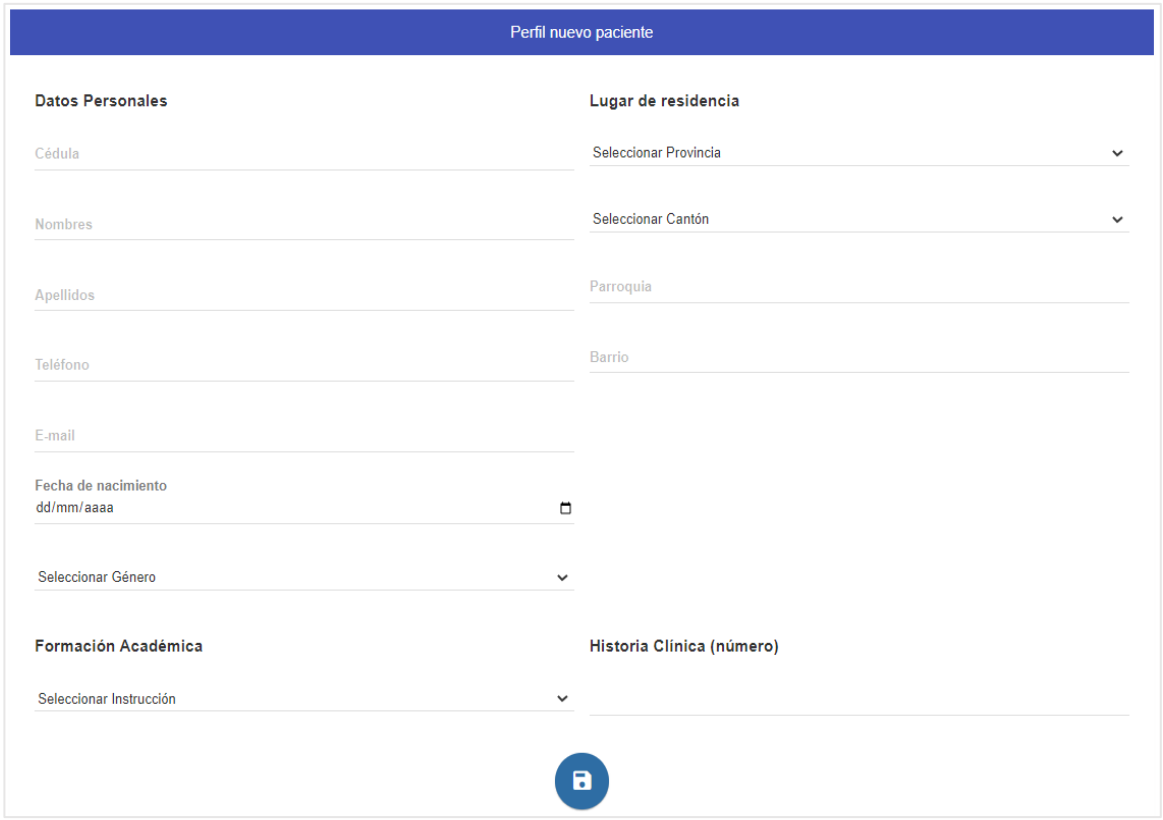


Figura 3.20: Interfaz - Añadir nuevo paciente

Elaborado por: El investigador

Editar datos del paciente

En una ventana modal se mostrará la información del paciente que está registrada en la base de datos se puede editar la información por nuevos datos a excepción del número de historia que no se podrá modificar debido a la política del centro odontológico.

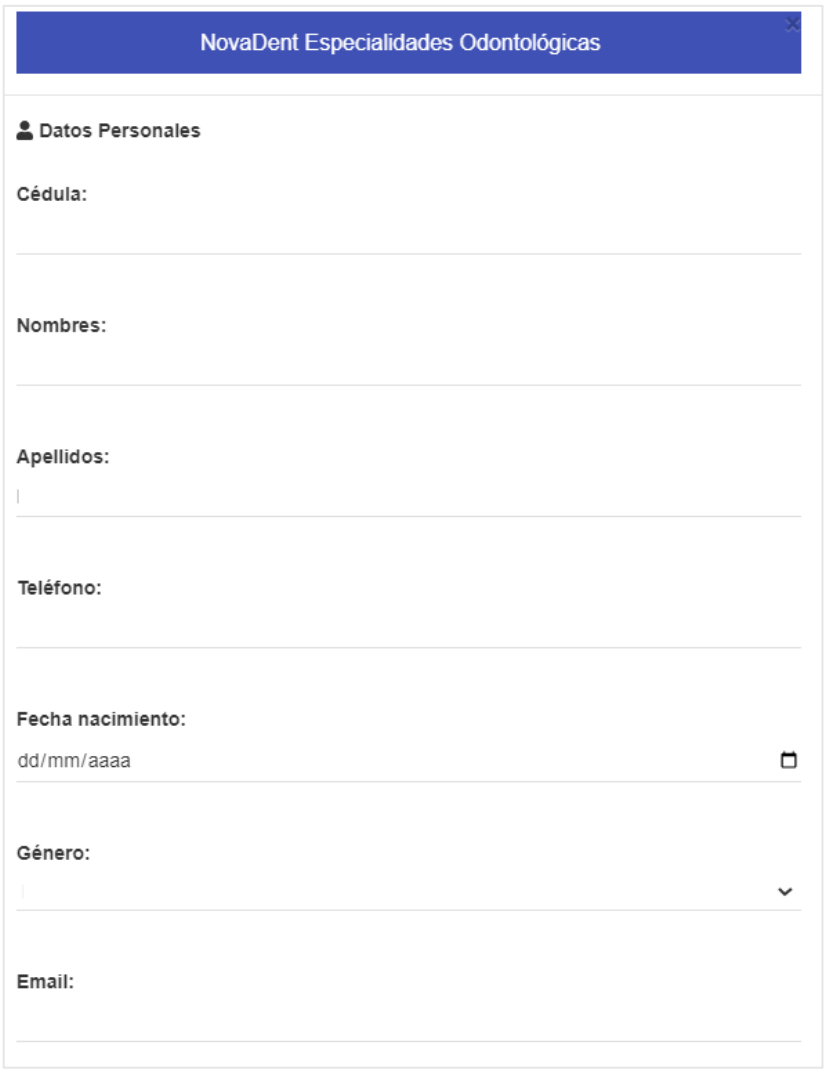


Figura 3.21: Interfaz - Editar datos personales del paciente

Elaborado por: El investigador

Figura 3.22: Interfaz - Editar lugar de residencia del paciente

Elaborado por: El investigador

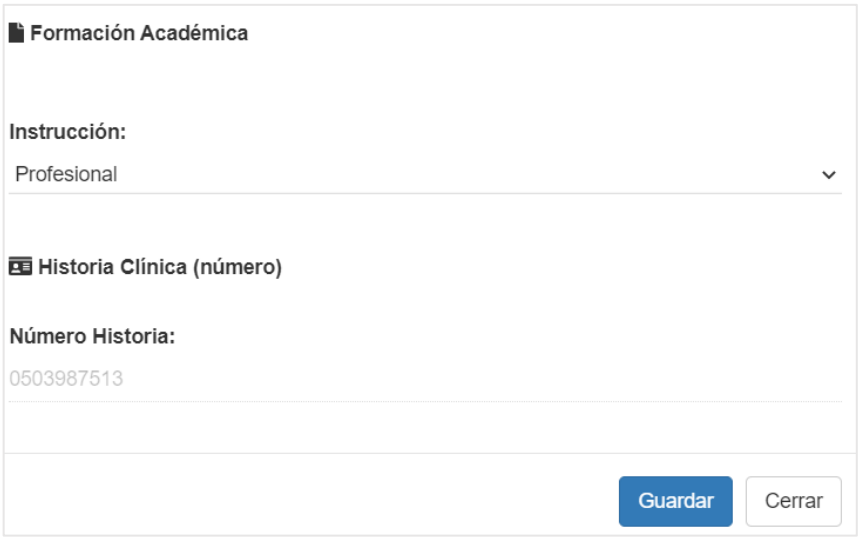


Figura 3.23: Interfaz - Editar Formación académica del paciente

Elaborado por: El investigador

Ver datos del paciente

En el siguiente módulo se podrá visualizar toda la información del paciente (datos personales, información médica y tratamientos actuales, con la posibilidad de generar un documento tipo PDF con dicha infromación.

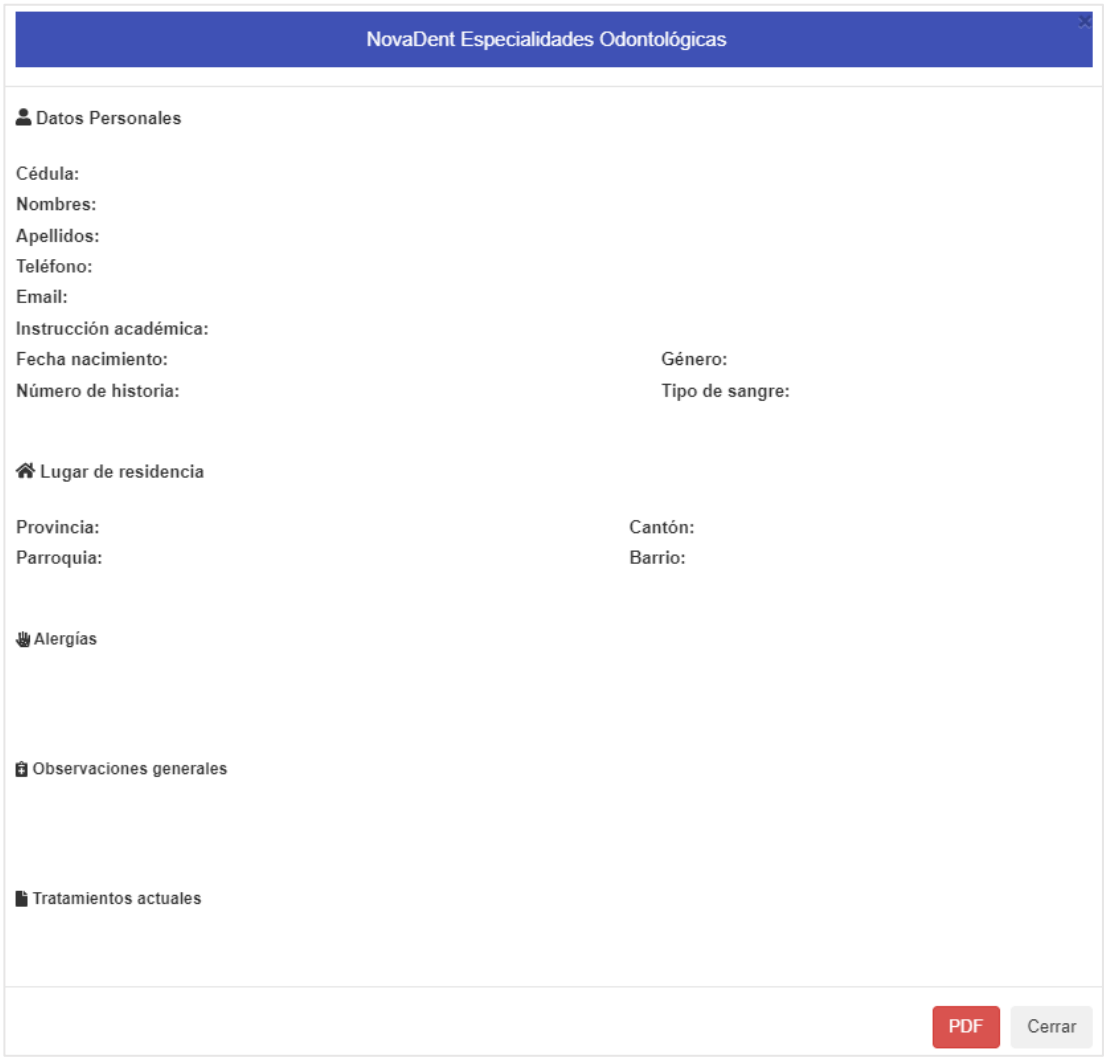


Figura 3.24: Interfaz - Generar PDF con la información del paciente

Elaborado por: El investigador

Gestión de tratamientos del paciente

Se desplegará la información del paciente además de los tratamientos actuales, si no existe ningún registro el odontólogo puede asignar un tratamiento nuevo.

Figura 3.25: Interfaz - Tratamientos de paciente

Elaborado por: El investigador

Lista de tratamientos requeridos por el centro odontológico.

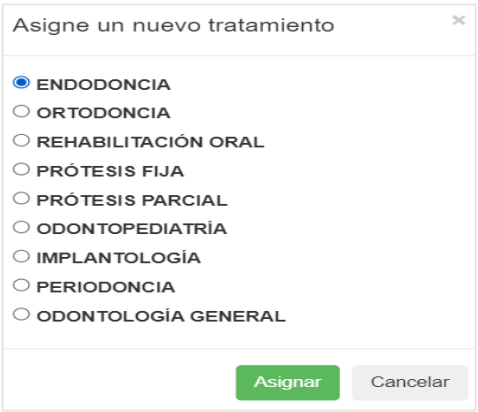


Figura 3.26: Interfaz - Lista de tratamientos requeridos por el centro odontológico

Elaborado por: El investigador

Una vez asignado un nuevo tratamiento se mostrará un listado en un panel de tratamientos, cada asignación de se muestra en forma de cuadrículas.

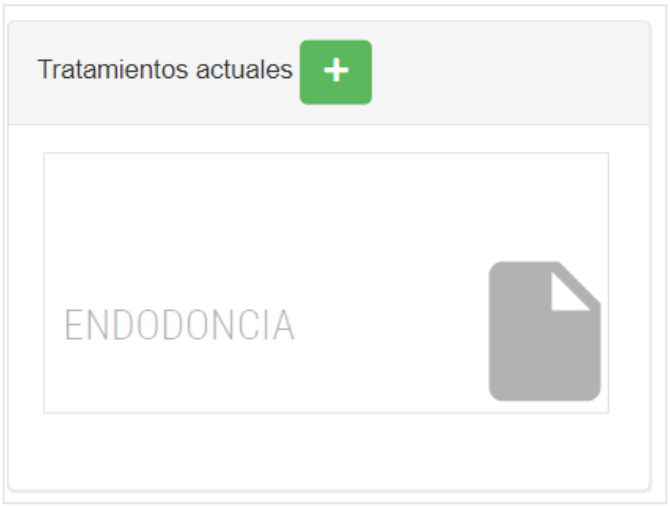


Figura 3.27: Interfaz - Tratamientos actuales del paciente

Elaborado por: El investigador

Todo tratamiento tiene observaciones realizadas por el especialistas, al seleccionar uno de estos se mostrará su información y la del paciente. En la parte inferior se desplegará la lista de observaciones ordenadas dependiendo de la fecha en la que inició el plan de tratamiento.

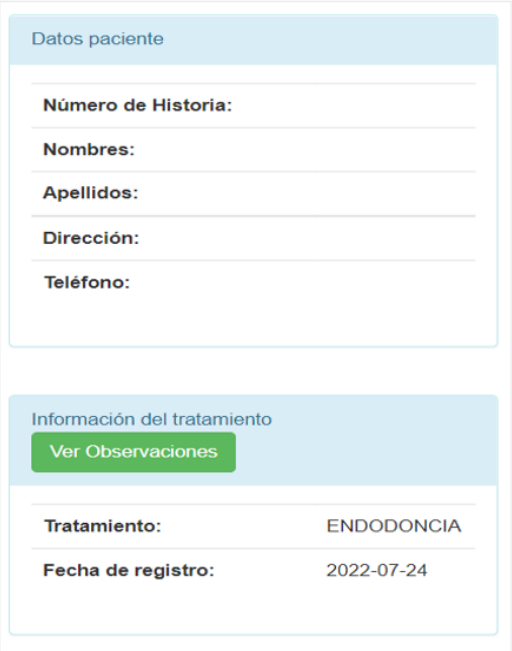


Figura 3.28: Interfaz - Observaciones de tratamientos

Elaborado por: El investigador

En caso de que no exista registro de observaciones el odontólogo tendrá la posibilidad de crear nuevas con la ayuda de una ventana modal que contiene en su cuerpo los campos necesarios.

Figura 3.29: Interfaz - Listado de observaciones según tratamiento

Elaborado por: El investigador

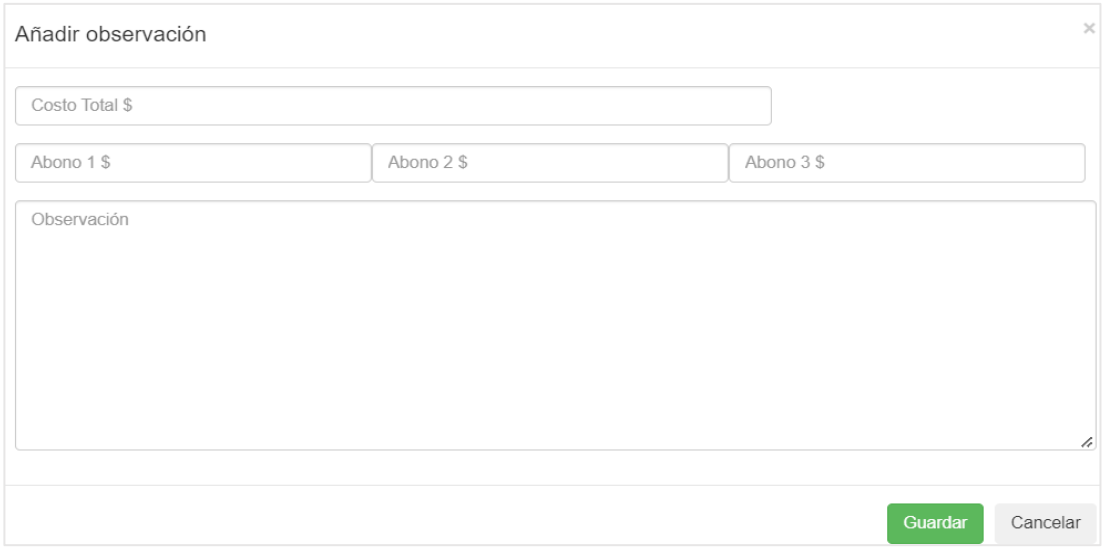


Figura 3.30: Interfaz - Modal nueva observación

Elaborado por: El investigador

Reseteo de contraseña

En el caso de la necesidad de resetear la clave de acceso se enviará un correo electrónico que contiene un link en donde se debe ingresar y resetear con una nueva contraseña.

Figura 3.31: Interfaz - Ingreso de correo de recuperación

Elaborado por: El investigador

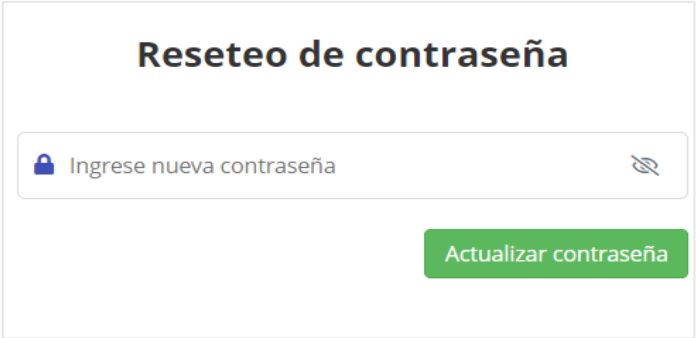


Figura 3.28: Interfaz - Ingreso de nueva contraseña

Elaborado por: El investigador

3.2.3.2 Iteración 2

Inicio de sesión

Para el ingreso al sistema se necesita de un usuario y una contraseña, previo a esto se debe realizar un registro de usuario en la cual la contraseña se encripta por razones de seguridad.

Tarea 1	
Nombre de tarea: Inicio de sesión	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 2
Programador responsable: El investigador	
Descripción: El usuario podrá ingresar al sistema una vez que se haya registrado. Se hará uso del módulo bcrypt.js para comparar la contraseña ingresada con las registradas en la base de datos en conjunto con su respectivo usuario, si existe coincidencia se permitirá el ingreso al sistema. Una vez ingresado al sistema se creará una sesión y un identificador de usuario con el módulo **req.session.loggedin** los mismos que serán usando para la destrucción de la instancia al cerrar sesión.	

Tabla 3.21: Tarea - Inicio de sesión

Elaborado por: El investigador

Reseteo de contraseña

En caso de que uno de los administradores requiera resetear su clave de acceso. Al pulsar en el enlace "¿Olvidaste tu contraseña?" ,se mostrará una ventana en la cual debe especificar un correo que deberá estar previamente registrado, a continuación se envía un correo electrónico.

Tarea 2	
Nombre de tarea: Reseteo de contraseña	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 2
Programador responsable: Esteban Chanatasig	
Descripción: Se hará uso de un token de autenticación que durará 24 horas para comprobar la validez y mantener la seguridad de los datos.	

Tabla 3.22: Tarea - Reseteo de contraseña

Elaborado por: El investigador

3.2.3.3 Iteración 3

Gestión de pacientes

La información de los pacientes se gestionará según las necesidades del administrador, en el sistema se puede realizar las siguientes acciones:

- **Añadir paciente.-** Se almacenará en la base de datos un nuevo registro, toda la información del paciente no es relevante a excepción del número de cédula de identidad o RUC, nombres y apellidos; los campos restantes no son obligatorios pero se podrán actualizar en caso de ser necesario.
- **Eliminar paciente.-** Borra un registro desde la base de datos.
- **Actualizar datos del paciente.-** Se podrá actualizar cualquiera de los campos según sea necesario.

- **Listado de pacientes.-** Muestra una lista de información de todos los pacientes que forman parte del centro odontológico sin importar si cuentan con un tratamiento actual o únicamente tienen revisiones de rutina.

Tarea 1	
Nombre de tarea: Gestión de pacientes	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Programador responsable: Esteban Chanatasig	
Descripción: La ejecución de cada uno de los eventos se los hará mediante la implementación de las interfaces relacionadas con la gestión de pacientes, se hará una validación de la cédula de identidad o RUC, además del número de teléfono y correo electrónico para evitar inconvenientes de comunicaciones.	

Tabla 3.23: Tarea - Gestión de pacientes

Elaborado por: El investigador

Galería de imágenes

El usuario podrá subir imágenes del avance de los tratamientos de los pacientes al servidor y se las almacenará en una carpeta local con el nombre de **uploads.**

Tarea 1	
Nombre de tarea: Galería de imágenes	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Programador responsable: Esteban Chanatasig	

Descripción: Se hará uso del módulo **multer** propio de express para la carga de archivos al servidor; los formatos admitidos para la subida de imágenes serán: jpg, png, svg y el límite de tamaño es 10 MB

Tabla 3.24: Tarea - Galería de imágenes

Elaborado por: El investigador

Gestión de citas

El usuario podrá elegir la fecha y los horarios disponibles para agendar un turno de atención, todo el proceso dependerá de la petición del solicitante, para la agenda de la cita se seguirá el siguiente proceso:

1. Verificar si el paciente ya tiene historial en el centro odontológico. En caso de que ya se encuentre en la base de datos los campos restantes se completaran automáticamente o de lo contrario, si se trata de un nuevo registro primero se deberá guardar la nueva información.
2. El paciente deberá proporcionar un número de teléfono con una cuenta activa en WhatsApp, en caso de no poseer uno no existirá ninguna restricción para agendar el turno.
3. Se deberá elegir una fecha de atención y verificar si existen horarios disponibles. Finalmente se guarda se guarda la solicitud.

El usuario podrá cancelar un turno de atención en caso de que el paciente lo requiera a través del siguiente proceso:

1. Deberá consultar a la base de datos mediante la interfaz de consulta de turno mostrada en la figura 3.14.
2. Se desplegarán las citas programadas, al elegir una de las opciones se podrá cancelar. Una vez cancelada el paciente recibirá un mensaje a través de WhatsApp.

Tarea 2	
Nombre de tarea: Gestión de citas	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Programador responsable: Esteban Chanatasig	
Descripción: Se hará uso de datatables para el despliegue de los horarios disponibles según la fecha seleccionada por el usuario, para los mensajes informativos de advertencia o de éxito de las solicitudes generadas se usará **sweeralert,** librería JavaScript direccionada a la creación de alertas.	

Tabla 3.25: Tarea - Gestión de citas

Elaborado por: El investigador

Gestión de tratamientos

Los pacientes que se encuentren en los registros de la base de datos podrán estar asignados en uno o varios tratamientos. El usuario podrá asignar nuevos tratamientos según lo requiera.

Cada uno de los tratamientos asignados a pacientes tendrán observaciones realizadas por el usuario con las fechas de las visitas, los pagos realizados y los avances descritas en una breve descripción.

Tarea 1	
Nombre de tarea: Gestión de tratamientos	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Programador responsable: Esteban Chanatasig	
Descripción: En el listado de los pacientes se mostrará un botón la cual nos desplegará una nueva ventana con la información de los pacientes y los tratamientos	

con los que cuenta, se podrán añadir más tratamientos si es necesario. El usuario podrá seleccionar un tratamiento y permitirá visualizar una ventana con los datos del tratamiento, del paciente y las observaciones que tiene hasta la fecha. Las nuevas observaciones se las hará con la ayuda de una ventana modal especificada en la figura 3.29 diseñada con la librería Bootstrap.

Tabla 3.26: Tarea - Gestión de tratamientos

Elaborado por: El investigador

3.2.3.4 Iteración 4

Número de historia clínica

El número de historia clínica es el identificador de cada paciente dentro del centro odontológico, según los requerimientos del usuario este identificador debe ser el mismo valor que el número de cédula.

Tarea 1	
Nombre de tarea: Número de historia clínica	
Tipo de Tarea: Control	**Puntos Estimados:** 2
Programador responsable: Esteban Chanatasig	
Descripción: Se hará uso del evento **onblur** para duplicar el número de cédula introducido en el input, al perder el foco en este elemento, el input correspondiente al número de historia contendrá el mismo valor.	

Tabla 3.27: Tarea - Número de historia clínica

Elaborado por: El investigador

Cerrar sesión

El usuario cerrará la sesión y destruirá la instancia de su identificador de ingreso.

Tarea 2	
Nombre de tarea: Cerrar sesión	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 2
Programador responsable: Esteban Chanatasig	
Descripción: Se hará uso de **req.session.destroy** para la destrucción de la instancia de ingreso de usuario. Se limpia el caché y se evita la navegación a páginas visitadas anteriormente.	

Tabla 3.28: Tarea - Cerrar sesión

Elaborado por: El investigador

3.2.3.5 Iteración 5

Notificaciones WhatsApp

El envío de mensajes WhatsApp de notificaciones se hará de manera automática ya sea al agendar o cancelar un turno de atención.

Tarea 1	
Nombre de tarea: Notificaciones WhatsApp	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 2
Programador responsable: Esteban Chanatasig	
Descripción: Para el envío de mensaje de WhatsApp se hará uso de la librería **whatsapp-web.js** que proporcionar el ambiente necesario para la captura de eventos relacionadas con la mensajería de la red social.	

Tabla 3.29: Tarea - Notificaciones WhatsApp

Elaborado por: El investigador

Notificaciones WhatsApp de recordatorio

El envío automático de mensajes de recordatorio se lo hará de forma automática con la ayuda de la programación de una tarea programada.

Tarea 2	
Nombre de tarea: Notificaciones WhatsApp de recordatorio	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 2
Programador responsable: Esteban Chanatasig	
Descripción: Se realizará una solitud a la base de datos con la información acerca de los pacientes que tienen turnos registrados para el día siguiente, con la ayuda de la librería **node-cron** se procederá a crear una tarea programada para el envío automático de mensajes WhatsApp con un recordatorio.	

Tabla 3.30: Tarea - Notificaciones WhatsApp de recordatorio

Elaborado por: El investigador

3.2.4 Fase 4: Codificación

3.2.4.1 Configuración del servidor Node.js

Para la configuración del servidor se hace uso de express para la importación de módulos, a continuación se declara una variable **app** como se visualiza en la figura en la cual creamos una nueva instancia de la librería express.

Todos las importaciones locales y externas, declaración de variables, uso de librerías y middlewares se los define en el archivo **index.js** que básicamente se considera el núcleo de la aplicación; es decir, al iniciar el aplicativo primero se carga el archivo de manera global.

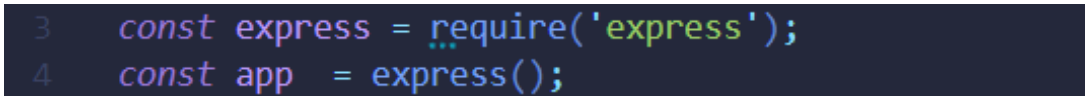


Figura 3.32: Importación del módulo express

Elaborado por: El investigador

El puerto 3000 que normalmente es utilizado en sistemas basados en Node.js y express está definido como el puerto de salida de la aplicación y se establece la conexión con el servidor.

Figura 3.33: Definición de puerto

Elaborado por: El investigador

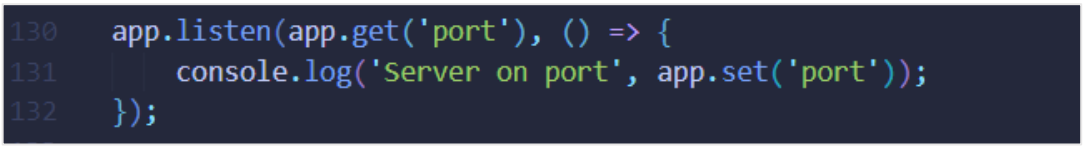


Figura 3.34: Establecer conexión con el servidor

Elaborado por: El investigador

3.2.4.2 Conexión a la base de datos

Para la creación de una conexión a la base de datos MariaDB se establecen los parámetros de autenticación como lo son: host, puerto 3306 (generalmente utilizado por el gestor de base de datos actual), database, user y password y a su vez se capturan los errores en caso de fallo de la conexión.

```
const mysql = require('mysql');

const conexion = mysql.createConnection({
    host: 'localhost',
    port: 3306,
    database: 'novadent',
    user: 'root',
    password: 'novadent2022',
});




conexion.connect(function (error) {
    if (error) {
        throw error;
    }else{
        console.log('conectado');
    }
});

module.exports = conexion;
```

Figura 3.35: Conexión con el gestor de base de datos MariaDB

Elaborado por: El investigador

Se requiere la importación de la conexión de a la base de datos en el archivo principal del aplicativo (index.js).

```
const con = require("./config/db.js")

app.use(function(req, res, next) {
    req.con = con
    next()
  })
```

Figura 3.36: Importación de la conexión a la base de datos

Elaborado por: El investigador

3.2.4.3 Middlewares

Los middlewares mostrados en la figura 3.34 se definen con la finalidad de interpretar las peticiones realizadas por los clientes hacia el servidor.

```
101   app.use(express.urlencoded({extended: true}));
102   app.use(bodyParser.json());
103   app.use(bodyParser.urlencoded({ extended: true }));
104   app.use(flash());
```

Figura 3.37: Definición de Middlewares

Elaborado por: El investigador

3.2.4.4 Archivos Estáticos

Se definen los directorios de la figura 3.35 con el fin de que el aplicativo tenga acceso a sus recursos (vistas, hojas de estilos, archivos js, imágenes, etc); todo esto debido a que al inicializar express los archivos estáticos son establecidos por defecto por lo que es necesario la declaración de directorios propios.

```
108   app.use(express.static(path.join(__dirname,'public')));
109   app.use(express.static(path.join(__dirname,'views')));
```

Figura 3.38: Definición de archivos estáticos

Elaborado por: El investigador

3.2.4.5 Instancia de las rutas

Se hace una referencia a las carpetas **routers** y **passRouter** quienes contienen las rutas con los métodos (PUT, GET, POST) que se ejecutan en las distintas vistas según el tipo de solicitud; es necesario declarar las líneas de la figura en el archivo index.js que es el núcleo de la aplicación.

```
app.use(require('./routers/routers'));
app.use(require('./routers/passRouter'));
app.use(require('./routers/imagenesRouter'));
app.use(require('./routers/diagnosticoRouter'));
```

Figura 3.39: Referencia a las rutas

Elaborado por: El investigador

3.2.4.6 Inicio de sesión

Creación de la función **auth** de ingreso al sistema.

```
function auth(req,res) {
    const data = req.body;
    mysql.query('select * from admins where user = ?' , [data.user], (err, userData)=>{
        if (userData.length > 0) {
            userData.forEach(element => {
            bcryptjs.compare(data.pass, element.pass,(err, isMatch)=>{

                        if (!isMatch) {
                          var error = 'Password Incorrecto';
                          res.render('index', {error: error});
                        } else {
                           req.session.loggedin = true;
                           req.session.name = element;
                           res.redirect('/home');
                        }
                    });

            });

        } else {
            var error = 'Usuario no existe';
            res.render('index', {error: error});
        }

    });
}
```

Figura 3.40: Creación de la función auth para el ingreso al sistema

Elaborado por: El investigador

Declaración del método en la ruta.

```
61    router.post('/login', authControllerIngreso.auth);
62
```

Figura 3.41: Declaración del método en la ruta.

Elaborado por: El investigador

3.2.4.7 Envío de correo electrónico

Protocolo SMT (Simple Mail Transfer Protocol).

```
function sendEmail(email, token) {
   var email = email;
   var token = token;
   var mail = nodemailer.createTransport({
   //service: 'gmail',
   host: "smtp.gmail.com",
   port: 465,
   secure: true,
   requireTLS: true,
   auth: {
     user: "novadentEO.ec@gmail.com",
     pass: "gpnxcyygmdvjckza",
   },
   tls:{
      rejectUnauthorized:false
  }
   });
   var mailOptions = {
   from: 'novadentEO.ec@gmail.com',
   to: email,
   subject: 'Resetear contraseña de acceso - novadent-ec.com',
   html: '<p>Acceda al siguiente <a href="http://localhost:3000/reset-password?token=' +
   token + '">link</a> para resetear su contraseña </p>'
   };
   mail.sendMail(mailOptions, function(error, info) {
   if (error) {
   console.log(error);
   } else {
   console.log(0)
   }
   });
   } //fin function
```

Figura 3.42: Protocolo SMT (Simple Mail Transfer Protocol)

Elaborado por: El investigador

Envío de correo electrónico para el reseteo de contraseña.

```
router.post('/reset-password-email', function(req, res, next) {
  var email = req.body.email;
  con.query('SELECT * FROM admins WHERE user ="' + email + '"', function(err, result) {
  if (err) throw err;
  if (result.length > 0) {
     if (result[0].user.length > 0) {
     var token = randtoken.generate(20);
     var sent = sendEmail(email, token);
     if (sent != '0') {
     var data = { token: token};
     con.query('UPDATE admins SET ? WHERE user ="' + email + '"', data, function(err, result) {
     if(err) throw err
     })
     } else {
        console.log('Ocurrió un error');

     }
     }

}else{
   res.render('passwordReset/send-email', {title: 'Correo de verificación' ,
   message: 'Usuario no registrado, comuniquese con el administrador del sistema!'})

}
   res.redirect('/login');
   });
   })
```

Figura 3.43: Envío de correo electrónico para el reseteo de contraseña

Elaborado por: El investigador

Reseteo de contraseña.

```
router.post('/update-password', function(req, res, next) {
var token = req.body.token;
var pass = req.body.password;
con.query('SELECT * FROM admins WHERE token ="' + token + '"', function(err, result) {
if (err) throw err;
var type
var msg
if (result.length > 0) {
var saltRounds = 10;
bcryptjs.genSalt(saltRounds, function(err, salt) {
bcryptjs.hash(pass, salt, function(err, hash) {
var data = {
pass: hash
}
con.query('UPDATE admins SET ? WHERE user ="' + result[0].user + '"', data, function(err, result) {
if(err) throw err
});
});
});
type = 'success';
msg = 'Your password has been updated successfully';
} else {
console.log('2');
type = 'success';
msg = 'Invalid link; please try again';
}
req.flash(type, msg);
res.redirect('/login');
});
}) //fin metodo
```

Figura 3.44: Reseteo de contraseña

Elaborado por: El investigador

3.2.4.8 Gestión de pacientes

En las rutas con los diferentes métodos (PUT, GET , POST) se declaran las funciones creadas en los controladores.

```
const mysql = require("../config/db");
//constructor
const PacienteModelo = function (paciente) {
    this.cedula_paciente = paciente.cedula_paciente;
    this.nombres_paciente = paciente.nombres_paciente;
    this.apellidos_paciente = paciente.apellidos_paciente;
    this.telefono_paciente = paciente.telefono_paciente;
    this.fecha_nacimiento = paciente.fecha_nacimiento;
    this.email_paciente = paciente.email_paciente;
    this.provincia_paciente = paciente.provincia_paciente;
    this.canton_paciente = paciente.canton_paciente;
    this.parroquia_paciente = paciente.parroquia_paciente;
    this.barrio_paciente = paciente.barrio_paciente;
    this.genero_paciente = paciente.genero_paciente;
    this.instruccion_paciente = paciente.instruccion_paciente;
    this.numero_historia = paciente.numero_historia;
  };
```

Figura 3.45: Importación de la conexión a la base de datos y creación del constructor PacienteModelo

Elaborado por: El investigador

Agregar nuevo paciente.

```
PacienteModelo.create = (nuevoPaciente, result) => {
  mysql.query("INSERT INTO paciente SET ?", nuevoPaciente, (err, res) => {
    if (err) {
      console.log("error: ", err);
      result(err, null);
      return;
    }
    result(null, { cedula: res.cedula_paciente, nuevoPaciente});
  });
};
```

Figura 3.46: Capa Modelo - Sentencia SQL para la inserción de un nuevo paciente

Elaborado por: El investigador

```
exports.create = (req, res) => {
  if (!req.body) {
      res.status(400).send({
      message: "Content can not be empty!"
      });
  }
  const nuevoPaciente = new PacienteModelo({
    cedula_paciente: req.body.paciente_cedula,
    nombres_paciente:  req.body.paciente_nombre,
    apellidos_paciente: req.body.paciente_apellido,
    telefono_paciente: req.body.paciente_telefono,
    fecha_nacimiento: req.body.paciente_fecha,
    email_paciente: req.body.paciente_email,
    provincia_paciente: req.body.paciente_provincia,
    canton_paciente: req.body.paciente_canton,
    parroquia_paciente: req.body.paciente_parroquia,
    barrio_paciente: req.body.paciente_barrio,
    genero_paciente: req.body.paciente_genero,
    instruccion_paciente: req.body.paciente_instruccion,
    numero_historia: req.body.paciente_numeroHistoria
  });

  PacienteModelo.create(nuevoPaciente, (err, paciente) => {
      if (err)
      res.status(500).send({ message:  err.message || "Error al crear nuevo paciente" });
      else
      res.redirect('/pacientes');
    });
};
```

Figura 3.47: Capa Controlador - Definición del método crear

Elaborado por: El investigador

```
router.post('/nuevoPaciente', pacienteController.create);
```

Figura 3.48: Definición del método POST en las rutas con la función crear

Elaborado por: El investigador

Eliminar registro de paciente.

```
PacienteModelo.delete = function(cedula, result){
  mysql.query("DELETE FROM paciente WHERE cedula_paciente = ?", [cedula], function (err, res) {
    if(err) {
        console.log("error: ", err);
        result(null, err);
    }
    else{
        result(null, res);
    }
  });
};
```

Figura 3.49: Capa Modelo - Sentencia SQL para la eliminación de un registro de paciente

Elaborado por: El investigador

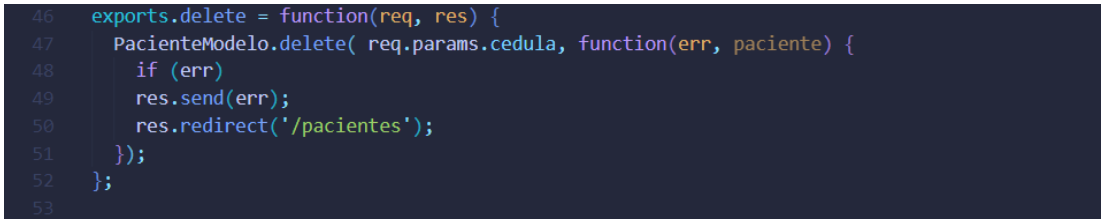


Figura 3.50: Capa Controlador - Definición del método delete

Elaborado por: El investigador

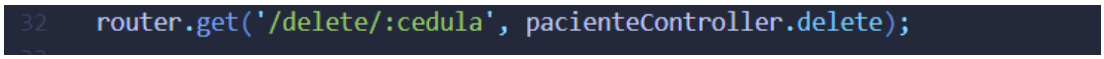


Figura 3.51: Definición del método GET en las rutas con la función delete

Elaborado por: El investigador

Actualizar datos del paciente.

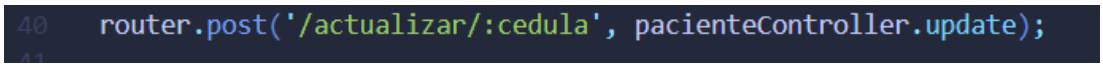
Figura 3.52: Capa Modelo - Sentencia SQL para la actualizar datos de un paciente

Elaborado por: El investigador

```
exports.update = function(req, res) {
  if(req.body.constructor === Object && Object.keys(req.body).length === 0){
      res.status(400).send({ error:true, message: 'Datos vacios' });
  }else{
    const actualizarPaciente = new PacienteModelo({
      cedula_paciente: req.body.cedula_paciente,
      nombres_paciente: req.body.nombres_paciente,
      apellidos_paciente: req.body.apellidos_paciente,
      telefono_paciente: req.body.telefono_paciente,
      fecha_nacimiento: req.body.fecha_nacimiento,
      email_paciente: req.body.email_paciente,
      provincia_paciente: req.body.provincia_paciente,
      canton_paciente: req.body.canton_paciente,
      parroquia_paciente: req.body.parroquia_paciente,
      barrio_paciente: req.body.barrio_paciente,
      genero_paciente: req.body.genero_paciente,
      instruccion_paciente: req.body.instruccion_paciente,
      numero_historia: req.body.numero_historia
     });
      PacienteModelo.update(req.body.cedula_paciente, actualizarPaciente, function(err, data) {
        if (err){
          if (err.kind === 'no_found ') {
            res.status(400).send({ message: `Not found paciente with id ${req.params.cedula_paciente}.`  });
            console.log('me quedo en el 400');
          }else
          {
            res.status(500).send({message: "Error updating paciente with cedula " + req.params.cedula_paciente   });
            console.log({actualizarPaciente});
            console.log('me quedo en el 500');
          }

        }else
        res.send(data);
        });
  }
};
```

Figura 3.53: Capa Controlador - Definición del método update

Elaborado por: El investigador

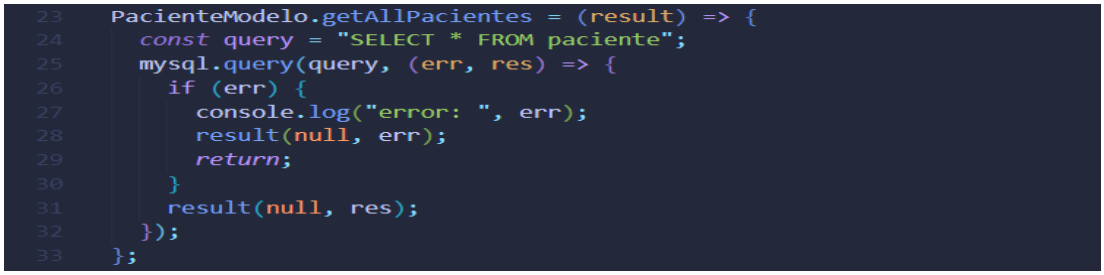


```
router.post('/actualizar/:cedula', pacienteController.update);
```

Figura 3.54: Definición del método POST en las rutas con la función update

Elaborado por: El investigador

Listado de pacientes.



```
PacienteModelo.getAllPacientes = (result) => {
  const query = "SELECT * FROM paciente";
  mysql.query(query, (err, res) => {
    if (err) {
      console.log("error: ", err);
      result(null, err);
      return;
    }
    result(null, res);
  });
};
```

Figura 3.55: Capa Modelo - Sentencia SQL para listar todos los pacientes

Elaborado por: El investigador

```
exports.getAllPacientes = (req, res) => {
  PacienteModelo.getAllPacientes(function (err, paciente) {
    if (err) {
        throw err;
    } else {

      if (req.session.loggedin == true) {
        res.render('pacientes', {title:'Listado', pacientes: paciente , user:req.session.name});
      } else {
        res.redirect('/login');
      }
     }
  });
};
```

Figura 3.56: Capa Controlador - Definición del método getAllPacientes

Elaborado por: El investigador

```
router.get('/especialistas',pacienteController.getAllPacientes);
```

Figura 3.57: Definición del método GET en las rutas con la función getAllPacientes

Elaborado por: El investigador

3.2.4.9 Galería de imágenes

Subida de archivos al servidor

El middleware multer figura permite la subida de archivos (imágenes) al directorio local upload haciendo uso del método POST, de igual forma se hace una inserción en la tabla **paciente_ruta** con los datos capturados desde el cuerpo de la solicitud.

```
var multer = require('multer');
const path = require('path');
const con = require("../config/db.js");
var storage = multer.diskStorage({
  destination: function (req, file, cb) {
     cb(null, 'app/public/upload');
  },
  filename: function (req, file, cb) {
    cb(null,`${file.originalname}`);
 }
});
var upload = multer({ storage: storage });
router.post("/upload", upload.array('file'), (req, res) => {
    if (!req.files) {
        console.log("No file upload");
   } else {
        for (let index = 0; index < req.files.length; index++) {
            let date = new Date();
            let day = `${(date.getDate())}`.padStart(2,'0');
            let month = `${(date.getMonth()+1)}`.padStart(2,'0');
            let year = date.getFullYear();
            var imgsrc = req.files[index].filename;
            var id_paciente = req.body.id_paciente;
            var fecha_captura = `${year}-${month}-${day}`
            var id_tratamiento = req.body.id_tratamiento;
            var sql  = "INSERT INTO `paciente_ruta`(`id_paciente`,`fecha_captura`, `ruta`,`id_tratamiento`) VALUES  ('" +
             id_paciente + "',   '" +  fecha_captura + "', '" + imgsrc + "','" +  id_tratamiento + "')";
            con.query(sql, function (err, result){
                if (err) throw err
            })
        }
        res.redirect('back');
    }
  });
```

Figura 3.58: Definición del método POST y declaración de la carga de archivo a través del módulo multer

Elaborado por: El investigador

3.2.4.10 Gestión de citas

```
const mysql = require("../config/db");
const HorarioModelo = function (horario) {
    this.id_horario = horario.id_horario;
    this.fecha_horario = horario.fecha_horario;
    this.hora_inicio = horario.hora_inicio;
    this.hora_final = horario.hora_final;
    this.estado = horario.estado;
  };
```

Figura 3.59: Importación de la conexión a la base de datos y creación del constructor Horario-Modelo

Elaborado por: El investigador

Limitar subida de archivos a 10 MB.

```
var bodyParser = require('body-parser');
app.use(bodyParser.json({limit: '10mb'}));
app.use(bodyParser.urlencoded({limit: '10mb', extended: true}));
```

Figura 3.60: Capacidad de tamaño de carga de imágenes

Elaborado por: El investigador

Obtener horarios disponibles

```
HorarioModelo.findByFecha = function (fecha, result) {
 var estado= 0 ;
 mysql.query("SELECT * FROM horario WHERE fecha_horario = ? AND estado = ? AND hora_inicio >  DATE_FORMAT(NOW( ), '%H:%i')",
  [fecha, estado],   function (err, res) {
   if (err) {
     console.log("error: ", err);
     result(err, null);
   } else {
     //console.log(res);
     result(null, res);
   }
  });
};
```

Figura 3.61: Capa Modelo - Sentencia SQL para obtener los horarios disponibles según su cláusula WHERE

Elaborado por: El investigador

```
exports.findByFecha = function (req, res) {
  HorarioModelo.findByFecha(req.params.fecha, function (err, data) {
      if (err){
        res.send(err);
      }else{
        res.json(data);
      }
    });
  };
```

Figura 3.62: Capa Controlador - Definición del método findByFecha

Elaborado por: El investigador

```
47   router.get('/consultar/:fecha', horarioController.findByFecha);
48
49   router.get('/consultarNuevo/:fecha', horarioController.findByFecha);
```

Figura 3.63: Definición del método GET en las rutas con la función findByFecha

Elaborado por: El investigador

Reservar turno

```
const mysql = require("../config/db");

//constructor

const TurnoModel = function (turnoModel) {
    this.id_paciente = turnoModel.id_paciente;
    this.id_horario = turnoModel.id_horario;
    this.fecha_registrada = turnoModel.fecha_registrada;
  };
```

Figura 3.64: Importación de la conexión a la base de datos y creación del constructor TurnoModel

Elaborado por: El investigador

```
14    TurnoModel.create = ( nuevoTurno, result) => {
15      mysql.query("INSERT INTO cita SET ?", nuevoTurno, (err, res) => {
16        if (err) {
17          console.log("error: ", err);
18          result(err, null);
19          return;
20        }
21        result(null, { id_paciente: res.id_paciente, nuevoTurno});
22      });
23  };
```

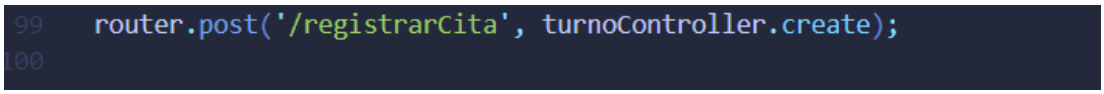
Figura 3.65: Capa Modelo - Sentencia SQL para insertar un nuevo turno

Elaborado por: El investigador

```
var date = new Date()
var day = `${(date.getDate())}`.padStart(2,'0');
var month = `${(date.getMonth()+1)}`.padStart(2,'0');
var year = date.getFullYear();
var fecha_captura = `${year}-${month}-${day}`

exports.create = (req, res) => {
    if (!req.body) {
        res.status(400).send({
        message: "Content can not be empty!"
        });
    }
    const nuevoTurno = new TurnoModel({
      id_paciente: req.body.id_paciente,
      id_horario: req.body.id_horario,
      fecha_registrada: fecha_captura,
    });

    TurnoModel.create(nuevoTurno, (err, data) => {
        if (err)
        res.status(500).send({
            message:
            err.message || "Error al crear nuevo paciente"
        });

        else
        res.json(data);
      });
  };
```

Figura 3.66: Capa Controlador - Definición del método create

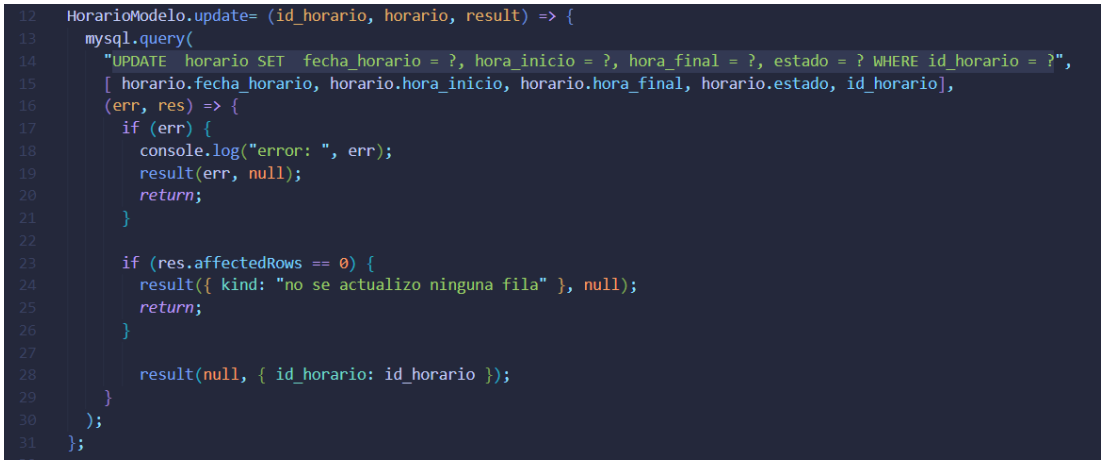
Elaborado por: El investigador



```
router.post('/registrarCita', turnoController.create);

```

Figura 3.67: Definición del método POST en las rutas con la función create

Elaborado por: El investigador



```
HorarioModelo.update= (id_horario, horario, result) => {
  mysql.query(
    "UPDATE  horario SET  fecha_horario = ?, hora_inicio = ?, hora_final = ?, estado = ? WHERE id_horario = ?",
    [ horario.fecha_horario, horario.hora_inicio, horario.hora_final, horario.estado, id_horario],
    (err, res) => {
      if (err) {
        console.log("error: ", err);
        result(err, null);
        return;
      }

      if (res.affectedRows == 0) {
        result({ kind: "no se actualizo ninguna fila" }, null);
        return;
      }

        result(null, { id_horario: id_horario });
    }
  );
};
```

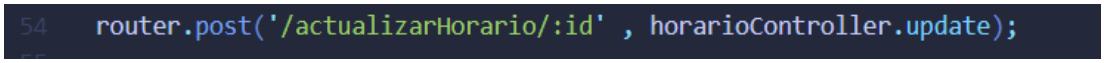
Figura 3.68: Capa Modelo - Sentencia SQL para actualizar el estado del horario a reservado

Elaborado por: El investigador

```
exports.update = function( req , res) {
   if(req.body.constructor === Object && Object.keys(req.body).length === 0){
       res.status(400).send({ error:true, message: 'Datos vacios' });
   }else{

     const nuevoEstado = new HorarioModelo({
       id_horario: req.body.id_horario,
       fecha_horario: req.body.fecha_horario,
       hora_inicio: req.body.hora_inicio,
       hora_final: req.body.hora_final,
       estado: req.body.estado

   });


       HorarioModelo.update(req.body.id_horario, nuevoEstado, function(err, data) {
           if (err){
             if (err.kind === 'no_found ') {
               res.status(400).send({ message: `Not found horario with id ${req.body.id_horario}.`  });
               console.log('me quedo en el 400');

             }else {
               res.status(500).send({message: "Error updating horario with id " + req.body.id_horario});
               console.log({nuevoEstado});
               console.log('me quedo en el 500');
             }
           }else
           res.json(data);
         });
   }
 };
```

Figura 3.69: Capa Controlador - Definición del método update

Elaborado por: El investigador

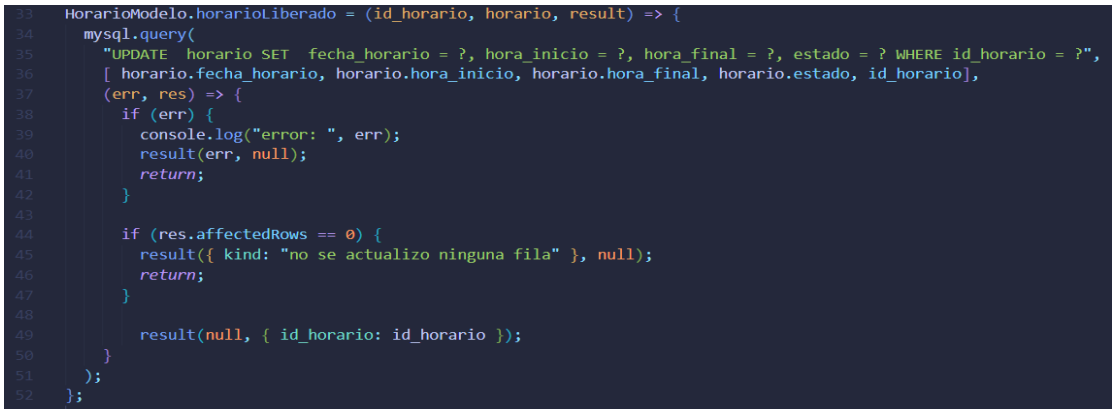


```
router.post('/actualizarHorario/:id' , horarioController.update);
```

Figura 3.70: Definición del método POST en las rutas con la función update

Elaborado por: El investigador

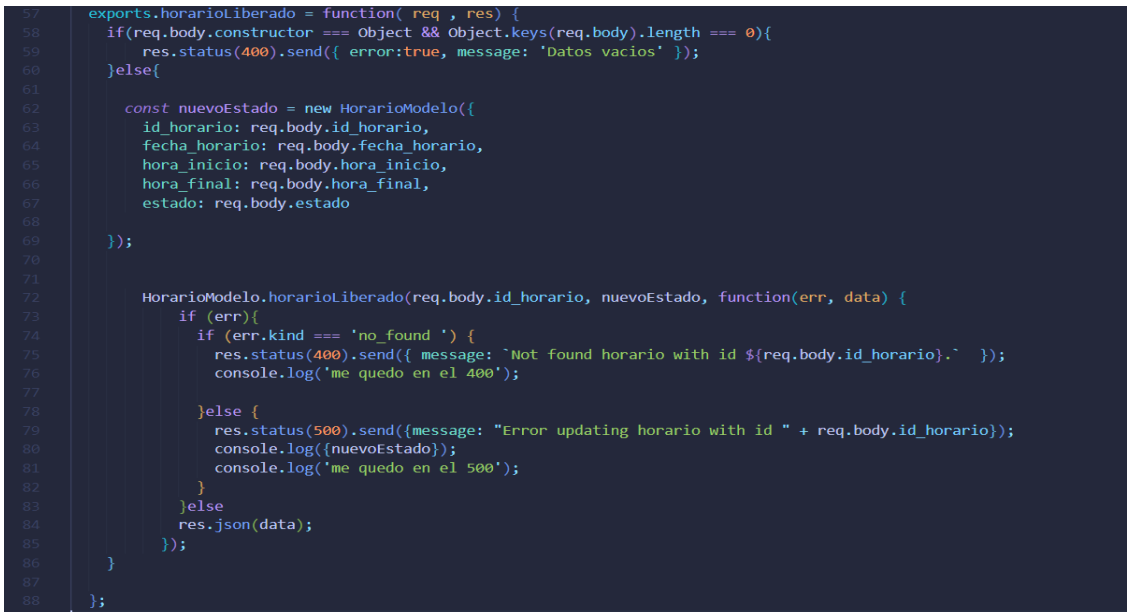
Cancelar turno



```
HorarioModelo.horarioLiberado = (id_horario, horario, result) => {
  mysql.query(
    "UPDATE  horario SET  fecha_horario = ?, hora_inicio = ?, hora_final = ?, estado = ? WHERE id_horario = ?",
    [ horario.fecha_horario, horario.hora_inicio, horario.hora_final, horario.estado, id_horario],
    (err, res) => {
      if (err) {
        console.log("error: ", err);
        result(err, null);
        return;
      }

      if (res.affectedRows == 0) {
        result({ kind: "no se actualizo ninguna fila" }, null);
        return;
      }

        result(null, { id_horario: id_horario });
    }
  );
};
```

Figura 3.71: Capa Modelo - Sentencia SQL para actualizar el estado de un horario ha habilitado

Elaborado por: El investigador



```
exports.horarioLiberado = function( req , res) {
  if(req.body.constructor === Object && Object.keys(req.body).length === 0){
      res.status(400).send({ error:true, message: 'Datos vacios' });
  }else{

    const nuevoEstado = new HorarioModelo({
      id_horario: req.body.id_horario,
      fecha_horario: req.body.fecha_horario,
      hora_inicio: req.body.hora_inicio,
      hora_final: req.body.hora_final,
      estado: req.body.estado

  });


      HorarioModelo.horarioLiberado(req.body.id_horario, nuevoEstado, function(err, data) {
          if (err){
            if (err.kind === 'no_found ') {
              res.status(400).send({ message: `Not found horario with id ${req.body.id_horario}.`  });
              console.log('me quedo en el 400');

            }else {
              res.status(500).send({message: "Error updating horario with id " + req.body.id_horario});
              console.log({nuevoEstado});
              console.log('me quedo en el 500');
            }
          }else
          res.json(data);
        });
  }

};
```

Figura 3.72: Capa Controlador - Definición del método horarioLiberado

Elaborado por: El investigador

```
router.post('/actualizarTurno/:id_horario', horarioController.horarioLiberado);
```

Figura 3.73: Definición del método POST en las rutas con la función horarioLiberado

Elaborado por: El investigador

```
CitaPaciente.delete = function(id_horario, cedula_paciente,result){
  mysql.query("DELETE C FROM cita AS C INNER JOIN paciente AS P ON C.id_paciente = P.id_paciente WHERE C.id_horario = ? AND"+
  " P.cedula_paciente = ?", [ id_horario, cedula_paciente], function (err, res) {
     if(err) {
         console.log("error: ", err);
         result(null, err);
     }
     else{
         result(null, res);
     }
  });
};
```

Figura 3.74: Capa Modelo - Sentencia SQL para eliminar una cita

Elaborado por: El investigador

```
exports.delete = function(req, res) {
  CitaPaciente.delete( req.params.id_horario,req.params.cedula_paciente, function(err, data) {
    if (err){
      res.send(err);
    } else{
      res.json(data);
    }
  });
};
```

Figura 3.75: Capa Controlador - Definición del método delete

Elaborado por: El investigador

```
router.get('/borrarCita/:id_horario-:cedula_paciente', citaController.delete);
```

Figura 3.76: Definición del método GET en las rutas con la función delete

Elaborado por: El investigador

3.2.4.11 Gestión de tratamiento médico

```
const mysql = require("../config/db");
const { json } = require("body-parser");

//constructor
  const TratamientoPacienteModelo = function (tratamiento){
    this.id_paciente = tratamiento.id_paciente;
    this.id_tratamiento= tratamiento.id_tratamiento;
    this.fecha_registro = tratamiento.fecha_registro;
  }
```

Figura 3.77: Importación de la conexión a la base de datos y creación del constructor TratamientoPacienteModelo

Elaborado por: El investigador

```
const mysql = require("../config/db");
const { json } = require("body-parser");

//constructor
  const ObservacionTratamientoModelo = function (observacion){
    this.id_paciente = observacion.id_paciente;
    this.id_tratamiento= observacion.id_tratamiento;
    this.costo = observacion.costo;
    this.abono1 = observacion.abono1;
    this.abono2 = observacion.abono2;
    this.abono3 = observacion.abono3;
    this.fecha_observacion = observacion.fecha_observacion;
    this.observacion = observacion.observacion;
  }
```

Figura 3.78: Importación de la conexión a la base de datos y creación del constructor ObservacionTratamientoModelo

Elaborado por: El investigador

Asignación de tratamientos a pacientes

```
TratamientoPacienteModelo.create = (nuevoPacienteTratamiento, result) => {
mysql.query("INSERT INTO paciente_tratamiento SET ?", nuevoPacienteTratamiento, (err, res) => {
  if (err) {
    console.log("error: ", err);
    result(err, null);
    return;
  }
  result(null, { tratamiento: nuevoPacienteTratamiento});
 });
};
```

Figura 3.79: Capa Modelo - Sentencia SQL para asignar un tratamiento a un paciente

Elaborado por: El investigador

```
exports.create = (req, res) => {
  if (!req.body) {
      res.status(400).send({
      message: "Content can not be empty!"
      });
  }
  const nuevoPacienteTratamiento = new TratamientoPacienteModelo({
    id_paciente : req.body.id_paciente,
    id_tratamiento: req.body.id_tratamiento,
    fecha_registro: req.body.fecha_registro

  });

  TratamientoPacienteModelo.create(nuevoPacienteTratamiento, (err, data) => {
      if (err)
      res.status(500).send({
          message:
          err.message || "Error al crear nuevo paciente-tratamiento"
      });

      else
      res.json(data);

    });
};
```

Figura 3.80: Capa Controlador - Definición del método create

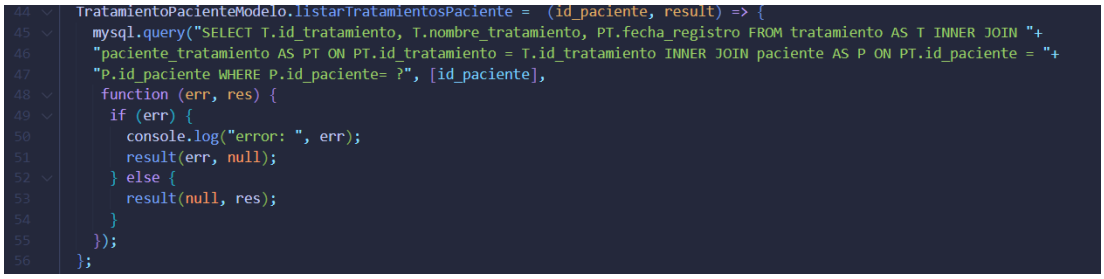
Elaborado por: El investigador

```
router.post('/nuevoTratamientoPaciente', tratamientoPacienteController.create);

```

Figura 3.81: Definición del método POST en las rutas con la función create

Elaborado por: El investigador

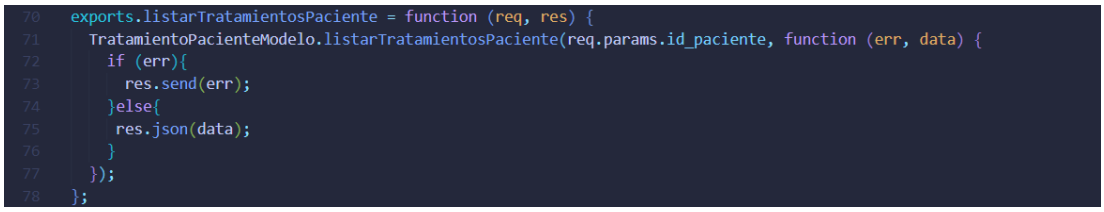
Listar tratamientos según pacientes



```
TratamientoPacienteModelo.listarTratamientosPaciente =  (id_paciente, result) => {
  mysql.query("SELECT T.id_tratamiento, T.nombre_tratamiento, PT.fecha_registro FROM tratamiento AS T INNER JOIN "+
  "paciente_tratamiento AS PT ON PT.id_tratamiento = T.id_tratamiento INNER JOIN paciente AS P ON PT.id_paciente = "+
  "P.id_paciente WHERE P.id_paciente= ?", [id_paciente],
   function (err, res) {
    if (err) {
      console.log("error: ", err);
      result(err, null);
    } else {
      result(null, res);
    }
  });
};
```

Figura 3.82: Capa Modelo - Sentencia SQL para listar los tratamientos según el paciente

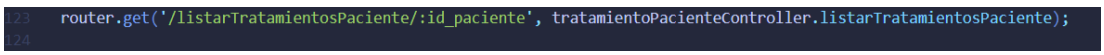
Elaborado por: El investigador



```
exports.listarTratamientosPaciente = function (req, res) {
  TratamientoPacienteModelo.listarTratamientosPaciente(req.params.id_paciente, function (err, data) {
    if (err){
      res.send(err);
    }else{
     res.json(data);
    }
  });
};
```

Figura 3.83: Capa Controlador - Definición del método listarTratamientosPacientes

Elaborado por: El investigador



```
router.get('/listarTratamientosPaciente/:id_paciente', tratamientoPacienteController.listarTratamientosPaciente);
```

Figura 3.84: Definición del método GET en las rutas con la función listarTratamientosPacientes

Elaborado por: El investigador

Asignación de observaciones a tratamientos

```
ObservacionTratamientoModelo.create = (nuevaObservacion, result) => {
mysql.query("INSERT INTO paciente_tratamiento_observacion SET ?", nuevaObservacion, (err, res) => {
  if (err) {
    console.log("error: ", err);
    result(err, null);
    return;
  }
  result(null, { observacion: nuevaObservacion});
});
};
```

Figura 3.85: Capa Modelo - Sentencia SQL para crear una nueva observación

Elaborado por: El investigador

```
exports.create = (req, res) => {
  if (!req.body) {
      res.status(400).send({
      message: "Content can not be empty!"
      });
  }
  const nuevaObservacion = new ObservacionTratamientoModelo({
    id_paciente : req.body.id_paciente,
    id_tratamiento: req.body.id_tratamiento,
    costo: req.body.costo,
    abono1: req.body.abono1,
    abono2: req.body.abono2,
    abono3: req.body.abono3,
    fecha_observacion: req.body.fecha_observacion,
    observacion: req.body.observacion

  });

  ObservacionTratamientoModelo.create(nuevaObservacion, (err, data) => {
      if (err)
      res.status(500).send({
          message:
          err.message || "Error al crear nuevo paciente-tratamiento"
      });
      else
      res.json(data);

    });
};
```

Figura 3.86: Capa Controlador - Definición del método create

Elaborado por: El investigador

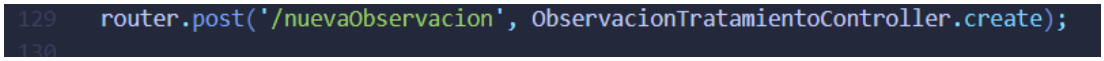


Figura 3.87: Definición del método POST en las rutas con la función create

Elaborado por: El investigador

Listar las observaciones de los tratamientos por paciente

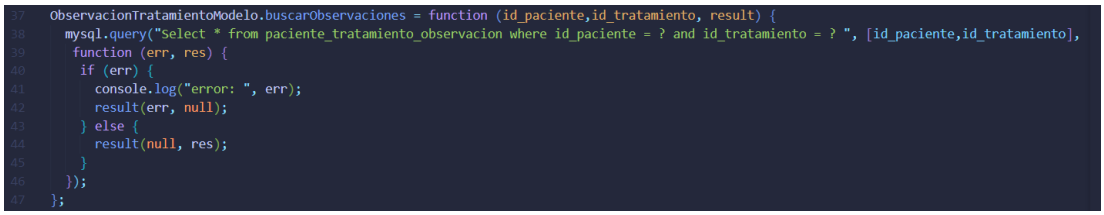


Figura 3.88: Capa Modelo - Sentencia SQL par listas todas las observaciones hechas a un tratamiento

Elaborado por: El investigador

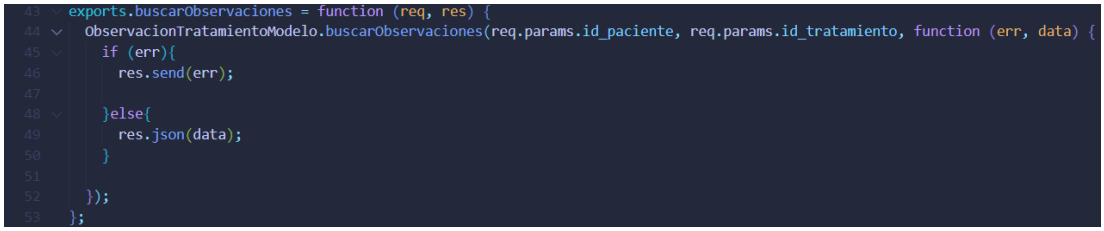


Figura 3.89: Capa Controlador - Definición del método buscarObservaciones

Elaborado por: El investigador

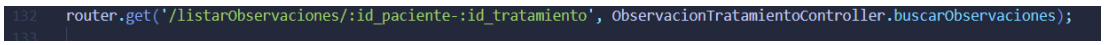


Figura 3.90: Definición del método GET en las rutas con la función buscarObservaciones

Elaborado por: El investigador

3.2.4.12 Número de historia clínica

```
var x = document.getElementById("nuevoPaciente");
x.addEventListener("blur", myBlurFunction, true);

function myBlurFunction() {
   var cedula= document.getElementById("paciente_cedula").value;
   var numeroH = document.getElementById("paciente_numeroHistoria");

   numeroH.value = cedula;
}
```

Figura 3.91: Uso del método onblur para la duplicación del valor del campo paciente_cedula

Elaborado por: El investigador

3.2.4.13 Notificaciones WhatsApp

Inicio de sesión

Para la creación del inicio de sesión del cliente WhatsApp existen dos posibilidades.

1. El cliente se registra por primera vez. Se debe guardar la sesión en un archivo tipo JSON en un directorio con el nombre **.wwebjs_auth** para futuras conexiones.

   Para generar un código QR (Quick Response) para el registro del cliente WhatsApp se hace uso de la librería **qrcode-terminal.** Una vez terminado el registro la sesión se guarda en el directorio .wwebjs_auth.

Figura 3.92: Generación del código QR y creación del cliente

Elaborado por: El investigador

2. El cliente se encuentra registrado por lo que es necesario instanciar su sesión del archivo JSON para inicializar la sesión.

```
var sessionData;
let SESSION_FILE_PATH = '.wwebjs_auth';
if (fs.existsSync(SESSION_FILE_PATH)) {
    sessionData = SESSION_FILE_PATH;
}

const client = new Client({
authStrategy: new LocalAuth({
    clienId:'Cliente1',
    dataPath:SESSION_FILE_PATH
})
});

client.on("qr", qr => {
    qrcode.generate(qr,{small:true});
});

client.on('authenticated', () => {
    if (sessionData) {
        console.log('bienvenido de nuevo');
    }
});

client.on('ready', () => {
    console.log('Client is ready!');

});
```

Figura 3.93: Creacion del inicio de sesión de un cliente WhatsAPP

Elaborado por: El investigador

Registro de turno de atención

```
app.post('/enviarWhatsNuevoTurno', async (req, res, next) => {
    try {
      const number = req.body.number;
      const message  = req.body.message;

      client.isRegisteredUser(`${number}@c.us`).then(function(isRegistered) {
        if(isRegistered) {
          client.sendMessage(`${number}@c.us`, message);
          //res.send({ msg });
        }else{
          //res.json();
          console.log('numero no registrado');
        }
    })
    } catch (error) {
      next(error);
    }
  });
```

Figura 3.94: Envío de mensaje WhatsApp al paciente al registrar un turno

Elaborado por: El investigador

Cancelación de turno de atención.

```
app.post('/enviarWhatsCancelarTurno', async (req, res, next) => {
  try {
    const number = req.body.number;
    const message  = req.body.message;

    client.isRegisteredUser(`${number}@c.us`).then(function(isRegistered) {
      if(isRegistered) {
        client.sendMessage(`${number}@c.us`, message);
        //res.send({ msg });
      }else{
        //res.json();
        console.log('numero no registrado');
      }
  })
  } catch (error) {
    next(error);
  }
});
```

Figura 3.95: Envío de mensaje WhatsApp al paciente al cancelar un turno

Elaborado por: El investigador

Notificaciones automáticas

```
class Main{
 static  async enviarNotificaciones(){
   const query = "SELECT P.nombres_paciente, P.apellidos_paciente, P.telefono_paciente, H.fecha_horario, H.hora_inicio, H.hora_final"+
   " FROM cita AS C INNER JOIN paciente AS P ON C.id_paciente = P.id_paciente INNER JOIN horario  AS H ON C.id_horario = H.id_horario";
   con.query(query, (err, rows) => {
     if (err) {
       console.log("error: ", err);
     }
     if (rows.length > 0) {

       for (let index = 0; index < rows.length; index++) {
         try {
           const number = '593'+rows[index].telefono_paciente.slice(1);
           const message = 'Buen día estimado/a ' + rows[index].nombres_paciente + ' ' + rows[index].apellidos_paciente +
           ' se le recuerda que su cita está registrada a las  ' + rows[index].fecha_horario+  ' de ' + rows[index].hora_inicio +
           ' se le recomienda puntualidad - NovaDent-Especialidades Odontológicas.';
           client.isRegisteredUser(`${number}@c.us`).then(function(isRegistered) {
            if(isRegistered) {
             client.sendMessage(`${number}@c.us`, message);
             console.log(number);
            }else{
              console.log('numero no registrado');
            }
           })
         } catch (error) {
           console.log(error)
         }
       }
     }

   });
 }
}
```

Figura 3.96: Envío automático de mensajes WhatsApp a todos los pacientes con turnos al día siguiente

Elaborado por: El investigador

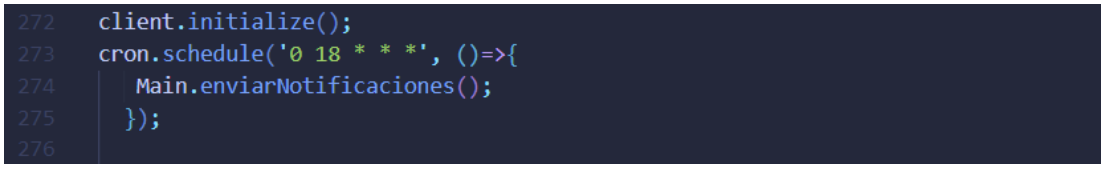


Figura 3.97: Uso de cron-job para programar envios de mensajes WhatsApp automáticos

Elaborado por: El investigador

3.2.5 Fase 5: Pruebas

3.2.5.1 Pruebas de aceptación

Pruebas de Aceptación	
Número: 1	**Historia de usuario:** 1
Nombre: Ingreso al sistema	
Descripción: El usuario debe ingresar su correo y contraseña para poder ingresar al sistema, en caso de que la clave de acceso o el usuario no estén registrados o estén erradas se mostrarán mensajes de feedback.	
Condiciones de ejecución: El usuario debe ser el administrador del sistema.	
Entrada: Se debe proporcionar un usuario y contraseña previamente registrado como administrador.	
Resultado Esperado: Ingreso exitoso al sistema	
Evaluación de prueba: Satisfactoria	

Tabla 3.31: Prueba de aceptación - Ingreso al sistema

Elaborado por: El investigador

Pruebas de Aceptación	
Número: 2	**Historia de usuario:** 2
Nombre: Reseteo de contraseña	

Descripción: El usuario debe solicitar el cambio de contraseña y rellenar el campo del correo electrónico, una vez que reciba el correo de petición de reseteo de contraseña deberá ingresar al link proporcionado e ingresar una nueva clave, si la petición tuvo éxito se redirigirá al inicio de sesión o de lo contrario se mostrará mensajes feedback.
Condiciones de ejecución: El usuario debe ser el administrador del sistema.
Entrada: Rellenar el campo requerido del correo electrónico para el envío de email de petición de cambio de contraseña.
Resultado Esperado: Obtener nueva contraseña
Evaluación de prueba: Satisfactoria

Tabla 3.32: Prueba de aceptación - Reseteo de contraseña

Elaborado por: El investigador

Pruebas de Aceptación	
Número: 3	**Historia de usuario:** 3
Nombre: Gestión de pacientes	
Descripción: El usuario puede añadir, actualizar o eliminar datos de los pacientes según sea necesario. Al añadir un nuevo registro no todos los campos serán requeridos.	
Condiciones de ejecución: El usuario debe ser el administrador del sistema.	
Entrada: Rellenar los campos requeridos para el ingreso de un nuevo paciente, actualizar los campos que se necesiten y eliminar la tupla requerida por el usuario.	
Resultado Esperado: Manipulación de información de los pacientes.	
Evaluación de prueba: Satisfactoria	

Tabla 3.33: Prueba de aceptación - Gestión de pacientes

Elaborado por: El investigador

Pruebas de Aceptación	
Número: 4	**Historia de usuario:** 4
Nombre: Galería de imágenes	

Descripción: El usuario debe elegir un paciente del listado dependiendo del tratamiento seleccionado de las opciones, abrir el espacio de carga de imágenes y subir el archivo deseado. Se mostrará las imágenes subidas de forma cronológica.
Condiciones de ejecución: El usuario debe ser el administrador del sistema.
Entrada: Se debe cargar una imagen tipo jpg, png o svg no mayor a 10 MB.
Resultado Esperado: Las imágenes se almacenan y se visualizan según el paciente y el tratamiento en el que se encuentra.
Evaluación de prueba: Satisfactoria

Tabla 3.34: Prueba de aceptación - Galería de imágenes

Elaborado por: El investigador

Pruebas de Aceptación	
Número: 5	**Historia de usuario:** 5
Nombre: Gestión de citas	
Descripción: El usuario puede desplegar los horarios disponibles dependiendo de la fecha solicitada por el paciente y asignar un turno de atención, podrá cancelar un turno de atención si el paciere lo necesita.	
Condiciones de ejecución: El usuario debe ser el administrador del sistema.	
Entrada: Se debe elegir una fecha y seleccionar uno o varios horarios marcando los checkbox. Para cancelar un turno de atención el usuario debe consultar por número de cédula del paciente y listar todos los turnos asignados, marcar los checkbox requeridos y cancelar.	
Resultado Esperado: Se registra una nueva cita, si se requiere la cita se cancelará.	
Evaluación de prueba: Satisfactoria	

Tabla 3.35: Prueba de aceptación - Gestión de citas

Elaborado por: El investigador

Pruebas de Aceptación	
Número: 6	**Historia de usuario:** 6
Nombre: Gestión de tratamiento médico	
Descripción: El usuario puede asignar un nuevo tratamiento a un paciente seleccionado del listado, se mostrarán los tratamientos con los cuales consta actualmente. Se muestra un listado de observaciones según el tratamiento del paciente en orden cronológico.	
Condiciones de ejecución: El usuario debe ser el administrador del sistema.	
Entrada: Se debe seleccionar un tratamiento de la lista de opciones.	
Resultado Esperado: El usuario puede asignar tratamientos.	
Evaluación de prueba: Satisfactoria	

Tabla 3.36: Prueba de aceptación - Gestión de tratamiento médico

Elaborado por: El investigador

Pruebas de Aceptación	
Número: 7	**Historia de usuario:** 6
Nombre: Gestión de tratamiento médico	
Descripción: El usuario puede añadir observaciones dependiendo del tratamiento que tiene un paciente. Se muestra un listado de observaciones según el tratamiento del paciente en orden cronológico.	
Condiciones de ejecución: El usuario debe ser el administrador del sistema.	
Entrada: Rellenar los campos solicitados.	
Resultado Esperado: El usuario puede añadir observaciones a las tratamientos de los pacientes.	
Evaluación de prueba: Satisfactoria	

Tabla 3.37: Prueba de aceptación – Observaciones de tratamientos

Elaborado por: El investigador

Pruebas de Aceptación	
Número: 8	**Historia de usuario:** 7
Nombre: Número de historia clínica	
Descripción: El número de historia clínica es el mismo que el número de cédula por lo cual es necesario duplicar dicha entrada.	
Condiciones de ejecución: El usuario debe ser el administrador del sistema.	
Entrada: Se debe ingresar un número de cédula de acuerdo con el formato requerido.	
Resultado Esperado: El número de historia clínica tiene el mismo valor que el campo de la cédula.	
Evaluación de prueba: Satisfactoria	

Tabla 3.38: Prueba de aceptación - Número de historia clínica

Elaborado por: El investigador

Pruebas de Aceptación	
Número: 9	**Historia de usuario:** 8
Nombre: Cerrar sesión	
Descripción: El usuario puede cerrar la sesión, a la vez se hace una limpieza de caché y destrucción de cookies para evitar el regreso a páginas anteriores sin antes primero iniciar sesión.	
Condiciones de ejecución: El usuario debe ser el administrador del sistema.	
Entrada:	
Resultado Esperado: La sesión se cierra limpiando el caché y destruyendo las cookies.	
Evaluación de prueba: Satisfactoria	

Tabla 3.39: Prueba de aceptación - Cerrar sesión

Elaborado por: El investigador

Pruebas de Aceptación	
Número: 10	**Historia de usuario:** 9
Nombre: Notificaciones WhatsApp	
Descripción: Al momento de registrar un cita o cancelarla el paciente recibirá un mensaje de WhatsApp con la información establecida de acuerdo con la acción.	
Condiciones de ejecución: El usuario debe ser el administrador del sistema	
Entrada: Se debe ingresar un número de teléfono con una cuenta activa en WhatsApp.	
Resultado Esperado: Los mensajes WhatsApp se envían de forma correcta.	
Evaluación de prueba: Satisfactoria	

Tabla 3.40: Prueba de aceptación - Notificaciones WhatsApp

Elaborado por: El investigador

Pruebas de Aceptación	
Número: 11	**Historia de usuario:** 10
Nombre: Notificaciones WhatsApp de recordatorio	
Descripción: Los envíos de mensajes WhatsApp se harán de forma automática todos los días a las 18:00 a todos los pacientes que tienen registrado un turno de atención al día siguiente.	
Condiciones de ejecución: El usuario debe ser el administrador del sistema	
Entrada: Se hace una consulta a la base de datos y se obtiene el número de teléfono de los pacientes que tienen cuentas de WhatsApp y se les envía las notificaciones.	
Resultado Esperado: Los mensajes WhatsApp se envían de forma automática.	
Evaluación de prueba: Satisfactoria	

Tabla 3.41: Prueba de aceptación - Notificaciones WhatsApp de recordatorio

Elaborado por: El investigador

3.2.6 Fase 6: Producción

3.2.6.1 Servidor

Sistema operativo	Distribución	CPU	RAM	Espacio en disco	Servidor web	Dominio:
Linux	Alma Linux	2 Cores	2 GB	80 GB SSD	nginx	novaden-ec.com

Tabla 3.42: Características del VPS (Virtual Private Server)

Elaborado por: El investigador

3.2.6.2 Despliegue de la aplicación Node.js

Instalación de paquetes del servidor

```
dnf install epel-release
dnf install nodejs
dnf install npm
dnf install mariadb-server
dnf install nginx
dnf install wget
wget https://dl.google.com/linux/direct/google-chrome-stable_current_x86_64.rpm
dnf localinstall google-chrome-stable_current_x86_64.rpm
```

Figura 3.98: Paquetes del servidor

Elaborado por: El investigador

Instalación de paquetes para Node.js

```
npm install -g npm@latest
npm i puppeteer
npm install uuid@latest
npm run install
npm install pm2 -g
```

Figura 3.99: Paquetes para Node.js

Elaborado por: El investigador

3.2.6.3 Módulo pm2

El módulo pm2 permite la ejecución del aplicativo Node.js en segundo plano permitiendo la edición de los archivos del sistema web en caso de ser necesario sin detener el servicio. El siguiente comando permitite inicializar el archivo núcleo del sistema; *pm2 start /var/www/novadent/app/index.js*

Verificar estado del demonio pm2

```
[root@sn conf.d]# pm2 status
```

id	name	namespace	version	mode	pid	uptime	↺	status	cpu	mem	user	watching
0	index	default	1.0.0	fork	196638	72m	18	online	0%	78.1mb	root	disabled

Figura 3.100: Estado de ejecución del demonio pm2

Elaborado por: El investigador

3.2.6.4 Configuración del servidor

```
server {
    listen       443 ssl http2 default_server;
    listen       [::]:443 ssl http2 default_server;
    server_name  novadent-ec.com;
    root         /var/www/novadent/app;

    ssl_certificate "/etc/letsencrypt/live/novadent-ec.com/fullchain.pem";
    ssl_certificate_key "/etc/letsencrypt/live/novadent-ec.com/privkey.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout  10m;
    ssl_ciphers PROFILE=SYSTEM;
    ssl_prefer_server_ciphers on;

    include /etc/nginx/default.d/*.conf;

    location / {
        proxy_pass http://localhost:3000/;
    }
         error_page 404 /404.html;
        location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
        location = /50x.html {
    }
}
```

Figura 3.101: Configuración de certificados SSL y del servidor web nginx

Elaborado por: El investigador

Una vez terminada la configuración y la instalación de certificados SSL (Secure Sockets Layer) se puede observar el correcto funcionamiento del sistema web como sitio seguro.

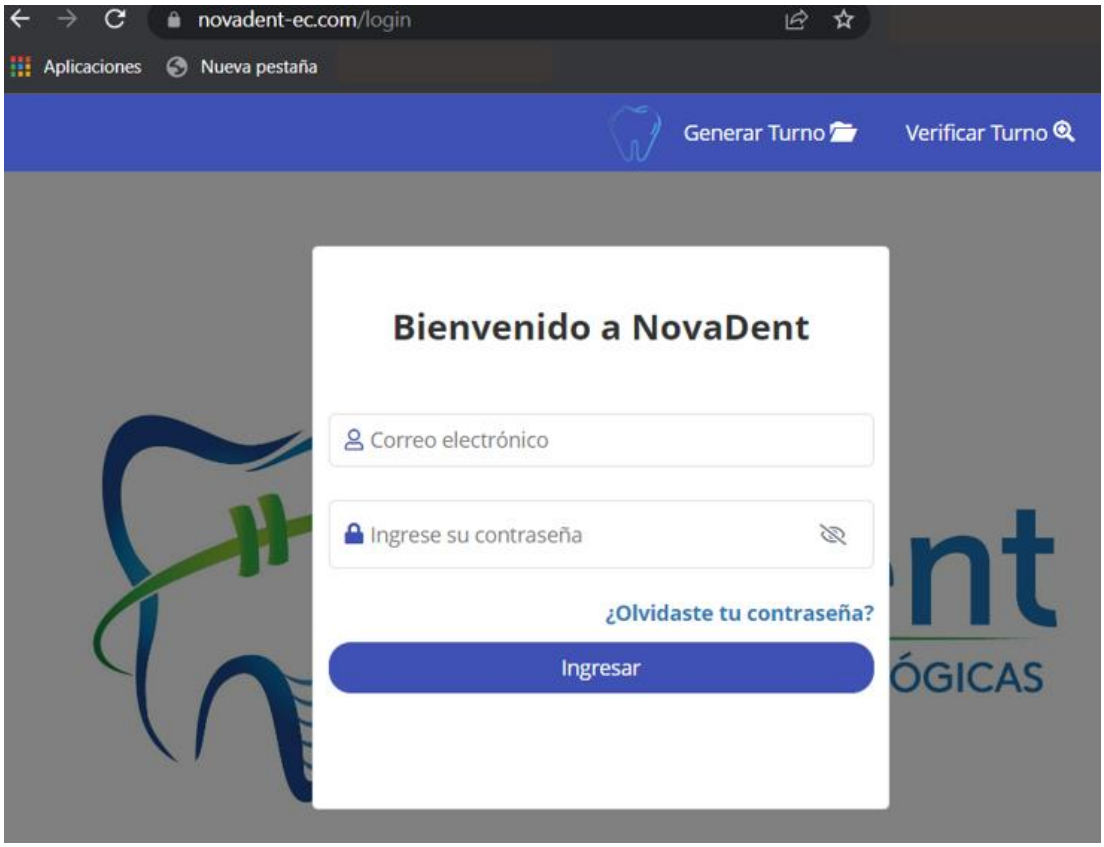


Figura 3.102: Verificación del correcto funcionamiento del sistema web

Elaborado por: El investigador

3.2.7 Fase 7: Finalización

Una vez finalizada con la etapa de implantación de sistema web se realizó una capacitación al usuario quien manipulará el sistema con el siguiente cronograma.

No.	Actividades	Instructor	Fecha de inicio	Fecha fin
1	Inicio de sesión	Esteban Chanatasig	29/07/2022	29/07/2022

2	Reseteo de contraseña	Esteban Chanatasig	29/07/2022	29/07/2022
3	Gestión de pacientes	Esteban Chanatasig	29/07/2022	29/07/2022
4	Galería de imágenes	Esteban Chanatasig	29/07/2022	29/07/2022
5	Gestión de citas	Esteban Chanatasig	29/07/2022	29/07/2022
6	Gestión de tratamiento médico	Esteban Chanatasig	29/07/2022	29/07/2022
8	Cerrar sesión	Esteban Chanatasig	29/07/2022	29/07/2022
9	Notificaciones WhatsApp	Esteban Chanatasig	29/07/2022	29/07/2022
10	Notificaciones WhatsApp de recordatorio	Esteban Chanatasig	29/07/2022	29/07/2022

Tabla 3.43: Cronograma de capacitación a los usuarios

Elaborado por: El investigador

Además, de entregar un manual de usuario detallado en el Anexo A.

CAPÍTULO IV.- CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- Al aplicar las técnicas de recolección de información del centro odontológico permitió conocer las deficiencias en los procesos de atención al paciente en cuanto a manipulación de información se refiere, por ejemplo los registro de citas y datos personales se los lleva en documentación física sin respaldo alguno, razón por la cual permitió la factibilidad del desarrollo de la aplicación pues su finalidad es la gestión de las actividades del odontólogo en turno.
- Se evidenció la importancia de la implementación de la arquitectura MVC en el desarrollo del proyecto debido a la mantenibilidad además de la organización de código haciendo más sencillo identificar posibles errores y posteriormente corregirlos.
- Se concluyó que el modelo de servicio PaaS es el más apropiado debido a que provee un ambiente adecuado para la implantación del sistema web y así dando solución a una de las problemáticas el cual era no contar con un servidor físico.
- La metodología ágil Extreme Programming (XP) elegida para gestionar las actividades del proyecto fue la adecuada ya que esta tiene la característica de cubrir proyectos pequeños y de corta duración permitiendo un desarrollo ágil para así cumplir con los objetivos establecidos.
- Node.js facilitó la estructuración y ejecución del proyecto debido a una gran variedad de paquetes que este posee para un correcto funcionamiento permitiendo desarrollar las funcionalidad del sistema web de manera fácil y de este modo cumplir con los requerimientos establecidos.

4.2 Recomendaciones

- Se recomienda mantener una organización adecuada de los modelos y controladores en la aplicación de la arquitectura MVC y especificar de manera

correcta las rutas que permiten las salidas a las distintas vistas, además de tener actualizados los paquetes y middlewares.

- Se recomienda buscar un proveedor de servicios en la nube confiable que garantice el cumplimiento de las demandas (seguridad, disponibilidad de datos) de la organización que contrata su servicio en este caso del centro odontológico.
- En proyectos medianos y pequeños es recomendable la aplicación de metodologías ágiles por la facilidad que estas otorgan pues ayudan a la identificación de los puntos prioritarios y garantiza el trabajo en equipo.
- Se recomienda revisar la documentación oficial y fuentes confiables acerca de node.js en foros que sean especializados en el tema para solucionar los posibles conflictos que se presenten durante el desarrollo.

Bibliografía

[1] R. L. Katz, *El Papel de las TIC en el Desarrollo*. Fundación Telefónica, 2009. [Online]. Available: https://books.google.com.ec/books?id=4JL5qp5RSWYC

[2] W. Sánchez, "La usabilidad en Ingeniería de Software : definición y características," *Ing-novación. Reporte de investigación*, no. 2, pp. 7–21, 2011, [Online]. Available: http://www.redicces.org.sv/jspui/bitstream/10972/1937/1/2. La usabilidad en Ingenieria de Software- definicion y caracteristicas.pdf

[3] D. Fernández-Quijada, *La innovación tecnológica: Creación, difusión y adopción de las TIC*. Editorial UOC, S.L., 2014. [Online]. Available: https://books.google.com.ec/books?id=VjXkAgAAQBAJ

[4] José Bernabe Vera Marin, "PLATAFORMA COMO SERVICIO (PAAS) PARA LA CREACIÓN, DESARROLLO Y DESPLIEGUE DE APLICACIONES WEB EN LA FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL," p. 78, 2018.

[5] D. A. V. Carrillo, "DISEÑO DE UN SISTEMA DE PEDIDOS ONLINE A TRAVÉS DE UNA INFRAESTRUCTURA DE CLOUD COMPUTING," vol. 523, 2017.

[6] D. A. A. NAPA and J. C. G. VALENZUELA, "Modalidad : Sistematización De Experiencias Tema : Desarrollo De Una Aplicación Web De Gestión Y Médico De La Espam Mfl Autores : Diego Armando Alvarado Napa Jefferson Clíder Guillén Valenzuela Tutor :," 2018.

[7] E. J. González Tumbaco, "Implementar un sistema web para la gestión clínica dental, aplicando tecnologías open source: caso Consultorio Odontológico Navarro.," p. 138, 2017.

[8] U. Universidad de Posgrados, G. Vicente Salgado Andrade, and W. Ruiz Buchelli, “Universidad Politécnica Salesiana Sede Guayaquil DIRECTOR,” *Revista EIA, ISSN 1794-1237*, vol. Volumen 17, pp. 1–10, 2015.

[9] M. J. M. Galindo and J. M. M. Simó, *Escaneando la informática*. Editorial UOC, S.L., 2010. [Online]. Available: https://books.google.com.ec/books?id=svpzjkMpdiUC

[10] S. Murugesan and Y. Deshpande, *Web Engineering: Managing Diversity and Complexity of Web Application Development*, no. n.º2016. Springer, 2001. [Online]. Available: https://books.google.com.ec/books?id=B2Ge5waz9YsC

[11] G. Popoter, “Rediseño de aplicaciones utilizando las tecnologías modernas para el desarrollo web en su parte Front-end,” 2015.

[12] D. C. Langreo and G. G. Urtiaga, *Cómo hacerse informático: Y ganar dinero de manera estable*. AprendeIT. [Online]. Available: https://books.google.com.ec/books?id=1xz0DwAAQBAJ

[13] M. BELTRÁN PARDO and F. SEVILLANO JAÉN, *Cloud Computing, tecnolog{\'\i}a y negocio*. Ediciones Paraninfo, S.A, 2013. [Online]. Available: https://books.google.com.ec/books?id=f5jLAgAAQBAJ

[14] J. W. van den Bent, *EXIN Cloud Computing Foundation*. Van Haren Publishing, 2012. [Online]. Available: https://books.google.com.ec/books?id=8VVeAgAAQBAJ

[15] Á. Vázquez, J. A. Gómez, and R. Serrano, *Android: del diseño de la arquitectura al despliegue profesional*. Marcombo, 2019. [Online]. Available: https://books.google.com.ec/books?id=zEtOEAAAQBAJ

[16] M. U. Durán, *Gestión de calidad*. D{\'\i}az de Santos, 1992. [Online]. Available: https://books.google.com.ec/books?id=hoRlEGdLGxIC

[17] A. Hernandez, “Los Sistemas de Información: Evolución y Desarrollo,” *Dialnet*, no. 10, p. 14, 1996, [Online]. Available: https://dialnet.unirioja.es/descarga/articulo/793097.pdf

[18] P. Beynon-Davies, *Sistemas de información: Introducción a la informática en las organizaciones*. Reverte, 2018. [Online]. Available: https://books.google.com.ec/books?id=5jbeDwAAQBAJ

[19] G. Ponjuán, “Gestión documental, de información y del conocimiento... puntos de contacto y diferencias,” *Ciencias de la información*, vol. 34, no. 3, pp. 55–63, 2003.

[20] J. P. Subra and A. Vannieuwenhuyse, *Scrum: un método ágil para sus proyectos*. Ediciones Eni, 2018. [Online]. Available: https://books.google.com.ec/books?id=TyQuFpGhZ8sC

[21] M. Hernández Bejarano and L. E. Baquero Rey, *Ciclo de vida de desarrollo ágil de software seguro*. Fundación Universitaria Los Libertadores, 2020. [Online]. Available: https://books.google.com.ec/books?id=XdQ7EAAAQBAJ

[22] J. C. C. F. J. C. Fernando Berzal, *Desarrollo Profesional de Aplicaciones Web con ASP.NET*. iKor Consulting. [Online]. Available: https://books.google.com.ec/books?id=J1d%5C_9l6zlAIC

[23] T. Dimes and A. Parraud, *JavaScript Una Gu{\'\i}a de Aprendizaje para el Lenguaje de Programación JavaScript*. Babelcube Incorporated, 2015. [Online]. Available: https://books.google.com.ec/books?id=-4zGCQAAQBAJ

[24] S. G. Herrador, *HTML \& CSS Fácil y sencillo*. Lulu Enterprises Incorporated, 2010. [Online]. Available: https://books.google.com.ec/books?id=TZnXAQAAQBAJ

[25] M. A. Arias and Valencia, *Webs Responsivas. Responsive Design con Bootstrap*. Createspace Independent Pub, 2014. [Online]. Available: https://books.google.com.ec/books?id=rl3hwAEACAAJ

[26] A. D. Sole, *Visual Studio Code Distilled: Evolved Code Editing for Windows, macOS, and Linux*. Apress, 2018. [Online]. Available: https://books.google.com.ec/books?id=YqR8DwAAQBAJ

[27] G. G. Urtiaga, *Administrar MySQL y MariaDB: Aprende a administrar MySQL y MariaDB fácilmente*. AprendeIT. [Online]. Available: https://books.google.com.ec/books?id=3DPwDwAAQBAJ

[28] L. Puciarelli, *Node JS - Vol. 1: Instalación - Arquitectura - node y npm*. RedUsers, 2020. [Online]. Available: https://books.google.com.ec/books?id=GOfqDwAAQBAJ

[29] J. M. O. Candel, *Desarrollo seguro en ingenier{\'\i}a del software: Aplicaciones seguras con Android, NodeJS, Python y C++*. Alpha Editorial, 2020. [Online]. Available: https://books.google.com.ec/books?id=x3J6EAAAQBAJ

[30] S. Aguirre, *.NET Aplicaciones Web - Vol.2: ASP. NET Core. Modelo MVC*. RedUsers, 2021. [Online]. Available: https://books.google.com.ec/books?id=AjNREAAAQBAJ

Anexos

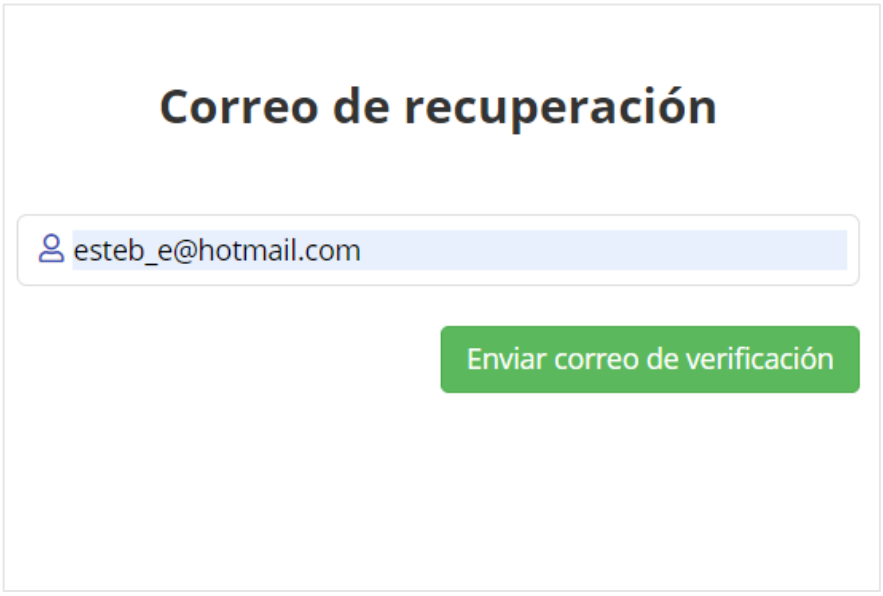
Anexo A: Manual de usuario

Inicio de sesión

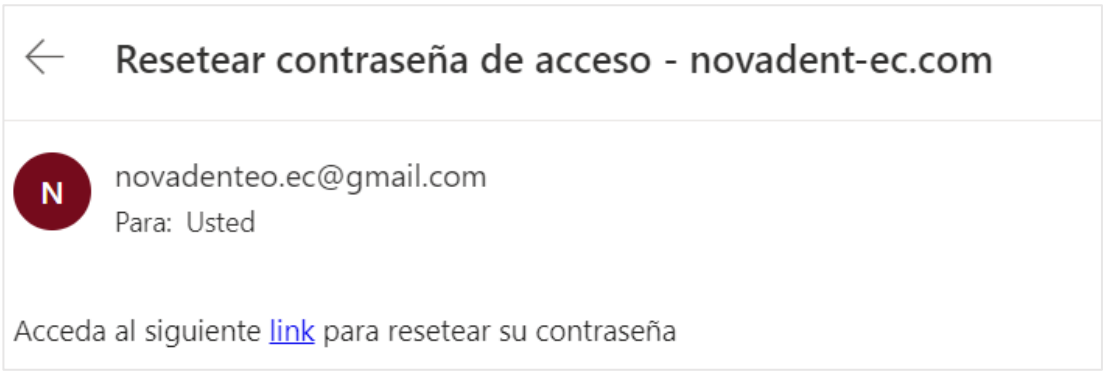
Nota. Los datos proporcionados para el ejemplo no son verdaderos.

1. El usuario debe ingresar un correo electrónico al cual tenga acceso a verificar su bandeja de entrada.
2. La contraseña debe estar compuesta por 10 dígitos sean números, letras o una combinación de ambas.
3. Al pulsar en el enlace ¿Olvidaste tu contraseña? Se abrirá una ventana nueva.

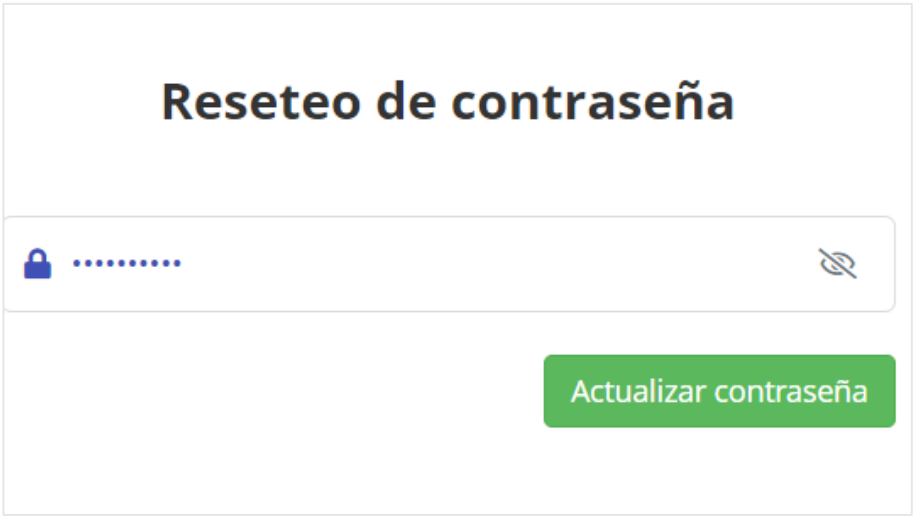
4. En la nueva ventana llamada “Correo de recuperación” el usuario debe ingresar el correo electrónico con el cual está registrado en el sistema y pulsar **Enviar correo de verificación.**



5. En la bandeja de entrada del correo electrónica llegará el siguiente mensaje.



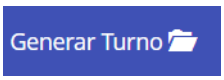
6. Ingresar el link proporcionado.
7. Una vez abierto el link nos mostrará la siguiente ventana en la cual se debe ingresar una nueva contraseña.



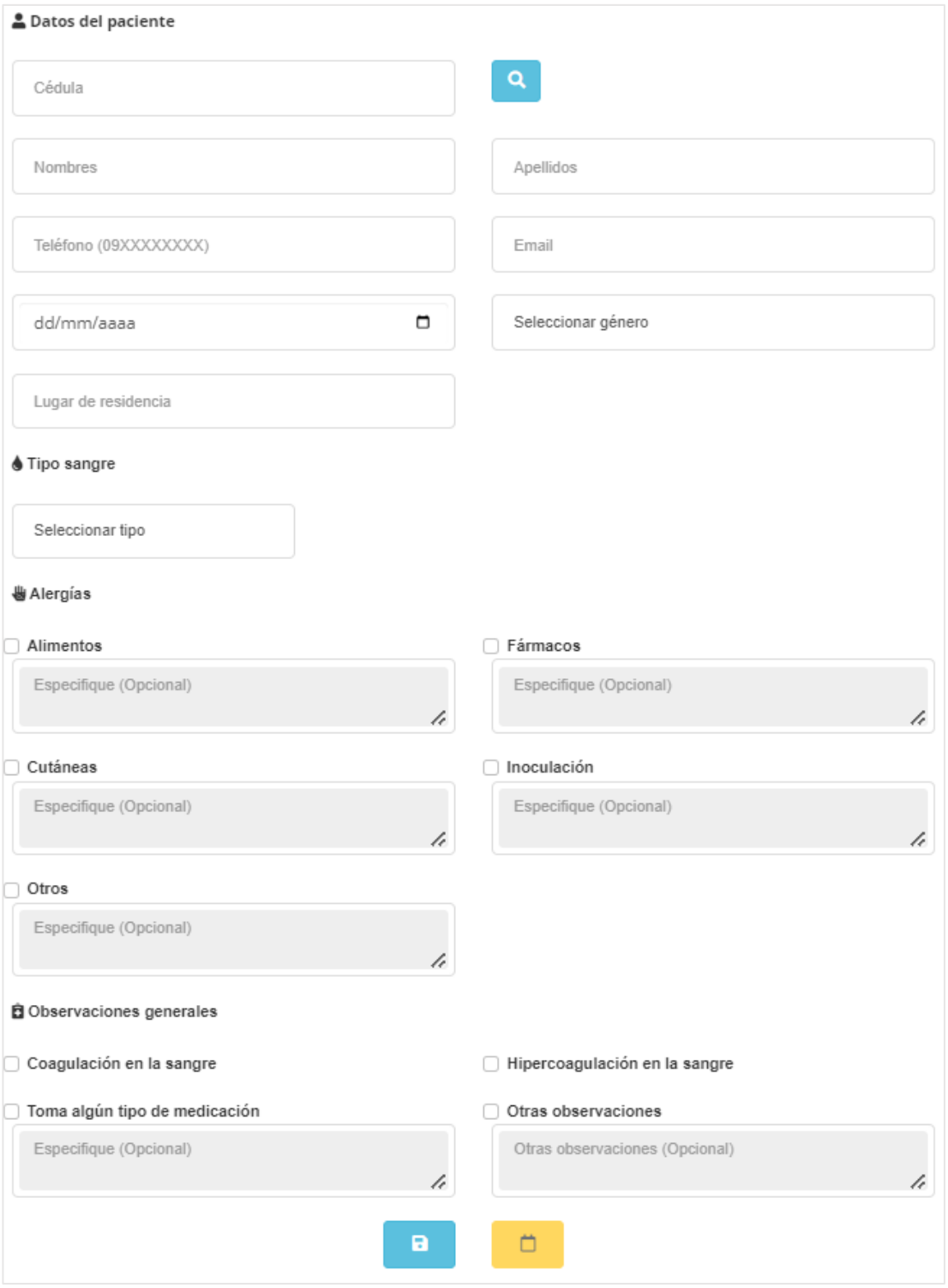
8. Finalmente, al cambiar la contraseña el usuario podrá ingresar el sistema.

Creación de turnos de atención

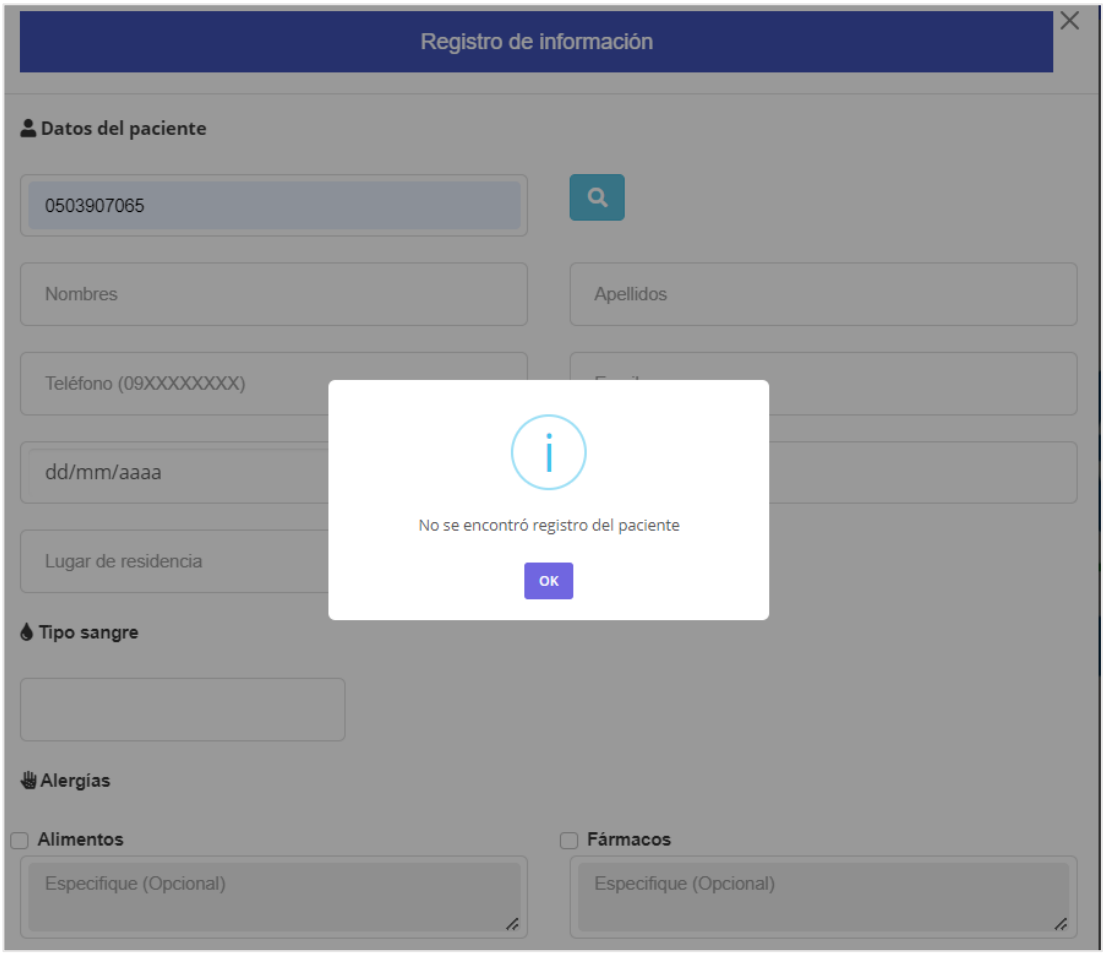
1. El usuario debe ingresar en el siguiente módulo.



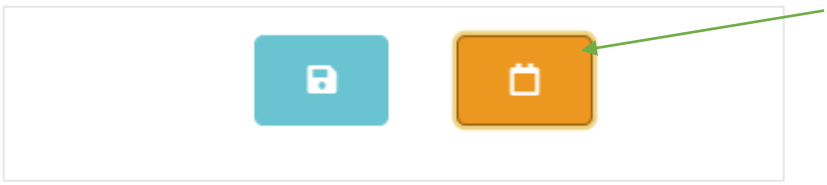
2. Se mostrará una ventana en la cual se podrá buscar si el paciente ya se encuentra registrado en el sistema.

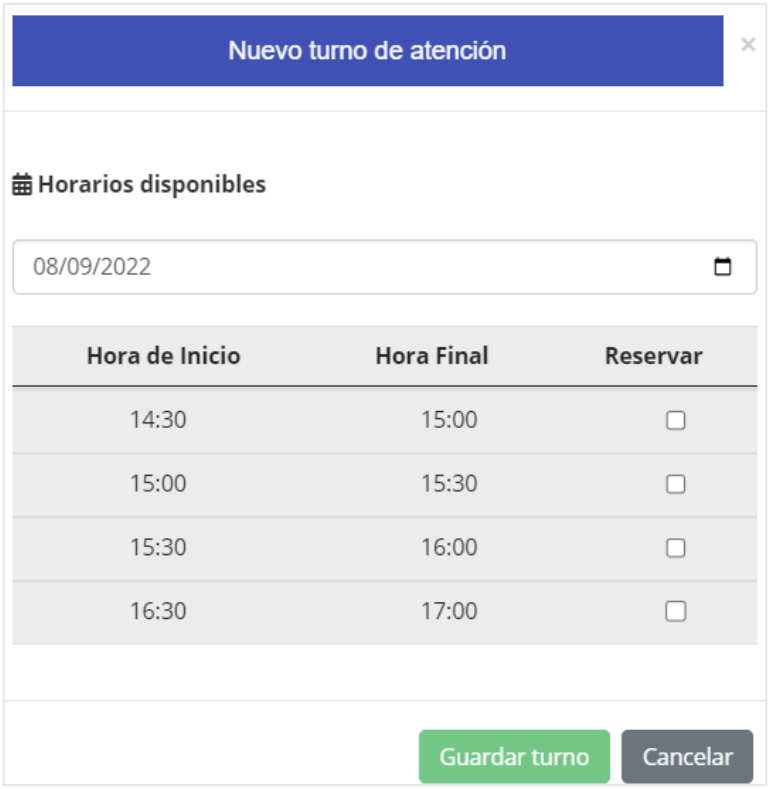


3. Si el paciente no está registrado se habilitan los campos y el usuario deberá completar los campos (Datos personales, Tipo de sangre, Alergias, y otros padecimientos médicos).



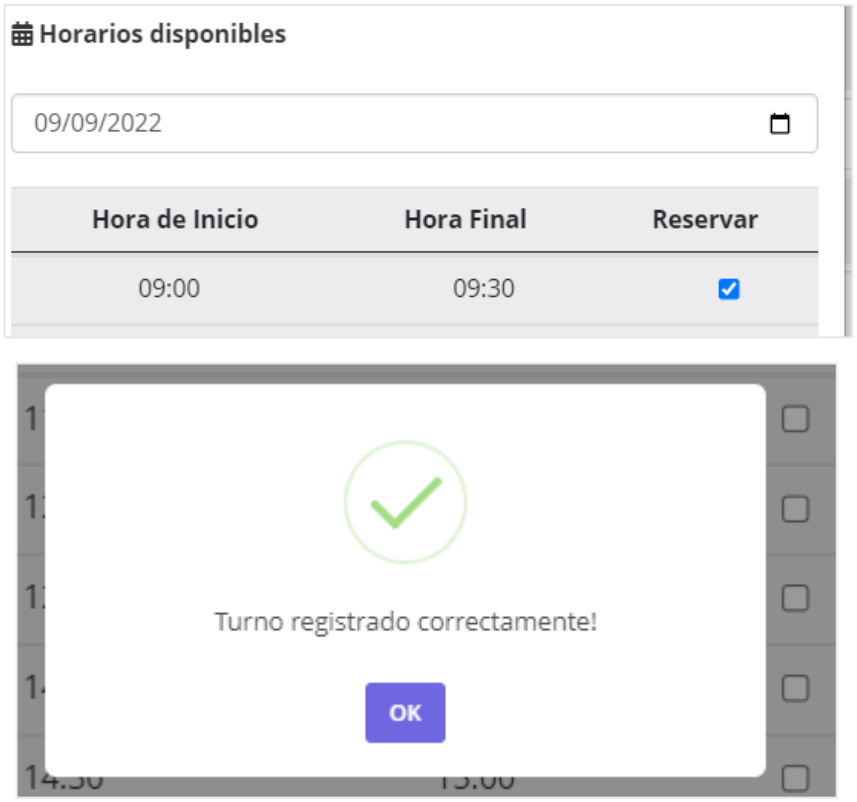
4. Una vez registrado el paciente el campo fecha se habilitará el botón registrar cita, el cual despliega un modal en donde se podrá especificar la fecha deseada y los horarios disponibles para dicha fecha.





5. Una vez elegida la fecha se desplegará un cuadro con los horarios disponibles, si en la fecha seleccionada no existen horarios disponibles te mostrará el siguiente mensaje.

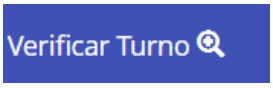
6. De lo contrario se selecciona el horario requerido y se registra la cita.



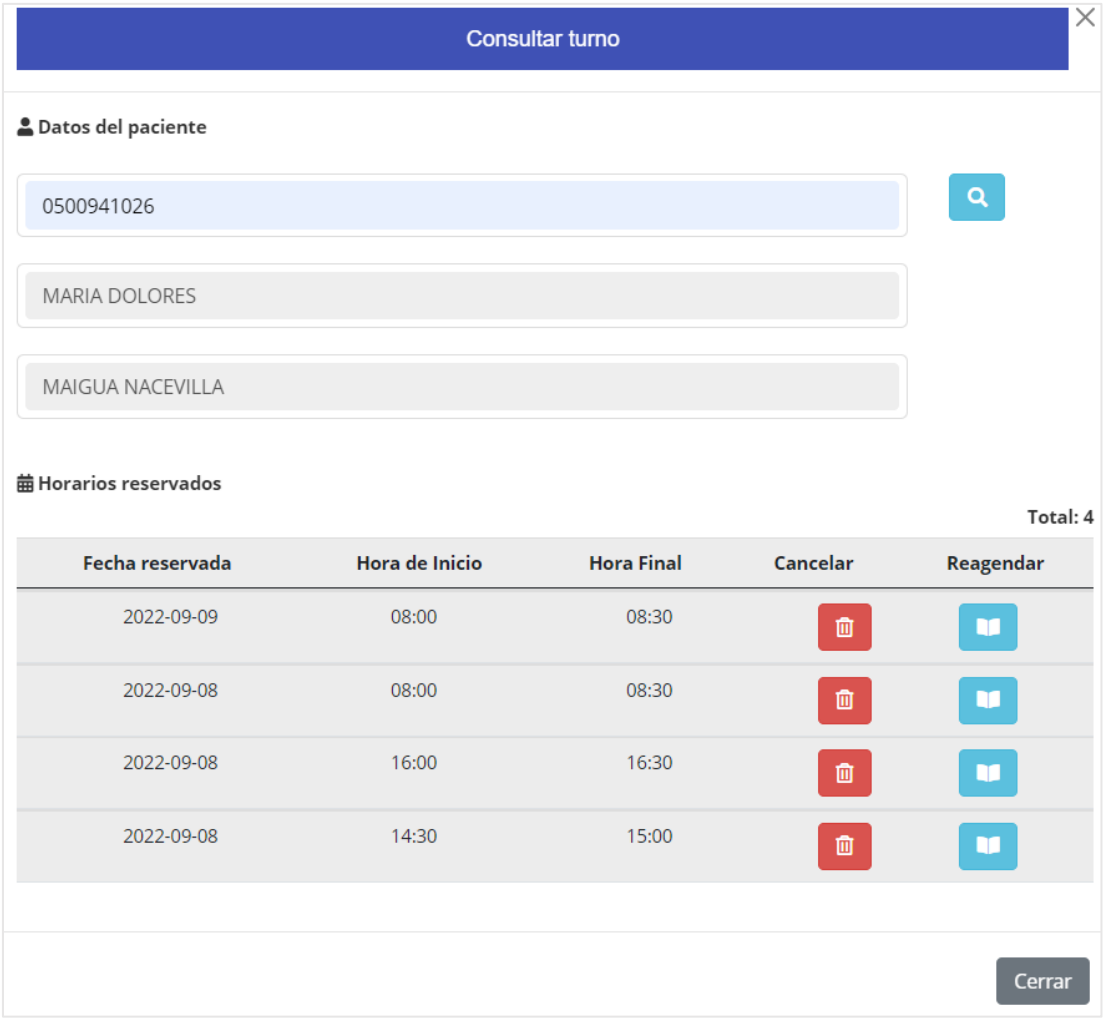
7. Si el registro tuvo éxito se realizará una validación al número de teléfono (en caso de proporcionarse) si tiene cuenta en WhatsApp y se procederá a enviar un mensaje de notificación.

Consultar turnos de atención

1. Al pulsar en el siguiente botón se mostrará una ventana de consulta.

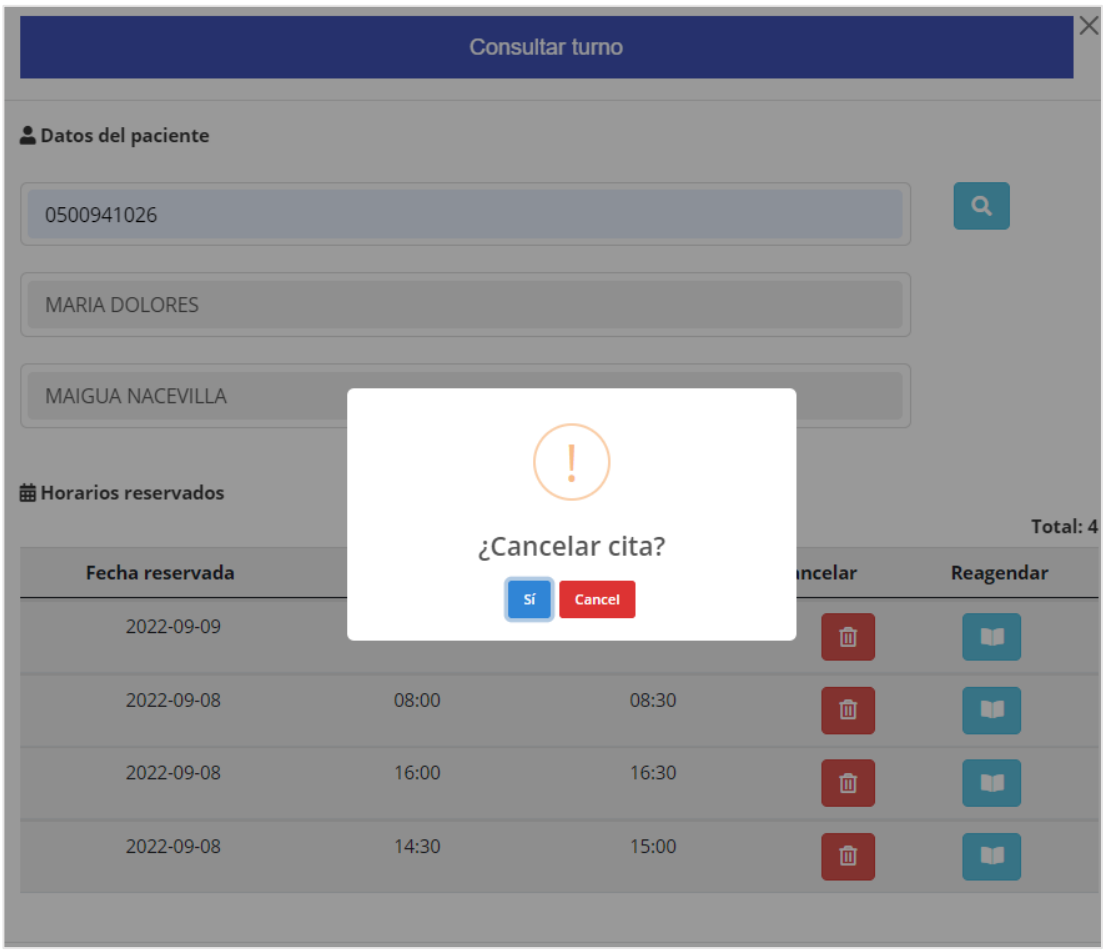


2. En la ventana desplegada el usuario deberá ingresar el número de cédula del paciente de quien se desea hacer la consulta.
3. Si el paciente tiene turnos registrado se enlistarán en un cuadro informativo en la cual el usuario podrá cancelar si el paciente lo requiera.
4. Si el paciente no tiene turnos registrados se mostrará un mensaje de advertencia.



Cancelar turnos de atención

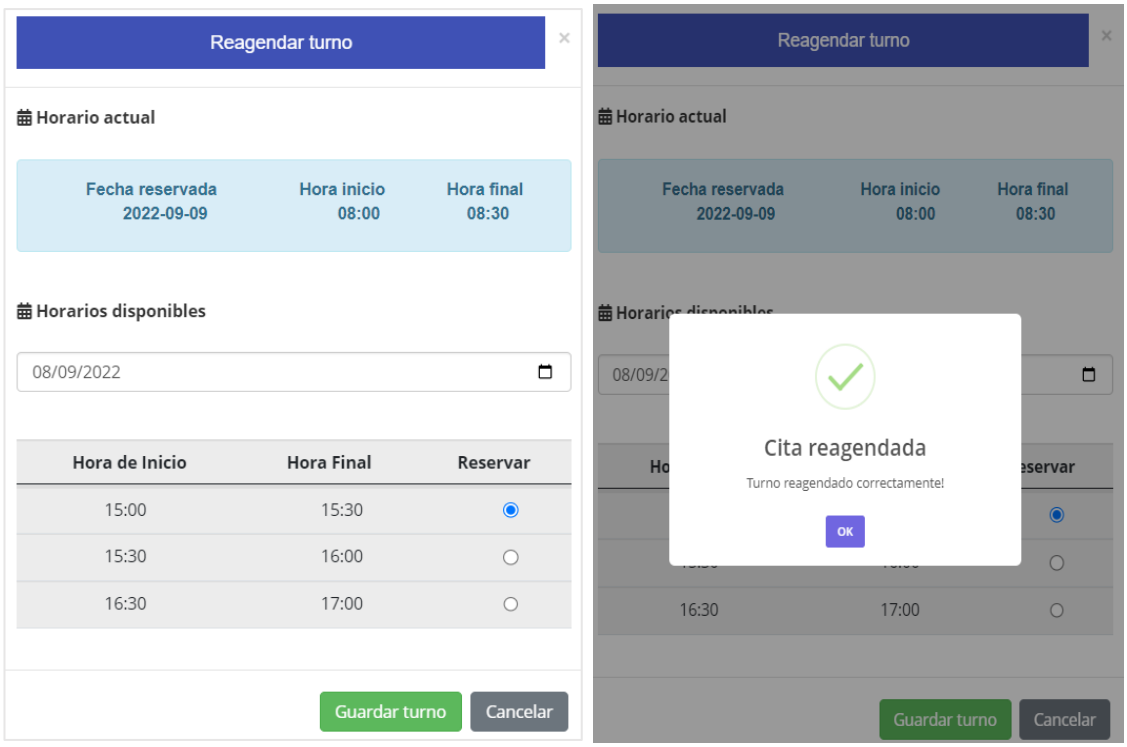
1. Al pulsar el icono  se podrá cancelar un turno de atención.



2. Al cancelar el turno de igual forma se verificar el número de teléfono (en caso de poseer) y se enviará un mensaje de WhatsApp con la notificación de cancelación.

Reagendar turnos de atención

1. Al pulsar en el icono se podrá reagendar un turno de atención en el horario.

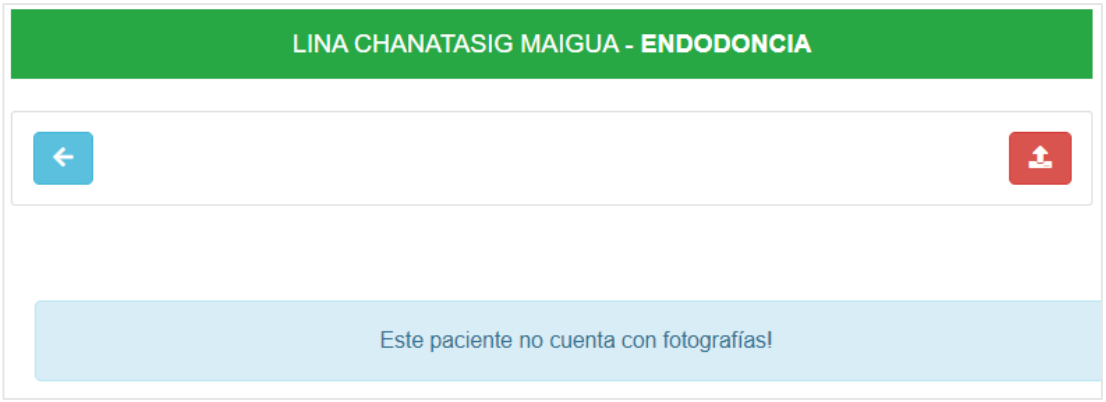


2. Al reagendar el turno de igual forma se verificar el número de teléfono (en caso de poseer) y se enviará un mensaje de WhatsApp con la notificación de cancelación.

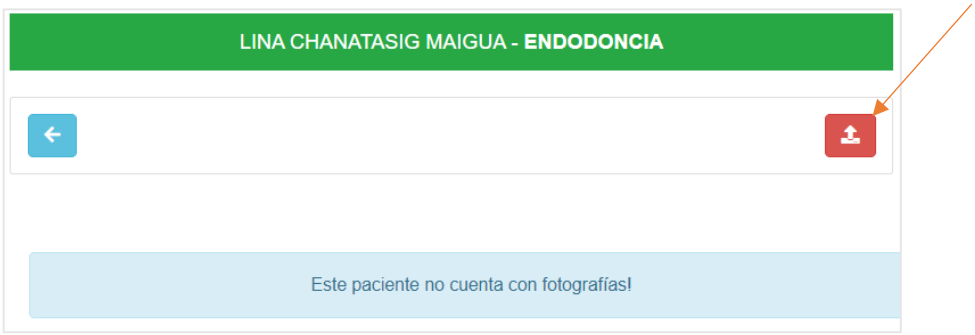
Galería de imágenes

1. El usuario deberá seleccionar un tratamiento de la lista de opciones.

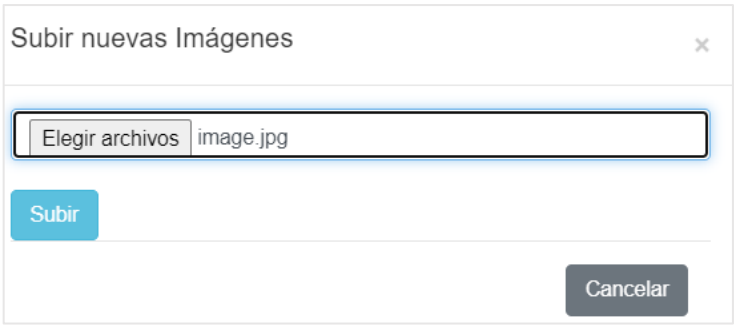
2. Una vez elegido al paciente del listado se debe pulsar en el botón Ver Galería, y se mostrará la siguiente ventana.



3. Si el paciente cuenta con imágenes en su galería se las mostrará en orden cronológico, de lo contrario se desplegará un mensaje de no contar con ningún registro de fotografías. Para agregar una imagen pulsar en el botón ubicado en la parte superior izquierda.



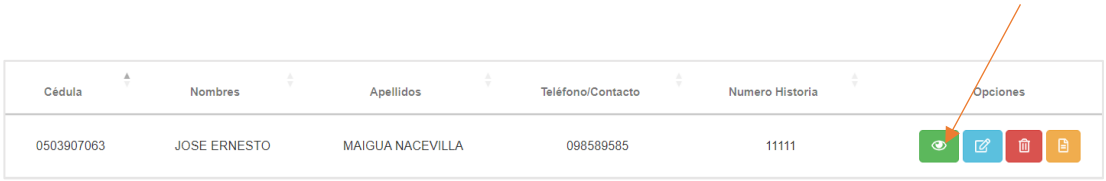
4. Seleccionar la imagen que se requiera y presionar **Subir**.



5. Una vez subida la imagen se podrá visualizarla, eliminar o descargarla.

Ver datos del paciente

1. Del listado del paciente, se debe elegir el icono de visualización para poder ver los datos del paciente.



Actualizar datos del paciente

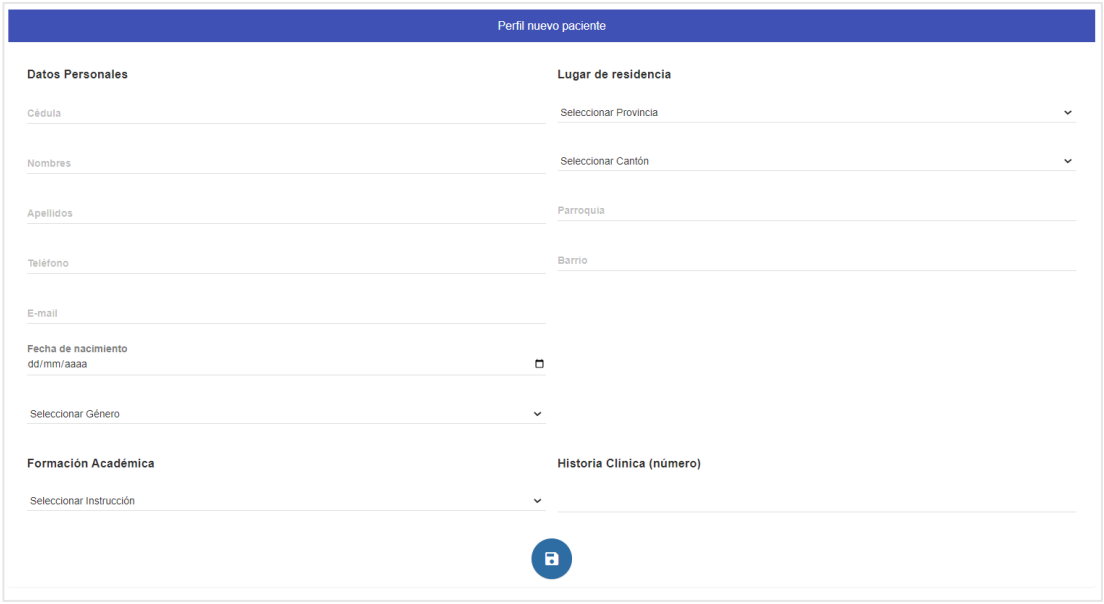
1. Para actualizar datos del paciente se debe pulsar el icono de edición, el cual desplegará una ventana en la cual se deberán especificar la nueva información.

Borrar un paciente

1. Para la borrar un paciente se deberá pulsar el icono de eliminación.

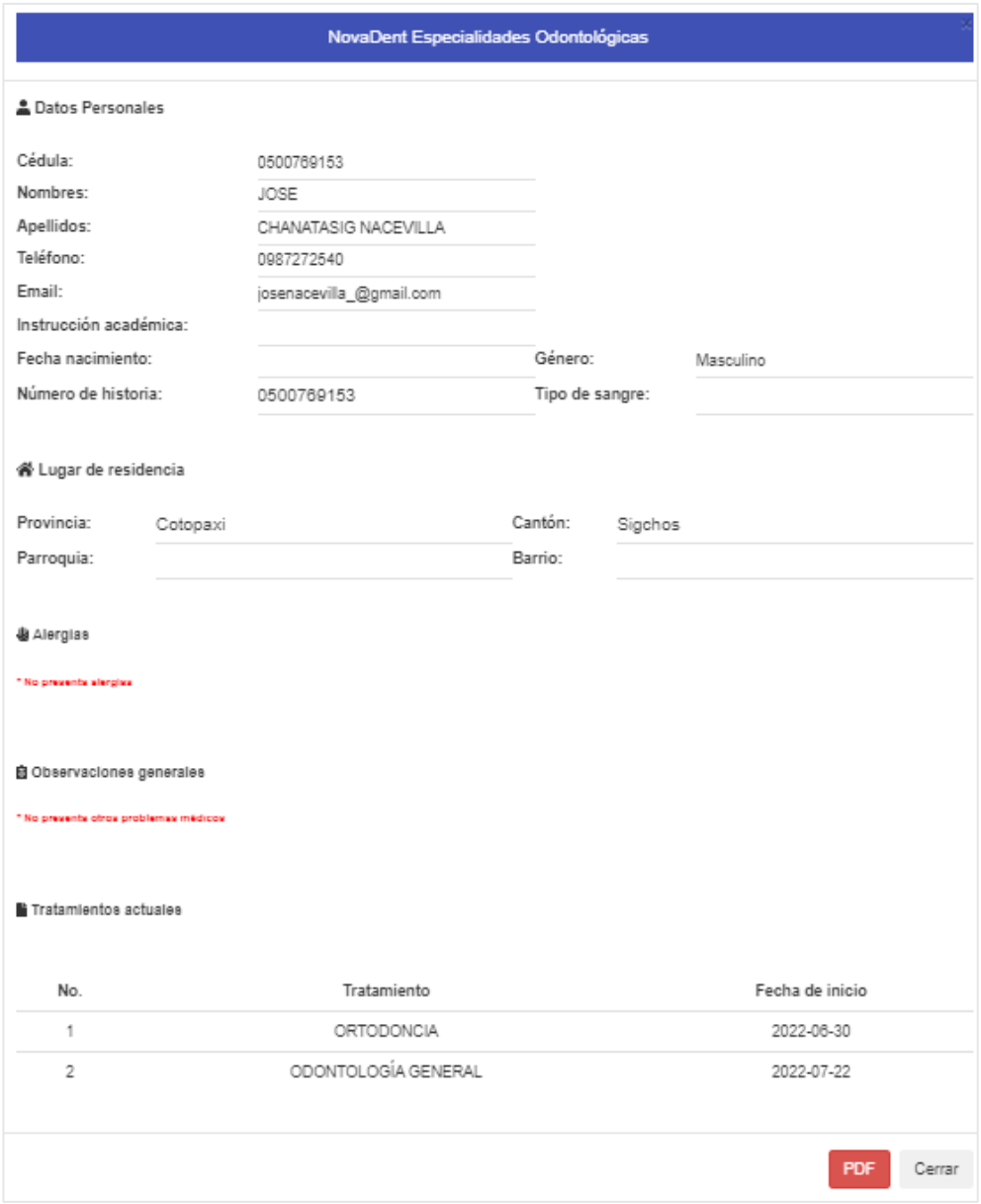
Anadir nuevo paciente

2. Para añadir un nuevo paciente se debe pulsar el icono + y mostrará la siguiente ventana. En la cual se deberán rellenar los campos necesarios. No todos los campos son obligatorios únicamente la cédula, nombres y apellidos del paciente.



Generar PDF de ficha médica de un paciente

1. Al presionar el icono se desplegará una ventana modal con toda la información del paciente seleccionado, tanto de tratamientos, alergias y otros padecimientos.
2. Si es necesario también se podrá generar un PDF con la información mostrada.



Generar documento PDF y Excel de todos los pacientes del centro odontológico.

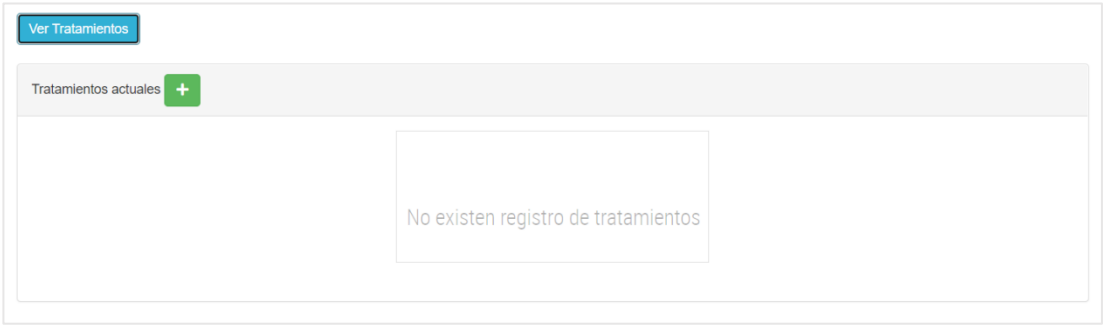
1. Al pulsar en los botones 

se generan los documentos respectivamente.

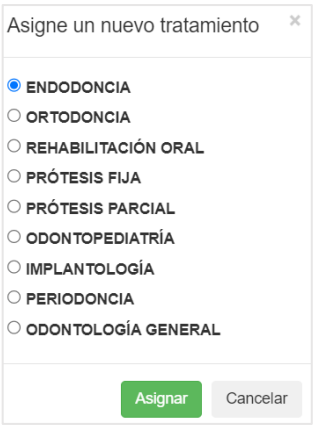
Asignación de Tratamientos

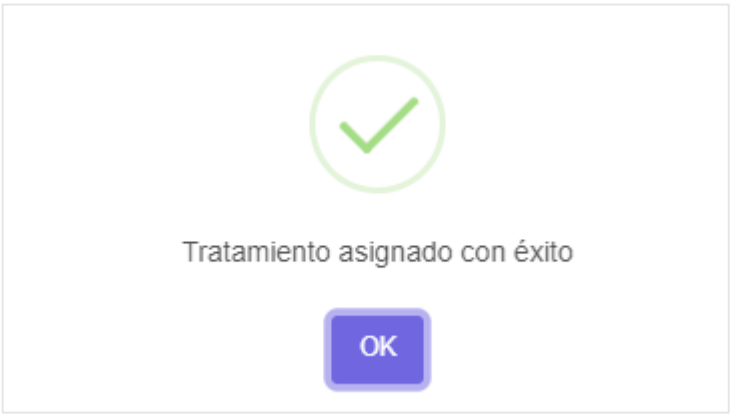
1. El usuario deberá pulsar en el botón  que se encuentra en la lista de opciones de los pacientes de la lista.
2. Mostrará la siguiente información.

3. Al pulsar Ver Tratamientos se desplegará un panel con los tratamientos actuales del paciente seleccionado.

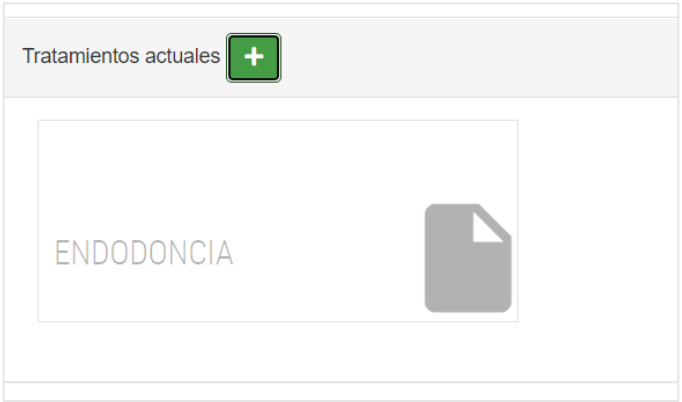


4. Si no cuenta con tratamientos se debe pulsar el botón **Agregar Tratamiento.**



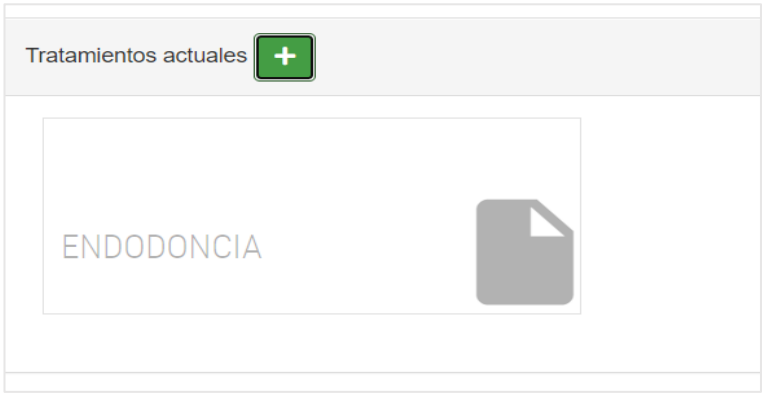


5. Una vez agregado el tratamiento el panel actualizará su contenido.

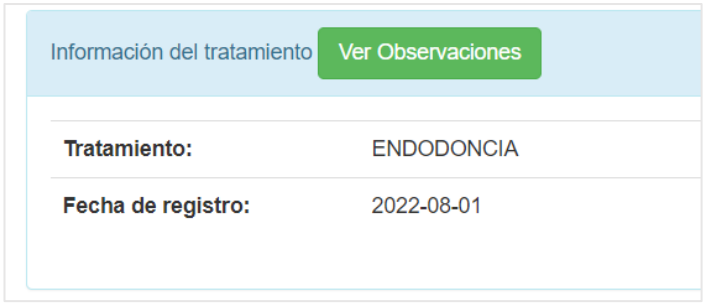


Agregación de observaciones

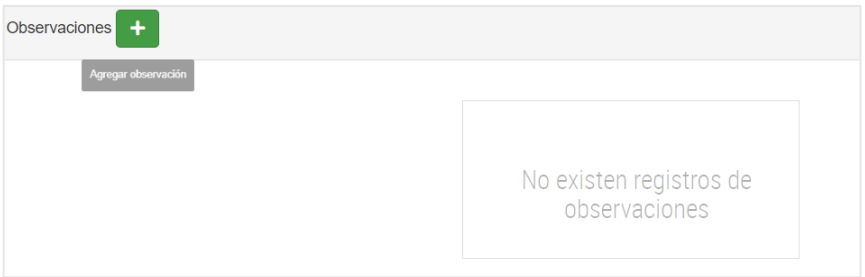
1. Seleccionar una tratamiento.



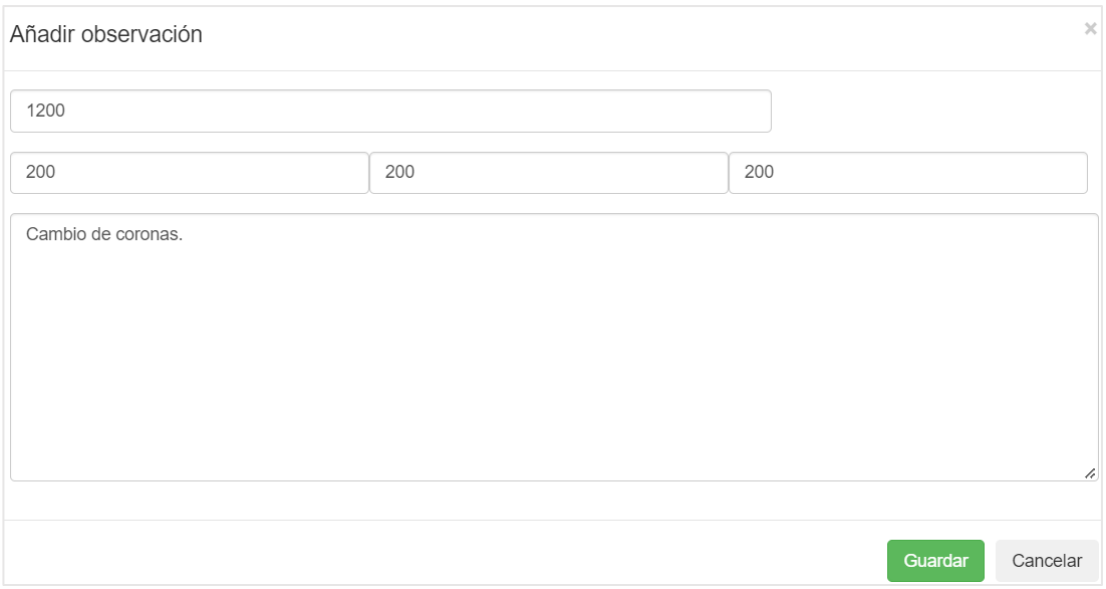
2. Pulsar en **Ver Observaciones.**

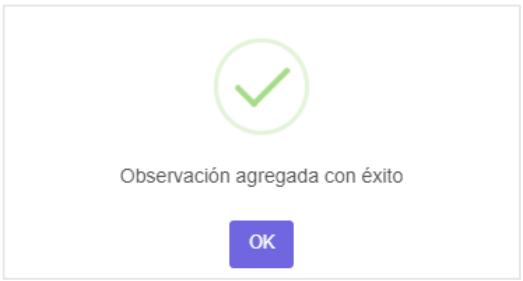


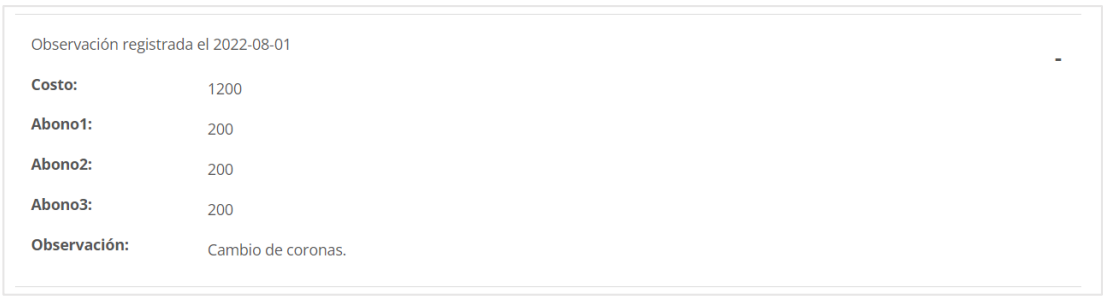
3. Si el tratamiento tiene observaciones se desplegará un panel en forma de acordeón, de lo contrario se mostrará el siguiente mensaje.



4. Pulsar el botón **Agregar Observación,** se abrirá una ventana modal de ingreso.







Cerrar sesión

El usuario debe pulsar el icono de salida de sesión ubicada en la parte superior derecha del menú  para abandonar el sistema, se mostrará una ventana de advertencia y el usuario tomará la decisión.