Proyecto de Grado

Presentado ante la ilustre Universidad de Los Andes como requisito parcial para obtener el Título de Ingeniero de Sistemas

Desarrollo de un Sistema web para la aplicación de Teleodontología en la Facultad de Odontología de la Universidad de Los Andes.

Por

Br. Hildayra Colmenares

Tutor: Prof. Jormany Quintero

Cotutor: Prof. Lorena Bustillos

Noviembre 2022

UNIVERSIDAD DE LOS ANDES
VENEZUELA

Desarrollo de un Sistema web para la aplicación de Teleodontología en la Facultad de Odontología de la Universidad de Los Andes.

Br. Hildayra Colmenares

Proyecto de Grado — Sistemas Computacionales, 108 páginas
Escuela de Ingeniería de Sistemas, Universidad de Los Andes, 2022

Resumen: La telemedicina es la parte clínica de la telesalud dedicada a la prevención, diagnóstico, tratamiento y monitoreo de forma remota. Dentro de este contexto nace el término teleodontología, cuyo propósito es hacer uso de la telemedicina para prestar sus servicios a los pacientes de forma remota. Ante el escenario actual de la salud a nivel mundial, es de carácter urgente que se continúen generando soluciones tecnológicas para establecer puentes de conexión entre los profesionales de la salud y sus pacientes, de modo que se puedan brindar servicios de manera oportuna, eficiente y segura. En este sentido, el objetivo de este trabajo fue desarrollar un sistema web que permita la aplicación de teleodontología en la Clínica Integral de la Facultad de Odontología de la Universidad de Los Andes (CIA-FOULA), con el propósito de automatizar los procesos que actualmente se llevan a cabo de forma manual y presencial, produciendo retrasos, duplicidad de esfuerzos, pérdida de información y gastos de traslado innecesarios, ofreciendo una herramienta de teletriaje y gestión de citas online. Con la finalidad de alcanzar este objetivo, se realizó un estudio previo del comportamiento del sistema, mismo que fue posteriormente modelado y desarrollado bajo una arquitectura Cliente Servidor y un patrón de diseño Modelo Vista Controlador, con un diseño de interfaz agradable, modular, responsivo e intuitivo, que permite la ejecución de las actividades de los distintos actores de manera automatizada.

Palabras clave: telemedicina, teleodontología, teletriaje, TIC, sistemas web, citas online.

Este trabajo fue procesado en LaTeX.

Índice general

Índice de Tablas vi

Índice de Figuras vii

1. Introducción 1

1.1. Antecedentes 2

1.1.1. Sistemas de gestión de citas a través de páginas web 3

1.1.2. Sistemas gestión de citas a través de aplicaciones móviles 5

1.1.3. Sistemas de gestión de citas multiplataformas (Web + App) . . 7

1.1.4. Teleodontología 8

1.2. Planteamiento del Problema 11

1.3. Justificación 15

1.4. Objetivos 16

1.4.1. Objetivo General 16

1.4.2. Objetivos Específicos 17

1.5. Metodología 17

1.6. Alcance 18

2. Marco Teórico 19

2.1. Arquitectura cliente servidor 19

2.2. Modelo Vista Controlador (MVC) 21

2.2.1. Niveles de abstracción del MVC: 21

2.3. Bases de datos 22

2.3.1. Sistema de gestión de bases de datos 23

2.3.2. Modelos de datos 25
2.3.3. Lenguaje SQL (Structured Query Language) 26
2.3.4. PostgreSQL 29
2.4. Sistemas de información 29
2.4.1. Sistemas de información informáticos 29
2.4.2. Sistemas Web 30
2.5. Telemedicina 30
2.6. Teleodontología 30
2.7. Teletriaje 31
2.8. Metodología Scrum 31
2.9. Framework 33
2.10. Frontend 34
2.10.1. HTML(HyperText Markup Language) 34
2.10.2. CSS(Cascading Style Sheets) 34
2.10.3. JavaScript 34
2.10.4. Boostrap 35
2.10.5. Vue.JS 35
2.11. Backend 35
2.11.1. PHP(Hypertext Preprocessor) 35

3. Desarrollo del sistema (Backend) 37
3.1. Arquitectura del sistema 37
3.2. Debian GNU/Linux 39
3.3. Lenguaje: PHP 39
3.4. Laravel 39
3.5. Requrimientos de hardware y software 41
3.6. Modelado de la base de datos 46
3.7. Construcción de la base de datos 47

4. Desarrollo del sistema (Frontend) 53
4.1. Especificación de requerimientos 53
4.1.1. Descripción general de los Módulos del sistema 54

4.1.2. Roles y permisos dentro del sistema 55
4.1.3. Flujos principales del sistema 58

5. Conclusiones **102**
5.1. Recomendaciones 104

Índice de tablas

2.1. Comandos SQL 26
2.2. Cláusulas SQL 27
2.3. Operadores lógicos y de comparación 28
2.4. Funciones de agregado 29

3.1. Tabla de requerimientos de hardware 42
3.2. Tabla de Usuarios 49
3.3. Tabla de perfiles 49
3.4. Tabla de antecedentes médicos familiares 50
3.5. Tabla de antecedentes médicos personales (parte 1) 51
3.6. Tabla de antecedentes médicos personales (parte 2) 52

4.1. Módulos del sistema 55
4.2. Roles y Permisos del sistema 57

Índice de figuras

2.1. Modelo Vista Controlador 22

3.1. Arquitectura Cliente servidor 38
3.2. Entorno de desarrollo en *Visual Studio Code* 41
3.3. Paquetes de laravel 43
3.4. Paquetes de JavaScript 44
3.5. Entorno de desarrollo 45
3.6. Modelo Entidad - Relación. 47

4.1. Módulos del sistema 54
4.2. Home de Jefe de Cátedra 58
4.3. Diagrama de Flujo: Registrar Profesor 59
4.4. Caso de uso: Registrar Profesor 60
4.5. Captura de pantalla: Correo de registro 61
4.6. Caso de uso: Editar Profesor 62
4.7. Caso de uso: Detallar Profesor 63
4.8. Caso de uso: Eliminar Profesor 63
4.9. Diagrama de Flujo: Generar reportes 64
4.10. Caso de uso: Generar reportes 65
4.11. Captura de pantalla: Generar reportes 65
4.12. Ejemplo: Generar reporte de profesor 66
4.13. Home de Profesor 67
4.14. Diagrama de flujo: Registrar Estudiante 68
4.15. Caso de uso: Registrar Estudiante 69
4.16. Caso de uso: Editar Estudiante 70

4.17. Caso de uso: Detallar Estudiante 71
4.18. Caso de uso: Eliminar Estudiante 71
4.19. Vista de pacientes con usuario Profesor 72
4.20. Diagrama de Flujo: Aprobar Paciente 73
4.21. Caso de uso: Aprobar Paciente 74
4.22. Diagrama de Flujo: Rechazar Paciente 75
4.23. Caso de uso: Rechazar Paciente 76
4.24. Diagrama de Flujo: Generar Reportes 77
4.25. Caso de uso: Generar Reportes 78
4.26. Captura de pantalla: Generar Reportes 78
4.27. Home de Estudiante 79
4.28. Vista de pacientes con usuario Estudiante 80
4.29. Diagrama de Flujo: Reservar Paciente 81
4.30. Caso de uso: Reservar Paciente 82
4.31. Captura de pantalla: Correo de notificación de reserva para profesor . . 83
4.32. Diagrama de Flujo: Asignar cita 84
4.33. Caso de uso: Asignar Cita 85
4.34. Captura de pantalla: Asignar Cita 85
4.35. Captura de pantalla: Correo recibido por paciente 86
4.36. Diagrama de Flujo: Agregar historial 87
4.37. Caso de uso: Agregar historial 88
4.38. Captura de pantalla: Agregar historial 89
4.39. Diagrama de Flujo: Dar de alta 90
4.40. Caso de uso: Dar de alta 91
4.41. Diagrama de Flujo: Generar Reportes 92
4.42. Caso de uso: Generar Reportes 93
4.43. Captura de pantalla: Generar Reportes 93
4.44. Home Principal 94
4.45. Registro de usuario paciente 95
4.46. Home de Paciente 95
4.47. Diagrama de Flujo: Registrar Paciente 96

4.48. Caso de uso: Registrar Paciente 97
4.49. Pretriaje de paciente 98
4.50. Inicio de sesión 99
4.51. Captura de pantalla: Cambiar Contraseña (Paso 1) 100
4.52. Captura de pantalla: Cambiar Contraseña (Paso 2) 100
4.53. Captura de pantalla: Cambiar Contraseña (Paso 3) 101
4.54. Captura de pantalla: Cambiar Contraseña (Paso 4) 101

Capítulo 1

Introducción

La innovación en tecnología ha contribuido de manera positiva en los procesos del sector salud, mejorando los tiempos de respuesta, permitiendo mayor acceso a la información, acortando las distancias entre el personal sanitario y sus pacientes, lo que finalmente se traduce en una mejor calidad de vida para las personas. En este sentido, el presente trabajo de investigación, parte de la necesidad que presenta actualmente la Clinica Integral del Adulto (CIA) de la Facultad de Odontología de la Universidad de Los Andes (FOULA), quienes actualmente realizan los procesos asociados a la gestión de citas y de pretriaje de manera manual y presencial, lo que ocasiona duplicidad de esfuerzos, pérdida de la información, demoras en los procesos, e inclusive gastos económicos de traslado de los pacientes de manera innecesaria.

Aunado a los problemas mencionados, converge a la presencia de la COVID-19, en la que se requiere mitigar las aglomeraciones para evitar la propagación de la misma, lo que hace apremiante la necesidad de la implementación de sistemas informáticos para proporcionar la oportunidad de gestionar las citas odontológicas y la realización de pretriaje de manera virtual.

A fin de dar solución a la problemática expuesta, el presente trabajo se encuentra estructurado de la siguiente manera:

Capítulo 1: Además de contar con la presente introducción, detalla el estado del arte sobre sistemas para agendar citas online y a la teleodontontología, este marco de antecedentes está descrito en orden temático y cronológico del más reciente al más antiguo, seguidamente se encuentra el planteamiento del problema, cuyo enfasis está en la condición actual de la CIA-FOULA, lo que da pie a la justificación de la investigación, el planteamiento de los objetivos, la metodología escogida para el desarrollo y el alcance.

Capítulo 2: Contiene el marco teórico donde se consideran aspectos teóricos fundamentales, conceptos y definiciones que guardan estrecha relación con el tema de estudio.

Capítulo 3: En este capítulo se contemplan las herramientas usadas para el desarrollo desde el Backend o lado del servidor (Framework, Sistema Operativo, hardware, lenguaje, modelos y base de datos).

Capítulo 4: En este capítulo se epecifícan los requerimientos desde el lado del Frontend, se realiza una descrpición general de los módulos del sistema, los roles dentro del mismo y la especificación de la interacción del usuario con el sistema a través de casos de uso, diagramas de flujo y finalmente capturas de la aplicación en ejecución de las funcionalidades descritas.

Capítulo 5: En este capítulo se expresan las conclusiones de la investigación y las recomendaciones para trabajos futuros.

1.1. Antecedentes

En esta sección se consideran aspectos teóricos fundamentales del estudio. Por tal motivo se incluyen estudios previos que poseen relación con la presente investigación, los cuales serán descritos a continuación en orden temático y cronológico iniciando con sistemas de gestión de citas a través de páginas web, seguido de sistemas de gestión de citas a través de aplicaciones móviles, continuando con sistemas de gestión de citas multiplataformas (Web + App) y finalmente Teleodontología. El orden cronológico está dado de los más recientes a los más antiguos.

1.1.1. Sistemas de gestión de citas a través de páginas web

Con el propósito de gestionar la asignación de citas previas a los usuarios, Segovia (2019), implementó un sistema de gestión de citas médicas para un centro de salud. En dicho proceso se hizo uso de las metodologías ágiles, donde se realizó una primera fase de ingeniería de requisitos y diseño e implementación sobre el conjunto de la aplicación obteniendo así sus características principales, una visión general de cómo sería la navegación y unos primeros bocetos de cómo sería la interfaz junto con la configuración del servidor. Posteriormente se itera sobre cada componente o característica principal dando lugar a las funcionalidades del proyecto establecidas. Finalmente, se realizaron pruebas sobre todas las funcionalidades y características de la aplicación. De esta manera se obtuvo un sistema funcional con todos los módulos propuestos que se puede adaptar a otros ámbitos distintos al sector.

De forma similar, Pinela (2018), diseñó una página web de gestión de citas médicas para el centro de salud Trinitaria 2, con el propósito de ofrecer a los usuarios de la institución una forma más eficiente y eficaz de agendar una cita médica a través de sus dispositivos electrónicos con acceso a internet. Este proyecto se sujetó a la investigación descriptiva ya que con ella se pudo conocer las propiedades y características de los objetos de estudios y así poder alcanzar una óptima percepción del estado actual de la problemática, además es explicativa ya que mediante este tipo de investigación se podrá manifestar de una manera adecuada todos los datos obtenidos en la misma, por último, la correlacionar servirá para identificar el grado de satisfacción de los pacientes. Como producto final se obtuvo: Diagrama de casos de uso general, los requerimientos funcionales y no funcionales del sistema, se presentó el diseño del diagrama entidad-relación de la base de datos y la propuesta del mapa de navegación del sitio web.

En este mismo sentido, Tello, Polo y Tavera (2019) desarrollaron un sistema de gestión de citas médicas para estudiantes de la Unidades Tecnológicas de Santander (UTS). El propósito de la aplicación fue facilitar la solicitud de citas médicas a los estudiantes de UTS, a través de un sistema de información, que permitiera crear un perfil de cada

estudiante y luego realizar varios procesos, como solicitar citas médicas, tratamiento o pedir una receta de los especialistas. El modelo utilizado para llevar a cabo dicho desarrollo fue el secuencial, también conocido como método de la cascada, el cual posee un conjunto de etapas que van una tras otra, las cuales se revisan de manera robusta antes de avanzar a la siguiente fase. La codificación del programa se realizó a través del lenguaje de programación PHP, Laravel Framework y un motor de base de datos en MySQL, logrando así la creación de tres roles diferentes: Administrador, paciente, especialista, donde cada usuario puede realizar su propio proceso de autenticación, además se optimizaron los tiempos de programación, cancelación y reprogramación de citas médicas para las áreas de odontología y fisioterapia de los estudiantes de las UTS.

Dentro de este contexto, Gonzales (2017) implementó un sistema web para la gestión de una clínica dental (consultorio odontológico Navarro) aplicando tecnologías open source. Ésto, Con el objetivo principal de automatizar todos los procesos relacionados con manejo de información, a fín de que la misma puedan ser establecida en una base de datos centralizada y disponible, tanto para los especialistas del centro dental como para los pacientes que acuden al mismo; lo que permitió agilizar el tiempo de respuesta en obtención de información para toma de decisiones de ambas partes. Para el proceso de investigación se utilizó la investigación documental y de campo y como metodología de desarrollo se hizo uso de las metodologías ágiles. La construcción de esta herramienta se llevó a cabo a través del framework de código abierto Laravel 5.2 mediante el cual se desarrollan servicios y aplicaciones de PHP 5 tomando como motor de base de datos MySQL. Finalmente, el sistema implementado en el centro médico dental facilitó el acercamiento de pacientes, mejorando la eficiencia y eficacia en el momento de generar citas y tener reportes actualizados de las mismas.

En este mismo orden Castillo y Morales (2016) desarrollaron un sistema web para el consultorio odontológico Denti Dana, con el propósito de generar citas de manera eficiente y ordenada, a su vez para suplir la necesidad de tener un inventario tanto de los implementos del consultorio como de los productos que ofrece a la venta. El desarrollo de la investigación se fundamenta teóricamente a partir de estudios

realizados sobre: metodología de desarrollo OPEN UP, sistemas web, arquitectura MVC (modelo vista controlador), lenguajes PHP/5.6 y HTML5 con Java script. Finalmente se diseñó un sistema web constituido por varios módulos los cuales se ocuparán de la administración de los reportes, las citas, el inventario y los usuarios. La implementación de la arquitectura MVC permitió de manera ordenada la construcción de los módulos planteados. Para el desarrollo del módulo de reportes se incorporaron las librerías Dompdf y Highcharts las cuales permitieron la generación de PDF y gráficas mejorando la presentación de los reportes. El uso del framework de Bootstrap 3 dio como resultado la creación de una interfaz amigable y sencilla, también el uso de calendarios para la toma de fecha la cual proporciona al usuario confianza en el sistema web, lo que produce que éste lo usen regularmente y cumpla con el fin para el cual fue creado. Otra incorporación fue las API de Jquery y Amaran.js las cuales permitieron personalizar y animar los componentes de html, dando un sistema web intuitivo y fácil de manipular para el usuario. Jquery también dio como resultado una comunicación más rápida y óptima entre el usuario y el servidor por medio de la tecnología de Ajax.

1.1.2. Sistemas gestión de citas a través de aplicaciones móviles

En Cuenca Ecuador Avila (2018), desarrolló un sistema web con app móvil para gestión de citas médicas y estadísticas. Para cumplir este propósito se utilizó la metodología Rational Process Unified (RUP) para la estructura del software, el lenguaje de programación Java, la base de datos SQL para el almacenamiento de la información, y la aplicación móvil con Android Studio. El sistema web obtenido es fácilmente adaptable al medio y modificable de acuerdo a las necesidades específicas de cada empresa; además de tener una interfaz gráfica amigable.

En esta misma línea temática Huaylinos (2017), haciendo uso de las metodologías ágiles, implementó una aplicación móvil para la gestión de citas médicas en la clínica

dental PERIO DENT - Huncayo, obteniendo una aplicación que permitió automatizar el proceso de agendar una cita, lo que repercutió significativamente en la disminución de los tiempos usados para llevar a cabo dicho proceso. La aplicación permitió a los usuarios consultar la disponibilidad de los médicos y agendar desde cualquier lugar haciendo uso de su dispositivo móvil. Para el desarrollo de este sistema se usó la siguiente técnología: CodeIgniter como framework, se trabajó con el (MVC) en su arquitectura lógica y en la física con un modelo de servicio de almacenamiento en la nube híbrida, se utilizó la arquitectura Cliente-Servidor para el desarrollo de la aplicación, lenguaje de programación JAVA, servidor de base de datos MySQL, sistema operativo Linux o Microsoft Windows 8, cliente FTP FiliZilla Client, diseñador de base de datos MySQLWorkBench, IDE de manejo de datos MySQL Control Center Versión 0.9.2, IDE de desarrollo Eclipse/Netbeans/Android Studio.

De forma similar Miranda (2015), realizó el análisis y diseño de una aplicación móvil para citas en consultorios odontológicos particulares de la ciudad de Piura Perú. Esta solución consiste en una aplicación móvil que permita a los pacientes obtener un mejor servicio reservando sus citas de manera sencilla desde la comodidad de su hogar o trabajo, y en cuanto al odontólogo tener una agenda organizada con todos sus pacientes. De esta forma el odontólogo y los pacientes encontrarán una mejor organización y reducción de tiempos valiosos de espera. Se determinó el alcance del producto y del proyecto, así como también los requerimientos funcionales y no funcionales que deberían tener en cuenta para el diseño de la aplicación móvil. Para un mejor análisis del proyecto, se realizó un estudio de factibilidad técnica, económica y operativa. Luego de la investigación realizada (encuesta a 37 Odontólogos en consultorios particulares y 100 pacientes) se determinó que el 78.38 % de los Odontólogos les gustaría implementar su sistema de información, y al 67.57 % le gustaría que fuese mediante una aplicación móvil. Además, el 74 % de los pacientes prefirió reservar a través de una aplicación móvil. Por otra parte, se determinó que su inversión sería de USD 2.506. Luego de estos resultados se concluyó que el proyecto es viable.

Además, Rodríguez (2015), realizó el diseño de un prototipo de aplicación orientada a dispositivos móviles para la administración de citas médicas por pacientes de entidades de salud odontológica. La metodología utilizada es la adaptación del lenguaje UML al diseño de prototipos de aplicación móvil. Finalmente, se adaptaron factores de calidad de software al ambiente móvil, entregando un diseño de prototipo de aplicación que muestra un producto que cumple con los requerimientos definidos para ser programable, funcional y mantenible.

1.1.3. Sistemas de gestión de citas multiplataformas (Web + App)

En cuanto a los sistemas de gestión multiplataforma López y Rodríguez (2020), implementaron un aplicativo multiplataforma para la gestión de citas médicas en la clínica odontológica Dental Care, haciendo uso del modelo incremental para el desarrollo del sistema y con las herramientas *visual.Net*, *Xamarin* y SQL server para el aplicativo móvil, para el levantamiento de requerimientos se realizó una encuesta y una entrevista dirigida a los doctores y pacientes del centro odontológico con el propósito de conocer las verdaderas necesidades a suplir. Se concluyó que la implementación del proyecto representó un avance tecnológico dentro de la institución, en el que se logró la automatización de los procesos dentro del departamento odontológico, a su vez permitió evaluar y demostrar a los pacientes los beneficios que brinda agendar citas por medio de una app, en cuanto a ahorro de tiempo y costos.

Por su parte, (Llivicura & Ulloa, 2018), realizaron el diseño y despliegue de una plataforma integral de telemedicina en una nube privada basada en *OpenStack*, el proyecto se dirigió a diseñar la arquitectura y desplegar una plataforma de telemedicina enfocada a la odontología, y elaborar un escenario de pruebas, a fin de obtener información relacionada con el uso de los recursos de hardware, la plataforma pretende brindar un sistema de teleconsultas a los pacientes para así poder realizar de manera eficiente el diagnóstico, tratamiento, prescripción y seguimiento del paciente. Para ello

hicieron uso de la metodología Scrum, debido a que les permitió dividir al proyecto en fases con respecto a los objetivos específicos. La arquitectura propuesta se dividió en cuatro capas: La primera capa la compone una nube privada, bajo el software OpenStack, dentro de este módulo se crearon instancias en las que se encontraron la base de datos relacional MySQL, la segunda instancia un servidor Web para la interfaz de acceso a los clientes, la instancia de servicios de comunicaciones personales *WebRTC*. Además de la estructura de software y programación del sistema. La segunda capa es el servicio de comunicaciones cliente servidor, a través de redes privadas virtuales VPN que permitió conexiones desde los clientes hacia los servidores alojados en *OpenStack*. La tercera capa la compone el sistema de estadísticas, que permitió contar con información procesada para medir el impacto del proyecto. La cuarta capa la compone el sistema de seguimiento proactivo de medicación de los pacientes, el que está enfocado generar mensajes de texto para recordar el tratamiento y posología.

1.1.4. Teleodontología

Mutis, Morón y Medina (2020), como parte del equipo interdisciplinario de COVID-19 de la Asociación Latinoamericana de Odontopediatría, lideraron el desarrollo de un artículo denominado: Teleodontología: Aplicación a la Odontopediatría durante la pandemia del COVID-19. En el mismo se exponen una serie de recomendaciones para atender y dar seguimiento a pacientes odontológicos infantiles (Odontopediatría), tomando en cuenta los nuevos escenarios que trajo consigo la presencia del COVID-19. En este sentido, como primer paso en la atención surge un cambio importante recomendado, el cual consiste en la inclusión de la atención a distancia o telemedicina, en el caso de la Odontopediatría: Teleodontología, involucrando atención telefónica, por medios digitales o plataformas virtuales, utilizando toda la tecnología disponible para poder dar un diagnóstico, orientación terapéutica, seguimiento de los casos, y determinar los casos que requieren atención presencial. El artículo hace mención al uso de la teleodontología como primer paso en la atención a los pacientes, el cual consiste en una pre-consulta virtual (asincrónica), en el que se envía un cuestionario previo por medios digitales con el propósito de prepararse para la consulta sincrónica

(en tiempo real) y una guía de consulta virtual que debe ser firmada por el paciente, si acepta las estipulaciones de la misma. Seguidamente se plantea una consulta virtual, en la que se puede dar un diagnóstico presuntivo y orientaciones terapéuticas, además, se recomienda realizar un registro físico de la consulta con todos los datos e historia clínica del paciente, como también, todo lo referente al episodio o consulta actual. Post consulta virtual (sincrónica y asincrónica), el especialista podrá definir el siguiente paso a seguir del paciente, si requiere de una atención presencial de urgencia o si, por el contrario, se puede continuar con un seguimiento virtual. Finalmente, el artículo hace mención a los tipos de afecciones emergentes (que requieren atención presencial) y realiza recomendaciones para la prescripción de medicamentos haciendo uso de la telemedicina.(Mutis et al., 2020)

En este mismo sentido Morón (2021), en su artículo: La Teleodontología una herramienta fundamental en tiempos de pandemia y post COVID-19, su utilidad en las diferentes especialidades odontológicas, hace mención de la importancia del uso de las tecnologías de la información para la atención médica odontológica a distancia, con el propósito de evitar la propagación del COVID-19, el mismo también hace énfasis en que la teleodontología no solo es relevante en tiempos de pandemia, sino aún, cuando los escenarios post pandemia permitan de manera más segura las consultas presenciales, debido a que la telemedicina no solo se ha convertido en una medida emergente, sino que además, se presenta como un medio alterno de economía, seguridad, comodidad y fiabilidad. En este sentido, el autor hizo una revisión de literatura con el objetivo de describir el uso y aplicación de teleodontología en las diferentes especialidades odontológicas, entre las cuales señala: Endodoncia, en la que, según el autor, las lesiones periapicales son la patología más común, mismas que pueden ser evaluadas adecuadamente usando métodos de teleodontología y se puede diseñar un plan necesario para el tratamiento endodóntico adecuado a las lesiones.

Otra especialidad mencionada es la Odontopediatría, para la cual se registran diagnósticos de caries detectados haciendo uso de métodos de teleodontología. Por otra parte, tenemos la Periodoncia la cual, al ser detectada y diagnosticada a tiempo

pueden prolongar la vida de los dientes en la cavidad bucal, el uso de teleodontología para realizar televigilancia, puede reducir considerablemente las visitas presenciales al especialista. En este contexto también encontramos, la Ortodoncia, en la que se ha usado teleodontología para evaluar la necesidad de ortodoncia y para proporcionar instrucciones a los pacientes que estén en dicho tratamiento. Finalmente hace mención al caso de las cirugías orales y/o maxilofaciales, el uso de la teleodontología permite monitorear las condiciones postoperatorias, por otra parte, hace mención a las herramientas de teleodontología Mouth Watch y Dental Monitoring, donde la primera consiste en un software basado en la nube que permite que toda la información del paciente cargada previamente, se integre perfectamente con la conexión segura de transmisión en vivo junto a las imágenes generadas por la cámara intraoral Mouth Watch. La segunda es una aplicación basada en inteligencia artificial que implementa un sistema dinámico y agradable que facilita la relación entre el odontólogo y el paciente.

Asimismo Treichler (2021) realizó un estudio de mercado acerca de las mejores compañías o empresas que prestan servicios de teleodontología, para realizar dicha selección tomó en cuenta, los siguientes factores: Seguridad, solo se realizó la selección de aquellas plataformas que cumplen con la norma de confidencialidad de la Ley de Portabilidad y Responsabilidad de Seguros Médicos (Health Insurance Portability and Accountability Act, HIPAA) y que protegen la información del paciente con cifrado de datos y otras medidas de seguridad. Otro factor importante que se tomó en cuenta fue, los métodos de comunicación, priorizando aquellos que ofrecen múltiples formas de comunicación, incluyendo videos y chats telefónicos, para comunicación sincrónica y mensajería de texto o correo electrónico para consultas asincrónicas. Finalmente hace mención al factor de conveniencia, el cual consiste en dar prioridad a aquellas plataformas que permiten a los pacientes que programen sus citas a conveniencia o soliciten atención a pedido, dando especial valor a aquellas que cuentan con aplicaciones móviles.

En este sentido, las 10 mejores plataformas seleccionadas son las siguientes: 1. *Live Dentist* quien según la autora es la mejor en líneas generales, 2. *DentalChat*

, ofrece los servicios más asequibles(comentarios del autor), 3. *Dentaractive*, ofrece servicios 24/7 y atiende pacientes sin previa cita, 4. *Virtudent, Inc.*, mejor plan dental para empleados al aceptar la mayor parte de los planes de seguros dentales y ofrecer servicios odontológicos virtuales y presenciales hasta el lugar de trabajo, 5. *Aspend Dental* el mejor en cobertura de seguros, 6. *Toothpic* ofrece los mejores recursos educativos para los pacientes que desean obtener más información sobre el cuidado bucal, 7. *The TeleDentists* además, de ayudar a los pacientes de forma virtual, también puede ayudar a los pacientes a conectarse con un dentista local en persona para garantizar que reciban una atención integral, 8. *Dentulu*, ofrece servicios virtuales de odontología y en caso de requerirlo, envía un odontólogo al hogar o al lugar que mejor se adapte a las necesidades del paciente, 9. *Smile Virtual*, según la autora, la mejor en cuanto a la odontología cosmética, finalmente, 10. *Alpha Dental Excellence*, ofrece consultas virtuales gratuitas, lo que permite acceder a segundas opiniones expertas, sin tener que acudir personalmente a un consultorio.

1.2. Planteamiento del Problema

El crecimiento vertiginoso del uso de las Tecnologías de la Información y Comunicación (TIC) ha penetrado y cambiado profundamente la forma en la que la humanidad realiza sus actividades cotidianas, lo que a su vez ha obligado a las diversas organizaciones a implementar herramientas tecnológicas para optimizar la gestión de sus procesos. Las organizaciones del sector salud no escapan de esta realidad, pues la aplicación de herramientas informáticas han mejorado el quehacer médico, permitiendo la automatización de sus procesos y el acceso a la información de manera rápida y oportuna (López y Rodríguez, 2020). Dentro de este contexto, surge la práctica de los servicios de salud y de sus actividades inherentes de forma remota, tales como: educación, formación, gestión, y dirección de sistemas de salud, haciendo uso de sistemas basados en TIC, a dicha práctica se le denomina como telesalud. En este mismo sentido, nace el término Telemedicina (TM), definido como la parte clínica de la telesalud, dedicada a la práctica médica (prevención, diagnóstico, tratamiento,

monitoreo) a distancia, la que a su vez puede ser en tiempo real (forma sincrónica) o diferido (forma asincrónica). (Chá, 2020).

Como parte de las especialidades que han adoptado el uso de la TM, se encuentra la odontología, cuya práctica a distancia está definida como teleodontología, y fue planteada por el ejército estadounidense en 1994 para brindar asistencia a sus tropas que se encontraban desplegadas en el mundo. (Segura & Atoche, 2021). Si bien es cierto que el origen de las prácticas de teleodontología se remonta a algunas décadas atrás como medida alternativa a las consultas presenciales, no obstante, con la declaración de la pandemia global del COVID-19 por parte de Organización Mundial de la Salud (OMS), aumentó la necesidad de implementar herramientas tecnológicas que mitiguen dicha problemática. Para esto, las organizaciones sanitarias tomaron medidas alternas para adecuarse a la nueva normalidad y poder seguir ofreciendo asistencia médica de forma oportuna y eficiente, procurando el cuidado tanto de los pacientes como del personal sanitario.Teniendo en cuenta que el virus causante de la enfermedad (SARS-CoV-2), infecta las células del sistema respiratorio y es trasmitido a través de gotitas respiratorias de persona a persona, la OMS recomienda tomar medidas estrictas de prevención, entre ellas el distanciamiento físico. Por tal motivo con la declaración de la emergencia, se creó una necesidad vital a nivel mundial de generar soluciones de conexión entre los pacientes y los especialistas de las distintas áreas del sector salud.

Actualmente la asistencia odontológica presencial, constituye un alto riesgo de contagio y propagación del virus causante de la COVID-19, ya que las prácticas dentales incluyen el uso de instrumentos que aerosolizan agua, saliva, sangre y otros desechos en la implementación de algunos tratamientos, lo que expone a los profesionales dentales y sus pacientes. Sin embargo, ante la latente necesidad de este tipo de atención, los profesionales han hecho uso de herramientas tecnológicas de comunicación como (Whatsapp, Telegram, Facebook, zoom, Google meet, entre otros), con el fin de realizar el seguimiento de tratamientos, prescripción de medicinas para el dolor o infecciones, realizar triaje dental y emitir diagnósticos basándose en el intercambio de imágenes, vídeos y/o fotografías. (Segura & Atoche, 2021).

En este escenario, llamado "nueva normalidad", se hace necesario la generación de protocolos, guías y recomendaciones para los profesionales que brindan atención odontológica. En el caso de Latinoamérica, la Asociación Latinoaméricana de Odontopediatría generó una guía de recomendaciones adecuada a la práctica odontológica en tiempos de pandemia, con el fin de orientar a los profesionales de la rama en la ejecución de sus actividades, como tamizaje de consultas, es decir, poder hacer uso de la teleodontología para conocer cuales casos requieren necesariamente la atención presencial por urgencia, dependiendo de las regulaciones locales. De acuerdo con (Asociación Latinoaméricana de Odontopediatría, 2020), ésta serie de recomendaciones "constituyen una guía de orientación, sin sustituir regulaciones locales, protocolos, específicos, ni leyes vigentes en cada país", además, las mismas son susceptibles a cambios, de acuerdo a las modalidades que se vayan generando en las distintas etapas de la pandemia.

Con respecto a las leyes asociadas a los protocolos de salud, en Venezuela se creó la ley de telesalud en 2015, con el propósito de establecer los lineamientos, principios, bases, regulación y control del funcionamiento de la red de telesalud, a fin de garantizar el acceso, cobertura y calidad de atención a la población, haciendo uso de las TIC, especialmente de aquellas herramientas desarrolladas bajo el contexto de software libre. (Asamblea Nacional de Venezuela, 2015) Sin embargo, la misma es desconocida por número considerable de odontólogos participantes de la encuesta realizada por (Betancourt & Bermúdez, 2021), en su estudio acerca de la percepción de los odontólogos sobre la viabilidad de la aplicación de teleodontología en Mérida, Venezuela, lo que indica la poca difusión y aplicación de la misma. Tomando en cuenta el mencionado estudio, es importante destacar que se llegó a la conclusión de que los odontólogos que hicieron parte de la encuesta, consideran que la aplicación de teleodontología es viable en Mérida, Venezuela.

Con base en lo anteriormente mencionado, en cuanto a la necesidad de hacer uso de herramientas tecnológicas, es menester mencionar la importancia de los sistemas de

información en el sector salud, cuyos tres objetivos principales de acuerdo con (Valeri, 2016) son: automatizar los procesos operativos, proporcionar información que apoye la toma de decisiones y lograr ventajas competitivas a través de su uso. No obstante, la gestión de citas médicas continúa siendo un proceso manual y presencial en múltiples centros de salud en Venezuela, en algunos casos "más favorecidos" se realizan vía llamadas telefónicas, mensajería, correo electrónico y/o mensajería instantánea, lo que implica en la necesidad de contratación del personal de atención lo que a su vez aumenta la posibilidad de errores en el proceso como: asignar 2 citas en un mismo horario, duplicidad de información en el registro, un prolongado tiempo de espera al teléfono, gastos de traslado del paciente para agendar cita, etc. sumado a esta realidad converge la aparición de la COVID-19, la cual pone de manifiesto no sólo la necesidad de automatizar el agendamiento de citas, sino también la de priorizar y filtrar las mismas de acuerdo al motivo de la consulta y también el estado del paciente con respecto a los síntomas asociados a la presencia de la COVID-19.

Debido al alto riesgo de contagio que representa la misma, es actualmente imprescindible la implementación de sistemas de información web, que permiten la automatización de procesos en línea, en este caso, para la realización de pretriaje haciendo uso del teletriaje, con el fin de priorizar la demanda asistencial urgente, evitando contagio por aglomeraciones de pacientes que no requieren atención presencial de urgencia y descartando previamente la presencia de contagio en alguno de los pacientes. Sin embargo, en la actualidad, los centros públicos de atención odontológica, incluyendo universidades y en especial la Universidad de Los Andes (ULA) carecen de herramientas que le permitan realizar de forma segura y eficiente la práctica de teleodontología. Además, es importante señalar que el escenario que brinda la pandemia, sigue siendo lleno de incertidumbre, pues la COVID-19 sigue cobrando vidas en todo el mundo a pesar de todas las medidas de atención que se toman actualmente como la vacunación. Por tanto, el riesgo de contagio y propagación sigue estando presente para los profesionales de la salud como para los pacientes, y en un alto índice para lo especialistas en el cuidado bucal. En este sentido la Clínica Integral del Adulto (CIA) de la Facultad de Odontología de la Universidad de Los Andes

(FOULA), requiere de un sistema informático que le permitan continuar brindado sus servicios de una manera oportuna, eficiente y pertinente al estado emergencia actual de la nación y el estado Mérida.

Es por ello que a través del presente trabajo de grado se pretende crear un sistema informático web que permita aplicar teleodontontología realizando teletriaje y el agendamiento de citas odontológicas vía online, con el propósito de mitigar la propagación del COVID-19, y brindar una asistencia odontológica oportuna y eficiente.

1.3. Justificación

El uso de las tecnologías de la información ha roto las barreras de la comunicación y contribuido al desarrollo social, mejorando el quehacer de múltiples organizaciones y sectores de la sociedad, sin escapar de ellos el sector salud. Sin embargo, en plena era de la información, algunos centros de salud en Venezuela carecen de herramientas tecnológicas para la ejecución de sus procesos, ocasionando demoras, errores, duplicidad de esfuerzos y gastos innecesarios. Tal es el caso de la CIA-FOULA, la cual realiza sus procesos para agendar citas de forma manual, provocando la asistencia presencial de los pacientes al centro de atención para programar citas o procurando ser atendidos en la misma ocasión, lo que generalmente desemboca en largos tiempos de espera, aglomeración de múltiples pacientes en la sala y quejas de mala atención.

A este contexto se suma la pandemia por la COVID-19, poniendo de manifiesto la necesidad no sólo de aportar agilidad al proceso de agendar citas, sino también la de priorizar la demanda asistencial con el propósito de evitar las aglomeraciones para mitigar la propagación del virus. En este sentido, otro agente de preocupación es conocer el estado de salud del paciente con respecto a los síntomas asociados al COVID-19, antes de la consulta odontológica, ya que, la aplicación de los tratamientos odontológicos presuponen un alto riesgo de contagio vía aérea, debido a la aerosolizacion de agua, agua, saliva y otros desechos en los cuales puede estar alojado el virus, exponiendo no sólo al personal especializado sino también a los demás pacientes y por

consiguiente al resto de la sociedad que los rodea.

Ante el escenario actual que atraviesa la salud a nivel mundial, aportar soluciones en este campo es de carácter vital. Es apremiante que los profesionales de la salud se apropien de las herramientas tecnológicas para establecer puentes de conexión con sus pacientes y seguir brindando sus servicios de manera oportuna, eficiente y segura. De igual forma, es necesario que los centros de formación y/o universidades hagan uso de estas herramientas para continuar con su proceso preparación, debido a que los estudiantes requieren la realización de actividades clínicas. Como es el caso de los estudiantes de la FOULA, quienes actualmente tienen suspendidas sus prácticas académicas, sin embargo, se proyecta al reinicio de las mismas, para las cuales es importante implementar modalidades seguras.

Con base en lo anteriormente expuesto, se considera pertinente e importante el desarrollo de un sistema web para la aplicación de teleodontología en la CIA-FOULA, con el fín de gestionar el agendamiento de citas y realizar teletriaje a los pacientes que soliciten atención en la misma. Finalmente se supone un beneficio metodológico al establecer una base de investigación para futuras iniciativas de despliegue en otras instituciones y/o especialidades del sector salud.

1.4. Objetivos

1.4.1. Objetivo General

- Desarrollar un sistema web para la aplicación de teleodontología en la Clínica Integral del Adulto (CIA) de la Facultad de Odontología de la Universidad de Los Andes (FOULA).

1.4.2. Objetivos Específicos

- Describir el comportamiento del sistema a través de un modelo que especifique los diferentes casos de uso
- Implementar metodologías ágiles para el desarrollo del sistema.
- Implementar un sistema web para la gestión de citas y aplicación de teletriaje, a los pacientes que solicitan atención en la CIA-FOULA.
- Evaluar el o los prototipos del sistema de gestión tomando en consideración la experiencia de los usuarios

1.5. Metodología

En primer lugar, se realizará una síntesis conceptual de las investigaciones y trabajos realizados previamente acerca de sistemas web para la aplicación de teleodontología y la gestión citas médicas online. Para el desarrollo de la aplicación se utilizará la metodología ágil Scrum, cuyo marco de trabajo o modelo de gestión permite desarrollar productos de una forma iterativa e incremental, esta estrategia iterativa posibilita regresar a ciertos puntos para refinar o rediseñar dependiendo de la magnitud del error.

A continuación, se definen los pasos seguidos para el desarrollo del proyecto.

1. Realizar una descripción detallada del sistema: En este apartado se llevará a cabo una entrevista con parte del personal docente y estudiantes de la cátedra, para conocer el proceso que se ejecuta actualmente, posteriormente se realizará un modelado del sistema e identificaran las funcionalidades que lo conformaran.
2. Diseñar y configurar un servidor WEB bajo una plataforma Cliente/Servidor para desarrollar un sistema web que permitirá la aplicación de teleodontología en CIA - FOULA.

3. Se creará un repositorio web o controlador de versiones para alojar el código del sistema.

4. Se seleccionará la tecnología a emplear para el desarrollo (Lenguaje, Framework, etc).

5. Se aplicará la metodología Scrum para el desarrollo de cada funcionalidad, módulos o subsistemas que se identifiquen en el modelo: En esta sección se definirán las iteraciones (tareas), así como su duración para estimar su proceso de desarrollo. Seguidamente se iniciará la codificación, pruebas y correcciones en cada iteración, en caso de ameritarlo.

6. Una vez culminado el desarrollo de cada iteración, se probará y depurará el sistema integrado.

7. Se realizará el despliegue del sistema en el servidor seleccionado por la CIA de la FOULA.

1.6. Alcance

El desarrollo del sistema web esta orientado a gestionar las citas odontológicas de la CIA-FOULA, con el fin de que los pacientes y/o usuarios puedan acceder para programar una cita, haciendo uso de cualquier dispositivo inteligente (computadora, smartphones, tablets, etc) con conexión a internet, además el sistema debe permitir priorizar las citas con carácter de urgencia y realizar un previo triaje virtual (teletriaje) para descartar posibles casos de COVID-19 a través de un formulario que el paciente debe responder.

Capítulo 2

Marco Teórico

En este capítulo se consideran aspectos teóricos fundamentales, conceptos y definiciones que guardan estrecha relación con el tema de estudio.

2.1. Arquitectura cliente servidor

Según (Marini, 2012), es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. En éste, el denominado cliente (una o más aplicaciones), realiza una o varias peticiones a las aplicaciones servidores, quienes ejecutan acciones para atender dichas solicitudes, permitiendo así diversificar el trabajo que realiza cada aplicación, de forma que los clientes no se sobrecarguen desempeñando las funciones que le fueran proporcionadas de forma directa. En este sentido, la arquitectura de este modelo distribuye la capacidad del proceso entre los clientes y los servidores, además aporta ventajas de tipo organizativo debido a que centraliza la gestión de la información y la separación de las responsabilidades, clarificando el diseño del sistema.

De acuerdo con (Marini, 2012) este modelo establece los siguientes conjuntos que comúnmente se denominan lógicas:

- Lógica de presentación: Esta lógica es la responsable del control de todos los aspectos relacionados con la interacción entre el usuario y la aplicación.

- Lógica de negocio: Es la lógica de la aplicación que controla la secuencia de acciones y fuerza el cumplimiento de las reglas del negocio propias de cada empresa, además, asegura la integridad de las transacciones de las operaciones necesarias que haya que realizar para que se cumplan dichas reglas.

- Lógica de datos: En este conjunto entran los procesos encargados del mantenimiento de los datos, de garantizar las reglas de integridad referencial establecidas, así como la gestión de las transacciones, dichas tareas son realizadas por un sistema de gestión de base de datos relacionales como SQL Server, Oracle, MySQL, Informix, etc.

Los modelos más frecuentes de aplicaciones según Ibarra, citado por (Quintero, 2015) son los siguientes:

- Mono Capa: Son aquellas aplicaciones donde, tanto ellas, como los datos que usa, se encuentran alojados en la misma máquina y a su vez son administradas por la misma herramienta.

- Dos Capas: Consiste en una arquitectura Cliente/Servidor con dos capas, en la que el cliente implementa la interfaz y el gestor de base de datos funciona como servidor tratando las peticiones recibidas desde el cliente.

- Tres Capas: Mantiene la arquitectura Cliente/Servidor añadiendo una nueva capa entre el cliente y el servidor, donde se implementa la lógica de la aplicación, de esta manera el cliente es básicamente una interfaz aislada del acceso a los datos. Dicha lógica puede ser alojada en el cliente, en el servidor de base de datos o en otro servidor, debido a que se posee completa libertad para escoger tal ubicación, al igual que se posee para la elección del lenguaje a utilizar. Además, cada uno de los componentes que forman esta arquitectura de tres capas, se separa en una sola entidad, lo que permite implementar los componentes de una manera más flexible. Todas las peticiones del o los clientes se controlan en la capa intermedia, esto con el fin de que el cliente no tenga inconvenientes de comunicación producto a la instalación de controladores(drivers) adicionales.

2.2. Modelo Vista Controlador (MVC)

Es un patrón de diseño (de software) que se encarga de separar la lógica de negocio de la interfaz de usuario y es el mas utilizado en aplicaciones web, framework, etc, ya que facilita la funcionalidad, mantenibilidad, y escalabilidad del sistema, a la vez que ayuda a no mezclar lenguajes de programación en el mismo código, evitando el conocido¸"código espagueti". (Robles, 2013).

2.2.1. Niveles de abstracción del MVC:

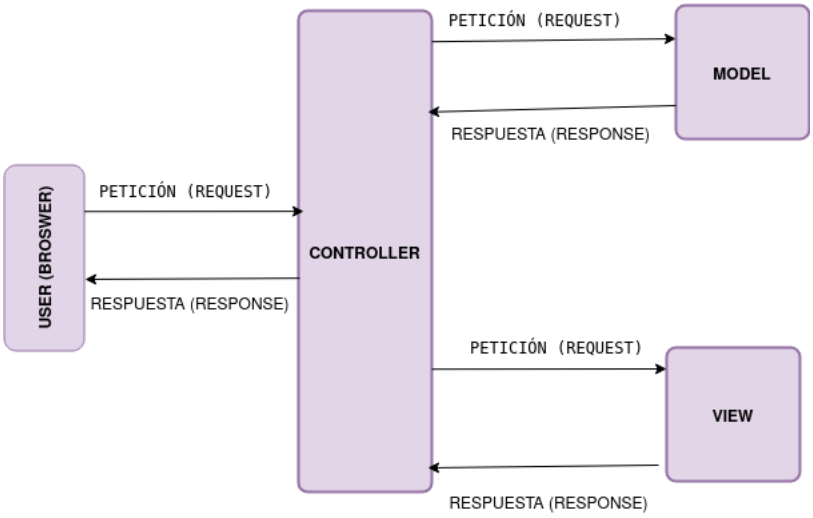
Modelo: Es la lógica de negocios. Es decir las clases y métodos que se comunican directamente con la base de datos. Las peticiones de acceso o manipulación de información llegan al modelo a través del controlador.

Vista: Es la encargada de mostrar la información al usuario, de forma gráfica y legible. Por tanto requiere del modelo la información que debe representar como salida.

Controlador: Es el intermediario entre la vista y el modelo, se encarga de controlar las interacciones del usuario en la vista, es decir, responde a eventos e invoca peticiones al modelo cuando se hace alguna solicitud sobre la información, y los devuelve de nuevo a la vista para que esta los muestre al usuario.

En líneas generales el funcionamiento básico del patrón MVC, puede resumirse de la siguiente manera:

Figura 2.1: Modelo Vista Controlador



Fuente: Elaboración propia.

1. El usuario realiza una petición.
2. El controlador captura la petición.
3. Hace la llamada al modelo correspondiente.
4. El modelo sera el encargado de interactuar con la base de datos.
5. El controlador recibe la información y la envia a la vista.
6. Se devuelve La vista seleccionada al controlador.
7. El controlador devuelve la vista que contiene la información extraída del modelo al usuario.

2.3. Bases de datos

Una base de datos es un conjunto de datos almacenados en memoria externa, que están organizados mediante una estructura de datos. Las mismas son diseñadas y creadas para satisfacer los requisitos de información de una empresa u otro tipo de organización tal como un hospital o una universidad. Una manera de percibirlo es como

un gran almacén de datos que se define y se crea una sola vez y donde se integran los datos con una mínima cantidad de duplicidad, de modo que puede ser utilizada al mismo tiempo por distintos usuarios de los distintos departamentos de la organización. La base datos no sólo contiene los datos, sino que además, almacena una descripción de dichos datos a la cual se le denomina metadatos, estos metadatos se almacenan en el diccionario de datos o catálogo y es lo que permite que exista independencia de datos lógica-física. (Marquéz, 2011).

2.3.1. Sistema de gestión de bases de datos

El sistema de gestión de base de datos(SGBD) es una aplicación que permite a los usuarios definir, crear y mantener la base de datos, además de proporcionar un acceso controlado a la misma. La definición de la base de datos se realiza mediante un un lenguaje de definición de datos, que permite especificar la estructura y el tipo de datos, así como las restricciones sobre los mismos. Por su parte las acciones de inserción, actualización, eliminación y consulta de datos se realizan mediante un lenguaje de manejo de datos, de los cuales existen dos tipos: procedurales y no procedurales, los procedurales manipulan la base de datos registro a registro y se especifican que operaciones se deben realizar para obtener los datos resultado, mientras que los no procedurales operan sobre conjuntos de registros y en éstos se especifica que datos deben obtenerse sin decir como hacerlo. El lenguaje procedural más utilizado es el SQL (Structured Query Language) que, de hecho, es un estándar y es el lenguaje de los SGBD relacionales (Marquéz, 2011).

De acuerdo con (Marquéz, 2011) un SGBD proporciona un acceso controlado a la base de datos mediante:

- Un sistema de seguridad que impide el acceso a la base de datos a usuarios no autorizados.
- Un sistema que mantiene la integridad y consistencia de los datos.
- Un sistema de control de concurrencia que permite el acceso compartido a la base de datos.

- Un sistema de control de recuperación que restablece la base de datos en caso de que exista un fallo en el hardware o el software.
- Un diccionario de datos o catálogo, accesible por el usuario, que contiene la descripción de los datos de la base de datos.

Es importante destacar que no todos los SGBD cuentan con todas las funcionalidades antes descritas, ya que los mismos dependen de los productos, existen algunos de ellos en los que se requieren todas, menos ó más de las citadas, en conclusión éstos se pueden ajustar a las dimensiones y requerimientos del producto.

(Marquéz, 2011) describe las siguientes ventajas de los SGBD:

- Control sobre la redundancia de datos.
- Control sobre la consistencia de datos.
- Compartición de datos.
- Mantenimiento de estándares.
- Mejora en la integridad de datos.
- Mejora en la seguridad.
- Mejora en la accesibilidad a los datos.
- Mejora en la productividad.
- Mejora en el mantenimiento.
- Aumento de la concurrencia.
- Mejora en los servicios de copias de seguridad de recuperación ante fallos.

No obstante, la integración de los datos y la existencia del SGBD supone algunas desventajas como las siguientes:

- Alta complejidad, lo que requiere un conocimiento exhaustivo de la funcionalidad para sacar el máximo provecho de la misma.

- Requieren una gran cantidad de espacio en el disco duro.
- Altos costos, que varían dependiendo del entorno y funcionalidades que ofrece. Además, se pueden incurrir en gastos de equipamiento adicional para almacenamiento y en algunas ocasiones se requiere convertir la aplicación actual de la empresa u organización en un sistema de base de datos, para lo cual se requiere invertir en personal especializado.
- Prestaciones, al realizar la conversión del sistema de ficheros a los SGBD es posible que se aumenten los tiempos de respuesta, debido a que los sistemas de ficheros están diseñados para atender una aplicación específica y los SGBD están escritos para ser mas generales.
- Vulnerable a fallos, debido al hecho de que todo está centralizado en él.

2.3.2. Modelos de datos

Un modelo de datos es un conjunto de conceptos que sirven para describir la estructura de una base de datos, es decir, los datos, las relaciones entre los datos y las restricciones que deben cumplirse sobre los datos. (Marquéz, 2011, p.14).

De acuerdo con (Silberschatz, Korth & Sudarshan, 2002), los diferentes modelos que se han propuesto se clasifican en tres grupos diferentes, modelos lógicos basados en objetos, modelos lógicos basados en registros y modelos físicos, a continuacion la descripción de los más relevantes actualmente:

- Modelo entidad-relación (E-R): Está basado en una percepción del mundo real que consta de una colección de objetos básicos, llamados entidades, y de relaciones entre los objetos. Éste modelo se utiliza habitualmente en el proceso de diseño de base de datos.
- Modelo Relacional: En este modelo se utilizan un grupo de tablas para representar los datos y las relaciones entre ellos
- Modelo Orientado a objetos: Se puede observar como una extensión del modelo E-R con las nociones de encapsulación, métodos(funciones) e identidad de objeto.

- Modelo de datos relacional orientado a objetos: Combina las características del modelo de datos orientados a objetos y el modelo de datos relacional.

2.3.3. Lenguaje SQL (Structured Query Language)

Según (Marquéz, 2011) es un lenguaje estándar que permite manejar los datos de una base de datos relacional, donde la misma como se ha expresado anteriormente, esta formada por un conjunto de relaciones, dichas relaciones en SQL, se les denomina tablas, donde cada columna representa los atributos (pueden ser del tipo entero, real, carácter, fecha, etc). En las tablas, se insertan filas (tuplas), que después se pueden consultar, modificar o borrar.

De acuerdo con (Pinto, 2013) el lenguaje SQL está compuesto por comandos, cláusulas, operadores y funciones de agregado, los cuales se combinan en las instrucciones para crear, actualizar y manipular las bases de datos.

Comandos SQL

Tabla 2.1: Comandos SQL

Comandos DLL	Comandos DML
CREATE	SELECT
DROP	INSERT
ALTER	UPDATE
	DELETE

Fuente:Elaboración Propia

- Comandos DLL: Permiten crear y definir nuevas bases de datos, campos e índices.
- Comandos DML: Permiten generar consultas para ordenar, filtrar y extraer datos de la base datos.

Cláusulas SQL

Las cláusulas realizan funciones de una instrucción SQL, estas a su vez son condiciones de modificación utilizadas para definir los datos que desea seleccionar o manipular.

Tabla 2.2: Cláusulas SQL

Cláusulas SQL	
FROM	Permite especificar la tabla de la cual se van a seleccionar los registros.
WHERE	Especifica las condiciones que deben reunir los registros.
GROUP BY	Permite separar los registros seleccionados en grupos específicos.
HAVING	Permite realizar una restricción sobre los grupos obtenidos por la cláusula GROUP BY.
ORDER BY	Ordena los registros seleccionados de acuerdo a un orden específico.

Fuente:Elaboración Propia

Operadores Lógicos y de comparación

Tabla 2.3: Operadores lógicos y de comparación

Operadores	
Lógicos	
AND	Evalúa 2 condiciones y devuelve un valor verdadero.
OR	Evalúa 2 condiciones y devuelve el valor de verdad.
NOT	Invierte el resultado de la condición a la cual antecede.
Comparación	
<	Menor que.
>	Mayor que.
<>	Distinto de.
<=	Menor o igual que.
>=	Mayor o igual que.
BETWEEN	Permite probar fácilmente si una expresión está dentro de un rango de valores.
LIKE	Se utiliza para buscar un patrón especificado en una columna.
IN	Especifica registros en base de datos.

Fuente:Elaboración Propia

Funciones de agregado

Las funciones de agregado se usan dentro de una cláusula SELECT en grupos de registros para devolver un único valor que se aplica a un grupo de registros.

Tabla 2.4: Funciones de agregado

Funciones de agregado	
AVG	Estima el promedio de los valores de un campo determinado.
COUNT	Devuelve el número de registros de la selección.
SUM	Calcula la suma de todos los valores de un campo determinado y devuelve dicho resultado
MAX	Obtiene el valor mas alto de un campo específico.
MIN	Obtiene el valor mas bajo de un campo específico.

Fuente:Elaboración Propia

2.3.4. PostgreSQL

Es es un sistema de base de datos relacional de objetos de código abierto con más de 35 años de desarrollo activo en la plataforma central. Este amplía el lenguaje SQL combinado con muchas características que almacenan y escalan de forma segura las cargas de trabajo de datos más complicadas.(The PostgreSQL Global Development Group, 2022)

2.4. Sistemas de información

(Laudon & Laudon, 2016, p.16), plantean la siguiente definición técnica acerca de los sistemas de información: “un sistema de información es un conjunto de componentes interrelacionados que recolectan (o recuperan), procesan, almacenan y distribuyen información para apoyar los procesos de toma de decisiones y de control en una organización.”

2.4.1. Sistemas de información informáticos

De acuerdo con (Benítez, 2012), los sistemas de información informáticos, se pueden ver como un subsistema de los sistemas de información, estos primeros están basados

en computadoras para automatizar los procesos de entrada , procesamiento y distribución de la información, utilizando las ventajas ofrecidas por las tecnologías de la información y las comunicaciones. Los sistemas de información informáticos incluyen obligatoriamente el uso de hardware, software, bases de datos y redes.

2.4.2. Sistemas Web

Los sistemas ó aplicaciones web, son sistemas informáticos que se alojan en un servidor en internet o sobre una intranet, dependiendo de los requeremientos del cliente. Para acceder a estos sistemas sólo se requiere hacer uso de cualquier navegador web, desde cualquier dispositivo y cualquier sistema operativo, debido a que no ameritan instalación. Además, estos trabajan con bases de datos que permiten procesar y mostrar información de forma dinámica para el usuario y se codifican en lenguajes soportados por los navegadores web. (Baez, 2012).

2.5. Telemedicina

Como expresa (Chá, 2020), actualmente, hay consenso internacional en el que se define la telemedicina como la parte clínica de la telesalud (prevención, diagnóstico, tratamiento, monitoreo), que corresponde a la práctica médica, realizada de manera asincrónica ó sincrónica, haciendo uso de las TIC, donde uno de los actores es integrante del equipo de la salud y el otro es un médico o paciente, o ambos.

2.6. Teleodontología

Se conoce como la atención odontólogica a distancia, haciendo uso de las TIC, a través de este soporte tecnológico se puede realizar despistaje (triaje) y el manejo sintomático de casos susceptibles e identificación de los casos que requieren atención presencial. (Mutis et al., 2020).

2.7. Teletriaje

(Segura & Atoche, 2021) expresa que el triaje permite recopilar datos para determinar el tipo de atención estomatológica. En el mismo se conocen los antecedentes, el motivo de consulta, y se define, en los casos en los que se requiera, el apoyo de diferentes especialidades. En este sentido, la práctica del triaje haciendo uso de las TIC , se le conoce como Teletriaje.

2.8. Metodología Scrum

La primera vez que se usó el término Scrum para describir el contenido ágil fue en 1986 en *Harvard Business Review*, sin embargo, este término tiene su origen en el rugby, en el cual, un "scrum" o melé es una formación fija cuya función es conseguir la pelota y volver a ponerla en juego, tras una falta menor durante el juego. En este sentido, se toma dicha analogía para la gestión de proyectos de desarrollo de software con scrum, tomando en cuenta que durante la generación de estos productos pueden surgir nuevos requisitos, debido a la velocidad en la que evolucionan los sectores de negocio, así que la metodología no se formula sobre la necesidad de anticipación, sino sobre la de adaptación continua. (Palacio, 2015) .

Este modelo fue identificado y definido por Ikujiro Nonaka e Hirotaka Takeuchi a principios de los 80 y puede ser usado para la gestión de proyectos de diversa índole. Esta marco de gestión posee las siguientes características:

- Adoptar una estrategia de desarrollo incremental, en lugar de la planificación y ejecución completa del producto.
- Basar la calidad del resultado más en el conocimiento tácito de las personas en equipos autoorganizados, que en la calidad de los procesos empleados.
- Solapamiento de las diferentes fases del desarrollo, en lugar de realizarlas una tras otra en un ciclo secuencial o de cascada.

De acuerdo con (Palacio, 2015) , el marco de reglas para el desarrollo de software creado por Ken Schwaber y Jeff Sutherland ¨Scrum Development Process OOPSLA95" 1995, presenta las siguientes características:

Roles

Dueño de producto: es la persona responsable de lograr el mayor valor de producto para los clientes, usuarios y resto de implicados.

Equipo de desarrollo: grupo o grupos de trabajo que desarrollan el producto.

Scrum Master: Es el responsable del cumplimiento de las reglas de un marco de scrum técnico, propociona la asesoría y formación necesaria al propietario del producto y al equipo.

Eventos

El Sprint: se denomina sprint a cada ciclo o iteración de trabajo que produce una parte del producto terminada y funcionalmente operativa (incremento).

Reunión de planificación: reunión de trabajo previa al inicio de cada sprint en la que se determina cuál va a ser el objetivo del sprint y las tareas necesarias para conseguirlo.

Scrum diario: breve reunión diaria del equipo, en la que cada miembro responde a tres cuestiones:

1. El trabajo realizado el día anterior.
2. El que tiene previsto realizar.
3. Cosas que puede necesitar o impedimentos que deben eliminarse para poder realizar el trabajo.

Revisión de sprint: análisis e inspección del incremento generado, y adaptación de la pila del producto si resulta necesario.

Retrospectiva de sprint: revisión de lo sucedido durante el *sprint*. Reunión en la que el equipo analiza aspectos operativos de la forma de trabajo y crea un plan de mejoras para aplicar en el próximo *sprint*.

Artefactos

Pila de product (product backlog): lista de requisitos de usuario, que a partir de la visión inicial del producto crece y evoluciona durante el desarrollo.

Pila de sprint (sprint backlog): lista de los trabajos que debe realizar el equipo durante el *sprint* para generar el incremento previsto.

Incremento: resultado de cada *sprint*.

En líneas generales la metodología funciona de la siguiente manera:

Inicia con la visión general de lo que se desea obtener, y a partir de ella se especifica y da detalle a las partes de mayor prioridad, y que se desean tener cuanto antes. Cada ciclo de desarrollo o iteración (*sprint*) finaliza con la entrega de una parte operativa del producto (incremento). El tiempo asignado a cada sprint puede estar comprendido entre una y seis semanas, aunque lo recomendado es que no excedan a un mes. De manera diaria el equipo debe monitorear la evolución de cada sprint por medio de reuniones diarias de 5 a 15 min en los que de manera conjunta se revisa el trabajo realizado pot cada miembro el día anterior y el previsto para el día en curso, se realiza de pie junto a un tablero o pizarra con información de las tareas del sprint, y el trabajo pendiente en cada una. Esta reunión se denomina, reunion de pie o scrum diario y si se emplea la terminología inglesa: *stand-up meeting*, también: *daily scrum* o *morning rollcall*.

2.9. Framework

Un framework es un esquema o marco de trabajo que ofrece una estructura base para elaborar un proyecto con objetivos específicos, es decír, una especie de plantilla

ó estructura conceptual y tecnológica de asistencia definida que sirve como punto de partida para la organización y desarrollo de un software. Estos entornos permiten agilizar el proceso de desarrollo debido a posibilidad de reutilizar herramientas o módulos tantas veces como se requiera, así como encontrar en los mismos: soporte de programas, bibliotecas, lenguaje interpretado, entre otras. (Edex, 2022).

2.10. Frontend

Es la parte de la aplicación que interactúa con los usuarios, lo que se conoce como el desarrollo del lado del cliente. En este tipo de desarrollo se maqueta la estructura semántica del contenido y se codifica el diseño en hojas de estilo, es decir, se da forma a la parte frontal de la aplicación (fondos, colores, texto, animaciones o efectos) haciedo uso de estos tres elementos: *HTML*, *CSS* y *JavaScript* respectivamente. (Coppola, 2022).

2.10.1. HTML(HyperText Markup Language)

Es un Lenguaje de Marcas de Hipertexto, que definen el significado y la estructura del contenido web.

2.10.2. CSS(Cascading Style Sheets)

Es el lenguaje de estilos utilizado para describir la presentación de documentos *HTML* o *XML* (en-US) (incluyendo varios languages basados en *XML* como *SVG*, *MathML* o *XHTML*). *CSS* describe como debe ser renderizado el elemento estructurado en la pantalla, en papel, en el habla o en otros medios.

2.10.3. JavaScript

Lenguaje interpretado que se ejecuta en el lado del cliente, en el navegador. Con *Javascript* se pueden crear efectos especiales en las páginas y definir interactividades con el usuario.

2.10.4. Boostrap

Es un framework de desarrollo web gratuito y de código abierto. Está diseñado para facilitar el proceso de desarrollo de los sitios web responsivos y orientados a los dispositivos móviles, proporcionando una colección de sintaxis para diseños de plantillas y posse complementos de JavaScrit. Además garantiza que todos los elementos de la interfaz de un sitio web funcionen de forma óptima en todos los tamaños de pantalla. (Boostrap, 2022).

2.10.5. Vue.JS

Esta herramienta permite construir interfaces de usuarios y fue creado por Evan You ex trabajador de Google, quien además fue desarrollador de Angular. Una de sus características más resaltantes es el trabajo con componentes, en los cuales se encapsula código reutilizable. Dentro de un componente vue se encuentran etiquetas *HTML*, estilos de *CSS* y código *JavaScript*. Lo que permite desarrollar proyectos modularizados y fáciles de escalar, debido a que es posible reemplazar un componente por otro de una forma muy sencilla. (García, 2019).

2.11. Backend

El backend de una aplicación consiste en lo que se aloja del lado del servidor, es decír, lo que se encuentra en el interior de las mismas como: marcos, bases de datos y códigos de las distintas funcionalidades que componen el sistema. En este lado de la aplicación se procesa la información para posteriormente enviar al usuario. (Coppola, 2022).

Los lenguajes más utilizados dentro del Backend son: *ASP.NET*, *PHP*, *Ruby*, *Python*, *Node.js*.

2.11.1. PHP(Hypertext Preprocessor)

Es un lenguaje de código abierto adecuado para el desarrollo web que puede ser incrustado en html, generalmente es definido como un lenguaje del lado del servidor y

posee la capacidad de conectar el servidor con la interfaz de usuario. Lo que lo distingue de algo del lado del cliente como Javascript es que el código es ejecutado en el servidor, generando HTML y enviándolo al cliente, éste recibirá el resultado de ejecutar el script, aunque no se sabrá el código subyacente que era.(php group,2022).

Capítulo 3

Desarrollo del sistema (Backend)

Dentro de esta sección se contemplan las herramientas usadas para el desarrollo del lado del servidor (Framework, Sistema Operativo, hardware, lenguaje, modelos y base de datos).

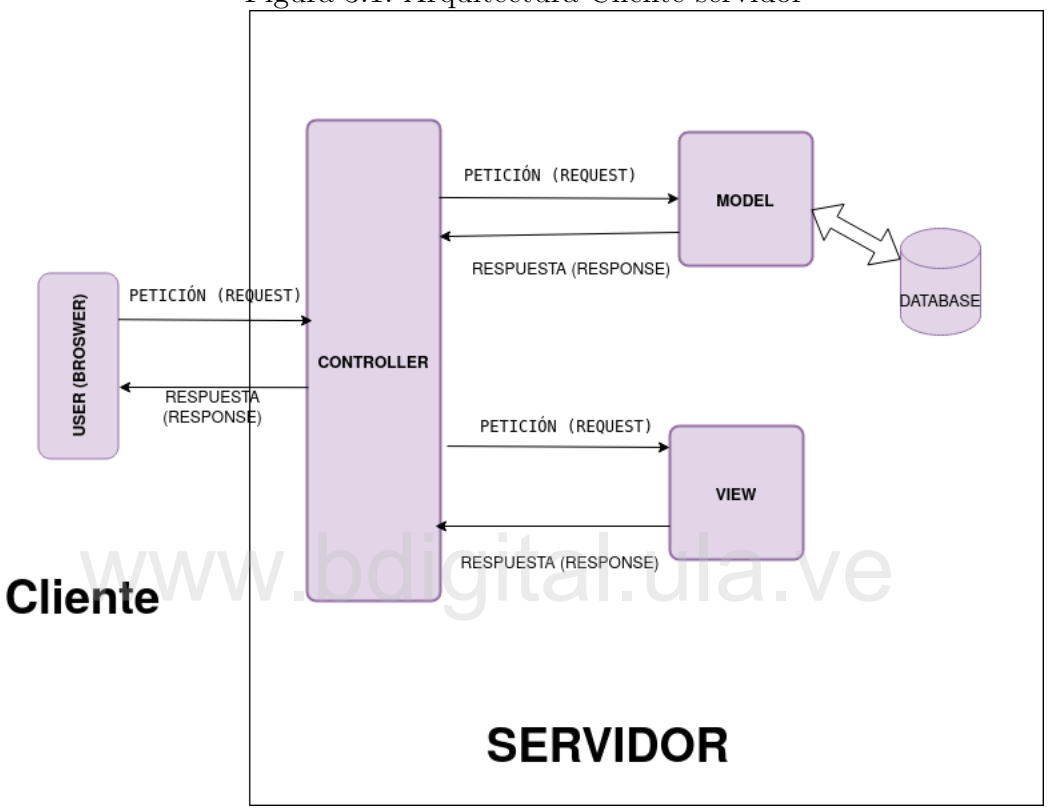
Para el desarrollo de lado del servidor se usó un entorno basado en el sistema operativo linux, distribución Debian 10 (Buster), una arquitectura Cliente Servidor, un patrón de diseño Modelo Vista Controlador, un gestor de base de datos *postgreSQL*, *composer* como gestor de paquetes de PHP para hacer la instalación de laravel, lenguaje PHP y framework laravel.

3.1. Arquitectura del sistema

El sistema consta de una arquitectura cliente servidor, con un patrón de diseño Modelo Vista Controlador, donde el cliente está representado por los diversos navegadores, a través de los cuales el usuario tiene acceso a la aplicación. En cuanto a la lógica del servidor, esta se encuentra distribuida en el modelo, la vista y el controlador, cuyo comportamiento se puede expresar de la siguiente manera: el cliente realiza peticiones al servidor, las cuales son recibidas por el controlador, quien funciona como un intermediario entre el modelo y la vista, dicha petición es enviada al modelo, en caso de que requiera información de la base de datos el *Object Relational Mapping* o Mapeador de Objetos Relacionales (ORM) *eloquent*, consulta la base de datos y extrae

los datos requeridos, retornandolos al modelo, quien a su vez los envía al controlador y este los envía a la vista para que finalmente la vista renderice y lo muestre al cliente (ver figura 3.1).

Figura 3.1: Arquitectura Cliente servidor



Fuente: Elaboración propia.

Contar con un servidor web, aporta a la apliación una serie de venjatas como las siguientes: no necesitan ser instalados en un sistema operativo específico para funcionar, usan menos recursos de hardware que los programas instalados, no requieren canales de distribución, pueden ser usados por múltiples usuarios al mismo tiempo, son menos propensos a colgarse por problemas con el hardware, se puede acceder a ellos desde cualquier dispositivo con internet (Computador, tablets, telefonos móviles, etc).

3.2. Debian GNU/Linux

Debian es un sistema operativo basado en GNU/Linux de código abierto, que ofrece estabilidad en los procesos de actualización de paquetes o la distribución entera sin complicaciones. Es usado ampliamente en el sector de desarrollo de software y hardware porque funciona en numerosas arquitecturas y dispositivos, además ofrece un sistema de seguimiento de incidencias público y otras herramientas para el desarrollo. (Debian, 2022).

En este trabajo se usó especificamente Debian 10 (buster).

3.3. Lenguaje: PHP

El lenguaje seleccionado para el desarrollo de la aplicación del lado del servidor fue PHP, debido a su compatibilidad con la mayoría de los navegadores web y con los principales servidores web, lo que facilita la implementación en diferentes sistemas y plataformas con un coste adicional mínimo. Además, cuenta con una gran comunidad, lo que permite acceder a múltiples tutoriales, preguntas frecuentes y consejos para ayudar a los nuevos desarrolladores, disminuyendo así la complejidad en el proceso de aprendizaje. Para este trabajo se utilizó PHP 8.1.

3.4. Laravel

Es un framework PHP multiplataforma para construir aplicaciones web. Proporciona un conjunto de bibliotecas de código que contienen módulos pre programados que permiten al usuario construir aplicaciones más rápidamente. Brinda características poderosas, como una inyección de dependencia completa, una capa de abstracción de base de datos expresiva, colas y trabajos programados, pruebas unitarias y de integración, y más. Laravel (2022).

Algunas de las características más significativas de laravel son las siguientes:

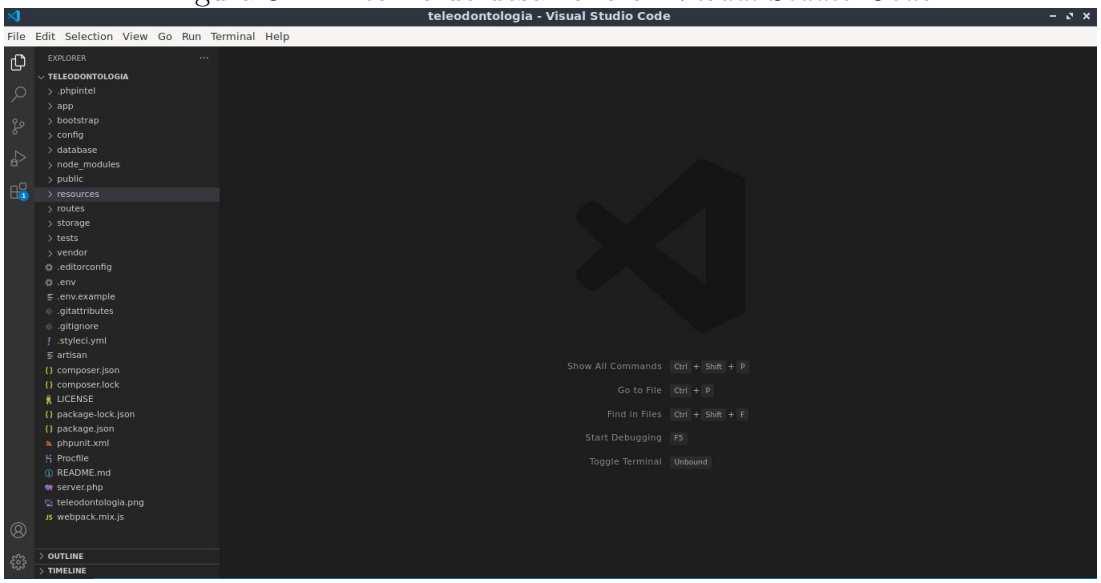
- Motor de plantilla, llamado Blade, da numerosas posibilidades para hacer unas

páginas visualmente muy potentes y eficaces, capaz de utilizar sus propias variables y reutilizarlas.

- Arquitectura conocida como MVC (Modelo-Vista-Controlador).
- Eloquent ORM, es muy intuitivo para escribir consultas en PHP sobre objetos.
- En seguridad, ofrece un nivel bastante fuerte con mecanismos de hash y salt para encriptar por medio de librerías como Bcrypt.
- Artisan, su sistema de comandos otorga al framework gran poder y a los programadores grandes facilidades y posibilidades, para crear controladores, entidades o actualizar la base de datos, así como la ventaja de contar con un servidor integrado, usando el comando php artisar serve.
- Librerías y modularidad. Laravel aparte de sus propias librerías utiliza lo mejor de otros frameworks y aprovecha las características de las últimas versiones de PHP.
- Base de datos y migraciones. Permite actualizar y migrar la base de datos, es decir las migraciones son como el control de versiones para la base de datos.

En este trabajo se escogió la versión 8 del framework cuya distribución de paquetes se observa en la figura 3.2 desde el editor *Visual Studio Code*.

Figura 3.2: Entorno de desarrollo en *Visual Studio Code*



Fuente: Elaboración propia.

Dentro de la carpeta App mostrada en la Figura 3.2, se encuentra otra llamada Http, en la misma se encuentra la carpeta *Controllers* (controladores), la cual aloja los distintos controladores de la aplicación, dentro de esta misma carpeta *App*, se encuentra la carpeta *Models* (Modelos) que contiene los modelos de la aplicación. En cuanto a las vistas, estos archivos se encuentran ingresando a la carpeta *resources*, y posteriormente a las carpertas *components* y *views*. De igual manera, el entorno ofrece las herramientas complementarias para el desarrollo de la aplicación.

3.5. Requrimientos de hardware y software

A continuación, se describen los requerimientos mínimos y recomendados de hardware y software para el correcto funcionamiento del sistema, como se puede observar en la tabla 3.1, la capacidad en disco recomendada es de 320GB, es decír, esta es la capacidad recomendada para un equipo (computador) que se quiera usar como servidor. Sin embargo es necesario resaltar que la aplicación sólo requiere un espacio libre en disco 5GB a 10GB para su funcionamiento. Además, es importante mencionar que para el desarrollo del sistema se usó un equipo con las mismas características de la

sección (Recomendado) de la tabla 3.1.

Tabla 3.1: Tabla de requerimientos de hardware

Requerimientos de hardware		
Hardware	**Mínimo**	**Recomendado**
Procesador	Pentium® Dual-Core CPU T4400 @ 2.20GHz	Intel Core® i5-2400 CPU@ 3.10GHz x 4
RAM	2 GB	8GB
HDD	10GB	320GB
Tipo SO	64bits	64bits

Fuente:Elaboración Propia

Requerimientos de software

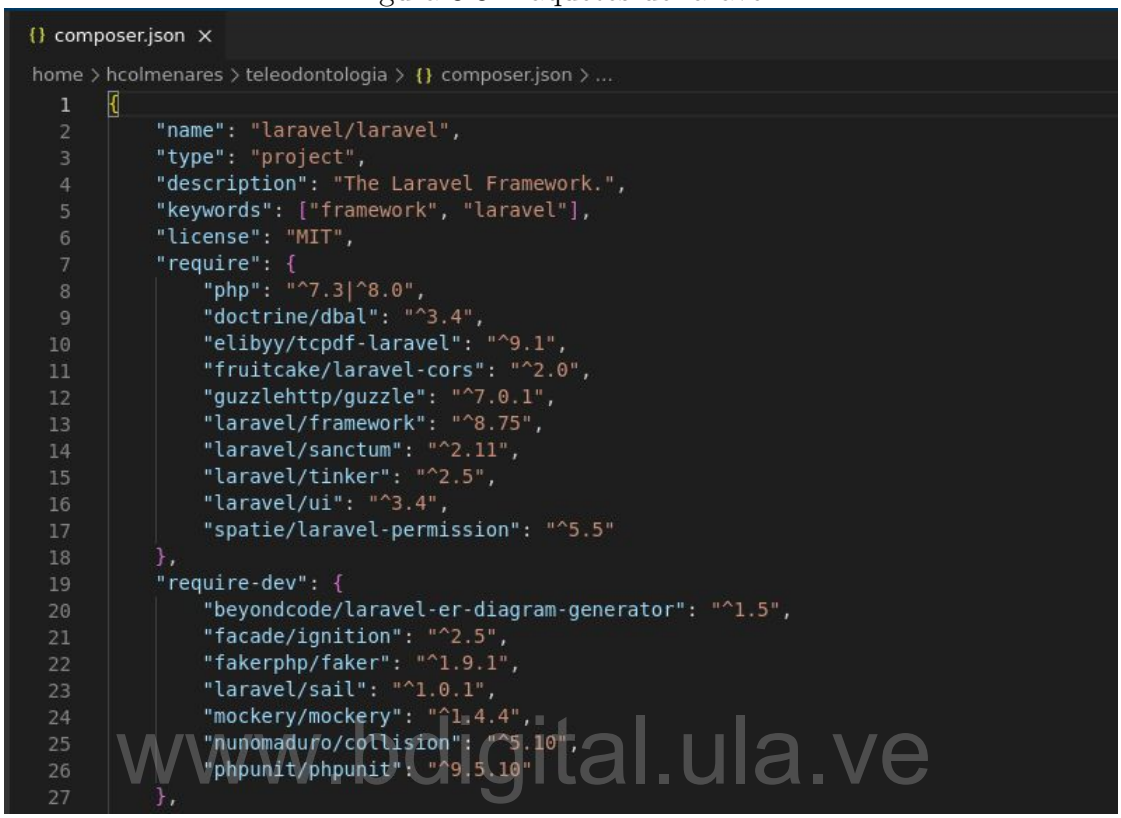
En esta sección se describen los requisitos o paquetes principales para el despliegue y buen funcionamiento del sistema:

A nivel de sistema operativo es necesario instalar los siguientes paquetes:

- Composer version 2.3.4
- node v16.18.0
- npm 8.19.2
- PostgreSQL 12.12
- PHP 8.1.11

Una vez instalados los paquetes anteriores, se procede a ejecutar el comando de *composer*, que se encarga de instalar laravel y los paquetes asociados al mismo, éstos se pueden observar en la Figura 3.3.

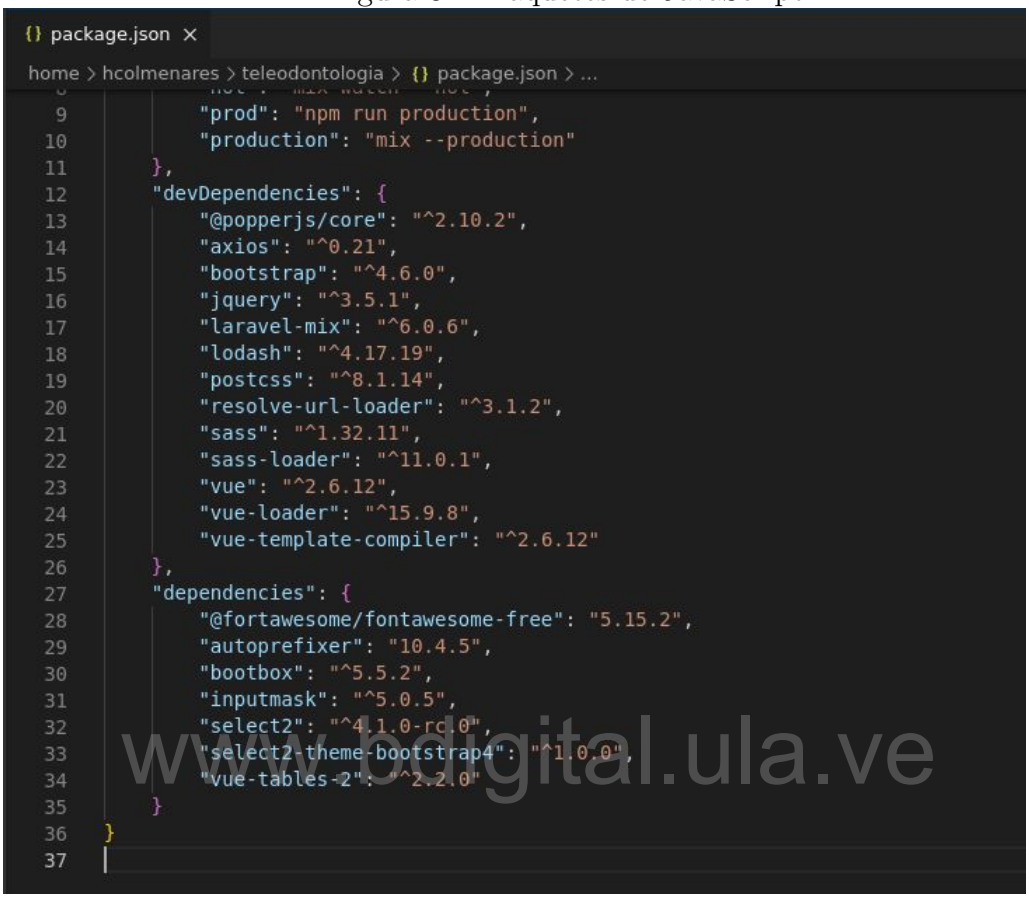
Figura 3.3: Paquetes de laravel



Fuente: Elaboración propia.

En este mismo orden se ejecuta el comando *npm* para instalar los paquetes que se detallan en la figura 3.4, los cuales son necesario para el correcto funcionamiento del *frontend*.

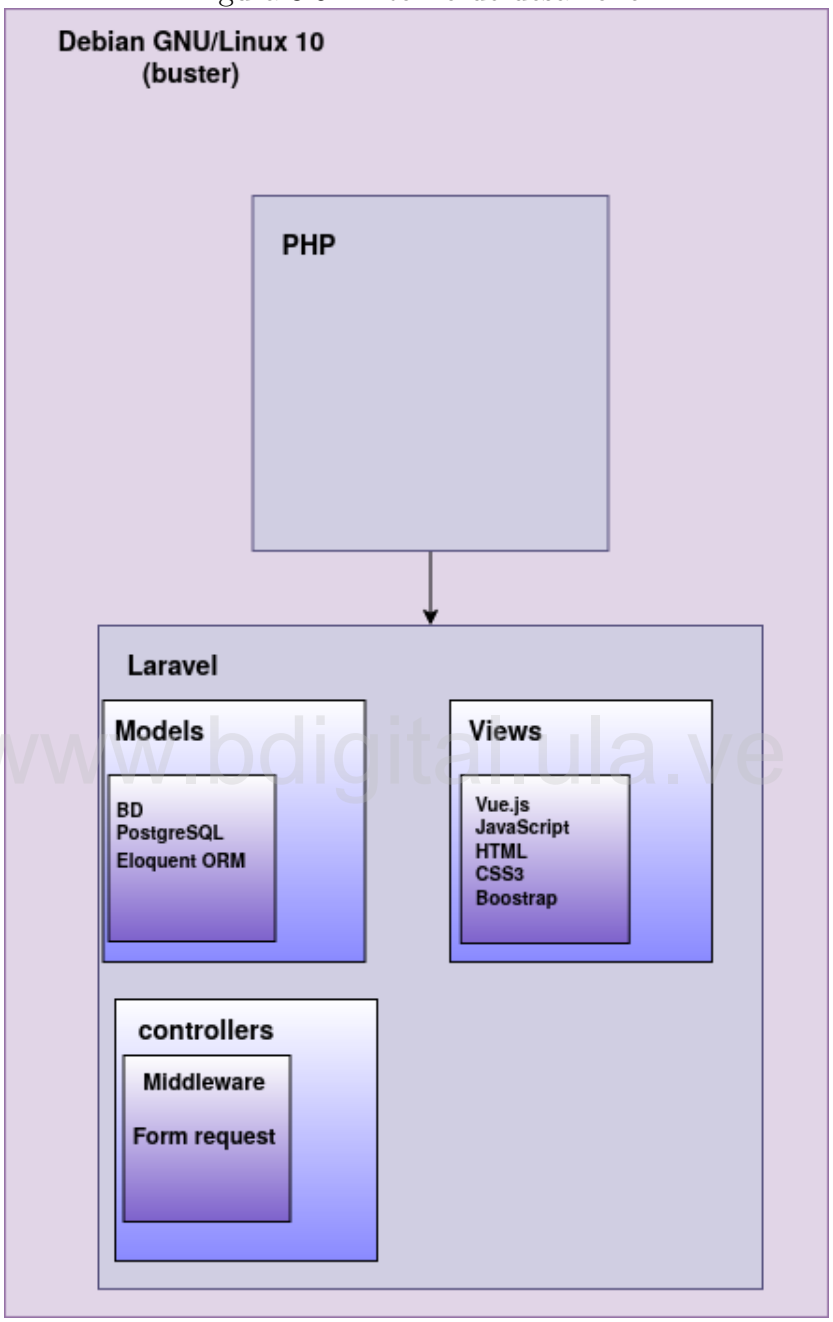
Figura 3.4: Paquetes de JavaScript



Fuente: Elaboración propia.

Una vez ejecutados los comandos mencionados queda preparado el entorno de desarrollo (Ver Figura 3.5) y se puede levantar el servidor a través del comando php artisan serve. Donde artisan es la interfaz de línea de comandos de laravel. Laravel (2022).

Figura 3.5: Entorno de desarrollo



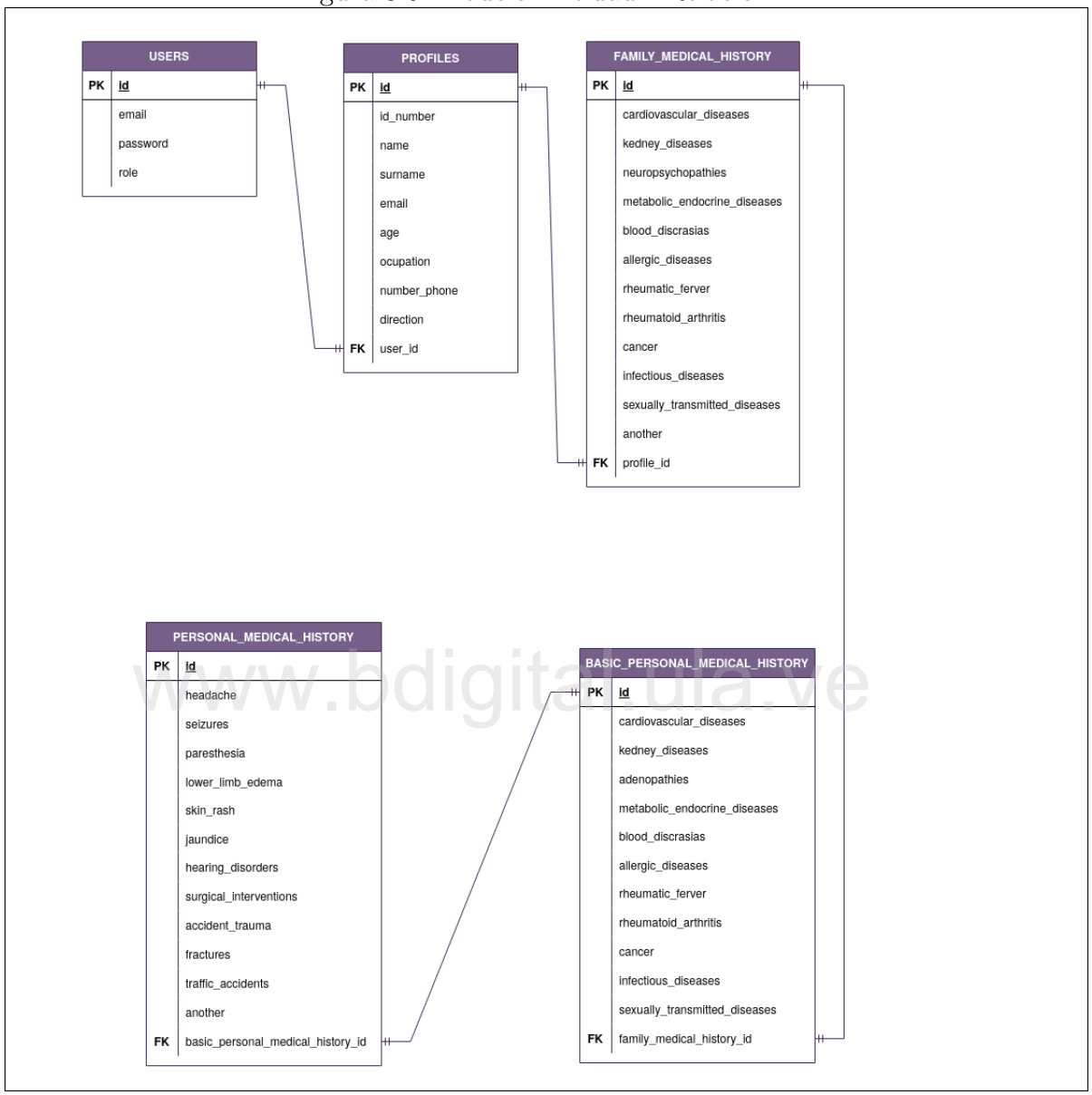
Fuente: Elaboración propia.

3.6. Modelado de la base de datos

El modelo de datos nos permite identificar y representar las entidades y sus atributos así como el tipo de relaciones entre ellas, lo que facilita la definición de las funcionalidades que debe ofrecer el sistema en función de los datos que este procesará. Por otra parte nos permite evaluar el esquema de almacenamiento previo a su implementación, con el fin de garantizar la integridad de los datos, evitar la duplicidad de la información y establecer las conexiones correctas entre las tablas.

En este sentido y como punto de partida del diseño de la aplicaión se definieron las entidades: USERS, PROFILES, FAMILY_MEDICAL_HISTORY ,BASIC_PERSONAL_MEDICAL_HISTORY, PERSONAL_MEDICAL_HISTORY), las que poseen una relación uno a uno entre ellas y representan el esquema lógico de la aplicación. Cada una de éstas entidades posee atributos que se pueden observar en la figura 3.6 y que serán descritos en la sección de la construcción de la base de datos.

Figura 3.6: Modelo Entidad - Relación.



Fuente: Elaboración propia.

3.7. Construcción de la base de datos

En esta sección se describen los campos de cada una de las entidades, a fin de construir la base de datos en términos de la estructura de datos que puede procesar un sistema gestor de base de datos, como lo es en este caso el sistema de gestión de base

de datos relacional, *PostgreSQL*. En este sentido, para la descripción de cada uno de los atributos se tomaron en cuenta los siguientes campos:

- Nombre de la columna: En esta campo se describen los nombre que tendran las columnas de la tabla de usuarios, los que a su vez corresponde a los nombres de los atributos de la entidad.
- Tipo de dato: En este campo se describe el tipo de dato del atributo.
- ¿Es nulo?: En este campo se define con un si o un no si el campo es nulo o no.
- ¿Incremental?: En este campo se define si la variable o atributo es de tipo incremental o no.
- ¿Llave primaria? : En este campo se define si el atributo es el identificador o llave primaria de la tabla.
- ¿Llave Foránea? : En este campo se define si el atributo es un identificador de la tabla con la cual guarda relación.
- Nombre del formulario: En este campo se describe el nombre que tiene el atributo en el formulario.
- Tabla de referencia : En caso que exista una llave foránea en la tabla, en este campo se escribe el nombre nombre de la tabla con la que guarda relación.
- Tipo de relación: En este campo se describe el tipo de relación (1:1, 1:N; N:N, etc), que tienen las tablas, en caso de que el atributo sea una llave foránea.

A continuación las tablas que muestran de manera detallada los atributos de cada una de las entidades.

Tabla USERS:

En la tabla 3.2 se definen los atributos de los usuarios para el inicio de sesión.

Tabla 3.2: Tabla de Usuarios

USERS								
Nombre de la columna	**Tipo de dato**	**¿es nulo?**	**¿Incrementable?**	**¿Llave primaria?**	**¿Llave foránea?**	**Nombre en formulario**	**Tabla de referencia**	**Relación**
id	Integer	Si	No	Si	No	No aplica	No aplica	No aplica
email	String	Si	No	No	No	Correo	No aplica	No aplica
Pasword	String	Si	No	No	No	Contraseña	No aplica	No aplica

Fuente:Elaboración Propia

Tabla PROFILES:

En esta tabla se describen los atributos del el formulario de registro de los usuarios. En la 3.3 se puede observar que posee una relación 1:1 con la tabla o entidad 3.2.

Tabla 3.3: Tabla de perfiles

PROFILES								
Nombre de la columna	**Tipo de dato**	**¿es nulo?**	**¿Incrementable?**	**¿Llave primaria?**	**¿Llave foránea?**	**Nombre en formulario**	**Tabla de referencia**	**Relación**
id	Integer	Si	No	Si	No	No aplica	No aplica	No aplica
id_number	String	Si	No	No	No	Cédula	No aplica	No aplica
firts_name	String	Si	No	No	No	Nombre	No aplica	No aplica
last_name	String	Si	No	No	No	Apellido	No aplica	No aplica
email	String	Si	No	No	No	Correo	No aplica	No aplica
phone	String	Si	No	No	No	Teléfono	No aplica	No aplica
family_phone	String	Si	No	No	No	Teléfono Familiar	No aplica	No aplica
adress	String	Si	No	No	No	Dirección	No aplica	No aplica
ocupation	String	Si	No	No	No	Ocupación	No aplica	No aplica
age	integer	Si	No	No	No	Edad	No aplica	No aplica
gender	String	Si	No	No	No	Género	No aplica	No aplica
user_id	integer	Si	No	No	Si	No aplica	USERS	1A1

Fuente:Elaboración Propia

Tabla FAMILY_MEDICAL_HISTORY:

En esta tabla se describen los atributos de los antecedentes médicos familiares del

usuario aciente. Como se puede observar en la tabla 3.4 esta guarda una relación 1:1 con la tabla 3.3.

Tabla 3.4: Tabla de antecedentes médicos familiares

FAMILY_MEDICAL_HISTORY								
Nombre de la columna	**Tipo de dato**	**¿es nulo?**	**¿Increm-entable?**	**¿Llave primaria?**	**¿Llave foránea?**	**Nombre en formulario**	**Tabla de referencia**	**Relación**
id	Integer	Si	No	Si	No	No aplica	No aplica	No aplica
cardiovascular_diseases	String	Si	No	No	No	Enfermedades cardiovasculares	No aplica	No aplica
kidney_diseases	String	Si	No	No	No	Enfermedades renales	No aplica	No aplica
neuropsychopathies	String	Si	No	No	No	Neuropsicopatias	No aplica	No aplica
metabolic_endocrine_diseases	String	Si	No	No	No	Enfermedades metabólicas y endocrinas	No aplica	No aplica
blood_discrasias	String	Si	No	No	No	Discrasias sanguíneas	No aplica	No aplica
allergic_diseases	String	Si	No	No	No	Enfermedades alérgicas	No aplica	No aplica
rheumatic_ferver	String	Si	No	No	No	Fiebre reumática	No aplica	No aplica
rheumatoid_arthritis	String	Si	No	No	No	Artritis rehumantoidea	No aplica	No aplica
cancer	String	Si	No	No	No	Cancer	No aplica	No aplica
infectious_diseases	String	Si	No	No	No	Enfermedades infecciosas	No aplica	No aplica
sexually_transmitted_diseases	String	Si	No	No	No	Enfermedades de trasmisión sexual	No aplica	No aplica
another	String	Si	No	No	No	Otras	No aplica	No aplica
profile_id	integer	Si	No	No	Si	No aplica	PROFILES	1A1

Fuente:Elaboración Propia

Tabla BASIC_PERSONAL_MEDICAL_HISTORY:

En esta tabla se describen los atributos de los antecendentes médicos personales del usuario. Debido a la gran cantidad de los mismos, estos se dividieron en 2 tablas, para facilitar la lectura de los mismos. Y como se puede observar en la tabla 3.5 esta guarda una relación 1:1 con la tabla 3.4.

Tabla 3.5: Tabla de antecedentes médicos personales (parte 1)

BASIC_PERSONAL_MEDICAL_HISTORY								
Nombre de la columna	**Tipo de dato**	**¿es nulo?**	**¿Increm-entable?**	**¿Llave primaria?**	**¿Llave foránea?**	**Nombre en formulario**	**Tabla de referencia**	**Relación**
id	Integer	Si	No	Si	No	No aplica	No aplica	No aplica
cardiovascular_diseases	String	Si	No	No	No	Enfermedades cardiovasculares	No aplica	No aplica
kidney_diseases	String	Si	No	No	No	Enfermedades renales	No aplica	No aplica
adenopathies	String	Si	No	No	No	Adenopatias	No aplica	No aplica
metabolic_endocrine_diseases	String	Si	No	No	No	Enfermedades metabólicas y endocrinas	No aplica	No aplica
blood_discrasias	String	Si	No	No	No	Discrasias sanguíneas	No aplica	No aplica
allergic_diseases	String	Si	No	No	No	Enfermedades alérgicas	No aplica	No aplica
rheumatic_ferver	String	Si	No	No	No	Fiebre reumática	No aplica	No aplica
rheumatoid_arthritis	String	Si	No	No	No	Artritis rehumantoidea	No aplica	No aplica
cancer	String	Si	No	No	No	Cancer	No aplica	No aplica
infectious_diseases	String	Si	No	No	No	Enfermedades infecciosas	No aplica	No aplica
sexually_transmitted_diseases	String	Si	No	No	No	Enfermedades de trasmisión sexual	No aplica	No aplica
another	String	Si	No	No	No	Otras	No aplica	No aplica
family_medical_history_id	integer	Si	No	No	Si	No aplica	FAMILY_MEDICAL_HISTORY	1A1

Fuente:Elaboración Propia

Tabla PERSONAL_MEDICAL_HISTORY

Esta tabla contine la segunda parte de los atributos correspondientes a los antecedentes personales del usuario paciente. Y como se puede observar en la tabla 3.6 esta guarda una relación 1:1 con la tabla 3.5.

Tabla 3.6: Tabla de antecedentes médicos personales (parte 2)

PERSONAL_MEDICAL_HISTORY								
Nombre de la columna	**Tipo de dato**	**¿es nulo?**	**¿Increm-entable?**	**¿Llave primaria?**	**¿Llave foránea?**	**Nombre en formulario**	**Tabla de referencia**	**Relación**
id	Integer	Si	No	Si	No	No aplica	No aplica	No aplica
headache	String	Si	No	No	No	Dolor de cabeza	No aplica	No aplica
seizures	String	Si	No	No	No	Convulsiones	No aplica	No aplica
paresthesia	String	Si	No	No	No	Parestesias	No aplica	No aplica
lower_limb_edema	String	Si	No	No	No	edema de miembros inferiores	No aplica	No aplica
Skin_rash	String	Si	No	No	No	Erupción cutanea	No aplica	No aplica
jaundice	String	Si	No	No	No	Ictericia	No aplica	No aplica
hearing_disorders	String	Si	No	No	No	Trastornos auditivos	No aplica	No aplica
surgical_interventions	String	Si	No	No	No	Intervenciones quirúrgicas	No aplica	No aplica
accident trauma	String	Si	No	No	No	Traumas por accidentes	No aplica	No aplica
fractures	String	Si	No	No	No	Fracturas	No aplica	No aplica
traffic accidents	String	Si	No	No	No	Accidentes de tránsito	No aplica	No aplica
dentist	String	Si	No	No	No	Última visita al odontólogo	No aplica	No aplica
teeth_extracted	String	Si	No	No	No	¿Le ha extraído dientes?	No aplica	No aplica
dentures	String	Si	No	No	No	¿Tiene Prótesis?	No aplica	No aplica
dentures_state	String	Si	No	No	No	Estado de la prótesis	No aplica	No aplica
reason	String	Si	No	No	No	Motivo de la Consulta	No aplica	No aplica
basic_personal_ medical_history_id	integer	Si	No	No	Si	No aplica	BASIC_PERSONAL_ MEDICAL_HISTORY	1A1

Fuente:Elaboración Propia

Capítulo 4

Desarrollo del sistema (Frontend)

4.1. Especificación de requerimientos

En esta sección se realiza una descrpición general de los módulos del sistema, los roles dentro del mismo y la especificación de la interacción del usuario con el sistema a través de casos de uso, diagramas de flujo y finalmente capturas de la aplicación en ejecución de las funcionalidades descritas.

Previo a la especificación de requerimientos y la defínción de los modelos, se indagó acerca del comportamiento actual de la clínica, cuyo proceso puede definirse de la siguiente manera: La cátedra o clínica está compuesta por varios profesores, donde uno de ellos funge como Jefe de cátedra, quien es el responsable de canalizar todo lo referente a la misma en temas administrativos, así como, de coordinar las secciones con los diferentes grupos de estudiantes y profesores que las conformarán. En cuanto a los pacientes, éstos acuden a la clínica para solicitar atención de tratamientos de tipo protésicos, o son citados por los estudiantes con este mismo fin. En este sentido, los profesores son los encargados de diagnósticar y aprobar o rechazar el caso, si este no es apto para el estudiante por razones de complejidad, riesgos o no aplicar para la práctica específica de la tutoría. En el caso de que el paciente sea aprobado por el profesor para la práctica clínica, el estudiante debe rehabilitar totalmente la condición odontológica del paciente durante el año académico. Por tanto, el estudiante debe asignar citas

periódicas al paciente e ir llevado un registro histórico de los tratamientos aplicados al paciente hasta culminar la rehabilitación y finalmente darle de alta. Es importante señalar, que la clínica también ofrece otros tipos tramientos odontolólgicos, es por ello que recibe el nombre de Clinica Integral.

4.1.1. Descripción general de los Módulos del sistema

Con el propósito de distribuir de manera armónica las acciones de los distintos actores, el sistema se estructuró en cuatro (04) módulos que representan cada uno de los roles de los usuarios que interactúan con el mismo. En la Figura 4.1, en el panel lateral izquierdo se pueden detallar los mencionados módulos del sistema.

Figura 4.1: Módulos del sistema

Fuente: Elaboración propia.

A continuación una descripción general de los módulos del sistema, es decír las partes o porciones que componen la aplicación (Ver Figura 4.1).

Tabla 4.1: Módulos del sistema

Módulos	Descripción
Profesores	Este módulo contiene el CRUD de los profesores de la cátedra. De modo que sólo el jefe de cátedra puede Registrar, Detallar, Editar y Eliminar Profesores.
Estudiantes	Este módulo contiene el CRUD de los Estudiantes. Donde cada profesor puede Registrar, Detallar, Editar y Eliminar estudiantes de su tutoría.
Pacientes	En este módulo se visualizan las tablas de los pacientes regitrados con sus distintos estados (En espera, Reservado, En atención, No apto). De modo que el estudiante puede reservar, el profesor puede aprobar la reserva y asignar el pacientes al estudiante o rechazar y declarar no apto para la atención o tratamiento específico. Además en caso de serle aprobado el paciente, el estudiante puede, asignarle citas al paciente, realizarle registro de historial, y darle de alta cuando culmine el proceso.
Reportes	En este módulo se generan los reportes asociados a los distintos usuarios del sistema, a los mismos se accede de acuerdo a los permisos otorgados por roles.

Fuente:Elaboración propia.

la programación modular permite disminuír la complejidad de un problema al subdividirlo en varias partes, de esta manera es mucho mas sencillo ir abordando subproblemas hasta completar la solución global u objetivos del proyecto.

4.1.2. Roles y permisos dentro del sistema

El manejo de roles nos permite poder controlar los permisos que tendrán los usuarios dentro del sistema, es decír otorgar o bloquear el acceso a ciertas funcionalidades

del mismo. En este sentido, se crearon los roles y permisos mostrados en la Tabla 4.2.

Tabla 4.2: Roles y Permisos del sistema

Roles	Permisos
Jefe de cátedra	El usuario puede Registrar, Editar, Detallar, y Eliminar profesores dentro del sistema, además, puede detallar la información de todos los demás usuarios registrados (Estudiantes, pacientes), así como, aprobar o rechazar pacientes a estudiantes y generar reportes de todos los usuarios.
Profesor	El usuario puede Registrar, Editar, Detallar, y Eliminar Estudiantes que tutorea, así como, aprobar o rechazar pacientes a estudiantes, visualizar la lista de pacientes registrados, los pacientes reservados por sus estudiantes, los que estan en proceso de atención, y los calificados como no aptos. Además, puede generar reportes de estudiantes y pacientes atendidos por ellos.
Estudiante	El usuario puede visualizar la lista de pacientes registrados, detallar registro de paciente, reservar pacientes, enviar cita a pacientes, agregar historial de paciente, dar de alta al paciente. Así como, generar reportes de sus pacientes.
Paciente	El usuario puede crear usuario único dentro del sistema, puede realizar pretriaje y cargar dichos datos para solicitar atención.

Fuente:Elaboración propia.

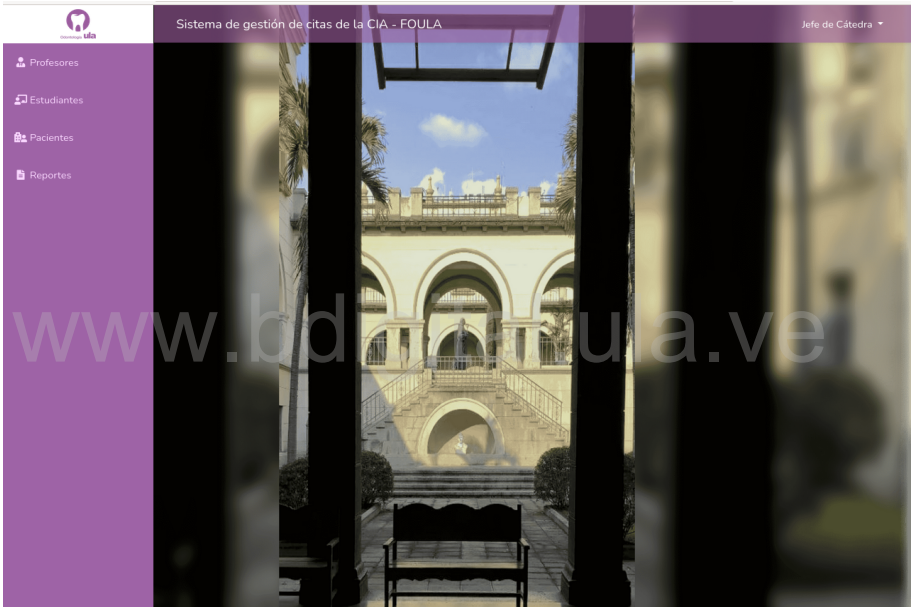
4.1.3. Flujos principales del sistema

A continuación se describen los flujos principales del sistema, realizados por cada actor o usuario(Rol) de la aplicación.

Rol: Jefe de cátedra

Como se muestra en la Figura 4.2, el usuario Jefe de cátedra tiene acceso a los 4 módulos del sistema, mismos que se pueden visualizar en el panel lateral izquierdo.

Figura 4.2: Home de Jefe de Cátedra

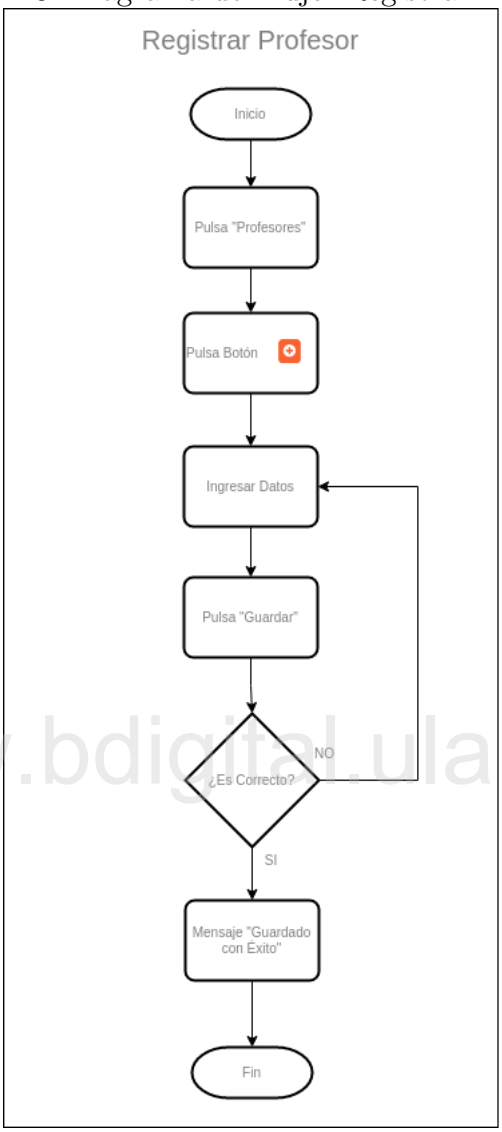


Fuente: Elaboración propia.

Registro de Profesores

El usuario Jefe de cátedra puede realizar operaciones CRUD (Create, Read, Update, Delete) ó en español (Crear ó registrar, Leer ó detallar, Actualizar ó editar y Borrar) sobre Profesores (Ver Figura 4.3, Figura 4.4). Una vez ha culminado el registro del profesor, el sistema envía de manera automática un correo electrónico al nuevo usuario con sus credenciales de acceso y direccion web de la aplicación.(Ver Figura 4.5

Figura 4.3: Diagrama de Flujo: Registrar Profesor



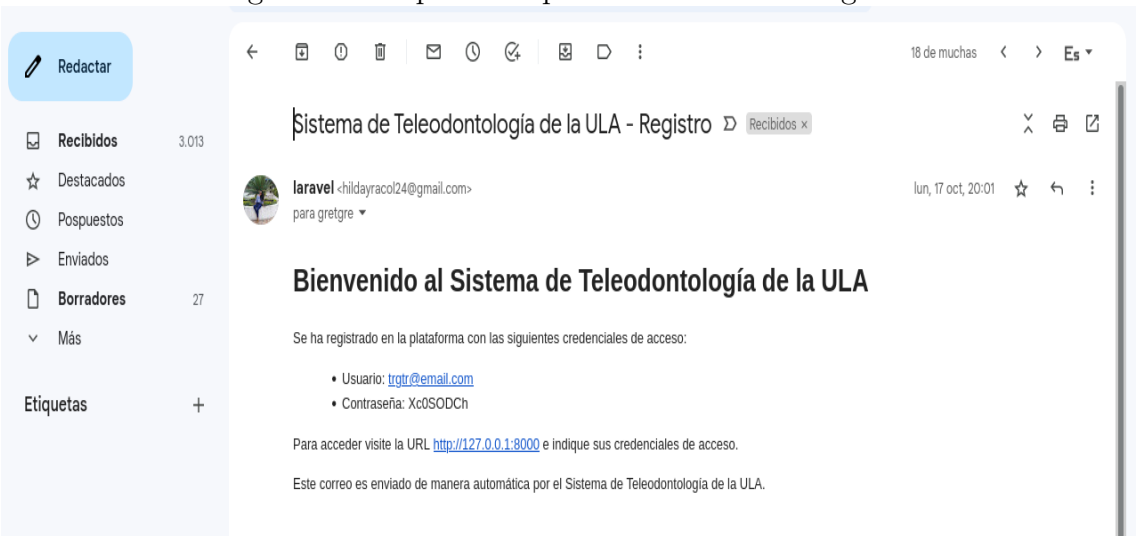
Fuente: Elaboración propia.

Figura 4.4: Caso de uso: Registrar Profesor

Caso de uso "Registrar Profesor"		
Actor (s)	Jefe de Cátedra (Admin)	
Condiciones iniciales	Información del Profesor	
Condiciones de salida	Profesor registrado en el sistema	
Flujo Principal	**Usuario**	**Sistema**
	1. El Admin pulsa el botón "Registrar Profesor".	El sistema muestra un formulario con los siguientes campos: 1- Nombres. 2- Apellidos. 3- Email 4- Cédula 5- Teléfono 6-Telefono familiar 7- Edad 8- Género 9- Dirección 10- Ocupación. El sistema muestra el botón "Guardar" y "Cancelar".
	2.El Admin ingresa los datos solicitados y pulsa "Guardar"	El sistema envía un correo electrónico al profesor con las credenciales de acceso a la aplicación (Usuario y Clave) y muestra el siguiente mensaje por pantalla: "Profesor registrado con éxito".
Flujo Alternativo	**Usuario**	**Sistema**
	1. Si el Admin introduce datos de manera incorrecta u omite datos.	El sistema muestra por pantalla el siguiente mensaje "Por favor introduzca correctamente los datos solicitados".
	2. Si el Admin pulsa el botón "Cancelar".	El sistema limpia los campos llenos y retorna a la página principal del sistema.
Requisitos Especiales	1. El correo electrónico debe ser válido. 2. La contraseña debe contener una longitud de 8 a 16 caracteres, al menos una letra mayúscula y minúscula, dos números y dos caracteres especiales. 3. El campo cédula debe ser numérico. 4. Número de teléfono válido.	

Fuente: Elaboración propia.

Figura 4.5: Captura de pantalla: Correo de registro



Fuente: Elaboración propia.

Figura 4.6: Caso de uso: Editar Profesor

Caso de Uso "Editar Profesor"		
Actor (s)	Jefe de Cátedra (Admin)	
Condiciones iniciales	Profesor registrado del sistema	
Condiciones de salida	Modificación de Profesor registrada en el sistema	
Flujo Principal	**Usuario**	**Sistema**
	1. El Admin realiza una búsqueda en la tabla de registros de profesores, una vez seleccionado el registro, pulsa el botón "Modificar".	El sistema muestra un formulario con los campos registrados en forma de edición, junto a las opciones "Guardar" y "Cancelar".
	2. El Admin modifica los campos deseados y pulsa "Guardar".	El sistema registra la modificación realizada y muestra un mensaje por pantalla "Modificación exitosa" y muestra la tabla de profesores registrados.
Flujo Alternativo	**Usuario**	**Sistema**
	1. Si el Admin introduce un datos de manera incorrecta.	El sistema muestra por pantalla el siguiente mensaje "Por favor introduzca correctamente los datos solicitados".
	2. Si el Admin omite llenar un campo solicitado.	El sistema muestra por pantalla el siguiente mensaje "Por favor introduzca la información solicitada".
	3. Si el Admin pulsa el botón "Cancelar".	El sistema limpia los campos llenos y retorna a la página principal del sistema.
Requisitos Especiales	1. El correo electrónico debe ser válido. 2. La contraseña debe contener una longitud de 8 a 16 caracteres, al menos una letra mayúscula y minúscula, dos números y dos caracteres especiales. 3. El campo cédula debe ser numérico. 4. Número de teléfono válido (debe contener números).	

Fuente: Elaboración propia.

Figura 4.7: Caso de uso: Detallar Profesor

Caso de Uso "Detallar Profesor".		
Actor (s)	Jefe de Cátedra (Admin)	
Condiciones iniciales	Profesor registrado del sistema	
Condiciones de salida	Detallar información de profesor por pantalla.	
Flujo Principal	**Usuario**	**Sistema**
	1. El Admin pulsa la opción "Profesores".	1. El sistema muestra la tabla de registros de profesores.
	2. El Admin realiza una búsqueda en la tabla de registros de profesores, una vez seleccionado el registro, pulsa el botón "Detallar".	2. El sistema muestra la información de profesor por pantalla en campos no editables.

Fuente: Elaboración propia.

Figura 4.8: Caso de uso: Eliminar Profesor

Caso de Uso "Eliminar Profesor".		
Actor (s)	Jefe de Cátedra (Admin)	
Condiciones iniciales	Profesor registrado del sistema	
Condiciones de salida	Registro de profesor eliminado del sistema	
Flujo Principal	**Usuario**	**Sistema**
	1. El Admin realiza una búsqueda en la tabla de registros de profesores, una vez seleccionado el registro, pulsa el botón "Eliminar".	El sistema muestra un mensaje por pantalla "¿Está seguro de querer eliminar este registro?" Junto a las opciones "Confirmar" y "Cancelar".
	2. El Admin pulsa "Confirmar".	El sistema elimina el registro y muestra un mensaje por pantalla "Profesor Eliminado".
Flujo Alternativo	**Usuario**	**Sistema**
	1. Si el Admin pulsa el botón "Cancelar".	El sistema no ejecuta ninguna acción.

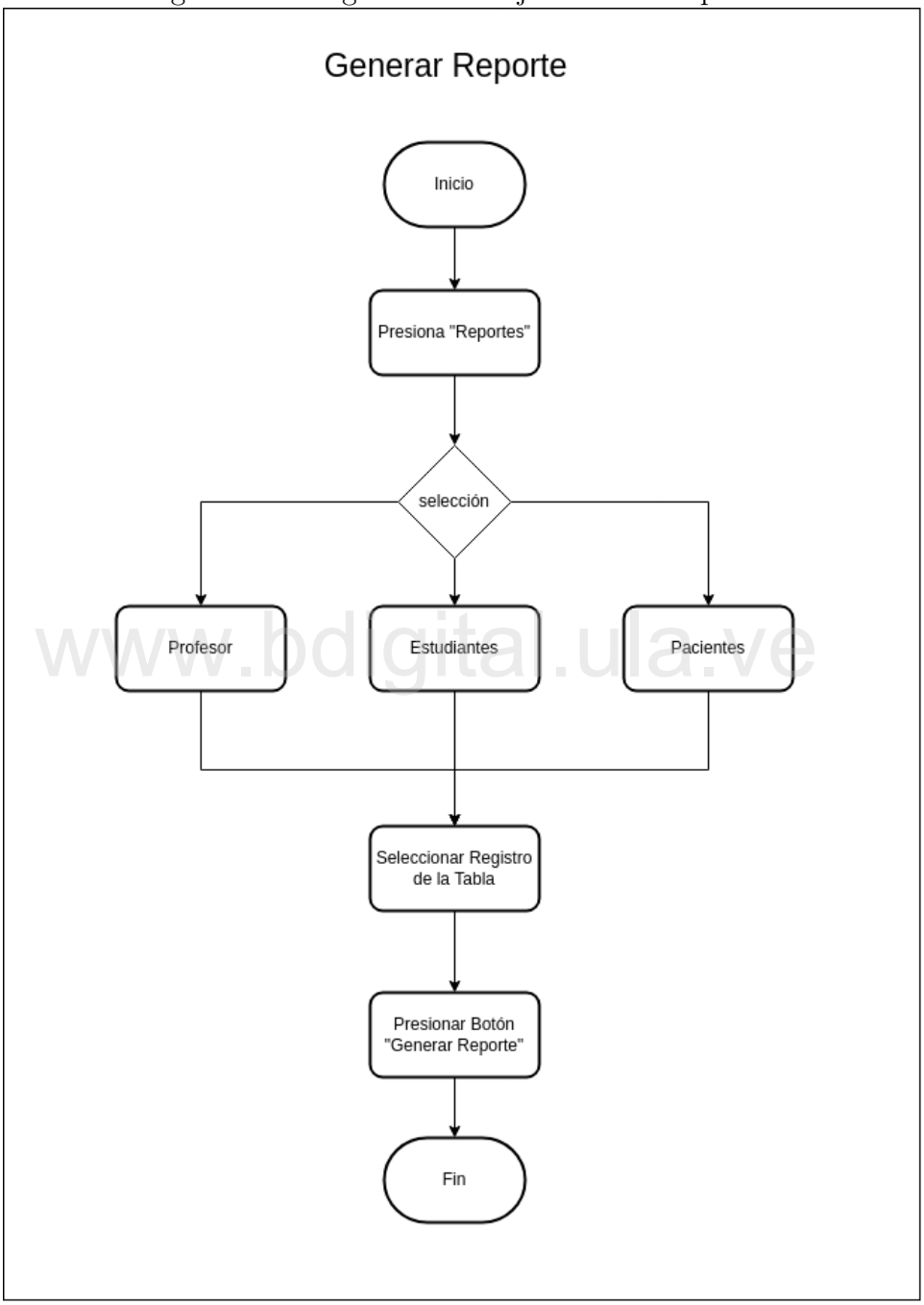
Fuente: Elaboración propia.

Reportes

El sistema permite la generación de reportes de acuerdo los niveles de usuarios registrados en el sistema, es decír, el usuario jefe de cátedra podrá generar reportes asociados

a Profesores, Estudiantes y Pacientes (Ver Figura 4.10, la Figura 4.9) y 4.11

Figura 4.9: Diagrama de Flujo: Generar reportes



Fuente: Elaboración propia.

Figura 4.10: Caso de uso: Generar reportes

Caso de uso Generar Reportes		
Actor (s)	Jefe de cátedra (Admin).	
Condiciones iniciales	Registro en el sistema.	
Condiciones de salida	Impresión de reporte por pantalla.	
Flujo Principal	**Usuario**	**Sistema**
	1. El Admin pulsa **Reportes.** 2- El Admin selecciona Profesores, Estudiantes o pacientes, 3- El Admin selecciona registro y pulsa la opción **generar reporte**.	1. El sistema despliega las opciones : Profesores, Estudiantes y pacientes. 2. El sistema muestra la tabla de registros de la opción seleccionada. 3. El sistema muestra el PDF del registro seleccionado por pantalla.

Fuente: Elaboración propia.

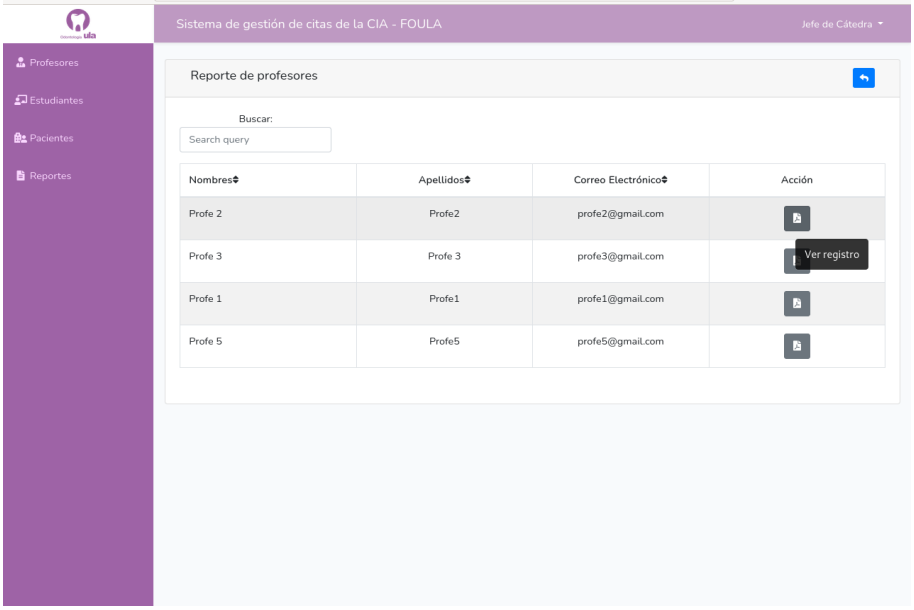
Figura 4.11: Captura de pantalla: Generar reportes

Fuente: Elaboración propia.

En la Figura 4.12 se puede observar un ejemplo de como se genera el reporte de

profesores.

Figura 4.12: Ejemplo: Generar reporte de profesor

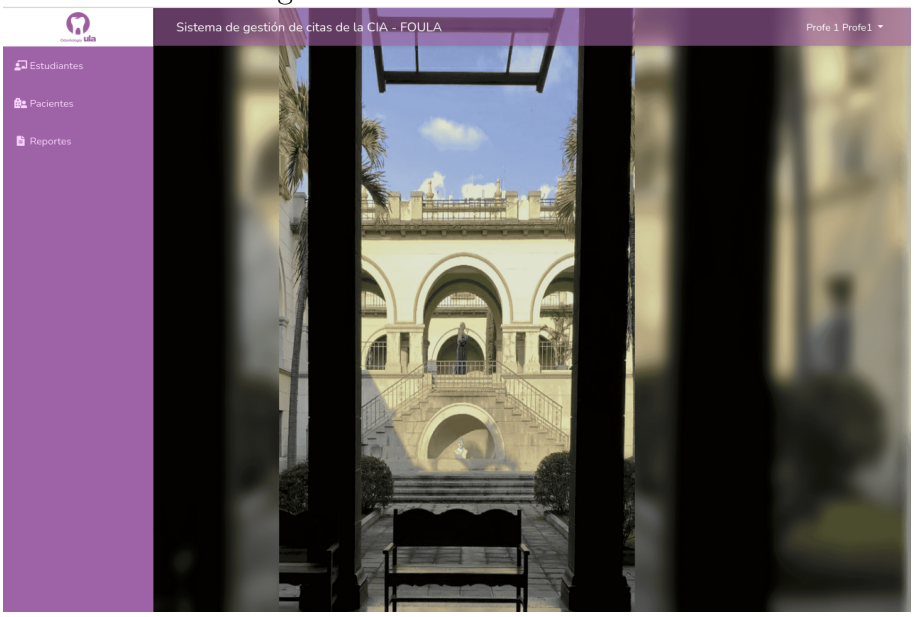


Fuente: Elaboración propia.

Rol: Profesor

En la Figura 4.13 se puede detallar el home del usuario profesor, el cual tiene acceso a los módulos de Estudiantes, Pacientes y Reportes.

Figura 4.13: Home de Profesor

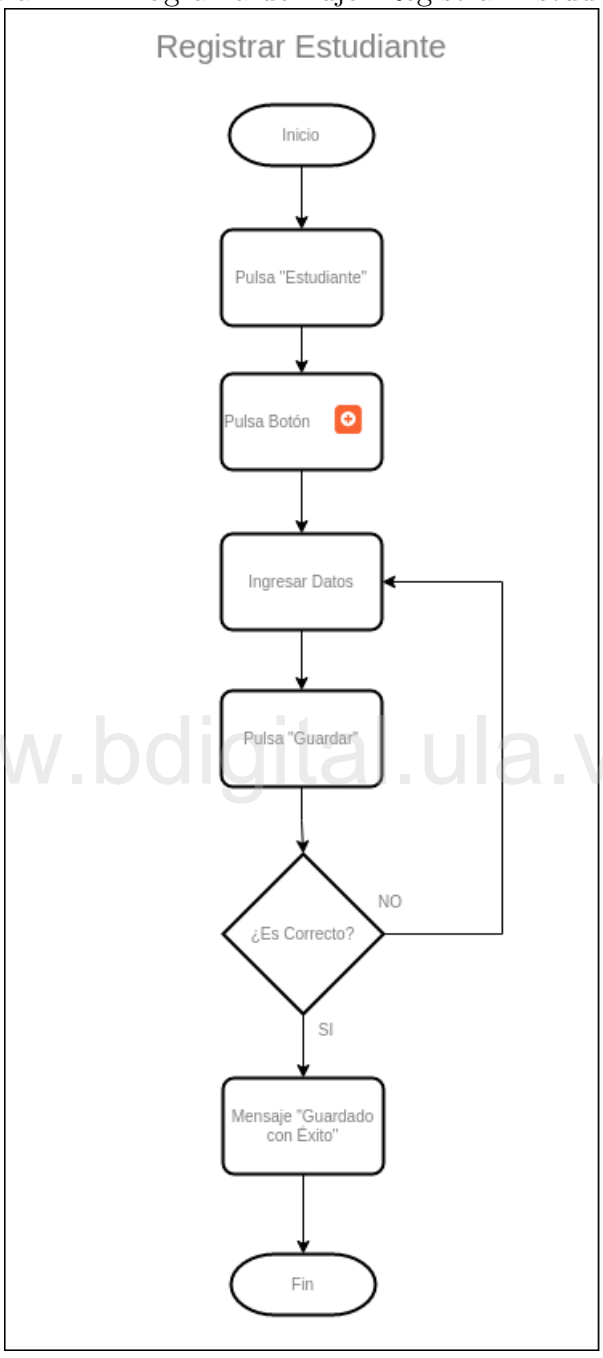


Fuente: Elaboración propia.

Registrar Estudiante

El usuario profesor puede realizar operaciones CRUD sobre estudiantes, (Ver Figura 4.14, 4.15, Figura 4.16, 4.17 y 4.18).Una vez culminado el registro del estudiante el sistema envía un correo electrónico con las credenciales de acceso y dirección web de la aplicación como se mostró en la Figura 4.5

Figura 4.14: Diagrama de flujo: Registrar Estudiante



Fuente: Elaboración propia.

Figura 4.15: Caso de uso: Registrar Estudiante

Caso de uso "Registrar Estudiante"		
Actor (s)	Profesor	
Condiciones iniciales	Información del estudiante	
Condiciones de salida	Estudiante registrado en el sistema	
Flujo Principal	**Usuario**	**Sistema**
	1. El Profesor pulsa el botón "Estudiantes". 2. El profesor pulsa el botón (Crear nuevo registro).	1. El sistema muestra una tabla con los registros de pacientes junto a los botones (Ir atrás y Crear nuevo registro). 2.El sistema muestra un formulario con los siguientes campos: 1- Nombres. 2- Apellidos. 3- Email 4- Cédula 5- Teléfono 6-Telefono familiar 7- Edad 8- Género 9- Dirección 10- Ocupación. Junto a los botones "Guardar", "Cancelar" y "Borrar"
	3. El profesor ingresa los datos solicitados y pulsa "Guardar"	3. El sistema envía un correo electrónico al Estudiante con las credenciales de acceso a la aplicación (Usuario y Clave) y muestra el siguiente mensaje por pantalla: "Estudiante registrado con éxito".
Flujo Alternativo	**Usuario**	**Sistema**
	1. Si el profesor introduce datos de manera incorrecta u omite datos obligatorios.	1. El sistema muestra por pantalla el siguiente mensaje "Por favor introduzca correctamente los datos solicitados".
	2. Si el profesor pulsa el botón "Cancelar".	El sistema limpia los campos llenos y retorna a la página principal del sistema.

Fuente: Elaboración propia.

Figura 4.16: Caso de uso: Editar Estudiante

Caso de Uso "Editar Estudiante"		
Actor (s)	Profesor	
Condiciones iniciales	Estudiante registrado en el sistema	
Condiciones de salida	Modificación de Estudiante registrada en el sistema	
Flujo Principal	**Usuario**	**Sistema**
	1. El profesor pulsa "Estudiantes". 2. El profesor realiza una búsqueda en la tabla de registros de estudiantes, una vez seleccionado el registro, pulsa el botón "Modificar".	1. El sistema muestra una tabla con los estudiantes registrados. 2. El sistema muestra un formulario con los campos registrados en forma de edición, junto a las opciones "Guardar" y "Cancelar".
	3. El profesor modifica los campos deseados y pulsa "Guardar".	3. El sistema registra la modificación realizada y muestra un mensaje por pantalla "Modificación exitosa" y muestra la tabla de estudiantes registrados.
Flujo Alternativo	**Usuario**	**Sistema**
	1. Si el profesor introduce un datos de manera incorrecta.	1. El sistema muestra por pantalla el siguiente mensaje "Por favor introduzca correctamente los datos solicitados".
	2. Si el profesor omite llenar un campo solicitado.	2. El sistema muestra por pantalla el siguiente mensaje "Por favor introduzca la información solicitada".
	3. Si el Admin pulsa el botón "Cancelar".	3. El sistema limpia los campos llenos y retorna a la página principal del sistema.
Requisitos Especiales	1. El correo electrónico debe ser válido. 2. La contraseña debe contener una longitud de 8 a 16 caracteres, al menos una letra mayúscula y minúscula, dos	

Fuente: Elaboración propia.

Figura 4.17: Caso de uso: Detallar Estudiante

Caso de Uso "Detallar Estudiante".		
Actor (s)	Profesor	
Condiciones iniciales	Estudiante registrado en el sistema	
Condiciones de salida	Detallar información de estudiante por pantalla.	
Flujo Principal	**Usuario**	**Sistema**
	1. El Profesor pulsa la opción "Estudiantes".	1. El sistema muestra la tabla de registros de Estudianes.
	2. El Profesor realiza una búsqueda en la tabla de registros de Estudiantes, una vez seleccionado el registro, pulsa el botón "Detallar".	2. El sistema muestra la información de Estudiante por pantalla en campos no editables.

Fuente: Elaboración propia.

Figura 4.18: Caso de uso: Eliminar Estudiante

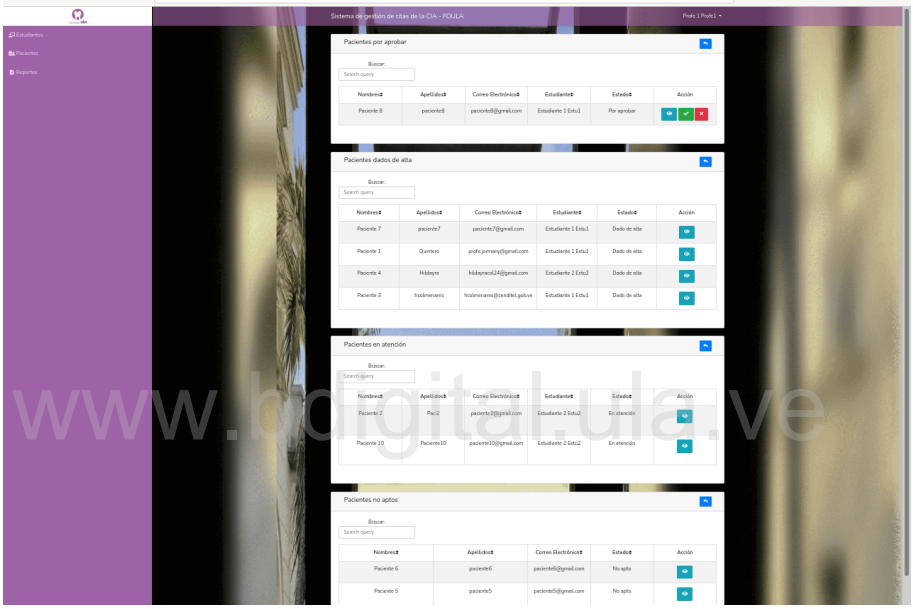
Caso de Uso "Eliminar Estudiante".		
Actor (s)	Profesor	
Condiciones iniciales	Estudiante registrado en el sistema	
Condiciones de salida	Registro de Estudiante eliminado del sistema	
Flujo Principal	**Usuario**	**Sistema**
	1. El profesor pulsa "Estudiantes". 2. El Profesor realiza una búsqueda en la tabla de registros de Estudiantes, una vez seleccionado el registro, pulsa el botón "Eliminar".	1. El sistema muestra una tabla con el registro de estudiantes. 2. El sistema El sistema muestra un mensaje por pantalla "¿Está seguro de querer eliminar este registro?" Junto a las opciones "Confirmar" y "Cancelar".
	3. El Profesor pulsa "Confirmar".	3. El sistema elimina el registro y muestra un mensaje por pantalla "Estudiante Eliminado con éxito".
Flujo Alternativo	**Usuario**	**Sistema**
	1. Si el Profesor pulsa el botón "Cancelar".	El sistema no ejecuta ninguna acción.

Fuente: Elaboración propia.

Aprobar Paciente

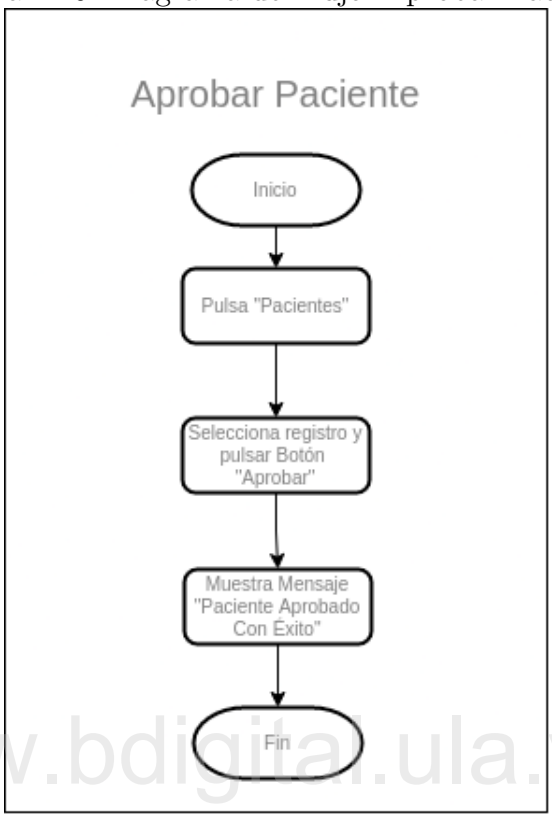
El proceso de aprobar paciente es realizado por el usuario profesor, una vez sus estudiantes han reservado el caso de un paciente para su evaluación (Ver figura 4.20 y la Figura 4.21). En la Figura 4.19, se puede detallar la vista de los pacientes que tiene el usuario profesor para estudiar los casos y aprobar o rechazar los reservados por los estudiantes de su tutoría.

Figura 4.19: Vista de pacientes con usuario Profesor



Fuente: Elaboración propia.

Figura 4.20: Diagrama de Flujo: Aprobar Paciente



Fuente: Elaboración propia.

Figura 4.21: Caso de uso: Aprobar Paciente

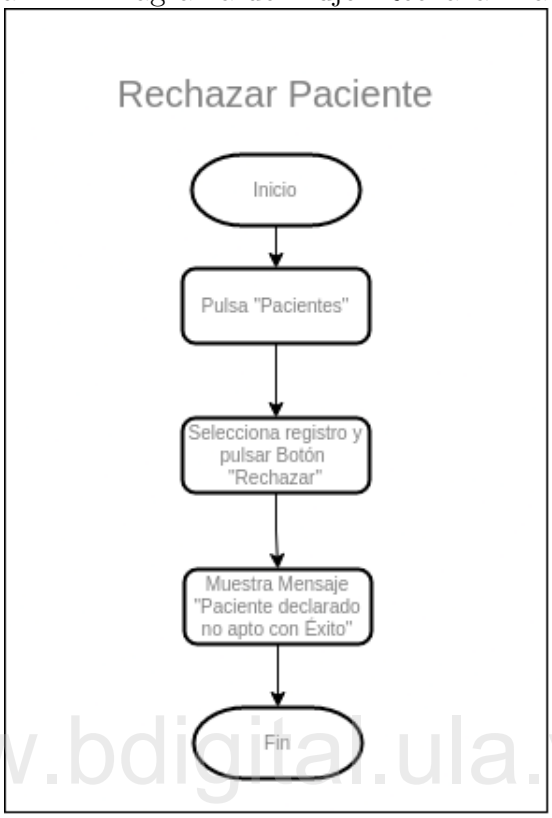
Caso de Uso Aprobar paciente		
Actor (s)	Profesor	
Condiciones iniciales	Paciente Reservado por estudiante de su catedra.	
Condiciones de salida	Paciente asignado a estudiante.	
Flujo Principal	**Usuario**	**Sistema**
	1. El profesor pulsa "Pacientes". 2. El Profesor selecciona registro de la tabla " Pacientes por aprobar y pulsa el botón **"Aprobar"**.	1- El sistema muestra las tablas de registros de "Pacientes Por aprobar", "Pacientes dados de alta", "Pacientes en atención", " Pacientes no aptos". 2. El sistema cambia el estado del paciente a "En atención", y lo asiga al estudiante que hizo la reserva en una tabla con las acciones (**Detallar, Asignar cita, Registrar historial, Dar de alta**).

Fuente: Elaboración propia.

Rechazar Paciente

El proceso de rechazar paciente es realizado por el usuario profesor, una vez sus estudiantes han reservado el caso de un paciente para su evaluación, en este caso, si el paciente no está apto para recibir el tratamiento por algún tipo de patología contraproducente, el profesor rechaza el caso y el estudiante continúa la búsqueda de un paciente que si pueda aplicar para sus prácticas clínicas (Ver Figura 4.22 y la Figura 4.23).

Figura 4.22: Diagrama de Flujo: Rechazar Paciente



Fuente: Elaboración propia.

Figura 4.23: Caso de uso: Rechazar Paciente

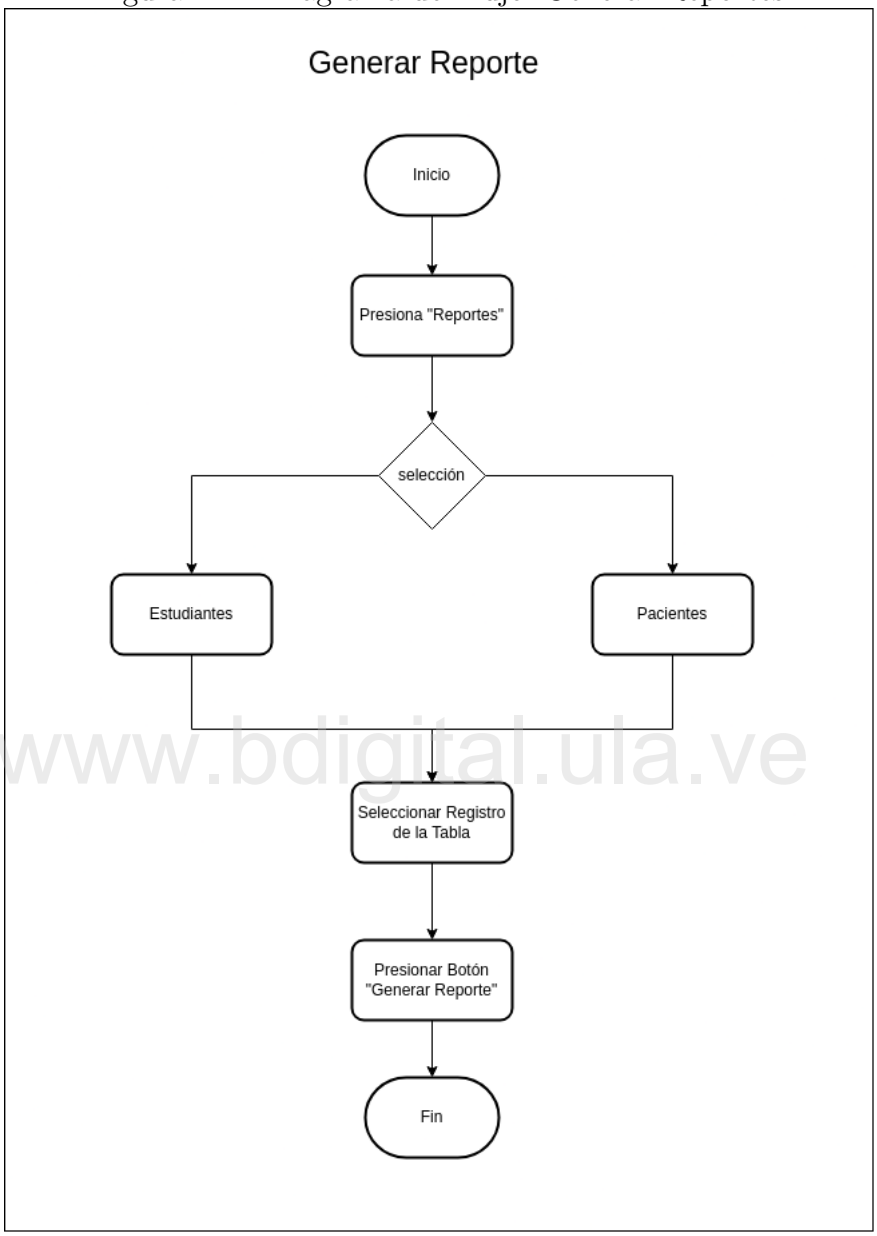
Caso de Uso Rechazar Paciente		
Actor (s)	Profesor	
Condiciones iniciales	Paciente Reservado por estudiante de su tutoría.	
Condiciones de salida	Paciente calificado como no apto	
Flujo Principal	**Usuario**	**Sistema**
	1. El profesor pulsa "Pacientes". 2. El Profesor selecciona registro de la tabla " Pacientes por aprobar y pulsa el botón **"Rechazar"**.	1- El sistema muestra las tablas de registros de "Pacientes Por aprobar", "Pacientes dados de alta", "Pacientes en atención", " Pacientes no aptos". 2. El sistema cambia el estado del paciente a "No apto", lo elimina de la tabla de pacientes por aprobar del estudiante y lo asiga a la tabla de no aptos del profesor, su registro ya no será visible en otras tablas y no se podrá reservar por otro estudiante.

Fuente: Elaboración propia.

Generar reporte

El usuario profesor puede generar reportes de los estudiantes de su tutoría y de pacientes (Ver Figura 4.25 y la Figura 4.24).

Figura 4.24: Diagrama de Flujo: Generar Reportes



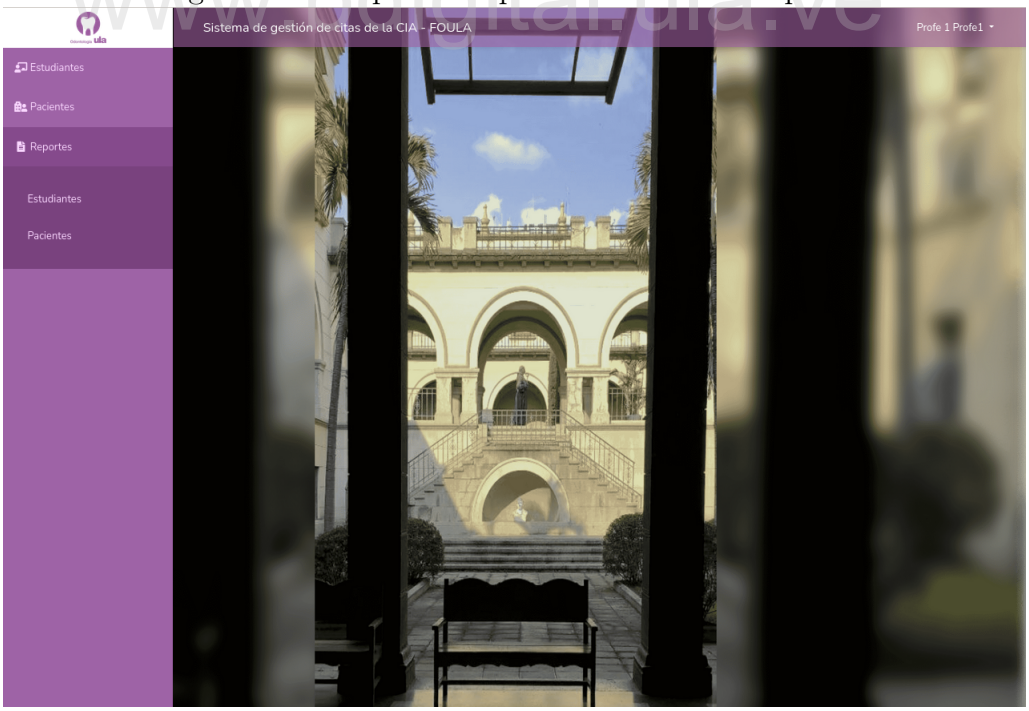
Fuente: Elaboración propia.

Figura 4.25: Caso de uso: Generar Reportes

Caso de uso Generar Reportes		
Actor (s)	Profesor.	
Condiciones iniciales	Registro en el sistema.	
Condiciones de salida	Impresión de reporte por pantalla.	
Flujo Principal	**Usuario**	**Sistema**
	1. El profesor pulsa **Reportes.** 2- El profesor selecciona Estudiantes o pacientes. 3- El profesor selecciona registro y pulsa la opción **generar reporte**.	1. El sistema despliega las opciones : Estudiantes y pacientes. 2. El sistema muestra la tabla de registros de la opción seleccionada. 3. El sistema muestra el PDF del registro seleccionado por pantalla.

Fuente: Elaboración propia.

Figura 4.26: Captura de pantalla: Generar Reportes

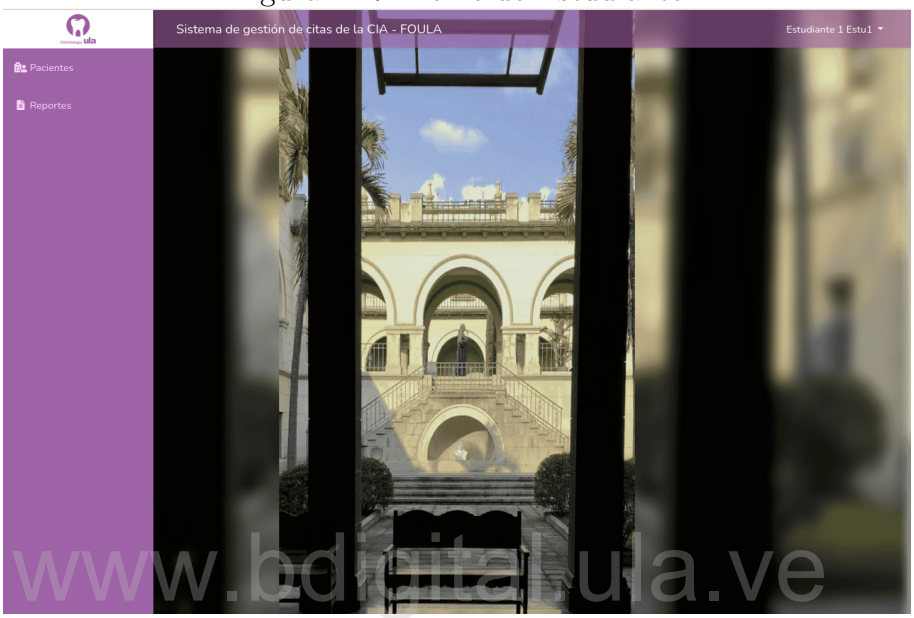


Fuente: Elaboración propia.

Rol: Estudiante

En la Figura 4.27 se puede detallar el Home del usuario o rol estudiante, y como se observa en el panel lateral izquierdo, este tiene acceso al módulo de Pacientes y Reportes.

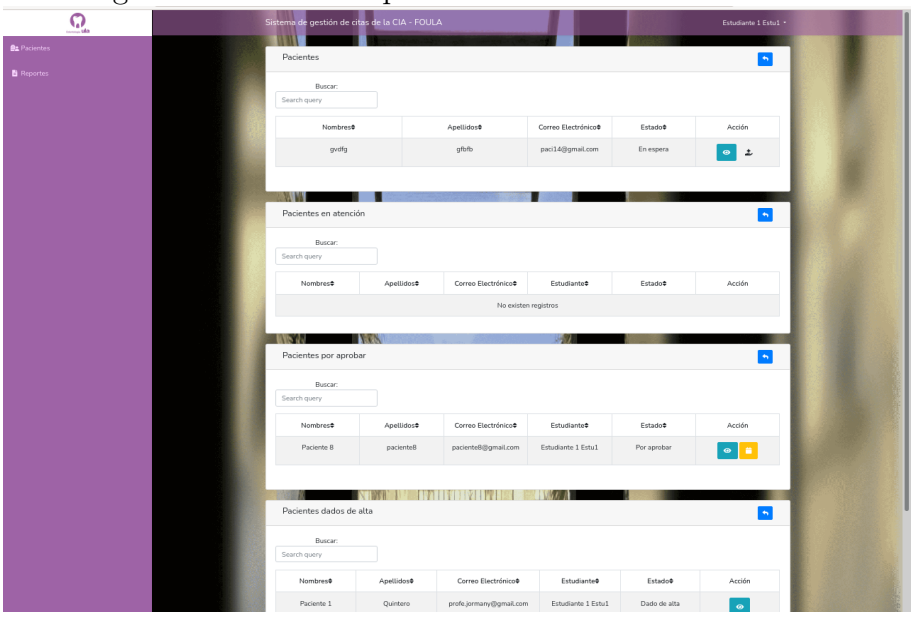
Figura 4.27: Home de Estudiante



Fuente: Elaboración propia.

En la Figura 4.28, se pueden detallar las tablas de pacientes con el rol estudiante, en las cuales el mismo puede reservar, asignar citas, registrar historial y dar de alta a sus pacientes en proceso de atención.

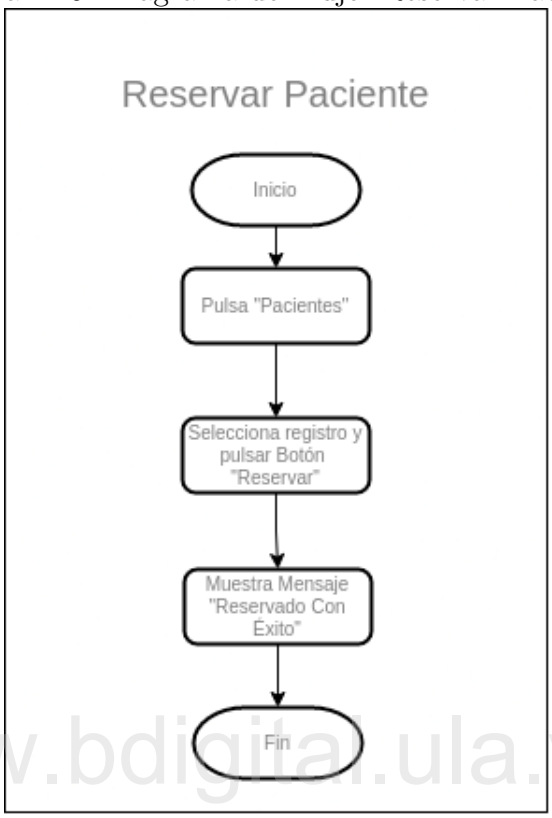
Figura 4.28: Vista de pacientes con usuario Estudiante



Fuente: Elaboración propia.

Reservar paciente: La reserva del paciente es realizada por el estudiante con el propósito de estudiar junto al profesor si el caso aplica para realizar el tratamiento específico de la práctica clínica.(Ver Figura 4.29 y Figura 4.30). Una vez el estudiante reserva el paciente, se envía un correo al profesor para notificar esta acción. (Ver Figura 4.31

Figura 4.29: Diagrama de Flujo: Reservar Paciente



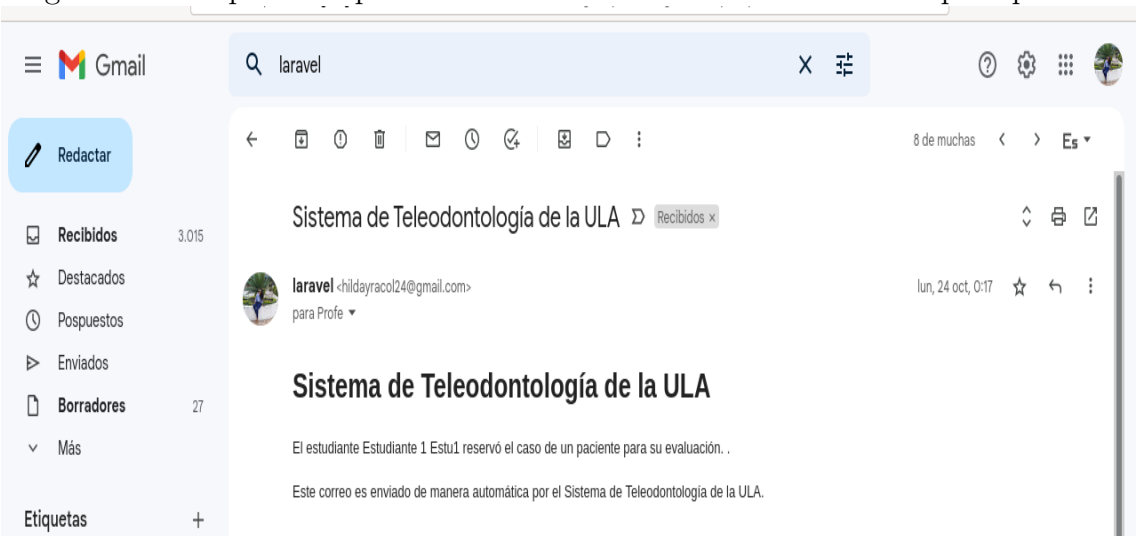
Fuente: Elaboración propia.

Figura 4.30: Caso de uso: Reservar Paciente

Caso de Uso Reservar Paciente		
Actor (s)	Estudiante.	
Condiciones iniciales	Paciente Registrado	
Condiciones de salida	Paciente reservado por estudiante.	
Flujo Principal	**Usuario**	**Sistema**
	1. El Estudiante pulsa " "Pacientes". 2. El Estudiante realiza una búsqueda en la tabla de registros de pacientes, pulsa la opción **Reservar paciente**.	1. El sistema muestra las tablas de registros de "Pacientes" "Pacientes dados de alta", "Pacientes en atención". 2. El sistema quita el registro de la tabla de pacientes, lo pasa a la tabla de "Pacientes reservados" y envía una notificación (Pre-diseñada) vía correo electrónico al profesor que el estudiante hizo una reserva de paciente para el estudio de su caso y espera su aprobación. Además, el sistema impide el acceso a otros estudiantes para detallar el caso.

Fuente: Elaboración propia.

Figura 4.31: Captura de pantalla: Correo de notificación de reserva para profesor

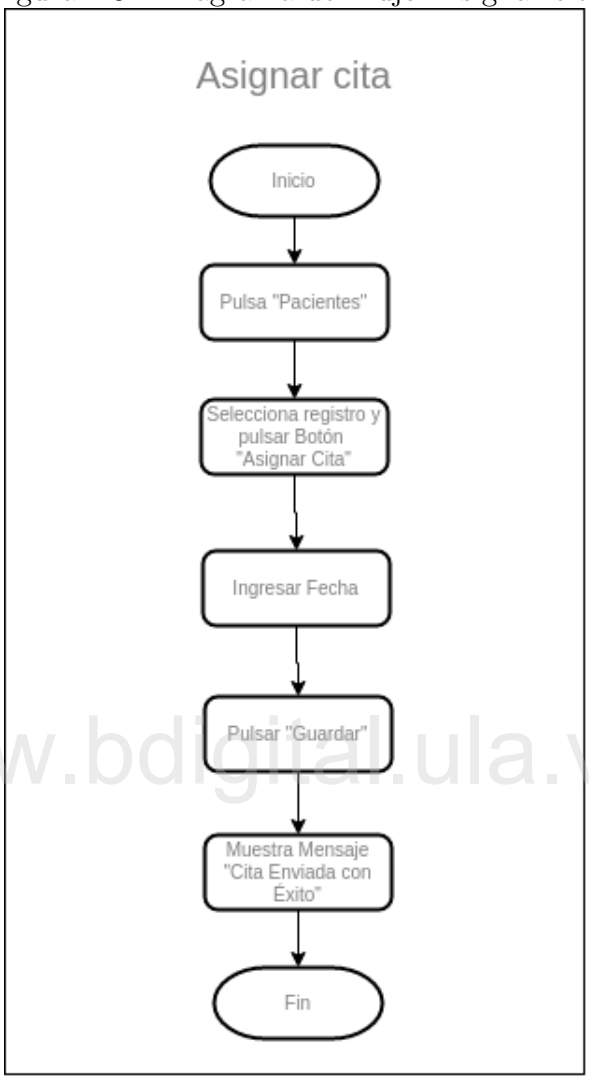


Fuente: Elaboración propia.

Asignar cita al paciente

Esta acción es ejecutada una vez luego de que el estudiante ha reservado al paciente, de modo que en esta primera cita se realice la evalución presencial junto al profesor, para su posterior aprobación o declaración de (No apto). De igual manera si el profesor aprueba la reserva del paciente, el estudiante podrá continuar asignando citas al paciente hasta finalizar el proceso de atención. Ver Figura 4.32, Figura 4.33, Figura 4.34 y Figura 4.35.

Figura 4.32: Diagrama de Flujo: Asignar cita



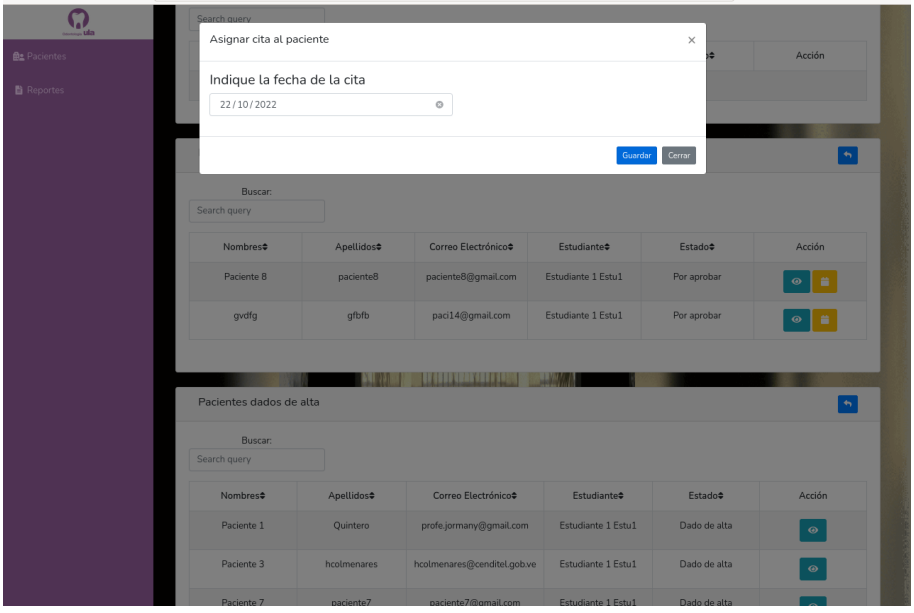
Fuente: Elaboración propia.

Figura 4.33: Caso de uso: Asignar Cita

Caso de Uso Asignar Cita		
Actor (s)	Estudiante.	
Condiciones iniciales	Paciente Reservado o en atención por estudiante.	
Condiciones de salida	Enviar Cita	
Flujo Principal	**Usuario**	**Sistema**
	1. El Estudiante pulsa “ “Pacientes”. 2. El estudiante realiza una búsqueda en la tabla de registros de “Pacientes por aprobar” o en “Pacientes en atención”, selecciona registro y pulsa la opción **Asignar cita, ingresa la fecha y pulsa “Guardar”**.	1. El sistema muestra las tablas de registros de “Pacientes” “Pacientes Por aprobar”, “Pacientes dados de alta”, “Pacientes en atención”. 2. El sistema envía un correo electrónico (Prediseñado) al paciente indicando el nombre del estudiante , el día y la fecha que lo atenderá.

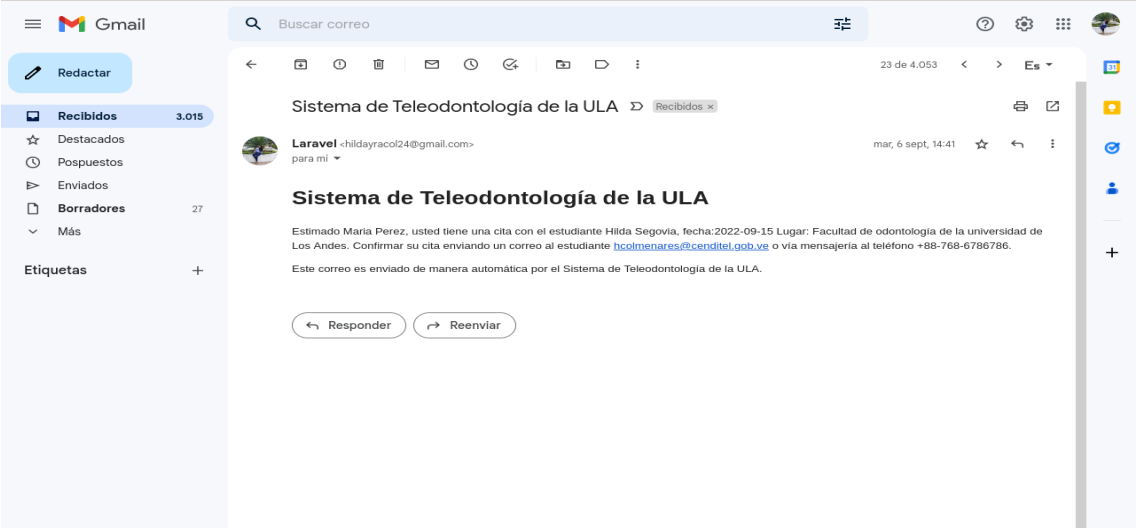
Fuente: Elaboración propia.

Figura 4.34: Captura de pantalla: Asignar Cita



Fuente: Elaboración propia.

Figura 4.35: Captura de pantalla: Correo recibido por paciente

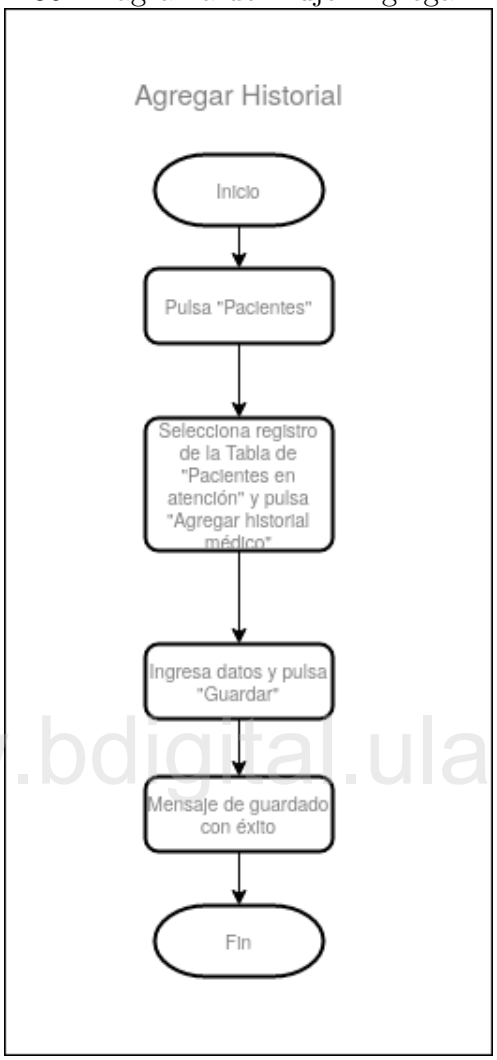


Fuente: Elaboración propia.

Agregar historial

Esta acción es ejecutada por el estudiante a sus pacientes en atención, de modo que puede llevar un registro histórico de la atención dada a sus pacientes y el profesor puede tener acceso a dicha información. Para detallar el fujo de este proceso Ver Figura 4.36, la Figura 4.37 y 4.38.

Figura 4.36: Diagrama de Flujo: Agregar historial



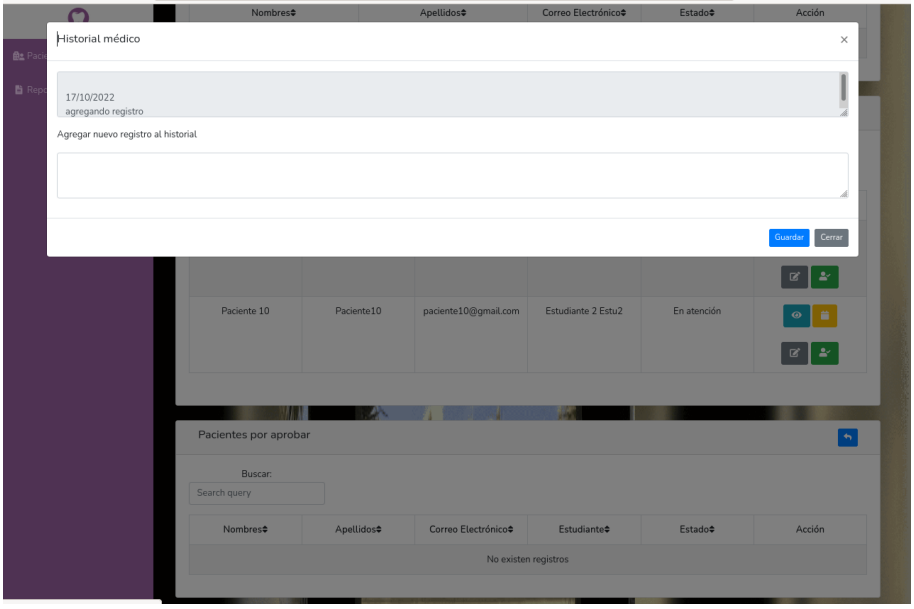
Fuente: Elaboración propia.

Figura 4.37: Caso de uso: Agregar historial

Caso de uso Registrar Historial		
Actor (s)	Estudiante	
Condiciones iniciales	Paciente en atención.	
Condiciones de salida	Historial guardado en el sistema.	
Flujo Principal	**Usuario**	**Sistema**
	1. El estudiantes selecciona "Pacientes". 2. El estudiante selecciona registro de pacientes en atención y pulsa **Agregar historial médico al paciente.** 3. El estudiante igresa el historial y pulsa "Guardar".	1. El sistema muestra las tablas de registros de "Pacientes dados de alta", "Pacientes en atención", "Pacientes por aprobar". 2. El sistema muestra el campo "Agregar nuevo registro al historial", en editable. También muestra el registro anterior en campo no editable y las opciones "Guardar" y "Cancelar". 3. El sistema guarda el registro y muestra por pantalla un mensaje " Registro actualizado con éxito".
Flujo Alternativo	1. Si el estudiante pulsa "Cancelar".	1. El sistema no ejecuta ninguna acción.

Fuente: Elaboración propia.

Figura 4.38: Captura de pantalla: Agregar historial

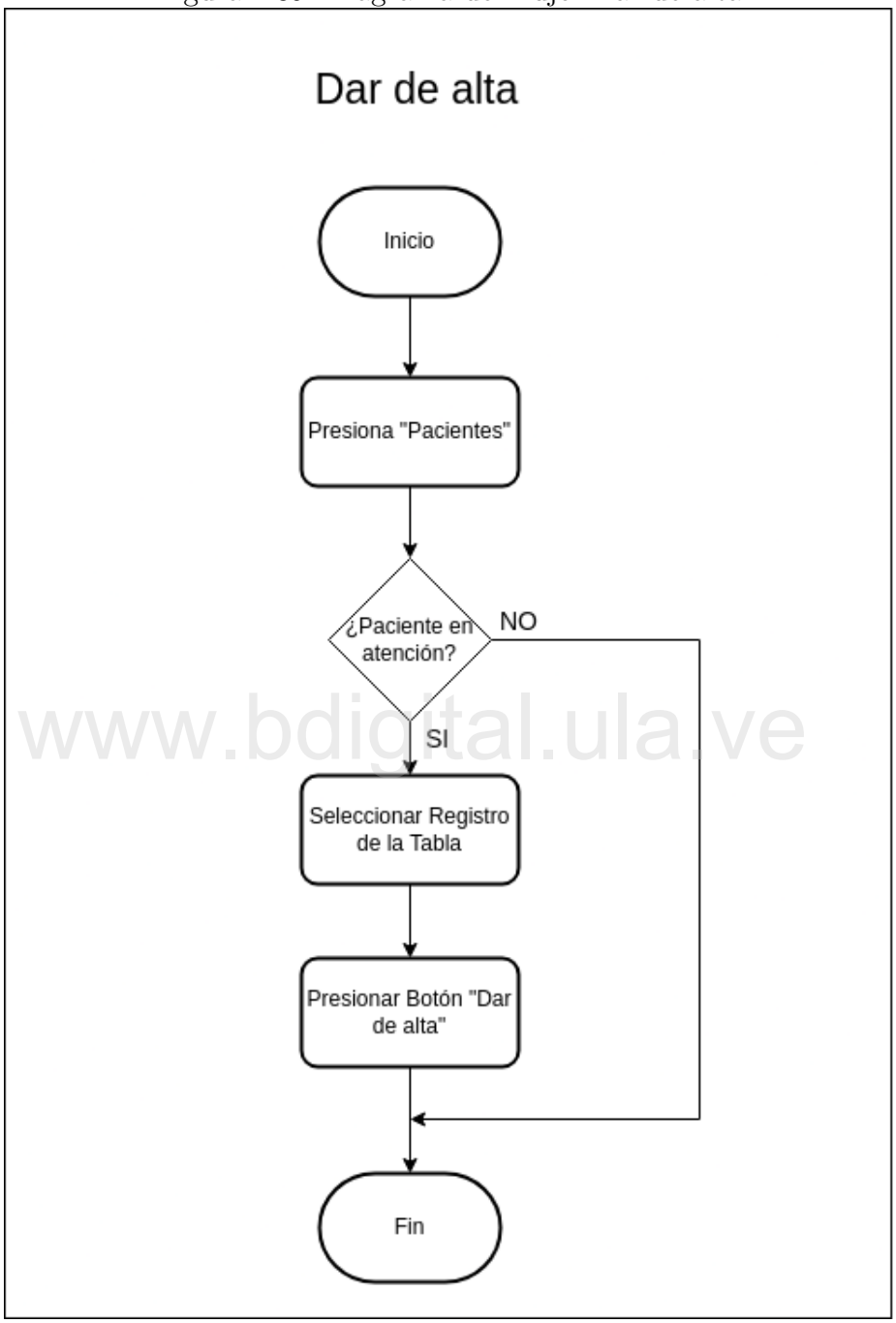


Fuente: Elaboración propia.

Dar de alta

Esta acción es realizada por el paciente para definir la finalización de su proceso de atención al paciente. Para detallar el fujo de este proceso Ver Figura 4.39 y la Figura 4.40.

Figura 4.39: Diagrama de Flujo: Dar de alta



Fuente: Elaboración propia.

Figura 4.40: Caso de uso: Dar de alta

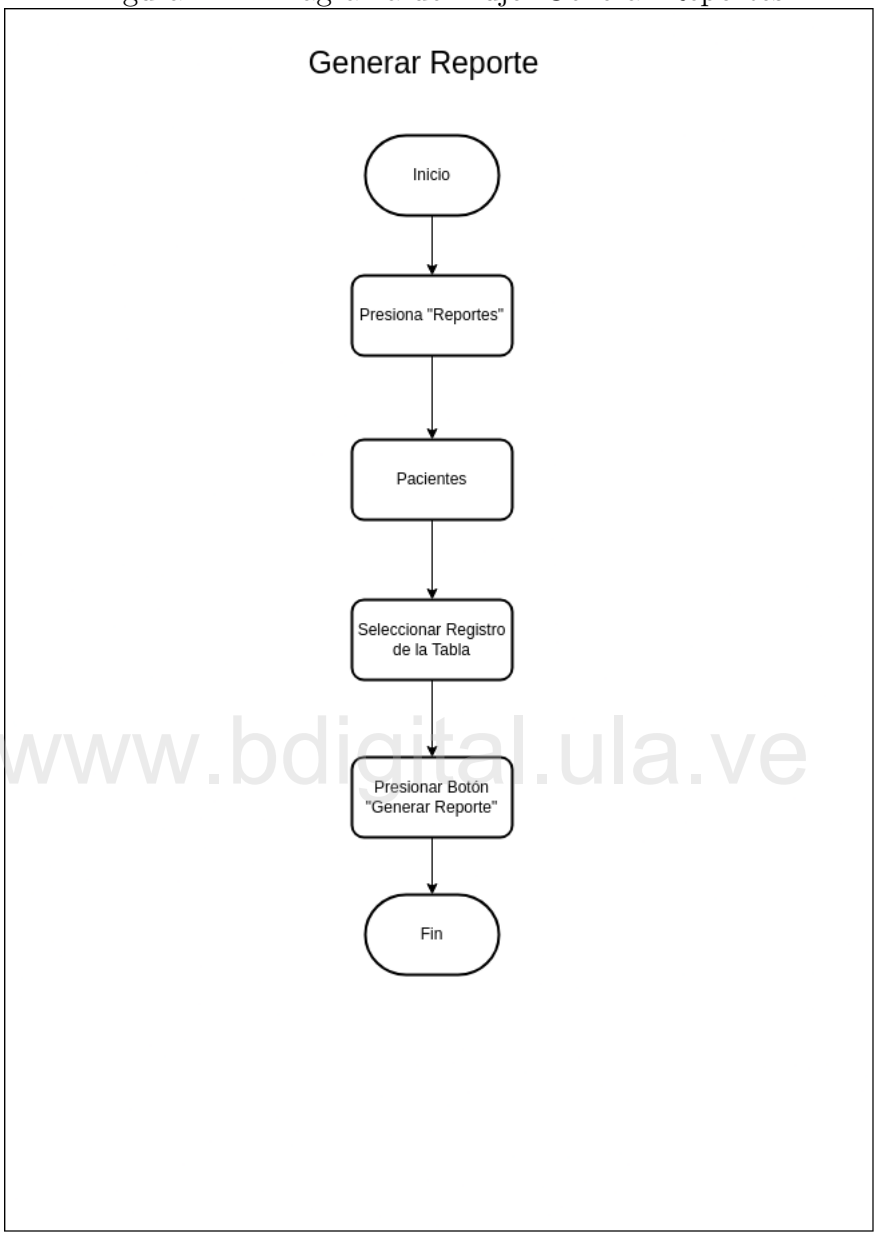
Caso de Uso Dar de alta		
Actor (s)	Estudiante.	
Condiciones iniciales	Paciente En atención por el estudiante	
Condiciones de salida	Paciente retirado de la tabla de pacientes en atención a la de pacientes dados de alta.	
Flujo Principal	**Usuario**	**Sistema**
	1. El Estudiante pulsa " "Pacientes". 2. El estudiante realiza una búsqueda en la tabla de registros de "Pacientes en atención", selecciona registro y pulsa la opción **Dar de alta.**	1. El sistema muestra las tablas de registros de "Pacientes" "Pacientes Por aprobar", "Pacientes dados de alta", "Pacientes en atención". 2. El sistema modifica el estado del paciente a "de alta", y muda el registro a la tabla de pacientes dados de alta. Además muestra un mensaje por pantalla: "Paciente dado de alta con éxito"

Fuente: Elaboración propia.

Generar reporte

El usuario Estudiante, podrá generar reportes asociados a los pacientes que posea en reserva y proceso de atención.(Ver Figura 4.41, Figura 4.42 y Figura 4.43).

Figura 4.41: Diagrama de Flujo: Generar Reportes



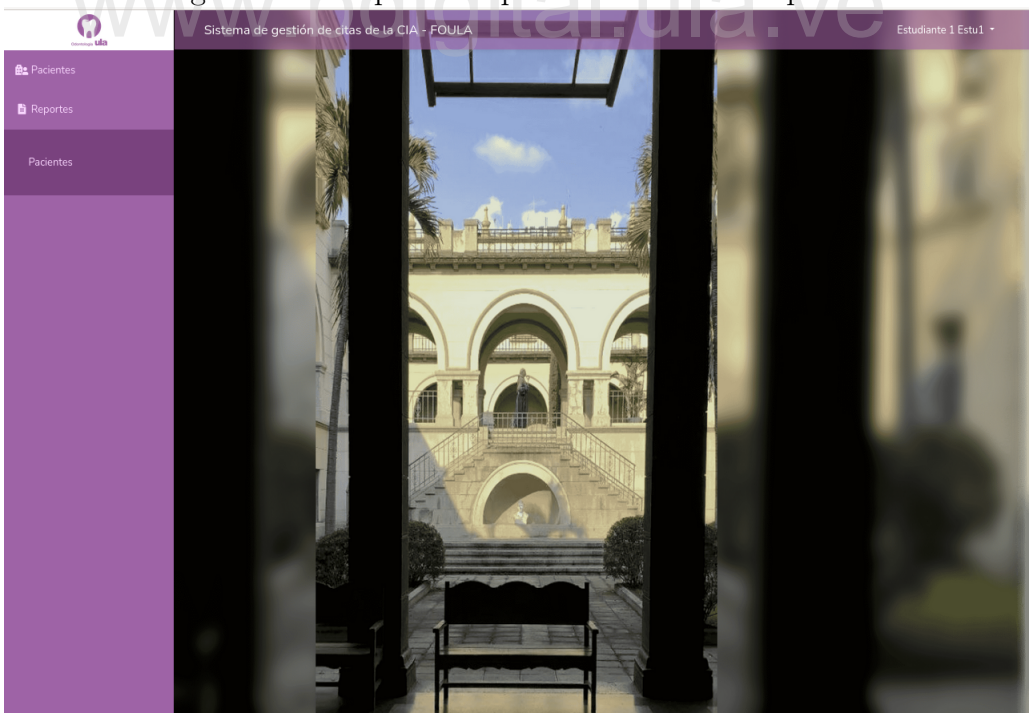
Fuente: Elaboración propia.

Figura 4.42: Caso de uso: Generar Reportes

Caso de uso Generar Reportes		
Actor (s)	Estudiante.	
Condiciones iniciales	Registro en el sistema.	
Condiciones de salida	Impresión de reporte por pantalla.	
Flujo Principal	**Usuario**	**Sistema**
	1. El Estudiante pulsa **Reportes.** 2- El Estudiante selecciona pacientes. 3- El Estudiante selecciona registro y pulsa la opción **generar reporte**.	1. El sistema despliega las opciones : Estudiantes y pacientes. 2. El sistema muestra la tabla de registros de pacientes. 3. El sistema muestra el PDF del registro seleccionado por pantalla.

Fuente: Elaboración propia.

Figura 4.43: Captura de pantalla: Generar Reportes



Fuente: Elaboración propia.

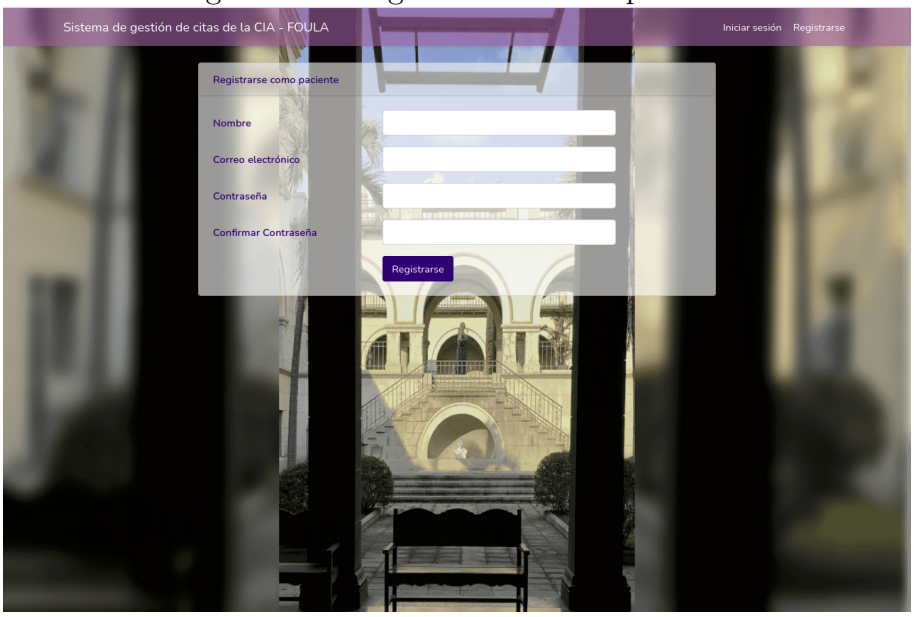
Rol: Paciente

En la Figura 4.44 se pueden detallar en la esquina superior derecha, las opciones de inicio de sesión y registrarse, pulsando esta segunda opción el usuario paciente puede realizar su registro de usuario como se muestra en la Figura 4.45 y posteriormente acceder al home mostrado en la figura 4.46.

Figura 4.44: Home Principal

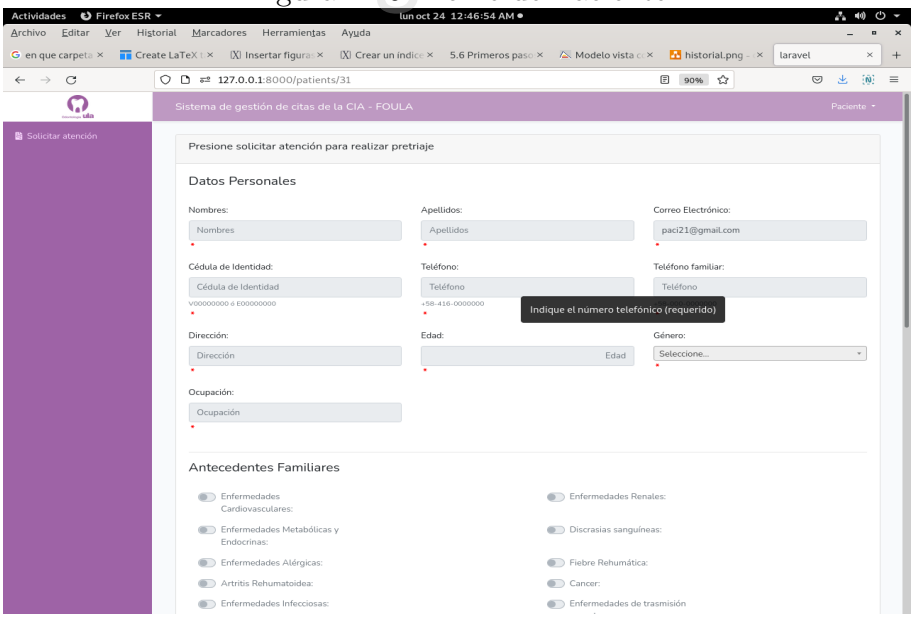
Fuente: Elaboración propia.

Figura 4.45: Registro de usuario paciente



Fuente: Elaboración propia.

Figura 4.46: Home de Paciente



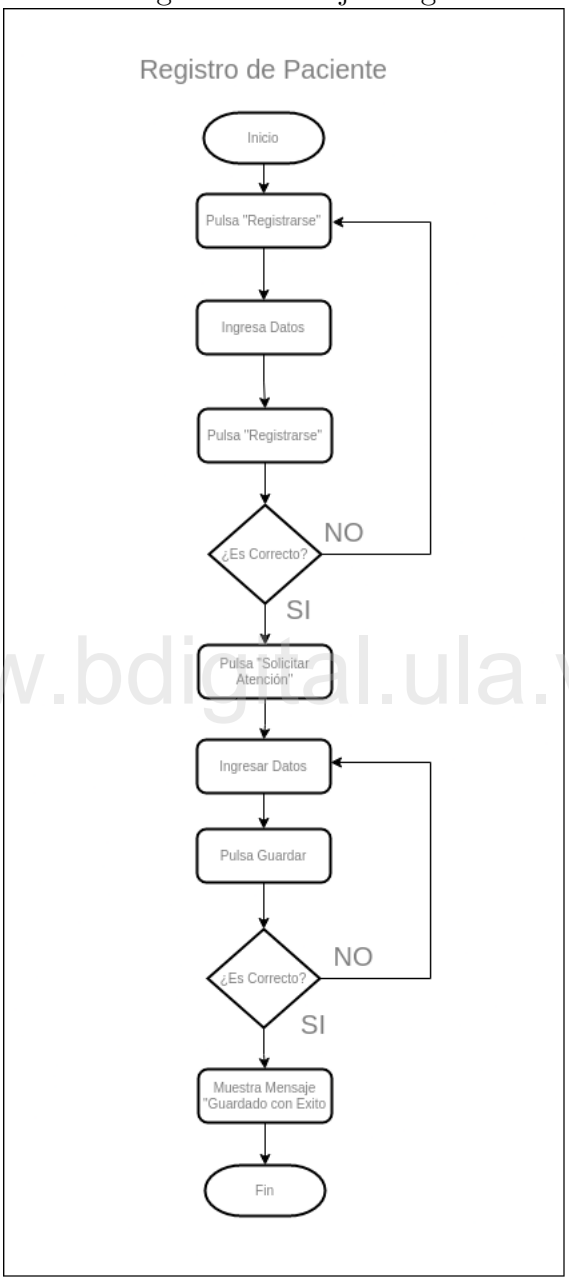
Fuente: Elaboración propia.

Registro de paciente y pretriaje

Una vez generadas sus credenciales de usuario, el paciente puede solicitar atención

realizando el pretriaje (Ver Figura 4.47, Figura 4.48 y Figura 4.49).

Figura 4.47: Diagrama de Flujo: Registrar Paciente



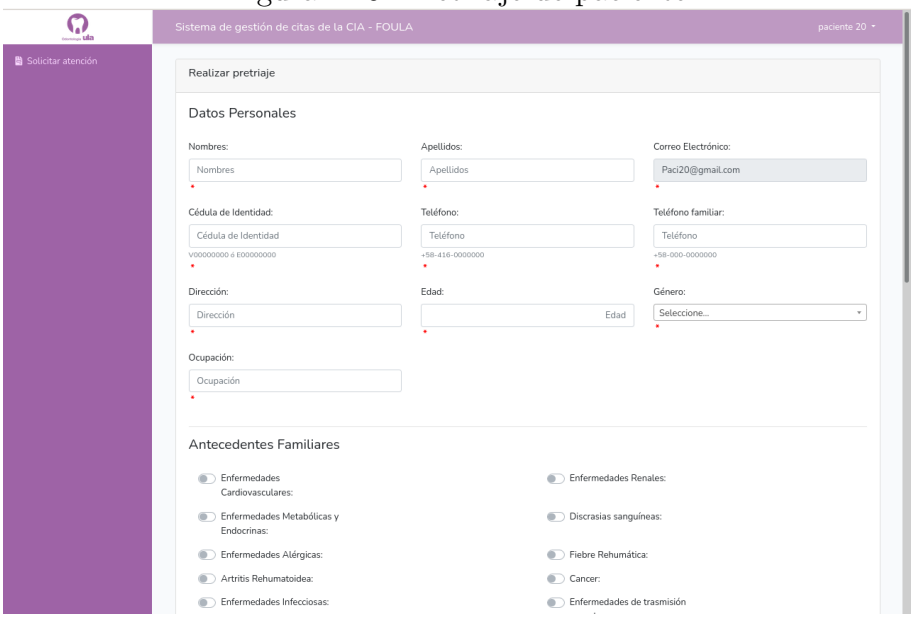
Fuente: Elaboración propia.

Figura 4.48: Caso de uso: Registrar Paciente

Caso de uso Registrar paciente		
Actor (s)	Paciente	
Condiciones iniciales	Correo no registrado en el sistema.	
Condiciones de salida	Información de paciente registrada en el sistema.	
Flujo Principal	**Usuario**	**Sistema**
	1. El usuario pulsa "Registrarse" en la página de inicio de sesión. 2. El usuario ingresa los datos y pulsa Registrarse. 3. El usuario pulsa "Solicitar atención". 4. El usuario ingresa los datos solicitados y pulsa "guardar".	1. El sistena muestra el formulario de registro de paciente con los siguientes campos: a. Nombre b. Correo Electrónico c. Contraseña d. Confirmar Contraseña 2. El sistema muestra el formulario de pretriaje en campos no editables, la sección solicitar atención en la barra lateral izquierda y muestra el mensaje "Presione solicitar atención para realizar pretriaje". 3. El sistema muestra un formulario en 3 secciones: A- DATOS PERSONALES. B- ANTECEDENTES FAMILIARES, C-ANTECEDENTES PERSONALES). 4. El sistema guarda la información y muestra un mensaje por pantalla. "Registrado con éxito".
Flujo Alternativo	**Usuario**	**Sistema**
	1. Si el Paciente introduce un datos de manera incorrecta u omite un campo obligatorio. 2. Si el Paciente pulsa "Cancelar"	1. El sistema muestra por pantalla el siguiente mensaje "Por favor introduzca correctamente los datos". 2. El sistema no ejecuta ninguna acción.
Requisitos especiales	1. El correo electrónico debe ser válido. 2. El campo cédula debe ser numérico. 3. Número de teléfono válido 4. La imagen u archivo debe estar en formato jpj, menor o igual a 1080x1080	

Fuente: Elaboración propia.

Figura 4.49: Pretriaje de paciente

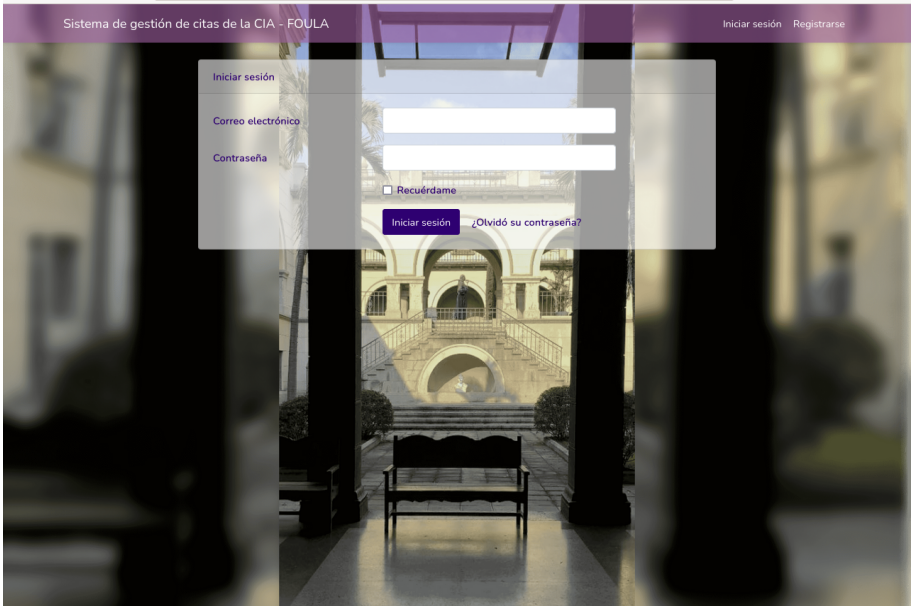


Fuente: Elaboración propia.

Funcionalidades comunes

En la figura 4.50, se puede detallar la pantalla de inicio de sesion, la cual es común para los roles Jefe de cátedra, Profesor, Estudiante y Paciente, quienes podrán acceder al sistema con sus respectivas credenciales previamente generadas.

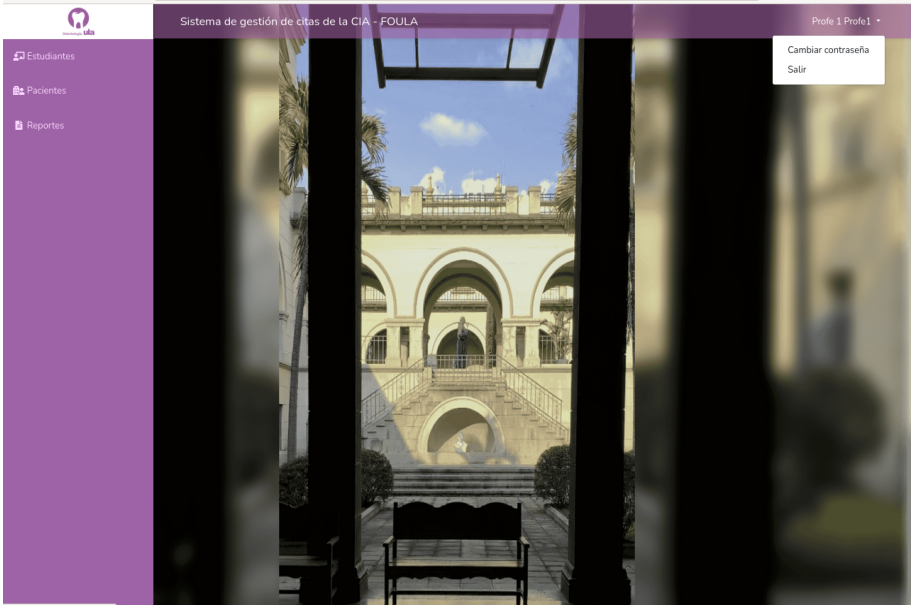
Figura 4.50: Inicio de sesión



Fuente: Elaboración propia.

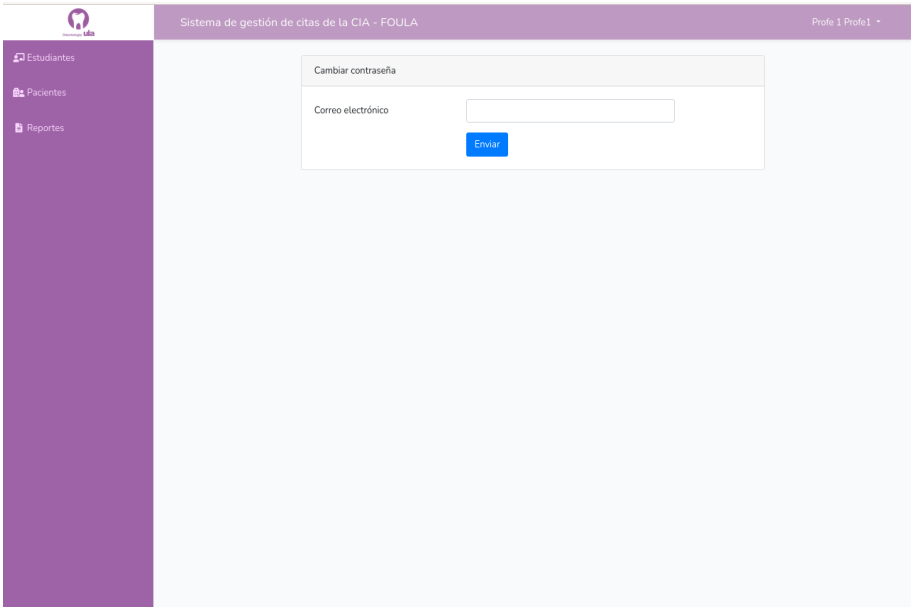
En la Figura 4.51, se puede observas la funcionalidad de cambiar contraseña al pulsar sobre el nombre del usuario ubicado en la esquina superior izquierda de la pantalla. Así mismo en las Figuras 4.51, 4.52, 4.53 y 4.54 se muestra el flujo del sistema para realizar dicha acción.

Figura 4.51: Captura de pantalla: Cambiar Contraseña (Paso 1)



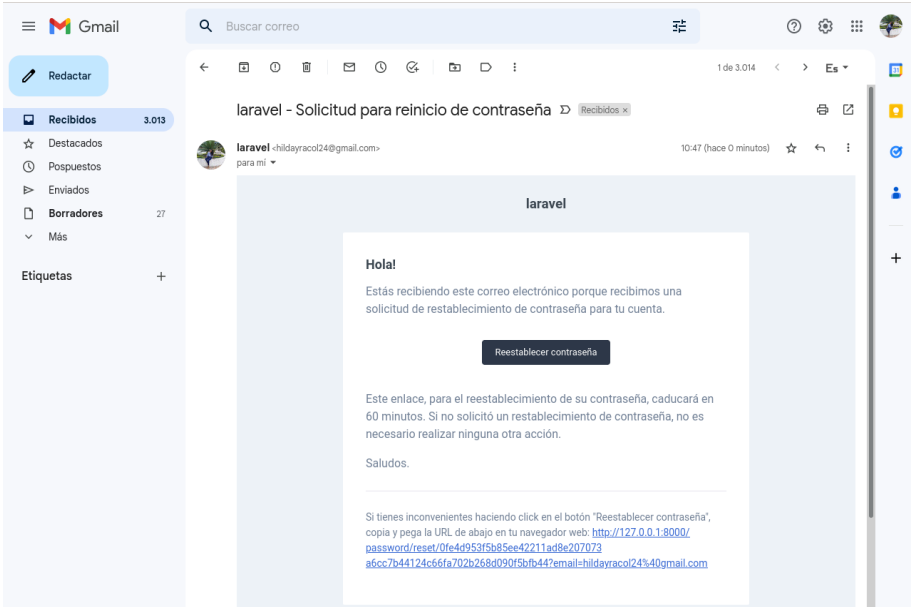
Fuente: Elaboración propia.

Figura 4.52: Captura de pantalla: Cambiar Contraseña (Paso 2)



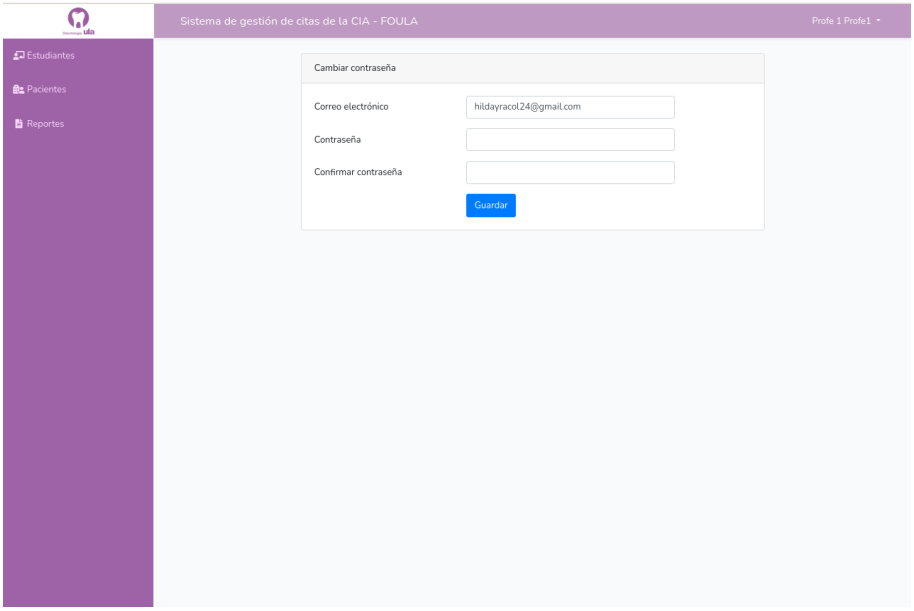
Fuente: Elaboración propia.

Figura 4.53: Captura de pantalla: Cambiar Contraseña (Paso 3)



Fuente: Elaboración propia.

Figura 4.54: Captura de pantalla: Cambiar Contraseña (Paso 4)



Fuente: Elaboración propia.

Capítulo 5

Conclusiones

Actualmente la CIA-FOULA, realiza sus procesos para agendar citas de forma manual, de modo que los pacientes deben dirigirse de manera presencial al centro de atención para programar citas, llenando el formulario de pretriaje para el estudio de su caso, teniendo en cuenta que puede o no aplicar para recibir el tratamiento solicitado, en este sentido el paciente puede no aplicar para recibir la atención por diversos factores que pudieron ser previstos en sus antecendentes, o aplicar pero tener que regresar en otra ocasión debido a la aglomeración de personas que llegaron previamente. este tipo de situaciones desembocan en largos tiempos de espera, aglomeración de pacientes, pérdida de tiempo y gastos innecesarios en el traslado.

Por otro lado se encuentra el manejo de la información descentralizada, lo que ocasiona retrasos en los procesos, perdida de información, molestias en profesores y estudiantes.

Por tal motivo, con la intención de brindar una herramienta tecnológica de apoyo, se planteó el desarrollo del presente trabajo de grado, a fin de gestionar las citas odontológicas de la CIA-FOULA, brindando la posibilidad de que los pacientes puedan realizar la solicitud de atención via online desde el lugar de su comodidad, aportando una herramienta de pretriaje para el estudio de los casos, un canal de seguimiento entre el profesor y el alumno y la centralización y resguardo de la información.

En consecuencia a lo anteriormente expresado, se lograron alacazar los objetivos propuestos:

Objetivo 1: Se identificó y describió el comportamiento del sistema a través de la especificación de casos de uso, partiendo de un levantamiento de requerimientos realizado en reunión con parte del personal docente y estudiantil de la Facultad de Odontología.

Objetivo 2: Se implementó la metodología ágil scrum en su aspecto técnico para desarrollar las funcionalidades descritas y se validaron constantemente los productos terminados a fin de que se ajustaran a la realidad de los procesos ejecutados actualmente en la CIA-FOULA.

Objetivo 3: Se definieron y desarrollaron 4 módulos para gestionar los registros del personal docente encargado de evaluar las clínicas, el personal estudiantil que cursa la materia y los pacientes que son atendidos durante estos periodos, así como lo relacionado a los reportes asociados a estos procesos y usuarios del sistema.

Objetivo 4: Se validaron los prototipos del sistema tomando en cuenta la experiencia de parte del personal estudiantil, conclueyendo que los procesos estan ajustados a la dinamica de la clínica.

En este sentido, el sistema de gestión de citas para la CIA-FOULA, cuenta con los módulos Profesores, estudiantes, pacientes y reportes, con funcionalidades como registros CRUD, reservar pacientes, asignar citas de manera virtual a través del envío de correos electrónicos, registrar historial de pacientes, dar de alta y rechazar caso de paciente. El mismo puede ser accedido desde el portal web de cualquier dispositivo con internet, cuenta con una interfaz agradable, responsiva e intuitiva, lo que supone una mejora en cuanto a la optimización del tiempo empleado en el registro y acceso a la

información, disminución de gastos de traslado, y mitigar la propagación del COVID-19.

5.1. Recomendaciones

Se recomienda la implementación del sistema en un servidor web, que permita el aprovechamiento de los distintos usuarios contemplados en la investigación, siguiendo las instrucciones expresadas en este documento y en el README generado junto al código del sistema. Por otra parte, dada la bondad escalable del sistema, se recomienda la creación de nuevos módulos para la atención en tiempo real, la actualización de las versiones de los lenguajes usados para el desarrollo. Contemplar la posibilidad de la implementación de un aplicativo móvil a futuro.

Bibliografía

Asamblea Nacional de Venezuela. (2015). *Ley de Telesalud*.

Avila, T. (2018). *Sistema Web con App Móvil para Gestión de Citas Médicas y Estadísticas* (Máster universitario en Ingeniería de software y sistemas informáticos, Universidad Internacional de Rioja). Recuperado desde https://reunir.unir.net/bitstream/handle/123456789/7434/TRELLES%5C%20AVILA%5C%2c%5C%20WILLIAMS%5C%20HIDALGO.pdf?sequence=1%5C&isAllowed=y

Baez, S. (2012). Sistemas web. KnowDo. Recuperado desde http://www.knowdo.org/knowledge/39-sistemas-web

Benítez, D. (2012). *Sistemas de información, aplicación en empresas*. [PDF file]. Recuperado desde https://www.eumed.net/ce/2012/ddb.pdf

Betancourt, N. & Bermúdez, J. (2021). Percepción de los odontólogos sobre la viabilidad de la aplicación de teleodontología en Mérida. *Rev Venez Invest Odont IADR*, (9), 40-59.

Castillo, A. & Morales, L. (2016). *Sistema web para el consultorio odontológico DENTI DANNA* (Ingeniero de sistemas, Universidad Distrital Francisco José de Caldas). Recuperado desde https://repository.udistrital.edu.co/bitstream/handle/11349/7279/ANDRES%5C%20CAMILO%5C%20CASTILLO%5C%20GONZALEZ%5C%202016.pdf?sequence=1

Chá, M. (2020). Telemedicina: su rol en las organizaciones de salud. *Rev Méd urug*. Recuperado desde https://doi.org/10.29193/RMU.36.4.9

Edex. (2022). Framework. Edex. Recuperado desde https://www.edix.com/es/instituto/framework/

García, E. (2019). ¿Que es Vue.JS? Códigofacilito. Recuperado desde https://codigofacilito.com/articulos/que-es-vue

Gonzales, E. (2017). *Implementar un sistema web para la gestión clínica dental, aplicando tecnologías open source: caso Consultorio Odontológico Navarro* (Ingeniero de sistemas, Universidad Estatal Península de Santa Elena). Recuperado desde http://repositorio.uncp.edu.pe/bitstream/handle/20.500.12894/4631/Huaylinos%5C%20Gonzale.pdf?sequence=1%5C&isAllowed=y

Huaylinos, G. (2017). *Metodologías ágiles en la implementación de una aplicación móvil para la gestión de citas en la clínica dental PERIO DENT - Huncayo* (Maestro en ingeniería de sistemas mención en gerencia de sistemas empresariales, Universidad Nacional del centro de Perú). Recuperado desde http://repositorio.uncp.edu.pe/bitstream/handle/20.500.12894/4631/Huaylinos%5C%20Gonzale.pdf?sequence=1

Laravel. (2022). Laravel. https://laravel.com/docs/8.x.

Laudon, k. & Laudon, P. (2016). *Sistemas de información gerencial* (Décimocuarta edición). PEARSON. Recuperado desde http://cotana.informatica.edu.bo/downloads/ld-Sistemas_de_informacion_gerencial_14%20edicion.pdf

Llivicura, W. & Ulloa, M. (2018). *Diseño y despliegue de una plataforma integral de telemedicina en una nube privada basada en OpenStack* (Ingeniero de sistemas, Universidad Politécnica Salesiana). Recuperado desde https://dspace.ups.edu.ec/bitstream/123456789/15751/1/UPS-CT007729.pdf

López, M. & Rodríguez, D. (2020). *Implementación de un aplicativo multiplataforma para la gestión de citas e historias médicas en la clínica odontológica Dental Care* (Ingeniero en computación e informática, Universidad Agraria del Ecuador). Recuperado desde https://181.198.35.98/Archivos/LOPEZ%5C%20SALAZAR%5C%20MARTHA%5C%20DEL%5C%20ROSARIO%5C%20.pdf

Marini, E. (2012). *El modelo cliente servidor*. [PDF file]. Recuperado desde https://www.linuxito.com/docs/el-modelo-cliente-servidor.pdf

Marquéz, M. (2011). *Bases de datos* (Primera edición). Publicacions de la Universitat Jaume I. Recuperado desde libros.metabiblioteca.com.co/bitstream/001/353/5/978-84-693-0146-3.pdf

Miranda, S. (2015). *Análisis y diseño de aplicación móvil para citas en consultorios odontológicos particulares en la ciudad de Piura* (Ingeniero industrial y de sistemas, Universidad de Piura). Recuperado desde https://pirhua.udep.edu.pe/bitstream/handle/11042/2445/ING%5C_559.pdf

Morón, M. (2021). La teleodontología una herramienta fundamental en tiempos de pandemia y post COVID -19, su utilidad en las diferentes especialidades odontológicas. *Int. J. Odontostomat*, (1). Recuperado desde https://scielo.conicyt.cl/pdf/ijodontos/v15n1/0718-381X-ijodontos-15-01-43.pdf

Mutis, M., Morón, E. & Medina, C. (2020). Teleodontología: Aplicación a la Odontopediatría durante la pandemia COVID-19. *Revista de Odontopediatría Latinoamericana*. Recuperado desde https://doi.org/10.47990/alop.v10i2.192

Pinela, M. (2018). *Diseño de una página web de gestián de citas médicas para el centro de salud TRINITARIA 2* (Tecnólogo en análisis de sistemas, Instituto Superior Tecnológico Bolivariano). Recuperado desde https://repositorio.itb.edu.ec/bitstream/123456789/2273/1/PROYECTO%5C%20DE%5C%20GRADO%5C%20DE%5C%20MIRANDA%5C%20PINELA.pdf

Pinto, C. (2013). *Implementación de una aplicación cliente servidor para el control de inventarios y facturación para la empresa American System* (Universidad Regional Autonoma de los Andes). Tesis de grado. Recuperado desde https://dspace.uniandes.edu.ec/handle/123456789/2321

Quintero, J. (2015). *Desarrollo de un portal WEB para el análisis tridimensional de estructuras aporticadas en concreto armado* (Tesis de maestría, Universidad de Los Andes). Tesis de maestría. Recuperado desde https://www.researchgate.net/publication/345773593

Rodríguez, S. (2015). *Diseño de prototipo de aplicación orientada a dispositivos móviles para administración de citas médicas por pacientes de entidades de salud odontológica* (Ingeniero de sistemas, Universidad Libre de Colombia). Recuperado desde https://repository.unilibre.edu.co/bitstream/handle/10901/10911/Proyecto%5C%20de%5C%20Grado.pdf?sequence=1%5C&isAllowed=y

Segovia, A. (2019). *Sistema de Gestión de Citas Médicas para un Centro de Salud* (Ingeniero en Informática de servicios y aplicaciones, Universidad de Los Andes).

Recuperado desde https://uvadoc.uva.es/bitstream/handle/10324/36492/TFG-B.1225.pdf?sequence=1%5C&isAllowed=y

Segura, G. & Atoche, S. (2021). Teleodontología en tiempos de la COVID-19. *Rev Científica Odontol*, (9). Recuperado desde https://doi.org/10.21142/2523-2754-0902-2021-062

Silberschatz, N., Korth, H. & Sudarshan. (2002). *Fundamentos de bases de datos* (Cuarta edición). McGRAW-HILL. Recuperado desde https://www.academia.edu/23593218/FUNDAMENTOS_DE_BASES_DE_DATOS_Cuarta_edici%C3%B3n?from=cover_page

Tello, A., Polo, A. & Tavera, N. (2019). Sistema de gestión y solicitud de citas médicas para estudiantes de las Unidades Tecnológicas de Santander. *III Congreso Internacional en inteligencia ambiental, ingeniería de software y salud electrónica y móvil*.

Treichler, C. (2021). The 10 best teledentistry companies of 2021. Recuperado desde https://www.onlinedoctor.com/best-teledentistry-companies/%5C#Virtudent

Valeri, L. (2016). Los sistemas de información para la gerencia en salud pública. *Visión Gerencial*, (9). Recuperado desde https://www.redalyc.org/journal/4655/465549558009/html/