**UNIVERSIDAD DE AQUINO BOLIVIA**

**FACULTAD DE CIENCIAS EXACTAS Y TECNOLOGÍA**

**CARRERA DE INGENIERIA DE SISTEMAS**



**SISTEMA WEB DE GESTIÓN DE HISTORIAS CLÍNICAS PARA LA CLÍNICA “FUNDACIÓN BARTIMEO”**

**ARQUITECTURA DE SOFTWARE**

**EXAMEN DE GRADO PARA OPTAR EL TÍTULO DE LICENCIATURA EN INGENIERÍA EN SISTEMAS**

**POSTULANTE:**  VERONICA LOPEZ TERRAZAS.

**SANTA CRUZ DE LA SIERRA – BOLIVIA**

**2024**

**INDICE GENERAL**

[RESUMEN EJECUTIVO 1](#_Toc179883814)

[ABSTRACT 2](#_Toc179883815)

[CAPITULO I.- GENERALIDADES 3](#_Toc179883816)

[1.1 INTRODUCCIÓN. 3](#_Toc179883817)

[1.2 ANTECEDENTES. 4](#_Toc179883818)

[1.3. PLANTEAMIENTO DEL PROBLEMA 5](#_Toc179883819)

[1.4 OBJETIVOS. 6](#_Toc179883820)

[1.4.1 OBJETIVOS GENERAL. 6](#_Toc179883821)

[1.4.2 OBJETIVO ESPECÍFICOS. 6](#_Toc179883822)

[1.5 DELIMITACIÓN DEL TEMA. 7](#_Toc179883823)

[1.5.1 DELIMITACIÓN ESPACIAL. 7](#_Toc179883824)

[1.5.2 DELIMITACIÓN TEMPORAL. 8](#_Toc179883825)

[3.6.3. DELIMITACIÓN SUSTANTIVA. 8](#_Toc179883826)

[1.6 JUSTIFICACIÓN 11](#_Toc179883827)

[1.6.1 JUSTIFICACIÓN PERSONAL. 11](#_Toc179883828)

[1.6.2 JUSTIFICACIÓN SOCIAL. 12](#_Toc179883829)

[1.6.3 JUSTIFICACIÓN PRÁCTICA. 12](#_Toc179883830)

[1.7 TECNICAS 13](#_Toc179883831)

[1.7.1 Técnicas de Recolección 13](#_Toc179883832)

[1.7.2. Técnicas de Verificación. 13](#_Toc179883833)

[1.8 HERRAMIENTAS 13](#_Toc179883834)

[1.8.1 Lenguaje de Programación y FrameWorks 13](#_Toc179883835)

[CAPITULO II 16](#_Toc179883836)

[MARCO TEÓRICO 16](#_Toc179883837)

[2.1. MARCO TEÓRICO CONCEPTUAL DEL PROYECTO. 17](#_Toc179883838)

[2.1.1 ARQUITECTURA DE SOFTWARE. 17](#_Toc179883839)

[2.1.2 Sistema de Información 19](#_Toc179883840)

[2.1.3 Sistema de Información Web 19](#_Toc179883841)

[2.1.4 HTTP 19](#_Toc179883842)

[2.1.5 HTTPS 19](#_Toc179883843)

[2.1.6 Protocolo 19](#_Toc179883844)

[2.1.7 Visual Studio Code 20](#_Toc179883845)

[2.1.8 Lenguaje de programación Laravel 21](#_Toc179883846)

[2.1.9 Entorno de desarrollo local Laragon 23](#_Toc179883847)

[2.2 DISEÑO ARQUITECTÓNICO 24](#_Toc179883848)

[2.2.1 Conceptos Clave En El Diseño Arquitectónico 24](#_Toc179883849)

[2.2.2 Patrones De Diseño Arquitectónico 24](#_Toc179883850)

[2.3 ORGANIZACIÓN DE LA INFORMACIÓN 25](#_Toc179883851)

[2.3.1 Gestión De Datos En La Arquitectura 25](#_Toc179883852)

[2.3.2 Modelado De Datos Y Bases De Datos 26](#_Toc179883853)

[2.3.3. Modos de organizar la información 26](#_Toc179883854)

[2..4 ESTILO ARQUITECTÓNICO 27](#_Toc179883855)

[2.4.1. Tipos de arquitecturas de software. 27](#_Toc179883856)

[2.4.1.1 Arquitectura de Cliente-Servidor 28](#_Toc179883857)

[2.4.1.2 Arquitectura en Capas 28](#_Toc179883858)

[2.4.1.3 Arquitectura Orientada a Servicios (SOA). 28](#_Toc179883859)

[2.4.1.4 Arquitectura Basada en Microservicios 28](#_Toc179883860)

[2.4.1.5 Arquitectura de Redes Neuronales 28](#_Toc179883861)

[2.4.1.6 Arquitectura Basada en Eventos 28](#_Toc179883862)

[2.3. MARCO TEÓRICO REFERENCIAL 29](#_Toc179883863)

[2.3.1 PENDIENTE A BUSCAR 29](#_Toc179883864)

[CAPITULO III DESARROLLO DE LA INVESTIGACION 30](#_Toc179883865)

[3.8 Desarrollo De La Solución (Scrum) 31](#_Toc179883866)

[3.8.1 Roles Y Stakeholders. 31](#_Toc179883867)

[3.8.2 Historias De Usuario. 31](#_Toc179883868)

[3.8.3 Product Backlog 48](#_Toc179883869)

[3.8.4 Estimación Del Product Backlog 49](#_Toc179883870)

[3.8.5. Primer Sprint 52](#_Toc179883871)

[3.8.5.1. Diagrama de Caso de Uso – Sprint 1 53](#_Toc179883872)

[3.8.5.2. Especificación de Caso de uso – Sprint 1 53](#_Toc179883873)

[3.8.5.2 Sprint Backlog Sprint 1 56](#_Toc179883874)

**INDICE DE FÍGURAS**

Figura 1 - Servicios de la Clínica 4

Figura 2. Ubicación Centro Salud Fundación Bartimeo 7

Figura 3. Planificación del Proyecto. 8

Figura 4. LIMITE SUSTANTIVO. 9

Figura 5 – Herramienta Visual Studio Code 20

Figura 6 – Lenguaje de programación Laravel 22

Figura 7 – Logotipo de Laragon 23

Figura 8.- Representación visual de 3 patrones de diseño 25

Figura 9 - Representación visual de arquitectura MVC 29

Figura 10. DIAGRAMA DE CASO DE USO – ACCEDER AL SISTEMA 32

Figura 11 DIAGRAMA DE CASO DE USO – GESTIONAR USUARIOS 33

Figura 12 DIAGRAMA DE CASO DE USO – GESTIONAR ROLES 34

Figura 13 DIAGRAMA DE CASO DE USO – GESTIONAR MÓDULOS 35

Figura 14 DIAGRAMA DE CASO DE USO – GESTIONAR PERMISOS 36

Figura 15. DIAGRAMA DE CASO DE USO – GESTIONAR ESPECIALIDAD 37

Figura 16. DIAGRAMA DE CASO DE USO – GESTIONAR PACIENTES 38

Figura 17. DIAGRAMA DE CASO DE USO – GESTIONAR AGENDAR CITAS MEDICAS 39

Figura 18. DIAGRAMA DE CASO DE USO – GESTIONAR PERSONAL MEDICO. 40

Figura 19. DIAGRAMA DE CASO DE USO – GESTIONAR TRATAMIENTOS. 41

Figura 20. DIAGRAMA DE CASO DE USO – GESTIONAR EXAMENES. 42

Figura 21. DIAGRAMA DE CASO DE USO – GESTIONAR EXAMENES. 43

Figura 22 DIAGRAMA DE CASO DE USO – GESTIONAR TPOS DE PAGOS. 44

Figura 23. DIAGRAMA DE CASO DE USO – GESTIONAR LABORATORIOS 45

Figura 24 DIAGRAMA DE CASO DE USO – GESTIONAR REPORTES DE CONTROL Y ESTADISTICOS 46

Figura 25 DIAGRAMA DE CASO DE USO – GENERAR REPORTES CONSULTAS REALIZADAS. 47

Figura 26 PLANNING POCKER GUÍA 49

Figura 27. DIAGRAMA DE CASO DE USO - SPRINT 1 53

Figura 28. DIAGRAMA DE BASE DE DATOS 57

**INDICE DE CUADROS**

Tabla 1.HISTORIA DE USUARIO - INGRESAR COMO ADMINISTRADOR 20

Tabla 2 HISTORIA DE USUARIO - GESTIONAR USUARIOS 21

TABLA 3 HISTORIA DE USUARIO - GESTIONAR ROLES 22

TABLA 4.HISTORIA DE USUARIO - GESTIONAR MÓDULOS 23

TABLA 5.HISTORIA DE USUARIO - GESTIONAR PERMISOS 24

TABLA 6. HISTORIA DE USUARIO - GESTIONAR ESPECIALIDADES. 25

TABLA 7. HISTORIA DE USUARIO - GESTIONAR PACIENTE. 27

TABLA 8. HISTORIA DE USUARIO - GESTIONAR AGENDAR CITAS MEDICAS 28

TABLA 96 . HISTORIA DE USUARIO - GESTIONAR PERSONAL MEDICO 29

TABLA 10. HISTORIA DE USUARIO - GESTIONAR TRATAMIENTOS 30

TABLA 117. HISTORIA DE USUARIO – GESTIONAR EXÁMENES. 31

TABLA 128. HISTORIA DE USUARIO - GESTIONAR HISTORIA CLINICA 32

TABLA 139 HISTORIA DE USUARIO - GESTIONAR TIPO DE PAGO 33

TABLA 14. HISTORIA DE USUARIO - GESTIONAR LABORATORIOS 34

Tabla 15. HISTORIA DE USUARIO - GENERAR REPORTE DE HISTORIAS CLÍNICAS, CONTROL Y SEGUIMIENTOS 35

TABLA 16. HISTORIA DE USUARIO - GENERAR REPORTE DE RECEPCION Y CONSULTAS REALIZADAS 36

TABLA 17. LISTA DE REQUISITOS PRIORIZADA 38

TABLA 18. ESTIMACIÓN DEL PRODUCT-BACKLOG SPRINT 1 39

TABLA 1910. ESTIMACIÓN DEL PRODUCT-BACKLOG SPRINT 2 40

TABLA 20. ESTIMACIÓN DEL PRODUCT-BACKLOG SPRINT 3 41

TABLA 21. SPRINT PLANNING - SPRINT 1 41

TABLA 22. ESPECIFICACIÓN DE CASO DE USO HU1 43

TABLA 23. ESPECIFICACIÓN DE CASO DE USO HU2 43

TABLA 2411. ESPECIFICACIÓN DE CASO DE USO HU3 44

TABLA 25. ESPECIFICACIÓN DE CASO DE USO HU4 44

TABLA 26. ESPECIFICACIÓN DE CASO DE USO HU5 45

TABLA 2712. SPRINT BACKLOG SPRINT 1 45

Introducción a UML: Lenguaje para modelar objetos. (2017). (n.p.): Natsys. 47

Subra, J., Vannieuwenhuyse, A. (2018). Scrum: un método ágil para sus proyectos. España: Ediciones Eni. 47

Dimes, T. (2015). Conceptos Básicos De Scrum: Desarrollo De Software Agile Y Manejo De Proyectos Agile. Estados Unidos: Babelcube Incorporated. 47

Sánchez, C. (24 de enero de 2020). *Bibliografía*. Normas APA (7ma edición). https://normas-apa.org/bibliografia/ 47

Arias, Á., Durango, A. (2016). Ingeniería y Arquitectura Del Software: 2a Edición. (n.p.): CreateSpace Independent Publishing Platform. 47

FRAMEWORK TOTAL - Vol.2: Crea APPs desde Cero con Laravel + W3.CSS + MySQL. (2021). (n.p.): RedUsers. 47

Baker, C. (2023). Aprende Vue.js: Guía para desarrolladores web. (n.p.): Amazon Digital Services LLC - Kdp. 47

**DEDICATORIA**

El Presente Proyecto de Grado está dedicado principalmente a Dios quien me supo guiar dándome fuerzas, sabiduría y conocimientos para poder realizar este proyecto.

A mi Padre y a mi Madre que hoy en vida no están, pero desde el cielo están siempre conmigo y en mi corazón, a mis hermanos y hermana, los cuales son mi roca inquebrantable, a mi comadre y a todos mis parientes les agradezco por su apoyo incondicional, su comprensión y por ser mi fuente de inspiración.

A mis amistades y colegas quienes han estado a mi lado en cada paso del camino, no tengo palabras suficientes para expresar mi gratitud. Su amistad sincera y respaldo incondicional me han dado fuerzas cuando más lo necesitaba. Siempre he encontrado en ustedes un hombro en el que apoyarme, una sonrisa que ilumina mis días y una dosis de alegría que me impulsa a seguir adelante.

Hoy celebro no solo mi éxito, sino también el nuestro y de cada uno de ustedes como docente que ha dejado en mí, una huella imborrable en mi vida y en este proyecto. Gracias por creer en mí, por animarme a seguir adelante y por ser mis docentes, ser mis compañeros y mis concejeros de vida.

Hoy, al llegar al final de este arduo pero gratificante camino que ha sido mi proyecto de grado, quiero expresar mi más profundo agradecimiento a cada uno de ustedes. Su constante apoyo, aliento y experiencias que han sido pilares fundamentales en este logro que hoy culmino.

¡A mi familia y amigos y a todos mis docentes, mi eterno agradecimiento!

**AGRADECIMIENTO**

Quiero aprovechar este espacio para expresar mi más sincero agradecimiento a todas las personas que contribuyeron de manera significativa en la realización de este proyecto de grado.

En primer lugar, quiero expresar mi profunda gratitud a mi Tutor el Ing. Carlos Andrés Álvarez por su guía experta y al Ing. Iver Guerrero Ortiz por su apoyo constante y su valiosa orientación a lo largo de todo el proceso. Gracias por compartir su sabiduría y experiencia, y por brindarme la oportunidad de crecer y aprender como estudiante e investigador.

Agradezco enormemente al Ing. Carlos Edwin Balcázar por sus valiosas sugerencias y comentarios constructivos que contribuyeron en gran medida al enriquecimiento de este trabajo. Su dedicación y compromiso académico fueron una fuente de inspiración constante.

Un agradecimiento especial va dirigido a mis docentes, amigos, colegas y seres queridos, quienes me brindaron su incondicional apoyo emocional a lo largo de este desafío. Sus palabras de aliento, gestos de cariño y comprensión fueron un verdadero apoyo para superar momentos difíciles y mantenerme enfocado en mi objetivo.

Gracias a la Microempresa “BODEGA ARTESANAL TARIJEÑITO”, por brindarme la información necesaria para la realización del proyecto de grado.

A cada una de estas personas, mi más profundo agradecimiento. Su contribución ha sido invaluable y ha dejado una marca perdurable en mi vida académica y personal.

Sin ustedes, este proyecto de grado no habría sido más que una idea. Gracias por ser parte de mi camino y por hacerlo significativo. Con gratitud.

# RESUMEN EJECUTIVO

El proyecto consiste en Desarrollar un sistema de gestión de Historias Clínicas para la Clínica “Fundación Bartimeo” en Km. 9 El Carmen, La Guardia, Bolivia, Santa Cruz de la Sierra. El objetivo principal es mejorar la eficiencia en la gestión de registros, control de pacientes y diagnósticos. El sistema permitirá la programación de citas médicas, gestión de datos y horarios de manera eficiente a través de una plataforma centralizada.

El proyecto se divide en cuatro etapas fundamentales:

Introducción es el proceso del trabajo de investigación donde se examina la relevancia y situación actual de las consultas médicas en Santa Cruz, enfocándose especialmente en la ciudad de La Guardia Se identifica la necesidad de implementar un sistema de gestión para mejorar los procesos operativos y administrativos de la clínica " Fundación Bartimeo".

Estudio teórico: Se proporciona una exposición detallada de conceptos relacionados con tecnologías informáticas, incluyendo componentes, tendencias actuales y herramientas de desarrollo empleadas en el ámbito tecnológico.

Definición del proyecto: Se inicia con la introducción y antecedentes, seguido por la definición del problema u oportunidad. Se establecen objetivos generales y específicos, así como delimitaciones espaciales, temporales y sustantivas. Se describe la metodología utilizada, siendo Scrum, y se detallan historias de usuario, el Product Backlog y los sprints.

Conclusiones y recomendaciones: Se presentan las conclusiones obtenidas durante el desarrollo del proyecto, resaltando la optimización de los procesos de registros y control del centro de salud. Se ofrecen recomendaciones para mejorar el uso del sistema, garantizando su seguridad, confidencialidad y disponibilidad, con el fin de mejorar aún más los procesos de trabajo en el centro de salud " Fundación Bartimeo "

# ABSTRACT

The project consists of developing a medical record management system for the “Fundación Bartimeo” Clinic at Km. 9 El Carmen, La Guardia, Bolivia, Santa Cruz de la Sierra. The main objective is to improve efficiency in record management, patient control and diagnoses. The system will allow the scheduling of medical appointments, data management and schedules efficiently through a centralized platform.

The project is divided into four fundamental stages:

Context analysis: The relevance and current situation of medical consultations in Santa Cruz is examined, focusing especially on the city of La Guardia The need to implement a management system is identified to improve the operational and administrative processes of the clinic "Bartimeo Foundation ".

Theoretical study: A detailed exposition of concepts related to computer technologies is provided, including components, current trends and development tools used in the technological field.

Project definition: It begins with the introduction and background, followed by the definition of the problem or opportunity. General and specific objectives are established, as well as spatial, temporal and substantive delimitations. The methodology used is described, being Scrum, and user stories, the Product Backlog and the sprints are detailed.

Conclusions and recommendations: The conclusions obtained during the development of the project are presented, highlighting the optimization of the registration and control processes of the health center. Recommendations are offered to improve the use of the system, guaranteeing its security, confidentiality and availability, in order to further improve the work processes at the "Fundación Bartimeo" health center.

# CAPITULO I.- GENERALIDADES

A continuación, se presentará el primer capítulo del proyecto con la introducción.

# 1.1 INTRODUCCIÓN.

La clínica “Fundación Bartimeo” es una fundación sin fines de lucro que se dedica a la ayuda de sectores vulnerables con la dotación de manera gratuita de placas dentales y lentes. tradicional en un barrio tranquilo de la ciudad, se ofrece una atención médica personalizada con un enfoque meticuloso en la documentación de cada proceso a través del uso de papel y bolígrafo. El equipo médico, compuesto por profesionales con años de experiencia, se esfuerza por brindar un servicio de calidad a cada paciente que cruza por la puerta.

Cuyo dueño es el doctor Mario Salek fundada en el año 01 de septiembre del año 2009, ubicada en Km. 9 doble vía la Guardia El Carmen, Santa Cruz de la Sierra, Bolivia, se destaca por su compromiso en proporcionar servicios médicos de calidad a la comunidad local. Desde su establecimiento, la fundación ha abrazado una visión de atención integral, siempre buscando mejorar sus procesos para asegurar un servicio óptimo a sus pacientes de las cuales brinda son los siguientes: medicina general, odontología, oftalmología, cardiología, ginecología, pediatra, traumatología, internación (laboratorio análisis clínico, ecografía, electrocardiograma, monitoreo fetal, rayos X).

Ante el incremento constante de la demanda y la necesidad imperante de optimizar la gestión de las consultas médicas, se ha propuesto el desarrollo de un Sistema de Gestión para consultas médicas en la Fundación Bartimeo. Este sistema se erige con el propósito de mejorar la eficiencia en la administración de registros, control de pacientes y diagnósticos, en aras de contribuir a una atención médica más efectiva y satisfactoria.



Figura 1 - Servicios de la Clínica

Fuente: Clínica de Salud Fundación Bartimeo

# 1.2 ANTECEDENTES.

La clínica “Fundación Bartimeo” Cuyo dueño es el doctor Mario Salek fundada en el año 01 de septiembre del año 2009, ubicada en Km. 9 doble vía la Guardia El Carmen, Santa Cruz de la Sierra, Bolivia, se destaca por su compromiso en proporcionar servicios médicos de calidad a la comunidad local. Desde su establecimiento, la fundación ha abrazado una visión de atención integral, siempre buscando mejorar sus procesos para asegurar un servicio óptimo a sus pacientes de las cuales brinda son los siguientes: medicina general, odontología, oftalmología, cardiología, ginecología, pediatra, traumatología, internación (laboratorio análisis clínico, ecografía, electrocardiograma, monitoreo fetal, rayos X).

Desde el momento en que un paciente llega a la clínica, se le proporciona un formulario en papel para completar con su información personal y médica. Este formulario se convierte en el primer paso para la creación de su historial clínico, que será meticulosamente registrado a mano por el personal administrativo.

Durante la consulta con el médico, cada síntoma, diagnóstico y plan de tratamiento se anotan cuidadosamente en la historia clínica del paciente, asegurándose de capturar todos los detalles relevantes para su atención médica. Los exámenes de laboratorio y los resultados de las pruebas también se registran en el expediente del paciente, proporcionando un registro completo de su historial de salud.

Aunque el proceso de documentación se realiza de manera manual, el equipo médico se esfuerza por mantener una precisión y organización impecables en cada registro. Además, se toman medidas para garantizar la confidencialidad de la información del paciente, almacenando los expedientes en un lugar seguro y restringiendo el acceso solo al personal autorizado.

# 1.3. PLANTEAMIENTO DEL PROBLEMA

En respuesta a la creciente demanda y la necesidad de optimizar la gestión de historias clínicas, se ha propuesto el desarrollo de un Sistema de Gestión para historias clínicas en la Fundación Bartimeo. Este sistema tiene como objetivo mejorar la eficiencia en la gestión de registros, control de pacientes y diagnósticos, contribuyendo así a una atención médica más efectiva y satisfactoria.

El proyecto se basa en la implementación de una plataforma centralizada que permitirá la programación de citas médicas, la gestión de datos y horarios de manera eficiente. Con el uso de tecnología avanzada, se busca modernizar los procesos administrativos y operativos del centro de salud, brindando una experiencia más fluida tanto para los pacientes como para el personal médico.

En el trasfondo de estas circunstancias, surgen ciertos problemas cruciales y altamente complejos en la trama de los procesos involucrados.

En primer lugar, cabe destacar el proceso de captura de información, el cual se lleva a cabo de forma manual, lo que propicia la redacción de datos en un lenguaje posiblemente ilegible o poco claro. Esta situación dificulta la comprensión y accesibilidad de la información, y por ende, su potencial utilización efectiva en la toma de decisiones clínicas.

Otro desafío sustancial radica en la existencia de múltiples expedientes clínicos vinculados a un solo paciente. Este fenómeno conlleva al crecimiento constante y rápido de documentos en formato físico, ocupando valioso espacio en las instalaciones del centro de salud. Esta acumulación no solo resulta ineficiente en términos de espacio, sino que también puede provocar dificultades en la organización y recuperación de la información, lo que aumenta los riesgos de errores y retrasos en la atención médica.

Además, se presenta la problemática adicional de no contar con un registro preciso de los costos de comisiones de cada servicio que brinda la clínica. Esta falta de información detallada puede afectar la gestión financiera y la toma de decisiones, ya que dificulta la evaluación precisa del rendimiento económico de cada servicio y la planificación estratégica para maximizar la rentabilidad y eficiencia del centro de salud.

Por último, se destaca la lentitud en la recuperación y transferencia de la información específica de un paciente. Los procedimientos manuales actuales resultan en una respuesta demorada en la obtención de los datos necesarios, lo cual puede impactar negativamente en la calidad y eficacia de la atención médica brindada, así como en la experiencia del paciente.

# 1.4 OBJETIVOS.

### 1.4.1 OBJETIVOS GENERAL.

Desarrollar un Sistema de información para la gestión de Historias Clínicas en la clínica “Fundación Bartimeo”

## 1.4.2 OBJETIVO ESPECÍFICOS.

En concordancia con el objetivo general, se presentan los siguientes objetivos específicos:

* Elaborar una lista de requisitos, basados en observación y entrevistas a los funcionarios de la Clínica “Fundación Bartimeo”.
* Determinar la prioridad de cada requisito en la Lista de Producto (Product Backlog).
* Establecer los Ciclos de trabajo (Sprints), para organizar l­as etapas del desarrollo del sistema.
* Realizar las revisiones de cada ciclo de trabajo (Sprint Review), para registrar las tareas completadas.
* Ejecutar pruebas de funcionamiento del sistema

# ­­1.5 DELIMITACIÓN DEL TEMA.

## 1.5.1 DELIMITACIÓN ESPACIAL.

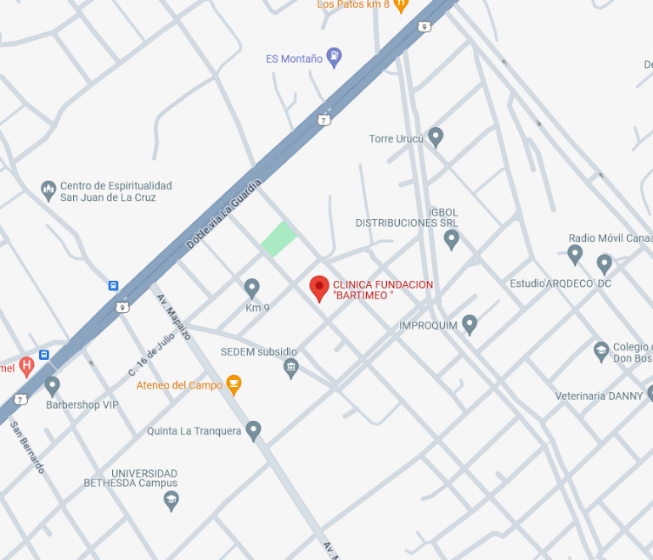
El presente sistema de información de gestión propuesto está dirigido para la Clínica Fundación Bartimeo. Que se encuentra ubicada en Km. 9 El Carmen, La Guardia - de Santa cruz de la Sierra – Bolivia.

Figura 2. Ubicación Centro Salud Fundación Bartimeo

Fuente: Google Maps.

## 1.5.2 DELIMITACIÓN TEMPORAL.

Gráfico

Descripción generada automáticamenteMediante el siguiente diagrama de Gantt podemos observar las tareas y actividades planificadas para este Proyecto, junto con las estimaciones de tiempo para cada una.

Figura 3**. Planificación del Proyecto**.

Fuente: Elaboración Propia

## 3.6.3. DELIMITACIÓN SUSTANTIVA.

El alcance del sistema tendrá como límite los siguientes módulos:

Figura 4. LIMITE SUSTANTIVO.

Fuente: Elaboración propia.

**Módulo Gestión de consultas:** permite administrar de manera eficiente y organizada la información relacionada con los pacientes y los servicios médicos que reciben.

* **Gestionar Citas médicas:** Permite programar, modificar y cancelar citas médicas para los pacientes, así como gestionar la disponibilidad de los médicos y las salas de consulta.
* **Gestionar Tratamientos y recetas:** Facilita la creación, registro y seguimiento de tratamientos médicos para cada paciente, así como la generación y gestión de recetas médicas.
* **Gestionar Historias médicas:** Ofrece un sistema para almacenar y gestionar de forma segura el historial médico de cada paciente, incluyendo diagnósticos, tratamientos previos, alergias, y otros datos relevantes.

**Módulo Parámetros y Configuración:** Este módulo permitirá al usuario gestionar los parámetros del sistema para que los demás módulos puedan realizar sus funciones.

* **Gestionar Pacientes:** Permitirá gestionar los pacientes que tenga la clínica, tener un control de cuantos pacientes se han atendido hasta la actualidad.
* **Gestionar Médicos:** Permite administrar la información del personal médico, incluyendo datos personales, especialidades, horarios de consulta y asignación de pacientes.
* **Gestionar Servicios:** Ofrece la capacidad de administrar diferentes servicios médicos ofrecidos por la institución, como historias clínicas generales, cirugías, exámenes de diagnóstico.

**Módulo Reportes:** Este módulo permite al usuario coordinar la información de forma inmediata entre las diferentes áreas de la empresa, además de que permitirá generar informes inmediatos.

* **Generar Reportes de historias clínicas:** Estos informes proporcionan un resumen detallado del historial médico de un paciente, incluyendo diagnósticos, tratamientos, medicamentos recetados, resultados de pruebas de laboratorio, y otros datos relevantes. Pueden ser útiles para realizar evaluaciones de salud a largo plazo, seguimiento de tratamientos, o para compartir información con otros profesionales de la salud.
* **Generar reportes por servicios:** Permite generar informes por fechas, tipos de servicios de cuantas atenciones se realizaron.
* **Generar reportes de citas médicas:** Este informe muestra la cantidad y tipo de servicios médicos proporcionados por cada médico durante un período de tiempo determinado. Puede incluir información como el número de historias clínicas realizadas, procedimientos médicos realizados, otros datos relacionados con la productividad y la eficiencia del personal médico. Este tipo de informe puede ser útil para evaluar el desempeño individual de los médicos, identificar áreas de mejora, y asignar recursos de manera más efectiva.

**Modulo administración de usuario, seguridad y auditoria**

En este módulo permite al usuario gestionar la seguridad del sistema definiendo parámetros, permisos de usuarios y perfiles. También permitirá realizar copias de seguridad y permitirá la auditoria del mismo a través de bitácoras del sistema.

* **Gestionar Usuario:** Permite gestionar los usuarios que tendrán acceso al sistema y la asignación de perfiles de usuario a los mismo.
* **Gestionar funciones del Sistema:** Permite gestionar todas las funciones del sistema para su respectivo control.
* **Asignar permisos a Usuarios**: Permite gestionar los perfiles de usuario y la asignación de permisos de acceso.

**Administración de Bitácoras de Operaciones por usuario**: Permite ver el registro de actividades del usuario de las distintas operaciones que realizan en el Sistema.

# 1.6 JUSTIFICACIÓN

## 1.6.1 JUSTIFICACIÓN PERSONAL.

Mi interés y entusiasmo por este tema se han fortalecido con el tiempo a medida que he profundizado en mis estudios, me brinda la oportunidad de aplicar y expandir el conocimiento que he adquirido a lo largo de mi formación. Además, este proyecto es de suma importancia para mi formación profesional, ya que me permite desarrollar habilidades y competencias esenciales que serán valiosas en mi futura carrera. A través de la investigación, el análisis y la resolución de problemas relacionados con este tema, estoy fortaleciendo mi capacidad de investigar de manera rigurosa, analizar datos de manera crítica y comunicar mis hallazgos de manera efectiva. Estas habilidades son fundamentales en el campo en el que planeo desarrollar mi carrera, y este proyecto me brinda la oportunidad de perfeccionarlas.

## 1.6.2 JUSTIFICACIÓN SOCIAL.

El propósito al realizar esta investigación es con la finalidad de conseguir modernizar el proceso de citas médicas, facilitándoles nuevas líneas de comunicación, permitiendo a los pacientes obtener una consulta de una manera más rápida, al centro de salud, lo que generará comodidad, satisfacción y además se logrará brindar el mejor servicio a todos los pacientes del Centro de salud Fundación Bartimeo es centro médico que se dedica a la prestación de servicios de historias clínicas se ha podido determinar la necesidad de un sistema de información para este tipo de centros médicos o rubros en específico por lo tanto este sería un aporte de mi persona para este tipo de emprendimientos buscando que este sea un punto de partida para la realización de futuros proyectos de este mismo rubro que ayuden a mejorar su organización y el manejo de datos.

## 1.6.3 JUSTIFICACIÓN PRÁCTICA.

El presente proyecto de investigación es de gran relevancia, ya que tiene como propósito fundamental la aplicación práctica de los conocimientos adquiridos en función de los objetivos establecidos. Busca abordar problemas críticos relacionados con las tareas manuales, el tiempo prolongado de atención y los errores en los registros y búsquedas de historias clínicas, aspectos que han sido objeto de quejas en la atención médica.

Por tanto, resulta imperativo proponer un sistema que mejore la gestión de citas médicas y, al mismo tiempo, sea adaptable a los procedimientos del centro médico. Este sistema tiene como meta principal organizar de manera eficiente los procesos de citas médicas, con el fin de optimizar el tiempo de atención y reducir los errores asociados con la gestión manual de registros médicos.

La importancia de este proyecto radica en su capacidad para no solo solucionar problemas específicos en la atención médica, sino también en su potencial para establecer estándares más altos en la calidad del servicio ofrecido. Al proponer un sistema flexible y adaptado a las necesidades del centro médico, se espera mejorar significativamente la eficiencia operativa y la experiencia del paciente.

## 1.7 TECNICAS

## 1.7.1 Técnicas de Recolección

La recolección de información es esencial para la validez y confiabilidad de una investigación. Entre las técnicas destacan:

**Observación**: Método fundamental que permite captar el comportamiento de grupos. Puede ser estructurada, abierta, semiestructurada o participante, cada una con diferentes grados de control y flexibilidad.

**Análisis** **de** **Documentos**: Implica la revisión de documentos para extraer información histórica, económica y social. Los documentos pueden contener datos cuantitativos y cualitativos que ayudan a entender la organización.

## 1.7.2. Técnicas de Verificación.

Para asegurar el correcto funcionamiento de un sistema, se aplican diversas pruebas:

**Pruebas Unitarias**: Revisión de componentes individuales.

**Pruebas de Integración**: Validación de interacciones entre módulos.

**Pruebas de Sistema y Aceptación**: Verificación general del sistema y aceptación por parte del usuario.

## 1.8 HERRAMIENTAS

## 1.8.1 Lenguaje de Programación y FrameWorks

**PHP**: Es un lenguaje de programación del lado del servidor muy utilizado para el desarrollo de aplicaciones web. Su sintaxis es fácil de aprender, lo que lo hace accesible para principiantes. PHP permite la creación de aplicaciones dinámicas que pueden interactuar con bases de datos.

**Snappy**: Es una biblioteca de PHP que permite la generación de archivos PDF a partir de contenido HTML. Se basa en el motor de renderizado wkhtmltopdf, que convierte HTML y CSS en documentos PDF de alta calidad. Snappy es especialmente útil para la creación de reportes, facturas, y cualquier otro tipo de documento que requiera un formato visual atractivo y bien estructurado.

**Framework PHP (Laravel)**: Laravel es un framework PHP que sigue el patrón de diseño MVC (Modelo-Vista-Controlador). Proporciona herramientas para la creación de aplicaciones web robustas y seguras, incluyendo enrutamiento, middleware, autenticación y migraciones de bases de datos, lo que acelera el desarrollo y mejora la organización del código.

**JavaScript:** Es un lenguaje de programación esencial para el desarrollo web del lado del cliente. Permite añadir interactividad y dinamismo a las páginas web.

**Framework JavaScript (Vue.js):** Es un framework progresivo que permite construir interfaces de usuario de manera sencilla y flexible. Facilita la creación de componentes reutilizables y la gestión del estado de la aplicación, lo que mejora la experiencia del usuario al hacer que las interfaces sean más reactivas y eficientes.

**1.8.2 Servidor Apache**

Apache es un servidor web de código abierto que es ampliamente utilizado para alojar aplicaciones web. Su configuración es altamente personalizable, y es conocido por su estabilidad y capacidad de manejar múltiples solicitudes simultáneamente. Apache también es compatible con varios módulos que pueden extender su funcionalidad.

**1.8.3 Base de Datos MySQL**

MySQL es un sistema de gestión de bases de datos relacional de código abierto, utilizado para almacenar y gestionar datos en aplicaciones web. Es conocido por su rendimiento, escalabilidad y robustez. MySQL utiliza SQL (Structured Query Language) para realizar operaciones de consulta y manipulación de datos, lo que facilita la interacción con la base de datos desde las aplicaciones PHP.

**1.8.4 Herramientas de Desarrollo de Software**

**Visual Studio Code**

Visual Studio Code (VS Code) es un editor de código fuente ligero y poderoso que es popular entre los desarrolladores. Ofrece soporte para múltiples lenguajes de programación, autocompletado, depuración integrada y una amplia gama de extensiones que mejoran la productividad.

**Laragon**

Laragon es un entorno de desarrollo local para PHP que facilita la creación y gestión de aplicaciones web. Proporciona un entorno ligero y portátil que incluye servidores web, bases de datos y herramientas de línea de comandos. Laragon es fácil de usar y permite configurar proyectos de manera rápida y eficiente.

# CAPITULO II

# MARCO TEÓRICO

A continuación, se presentará el marco teórico del proyecto en sus apartados: Marco Teórico Conceptual Del Proyecto, Marco Teórico Conceptual De La Metodología, Marco Teórico Referencial.

## 2.1. MARCO TEÓRICO CONCEPTUAL DEL PROYECTO.

## 2.1.1 ARQUITECTURA DE SOFTWARE.

La arquitectura de software se refiere al diseño estructural de un sistema de software y a la manera en que interactúan sus componentes. Es una disciplina clave en el desarrollo de software, ya que proporciona un marco para la construcción y evolución de sistemas de software, asegurando que cumplan con los requisitos funcionales y no funcionales.

La arquitectura de software aborda decisiones importantes como la organización de los componentes del sistema, la asignación de responsabilidades, la gestión de la información y la definición de interfaces. Algunos conceptos clave incluyen:

* **Componentes:** Los bloques de construcción del sistema, que pueden ser módulos, clases, funciones, etc.
* **Conectores:** Definen cómo los componentes interactúan entre sí. Pueden ser protocolos de comunicación, llamadas a funciones, eventos, etc.

1. Patrones arquitectónicos: Soluciones probadas para problemas comunes de diseño. Algunos ejemplos incluyen la arquitectura en capas, la arquitectura cliente-servidor, la arquitectura orientada a microservicios, etc.
2. Estilos arquitectónicos: Definen cómo los componentes y los conectores son utilizados para lograr objetivos específicos. Ejemplos incluyen la arquitectura basada en eventos, la arquitectura basada en servicios, etc.
3. Vistas arquitectónicas: Representaciones específicas de la arquitectura para abordar preocupaciones particulares. Por ejemplo, una vista puede centrarse en la estructura estática del sistema, mientras que otra se enfoca en la dinámica de su comportamiento.

La arquitectura de software ayuda a los desarrolladores a entender la estructura del sistema, facilita la comunicación entre los miembros del equipo, y proporciona una base para la toma de decisiones durante todo el ciclo de vida del desarrollo del software. Un buen diseño arquitectónico puede contribuir a la flexibilidad, mantenibilidad, escalabilidad y rendimiento del sistema.

La arquitectura de software está compuesta por diversos componentes, cada uno de los cuales desempeña un papel específico en la construcción y funcionamiento del sistema. Comprender estos componentes es esencial para diseñar y desarrollar una arquitectura efectiva. A continuación, se describen algunos de los componentes clave:

* **Componentes de Software:** Estos son los bloques de construcción del sistema.Incluyen módulos, bibliotecas, clases y otros elementos que encapsulan la funcionalidad del software.
* **Interfaz de Usuario (UI):** La interfaz de usuario es la parte del sistema con la queinteractúan los usuarios. La forma en que se diseña y se presenta la interfaz de usuario es un componente crucial para la usabilidad y la experiencia del usuario.
* **Base de Datos:** La base de datos es donde se almacenan los datos del sistema. Puedeser una base de datos relacional, NoSQL u otro tipo de almacenamiento de datos.
* **Componentes Externos:** Los sistemas de software pueden depender de componentes externos, como bibliotecas de terceros, servicios en la nube o API de terceros. Estos componentes deben ser considerados en la arquitectura, ya que pueden influir en la elección de tecnologías y en la integración con el sistema.
* **Seguridad:** La seguridad es un componente crítico en la arquitectura. Incluye laimplementación de medidas de seguridad, como autenticación, autorización, cifrado y protección contra amenazas cibernéticas.
* **Escalabilidad y Rendimiento:** Esto implica considerar cómo se pueden distribuir lascargas de trabajo, optimizar el rendimiento y garantizar que el sistema pueda manejar un aumento en la demanda.
* **Gestión de Errores y Monitoreo**: La arquitectura debe incluir estrategias paramanejar errores y excepciones, así como para monitorear el sistema en busca de problemas y métricas de rendimiento.

Cada uno de estos componentes desempeña un papel importante en la arquitectura global de un sistema de software. La elección y la integración adecuada de estos componentes son fundamentales para el éxito del proyecto.

## 2.1.2 Sistema de Información

Un sistema de información es un conjunto organizado de elementos que recopilan, almacenan, procesan y distribuyen datos. Su propósito es apoyar la toma de decisiones, coordinar actividades y mejorar la eficiencia operativa. Incluye hardware, software, datos, procedimientos y personas, y se utiliza en diversas áreas como la gestión empresarial, la educación y la salud.

## 2.1.3 Sistema de Información Web

Un sistema de información web es una plataforma que permite el acceso y la gestión de datos a través de internet. Facilita la interacción en tiempo real y la colaboración entre usuarios. Estos sistemas utilizan tecnologías web para ofrecer interfaces intuitivas, accesibles desde cualquier dispositivo con conexión a internet, permitiendo la actualización y consulta de información de manera dinámica.

## 2.1.4 HTTP

HTTP (Hypertext Transfer Protocol) es un protocolo de comunicación utilizado para la transferencia de información en la web. Permite que los navegadores soliciten recursos, como páginas web y archivos, a los servidores. Su funcionamiento se basa en un modelo de petición-respuesta, donde el cliente envía una solicitud y el servidor responde con el contenido solicitado. Es fundamental para la navegación en internet.

## 2.1.5 HTTPS

HTTPS (Hypertext Transfer Protocol Secure) es una extensión de HTTP que añade una capa de seguridad mediante la criptografía. Utiliza SSL/TLS para cifrar la comunicación entre el navegador y el servidor, protegiendo la privacidad y la integridad de los datos. Es esencial para transacciones seguras en línea, como compras y transferencia de información sensible, mejorando la confianza del usuario.

## 2.1.6 Protocolo

Un protocolo es un conjunto de reglas y convenciones que definen cómo se comunican y transfieren datos entre dispositivos en una red. Esencial en la informática y las telecomunicaciones, los protocolos aseguran la interoperabilidad y la comprensión mutua entre sistemas, facilitando la conexión, el intercambio de información y la seguridad en la comunicación.

## 2.1.7 Visual Studio Code

Visual Studio Code es un editor de código fuente desarrollado por Microsoft que se ha vuelto extremadamente popular entre desarrolladores de software debido a su interfaz de usuario amigable, su amplia gama de extensiones y su capacidad de personalización. Aquí hay una descripción detallada de algunas de sus características principales:

* **Interfaz de usuario:** Visual Studio Code presenta una interfaz limpia y moderna que es altamente personalizable. La disposición de las ventanas y paneles se puede ajustar según las preferencias del usuario.
* **Editor de texto avanzado:** Ofrece resaltado de sintaxis para una amplia variedad de lenguajes de programación, así como funciones como autocompletado de código, sugerencias de código, refactorización y navegación rápida entre archivos y líneas de código.
* **Depuración integrada:** Visual Studio Code proporciona herramientas de depuración integradas para varios lenguajes de programación. Los desarrolladores pueden establecer puntos de interrupción, inspeccionar variables y controlar la ejecución del código paso a paso.
* **Control de versiones:** Integra herramientas de control de versiones como Git de manera nativa, lo que permite a los desarrolladores realizar operaciones de control de versiones directamente desde el editor.

****

Figura 5 – Herramienta Visual Studio CodeFuente: www.visualstudio.com

* **Extensibilidad:** Una de las características más potentes de Visual Studio Code es su ecosistema de extensiones. Los usuarios pueden instalar extensiones para agregar nuevas características y funcionalidades, como soporte para nuevos lenguajes de programación, temas de color, herramientas de productividad y mucho más.
* **Integración con la nube:** Visual Studio Code se integra con varios servicios en la nube, como Azure de Microsoft, facilitando el desarrollo y la implementación de aplicaciones en la nube.
* **Terminal integrada:** Incluye una terminal integrada que permite a los desarrolladores ejecutar comandos directamente desde el editor, lo que facilita la ejecución de scripts, la instalación de dependencias y otras tareas de desarrollo.

En resumen, Visual Studio Code es una herramienta poderosa y versátil que proporciona a los desarrolladores un entorno de desarrollo moderno y eficiente para escribir, depurar y colaborar en proyectos de software.

## 2.1.8 Lenguaje de programación Laravel

es un popular framework de desarrollo de aplicaciones web con PHP. Aquí tienes una descripción detallada:

* **Framework PHP:** Laravel es un framework de código abierto basado en PHP que sigue el patrón de arquitectura de software Modelo-Vista-Controlador (MVC). Está diseñado para facilitar y acelerar el desarrollo de aplicaciones web modernas y escalables.
* **Sintaxis elegante y expresiva:** Una de las características distintivas de Laravel es su sintaxis elegante y expresiva, que permite a los desarrolladores escribir código limpio y legible. Utiliza características como rutas declarativas, migraciones de base de datos y Eloquent ORM para simplificar tareas comunes de desarrollo.
* **MVC y separación de preocupaciones:** Laravel promueve el desarrollo siguiendo el principio de separación de preocupaciones, lo que significa que divide la lógica de la aplicación en modelos, vistas y controladores. Esto facilita la organización y el mantenimiento del código.
* **Eloquent ORM:** Laravel incluye Eloquent, un ORM (Mapeador Objeto-Relacional) que simplifica la interacción con la base de datos. Permite a los desarrolladores trabajar con datos de manera intuitiva utilizando objetos PHP en lugar de escribir consultas SQL directamente.
* **Sistema de enrutamiento:** Laravel proporciona un sistema de enrutamiento flexible que permite definir rutas de manera clara y concisa. Esto facilita la creación de URL amigables y el manejo de solicitudes HTTP.
* **Migraciones de base de datos:** Con Laravel, las migraciones de base de datos se utilizan para versionar y administrar los cambios en la estructura de la base de datos. Esto permite a los desarrolladores trabajar de manera colaborativa y mantener un historial de cambios en la estructura de la base de datos.
* **Seguridad integrada:** Laravel incluye características de seguridad integradas, como protección contra ataques CSRF (Cross-Site Request Forgery), protección XSS (Cross-Site Scripting) y hashing de contraseñas. También proporciona herramientas para la autenticación de usuarios y la autorización de roles y permisos.
* **Ecosistema robusto:** Además del framework principal, Laravel cuenta con un ecosistema robusto de paquetes y extensiones que amplían su funcionalidad. Estos paquetes pueden ser instalados fácilmente a través de Composer, el administrador de dependencias de PHP.

Figura 6 – Lenguaje de programación Laravel

Fuente: Elaboración Propia

## 2.1.9 Entorno de desarrollo local Laragon

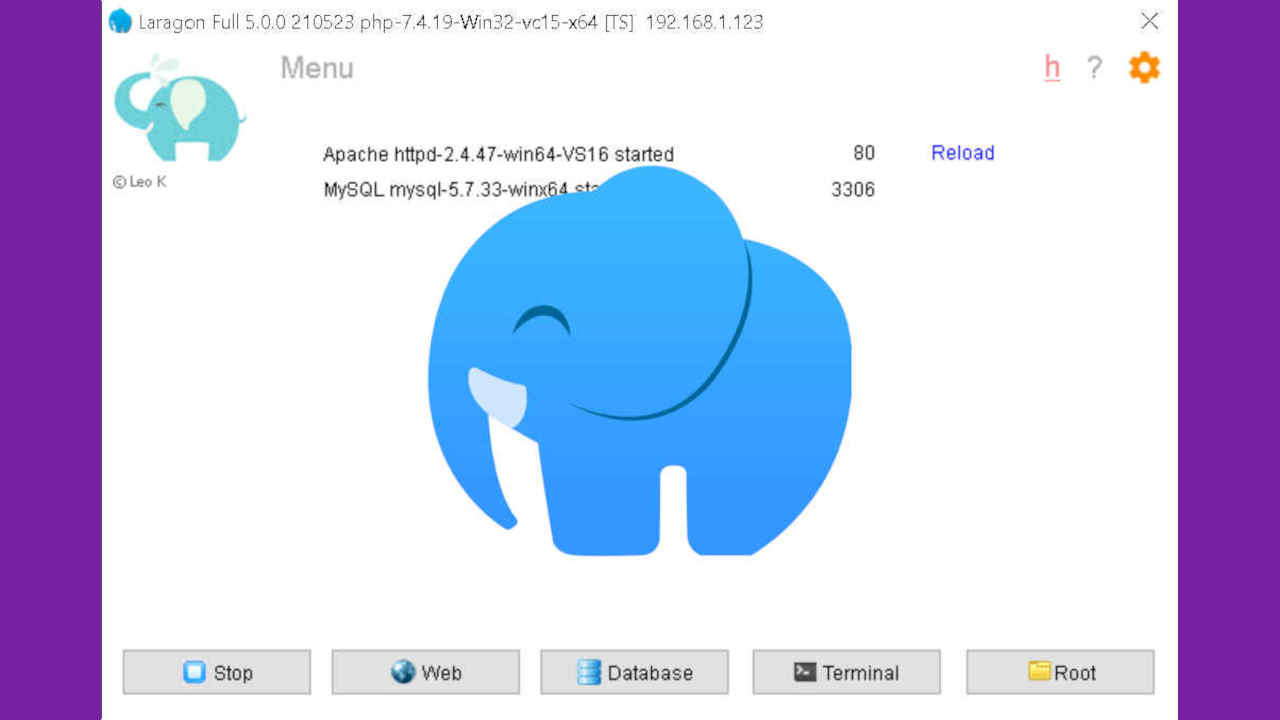
Laragon es un ambiente de desarrollo local que simplifica la creación y gestión de entornos de desarrollo web en tu computadora. Ofrece una instalación rápida y sencilla de un completo stack de desarrollo, que incluye servidores web Apache o Nginx, bases de datos MySQL, MariaDB, PHP, Node.js y otros componentes esenciales. Con Laragon, puedes crear múltiples proyectos web y alternar entre diferentes versiones de PHP, Apache, MySQL, y más con facilidad, lo que te permite trabajar en diversos proyectos sin conflictos de versiones. Además, Laragon proporciona una interfaz gráfica de usuario intuitiva para administrar tus proyectos, bases de datos y configuraciones de servidor de forma eficiente. También ofrece características avanzadas como la instalación de extensiones y herramientas adicionales con un solo clic, la configuración de dominios virtuales, la gestión de SSL, y la integración con herramientas de desarrollo como Composer y Git. En resumen, Laragon es una herramienta poderosa y versátil que agiliza el proceso de desarrollo web en tu entorno local.

Figura 7 – Logotipo de Laragon

Fuente: Herramientas de desarrollo Laragon

# 2.2 DISEÑO ARQUITECTÓNICO

La elección de la arquitectura de software para un proyecto de sistema de gestión de información depende de varios factores, incluyendo los requisitos específicos del proyecto, las necesidades del usuario, la escalabilidad, la mantenibilidad y otros. Aquí hay algunos tipos de arquitectura que podrían ser apropiados para un sistema de gestión de información:

## 2.2.1 Conceptos Clave En El Diseño Arquitectónico

El diseño arquitectónico en el desarrollo de software implica la creación de una estructura o esquema que define cómo se organizarán y comunicarán los componentes de un sistema de software. Los conceptos clave incluyen:

* **Abstracción**: Simplificación de complejidades del sistema, enfocándose en losniveles más altos de la arquitectura.
* **Separación de preocupaciones**: División del sistema en distintas secciones, cadauna con una responsabilidad única.
* **Modularidad**: Organización del software en módulos para facilitar la mantenibilidady la escalabilidad.
* **Reutilización**: Diseño de componentes que pueden utilizarse en diferentes partes delsistema o en diferentes proyectos.

## 2.2.2 Patrones De Diseño Arquitectónico

Los patrones de diseño arquitectónico son soluciones probadas a problemas comunes en el diseño de software. Algunos ejemplos incluyen:

* Modelo-Vista-Controlador (MVC): Separa la interfaz de usuario, la lógica de negocio y la manipulación de datos.
* Modelo-Vista-Presentador (MVP): Similar a MVC, pero con un enfoque más fuerte en la lógica de presentación.
* Modelo-Vista-ViewModel (MVVM): Facilita la separación entre la lógica de la interfaz de usuario y la lógica de negocio, especialmente en aplicaciones con interfaces ricas.

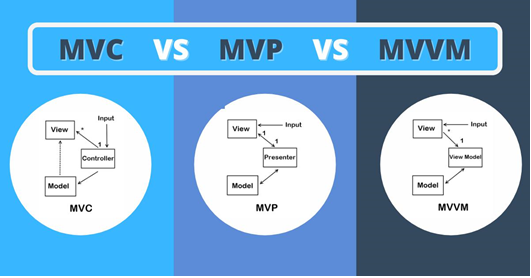


Figura 8.- Representación visual de 3 patrones de diseño

Fuente: Ingeniería y Arquitectura Del Software

# 2.3 ORGANIZACIÓN DE LA INFORMACIÓN

## 2.3.1 Gestión De Datos En La Arquitectura

La gestión de datos en la arquitectura de software se refiere a las estrategias y prácticas empleadas para manejar los datos a lo largo de todo el sistema. Esto incluye la forma en que los datos se almacenan, recuperan, actualizan y eliminan. La eficiencia en la gestión de datos es crucial para el rendimiento del sistema, la escalabilidad y la integridad de los datos. Aspectos importantes de la gestión de datos incluyen:

* Almacenamiento de Datos: Elección entre bases de datos relacionales, NoSQL, o almacenamiento en la nube, basándose en los requisitos de volumen, velocidad y variedad de datos.
* Caché y Persistencia: Implementación de mecanismos de caché para mejorar el rendimiento y estrategias de persistencia para garantizar la durabilidad de los datos.
* Seguridad de los Datos: Protección de los datos contra accesos no autorizados y asegurando la confidencialidad, integridad y disponibilidad.
* Transacciones y Concurrencia: Manejo de transacciones para mantener la consistencia de los datos y gestión de la concurrencia para soportar múltiples usuarios o procesos.

## 2.3.2 Modelado De Datos Y Bases De Datos

El modelado de datos es un proceso crítico en la arquitectura de software, que implica definir cómo se representarán y organizarán los datos. Las consideraciones clave en el modelado de datos incluyen:

* **Esquemas de Base de Datos:** Definición de tablas, campos, relaciones, índices y restricciones.
* **Normalización:** Proceso de estructurar una base de datos relacional para reducir la redundancia de datos y mejorar la integridad.
* **Diseño NoSQL:** Para bases de datos NoSQL, el enfoque se centra en la escalabilidad y la flexibilidad para manejar grandes volúmenes de datos no estructurados o semiestructurados.
* **Relaciones y Asociaciones:** Definición de cómo se relacionan los diferentes conjuntos de datos entre sí.

El modelado de datos y las bases de datos son fundamentales en la arquitectura de software, ya que proporcionan el marco para almacenar, recuperar y manipular datos de manera eficiente y segura, lo cual es esencial para el funcionamiento adecuado de cualquier sistema de software.

## 2.3.3. Modos de organizar la información

Los estilos arquitectónicos son patrones o paradigmas que se pueden organizar de la siguiente manera:

* **Arquitectura Basada en Componentes:** Descompone el sistema en componentes reutilizables y modulares. Cada componente encapsula una funcionalidad específica y se conecta con otros componentes a través de interfaces definidas.
* **Arquitectura Basada en Eventos:** Los componentes del sistema se comunican a través de eventos y reacción Facilita el desacoplamiento y la gestión de cambios en el sistema.
* **Arquitectura sin Servidor (Serverless):** Se centra en la ejecución de funciones individuales en lugar de mantener servidores continuamente en funcionamiento. Adecuado para aplicaciones con cargas de trabajo intermitentes y variables.
* **Arquitectura en Estrella:** Utilizada en sistemas de procesamiento de datos, donde un componente central (nodo central) coordina y conecta varios nodos periféricos
* **Arquitectura Cliente-Servidor Distribuida:** Similar a la arquitectura cliente-servidor, pero distribuida en una red de computadoras. Puede incluir múltiples servidores que realizan funciones específicas.
* **Arquitectura de Redes Neuronales:** Se compone de capas de nodos interconectados que procesan información de manera no lineal.

La elección del estilo arquitectónico dependerá de factores como los requisitos del sistema, la escalabilidad, la flexibilidad y las preferencias del equipo de desarrollo. Un hombre

# 2..4 ESTILO ARQUITECTÓNICO

## 2.4.1. Tipos de arquitecturas de software.

Existen varios tipos de arquitectura de software, cada uno con sus propias características y ventajas. Aquí hay algunos de los tipos de arquitectura más comunes: Arquitectura Monolítica:

En este enfoque, todos los componentes del sistema se combinan en una sola unidad. Es fácil de desarrollar y probar, pero puede volverse difícil de mantener y escalar a medida que el sistema crece. Generalmente, todas las funciones están interconectadas y comparten el mismo espacio de memoria.

## 2.4.1.1 Arquitectura de Cliente-Servidor

Divide la aplicación en dos partes principales: el cliente, que interactúa con el usuario, y el servidor, que maneja la lógica del negocio y la gestión de datos. Facilita la escalabilidad y la distribución de responsabilidades entre el cliente y el servidor.

## 2.4.1.2 Arquitectura en Capas

Organiza la aplicación en capas jerárquicas, donde cada capa tiene una función específica. Las capas pueden incluir la interfaz de usuario, la lógica de negocio y la capa de datos. Mejora el modularidad y la mantenibilidad del sistema.

## 2.4.1.3 Arquitectura Orientada a Servicios (SOA).

Descompone la aplicación en servicios independientes e interoperables que se comunican a través de estándares. Favorece la reutilización de servicios y la flexibilidad en el desarrollo y la evolución del sistema.

## 2.4.1.4 Arquitectura Basada en Microservicios

Similar a SOA, pero con un enfoque más granular. Los Microservicios son componentes pequeños e independientes que realizan funciones específicas.

Facilita la escalabilidad y la implementación continua, pero también introduce desafíos de gestión de la complejidad y la coherencia.

## 2.4.1.5 Arquitectura de Redes Neuronales

Relacionada con sistemas de inteligencia artificial y aprendizaje profundo. Se compone de capas de nodos interconectados, cada uno con conexiones ponderadas, que procesan información de manera no lineal.

## 2.4.1.6 Arquitectura Basada en Eventos

Los componentes del sistema se comunican a través de eventos y reaccionan a esos eventos. Proporciona desacoplamiento y flexibilidad en la gestión de cambios en el sistema. Estos son solo algunos ejemplos y, en la práctica, se pueden utilizar enfoques híbridos o personalizados según las necesidades específicas del proyecto. La elección de la arquitectura depende de factores como los requisitos del sistema, la escalabilidad, la mantenibilidad y las preferencias del equipo de desarrollo.

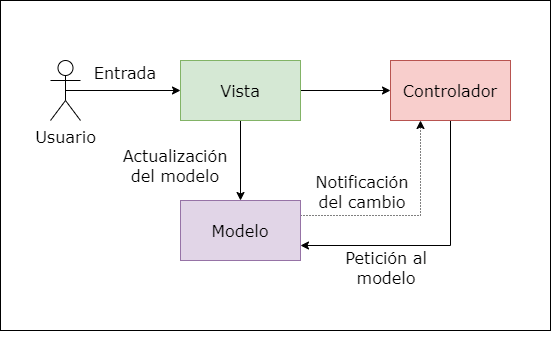


Figura 9 - Representación visual de arquitectura MVC  
 Fuente: Ingeniería y Arquitectura Del Software

## 2.3. MARCO TEÓRICO REFERENCIAL

## 2.3.1 PENDIENTE A BUSCAR

# CAPITULO III DESARROLLO DE LA INVESTIGACION

En este capítulo se desarrolla la documentación técnica del proyecto siguiendo los lineamientos del Proceso Unificado de Rational y la Notación UML.

# 3.8 Desarrollo De La Solución (Scrum)

## 3.8.1 Roles Y Stakeholders.

Los roles de Scrum son representados por una sola persona.

* Product Owner: Verónica López Terrazas
* Scrum Máster: Verónica López terrazas
* Development Team: Verónica López Terrazas
* Cliente: Empresa, Centro Médico Ofelia Sánchez

## 3.8.2 Historias De Usuario.

Se recopila información del usuario para determinar a través de historias de usuario la obtención de requisitos, estas son detalladas en las siguientes tablas:

Historia de Usuario con descripción de como ingresar al Sistema, que se quiere realizar y para que respectivamente se usa.

Tabla 1.HISTORIA DE USUARIO - INGRESAR COMO ADMINISTRADOR

|  |  |
| --- | --- |
| **HU1: Ingresar al Sistema como administrador** | |
| **Como** | Administrador. |
| **Quiero** | Ingresa al sistema a través de un correo y contraseña. |
| **Para** | Acceder a todas funcionalidades y configuraciones exclusivas de administrador. |
| **Criterio de Aceptación** | * Debe mostrar el formulario de inicio de sesión, antes de ingresar al Sistema. * Debe mostrar la pantalla principal con todos los módulos disponibles habilitados. * Debe ser capaz de crear, eliminar y editar cual información o dato de todos los módulos del Sistema. |

*Fuente: Propia Elaboración: Propia*

Diagrama

Descripción generada automáticamenteDiseño del diagrama de caso de uso para ingreso al Sistema.

**Sistema**

Figura 10. DIAGRAMA DE CASO DE USO – ACCEDER AL SISTEMA

**Fuente:** Elaboración propia

**HU2: Gestionar Usuario**

|  |  |
| --- | --- |
| **Como** | Administrador. |
| **Quiero** | Gestionar usuario |
| **Para** | Manipular el Sistema de acuerdo con su rol. |
| **Criterio de aceptación** | * Debe ser capaz de listar todos los usuarios. * Debe ser capaz de registrar, leer, modificar y eliminar usuarios. |

Tabla 2 HISTORIA DE USUARIO - GESTIONAR USUARIOS

Fuente: Elaboración propia.

Diseño del diagrama de caso de uso para Gestionar Usuarios.

Diagrama

Descripción generada automáticamente

Figura 11 DIAGRAMA DE CASO DE USO – GESTIONAR USUARIOS

Fuente: Elaboración propia

Historia de Usuario con descripción de Gestionar Roles, que se quiere realizar y para qué se utiliza.

|  |  |
| --- | --- |
| **HU3: Gestionar Roles** |  |
| **Como** | Administrador. |
| **Quiero** | Gestionar Roles. |
| **Para** | Asignar y controlar los roles a los usuarios |
|  | en el Sistema. |
| **Criterio de aceptación** | * Debe ser capaz de listar todos los roles registrados en el Sistema. * Debe ser capaz de crear, leer, modificar y eliminar. |

TABLA 3 HISTORIA DE USUARIO - GESTIONAR ROLES

Fuente: Elaboración Propia.

Diseño del diagrama de caso de uso para Gestionar Roles.

Diagrama

Descripción generada automáticamente

**Sistema**

Figura 12 DIAGRAMA DE CASO DE USO – GESTIONAR ROLES

Fuente: Elaboración propia

Historia de Usuario con descripción de Gestionar Módulos, que se quiere realizar y para qué se utiliza.

|  |  |
| --- | --- |
| **HU4: Gestionar Módulos** |  |
| **Como** | Administrador. |
| **Quiero** | Gestionar Módulos. |
| **Para** | Configurar y controlar el acceso a funcionalidades específicas. |
| **Criterio de aceptación** | • Debe ser capaz de listar todos los módulos registrados en el Sistema. |
|  | Debe ser capaz de crear, leer, modificar y eliminar. |

TABLA 4.HISTORIA DE USUARIO - GESTIONAR MÓDULOS

Fuente: Elaboración propia

Diseño del diagrama de caso de uso para Gestionar Módulos.

Diagrama

Descripción generada automáticamente

**Sistema**

Figura 13 DIAGRAMA DE CASO DE USO – GESTIONAR MÓDULOS

Fuente: Elaboración Propia.

Historia de Usuario con descripción de Gestionar Permisos, que se quiere realizar y para qué se utiliza.

|  |  |
| --- | --- |
| **HU5: Gestionar Permisos** |  |
| **Como** | Administrador. |
| **Quiero** | Gestionar Permisos. |
| **Para** | Controlar el acceso a las funcionalidades y garantizar la seguridad de los datos. |
| **Criterio de aceptación** | • Debe ser capaz de crear, leer, modificar y eliminar. |

TABLA 5.HISTORIA DE USUARIO - GESTIONAR PERMISOS

Fuente: Elaboración Propia

Diagrama

Descripción generada automáticamenteDiseño del diagrama de caso de uso para Gestionar Permisos.

Figura 14 DIAGRAMA DE CASO DE USO – GESTIONAR PERMISOS

Fuente: Elaboración Propia

Historia de Usuario con descripción de Gestionar Especialidad, que se quiere realizar y para qué se utiliza.

|  |  |
| --- | --- |
| **HU7: Gestionar Especialidades** | |
| **Como** | Administrador, Encargado |
| **Quiero** | Gestionar Especialidades. |
| **Para** | Para administrar y controlar la información de las especialidades |
| **Criterio de aceptación** | * Debe ser capaz de listar todos los Especialidades registradas en el Sistema. * Debe ser capaz de realizar búsquedas de especialidades por su nombre o rama. * Debe ser capaz de crear, leer, modificar y eliminar los detalles de una especialidad especifica. |

TABLA 6. HISTORIA DE USUARIO - GESTIONAR ESPECIALIDADES.

Fuente: Elaboración Propia.

Diagrama

Descripción generada automáticamente Diseño del diagrama de caso de uso para Gestionar Especialidad.

Figura 15. DIAGRAMA DE CASO DE USO – GESTIONAR ESPECIALIDAD

**Fuente**: Elaboración Propia.

Historia de Usuario con descripción de Gestión de pacientes.

|  |  |
| --- | --- |
| **HU8: Gestionar Pacientes** | |
| **Como** | Administrador, Encargado |
| **Quiero** | Gestionar Pacientes |
| **Para** | Mantener un registro actualizado de las existencias |
| **Criterio de aceptación** | * Debe ser capaz de listar todos los Pacientes registrados en el Sistema. * Debe ser capaz de realizar búsquedas de Pacientes por código y nombre y estado |
|  | * Debe ser capaz de registrar, leer, modificar y eliminar una entrada de un Paciente registrado. * Debe ser capaz de realizar búsquedas de Pacientes por código y nombre y estado |

TABLA 7. HISTORIA DE USUARIO - GESTIONAR PACIENTE.

Fuente: Elaboración Propia

Diseño del diagrama de caso de uso para Gestionar Pacientes.

Diagrama

Descripción generada automáticamente

Figura 16. DIAGRAMA DE CASO DE USO – GESTIONAR PACIENTES

Fuente: Elaboración Propia

Historia de Usuario con descripción de Gestionar Agendar Cita Médica, que se quiere realizar y para qué se utiliza.

|  |  |
| --- | --- |
| **HU9: Gestionar Agendar Citas** | |
| **Como** | Administrador, Encargado |
| **Quiero** | Agendar Cita medica |
| **Para** | Organizar y clasificar las citas médicas de manera adecuada. |
| **Criterio de aceptación** | * Debe ser capaz de listar todas las citas médicas existentes. * Debe ser capaz de crear, leer, modificar citas médicas del Sistema. |

TABLA 8. HISTORIA DE USUARIO - GESTIONAR AGENDAR CITAS MEDICAS

Fuente: Elaboración propia

Diagrama

Descripción generada automáticamenteDiseño del diagrama de caso de uso para Gestionar Agendar Citas Médicas.

Figura 17. DIAGRAMA DE CASO DE USO – GESTIONAR AGENDAR CITAS MEDICAS

Fuente: Elaboración Propia

Historia de Usuario con descripción de Gestionar Personal Médico, que se quiere realizar y para qué se utiliza.

|  |  |
| --- | --- |
| **HU10: Gestionar Persona Medico** | |
| **Como** | Administrador, Encargado |
| **Quiero** | Gestionar Personal Medico |
| **Para** | Administrar y controlar la información del personal médico. |
| **Criterio de aceptación** | * Debe ser capaz de listar todos los médicos disponibles. * Debe ser capaz de crear, leer, modificar y eliminar un médico que ya no pertenece a la clínica. |

TABLA 96 . HISTORIA DE USUARIO - GESTIONAR PERSONAL MEDICO

Fuente: Elaboración propia

Diseño del diagrama de caso de uso para Gestionar Personal Médico.

Diagrama

Descripción generada automáticamente

Figura 18. DIAGRAMA DE CASO DE USO – GESTIONAR PERSONAL MEDICO.

Fuente: Elaboración Propia

Historia de Usuario con descripción de Gestionar Tratamientos, que se quiere realizar y para qué se utiliza.

|  |  |
| --- | --- |
| **HU11: Gestionar Tratamientos** | |
| **Como** | Administrador, Dr. |
| **Quiero** | Gestionar Tratamientos |
| **Para** | Mantener un registro actualizado de los Tratamientos que llevan los Pacientes proveedores |
| **Criterio de aceptación** | * Debe ser capaz de listar todos los Tratamientos. * Debe ser capaz de crear, leer, modificar y eliminar un tratamiento. |

TABLA 10. HISTORIA DE USUARIO - GESTIONAR TRATAMIENTOS

Fuente: Elaboración propia

Diagrama

Descripción generada automáticamenteDiseño del diagrama de caso de uso para Gestionar Tratamientos.

Figura 19. DIAGRAMA DE CASO DE USO – GESTIONAR TRATAMIENTOS.

Fuente: Elaboración Propia.

Historia de Usuario con descripción de Gestionar Exámenes, que se quiere realizar y para qué se utiliza.

|  |  |
| --- | --- |
| **HU12: Gestionar Exámenes** | |
| **Como** | Administrador, Encargado. |
| **Quiero** | Gestionar Exámenes |
| **Para** | Mantener un registro actualizado de las enfermedades y tratamientos que lleva el Pacientes |
| **Criterio de aceptación** | * Debe ser capaz de listar todas las enfermedades registradas. * Debe ser capaz de crear, leer, modificar y eliminar una enfermedad. |

TABLA 117. HISTORIA DE USUARIO – GESTIONAR EXÁMENES.

Fuente: Elaboración Propia.

Diagrama

Descripción generada automáticamenteDiseño del diagrama de caso de uso para Gestionar Exámenes.

Figura 20. DIAGRAMA DE CASO DE USO – GESTIONAR EXAMENES.

Fuente: Elaboración Propia.

Historia de Usuario con descripción de Gestionar Historia Clínica, que se quiere realizar y para qué se utiliza.

|  |  |
| --- | --- |
| **HU13: Gestionar Historia Clínica** | |
| **Como** | Administrador, Encargado. |
| **Quiero** | Gestionar Historia Clínica. |
| **Para** | Para verificar el historial del Paciente y sus anteriores revisiones en la clínica |
| **Criterio de aceptación** | * Debe ser capaz de listar todos los tratamientos registrados. |
|  | * Debe ser capaz de mostrar. Consultar la historia clínica del Paciente |
|  |

TABLA 128. HISTORIA DE USUARIO - GESTIONAR HISTORIA CLINICA

Fuente: Elaboración propia

Diagrama

Descripción generada automáticamenteDiseño del diagrama de caso de uso para Gestionar Historia Clínica.

Figura 21. DIAGRAMA DE CASO DE USO – GESTIONAR EXAMENES.

Fuente: Elaboración Propia.

Historia de Usuario con descripción de Gestionar Tipo de Pago, que se quiere realizar y para qué se utiliza.

|  |  |
| --- | --- |
| **HU14: Gestionar Tipos de Pagos y** **Caja** | |
| **Como** | Administrador. |
| **Quiero** | Gestionar Tipo de Pago. |
| **Para** | Controlar las opciones de pago disponibles en el Sistema. |
| **Criterio de aceptación** | * Debe ser capaz de listar todos los tipos de pagos disponibles en el Sistema. * Debe ser capaz de crear, leer, modificar y eliminar un tipo de pago. |

TABLA 139 HISTORIA DE USUARIO - GESTIONAR TIPO DE PAGO

*Fuente: Elaboración propia*

Diagrama

Descripción generada automáticamenteDiseño del diagrama de caso de uso para Gestionar Tipo de Pago.

Figura 22 DIAGRAMA DE CASO DE USO – GESTIONAR TPOS DE PAGOS.

Fuente: Elaboración Propia.

Historia de Usuario con descripción de Gestionar Laboratorio que se quiere realizar y para qué se utiliza.

|  |  |
| --- | --- |
| **HU15: Gestionar Laboratorios** | |
| **Como** | Administrador, Encargado. |
| **Quiero** | Gestionar Laboratorio. |
| **Para** | Mantener un registro actualizado de los laboratorios y resultados. |
| **Criterio de aceptación** | * Debe ser capaz de listar todos los laboratorios registrados en el Sistema. * Debe ser capaz de realizar búsquedas de laboratorios pendientes * Debe ser capaz de crear, leer, modificar y eliminar un laboratorio. |

TABLA 14. HISTORIA DE USUARIO - GESTIONAR LABORATORIOS

Fuente: Elaboración propia.

Diagrama

Descripción generada automáticamenteDiseño del diagrama de caso de uso para Gestionar Laboratorios

Figura 23. DIAGRAMA DE CASO DE USO – GESTIONAR LABORATORIOS

**Fuente**: Elaboración Propia

Historia de Usuario con descripción de Gestionar Reporte de Estadísticos y de control de pacientes, que se quiere realizar y para qué se utiliza.

|  |  |
| --- | --- |
| **HU16: Generar Reporte de control y estadísticos** | |
| **Como** | Administrador. Encargado |
| **Quiero** | Generar Reporte Estadísticos y de control |
| **Para** | Reportes de control y seguimientos de pacientes |
| **Criterio de aceptación** | * Debe ser capaz de exportar los reportes de ventas en formato PDF. |
|  | * Debe ser capaz de personalizar los criterios de los reportes de pacientes clientes doctores y fecha inicial y fecha final. |
|  |

Tabla 15. HISTORIA DE USUARIO - GENERAR REPORTE DE HISTORIAS CLÍNICAS, CONTROL Y SEGUIMIENTOS

Fuente: Elaboración propia

Diseño del diagrama de caso de uso para Gestionar Reportes de control y estadísticas

Diagrama

Descripción generada automáticamente

Figura 24 DIAGRAMA DE CASO DE USO – GESTIONAR REPORTES DE CONTROL Y ESTADISTICOS

Fuente: Elaboración Propia

Historia de Usuario con descripción de Gestionar Reporte de recepción y consultas realizadas, que se quiere realizar y para qué se utiliza.

|  |  |
| --- | --- |
| **HU16: Generar Reporte de Recepción y consultas realizadas** | |
| **Como** | Administrador. Y Encargado |
| **Quiero** | Gestionar Reporte recepción y consultas realizadas |
| **Para** | Reportes de control y seguimientos de pacientes |
| **Criterio de aceptación** | • Debe ser capaz de exportar los reportes de ventas en formato PDF. |
|  | * Debe ser capaz de personalizar los criterios de los reportes de pacientes clientes doctores y fecha inicial y fecha final. |
|  |

TABLA 16. HISTORIA DE USUARIO - GENERAR REPORTE DE RECEPCION Y CONSULTAS REALIZADAS

Fuente: Elaboración propia

Diseño del diagrama de caso de uso para Gestionar Reportes de Recepción y consultas realizadas.

Diagrama

Descripción generada automáticamente

Figura 25 DIAGRAMA DE CASO DE USO – GENERAR REPORTES CONSULTAS REALIZADAS.

Fuente: Elaboración propia

## 3.8.3 Product Backlog

La lista de requisitos según a las historias de usuario y con orden dando prioridad a la implementación.

|  |  |  |
| --- | --- | --- |
| **Historia De Usuario** | **Descripción** | **Prioridad** |
| H01 | Ingresar al Sistema | 1 |
| H02 | Gestionar Usuarios | 1 |
| H03 | Gestionar Roles | 2 |
| H04 | Gestionar Módulos | 2 |
| H05 | Gestionar Permisos | 1 |
| H06 | Gestionar Triaje | 1 |
| H07 | Gestionar Especialidades | 1 |
| H08 | Gestionar Paciente | 1 |
|  |  | 2 |
| H09 | Gestionar Agendar citas Medicas |  |
| H10 | Gestionar Personal Medico | 2 |
| H11 | Gestionar Tratamientos | 2 |
| H12 | Gestionar Exámenes | 2 |
| H13 | Gestionar Historia Clínica | 2 |
| H14 | Gestionar Tipos de Pagos y Caja | 1 |
| H15 | Gestionar Laboratorio | 1 |
| H16 | Generar Reportes de control y estadísticos | 1 |
| H17 | Generar reporte de Recepción y Consultas Realizadas | 2 |

TABLA 17. LISTA DE REQUISITOS PRIORIZADA

Fuente: Elaboración propia

## 3.8.4 Estimación Del Product Backlog

Lista de todas las tareas que deben llevarse a cabo para finalizar el Proyecto. Con el fin de calcular el tiempo y esfuerzo requeridos para las historias de usuario se usa los puntos de historia y usaremos el Planning Pocker para asignar valores a cada una de ellas.

Teclado de ordenador

Descripción generada automáticamente con confianza media

Figura 26 PLANNING POCKER GUÍA

Fuente[: https://muyagile.com/lets-play-planning-poker-estimacion-agil/](https://muyagile.com/lets-play-planning-poker-estimacion-agil/)

Para definir la dificultad de las tareas y los valores respectivos a cada historia de usuario tendremos en cuenta la siguiente métrica:

* 0 la tarea ya está completa.
* 1/2: se define como una tarea muy pequeña.
* 1, 2, 3: se utilizan para pequeñas tareas.
* 5, 8, 13: se trata de una dificultad media.
* 20, 40: la tarea es grande.
* 100: la tarea es muy grande.
* Infinito: la tarea es enorme.
* Interrogación: no tengo idea de cuánto tiempo lleva completar esta tarea.
* Taza de café: Hora de Pausa.

Cada punto de historia de usuario vale 2 horas de trabajo. Un día es igual a 8 horas de trabajo hábiles. Entonces tenemos la siguiente formula: Donde PH = Puntos de Historias de Usuario.

**Duración = PH \* 2 / 8** En total tenemos 220 PH entre todos los Sprint.

**Duración en días** = 220 \* 2 / 8 = 57 días hábiles de trabajo.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Modulo** | **Requisito** | **Historia de Usuario** | **Prioridad** | **Puntos de historia** | **Tiempo**  **Estimado**  **(días)** |
| 1 | MODULO DE  ADMINISTRACION | 1 | Ingresar al Sistema como administrador | Media | 8 | 2 |
| 2 | Gestionar Usuarios | Baja | 4 | 1 |
| 3 | Gestionar Roles | Baja | 4 | 1 |
| 4 | Gestionar Módulos | Media | 8 | 2 |
| 5 | Gestionar Permisos | Media | 16 | 4 |
|  |  |  | | **Total, de días del Sprint 1** | | 10 |

TABLA 18. ESTIMACIÓN DEL PRODUCT-BACKLOG SPRINT 1

*Fuente: Elaboración propia.*

Descripción de la estimación del Product backlog del Sprint, detallando el módulo, historia de usuario, prioridad, importancia y tiempo.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Modulo** | **Requisito** | **Historia de Usuario** | **Prioridad** | **Puntos de historia** | **Tiempo**  **Estimado**  **(días)** |
| 2 |  | 6 | Gestionar Triaje | Alta | 40 | 10 |
| 7 | Gestionar Especialidades | Media | 8 | 2 |
| 8 | Gestionar Pacientes | Media | 8 | 2 |
| 9 | Gestionar Agendar Citas medicas | Baja | 4 | 1 |
| 10 | Gestionar personal médico | Baja | 4 | 1 |
| 11 | Gestionar tratamientos | Baja | 4 | 1 |
| 12 | Gestionar historia clínica | Media | 8 | 2 |
| 13 | Gestionar laboratorio | Media | 8 | 2 |
| 14 | Gestionar exámenes | Alta | 40 | 10 |
| 15  16  17 | Gestionar Pagos y caja  Gestionar reportes de control y estadística  Reportes recepción de pacientes y consultas realizadas | Alta  Alta  Alta | 40  40  40 | 10  10  10 |
|  |  | **Total, de días del Sprint 2** | | | | 45 |

TABLA 1910. ESTIMACIÓN DEL PRODUCT-BACKLOG SPRINT 2

MODULO GESTION DE PACIENTES

Fuente: Elaboración propia.

Descripción de la estimación del Product backlog del Sprint, detallando el módulo, historia de usuario, prioridad, importancia y tiempo.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Modulo** | **Requisito** | **Historia de Usuario** | **Prioridad** | **Puntos de historia** | **Tiempo**  **Estimado**  **(días)** |
| 3 | MODULO DE CLIENTE | 16 | Gestionar Clientes | Media | 10 | 3 |
| 17 | Gestionar Tipo de Pago | Media | 10 | 3 |
|  |  | **Total, de días del Sprint 3** | | | | 6 |

TABLA 20. ESTIMACIÓN DEL PRODUCT-BACKLOG SPRINT 3

Fuente: Elaboración propia

Descripción de la estimación del Product backlog del Sprint 3, detallando el módulo, historia de usuario, prioridad, importancia y tiempo.

El total del tiempo estimado para la realización de todos los Sprints es de 90 días.

## 3.8.5. Primer Sprint

En la siguiente tabla se mostrarán los días que se tomarán para el desarrollo el inicio y fin de la tarea en una reunión que se realiza al comienzo de cada sprint donde participa el equipo de scrum.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Historia De**  **Usuario** | **Descripción** | **Puntos De**  **Historia** | **Tiempo**  **Estimado (días)** | **Fecha Inicio** | **Fecha Fin** |
| H01 | Ingresar al sistema | 8 | 2 | 18/04/23 | 20/04/23 |
| H02 | Gestionar Usuarios | 4 | 1 | 20/04/23 | 21/04/23 |
| H03 | Gestionar Roles | 4 | 1 | 21/04/23 | 22/04/23 |
| H04 | Gestionar Módulos | 8 | 2 | 22/04/23 | 24/04/23 |
| H05 | Gestionar Permisos | 16 | 4 | 24/04/23 | 28/04/23 |

TABLA 21. SPRINT PLANNING - SPRINT 1

Fuente: Elaboración propia

## 3.8.5.1. Diagrama de Caso de Uso – Sprint 1

Diagrama de caso de uso general representando las historias de usuario del Sprint 1.

Diagrama

Descripción generada automáticamente

Sistema

Figura 27. DIAGRAMA DE CASO DE USO - SPRINT 1

Fuente: Elaboración propia.

## 3.8.5.2. Especificación de Caso de uso – Sprint 1

Para la especificación de los casos de uso utilizamos historias de usuario detallados.

Ingresa al Sistema como administrador con todos los privilegios.

|  |  |
| --- | --- |
| **Caso de uso HU1** | **HU1: Ingresar el Sistema al sistema** |
| **Descripción** | Ingresar al Sistema mediante un formulario de ingreso. |
| **Estado** | Completado. |
| **Actores** | Administrador. |
| **Asignación** | Verónica López |
| **Criterio de aceptación** | * Debe mostrar el formulario de inicio de sesión, antes de ingresar al Sistema. * Debe mostrar la pantalla principal con todos los módulos disponibles habilitados. * Debe ser capaz de crear, eliminar y editar cual información o dato de todos los módulos del Sistema. |
| **Pre-condición** | El usuario debe estar autenticado en el Sistema como administrador. |
| **Post-condición** | El Sistema permite gestionar roles. |

TABLA 22. ESPECIFICACIÓN DE CASO DE USO HU1

Fuente: Elaboración Propia

|  |  |
| --- | --- |
| **Caso de uso HU2** | **HU2: Gestionar Usuarios** |
| **Descripción** | Se podrá Gestionar al Usuario por medio de CRUD y las opciones que tengan con sus respectivos atributos. |
| **Estado** | Completado. |
| **Actores** | Administrador. |
| **Asignación** | Verónica López Terrazas |
| **Criterio de aceptación** | * Debe ser capaz de listar todos los usuarios. * Debe ser capaz de registrar, leer, modificar y eliminar usuarios. |
| **Pre-condición** | El usuario debe estar autenticado en el Sistema como administrador. |
| **Post-condición** | El Sistema permite gestionar usuarios. |

TABLA 23. ESPECIFICACIÓN DE CASO DE USO HU2

**Fuente**: Elaboración propia

|  |  |
| --- | --- |
| **Caso de uso HU3** | **HU3: Gestionar Roles** |
| **Descripción** | Se podrá Gestionar Roles por medio de CRUD y sus respectivos atributos. |
| **Estado** | Completado. |
| **Actores** | Administrador. |
| **Asignación** | Verónica López terrazas |
| **Criterio de aceptación** | * Debe ser capaz de listar todos los roles registrados en el Sistema. * Debe ser capaz de crear, leer, modificar y eliminar. |
| **Pre-condición** | El usuario debe estar autenticado en el Sistema como administrador. |
| **Post-condición** | El Sistema permite gestionar roles. |

TABLA 2411. ESPECIFICACIÓN DE CASO DE USO HU3

Fuente: Elaboración propia

|  |  |
| --- | --- |
| **Caso de uso HU4** | **HU4: Gestionar Módulos** |
| **Descripción** | Gestionar los módulos del sistema a través de operaciones CRUD (Crear, Leer, Actualizar y Eliminar) y sus respectivos atributos. |
| **Estado** | Completado. |
| **Actores** | Administrador. |
| **Asignación** | Verónica López Terrazas |
| **Criterio de aceptación** | * Debe ser capaz de listar todos los módulos registrados en el Sistema. * Debe ser capaz de crear, leer, modificar y eliminar. |
| **Pre-condición** | El usuario debe estar autenticado en el Sistema como administrador. |
| **Post-condición** | El Sistema permite gestionar módulos. |

TABLA 25. ESPECIFICACIÓN DE CASO DE USO HU4

Fuente: Elaboración propia

|  |  |
| --- | --- |
| **Caso de uso HU5** | **HU5: Gestionar Permisos** |
| **Descripción** | Se podrá Gestionar Permisos por medio de CRUD y sus respectivos atributos. |
| **Estado** | Completado. |
| **Actores** | Administrador. |
| **Asignación** | Verónica López Terrazas |
| **Criterio de aceptación** | • Debe ser capaz de crear, leer, modificar y eliminar. |
| **Pre-condición** | El usuario debe estar autenticado en el Sistema como administrador. |
| **Post-condición** | El Sistema permite gestionar permisos. |

TABLA 26. ESPECIFICACIÓN DE CASO DE USO HU5

Fuente: Elaboración propia

## 3.8.5.2 Sprint Backlog Sprint 1

Para determinar cómo se va a desarrollar el sprint, se identifican una o varias tareas por historia de usuario, las cuales son agrupadas en un sprint backlog y asignadas a los miembros del equipo de desarrollo.

|  |  |  |
| --- | --- | --- |
| **Sprint** | **Historia de Usuario** | **Tareas** |
| 1 | Ingresar al Sistema como administrador | 1. Diseñar vista para el formulario de ingreso. 2. Desarrollar interfaz del formulario de ingreso. 3. Validar acceso al administrador. |
| Gestionar Usuarios | 1. Desarrollar el CRUD. 2. Listar usuarios registrados. |
| Gestionar Roles | 1. Desarrollar el CRUD. 2. Listar roles registrados. |
|  | Gestionar Módulos | 1. Desarrollar el CRUD. 2. Listar módulos registrados. |
| Gestionar Permisos | 1. Desarrollar el CRUD. 2. Listar permisos registrados. |

TABLA 2712. SPRINT BACKLOG SPRINT 1

Fuente: Elaboración propia

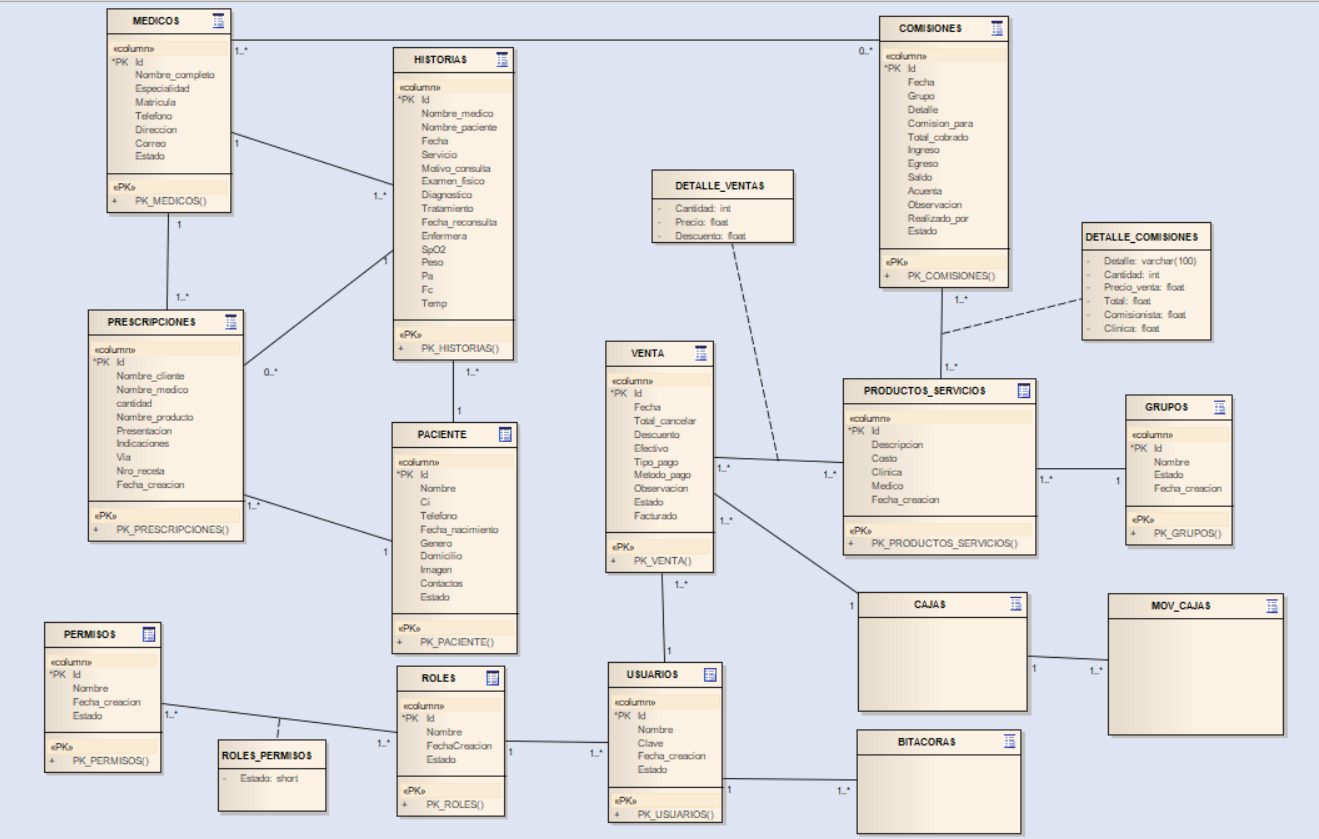


Figura 28. DIAGRAMA DE BASE DE DATOS

Fuente: Elaboración propia.

Introducción a UML: Lenguaje para modelar objetos. (2017). (n.p.): Natsys.

Subra, J., Vannieuwenhuyse, A. (2018). Scrum: un método ágil para sus proyectos. España: Ediciones Eni.

Dimes, T. (2015). Conceptos Básicos De Scrum: Desarrollo De Software Agile Y Manejo De Proyectos Agile. Estados Unidos: Babelcube Incorporated.

Sánchez, C. (24 de enero de 2020). Bibliografía. Normas APA (7ma edición). <https://normas-apa.org/bibliografia/>

Arias, Á., Durango, A. (2016). Ingeniería y Arquitectura Del Software: 2a Edición. (n.p.): CreateSpace Independent Publishing Platform.

FRAMEWORK TOTAL - Vol.2: Crea APPs desde Cero con Laravel + W3.CSS + MySQL. (2021). (n.p.): RedUsers.

Baker, C. (2023). Aprende Vue.js: Guía para desarrolladores web. (n.p.): Amazon Digital Services LLC - Kdp.