**UNIVERSIDAD AUTONOMA GABRIEL RENE MORENO**

**FACULTAD DE INGENIERÍA EN CIENCIAS DE LA COMPUTACIÓN Y TELECOMUNICACIONES**

**Grupo 18**



**‘‘Sistema de Información Web y Móvil para la gestión de historias clínicas electrónicas para el Seguro de Salud Vida Sana “SSVS”**

**Integrantes:**

* **Guzman Montalvan Marcos Lehonti 220030081**
* **Ramos Vino Abigail**

**Materia: Sistemas de Información II – INF 412**

**Grupo: SA**

**Nombre del Docente:** **Ing. M. Sc. Ing. Angélica Garzón Cuéllar**

Contenido

[**1.FUNDAMENTOS TEORICOS** 3](#_Toc180766288)

[1.1 Introducción 3](#_Toc180766289)

[1.2 Objetivo General 4](#_Toc180766290)

[1.3 Objetivos Específicos 4](#_Toc180766291)

[1.4 Alcance 5](#_Toc180766292)

[**2.MARCO TEORICO REFERENCIAL Y METODOLOGICO** 8](#_Toc180766293)

[**3.HERRAMIENTAS TECNOLOGICAS PARA EL DESARROLLO** 8](#_Toc180766294)

[3.1 Lenguaje de Programacion (Web y Movil) 8](#_Toc180766295)

[3.2 Frameworks y entornos de ejecución 10](#_Toc180766296)

[3.3 Gestor de Base de Datos 12](#_Toc180766297)

[3.4. Lenguaje de modelado de software (Modelo C4) 13](#_Toc180766298)

[3.5. Herramientas de diseño y modelado 13](#_Toc180766299)

[3.6. Entorno de Desarrollo 15](#_Toc180766300)

[3.7. Sistema de control de Versiones de código (Git) 16](#_Toc180766301)

[**4.REQUERIMIENTOS** 17](#_Toc180766302)

[4.1 Propósitos 17](#_Toc180766303)

[4.2 Ámbitos del Sistema 18](#_Toc180766304)

[4.3 Equipo Scrum 18](#_Toc180766305)

[3.4 Definiciones, Acrónimos y Abreviaturas (Referencias) 19](#_Toc180766306)

[3.5 Funciones del Producto (Describir características de funcionalidad) 20](#_Toc180766307)

[3.6 Product Backlog 21](#_Toc180766308)

[3.7 Requisitos Funcionales 21](#_Toc180766309)

[3.8 Requisitos No Funcionales 21](#_Toc180766310)

[3.9 Lista de Casos de Uso 21](#_Toc180766311)

[3.10 Paquetes y casos de uso 21](#_Toc180766312)

[3.11 Planificacion Sprint (Diagrama de Gantt) 21](#_Toc180766313)

[**5.PROCESO DE DESARROLLO DE SOFTWARE** 22](#_Toc180766314)

[**Sprint 1** 22](#_Toc180766315)

[1.Sprint Planning 22](#_Toc180766316)

[1.1 Objetivos del Sprint 22](#_Toc180766317)

[1.2 Historia de usuario (tarjetas 3C, Planning Poker, prototipos) 22](#_Toc180766318)

[1.3 Contexto del sistema 22](#_Toc180766319)

[1.4 Sprint backlog 22](#_Toc180766320)

[1.5 Equipo SCRUM 22](#_Toc180766321)

# **1.FUNDAMENTOS TEORICOS**

## 1.1 Introducción

En la era actual, la digitalización y la automatización de procesos son esenciales para el éxito en el sector de la salud, especialmente en áreas clave como la gestión de historias clínicas electrónicas. Las instituciones de salud que adoptan sistemas avanzados para manejar sus registros médicos tienen una ventaja significativa, ya que estos sistemas no solo reducen el tiempo y los costos asociados a la gestión manual de datos clínicos, sino que también mejoran la precisión y eficiencia en el cuidado de la salud y la toma de decisiones médicas.

La gestión de historias clínicas en un entorno complejo y de gran escala, como el del "Seguro de Salud Vida Sana," requiere atención especial debido a la variedad y el volumen de información de cada paciente que debe ser registrada, consultada y actualizada. La correcta administración de estos datos es crucial para garantizar la calidad de la atención y la seguridad de los pacientes, contribuyendo a la transparencia y la eficiencia operativa en todos los niveles de atención médica.

Ante estos desafíos, se propone el desarrollo de un sistema de información especializado que permita gestionar y controlar eficientemente las historias clínicas de cada paciente desde una plataforma centralizada. Este sistema optimizará los procesos de atención al ofrecer un acceso más rápido y preciso a la información médica de cada persona, facilitando la consulta de datos, la generación de reportes médicos y la toma de decisiones clínicas. Además, contribuirá a una mejor administración de los recursos de salud, asegurando que los profesionales puedan brindar un servicio de calidad en todo momento y que los pacientes reciban una atención oportuna y personalizada.

Este software web y móvil de gestión de historias clínicas permitirá también la integración de nuevas funcionalidades para un control exhaustivo de la salud de cada paciente, incluyendo alertas automáticas para seguimiento de tratamientos, recordatorios de citas y monitoreo de indicadores clave de salud. Al centralizar toda la información relevante en una plataforma segura y accesible, el sistema garantizará una experiencia de atención médica de alta calidad, con especial énfasis en la prevención y el manejo oportuno de condiciones crónicas, logrando así un impacto positivo y sostenible en la vida de los asegurados del "Seguro de Salud Vida Sana".

## 1.2 Objetivo General

Desarrollar un Software web y móvil para la gestión electrónica del historial clínico de los pacientes del seguro social vida sana.

## 1.3 Objetivos Específicos

* Administrar y controlar el acceso al sistema mediante roles definidos como Administrador, Médico y Paciente. Esto garantiza que cada usuario tenga permisos específicos según su función, permitiendo a los administradores gestionar la configuración del sistema, a los médicos acceder y actualizar la información clínica de los pacientes, y a los pacientes consultar su historial médico. Esta estructura asegura una gestión eficiente y segura de la información en la plataforma.
* Realizar la gestión de datos tanto de los pacientes como del personal médico, logrando almacenar la información que será utilizada para lograr un mejor manejo de los datos en el sistema.
* Organizar y administrar los horarios de atención de los médicos, permitiendo la asignación de turnos, control de disponibilidad y coordinación eficiente de citas para optimizar el tiempo y la atención al paciente.
* Administrar y clasificar las especialidades médicas disponibles, facilitando la asignación de profesionales según su especialidad y mejorando la gestión de consultas y tratamientos específicos.
* Administrar y actualizar la información de los pacientes, garantizando un acceso rápido y seguro a sus datos personales y médicos.
* Permitir a los pacientes solicitar una ficha de atención diaria en línea, asegurando un control para evitar múltiples fichas en un mismo día.
* Actualizar y gestionar la Historia Clínica del paciente en cada consulta, registrando diagnósticos, tratamientos y observaciones médicas para un historial completo y accesible.
* Facilitar la creación de reportes detallados de atención a pacientes, permitiendo a la administración filtrar según criterios como periodo de fechas, paciente, médico y especialidad. Estos reportes ayudarán a evaluar la eficiencia y calidad del servicio. Además, el sistema permitirá a los pacientes acceder a su historia clínica para consultar su información médica, asegurando que estos datos solo sean visibles y no modificables por ellos.
* Implementar la metodología ágil SCRUM en el desarrollo del software, lo que permitirá la entrega continua y iterativa de funcionalidades. Esta aproximación asegura que el software evolucione de forma constante, adaptándose de manera flexible a los cambios en los requisitos y garantizando una mejora continua según las necesidades de los usuarios.

## 1.4 Alcance

Este proyecto se enfocará en desarrollar un sistema de gestión de historias clínicas electrónicas que permita a los profesionales de la salud administrar la información clínica de los pacientes. Además, incluirá funcionalidades para registrar médicos, asegurados, citas y los servicios médicos ofrecidos, facilitando así un acceso eficiente y seguro a los datos clínicos.

Módulos:

Modulo Gestión de Usuario:

* Gestionar Roles: permite definir y administrar los diferentes roles de usuarios en el sistema, como Administrador, Médico y Paciente. Cada rol tiene permisos específicos que determinan las acciones que el usuario puede realizar dentro de la aplicación. La gestión de roles facilita el control de acceso y asegura que la información sensible esté protegida. Además, permite la personalización de las funciones según las necesidades de cada tipo de usuario.
* Gestionar Permisos: se encarga de asignar y modificar los derechos de acceso para cada rol dentro del sistema. Esto incluye la autorización para visualizar, editar o eliminar datos en diferentes secciones de la aplicación. Al establecer permisos granulares, se garantiza la seguridad y confidencialidad de la información, además de cumplir con las normativas vigentes en el manejo de datos personales. La flexibilidad en la configuración de permisos permite adaptarse a los cambios organizativos y operativos.
* Gestionar Administrador: permite la creación, edición y eliminación de cuentas de administrador, quienes tienen acceso total al sistema. Los administradores pueden gestionar usuarios, roles y permisos, asegurando el correcto funcionamiento del software. También se encarga de supervisar el desempeño general del sistema y realizar ajustes necesarios en la configuración. Además, permite generar reportes de actividad y asegurar el cumplimiento de protocolos de seguridad
* Gestionar Medico: permite registrar, actualizar y eliminar información sobre los profesionales de la salud dentro del sistema. Incluye datos como especialidades, horarios de atención, y la ubicación de sus consultorios. También facilita la asignación de citas y la visualización de la carga de trabajo de cada médico. Este módulo es crucial para optimizar la atención al paciente y asegurar una adecuada coordinación entre los diferentes profesionales de la salud.
* Gestión de Bitácora: La gestión de bitácora se encarga de registrar todas las acciones realizadas por los usuarios en el sistema, incluyendo accesos, modificaciones y eliminaciones de datos. Este registro es fundamental para auditar el uso del software y garantizar la trazabilidad de la información. Permite identificar comportamientos inusuales y posibles infracciones de seguridad, mejorando así la protección de datos sensibles. Además, facilita la generación de informes sobre la actividad del sistema y los usuarios, contribuyendo a la toma de decisiones informadas

Modulo Gestión de Pacientes:

* Registro de Pacientes: permite capturar y almacenar información básica de los pacientes, incluyendo datos personales como peso, temperatura, altura, contacto y detalles de aseguramiento. Los usuarios con permisos pueden agregar, editar y eliminar registros de pacientes de manera eficiente y segura. Además, facilita la organización de los datos, asegurando un acceso rápido y fácil a la información necesaria para la atención médica. El registro sistemático de pacientes es fundamental para el seguimiento y la continuidad de la atención.
* Gestión de Antecedentes: se encarga de registrar y actualizar la información sobre los antecedentes médicos de cada paciente, incluyendo enfermedades previas, alergias, tratamientos y cirugías. Permite a los profesionales de la salud tener una visión completa del historial clínico del paciente, mejorando la calidad de la atención. La gestión de antecedentes es esencial para la toma de decisiones clínicas informadas y para el manejo adecuado de tratamientos. Además, garantiza que se mantenga la confidencialidad de la información médica sensible.
* Gestión de Seguro: permite registrar y administrar la información relacionada con los planes de seguro de salud de los pacientes. Esto incluye detalles como el tipo de cobertura, número de póliza y fecha de vencimiento. Facilita la verificación de la elegibilidad de los pacientes para recibir servicios médicos y el manejo eficiente de reclamos y autorizaciones. Esto es crucial para asegurar que los pacientes accedan a los servicios adecuados y que los procesos de facturación se realicen de manera fluida.

Modulo Gestión de Consultas Médicas:

* Solicitar Cita: permite a los pacientes programar citas con médicos de manera sencilla y rápida. Los pacientes pueden seleccionar la especialidad, el médico y el horario disponibles, facilitando la planificación de su atención médica. Además, el sistema enviará confirmaciones y recordatorios automáticos a los pacientes, reduciendo las ausencias y mejorando la eficiencia del consultorio. Esta funcionalidad optimiza la gestión del tiempo tanto para los pacientes como para los profesionales de la salud.
* Gestionar Diagnostico: se encarga de registrar y actualizar los diagnósticos realizados a los pacientes durante las consultas. Permite a los médicos ingresar información sobre los hallazgos clínicos, pruebas realizadas y conclusiones sobre la salud del paciente. La gestión de diagnósticos es crucial para la continuidad de la atención, ya que proporciona un historial claro que puede ser consultado en futuras citas. Además, facilita la elaboración de informes y análisis estadísticos sobre la salud de la población atendida.
* Gestionar Tratamiento: permite a los profesionales de la salud registrar y administrar los tratamientos prescritos a los pacientes. Incluye detalles sobre medicamentos, terapias, procedimientos y seguimiento de la evolución del paciente. Asegura que cada paciente reciba el tratamiento adecuado y permite un seguimiento efectivo de su progreso. Además, facilita la comunicación entre los médicos y otros profesionales de la salud involucrados en la atención del paciente, garantizando una atención integral.

Modulo Gestión de Especialidades:

* Gestionar Especialidades: permite definir y administrar las diferentes áreas médicas disponibles en la institución, como pediatría, oftalmología y cardiología. Los administradores pueden añadir, editar o eliminar especialidades según las necesidades del centro de salud. La gestión de especialidades asegura que los pacientes tengan acceso a la atención especializada adecuada y ayuda a organizar la programación de citas de manera eficiente. Además, facilita la capacitación continua de los profesionales al identificar áreas de especialización requeridas.

# **2.MARCO TEORICO REFERENCIAL Y METODOLOGICO**

# **3.HERRAMIENTAS TECNOLOGICAS PARA EL DESARROLLO**

## 3.1 Lenguaje de Programacion (Web y Movil)

Java

Java es un lenguaje de programación orientado a objetos ampliamente utilizado en el desarrollo de aplicaciones empresariales, y Spring Boot es un framework que aprovecha las capacidades de Java para facilitar la creación de aplicaciones robustas y escalables. En este contexto, Java se destaca por varias razones:

* Versatilidad y Portabilidad: Java es conocido por su lema "escribe una vez, ejecuta en cualquier lugar". Esto significa que las aplicaciones desarrolladas en Java pueden ejecutarse en cualquier plataforma que tenga una Máquina Virtual de Java (JVM), lo que lo convierte en una excelente opción para aplicaciones que requieren alta portabilidad.
* Ecosistema Rico: Java cuenta con un extenso ecosistema de bibliotecas y frameworks que facilitan el desarrollo. Spring Boot se integra perfectamente en este ecosistema, permitiendo a los desarrolladores utilizar numerosas bibliotecas de Java para funcionalidades adicionales, como acceso a bases de datos, seguridad y gestión de transacciones.
* Simplicidad en la Configuración: Aunque Java puede requerir configuraciones más extensas en comparación con otros lenguajes, Spring Boot simplifica este proceso. Utiliza convenciones sobre configuraciones y proporciona configuraciones automáticas, permitiendo a los desarrolladores centrarse en la lógica del negocio en lugar de en la infraestructura.
* Programación Asincrónica y Multihilo: Java ofrece soporte robusto para la programación multihilo y asincrónica, lo cual es esencial para crear aplicaciones que pueden manejar múltiples solicitudes simultáneamente. Spring Boot permite implementar estas características de manera eficiente, mejorando el rendimiento de la aplicación.
* Amplia Comunidad y Soporte: Java tiene una comunidad grande y activa, lo que significa que hay abundantes recursos, tutoriales y foros disponibles para ayudar a los desarrolladores. Esto es especialmente valioso al trabajar con Spring Boot, ya que los desarrolladores pueden encontrar fácilmente soluciones y compartir experiencias.

Dart

Dart es un lenguaje de programación optimizado para el desarrollo de aplicaciones multiplataforma, especialmente en el contexto de aplicaciones móviles y web. Fue desarrollado por Google y es el lenguaje principal detrás del framework Flutter, que permite crear interfaces de usuario nativas para diversas plataformas. Algunas características clave de Dart son:

* Compilación Just-in-Time (JIT) y Ahead-of-Time (AOT): Dart utiliza ambos enfoques de compilación, lo que permite un desarrollo rápido durante la fase de prueba (JIT) y un rendimiento óptimo en producción (AOT).
* Sintaxis Clara y Concisa: La sintaxis de Dart es fácil de entender, lo que facilita el aprendizaje para nuevos desarrolladores y acelera el proceso de desarrollo.
* Soporte para Programación Asincrónica: Dart ofrece características como async y await, que permiten gestionar fácilmente operaciones asincrónicas, lo que es crucial en aplicaciones que requieren interacciones en red.
* Fuerte Enfoque en la UI: Con Flutter, Dart permite crear interfaces de usuario atractivas y responsivas que funcionan en diferentes dispositivos, ofreciendo una experiencia nativa en Android e iOS.

## 3.2 Frameworks y entornos de ejecución

Spring Boot

Spring Boot es un marco de trabajo (framework) de Java diseñado para facilitar la creación de aplicaciones empresariales. Basado en el popular framework Spring, Spring Boot elimina la necesidad de configuraciones manuales extensas al proporcionar configuraciones automáticas y predeterminadas que permiten a los desarrolladores concentrarse en la lógica de negocio. Algunas de sus características más destacadas incluyen:

* Configuración Automática: Spring Boot automáticamente configura la aplicación según las dependencias que se añaden en el proyecto, lo que reduce significativamente el tiempo de configuración.
* Servidor Embebido: Permite ejecutar la aplicación como una aplicación independiente, integrando servidores como Tomcat o Jetty, lo que simplifica el proceso de desarrollo y despliegue.
* Creación de APIs RESTful: Ofrece soporte nativo para la creación de APIs RESTful, facilitando la construcción de aplicaciones que se comunican a través de HTTP, lo que es esencial para el desarrollo de microservicios.
* Integración con Base de Datos: Soporta múltiples bases de datos (SQL y NoSQL) y permite la configuración de conexión mediante Spring Data, lo que facilita la implementación de la lógica de acceso a datos.
* Seguridad: Con Spring Security, se puede implementar fácilmente la autenticación y autorización, lo que proporciona un nivel adicional de protección a la aplicación.
* Métricas y Monitoreo: Incluye herramientas para la supervisión del rendimiento de la aplicación, así como para la gestión de sus métricas y logs, lo que facilita el mantenimiento y la optimización.

Angular

Angular es un framework de desarrollo de aplicaciones web de código abierto mantenido por Google, que permite crear aplicaciones de una sola página (SPA) con interfaces de usuario dinámicas y responsivas. Utiliza TypeScript, un superconjunto de JavaScript que incluye tipado estático, lo que mejora la calidad del código y la experiencia de desarrollo. Algunas características clave de Angular incluyen:

* **Componentes Reutilizables**: Angular se basa en una arquitectura basada en componentes, donde cada parte de la interfaz de usuario se encapsula en un componente independiente, lo que promueve la reutilización del código y una mejor organización del proyecto.
* **Enrutamiento Avanzado**: Incluye un sistema de enrutamiento robusto que permite la navegación entre diferentes vistas de manera eficiente y fácil, lo que es fundamental para las aplicaciones de una sola página.
* **Inyección de Dependencias**: Facilita la gestión de dependencias entre diferentes partes de la aplicación, promoviendo la modularidad y la facilidad de pruebas.
* **Gestión del Estado**: Ofrece soluciones para gestionar el estado de la aplicación, ya sea mediante el uso de servicios o bibliotecas adicionales como NgRx, que implementa el patrón Redux.
* **Soporte para Formularios**: Proporciona herramientas potentes para la creación y validación de formularios, permitiendo a los desarrolladores manejar entradas de usuario de manera eficiente y segura.
* **Pruebas Integradas**: Angular incluye soporte para pruebas unitarias y de integración, lo que permite a los desarrolladores asegurarse de que cada parte de la aplicación funcione correctamente.

## 3.3 Gestor de Base de Datos

PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos relacional de código abierto que se destaca por su robustez, flexibilidad y cumplimiento de estándares. Es conocido por su capacidad para manejar grandes volúmenes de datos y su compatibilidad con diferentes tipos de datos. Algunas características clave de PostgreSQL incluyen:

* **Soporte para Datos Avanzados:** PostgreSQL permite el uso de tipos de datos avanzados como JSON, XML, y tipos geoespaciales, lo que lo hace ideal para aplicaciones que manejan datos no estructurados o semiestructurados.
* **Transacciones y Concurrencia:** Proporciona un sólido soporte para transacciones ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad), asegurando la integridad de los datos incluso en situaciones de alta carga y concurrencia.
* **Extensibilidad:** Los usuarios pueden definir sus propios tipos de datos, funciones y operadores, lo que permite adaptar la base de datos a necesidades específicas del proyecto.
* **Funciones de Búsqueda Potente:** Incluye capacidades avanzadas de búsqueda y filtrado, así como la posibilidad de realizar búsquedas textuales completas, lo que mejora la eficiencia de las consultas.
* **Replicación y Escalabilidad:** Ofrece soporte para replicación y particionamiento, lo que permite escalar horizontalmente y gestionar cargas de trabajo grandes y complejas.
* **Seguridad:** Proporciona múltiples niveles de seguridad, incluyendo autenticación de usuarios, roles, y encriptación de datos, asegurando que la información esté protegida.

## 3.4. Lenguaje de modelado de software (Modelo C4)

El Modelo C4 (Context, Containers, Components, and Code) es una metodología de modelado arquitectónico diseñada para representar de manera clara y comprensible la estructura de un sistema de software. Esta metodología fue desarrollada por Simon Brown y se centra en proporcionar diferentes niveles de abstracción, lo que permite a los equipos de desarrollo y a las partes interesadas entender mejor cómo se organizan y comunican las diversas partes del software. El modelo se compone de cuatro niveles: el diagrama de contexto, que muestra el sistema en su entorno; el diagrama de contenedores, que detalla los diferentes contenedores que componen el sistema; el diagrama de componentes, que ilustra los componentes internos de cada contenedor; y el diagrama de código, que se centra en la implementación específica del código dentro de los componentes.

Cada uno de estos niveles ofrece una vista distinta del sistema, permitiendo a los desarrolladores y arquitectos comunicar de manera efectiva la arquitectura del software. Por ejemplo, el diagrama de contexto proporciona una visión general de cómo interactúa el sistema con usuarios y otros sistemas, mientras que el diagrama de contenedores detalla las aplicaciones, servicios y bases de datos que forman parte de la solución. Esta estructura jerárquica facilita la identificación de dependencias y relaciones entre los distintos elementos del software, promoviendo una mejor comprensión y colaboración en el desarrollo del proyecto. El Modelo C4 es especialmente útil en entornos complejos, donde la claridad en la comunicación y la documentación es esencial para el éxito del desarrollo de software.

## 3.5. Herramientas de diseño y modelado

Enterprise Architect

Enterprise Architect es una herramienta de modelado y diseño de software desarrollada por Sparx Systems. Se utiliza principalmente para la creación de modelos visuales de sistemas de software y arquitecturas empresariales, apoyando prácticas de desarrollo ágil y tradicional. Algunas características destacadas de Enterprise Architect son:

* Modelado Visual: Permite a los desarrolladores crear diagramas UML (Lenguaje de Modelado Unificado), que son representaciones gráficas de sistemas que ayudan a visualizar la estructura y el comportamiento del software.
* Soporte para Múltiples Métodos: Enterprise Architect no solo admite UML, sino también otros lenguajes de modelado, como BPMN (Business Process Model and Notation), SysML (Systems Modeling Language), y ArchiMate. Esto lo convierte en una herramienta versátil para diversas disciplinas de modelado.
* Integración con Herramientas de Desarrollo: Se puede integrar con diversas herramientas de desarrollo y gestión de proyectos, lo que permite una mejor colaboración y seguimiento del progreso en los proyectos de software.
* Generación de Código: Enterprise Architect ofrece capacidades de generación de código a partir de los modelos creados, facilitando la transición del diseño a la implementación.
* Gestión de Requisitos: Incluye funcionalidades para gestionar requisitos y trazabilidad, ayudando a asegurar que el software cumpla con las expectativas de los usuarios y partes interesadas.

StarUML

StarUML es una herramienta de modelado de software de código abierto que se utiliza para crear diagramas UML y otros tipos de modelos visuales. Es conocida por su interfaz intuitiva y sus potentes características de modelado. Algunas de sus características son:

* Interfaz Intuitiva: StarUML ofrece una interfaz de usuario fácil de usar que permite a los desarrolladores crear diagramas rápidamente mediante arrastrar y soltar elementos en el lienzo.
* Soporte para Múltiples Diagramas: Permite crear diversos tipos de diagramas, incluidos diagramas de clases, de secuencia, de casos de uso, de componentes, y más, lo que lo convierte en una herramienta integral para el modelado UML.
* Extensibilidad: StarUML se puede extender mediante plugins, lo que permite a los usuarios agregar nuevas funcionalidades según sus necesidades específicas de modelado.
* Generación de Código: Al igual que Enterprise Architect, StarUML permite la generación de código a partir de los modelos creados, lo que puede acelerar el proceso de desarrollo.
* Colaboración en Equipo: Facilita la colaboración entre equipos, permitiendo compartir y editar proyectos de modelado de manera conjunta, lo cual es esencial en entornos de desarrollo colaborativos.

## 3.6. Entorno de Desarrollo

Visual Studio Code

Visual Studio Code es un editor de código fuente desarrollado por Microsoft, ampliamente utilizado por desarrolladores debido a su flexibilidad, extensibilidad y soporte para múltiples lenguajes de programación. Es una herramienta ligera y potente que combina características de edición de texto con funcionalidades avanzadas para el desarrollo de software. Algunas de sus características destacadas incluyen:

* **Interfaz Intuitiva:** VS Code ofrece una interfaz de usuario limpia y fácil de usar, que permite a los desarrolladores navegar fácilmente entre archivos, carpetas y proyectos. La barra lateral proporciona acceso rápido a la exploración de archivos, búsqueda y control de versiones.
* **Extensibilidad:** Una de las mayores fortalezas de VS Code es su ecosistema de extensiones. Los usuarios pueden personalizar su entorno de desarrollo mediante la instalación de extensiones que añaden soporte para nuevos lenguajes, herramientas de desarrollo, temas y funcionalidades específicas.
* **Soporte Multilenguaje:** VS Code soporta una amplia variedad de lenguajes de programación, incluyendo JavaScript, Python, Java, C#, Go, y muchos más. Proporciona resaltado de sintaxis, autocompletado y navegación de código para mejorar la productividad.
* **Integración de Control de Versiones:** El editor tiene soporte incorporado para sistemas de control de versiones como Git. Los desarrolladores pueden gestionar repositorios, realizar commits, crear ramas y resolver conflictos directamente desde la interfaz de VS Code.
* **Depuración Avanzada:** VS Code incluye herramientas de depuración integradas que permiten a los desarrolladores establecer puntos de interrupción, inspeccionar variables y realizar seguimiento del flujo de ejecución del código. Esto facilita la identificación y corrección de errores en el código.
* **Terminal Integrada:** El editor incluye una terminal de comandos integrada, lo que permite a los desarrolladores ejecutar scripts y comandos sin salir del entorno de desarrollo. Esto mejora la eficiencia al reducir la necesidad de cambiar entre diferentes aplicaciones.
* **Intellisense:** VS Code ofrece autocompletado inteligente mediante su característica Intellisense, que proporciona sugerencias contextuales mientras se escribe código, lo que acelera el proceso de codificación y reduce errores.
* **Colaboración en Tiempo Real:** Con la extensión Live Share, los desarrolladores pueden colaborar en tiempo real con otros en la misma sesión de codificación, permitiendo compartir su entorno de trabajo y colaborar en el mismo código simultáneamente.
* **Integración con Herramientas de Desarrollo:** VS Code se integra fácilmente con herramientas y plataformas populares como Docker, Azure, y otras, lo que permite a los desarrolladores desplegar y gestionar aplicaciones de forma más eficiente.

## 3.7. Sistema de control de Versiones de código (Git)

GitHub

GitHub es una plataforma de desarrollo colaborativo que utiliza el sistema de control de versiones Git. Es ampliamente utilizada por desarrolladores y equipos de trabajo para gestionar proyectos de software, facilitar la colaboración y realizar un seguimiento de los cambios en el código. Algunas características destacadas de GitHub incluyen:

* **Control de Versiones**: GitHub permite a los desarrolladores gestionar el historial de cambios en su código. Esto facilita la colaboración y la restauración de versiones anteriores si es necesario.
* **Colaboración en Equipo**: Ofrece herramientas que facilitan la colaboración entre varios desarrolladores, incluyendo la capacidad de trabajar en ramas separadas y fusionar cambios mediante "pull requests". Los equipos pueden revisar, comentar y discutir cambios antes de integrarlos al proyecto principal.
* **Almacenamiento en la Nube**: GitHub proporciona un espacio de almacenamiento en la nube para los repositorios de código, lo que significa que el código puede ser accedido y gestionado desde cualquier lugar con conexión a Internet.
* **Integración Continua y Despliegue**: Se integra fácilmente con herramientas de integración continua (CI) y despliegue continuo (CD), permitiendo la automatización de pruebas y despliegues, lo que mejora la calidad del software y acelera el ciclo de desarrollo.
* **Documentación y Wiki**: Los proyectos pueden incluir documentación directamente en el repositorio, así como una sección de wiki para información más extensa, lo que ayuda a los nuevos colaboradores a entender el proyecto.
* **Seguimiento de Problemas**: GitHub incluye un sistema de seguimiento de problemas (issues) que permite a los usuarios reportar errores, sugerir mejoras y gestionar tareas pendientes. Esto es útil para mantener el enfoque en la calidad del software.
* **GitHub Actions**: Una herramienta de automatización que permite crear flujos de trabajo personalizados para CI/CD y otras tareas relacionadas con el desarrollo, como la ejecución de pruebas, la construcción de aplicaciones y la implementación de proyectos.
* **Código Abierto y Comunidad**: GitHub alberga una gran cantidad de proyectos de código abierto, lo que fomenta la colaboración y el aprendizaje dentro de la comunidad de desarrolladores. Los usuarios pueden contribuir a proyectos existentes o crear sus propios proyectos.

# **4.REQUERIMIENTOS**

## 4.1 Propósitos

El propósito del proyecto es desarrollar un sistema de información diseñado específicamente para gestionar, controlar y registrar las historias clínicas electrónicas de los pacientes en el contexto del Seguro de Salud Vida Sana. Este software se enfocará en proporcionar una solución sencilla y accesible, que permita a los médicos y administrativos manejar las actividades de atención médica esenciales con facilidad y eficiencia.

Nuestra meta principal es ofrecer una plataforma robusta pero simplificada que facilite la gestión de la información médica sin requerir conocimientos técnicos avanzados. El sistema será intuitivo, permitiendo a los usuarios realizar las operaciones relacionadas con la atención médica, como la gestión de citas y el seguimiento de diagnósticos, de manera rápida y clara.

## 4.2 Ámbitos del Sistema

El ámbito de nuestro Sistema de Información para Gestionar, Controlar y Registrar Historias Clínicas Electrónicas se centra en el desarrollo de una plataforma que facilite la gestión de la atención médica de manera sencilla y eficiente. Este sistema permitirá a los usuarios registrar, controlar y generar informes sobre el estado de salud de los pacientes de forma intuitiva. Las características incluirán el registro de pacientes, gestión de citas, control de diagnósticos y tratamientos, generación de reportes de atención médica.

El sistema se desarrollará utilizando tecnologías adecuadas para asegurar su usabilidad y rendimiento, garantizando un entorno seguro para el manejo de la información médica. El enfoque estará en la simplicidad, permitiendo que incluso usuarios sin experiencia en el manejo de sistemas de salud puedan utilizar la herramienta con facilidad. Además, se contemplará la posibilidad de integración con otros sistemas y herramientas existentes en el ámbito de la salud.

## 4.3 Equipo Scrum

**Scrum Master:** El Scrum Master desempeña el papel de facilitador dentro del marco de trabajo Scrum. Su principal responsabilidad es asegurar que el equipo Scrum comprenda y aplique correctamente los principios y prácticas de este enfoque. Además, se ocupa de eliminar cualquier obstáculo o impedimento que pueda interrumpir el progreso del equipo. El Scrum Master promueve un ambiente de trabajo colaborativo y productivo.

**Product Owner:** El Product Owner actúa como el representante de la voz del cliente o del negocio dentro del equipo Scrum. Su función principal consiste en definir y priorizar el backlog del producto, que es una lista que abarca todas las características, mejoras y tareas necesarias para el desarrollo del proyecto. El Product Owner determina qué funcionalidades deben desarrollarse en primer lugar y trabaja en estrecha colaboración con el equipo de desarrollo para asegurarse de que se entreguen características valiosas en cada iteración.

**Equipo de Desarrollo:** El equipo de desarrollo es responsable de crear el producto o sus incrementos en cada iteración (Sprint). Este equipo se organiza de manera autónoma y es multidisciplinario, lo que significa que cuenta con todas las habilidades necesarias para llevar a cabo el trabajo. Los miembros del equipo colaboran para convertir los elementos del backlog del producto en incrementos de software funcionales. El tamaño óptimo de un equipo Scrum suele estar entre 5 y 9 personas, aunque puede variar según las necesidades del proyecto. En general, los equipos Scrum son pequeños y altamente colaborativos.

## 3.4 Definiciones, Acrónimos y Abreviaturas (Referencias)

**Administrador:** El administrador es el responsable de supervisar y gestionar el funcionamiento general del sistema. Su función incluye la gestión de usuarios, permisos y roles, así como la configuración de las funcionalidades del software. Además, el administrador se encarga de asegurar que el sistema opere de manera eficiente y cumpla con las normativas y procedimientos establecidos.

**Paciente:** El paciente es el usuario final del sistema, encargado de acceder a sus propias historias clínicas relacionadas con su atención médica. Puede solicitar citas, ver sus diagnósticos y tratamientos, así como actualizar su información personal. La plataforma permite al paciente interactuar de manera sencilla y segura con sus datos médicos, garantizando su privacidad y facilidad de uso.

**Médico:** El médico es el profesional encargado de proporcionar atención médica a los pacientes a través del sistema. Su rol incluye la actualización de historias clínicas, la gestión de diagnósticos y la planificación de tratamientos. Además, el médico puede acceder a información relevante sobre sus pacientes y colaborar con otros profesionales de la salud para brindar una atención integral.

**CU:** Casos de Uso

**DB:** Base de Datos

**HU-M:** Historia de Usuario - Móvil

**HU-W:** Historia de Usuario - Web

**PF-M:** Prueba de Funcionalidad - Móvil

**PF-W:** Prueba de Funcionalidad - Web

**RF:** Requisitos Funcionales

**RNF:** Requisitos No Funcionales

**SP:** Sprint

**T:** Tarea

## 3.5 Funciones del Producto (Describir características de funcionalidad)

Funciones de Usuario:

1.1 Permitir a los usuarios autenticarse en el sistema.

1.2 Facilitar el cierre de sesión de los usuarios en el sistema.

1.3 Administrar la creación, actualización, eliminación y consulta de usuarios.

1.4 Permitir la administración de roles para usuarios.

1.5 Administrar los permisos de acceso y operación para los usuarios.

1.6 Registrar y visualizar las actividades de los usuarios en el sistema.

1.7 Permitir la creación de permisos para los usuarios.

Funciones de Paciente:

1.1 Permitir la administración de los datos del paciente.

1.2 Facilitar las especialidades médicas del seguro.

1.3 Permitir gestionar los antecedentes del paciente.

1.4 Administrar el seguro del paciente.

Funciones de Consultas:

1.1 Permitir gestionar las consultas.

1.2 Facilitar la solicitud de citas.

1.3 Permitir la administración de los antecedentes del paciente.

1.4 Permitir ver los antecedentes del paciente.

1.5 Permitir gestionar el tratamiento del paciente.

1.6 Ver los horarios disponibles para las fichas.

Funciones Especialidades:

* 1. Permitir gestionar las Especialidades de la clínica.

1.2 Permitir la asignación de médicos a especialidades.

1.3 Administrar las áreas de las especialidades.

## 3.6 Product Backlog

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **PRODUCT BACKLOG** | | | | |
| **Proyecto** | Sistema de Información Web y Móvil para la gestión de historias clínicas electrónicas para el Seguro de Salud Vida Sana “SSVS” | | | |
| **Product Owner:** | | | | |
| **Fecha de Inicio:** 15/10/2024 | | | | |
| **ID** | **Características y Funcionalidades** | **Tiempo Estimado** | **Estado** | **Sprint** |
| T001 | Introducción | 15 min | Terminado | 0 |
| T002 | Objetivo General | 15 min | Terminado | 0 |
| T003 | Objetivos Específicos | 20 min | Terminado | 0 |
| T004 | Alcance | 30 min | Terminado | 0 |
| T005 | Fundamentos de las historias clínicas HCE | 15 min | Terminado | 0 |
| T006 | Generalidades de las Historias Clínicas Electrónicas | 15 min | Terminado | 0 |
| T007 | Características de las Historias Clínicas Electrónicas en la Nube | 15 min | Terminado | 0 |
| T008 | Conceptos Específicos del Rubro de la Salud Digital | 15 min | Terminado | 0 |
| T009 | ABC de la Gestión de Historia Clínica | 15 min | Terminado | 0 |
| T010 | Marco de trabajo Ágil SCRUM en el contexto del Sistema del Salud | 15 min | Terminado | 0 |
| T011 | Lenguaje de Programación Web y Móvil | 5 min | Terminado | 0 |
| T012 | Frameworks de entorno y ejecucuion | 5 min | Terminado | 0 |
| T013 | Gestor de Base de Datos | 5 min | Terminado | 0 |
| T014 | Sistema de Control de Versiones de Código | 5 min | Terminado | 0 |
| T015 | Propósitos | 10 min | Terminado | 0 |
| T016 | Ámbitos del Sistema | 10 min | Terminado | 0 |
| T017 | Equipo SCRUM | 5 min | Terminado | 0 |
| T018 | Product Backlog | 25 min | Terminado | 0 |
| T019 | Requisitos Funcionales | 10 min | Terminado | 0 |
| T020 | Requisitos no Funcionales | 10 min | Terminado | 0 |
| T021 | Lista de Casos de Uso | 20 min | Terminado | 0 |
| T022 | Paquetes y Casos de uso | 30 min | Terminado | 0 |
| T023 | Planificación de Sprint (Diagrama de Gantt) | 1 h | Terminado | 0 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **PRODUCT BACKLOG** | | | | | |
| **Nro. de SPRINT: 1** | Tiempo Programado: 12 días | | | | |
| **Objetivos:** Desarrollar las funcionalidades básicas del software de historial clínico electrónico | | | | | |
| **Fecha de Inicio:** 15/10/2024 | | | **Fecha de Finalizacion:**27/10/2024 | | |
| **ID** | **Características y Funcionalidades** | **Tiempo Estimado** | | **Estado** | **Sprint** |
| SP1-T001 | Implementación del backend de Inicio de Sesión | 15 min | | Terminado | 0 |
| SP1-T002 | Implementación del Frontend de Inicio de Sesión | 15 min | | Terminado | 0 |
| SP1-T003 | Implementación del backend de Cierre de Sesión | 20 min | | Terminado | 0 |
| SP1-T004 | Implementación del Frontend de Cierre de Sesión | 30 min | | Terminado | 0 |
| SP1-T005 | Implementación del backend de Editar perfil | 15 min | | Terminado | 0 |
| SP1-T006 | Implementación del Frontend de Editar perfil | 15 min | | Terminado | 0 |
| SP1-T007 | Implementación del backend de Roles (Sprint Security) | 15 min | | Terminado | 0 |
| SP1-T008 | Implementación del backend de Permisos (Sprint Security) | 15 min | | Terminado | 0 |
| SP1-T009 | Implementación del Frontend de Roles (Sprint Security) | 15 min | | Terminado | 0 |
| SP1-T010 | Implementación del frontend de Permisos (Sprint Security) | 15 min | | Terminado | 0 |
| SP1-T011 | Conexión a la Base de Datos | 5 min | | Terminado | 0 |
| SP1-T012 | Implementación en Móvil de Inicio de Sesión | 5 min | | Terminado | 0 |
| SP1-T013 | Implementación en Móvil de Cierre de Sesión | 5 min | | Terminado | 0 |
| SP1-T014 | Implementación en Móvil de Editar perfil | 5 min | | Terminado | 0 |
| SP1-T015 |  | 10 min | | Terminado | 0 |
| SP1-T016 |  | 10 min | | Terminado | 0 |
| SP1-T017 |  | 5 min | | Terminado | 0 |
| SP1-T018 |  | 25 min | | Terminado | 0 |
| T019 |  | 10 min | | Terminado | 0 |
| T020 |  | 10 min | | Terminado | 0 |
| T021 |  | 20 min | | Terminado | 0 |
| T022 |  | 30 min | | Terminado | 0 |
| T023 |  | 1 h | | Terminado | 0 |

## 3.7 Requisitos Funcionales

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **FUNCIONALIDAD** | **WEB** | **MOVIL** |
| 1 | inicio de sesión (Web y móvil) | X | X |
| 2 | cierre de sesión (Web y móvil) | X | X |
| 3 | Gestionar Usuario (Web y móvil) | X | X |
| 4 | Gestionar roles (Web) | X |  |
| 5 | Gestionar Permisos (Web) | X |  |
| 6 | Gestionar Médicos (Web y móvil) | X | X |
| 7 | Gestionar Pacientes (Web y móvil) | X | X |
| 8 | Gestión de citas (Web y Móvil) | X | X |
| 9 | Administración de Horarios (Web) | X |  |
| 10 | Gestionar Antecedentes (Web y Móvil) | X | X |
| 11 | Gestionar Diagnostico (Web y Móvil) | X | X |
| 12 | Administrar Tratamiento (Web y Móvil) | X | X |
| 13 | Administración de Especialidades (Web y Móvil) | X | X |
| 14 | Gestión de Seguro Médico (Web y Móvil) | X | X |
| 15 | Administración de Bitácora (Web) | X |  |
| 16 | Gestión de Reportes (Web) | X |  |

## 3.8 Requisitos No Funcionales

**Almacenamiento**: La gestión eficiente de almacenamiento de datos es fundamental para garantizar la disponibilidad y confiabilidad de la información contable, mediante un sistema de almacenamiento que sea seguro y escalable.

**Rendimiento**: El sistema debe funcionar de forma altamente eficiente para ofrecer una experiencia fluida. Se busca que tanto la interfaz web como la aplicación móvil carguen de manera ágil, evitando demoras que puedan frustrar a los usuarios debido a tiempos de carga prolongados.

**Escalabilidad**: El sistema debe ser escalable para soportar un mayor volumen de usuarios y transacciones, particularmente en épocas de alta demanda contable o fiscal. Esto asegurará un rendimiento ágil y rápido conforme crezca la cantidad de usuarios.

**Compatibilidad**: La aplicación web debe ser compatible con los navegadores más utilizados, permitiendo acceso desde diversos dispositivos y sistemas operativos, lo que mejora la accesibilidad y facilita su uso.

**Interfaz de Usuario (UI)**: La interfaz debe ser intuitiva y fácil de navegar, con un diseño limpio y eficiente que facilite las tareas contables para que el usuario pueda trabajar sin dificultades.

**Experiencia del Usuario (UX)**: La aplicación móvil debe brindar una experiencia sobresaliente, permitiendo que las operaciones contables se realicen de forma rápida y sencilla, lo cual contribuirá a la satisfacción y lealtad de los usuarios.

**Tiempo de Respuesta de la Aplicación Móvil**: La aplicación móvil debe responder casi de inmediato a las acciones de los usuarios, con tiempos de carga y procesamiento rápidos para maximizar la satisfacción del usuario.

**Cumplimiento Normativo y Privacidad**: La plataforma deberá cumplir con todas las regulaciones aplicables en materia de privacidad de datos y otras normas pertinentes para el ámbito contable.

**Seguridad Contra Amenazas y Ataques**: Se deberán implementar medidas de seguridad avanzadas para proteger la plataforma frente a amenazas y ataques cibernéticos, manteniendo la integridad y continuidad del servicio.

**Mantenibilidad**: El código debe estar diseñado para facilitar el mantenimiento y futuras actualizaciones, permitiendo una expansión con mínima interrupción en los servicios.

**Documentación**: Se debe ofrecer documentación completa tanto para desarrolladores como para usuarios, lo cual facilitará el uso y comprensión del sistema.

**Pruebas de Rendimiento**: Es necesario realizar pruebas de carga que aseguren que el sistema puede manejar grandes volúmenes de tráfico y transacciones sin pérdida de rendimiento.

**Soporte Técnico**: Se debe ofrecer un soporte técnico eficiente para asistir a los usuarios ante consultas o inconvenientes.

**Disponibilidad**: La plataforma debe estar operativa de manera continua, con un objetivo de disponibilidad del 99%. Esto permitirá que los usuarios accedan a sus datos y realicen transacciones en cualquier momento, algo crucial para la gestión diaria en contabilidad.

**Accesibilidad**: La aplicación debe ser inclusiva, cumpliendo con los estándares de accesibilidad web para permitir su uso por personas con discapacidades.

**Internacionalización y Localización**: La plataforma deberá estar disponible en varios idiomas y monedas, lo que permitirá su uso en diferentes países y facilitará la expansión global.

**Integración con Servicios Externos**: El sistema debe permitir la integración con servicios de terceros, como pasarelas de pago, para ampliar sus funcionalidades y cubrir más necesidades de los usuarios.

**Consumo de Recursos**: La plataforma debe gestionar de forma eficiente el uso de recursos del servidor, optimizando costos y garantizando un rendimiento estable.

## 3.9 Lista de Casos de Uso

|  |  |  |
| --- | --- | --- |
| **UC** | **FUNCIONALIDAD** | **SPRINT** |
| UC01 | inicio de sesión (Web y móvil) | 1 |
| UC02 | cierre de sesión (Web y móvil) | 1 |
| UC03 | Gestionar Usuario (Web y móvil) | 1 |
| UC04 | Gestionar roles (Web) | 1 |
| UC05 | Gestionar Permisos (Web) | 1 |
| UC06 | Gestionar Médicos (Web y móvil) | 2 |
| UC07 | Gestionar Pacientes (Web y móvil) | 2 |
| UC08 | Gestión de citas (Web y Móvil) | 2 |
| UC09 | Administración de Horarios (Web) | 2 |
| UC10 | Gestionar Antecedentes (Web y Móvil) | 3 |
| UC11 | Gestionar Diagnostico (Web y Móvil) | 3 |
| UC012 | Administrar Tratamiento (Web y Móvil) | 3 |
| UC13 | Administración de Especialidades (Web y Móvil) | 2 |
| UC14 | Gestión de Seguro Médico (Web y Móvil) | 2 |
| UC15 | Administración de Bitácora (Web) | 3 |
| UC16 | Gestión de Reportes (Web) | 3 |

## 3.10 Paquetes y casos de uso

Administración de Usuario

UC01 inicio de sesión (Web y móvil) 1

UC02 cierre de sesión (Web y móvil) 1

UC03 Gestionar Usuario (Web y móvil) 1

UC04 Gestionar roles (Web) 1

UC05 Gestionar Permisos (Web) 1

UC15 Administración de Bitácora (Web) 3

-----------------------------------------------

Administración de Pacientes

UC07 Gestionar Pacientes (Web y móvil) 2

UC14 Gestión de Seguro Médico (Web y Móvil) 2

UC10 Gestionar Antecedentes (Web y Móvil) 3

------------------------------------------------

Gestión de Consultas

UC08 Gestión de citas (Web y Móvil) 2

UC09 Administración de Horarios (Web) 2

UC11 Gestionar Diagnostico (Web y Móvil) 3

UC012 Administrar Tratamiento (Web y Móvil) 3

------------------------------------------------

Administración de Especialidades

UC06 Gestionar Médicos (Web y móvil) 2

UC13 Administración de Especialidades (Web y Móvil) 2

UC16 Gestión de Reportes (Web) 3

## 3.11 Planificación Sprint (Diagrama de Gantt)

# **5.PROCESO DE DESARROLLO DE SOFTWARE**

# **Sprint 1**

## 1.Sprint Planning

## 1.1 Objetivos del Sprint

## 1.2 Historia de usuario (tarjetas 3C, Planning Poker, prototipos)

## 1.3 Contexto del sistema

## 1.4 Sprint backlog

## 1.5 Equipo SCRUM