Instituto Tecnológico de Costa Rica

Centro Académico de Alajuela

IC-4700: Lenguajes De Programación

Tarea Programada 02:

Programación orientada a objetos con Java

Estudiantes:

William Gerardo Alfaro Quirós – 2022437996

Jesús Gabriel Cordero Diaz - 2020081049

Profesora:

María Auxiliadora Mora Cross

Fecha de entrega:

09/10/23

II Semestre

Tabla de Contenidos

Descripción del Problema

Para la tarea programada se pretende implementar el desarrollo de un sistema de información para el área de Medicina General del Hospital México. Particularmente, se enfoca en la experiencia de usuario administrador, el cual tiene privilegios de acceder a los datos pertinentes tanto al personal médico como a los pacientes. Consecuentemente, el usuario podrá desplegar, editar, agregar, y borrar datos de médicos, pacientes y sus expedientes respectivos. Adicionalmente, se cuenta con una vista que permite obtener información relevante sobre padecimientos del paciente, sus medicamentos y el seguimiento de su médico general y/o especialista.

Los requerimientos del sistema son los siguientes:

- Todas las personas en el sistema tienen asociada información como: nombre, apellido, cédula, correo electrónico. Las personas pueden ser:
    - Médicos con diversas especialidades: estos tienen asociado un número identificador además de todos los datos asociados a la persona. Los médicos pueden ser generales (médicos de cabeceras) o especialistas.
    - Pacientes: cada paciente tiene asignado un médico de cabecera (Medicina General).
- Cada paciente tiene un expediente médico, en el que se ingresa información como fecha, padecimiento, procedimiento realizado y medicamentos.
- El sistema debe ser desarrollado utilizando las siguientes tecnologías:
    - Java
    - Spring Boot
    - Hibernate
    - Una base de datos relacional
- Cada paciente tiene asociada una agenda. La agenda del paciente muestra un listado de sus citas, con fecha, hora y nombre del médico.
- Los pacientes pueden solicitar citas de Medicina General, pero para obtener una cita con un especialista, debe ser referido por su médico de cabecera.
- Un paciente puede tener más de una referencia a especialistas y más de una cita a medicina general.

- Cada cita tiene una duración de 30 minutos, un médico y un paciente asociados, así como fecha y hora.
- Durante la cita el médico actualiza el expediente del paciente con datos como fecha, padecimiento, procedimiento realizado y medicamentos referidos.
- Se requieren al menos los siguientes informes:
    - Despliegue la agenda de un médico que muestra un listado de sus citas, con fecha, hora y paciente.
    - Despliegue el expediente de un paciente.
    - Despliegue la agenda de un paciente.
    - Liste todos los pacientes y sus citas.
    - Liste todos los médicos y su especialidad.

Descripción de la Solución

Para la segunda tarea programada, se deben tomar en cuenta los aspectos solicitados en la consigna; por lo tanto, la solución se centra en cómo se implementan las tecnologías en la aplicación. La solución propuesta para abordar este problema es un sistema de información web desarrollado en Java, haciendo uso de Spring Boot, Hibernate y una base de datos relacional elegida específicamente para este proyecto. Este sistema cumple con todos los requerimientos detallados en la delimitación del problema, y aprovecha las siguientes tecnologías y patrones de diseño:

Tecnologías utilizadas:

- **Java:** Este lenguaje de programación es la columna vertebral de la aplicación, proporcionando la robustez y flexibilidad necesarias para implementar todas las funcionalidades requeridas.
- **Spring Boot:** Como framework, Spring Boot simplifica el desarrollo de las aplicaciones web en Java, facilitando la configuración y gestión de componentes esenciales como controladores, servicios y seguridad.
- **Hibernate:** Hibernate, una herramienta de mapeo objeto-relacional (ORM), se encarga de la persistencia de datos en la base de datos relacional. Esto simplifica la interacción con la base de datos y permite que los objetos Java se mapeen directamente a tablas de la base de datos.

- **Base de datos relacional:** Se utiliza una base de datos relacional implementada en MySQL para almacenar y gestionar la información del sistema de manera estructurada y eficiente.

Patrón de diseño Modelo-Vista-Controlador (MVC):

- **Modelo:** Representa la capa de datos y lógica de negocio. En este caso, incluye las clases que definen médicos, pacientes, expedientes médicos, citas y referencias a especialistas.
- **Vista:** Representa la capa de presentación, que incluye las páginas web y formularios utilizados por los usuarios administradores para interactuar con el sistema.
- **Controlador:** Gestiona las solicitudes de los usuarios y coordina las acciones entre la vista y el modelo. En este contexto, los controladores se encargan de recibir las peticiones del usuario (como agregar un médico o mostrar la agenda de un paciente), procesar la lógica de negocio correspondiente y actualizar la vista según sea necesario.

Diagramas UML

Para implementar los requerimientos de este sistema de información, se pueden utilizar diagramas UML para visualizar la estructura y relaciones entre los elementos clave del sistema. Por tanto, se crea un diagrama de clases UML para representar las clases como Médico, Paciente, Expediente, Cita, y Usuario Abs (usuario abstracto), con sus atributos y relaciones. Además, se utilizan diagramas UML de uso de casos para modelar los casos de uso entre el usuario administrador y el sistema en escenarios como la solicitud de citas, la actualización de expedientes médicos y la generación de informes. Los diagramas de casos de uso UML también serían útiles para identificar y describir las diferentes funcionalidades que los usuarios pueden realizar en el sistema, como gestionar médicos, pacientes, citas y expedientes médicos. Estos diagramas UML resultan en una valiosa una guía visual durante el desarrollo y la implementación del sistema, ayudando a garantizar que se cumplan los requisitos y se mantenga la coherencia en la estructura y el comportamiento del sistema.

Diagramas de Casos de Uso

Diagrama de agenda de médico

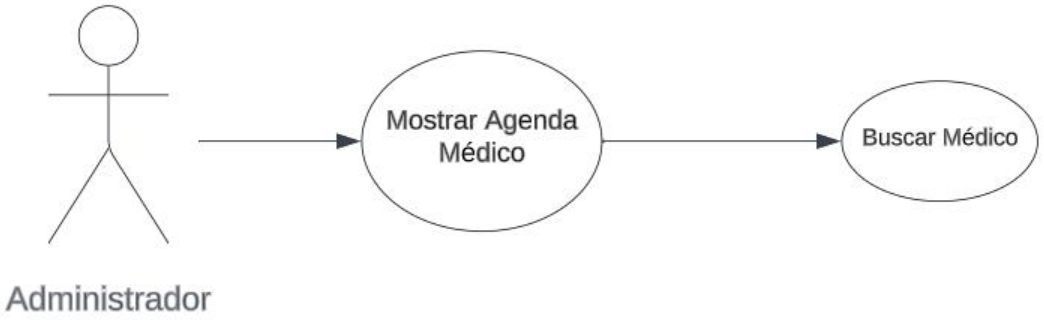


Diagrama de expediente del paciente

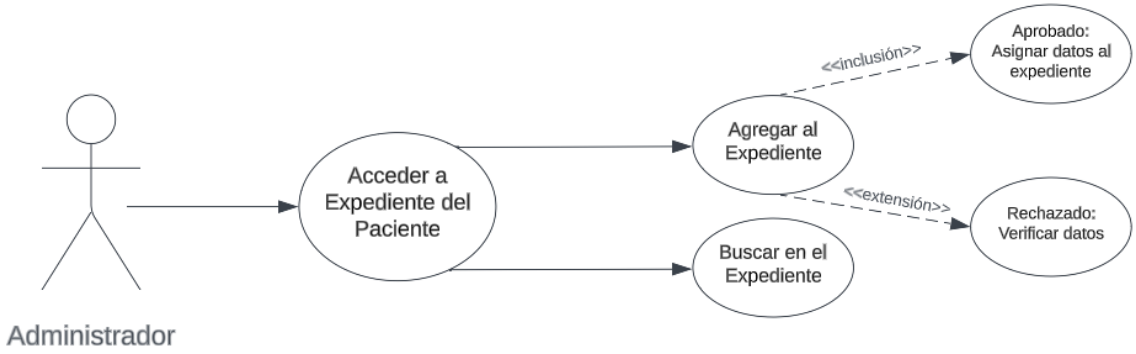


Diagrama de agenda de paciente

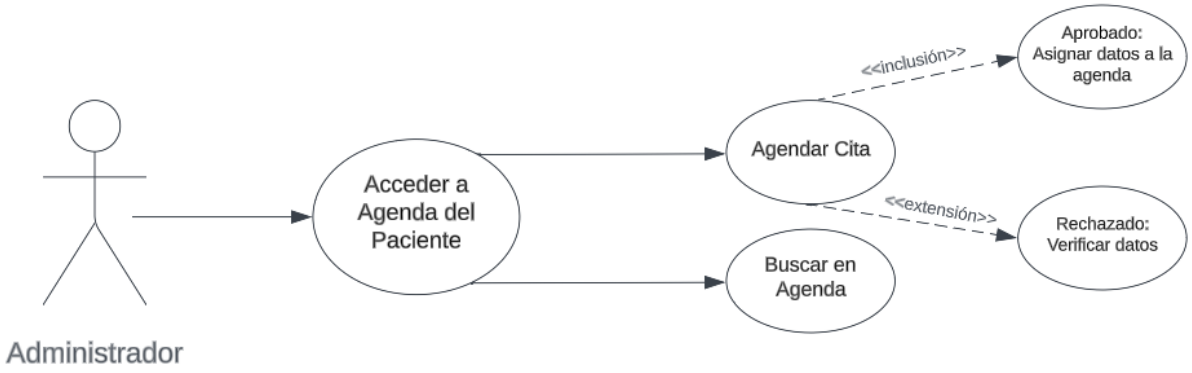


Diagrama de listado de pacientes

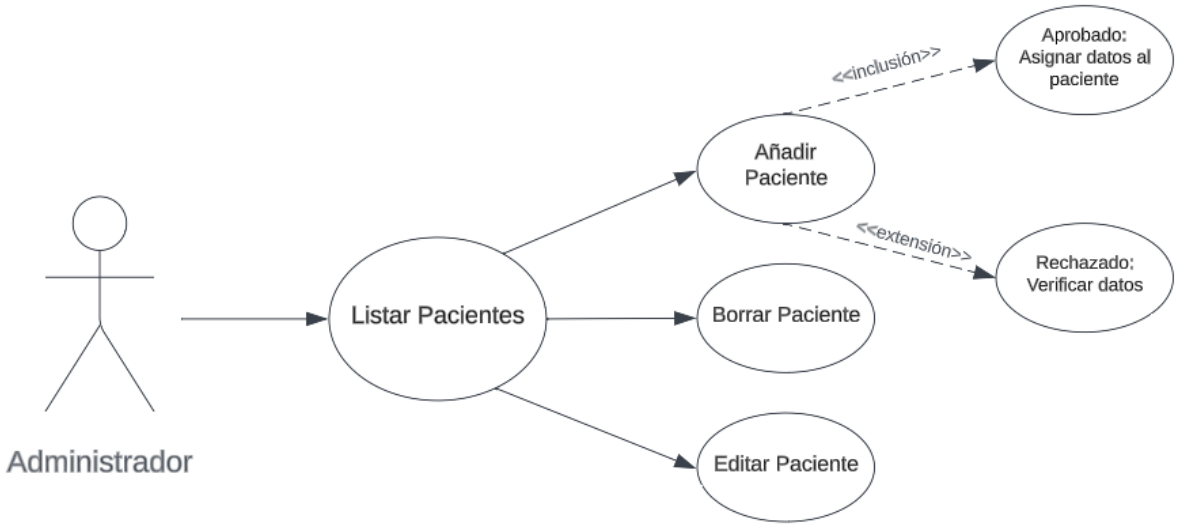


Diagrama de listado de médicos

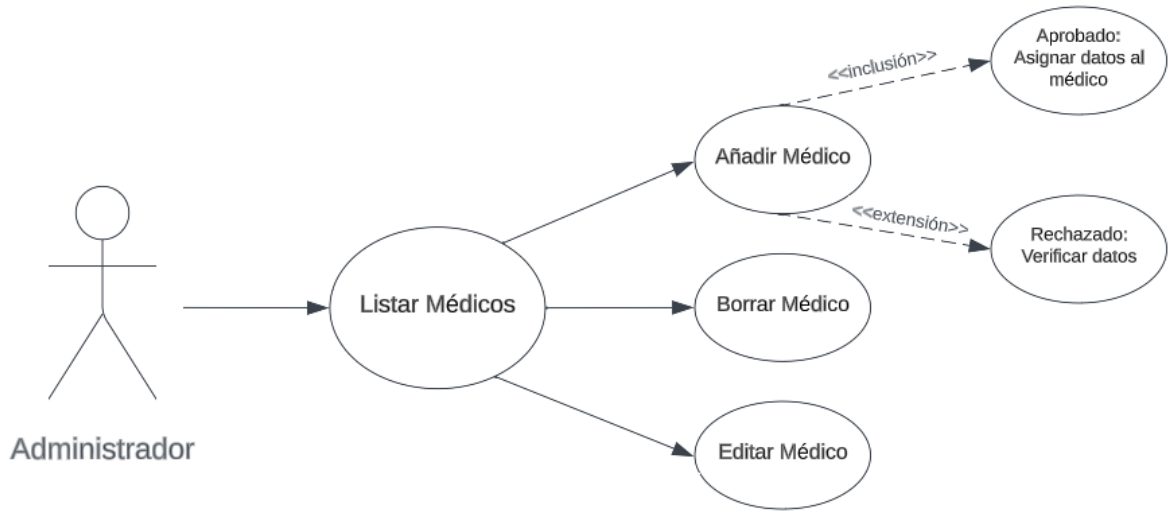


Diagrama de Clases

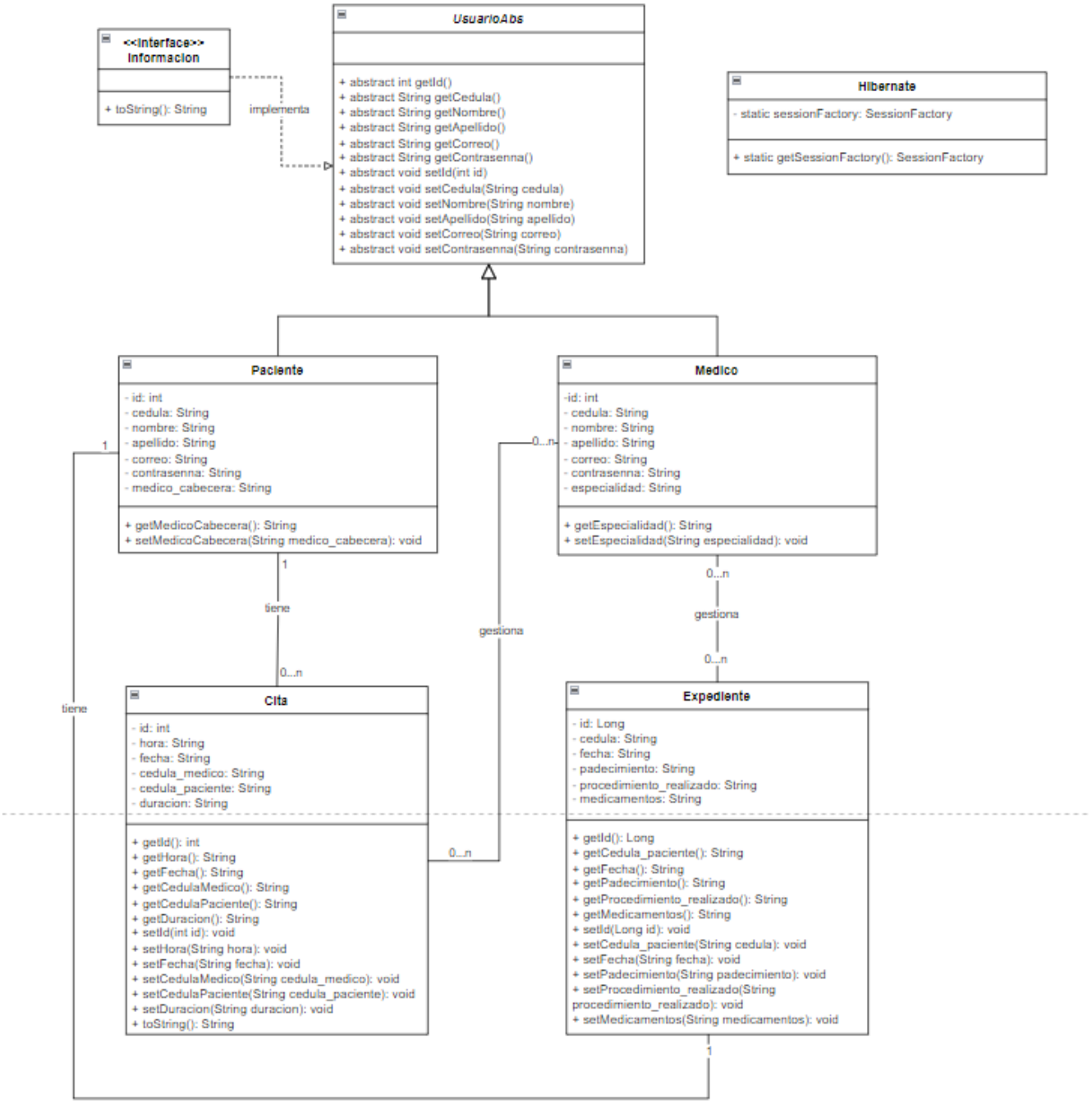
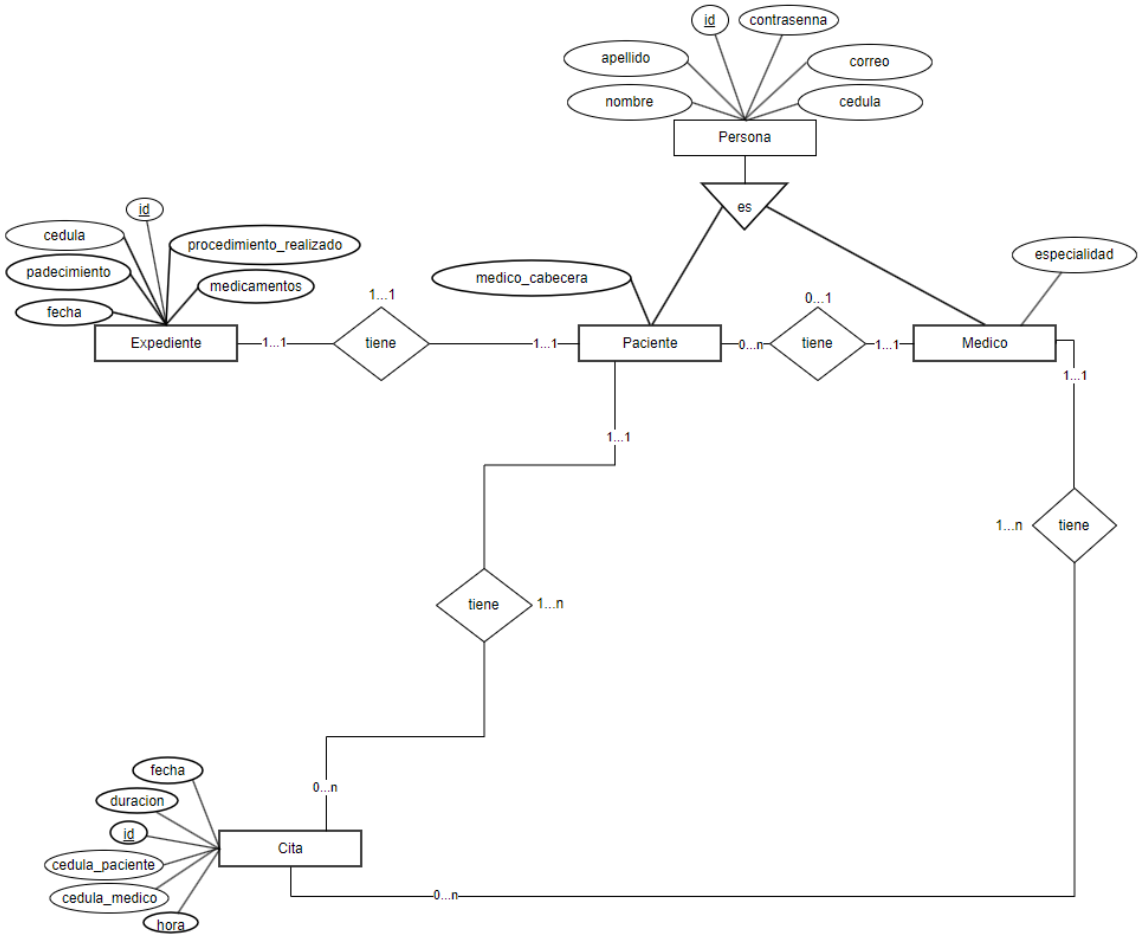


Diagrama Entidad-Relación de la Base de Datos



Descripción de Casos de Uso

Agenda de médico

Caso de Uso: Mostrar agenda de médico	
Actor Primario:	Usuario administrador
Actor Secundario:	Médico
Pre-Condiciones:	El usuario debe estar registrado y autenticado en el sistema. *Tomando en cuenta que se verifican las credenciales desde el sistema operativo Windows en el usuario correspondiente.
Post-Condiciones	El usuario queda asociado al evento
Flujo Principal:	1. El usuario selecciona la opción “Agenda de médico” en el menú principal 2. El usuario ingresa el número de cedula del médico que desea consultar

	3. El usuario selecciona “Buscar” 4. El sistema muestra la agenda del médico asociado a la cedula ingresada
Flujos Alternos:	2. (b) El sistema indica que el número de cedula del médico ingresado no está asociado a ningún médico registrado en el sistema

Expediente del paciente

Caso de Uso: Acceder al expediente del paciente	
Actor Primario:	Usuario administrador
Actor Secundario:	Médico
Pre-Condiciones:	El usuario debe estar registrado y autenticado en el sistema. *Tomando en cuenta que se verifican las credenciales desde el sistema operativo Windows en el usuario correspondiente.
Post-Condiciones	El usuario queda asociado al evento
Flujo Principal:	1. El usuario selecciona la opción “Expediente del paciente” en el menú principal 2. El usuario ingresa el número de cedula del paciente al que quiere agregar una entrada o consultar 3. (a) El usuario selecciona “Agregar a expediente” 3. (a) El sistema muestra un submenú para la creación de una entrada al expediente asociados al número de cedula del paciente 4. El usuario selecciona “Buscar” 4. (a) El sistema muestra un submenú con la información respectiva de todas las entradas en el expediente del paciente asociado al número de cedula
Flujos Alternos:	2. (b) El sistema indica que el número de cedula del paciente ingresado no está asociado a ningún médico registrado en el sistema

Agenda del paciente

Caso de Uso: Acceder a agenda del paciente	
Actor Primario:	Usuario administrador
Actor Secundario:	Médico
Pre-Condiciones:	El usuario debe estar registrado y autenticado en el sistema. *Tomando en cuenta que se verifican las credenciales desde el sistema operativo Windows en el usuario correspondiente.
Post-Condiciones	El usuario queda asociado al evento
Flujo Principal:	1. El usuario selecciona la opción “Agenda del paciente” en el menú principal 2. El usuario ingresa el número de cedula del paciente al que quiere agregar una cita al paciente o consultar 3. (a) El usuario selecciona “Agendar Cita” 3. (b) El sistema muestra un submenú para la creación de una entrada a la agenda asociados al número de cedula del paciente 4. (a) El usuario selecciona “Buscar” 4. (b) El sistema muestra un submenú con la información respectiva de todas las entradas en la agenda del paciente asociado al número de cedula
Flujos Alternos:	2. (b) El sistema indica que el número de cedula del paciente ingresado no está asociado a ningún médico registrado en el sistema

Listado de pacientes

Caso de Uso: Listar pacientes	
Actor Primario:	Usuario administrador
Actor Secundario:	Médico

Pre-Condiciones:	El usuario debe estar registrado y autenticado en el sistema. *Tomando en cuenta que se verifican las credenciales desde el sistema operativo Windows en el usuario correspondiente.
Post-Condiciones	El usuario queda asociado al evento
Flujo Principal:	1. El usuario selecciona la opción de “Lista de pacientes” en el menú principal 2. El sistema despliega todos los pacientes registrados en la base de datos con su información respectiva 3. El usuario selecciona “Añadir” 3. (a) El sistema muestra un submenú para la creación de una entrada al sistema añadiendo un nuevo paciente 4. El usuario selecciona “Borrar” 4. (a) El sistema muestra un submenú para la confirmación de la eliminación del paciente asociado con a su cédula 5. El usuario selecciona “Editar” 5. (a) El sistema muestra un submenú para la edición de un paciente registrado en la lista de pacientes
Flujos Alternos:	2. (b) El sistema indica que no existen pacientes en el sistema

Listado de médicos

Caso de Uso: Listar médicos	
Actor Primario:	Usuario administrador
Actor Secundario:	Médico
Pre-Condiciones:	El usuario debe estar registrado y autenticado en el sistema. *Tomando en cuenta que se verifican las credenciales desde el sistema operativo Windows en el usuario correspondiente.
Post-Condiciones	El usuario queda asociado al evento
Flujo Principal:	1. El usuario selecciona la opción de “Lista de médicos” en el menú principal 2. El sistema despliega todos los médicos registrados en la base de datos con su información respectiva

	3. El usuario selecciona “Añadir” 3. (a) El sistema muestra un submenú para la creación de una entrada al sistema añadiendo un nuevo médico 4. El usuario selecciona “Borrar” 4. (a) El sistema muestra un submenú para la confirmación de la eliminación del médico asociado con su cédula 5. El usuario selecciona “Editar” 5. (a) El sistema muestra un submenú para la edición de un médico registrado en la lista de médicos
Flujos Alternos:	2. (b) El sistema indica que no existen médicos en el sistema

Análisis de Resultados

Objetivos Alcanzados

Implementar un Sistema de Control de Citas Médicas

- Uno de los logros más destacados del proyecto es la implementación de un sistema de información diseñado para el área de Medicina General del Hospital México. Este sistema tiene la capacidad de gestionar eficazmente las citas de los pacientes, mejorando la atención médica en el hospital y la experiencia del usuario.

Aplicar Metodologías de Análisis y Diseño

- Se aplicaron metodologías de análisis y diseño de sistemas, lo que permitió un enfoque estructurado en la planificación y desarrollo del proyecto. Se consideraron las necesidades del área de Consulta Externa del hospital y se diseñó una solución modular que facilita la escalabilidad y el mantenimiento del sistema.

Utilizar Conceptos de Programación Orientada a Objetos

- El proyecto aprovechó plenamente los conceptos de programación orientada a objetos, como la herencia de clases, interfaces, polimorfismo y clases abstractas. Estos elementos se utilizaron para modelar adecuadamente las entidades del sistema y establecer relaciones efectivas entre ellas.

Implementar Persistencia de Datos con Hibernate

- Se logró la persistencia de datos en una base de datos relacional utilizando Hibernate, una herramienta de mapeo objeto-relacional. Esto garantiza que la información crítica, como los expedientes médicos y las agendas de los pacientes, se almacene de manera segura y esté disponible cuando sea necesario.

Cumplir de Requerimientos de Programación en Java

- El proyecto siguió rigurosamente los requerimientos de programación en Java, lo que incluyó el uso de modificadores de visibilidad, variables y métodos de clase, herencia, clases abstractas, interfaces, polimorfismo por sobrecarga de métodos y clases genéricas. Esto demuestra un sólido dominio de la programación en Java.

Aplicar un Patrón Modelo-Vista-Controlador (MVC)

- El sistema se desarrolló como una aplicación web, lo que proporciona una interfaz amigable y accesible. Además, se aplicó el patrón de diseño Modelo-Vista-Controlador (MVC) para separar de manera eficiente la lógica de negocios de la presentación y la interacción del usuario.

Crear Documentación Integral

- El proyecto se diseñó con una documentación interna y externa exhaustiva. La documentación interna incluyó comentarios descriptivos para clases, atributos y métodos, mientras que la documentación externa abarcó aspectos como la descripción del problema, la solución propuesta, diagramas UML, análisis de resultados y conclusiones personales.

Trabajar en Equipo

- El proyecto se desarrolló en grupos de máximo dos personas, lo que permitió colaborar eficazmente y aprender a trabajar en equipo en un entorno de desarrollo de software.

Objetivos No Alcanzados

Implementación de Seguridad:

- Aunque la gestión de datos de manera segura no era requisito para este proyecto, no se implementó un nivel de seguridad avanzado, como autenticación de usuarios, roles y permisos. Esto podría ser crucial para proteger la confidencialidad de los datos médicos de los pacientes.

Escalabilidad y Rendimiento en Grandes Volúmenes de Datos:

- El proyecto no abordó completamente la escalabilidad y el rendimiento del sistema cuando se enfrenta a grandes volúmenes de datos y una alta demanda de citas médicas. La optimización y el manejo eficiente de la base de datos podrían no haberse explorado a fondo.

Conclusiones Personales

Conclusión de William Alfaro

Este proyecto, que abarcó la implementación de un sistema de control de citas médicas utilizando Spring Boot, Bootstrap, Hibernate y MySQL, se convirtió en una experiencia profundamente enriquecedora y repleta de desafíos que me permitieron aprender y crecer en mi trayectoria de desarrollo de software. En las etapas iniciales, me encontré con la complejidad de configurar Spring Boot y entender su arquitectura subyacente. Este obstáculo me impulsó a profundizar en la documentación oficial y a explorar recursos en línea, lo que resultó ser un aprendizaje valioso en la gestión de configuraciones complejas en proyectos similares.

Por un lado, nos enfrentamos a dificultades relacionadas con la gestión de dependencias y la inyección de dependencias en Spring Boot. Esta etapa me enseñó la importancia de utilizar herramientas como Maven para administrar eficazmente las dependencias, garantizando así la coherencia en el proyecto. Por el otro lado, adquirí una comprensión sólida de los conceptos esenciales de la inyección de dependencias y cómo aplicar anotaciones como **@Autowired** de manera efectiva. Estas lecciones subrayaron la necesidad de una base sólida en programación orientada a objetos y un enfoque meticuloso en la gestión de componentes y servicios.

En el aspecto de diseño de interfaz de usuario, Bootstrap representó un desafío apasionante. La creación de un diseño totalmente responsivo y personalizado se convirtió en un objetivo fundamental. Aprendí a utilizar las clases proporcionadas por Bootstrap para construir diseños adaptables y a sobrescribir los estilos CSS para lograr una personalización precisa de los componentes. Garantizar la compatibilidad con diversos navegadores se convirtió en un ejercicio esencial, y adquirí experiencia en realizar pruebas cruzadas y aplicar soluciones específicas para asegurar una experiencia uniforme en diferentes plataformas y navegadores. La perspicacia de estas dificultades demostró la importancia de las habilidades de diseño y la adaptación ágil en el entorno del desarrollo web.

Conclusión de Jesus Cordero

El proceso de desarrollo de este proyecto, basado en Spring Boot, Bootstrap, Hibernate y MySQL, se transformó en una experiencia profundamente enriquecedora que amplió mis horizontes y mejoró mis habilidades en el ámbito de la programación y el desarrollo de software. Durante la fase de implementación con Hibernate, me encontré con el desafío de mapear objetos a la base de datos y optimizar las consultas SQL. Aprendí a utilizar de manera efectiva anotaciones fundamentales como **@Entity** y a aplicar estrategias avanzadas de optimización de consultas, lo que mejoró significativamente la eficiencia y la velocidad de las operaciones de base de datos.

En lo que respecta a MySQL, la configuración inicial y el modelado de datos se revelaron como desafíos iniciales. Aprendí a configurar la conexión de la base de datos utilizando archivos de configuración específicos de Spring Boot y a mantener una metodología iterativa para el diseño de la base de datos, garantizando que se alineara adecuadamente con las entidades definidas en Hibernate. La gestión de errores y excepciones relacionadas con la base de datos se convirtió en un componente crucial de la implementación. A lo largo de este proceso, aprendí a manejar de manera efectiva las excepciones de Hibernate y a implementar estrategias de manejo de errores personalizadas para garantizar la consistencia de los datos y una experiencia de usuario confiable.

En conjunto, estas dificultades resaltaron la importancia de una planificación cuidadosa, la práctica constante y la adaptación continua en el proceso de desarrollo de software. Me enseñaron a abordar obstáculos de manera sistemática, a buscar soluciones a través de la investigación y a colaborar de manera efectiva en equipos multidisciplinarios. En resumen, este proyecto no solo me proporcionó una base técnica sólida, sino que también me equipó con habilidades esenciales para abordar futuros desafíos en el emocionante y en constante evolución mundo del desarrollo de software.