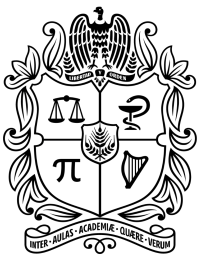
Taller requerimientos y clean code para proyecto
Ingeniería de software 1



UNIVERSIDAD
NACIONAL
DE COLOMBIA

Autores:
John Jairo Paez Albino
Nicolas Arturo Jaramillo Garrido
Juan Diego Velásquez Pinzón
Juan David Suarez Coronado

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Ingeniería de Software 1 (2016701)
31 de enero 2025

Punto 2

La idea de crear una aplicación para gestionar citas odontológicas nació de algo que todos hemos vivido: agendar una cita con el dentista puede ser un verdadero dolor de cabeza.

Problemas como llamar y no recibir contestación, envíar un mensaje por WhatsApp y esperar horas por respuesta, o simplemente no hay horarios disponibles. Al final muchos terminan posponiendo su consulta, lo que no solo afecta a los pacientes sino también a los odontólogos que podrían aprovechar mejor su tiempo y evitar cancelaciones con un sistema más ágil.

La idea empezó cuando, hablando con familiares y amigos, nos dimos cuenta de que este problema era más común de lo que pensábamos. Fue entonces cuando nos preguntamos: ¿y si pudiéramos hacer este proceso más fácil para todos?.

En nuestras primeras reuniones, cada uno del equipo propuso ideas diferentes. Hablamos de crear una app para agendar citas médicas en general o incluso una plataforma para gestionar inventarios en pequeñas tiendas. Pero al final nos quedamos con la gestión de citas odontológicas porque vimos una necesidad real y un mercado que podría mejorar muchísimo.

La problemática general es: como paciente, se busca una cita para odontología y se encuentra con que no hay disponibilidad, o toca esperar días para que confirmen; también puede ocurrir olvidarse de recordar la cita sin un aviso. Por otro lado, los odontólogos también tienen sus dolores de cabeza: cancelaciones de último momento, agendas desorganizadas y el tener que confirmar cada cita manualmente.

Nuestra aplicación quiere ser la solución a todo esto con un sistema de reserva en línea, recordatorios automáticos y la opción de reprogramar o cancelar citas de manera sencilla. Además, nos enfocamos en consultorios pequeños y medianos, ofreciendo una interfaz simple y accesible, sin costos exagerados.

Para los pacientes, la idea es que agendar una cita sea rápido y sin complicaciones, o si surge un imprevisto, que puedan modificar su cita sin tener que llamar o enviar mil mensajes. Para los odontólogos, queremos un sistema que les ayude a organizar su agenda sin esfuerzo reduciendo cancelaciones y optimizando su tiempo.

Para cumplir con nuestros objetivos, la aplicación incluirá:

- **Agendamiento de citas:** Los pacientes podrán elegir al odontólogo, ver su disponibilidad y reservar en pocos pasos.
- **Recordatorios automáticos:** Notificaciones por correo o mensaje para que nadie se olvide de su cita.
- **Gestión de cancelaciones y reprogramaciones:** Un sistema fácil de usar para que los pacientes puedan hacer cambios sin complicaciones.

- **Historial de citas:** Tanto pacientes como odontólogos tendrán acceso a un registro de consultas anteriores.

Cada una de estas funcionalidades está pensada para resolver los problemas que identificamos, ofreciendo una solución práctica y eficiente para todos.

Punto 3

✨Análisis de requerimientos mediante método MoSCoW✨

Must Have 📈

Estas son las funcionalidades esenciales sin las cuales la aplicación no cumpliría su propósito:

- **Registro e inicio de sesión:** Pacientes y odontólogos deben poder crear cuentas y acceder a la plataforma. **(3 días)**
- **Agendamiento de citas: (2 días)**
    - Ver disponibilidad de odontólogos.
    - Seleccionar fecha y hora de la cita.
    - Confirmación de la reserva.
- **Recordatorios automáticos: (3 días)**
    - Notificaciones por correo y/o mensaje de texto antes de la cita.
    - Avisos sobre cancelaciones o cambios en la cita.
- **Gestión de cancelaciones y reprogramaciones: (2 días)**
    - Opción de cancelar una cita con notificación al odontólogo.
    - Posibilidad de reprogramar la cita sin necesidad de llamadas.
- **Historial de citas: (2 días)**
    - Registro de citas pasadas para pacientes.
    - Registro de citas atendidas para odontólogos.

Should Have 🙋🏽‍♂️

Funciones que mejorarían la experiencia, pero no son críticas en la primera versión:

- Búsqueda de odontólogos por especialidad o ubicación. **(1 día)**
- Integración con calendarios (Google Calendar, Outlook, etc.). **(3 días)**
- Múltiples opciones de notificación (WhatsApp, SMS, email). **(2 días)**
- Dashboard para odontólogos con estadísticas de citas y cancelaciones. **(5 días)**

Could Have 🤷🏽‍♂️

Funcionalidades que serían buenas pero no son prioritarias:

- Perfiles de odontólogos con información y valoración de pacientes. **(5 días)**
- Posibilidad de que los odontólogos bloqueen horarios específicos. **(2 días)**
- Opciones de pago en línea por la consulta (si el usuario desea). **(3 días)**
- Chat interno entre paciente y consultorio. **(5 días)**

Won’t Have 🙅🏽‍♂️

Funciones que no se implementarán en esta versión:

- Gestión de inventarios para consultorios**.**

- Integración con seguros médicos.
- Video Consultas o telemedicina**.**

Must have		
Funcionalidad	Tiempo estimado	Motivo
• Registro e inicio de sesión	• **(3 día)**	Será necesario crear todo un entorno para la app.
• Agendamiento de cit**as**	• **(2 días)**	No parece ser una tarea difícil, solo es asignar.
• Recordatorios automáticos	• **(3 días)**	Uno de los ingenieros tienen conocimiento al respecto, comenta que no es una tarea muy complicada.
• Gestión de cancelaciones y reprogramaciones	• **(2 días)**	Solo ajustes en la base de datos con determinado feedback.
• Historial de citas	• **(3 días)**	Recolectar información desde la base de datos es sencillo, pero debemos guardarla de manera precisa.

Should have		
Funcionalidad	Tiempo estimado	Motivo
• Búsqueda de odontólogos	• **(1 día)**	Solo es una búsqueda en la base de datos, sabemos cómo realizarlo.
• Integración con calendarios	• **(5 días)**	No tenemos el conocimiento de cómo hacerlo, pero no parece ser muy complejo.
• Múltiples opciones de notificación	• **(2 días)**	Dos de los ingenieros tienen conocimiento al respecto.
• Dashboard para odontólogos	• **(5 días)**	Nos parece un trabajo elaborado, que llevará tiempo de aprendizaje.

Could have		
Funcionalidad	Tiempo estimado	Motivo
• Perfiles de odontólogos con información y valoración de pacientes	• **(5 días)**	No tenemos el conocimiento para realizarlo.
• odontólogos bloqueen horarios específicos	• **(3 días)**	No tenemos el conocimiento de cómo hacerlo, pero no parece ser muy complejo.
• Opciones de pago en línea por la consulta	• **(3 días)**	No estamos familiarizados con entornos de pago, pero no parece ser muy complejo.
• Chat interno entre paciente y consultorio	• **(5 días)**	Ningún ingeniero sabe cómo establecer una app de mensajería, llevará tiempo de aprendizaje.

Punto 4

Análisis de gestión del desarrollo

✨Tiempo✨

Dado el tiempo disponible para la fase final del semestre, junto con las etapas del proyecto completadas hasta la semana 11 y considerando que el equipo está conformado por cuatro estudiantes en formación en ingeniería de software, se ha diseñado una planificación detallada del tiempo para el desarrollo del sistema. Este se implementará utilizando Python con Django para el backend y Vue.js para el frontend, siguiendo el siguiente cronograma:

Fase	Tareas Principales	Duración Estimada
Análisis y Planificación	Definición de requerimientos, modelado de datos, arquitectura del sistema	1 semana
Diseño	Creación de wireframes, diseño de UI/UX	1 semana
Desarrollo Backend (Django)	Implementación de API REST, base de datos PostgreSQL, autenticación y seguridad.	4 semanas
Desarrollo Frontend (Vue.js)	Implementación de interfaz de usuario y consumo de API	2 semanas
Pruebas y Ajustes	Pruebas unitarias, de integración y usabilidad	1 semana
Despliegue y Documentación	Implementación en servidor (AWS/GCP), documentación técnica y manual de usuario	2 semana

✨Costo✨

Para analizar correctamente este aspecto, primero identificamos los recursos necesarios para el proyecto que implican un costo asociado. En el desarrollo del software se pueden considerar los siguientes:

- **Recursos humanos**: son los costos relacionados a los integrantes del equipo y su contratación.
- **Licencias de Software**: gastos asociados a licencias de herramientas de software que se puedan requerir en el desarrollo proyecto.

- **Cuentas o usuarios de servicios de nube**: asignación de cuentas de nube para realizar las diferentes pruebas que requieran el uso de servicio cloud.
- **Infraestructura**: costos de servidores, almacenamiento en la nube, y otros recursos tecnológicos necesarios

Adicionalmente, para el desarrollo del proyecto ya se han definido el uso de algunas tecnologías específicas, por lo cual, también es un insumo en la estimación del costo de la realización del proyecto, siendo estas las siguientes:

- **FrontEnd:**
  - **Vue 3**: Framework progresivo para la construcción de interfaces de usuario modernas y reactivas.
  - **Tailwind CSS**: Sistema de utilidades CSS para crear interfaces elegantes y responsivas.
- **BackEnd:**
  - **Django + Django REST Framework (DRF)**: Framework de alto rendimiento para la creación de APIs robustas y seguras.
- **Base de datos:**
  - **PostgreSQL**: Sistema de gestión de bases de datos relacional de alto rendimiento.
- **Infraestructura y automatización:**
  - **Heroku (Dynos Standard)**: Hosting para el backend con escalabilidad sencilla y despliegues sin tiempo de inactividad.
- **Herramientas de Desarrollo:**
  - **VS Code o PyCharm**: IDEs recomendados para desarrollo eficiente.
  - **Git + GitHub**: Control de versiones y colaboración en el código.
- **Servicios:**
  - **SendGrid**: Plataforma para el envío de correos electrónicos transaccionales y de notificación.
  - **Auth0 :** Plataforma para gestión de identidad y autenticación segura.

Con base en lo expuesto y las necesidades del proyecto, se presentan las siguientes tablas como síntesis de los costos. Estas estimaciones son investigativas y no se usarán en el presupuesto actual, pero servirán para futuros desarrollos y para concienciar sobre los costos reales del proyecto.

Nota: para precios en Dólares Estadounidenses (USD), su estimación equivalente en Pesos Colombianos (COP) se ha realizado utilizando el tipo de cambio con corte a 9 de febrero de 2025, siendo este $1(USD) = $4153(COP).

Tabla de costos Recursos Humanos (Equipo)			
Recurso	**Justificación**	**Responsabilidades**	**Costo**
Desarrollador Backend	Es necesario contar con un desarrollador especializado en Django y DRF para garantizar una API robusta y segura.	Desarrollo y mantenimiento del backend con Django y DRF, diseño de endpoints, implementación de lógica de negocio y gestión de la base de datos en PostgreSQL.	**Salario mensual promedio:** $5.500.000 COP/ $1.410 USD
Desarrollador Frontend	Un especialista en Vue.js asegurará una interfaz interactiva, eficiente y optimizada para la experiencia del usuario.	Desarrollo de la interfaz de usuario con Vue 3 y Tailwind CSS, integración con API REST, optimización de rendimiento y accesibilidad.	**Salario mensual promedio:** $4.000.000 COP/ $1.026 USD
Diseñador UX/UI	Es necesario para garantizar que la aplicación sea intuitiva y visualmente atractiva para los usuarios.	Diseño de interfaces gráficas, creación de prototipos interactivos, mejora de la usabilidad y experiencia de usuario.	**Salario mensual promedio:** $4.000.000 COP/ $1.026 USD
Tester QA	Garantiza que el sistema funcione sin errores antes de su lanzamiento, mejorando la estabilidad y la experiencia del usuario.	Diseño y ejecución de pruebas funcionales y automatizadas, detección y documentación de errores, validación de correcciones y regresión.	**Salario mensual promedio:** $2.600.000 COP/ $600 USD

Arquitecto de Software	Responsable del diseño técnico del sistema, asegurando escalabilidad, seguridad y eficiencia del desarrollo.	Definición de la arquitectura del software, selección de tecnologías, supervisión del desarrollo y revisión de código.	**Salario mensual promedio:** $6.600.000 COP/ $1.692 USD
Total (1 mes)			$22.700.000 COP/ $5.754 USD
Costo Total (11 semanas)			$62.425.000 COP / $15.818 USD.

Fuentes:

- https://co.indeed.com/career/backend-developer/salaries?from=top_sb
- https://co.talent.com/salary?job=desarrollador+front+end
- https://co.talent.com/salary?job=ux+ui
- https://co.indeed.com/career/tester/salaries
- https://co.computrabajo.com/salarios/arquitectoa-de-software

Dado que el total mensual es de $22.700.000 COP / $5.754 USD, el pago semanal sería de aproximadamente $5.675.000 COP / $1.438 USD. Por lo tanto, el costo total de recursos humanos para 11 semanas sería de $62.425.000 COP / $15.818 USD.

Tabla de costos Licencias, Cuentas Nube e Infraestructura				
Categoría	**Recurso**	**Justificación**	**Especificación**	**Costo**
Herramientas de Desarrollo	**Tecnologías BackEnd**	Herramientas y tecnologías para el desarrollo del backend.	Django y Django REST Framework - Son tecnologías de código abierto y gratuitas. Editor de código: Visual Studio Code - Gratuito.	$0

	Tecnologías FrontEnd	Herramientas y tecnologías para el desarrollo del frontend.	Vue 3 y Tailwind CSS - Son tecnologías de código abierto.	$0
Infraestructura y Hosting	**Base de Datos (PostgreSQL)**	Instancia de PostgreSQL para pruebas y almacenamiento.	Heroku - Base de Datos (Essential 1) - 10 GB de almacenamiento - 20 conexiones - Sin límite de filas.	**Costo mensual:** $9 USD/ $37.000 COP
	Servidor Backend y Frontend	Hosting para el backend con escalabilidad sencilla.	Heroku - Plan de Servidores (Dynos - Standard) - 512 MB o 1 GB de RAM - Escalabilidad horizontal simple - Despliegues sin tiempo de inactividad - Métricas y alertas - Trabajadores en segundo plano ilimitados.	**Costo mensual:** $50 USD/ $205.600 COP
Servicios Adicionales	**Sistema de notificaciones**	Confirmaciones y recordatorios de citas.	SendGrid – Plan gratuito - Envía 100 correos electrónicos al día de forma gratuita	$0

	Autenticación y Seguridad	Necesario para registro e inicio de sesión.	Auth0 – Plan gratuito Hasta 25.000 usuarios - Autenticación de contraseña con correo electrónico, nombre de usuario o número de teléfono - Autenticación social (Google, Facebook, etc.) - Protección básica contra ataques	$0
Total (1 mes)				$242.700 COP/$59 USD
Costo Total (11 semanas)				$667.425 COP / $162.25 USD

Dado que el total mensual es de $242.700 COP / $59 USD, el pago semanal sería de aproximadamente $60.675 COP / $14.75 USD. Por lo tanto, el costo total en licencias, cuentas en la nube e infraestructura para 11 semanas sería de $667.425 COP / $162.25 USD.

Producto mínimo viable (MVP)

El Sistema de Gestión de Citas Odontológicas está diseñado para cubrir las necesidades esenciales de pacientes y odontólogos. Las funcionalidades clave que se incluirán en el MVP son:

● Registro e inicio de sesión:

Pacientes y odontólogos podrán crear cuentas y acceder a la plataforma. Se implementará un sistema de autenticación seguro para garantizar la privacidad de los datos.

● Agendamiento de citas:

Los pacientes podrán:

- Ver la disponibilidad de los odontólogos en la plataforma.
- Seleccionar una fecha y hora para su cita.
- Confirmar la reserva de manera automática.

● Recordatorios automáticos:

Se enviarán notificaciones a los pacientes antes de la cita mediante correo electrónico y/o mensaje de texto. Además, se generarán alertas en caso de cancelaciones o cambios de horario.

● Gestión de cancelaciones y reprogramaciones:

Los pacientes podrán:

- Cancelar una cita con notificación automática al odontólogo.
- Reprogramar la cita sin necesidad de llamar al consultorio.

● Historial de citas:

- Los pacientes tendrán acceso a un registro de sus citas pasadas.
- Los odontólogos podrán ver un historial de citas atendidas.

Ahora, para garantizar que el MVP sea viable y se entregue a tiempo, algunas características deseables pero no esenciales se dejarán para futuras iteraciones del proyecto:

● Búsqueda de odontólogos por especialidad o ubicación:

No se incluirá en la versión inicial. Los pacientes solo podrán agendar citas sin filtrar por especialidad o ubicación.

● Integración con calendarios externos:

La plataforma no se integrará con Google Calendar, Outlook u otros servicios en esta versión.

● Múltiples opciones de notificación:

Inicialmente, solo se enviarán recordatorios por correo, sin integración con WhatsApp.

● Dashboard para odontólogos:

No se implementará un panel con estadísticas de citas y cancelaciones en esta versión.

● Perfiles de odontólogos con información y valoración de pacientes:

No se incluirá un sistema de perfiles donde los pacientes puedan ver información detallada de los odontólogos ni dejar valoraciones.

● Bloqueo de horarios específicos:

Los odontólogos no podrán bloquear horarios de disponibilidad en esta versión inicial.

● Opciones de pago en línea:

El sistema no permitirá realizar pagos en línea por las consultas en esta primera versión.

● Chat interno entre paciente y consultorio:

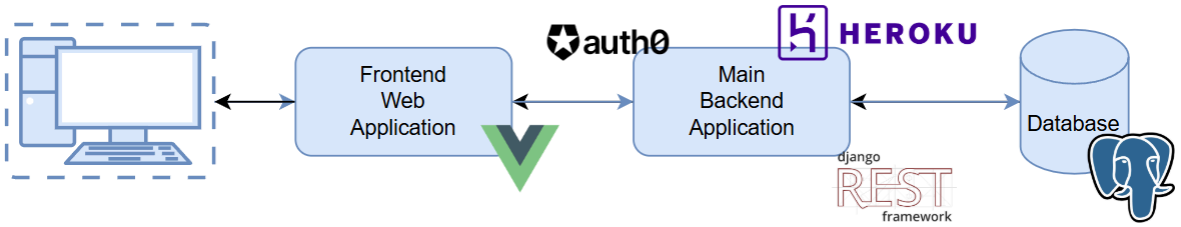
No se implementará un sistema de mensajería interna entre pacientes y consultorios en esta versión.

Punto 8

✨Arquitectura del sistema✨

La plataforma está diseñada utilizando una arquitectura de cliente-servidor, lo que permite una separación clara entre las responsabilidades del frontend y el backend, garantizando flexibilidad y facilidad de mantenimiento. La arquitectura se compone de los siguientes elementos principales:

- **Frontend**:
  Una aplicación web desarrollada con Vue.js que sirve como la interfaz principal para los usuarios. Esta interfaz permite a los usuarios interactuar con la plataforma, como autenticarse, realizar operaciones y visualizar información de manera intuitiva.
- **Backend**:
  Una aplicación principal basada en Django REST Framework que gestiona la lógica de negocio de la plataforma. Este servicio central se encarga de procesar las solicitudes provenientes del frontend, realizar operaciones en la base de datos, y coordinar la interacción con servicios externos.
- **Base de datos**:
  Una base de datos relacional PostgreSQL se utiliza para almacenar la información estructurada de la plataforma, como datos de usuarios, interacciones y registros transaccionales.

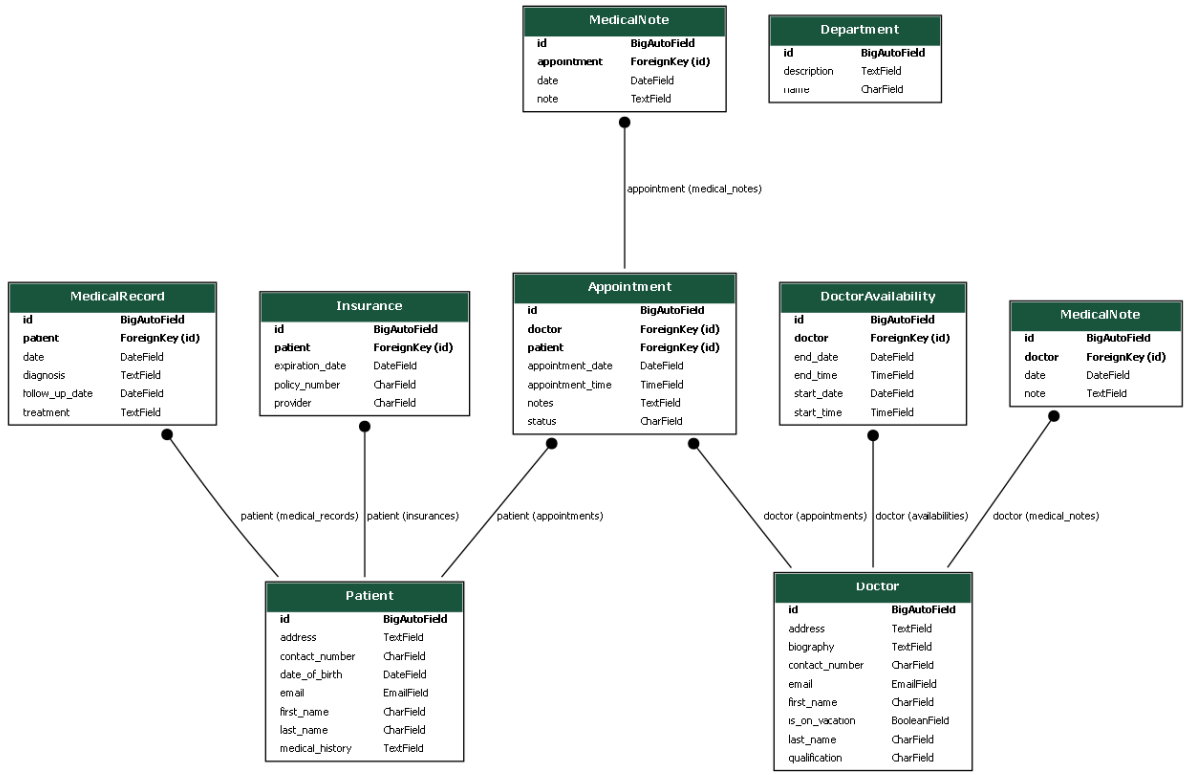


Como se observa en el diagrama, el cliente interactúa únicamente con el Frontend Web Application, el cual está desarrollado utilizando Vue.js. Este frontend se comunica con la Main Backend Application, que es el microservicio encargado de gestionar toda la lógica de negocio y conectarse con la base de datos. Vemos cómo este microservicio, alojado en Heroku, utiliza Django REST Framework para exponer las APIs necesarias. Además, este backend incorpora Auth0 para gestionar la autenticación y autorización de los usuarios. Finalmente, la Main Backend Application se comunica directamente con una base de datos PostgreSQL, que almacena toda la información del sistema de manera estructurada para su posterior recuperación y uso.

La principal razón para elegir esta estructura radica en su simplicidad y facilidad de manejo. Al mantener separadas la lógica de negocio y la interfaz de usuario, se facilita el desarrollo y mantenimiento de cada funcionalidad de forma independiente, permitiendo realizar modificaciones de manera rápida y eficiente. Además, esta arquitectura permite una

comunicación fluida entre el servidor y la base de datos, lo que hace más sencilla la gestión de información clave, como registros, horarios y detalles asociados.

✨Diseño de bases de datos✨



La base de datos está diseñada para gestionar la información de pacientes, doctores y citas médicas de manera eficiente. Se ha estructurado de forma relacional, permitiendo la asociación entre distintas entidades clave.

Las tablas principales incluyen **Patient** y **Doctor**, que almacenan la información de los pacientes y doctores respectivamente. **Patient** contiene datos personales como nombre, dirección, contacto e historial médico, mientras que **Doctor** almacena información como su biografía, calificación y disponibilidad. Ambas tablas tienen relaciones con otras entidades que permiten gestionar citas y registros médicos.

Appointment: es la encargada de registrar las citas médicas programadas. Contiene información sobre el doctor y el paciente involucrado, así como la fecha, hora, notas y estado de la cita. Esta tabla es fundamental para la funcionalidad de agendamiento y control de consultas dentro del sistema.

DoctorAvailability: gestiona los horarios disponibles de los doctores, incluyendo fechas y horas de inicio y fin. Es clave para garantizar que las citas sean programadas solo en horarios disponibles y evitar conflictos en la agenda de los doctores.

Adicionalmente, se encuentran **MedicalRecord** y **MedicalNote**. **MedicalRecord** registra diagnósticos, tratamientos y fechas de seguimiento de los pacientes. Es fundamental para la visualización del historial clínico de cada paciente. **MedicalNote**, por su parte, permite que los doctores agreguen anotaciones sobre las citas médicas realizadas, almacenando observaciones relevantes sobre la consulta. Ambas tablas tienen una llave foránea hacia **Patient** o **Doctor**, según corresponda, para identificar a qué paciente pertenece cada registro o qué doctor realizó la nota médica.

Insurance: gestiona la información de seguros médicos asociados a los pacientes, incluyendo proveedor, número de póliza y fecha de expiración. Tiene una llave foránea hacia **Patient** para identificar qué paciente posee cada seguro registrado.

Finalmente, **Department:** permite clasificar a los doctores según su área médica dentro de la clínica, almacenando información sobre el nombre y descripción del departamento.

La razón por la que se escogió una base de datos relacional es que, gracias a estas asociaciones entre las distintas entidades, se hace más fácil construir funcionalidades como la visualización del historial médico de los pacientes, la gestión de citas odontológicas y la disponibilidad de los doctores. Además, esta estructura facilita la validación de datos, evitando la duplicidad de información y asegurando la integridad en el manejo de pacientes y doctores dentro del sistema.

Punto 9

✨Patrones de Diseño aplicados para el proyecto.✨

Al desarrollar nuestra aplicación con Django, utilizamos el patrón de diseño “MVC” (Model-View-Controller), aunque en Django se implementa como MTV **(Model-Template-View)**. De esta manera, el modelo gestiona los datos y la lógica del centro odontológico , la **vista** se encarga de la presentación y el **controlador** maneja las solicitudes del usuario y la comunicación entre los componentes. Este patrón facilita la separación de responsabilidades, mejora la mantenibilidad del código y permite reutilizar componentes de manera eficiente 🫡.

Además, implementaremos el patrón **Singleton** para gestionar las sesiones de usuario, asegurando que cada usuario solo pueda tener una sesión activa a la vez. Esto ayuda a evitar problemas de concurrencia y posibles inconsistencias en la gestión de citas. Al restringir múltiples sesiones simultáneas, garantizamos un entorno más seguro y controlado dentro de la plataforma.