Tecnológico de Monterrey

Delivery#1 (Design & Architecture)

Implementación de un diagrama de clases UML con el patrón MVC

Planeación de sistemas de software (Gpo 104)

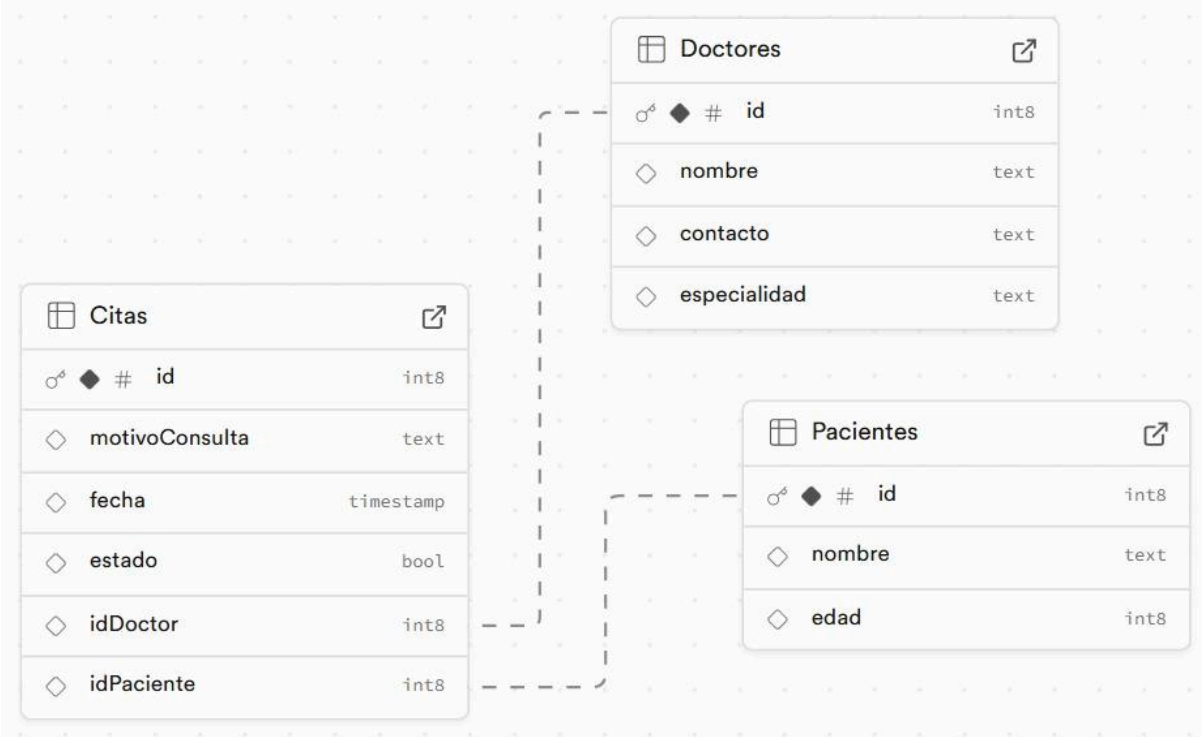
Marcela Beatriz De La Rosa Barrios A01637239

Juan Yael Ávalos Mayorga A01276329

05/03/2025

Implementación de un diagrama de clases UML con el patrón MVC

Diagrama UML Implementado:



A continuación, se describe la participación y el aporte de cada miembro del equipo:

1. Juan Yael Ávalos Mayorga

- **Configuración inicial de la base de datos:** Definió las credenciales de acceso a Supabase y estableció la estructura de las tablas (Doctores, Pacientes, Citas).
- **Creación de las rutas (endpoints) básicas:** Implementó los primeros endpoints en Flask para la creación y consulta de citas, así como para la obtención de datos de doctores y pacientes. Esto sirvió como base para la posterior refactorización.
- **Base del Front-End en React:** Escribió los componentes iniciales (por ejemplo, el `App.jsx`, la estructura de carpetas en `client/src`, y algunos componentes de ejemplo) que permitieron arrancar la interfaz gráfica y probar la comunicación con el servidor.

2. Marcela Beatriz De La Rosa Barrios

- **Organización en el patrón MVC y refactorización:** Estructuró las carpetas y archivos de acuerdo con el patrón de diseño MVC, modificando y separando los controladores, modelos y las vistas.
- **Implementación de modelos y DAO:** Creó las clases `Doctor.py`, `Paciente.py` y `Cita.py` en la carpeta `models/` y centralizó la conexión a la base de datos en la carpeta `dao/` para cumplir con los principios de abstracción y modularidad.
- **Refinamiento de las rutas y lógica de negocio:** Ajustó los endpoints para manejar correctamente la creación, consulta, aprobación y eliminación de citas, así como la obtención de información de doctores y pacientes.
- **Mejoras en la interfaz y experiencia de usuario:** Agregó componentes de React adicionales y estilos CSS para mostrar las citas pendientes, las citas aprobadas y la opción de agendar citas. También añadió validaciones y alertas para informar al usuario sobre el resultado de las operaciones (éxito o error).
- **Manejo de CORS y respuestas HTTP:** Configuró el uso de CORS en Flask y ajustó los métodos PUT y DELETE para que funcionaran correctamente con la aplicación React.

Gracias a esta colaboración, se logró un proyecto funcional que cumple con los criterios de la actividad:

- Separación clara de la lógica (MVC).
- Mínimo de 3 controladores.
- Al menos 2 clases de objetos.
- Un DAO para la conexión y consultas SQL.
- Múltiples vistas y componentes en React.