**salasdejunta/settings.py**

1) agregamos la aplicación para reservas

2) agregar los datos para la conexión a la base de datos MySQL

3) rest framework para la api

4) agregar dirección de la carpeta “static” con los archivos para el front

5) agregar cors para permitir las llamadas desde el front

6) cors middleware

7) establece los orígenes permitidos

8) cambiamos zona horaria a la local

**salasdejunta/urls.py**

1) se agregan las ubicaciones de los archivos estáticos

**reservas/models.py**

1) modelo/clase para las salas de junta, con su nombre y capacidad. Se regresa el nombre de la sala.

2) Modelo para las reservaciones hechas. Se manda llamar la sala correspondiente y se añaden atributos de hora de inicio y hora de fin

3) Si se borra una sala, también lo harán todas sus reservas

4) Función para validaciones de evitar reservas mayores a 2 horas y evitar que se reserve una sala ya reservada en un rango de tiempos.

5) Se buscan todas las salas con reservas y se excluye la actual

Después de crear los modelos de la base de datos en models.py, se ejecuta la migración para mostrar las tablas en MySQL. Automáticamente se añade un campo “id”.

python manage.py makemigrations reservas

python manage.py migrate

Imagen que contiene Texto

El contenido generado por IA puede ser incorrecto.

Nota: las horas se guardan en formato utc en la base de datos, pero las demás validaciones se hacen con utc-6, no afecta porque django hace automáticamente las conversiones al comparar

**reservas/serializers.py**

1) Se crean las serializaciones de los objetos para recibir un JSON

**reservas/views.py**

1) recuperar datos de la base de datos

2) Validar que la sala no esté ocupada

3) Validar que la reserva no supere las 2 horas

4) validar que la reservación no se haga con horas negativas

5) renderizar el index.html

6) formatear la hora actual al formato recibido del front

7) formatear los datos recibidos a la zona horaria del servidor

8) validar que la reservación no se haga en una fecha u hora pasada

9) se busca si ya existe una sala con el nombre recibido. En caso de que sí, se devuelve un error

**reservas/urls.py**

1) generar automáticamente las rutas de la api

**reservas/tasks.py**

1) ciclo infinito que elimina las reservaciones que su tiempo de fin ya haya pasado

2) se hace una búsqueda con un filtro donde al campo “hora\_fin” se le agrega “\_\_lt”, el cual es un lookup de Django ORM (Object-Relational Mapper) que significa “less than”, por lo que se filtran todas las reservaciones donde “hora\_fin” sea menor que el tiempo actual (variable tiempo\_actual) y se le indica que el registro se eliminará

3) se crea el hilo que manejará la liberación de salas de manera automática

**reservas/apps.py**

1) al ejecutar el proyecto, django ejecuta también las aplicaciones, y al estar dentro de la configuración de la aplicación, se ejecuta la función que comienza el hilo

**static/scripts.js**

1) función para obtener y mostrar las salas

2) función para obtener y mostrar las reservaciones

3) manejar envío del formulario de reserva

4) añadir las opciones de salas para reservar

5) desplegar el modal de creación de salas

6) cerrar modales

7) crear una nueva sala, se obtienen los datos del formulario en el modal y se hace la petición correspondiente

8) función que despliega el modal para modificar una sala, haciendo una petición para mostrar los datos de esa sala en específico

9) función para modificar la sala deseada, se toman los datos de su modal junto con el id de la sala a modificar. Se usa el método PUT pues este es el que maneja django rest para las modificaciones

10) función para eliminar una sala, se pide confirmación y se usa el método DELETE pues este es el que maneja django rest para las eliminaciones

11) función para liberar (eliminar) manualmente una sala, se pide confirmación y se usa el método DELETE pues este es el que maneja django rest para las eliminaciones

12) se hace una petición al mostrar las salas para visualizar el nombre de la sala en lugar del ID de sala asignado

13) se agrega una validación para saber si el back respondió con un mensaje de error y poder manejarlo

**static/styles.css**

1) desplegar elemento por encima de los demás