Memoria del Proyecto: API REST de Gestión de Reservas de Salas de Estudio con Node.js

# 1. Introducción

Este documento describe el desarrollo de una API RESTful implementada en Node.js utilizando el módulo HTTP nativo. La API permite gestionar la reserva de salas de estudio en una biblioteca o centro de coworking. La aplicación permite consultar salas disponibles, registrar nuevas reservas, cancelar reservas existentes y consultar todas las reservas activas. Los datos se almacenan de forma persistente en archivos JSON.

# 2. Objetivos del Proyecto

- Desarrollar una API REST utilizando Node.js sin frameworks externos como Express.

- Implementar operaciones CRUD básicas para la gestión de reservas de salas.

- Asegurar la persistencia de los datos mediante archivos JSON.

- Garantizar validaciones clave como disponibilidad horaria y capacidad máxima por sala.

# 3. Configuración del Entorno

Desde la terminal del proyecto:

npm init -y

npm install uuid

Estructura del Proyecto

API\_RESERVAS/

├── index.js

├── rooms.json

├── reservations.json

├── package.json

└── node\_modules/

Ejecución del Proyecto

node index.js

El servidor queda disponible en: http://localhost:3000

# 4. Desarrollo de la Aplicación

4.1 Persistencia de Datos

Se utilizaron dos archivos `.json`:

- rooms.json: contiene las salas disponibles.

- reservations.json: almacena las reservas activas.

4.2 Funcionalidad Principal

La API permite:

- Listar salas disponibles (GET /rooms)

- Registrar una nueva reserva (POST /reservations)

- Cancelar una reserva existente (DELETE /reservations/:id)

- Listar reservas activas (GET /reservations)

4.3 Validaciones Implementadas

- No se permiten reservas que superen la capacidad de la sala.

- No se permiten reservas duplicadas en la misma franja horaria.

- Las reservas deben estar asociadas a una sala válida.

# 5. Endpoints de la API

GET /rooms - Devuelve todas las salas disponibles.

Texto

El contenido generado por IA puede ser incorrecto.

GET /reservations - Devuelve todas las reservas activas.

Texto

El contenido generado por IA puede ser incorrecto.

POST /reservations - Registra una nueva reserva.

Texto

El contenido generado por IA puede ser incorrecto.

DELETE /reservations/:id - Cancela una reserva según su ID.

Captura de pantalla de un teléfono celular

El contenido generado por IA puede ser incorrecto.

# 6. Resultados y Observaciones

Resultados:

- Rutas y métodos funcionan correctamente.

- Validaciones correctas de horario y capacidad.

- Cancelación y consulta de reservas operativas.

Problemas:

- Comparaciones exactas de rutas causaban errores.

Soluciones:

- Se añadieron logs de depuración.

- Uso de uuid y modularización de funciones.

# 7. Conclusión

El proyecto permitió desarrollar una API RESTful funcional con Node.js puro. Cumple los objetivos de gestión de reservas, validaciones y persistencia.

Mejoras Futuras:

- Reemplazar archivos JSON por base de datos real.

- Añadir autenticación básica.

- Usar Express para mejorar escalabilidad.

- Añadir pruebas automatizadas.

# 8. Repositorio Git

https://github.com/RaulMatas/Fundamentos-del-Desarrollo-Backend-con-Node.js.git