Memoria del Proyecto: API REST con Express – Gestión de Recursos

1. Introducción

Este proyecto desarrolla una API RESTful utilizando Node.js y Express para la gestión de recursos genéricos. Se implementa un CRUD completo, validaciones con Joi, autenticación con JWT, middlewares personalizados y almacenamiento persistente en un archivo JSON. El objetivo es simular una solución de backend moderna y funcional sin el uso de una base de datos real.

2. Objetivos del Proyecto

- Desarrollar una API REST utilizando Express.

- Implementar autenticación segura con JSON Web Tokens.

- Validar la entrada de datos usando Joi.

- Registrar logs de solicitudes HTTP con morgan.

- Simular persistencia mediante archivos JSON.

3. Tecnologías Utilizadas

- Node.js

- Express

- Joi

- jsonwebtoken

- dotenv

- morgan

- uuid

- fs (sistema de archivos)

4. Estructura del Proyecto

api_express_recursos/

├── controllers/

├── middlewares/

├── routes/

├── data/

├── .env

├── index.js

├── package.json

├── README.md

5. Endpoints y Funcionalidades

POST /login: genera un token JWT si las credenciales son válidas.

GET /recursos: lista todos los recursos.

GET /recursos/:id: obtiene un recurso por ID.

POST /recursos: crea un recurso (requiere JWT).

PUT /recursos/:id: actualiza un recurso (requiere JWT).

DELETE /recursos/:id: elimina un recurso (requiere JWT).

6. Validaciones

- El campo 'nombre' es obligatorio y debe tener al menos 3 caracteres.

- El campo 'fecha' es obligatorio y debe tener formato ISO 8601 (YYYY-MM-DD).

- 'descripcion' es opcional.

- Validaciones realizadas con Joi.

7. Middlewares Aplicados

- Morgan: para registro de peticiones HTTP.

- Joi: para validación de datos.

- Middleware personalizado para autenticación con JWT.

- Manejador de errores para respuestas unificadas.

8. Pruebas Realizadas

Se utilizó Postman para verificar:

- Acceso protegido por token.

- Creación de recursos válidos e inválidos.

- Modificación y eliminación controlada.

- Validaciones correctamente aplicadas.

9. Conclusiones y Mejoras Futuras

El proyecto cumple todos los requisitos funcionales y técnicos propuestos.

Mejoras posibles:

- Incorporar una base de datos real (MongoDB, PostgreSQL).

- Añadir control de usuarios y roles.

- Desarrollar pruebas unitarias y de integración automatizadas.

10. Repositorio del Proyecto

https://github.com/RaulMatas/Librer-as-de-Backend-con-Node..git