Descripción

Este proyecto es un backend construido en **Node.js** para gestionar usuarios y proyectos dentro de una plataforma protegida con autenticacion. Está pensado para entornos donde se requiere un control claro de accesos mediante roles (por ejemplo, administrador y usuarios comunes). Se utiliza **Sequelize** como ORM para facilitar la comunicacion con una base de datos **PostgreSQL**, lo que permite mantener el codigo más limpio y organizado. Todo el entorno se configura a traves de variables de entorno para mayor seguridad y flexibilidad. El objetivo principal del sistema es proveer una base solida para aplicaciones que necesiten gestionar multiples usuarios, roles y recursos de forma segura y escalable.

Programas Usados

- **Node.js:** Ultima version
- **Express.js**: Framework que facilita la creación de servidores HTTP.
- **Sequelize**: ORM que permite interactuar con bases de datos relacionales.
- **PostgreSQL**: Base de datos relacional.
- **JWT**: Método de autenticación para gestionar sesiones de usuario de forma segura.
- **dotenv**: Permite cargar configuraciones desde un archivo .env.

Estructura Principal de la raíz del proyecto

- **database.js**: Configuración de la base de datos y establecimiento de la conexión con PostgreSQL.

- **dotenv.js**: Maneja las variables de entorno cargadas desde un archivo .env para configuraciones sensibles.

- **user.routes.js**: Rutas relacionadas con la gestión de usuarios (crear, actualizar, eliminar y consultar).

- **auth.controller.js**: Controlador que maneja el registro e inicio de sesión de usuarios.

- **app.js**: Configuración del servidor Express y asociación de rutas.

- **server.js**: Inicia la conexión a la base de datos y ejecuta el servidor.

Funcionalidades Principales

- **Autenticación**: Los usuarios pueden registrarse e iniciar sesión. Se utiliza JWT para generar un token que valida las peticiones futuras.

- **Gestión de Usuarios**: Los administradores pueden crear, actualizar, eliminar y consultar usuarios registrados.

- **Gestión de Proyectos**: Los administradores pueden crear, actualizar, eliminar y consultar proyectos disponibles en la API.

Autenticación usada:

- **Autenticación JWT**: Asegura que solo los usuarios con un token válido puedan acceder a rutas protegidas.

- **Control de Roles**: Se gestionan los permisos mediante roles, lo que limita el acceso a ciertas rutas solo a administradores o usuarios con permisos específicos.

Rutas usadas:

- **Autenticación**: Rutas para el inicio de sesión y registro de usuarios.
- **Usuarios**: Rutas para crear, modificar, eliminar y consultar usuarios.
- **Proyectos**: Rutas para gestionar proyectos, accesibles solo por administradores.

(La evidencia esta en el github de la conexion a la base de datos)