[**https://github.com/sharon3snc/EntregablesBackend.git**](https://github.com/sharon3snc/EntregablesBackend.git)

Archivo: PLATZI

**Conclusiones**

Utilizo el framework Express para crear una API REST en Node.js. Express simplifica la creación de rutas y el manejo de solicitudes HTTP.

Para la resolución del ejercicio he utilizado la API de Platzi: https://api.escuelajs.co/api/v1/products

La conexión a la base de datos MongoDB se realiza en el archivo database/index.js utilizando el módulo mongoose. Mongoose proporciona una interfaz de modelado de datos y facilita la interacción con MongoDB.

El archivo models/product.js define el esquema del producto utilizando el módulo mongoose. El esquema define la estructura de los datos del producto y cómo se deben almacenar en la base de datos.

El archivo routes/products.js contiene las rutas para manejar las solicitudes relacionadas con los productos. Incluye rutas para obtener todos los productos, crear un nuevo producto, actualizar un producto existente y eliminar un producto.

Para cada ruta, se manejan los posibles errores utilizando bloques try-catch. Si ocurre un error, se envía una respuesta con el código de estado y un mensaje con el error.

La aplicación se ejecuta en el puerto 3000 utilizando app.listen(). Esto permite que la API esté disponible para recibir solicitudes en http://localhost:3000.

He utilizado Postman para probar las rutas de la API. Se pueden enviar solicitudes GET, POST, PUT y DELETE a las rutas correspondientes para interactuar con los datos de los productos. Debajo dejo las capturas de pantalla correspondientes a las pruebas realizadas.

La integración con MongoDB permite almacenar y recuperar los datos de los productos. Al enviar una solicitud POST, por ejemplo, se debería crear un nuevo documento en la colección de productos de la base de datos. No he podido hacer la comprobación con la base de datos porque me da un error de conexión que no he podido solucionar.

En resumen, he creado una API REST utilizando Node.js, Express y MongoDB. Permite realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en una base de datos MongoDB, y maneja errores y códigos de estado adecuados para las respuestas.

**Comprobación con Postman: GET, POST, PUT, DELETE**

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente