DOCUMENTACION TIENDA DE LIBROS

**Introducción**

Esta API permite gestionar una colección de libros, permitiendo realizar operaciones como obtener, agregar, actualizar y eliminar libros. Los datos se almacenan en un archivo JSON.

**Requisitos Previos**

* Node.js (version 12 o superior)
* npm (gestor de paquetes de Node.js)

**Instalación**

1. **Clonar el repositorio**

**-git clone (url)**

1. **Instalar dependencias**

-npm install

1. **Iniciar el servidor**

-node index.js

4. Ver Frontend

- Debemos instalar una extensión llamada live server y en nuestra parte de abajo aparecerá esto  le damos ahí y el automáticamente nos abre nuestro frontend

**Uso de la API**

La API permite las siguientes operaciones a través de los siguientes endpoints:

Para hacer estas pruebas lo que debemos de hacer es crear un archivo request.http y ahí hacer las peticiones que queremos, no lo podemos hacer en el navegador porque solo permite peticiones tipo get

El segundo paso es estar trabajando en visual e instalar una extensión

**Texto

Descripción generada automáticamente**

Esta será la que nos va a permitir hacer nuestras pruebas de cada uno de los endpoints

**1. Obtener todos los libros**

* **Método:** GET **Endpoint:** /libros
* **Descripción:** Devuelve una lista de todos los libros del JSON

**Ejemplo de solicitud:**



Texto

Descripción generada automáticamenteRespuesta :

**2. Obtener un libro por ID**

* **Método:** GET **Endpoint:** /libros/:id
* **Descripción:** Devuelve un libro específico basado en su ID.

**Ejemplo de solicitud:**



Respuesta



**Errores:**

* **404 Not Found**: Si no se encuentra el libro.

**3. Agregar un nuevo libro**

* **Método:** POST **Endpoint:** /crear
* **Descripción:** Agrega un nuevo libro a la base de datos.

**Ejemplo de solicitud:**

Texto

Descripción generada automáticamente

**Respuesta:**

Texto

Descripción generada automáticamente con confianza media

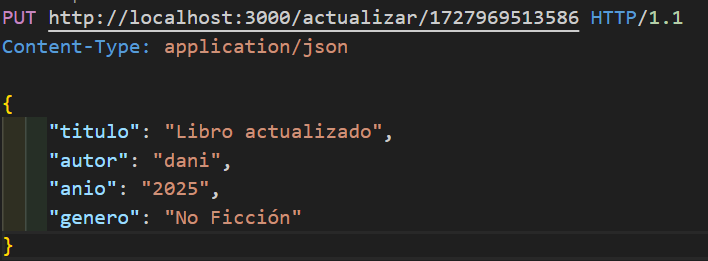
**Errores:**

* **400 Bad Request**: Si faltan campos requeridos.

**4. Actualizar un libro existente**

* **Método:** PUT **Endpoint:** /actualizar/:id
* **Descripción:** Actualiza la información de un libro existente.

**Ejemplo de solicitud:**



**Respuesta:**

Texto

Descripción generada automáticamente con confianza baja

Texto

Descripción generada automáticamente

**Errores:**

* **404 Not Found**: Si no se encuentra el libro.

**5. Eliminar un libro**

* **Método:** DELETE **Endpoint:** /eliminar/:id
* **Descripción:** Elimina un libro de la base de datos.

**Ejemplo de solicitud:**



**Respuesta:**

Texto

Descripción generada automáticamente

**Errores:**

* **404 Not Found**: Si no se encuentra el libro.

Funciones principales

-readData()

Lee el contenido del archivo db.json y devuelve los datos de los libros como un objeto JSON. Si ocurre un error, devuelve un objeto con una propiedad books que es un array vacío.

-writeData(data)

Recibe un objeto de datos y lo escribe en el archivo db.json, convirtiéndolo a formato JSON. Si ocurre un error durante el proceso de escritura, lo registra en la consola.