



Proyecto final - Back End

Se desarrollará un servidor que contenga los endpoints y servicios necesarios para gestionar los productos y carritos de compra en el e-commerce

Primer entrega.

▼ FILMINA 1:

☒ ~~Desarrollar el servidor basado en Node.js y express, que escuche en el puerto 8080.~~

☒ ~~Tiene que disponer de dos grupos de rutas:~~

▼ RUTAS:

☒ ~~/products.js~~

☒ ~~/carts.js~~

☒ ~~Dichos endpoints estarán implementados con el router de express, con las siguientes especificaciones:~~

PRODUCTS.JS:

☒ ~~Para el manejo de productos, tendrá su router en /api/products~~

☒ ~~Configurar las siguientes rutas:~~

▼ RUTAS:

- ✓ La ruta raíz GET / deberá listar todos los productos de la base.
- ✓ ~~debe incluir la limitación ?limit del desafío anterior. (ESTO NO SE SI ESTA BIEN)~~
- ✓ La ruta GET /:pid deberá traer sólo el producto con el id proporcionado.

▼ FILMINA 2:

PRODUCTS.JS:

▼ RUTAS:

- ✓ La ruta raíz POST / deberá agregar un nuevo producto con los campos:-

▼ CAMPOS:

- Todos los campos son obligatorios, a excepción de thumbnails.
- ✓ ~~id: Number/String~~
 - (A tu elección, el id NO se manda desde body, se autogenera como lo hemos visto desde los primeros entregables, asegurando que NUNCA se repetirán los ids en el archivo.)
- ✓ ~~title:String~~
- ✓ ~~description:String~~
- ✓ ~~code:String~~
- ✓ ~~price:Number~~
- ✓ ~~status:Boolean~~
 - Status es true por defecto.
- ✓ ~~stock:Number~~
- ✓ ~~category:String~~
- ✓ ~~thumbnails~~
 - Array de Strings que contenga las rutas donde están almacenadas las imágenes referentes a dicho producto

▼ FILMINA 3:

PRODUCTS.JS:

- ✓ La ruta PUT /:pid deberá tomar un producto y actualizarlo por los campos enviados desde body.
 - NUNCA se debe actualizar o eliminar el id al momento de hacer dicha actualización. (esto no lo entendí)
- ✓ La ruta DELETE /:pid deberá eliminar el producto con el pid indicado.

CARTS.JS:

- ✓ El carrito, tendrá su router en /api/carts/
- ✓ Configurar dos rutas:

▼ RUTAS:

- ✓ La ruta raíz POST / deberá crear un nuevo carrito con la siguiente estructura:

▼ ESTRUCTURAS DEL POST/:

- ✓ id: Number/String
 - (A tu elección, de igual manera como con los productos, debes asegurar que nunca se dupliquen los ids y que este se autogenera).
- ✓ products: Array
 - contendrá objetos que representen cada producto

▼ FILMINA 4:

CARTS.JS:

▼ RUTAS:

- ✓ La ruta GET /:cid deberá listar los productos que pertenezcan al carrito con el parámetro cid proporcionados.
- ✓ La ruta POST /:cid/product/:pid deberá agregar el producto al arreglo "products" del carrito seleccionado, agregándose como un objeto bajo el siguiente formato:

▼ FORMATO:

- ✓ ~~Product: SÓLO DEBE CONTENER EL ID DEL PRODUCTO (Es crucial que no agregues el producto completo)~~
- ✓ ~~Quantity: debe contener el número de ejemplares de dicho producto. El producto, de momento, se agregará de uno en uno.~~
- ✓ ~~Además, si un producto ya existente intenta agregarse al producto, incrementar el campo quantity de dicho producto.~~

▼ FILMINA 5:

PERSISTENCIA:

- ✓ ~~La persistencia de la información se implementará utilizando el file system.~~
- ✓ ~~los archivos "productos.json" y "carrito.json", respaldan la información.~~
- No es necesario realizar ninguna implementación visual, todo el flujo se puede realizar por Postman o por el cliente de tu preferencia.