Pablo Ausín Román
11 de junio de 2020

Proyecto Desarrollo Aplicación REST API

Para este proyecto elegí hacer una librería como temática de la practica.

Para la creación de la REST API primero iniciaremos un proyecto de NodeJS con el comando *npm init -yes*. Una vez inicializado empezaremos a instalar los paquetes necesarios con el comando: *npm i "nombre_paquetes"* . Los paquetes que usaremos son:

- **Connect-Flash**. Este paquete lo use para la creación de avisos cuando creamos, modificamos, borramos o compramos un libro.
- **Express**. Paquete necesario para la creación de nuestro servidor.
- **Express-Handlebars**. Paquete que utilizo para enriquecer HTML y poder crear paginas web mas complejas.
- **Express-Session**. Paquete necesario para utilizar connect-flash. Solo configuramos los parámetros básicos y no se utiliza para su finalidad concreta que es crear sesiones.
- **Mongoose**. Necesario para conectar nuestro servidor a la base de datos creada en MongoDB.
- **Nodemon**. Paquete para automatizar el reinicio del servidor cada vez que guardemos un cambio en nuestro proyecto.
- **Path**. Sirve para que el nuestro servidor reconozca las rutas de donde se esta ejecutando el programa ya sea en Ubuntu o en Windows.
- **Router**. Paquete para la creación de rutas GET y POST para el servidor.

Una vez instalados los paquetes creamos el archivo index.js. Este archivo es el principal de nuestro proyecto donde configuramos nuestro servidor, requerimos todos los paquetes instalados anteriormente, inicializamos el motor de plantillas handlebars, creamos las variables globales ‘correct’ e ‘incorrect’ para mostrar los avisos y especificamos en que puerto estará escuchando el servidor, por defecto el 3000.

Posteriormente configuramos el paquete Nodemon. Para ello, abrimos el archivo package.json y modificamos la descripción "dev:" poniendo la instrucción: "nodemon src/index.js". Con esto, iniciamos nuestro servidor usando la instrucción por terminal: *npm run dev.*

Para una mejor lectura del código creamos por separado un archivo llamado database.js donde establecemos la conexión a la base de datos.

En la carpeta models creamos un schema. Este schema sirve para la creación e inserción de datos desde el servidor hacia la base de datos. En él creamos la tabla y los datos que se van a introducir en ella. La variable id no la creamos porque la propia base de datos genera uno aleatoriamente.

En la carpeta routes creamos el archivo routes.js. Ahí están todas las rutas necesarias para la gestión de las paginas web donde se muestran los resultados de las consultas. Para añadir o modificar un libro recibimos los datos mediante un formulario enviado con el método POST. Para la compra de un articulo comprobamos la existencia del mismo y cuantas unidades hay disponibles. Si compramos la última unidad se eliminará el articulo y nos mostrará un aviso diciendo que es la última unidad disponible.

La ultima carpeta, views, contiene las paginas web. Dentro de esta se encuentra la carpeta layouts que contiene la pagina principal, 'main.hbs'. El archivo 'index.hbs' contiene el cuerpo de la pagina principal donde se mostrara todos los libros disponibles en nuestra base de datos. En la carpeta partials tenemos las vistas parcialesa que podrán cambiar dinámicamente conforme a los datos existentes. En ella encontramos los ficheros 'add.hbs', 'errors.hbs', 'modify.hbs' y 'search.hbs'.

En el primer archivo, 'add.hbs', crearemos el formulario para la introducción de un nuevo libro. En el siguiente,'errors.hbs', muestra el contenido de las variables globales 'correct' y 'incorrect' siempre que contengan algún valor que mostrar. El archivo 'modify.hbs' contiene el formulario para poder modificar cualquier campo de un libro existente y el último archivo, 'search.hbs', creamos la vista para que el usuario busque un libro por el atributo que elija, en este caso puede buscar por: ID, Autor, Titulo, Editorial, Género, Resumen, Número de Paginas, Año de Creación, Precio y Cantidad de Libros Disponibles. En todos los archivos .hbs utilice el framework de CSS Boostrap que para crear una interfaz mas agradable.