taller de construir una ApiRestque permita realizar un CRUD a una base de datos relacionalynorelacional con JavaScript

Aprendiz: Jairo Stiven Lopez Alvarez

Instructor: Wilson Martínez Saldarriaga

Centro de gestión y desarrollo sostenible sur colombiano

Análisis y desarrollo de software

Pitalito-Huila

2023

1. **Crear una Api Rest que permita realizar un crud por medio de EndPoint a una base de datos de una prendería**

Primero comenzamos creando la base de datos con el siguiente comando

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

El siguiente comando se utiliza para seleccionar la base de datos con la que deseamos trabajar.

Con los siguientes comandos comenzamos creando las tablas

Texto

Descripción generada automáticamente

Con el siguiente comando, creamos tres relaciones entre tablas utilizando la sentencia 'ALTER TABLE'. Al hacerlo, es fundamental identificar cuál tabla es la 'tabla principal' y cuál es la 'tabla secundaria' en cada una de estas relaciones."Texto

Descripción generada automáticamente

La base de datos ha sido creada y visualizada con éxito en XAMPP.Interfaz de usuario gráfica

Descripción generada automáticamente

|  |  |
| --- | --- |
| **Requerimiento Funcionales** | **Descripción** |
| RF1 | Crear un EndPoint que permita validar usuarios (identificación y password ) y generar un token |
| RF2 | Crear un endPoint que permita registrar clientes con token |
| RF3 | Crear un endPoint que permita actualizar un cliente con token |
| RF4 | Crear un endPoint que permita eliminar un cliente con token |
| RF5 | Crear un endPoint que permita listar los todos clientes |
| RF6 | Crear un endPoint que permita registrar nuevos artículos con token |
| RF7 | Crear un endPoint que permita desactivar un artículo con token |
| RF8 | Crear un endPoint que permita registrar un alquiler con token |
| RF9 | Crear un endPoint que permita actualizar un alquiler con token |
| RF10 | Crear un endPoint que permita registrar un interés con token |
| RF11 | Crear un endPoint que permita eliminar un interés con token |
| RF12 | Crear un endPoint que permita listar los interés pagados de los alquiler de un cliente (nombre del cliente, alquiler, articulo,mes,valor) |
| RF13 | Crear un endPont que permita listar el total de intereses recaudado en un mes y año |
| RF14 | Crear un endPoint que permita listar los meses y el interés pendiente por pagar de un alquiler. |
|  |  |

Comenzamos el proceso creando todas las carpetas necesarias y descargando los paquetes requeridos para la API

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente con confianza media

RF1 Crear un EndPoint que permita validar usuarios (identificación y password ) y generar un token

En primer lugar, establecemos la conexión a la base de datos y creamos una constante llamada 'validarCliente'. Luego, utilizamos una estructura 'try-catch' para recuperar la información de identificación y contraseña de la tabla 'clientes'. Si la sentencia tiene éxito, se mostrará un mensaje indicando que el cliente está autorizado. En caso contrario, se mostrará un mensaje que informa que el cliente no ha sido encontradoTexto

Descripción generada automáticamente

En esta sección, creamos la constante 'validar\_token' y utilizamos una estructura 'try-catch'. Validamos que el token del cliente sea requerido, y si no se proporciona el token, se generará un mensaje que indique que se necesita el token. Por otro lado, si se proporciona el token pero resulta ser inválido, se mostrará un mensaje que informa que el token no es válido. ¡Continuamos con la implementación! Captura de pantalla de computadora

Descripción generada automáticamente

Dentro de la carpeta 'routers', específicamente en clientes.jsla.Routes, se puede observar la importación de las constantes previamente definidas en clientes.controller.jsla, tales como 'registrar clientes', 'actualizar clientes', 'eliminar clientes' y 'listar clientes'. Además, se importa la ruta de la carpeta, así como las funciones de 'validar token' y 'validar usuario'. Se crea una nueva constante llamada 'clientesRouter', |

Texto

Descripción generada automáticamente

RF2 Crear un endPoint que permita registrar clientes con token

Comenzamos creando la constante registrar cliente Usamos una estructura 'try-catch' para gestionar la inserción de datos en la tabla de clientes mediante la sentencia 'insert into'. Incluimos todos los campos de la tabla, como 'identificación', 'nombres', 'dirección', 'teléfono', 'fecha' y 'password '. Si la operación tiene éxito, se devuelve un estado 200 con el mensaje se rigistro con éxito el cliente . En caso de un error en el servidor, se maneja con un estado 500 y se muestra un mensaje de error correspondiente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

RF3 Crear un endPoint que permita actualizar un cliente con token

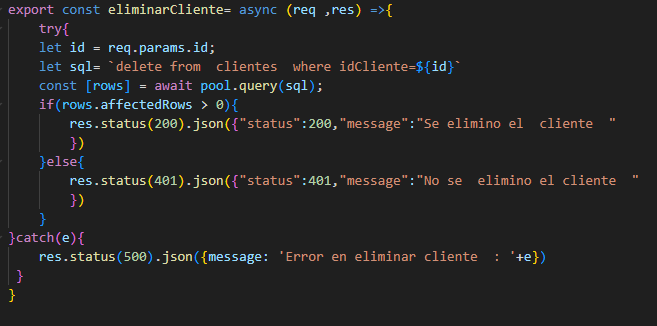
Primero lo que podemos hacer es utilizar el update para que actualice los datos del cliente en especifico para que solo se pueda actualizar un por uno lo que se actualizaría es el nombre la dirección el teléfono la fecha y respectiva mente su contraseña Si la operación tiene éxito, se devuelve un estado 200 con el mensaje se que se actualiza con éxito el cliente . En caso de un error en el servidor, se maneja con un estado 500 y se muestra un mensaje de error correspondiente

Texto

Descripción generada automáticamente

RF4 Crear un endPoint que permita eliminar un cliente con token

A este punto, ya hemos creado un cliente en la base de datos y ahora estamos listos para eliminarlo. Para llevar a cabo esta operación en la construcción de la API, utilizamos el método 'DELETE'. Sin embargo, es esencial proporcionar el ID del cliente que deseamos eliminar, ya que sin este parámetro podríamos eliminar datos necesarios accidentalmente. Además, para garantizar la seguridad y control de acceso, requerimos el uso de un token, lo que asegura que solo un administrador pueda realizar esta acción. Si la operación tiene éxito, se devuelve un estado 200 con el mensaje se que se elimino con éxito el cliente . En caso de un error en el servidor, se maneja con un estado 500 y se muestra un mensaje de error correspondiente



RF5 Crear un endPoint que permita listar los todos clientes

Para este endpoint, es necesario que ya tengamos clientes registrados en la base de datos para poder mostrarlos. Utilizamos el método 'GET' en este caso, que nos permite recuperar y mostrar todos los clientes registrados en la base de datos. En caso de éxito, responderemos con un estado 200 y una lista de todos los clientes que existen en la base de datos. Si no se obtiene este resultado, significa que ha ocurrido un error en el sistema o en el proceso de listado, en cuyo caso se generarán los mensajes de error correspondientes

Texto

Descripción generada automáticamente

Por último, en el módulo de clientes, es esencial contar con un archivo controlador que contenga todas las rutas necesarias para registrar, actualizar y listar clientes. Esto se hace para asegurar que cada funcionalidad esté organizada y no haya confusiones en el proceso de utilización de cada una de ellas

Texto

Descripción generada automáticamente

RF6 Crear un endPoint que permita registrar nuevos artículos con token

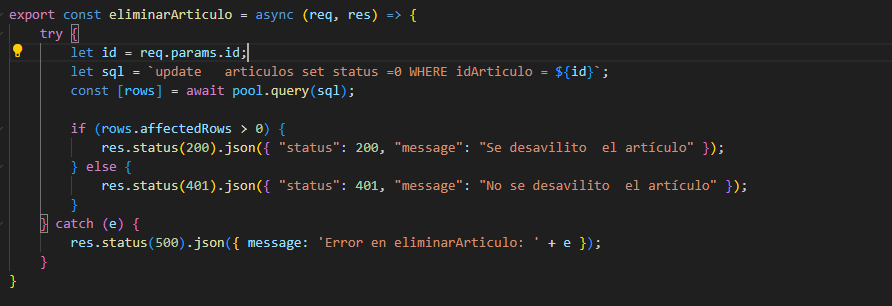
Iniciamos creando la constante 'registra articulo'. Utilizamos una estructura 'try-catch' para gestionar la inserción de datos en la tabla de artículos mediante la sentencia 'insert into'. Incluimos todos los campos necesarios, como 'nombre' y 'tipo'. Si la operación es exitosa, responderemos con un estado 200 y un mensaje que confirma el registro exitoso del artículo. Sin embargo, en caso de un error en el servidor, manejaremos la situación con estados 401 o 500, que indicarán errores de autenticación o problemas internos en la inserción de datos, respectivamente.

Captura de pantalla de computadora

Descripción generada automáticamente

RF7 Crear un endPoint que permita desactivar un artículo con token

En este contexto, utilizamos un campo de estado que inicialmente se establece en 1, lo que indica que el artículo está activo. Cuando realizamos una actualización que modifica este estado a 0, esto implica que el artículo ha sido desactivado. Esta funcionalidad se ha implementado para facilitar la desactivación de artículos y hacer el proceso más sencillo como se muestra a continuación



RF8 Crear un endPoint que permita registrar un alquiler con token }

Comenzamos creando la constante 'registrarAlquiler'. Utilizamos una estructura 'try-catch' para gestionar la inserción de datos en la tabla de alquiler mediante la sentencia 'insert into'. Todos los campos necesarios, como valor, fecha, meses, descripción, interés, así como las claves foráneas de las tablas de clientes y artículos (fkCliente y fkArticulo), se incluyen en esta operación.

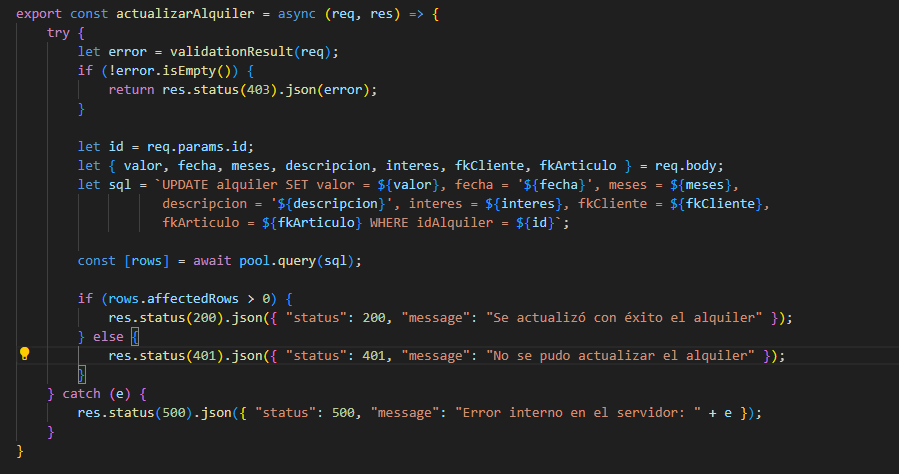
Si la operación se ejecuta con éxito, responderemos con un estado 200 y un mensaje que confirma el registro exitoso del alquiler. Sin embargo, en caso de un error en el servidor, manejaremos la situación con estados 401 o 500, que indicarán errores de autenticación o problemas internos en la inserción de datos, respectivamente.

Texto

Descripción generada automáticamente

RF9 Crear un endPoint que permita actualizar un alquiler con token

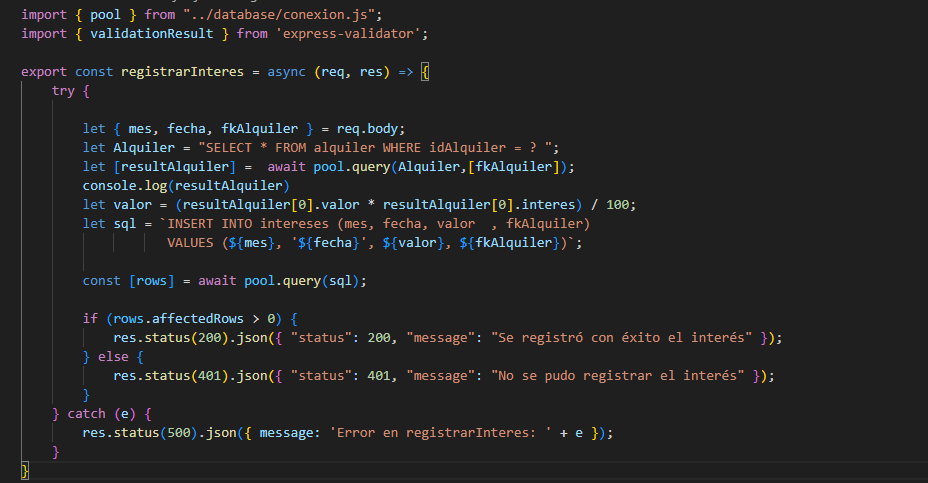
Primero lo que podemos hacer es utilizar el update para que actualice los datos del alquiler en especifico para que solo se pueda actualizar un por uno lo que se actualizaría es valor, fecha, meses, descripción, interés Si la operación tiene éxito, se devuelve un estado 200 con el mensaje se que Se actualizó con éxito el alquiler . En caso de un error en el servidor, se maneja con un estado 500 y 401 se muestra un mensaje de error correspondiente



RF10 Crear un endPoint que permita registrar un interés con token

Comenzamos creando la constante 'registrarInteres'. Utilizamos una estructura 'try-catch' para gestionar la inserción de datos en la tabla de intereses mediante la sentencia 'insert into'. Esta operación involucra campos esenciales, como mes, fecha, valor, descripción, interés, y también incluye la clave foránea de la tabla de alquiler (fkAlquiler). Durante esta operación, realizamos una multiplicación y división por 100 para ajustar el valor de acuerdo con el cálculo de intereses.

Si la operación se completa con éxito, responderemos con un estado 200 y un mensaje que confirma el registro exitoso del interés. Sin embargo, en caso de un error en el servidor, gestionaremos la situación con estados 401 o 500, que indicarán errores de autenticación o problemas internos en la inserción de datos, respectivamente.



RF11 Crear un endPoint que permita eliminar un interés con token

Hasta este punto, hemos creado un registro de interés en la base de datos y estamos preparados para eliminarlo. En la construcción de la API, utilizamos el método 'DELETE' para llevar a cabo esta operación. Es fundamental proporcionar el ID del interés que deseamos eliminar, ya que la omisión de este parámetro podría resultar en la eliminación accidental de datos importantes. Para garantizar la seguridad y el control de acceso, es necesario contar con un token, lo que asegura que solo un administrador puede realizar esta acción.

Si la operación se realiza con éxito, responderemos con un estado 200 y un mensaje que confirma la eliminación del interés. No obstante, en caso de un error en el servidor, gestionaremos la situación con estados 500 y 401, que indicarán errores internos o de autenticación, respectivamente, y se mostrará un mensaje de error correspondiente.

Pantalla de computadora

Descripción generada automáticamente con confianza media

RF12 Crear un endPoint que permita listar los interés pagados de los alquiler de un cliente (nombre del cliente, alquiler, articulo,mes,valor)

Para llevar a cabo esta operación en la tabla de intereses, es necesario realizar joins con tres tablas: la tabla de clientes para mostrar el nombre del cliente, la tabla de artículos para mostrar el artículo alquilado y la tabla de alquiler para obtener los valores correspondientes. Esta consulta se realiza a través del método 'GET' utilizando el ID del cliente como parámetro. El resultado es la presentación de todos estos datos en conjunto.

En caso de que no se muestre ningún resultado, se generará un mensaje de error en el servidor. A continuación, se muestra el código correspondiente.

Texto

Descripción generada automáticamente

RF13 Crear un endPont que permita listar el total de intereses recaudado en un mes y año

En este escenario, realizamos una consulta que se basa en el mes y el año para sumar el valor de los intereses correspondientes a ese período. El resultado de esta consulta es el cálculo del total de intereses para un mes y año específicos. La consulta se efectúa mediante el método 'GET', y proporciona información sobre los meses disponibles, los años y el valor total correspondiente a un mes y año determinados.

Si la consulta se lleva a cabo de manera exitosa, se mostrarán los meses disponibles, años y el valor total. Sin embargo, si no se encuentran intereses registrados para ese período, la respuesta será un estado 401, acompañado de un mensaje que indica la ausencia de intereses registrados. En caso de un error en el servidor, se manejará con un estado 500 y se mostrará un mensaje de error correspondiente

Texto

Descripción generada automáticamente

RF14 Crear un endPoint que permita listar los meses y el interés pendiente por pagar de un alquiler.

Finalmente, en esta parte, realizamos un JOIN con la tabla de alquiler para mostrar los meses y el valor de los intereses que están pendientes de pago. Además, proporciona información sobre el mes en curso. Cuando esta consulta se realiza exitosamente mediante el método 'GET', se presentan los meses, el mes actual y los valores correspondientes.

El código que ilustra esta funcionalidad se muestra a continuación

