Manual de usuario

A continuación, se presenta un manual de usuario que explica el funcionamiento de los códigos:

Iniciar sesión:

La ruta /login maneja las solicitudes de inicio de sesión. El código verifica si las credenciales de usuario y contraseña proporcionadas coinciden con las almacenadas en el array users. Si coinciden, se genera un token JWT y se devuelve al cliente. Si no coinciden, se devuelve un mensaje de error.

```
app.post('/login', (req, res) => {
  const { username, password } = req.body;
  const user = users.find(u => u.username === username && u.password === password);
  if (user) {
    const token = jwt.sign({ username }, secret, { expiresIn: '1h' });
    res.json({ token });
  } else {
    res.status(401).json({ message: 'Usuario o contraseña incorrectos' });
  }
});
```

Búsqueda de usuarios en GitHub:

La ruta /search maneja las solicitudes de búsqueda de usuarios en GitHub. El código verifica si el token JWT proporcionado por el cliente es válido. Si el token es válido, se realiza una solicitud a la API de GitHub para obtener información sobre los usuarios que coinciden con la consulta de búsqueda proporcionada. Luego, se devuelve la información de los usuarios al cliente.

```
app.get('/search', (req, res) => {
  const token = req.headers.authorization.split(' ')[1];
  jwt.verify(token, secret, (err, decoded) => {
    if (err) {
      res.status(401).json({ message: 'Token inválido o expirado' });
    } else {
      const query = req.query.q;
      axios.get(`https://api.github.com/search/users?q=${query}`)
        .then(response => {
          res.json(response.data);
        })
        .catch(error => {
          res.status(500).json({ message: 'Error al buscar usuarios en GitHub' });
        });
    }
  });
});
```

Interfaz de usuario:

El código HTML, CSS y JavaScript en el archivo index.html crea una interfaz de usuario que permite a los usuarios ingresar sus credenciales de inicio de sesión y realizar consultas de búsqueda de usuarios en GitHub. El Vue.js se utiliza para manejar el estado y las interacciones del usuario. Cuando el usuario ingresa sus credenciales y hace clic en "Ingresar", el código Vue.js realiza una solicitud a la ruta /login para verificar las credenciales. Si la verificación es exitosa, se establece el token JWT en el encabezado de la solicitud y se redirecciona a la ruta /search. En la ruta /search, el código verifica el token JWT y, si es válido, se realiza una solicitud a la API de GitHub para obtener información sobre los usuarios que coinciden con la consulta de búsqueda proporcionada. Luego, se muestra la información de los usuarios en la página index.html.

Página de contacto

Contiene un formulario con un servidor en Node.js.

Código del Servidor (Node.js)

Se utiliza Node.js con el framework Express y la base de datos MongoDB a través de Mongoose.

Configuración inicial

Se carga la configuración de las variables de entorno desde un archivo `.env` usando `require('dotenv').config()`.

Se importan los módulos necesarios: `express`, `body-parser` y `mongoose`.

Inicialización de la Aplicación.

Se crea una instancia de la aplicación Express: `const app = express()`.

Configuración de Middleware:

Se utiliza `bodyParser.urlencoded()` para analizar el cuerpo de las solicitudes entrantes con el tipo de codificación `urlencoded`.

Se especifica el directorio de archivos estáticos a través de `express.static('public')`.

Conexión a la Base de Datos:

Se establece la conexión a la base de datos MongoDB utilizando la variable de entorno `MONGO_URI`.

Definición del Esquema y Modelo:

Se define un esquema para los contactos que se almacenarán en la base de datos.

Se crea un modelo de Mongoose llamado `Contact` basado en el esquema definido.

Manejo de Solicitudes POST:

Se define una ruta `'/submit-form'` para manejar las solicitudes POST del formulario de contacto.

Cuando se recibe una solicitud, se crea un nuevo contacto utilizando los datos del formulario y se guarda en la base de datos.

Se envía una respuesta al cliente para confirmar que los datos se recibieron y guardaron con éxito.

Inicio del Servidor:

Se especifica el puerto en el que escuchará el servidor, utilizando el valor de la variable de entorno `PORT` o el puerto 3001 por defecto.

Se inicia el servidor y se imprime un mensaje en la consola para indicar que el servidor está en ejecución.

Código HTML

El código HTML proporciona la estructura del formulario de contacto, incluyendo campos para el correo electrónico, número, fecha, teléfono y mensaje. El formulario envía los datos a la ruta `'/submit-form'` utilizando el método POST.

El código del servidor utiliza Mongoose para interactuar con la base de datos MongoDB. Mongoose es una herramienta de modelado de objetos MongoDB para Node.js que proporciona una solución basada en esquemas para modelar los datos de la aplicación.