**DISEÑO DE SERVIDOR WEB**

**SNEYDER FABIÁN CÁCERES FUENTES**

**SERVICIO NACIONAL DE APRENDIZAJE SENA**

**PROGRAMA DE ANÁLISIS Y DESARROLLO DE SOFTWARE ADSO**

**FICHA: 2627044**

**NOMBRE DE LA INSTRUCTORA**

**ILMER CUELLO**

**10 DE MARZO DE 2024**

Tabla de contenido

[INTRODUCCIÓN 3](#_Toc160957982)

[OBJETIVOS 4](#_Toc160957983)

[DESARROLLO DEL TRABAJO 5](#_Toc160957984)

[CONCLUSION 8](#_Toc160957985)

# INTRODUCCIÓN

En la actual era digital, la seguridad y la autenticación son aspectos críticos para cualquier aplicación web que maneje información confidencial de los usuarios, con el fin de abordar estas necesidades, se ha desarrollado un servidor web que ofrece funcionalidades de registro e inicio de sesión de usuarios de manera segura y eficiente.

Este servidor web, construido utilizando tecnologías modernas como Node.js, Express.js y MongoDB, proporciona una plataforma robusta para gestionar el proceso de registro e inicio de sesión de usuarios. A través de este servicio, los usuarios pueden crear cuentas seguras y acceder a ellas de manera confiable, garantizando la privacidad y la integridad de sus datos personales.

# OBJETIVOS

**Objetivo general**

* Desarrollar un servidor web seguro y eficiente para gestionar el registro e inicio de sesión de usuarios.

**Objetivos específicos**

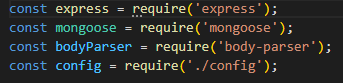
* Crear un servicio web para registrar y autenticar usuarios.
* Validar credenciales de usuario.
* Proporcionar respuestas claras de autenticación.

# DESARROLLO DEL TRABAJO

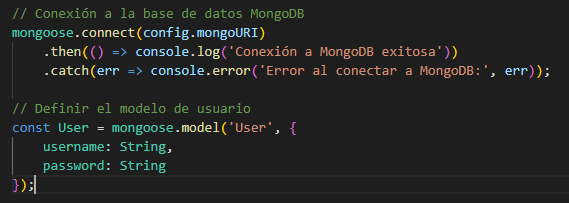
En el presente documento usted encontrara un servidor web el cual fue codificado según las diferentes convecciones y necesidades de los interesados del proyecto.

A continuación, mostremos una a una las líneas de código y se dará una corta justificación del uso de las misma.

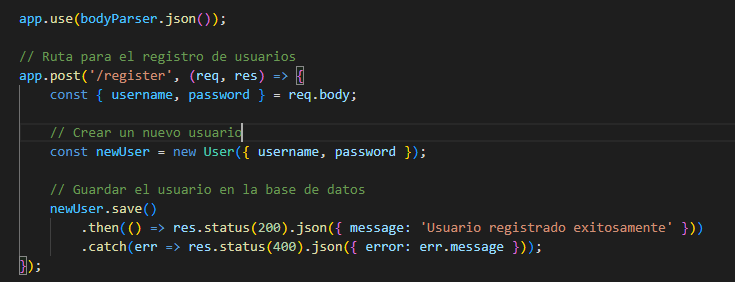
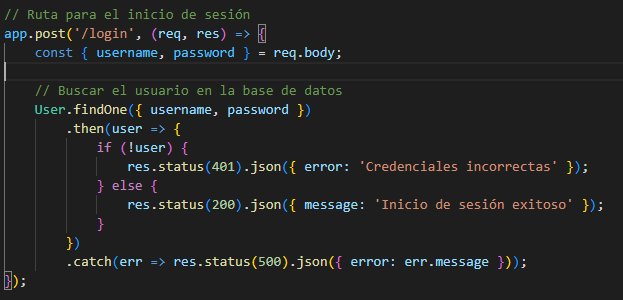
Lo primero es importar los módulos esenciales en una aplicación de Node.js. express se utiliza para crear el servidor web y manejar las rutas, mongoose facilita la interacción con la base de datos MongoDB, body-parser analiza los cuerpos de las solicitudes HTTP para acceder a los datos enviados, y config importa un archivo de configuración que contiene variables específicas de la aplicación.



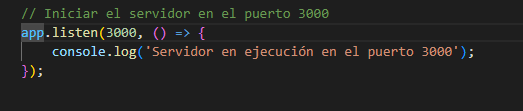
Posteriormente, Estas líneas de código inicializan una instancia de la aplicación Express y definen un modelo de datos de usuario utilizando Mongoose, además, establecen una conexión a la base de datos MongoDB utilizando la URI de conexión proporcionada en la configuración, esto asegura que la aplicación esté lista para manejar solicitudes HTTP y pueda interactuar con la base de datos para almacenar y recuperar información de usuarios.



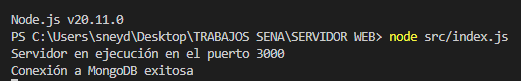
Estas líneas de código definen una ruta en la aplicación Express para el registro de usuarios. Utilizando el método app.post(), se establece una ruta POST en la ruta '/register', la cual espera recibir datos en formato JSON con un nombre de usuario (username) y una contraseña (password) a través del cuerpo de la solicitud (req.body).

Dentro de la función de manejo de la ruta, se crea un nuevo objeto de usuario utilizando el modelo definido anteriormente, se guarda este nuevo usuario en la base de datos utilizando el método save() proporcionado por Mongoose.   
Estas líneas de código definen un manejador de ruta en una aplicación Node.js con Express, maneja solicitudes POST a '/login', esperando un cuerpo JSON con 'username' y 'password', extrae estos campos, busca un usuario en la base de datos que coincida con ello, si encuentra un usuario, responde con un mensaje de éxito (código 200); de lo contrario, responde con un error de credenciales incorrectas (código 401). Si hay un error durante el proceso, responde con un error de servidor (código 500). 

Ya solo nos queda ejecutar el servidor en el puerto 3000 y la lkinea de codigo nos debe mostrar en la termianl que se ejecuto el servidor corresctamente.



Ejecutamos el codigo y nos sale el siguente resutlado en teminal.



Para que usted pueda verificar el codigo fuente que presentamos anteriormente, ingrese al siguente link: <https://github.com/FabianCF20/ServidorWebAppi/tree/main/src>

# CONCLUSION

El código presentado demuestra un enfoque efectivo para desarrollar un servicio web que aborda dos funciones esenciales en el contexto de aplicaciones de autenticación de usuarios: registro y inicio de sesión. Utilizando Node.js en combinación con Express para la creación del servidor y MongoDB como base de datos, se construye un sistema que facilita la interacción entre el cliente y el servidor a través de solicitudes HTTP.

Este servicio web ofrece una arquitectura escalable y eficiente, permitiendo la gestión de múltiples solicitudes de forma simultánea. La implementación del modelo de usuario mediante Mongoose proporciona una estructura sólida para la gestión de datos, garantizando la persistencia de la información de manera segura.

Además, se presta especial atención a la seguridad, dado que la autenticación de usuarios es un aspecto crítico en cualquier aplicación web. El sistema valida las credenciales proporcionadas por los usuarios durante el proceso de inicio de sesión, lo que contribuye a proteger la integridad de los datos y a prevenir posibles ataques maliciosos.