** Aplicaciones Web Progresivas**

**Uso de base de datos y autentificación**

**400 Bad Request**

**Integrantes**

**Juan Roberto Adolfo Esquivel Segovia 18505**

**Laura Janett Gonzalez Cruz 21143**

**Fernanda Lizeth García Antonio 21139**

**Danna Panela Yoselin Guerreo Álvarez 21084**

**Pedro Miguel Hernández Sánchez 21042**

**Grupo:**

**10°E**

**Maestra: Xóchilt Clara Villar Diego**

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.  
 **Imagen 3 – Inserción y verificación de datos**En esta imagen se observa la sección del código donde se insertan datos de prueba iniciales dentro de la tabla usuarios. Se agregan dos registros: un Administrador General con correo y un Usuario de Prueba. Finalmente, se ejecuta la instrucción SELECT \* FROM usuarios; para verificar que los registros se hayan insertado correctamente en la base de datos.  
  
Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.  
Esta imagen muestra la base de datos vitalence\_db dentro de phpMyAdmin.  
Contiene cinco tablas principales: dispositivos, pacientes, signos\_vitales, tokens y usuarios. Cada una almacena información específica del sistema Vitalence, como datos de usuarios, pacientes, dispositivos médicos y registros de signos vitales para el control y monitoreo de la salud.

Texto

El contenido generado por IA puede ser incorrecto.

Explicación del funcionamiento del código de inicio de sesión

El código presentado corresponde a un script en JavaScript que tiene como finalidad controlar el proceso de inicio de sesión (login) de un usuario dentro de una aplicación web. Su funcionamiento se basa en la interacción entre el formulario del navegador y una API alojada en el servidor.

En primer lugar, se obtiene una referencia al formulario HTML mediante el método getElementById("loginForm"). Posteriormente, se agrega un escuchador de eventos que detecta cuando el usuario presiona el botón de envío. Este evento se maneja de forma asíncrona y se utiliza la función preventDefault() para evitar que la página se recargue automáticamente.

Una vez interceptado el envío, el código obtiene los valores de los campos de correo electrónico y contraseña del formulario. Estos datos se convierten a formato JSON y se envían al servidor mediante una petición HTTP de tipo POST utilizando la función fetch(). La dirección del servidor a la que se envían los datos es http://localhost:3000/api/login, y la cabecera Content-Type: application/json indica que el contenido transmitido está en formato JSON.

El servidor recibe la solicitud y responde con un objeto JSON que contiene información sobre el resultado del intento de inicio de sesión. El código interpreta esta respuesta y valida si la autenticación fue exitosa mediante la propiedad res.ok. Si ocurre un error (por ejemplo, si las credenciales son incorrectas), se muestra un mensaje de alerta en pantalla con el texto "Error al iniciar sesión" o con el mensaje devuelto por el servidor.

En caso de que el inicio de sesión sea correcto, el programa almacena los datos del usuario en el almacenamiento local del navegador (Local Storage) utilizando el método localStorage.setItem(). De esta manera, se guardan el nombre, el identificador y el rol del usuario autenticado. Esta información permite mantener la sesión activa incluso si el usuario recarga la página o cierra temporalmente la pestaña del navegador.

En resumen, este código implementa de forma sencilla un sistema de autenticación en el lado del cliente, conectándose con una API en el servidor para validar las credenciales del usuario y guardar los datos necesarios para mantener la sesión iniciada dentro de la aplicación web.