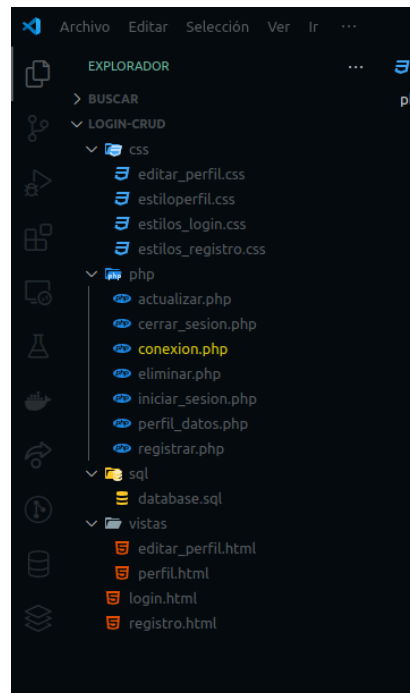


Documentación del Proyecto: Sistema de Gestión de Usuarios en PHP y MySQL

1. Introducción

Este proyecto es un sistema básico de gestión de usuarios que permite a los usuarios registrarse, iniciar sesión, ver su perfil, actualizar sus datos y eliminar su cuenta.

2. Estructura del Proyecto



3. Funcionalidad de los Archivos

3.1 Base de Datos (database.sql)

Define la estructura de la base de datos en MySQL.

Contenido principal:

- Creación de la base de datos login_system.
- Creación de la tabla usuarios con los siguientes campos:
 - id (clave primaria, autoincremental).
 - nombre.
 - correo (único).
 - telefono.
 - carrera.

- contraseña (almacenada con password_hash).

3.2 Conexión a la Base de Datos (conexion.php)

Maneja la conexión con MySQL.

Funcionalidad:

- Se conecta al servidor MySQL usando mysqli.
- Utiliza las credenciales definidas (\$servername, \$username, \$password, \$dbname).
- Si la conexión falla, muestra un error.

3.3 Registro de Usuario (registrar.php)

Procesa el registro de nuevos usuarios.

Flujo de trabajo:

1. Recibe los datos enviados desde el formulario.
2. Hashea la contraseña con password_hash().
3. Inserta el usuario en la base de datos.
4. Redirige a login.html si el registro es exitoso.

3.4 Iniciar Sesión (iniciar_sesion.php)

Verifica las credenciales y crea una sesión.

Flujo de trabajo:

1. Recibe correo y contraseña desde el formulario de login.
2. Busca el usuario en la base de datos.
3. Usa password_verify() para comprobar la contraseña.
4. Si es correcta, guarda usuario_id en \$_SESSION y redirige a perfil.html.
5. Si falla, muestra un mensaje de error.

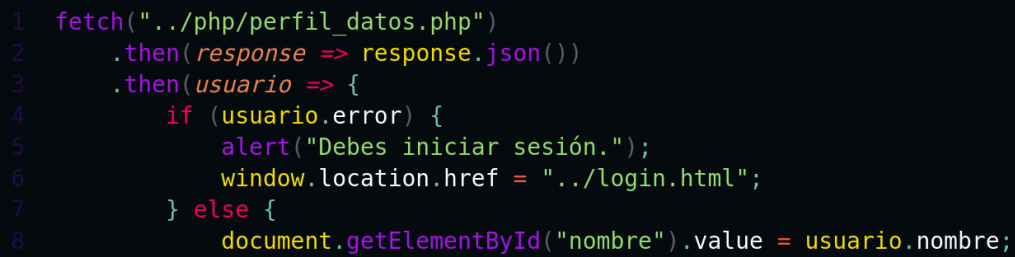
3.5 Perfil del Usuario (perfil.html)

Muestra la información del usuario autenticado.

Funcionalidad:

- Carga los datos del usuario usando perfil_datos.php.
- Muestra nombre, correo, teléfono y carrera.
- Permite editar perfil, cerrar sesión y eliminar cuenta.

Código clave:



```
1 fetch("../php/perfil_datos.php")
2   .then(response => response.json())
3   .then(usuario => {
4     if (usuario.error) {
5       alert("Debes iniciar sesión.");
6       window.location.href = "../login.html";
7     } else {
8       document.getElementById("nombre").value = usuario.nombre;
```

3.6 Obtener Datos del Perfil (perfil_datos.php)

Devuelve los datos del usuario en formato JSON.

Flujo de trabajo:

1. Verifica si el usuario tiene una sesión activa (\$_SESSION['usuario_id']).
2. Busca los datos del usuario en la base de datos.
3. Devuelve los datos en formato JSON.

3.7 Editar Perfil (editar_perfil.html)

Formulario para actualizar los datos del usuario.

Funcionalidad:

- Usa fetch() para precargar los datos del usuario.
- Permite modificar nombre, teléfono y carrera.
- Al enviar, redirige a actualizar.php.

```

1      <form id="form-editar" action="../../php/actualizar.php" method="POST">
2          <label for="nombre">Nombre:</label>
3          <input type="text" id="nombre" name="nombre" required>
4
5          <label for="telefono">Teléfono:</label>
6          <input type="text" id="telefono" name="telefono" required>
7
8          <label for="carrera">Carrera:</label>
9          <input type="text" id="carrera" name="carrera" required>
10
11         <button type="submit">Guardar Cambios</button>
12     </form>
13     <a href="perfil.html">Volver</a>
14 </div>

```

3.8 Actualizar Perfil (actualizar.php)

Guarda los cambios del perfil en la base de datos.

Flujo de trabajo:

1. Verifica si el usuario tiene sesión.
2. Recibe los datos (nombre, teléfono, carrera).
3. Ejecuta UPDATE usuarios SET ... WHERE id = \$_SESSION['usuario_id'].
4. Redirige a perfil.html.

```

1 $sql = "UPDATE usuarios SET nombre='$nombre', telefono='$telefono', carrera='$carrera' WHERE id=$usuario_id";
2 if ($conn->query($sql) === TRUE) {
3     echo "<script>
4         alert('✅ Datos actualizados correctamente. ');
5         window.location.href = '../vistas/perfil.html';
6     </script>";
7 } else {
8     echo "<script>
9         alert('❌ Error al actualizar: " . $conn->error . " ');
10        window.location.href = '../vistas/perfil.html';
11    </script>";
12 }
13 ?>

```

3.9 Eliminar Cuenta (eliminar.php)

Borra la cuenta del usuario y cierra su sesión.

Flujo de trabajo:

1. Verifica si el usuario tiene sesión activa.
2. Elimina el usuario con DELETE FROM usuarios WHERE id = \$_SESSION['usuario_id'].
3. Cierra la sesión con session_destroy().
4. Redirige a login.html.

```
1  $usuario_id = $_SESSION['usuario_id'];
2  $sql = "DELETE FROM usuarios WHERE id=$usuario_id";
3
4  if ($conn->query($sql) === TRUE) {
5      session_destroy();
6      echo "
7          <script>
8              alert('✓ Tu cuenta ha sido eliminada correctamente.');
```

```
9              window.location.href = '../login.html';
10             </script>
11         ";
12         exit();
13     } else {
14         echo "
15             <script>
16                 alert('✗ Error al eliminar la cuenta. Inténtalo de nuevo.');
```

```
17                 window.location.href = '../vistas/perfil.php';
18             </script>
19         ";
20     }
21
22     $conn->close();
23     ?>
24
```

3.10 Cerrar Sesión (cerrar_sesion.php)

Destruye la sesión del usuario y lo redirige al login.



```
1  <?php
2  session_start();
3  session_destroy();
4  header("Location: ../login.html");
5  exit();
6  ?>
```

4. Conclusión

Este sistema de gestión de usuarios en PHP y MySQL permite a los usuarios registrarse, iniciar sesión, visualizar su perfil, actualizar sus datos y eliminar su cuenta de manera sencilla y segura. Principalmente se usó docker como base de datos usando la Imagen de Mysql pero también se puede usar localhost como comúnmente se hace