



**UNIVERSIDAD  
NACIONAL AUTONOMA  
DE MEXICO**



**FACULTAD DES ESTUDIOS SUPERIORES  
ARAGON**

**SISTEMAS DE INFORMACION**

**REPORTE PROYECTO FINAL**

**PROF. VELASCO AGUSTÍN AARON**

**ALUMNOS:**

**FRAGOZO RODRIGUEZ OSWALDO**

**GARCIA CHAVEZ TOMAS ADRIAN**

**RIOS CORTES JUAN ULISES**

## CONTENIDO

1. INTRODUCCIÓN.....	3
2. ACCESO DENTRO DEL SISTEMA.....	3
2.1. Índice.php.....	3
2.2. verificacion.php .....	4
3. LECTURA DE REGISTROS DE UNA BASE DATOS. ....	5
3.1. consulta.php .....	5
3.2. conexión.php .....	8
3.3. salir.php.....	8
3. INSERTAR REGISTROS A UNA BASE DE DATOS.....	8
3.1 registro.php.....	9
3.2 logica_registro.php .....	10
4. ELIMINAR A LOS USUARIOS REGISTRADOS. ....	11
4.1 eliminar.php .....	11
4.2 logica_eliminar.php.....	13
5. Carpeta Assets. ....	13
5.1 header.php .....	13
5.2 estilo.php.....	14
6. VIRTUALIZACIÓN DEL SISTEMA.....	15
MySQL.....	15
XAMPP .....	16
VISUAL STUDIO CODE.....	17
7. IMPLEMENTACION DE NUESTRO SISTEMA DE INFORMACIÓN.....	18
8. CONCLUSION. ....	22

## 1. INTRODUCCIÓN.

Realizamos un sistema de información básico, el cual cumple con los principios de CRUD (crear, leer, actualizar y borrar), en base a los conocimientos adquiridos en la materia de Sistemas de información y recursos adquiridos en la búsqueda de problemas, obtuvimos un resultado satisfactorio, y por consiguiente se procede a documentarlo en este archivo.

Cabe resaltar, que el documento esta dividido en módulos requeridos en las especificaciones del proyecto y a su vez, en los archivos que los componen.

## 2. ACCESO DENTRO DEL SISTEMA.

Este primer módulo, nos pide la generación de un medio de acceso dentro del sistema para poder acceder al listado de registros almacenados en la base de datos, y también nos da el acceso a todo lo que conlleva el sistema de información, excluyendo este mismo modulo, ósea, nos permite modificar la base de datos y hacer uso del sistema de información en su totalidad.

A continuación, se presentan los archivos que conforman a esta sección y una breve explicación de su lógica e implementación.

### 2.1. Índex.php

El archivo **index.php** nos permitirá visualizar el login del sistema en donde se nos permitirá ingresar a este por medio de los dos campos que utiliza usuario y contraseña.

En este se utiliza el método POST el cual envía la información de manera invisible esto que quiere decir, que se envía a través del 'body' del 'HTTP request', por lo que no aparece en la url.

También contiene dos botones:

- **Iniciar sesión.**  
Es de tipo submit la cual envía directamente la información solicitada buscado en la DB la existencia del usuario y/o contraseña para redireccionarte a la página buscada y en caso de no encontrarla mandarte a una página de error, el proceso que se realiza aquí se especifica en **verificacion.php**.
- **Registrarse:**  
Este nos redirecciona directamente a otra ventana en la cual se nos permite realizar un registro, este proceso se especifica en **registro.php** y **logica\_registro.php**.

```

<!DOCTYPE html>
<html lang="en">

<head>
  <link rel="stylesheet" type="text/css" href="assets/estilo.php">
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
  <link rel="Shortcut icon" href="assets/coders2.png">
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Login</title>
</head>

<body class="body1">
  <?php
  include("assets/header.php");
  >
  <header class="shield">

    <svg class="bi bi-shield-lock-fill" width="5em" height="5em" viewBox="0 0 16 16" fill="white" xmlns="http://www.w3.org/2000/svg">
      <path fill-rule="evenodd" d="M5.187 1.025C6.23 7.337 5.8 .562 0 1.772 2.813 5.25a61.09 61.09 0 012.772 8.15c.528 6.908 0 01-1.007-.586 11.192 0 01-2.417-2.3C2.167 10.331 8.39 7.221 1.412 3.024A1.454 1.454 0 012.415 1.84a61.11 61.11 0 00-.037-.122z" clip-rule="evenodd" />
    </svg>
    <h2 class="h21" style="margin-left: 15px">LOGIN</h2>
  </header>
  <div class="div1" style="margin-top:50px">
    <div class="div2">
      <div align="center">
        <form id="form1" method="POST" action="./logica/verificacion.php" class="form1">
          <div class="div2">
            <label class="data">Usuario</label>
            <input type="text" name="txtusuario" class="text1" data="placeholder=" Nombre de usuario " required/>
          </div>
          <div class="div2">
            <label class="data">Contraseña</label>
            <input type="password" name="txtpassword" class="text1" data="placeholder=" Contraseña de usuario " required/>
          </div>
          <br>
          <input type="submit" value="INICIAR SESION" class="btn1" />
          <input type="button" onclick="location.href='registro.php'" value="REGISTRARSE" class="btn1">
        </form>
      </div>
    </div>
  </div>
</body>
</html>

```

## 2.2. verificacion.php

Desde el login al enviar datos por medio del botón iniciar sesión se realiza la comprobación desde **verificación.php** donde los datos proporcionados por usuario son verificados primero definiendo que \$user: obtendrá la información del campo usuario y \$pass: obtendrá la información en el campo contraseña, luego escribimos la consulta con \$query para contar en que índice hay un usuario y contraseña iguales, se manda a llamar la variable global \$consulta que guardara los parámetros en otro archivo y luego se obtendrá el resultado de la fila guardando el índice si se encontró en \$array.

Si se encontró una coincidencia te redireccionara a la página **consulta.php** asociada a este, en caso contrario o en caso de haber ingresado información errónea te redirecciona a una página de error.

\$user: obtiene la información solicitada en usuario.

\$pass: obtiene la información solicitada en contraseña.

\$query: nos ayuda a escribir la consulta a la base de datos.

\$consulta: variable global para guardar parámetros.

\$array: obtiene el resultado del índice de la fila deseada si es que se encontró.

```

arrollo-app > logica > verification.php > ...
<?php
require "../conexion.php";
session_start();

$user=$_POST["txtusuario"];
$pass=$_POST["txtpassword"];

$query= "SELECT COUNT(*) AS contar FROM usuarios WHERE user ='$user' and PASSWORD = '$pass'";

$consulta=mysqli_query($conn, $query);

$array=mysqli_fetch_array($consulta);

if($array['contar'] > 0){
    //header("Location: pagina.html")
    $_SESSION['noCuenta']=$user;
    header("Status: 301 Moved Permanently");
    // header("Location: https://crud-fesa.000webhostapp.com/consulta.php");
    header("Location: ../consulta.php");
    exit;
}
else {
    echo<script type="text/javascript">
    alert("Error en usuario / contraseña, intenta de nuevo. ");
    window.location.href="../index.php";
    </script>';
}
?>

```

### 3. LECTURA DE REGISTROS DE UNA BASE DATOS.

En el módulo actual veremos que es posible hacer una consulta a un sistema de información por medio de una conexión a MySQL utilizando un archivo específico que definimos como **conexión.php**, también anexamos las capturas del código necesario para que todo esto sea posible.

#### 3.1. consulta.php

**consulta.php** nos mostrará los usuarios registrados en la base de datos y establecerá la variable superglobal `$_SESSION` con los datos de la sesión iniciada. Cuando PHP se cierra, automáticamente toma el contenido de la variable superglobal `$_SESSION`, la serializa, y la envía para almacenarla usando el gestor de almacenamiento de sesiones.

Se establece la conexión con la base de datos y se guarda un query que seleccione la tabla “usuarios” de la BD en `$consulta_sql` y luego se manda por medio de la conexión y se almacena en una variable `$resultado`, al introducir el resultado en `$count` este nos regresara el número de filas del resultado. Si encuentra más de uno, lo usamos para imprimir el resultado en nuestra tabla mediante un `while` y mostrara tres botones:

- **AGREGAR USUARIO**

Este botón nos redireccionara a **registro.php** para poder agregar un usuario a la tabla.

- **ELIMINAR USUARIO**

Este botón nos redireccionara a **eliminar.php** donde se realiza el proceso para eliminar un usuario.

- **SALIR**

Este nos redirecciona a **salir.php** el cual se muestra más adelante.

En caso de que no encuentre más de un usuario mostrara el mensaje sin ningún registro y de nuevo el botón SALIR.

\$consulta\_sql: Guarda el query que selecciona la tabla `usuarios` en la BD.

\$resultado: Manda el query por medio de la conexión y se almacena en esta variable para poder imprimir.

\$count (mysqli\_num\_rows): Regresara el número de filas en el resultado. El comportamiento de mysqli\_num\_rows (rows ()) depende de si es que se utilizan resultsets con o sin buffer. En caso de emplearlos sin buffer mysqli\_num\_rows (rows ()) no retornará el número de filas correcto hasta que todas las filas del resultado hayan sido recuperadas.

while (\$row = mysqli\_fetch\_assoc(\$resultado)) {.....Pinta nuestra tabla por medio de la varibale \$row que contiene mysqli\_fetch\_assoc de la cual retorna un array asociativo correspondiente a la fila obtenida o NULL si no hubiera más filas.

```

<?php
session_start();
$carry = $_SESSION['noCuenta'];

if (!isset($carry)) {
    header("Status: 301 Moved Permanently");
    //header("Location: https://crud-fesa.000webhostapp.com/index.php");
    header("Location: index.php");
    exit;
} else {
    echo '<link rel="Shortcut icon" href="assets/coders2.png">
    <link rel="stylesheet" type="text/css" href="assets/estilo.php">
    <title>Consulta</title>';

    include("assets/header.php");
    echo "<br>";
    echo "<p style='margin-left:5%'>Bienvenido: $carry</p>";

    //se usa el require para requerir obligatoriamente el archivo conexion.php
    //no es requisito obligatorio, independiente de los errores
    require("conexion.php");
    //$conexion = new mysqli('127.0.0.1', 'root', '', 'php_test');
    //generar el query
    $consulta_sql = "SELECT * FROM USUARIOS";
    //mandar el query por medio de la conexion y almacenaremos en una variable
    $resultado = $conn->query($consulta_sql);
    //retorna el numero de filas del resultado. Si encuentra más de una fila
    $count = mysqli_num_rows($resultado);

    if ($count > 0) {
        echo "<h1 class='h21'>REGISTROS</h1>";

        echo "<body class='sansserif'>
        <div align='center' style='overflow-x:auto'>
        <table>
        <tr>
        <th>ID</th>
        <th>Usuario</th>
        <th>Nombre completo</th>
        <th>Correo</th>
        <th>Contraseña</th>
        <th>Fecha de registro</th>
        <th>Nivel de permisos</th>

        ";

        //aquí se pintarían los registros de la BD
        while ($row = mysqli_fetch_assoc($resultado)) {
            echo "<tr>";
            echo "<td>" . $row['id'] . " " . "</td>";
            echo "<td>" . $row['user'] . " " . "</td>";
            echo "<td>" . $row['nombre'] . " " . "</td>";
            echo "<td>" . $row['correo'] . " " . "</td>";
            echo "<td>" . $row['password'] . " " . "</td>";
            echo "<td>" . $row['date'] . " " . "</td>";
            echo "<td>" . $row['perm'] . " " . "</td>";
            echo "</tr>";
        }
        echo "</table><br>";
    }
}

```

### 3.2. conexión.php

Como se mencionó anteriormente este archivo hace posible la conexión a la base de datos que tenemos en el gesto MySQL además de que regresa un mensaje de error si es que por alguna razón no hay conexión, **conexión.php** es usado de manera global en todos los archivos que requieren la conexión a la base de datos.

```
<?php
$host_db="localhost";
$user_db="root";
$password_db="";
$name_db="php_test";

// $user_db="id13837169_crudb";
// $password_db="]fY9M=ZY^WCo-N)%";
// $name_db="id13837169_coders";

$conn=new mysqli($host_db,$user_db,$password_db,$name_db);

if(!$conn){
    die("No hay conexion RIP: ".mysqli_connect_error());
}
```

### 3.3. salir.php

En **salir.php** se terminará la sesión iniciada y nos regresará a **index.php**

```
<?php
session_start();
session_destroy();
header("Status: 301 Moved Permanently");
//header("Location: https://crud-fesa.000webhostapp.com/index.php");
header("Location: ../index.php");
exit;
exit(); //buena practica
?>
```

## 3. INSERTAR REGISTROS A UNA BASE DE DATOS.

Este módulo es parecido al anterior, ya que como se dijo, es necesario de aquí en adelante el archivo **conexión.php** que es el enlace con la base de datos que tenemos, solo que esta vez en lugar de solo consultar información de nuestro sistema, también vamos a mandar información para registrarla, y para esto son necesarios los siguientes archivos.



### 3.1 registro.php

Este nos permite realizar el registro de un nuevo usuario utilizando el método POST de la cual se requiere para insertar los nuevos datos a la base de datos.

En este caso los datos que se requerirán son:

- Usuario
- Nombre
- Email
- Contraseña

También contiene dos botones:

- Aceptar  
Esta manda la información proporcionada por el usuario a **logica\_registro.php** y este a su vez insertara los datos a la BD.
- Borrar campos  
Este botón restablecerá todos los campos para poder volver a llenarlos.

```
<!DOCTYPE html>
<html lang="en">
<head>
<link rel="stylesheet" type="text/css" href="assets/estilo.php">
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
<link rel="Shortcut icon" href="assets/coders2.png">
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Registro</title>
</head>
<body class="body1">
<?php
include("assets/header.php");
?>
<header class="shield">
<svg class="bi bi-person-plus" width="5em" height="5em" viewBox="0 0 16 16" fill="white" xmlns="http://www.w3.org/2000/svg">
<path fill-rule="evenodd" d="M11 14s1 0 1-1-4-6-4-6 3-6 4 1 1 1h10zm-9.995-.944v-.002.002M1.022 13h9.956a.274.274 0 00.014.014" />
<path fill-rule="evenodd" d="M13 7.5a.5.5 0 01.5-.5h2a.5.5 0 010 1H14v1.5a.5.5 0 01-1 0v-2z" clip-rule="evenodd" />
</svg>
<h2 class="h21" style="margin-left: 30px">REGISTRO</h2>
</header>
<div class="div1" style="margin-top:50px">
<div align="center">
<form action="logica/logica_registro.php" method="POST" class="form1">
<div class="div2">
<label for="nombre_usuario" class="data"> Usuario: </label><br>
<input type="text" class="text1 data" name="nombre_usuario" placeholder=" Nombre de Usuario" maxlength="155" required>
<br><br>
<label for="nombre_completo" class="data"> Nombre: </label><br>
<input type="text" class="text1 data" name="nombre_completo" maxlength="255" placeholder=" Nombre completo" required>
<br><br>
<label for="email" class="data"> Email: </label><br>
<input type="email" class="text1 data" name="email" maxlength="35" placeholder=" Correo electronico" required>
<br><br>
<label for="password" class="data"> Contraseña: </label><br>
<input type="password" class="text1 data" name="password" maxlength="10" placeholder=" Contraseña elegida" required>
<br><br>
</div>
<input type="submit" name="submit" value="ACEPTAR" class="btn1">
<input type="reset" name="clear" value="BORRAR CAMPOS" class="btn1">
</form>
</div>
</div>
</body>
</html>
```

## 3.2 logica\_registro.php

Lo primero que observamos en el código es que requiere conexión con la base de datos y esta se ve en **conexión.php**, con `mysqli_set_charset` estableceremos el conjunto de caracteres para mandar datos a la BD y declararemos las variables para el formulario con `$form_pass`, luego se escribe una consulta con `$buscarUsuario` y realizamos la consulta con `$result` y el resultado se guardara en una variable `$count`, esta variable pasa a ser validad mediante un `if`, donde si ya se encuentra algún registro nos mandara un error y nos dirá que el usuario ya se encuentra registrado, también aparecerá un botón “NUEVO REGISTRO” que nos volverá a redireccionar a **registro.php**, en caso contrario se realizara una query para registrar el usuario en la base de datos y mostrara un mensaje de usuario registrado exitosamente, Bienvenido “nombre de usuario”, también hay un botón “INICIAR SESION” que nos va a redirigir de nuevo al **index.php**.

- `mysqli_set_charset`: establece el conjunto de caracteres predeterminado a usar cuando se envían datos desde **registro.php** y hacia el servidor de la base de datos.
- `$form_pass`: solicita los datos del formulario.
- `$buscarUsuario`: escribe la consulta de búsqueda en la base de datos para saber si existe el nombre de usuario por el método `$_POST`.
- `$result`: manda la consulta a la base de datos.
- `mysqli_num_rows`: está función devuelve el índice de la fila del conjunto. `$count` de la cual contiene la función si encuentra un resultado idéntico al nombre de usuario lanza un mensaje de error de la cual muestra que ya existe ese nombre, pero si no se encuentra mandamos el query de la cual se hace la consulta de insertar nuevos datos en la base de datos mandando un mensaje de que se realizó con éxito el registro.

```

<?php
echo<link rel="stylesheet" type="text/css" href="../../assets/estilo.php">
<link rel="Shortcut icon" href="../../assets/coders2.png">;
require "../../conexion.php";
//conexion a la DB

mysqli_set_charset($conn, 'utf8');
//declaracion de variables para formulario
$form_pass= $_POST ['password'];
//validacion si el usuario ya esta registrado
$buscarUsuario="SELECT * FROM usuarios WHERE user = '$_POST[nombre_usuario]'";
//mandar al query
$result=$conn->query($buscarUsuario);
//saber si tenemos un resultado y se almacena en una variable
$count=mysqli_num_rows($result);
//se hace la validacion del resultado para saber si esta registrado
if ($count == 1 ) { //si se encuentra algun reegistro
    echo<body class='sansserif'>
    <title>ERROR</title>
    <div align='center'>
    <h1><br><br><br>El Usuario: " $_POST['nombre_usuario']. "
    <br>
    Ya esta registrado</h1>
    <br>
    <input type='button' style='font-size:.6em' onclick='location.href="../../registro.php`' value='NUEVO REGISTRO' class='btn1'>

    </div>
    </body>;
}
else{//se registra al usuario
    //query para mandar el registro
    mysqli_query($conn, "INSERT INTO USUARIOS (
    user,
    nombre,
    correo,
    password)
    VALUES(
    '$_POST[nombre_usuario]',
    '$_POST[nombre_completo]',
    '$_POST[email]',
    '$_POST[password]'

    )");
    echo<body class='sansserif'>
    <title>REGISTRADO</title>
    <div align='center'>
    <h1> <br><br><br>Usuario registrado exitosamente</h1> <br>
    <h2>Bienvenido: " $_POST['nombre_usuario']. "</h2> <br>
    <input type='button' style='font-size:.6em' onclick='location.href="../../index.php`' value='INICIAR SESION' class='btn1'>
    </div>
    </body>
    ";
}
}

```

## 4. ELIMINAR A LOS USUARIOS REGISTRADOS.

Esta vez vamos a combinar algo de los módulos pasados, vamos a hacer una consulta para que nos muestre los usuarios registrados actualmente y después vamos a seleccionar de entre ellos al que queremos eliminar.

### 4.1 eliminar.php

Nos permite eliminar algún usuario registrado, pero para ello se realiza una consulta por medio de la variable \$sSQL de la cual se genera el query seleccionando la tabla usuario ordenando por medio del id la variable, nos mostrara los registros en forma de menú desplegable con título usuario a eliminar, en este podremos seleccionar el usuario, también hay dos botones:

- Este es un botón de tipo submit que hará correr el código de **logica\_eliminar.php** para borrar al usuario.

- Este nos redireccionara a **consulta.php**

\$result: Es donde mandamos el query para realizar la consulta a la base de datos y almacenamos el resultado aqui.

```

<?php
session_start();
$carry = $_SESSION['noCuenta'];
if (!isset($carry)) {
    header("Status: 301 Moved Permanently");
    //header("Location: https://crud-fesa.000webhostapp.com/index.php");
    header("Location: index.php");
    exit;
} else {
    echo'<!DOCTYPE html>
    <html lang="en">

    <head>
        <link rel="stylesheet" type="text/css" href="/assets/estilo.php">
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
        <link rel="Shortcut icon" href="assets/coders2.png">
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Eliminar</title>

    </head>

    <body class="body1">;

include("./assets/header.php");
// $aVar = mysqli_connect('localhost', 'root', '', 'php_test');
echo '
<header class="shield">
    <svg class="bi bi-person-dash" width="5em" height="5em" viewBox="0 0 16 16" fill="white" xmlns="http://www.w3.org/2000/svg">
        <path fill-rule="evenodd" d="M11 14s1-1 4-6-4-6 3-6 4 1 1 1h10zm-9.995-.944v-.002.002zM1.022 13h9.956a.274.274 0
    </svg>
    <h2 class="h21" style="margin-left: 30px">ELIMINAR</h2>
</header>
<div class=" div1 " style=" margin-top:50px">
<div align="center">

';

//Conexion con la base -----
require "conexion.php";

echo "<FORM METHOD='POST' ACTION='./logica/logica_eliminar.php' class='form1 data'> Usuario a eliminar: <br>
    <div align='center'>";
//Creamos la sentencia SQL y la ejecutamos
$$SQL = "SELECT user FROM usuarios ORDER BY id";
$result = $conn->query($$SQL);

//Mostramos los registros en forma de menú desplegable
echo '<select name="nombre" class="text1 data2 ">';
while ($row = mysqli_fetch_array($result)) {
    echo '<option>' . $row["user"];
}
mysqli_free_result($result);
echo '<br>
<INPUT TYPE="SUBMIT" value="BORRAR" class="btn1" style="font-size:.6em">
<input type="button" style="font-size:.6em" onclick="location.href='./consulta.php'" value="REGRESAR" class="btn1">
</div>
</form>
</select>
</div>
</div>
';

```

## 4.2 logica\_eliminar.php

Mostrará un título ELMINADO, se crea la conexión con la base de datos y se guarda la query en \$sSQL para eliminar el usuario de la tabla y se manda a través de la conexión en \$result, luego se imprimirá en pantalla “Usuario eliminado exitosamente” y se mostrara un botón “VER CAMBIOS” que nos redirige a **consulta.php**.

```
<link rel="Shortcut icon" href="../../assets/coders2.png">
<link rel="stylesheet" type="text/css" href="../../assets/estilo.php">
<title>ELIMINADO</title>

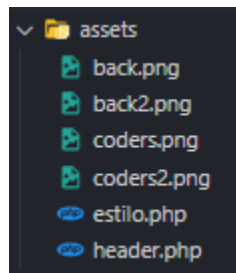
<?php
session_start();
$carry = $_SESSION['noCuenta'];

//Conexion con la base
require "../../conexion.php";
//Creamos la sentencia SQL y la ejecutamos
$sSQL = "DELETE FROM USUARIOS WHERE user='{$_POST['nombre']}'";
$result = $conn->query($sSQL);

echo"<body class='sansserif'>
<div align='center'>
<br><br><br><h1>Usuario eliminado exitosamente</h1><br>
<input type='button' style='font-size:.6em' onclick='location.href=`../../consulta.php`' value='VER CAMBIOS' class='btn1'>
</div>
</body>";
</html>";
```

## 5. CARPETA ASSETS.

Esta carpeta contiene los recursos gráficos utilizados en el proyecto.



### 5.1 header.php

**header.php** es un archivo usado en todas las páginas de nuestro sistema de información, es por esto por lo que tiene su propio archivo de código para usarlo de forma global, y solo mandarlo a llamar con “require” o “include” y es considerado un recurso gráfico por esta razón.

```

<?php
echo "
    <header >
        <a href='index.php' class='a2' title='INICIO'>
            <img src='assets/coders.png' alt='Logo' class='img1'>
        </a>
    </header>
    ";
?>
<style>
    .a2 {
        display: flex;
        justify-content: flex-start;
        line-height: 110%;
        margin-left: 1%;
        margin-top: 1%;
        width: 262px;
        height: 85px;
    }

    .img1 {
        width: 100%;
        height: 100%;
    }
</style>

```

## 5.2 estilo.php

**estilo.php** es donde se alojan la mayoría de los estilos tipo css, pero está en formato php para que haya compatibilidad con todo nuestro proyecto, ya que esta maquetado en este mismo lenguaje y podría presentar problemas si no es así, cabe recalcar que no están todos los estilos en este archivo, ya que algunos están de forma “local” en su propio archivo como es el caso de **header.php**.

Disclamer: Solo se agregaron unos estilos de ejemplos, ya que en total son cerca de 100 líneas de código css y es innecesario poner todo.

```

<?php header("Content-type: text/css");
echo"
.body1{
    background: url('back.png') repeat center fixed;
    background-size: 100% 100%;
    position:relative;
    height:auto;
}
.div1{
    margin-left:auto;
    margin-right: auto;
    margin-bottom: 20px;
    display: block;
    text-align:left;
    font-weight: lighter;
}

```

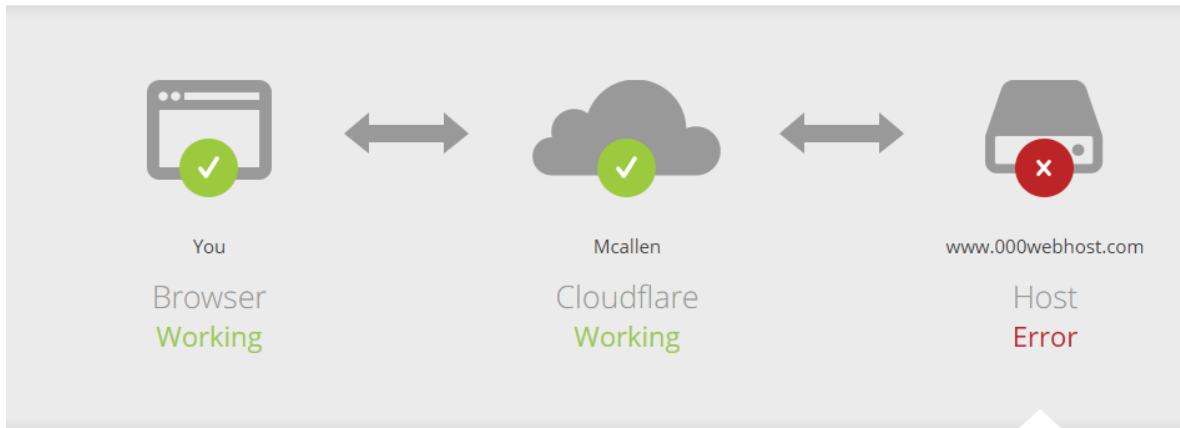
## 6. VIRTUALIZACIÓN DEL SISTEMA.

Por último, tenemos la virtualización del sistema, para que cualquiera pueda ver nuestro sistema de información desde cualquier lugar con conexión a internet sin la necesidad de tener nuestro proyecto completamente ni montarlo localmente, pero desafortunadamente tuvimos algunos errores fuera de nuestro control en esta parte, ya que el servidor estaba experimentando algunas fallas con las conexiones a las bases de datos, así que procedimos a montar nuestro sistema de manera local y es lo que mostramos a continuación.

### Error 502

Ray ID: 59cc3fb9ab6c9a62 • 2020-06-01 22:07:41 UTC

Bad gateway



#### What happened?

The web server reported a bad gateway error.

#### What can I do?

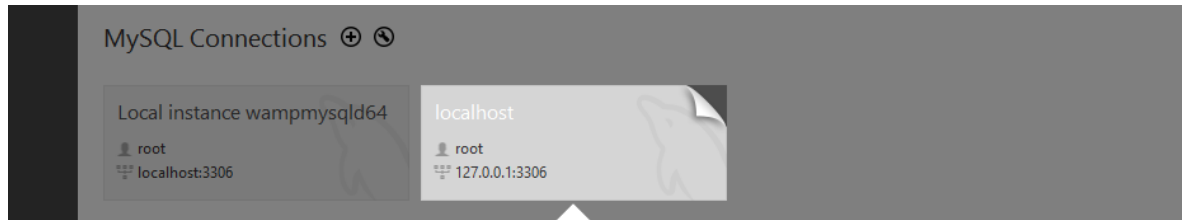
Please try again in a few minutes.

## MySQL

Se trata de una aplicación de bases de datos que controla Apache y que permite la creación de nuestras bases de datos que son necesarias para desarrollar el proyecto.

Lo único que debemos de hacer es crear un archivo sql en donde introduciremos lo esencial para crear la base de datos y la información y configuración inicial.

Además de que el archivo **conexión.php** ocupa las mismas credenciales que configuramos en MySQL.



## localhost

MySQL Version  
Last connected

10.4.11-MariaDB  
14 June 2020 22:38

User Account  
Password  
Network Address  
TCP/IP Port

root  
<not stored>  
127.0.0.1  
3306

Edit Connection...

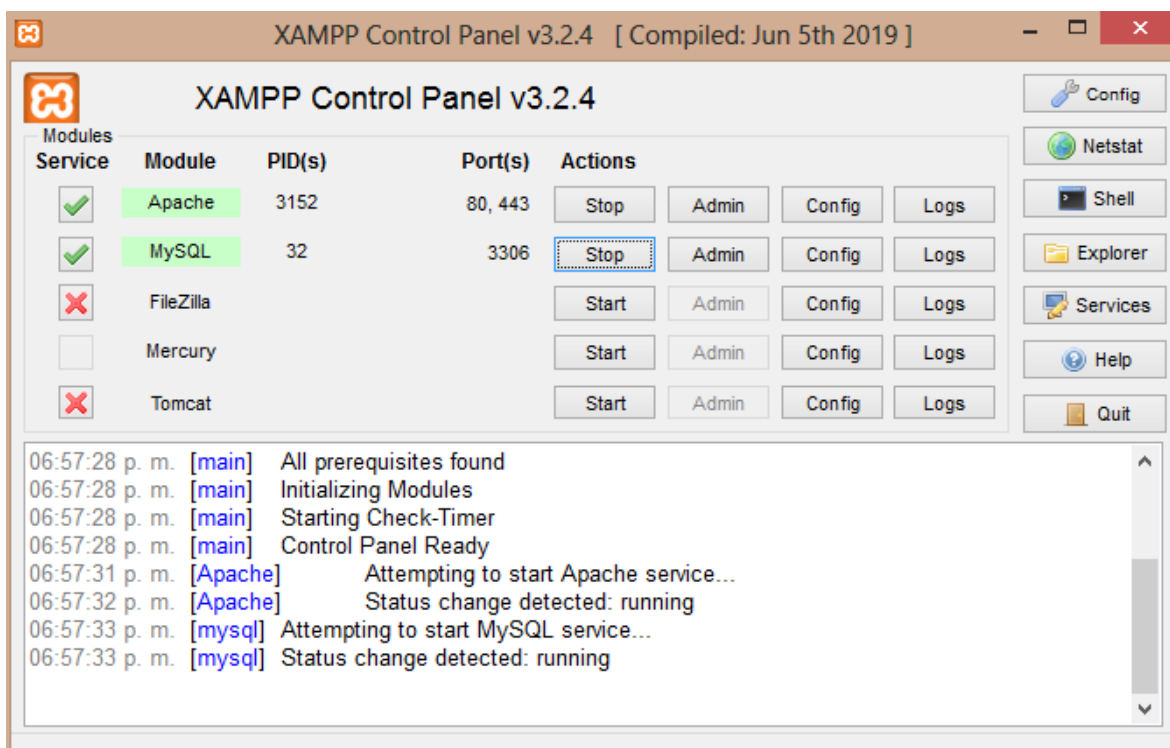
Configure Local Management...

```
DROP SCHEMA IF EXISTS `php_test`;  
CREATE SCHEMA IF NOT EXISTS `php_test` DEFAULT CHARACTER SET utf8 collate utf8_spanish2_ci;  
USE `php_test`;  
  
CREATE TABLE `usuarios` (  
  `id` int(99) NOT NULL,  
  `user` varchar(30) NOT NULL,  
  `nombre` varchar(300) NOT NULL,  
  `correo` varchar(300) NOT NULL,  
  `password` varchar(450) NOT NULL,  
  `date` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,  
  `perm` int(11) NOT NULL DEFAULT '3'  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
  
INSERT INTO `usuarios` (`id`,`user`,`nombre`,`correo`,`password`,`date`,`perm`) VALUES  
(1, 'root','Gabriel Ore Durán', 'allright@gmail.com', '0000', '0000-0-00 00:00:00',1),  
(2, 'admin','Alucard Van Hellsing', 'inhisworld@gmail.com', '1111', '1998-1-10 00:00:00',2);  
  
ALTER TABLE `usuarios`  
  ADD PRIMARY KEY (`id`);  
  
ALTER TABLE `usuarios`  
MODIFY `id` int(99) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=3;  
COMMIT;
```

## XAMPP

XAMPP es un paquete de software libre, que consiste principalmente en el sistema de gestión de bases de datos MySQL, el servidor web Apache que es la aplicación fundamental para convertir el equipo en servidor local.



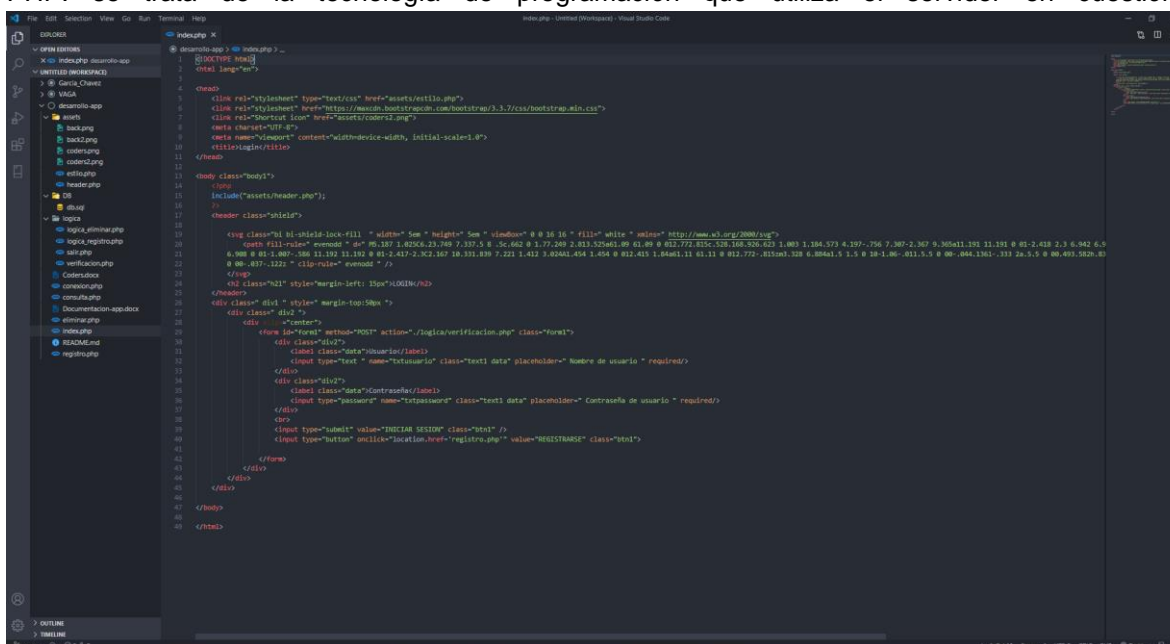


## VISUAL STUDIO CODE

Visual Studio Code es un editor de programación multiplataforma desarrollado por Microsoft. Es un proyecto de software libre que se distribuye bajo la licencia MIT, aunque los ejecutables se distribuyen bajo una licencia gratuita no libre.

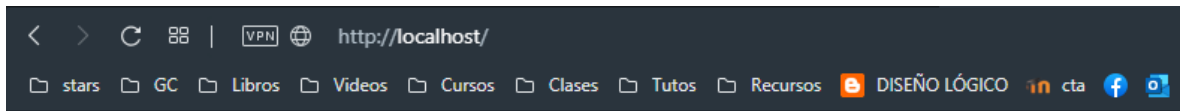
En este caso utilizamos VS Code para editar nuestro código en php para el desarrollo de nuestro sistema de información.

PHP: se trata de la tecnología de programación que utiliza el servidor en cuestión.



## 7. IMPLEMENTACION DE NUESTRO SISTEMA DE INFORMACIÓN.

Ya que tenemos todas estas herramientas listas solo accedemos a nuestro navegador favorito para acceder a nuestro host local y ver nuestro proyecto montado de forma local.

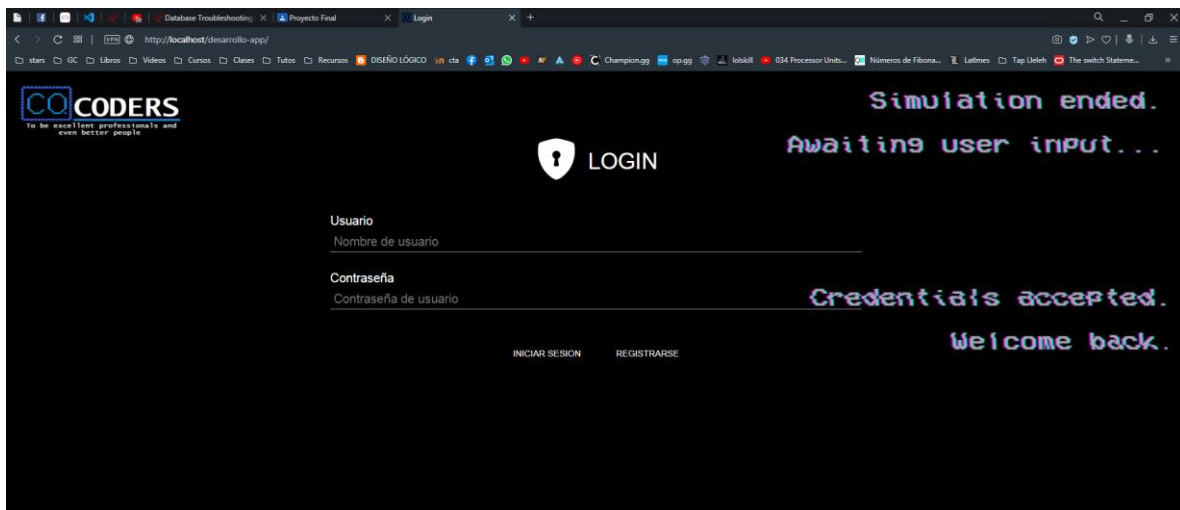


### Index of /

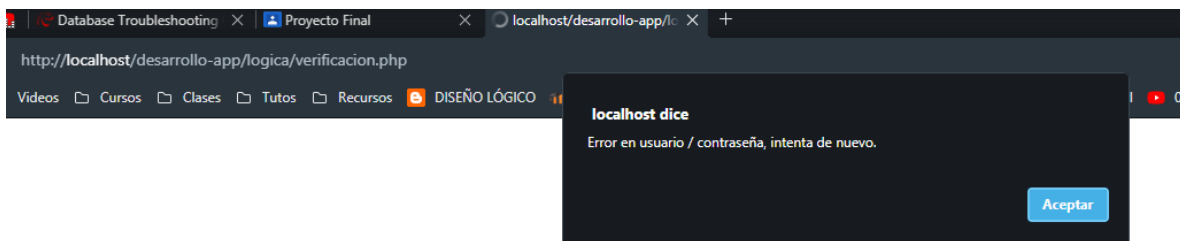
Name	Last modified	Size	Description
<a href="#">Garcia_Chavez/</a>	2020-05-21 17:21	-	
<a href="#">VAGA/</a>	2020-05-23 22:30	-	
<a href="#">desarrollo-app/</a>	2020-06-12 17:59	-	

Apache/2.4.43 (Win64) OpenSSL/1.1.1g PHP/7.2.30 Server at localhost Port 80

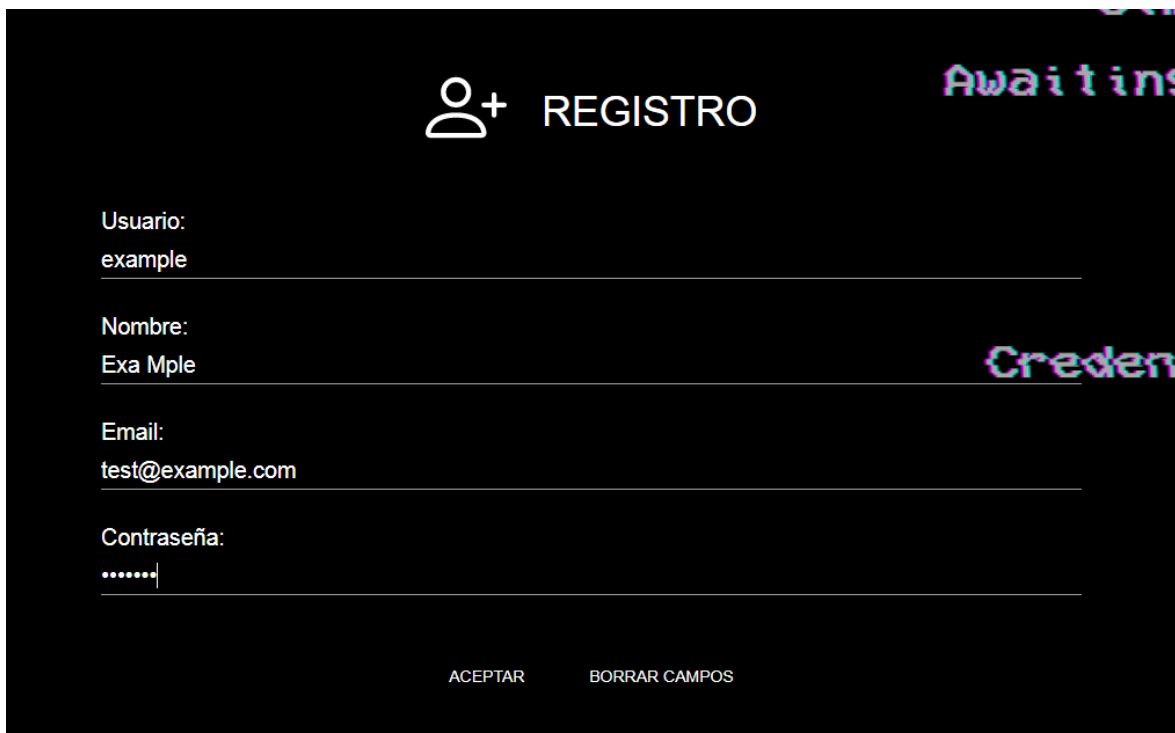
### Índex.php



Lo que pasa si cometemos un error, o un usuario no está registrado:



Desde **index.php** tenemos la opción de iniciar sesión, pero supongamos que no tenemos un usuario asignado aun y vamos a realizar el registro primero.



The image shows a registration form on a dark background. At the top, there is a user icon followed by the word "REGISTRO". Below this, there are four input fields with labels: "Usuario:" with the value "example", "Nombre:" with the value "Exa Mple", "Email:" with the value "test@example.com", and "Contraseña:" with masked characters ".....". At the bottom of the form, there are two buttons: "ACEPTAR" and "BORRAR CAMPOS".

REGISTRO

Usuario:  
example

Nombre:  
Exa Mple

Email:  
test@example.com

Contraseña:  
.....

ACEPTAR BORRAR CAMPOS

Aviso de registro exitoso y botón hacia el login.



En caso de que el usuario ya este registrado nos aparecerá el siguiente mensaje y un botón que nos regresara al registro:



Procedemos a iniciar sesión con el nuevo usuario.

A screenshot of a login form on a dark background. At the top, there is a white shield icon with a keyhole and the word "LOGIN" in white capital letters. Below this, there are two input fields. The first field is labeled "Usuario" and contains the text "example". The second field is labeled "Contraseña" and contains a series of dots. To the right of the form, there is some faint, colorful text that appears to be "Await" and "Crea".

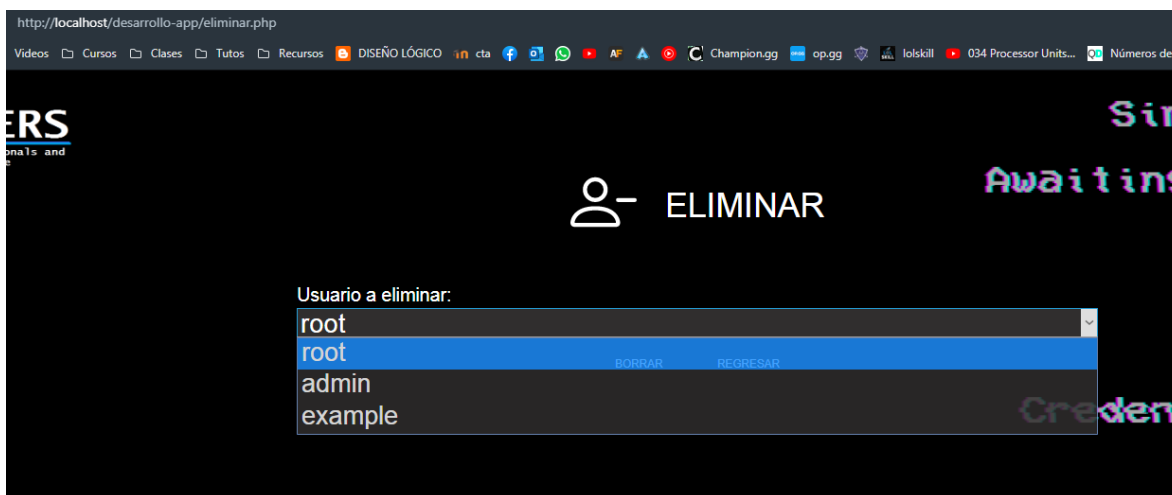
Pasaremos directamente a la página de consulta para ver los usuarios registrados y sus datos.

A screenshot of a web interface showing a table of registered users. The table has columns for ID, Usuario, Nombre completo, Correo, Contraseña, Fecha de registro, and Nivel de permisos. Below the table, there are buttons for "AGREGAR USUARIO", "ELIMINAR USUARIO", and "SALIR". The background features a dark theme with a large, light-colored circular graphic.

ID	Usuario	Nombre completo	Correo	Contraseña	Fecha de registro	Nivel de permisos
1	root	Gabriel Oro Durán	alright@gmail.com	0000	0000-00-00 00:00:00	1
2	admin	Alucard Van Helsing	maswold@gmail.com	123	1998-01-10 00:00:00	2
7	example	Exa Mple	test@example.com	hola123	2020-06-14 22:57:04	3

Como podemos observar nuestro sql está configurado de manera que todos los nuevos usuarios tengan un nivel de permisos de nivel 3, inferior al 1 y 2, para que si queremos avanzar más allá de lo solicitado en este proyecto, asignar roles a cada nivel de permisos, dejando de fuera eso, tenemos 3 botones, “agregar usuario” que nos llevara al registro para agregar más usuarios, “eliminar usuario” que es exactamente para eso, y “salir” que cerrara nuestra sesión y se nos impedirá acceder a **consulta.php** modificando la url y nos redireccionara al login.

## eliminar.php



Después de eliminar al usuario especificado nos aparecerá el siguiente mensaje junto con un botón para ver los cambios:

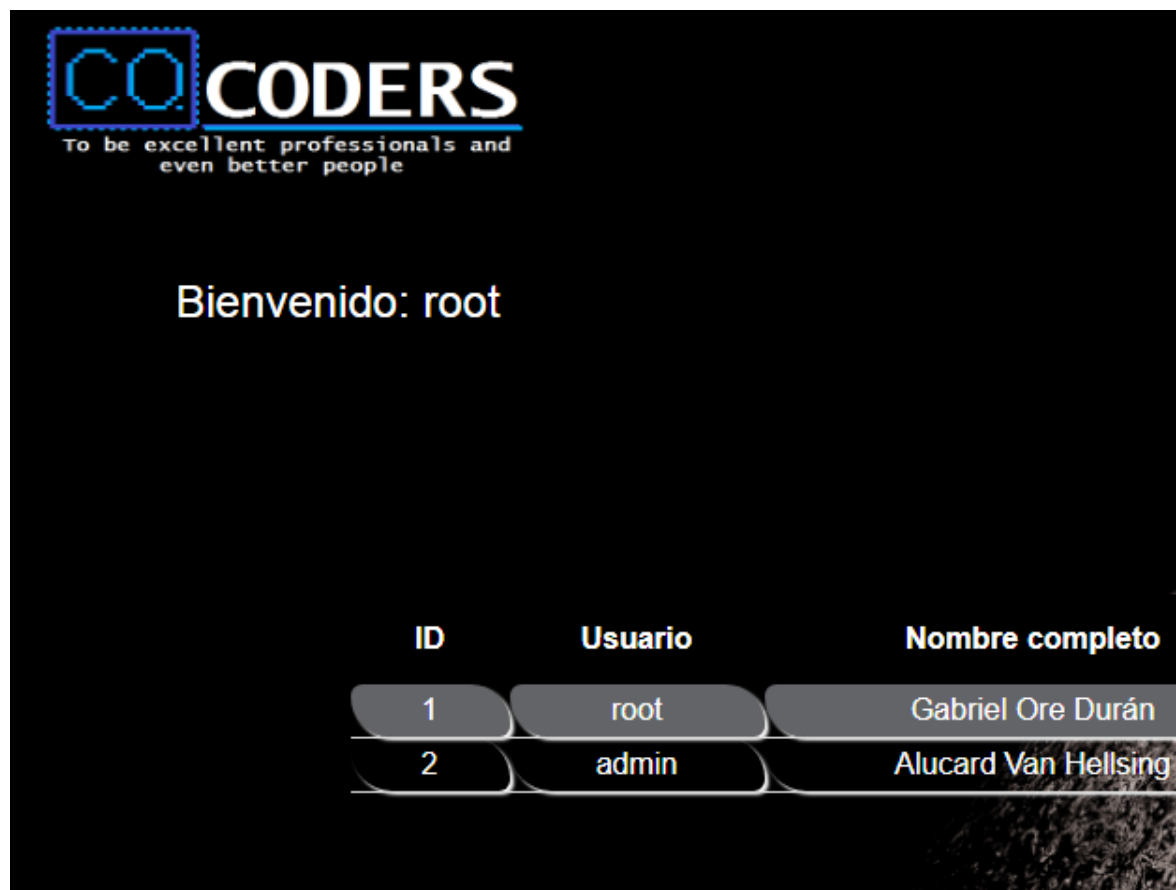
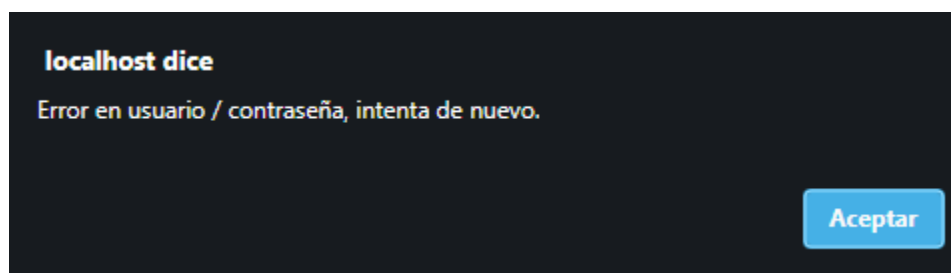


De momento aun reconoce al usuario ejemplo porque no hemos salido de la sesión.





Pero después de salir de la sesión o cerrar y reabrir la página, no podremos acceder con el mismo usuario.



## 8. CONCLUSION.

Fue un proyecto divertido dentro de lo que cabe, donde pudimos demostrar nuestros conocimientos adquiridos durante el semestre, este trabajo aún tiene mucho potencial, y nos sirve de base para probar muchas cosas nuevas y más complejas, siento que el trabajo quedo muy completo y como todo caben mejoras, pero para iniciar estuvo muy bien. Gracias al profesor por motivarnos y darnos las herramientas para hacer cosas como estas.

Gracias por su atención.