

Universidad de las fuerzas armadas ESPE

Integrantes: Emilio Ñacato, Sebastian Falconi

NRC: 10047

Fecha: 5/6/2023

Tema: Actividades en clase con php y mysql.

Tarea 001

Objetivo: Implementar un sistema de login utilizando PHP y sesiones para autenticar a los usuarios en una aplicación web.

Descripción: Desarrollar un sistema de login es fundamental para proteger y controlar el acceso a una aplicación web. En esta tarea, se te solicita implementar un sistema de login utilizando PHP y sesiones. Deberás crear un formulario de inicio de sesión que solicite al usuario su nombre de usuario y contraseña. Al recibir los datos, deberás verificar si corresponden a un usuario válido almacenado en la base de datos. Si la autenticación es exitosa, deberás iniciar una sesión y redirigir al usuario a una página de inicio. En caso de que la autenticación falle, se deberá mostrar un mensaje de error adecuado al usuario.

“index.php”

Este código HTML y JavaScript representa un bloque de diseño de una sección en una página web. Veamos qué hace cada parte del código:

Dentro de la etiqueta <section>, se encuentra un bloque de estilos en línea definido con la etiqueta <style>. Estos estilos definen las propiedades de borde para dar forma redondeada a ciertos elementos.

Dentro de la etiqueta <div class="card mb-3"> se encuentra una tarjeta que contiene una fila (<div class="row g-0 d-flex align-items-center">) con dos columnas (<div class="col-lg-4 d-none d-lg-flex"> y <div class="col-lg-8">). La primera columna contiene una imagen, mientras que la segunda columna contiene el formulario de inicio de sesión.

Dentro del formulario (<form method="POST" action="php/login.php" onsubmit="return validarFormulario()">), se encuentran dos campos de entrada de texto para el nombre de usuario y la contraseña. Los valores de estos campos se enviarán al archivo "login.php" cuando se envíe el formulario.

Después del formulario, hay un script en JavaScript que define una función llamada validarFormulario(). Esta función se ejecuta cuando se envía el formulario y realiza algunas validaciones en los campos de usuario y contraseña.

La función validarFormulario() obtiene los valores de los campos de usuario y contraseña y realiza varias comprobaciones. Verifica que los campos no estén vacíos y que la longitud del usuario y la contraseña esté dentro de ciertos límites.

Si alguna de las validaciones falla, se muestra una alerta y se detiene el envío del formulario. Si todas las validaciones son exitosas, el formulario se enviará al archivo "login.php".

En resumen, este código HTML y JavaScript representa un bloque de diseño que muestra una tarjeta con un formulario de inicio de sesión. El formulario se envía al archivo "login.php" y se realiza la validación de los campos de usuario y contraseña antes de enviar el formulario.

```
<!-- Section: Design Block -->
<section class="text-center text-lg-start">
  <style>
    .rounded-t-5 {
      border-top-left-radius: 0.5rem;
      border-top-right-radius: 0.5rem;
    }

    @media (min-width: 992px) {
      .rounded-tr-lg-0 {
        border-top-right-radius: 0;
      }

      .rounded-bl-lg-5 {
        border-bottom-left-radius: 0.5rem;
      }
    }
  </style>
  <div class="card mb-3">
    <div class="row g-0 d-flex align-items-center">
      <div class="col-lg-4 d-none d-lg-flex">
        
        </div>
        <div class="col-lg-8">
          <div class="card-body py-5 px-md-5">
            <form method="POST" action="php/login.php"
onsubmit="return validarFormulario()">
              <div class="form-group">
                <label for="username">Usuario:</label>
                <input type="text" class="form-control"
name="username" id="username"
                  aria-describedby="emailHelp"
placeholder="Ingresa el usuario">
              </div>
              <div class="form-group">
                <label for="password">Contraseña</label>
                <input type="password" class="form-control"
name="password" id="password"
                  placeholder="Contraseña">
              </div>
            </div>
          <!-- 2 column grid layout for inline styling -->
```

```

        <div class="row mb-4">
            <div class="col d-flex justify-content-
center">
                <!-- Checkbox -->
            </div>
        </div>
        <!-- Submit button -->
        <button type="submit" class="btn btn-
primary">Enviar</button>
    </form>
</div>
</div>
</div>
</div>
</section>
<!-- Section: Design Block -->

<script>
    function validarFormulario() {
        // Obtener los valores de los campos de usuario y contraseña
        var username = document.getElementById("username").value;
        var password = document.getElementById("password").value;

        // Verificar que los campos no estén vacíos
        if (username.trim() === '' || password.trim() === '') {
            alert("Todos los campos son obligatorios");
            return false; // Detener el envío del formulario
        }

        // Verificar la longitud del campo de usuario
        if (username.length < 2 || username.length > 15) {
            alert("El usuario debe tener entre 2 y 15 caracteres");
            return false; // Detener el envío del formulario
        }

        // Verificar la longitud del campo de contraseña
        if (password.length < 2 || password.length > 15) {
            alert("La contraseña debe tener entre 2 y 15 caracteres");
            return false; // Detener el envío del formulario
        }

        // Si todas las validaciones son exitosas, puedes enviar el
formulario
        return true;
    }
</script>

```

“inicio.php”

Este código PHP se encarga de manejar la autenticación y mostrar la página de inicio personalizada para un usuario que ha iniciado sesión en el sistema. Veamos qué hace cada parte del código:

En primer lugar, se llama a la función `session_start()`, que inicia una sesión de PHP y permite el uso de variables de sesión. Las variables de sesión son utilizadas para almacenar información del usuario durante su visita al sitio web.

A continuación, se verifica si la variable de sesión 'username' está definida. Esta variable almacena el nombre de usuario del usuario que ha iniciado sesión. Si la variable no está definida, significa que el usuario no ha iniciado sesión o no tiene una sesión activa. En ese caso, se redirige al usuario al formulario de inicio de sesión "login.php" utilizando la función `header()`. Esto redirige al usuario a la página de inicio de sesión para que pueda ingresar sus credenciales y autenticarse. Luego, la función `exit()` se utiliza para detener la ejecución del resto del código, evitando que se muestre el contenido de la página de inicio personalizada.

Si la variable de sesión 'username' está definida, se procede a obtener el valor almacenado en esa variable y asignarlo a la variable `$username`. Esto permitirá mostrar el nombre de usuario en la página de inicio personalizada para darle la bienvenida de manera personalizada.

A continuación, se muestra el contenido HTML de la página. El título de la página se establece como "Bienvenido", seguido del nombre de usuario obtenido de la variable de sesión. Esto se logra utilizando la etiqueta `<?php ?>` para incrustar el valor de la variable `$username` dentro del código HTML.

Luego, se muestra un encabezado de nivel 2 que da la bienvenida al usuario por su nombre de usuario utilizando la etiqueta `<?php ?>` para mostrar el valor de la variable `$username`. A continuación, se muestra un mensaje indicando que el usuario ha iniciado sesión correctamente.

Por último, se muestra un enlace para cerrar sesión. Al hacer clic en este enlace, se redirige al usuario al archivo "logout.php", que se encargará de cerrar la sesión y redirigir al usuario a la página de inicio de sesión.

En resumen, este código PHP verifica si el usuario ha iniciado sesión y muestra una página de inicio personalizada con su nombre de usuario. Si el usuario no ha iniciado sesión, se redirige automáticamente al formulario de inicio de sesión. La página de inicio personalizada muestra un mensaje de bienvenida, información sobre la sesión iniciada y un enlace para cerrar sesión.

```
<?php
session_start();

// Verificar si el usuario ha iniciado sesión
if (!isset($_SESSION['username'])) {
    header("Location: login.php"); // Redirigir al usuario al formulario
    de inicio de sesión si no ha iniciado sesión
    exit();
}

// Obtener el nombre de usuario de la sesión
$username = $_SESSION['username'];
?>
```

```

<!DOCTYPE html>
<html>
<head>
    <title>Bienvenido <?php echo $username; ?></title>
</head>
<body>
    <h2>Bienvenido, <?php echo $username; ?>!</h2>
    <p>Has iniciado sesión correctamente.</p>
    <a href="logout.php">Cerrar sesión</a>
</body>
</html>

```

“login.php”

En este código PHP, se realiza una verificación de autenticación de usuarios utilizando consultas preparadas para evitar la inyección SQL. La autenticación se realiza a través de un formulario de inicio de sesión.

Cuando el formulario es enviado, se obtienen los datos ingresados por el usuario: el nombre de usuario y la contraseña. Estos valores se guardan en las variables \$username y \$password respectivamente, utilizando el arreglo \$_POST.

Luego, se prepara una consulta SQL que busca en la tabla "users" registros que coincidan con el nombre de usuario y la contraseña proporcionados. Para evitar la inyección SQL, la consulta utiliza marcadores de posición (?) en lugar de los valores reales.

La consulta SQL preparada se ejecuta utilizando el método execute() del objeto \$stmt. Los valores del nombre de usuario y la contraseña se vinculan a los marcadores de posición utilizando bind_param(). Esto asegura que los valores se traten de forma segura y evita posibles ataques de inyección SQL.

Después de ejecutar la consulta, se obtiene el resultado utilizando el método get_result(). Si el número de filas devuelto por el resultado es igual a 1, significa que la autenticación fue exitosa. En caso de autenticación exitosa, se inicia una sesión y se guarda el nombre de usuario en la variable \$_SESSION['username']. Luego, se redirige al usuario a la página de inicio (inicio.php).

```

<?php
session_start();
include("../db.php");

// Obtener los datos del formulario
$username = $_POST['username'];
$password = $_POST['password'];

// Consultar la base de datos para verificar la autenticación
$query = "SELECT * FROM users WHERE username=? AND password=?";
$stmt = $conn->prepare($query);

```

```

$stmt->bind_param("ss", $username, $password);
$stmt->execute();
$result = $stmt->get_result();

if ($result->num_rows == 1) {
    // Autenticación exitosa, iniciar sesión
    $_SESSION['username'] = $username;
    header("Location: inicio.php"); // Redirigir a la página de inicio
} else {
    // Autenticación fallida, mostrar mensaje de error
    echo "Usuario o contraseña incorrectos.";
}

$stmt->close();
$conn->close();
?>

```

“logout.php”

Este código PHP se encarga de cerrar la sesión actual del usuario y redirigirlo al formulario de inicio de sesión. Veamos qué hace cada parte del código:

En primer lugar, se inicia la sesión de PHP utilizando `session_start()`. Esto permite acceder y manipular las variables de sesión. A continuación, se utiliza la función `session_destroy()` para destruir todos los datos registrados en la sesión actual. Esto incluye eliminar todas las variables de sesión y cerrar la sesión.

Luego, se utiliza la función `header()` para enviar una cabecera de redirección al navegador. El encabezado redirige al usuario al formulario de inicio de sesión, que se encuentra en el archivo “index.php” en el directorio superior (“../”) al actual.

Por último, se utiliza la función `exit()` para detener la ejecución del código y evitar que se muestre cualquier contenido adicional. Esto asegura que el usuario sea redirigido de inmediato al formulario de inicio de sesión y no se procese ninguna instrucción adicional. En resumen, este código PHP cierra la sesión actual del usuario, elimina todas las variables de sesión y redirige al usuario al formulario de inicio de sesión.

```

<?php
session_start();
session_destroy(); // Cerrar la sesión

header("Location: ../index.php"); // Redirigir al usuario al formulario
de inicio de sesión
exit();
?>

```

Código para crear la tabla de los usuarios en la base de datos.

```

CREATE TABLE users (

id INT PRIMARY KEY AUTO_INCREMENT,

username VARCHAR(50),

password VARCHAR(50)

);


INSERT INTO `users` (`id`, `username`, `password`) VALUES

(1, 'admin', 'admin123'),

(2, 'usuario1', 'pass123'),

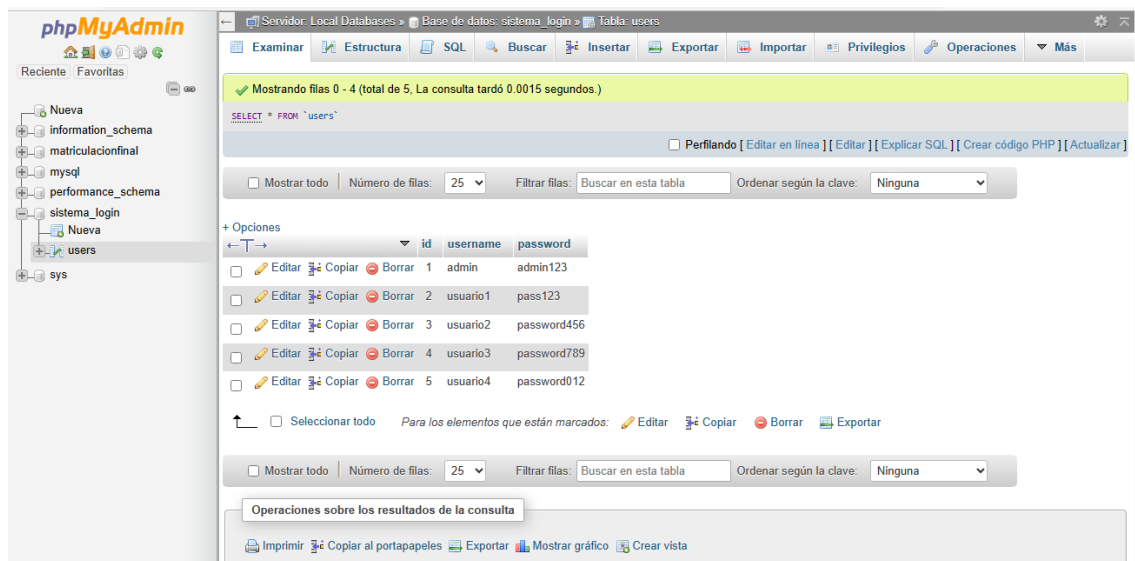
(3, 'usuario2', 'password456'),

(4, 'usuario3', 'password789'),

(5, 'usuario4', 'password012');

```

Base de datos en mysql llamada “sistema_login”



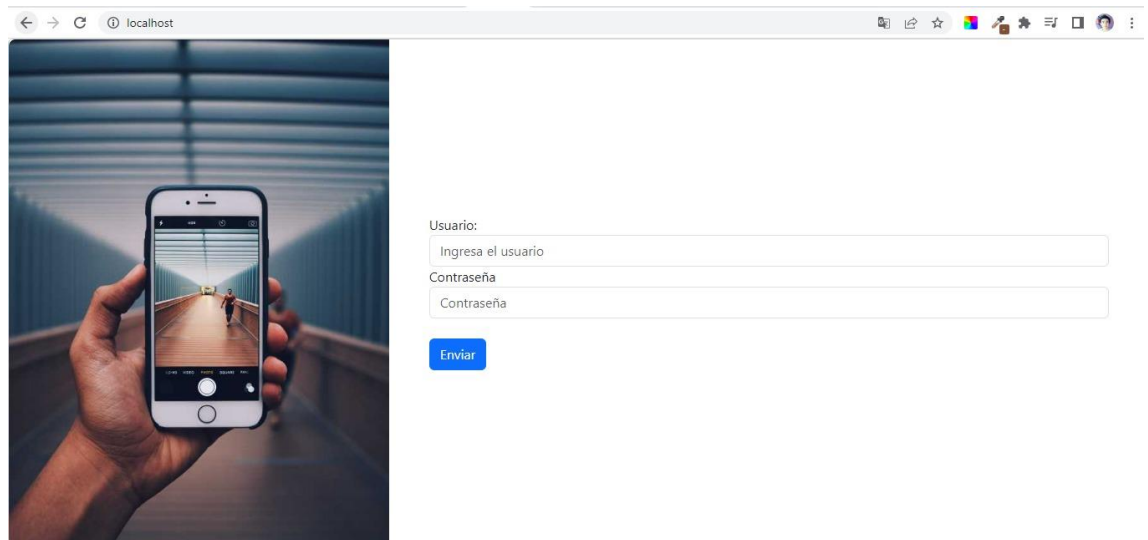
The screenshot shows the phpMyAdmin interface. On the left, the database structure is visible, with 'sistema_login' selected. The main area displays the 'users' table with the following data:

id	username	password
1	admin	admin123
2	usuario1	pass123
3	usuario2	password456
4	usuario3	password789
5	usuario4	password012

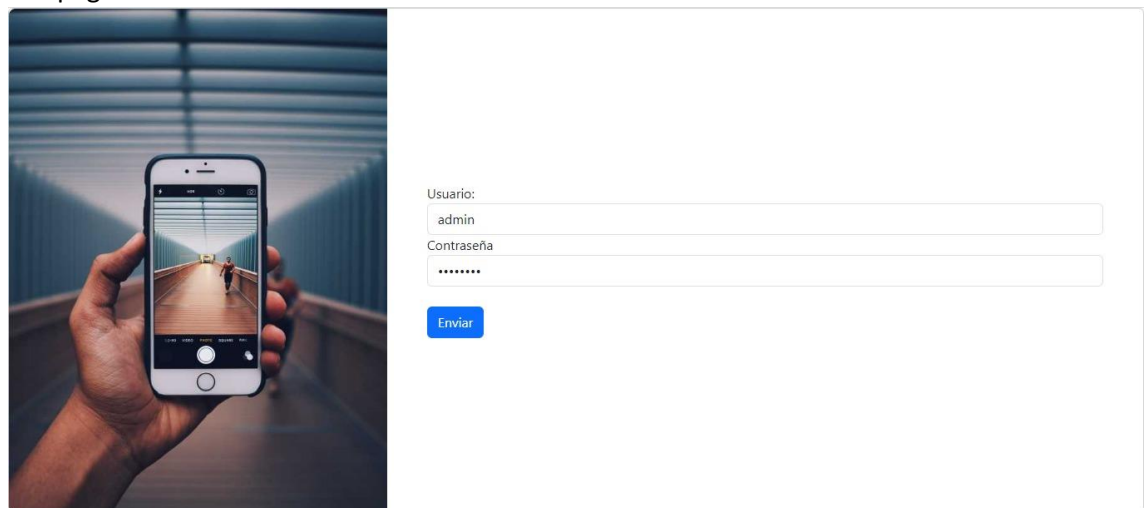
The interface also shows the SQL query used to retrieve the data: `SELECT * FROM `users``. At the bottom, there are options to perform operations on the query results, such as 'Imprimir', 'Copiar al portapapeles', 'Exportar', 'Mostrar gráfico', and 'Crear vista'.

Capturas de pantalla de su funcionamiento.

1. Ejecutamos nuestro “index.php” en el localhost.



2. Ingresamos los datos que se pide como el usuario y la contraseña, y así poder ingresar a la página de inicio.



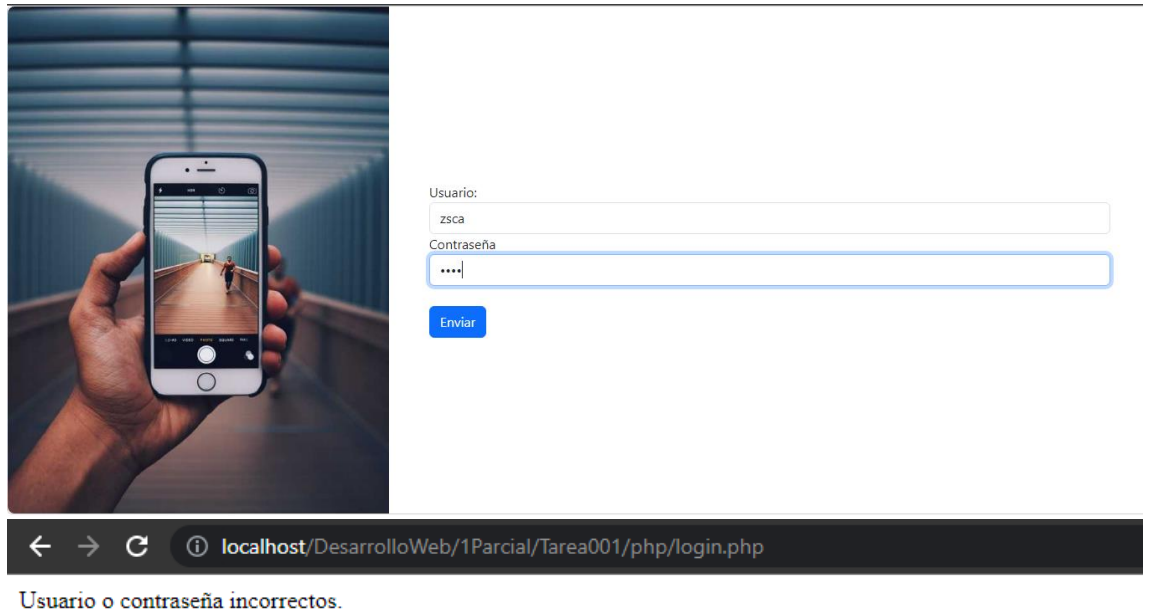
3. En caso de ser el usuario y la contraseña correcta nos dirigirá a una página de inicio, donde nos mostrará un mensaje que se inició de sesión correcto.

Bienvenido, admin!

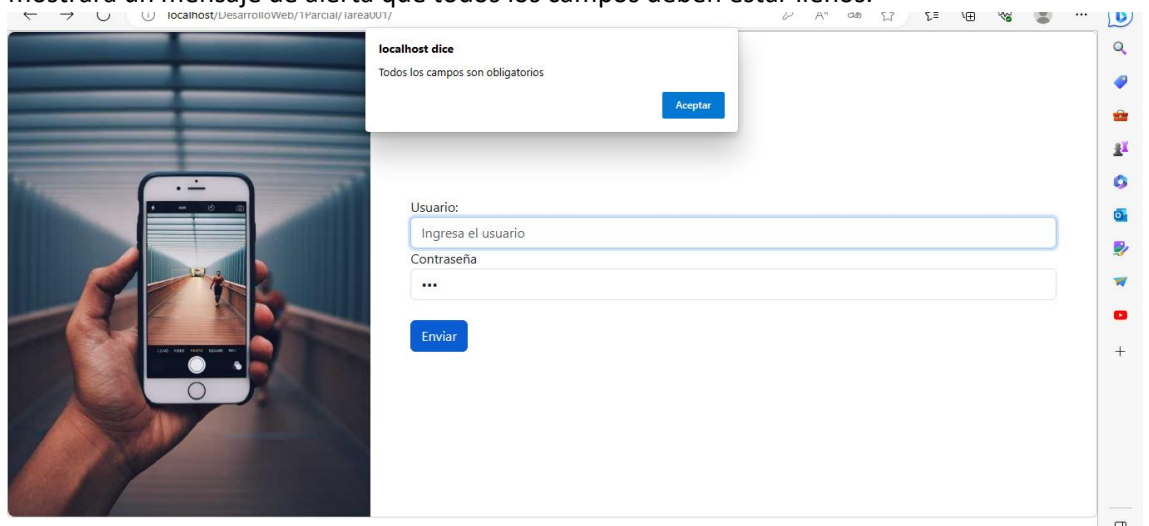
Has iniciado sesión correctamente.

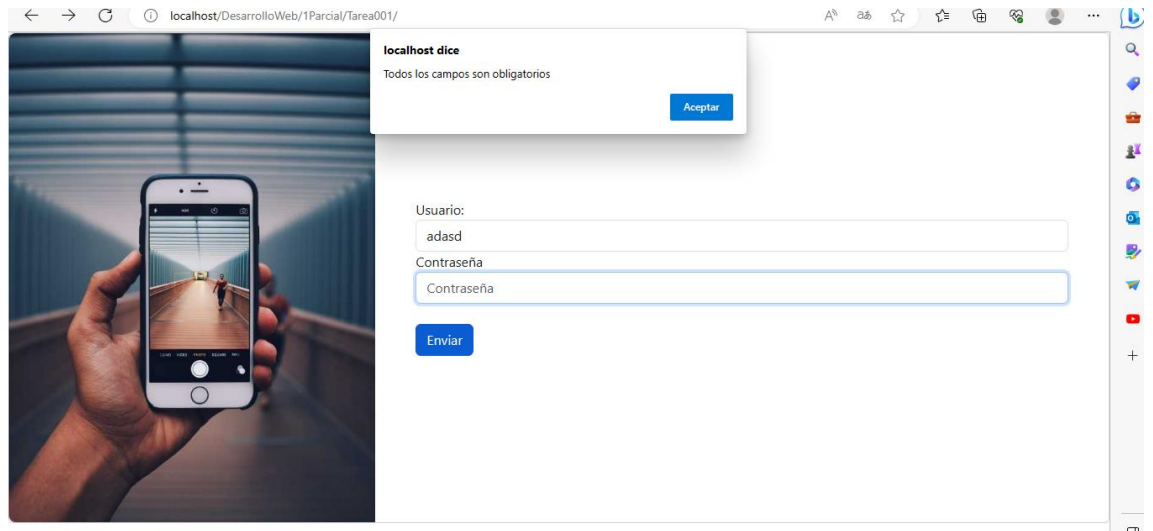
[Cerrar sesión](#)

4. En caso de ser incorrectos nos mostrara un mensaje de usuario o contraseña incorrectos.

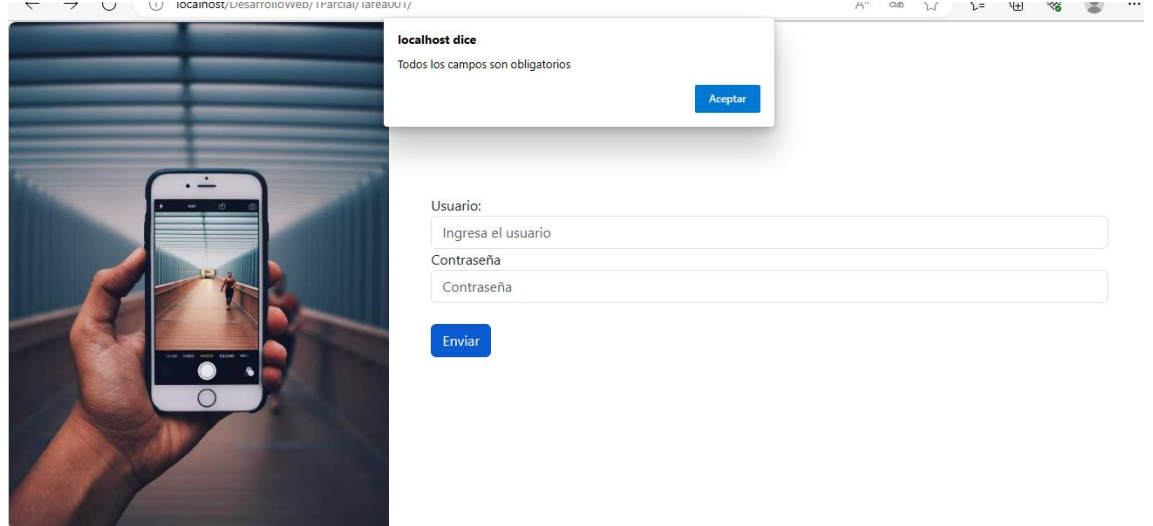


5. En el caso que un usuario o contraseña no se ingresen y envíen el formulario este se mostrara un mensaje de alerta que todos los campos deben estar llenos.

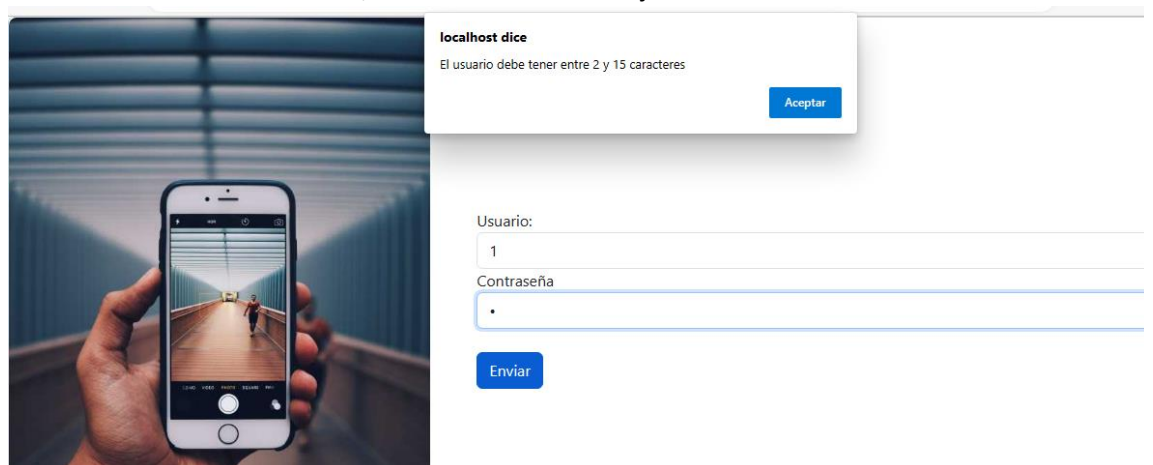




6. Mediante una validación al formulario podremos controlar que el usuario tenga que llenar los campos y no ingrese campos vacíos.



7. De la misma forma se añadió una longitud a los campos que ingresen más de un carácter y menos de 15 en el formulario, esto para los dos campos de usuario y contraseña. En caso contrario, se mostrará un mensaje de alerta.





localhost dice

La contraseña debe tener entre 2 y 15 caracteres

Aceptar

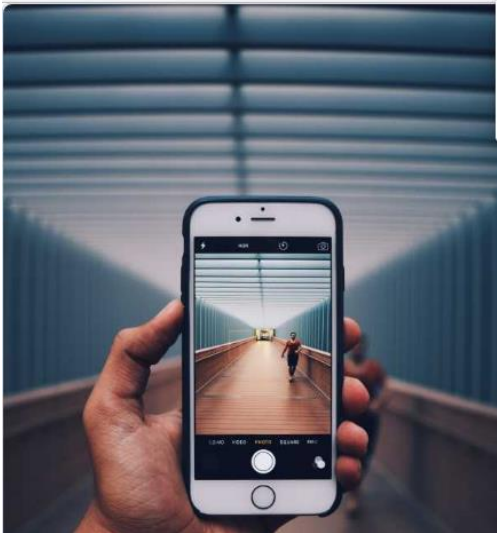
Usuario:

1123

Contraseña

•

Enviar



localhost dice

El usuario debe tener entre 2 y 15 caracteres

Aceptar

Usuario:

1123123123123123323131232131232131312

Contraseña

.....

Enviar