



HITO 2 DEL 2º TRIMESTRE DE PROGRAMACIÓN

Alberto Serna Rojas



12 DE FEBRERO DE 2025

CAMPUSFP
Getafe

INDICE

1. Página de Inicio:	2
- Crear una página de inicio que presente la opción de registro o inicio de sesión.	2
Index.php.....	2
2. Registro de Usuarios:	4
- Implementar un formulario de registro con los campos obligatorios: nombre de usuario, correo electrónico y contraseña.	4
- Validar que el correo electrónico sea único en la base de datos.	4
- Almacenar la información del usuario en la base de datos de MySQL de manera segura (hash de contraseña).	4
Registro.php	4
3. Inicio de Sesión:	9
- Crear un formulario de inicio de sesión que valide las credenciales ingresadas.	9
- Utilizar sesiones para mantener al usuario autenticado durante su sesión.	9
Login.php	9
4. CR de Tareas: sólo será visible para los usuarios validados	12
- Desarrollar una página donde los usuarios puedan agregar nuevas tareas.	12
- Mostrar el listado de tareas asociados al usuario identificado en el sistema	12
Tareas.php	12
5. Gestión de Sesiones:	21
- Crear un sistema que maneje el cierre de sesión de forma segura.	21
Cerrar_sesion.php	21
Conexión.php	21
Base_datos.sql.....	23
Styles.css.....	24
FUNCIONAMIENTO	25
ENLACE GITHUB:	28
BIBLIOGRAFIA.....	29

1. Página de Inicio:

- Crear una página de inicio que presente la opción de registro o inicio de sesión.

Index.php

CÓDIGO:

```
<?php
session_start(); // Inicia una sesión
?>

<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Gestión de Tareas</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <h1>Bienvenido</h1>

    <div class="container">
        <h2>Elige una opción:</h2>
        <a href="registro.php">Registrarse</a> <!-- Te lleva a la página de registro -->
        <a href="login.php">Iniciar Sesión</a> <!-- Te lleva a la página de inicio de sesión -->
    </div>
</body>
</html>
```

CAPTURA:

```
index.php > ...
1  <?php
2  session_start(); // Inicia una sesión
3  ?>
4
5  <!DOCTYPE html>
6  <html lang="es">
7  <head>
8      <meta charset="UTF-8">
9      <meta name="viewport" content="width=device-width, initial-scale=1.0">
10     <title>Gestión de Tareas</title>
11     <link rel="stylesheet" href="styles.css">
12 </head>
13 <body>
14     <h1>Bienvenido</h1>
15
16     <div class="container">
17         <h2>Elige una opción:</h2>
18         <a href="registro.php">Registrarse</a> <!-- Te lleva a la página de registro -->
19         <a href="login.php">Iniciar Sesión</a> <!-- Te lleva a la página de inicio de sesión -->
20     </div>
21 </body>
22 </html>
23
```

PARTES PRINCIPALES:

session_start();

Sirve para el uso de sesiones en PHP, esto sirve para mantener la información de un usuario entre páginas distintas

<div class="container">

<h2>Elige una opción:</h2>

Registrarse

Iniciar Sesión

</div>

Sirve para acceder a la página de registro y de inicio de sesión.

2. Registro de Usuarios:

- Implementar un formulario de registro con los campos obligatorios: nombre de usuario, correo electrónico y contraseña.
- Validar que el correo electrónico sea único en la base de datos.
- Almacenar la información del usuario en la base de datos de MySQL de manera segura (hash de contraseña).

Registro.php

CÓDIGO:

```
<?php
session_start();

include('conexion.php'); // Incluye el archivo de conexión a la base de datos

// Verifica si el formulario fue enviado con el método POST
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    // Recibe y limpia los datos del formulario
    $nombre_usuario = trim($_POST['nombre_usuario']);
    $email = trim($_POST['email']);
    $contraseña = $_POST['contraseña'];

    // Verifica que los campos no estén vacíos
    if (empty($nombre_usuario) || empty($email) || empty($contraseña)) {
        $error = 'Todos los campos son obligatorios.';
    } else {
        $contraseña_hash = password_hash($contraseña, PASSWORD_DEFAULT); // Encripta la
        contraseña y luego la almacena en la base de datos

        try {
            // Verifica si el correo ya está registrado en la base de datos
            $sql = "SELECT id FROM usuarios WHERE email = :email";
            $stmt = $pdo->prepare($sql);
            $stmt->bindParam(':email', $email);
```

```

$stmt->execute();

if ($stmt->fetch()) {
    $error = 'El correo electrónico ya está registrado.';
} else {
    // Agrega el nuevo usuario en la base de datos
    $sql = "INSERT INTO usuarios (nombre_usuario, email, contraseña)
        VALUES (:nombre_usuario, :email, :password_hash)";

    $stmt = $pdo->prepare($sql);
    $stmt->bindParam(':nombre_usuario', $nombre_usuario);
    $stmt->bindParam(':email', $email);
    $stmt->bindParam(':password_hash', $contraseña_hash);
    $stmt->execute();

    $success = 'Usuario registrado exitosamente.';
}
} catch (PDOException $e) {
    $error = "Error en la base de datos: " . $e->getMessage();
}
}
?>

```

```

<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Registro de Usuario</title>
    <link rel="stylesheet" href="styles.css">

```

```
</head>

<body>

  <h2>Registro de Usuario</h2>

  <!-- Muestra mensajes de error si existe alguno-->

  <?php if ($error): ?>

    <p style="color: red;"><?php echo $error; ?></p>

  <?php endif; ?>

  <!-- Muestra mensaje de éxito si el registro se hizo correctamente -->

  <?php if ($success): ?>

    <p style="color: green;"><?php echo $success; ?></p>

  <?php endif; ?>

  <!-- Crea el formulario de registro de usuarios -->

  <form action="registro.php" method="POST">

    <label for="nombre_usuario">Nombre de Usuario:</label>

    <input type="text" name="nombre_usuario" required>

    <label for="email">Correo Electrónico:</label>

    <input type="email" name="email" required>

    <label for="contraseña">Contraseña:</label>

    <input type="password" name="contraseña" required>

    <button type="submit">Registrarse</button>

  </form>

  <p>¿Ya tienes cuenta? <a href="login.php">Inicia sesión aquí</a></p>

</body>

</html>
```

CAPTURA:

```
1 <?php
2 session_start();
3
4 include('conexion.php'); // Incluye el archivo de conexión a la base de datos
5
6 // Verifica si el formulario fue enviado con el método POST
7 if ($_SERVER['REQUEST_METHOD'] == 'POST') {
8     // Recibe y limpia los datos del formulario
9     $nombre_usuario = trim($_POST['nombre_usuario']);
10    $email = trim($_POST['email']);
11    $contraseña = $_POST['contraseña'];
12
13    // Verifica que los campos no estén vacíos
14    if (empty($nombre_usuario) || empty($email) || empty($contraseña)) {
15        $error = 'Todos los campos son obligatorios.';
16    } else {
17        $contraseña_hash = password_hash($contraseña, PASSWORD_DEFAULT); // Encripta la contraseña y luego la almacena en la base de datos
18        try {
19            // Verifica si el correo ya está registrado en la base de datos
20            $sql = "SELECT id FROM usuarios WHERE email = :email";
21            $stmt = $pdo->prepare($sql);
22            $stmt->bindParam(':email', $email);
23            $stmt->execute();
24
25            if ($stmt->fetch()) {
26                $error = 'El correo electrónico ya está registrado.';
27            } else {
28                // Agrega el nuevo usuario en la base de datos
29                $sql = "INSERT INTO usuarios (nombre_usuario, email, contraseña)
30                    VALUES (:nombre_usuario, :email, :password_hash)";
31
32                $stmt = $pdo->prepare($sql);
33                $stmt->bindParam(':nombre_usuario', $nombre_usuario);
34                $stmt->bindParam(':email', $email);
35                $stmt->bindParam(':password_hash', $contraseña_hash);
36                $stmt->execute();
37
38                $success = 'Usuario registrado exitosamente.';
39            }
40        } catch (PDOException $e) {
41            $error = "Error en la base de datos: " . $e->getMessage();
42        }
43    }
44 }
45 ?>
```

```
47 <!DOCTYPE html>
48 <html lang="es">
49 <head>
50     <meta charset="UTF-8">
51     <meta name="viewport" content="width=device-width, initial-scale=1.0">
52     <title>Registro de Usuario</title>
53     <link rel="stylesheet" href="styles.css">
54 </head>
55 <body>
56     <h2>Registro de Usuario</h2>
57
58     <!-- Muestra mensajes de error si existe alguno -->
59     <?php if ($error): ?>
60         <p style="color: red;"><?php echo $error; ?></p>
61     <?php endif; ?>
62
63     <!-- Muestra un mensaje de éxito si el registro se hizo correctamente -->
64     <?php if ($success): ?>
65         <p style="color: green;"><?php echo $success; ?></p>
66     <?php endif; ?>
67
68     <!-- Crea el formulario de registro de usuarios -->
69     <form action="registro.php" method="POST">
70         <label for="nombre_usuario">Nombre de Usuario:</label>
71         <input type="text" name="nombre_usuario" required>
72
73         <label for="email">Correo Electrónico:</label>
74         <input type="email" name="email" required>
75
76         <label for="contraseña">Contraseña:</label>
77         <input type="password" name="contraseña" required>
78
79         <button type="submit">Registrarse</button>
80     </form>
81
82     <p>¿Ya tienes cuenta? <a href="login.php">Inicia sesión aquí</a></p>
83 </body>
84 </html>
```


PARTES PRINCIPALES:

Esta parte sirve para encriptar la contraseña y guardarla en la base de datos, lo que servirá para los futuros inicios de sesión ya que los datos de registro estarán guardados.

```
$contraseña_hash = password_hash($contraseña, PASSWORD_DEFAULT);
```

Esta parte verifica si los usuarios están registrados, parte muy importante, ya que un usuario existente no podrá registrarse.

```
$sql = "SELECT id FROM usuarios WHERE email = :email";
```

```
$stmt = $pdo->prepare($sql);
```

```
$stmt->bindParam(':email', $email);
```

```
$stmt->execute();
```

Esta parte es el formulario del registro en el que el usuario introducirá sus datos para registrarse.

```
<form action="registro.php" method="POST">
```

```
<label for="nombre_usuario">Nombre de Usuario:</label>
```

```
<input type="text" name="nombre_usuario" required>
```

```
<label for="email">Correo Electrónico:</label>
```

```
<input type="email" name="email" required>
```

```
<label for="contraseña">Contraseña:</label>
```

```
<input type="password" name="contraseña" required>
```

```
<button type="submit">Registrarse</button>
```

```
</form>
```

```
<p>¿Ya tienes cuenta? <a href="login.php">Inicia sesión aquí</a></p>
```

3. Inicio de Sesión:

- Crear un formulario de inicio de sesión que valide las credenciales ingresadas.
- Utilizar sesiones para mantener al usuario autenticado durante su sesión.

Login.php

CÓDIGO:

```
<?php

session_start();

include('conexion.php');

if ($_SERVER['REQUEST_METHOD'] == 'POST') {

    //Recibe y limpia los datos del formulario

    $email = trim($_POST['email']);

    $contraseña = $_POST['contraseña'];

    //Valida que los campos no estén vacíos

    if (empty($email) || empty($contraseña)) {

        $error = 'Todos los campos son obligatorios.';

    } else {

        // Verifica si el usuario existe en la base de datos

        $sql = "SELECT * FROM usuarios WHERE email = :email";

        $stmt = $pdo->prepare($sql);

        $stmt->execute(['email' => $email]);

        $user = $stmt->fetch();

        //Comprueba si la contraseña es correcta para ese usuario

        if ($user && password_verify($contraseña, $user['contraseña'])) {

            // Se inicia la sesión y se guarda la información del usuario

            $_SESSION['usuario_id'] = $user['id'];

            $_SESSION['nombre_usuario'] = $user['nombre_usuario'];
```

```

        // Redirige a la página de las tareas
        header("Location: tareas.php");
        exit();
    } else {
        $error = 'Correo electrónico o contraseña incorrectos.';
    }
}
}
?>

<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Iniciar Sesión</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>

    <h1>Iniciar Sesión</h1>

    <!-- Muestra un mensaje de error si las credenciales son incorrectas -->
    <?php if (!empty($error)): ?>
        <p style="color: red;"><?php echo $error; ?></p>
    <?php endif; ?>

    <!-- Formulario de inicio de sesión -->
    <form action="login.php" method="POST">
        <div>
            <label for="email">Correo Electrónico:</label>
            <input type="email" name="email" required>

```

```

</div>

<div>

    <label for="contraseña">Contraseña:</label>

    <input type="password" name="contraseña" required>

</div>

<div>

    <button type="submit">Iniciar Sesión</button>

</div>

</form>

<p>¿No tienes una cuenta? <a href="registro.php">Regístrate aquí</a></p>

</body>

</html>

```

CAPTURA:

```

login.php > html > body > h1
1  <?php
2  session_start();
3  include('conexion.php');
4
5  if ($_SERVER['REQUEST_METHOD'] == 'POST') {
6      //Recibe y limpia los datos del formulario
7      $email = trim($_POST['email']);
8      $contraseña = $_POST['contraseña'];
9
10     //Valida que los campos no estén vacíos
11     if (empty($email) || empty($contraseña)) {
12         $error = 'Todos los campos son obligatorios.';
13     } else {
14         // Verifica si el usuario existe en la base de datos
15         $sql = "SELECT * FROM usuarios WHERE email = :email";
16         $stmt = $pdo->prepare($sql);
17         $stmt->execute(['email' => $email]);
18         $user = $stmt->fetch();
19
20         //Comprueba si la contraseña es correcta para ese usuario
21         if ($user && password_verify($contraseña, $user['contraseña'])) {
22             // Se inicia la sesión y se guarda la información del usuario
23             $_SESSION['usuario_id'] = $user['id'];
24             $_SESSION['nombre_usuario'] = $user['nombre_usuario'];
25
26             // Redirige a la página de las tareas
27             header("Location: tareas.php");
28             exit();
29         } else {
30             $error = 'Correo electrónico o contraseña incorrectos.';
31         }
32     }
33 }
34 ?>

```

PARTES PRINCIPALES:

Esta parte verifica que el usuario exista en la base de datos, lo que es vital ya que, si el usuario no se encuentra en la base de datos como registrado, no se podrá iniciar sesión con ese usuario o email y contraseña.

```
$sql = "SELECT * FROM usuarios WHERE email = :email";
```

```
$stmt = $pdo->prepare($sql);
```

```
$stmt->execute(['email' => $email]);
```

```
$user = $stmt->fetch();
```

Esta parte, primero se comprueba que la contraseña sea la correcta para ese usuario y una vez comprobado se inicia la sesión y se guarda la información de dicho usuario.

```
if ($user && password_verify($contraseña, $user['contraseña'])) {
```

```
    // Se inicia la sesión y se guarda la información del usuario
```

```
    $_SESSION['usuario_id'] = $user['id'];
```

```
    $_SESSION['nombre_usuario'] = $user['nombre_usuario'];
```

4. CR de Tareas: sólo será visible para los usuarios validados

- Desarrollar una página donde los usuarios puedan agregar nuevas tareas.
- Mostrar el listado de tareas asociados al usuario identificado en el sistema

Tareas.php

CÓDIGO:

```
<?php
```

```
session_start();
```

```
include('conexion.php');
```

```
if (!isset($_SESSION['usuario_id'])) {
```

```
    header("Location: login.php"); // Si no está logueado, redirigir al login
```

```
    exit();
```

```
}
```

```
// Obtener el ID del usuario autenticado
```

```

$usuario_id = $_SESSION['usuario_id'];

//Agrega una nueva tarea
if ($_SERVER['REQUEST_METHOD'] == 'POST' && isset($_POST['nombre_tarea']) &&
isset($_POST['descripcion_tarea'])) {
    $nombre_tarea = trim($_POST['nombre_tarea']);
    $descripcion_tarea = trim($_POST['descripcion_tarea']);

    if (empty($nombre_tarea) || empty($descripcion_tarea)) {
        $error = 'Todos los campos son obligatorios.';
    } else {
        try {
            // Añade una nueva tarea en la base de datos
            $sql = "INSERT INTO tareas (usuario_id, nombre_tarea, descripcion_tarea)
                VALUES (:usuario_id, :nombre_tarea, :descripcion_tarea)";
            $stmt = $pdo->prepare($sql);
            $stmt->execute([
                'usuario_id' => $usuario_id,
                'nombre_tarea' => $nombre_tarea,
                'descripcion_tarea' => $descripcion_tarea
            ]);
            $success = 'Tarea agregada exitosamente.';
        } catch (PDOException $e) {
            $error = "Error en la base de datos: " . $e->getMessage();
        }
    }
}

// Marca una tarea como completada
if (isset($_GET['completar']) && is_numeric($_GET['completar'])) {
    $tarea_id = $_GET['completar'];

```

```

try {
    $sql = "UPDATE tareas SET completada = TRUE WHERE id = :tarea_id AND usuario_id = :usuario_id";

    $stmt = $pdo->prepare($sql);

    $stmt->execute([
        'tarea_id' => $tarea_id,
        'usuario_id' => $usuario_id
    ]);
} catch (PDOException $e) {
    $error = "Error en la base de datos: " . $e->getMessage();
}
}

```

//Elimina una tarea

```

if (isset($_GET['eliminar']) && is_numeric($_GET['eliminar'])) {
    $tarea_id = $_GET['eliminar'];

    try {
        $sql = "DELETE FROM tareas WHERE id = :tarea_id AND usuario_id = :usuario_id";

        $stmt = $pdo->prepare($sql);

        $stmt->execute([
            'tarea_id' => $tarea_id,
            'usuario_id' => $usuario_id
        ]);
    } catch (PDOException $e) {
        $error = "Error en la base de datos: " . $e->getMessage();
    }
}

```

//Obtener todas las tareas del usuario

```

$sql = "SELECT * FROM tareas WHERE usuario_id = :usuario_id ORDER BY id DESC";
$stmt = $pdo->prepare($sql);

```

```
$stmt->execute(['usuario_id' => $usuario_id]);  
$areas = $stmt->fetchAll();  
?>
```

```
<!DOCTYPE html>  
<html lang="es">  
<head>  
  <meta charset="UTF-8">  
  <meta name="viewport" content="width=device-width, initial-scale=1.0">  
  <title>Mis Tareas</title>  
  <link rel="stylesheet" href="styles.css">  
</head>  
<body>  
  <h2>Mis Tareas</h2>  
  <?php if (!empty($error)): ?>  
    <p style="color: red;"><?php echo $error; ?></p>  
  <?php endif; ?>  
  
  <?php if (!empty($success)): ?>  
    <p style="color: green;"><?php echo $success; ?></p>  
  <?php endif; ?>  
  
  <!--Formulario para agregar nuevas tareas -->  
  <h3>Agregar nueva tarea</h3>  
  <form action="tareas.php" method="POST">  
    <label for="nombre_tarea">Nombre de la tarea:</label>  
    <input type="text" name="nombre_tarea" required>  
  
    <label for="descripcion_tarea">Descripción de la tarea:</label>  
    <textarea name="descripcion_tarea" required></textarea>
```



```

        <button type="submit">Agregar tarea</button>
    </form>

    <!--Lista de tareas pendientes -->

    <h3>Tareas pendientes</h3>
    <ul>
        <?php foreach ($tareas as $tarea): ?>
            <li>
                <?php echo htmlspecialchars($tarea['nombre_tarea']); ?>
                - <?php echo htmlspecialchars($tarea['descripcion_tarea']); ?>
                <?php if (!$tarea['completada']): ?>
                    <a href="tareass.php?completar=<?php echo $tarea['id']; ?>">Marcar como
completada</a>
                <?php else: ?>
                    <span>(Completada)</span>
                <?php endif; ?>
                <a href="tareass.php?eliminar=<?php echo $tarea['id']; ?>">Eliminar</a>
            </li>
        <?php endforeach; ?>
    </ul>

    <!--Cierra sesión -->
    <p><a href="cerrar_sesion.php">Cerrar sesión</a></p>
</body>
</html>

```

CÁPTURA:

```
* tareas.php > ...
1  <?php
2  session_start();
3  include('conexion.php');
4
5  if (!isset($_SESSION['usuario_id'])) {
6      header("Location: login.php"); // Si no está logueado, redirigir al login
7      exit();
8  }
9  // Obtener el ID del usuario autenticado
10 $usuario_id = $_SESSION['usuario_id'];
11
12 //Agrega una nueva tarea
13 if ($_SERVER['REQUEST_METHOD'] == 'POST' && isset($_POST['nombre_tarea']) && isset($_POST['descripcion_tarea'])) {
14     $nombre_tarea = trim($_POST['nombre_tarea']);
15     $descripcion_tarea = trim($_POST['descripcion_tarea']);
16
17     if (empty($nombre_tarea) || empty($descripcion_tarea)) {
18         $error = 'Todos los campos son obligatorios.';
19     } else {
20         try {
21             // Añade una nueva tarea en la base de datos
22             $sql = "INSERT INTO tareas (usuario_id, nombre_tarea, descripcion_tarea)
23                 VALUES (:usuario_id, :nombre_tarea, :descripcion_tarea)";
24             $stmt = $pdo->prepare($sql);
25             $stmt->execute([
26                 'usuario_id' => $usuario_id,
27                 'nombre_tarea' => $nombre_tarea,
28                 'descripcion_tarea' => $descripcion_tarea
29             ]);
30             $success = 'Tarea agregada exitosamente.';
31         } catch (PDOException $e) {
32             $error = "Error en la base de datos: " . $e->getMessage();
33         }
34     }
35 }
36
```

```
37 // Marca una tarea como completada
38 if (isset($_GET['completar']) && is_numeric($_GET['completar'])) {
39     $tarea_id = $_GET['completar'];
40     try {
41         $sql = "UPDATE tareas SET completada = TRUE WHERE id = :tarea_id AND usuario_id = :usuario_id";
42         $stmt = $pdo->prepare($sql);
43         $stmt->execute([
44             'tarea_id' => $tarea_id,
45             'usuario_id' => $usuario_id
46         ]);
47     } catch (PDOException $e) {
48         $error = "Error en la base de datos: " . $e->getMessage();
49     }
50 }
51
52 //Elimina una tarea
53 if (isset($_GET['eliminar']) && is_numeric($_GET['eliminar'])) {
54     $tarea_id = $_GET['eliminar'];
55     try {
56         $sql = "DELETE FROM tareas WHERE id = :tarea_id AND usuario_id = :usuario_id";
57         $stmt = $pdo->prepare($sql);
58         $stmt->execute([
59             'tarea_id' => $tarea_id,
60             'usuario_id' => $usuario_id
61         ]);
62     } catch (PDOException $e) {
63         $error = "Error en la base de datos: " . $e->getMessage();
64     }
65 }
66
67 //Obtener todas las tareas del usuario
68 $sql = "SELECT * FROM tareas WHERE usuario_id = :usuario_id ORDER BY id DESC";
69 $stmt = $pdo->prepare($sql);
70 $stmt->execute(['usuario_id' => $usuario_id]);
71 $tareas = $stmt->fetchAll();
72 ?>
```

```

74 <!DOCTYPE html>
75 <html lang="es">
76 <head>
77     <meta charset="UTF-8">
78     <meta name="viewport" content="width=device-width, initial-scale=1.0">
79     <title>Mis Tareas</title>
80     <link rel="stylesheet" href="styles.css">
81 </head>
82 <body>
83     <h2>Mis Tareas</h2>
84     <?php if (!empty($error)): ?>
85         <p style="color: red;"><?php echo $error; ?></p>
86     <?php endif; ?>
87
88     <?php if (!empty($success)): ?>
89         <p style="color: green;"><?php echo $success; ?></p>
90     <?php endif; ?>
91
92     <!--Formulario para agregar nuevas tareas -->
93     <h3>Agregar nueva tarea</h3>
94     <form action="tareas.php" method="POST">
95         <label for="nombre_tarea">Nombre de la tarea:</label>
96         <input type="text" name="nombre_tarea" required>
97
98         <label for="descripcion_tarea">Descripción de la tarea:</label>
99         <textarea name="descripcion_tarea" required></textarea>
100
101         <button type="submit">Agregar tarea</button>
102     </form>
103

```

```

104     <!--Lista de tareas pendientes -->
105     <h3>Tareas pendientes</h3>
106     <ul>
107         <?php foreach ($tareas as $tarea): ?>
108             <li>
109                 <?php echo htmlspecialchars($tarea['nombre_tarea']); ?>
110                 - <?php echo htmlspecialchars($tarea['descripcion_tarea']); ?>
111                 <?php if (!$tarea['completada']): ?>
112                     <a href="tareas.php?completar=<?php echo $tarea['id']; ?>">Marcar como completada</a>
113                 <?php else: ?>
114                     <span>(Completada)</span>
115                 <?php endif; ?>
116                 <a href="tareas.php?eliminar=<?php echo $tarea['id']; ?>">Eliminar</a>
117             </li>
118         <?php endforeach; ?>
119     </ul>
120
121     <!--Cierra sesión -->
122     <p><a href="cerrar_sesion.php">Cerrar sesión</a></p>
123 </body>
124 </html>
125

```

PARTES PRINCIPALES:

Esta parte sirve para agregar una nueva tarea con su nombre, y su descripción.

```

if ($_SERVER['REQUEST_METHOD'] == 'POST' && isset($_POST['nombre_tarea']) &&
isset($_POST['descripcion_tarea'])) {

```

```

    $nombre_tarea = trim($_POST['nombre_tarea']);

```

```

    $descripcion_tarea = trim($_POST['descripcion_tarea']);

```

Esta parte, añade la tarea a la base de datos, asociando esa tarea al usuario que esta iniciado en sesión, y la guardara.

```
$sql = "INSERT INTO tareas (usuario_id, nombre_tarea, descripcion_tarea)
VALUES (:usuario_id, :nombre_tarea, :descripcion_tarea)";

$stmt = $pdo->prepare($sql);

$stmt->execute([
    'usuario_id' => $usuario_id,
    'nombre_tarea' => $nombre_tarea,
    'descripcion_tarea' => $descripcion_tarea
]);

$success = 'Tarea agregada exitosamente.';
} catch (PDOException $e) {
    $error = "Error en la base de datos: " . $e->getMessage();
}
```

Esta parte sirve para marcar como completada una tarea, por lo que saldrá al lado de la tarea un texto de completada.

```
if (isset($_GET['completar']) && is_numeric($_GET['completar'])) {
    $tarea_id = $_GET['completar'];

    try {
        $sql = "UPDATE tareas SET completada = TRUE WHERE id = :tarea_id AND usuario_id = :usuario_id";

        $stmt = $pdo->prepare($sql);

        $stmt->execute([
            'tarea_id' => $tarea_id,
            'usuario_id' => $usuario_id
        ]);
    } catch (PDOException $e) {
        $error = "Error en la base de datos: " . $e->getMessage();
    }
}
```

Esta parte parte sirve para eliminar una tarea ya existente.

```
if (isset($_GET['eliminar']) && is_numeric($_GET['eliminar'])) {
    $tarea_id = $_GET['eliminar'];

    try {
        $sql = "DELETE FROM tareas WHERE id = :tarea_id AND usuario_id = :usuario_id";
    } catch (PDOException $e) {
        $error = "Error en la base de datos: " . $e->getMessage();
    }
}
```

```

$stmt = $pdo->prepare($sql);

$stmt->execute([
    'tarea_id' => $tarea_id,
    'usuario_id' => $usuario_id
]);

```

Esta parte obtiene una lista con todas las tareas que tenga un usuario, estén completadas o sin completar.

```

$sql = "SELECT * FROM tareas WHERE usuario_id = :usuario_id ORDER BY id DESC";
$stmt = $pdo->prepare($sql);
$stmt->execute(['usuario_id' => $usuario_id]);
$tareas = $stmt->fetchAll();

```

Esta parte muestra la lista de las tareas pendientes.

```
<h3>Tareas pendientes</h3>
```

```
<ul>
```

```
<?php foreach ($tareas as $tarea): ?>
```

```
<li>
```

```
<?php echo htmlspecialchars($tarea['nombre_tarea']); ?>
```

```
- <?php echo htmlspecialchars($tarea['descripcion_tarea']); ?>
```

```
<?php if (!$tarea['completada']): ?>
```

```
<a href="tareas.php?completar=<?php echo $tarea['id']; ?>">Marcar como
completada</a>
```

```
<?php else: ?>
```

```
<span>(Completada)</span>
```

```
<?php endif; ?>
```

```
<a href="tareas.php?eliminar=<?php echo $tarea['id']; ?>">Eliminar</a>
```

```
</li>
```

```
<?php endforeach; ?>
```

```
</ul>
```

5. Gestión de Sesiones:

- Crear un sistema que maneje el cierre de sesión de forma segura.

Cerrar_session.php

CÓDIGO:

```
<?php

session_start(); //Iniciar la sesión para poder utilizarla (se incluye en todos los archivos)

session_unset(); // Elimina todas las variables de dicha sesión

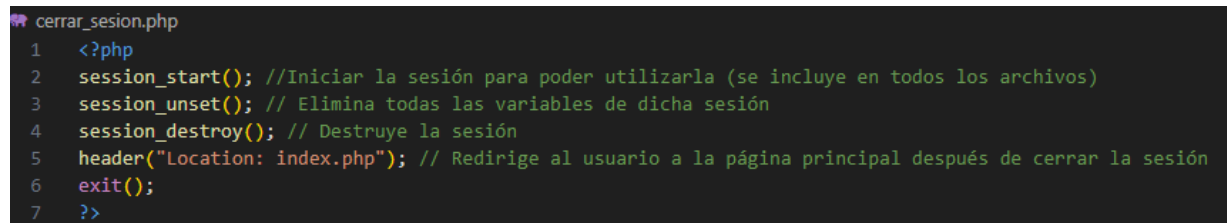
session_destroy(); // Destruye la sesión

header("Location: index.php"); // Redirige al usuario a la página principal después de cerrar la sesión

exit();

?>
```

CAPTURA:

A screenshot of a code editor showing the PHP code for closing a session. The code is as follows:

```
cerrar_session.php
1  <?php
2  session_start(); //Iniciar la sesión para poder utilizarla (se incluye en todos los archivos)
3  session_unset(); // Elimina todas las variables de dicha sesión
4  session_destroy(); // Destruye la sesión
5  header("Location: index.php"); // Redirige al usuario a la página principal después de cerrar la sesión
6  exit();
7  ?>
```

PARTES PRINCIPALES:

Todo el código es importante para cerrar la sesión, básicamente lo que aparece en los comentarios.

Conexión.php

CÓDIGO:

```
<?php

$host = 'localhost'; // El nombre del servidor de la base de datos

$dbname = 'hito_progra2'; // Nombre de la base de datos que hemos creado

$username = 'root'; // Nombre del usuario de la base de datos

$password = 'curso'; // Contraseña del usuario en el caso que haya

$charset = 'utf8mb4'; // Es una codificación de caracteres para soportar caracteres especiales
```

```

try {

    //Crear el Data Source Name (DSN) para la conexión a MySQL
    $dsn = "mysql:host=$host;dbname=$dbname;charset=$charset";

    //Crea una instancia de PDO para manejar la conexión a la base de datos
    $pdo = new PDO($dsn, $username, $password, [
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
        PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
        PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8mb4"
    ]);

} catch (PDOException $e) {

    die('Error de conexión: ' . $e->getMessage());

}

?>

```

CAPTURA:

```

conexion.php > ...
1  <?php
2  $host = 'localhost'; // El nombre del servidor de la base de datos
3  $dbname = 'hito_progra2'; // Nombre de la base de datos que hemos creado
4  $username = 'root'; // Nombre del usuario de la base de datos
5  $password = 'curso'; // Contraseña del usuario en el caso que haya
6  $charset = 'utf8mb4'; // Es una codificación de caracteres para soportar caracteres especiales
7
8  try {
9      //Crear el Data Source Name (DSN) para la conexión a MySQL
10     $dsn = "mysql:host=$host;dbname=$dbname;charset=$charset";
11
12     //Crea una instancia de PDO para manejar la conexión a la base de datos
13     $pdo = new PDO($dsn, $username, $password, [
14         PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
15         PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
16         PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8mb4"
17     ]);
18 } catch (PDOException $e) {
19     die('Error de conexión: ' . $e->getMessage());
20 }
21 ?>
22

```

Base_datos.sql

CÓDIGO:

```
CREATE DATABASE hito_progra2;
```

```
USE hito_progra2;
```

```
CREATE TABLE usuarios (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    nombre_usuario VARCHAR(50) NOT NULL,
```

```
    email VARCHAR(100) NOT NULL UNIQUE,
```

```
    contraseña VARCHAR(255) NOT NULL
```

```
);
```

```
CREATE TABLE tareas (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    usuario_id INT NOT NULL,
```

```
    nombre_tarea VARCHAR(255) NOT NULL,
```

```
    descripcion_tarea TEXT,
```

```
    completada BOOLEAN DEFAULT FALSE,
```

```
    FOREIGN KEY (usuario_id) REFERENCES usuarios(id) ON DELETE CASCADE
```

```
);
```

CAPTURA:

```
base_datos.sql
1  CREATE DATABASE hito_progra2;
2  USE hito_progra2;
3
4  CREATE TABLE usuarios (
5      id INT AUTO_INCREMENT PRIMARY KEY,
6      nombre_usuario VARCHAR(50) NOT NULL,
7      email VARCHAR(100) NOT NULL UNIQUE,
8      contraseña VARCHAR(255) NOT NULL
9  );
10
11
12  CREATE TABLE tareas (
13      id INT AUTO_INCREMENT PRIMARY KEY,
14      usuario_id INT NOT NULL,
15      nombre_tarea VARCHAR(255) NOT NULL,
16      descripcion_tarea TEXT,
17      completada BOOLEAN DEFAULT FALSE,
18      FOREIGN KEY (usuario_id) REFERENCES usuarios(id) ON DELETE CASCADE
19  );
```

Es básicamente la base de datos a la que se conecta.

Styles.css

CÓDIGO:

```
body {  
    font-family: Arial, sans-serif;  
    text-align: center;  
    margin: 50px;  
}
```

```
.container {  
    width: 300px;  
    margin: auto;  
    padding: 20px;  
    border: 1px solid #ccc;  
    border-radius: 10px;  
}
```

```
a {  
    display: block;  
    margin: 10px;  
    padding: 10px;  
    background-color: #007BFF;  
    color: white;  
    text-decoration: none;  
    border-radius: 5px;  
}
```

```
a:hover {  
    background-color: #0056b3;  
}
```

CAPTURA:

```
# styles.css > ...
1  body {
2      font-family: Arial, sans-serif;
3      text-align: center;
4      margin: 50px;
5  }
6
7  .container {
8      width: 300px;
9      margin: auto;
10     padding: 20px;
11     border: 1px solid #ccc;
12     border-radius: 10px;
13 }
14
15 a {
16     display: block;
17     margin: 10px;
18     padding: 10px;
19     background-color: #007BFF;
20     color: white;
21     text-decoration: none;
22     border-radius: 5px;
23 }
24
25 a:hover {
26     background-color: #0056b3;
27 }
28 |
```

Es básicamente el css que servirá para darle un estilo a todas las páginas y que se vean más bonitas.

FUNCIONAMIENTO

Index.php:

En el índice veremos los apartados para registrarnos o iniciar sesión



Registro.php

Aquí pondremos el nombre del usuario, el correo y su contraseña para el registro

Registro de Usuario

Nombre de Usuario: Correo Electrónico: Contraseña:

¿Ya tienes cuenta?

Inicia sesión aquí

Registro de Usuario

Usuario registrado exitosamente

Nombre de Usuario: Correo Electrónico: Contraseña:

¿Ya tienes cuenta?

Inicia sesión aquí

Login.php

Escribiremos el correo y la contraseña del usuario que hemos registrado y una vez iniciada la sesión nos llevará a Tareas.php.

Iniciar Sesión

Correo Electrónico: Contraseña:

¿No tienes una cuenta?

Regístrate aquí

Tareas.php

Vamos a agregar una nueva tarea

The screenshot shows a web browser window with the URL `localhost:8080/programacion/hito2/tareas.php`. The page title is "Mis Tareas". Below the title is a section "Agregar nueva tarea" with two input fields: "Nombre de la tarea" (containing "Prueba1") and "Descripción de la tarea" (containing "Primera tarea, prueba para ver el"). There is a dropdown menu next to the description field and an "Agregar tarea" button. Below this is a section "Tareas pendientes" with a "Cerrar sesión" button.

Como vemos, aparece que la tarea se ha agregado con éxito y nos aparecerá una lista con las taras que tenemos pendientes.

The screenshot shows the same web browser window. A green message "Tarea agregada exitosamente" is displayed above the "Agregar nueva tarea" section. The "Tareas pendientes" section now contains a list of tasks. The first task is "Prueba1 - Primera tarea, prueba para ver el funcionamiento". Below the task name are two buttons: "Marcar como completada" and "Eliminar". The "Cerrar sesión" button is still at the bottom.

Vamos a marcar la tarea como completada y como vemos nos aparecerá que se ha completado, y se quita la opción de marcar como completada.

The screenshot shows the same web browser window. The URL now includes `?completar=6`. The "Tareas pendientes" section shows the task "Prueba1 - Primera tarea, prueba para ver el funcionamiento" with the word "Completada" in parentheses next to it. The "Marcar como completada" button is no longer visible, and the "Eliminar" button is still present. The "Cerrar sesión" button is at the bottom.

Ahora vamos a eliminar la tarea

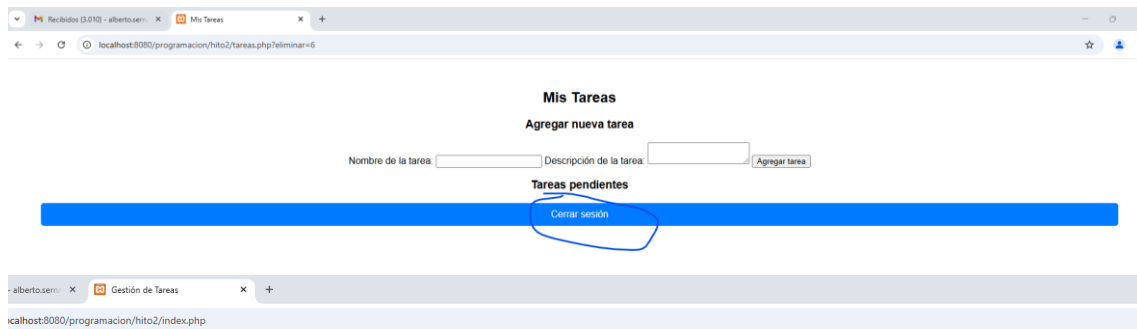


Como vemos se ha eliminado



Cerrar sesión.php

Por último, le daremos a cerrar sesión y nos cerrará la sesión y nos redijera al index.php.



ENLACE GITHUB: https://github.com/sserna45/Programacion_H2_2-T_AlbertoSerna

BIBLIOGRAFIA

ChatGPT. (n.d.). Chatgpt.com. Retrieved February 13, 2025, from

<https://chatgpt.com/>

Sesiones en PHP. (n.d.). Com.Es. Retrieved February 13, 2025, from

<https://diego.com.es/sesiones-en-php>

W3schools online web tutorials. (n.d.). W3schools.com. Retrieved February 13,

2025, from <https://www.w3schools.com/>