Índice

Sumario

Introducción

Dentro de nuestros sitios web puede ser que necesitemos disponer de una **página separada que solo sea accesible por medio de una contraseña**. Un lugar donde **solo el usuario** pueda entrar y muestre una información sensible:

- Configuración.
- Perfil.
- Datos bancarios.
- Conversaciones de chat.
- Historial.
- …

Cualquier elemento que sea único y privado para el visitante.

La estrategia que siempre se ha usado desde el Back-End es por medio de **sesiones**. Al introducir un usuario/contraseña se creará una **llave única** que dejará entrar a nuestro visitante, y ¡solo a él!. Si quisiera irse dispondrá de un botón para romper la sesión haciendo imposible que nadie pudiera entrar con el equipo. A no ser que vuelva a identificarse, claro.

Para el ejemplo necesitaremos 3 páginas: **login.php** (donde nos identificamos), **privado.php** (donde entraremos) y **cerrar.php** (código que destruirá el acceso).

Cuando hablamos de un login nos referimos a un formulario donde podamos dar nuestro nombre de usuario y contraseña. En el ejemplo vamos a usar un email en lugar del nombre.

Si los datos son incorrectos mostrará un mensaje de aviso.

Importante: Nunca debe desvelarse si el usuario se ha equivocado en el email o en la contraseña. Revelarías a un posible atacante que uno de los 2 datos son correctos. En su lugar dar una información más genérica: "**El email o contraseña es inválido**", **"Sus datos de acceso no son correctos**", etc…

Cifrando nuestra contraseña

Las contraseñas, por razones de seguridad, deben estar encriptadas en la base de datos. Para ello uno de los métodos recomendables es el siguiente:

```
<?php
// La contraseña es '123'
echo password_hash('123', PASSWORD_BCRYPT);
?>
```

```
$2y$10$OuIiaiZMrVb5nAzrBU4U8eBjCB/rMPEAnNlmM8krh0nZ5Fru/nO7q
```

La función **password_hash()** crea un nuevo hash de contraseña usando un algoritmo de hash fuerte de único sentido. Actualmente se admiten los siguientes algoritmos:

PASSWORD_DEFAULT - Usa el algoritmo bcrypt (predeterminado a partir de PHP 5.5.0). Observe que esta constante está diseñada para cambiar siempre que se añada un algoritmo nuevo y más fuerte a PHP. Por esta razón, la longitud del resultado de usar este identificador puede cambiar con el tiempo. Por lo tanto, se recomienda almacenar el

resultado en una columna de una base de datos de más de 60 caracteres (255 caracteres sería una buena elección).

PASSWORD_BCRYPT - Usa el algoritmo CRYPT_BLOWFISH para crear el hash. Producirá un hash estándar compatible con crypt(). El resultado siempre será un string de 60 caracteres, o false en caso de error.

Si ejecutas el código comprobarás para tu sorpresa que es diferente del mío. Incluso si lo vuelves a ejecutar… será ¡diferente al anterior!

El resultado varía en cada ocasión. ¿Cómo diablos se puede comprobar un sistema tan aparentemente aleatorio? Con la magia de la criptografía.

```
<?php
if(password_verify('123',
'$2y$10$7MVMltyzE/cWFEbfqsz24.QgQArmlShk8VvnPpVGISlAt9mDmjaH2')){
echo "<p>Correcta</p>";
} else {
echo "<p>Incorrecta</p>";
}
?>
```

```
<p>Correcta</p>
```

La función **password_verify** comprueba que la contraseña coincida con un hash. Ahora veamos todo unido. En los siguientes ficheros haremos uso de las sesiones vistas en una sección anterior.

Login

En primer lugar vamos a crear el archivo **login.php**.

```
<!DOCTYPE html> <html lang="es">
  <head>
  <meta charset="utf-8">
  <title>Login</title>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>
<?php
// Comprobamos que nos llega los datos del formulario
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
  // Base de datos ficticia que se usará en el ejemplo.
  $baseDeDatos = [ 'email' => 'correo@ejemplo.com', 'password' => password_hash('123',
PASSWORD_BCRYPT) ];
  // Variables del formulario
  $emailFormulario = isset($_REQUEST['email']) ?
  $_REQUEST['email'] : null;
  $contrasenaFormulario = isset($_REQUEST['contrasena']) ?
  $_REQUEST['contrasena'] : null;

  // Comprobamos si los datos son correctos
     if ($baseDeDatos['email'] == $emailFormulario &&
```

```
password_verify($contrasenaFormulario, $baseDeDatos['password'])) {
      // Si son correctos, creamos la sesión
      session_name("sesion-privada");
      session_start();
      $_SESSION['email'] = $_REQUEST['email'];
      // Redireccionamos a la página privada
      header('Location: privado.php');
      exit();
      }
    else {
      // Si no son correctos, informamos al usuario
      print'<p style="color: red">El email o la contraseña es incorrecta.</p>';
      }
}
?>
<form method="post">
  <p> <input type="text" name="email" placeholder="Email"> </p>
  <p> <input type="password" name="contrasena" placeholder="Contraseña"> </p>
  <p> <input type="submit" value="Entrar"> </p>
</form>
</body>
</html>
```

Zona privada

Comprobamos si existe la sesión, en caso contrario devolvemos a login.php de forma automática.

Llamaremos el archivo privado.php.

```
<?php
// Activa las sesiones
session_name("sesion-privada");
session_start();
// Comprueba si existe la sesión "email", en caso contrario vuelve a la página de login
if (!isset($_SESSION["email"])) header("Location: login.php");
?><!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width,
initial-scale=1.0">
<title>Privado</title>
</head>
<body>
<p>
¡Te encuentras en una zona secreta!, solo visible por una persona identificada.
</p>
<p>
<a href="cerrar-sesion.php">Cerrar sesión</a>
</p>
</body>
</html>
```

Cerrar sesión

Destruimos las sesiones y redireccionamos a **login.php**.

Llamaremos el archivo **cerrar-sesion.php**.

```php
<?php
 // Inicia las sesiones
 session_name("sesion-privada");
 session_start();
 // Destruye cualquier sesión del usuario
 session_destroy();
 // Redirecciona a login.php
 header('Location: login.php');
?>
```

Recuperar contraseña

Lo peor que puede ocurrir es que un usuario pierda una contraseña, tanto para él como para el programador que ha montado el sitio web. Para poder recuperar la contraseña tendremos que seguir una serie de pasos.

1. **Comprobar que el usuario es legítimo** y no un oportunista mediante un **enlace seguro**.
2. Al estar las contraseñas cifradas, como hemos visto anteriormente, no será posible enseñar la misma al usuario. No quedará más remedio que pedir una nueva contraseña al usuario.
3. **Sobrescribir la contraseña anterior** con la nueva.
4. Regañar al usuario. Puede ser sutilmente con un texto donde se le sugiere a guardarla en un sitio seguro.
5. Llevar al usuario nuevamente a la página de identificación (login.php).

La estrategia puede ser por medio de un sistema de verificación fiable. Como puede ser un SMS, un email, una notificación, un mensaje por medio de un chat… de aquí la importancia de dar unos datos reales.

¿Cómo puedo crear un enlace seguro?

Enviaremos un enlace por medio de un email con un número aleatorio el cual nosotros guardaremos y nos servirá para averiguar si el usuario es el mismo que nos está reclamando la nueva contraseña. En caso contrario lo ignoramos.
Por ejemplo un enlace con la siguiente estructura:

```
recuperar_contrasena.php?verificar=123456789
```

El valor de verificar se tiene que guardar para que se compruebe más adelante.
Una forma segura de generar el número es por medio de **openssl**.

```
<?php
echo bin2hex(openssl_random_pseudo_bytes(16));
// 60727b7a5f11688aad3662bbd62e065a
?>
```

La función **openssl_random_pseudo_bytes** genera una cadena de bytes pseudo-aleatoria, con el número de bytes determinado por el primer parámetro.

La función **bin2hex** convierte datos binarios en su representación hexadecimal. Devuelve por tanto una cadena ASCII que contiene la representación hexadecimal.

Siempre que ejecutemos el código anterior nos proporcionará un string alfanumérico fantásticamente difícil de predecir.

El código para generar el mensaje que deberíamos enviar al usuario quedaría tal que así.

```
<?php
  $tokenSeguro = bin2hex(openssl_random_pseudo_bytes(16));
  // Nuestro mensaje en HTML
  $mensaje = "
    <html>
      <head>
        <title>Recupera la contraseña</title>
      </head>
      <body>
        <a href=\"ejemplo.com?token=$tokenSeguro\">Pulsa aquí para cambiarla</a>
      </body>
    </html>
  ";
  // Cabeceras. Y la dirección del emisor.
  $headers = [
  'MIME-Version' => '1.0',
  'Content-type' => 'text/html; charset=utf-8',
  'From' => 'dwes@php.com'
  ];
  // Lo enviamos
  mail('correo@ejemplo.com', 'Recuperar contraseña', $mensaje, $headers);
?>
```

Para que realmente se envíe el correo anterior debemos tener montado un servidor SMTP que se monta en entornos de producción y que en nuestro caso no vamos a montar en local. Esto no quiere decir que no tengamos un medio para probar que el envío de email funciona y salen con el formato deseado.

En nuestro **php.ini** tenemos que modificar el servidor **SMTP** que vamos a usar para el envío del correo. Generalmente diremos que nuestro servidor de SMTP es "localhost" osea, nuestro propio ordenador local.

Abriremos por tanto nuestro fichero **php.ini** y configuraremos un par de líneas si no vienen ya por defecto.

```
SMTP = localhost
smtp_port = 25
```

En la versión 8.1.6 de XAMPP estos valores ya están por defecto con lo que no tendremos que modificar nada.

Ahora sólo nos queda configurar un "fake smtp" osea, un servidor falso. Con un servidor falso se puede apreciar que tu correo se ha creado y puedes ver el contenido de lo que le llegaría al email de destino. Aunque el email nunca se envía realmente.

En XAMPP existe en el php.ini una línea que hace referencia a un servidor falso que depende de la versión puede que ya esté configurado.

En la versión 8.1.6 de XAMPP no viene configurado y aparecen estas líneas:

```
; For Unix only. You may supply arguments as well (default:
"sendmail -t -i").
; https://php.net/sendmail-path
;sendmail_path =
```

En este caso vamos a guardar el email en el disco duro, en un archivo de texto, en vez de enviarse. Para ello vamos a hacer uso de un ejecutable proporcionado por XAMPP y que se encuentra en la carpeta “mailtodisk”. Por tanto en el fichero **php.ini** tendremos que descomentar la línea que hace referencia a **sendmail_path** y modificar su valor para que haga referencia a la ruta donde se encuentra **mailtodisk.exe.**

```
; XAMPP: Comment out this if you want to work with mailToDisk...
sendmail_path="C:\xampp\mailtodisk\mailtodisk.exe"
```

Ese servidor de SMTP falso simplemente te deja los emails en la carpeta de xampp/mailoutput. Dentro de ese directorio encontrarás un archivo de texto para cada email enviado, con nombres como "mail-20211128-1304-462000" (fíjate que el nombre del archivo contiene ya la fecha y hora en la que se generó). Simplemente es abrirlo para ver el email que generaste desde PHP y comprobar si era lo que esperabas.

Ya solo quedaría generar la página para comprobar que el token que hemos guardado es el mismo que nos ha llegado y si es así la realizar la gestión para que el usuario introduzca la nueva contraseña.

Registro

Registrar un usuario en nuestro sitio web es sencillo, solo hay que guardar los datos en la base de datos. Pero el asunto se complica cuando tenemos que comprobar si es un ser humano o no. Ahora mismo nos vamos a conformar con saber si la de correo electrónico es real. Y la mejor estrategia es enviar un correo con un texto secreto (token) que nosotros conozcamos previamente.

Los pasos para hace un buen registro por tanto son:

1. Pedir los datos para crear la cuenta.
2. Validar que los datos son compatibles con la base de datos (el email no existe, tiene un formato válido…).
3. Guardar los datos en la base de datos, y marcamos de momento que el usuario no

está activo.

4. Enviamos un token a la dirección de email que nos han dado para comprobar que es un usuario real.
5. Activar la cuenta si pulsan sobre el enlace y el token es correcto.
6. Llevar a la página de identificación y notificamos que su cuenta ha sido activada.

Ejemplo completo

A nuestra base de datos **dwes_tarde_pruebas** vamos a añadirle una nueva tabla llamada usuarios con la siguiente estructura.

Campo	Tipo	Clave Primaria
email	Cadena	Sí
password	Cadena	No
activo	Entero	No
token	Cadena	No

Página registro.php

En esta página vamos a tener un formulario, la validación de que el email no existe en la base de datos, la capacidad para guardar los datos y enviar el correo con el enlace para validar nuestra cuenta.

```php
<?php
  //-------------------------------------------------------
  // Funciones Para Validar
  //-------------------------------------------------------

  /**
   * Método que valida si un texto no está vacío
   * @param {string} - Texto a validar
   * @return {boolean}
  */
  function validarRequerido(string $texto): bool
  {
    return !(trim($texto) == "");
  }

  /**
  * Método que valida si el texto tiene un formato válido de email
  * @param {string} - Email
  * @return {bool}
  */
  function validarEmail(string $texto): bool
  {
    return (filter_var($texto, FILTER_VALIDATE_EMAIL) === FALSE) ? False : True;
  }

  /**
  * Método que establece la conexión con la base de datos
  * @param {string} - Host
  * @param {string} - Usuario
```

```php
 * @param {string} - Contraseña
 * @param {string} - Esquema de la base de datos
 * @return {PDO}
 */
function conectarPDO(string $host, string $user, string $password, string $bbdd): PDO
{
  try
  {
   $mysql="mysql:host=$host;dbname=$bbdd;charset=utf8";
   $conexion = new PDO($mysql, $user, $password);
   // set the PDO error mode to exception
   $conexion->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

  }
  catch (PDOException $exception)
  {
    exit($exception->getMessage());
  }
  return $conexion;
}

//-----------------------------------------------------
// Variables
//-----------------------------------------------------
$errores = [];
$email = isset($_REQUEST["email"]) ? $_REQUEST["email"] : "";
$password = isset($_REQUEST["password"]) ? $_REQUEST["password"] : "";
$host = "localhost";
$user = "dwes_tarde";
$passwordDB = "dwes_2223";
$bbdd = "dwes_tarde_pruebas";

// Comprobamos si nos llega los datos por POST
if ($_SERVER['REQUEST_METHOD'] == 'POST')
{

  //-----------------------------------------------------
  // Validaciones
  //-----------------------------------------------------
  // Email
  if (!validarRequerido($email)) {
    $errores[] = "Campo Email obligatorio.";
  }

  if (!validarEmail($email)) {
    $errores[] = "Campo Email no tiene un formato válido";
  }

  // Contraseña
  if (!validarRequerido($password)) {
    $errores[] = "Campo Contraseña obligatorio.";
  }

  /* Verificar que no existe en la base de datos el mismo email */
  // Conecta con base de datos
  $conexion = conectarPDO($host, $user, $passwordDB, $bbdd);
  // Cuenta cuantos emails existen
  $select = "SELECT COUNT(*) as numero FROM usuarios WHERE email = :email";
  $consulta = $conexion->prepare($select);
  // Ejecuta la búsqueda
```

```
    $consulta->execute([
      "email" => $email
    ]);
    // Recoge los resultados
    $resultado = $consulta->fetch();
    $consulta = null;
    // Comprueba si existe
    if ($resultado["numero"] > 0) {
      $errores[] = "La dirección de email ya esta registrada.";
    }

    //--------------------------------------------------
    // Crear cuenta
    //--------------------------------------------------
    if (count($errores) === 0) {
      /* Registro En La Base De Datos */
      // Prepara INSERT
      $token = bin2hex(openssl_random_pseudo_bytes(16));
      $insert = "INSERT INTO usuarios (email, password, activo, token) VALUES
(:email, :password, :activo, :token)";
      $consulta = $conexion->prepare($insert);
      // Ejecuta el nuevo registro en la base de datos
      $consulta->execute([
        "email" => $email,
        "password" => password_hash($password, PASSWORD_BCRYPT),
        "activo" => 0,
        "token" => $token
      ]);
      $consulta = null;
      /* Envío De Email Con Token */
      // Cabecera
      $headers = [
        "From" => "dwes@php.com",
        "Content-type" => "text/plain; charset=utf-8"
      ];
      // Variables para el email
      $emailEncode = urlencode($email);
      $tokenEncode = urlencode($token);
      // Texto del email
      $textoEmail = "
      Hola!\n
      Gracias por registrate en la mejor plataforma de internet, demuestras inteligencia.\
n
      Para activar entra en el siguiente enlace:\n
      http://localhost.../verificar-cuenta.php?email=$emailEncode&token=$tokenEncode
        ";
      // Envio del email
      mail($email, 'Activa tu cuenta', $textoEmail, $headers);

      /* Redirección a login.php con GET para informar del envío del email */

      header('Location: identificarse.php?registrado=1');
      exit();
    }

  }
?>
<!DOCTYPE html>
<html lang="es">
<head>
```

```
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Registro</title>
</head>
<body>
  <h1>Registro</h1>
  <!-- Mostramos errores por HTML -->
  <?php if (isset($errores)): ?>
  <ul class="errores">
    <?php
      foreach ($errores as $error)
      {
        echo '<li>' . $error . '</li>';
      }
    ?>
  </ul>
  <?php endif; ?>
  <!-- Formulario -->
  <form action="" method="post">
    <p>
      <!-- Campo de Email -->
      <label>
        Correo electrónio
        <input type="text" name="email">
      </label>
    </p>
    <p>
      <!-- Campo de Contraseña -->
      <label>
        Contraseña
        <input type="password" name="password">
      </label>
    </p>
    <p>
      <!-- Botón submit -->
      <input type="submit" value="Registrarse">
    </p>
  </form>
</body>
</html>
```

Página verificar-cuenta.php

A esta página se llega a través del enlace enviado después del registro. Si el email y token coinciden con lo guardado en la base de datos, cambiaremos el campo activo a 1 (marcamos como habilitado o verificado).

```
<?php
 /**
 * Método que establece la conexión con la base de datos
 * @param {string} - Host
 * @param {string} - Usuario
 * @param {string} - Contraseña
 * @param {string} - Esquema de la base de datos
 * @return {PDO}
 */
```

```php
function conectarPDO(string $host, string $user, string $password, string $bbdd): PDO
{
 try
 {
  $mysql="mysql:host=$host;dbname=$bbdd;charset=utf8";
  $conexion = new PDO($mysql, $user, $password);
  // set the PDO error mode to exception
  $conexion->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

 }
 catch (PDOException $exception)
 {
   exit($exception->getMessage());
 }
 return $conexion;
}

//-----------------------------------------------------
// Variables
//-----------------------------------------------------
$email = isset($_REQUEST["email"]) ? urldecode($_REQUEST["email"]) : "";
$token = isset($_REQUEST["token"]) ? urldecode($_REQUEST["token"]) : "";
$host = "localhost";
$user = "dwes_tarde";
$passwordDB = "dwes_2223";
$bbdd = "dwes_tarde_pruebas";

//-----------------------------------------------------
// COMPROBAR SI SON CORRECTOS LOS DATOS
//-----------------------------------------------------
// Conecta con base de datos
$conexion = conectarPDO($host, $user, $passwordDB, $bbdd);
// Prepara SELECT para obtener la contraseña almacenada del usuario
$select = "SELECT COUNT(*) as numero FROM usuarios WHERE email = :email AND token
= :token AND activo = 0";
$consulta = $conexion->prepare($select);
// Ejecuta consulta
$consulta->execute([
  "email" => $email,
  "token" => $token
]);

$resultado = $consulta->fetch();
$consulta = null;

// Existe el usuario con el token
if ($resultado["numero"] > 0)
{
 //-----------------------------------------------------
 // ACTIVAR CUENTA
 //-----------------------------------------------------
 // Prepara la actualización
 $update = "UPDATE usuarios SET activo = 1 WHERE email = :email";
 $consulta = $conexion->prepare($update);
 // Ejecuta actualización
 $consulta->execute([
   "email" => $email
 ]);
 //-----------------------------------------------------
 // REDIRECCIONAR A IDENTIFICACIÓN
```

```
  //-------------------------------------------------------
  header('Location: identificarse.php?activada=1');
  exit();
 }

 // No es un usuario válido, le enviamos al formulario de identificación
 header('Location: identificarse.php');
 exit();
?>
```

Página identificarse.php

En esta página se muestra información relativa a si la cuenta está activada o el usuario registrado, pero también es la encargada de comprobar si los datos son correctos. Y en ese caso, crear una sesión y llevar a la página privada.

```
<?php

 /**
 * Método que establece la conexión con la base de datos
 * @param {string} - Host
 * @param {string} - Usuario
 * @param {string} - Contraseña
 * @param {string} - Esquema de la base de datos
 * @return {PDO}
 */
 function conectarPDO(string $host, string $user, string $password, string $bbdd): PDO
 {
  try
  {
   $mysql="mysql:host=$host;dbname=$bbdd;charset=utf8";
   $conexion = new PDO($mysql, $user, $password);
   // set the PDO error mode to exception
   $conexion->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

  }
  catch (PDOException $exception)
  {
    exit($exception->getMessage());
  }
  return $conexion;
 }

 //-------------------------------------------------------
 // Variables
 //-------------------------------------------------------
 $errores = [];
 $email = isset($_REQUEST["email"]) ? $_REQUEST["email"] : null;
 $password = isset($_REQUEST["contrasena"]) ? $_REQUEST["contrasena"] : null;
 $host = "localhost";
 $user = "dwes_tarde";
 $passwordDB = "dwes_2223";
 $bbdd = "dwes_tarde_pruebas";

 // Comprobamos que nos llega los datos del formulario
 if ($_SERVER['REQUEST_METHOD'] == 'POST')
 {
```

```php
  //-------------------------------------------------------
  // COMPROBAR SI LA CUENTA ESTÁ ACTIVA
  //-------------------------------------------------------
  // Conecta con base de datos
  $conexion = conectarPDO($host, $user, $passwordDB, $bbdd);
  // Prepara SELECT para obtener la contraseña almacenada del usuario
  $select = "SELECT activo, password FROM usuarios WHERE email = :email;";
  $consulta = $conexion->prepare($select);
  // Ejecuta consulta
  $consulta->execute([
     "email" => $email
  ]);
  // Guardo el resultado
  $resultado = $consulta->fetch();

  // Al hacer comprobación con el tipo convertimos el resultado a entero
  if ((int) $resultado["activo"] !== 1)
  {
   $errores[] = "Tu cuenta aún no está activa. ¿Has comprobado tu bandeja de correo?";
  }
  else
  {

   //-------------------------------------------------------
   // COMPROBAR LA CONTRASEÑA
   //-------------------------------------------------------
   // Comprobamos si es válida
   if (password_verify($password, $resultado["password"]))
   {
    // Si son correctos, creamos la sesión
    session_start();
    $_SESSION["email"] = $email;
    // Redireccionamos a la página segura
    header("Location: privado.php");
    exit();
   }
   else
   {
    $errores[] = "El email o la contraseña es incorrecta.";
   }
  }
 }
?>
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Entrada</title>
</head>
<body>
  <h1>Entrar</h1>
  <!-- Mostramos errores por HTML -->
  <?php if (count($errores) > 0): ?>
   <ul class="errores">
     <?php
       foreach ($errores as $error)
         {
```

```
            echo "<li>" . $error . "</li>";
        }
      ?>
    </ul>
  <?php endif; ?>
  <!-- Mensaje de aviso al registrarte -->
  <?php if(isset($_REQUEST["registrado"])): ?>
    <p>¡Gracias por registrarte! Revisa tu bandeja de correo para activar la cuenta.</p>
  <?php endif; ?>
  <!-- Mensaje de cuenta activa -->
  <?php if(isset($_REQUEST["activada"])): ?>
    <p>¡Cuenta activada!</p>
  <?php endif; ?>
  <!-- Formulario de identificación -->
  <form method="post">
    <p>
      <input type="text" name="email" placeholder="Email">
    </p>
    <p>
      <input type="password" name="contrasena" placeholder="Contraseña">
    </p>
    <p>
      <input type="submit" value="Entrar">
    </p>
  </form>
</body>
</html>
```

Página privado.php

Al igual que vimos anteriormente, la zona privada va tras la identificación.

```
<?php
// Activa las sesiones
session_start();
// Comprueba si existe la sesión 'email', en caso contrario vuelve a la página de
identificación
if (!isset($_SESSION['email']))
{
  header('Location: identificarse.php');
  die();
}
?>
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>>Panel secreto</title>
</head>
<body>
  <h1>Tu panel secreto</h1>
</body>
</html>
```