

Segundo trimestre

Programación

Sergio Felipe García

28 de Febrero del 2023



Índice

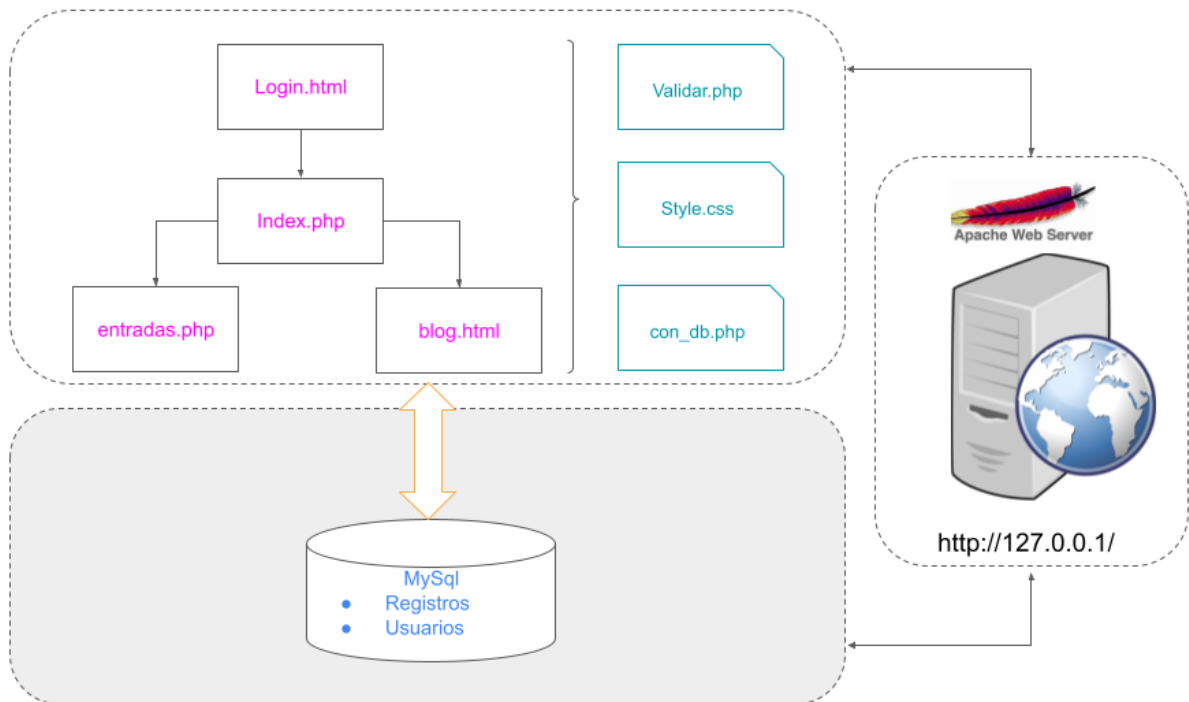
Fase 1	2
Arquitectura de la Solución.	2
Tareas a desarrollar o pasos	4
Algoritmo	5
Fase 2	6
Diseño visual del Front End.	6
Inicio de sesión de un usuario.	8
Página de inicio.	11
Script de almacenamiento de la cookie	12
Almacenamiento de los datos del post.	15
Recuperar los datos de las entradas desde BBDD.	17
Fase 3	18
Errores encontrados durante el desarrollo.	18

Fase 1

Esta primera fase de diseño y análisis nos debe permitir abordar la implementación de la aplicación web con garantías. Para ello, proponemos explicar con un algoritmo la tarea de publicar una entrada. Tras definir el algoritmo, puedes utilizar un caso de uso para dejar más clara la funcionalidad.

Arquitectura de la Solución.

Para que me ayude explicar gráficamente la implementación de la aplicación web, voy a plasmar de forma básica los componentes que he utilizado en la siguiente figura:

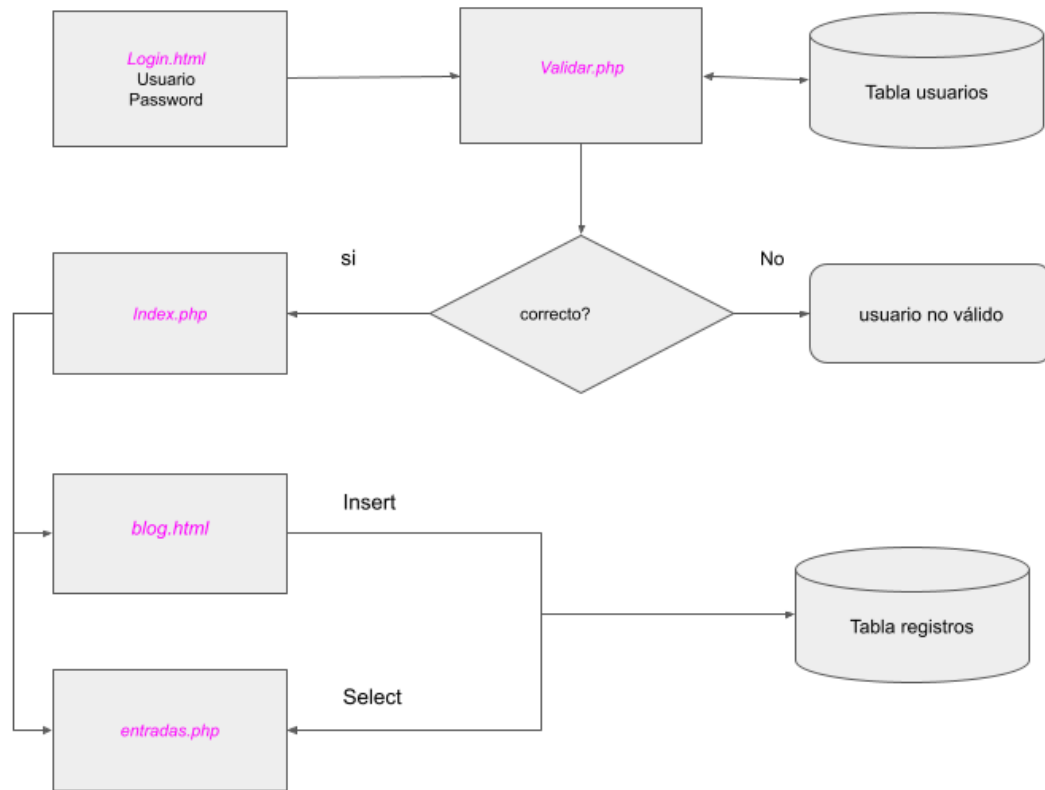


Tareas a desarrollar o pasos

Por consiguiente teniendo en cuenta todos los componentes que aquí aparecen, es necesario realizar una serie de tareas que voy a detallar a continuación:

	TAREAS
1	<ol style="list-style-type: none">1. Instalar servidor Apache (Xamp) en el equipo local2. Arrancar una instancia de BBDD MySql.3. Crear una bbdd llamada <i>registros.sql</i>
2	<ol style="list-style-type: none">1. Diseñar y desarrollar una página de <i>Login.html</i> (utilizando Html y Css) <i>style.css</i>2. Crear la <i>tabla usuarios</i> en MySql3. Insertar usuarios con los privilegios de acceso.4. Guardar las cookies de la ip y fecha de acceso para mantener la sesión.5. Validar contra la base de datos que un usuario y su clave son correctos. <i>validar.php</i>
3	<ol style="list-style-type: none">1. Diseñar y desarrollar una página <i>Index.php</i> (utilizando Html y Css)2. Insertar el contenido de la pág. para su correcta visualización.
4	<ol style="list-style-type: none">1. Diseñar y desarrollar un formulario que permita al usuario escribir un post y confirmarle que el registro fue exitoso. <i>blog.html</i>2. Utilizar PHP para almacenar el contenido del post en BBDD. <i>con_db.php</i>.3. Crear una <i>tabla datos</i> en la BBDD para almacenar dicho contenido.
5	<ol style="list-style-type: none">1. Diseñar y desarrollar una página <i>entradas.php</i> que muestre todas las entradas publicadas por los usuarios.2. Leer los datos de la base de datos para mostrarlos en la pág.

Algoritmo





Fase 2

En esta segunda fase se realiza la implementación de la aplicación y la construcción de todos los elementos web necesarios para su correcto funcionamiento. Debemos poner especial cuidado en cumplir los estándares de accesibilidad y usabilidad establecidos por W3C, por ejemplo, en lo referente a validación de formularios, confirmación de registro, comunicación y recepción de datos.




Diseño visual del Front End.

Dado que se nos permite elegir las herramientas, He optado para el diseño visual el toolkit de Bootstrap ya que me permite de forma rápida crear un interfaz para el usuario final atractivo y que sea responsive.

Además también me permite crear una barra de navegación clara y sencilla que permita al usuario moverse por las diferentes funcionalidades.

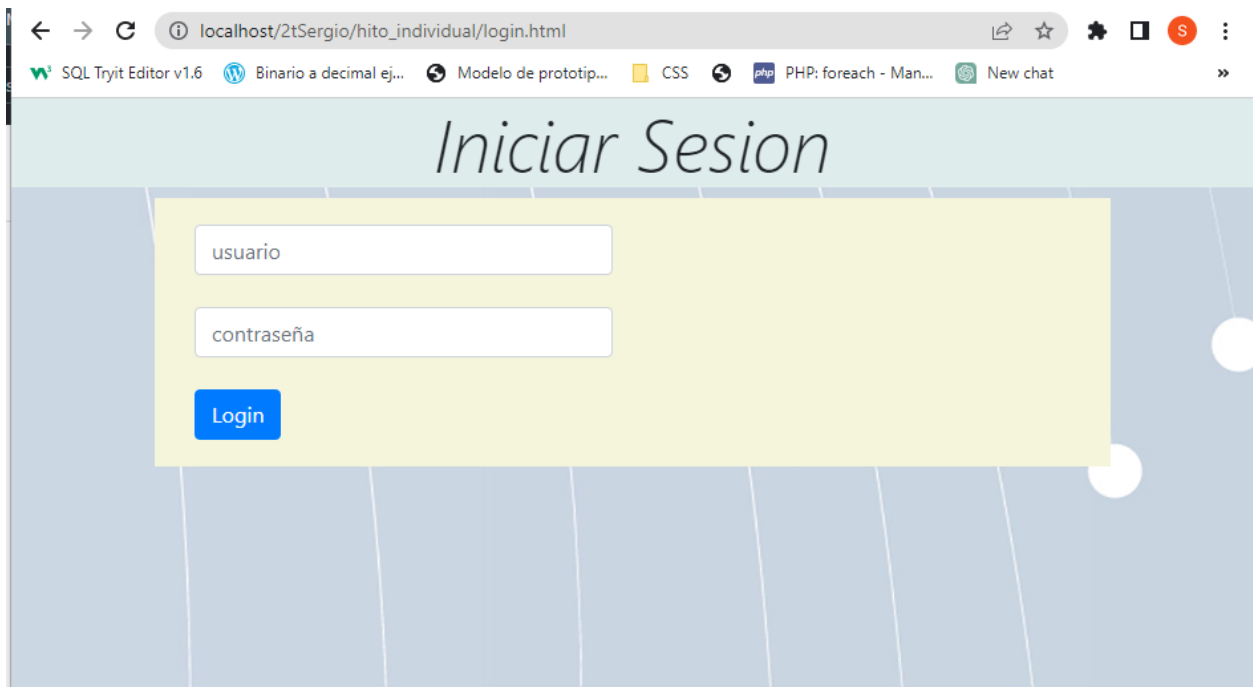
A continuación muestro a modo de ejemplo un trózo del código “ style.css” que voy a usar para toda la aplicación:

```

# style.css >  html
1
2  html {
3      height: 100%; /* Deseo que mi página ocupe toda la pantalla */
4      width: 100%;
5  }
6  .header{
7      text-align: center;
8      background-color:  rgb(222, 236, 236);
9      margin: 0%;
10     padding: 0%;
11 }
12 }
13 body{
14     background-image: url(img/footer.jpg);
15 }
16 .header h1{
17     text-align: center;
18 }
19
20 nav ul li { /* podemos elegir más de un elemento a formatear*/
21     float: left; /* con esto hacemos que el texto salga en horizontal*/
22     list-style: none;
23     margin: 8px;
24     text-align: center;
25 }
26 }
27
28 .container {
29     max-width: 800px;
30     margin: 0 auto;
31     padding: 20px;
32     background-color:  beige;
33 }
34 h1 {
35     font-size: 32px;
36     margin-bottom: 20px;
37     margin: 0%;
38     padding: 0%;
39 }
40
41
42 h2 {
43     font-size: 24px;
44     margin-top: 40px;
45     margin-bottom: 10px;
46 }
47 p {
48     font-size: 18px;
49     line-height: 1.5;

```

Inicio de sesión de un usuario.



The image shows a web browser window with the address bar displaying 'localhost/2tSergio/hito_individual/login.html'. The browser's taskbar at the top includes icons for 'SQL Tryit Editor v1.6', 'Binario a decimal ej...', 'Modelo de prototip...', 'CSS', 'PHP: foreach - Man...', and 'New chat'. The webpage itself has a light blue header with the title 'Iniciar Sesión' in a large, elegant font. Below the header is a yellow rectangular form containing two white input fields: the first is labeled 'usuario' and the second is labeled 'contraseña'. A blue button with the text 'Login' is positioned at the bottom left of the form. The background of the page is a light blue with a subtle pattern of white circles.

Para poder acceder a la aplicación, un usuario debe poseer un usuario y contraseña. Estos datos serán introducidos en el formulario que se muestra en la imagen. Al enviar los datos se comprueba si dicho usuario está en la BBDD para poder continuar navegando.

Esta comprobación la realizamos utilizando el metodo “post” y llamando al programa [validar.php](#)

```
login.html > html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <link rel="stylesheet" type="text/css" href="style.css">
8   <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.4.1/dist/css/bootstrap.min.css"
9     integrity="sha384-Vkoo8x4CGs03+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh" crossorigin="anonymous">
10  <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.4.1/dist/js/bootstrap.min.js"
11    integrity="sha384-wfSDF2E50V2D1uUdj003uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl30g8ifw86" crossorigin="anonymous"></script>
12  <title>Formulario de login</title>
13 </head>
14 <body>
15   <header class="header">
16     <h1 class="display-4 font-italic">Iniciar Sesion</h1>
17   </header>
18   <div class="container">
19     <form action="validar.php" method="post">
20       <div class="col-md-6">
21         <input type="text" class="form-control" id="inputEmail4" placeholder="usuario" name="usuario">
22         <br>
23       </div>
24       <div class="col-md-6">
25         <input type="password" class="form-control" placeholder="contraseña" name="clave">
26         <br>
27       </div>
28       <div class="col-12">
29         <input type="submit" class="btn btn-primary" value="Login">
30       </div>
31     </form>
32   </div>
33 </body>
34 </html>
```

Validar.php como su nombre indica, válida si los datos de usuario introducidos en el formulario se encuentran en la BBDD.

Para ello se realiza una conexión utilizando la función [mysqli_connect \(\)](#) a la cual le pasamos los datos de la instancia que hemos creado anteriormente en Mysql desde MySql admin del cual mostraremos un pantallazo posteriormente.

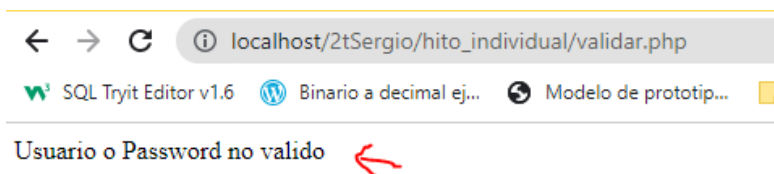
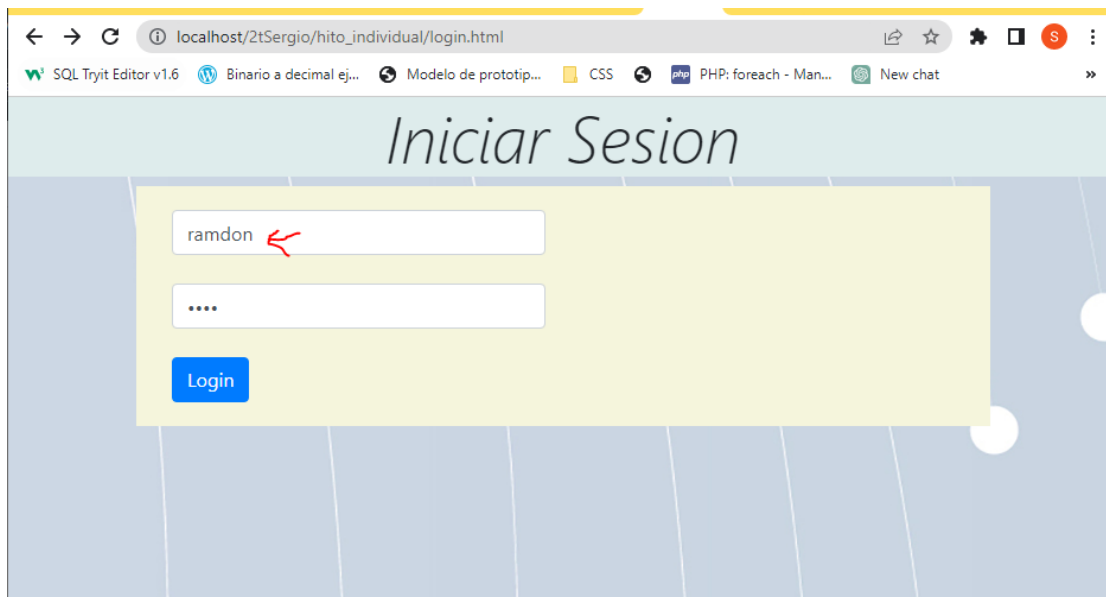
Este script en PHP se encarga de verificar las credenciales de un usuario y redirigirlo a una página de inicio de sesión en caso de que las credenciales sean correctas, o mostrar un mensaje de error en caso contrario.

```

validar.php
1  <?php
2  $usuario=$_POST ['usuario'];
3  $clave=$_POST ['clave'];
4
5  $conexion=mysqli_connect("localhost","root","","registros");
6
7  $consulta="SELECT * FROM usuarios WHERE usuario='$usuario' and clave='$clave' ";
8
9  $resultado=mysqli_query($conexion,$consulta);
10
11 $filas=mysqli_num_rows($resultado);
12 if ($filas>0){
13     header("location:index.php");
14 }
15 }
16 else {
17     echo('Usuario o Password no valido');
18 }
19 mysqli_close($conexion);
20 ?>

```

A continuación muestro un ejemplo de una validación incorrecta.

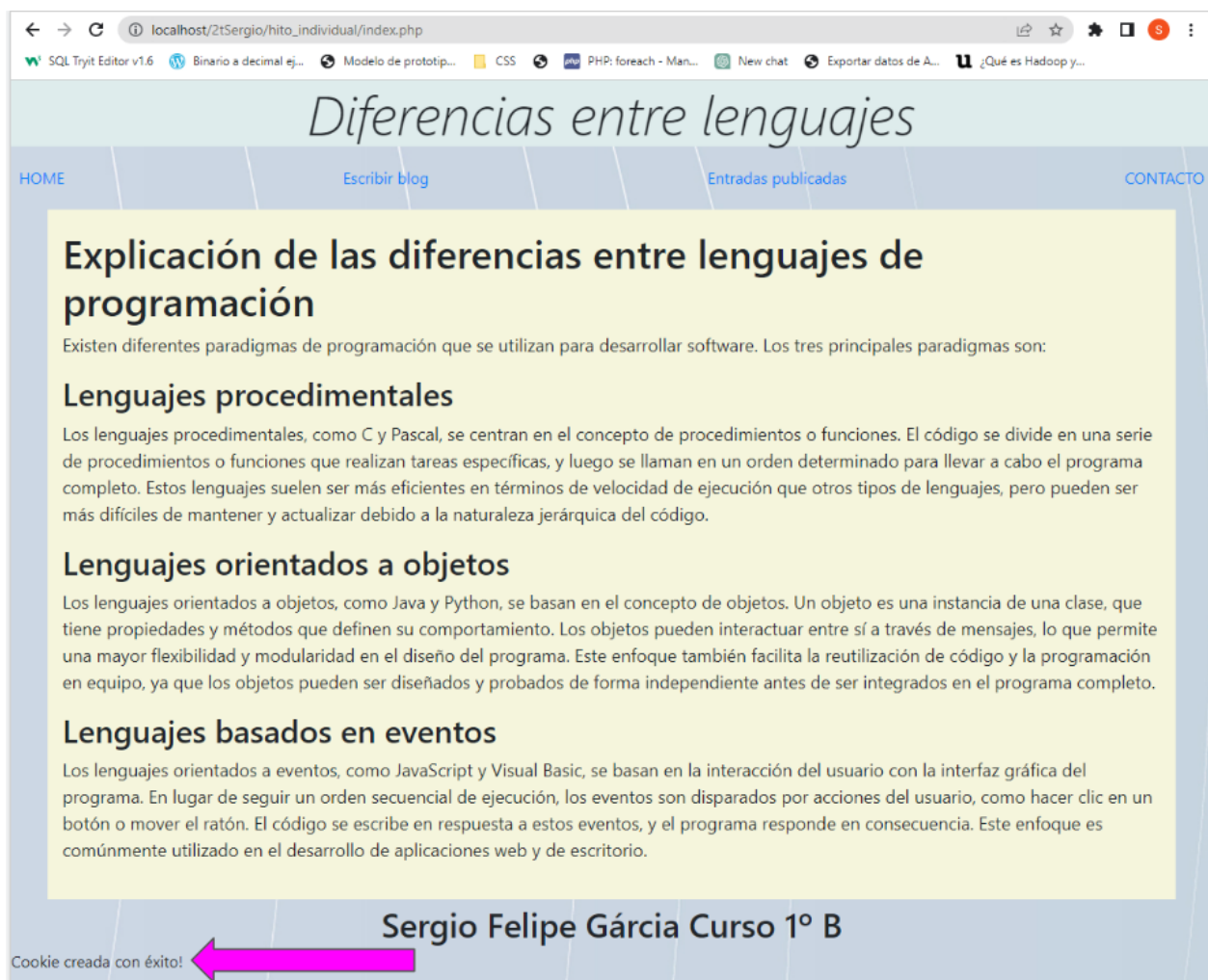


Página de inicio.

Utilizando la plantilla generada con nuestra hoja de estilo, diseñamos una página *index.php* cuyo contenido es una explicación de las diferencias entre lenguajes de programación orientada a objetos, a eventos y lenguajes procedimentales.

Además incluimos una barra de navegación que permita al usuario navegar por las diferentes opciones de nuestra aplicación.

Por último tal y como se solicita, al acceder a esta página almaceno en una cookie la IP del equipo que está accediendo y la fecha de acceso (lo hacemos un script PHP) y mostramos en pantalla a modo de traceo que la operación fue correcta. Lo indico con una flecha en la imagen.



Script de almacenamiento de la cookie

```
<?php

// Obtener la dirección IP del usuario
$ip = $_SERVER['REMOTE_ADDR'];


// Obtener la fecha y hora actual
$date = date("Y-m-d H:i:s");

// Crear la cookie con la IP y fecha de acceso
setcookie('acceso', $ip . ',' . $date, time() + (60 * 60 * 24 * 30), '/');

// Mostrar un mensaje al usuario confirmando la creación de la cookie
echo "Cookie creada con éxito!";

?>
```

1. Obtenemos la dirección IP del usuario utilizando la variable superglobal `$_SERVER['REMOTE_ADDR']`. Esta variable contiene la dirección IP del cliente que realizó la solicitud al servidor web.
 - a. Una variable superglobal en PHP es una variable predefinida que está disponible en cualquier parte del código PHP, incluyendo dentro de funciones y clases, sin necesidad de hacer nada especial para acceder a ella. Contienen información útil sobre el entorno en el que se está ejecutando el código PHP.
 - b. Las variables superglobales en PHP siempre empiezan con un guión bajo seguido del nombre de la variable. Por ejemplo, `$_SERVER` es una variable que contiene información, como el nombre del servidor, el protocolo utilizado, la dirección IP del cliente, entre otros.
 - c. Algunas otras variables superglobales importantes en PHP son `$_GET`, `$_POST`, `$_COOKIE`, `$_SESSION`, `$_FILES`, entre otras. Es recomendable que lea que datos puedo obtener de estas variables ya que pueden ayudarme de forma muy útil.
2. La función `date()` devuelve la fecha y hora. En este caso, el formato es "Y-m-d H:i:s", que representa el año, mes, día, hora, minuto y segundo actual.

- 
3. Llamo a la cookie "acceso" utilizando la función `setcookie()`. Esta cookie contiene la dirección IP del usuario y la fecha y hora de acceso concatenados en una cadena separada por comas. La cookie se establece para todo el sitio web ("/") y dura 30 días.
 4. A modo de comprobación del código muestro un mensaje indicando que la cookie ha sido creada con éxito.

El almacenamiento de cookies es un método que me permite acceder a ella `$_COOKIE` para tomar decisiones dentro de mi código. OJO es importante saber que estos datos se almacenan en el equipo del usuario final en un fichero tipo texto que puede ser modificado por lo que tenemos que tomar ciertas precauciones y no sirven para todo.

Insertar un nuevo Post

Página que permite al usuario escribir un post del blog mediante un formulario del cual recoge los datos y los almacena en la base de datos.

El diseño del formulario se ha realizado siguiendo criterios de usabilidad como el muestra la imagen el cual indica en un mensaje al usuario que debe introducir una dirección de correo electrónico.

The screenshot displays a web browser window with the URL `localhost/2tSergio/hito_individual/blog.html`. The page title is *Escribe tu Post*. The navigation bar includes links for [HOME](#), [Escribir blog](#), and [Entradas publicadas](#). The form contains the following elements:

- Título Post:** A text input field with the placeholder text "Titulo post".
- Autor escribe tu email:** A text input field containing the text "ñlkjñlkajdfs". A validation error message is overlaid on this field: "Incluye un signo "@" en la dirección de correo electrónico. La dirección "ñlkjñlkajdfs" no incluye el signo "@"."
- Contenido:** A large text area for the post content.
- Fecha de publicación:** A date picker set to "dd/mm/aaaa".
- File Upload:** A section with a "Seleccionar archivo" button, the text "Ninguno archivo selec.", and a "Cargar Archivo" button.
- Enviar Post:** A blue button to submit the form.

Almacenamiento de los datos del post.

Para insertar los datos en la base de datos utilizamos un script PHP llamado `con_db.php`.

He comentado todos los pasos en propio script tal y como muestra la imagen.

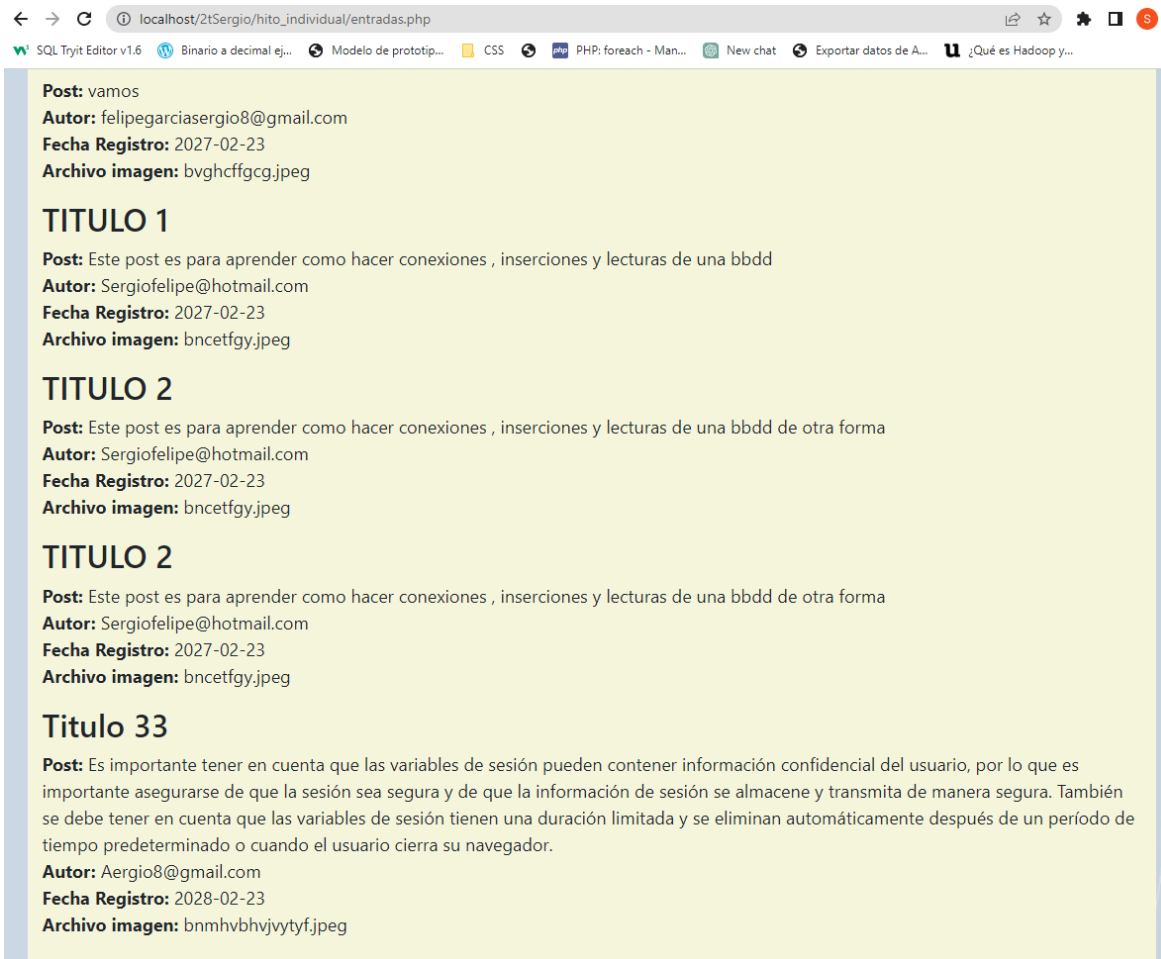
```
con_db.php
43
44 // Con la función strlen comprobamos que al menos tenemos información en los campos más importantes del formulario.
45 // Para asegurarme de que no se almacenan espacios en blanco innecesarios en la base de datos utilizo la función trim() que los quita de delante y detrás.
46
47 if (isset($_POST['register'])) {
48     if (strlen($_POST['email']) >= 1 && strlen($_POST['titulo']) >= 1 && strlen($_POST['contenido']) >= 1) {
49
50         // Obtenemos los datos del formulario
51         $email = trim($_POST['email']);
52         $titulo = trim($_POST['titulo']);
53         $contenido = $_POST['contenido'];
54         $fecha = date("d/m/y");
55         $imagen = $_POST['imagen'];
56
57         // creo la variable $consulta para que contenga la cadena de la sentencia SQL
58         $consulta = "INSERT INTO datos(email,titulo,fecha,contenido,imagen) VALUES ('$email','$titulo','$fecha','$contenido','$imagen')";
59
60         // la función mysqli_query() de PHP sirve ejecutar la sentencia SQL en la base de datos almacenamos el resultado en su variable.
61         $resultado = mysqli_query($conexion,$consulta);
62         if ($resultado) {
63             ?>
64             <h3 class="ok">¡Tu post se ha grabado correctamente en nuestro blog!</h3>
65             <?php
66         } else {
67             ?>
68             <h3 class="bad">¡Ups ha ocurrido un error!</h3>
69             <?php
70         }
71     } else {
72         ?>
73         <h3 class="bad">¡Por favor complete los campos!</h3>
74         <?php
75     }
76 }
77
78 ?>
79
80
81 </body>
82 </html>
```

Una vez grabado el post mostramos un mensaje al usuario indicando que ha ido bien.



Consultar las entradas del blog.

Página que permite al usuario leer las entradas que se han publicado por los usuarios.



Recuperar los datos de las entradas desde BBDD.

A Continuación muestro el código con los comentarios explicando que hace cada parte del código. Básicamente este script el lugar de hacer una inserción en base de datos hace una consulta.

```
<div class="container">
  <?php
  // nos conectamos a MySql con la funcion mysqli_connect()
  $inc = mysqli_connect("localhost","root","","registros");

  // este if comprueba que la conexión fue ok
  if ($inc) {

    // metemos en la variable consulta la cadena SQL que queremos lanzar y la lanzamos con la funcion mysqli_query() y guardo en resultado en $resultado.

    $consulta = "SELECT * FROM datos order by fecha";
    $resultado = mysqli_query($inc,$consulta);

    // si la consulta fue ok, hago un bucle para ir guardando fila a fila los registros. los almaceno en $row
    if ($resultado) {
      while ($row = $resultado->fetch_array()) {
        // la funcion fetch_array() devuelve una fila de la tabla y los miento a la variable $row en cada iteración del bucle
        // ahora saco los valores de las columnas

        $email = $row['email'];
        $titulo = $row['titulo'];
        $contenido = $row['contenido'];
        $fecha = $row['fecha'];
        $imagen = $row['imagen'];
        ?>
        <div>
          <h2><?php echo $titulo; ?></h2>
          <div>
            <p>
              <b>Post: </b> <?php echo $contenido; ?><br>
              <b>Autor: </b> <?php echo $email; ?><br>
              <b>Fecha Registro: </b> <?php echo $fecha; ?><br>
              <b>Archivo imagen: </b> <?php echo $imagen; ?><br>
            </p>
          </div>
        </div>
        <?php
      }
    }
  }
  ?>
</div>
</body>
```

Fase 3

Errores encontrados durante el desarrollo.

Durante el desarrollo de la aplicación he tratado de identificar y corregir errores según los he ido encontrando.

Errores en la hoja de estilo, aunque Bootstrap me permite diseñar rápidamente un sitio utilizando una plantilla más o menos adecuada. Todavía no conozco la funcionalidad y eso hace que algunas veces encuentre errores en el formato.

Errores en las conexiones a la base de datos. He intentado utilizar PDO para establecer una conexión. Sin embargo no lo he logrado porque en todo momento mostraba el error de *“el usuario no tiene permisos de acceso”*. Sin embargo utilizando los mismos detalles de conexión (host, nombre bbdd, usuario y contraseña) con la función *mysqli_connect ()*, ha funcionado correctamente. Aunque se supone que PDO proporciona más información para corregir errores yo no he conseguido solucionar el problema.

Entiendo que debo realizar más pruebas y verificaciones ya que PDO me parece el método que debería haber utilizado para este ejercicio ya que me permite trabajar con diferentes motores de bases de datos. Por lo que una vez controlado y entendido me será más útil que utilizar la función *mysqli_connect ()*

He tratado de usar `$_SESSION` para almacenar y acceder a datos de sesión. Me he encontrado con el error de que el contenido de dicha variable no se mantenía de una página a otra o yo no he sabido leerla correctamente. Esta variable se almacena en el servidor, en lugar de en el navegador del usuario, y se asocian con un identificador único de sesión que se almacena en una cookie en el navegador del usuario. Cada vez que el usuario realiza una solicitud HTTP al servidor, el identificador de sesión se envía junto con la solicitud, lo que permite al servidor recuperar los datos de sesión asociados con ese usuario.