28 DE FEBRERO DE 2023

HITO INDIVIDUAL - PROGRAMACIÓN

ALVARO BARRENA REVILLA

INDICE

CUESTIÓN 1

- *¿Qué es un algoritmo?*

Un algoritmo es un conjunto de instrucciones que se utilizan para realizar una tarea específica.

En el caso de este proyecto, el algoritmo seguirá los siguientes pasos:

1. *Recopilar información del usuario:* solicitar al usuario la información necesaria para crear la entrada en el blog, como el título, el contenido, la fecha de publicación, el correo electrónico del autor y la imagen.

2. *Validar la información del usuario:* verificar que la información proporcionada por el usuario sea válida y esté en el formato correcto. Si la información no es válida, informar al usuario del problema y pedirle que la corrija.

3. *Crear y conectar la base de datos:* establecer una conexión con la base de datos para almacenar la entrada del blog.

4. *Insertar la entrada en la base de datos:* una vez que se ha establecido la conexión, insertar la entrada del blog en la base de datos. Esta entrada debe incluir toda la información proporcionada por el usuario, así como cualquier otra información necesaria para el funcionamiento del sistema, como un ID único para la entrada.

5. *Confirmar al usuario:* una vez que la entrada se ha insertado correctamente en la base de datos, confirmar al usuario que se ha publicado correctamente. Enviar un correo electrónico al autor de la entrada para informarle de que su entrada se ha publicado correctamente.

- *Caso de uso 1: Publicar una entrada en el blog*

Actores: Sistema y usuario

Precondiciones:

- El usuario inicia sesión en la aplicación.
- El usuario ingresa en el formulario de publicación de la entrada del blog.

Flujo de básico:

1. El usuario introduce la información requerida en el formulario de publicación de entrada del blog, incluyendo título, contenido, fecha de publicación e imagen.
2. El sistema verifica que se hayan completado todos los campos requeridos y valida la información ingresada.
3. Si la información ingresada es válida, el sistema almacena la entrada en la base de datos y muestra un mensaje de confirmación al usuario.
4. Si la información ingresada no es válida, el sistema muestra un mensaje de error indicando los campos que deben ser corregidos.

Flujos alternativos:

1. Si el usuario no ha iniciado sesión en la aplicación, el sistema redirige al usuario a la página de inicio de sesión y luego vuelve al formulario de publicación de entrada del blog.
2. Si el usuario no ha completado todos los campos requeridos, el sistema muestra un mensaje de error indicando los campos que deben ser completados.
3. Si la información ingresada no es válida, el sistema muestra un mensaje de error indicando el motivo de la invalidez (por ejemplo, si la fecha de publicación es anterior a la fecha actual).

- *Caso de uso 2: Recuperar contraseña*

Actores: Usuario y sistema

Precondiciones: El usuario debe haber olvidado su contraseña y estar en la página de inicio de sesión.

Flujo básico:

1. El usuario hace clic en el enlace de "¿He olvidado la contraseña?" en la página de inicio de sesión.
2. El sistema muestra un formulario para que el usuario ingrese su dirección de correo electrónico asociada con su cuenta.
3. El usuario ingresa su dirección de correo electrónico y envía el formulario.
4. El sistema valida la dirección de correo electrónico y busca en la base de datos si existe una cuenta asociada a ella.
5. Si la dirección de correo electrónico no es válida o no se encuentra una cuenta asociada, el sistema muestra un mensaje de error y devuelve al usuario al formulario anterior.
6. Si la dirección de correo electrónico es válida y se encuentra una cuenta asociada, el sistema genera un token de seguridad y envía un correo electrónico al usuario con un enlace para restablecer su contraseña.
7. El usuario hace clic en el enlace en el correo electrónico y se le lleva a una página donde puede ingresar una nueva contraseña.
8. El sistema valida el token de seguridad y muestra un formulario para que el usuario ingrese y confirme su nueva contraseña.
9. El usuario ingresa su nueva contraseña y la confirma.
10. El sistema valida la nueva contraseña y actualiza la base de datos con la nueva información de la cuenta del usuario.
11. El sistema redirige al usuario a la página de inicio de sesión con un mensaje de confirmación de que su contraseña ha sido actualizada.

Flujos alternativos:

- Si el token de seguridad ha caducado:
    - El sistema muestra un mensaje de error y ofrece al usuario la posibilidad de volver a intentarlo.
- Si la nueva contraseña no cumple con los requisitos de seguridad:
    - El sistema muestra un mensaje de error y ofrece al usuario la posibilidad de ingresar una nueva contraseña que cumpla con los requisitos.

- *Pasar del algoritmo a la implementación de la aplicación*

Para pasar del algoritmo a la implementación de la aplicación, es necesario seguir un proceso que involucra tanto la parte visual como la programática.

Pasos a seguir:

1. *Elección del lenguaje de programación:* el primer paso es seleccionar el lenguaje de programación que se va a utilizar para implementar el algoritmo. Es importante elegir un lenguaje que sea adecuado para la tarea, en este caso es PHP.

2. *Escribir el código:* con el lenguaje de programación elegido, se empieza a escribir el código que implementa el algoritmo siguiendo las estructuras y la lógica del algoritmo para garantizar que la implementación sea correcta.

3. *Integrar con la parte visual:* una vez que se ha implementado la lógica del algoritmo, se debe integrar con la parte visual de la aplicación. Esto implica diseñar y crear la interfaz de usuario (UI) que permitirá a los usuarios interactuar con la aplicación.

4. *Depuración y prueba:* después de implementar la lógica del algoritmo e integrar la parte visual, se depura el código para corregir errores y asegurar el funcionamiento de la aplicación.

5. *Mantenimiento:* una vez que la aplicación ha sido implementada y puesta en marcha, se realiza el mantenimiento. Esto implica la corrección de errores, actualización de funcionalidades y optimización del código para garantizar que la aplicación continúe funcionando correctamente y satisfaciendo las necesidades de los usuarios.

- ## *Diferencias entre algoritmo, caso de uso y desarrollo de código en la implementación*

El ***algoritmo*** es una secuencia ordenada de pasos que describe cómo realizar una tarea específica. En el contexto del desarrollo de software, un algoritmo se utiliza para diseñar la lógica de un programa antes de que se escriba el código.

Un ***caso de uso*** describe cómo un usuario interactúa con un sistema para realizar una tarea específica. Es una descripción detallada de cómo se utiliza un sistema para llevar a cabo una función en particular. Los casos de uso se utilizan para comprender y diseñar los requisitos de un sistema y son una herramienta valiosa para la comunicación entre los desarrolladores y los clientes.

El ***desarrollo de código*** es el proceso de escribir código para implementar las funcionalidades descritas por los algoritmos y casos de uso. El código es la expresión final del diseño lógico y se escribe en un lenguaje de programación específico.

- ## *Fases en el proceso de diseño de la aplicación*

A continuación se explican las posibles fases para el proceso de diseño de la aplicación:

- *Análisis de requisitos:* En esta fase, se analiza la tarea propuesta y se identifican los requisitos funcionales y no funcionales de la aplicación.

- *Diseño del sistema:* En esta fase, se diseña la arquitectura del programa y se elaboran los diagramas de flujo, las pantallas y los formularios necesarios para implementar la tarea.

- *Desarrollo de código:* En esta fase, se escribe el código que implementa la funcionalidad de la aplicación. Se utilizan los algoritmos y casos de uso para guiar el desarrollo del código.

- *Pruebas y corrección de errores:* En esta fase, se prueban todas las funcionalidades de la aplicación y se corrigen los errores que se encuentran.

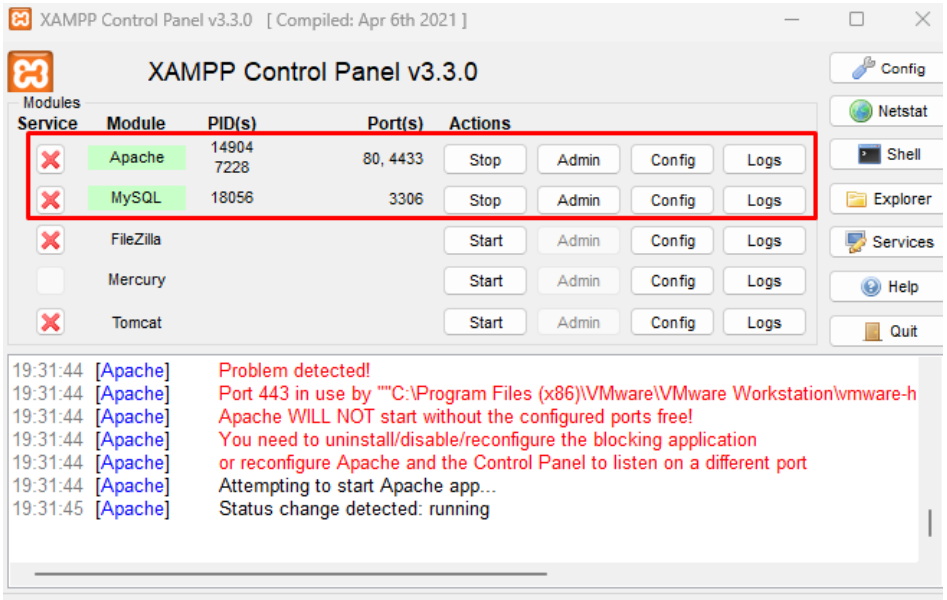
- *Implementación y puesta en marcha:* En esta fase, se implementa la aplicación en el entorno de producción y se pone en marcha para su uso.

CUESTIÓN 2

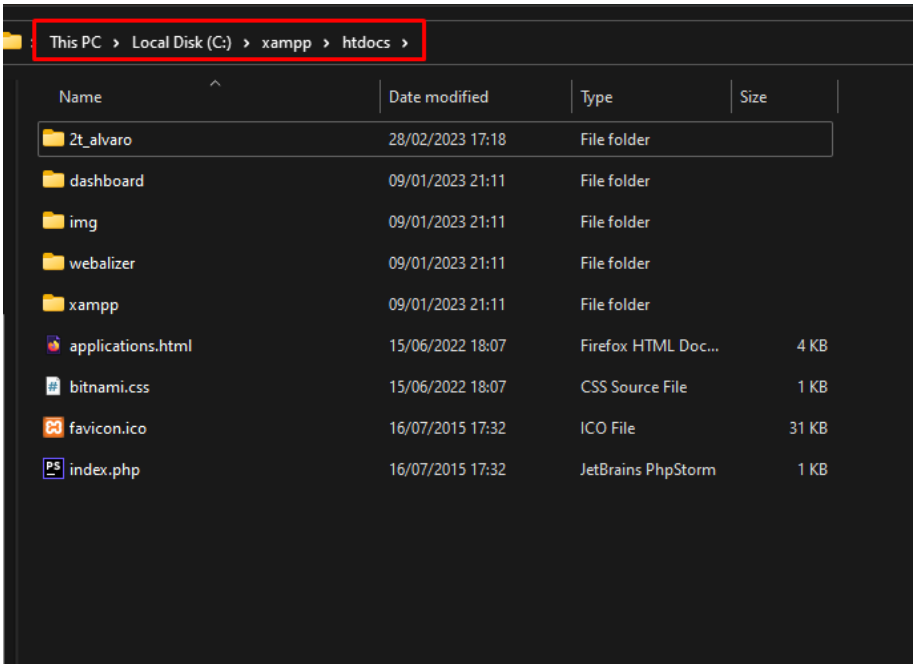
– *Ítem 1*

Antes de empezar con el código y la página web, hay que comprobar varias cosas:

1. Tener XAMPP instalado y configurado. Es un paquete de software libre y gratuito que se utiliza para crear un servidor web en un entorno de desarrollo local permite crear y probar aplicaciones web en el equipo local antes de subirlas a un servidor web de internet. Se tiene que tener activado Apache y MySQL.



2. Una vez activados los módulos de Apache y MySQL, procedemos a entrar en la carpeta de XAMPP que va a permitir desplegar el sitio web en local.



Para hacer este ítem he decidido usar el IDE PhpStorm, ya que a la hora de programar me parece mucho mejor que Visual Studio Code por la funcionalidad más avanzada que tiene

- *Ítem 2*

Para este segundo ítem he empezado por hacer la estructura general de HTML y a partir de ahí he metido Bootstrap para darle un mejor estilo, un menú adaptado a Bootstrap y la explicación de las diferencias entre lenguajes de programación orientada a objetos, a eventos y lenguajes procedimentales. No he usado CSS ya que todos los estilos que he dado han sido con Bootstrap.

```html
<!DOCTYPE html>
<html lang="es">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
        <meta http-equiv="X-UA-Compatible" content="ie=edge">
        <link rel="stylesheet" type="text/css" href="bootstrap/css/bootstrap.min.css">
        <link rel="stylesheet" type="text/css" href="css/styles.css">
        <title>Hito programación</title>
    </head>
    <body>
        <div class="container">
            <div class="row">
                <header id="header" class="col-12 p-3 text-center">
                    <h1 class="text-uppercase fw-bold">Diferencias lenguajes de programación</h1>
                </header>
            </div>

            <nav id="menu" class="col-12 p-2">
                <ul class="nav justify-content-center">
                    <li class="nav-item">
                        <a class="nav-link active" href="index.php">Inicio</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="publica/login.php">Iniciar sesión</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="privada/add.html">Formulario</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="publica/verPosts.php">Ver publicaciones</a>
                    </li>
                    <li>
                        <a class="nav-link" href="privada/update.html">Actualizar publicaciones</a>
                    </li>
                    <li>
                        <a class="nav-link" href="privada/update.php">Eliminar publicaciones</a>
                    </li>
                </ul>
            </nav>
```

```
        <div class="row">
            <div class="col-12">
                <h2>Programación orientada a objetos (POO)</h2>
                <p class="text-justify">
                    En la programación orientada a objetos, los programas están diseñados en torno a objetos, que son entidades que contienen datos y métodos. Los objetos pueden interactuar
                     entre sí mediante la llamada a métodos y el intercambio de mensajes. La POO permite la encapsulación de datos, lo que significa que los datos están protegidos de
                    accesos no autorizados y solo pueden ser manipulados mediante los métodos del objeto. Los lenguajes de programación orientados a objetos incluyen Java, Python, C++,
                    Ruby, y muchos más.
                </p><hr>
            </div>
            <div class="col-12">
                <h2>Programación orientada a eventos (POE)</h2>
                <p class="text-justify">
                    En la programación orientada a eventos, el programa responde a eventos externos, como la entrada del usuario o la llegada de datos a través de la red. El programa se
                    divide en pequeñas secciones que se activan cuando se produce un evento. Los lenguajes de programación orientados a eventos incluyen JavaScript, Visual Basic,
                    ActionScript, y otros.
                </p><hr>
            </div>
            <div class="col-12">
                <h2>Programación procedimental</h2>
                <p class="text-justify">
                    En la programación procedimental, el programa está compuesto por una secuencia de instrucciones que se ejecutan una tras otra. La programación procedimental se basa en
                    la idea de un procedimiento o función, que es una sección de código que realiza una tarea específica. Los lenguajes de programación procedimentales incluyen C, Pascal,
                    Fortran y BASIC.
                </p><hr>
            </div>
        </div>
    </div>
    <?php
        require_once 'cookie.php'
    ?>
    <!--Cargar el archivo jquery-->
    <script type="text/javascript" src="jquery/jquery-3.6.3.min.js"></script>
    <!--Cargar el archivo javascript de bootstrap-->
    <script type="text/javascript" src="bootstrap/js/bootstrap.min.js"></script>
</body>
</html>
```

Por último, como se pedía en este ítem, he creado una cookie en la que aparece la IP desde el equipo que se está accediendo, la fecha y la hora. Esta cookie tiene una duración de 1 hora (3600 segundos).

```
<?php
$ip = $_SERVER['REMOTE_ADDR']; //Almacena la IP del equipo que está accediendo
$fecha = date( format: "d/m/Y H:i:s"); //Almacena la fecha y hora de acceso

//Crear la cookie
setcookie("ip", $ip, time() + 3600); //La cookie se crea con el nombre "ip" y el valor de la variable $ip y con una duración de 1 hora.
setcookie("fecha", $fecha, time() + 3600); //La cookie se crea con el nombre "fecha" y el valor de la variable $fecha y con una duración de 1 hora.

//Comprobar si la cookie existe
if (isset($_COOKIE['ip'])) {
    echo "<p class='text-center fw-bold'>Estás accediendo desde la IP: ".$_COOKIE['ip']. " y la fecha de acceso es: ".$_COOKIE['fecha']."</p>";
} else {
    echo "<p class='text-center fw-bold'>No se ha creado la cookie, recarga la página</p>";
}
```

Este es el resultado final de este ítem:

DIFERENCIAS LENGUAJES DE PROGRAMACIÓN

Inicio Iniciar sesión Formulario Ver publicaciones

Actualizar publicaciones Eliminar publicaciones

Programación orientada a objetos (POO)

En la programación orientada a objetos, los programas están diseñados en torno a objetos, que son entidades que contienen datos y métodos. Los objetos pueden interactuar entre sí mediante la llamada a métodos y el intercambio de mensajes. La POO permite la encapsulación de datos, lo que significa que los datos están protegidos de accesos no autorizados y solo pueden ser manipulados mediante los métodos del objeto. Los lenguajes de programación orientados a objetos incluyen Java, Python, C++, Ruby, y muchos más.

Programación orientada a eventos (POE)

En la programación orientada a eventos, el programa responde a eventos externos, como la entrada del usuario o la llegada de datos a través de la red. El programa se divide en pequeñas secciones que se activan cuando se produce un evento. Los lenguajes de programación orientados a eventos incluyen JavaScript, Visual Basic, ActionScript, y otros.

Programación procedimental

En la programación procedimental, el programa está compuesto por una secuencia de instrucciones que se ejecutan una tras otra. La programación procedimental se basa en la idea de un procedimiento o función, que es una sección de código que realiza una tarea específica. Los lenguajes de programación procedimentales incluyen C, Pascal, Fortran y BASIC.

Estás accediendo desde la IP: 192.168.1.44 y la fecha de acceso es: 28/02/2023 20:10:01

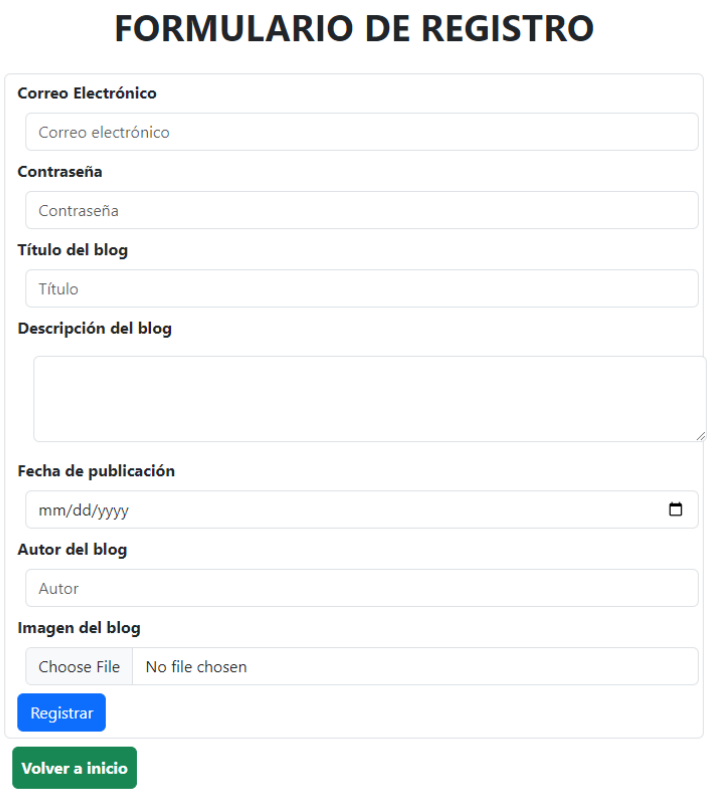
– *Ítem 3*

Para este tercer ítem, primero hay que crear la base de datos desde phpmyadmin con la siguiente estructura:

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
☐	1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
☐	2	email	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
☐	3	password	varchar(200)	utf8mb4_general_ci		No	None			Change Drop More
☐	4	titulo	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
☐	5	contenido	varchar(200)	utf8mb4_general_ci		No	None			Change Drop More
☐	6	fecha	date			No	None			Change Drop More
☐	7	autor	varchar(80)	utf8mb4_general_ci		No	None			Change Drop More

En mi caso no he puesto la imagen, ya que cada vez que intentaba subirla a la base de datos desde el formulario no se ejecutaba la sentencia y no conseguí solucionarlo.

Lo segundo que hice fue crear el formulario, con sus estilos de Bootstrap correspondientes para poder hacer la sentencia correspondiente para insertar los datos como explicaré a continuación.



Los datos del formulario son los mismos que los que se piden en este ítem.

```
<body>
    <div class="container">
        <h1 class="text-uppercase text-center fw-bold p-3">Formulario de registro</h1>
        <form class="form-control" action="add.php" method="post">
            <label for="email" class="fw-bold">Correo electrónico</label>
            <input type="email" class="form-control m-2" id="email" name="email"
            placeholder="Correo electrónico" required>

            <label for="password" class="fw-bold">Contraseña</label>
            <input type="password" class="form-control m-2" id="password" name="password"
             placeholder="Contraseña" required>

            <label for="title" class="fw-bold">Título del blog</label>
            <input type="text" class="form-control m-2" id="title" name="title"
             placeholder="Título">

            <label for="content" class="fw-bold">Descripción del blog</label>
            <textarea class="form-control m-3" id="content" name="content" rows="3"></textarea>

            <label for="fecha" class="fw-bold">Fecha de publicación</label>
            <input type="date" class="form-control m-2" id="fecha" name="fecha">

            <label for="autor" class="fw-bold">Autor del blog</label>
            <input type="text" class="form-control m-2" id="autor" name="autor"
             placeholder="Autor">

            <label for="imagen" class="fw-bold">Imagen del blog</label>
            <input type="file" class="form-control m-2" id="imagen" name="imagen">

            <button type="submit" class="btn btn-primary">Registrar</button>
        </form>
        <a href='../index.php' class='btn btn-success fw-bold p-2 m-2'>Volver a inicio</a>
    </div>
```

A continuación se muestra la parte de PHP donde se insertan los registros:

```php
<?php
session_start();
// Recoger los datos del formulario
$correo = $_POST['email'];
$password = $_POST['password'];
$titulo = $_POST['title'];
$contenido = $_POST['content'];
$fecha = $_POST['fecha'];
$autor = $_POST['autor'];
$imagen = $_POST['imagen'];
```

Se inicia la sesión y se recogen los datos previamente datos en el formulario.

```php
//Conexión a la base de datos
$conn = new PDO( dsn: 'mysql:host=localhost;port=3306;dbname=hito', username: 'root', password: '');
//Preparar la consulta
$consulta = "INSERT INTO `registro` (`id`, `email`, `password`, `titulo`, `contenido`, `fecha`,
`autor`) VALUES (NULL, '".$correo."', MD5('".$password."'), '".$titulo."', '".$contenido."',
'".$fecha."', '".$autor."');";
//Ejecutar la consulta
$resultado = $conn->query($consulta);
```

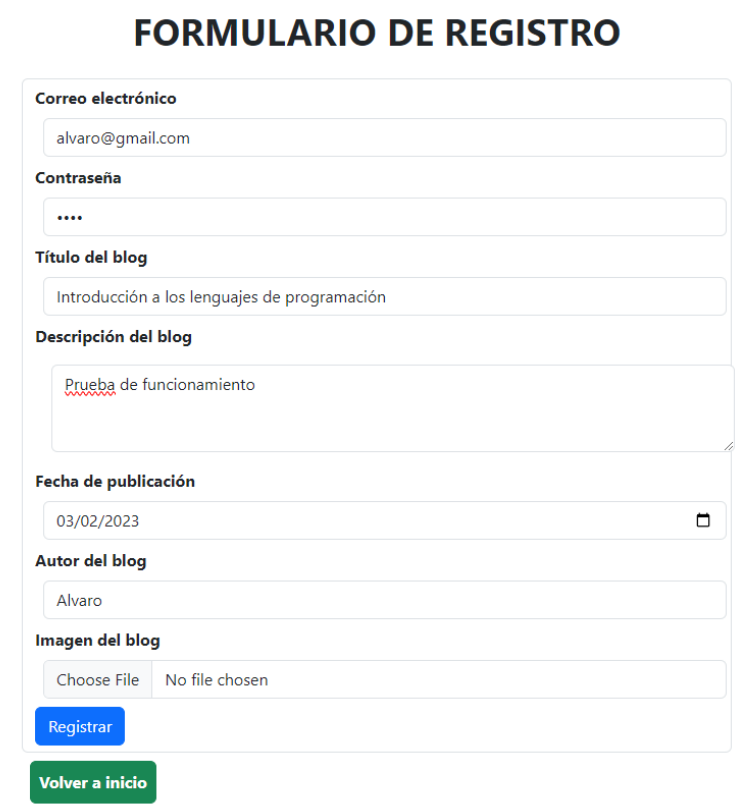
Se conecta a la base de datos desde una PDO, en mi caso he elegido una PDO, ya que permite cambiar la base de datos más adelante por otra, por el contrario si lo hubiera hecho con mysqli o postresql, si en un futuro se quisiera migrar a otra base de datos habría que cambiar todo el proceso de conexión.

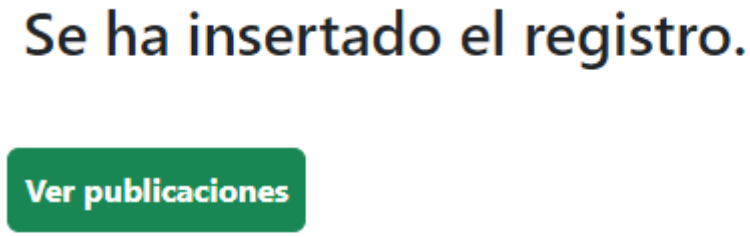
Posteriormente se crea la variable consulta en la que irá la inserción de los datos del formulario y después se ejecuta la consulta.

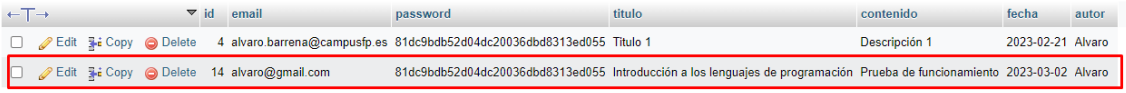
```php
19  //Comprobar si se ha insertado el registro
20  if ($resultado) {
21      echo "<link rel='stylesheet' type='text/css' href='../bootstrap/css/bootstrap.min.css'>";
22      echo "<h2 class='p-2 m-3'>Se ha insertado el registro.</h2>";
23      echo "<a href='../publica/verPosts.php' class='btn btn-success fw-bold p-2 m-3'>Ver
        publicaciones</a>";
24
25  } else {
26      echo "<h2>No se ha insertado el registro</h2>";
27  }
```

Por último, para validar si lo que he hecho funciona he creado una validación mediante un if/else para que me muestre un feedback acorde con el resultado.

A continuación muestro un ejemplo del resultado:

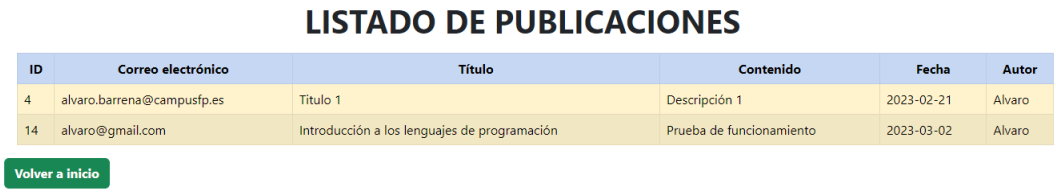






	id	email	password	titulo	contenido	fecha	autor
Edit Copy Delete	4	alvaro.barrena@campusfp.es	81dc9bdb52d04dc20036dbd8313ed055	Titulo 1	Descripción 1	2023-02-21	Alvaro
Edit Copy Delete	14	alvaro@gmail.com	81dc9bdb52d04dc20036dbd8313ed055	Introducción a los lenguajes de programación	Prueba de funcionamiento	2023-03-02	Alvaro

Una vez que se ha insertado el registro en la base de datos he creado otra página en la que muestro una tabla con estilos en Bootstrap donde se muestran los datos que están registrados en la base de datos. En el ítem 4 explicaré el proceso de la visualización de los datos de esta tabla.



LISTADO DE PUBLICACIONES

ID	Correo electrónico	Título	Contenido	Fecha	Autor
4	alvaro.barrena@campusfp.es	Titulo 1	Descripción 1	2023-02-21	Alvaro
14	alvaro@gmail.com	Introducción a los lenguajes de programación	Prueba de funcionamiento	2023-03-02	Alvaro

Volver a inicio

- *Ítem 4*

Primero he creado la conexión mediante una PDO y he creado la variable consulta con su correspondiente sentencia para mostrar la lista de las publicaciones. Por último he ejecutado la consulta.

```php
<?php
$conn = new PDO( dsn: 'mysql:host=localhost;dbname=hito', username: 'root', password: '');
$consulta = "SELECT * FROM REGISTRO";
$resultado = $conn->query($consulta);
```

Lo segundo que he hecho ha sido crear la cabecera de la tabla para que muestre los datos que se van a ver. Los estilos han sido dados también con Bootstrap.

```php
echo "<link rel='stylesheet' type='text/css' href='../bootstrap/css/bootstrap.min.css'>";
echo "<h1 class='text-uppercase fw-bold text-center m-3'>Listado de publicaciones</h1>";
echo "<div class='container'>";
echo "<table class='table table-striped table-bordered table-hover table-warning m-3'>";
echo "<tr class='table-primary'>";
echo "<th class='text-center'>ID</th>";
echo "<th class='text-center'>Correo electrónico</th>";
echo "<th class='text-center'>Titulo</th>";
echo "<th class='text-center'>Contenido</th>";
echo "<th class='text-center'>Fecha</th>";
echo "<th class='text-center'>Autor</th>";
echo "</tr>";
```

Por último, he creado un bucle while para que recorra y muestre todos los registros en forma de tabla. También como funcionalidad extra he creado un botón para que redirija a la página de inicio.

```php
while ($registro = $resultado->fetch()) {
    echo "<tr>";
    echo "<td class='fw-bold'>" . $registro['id'] . "</td>";
    echo "<td>" . $registro['email'] . "</td>";
    echo "<td>" . $registro['titulo'] . "</td>";
    echo "<td>" . $registro['contenido'] . "</td>";
    echo "<td>" . $registro['fecha'] . "</td>";
    echo "<td>" . $registro['autor'] . "</td>";
    echo "</tr>";
}
echo "</table>";
echo "<a href='../index.php' class='btn btn-success fw-bold'>Volver a inicio</a>"
echo "</div>";
```

- *Ítem 5*

En este ítem, principalmente están las funciones de eliminar y actualizar.

Eliminar

En esta parte, primero he creado un formulario con la etiqueta del correo, para que a la hora de ejecutar la sentencia DELETE, busque el correo en la base de datos y lo elimine.

```html
<div class="container">
    <h1 class="text-uppercase text-center fw-bold p-3">Eliminar
     registro</h1>
    <form class="form-control" action="delete.php" method="post">
        <label for="email" class="fw-bold">Correo
         electrónico</label>
        <input type="email" class="form-control m-2" id="email"
         name="email" placeholder="Correo electrónico" required>
        <button type="submit" class="btn btn-primary">Eliminar
         registro</button>
    </form>
    <a href='../index.php' class='btn btn-success fw-bold p-2
     m-2'>Volver a inicio</a>
</div>
```

```php
<?php
$correo = $_POST['email'];

$conn = new PDO( dsn: 'mysql:host=localhost;port=3306;dbname=hito', username: 'root', password: '');
//Preparar la consulta
$consulta = "DELETE FROM `registro` WHERE `registro`.`email` = '".$correo."';";
//Ejecutar la consulta
$resultado = $conn->query($consulta);
header( header: "Location: ../publica/verPosts.php");
```

Actualizar

En esta parte, he creado un formulario para actualizar el título, el contenido y la fecha de publicación del post dado el correo electrónico.

```php
<div class="container">
    <h1 class="text-uppercase text-center fw-bold m-3">Actualizar registro</h1>
    <form action="update.php" method="post" class="form-control">
        <label for="email">Correo electrónico</label>
        <input type="email" name="email" id="email" placeholder="Correo electrónico..."
         class="form-control">

        <label for="titulo">Título</label>
        <input type="text" name="titulo" id="titulo" placeholder="Título..."
        class="form-control">

        <label for="contenido">Contenido</label>
        <textarea name="contenido" id="contenido" placeholder="Contenido..." rows="5"
         class="form-control"></textarea>

        <label for="fecha">Fecha de publicación</label>
        <input type="date" name="fecha" id="fecha" placeholder="Fecha..." class="form-control">


        <input type="submit" value="Actualizar registro" class="btn btn-primary m-2">
    </form>
</div>
```

```php
<?php
session_start();
$correo = $_POST['email'];
$titulo = $_POST['titulo'];
$contenido = $_POST['contenido'];
$fecha = $_POST['fecha'];

//Conexión a la base de datos
$conn = new PDO( dsn: 'mysql:host=localhost;port=3306;dbname=hito', username: 'root', password: '');

//Preparar la consulta
$consulta = "UPDATE `registro` SET `titulo` = '".$titulo."', `contenido` = '".$contenido."', `fecha` =
 '".$fecha."' WHERE `registro`.`email` = '".$correo."';";
//Ejecutar la consulta
$resultado = $conn->query($consulta);

header( header: "Location: ../publica/verPosts.php");
```

En la parte del login he tenido muchos problemas con las sesiones porque no conseguía loguearme, por lo que esta parte no la he podido cumplir, aunque dejo el proceso del código.

Formulario de login

```
<div class="container">
    <h1 class="text-uppercase text-center fw-bold p-2">Inicio de sesión</h1>
    <form class="form-control">
        <label for="correo">Correo electrónico</label>
        <input type="email" name="correo" id="correo" class="form-control m-2"
         placeholder="Introduce tu correo electrónico">
        <label for="passwd">Contraseña</label>
        <input type="password" name="passwd" id="passwd" class="form-control m-2"
         placeholder="Introduce tu contraseña">
        <input type="submit" value="Iniciar sesión" class="btn btn-primary m-2">
    </form>
</div>
```

Código PHP en el que he tenido problemas

```
<?php
    $login_user = "admin@gmail.com";
    $login_pass = "admin";
    //Conexión a la base de datos
    $conn = new PDO( dsn: 'mysql:host=localhost;port=3306;dbname=hito', username: 'root',
      password: '');

    $consulta = $conn->query ( statement: "SELECT * FROM registro ");

    while ($usuario = $consulta->fetch()) {
        if ($usuario['email'] == $login_user && $usuario['password'] == $login_pass) {
            echo "Login correcto";
            session_start();
            $_SESSION['token'] = session_id();
             echo $_SESSION['token'];
        } else {
            echo "Login incorrecto";
        }
    }

?>
<p class="text-center">¿No tienes cuenta? <a href="../privada/add.html">Registrate</a></p>
<a href="../index.php" class="btn btn-success fw-bold m-2 p-2">Volver a inicio</a>
```

CUESTIÓN 3

Para este proyecto web he utilizado las tecnologías de HTML, CSS, Bootstrap y el lenguaje de programación PHP. He usado un entorno de desarrollo, PhpStorm, ya que con Visual Studio no tenía tantas funcionalidades y no me ayudaba a ver los errores que tenía como con PhpStorm.

Los principales problemas que he tenido con este sitio web han sido la organización de las carpetas y la navegación entre ellas y también el mayor problema ha sido el login, ya que no me enteraba al hacer las sesiones y no sabía bien cómo implementarlas para su correcto funcionamiento.