



PROYECTO EXAMEN

Pablo Arcas Roldan

2º Desarrollo Web Entorno Servidor

INDICE

REQUISITOS PARA PRÁCTICA EXAMEN	3
QUE INCORPORA LA APLICACIÓN RESPECTO A LOS REQUISITOS	3
DESARROLLO DEL PROYECTO	4
BASE DE DATOS – CLUB MOTOR.....	4
APLICACIÓN – CLUB MOTOR	5
FUNCIONALIDAD APLICACIÓN	6
conexionDB.php	6
contraseñaDB.php.....	6
login.php	7
registro_usuario.php	7
Validaciones lado cliente y lado servidor en ambos formularios.	8
principal.php	9
mostrar_principal.php	10
Ver Vehiculos:	10
Ver Socios:	13
Logout.php	14
modificar.php, insertar.php y borrar.php.....	15
controlVisitas.php	15
CONCLUSIÓN	15

REQUISITOS PARA PRÁCTICA EXAMEN

- Tiene que ser una aplicación que se conecte a una base datos, una tabla hacer CRUD sobre ella.
- Formularios 3-4, listar, insertar, actualizar borrar.
- Uso de arrays asociativos.
- Uso de cookies y sesiones.
- Uso de get y post.
- Entrada al sistema.
- Validaciones cliente y servidor.
- Mejoras
 - Mas de una tabla, 1-n Eje: un cliente y sus facturas, verlo en la aplicación.
 - Mas formularios.
 - MVC.

QUE INCORPORA LA APLICACIÓN RESPECTO A LOS REQUISITOS

- 1 Base de Datos (club_motor) y 3 tablas (socios, vehiculos y usuarios) con relación 1-N entre socios – vehiculos.
- Hacemos CRUD sobre la BD, leer, insertar, actualizar y borrar en diversas tablas.
- 6 Formularios ->POST y 2 Formularios ->GET.
- En PHP los arrays son de tipo asociativo, por lo que usamos en casi todas las páginas del proyecto almacenar los resultados de las consultas.
- Login de entrada al sistema y registros de nuevo usuarios.
- Validaciones en lado cliente y validaciones en lado servidor.
- Cookies para el control de visitas a determinadas páginas y 2 cookies que guardan el ID de usuario y el nombre de usuario una vez logueado durante 1 día.
- Sesiones durante todo el proyecto. Controlamos que el usuario se ha logueado creando `$_SESSION['id']`, `$_SESSION['usuario']` y `$_SESSION['contraseña']` y así poder entrar en la aplicación. También creamos una `$_SESSION['insertar']` en la que controlamos que ha seleccionado el usuario en la pagina principal para su posterior utilización. Una vez el usuario pulsa cerrar sesión, todas las sesiones activas se destruyen.

DESARROLLO DEL PROYECTO

El proyecto consta de una aplicación que nos permite entrar a un garaje online y mostrarnos su exposición de vehículos y sus socios. Esta aplicación se conecta a una base de datos llamada **club_motor** la cual nos permite hacer CRUD entre ambas. Las herramientas de trabajo utilizadas para este proyecto son para la BD -> MySQL Workbench y para la configuración de la aplicación VS Code. La conexión entre ambas la hemos realizado a través de XAMPP en local.

BASE DE DATOS – CLUB MOTOR

Actualmente compuesta por 3 tablas que son:

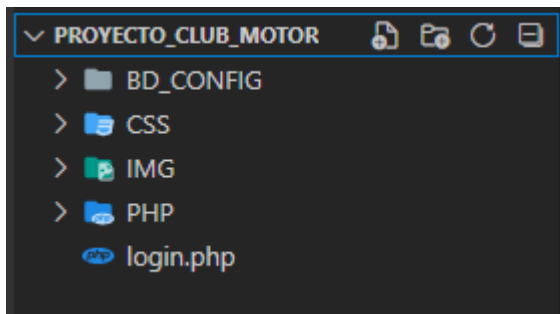
- usuarios: Compuesta por los campos usuario_id, username y contraseña. Esta tabla es la encargada de gestionar tanto el login como el registro de un nuevo usuario. La aplicación cuenta con 4 usuarios registrados con las siguientes credenciales para la realización de pruebas que son:
 - username: pablo@prueba.com – contraseña: 1234ñlkj
 - username: jorge@prueba.com – contraseña: ñlkj1234
 - username: enrique@prueba.com – contraseña: asdf1234
 - username: irene@prueba.es – contraseña: sanclemente7
- socios: Compuesta por los campos socio_id (Primary Key), nombre, apellidos, ciudad. En esta tabla se encuentran los socios/propietarios de los vehículos de la exposición. Actualmente el club cuenta con 5 socios registrados.
- vehículos: Compuesta por los campos vehículo_id (Primary Key), marca, modelo, precio, imagen y socio_id (Foreign Key). En esta tabla se encuentran los vehículos disponibles en la exposición. Actualmente el club cuenta con 14 vehículos registrados.

La relación entre las tablas socios y vehículos es 1-N siendo el campo socio_id el nexo en la relación entre ambas.

APLICACIÓN – CLUB MOTOR

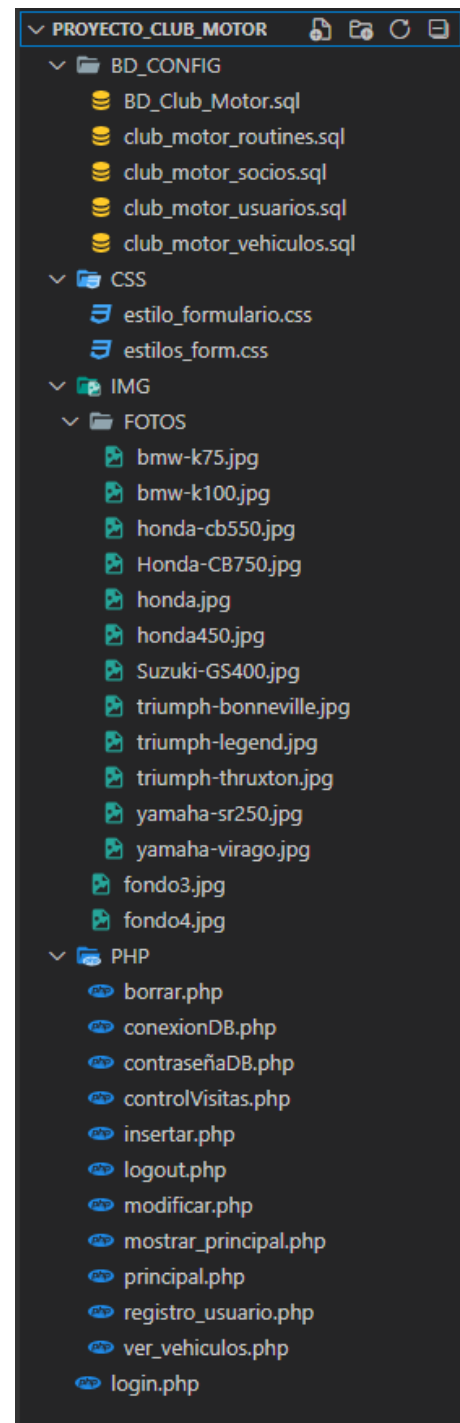
La aplicación cuenta con la siguiente estructura:

DIRECTORIO DE CARPETAS



- **BD_CONFIG:** Contiene los archivos .sql de creación de la BD y sus tablas. También contiene un archivo copia de la BD por si ocurriese algún problema importando el archivo. Hay dos posibilidades de importar los archivos: 1- En un solo documento. 2-Cada tabla en documento aparte.
- **CSS:** Contiene los archivos encargados de dar los estilos a la aplicación.
- **IMG:** Contiene las fotos tanto de la BD como las fotos que utilizamos en css.
- **PHP:** Contiene todos los archivos con la gestión y lógica de la aplicación.

DIRECTORIO DE FICHEROS



FUNCIONALIDAD APLICACIÓN

Vamos a explicar paso a paso la funcionalidad de la aplicación. Cabe decir que en cada archivo correspondiente hemos ido explicando la lógica y funcionalidad del código en detalle, esto es una breve explicación y aconsejamos interactuar con la aplicación para visualizar todo más en detalle.

conexionDB.php

En este archivo hemos creado la función conectarDB() la cual utilizaremos durante toda la aplicación para conectarnos a nuestra BD. La conexión utilizada para este proyecto a sido mediante PDO ya que nos permite el manejo de excepciones. Las credenciales para la conexión a la BD:

- dsn: "mysql:host=localhost;dbname=club_motor"
- usuario: "root"
- contraseña: La configuración de nuestra BD no requiere contraseña por lo que dejamos en blanco.

Cuando un usuario vaya a probar la aplicación deberá cambiar las credenciales y poner las suyas.

```
1 <?php
2 // Hemos creado una funcion para conectar a la base de datos en un fichero que iremos incluyendo donde lo necesitemos
3 function conectarDB(){
4     //Creamos las variables de conexion con la dsn, el usuario y la contraseña.
5     $dsn = "mysql:host=localhost;dbname=club_motor";
6     $usuario = "root";
7     $contrasena = "";
8     //Mediante el bloque try-catch vamos a manejar la conexion que en este caso la haremos mediante PDO y los posibles errores.
9     try{
10        //Creamos el objeto para realizar la conexion.
11        $conexion = new PDO($dsn, $usuario, $contrasena);
12        //Con esta linea estamos diciendo que si hay algun problema que salte la excepcion.
13        $conexion->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
14        return $conexion;
15    }
16 } catch (PDOException $e){ //Hemos creado un objeto tipo exception para asi poder trabajar con el la excepcion, si saltara, mandariamos el mensaje de error.
17     die("Error al conectar con la base de datos: " . $e->getMessage());
18 }
19 }
20
21 ?>
```

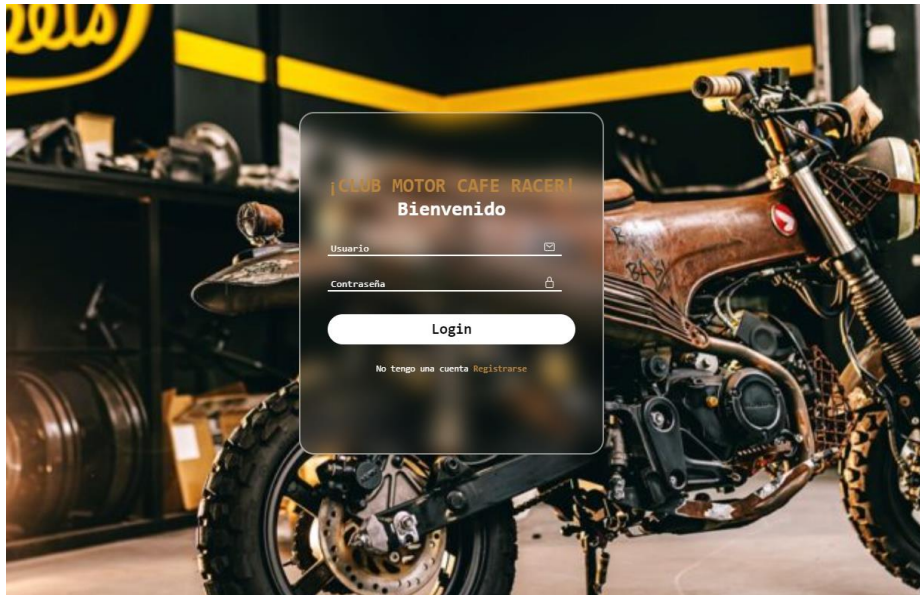
contraseñaDB.php

Este archivo contiene la función para encriptar y verificar las contraseñas de acceso a la aplicación.

```
1 <?php
2 /* Vamos a crear una funcion para encriptar las contraseñas y proporcionar mas seguridad al proyecto, esto implica que tendremos
3    que ir dando de alta los usuarios en la base de datos directamente desde el navegador. */
4
5 function hashContrasenia($contrasenia){
6     return password_hash($contrasenia, PASSWORD_BCRYPT);
7 }
8 function verificarContrasenias ($contrasenia, $contraseniaBD){
9     return password_verify($contrasenia, $contraseniaBD);
10 }
11 ?>
```

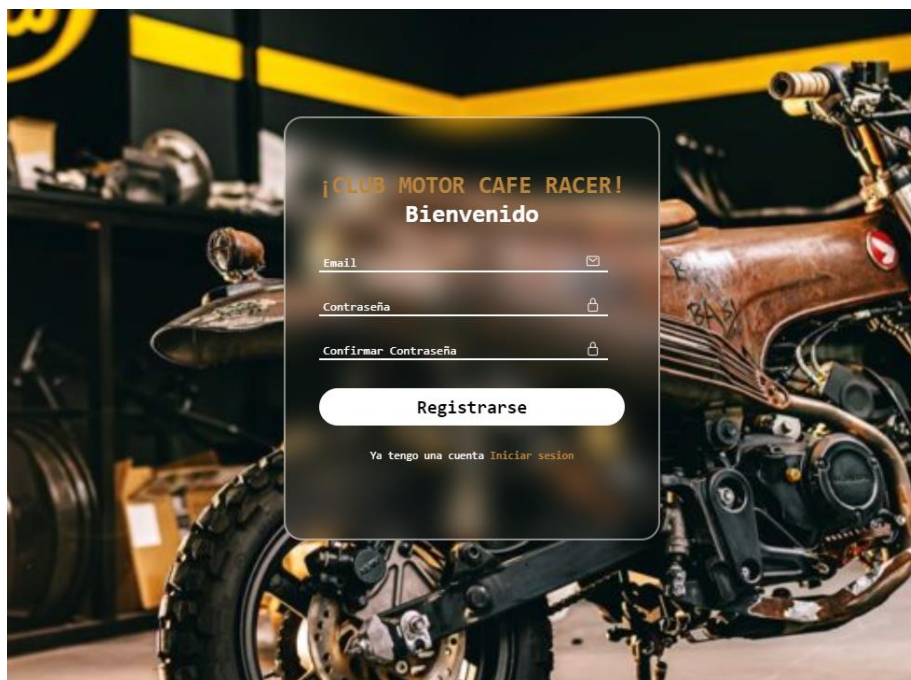
login.php

Es la primera pagina que te encuentras al lanzar la aplicación. Te permite la entrada al sistema si eres usuario y si no te da la opción de registrarte como nuevo usuario. Una vez logueado te redirige a la pagina principal.



registro_usuario.php

Consta de un formulario para registrar usuarios nuevos. Una vez registrado como nuevo usuario te redirige al login para acceder al sistema.



Validaciones lado cliente y lado servidor en ambos formularios.

Para validar en lado cliente en los formularios hemos optado por las propias validaciones de html5 como son el type para indicar el tipo de entrada de dato, required para los datos obligatorios y hemos controlado un mínimo de caracteres tanto en las contraseñas como en usuarios con minlength.

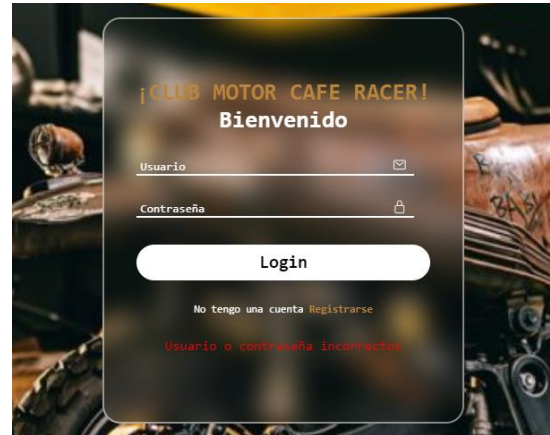
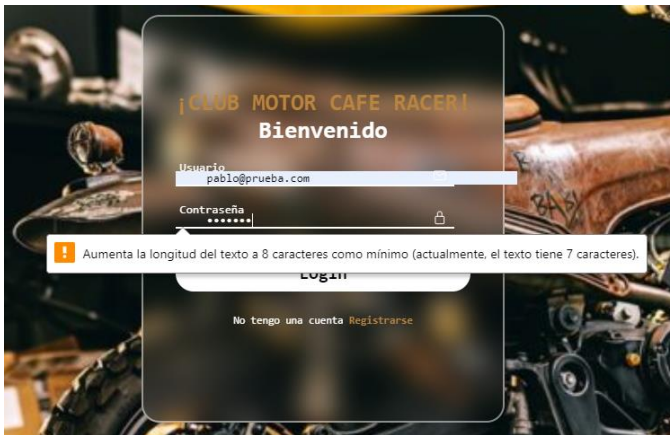
Para validar en lado servidor hemos utilizado la función contraseñasDB() que nos encripta la contraseña una vez te das de alta como nuevo usuario y la función filter_var que nos validad si son correctos los formatos de dirección de email. En el login en lado servidor una vez intentas loguearte con la función contraseñasDB() verifica con la BD si ha cambiado y es correcta la contraseña o no.

A parte en la lógica para procesar la entrada de datos de ambos formularios

```
1 if (isset($_POST['email']) && isset($_POST['contrasena']) && isset($_POST['registrarse'])) {
2     //Guardamos la credenciales obtenidas en dos variables
3     $email = $_POST['email'];
4     $contrasena = htmlspecialchars($_POST['contrasena']);
5     $confirmar_contrasena = htmlspecialchars($_POST['confirmar_contrasena']);
6     //Validamos el formato del email
7     if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
8         $error_message = "El formato del correo electronico no es valido. Inténtelo de nuevo";
9     } else {
10        //Vamos a proceder a comprobar que las dos contraseñas introducidas son iguales y encryptamos la contraseña.
11        if ($contrasena === $confirmar_contrasena) {
12            $contrasena_encryptada = hashContrasena($contrasena);
13            //Consulta donde insertamos el nuevo usuario en la BD.
14            $consulta = $conexion->prepare("INSERT INTO usuarios (username, contrasena) VALUES ( :correo, :contrasena)");
15            //Asignamos los parametros del registro mediante Bind.
16            $consulta->bindParam(":correo", $email);
17            $consulta->bindParam(":contrasena", $contrasena_encryptada);
18            //Ejecutamos la consulta para introducir los datos.
19            $consulta->execute();
20            //Redirigimos a la pagina login.
21            header("Location: ../login.php");
22            exit();
23        } else {
24            //Si entra por aqui quiere decir que las contraseñas introducidas no son iguales.
25            $error_message = "Las contraseñas introducidas no coinciden. Inténtelo de nuevo";
26        }
27    }
28 }
29 } else {
30     //Si entra por aqui los campos usuario y contraseña no estarían rellenos.
31     $error_message = "Por favor, rellene los campos de usuario y contraseña";
32 }
```

```
1 if ($row) {
2     $contrasenaDB = $row['contrasena'];
3     if (verificarContrasenas($contrasena, $contrasenaDB)) {
4         //Usuario estaría autenticado correctamente. Creamos la sesion y guardamos los datos en dos variables.
5         session_start();
6         $_SESSION['id'] = $row['usuario_id'];
7         $_SESSION['usuario'] = $row['username'];
8         $_SESSION['contrasena'] = $row['contrasena'];
9         //Establecemos unas cookies que serán accesibles en todos los directorios una vez que el usuario está logueado en la que guardaremos el id y usuario por 1 día.
10        setcookie("usuario_id", $row['usuario_id'], time() + 86400, "/");
11        setcookie("usuario_nombre", $row['username'], time() + 86400, "/");
12        //Redirigimos al usuario a la pagina principal.
13        header("location: PHP/principal.php");
14        die();
15    }
16 } else {
17     //Las credenciales son incorrectas. Guardamos el mensaje en una variable.
18     $error_message = "Usuario o contraseña incorrectos";
19 }
```


Mostramos distintos errores que controla la lógica de los formularios.



principal.php

Página que te encuentras una vez logueado correctamente. Consta de un pequeño formulario que te da la opción de ver todos los vehículos de la exposición del club o todos los socios del club. En esta página es donde el usuario decide si cerrar sesión y salir de la aplicación o no.

La lógica controla si el usuario se ha logueado correctamente para mostrar la página o no.

```
1 <?php
2 //Activamos la sesion
3 session_start();
4 //Comprobamos que las sesiones existen, es decir, verificamos que el cliente se ha logueado correctamente.
5 if(!isset($_SESSION['id']) && !isset($_SESSION['usuario'])){
6     //Si no existen lo redirigimos
7     header('Location: ../login.php');
8     exit();
9 }else{
10
11     //Creamos esta variable booleana para controlar si mostramos el formulario. Se mostrara siempre que el usuario se haya
12     //logueado y esten las sesiones creadas.
13     $Form_principal = true;
14
15 }
16
17
18 ?>
```



mostrar_principal.php




Esta página recibe los datos enviados mediante el formulario principal y la lógica gestiona que mostrar. Por la extensión del archivo aconsejamos mirarlo para observar bien como es la lógica y entenderla, en él viene detallado paso a paso el código. Mostramos lo que muestra la aplicación en cada caso.



Ver Vehículos:

Muestra una tabla con todos los vehículos de la exposición. La tabla la componen todos los campos de la BD y un botón modificar y otro borrar por cada vehículo.

En la parte inferior de la tabla tenemos dos botones que nos permiten volver a la pantalla principal o dar de alta un nuevo vehículo.

Seguido tenemos otro pequeño formulario que nos permite filtrar solo los vehículos de una determinada marca.

ID	MARCA	MODELO	PRECIO	IMAGEN	MODIFICAR	BORRAR
1	Honda	CB750	3000		Modificar	Borrar
2	Yamaha	SR250	2500		Modificar	Borrar
3	Honda	CB550	4300		Modificar	Borrar

13	Honda	CB450	7800		Modificar	Borrar
14	Honda	300	2430		Modificar	Borrar

[VOLVER PRINCIPAL](#)
[NUEVO VEHICULO](#)

Ver Vehículos de la marca

[VER](#)

- Pulsando modificar: Si el usuario pulsa modificar le aparece un pequeño formulario que le permite modificar el precio y el campo socio_id (FK). Solo permite modificar esos campos porque consideramos que la marca, modelo e imagen siempre serán las mismas pero el precio puede variar y su propietario (socio) también. Una vez hemos actualizado nos redirige automáticamente a principal por si queremos comprobar la modificación.

¡CLUB MOTOR CAFE RACER!

Introduce los datos para modificar el vehiculo 3

Precio

ID Socio

[ACTUALIZAR](#)
[BORRAR](#)
[PRINCIPAL](#)



- Pulsando borrar: Si el usuario pulsa borrar automáticamente el registro de ese vehículo queda borrado, esto lo hace por el campo vehículo_id (PK). Una vez borrado nos redirecciona a principal.
- Pulsando nuevo vehículo: Si el usuario pulsa le lleva a un formulario de inserción. Este formulario contiene los campos marca, modelo, precio, socio_id y subir el archivo imagen. Automáticamente al pulsar ingresar nos redirige a la página principal.

¡CLUB MOTOR CAFE RACER!

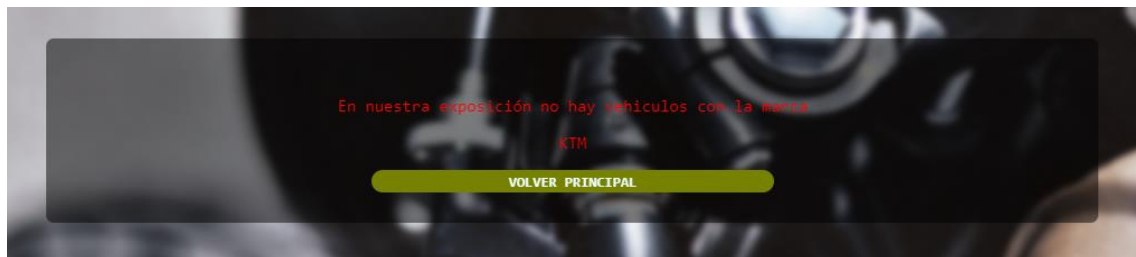
Introduce los datos del nuevo vehículo

Marca
 Modelo
 Precio
 Socio ID
 Imagen Seleccionar archivo Ninguno archivo selec.

- Con el formulario ver marcas: Si el usuario introduce una marca concreta y coincide con alguna de la BD devuelve una tabla con los resultados coincidentes, si no es así, devuelve un mensaje informando que no hay ninguna marca en la BD.
En este ejemplo el usuario a buscado los vehiculos con la marca Triumph.

ID VEHICULO	MARCA	MODELO	PRECIO	IMAGEN
7	Triumph	Bonneville	7900	
8	Triumph	Thruxton	3300	
9	Triumph	Legend TT	4150	

VOLVER PRINCIPAL



En este caso devuelve un mensaje informando que no hay vehiculos de esa marca.



Ver Socios:

Muestra una tabla con todos los socios/propietarios del club. La tabla la componen todos los campos de la BD, un botón que nos permite filtrar los vehiculos que tiene en posesión un determinado socio, dar de alta un nuevo socio, modificar y borrar.

ID SOCIO	NOMBRE	APELLIDOS	CIUDAD	SUS VEHICULOS	MODIFICAR	BORRAR
1	Pablo	Arcas Roldan	Albacete	Ver Vehículos	Modificar	Borrar
2	Irene	García Muñoz	Segovia	Ver Vehículos	Modificar	Borrar
3	Jorge	Perez Cuevas	Segovia	Ver Vehículos	Modificar	Borrar
4	Enrique	Arcas Fernandez	Madrid	Ver Vehículos	Modificar	Borrar
5	Fernando	Ortega Luzon	Valencia	Ver Vehículos	Modificar	Borrar

[VOLVER PRINCIPAL](#)
[NUEVO SOCIO](#)

- Pulsando ver vehículos: Si el usuario pulsa ese botón, se mostrará una tabla con los vehiculos que tiene ese socio en propiedad.

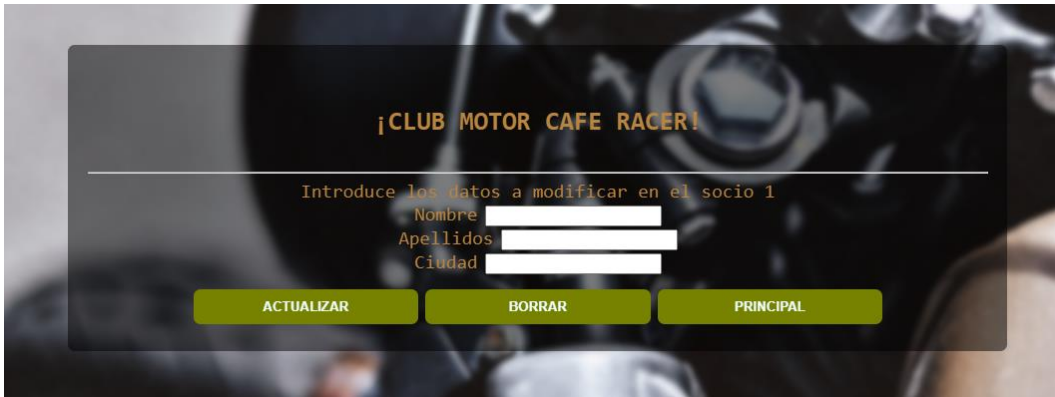
ID VEHICULO	MARCA	MODELO	PRECIO	IMAGEN
7	Triumph	Bonneville	7900	
13	Honda	CB450	7800	

[VOLVER PRINCIPAL](#)

En

este ejemplo, el socio Enrique Arcas Fernández dispone de 2 vehiculos.

- Pulsando modificar: Si el usuario pulsa modificar le permite cambiar los campos nombre, apellidos y ciudad del socio correspondiente.



- Pulsando borrar: Si el usuario pulsa borrar automáticamente el registro de ese socio queda borrado, esto lo hace por el campo socio_id (PK). Una vez borrado nos redirecciona a principal.

Nota: Si se da de baja un socio, en la tabla vehiculos, todos sus vehiculos quedan con el campo socio_id vacío a la espera de que un nuevo socio adquiera ese vehículo, esto lo haría en el formulario modificar de los vehiculos como hemos descrito anteriormente.

Logout.php

Una vez hemos pasado por todas las pantallas de la aplicación, si queremos cerrar sesión y salir de la aplicación lo haremos desde la pantalla principal en el botón cerrar sesión. La lógica de este botón la maneja el archivo logout.php en el cual destruimos la sesión existente y nos redirige al login por si queremos entrar de nuevo en la aplicación.



```
1 <body>
2 <!-- Aqui cerramos todas las sesiones existentes. -->
3 <?php
4 //activamos sesiones
5 session_start();
6 //destruimos la sesion
7 session_destroy();
8 //redirigimos al login
9 header('Location: ../Login.php');
10 exit();
11 ?>
12
13 </body>
```

modificar.php, insertar.php y borrar.php

Estos archivos comparten la misma lógica de programación, gestionamos tanto la tabla vehiculos como la tabla socios de nuestra BD en el mismo archivo. También incluimos tanto ambos formularios como la lógica de que hacer según los datos recibidos en esos formularios. Según la opción que seleccione el usuario mostraremos una cosa o otra, todo esto (bien detallado en los archivos paso por paso).

Durante todo el programa las sentencias SQL utilizadas han sido sentencias preparadas con PDO.

controlVisitas.php

También tenemos un archivo llamado que incluye una función llamada contadorVisitas(). Lo que queremos lograr con esto mediante cookies es tener un registro de las veces que un usuario pasa por las paginas que estimemos oportuno, en este caso por la principal.

```
1 <?php
2
3 function contadorVisitas(){
4     //Vamos a crear una cookie en la que vamos a contar las veces que el usuario visita las paginas que estimemos oportuno.
5     if (!isset($_COOKIE['contador_visita'])){
6         //Si no esta definida la cookie la inicializamos a 1, es decir, el usuario no ha visitado la pagina.
7         $contador = 1;
8     }else {
9         //Si la cookie esta definida incrementamos su valor, es decir, el usuario ya ha visitado la pagina.
10        $contador = $_COOKIE['contador_visita'] + 1;
11    }
12
13    //Creamos la cookie con el nuevo valor. Asi nos aseguramos que el contador se ha incrementado en lugar de reiniciarse cada vez
14    //Que el usuario visita la pagina en concreto. En este caso la cookie la mantenemos durante 1 dia.
15    setcookie('contador_visita', $contador, time() + 86400, "/");
16    //Devolvemos el valor actualizado.
17    return $contador;
18
19 }
20 ?>
```

CONCLUSIÓN

Me ha parecido una práctica muy interesante con la que me ha hecho afianzar conceptos y conocer otros nuevos.

Personalmente me ha sido muy laboriosa pero muy gratificante al ver los resultados. Creo que me servirá de mucho en un futuro próximo todo lo aprendido y estoy bastante contento por ello.