

MEMORIA DE LA PRÁCTICA 2 DE PROGRAMACIÓN WEB

En esta práctica 2 se ha dotado más dinamismo y funcionalidad al sitio web de venta de vehículos desarrollado en la práctica 1, gracias al uso de PHP, MySQL y JavaScript. Se han integrado funcionalidades como el registro (*punto1*), login (*punto2* y *punto3*), actualización (*punto5*) y borrado de usuario y de secciones en la web, y el registro y borrado de vehículos en la plataforma por el usuario “admin” (*punto4*). También se han validado todos los campos necesarios de los formularios (*punto6*) y se ha creado un pop-up simple con JavaScript (*punto7*).




El proyecto cuenta con la siguiente estructura y páginas que iremos comentado a continuación:

| Nombre | Fecha de modificación | Tipo |
|-------------------------|-----------------------|---------------------------|
| font | 31/05/2021 17:52 | Carpeta de archivos |
| Imágenes | 31/05/2021 17:52 | Carpeta de archivos |
| js | 19/06/2021 21:52 | Carpeta de archivos |
| php_PDO_POO | 19/06/2021 21:59 | Carpeta de archivos |
| cerrar_sesion | 10/06/2021 20:54 | PHP Script |
| Cómo_se_hizo | 15/05/2021 20:55 | Documento de Microsoft... |
| Cómo_se_hizo | 15/05/2021 20:53 | Adobe Acrobat Document |
| comprueba_login | 06/06/2021 13:01 | PHP Script |
| Contacto | 06/06/2021 20:07 | Opera Web Document |
| eliminar_perfil | 19/06/2021 21:50 | PHP Script |
| index | 19/06/2021 20:58 | PHP Script |
| item_registrado | 19/06/2021 21:29 | PHP Script |
| item1 | 19/06/2021 21:03 | PHP Script |
| item2 | 12/05/2021 20:18 | Opera Web Document |
| item3 | 12/05/2021 20:18 | Opera Web Document |
| procesar_act_perfil | 06/06/2021 19:30 | PHP Script |
| procesar_act_seccion | 19/06/2021 20:56 | PHP Script |
| procesar_alta_item | 10/06/2021 20:17 | PHP Script |
| procesar_alta_seccion | 19/06/2021 20:06 | PHP Script |
| procesar_borr_perfil | 06/06/2021 19:39 | PHP Script |
| procesar_borrar_seccion | 14/06/2021 23:18 | PHP Script |
| procesar_registro | 10/06/2021 20:24 | PHP Script |
| recuperar_contraseña | 10/06/2021 20:45 | Opera Web Document |
| recuperar_contraseña2 | 10/06/2021 20:47 | Opera Web Document |
| registro_item | 19/06/2021 21:29 | PHP Script |
| registro_seccion | 19/06/2021 20:35 | PHP Script |
| registro_usuario | 19/06/2021 21:30 | PHP Script |
| seccion_registrada | 14/06/2021 21:22 | PHP Script |
| seccion1 | 14/06/2021 21:21 | PHP Script |
| style_index | 13/06/2021 19:28 | Documento de hoja de e... |
| updatear_perfil | 19/06/2021 21:30 | PHP Script |
| updatear_secciones | 19/06/2021 20:57 | PHP Script |
| usuario_registrado | 19/06/2021 21:54 | PHP Script |
| warning | 19/06/2021 21:31 | PHP Script |

El directorio *js* contiene las funciones para validar los distintos formularios de la plataforma:

| | | |
|--------------------------|------------------|--------------------|
| validadorEliminarUsuario | 19/06/2021 21:52 | Archivo JavaScript |
| validadorLogin | 12/06/2021 22:17 | Archivo JavaScript |
| validadorRegistroItem | 13/06/2021 14:33 | Archivo JavaScript |
| validadorRegistroSeccion | 19/06/2021 20:36 | Archivo JavaScript |
| validadorRegistroUsuario | 13/06/2021 13:45 | Archivo JavaScript |
| validadorUpdateoSeccion | 19/06/2021 20:56 | Archivo JavaScript |
| validadorUpdateoUsuario | 19/06/2021 21:50 | Archivo JavaScript |

Clases y archivos de configuración del directorio “/pe2/php_PDO_POO”:





























| | | |
|---|------------------|-------------|
|  claseltems | 10/06/2021 20:07 | PHP Script |
|  clasePaginacion | 14/06/2021 19:29 | PHP Script |
|  claseSecciones | 19/06/2021 19:45 | PHP Script |
|  claseUsuarios | 06/06/2021 19:08 | PHP Script |
|  configuracion | 19/06/2021 21:59 | Archivo INC |
|  datosObject.class | 19/06/2021 18:44 | Archivo INC |

En los ficheros de este directorio definimos todas las operaciones que se podrán realizar con la base de datos de nuestra plataforma:


- “*configuracion.inc*”: Se encuentran definidas una serie de constantes, tanto para la conexión con la base de datos como para acceder a las tablas correspondientes de nuestro sistema (usuarios, ítems, secciones), así como la constante del número de ítems que queremos que se muestren por página.
- “*datosObject.class.inc*”: Definición de una clase abstracta que contiene tanto la conexión/desconexión con la base de datos como el constructor de los objetos y un método para recuperar el valor de un campo de un objeto determinado. Clase padre.
- “*claseUsuarios.php*”: Clase hija que hereda de “*datosObject.class*” donde podemos encontrar los métodos para registrar, hacer login, recuperar información, actualizar la información y eliminar la cuenta de los usuarios de nuestro sistema.
- “*claseItems.php*”: Clase hija que hereda de “*datosObject.class*” donde podemos encontrar los métodos para registrar, recuperar información, actualizar información y eliminar vehículos de la plataforma.
- “*clasePaginacion.php*”: Clase hija que hereda de “*datosObject.class*” y nos provee de dos funciones necesarias para realizar la paginación de los ítems en nuestro sistema. Por un lado, contar los registros que tiene una tabla; y por otro, recuperar los ítems que caben por cada página (según el tamaño de página definido en *configuracion.inc*).
- “*claseSecciones.php*”: Clase hija que hereda de “*datosObject.class*” donde podemos encontrar los métodos para registrar, recuperar información, recuperar todas las secciones de la tabla, actualizar sección y eliminar sección.

Las tablas de nuestra base de datos son las siguientes:

- Tabla “usuarios”

| # | Nombre | Tipo | Cotejamiento | Atributos | Nulo | Predeterminado | Comentarios | Extra | Acción |
|----------------------------|---|-------------|-------------------|-----------|------|----------------|-------------|-------|--|
| <input type="checkbox"/> 1 | usuario  | varchar(30) | latin1_spanish_ci | | No | Ninguna | | |  Cambiar  Eliminar  Más |
| <input type="checkbox"/> 2 | password | varchar(15) | latin1_spanish_ci | | No | Ninguna | | |  Cambiar  Eliminar  Más |
| <input type="checkbox"/> 3 | nombre | varchar(30) | latin1_spanish_ci | | No | Ninguna | | |  Cambiar  Eliminar  Más |
| <input type="checkbox"/> 4 | apellidos | varchar(30) | latin1_spanish_ci | | No | Ninguna | | |  Cambiar  Eliminar  Más |
| <input type="checkbox"/> 5 | email | varchar(30) | latin1_spanish_ci | | No | Ninguna | | |  Cambiar  Eliminar  Más |
| <input type="checkbox"/> 6 | telefono | varchar(15) | latin1_spanish_ci | | No | Ninguna | | |  Cambiar  Eliminar  Más |
| <input type="checkbox"/> 7 | nacimiento | varchar(15) | latin1_spanish_ci | | No | Ninguna | | |  Cambiar  Eliminar  Más |
| <input type="checkbox"/> 8 | intereses | varchar(15) | latin1_spanish_ci | | No | Ninguna | | |  Cambiar  Eliminar  Más |
| <input type="checkbox"/> 9 | conociste | varchar(30) | latin1_spanish_ci | | No | Ninguna | | |  Cambiar  Eliminar  Más |

- Tabla “secciones”

| # | Nombre | Tipo | Cotejamiento | Atributos | Nulo | Predeterminado | Comentarios | Extra | Acción |
|----------------------------|--|---------------|-------------------|-----------|------|----------------|-------------|----------------|--|
| <input type="checkbox"/> 1 | id  | int(11) | | | No | Ninguna | | AUTO_INCREMENT |  Cambiar  Eliminar  Más |
| <input type="checkbox"/> 2 | nombre | varchar(18) | latin1_spanish_ci | | No | Ninguna | | |  Cambiar  Eliminar  Más |
| <input type="checkbox"/> 3 | items | varchar(1000) | latin1_spanish_ci | | Sí | NULL | | |  Cambiar  Eliminar  Más |
| <input type="checkbox"/> 4 | titulo | varchar(50) | latin1_spanish_ci | | No | Ninguna | | |  Cambiar  Eliminar  Más |

- Tabla “items”

| # | Nombre | Tipo | Cotejamiento | Atributos | Nulo | Predeterminado | Comentarios | Extra | Acción |
|----------------------------|-------------|--------------|-------------------|-----------|------|----------------|-------------|-------|------------------------|
| <input type="checkbox"/> 1 | url_img | varchar(200) | latin1_spanish_ci | | No | Ninguna | | | Cambiar Eliminar Más |
| <input type="checkbox"/> 2 | nombre | varchar(100) | latin1_spanish_ci | | No | Ninguna | | | Cambiar Eliminar Más |
| <input type="checkbox"/> 3 | precio | int(11) | | | No | Ninguna | | | Cambiar Eliminar Más |
| <input type="checkbox"/> 4 | kilometraje | int(11) | | | No | Ninguna | | | Cambiar Eliminar Más |
| <input type="checkbox"/> 5 | potencia | int(11) | | | No | Ninguna | | | Cambiar Eliminar Más |
| <input type="checkbox"/> 6 | año | int(11) | | | No | Ninguna | | | Cambiar Eliminar Más |
| <input type="checkbox"/> 7 | ubicacion | varchar(100) | latin1_spanish_ci | | No | Ninguna | | | Cambiar Eliminar Más |

Gestión de usuarios:

- Registro de usuarios

registro_usuario.php -> procesar_registro -> usuario_registrado

El potencial usuario rellenará los campos del formulario de “registro_usuario.php” (debidamente validados con JavaScript), estos datos serán enviados por POST a otro fichero cuya única funcionalidad será llamar con los datos recibidos al método de la clase “claseUsuarios” llamado “registroUsuario” encargado de almacenar los datos del usuario en la base de datos. Una vez añadido el usuario, se almacena su sesión y es redirigido a la página “usuario_registrado”, en donde rescatamos su nombre de la sesión y se le agradece su registro. Posteriormente ya puede navegar por nuestra web como un usuario registrado más.

Se le habrán habilitado en el menú de la esquina superior derecha la opción de “actualizar su perfil”, en donde podrá actualizar su información e incluso eliminar su cuenta de usuario.

- Login

El login se podrá realizar desde cualquier página pública (que no requiera estar previamente logeado para acceder a ella) de nuestra plataforma. Una vez rellenados los campos del formulario (debidamente validados con JavaScript) se envían los datos de usuario y contraseña por POST a “comprueba_login.php” en donde se comprueba que los datos del usuario son correctos para iniciar sesión, llamando al método “loginUsuario” de la clase “claseUsuarios” con los valores de “usuario” y “contraseña” introducidos en el formulario. Si los datos son correctos, almacenamos una nueva sesión con el nombre de usuario introducido y lo redirigimos al “index.php” estando ya logeado.

Se le habrán habilitado en el menú de la esquina superior derecha la opción de “actualizar su perfil”, en donde podrá actualizar su información e incluso eliminar su cuenta de usuario. También podrá observar en ese mismo lugar un botón para cerrar sesión, que se procesará en el archivo “cerrar_sesion.php” donde se establecerán nulos los valores de la sesión, se destruye la sesión y se dirige al “index.php” en donde podrá iniciar sesión otra vez si lo desea.

- Actualizar información de usuario

updatear_perfil.php -> procesar_act_perfil

Para actualizar su información de usuario, el botón solo lo verá en la esquina superior derecha una vez que esté logeado.

El usuario verá su información actual en el propio formulario, y decidirá qué quiere actualizar. Puede actualizar solo su nombre, o solo su contraseña, o su nombre y contraseña. Rellenará los campos que desea actualizar (debidamente validados con JavaScript) y deberá introducir su

contraseña actual para confirmar cambios antes de enviar los datos a *“procesa_act_perfil.php”*, en donde se comprueba que la contraseña se corresponde con el usuario a actualizar y si es así, se llama al método *“updatearUsuario”* de la clase *“claseUsuarios”* en donde se actualizan los nuevos valores de los campos en la base de datos, comprobando también si la contraseña se ha actualizado o no. En este caso, se redirigirá siempre a la página *“updatear_perfil.php”* en donde se podrá comprobar desde el propio formulario si se han guardado los cambios correctamente (dando la sensación de no cambiar de página).

- **Eliminar cuenta de usuario**

Esta opción es solo visible cuando el usuario está logeado y en la página de *“updatear_perfil.php”*, si clicamos en ella nos dirige a *“eliminar_perfil.php”* donde se le solicitará la contraseña para confirmar que quiere borrar su cuenta de usuario. Una vez introducida se le redirigirá a *“procesar_borr_perfil.php”* en donde se comprueba si la contraseña introducida corresponde con la del usuario a través del método de *“loginUsuario”* y si es así se llamará al método encargado de eliminar la cuenta *“eliminarMiUsuario”* de la clase *“claseUsuarios”* encargado de recibir el nombre de usuario de la sesión por parámetros y eliminarlo de la base de datos. Si se elimina correctamente se destruye la sesión de usuario y se redirige al *“index.php”* en donde ya no estará registrado. Si no se elimina (no se ha insertado la contraseña correcta) se le redirige a la página *“eliminarPerfil.php”*.

Gestión de ítems

Existen dos tipos de usuario en nuestro sistema, usuarios genéricos y un usuario administrador por defecto llamado “admin”. Este último, aparte de toda la funcionalidad de los usuarios genéricos descrita en el punto anterior se le dota de capacidad para dar de alta ítems a partir del formulario desarrollado en la practica 1.

- **Registro de vehículos**

registro_item.php -> procesar_alta_item.php -> item_registrado.php

Para registrar un vehículo nuevo a nuestro sistema, deberemos ser el usuario “admin” y pulsar en el boton “Registrar vehículo” del menú de la esquina superior izquierda. Una vez en *“registro_item.php”* deberemos rellenar el formulario (debidamente validado con JavaScript) para posteriormente proceder con el registro en *“procesar_alta_item.php”*. Cabe destacar que lo que estamos haciendo es almacenando en la base de datos la dirección de la imagen que queremos (pero ésta ya debe estar en el servidor en dicha dirección), esto lo hacemos así debido a temas de seguridad del servidor, por lo que el campo del formulario tipo “file” es pura estética, lo importante son los datos del vehículo y la ubicación de la imagen en el servidor.

Una vez que enviamos el formulario, en *“procesar_alta_item.php”* comprobamos que la sesión es del usuario “admin” y llamamos al método *“registroItem”* de la clase *“claseItems”* pasándole por parámetros la información recibida del formulario anterior. Si el vehículo se registra correctamente nos redirige a *“item_registrado.php”*, en caso contrario volverá a *“index.php”*.

- **Paginación de vehículos**

Una vez que tenemos una serie de vehículos en nuestro sistema, nos damos cuenta de que necesitamos visualizarlos de forma ordenada dentro de una sección. Y montamos un sistema de paginación básico en *“seccion1.php”*. Para ello, tenemos definidos un par de métodos en la clase *“clasePaginacion”* y una constante con el número de vehículos a mostrar por página.

En la etiqueta php del main de *“seccion1.php”*, antes de cargar los vehículos, hacemos los cálculos para esta paginación. Primero debemos saber en qué página estamos, y existen dos opciones:

- Que estemos en la página 1, por ejemplo, cuando acabamos de entrar a “*seccion1.php*”.
- O que ya hayamos navegado por las páginas, en cuyo caso existirá un identificador de página en la URL para saber en qué página estamos en cada momento (GET).

Para verlo más claro:

```
<main>
<?php
    /*CALCULOS PAGINACIÓN*/
    require_once("php_PDO_POO/clasePaginacion.php");

    //obtenemos el parámetro que nos dice en qué página estamos actualmente

    $pagina=1; //inicializamos a la primera página por defecto

    if(array_key_exists('pag',$_GET)){ //si el valor pag existe en nuestra URL significa que
    estamos en una página específica (no la primera), es decir, ya se ha pulsado en alguna pagina
        $pagina = $_GET['pag'];
    }

    //calculamos el número de registros respecto de la página
    $contador_registros = clasePaginacion::contarRegistros(TABLA_ITEMS);

    //ya sabemos en qué página estamos en cada momento y cuántos registros tenemos en total

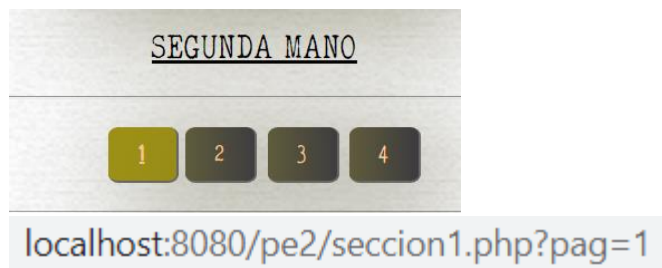
    $max_num_paginas = ceil($contador_registros/TAMANIO_PAGINA);//dividimos el contador del
    total de registros por el numero de registros que queremos por página // con intval no redondea
    correctamente, por eso usamos ceil

    //obtenemos el segmento paginado que corresponde con el número de página actual
    $segmento = clasePaginacion::registrosPagina($pagina, TABLA_ITEMS);
    ??
<hr>
<h2 class="tituloseccion"> SEGUNDA MANO </h2>

<hr>
<!--Código paginación al principio de la sección (y al final) -->
<section class="paginacion">
```

Esto es un poco más lioso por tanta concatenación, pero básicamente lo que estamos haciendo es creando los botones, es decir, los enlaces con el identificador de la página en la que nos encontramos en cada momento dentro de “*seccion1.php*”.

```
<ul>
<?php
    for ($i=0; $i < $max_num_paginas; $i++){
        if ($pagina == $i+1){
            //con ?pag hacemos que el valor vaya por la url
            echo '<li><a id="active" class="lbar" href="seccion1.php?pag=' . ($i + 1) . '>' . ($i + 1) . '
        </a></li>';
        }
        else{
            echo '<li><a class="lbar" href="seccion1.php?pag=' . ($i + 1) . '>' . ($i + 1) . '</a></li>';
        }
        //echo ...
        //echo ...
    }
    ??
</ul>
</section>
<hr>
```




Seguidamente creamos las cards de cada vehículo, cargando los items de nuestro sistema a mostrar en la página recorriendo el segmento y rescatando la información de los vehículos en su lugar correspondiente.


```
<section id="segundamano">
  <?php
    foreach ($segmento as $item){
      ?>
      <article class="img_card">
        <a href="item1.php">
          <table border="1">
            <tr>
              <th rowspan="6">
                
              </th>
              <td><h3><strong><?php echo $item['nombre'] ?></strong></h3></td>
            </tr>
            <tr>
              <td><?php echo $item['precio'] ?> € </td>
            </tr>
            <tr>
              <td><?php echo $item['kilometraje'] ?> KM </td>
            </tr>
            <tr>
              <td><?php echo $item['potencia'] ?> CV </td>
            </tr>
            <tr>
              <td>Año <?php echo $item['anio'] ?> </td>
            </tr>
            <tr>
              <td><?php echo $item['ubicacion'] ?> </td>
            </tr>
          </table>
        </a>
      </article>
    }
  </section>
</hr>
```

SEGUNDA MANO


1 2 3 4



**Seat Ibiza
Copa**
8500 €
110000 KM
140 CV
Año 2019
Soria



**Chevrolet
Orlando x7**
8500 €
175000 KM
120 CV
Año 2018
Vitoria




**Mercedes
Benz CLA**
12500 €
150000 KM
147 CV
Año 2017
Granada


1 2 3 4

SEGUNDA MANO


1 2 3 4



**Mercedes
Benz CLA
Sport**
15000 €
210000 KM
170 CV
Año 2016
Badajoz



**Mini 3
Countryman**
9800 €
95000 KM
140 CV
Año 2020
Cádiz




**Aston
Martin
Vanquish**
34500 €
90000 KM
340 CV
Año 2019
Madrid


1 2 3 4

SEGUNDA MANO


1 2 3 4



**Toyota Prius
Plus**
7500 €
180000 KM
140 CV
Año 2015
Valencia



**Audi S3
Sportback**
17500 €
75000 KM
140 CV
Año 2019
Zamora



**Land Rover
Defender**
12500 €
40000 KM
240 CV
Año 2014
Jaén

1 2 3 4

Gestión de secciones

- Registro secciones

registro_seccion.php -> procesar_alta_seccion.php -> seccion_registrada

Para añadir una sección más a nuestro sistema, deberemos ser el usuario “admin” y pulsar en el botón “Gestionar secciones” del menú de la esquina superior izquierda. Una vez en “*registro_seccion.php*” deberemos rellenar el formulario (debidamente validado con JavaScript) para posteriormente proceder con el registro en “*procesar_alta_seccion.php*”. Cabe destacar que lo que estamos haciendo es almacenando en la base de datos el nombre de la sección que queremos (y ésta ya debe estar en el servidor con dicho nombre en el directorio *pe2* para que se visualice correctamente) y el título que queremos que se muestre en la cabecera de la sección, en nuestro caso “*seccion1.php*”.

Una vez que enviamos el formulario, en “*procesar_alta_seccion.php*” comprobamos que la sesión es del usuario “admin” y llamamos al método “*registroSeccion*” de la clase “*claseSecciones*” pasándole por parámetros la información recibida del formulario anterior. Si la sección se registra correctamente nos redirige a “*seccion_registrada.php*” y podremos visualizarla en el menú de secciones, en caso contrario volverá a “*index.php*” y no podremos visualizarla.

- Gestionar secciones (actualizar y borrar)

➔ *procesar_act_seccion.php*

updatear_secciones.php:

➔ *procesar_borrar_seccion.php*

En este caso hemos creado un formulario dentro de una tabla en el archivo “*updatear_secciones.php*” para poder administrar todas las secciones de un vistazo siempre y cuando lo haga el usuario “admin”, una especie de panel de administración de secciones.

Cuando accedemos mediante el botón de “Actualizar secciones” de “*registro_seccion.php*” lo primero que se hace es comprobar la sesión del “admin” (como siempre) y luego recuperamos los datos actuales de la sección, dado que lo usaremos para poder actualizar en una fila de la tabla, los campos correspondientes a la sección seleccionada con el botón “Actualizar” (pasando su id por parámetros y volviendo a esta página). Posteriormente, llamamos al método “*datosSecciones*” de la clase “*claseSecciones.php*” para recuperar todas las secciones almacenadas en la base de datos y cargarlas en la tabla.

Si se pulsa en el botón de “Borrar” se eliminará la sección de la base de datos enviado el identificador de la sección a borrar por parámetros (en la URL) a “*procesar_borrar_seccion.php*” en donde se rescata el identificador de la sección a borrar y se le pasa al método “*EliminarSeccion*” de la clase “*claseSecciones.php*”. Tanto si se ha eliminado correctamente como si no, se nos redirigirá a la página “*updatear_secciones.php*” en donde podremos comprobar de un vistazo en la misma tabla si se ha eliminado correctamente o no.

Si se pulsa el botón “Actualizar” de una sección, ésta se cargará en última fila para así poder ver los valores actuales de sus campos (Nombre del fichero .php y Título) según el identificador de la sección seleccionada, el cual no se podrá modificar. Entonces, al modificar algún campo y pulsar en el botón de “Guardar cambios” del formulario (botón submit), éste enviará los datos por POST a “*procesar_act_seccion.php*” en donde se recuperarán y se le pasarán al método “*updatearSeccion*” de la clase “*claseSecciones.php*” para que proceda con la actualización de los campos. Tanto si se ha actualizado correctamente como si no, se nos redirigirá a la página

“*updatear_secciones.php*” en donde podremos comprobar de un vistazo en la misma tabla si se ha actualizado correctamente o no.

- **Mostrar secciones**

Para mostrar las secciones en el menú de todas y cada una de las páginas, debemos hacer una consulta a la tabla secciones justo antes de cargarlo para recuperarlas y posteriormente construir los botones de la barra de navegación de la siguiente manera:

```
<nav id="navigation">
    <?php
        /*CALCULOS PAGINACIÓN*/
        require_once("php_PDO_POO/clasePaginacion.php");
        require_once("php_PDO_POO/claseSecciones.php");

        $contador_registros = clasePaginacion::contarRegistros(TABLA_SECCIONES);
        $secciones = clasePaginacion::registrosPagina(0, TABLA_SECCIONES);
        //$secciones = claseSecciones::datosSecciones();

    ?>

    <article class="menu">
        <a href="#navigation" class="open" id="lbar"> Menú </a>
        <a href="#" class="close" > Cerrar </a>
    </article>

    <ul id="bar">

        <?php
            foreach($secciones as $seccion){

                echo '<a class="lbar" href="' . $seccion['nombre'] . '.php"><li class="linksBar">' . $seccion['titulo'] . '</li></a>';

            }

        ?>
    </ul>
</nav>
```