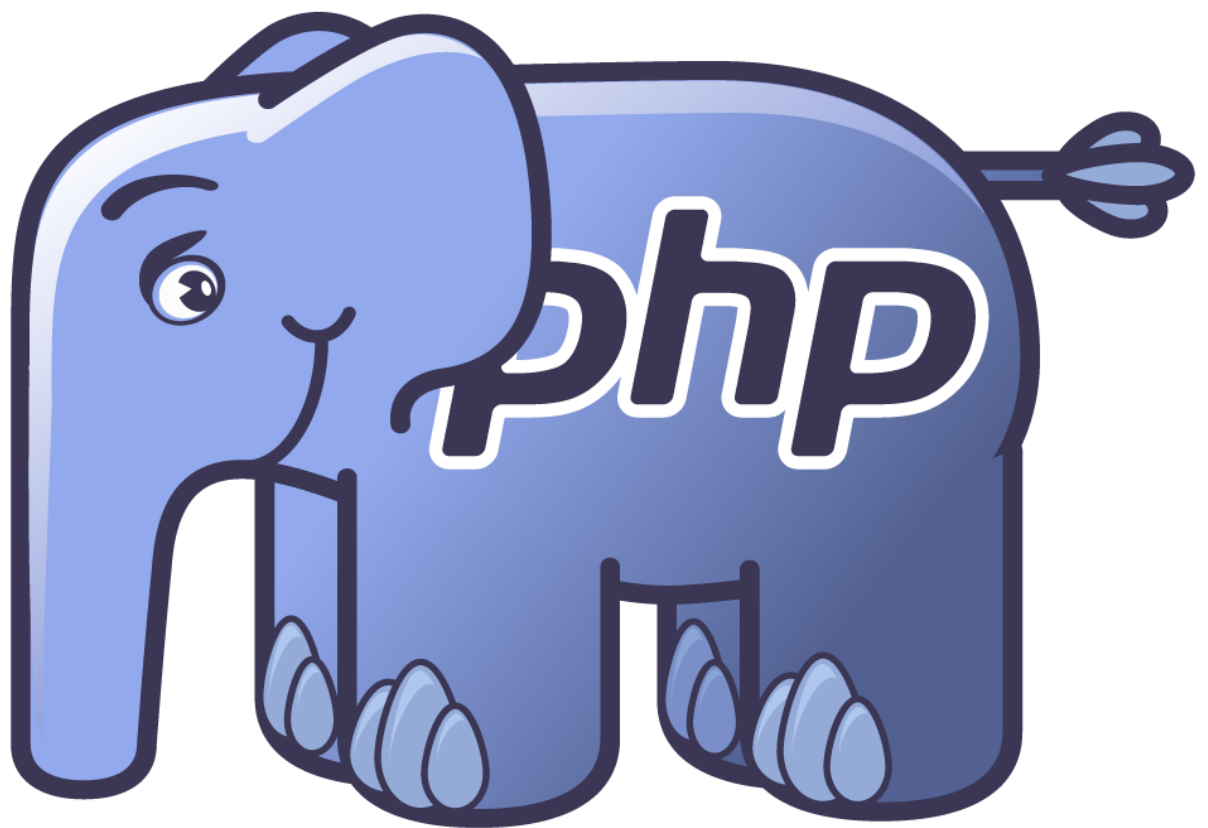


# PROVEEDORES CARRITO



I.E.S Severo Ochoa  
Desarrollo de Aplicaciones Web en Entorno Servidor

## Tabla de contenido

1.	Manual de usuario .....	2
1.1.	Login / logout .....	2
1.2.	Inicio .....	2
1.3.	Buscador de artículos.....	3
1.4.	Buscador pedidos.....	4
1.5.	Ver carrito .....	5
2.	Manual técnico.....	6
2.1.	Login / logout .....	6
2.2.	Inicio .....	7
2.3.	Buscador.....	8
2.4.	Buscador Pedidos.....	10
2.5.	Ver Carrito / Clases.....	11
2.5.1.	Clase Pieza.....	11
2.5.2.	Clase Carrito .....	11
2.5.3.	Ver Carrito .....	12

## 1. Manual de usuario

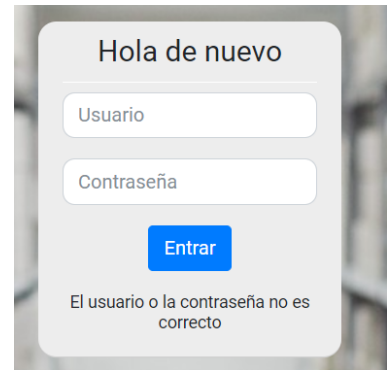
### 1.1. Login / logout

Introduzca su usuario y contraseña y haga clic en entrar una vez realizado se le mostrara un mensaje si tiene algún error en el proceso de login, de lo contrario se pasara a la página de inicio.

Un usuario para realizar pruebas es el siguiente:

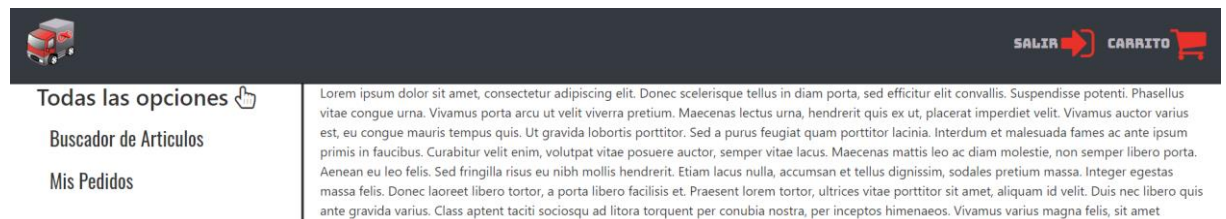
Usuario: 1

Contraseña: 965758745

A login form with a light gray background. At the top, it says "Hola de nuevo". Below that are two input fields: "Usuario" and "Contraseña". A blue button labeled "Entrar" is positioned below the password field. At the bottom, a message reads "El usuario o la contraseña no es correcto".

### 1.2. Inicio

Esta será la pagina principal de la web, dispondrá de un menú lateral y uno superior



### 1.3. Buscador de artículos

Este será el contenido inicial de la página.

Artículos

Campo donde buscar:


Nombre ▾

Modo en el que buscar

Exacto ▾

Nombre:

Buscar

Imagen	Numero	Nombre	Precio	Añadir al carrito
	2	Teclado ErgonOmico	15000	
	3	Teclado ps/2	2570	

Puede cambiar los campos y el método de búsqueda y dispondrá de un botón a la izquierda de los productos para añadirlos al carrito

## 1.4. Buscador pedidos

Este será el contenido inicial de la página.

### Mis Pedidos

**Recuerda:** Al hacer click en eliminar eliminarás el pedido completo, es decir, todos los articulos que tengan el mismo numero de pedido

Campo donde buscar:

Numero de pedido

Numero:

Buscar

Imagen	Num Pedido	Fecha	Num Pieza	Nom Pieza	Cantidad	Precio Unitario	Eliminar
	1	1992-05-05 00:00:00	M-0001-C	MONITOR LG 17P	10	30000	
	1	1992-05-05 00:00:00	P-0001-33	PLACA BASE SOLTEK 33Mhz	20	25000	

Puede cambiar los campos y el método de buscado y dispondrá de un botón a la izquierda de los pedidos para eliminar el pedido completo.

¡Este botón eliminara todos los productos del pedido!

## 1.5. Ver carrito

Este será el contenido inicial de la página.

**Recuerda:** Al cambiar la cantidad de un producto debes pulsar ✓ para confirmarlo

Numero	Nombre	Precio	Cantidad	Eliminar
2	Teclado ErgonOmico	15000	<input type="text" value="1"/> ✓	
C-1002-H		400	<input type="text" value="3"/> ✓	
3	Teclado ps/2	2570	<input type="text" value="1"/> ✓	
<b>Total:</b> 18770				
<div><button>Realizar Pedido</button><button>Borrar Carrito</button></div>				

Podrá eliminar un producto del carrito realizar el pedido o borrar el carrito con todos sus productos, si desea cambiar la cantidad de un producto puede hacerlo, no hay ningún limite mas que el que sea capaz de pagar, eso sí, debe pulsar el botón de chek para confirmar la cantidad antes de realizar el pedido.

Tambien cuenta con la opción de dejar un pedido a medias ya que en el caso de salir (con el botón de salir, no cerrando la ventana ) cuando tiene productos en el carrito, cuando vuelva los seguirá teniendo

## 2. Manual técnico

Para llevar acabo este proyecto se han usado las siguientes herramientas:

Font Awesome, usada para los iconos de la web, más información [aquí](#).

Bootstrap 4, un framework de css, más información [aquí](#).

Pooper, librería de javascript que sinceramente no tengo clara su utilidad dado que la pagina funciona correctamente sin el, pero Bootstrap 4 da un error de consola si no esta incluida, mas información [aquí](#) y [aquí](#).

Prefix free, una librería javascript para poner automaticamente los prefijos necesarios en algunos atributos css, más información [aquí](#).

jQuery, un framework de javascript, más información [aquí](#).

Sass, un preprocesador de css, más información [aquí](#).

Reset, una librería de css que reestablece todos los valores que pone el navegador de modo que se parte de una pantalla totalmente en blanco sin ninguna modificación css. Mas información [aquí](#).

Nota: A lo largo del documento se ara referencia a un carrito y sus acciones pero esto se explicara mas adelante en el apartado correspondiente

### 2.1. Login / logout

He dividido este apartado en 3 paginas:

Login.php: Muestra un formulario y envía los datos a comprobar.php.

comprobar.php: Recoge los datos que envía login, y ejecuta unas consultas preparadas para comprobar que los datos que se han introducido son correspondientes con los que figuran en la base de datos, de no serlo se retornara a login.php notificándolo, por el contrario, si son correctos se genera una sesión para el usuario en la que se guarda su nombre de usuario en `$_SESSION["usuario"]` y se pasa a inicio.php.

Logout: Se guarda el carrito y posteriormente se destruye completamente la sesión

```
$_SESSION = [];  
unset($_SESSION);  
session_destroy();
```

## 2.2. Inicio

Se comprobará que esta logeado el usuario y de no estarlo se le redirigirá a login.php cerrando la sesión que se ha iniciado para comprobar si existía `$_SESSION["usuario"]`, no se le redirige a logout ya que logout guarda un carrito antes de destruir la sesión.

Llegados a este punto, siempre nos mantendremos en esta página hasta el momento de Salir, los menus y el footer serán exactamente los mismos y solo variará el contenido de `#contenido`, sabremos a que pagina nos queremos dirigir por medio de una variable que pasaremos mediante un get implícito llamada `page`, esta contendrá el nombre del fichero php que queremos que se visualice en `#contenido`.

Se generara en `$_SESSION["carrito"]` un objeto carrito.

Todos los próximos apartados, a pesar de ser puntos diferentes a este, formaran parte de la pagina inicio dado que como he dicho, esta será la página que se cargara con un contenido u otro pero no nos moveremos de la página inicio.php.



## 2.3. Buscador

Buscador es un buscador de piezas.

Se mostrará un formulario dinámico realizado con jQuery.

Dependiendo de las opciones seleccionadas se llevará a cabo una sentencia SQL diferente, se sabrá que campo se ha seleccionado dado que será el único que estará creado ya que con jQuery eliminamos y creamos los campos, no simplemente los ocultamos y mostramos.

De modo que

```
if (isset($_REQUEST["inombre"])) {  
    $texto = $_REQUEST["inombre"];  
  
} else if (isset($_REQUEST["inúmero"])) {  
    $texto = $_REQUEST["inúmero"];  
  
} else if (isset($_REQUEST["imenor"])) {  
    $menor = $_REQUEST["imenor"];  
    $mayor = $_REQUEST["imayor"];  
    if ($menor > $mayor) {  
        $aux = $menor;  
        $menor = $mayor;  
        $mayor = $aux;  
    }  
    $texto = "BETWEEN $menor AND $mayor";  
}
```

Esto no se ha realizado con un switch dado que no he encontrado forma de implementar correctamente un isset() en un switch.

Cabe destacar que en el caso de que se realice una búsqueda por precio se comprobará que se haya introducido, por ejemplo entre 10 y 20, de modo que si se ha introducido entre 20 y 10, se le dará la vuelta a estos números, además, en lugar de igualar \$texto al contenido del input, se iguala directamente a una sentencia SQL.

Se realizara una conexión y se pasara por este filtro en donde \$modo hace referencia al modo de búsqueda que se ha introducido y \$texto al contenido a buscar, se comprueba que el campo sea diferente a PRECIOVENT dado que este se busca entre dos precios y no se puede filtrar de esta manera.

```
$sentencia = "select * from pieza where 1 = 1 ";

if ($campo != "PRECIOVENT") {
    switch ($modo) {
        case "Exacto":
            $sentencia .= "and $campo like '" . $texto . "'";
            break;
        case "Empieza":
            $sentencia .= "and $campo like '" . $texto . "%'";
            break;

        case "Acaba":
            $sentencia .= "and $campo like '%" . $texto . "'";
            break;

        case "Contiene":
            $sentencia .= "and $campo like '%" . $texto . "%'";
            break;
    }
} else {
    $sentencia .= "and $campo $texto";
}
```

Posteriormente se ejecuta la sentencia y se muestran los datos, en caso de que no se introduzca ningún filtro, se mostraran todos los productos, esto ha quedado así dado que la idea principal era incluir a la sentencia select un limit de modo que se mostraran x resultados y se pudiera realizar clic en una flecha para cambiar de pagina, pero esto no se ha llevado a cabo por falta de conocimientos.

Además, cada producto contara con un botón para añadirlo al carrito, de nuevo para esto se usara un get implícito que pasara únicamente el numero de la pieza.

## 2.4. Buscador Pedidos

Este tiene el mismo funcionamiento que el buscador simple, con la diferencia de que las sentencias SQL son mas complejas dado que debemos hacer select con inner join para juntar todos los datos correctamente.

Este inner join es necesario dado que están las tablas linped y pedidos, en pedidos se guarda el usuario al que pertenece el pedido, cosa que no se guarda en linped por lo que es necesario relacionar estas dos tablas para mostrar únicamente los datos de este usuario así como la tabla piezas para recoger los atributos de las piezas.

Cada pedido dispondrá de un botón para eliminar el pedido, que será una llamada a la función `eliminarpedido()`.

`Eliminarpedido()`

Se inicia el proceso de transacción estableciendo los autocomits en false.

Se declara una variable booleana `$correcto` que será true.

En el caso de que alguna consulta ( `$conexion -> Query()` ) sea errónea, se ejecutara un rollback.

Se ejecutara un commit y se volverá a reestablecer el valor de autocomits.

## 2.5. Ver Carrito / Clases

### 2.5.1. Clase Pieza

Contendra los atributos privados:

```
private $NUMPIEZA;  
private $NOMPIEZA;  
private $PRECIOVENT;  
private $RUTAIMG;  
  
private $cantidad=1;
```

Así como los get y set correspondientes y el método `aumentarcantidad()`, que sumara 1 al campo `cantidad`.

La separación éntrelos atributos es debido a que los 4 primeros se recogen de la base de datos y además tienen exactamente el mismo nombre, pero el campo `cantidad` no figura en la base de datos.

También se han definido los métodos mágicos necesarios.

### 2.5.2. Clase Carrito

Básicamente se trata de una clase que contiene un array asociativo de piezas y métodos para añadir piezas y borrar piezas.

Siempre que se quiera añadir una pieza se comprobará que esta no esté ya en el array y de estarlo, se accederá a ella y se le llamara al método `amentarcantidad()` de modo que utilizaremos el mismo método para añadir una pieza al array o aumentar la cantidad de una pieza del array.

Para eliminarlo se comprobará que existe en el array, en ambos métodos se comprobará que lo que se le pasa es un objeto de clase `Pieza`, mediante el método `instanceof`.

### 2.5.3. Ver Carrito

Como hemos dicho anteriormente, en `$_SESSION["carrito"]` se ha guardado un objeto Carrito, cuando se selecciona un producto a añadir al carrito se genera una pieza únicamente con su numero de pieza y se llamara a la función `addpieza()`, nombrada de esta forma porque no podía ser llamada `añadirpieza()`, la gran ventaja que nos ofrece guardar el carrito en una sesion es que no necesitamos serializar ni des serializar el carrito (que en realidad si se hace internamente por php), ahora:

¿Porque solo guardando el numero?

Esto es parte de algo que me ha gustado bastante, en un principio guardaba todos los datos en la clase pieza, pero posteriormente pensé que este es un campo clave el cual no se va a repetir, por lo cual, podemos acceder a cualquier contenido relacionado con este sabiendo su id, dicho de otro modo: no va a ser necesario para cada pieza que se pida, generar un nuevo objeto con su nombre, su ruta de imagen, su precio, etc esto es totalmente innecesario dado que toda esa información ya la tenemos en la base de datos (a excepción de la cantidad) y no tiene sentido duplicarla.

Dicho esto la forma de visualizar el carrito es la siguiente:

Se recogen todas las piezas que forman parte del carrito y para cada una de ellas se llama a sus métodos `get` para imprimirla en una tabla, se añade también un botón para borrar un producto el cual llamara a `delpieza()` así como la posibilidad de modificar la cantidad de un producto mediante un `input` de tipo `numero`, eso si, tratándose de php ha sido necesaria la incorporación de un formulario en dicho campo dado que se tiene que enviar un formulario para poder acceder a sus datos, este botón ha sido camuflado de la siguiente forma:

Junto al `input` de `numero` figura un `submit`, pero este tiene la propiedad `display:none`, este `input` tiene un `label` apunando a el, el cual es un icono de `check` por lo cual, al pulsar en el icono, pulsamos en el `submit`.

Tambien contamos con un botón para eliminar todas las piezas, la cual llamara a `carrito->vaciar()`.

Y otro para realizar el pedido, la cual llamara a la función `realizarpedido()`.