

# **SISTEMAS GESTORES DE BASES DE DATOS.**

## **IMPLANTACION DE UNA TIENDA WEB CON MYSQL y PHP**

FRANCISCO MARCET PRIETO  
1<sup>a</sup>ASIX

# Índice

1. INTRODUCCIÓN.....	4
2. ESPECIFICACIONES Y REQUISITOS.....	5
2.1 INTRODUCCIÓN.....	5
2.1.1 ÁMBITO.....	5
2.1.2 REFERENCIAS.....	5
2.2 DESCRIPCIÓN GENERAL.....	6
2.3 DEFINICIONES, ACRÓNIMOS Y ABREVIATURAS.....	8
2.3.1 FUNCIONES DEL PRODUCTO.....	9
2.3.2 CARACTERÍSTICAS DEL USUARIO.....	11
2.3.3 RESTRICCIONES GENERALES.....	11
2.4 REQUERIMIENTOS ESPECÍFICOS Y FUNCIONALES.....	12
2.4.1. CLIENTES ANÓNIMOS.....	12
2.4.2 CLIENTES REGISTRADOS.....	13
2.4.3 EMPLEADO/ADMINISTRADOR DE LA TIENDA WEB.....	14
2.5 REQUERIMIENTOS DE EFICIENCIA.....	15
2.6 SEGURIDAD.....	15
2.7. MANTENIMIENTO.....	15
2.8 OTROS REQUERIMIENTOS.....	15
2.8.1 BASE DE DATOS.....	15
3. ANÁLISIS.....	16
3.1 CASOS DE USO.....	16
3.2 DIAGRAMA DE SECUENCIA.....	17
4. DISEÑO.....	19
4.1 NIVEL DE INTERFAZ.....	20
4.1 DIAGRAMA DE NAVEGABILIDAD.....	21
4.1.1 USUARIO REGISTRADO.....	21
4.1.2 EMPLEADO/ADMINISTRADOR TIENDA VIRTUAL.....	22
4.3 NIVEL LÓGICO.....	22
4.4 NIVEL DE PERSISTENCIA.....	23
4.4.1. DISEÑO ENTIDAD RELACIÓN.....	23
4.4.2. DISEÑO LÓGICO.....	24
4.2.3 DIAGRAMA E-R.....	26
4.3.4 TIPO DE DATOS.....	26
5. MAPA DEL SITIO.....	29
6. IMPLEMENTACIÓN E INTEGRACIÓN.....	30
6.1 HERRAMIENTAS.....	30
6.2 IMPLEMENTACIÓN DE LA ADMINISTRACIÓN(BACKEND).....	31
6.2.1 PLANTILLA ADMINISTRACIÓN.....	31
6.2.2 PAGINA PRINCIPAL ADMINISTRACIÓN.....	32
6.2.3 LISTA DE PRODUCTOS.....	35
6.2.4 AÑADIR UN PRODUCTO.....	37
6.2.5 CATEGORÍAS.....	40
6.2.6 AÑADIR CATEGORÍA.....	43
6.2.7 EDITAR CATEGORÍAS.....	44
6.2.8 AÑADIR IMAGEN A PRODUCTOS.....	47
6.2.9 EDITAR PRODUCTOS.....	51
6.2.10 ELIMINAR PRODUCTOS.....	52
6.2.11 VALIDACIÓN DE CAMPOS.....	54

6.2.12 GESTIÓN DE CLIENTES.....	54
6.2.13 STOCK DE PRODUCTOS.....	56
<b>6.3 IMPLEMENTACIÓN PARTE PÚBLICA PARA CLIENTES (FRONTEND).....</b>	<b>57</b>
6.3.1 PLANTILLA PÁGINA PRINCIPAL.....	57
6.3.2 PÁGINA PRINCIPAL.....	58
6.3.3 ALTA USUARIO.....	59
6.3.4 LOGIN DE USUARIO.....	63
6.3.5 LOGOUT USUARIO.....	69
6.3.6 EXPIRACIÓN DE SESIÓN.....	71
6.3.7 MODIFICAR DATOS USUARIO REGISTRADO.....	73
6.3.8 INTRODUCCIÓN DATOS EN CATEGORÍAS.....	74
6.3.9 DETALLE DEL PRODUCTO.....	79
6.3.10 CARRITO DE COMPRA.....	80
6.3.11 FORMA DE PAGO.....	93
6.3.12 HISTORIAL DE COMPRAS DE UN CLIENTE. MIS COMPRAS.....	113
6.3.13 ELIMINAR PRODUCTOS CARRITO.....	121
<b>7. MEJORAS EN LA TIENDA.....</b>	<b>123</b>
<b>7.2 GESTIÓN TALLAS.....</b>	<b>130</b>
7.2.1 AÑADIR TALLAS.....	131
7.2.2 EDITAR TALLAS.....	133
7.2.3 ELIMINAR TALLAS.....	134
7.3.4 ASIGNACIÓN DE TALLAS A DIFERENTES PRODUCTOS.....	135
7.3.5 MOSTRAR TALLAS EN FRONTEND.....	137
7.3.6 COMPRA SIMULTANEA DE VARIOS PRODUCTOS.....	141
7.3 MEJORA DE ASPECTO.CSS .....	146
7.4 NOMBRE DE LA CATEGORÍA EN CADA SECCIÓN DE PRODUCTOS.....	147
7.5 PRODUCTOS EN OFERTA.....	148
7.6 BUSCADOR DE PRODUCTOS.....	151
7.8 PÁGINA DE CONTACTO.....	153
<b>8. CONCLUSIONES.....</b>	<b>154</b>
<b>9. AGRADECIMIENTOS.....</b>	<b>155</b>
<b>10. BIBLIOGRAFÍA.....</b>	<b>156</b>

# **1.INTRODUCCIÓN**

El diseño y la implementación de una tienda virtual tiene como resultado una aplicación web desde la cual, una empresa pueda ampliar sus ventas mediante el comercio electrónico, de manera fácil y sencilla. De esta manera se facilita a los clientes la opción de realizar sus compras sin salir de casa o simplemente visualizar los artículos disponibles, consulta de precios, el estado de algún pedido realizado anteriormente.

Por otra parte el personal autorizado(empleados y personal autorizado)podrá realizar operaciones como la modificación de pedidos, insertar nuevos artículos al inventario de la base de datos, o la corrección de datos erróneos que puedan aparecer en el registro de algún cliente

## **MOTIVACIONES**

El motivo principal por el cual he decidido realizar este proyecto es por adentrarme en lo posible en un sector que cada vez está más en auge como el comercio electrónico.

En cuanto a programación se refiere pese a tener conocimientos avanzados en C, C++,JAVA adquiridos cuando curse ingeniería industrial mis nociones de PHP son nulas, a día de hoy podría decirse que es la primera vez que me enfrento a una aplicación real de este lenguaje,ademas con los conocimientos de html y css adquiridos en el presente curso , y por ultimo este tipo de aplicación web creado y trabajado con bases de datos mediante Mysql me ha permitido conocer más a fondo el funcionamiento de este sistema de gestión de bases de datos, el cual el día que vuelva a mi trabajo de ingeniero me será de gran utilidad para la gestión de algún proyecto en el cual requiera de un manejo de bases de datos.

## **PLANTEAMIENTO TÉCNICO**

Tras decidir que la tienda web iba a ser una tienda de artículos de zapatería, escogí una tienda porque vi atractivo poder desarrollar dada la multitud de opciones que poseen las tiendas de comercio y el reto de poder implementarlas desde 0

Además tras haber consultado un poco de información sobre nuevas tecnologías y su diseño hemos de entender que un portal web eficaz es un portal valga la redundancia que centraliza la información y simplifica al máximo la tarea a realizar por el usuario creando una interfaz simple y evitando información extra que desvíe la atención del cliente.

## **2.ESPECIFICACIONES Y REQUISITOS**

### **2.1 INTRODUCCIÓN**

#### **2.1.1 ÁMBITO**

La tienda virtual va a consistir en una web dinámica que permita por una parte a los clientes de la empresa consultar el catalogo de productos, catalogo de ofertas .Y por otra parte al personal de la empresa por un intranet, a gestionar la base de datos tanto para la modificación como para la gestión de datos.

Ademas para aquellos usuarios que lo deseen mediante un registro muy sencillo podrán realizar sus comprar desde cualquier terminal conectado a internet, pudiendo estos disfrutar de opciones solo disponibles para usuarios registrados en el sistema

Otra función que se va a implementar es un buscador, el cual facilita a los clientes la localización de productos u ofertas.

## 2.1.2 CRONOGRAMA.

				mar 31	abr 7	abr 14	abr 21	abr 28	may 5
1	Requisitos Tienda web	2h	02/04/19	02/04/19					
2	Esquema ER y paso a tablas	3h	04/04/19	04/04/19					
3	Introduccion tablas Xampp	1h	05/04/19	05/04/19					
4	Plantilla Administracion	3h	06/04/19	06/04/19					
5	Conexion Base de datos	2h	06/04/19	06/04/19					
6	Gestion de Productos Admin	4d	07/04/19	10/04/19					
7	Gestion de Categorias Admin	10h	11/04/19	12/04/19					
8	Plantilla Parte Publica	2h	14/04/19	14/04/19					
9	Alta usuario	4h	15/04/19	15/04/19					
10	Login Usuario	3h	17/04/19	17/04/19					
11	Añadir imagenes	4h	18/04/19	18/04/19					
12	Validacion de Campos	2h	19/04/19	19/04/19					
13	Gestion Clientes Admin	2h	20/04/19	20/04/19					
14	Logout	1h	21/04/19	21/04/19					
15	Expiracion de la sesion	2d	23/04/19	24/04/19					
16	Modificacion Datos Usuario	2h	25/04/19	25/04/19					
17	Datos en categorias	6h	26/04/19	26/04/19					
18	Carrito de la Compra 1	1d	27/04/19	27/04/19					
19	Carrito de la Compra 2	8h	28/04/19	28/04/19					
20	Forma de Pago 1	4h	30/04/19	30/04/19					
21	Forma de Pago 2	4h	01/05/19	01/05/19					
22	Historial Compras 1	5h	03/05/19	03/05/19					
23	Historial Compras 1	5h	04/05/19	04/05/19					
24	Eliminar Carrito	2h	05/05/19	05/05/19					
25	Gestion Tallas 1	6h	07/05/19	07/05/19					
26	Gestion Tallas 2	6h	09/05/19	09/05/19					

Nombre de la tarea	Duración	Inicio	Finalizar	abr					may		
				mar 31	abr 7	abr 14	abr 21	abr 28	may 5	may 12	may 19
27	Compra Simultanea de Productos	1d	11/05/19	11/05/19							
28	CSS 1	3h	12/05/19	12/05/19							
29	CSS 2	3h	13/05/19	13/05/19							
30	Nombre Categoría en Productos	1h	21/05/19	21/05/19							
31	Ofertas	6h	14/05/19	14/05/19							
32	Buscador de Productos	4h	15/05/19	15/05/19							
33	Memoria Proyecto 1	8h	18/05/19	18/05/19							
34	Memoria Proyecto 1	8h	19/05/19	19/05/19							

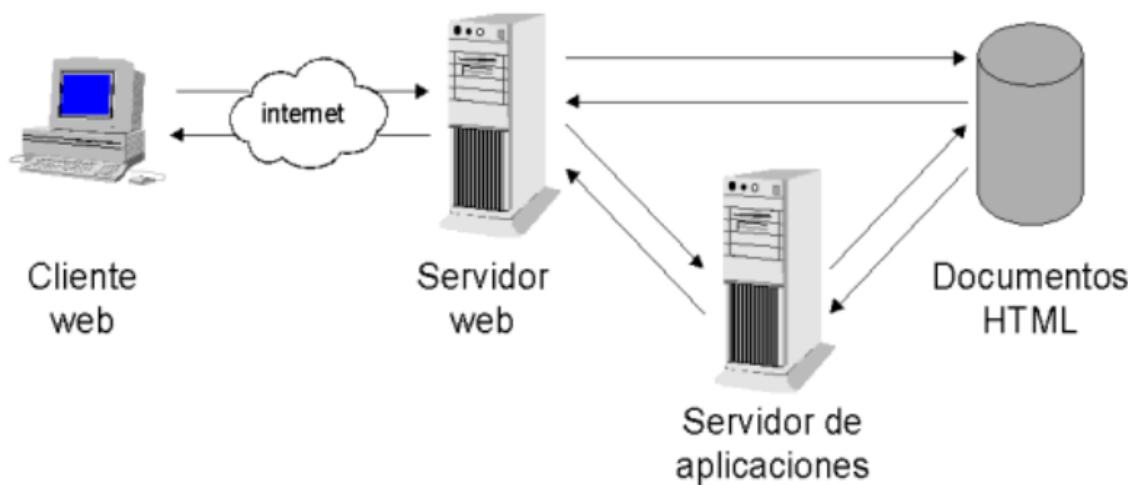
## 2.2 DESCRIPCIÓN GENERAL

El lenguaje utilizado para implementar la tienda virtual han sido los siguientes :

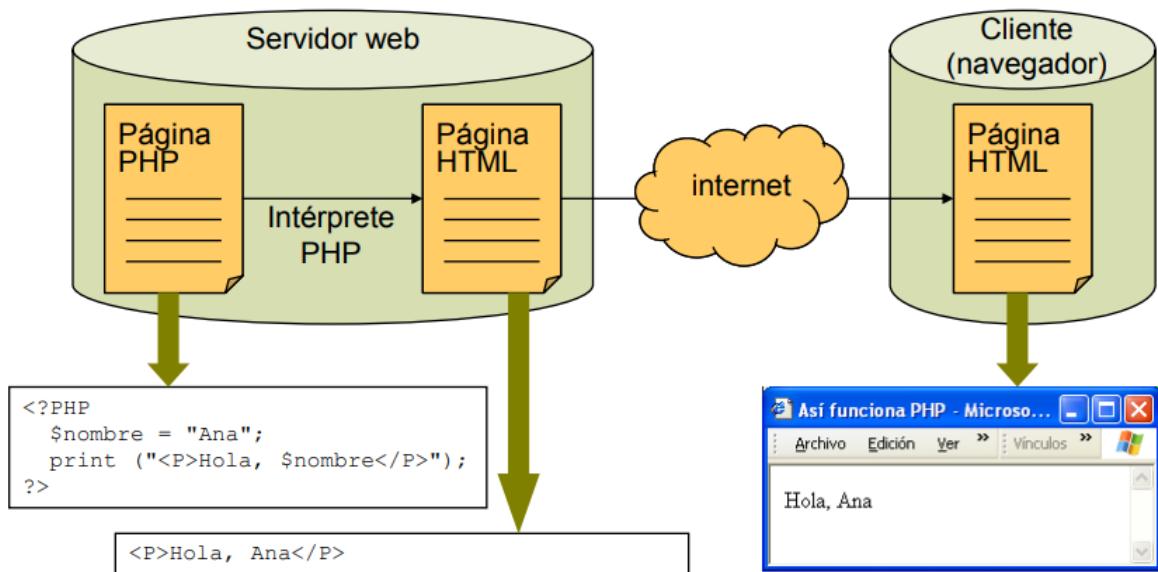
### PHP



PHP es un lenguaje de script del lado del servidor, los scripts PHP están incrustados en los documentos HTML y el servidor los interpreta y ejecuta antes de servir las páginas al cliente. El cliente/Usuario final no ve el código PHP sino los resultados que produce

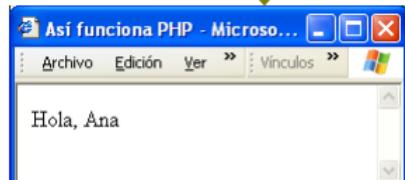


Mediante este grafico lo entendamos mejor



```
<?PHP  
$nombre = "Ana";  
print ("<P>Hola, $nombre</P>");  
?>
```

```
<P>Hola, Ana</P>
```



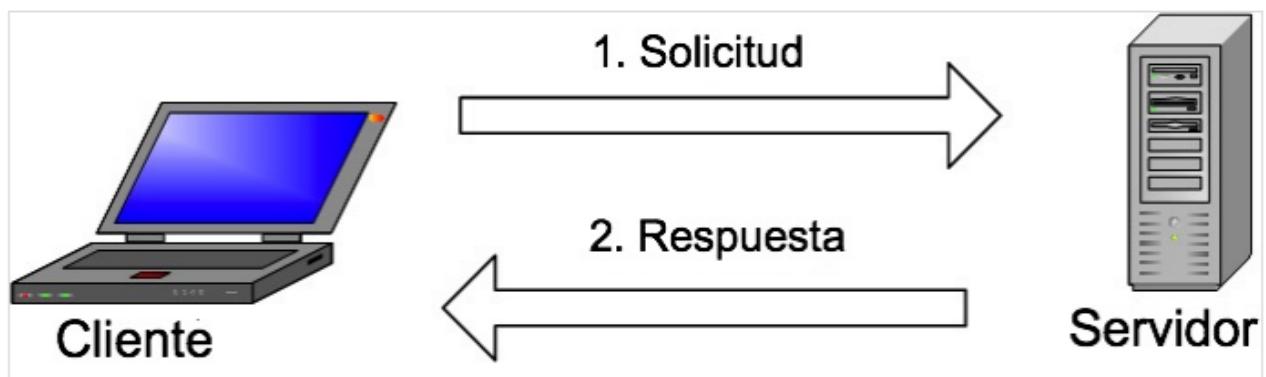
## MySQL



MySQL es un servidor de Bases de Datos SQL (Structured Query Language)

SQL -siglas de *Structured Query Language*-, es el lenguaje de consultas a bases de datos, que nos permitirá crear, modificar, consultar y eliminar tanto bases de datos como sus tablas y registros, desde el shell interactivo de MySQL

¿Cómo funciona MySQL?



La imagen explica la estructura básica cliente-servidor. Uno o más dispositivos (clientes) se conectan a un servidor a través de una red específica. Cada cliente puede realizar una solicitud desde la interfaz gráfica de usuario (GUI) en sus pantallas, y el servidor producirá el output deseado, siempre que ambas partes entiendan la instrucción. Sin meternos demasiado a fondo en temas técnicos, los procesos principales que tienen lugar en un entorno MySQL son los mismos, y son:

1. MySQL crea una base de datos para almacenar y manipular datos, definiendo la relación de cada tabla.
2. Los clientes pueden realizar solicitudes escribiendo instrucciones SQL específicas en MySQL.
3. La aplicación del servidor responderá con la información solicitada y esta aparecerá frente a los clientes.

Ademas de JAVASCRIPT, HTML,y CSS los cuales no entrare en detalle dado que no son objeto de este curso que como bien hemos mencionado antes, nos permitirán la creación de una pagina Web dinámica. Una vez decidido esto,necesitamos un servidor Web que nos permita ejecutar el código en PHP a la vez que nos permita acceder a la base de datos MySQL.

El servidor Web utilizado es Apache

## **APACHE**



Apache es un software de servidor web gratuito y de código abierto. El nombre oficial es Apache HTTP SERVER, y es mantenido y desarrollado por la Apache Software Foundation.

No vamos a entrar en detalle de su funcionamiento dada su complejidad de cómo funciona un servidor web.

El trabajo básico de todos los servidores web es aceptar solicitudes de clientes (por ejemplo, el navegador web de un visitante) y luego enviar la respuesta a esa solicitud (por ejemplo, los componentes de la página que el visitante quiere ver).

El servidor web Apache tiene módulos que agregan más funciones a su software, como MPM (para el manejo de modos de procesamiento múltiple) o mod\_ssl para habilitar la compatibilidad con SSL v3 y TLS

## **XAMPP**

Es una distribución de Apache que incluye MySQL, PHP y otras herramientas para el desarrollo de aplicaciones web, como phpMyAdmin , nos permite tener un servidor local en la dirección IP 127.0.0.0 en el cual podemos realizar pruebas de implementación,testeo de nuestra aplicación,y así comprobar su correcto funcionamiento antes de subirla al host real .

En muchas aplicaciones informáticas en tiendas físicas u otros locales estas aplicaciones también funcionan en un servidor local sin necesidad de tener acceso a Internet

## 2.3 DEFINICIONES, ACRÓNIMOS Y ABREVIATURAS

**TV:**Abreviatura de Tienda Virtual

**Web:** La World Wide Web (del inglés, Telaraña Mundial), la Web o WWW, es un sistema hipertexto que funciona sobre Internet. Para ver la información se utiliza una aplicación llamada navegador Web para extraer elementos de información (llamados “documentos” o “páginas Web”) de los servidores Web (o “sitios”) y mostrarlos en la pantalla del usuario.

**Web dinámica:** Existen dos tipos de páginas Web, de contenido estático (HTML) y de contenido dinámico que se generan a partir de lo que el usuario introduce en un Web o formulario y que utiliza el servidor para construir una Web personalizada que envía al cliente.

**MySQL:** MySQL es el servidor de datos relacionales más popular, desarrollado y proporcionado por MySQL AB. MySQL AB es una empresa cuyo negocio consiste en proporcionar servicios en torno al servidor de bases de datos MySQL. Una de las razones para el rápido crecimiento de popularidad de MySQL, es que se trata de un producto Open Source, y por tanto, va de la mano con este movimiento.

**HTML:** Acrónimo inglés de Hyper Text Markup Language (lenguaje de marcación de hipertexto), es un lenguaje informático diseñado para estructurar textos y presentarlos en forma de hipertexto, que es el formato estándar de las páginas Web.

**CSS:** Las hojas de estilo en cascada (Cascading Style Sheets, CSS) son un lenguaje formal de ordenador usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión XHTML). La W3C (World Wide Web Consortium) es la encargada de formular la especificación de las hojas de estilo que servirá de estándar para los agentes de usuario o navegadores.

**XHTML:** Es el lenguaje de marcación pensado para sustituir a HTML como estándar para las páginas Web. XHTML es la versión XML de HTML, por lo que tiene, básicamente, las mismas funcionalidades, pero cumple las especificaciones, más estrictas, de XML. Su objetivo es avanzar en el proyecto del World Wide Web Consortium de lograr una Web semántica, donde la información, y la forma de presentarla estén claramente separadas.

### **2.3.1 FUNCIONES DEL PRODUCTO**

Las funciones que va a realizar la tienda web son las siguientes:

- **Consultas sobre la colección**
  - Consulta según categoría.
  - Consulta de los detalles de un artículo.
  - Consulta de ofertas.
  - Consulta del carrito.
- **Gestión/administración usuarios.**
  - Insertar clientes en la base de datos
  - Modificar datos cliente en la BD
  - Control de Usuarios
- **Modificación del catálogo de productos**
  - Insertar nuevos artículos
  - Eliminar artículos del catálogo
- **Operaciones sobre los productos**
  - búsqueda de productos
  - modificación estado pedido
  - insertar pedidos base de datos
- **Tratamiento sobre usuarios**
  - mostrar información de los usuarios
  - mostrar información empleados
  - Iniciar sesión como usuario/empleado autenticado
- **Modificación del carrito**
  - Añadir artículos
  - Eliminar artículos
  - Modificar cantidad de artículos

### **2.3.2 CARACTERÍSTICAS DEL USUARIO**

Vamos a separar los usuarios de la TV en tres grupos: cliente anónimo, cliente registrado y empleado/Administrador de la tienda virtual

El **cliente anónimo** será un usuario que simplemente desee realizar consultas sobre el catálogo o los precios y no se haya registrado todavía como cliente. Pese a no haberse registrado, a este usuario se le permitirá visualizar productos y cuando desee realizar su compra ha de registrarse en el sistema para que se vaya familiarizando con la página y se le facilite al máximo la tarea de realizar su compra. Una vez decidida registrarse como cliente registrado, todos los artículos añadidos al carrito previamente permanecerán ahí para poder finalizar la compra directamente.

El **cliente registrado** es un usuario que anteriormente ha introducido sus datos como cliente de la tienda virtual y ya cuenta con un código en nuestro caso el email ya que a día de hoy es un identificador único ya que no existe la posibilidad de que 2 emails sean idénticos y una contraseña para registrarse. Este cliente verá su nombre en el cuadro de la izquierda con la información actualizada de su carrito. Además tendrá la opción de visualizar sus pedidos anteriores y el estado de estos. también, si así lo desea, podrá finalizar su compra pasando por caja y realizar así el pedido de los artículos seleccionados.

El **empleado registrado/Administrador** tendrá acceso a la intranet de la tienda virtual para realizar las operaciones y gestiones que esta facilita.

### **2.3.3 RESTRICCIONES GENERALES**

Como restricción general, mencionar que la posibilidad de modificar la base de datos por parte del personal de la tienda virtual es una modificación de nivel básico. Como única modificación, el empleado podrá modificar el contenido de la base de datos pero nunca su estructura.

Esta modificación, en caso de ser necesario, será llevada a cabo por parte del administrador del sistema de la base de datos ya que, en caso de modificarse la estructura, se necesitará también una pequeña modificación en el código de la tienda virtual.

Otra cosa a tener en cuenta es la posibilidad de que haya una gran afluencia de usuarios en la tienda virtual al mismo tiempo. Este problema podría causar una saturación del servidor así como lentitud en el funcionamiento de la página. Estos problemas podrían solucionarse simplemente aumentando la potencia del servidor con nuevo hardware, aplicando técnicas de trabajo distribuido, o construyendo un clúster de servidores.

La velocidad del ancho de banda del que dispone el servidor también puede ser un problema a tener en cuenta ya que, puede convertirse en el principal cuello de botella. Por tanto, esta elección no debe obviarse como sencilla y realizarse correctamente. también debe existir una política de seguridad adecuada en toda la tienda virtual. Los accesos incorrectos a páginas de la tienda virtual sin haberse registrado es un problema que hay que evitar. también tendremos en cuenta la seguridad dentro del servidor instalando cortafuegos, antivirus, cuentas de usuario, permisos, etc., además de realizar copias de seguridad periódicamente.

## **2.4 REQUERIMIENTOS ESPECÍFICOS Y FUNCIONALES**

### **REQUERIMIENTOS FUNCIONALES**

#### **2.4.1. CLIENTES ANÓNIMOS**

##### **Consulta según categoría.**

Este tipo de consulta consiste en una consulta SQL que retorna todas las diferentes categorías de la base de datos. Estas categorías aparecerán en el submenú de la Tienda web desde el cual, el cliente podrá seleccionar la que deseé y así serán mostrados todos y cada uno de los artículos pertenecientes a esa categoría.

##### **Consulta de los detalles de un artículo.**

Con esta consulta, el cliente o usuario anónimo podrá ver una información mas detallada acerca del artículo que deseé. Pinchando en la fotografía del artículo que se deseé, esta acción devolverá un número de referencia mediante el cual, una consulta SQL seleccionará todos los detalles del artículo con esa referencia y serán mostrados.

##### **Consulta de Ofertas.**

A esta consulta se puede acceder mediante dos vías. La primera es desde la página principal, una vez se abre la página de inicio de la tienda web aparecen todos los artículos referenciados en la base de datos como oferta (Oferta = 1). La segunda forma es desde la parte de Administración o Backend, en la cual podremos seleccionar que productos van a aparecer en la sección de ofertas

##### **Búsqueda de producto.**

Existe un buscador para facilitar esta opción al cliente. Una consulta SELECT en SQL buscará el producto en cuestión lo mostrará en la página web. El cliente podrá buscarlo directamente en el listado o filtrarlo mediante este buscador.

##### **Insertar cliente en la base de datos.**

Esta función la tendrá que realizar todo usuario que desee realizar una compra en la TV y no la haya realizado previamente. Mediante un formulario sencillo, el cliente no registrado rellenará todos los campos necesarios con sus datos personales. Una vez cumplimentado este paso, pinchará el botón Registrarse que generará una consulta inserten SQL que insertará los datos del cliente en la tabla usuarios de la base de datos. En caso de dejarse algún campo por llenar, el sistema generará un error visual para el cliente que tendrá que llenarlo de nuevo.

## **2.4.2 CLIENTES REGISTRADOS**

La mayoría de las funciones mencionadas anteriormente son para clientes registrados.

### **Insertar pedido en la base de datos.**

Una vez finalizada la compra por parte del cliente, siempre y cuando se haya registrado, podrá finalizar su pedido pasando por caja. Esta opción genera un INSERT en la base de datos que introduce los datos la tabla de la base de datos. Se introduce en la tabla pedidos el pedido con el código del cliente, el estado y la fecha. Por otra parte otro INSERT en SQL introduce el pedido detallado con cada artículo y cantidad en la tabla carrito, esperando a ser validada.

### **Mostrar información de los usuarios.**

Todo usuario que se encuentre registrado en la TV verá su nombre en el marco izquierdo con toda la información actualizada del carrito. Además, permite la opción de desconectarse en todo momento.

### **Inicio de sesión como usuario autenticado.**

Todo usuario que previamente se haya registrado en la TV, podrá identificarse introduciendo su DNI y su contraseña. Esta función crea una variable sesión (email) que será la que indique que ese usuario está autenticado. Como hemos mencionado en el punto anterior, también podrá desconectarse mediante la opción desconectar. Esta opción elimina la variable sesión MM\_USUARIO.

### **Búsqueda de producto.**

Existe un buscador para facilitar esta opción al cliente. Una consulta SELECT en SQL buscará el producto en cuestión lo mostrará en la página web. El cliente podrá buscarlo directamente en el listado o filtrarlo mediante este buscador.

### **Mostrar información de los usuarios.**

Una vez registrado el usuario, este tendrá la opción de visualizar todos los pedidos realizados anteriormente así como el estado en el cual estos se encuentran. El empleado de la TV será el encargado de actualizar este estado, pudiendo confirmar o cancelar la compra remiendo un email al comprador avisándole del estado de su compra.

Además este usuario podrá visualizar el estado de su pedido en Mis compras, el cual es un histórico de las compras efectuadas a lo largo del tiempo

### **Modificación de datos usuarios**

Una vez registrado el usuario, este tendrá la opción de modificar sus datos personales, junto con sus datos de acceso al sistema email y contraseña

### **Consulta del carrito.**

Solo el cliente registrado y que ha accedido al sistema podrá consultar en todo momento el carrito independientemente de donde se encuentre, podrá visualizar los artículos que ha seleccionado para ser comprados. Desde el marco situado en la izquierda de la página en el que aparece el resumen del carrito, se accederá a la pantalla que muestra la tabla con todos los artículos añadidos al carrito.

### **Añadir artículo al carrito.**

Cualquier usuario que entre a la página podrá añadir artículos en el carrito. Cuando se abre la página se crea automáticamente una variable sesión que será la que contendrá los artículos que el cliente desee adquirir. Estos artículos se añaden simplemente desde la pantalla de los detalles del producto pinchando en el botón “Comprar”.

### **Eliminar artículo del carrito.**

Desde la pagina de detalles del carrito, el cliente puede eliminar todos los artículos que desee. La tabla que muestra los artículos que hay en el carrito; Este podrá dispone del campo “Eliminar” con un enlace para cada artículo. Seleccionando el que se deseé eliminar y haciendo clic sobre el botón Eliminar, se eliminarán todos los artículos marcados eliminando los artículos de la variable sesión.

Este procedimiento se podrá realizar tanto como para cada producto individualmente, como vaciar el carrito completamente.

### **Modificar cantidad en el carrito.**

De igual manera que existe la columna borrar en la tabla que muestra los artículos del carrito, está la columna “cantidad” que contiene un + para añadir nuevas unidades y - para eliminar unidades para cada artículo indicando el numero de unidades que se desea comprar. Independientemente de la variable sesión que contiene los artículos, existe otra variable con las cantidades que, en caso de querer modificarse.

## **2.4.3 EMPLEADO/ADMINISTRADOR DE LA TIENDA WEB**

### **Visualización de clientes en la base de datos.**

Un sencillo select en SQL permite a los empleados de la TV visualizar los datos de cualquier cliente. Tras ser mostrados en una tabla los datos actuales que existen en la base de datos.

### **Mostrar información sobre compras**

Uno de los privilegios de los que disponen los empleados es el de tener acceso a la intranet de la TV. Cuando el empleado desee acceder a esta intranet, deberá introducir su email y su contraseña. El sistema comprobará que esos datos se encuentran en la tabla compras de la base de datos. En caso afirmativo, permitirá la entrada del empleado a dicho espacio.

### **Modificación del estado de un pedido.**

Todos los pedidos realizados por clientes en la TV disponen de un código de estado (Pendiente, Aceptado y enviado, Cancelado).

### **Mostrar información de los empleados.**

Igual que ocurría con los clientes, el empleado registrado en la TV podrá ver su nombre en el marco que existe a la izquierda de la TV en todo momento.

### **Inicio de sesión como empleado autenticado.**

Todo empleado que desee acceder a la intranet deberá introducir su email y su contraseña. Esta función creará una variable sesión (empleado) que será la que controle en todo momento que el empleado está registrado y se le permite el acceso a la intranet.

## **2.5 REQUERIMIENTOS DE EFICIENCIA**

Puesto que el fin de esta TV es la venta de artículos, se pretende esta tenga un acceso simultaneo de varios clientes al mismo tiempo. El servidor Apache en el que se alojará la TV tiene que ser capaz de proporcionar un acceso concurrente a un numero considerable de clientes. Desde el propio Apache se podrá configurar este comportamiento así como el numero máximo de conexiones simultaneas que deseamos. El rendimiento de la TV podría verse afectado directamente por el numero de clientes que se encuentren conectados al mismo tiempo. Por este motivo se desea asegurar un acceso optimo en condiciones de carga del servidor normales.

## **2.6 SEGURIDAD**

En primer lugar, ningún usuario podrá acceder a ninguna pagina de la TV sin haber pasado por el index o sin haberse registrado en las paginas en las que así se necesite. Cuando un usuario entra por primera vez en la TV se crean las variables necesarias para poder avanzar dentro de la pagina por lo que un acceso directo a otra no será permitido. Si un usuario quisiera entrar dentro de la intranet poniendo la dirección en la barra de direcciones se le denegará el acceso.

En segundo lugar, como en cualquier empresa, la información es privilegiada y por lo tanto debe almacenarse de una forma segura. La información de los clientes se guardará en la base de datos y las contraseñas de estos tendrán un formato cifrado. El paso en los que el usuario introduce la contraseña en la TV también se encuentra cifrado.

De esta forma se ha intentado reducir los riesgos al mínimo

## **2.7. MANTENIMIENTO**

La sencillez en el manejo de la TV permitirá a los propios empleados de la tienda a llevar un mantenimiento básico de la Web; gestión de usuarios, modificación del stock, etc ...

Sin embargo, la modificación en el diseño de la Web o cualquier modificación en la estructura de la base de datos tendrá que ser llevada a cabo por el administrador del portal.

## **2.8 OTROS REQUERIMIENTOS**

### **2.8.1 BASE DE DATOS**

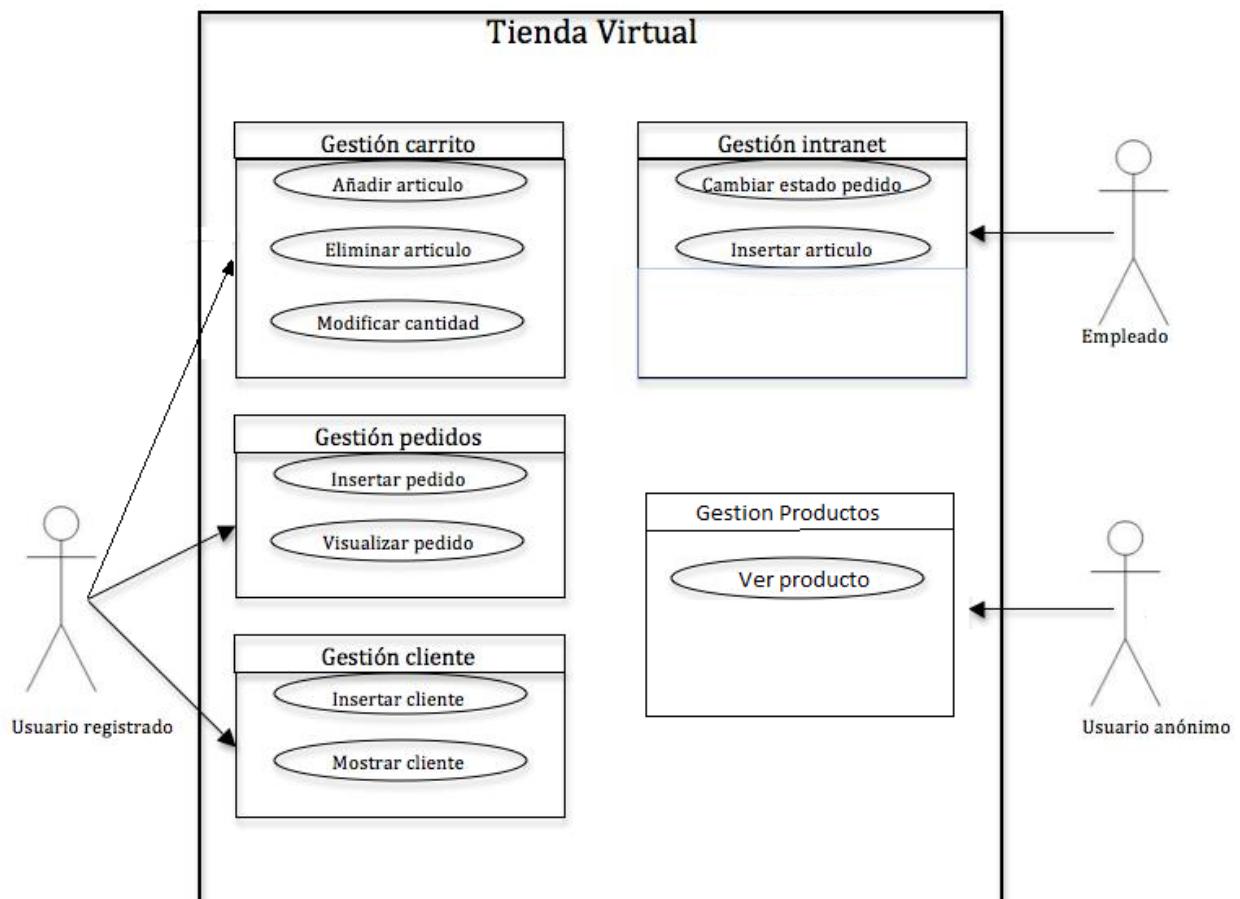
La Tienda web utilizará una base de datos en MySQL, la cual almacenará toda la información referente al catalogo, a los usuarios, y toda la información de los pedidos efectuados por los clientes.

Las consultas a la base de datos se realizarán por parte del servidor Web mediante PHP y su API de acceso a bases de datos MySQL.

# 3. ANÁLISIS.

## 3.1 CASOS DE USO.

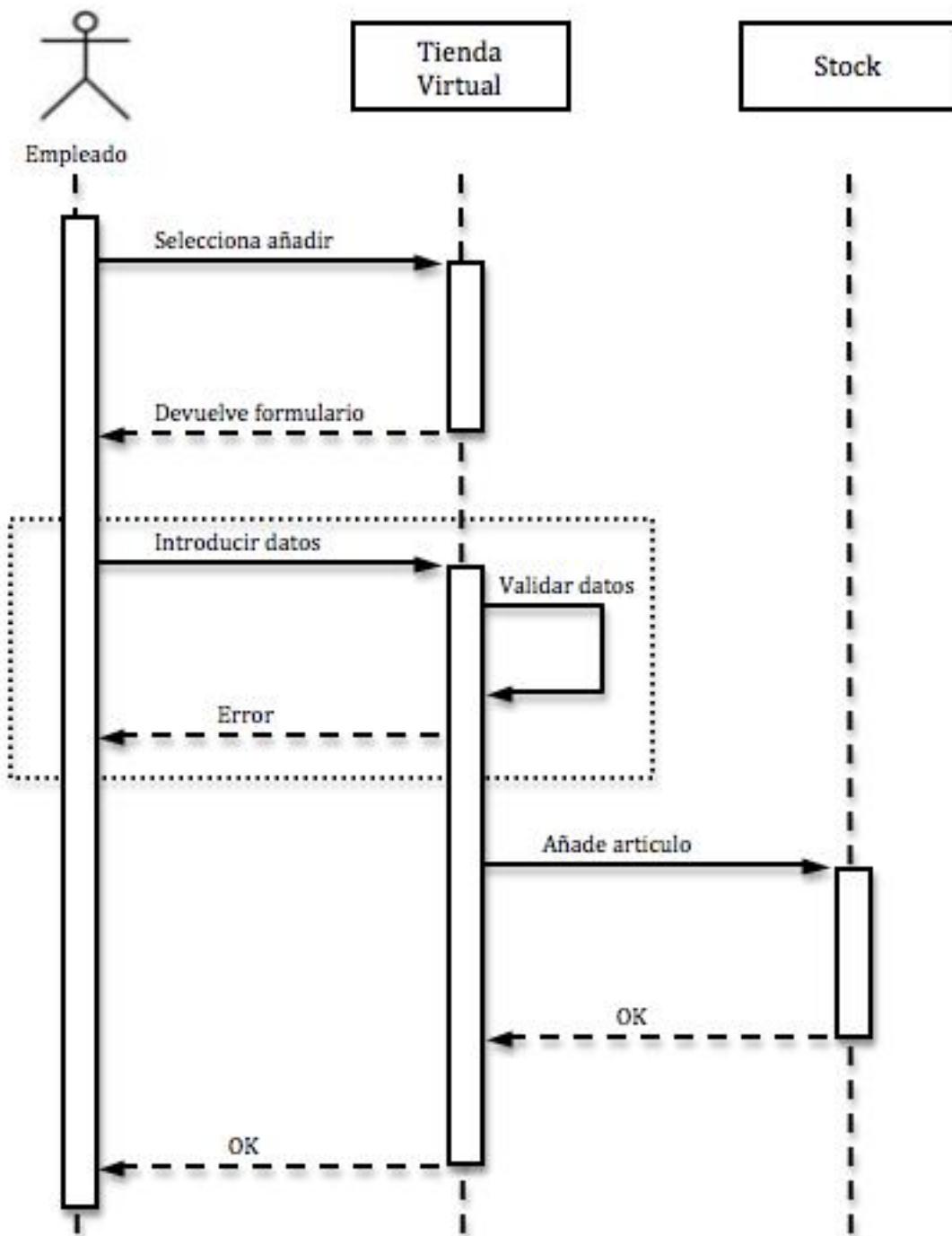
Para un entendimiento mas intuitivo de las acciones que se pueden realizar, al menos las mas importantes, se adjunta una figura resumiendo dichas funciones.



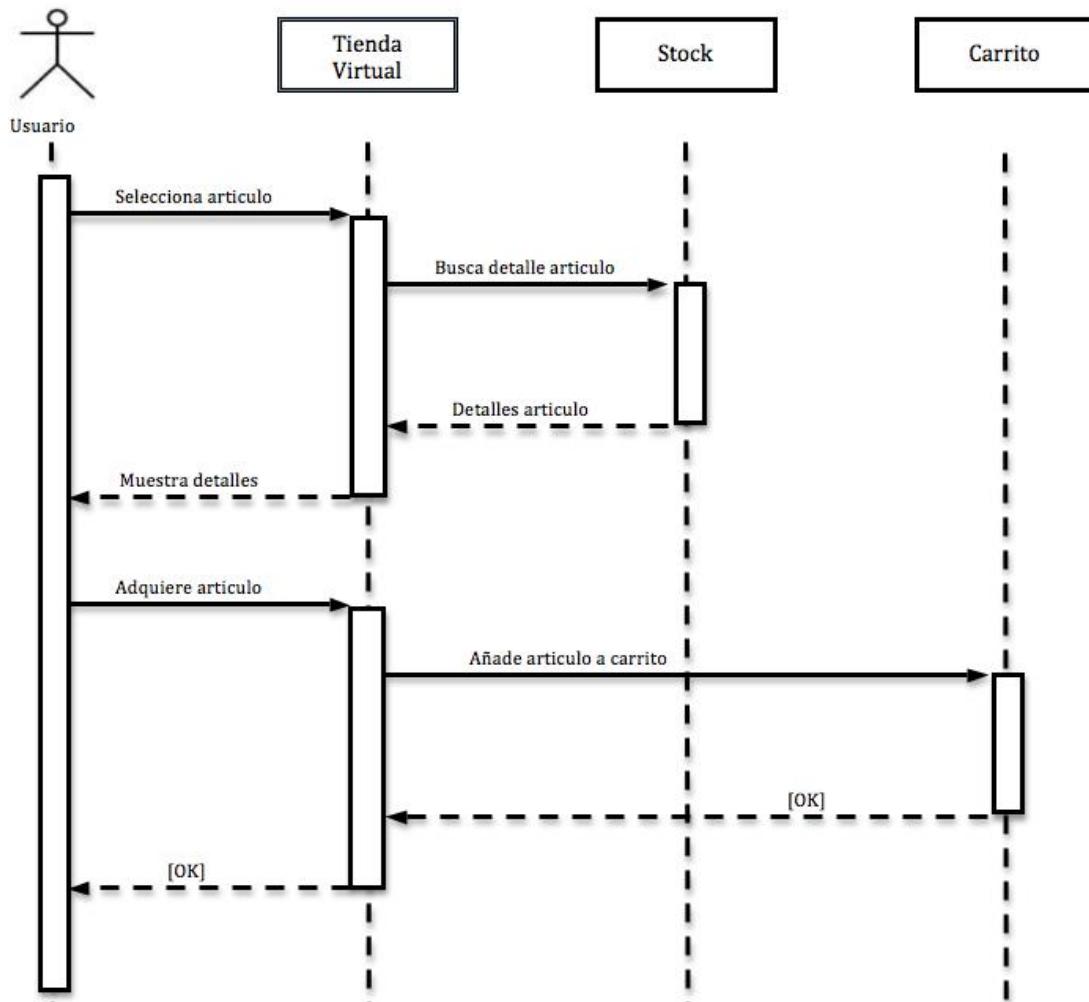
## 3.2 DIAGRAMA DE SECUENCIA

A continuación se muestran varios ejemplos de interacción con el sistema representados mediante unos diagramas de secuencia que reflejan a nivel de ejecución los pasos que sigue la aplicación para llevar a cabo las acciones indicadas arriba de cada ejemplo. Se muestra un ejemplo para el empleado o Administrador y otro para el usuario

### · Añadir articulo a la base de datos



## · Añadir articulo al carrito.



# 4. DISEÑO

El diseño de la Tienda Virtual se ha basado en una arquitectura multicapas de tres capas lógicas

- Nivel de presentación o de interfaz de usuario.
- Nivel lógico o de aplicación.
- Nivel de persistencia.

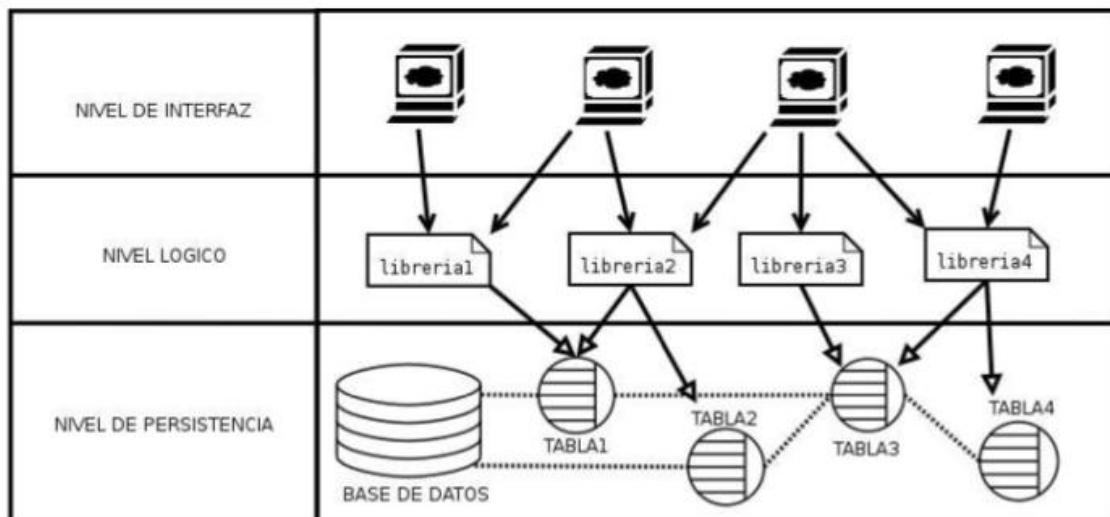


Figura 4.1 - Arquitectura multicapa

El nivel de interfaz está formado por todos los documentos que envía el servidor portal al navegador y que éste presenta al usuario, de forma que le proporcionan la información que ha solicitado acerca del portal y le permite interactuar con el mismo a través de enlaces y formularios

El nivel de aplicación o lógico está formado por un conjunto de librerías que implementan las clases del dominio. Este nivel es el encargado de realizar todas las operaciones a nivel de aplicación.

El nivel de persistencia lo forman la base de datos y el SGBD, los encargados de almacenar toda la información del portal y permitir el acceso a la misma de forma controlada y segura.

## 4.1 NIVEL DE INTERFAZ

En la siguiente imagen se puede observar una imagen general del diseño de la TV.



Como ya se comentó al principio de este documento, el interfaz de usuario ha sido diseñado de tal manera que resulte ameno, intuitivo y fácil de utilizar. De este modo, todas las paginas de la TV están compuestas por cuatro fases distinguibles fácilmente:

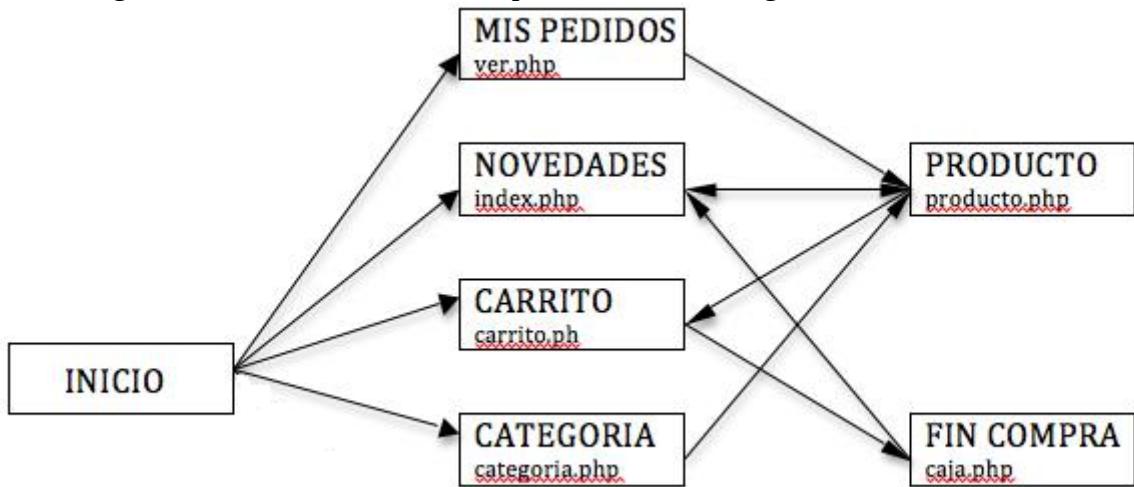
- Una cabecera en la que se muestra el logo y se deja un espacio para añadir publicidad.
- Un menú horizontal desde el que se puede acceder a todas las funciones de la pagina Web desde el que en determinadas funciones, podrá haber un submenú.
- La columna izquierda con la ventana del carrito en la que se muestra una información resumida de el contenido actual del carrito.
- La ventana principal en la que se muestra toda la información al usuario y se recogen los datos que este introduzca

## 4.1 DIAGRAMA DE NAVEGABILIDAD.

En este punto, se va a explicar mas detalladamente la estructura de la TV mostrando como acceder desde cada punto a cualquier parte de la Web. Además se incluirán los nombres de cada pagina para detallar de una manera mas sencilla la navegación posible que tendrá cada tipo de usuario.

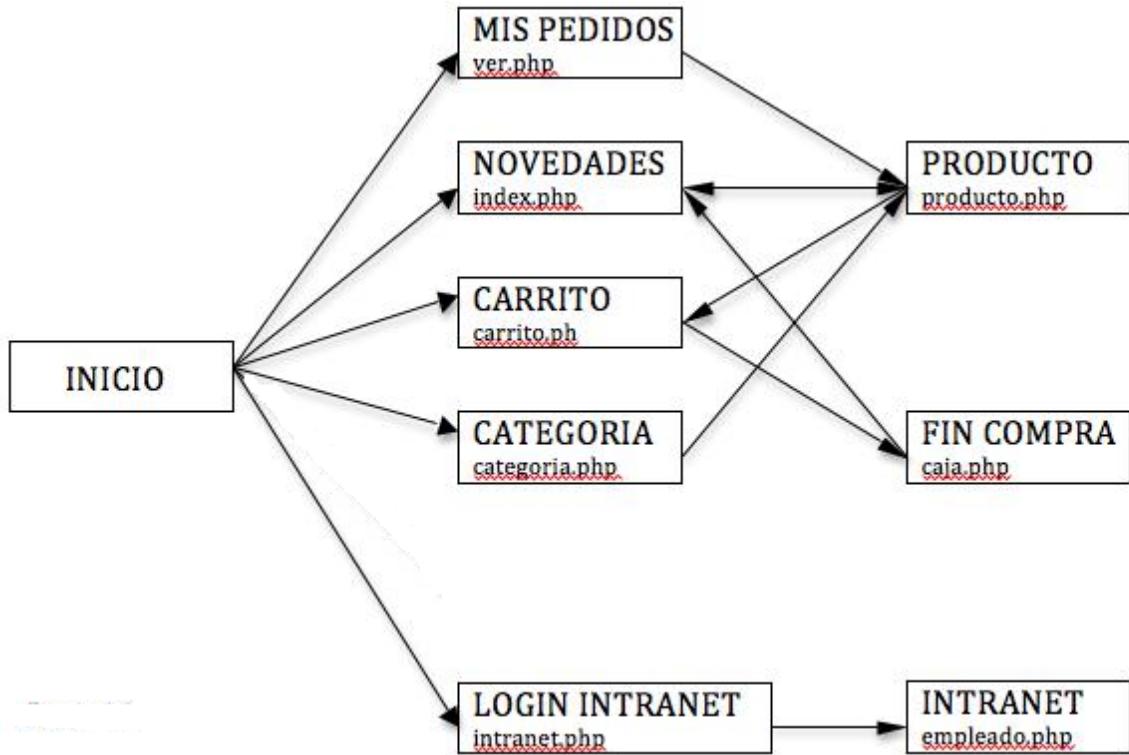
### 4.1.1 USUARIO REGISTRADO.

Esta tienda virtual se ha diseñado de tal manera que en las paginas en las que un usuario necesita estar registrado para ver información privilegiada, tendrá la opción de registrarse en caso de no haberlo hecho o de introducir sus datos de cliente. Por tanto, entendemos que existe diferencia entre la navegación del usuario anónimo respecto del usuario registrado .



#### 4.1.2 EMPLEADO/ADMINISTRADOR TIENDA VIRTUAL

Para el empleado, la navegación será la misma que para el usuario salvo dos excepciones. En primer lugar éste no podrá llegar hasta finalizar compra a no ser que se registre como usuario (caso en el que dejaría de ser empleado y pasaría a ser usuario registrado). En segundo lugar, tendrá el privilegio de poder acceder al menú Intranet como se muestra a continuación .



## **4.3 NIVEL LÓGICO.**

El nivel lógico o de aplicación consiste en una serie de librerías, gracias a las cuales la TV puede funcionar. Estas librerías contienen todas las clases necesarias para la ejecución de la Web. Sus funciones son: operaciones de calculo, comprobación de condiciones y niveles de acceso, generación de peticiones a la base de datos, transformación y validación de datos, etc.

Este nivel es el que nos permite una total independencia entre el nivel de interfaz y el nivel de persistencia (detallado en el siguiente punto). Esto quiere decir que sería posible realizar modificaciones en el nivel de persistencia sin que el de interfaz se vieran afectados. Únicamente habría que realizar pequeños cambios en el nivel lógico para adaptarse de nuevo al nivel que tiene por debajo.

## **4.4 NIVEL DE PERSISTENCIA**

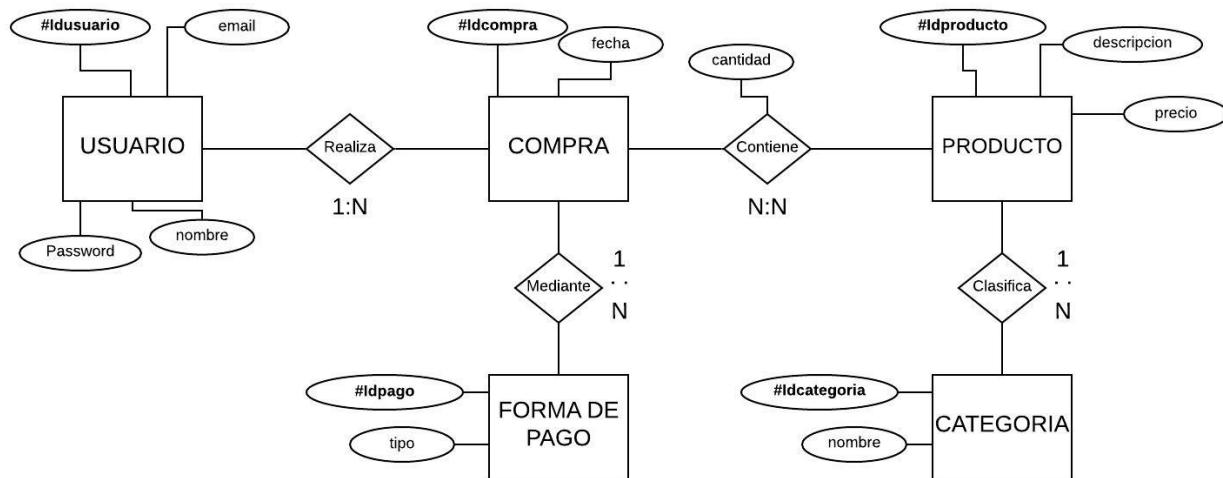
Este nivel de persistencia o de acceso de datos va a utilizar una base de datos relacional que será la que contenga toda la información de la TV (catalogo de ropa, información de usuarios, pedidos, etc.). Para el buen funcionamiento se utilizará una base de datos perfectamente estructurada y diseñada con una serie de entidades relacionadas entre sí de una manera coherente y con un cierto nivel de eficiencia.

La base de datos inicial contenía cuatro entidades: usuarios, pedidos, stock y tiendas. Tras realizar el modelo Entidad-Relación y efectuar varias pruebas de uso en la TV decidí separar usuarios en dos entidades diferentes (clientes y empleados) y añadir una nueva tabla (detallepedidos) que sería la que contendría toda la información detallada de los pedidos realizados. Esta tabla será una entidad débil respecto de Pedido. Por tanto, finalmente la base de datos contendrá las siguientes seis entidades: clientes, detallepedidos, empleados, pedidos, stock y tiendas.

Por otra parte, todas las entidades contendrán sus atributos. Uno de ellos será la clave primaria para todas las entidades salvo para detallepedidos que en este caso su clave primaria estará compuesta por dos atributos. Uno de ellos lo heredará de la tabla Pedido.

#### 4.4.1. DISEÑO ENTIDAD RELACIÓN

Este seria el diagrama Entidad Relación que realizamos inicialmente:



Como podemos observar es algo simple pero para empezar a construir nuestro sistema es mas que valido como veremos posteriormente en el diagrama relacional tanto el numero de entidades como de atributos ha ido incrementando en función de las necesidades del proyecto

#### 4.4.2. DISEÑO LÓGICO.

Con este diagrama entidad relación mencionado anteriormente ya podemos empezar a crear nuestras primeras tablas. Mostrare el procedimiento para introducir un par tablas con sus correspondientes Claves primarias (PK) y Claves foráneas (FK).

Creamos la base de datos denominada zapatos:

La captura de pantalla muestra la interfaz de MySQL Workbench. En la barra superior, se indica "Servidor: 127.0.0.1". La barra de menú tiene pestañas para "Bases de datos", "SQL", "Estado actual", "Usuarios" y "Más". En la sección central, hay un cuadro de texto que dice "Ejecute la o las consultas SQL en el servidor '127.0.0.1':". Abajo de ese cuadro, se ve una línea de comando SQL: "1 CREATE DATABASE zapatos;".

Creación: tabla **tblusuario**

The screenshot shows the MySQL Workbench interface. The title bar says "Servidor: 127.0.0.1". The tabs at the top are "Bases de datos", "SQL" (which is selected), "Estado actual", and "Usuarios". A message box in the center says "Ejecute la o las consultas SQL en el servidor \"127.0.0.1\"". Below the message box is the SQL code for creating the "usuario" table:

```
1 CREATE TABLE usuario
2 (
3     idusuario int PRIMARY KEY,
4     strNombre varchar(50),
5     strEmail varchar(50),
6     intActivo int,
7     strPassword varchar(20)
8 )
9 ;
```

Creación tabla **tblcompra**:

The screenshot shows the MySQL Workbench interface. The title bar says "Servidor: 127.0.0.1". The tabs at the top are "Bases de datos", "SQL" (which is selected), "Estado actual", and "Usuarios". A message box in the center says "Ejecute la o las consultas SQL en el servidor \"127.0.0.1\"". Below the message box is the SQL code for creating the "compra" table:

```
1 CREATE TABLE compra
2 (
3     idCompra int PRIMARY KEY,
4     idUsuario int,
5     fchCompra datetime,
6     intActivo int,
7     intTipoPago int,
8     dblPrecio double(6,2),
9     intEstado int
10 )
11 ;
```

Claves foráneas (FK):

(FK)idUsuario → (PK) idusuario

PD: Puede parecer en principio confuso que se llamen de una manera muy similar la PK y la FK ,pero en mi caso es completamente al revés a la hora de programar me es mucho mas sencillo dado que así se en todo momento que tipo de registro estoy usando a la hora de implementar código.

The screenshot shows the phpMyAdmin interface with the following details:

- Servidor: 127.0.0.1
- Bases de datos: (selected)
- SQL: (selected)
- Estado actual
- Usuarios
- Más

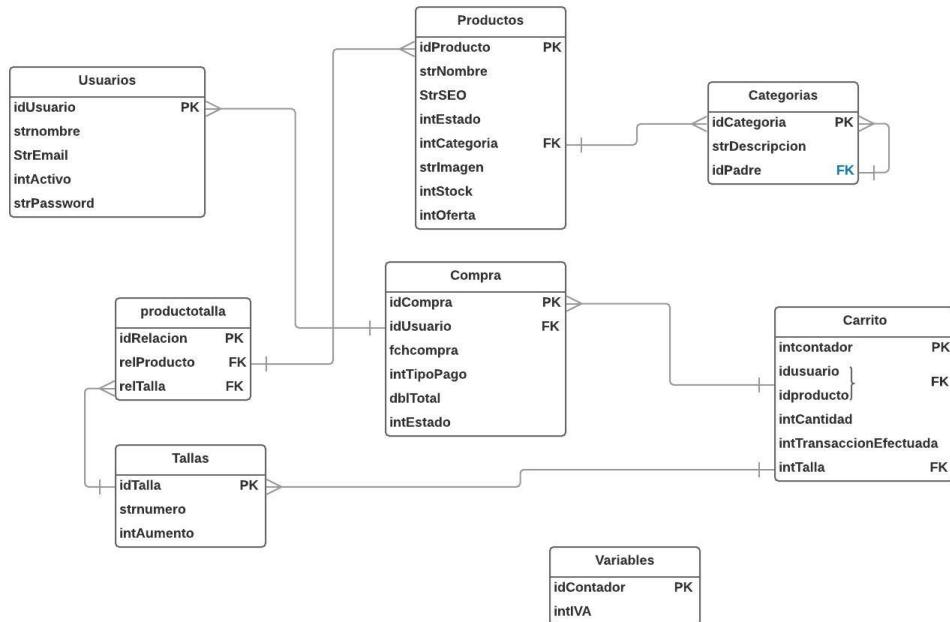
In the SQL tab, there is a text input field containing the following SQL code:

```
1 ALTER TABLE compra
2 ADD CONSTRAINT FK_idUsuario
3 FOREIGN KEY (idUsuario) REFERENCES usuario(idusuario);
```

Este proceso es similar para las demás tablas,aunque he de ser sincero y metí las 3 primeras tablas mediante código Mysql con sus correspondientes PK y sus FK el resto me servido de la interfaz grafica que ofrece phpMyAdmin, ya que considero esencial saber introducir tablas, modificar y borrar registros etc mediante comandos pero también nos hemos de servir de las facilidades que nos dan las aplicaciones para así ahorrar tiempo y ser mas eficientes en nuestro trabajo.

### 4.2.3 DIAGRAMA E-R.

ER TIENDA ZAPATOS



Este seria el diagrama completo que he utilizado para realizar el proyecto. A continuación veremos que tipos de datos hemos seleccionado para cada registro.

### 4.3.4 TIPO DE DATOS.

TABLA USUARIO → `tblusuario`

#	Nombre	Tipo
1	<code>idUsuario</code>	int(11)
2	<code>strNombre</code>	varchar(50)
3	<code>strEmail</code>	varchar(100)
4	<code>intActivo</code>	int(11)
5	<code>strPassword</code>	varchar(50)

NOTA: Me parece de gran ayuda añadir antes del campo del atributo al tipo de dato al cual pertenece por ejemplo strNombre el campo Nombre es un varchar así que añado str al inicio, esto ahorra bastante tiempo al no tener que estar continuamente consultando nuestra base de datos para averiguar que tipo de dato le asignamos al registro. De manera análoga para las tablas con `tbl`

## TABLA PRODUCTO → **tblproducto**

#	Nombre	Tipo
1	<b>idProducto</b>	int(11)
2	<b>strNombre</b>	varchar(100)
3	<b>strSEO</b>	varchar(100)
4	<b>dblPrecio</b>	double
5	<b>intEstado</b>	int(11)
6	<b>intCategoria</b>	int(11)
7	<b>strImagen</b>	varchar(50)
8	<b>intStock</b>	int(11)
9	<b>intOferta</b>	int(11)

## TABLA CATEGORÍA → **tblcategoria**

#	Nombre	Tipo
1	<b>idCategoria</b>	int(11)
2	<b>strDescripcion</b>	varchar(50)
3	<b>idPadre</b>	int(11)

El campo idPadre lo utilizaremos para las subcategorias , en el apartado de la creación de las categorías se explica mas detalladamente su funcionamiento

## TABLA COMPRA → **tblcompra**

#	Nombre	Tipo
1	<b>idCompra</b>	int(11)
2	<b>idUsuario</b>	int(11)
3	<b>fchCompra</b>	datetime
4	<b>intTipoPago</b>	int(11)
5	<b>dblTotal</b>	double
6	<b>intEstado</b>	int(11)

### TABLA CARRITO → tblcarrito

Nombre	Tipo
intContador	int(11)
idUsuario	int(11)
idProducto	int(11)
intCantidad	int(11)
intTransaccionEfectuada	int(11)
intTalla	int(11)

### TABLA TALLAS → tlbtallas

#	Nombre	Tipo
1	idTalla	int(11)
2	strNombre	varchar(10)
3	intAumento	int(11)

### TABLA PRODUCTOTALLA → tblproductotalla

#	Nombre	Tipo
1	idRelacion	int(11)
2	relProducto	int(11)
3	relTalla	int(11)

Esta tabla nos servirá para relacionar las tallas con los productos

### TABLA VARIABLES → tblvariables

#	Nombre	Tipo
1	idContador	int(11)
2	intIVA	int(11)

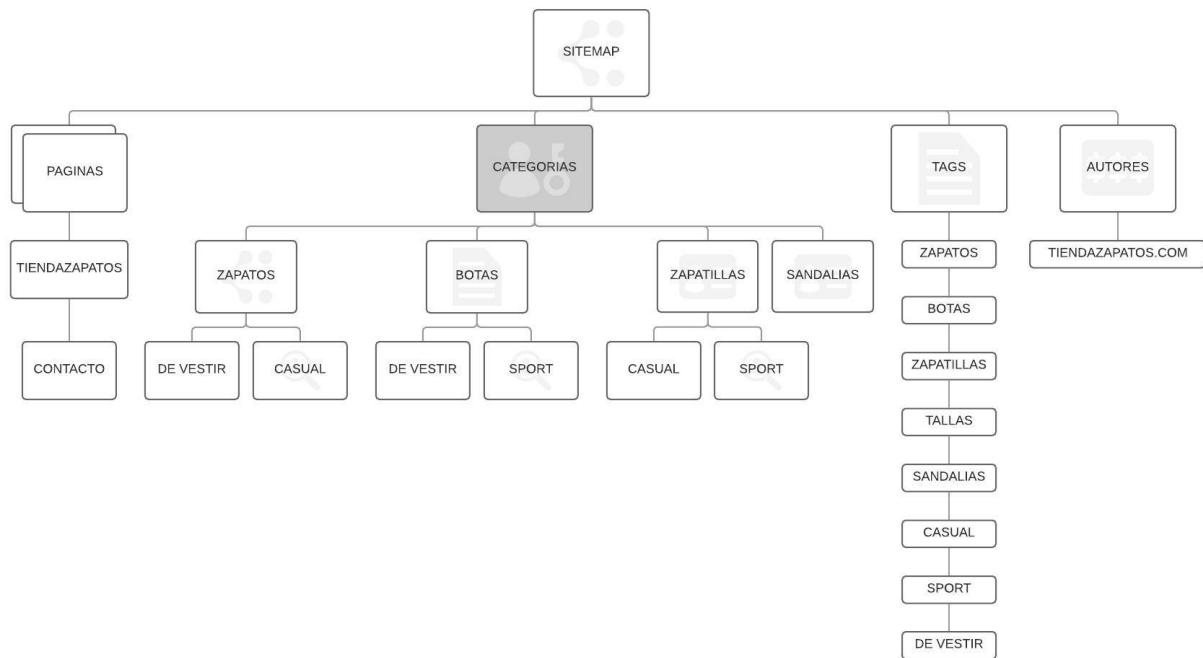
Esta tabla auxiliar servirá para el calculo del IVA en el carrito de la compra

# 5. MAPA DEL SITIO.

## MAPA SITIO TIENDAZAPATOS.COM

Texto

francisco marcet prieto | May 19, 2019



# **6. IMPLEMENTACIÓN E INTEGRACIÓN**

## **6.1 HERRAMIENTAS**

La TV es una aplicación Web formada por el portal Web en el lenguaje de programación PHP y una base de datos relacional en MySQL interconectadas entre si. Además, recordar que ambas se encuentran en una maquina con sistema operativo Windows 8 y un servidor apache.

Un entorno de desarrollo integrado (IDE de sus siglas en inglés) es un programa compuesto por un conjunto de herramientas para el programador. Los documentos de los cuales se compone la aplicación (HTML y scripts en PHP) se han desarrollado utilizando un IDE de programación llamado Sublime Text.

Sublime Text es un editor de código fuente bastante avanzado y extensible. No solo se limita a proveer de las diversas herramientas habituales de los editores populares para programadores, sino que va más allá, proporcionando algunas de las utilidades típicas de los entornos de desarrollo profesionales. Además, incorpora un sistema de complementos o add-ons similar al que se conoce por el navegador Firefox, que hace que todavía se pueda disfrutar de diversas otras utilidades que son de agradecer.

Las gestiones con la base de datos se han realizado con dos aplicaciones distintas. La creación de tablas y atributos así como gestiones simples se han realizado con phpMyAdmin. Esta es una herramienta escrita en PHP con la intención de manejar la administración de MySQL a través de páginas Web, utilizando Internet. Se pueden crear y eliminar Bases de Datos, crear, eliminar y alterar tablas, borrar, editar y añadir campos, ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios y exportar datos en varios formatos. Sin embargo, para introducir los datos que compondrán estas tablas, se ha utilizado la aplicación Xampp. Esta aplicación es libre para Windows cuyo fin es la gestión de bases de datos MySQL. Xampp proporciona todo lo necesario para conectarse a cualquier servidor MySQL ya sea local o remoto, una infinidad de opciones a la hora de consultar, insertar o eliminar datos, soportes de vistas, así como una gran lista de posibilidades.

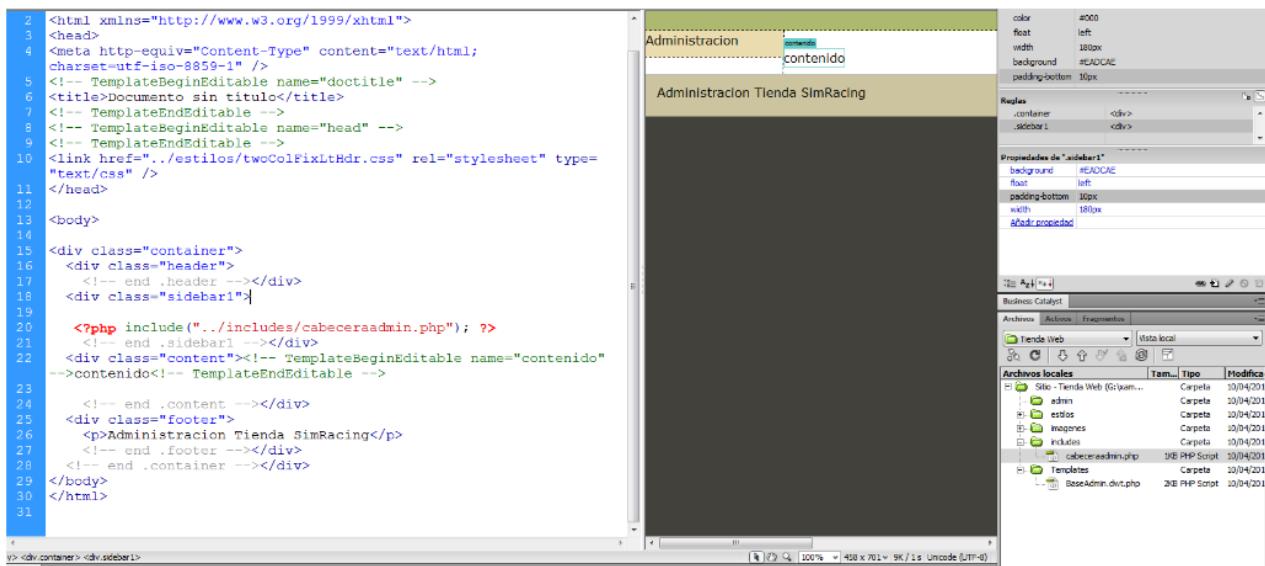
Por último, y en cuanto a diseño se refiere, Dreamviewer CS5 ha sido la herramienta con la que se han diseñado las plantillas que forman la estructura de la TV.

## 6.2 IMPLEMENTACIÓN DE LA ADMINISTRACIÓN(BACKEND).

### 6.2.1 PLANTILLA ADMINISTRACIÓN

#### BaseAdmin.dwt.php

En primer lugar hemos procedido a la creación de una plantilla la cual nos servirá para crear todas las páginas que componen la administración.



Una vez tenemos la plantilla añadimos mediante un include el archivo cabeceraadmin.php el cual nos gestionara el menú de la izquierda. De esta manera podremos añadir de una manera sencilla opciones en el menú.

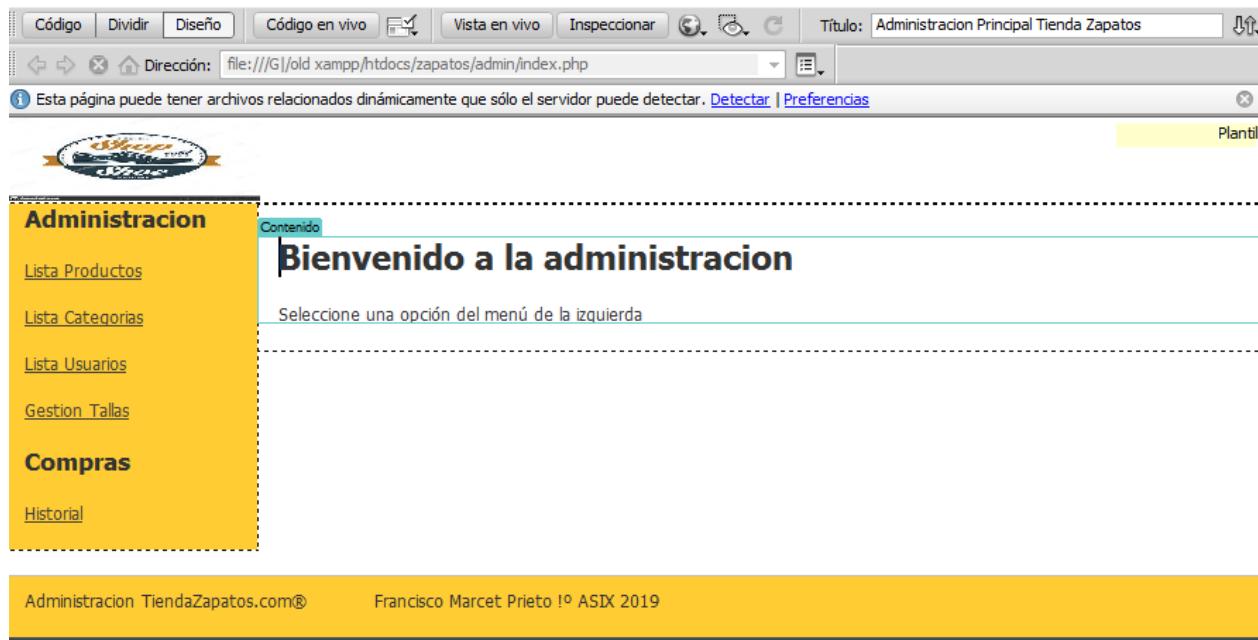
The screenshot shows a browser window displaying the generated menu from the template. The URL is 'file:///G:/old xampp/htdocs/zapatos/includes/cabeceraadmin.php'. The page content is a list of administrative links: 'Administracion', 'Lista Productos', 'Lista Categorias', 'Lista Usuarios', 'Gestion Tallas', 'Compras', 'Historial'. To the right, there is a sidebar with the title 'Administracion' and links: 'Lista Productos', 'Lista Categorias', 'Lista Usuarios', 'Gestion Tallas', 'Compras', 'Historial'. The sidebar has a dark background and light-colored text.

```
<h2>Administracion</h2>
<p> <a href="../admin/productos_lista.php">Lista Productos</a></p>
<p> <a href= "../admin/categorias_lista.php?recordID=0">Lista Categorias</a></p>
<p> <a href= "../admin/usuarios_lista.php">Lista Usuarios</a></p>
<p> <a href= "../admin/tallas_lista.php">Gestion Tallas</a></p>
<h2>Compras</h2>
<p> <a href= "../admin/compras_lista.php">Historial</a></p>
```

## 6.2.2 PAGINA PRINCIPAL ADMINISTRACIÓN.

### index.php

Será nuestra pagina principal de la parte de administración. o backend, en tas pagina no sera necesario realizar ningún tipo de consulta ya que simplemente sera la pagina indice



Este ya es un buen momento para conectar nuestro servidor local con nuestra base de datos aun que no vayamos a obtener ningún registro dado que no tenemos ningún valor introducido en ningún campo, pero nos servirá para comprobar el archivo de conexiones, y si los datos de estos están correctos.

Para ello vamos a implementar el archivo **conexionzapatos.php**, el cual establecerá la conexión con la base de datos alojada en el servidor local de apache

```
<?php if (!isset($_SESSION)) {  
    session_start();  
}?  
<?php  
# FileName="Connection_php_mysql.htm"  
# Type="MySQL"  
# HTTP="true"  
$hostnameConexionzapatos = "localhost";  
$databaseConexionzapatos = "zapatos";  
$usernameConexionzapatos = "root";  
$passwordConexionzapatos = "";  
$conexionzapatos = mysql_pconnect($hostnameConexionzapatos, $usernameConexionzapatos, $passwordConexionzapatos) or  
trigger_error(mysql_error(),E_USER_ERROR);  
?  
<?php  
if (is_file("includes/funciones.php")){  
    include("includes/funciones.php");  
}  
else  
{  
    include("../includes/funciones.php");  
}  
?>
```

Vamos a comentar algunos detalles relevantes.

En primer lugar mediante este bucle de control If nos ahorraremos tener que añadir en cada pagina la variable session\_start , siempre cuando tome el valor de \$\_SESSION

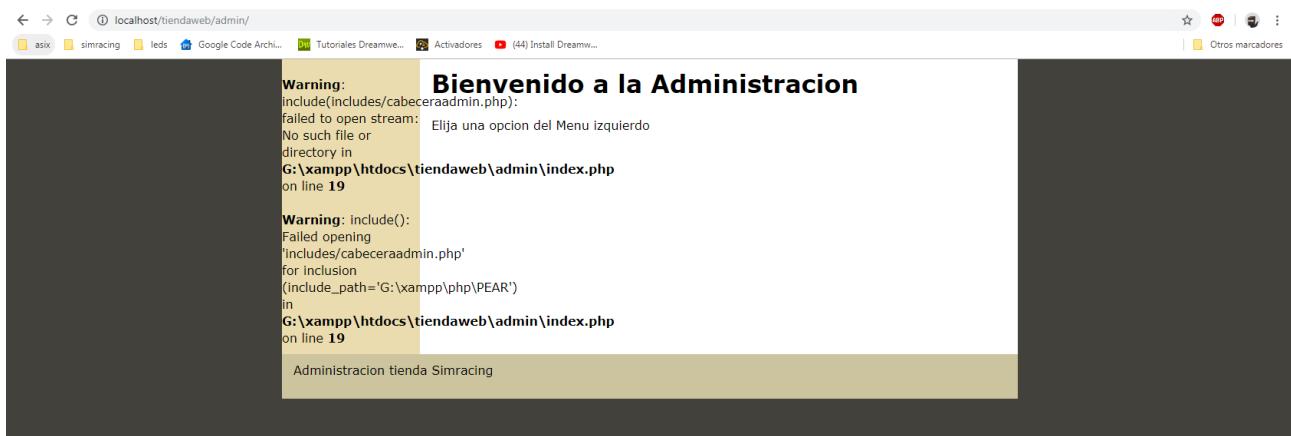
```
<?php if (!isset($_SESSION)) {  
    session_start();}  
?>  
<?php
```

En segundo lugar :

```
<?php  
if (is_file("includes/funciones.php")){  
    include("includes/funciones.php");  
}  
else  
{  
    include("../includes/funciones.php");  
}  
?>
```

Este bucle If nos permitirá usar nuestro archivo de funciones ,sin importar el directorio donde este alojado ya que la parte de la administración. todos los archivos están dentro de la carpeta admin y su ruta de acceso requiere de ../ mientras que los archivos que no pertenecen a la administración. no están alojados en ninguna carpeta y su ruta sera absoluta. Con esto conseguimos que nuestro

Dejando aparte el archivo de funciones el cual posteriormente comentaremos con detalle, vemos como tenemos conexión con el servidor local



## 6.2.3 LISTA DE PRODUCTOS

### listaproductos.php

Para visualizar la lista de productos vamos a crear una tabla en la cual se mostraran, para obtener los productos debemos realizar una consulta SELECT a la BD que obtenga dichos registros

```
<?php require_once('../Connections/conexionzapatos.php'); ?>
<?php
if (!function_exists("GetSQLValueString")) {
function GetSQLValueString($theValue, $theType, $theDefinedValue = "", $theNotDefinedValue = "") {
    if (PHP_VERSION < 6) {
        $theValue = get_magic_quotes_gpc() ? stripslashes($theValue) : $theValue;
    }

    $theValue = function_exists("mysql_real_escape_string") ? mysql_real_escape_string($theValue) : mysql_escape_string($theValue);

    switch ($theType) {
        case "text":
            $theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
            break;
        case "long":
        case "int":
            $theValue = ($theValue != "") ? intval($theValue) : "NULL";
            break;
        case "double":
            $theValue = ($theValue != "") ? doubleval($theValue) : "NULL";
            break;
        case "date":
            $theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
            break;
        case "defined":
            $theValue = ($theValue != "") ? $theDefinedValue : $theNotDefinedValue;
            break;
    }
    return $theValue;
}
}

$maxRows_Recordset1 = 10;
$pageNum_Recordset1 = 0;
if (isset($_GET['pageNum_Recordset1'])) {
    $pageNum_Recordset1 = $_GET['pageNum_Recordset1'];
}
$startRow_Recordset1 = $pageNum_Recordset1 * $maxRows_Recordset1;

mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_Recordset1 = "SELECT * FROM tblproducto ORDER BY strNombre ASC";
$query_limit_Recordset1 = sprintf("%s LIMIT %d, %d", $query_Recordset1, $startRow_Recordset1, $maxRows_Recordset1);
$Recordset1 = mysql_query($query_limit_Recordset1, $conexionzapatos) or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($Recordset1);
?>
```

En la primera linea de código vemos la función de php **require\_once** , la cual va a efectuar la conexión con la base de datos , esta linea de código sera necesaria en todas nuestros archivos de nuestra tienda web.

Esta sección nos realizara una paginación, es decir solo nos mostrara 10 objetos por pagina , este bucle If tendrá efecto sobre la consulta que ahora comentaremos

```
$maxRows_Recordset1 = 10;
$pageNum_Recordset1 = 0;
if (isset($_GET['pageNum_Recordset1'])) {
    $pageNum_Recordset1 = $_GET['pageNum_Recordset1'];
}
$startRow_Recordset1 = $pageNum_Recordset1 * $maxRows_Recordset1;
```

Finalmente tenemos la consulta para obtener los registros , mediante una SELECT de nuestra tabla productos ordenados por nombre y ascendente para obtener el listado ordenado

```
mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_Recordset1 = "SELECT * FROM tblproducto ORDER BY tblproducto.strNombre ASC";
$query_limit_Recordset1 = sprintf("%s LIMIT %d, %d", $query_Recordset1, $startRow_Recordset1, $maxRows_Recordset1);
$Recordset1 = mysql_query($query_limit_Recordset1, $conexionzapatos) or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($Recordset1);
```

PD el parámetro LIMIT %d lo utilizamos para la paginación que citamos anteriormente.

Utilizamos echo en los registros de la base de datos para identificarlos y mostrarlos por pantalla

```
<h1>Lista de Productos</h1>
<table width="100%" border="0" cellspacing="0" cellpadding="0">
<tr>
    <td>Nombre Producto</td>
    <td>Estado</td>
    <td>Acciones</td>
</tr>
<tr>
    <td><?php echo $row_Recordset1['strNombre']; ?></td>
    <td><?php echo $row_Recordset1['intEstado']; ?></td>
    <td>Editar - Eliminar</td>
</tr>
</table>
```

Ahora vamos a necesitar que estos registros se repitan para así poder visualizar la lista completa de productos, ya que con el “echo” anterior solo mostramos 1 registro. Podemos implementar esto mediante un bucle “Do-While” , que mostraremos a continuación:

```
<?php do { ?>
<tr>
    <td><?php echo $row_Recordset1['strNombre']; ?></td>
    <td><?php echo $row_Recordset1['intEstado']; ?></td>
    <td>Editar - Eliminar</td>
</tr>
<?php } while ($row_Recordset1 = mysql_fetch_assoc($Recordset1)); ?>
```

## 6.2.4 AÑADIR UN PRODUCTO

### productos\_add.php

Vamos a continuar creando la pagina para Añadir productos → productos add.php.

En la cual vamos a generar un formulario para la inserción de productos, el el cual realizaremos mediante una operación INSERT.

En principio no vamos a añadir al formulario el IdProducto o clave primaria de la tabla,ya que se trata de un valor auto incremental y no vamos nosotros a introducirlo

Añadiremos también al registro intEstado un valor igual a 1 cuando el usuario este activo y un valor igual a 0 cuando este inactivo, es decir cuando este este conectado al sistema y cuando no lo esté .

Mediante el código :

```
<td><select name="intEstado">
....<option value="1"><?php if (!(strcmp(1, ""))){echo "SELECTED";} ?>>Activo</option>
....<option value="0"><?php if (!(strcmp(0, ""))){echo "SELECTED";} ?>>Inactivo</option>
</select></td>
```

Con un bucle If podemos seleccionar el valor 0 o 1, en función de si el usuario esta activo o inactivo, con la función :

**strcmp(string1, string2)** , nos compara 2 strings y nos devuelve un valor

- 0 → Si los 2 strings son iguales
- <0 → Si string1 es menor que string2
- >0 → Si string1es mayor que string2

Devolverá el valor 0 o 1 en función de la opción seleccionada.

```
<form action=<?php echo $editFormAction; ?>" method="post" name="form1" id="form1">
<table align="center">
<tr valign="baseline">
  <td nowrap="nowrap" align="right">Nombre:</td>
  <td><input type="text" name="strNombre" value="" size="32" /></td>
</tr>
<tr valign="baseline">
  <td nowrap="nowrap" align="right">SEO:</td>
  <td><input type="text" name="strSEO" value="" size="32" /></td>
</tr>
<tr valign="baseline">
  <td nowrap="nowrap" align="right">Precio:</td>
  <td><input type="text" name="dbPrecio" value="" size="32" /></td>
</tr>
<tr valign="baseline">
  <td nowrap="nowrap" align="right">Estado:</td>
  <td><select name="intEstado">
    <option value="1"><?php if (!(strcmp(1, ""))){echo "SELECTED";} ?>>Activo</option>
    <option value="0"><?php if (!(strcmp(0, ""))){echo "SELECTED";} ?>>Inactivo</option>
  </select></td>
</tr>
<tr valign="baseline">
  <td nowrap="nowrap" align="right">Categoria:</td>
  <td><input type="text" name="intCategoria" value="" size="32" /></td>
</tr>
<tr valign="baseline">
  <td nowrap="nowrap" align="right">&ampnbsp</td>
  <td><input type="submit" value="Insertar registro" /></td>
</tr>
</table>
<input type="hidden" name="MM_insert" value="form1" />
</form>
```

Vamos a comentar mas detenidamente la operación INSERT, en la cual vamos a pasar los datos mediante parámetros.

Las consultas con parámetros son aquellas cuyas condiciones de búsqueda se definen mediante parámetros. Considero que programar mediante parámetros da una mayor flexibilidad , y en este caso en una tienda web en muchas ocasiones requerirán de parámetros. por ejemplo cuando hagamos clic en algún producto el sistema deberá reconocer a que producto nos referimos y esto lo conseguiremos mediante el paso de una variables por parámetro.

Mediante un bucle If con un **isset**, que determina si una variable está definida y toma dicho valor y no es NULL , esta devuelve el valor TRUE si la **variable** existe y tiene un valor distinto de NULL, FALSE de lo contrario.

Las variables se pasaran por método POST dado que provienen de un formulario , si cumple las condiciones anteriormente citadas se realizara el INSERT en la Base de datos .

```
$editFormAction = $_SERVER['PHP_SELF'];
if (isset($_SERVER['QUERY_STRING'])) {
    $editFormAction .= "?" . htmlentities($_SERVER['QUERY_STRING']);
}

if ((isset($_POST["MM_insert"])) && ($_POST["MM_insert"] == "form1")) {
    $insertSQL = sprintf("INSERT INTO tbproductos (strNombre, strSEO, dbPrecio, intEstado, intCategoria) VALUES (%s, %s, %s, %s, %s",
        GetSQLValueString($_POST['strNombre'], "text"),
        GetSQLValueString($_POST['strSEO'], "text"),
        GetSQLValueString($_POST['dbPrecio'], "double"),
        GetSQLValueString($_POST['intEstado'], "int"),
        GetSQLValueString($_POST['intCategoria'], "int"));

    mysql_select_db($databaseConexionTienda, $conexionTienda);
    $Result1 = mysql_query($insertSQL, $conexionTienda) or die(mysql_error());

    $insertGoTo = "productos_list.php";
    if (isset($_SERVER['QUERY_STRING'])) {
        $insertGoTo .= (strpos($insertGoTo, '?')) ? "&" : "?";
        $insertGoTo .= $_SERVER['QUERY_STRING'];
    }
    header(sprintf("Location: %s", $insertGoTo));
}
```

**GetSQLValueString** → Formatea la variable para que se introduzca correctamente en la BD

Vamos a realizar alguna modificación el campo categoría para poder seleccionar las diferentes categorías en el formulario para añadir productos mediante una consulta SELECT a la base de datos.

Modificamos el nombre de la conexión para no confundir con el anterior denominado Recordset1 a este nuevo le llamaremos ConsultaCategorias

```
mysql_select_db($database_ConexionZapatos, $conexionZapatos);
$query_ConsultaCategorias = "SELECT * FROM tblcategoria ORDER BY tblcategoria.strDescripcion ASC";
$ConsultaCategorias = mysql_query($query_ConsultaCategorias, $conexionZapatos) or die(mysql_error());
$row_ConsultaCategorias = mysql_fetch_assoc($ConsultaCategorias);
$totalRows_ConsultaCategorias = mysql_num_rows($ConsultaCategorias);
```

Y de nuevo hacemos un menú desplegable mediante html para poder seleccionar las categorías y de nuevo implementamos un bucle de control Do-While para que nos muestren siempre todas las categorías existentes

```
<tr valign="baseline">
..... <td nowrap="nowrap" align="right">Categoria:</td>
..... <td><select name="">
..... <?php
do {
?>
..... <option value=<?php echo $row_ConsultaCategorias['idCategoria']?>><?php echo $row_ConsultaCategorias['strDescripcion']?></
..... option>
..... <?php
} while ($row_ConsultaCategorias = mysql_fetch_assoc($ConsultaCategorias));
$rows = mysql_num_rows($ConsultaCategorias);
if($rows > 0) {
..... mysql_data_seek($ConsultaCategorias, 0);
$row_ConsultaCategorias = mysql_fetch_assoc($ConsultaCategorias);
}
?>
```

Una vez tenemos en el formulario de insertar categorías en la sección de añadir un producto

The screenshot shows a Microsoft Access form with the following fields and their values:

- Nombre: {DatosProducto.strNombre}
- SEO: {DatosProducto.strSEO}
- Precio: {DatosProducto.dblPrecio}
- Estado: Inactivo
- Categoría: (dropdown menu)

Con esto concluido vamos con las categorías.

## 6.2.5 CATEGORÍAS.

### categorías\_lista.php

Para añadir las categorías en nuestro proyecto de la tienda de zapatos añadimos el enlace en la cabeceraadmin.php, recuerdo era donde añadíamos todas las secciones del menú del parte de administración del sistema

```
1 <h2>Administracion</h2>
2 <p> <a href="../admin/productos_lista.php">
3   Lista Productos</a></p>
4 <p> <a href=
5   "../admin/categorias_lista.php?recordID=0">
6   Lista Categorias</a></p>
7 <p> <a href="../admin/usuarios_lista.php">
8   Lista Usuarios</a></p>
9 <p> <a href="../admin/tallas_lista.php">
10  Gestión Tallas</a></p>
11 <h2>Compras</h2>
12 <p> <a href="../admin/compras_lista.php">
13  Historial</a></p>
```

### Administracion

[Lista Productos](#)

[Lista Categorias](#)

[Lista Usuarios](#)

[Gestion Tallas](#)

### Compras

[Historial](#)

Al haber creado una plantilla nos hacemos servir de ella así que mediante un copia y pega de index.php(admin) la cual modificaremos y la llamaremos categorías\_lista.php ahora realizaremos una consulta a la base de datos para extraer la lista de categorías de manera idéntica a como hicimos con la lista productos

El proceso es idéntico a como realizamos con la lista de productos

Insertamos una tabla y posteriormente hacemos que los registros se repitan con un bucle de control Do-while mostramos todas las categorías existentes en la misma pagina

```
<h1>Lista de Categorias</h1>
<p><a href="categorias_add.php">Añadir Categoria</a></p>
<table width="100%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td bgcolor="#FF9900">Nombre de Categoria</td>
    <td bgcolor="#FF9900">Acciones</td>
  </tr>
  <?php do { ?>
  <tr>
    <td><?php echo $row_Recordset1['strDescripcion']; ?></td>
    <td>Editar</td>
  </tr>
  <?php } while ($row_Recordset1 = mysql_fetch_assoc($Recordset1)); ?>
</table>
```

En este punto cabe destacar la función **mysql\_fetch\_assoc()**, ya que la considero interesante dada su utilidad .

Su cometido principal es recuperar una fila de resultados como un array asociativo, es decir devuelve un array asociativo que corresponde a la fila recuperada y mueve el puntero de datos interno hacia adelante únicamente devuelve un array asociativo.

Aclaremos de que se trata cuando hablamos de un array asociativo, son simplemente un array cuyos keys son strings personalizados

Su parámetro esta evaluado por el resultado . Este resultado que está siendo evaluado proviene de una llamada a mysql\_query(), que contendrá el parámetro obtenido de la consulta como resultado. Por ultimo como valores devueltos, obtendremos un array asociativo de strings que corresponde con la fila recuperada o FALSE si no hay mas filas disponibles.

Siento si me extiendo demasiado en algunas explicaciones pero me parece completamente indispensable comentar el uso de las funciones interviniéntes en el código implementado.

Continuamos con la consulta SELECT para obtener las categorías de nuestra base de datos , y junto a los echo que añadimos obtendremos los registros seleccionados

```
<?php require_once('../Connections/conexionzapatos.php'); ?>
<?php
if (!function_exists("GetSQLValueString")) {
function GetSQLValueString($theValue, $theType, $theDefinedValue = "", $theNotDefinedValue = "") {
{
    if (PHP_VERSION < 6) {
        $theValue = get_magic_quotes_gpc() ? stripslashes($theValue) : $theValue;
    }

    $theValue = function_exists("mysql_real_escape_string") ? mysql_real_escape_string($theValue) : mysql_escape_string($theValue);

    switch ($theType) {
        case "text":
            $theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
            break;
        case "long":
        case "int":
            $theValue = ($theValue != "") ? intval($theValue) : "NULL";
            break;
        case "double":
            $theValue = ($theValue != "") ? doubleval($theValue) : "NULL";
            break;
        case "date":
            $theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
            break;
        case "defined":
            $theValue = ($theValue != "") ? $theDefinedValue : $theNotDefinedValue;
            break;
    }
    return $theValue;
}
}

mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_Recordset1 = "SELECT * FROM tblcategoria ORDER BY tblcategoria.strDescripcion ASC";
$Recordset1 = mysql_query($query_Recordset1, $conexionzapatos) or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($Recordset1);
$totalRows_Recordset1 = mysql_num_rows($Recordset1);
?>
```

Añadimos de nuevo un bucle de control Do-While para mostrar todas las categorías

```
<h1>Lista de Categorías</h1>
<p><a href="categorias_add.php">Añadir Categoría</a></p>
<table width="100%" border="0" cellspacing="0" cellpadding="0">
<tr>
    <td bgcolor="#FF9900">Nombre de Categoría</td>
    <td bgcolor="#FF9900">Acciones</td>
</tr>
<?php do { ?>
<tr>
    <td><?php echo $row_Recordset1['strDescripcion']; ?></td>
    <td>Editar</td>
</tr>
<?php } while ($row_Recordset1 = mysql_fetch_assoc($Recordset1)); ?>
</table>
```

## 6.2.6 AÑADIR CATEGORÍA

### categorías\_add.php

Vinculamos el enlace Añadir categorías a nueva pagina categorías\_add.php .Creamos la pagina de manera idéntica a las anteriores copiando index.php y modificarlo a nuestros requisitos del sistema e insertamos un formulario para poder añadir las categorías con bucle Do-While para poder visualizar todas las categorías, proceso idéntico a lo realizado anteriormente.

La particularidad de añadir una categoría es que realizaremos una operación INSERT en nuestra base de datos

```
$editFormAction = $_SERVER['PHP_SELF'];
if (isset($_SERVER['QUERY_STRING'])) {
    $editFormAction .= "?" . htmlentities($_SERVER['QUERY_STRING']);
}

if ((isset($_POST["MM_insert"])) && ($_POST["MM_insert"] == "form1")) {
    $insertSQL = sprintf("INSERT INTO tblcategoria (idPadre, strDescripcion) VALUES (%s, %s)",
        GetSQLValueString($_POST['idPadre'], "text"),
        GetSQLValueString($_POST['strDescripcion'], "text"));

    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $Result1 = mysql_query($insertSQL, $conexionzapatos) or die(mysql_error());

    $insertGoTo = "categorias_lista.php";
    if (isset($_SERVER['QUERY_STRING'])) {
        $insertGoTo .= (strpos($insertGoTo, '?')) ? "&" : "?";
        $insertGoTo .= $_SERVER['QUERY_STRING'];
    }
    header(sprintf("Location: %s", $insertGoTo));
}
```

PD no tengas ahora en cuenta el atributo intPadre el cual lo utilizare para crear las subcategorías y posteriormente explicaremos con detalle.

En esta INSERT podemos comentar varios aspectos, en primer lugar el bucle If que sera el encargado de decidir si se ejecuta el INSERT o no, Este ultimo se ejecutara cuando se cumpla que tome el valor por método post → \$\_POST de la variable que le envía el formulario MM\_insert y ademas este sea igual al identificador del formulario , veamos esto :

```
<form action="<?php echo $editFormAction; ?>" method="post" name="form1" id="form1">
<table align="center">
```

```
if ((isset($_POST["MM_insert"])) && ($_POST["MM_insert"] == "form1")) {
$insertSQL = sprintf("INSERT INTO tblcategoria (idPadre, strDescripcion) VALUES (%s, %s)",
GetSQLValueString($_POST['idPadre'], "text"),
GetSQLValueString($_POST['strDescripcion'], "text"));
```

En cuanto a la INSERT , introducimos el registro descripción pasando por parámetro la variable, si la insert se ejecuta con éxito almacena el valor en \$result1 y la variable \$insertGoTo nos redireccionará a categorías\_list.php mediante el header se obtiene el valor de la INSERT por pantalla.

Podemos ver como funciona y nos va introduciendo las diferentes categorías

Administracion	Lista de Categorías	
	Añadir Categoría	Acciones
Listar Productos	Nombre de Categoría	
	Botas	<a href="#">Editar</a>
	Zapatillas	<a href="#">Editar</a>
	Zapatos	<a href="#">Editar</a>
	Zapatos de tacón	<a href="#">Editar</a>
Lista Categorias		
Lista Usuarios		
Administracion Tienda Zapatos		

## 6.2.7 EDITAR CATEGORÍAS.

### categorías\_edit.php

Por requerimientos del sistema necesitamos poder modificar o editar una categorías en el caso que exista un error .Vamos a introducir conscientemente una categoría errónea por ejemplo “Botoxadjkak” cuando realmente debería ser “Botas”

Nombre de Categoría	Acciones
Botas	<a href="#">Editar</a>
Botosxdhrd	<a href="#">Editar</a>
Zapatillas	<a href="#">Editar</a>
Zapatos	<a href="#">Editar</a>
Zapatos de tacón	<a href="#">Editar</a>

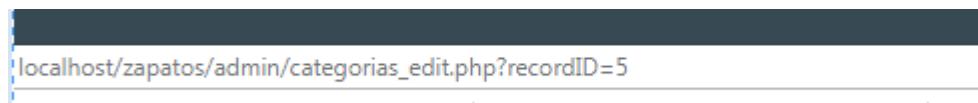
Veamos como solucionar este problema. La solución a mi parecer mas sencilla y eficiente es crear un enlace a una nueva pagina llamada categorías\_edit.php ,pero existe una dificultad añadida es como hacerle saber al sistema que categoría en concreto queremos modificar para ello vamos a insertar un parámetro para ello vamos a nombrarlo como ?recordID , lo vamos a referenciar mediante un ID y lo asociamos al identificador de categoría idCategoria mediante un echo.

Donde

```
<td><a href="categorias_edit.php?recordID=<?php echo $row_Recordset1['idCategoria']; ?>">Editar</a></td>
```

Mediante esta linea de código ya nos dejara seleccionar las categorías y el navegador sabrá cual estamos seleccionando

Podemos ver como en la parte inferior izquierda ya nos asigna un ID a cada categoría



Ahora si que podemos crear la pagina categorías\_edit.php copiando la pagina index.php y modificarla, bien una vez llegado hasta aquí necesitamos recuperar el valor que queremos editar y posteriormente modificarlo para ello realizamos una consulta a la base de datos de la tabla categorías donde vamos a obtener todos los valores asociados mediante una SELECT donde el campo idcategoria sea la que hemos seleccionado.

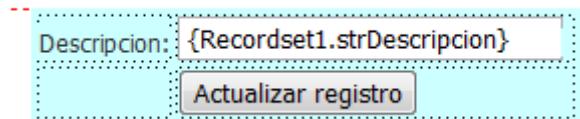
Mediante \$\_GET["recordID"] esto nos pasara un parámetro por recordID y de esta manera lo podremos recuperar .También se podría haber efectuado mediante un POST pero me pareció mas sencillo esto

Veamos esto como se implementaría mediante código :

```
$varCategoria_Recordset1 = "0";
if (isset($_GET["recordID"])) {
    $varCategoria_Recordset1 = $_GET["recordID"];
}
mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_Recordset1 = sprintf("SELECT * FROM tblcategoria WHERE tblcategoria.idCategoria = %s",
    GetSQLValueString($varCategoria_Recordset1, "int"));

$Recordset1 = mysql_query($query_Recordset1, $conexionzapatos) or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($Recordset1);
$totalRows_Recordset1 = mysql_num_rows($Recordset1);
```

Para editar las categorías vamos a necesitar una pequeña tabla en la que el contenido sera la descripción :



Donde el campo que nos mostrara por pantalla sera la descripción

```
| $row_Recordset1['strDescripcion'] |
```

Quedaría el código así :

```
<h1>Editar Categoría</h1>
<p>&nbsp;</p>
<form action=<?php echo $editFormAction; ?>" method="post" name="form1" id="form1">
    <table align="center">
        <tr valign="baseline">
            <td nowrap="nowrap" align="right">Descripción:</td>
            <td><span id="sprytextfield1">
                <input type="text" name="strDescripcion"
                    value=<?php echo htmlspecialchars($row_Recordset1['strDescripcion'], ENT_COMPAT, 'iso-8859-1'); ?>" size="32" />
                <span class="textfieldRequiredMsg">Se necesita un valor.</span></span></td>
            </tr>
```

Como vemos en value tomara el valor de descripción mencionado anteriormente

Como ultimo paso y no menos importante nos queda realizar un UPDATE a nuestra base de datos para que actualice el nuevo registro

```

if ((isset($_POST["MM_update"])) && ($_POST["MM_update"] == "form1")) {
    $updateSQL = sprintf("UPDATE tblcategoria SET strDescripcion=%s WHERE idCategoria=%s",
        GetSQLValueString($_POST['strDescripcion'], "text"),
        GetSQLValueString($_POST['idCategoria'], "int"));

    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $Result1 = mysql_query($updateSQL, $conexionzapatos) or die(mysql_error());

    $updateGoTo = "categorias_lista.php";
    if (isset($_SERVER['QUERY_STRING'])) {
        $updateGoTo .= (strpos($updateGoTo, '?')) ? "&" : "?";
        $updateGoTo .= $_SERVER['QUERY_STRING'];
    }
    header(sprintf("Location: %s", $updateGoTo));
}

```

Esta operación Update funciona de manera análoga a la Insert que comentamos en el apartado anterior quitando el hecho claramente que una inserta valores y otra actualiza pero el funcionamiento en cuanto a variables y parámetros es idéntica así que no me extenderé en este aspecto.

Ya podríamos supuestamente poder modificar la categoría que introducimos antes erróneamente “Botosxdhrc” y editarla correctamente

Administración

[Lista Productos](#)

[Lista Categorías](#)

[Lista Usuarios](#)

## Editar Categoría

Descripción:

Podemos comprobar como se ha modificado correctamente

Lista de Categorías		
Añadir Categoría		
Nombre de Categoría	Acciones	
Botas	<a href="#">Editar</a>	
Botas de Campo	<a href="#">Editar</a>	
Zapatillas	<a href="#">Editar</a>	
Zapatos	<a href="#">Editar</a>	
Zapatos de tacón	<a href="#">Editar</a>	

Administración Tienda Zapatos

Ya tenemos una administración de las categorías probemos ahora a introducir un producto por ejemplo un zapato

## Añadir Producto

Nombre:	<input type="text" value="zapato amarillo"/>	*
SEO:	<input type="text" value="zapato - amarillo"/>	*
Precio:	<input type="text" value="30"/>	*
Imagen:	<input type="text"/>	<input type="button" value="Subir Imagen"/>
Estado:	<input type="button" value="Activo"/>	
Categoría:	<input type="button" value="Zapatos"/>	
<input type="button" value="Insertar Producto"/>		

Administracion Tienda Zapatos

## Lista de Productos

[Añadir Producto](#)

NOMBRE PRODUCTO	ESTADO	ACCIONES
Bota de Agua	1	<a href="#">Editar</a> - <a href="#">Eliminar</a>
Tacon Economico	1	<a href="#">Editar</a> - <a href="#">Eliminar</a>
<b>zapato amarillo</b>	<b>1</b>	<a href="#">Editar</a> - <a href="#">Eliminar</a>
Zapato azul	1	<a href="#">Editar</a> - <a href="#">Eliminar</a>
Zapato standard	1	<a href="#">Editar</a> - <a href="#">Eliminar</a>
Zapato Tacon	1	<a href="#">Editar</a> - <a href="#">Eliminar</a>

Funciona correctamente

### 6.2.8 AÑADIR IMAGEN A PRODUCTOS

Veamos como podemos añadir una imagen del producto consultando, por internet hay un millón de formas de subir una imagen ,yo he elegido esta viendo que no requiere una gran complejidad aunque he de decir que ha sido la parte mas complicada hasta el momento.

El primer lugar vamos a necesitar un campo donde nos guarde el nombre del fichero para ello añadimos un campo en la tabla productos que se llamara **strImagen** ademas necesitaremos en el formulario de añadir productos un nuevo campo , al añadir este campo necesitamos modificar la operación INSERT que realizamos anteriormente y añadirle los nuevos campos

```
<td><label for="strImagen"></label>
    <input type="text" name="strImagen" id="strImagen" /></td>
</tr>
```

```

if ((isset($_POST["MM_insert"])) && ($_POST["MM_insert"] == "form1")) {
$insertSQL = sprintf("INSERT INTO tblproducto (strNombre, strSEO, dblPrecio, intEstado, intCategoria, strImagen,
intStock) VALUES (%s, %s, %s, %s, %s, %s, %s)",

GetSQLValueString($_POST['strNombre'], "text"),
GetSQLValueString($_POST['strSEO'], "text"),
GetSQLValueString($_POST['dblPrecio'], "double"),
GetSQLValueString($_POST['intEstado'], "int"),
GetSQLValueString($_POST['intCategoria'], "int"),
GetSQLValueString($_POST['strImagen'], "text"),
GetSQLValueString($_POST['intStock'], "int"));

mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$result1 = mysql_query($insertSQL, $conexionzapatos) or die(mysql_error());

$insertGoTo = "productos_lista.php";
if (isset($_SERVER['QUERY_STRING'])) {
$insertGoTo .= (strpos($insertGoTo, '?')) ? "&" : "?";
$insertGoTo .= $_SERVER['QUERY_STRING'];
}
header(sprintf("Location: %s", $insertGoTo));

```

Por lo que he podido investigar recomiendan guardar las imágenes en una carpeta del servidor en vez de en la propia base de datos ya que esto nos dará una mayor respuesta del sistema pues no tendremos que guardar imágenes en formato binario en nuestra BD.

Para ello creamos una nueva carpeta en nuestro Sitio llamada documentos y dentro de esta una llamada productos así podremos introducir toda la información que deseemos como documentos pdf etc

Ahora necesitamos crear un botón nuevo para poder subir las imágenes para ello un input type=button ya que no necesitamos que envié ninguna información al formulario → input type=submit. Y junto con un sencillo JavaScript que cuando realicemos clic sobre el botón nos abra una ventana que se encargue la gestión de la imagen .

Con un onclick generaremos un evento al pulsar el ratón encima seria simplemente:

```

<td><label for="strImagen"></label>
    <input type="text" name="strImagen" id="strImagen" />
    <input type="button" name="button" id="button" value="Subir Imagen" onclick="javascript:subirimagen();"/>
</td>

```

ahora vamos a generar el código para que nos abra dicha ventana y probar si funciona correctamente

```

<script>
function subirimagen()
{
self.name = 'opener';
remote = open('gestionImagen.php', 'remote',
width=400,height=150,location=no,scrollbars=yes,menubars=no,toolbars=no,resizable=yes,fullscreen=no, status=yes');
remote.focus();
}

</script>

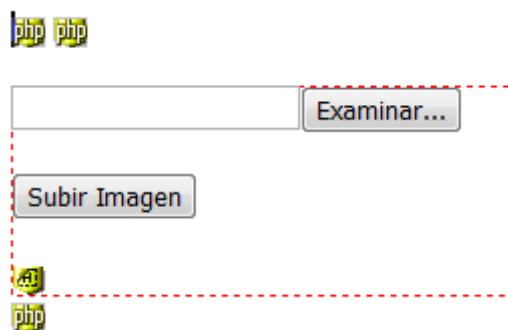
```

```
<input type="button" name="button" id="button" value="Subir Imagen" onclick="javascript:subirimagen();"/>
```

Al hacer clic abre una nueva ventana que sera donde introduzcamos la imagen en cuestión así que creamos una nueva pagina **gestionImagen.php** , la cual se aprovechara del script anteriormente citado



La pagina **gestionImagen.php** , contiene un formulario



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Subir Imagen </title>
</head>

<body>
<form action="gestionImagen.php" method="post" enctype="multipart/form-data" id="form1">

<p>
<input name="userfile" type="file" />

</p>
<p>
<input type="submit" name="button" id="button" value="Subir Imagen" />
</p>
<input type="hidden" name="Enviado" value="form1" />
</form>
```

En este punto hemos de destacar varios aspectos importantes, ya que este formulario tiene un pequeña dificultad añadida

El sistema necesita saber que hemos pulsado el botón de subir imagen, en que ventana se encuentra pulsando ya que podríamos tener varias ventanas abiertas y el sistema debe reconocer donde estamos pulsando.

Así que cuando pulsemos el botón de acción ira a gestionImagen.php. Por otra parte cuando enviamos datos en binario en un formulario como podría ser un pdf o la imagen que vamos a subir se utiliza un tipo de paquete llamado “**multipart/form-data**”, el tipo de formulario será file y en input name “userfile” para el botón de examinar nos abra el disco duro y poder seleccionar la imagen , mientras que el de enviar el formulario sera de tipo submit.

Lo particular llega aquí y es vamos a añadir un campo oculto. Esto lo entenderemos mejor al explicar el Bucle If-Else ,este bucle nos va a detectar cuando ya hemos seleccionado la imagen y hemos pulsado el botón subir imagen , entonces la forma de realizar esto es con name=enviado , el cual pertenece al campo oculto tiene el valor form1, isset comprobara si toma el valor pasado por \$\_POST[“enviado”] y si toma el valor de form1, si esto sucede significara que hemos entrado la pagina de subir imagen , hemos seleccionado algo de ahí y hemos pulsado el botón imagen.

La variable \$\_FILES es utilizada por php para recoger el nombre del archivo con extensión , a esta variable le vamos a asignar la variable \$nombre\_archivo obtendrá el nombre con extensión del archivo .

La función **move\_uploaded\_file** coge el archivo que se ha subido y guardarlo con el nombre que deseemos

Por otra parte la instrucción Else si no se ha cumplido lo comentado anteriormente nos mostrara el formulario.

Vemos el código implementado .

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Subir Imagen </title>
</head>

<body>
<?php if ((isset($_POST["enviado"])) && ($_POST["enviado"] == "form1")) {
    $nombre_archivo = $_FILES['userfile']['name'];
    move_uploaded_file($_FILES['userfile']['tmp_name'], "../documentos/productos/".$nombre_archivo);

    ?>
    <script>
        opener.document.form1.strImagen.value=<?php echo $nombre_archivo; ?>;
        self.close();
    </script>
    <?php
}
else
{?>

<form action="gestionImagen.php" method="post" enctype="multipart/form-data" id="form1">

    <p>
        <input name="userfile" type="file" />
    </p>
    <p>
        <input type="submit" name="button" id="button" value="Subir Imagen" />
    </p>
    <input type="hidden" name="enviado" value="form1" />
</form>
<?php }?>

</body>
</html>
```

Por ultimo nos queda explicar que al cerrar la ventana nos pase el nombre de la imagen al formulario , eso se consigue mediante un script

```
<script>
opener.document.form1.strImagen.value=<?php echo $nombre_archivo; ?>;
self.close();
</script>
```

## 6.2.9 EDITAR PRODUCTOS

### productos\_edit.php

Otra parte principal de la administración. es poder editar/modificar los productos introducidos ademas de la imagen,de nuevo creamos un enlace en EDITAR

```
<td><a href="productos_edit.php">Editar</a> - Eliminar</td>
```

Introducimos una variable por parámetro como hicimos con la edición de categorías:

```
<td><a href="productos_edit.php?recordID=<?php echo $row_Recordset1['idProducto']; ?>">Editar</a> - Eliminar</td>
```

Como indicamos en puntos anteriores realizamos una SELECT en este caso a la tabla categorías y a la tabla productos y un UPDATE a la tabla productos.

```
$varProducto_DatosProducto = "0";
if (isset($_GET["recordID"])) {
    $varProducto_DatosProducto = $_GET["recordID"];
}
mysql_select_db($databaseConexionZapatos, $conexionZapatos);
$query_DatosProducto = sprintf("SELECT * FROM tblproducto WHERE tblproducto.idProducto = %s",
    GetSQLValueString($varProducto_DatosProducto, "int"));
$DatosProducto = mysql_query($query_DatosProducto, $conexionZapatos) or die(mysql_error());
$row_DatosProducto = mysql_fetch_assoc($DatosProducto);
$totalRows_DatosProducto = mysql_num_rows($DatosProducto);

mysql_select_db($databaseConexionZapatos, $conexionZapatos);
$query_ConsultaCategorias = "SELECT * FROM tblcategoria WHERE idPadre = 0 ORDER BY tblcategoria.
    strDescripcion ASC";
$ConsultaCategorias = mysql_query($query_ConsultaCategorias, $conexionZapatos) or die(mysql_error());
$row_ConsultaCategorias = mysql_fetch_assoc($ConsultaCategorias);
$totalRows_ConsultaCategorias = mysql_num_rows($ConsultaCategorias);
```

Para modificar los productos vamos a realizar una operación de UPDATE

```
if ((isset($_POST["MM_update"])) && ($_POST["MM_update"] == "form1")) {
    $updateSQL = sprintf("UPDATE tblproducto SET strNombre=%s, strSEO=%s, dblPrecio=%s, intEstado=%s,
        intCategoria=%s, strImagen=%s, intStock=%s, intOferta=%s WHERE idProducto=%s",
        GetSQLValueString($_POST['strNombre'], "text"),
        GetSQLValueString($_POST['strSEO'], "text"),
        GetSQLValueString($_POST['dblPrecio'], "double"),
        GetSQLValueString($_POST['intEstado'], "int"),
        GetSQLValueString($_POST['intCategoria'], "int"),
        GetSQLValueString($_POST['strImagen'], "text"),
        GetSQLValueString($_POST['intStock'], "int"),
        GetSQLValueString($_POST['intOferta'], "int"),
        GetSQLValueString($_POST['idProducto'], "int"));

    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $Result1 = mysql_query($updateSQL, $conexionzapatos) or die(mysql_error());

    $updateGoTo = "productos_lista.php";
    if (isset($_SERVER['QUERY_STRING'])) {
        $updateGoTo .= (strpos($updateGoTo, '?')) ? "&" : "?";
        $updateGoTo .= $_SERVER['QUERY_STRING'];
    }
    header(sprintf("Location: %s", $updateGoTo));
```

En cuanto a crear el formulario seria similar al expuesto anteriormente así que no lo mostrare y ya tendríamos creada la pagina de editar productos

## 6.2.10 ELIMINAR PRODUCTOS.

### productos\_delete.php

Para poder Eliminar los productos de nuevo enlazamos Eliminar con una nueva pagina llamada productos\_delete.php y le volvemos a enviar por parámetros. como hicimos con el edit,recordID sera el valor que enviamos desde el botón eliminar producto

```
<a href="productos_delete.php?recordID=<?php echo $row_Recordset1['idProducto']; ?>">Eliminar</a></td>
```

Mediante una operación DELETE

```
if ((isset($_GET['recordID'])) && ($_GET['recordID'] != "")) {
    $deleteSQL = sprintf("DELETE FROM tblproducto WHERE idProducto=%s",
        GetSQLValueString($_GET['recordID'], "int"));

    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $Result1 = mysql_query($deleteSQL, $conexionzapatos) or die(mysql_error());

    $deleteGoTo = "productos_lista.php";
    if (isset($_SERVER['QUERY_STRING'])) {
        $deleteGoTo .= (strpos($deleteGoTo, '?')) ? "&" : "?";
        $deleteGoTo .= $_SERVER['QUERY_STRING'];
    }
    header(sprintf("Location: %s", $deleteGoTo));
}
```

Donde en el where el idproducto sea el valor que pasamos por parámetro que es \$\_GET['recordID']

## CONFIRMACIÓN ELIMINAR PRODUCTOS

Confirmación de eliminación de productos → JavaScript al pulsar eliminar que no confirme si realmente queremos eliminar.

Este sencillo JavaScript nos devuelve el valor de la función asegurar

- onclick="JavaScript:return asegurar();"

```
<td><a href="carrito_eliminar.php?recordID=<?php echo $row_DatosCarrito['intContador']; ?>"  
    onclick="javascript:return asegurar();" >Eliminar</a></td>
```

```
<script>  
function asegurar()  
{  
    rc = confirm("¿Seguro que deseas| eliminar este producto?");  
    return rc;  
}  
</script>
```

### 6.2.11 VALIDACIÓN DE CAMPOS.

La validación de campos no es mas que impedir que se envié la información si existen campos vacíos en el formulario. Para ello vamos a utilizar Ajax y un poco de JavaScript y Spry

```
<script type="text/javascript">  
var sprytextfield1 = new Spry.Widget.ValidationTextField("sprytextfield1");  
var sprytextfield2 = new Spry.Widget.ValidationTextField("sprytextfield2");  
var sprytextfield3 = new Spry.Widget.ValidationTextField("sprytextfield3");  
</script>
```

,esta variable llamara a la función ValidationTextField que ejecutara la comprobación

## Editar Producto

Nombre:	<input style="background-color: #f08080; border: 1px solid black; width: 150px; height: 20px;" type="text" value=""/>	Se necesita un valor.
SEO:	<input style="width: 150px; height: 20px;" type="text" value="bota-de-agua"/>	
Precio:	<input style="background-color: #f08080; border: 1px solid black; width: 150px; height: 20px;" type="text" value=""/>	Se necesita un valor.
Estado:	<input style="width: 100px; height: 20px;" type="button" value="Activo"/>	
Categoría:	<input style="width: 100px; height: 20px;" type="button" value="Zapatillas"/>	
Imagen:	<input style="width: 300px; height: 20px;" type="text" value="zapato1.jpg"/> <input style="width: 150px; height: 20px;" type="button" value="Subir Imagen"/>	
<input style="width: 250px; height: 25px;" type="button" value="Actualizar producto"/>		

## 6.2.12 GESTIÓN DE CLIENTES

### usuarios\_lista.php

Sería también interesante que el administrador del sistema pudiera conocer quienes son los usuarios registrados , o clientes que han efectuado una compra por si hubiera algún tipo de problema siempre es aconsejable tener acceso a ello a un registro de clientes.

Necesitamos una consulta SELECT para obtener dichos usuarios

```
mysql_select_db($databaseConexionZapatos, $conexionZapatos);
$query_DatosUsuarios = "SELECT * FROM tblusuario ORDER BY tblusuario.strNombre ASC";
$DatosUsuarios = mysql_query($query_DatosUsuarios, $conexionZapatos) or die(mysql_error());
$row_DatosUsuarios = mysql_fetch_assoc($DatosUsuarios);
$totalRows_DatosUsuarios = mysql_num_rows($DatosUsuarios);
```

Para ello vamos a mostrar los campos en una tabla

```
<!-- end .sidebar1 --></div>
<div class="content"><!-- InstanceBeginEditable name="Contenido" -->
<h1>Lista de Usuarios</h1>
<p>&nbsp;</p>
<table border="0" align="center" cellpadding="0" cellspacing="0">
  <tr class="tablaprincipal">
    <td width="147">id</td>
    <td width="182">Nombre</td>
    <td width="112">ACCIONES</td>
  </tr>
  <?php do { ?>
    <tr class="brillo">
      <td><a href="usuarios_datos.php?recordID=<?php echo $row_DatosUsuarios['idUsuario']; ?>">
          <?php echo $row_DatosUsuarios['idUsuario']; ?>&nbsp; </a></td>
      <td><?php echo $row_DatosUsuarios['strNombre']; ?>&nbsp; </td>
      <td>Editar - Eliminar</td>
    </tr>
    <?php } while ($row_DatosUsuarios = mysql_fetch_assoc($DatosUsuarios)); ?>
</table>
```

sacamos por pantalla los datos con echo y añadimos un bucle Do-While para que nos muestren todos los usuarios que hay en nuestra base de datos

Repetir	ID	NOMBRE	ACCIONES
{DatosUsuarios.TotalRecords}	{DatosUsuarios.idUsuario}	{DatosUsuarios.strNombre}	Editar - Eliminar

Podemos obtener el numero de usuarios registrado en el sistema mediante:

```
<?php echo $totalRows_DatosUsuarios ?> Registros Total
```

Este seria el resultado final

ID	NOMBRE	ACCIONES
7	aaagh	Editar - Eliminar
4	Frank	Editar - Eliminar
5	frankz	Editar - Eliminar
2	Jorge Pepe	Editar - Eliminar
6	Lola	Editar - Eliminar
1	Pepe Luis	Editar - Eliminar
3	sdfsdf	Editar - Eliminar

7 Registros Total

PD: Se podrían añadir tantos atributos como quisiéramos visualizar del cliente con el mismo procedimiento.

Posteriormente iremos añadiendo mas opciones de administración., a medida que vayamos implementando la parte publica, y las vea necesarias te lo indicare.

#### NOTA

Se que puede parecer confuso que te explique posteriormente partes de la administración. en la parte del al pagina publica, pero es un poco complicado de indexar esto, ya que constantemente voy modificando cosas de ambas partes, así que creo que la mejor manera de mostrártelo es en el orden que he seguido yo mientras iba haciendo el proyecto, te aseguro que haré referencia a que parte pertenece en el momento de la explicación.

## 6.2.13 STOCK DE PRODUCTOS.

Editamos la tabla tblproducto y añadimos el campo intStock.

Modificamos la pagina productos\_lista.php y le añadimos un nuevo campo a la tabla

```
</tr>
<?php do { ?>
<tr class="brillo">
<td><?php echo $row_Recordset1['strNombre']; ?></td>
<td><?php echo $row_Recordset1['intEstado']; ?></td>
<td><?php echo $row_Recordset1['intStock']; ?></td>
<td><a href="productos_edit.php?recordID=<?php echo
$row_Recordset1['idProducto']; ?>">Editar</a> - <a href=
"productos_delete.php?recordID=<?php echo $row_Recordset1[
'idProducto'; ?>">Eliminar</a></td>
</tr>
<?php } while ($row_Recordset1 = mysql_fetch_assoc($Recordset1
)); ?>
</table>
<!-- InstanceEndEditable -->
```

ESTADO	STOCK
{Recordset1.intEstado}	{Recordset1.intStock}

También añadiremos el campo Stock en la pagina de añadir productos → productos\_add.php insertamos el nuevo valor en la consulta y lo mismo con productos\_edit.php

## Editar Producto

Nombre:	<input type="text" value="Zapato 2"/>
SEO:	<input type="text" value="sdfsdf"/>
Precio:	<input type="text" value="20"/>
Estado:	Activo ▼
Categoría:	Botas ▼
Imagen:	<input type="text" value="zapato5.jpg"/> <input type="button" value="Subir Imagen"/>
Stock:	<input type="text" value="20"/>
<input type="button" value="Actualizar registro"/>	

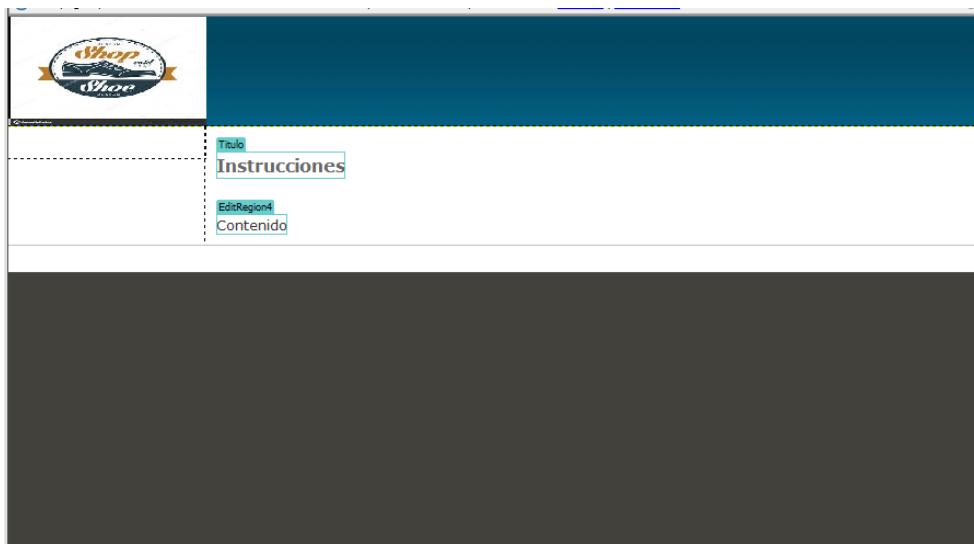
## 6.3 IMPLEMENTACIÓN PARTE PUBLICA PARA CLIENTES (FRONTEND).

Vamos a dejar de momento la parte de la administración. ya que ya tenemos una parte bastante funcional de ella posteriormente volveremos para gestionar las compras, control de pedidos etc, nos dedicaremos ahora a la parte publica que ve el cliente también conocida como Frontend.

### 6.3.1 PLANTILLA PAGINA PRINCIPAL.

#### Principal.dwt.php

Realizaremos una plantilla para así poder hacer mas rápidamente las paginas que necesitemos para gestionar la parte publica de nuestra tienda.



### 6.3.2 PAGINA PRINCIPAL

#### index.php

En principio en esta añadimos un include para el menú lateral recordemos que el menú se gestiona desde la pagina includes/catalogo.php

```
</head>
<body>

<div class="container">
    <div class="header"><div class="headerinterior"></div></div>
    <div class="subcontenedor">
        <div class="sidebar1">
            <?php include("includes/catalogo.php"); ?>
        <!-- end .sidebar1 --></div>
        <div class="content">
            <h1>Instrucciones</h1>
            |
            <!-- end .content --></div>
    <!-- end .subcontenedor -->
</div>
<div class="footer">
    <p>Pie de pagina</p>
    <!-- end .footer --></div>
<!-- end .container --></div>
</body>
<!-- InstanceEnd --></html>
```

En **catalogo.php** iremos añadiendo todos los elementos que queramos visualizar en el menú, en el podremos generar las consultas o modificaciones que creamos necesarias para obtener la información deseada.

Primeramente obtendremos las categorías para ello necesitamos sacar una lista del directorio es decir un SELECT con un bucle Do-while para ver las todas las categorías

```
mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_Recordset1 = "SELECT * FROM tblcategoria WHERE idPadre = 0 ORDER BY tblcategoria.strDescripcion";
$Recordset1 = mysql_query($query_Recordset1, $conexionzapatos) or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($Recordset1);
$totalRows_Recordset1 = mysql_num_rows($Recordset1);
```

```
<?php do { ?>
<li class="lineacatalogo"><a href="categoria_ver.php?cat=<?php echo $row_Recordset1['idCategoria']; ?>">
    <?php echo $row_Recordset1['strDescripcion']; ?></a></li>
    <?php mostrar_subcategorias($row_Recordset1['idCategoria']);?>
    <?php } while ($row_Recordset1 = mysql_fetch_assoc($Recordset1)); ?>
</ul>
<p>
<?php
mysql_free_result($Recordset1);
?>
```

Este seria el resultado final:



### 6.3.3 ALTA USUARIO

#### alta\_usuario.php

Creamos un formulario de alta

Titulo  
Alta Usuario

EditRegion4

Formulario de alta en la tienda:

Nombre:

Email:

Contraseña:

Darme de Alta

① ②

Formulario:

```
<p>Formulario de alta en la tienda:</p>
<p>&nbsp;</p>
<form action=<?php echo $editFormAction; ?>" method="post" name="form1" id="form1">
  <table align="center">
    <tr valign="baseline">
      <td nowrap="nowrap" align="right">Nombre:</td>
      <td><span id="sprytextfield1">
        <input type="text" name="strNombre" value="" size="32" />
        <span class="textfieldRequiredMsg">Se necesita un valor.</span></span></td>
      </tr>
      <tr valign="baseline">
        <td nowrap="nowrap" align="right">Email:</td>
        <td><span id="sprytextfield2">
          <input type="text" name="strEmail" value="" size="32" />
          <span class="textfieldRequiredMsg">Se necesita un valor.</span><span class="textfieldInvalidFormatMsg">Formato no válido.</span></span></td>
        </tr>
        <tr valign="baseline">
          <td nowrap="nowrap" align="right">Contraseña:</td>
          <td><span id="sprytextfield3">
            <input type="password" name="strPassword" value="" size="32" />
            <span class="textfieldRequiredMsg">Se necesita un valor.</span></span></td>
          </tr>
          <tr valign="baseline">
            <td nowrap="nowrap" align="right">&nbsp;</td>
            <td><input type="submit" value="Darme de Alta" /></td>
          </tr>
  </table>
  <input type="hidden" name="intActivo" value="1" />
  <input type="hidden" name="MM_insert" value="form1" />
</form>
```

Añadimos la validación de campos para que ninguno se pueda quedar vacío a la hora de introducir datos.

```
<script type="text/javascript">

var sprytextfield1 = new Spry.Widget.ValidationTextField("sprytextfield1");
var sprytextfield2 = new Spry.Widget.ValidationTextField("sprytextfield2", "email");
var sprytextfield3 = new Spry.Widget.ValidationTextField("sprytextfield3");

</script>
```

Para introducir los datos en el formulario de registro, realizamos un INSERT

```
$editFormAction = $_SERVER['PHP_SELF'];
if (isset($_SERVER['QUERY_STRING'])) {
    $editFormAction .= "?" . htmlentities($_SERVER['QUERY_STRING']);
}

if ((isset($_POST["MM_insert"])) && ($_POST["MM_insert"] == "form1")) {
    $insertSQL = sprintf("INSERT INTO tbliusuario (strNombre, strEmail, intActivo, strPassword)
                          VALUES (%s, %s, %s, %s)",
                          GetSQLValueString($_POST['strNombre'], "text"),
                          GetSQLValueString($_POST['strEmail'], "text"),
                          GetSQLValueString($_POST['intActivo'], "int"),
                          GetSQLValueString($_POST['strPassword'], "text"));

    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $Result1 = mysql_query($insertSQL, $conexionzapatos) or die(mysql_error());

    $insertGoTo = "alta_ok.php";
    if (isset($_SERVER['QUERY_STRING'])) {
        $insertGoTo .= (strpos($insertGoTo, '?')) ? "&" : "?";
        $insertGoTo .= $_SERVER['QUERY_STRING'];
    }
    header(sprintf("Location: %s", $insertGoTo));
}
```

Antes de efectuar la operación de inserción de registros añadimos un bucle de control If , el cual realizara una comparación entre strings mediante un AND , si esta condición se satisface se ejecutara la INSERT, de nuevo realizamos la inserción de registros mediante un paso de variables por parámetros. , si tiene éxito nos lleva a la pagina alta\_ok.php

## COMPROBACIÓN DE EXISTENCIA DEL EMAIL

Es interesante que el sistema pueda reconocer si un email ya ha sido introducido con anterioridad , y este ya consta en la base de datos.

Veamos como realizamos esto mediante programación:

```
// *** Redirect if username exists
$MM_flag="MM_insert";
if (isset($_POST[$MM_flag])) {
    $MM_dupKeyRedirect="alta_emailrepetido.php";
    $loginUsername = $_POST['strEmail'];
    $LoginRS__query = sprintf("SELECT strEmail FROM tblusuario WHERE strEmail=%s",
        GetSQLValueString($loginUsername, "text"));

    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $LoginRS=mysql_query($LoginRS__query, $conexionzapatos) or die(mysql_error());
    $loginFoundUser = mysql_num_rows($LoginRS);

    //if there is a row in the database, the username was found - can not add the requested username
    if($loginFoundUser){
        $MM_qsChar = "?";
        //append the username to the redirect page
        if (substr_count($MM_dupKeyRedirect,"?") >=1) $MM_qsChar = "&";
        $MM_dupKeyRedirect = $MM_dupKeyRedirect . $MM_qsChar . "requsername=".$loginUsername;
        header ("Location: $MM_dupKeyRedirect");
        exit;
    }
}
```

Cabe destacar 2 partes :

1.- La variable MM\_INSERT la cual recibimos del formulario , se le pasa el valor a la variable \$MM\_flag, con el bucle de control If y mediante isset comprobamos el valor, si esta condición se cumple se ejecuta la consulta SELECT a la base de datos , si esta consulta se efectuá correctamente nos redirecciona → **alta\_ok.php**

2.- Si no se ha ejecutado el bucle anterior mediante 2 If podemos controlar si el nombre el registro ya existe en la base de datos, este tomara el valor de la variable \$MM\_dupKeyRedirect redireccionando a la pagina **alta\_emailrepetido.php**.

De manera análoga a los puntos anteriores realizamos una validación de los campos del formulario

### 6.3.4 LOGIN DE USUARIO

Para que un usuario pueda loguearse en el sistema necesitamos un formulario de inserción de datos, el cual especifique los datos de acceso al sistema.

```
<p>Accede a la pagina con tus datos:</p>
<form id="form1" name="form1" method="POST" action=<?php echo $loginFormAction; ?>>
<p>Email:<br>
    <label for="srtEmail"></label>
    <input type="text" name="srtEmail" id="srtEmail" />
</p>
<p>Contraseña:<br>
    <label for="strPassword"></label>
    <input type="text" name="strPassword" id="strPassword" />
</p>
<p>
    <input type="submit" name="button" id="button" value="Enviar" />
</p>
</form>
```



### VARIABLES DE SESIÓN

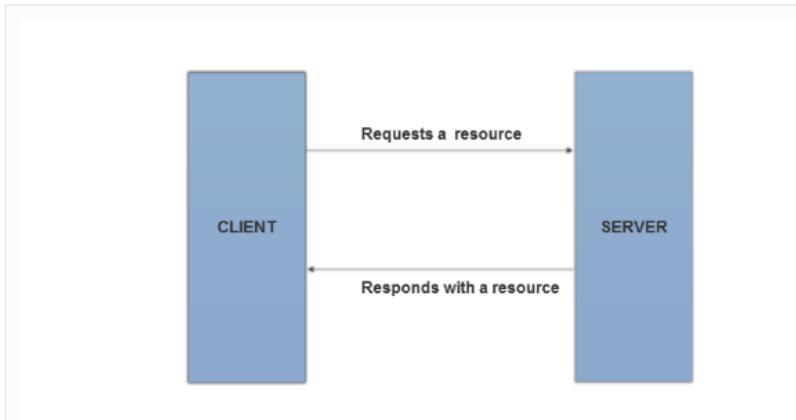
En este punto cabe explicar como van a entrar en juego las variables de sesión de sistema, vamos a explicar en que consisten estas:

El manejo de sesiones es un concepto clave en PHP que permite que la información de usuario persista entre todas las páginas de un sitio web o app.

Empezaremos con una explicación de cómo funcionan las sesiones y cómo estas están relacionadas. Para ver porqué las sesiones son necesarias, tenemos que viajar atrás y ver como esta diseñado el protocolo HTTP.

El protocolo HTTP es un protocolo sin estado, lo que significa que no hay forma de que un servidor recuerde a un usuario específico entre múltiples peticiones. Por ejemplo, cuando accedes a una página web, el servidor sólo es responsable de proveer el contenido de la página solicitada. Así que cuando accedes a otras páginas en el mismo sitio web, el servidor web interpreta cada petición separadamente, como si no estuvieran relacionadas unas con otras. No hay forma para el servidor de saber que cada petición fue originada por el mismo usuario.

El siguiente diagrama refleja el protocolo HTTP en pocas palabras.



En este modelo, si quieras mostrar información relativa a un usuario específico, deberás autenticar al usuario en cada petición. ¡Imagina que tuvieras que introducir tu nombre de usuario y contraseña en cada página que muestre tu información de perfil! Sí, sería incómodo y nada práctico, y aquí es donde las sesiones entran en juego.

Una sesión permite compartir información entre las diferentes páginas de un único sitio web o app, así que ayuda a mantener el estado. Esto permite al servidor conocer que todas las peticiones se originan desde el mismo usuario, permitiendo al sitio web mostrar información y preferencias específicas de ese usuario.

Ahora que hemos visto una breve introducción a cómo funcionan las sesiones, veamos como iniciar una sesión

Cuando quieras tratar con variables de sesión, necesitas asegurarte de que la sesión ya haya empezado . Para ello utilizamos la función `session_start`. Una vez tenemos iniciada la sesión esta debe contener la variables de sesión,

Como discutimos anteriormente, una vez que una sesión es iniciada, el array super-global `$_SESSION` es inicializado con la correspondiente información de sesión. Por defecto, se inicializa con un array vacío, y puedes almacenar más información usando un par clave-valor.

Cuando almacenas datos en una sesión usando la super-global `$_SESSION`, finalmente se almacenan en su correspondiente fichero de sesión en el servidor que fue creado cuando la sesión fue iniciada. De esta forma, los datos de sesión son compartidos entre múltiples peticiones.

La información de sesión se comparte entre peticiones, y de esta forma las variables de sesión inicializadas en una página pueden ser accedidas desde otras páginas también, hasta que la sesión expira. Generalmente, una sesión expira cuando se cierra el navegador.

Una vez visto como funciona una variable de sesión vamos a implementar el código para el inicio de sesión. Nuestra variable de sesión vendrá definida por :

```
$SESSION['MM_IdUsuario'] = $row_LoginRS["idUsuario"];
```

La cual toma el valor de \$row\_loginRS al iniciar la sesión, también cabe destacar el bucle de control If el cual mediante un isset comprobara si toma el valor, la variable strEmail pasada por POST ya que proviene de un formulario

```
<?php
// *** Validate request to login to this site.
if (!isset($_SESSION)) {
| session_start();
}

$loginFormAction = $_SERVER['PHP_SELF'];
if (isset($_GET['accesscheck'])) {
| $_SESSION['PrevUrl'] = $_GET['accesscheck'];
}

if (isset($_POST['strEmail'])) {
| $loginUsername=$_POST['strEmail'];
| $password=$_POST['strPassword'];
| $MM_fldUserAuthorization = "";
| $MM_redirectLoginSuccess = "acceso_ok.php";
| $MM_redirectLoginFailed = "acceso_error.php";
| $MM_redirecttoReferrer = false;
mysql_select_db($databaseConexionzapatos, $conexionzapatos);

$LoginRS__query=printf("SELECT idUsuario, strEmail, strPassword FROM tblusuario WHERE strEmail=%s AND strPassword=%s",
| GetSQLValueString($loginUsername, "text"), GetSQLValueString($password, "text"));

$LoginRS = mysql_query($LoginRS__query, $conexionzapatos) or die(mysql_error());
$row_LoginRS = mysql_fetch_assoc($LoginRS);
$loginFoundUser = mysql_num_rows($LoginRS);
if ($loginFoundUser) {
| $loginStrGroup = "";
```

Al cumplir esta condición se realizara un SELECT para comprobar si los registros introducidos coinciden con los de nuestra base de datos que corresponden con un usuario dado de alta previamente.

A continuación mostramos las variables de sesión implementadas

```
if (PHP_VERSION >= 5.1) {session_regenerate_id(true);} else {session_regenerate_id();}
//declare two session variables and assign them
$_SESSION['MM_Username'] = $loginUsername;
$_SESSION['MM_UserGroup'] = $loginStrGroup;
$_SESSION['MM_IdUsuario'] = $row_LoginRS["idUsuario"];

if (isset($_SESSION['PrevUrl']) && false) {
| $MM_redirectLoginSuccess = $_SESSION['PrevUrl'];
}
header("Location: " . $MM_redirectLoginSuccess );
}
else {
| header("Location: ". $MM_redirectLoginFailed );
}
```

Con el bucle If-If-Else conseguiremos la redirección a las páginas acceso\_ok.php y acceso\_herropea, mediante las variables **\$MM\_redirectLoginSuccess** y **\$MM\_redirectLoginFailed**.

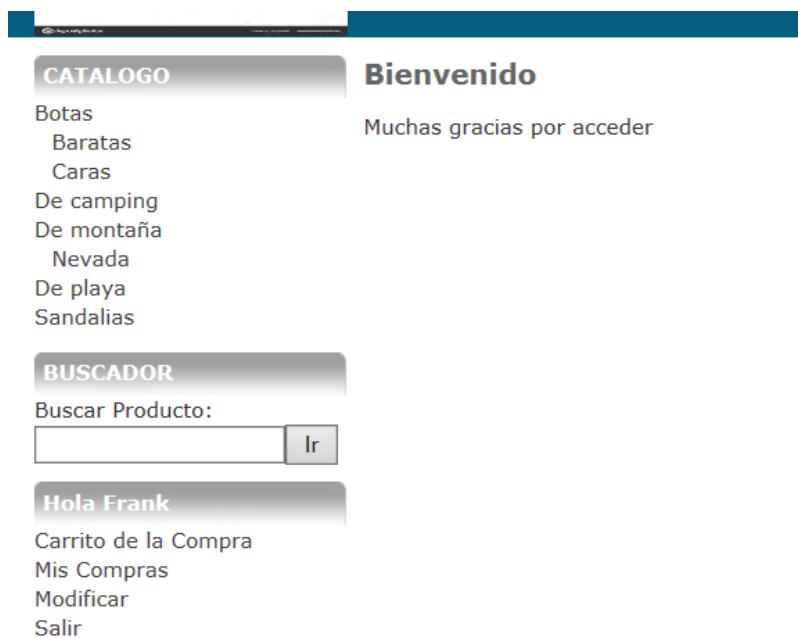
Veamos como funciona correctamente

## Acceso usuario

Accede a la página con tus datos:

Email:

Contraseña:



## MOSTRAR NOMBRE USUARIO AL ACCEDER AL SISTEMA

Una vez el usuario haya accedido al sistema sería interesante que pudiera visualizar su nombre al loguear como usuario. Tenemos un problema añadido, al efectuar el login el usuario accede con su Email y su contraseña, así que de la consulta solo podemos obtener el email, quedaría más presentable que visualizara su nombre en vez de su email, eso se conseguirá mediante una función que obtenga el nombre del usuario.

En este punto vamos a comentar el archivo funciones.php, el cual vamos a utilizar en diversas partes del proyecto. En este archivo he ido implementando una serie de funciones para obtener registros determinados de nuestra base de datos .

Mostrare algún ejemplo en este punto dado que finalmente este archivo contiene unas 400 lineas de código y me parece excesivo analizar función por función.

Si nos centramos en el apartado nos lleva obtener un nombre de un usuario:

```
function ObtenerNombreUsuario($identificador)
{
    global $databaseConexionzapatos, $conexionzapatos;
    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $query_ConsultaFuncion = sprintf("SELECT tblusuario.strNombre
        FROM tblusuario
        WHERE tblusuario.idUsuario = %s", $identificador);

    $ConsultaFuncion = mysql_query($query_ConsultaFuncion, $conexionzapatos) or die(mysql_error());
    $row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion);
    $totalRows_ConsultaFuncion = mysql_num_rows($ConsultaFuncion);

    return $row_ConsultaFuncion['strNombre'];
    mysql_free_result($ConsultaFuncion);
}
```

Implementamos la función ObtenerNombreUsuario , en la cual le pasamos la variable \$identificador, establecemos como variables globales ,\$database\_Conexionzapatos, \$conexionzapatos.

Tan solo ahora tenemos que realizar una SELECT que obtendrá por parámetro la variable \$identificador obtenida de la consulta .

Cabe destacar la función de php **return** que devolverá el valor strNombre a partir de la consulta que es el que estábamos buscando.

Puede parecer en principio que realizar estas funciones no es necesario ya que podríamos añadir directamente el código en la pagina en la cual necesitamos obtener el registro, bajo mi punto de vista veo mucho mas eficiente crear una función y cada vez que tengamos que obtener dicho valor realizamos una llamada a la función.

```
<?php
if ((isset($_SESSION['MM_Username'])) && ($_SESSION['MM_Username'] != ""))
{
    ?>
    <ul class="listacatalogo">
        <?php
        echo '<div class="cabeceracatalogo">Hola ';
        echo ObtenerNombreUsuario($_SESSION['MM_IdUsuario'])."</div>";
    ?>
```

Ya tendríamos solucionado obtener el nombre de la persona que ha realizado el login en cualquier parte de la web que se ha accedido con tus credenciales de acceso.

Finalmente el código quedaría:

```
<?php require_once('Connections/conexiontienda.php'); ?>
<?php
if (!function_exists("GetSQLValueString")) {
function GetSQLValueString($theValue, $theType, $theDefinedValue = "", $theNotDefinedValue = "") {
{
    if (PHP_VERSION < 6) {
        $theValue = get_magic_quotes_gpc() ? stripslashes($theValue) : $theValue;
    }

    $theValue = function_exists("mysql_real_escape_string") ? mysql_real_escape_string($theValue) : mysql_escape_string($theValue);

    switch ($theType) {
        case "text":
            $theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
            break;
        case "long":
        case "int":
            $theValue = ($theValue != "") ? intval($theValue) : "NULL";
            break;
        case "double":
            $theValue = ($theValue != "") ? doubleval($theValue) : "NULL";
            break;
        case "date":
            $theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
            break;
        case "defined":
            $theValue = ($theValue != "") ? $theDefinedValue : $theNotDefinedValue;
            break;
    }
}
return $theValue;
}

mysql_select_db($databaseConexionTienda, $conexionTienda);
$query_Recordset1 = "SELECT * FROM tbcategoria ORDER BY tbcategoria.strDescripcion";
$Recordset1 = mysql_query($query_Recordset1, $conexionTienda) or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($Recordset1);
$totalRows_Recordset1 = mysql_num_rows($Recordset1);
?>

<?php do { ?>
    <?php echo $row_Recordset1['strDescripcion']; ?><br>
    <?php } while ($row_Recordset1 = mysql_fetch_assoc($Recordset1)); ?>

<p>
    <?php
mysql_free_result($Recordset1);
?>

</br>
<a href="alta_usuario.php">Darme de alta</a></p>
<?php
if ((isset($_SESSION['MM_Username'])) && ($_SESSION['MM_Username'] != ""))
{
    echo "Hola". $_SESSION['MM_Username'];
}
else
{?>
<p><a href="acceso.php">Acceder</a></p>
<?php }?>
?>
```

Otro detalle interesante es que no aparezca el enlace de darse de alta cuando ya hemos accedido al sistema con sus credenciales de acceso.

Para que no aparezca el enlace de dar de alta movemos el enlace que contiene Dar de alta a dentro del ELSE

```
<?php
if ((isset($_SESSION['MM_Username'])) && ($_SESSION['MM_Username'] != ""))
{
    echo "Hola". $_SESSION['MM_Username'];
}
else
{?> </br>
<a href="alta_usuario.php">Darme de alta</a></p>
<p><a href="acceso.php">Acceder</a></p>
<?php }?>
?>
```

Este seria el resultado:



### 6.3.5 LOGOUT USUARIO.

#### usuario\_cerrarsesion.php

Como comentamos anteriormente el protocolo HTTP que gestiona las conexiones, no tiene estado, cada conexión entre el cliente y el servidor es independiente de las demás .

Existen diferentes maneras de cerrar una sesión. de usuario, ya que PHP gestiona una biblioteca de funciones para la gestión de las sesiones

Entre ellas podemos encontrar:

- **session\_unregister (variable)** → Elimina una variable de sesión
- **session\_destroy ()** → Cierra una sesión
- **unset (\$\_SESSION['nombre']);** → Elimina una variable de sesión

Hay que tener en cuenta que muchas de estas funciones se crearon hace tiempo cuando los navegadores no finalizaban las variables de sesión. A día de hoy todos los navegadores tras un tiempo de inactividad de 10-20m dependiendo el navegador automáticamente cierra las variables de sesión.

Aun con este elemento un usuario debe poder salir de su sesión. iniciada en cualquier momento, así que debemos implementar el logout del nuestra aplicación web.

Tras analizar los requisitos del sistema he optado por lo siguiente :

```
<?php require_once('Connections/conexionzapatos.php'); ?>
<?php
    $_SESSION['MM_Username'] = "";
    $_SESSION['MM_UserGroup'] = "";
    $_SESSION['MM_IdUsuario'] = ""; ?>
```

Tras pulsar el botón de logout vamos a a inicializar las variable de sesión. a un valor nulo, con lo que la sesión. acabara.

Una buena pregunta es porque he utilizado este método y por ejemplo no con la función **session\_destroy ()** o **unset (\$\_SESSION['nombre']);**, por lo que he visto estas funciones están mayormente prediseñadas para mantener mas de una variable de sesión. conjuntamente , en nuestro caso solo tenemos la variable de sesión. MM\_IdUsuario, así que cambiando su valor conseguiremos el mismo efecto que las citadas anteriormente.

Una vez explicado esto vamos a ver como he conseguido esto en la tienda web. Al pulsar el botón de logout nos redireccionará a `usuario_cerrarsesion.php`, en la cual realizaremos cambio de valor en la variable de sesión. .

[Botas](#)  
[Botas de Campo](#)  
[Zapatillas](#)  
[Zapatos](#)  
[Zapatos de tacón](#)

[Darme de Alta](#)

[Acceder](#)

## Cerrando Sesión

Muchas gracias por comprar en nuestro sitio.

### 6.3.6 EXPIRACIÓN DE SESIÓN.

Las ultimas versiones de navegadores automáticamente al no realizar ninguna acción durante unos 10-20min en función de navegador darán por expirada la sesión. y no podremos continuar navegando en dicha pagina a no ser que volvamos a loguear.

Si nos fijamos un poco en el archivo de acceso al sistema → acceso.php, cuando le damos acceso o loguear esta tomando el valor de estas variables

```
//declare two session variables and assign them
$_SESSION['MM_Username'] = $loginUsername;
$_SESSION['MM_UserGroup'] = $loginStrGroup;
$_SESSION['MM_IdUsuario'] = $row_LoginRS["idUsuario"];
```

Para nosotros el principal es MM\_IdUsuario, que recoge el valor del idUsuario,estas variables son las que realmente expiran, por seguridad , este tiempo podrá variar en función de la seguridad del servidor.

Vamos a crear una pagina que nos avise que la sesión. ha expirado->usuario\_sesión.\_caducada

Acabo de detectar un serio problema de seguridad en la pagina si yo tengo abierto el carrito de la compra por ejemplo al que solo tienen acceso los clientes registrados en 2 pestañas distintas y cierro la sesión. en una de ellas al actualizar la otra pestaña no debería dejarnos continuar ya que hemos cerrado la sesión., en mi caso no esta ocurriendo .

Buscando un poco sobre el tema he encontrado una posible solución que muestro en la siguiente hoja.

```

<?php
if (!isset($_SESSION)) {
    session_start();
}
$MM_authorizedUsers = "";
$MM_donotCheckaccess = "true";

// *** Restrict Access To Page: Grant or deny access to this page
function isAuthorized($strUsers, $strGroups, $UserName, $UserGroup) {
    // For security, start by assuming the visitor is NOT authorized.
    $isValid = False;

    // When a visitor has logged into this site, the Session variable MM_Username set equal to their
    // username.
    // Therefore, we know that a user is NOT logged in if that Session variable is blank.
    if (!empty($UserName)) {
        // Besides being logged in, you may restrict access to only certain users based on an ID
        // established when they login.
        // Parse the strings into arrays.
        $arrUsers = Explode(",", $strUsers);
        $arrGroups = Explode(",", $strGroups);
        if (in_array($UserName, $arrUsers)) {
            $isValid = true;
        }
        // Or, you may restrict access to only certain users based on their username.
        if (in_array($UserGroup, $arrGroups)) {
            $isValid = true;
        }
        // Or, you may restrict access to only certain users based on their username.
        if (in_array($UserGroup, $arrGroups)) {
            $isValid = true;
        }
        if (($strUsers == "") && true) {
            $isValid = true;
        }
    }
    return $isValid;
}

$MM_restrictGoTo = "usuario_sesion_caducada.php";
if (!((isset($_SESSION['MM_Username'])) && (isAuthorized("", $MM_authorizedUsers, $_SESSION['
    MM_Username'], $_SESSION['MM_UserGroup'])))) {
    $MM_qsChar = "?";
    $MM_referrer = $_SERVER['PHP_SELF'];
    if (strpos($MM_restrictGoTo, "?")) $MM_qsChar = "&";
    if (isset($_SERVER['QUERY_STRING']) && strlen($_SERVER['QUERY_STRING']) > 0)
        $MM_referrer .= "?" . $_SERVER['QUERY_STRING'];
    $MM_restrictGoTo = $MM_restrictGoTo . $MM_qsChar . "accesscheck=" . urlencode($MM_referrer);
    header("Location: " . $MM_restrictGoTo);
    exit;
}
?>

```

No he sido yo el que ha implementado este código, Pero comentemos como funciona lo que realiza es una comparación entre variables de sesión., estas variables son pasadas a un array multivalor, en el cual se asigna una posición de memoria para cada una de estas variables en función asignando valores true o false verificando la condición que se necesita en ese momento .

## **6.3.7 MODIFICAR DATOS USUARIO REGISTRADO**

## **usuario\_modificar.php**

Es de vital importancia en una aplicación web en la cual se accede mediante unas credenciales, que estas puedan ser modificadas por el usuario, ya sea por un error al darse de alta o bien porque desea cambiar su email o contraseña.

Para ello en nuestra pagina principal index.php creamos un enlace Modificar, el cual nos redirecciona a **usuario\_modificar.php** , que es donde se efectuaran las modificaciones .

En todo este proceso hemos de tener en cuenta que la variable de sesión. **MM\_IdUsuario**, estará activa en todo momento ya que estamos logueados al efectuar cualquier tipo de operación , el sistema deberá tenerla activa para saber en todo momento de que usuario se trata.

Para modificar los datos del usuario activo, implementaremos un formulario en el cual se va a efectuar un UPDATE de nuestra base de datos , pero primeramente necesitamos una consulta SELECT para obtener dichos registros.

Veamos como hemos implementado esto:

```
$varUsuario_DatosUsuario = "0";
if (isset($_SESSION["MM_IdUsuario"])) {
    $varUsuario_DatosUsuario = $_SESSION["MM_IdUsuario"];
}
mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_DatosUsuario = sprintf("SELECT * FROM tblusuario WHERE tblusuario.idUsuario = %s",
    GetSQLValueString($varUsuario_DatosUsuario, "int"));

$DatosUsuario = mysql_query($query_DatosUsuario, $conexionzapatos) or die(mysql_error());
$row_DatosUsuario = mysql_fetch_assoc($DatosUsuario);
$totalRows_DatosUsuario = mysql_num_rows($DatosUsuario);
```

Inicializamos la variable \$varUsuario\_DatosUsuario = "0";, la cual mediante If isset tomara el valor de \$\_SESSION["MM\_IdUsuario"] , esto ocurrirá siempre que accedamos al sistema. Una vez esta realizada esta comprobación se ejecutara la SELECT con paso por parámetros , \$varUsuario\_DatosUsuario tomara dicho valor .

Una vez el sistema tiene este valor veamos como lo actualizamos, y esto se consigue mediante un **UPDATE**

```

$editFormAction = $_SERVER['PHP_SELF'];
if (isset($_SERVER['QUERY_STRING'])) {
    $editFormAction .= "?" . htmlentities($_SERVER['QUERY_STRING']);
}

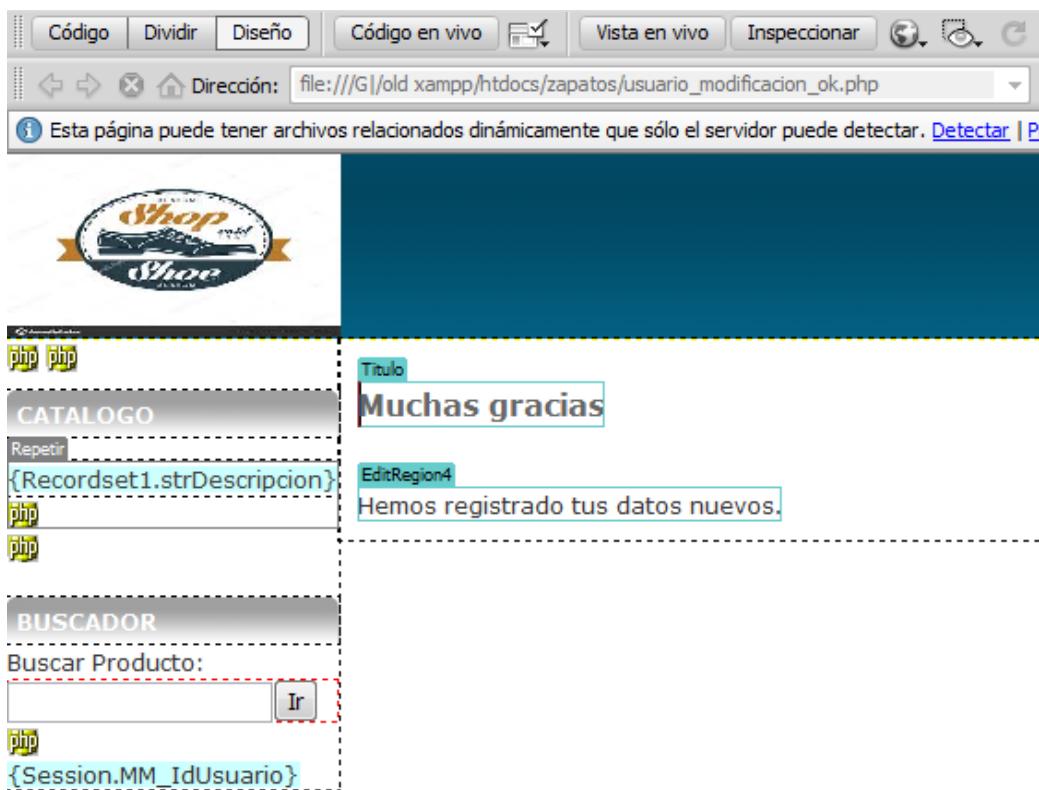
if ((isset($_POST["MM_update"])) && ($_POST["MM_update"] == "form1")) {
    $updateSQL = sprintf("UPDATE tblusuario SET strNombre=%s, strEmail=%s, strPassword=%s WHERE idUsuario=%s",
        GetSQLValueString($_POST['strNombre'], "text"),
        GetSQLValueString($_POST['strEmail'], "text"),
        GetSQLValueString($_POST['strPassword'], "text"),
        GetSQLValueString($_POST['idUsuario'], "int"));

    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $Result1 = mysql_query($updateSQL, $conexionzapatos) or die(mysql_error());

    $updateGoTo = "usuario_modificacion_ok.php";
    if (isset($_SERVER['QUERY_STRING'])) {
        $updateGoTo .= (strpos($updateGoTo, '?')) ? "&" : "?";
        $updateGoTo .= $_SERVER['QUERY_STRING'];
    }
    header(sprintf("Location: %s", $updateGoTo));
}

```

Si todos los pasos anteriores han tenido éxito nos redirecciona a **usuario\_modificacion\_ok.php**, cual nos informara que nuestros cambios se realizaron correctamente.



### 6.3.8 INTRODUCCIÓN DATOS EN CATEGORÍAS

#### categorías\_ver.php

Queremos que al insertar una categoría desde la administración , esta aparezca en nuestra parte publica de nuestra tienda en nuestro menú lateral.

Si recordamos en `includes/catalogo.php` teníamos la consulta para obtener cada uno de los resultados y los repetía con el Do-While. Añadimos una nueva pagina `categorías_ver.php` la cual vincularímos con `catalogo.php` donde nos repetía los resultados para ver todas las categorías .

Como podemos comprobar no tenemos hasta el momento ningún enlace en la pagina `catalogo` mediante lo comentado anteriormente conseguiremos la redireccion la pagina deseada.

Ahora el sistema debe reconocer al seleccionar una categoría a cual esta referida mediante el envío de un parámetro el sistema podrá reconocer al ID de la categoría seleccionada

```
<a href="categoría_ver.php?cat=<?php echo $row_Recordset1['idCategoria']; ?>"
```

Con esto conseguimos que al poner el ratón encima en la parte inferior izquierda nos asocia cada categoría a un ID



Nos disponemos a crear la pagina categorías\_ver.php.Para obtener la lista de categorías en esta pagina vamos a realizar una consulta con el campo categoría que hemos enviado por parámetros para poder obtener los productos listados en esa categoría.

En esta caso el parámetro se pasara por método GET , ya que no proviene de un formulario

```
$varCategoria_DatosProductos = "0";
if (isset($_GET["cat"])) {
    $varCategoria_DatosProductos = $_GET["cat"];
}
mysql_select_db($database_zapatos, $conexionzapatos);
$query_DatosProductos = sprintf("SELECT * FROM tblproducto WHERE tblproducto.intCategoria = %s", GetSQLValueString($varCategoria_DatosProductos, "int"));
$query_limit_DatosProductos = sprintf("%s LIMIT %d, %d", $query_DatosProductos, $startRow_DatosProductos, $maxRows_DatosProductos);
$DatosProductos = mysql_query($query_limit_DatosProductos, $conexionzapatos) or die(mysql_error());
$row_DatosProductos = mysql_fetch_assoc($DatosProductos);
```

Añadimos en la pagina para mostrar imagen,Nombre y precio

```
<?php echo $row_DatosProductos['strImagen']; ?>
<?php echo $row_DatosProductos['strNombre']; ?>.
Precio:<?php echo $row_DatosProductos['dbPrecio']; ?>
```

y mediante un bucle Do-While visualizamos todos los registros disponibles

```
<?php do { ?>
    <?php echo $row_DatosProductos['strImagen']; ?><?php echo $row_DatosProductos['strNombre']; ?>. Precio:<?php echo $row_DatosProductos['dbPrecio']; ?>
<?php } while ($row_DatosProductos = mysql_fetch_assoc($DatosProductos)); ?>
```

Como puede que existan demasiados productos de una determinada categoría vamos a realizar una paginación

El código implementado quedaría:

```
$currentPage = $_SERVER["PHP_SELF"];  
  
$maxRows_DatosProductos = 9;  
$pageNum_DatosProductos = 0;  
if (isset($_GET['pageNum_DatosProductos'])) {  
    $pageNum_DatosProductos = $_GET['pageNum_DatosProductos'];  
}  
$startRow_DatosProductos = $pageNum_DatosProductos * $maxRows_DatosProductos;  
  
$varCategoria_DatosProductos = "0";  
if (isset($_GET["cat"])) {  
    $varCategoria_DatosProductos = $_GET["cat"];  
}  
mysql_select_db($databaseConexionzapatos, $conexionzapatos);  
$query_DatosProductos = sprintf("SELECT * FROM tblproducto WHERE tblproducto.intCategoria = %s", GetSQLValueString($varCategoria_DatosProductos, "int"));  
$query_limit_DatosProductos = sprintf("%s LIMIT %d, %d", $query_DatosProductos, $startRow_DatosProductos, $maxRows_DatosProductos);  
$DatosProductos = mysql_query($query_limit_DatosProductos, $conexionzapatos) or die(mysql_error());  
$row_DatosProductos = mysql_fetch_assoc($DatosProductos);  
  
if (isset($_GET['totalRows_DatosProductos'])) {  
    $totalRows_DatosProductos = $_GET['totalRows_DatosProductos'];  
} else {  
    $all_DatosProductos = mysql_query($query_DatosProductos);  
    $totalRows_DatosProductos = mysql_num_rows($all_DatosProductos);  
}  
$totalPages_DatosProductos = ceil($totalRows_DatosProductos/$maxRows_DatosProductos)-1;
```

También estaría bien que solo nos mostrara cuando existan resultados añadiendo un bucle de control If, ademas que muestre “Todavía no hay productos de esta categoría” si esta vacío el registro, finalmente quedaría el código

```
<?php if ($totalRows_DatosProductos > 0) { // Mostrar si recordset no esta vacio ?>  
    <?php do { ?>  
        <?php echo $row_DatosProductos['strImagen']; ?><?php echo $row_DatosProductos['strNombre']; ?>. Precio:<?php echo $row_DatosProductos['dbPrecio']; ?>  
    <?php } while ($row_DatosProductos = mysql_fetch_assoc($DatosProductos)); ?>  
    <?php } // Mostrar si recordset no esta vacio ?>  
  
<?php if ($totalRows_DatosProductos == 0) { // Mostrar si recordset no esta vacio ?>  
    · Todavia no hay productos de esta categoria  
    <?php } // Mostrar si recordset no esta vacio ?>
```

## VISUALIZACIÓN IMÁGENES

Para poder visualizar las imágenes , en este momento no conseguimos visualizar la imagen sino que solo muestra la ruta y nosotros realmente deseamos visualizar la imagen en cuestión

Vamos a redimensionar a 200 x 200px todas las imágenes con una de las diversas apps existentes

Lo lógico seria pensar en visualizar todos los productos en una tabla y en cada una de las celdas los productos, pero vamos a implementar otra alternativa a mi entender mucho mas profesional y es mediante divs, para ello crearemos 2 divs:

1. <div class="resultadoproductos"></div>
2. <div class="productos"></div>

Creamos una capa o div que contenga completamente todos los productos que vayamos a mostrar

y otra solo para cada producto. Con un echo ya podremos visualizar la imagen y no su ruta, que recordemos la obteníamos del script de subir imagen

```
<div class="resultadoproductos">

    <?php if ($totalRows_DatosProductos > 0) { ?>
    <?php do { ?>
        <div class="producto"> <?php echo $row_DatosProductos['strImagen']; ?>
        <?php echo $row_DatosProductos['strNombre']; ?>.
        Precio:<?php echo $row_DatosProductos['dbPrecio']; ?>
    </div>
    <br />
    <?php } while ($row_DatosProductos = mysql_fetch_assoc($DatosProductos)); ?>
    <?php } ?>
    <?php if ($totalRows_DatosProductos == 0) { ?>
        Todavia no hay productos de esta categoria
    <?php } ?>
</div>
```

Añadimos paginación para una mejor visualización, para mostrar unos 9 archivos por pagina para ello con una tabla y en cada <td> añadimos un If que le pasamos como variables el numero de pagina de DatosProductos y los datos asociados a la consulta de DatosProductos

```
<table border="0">
<tr>
    <td><?php if ($pageNum_DatosProductos > 0) { // Show if not first page ?>
        <a href="php printf("%s?pageNum_DatosProductos=%d%s", $currentPage, 0, $queryString_DatosProductos); ?&gt;"Primero</a>
    <?php } // Show if not first page ?></td>
    <td><?php if ($pageNum_DatosProductos > 0) { // Show if not first page ?>
        <a href="php printf("%s?pageNum_DatosProductos=%d%s", $currentPage, max(0, $pageNum_DatosProductos - 1), $queryString_DatosProductos); ?&gt;"Anterior</a>
    <?php } // Show if not first page ?></td>
    <td><?php if ($pageNum_DatosProductos < $totalPages_DatosProductos) { // Show if not last page ?>
        <a href="php printf("%s?pageNum_DatosProductos=%d%s", $currentPage, min($totalPages_DatosProductos, $pageNum_DatosProductos + 1), $queryString_DatosProductos); ?&gt;"Siguiente</a>
    <?php } // Show if not last page ?></td>
    <td><?php if ($pageNum_DatosProductos < $totalPages_DatosProductos) { // Show if not last page ?>
        <a href="php printf("%s?pageNum_DatosProductos=%d%s", $currentPage, $totalPages_DatosProductos, $queryString_DatosProductos); ?&gt;"&ltimo</a>
    <?php } // Show if not last page ?></td>
</tr>
</table>
```

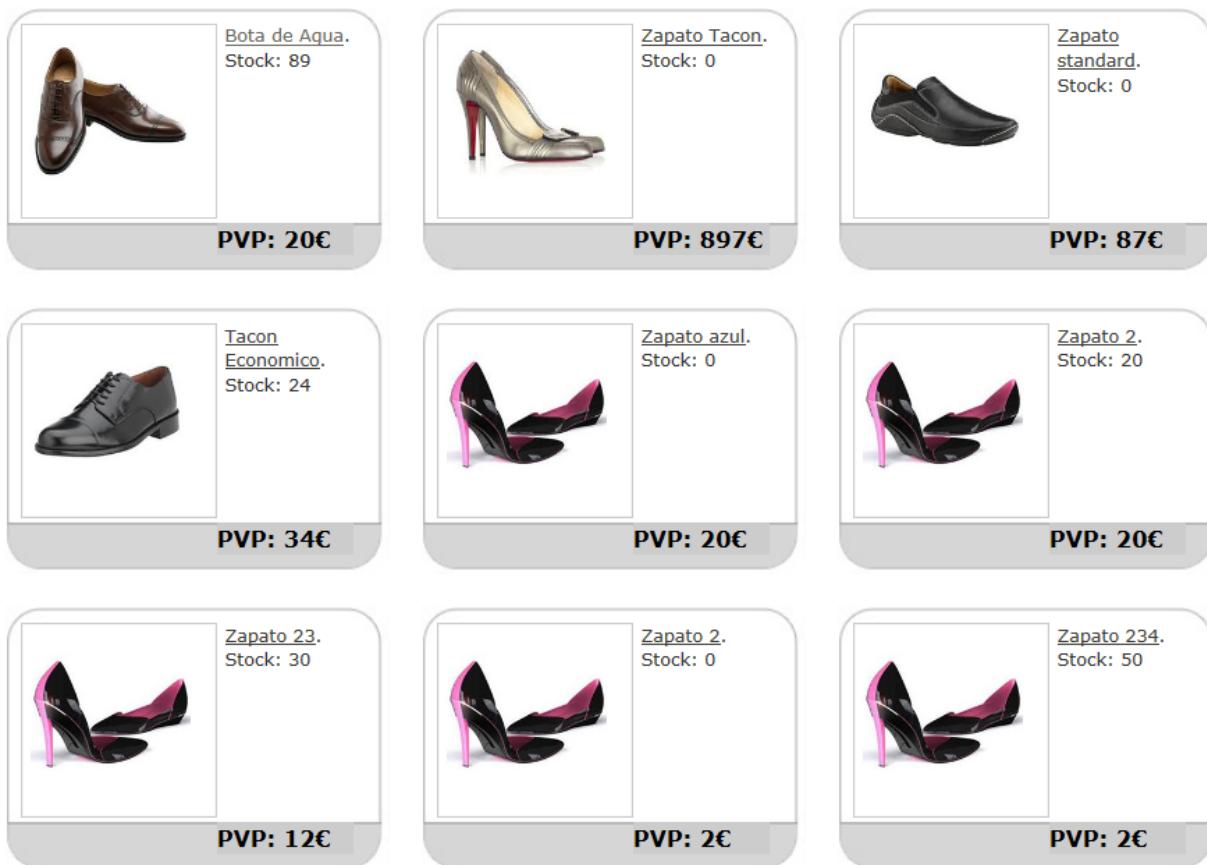
```

$maxRows_DatosProductos = 9;
$pageNum_DatosProductos = 0;
if (isset($_GET['pageNum_DatosProductos'])) {
    $pageNum_DatosProductos = $_GET['pageNum_DatosProductos'];
}
$startRow_DatosProductos = $pageNum_DatosProductos * $maxRows_DatosProductos;

$varCategoria_DatosProductos = "0";
if (isset($_GET["cat"])) {
    $varCategoria_DatosProductos = $_GET["cat"];
}

```

Mediante la variable \$maxRows\_DatosProductos = 9; obtenemos 9 productos por pagina



## 6.3.9 DETALLE DEL PRODUCTO

### ver\_producto.php

Al pulsar sobre un articulo, este nos debería mostrar una información mas detallada del producto, para que así el comprador tenga mas información sobre el producto que desea comprar

Cuando pulsamos sobre un producto el sistema debe reconocer de que producto se trata en la base de datos , de nuevo mediante un parámetro , el sistema reconocerá el registro seleccionado.

Crearemos una nueva pagina llamada ver\_productos.php y realizaremos una consulta SELECT con un parámetro recordID por método GET

```
$varProducto_DatosProducto = "0";
if (isset($_GET["recordID"])) {
    $varProducto_DatosProducto = $_GET["recordID"];
}
mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_DatosProducto = sprintf("SELECT * FROM tblproducto WHERE tblproducto.idProducto = %s",
                                |GetSQLValueString($varProducto_DatosProducto, "int"));

$DatosProducto = mysql_query($query_DatosProducto, $conexionzapatos) or die(mysql_error());
$row_DatosProducto = mysql_fetch_assoc($DatosProducto);
$totalRows_DatosProducto = mysql_num_rows($DatosProducto);
```

Insertamos echo para visualizar estos registros

```
<div class="container">
    <div class="header"><div class="headerinterior"></div></div>
    <div class="subcontenedor">
        <div class="sidebar1">
            <?php include("includes/catalogo.php"); ?>
    <!-- end .sidebar1 --></div>
        <div class="content">
            <h1><!-- InstanceBeginEditable name="Titulo" -->Productos de <?php echo ObtenerNombreCategoria($_GET["cat"]); ?> <!-- InstanceEndEditable --></h1>
            <!-- InstanceBeginEditable name="EditRegion4" -->
            <div class="resultadoproductos">
                <?php if ($totalRows_DatosProductos > 0) { // Show if recordset not empty ?>
                    <?php do { ?>
                        ...<div class="producto">
                            ...<div class="fotoproducto"></div>
                            ...<div class="productoderecha">
                                ...<div class="textoproducto"><a href="ver_producto.php?recordID=<?php echo $row_DatosProductos['idProducto']; ?>">
                                    <?php echo $row_DatosProductos['strNombre']; ?></a>.
                                ...</div>
                            ...</div>
                        ...</div>
                    ...<?php } while ($row_DatosProductos = mysql_fetch_assoc($DatosProducto)); ?>
                </div>
            <!-- InstanceEndEditable -->
        </div>
    </div>
</div>
```

Visualmente quedaría :



Bota de Agua

20 €

Tallas:

Unidades:

Necesitas darte de alta para comprar.  
Es gratuito.

### 6.3.10 CARRITO DE COMPRA

#### carrito\_lista.php

Como toda pagina de venta Online vamos a necesitar un botón de compra ,y por supuesto que solo puedan comprar aquellas personas que estén registradas en el sistema.

El primer problema que se presenta es que el botón de comprar producto solo lo vean aquellas personas que estén registradas en el sistema, para ello miramos cual era la variable de sesión. que se le asignaba a un usuario que esta dado de alta y ha accedido al sistema en nuestro caso :

**`$_SESSION['MM_IdUsuario']`**

Con esta variable en la pagina **ver\_producto.php** podemos asignar que vea el botón Comprar cuando el cliente ha accedido al sistema y que solamente vea Darse de alta cuando no lo esta .

Con un

```
<?php if ((isset( $_SESSION['MM_IdUsuario'])) && ($_SESSION['MM_IdUsuario'] != "0")) ?>
```

donde **isset** nos dirá si el valor esta definido

Esto quiere decir si la sesión. de usuario esta conectada o a set(isset) y ademas no tiene el valor 0 vamos a dar por hecho que esta persona ha accedido al sistema

```


|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  </td> <td align="top">&lt;p&gt;&lt;?php echo \$row_DatosProducto['strNombre']; ?&gt;&lt;/p&gt; &lt;p&gt;&lt;?php echo \$row_DatosProducto['dblPrecio']; ?&gt;&lt;/p&gt; &lt;?php if ((isset(\$_SESSION['MM_IdUsuario'])) &amp;&amp; (\$_SESSION['MM_IdUsuario']!="0")) {? &lt;p&gt;&lt;a href="carrito_add.php?recordID=&lt;?php echo \$row_DatosProducto['idProducto']; ?&gt;"&gt;Comprar Producto&lt;/a&gt;&lt;/p&gt; &lt;?php } else {?     Necesitas &lt;a href="alta_usuario.php"&gt;darte de alta&lt;/a&gt; para comprar. &lt;?php }?&gt; &lt;/p&gt; &lt;p&gt; &lt;?php echo \$row_DatosProducto['strDescripcion']; ?&gt;&lt;/p&gt;&lt;/td&gt; </td> | <p><?php echo \$row_DatosProducto['strNombre']; ?></p> <p><?php echo \$row_DatosProducto['dblPrecio']; ?></p> <?php if ((isset(\$_SESSION['MM_IdUsuario'])) && (\$_SESSION['MM_IdUsuario']!="0")) {? <p><a href="carrito_add.php?recordID=<?php echo \$row_DatosProducto['idProducto']; ?>">Comprar Producto</a></p> <?php } else {?     Necesitas <a href="alta_usuario.php">darte de alta</a> para comprar. <?php }?> </p> <p> <?php echo \$row_DatosProducto['strDescripcion']; ?></p></td> |
| &nbsp;</td> <td>&amp;nbsp;&lt;/td&gt; </td>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | &nbsp;</td>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |


```

[Botas](#)  
[Botas de Campo](#)  
[Zapatillas](#)  
[Zapatos](#)  
[Zapatos de tacón](#)  
**Hola Fran**  
[Modificar](#) - [Salir](#)

## Botas verdes



[Botas verdes](#)

54

[Comprar Producto](#)

Botas color verde tallas todas

Para que cuando pulsemos nos lleve a una pagina donde muestre lo que hemos comprado creamos **carrito\_add.php**. Al seleccionar comprar al enlace le pasaremos el parámetro recordID para que el sistema reconozca que producto estamos comprando

```

<p><a href="carrito_add.php?recordID=<?php echo $row_DatosProducto['idProducto']; ?>">Comprar Producto</a></p>
<p>
```

Para este carrito de la compra hemos de añadir una tabla a nuestra base de datos llamada tblcarrito

El campo mas destacable seria “intTransaccionEfectuada” el cual utilizaremos para saber si un usuario ha efectuado una compra o no, le asignaremos los siguientes valores:

- 0 → Si no ha efectuado la compra todavía.
- 1 → Si ya ha efectuado la compra y pasara al historial de compras.

El propósito es que cuando pulsemos comprar inserte el producto en el carrito de la compra para tener la típica pagina con el listado de objetos que deseamos comprar y posteriormente se efectué el pago como veremos.

Para ello creamos una nueva pagina **carrito\_add.php** donde vamos a ejecutar una sentencia INSERT igual que hicimos en categorías\_add.php y en productos\_add.php así nos valdría el código pero cambiando algunas variables.

```
$insertSQL = sprintf("INSERT INTO tblcarrito (idUsuario, idProducto, intCantidad) VALUES (%s, %s, %s)",  
    GetSQLValueString($_SESSION['MM_IdUsuario'], "int"),  
    GetSQLValueString($_GET['recordID'], "text"),1);  
  
mysql_select_db($databaseConexionzapatos, $conexionzapatos);  
$result1 = mysql_query($insertSQL, $conexionzapatos) or die(mysql_error());  
  
$insertGoTo = "carrito_lista.php";  
if (isset($_SERVER['QUERY_STRING'])) {  
    $insertGoTo .= (strpos($insertGoTo, '?')) ? "&" : "?";  
    $insertGoTo .= $_SERVER['QUERY_STRING'];  
}  
header(sprintf("Location: %s", $insertGoTo));  
?>
```

A la variable **intCantidad** en principio le ponemos 1 ya que de momento solo vamos a comprar los productos de 1 en 1 ya solucionaremos esto mas adelante para poder comprar mas de un producto simultáneamente.

Mediante esta INSERT vamos a introducir en nuestra BD los registros idUsuario, idProducto, intCantidad

Como hemos comentado infinidad de veces le hemos pasado un parámetro para que el sistema reconozca que estamos seleccionando; Es decir tomara los valores de la variable

**GetSQLValueString(\$\_SESSION['MM\_IdUsuario'], "int"),**

La cual informara al sistema de que idUsuario esta conectado al sistema y al ser una variable de tipo \$\_SESSION perdurará el valor mientras exista la sesión.

Y por ultimo

**GetSQLValueString(\$\_GET['recordID'], "text"),1);** GET → paso por parámetro y nos redirecciona a la pagina **carrito\_lista.php** si la INSERT se ha efectuado correctamente

header(sprintf("Location: %s", \$insertGoTo));

Mediante este proceso hemos conseguido insertar nuestro primer producto en el carrito de compra.

He observado un problema que es que al cerrar la sesión, me sigue saliendo la opción comprar cuando no debería ser así para ello tendremos que modificar en **ver\_producto.php**

```
<?php if ((isset($_SESSION['MM_IdUsuario'])) && ($_SESSION['MM_IdUsuario']!="0"))|
```

Este era el código que implementamos para ello, el error esta en:

```
($_SESSION['MM_IdUsuario']!="0"))
```

Ya que si miramos como confeccionamos el cierre de sesión. en la pagina **usuario\_cerrarsesion.php** teníamos que para finalizar la sesión. establecíamos a valor nulo las variables de sesión.

```
$_SESSION['MM_Username'] = "";  
$_SESSION['MM_UserGroup'] = "";  
$_SESSION['MM_IdUsuario'] = ""; ?>
```

Cambiando el valor “0” por “ ” en **ver\_producto.php** corregimos esto

```
<?php if ((isset($_SESSION['MM_IdUsuario'])) && ($_SESSION['MM_IdUsuario']!=""))
```

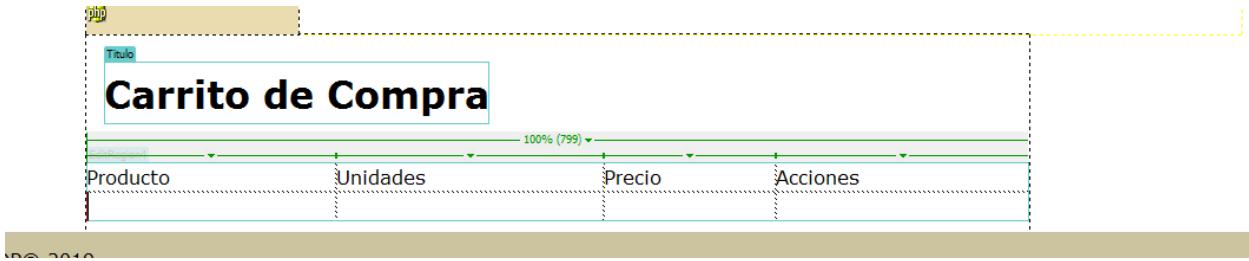
Con esta modificación resolvemos el error

Seguimos completando nuestro carrito de la compra para poder visualizar los productos antes de efectuar la compra , este carrito de la compra sería el equivalente al proceso de emitir una factura, ya que vamos a necesitar datos del usuario, del producto y de su categoría.

A continuación vamos a obtener los datos del carrito como le hemos llamado para ello vamos a realizar una consulta SELECT en **carrito\_lista.php** a la BD donde estamos almacenando los productos comprados → tblcarrito para obtener dicho registros pasamos por parámetro el valor idUsuario

```
$varUsuario_DatosCarrito = "0";  
if (isset($_SESSION["MM_IdUsuario"])) {  
    $varUsuario_DatosCarrito = $_SESSION["MM_IdUsuario"];  
}  
mysql_select_db($databaseConexionzapatos, $conexionzapatos);  
$query_DatosCarrito = sprintf("SELECT * FROM tblcarrito WHERE tblcarrito.idUsuario = %s AND tblcarrito.  
    intTransaccionEfectuada = 0", GetSQLValueString($varUsuario_DatosCarrito, "int"));  
$DatosCarrito = mysql_query($query_DatosCarrito, $conexionzapatos) or die(mysql_error());  
$row_DatosCarrito = mysql_fetch_assoc($DatosCarrito);  
$totalRows_DatosCarrito = mysql_num_rows($DatosCarrito);
```

**intTransaccionEfectuada=0**, ya que todavía. no se ha completado la compra.  
Añadiendo una tabla para visualizar estos registros



Pero surge otro problema con esta consulta obtenemos el idProducto el cual no nos interesa en principio, nosotros queremos visualizar el nombre del producto no su ID asociado.

Para ello creamos una función en **includes/funciones.php** (Si realizáramos una consulta mas compleja de mas de una tabla si que podríamos obtener este registro) para vemos a simplificar el trabajo dado que no me quiero meter en consultas multitabla de momento

```
function ObtenerNombreProducto($identificador)
{
    global $database_conexionzapatos, $conexionzapatos;
    mysql_select_db($database_conexionzapatos, $conexionzapatos);
    $query_ConsultaFuncion = sprintf("SELECT strNombre FROM tblproducto
    WHERE idProducto = %s", $identificador);
    $ConsultaFuncion = mysql_query($query_ConsultaFuncion, $conexionzapatos) or die(mysql_error());
    $row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion);
    $totalRows_ConsultaFuncion = mysql_num_rows($ConsultaFuncion);

    return $row_ConsultaFuncion['strNombre'];
    mysql_free_result($ConsultaFuncion);
}
```

De momento no consigo que funcione así que vamos a ver que error esta ocurriendo para ello podemos visualizar los errores mediante un **echo \$query\_DatosCarrito**, justo a continuación de la consulta realizada

localhost/zapatos/carrito\_listar.php?recordID=7

SELECT \* FROM tblcarrito WHERE tblcarrito.idUsuario = 0 AND tblcarrito.intTransaccionEfectuada = 0

Producto	Unidades	Precio	Acciones
			Eliminar

Se

Vemos claramente que se esta realizando una SELECT donde idUsuario=0 , si nos vamos al Xampp y consultamos la tabla

SELECT \* FROM `tblcarrito`

Perfilando [ En línea ] [ Ed ]

Número de filas: 25 ▾

Ordenar según la clave: Ninguna ▾

Opciones

HT → intContador idUsuario idProducto intCantidad intTransaccionEfectuada

		intContador	idUsuario	idProducto	intCantidad	intTransaccionEfectuada
<input type="checkbox"/>	Editar  Copiar  Borrar	1	4	1	1	0
<input type="checkbox"/>	Editar  Copiar  Borrar	2	4	7	1	0

↑  Marcar todos Para los elementos que están marcados: Cambiar Borrar Exportar

Número de filas: 25 ▾

Operaciones sobre los resultados de la consulta

el idUsuario es 4

Tras mirar detenidamente el código el error radica en el nombre del parámetro yo puse **MM\_idUsuario** mientras que lo correcto seria **MM\_IdUsuario**, con esto ya conseguimos visualizarlo

Pero seguimos manteniendo el error que solo me muestra una linea de producto cuando en realidad hemos comprado mas de 1 producto deberemos repetir los registro mediante un bucle Do-while

```

<?php do {
?>
    <tr>
        <td><?php echo $row_DatosCarrito['idProducto']; ?></td>
        <td><?php echo $row_DatosCarrito['intCantidad']; ?></td>
        <td><?php echo $row_DatosCarrito['idProducto']; ?></td>
        <td>Eliminar</td>
    </tr>
<?php }
while ($row_DatosCarrito = mysql_fetch_assoc($DatosCarrito));
?>

```

**Carrito de Compra**

Producto	Unidades	Precio	Acciones
Zapatos de tacón Hola Fran	1 7	1 7	Eliminar Eliminar Eliminar

Retornamos con el error para mostrar el nombre del producto no el ID así que vamos a insertar una nueva función que obtenga el nombre del producto en el archivo **includes/funciones.php** nos hacemos valer de la función que implementamos para obtener el nombre de un usuario registrado con solo modificar la consulta ya lo obtendríamos

```

function ObtenerNombreProducto($identificador)
{
    global $databaseConexionZapatos, $conexionZapatos;
    mysql_select_db($databaseConexionZapatos, $conexionZapatos);
    $query_ConsultaFuncion = sprintf("SELECT strNombre FROM tblproducto
    WHERE idProducto = %s", $identificador);
    $ConsultaFuncion = mysql_query($query_ConsultaFuncion, $conexionZapatos) or die(mysql_error());
    $row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion);
    $totalRows_ConsultaFuncion = mysql_num_rows($ConsultaFuncion);

    return $row_ConsultaFuncion['strNombre'];
    mysql_free_result($ConsultaFuncion);
}
?>

```

**sprintf** → Sustituye cada una de las variables por la indicada a la derecha en este caso **\$identificador**.

Esta función ya nos devuelve el nombre del producto de cuyo identificador queremos averiguar.

Ahora tenemos que llamar a la función en nuestra pagina **carrito\_lista.php**

```
<table width="100%" border="0" cellspacing="0" cellpadding="0">
<tr>
<td>Producto</td>
<td>Unidades</td>
<td>Precio</td>
<td>Acciones</td>
</tr>
<?php do { ?>
<tr>
<td><?php echo ObtenerNombreProducto ($row_DatosCarrito['idProducto']); ?></td>
<td><?php echo $row_DatosCarrito['intCantidad']; ?></td>
<td><?php echo $row_DatosCarrito['idProducto']; ?></td>
<td>Eliminar</td>
</tr>
<?php } while ($row_DatosCarrito = mysql_fetch_assoc($DatosCarrito)); ?>
</table>
```

y nos mostraría por pantalla los nombres de los productos

Producto	Unidades	Precio	Acciones
Bota de Agua	1	1	Eliminar
Tacon Economico	1	7	Eliminar
Tacon Economico	1	7	Eliminar

para mostrar el precio realizamos la misma acción, una nueva función que obtenga el valor dblPrecio en **funciones.php**

```
function ObtenerPrecioProducto($identificador)
{
    global $databaseConexionZapatos, $conexionZapatos;
    mysql_select_db($databaseConexionZapatos, $conexionZapatos);
    $query_ConsultaFuncion = sprintf("SELECT dblPrecio FROM tblproducto
WHERE idProducto = %s", $identificador);
    $ConsultaFuncion = mysql_query($query_ConsultaFuncion, $conexionZapatos) or die(mysql_error());
    $row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion);
    $totalRows_ConsultaFuncion = mysql_num_rows($ConsultaFuncion);

    return $row_ConsultaFuncion['dblPrecio'];
    mysql_free_result($ConsultaFuncion);
}
```

Producto	Unidades	Precio	Acciones
Bota de Agua	1	20	Eliminar
Tacon Economico	1	34	Eliminar
Tacon Economico	1	34	Eliminar
Zapato standard	1	87	Eliminar

## CALCULO TOTAL PEDIDO

El cliente deberá conocer el precio total de su compra. En la pagina **carrito\_lista.php**, añadimos una nueva fila con el Total

Producto	Unidades	Precio	Acciones
{DatosCarrito.idProducto}	{DatosCarrito.intCantidad}	{DatosCarrito.idProducto} Euros	Eliminar
Total:			

Producto	Unidades	Precio	Acciones
Bota de Agua	1	20Euros	Eliminar
Tacon Economico	1	34Euros	Eliminar
Tacon Economico	1	34Euros	Eliminar
Zapato standard	1	87Euros	Eliminar
Total:			

Vemos que existe un error ya que se nos repite el Total esto es porque lo hemos puesto dentro del Bucle Do-While hemos de sacarlo fuera del bucle

Producto	Unidades	Precio	Acciones
Bota de Agua	1	20Euros	Eliminar
Tacon Economico	1	34Euros	Eliminar
Tacon Economico	1	34Euros	Eliminar
Zapato standard	1	87Euros	Eliminar
Total:			

Para calcular el total vamos a declarar una nueva variable y la inicializamos a 0, **\$preciototal = 0** la cual debe ir incrementando cada vez que se añade un producto y necesitaremos sacarla por pantalla

```
<?php $preciototal = $preciototal + ObtenerPrecioProducto ($row_DatosCarrito['idProducto']);?>
```

Añadiendo un echo de la variable \$preciototal visualizamos el valor

Zapatos	Producto	Unidades	Precio	Acciones
Zapatos de tacón	Bota de Agua	1	20 Euros	Eliminar
Hola Fran	Tacon Economico	1	34 Euros	Eliminar
<a href="#">Modificar</a> - <a href="#">Salir</a>	Tacon Economico	1	34 Euros	Eliminar
	Zapato standard	1	87 Euros	Eliminar
			Total:175 Euros	

SHOP® 2019

## CALCULO DE IMPUESTOS

También seria interesante que nos calculase el IVA sobre el precio total, para ello creamos una nueva tabla **tblvariables** donde almacenaremos estos valores no lo ponemos directamente en el código ya que si mañana cambia el valor del IVA será mas sencillo modificarlo de esta manera

Vamos a suponer que vamos a obtener el valor de los productos sin IVA inicialmente y luego se le aplicara el 21%

De nuevo vamos a obtener el valor del IVA de funciones.php en la cual vamos a crear una nueva función ObtenerIVA(), la cual no va a ser necesario pasar la variable \$indentificador ya que es un valor que tenemos en las tablas , sprintf tampoco sera necesario ya que no le estamos pasando ninguna variable. Esta consulta es un poco singular ya que en where vamos a seleccionar idContador=1 que coincide con el valor del registro deseado 21% de IVA

```
function ObtenerIVA()
{
    global $databaseConexionZapatos, $conexionZapatos;
    mysql_select_db($databaseConexionZapatos, $conexionZapatos);
    $query_ConsultaFuncion = "SELECT intIVA FROM tblvariables WHERE idContador = 1";
    $ConsultaFuncion = mysql_query($query_ConsultaFuncion, $conexionZapatos) or die(mysql_error());
    $row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion);
    $totalRows_ConsultaFuncion = mysql_num_rows($ConsultaFuncion);

    return $row_ConsultaFuncion['intIVA'];
    mysql_free_result($ConsultaFuncion);
}
```

Con echo visualizamos el valor

```
<tr>
    <td>&ampnbsp</td>
    <td align="right">IVA:</td>
    <td><?php echo ObtenerIVA(); ?>% </td>
    <td>&ampnbsp</td>
</tr>
```

Ahora necesitamos calcular el Total + IVA

Para calcular el IVA al 21% de un producto necesitamos calcular el 1,21\*precio del producto

sabemos que en la tabla el valor intIva=21, así que necesitaremos definir para realizar el calculo lo siguiente

```
<?php  
    $multiplicador = (100 + ObtenerIVA())/100;  
    $valorconiva = $preciototal * $multiplicador  
?>
```

**\$Multiplicador** → Obtiene el 1,21.

**\$valorconiva** → Obtendrá el valor del producto con un 21% de IVA

```
<?php  
    $multiplicador = (100 + ObtenerIVA())/100;  
    $valorconiva = $preciototal * $multiplicador  
    <?php echo $valorconiva; ?>  
?>
```

```
<?php  
    $multiplicador = (100 + ObtenerIVA())/100;  
    $valorconIVA = $preciototal * $multiplicador;  
    $_SESSION["TotalCompra"] = $valorconIVA;  
    echo $valorconIVA;?> Euros</td>
```

Hemos desglosado el precio el valor del IVA sobre ese precio y el total del precio con IVA

Finalmente queda

```

<tr>
<td>&ampnbsp</td>
<td align="right">Subtotal:</td>
<td><?php echo $preciototal; ?> Euros</td>
<td>&ampnbsp</td>
</tr>
<tr>
<td>&ampnbsp</td>
<td align="right">IVA:</td>
<td><?php echo ObtenerIVA(); ?>%</td>
<td>&ampnbsp</td>
</tr>
<tr>
<td>&ampnbsp</td>
<td align="right">Valor del IVA:</td>
<td><?php
$multiplicador = ObtenerIVA()/100;
$valordelIVA = $preciototal * $multiplicador;
echo $valordelIVA; ?> Euros</td>
<td>&ampnbsp</td>
</tr>
<tr>
<td>&ampnbsp</td>
<td align="right">Total con IVA:</td>
<td><?php
$multiplicador = (100 + ObtenerIVA())/100;
$valorconIVA = $preciototal * $multiplicador;
echo $valorconIVA; ?> Euros</td>
<td>&ampnbsp</td>
</tr>

```

Y este seria el resultado por pantalla

Zapatillas	Producto	Unidades	Precio	Acciones
Zapatos	Bota de Agua	1	20 Euros	Eliminar
Zapatos de tacón	Tacon Economico	1	34 Euros	Eliminar
Hola Fran	Tacon Economico	1	34 Euros	Eliminar
<a href="#">Modificar</a> - <a href="#">Salir</a>	Zapato standard	1	87 Euros	Eliminar
Subtotal:175 Euros				
IVA:21%				
Valor del IVA:36.75 Euros				
Total con IVA:211.75 Euros				

HOP® 2019

Continuamos creando un enlace para cuando estemos en el detalle de un producto ([ver\\_producto.php](#)) poder acceder al carrito .

El carrito de la compra se visualiza en **carrito\_lista.php**, para ello en catalogo.php(includes) y situando el enlace dentro del bucle de control para cuando esta conectado un usuario

```

<?php
if ((isset($_SESSION['MM_Username'])) && ($_SESSION['MM_Username'] != ""))
{
    echo "Hola ";
    echo ObtenerNombreUsuario($_SESSION['MM_IdUsuario']);
}
<br />
<a href="carrito_lista.php" class="modificacionusuario">Carrito de la Compra</a><br />

<a href="usuario_modificar.php" class="modificacionusuario">Modificar</a> - <a href="usuario_cerrarsesion.php" class="modificacionusuario">Salir</a>
<p><a href="alta_usuario.php">Darme de Alta</a></p>
<p><a href="acceso.php">Acceder</a></p>
<?php }?>

```

Este seria el resultado:

The screenshot shows a web browser window. At the top, the address bar displays "localhost/zapatos/ver\_producto.php?recordID=1". Below the address bar, there's a navigation bar with links: "asix", "simracing", "leds", "Google Code Archi...", "Activadores", "Sistema de usuarios...", and a "P" icon. The main content area has a yellow sidebar on the left containing a user profile: "Botas", "De montaña", "De playa", "Sandalias", "Hola Frank", and "Carrito de la Compra". To the right of the sidebar, the main content area features a large title "Bota de Agua" and a small image of a boot.

### 6.3.11 FORMA DE PAGO

#### carrito\_forma\_pago.php

Continuamos añadiendo opciones a nuestra tienda web para que pueda ser mínimamente funcional

Vamos a considerar 3 formas de pago :

1. PayPal
2. Transferencia Bancaria
3. VISA/MasterCard

Creamos un link desde **carrito\_lista.php** que direccional a **carrito\_forma\_pago.php** cuando pulsamos Seleccionar forma de pago

Mediante un formulario, tipo radio , donde podremos seleccionar una de las 3 opciones

```

<form id="form1" name="form1" method="post" action="carrito_finalizacion.php">
    <p>
        <input name="radio" type="radio" id="radio" value="1" checked="checked" />
        <label for="radio">PayPal</label><br />
    <input type="radio" name="radio" id="radio" value="2" />
        <label for="radio">Transferencia</label>
        <br />
    <input type="radio" name="radio" id="radio" value="3" />
        <label for="radio">VISA/Mastercard</label>
    </p>
    <p>
        <input type="submit" name="button" id="button" value="Pagar" />
    </p>
    <p>&nbsp;</p>
</form>

```

Este seria el aspecto:



Establecemos valores para cada radio button identificando la forma de pago:

- PayPal → Value=1
- Transferencia → Value=2
- VISA/MasterCard → Value=3

## TRANSFERENCIA BANCARIA

### **carrito\_finalizacion.php**

Cuando seleccionemos el botón pagar se tiene que iniciar la compra y efectuar cambios en nuestra BD, Al pulsar nos llevara a la pagina **carrito\_finalizacion.php**

En la cual vamos a pasar la variable por (POST → Formulario) [Las variables tipo POST no se verán en la barra navegación] recogida del formulario mediante un If pasando la variable radio cuando sea igual a 2 que es el valor al seleccionar Transferencia bancaria.

```

<?php if ($_POST["radio"] == 2)
{
    ConfirmacionPago($_POST["radio"]);
?>
    <p>Has elegido pago por transferencia.</p>
    <p>Deber&aacute;s remitirnos un email con el justificante de pago a zapatos@zapatos.com, realizado a este n&uacute;mero de cuenta:</p>
    <p>IBAN: 12341234123412341234</p>

<?php

```

El administrador debe conocer que esa compra esta inicializada para poder luego confirmar compra cuando reciba la transferencia o bien cancelarla por otros motivos.

Para manejar esto vamos a crear una nueva tabla **tblcompra**

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra
1	<b>idCompra</b>	int(11)			No	Ninguna	AUTO_INCREMENT
2	<b>idUsuario</b>	int(11)			Sí	NULL	
3	<b>fchCompra</b>	datetime			Sí	NULL	
4	<b>intTipoPago</b>	int(11)			Sí	NULL	
5	<b>dblTotal</b>	double			Sí	NULL	
6	<b>intEstado</b>	int(11)			No	0	

Al pulsar transferencia en la tabla carrito tendrá que variar el campo **intTransaccionEfectuada** de 0 a 1

Mediante la función **ConfirmacionPago** que nos realice un INSERT en la tabla compra

```

function ConfirmacionPago($tipopago)
{
    global $databaseConexionzapatos, $conexionzapatos;
    mysql_select_db($databaseConexionzapatos, $conexionzapatos);

    $insertSQL = sprintf("INSERT INTO tblcompra (idUsuario, fchCompra, intTipoPago, dblTotal) VALUES (%s, NOW(), %s, %s)",
        GetSQLValueString($_SESSION['MM_IdUsuario'], "int"), $tipopago, $_SESSION["TotalCompra"]);

    $result1 = mysql_query($insertSQL, $conexionzapatos) or die(mysql_error());
    $ultimacompra = mysql_insert_id();
    ActualizacionCarrito($ultimacompra);
}

```

Aquí hay que detenernos y comentar que hemos añadido una variable de sesión. TotalCompra, la cual vamos a necesitar para efectuar las operación de pago.

- **\$\_SESSION["TotalCompra"];**

Al realizar los pagos fuera de nuestra pagina por medio de un sistema externo de pago, necesitamos tanto la variable de sesión. de usuario como el total de la compra, para asociar el importe a un

usuario en particular y cargarle el pago a este usuario y no otro

Haciendo uso de la función **mysql\_insert\_id()**; la cual selecciona el ultimo id que corresponde con la ultima compra a no ser que 2 personas compren el mismo producto en la misma fracción de segundo algo altamente improbable

Mediante la sentencia INSERT ya se introducen los datos en la tabla **tblcompra**. Ahora necesitamos conocer si la compra ya esta realizada y pase a otro estado el valor **intTransaccionEfectuada** mediante un UPDATE la cual la crearemos de nuevo en una función **ActualizacionCarrito()**

```
function ActualizacionCarrito($varcompra)
{
    global $databaseConexionzapatos, $conexionzapatos;
    $updateSQL = sprintf("UPDATE tblcarrito SET intTransaccionEfectuada=%s WHERE idUsuario=%s AND intTransaccionEfectuada = 0",
        $varcompra, $_SESSION['MM_IdUsuario']);
    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $Result1 = mysql_query($updateSQL, $conexionzapatos) or die(mysql_error());
}

//*****
```

El ultimo paso sera pasar el valor de la variable de sesión. del total de la compra.

Con esto ya tendríamos la compra por transferencia efectuada, a falta de posteriormente realizar un envío anunciando la compra.

Como te comente al final del punto 6.2.13 requerimos volver a implementar una parte de la administración ya que la forma de pago va a requerir modificaciones en la administración, y por no hacer el indice interminable lo comentamos en este punto

## VISUALIZACIÓN COMPRAS POR PARTE DEL ADMINISTRADOR

### compras\_listado.php

Otro aspecto importante es que el administrador pueda ver un listado de las compras que han efectuado .Realizaremos pasos similares a cuando obtuvimos la lista de productos le llamaremos a esta nueva pagina **compras\_listado.php**

The screenshot shows the administration interface of the Elite Feet website. At the top, there is a logo for 'ELITE FEET' and a navigation bar with links for 'Administracion' (Administration), 'Compras' (Purchases), and 'Listado' (List). The main content area has a title 'Lista de Compras' (Purchase List). Below the title is a table with the following columns:

RE PERSONA (Recordset1.idUsuario)	FECHA (Recordset1.fchCompra)	FORMA DE PAGO (Recordset1.intTipoPago)	ACCIONES Ver

At the bottom of the page, there is a footer bar with the text 'Administracion Tienda Zapatos'.

Con una sentencia SELECT obtenemos los registros:

```

$maxRows_Recordset1 = 10;
$pageNum_Recordset1 = 0;
if (isset($_GET['pageNum_Recordset1'])) {
    $pageNum_Recordset1 = $_GET['pageNum_Recordset1'];
}
$startRow_Recordset1 = $pageNum_Recordset1 * $maxRows_Recordset1;

mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_Recordset1 = "SELECT * FROM tblcompra ORDER BY fchCompra DESC";
$query_limit_Recordset1 = sprintf("%s LIMIT %d, %d", $query_Recordset1, $startRow_Recordset1, $maxRows_Recordset1);
$Recordset1 = mysql_query($query_limit_Recordset1, $conexionzapatos) or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($Recordset1);

if (isset($_GET['totalRows_Recordset1'])) {
    $totalRows_Recordset1 = $_GET['totalRows_Recordset1'];
} else {
    $all_Recordset1 = mysql_query($query_Recordset1);
    $totalRows_Recordset1 = mysql_num_rows($all_Recordset1);
}
$totalPages_Recordset1 = ceil($totalRows_Recordset1/$maxRows_Recordset1)-1;

```

Para que nos devuelva el valor del tipo de transacción efectuada necesitaremos otra función llamada `TextoFormaPago()`, con la llamada a esta obtendremos el nombre tipo pago en función del tipo de pago

```

function TextoFormaPago($vartipopago)
{
    if ($vartipopago == 1) return "PayPal";
    if ($vartipopago == 2) return "Transferencia";
    if ($vartipopago == 3) return "VISA/Mastercard";
}

```

Con una tabla para poder visualizar los datos con `echo`

```

<?php do { ?>
<tr class="brillo">
    <td><?php echo ObtenerNombreUsuario($row_Recordset1['idUsuario']); ?></td>
    <td><?php echo $row_Recordset1['fchCompra']; ?></td>
    <td><?php echo TextoFormaPago($row_Recordset1['intTipoPago']); ?></td>
    <td><a href="compras_edit.php?recordID=<?php echo $row_Recordset1['idCompra']; ?>">Ver</a></td>
</tr>
<?php } while ($row_Recordset1 = mysql_fetch_assoc($Recordset1)) ?>
</table>
<!-- InstanceEndEditable -->

```

## GESTIÓN COMPRA ADMINISTRACIÓN

### compras\_edit.php.

En **compras\_lista.php** al seleccionar el enlace Ver, deseamos poder visualizar la compra realizada de cada cliente con mas detalle, y ahí poder aceptarla o cancelarla. Al pulsar sobre Ver nos redirecciona a **compras\_edit.php**.

Hemos de tener en cuenta que desde **compras\_lista.php** nos esta llegando una variable por url llamada recordID, que nos pasa el idCompra .

Para obtener los registros que nos interesan vamos a realizar la siguiente consulta SELECT :

```
varCompra_Datoscompra = "0";
if (isset($_GET["recordID"])) {
    $varCompra_Datoscompra = $_GET["recordID"];
}

mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_Datoscompra = sprintf("SELECT * FROM tblcompra WHERE tblcompra.idCompra = %s",
    GetSQLValueString($varCompra_Datoscompra, "int"));
$Datoscompra = mysql_query($query_Datoscompra, $conexionzapatos) or die(mysql_error());
$row_Datoscompra = mysql_fetch_assoc($Datoscompra);
$totalRows_Datoscompra = mysql_num_rows($Datoscompra);
```

En la cual le pasamos por parámetro a varCompra → \$\_GET[“recordID”], ya con esta consulta podremos obtener los datos de la cabecera necesarios

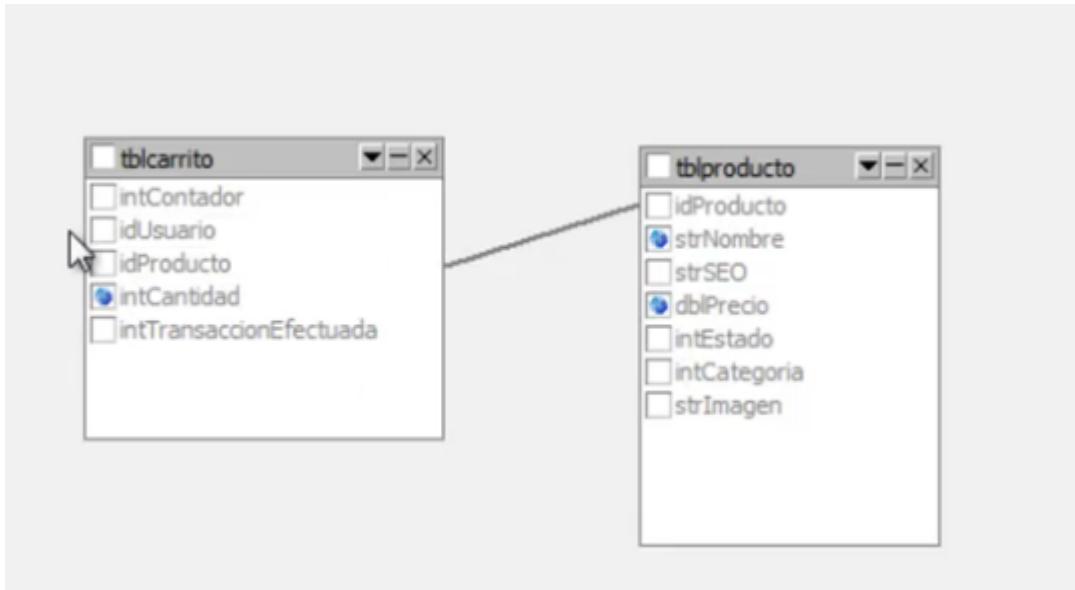
The screenshot shows a web page with a header 'Contenido'. Below it is a section titled 'Consultar Compra'. Inside this section, there are four input fields with placeholder text: 'Nombre: {Datoscompra.idUsuario}', 'Fecha: {Datoscompra.fchCompra}', 'Forma de Pago: {Datoscompra.intTipoPago}', and 'Total: {Datoscompra dblTotal}'.

Con esta consulta hemos podido obtener algunos registros, pero no todos los necesarios. Aquí requerimos también de una lista de los productos que ha comprado, para ello necesitamos la compra realizada y la persona asociada a ella, este listado lo obtendremos mediante una consulta que obtenga idUsuario y el idCompra, en este caso tenemos una consulta con 2 variables que le vamos a pasar por parámetros:

- \$\_GET[“recordID”]
- \$\_GET[“idUsuario”]

Ademas de esto también necesitamos el precio del producto,ya que no lo tenemos en la tabla consultada tblcarrito.

Analicemos las tablas y veamos como se relacionan



Al tratarse de una consulta multitabla en este caso entre 2 tablas podemos usar una SELECT con INNER JOIN

```
$varCarrito_ProductosCompra = "0";
if (isset($_GET["recordID"])) {
    $varCarrito_ProductosCompra = $_GET["recordID"];
}
$varUsuario_ProductosCompra = "0";
if (isset($row_Datoscompra['idUsuario'])) {
    $varUsuario_ProductosCompra = $row_Datoscompra['idUsuario'];
}
mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_ProductosCompra = sprintf("SELECT tblproducto.strNombre, tblproducto.dblPrecio, tblcarrito.intCantidad
FROM tblcarrito
Inner Join tblproducto ON tblcarrito.idProducto = tblproducto.idProducto
WHERE tblcarrito.intTransaccionEfectuada = %s AND tblcarrito.idUsuario = %s",
    GetSQLValueString($varCarrito_ProductosCompra, "int"),
    GetSQLValueString($varUsuario_ProductosCompra, "int"));

$ProductosCompra = mysql_query($query_ProductosCompra, $conexionzapatos) or die(mysql_error());
$row_ProductosCompra = mysql_fetch_assoc($ProductosCompra);
$totalRows_ProductosCompra = mysql_num_rows($ProductosCompra);
```

Y con esto ya obtenemos todos los datos .

Aquí cabe destacar hemos recurrido a las funciones que rescatan los datos para obtener registros de una manera simple y no tener que recurrir a estas consultas de mayor complejidad, o bien mediante consultas mas complejas.

La pagina **compras\_edit.php** quedaría:

The screenshot shows a web page titled "Consultar Compra". At the top, there is a header "Contenido" and a title "Consultar Compra". Below the title, there is a summary of purchase details: Nombre: {Datoscompra.idUsuario}, Fecha: {Datoscompra.fchCompra}, Forma de Pago: {Datoscompra.intTipoPago}, and Total: {Datoscompra.dblTotal}. A section labeled "Productos:" follows, containing a table with three columns: "Repetir Icto", "Precio", and "Cantidad". The table has one row with data: {ProductosCompra.strNombre}, {ProductosCompra.dblPrecio}, and {ProductosCompra.intCantidad}.

Mostramos todos los registros existentes con un bucle de control Do-While

```
<?php do { ?>
<tr>
    <td><?php echo $row_ProductosCompra['strNombre']; ?></td>
    <td><?php echo $row_ProductosCompra['dblPrecio']; ?></td>
    <td><?php echo $row_ProductosCompra['intCantidad']; ?></td>
</tr>
<?php } while ($row_ProductosCompra = mysql_fetch_assoc($ProductosCompra)); ?>
</table>
```

El proceso de compra por transferencia quedaría prácticamente finalizado, aunque se podría mejorar si al cliente se le remitiera un email con la compra efectuada y los requisitos para efectuar el pago. En el siguiente punto veremos como implementar esto.

## ENVÍO EMAIL COMPRADORES

Es conveniente avisar al comprador mediante un email de la compra efectuada y las instrucciones que debería seguir para completar su compra. Este email lo debe recibir tanto el cliente como el sistema para saber que se ha producido la compra.

Creamos una nueva función llamada **EnviarCorreoHTML()**

```
function EnvioCorreoHTML($destinatario, $contenido, $asunto)
{
    $mensaje = '<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Documento sin título</title>
</head>

<body>
<table width="100%" border="0" cellspacing="3" cellpadding="3">
<tr>
    <td></td>
</tr>
<tr>
    <td><p>Estimado Cliente:</p>
        <p>';
    $mensaje .= $contenido;
    $mensaje .= '</p></td>
</tr>
<tr>
    <td>Muchas gracias, puede contactarnos a través de nuestro correo electrónico:<br />
        <a href="mailto:info@tiendazapatos.com">info@tiendazapatos.com</a></td>
</tr>
</table>
</body>
</html>';
```

PD: mediante el .= podemos anexionar funciones y variable unas una al lado de otra

```
$mensaje .= $contenido;
```

```
// Para enviar correo HTML, la cabecera Content-type debe definirse
$cabeceras = 'MIME-Version: 1.0' . "\n";
$cabeceras .= 'Content-type: text/html; charset=iso-8859-1' . "\n";
// Cabeceras adicionales
$cabeceras .= 'From: info@tiendazapatos.com' . "\n";
$cabeceras .= 'Bcc: info@tiendazapatos.com' . "\n";

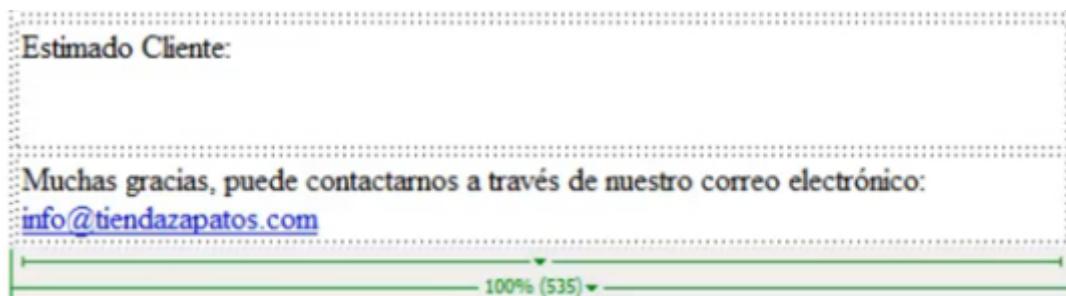
// Enviarlo
mail($destinatario, $asunto, $mensaje, $cabeceras);
echo $mensaje;
```

Este sería el esquema básico para poder enviar un email donde la función:

**EnvioCorreoHTML(\$destinatario, \$contenido, \$asunto)**, va a obtener estas tres variables, la función **mail** de PHP sera la encargada de enviar el email con las siguientes variables

```
mail($destinatario, $asunto, $mensaje, $cabeceras);
```

Creamos la plantilla para el email:



Una vez realizada la función vemos en que punto tenemos que llamar a la función.

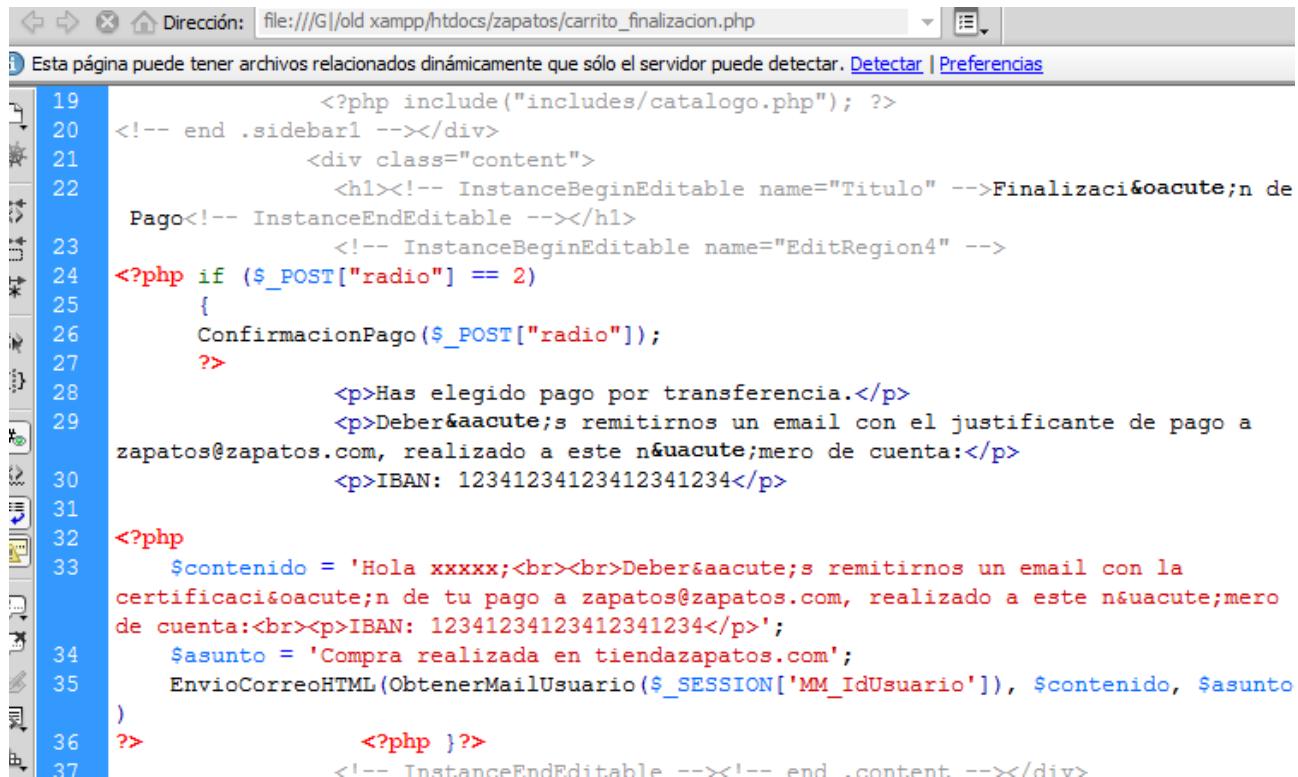
Cuando tengamos seleccionado el modo de pago y se pulse pagar en ese momento se deberá enviar el email esto ocurrirá en **carrito\_finalizacion.php**, añadimos la llamada a la función la cual le vamos a pasar 3 variables :

El Email del usuario activo en este momento (Creamos función ObtenerEmailUsuario() es similar a la que utilizamos para obtener nombre, podía parecer que crear una librería de funciones al principio podía parecer inútil, pero nos ahorra muchísimo trabajo conforme avanzamos y necesitamos obtener registros

```
function ObtenerMailUsuario($identificador)
{
    global $databaseConexionZapatos, $conexionZapatos;
    mysql_select_db($databaseConexionZapatos, $conexionZapatos);
    $query_ConsultaFuncion = sprintf("SELECT tblusuario.strEmail FROM tblusuario WHERE tblusuario.
        idUsuario = %s", $identificador);
    $ConsultaFuncion = mysql_query($query_ConsultaFuncion, $conexionZapatos) or die(mysql_error());
    $row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion);
    $totalRows_ConsultaFuncion = mysql_num_rows($ConsultaFuncion);

    return $row_ConsultaFuncion['strEmail'];
    mysql_free_result($ConsultaFuncion);
}
```

Vamos a realizar la llamada a la función.



The screenshot shows a browser window with the address bar set to "file:///G:/old xampp/htdocs/zapatos/carrito\_finalizacion.php". A status bar at the top says "Esta página puede tener archivos relacionados dinámicamente que sólo el servidor puede detectar. Detectar | Preferencias". The main content area displays the following PHP code:

```
19      <?php include("includes/catalogo.php"); ?>
20  <!-- end .sidebar1 --></div>
21      <div class="content">
22          <h1><!-- InstanceBeginEditable name="Titulo" -->Finalizaci&on de
23          Pago<!-- InstanceEndEditable --></h1>
24          <!-- InstanceBeginEditable name="EditRegion4" -->
25          <?php if ($_POST["radio"] == 2)
26          {
27              ConfirmacionPago($_POST["radio"]);
28          ?>
29              <p>Has elegido pago por transferencia.</p>
30              <p>Deber&aacute;s remitirnos un email con el justificante de pago a
31              zapatos@zapatos.com, realizado a este n&uacute;mero de cuenta:</p>
32              <p>IBAN: 12341234123412341234</p>
33
34          <?php
35              $contenido = 'Hola xxxx;<br><br>Deber&aacute;s remitirnos un email con la
36              certificaci&on de tu pago a zapatos@zapatos.com, realizado a este n&uacute;mero de cuenta:<br><p>IBAN: 12341234123412341234</p>';
37              $asunto = 'Compra realizada en tiendazapatos.com';
38              EnvioCorreoHTML(ObtenerMailUsuario($_SESSION['MM_IdUsuario']), $contenido, $asunto
39          )
40      ?>
41      <?php }?>
42      <!-- InstanceEndEditable --><!-- end .content --></div>
```

Con la función mail tenemos un problema en ya que estamos en Localhost ,y no encuentra un servidor de correo pero en un servidor real no daría ningún problema

Has elegido pago por transferencia.

Deberás remitirnos un email con la certificación de tu pago a [zapatos@zapatos.com](mailto:zapatos@zapatos.com), realizado a este número de cuenta:

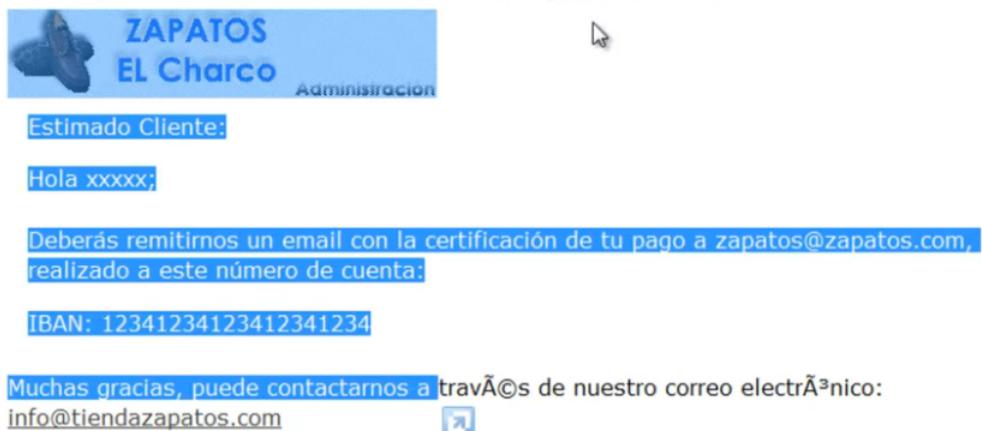
IBAN: 12341234123412341234

**Warning:** mail() [[function.mail](#)]: Failed to connect to mailserver at "localhost" port 25, verify your "SMTP" and "smtp\_port" setting in php.ini or use ini\_set() in **C:\wamp\www\zapatos\includes\funciones.php** on line **197**

Si hacemos un **echo \$mensaje;** , podemos comprobar que realmente esta funcionando aun a pesar del error comentado anteriormente

IBAN: 12341234123412341234

**Warning:** mail() [[function.mail](#)]: Failed to connect to mailserver at "localhost" port 25, verify your "SMTP" and "smtp\_port" setting in php.ini or use ini\_set() in **C:\wamp\www\zapatos\includes\funciones.php** on line **197**



Podemos comprobarlo también si vamos a la carpeta del Xampp **mailoutput** tendremos un listado de los email enviados por el sistema

Nombre	Fecha de modifica...	Tipo	Tamaño
mail-20190502-2243-329000	03/05/2019 0:43	Documento de tex...	1 KB
mail-20190502-2245-489000	03/05/2019 0:45	Documento de tex...	1 KB
mail-20190503-1204-35000	03/05/2019 14:04	Documento de tex...	1 KB
mail-20190504-1037-116000	04/05/2019 12:37	Documento de tex...	1 KB
mail-20190504-1040-157000	04/05/2019 12:40	Documento de tex...	1 KB
mail-20190504-1046-709000	04/05/2019 12:46	Documento de tex...	1 KB
mail-20190504-1046-732000	04/05/2019 12:46	Documento de tex...	1 KB
mail-20190504-1047-346000	04/05/2019 12:47	Documento de tex...	1 KB
mail-20190504-1048-447000	04/05/2019 12:48	Documento de tex...	1 KB
mail-20190505-0916-553000	05/05/2019 11:16	Documento de tex...	1 KB

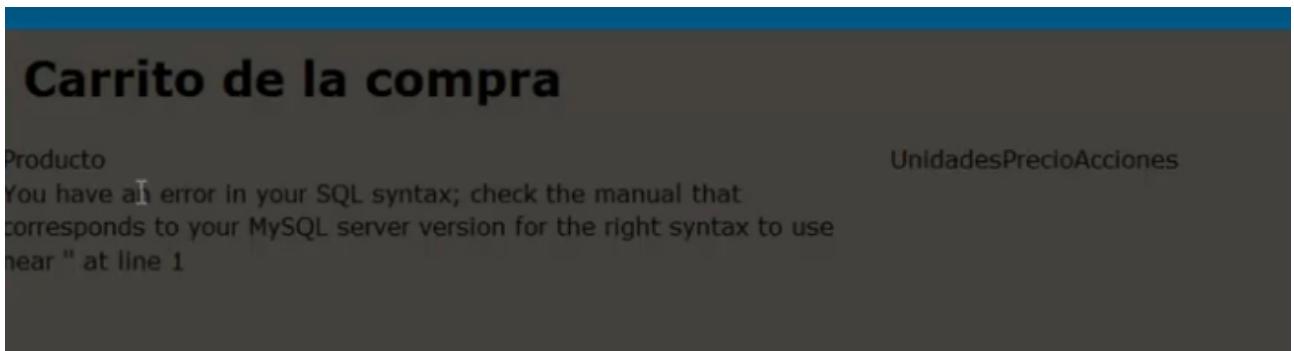
Podemos abrir uno de ellos para verlo mas detalladamente :

```
mail-20190502-2243-329000 - Bloc de notas
Archivo Edición Formato Ver Ayuda
To: fcomarcet1@gmail.com
Subject: Compra realizada en tiendazapatos.com
MIME-Version: 1.0
Content-Type: text/html; charset=iso-8859-1
From: info@tiendazapatos.com
BCC: info@tiendazapatos.com

<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Documento sin título</title>
</head>
<body>
<table width="100%" border="0" cellspacing="3" cellpadding="3">
<tr>
<td></td>
</tr>
<tr>
<td><p>Estimado Cliente:</p>
<p>Hola xxxx;<br><br>Deberás remitirnos un email con la certificación de tu pago a zapatos@zapatos.com</p>
</td>
<td><p>Muchas gracias, puede contactarnos a través de nuestro correo electrónico:<br /> <a href="mailto:info@tiendazapatos.com">info@tiendazapatos.com</a></p>
</td>
</tr>
</table>
</body>
</html>
```

Hemos de mencionar un problema que no me había dado cuenta y es el siguiente:

Al seleccionar el carrito de la compra si este esta vacío salta un error



Vamos a solucionar esto por ejemplo evitando que se visualice este error y nos muestre que el carrito esta vacío.

He pensado en que parte es la que no deseamos que nos muestre si el carrito esta vacío podemos implementar un Bucle if para solucionar esto

```
<?php if ($totalRows_DatosCarrito > 0) // mostrar registros si no esta vacio
{
    TABLA

<?php } // mostrar registros si no esta vacio ?>
<?php if ($totalRows_DatosCarrito == 0)
{
    // Mostrar si el registro esta Vacio ?>
    Tu carrito esta vacio.
<?php } // Mostrar si el registro esta Vaci? ?>
```

Añadiendo el resto de código quedaría :

```
<?php if ($totalRows_DatosCarrito > 0) { // Show if recordset not empty ?>
    <table width="100%" border="0" cellspacing="0" cellpadding="0">
        <tr>
            <td>Producto</td>
            <td>Unidades</td>
            <td>Precio</td>
            <td>Acciones</td>
        </tr>
        <?php $preciototal = 0;?>
        <?php do { ?>
        <tr>
            <td><?php echo ObtenerNombreProducto($row_DatosCarrito['idProducto']); ?></td>
            <td><?php echo $row_DatosCarrito['intCantidad']; ?></td>
            <td><?php echo ObtenerPrecioProducto($row_DatosCarrito['idProducto']); ?> Euros</td>
            <td>Eliminar</td>
        </tr>
        <?php $preciototal = $preciototal + ObtenerPrecioProducto($row_DatosCarrito['idProducto']);?>
        <?php } while ($row_DatosCarrito = mysql_fetch_assoc($DatosCarrito)); ?>
        <tr>
            <td>&nbsp;</td>
            <td align="right">Subtotal:</td>
            <td><?php echo $preciototal; ?> Euros</td>
            <td>&nbsp;</td>
        </tr>
        <tr>
            <td>&nbsp;</td>
            <td align="right">IVA:</td>
            <td><?php echo ObtenerIVA(); ?>%</td>
            <td>&nbsp;</td>
        </tr>
        <tr>
            <td>&nbsp;</td>
            <td align="right">IVA:</td>
            <td><?php echo ObtenerIVA(); ?>%</td>
            <td>&nbsp;</td>
        </tr>
        <tr>
            <td>&nbsp;</td>
            <td align="right">Valor del IVA:</td>
            <td><?php
                $multiplicador = ObtenerIVA()/100;
                $valordelIVA = $preciototal * $multiplicador;
                echo $valordelIVA;?> Euros</td>
                <td>&nbsp;</td>
            </tr>
            <tr>
                <td>&nbsp;</td>
                <td align="right">Total con IVA:</td>
                <td><?php
                    $multiplicador = (100 + ObtenerIVA())/100;
                    $valorconIVA = $preciototal * $multiplicador;
                    $_SESSION["TotalCompra"] = $valorconIVA;
                    echo $valorconIVA;?> Euros</td>
                    <td>&nbsp;</td>
                </tr>
            </table>
            <a href="carrito_forma_pago.php">Seleccionar Forma de Pago</a>
            <?php } // Show if recordset not empty ?>
            <?php if ($totalRows_DatosCarrito == 0) { // Show if recordset empty ?>
                Tu carrito esta vacio.
            <?php } // Show if recordset empty ?>
```

Continuemos con las formas de pago

## PAGO CON TARJETA

Para entender este proceso el cliente debe contratar un TPV (Terminal Punto de Venta ), con su entidad bancaria, el cliente deberá enviarnos una serie de datos para confirmar la transacción.

Vamos a escoger el pago por 4B, para ello seguimos el manual ofrecido en la pagina de Tarjetas 4B España.

En primer lugar cuando seleccionamos el pago por tarjeta nos redirecciona a la pagina carrito\_finalizacion.php y tomaba el valor 3, mediante un bucle de control If similar al que realizamos con la transferencia bancaria pero con valor 3

Con este tipo de servicio el pago no se realizara en ninguna de nuestras paginas sino en un servicio externo el cual nos facilita servidor seguro de TPV.

En principio con enviar un Identificador de la compra y un total de la compra a la empresa TPV y que allí se realice el pago. Si miramos un poco el manual al cual nos remite el banco para efectuar pagos por TPV nos indican que el formulario para introducir la tarjeta de crédito debe tener la siguiente forma:

```
<form id="form2" name="form2" method="post" action="https://tpv2.4b.es/simulador/teargral.exe">
    <input name="order" type="hidden" id="order" value="" />
    <input name="store" type="hidden" id="store" value="PV00002287633" />
    <input type="submit" name="button3" id="button3" value="Confirmar Pago por Tarjeta" />
</form>
```

El banco durante un periodo de prueba nos redirecciona a un simulador de pago el cual nos servirá para comprobar el correcto funcionamiento sin realizar gastos en las tarjetas indicadas, nosotros deberemos indicarles cuando el sistema funciona correctamente para que también el simulador por un servicio real donde ya si se efectúen los pagos certificados.

El servicio de TPV demanda en el formulario 2 variables que van a ser ocultas.(type hidden) 4B nos va a pedir 2 valores el id del pedido, y el id de la tienda (Código de tienda), nos lo facilitara la empresa de TPV, en mi caso me lo voy a inventar PV00002287633.

En el campo order vamos a enviarles el id de nuestra compra, Nosotros todavía no le hemos pasado este parámetro, mirando la función ConfirmacionPago().

De algún modo como cuando confirmábamos el pago necesitábamos que el sistema reconocza cual ha sido el ultimo Id y así asociarlo a esa ultima compra, usábamos la función:

```
$ultimacompra = mysql_insert_id();
```

→ Identifica ultimo id introducido.

Añadimos una nueva variable de sesión llamada `$_SESSION["compraactivavisa"]`, la cual sea igual a la variable `$ultimacompra`

Implementemos esto en la función `Confirmaciopago()`, volvemos a ver la eficiencia de crear la librería de funciones, con pequeñas modificación obtenemos los resultados deseados inmediatamente y no tenemos que cambiar el código de una pagina entera.

```
function ConfirmacionPago($tipopago)
{
    global $databaseConexionZapatos, $conexionZapatos;
    mysql_select_db($databaseConexionZapatos, $conexionZapatos);

    $insertSQL = sprintf("INSERT INTO tblcompra (idUsuario, fchCompra, intTipoPago, dblTotal) VALUES (%s, NOW(), %s, %s)",
        GetSQLValueString($_SESSION['MM_IdUsuario'], "int"),
        $tipopago,
        $_SESSION["TotalCompra"]);
    $Result1 = mysql_query($insertSQL, $conexionZapatos) or die(mysql_error());
    $ultimacompra = mysql_insert_id();
    $_SESSION["compraactivavisa"] = $ultimacompra;
    ActualizacionCarrito($ultimacompra);
}
```

Recuperaremos en **order** este valor mediante un echo

```
<?php if ($_POST["radio"] == 3)
//PAGO POR VISA/MASTERCARD/TARJETA
{
    ConfirmacionPago($_POST["radio"]);
?>
<p>Has elegido pago por Tarjeta de credito/debito.</p>
<p>Haz click aqui para saltar a un servidor seguro donde podr&aacute;s realizar el pago.</p>

<form id="form2" name="form2" method="post" action="https://tpv2.4b.es/simulador/teargral.exe">
    <input name="order" type="hidden" id="order" value="<?php echo $_SESSION["compraactivavisa"]; ?>" />
    <input name="store" type="hidden" id="store" value="PV00002287633" />
    <input type="submit" name="button3" id="button3" value="Confirmar Pago por Tarjeta" />
</form>

<?php ?>
```

El manual de TPV explica que al pulsar el botón que nos lleva a <https://tpv2.4b.es/simulador/teargral.exe>, va a reesolicitar a la pagina los datos del total de la compra ademas de otros .En principio no tengo acceso al simulador dado que no he solicitado ningún ID de mi tienda.

Ademas el sistema TPV nos va a requerir una serie de urls donde envíen información solicitada. Vamos a crear un archivo nuevo llamado **carrito\_tpv.php** donde devolveremos el total que nos solicitan ,esta pagina solo servirá para enviar datos no va a realizar otra función.

El servidor TPV nos solicita que le enviemos una cadena con un formato predeterminado de texto, veamos

```
echo "M978".$total."\r\n1\r\nZAPATOS\r\nProductos Zapatos\r\n1\r\n".$total;|
```

- El M978 indica que la moneda serán euros .
- El valor de la compra también tendrá un formato predeterminado, por ejemplo si vamos a realizar un pago de 129,55 se tendrá que enviar de la forma 12955

En PHP existe una función llamada **number\_format()** , que nos formatea un valor con el numero de decimales que le indicamos y sustituye las comas por espacios en blanco

```
$total = number_format($_SESSION["TotalCompra"], 2, ".", "");
```

Finalmente el código quedaría:

```
<?php require_once('Connections/conexionzapatos.php'); ?>
?>

$total = number_format($_SESSION["TotalCompra"], 2, ".", "");
$total = $total *100;

echo "M978". $total . "\r\n1\r\nZAPATOS\r\nProductos Zapatos\r\n1\r\n". $total;

?>
```

Esto seria lo que recibiría el servidor TPV



No puedo comprobar mas el funcionamiento dado que no tengo ningn ID de tienda y tendría que solicitarlo mediante un pago a 4B.

Con esto estaría finalizado el pago con tarjeta.

PD: Después de investigar un poco mas si podía hacer funcionar el simulador de pagos me di cuentas que el servicio de TPV de 4B realizo un cambio de empresa en 2018 a Redsys, así que los servidores de 4B que he usado ya no funcionan, así que pido te disculpas por no haberme dado cuenta antes de este significativo cambio y haber implementado un sistema que no funcione correctamente a día de hoy si tuviera un poco mas de tiempo intentare solucionarlo.

## PAGO POR PAYPAL

### PAGO CON PAYPAL

En el pago por PayPal en el formulario de carrito\_forma\_pago.php tenia el valor 1 y nos redirecciona a carrito\_finalizacion.php

```
<input name="radio" type="radio" id="radio" value="1" checked="checked" />
<label for="radio">PayPal</label><br />
```

de nuevo creamos un If similar a las demás formas de pago que ya realizamos ,al igual que el pago con 4B , Paypal nos va a suministrar un un manual de como efectuar pagos mediante su servicio de pago seguro, nos dan un modelo de formulario básico a insertar para efectuar el pago.

```
<html>
  <body>
    <form action="https://www.sandbox.paypal.com/cgi-bin/webscr" method="post">
      <input type="hidden" name="cmd" value="_xclick">
      <input type="hidden" name="business" value="openalfa-facilitator@openalfa.com">
      <input type="hidden" name="item_name" value="Premium Subscription">
      <input type="hidden" name="currency_code" value="USD">
      <input type="hidden" name="amount" value="20.00">
      <input type="image" src="http://www.paypal.com/es_XC/i/btn/x-click-but01.gif"
            name="submit"
            alt="Make payments with PayPal - it's fast, free and secure!">
    </form>
  </body>
</html>
```

En el cual solo tendremos que adaptar nuestros datos al formulario en cuestión

```
<FORM action="https://www.paypal.com/cgi-bin/webscr" method="post" id="paypal_form">
  <input type="hidden" name="upload" value="1" />
  <input type="hidden" name="amount" value=<?php echo $_SESSION["TotalCompra"]; ?>" />
  <input type="hidden" name="business" value="cuentabusiness@zapatos.com" />
  <input type="hidden" name="receiver_email" value="cuentabusiness@zapatos.com" />
  <input type="hidden" name="cmd" value="_xclick" />
  <input type="hidden" name="charset" value="utf-8" />
  <input type="hidden" name="currency_code" value="EUR" />
  <input type="hidden" name="item_name" value="Compra en la Web de TiendaZapatos.com " />
  <input type="hidden" name="payer_id" value=<?php echo $_SESSION['MM_IdUsuario']; ?>" />
  <input type="hidden" name="payer_email" value=<?php echo ObtenerMailUsuario($_SESSION['MM_IdUsuario']); ?>" />
  <input type="hidden" name="return"
        value="http://www.tiendazapatos.com/carrito_ok.php?control=<?php echo $_SESSION['compraactivavisa']; ?>" />
  <input type="hidden" name="cancel_return"
        value="http://www.tiendazapatos.com/carrito_ko.php?control=<?php echo $_SESSION['compraactivavisa']; ?>" />
  <input type="hidden" name="rm" value="2" />
  <input type="hidden" name="bn" value="PRESTASHOP_WPS" />
  <input type="hidden" name="cbt" value="Volver a www.tiendazapatos.com" />

  <input type="image" src="https://www.paypal.com/es_ES/ES/i/btn/btn_xpressCheckout.gif" name="image">
</FORM>
```

Y simplemente con esto ya podríamos comprobar si funciona vamos a comprar un producto de 20 euros

## Carrito de la compra

Producto	Unidades	Precio	Acciones
Bota de Agua	1	20 Euros	Eliminar
		Subtotal: 20 Euros	
		IVA: 21%	
		Valor del IVA: 4.2 Euros	
		Total con IVA: 24.2 Euros	

[Seleccionar Forma de Pago](#)

Seleccionamos forma de pago PayPal y nos lleva a

### Finalización de Pago

Has elegido pago por Paypal

Haz click aqui para saltar al servidor seguro de Paypal



Una vez seleccionamos la imagen nos lleva a la pagina de PayPal para identificarnos y efectuar el pago

The screenshot shows the PayPal payment process page. At the top, it says "Proceso de pago de PayPal" and "24,20 EUR". There are links for "Iniciar sesión" and "¿Ya tiene una cuenta PayPal?". Below this, there's a section for "Pagar con tarjeta de débito o crédito" with a note: "No compartimos su información financiera con el vendedor." It includes dropdown menus for "País" (España), "Tipo de tarjeta", and "Número de tarjeta". There are also fields for "Vencimiento", "CSC", "Nombre", "Apellidos", "Tipo de teléfono Móvil", and "Número de teléfono +34". On the right side, there's a shield icon and the text "La forma rápida y segura de pagar." with a note: "Da igual dónde compre porque no compartimos su información de pago con el vendedor."

### 6.3.12 HISTORIAL DE COMPRAS DE UN CLIENTE. MIS COMPRAS

#### usuario\_compras.php

Todo cliente quiere tener acceso a su historial de compras y poder revisar el estado de su compra, y comprobar que su pago ha sido aceptado o en su defecto cancelada su compra

Añadimos en nuestro **catalogo.php** un nuevo enlace :

#### Mis Compras

```
if (MM_Username != "") {
    echo "Hola ";
    echo ObtenerNombreUsuario($_SESSION[
'MM_IdUsuario']);
    ?>
    <br />
    <a href="carrito_lista.php" class=
"modificacionusuario">Carrito de la Compra</a><br />
    <a href="usuario_compras.php" class=
"modificacionusuario">Mis Compras</a><br />
    <br />
    <a href="usuario_modificar.php" class=
"modificacionusuario">Modificar</a> - <a href=
"usuario_cerrarsesion.php" class=
"modificacionusuario">Salir</a>
<?php
}
~1~
```



Al pulsar en **Mis Compras**, nos lleva a **usuario\_compras.php**, en la cual obtendremos la lista de compras de la persona identificada en el sistema.

Primero obtenemos una lista de las compras realizadas mediante una consulta SELECT a la tabla **tblcompra**, como siempre le pasamos por parámetro la variable de sesión **\$\_SESSION["MM\_IdUsuario"]** el cual nos realizará la consulta en función del usuario que está en la variable de sesión

```
$varUsuario_DatosCompra = "0";
if (isset($_SESSION["MM_IdUsuario"])) {
    $varUsuario_DatosCompra = $_SESSION["MM_IdUsuario"];
}
mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_DatosCompra = sprintf("SELECT * FROM tblcompra WHERE tblcompra.idUsuario = %s", GetSQLValueString(
    $varUsuario_DatosCompra, "int"));
$DatosCompra = mysql_query($query_DatosCompra, $conexionzapatos) or die(mysql_error());
$row_DatosCompra = mysql_fetch_assoc($DatosCompra);
$totalRows_DatosCompra = mysql_num_rows($DatosCompra);
```

Implementamos los echo para que se muestre por pantalla:

```
<table width="100%" border="0" cellspacing="0" cellpadding="0">
<tr>
<td bgcolor="#E0E0E0">Compra</td>
<td bgcolor="#E0E0E0">Total</td>
<td bgcolor="#E0E0E0">Estado</td>
</tr>
<?php do { ?>
<tr>
<td><?php echo $row_DatosCompra['fchCompra']; ?><br /><?php Mostrar_Carrito_Usuario($row_DatosCompra['idCompra']); ?></td>
<td><?php echo $row_DatosCompra['dblTotal']; ?></td>
<td><?php echo TextoEstadoCompra($row_DatosCompra['intEstado']); ?></td>
</tr>
<?php } while ($row_DatosCompra = mysql_fetch_assoc($DatosCompra)); ?>
<tr>
<td>&ampnbsp</td>
<td>&ampnbsp</td>
<td>&ampnbsp</td>
</tr>
</table>
```

Sería interesante que al seleccionar una compra muestre la compra mas detallada. Para ello implementamos una nueva función **MostrarCarritoUsuario()**

```
function Mostrar_Carrito_Usuario($identificador)
{
    global $databaseConexionZapatos, $conexionZapatos;
    mysql_select_db($databaseConexionZapatos, $conexionZapatos);
    $query_ConsultaFuncion = sprintf("SELECT * FROM tblcarrito WHERE intTransaccionEfectuada = %s", $identificador);
    $ConsultaFuncion = mysql_query($query_ConsultaFuncion, $conexionZapatos) or die(mysql_error());
    $row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion);
    $totalRows_ConsultaFuncion = mysql_num_rows($ConsultaFuncion);
?>
    <div class="subproductos">
        <?php
            do {
                echo ObtenerNombreProducto($row_ConsultaFuncion['idProducto']);
                echo "<br>";
            } while ($row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion));
?>
    </div>
    <?php
```

Para mostrar mas clara la información hemos creado un **<div class="subproducto">** que mejora la visualización

Compra	Total
2019-05-09 22:54:30	
Bota de Agua	614.68
Tacon Alto	
2019-05-16 09:38:08	
Bota de Agua	72.6

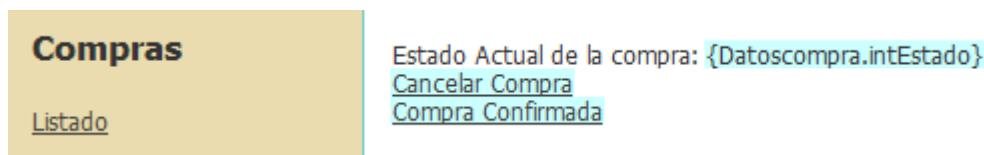
Ahora debemos volver a la parte de la administración de la tienda ya que queremos poder cambiar el estado de la compra y que el cliente en Mis compras vea si su pedido se ha aceptado y se ha enviado o por el contrario se ha cancelado por algún problema en el pago

## ESTADO DE LA COMPRA en Mis Compras

### compras\_edit.php

Vamos a crea unos enlaces:

- Cancelar Compra → compra\_aceptar.php
- Compra Confirmada → compra\_cancelar.php



Estos enlaces nos llevaran las paginas mencionadas y estas realizaran las operaciones necesarias para mostrar dicha información.

Si observamos la base de datos, en la tabla **tblcompra** podemos observar que nos falta un campo que nos identifique el estado de la compra → **intEstado** en principio vamos a ponerlo a 0 por defecto .

Creamos una nueva función , que nos devuelva el nombre del tipo de pago seleccionado

```
function TextoEstadoCompra($varestado)
{
    if ($varestado == 0) return "Pendiente";
    if ($varestado == 1) return "Pagado y enviado";
    if ($varestado == 2) return "Compra cancelada";
}
```

con la cual vamos a tener los 3 estados básicos

```
<p>Estado Actual de la compra: <?php echo TextoEstadoCompra($row_Datoscompra['intEstado']); ?><br />
<a href="compra_cancelar.php?recordID=<?php echo $row_Datoscompra['idCompra']; ?>&usuario=<?php echo $row_Datoscompra['idUsuario']; ?>">Cancelar Compra</a><br />
<a href="compra_aceptar.php?recordID=<?php echo $row_Datoscompra['idCompra']; ?>&usuario=<?php echo $row_Datoscompra['idUsuario']; ?>">Compra Confirmada</a></p>
```

Vamos a añadir mas atributos de la compra para tener mas conocimiento de ella antes de proceder a la confirmación o cancelación

The screenshot shows a web browser window with the URL 'xampp/htdocs/zapatos/admin/compras\_edit.php'. The page title is 'ELITE FEET'. Below it, a section titled 'Contenido' contains the heading 'Consultar Compra'. Underneath, there is a table of purchase details:

Nombre:	{Datoscompra.idUsuario}
Fecha:	{Datoscompra.fchCompra}
Forma de Pago:	{Datoscompra.intTipoPago}
Total:	{Datoscompra.dblTotal}

Below the table, a message says 'Estado Actual de la compra: {Datoscompra.intEstado}'. There are two buttons: 'Cancelar Compra' and 'Compra Confirmada'.

Tanto en **compra\_cancelar.php** como en **compra\_aceptar.php** realizaremos una operación de UPDATE de nuestra BD, que la realizaremos mediante una nueva función llamada **ActualizacionEstadoCarrito()** la cual le pasaremos la variable de la compra y del estado

```
function ActualizacionEstadoCarrito($varcompra, $varestado)
{
    global $databaseConexionzapatos, $conexionzapatos;
    $updateSQL = sprintf("UPDATE tblcompra SET intEstado=%s WHERE idCompra = %s",
    $varestado,$varcompra);
    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $Result1 = mysql_query($updateSQL, $conexionzapatos) or die(mysql_error());
}
```

Ahora en ambas páginas ,**compra\_cancelar.php** y **compra\_aceptar.php** realizamos la llamada a la función

The screenshot shows the source code of the file 'compra\_cancelar.php'. At the top, there is a header bar with tabs like 'Dividir', 'Diseño', 'Código en vivo', 'Vista en vivo', 'Inspeccionar', etc. The address bar shows the file path: 'file:///G|/old xampp/htdocs/zapatos/admin/compra\_cancelar.php'. The main content area contains the PHP code:

```
<?php require_once('../Connections/conexionzapatos.php'); ?>
<?php ActualizacionEstadoCarrito($_GET["recordID"], 2);?>
```

Donde le vamos a pasar las variables que demanda la compra en la que estamos ahora que la estamos pasando por parámetro y es `$_GET["recordID"]`, y el estado que sera el valor=2, en el caso de cancelar la compra .

```
if ($varestado == 2) return "Compra cancelada";
```

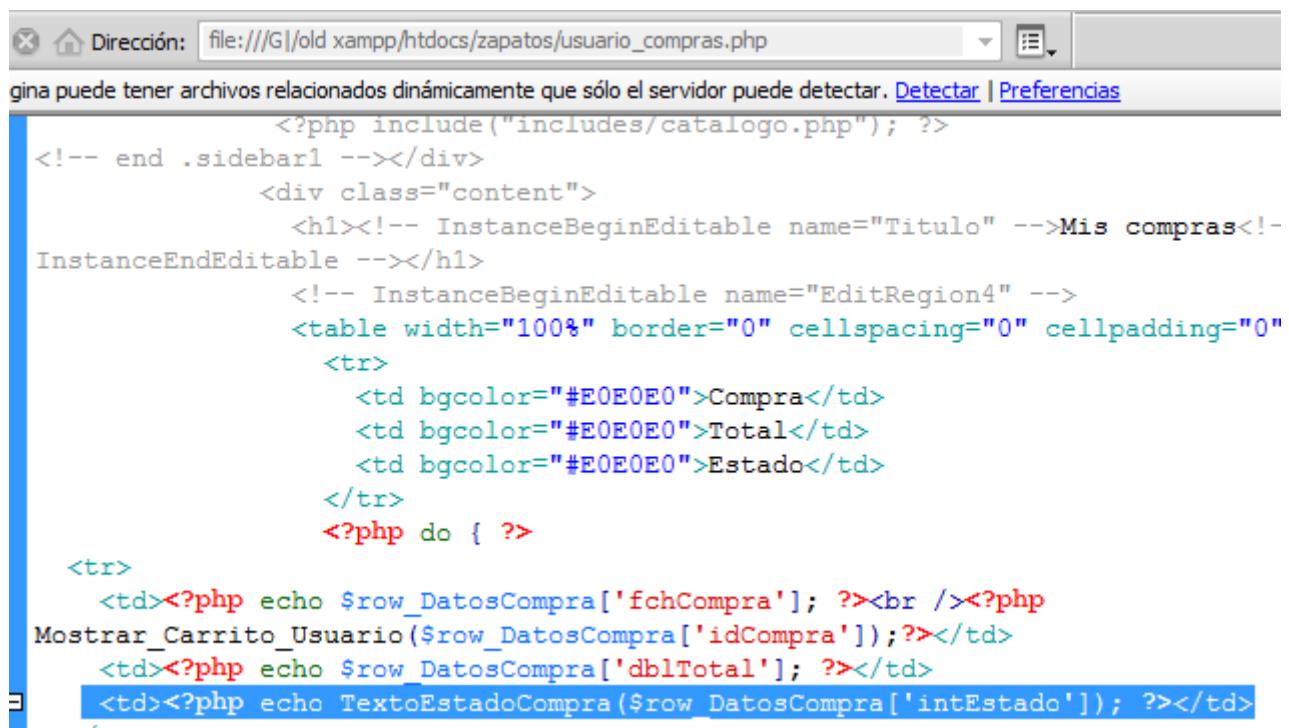
Veamos si funciona vamos a cancelar la compra del 18/5/2019 a las 13:21

Hemos cancelado la compra y ahora al usuario en su historial de compras le debería salir como cancelado el pedido

Ahora nos falta desde la parte del usuario pueda ver en que estado esta su compra, para ello en **compras\_edit.php** en la parte de Estado, con un echo podemos visualizar esto :

Total	Estado
23.6	0
87.32	2
23.6	0

Surge un problema es que solo visualizamos el valor no el nombre, y esto es porque todavía no hemos realizado la llamada a la función `TextoEstadoCompra`



The screenshot shows a browser window with the address bar containing 'file:///G:/old xampp/htdocs/zapatos/usuario\_compras.php'. The page content displays the source code of the PHP file. The code includes HTML structure for a table and several PHP blocks. One specific PHP block is highlighted in blue:

```
<?php echo TextoEstadoCompra($row_DatosCompra['intEstado']); ?></td>
```

## Mis compras

Compra	Total	Estado
2019-05-09 22:54:30 Bota de Agua Tacon Alto	614.68	Compra cancelada
2019-05-16 09:38:08 Bota de Agua	72.6	Pendiente
2019-05-18 13:21:56 Bota de Agua	24.2	Pendiente

## Mis compras

Compra	Total	Estado
2019-05-09 22:54:30 Bota de Agua Tacon Alto	614.68	Compra cancelada
2019-05-16 09:38:08 Bota de Agua	72.6	Pendiente
2019-05-18 13:21:56 Bota de Agua	24.2	Compra cancelada

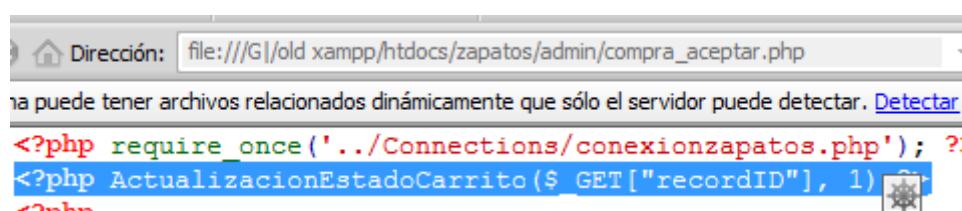
Efectivamente vemos como la compra con fecha 18/5/2019 a las 13:21 le sale al cliente cancelada

Por ultimo en este punto vamos a añadir un botón volver en **compra\_cancelar.php** y **compra\_aceptar.php** que nos devuelva a la pagina anterior pero que de alguna manera identifique cual es el ID(recordID) , que era el parámetro que le pasamos en la UPDATE como `$_GET[""]` que era **idCompra** de la compra y nos devuelva a la pagina adecuada

```
<h1>Compra Cancelada</h1>
| | <p><a href="compras_edit.php?recordID=<?php echo $_GET["recordID"]; ?>">Volver</a></p>
```

Para compra\_aceptar.php el proceso es similar solo cambia el estado a valor 1

```
if ($varestado == 1) return "Pagado y enviado";
```



Ya que tenemos implementada la función de enviar emails podríamos hacer que cada vez que cambia el estado del pedido al cliente se le enviara un email con la modificación de este.

En **compra\_aceptar.php** añadimos el código para ello :

```

<?php require_once('../Connections/conexionzapatos.php'); ?>
<?php ActualizacionEstadoCarrito($_GET["recordID"], 1);?>
<?php
    $contenido = 'Hola xxxx;<br><br>Tu compra ha sido aceptada y en estos
    momentos la estamos enviando a la dirección que nos indicaste<br>';
    $asunto = 'Compra enviada en tiendazapatos.com';
    EnvioCorreoHTML(ObtenerMailUsuario($_GET['usuario']), $contenido, $asunto)
?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"><!-- InstanceBegin
template="/Templates/BaseAdmin.dwt.php" codeOutsideHTMLIsLocked="false" -->
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<!-- InstanceBeginEditable name="doctitle" -->
<title>Administración Principal Tienda Zapatos</title>

```

Solo hemos tenido que modificar la variable de sesión que teníamos en ObtenerMailUsuario, que era **`$_SESSION["MM_IdUsuario"]`** que se utilizaba cuando la persona estaba comprando, en este caso no es la variable de sesión sino es la variable del usuario que estamos aceptando o cancelando la compra.

Esto es mas sencillo de entender si miramos un poco como en **compras\_edit.php** obteníamos el nombre del cliente , este parámetro también se lo podemos enviar a **compra\_aceptar.php** y a **compra\_cancelar.php** .

Veamos como hacemos esto en **compra\_cancelar.php** teníamos la variable **recordID**, para añadir el usuario utilizamos :

**&usuario=<?php...?>**

```

<a href="compra_aceptar.php?recordID=<?php echo $row_Datoscompra['idCompra']; ?>
&usuario=<?php echo $row_Datoscompra['idUsuario']; ?>">Compra Confirmada</a></p>

```

Ahora ya si podemos pasar la variable usuario y enviarla al sistema

```

EnvioCorreoHTML(ObtenerMailUsuario($_GET['usuario']), $contenido, $asunto)

```

de manera análoga para cuando se cancele un pedido en **compra\_cancelar.php** realizaremos los mismos pasos.

The screenshot shows the Dreamweaver interface with the code editor on the left and the visual editor on the right. The code editor displays PHP code for canceling a purchase. The visual editor shows a page structure with a header, a sidebar labeled 'Administración' containing links for 'Lista Productos', 'Lista Categorías', and 'Lista Usuarios', and a main content area labeled 'Compras' with a link 'Listado'. The footer of the page is labeled 'Administracion Tienda Zapatos'.

```

1 <?php require_once('../Connections/conexionzapatos.php'); ?>
2 <?php ActualizacionEstadoCarrito($_GET['recordID'], 2);?>
3 <?php
4     $contenido = 'Hola xxxx;<br><br>Tu compra ha sido cancelada. Contacta
5 con nosotros para cualquier problema<br>';
6     $asunto = 'Compra cancelada en tiendazapatos.com';
7     EnvioCorreoHTML(ObtenerMailUsuario($_GET['usuario']), $contenido,
8     $asunto)
9 ?>
10
11 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
12 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
13 <html xmlns="http://www.w3.org/1999/xhtml"><!-- InstanceBegin
14 template="/Templates/BaseAdmin.dwt.php" codeOutsideHTMLIsLocked="false" -->
15 <head>
16 <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<!-- InstanceBeginEditable name="doctitle" -->
<title>Administracion Principal Tienda Zapatos</title>
<!-- InstanceEndEditable -->
<!-- InstanceBeginEditable name="head" -->

```

Como mencionamos anteriormente el mail no funciona correctamente en Localhost pero la implementación es correcta.

### 6.3.13 ELIMINAR PRODUCTOS CARRITO

#### carrito\_eliminar.php

El producto que deseamos borrar del carrito esta identificado con un ID en este caso **idProducto**, vamos a vincular eliminar con la pagina **carrito\_eliminar.php** que le vamos a enviar al parámetro **recordID** el la variable **intContador** si vamos a la base de datos y vemos la tabla carrito esto se entenderá mejor

intContador	idUsuario	idProducto	intCantidad	intTransaccionEfectuada
1	2	1	1	0
2	2	9	1	0
3	2	7	1	0
4	2	1	1	0
5	2	7	1	0
6	4	1	1	1

Como vemos el **intContador** nos identifica la fila en la cual esta el registro que es lo que necesitamos aunque a primera vista lo lógico seria eliminar el **idProducto**, pero no seria correcto. Es mucho mas sencillo eliminar la fila por ese identificador

```
<td><a href="carrito_eliminar.php?recordID=<?php echo $row_DatosCarrito['intContador']; ?>">Eliminar</a></td>
```

En la pagina **carrito\_eliminar.php** realizaremos una operación DELETE de la base de datos

```
if (isset($_GET['recordID'])) {
    $deleteSQL = sprintf("DELETE FROM tblcarrito WHERE intContador=%s",
                         GetSQLValueString($_GET['intContador'], "int"));

    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $Result1 = mysql_query($deleteSQL, $conexionzapatos) or die(mysql_error());
}
```

En la cual le pasamos por parámetro **intContador**

Veamos si esto funciona, tenemos el carrito con 3 productos

Producto	Unidades	Precio	Acciones
Bota de Agua	1	20 Euros	<a href="#">Eliminar</a>
Bota de Agua	1	20 Euros	<a href="#">Eliminar</a>
Bota de Agua	1	20 Euros	<a href="#">Eliminar</a>
Subtotal:60 Euros			
IVA:21%			
Valor del IVA:12.6 Euros			
Total con IVA:72.6 Euros			
<a href="#">Seleccionar Forma de Pago</a>			

Al seleccionar eliminar :

Producto	Unidades	Precio	Acciones
Bota de Agua	1	20 Euros	<a href="#">Eliminar</a>
Bota de Agua	1	20 Euros	<a href="#">Eliminar</a>
Subtotal:40 Euros			
IVA:21%			
Valor del IVA:8.4 Euros			
Total con IVA:48.4 Euros			
<a href="#">Seleccionar Forma de Pago</a>			

## VACIAR CARRITO

### carrito\_eliminar\_todo.php

Añadimos un enlace para vaciar carrito y lo redireccionamos a carrito\_eliminar\_todo.php en la cual eliminaremos todos los productos que pertenezcan a ese cliente y tengan el registro de intTransaccionEfectuada sea 0, no sera necesario pasar ningun parámetro.

Con un simple DELETE

```
$deleteSQL = sprintf("DELETE FROM tblcarrito WHERE idUsuario=%s AND intTransaccionEfectuada = 0",
    GetSQLValueString($_SESSION['MM_IdUsuario'], "int"));

mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$result1 = mysql_query($deleteSQL, $conexionzapatos) or die(mysql_error());
```

## Lista de Productos

[Añadir Producto](#)

NOMBRE PRODUCTO	ESTADO	STOCK	ACCIONES
Bota de Agua	1	0	<a href="#">Editar</a> - <a href="#">Eliminar</a>
Tacon Economico	1	0	<a href="#">Editar</a> - <a href="#">Eliminar</a>
Zapato 2	1	0	<a href="#">Editar</a> - <a href="#">Eliminar</a>
Zapato 2	1	0	<a href="#">Editar</a> - <a href="#">Eliminar</a>
Zapato 2	1	0	<a href="#">Editar</a> - <a href="#">Eliminar</a>
Zapato 2	1	0	<a href="#">Editar</a> - <a href="#">Eliminar</a>
Zapato 2	1	0	<a href="#">Editar</a> - <a href="#">Eliminar</a>
Zapato 2	1	0	<a href="#">Editar</a> - <a href="#">Eliminar</a>
Zapato 2	1	0	<a href="#">Editar</a> - <a href="#">Eliminar</a>
Zapato 2	1	20	<a href="#">Editar</a> - <a href="#">Eliminar</a>

# 7. MEJORAS EN LA TIENDA

Llegado a este punto tenemos una tienda web funcional, pero considero que se podrían realizar unas mejoras, ya que de momento la pagina es funcional pero muy poco atractiva

## 7.1 SUBCATEGORIAS ADMINISTRACIÓN

Estaría bien aparte de las categorías principales, que los productos tuvieran subcategorías

Vamos a añadir un campo mas a la tabla **tblcategorias**, el cual nos gestione las subcategorías → **idPadre** (lo he nombrado así ya que en muchos manuales de programación relacionan una categoría con una subcategoria mediante padre-hijo), toda estructura de categorías seria como un árbol, y ha de saber que subcategorías pertenecen a cada categoría.

Vamos por ejemplo a identificar la categoría con un valor 0, así cada nueva subcategoria tendrá otro valor y estará representada a su vez por el **idCategoria**.

Cada vez que se modifique una categoría nos dejara indicar a que subcategoria pertenece.

Las categorías padre serán aquellas que su idPadre=0, vamos a realizar una consulta llamada DatosCategoriasPrincipales

```
mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_DatosCategoriasPrincipales = "SELECT * FROM tblcategoria WHERE tblcategoria.idPadre = 0";
$DatosCategoriasPrincipales = mysql_query($query_DatosCategoriasPrincipales, $conexionzapatos)
or die(mysql_error());
$row_DatosCategoriasPrincipales = mysql_fetch_assoc($DatosCategoriasPrincipales);
$totalRows_DatosCategoriasPrincipales = mysql_num_rows($DatosCategoriasPrincipales);
```

Ahora necesitamos en el formulario desplegable introducir los datos de la consulta SELECT

```
<td><label for="idPadre"></label>
<select name="idPadre" id="idPadre">
<option value="0">Categoria Principal</option>
<?php
do
{
?>
    <option value=<?php echo $row_DatosCategoriasPrincipales['idCategoria']?>>
        <?php echo $row_DatosCategoriasPrincipales['strDescripcion']?></option>
    <?php
}
?>
    while ($row_DatosCategoriasPrincipales = mysql_fetch_assoc($DatosCategoriasPrincipales));
    $rows = mysql_num_rows($DatosCategoriasPrincipales);

    if($rows > 0)
    {
        mysql_data_seek($DatosCategoriasPrincipales, 0);
        $row_DatosCategoriasPrincipales = mysql_fetch_assoc($DatosCategoriasPrincipales);
    }
?>
```

Y también modificar la la INSERT para que obtenga el valor asociado a **idPadre**, con lo cual cuando realicemos este INSERT nos crea dentro de la tabla **tblcategoria** un registro con los valores de **idPadre** y **strDescripcion**

```

if ((isset($_POST["MM_insert"])) && ($_POST["MM_insert"] == "form1")) {
    $insertSQL = sprintf("INSERT INTO tblcategoria (idPadre, strDescripcion) VALUES (%s, %s)",
        GetSQLValueString($_POST['idPadre'], "text"),
        GetSQLValueString($_POST['strDescripcion'], "text"));

    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $Result1 = mysql_query($insertSQL, $conexionzapatos) or die(mysql_error());

    $insertGoTo = "categorias_lista.php";
    if (isset($_SERVER['QUERY_STRING'])) {
        $insertGoTo .= (strpos($insertGoTo, '?')) ? "&" : "?";
        $insertGoTo .= $_SERVER['QUERY_STRING'];
    }
    header(sprintf("Location: %s", $insertGoTo));
}

```

Con esto ya podemos añadir subcategorías posteriormente ya intentaremos eliminarlas ya que es algo mas complejo ya que tienen dependencia sobre esto los productos y las subcategorías.

Si miramos en la pagina ya las tenemos pero tendremos que modificarlas para poder visualizarlas mejor



Modificando la consulta SELECT de **catalogo.php** que es donde visualizamos las categorías, añadimos una condición **WHERE idPadre=0**, también queremos que se visualicen tipo árbol .

Vamos a crear una nueva función denominada → **mostrar\_subcategorias()**, a la cual le enviemos como variable la categoría, la cual queremos que busque la subcategoría

```
<?php mostrar_subcategorias($row_Recordset1['idCategoria']);?>
```

Para la función **mostrar\_subcategorias()** podemos apoyarnos en el código que nos mostraba el carrito del usuario → **mostrar\_carrito\_usuario.php**

```

function mostrar_subcategorias($identificador)
{
    global $database_conexionzapatos, $conexionzapatos;
    mysql_select_db($database_conexionzapatos, $conexionzapatos);
    $query_ConsultaFuncion = sprintf("SELECT * FROM tblcategoria WHERE idPadre = %s ORDER BY
        tblcategoria.strDescripcion", $identificador);
    $ConsultaFuncion = mysql_query($query_ConsultaFuncion, $conexionzapatos) or die(mysql_error());
    $row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion);
    $totalRows_ConsultaFuncion = mysql_num_rows($ConsultaFuncion);
    ?>
    <?php
    do {
        echo " &nbsp;&nbsp;".$row_ConsultaFuncion['strDescripcion'];
        echo "<br>";

    } while ($row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion));
    ?>

    <?php

    mysql_free_result($ConsultaFuncion);
}

```

Esta función nos llega un identificador, queremos que seleccione de la tabla categorías todas las subcategorías que dependan de la que nosotros enviamos.

También es interesante poder ver las subcategorías, creamos un enlace en Ver Categorías que nos redirecciona a la misma pagina en cuestión **categorías\_lista.php**, deberemos enviarle un parámetro para que seleccione la subcategoria de la categoría seleccionada

```

<tr>
    <td><?php echo $row_Recordset1['strDescripcion']; ?></td>
    <td><a href="categorias_edit.php?recordID=<?php echo $row_Recordset1['idCategoria']; ?>">Editar
        </a> -
        <a href="categorias_lista.php?recordID=<?php echo $row_Recordset1['idCategoria']; ?>">Ver
        Subcategorias</a></td>
</tr>

```

Con esto no funciona todavía ya que todavía no le hemos indicado a el parámetro correctamente que en nuestro caso sera igual a cero que corresponde con el valor de la categoría padre , para ello en el menú de la admin que teníamos en **cabeceraadmin.php** vamos a pasar el parámetro al enlace recordID=0

```

<h2>Administracion</h2>
<p> <a href="../admin/productos_lista.php">Lista Productos</a></p>
<p> <a href="../admin/categorias_lista.php?recordID=0">Lista
    Categorias</a></p>
<p> <a href="../admin/usuarios_lista.php">Lista Usuarios</a></p>
<h2>Compras</h2>
<p> <a href="../admin/compras_lista.php">Listado</a></p>

```

## Administracion

[Lista Productos](#)

[Lista Categorias](#)

[Lista Usuarios](#)

## Compras

[Listado](#)

y ahora en la SELECT que nos obtenía el campo **idPadre** le pasamos el valor obtenido por parámetro

`$_GET["recordID"]`

```
mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_Recordset1 = "SELECT * FROM tblcategoria WHERE idPadre=".$_GET["recordID"]." ORDER BY tblcategoria.strDescripcion ASC";
$Recordset1 = mysql_query($query_Recordset1, $conexionzapatos) or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($Recordset1);
$totalRows_Recordset1 = mysql_num_rows($Recordset1);
```

Añadimos también en la consulta de la función **mostrar\_subcategorias\_admin()**, un **WHERE idPadre=%s** , de igual forma modificamos el código en **productos\_add.php** para que nos pueda mostrar las subcategorias mediante una llamada a la función **mostrar\_subcategoria\_admin()**

```
function mostrar_subcategorias_admin($identificador)
{
    global $databaseConexionzapatos, $conexionzapatos;
    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $query_ConsultaFuncion = sprintf("SELECT * FROM tblcategoria WHERE idPadre = %s ORDER BY
        tblcategoria.strDescripcion", $identificador);
    $ConsultaFuncion = mysql_query($query_ConsultaFuncion, $conexionzapatos) or die(mysql_error());
    $row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion);
    $totalRows_ConsultaFuncion = mysql_num_rows($ConsultaFuncion);
    ?>
    <?php
    if ($totalRows_ConsultaFuncion > 0) {
        do {
            ?>
<option value=<?php echo $row_ConsultaFuncion['idCategoria']?>>&nbsp;&nbsp;&nbsp;<?php echo $row_ConsultaFuncion['strDescripcion']?>>/option>
        <?php
            } while ($row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion));
    }
    mysql_free_result($ConsultaFuncion);
}
```

Como punto final a las subcategorias vamos a intentar que cuando editemos un producto, nos seleccione automáticamente que subcategoria es, a mi entender esto seguramente sea la parte mas compleja del proyecto no en cuanto a implementación de código sino en analizar como se puede hacer.

En primer lugar tener en cuenta que vamos a mostrar 2 registros de la misma tabla en mismo menú desplegable , para ello mediante una función basándome en la que me mostraba las categorías → **mostrar\_categorías\_admin()** a la cual le pasamos 2 parámetros vamos a crear **mostrar\_categorías\_admin\_seleccion()**

```
function mostrar_subcategorias_admin_seleccion($identificador, $categoria_de_producto)|
```

Donde **\$identificador** me está obteniendo el parámetro de la consulta que nos devuelve las subcategorías de la categoría actual , si la consulta SELECT se realiza correctamente toma los valores del option value.

Si miramos en **mostrar\_subcategorias\_admin()**, realizábamos una comparación entre el primer nivel que corresponde a la categoría padre **idPadre=0** con el segundo nivel .

Con la función SELECTED de PHP, podemos hacer que el formulario obtenga ese valor cuando entremos pero desconocemos ese valor , ya que se recibe por parámetro .

Podemos añadir un If que le pasamos el primer parámetro de la función y por otro lado le vamos a enviar también la 2º variable, se que es un poco confuso todo esto así que te muestro el código con el cual lo entenderemos mejor

```
function mostrar_subcategorias_admin_seleccion($identificador, $categoria_de_producto)
{
    global $databaseConexionZapatos, $conexionZapatos;
    mysql_select_db($databaseConexionZapatos, $conexionZapatos);
    $query_ConsultaFuncion = sprintf("SELECT * FROM tblCategoria WHERE idPadre = %s ORDER BY
        tblCategoria.strDescripcion", $identificador);
    $ConsultaFuncion = mysql_query($query_ConsultaFuncion, $conexionZapatos) or die(mysql_error());
    $row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion);
    $totalRows_ConsultaFuncion = mysql_num_rows($ConsultaFuncion);
    ?>
    <?php
    if ($totalRows_ConsultaFuncion > 0) {
        do {
            ?>
<option value=<?php echo $row_ConsultaFuncion['idCategoria']?>> <?php if (!strcmp($
    row_ConsultaFuncion['idCategoria'], $categoria_de_producto)) {echo "SELECTED";}>>&nbsp;&nbsp;
;?><?php echo $row_ConsultaFuncion['strDescripcion']?>></option>
        <?php
            } while ($row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion));
    }
    mysql_free_result($ConsultaFuncion);
}
```

Hay que tener muy en cuenta que la variable del bucle If no debe ser **strcmp(\$identificador)** sino **\$row\_ConsultaFuncion**.

Una vez tenemos esta función hemos de realizar la llamada a la función en el formulario de **productos\_edit.php**

```

<td>

    <select name="intCategoria">
        <?php
do {
?>
        | <option value=<?php echo $row_ConsultaCategorias['idCategoria']?>">
        | <?php if (!(strcmp($row_ConsultaCategorias['idCategoria'], htmlentities($row_DatosProducto['
            intCategoria'], ENT_COMPAT, 'iso-8859-1')))) {echo "SELECTED";} ?>>
        | <?php echo $row_ConsultaCategorias['strDescripcion']?>>
        | </option>
        |     <?php mostrar_subcategorias_admin_seleccion($row_ConsultaCategorias['idCategoria'], $>
            row_DatosProducto['intCategoria']);?>
        | <?php
} while ($row_ConsultaCategorias = mysql_fetch_assoc($ConsultaCategorias));
?>
    | </select></td>

```

Con esto ya nos seleccionará automáticamente la subcategoria correcta al editar el producto

The screenshot shows a form for editing a product. The fields are as follows:

- Nombre: Zapato 2
- SEO: sdfsfdf
- Precio: 20
- Estado: Activo ▾
- Categoría: Botas ▾
- Imagen: (empty)
- Stock: (empty)

A dropdown menu is open under the 'Categoria' field, listing subcategories:

- Botas
- Baratas
- Caras
- De montaña
- camino
- nieve
- De playa
- Caras
- Sandalias

Vamos a resumir esto para que así pueda quedar mas claro

A la función

```
function mostrar_subcategorias_admin_seleccion($identificador, $categoria_de_producto)|
```

le vamos a pasar la 1º variable **\$identificador** que sería la categoría actual con el fin de a través de la consulta SELECT sacar las subcategorias, con lo cual lo que realmente debe comparar es la subcategoria que yo estoy sacando en este momento con la que estamos enviando por parámetro para ver si coincide, si es la misma me aparecerá el

```
{echo "SELECTED";}
```

que lo que hace es seleccionar ese campo como tal .

## 7.2 GESTIÓN TALLAS.

### tallas\_lista.php

Siendo una tienda de zapatos seria interesante añadirle al menos las tallas , ademas que cada producto varié su precio en función de la talla .

Esto a priori no parece excesivamente interesante para vender zapatos puesto que en la vida real no varia el precio en función de la talla .Pero seria bastante interesante implementar esta característica ya que multitud de productos varia el precio en función por ejemplo de la cantidad que se compre,en función de la antigüedad etc. Otro ejemplo podría ser un hotel en el cual en función del numero de habitaciones que reserves varié su precio etc.

Tampoco vamos a aumentar la complejidad excesivamente dado que la variable que aumentara o reducirá el precio va a ser de tipo entero, si lo realizáramos con un varchar en el cual poder meter formulas matemáticas,seria bastante mas complejo, soy consciente que es la solución optima pero también mis conocimientos sobre php y mysql son limitados y he de saber hasta donde puedo llegar.

Realizamos un apartado nuevo en la administración para las tallas , añadimos un enlace GESTIÓN TALLAS a una nueva pagina → **tallas\_lista.php**

Antes de crear la nueva pagina vamos a insertar una nueva tabla en nuestra BD → **tbltallas**

The screenshot shows the MySQL Workbench interface with the following details:

- Servidor: 127.0.0.1
- Base de datos: zapatos
- Tabla: tbltallas
- Toolbar buttons: Examinar, Estructura, SQL, Buscar, Insertar
- Table structure:

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Pred
1	<b>idTalla</b>	int(11)			No	Ninguna
2	<b>strNombre</b>	varchar(10)	latin1_swedish_ci		No	Ninguna
3	<b>intAumento</b>	int(11)			No	Ninguna

Para gestionar las tallas nos vale una pagina similar a **productos\_lista.php** que modificaremos

Vamos a realizar una consulta SELECT a la tabla **tbltallas** que nos muestre todos los registros ordenados por nombre que serán el numero de talla ademas realizaremos una paginación para que nos muestre 10 tallas por pagina lo cual sera mas que suficiente

```
$maxRows_Recordset1 = 10;
$pageNum_Recordset1 = 0;
if (isset($_GET['pageNum_Recordset1'])) {
    $pageNum_Recordset1 = $_GET['pageNum_Recordset1'];
}
$startRow_Recordset1 = $pageNum_Recordset1 * $maxRows_Recordset1;

mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_Recordset1 = "SELECT * FROM tbltallas ORDER BY tbltallas.strNombre ASC";
$query_limit_Recordset1 = sprintf("%s LIMIT %d, %d", $query_Recordset1, $startRow_Recordset1, $maxRows_Recordset1);
$Recordset1 = mysql_query($query_limit_Recordset1, $conexionzapatos) or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($Recordset1);

if (isset($_GET['totalRows_Recordset1'])) {
    $totalRows_Recordset1 = $_GET['totalRows_Recordset1'];
} else {
    $all_Recordset1 = mysql_query($query_Recordset1);
    $totalRows_Recordset1 = mysql_num_rows($all_Recordset1);
}
$totalPages_Recordset1 = ceil($totalRows_Recordset1/$maxRows_Recordset1)-1;
?>
```

No he nombrado antes la clausula LIMIT pero es para que nos realice la paginación, añadiremos también que cuando no existan tallas activas nos muestre por pantalla que no hay tallas disponibles ya hemos explicado esto anteriormente y no lo faremos de nuevo aquí.

## 7.2.1 AÑADIR TALLAS.

### tallas\_add.php

Vamos con el proceso de añadir tallas y añadir las tallas a los productos .Tanto añadir tallas como editar y eliminar ya lo hemos realizado varias veces,

En tallas\_add.php, y creamos un formulario para insertar registros y realizamos una INSERT

The screenshot shows a web application interface. On the left, there's a sidebar with a yellow background containing navigation links: 'Productos', 'Categorías', 'Usuarios', 'Tallas', 'Compras', and 'Otras'. The main content area has a light blue background. At the top, it says 'Contenido' and 'Añadir Talla'. Below this is a form with two text input fields labeled 'Nombre:' and 'Aumento:', and a submit button labeled 'Insertar Talla'.

```
<form action=<?php echo $editFormAction; ?>" method="post" name="form1" id="form1">
    <table align="center">
        <tr valign="baseline">
            <td nowrap="nowrap" align="right">Nombre:</td>
            <td><input type="text" name="strNombre" value="" size="32" /></td>
        </tr>
        <tr valign="baseline">
            <td nowrap="nowrap" align="right">Aumento:</td>
            <td><input type="text" name="intAumento" value="" size="10" /></td>
        </tr>
        <tr valign="baseline">
            <td nowrap="nowrap" align="right">&ampnbsp</td>
            <td><input type="submit" value="Insertar Talla" /></td>
        </tr>
    </table>
    <input type="hidden" name="MM_insert" value="form1" />
</form|
```

Y a continuación la consulta

```
if ((isset($_POST["MM_insert"])) && ($_POST["MM_insert"] == "form1")) {
    $insertSQL = sprintf("INSERT INTO tbltallas (strNombre, intAumento) VALUES (%s, %s)",
        GetSQLValueString($_POST['strNombre'], "text"),
        GetSQLValueString($_POST['intAumento'], "int"));

    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $Result1 = mysql_query($insertSQL, $conexionzapatos) or die(mysql_error());

    $insertGoTo = "tallas_lista.php";
    if (isset($_SERVER['QUERY_STRING'])) {
        $insertGoTo .= (strpos($insertGoTo, '?')) ? "&" : "?";
        $insertGoTo .= $_SERVER['QUERY_STRING'];
    }
    header(sprintf("Location: %s", $insertGoTo));
}
```

<a href="#">Lista Categorías</a>	<a href="#">Añadir Talla</a>		
	TALLA	AUMENTO	ACCIONES
<a href="#">Lista Usuarios</a>	36	0	<a href="#">Editar</a> - <a href="#">Eliminar</a>
<a href="#">Gestion Tallas</a>	37	0	<a href="#">Editar</a> - <a href="#">Eliminar</a>
<b>Compras</b>	38	2	<a href="#">Editar</a> - <a href="#">Eliminar</a>
	39	2	<a href="#">Editar</a> - <a href="#">Eliminar</a>
	40	2	<a href="#">Editar</a> - <a href="#">Eliminar</a>
<a href="#">Listado</a>	41	5	<a href="#">Editar</a> - <a href="#">Eliminar</a>
	42	5	<a href="#">Editar</a> - <a href="#">Eliminar</a>
	43	5	<a href="#">Editar</a> - <a href="#">Eliminar</a>

Administracion Tienda Zapatos

## 7.2.2 EDITAR TALLAS.

### tallas\_edit.php

Para editar las tallas necesitamos recuperar el valor de la talla que deseamos modificar mediante una consulta SELECT a la cual le vamos a pasar un parámetro varTalla que recogerá por parámetro del recordID `$_GET["recordID"]`

```
$varTalla_Recordset1 = "0";
if (isset($_GET["recordID"])) {
    $varTalla_Recordset1 = $_GET["recordID"];
}
mysql_select_db($databaseConexionZapatos, $conexionZapatos);
$query_Recordset1 = sprintf("SELECT * FROM tbltallas WHERE tbltallas.idTalla = %s",
    GetSQLValueString($varTalla_Recordset1, "int"));
$Recordset1 = mysql_query($query_Recordset1, $conexionZapatos) or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($Recordset1);
$totalRows_Recordset1 = mysql_num_rows($Recordset1);
?>
```

También necesitamos un UPDATE

```
if ((isset($_POST["MM_update"])) && ($_POST["MM_update"] == "form1")) {
    $updateSQL = sprintf("UPDATE tbltallas SET strNombre=%s, intAumento=%s WHERE idTalla=%s",
        GetSQLValueString($_POST['strNombre'], "text"),
        GetSQLValueString($_POST['intAumento'], "int"),
        GetSQLValueString($_POST['idTalla'], "int"));

    mysql_select_db($databaseConexionZapatos, $conexionZapatos);
    $Result1 = mysql_query($updateSQL, $conexionZapatos) or die(mysql_error());

    $updateGoTo = "tallas_listado.php";
    if (isset($_SERVER['QUERY_STRING'])) {
        $updateGoTo .= (strpos($updateGoTo, '?')) ? "&" : "?";
        $updateGoTo .= $_SERVER['QUERY_STRING'];
    }
    header(sprintf("Location: %s", $updateGoTo));
```

## 7.2.3 ELIMINAR TALLAS.

### tallas\_delete.php

Redireccionamos a **tallas\_delete.php** y ejecutamos un DELETE.

Para eliminar las tallas necesitamos recuperar el valor de la talla que deseamos modificar mediante una consulta SELECT a la cual le vamos a pasar un parámetro varTalla que recogerá por parámetro del recordID `$_GET["recordID"]`

```
}

if ((isset($_GET['recordID'])) && ($_GET['recordID'] != "")) {
    $deleteSQL = sprintf("DELETE FROM tbltallas WHERE idtalla=%s",
                         GetSQLValueString($_GET['recordID'], "int"));

    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $Result1 = mysql_query($deleteSQL, $conexionzapatos) or die(mysql_error());

    $deleteGoTo = "tallas_lista.php";
    if (isset($_SERVER['QUERY_STRING'])) {
        $deleteGoTo .= (strpos($deleteGoTo, '?')) ? "&" : "?";
        $deleteGoTo .= $_SERVER['QUERY_STRING'];
    }
    header(sprintf("Location: %s", $deleteGoTo));
}
```

## 7.3.4 ASIGNACIÓN DE TALLAS A DIFERENTES PRODUCTOS.

### productotalla\_lista.php

Vamos a utilizar una tabla `tblproductotalla` que nos relacione cada producto con las tallas que tiene. Con esto conseguimos que nos relacione el id de cada producto con el id de cada talla . A efectos de funcionamiento de la pagina no va tener importancia solo nos servira para relacionar 2 tablas .

Creamos un enlace en **producto\_lista.php** a tallas lo redireccionamos **productotalla\_lista.php**, le vamos a enviar el parametro recordID que sera el id del producto del cual queremos sacar las tallas

Vamos con la consulta como se trata de una consulta multitabla donde queremos obtener el nombre(strNombre) y el aumento(intAumento) dado un producto(relProducto ) mediante un INNER JOIN podemos obtener estos valores

```

}

$maxRows_Recordset1 = 10;
$pageNum_Recordset1 = 0;
if (isset($_GET['pageNum_Recordset1'])) {
    $pageNum_Recordset1 = $_GET['pageNum_Recordset1'];
}
$startRow_Recordset1 = $pageNum_Recordset1 * $maxRows_Recordset1;

mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_Recordset1 = "SELECT tbltallas.strNombre, tbltallas.intAumento
FROM tblproductotalla
Inner Join tbltallas ON tblproductotalla.relTalla = tbltallas.idTalla
WHERE tblproductotalla.relProducto = '".$_GET["recordID"]';

$query_limit_Recordset1 = sprintf("%s LIMIT %d, %d", $query_Recordset1, $startRow_Recordset1, $maxRows_Recordset1);
$Recordset1 = mysql_query($query_limit_Recordset1, $conexionzapatos) or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($Recordset1);

if (isset($_GET['totalRows_Recordset1'])) {
    $totalRows_Recordset1 = $_GET['totalRows_Recordset1'];
} else {
    $all_Recordset1 = mysql_query($query_Recordset1);
    $totalRows_Recordset1 = mysql_num_rows($all_Recordset1);
}
$totalPages_Recordset1 = ceil($totalRows_Recordset1/$maxRows_Recordset1)-1;

```

Vamos a añadir tambien un boton de eliminar que nos llevara a la pagina **productotallas\_delete.php** ademas de un boton para añadir Talla a un producto determinado que nos llevara a **productotallas\_add.php**, para añadir hacemos un SELECT y posteriormente un INSERT

```

mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_Recordset1 = "SELECT * FROM tbltallas WHERE tbltallas.strNombre";
$Recordset1 = mysql_query($query_Recordset1, $conexionzapatos) or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($Recordset1);
$totalRows_Recordset1 = mysql_num_rows($Recordset1);

```

```

    $editFormAction .= ". . . PREMICIENTES($_SERVER[ 'QUERY_STRING ']),
}

if ((isset($_POST["MM_insert"])) && ($_POST["MM_insert"] == "form1")) {
$insertSQL = sprintf("INSERT INTO tblproductotalla (relProducto, relTalla) VALUES (%s, %s)",
    GetSQLValueString($_POST['relProducto'], "int"),
    GetSQLValueString($_POST['relTalla'], "int"));

mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$Result1 = mysql_query($insertSQL, $conexionzapatos) or die(mysql_error());

$insertGoTo = "productotalla_lista.php?recordID=".$_POST['relProducto'];
if (isset($_SERVER[ 'QUERY_STRING '])) {
    $insertGoTo .= (strpos($insertGoTo, '?')) ? "&" : "?";
    $insertGoTo .= $_SERVER[ 'QUERY_STRING '];
}
header(sprintf("Location: %s", $insertGoTo));
}

```

Dado que estamos seleccionando datos de 2 tablas el formulario es un poco mas complejo, sera necesario introducir campos que no nos interesan pero son necesarios para su funcionamiento así que los pondremos como ocultos (type hidden)

```

<form action=<?php echo $editFormAction; ?>" method="post" name="form1" id="form1">

    <table align="center">
        <tr valign="baseline">
            <td nowrap="nowrap" align="right">Talla:</td>
            <td><select name="relTalla">
                <?php
                    do {?>
                        <option value=<?php echo $row_Recordset1['idTalla']?>><?php echo $row_Recordset1['strNombre']?></option>
                    <?php
                }
                while ($row_Recordset1 = mysql_fetch_assoc($Recordset1));
            ?>
                </select></td>
            </tr>
            <tr> </tr>
            <tr valign="baseline">
                <td nowrap="nowrap" align="right">&ampnbsp</td>
                <td><input type="submit" value="Insertar Talla En Producto" /></td>
            </tr>
        </table>
        <input type="hidden" name="relProducto" value=<?php echo $_GET["recordID"]; ?>" />
        <input type="hidden" name="MM_insert" value="form1" />
    </form>

```

En **relProducto** le pasaremos el parámetro que nos relaciona el producto con la talla.

Tendremos que añadir en la pagina **productostalla\_lista.php** al enlace Añadir Talla producto el parámetro en este caso recordID

```

<p><a href="productotallas_add.php?recordID=<?php echo $_GET["recordID"]; ?>">Añadir
Talla a Producto</a></p>

```

Ahora ya pasamos el parámetro a **relProducto** que es el campo que ocultamos

```

<input type="hidden" name="relProducto" value=<?php echo $_GET["recordID"]; ?>" />
<input type="hidden" name="MM_insert" value="form1" />

```

Tambien hemos e pasar el parámetro en la consulta INSERT que realizamos anteriormente en **productotallas\_add.php** cuando realice la operación que nos lleva a **productotalla\_lista.php** y al producto que estamos mirando actualmente, como se lo pasamos por formulario → **\$\_POST[“relProducto”]**

```

if ((isset($_POST["MM_insert"])) && ($_POST["MM_insert"] == "form1")) {
    $insertSQL = sprintf("INSERT INTO tblproductotalla (relProducto, relTalla) VALUES (%s, %s)",
        GetSQLValueString($_POST['relProducto'], "int"),
        GetSQLValueString($_POST['relTalla'], "int"));

    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $Result1 = mysql_query($insertSQL, $conexionzapatos) or die(mysql_error());

    $insertGoTo .= "productotalla_lista.php?recordID=".$_POST['relProducto'];
    if (isset($_SERVER['QUERY_STRING'])) {
        $insertGoTo .= (strpos($insertGoTo, '?')) ? "&" : "?";
        $insertGoTo .= $_SERVER['QUERY_STRING'];
    }
    header(sprintf("Location: %s", $insertGoTo));
}

```

### 7.3.5 MOSTRAR TALLAS EN FRONTEND.

(Parte publica de la tienda)

La idea es mostrar las tallas cuando un cliente compra un zapato, esa talla ira referenciada en el carrito de la compra y en el caso que tenga un aumento de precio se sume al precio final .

Primero en la pagina **ver\_producto.php** vamos a añadir un selector de tallas . Este selector los podrán ver tanto usuarios registrados como no registrados en el sistema .

Vamos a implementar una función la cual le pasaremos por parámetro el identificador del producto , la cual llamaremos en ver\_producto.php

```
<?php mostrartallasdisponibles($row_DatosProducto['idProducto']);?>
```

Vamos a implementar la función la dificultad de esta función se basa en que tenemos que hacer una consulta multitable en la cual utilizaremos INNER JOIN ya que deseamos conocer dado un idproducto obtener las tallas que tiene asociada .

La consulta sera entre las tablas **tblproductotallas** y **tbltallas**

```

global $databaseConexionzapatos, $conexionzapatos;
mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_ConsultaFuncion = sprintf("SELECT tbltallas.idTalla, tbltallas.strNombre, tbltallas.
    intAumento FROM tblproductotalla Inner Join tbltallas ON tblproductotalla.relTalla =
    tbltallas.idTalla WHERE tblproductotalla.relProducto = %s", $identificador);
$ConsultaFuncion = mysql_query($query_ConsultaFuncion, $conexionzapatos) or die(mysql_error());
$row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion);
$totalRows_ConsultaFuncion = mysql_num_rows($ConsultaFuncion);

```

Ya tenemos la consulta ahora hemos ahora establecemos el bucle de control Do-While-Else para que se ejecute correctamente

```

<?php
if ($totalRows_ConsultaFuncion > 0) {
?>
Tallas:<select name="FTalla">
<?php
    do {
        ?>
<option value=<?php echo $row_ConsultaFuncion['strNombre']?>>">
    <?php echo $row_ConsultaFuncion['strNombre']?></option>
    <?php
        } while ($row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion));
    ?>
    </select>
    <?php
}
else
{
    echo "No hay tallas disponibles";
}

mysql_free_result($ConsultaFuncion);

```

Finalmente la función quedaría:

```

function mostrartallasdisponibles($identificador)
{
    global $databaseConexionzapatos, $conexionzapatos;
    mysql_select_db($databaseConexionzapatos, $conexionzapatos);
    $query_ConsultaFuncion = sprintf("SELECT tbltallas.idTalla, tbltallas.strNombre, tbltallas.intAumento FROM
        tblproductotalla Inner Join tbltallas ON tblproductotalla.relTalla = tbltallas.idTalla WHERE tblproductotalla.
        relProducto = %s", $identificador);
    $ConsultaFuncion = mysql_query($query_ConsultaFuncion, $conexionzapatos) or die(mysql_error());
    $row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion);
    $totalRows_ConsultaFuncion = mysql_num_rows($ConsultaFuncion);
    ?>
    <?php
    if ($totalRows_ConsultaFuncion > 0) {
    ?>
    Tallas:<select name="FTalla">
    <?php
        do {
            ?>
<option value=<?php echo $row_ConsultaFuncion['strNombre']?>>">
        <?php echo $row_ConsultaFuncion['strNombre']?></option>
        <?php
            } while ($row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion));
        ?>
        </select>
        <?php
    }
    else
    {
        echo "No hay tallas disponibles";
    }

    mysql_free_result($ConsultaFuncion);
}

```

Hemos insertado un elemento html dentro del If para meter un formulario de lista de menú, dentro de este **select** irán los valores del options y añadimos un **Else** para cuando no existan tallas.

Con esto ya nos mostraría las tallas disponibles en cada producto

# Zapato Tacon



Zapato Tacon

897 €

Tallas: 37 ▾

Ahora nos falta a la hora de comprar el producto que nos añada la talla a la compra y si posee un suplemento de precio que se sume al total .

Cuando pulsamos comprar es un enlace que nos llevaba a **carrito\_add.php** y le enviaba el id del producto, como hemos añadido un campo nuevo y es un campo de un formulario hemos de transformar la información que nos muestra.

Es decir no podemos obtener las variables de un formulario con solo echo, hemos de añadir algo mas para que funcione correctamente, en nuestro caso vamos a modificar los enlaces por botones tipo submit

Así que el botón comprar va a pasar de ser un link de texto a ser un botón, realizaremos las modificaciones en **ver\_producto.php**

Aquí hay algo curioso que comentar en el method del formulario no vamos a usar POST sino GET, ya que todos los datos del formulario se están enviando por url, esto facilita el tema de no tener que modificar completamente el carrito de la compra .

El enlace comprar se transforma en un formulario tipo submit (botón), ademas vamos a necesitar para gestionar esto un campo mas el idproducto , sera el que enviemos a **carrito\_add.php** pero no lo mostraremos así que lo pondremos como hidden

```
<input name="recordID" type="hidden" value=<?php echo $row_DatosProducto['idProducto']; ?>/>
<input name="" type="submit" value="Comprar" /></p>
</div>
```

En resumen tenemos un formulario que incluye una función que obtiene un menú desplegable que permite ver los tipos de productos que hay disponibles y posteriormente un botón que envía el formulario, este botón de enviar solo tendrá sentido cuando el usuario este registrado.

Modificamos también la consulta INSERT que realizábamos en **carrito\_add.php**, añadiendo la talla pues que es un registro que también deseamos obtener.

Añadimos un campo nuevo en nuestra tabla **tblcarrito** que sera **intTalla** que referencia la tabla **tbltallas**

```
$insertSQL = sprintf("INSERT INTO tblcarrito (idUsuario, idProducto, intCantidad, intTalla) VALUES (%s, %s, %s, %s)",
    GetSQLValueString($_SESSION['MM_IdUsuario'], "int"),
    GetSQLValueString($_GET['recordID'], "text"),
    GetSQLValueString($_GET['intCantidad']),
    GetSQLValueString($_GET['FTalla']));
```

**intTalla** tomara el valor de **FTalla**, que coincide con como llamamos al menú desplegable .

Solamente ahora nos falta que el carrito nos muestre la talla añadimos un echo para que nos lo muestre

```
(<?php echo $row_DatosCarrito['intTalla']; ?>)</td>
```

Ya tendríamos la talla en el carrito de la compra:



Nuestra base de datos reconocerá esta talla gracias al campo que añadimos **intTalla**,

### 7.3.6 COMPRA SIMULTANEA DE VARIOS PRODUCTOS.

Vamos a realizar la compra de varios productos de un mismo tipo siempre y cuando existan tallas disponibles.

El proceso va a ser muy parecido a como implementamos las tallas, realizaremos también un desplegable para seleccionar el numero de unidades vamos a introducir una lista de valores para cada elemento es decir para cada numero de unidades.

```
Unidades: <select name="intCantidad" id="intCantidad">
    <option value="1">1</option>
    <option value="2">2</option>
    <option value="3">3</option>
    <option value="4">4</option>
    <option value="5">5</option>
    <option value="6">6</option>
    <option value="7">7</option>
    <option value="8">8</option>
    <option value="9">9</option>
    <option value="10">10</option>
    <option value="15">15</option>
    <option value="20">20</option>
    <option value="25">25</option>
</select>
```

Estas unidades hemos de pasarlas también a nuestro carrito de la compra en **carrito\_add.php** el cual ha de recoger el valor, modificamos la consulta INSERT

```
$insertSQL = sprintf("INSERT INTO tblcarrito (idUsuario, idProducto, intCantidad, intTalla) VALUES (%s, %s, %s, %s)",
    GetSQLValueString($_SESSION['MM_1dUsuario'], "int"),
    GetSQLValueString($_GET['recordID'], "text"),
    GetSQLValueString($_GET['intCantidad']),
    GetSQLValueString($_GET['FTalla']));
```

Necesitamos también que nos multiplique el valor del producto por el numero de unidades seleccionadas en el carrito

```
<td><?php echo $row_DatosCarrito['intCantidad']*ObtenerPrecioProducto($row_DatosCarrito['idProducto'])
; ?> Euros</td>
```

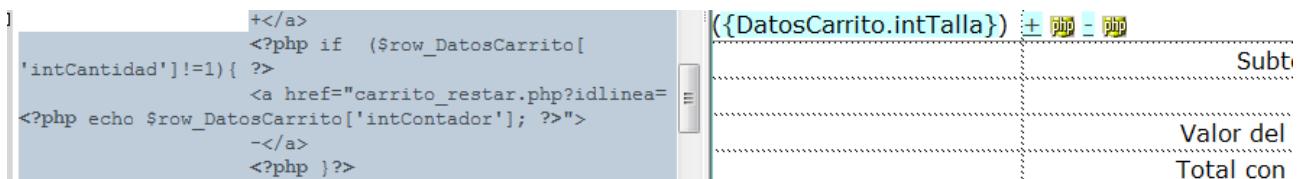
De manera análoga para el precio total

```
<?php
$preciototal = $preciototal + ($row_DatosCarrito['intCantidad']*ObtenerPrecioProducto($row_DatosCarrito['idProducto']));
; ?>
```

Vamos añadir también el precio por unidad para que sea la información sea visualmente mas clara, Nos aprovechamos de la función que implementamos de obtener el precio para ello

```
<td><?php echo ObtenerPrecioProducto($row_DatosCarrito['idProducto']);?> &euro;</td>
```

Añadimos también unos botones para añadir unidades en el carrito mediante dos enlaces que actuaran como botones de + y -



Estos botones nos redireccionan a dos paginas llamadas **carrito\_sumar.php** y **carrito\_restar.php**.

Cuando pulsemos en el botón sumar ha de incrementar el valor en 1 unidad en el campo en la base de datos , tenemos el problema que el campo **intCantidad** sus valores se repiten en la tabla dado que podemos tener muchos productos con la misma cantidad, así que no es un campo efectivo a la hora de incrementar unidades y la base de datos sepa donde actualizar esos datos.

Podemos usar la clave primaria de la tabla la cual su valor para cada producto es único y sera fácil de referenciar a la hora de modificar la cantidad .

A la pagina **carrito\_sumar.php** le vamos a pasar un parámetro, que sera **idlinea** que contendrá el campo **intContador** , la cual es la PK de la tabla y realizaremos lo mismo para **carrito\_restar.php**

```
<a href="carrito_sumar.php?idlinea=<?php echo $row_DatosCarrito['intContador']; ?>">+</a>
<?php if  ($row_DatosCarrito['intCantidad']!=1){ ?>
<a href="carrito_restar.php?idlinea=<?php echo $row_DatosCarrito['intContador']; ?>">-</a>
```

En **carrito\_sumar.php**, vamos a realizar un UPDATE, aumentando en valor intCantidad en 1

```
$insertSQL = sprintf("UPDATE tblcarrito SET intCantidad = intCantidad + 1 WHERE intContador = %s",
| | | | | GetSQLValueString($_GET['idlinea']));
mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$result1 = mysql_query($insertSQL, $conexionzapatos) or die(mysql_error());

$insertGoTo = "carrito_lista.php";
if (isset($_SERVER['QUERY_STRING'])) {
| $insertGoTo .= (strpos($insertGoTo, '?')) ? "&" : "?";
| $insertGoTo .= $_SERVER['QUERY_STRING'];
}
header(sprintf("Location: %s", $insertGoTo));
```

Para **carrito\_restar.php** similar pero disminuyendo una unidad.

```
$insertSQL = sprintf("UPDATE tblcarrito SET intCantidad = intCantidad - 1 WHERE intContador = %s",
| | | | | GetSQLValueString($_GET['idlinea']));
mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$result1 = mysql_query($insertSQL, $conexionzapatos) or die(mysql_error());

$insertGoTo = "carrito_lista.php";
if (isset($_SERVER['QUERY_STRING'])) {
| $insertGoTo .= (strpos($insertGoTo, '?')) ? "&" : "?";
| $insertGoTo .= $_SERVER['QUERY_STRING'];
}
header(sprintf("Location: %s", $insertGoTo));
```

Veo bastante complicado implementar que cuando solo exista una unidad y restemos uno el producto desaparezca del carrito.

Una solución es que cuando solo haya una unidad no se te permita pulsar el botón restar unidades .

Añadimos la condición If para que solo se vea cuando el valor es mayor que 1

```
<a href="carrito_sumar.php?idlinea=<?php echo $row_DatosCarrito['intContador']; ?>">+</a>
<?php
if ($row_DatosCarrito['intCantidad']!=1)
{ ?>
<a href="carrito_restar.php?idlinea=<?php echo $row_DatosCarrito['intContador']; ?>">-</a>
<?php }
?>
```

Botas
Baratas
Caras
De montaña
camino
baratas
playa
Baratas
Caras
Sandalias
Hola Frank
Carrito de la Compra
Mis Compras

## Carrito de la compra

Siquieres, puedes [vaciar](#) tu carrito de la compra.

Producto	Unidades	Precio unidad	Precio	Acciones
Botas nuevas (40)	3 ±	34 €	102 Euros	<a href="#">Eliminar</a>
Zapato Tacon (37)	1 ±	897 €	897 Euros	<a href="#">Eliminar</a>
Subtotal:				999 Euros
IVA:				21%
Valor del IVA:				209.79 Euros
Total con IVA:				1208.79 Euros

[Seleccionar Forma de Pago](#)

Otro error que he encontrado es que al añadir un mismo producto en el carrito 2 veces aparecen como 2 productos y lo ideal es que se sume si ya existe ese producto

## Carrito de la compra

Siquieres, puedes [vaciar](#) tu carrito de la compra.

Producto	Unidades	Precio unidad	Precio	Acciones
Zapato Tacon ()	1 ±	897 €	897 Euros	<a href="#">Eliminar</a>
Bota de Agua ()	3 ±	20 €	60 Euros	<a href="#">Eliminar</a>
Tacon Alto (36)	4 ±	234 €	936 Euros	<a href="#">Eliminar</a>
Bota de Agua (40)	4 ±	20 €	80 Euros	<a href="#">Eliminar</a>
Tacon Alto (39)	3 ±	234 €	702 Euros	<a href="#">Eliminar</a>
Bota de Agua (40)	1 ±	20 €	20 Euros	<a href="#">Eliminar</a>
Subtotal:				2695 Euros
IVA:				18%
Valor del IVA:				485.1 Euros
Total con IVA:				3180.1 Euros

[Seleccionar Forma de Pago](#)

En **carrito\_add.php** tenemos la INSERT que añade el producto, si justo antes de esta operación deberíamos comprobar si este producto existe ya en nuestro carrito, vamos a crear una nueva función llamada **comprobareexistencia()**, cual va necesitar diversos parámetros

```
comprobareexistencia($_GET['recordID'],$_GET['FTalla']);
```

Le pasamos los parámetros y mediante un If va a comprobar la existencia y va a realizar el UPDATE en ese caso

```
$valorrespuesta = comprobareexistencia($_GET['recordID'],$_GET['FTalla']);
if ($valorrespuesta!=0){
    //UPDATE
    $insertSQL = sprintf("UPDATE tblcarrito SET intCantidad = intCantidad + %s WHERE intContador = %s", $_GET['intCantidad'],
    $valorrespuesta);
}
else
{
    $insertSQL = sprintf("INSERT INTO tblcarrito (idUsuario, idProducto, intCantidad, intTalla) VALUES (%s, %s, %s, %s)",
    GetSQLValueString($_SESSION['MM_IdUsuario'], "int"),
    GetSQLValueString($_GET['recordID'], "text"),
    GetSQLValueString($_GET['intCantidad']),
    GetSQLValueString($_GET['FTalla']));
}
```

Una vez esto vamos a implementar la función

```
function comprobareexistencia($idproducto, $idtalla)
{
    global $database_conexionzapatos, $conexionzapatos;
    mysql_select_db($database_conexionzapatos, $conexionzapatos);
    $query_ConsultaFuncion = sprintf("SELECT * FROM tblcarrito WHERE idUsuario = %s AND idProducto=%s AND intTalla = %s",
        $_SESSION['MM_IdUsuario'],$idproducto, $idtalla );
    $ConsultaFuncion = mysql_query($query_ConsultaFuncion, $conexionzapatos) or die(mysql_error());
    $row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion);
    $totalRows_ConsultaFuncion = mysql_num_rows($ConsultaFuncion);

    if ($totalRows_ConsultaFuncion >0)
        return $row_ConsultaFuncion['intContador'];
    else
        return 0;
    mysql_free_result($ConsultaFuncion);
}
```

Esta función es una sencilla SELECT con la peculiaridad de de un bucle de control al final que si encuentra un producto con esas características nos devuelve el valor del **intContador** y sino devuelve 0, y con esto estamos averiguando el valor de la linea donde tenemos que modificar el dato.

Por ultimo añadimos una variable **\$valorrespuesta** que contendrá dicha función y con un If cuando el valor sea distinto de 0 me realice el UPDATE e introduzca el valor de **\$valorrespuesta** y sino se cumple realiza el INSERT normal , viendo el código se comprenderá mejor esto

```
}
{
    $valorrespuesta = comprobareexistencia($_GET['recordID'],$_GET['FTalla']);
    if ($valorrespuesta!=0){
        //UPDATE
        $insertSQL = sprintf("UPDATE tblcarrito SET intCantidad = intCantidad + %s WHERE intContador = %s",$_GET['intCantidad'],
            $valorrespuesta);
    }
    else
    {
        $insertSQL = sprintf("INSERT INTO tblcarrito (idUsuario, idProducto, intCantidad, intTalla) VALUES (%s, %s, %s, %s)",
            GetSQLValueString($_SESSION['MM_IdUsuario'], "int"),
            GetSQLValueString($_GET['recordID'], "text"),
            GetSQLValueString($_GET['intCantidad']),
            GetSQLValueString($_GET['FTalla']));
    }
}
```

He detectado otro pequeño error si yo quiero comprar un producto que ya había comprado otra vez, el carrito sale vacío, mirando el código de lo ultimo que realizamos he encontrado el error en la función que implementamos comprobar existencia no funciona correctamente la consulta SELECT voy a mostrarla para recordar

```
$query_ConsultaFuncion = sprintf("SELECT * FROM tblcarrito WHERE idUsuario = %s AND idProducto=%s AND intTalla = %s
    AND intTransaccionEfectuada = 0", $_SESSION['MM_IdUsuario'],$idproducto, $idtalla );
```

el error se ve claramente, la consulta nos debe mostrar los registros cuando el campo intTransaccionEfectuada es 0 , eso significa que estamos en el carrito actual, antes de modificar esto nos seleccionaba un carrito anterior.

## 7.3 MEJORA DE ASPECTO.CSS .

Vamos a mejorar un poco el aspecto de la pagina la cual hasta el momento no es nada atractiva. No vamos a entrar en detalle de las propiedades CSS utilizadas ya que no es el objeto del proyecto, sino implementar código PHP.

Mostrare como va quedando con unos pequeños retoques de CSS.

The screenshot shows a web application interface. On the left side, there is a sidebar with a dark blue header labeled "CATALOGO". Below it, there is a list of categories: Botas, Baratas, Caras, De montaña, camino, baratas, playa, Baratas, Caras, Sandalias. Underneath this list, the user's name "Hola Frank" is displayed, followed by two links: "Carrito de la Compra" and "Mis Compras". At the bottom of the sidebar, there are two more links: "Modificar" and "Salir". The main content area on the right has a large title "Instrucciones" centered above a "Contenido" section.

This screenshot shows the same web application after applying CSS improvements. The overall design is more polished. A prominent brown header bar at the top contains the title "Instrucciones" in large, bold, black font. Below the header, the sidebar and main content area are identical to the first screenshot, maintaining the same layout and content structure.

<b>CATALOGO</b>	<b>Productos ...</b>
Botas Baratas Caras De camping De montaña Nevada De playa Sandalias	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Boota de Agua. Precio: 20 Stock: 89</p> </div> <div style="text-align: center;"> <p>Zapato Tacón. Precio: 897 Stock: 0</p> </div> <div style="text-align: center;"> <p>Zapato standard. Precio: 87 Stock: 0</p> </div> </div> <div style="margin-top: 10px;"> <b>Hola Jorge Pepe</b> </div> <div style="display: flex; justify-content: space-between;"> <div style="flex: 1;"> <a href="#">Carrito de la Compra</a>  <a href="#">Mis Compras</a>  <a href="#">Modificar</a>  <a href="#">Salir</a> </div> <div style="flex: 1;"> </div> </div>

## 7.4 NOMBRE DE LA CATEGORÍA EN CADA SECCIÓN DE PRODUCTOS.

Cuando estamos visualizando un producto, sera conveniente informar a la categoría que pertenece.

Para ello en **categorias\_ver.php**, necesitamos una función que dada una categoría nos muestre el nombre de esa categoría, no tenemos ninguna función implementada que realice esto pero nos podemos basar en otra función que nos obtenía el nombre de un producto, claro esta modificando algunos parámetros

Implementamos la función **ObtenerNombreCategoria()**

```
function ObtenerNombreCategoria($identificador)
{
  global $databaseConexionZapatos, $conexionZapatos;
  mysql_select_db($databaseConexionZapatos, $conexionZapatos);
  $query_ConsultaFuncion = sprintf("SELECT strDescripcion FROM tblcategoria WHERE idCategoria = %s", $identificador);
  $ConsultaFuncion = mysql_query($query_ConsultaFuncion, $conexionZapatos) or die(mysql_error());
  $row_ConsultaFuncion = mysql_fetch_assoc($ConsultaFuncion);
  $totalRows_ConsultaFuncion = mysql_num_rows($ConsultaFuncion);

  return $row_ConsultaFuncion['strDescripcion'];
  mysql_free_result($ConsultaFuncion);
}
```

Añadimos un echo donde queremos mostrar el valor de la categoría asociada a un producto

```
Productos de <?php echo ObtenerNombreCategoria($_GET["cat"]); ?>
```

En este caso el identificador de la función es un poco mas complejo de encontrar, si situamos el ratón encima de un producto podemos ver en la dirección del enlace como esta obteniendo la categoría asociada

```
http://localhost/zapatos/categoria_ver.php?cat=1
```

Por lo tanto cat sera el valor que le estamos pasando por parámetro GET

```
[$_GET["cat"]]
```

Con esto ya tendríamos el nombre de las categorias cuando estemos en un producto.

CATALOGO

Botas  
Baratas  
Caras  
De camping  
De montaña  
Nevada  
De playa  
Sandalias

Productos de Botas

USUARIO

Darme de Alta  
Acceder

Bota de Agua.  
Stock: 89  
PVP: 20€

Zapato Tacon.  
Stock: 0  
PVP: 897€

Tacon Economico.  
Stock: 24

Zapato azul.  
Stock: 0

## 7.5 PRODUCTOS EN OFERTA.

En **index.php** estaría bien que saliera un lista de productos en oferta nada mas acceder a la web, ya le añadiría un plus para que el cliente comprara algún ítem. Para ello necesitamos un nuevo campo en nuestra BD .

Añadimos en la tabla **tblproductos** el campo **intOferta**:

- intOferta=1 → Es oferta
- intOferta=0 → No es oferta

Valor por defecto 0 , ya que en principio solo serán ofertas aquellos que nosotros deseemos y no cualquier producto introducido en la BD.

Vamos a necesitar también cuando editemos un producto poder seleccionar si es una oferta o no , podemos añadir una casilla de verificación, → Check , cuyo valor activado es 1, y su estado inicial sera desactivado .

```
<td>
<input <?php if (!(strcmp($row_DatosProducto['intOferta'],1))) {echo "checked=\\"checked\\\";}>
name="checkbox" type="checkbox" id="checkbox" value="1" />
<label for="checkbox"></label>
</td>
```

El echo nos mostrar la casilla marcada o no en función del valor del producto en la base de datos , solamente con esto no nos funcionaria ya que la actualización se realiza en base a una consulta tendremos que modificar la consulta de nuevo

```
if ((isset($_POST["MM_update"])) && ($_POST["MM_update"] == "form1")) {
$updateSQL = sprintf("UPDATE tblproducto SET strNombre=%s, strSEO=%s, dblPrecio=%s, intEstado=%s, intCategoria=%s,
strImagen=%s, intStock=%s, intOferta=%s WHERE idProducto=%s",
GetSQLValueString($_POST['strNombre'], "text"),
GetSQLValueString($_POST['strSEO'], "text"),
GetSQLValueString($_POST['dblPrecio'], "double"),
GetSQLValueString($_POST['intEstado'], "int"),
GetSQLValueString($_POST['intCategoria'], "int"),
GetSQLValueString($_POST['strImagen'], "text"),
GetSQLValueString($_POST['intStock'], "int"),
GetSQLValueString($_POST['intOferta'], "int"),
GetSQLValueString($_POST['idProducto'], "int"));

mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$result1 = mysql_query($updateSQL, $conexionzapatos) or die(mysql_error());

$updateGoTo = "productos_lista.php";
```

Una vez tenemos esto ya podemos hacer que nos salgan las ofertas en la pagina principal **index.php**

Si miramos un poco seguro que alguna pagina que ya tenemos hecha nos puede valer como plantilla, en efecto la pagina **categoría\_ver.php** nos sirve, nos podemos aprovechar de la consulta cambiando los parámetros.

El campo que nos indica que es una oferta es **intOferta** con valor 1 es una oferta, con esto podemos implementar una infinidad de opciones para la pagina, que el producto este disponible o no disponible, bajo pedido o con diferentes estados

```
mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_DatosProductos = "SELECT * FROM tblproducto WHERE tblproducto.intOferta = 1";
$DatosProductos = mysql_query($query_DatosProductos, $conexionzapatos) or die(mysql_error());
$row_DatosProductos = mysql_fetch_assoc($DatosProductos);
$totalRows_DatosProductos = mysql_num_rows($DatosProductos);
```

Usamos también toda la etiqueta <div> que nos mostraba el producto, lo único que descartaremos es la paginación ya que nos interesa mostrar todo en la misma pagina

```

<div class="resultadoproductos">
    <?php if ($totalRows_DatosProductos > 0) { // Show if recordset not empty ?>
        <?php do { ?>
            <div class="producto">
                <div class="fotoproducto"></div>
                <div class="productoderecha">
                    <div class="textopropucto"><a href="ver_producto.php?recordID=<?php echo $row_DatosProductos['idProducto']; ?>">
                        <?php echo $row_DatosProductos['strNombre']; ?></a>.
                    Stock: <?php echo $row_DatosProductos['intStock']; ?></div>
                    <div class="textopropuctoprecio">PVP: <?php echo $row_DatosProductos['dblPrecio']; ?>&euro;</div>
                </div>
            </div>
        <?php } while ($row_DatosProductos = mysql_fetch_assoc($DatosProductos)); ?>
        <?php } // Show if recordset not empty ?>
        <?php if ($totalRows_DatosProductos == 0) { // Show if recordset empty ?>
            Todavia no hay productos de esta categoria.
        <?php } // Show if recordset empty ?>
    </div>

```

Con esto ya nos mostraría los productos en oferta

## Nuestros productos de Oferta



Bota de Agua.  
Stock: 89  
**PVP: 20€**

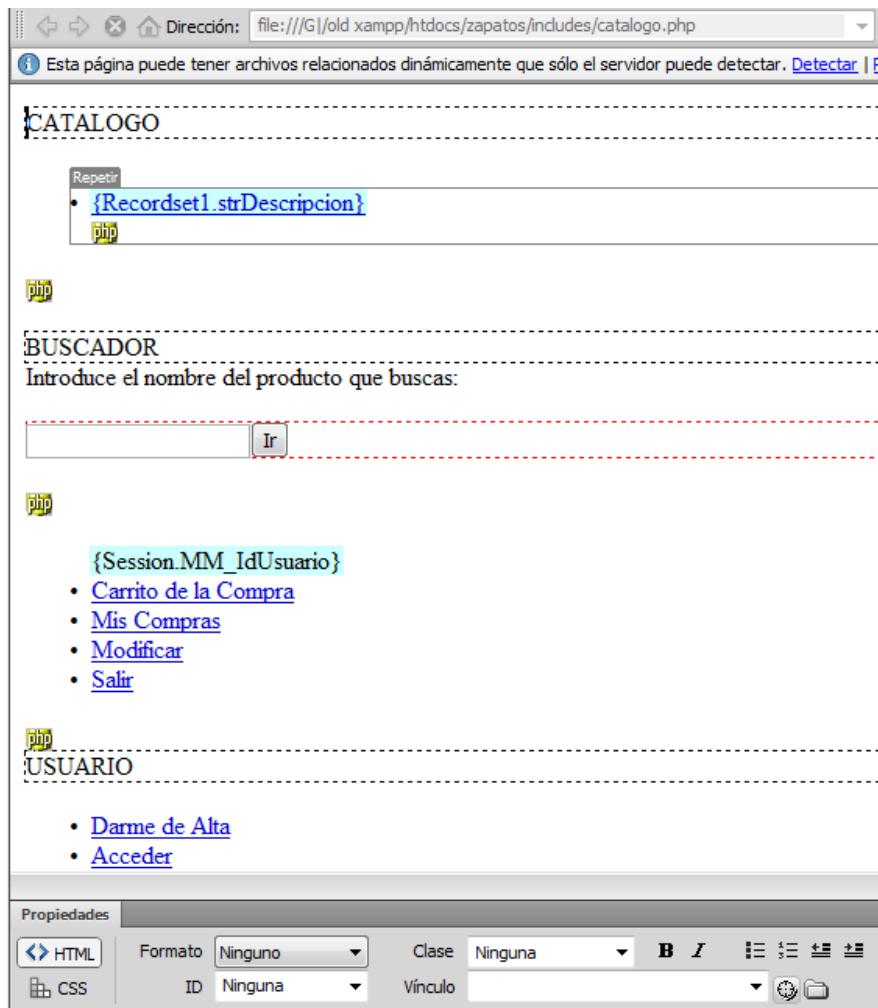


Tacon Economico.  
Stock: 24  
**PVP: 34€**

## 7.6 BUSCADOR DE PRODUCTOS.

### categoría\_resultados.php

En **catalogo.php** añadimos un formulario de texto que es lo normal para un buscador de productos



El botón enviar sera tipo submit, con method **GET** y mediante el action señalamos donde queremos ir cuando hagamos clic, en nuestro caso a **categoría\_resultados.php**, que crearemos ahora, y por ultimo el input name **Fbuscar**(añado la F para conocer que se trata de un Formulario).

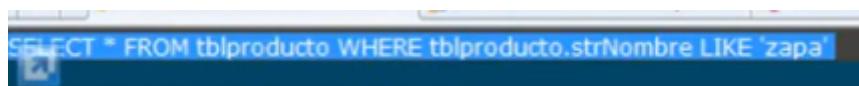
La pagina **categoría\_resultados.php** sera algo similar a cuando estamos en una categoría y nos muestra los productos, podemos aprovechar de **categoría\_ver.php**



Creamos una nueva consulta que nos devuelva valores sobre el nombre del producto, pero no iba a ser tan simple existe una pequeña complicación es que como hacemos que nos haga la SELECT pero seleccionado en el campo WHERE lo que introducimos en el formulario de buscar mediante LIKE %s obtenemos el valor que le pasamos por el buscador .

Con esto no ha funcionado así que vamos a sacar por pantalla la consulta que le esta llegando

Si buscamos zapa obtenemos:



Entonces hay una forma de indicarle a mysql la palabra que estamos buscando contenida en el nombre si añadimos a %s → % %s% tampoco funciona

El problema radica en el sprintf que predecía la consulta y el valor al que asociábamos la cláusula LIKE

```
$varCategoria_DatosProductos = "0";
if (isset($_GET["FBuscar"])) {
    $varCategoria_DatosProductos = $_GET["FBuscar"];
}
mysql_select_db($databaseConexionzapatos, $conexionzapatos);
$query_DatosProductos = "SELECT * FROM tblproducto WHERE tblproducto.strNombre LIKE '%".$varCategoria_DatosProductos."%' ";
//echo $query_DatosProductos;
$query_limit_DatosProductos = sprintf("%s LIMIT %d, %d", $query_DatosProductos, $startRow_DatosProductos, $maxRows_DatosProductos);
$DatosProductos = mysql_query($query_limit_DatosProductos, $conexionzapatos) or die(mysql_error());
$row_DatosProductos = mysql_fetch_assoc($DatosProductos);
$varCategoria_DatosProductos = "
```

Concluimos el proyecto en la parte de programación de PHP con esta parte, se que es muy hay partes muy mejorables ya que podríamos de añadir mas elementos a la tienda, a mi parecer con lo que hemos realizado tenemos una tienda funcional .

## 7.8 PAGINA DE CONTACTO

### contacto\_formulario.php

Como toda pagina web que oferta servicios, necesitamos una pagina de contacto en la cual los usuarios o clientes puedan contactar con la tienda en cuestión.

En el pie de pagina vamos añadir un enlace [Contacto TiendaZapatos.com](#), al pulsar sobre el nos redirecciona a **contacto\_formulario.php**

## Contacto

Escríba o comuníqueme aquello que necesita, y me pondré en marcha para enviarle una respuesta, una consulta o un presupuesto, en el menor tiempo posible.

Si prefiere utilizar otro medio, puede hacerlo a través de las siguientes alternativas:

Francisco Marcet Prieto.

Teléfono: 669.126.284

Domicilio: Avda/ País Valenciano 35, (Villajoyosa, España)

e-mail: [info@tiendazapatos.com](mailto:info@tiendazapatos.com)

## Formulario de contacto

Nombre:  \*

Telefono:  \*

Email:  \*

Mensaje:

Añadimos también la validación de campos en el formulario.

Al pulsar Enviar nos redirecciona a **contacto\_formulario\_ok.php**, la nos informara que nuestra consulta se ha enviado correctamente



## Tu consulta se envio correctamente.

En breves recibira una respuesta de su consulta.

[Volver a inicio](#)

TiendaZapatos.com® info@tiendazapatos.com Tel:+34-666.66.66.66

Avda País Valencià nº35 Villajoyosa(Alicante)

## 8. CONCLUSIONES

En este punto del Proyecto, se me pasan muchas conclusiones por la cabeza. La finalización de este proyecto me aporta una gran satisfacción ya que, hace un par de meses, cuando este Proyecto solo era una idea en mi cabeza, parecía como algo inalcanzable y difícil de conseguir al mismo tiempo.

Con esta aplicación he aprendido a enfrentarme a un trabajo que podría considerarse como una tarea real en cualquier empresa. He sido capaz de utilizar parte de los conocimientos adquiridos durante este curso y plasmarlos en un trabajo del que me siento orgulloso de cómo ha quedado. Un trabajo que no dista mucho de la idea que en un principio tenía en mente.

Además me ha servido para darme cuenta de que tareas triviales que a simple vista parecían sencillas, se han convertido en un quebradero de cabeza y tareas que parecían imposibles han sido mucho más fáciles de llegar a cabo. También decir, que a medida que el Proyecto iba tomando forma, han surgido muchas dudas que al resolverlas me servían de ejemplo para aprender nuevas cosas dentro de la programación Web.

Pese a estar finalizado el proyecto, la motivación que iba naciendo en mi paso a paso de ir descubriendo nuevas posibilidades me hará seguir investigando en nuevas opciones a añadir dentro de la Tienda Virtual.

Con esta memoria finaliza mi Proyecto y también este año de curso en Sistemas Gestores de Bases de Datos.

## **9. AGRADECIMIENTOS.**

En primer lugar, quiero agradecer a mi profesora, Virginia Checa Galpeno, que, sin su ayuda, no habría sido posible la realización de este trabajo. En concreto, agradecer todas las tutorías hechas, los e-mails enviados, que haya resuelto la cantidad de dudas que me han ido surgiendo a lo largo del trabajo, y por las correcciones hechas. Con todo ello, ha hecho posible que la tarea de realizar este trabajo haya sido más fácil y llevadera, dentro de las dificultades y el tiempo que conlleva.

A mis padres, por darme la vida y apoyarme en todo lo que me he propuesto.

A mi padre, por ser el apoyo más grande durante mi educación universitaria, ya que sin él no hubiera logrado mis metas y sueños. Por ser mí ejemplo a seguir, por enseñarme a seguir aprendiendo todos los días sin importar las circunstancias y el tiempo. Y que siempre el esfuerzo y la constancia da sus frutos.

A mi madre, la cual te agradezco el estar siempre conmigo, en mi mente, mi corazón y acciones. Tu eres parte de esto.

A mis abuelas Ola y Victoria, a mi abuelo Germán, por brindarme su cariño, su vida y por apoyarme siempre, no importando que tan lejos estén.

A mis amigos, por ser parte de mi vida, de mis momentos tristes y alegres, por apoyarme, por nunca dejarme caer, por estar siempre ahí.

# **10. BIBLIOGRAFÍA.**

Referencias Web:

- IEEE Recommended Practice for Software Requirements Specifications, IEEE Std 830-1998.
- Foros del Web: <http://www.forosdelweb.com>
- Wikipedia la enciclopedia libre: <http://es.wikipedia.org>
- Manual de PHP: <http://php.net/index.php>
- Especificaciones de XHTML 1.0 <http://www.w3c.org/TR/xhtml1>
- Especificaciones de CSS 2.1 <http://www.w3c.org/TR/CSS21>
- Xampp: Windows, Apache, MySQL, PHP: <http://www.xampp.info>