CARRITO DE LA COMPRA EN PHP VER. 2.0

CONSIDERACIONES PREVIAS
(el proyecto se entregará en formato zip y será un archivo cuyo nombre estará fomado por vuestros apellidos y nombre y 'proyecto1T') ej:GarciaBlascoJavierProyecto1T)

- En todo el proyecto será obligatorio respetar la estructura del **MODELO VISTA CONTROLADOR.**
- Se debe de respetar la estructura de carpetas vista en la unidad 6
- En el cliente, **se almacenará solo el HASH de la contraseña.**
- En esta ocasión el carrito se almacenará en una tabla nueva, dentro de la base de datos.

SERVICIOS

- Se implementará un servicio para obtener los **productos** de la base de datos, por tanto, solo tendrá contemplado el método GET. De manera que si se le pasa un idProducto devuelve ese producto y sino devolverá todos los productos. Siempre que necesitemos algún producto se usará el servicio.
- Un servicio para validación y registro de **clientes**, que se usará en DWC.**(LOS ALUMNOS QUE SOLO TIENEN DWS LO HARÁN EN PHP)**
- Otro servicio para la gestión total del **carrito** (también la haréis en el lado cliente)

El intercambio de información será en formato json. **(LOS ALUMNOS QUE SOLO TIENEN DWS LO HARÁN EN PHP SIN SERVICIOS)**
Los servicios se crearán cada uno en una carpeta dentro de la carpeta principal del proyecto, y dentro de esta carpeta tendrán la misma estructura de carpetas que cualquier aplicación (con las carpetas clases, config, controladores ...).

MODELO
Crearemos todo el modelo en una carpeta llamada **clases** de manera que cada definición de clase será un archivo y se utilizará *autocarga* de clases (según hemos visto en clase)
Para el servicio las clases se pondrán en su propia carpeta y se utilizará también la autocarga de clases.
En esta ocasión **almacenaremos el carrito en la base de datos** (crearéis la tabla con estructura libre) por lo tanto hay que tener esta tabla en cuenta para el modelo
El diseño de este modelo es libre, pero ha de cumplir los siguientes requisitos:

- Las propiedades han de ser **privadas**
- Si es necesario se crearan **set** y **get** para acceder a algunas de ellas, utilizando los métodos mágicos.
- Todos aquellos métodos que sean necesarios para relacionarse con la base de datos, recibirán **la conexión como parámetro** y si necesitan algún valor para un campo **se obtendrán de las propiedades** que antes se habrán cargado con ellos. Nunca estos datos se pasarán como parámetros.
- Todos los métodos que accedan a más de un registro (del tipo *getAll*) serán **estáticos**.

CONTROLADORES
Además del controlador frontal colocado en la carpeta principal, serán obligatorios los siguientes controladores.

1.- **validar.php**.
(toda la lógica de acceso a datos ser realizará en el lado del cliente usando el servicio creado para

este caso)
Solo será llamado este controlador cuando el usuario quiera, pulsando el botón validar que aparecerá en la pantalla principal. Si el usuario no se valida, se le pedirá obligatoriamente cuando quiera comprar, para obtener sus datos de envío.
Si es correcto, se crearán 2 sesiones, una con el **nombre** y otra con el **dni.** Se comprobará si existe una variable de sesión **idUnico** y si es así se buscará su valor en la tabla carrito y para todos los registros que existan con ese **idUnico** se le añadirá el dni del que se valida.y se saltará a *principal.php*. (se puede usar header).
El **nombre** se mostrará en todas las páginas php del trabajo y el dni se utilizará para poder agilizar las tareas de recuperar sus datos para dar de alta el pedido.

2.- **principal.php**. Esté será el primer controlador a ejecutar. En el se mostrarán, todos los productos con la imagen de cada producto y bajo de la foto su nombre , su precio y un vínculo para mostrar "detalle.php"
En la parte superior de la pantalla se pondrá un vinculo con un dibujo de un carrito y al pulsarlo se verá el estado actual del carrito (vercarrito.php). Debajo de este carrito aparecerá el total de registros que haya en la tabla **carrito .**
En cualquier parte de la pantalla se pondrá un icono para *validarse* (ir a validar) o *registrarse* (el controlador registrarse.php)

3.- **Detalle.php**, se mostrarán todos los datos del producto incluida la foto. Además tendrá un cuadro de texto con la cantidad a comprar (inicialmente valdrá 1) y el botón comprar.
Realmente será un formulario que al enviarse (pulsar botón comprar) llamará a vercarrito.php, pasándole como parámetros el id del producto, el nombre, el precio y la cantidad.

4.- **Vercarrito.php.**
(toda la lógica de acceso a datos ser realizará en el lado del cliente usando el servicio creado para este caso)
A este archivo se puede llegar desde dos sitios, desde el icono del carrito de la página principal, o desde detalle.php (al comprar un producto, se sabrá porque se habrá pulsado el botón de comprar).

a.- Si se llega desde la página principal, simplemente se muestra el contenido del carrito, listando los productos, su cantidad, su precio y calculando su importe. Al final se sumará el total de importes y también se mostrará.

b.- Si se llega después de comprar:

- Primero se comprobará si existe una variable de sesión llamada **idUnico**. Si no existe se crea con la función vista para generar ids únicos.
- Se añadirá la linea del carrito a la base de datos, incluyendo el **idUnico** y **dniCliente** (si no se ha validado se incluye pero vacio)

Siempre que se muestra el carrito la cantidad aparece como un cuadro de texto y añadimos un botón llamado **actualizar**. De esta manera si modificamos alguna cantidad con el botón actualizar se modifica el registro de la tabla correspondiente. Ojo un solo botón actualizar para todo el carrito.(se puede utilizar un *array de controles*: básicamente cada elemento del formulario cantidad se llamará cantidad[] y de esta manera se crea un array de cantidades según el orden)
También se añadirá un vínculo "borrar" ,en cada linea, para eliminar ese producto de la tabla **carrito,** en la base de datos.
Al final del carrito se mostrará el botón de *confirmar pedido* (confirmar.php) y el botón de *seguir comprando* (este último enviará a la página principal).

5.- **Confirmar.php**.
Lo primero es comprobar si el usuario está validado, si no es así se le obliga a validarse o registrarse, esto es necesario para poder obtener la dirección de envío.
Una vez superado este paso se le informa de que la forma de pago es por transferencia y se le muestra los datos para hacer la transferencias
A continuación recogemos todos los datos del carrito almacenados en la base de datos, así como el dni del cliente almacenado en la sesión y con esto damos de alta el pedido en la tabla **pedidos** y en la tabla **lineaspedidos**.

- Alta en **pedidos**
  idpedido: Calcularemos el máximo de los valores que tiene en la tabla *pedidos* el campo *idPedido* y le sumaremos uno.
  fecha: La de hoy.
  Dnicliente : El dni de la variable de sesión
  dirEntrega: La dirección del cliente.
  El resto de campos vacios

Alta en **lineaspedidos**:
Idpedido es el que hemos calculado antes,
nlinea es un contador desde 1 hasta el número de productos a añadir para ese pedido.
El resto de campos los sacamos del carrito.
En esta tabla tendremos que dar de alta tantos registros como productos haya en el carrito.

Se creará un string que contendrá las sentencias en html necesarias para visualizar el pedido completo, y al final se mostrará por pantalla.
Se borrarán todas las sesiones y todos los registros del carrito.

6.- **registrarse.php**.
(toda la lógica de acceso a datos ser realizará en el lado del cliente usando el servicio creado para este caso)
Es un controlador para crear un nuevo cliente. Al crearlo se añaden también las sesiones. Y salta a principal.php.

INVESTIGACIÓN:
El pedido que hemos generado antes en un string se convertirá a pdf

NOTA

- Cuando un usuario se valida se tienen que tener en cuenta si tiene un carrito ya creado para asignárselo.
- Se tiene que permitir que todo el mundo que acceda a la tienda pueda crear un carrito aunque no esté validado.