

Práctica Final

2 DAW



Restaurante TPV

Pedro Sandoval Martínez

Índice

[Primer Sprint 2](#_Toc183084906)

[Diseño y creación de la base de datos 2](#_Toc183084907)

[Esquema de Archivos 3](#_Toc183084908)

[Inicio de sesión y ruta de acceso 3](#_Toc183084909)

[Menús de opciones 4](#_Toc183084910)

[Spring 2 5](#_Toc183084911)

[Gestión y visualización de mesas 5](#_Toc183084912)

[Creación de pedidos 6](#_Toc183084913)

[EXPLICAMOS LO QUE QUERIAMOS HACER CON JAVASCRIPT, el carrito. 6](#_Toc183084914)

[Abrir mesas activas 7](#_Toc183084915)

[Spring 3 7](#_Toc183084916)

[Conexión con la impresora 7](#_Toc183084917)

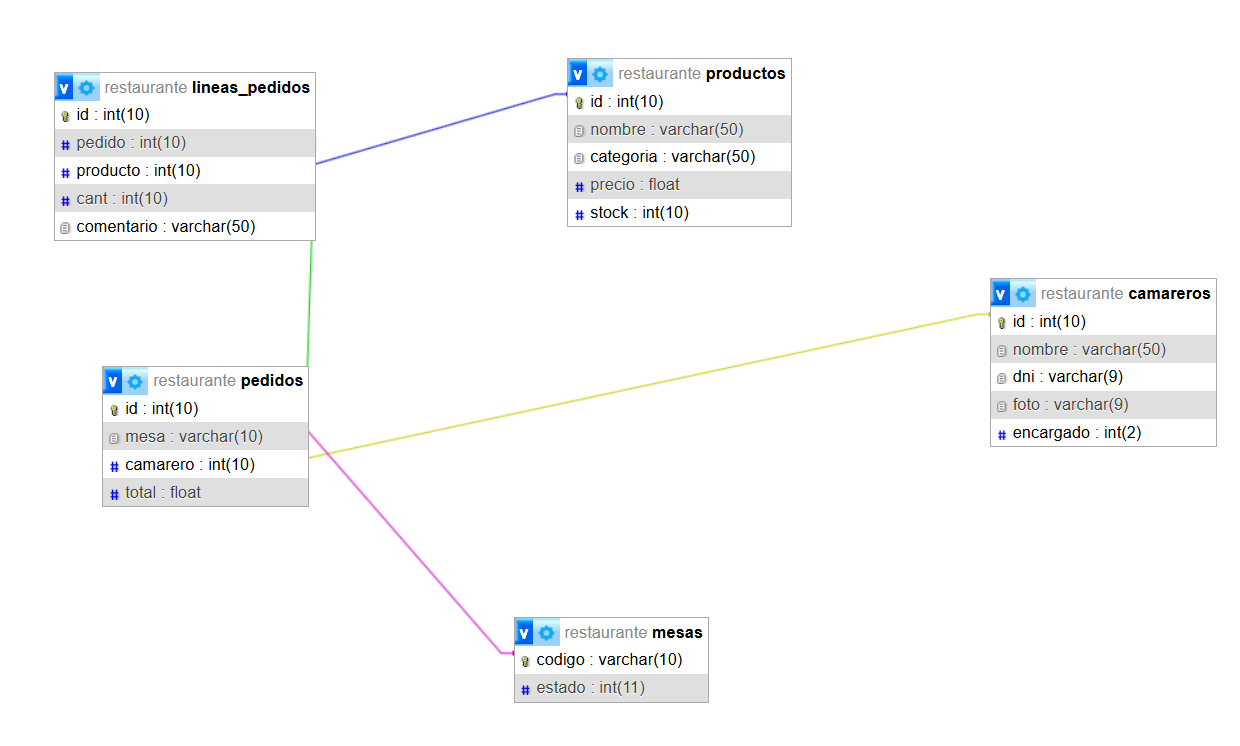
[Cuentas (tickets) 8](#_Toc183084918)

[Cerrar mesas (pagar pedido) 8](#_Toc183084919)

# Primer Sprint

## Diseño y creación de la base de datos

Primer diseño de la base de datos. Centrada sobre todo en la tabla camareros, en la cual he comenzado a trabajar. En principio el id del camarero estará en el pedido para conocer quién registra el mismo.



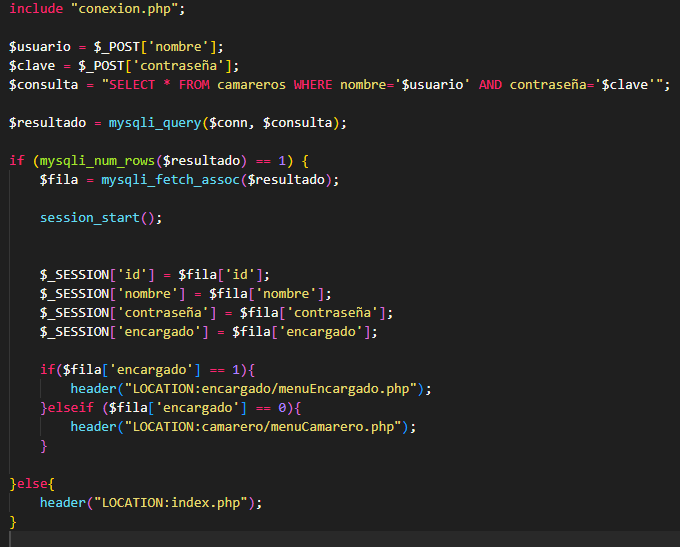
## Esquema de Archivos

De momento he separado la visión de camarero y encargado.

Los archivos principales que conectan con la base de datos se encuentran en la carpeta raíz, al igual que los estilos del index (inicio de sesión) y el archivo que utilizo para validar la sesión.

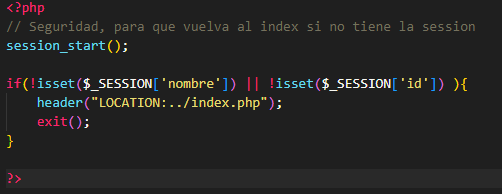
## Inicio de sesión y ruta de acceso

Previamente insertado el primer camarero (admin), comprobamos que el usuario que intenta acceder se encuentre en la base de datos.



Si el usuario, es encargado (atributo encargado = 1), la página es redirigida al apartado de encargado. En cambio, si el atributo es 0, te lleva a la vista del camarero.

La seguridad del sitio web es comprobada por este archivo, que comprueba en cada una de las páginas la variable de sesión:



## Menús de opciones

En la vista de **camarero**, únicamente encontramos la opción de entrar al salón donde visualizamos las mesas, y dentro de ellas la toma de los pedidos.

El menú de **encargado** lo presentamos inicialmente así. Entre sus opciones encontramos la modalidad de añadir camareros, el listado de los mismos (sus datos), añadir nuevos productos y visualizarlos.



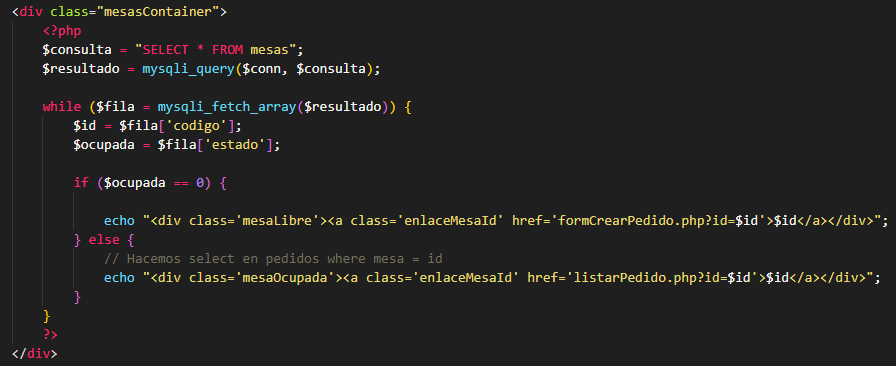
Además, introduzco un enlace para acceder a la vista de camareros por si el encargado lo necesitase.

# Spring 2

## Gestión y visualización de mesas

Breve explicación: las mesas las utilizamos como contenedor del pedido, para localizarlo en el salón y poder acceder a las opciones (crear pedido y listarlo).

Archivo salón.php :



Mostramos las mesas que se encuentren en la base de datos, las que tengan estado 1 (ocupada) se mostrarán con una clase distinta, para que cambie el diseño y sea más fácil localizarlas en el salón.

## Creación de pedidos

### Intento fallido con Javascript

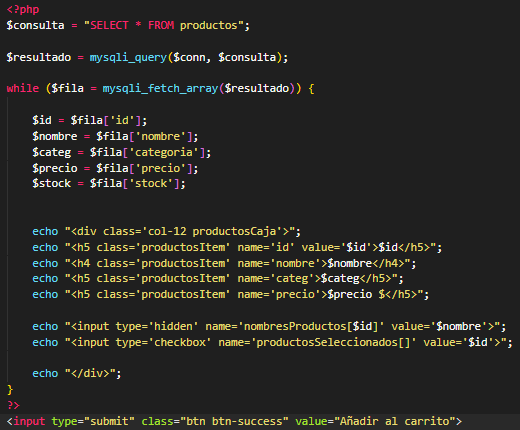
Aquí surge mi primer problema. La idea inicial era implementar parte de código javascript para hacer una “cesta” dinámica, sin recargar la página, puesto que sería un borrador antes de hacer la inserción del pedido.

* Ventajas: no es necesario guardar la información en la base de datos, mejoraría el rendimiento en un programa pesado.
* Problemática: la consulta de los productos almacenados debe ser transformada en un json, mostrarla entre etiquetas php, y transformarla una vez más para mostrarla en el carrito (y posteriormente debería ser tratada en formato php para la inserción).

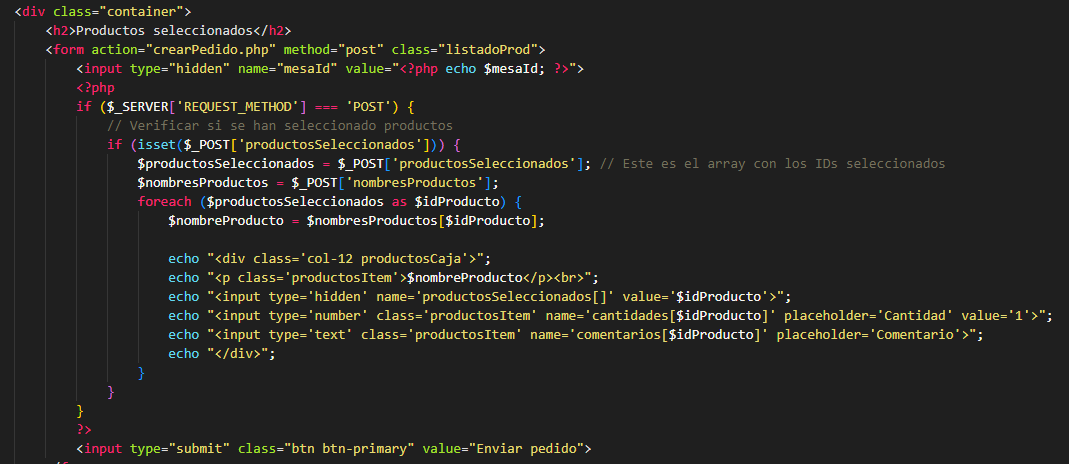
Por lo tanto, descartamos la idea de usar Javascript.

### Funcionalidad final

Una vez que entramos a la mesa, mostramos en un formulario la consulta de los productos. Los artículos que seleccionemos pasarán a otro formulario (líneas\_carrito).



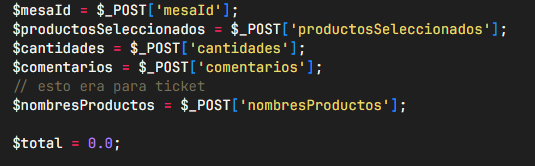
En el mismo archivo (formCrearPedido), hay otro formulario que usamos a modo de carrito, donde nos llevamos el array de productos seleccionados (con el checkbox).



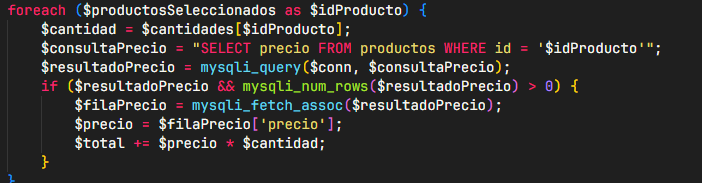
Este formulario es el que lleva a crearPedido, donde finalmente se hace la inserción en líneas\_pedidos. Para ello he tenido que llevarme el id del pedido, de la mesa y de los artículos seleccionados. Todo ello en forma de array con cada uno de los elementos ordenados por su id.

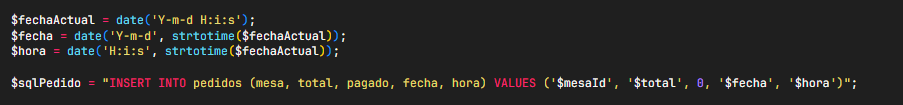
### Crear Pedido

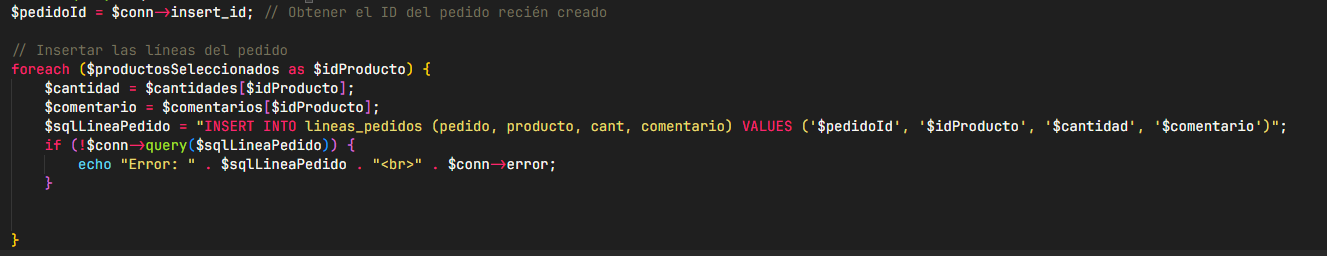
La cabecera del archivo contiene las variables más importantes con las que trabajamos:



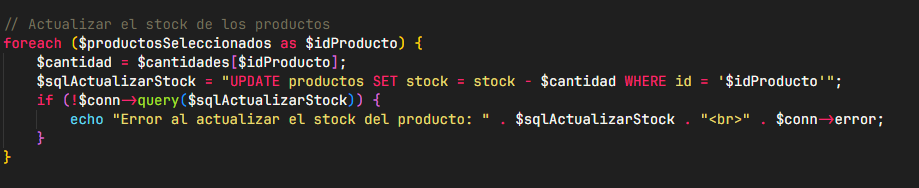
Calculamos total del pedido para su posterior inserción:



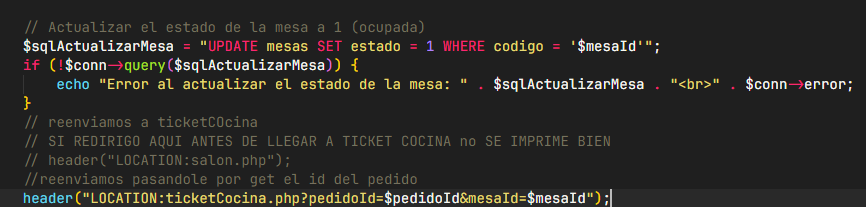
Guardamos fecha de creación y realizamos la inserción del pedido:

Y ahora sí, procedemos a la **inserción de las líneas del pedido**:

En este punto actualizo el stock de los productos, que nos será útil en el futuro para la gestión de productos, y que el camarero sepa a tiempo real si dispone de dicho artículo.

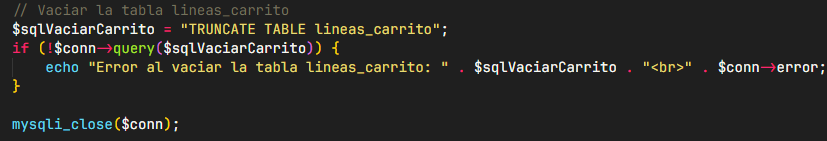


Y actualizamos el estado de la mesa



Aquí ya vemos anotaciones de lo que se avecinaba en la siguiente sección, que explicaremos a continuación.

Lo último que modifica este archivo es la tabla del carrito, que debe ser vaciada una vez realizado el pedido:



## Abrir mesas activas (listar pedido)

### Anotación Importante

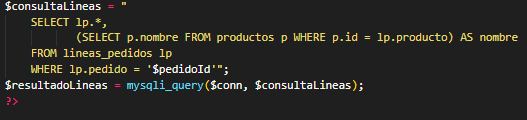
En todos los archivos relacionados con los pedidos, debemos comprobar que realmente nos llevamos la información adecuada, y trabajamos sobre la mesa y pedido que deseamos. Para ello en listar pedido encontramos estas líneas:



Así fui descartando errores y optimizando el código.

### Mesa ocupada

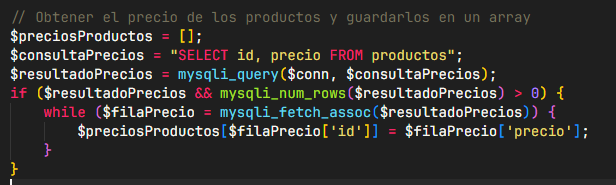
Una vez que la mesa aparece ocupada, si accedemos a ella desde el salón nos llevará a la página donde se listan las líneas del pedido que hay alojado en el mismo. Para ello he creado una consulta con producto cartesiano utilizando el pedidoId de arriba:

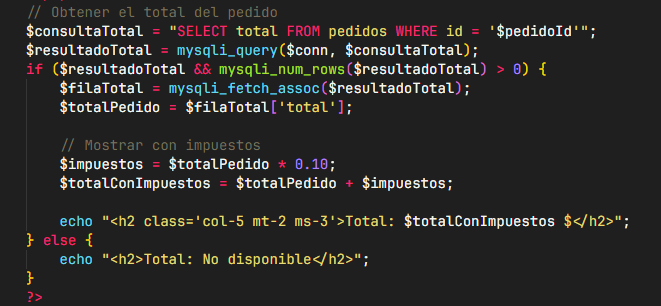


El producto cartesiano se utiliza para mostrar el listado de los productos en forma de tabla, a la que después según su categoría se le otorga un estilo distinto:



Para mostrar el precio del producto e insertar el total de la cuenta en el pedido, genero la siguiente consulta:



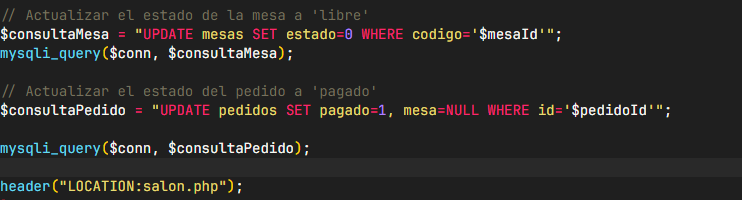


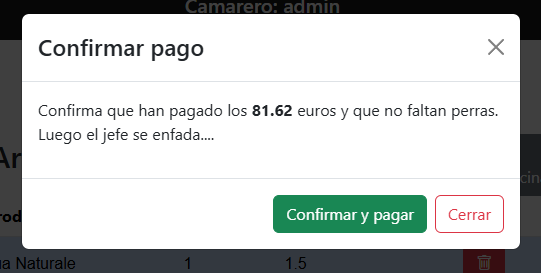
Lo utilizaremos para mostrar al final del listado el total de la cuenta, junto a los botones para imprimir el ticket y pagar la cuenta (y liberar la mesa).

Ambos tickets los pospuse para el siguiente spring, puesto que no disponía de la impresora, lo veremos en el siguiente apartado.

## Cerrar mesa (pagar pedido)

Añado en este apartado la funcionalidad de liberar la mesa. La lógica es simple. Mi columna “mesa” admite nulos, así que cuando un pedido está pagado, este pedido asociado a una mesa deja de estarlo, y el atributo pasa a ser NULL.



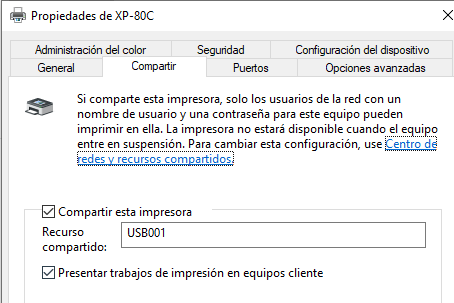
Cuando pulsamos el botón se abre un modal para confirmar la operación, y se ejecuta el código de arriba.

# Spring 3

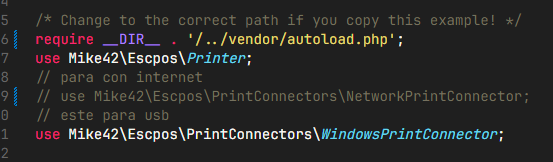
## Conexión con la impresora

Inicialmente hice las pruebas conectando la impresora (modelo XP-80C) por usb y posteriormente instalando los drivers pertinentes, en modo local con los propios archivos que nos facilitó el profesor D. Fernando Ureña.

El nombre que le otorgues es irrelevante, sin embargo, un paso indispensable es la configuración del puerto al que está conectada. En Windows (preferencias de impresora) tuve que activar el uso compartido de la misma, asignándole el puerto en el que será reconocida por nuestro servidor y por tanto en nuestros archivos PHP.



Por supuesto otro requisito indispensable es asegurarnos que el composer y el vendor previamente instalados en el repositorio están enlazados con el archivo donde creamos el ticket, que no haya errores en la ruta.

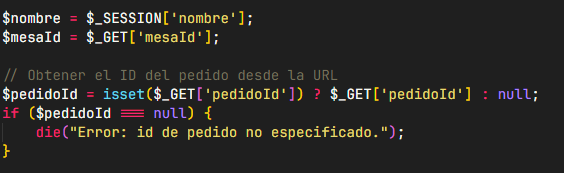


Esto incluye en nuestro archivo las librerías que necesitamos.

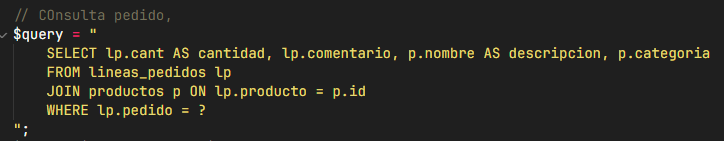
## Tickets

### Para cocina

En el apartado donde creamos el pedido, ya vemos cómo se enlaza con este archivo, que tras la configuración inicial, recibe por get el id de la mesa donde se encuentra el pedido:

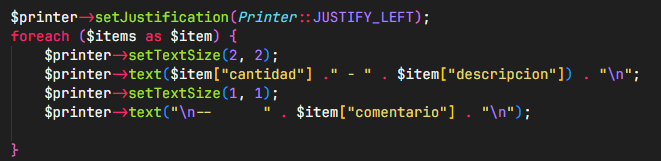


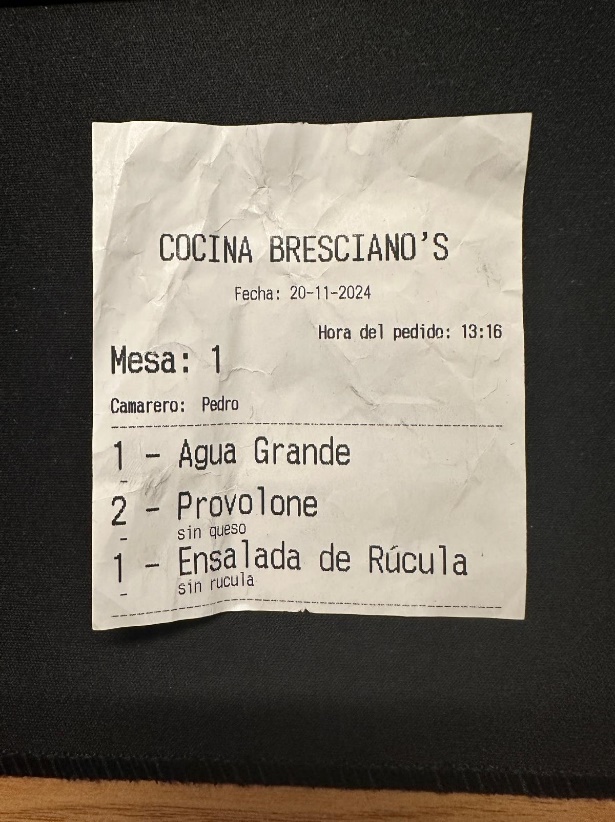
Hacemos la consulta utilizando join:



Ejecutamos la consulta y creamos array de productos (ítems) que deben imprimirse, teniendo en cuenta su descripción, cantidad y el comentario (si hay)

Tras la cabecera del ticket donde mostramos fecha, hora mesa y nombre del camarero, imprimimos array con saltos de línea para cada elemento:



Obteniendo como resultado este ticket:

Tras imprimirlo, nos devolverá al salón donde podremos comprobar y volver a reimprimir el ticket de cocina si lo necesitamos.

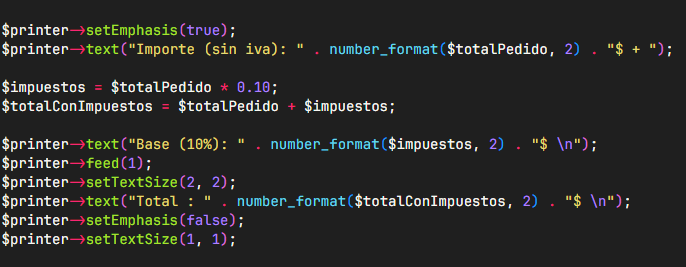
# 

### Para cliente

Utilizamos la misma configuración, las mismas variables pero trabajamos con distintos datos. En el ticket del cliente no necesitamos los comentarios de las líneas del pedido.

En cambio, vamos a necesitar llevarnos del formulario los precios del producto y el total del precio para trabajar sobre él y conseguir el importe total con IVA.

Imprimimos de la misma forma pero lo mostramos con otro formato. Los cálculos finales son los siguientes :



Y el resultado es el siguiente:



Y con esto finalizamos el apartado de los tickets

## Auditoría de fechas

Añado columna fecha para el pedido, y hacer búsqueda de pedidos de ese día. Esta opción ya está añadida en las capturas donde creamos el pedido, pero fue de los últimos cambios que realicé en el código.

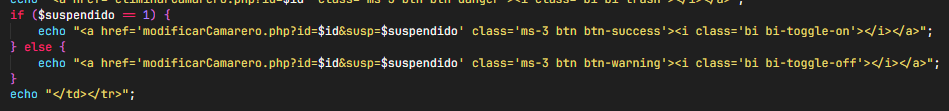


Camareros

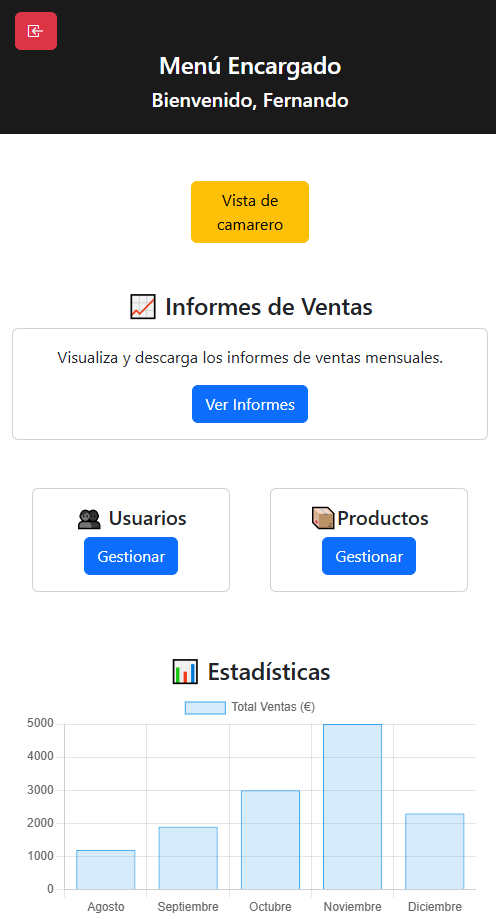
Por defecto, suspendido es igual a 1, no están activos por defecto.



Así mostramos los botones para administrar el estado de los camareros



**Spring 4: Funciones administrativas**



Para la vista del encargado he desarrollado el siguiente menú.

En el primer apartado podemos acceder a la sección de los camareros, para realizar labores relacionadas con los pedidos actuales (poder modificarlos y eliminarlos).

En informe de ventas podremos seleccionar la fecha en la que se realizaron pedidos, y nos generará una factura detalla con información sobre ese día

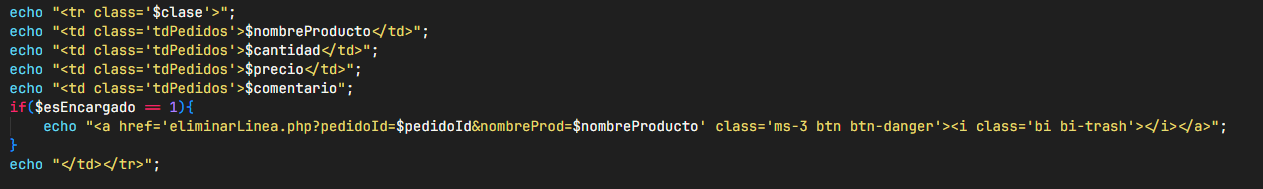
Se despliegan distintos modales donde podremos dirigirnos al apartado donde se muestran los usuarios y su gestión, y otro con un formulario para añadir nuevos.

Además de incluir el apartado de gestión de productos donde modificamos el stock, añadimos y eliminamos productos.

## Gestionar pedidos:

Cuando accedemos a la vista del camarero y posteriormente a una mesa con el pedido activo, el archivo comprueba que realmente es encargado.

## 

Esta opción nos permite generar junto al producto, un botón que elimine el mismo del pedido, accediendo por el enlace a la consulta donde se genera la modificación:

El código de eliminar línea va a utilizar esos dos parámetros para eliminar el producto. Es similar a los archivos que utilizo para eliminar camareros y productos, los cuales muestro abajo.

**Gestionar productos:**

En este apartado encontramos un formulario para insertar nuevos productos, según su categoría y el resto de parámetros necesarios (precio, stock).  La inserción en la base de datos requiere rellenar todos los campos.



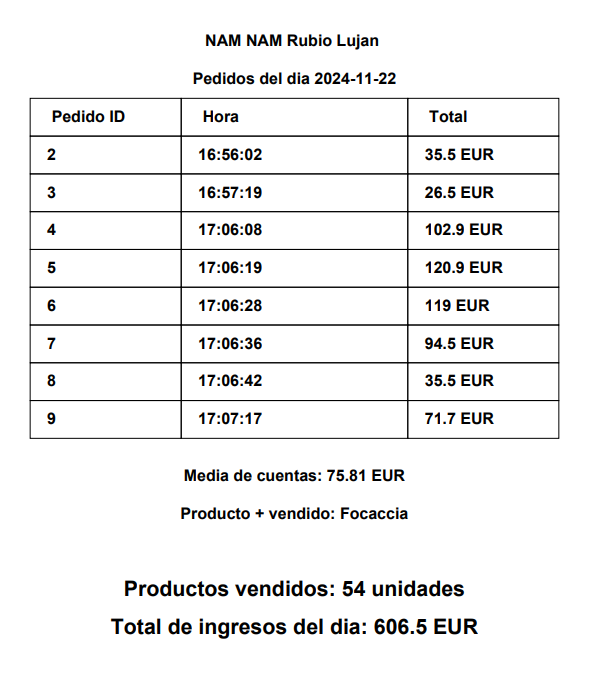
**Gestionar camareros**

### Agregar camareros, suspender o eliminar.

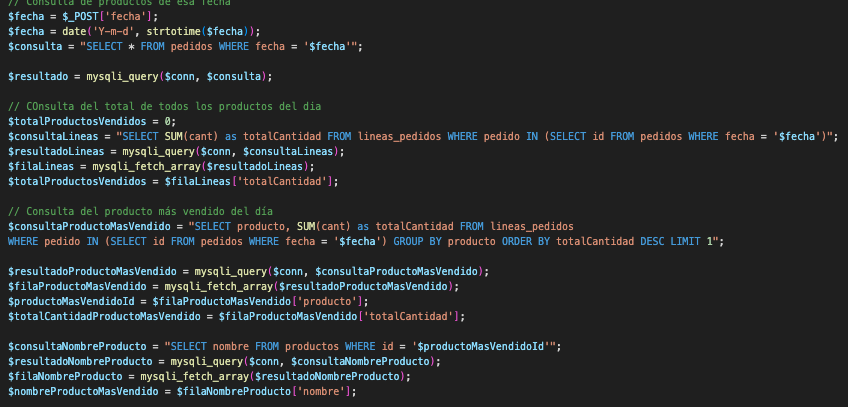
**Informes de ventas e historial de pedidos**

Volvemos al menú. Al pulsar en “ver informes” nos saltará un formulario donde podemos insertar el día del informe que estamos buscando. Automáticamente nos redirige al pdf generado, tomando por atributo indispensable la fecha elegida.

**INSERTAR CAPTURA CON media, total productos y eso.**



**Consultas a la base de datos que requiere esta funcionalidad:**

****