

CantinAPP



Una aplicación web para
gestión de comandas

INDICE

Contenido

INDICE.....	2
INTRODUCCIÓN	3
HERRAMIENTAS	4
DESCRIPCIÓN DEL SISTEMA.....	5
MEMORIA DEL PROYECTO	9
MEJORAS DEL SISTEMA.....	11
CONCLUSIÓN	12
REFERENCIAS	13

INTRODUCCIÓN

Este proyecto se enmarca en la necesidad de manejar los pedidos que un restaurante corporativo recibe de sus empleados, así como la información que facilita los cobros del servicio.

Se trata de una empresa con aproximadamente 450 trabajadores, de los cuales, la mayor parte hace uso diario o muy habitual del servicio. Como mínimo, 1/4 parte de la plantilla de la empresa hace uso diario del servicio (datos oficiales), así como también clientes ajenos a la empresa, aunque éstos últimos no estarán contemplados en este sistema.

Actualmente, tanto la gestión de los pedidos, como el cobro del servicio, es gestionado de manera completamente manual e independiente para cada trabajador o departamento, existiendo así departamentos muy organizados y que facilitan el servicio, y otros que prefieren su gestión individual. Es por esto, que queda justificado el desarrollo de este sistema, y enmarcado en el Módulo de Proyecto del ciclo formativo de grado superior, Desarrollo de Aplicaciones Web.

Los pedidos, normalmente, se reúnen y gestionan por grupos (departamentos), aunque también se puede tratar de pedidos individuales, como se ha comentado.

HERRAMIENTAS

- Se instala y configura un sistema LAMP para soportar la construcción del proyecto, compuesto por PHP, Apache, MySQL, y Ubuntu como sistema base.
- Se utiliza el editor de código Visual Studio Code para el desarrollo principal.
- HeidiSQL es el cliente de BBDD elegido desde el que manejar la creación y modificación del esquema de datos.
- Postman se utiliza para la prueba de peticiones HTTP contra el servidor. Es un software que permite el envío de peticiones HTTP REST sin necesidad de desarrollar un cliente.
- También utilizo Git para el control de versiones, mediante GitHub. Integrado en VSCode, simplifica el control de versiones desde el editor de código.

DESCRIPCIÓN DEL SISTEMA

El sistema debe ser capaz de permitir a los usuarios (empleados) el manejo de sus pedidos, desde la creación de la manera más sencilla posible, hasta la modificación de la composición de sus pedidos.

Por otra parte, debe de permitir la gestión del servicio por parte del restaurante, principalmente la creación de menús diarios la gestión de la cola de pedidos.

Para la elaboración de este sistema, se ha diseñado y construido una aplicación Web con PHP como lenguaje principal, acompañado de un SGBD MySQL para la persistencia de datos, utilizado mediante la interfaz PDO que ofrece PHP. Por supuesto también se han utilizado las distintas tecnologías que normalmente pueden orquestar un entorno web, como son JavaScript, AJAX, CSS, Bootstrap, etc.

El enfoque del proyecto es mediante el paradigma POO, para lo que se han creado las clases que soportan una posible solución al problema propuesto.

Además de esto, y como filosofía personal, adoptada en base al desarrollo en diferentes frameworks, cada clase contiene la definición de atributos, constructores, métodos, etc., además de la funcionalidad de comunicación directa con la BBDD a través de PDO.

En este sentido, resulta muy sencillo instanciar una clase, pasarle la información que requiere para su construcción, y hacer la misma persistente empleando el mínimo esfuerzo:

Creamos una composición de pedido

```
$idsComposicion = $_POST['nuevoPedido'];  
foreach ($idsComposicion as $idC)  
{  
    $composicion[] = Composicion::CargaComposicion($idC);  
}
```

Creamos un pedido y lo grabamos en el sistema

```
$pedido = new Pedido($fechaPedido, $idCliente, $composicion);  
if($pedido->creaPedido() != -1)  
{  
    if($pedido->guardaLineasPedido())  
    {  
        return $pedido->__get('idPedido');  
    }  
    else  
    {  
        return 'No se pudo crear el pedido especificado';  
    }  
}
```

Para cada pedido tendremos:

- Información sobre la composición del menú (Un primero y/o un segundo y/o postre). Esto permite la preparación de menús y mesas por parte de la plantilla del restaurante.
- Información respecto de los trabajadores (código de empleado). Esto facilita el cobro del servicio, ya que en el acto del cobro del servicio se requiere a cada trabajador su código de empleado, y ayudará en gran medida a agilizar el servicio.
- Fecha y Hora del servicio. Con el mismo fin que el primer punto, para la elaboración y preparación de los distintos menús y mesas.
- Estado. Si se encuentra servido (servido y pagado) o pendiente (sin servir y sin pagar).

Cabecera de pedido


pedido (1r x 4c)			
IdPedido	FechaServicio	Servido	IdEmpleado
322	2021-06-05 14:30:00	1	5

Desglose de pedido

IdPedido	Nombre	Descripcion	Precio
322	Asado de cordero	Plato	3,00
322	Asado de carrillera	Plato	3,00

Para cada menú del día, se debe poder establecer una oferta de:

- Fecha del servicio
- 2 primeros, 2 segundos y 2 postres

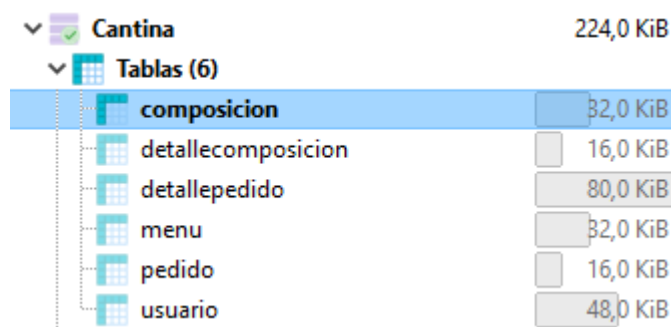
IdMenu		Fecha	Nombre	Descripcion	precio
152		2021-06-08	Paella de magra y costillejas	Plato	3,00
151		2021-06-08	Paella de Carne y Verduras	Plato	3,00
153		2021-06-08	Paella de marisco	Plato	3,00
149		2021-06-08	Espaguetis a la carbonara	Plato	3,00
148		2021-06-08	Mousse de limón	Postre	1,50
150		2021-06-08	Natillas de Chocolate	Postre	1,50

MEMORIA DEL PROYECTO

1. Análisis y diseño del esquema de BBDD a utilizar. Creación de tablas y volcado de registros base. La estructura debe soportar la persistencia de:
 - a. Oferta de menú para cada día. Serán introducidas por el administrador el día anterior al servicio.
 - b. Pedidos para cada día. Un pedido por cada trabajador, aunque puede permitir la creación de un usuario para cada departamento, y, por lo tanto, la creación de pedidos de grupo.
 - c. Usuarios de la aplicación. Normalmente Administrador y Clientes.

Por lo tanto, la estructura de la BBDD quedaría de la siguiente manera:

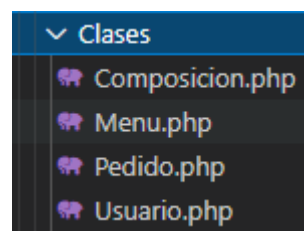
- a. Composicion (Composición de menú, plato)
- b. DetalleComposicion (Tipo: plato, postre o bebida y precio)
- c. Pedido (Cabecera de pedido)
- d. DetallePedido(Desglose de pedido)
- e. Menú (Información de menús diarios)
- f. Usuario (Usuarios de la aplicación)



▼ Cantina	224,0 KiB
▼ Tablas (6)	
composicion	32,0 KiB
detallecomposicion	16,0 KiB
detallepedido	80,0 KiB
menu	32,0 KiB
pedido	16,0 KiB
usuario	48,0 KiB

2. Análisis y construcción de Clases PHP que soporten una posible solución del problema propuesto. Utilizando el paradigma POO. Se compondrá de las siguientes clases PHP:

- a. Composicion (Composición de menú, plato)
- b. Menú
- c. Pedido
- d. Usuario



MEJORAS DEL SISTEMA

Tras el desarrollo se planifican diversas mejoras para futuras implementaciones, algunas especialmente necesarias para la puesta en producción, entre las que podemos destacar:

- Conectar módulo de pago. Es necesario que la aplicación pueda gestionar los pagos de los pedidos en un entorno final.
- Impresión de tickets de pedido. Una vez un pedido está confirmado y pagado, imprimir un ticket con el desglose del pedido por la cocina. Con esto, el jefe de cocina cuenta con las comandas impresas por orden para organizar el servicio.
- Creación de aplicación para dispositivos móviles, una simple aplicación con un visor web que utilice el entorno creado podría ser un buen punto de partida para tener una aplicación web simple de crear y funcional.
- Sistema de notificaciones de entrada de pedidos en tiempo real (integrar en la vista “Pedidos pendientes”).
- Agregar “fichas de producto” en las vistas de cliente, de manera que el usuario pueda tener información sobre ingredientes / alergenicos y una imagen para poder visualizar lo que está pidiendo.
- Añadir apartado con formulario para recibir Feedback de los clientes sobre la aplicación o el servicio de cantina.
- Añadir apartado para posibilitar un control de stock de ingredientes.
- Añadir apartado para controlar y gestionar los pedidos a proveedores.

CONCLUSIÓN

Como conclusión, creo que el sistema cumple con sus objetivos iniciales de permitir la simple creación / modificación de pedidos por parte de los clientes y la gestión de menús y pedidos por parte del servicio de cantina. No requiere de un manual de uso, ya que su usabilidad es la suficiente para que el usuario pueda comprender cómo se utiliza.

Como dato, se ha testado el sistema con 10 usuarios que desconocían por completo la aplicación, y el 100% de ellos ha sabido completar el siguiente flujo sin problemas:

- Registro de usuario
- Login de usuario
- Creación de pedido
- Modificación de pedido

CantinAPP es un sistema diseñado utilizando un patrón “mobile first”, de manera que es totalmente responsive y se adapta a los distintos dispositivos (PC, móviles, Tablets, etc.), por lo que está preparado para su uso por parte de los clientes, que van a utilizar este servicio principalmente desde sus smartphones.

Además de esto, su funcionamiento se basa en peticiones AJAX para recargar el contenido del div principal de manera dinámica. El motor de funcionamiento de la aplicación utiliza el lenguaje Javascript para comunicarse con el servidor PHP, recoger la información solicitada y maquetar y enviar la respuesta hacia el cliente. Con esto se consigue una interacción con el usuario bastante cómoda, ya que no hay recarga en las páginas, y da sensación de aplicación web y no de página.

A nivel interno, como ya se ha comentado, las clases “Core” de funcionamiento de PHP interactúan directamente con la BBDD a través de PDO, por lo que realizar cualquier modificación sobre Pedidos / Menús / etc. resulta bastante simple.

A nivel de despliegue, el proyecto se encuentra alojado en GitHub (código QR en la siguiente página) y cuenta con precisas instrucciones para ponerlo en funcionamiento en cualquier servidor Linux / Windows

REFERENCIAS

GitHub

