

Pedro Sandoval Martínez



2 DAW



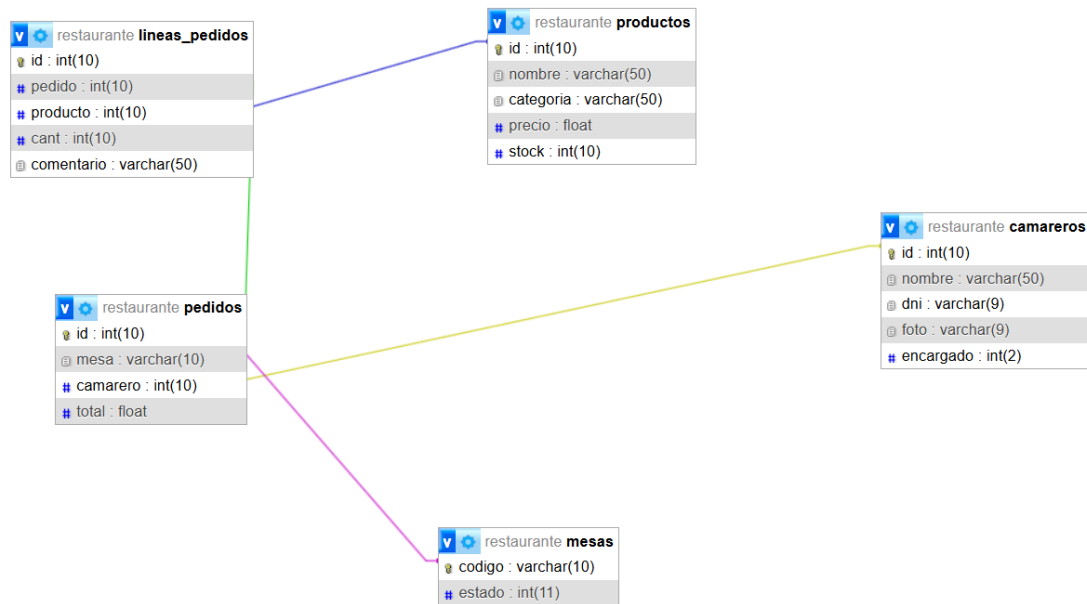
Contenido

Sprint 1	2
Diseño y creación de la base de datos	2
Esquema de Archivos	2
Inicio de sesión y ruta de acceso	3
Menús de opciones	4
Sprint 2	5
Gestión y visualización de mesas	5
Creación de pedidos	5
Intento fallido con Javascript	5
Funcionalidad final	6
Crear Pedido	7
Abrir mesas activas (listar pedido)	8
Anotación Importante	8
Mesa ocupada	9
Cerrar mesa (pagar pedido)	10
Sprint 3	11
Conexión con la impresora	11
Tickets	12
Auditoría de fechas	14
Sprint 4	15
Gestionar pedidos:	15
Gestionar productos:	16
Gestionar camareros	17
Agregar camareros, suspender o eliminar	17
Informes de ventas e historial de pedidos	18
Sprint 5	19
Cambios en el proyecto	19
Referente a los pedidos:	20
Añadir al pedido	21
Productos sin stock en el formulario	22
Guía de uso	24

Sprint 1

Diseño y creación de la base de datos

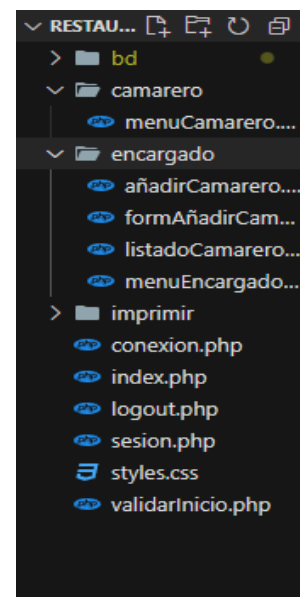
Primer diseño de la base de datos. Centrada sobre todo en la tabla camareros, en la cual he comenzado a trabajar. En principio el id del camarero estará en el pedido para conocer quién registra el mismo.



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.

```
include "conexion.php";

$usuario = $_POST['nombre'];
$clave = $_POST['contraseña'];
$consulta = "SELECT * FROM camareros WHERE nombre='$usuario' AND contraseña='$clave'";

$resultado = mysqli_query($conn, $consulta);

if (mysqli_num_rows($resultado) == 1) {
    $fila = mysqli_fetch_assoc($resultado);

    session_start();

    $_SESSION['id'] = $fila['id'];
    $_SESSION['nombre'] = $fila['nombre'];
    $_SESSION['contraseña'] = $fila['contraseña'];
    $_SESSION['encargado'] = $fila['encargado'];

    if($fila['encargado'] == 1){
        header("LOCATION:encargado/menuEncargado.php");
    }elseif ($fila['encargado'] == 0){
        header("LOCATION:camarero/menuCamarero.php");
    }
} else {
    header("LOCATION:index.php");
}
```

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:

```
<?php
// Seguridad, para que vuelva al index si no tiene la session
session_start();

if(!isset($_SESSION['nombre']) || !isset($_SESSION['id']) ){
    header("LOCATION:../index.php");
    exit();
}

?>
```

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.

```
<div class="centrar">
  <h1>Menu Camareros</h1>
  <!-- Saludar camarero -->
  <?php
    $nombre = $_SESSION['nombre'];
    echo "<h3>Bienvenido, $nombre</h3>";
  ?>
</div>
</nav>
<section>
  <div>
    <a href="salon.php">
      <h3>Salón y comandas</h3>
    </a>
  </div>
  <!-- Añadir mesa falta -->
```

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.

```
<body>
  <nav>
    <div class="centrar">
      <h2>Menú Encargado</h2>
      <!-- Saludar camarero -->
      <?php
        $nombre = $_SESSION['nombre'];
        echo "<h3>Bienvenido, $nombre</h3>";
      ?>
    </div>
  </nav>
  <section class="container">
    <div class="row">
      <a href="listadoCamareros.php">
        <h3>Listado de los camareros</h3>
      </a>
      <a href="formAñadirCamarero.php">
        <h3>Añadir camareros</h3>
      </a>
      <a href=" ../camarero/menuCamarero.php">
        <h3>Vista de Camarero</h3>
      </a>
    </div>
    <!-- Productos -->
    <h4>Gestión de productos</h4>
    <div class="row">
      <a href="ListadoProductos.php">
        <h3>Listado de productos</h3>
      </a>
      <a href="formAñadirProductos.php">
        <h3>Añadir productos</h3>
      </a>
    </div>
  </div>
```

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

Sprint 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 “salon.php” :

```
<div class="mesasContainer">
  <?php
  $consulta = "SELECT * FROM mesas";
  $resultado = mysqli_query($conn, $consulta);

  while ($fila = mysqli_fetch_array($resultado)) {
    $id = $fila['codigo'];
    $ocupada = $fila['estado'];

    if ($ocupada == 0) {
      echo "<div class='mesaLibre'><a class='enlaceMesaId' href='formCrearPedido.php?id=$id'>$id</a></div>";
    } else {
      // Hacemos select en pedidos where mesa = id
      echo "<div class='mesaOcupada'><a class='enlaceMesaId' href='listarPedido.php?id=$id'>$id</a></div>";
    }
  }
  ?>
</div>
```

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**, tratarla 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).

```
<?php
$consulta = "SELECT * FROM productos";

$resultado = mysqli_query($conn, $consulta);

while ($fila = mysqli_fetch_array($resultado)) {

    $id = $fila['id'];
    $nombre = $fila['nombre'];
    $categ = $fila['categoria'];
    $precio = $fila['precio'];
    $stock = $fila['stock'];

    echo "<div class='col-12 productosCaja'>";
    echo "<h5 class='productosItem' name='id' value='$id'>$id</h5>";
    echo "<h4 class='productosItem' name='nombre'>$nombre</h4>";
    echo "<h5 class='productosItem' name='categ'>$categ</h5>";
    echo "<h5 class='productosItem' name='precio'>$precio </h5>";

    echo "<input type='hidden' name='nombresProductos[$id]' value='$nombre'>";
    echo "<input type='checkbox' name='productosSeleccionados[' value='$id'>";

    echo "</div>";
}
?>
<input type="submit" class="btn btn-success" value="Añadir al 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).

```
<div class="container">
<h2>Productos seleccionados</h2>
<form action="crearPedido.php" method="post" class="listadoProd">
<input type="hidden" name="mesaId" value="<?php echo $mesaId; ?>">
<?php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Verificar si se han seleccionado productos
    if (isset($_POST['productosSeleccionados'])) {
        $productosSeleccionados = $_POST['productosSeleccionados']; // Este es el array con los IDs seleccionados
        $nombresProductos = $_POST['nombresProductos'];
        foreach ($productosSeleccionados as $idProducto) {
            $nombreProducto = $nombresProductos[$idProducto];

            echo "<div class='col-12 productosCaja'>";
            echo "<p class='productosItem'>$nombreProducto</p><br>";
            echo "<input type='hidden' name='productosSeleccionados[' value='$idProducto'>";
            echo "<input type='number' class='productosItem' name='cantidades[$idProducto]' placeholder='Cantidad' value='1'>";
            echo "<input type='text' class='productosItem' name='comentarios[$idProducto]' placeholder='Comentario'>";
            echo "</div>";
        }
    }
}
?>
<input type="submit" class="btn btn-primary" value="Enviar pedido">
```

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:

```
$mesaId = $_POST['mesaId'];
$productosSeleccionados = $_POST['productosSeleccionados'];
$cantidades = $_POST['cantidades'];
$comentarios = $_POST['comentarios'];
// esto era para ticket
$nombrsProductos = $_POST['nombrsProductos'];

$total = 0.0;
```

Calculamos total del pedido para su posterior inserción:

```
foreach ($productosSeleccionados as $idProducto) {
    $cantidad = $cantidades[$idProducto];
    $consultaPrecio = "SELECT precio FROM productos WHERE id = '$idProducto'";
    $resultadoPrecio = mysqli_query($conn, $consultaPrecio);
    if ($resultadoPrecio && mysqli_num_rows($resultadoPrecio) > 0) {
        $filaPrecio = mysqli_fetch_assoc($resultadoPrecio);
        $precio = $filaPrecio['precio'];
        $total += $precio * $cantidad;
    }
}
```

Guardamos fecha de creación y realizamos la inserción del pedido:

```
$fechaActual = date('Y-m-d H:i:s');
$fecha = date('Y-m-d', strtotime($fechaActual));
$hora = date('H:i:s', strtotime($fechaActual));

$sqlPedido = "INSERT INTO pedidos (mesa, total, pagado, fecha, hora) VALUES ('$mesaId', '$total', 0, '$fecha', '$hora')";
```

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

```
$pedidoId = $conn->insert_id; // Obtener el ID del pedido recién creado

// Insertar las líneas del pedido
foreach ($productosSeleccionados as $idProducto) {
    $cantidad = $cantidades[$idProducto];
    $comentario = $comentarios[$idProducto];
    $sqlLineaPedido = "INSERT INTO lineas_pedidos (pedido, producto, cant, comentario) VALUES ('$pedidoId', '$idProducto', '$cantidad', '$comentario')";
    if (!$conn->query($sqlLineaPedido)) {
        echo "Error: " . $sqlLineaPedido . "<br>" . $conn->error;
    }
}
```

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.

```
// Actualizar el stock de los productos
foreach ($productosSeleccionados as $idProducto) {
    $cantidad = $cantidades[$idProducto];
    $sqlActualizarStock = "UPDATE productos SET stock = stock - $cantidad WHERE id = '$idProducto'";
    if (!$conn->query($sqlActualizarStock)) {
        echo "Error al actualizar el stock del producto: " . $sqlActualizarStock . "<br>" . $conn->error;
    }
}
```


Y actualizamos el estado de la mesa:

```
// Actualizar el estado de la mesa a 1 (ocupada)
$sqlActualizarMesa = "UPDATE mesas SET estado = 1 WHERE codigo = '$mesaId'";
if (!$conn->query($sqlActualizarMesa)) {
    echo "Error al actualizar el estado de la mesa: " . $sqlActualizarMesa . "<br>" . $conn->error;
}
// reenviamos a ticketCocina
// SI REDIRIGO AQUI ANTES DE LLEGAR A TICKET COCINA no SE IMPRIME BIEN
// header("LOCATION:salon.php");
//reenviamos pasandole por get el id del pedido
header("LOCATION:ticketCocina.php?pedidoId=$pedidoId&mesaId=$mesaId");|
```

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:

```
// Vaciar la tabla lineas_carrito
$sqlVaciarCarrito = "TRUNCATE TABLE lineas_carrito";
if (!$conn->query($sqlVaciarCarrito)) {
    echo "Error al vaciar la tabla lineas_carrito: " . $sqlVaciarCarrito . "<br>" . $conn->error;
}

mysqli_close($conn);
```

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:

```
$mesaId = isset($_GET['mesaId']) ? $_GET['mesaId'] : null;
if ($mesaId == null) {
    die("Error: id de mesa no especificado.");
}

// Obtener el ID del pedido asociado a la mesa y la fecha
$consultaPedido = "SELECT id, hora FROM pedidos WHERE mesa = '$mesaId'"; // Asegúrate de que el estado del pedido es 0 (activo)
$resultadoPedido = mysqli_query($conn, $consultaPedido);
if ($resultadoPedido && mysqli_num_rows($resultadoPedido) > 0) {
    $filaPedido = mysqli_fetch_assoc($resultadoPedido);
    $pedidoId = $filaPedido['id'];
    $horaPedido = $filaPedido['hora'];
} else {
    die("Error: No se encontró un pedido activo para esta mesa.");
}
```

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:

```
$consultaLineas = "
    SELECT lp.*,
           (SELECT p.nombre FROM productos p WHERE p.id = lp.producto) AS nombre
    FROM lineas_pedidos lp
    WHERE lp.pedido = '$pedidoId';
$resultadoLineas = mysqli_query($conn, $consultaLineas);
?>
```

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:

```
if ($resultadoLineas && mysqli_num_rows($resultadoLineas) > 0) {
    echo "<table class='table'>";
    echo "<thead><tr><th>Producto</th><th>Cantidad</th><th>Precio</th><th>Comentario</th></tr></thead>";
    echo "<tbody>";
    while ($filaLinea = mysqli_fetch_assoc($resultadoLineas)) {
        $nombreProducto = $filaLinea['nombre'];
        $cantidad = $filaLinea['cant'];
        $comentario = $filaLinea['comentario'];
        $precio = $filaLinea['precio'];

        $consultaCategoria = "SELECT categoria FROM productos WHERE nombre = '$nombreProducto'";
        $resultadoCategoria = mysqli_query($conn, $consultaCategoria);
        $categoria = '';
        if ($resultadoCategoria && mysqli_num_rows($resultadoCategoria) > 0) {
            $filaCategoria = mysqli_fetch_assoc($resultadoCategoria);
            $categoria = $filaCategoria['categoria'];
        }
    }
}
```

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

```
// Obtener el precio de los productos y guardarlos en un array
$preciosProductos = [];
$consultaPrecios = "SELECT id, precio FROM productos";
$resultadoPrecios = mysqli_query($conn, $consultaPrecios);
if ($resultadoPrecios && mysqli_num_rows($resultadoPrecios) > 0) {
    while ($filaPrecio = mysqli_fetch_assoc($resultadoPrecios)) {
        $preciosProductos[$filaPrecio['id']] = $filaPrecio['precio'];
    }
}
```

Consulta para obtener el total del pedido:

```
// Obtener el total del pedido
$consultaTotal = "SELECT total FROM pedidos WHERE id = '$pedidoId'";
$resultadoTotal = mysqli_query($conn, $consultaTotal);
if ($resultadoTotal && mysqli_num_rows($resultadoTotal) > 0) {
    $filaTotal = mysqli_fetch_assoc($resultadoTotal);
    $totalPedido = $filaTotal['total'];

    // Mostrar con impuestos
    $impuestos = $totalPedido * 0.10;
    $totalConImpuestos = $totalPedido + $impuestos;

    echo "<h2 class='col-5 mt-2 ms-3'>Total: $totalConImpuestos </h2>";
} else {
    echo "<h2>Total: No disponible</h2>";
}
?>
```

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 puse para el siguiente sprint, 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.

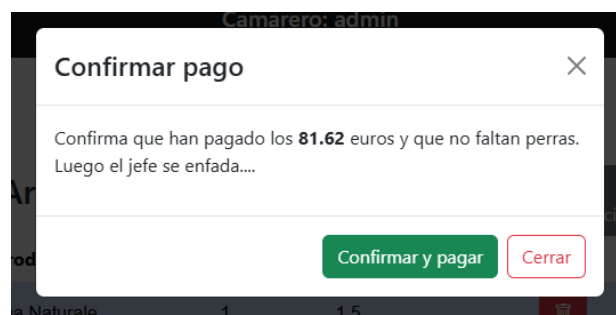
```
// Actualizar el estado de la mesa a 'libre'
$consultaMesa = "UPDATE mesas SET estado=0 WHERE codigo='$mesaId'";
mysqli_query($conn, $consultaMesa);

// Actualizar el estado del pedido a 'pagado'
$consultaPedido = "UPDATE pedidos SET pagado=1, mesa=NULL WHERE id='$pedidoId'";
mysqli_query($conn, $consultaPedido);

header("LOCATION:salon.php");
```

Es una manera fácil de localizar los pedidos más tarde, puesto que no necesitamos la información de la mesa asociada al pedido para futuras auditorías.

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

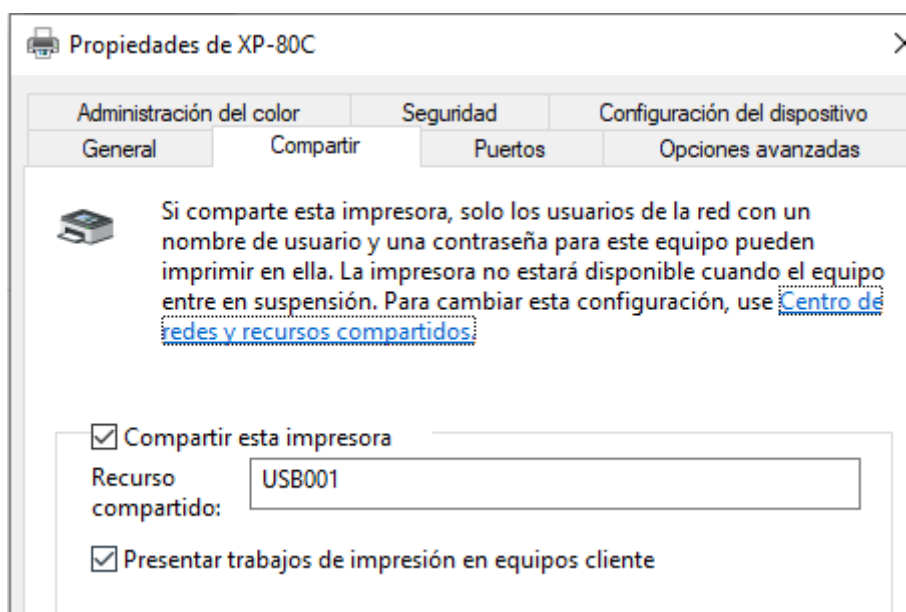


Sprint 3

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.

```
5  /* Change to the correct path if you copy this example! */
6  require __DIR__ . '/../vendor/autoload.php';
7  use Mike42\Escpos\Printer;
8  // para con internet
9  // use Mike42\Escpos\PrintConnectors\NetworkPrintConnector;
10 // este para usb
11 use Mike42\Escpos\PrintConnectors\WindowsPrintConnector;
12
```

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:

```
$nombre = $_SESSION['nombre'];
$mesaId = $_GET['mesaId'];

// Obtener el ID del pedido desde la URL
$pedidoId = isset($_GET['pedidoId']) ? $_GET['pedidoId'] : null;
if ($pedidoId === null) {
    die("Error: id de pedido no especificado.");
}
```

Hacemos la consulta utilizando join:

```
// Consulta pedido,
$query = "
    SELECT lp.cant AS cantidad, lp.comentario, p.nombre AS descripcion, p.categoria
    FROM lineas_pedidos lp
    JOIN productos p ON lp.producto = p.id
    WHERE lp.pedido = ?
";
```

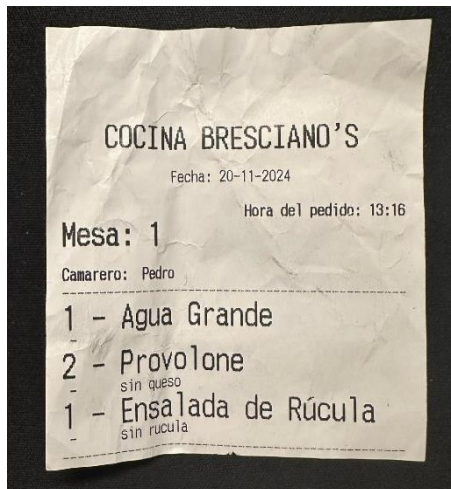
Ejecutamos la consulta y creamos array de productos (ítems) que deben imprimirse, teniendo en cuenta su descripción, cantidad y el comentario (si hay)

```
$items = [];
while ($row = $result->fetch_assoc()) {
    $items[] = [
        "descripcion" => $row['descripcion'],
        "cantidad" => $row['cantidad'],
        "comentario" => $row['comentario']
    ];
}
```

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

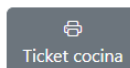
```
$printer->setJustification(Printer::JUSTIFY_LEFT);
foreach ($items as $item) {
    $printer->setTextSize(2, 2);
    $printer->text($item["cantidad"] . " - " . $item["descripcion"]) . "\n";
    $printer->setTextSize(1, 1);
    $printer->text("\n--      " . $item["comentario"] . "\n");
}
```

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.

Artículos del Pedido : 11:39



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:

```
$printer->setEmphasis(true);
$printer->text("Importe (sin iva): " . number_format($totalPedido, 2) . "$ + ");

$impuestos = $totalPedido * 0.10;
$totalConImpuestos = $totalPedido + $impuestos;

$printer->text("Base (10%): " . number_format($impuestos, 2) . "$ \n");
$printer->feed(1);
$printer->setTextSize(2, 2);
$printer->text("Total : " . number_format($totalConImpuestos, 2) . "$ \n");
$printer->setEmphasis(false);
$printer->setTextSize(1, 1);
```

También creamos un CIF ficticio de la empresa y datos como la fecha de impresión, el camarero y la mesa, además de mostrar el total sin y con impuestos. Esto si o si debe aparecer en una factura simplificada como es un ticket.

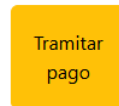
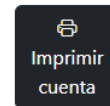
Estos datos deberían ir almacenados en la base de datos del propio restaurante, por si sufriese cambios en el futuro. De momento lo dejamos como posible mejora.

Y el resultado es el siguiente:



En la interfaz, encontramos el botón de imprimir junto al modal donde se confirma el pedido, para poder hacer uso de la función cuantas veces necesitemos.

Total: 33.22 \$



Auditoría de fechas

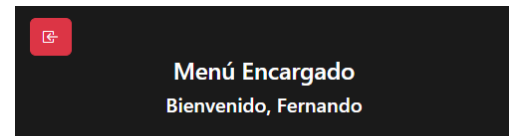
Añado columna fecha para el pedido, y hacer búsqueda de pedidos de ese día. Esta opción ya está añadida en las capturas donde creamos el pedido, pero fue de los últimos cambios que realicé en el código.

```
ALTER TABLE pedidos ADD COLUMN fecha DATETIME DEFAULT CURRENT_TIMESTAMP;
```

Este código cambia la columna "Hora" que usábamos para mostrarla en el ticket, y almacena la fecha completa para permitir la auditoría de los pedidos. No hay cambios al crear el pedido.

Sprint 4

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).

Vista de
camarero


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

Informes de Ventas


Visualiza y descarga los informes de ventas mensuales.

Ver Informes

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.

 Usuarios

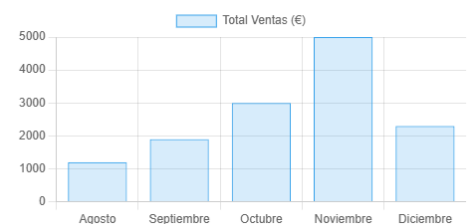
Gestionar

 Productos

Gestionar

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

Estadísticas



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.

```
// Comprobar si el camarero es encargado
$nombre = $_SESSION['nombre'];
$consultaEncargado = "SELECT encargado FROM camareros WHERE nombre = '$nombre'";
$resultadoEncargado = mysqli_query($conn, $consultaEncargado);
if ($resultadoEncargado && mysqli_num_rows($resultadoEncargado) > 0) {
    $filaEncargado = mysqli_fetch_assoc($resultadoEncargado);
    $esEncargado = $filaEncargado['encargado'];
} else {
    $esEncargado = 0; // Asumimos que no es encargado si no se encuentra en la base de datos
}
?>
```

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:

```
echo "<tr class='{$clase}'>";
echo "<td class='tdPedidos'>$nombreProducto</td>";
echo "<td class='tdPedidos'>$cantidad</td>";
echo "<td class='tdPedidos'>$precio</td>";
echo "<td class='tdPedidos'>$comentario";
if($esEncargado == 1){
    echo "<a href='eliminarLinea.php?pedidoId=$pedidoId&nombreProd=$nombreProducto' class='ms-3 btn btn-danger'><i class='bi bi-trash'></i></a>";
}
echo "</td></tr>";
```


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.

Añadir Producto

Pizzas
▼

Precio

Stock

Enviar

Podremos ajustar el stock por defecto que tenga el producto, aunque más tarde sea modificado.

La consulta que realiza es la siguiente:

```
$consulta = "INSERT INTO productos VALUES ('NULL','$nombre','$categ','$precio','$stock')";
```

Junto al formulario encontramos el listado de productos, con la siguiente visual:

Los productos están organizados por categorías, cada una tiene una tabla con un color distinto.

Podemos restablecer stock de todos los productos (a 100), eliminar productos (pone stock a 0) o añadir (+) 50 de stock al producto.

Listado de Productos

Restablecer stock

Bebidas			
Nombre	Precio	Stock	Modificar
Acqua Naturale	1.5 \$	100 U	<div><div></div><div></div></div>
Acqua frizzante	1.5 \$	0 U	<div><div></div><div></div></div>
Coca Cola	2 \$	100 U	<div><div></div><div></div></div>
Fanta Arancia	2 \$	100 U	<div><div></div><div></div></div>
Moretti bottiglia	2.5 \$	100 U	<div><div></div><div></div></div>
Piccola birra	1.5 \$	100 U	<div><div></div><div></div></div>
Vino Rosso	3.5 \$	50 U	<div><div></div><div></div></div>
Vino Bianco	3.5 \$	0 U	<div><div></div><div></div></div>

Los archivos de modificación de stock contienen estas consultas:

```
$id = $_GET['id'];
$sql = "UPDATE productos SET stock = 50 WHERE id = ?";

$id = $_GET['id'];
$sql = "UPDATE productos SET stock = 0 WHERE id = ?";
```

Gestionar camareros

Agregar camareros, suspender o eliminar.









Añadir camareros tiene la misma dinámica que el formulario de añadir productos, con la peculiaridad de que por defecto el usuario añadido está desactivado.

```
$consulta = "INSERT INTO camareros VALUES ('NULL','$usuario','$contra','$dni','$foto','$encargado',1)";
```

Así mostramos los botones para administrar el estado de los camareros:

```
if ($suspendido == 1) {
    echo "<a href='modificarCamarero.php?id=$id&susp=$suspendido' class='ms-3 btn btn-success'><i class='bi bi-toggle-on'></i></a>";
} else {
    echo "<a href='modificarCamarero.php?id=$id&susp=$suspendido' class='ms-3 btn btn-warning'><i class='bi bi-toggle-off'></i></a>";
}
echo "</td></tr>";
```

El listado de camareros tiene esta interfaz. El fondo rojo aparece cuando el usuario está desactivado, y encontramos los botones de activar/desactivar a la derecha.

ID	Nombre	Contraseña	DNI	Foto	¿Encargado?	Opciones
1	admin	admin	11111111E		Si	
3	Rosa Martinez	rosa	33333333A		No	
6	Pedro	redy	48748246E		No	
8	Fernando	maker	46789823F		Si	

La opción de eliminar de la base de datos **no está contemplada**, para evitar problemas en futuras auditorías. Como mejora, propondría hacer otra tabla de antiguos camareros para guardar sus perfiles.

El archivo donde hacemos modificación comprueba primeramente qué vamos a modificar. Si está suspendido (1) lo activa, de lo contrario lo desactiva.

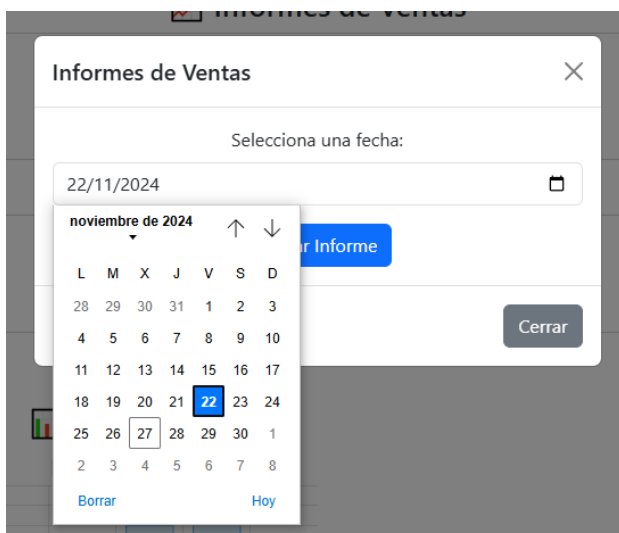
```
<?php
include "sesionEncargado.php";
include "../conexion.php";

$id = $_GET['id'];
$susp = $_GET['susp'];

if ($susp == 1) {
    $sql = "UPDATE camareros SET suspendido = 0 WHERE id = ?";
} else {
    $sql = "UPDATE camareros SET suspendido = 1 WHERE id = ?";
}
```

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.



NAM NAM Rubio Lujan

Pedidos del día 2024-11-22

Pedido ID	Hora	Total
2	16:56:02	35.5 EUR
3	16:57:19	26.5 EUR
4	17:06:08	102.9 EUR
5	17:06:19	120.9 EUR
6	17:06:28	119 EUR
7	17:06:36	94.5 EUR
8	17:06:42	35.5 EUR
9	17:07:17	71.7 EUR

Media de cuentas: 75.81 EUR

Producto + vendido: Focaccia

Productos vendidos: 54 unidades
Total de ingresos del día: 606.5 EUR

Consultas a la base de datos que requiere esta funcionalidad:

```
// Consulta de productos de esa fecha
$fecha = $_POST['fecha'];
$fecha = date('Y-m-d', strtotime($fecha));
$consulta = "SELECT * FROM pedidos WHERE fecha = '$fecha'";

$resultado = mysqli_query($conn, $consulta);

// Consulta del total de todos los productos del día
$totalProductosVendidos = 0;
$consultaLineas = "SELECT SUM(cant) as totalCantidad FROM lineas_pedidos WHERE pedido IN (SELECT id FROM pedidos WHERE fecha = '$fecha')";
$resultadoLineas = mysqli_query($conn, $consultaLineas);
$filasLineas = mysqli_fetch_array($resultadoLineas);
$totalProductosVendidos = $filasLineas['totalCantidad'];

// Consulta del producto más vendido del día
$consultaProductoMasVendido = "SELECT producto, SUM(cant) as totalCantidad FROM lineas_pedidos
WHERE pedido IN (SELECT id FROM pedidos WHERE fecha = '$fecha') GROUP BY producto ORDER BY totalCantidad DESC LIMIT 1";

$resultadoProductoMasVendido = mysqli_query($conn, $consultaProductoMasVendido);
$filaProductoMasVendido = mysqli_fetch_array($resultadoProductoMasVendido);
$productoMasVendidoId = $filaProductoMasVendido['producto'];
$totalCantidadProductoMasVendido = $filaProductoMasVendido['totalCantidad'];

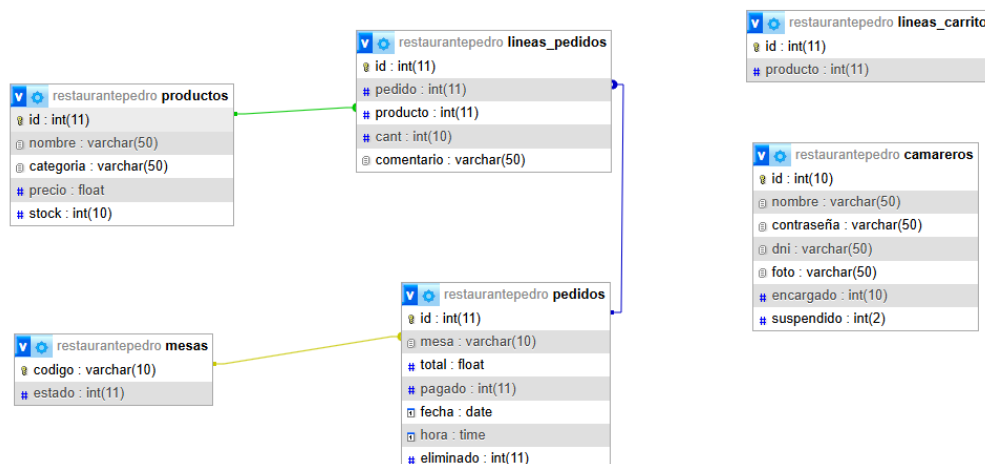
$consultaNombreProducto = "SELECT nombre FROM productos WHERE id = '$productoMasVendidoId'";
$resultadoNombreProducto = mysqli_query($conn, $consultaNombreProducto);
$filaNombreProducto = mysqli_fetch_array($resultadoNombreProducto);
$nombreProductoMasVendido = $filaNombreProducto['nombre'];
```

Realizamos subconsultas sencillas haciendo uso de lenguaje SQL y las relaciones entre tablas. Calculamos la media, el producto que más aparece en las líneas de pedido (más vendido), el total de productos que se han pedido y el total de ingresos (suma del total de todos los pedidos).

Sprint 5

Cambios en el proyecto

La base de datos a lo largo del proyecto ha sufrido cambios, culpa del mal diseño y planificación de proyecto y los requisitos del mismo.



Como modificaciones importantes encontramos el atributo “eliminado” en pedidos, y “suspendido” en camareros, que ya hemos implementado en los apartados de gestión.

Referente a los pedidos:

El encargado tiene opción de borrar artículos del pedido o (imaginemos en caso de error) eliminar el pedido al completo. Interesa llevar un registro de estos pedidos también, por eso lo añadimos en forma de atributo.

Artículos del Pedido : 16:26

 Ticket cocina

Producto	Cantidad	Precio	Comentario
Acqua Naturale	1	1.5	
Moretti bottiglia	1	2.5	
Vino Rosso	1	3.5	
Speck	1	10.2	
Tagliere formaggi	1	14.2	
Spaghetti Carbonara	1	9.5	
Spaghetti Carbonara	1	9.5	
Lasagna	1	17	

Eliminar pedido

Esta opción la implementamos también en la auditoría de ventas, donde reflejamos que efectivamente hay pedidos eliminados. Estos no cuentan de cara a los productos vendidos ni al total de la caja, como se ve reflejado en el total de las cuentas.

NAM NAM Rubio Lujan

Pedidos del día 2024-11-26

Pedido ID	Hora	Total
3	16:29:14	17.5 EUR
4	16:29:48	17.5 EUR
6	16:39:38	35.2 EUR
7	16:50:28	17.5 EUR

Pedidos Eliminados

Pedido ID	Hora	Total
1	16:26:24	67.9 EUR
2	16:26:41	30.2 EUR
5	16:30:39	30.2 EUR

Media de cuentas: 12.53 EUR

Producto + vendido: Insalata CÃ©sar

Productos vendidos: 9 unidades

Total de ingresos del día: 87.7 EUR

Para no tener en cuenta los pedidos eliminados en los resultados, añadimos esta sencilla cláusula a las consultas:

```
WHERE fecha = '$fecha' AND eliminado = 0);
```

Añadir al pedido

Añado en este punto la opción de añadir más productos al pedido una vez creado, por un malentendido tuve que crear la funcionalidad más tarde. Creamos el archivo “añadirPedido”, que incluye las mismas inserciones que “crearPedido” salvo estas modificaciones:

```
// actualizar pedido

$sqlPedido = "UPDATE pedidos SET total = total + '$total' WHERE id = '$pedidoId'";
if ($conn->query($sqlPedido)) {
    // Insertar las líneas del pedido
    foreach ($productosSeleccionados as $idProducto) {
        $cantidad = $cantidades[$idProducto];
        $comentario = $comentarios[$idProducto];

        $sqlLineaPedido = "INSERT INTO lineas_pedidos (pedido, producto, cant, comentario)
        VALUES ('$pedidoId', '$idProducto', '$cantidad', '$comentario')";
        if (!$conn->query($sqlLineaPedido)) {
            echo "Error: " . $sqlLineaPedido . "<br>" . $conn->error;
        }
    }

    // Actualizar el stock de los productos
    foreach ($productosSeleccionados as $idProducto) {
        $cantidad = $cantidades[$idProducto];
        $sqlActualizarStock = "UPDATE productos SET stock = stock - $cantidad WHERE id = '$idProducto'";
        if (!$conn->query($sqlActualizarStock)) {
            echo "Error al actualizar el stock del producto: " . $sqlActualizarStock . "<br>" . $conn->error;
        }
    }
}
```

En este código observamos la actualización del total del pedido teniendo en cuenta los nuevos productos. Los añadimos a las líneas del pedido y además actualizamos el stock de la tabla productos para su correspondiente auditoría.

La interfaz sigue siendo igual, salvo que apreciamos el formulario debajo del listado del pedido para añadir esta funcionalidad.

Artículos del Pedido : 16:26 Ticket cocina

Producto	Cantidad	Precio	Comentario
Acqua Naturale	1	1.5	
Moretti bottiglia	1	2.5	
Vino Rosso	1	3.5	
Speck	1	10.2	
Tagliere formaggi	1	14.2	
Spaghetti Carbonara	1	9.5	
Spaghetti Carbonara	1	9.5	
Lasagna	1	17	

Añadir al Pedido

Bebidas
Entrantes
Ensaladas
Pastas
Pizzas
Postres

Añadir al carrito

Productos seleccionados

Producto	Cantidad	Comentario	Editar
----------	----------	------------	--------

Añadir al pedido

Productos sin stock en el formulario

Implementamos el stock de forma dinámica a la hora de tomar nota, esta funcionalidad es muy útil para los camareros, ya que saben a tiempo real los productos disponibles sin necesidad de preguntar o mirar en el listado.

Comprobamos el stock antes de mostrar el formulario con los productos, y ajustamos los parámetros con las condiciones:

```

if ($stock < 10 && $stock > 1) {
    echo "<div class='d-grid col-6 col-sm-6 col-md-4 col-lg-3 mb-1'>";
    echo "<input type='checkbox' name='productosSeleccionados[]' id='$nombre' class='btn-check' value='$id'>";
    echo "<label for='$nombre' class='$clase'$nombre <br>(consultar stock)</label>";
    echo "</div>";
} else if ($stock == 0) {
    echo "<div class='d-grid col-6 col-sm-6 col-md-4 col-lg-3 mb-1'>";
    echo "<input type='checkbox' name='productosSeleccionados[]' id='$nombre' class='btn-check' value='$id' disabled>";
    echo "<label for='$nombre' class='$clase'$nombre <br>SIN STOCK</label>";
    echo "</div>";
} else {
    echo "<div class='d-grid col-6 col-sm-6 col-md-4 col-lg-3 mb-1'>";
    echo "<input type='checkbox' name='productosSeleccionados[]' id='$nombre' class='btn-check' value='$id'>";
    echo "<label for='$nombre' class='$clase'$nombre</label>";
    echo "</div>";
}

```

Si el stock del producto es menor que 10 (y mayor que 1), avisa que no quedan muchas unidades. Si directamente es 0, el input es “disabled”, no nos dejaría pedirlo. Finalmente, si no se cumple ninguna de esas condiciones, se muestra normalmente.



Bebidas	
Acqua Naturale	Acqua frizzante SIN STOCK
Coca Cola (consultar stock)	Fanta Arancia
Moretti bottiglia	Piccola birra
Vino Rosso	Vino Bianco SIN STOCK

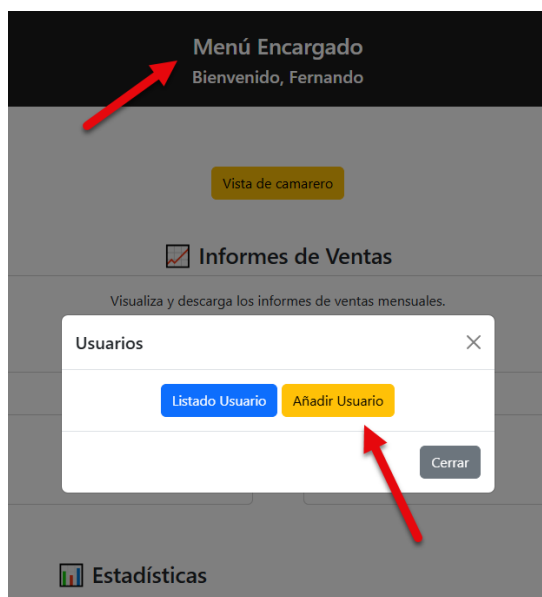
Hasta aquí las alteraciones en el código. Después de muchas horas de trabajo, decidí terminar aquí. Tras mucho esfuerzo e ideas, esto me ha servido de base para futuros proyectos, y me queda pendiente implementar todas aquellas mejoras en la aplicación para hostelería que quiero diseñar.

Guía de uso

Vamos a realizar una guía de uso sencilla para los supuestos trabajadores que utilicen la aplicación. Primero comenzamos con el registro e inicio de sesión.

Primer paso

El encargado debe dar de alta a los camareros a través de su perfil.



Esto nos llevará al formulario que debe completarse con los datos del camarero.

Segundo paso

Acceder a la web principal e introducir tus credenciales.



Menú del camarero

Nos aparecerán dos secciones, la primera de ellas de acceso al salón, donde tramitaremos las comandas, y la segunda a un listado de los productos y sus cantidades, por si requerimos consultar esa información antes del servicio.

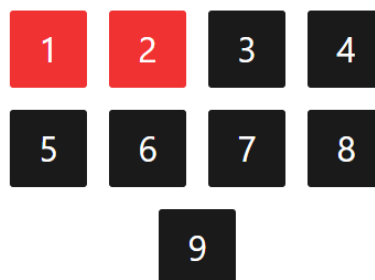


Abajo podremos observar el día del servicio al que estamos accediendo.

Salón

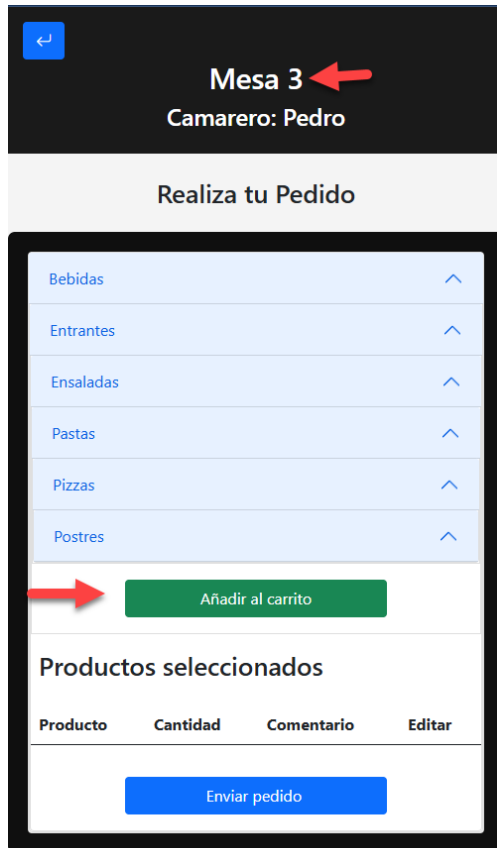
Una vez accedemos al salón, veremos el diseño de las mesas y su distribución. Las que se encuentren marcadas en rojo significa que están ocupadas (ya hay un pedido en ellas).

En cambio, las de color negro están libres. Hasta que no se realice el pago de la mesa o se elimine el pedido (solo el encargado puede eliminar), no se liberará la mesa.



Toma de pedido

Cuando accedemos a una mesa libre, nos aparecerá la interfaz para tomar un pedido.



Arriba nos aparece la mesa en la que estamos realizándolo.

Abajo tenemos menús desplegables divididos en secciones donde encontraremos los productos.

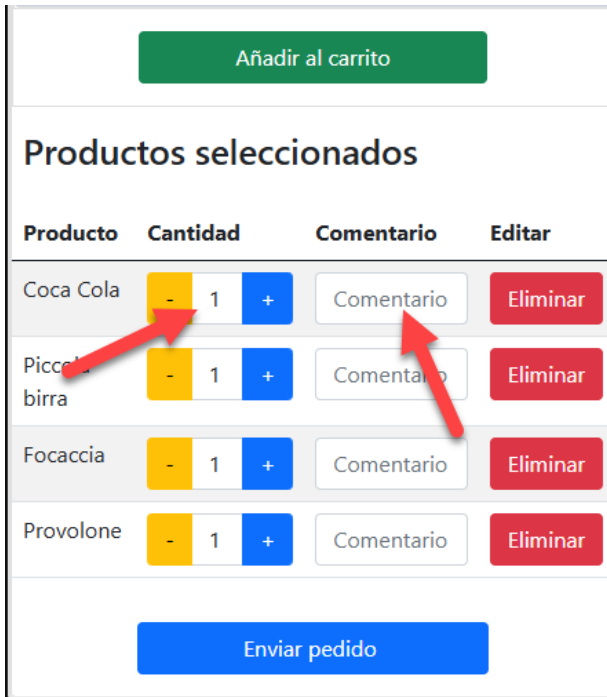
El botón verde añade las cosas al carrito, donde confirmaremos la cantidad del producto seleccionado y el comentario (si tiene).

Cuando pulsamos sobre una categoría nos aparecen los productos.

Atento a las indicaciones, nos indican si hay stock o si debemos consultar porque quedan pocos.



Cuando pulsamos sobre el botón del producto y añadimos al carrito, nos aparecerá el listado de productos seleccionados.



Productos seleccionados

Producto	Cantidad	Comentario	Editar
Coca Cola	- 1 +	Comentario	Eliminar
Piccola birra	- 1 +	Comentario	Eliminar
Focaccia	- 1 +	Comentario	Eliminar
Provolone	- 1 +	Comentario	Eliminar

Añadir al carrito

Enviar pedido

Aumentamos la cantidad si es necesario y añadimos un comentario.

Atención antes de pulsar el botón para enviar el pedido, confirma que el cliente ha pedido esos productos para evitar errores.

Recuerda que sólo el encargado puede eliminar los artículos una vez enviado a cocina.

Cuando enviamos el pedido, el ticket llega directamente a cocina.

Cuando entremos a una mesa ocupada, nos aparecerá el listado de lo que han pedido.

Más abajo, encontramos la visual de antes por si necesitamos pedir algún artículo más.



Mesa 1
Camarero: Pedro

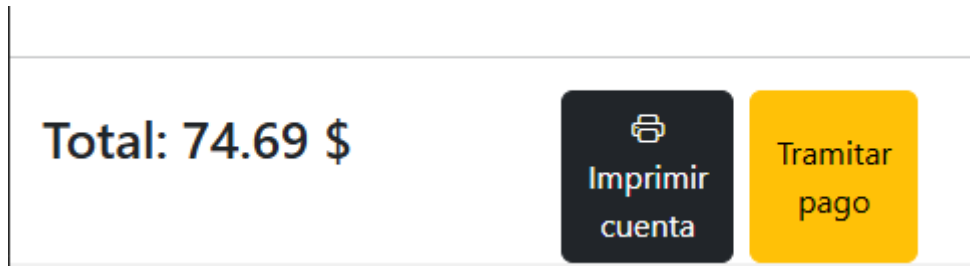
Artículos del Pedido : 16:26 Ticket cocina

Producto	Cantidad	Precio	Comentario
Acqua Naturale	1	1.5	
Moretti bottiglia	1	2.5	
Vino Rosso	1	3.5	
Speck	1	10.2	
Tagliere formaggi	1	14.2	
Spaghetti Carbonara	1	9.5	
Spaghetti Carbonara	1	9.5	
Lasagna	1	17	

Añadir al Pedido

- Bebidas
- Entrantes
- Ensaladas

Pagar cuenta y liberar mesa

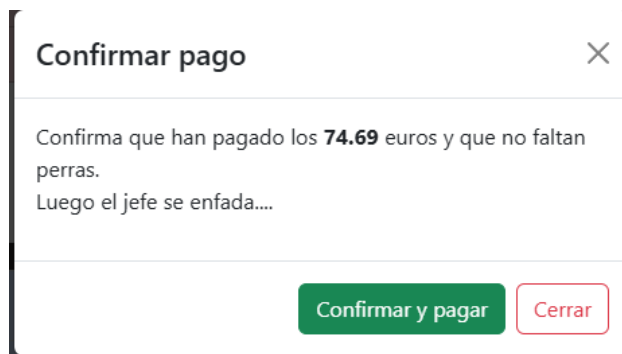


Total: 74.69 \$

Imprimir cuenta

Tramitar pago

Nos encontramos estos dos botones al final del listado de la cuenta. Podemos imprimir el ticket, y una vez nos aseguremos que está pagada, confirmamos que así sea.



Confirmar pago

Confirma que han pagado los **74.69** euros y que no faltan perras.
Luego el jefe se enfada....

Confirmar y pagar Cerrar

Tras esto, la mesa será liberada y podremos tomar nuevamente nota en ella.

Anotaciones

Si quiere realizar una modificación de un producto ya enviado, recurra a su encargado para que elimine el producto anterior, y envíelo de nuevo con la modificación.

Si ha tenido algún error y ha mandado productos sin querer, indique a su encargado el número de la mesa para eliminarlos.

Si abre una mesa sin querer y queda ocupada, siga el mismo procedimiento anterior.

Final

Si durante el uso encuentra más errores o problemas que dificulten su trabajo, por favor contacten con el soporte técnico y estaremos encantados de ayudarles.

- Tlf: 622468052
- Correo: pedro.dev.business@gmail.com