



# MEMORIA PRÁCTICA RESTAURANTE PHP

Web Restaurante

Antonio Jesús Ibáñez López

## Contenido

Memoria del Proyecto Restaurante .....	4
Sprint 1 - Fundamentos del Sistema y Roles .....	4
1. Introducción.....	4
2. Objetivos .....	4
3. Estructura del Proyecto.....	4
4. Proceso de Desarrollo.....	5
Sprint 2 - Módulo Camarero .....	8
1. Introducción.....	8
2. Estructura del Módulo.....	8
3. Funcionalidades Implementadas.....	9
4. Mejoras en la Interfaz de Usuario.....	10
5. Seguridad Implementada.....	11
6. Funcionalidades de Gestión de Pedidos .....	12
7. Mejoras en la Experiencia de Usuario.....	12
8. Próximas Mejoras.....	13
9. Conclusiones .....	13
Sprint 3 - Gestión de Cuentas y Pagos .....	14
1. Base de Datos.....	14
2. Nuevas Funcionalidades.....	14
3. Mejoras en la Interfaz .....	16
4. Optimizaciones .....	17
5. Archivos Modificados.....	17
Próximos Pasos.....	17
6. Conclusiones:.....	18
<b>Sprint 4: Mejora del Sistema de Tickets y Manejo de Errores.....</b>	<b>18</b>
<b>1. Sistema de Generación de Tickets Mejorado.....</b>	<b>18</b>
<b>2. Integración con Impresora Térmica.....</b>	<b>18</b>
<b>3. Sistema de Notificaciones.....</b>	<b>18</b>
Sprint 4.5 - Mejoras visuales y funcionalidad PDF .....	23
1. Unificación del diseño.....	23
2. Mejoras en la interfaz de usuario .....	23
3. Implementación de exportación PDF .....	24

4. Mejoras en la gestión de cuentas .....	25
5. Optimizaciones generales.....	26
6. Próximos pasos.....	26
7. Conclusiones .....	26
Sprint 5: Mejoras en la Interfaz del Panel de Encargado .....	27
1. Adaptación Responsive del Panel Principal .....	27
2. Mejora en el Listado de Camareros .....	27
3. Optimización del Formulario de Registro.....	28
4. Características Técnicas Implementadas .....	28
5. Aspectos de Seguridad.....	28
6. Mejoras en la Experiencia de Usuario .....	28
7. Conclusiones .....	29
Sprint 6: Añadir mejoras y gestión de productos en (encargado).....	29
Introducción.....	29
1. Registro de Usuarios.....	30
2. Listado y Gestión de Usuarios .....	30
3. Gestión de Productos .....	31
4. Interfaz Responsiva.....	32
Conclusiones .....	34
BASE DE DATOS FINAL:.....	34
GUIA DE USO: .....	36
Introducción.....	36
Pantalla de inicio del portal: .....	36
SECCION DE ENCARGADO:.....	37
Index.php(encargado) .....	37
Registrar Usuario: .....	38
Listar Usuarios:.....	38
Acceder como Camarero: .....	39
Gestionar Productos: .....	40
Generar Informe: .....	41
Informe de ventas: .....	41
.....	41
SECCION DE CAMARERO: .....	42
Index.php (camarero):.....	42

Gestionar mesas: .....	43
Gestionar pedido: .....	44
Cuenta:.....	45
Historial de cuentas pagadas: .....	47
Instalación de impresoras de tickets y PrintNode (para su uso a través de web) .....	49
Instalación local y conexión con composer y librerías mike42: .....	49
Instalación web con PrintNode: .....	54
Página Web del Restaurante.....	57
<b>Conclusión Final</b> .....	58

## Memoria del Proyecto Restaurante

### Sprint 1 - Fundamentos del Sistema y Roles

#### 1. Introducción

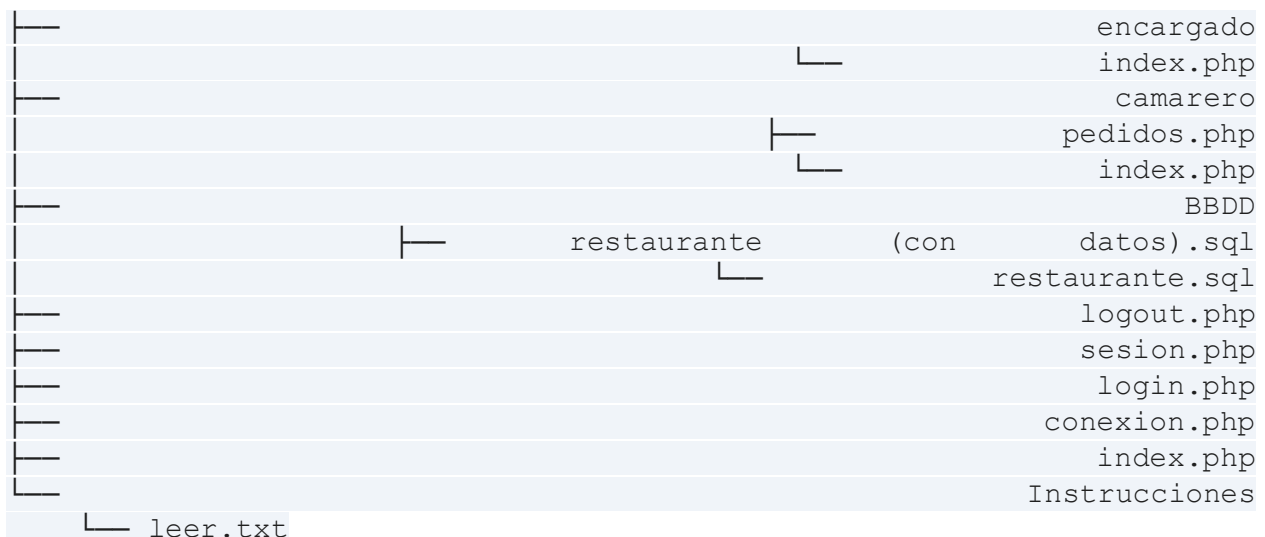
Este proyecto se centra en el desarrollo de un sistema de gestión integral para un restaurante. El sistema abarca funcionalidades esenciales como la gestión de usuarios (con roles diferenciados), la administración de mesas, el procesamiento de pedidos y el control de productos. La modularidad del sistema permite adaptarlo a las necesidades específicas del restaurante, ofreciendo interfaces personalizadas para camareros y encargados.

#### 2. Objetivos

- **Automatizar la gestión de pedidos:** Agilizar la toma de pedidos, la comunicación con la cocina y la generación de cuentas.
- **Controlar el inventario:** Mantener un registro actualizado de los productos disponibles, facilitando la gestión de stock y la previsión de compras.
- **Mejorar la atención al cliente:** Optimizar los tiempos de espera y reducir errores en las comandas.
- **Facilitar la administración del restaurante:** Proporcionar al encargado herramientas para la gestión de usuarios, el análisis de ventas y la generación de informes.

#### 3. Estructura del Proyecto

La estructura de archivos y directorios del proyecto se organiza de la siguiente manera:



### 3.1. Descripción de Archivos

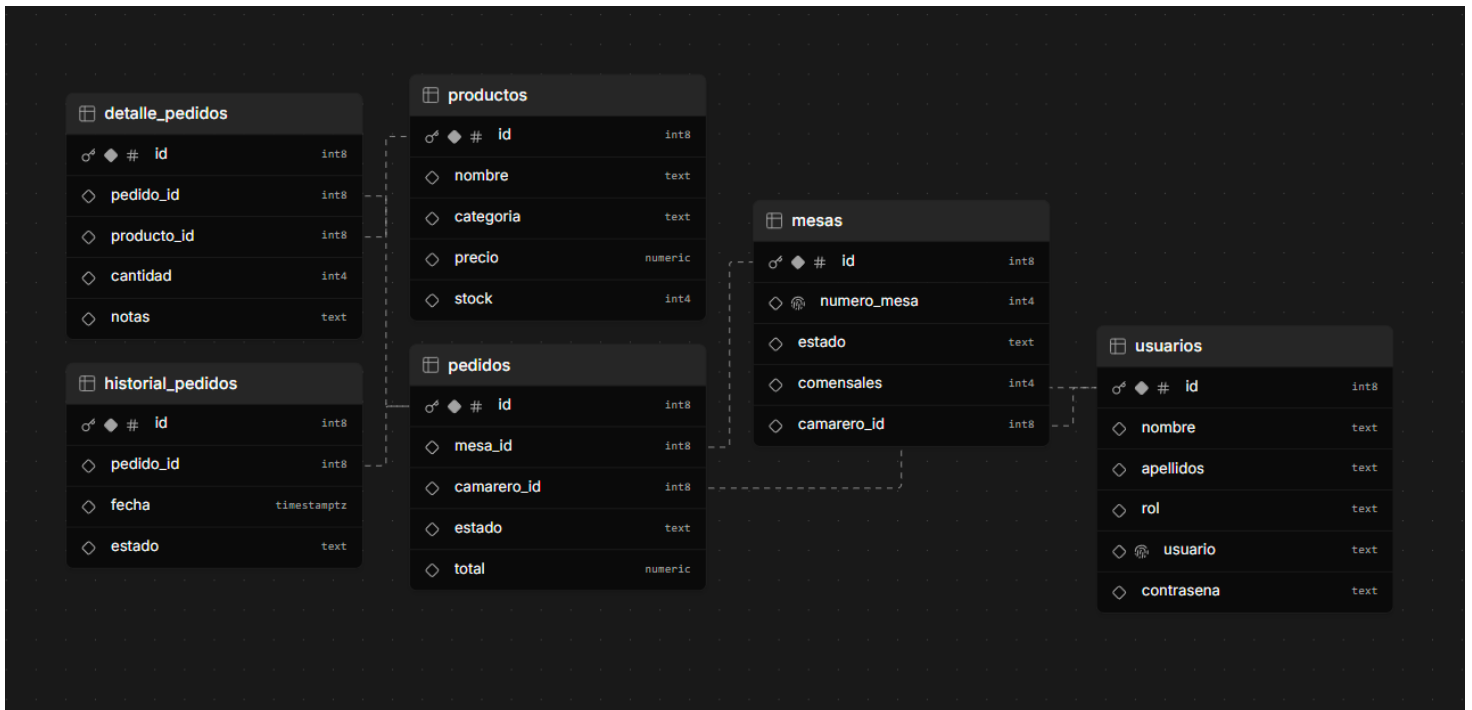
- **BBDD/restaurante.sql:** Define la estructura de la base de datos, incluyendo tablas, campos y relaciones.
- **BBDD/restaurante (con datos).sql:** Contiene la estructura de la base de datos con datos de ejemplo para facilitar las pruebas iniciales.
- **camarero/index.php:** Panel principal para camareros, con acceso a la gestión de mesas y pedidos.
- **camarero/pedidos.php:** Permite a los camareros tomar nuevos pedidos, modificar existentes y enviarlos a la cocina.
- **encargado/index.php:** Interfaz de administración para el encargado, con funcionalidades para la gestión de usuarios, productos, informes, etc.
- **conexion.php:** Centraliza la configuración de la conexión a la base de datos.
- **index.php:** Página principal del sistema, redirige a los usuarios según su rol.
- **login.php:** Gestiona la autenticación de usuarios, controlando el acceso al sistema.
- **logout.php:** Cierra la sesión del usuario.
- **sesion.php:** Implementa la gestión de sesiones para mantener el estado del usuario durante su interacción con el sistema.

## 4. Proceso de Desarrollo

El desarrollo del proyecto se llevó a cabo siguiendo las siguientes etapas:

### 4.1. Diseño de la Base de Datos

Se definieron las entidades necesarias para el sistema (usuarios, mesas, pedidos, productos) y se establecieron las relaciones entre ellas. Se utilizaron claves primarias y foráneas para asegurar la integridad referencial de los datos.

Primer esquema relacional:

#### 4.2. Implementación del Sistema de Autenticación

Se implementó un sistema de autenticación de usuarios para restringir el acceso a las funcionalidades del sistema.

Ejemplo de código PHP (sesion.php):

##### 1. Archivo `sesion.php`

- Este archivo se encarga de iniciar la sesión y verificar si el usuario está autenticado.
- Código:

```
<?php
session_start();

if (!isset($_SESSION['id']) && !isset($_SESSION['usuario'])) {
    header("Location: index.php");
    exit();
}
```

Ejemplo de código PHP (logout.php):

## 2. Archivo `logout.php`

- Este archivo se encarga de cerrar la sesión del usuario.
- Código:

```
<?php
session_start();
session_destroy();
header("Location: index.php");
exit();
```

Ejemplo de código PHP (login.php):

## 3. Archivo `login.php`

- Este archivo se encarga de manejar el inicio de sesión del usuario.
- Código:

```
<?php
include 'conexion.php';

$usuario = $_POST['usuario'];
$contrasena = $_POST['contrasena'];

$query = "SELECT * FROM usuarios WHERE usuario='$usuario' AND contrasena='$contrasena'";
$result = mysqli_query($conexion, $query);

if (mysqli_num_rows($result) == 1) {
    $row = mysqli_fetch_assoc($result);
    session_start();
    $_SESSION['usuario'] = $usuario;
    $_SESSION['rol'] = $row['rol'];
    $_SESSION['id'] = $row['id'];

    if ($row['rol'] == 'camarero') {
        header("Location: camarero/index.php");
    } elseif ($row['rol'] == 'encargado') {
        header("Location: encargado/index.php");
    }
} else {
    echo "Usuario o contraseña incorrectos.";
    echo "<br>";
    echo "<a href='index.php'>Volver</a>";
}
```



#### 4.3. Desarrollo de la Interfaz de Usuario

Se diseñaron interfaces intuitivas y fáciles de usar para cada tipo de usuario (camarero y encargado). Se utilizaron tecnologías web como HTML, CSS y JavaScript para crear las interfaces.

#### 4.4. Implementación de la Lógica de Negocio

Se programaron las funcionalidades del sistema, como la toma de pedidos, la gestión de mesas, el control de stock, etc. Se utilizó PHP y consultas SQL para interactuar con la base de datos.

#### 4.5. Pruebas y Validación

Se realizaron pruebas exhaustivas para asegurar el correcto funcionamiento del sistema y la integridad de los datos. Se probaron diferentes escenarios y casos de uso para identificar y corregir posibles errores.

## Sprint 2 - Módulo Camarero

### 1. Introducción

Durante el Sprint 2, se ha desarrollado y mejorado el módulo de camarero del sistema de gestión de restaurante. Este módulo permite la gestión eficiente de mesas, pedidos y productos, con una interfaz responsive y fácil de usar.

### 2. Estructura del Módulo

#### 2.1 Organización de Archivos

```
camarero/  
├── index.php           # Panel principal  
├── gestionar_mesas.php # Gestión de mesas  
├── gestionar_pedido.php # Gestión de pedidos  
├── seleccionar_producto.php # Selección de productos  
├── productos_anadidos.php # Lista de productos en pedido  
└── js/  
    └── scripts.js      # Funciones JavaScript
```

### 3. Funcionalidades Implementadas

#### 3.1 Gestión de Mesas

Ejemplo de código para la gestión de mesas activas:

```
<?php
function obtener_mesas_activas($conexion) {
    return mysqli_query($conexion, "SELECT * FROM mesas WHERE estado = 'activa'");
}

// Lógica para activar mesa
if ($_SERVER['REQUEST_METHOD'] == 'POST' && isset($_POST['activar_mesa'])) {
    $mesa_id = $_POST['mesa_id'];
    $comensales = $_POST['comensales'];

    mysqli_query($conexion, "UPDATE mesas SET estado = 'activa', comensales = $comensales WHERE id = $me
    mysqli_query($conexion, "INSERT INTO pedidos (mesa_id, estado, total) VALUES ($mesa_id, 'pendiente',
}
```

#### 3.2 Gestión de Pedidos

Se implementó un sistema completo de gestión de pedidos con las siguientes características:

**Código de productos añadidos:**

```
<?php
function obtener_detalle_pedidos($conexion, $mesa_id) {
    $query = "SELECT dp.*, p.nombre as nombre_producto, p.precio
              FROM detalle_pedidos dp
              INNER JOIN productos p ON dp.producto_id = p.id
              INNER JOIN pedidos ped ON dp.pedido_id = ped.id
              WHERE ped.mesa_id = $mesa_id AND ped.estado = 'pendiente'";
    return mysqli_query($conexion, $query);
}
```

## Sistema de categorías de productos:

```
<?php
$categorias = [
    'pizzas' => 'Pizzas',
    'ensalada' => 'Ensaladas',
    'bebida' => 'Bebidas',
    'carne' => 'Carnes',
    'pasta' => 'Pasta',
    'pescado' => 'Pescado',
    'vino' => 'Vinos'
];
```

## 4. Mejoras en la Interfaz de Usuario

### 4.1 Diseño Responsive

```
@media (max-width: 768px) {
    .mesa-card .card-body {
        padding: 1rem;
    }

    .mesa-card .fa-2x {
        font-size: 1.5em;
    }

    .mesa-card .card-title {
        font-size: 1rem;
    }

    .mesa-card .card-text {
        font-size: 0.8rem;
    }
}
```

## 4.2 Interactividad con JavaScript

```
function modificarCantidad(detalle_id, cantidad, notas) {  
    document.getElementById('mod_detalle_id').value = detalle_id;  
    document.getElementById('mod_cantidad').value = cantidad;  
    document.getElementById('mod_notas').value = notas;  
    new bootstrap.Modal(document.getElementById('modificarModal')).show();  
}
```

## 5. Seguridad Implementada

### 5.1 Validación de Sesiones

```
<?php  
if (!isset($_SESSION['rol']) || $_SESSION['rol'] !== 'camarero') {  
    header('Location: ../index.php');  
    exit;  
}
```

### 5.2 Sanitización de Datos

```
<?php  
$mesa_id = filter_input(INPUT_GET, 'mesa_id', FILTER_VALIDATE_INT);  
if (!$mesa_id) {  
    header('Location: gestionar_mesas.php');  
    exit;  
}
```

## 6. Funcionalidades de Gestión de Pedidos

### 6.1 Cálculo de Totales

```
<?php
$total = 0;
while ($detalle = mysqli_fetch_assoc($detalle_pedidos_result)) {
    $subtotal = $detalle['cantidad'] * $detalle['precio'];
    $total += $subtotal;
}
```

### 6.2 Sistema de Confirmación

```
function confirmarEliminacion() {
    return confirm('¿Está seguro de eliminar este producto?');
}
```

## 7. Mejoras en la Experiencia de Usuario

### 7.1 Notificaciones y Feedback

- Mensajes de confirmación para acciones importantes
- Indicadores visuales de estado
- Transiciones suaves en la interfaz

### 7.2 Optimización de Rendimiento

- Consultas SQL optimizadas
- Carga asíncrona de datos

- Minimización de recargas de página

## 8. Próximas Mejoras

- Implementación de sistema de notificaciones en tiempo real
- Mejora en la gestión de inventario
- Implementación de estadísticas de ventas
- Sistema de reserva

## 9. Conclusiones

**El Sprint 2 ha logrado implementar un sistema robusto y fácil de usar para la gestión de mesas y pedidos, con especial énfasis en la experiencia de usuario y la adaptabilidad a dispositivos móviles. Las mejoras en la interfaz y la implementación de funcionalidades clave han resultado en un sistema más eficiente y fácil de usar para los camareros. He tenido que ajustar la base de datos de nuevo y tener que quitar el "id" de las mesas ya que me hacía tener que comprobar el camarero con su "id" y me creaba demasiados errores, por lo demás la base de datos creo que hacerle con ella y estoy teniendo muy buenos resultados.**

## Sprint 3 - Gestión de Cuentas y Pagos

### 1. Base de Datos

- Creación de nueva tabla cuentas\_pagadas para almacenar el historial de pagos
- Eliminación de tablas innecesarias (temp\_ticket, historial\_pedidos)
- Optimización de la estructura de la tabla cuenta actual

### 2. Nuevas Funcionalidades

#### 2.1 Gestión de Cuentas

- Implementación de vista detallada de cuenta por mesa
- Cálculo automático de subtotales y total
- Interfaz responsive para visualización de productos en la cuenta
- Funcionalidad para procesar pagos

Ejemplo de procesamiento de pago:

```
<?php
try {
    mysqli_begin_transaction($conexion);

    foreach ($productos_cuenta as $item) {
        $insert_cuenta = "INSERT INTO cuentas_pagadas
                           (mesa_id, producto, cantidad, precio_unitario, subtotal)
                           VALUES (?, ?, ?, ?, ?)";
        $stmt = mysqli_prepare($conexion, $insert_cuenta);
        mysqli_stmt_bind_param($stmt, "isids",
                                $mesa_id,
                                $item['nombre_producto'],
                                $item['cantidad'],
                                $item['precio_unitario'],
                                $item['subtotal']
        );
        mysqli_stmt_execute($stmt);
    }

    mysqli_commit($conexion);
} catch (Exception $e) {
    mysqli_rollback($conexion);
}
```

## 2.2 Historial de Cuentas

- Nueva sección de historial de cuentas pagadas
- Visualización de cuentas agrupadas por mesa y fecha
- Modal para ver detalles de cuentas históricas
- Sistema de consulta mediante AJAX para cargar detalles
- Ejemplo de consulta para obtener historial:



```
<?php
$query = "SELECT
    cp.mesa_id,
    m.numero_mesa,
    DATE(cp.fecha_hora) as fecha,
    TIME(cp.fecha_hora) as hora,
    COUNT(*) as num_productos,
    SUM(cp.subtotal) as total
FROM cuentas_pagadas cp
INNER JOIN mesas m ON cp.mesa_id = m.id
GROUP BY cp.mesa_id, DATE(cp.fecha_hora), TIME(cp.fecha_hora)
ORDER BY cp.fecha_hora DESC";
```

### 2.3 Dashboard del Camarero

- Actualización del panel principal con últimas cuentas pagadas
- Widget de resumen de actividad reciente
- Mejora en la navegación entre secciones

## 3. Mejoras en la Interfaz

### 3.1 Diseño Responsive

- Optimización para dispositivos móviles
- Implementación de tablas responsive
- Mejora en la visualización de datos en pantallas pequeñas

### 3.2 Experiencia de Usuario

- Nuevo sistema de navegación más intuitivo
- Botones de acción contextuales
- Feedback visual mejorado para acciones importante

## 4. Optimizaciones

- Implementación de prepared statements para consultas seguras
- Mejora en el manejo de transacciones para pagos
- Optimización de consultas SQL para mejor rendimiento

## 5. Archivos Modificados

1. camarero/cuenta.php - Nueva implementación
2. camarero/cuentas\_pagadas.php - Nueva implementación
3. camarero/obtener\_detalle\_cuenta.php - Nuevo archivo
4. camarero/index.php - Actualizado con historial
5. BBDD/restaurante.sql - Actualización de estructura

```
-- Crear tabla para almacenar cuentas pagadas
CREATE TABLE `cuentas_pagadas` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `mesa_id` bigint(20) NOT NULL,
  `fecha_hora` datetime DEFAULT CURRENT_TIMESTAMP,
  `producto` varchar(100) NOT NULL,
  `cantidad` int(11) NOT NULL,
  `precio_unitario` decimal(10,2) NOT NULL,
  `subtotal` decimal(10,2) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `mesa_id` (`mesa_id`),
  CONSTRAINT `cuentas_pagadas_ibfk_1` FOREIGN KEY (`mesa_id`) REFERENCES `mesas` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

## Próximos Pasos

1. Implementar sistema de tickets PDF
2. Añadir filtros en el historial de cuentas
3. Implementar composer y añadido para el corte de los tickets
4. Mejorar la gestión de errores
5. Añadir funcionalidad de búsqueda en el historial

## 6. Conclusiones:

Este sprint se ha centrado en mejorar la gestión financiera y el seguimiento de pagos, proporcionando herramientas más robustas para los camareros y una mejor experiencia de usuario.

### **Sprint 4: Mejora del Sistema de Tickets y Manejo de Errores**

#### OBJETIVO

Mejorar el sistema de generación de tickets y el manejo de errores en la impresión de tickets y envío a cocina, implementando un sistema robusto de redirección y notificaciones de errores.

#### *Funcionalidades Implementadas*

#### **1. Sistema de Generación de Tickets Mejorado**

- Implementación de validaciones robustas
- Manejo de errores centralizado
- Redirección inteligente en caso de fallos

#### **2. Integración con Impresora Térmica**

- Timeout en la conexión para evitar bloqueos
- Verificación previa de conexión
- Manejo de errores de impresión

#### **3. Sistema de Notificaciones**

- Mensajes de error específicos
- Alertas visuales en la interfaz
- Persistencia de mensajes entre redirecciones

## CÓDIGO RELEVANTE

### Generación de Ticket (generar\_ticket.php)

```
<?php
try {
    if (!$mesa_id) {
        throw new Exception("ID de mesa no válido");
    }

    // Obtener productos
    $query = "SELECT c.*, p.nombre as nombre_producto
              FROM cuenta c
              INNER JOIN productos p ON c.producto_id = p.id
              WHERE c.mesa_id = ?";

    // ...

    $_SESSION['ticket_data'] = [
        'mesa_numero' => $mesa['numero_mesa'],
        'productos' => $productos,
        'total' => $total,
        'fecha' => date('Y-m-d H:i:s'),
        'mesa_id' => $mesa_id,
        'return_url' => 'cuenta.php'
    ];

} catch (Exception $e) {
    $_SESSION['error_ticket'] = "Error al generar ticket: " . $e->getMessage();
    header("Location: cuenta.php?mesa_id=$mesa_id&status=error");
    exit;
}
```

### Impresión de Ticket (ethernet.php)

```

<?php
try {
    // Verificación de conexión con timeout
    $connector = @fsockopen("10.x.x.x", 9100, $errno, $errstr, 5);

    if (!$connector) {
        throw new Exception("No se pudo conectar con la impresora");
    }

    // Impresión del ticket
    $printer = new Printer($connector);
    $printer->setJustification(Printer::JUSTIFY_CENTER);
    $printer->text("RESTAURANTE CHAMPIÑON\n");
    // ...

} catch (Exception $e) {
    $_SESSION['error_ticket'] = "Error al imprimir: " . $e->getMessage();
    header("Location: $return_url?mesa_id=$mesa_id&status=error");
    exit;
}

```

### Visualización de Errores (cuenta.php)

```

<?php if (isset($error_mensaje)): ?>
    <div class="alert alert-danger alert-dismissible fade show mx-3 mt-3" role="alert">
        <?php echo htmlspecialchars($error_mensaje); ?>
        <button type="button" class="btn-close" data-bs-dismiss="alert"></button>
    </div>
<?php endif; ?>

```

## **MEJORAS TÉCNICAS IMPLEMENTADAS**

### **1. Manejo de Errores**

- Try-catch en operaciones críticas
- Mensajes de error específicos
- Rollback en transacciones fallidas

### **2. Seguridad**

- Validación de datos de entrada
- Escapado de caracteres especiales
- Preparación de consultas SQL

### **3. Experiencia de Usuario**

- Mensajes claros de error
- Redirecciones automáticas
- Alertas visuales informativas

## ***CONCLUSIONES Y BENEFICIOS***

### **1. Mayor Robustez**

- El sistema no se bloquea por errores de impresora
- Manejo gracioso de fallos
- Recuperación automática de errores

## 2. Mejor Experiencia de Usuario

- Feedback inmediato de errores
- Flujo continuo de trabajo
- Mensajes claros y específicos

## 3. Mantenibilidad

- Código centralizado de manejo de errores
- Estructura clara de logs y mensajes
- Facilidad para debugging

### PRÓXIMOS PASOS

- **Desarrollar la funcionalidad para que los encargados puedan agregar, modificar, y eliminar productos en el menú y ajustar el stock**
- **Implementar la capacidad para que el encargado registre, suspenda o elimine perfiles de camareros.**
- **Desarrollar el sistema de generación de informes de rendimiento del restaurante, con opciones de consulta sobre ingresos, pedidos y número de comensales**
- **Proveer acceso al historial de pedidos para auditoría o gestión de incidencias.**

## Sprint 4.5 - Mejoras visuales y funcionalidad PDF

### 1. Unificación del diseño

Se ha implementado una paleta de colores común basada en tonos azules oscuros para mantener consistencia visual en toda la aplicación:

```
:root {  
  --restaurant-primary: #2c3e50; /* Azul oscuro principal */  
  --restaurant-secondary: #34495e; /* Azul oscuro secundario */  
  --restaurant-accent: #3498db; /* Azul claro para acentos */  
  --restaurant-light: #ecf0f1; /* Gris muy claro para fondos */  
  --restaurant-dark: #1a252f; /* Azul muy oscuro */  
}
```

### 2. Mejoras en la interfaz de usuario

#### 2.1. Navegación mejorada

- Implementación de sombras y efectos hover
- Mejora en la visibilidad de elementos activos
- Transiciones suaves para mejor feedback visual

```
.card {  
  border: none;  
  border-radius: 8px;  
  box-shadow: 0 2px 12px rgba(0,0,0,0.1);  
  transition: all 0.3s ease;  
}  
  
.btn-primary {  
  background-color: var(--restaurant-primary);  
  transition: all 0.3s ease;  
}
```



## 2.2. Responsive Design

- Optimización para dispositivos móviles
- Mejoras en la visualización de tablas en pantallas pequeñas
- Ajustes de padding y márgenes para mejor usabilidad

## 3. Implementación de exportación PDF

### 3.1. Instalación de dependencias

**composer require dompdf/dompdf**

### 3.2. Configuración del generador PDF

```
<?php
// generar_ticket_pdf.php
require_once 'vendor/autoload.php';
use Dompdf\Dompdf;
use Dompdf\Options;

$options = new Options();
$options->set('isHtml5ParserEnabled', true);
$dompdf = new Dompdf($options);
```

### 3.3. Botón de descarga PDF

Se agregó el botón en las vistas de cuenta y cuentas pagadas:

```
<a href="descargar_ticket_pdf.php?mesa_id=<?php echo $mesa_id; ?>"
class="btn btn-sm btn-outline-danger">
  <i class="fas fa-file-pdf"></i>
  <span class="btn-text">PDF</span>
</a>
```

### 3.4. Generación del ticket PDF

```
<?php
// descargar_ticket_pdf.php
function generarTicketPDF($datos_mesa, $productos) {
    $html = '<html><head>';
    $html .= '<style>
        body { font-family: Arial, sans-serif; }
        .ticket { max-width: 300px; margin: 0 auto; }
        .header { text-align: center; margin-bottom: 20px; }
        .total { font-weight: bold; margin-top: 20px; }
    </style>';
    $html .= '</head><body>';
    // ... generación del contenido del ticket ...
    $html .= '</body></html>';

    $dompdf->loadHtml($html);
    $dompdf->setPaper('A4', 'portrait');
    $dompdf->render();
    return $dompdf->output();
}
```

## 4. Mejoras en la gestión de cuentas

### 4.1. Vista detallada de cuentas

- Implementación de modal para ver detalles
- Mejora en la presentación de datos
- Funcionalidad de reimpresión de tickets

### 4.2. Historial de cuentas

- Visualización mejorada del historial
- Filtros y ordenación de datos
- Acceso rápido a tickets anteriores

## 5. Optimizaciones generales

### 5.1. Rendimiento

- **Optimización de consultas SQL**
- **Mejora en el tiempo de carga de páginas**
- **Reducción de recursos utilizados**

### 5.2. Usabilidad

- **Mensajes de confirmación más claros**
- **Mejor feedback para acciones del usuario**
- **Navegación más intuitiva**

## 6. Próximos pasos

- **Implementar más opciones de exportación**
- **Mejorar la gestión de errores**
- **Añadir estadísticas y reportes**
- **Optimizar más la experiencia móvil**

## 7. Conclusiones

Las mejoras implementadas en este sprint han resultado en una interfaz más profesional y coherente, además de añadir funcionalidad valiosa como la exportación a PDF. La usabilidad general del sistema ha mejorado significativamente, especialmente en dispositivos móviles.

## Sprint 5: Mejoras en la Interfaz del Panel de Encargado

### 1. Adaptación Responsive del Panel Principal

Se mejoró la interfaz del panel de encargado para garantizar una experiencia de usuario óptima en todos los dispositivos. Se implementaron tarjetas de acción con efectos visuales y transiciones suaves.

```
<?php
// Ejemplo de estilo de tarjetas implementado
.action-card {
    transition: all 0.3s ease;
    border-radius: 10px;
    box-shadow: 0 2px 4px rgba(0,0,0,0.1);
}
.action-card:hover {
    transform: translateY(-3px);
    box-shadow: 0 4px 8px rgba(0,0,0,0.2);
}
```

### 2. Mejora en el Listado de Camareros

Se desarrolló una vista dual para el listado de camareros:

- *Vista de escritorio*: Tabla completa con todos los datos
- *Vista móvil*: Tarjetas individuales con información detallada y acciones accesibles

```
<?php
// Estructura de tarjeta móvil para camareros
<div class="card-mobile">
    <div class="user-info">
        <div class="user-image">
            
        </div>
        <div class="user-details">
            <h6 class="user-name"><?php echo $camarero['nombre']; ?></h6>
            <span class="status-badge"><?php echo $camarero['estado'] ? 'Activo' : 'Inactivo'; ?></span>
        </div>
    </div>
    <!-- Información detallada y botones de acción -->
</div>
```

### 3. Optimización del Formulario de Registro

Se implementó un diseño más intuitivo para el formulario de registro de camareros, con validación de campos y mejor manejo de la carga de imágenes.

### 4. Características Técnicas Implementadas

- **Diseño Responsive:** Uso de Media Queries para adaptación a diferentes dispositivos
- **Efectos Visuales:** Transiciones y animaciones suaves para mejor feedback
- **Gestión de Imágenes:** Manejo optimizado de fotos de perfil
- **Validación de Formularios:** Implementación de validación del lado del cliente
- **Interfaz Intuitiva:** Diseño centrado en la experiencia de usuario

### 5. Aspectos de Seguridad

```
<?php
// Ejemplo de manejo seguro de imágenes
if(isset($_FILES['foto']) && $_FILES['foto']['error'] == 0) {
    $target_dir = "../uploads/";
    $foto = uniqid() . "_" . basename($_FILES["foto"]["name"]);
    $target_file = $target_dir . $foto;

    if (move_uploaded_file($_FILES["foto"]["tmp_name"], $target_file)) {
        // Imagen procesada correctamente
    }
}
```

### 6. Mejoras en la Experiencia de Usuario

- Botones de acción más grandes en móvil
- Información claramente estructurada
- Feedback visual en interacciones
- Navegación simplificada
- Acciones críticas con confirmación

## 7. Conclusiones

Las mejoras implementadas en este sprint han resultado en una interfaz más moderna, accesible y funcional para el panel de encargado, facilitando la gestión eficiente de camareros tanto en dispositivos de escritorio como móviles. Se ha modificado el registro para poder subir una imagen y visualizarla junto con todos los datos de los camareros.

### Sprint 6: Añadir mejoras y gestión de productos en (encargado)

#### Introducción

En el Sprint 6 del proyecto “Restaurante”, se implementaron diversas mejoras y nuevas funcionalidades para optimizar la gestión de usuarios y productos, así como para mejorar la interfaz de usuario. A continuación, se detallan los cambios realizados en los diferentes archivos del sistema.

#### Objetivos del Sprint 6

- **Mejorar la gestión de usuarios** añadiendo roles y funcionalidades de administración.
- **Implementar la carga y gestión de imágenes** para usuarios.
- **Optimizar la interfaz de usuario** para ser más responsiva y amigable.
- **Mejorar la gestión de productos** con funcionalidades completas de CRUD (Crear, Leer, Actualizar, Eliminar).
- **Aumentar la seguridad** en las operaciones de base de datos mediante el uso de consultas preparadas.

#### Cambios Realizados

## 1. Registro de Usuarios

### Archivos Modificados:

- encargado/registro.php
- encargado/registrar\_usuario.php

### Descripción de Cambios:

- **Añadido Campo 'Rol':** Se incorporó un nuevo campo rol en el formulario de registro para asignar roles específicos a los usuarios (camarero, encargado).
- **Carga de Fotos de Usuario:** Se implementó la funcionalidad para subir imágenes de perfil de los usuarios, almacenándolas en la carpeta uploads.

### Código Relevante:

```
<?php
// archivo: /c:/xampp/htdocs/Restaurante/encargado/registro.php
$rol = $_POST['rol']; // Nuevo campo para el rol

// Procesar la imagen
$foto = null;
if(isset($_FILES['foto']) && $_FILES['foto']['error'] == 0) {
    $target_dir = "../uploads/";
    if (!file_exists($target_dir)) {
        mkdir($target_dir, 0777, true);
    }
    // ...código para manejar la subida de la imagen...
}
```

## 2. Listado y Gestión de Usuarios

### Archivos Modificados:

- encargado/listar\_usuarios.php
- encargado/gestionar\_usuario.php

#### *Descripción de Cambios:*

- **Filtrado por Rol:** Se añadió la posibilidad de filtrar usuarios por rol, permitiendo una gestión más específica.
- **Exclusión del Usuario Principal:** Se excluye al usuario con id = 1 de las operaciones de modificación y eliminación para proteger el usuario administrador principal.
- **Confirmaciones Modales:** Implementación de modales de confirmación para acciones críticas como suspender, activar o eliminar usuarios.

#### *Código Relevante:*

```
<?php
// archivo: /c:/xampp/htdocs/Restaurante/encargado/listar_usuarios.php
// Modificar la consulta para excluir al usuario con ID 1 y filtrar por rol si está definido
if ($rol) {
    $rol = mysqli_real_escape_string($conexion, $rol);
    $sql = "SELECT * FROM usuarios WHERE rol = '$rol' AND id != 1";
} else {
    $sql = "SELECT * FROM usuarios WHERE rol IN ('camarero', 'encargado') AND id != 1";
}
```

### 3. Gestión de Productos

#### *Archivos Modificados:*

- encargado/listar\_productos.php
- encargado/agregar\_producto.php
- encargado/modificar\_producto.php
- encargado/eliminar\_producto.php
- encargado/gestionar\_producto.php



*Descripción de Cambios:*

- **CRUD Completo:** Se implementaron todas las operaciones de CRUD para la gestión de productos, permitiendo añadir, listar, modificar y eliminar productos.
- **Filtrado por Categoría:** Se añadió la funcionalidad de filtrar los productos por categoría para facilitar la búsqueda y organización.
- **Interfaz Responsiva:** Se mejoró la visualización de las tablas de productos en dispositivos móviles mediante estilos CSS específicos.

*Código Relevante:*

```
<?php
// archivo: /c:/xampp/htdocs/Restaurante/encargado/listar_productos.php
<form method="GET" action="">
  <div class="form-group">
    <label for="categoria">Filtrar por Categoría:</label>
    <select name="categoria" id="categoria" class="form-control" onchange="this.form.submit()">
      <option value="">Todas</option>
      <?php
        $categorias = mysqli_query($conexion, "SELECT DISTINCT categoria FROM productos");
        while($categoria = mysqli_fetch_assoc($categorias)) {
          $selected = (isset($_GET['categoria']) && $_GET['categoria'] == $categoria['categoria'])
          echo "<option value='{$categoria['categoria']}' {$selected}>{$categoria['categoria']}</
        }
      ?>
    </select>
  </div>
</form>
```

## 4. Interfaz Responsiva

*Archivos Modificados:*

- encargado/registrar\_usuario.php
- encargado/listar\_usuarios.php
- encargado/listar\_productos.php
- encargado/index.php

*Descripción de Cambios:*

- **Medias Queries en CSS:** Se añadieron reglas CSS para mejorar la visualización en dispositivos móviles, ajustando tamaños, márgenes y disposición de elementos.
- **Tarjetas Interactivas:** Se implementaron tarjetas con efectos hover para una mejor experiencia de usuario en la página principal del encargado.

*Código Relevante:*

```

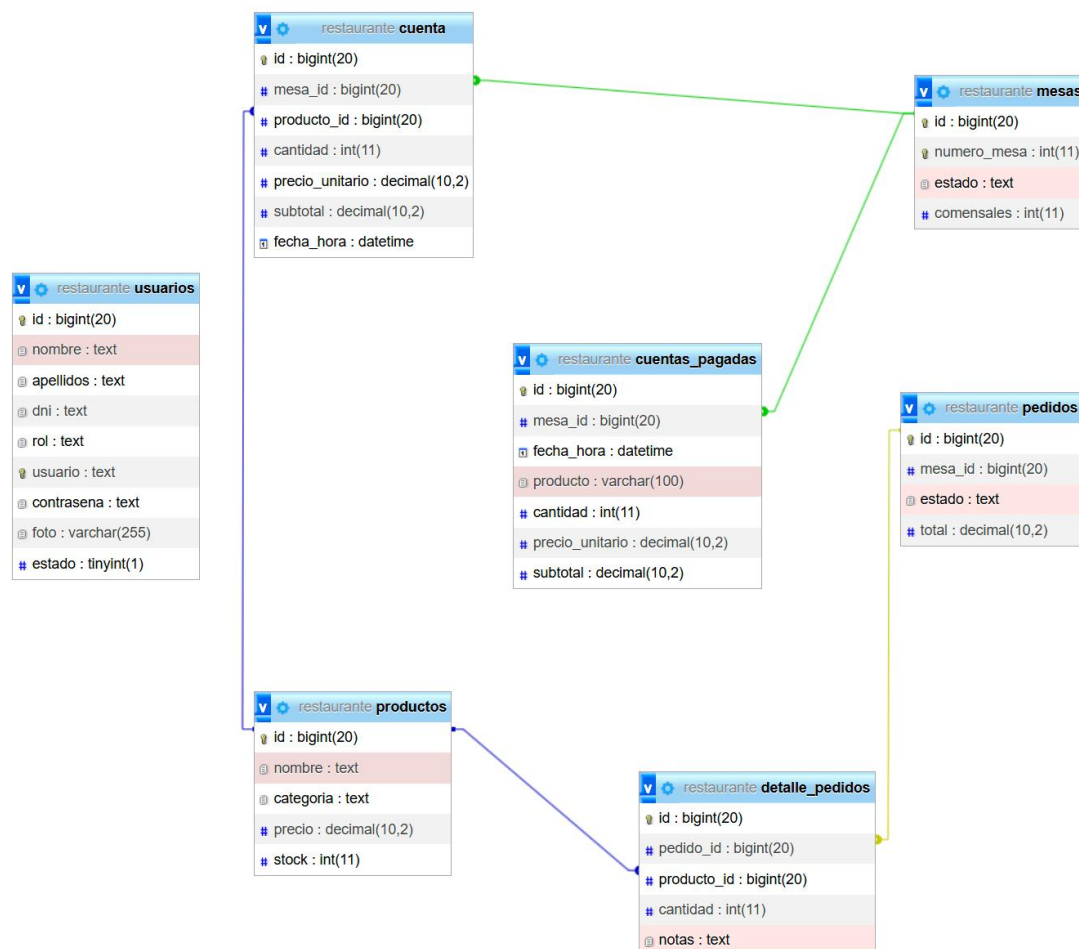
/* archivo: /c:/xampp/htdocs/Restaurante/encargado/index.php */
/* Estilos para tarjetas interactivas */
@media (max-width: 768px) {
  .container {
    padding: 10px;
  }
  .btn-block {
    width: 100%;
    margin-bottom: 10px;
  }
  h1 {
    font-size: 1.5rem;
    margin-bottom: 20px;
  }
}
/* Estilos de las tarjetas */
.action-card {
  transition: all 0.3s ease;
  margin-bottom: 1rem;
  border-radius: 10px;
  box-shadow: 0 2px 4px rgba(0,0,0,0.1);
}
.action-card:hover {
  transform: translateY(-3px);
  box-shadow: 0 4px 8px rgba(0,0,0,0.2);
}

```

## Conclusiones

Durante el Sprint 6, se lograron significativas mejoras en la gestión de usuarios y productos, así como en la experiencia de usuario a través de una interfaz más responsiva y funcional. La implementación de roles y la seguridad en las operaciones garantizan un control adecuado y protegido del sistema, mientras que las nuevas funcionalidades de gestión de productos permiten una administración más eficiente del inventario del restaurante.

## BASE DE DATOS FINAL:



**Tablas:**

- **restaurante\_mesas:** Almacena información sobre las mesas del restaurante, incluyendo su ID, número, estado (libre, ocupada, etc.) y capacidad de comensales.
- **restaurante\_productos:** Contiene información sobre los productos que ofrece el restaurante, como su ID, nombre, categoría (bebidas, entrantes, platos principales, postres), precio y stock disponible.
- **restaurante\_usuarios:** Guarda información sobre los usuarios del sistema, que podrían ser empleados del restaurante. Incluye datos como ID, nombre, apellidos, DNI, rol (camarero, cocinero, administrador, etc.), usuario, contraseña, foto y estado (activo, inactivo).
- **restaurante\_pedidos:** Almacena información sobre los pedidos realizados, incluyendo ID, mesa asociada, estado (en proceso, servido, pagado, etc.) y total a pagar.
- **restaurante\_detalle\_pedidos:** Describe los productos que componen cada pedido, relacionando el ID del pedido con el ID del producto y la cantidad. También podría incluir notas sobre el producto (como preferencias de cocción o alergias).
- **restaurante\_cuenta:** Parece ser una tabla intermedia que relaciona los pedidos con los productos y sus cantidades, precios unitarios y subtotales. Es posible que esta tabla se pueda simplificar o integrar con restaurante\_detalle\_pedidos, dependiendo de la lógica del sistema.
- **restaurante\_cuentas\_pagadas:** Almacena información sobre las cuentas que ya han sido pagadas, incluyendo la mesa, fecha y hora de pago, producto, cantidad, precio unitario y subtotal.

**Relaciones:**

Las líneas entre las tablas representan las relaciones entre ellas. Estas relaciones se basan en claves foráneas que vinculan los datos de una tabla con otra. Por ejemplo:

- **restaurante\_pedidos** tiene una relación con **restaurante\_mesas**, lo que significa que cada pedido está asociado a una mesa específica.
- **restaurante\_detalle\_pedidos** tiene una relación con **restaurante\_pedidos** y **restaurante\_productos**, indicando que cada detalle del pedido se refiere a un producto específico dentro de un pedido particular.
- **restaurante\_cuenta** se relaciona con **restaurante\_mesas**, **restaurante\_productos** y **restaurante\_pedidos**, lo que sugiere que registra información sobre los productos pedidos en cada mesa.
- **restaurante\_cuentas\_pagadas** se relaciona con **restaurante\_mesas** y **restaurante\_cuenta**, indicando que almacena información de las cuentas pagadas asociadas a una mesa.

## GUIA DE USO:

### Introducción

Bienvenido a la guía de uso de la página web del restaurante. Aquí aprenderás a gestionar mesas, pedidos y productos de manera sencilla, con herramientas diseñadas tanto para camareros como para encargados.

### ¿Qué incluye esta guía?

- Organización de mesas y asignación de productos
- Gestión de pedidos y actualizaciones en tiempo real.
- Control de productos e inventario.
- Funciones específicas para camareros y encargados.

### Pantalla de inicio del portal:

Desde aquí podemos acceder a la web con nuestros usuarios, tenemos 2 preestablecidos:

1-Encargado:

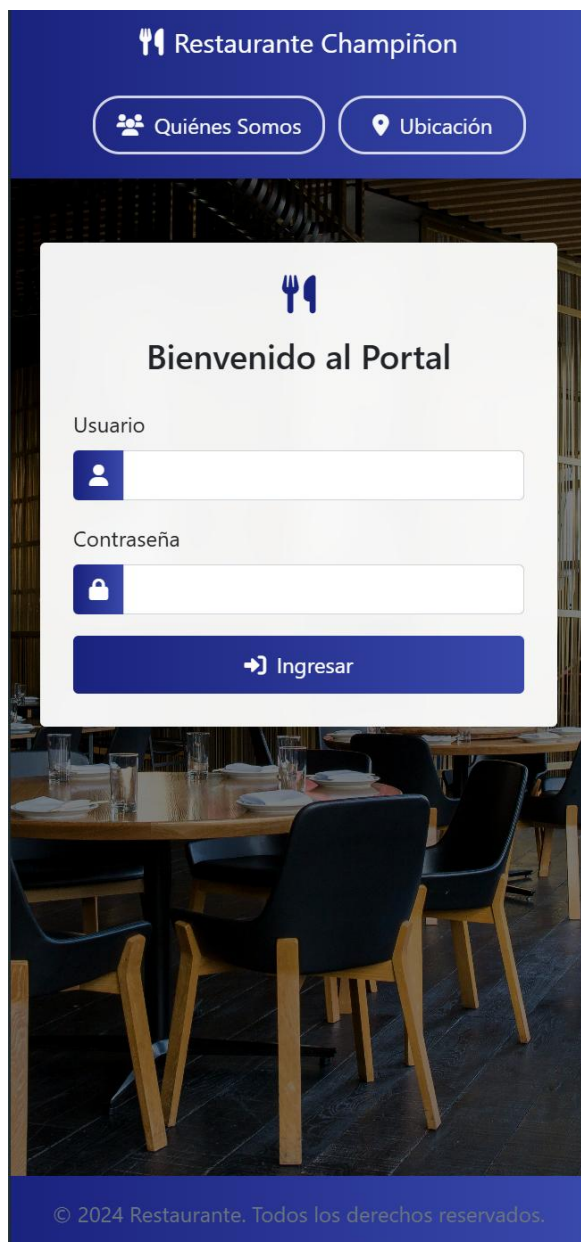
Usuario: master

Contraseña: pizza

2-Camarero:

Usuario: periko\_elmaki

Contraseña: 1234



## SECCION DE ENCARGADO:

### Index.php(encargado)

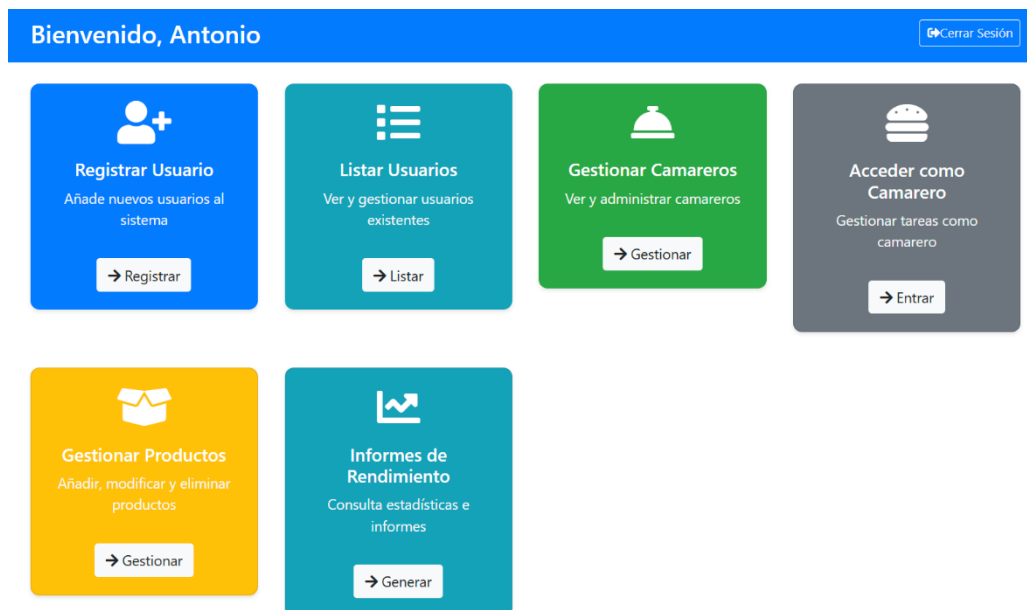
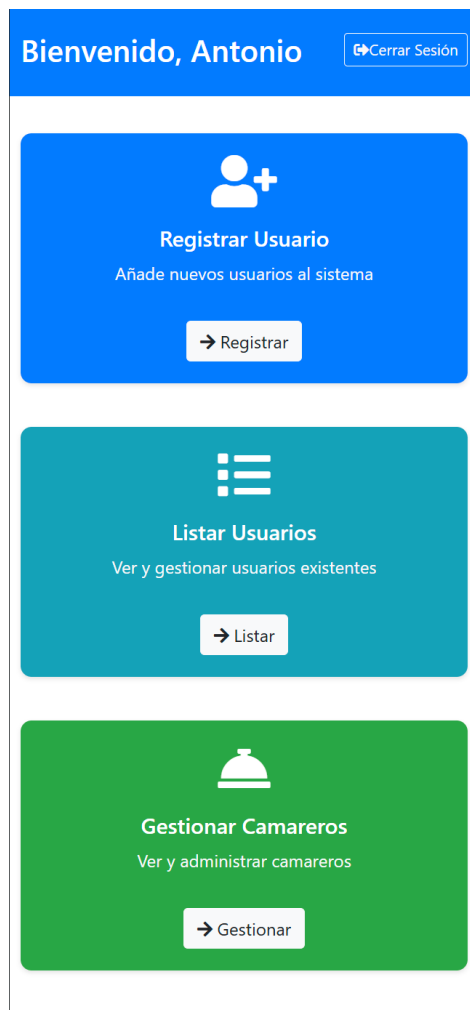
Es el panel de control de los encargados (solo pueden acceder con un usuario encargado).

Tenemos varias formas de visualizar siempre la web, con tamaño adaptado para móvil responsive, en una Tablet o monitor.

Encontramos 5 secciones en el panel:

1. Registrar usuario
2. Listar Usuarios
3. Gestionar camareros
4. Acceder como Camarero
5. Gestionar Productos
6. Informe de Rendimiento

A continuación, detallamos las secciones:



## Registrar Usuario:

Página para registrar un nuevo usuario y asignarle un

Rol: encargado/camarero


## Listar Usuarios:

Forma de visualizar los usuarios de la base de datos de restaurante.

Bienvenido, Antonio

Volver al Panel

Listado de Usuarios



**Pedro**  
Activo

Apellidos: Salvador  
DNI: 23445667F  
Usuario: periko\_elmaki  
Contraseña: 1234

Suspender
Eliminar

Bienvenido, Antonio

Volver al Panel

Registrar Nuevo Usuario

Nombre:   
Apellido:   
DNI:   
Usuario:   
Contraseña:   
Rol: 

Seleccione un rol

  
Foto: 

Seleccionar archivo

Ningún archivo seleccionado

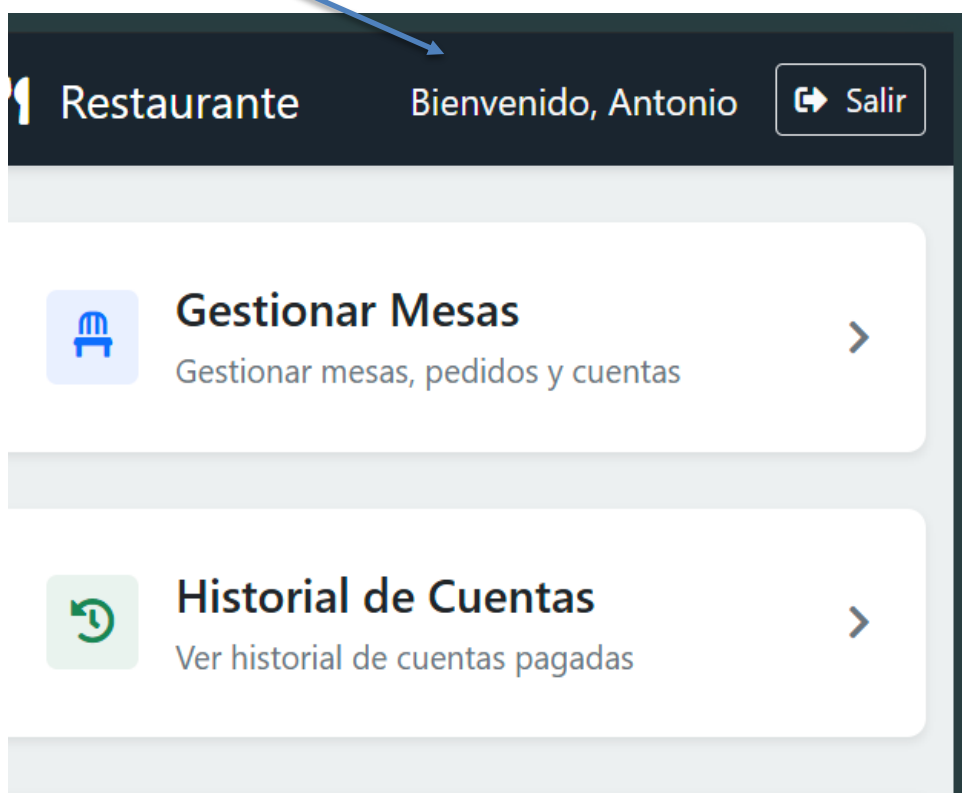
Seleccione una imagen para el usuario (opcional)

Registrar

### Acceder como Camarero:

Desde encargado también podemos acceder a la parte de camarero y poder utilizar todas sus secciones.

Tu nombre aparecerá en el panel inicial de camarero.





## Gestionar Productos:

Apartado en el cuál puedes añadir, modificar o eliminar productos de la base de datos.

### Listado de Productos

[+ Añadir Producto](#)[← Volver](#)

Filtrar por Categoría:

Todas

ID	Nombre	Categoría	Precio	Stock	Acciones
1	Coca Cola	Bebida	2.50	20	<a href="#">✎ Modificar</a> <a href="#">🗑 Eliminar</a>
2	Pizza Margarita	pizzas	8.00	10	<a href="#">✎ Modificar</a> <a href="#">🗑 Eliminar</a>
3	Pizza 4 Quesos	pizzas	9.50	10	<a href="#">✎ Modificar</a> <a href="#">🗑 Eliminar</a>
4	Pizza Napolitana	pizzas	9.00	10	<a href="#">✎ Modificar</a> <a href="#">🗑 Eliminar</a>
5	Pizza Pepperoni	pizzas	9.50	10	<a href="#">✎ Modificar</a> <a href="#">🗑 Eliminar</a>
6	Pizza Hawaiana	pizzas	9.00	10	<a href="#">✎ Modificar</a> <a href="#">🗑 Eliminar</a>
7	Pizza Vegetariana	pizzas	8.50	10	<a href="#">✎ Modificar</a> <a href="#">🗑 Eliminar</a>
8	Pizza Barbacoa	pizzas	10.00	10	<a href="#">✎ Modificar</a> <a href="#">🗑 Eliminar</a>
9	Pizza Prosciutto	pizzas	9.00	10	<a href="#">✎ Modificar</a> <a href="#">🗑 Eliminar</a>
10	Pizza Diavola	pizzas	9.50	10	<a href="#">✎ Modificar</a> <a href="#">🗑 Eliminar</a>
11	Ensalada Griega	ensalada	6.50	15	<a href="#">✎ Modificar</a> <a href="#">🗑 Eliminar</a>
12	Ensalada Caprese	ensalada	7.00	15	<a href="#">✎ Modificar</a> <a href="#">🗑 Eliminar</a>
13	Ensalada de Pollo	ensalada	7.50	15	<a href="#">✎ Modificar</a> <a href="#">🗑 Eliminar</a>

## Generar Informe:

Puedes seleccionar la fecha entre los días que quieras conocer la cantidad total de ingresos y la cantidad de pedidos.

**Generar Informe**

← Volver

Fecha Inicio:

Fecha Fin:

Generar Informe

## Informe de ventas:

Aquí se visualiza el informe con la fecha, el número de ventas y el total de ingreso ese día.

Informe de Ventas		
Período: 19/11/2024 - 20/11/2024		
Fecha	Número de Ventas	Total del Día
20/11/2024	14	164.00€
<b>TOTALES</b>	<b>14</b>	<b>164.00€</b>

Volver

Imprimir

## SECCION DE CAMARERO:

**Index.php (camarero):**


Panel de control de la página de camarero.

*Gestionar Mesas:*

- Activar mesas
- Gestionar pedidos
- Cuentas


*Historial de Cuentas:*

**Página donde poder acceder a las cuentas pagadas anteriores ordenadas por fechas.**


Restaurante
Bienvenido, Pedro
Salir



**Gestionar Mesas**
Gestionar mesas, pedidos y cuentas



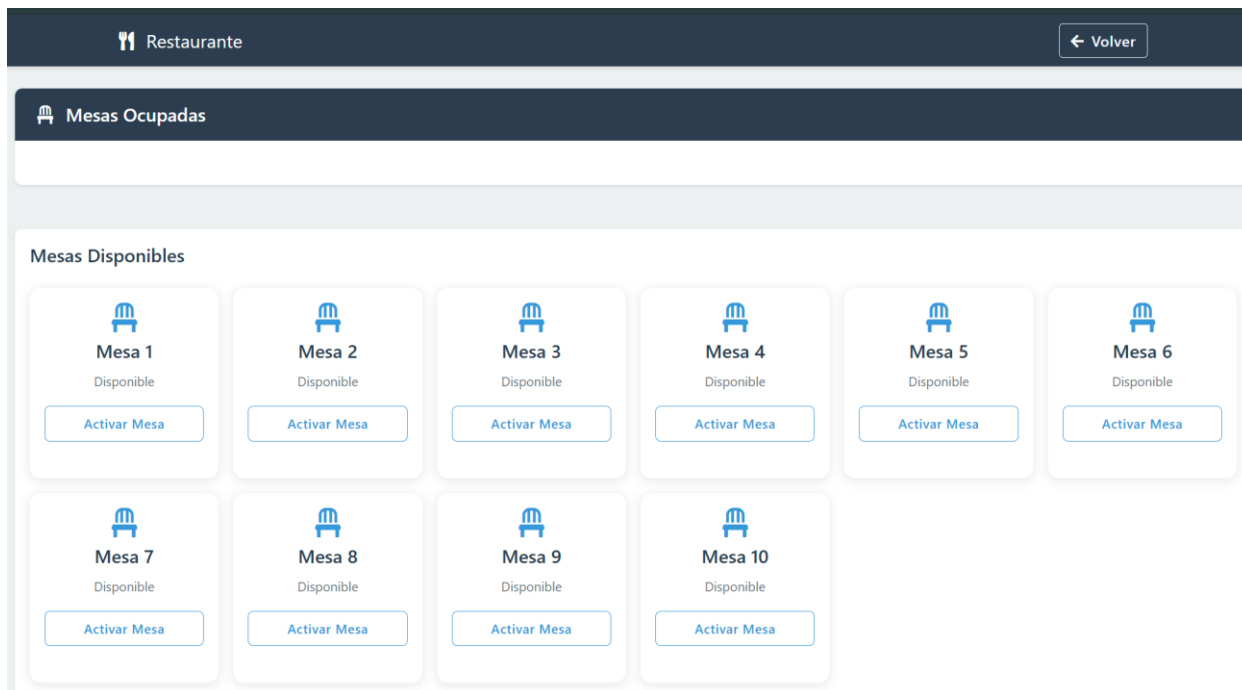
**Historial de Cuentas**
Ver historial de cuentas pagadas


**Últimas Cuentas Pagadas**

Mesa 1	15/11/2024 16:36	4 productos	22.50€
Mesa 1	15/11/2024 16:32	2 productos	17.50€
Mesa 1	15/11/2024 16:18	1 productos	2.50€
Mesa 1	15/11/2024 16:18	6 productos	50.00€
Mesa 1	15/11/2024 16:01	3 productos	7.50€
Mesa 1	15/11/2024 14:18	3 productos	6.50€

## Gestionar mesas:

Sección para visualizar las mesas y activarlas para poder gestionar el pedido.



Al activar una mesa, esa mesa pasará a ser “ocupada” y desde el menú de las mesas se podrá entrar en varias secciones como:

- *Gestionar* (para agregar productos)
- *Cuenta* (acceder a la cuenta de la mesa)
- *Cerrar* (posibilidad de cerrar la mesa, si no hay productos en ella)



## Gestionar pedido:

Restaurante Mesa 1 Cuenta Volver

Añadir Productos Desliza para más categorías

Bebida Carne Ensalada Extras Pasta Pescado Pizzas Vino

Coca Cola 2.50€ +

Fanta Naranja 2.50€ +

Fanta Limón 2.50€ +

Sprite 2.50€ +

Agua Mineral 1.50€ +

Cerveza 3.00€ +

Coca Cola zero 2.50€ +

Productos en el Pedido

Producto	Cant.	Precio	Subtotal	Notas	Acciones
Entrecot de Ternera	1	15.00€	15.00€	al punto	Modificar Eliminar
Cerveza	3	3.00€	9.00€		Modificar Eliminar
Total:		24.00€			

Enviar a Cocina

Aquí podrás seleccionar los productos a la mesa y que posteriormente se envían a la cocina para su preparación.

Los productos están organizados por categorías según su tipo y puede seleccionar la cantidad y añadir una nota cuando seleccione el producto.

Se puede modificar y eliminar los productos en el pedido si hiciera falta y visualizar lo que hay en ese momento.

Al pulsar "Enviar a Cocina" los productos pasarán a la cuenta, pero antes se generará un ticket a modo de comanda para la cocina con la cantidad y notas si las hubiera.



**Cuenta:**

Tras pulsar en “Enviar a Cocina” llegamos a *la cuenta* donde visualizamos los productos que llevamos.

Posibilidad de modificar y eliminar productos de la cuenta.

Aquí se puede imprimir un ticket con la cuenta que hay en ese momento, aunque una vez que pulsemos en “Procesar pago”, también nos genera el ticket de la cuenta con todos sus detalles.

**Cuenta Mesa 1** Ticket Cuenta

Producto	Cantidad	Precio	Subtotal
Entrecot de Ternera	1	15.00€	15.00€
Cerveza	3	3.00€	9.00€
<b>Total:</b>			<b>24.00€</b>

Procesar Pago

Antes de procesar el pago, se pide confirmación:

**Confirmar Pago**

Total a pagar:

**24.00€**

Cancelar Pagar

**Total: 24.00€**

Procesar Pago

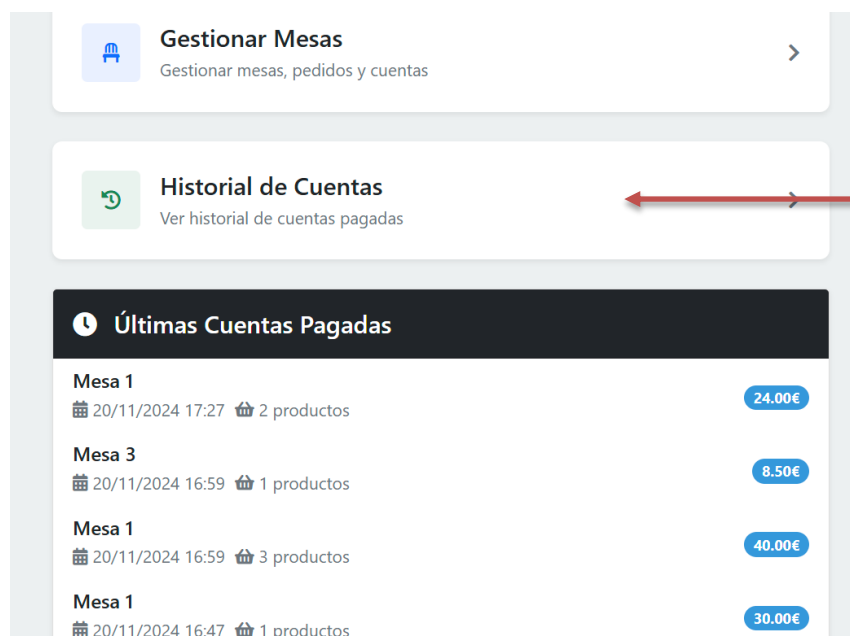
Se genera un ticket de la cuenta con todo lo que se encontraba en ella en el momento de procesar el pago.

También se libera la mesa para poder activarla de nuevo con un pedido nuevo.

Una que ya hemos completado todo el proceso con la mesa y la cuenta, solo queda visualizar el apartado de *historial de cuentas* para ver todas las cuentas anteriores con sus fechas correspondientes.



Entramos en la sección en el panel principal del camarero en *Historial de Cuentas*



## Historial de cuentas pagadas:

Sección en la que se visualizan todas las cuentas por fechas y número de mesa y en la que se pueden realizar ciertas acciones:

Historial de Cuentas Pagadas						
Mesa	Fecha	Hora	Productos	Total	Acciones	
Mesa 1	20/11/2024	17:27	2 productos	24.00€	<a href="#">Ver Detalle</a>	<a href="#">Reimprimir</a> <a href="#">PDF</a>
Mesa 3	20/11/2024	16:59	1 productos	8.50€	<a href="#">Ver Detalle</a>	<a href="#">Reimprimir</a> <a href="#">PDF</a>
Mesa 1	20/11/2024	16:59	3 productos	40.00€	<a href="#">Ver Detalle</a>	<a href="#">Reimprimir</a> <a href="#">PDF</a>
Mesa 1	20/11/2024	16:47	1 productos	30.00€	<a href="#">Ver Detalle</a>	<a href="#">Reimprimir</a> <a href="#">PDF</a>
Mesa 3	20/11/2024	16:38	1 productos	6.00€	<a href="#">Ver Detalle</a>	<a href="#">Reimprimir</a> <a href="#">PDF</a>
Mesa 10	20/11/2024	15:03	2 productos	24.50€	<a href="#">Ver Detalle</a>	<a href="#">Reimprimir</a> <a href="#">PDF</a>
Mesa 1	20/11/2024	14:15	3 productos	26.50€	<a href="#">Ver Detalle</a>	<a href="#">Reimprimir</a> <a href="#">PDF</a>
Mesa 2	20/11/2024	14:13	3 productos	28.50€	<a href="#">Ver Detalle</a>	<a href="#">Reimprimir</a> <a href="#">PDF</a>

- *Ver Detalle:*

Aquí surgirá una ventana (modal) para ver en detalle todo lo que contiene esa cuenta.

Detalle de Cuenta					
Producto	Cant.	Precio	Subtotal		
Entrecot de Ternera	1	15.00€	15.00€		
Cerveza	3	3.00€	9.00€		
Total:			24.00€		



- Reimprimir:

Opción para volver a generar un ticket de cuenta en formato de COPIA

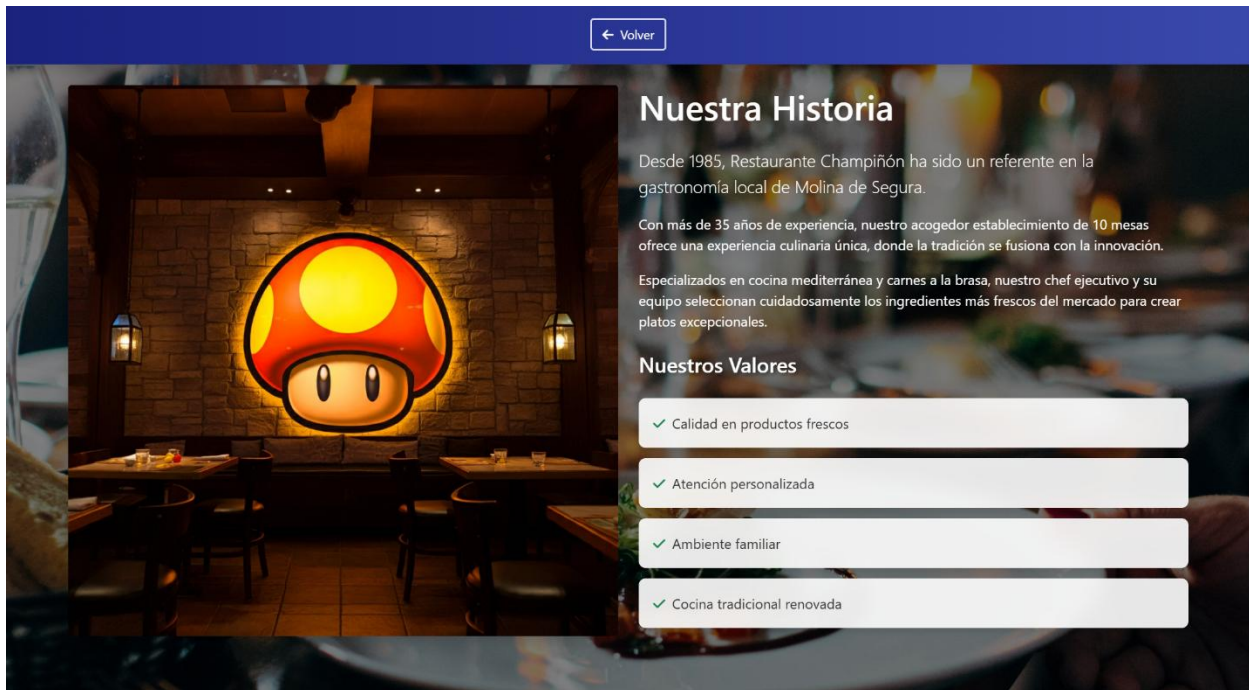


- PDF:

Generación de un PDF con los datos de la cuenta en tablas para su visualización o posterior impresión.

RESTAURANTE CHAMPIÑÓN			
C/ Example, 123 - Ciudad			
Tel: 912345678 - CIF: B12345678			
TICKET DE VENTA			
Mesa: 1 - Fecha: 20/11/2024 17:27			
Producto	Cant.	Precio	Subtotal
Entrecot de Ternera	1	15.00 €	15.00 €
Cerveza	3	3.00 €	9.00 €
TOTAL:			24.00 €

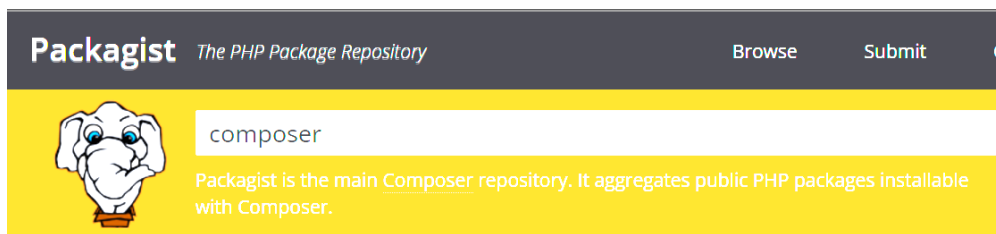
Guía terminada sobre el uso de la página web de Restaurante.



Instalación de impresoras de tickets y PrintNode (para su uso a través de web)

Instalación local y conexión con composer y librerías mike42:

Descargar composer en nuestro proyecto.



**composer/composer**

Composer helps you declare, manage and install dependencies of PHP projects. It ensures you have the right stack everywhere.

PHP 146 212 867

★ 28 699

P  
—  
1  
a  
a

Copiar la ruta resaltada.

## composer/composer

↓ `composer require composer/composer`

*Composer helps you declare, manage and install dependencies of PHP projects. It ensures you have the right stack everywhere.*

Una de las formas más rápidas de hacer con visual studio code sería accediendo a nuestro terminal (control-ñ) y pegando la ruta anterior.

```

80  $pdf->Ln(5);
81
82  // Cabecera de la tabla

```

PROBLEMAS   SALIDA   CONSOLA DE DEPURACIÓN   TERMINAL   PUERTOS   COMENTARIOS

```

PS C:\xampp\htdocs\Restaurante> composer require composer/composer

```

Buscar en la página oficial de composer la librería de mike42/escpos-php

## mike42/escpos-php

↓ `composer require mike42/escpos-php`

*PHP receipt printer library for use with ESC/POS-compatible thermal and impact printers*

Mismo proceso que con la librería de composer:

```
2 // Cabecera de la tabla
```

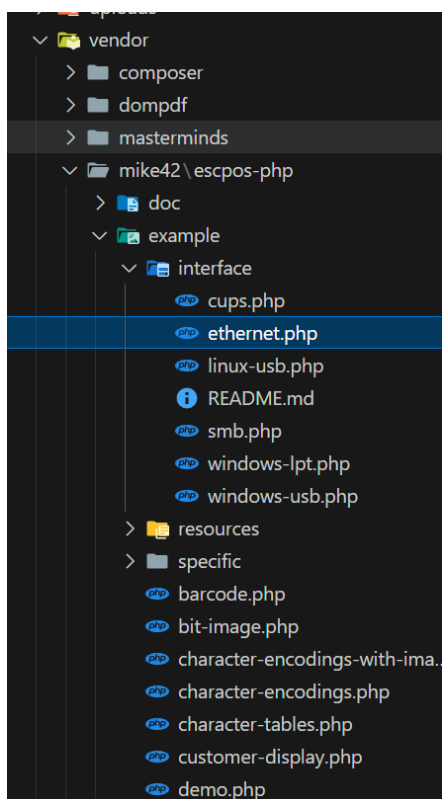
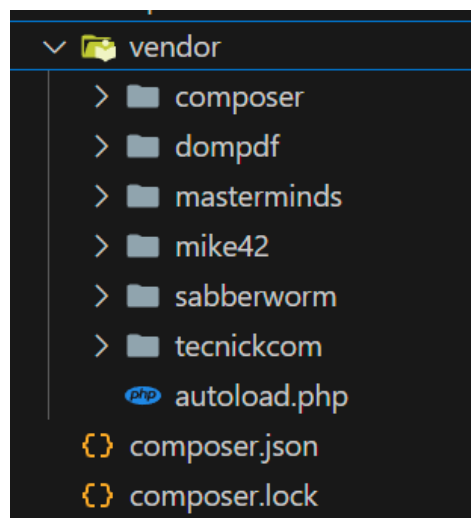
OBLEMAS   SALIDA   CONSOLA DE DEPURACIÓN   TERMINAL   PUERTOS   COMENTARIOS

```
C:\xampp\htdocs\Restaurante> composer require mike42/escpos-php
```

Se nos creará en la raíz de nuestro proyecto 1 carpeta (Vendor) y dos archivos de composer:

Dentro de la carpeta de vendor se encuentra la carpeta de mike42 con todo lo necesario para conectar la impresora de ticket.

Accedemos a ella y podemos encontrar ejemplos de varias formas de conectar la impresora:



En mi caso yo he utilizado la conexión por IP, y he creado un php específico para la impresión de ticket.

Así sería uno del formato para la conexión con la impresora y estructura del ticket.

```

1  <?php
2  include '../sesion.php';
3  include '../conexion.php';
4  require_once '../vendor/autoload.php';
5
6  use Mike42\Escpos\Printer;
7  use Mike42\Escpos\PrintConnectors\NetworkPrintConnector;
8  use Mike42\Escpos\EscposImage;
9
10 $mesa_id = isset($_GET['mesa_id']) ? (int)$_GET['mesa_id'] : null;
11
12 try {
13     // Obtener datos de la cuenta
14     $query = "SELECT c.*, p.nombre as nombre_producto
15             FROM cuenta c
16             INNER JOIN productos p ON c.producto_id = p.id
17             WHERE c.mesa_id = ?";
18
19     $stmt = mysqli_prepare(mysql: $conexion, query: $query);
20     mysqli_stmt_bind_param(statement: $stmt, types: "i", var: &$mesa_id);
21     mysqli_stmt_execute(statement: $stmt);
22     $cuenta_result = mysqli_stmt_get_result(statement: $stmt);
23
24     // Configurar impresora - Usar conexión de red
25     $ipImpresora = "192.168.0.207"; // Cambiar a la IP de tu impresora
26     $puertoImpresora = 9100; // Puerto por defecto para impresoras ESC/POS
27     $connector = new NetworkPrintConnector($ipImpresora, $puertoImpresora);
28     $printer = new Printer($connector);

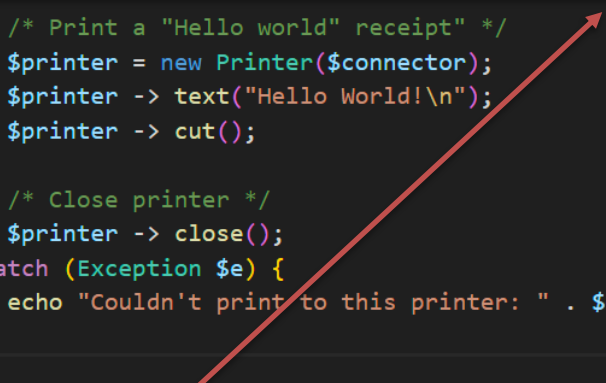
```

También se puede conectar mediante USB añadiendo estos cambios:

```

1  <?php
2  /* Change to the correct path if you copy this example! */
3  require_once __DIR__ . '/../../../../autoload.php';
4  use Mike42\Escpos\Printer;
5  use Mike42\Escpos\PrintConnectors\WindowsPrintConnector;
6
7  /**
8   * Install the printer using USB printing support, and the "Generic / Text Only" driver,
9   * then share it (you can use a firewall so that it can only be seen locally).
10   *
11   * Use a WindowsPrintConnector with the share name to print.
12   *
13   * Troubleshooting: Fire up a command prompt, and ensure that (if your printer is shared as
14   * "Receipt Printer), the following commands work:
15   *
16   * echo "Hello World" > testfile
17   * copy testfile "\\%COMPUTERNAME%\Receipt Printer"
18   * del testfile
19   */
20  try {
21      // Enter the share name for your USB printer here
22
23      $connector = new WindowsPrintConnector("Receipt Printer");
24
25      /* Print a "Hello world" receipt */
26      $printer = new Printer($connector);
27      $printer -> text("Hello World!\n");
28      $printer -> cut();
29
30      /* Close printer */
31      $printer -> close();
32  } catch (Exception $e) {
33      echo "Couldn't print to this printer: " . $e -> getMessage() . "\n";
34  }
35

```



Aquí se debe introducir el puerto de la impresora que generalmente es "USB001" si no existe otra impresora conecta.

## Instalación web con PrintNode:


<https://www.printnode.com/es>

## Descargas

Para utilizar su impresora con PrintNode, deberá instalar nuestro software Cliente en una computadora que tenga acceso a la impresora. Cuando esté instalado, el Cliente de PrintNode detectará automáticamente cualquier impresora conectada a su computadora y la sincronizará con su Cuenta de PrintNode.

Los requisitos del sistema son mínimos. El Cliente de PrintNode podrá ejecutarse en cualquier equipo capaz de funcionar con los sistemas operativos compatibles. El Cliente de PrintNode también podrá ejecutarse en un dispositivo de bajo consumo, como el Raspberry Pi; aunque, al imprimir grandes documentos con dispositivos de bajo consumo, puede notarse algo de retraso.

## Últimas versiones del cliente

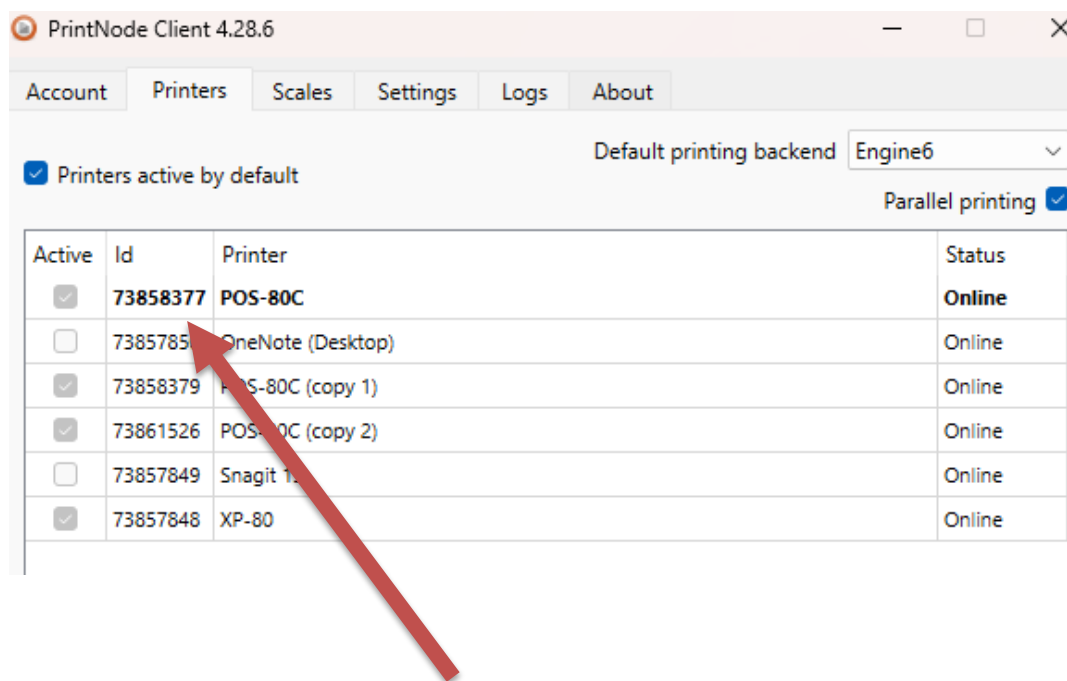


**PrintNode-4.28.6.exe**  
 Publicado: 2024-10-31  
 SHA1: 8eed3a49695cd4c8ba7ae477f425302101e7ad59  
 96.3 MB

Sistemas operativos compatibles  
 Windows 10 y Windows 11  
 (NOTA: puede descargar PrintNode para versiones antiguas de Windows en nuestra [aplicación web](#))

DESCARGAR

Una vez instalado, localizará todas las impresoras conectadas:



Lo importante aquí será guardarnos el "Id" de la impresora activa que nos interesa.

Para instalar la librería de PrintNode en php la podemos instalar con composer buscando el paquete en la página oficial:

## printnode/printnode-php

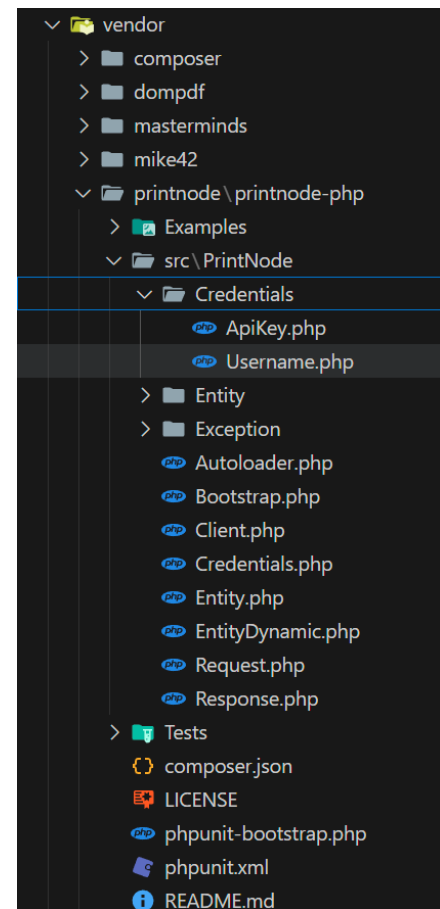
↓ composer require printnode/printnode-php

There is no license information available for the latest version (2.1.0) of this package.

*Connect any printer to your application with PrintNode Client and easy to use JSON API*

<https://packagist.org/packages/printnode/printnode-php>

Se instalará la librería necesaria para la conexión y debemos acceder dentro de ella a la carpeta "*Credentials*" y allí debemos introducir nuestros datos de usuario el cual nos dan al registrarnos en PrintNode.



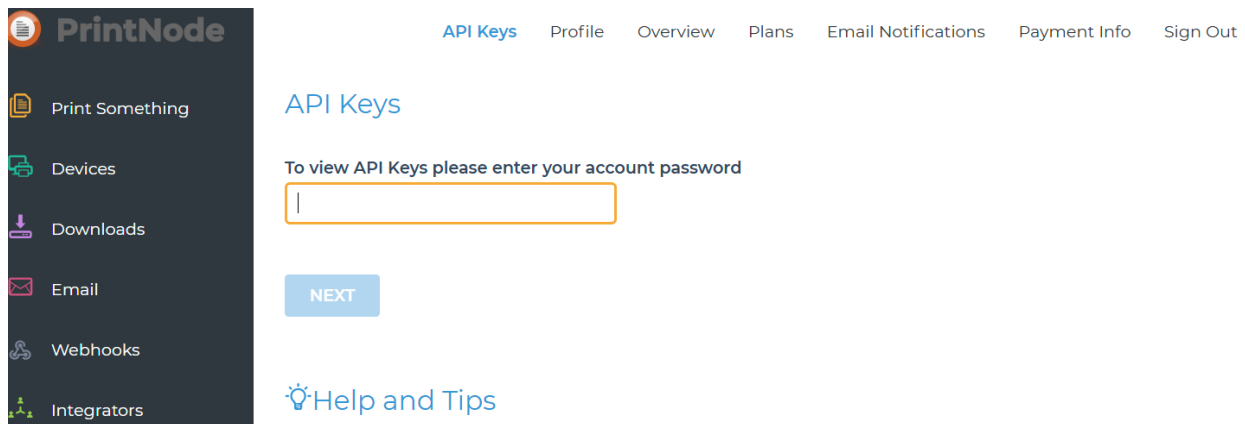
Dentro de *Username.php* tenemos que cambiar esta parte:

```
$this->username = 'ibanez.1314@gmail.com'; // Aquí debes poner tu username
$this->password = ' '; // Aquí debes poner tu password
```

```
}
```



En la web de PrintNode dentro de nuestra cuenta, tenemos varias opciones, pero la más importante es la de generar una API key para conectar con nuestra impresora a través de php.



Una vez que la tengamos creada, en nuestro entorno de desarrollo debemos crear un archivo para la conexión con la impresora a través del módulo de PrintNode:

Un ejemplo de donde debería colocarse la [API](#) sería así:

```
use PrintNode\Credentials\ApiKey;
use PrintNode\Client;
use PrintNode\Entity\PrintJob;

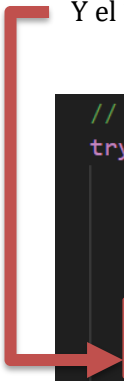
$mesa_id = isset($_GET['mesa_id']) ? (int)$_GET['mesa_id'] : null;

try {
    // Obtener datos de la cuenta
    $query = "SELECT c.*, p.nombre as nombre_producto
    FROM cuenta c
    INNER JOIN productos p ON c.producto_id = p.id
    WHERE c.mesa_id = ?";

    $stmt = mysqli_prepare(mysqli $conexion, query: $query);
    mysqli_stmt_bind_param(statement: $stmt, types: "i", var: &$mesa_id);
    mysqli_stmt_execute(statement: $stmt);
    $cuenta_result = mysqli_stmt_get_result(statement: $stmt);

    // Configurar impresora usando PrintNode
    $credentials = new ApiKey('teN-tTKpID...', 'UtudC1M'); // Aquí debes poner tu apiKey
    $client = new Client($credentials);
```

Y el "ID" de nuestra impresora debe colocarse así:



```
// Manejo de errores y depuración
try {
    $printers = $client->viewPrinters();
    if (empty($printers)) {
        throw new Exception(message: "No se encontraron impresoras disponibles.");
    }
    // Seleccionar la primera impresora disponible
    $printerId = '73857848';
    // O especificar directamente el ID de la impresora que deseas usar
    // $printerId = 'YOUR_PRINTER_ID';
} catch (Exception $e) {
    throw new Exception(message: "Error al obtener impresoras: " . $e->getMessage());
}
```

Página Web del Restaurante

<https://proyectos.xn--ibaez-qta.site/restaurante/>

## Conclusión Final

*El desarrollo de este proyecto ha sido una experiencia enriquecedora que ha permitido aplicar y consolidar conocimientos adquiridos en el ámbito de la programación web y la gestión de bases de datos. A lo largo del proceso de elaboración, se ha diseñado e implementado un sistema integral para la gestión de un restaurante, abarcando desde la administración de mesas y pedidos hasta la generación de tickets de cocina y el manejo de pagos.*

*Inicialmente, se establecieron las bases del sistema mediante la creación de una estructura de base de datos robusta, asegurando la correcta relación entre las diferentes entidades como mesas, productos, pedidos y usuarios. La utilización de prácticas seguras, como el uso de consultas preparadas, ha garantizado la integridad y protección de los datos contra posibles vulnerabilidades de inyección SQL.*

*La implementación de funcionalidades clave, como la gestión de pedidos y la generación automática de tickets para la cocina, ha optimizado notablemente el flujo de trabajo dentro del restaurante. La integración de librerías externas, como Mike42\Escpos para la impresión de tickets, demostró la importancia de aprovechar recursos existentes para enriquecer las capacidades del sistema sin incurrir en desarrollos redundantes.*