




DOCUMENTACIÓN TÉCNICA

SISTEMA WEB DE GESTIÓN DE RESTAURANTE “LA
MURALLA”

Tabla de contenido

Introducción	2
Descripción	2
Alcance	2
Arquitectura del sistema.....	3
Diagrama de arquitectura.....	3
Componentes Principales	3
Requisitos del sistema	5
Requisitos de Hardware	5
Requisitos de Software	5
Descripción de los módulos.....	6
Módulo de Login	6
Módulo del Administrador	7
Módulo del Mesero	8
Módulo del Cocinero	9
Base de datos.....	10
Modelo de base de datos	10
Consultas comunes	10
Mantenimiento y actualización.....	13

	DOCUMENTACIÓN TÉCNICA SISTEMA WEB DE GESTIÓN DE RESTAURANTE "LA MURALLA"	CÓDIGO: DT-002
		VERSIÓN: 1.0
		JULIO 2024

Introducción

Descripción

El Sistema Web de Gestión de Restaurante "La Muralla" está diseñado para facilitar la gestión de pedidos, cocina y administración de un restaurante. Proporciona interfaces específicas para los roles de mesero, cocinero y administrador.

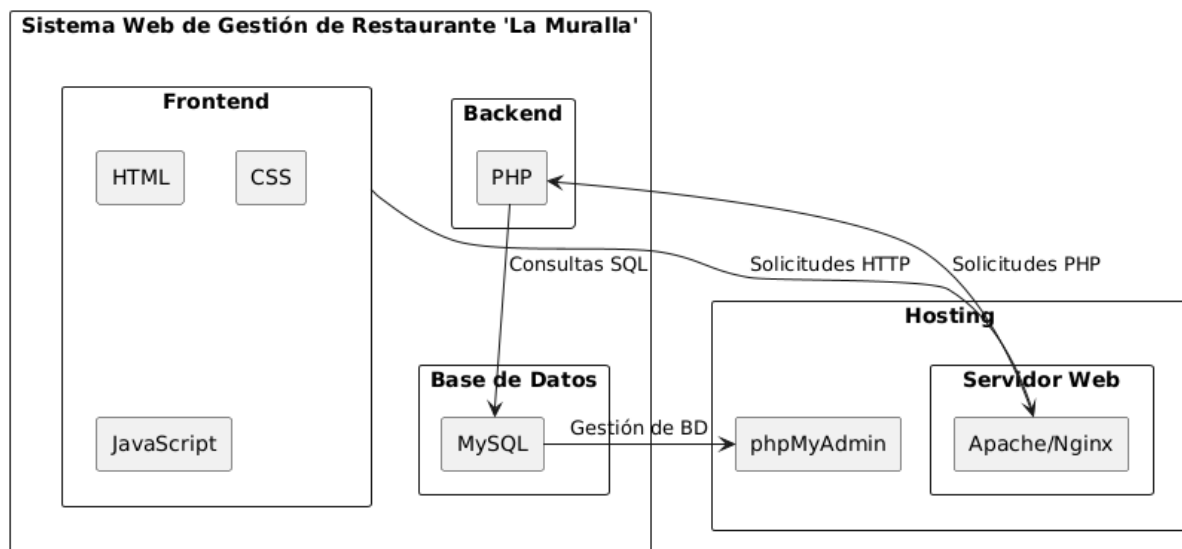
Alcance

La documentación abarca la configuración, implementación, mantenimiento y uso del sistema web. Incluye aspectos técnicos del frontend, backend, base de datos y despliegue en el hosting.

Arquitectura del sistema

Diagrama de arquitectura

Este diagrama representa la arquitectura general del sistema, donde se muestra la interacción entre el frontend (HTML, CSS, JavaScript), el backend (PHP), la base de datos (MySQL) y el hosting (como por ejemplo un Servidor Web con Apache/Nginx y phpMyAdmin).




Componentes Principales

Frontend: El frontend del sistema utiliza tecnologías estándar de desarrollo web como HTML, CSS y JavaScript para construir la interfaz de usuario que interactúa directamente con los clientes y los usuarios del restaurante.

Backend: El backend está implementado en PHP, encargado de procesar las solicitudes del cliente, realizar la lógica de negocio y gestionar la comunicación con la base de datos.

Base de Datos (MySQL): MySQL se utiliza como sistema de gestión de bases de datos relacional para almacenar y gestionar la información crítica del restaurante, como pedidos, menús y usuarios.

Hosting: Por ejemplo, consideremos que el sistema está alojado en un entorno de hosting que utiliza un servidor web (como Apache o Nginx) para gestionar las

	DOCUMENTACION TECNICA SISTEMA WEB DE GESTIÓN DE RESTAURANTE "LA MURALLA"	CÓDIGO: DT-002
		VERSIÓN: 1.0
		JULIO 2024

solicitudes HTTP que llegan y para servir tanto el contenido estático como dinámico del sitio web.

phpMyAdmin: Se utiliza phpMyAdmin como herramienta de gestión de la base de datos MySQL, facilitando tareas como la administración de tablas, consultas y mantenimiento del esquema de la base de datos.

El frontend envía solicitudes HTTP al servidor web, que a su vez las redirige al backend implementado en PHP. El backend procesa estas solicitudes, realiza consultas SQL a la base de datos MySQL a través de phpMyAdmin cuando es necesario, y luego devuelve los resultados al frontend para su visualización y manejo por parte de los usuarios del restaurante. Este diseño arquitectónico garantiza que el Sistema Web de Gestión de Restaurante 'La Muralla' sea robusto y eficiente, cumpliendo con todos los requisitos necesarios tanto funcionales como no funcionales.

Requisitos del sistema

Requisitos de Hardware

Es esencial tener un entorno de hardware y software apropiado para asegurar el desempeño eficaz y óptimo del sistema. Aquí se describen los requerimientos técnicos mínimos.

- Procesador: Mínimo Intel Core i5 o equivalente.
- Memoria RAM: Recomendado 8 GB o superior.
- Almacenamiento: Espacio suficiente para alojar el sistema operativo, software de servidor web, y archivos del sistema, aproximadamente 1 GB de espacio libre.
- Conectividad de Red: Conexión a Internet estable para acceder al sistema alojado en el servidor web.

Requisitos de Software

Sistema Operativo: Se recomienda utilizar Linux (Ubuntu Server) o Microsoft Windows.

Servidor Web: Dado el ejemplo, se debe configurar con Apache 2.x o Nginx. O el servidor de preferencia.

PHP: Se requiere PHP versión 7.4 o superior.

Base de Datos: MySQL versión 5.7 o superior, administrada a través de phpMyAdmin en el entorno de hosting.

phpMyAdmin: Debe estar instalado y correctamente configurado en el servidor para la gestión de la base de datos.

Navegador Web: Se recomienda utilizar Google Chrome o Mozilla Firefox para una experiencia óptima de usuario.

Descripción de los módulos

Módulo de Login

En esta interfaz hará la consulta a la base de datos para validar el usuario y contraseña, esta dará una respuesta dando confirmación o un mensaje predeterminado en el código y mostrándose en la web. Como se observa en el siguiente diagrama de flujo.

Diagrama de flujo de validación de datos para el inicio de sesión.

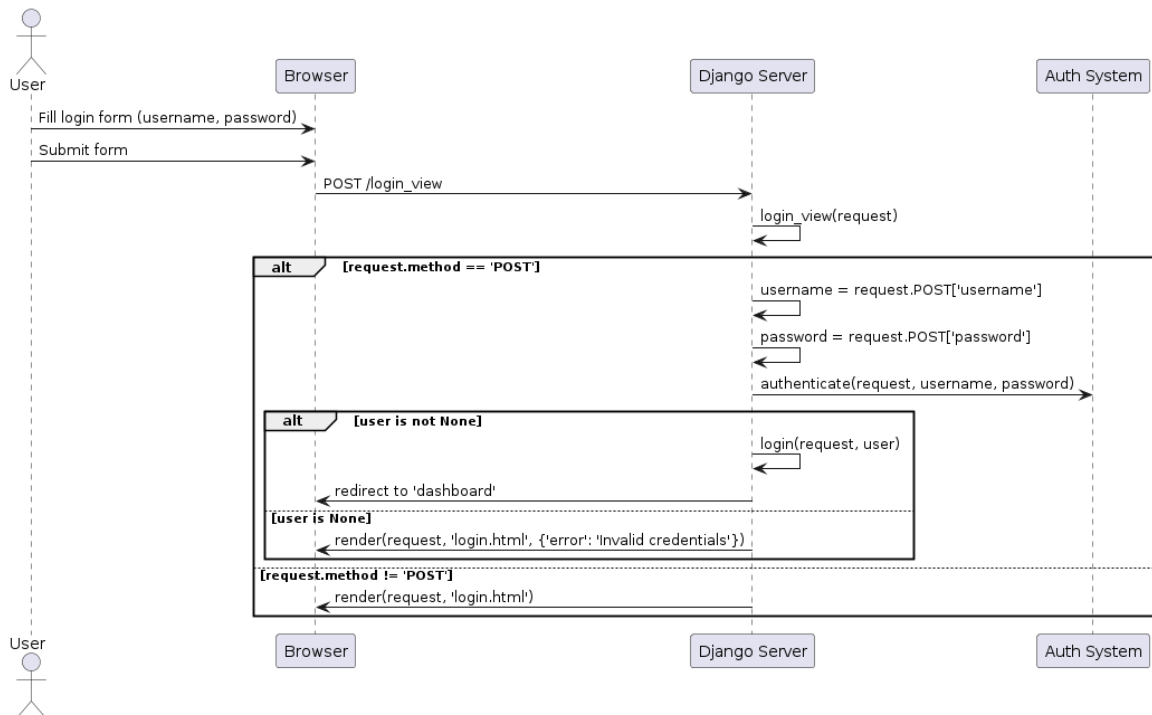
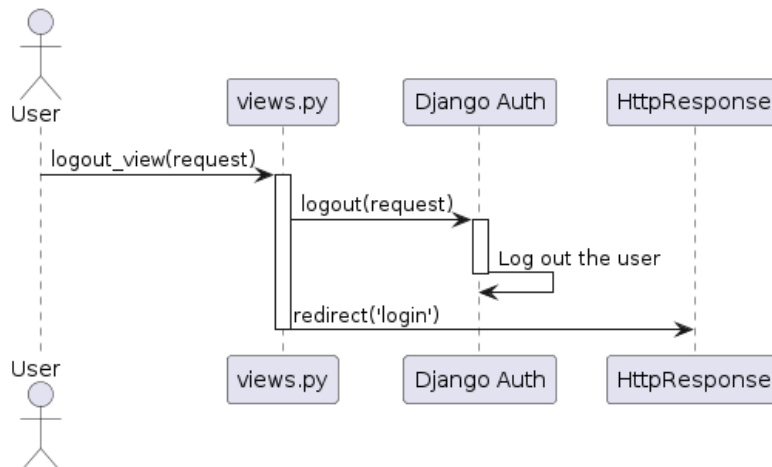


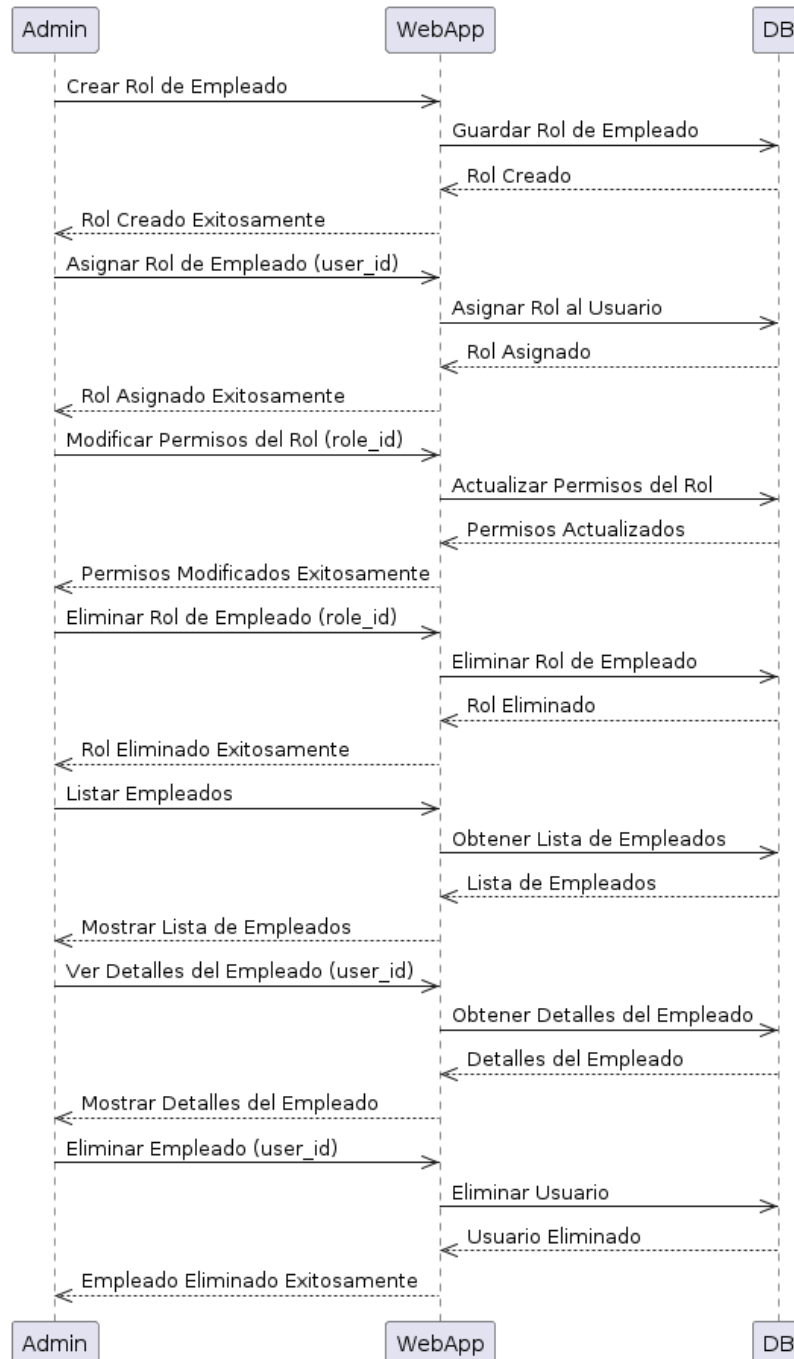
Diagrama de flujo de cerrar sesión.



Módulo del Administrador

Una vez validado los datos para el caso de administrador, el sistema mostrara el modulo del administrador. En este módulo se gestionará los menús, el personal y visualizará el resumen de ventas.

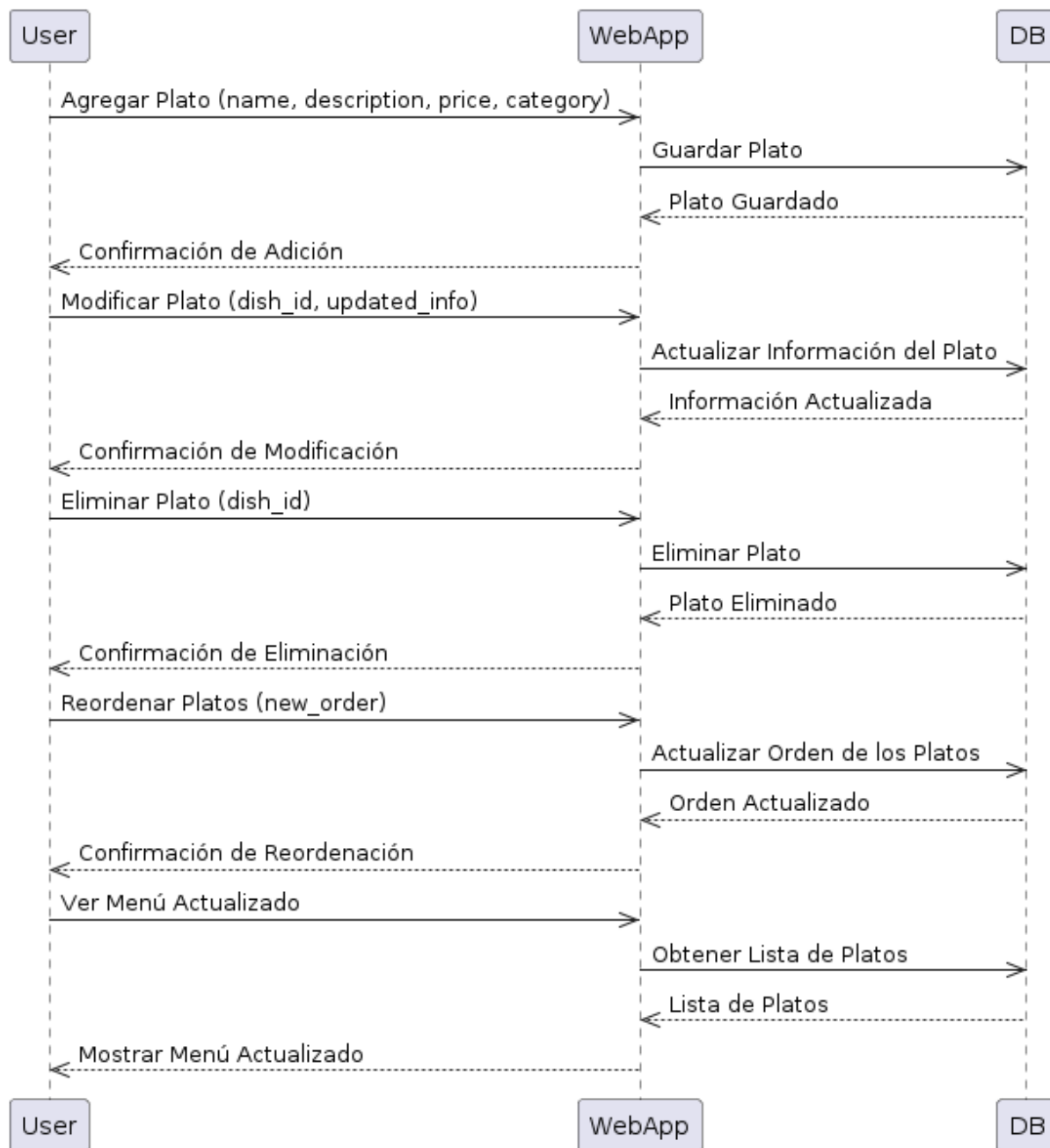
Diagrama de flujo de procesos del modulo administrador.



Módulo del Mesero

Una vez validado los datos para el caso del mesero, el sistema mostrara el módulo del mesero. En este módulo se gestionará el pedido de los menús y, se visualizará los pedidos que se encuentran listos para el recojo y despacho al cliente.

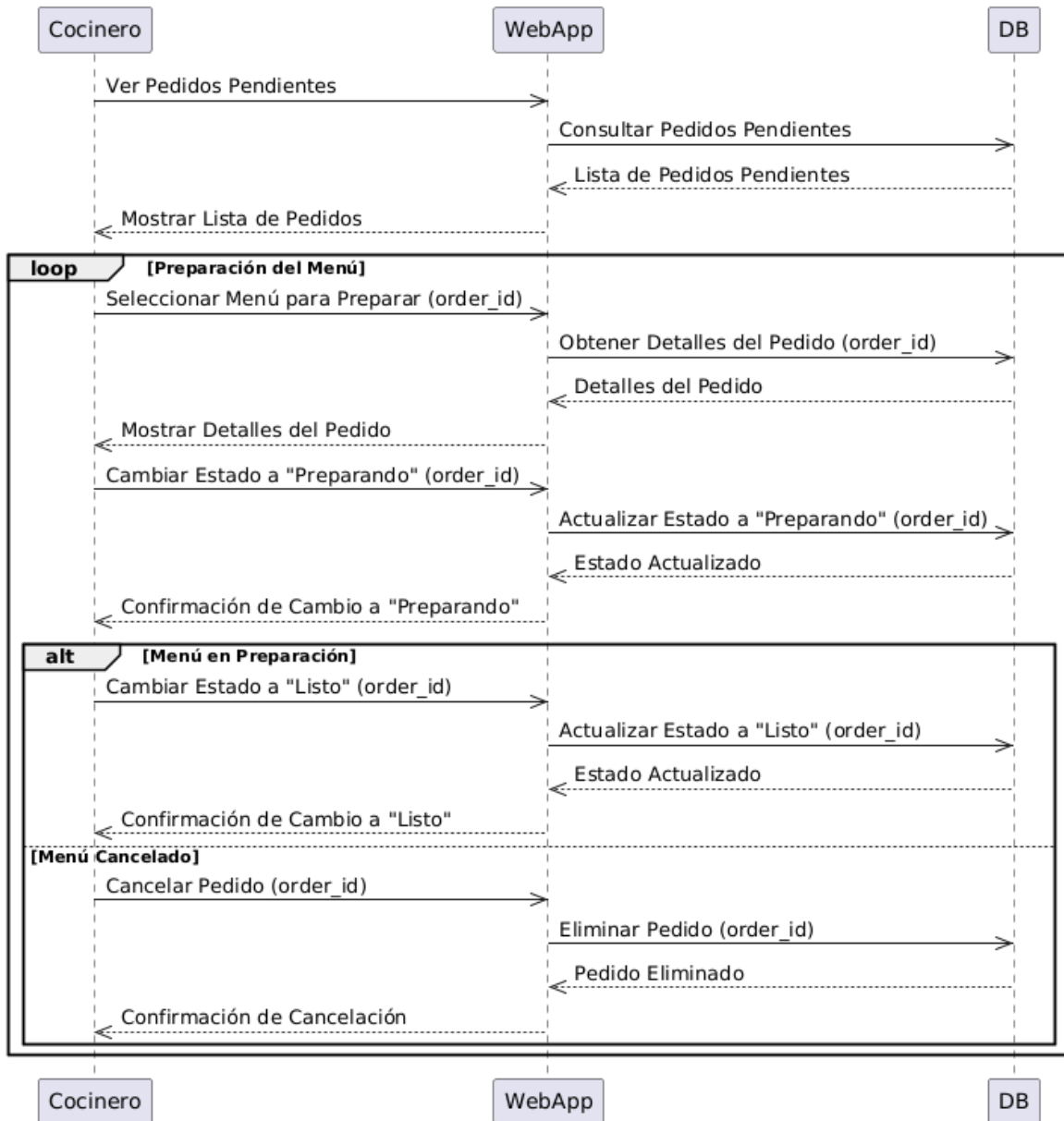
Diagrama de flujo de procesos del módulo del mesero.



Módulo del Cocinero

Una vez validado los datos para el caso del mesero, el sistema mostrara el módulo del cocinero. En este módulo se gestionará el estado de los menús y, también se visualizará los pedidos que se encuentran listos para el recojo y despacho al cliente.

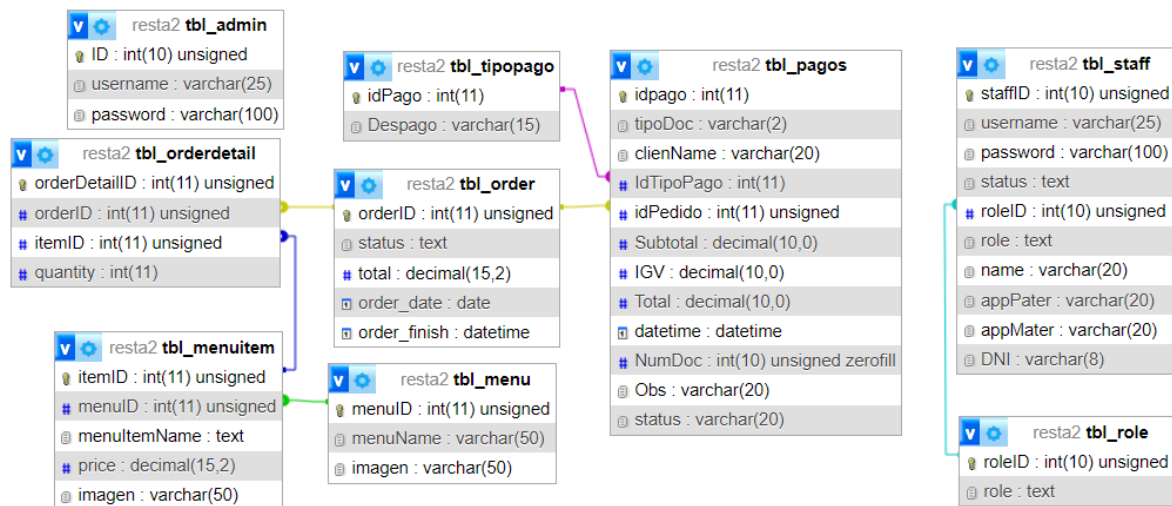
Diagrama de flujo de procesos del módulo del cocinero.



Base de datos

Modelo de base de datos

El modelo de base de datos del Sistema Web de Gestión de Restaurante "La Muralla" está diseñado para gestionar eficientemente la información relacionada con usuarios, mesas, pedidos y el menú del restaurante. El modelo incluye las siguientes entidades principales: Usuarios, Mesas, Pedidos, Ítems del Pedido y Menús. A continuación, se detalla cada entidad y sus relaciones:



Consultas comunes

Para la conexión mediante php a la base de datos.

```

dbconnection.php
1  <?php
2
3  require("config.php");
4
5  // Create connection
6  $sqlconnection = new mysqli($servername, $username, $password, $dbname);
7
8  // Check connection
9  if ($sqlconnection->connect_error) {
10     die("Connection failed: " . $sqlconnection->connect_error);
11 }
12
  
```

Para la verificación de usuario y contraseña se realiza la siguiente consulta. En el caso de Administrador.

```
admin > process.php
1 <?php
2 include("../functions.php");
3
4 //checking username and password input
5 if (isset($_POST['username']) && isset($_POST['password'])) {
6
7     //prevent sql injection by escaping special characters
8     $username = $sqlconnection->real_escape_string($_POST['username']);
9     $password = $sqlconnection->real_escape_string($_POST['password']);
10
11     //sql statement
12     $sql = "SELECT * FROM tbl_admin WHERE username = '$username' AND password = '$password'";
13
14     if ($result = $sqlconnection->query($sql)) {
15
16         if ($row = $result->fetch_array(MYSQLI_ASSOC)) {
17
18             $uid = $row['ID'];
19             $username = $row['username'];
20
21             $_SESSION['uid'] = $uid;
22             $_SESSION['username'] = $username;
23             $_SESSION['user_level'] = "admin";
24
25             echo "correct";
26         }
27     } else {
28         echo "Usuario o contraseña incorrecto.";
29     }
30 }
31
32 }
33
34 }
35
36 ?>
```

En el caso de mesero y cocinero.

```
staff > process.php
1 <?php
2 include("../functions.php");
3
4 //checking username and password input
5 if (isset($_POST['username']) && isset($_POST['password'])) {
6
7     //prevent sql injection by escaping special characters
8     $username = $sqlconnection->real_escape_string($_POST['username']);
9     $password = $sqlconnection->real_escape_string($_POST['password']);
10
11     //sql statement
12     $sql = "SELECT * FROM tbl_staff WHERE username = '$username' AND password = '$password'";
13
14     if ($result = $sqlconnection->query($sql)) {
15
16         if ($row = $result->fetch_array(MYSQLI_ASSOC)) {
17
18             $uid = $row['staffID'];
19             $username = $row['username'];
20             $role = $row['role'];
21
22             $_SESSION['uid'] = $uid;
23             $_SESSION['username'] = $username;
24             $_SESSION['user_level'] = "staff"; // 1 - admin 2 - staff
25             $_SESSION['user_role'] = $role;
26
27             echo "correct";
28         } else {
29             echo "Usuario o contraseña incorrecto.";
30         }
31     }
32 }
33
```

Actualizar pedido

```

135 function updateTotal($orderId)
136 {
137     global $sqlconnection;
138
139     $query = "
140         UPDATE tbl_order o
141         INNER JOIN (
142             SELECT SUM(OD.quantity*mi.price) AS total
143             FROM tbl_order o
144             LEFT JOIN tbl_orderdetail OD
145             ON o.orderID = OD.orderID
146             LEFT JOIN tbl_menuitem MI
147             ON OD.itemID = MI.itemID
148             LEFT JOIN tbl_menu M
149             ON MI.menuID = M.menuID
150
151             WHERE o.orderID = " . $orderId . "
152         ) x
153         SET o.total = x.total
154         WHERE o.orderID = " . $orderId . "
155     ";
156
157     if ($sqlconnection->query($query) === TRUE) {
158         echo "updated.";
159     } else {
160         //handle
161         echo "something wong";
162         echo $sqlconnection->error;
163     }
164 }

```

Ingresar nuevo empleado.

```

admin > addstaff.php
7  if ($SESSION['user_level'] != "admin")
10  if (isset($_POST['addstaff'])) {
11  if (!empty($_POST['staffname']) && !empty($_POST['staffrole'])) {
12      $staffUsername = $sqlconnection->real_escape_string($_POST['staffname']);
13      $staffRole = $sqlconnection->real_escape_string($_POST['staffrole']);
14
15      // Verificar si el usuario ya existe
16      $checkUserQuery = "SELECT COUNT(*) as count FROM tbl_staff WHERE username = '{$staffUsername}'";
17      $result = $sqlconnection->query($checkUserQuery);
18      $row = $result->fetch_assoc();
19      $userCount = $row['count'];
20
21  if ($userCount > 0) {
22      // El usuario ya existe, mostrar mensaje
23      echo "<script>
24          alert('El nombre de usuario ya existe. Por favor, elija otro.');"
25          window.location.href = 'staff.php';
26      </script>";
27  } else {
28      // Insertar el nuevo usuario
29      $addStaffQuery = "INSERT INTO tbl_staff (username, password, status, role) VALUES ('{$staffUsername}', '1234abcd..', '
30
31  if ($sqlconnection->query($addStaffQuery) === TRUE) {
32      echo "<script>
33          alert('Usuario agregado exitoso');"
34          window.location.href = 'staff.php';
35      </script>";
36      //header("Location: staff.php");
37      exit();
38  } else {
39      // Manejar otros errores
40      echo "Algo salió mal al agregar el usuario.";
41      echo $sqlconnection->error;
42  }
43  }
44  }
45  }

```

Mantenimiento y actualización

Antes de realizar cualquier mantenimiento o actualización en el código o la base de datos, es fundamental efectuar copias de seguridad tanto de la base de datos como del sistema. Esto es crucial para prevenir la pérdida de datos y garantizar la continuidad del servicio.

También se recomienda que el departamento de TI, genere lo siguientes políticas basadas en:

- Copias de seguridad (backups): Es esencial crear copias de seguridad completas y diferenciales, y almacenarlas en ubicaciones seguras y separadas del sistema principal.
- Procedimientos de recuperación (disaster recovery procedures): Asegúrese de tener planes de recuperación ante desastres y procedimientos de restauración bien definidos y probados regularmente.
- Pruebas de integridad (integrity checks): Realice pruebas de integridad en las copias de seguridad para asegurarse de que los datos no estén corruptos y sean recuperables.
- Plan de mantenimiento (maintenance plan): Detalle los pasos específicos para el mantenimiento y la actualización, incluyendo puntos de verificación (checkpoints) y procedimientos de validación post-mantenimiento.
- Documentación detallada: Mantenga una documentación completa y actualizada de todo el proceso de mantenimiento y actualización, incluyendo cambios en el código, configuraciones de la base de datos, y cualquier incidencia ocurrida durante el proceso.

A continuación, se recomienda seguir el diagrama de flujo adjunto para el procedimiento de mantenimiento y actualización.

