# **Formulario: Vista Lógica del Proyecto de Software**

**Proyecto:** Sistema\_restaurante  
**Grupo:** 7   
**Fecha:** 19/10/2024

**1. Descripción General del Sistema**

**Descripción del sistema:**

El sistema es una plataforma integral de gestión para un restaurante, que abarca múltiples aspectos de las operaciones del negocio. Esta aplicación web permite la interacción de diferentes tipos de usuarios (administradores, meseros, cocineros, clientes) y gestiona diversas funcionalidades como pedidos, reservas, inventario, y reportes.

* Principales componentes lógicos:
  + Sistema de Autenticación y Gestión de Usuarios:
  + Maneja el inicio de sesión de diferentes tipos de usuarios (login.php, procesar\_login.php).
  + Permite la administración de usuarios, incluyendo creación, modificación y eliminación (admin\_usuarios.php).
* Gestión de Pedidos y Mesas:
  + Permite a los clientes hacer pedidos (cliente.php).
  + Los meseros gestionan las mesas y los pedidos (metre.php, asignar\_mesa.php).
  + La cocina visualiza y actualiza el estado de los pedidos (cocina.js).
* Sistema de Reservas:
  + Permite a los clientes hacer reservas.
  + Los empleados gestionan y actualizan el estado de las reservas (cambio\_reservacion.php).
* Gestión de Menú:
  + Permite agregar, modificar y eliminar elementos del menú.
  + Los clientes pueden ver el menú y agregar items a su carrito (carrito.css, remove.php, check.php).
* Panel de Administración:
  + Proporciona una visión general de las operaciones del restaurante (administrador.css).
  + Genera informes y estadísticas (reportes.php, reportes.css).
* Sistema de Comunicación Interna:
  + Incluye un chat para la comunicación entre empleados (chat.js).
* Interfaz de Usuario:
  + Utiliza HTML, CSS y JavaScript para crear una interfaz de usuario responsive y amigable.
  + Implementa diferentes estilos para distintas secciones del sistema (inicio.css, nosotros.css, reseñas.css).
* Base de Datos:
  + Utiliza MySQL para almacenar y gestionar todos los datos del sistema.
* Funcionalidades Adicionales:
  + Sistema de reseñas de clientes.
  + Exportación de datos a Excel y PDF (exportar\_pdf.php).

Este sistema está diseñado para mejorar la eficiencia operativa del restaurante, proporcionar una mejor experiencia al cliente y facilitar la gestión y toma de decisiones para los administradores.

**2. Principales Clases y Entidades del Sistema**

Clase 1 (Usuario):

* Atributos:
  + id\_usuario, nombre\_usuario, contrasena, tipo\_usuario
* Métodos:
  + login(), logout(), cambiarContrasena()
* Propósito:
  + Gestionar la autenticación y los roles de los usuarios en el sistema.

Clase 2 (Mesa):

* Atributos:
  + id\_mesa, cantidad\_asientos, estado
* Métodos:
  + asignar(), liberar(), limpiar()
* Propósito:
  + Representar y gestionar las mesas del restaurante.

Clase 3 (Pedido):

* Atributos:
  + id\_pedido, id\_detalle\_mesero\_mesa, total\_cuenta, hora, fecha, estado, tipo
* Métodos:
  + crearPedido(), actualizarEstado(), calcularTotal()
* Propósito:
  + Manejar los pedidos de los clientes, incluyendo su estado y total.

Clase 4 (Platillo):

* Atributos:
  + id\_platillo, nombre\_platillo, descripcion\_platillo, precio, ruta\_foto, tipo\_platillo, estado
* Métodos:
  + agregarAlMenu(), cambiarDisponibilidad(), actualizarPrecio()
* Propósito:
  + Representar los platillos disponibles en el menú del restaurante.

Clase 5 (Reserva):

* Atributos:
  + id\_reserva, nombre\_reserva, apellido\_reserva, fecha, hora, cantidad\_personas, estado\_reserva
* Métodos:
  + crearReserva(), cancelarReserva(), modificarReserva()
* Propósito:
  + Gestionar las reservaciones de los clientes.

Clase 6 (DetallePedidoPlatillo):

* Atributos:
  + id\_detalle, id\_pedido, id\_platillo, cantidad
* Métodos:
  + agregarPlatillo(), removerPlatillo(), actualizarCantidad()
* Propósito:
  + Manejar los detalles de los platillos en cada pedido.

Clase 7 (EstadoDia):

* Atributos:
  + fecha, estado, hora\_apertura, hora\_cierre
* Métodos:
  + iniciarDia(), cerrarDia(), actualizarHorario()
* Propósito:
  + Controlar el estado del restaurante en un día específico.

Clase 8 (Administrador):

* Atributos: (heredados de Usuario)
* Métodos:
  + generarReporte(), gestionarUsuarios(), configurarSistema()
* Propósito:
  + Manejar funciones administrativas del sistema.

Clase 9 (Metre (hereda de Usuario

* Atributos: (heredados de Usuario)
* Métodos:
  + asignarMesa(), gestionarReservas(), monitorearEstadoMesas()
* Propósito:
  + Gestionar la asignación de mesas y reservas.

Clase 10 (Mesero (hereda de Usuario)):

* Atributos: (heredados de Usuario)
* Métodos:
  + tomarPedido(), actualizarEstadoPedido(), atenderMesa()
* Propósito:
  + Manejar las interacciones con los clientes y los pedidos en las mesas.

Clase 11 (Cocina (hereda de Usuario:

* Atributos: (heredados de Usuario)
* Métodos:
  + recibirPedido(), actualizarEstadoPedido(), marcarPedidoListo()
* Propósito:
  + Gestionar la preparación de los pedidos en la cocina.

**3. Relaciones entre clases**

**Herencia**

* Administrador hereda de Usuario:  
  Esta relación permite que el Administrador tenga todas las propiedades y métodos básicos de Usuario, además de sus funcionalidades específicas para gestionar el sistema.
* Metre hereda de Usuario:  
  El Metre extiende Usuario, permitiendo que tenga las propiedades y métodos básicos, así como funcionalidades específicas para gestionar las reservas y la asignación de mesas.
* Mesero hereda de Usuario:  
  Similarmente, el Mesero hereda de Usuario, lo que le proporciona los atributos y métodos básicos, además de capacidades para interactuar con los clientes y manejar pedidos.
* Cocina hereda de Usuario:  
  La clase Cocina también hereda de Usuario, facilitando la administración de los pedidos en la cocina.

**Composición**

* Pedido está compuesto por uno o más DetallePedidoPlatillo:  
  Un Pedido no puede existir sin sus detalles (DetallePedidoPlatillo), ya que estos contienen información específica sobre los platillos incluidos en el pedido.
* EstadoDia está compuesto por el estado de todas las Mesas y Platillos en un día específico:  
  EstadoDia mantiene información integral sobre el funcionamiento del restaurante en un día determinado, incluyendo el estado de todas las mesas y la disponibilidad de los platillos.

**Agregación**

* Mesa agrega Pedido:  
  Una Mesa puede tener un Pedido asociado, pero un Pedido puede existir sin estar asociado a una Mesa (por ejemplo, en pedidos para llevar).
* Platillo agrega Ingredientes:  
  Un Platillo está formado por varios Ingredientes, pero los Ingredientes pueden existir independientemente de los Platillos.

**Asociación**

* Usuario (Mesero) se asocia con Mesa:  
  Un Mesero puede atender múltiples Mesas, y una Mesa puede ser atendida por diferentes Meseros en distintos momentos.
* Reserva se asocia con Mesa:  
  Una Reserva está ligada a una Mesa específica, pero una Mesa puede tener múltiples Reservas en diferentes momentos.
* Metre se asocia con Reserva:  
  Un Metre gestiona múltiples Reservas, y cada Reserva es gestionada por un Metre.

**Dependencia**

* Cocina depende de Pedido:  
  La clase Cocina necesita la información de los Pedidos para realizar sus funciones, pero no es parte integral de la clase Pedido.
* Administrador depende de todas las demás clases:  
  El Administrador interactúa con todas las demás clases para gestionar el sistema, pero no es parte integral de ninguna de ellas.

**Relación Muchos a Muchos**

* Platillo tiene una relación muchos a muchos con Ingrediente:  
  Esta relación se implementa a través de una tabla intermedia (Platillo\_Ingrediente en la base de datos), lo que permite que un Platillo contenga múltiples Ingredientes y viceversa.

**Detalles Adicionales de las Relaciones**

* Usuario y Pedido:  
  Un Usuario (específicamente un Mesero) puede crear múltiples Pedidos, pero cada Pedido está asociado a un único Usuario que lo creó.
* Mesa y Reserva:  
  Una Mesa puede tener múltiples Reservas a lo largo del tiempo, pero cada Reserva está asociada a una única Mesa.
* Platillo y DetallePedidoPlatillo:  
  Un Platillo puede aparecer en múltiples DetallePedidoPlatillo, y cada DetallePedidoPlatillo se refiere a un único Platillo.
* EstadoDia y otras clases:  
  EstadoDia tiene una relación de agregación con Mesa y Platillo, ya que contiene información sobre el estado de todas las mesas y la disponibilidad de los platillos en un día específico.
* Administrador y otras clases:  
  El Administrador tiene una relación de dependencia con prácticamente todas las demás clases, ya que necesita acceder y manipular la información de todas ellas para gestionar el sistema en su conjunto.

**4. Interfaces y dependencias**

**Interfaz de Usuario (IUsuario)**

* **Implementada por:** Usuario, Administrador, Metre, Mesero, Cocina
* **Métodos:** login(), logout(), cambiarContrasena()
* **Propósito:** Proporciona una interfaz común para todas las clases de usuario, asegurando que todas tengan funcionalidades básicas de autenticación.

**Interfaz de Gestión de Pedidos (IGestionPedidos)**

* **Implementada por:** Mesero, Cocina
* **Métodos:** crearPedido(), actualizarEstadoPedido(), visualizarPedidos()
* **Propósito:** Define las operaciones comunes relacionadas con la gestión de pedidos para los roles de Mesero y Cocina.

**Interfaz de Gestión de Mesas (IGestionMesas)**

* **Implementada por:** Metre, Mesero
* **Métodos:** asignarMesa(), liberarMesa(), actualizarEstadoMesa()
* **Propósito:** Establece las operaciones necesarias para la gestión de mesas, compartidas por Metre y Mesero.

**Interfaz de Gestión de Menú (IGestionMenu)**

* **Implementada por:** Administrador
* **Métodos:** agregarPlatillo(), modificarPlatillo(), eliminarPlatillo()
* **Propósito:** Define las operaciones para gestionar el menú del restaurante.

**Interfaz de Reportes (IReportes)**

* **Implementada por:** Administrador
* **Métodos:** generarReporteVentas(), generarReporteInventario(), exportarReporte()
* **Propósito:** Establece las operaciones para la generación y exportación de reportes.

**Dependencias**

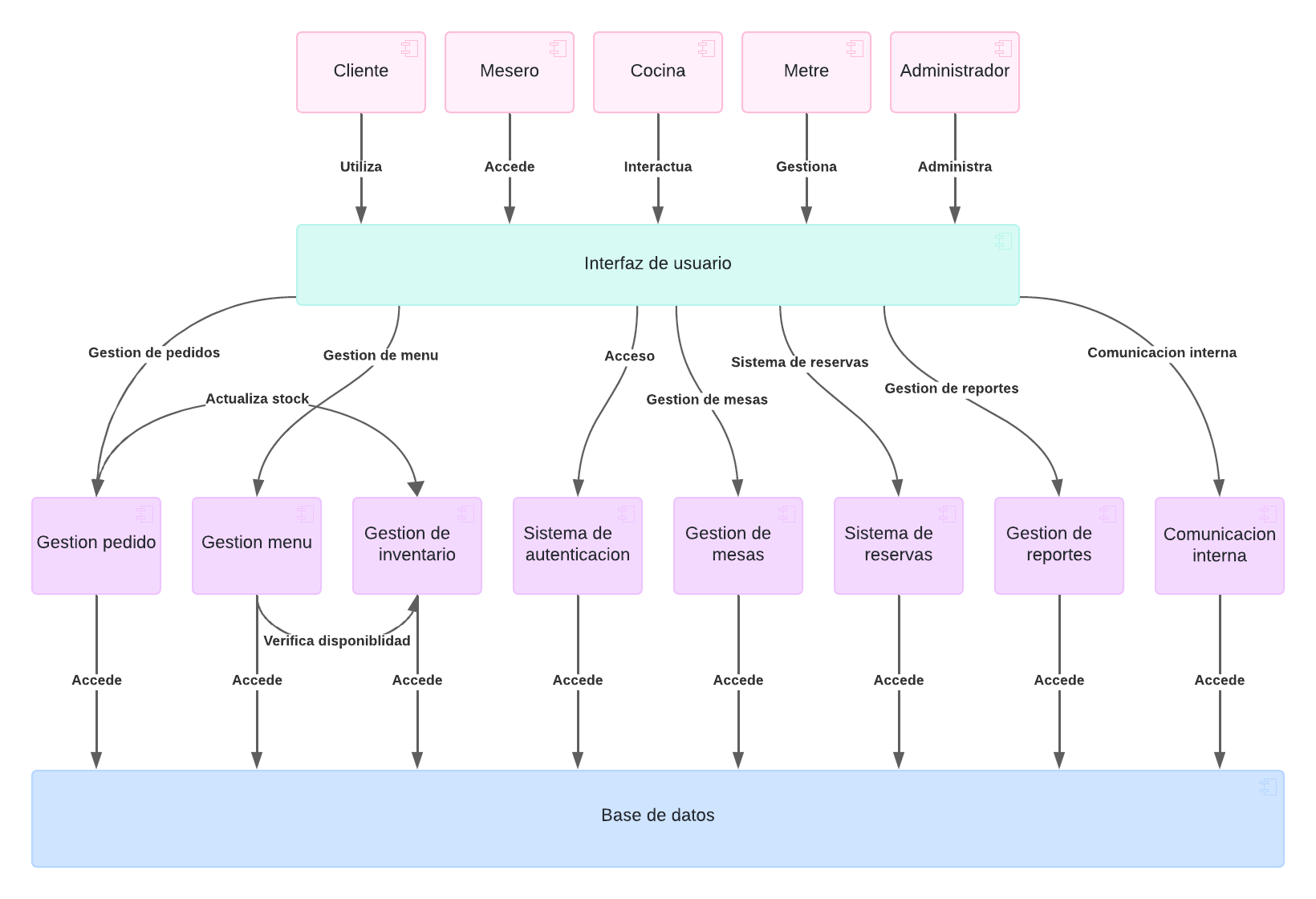
1. **Módulo de Autenticación**
   * **Dependencias:** Todas las clases de usuario (Usuario, Administrador, Metre, Mesero, Cocina)
   * **Interacción:** Proporciona servicios de autenticación y gestión de sesiones para todos los usuarios del sistema.
2. **Módulo de Gestión de Pedidos**
   * **Dependencias:** Pedido, DetallePedidoPlatillo, Platillo, Mesero, Cocina
   * **Interacción:** Coordina la creación, actualización y seguimiento de pedidos entre Meseros y Cocina.
3. **Módulo de Gestión de Mesas y Reservas**
   * **Dependencias:** Mesa, Reserva, Metre, Mesero
   * **Interacción:** Maneja la asignación de mesas, el estado de las mismas y la gestión de reservas.
4. **Módulo de Gestión de Inventario**
   * **Dependencias:** Platillo, Ingrediente, Administrador
   * **Interacción:** Controla el inventario de ingredientes y su relación con los platillos del menú.
5. **Módulo de Reportes y Estadísticas**
   * **Dependencias:** Administrador, Pedido, Mesa, Platillo
   * **Interacción:** Genera informes y estadísticas basados en los datos de ventas, mesas y menú.
6. **Módulo de Comunicación Interna**
   * **Dependencias:** Todas las clases de usuario
   * **Interacción:** Facilita la comunicación entre los diferentes roles de usuarios mediante un sistema de chat.

**5. Diagrama de Clases del Sistema**

**Diagrama de clases:**Diagrama

Descripción generada automáticamente

**6. Diagrama de componentes (opcional)**

**Diagrama de componentes:**

**7. Comentarios adicionales**

**Comentarios:**

**Arquitectura Modular:** El sistema está diseñado con una arquitectura modular que separa claramente las responsabilidades, facilitando mantenimiento y escalabilidad.

**Uso de Interfaces:** Las interfaces (IUsuario, IGestionPedidos, etc.) promueven un diseño flexible y desacoplado, permitiendo que diferentes clases implementen comportamientos comunes sin afectar la funcionalidad existente.

**Herencia y Polimorfismo:** La herencia en las clases de usuario (Administrador, Metre, Mesero, Cocina) permite reutilizar código y manejar distintos tipos de usuarios uniformemente.

**Gestión de Estado:** La clase EstadoDia es clave para gestionar la disponibilidad de mesas y platillos, esencial para una operación eficiente.

**Relaciones Complejas:** El sistema gestiona relaciones complejas, como la relación muchos a muchos entre Platillo e Ingrediente, utilizando clases intermedias como DetallePedidoPlatillo.

**Seguridad y Autenticación:** El Sistema de Autenticación asegura que solo usuarios autorizados accedan a funcionalidades según su rol, manteniendo la integridad de los datos.

**Integración de Procesos:** La creación de un pedido por un mesero se vincula directamente con la gestión de inventario y preparación en cocina.

**Escalabilidad:** La estructura modular permite agregar nuevos tipos de usuarios y funcionalidades sin cambios drásticos en la arquitectura.

**Gestión de Datos en Tiempo Real:** Los módulos de Gestión de Pedidos y Mesas están diseñados para manejar datos en tiempo real, crucial para la eficiencia operativa.

**Flexibilidad en Reportes:** El módulo de Generación de Reportes permite crear informes adaptados a las necesidades del negocio.

**Comunicación Interna:** La inclusión de un componente de Comunicación Interna resalta la importancia de la coordinación entre el personal.

**Futuras Expansiones:** El diseño considera posibles integraciones futuras, como sistemas de delivery externos o programas de fidelización de clientes.