**Sistema de Reservas de Restaurante**

**1. Justificación del Proyecto**

En la actualidad, los restaurantes requieren sistemas eficientes que les permitan gestionar sus reservas de forma automatizada, evitando la saturación de mesas y mejorando la experiencia tanto del cliente como del administrador.

* El sistema desarrollado en Spring Boot con base de datos H2 (inicialmente en memoria, adaptable a MySQL) busca resolver los siguientes problemas:
* Automatización de reservas: El cliente puede realizar reservas en línea ingresando sus datos básicos (nombre, teléfono, correo electrónico).
* Disponibilidad de mesas: El sistema valida si una mesa está disponible en la fecha solicitada, evitando conflictos en las reservas.
* Gestión administrativa: El administrador puede visualizar las reservas registradas, confirmar o rechazar reservas y mantener control sobre el estado de las mesas.
* Persistencia de información: Se registra la relación entre clientes, mesas y reservas, permitiendo generar reportes futuros y análisis de ocupación.

Con esto se busca optimizar la operación del restaurante, garantizando que los clientes tengan una experiencia confiable y que el personal administrativo cuente con herramientas para gestionar las reservas de forma eficiente.

2. **Modelo Entidad – Relación (E–R)**

El modelo E–R del sistema contempla las siguientes entidades principales:

* **Cliente**: almacena los datos del usuario que realiza la reserva.
* **Mesa**: representa cada mesa física del restaurante con su número y capacidad.
* **Reserva**: almacena las solicitudes de reserva realizadas por los clientes.

**Relaciones**

* Un Cliente puede hacer muchas Reservas (relación 1:N).
* Una Mesa puede estar asociada a muchas Reservas en diferentes fechas, pero nunca más de una por el mismo día (restricción de negocio).
* Cada Reserva se vincula exactamente con un Cliente y una Mesa.

**3. Casos de Uso Principales**

**Caso de Uso 1**: Realizar Reserva

**Actor**: Cliente

**Descripción**: El cliente ingresa sus datos (nombre, teléfono, email), selecciona fecha, hora y número de personas. El sistema valida la disponibilidad de mesas y, si existe, crea la reserva con estado Pendiente.

**Flujo** **Normal**:

* El cliente abre el formulario de reserva.
* Ingresa sus datos personales.
* Selecciona fecha, hora y cantidad de personas.
* El sistema busca una mesa disponible.
* Se guarda la reserva y se informa al cliente.

**Excepción**: Si no hay mesas disponibles en la fecha seleccionada, el sistema muestra un mensaje de error.

**Caso de Uso 2**: Confirmar/Cancelar Reserva

**Actor**: Administrador

**Descripción**: El administrador puede visualizar las reservas y actualizar su estado de Pendiente a Confirmada o Cancelada.

**Flujo Normal:**

* El administrador accede al panel de gestión de reservas.
* Selecciona una reserva pendiente.
* Actualiza el estado a “Confirmada” o “Cancelada”.
* El sistema actualiza la base de datos y libera/ocupa la mesa según corresponda.

**Caso de Uso 3**: Consultar Historial de Reservas

**Actor**: Administrador

**Descripción**: El administrador puede visualizar todas las reservas realizadas por cada cliente, así como el historial de ocupación de mesas.

**Flujo Normal:**

* El administrador ingresa al panel de clientes.
* Selecciona un cliente específico.

El sistema muestra todas las reservas pasadas y actuales asociadas al cliente.