Universidad Autónoma de Chihuahua

Facultad de Ingeniería

Asignatura: Fundamentos de bases de datos

Clave: CI675

Grupo: 6CC2

Semestre: Ago-Dic 2024

Docente: José Saul De Lira Miramontes

Manual Tecnico

Integrantes:

- Allan Hall Solorio 358909
- Erick Mendoza Escarzaga 357307
- Jorge Alejandro Beltran Rosales 348635

1. Especificaciones del sistema

Sistema/Aplicación

a. **Nombre del sistema:** GastroRes - Sistema de Gestión de Reservas y Pedidos para Restaurantes

b. **Descripción y delimitación del sistema:**

GastroRes es una aplicación de escritorio diseñada para facilitar la gestión integral de un restaurante. Permite administrar información clave como clientes, mesas, empleados, menús, reservas y pedidos. El sistema se centra en optimizar las operaciones diarias, especialmente en la gestión de reservas de mesas y la toma de pedidos asociados a dichas reservas.

Delimitación del sistema:

- **Alcance:** Gestión interna del restaurante, enfocada en reservas y pedidos.
- **No incluye:** Módulos de facturación, contabilidad, inventario de insumos o acceso en línea para clientes.
- **Usuarios objetivo:** Personal del restaurante como administradores, recepcionistas y meseros.
- **Entorno:** Aplicación de escritorio que opera en un entorno local sin funcionalidades en línea o acceso remoto.

c. **Objetivo general:**

Desarrollar una aplicación eficiente y amigable que optimice la gestión de reservas y pedidos en un restaurante, mejorando la organización interna y la satisfacción del cliente.

d. **Objetivos específicos:**

- Implementar un módulo para registrar y administrar información de clientes, incluyendo datos de contacto y preferencias.
- Gestionar la información de mesas, incluyendo número, capacidad y ubicación, asegurando una asignación óptima.
- Administrar el personal del restaurante, permitiendo agregar, actualizar y eliminar empleados y sus roles.
- Crear y mantener un catálogo de menús, incluyendo platos, descripciones y precios.
- Facilitar la creación, modificación y cancelación de reservas, verificando disponibilidad y evitando conflictos.
- Gestionar pedidos asociados a reservas, incluyendo selección de menús y cantidades.
- Proporcionar una interfaz gráfica intuitiva y fácil de usar que agilice las tareas del personal.
- Asegurar la integridad y consistencia de los datos mediante el uso de una base de datos robusta (Oracle Database).

e. **Descripción de tipos de usuarios:**

- **Administradores:**
    - **Funciones:** Configuración inicial del sistema, gestión de empleados, mesas y menús, supervisión general.
    - **Accesos:** Acceso completo a todas las funcionalidades y módulos del sistema.
- **Recepcionistas:**
    - **Funciones:** Gestión de reservas, registro y actualización de información de clientes, asignación de mesas.
    - **Accesos:** Módulos de clientes, reservas y mesas.
- **Meseros/Camareros:**
    - **Funciones:** Consulta de reservas y pedidos, actualización del estado de pedidos.
    - **Accesos:** Módulos de reservas y pedidos.

f. **Entorno operativo del sistema:**

- **Software requerido:**
    - **Python:** Interpretado en Python, versión 3.x.
    - **Bibliotecas:** customtkinter para la interfaz gráfica, cx_Oracle para la conexión con la base de datos, tkinter y ttk para componentes GUI adicionales.
    - **Base de datos:** Oracle Database (conexión mediante cx_Oracle).
- **Hardware requerido:**
    - **Ordenador personal:** Capaz de ejecutar Python y Oracle Database.
    - **Almacenamiento:** Espacio suficiente para instalar Oracle Database y almacenar datos operativos.
- **Sistema operativo:**
    - **Compatibilidad:** Windows, Linux o macOS con soporte para Python y Oracle Database.
- **Configuración adicional:**
    - **Conexión a la base de datos:** Configuración correcta del cliente de Oracle y variables de entorno necesarias.
    - **Entorno de red:** Aunque es una aplicación local, se requiere configuración de red si la base de datos está en un servidor remoto.

2. Especificación de requerimientos

a. Requerimientos Funcionales

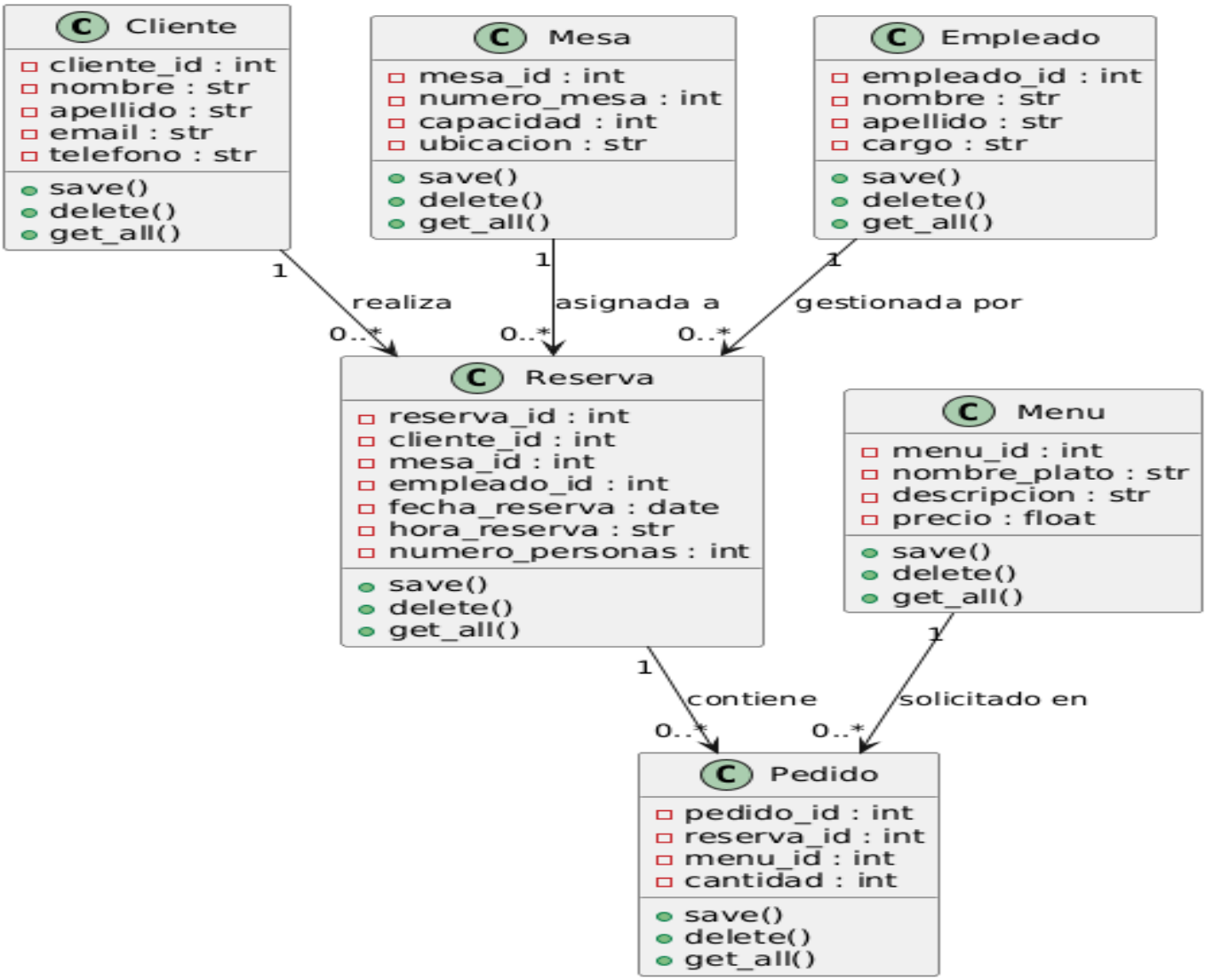
1. **Gestión de Clientes**
    - Registrar nuevos clientes con información personal (nombre, apellido, email, teléfono).
    - Actualizar la información de clientes existentes.
    - Eliminar clientes del sistema.
    - Listar y visualizar información de clientes.
2. **Gestión de Mesas**
    - Registrar nuevas mesas con detalles como número, capacidad y ubicación.
    - Actualizar la información de mesas existentes.
    - Eliminar mesas del sistema.
    - Listar y visualizar información de mesas.
3. **Gestión de Empleados**

- Registrar nuevos empleados con información personal y cargo.
- Actualizar la información de empleados existentes.
- Eliminar empleados del sistema.
- Listar y visualizar información de empleados.

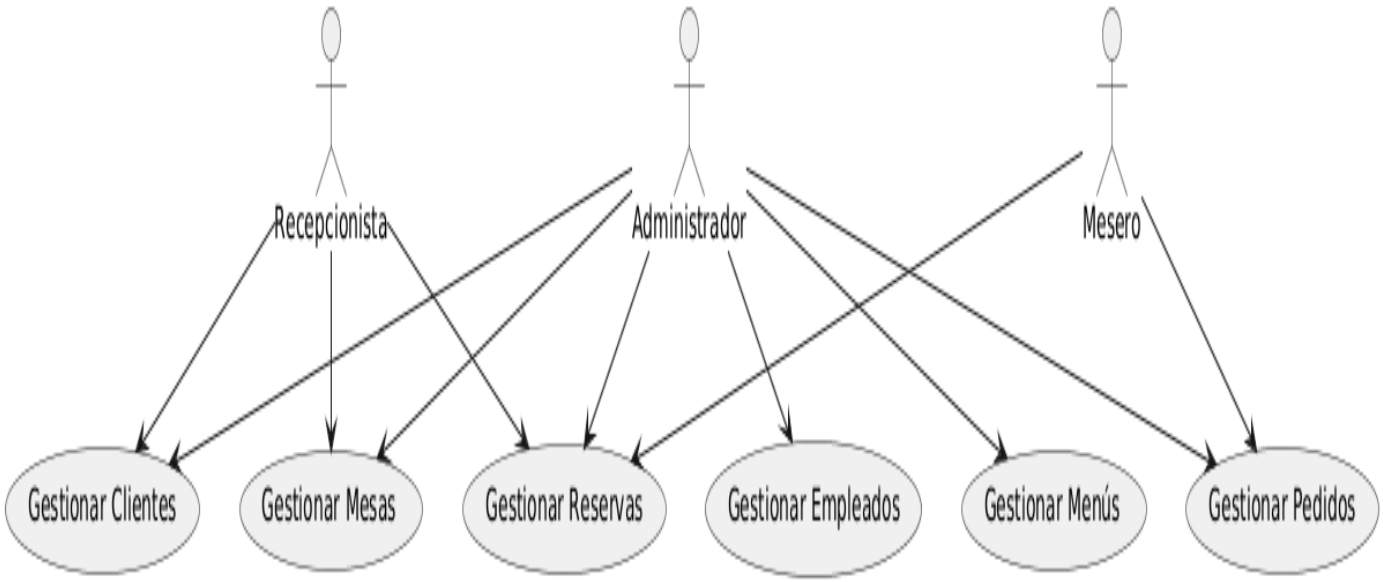
4. **Gestión de Menús**
   - Registrar nuevos menús con detalles de platos, descripciones y precios.
   - Actualizar la información de menús existentes.
   - Eliminar menús del sistema.
   - Listar y visualizar información de menús.
5. **Gestión de Reservas**
   - Crear nuevas reservas asociadas a clientes, mesas y empleados.
   - Verificar disponibilidad de mesas para evitar conflictos en fechas y horas.
   - Actualizar la información de reservas existentes.
   - Eliminar reservas del sistema.
   - Listar y visualizar información de reservas.
6. **Gestión de Pedidos**
   - Registrar nuevos pedidos asociados a reservas y menús.
   - Actualizar la información de pedidos existentes.
   - Eliminar pedidos del sistema.
   - Listar y visualizar información de pedidos.
7. **Interfaz de Usuario**
   - Proporcionar una interfaz gráfica intuitiva para facilitar la interacción con el sistema.
   - Implementar pestañas o secciones para cada módulo de gestión.
   - Ofrecer funcionalidades de búsqueda y filtrado en las listas.
8. **Validación de Datos**
   - Validar entradas de usuario para asegurar que los datos sean correctos y completos.
   - Mostrar mensajes de advertencia o error en caso de entradas inválidas.
9. **Persistencia de Datos**
   - Conectar y operar con una base de datos Oracle para almacenar información.
   - Manejar excepciones y errores de conexión a la base de datos.
10. **Seguridad Básica**
    - Confirmar acciones críticas como eliminaciones mediante cuadros de diálogo de confirmación.

b. Modelado del sistema

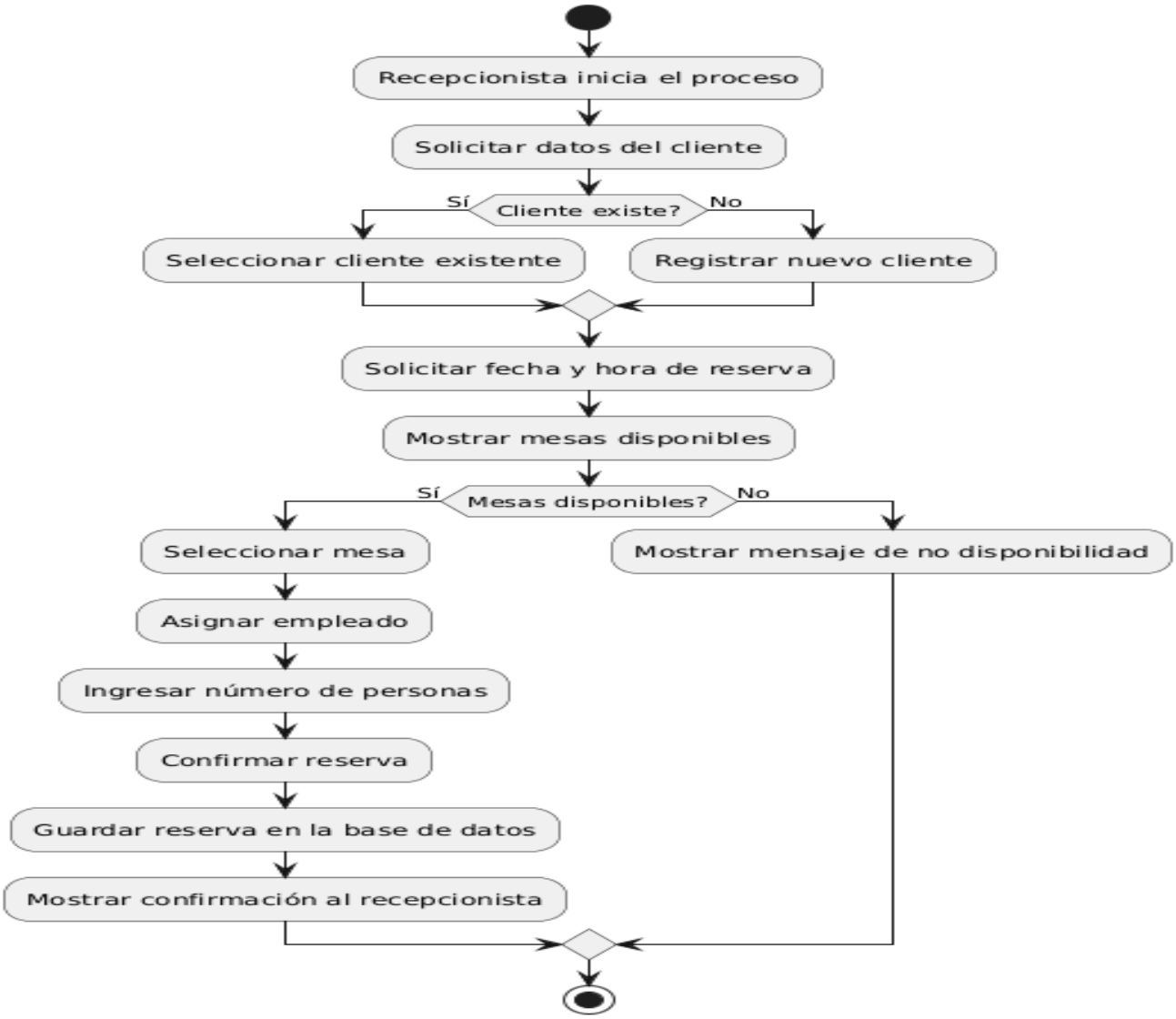
I. Diagrama de clases



II. Diagrama de casos de uso



III. Diagrama de actividades



c. Requerimientos no funcionales

Usabilidad

- La interfaz debe ser intuitiva y fácil de navegar para usuarios con conocimientos básicos de informática.
- Utilizar mensajes claros y concisos para guiar al usuario en las operaciones.

Fiabilidad

- El sistema debe ser robusto y manejar adecuadamente los errores y excepciones.
- Asegurar la integridad de los datos durante las operaciones CRUD (Crear, Leer, Actualizar, Eliminar).

Rendimiento

- Las operaciones comunes (e.g., guardar, actualizar, eliminar registros) deben realizarse en tiempos aceptables (<2 segundos).

- Optimizar consultas a la base de datos para minimizar tiempos de respuesta.

Compatibilidad

- El sistema debe ser compatible con los principales sistemas operativos que soportan Python y Oracle Database (Windows, Linux, macOS).
- Las dependencias de terceros deben ser instalables mediante gestores de paquetes estándar (e.g., pip).

Seguridad

- Implementar confirmaciones para operaciones destructivas como eliminaciones.
- Validar y sanitizar entradas de usuario para prevenir inyecciones o corrupción de datos.

Mantenibilidad

- El código debe ser modular y seguir buenas prácticas de programación para facilitar futuras modificaciones.
- Utilizar nomenclatura consistente y comentarios cuando sea necesario.

Escalabilidad

- La arquitectura del sistema debe permitir la incorporación de nuevas funcionalidades sin requerir reescrituras significativas.
- Considerar la posibilidad de migrar a una base de datos más robusta o distribuir la carga en caso de crecimiento.

Documentación

- Proveer documentación adecuada para usuarios y desarrolladores, incluyendo manuales y comentarios en el código.
- Incluir diagramas y especificaciones para facilitar la comprensión del sistema.

Legal y Ética

- Cumplir con las regulaciones locales en cuanto al manejo y almacenamiento de datos personales de clientes y empleados.
- Asegurar que la información sensible sea tratada con confidencialidad.

Accesibilidad

- Considerar principios de diseño accesible para usuarios con discapacidades (e.g., soporte para lectores de pantalla).

3. Arquitectura

a. Capas/Niveles (Layers/Tiers)

El sistema está diseñado siguiendo una arquitectura en capas que separa claramente las responsabilidades de cada componente, lo que facilita el mantenimiento y la escalabilidad del sistema. Las capas principales son:

1. **Capa de Presentación (Interfaz de Usuario):**
    - **Descripción:** Es la capa responsable de la interacción con el usuario. Utiliza la biblioteca customtkinter para crear una interfaz gráfica moderna y amigable.
    - **Componentes:**
        - **Clase App:** Es la ventana principal de la aplicación que contiene las pestañas y los elementos gráficos.
        - **Pestañas (Tabview):** Cada pestaña corresponde a un módulo de gestión (Clientes, Mesas, Reservas, Empleados, Menús y Pedidos).
        - **Elementos de interfaz:** Formularios, entradas de datos, botones y listas que permiten al usuario interactuar con el sistema.
2. **Capa de Lógica de Negocio (Business Logic Layer):**
    - **Descripción:** Contiene las clases que representan las entidades del sistema y encapsulan la lógica de negocio.
    - **Componentes:**
        - **Clases Modelo:** Cliente, Mesa, Empleado, Menu, Reserva, Pedido.
        - **Métodos de las clases:**
            - **save():** Maneja la creación y actualización de registros en la base de datos.
            - **delete():** Elimina registros de la base de datos.
            - **get_all():** Recupera todos los registros de una entidad.
3. **Capa de Acceso a Datos (Data Access Layer):**
    - **Descripción:** Gestiona la comunicación con la base de datos Oracle utilizando la biblioteca cx_Oracle.
    - **Componentes:**
        - **Función get_connection():** Establece y devuelve una conexión a la base de datos.
        - **Operaciones CRUD en las clases modelo:** Las interacciones directas con la base de datos (consultas SQL) se realizan dentro de los métodos de las clases modelo.

b. Frontend/Backend

Frontend:

- **Descripción:** Comprende todos los componentes que interactúan directamente con el usuario.
- **Componentes:**
    - **Interfaz Gráfica de Usuario (GUI):** Construida con customtkinter, incluye ventanas, formularios, botones, tablas y otros elementos interactivos.
    - **Eventos y Controladores:** Métodos que responden a acciones del usuario, como clics en botones o selecciones en listas.

Backend:

- **Descripción:** Incluye la lógica de negocio y el acceso a datos que soportan las operaciones del sistema.
- **Componentes:**

- **Clases Modelo:** Gestionan la lógica relacionada con cada entidad (validaciones, cálculos, etc.).
- **Interacción con la Base de Datos:** A través de cx_Oracle, maneja las consultas y transacciones con la base de datos Oracle.
- **Gestión de Excepciones y Errores:** Mecanismos para manejar errores durante las operaciones de base de datos y asegurar la integridad de los datos.

c. Estructura Modular del Sistema

El sistema está estructurado de manera modular, permitiendo una separación clara de responsabilidades y facilitando la extensibilidad. A continuación se detalla la estructura modular:

1. **Módulos de Entidades (Clases Modelo):**
   - **Cliente:**
     - **Atributos:** cliente_id, nombre, apellido, email, telefono.
     - **Métodos:** save(), delete(), get_all().
   - **Mesa:**
     - **Atributos:** mesa_id, numero_mesa, capacidad, ubicacion.
     - **Métodos:** save(), delete(), get_all().
   - **Empleado:**
     - **Atributos:** empleado_id, nombre, apellido, cargo.
     - **Métodos:** save(), delete(), get_all().
   - **Menú:**
     - **Atributos:** menu_id, nombre_plato, descripcion, precio.
     - **Métodos:** save(), delete(), get_all().
   - **Reserva:**
     - **Atributos:** reserva_id, cliente_id, mesa_id, empleado_id, fecha_reserva, hora_reserva, numero_personas.
     - **Métodos:** save(), delete(), get_all().
   - **Pedido:**
     - **Atributos:** pedido_id, reserva_id, menu_id, cantidad.
     - **Métodos:** save(), delete(), get_all().

2. **Módulo de Conexión a la Base de Datos:**
   - **Función get_connection():** Maneja la conexión con la base de datos Oracle, encapsulando detalles de la conexión y facilitando su reutilización.
3. **Módulo de Interfaz de Usuario (Clase App):**
   - **Estructura:**
     - **Inicialización:** Configura la ventana principal y establece el tema de la aplicación.
     - **Creación de Widgets:** Genera las pestañas y llama a los métodos de configuración para cada módulo.
     - **Métodos de Configuración:**
       - **setup_clientes_tab():** Configura la pestaña de gestión de clientes.
       - **setup_mesas_tab():** Configura la pestaña de gestión de mesas.

        - **setup_empleados_tab():** Configura la pestaña de gestión de empleados.
        - **setup_reservas_tab():** Configura la pestaña de gestión de reservas.
        - **setup_menus_tab():** Configura la pestaña de gestión de menús.
        - **setup_pedidos_tab():** Configura la pestaña de gestión de pedidos.
4. **Módulos Funcionales por Pestaña:**
    - Cada pestaña en la interfaz gráfica corresponde a un módulo funcional que agrupa funcionalidades relacionadas.
    - **Componentes Comunes:**
        - **Formularios de Entrada:** Campos para ingresar o editar información.
        - **Botones de Acción:** Agregar, actualizar y eliminar registros.
        - **Listas/Tablas:** Visualización de registros existentes utilizando ttk.Treeview.
        - **Métodos Asociados:**
            - **Carga de Datos:** Métodos como load_clientes(), load_mesas(), etc., que obtienen datos de la base de datos y actualizan las vistas.
            - **Eventos de Selección:** Métodos como on_cliente_select(), que manejan la selección de elementos en las listas.
            - **Validación y Manejo de Errores:** Verificación de entradas de usuario y manejo de excepciones durante las operaciones.
5. **Módulo de Utilidades:**
    - **Validaciones Comunes:** Validación de entradas, como formatos de fecha y números.
    - **Actualización de Comboboxes:** Métodos como update_reserva_comboboxes(), que actualizan las opciones disponibles en los campos desplegables.

Flujo de Operaciones:

- **Interacción del Usuario:**
    - El usuario interactúa con la interfaz gráfica, ingresando datos y solicitando acciones.
- **Procesamiento en la Capa de Presentación:**
    - Los eventos son capturados por los controladores asociados a los widgets.
- **Llamadas a la Lógica de Negocio:**
    - Los métodos de la interfaz llaman a los métodos de las clases modelo para realizar operaciones específicas.
- **Acceso a Datos:**
    - Los métodos de las clases modelo interactúan con la base de datos utilizando cx_Oracle.
- **Actualización de la Interfaz:**
    - Tras completar una operación, la interfaz se actualiza para reflejar los cambios (por ejemplo, recargando las listas de registros)

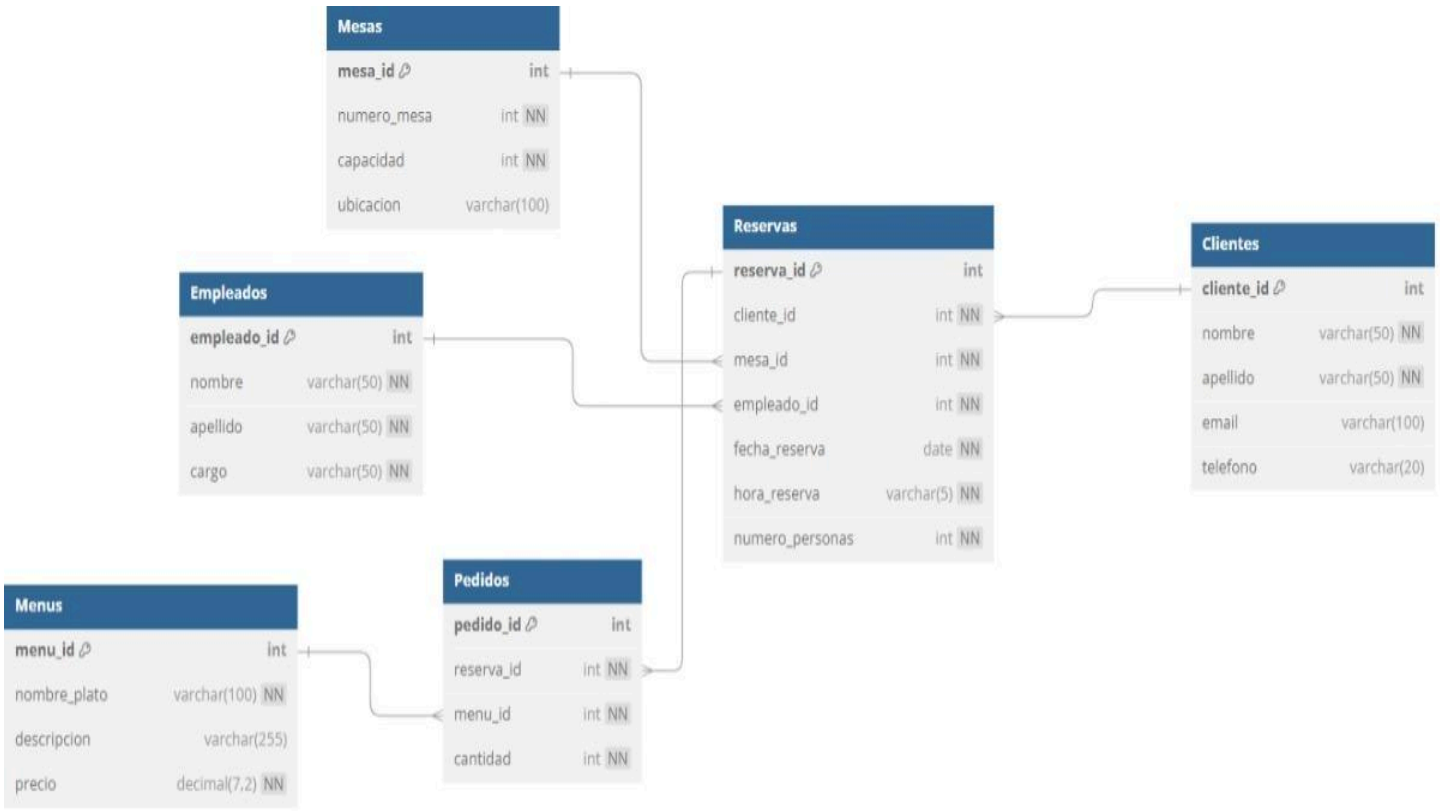
Consideraciones de Diseño:

- **Separación de Responsabilidades:** La lógica de negocio y el acceso a datos están encapsulados en las clases modelo, mientras que la interfaz gráfica se encarga únicamente de la interacción con el usuario.
- **Reutilización de Código:** Funciones como get_connection() y métodos estáticos como get_all() permiten reutilizar código y facilitan el mantenimiento.

- **Manejabilidad y Extensibilidad:** La estructura modular facilita la incorporación de nuevas funcionalidades o modificaciones sin afectar significativamente otras partes del sistema.
- **Manejo de Excepciones:** Se implementa manejo de errores y excepciones para garantizar la estabilidad del sistema y proporcionar retroalimentación al usuario en caso de errores.

4. Bases de datos

a. Diagrama conceptual (Entidad relación)



b. Esquema Lógico de la Base de Datos

I. Especificación de Tablas

Tabla Clientes

- **Descripción:** Almacena información de los clientes del restaurante.
- **Columnas:**
  - cliente_id: Número entero, clave primaria, generado automáticamente (IDENTITY).
  - nombre: Cadena de caracteres (máximo 50), **NO NULO**.
  - apellido: Cadena de caracteres (máximo 50), **NO NULO**.
  - email: Cadena de caracteres (máximo 100), opcional.
  - telefono: Cadena de caracteres (máximo 20), opcional.

Tabla Mesas

- **Descripción:** Contiene información sobre las mesas disponibles en el restaurante.
- **Columnas:**
    - mesa_id: Número entero, clave primaria, generado automáticamente (IDENTITY).
    - numero_mesa: Número entero que identifica la mesa, **NO NULO**.
    - capacidad: Número entero que indica el número de personas que caben en la mesa, **NO NULO**.
    - ubicacion: Cadena de caracteres (máximo 100), opcional.

Tabla Empleados

- **Descripción:** Registra los datos del personal que trabaja en el restaurante.
- **Columnas:**
    - empleado_id: Número entero, clave primaria, generado automáticamente (IDENTITY).
    - nombre: Cadena de caracteres (máximo 50), **NO NULO**.
    - apellido: Cadena de caracteres (máximo 50), **NO NULO**.
    - cargo: Cadena de caracteres (máximo 50), indica el puesto del empleado, **NO NULO**.

Tabla Reservas

- **Descripción:** Almacena las reservas realizadas por los clientes.
- **Columnas:**
    - reserva_id: Número entero, clave primaria, generado automáticamente (IDENTITY).
    - cliente_id: Número entero, **NO NULO**, clave foránea que referencia a Clientes.cliente_id.
    - mesa_id: Número entero, **NO NULO**, clave foránea que referencia a Mesas.mesa_id.
    - empleado_id: Número entero, **NO NULO**, clave foránea que referencia a Empleados.empleado_id.
    - fecha_reserva: Tipo fecha, indica la fecha de la reserva, **NO NULO**.
    - hora_reserva: Cadena de caracteres (máximo 5), indica la hora de la reserva en formato HH:MM, **NO NULO**.
    - numero_personas: Número entero, indica cuántas personas asistirán, **NO NULO**.

Tabla Menus

- **Descripción:** Contiene los platos disponibles en el menú del restaurante.
- **Columnas:**
    - menu_id: Número entero, clave primaria, generado automáticamente (IDENTITY).
    - nombre_plato: Cadena de caracteres (máximo 100), nombre del plato, **NO NULO**.
    - descripcion: Cadena de caracteres (máximo 255), descripción del plato, opcional.
    - precio: Número decimal con 7 dígitos y 2 decimales (formato 7,2), precio del plato, **NO NULO**.

Tabla Pedidos

- **Descripción:** Registra los pedidos asociados a cada reserva.
- **Columnas:**
    - pedido_id: Número entero, clave primaria, generado automáticamente (IDENTITY).
    - reserva_id: Número entero, **NO NULO**, clave foránea que referencia a Reservas.reserva_id.
    - menu_id: Número entero, **NO NULO**, clave foránea que referencia a Menus.menu_id.
    - cantidad: Número entero, indica la cantidad del plato pedido, **NO NULO**.

II. Integridad de Datos (Constraints)

Claves Primarias (PRIMARY KEY):

- Garantizan la unicidad de cada registro en la tabla.
    - Clientes.cliente_id
    - Mesas.mesa_id
    - Empleados.empleado_id
    - Reservas.reserva_id
    - Menus.menu_id
    - Pedidos.pedido_id

Claves Foráneas (FOREIGN KEY):

- Aseguran la integridad referencial entre tablas relacionadas.
    - Reservas.cliente_id referencia a Clientes.cliente_id
    - Reservas.mesa_id referencia a Mesas.mesa_id
    - Reservas.empleado_id referencia a Empleados.empleado_id
    - Pedidos.reserva_id referencia a Reservas.reserva_id
    - Pedidos.menu_id referencia a Menus.menu_id

Restricciones de NO NULO (NOT NULL):

- Impiden que se ingresen valores nulos en columnas críticas.
    - Clientes.nombre
    - Clientes.apellido
    - Mesas.numero_mesa
    - Mesas.capacidad
    - Empleados.nombre
    - Empleados.apellido
    - Empleados.cargo
    - Reservas.cliente_id
    - Reservas.mesa_id
    - Reservas.empleado_id
    - Reservas.fecha_reserva
    - Reservas.hora_reserva
    - Reservas.numero_personas
    - Menus.nombre_plato
    - Menus.precio
    - Pedidos.reserva_id
    - Pedidos.menu_id
    - Pedidos.cantidad

Restricciones Adicionales:

- **Unicidad:**
    - Podríamos añadir restricciones UNIQUE en Mesas.numero_mesa si cada número de mesa es único.
- **Checks:**
    - Asegurar que Menus.precio sea mayor que cero.
    - Verificar que Pedidos.cantidad sea mayor que cero.
    - Validar que Reservas.hora_reserva siga un formato válido y esté dentro de un rango de horas operativas.

Integridad Referencial:

- **Acciones en Cascada:**
    - Al eliminar un Cliente, podríamos restringir la eliminación si tiene Reservas asociadas, o bien eliminar en cascada.
    - Es importante definir las políticas de eliminación y actualización para mantener la consistencia de los datos.

Índices:

- Para mejorar el rendimiento en consultas, se han creado índices en las columnas que son claves foráneas:
    - idx_reservas_cliente en Reservas(cliente_id)
    - idx_reservas_mesa en Reservas(mesa_id)
    - idx_pedidos_reserva en Pedidos(reserva_id)
    - idx_pedidos_menu en Pedidos(menu_id)

5. Lógica/reglas del negocio

La lógica de negocio del sistema define las reglas y procesos que gobiernan las operaciones del restaurante en términos de gestión de clientes, mesas, empleados, menús, reservas y pedidos. A continuación, se detallan las principales reglas implementadas:

Gestión de Clientes:

- **Registro de Clientes:**
    - Los clientes deben proporcionar al menos su nombre y apellido.
    - El email y teléfono son opcionales pero recomendados para facilitar la comunicación.
- **Actualización y Eliminación:**
    - Se permite modificar la información de los clientes existentes.
    - Los clientes pueden ser eliminados siempre y cuando no afecten a reservas o pedidos activos.

Gestión de Mesas:

- **Registro de Mesas:**
    - Cada mesa tiene un número único y una capacidad definida.
    - La ubicación es opcional y sirve para identificar mejor la mesa dentro del restaurante.
- **Disponibilidad de Mesas:**
    - Una mesa no puede ser reservada por más de un cliente en el mismo horario.
    - La capacidad de la mesa debe ser adecuada al número de personas de la reserva.

Gestión de Empleados:

- **Registro de Empleados:**
    - Se debe ingresar el nombre, apellido y cargo del empleado.
    - Los cargos pueden ser roles como mesero, administrador, etc.
- **Asignación de Reservas:**
    - Los empleados pueden ser asignados a reservas para gestionar mejor la atención al cliente.

Gestión de Menús:

- **Registro de Menús:**

    - Cada plato debe tener un nombre y precio definidos.
    - La descripción es opcional pero útil para detallar ingredientes o características del plato.
- **Validación de Precio:**
    - El precio debe ser un valor numérico positivo.

Gestión de Reservas:

- **Creación de Reservas:**
    - Una reserva debe estar asociada a un cliente, una mesa y un empleado.
    - Se debe especificar la fecha, hora y número de personas.
- **Verificación de Disponibilidad:**
    - Antes de confirmar una reserva, el sistema verifica que la mesa esté disponible en la fecha y hora solicitadas.
- **Restricciones de Capacidad:**
    - El número de personas no puede exceder la capacidad de la mesa reservada.
- **Actualización y Cancelación:**
    - Las reservas pueden ser modificadas o canceladas según las necesidades del cliente y disponibilidad.

Gestión de Pedidos:

- **Registro de Pedidos:**
    - Los pedidos se asocian a una reserva y un menú específico.
    - Se debe especificar la cantidad de cada plato solicitado.
- **Validación de Cantidad:**
    - La cantidad debe ser un número entero positivo.
- **Actualización y Eliminación:**
    - Los pedidos pueden ser modificados o eliminados antes de su preparación o entrega.

Validaciones Generales:

- **Entradas Obligatorias:**
    - Los campos obligatorios deben ser completados para procesar la información.
- **Formato de Datos:**
    - Fechas en formato YYYY-MM-DD, horas en HH:MM.
    - Números y precios deben ser valores numéricos válidos.
- **Confirmaciones:**
    - Acciones críticas como eliminaciones requieren confirmación del usuario.
- **Manejo de Errores:**
    - El sistema maneja excepciones y muestra mensajes claros al usuario en caso de errores.

6. Descripción de la interfaz de la aplicación

La aplicación cuenta con una interfaz gráfica amigable e intuitiva, desarrollada utilizando customtkinter. Está estructurada mediante pestañas que separan las diferentes funcionalidades, permitiendo al usuario navegar fácilmente entre los módulos.

Estructura Principal:

- **Ventana Principal:**
  - Título: "Sistema de Reservas de Restaurante".
  - Tamaño: 1000x700 píxeles.
  - Incluye un switch para alternar entre modo oscuro y claro.
- **Pestañas de Navegación:**
  - **Clientes**
  - **Mesas**
  - **Empleados**
  - **Reservas**
  - **Menús**
  - **Pedidos**

Elementos Comunes en las Pestañas:

- **Formulario de Entrada:**
  - Situado generalmente a la izquierda.
  - Incluye etiquetas y campos de entrada para cada atributo de la entidad.
  - Botones para agregar, actualizar y eliminar registros.
- **Listado de Registros:**
  - Situado a la derecha.
  - Utiliza ttk.Treeview para mostrar datos en formato de tabla.
  - Columnas ajustadas según la entidad (por ejemplo, ID, Nombre, Apellido, etc.).
  - Permite seleccionar registros para editar o eliminar.

Detalles por Pestaña:

1. **Clientes:**
   - **Campos:**
     - Nombre (obligatorio).
     - Apellido (obligatorio).
     - Email.
     - Teléfono.
   - **Funciones:**
     - Agregar nuevo cliente.
     - Actualizar información de cliente existente.
     - Eliminar cliente seleccionado.
2. **Mesas:**
   - **Campos:**
     - Número de Mesa (obligatorio).
     - Capacidad (obligatorio).
     - Ubicación.
   - **Funciones:**
     - Agregar nueva mesa.
     - Actualizar información de mesa existente.
     - Eliminar mesa seleccionada.
3. **Empleados:**
   - **Campos:**
     - Nombre (obligatorio).
     - Apellido (obligatorio).
     - Cargo (obligatorio).
   - **Funciones:**

        - Agregar nuevo empleado.
        - Actualizar información de empleado existente.
        - Eliminar empleado seleccionado.
4. **Reservas:**
    - **Campos:**
        - Cliente (selección de lista desplegable).
        - Mesa (selección de lista desplegable).
        - Empleado (selección de lista desplegable).
        - Fecha de Reserva (formato YYYY-MM-DD).
        - Hora de Reserva (formato HH:MM).
        - Número de Personas.
    - **Funciones:**
        - Agregar nueva reserva.
        - Actualizar reserva existente.
        - Eliminar reserva seleccionada.
        - Verificar disponibilidad de mesas.
5. **Menús:**
    - **Campos:**
        - Nombre del Plato (obligatorio).
        - Descripción.
        - Precio (obligatorio).
    - **Funciones:**
        - Agregar nuevo menú.
        - Actualizar información del menú existente.
        - Eliminar menú seleccionado.
6. **Pedidos:**
    - **Campos:**
        - Reserva (selección de lista desplegable).
        - Menú (selección de lista desplegable).
        - Cantidad.
    - **Funciones:**
        - Agregar nuevo pedido.
        - Actualizar pedido existente.
        - Eliminar pedido seleccionado.

Características Adicionales:

- **Interactividad:**
    - Los campos se llenan automáticamente al seleccionar un registro en la tabla.
    - Las listas desplegables se actualizan dinámicamente al agregar nuevos clientes, mesas, etc.
- **Mensajes de Usuario:**
    - Se utilizan cuadros de diálogo para confirmaciones, advertencias y notificaciones de éxito o error.
- **Validaciones:**
    - Antes de realizar operaciones, se valida que los campos obligatorios estén completos y correctos.

7. Descripción de reglas de seguridad (Acceso/operación)

Aunque el sistema no implementa un sistema de autenticación de usuarios, se han establecido ciertas reglas y prácticas para asegurar la integridad y seguridad de las operaciones:

Validación de Datos:

- **Entradas de Usuario:**
  - Se verifica que los campos obligatorios estén completos antes de procesar la información.
  - Se validan tipos de datos (números, fechas, horas) para evitar errores o inyecciones maliciosas.
- **Valores Permitidos:**
  - Números de mesa y capacidades deben ser enteros positivos.
  - Precios deben ser números decimales positivos.
  - Cantidades en pedidos deben ser enteros positivos.

Manejo de Excepciones:

- **Errores de Base de Datos:**
  - Las operaciones con la base de datos están envueltas en bloques try-except para capturar y manejar excepciones.
  - En caso de error, se revierte la transacción y se notifica al usuario sin exponer detalles sensibles.

Confirmaciones de Usuario:

- **Operaciones Críticas:**
  - Antes de eliminar registros, se solicita confirmación al usuario mediante cuadros de diálogo.
  - Esto previene eliminaciones accidentales y permite al usuario reconsiderar la acción.

Integridad Referencial:

- **Restricciones en Base de Datos:**
  - Claves foráneas y constraints aseguran que las relaciones entre tablas sean coherentes.
  - No se puede eliminar un registro que esté siendo referenciado en otra tabla, evitando inconsistencias.

Acceso a la Base de Datos:

- **Credenciales:**
  - Las credenciales de acceso a la base de datos están definidas en el código, lo cual es adecuado para un entorno controlado.
  - En entornos de producción, se recomienda gestionar las credenciales de forma segura, por ejemplo, utilizando variables de entorno.

Limitaciones Actuales:

- **Sin Autenticación de Usuarios:**

    - Cualquier persona con acceso a la aplicación puede realizar todas las operaciones.
- **Sin Registro de Actividades:**
    - No se registra quién realiza cada operación, lo que dificulta el seguimiento y auditoría.

Recomendaciones:

- **Implementar Control de Acceso:**
    - Añadir un sistema de login para identificar a los usuarios y asignar roles (administrador, recepcionista, mesero).
- **Registrar Actividades:**
    - Mantener un log de operaciones para auditar acciones y detectar posibles irregularidades.
- **Seguridad en Conexión:**
    - Utilizar conexiones seguras y encriptación si se accede a la base de datos a través de una red.

9. Conclusión

El sistema desarrollado es una herramienta completa para la gestión interna de un restaurante, abarcando desde la administración de clientes hasta la gestión de reservas y pedidos. Gracias a su interfaz intuitiva y su arquitectura modular, facilita las operaciones diarias y mejora la eficiencia del personal.

Las reglas de negocio implementadas aseguran que las operaciones se realicen de manera consistente y acorde a las políticas del restaurante. Sin embargo, se identifican áreas de mejora, especialmente en aspectos de seguridad y control de acceso, que podrían fortalecerse con la implementación de sistemas de autenticación y registro de actividades.

En general, el sistema cumple con los objetivos planteados, proporcionando una base sólida que puede ser ampliada y mejorada en futuras iteraciones. Con la incorporación de funcionalidades adicionales y mejoras en seguridad, el sistema tiene el potencial de adaptarse a las necesidades cambiantes del restaurante y ofrecer un valor añadido en la gestión operativa.