Explicación Diagramas Proyecto Sistema de Restaurante

Diagramas Comportamentales

Diagramas de Actividad

```
@startuml Actividad
|Cliente|
start
:Decide qué mesa ocupar;
:Selecciona tipo de servicio;
if (Comer en el restaurante?) then (sí)
  :Escoge comida;
  if (¿Desea sopa?) then (sí)
    :Selecciona sopa;
  else (no)
    :No selecciona sopa;
  endif
  if (¿Desea postre?) then (sí)
    :Selecciona postre;
  else (no)
    :No selecciona postre;
  endif
else (no)
  :Selecciona comida para llevar;
endif

:Ordena;
:Se le sirve la orden;
:Genera factura;
:Factura detallada para el dueño;

stop
@enduml
```

Diagrama Comportamental de Actividad: Proceso de Servicio en un Restaurante

Este diagrama representa el flujo de actividades que un cliente sigue al decidir cómo disfrutar su comida en un restaurante. A continuación se detallan los pasos involucrados en el proceso:

Descripción del Diagrama

1. **Inicio**: El proceso comienza cuando el cliente entra al restaurante y decide qué mesa ocupar.
2. **Selección de Servicio**: El cliente elige el tipo de servicio que desea, que puede ser comer en el restaurante o pedir comida para llevar.
3. **Comer en el Restaurante**:
    - Si el cliente decide comer en el restaurante, se le presenta la opción de escoger su comida.
    - **Sopa**: Se pregunta al cliente si desea sopa:
        - Si responde afirmativamente, selecciona una sopa.
        - Si responde negativamente, simplemente no selecciona sopa.
    - **Postre**: Después de la sopa, se pregunta si desea un postre:
        - Si dice que sí, elige un postre.
        - Si dice que no, omite esta selección.
4. **Comida para Llevar**:
    - Si el cliente decide no comer en el restaurante, procede a seleccionar la comida para llevar.
5. **Orden**: Una vez que el cliente ha realizado sus selecciones, ordena su comida.
6. **Servicio**: El restaurante sirve la orden al cliente.
7. **Factura**: Al finalizar, se genera una factura.
    - Esta factura es detallada para el dueño, lo que facilita la gestión administrativa del restaurante.
8. **Fin**: El proceso termina una vez que el cliente ha completado su experiencia.

Conclusiones

Este diagrama muestra claramente las decisiones que debe tomar un cliente al interactuar con un restaurante, así como las diferentes ramificaciones según sus elecciones. Es una herramienta útil para entender y optimizar el servicio al cliente en el sector de la restauración.

Diagramas de Comunicación

```
@startuml Comportamental
skinparam actorPadding 10
skinparam rectanglePadding 10

actor Cliente
actor Camarero
actor "Dueño del Restaurante" as Dueño

rectangle "Sistema de Pedido" {
    Cliente -> Sistema : SeleccionaMesa()
    Cliente -> Sistema : EligeServicio(tipo)
    Cliente -> Sistema : SeleccionaComida(opciones)
```

```
    Cliente -> Sistema : RealizaPedido()

    Sistema -> Camarero : TomaPedido()
    Camarero -> Cliente : SirveOrden()
    Sistema -> Cliente : RecibeOrden()
    Sistema -> Cliente : GeneraFactura()
    Sistema -> Dueño : EnviaFacturaDetallada()
}
@enduml
```

Diagrama de Comunicación: Proceso de Pedido en un Restaurante

Este diagrama ilustra la interacción entre los actores involucrados en el proceso de pedido dentro de un restaurante, destacando la comunicación entre el cliente, el camarero, el dueño del restaurante y el sistema de pedidos.

Descripción del Diagrama

1. **Actores Involucrados**:
    - **Cliente**: Persona que visita el restaurante y realiza el pedido.
    - **Camarero**: Empleado que toma el pedido del cliente y sirve la comida.
    - **Dueño del Restaurante**: Responsable de la gestión administrativa del establecimiento.
    - **Sistema de Pedido**: Plataforma que gestiona el flujo de información entre los actores.

2. **Flujo de Comunicación**:
    - **Selecciona Mesa**: El cliente inicia el proceso seleccionando una mesa a través del sistema.
    - **Elige Servicio**: El cliente elige el tipo de servicio que desea (comer en el restaurante o para llevar).
    - **Selecciona Comida**: El cliente escoge las opciones de comida disponibles.
    - **Realiza Pedido**: Finalmente, el cliente envía el pedido al sistema.

3. **Interacción con el Camarero**:
    - **Toma Pedido**: El sistema envía el pedido al camarero para que lo procese.
    - **Sirve Orden**: El camarero sirve la orden al cliente, completando la experiencia gastronómica.

4. **Comunicación con el Cliente**:
    - **Recibe Orden**: El sistema confirma la recepción de la orden al cliente.
    - **Genera Factura**: Al finalizar el servicio, se genera una factura para el cliente.

5. **Comunicación con el Dueño**:

- **Envía Factura Detallada**: El sistema envía una factura detallada al dueño del restaurante para su gestión administrativa.

Conclusiones

Este diagrama de comunicación es esencial para entender cómo fluyen las interacciones entre el cliente, el camarero, el dueño del restaurante y el sistema de pedidos. Facilita la identificación de posibles áreas de mejora en el proceso de atención al cliente y la eficiencia operativa del restaurante.

Diagramas de Secuencia

```
@startuml Secuencia
actor Cliente
participant "Sistema de Restaurante" as Sistema
participant "Dueño del Restaurante" as Dueno

Cliente -> Sistema : Mostrar mesas disponibles

Cliente -> Sistema : Seleccionar mesa (ej. Mesa 1)

Cliente -> Sistema : Elegir tipo de servicio (1: Restaurante, 2: Para llevar)
Sistema --> Cliente : Confirmar tipo de servicio

alt Servicio en Restaurante
    Sistema -> Cliente : Mostrar menú del día
    Cliente -> Sistema : Seleccionar sopa (Sí/No)
    alt Sopa
        Sistema -> Cliente : Mostrar opciones de sopa
        Cliente -> Sistema : Seleccionar sopa
    end

    Cliente -> Sistema : Seleccionar plato principal
    Cliente -> Sistema : Seleccionar postre (Sí/No)
    alt Postre
        Sistema -> Cliente : Mostrar opciones de postre
        Cliente -> Sistema : Seleccionar postre
    end

    Cliente -> Sistema : Confirmar pedido
    Sistema --> Cliente : Servir pedido
    Sistema -> Dueno : Generar factura
    Dueno --> Sistema : Generar factura detallada

else Servicio para llevar
    Sistema -> Cliente : Mostrar menú del día
    Cliente -> Sistema : Seleccionar sopa (Sí/No)
    alt Sopa
        Sistema -> Cliente : Mostrar opciones de sopa
        Cliente -> Sistema : Seleccionar sopa
    end
```

```
    Cliente -> Sistema : Seleccionar plato principal
    Cliente -> Sistema : Seleccionar postre (Sí/No)
    alt Postre
        Sistema -> Cliente : Mostrar opciones de postre
        Cliente -> Sistema : Seleccionar postre
    end

    Cliente -> Sistema : Preguntar por desechables (Sí/No)
    alt Desechables
        Sistema -> Cliente : Costo adicional por desechables
    else
        Sistema -> Cliente : Sin costo adicional
    end

    Cliente -> Sistema : Confirmar pedido
    Sistema --> Cliente : Preparar pedido para llevar
    Sistema -> Dueno : Generar factura
    Dueno --> Sistema : Generar factura detallada
end
@enduml
```

Diagrama de Secuencia: Proceso de Pedido en un Restaurante

Este diagrama de secuencia detalla el flujo de interacciones entre el cliente, el sistema de restaurante y el dueño del restaurante durante el proceso de selección y pedido de comida. Se muestran dos escenarios: comer en el restaurante y pedir comida para llevar.

Descripción del Diagrama

Actores y Participantes

- **Cliente**: Usuario que interactúa con el sistema para realizar su pedido.
- **Sistema de Restaurante**: Plataforma que gestiona las interacciones entre el cliente y el dueño del restaurante.
- **Dueño del Restaurante**: Responsable de la administración y gestión del restaurante.

Flujo de Interacciones

1. **Mostrar Mesas Disponibles**: El cliente solicita ver las mesas disponibles a través del sistema.
2. **Seleccionar Mesa**: El cliente elige una mesa específica (por ejemplo, Mesa 1).
3. **Elegir Tipo de Servicio**: El cliente selecciona el tipo de servicio (comer en el restaurante o para llevar). El sistema confirma la selección.

Servicio en Restaurante

- **Mostrar Menú del Día**: Si el cliente opta por comer en el restaurante, el sistema muestra el menú del día.
- **Selección de Comida**:
  - **Sopa**: Se pregunta al cliente si desea sopa. Si dice que sí, se muestran las opciones disponibles.
  - **Plato Principal**: El cliente selecciona el plato principal.
  - **Postre**: Se pregunta si desea postre. Si elige, se presentan las opciones disponibles.
- **Confirmar Pedido**: El cliente confirma su pedido, y el sistema procede a servirlo.
- **Generación de Factura**: El sistema genera una factura, la cual es procesada por el dueño del restaurante para crear una factura detallada.

Servicio para Llevar

- **Mostrar Menú del Día**: Si el cliente elige comida para llevar, también se le muestra el menú del día.
- **Selección de Comida**:
  - **Sopa**: Igual que en el servicio en restaurante, se pregunta sobre la sopa.
  - **Plato Principal**: El cliente elige su plato principal.
  - **Postre**: Se pregunta por el postre y se muestran las opciones si elige.
- **Desechables**: El sistema pregunta si el cliente necesita desechables. Si responde que sí, se informa sobre un costo adicional; si no, se indica que no hay costo.
- **Confirmar Pedido**: El cliente confirma su pedido, y el sistema prepara el pedido para llevar.
- **Generación de Factura**: Similar al servicio en restaurante, se genera una factura que el dueño del restaurante procesa para crear una factura detallada.

Conclusiones

Este diagrama de secuencia es fundamental para comprender cómo fluye la comunicación durante el proceso de pedido en un restaurante, ya sea para consumo en el lugar o para llevar. Ayuda a identificar pasos críticos y posibles mejoras en la experiencia del cliente.

Diagramas de Estado

```
@startuml estado-transicion
[*] --> SeleccionMesa

SeleccionMesa --> EligeServicio : Mesa seleccionada
EligeServicio --> EscogeComida : Servicio elegido

EligeServicio --> EscogeComidaParaLlevar : Llevar
EligeServicio --> EscogeComidaRestaurante : Comer en restaurante

EscogeComidaRestaurante --> EscogeSopa : Comida seleccionada
EscogeSopa --> EscogePostre : Sopa seleccionada
```

```
EscogeSopa --> RealizaPedido : No sopa

EscogePostre --> RealizaPedido : Postre seleccionado
EscogePostre --> RealizaPedido : No postre

EscogeComidaParaLlevar --> RealizaPedido : Comida seleccionada para llevar

RealizaPedido --> SirveOrden : Pedido realizado
SirveOrden --> GeneraFactura : Orden servida
GeneraFactura --> [*] : Factura generada
@enduml
```

Diagrama de Estados y Transiciones: Proceso de Pedido en un Restaurante

Este diagrama representa los estados y transiciones que ocurren en el proceso de selección y pedido de comida en un restaurante. Cada estado refleja una etapa en la que se encuentra el cliente durante su experiencia.

Descripción del Diagrama

Estados Clave

1. **SeleccionMesa**:
    - Estado inicial donde el cliente elige una mesa disponible.
2. **EligeServicio**:
    - Estado donde el cliente decide el tipo de servicio: para llevar o comer en el restaurante.
3. **EscogeComida**:
    - Estado que se divide en dos caminos según la elección del servicio:
        - **EscogeComidaParaLlevar**: Si el cliente elige llevar.
        - **EscogeComidaRestaurante**: Si elige comer en el restaurante.
4. **EscogeComidaRestaurante**:
    - **EscogeSopa**: El cliente decide si quiere sopa.
        - Si elige sopa, avanza a **EscogePostre**.
        - Si no, va directamente a **RealizaPedido**.
5. **EscogePostre**:
    - El cliente decide si quiere un postre.
        - Si elige un postre, avanza a **RealizaPedido**.
        - Si no, también avanza a **RealizaPedido**.

6. **EscogeComidaParaLlevar**:
    - Si el cliente elige comida para llevar, selecciona la comida y luego avanza a **RealizaPedido**.
7. **RealizaPedido**:
    - Estado donde el cliente confirma su pedido.
8. **SirveOrden**:
    - El estado en el que se sirve el pedido al cliente.
9. **GeneraFactura**:
    - Se genera la factura una vez que la orden ha sido servida.
10. **[*]**:
    - Estado final que indica que el proceso ha concluido tras la generación de la factura.

Conclusiones

Este diagrama de estados y transiciones permite visualizar claramente el flujo del proceso de pedido en un restaurante, desde la selección de la mesa hasta la generación de la factura. Facilita la identificación de las decisiones clave que toma el cliente y los estados que atraviesa, lo que puede ser útil para mejorar la experiencia del cliente y la eficiencia operativa.

Diagramas de Tiempo

```
@startuml Tiempo
title Diagrama de Tiempo del Sistema de Restaurante

participant Cliente
participant Sistema
participant Camarero
participant Dueño

Cliente -> Sistema : SeleccionaMesa()
note right of Cliente : t1: Selecciona mesa
Cliente -> Sistema : EligeServicio(tipo)
note right of Cliente : t2: Elige servicio

alt Comer en el restaurante
    Cliente -> Sistema : EscogeComida()
    note right of Cliente : t3: Escoge comida
    alt ¿Desea sopa?
        Cliente -> Sistema : SeleccionaSopa()
        note right of Cliente : t4: Selecciona sopa
    else
        Cliente -> Sistema : NoSeleccionaSopa()
        note right of Cliente : t4: No selecciona sopa
    end
```

```
    alt ¿Desea postre?
        Cliente -> Sistema : SeleccionaPostre()
        note right of Cliente : t5: Selecciona postre
    else
        Cliente -> Sistema : NoSeleccionaPostre()
        note right of Cliente : t5: No selecciona postre
    end
else Para llevar
    Cliente -> Sistema : EscogeComidaParaLlevar()
    note right of Cliente : t6: Escoge comida para llevar
end

Cliente -> Sistema : RealizaPedido()
note right of Cliente : t7: Realiza pedido
Sistema -> Camarero : TomaPedido()
note right of Sistema : t8: Toma pedido
Camarero -> Cliente : SirveOrden()
note right of Camarero : t9: Sirve orden
Sistema -> Cliente : GeneraFactura()
note right of Sistema : t10: Genera factura
Sistema -> Dueño : EnviaFacturaDetallada()
note right of Sistema : t11: Envia factura detallada
@enduml
```

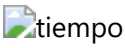


Diagrama de Tiempo del Sistema de Restaurante

Este diagrama ilustra el flujo temporal de interacciones entre el cliente, el sistema, el camarero y el dueño del restaurante durante el proceso de pedido. Cada paso incluye un tiempo asociado que refleja el orden en el que ocurren las acciones.

Descripción del Diagrama

Participantes

- **Cliente**: Usuario que interactúa con el sistema para realizar su pedido.
- **Sistema**: Plataforma que gestiona las interacciones entre el cliente, el camarero y el dueño del restaurante.
- **Camarero**: Empleado encargado de servir al cliente.
- **Dueño**: Persona responsable de la gestión administrativa del restaurante.

Flujo Temporal de Interacciones

1. **Selecciona Mesa** (`t1`):
    - El cliente selecciona una mesa disponible a través del sistema.
2. **Elige Servicio** (`t2`):

- El cliente elige el tipo de servicio que desea (comer en el restaurante o para llevar).

Opción: Comer en el Restaurante

3. **Escoge Comida** (`t3`):
   - El cliente escoge su comida.
   - **¿Desea sopa?**:
     - Si elige sopa, el cliente selecciona una sopa (`t4`).
     - Si no desea sopa, indica que no selecciona sopa (`t4`).
   - **¿Desea postre?**:
     - Si elige postre, el cliente lo selecciona (`t5`).
     - Si no desea postre, indica que no selecciona postre (`t5`).

Opción: Para Llevar

4. **Escoge Comida para Llevar** (`t6`):
   - Si el cliente decide llevar la comida, elige las opciones disponibles.
5. **Realiza Pedido** (`t7`):
   - El cliente confirma su pedido al sistema.
6. **Toma Pedido** (`t8`):
   - El sistema envía el pedido al camarero para su procesamiento.
7. **Sirve Orden** (`t9`):
   - El camarero sirve la orden al cliente.
8. **Genera Factura** (`t10`):
   - El sistema genera la factura una vez que se ha servido la orden.
9. **Envía Factura Detallada** (`t11`):
   - El sistema envía una factura detallada al dueño del restaurante para su gestión.

Conclusiones

Este diagrama de tiempo permite visualizar el flujo de acciones y decisiones en el proceso de pedido en un restaurante, destacando los tiempos asociados a cada paso. Proporciona una herramienta valiosa para optimizar el servicio al cliente y mejorar la eficiencia operativa.

Diagramas de Casos de uso

```
@startuml Casos_De_Uso
left to right direction
actor Cliente
actor Camarero
actor "Dueño del Restaurante" as Dueño
package "Restaurante" {
    usecase "Seleccionar Mesa" as UC1
    usecase "Elegir Servicio" as UC2
    usecase "Escoger Comida" as UC3
    usecase "Seleccionar Sopa" as UC4
    usecase "Seleccionar Postre" as UC5
    usecase "Realizar Pedido" as UC6
    usecase "Servir Orden" as UC7
    usecase "Generar Factura" as UC8
    usecase "Enviar Factura Detallada" as UC9

    Cliente --> UC1
    Cliente --> UC2
    Cliente --> UC3
    UC3 --> UC4 : <<include>>
    UC3 --> UC5 : <<include>>
    Cliente --> UC6
    Camarero --> UC7
    UC6 --> UC8
    Dueño --> UC9
}
@enduml
```

Diagrama de Casos de Uso: Sistema de Restaurante

Este diagrama ilustra los casos de uso del sistema de un restaurante, mostrando las interacciones entre los actores (cliente, camarero y dueño) y las funcionalidades que ofrece el sistema.

Participantes

- **Cliente**: Usuario que interactúa con el sistema para realizar pedidos y seleccionar servicios.
- **Camarero**: Empleado responsable de servir al cliente y gestionar las órdenes.
- **Dueño del Restaurante**: Persona encargada de la administración y gestión del restaurante.

Casos de Uso

Funcionalidades del Sistema

1. **Seleccionar Mesa** (`UC1`):
    - El cliente elige una mesa disponible en el restaurante.
2. **Elegir Servicio** (`UC2`):

    - El cliente decide el tipo de servicio que desea (comer en el restaurante o para llevar).

3. **Escoger Comida** (UC3):
    - El cliente selecciona las opciones de comida disponibles.
    - **Seleccionar Sopa** (UC4):
        - Caso de uso incluido donde el cliente elige si quiere sopa.
    - **Seleccionar Postre** (UC5):
        - Caso de uso incluido donde el cliente elige si quiere postre.
4. **Realizar Pedido** (UC6):
    - El cliente confirma su pedido al sistema.
5. **Servir Orden** (UC7):
    - El camarero sirve la orden al cliente.
6. **Generar Factura** (UC8):
    - El sistema genera una factura una vez que el pedido ha sido realizado.
7. **Enviar Factura Detallada** (UC9):
    - El dueño del restaurante recibe una factura detallada para su gestión administrativa.

Relaciones

- Las relaciones entre los actores y los casos de uso son representadas mediante flechas:
    - El **cliente** interactúa directamente con los casos de uso relacionados con la selección de mesa, servicio, comida y pedidos.
    - El **camarero** se encarga de servir la orden.
    - El **dueño** del restaurante se ocupa de recibir la factura detallada generada por el sistema.

Conclusiones

Este diagrama de casos de uso es fundamental para entender las interacciones y funcionalidades del sistema de un restaurante. Proporciona una visión clara de cómo los diferentes actores utilizan el sistema y cuáles son los procesos clave involucrados en la experiencia del cliente.

Diagramas Estructurales

Diagramas de Clases

```
@startuml Clases
```

```
class Plato {
    - ID_Plato: int
    - Nombre: string
    - Descripción: string
    - Precio: decimal
    - Foto: string
}

class Detalle_Orden {
    - ID_Detalle: int
    - ID_Orden: int
    - ID_Plato: int
    - Cantidad: int
    - Sopa: boolean
    - Postre: boolean
    - Domicilio: boolean
    - Recipiente: boolean
}

class Orden {
    - ID_Orden: int
    - ID_Mesa: int
    - HoraPedido: datetime
    - Total: decimal
}

class Mesa {
    - ID_Mesa: int
    - Numero_Mesa: int
    - Capacidad: int
    - Estado: string
}

class Usuario {
    - ID_Usuario: int
    - Nombre: string
    - Teléfono: string
    - Dirección: string
}

class Factura {
    - ID_Factura: int
    - ID_Orden: int
    - Fecha: date
    - Total: decimal
    - MetodoPago: string
}


Plato o--  Detalle_Orden : contiene >
Orden  --*  Detalle_Orden : compone >
Orden  o--  Mesa : se realiza en >
Usuario  o--  Orden : realiza >
Factura  -- Orden : se genera para >
```

```
@enduml
```

 clases

Este diagrama de clases representa la estructura de un sistema de gestión de restaurante, mostrando las principales entidades y sus relaciones. A continuación, se describen cada una de las clases y sus atributos, así como las relaciones entre ellas.

Clases y Atributos

1. Plato

- *ID_Plato*: int
- *Nombre*: string
- *Descripción*: string
- *Precio*: decimal
- *Foto*: string

Esta clase representa los diferentes platos disponibles en el restaurante. Cada plato tiene un identificador único, nombre, descripción, precio y una foto.

---

2. Detalle_Orden

- *ID_Detalle*: int
- *ID_Orden*: int
- *ID_Plato*: int
- *Cantidad*: int
- *Sopa*: boolean
- *Postre*: boolean
- *Domicilio*: boolean
- *Recipiente*: boolean

Esta clase detalla cada elemento de una orden, indicando qué platos se han pedido, su cantidad, y opciones adicionales como si se incluye sopa, postre, si es para domicilio y si se requiere recipiente.

---

3. Orden

- *ID_Orden*: int
- *ID_Mesa*: int
- *HoraPedido*: datetime
- *Total*: decimal

Representa una orden realizada en el restaurante. Incluye un identificador único, el número de mesa donde se realiza la orden, la hora en que fue pedido y el total de la orden.

---

4. Mesa

- *ID_Mesa*: int
- *Numero_Mesa*: int
- *Capacidad*: int
- *Estado*: string

Esta clase describe las mesas del restaurante. Cada mesa tiene un identificador, un número, capacidad de personas y su estado (disponible, ocupada, reservada).

---

5. Usuario

- *ID_Usuario*: int
- *Nombre*: string
- *Teléfono*: string
- *Dirección*: string

Representa a los usuarios que realizan las órdenes. Incluye un identificador, nombre, teléf

Este diagrama de clases representa la estructura de un sistema de gestión de restaurante, mostrando las principales entidades y sus relaciones. A continuación, se describen cada una de las clases y sus atributos, así como las relaciones entre ellas.

Clases y Atributos

1. Plato

- *ID_Plato*: int
- *Nombre*: string
- *Descripción*: string
- *Precio*: decimal
- *Foto*: string

Esta clase representa los diferentes platos disponibles en el restaurante. Cada plato tiene un identificador único, nombre, descripción, precio y una foto.

---

2. Detalle_Orden

- *ID_Detalle*: int
- *ID_Orden*: int
- *ID_Plato*: int
- *Cantidad*: int
- *Sopa*: boolean
- *Postre*: boolean
- *Domicilio*: boolean
- *Recipiente*: boolean

Esta clase detalla cada elemento de una orden, indicando qué platos se han pedido, su cantidad, y opciones adicionales como si se incluye sopa, postre, si es para domicilio y si se requiere recipiente.

---

3. Orden

- *ID_Orden*: int
- *ID_Mesa*: int
- *HoraPedido*: datetime
- *Total*: decimal

Representa una orden realizada en el restaurante. Incluye un identificador único, el número de mesa donde se realiza la orden, la hora en que fue pedido y el total de la orden.

---

4. Mesa

- *ID_Mesa*: int
- *Numero_Mesa*: int
- *Capacidad*: int
- *Estado*: string

Esta clase describe las mesas del restaurante. Cada mesa tiene un identificador, un número, capacidad de personas y su estado (disponible, ocupada, reservada).

---

5. Usuario

- *ID_Usuario*: int
- *Nombre*: string
- *Teléfono*: string
- *Dirección*: string

Representa a los usuarios que realizan las órdenes. Incluye un identificador, nombre, teléfono y dirección de contacto.

---

6. Factura

- *ID_Factura*: int
- *ID_Orden*: int
- *Fecha*: date
- *Total*: decimal
- *MetodoPago*: string

Esta clase representa la factura generada para una orden específica. Incluye un identificador, la fecha de emisión, el total de la factura y el método de pago utilizado.

Relaciones

- *Plato o-- Detalle_Orden*: Un plato puede estar contenido en múltiples detalles de orden.
- *Orden -- Detalle_Orden**: Una orden se compone de múltiples detalles de orden.
- *Orden o-- Mesa*: Una orden se realiza en una mesa específica.
- *Usuario o-- Orden*: Un usuario realiza una orden.
- *Factura -- Orden*: Se genera una factura para una orden específica.

Este diagrama es esencial para entender cómo interactúan las diferentes entidades dentro del sistema de gestión de un restaurante, permitiendo una mejor organización y seguimiento de las órdenes y los usuarios.

Diagramas de Componentes

```
@startuml Componentes

package "Sistema de Restaurante" {

    [PlatoService] --> [BaseDeDatos] : consulta y actualiza platos
    [OrdenService] --> [BaseDeDatos] : consulta y actualiza órdenes
    [DetalleOrdenService] --> [BaseDeDatos] : consulta y actualiza detalles de
órdenes
    [UsuarioService] --> [BaseDeDatos] : consulta y actualiza usuarios
    [FacturaService] --> [BaseDeDatos] : consulta y actualiza facturas

    [PlatoService] --> [DetalleOrdenService] : proporciona información del plato
    [OrdenService] --> [DetalleOrdenService] : agrega detalles a la orden
    [OrdenService] --> [FacturaService] : genera factura
    [UsuarioService] --> [OrdenService] : usuario realiza una orden
}

@enduml
```

 componentes

Este diagrama de componentes representa un sistema de restaurante que organiza sus servicios en módulos específicos para gestionar platos, órdenes, detalles de órdenes, usuarios y facturas. A continuación, se detallan los roles y las interacciones entre los componentes:

1. *PlatoService*: se encarga de la gestión de platos (consulta y actualización) en la base de datos. Además, proporciona información del plato a DetalleOrdenService para detallar cada plato en una orden específica.

2. *OrdenService*: maneja las órdenes del sistema. Consulta y actualiza las órdenes en la base de datos. Tiene varias interacciones:

    - Agrega detalles a la orden mediante DetalleOrdenService.
    - Colabora con FacturaService para generar la factura de una orden completa.
    - Recibe órdenes iniciadas por los usuarios a través de UsuarioService.

3. *DetalleOrdenService*: gestiona los detalles de cada orden. Consulta y actualiza los detalles en la base de datos, recibe información de platos desde PlatoService, y permite que OrdenService agregue detalles

de cada plato a la orden.

4. *UsuarioService*: se encarga de la administración de los usuarios en la base de datos y permite que los usuarios realicen órdenes, las cuales se envían a OrdenService.

5. *FacturaService*: consulta y actualiza las facturas en la base de datos y trabaja en colaboración con OrdenService para generar la factura cuando se completa una orden.

Diagramas de Despliegue

```
@startuml Despliegue
node "Servidor de Aplicación" {
    component "PlatoService"
    component "OrdenService"
    component "DetalleOrdenService"
    component "UsuarioService"
    component "FacturaService"
}

node "Base de Datos" {
    database "BaseDeDatos" {
        [Plato]
        [Detalle_Orden]
        [Orden]
        [Mesa]
        [Usuario]
        [Factura]
    }
}

node "Cliente" {
    [Interfaz de Usuario]
}

[Interfaz de Usuario] --> [PlatoService] : realiza peticiones
[PlatoService] --> [BaseDeDatos] : consulta y actualiza platos
[OrdenService] --> [BaseDeDatos] : consulta y actualiza órdenes
[DetalleOrdenService] --> [BaseDeDatos] : consulta y actualiza detalles de órdenes
[UsuarioService] --> [BaseDeDatos] : consulta y actualiza usuarios
[FacturaService] --> [BaseDeDatos] : consulta y actualiza facturas

@enduml
```


despliege

Este diagrama de despliegue muestra cómo los componentes de un sistema de restaurante están distribuidos en diferentes nodos, específicamente un Servidor de Aplicación, una Base de Datos, y un Cliente que representa la interfaz de usuario. Aquí están los detalles de cada nodo y sus interacciones:

- *Servidor de Aplicación*: aloja los servicios principales del sistema. Cada servicio es un componente que maneja distintas funcionalidades del sistema:
    - PlatoService: consulta y actualiza la información de platos en la base de datos.
    - OrdenService: gestiona la creación y actualización de órdenes.
    - DetalleOrdenService: administra los detalles específicos de cada plato en una orden.
    - UsuarioService: maneja la información de los usuarios.
    - FacturaService: gestiona la creación y actualización de facturas.
- *Base de Datos*: almacena todas las entidades del sistema en tablas:
    - *Plato*: tabla con la información de los platos.
    - *Detalle_Orden*: tabla que almacena los detalles de cada orden.
    - *Orden*: tabla que guarda las órdenes realizadas.
    - *Mesa*: tabla para la información sobre las mesas en el restaurante.
    - *Usuario*: tabla que contiene los datos de los usuarios.
    - *Factura*: tabla para las facturas generadas.
- *Cliente*: representa la interfaz de usuario a través de la cual los clientes o empleados interactúan con el sistema. La interfaz envía peticiones al PlatoService para acceder o actualizar la información de los platos, y a través de este, puede interactuar con otros servicios del sistema.

Interacciones

- La Interfaz de Usuario realiza peticiones al PlatoService en el servidor de aplicación.
- Cada servicio en el Servidor de Aplicación interactúa con la Base de Datos para consultar y actualizar la información correspondiente a sus responsabilidades.

Diagramas de Objetos

```
@startuml Objetos

object plato  {
    - ID_Plato = 1
    - Nombre = "Ensalada César"
    -  Descripción = "Ensalada fresca con pollo y aderezo César"
    - Precio = 12.50
    - Foto = "url_foto_ensalada"
}

object detalleOrden {
    - ID_Detalle = 1
    - ID_Orden = 1001
    - ID_Plato = 1
    - Cantidad = 2
    - Sopa = false
    - Postre = true
    - Domicilio = false
}
```

```
object orden{
    - ID_Orden = 1001
    - ID_Mesa = 1
    - HoraPedido = "2024-10-30 12:30"
    - Total = 35.00
}

object usuario {
    - ID_Usuario = 1
    - Nombre = "Juan Pérez"
    - Teléfono = "123456789"
    - Dirección = "Calle Falsa 123"
}

object factura {
    - ID_Factura = 5001
    - ID_Orden = 1001
    - Fecha = "2024-10-30"
    - Total = 35.00
    - MetodoPago = "tarjeta de crédito"
}

plato -- detalleOrden : "es parte de"
orden -- detalleOrden : "contiene"
orden -- factura : "genera"
usuario -- orden : "realiza"
detalleOrden -- plato : "incluye el plato"

@enduml
```

 objetos

Este diagrama de objetos muestra instancias específicas de las entidades de un sistema de restaurante, con atributos que representan información detallada en un momento dado. A continuación, se describen los objetos y sus relaciones:

1. *plato*: instancia de un plato, en este caso "Ensalada César", con sus atributos:
    - ID_Plato: identificador único del plato.
    - Nombre: nombre del plato.
    - Descripción: breve descripción del plato.
    - Precio: costo del plato.
    - Foto: URL de la imagen del plato.
2. *detalleOrden*: representa un detalle específico de una orden, con atributos:
    - ID_Detalle: identificador único del detalle de la orden.
    - ID_Orden: identificador de la orden a la que pertenece este detalle.
    - ID_Plato: identificador del plato específico.
    - Cantidad: número de platos pedidos.
    - Sopa, Postre, Domicilio: indican si el detalle incluye sopa, postre o si es un pedido a domicilio.

3. *orden*: instancia de una orden con atributos:
    - ID_Orden: identificador único de la orden.
    - ID_Mesa: número de la mesa asociada a la orden.
    - HoraPedido: momento en que se realizó el pedido.
    - Total: monto total de la orden.
4. *usuario*: representa a un usuario del sistema, con los atributos:
    - ID_Usuario: identificador único del usuario.
    - Nombre, Teléfono, Dirección: información personal del usuario.
5. *factura*: instancia de una factura generada a partir de una orden, con atributos:
    - ID_Factura: identificador único de la factura.
    - ID_Orden: identifica la orden a la que está asociada la factura.
    - Fecha: fecha de emisión de la factura.
    - Total: monto total facturado.
    - MetodoPago: método de pago usado por el cliente.

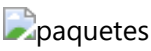
Relaciones

- detalleOrden es "parte de" orden (la orden contiene los detalles de los platos pedidos).
- orden "genera" una factura una vez que el pedido está completo.
- usuario "realiza" una orden.
- detalleOrden "incluye el plato" específico, en este caso, "Ensalada César".

Diagramas de Paquetes

```
@startuml Paquetes
package "Modelo" {
    class Plato {
        - ID_Plato: int
        - Nombre: string
        - Descripción: string
        - Precio: decimal
        - Foto: string
    }

    class Detalle_Orden {
        - ID_Detalle: int
        - ID_Orden: int
        - ID_Plato: int
        - Cantidad: int
        - Sopa: boolean
        - Postre: boolean
        - Domicilio: boolean
        - Recipiente: boolean
    }
```

```
    class Orden {
        - ID_Orden: int
        - ID_Mesa: int
        - HoraPedido: datetime
        - Total: decimal
    }

    class Mesa {
        - ID_Mesa: int
        - Numero_Mesa: int
        - Capacidad: int
        - Estado: string
    }

    class Usuario {
        - ID_Usuario: int
        - Nombre: string
        - Teléfono: string
        - Dirección: string
    }

    class Factura {
        - ID_Factura: int
        - ID_Orden: int
        - Fecha: date
        - Total: decimal
        - MetodoPago: string
    }
}

package "Servicios" {
    class PlatoService
    class OrdenService
    class DetalleOrdenService
    class UsuarioService
    class FacturaService
}

package "Base de Datos" {
    class Database
}

"Modelo" --> "Servicios" : utiliza >
"Servicios" --> "Base de Datos" : accede a >
"Modelo" --> "Base de Datos" : almacena datos en >

@enduml
```

 paquetes

Este diagrama de paquetes representa la estructura general de un sistema de restaurante, dividiendo sus componentes en tres paquetes principales: Modelo, Servicios y Base de Datos. Aquí está la explicación de

cada paquete y sus interacciones:

1. *Paquete* "*Modelo*": contiene las clases que representan las entidades principales del sistema. Estas clases contienen atributos que describen los datos asociados a cada entidad:
    - *Plato*: describe un plato del menú, con atributos como ID_Plato, Nombre, Descripción, Precio, y Foto.
    - *Detalle_Orden*: representa los detalles de cada plato en una orden, con atributos como ID_Detalle, ID_Orden, ID_Plato, Cantidad, y varios booleanos para indicar opciones como Sopa, Postre, Domicilio, y Recipiente.
    - *Orden*: representa una orden realizada por un cliente, con atributos como ID_Orden, ID_Mesa, HoraPedido, y Total.
    - *Mesa*: describe una mesa en el restaurante, con atributos como ID_Mesa, Numero_Mesa, Capacidad, y Estado.
    - *Usuario*: representa a un usuario del sistema, con ID_Usuario, Nombre, Teléfono, y Dirección.
    - *Factura*: representa una factura generada a partir de una orden, con atributos como ID_Factura, ID_Orden, Fecha, Total, y MetodoPago.
2. *Paquete* "*Servicios*": contiene las clases de servicio que manejan la lógica de negocio asociada a cada entidad:
    - *PlatoService*: gestiona las operaciones relacionadas con los platos.
    - *OrdenService*: maneja las operaciones de las órdenes.
    - *DetalleOrdenService*: gestiona los detalles específicos de cada orden.
    - *UsuarioService*: maneja la información de los usuarios.
    - *FacturaService*: gestiona las operaciones relacionadas con las facturas.
3. *Paquete* "*Base de Datos*": contiene la clase Database, que representa la base de datos donde se almacenan todas las entidades del sistema.

Interacciones entre Paquetes

- "*Modelo*" → "*Servicios*": El paquete Modelo es utilizado por el paquete Servicios, lo que significa que los servicios manejan las instancias de las clases del modelo (por ejemplo, OrdenService puede crear y actualizar objetos de la clase Orden).
- "*Servicios*" → "*Base de Datos*": Los servicios acceden a la base de datos para realizar operaciones de lectura y escritura.
- "*Modelo*" → "*Base de Datos*": El paquete Modelo se almacena en la Base de Datos, lo que significa que las clases del modelo representan las entidades que se guardan en la base de datos.

Diagramas de Perfil

```
@startuml

package "Perfil Restaurante" {
    class Plato {
        - Nombre : string
```

```
        - Descripción : string
        - Precio : decimal
        - Foto : string

    }

    class Orden <<stereotype>> {
        - ID_Orden : int
        - HoraPedido : datetime
        - Total : decimal
    }

    class Usuario <<stereotype>> {
        - Nombre : string
        - Teléfono : string
        - Dirección : string
    }


    Plato --> Orden : "incluye"
    Usuario --> Orden : "realiza"
}

@enduml
```

 perfil

Elementos del Diagrama:

1. *Clases*:
    - *Plato*: Define los platos del menú con atributos como:
        - Nombre: El nombre del plato.
        - Descripción: Una breve descripción del plato.
        - Precio: Costo del plato.
        - Foto: URL o referencia a una imagen del plato.
    - *Orden*: Representa una orden hecha por un usuario, con atributos como:
        - ID_Orden: Identificador único de la orden.
        - HoraPedido: Marca de tiempo que indica cuándo se realizó el pedido.
        - Total: Monto total de la orden.
    - *Usuario*: Representa a los clientes del restaurante, con atributos como:
        - Nombre: Nombre del usuario.
        - Teléfono: Número de contacto.
        - Dirección: Dirección del usuario.
2. *Estereotipos*:

    - El estereotipo aplicado a la clase Orden y Usuario indica que estas clases tienen características específicas en el contexto del restaurante, pero no se definen roles adicionales en este diagrama.

Relaciones:

- Plato --> Orden: La relación "incluye" indica que una orden puede contener uno o más platos. Esto refleja la naturaleza de un sistema de pedidos, donde un cliente puede pedir varios platos a la vez.
- Usuario --> Orden: La relación "realiza" indica que un usuario (cliente) puede realizar una o más órdenes. Esto muestra que cada pedido está asociado con un cliente específico.

El diagrama de perfil representa las clases principales de un sistema de restaurante, sus atributos y las relaciones entre ellos. Es útil para entender la estructura del sistema y cómo interactúan los diferentes componentes en el proceso de realizar pedidos.