Diagrama de clases UML

–

Aplicación de pedidos de comida a domicilio

Entorns de desenvolupament
1 DAM

Hector Rodríguez Lozano

Índice

Introducción

En esta actividad se ha diseñado un diagrama de clases para una aplicación de pedidos de comida a domicilio, similar a plataformas como Glovo, Uber Eats o Just Eat. El objetivo es gestionar la información de los clientes, restaurantes, productos y pedidos

Para ello usaré UML, un lenguaje de modelo unificado, es una herramienta muy utilizada en el desarrollo de software para representar de forma visual cómo se estructura y funciona un sistema. Sirve para diseñar y entender mejor la arquitectura de una aplicación antes de empezar a programarla, facilitando la comunicacion entre los miembros del equipo y ayudando a detectar errores o mejoras

En este trabajo se ha utilizado UML para organizar y definir las principales entidades mostrando sus atributos, métodos y relaciones entre ellas

Definición de las clases

Al analizar cómo funciona una aplicación de pedidos de comida a domicilio son necesarias estas clases:

-Cliente: es el usuario que utiliza la aplicación, es decir, la persona que realiza pedidos

-Restaurante:representa a un negocio que ofrece productos

-Producto: representa un alimento disponible para ser pedido

-Pedido: representa un conjunto de productos que el cliente solicita a un restaurante en una fecha

Definición de atributos y métodos

Descripción de los atributos y métodos de cada clase a detalle:

Cliente

Atributos:

-Nombre: nombre completo del cliente (String)

-Correo: almacena el correo electrónico del cliente, que puede servir para iniciar sesión(String)

-Dirección: es la dirección donde se realizan los envíos(String)

-Pedidos: es una lista con los pedidos realizados por el cliente(List de pedido)

Métodos:

-Registrarse(): permite a un cliente registrarse en la aplicación utilizando sus datos

-RealizarPedido(restaurante, productos): permite al cliente realizar un nuevo pedido indicando un restaurante y una lista de productos.

-TodosMisPedidos(): devuelve todos los pedidos realizados por el cliente

Relaciones:

-Un cliente puede tener muchos pedidos (1, N), pero cada pedido solo pertenece a un cliente (1,1)

Restaurante

Atributos:

-Nombre: indica el nombre del restaurante (String)

-Ubicación: indica la dirección en la que se encuentra el restaurante (String)

-Productos: es una lista con los productos disponibles en el restaurante (List de productos)

Métodos:

-AgregarProducto(producto): permite añadir un nuevo producto al menú del restaurante

-EliminarProducto(producto): permite eliminar un producto del menú del restaurante

-TodosLosProductos(): devuelve la lista de productos disponibles

Relaciones:

-Un restaurante ofrece muchos productos (1, N) y cada producto pertenece a un solo restaurante, si no hay restaurante no hay producto por eso el rombo

-Un restaurante tiene muchos pedidos (1,N), pero un pedido sólo puede ser de un restaurante (1,1

Producto

Atributos:

-Nombre: el nombre del producto (String)

-Precio: el precio del producto (double)

-Descripción: una descripción que puede incluir información nutricional,ingredientes y alérgenos (String)

Métodos:

-MostrarDetalles(): muestra todos los datos del producto

-NuevoPrecio(precio double): actualiza el precio del producto

Relación:

-Un producto es de un único restaurante (1,1) y un producto puede estar en ningún o muchos pedidos (0,N)

Pedido

Atributos:

-Fecha: la fecha y hora en la que se realiza el pedido(Date)

-Restaurante: referencia al restaurante al que pertenece el pedido(restaurante)

-Cliente: referencia al cliente que ha hecho el pedido(cliente)

-Productos: una lista de los productos solicitados en el pedido(list de productos)

Métodos:

-CalcularTotal(): devuelve el precio total del pedido sumando el precio de todos los productos

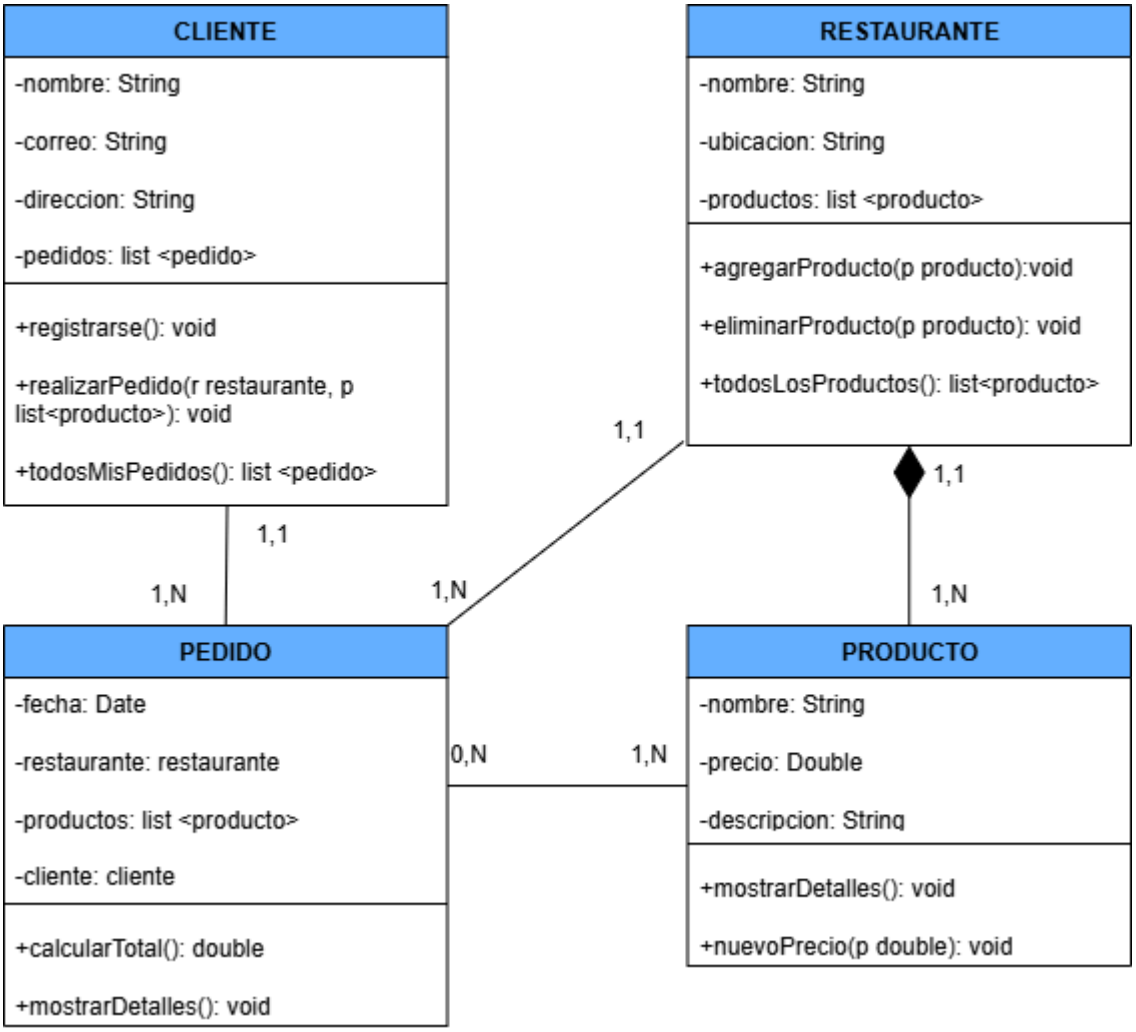
-MostrarDetalles(): muestra toda la información del pedido

Relación:

-Cada pedido está relacionado con un solo restaurante y un solo cliente (1,1)

-Un pedido puede contener varios productos (1, N

Diagrama de clases



Conclusiones

Con esta actividad he aprendido a diseñar un diagrama de clases UML aplicado a una aplicación real, como las de pedidos de comida a domicilio. Me ha servido para entender mejor cómo organizar la información y cómo se relacionan las distintas partes de un programa, también he visto la importancia de definir bien los atributos y métodos de cada clase antes de empezar a programar
En general, creo que UML es una herramienta muy útil para planificar bien una aplicación y evitar errores.
Para hacer el diagrama he utilizado la herramienta draw.io y me he guiado con un vídeo explicativo Tutorial - Diagrama de Clases UML