Лабораторна робота № 1

Варіант № 18

з дисципліни «Конструювання та документація програмного забезпечення»

На тему: «Проєктування в конструюванні програмного забезпечення.»

| Виконав:  студент Б-121-23-4-ПІ  Орлов Антон Костянтинович  Допущено до захисту | Перевірив:  доцент кафедри ІПЗ  Дмитро Олександрович Гололобов  Захищено з оцінкою |
| --- | --- |

**Лабораторна робота 1**

**Тема роботи:** Проєктування в конструюванні програмного забезпечення.

**Мета роботи:** Розробити UML-діаграми застосунку відповідно до

індивідуального завдання.

**Теоретичні відомості:** див. відповідні лекції.

**Практична частина**

Для застосунку, розроблюваного відповідно до індивідуального завдання:

1. Описати призначення та функціонал відповідно до предметної області.

2. Спроєктувати класи, їх атрибути та методи, навести їх опис.

3. Побудувати діаграму класів.

4. Побудувати не менше двох діаграм об'єктів.

5. Побудувати діаграму поведінки (на вибір, але таку, щоб відносилась до

конструювання).

**Хід роботи**

Варіант – Автоматизована система піцерії

1. Призначення та функціонал системи

Призначення: Автоматизована система піцерії призначена для управління замовленнями, клієнтами, меню, доставкою та платежами в піцерії.

Основні функції:

Управління клієнтами (додавання, пошук, оновлення)

Управління меню піц (додавання, видалення, перегляд)

Обробка замовлень (створення, додавання позицій, оновлення статусу)

Управління доставкою (призначення кур'єрів, оновлення статусу)

Обробка платежів (готівка, картка, онлайн)

Генерація звітів (продажі, популярні піци)

1. Клас Customer (Клієнт)

Призначення: Моделює клієнта піцерії, зберігає його дані.  
Атрибути:

CustomerId (int) — унікальний ідентифікатор клієнта.

Name (string) — ім'я клієнта.

Phone (string) — номер телефону (обов'язкове поле).

Email (string) — електронна пошта.

Address (string) — адреса доставки.

Методи:

UpdateCustomer() — оновлює інформацію про клієнта.

GetCustomerInfo() — повертає рядок з усіма даними клієнта.

ToString() — перевизначений метод для виведення імені та телефону.

Клас Pizza (Піца)

Призначення: Описує піцу в меню, її характеристики.  
Атрибути:

PizzaId (int) — унікальний ідентифікатор.

Name (string) — назва піци (наприклад, "Маргарита").

Ingredients (List<string>) — список інгредієнтів.

Size (enum PizzaSize) — розмір (Small, Medium, Large).

BasePrice (double) — базова ціна для розміру Medium.

CookingTime (int) — час приготування у хвилинах.

Методи:

CalculatePrice() — розраховує ціну залежно від розміру (наприклад, Large = +30% до базової).

AddIngredient() / RemoveIngredient() — додає або видаляє інгредієнт.

ToString() — виводить назву, розмір, ціну та інгредієнти.

Клас OrderItem (Позиція замовлення)

Призначення: Описує одну позицію в замовленні (піцу + кількість).  
Атрибути:

Pizza (об'єкт Pizza) — піца, яку замовили.

Quantity (int) — кількість одиниць.

Subtotal (double) — вартість позиції (автоматично розраховується).

Методи:

CalculateSubtotal() — множить ціну піци на кількість.

ToString() — виводить назву піци, кількість та підсумкову вартість.

Клас Order (Замовлення)

Призначення: Моделює замовлення клієнта.  
Атрибути:

OrderId (int) — унікальний номер замовлення.

Customer (об'єкт Customer) — хто замовив.

OrderItems (List<OrderItem>) — список позицій.

OrderDate (DateTime) — дата та час створення.

Status (enum OrderStatus) — стан (Pending, Cooking, Ready, Delivered).

TotalAmount (double) — загальна сума (автоматичний розрахунок).

Методи:

AddOrderItem() / RemoveOrderItem() — додає або видаляє позицію.

CalculateTotal() — підраховує загальну вартість.

UpdateStatus() — змінює статус замовлення.

ToString() — виводить деталі замовлення (клієнт, список піц, суму).

Клас Payment (Платіж)

Призначення: Відповідає за обробку оплати замовлення.  
Атрибути:

PaymentId (int) — унікальний ідентифікатор.

Order (об'єкт Order) — пов'язане замовлення.

Amount (double) — сума оплати.

PaymentMethod (enum PaymentMethod) — спосіб (Cash, Card, Online).

PaymentDate (DateTime) — дата транзакції.

Status (enum PaymentStatus) — стан (Pending, Completed, Failed).

Методи:

ProcessPayment() — імітує оплату (90% успіху для демонстрації).

Refund() — повертає кошти.

ToString() — виводить ID, суму, спосіб оплати та статус.

Клас Delivery (Доставка)

Призначення: Керує доставкою замовлення.  
Атрибути:

DeliveryId (int) — унікальний ідентифікатор.

Order (об'єкт Order) — замовлення для доставки.

Courier (string) — ім'я кур'єра.

DeliveryAddress (string) — адреса (береться з Customer.Address).

EstimatedTime (int) — очікуваний час доставки (хвилини).

ActualTime (int) — реальний час (заповнюється після доставки).

Status (enum DeliveryStatus) — стан (Assigned, InProgress, Delivered).

CreatedAt (DateTime) — час створення доставки.

Методи:

AssignCourier() — призначає кур'єра.

UpdateDeliveryStatus() — змінює статус (якщо Delivered, записує ActualTime).

CalculateDeliveryTime() — вираховує час доставки (базовий час + час приготування найдовшої піци).

ToString() — виводить ID, адресу, кур'єра та статус.

Клас PizzeriaSystemCore (Ядро системи)

Призначення: Координує всі компоненти системи.  
Атрибути:

Списки об'єктів: Customers, Menu, Orders, Payments, Deliveries.

Лічильники ID (для генерації унікальних значень).

Методи:

Управління даними:

AddCustomer(), FindCustomerByPhone() — робота з клієнтами.

AddPizzaToMenu(), RemovePizzaFromMenu() — управління меню.

CreateOrder(), ProcessOrder() — обробка замовлень.

AssignCourierToDelivery(), UpdateDeliveryStatus() — доставка.

Звіти:

GenerateSalesReport() — продажі за період.

GeneratePopularPizzasReport() — топ-10 піц.

DisplaySystemStats() — статистика системи (кількість замовлень тощо).

Перелік Enum-ів (перелічувані типи)

PizzaSize: Small, Medium, Large.

OrderStatus: Pending, Cooking, Ready, Delivered.

PaymentMethod: Cash, Card, Online.

PaymentStatus: Pending, Completed, Failed.

DeliveryStatus: Assigned, InProgress, Delivered.

Кожен enum має коментарі з українськими відповідниками (наприклад, Pending — "Очікує обробки").

Взаємозв'язки класів

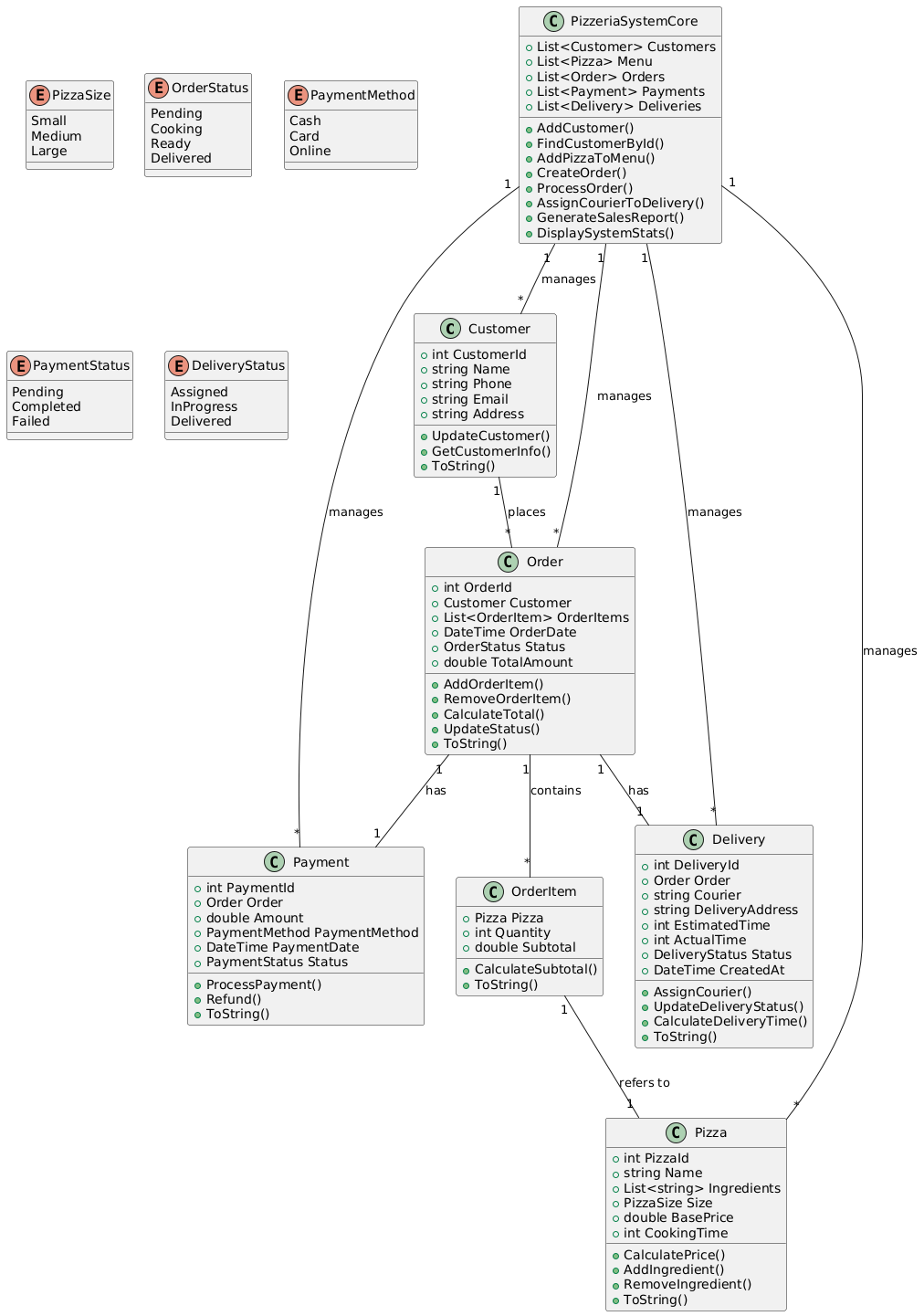
Order містить список OrderItem і посилається на Customer.

Payment та Delivery прив'язані до Order.

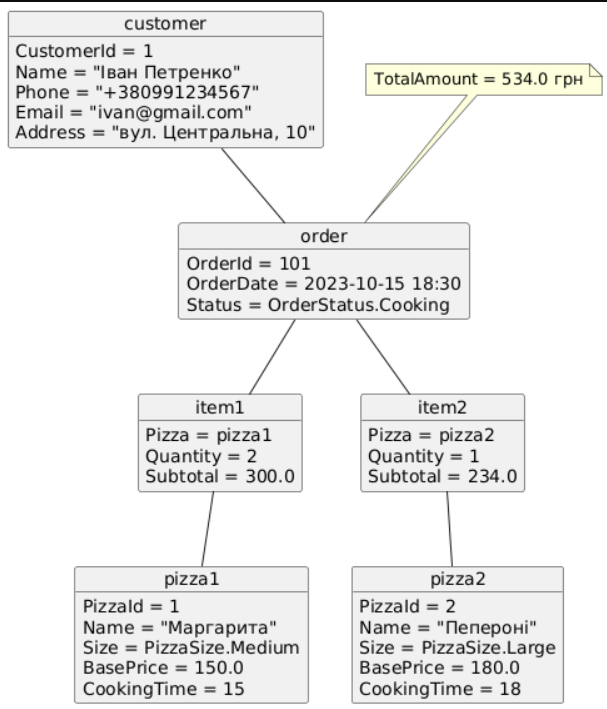
PizzeriaSystemCore керує всіма списками об'єктів.

Ця структура дозволяє ефективно моделювати роботу піцерії: від створення замовлення до доставки та аналітики.

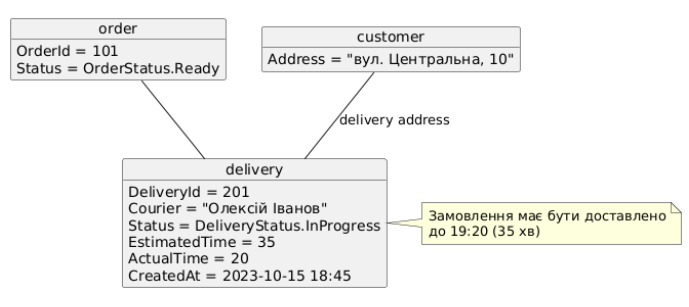
1. Діаграма класів:



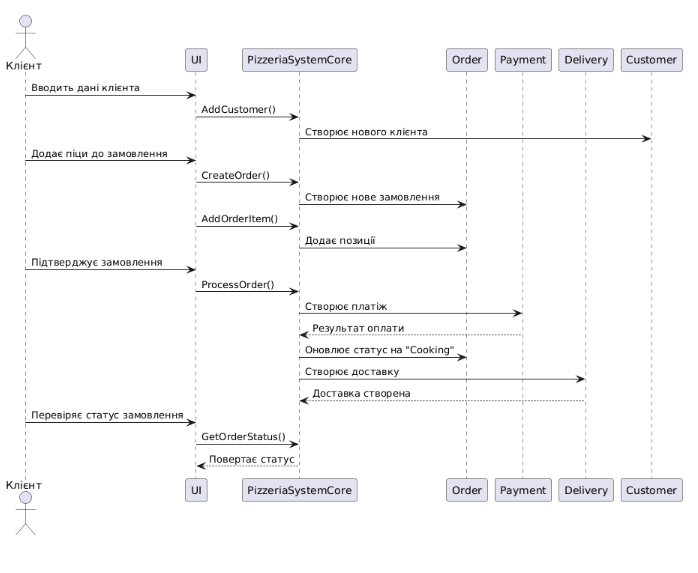
1. Діаграма об’єктів: замовлення.



Діаграма об’єктів: доставка.



1. Діаграма поведінки (діаграма послідовності для процесу замовлення).



**Висновок**

У результаті виконання лабораторної роботи було спроєктовано автоматизовану систему для піцерії, яка охоплює всі основні аспекти її функціонування: управління клієнтами, меню, замовленнями, оплатою та доставкою. Була побудована діаграма класів, яка відображає структуру системи, а також діаграми об'єктів і поведінки, що демонструють взаємодію між компонентами на прикладі реальних сценаріїв.

Отримані результати засвідчують важливість попереднього проектування в процесі конструювання програмного забезпечення. Застосування UML-діаграм дозволяє наочно подати архітектуру застосунку, полегшує подальшу реалізацію та тестування. Робота продемонструвала навички аналізу предметної області, проектування класів та моделювання взаємодій у межах програмної системи.