1. ДІАГРАМИ ПРОГРАМНИХ КЛАСІВ

Через аналіз відношень класів, самих змісту пунктів першого та другого прецеденту, а також проектних рішень, були сформульовані наступні діаграма на рисунку 6.1, 6.2, 6.3, 6.4, 6.5, 6.6, 6.7, 6.8 та 6.9.

6.1 Специфікація програмного класу Money

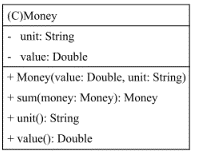


Рисунок 6.1  - Специфікація програмного класу Money

6.2 Специфікація програмного класу Operator

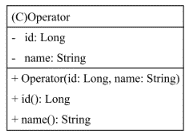


Рисунок 6.2  - Специфікація програмного класу Operator

6.3 Специфікація програмного класу Order

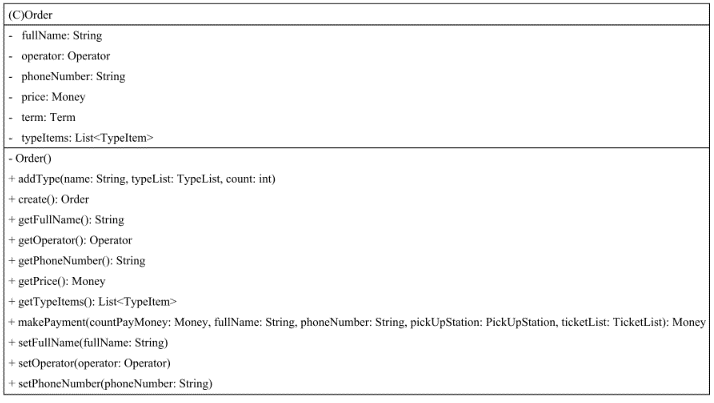


Рисунок 6.3  - Проектне рішення Order

6.4 Специфікація програмного класу OrderIDAndOddMoney

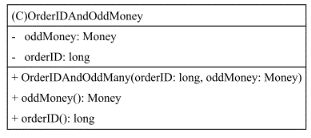


Рисунок 6.4  - Проектне рішення OrderIDAndOddMoney

6.5 Специфікація програмного класу PickUpStation

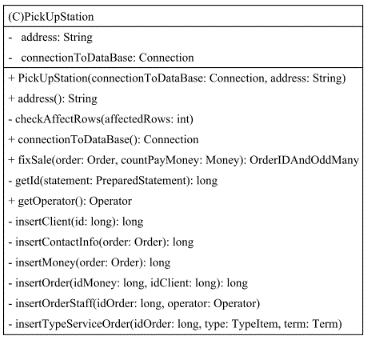


Рисунок 6.5  - Специфікація програмного класу TypeItem

6.6 Специфікація програмного класу Register

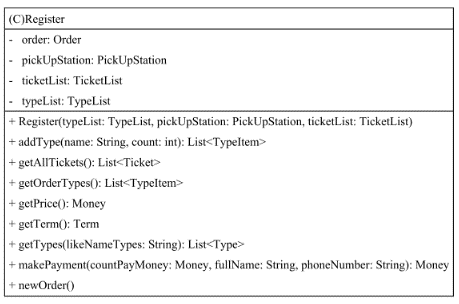


Рисунок 1.6  - Специфікація програмного класу Register

6.7 Специфікація програмного класу Term

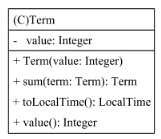


Рисунок 6.7  - Специфікація програмного класу Term

6.8 Специфікація програмного класу Ticket

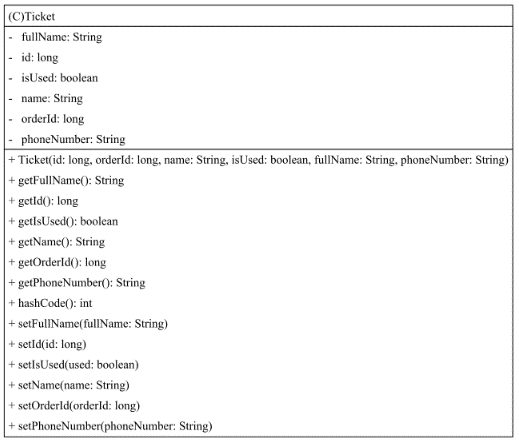


Рисунок 6.8  - Специфікація програмного класу Ticket

6.9 Специфікація програмного класу TicketList

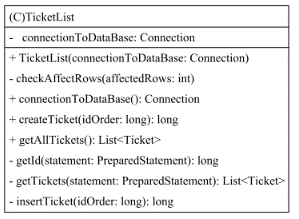


Рисунок 6.9  - Специфікація програмного класу TicketList

6.10 Специфікація програмного класу Type

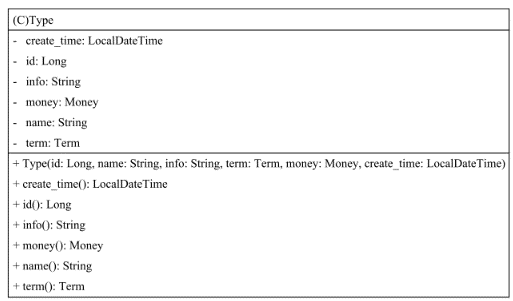


Рисунок 6.10  - Специфікація програмного класу Type

6.11 Специфікація програмного класу Type

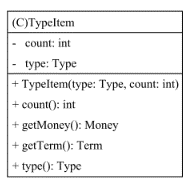


Рисунок 6.11  - Специфікація програмного класу Type

6.12 Специфікація програмного класу TypeList

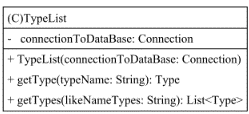


Рисунок 6.12  - Специфікація програмного класу TypeList

6.13 обґрунтування прийнятих рішень, щодо визначення методів і атрибутів класів; Діаграма програмних класів.

Для класу Register методи випливають із системних операцій, діаграм взаємодій та необхідності у створенні екземпляра класу. Атрибути так само з діаграм взаємодій. Цей клас є точкою входу даного модуля та виконання прецеденту.

Для класу Order методи з атрибутами випливають із діаграм взаємодій, а типи з пункту варіанти використання прецеденту. Методи доступу необхідні подальшого використання інших частинах системи, які з даного модуля.

Для класу PickUpStation методи представлятимуться так само з діаграм взаємодій. Метод конструктор необхідний створення екземпляра класу, атрибут необхідний визначення шляху бази даних. Загалом способи представляють функції фіксації та доступу до вільного Оператора.

Для класу TypeList методи представлятимуться так само з діаграм взаємодій. Метод конструктор необхідний створення екземпляра класу, атрибут необхідний визначення шляху бази даних. Функціонал методів являє собою видачу самого типу або часу виготовленняданого типу.

Для класу Term методи представлятимуться так само з діаграм взаємодій. Метод статичний необхідний створення екземпляра класу, атрибут необхідний зберігання асоційованих даних. Функція способу надання тривалості.

Для класу Type методи з атрибутами випливають із діаграм взаємодій, а типи з пункту варіанти використання прецеденту. Методи доступу необхідні подальшого використання інших частинах системи, які з даного модуля.

Для класу Money методи з атрибутами випливають із діаграм взаємодій, а типи з пункту варіанти використання прецеденту. Методи доступу необхідні подальшого використання інших частинах системи, які з даного модуля.

Для класу Operator методи з атрибутами випливають із діаграм взаємодій, а типи з пункту варіанти використання прецеденту. Методи доступу необхідні подальшого використання інших частинах системи, які з даного модуля.

Для класу OrderIDAndOddMoney методи з атрибутами випливають із діаграм взаємодій, а типи з пункту варіанти використання прецеденту. Методи доступу необхідні подальшого використання інших частинах системи, які з даного модуля.

Для класу Ticket методи з атрибутами випливають із діаграм взаємодій, а типи з пункту варіанти використання прецеденту. Методи доступу необхідні подальшого використання інших частинах системи, які з даного модуля.

Для класу TicketList методи представлятимуться так само з діаграм взаємодій. Метод конструктор необхідний створення екземпляра класу, атрибут необхідний визначення шляху бази даних. Функціонал методів являє собою видачу самого типу або часу виготовленняданого типу.

Через аналіз відношень класів, самих змісту пунктів першого та другого прецеденту, проектних рішень, а також специфікацій було отримано наступна діаграма програмних класів на рисунку 6.13.

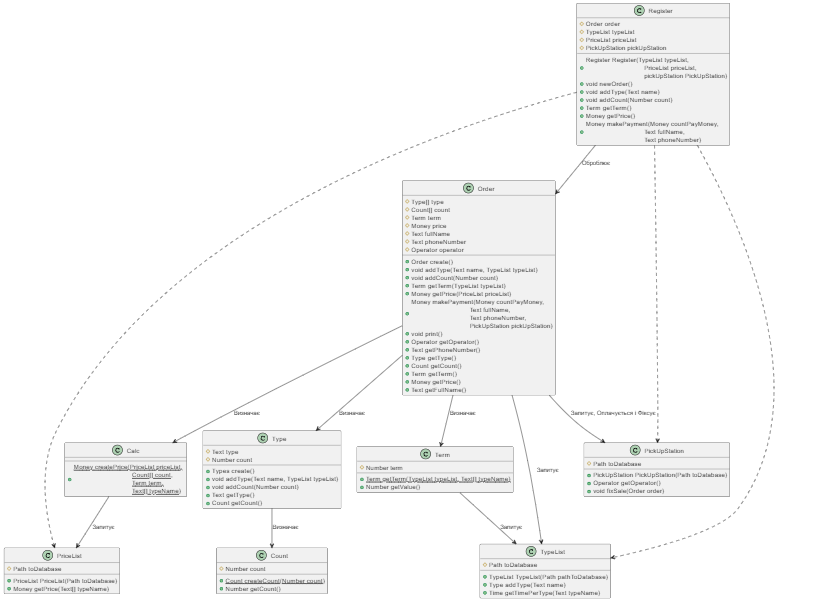


Рисунок 6.13  - Діаграма програмних класів

6.14 обґрунтування виявлених відношень між класами:

Для класу Register методи випливають із системних операцій, діаграм взаємодій та необхідності у створенні екземпляра класу. Атрибути так само з діаграм взаємодій. Цей клас є точкою входу даного модуля та виконання прецеденту. Він потребує об’єкти класів інфо-експертів для використання саме у цієї сесії регістрації замовлення.

Для класу Order методи з атрибутами випливають із діаграм взаємодій, а типи з пункту варіанти використання прецеденту. Методи доступу необхідні подальшого використання інших частинах системи, які з даного модуля. Він потребує об’єкти класів з якими він прямо взаємодіє. З даних об’єктів він тільки потребує інформацію, яка відповідає атрибутам замовлення.

Для класу PickUpStation методи представлятимуться так само з діаграм взаємодій. Метод конструктор необхідний створення екземпляра класу, атрибут необхідний визначення шляху бази даних. Загалом способи представляють функції фіксації та доступу до вільного Оператора, оплата та фіксування замовлення.

Для класу TypeList методи представлятимуться так само з діаграм взаємодій. Метод конструктор необхідний створення екземпляра класу, атрибут необхідний визначення шляху бази даних. Функціонал методів являє собою видачу самого типу або часу виготовлення даного типу.

Для класу PriceList методи представляються так само з діаграм взаємодій. Метод конструктор необхідний створення екземпляра класу, атрибут необхідний визначення шляху бази даних. Функція способу надання ціни за вибраний тип.

Для класу Count методи представлятимуться так само з діаграм взаємодій. Метод конструктор необхідний створення екземпляра класу, атрибут необхідний зберігання асоційованих даних. Функція способу надання кількості.

Для класу Term методи представлятимуться так само з діаграм взаємодій. Метод статичний необхідний створення екземпляра класу, атрибут необхідний зберігання асоційованих даних. Функція способу надання тривалості.

Для класу Calc метод представлятиметься так само з діаграм взаємодій. Метод статичний необхідний надання відповідної підсумкової ціни. Він потребує взаємодіяти з інфо-експертом бази цін.

Для класу Type методи з атрибутами випливають із діаграм взаємодій, а типи з пункту варіанти використання прецеденту. Методи доступу необхідні подальшого використання інших частинах системи, які з даного модуля. Він потребує об’єкти класів з якими він прямо взаємодіє. З даних об’єктів він тільки потребує інформацію, яка відповідає атрибутам типу послуги.