6 ДІАГРАМИ ПРОГРАМНИХ КЛАСІВ

Діаграма окремих класів

Через аналіз відношень класів, самих змісту пунктів першого та другого прецеденту, а також проектних рішень, були сформульовані наступні діаграма на рисунку 1.1, 1.2, 1.3, 1.4, 1.5 та 1.6.

Специфікація програмного класу Register

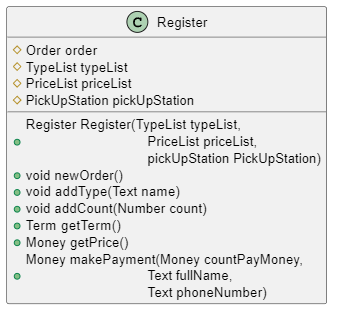


Рисунок 1.1  - Специфікація програмного класу Register

Специфікація програмного класу Order

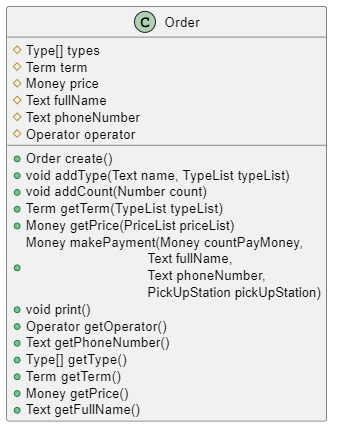


Рисунок 1.2  - Специфікація програмного класу Order

Специфікація програмного класу PickUpStation

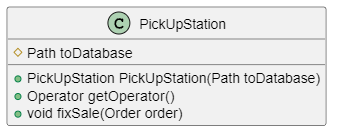


Рисунок 1.3  - Проектне рішення PickUpStation

Специфікація програмного класу TypeList

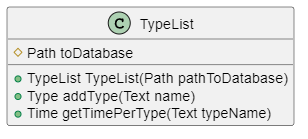


Рисунок 1.4  - Проектне рішення TypeList

Специфікація програмного класу Term

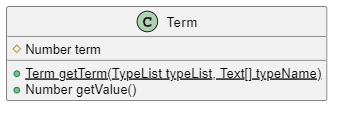


Рисунок 1.5  - Специфікація програмного класу Term

Специфікація програмного класу PriceList

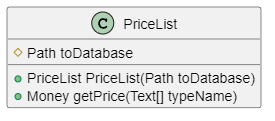


Рисунок 1.6  - Специфікація програмного класу PriceList

Специфікація програмного класу Count

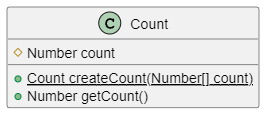


Рисунок 1.7  - Специфікація програмного класу Count

Специфікація програмного класу Calc

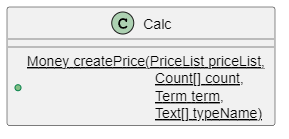


Рисунок 1.8  - Специфікація програмного класу Calc

Специфікація програмного класу Type

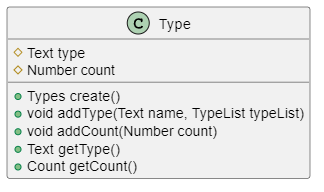


Рисунок 1.8  - Специфікація програмного класу Type

2 обґрунтування прийнятих рішень, щодо визначення методів і атрибутів 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 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.

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.

Через аналіз відношень класів, самих змісту пунктів першого та другого прецеденту, проектних рішень, а також специфікацій було отримано наступна діаграма програмних класів на рисунку 1.

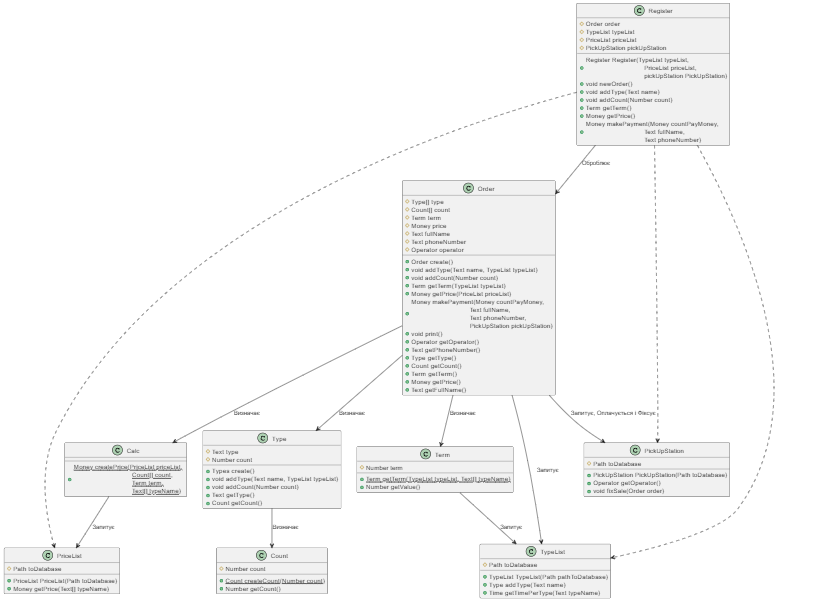


Рисунок 1  - Діаграма програмних класів

2 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.

Загальна діаграма