**МОДУЛЬ 2**

***Лабораторна робота 2.1***

**ВІДОКРЕМЛЕННЯ РІВНЯ ДОСТУПУ ДО ДАНИХ (DAL) З ВИКОРИСТАННЯМ ORM**

**Завдання**

1. Спроектувати застосування, представити у вигляді UML діаграми(-ам) функціонал згідно з варіантом.
   1. Представити загальну архітектуру застосування.
   2. Представити сутності ORM у вигляді діаграми класів.
2. Розробити застосування на мові C#, яка відповідає вимогам у варіанті. Відокремити рівень доступу до даних. Для демонстрації роботи можна використати будь-який UI: консольний ввід-вивід, WinForms, WPF чи Web.
   1. В окрему бібліотеку виділити рівень доступу до даних. Використати Entity Framework code first.
   2. Функціонал застосування (бізнес логіка) може знаходитися в тому ж проекті, що і UI. При цьому бізнес логіка повинна бути окремо від UI операцій. Якщо використовується консольне застосування, то розрахунки, пошук, фільтри чи інші бізнес операції не повинні бути безпосередньо в самому методі, що виводить дані на екран. Якщо GUI, то ці операції не повинні бути в методах, які виконуються при певних діях користувача (наприклад, натискання на кнопку). UI повинен звертатися до методів бізнес логіки.
   3. UI повинен бути максимально простим, неперевантаженим великою кількістю операцій.
3. Діаграма(-и) та вихідний код повинні відповідати основним принципам проектування: OOP, SOLID, Law of Demeter (LoD), DRY, YAGNI, KISS, cohesion – coupling, inheritance with caution.
4. Для отримання балів, що відповідають «задовільно» необхідно реалізувати завдання, відповідно варіанту (табл. 1) без використання зв’язків між сутностями та без використання репозиторію Git. Обов’язково повинні бути реалізовані CRUD операції.
5. Для отримання балів, що відповідають «добре» необхідно у реалізацію завдання з табл. 1 додати імплементацію зв’язків між сутностями (1:1, 1:М, М:N) та використовувати репозиторій Git. Продемонструвати та пояснити дерево комітів та використані гілки. Локальний репозиторій повинен бути обов’язково зв’язаний з віддаленим приватним репозиторієм.
6. Для отримання балів, що відповідають «відмінно» необхідно реалізувати завдання у варіанті з табл. 1 зі зв’язками між сутностями (як для «добре»), а також використати та пояснити «жадібне» та «ліниве» завантаження. Обов’язковим є використання (чи власна реалізація) одного та більше шаблонів проектування (GoF). Також залишаються вимоги з попереднього пункту, що стосуються репозиторію Git.

**Методичні рекомендації**

При виконанні роботи слід пам’ятати (незалежно від рівня):

* різні характеристики/ сутності та специфічні операції не повинні бути в одному класі/ модулі;
* функціонал повинен бути легко розширюваним без необхідності внесення змін в існуючий код (чи при мінімальних змінах) – якщо в завданні не сказано протилежне;
* при необхідності реалізації однотипних операцій варто використати абстракцію;
* код не повинен повторюватися і бути читабельним;
* повинно бути реалізовано тільки те, що вказане в завданні і ніякого зайвого функціоналу.

**Вимоги по реалізації застосування**

В лабораторній роботі необхідно виокремити рівень доступу до даних, а також не «змішувати» логіку роботи програми з інтерфейсом користувача.

Головним критерієм якості є ізоляція DAL, реалізація функціоналу та відповідність базовим принципам проектування. Для побудови залежностей між класами та їх об’єктами використовувати залежності, асоціації (в тому числі композиції та агрегації), наслідування (узагальнення) та реалізації.

За необхідності можна додавати нові властивості, сутності на рівні DAL та розширювати функціонал бізнес-логіки та UI. Але ця необхідність повинна бути обґрунтована.

Управління та відповідь застосування здійснюється через UI. Використання UI, наприклад, для виведення повідомлень, потрібно винести за межі функціоналу предметної області. Тобто, взаємодія з користувачем повинна бути за межами методів, що виконують певні дії.

Методи класів предметної області мають змінювати певним чином стан об’єкту, а не просто виводити повідомлення. Виконання операцій повинно здійснюватися тільки при настанні певних умов.

Для демонстрації роботи програми необхідно підготувати набір вхідних даних.

**Варіанти**

|  |  |  |
| --- | --- | --- |
| Номер  варіанту | Предметна область | Вимоги та компоненти |
| 7 | Доставка їжі | меню з розділенням по днях, видах страв (страви в різні дні можуть повторюватися). Можна замовити комплексний обід або зібрати обід з доступних страв. Реалізувати основні операції по пошуку та замовленню страв. |

**Контрольні запитання**

1. Що таке багаторівнева архітектура ПЗ та які переваги її використання?
2. Які існують шаблони проектування для організації шару доступу до сховищ даних?
3. Які типи сховищ даних існують?
4. Що таке ОRM?
5. Поясніть ціль використання технології Entity Framework (EF) в архітектурі програмних систем.
6. Наведіть основні складові EF.
7. Поясніть призначення класів сутностей (entities) у EF?
8. Поясніть призначення класу контексту (context) у EF?
9. Які підходи до створення сутностей EF існують? В чому їх переваги та недоліки?
10. Що таке .edmx-модель? Яким чином вона створюється та описується?
11. Поясніть принцип роботи підходу code first.
12. Поясніть призначення утиліти Nuget package.
13. Як працюють операції CRUD у EF?
14. Що таке «жадібне» та «ліниве» завантаження?
15. Як реалізуються зв’язки 1:1, 1:М, М:N між сутностями EF? Навести приклади, використовуючи класи-сутності, розроблені у проекті.