МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра інформаційних систем та мереж

Лабораторна робота №7

з дисципліни

СПЕЦІАЛІЗОВАНІ МОВИ ПРОГРАМУВАННЯ

на тему

Робота з API та веб-сервісами

Виконав:

студент групи РІ-21сп

Богдан Д.А.

Прийняв:

Шербак С.С.

Львів-2024

**Мета лабораторної роботи:** Створення консольного об’єктно - орієнтованого додатка з використанням API та патернів проектування

**Завдання**

**Завдання 1: Вибір провайдера API та патернів проектування.** Виберіть надійний API, який надає через HTTP необхідні дані для віддаленого зберігання, вивантаження або реалізуйте свій. Для прикладу це може бути jsonplaceholder.org. Крім того, оберіть 2-3 патерна проектування для реалізаціі імплементаціі цієї лабораторноі роботи. Для прикладу, це може бути патерн Unit of Work та Repository.

**Завдання 2: Інтеграція API.** Виберіть бібліотеку для роботи з API та обробки HTTP запитів (для прикладу це може бути бібліотека Requests). Інтегруйте обраний API в ваш консольний додаток на Python. Ознайомтеся з документацією API та налаштуйте необхідний API-ключ чи облікові дані.

**Завдання 3: Введення користувача.** Розробіть користувальницький інтерфейс, який дозволяє користувачам візуалізувати всі доступні дані в табличному вигляді та у вигляді списку. Реалізуйте механізм для збору та перевірки введеного даних користувачем.

**Завдання 4: Розбір введення користувача.** Створіть розбірник для видобування та інтерпретації виразів користувача на основі регулярних виразів, наприклад, для візуалізації дат, телефонів, тощо. Переконайтеся, що розбірник обробляє різні формати введення та надає зворотний зв'язок про помилки.

**Завдання 5: Відображення результатів.** Реалізуйте логіку для візуалізації даних через API в консолі. Обробляйте відповіді API для отримання даних у вигляді таблиць, списків. Заголовки таблиць, списків мають виділяться кольором та шрифтом, які задається користувачем.

**Завдання 6: Збереження даних.** Реалізуйте можливості збереження даних у чіткому та читабельному форматі JSON, CSV та TXT.

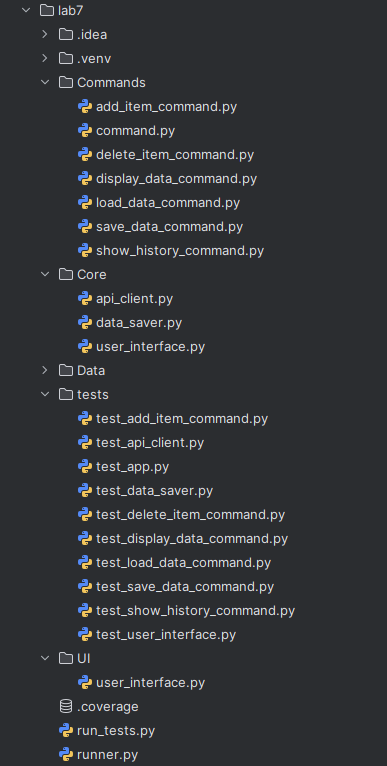
**Завдання 7: Обробка помилок.** Розробіть надійний механізм обробки помилок для керування помилками API, некоректним введенням користувача та іншими можливими проблемами. Надавайте інформативні повідомлення про помилки.

**Завдання 8: Ведення історії обчислень.** Включіть функцію, яка реєструє запити користувача, включаючи введені запити та відповідні результати. Дозвольте користувачам переглядати та рецензувати історію своїх запитів.

**Завдання 9: Юніт-тести.** Напишіть юніт-тести для перевірки функціональності вашого додатку. Тестуйте різні операції, граничні випадки та сценарії помилок.

**Виконання роботи**

**Структура файлів:**

****

**Вміст файлу user\_interface.py:**

import os  
import sys  
  
lab7\_root = os.path.abspath(os.path.join(os.path.dirname(\_\_file\_\_), ".."))  
sys.path.append(lab7\_root)  
  
from Commands.load\_data\_command import LoadDataCommand  
from Commands.display\_data\_command import DisplayDataCommand  
from Commands.add\_item\_command import AddItemCommand  
from Commands.delete\_item\_command import DeleteItemCommand  
from Commands.save\_data\_command import SaveDataCommand  
from Commands.show\_history\_command import ShowHistoryCommand, History  
from Core.user\_interface import UserInterface  
from Core.data\_saver import DataSaver  
  
def main():  
 user\_interface = UserInterface()  
 history = History()  
 data\_saver = DataSaver()  
  
 posts = []  
 local\_posts = []  
 albums = []  
 local\_albums = []  
  
 while True:  
 print("\n=== Меню ===")  
 print("1. Завантажити дані")  
 print("2. Відобразити пости")  
 print("3. Додати новий пост")  
 print("4. Видалити пост")  
 print("5. Відобразити альбоми")  
 print("6. Додати новий альбом")  
 print("7. Видалити альбом")  
 print("8. Зберегти пости у файл")  
 print("9. Зберегти альбоми у файл")  
 print("10. Показати історію")  
 print("0. Вихід")  
  
 choice = input("Оберіть опцію: ").strip()  
  
 match choice:  
 case "1":  
 LoadDataCommand(user\_interface, history, posts, albums).execute()  
 case "2":  
 DisplayDataCommand(user\_interface, posts + local\_posts, "posts").execute()  
 case "3":  
 AddItemCommand(user\_interface, history, posts, local\_posts, "posts").execute()  
 case "4":  
 DeleteItemCommand(user\_interface, history, local\_posts, "posts").execute()  
 case "5":  
 DisplayDataCommand(user\_interface, albums + local\_albums, "albums").execute()  
 case "6":  
 AddItemCommand(user\_interface, history, albums, local\_albums, "albums").execute()  
 case "7":  
 DeleteItemCommand(user\_interface, history, local\_albums, "albums").execute()  
 case "8":  
 SaveDataCommand(user\_interface, data\_saver, posts + local\_posts, "posts").execute()  
 case "9":  
 SaveDataCommand(user\_interface, data\_saver, albums + local\_albums, "albums").execute()  
 case "10":  
 ShowHistoryCommand(user\_interface, history).execute()  
 case "0":  
 break  
 case \_:  
 print("Неправильний вибір. Спробуйте ще раз.")

Результатом виконання лабораторної роботи є консольний об’єктно - орієнтований додаток з використанням API та патерну Command.

На рисунку 1 зображено інтерфейс користувача.

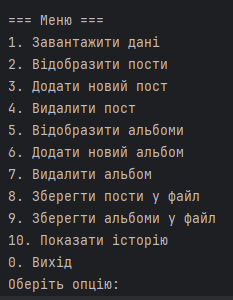


Рисунок 1 – Інтерфейс користувача

На рисунку 2 зображено доданий пост до списку.

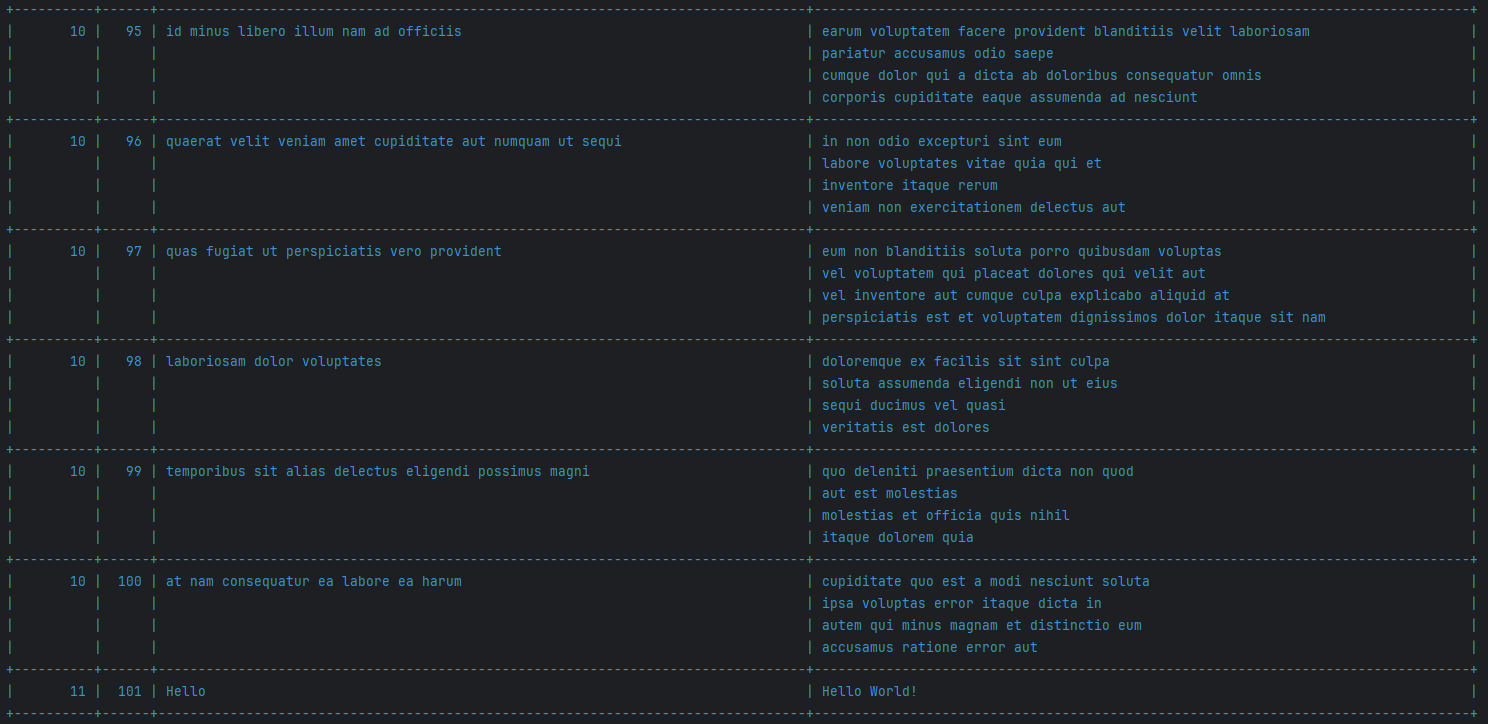
****

Рисунок 2 – Виведення постів

На рисунку 3 зображено результат тестування додатку

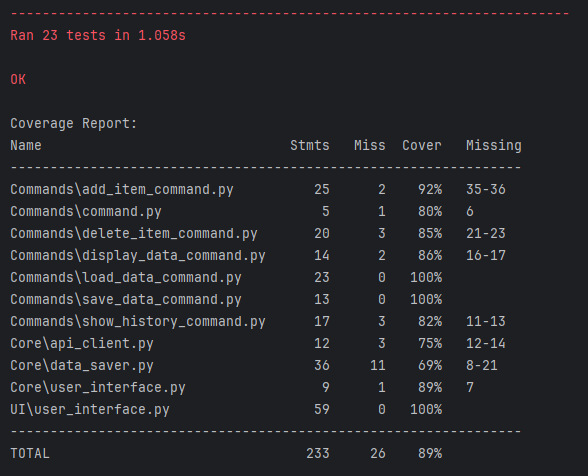
****

Рисунок 3 **–** Результат тестування

**Висновок:** У ході виконання ЛР я створив проект, який надав цінний досвід роботи з API, дизайну користувацького інтерфейсу, валідації введення, обробки помилок та тестування. Цей проект виконує різні дії з використанням API, а також тестує додаток та виводить Coverage Report.