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

Владислав РИБАК

Львів – 2024

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

План роботи

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 реалізації імплементації цієї лабораторної 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.

Текст програмної реалізації:

api_repository.py:

```
import requests

import json

import csv


class APIRepository:

    def __init__(self, base_url):

        self.base_url = base_url


    def get_data(self, endpoint):

        try:

            response = requests.get(f"{self.base_url}/{endpoint}")

            response.raise_for_status()

            return response.json()

        except requests.exceptions.RequestException as e:

            raise RuntimeError(f"Помилка при отриманні даних з API: {e}")


    def save_as_json(self, data, filename):

        with open(filename, 'w') as f:

            json.dump(data, f, indent=4)
```

```
    def save_as_csv(self, data, filename):
        keys = data[0].keys()
        with open(filename, 'w', newline='') as f:
            dict_writer = csv.DictWriter(f, fieldnames=keys)
            dict_writer.writeheader()
            dict_writer.writerows(data)

    def save_as_txt(self, data, filename):
        with open(filename, 'w') as f:
            for item in data:
                f.write(f"{item}\n")
```

main.py:

```
from dal.api_repository import APIRepository
from bll.data_service import DataService
from lab_7.ui.console_ui import ConsoleUI

class Command:
    def execute(self):
        base_url = "https://jsonplaceholder.typicode.com"
        repository = APIRepository(base_url)
        data_service = DataService(repository)
        console = ConsoleUI(data_service)
        console.run()
```

Результати тестування:

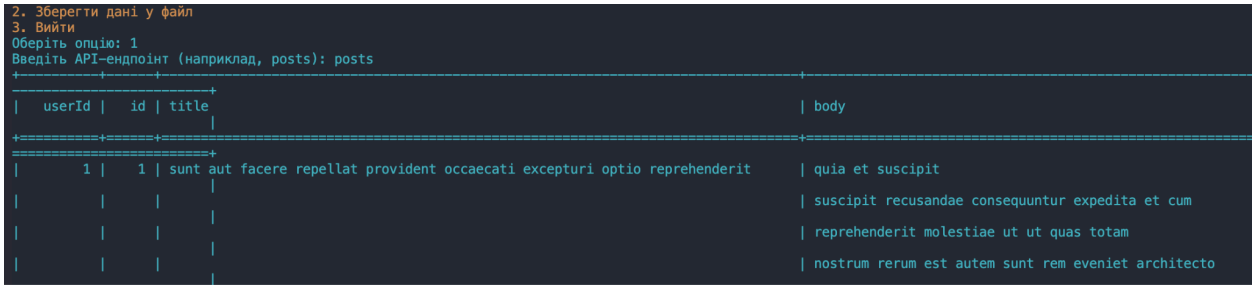

Рис. 1. Результат отримання даних з ендпоінта posts

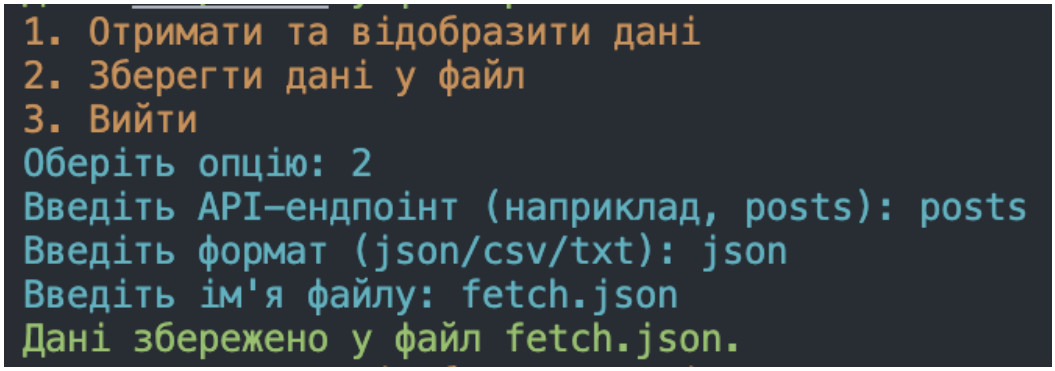

Рис. 2. Збереження даних з ендпоінту posts у json файл

Рис. 3. Вміст даних які було збережні з ендпоінту posts

Висновки: на цій лабораторній роботі я створив консольний об’єктно - орієнтований додаток з використанням API та патернів проектування.