Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Лабораторна роботи №7

з дисципліни «Спеціалізовані мови програмування»

на тему

«Робота з API та веб-сервісами»

Виконав:

Ложовський В. А.

Перевірив:

Щербак С. С.

Львів 2024

**Мета:** 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 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.

**Реалізація:**

**Файл save\_data.py:**  
import json

import csv

def save\_to\_json(data, filename="lab7/output.json"):

    with open(filename, "w") as f:

        json.dump(data, f, indent=4)

def save\_to\_csv(data, filename="lab7/output.csv"):

    with open(filename, "w") as f:

        writer = csv.DictWriter(f, fieldnames=data[0].keys())

        writer.writeheader()

        writer.writerows(data)

**Папка utils, файл display.py:**

from prettytable import PrettyTable

from colorama import Fore, Style

def display\_posts(posts):

    table = PrettyTable(["ID", "Title", "Body"])

    for post in posts:

        table.add\_row([post['id'], post['title'], post['body']])

    print(Fore.CYAN + table.get\_string() + Style.RESET\_ALL)

**Папка data\_storage, файл data\_storage.py:**import json

class DataStorage:

def save(self, data):

try:

with open('data.json', 'w') as json\_file:

json.dump(data, json\_file, indent=4) # Записуємо дані у JSON форматі

except Exception as e:

print(f"Помилка при збереженні даних: {e}")

**Файл tests.py:**

import unittest

from lab7.api.client import APIClient # type: ignore

class TestAPIClient(unittest.TestCase):

    def test\_fetch\_posts(self):

        client = APIClient()

        posts = client.fetch\_posts()

        self.assertIsInstance(posts, list)

        self.assertGreater(len(posts), 0)

**Файл main.py:**

from lab7.api.unit\_of\_work import UnitOfWork

from lab7.utils.user\_input import get\_user\_command

from lab7.utils.display import display\_posts

from lab7.utils.history\_logger import log\_user\_request

from lab7.save\_data import save\_to\_json, save\_to\_csv

def main():

    uow = UnitOfWork()

    command = get\_user\_command()

    if command == "show posts":

        posts = uow.fetch\_and\_commit()

        display\_posts(posts)

        log\_user\_request(command, posts)

        save\_to\_json(posts)

        save\_to\_csv(posts)

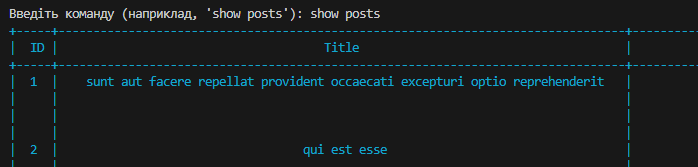
    else:

        print("Невідома команда")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**Результат виконання:**

****

**Висновок:** на цій лабораторній роботі я створив API-проект використовуючи сторонній ресурс з даними