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-32:

Фігура Т.Я

Прийняв:

Щербак С.С

Львів - 2024

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

**Хід виконання 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](https://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

class ILogger(ABC):

@abstractmethod

def log\_query(self, query, result):

pass

# Concrete implementations

class FileLogger(ILogger):

def log\_query(self, query, result):

with open("log.txt", "a") as file:

file.write(f"Query: {query}, Result: {result}\n")

class ConsoleLogger(ILogger):

def log\_query(self, query, result):

print(f"Query: {query}, Result: {result}")

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.

API\_URL = "https://api.openweathermap.org/data/2.5/weather"

API\_KEY = "bd89d8d6bee62af403385dff3b2b490a"

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.

city = input("Enter the city: ")

if not city:

print("Please enter a valid city name.")

continue

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.

if not city:

print("Please enter a valid city name.")

continue

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

weather = get\_weather(city)

if weather:

print(f"Weather in {weather['city']}: {weather['temperature']}°C, {weather['description']}")

logger.log\_query(city, weather)

else:

print(f"Could not fetch weather for {city}.")

Завдання 6: Збереження даних

Реалізуйте можливості збереження даних у чіткому та читабельному форматі JSON, CSV та TXT

def show\_history(self):

if not self.query\_history:

print("No queries have been logged yet.")

else:

print("\nQuery History:")

for i, entry in enumerate(self.query\_history, 1):

print(f"{i}. Query: {entry['query']}, Result: {entry['result']}")

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.

if not city:

print("Please enter a valid city name.")

continue

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.

logger.show\_history()

Завдання 9: Юніт-тести

Напишіть юніт-тести для перевірки функціональності вашого додатку. Тестуйте різні операції, граничні випадки та сценарії помилок.

def test\_log\_query(self):

self.logger.log\_query("New York", {"temperature": 20})

self.assertEqual(len(self.logger.query\_history), 1)

self.assertEqual(self.logger.query\_history[0]['query'], "New York")

self.assertEqual(self.logger.query\_history[0]['result']['temperature'], 20)

**Посилання на GitHub:** [***https://github.com/FihuraTaras/SPL\_Python\_Project.git***](https://github.com/FihuraTaras/SPL_Python_Project.git)

**Висновок:** Виконавши ці завдання, ви створите проект, який надасть вам цінний досвід роботи з API, дизайну користувацького інтерфейсу, валідації введення, обробки помилок та тестування.