

Міністерство освіти і науки України  
Національний університет «Львівська політехніка»

Звіт про виконання лабораторної роботи №7  
з дисципліни «Спеціалізовані мови програмування»  
на тему «Робота з API та веб-сервісами»

Виконав:  
студент групи РІ-32  
Гусак Віктор  
Прийняв:  
Щербак С.С.

**Мета 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](https://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 користувацький 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 виділятися 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.

### Хід роботи

#### app.py:

```
import sys
import os

sys.path.append(os.path.dirname(os.path.dirname(os.path.abspath(
__file__))))

from service.client import ApiClient
from repository.repository import Repository
from ui.data_visualizer import DataVisualizer
from input.input_handler import UserInputHandler

class ApiApp:
    def __init__(self):
        self.api_client =
ApiClient("https://jsonplaceholder.typicode.com")
        self.repository = Repository(self.api_client)

    def run(self):
        while True:
            choice = UserInputHandler.get_user_choice()
            if choice == '1':
                posts = self.repository.get_posts()
                if posts:
                    DataVisualizer.display_data(posts, "Список
постів")
                    DataVisualizer.save_to_json(posts,
'posts.json')
                    DataVisualizer.save_to_csv(posts,
'posts.csv')
            elif choice == '2':
                comments = self.repository.get_comments()
                if comments:
                    DataVisualizer.display_data(comments,
"Список коментарів")
```

```

        DataVisualizer.save_to_json(comments,
'comments.json')
        DataVisualizer.save_to_csv(comments,
'comments.csv')
        elif choice == '3':
            print("Вихід...")
            break
        else:
            print("Невірний вибір, спробуйте ще раз.")

if __name__ == "__main__":
    app = App()
    app.run()

```

### **input\_handler.py:**

```

import re

class UserInputHandler:
    @staticmethod
    def get_user_choice():
        print("1. Показати пости")
        print("2. Показати коментарі")
        print("3. Вийти")
        choice = input("Введіть номер вибору: ")
        return choice

    @staticmethod
    def validate_input(input_value, pattern):
        return bool(re.match(pattern, input_value))

```

### **repository.py:**

```

class Repository:
    def __init__(self, api_client):
        self.api_client = api_client

    def get_posts(self):
        return self.api_client.get_data("/posts")

    def get_comments(self):
        return self.api_client.get_data("/comments")

```

### **service.py:**

```
import requests

class ApiClient:
    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:
            print(f"Помилка API: {e}")
            return None
```

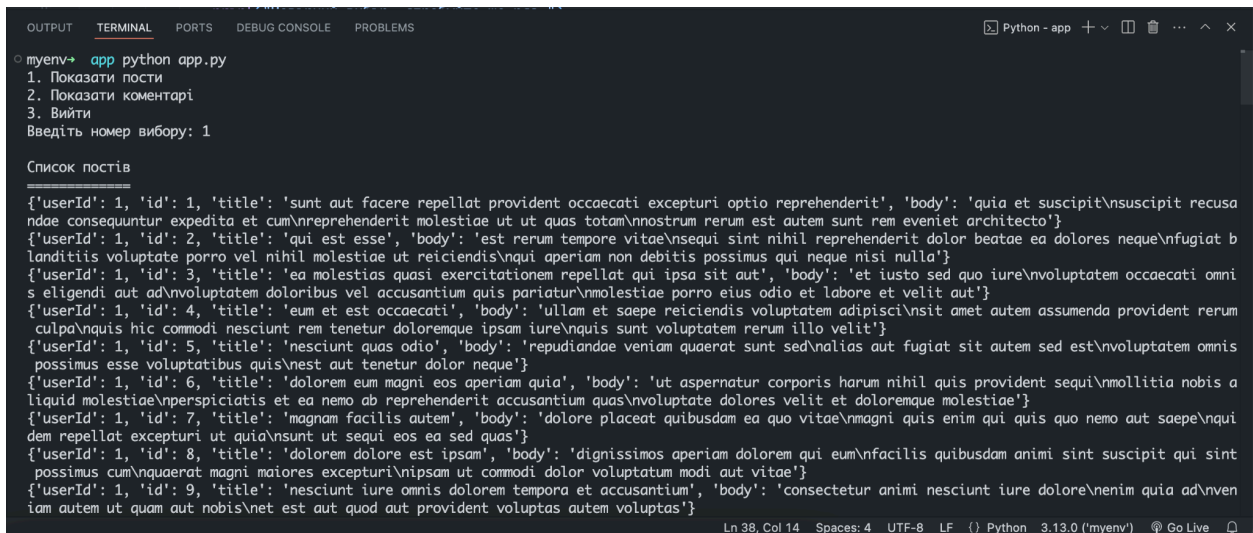
### **ui.py:**

```
import json
import csv

class DataVisualizer:
    @staticmethod
    def display_data(data, title):
        print(f"\n{title}")
        print("=" * len(title))
        for item in data:
            print(item)

    @staticmethod
    def save_to_json(data, filename):
        with open(filename, 'w') as json_file:
            json.dump(data, json_file, indent=4)
        print(f"Дані збережено в {filename}")

    @staticmethod
    def save_to_csv(data, filename):
        with open(filename, 'w', newline='') as csv_file:
            writer = csv.writer(csv_file)
            writer.writerow(data[0].keys())
            for item in data:
                writer.writerow(item.values())
        print(f"Дані збережено в {filename}")
```



```
myenv➤ app python app.py
1. Показати пости
2. Показати коментарі
3. Вийти
Введіть номер вибору: 1

Список постів

{'userId': 1, 'id': 1, 'title': 'sunt aut facere repellat provident occaecati excepturi optio reprehenderit', 'body': 'quia et suscipit\nsuscipit recusa
ndae consequuntur expedita et cum\nreprehenderit molestiae ut ut quas totam\nnostrum rerum est autem sunt rem eveniet architecto'}
{'userId': 1, 'id': 2, 'title': 'qui est esse', 'body': 'est rerum tempore vitae\nsequi sint nihil reprehenderit dolor beatae ea dolores neque\nfugiat b
landitiis voluptate porro vel nihil molestiae ut reiciendis\nqui aperiam non debitis possimus qui neque nisi nulla'}
{'userId': 1, 'id': 3, 'title': 'ea molestias quasi exercitationem repellat qui ipsa sit aut', 'body': 'et iusto sed quo iure\nvoluptatem occaecati omni
s eligendi aut ad\nvoluptatem doloribus vel accusantium quis pariatur\nmolestiae porro eius odio et labore et velit aut'}
{'userId': 1, 'id': 4, 'title': 'eum et est occaecati', 'body': 'ullam et saepe reiciendis voluptatem adipisci\nsit amet autem assumenda provident rerum
culpa\nquis hic commodi nesciunt rem tenetur doloremque ipsam iure\nquis sunt voluptatem rerum illo velit'}
{'userId': 1, 'id': 5, 'title': 'nesciunt quas odio', 'body': 'repudiandae veniam quaerat sunt sed\nalias aut fugiat sit autem sed est\nvoluptatem omnis
possimus esse voluptatibus quis\nest aut tenetur dolor neque'}
{'userId': 1, 'id': 6, 'title': 'dolorem eum magni eos aperiam quia', 'body': 'ut aspernatur corporis harum nihil quis provident sequi\nmollitia nobis a
liquid molestiae\nperspiciatis et ea nemo ab reprehenderit accusantium quas\nvoluptate dolores velit et doloremque molestiae'}
{'userId': 1, 'id': 7, 'title': 'magnam facilis autem', 'body': 'dolore placeat quibusdam ea quo vitae\nmagni quis enim qui quis quo nemo aut saepe\nqui
dem repellat excepturi ut quia\nsunt ut sequi eos ea sed quas'}
{'userId': 1, 'id': 8, 'title': 'dolorem dolore est ipsam', 'body': 'dignissimos aperiam dolorem qui eum\nfacilis quibusdam animi sint suscipit qui sint
possimus cum\nquaerat magni maiores excepturi\nipsam ut commodi dolor voluptatum modi aut vitae'}
{'userId': 1, 'id': 9, 'title': 'nesciunt iure omnis dolorem tempora et accusantium', 'body': 'consectetur animi nesciunt iure dolore\nenim quia ad\nven
iam autem ut quam aut nobis\net est aut quod aut provident voluptas autem voluptas'}
```

Рис. 1. Приклад роботи програми

**Висновки:** Виконавши ці завдання, було створено проект, що надав цінний досвід роботи з API, дизайну користувацького інтерфейсу, валідації введення, обробки помилок та тестування.