ЗВІТ

про виконання лабораторної роботи № 7

«Робота з API та веб-сервісами»

з дисципліни

«Спеціалізовані мови програмування»

студента групи РІ-31

Висоцького Володимира Володимировича

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

**Умова 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.

**Текст програми:**

from DAL.DataSaver import DataSaver  
from Lab\_7.APIClient import APIClient  
from Lab\_7.DataDisplay import DataDisplay  
from Lab\_7.ErrorHandler import ErrorHandler  
from Lab\_7.HistoryLogger import HistoryLogger  
from UI.MenuItem import MenuItem  
from UI.MenuBuilder import MenuBuilder  
  
  
def show\_all\_posts(client, display, logger):  
 """Fetch and display all posts."""  
 try:  
 posts = client.get\_data("posts")  
 display.show\_table(posts)  
 logger.log("показати всі пости", posts)  
 except Exception as error:  
 ErrorHandler.handle\_error(error)  
  
  
def show\_post\_by\_id(client, display, logger):  
 """Fetch and display a post by its ID."""  
 try:  
 post\_id = input("Введіть ID поста: ")  
 post = client.get\_data\_by\_id("posts", post\_id)  
 display.show\_table([post])  
 logger.log(f"показати пост з ID {post\_id}", post)  
 except Exception as error:  
 ErrorHandler.handle\_error(error)  
  
  
def save\_data(client, saver):  
 """Save data to a specified file format."""  
 format\_choice = input("Введіть формат збереження (json/csv/txt): ").strip().lower()  
 filename = input("Введіть ім'я файлу: ").strip()  
 if format\_choice in ['json', 'csv', 'txt']:  
 try:  
 saver\_method = {  
 'json': saver.save\_as\_json,  
 'csv': saver.save\_as\_csv,  
 'txt': saver.save\_as\_txt  
 }[format\_choice]  
 posts = client.get\_data("posts") # Fetch posts before saving  
 saver\_method(posts, filename)  
 print(f"Дані збережено у {filename}.{format\_choice}")  
 except Exception as error:  
 ErrorHandler.handle\_error(error)  
 else:  
 print("Невідомий формат.")  
  
  
def show\_all\_users(client, display, logger):  
 """Fetch and display all users."""  
 try:  
 users = client.get\_all\_users()  
 display.show\_users(users)  
 logger.log("показати всіх користувачів", users)  
 except Exception as error:  
 ErrorHandler.handle\_error(error)  
  
  
def show\_user\_by\_id(client, display, logger):  
 """Fetch and display a user by its ID."""  
 try:  
 user\_id = input("Введіть ID користувача: ")  
 user = client.get\_user\_by\_id(user\_id)  
 display.show\_users([user])  
 logger.log(f"показати користувача з ID {user\_id}", user)  
 except Exception as error:  
 ErrorHandler.handle\_error(error)  
  
  
def show\_all\_comments(client, display, logger):  
 """Fetch and display all comments."""  
 try:  
 comments = client.get\_all\_comments()  
 display.show\_comments(comments)  
 logger.log("показати всі коментарі", comments)  
 except Exception as error:  
 ErrorHandler.handle\_error(error)  
  
  
def show\_comment\_by\_id(client, display, logger):  
 """Fetch and display a comment by its ID."""  
 try:  
 comment\_id = input("Введіть ID коментаря: ")  
 comment = client.get\_comment\_by\_id(comment\_id)  
 display.show\_comments([comment])  
 logger.log(f"показати коментар з ID {comment\_id}", comment)  
 except Exception as error:  
 ErrorHandler.handle\_error(error)  
  
  
def show\_history(logger):  
 """Show the history of logs."""  
 logger.show\_history()  
  
  
def exit\_program():  
 """Exit the program."""  
 print("Вихід...")  
 return  
  
  
def run():  
 """Main function to run the program."""  
 client = APIClient()  
 display = DataDisplay()  
 saver = DataSaver()  
 logger = HistoryLogger()  
  
 menu\_items = [  
 MenuItem("1", "Показати всі пости", lambda: show\_all\_posts(client, display, logger)),  
 MenuItem("2", "Показати пост за ID", lambda: show\_post\_by\_id(client, display, logger)),  
 MenuItem("3", "Зберегти дані", lambda: save\_data(client, saver)),  
 MenuItem("4", "Показати всіх користувачів", lambda: show\_all\_users(client, display, logger)),  
 MenuItem("5", "Показати користувача за ID", lambda: show\_user\_by\_id(client, display, logger)),  
 MenuItem("6", "Показати всі коментарі", lambda: show\_all\_comments(client, display, logger)),  
 MenuItem("7", "Показати коментар за ID", lambda: show\_comment\_by\_id(client, display, logger)),  
 MenuItem("8", "Показати історію", lambda: show\_history(logger)),  
 MenuItem("0", "Вийти", exit\_program),  
 ]  
  
 menu = MenuBuilder(menu\_items)  
 while True:  
 menu.initialize()

**Висновки:** Виконуючи ці завдання я створив проект, який надасть мені цінний досвід роботи з API, дизайну користувацького інтерфейсу, валідації введення, обробки помилок та тестування.