МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Кафедра інформаційних систем та мереж

Лабораторна робота №9

з дисципліни

«Спеціалізовані мови програмування»

на тему

«Створення та рефакторінг програмно-інформаційного продукту засобами Python»

Виконав:

ст. гр. РІ-32

Юліан МЕЛЬНИЧУК

Прийняв:

доц. каф. ІСМ

Сергій ЩЕРБАК

Львів - 2024

**Мета**

Розробка програмно-інформаційного продукту засобами Python

**Хід виконання роботи**

**Завдання 1. Створення скрипту запуску лабораторних робіт через FACADE**

Було створено скрипт запуску лабораторних робіт 1-8, що включає єдине меню для управління додатками. Це меню використовує патерн FACADE, що спрощує взаємодію користувача з різними компонентами програми.

**Завдання 2. Рефакторинг додатків для підтримки запуску через Runner**

Були проведені зміни в додатках, створених в лабораторних роботах 1-8, щоб вони могли бути запущені через єдиний скрипт Runner. Це забезпечило інтеграцію додатків без зміни їх основного функціоналу.

**Завдання 3. Рефакторинг додатків з використанням багаторівневої архітектури**

Додатки були рефакторені з використанням багаторівневої архітектури, що дозволило покращити структуру програм, поділивши їх на рівні відповідальності за різні частини додатка, зокрема для обробки даних, бізнес-логіки та управління інтерфейсом.

**Завдання 4. Створення бібліотеки загальних класі**

Була створена бібліотека класів, які можуть бути повторновикористані у всіх лабораторних роботах. Бібліотека включає мінімум п'ять класів для спільної функціональності, як-от робота з файлами, обробка помилок та налаштування.

**Завдання 5. Додавання логування до функцій бібліотеки**

До класів бібліотеки було додано логування важливих функцій та методів, що дозволяє відслідковувати роботу додатків і діагностувати можливі помилки.

**Завдання 6. Додавання коментарів і документації за допомогою pydoc**

Код був прокоментований для підвищення зрозумілості, а також була згенерована документація за допомогою інструменту pydoc. Документація доступна як через консоль, так і через веб-браузер та в HTML файлах.

**Завдання 7. Розміщення документації та коду в GIT репозиторії**

Документація та весь код програмного продукту були розміщені в Git репозиторії. Репозиторій містить вихідний код, згенеровану документацію та всі необхідні інструкції для запуску і використання продукту.

**Завдання 8. Статичний аналіз коду за допомогою PYLINT**

Було проведено статичний аналіз коду за допомогою інструменту Pylint. Усі виявлені помилки були виправлені.

Зображення, що містить текст, знімок екрана, Шрифт, число

Автоматично згенерований опис

Зображення, що містить текст, знімок екрана, Шрифт, ряд

Автоматично згенерований опис

Рис. 1-2. Статистичний аналіз коду

**Завдання 9. Підготовка звіту до лабораторної роботи**

Звіт до лабораторної роботи був підготовлений, в якому детально описано виконання всіх завдань, рефакторинг додатків, інтеграція з Runner, а також результати статичного аналізу коду та виправлені помилки.

**Код програмного продукту**

class Runner(RunnerInterface):

"""

Class Runner inherits from RunnerInterface to implement

a specific way to execute tasks by calling the main function.

Methods:

@staticmethod

def run():

Calls the main function to execute the primary logic of this runner.

"""

@staticmethod

def run():

main()

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

Runner.run()

import os

import subprocess

from config.settings\_paths import current\_dir

def generate\_docs(directory):

"""

:param directory: The base directory from which to gather .py files and generate documentation.

:return: None. Generates documentation files.

"""

# Get the absolute path of the directory containing your codebase

directory = os.path.abspath(os.path.join(directory, '..'))

for root, dirs, files in os.walk(directory):

# Ignore directories starting with a dot (e.g., .git, .vscode)

dirs[:] = [d for d in dirs if not d.startswith('.')]

for file in files:

if file.endswith('.py'):

# Get the relative path from the base directory

relative\_path = os.path.relpath(os.path.join(root, file), directory)

# Replace OS-specific path separators with dots and remove the .py extension

module\_name = os.path.splitext(relative\_path)[0].replace(os.sep, '.')

# Exclude \_\_init\_\_ files if you don't want docs for them

if module\_name.endswith(".\_\_init\_\_"):

continue

print(f"Generating docs for: {module\_name}")

# Call pydoc to generate documentation

subprocess.run(['python', '-m', 'pydoc', '-w', module\_name])

if \_\_name\_\_ == '\_\_main\_\_':

generate\_docs(current\_dir)

**Висновки**

Завершення цих завдань дозволило створити багатофункціональний додаток для візуалізації CSV-наборів даних за допомогою Matplotlib. Проект надав цінний досвід в області візуалізації даних та обробки різноманітних наборів інформації.