

Практична робота №7. Проєкт, модулі, імпорт бібліотек, рір. Робота з файлами у Python.

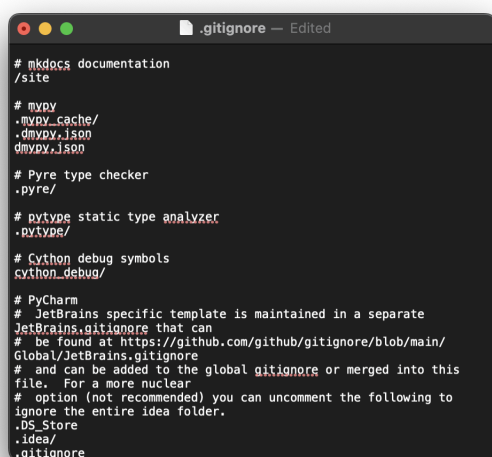
Андрій Суліменко, група 4

GitHub: [link](#)

0. Підготувати звіт, де в репозиторії та скріншотах відображається кожен етап, який пізніше Ви зможете прикріпити у мудл. Цей файл використовувати в якості шаблону для звіту.

1. Створення нового проєкту.

- a. Створити новий проєкт локально у PyCharm або VSCode (можна частково використовувати інструкції з ПЗ 1). При створенні проєкта, назвіть його «project_template» та оберіть створення віртуального середовища `venv`, `main.py` створимо пізніше. Назву папки віртуального середовища запам'ятайте, ми використаємо її пізніше.
- b. Підготуйте файл `.gitignore`, щоб папки типу `venv` або `.idea` і.т.п. не потрапили до репозиторію, який призначений суто для коду проєкту.



```
# mkdocs documentation
/site

# mypy
.mypy_cache/
.dmypy.json
dmypy.json

# Pyre type checker
.pyre/

# pytype static type analyzer
.pytype/

# Cython debug symbols
cython_debug/

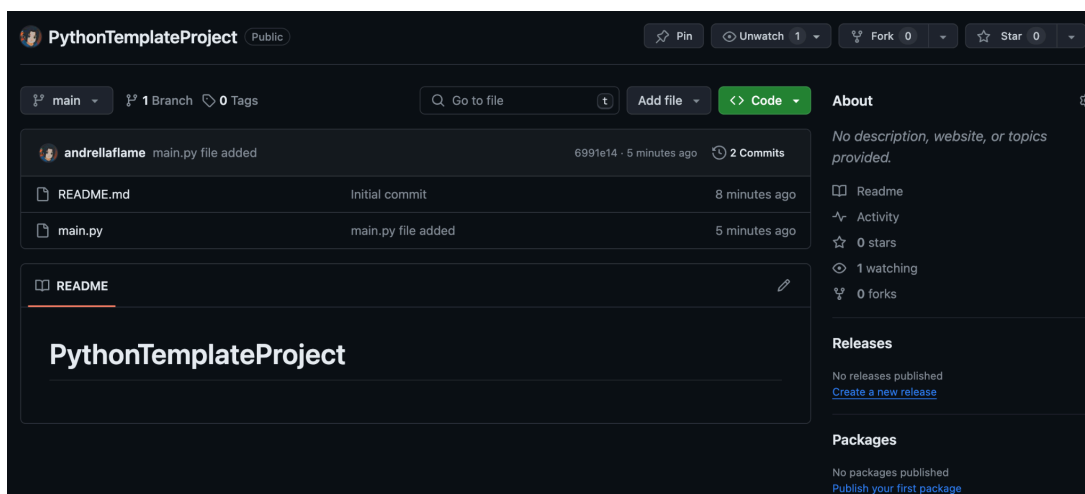
# PyCharm
# JetBrains specific template is maintained in a separate
# JetBrains.gitignore that can
# be found at https://github.com/github/gitignore/blob/main/
# Global/JetBrains.gitignore
# and can be added to the global gitignore or merged into this
# file. For a more nuclear
# option (not recommended) you can uncomment the following to
# ignore the entire idea folder.
.DS_Store
.idea/
.gitignore
```

- с. Створіть файл `main.py` у директорії проєкту, який матиме наступний вигляд:



```
def main():  
    pass  
  
if __name__ == '__main__':  
    main()
```

- d. Створіть також новий репозиторій на GitHub (теж підглянути, як це робиться, можете у ПЗ 1).
- е. Об'єднайте локальний та віддалений репозиторії. Залийте зміни на віддалений репозиторій (тут теж можете згадати ПЗ 1). Посилання на нього додайте на початок звіту.

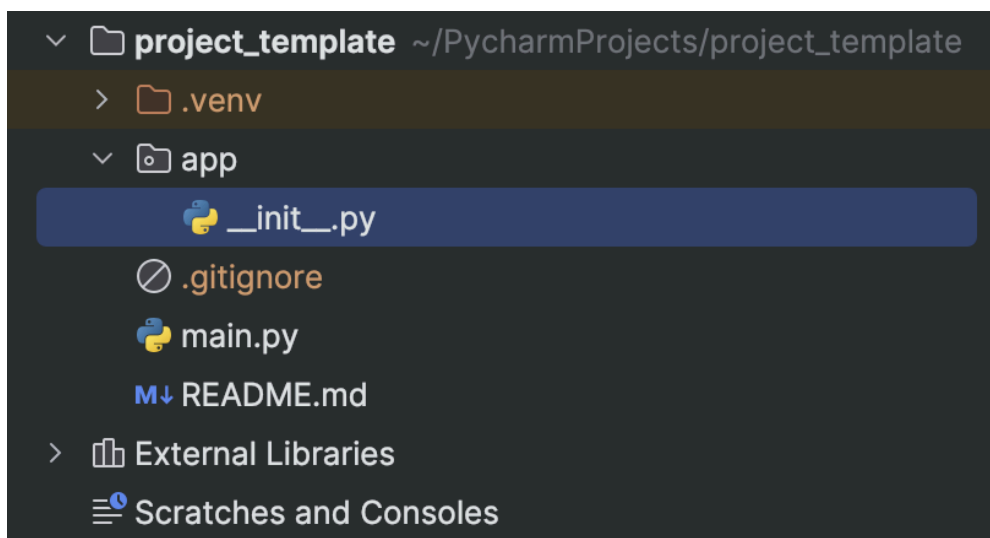


2. Структура проєкту.

- а. Створити в директорії проєкту нову папку (Python Package – директорія, яка має одразу пустий файл `__init__.py`) і

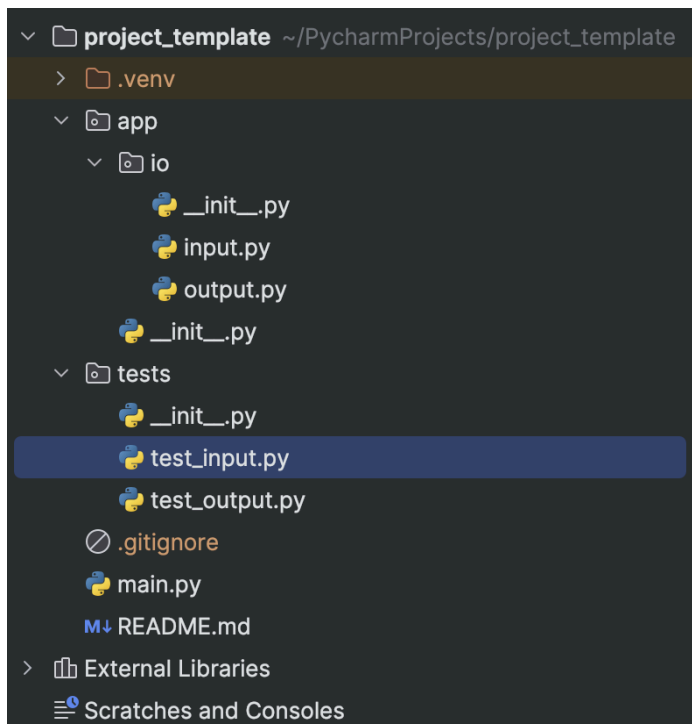
назвати її «app».

Це є місце, де структуровано зберігаються модулі проєкту з кодом, який безпосередньо бере участь у запуску та виконанні задач застосунку. Тобто це код, який запускається користувачем (у його ролі може бути як людина, що на кнопку на фронтенді натиснула, так і інша система, яка, наприклад, використовує результати поточної).



- b. У середині цієї директорії app створити Python Package «io» (скорочено input-output).
- c. У цій директорії io створити два файли: input.py та output.py.
- d. Створити ще один Python Package і назвати його «tests». *Це є директорія, що містить unit тести, та буде дзеркальною для app (тобто, наприклад, файл test_input.py у tests відповідатиме файлу input.py у app, і те саме для піддиректорії io у app та test_io у tests і т.д.).*
- e. Залити зміни на віддалений репозиторій з відповідним повідомленням у коміті.

 andrellafame	io and tests added	e6da7e1 · now	🕒 3 Commits
📁 app	io and tests added		now
📁 tests	io and tests added		now
📄 README.md	Initial commit		21 minutes ago
📄 main.py	main.py file added		19 minutes ago



3. Робота з модулями.

1. Якщо ви працюєте з `pipenv`, перейдіть до кроку 3.

Переконайтесь, що ваше віртуальне середовище активовано. Якщо ні, переходьте до кроку 2. Щоби перевірити, що середовище активовано, використайте відповідну команду, яка покаже, який інтерпретатор використовується в даний момент у проєкті.

Для Windows:

```
where python
```

Для Unix/MacOS:

```
which python
```

```
Terminal  Local  ×  +  ∨  
(.venv) andreysulimenko@mbp--andrii project_template % which python  
/Users/andreysulimenko/PycharmProjects/project_template/.venv/bin/python
```

2. Активуйте його самостійно за допомогою наступних команд у терміналі у директорії проєкту можна переключитись за допомогою команди
`cd path/to/proj_dir)`

Для Windows:

```
nazva_venv\Scripts\activate
```

Для Unix/MacOS:

```
source nazva_venv/bin/activate
```

де `nazva_venv` – це назва папки з віртуальним середовищем при створенні у вашому проєкті, скоріше за все, вона має назву `venv`.

```
(.venv) andreysulimenko@mbp--andrii project_template % source .venv/bin/activate
```

3. Підготуйте `pip`.

Для Windows:

```
py -m pip install --upgrade pip  
py -m pip --version
```

Для Unix/MacOS:

```
python3 -m pip install --upgrade pip  
python3 -m pip --version
```

Після цього маєте побачити свіжу версію менеджера пакетів pip.

```
Terminal Local x + v
(.venv) andreysulimenko@mbp--andrii project_template % python3 -m pip install --upgrade pip
Requirement already satisfied: pip in ./venv/lib/python3.12/site-packages (23.2.1)
Collecting pip
  Obtaining dependency information for pip from https://files.pythonhosted.org/packages/8a/6a/19e9fe04fca059ccf770861c7d5721ab4c2aebc539889e97c7977528a53b/pip-24.0-py3-none-any.whl.metadata
  Downloading pip-24.0-py3-none-any.whl.metadata (3.6 kB)
  Downloading pip-24.0-py3-none-any.whl (2.1 MB)
    2.1/2.1 MB 3.6 MB/s eta 0:00:00
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 23.2.1
    Uninstalling pip-23.2.1:
      Successfully uninstalled pip-23.2.1
Successfully installed pip-24.0
(.venv) andreysulimenko@mbp--andrii project_template % python3 -m pip --version
pip 24.0 from /Users/andreysulimenko/PycharmProjects/project_template/.venv/lib/python3.12/site-packages/pip (python 3.12)
(.venv) andreysulimenko@mbp--andrii project_template %
```

4. Встановлюємо пакети через pip.

Якщо ви працюєте з `pipenv`, після прочитання цієї статті <https://realpython.com/pipenv-guide/> виконайте аналогічні для `pipenv` інструкції нижче (мається на увазі не виконання 1-в-1, а знаходження інструкцій, як зробити ту ж саму логіку, але через `pipenv`).

4.а. Встановлення останньої версії пакету.

Для цього рекомендую вам перейти на сайт <https://pypi.org> та в пошуку знайти пакет `numpy`.

Скопіюйте цю команду з верхньої частини сторінки та запустіть її в терміналі.

Після цього ви маєте бачити повідомлення про успішну інсталяцію пакету `numpy` та його `dependencies` (залежностей - пакетів).

```
(.venv) andreysulimenko@mbp--andrii project_template % pip install numpy
Collecting numpy
  Downloading numpy-1.26.4-cp312-cp312-macosx_11_0_arm64.whl.metadata (61 kB)
    61.1/61.1 kB 1.8 MB/s eta 0:00:00
  Downloading numpy-1.26.4-cp312-cp312-macosx_11_0_arm64.whl (13.7 MB)
    13.7/13.7 MB 3.4 MB/s eta 0:00:00
Installing collected packages: numpy
Successfully installed numpy-1.26.4
(.venv) andreysulimenko@mbp--andrii project_template %
```

4.b. Встановлення конкретної версії пакету (рекомендований спосіб для подальшого використання).

Тепер знайдіть у рурі бібліотеку pandas, в історії версій (релізів) знайдіть **передостанню** версію та введіть у терміналі команду, щоб встановити його з відповідною версією:

Для Windows:

```
python -m pip install "SomeProject==1.4"  
або  
py -m pip install "SomeProject==1.4"
```

Для Unix/macOS:

```
python3 -m pip install "SomeProject==1.4"
```

де SomeProject – назва бібліотеки для інсталювання, == це визначення для того, яка конкретна версія потрібна і 1.4 – це цифри, що відповідають номерам версії для встановлення. Наприклад,

```
python -m pip install "seaborn==0.13.1"
```

pandas 2.2.1

✓ Latest version

`pip install pandas`

Released: Feb 23, 2024

Powerful data structures for data analysis, time series, and statistics

Navigation

Project description

Release history

Download files

Project links

Homepage

Release history

[Release notifications](#) | [RSS feed](#)

THIS VERSION



2.2.1

Feb 23, 2024



2.2.0

Jan 19, 2024

```
(.env) andreysulimenko@mbp--andrii project_template % python3 -m pip install "pandas==2.2.0"
Collecting pandas==2.2.0
  Downloading pandas-2.2.0-cp312-cp312-macosx_11_0_arm64.whl.metadata (19 kB)
Requirement already satisfied: numpy<2, >=1.26.0 in ./venv/lib/python3.12/site-packages (from pandas==2.2.0) (1.26.4)
Collecting python-dateutil>=2.8.2 (from pandas==2.2.0)
  Downloading python_dateutil-2.9.0.post0-py2.py3-none-any.whl.metadata (8.4 kB)
Collecting pytz>=2020.1 (from pandas==2.2.0)
  Downloading pytz-2024.1-py2.py3-none-any.whl.metadata (22 kB)
Collecting tzdata>=2022.7 (from pandas==2.2.0)
  Downloading tzdata-2024.1-py2.py3-none-any.whl.metadata (1.4 kB)
Collecting six>=1.5 (from python-dateutil>=2.8.2->pandas==2.2.0)
  Downloading six-1.16.0-py2.py3-none-any.whl.metadata (1.8 kB)
Downloading pandas-2.2.0-cp312-cp312-macosx_11_0_arm64.whl (11.7 MB)
  11.7/11.7 MB 1.2 MB/s eta 0:00:00
Downloading python_dateutil-2.9.0.post0-py2.py3-none-any.whl (229 kB)
  229.9/229.9 kB 1.1 MB/s eta 0:00:00
Downloading pytz-2024.1-py2.py3-none-any.whl (505 kB)
  505.5/505.5 kB 1.1 MB/s eta 0:00:00
Downloading tzdata-2024.1-py2.py3-none-any.whl (345 kB)
  345.4/345.4 kB 1.1 MB/s eta 0:00:00
Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: pytz, tzdata, six, python-dateutil, pandas
Successfully installed pandas-2.2.0 python-dateutil-2.9.0.post0 pytz-2024.1 six-1.16.0 tzdata-2024.1
(.env) andreysulimenko@mbp--andrii project_template %
```

Більше про встановлення бібліотек можете прочитати тут:

<https://packaging.python.org/en/latest/guides/installing-using-pip-and-virtual-environments/>

5. Тепер пропоную вам встановити самостійно додатково

пакети matplotlib та pylint, black.

```
(.venv) andreysulimenko@mbp--andrii project_template % pip install matplotlib
Collecting matplotlib
  Downloading matplotlib-3.8.3-cp312-cp312-macosx_11_0_arm64.whl.metadata (5.8 kB)
Collecting contourpy>=1.0.1 (from matplotlib)
  Downloading contourpy-1.2.0-cp312-cp312-macosx_11_0_arm64.whl.metadata (5.8 kB)
Collecting cycler>=0.10 (from matplotlib)
  Downloading cycler-0.12.1-py3-none-any.whl.metadata (3.8 kB)
Collecting fonttools>=4.22.0 (from matplotlib)
  Downloading fonttools-4.49.0-cp312-cp312-macosx_10_9_universal2.whl.metadata (159 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 159.1/159.1 kB 944.6 kB/s eta 0:00:00
Collecting kiwisolver>=1.3.1 (from matplotlib)
  Downloading kiwisolver-1.4.5-cp312-cp312-macosx_11_0_arm64.whl.metadata (6.4 kB)
Requirement already satisfied: numpy<2,>=1.21 in ./venv/lib/python3.12/site-packages (from matplotlib) (1.26.4)
Collecting packaging>=20.0 (from matplotlib)
  Downloading packaging-23.2-py3-none-any.whl.metadata (3.2 kB)
Collecting pillow>=8 (from matplotlib)
  Downloading pillow-10.2.0-cp312-cp312-macosx_11_0_arm64.whl.metadata (9.7 kB)
Collecting pyparsing>=2.3.1 (from matplotlib)
  Downloading pyparsing-3.1.2-py3-none-any.whl.metadata (5.1 kB)
Requirement already satisfied: python-dateutil>=2.7 in ./venv/lib/python3.12/site-packages (from matplotlib) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in ./venv/lib/python3.12/site-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
Downloading matplotlib-3.8.3-cp312-cp312-macosx_11_0_arm64.whl (7.5 MB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.5/7.5 MB 1.1 MB/s eta 0:00:00
Downloading contourpy-1.2.0-cp312-cp312-macosx_11_0_arm64.whl (242 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 242.6/242.6 kB 1.5 MB/s eta 0:00:00
Downloading cycler-0.12.1-py3-none-any.whl (8.3 kB)
Downloading fonttools-4.49.0-cp312-cp312-macosx_10_9_universal2.whl (2.8 MB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.8/2.8 MB 1.4 MB/s eta 0:00:00
Downloading kiwisolver-1.4.5-cp312-cp312-macosx_11_0_arm64.whl (64 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 65.0/65.0 kB 1.4 MB/s eta 0:00:00
Downloading packaging-23.2-py3-none-any.whl (53 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 53.0/53.0 kB 1.1 MB/s eta 0:00:00
Downloading pillow-10.2.0-cp312-cp312-macosx_11_0_arm64.whl (3.3 MB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.3/3.3 MB 1.4 MB/s eta 0:00:00
```

```
(.venv) andreysulimenko@mbp--andrii project_template % pip install pylint
Collecting pylint
  Downloading pylint-3.1.0-py3-none-any.whl.metadata (12 kB)
Collecting platformdirs>=2.2.0 (from pylint)
  Downloading platformdirs-4.2.0-py3-none-any.whl.metadata (11 kB)
Collecting astroid<=3.2.0-dev0,>=3.1.0 (from pylint)
  Downloading astroid-3.1.0-py3-none-any.whl.metadata (4.5 kB)
Collecting isort!=5.13.0,<6,>=4.2.5 (from pylint)
  Downloading isort-5.13.2-py3-none-any.whl.metadata (12 kB)
Collecting mccabe<0.8,>=0.6 (from pylint)
  Downloading mccabe-0.7.0-py2.py3-none-any.whl.metadata (5.0 kB)
Collecting tomlkit>=0.10.1 (from pylint)
  Downloading tomlkit-0.12.4-py3-none-any.whl.metadata (2.7 kB)
Collecting dill>=0.3.6 (from pylint)
  Downloading dill-0.3.8-py3-none-any.whl.metadata (10 kB)
Downloading pylint-3.1.0-py3-none-any.whl (515 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 515.6/515.6 kB 2.2 MB/s eta 0:00:00
Downloading astroid-3.1.0-py3-none-any.whl (275 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 275.6/275.6 kB 3.1 MB/s eta 0:00:00
Downloading dill-0.3.8-py3-none-any.whl (116 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 116.3/116.3 kB 2.6 MB/s eta 0:00:00
Downloading isort-5.13.2-py3-none-any.whl (92 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 92.3/92.3 kB 2.6 MB/s eta 0:00:00
Downloading mccabe-0.7.0-py2.py3-none-any.whl (7.3 kB)
Downloading platformdirs-4.2.0-py3-none-any.whl (17 kB)
Downloading tomlkit-0.12.4-py3-none-any.whl (37 kB)
Installing collected packages: tomlkit, platformdirs, mccabe, isort, dill, astroid, pylint
Successfully installed astroid-3.1.0 dill-0.3.8 isort-5.13.2 mccabe-0.7.0 platformdirs-4.2.0 pylint-3.1.0 tomlkit-0.12.4
(.venv) andreysulimenko@mbp--andrii project_template %
```

```
(.venv) andreysulimenko@mbp--andrii project_template % pip install black
Collecting black
  Downloading black-24.2.0-cp312-cp312-macosx_11_0_arm64.whl.metadata (74 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 74.6/74.6 kB 2.2 MB/s eta 0:00:00
Collecting click>=8.0.0 (from black)
  Downloading click-8.1.7-py3-none-any.whl.metadata (3.0 kB)
Collecting mypy_extensions>=0.4.3 (from black)
  Downloading mypy_extensions-1.0.0-py3-none-any.whl.metadata (1.1 kB)
Requirement already satisfied: packaging>=22.0 in ./venv/lib/python3.12/site-packages (from black) (23.2)
Collecting pathspec>=0.9.0 (from black)
  Downloading pathspec-0.12.1-py3-none-any.whl.metadata (21 kB)
Requirement already satisfied: platformdirs>=2 in ./venv/lib/python3.12/site-packages (from black) (4.2.0)
Downloading black-24.2.0-cp312-cp312-macosx_11_0_arm64.whl (1.4 MB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.4/1.4 MB 3.7 MB/s eta 0:00:00
Downloading click-8.1.7-py3-none-any.whl (97 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 97.9/97.9 kB 3.2 MB/s eta 0:00:00
Downloading mypy_extensions-1.0.0-py3-none-any.whl (4.7 kB)
Downloading pathspec-0.12.1-py3-none-any.whl (31 kB)
Installing collected packages: pathspec, mypy_extensions, click, black
Successfully installed black-24.2.0 click-8.1.7 mypy_extensions-1.0.0 pathspec-0.12.1
(.venv) andreysulimenko@mbp--andrii project_template %
```

6. Після цього утворимо список з усіма пакетами та їхніми версіями для зручнішої роботи у команді. Зазвичай це робиться через файл requirements.txt або pipfile при роботі з ріреnv. Отже, якщо ви робите цю роботу з ріреnv, вам необхідно додати до репозиторію рірfile та рірfile.lock, а при використанні venv – requirements.txt.

Для venv:

Для Windows:

```
python -m pip freeze
або
py -m pip freeze
```

Для Unix/macOS:

```
python3 -m pip freeze
```

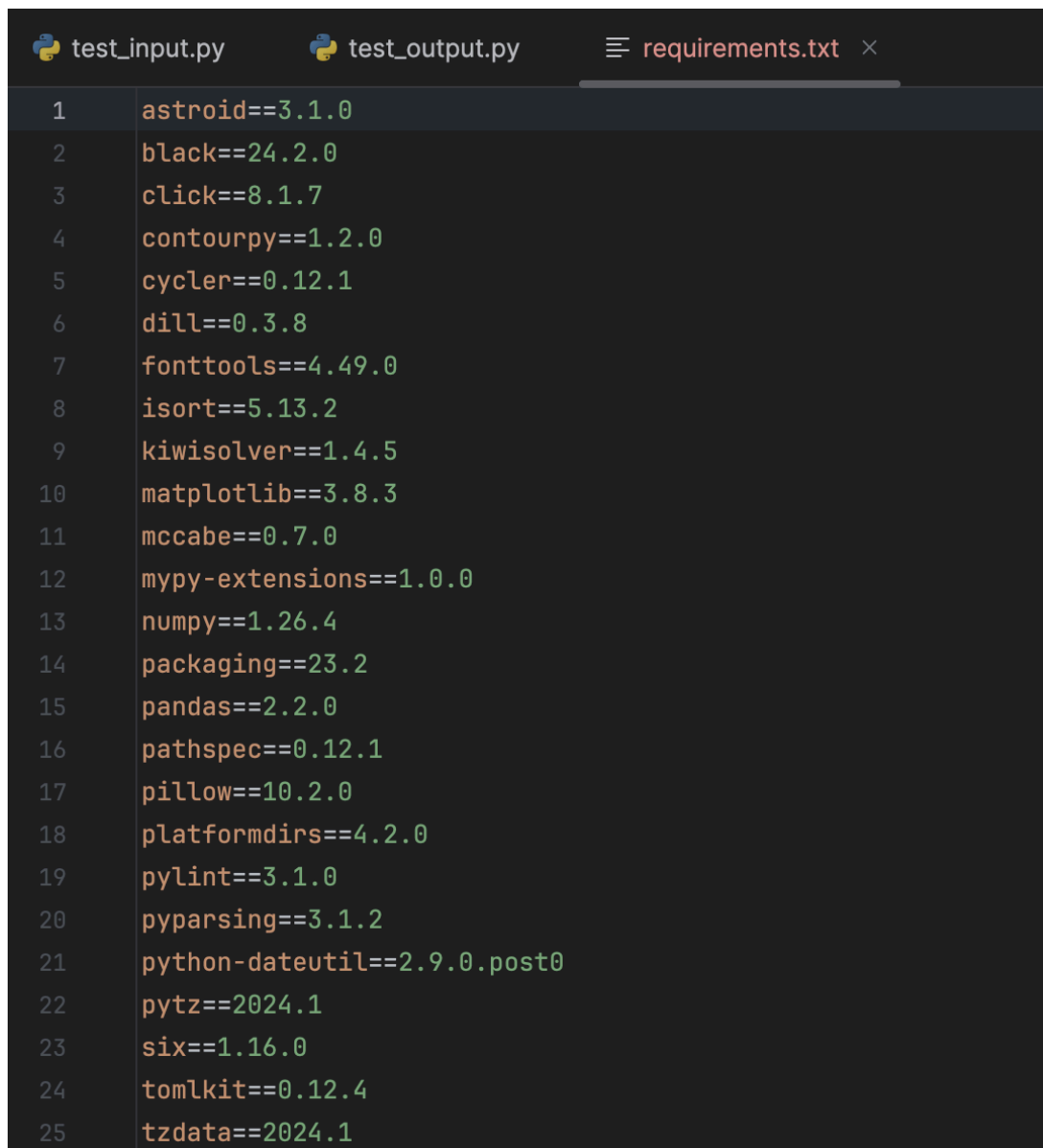
Тепер інші розробники, маючи цей файл можуть

автоматично інсталювати всі ті самі пакети та версії за допомогою команди `python -m pip install -r requirements.txt`

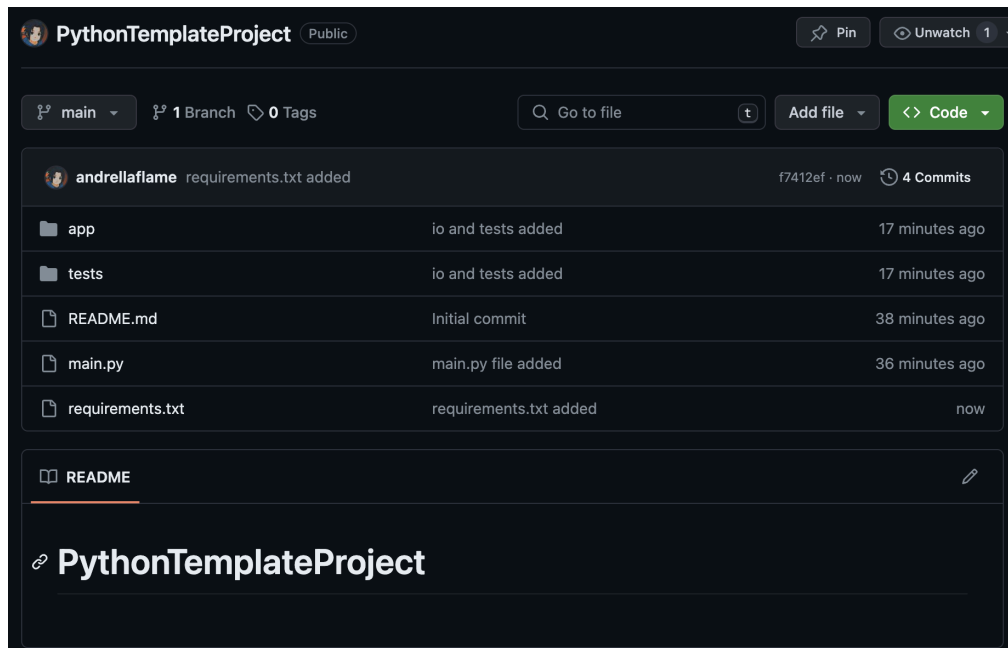
Більше про цей файл та випадки використання можна прочитати тут:

https://pip.pypa.io/en/latest/user_guide/#requirements-file

s 7. Зробіть commit з відповідним повідомленням.



```
test_input.py  test_output.py  requirements.txt x
1  astroid==3.1.0
2  black==24.2.0
3  click==8.1.7
4  contourpy==1.2.0
5  cyclер==0.12.1
6  dill==0.3.8
7  fonttools==4.49.0
8  isort==5.13.2
9  kiwisolver==1.4.5
10 matplotlib==3.8.3
11 mccabe==0.7.0
12 mypy-extensions==1.0.0
13 numpy==1.26.4
14 packaging==23.2
15 pandas==2.2.0
16 pathspec==0.12.1
17 pillow==10.2.0
18 platformdirs==4.2.0
19 pylint==3.1.0
20 pyparsing==3.1.2
21 python-dateutil==2.9.0.post0
22 pytz==2024.1
23 six==1.16.0
24 tomlkit==0.12.4
25 tzdata==2024.1
```



4. Робота з файлами.

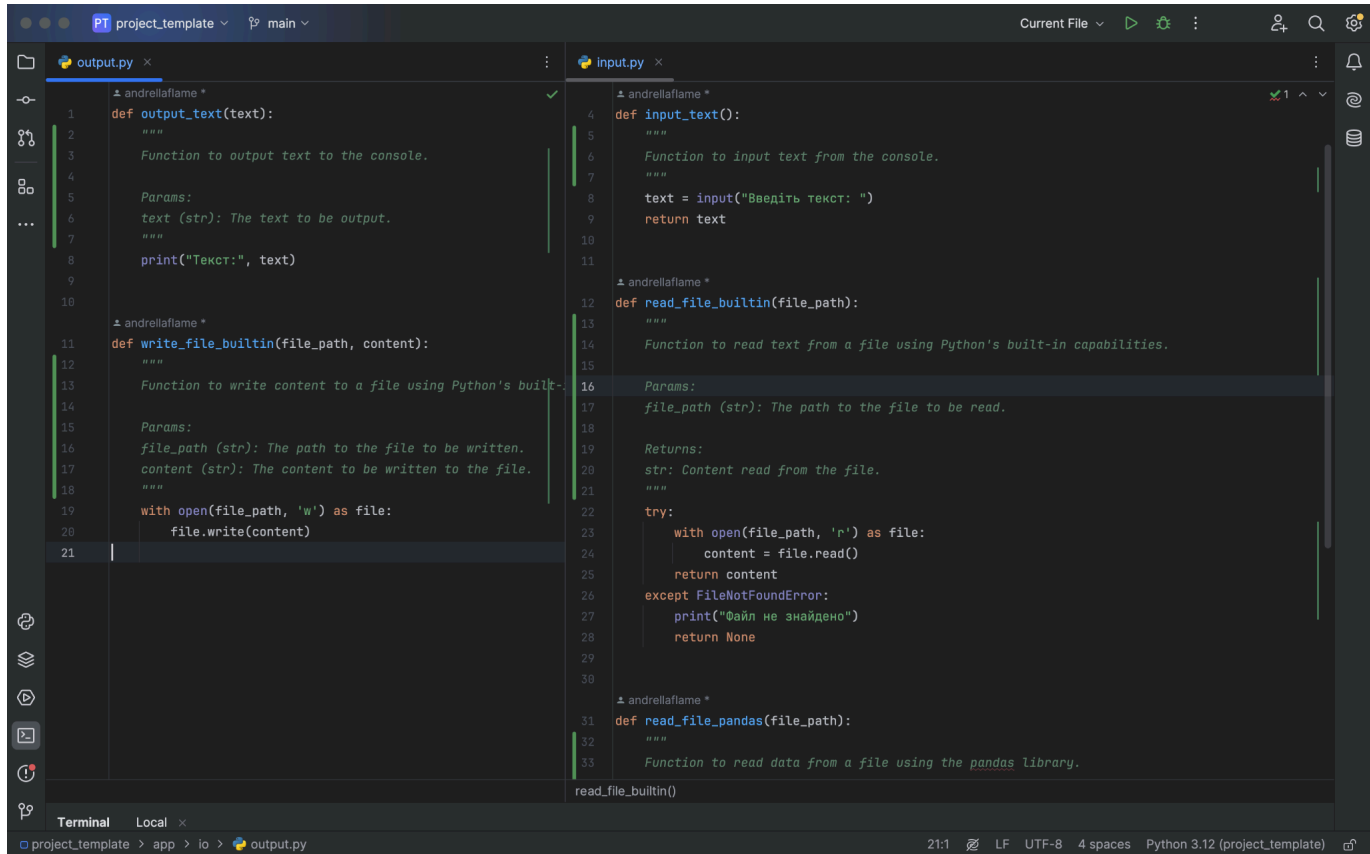
1. У файлі `input.py` створіть пусті 3 функції: 1) для вводу тексту з консолі, 2) для зчитування з файлу за допомогою вбудованих можливостей python, 3) для зчитування з файлу за допомогою бібліотеки `pandas`.
2. У файлі `output.py` створіть пусті 3 функції: 1) для виводу тексту у консоль, 2) для запису до файлу за допомогою вбудованих можливостей python.
3. Зробіть ще один commit з відповідним повідомленням на цьому кроці.



4. Створіть docstrings для всіх цих функцій.
5. У `main.py` у функції `main()` доповніть її тіло викликами виществорених функцій так, щоб текстові результати, що

повертаються функціями 4.1.1) , 4.1.2) та 4.1.3) були виведені у консоль, а також записані до файлу через вбудовані можливості python.

6. Реалізуйте ці функції.




```
1  # andrellafame *
2  def output_text(text):
3      """
4      Function to output text to the console.
5
6      Params:
7      text (str): The text to be output.
8      """
9      print("Текст:", text)
10
11 # andrellafame *
12 def write_file_builtin(file_path, content):
13     """
14     Function to write content to a file using Python's built-in capabilities.
15
16     Params:
17     file_path (str): The path to the file to be written.
18     content (str): The content to be written to the file.
19     """
20     with open(file_path, 'w') as file:
21         file.write(content)
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

7. За потреби, ви можете створити окрему папку для даних (файлів) у кореневій папці проєкту з назвою data. Обов'язково додайте її до .gitignore.

```
output.py  main.py x
1 from app.io.input import input_text, read_file_builtin, read_file_pandas
2 from app.io.output import output_text, write_file_builtin
3
4
5 1 usage  andrellafame
6 def main():
7     text_input = input_text()
8
9     file_content_builtin = read_file_builtin('data/input_file.txt')
10    file_data_pandas = read_file_pandas('data/data.csv')
11
12    output_text(text_input)
13    output_text(file_content_builtin)
14    output_text(file_data_pandas)
15
16    write_file_builtin(filename: 'data/output_file.txt', text_input)
17    write_file_builtin(filename: 'data/output_file_builtin.txt', file_content_builtin)
18
19 if __name__ == '__main__':
20     main()
```

8. Зробіть commit з відповідним повідомленням.

 andrellafame	data folder added	f3fb5a4 · now	6 Commits
app	data folder added	now	
tests	io and tests added	35 minutes ago	
README.md	Initial commit	1 hour ago	
main.py	data folder added	now	
requirements.txt	requirements.txt added	18 minutes ago	

5. *(На оцінку 90+). Написання тестів.

Використовуючи пакети unittest або pytest на ваш вибір, напишіть по три тести до функцій 2 та 3 (зчитування з файлів) з файлу input.py. Після написання тестів для кожної окремої функції дуже рекомендую робити commit.

```

new *
class TestInput(unittest.TestCase):
    new *
    def test_read_file_builtin(self):
        content = read_file_builtin('../data/input_files/input_file.txt')
        print(content)
        self.assertIsInstance(content, str)
        self.assertNotEqual(content, second: '')

        content = read_file_builtin('nonexistent_file.txt')
        self.assertIsNone(content)

    new *
    def test_read_file_pandas(self):
        data = read_file_pandas('../data/input_files/data.csv')
        self.assertIsInstance(data, pd.DataFrame)
        self.assertNotEqual(data.shape, second: (0, 0))

        data = read_file_pandas('nonexistent_data.csv')
        self.assertIsNone(data)

if __name__ == '__main__':
    unittest.main()

```

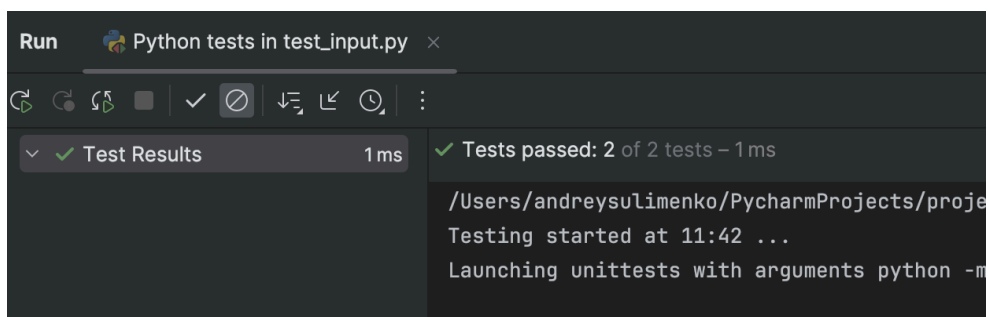
Ресурси, які можуть вам бути корисні:

<https://docs.python.org/3/library/unittest.html>

<https://docs.pytest.org/en/7.4.x/getting-started.html>

<https://realpython.com/python-testing/>

<https://www.dataquest.io/blog/unit-tests-python/>



6. Висновки.

а. Що зробили?

У цій роботі ми попрактикувались у створенні проектів, проектних модулів, роботі з бібліотеками, рір та написанням тестів. Ми реалізували функції для вводу тексту з консолі, читання з файлу за допомогою вбудованих можливостей Python та читання з файлу за допомогою бібліотеки pandas. Також ми створили функції для виведення тексту у консоль та запису до файлу за допомогою вбудованих можливостей Python.

б. Що нового дізнались для себе?

Для себе, я розібрався, як працювати з проектними модулями та сторонніми бібліотеками в Python, а також з використанням рір.

с. Що було корисним? Що б Ви використали в майбутньому?

Вся робота була корисною. В майбутньому точно використав би рір та unit тести.

д. Що можна було б покращити нам для студентів в цій роботі?

Все було чудово. Не може сказати однозначно, що можна було б покращити.

7. Надсилання звіту.

а. Готовий звіт прикріпити у Мудл згідно дедлайнів.

8. Наостанок.

Похваліть себе, Ви дуже багато зусиль доклали! Побалуйте себе відпочинком або якимось смаколиком.

Дякую, що доклали зусиль, у Вас вийшло!