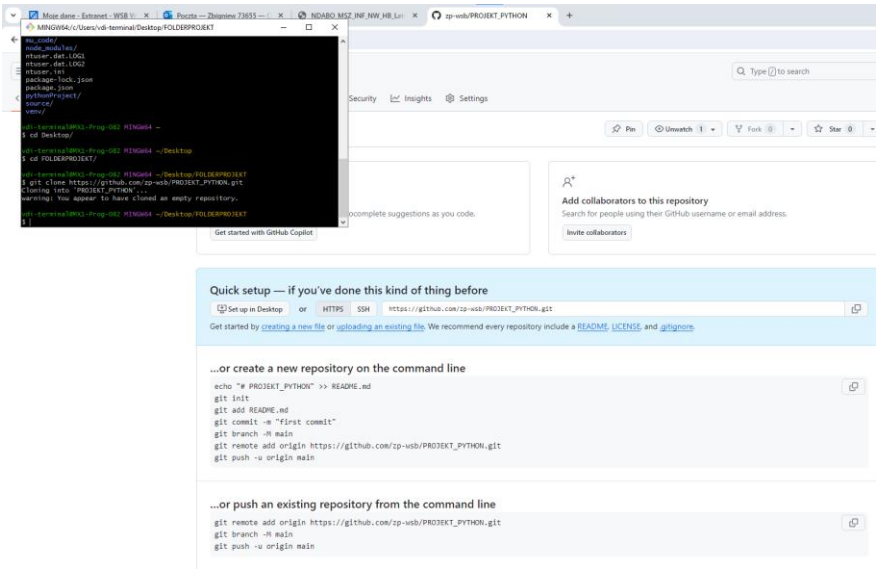
LIASTA NR 2

Zadanie 1: Tworzenie i konfigurowanie CI/CD z użyciem GitHub Actions

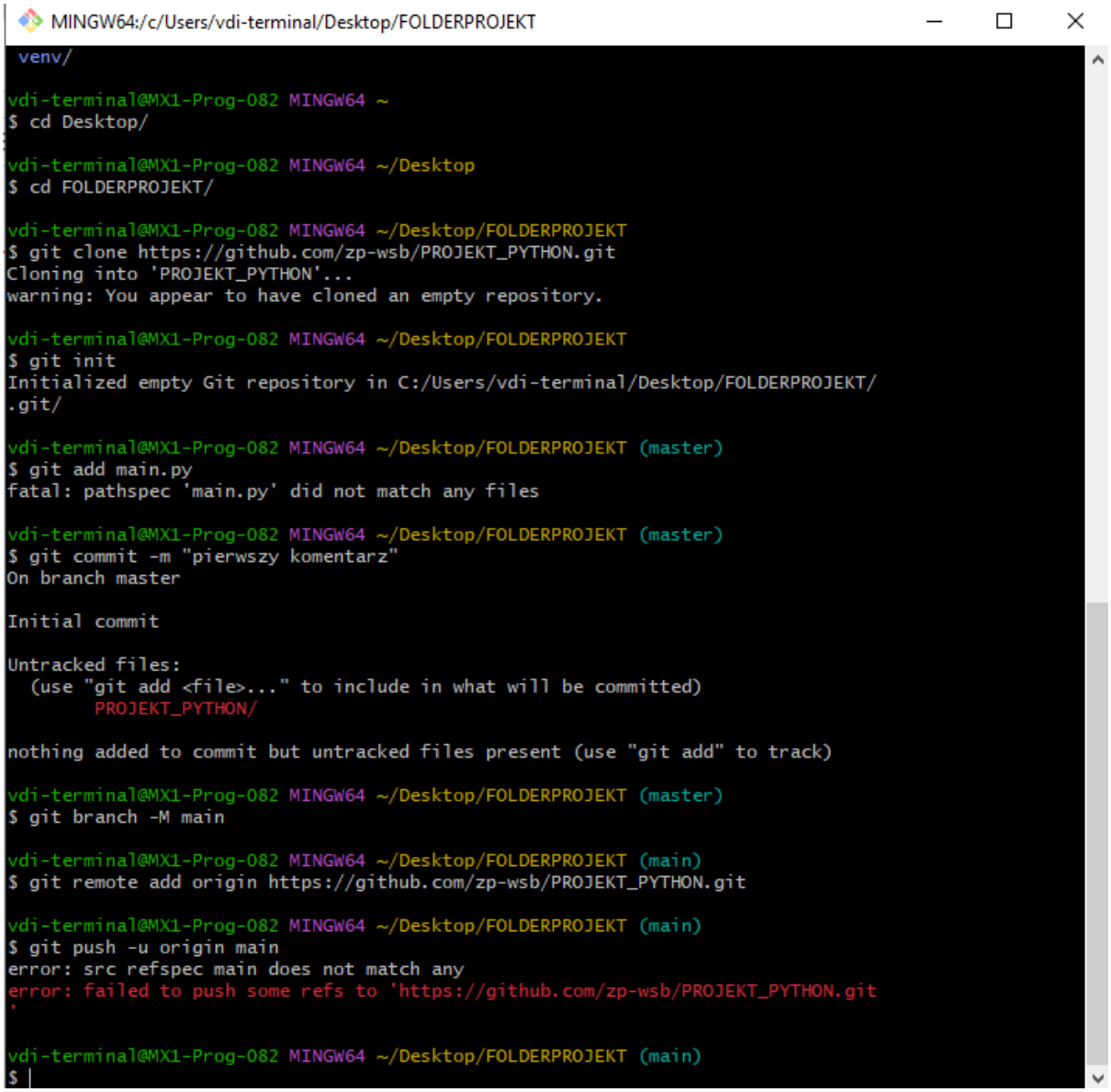
Stwórz projekt w języku Python, który zawiera prostą aplikację webową z użyciem frameworka Flask. Następnie skonfiguruj CI/CD za pomocą GitHub Actions, aby automatycznie budować, testować i wdrażać aplikację przy każdym pushu do repozytorium. Link do repozytorium z kodem należy umieścić na platformie Moodle.

Kroki do wykonania:

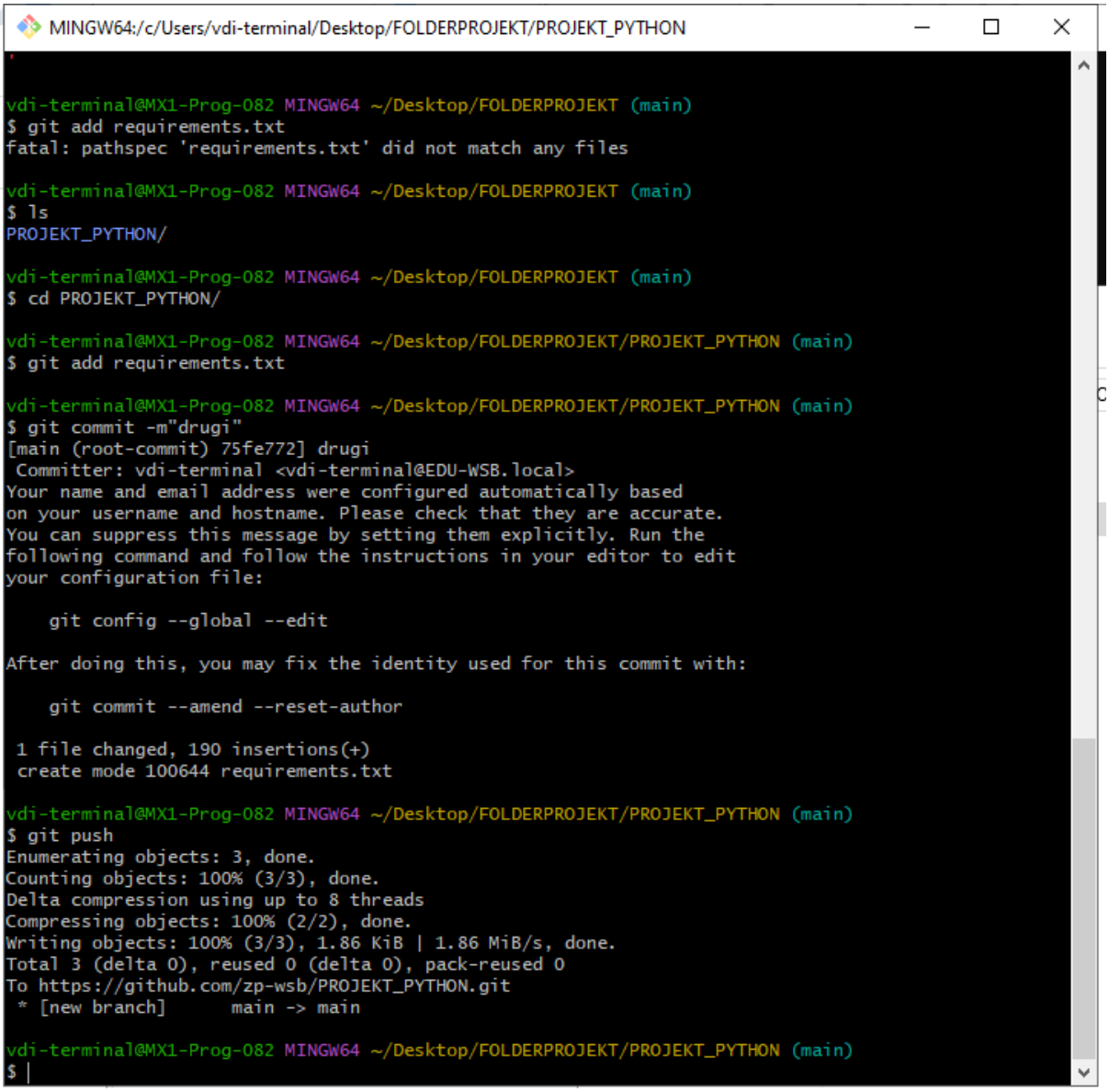
1. **Stworzenie repozytorium**:
    - Utwórz nowe repozytorium na GitHubie i sklonuj je na swój komputer.



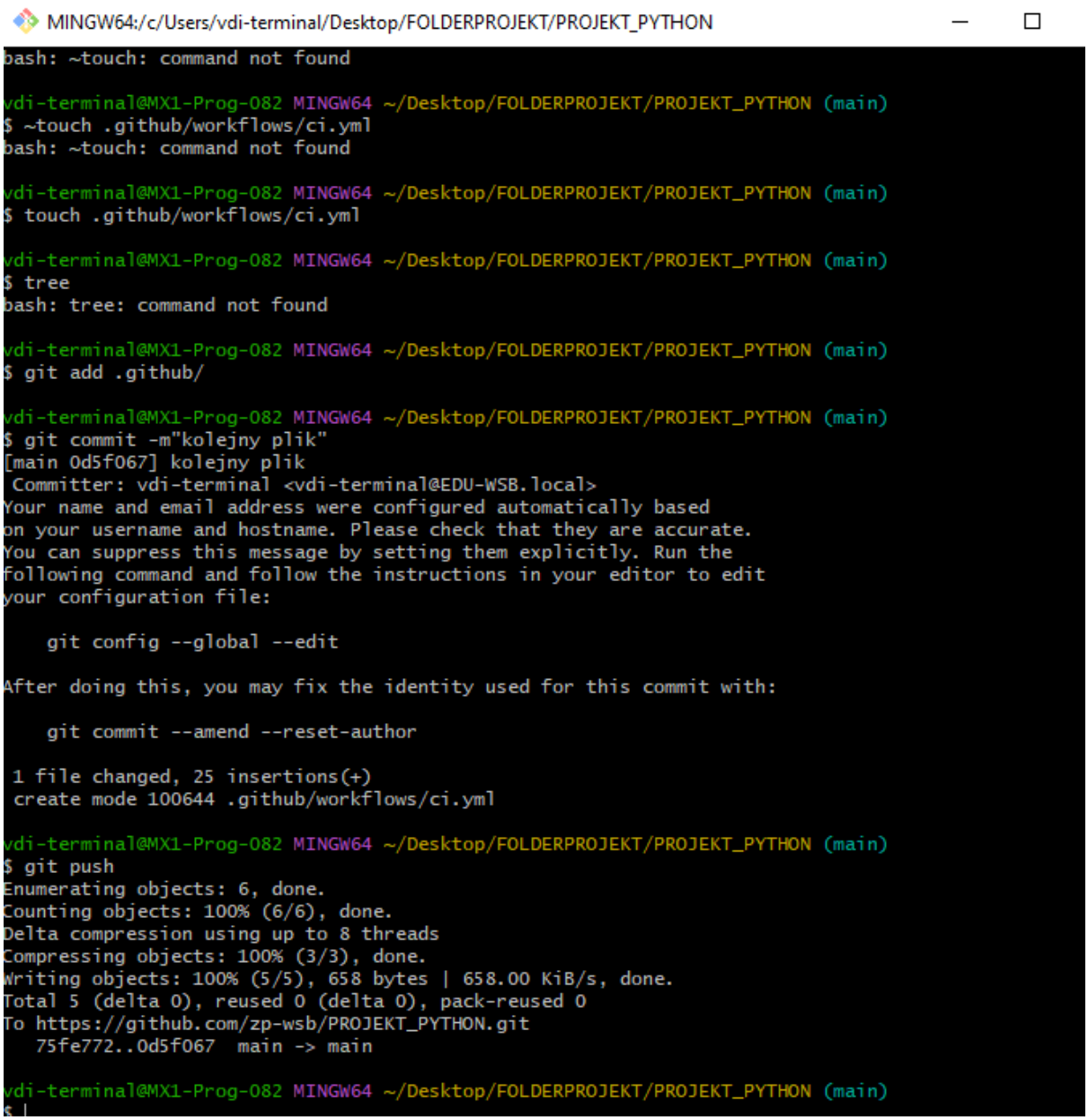
    - Dodaj prostą aplikację Flask do repozytorium (np. app.py), która wyświetla "Hello, World!" na stronie głównej.



2. **Utworzenie plików konfiguracyjnych**:
    - Dodaj plik requirements.txt z listą wymaganych bibliotek (np. Flask).

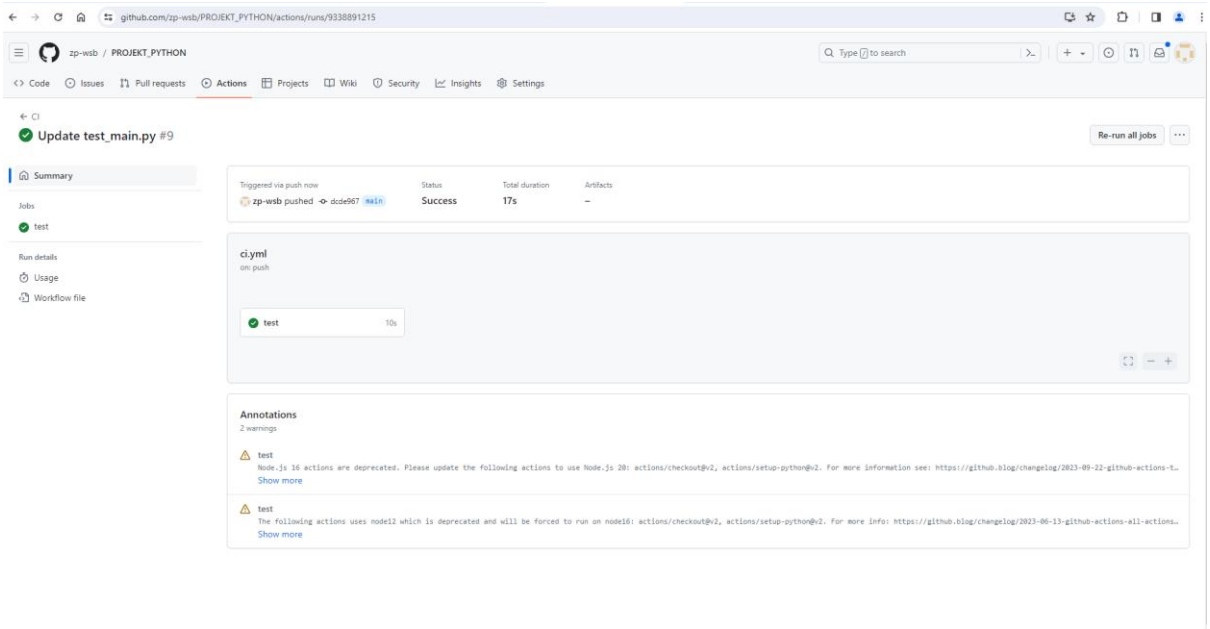


3. **Konfiguracja GitHub Actions**:
    - W repozytorium, w katalogu .github/workflows, utwórz plik konfiguracyjny GitHub Actions (np. ci.yml).
    - Skonfiguruj CI/CD, który:
        - ✦ Uruchamia testy (np. używając pytest).



4. **Dodanie testów**:

- Stwórz katalog tests i dodaj prosty test sprawdzający odpowiedź aplikacji Flask.



Wymagania:

- Plik konfiguracyjny GitHub Actions.
- Plik requirements.txt.
- Prosta aplikacja Flask (app.py).
- Testy jednostkowe w katalogu tests.

Przydatna dokumentacja:

- https://blog.miguelgrinberg.com/post/the-flask-mega-tutorial-part-i-hello-world
- https://flask.palletsprojects.com/en/3.0.x/tutorial/
- https://docs.github.com/en/actions/learn-github-actions/understandinggithub-actions
- https://docs.github.com/en/actions/quickstart

Zadanie 2: Tworzenie pliku Docker dla aplikacji Flask

Opis zadania:

Przygotuj plik Dockerfile dla aplikacji Flask z Zadania 1, który umożliwi uruchomienie aplikacji w kontenerze Docker. Następnie uruchom kontener lokalnie i upewnij się, że aplikacja działa poprawnie. Screenshot potwierdzający działanie aplikacji należy wrzucić do repozytorium. Wszystkie dodatkowe pliki powinny znaleźć się w repozytorium utworzonym do Zadania 1.

Kroki do wykonania:

1. **Przygotowanie plików:**

• Upewnij się, że masz aplikację Flask z Zadania 1 w swoim repozytorium, w tym pliki **app.py** i **requirements.txt**.

2. **Tworzenie pliku Dockerfile:**

- W głównym katalogu projektu stwórz plik **Dockerfile**.
- Skonfiguruj Dockerfile w taki sposób, aby:
    - Bazował na oficjalnym obrazie Python (min. python:3.9).
    - Skopiował pliki aplikacji do obrazu.
    - Zainstalował wymagane biblioteki z **requirements.txt**. o Ustawił zmienną środowiskową **FLASK_APP** na **app.py**.
    - Uruchamiał aplikację Flask za pomocą flask run.

3. **Budowanie obrazu Docker**:

- Otwórz terminal i przejdź do katalogu projektu.
- Wykonaj polecenie, aby zbudować obraz Docker: • `docker build -t flask-app .`

4. **Uruchamianie kontenera**:

• Po zbudowaniu obrazu uruchom kontener za pomocą polecenia: •

```
docker run -p 5000:5000 flask-app
```

5. Weryfikacja działania:

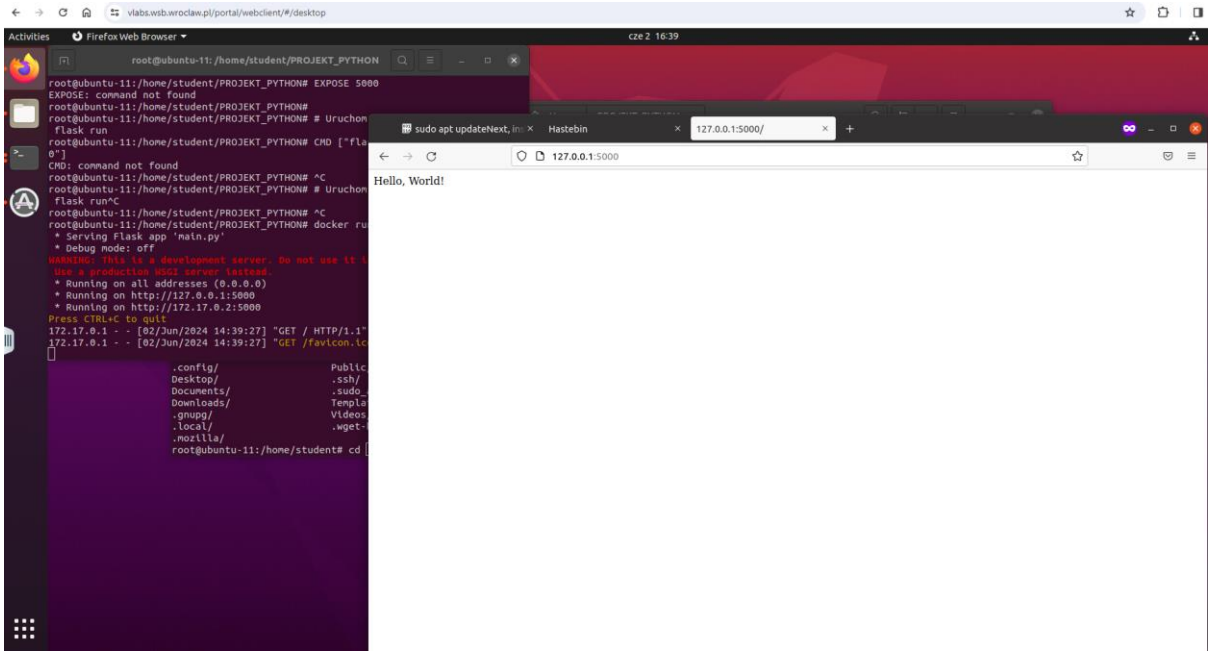
Otwórz przeglądarkę i przejdź do adresu http://localhost:5000. Upewnij się, że aplikacja Flask wyświetla "Hello, World!".

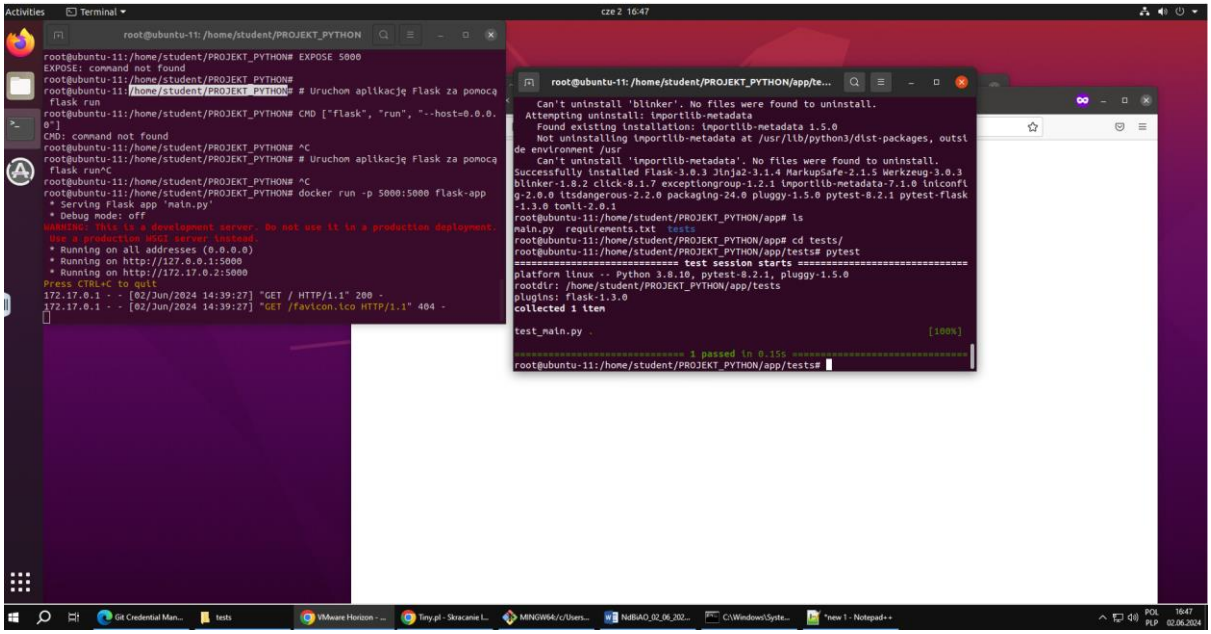
Wymagania:

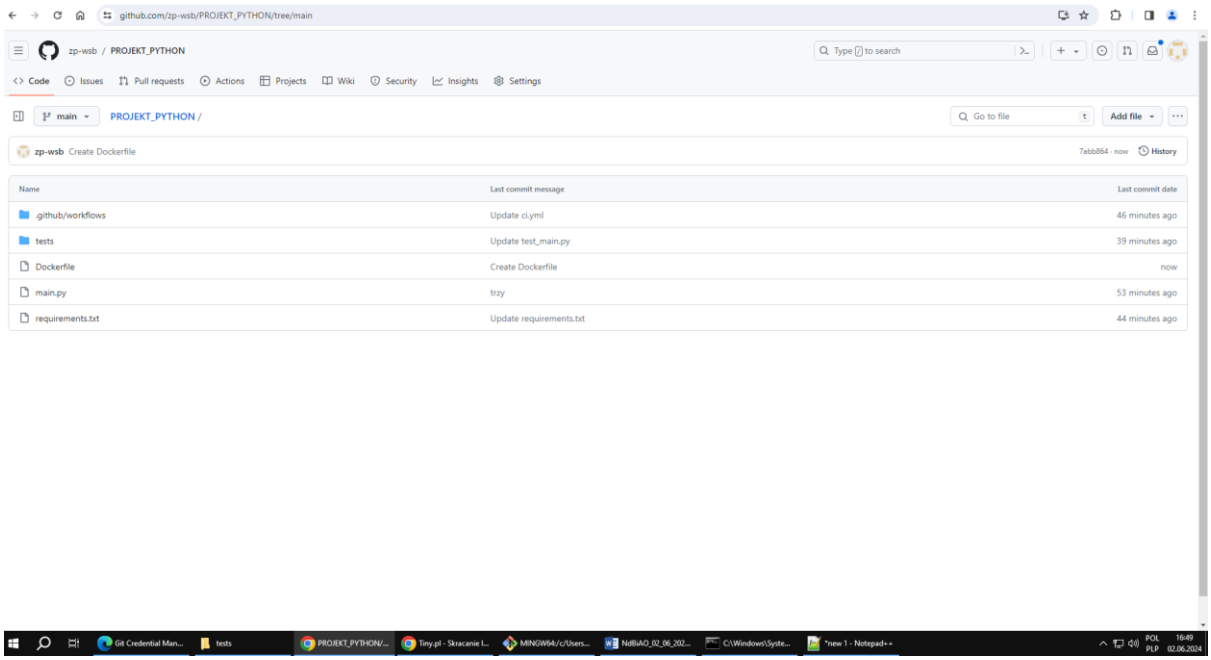
- Plik Dockerfile w głównym katalogu projektu.
- Aplikacja Flask (app.py).
- Plik requirements.txt.
- Działający obraz Docker uruchamiający aplikację Flask.

Przydatna dokumentacja:

- https://docs.docker.com/get-started/
- https://www.digitalocean.com/community/tutorials/how-to-build-anddeploy-a-flask-application-using-docker-on-ubuntu-20-04


vlabs.wsb.wroclaw.pl/portal/webclient/#/desktop
Activities
Firefox Web Browser
cze 2 16:39
root@ubuntu-11: /home/student/PROJEKT_PYTHON
root@ubuntu-11:/home/student/PROJEKT_PYTHON# EXPOSE 5000
EXPOSE: command not found
root@ubuntu-11:/home/student/PROJEKT_PYTHON#
root@ubuntu-11:/home/student/PROJEKT_PYTHON# # Uruchom
flask run
root@ubuntu-11:/home/student/PROJEKT_PYTHON# CMD ["fla
0"]
CMD: command not found
root@ubuntu-11:/home/student/PROJEKT_PYTHON# ^C
root@ubuntu-11:/home/student/PROJEKT_PYTHON# # Uruchom
flask run^C
root@ubuntu-11:/home/student/PROJEKT_PYTHON# ^C
root@ubuntu-11:/home/student/PROJEKT_PYTHON# docker ru
* Serving Flask app 'main.py'
* Debug mode: off
WARNING: This is a development server. Do not use it
Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://172.17.0.2:5000
Press CTRL+C to quit
172.17.0.1 - - [02/Jun/2024 14:39:27] "GET / HTTP/1.1"
172.17.0.1 - - [02/Jun/2024 14:39:27] "GET /favicon.ic
.config/
Desktop/
Documents/
Downloads/
.gnupg/
.local/
.mozilla/
root@ubuntu-11:/home/student# cd
sudo apt updateNext, ins
Hastebin
127.0.0.1:5000/
127.0.0.1:5000
Hello, World!


Activities
Terminal
cze 2 16:47
root@ubuntu-11: /home/student/PROJEKT_PYTHON
root@ubuntu-11:/home/student/PROJEKT_PYTHON# EXPOSE 5000
EXPOSE: command not found
root@ubuntu-11:/home/student/PROJEKT_PYTHON#
root@ubuntu-11:/home/student/PROJEKT_PYTHON# # Uruchom aplikację Flask za pomocą
flask run
root@ubuntu-11:/home/student/PROJEKT_PYTHON# CMD ["flask", "run", "--host=0.0.0.
0"]
CMD: command not found
root@ubuntu-11:/home/student/PROJEKT_PYTHON# ^C
root@ubuntu-11:/home/student/PROJEKT_PYTHON# # Uruchom aplikację Flask za pomocą
flask run^C
root@ubuntu-11:/home/student/PROJEKT_PYTHON# ^C
root@ubuntu-11:/home/student/PROJEKT_PYTHON# docker run -p 5000:5000 flask-app
* Serving Flask app 'main.py'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://172.17.0.2:5000
Press CTRL+C to quit
172.17.0.1 - - [02/Jun/2024 14:39:27] "GET / HTTP/1.1" 200 -
172.17.0.1 - - [02/Jun/2024 14:39:27] "GET /favicon.ico HTTP/1.1" 404 -
root@ubuntu-11: /home/student/PROJEKT_PYTHON/app/te...
Can't uninstall 'blinker'. No files were found to uninstall.
Attempting uninstall: importlib-metadata
Found existing installation: importlib-metadata 1.5.0
Not uninstalling importlib-metadata at /usr/lib/python3/dist-packages, outsi
de environment /usr
Can't uninstall 'importlib-metadata'. No files were found to uninstall.
Successfully installed Flask-3.0.3 Jinja2-3.1.4 MarkupSafe-2.1.5 Werkzeug-3.0.3
blinker-1.8.2 click-8.1.7 exceptiongroup-1.2.1 importlib-metadata-7.1.0 inlconfl
g-2.0.0 itsdangerous-2.2.0 packaging-24.0 pluggy-1.5.0 pytest-8.2.1 pytest-flask
-1.3.0 tomli-2.0.1
root@ubuntu-11:/home/student/PROJEKT_PYTHON/app# ls
main.py requirements.txt tests
root@ubuntu-11:/home/student/PROJEKT_PYTHON/app# cd tests/
root@ubuntu-11:/home/student/PROJEKT_PYTHON/app/tests# pytest
test session starts
platform linux -- Python 3.8.10, pytest-8.2.1, pluggy-1.5.0
rootdir: /home/student/PROJEKT_PYTHON/app/tests
plugins: flask-1.3.0
collected 1 item
test_main.py .
[100%]
1 passed in 0.15s
root@ubuntu-11:/home/student/PROJEKT_PYTHON/app/tests#


github.com/zp-wsb/PROJEKT_PYTHON/tree/main
zp-wsb / PROJEKT_PYTHON
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
main
PROJEKT_PYTHON /
Go to file
Add file
zp-wsb Create Dockerfile
7abb864 · now
History
Name
Last commit message
Last commit date
.github/workflows
Update ci.yml
46 minutes ago
tests
Update test_main.py
39 minutes ago
Dockerfile
Create Dockerfile
now
main.py
trzy
53 minutes ago
requirements.txt
Update requirements.txt
44 minutes ago