

Uniwersytet Śląski w Katowicach		Wydział Nauk Ścisłych i Technicznych			
		Instytut Fizyki			
		Rok	III	Semestr	V
Kierunek	Informatyka stosowana				
Przedmiot	SiNWO - laboratorium				
Prowadzący	dr Wojciech Gurdziel				
Tytuł ćwiczenia	Wprowadzenie do konteneryzacji			Nr ćwiczenia	V
Sprawozdanie wykonał: (Imię i Nazwisko)	Jakub Kraus				
Data wykonania ćwiczenia	24.01.2024	Data oddania sprawozdania		29.01.2024	

Spis treści

1	Cel ćwiczenia	3
2	Przebieg ćwiczenia	3
2.1	Aplikacja Flask	3
2.1.1	app.py	3
2.1.2	requirements.txt	3
2.2	Konteneryzacja[1]	4
2.2.1	Dockerfile	4
2.3	Budowa obrazu	5
2.4	Uruchomienie kontenera	5
2.5	Umieszczenie obrazu w repozytorium	6
3	Wnioski	7
	Bibliografia	7

1 Cel ćwiczenia

Celem ćwiczenia jest zapoznanie się z podstawowymi pojęciami związanymi z konteneryzacją oraz zbudowanie własnego obrazu Docker z aplikacją webową.

2 Przebieg ćwiczenia

Do wykonania ćwiczenia użyłem systemu **openSUSE Tumbleweed**[4], **Flask**[3] do utworzenia aplikacji webowej przy użyciu Python'a oraz **Podman**[5] jako zamiennik 1:1 dla dockera i można go zaliasować do

```
alias docker='podman'
```

Wybrałem go ponieważ jest otwartoźródłowy. Wszystkie polecenia wykonywałem w terminalu.

2.1 Aplikacja Flask

2.1.1 app.py

```
1 from flask import Flask
2 from datetime import datetime
3
4 app = Flask(__name__)
5
6 @app.route('/')
7 def display_info():
8     name = "Jakub_Kraus"
9     index_number = "344120"
10    current_datetime = datetime.now().strftime("%Y-%m-%d_%H:%M:%S")
11
12    return f"{name}\n{index_number}\n{current_datetime}"
13
14 if __name__ == '__main__':
15    app.run(debug=True, host='0.0.0.0', port=7777)
```

2.1.2 requirements.txt

```
1 flask==3.0.0
2 gunicorn==20.1.0
```

2.2 Konteneryzacja[1]


2.2.1 Dockerfile

```
1      # Używamy obrazu Pythona w wersji 3.10, który jest lekki (slim).
2      FROM python:3.10-slim
3
4      # Otwarcie i przypisanie portu 7777, aby można było korzystać z
5      # aplikacji na tym porcie.
6      EXPOSE 7777
7
8      # Ustawiamy zmienną środowiskową, aby zapobiec tworzeniu plików
9      # pycache.
10     ENV PYTHONDONTWRITEBYTECODE=1
11
12     # Ustawiamy zmienną środowiskową, aby uniknąć buforowania w Pythonie.
13     ENV PYTHONUNBUFFERED=1
14
15     # Kopiujemy plik requirements.txt do obecnego katalogu.
16     COPY requirements.txt .
17
18     # Instalujemy zależności wymienione w pliku requirements.txt.
19     RUN python -m pip install -r requirements.txt
20
21     # Ustawiamy katalog roboczy na /app.
22     WORKDIR /app
23
24     # Kopiujemy wszystkie pliki z obecnego katalogu do katalogu /app w
25     # kontenerze.
26     COPY . /app
27
28     # Dodajemy użytkownika o identyfikatorze 5678, bez hasła, bez
29     # interaktywnego prompta.
30     # Następnie zmieniamy właściciela katalogu /app na użytkownika appuser
31     .
32     RUN adduser -u 5678 --disabled-password --gecos "" appuser && chown -R
33     appuser /app
34
35     # Ustawiamy użytkownika appuser jako użytkownika, który będzie używany
36     # do uruchamiania kontenera.
37     USER appuser
38
39     # Komenda, która zostanie wykonana, gdy kontener zostanie uruchomiony.
40     # Uruchamiamy Gunicorn (web server dla Pythona) naszej aplikacji na
41     # porcie 7777.
42     CMD ["gunicorn", "--bind", "0.0.0.0:7777", "app:app"]
```

2.3 Budowa obrazu

Aby zbudować obraz, należy wykonać polecenie:

```
1 docker build -t sinwo:latest .
```



```
SIPD_5 on /y main via v3.11.7
podman build
STEP 1/11: FROM python:3.10-slim
STEP 2/11: EXPOSE 7777
-> Using cache 2e494f97e85243addaa8767e01fa2b05bd8f8dd8ffbc8f90c610773c3e0c72
-> 2e494f97e852
STEP 3/11: ENV PYTHONDONTWRITEBYTECODE=1
-> Using cache 8345ad78aa0236edb5b07c6e0dc77db25f60f40ecb61ea23b5f60f66f1b3f8
-> 8345ad78aa02
STEP 4/11: ENV PYTHONUNBUFFERED=1
-> Using cache 6148167749808b76f8bb46d9655150e1662cde8594356575bf157125a2583ae
-> 614816774980
STEP 5/11: COPY requirements.txt .
-> 674ec3e22c85
STEP 6/11: RUN python -m pip install -r requirements.txt
Collecting flask==3.0.0
  Downloading flask-3.0.0-py3-none-any.whl (99 kB)
    99.7/99.7 kB 1.5 MB/s eta 0:00:00
Collecting gunicorn==20.1.0
  Downloading gunicorn-20.1.0-py3-none-any.whl (79 kB)
    79.5/79.5 kB 7.9 MB/s eta 0:00:00
Collecting Werkzeug<=3.0.0
  Downloading werkzeug-3.0.1-py3-none-any.whl (226 kB)
    226.7/226.7 kB 5.8 MB/s eta 0:00:00
Collecting blinker<=1.6.2
  Downloading blinker-1.7.0-py3-none-any.whl (13 kB)
Collecting Jinja2<=3.1.2
  Downloading Jinja2-3.1.3-py3-none-any.whl (133 kB)
    133.2/133.2 kB 23.4 MB/s eta 0:00:00
Collecting click<=8.1.3
  Downloading click-8.1.7-py3-none-any.whl (97 kB)
    97.9/97.9 kB 23.1 MB/s eta 0:00:00
Collecting itsdangerous<=2.1.2
  Downloading itsdangerous-2.1.2-py3-none-any.whl (15 kB)
Requirement already satisfied: setuptools<=3.0 in /usr/local/lib/python3.10/site-packages (from gunicorn==20.1.0->-r requirements.txt (line 2)) (65.5.1)
Collecting MarkupSafe<=2.0
  Downloading MarkupSafe-2.1.4-cp310-cp310-manylinux_2_17_x86_64_manylinux2014_x86_64.whl (25 kB)
Installing collected packages: MarkupSafe, itsdangerous, gunicorn, click, blinker, Werkzeug, Jinja2, flask
Successfully installed Jinja2-3.1.3 MarkupSafe-2.1.4 Werkzeug-3.0.1 blinker-1.7.0 click-8.1.7 flask-3.0.0 gunicorn-20.1.0 itsdangerous-2.1.2
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv

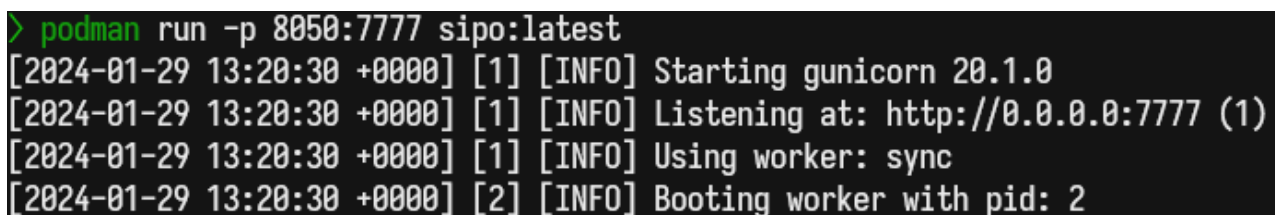
[notice] A new release of pip is available: 23.0.1 -> 23.3.2
[notice] To update, run: pip install --upgrade pip
-> 8a9d5308c306
STEP 7/11: WORKDIR /app
-> 218aa6b57c9c
STEP 8/11: COPY . /app
-> c59c88b55090
STEP 9/11: RUN adduser -u 5678 --disabled-password --gecos "" appuser && chown -R appuser /app
Adding user 'appuser' ...
Adding new group 'appuser' (5678) ...
Adding new user 'appuser' (5678) with group 'appuser' (5678) ...
Creating home directory /home/appuser ...
Copying files from /etc/skel ...
Adding new user 'appuser' to supplemental / extra groups 'users' ...
Adding user 'appuser' to group 'users' ...
-> 3a835fc90a7b
STEP 10/11: USER appuser
-> 71adca68975b
STEP 11/11: CMD ["gunicorn", "--bind", "0.0.0.0:7777", "app:app"]
COMMIT
-> 71d5b6efb271
71d5b6efb271ad5722a16c50f0610647fee900e01259b47d07efdb9cd9eca759
```

Rysunek 1: Wynik wykonania polecenia docker build

2.4 Uruchomienie kontenera

Aby uruchomić kontener, należy wykonać polecenie:

```
1 docker run -d -p 8050:7777 sinwo:latest
```



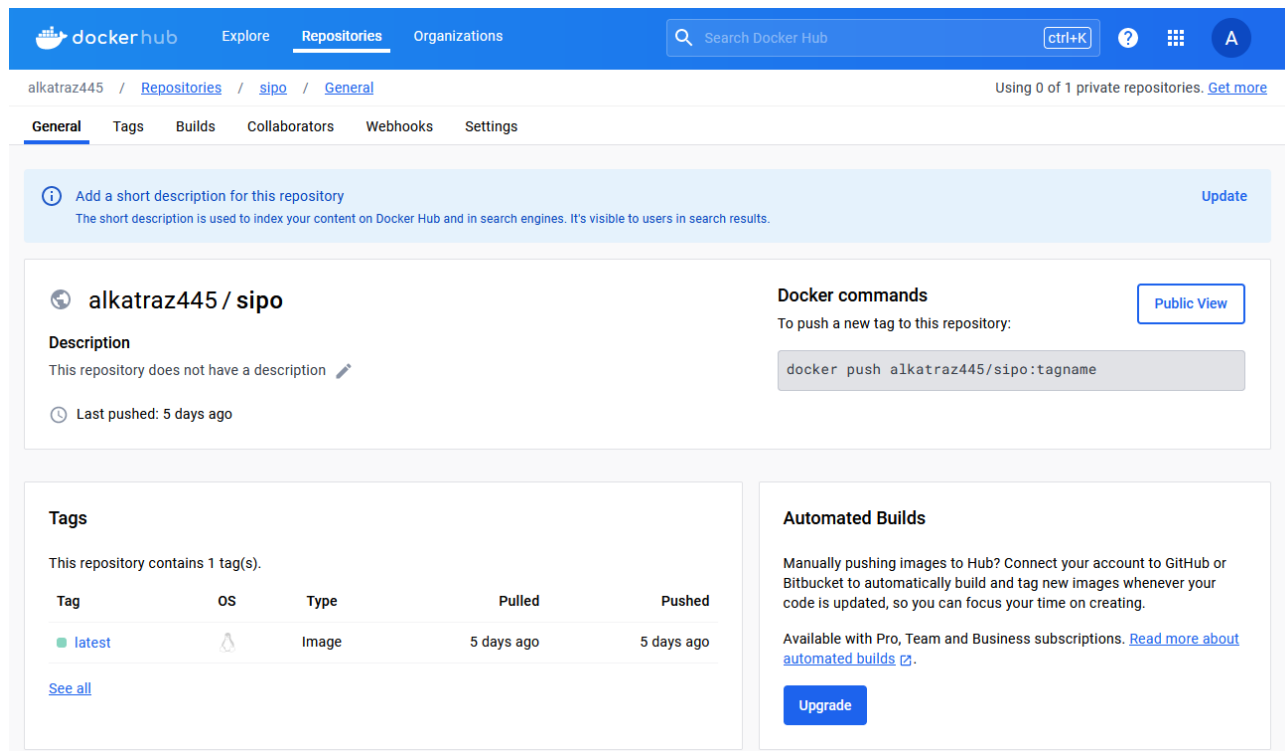
```
> podman run -p 8050:7777 sipo:latest
[2024-01-29 13:20:30 +0000] [1] [INFO] Starting gunicorn 20.1.0
[2024-01-29 13:20:30 +0000] [1] [INFO] Listening at: http://0.0.0.0:7777 (1)
[2024-01-29 13:20:30 +0000] [1] [INFO] Using worker: sync
[2024-01-29 13:20:30 +0000] [2] [INFO] Booting worker with pid: 2
```

Rysunek 2: Wynik wykonania polecenia docker run

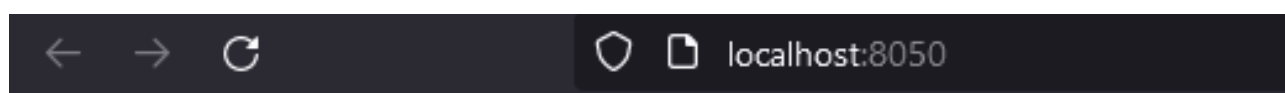
2.5 Umieszczenie obrazu w repozytorium

Aby umieścić obraz w repozytorium, należy wykonać polecenia:

- 1 `docker tag sinwo:latest nazwa_użytkownika_repozytorium/sinwo:latest`
- 2 `docker push nazwa_użytkownika_repozytorium/sinwo:latest`



Rysunek 3: Docker Hub z obrazem[2]



Rysunek 4: Strona internetowa

3 Wnioski

Konteneryzacja pozwala na łatwe i szybkie uruchamianie aplikacji na różnych systemach operacyjnych. Dzięki temu, że kontenery są odizolowane od siebie, można uruchomić wiele kontenerów z różnymi aplikacjami na jednym systemie.

Tworzenie obrazów jest bardzo proste dzięki dockerfile, a dzięki temu, że można je umieszczać w repozytoriach, można łatwo udostępniać swoje aplikacje innym użytkownikom.

Cały proces używania kontenerów jest przystępny i łatwy do zrozumienia, co spowodowało gwałtowną adaptację w środowiskach produkcyjnych. Idea konteneryzacji również zapewnia, że nie pojawiają się problemy znane ze świata informatyki, takie jak “U mnie działa”.

Bibliografia

- [1] *Docker documentation*. en. Sty. 2024. URL: <https://docs.docker.com/> (term. wiz. 24.01.2024).
- [2] *Docker Hub*. Repozytorium. URL: <https://hub.docker.com/repository/docker/alkatr445/sipo/general> (term. wiz. 24.01.2024).
- [3] *Flask*. URL: <https://flask.palletsprojects.com/en/3.0.x/> (term. wiz. 24.01.2024).
- [4] *openSUSE Tumbleweed*. en. URL: <https://get.opensuse.org/tumbleweed.html> (term. wiz. 24.01.2024).
- [5] *Podman*. en. URL: <https://podman.io/> (term. wiz. 24.01.2024).

Korzystałem również z pliku dostarczonego przez prowadzącego.