

1. Wprowadzenie

Celem zadania było przygotowanie prostej aplikacji API w Pythonie, umieszczenie jej w kontenerze Dockera oraz uruchomienie za pomocą narzędzia docker-compose. Efektem końcowym jest działający endpoint HTTP, zwracający odpowiedź JSON.

Projekt wykorzystuje:

- Flask jako framework do budowy API
- Gunicorn jako produkcyjny serwer aplikacji
- Docker i Docker Compose do konteneryzacji i automatycznego uruchamiania środowiska
- Multi-stage Dockerfile do optymalizacji obrazu

2. Struktura projektu

Name	Date
.dockerignore	Today
app.py	Today
docker-compose.yml	Today
Dockerfile	Today
README.md	Today
requirements.txt	Today

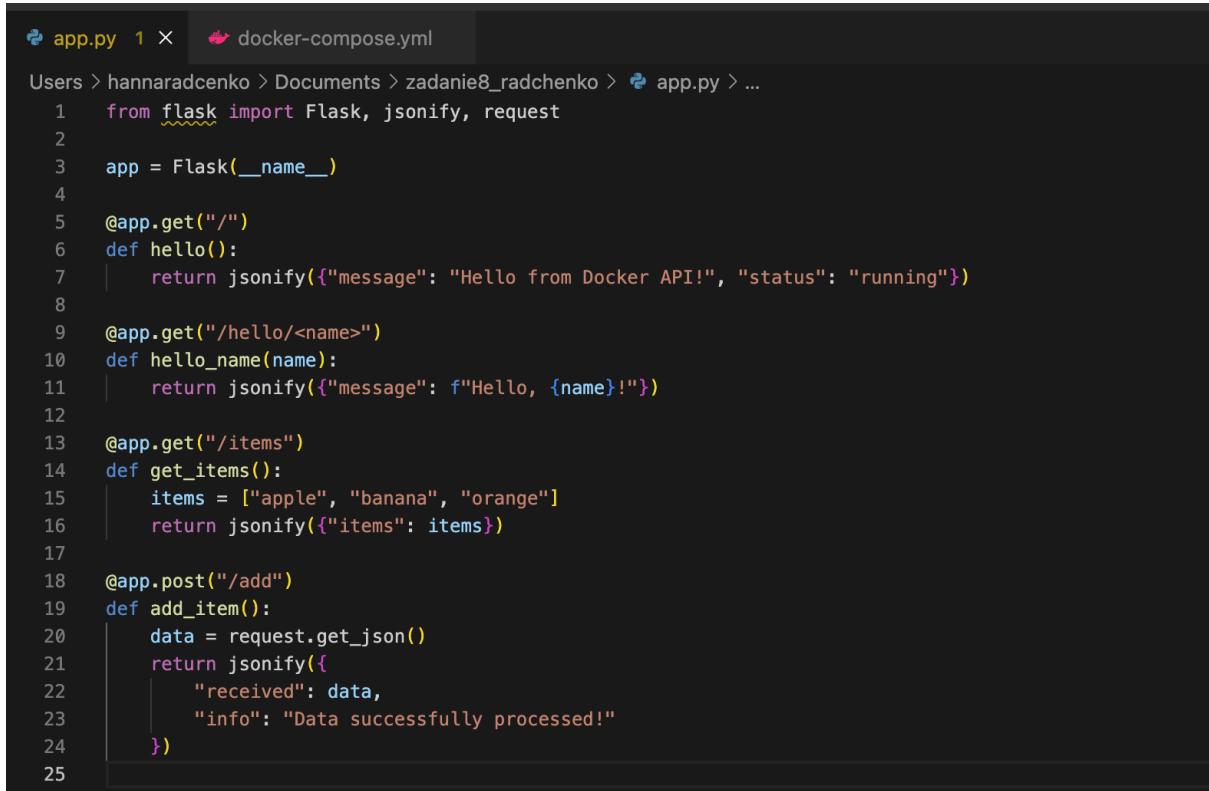
3. Kod aplikacji (API)

Aplikacja została napisana w Pythonie z użyciem frameworka Flask. Serwer zwraca prosty komunikat JSON pod adresem.

Kod tworzyinstancję aplikacji Flask i definiuje jedną ścieżkę – /.

Po jej wywołaniu serwer zwraca odpowiedź JSON za pomocą funkcji jsonify().

Kod:



```
app.py 1 × docker-compose.yml

Users > hannaradcenko > Documents > zadanie8_radchenko > app.py > ...
1  from flask import Flask, jsonify, request
2
3  app = Flask(__name__)
4
5  @app.get("/")
6  def hello():
7      return jsonify({"message": "Hello from Docker API!", "status": "running"})
8
9  @app.get("/hello/<name>")
10 def hello_name(name):
11     return jsonify({"message": f"Hello, {name}!"})
12
13 @app.get("/items")
14 def get_items():
15     items = ["apple", "banana", "orange"]
16     return jsonify({"items": items})
17
18 @app.post("/add")
19 def add_item():
20     data = request.get_json()
21     return jsonify({
22         "received": data,
23         "info": "Data successfully processed!"
24     })
25
```

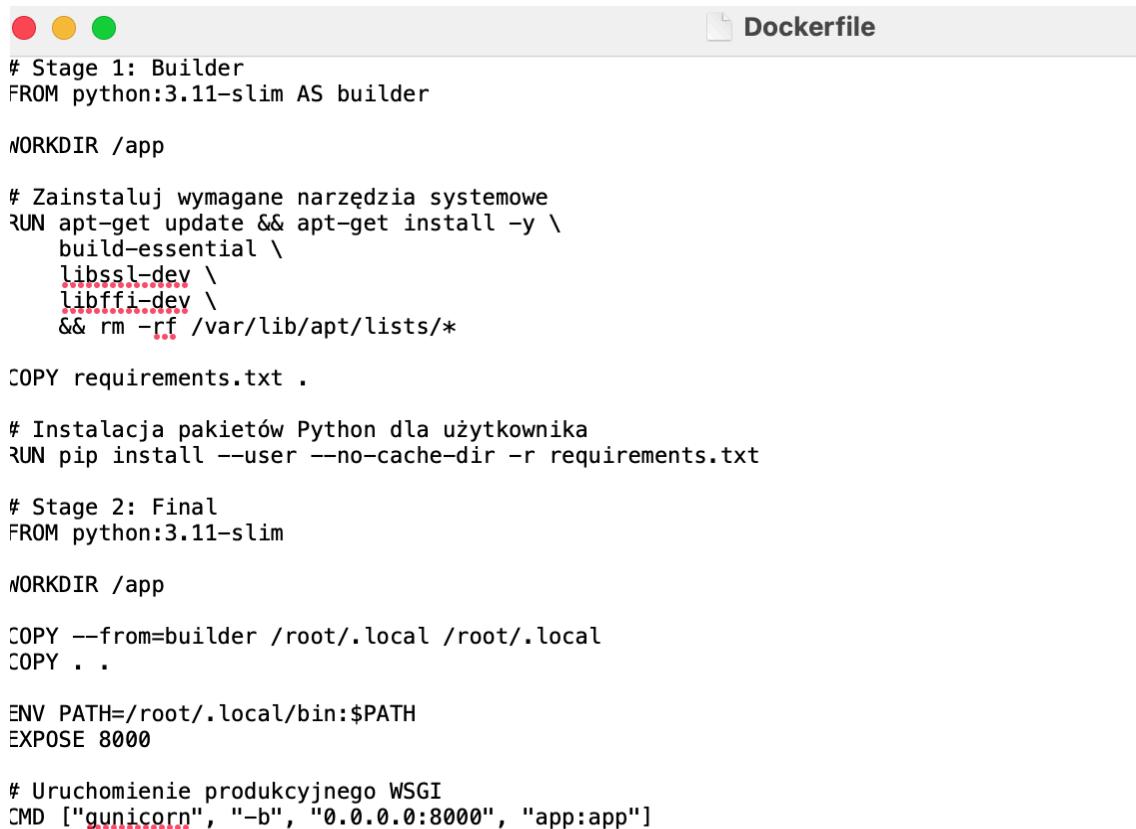
Dostępne endpointy:

Endpoint	Metoda	Opis
/	GET	Zwraca podstawowy komunikat JSON z informacją, że API działa.
/hello/<name>	GET	Zwraca spersonalizowany komunikat powitalny dla podanego imienia.

/items	GET	Zwraca listę przykładowych przedmiotów w formacie JSON.
/add	POST	Przyjmuje dane JSON w treści żądania i zwraca potwierdzenie ich odebrania.

4. Dockerfile

Dockerfile zawiera instrukcje do zbudowania obrazu Dockera z aplikacją Pythonową. Wykorzystuje oficjalny obraz `python:3.9-slim`, kopiuje pliki projektu, instaluje wymagane zależności i przygotowuje serwer Gunicorn.



```

# Stage 1: Builder
FROM python:3.11-slim AS builder

WORKDIR /app

# Zainstaluj wymagane narzędzia systemowe
RUN apt-get update && apt-get install -y \
    build-essential \
    libssl-dev \
    libffi-dev \
    && rm -rf /var/lib/apt/lists/*

COPY requirements.txt .

# Instalacja pakietów Python dla użytkownika
RUN pip install --user --no-cache-dir -r requirements.txt

# Stage 2: Final
FROM python:3.11-slim

WORKDIR /app

COPY --from=builder /root/.local /root/.local
COPY . .

ENV PATH=/root/.local/bin:$PATH
EXPOSE 8000

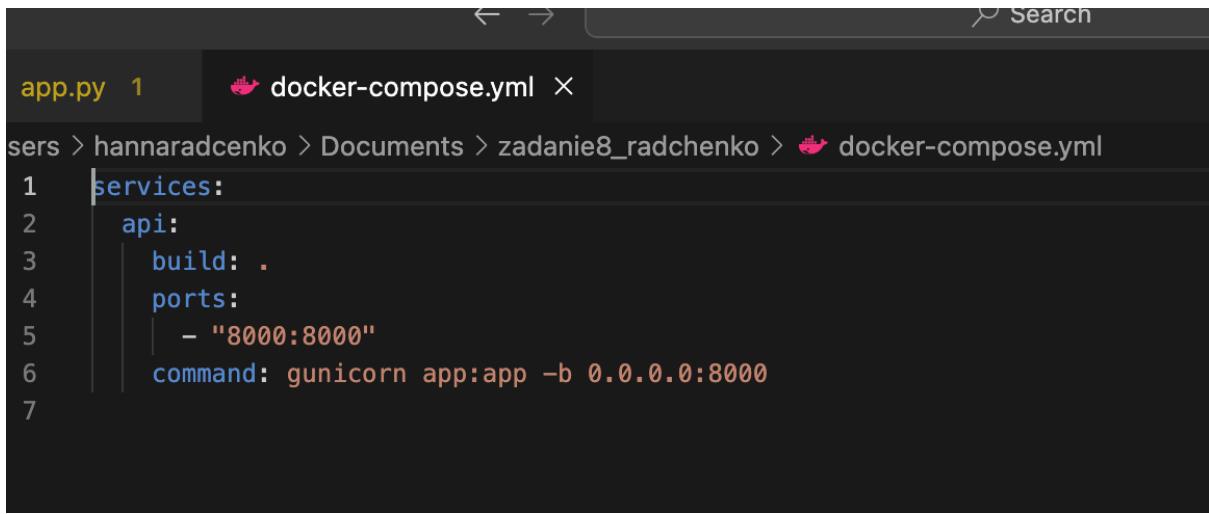
# Uruchomienie produkcyjnego WSGI
CMD ["gunicorn", "-b", "0.0.0.0:8000", "app:app"]

```

5. Plik docker-compose.yml

Plik `docker-compose.yml` definiuje usługę o nazwie `api`.

Określa porty oraz sposób budowania kontenera.



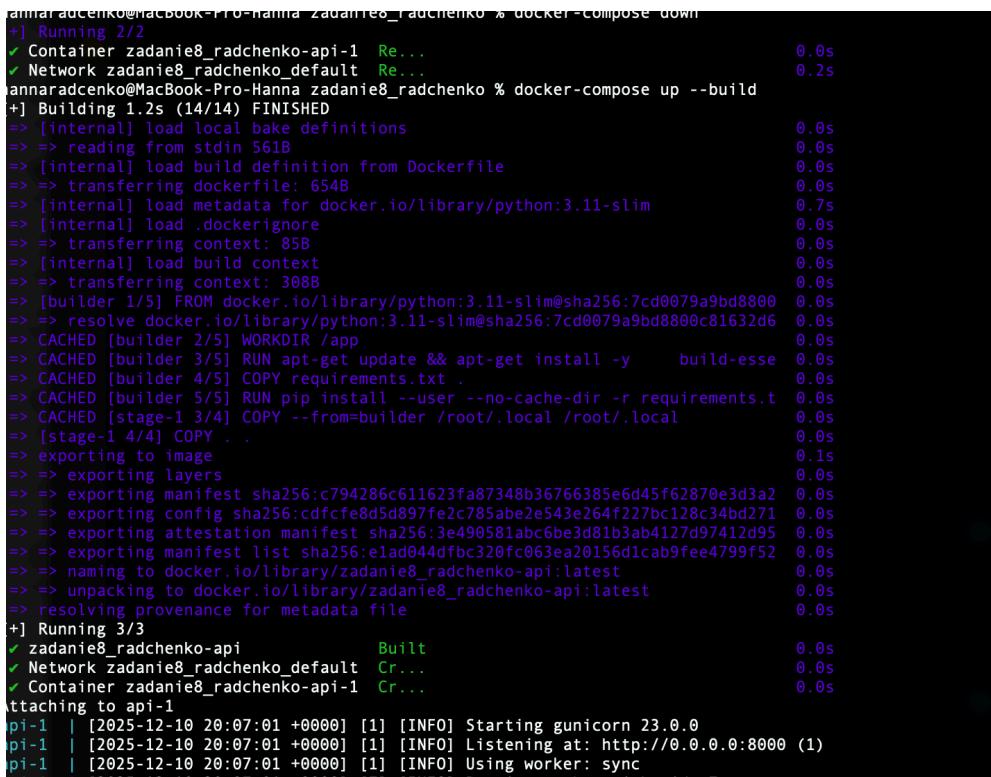
The screenshot shows a code editor window with two tabs: 'app.py' and 'docker-compose.yml'. The 'docker-compose.yml' tab is active, displaying the following YAML configuration:

```
services:
  api:
    build: .
    ports:
      - "8000:8000"
    command: gunicorn app:app -b 0.0.0.0:8000
```

6. Uruchamianie aplikacji

Aby uruchomić aplikację, należy wykonać:

```
docker-compose up --build
```



The terminal output shows the execution of the command 'docker-compose up --build'. It starts by stopping existing containers, then builds the 'api' service, and finally starts the 'api' container. The output includes detailed logs of the build process, such as loading Dockerfiles, resolving dependencies, and copying files into the container.

```
[+] Running 2/2
✓ Container zadanie8_radchenko-api-1  Re...
✓ Network zadanie8_radchenko_default  Re...
hannaradcenko@MacBook-Pro-Hanna_zadanie8_radchenko % docker-compose up --build
[+] Building 1.2s (14/14) FINISHED
=> [internal] load local bake definitions          0.0s
=> => reading from stdin 5618                      0.0s
=> [internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 654B                0.0s
=> [internal] load metadata for docker.io/library/python:3.11-slim 0.7s
=> [internal] load .dockerrcignore                  0.0s
=> => transferring context: 85B                  0.0s
=> [internal] load build context                  0.0s
=> => transferring context: 3088B                 0.0s
=> [builder 1/5] FROM docker.io/library/python:3.11-slim@sha256:7cd0079a9bd8800c81632d6 0.0s
=> => resolve docker.io/library/python:3.11-slim@sha256:7cd0079a9bd8800c81632d6 0.0s
=> CACHED [builder 2/5] WORKDIR /app               0.0s
=> CACHED [builder 3/5] RUN apt-get update && apt-get install -y     build-essential 0.0s
=> CACHED [builder 4/5] COPY requirements.txt .       0.0s
=> CACHED [builder 5/5] RUN pip install --user --no-cache-dir -r requirements.txt 0.0s
=> CACHED [stage-1 3/4] COPY --from=builder /root/.local /root/.local 0.0s
=> [stage-1 4/4] COPY . .                           0.0s
=> exporting to image                            0.1s
=> => exporting layers                          0.0s
=> => exporting manifest sha256:c794286c611623fa87348b36766385e6d45f62870e3d3a2 0.0s
=> => exporting config sha256:cdfcfe8d5d897fe2c785abe2e543e264f227bc128c34bd271 0.0s
=> => exporting attestation manifest sha256:3e490581abc6be3d81b3ab4127d97412d95 0.0s
=> => exporting manifest list sha256:e1ad044dfbc320fc063ea20156d1cab9fee4799f52 0.0s
=> => naming to docker.io/library/zadanie8_radchenko-api:latest 0.0s
=> => unpacking to docker.io/library/zadanie8_radchenko-api:latest 0.0s
=> resolving provenance for metadata file        0.0s
[+] Running 3/3
✓ zadanie8_radchenko-api           Built          0.0s
✓ Network zadanie8_radchenko_default  Created       0.0s
✓ Container zadanie8_radchenko-api-1  Created       0.0s
Attaching to api-1
pi-1 | [2025-12-10 20:07:01 +0000] [1] [INFO] Starting gunicorn 23.0.0
pi-1 | [2025-12-10 20:07:01 +0000] [1] [INFO] Listening at: http://0.0.0.0:8000 (1)
pi-1 | [2025-12-10 20:07:01 +0000] [1] [INFO] Using worker: sync
```

Uruchomienie w tle:

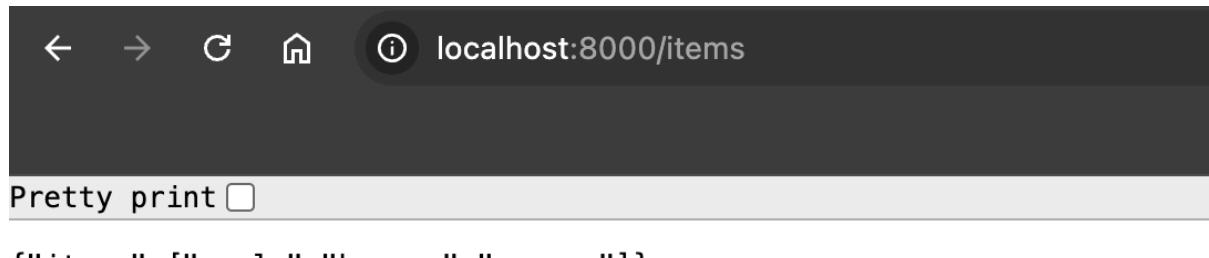
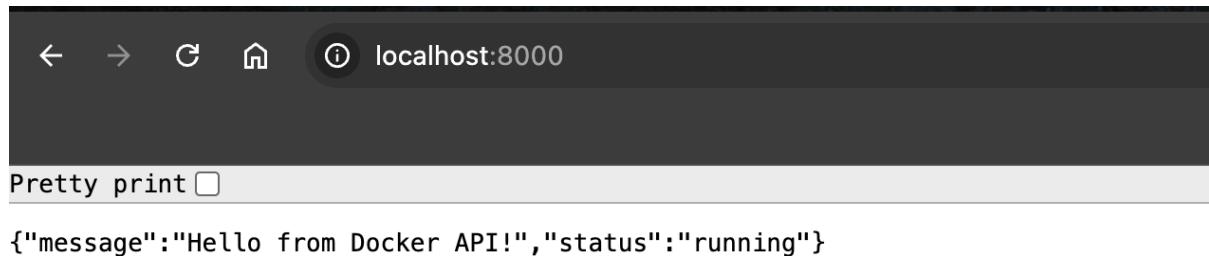
```
docker-compose up -d
```

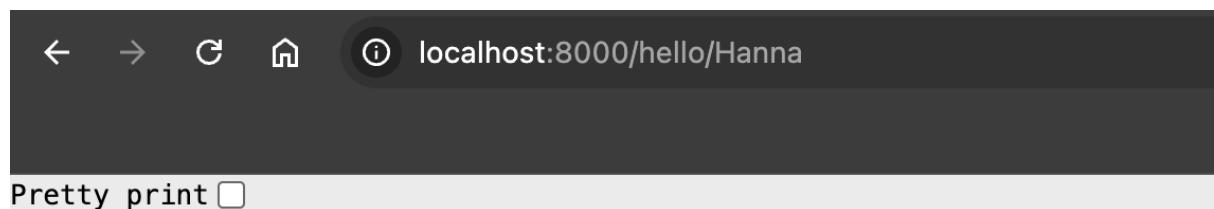
Sprawdzenie działania:

```
curl http://localhost:8000/
```

```
[radchenko@MacBook-Pro-Hanna zadanie8_radchenko % curl http://localhost:8000/
{"message": "Hello from Docker API!", "status": "running"}
[radchenko@MacBook-Pro-Hanna zadanie8_radchenko % curl http://localhost:8000/hello/Hanna
 {"message": "Hello, Hanna!"}
[radchenko@MacBook-Pro-Hanna zadanie8_radchenko % curl http://localhost:8000/items
 {"items": ["apple", "banana", "orange"]}
radchenko@MacBook-Pro-Hanna zadanie8_radchenko % >
```

7. Test połączenia z API





Wszytko działa.