CZĘŚĆ OBOWIĄZKOWA

1. (max. 30%)

Najpierw stworzyłam konto na stronie [Current weather and forecast - OpenWeatherMap]

Potem stworzyłam plik app.py i index.html

Po uruchomieniu aplikacji w konsoli wyświetlają się data, imię oraz port (5000), a także tworzy się strona internetowa, na której można wybrać miasto z listy i zobaczyć pogodę.

Plik: app.py:

```
from flask import Flask, request, render_template
import logging
import datetime
import requests
import os

#Konfiguracja logowania
logging.basicConfig(level=logging.INFO)
logger = logging.getLogger(__name__)

#Inicjalizacja aplikacji Flask
app = Flask(__name__)

AUTHOR = "Yelyzaveta Zlydnieva"
PORT = os.getenv("PORT", 5000)

COUNTRIES_CITIES = {
   "Poland": ["Warszawa", "Krakow", "Gdansk", "Wroclaw",
"Poznan"],
   "Germany": ["Berlin", "Munich", "Hamburg", "Cologne",
"Frankfurt"],
   "France": ["Paris", "Lyon", "Marseille", "Toulouse", "Nice"],
   "Italy": ["Rome", "Milan", "Naples", "Turin", "Florence"]
}

#Logowanie informacji przy starcie aplikacji
startup_time = datetime.datetime.now().strftime("%Y-%m-%d
%H:%M:%S")
logger.info(f"Data uruchomienia: {startup_time}")
logger.info(f"Autor: {AUTHOR}")
logger.info(f"Port TCP: {PORT}")

#Strona główna aplikacji
@app.route("/", methods=["GET", "POST"])
def index():
   weather_data = None
   selected_country = None
   cities = []

   #Jeśli użytkownik wysłał formularz
   if request.method == "POST":
```

```
        selected_country = request.form.get("country")
        city = request.form.get("city")

        #Zaktualizujemy listę miast na podstawie wybranego kraju
        if selected_country in COUNTRIES_CITIES:
            cities = COUNTRIES_CITIES[selected_country]

            #Pobieramy dane pogodowe, jeśli miasto zostało wybrane
            if city and city in cities:
                api_key = "e3752818441fd5ae6c1fb0940dbe8f5d"
                url =
f"http://api.openweathermap.org/data/2.5/weather?q={city},{selecte
d_country}&appid={api_key}&units=metric"
                response = requests.get(url)
                if response.status_code == 200:
                    data = response.json()
                    weather_data = {
                        "city": city,
                        "country": selected_country,
                        "temperature": data["main"]["temp"],
                        "description":
data["weather"][0]["description"],
                        "humidity": data["main"]["humidity"]
                    }

    #Jeśli żaden kraj nie jest wybrany, ustawimy pierwszy z listy
    if not selected_country:
        selected_country = list(COUNTRIES_CITIES.keys())[0]
        cities = COUNTRIES_CITIES[selected_country]


    return render_template("index.html",
countries=COUNTRIES_CITIES.keys(), cities=cities,
selected_country=selected_country, weather_data=weather_data)


if __name__ == "__main__":
    app.run(host="0.0.0.0", port=int(PORT))
```

Plik index.html:

```
<!DOCTYPE html>
<html>
<head>
    <title>Pogoda</title>
</head>
<body>
    <!-- Nagłówek strony -->
```

```
    <h1>Sprawdź pogodę</h1>

    <!-- Formularz do wyboru kraju i miasta -->
    <form method="POST">
              <label for="country">Wybierz kraj:</label>
        <select name="country" id="country"
onchange="this.form.submit()">
            {% for country in countries %}
                <option value="{{ country }}" {% if country ==
selected_country %}selected{% endif %}>{{ country }}</option>
            {% endfor %}
        </select>

        <!-- Wybór miasta -->
        <label for="city">Wybierz miasto:</label>
        <select name="city" id="city">
            {% for city in cities %}
                <option value="{{ city }}">{{ city }}</option>
            {% endfor %}
        </select>

        <!-- Przycisk do zatwierdzenia wyboru -->
        <button type="submit">Sprawdź</button>
    </form>

    <!-- Wyświetlenie pogody, jeśli dane są dostępne -->
    {% if weather_data %}
        <h2>Pogoda w {{ weather_data.city }}, {{
weather_data.country }}</h2>
        <p>Temperatura: {{ weather_data.temperature }}°C</p>
        <p>Opis: {{ weather_data.description }}</p>
        <p>Wilgotność: {{ weather_data.humidity }}%</p>
    {% endif %}
</body>
</html>
```

```
PS C:\Users\Who\OneDrive\Рабочий стол\weather_app> docker logs
weather-container
>>
INFO:__main__:Data uruchomienia: 2025-04-26 10:28:57
INFO:__main__:Autor: Yelyzaveta Zlydnieva
INFO:__main__:Port TCP: 5000
```

2. (max. 50%)

Stworzymy plik Dockerfile

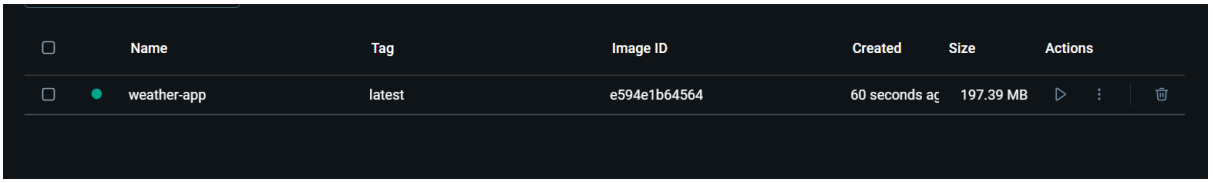
```
# Budowanie środowiska
FROM python:3.12-slim as builder
WORKDIR /app
COPY requirements.txt .

# Instalujemy zależności do katalogu lokalnego
RUN pip install --no-cache-dir --prefix=/install -r
requirements.txt

# Tworzenie finalnego obrazu
FROM python:3.12-slim


LABEL maintainer="Yelyzaveta Zlydnieva"

WORKDIR /app

COPY app.py .
COPY templates/ templates/

# Kopiujemy zainstalowane biblioteki z etapu buildera
COPY --from=builder /install /usr/local
ENV PORT=5000
EXPOSE 5000

# Healthcheck, czy aplikacja działa
HEALTHCHECK --interval=30s --timeout=5s --start-period=5s
--retries=3 \
CMD curl --fail http://localhost:5000 || exit 1

# Startujemy aplikację
CMD ["python", "app.py"]
```

3. (max. 20%)

Piszemy polecenie do utworzenia obrazu:

PS C:\Users\Who\OneDrive\Рабочий стол\weather_app> docker build -t weather-app .

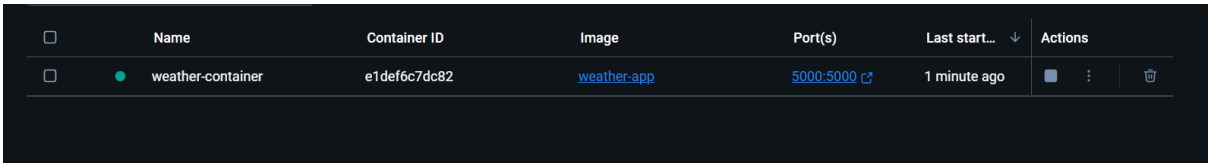
```
PS C:\Users\Who\OneDrive\Рабочий стол\weather_app> docker build -t weather-app .
>>
[+] Building 6.9s (13/13) FINISHED                          docker:desktop-linux
 => [internal] load build definition from Dockerfile                        0.0s
 => => transferring dockerfile: 950B                                        0.0s
 => WARN: FromAsCasing: 'as' and 'FROM' keywords' casing do not match (     0.0s
 => [internal] load metadata for docker.io/library/python:3.12-slim         1.1s
 => [auth] library/python:pull token for registry-1.docker.io               0.0s
 => [internal] load .dockerignore                                           0.0s
 => => transferring context: 2B                                             0.0s
 => [builder 1/4] FROM docker.io/library/python:3.12-slim@sha256:858243     0.0s
 => => resolve docker.io/library/python:3.12-slim@sha256:85824326bc4ae2     0.0s
 => [internal] load build context                                           0.0s
 => => transferring context: 132B                                           0.0s
 => [builder 3/4] COPY requirements.txt .                                   0.0s
 => [stage-1 3/5] COPY app.py .                                             0.0s
 => [builder 4/4] RUN pip install --no-cache-dir --prefix=/install -r r     4.8s
 => [stage-1 4/5] COPY templates/ templates/                                0.0s
 => [stage-1 5/5] COPY --from=builder /install /usr/local                   0.1s
 => exporting to image                                                      0.5s
 => => exporting layers                                                     0.3s
 => => exporting manifest sha256:df259f8a0b7e541566648ec93556862492849e     0.0s
 => => exporting config sha256:56860112e1ac3c2f18086e982bf4d709256a55e9     0.0s
 => => exporting attestation manifest sha256:45d3121de6cfc35f5f4e50baef     0.0s
 => => exporting manifest list sha256:e594e1b64564706945458f98e48bfa6fb     0.0s
```



Polecenie do Uruchomienia kontenera:

PS C:\Users\Who\OneDrive\Рабочий стол\weather_app> docker run -d -p 5000:5000 --name weather-container weather-app

```
PS C:\Users\Who\OneDrive\Рабочий стол\weather_app> docker run -d -p 5000:5000 --name weather-container weather-app
>> C:\Users\Who\OneDrive\Рабочий стол\weather_app>
e1def6c7dc82e7b32d4f2d75c7af018c5a76bde72fef6798bcc54116b9a0860a
PS C:\Users\Who\OneDrive\Рабочий стол\weather_app>
```



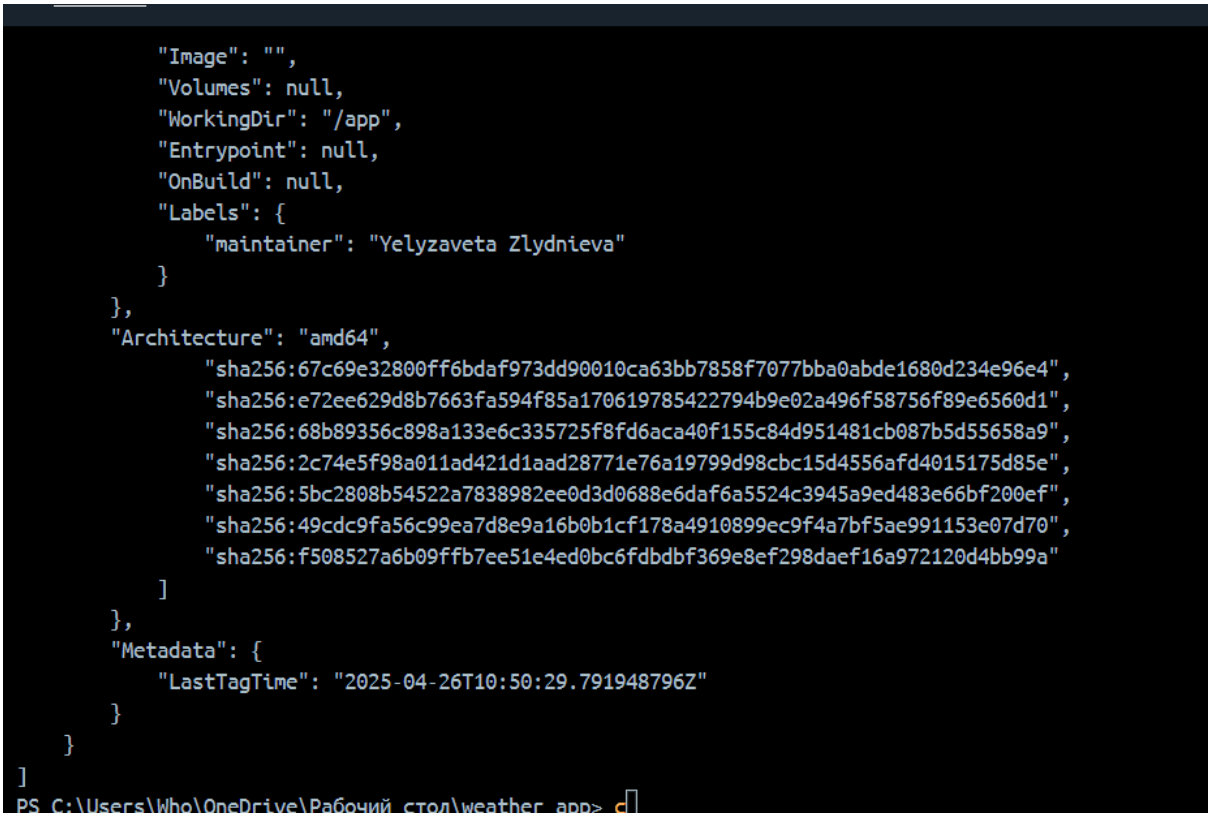
Sprawdzenie logów:

```
PS C:\Users\Who\OneDrive\Рабочий стол\weather_app> docker logs weather-container
>> C:\Users\Who\OneDrive\Рабочий стол\weather_app>
INFO:__main__:Data uruchomienia: 2025-04-26 10:51:29
INFO:__main__:Autor: Yelyzaveta Zlydnieva
INFO:__main__:Port TCP: 5000
 * Serving Flask app 'app'
 * Debug mode: off
INFO:werkzeug: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
INFO:werkzeug:Press CTRL+C to quit
PS C:\Users\Who\OneDrive\Рабочий стол\weather_app>
```

Sprawdzamy liczby warstw i rozmiar:
Mam 7 warstw

```
PS C:\Users\Who\OneDrive\Рабочий стол\weather_app> docker image inspect weather-app
>>
[
    {
        "Id": "sha256:e594e1b64564706945458f98e48bfa6fbaba0c5bc1b2ddf2ca3766e5e11663c1",
        "RepoTags": [
            "weather-app:latest"
        ],
        "RepoDigests": [
            "weather-app@sha256:e594e1b64564706945458f98e48bfa6fbaba0c5bc1b2ddf2ca3766e5e11663c1"
        ],
        "Parent": "",
        "Comment": "buildkit.dockerfile.v0",
        "Created": "2025-04-26T10:50:29.380605401Z",
        "DockerVersion": "27.5.1",
        "Author": "",
        "Config": {
            "Hostname": "",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "ExposedPorts": {
                "5000/tcp": {}
```

```
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
                "PATH=/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
                "LANG=C.UTF-8",
                "GPG_KEY=7169605F62C751356D054A26A821E680E5FA6305",
                "PYTHON_VERSION=3.12.10",
                "PYTHON_SHA256=07ab697474595e06f06647417d3c7fa97ded07afc1a7e4454c5639919b46eaea",
                "PORT=5000"
            ],
            "Cmd": [
                "python",
                "app.py"
            ],
            "Healthcheck": {
                "Test": [
                    "CMD-SHELL",
                    "curl --fail http://localhost:5000 || exit 1"
                ],
                "Interval": 30000000000,
                "Timeout": 5000000000,
                "StartPeriod": 5000000000,
                "Retries": 3
            },
```


```
            "Image": "",
            "Volumes": null,
            "WorkingDir": "/app",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": {
                "maintainer": "Yelyzaveta Zlydnieva"
            }
        },
        "Architecture": "amd64",
                "sha256:67c69e32800ff6bdaf973dd90010ca63bb7858f7077bba0abde1680d234e96e4",
                "sha256:e72ee629d8b7663fa594f85a170619785422794b9e02a496f58756f89e6560d1",
                "sha256:68b89356c898a133e6c335725f8fd6aca40f155c84d951481cb087b5d55658a9",
                "sha256:2c74e5f98a011ad421d1aad28771e76a19799d98cbc15d4556afd4015175d85e",
                "sha256:5bc2808b54522a7838982ee0d3d0688e6daf6a5524c3945a9ed483e66bf200ef",
                "sha256:49cdc9fa56c99ea7d8e9a16b0b1cf178a4910899ec9f4a7bf5ae991153e07d70",
                "sha256:f508527a6b09ffb7ee51e4ed0bc6fdbdbf369e8ef298daef16a972120d4bb99a"
            ]
        },
        "Metadata": {
            "LastTagTime": "2025-04-26T10:50:29.791948796Z"
        }
    }
]
PS C:\Users\Who\OneDrive\Рабочий стол\weather_app> d
```


```
PS C:\Users\Who\OneDrive\Рабочий стол\weather_app> docker images
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
weather-app   latest    e594e1b64564   47 hours ago    197MB
PS C:\Users\Who\OneDrive\Рабочий стол\weather_app>
```

Działanie aplikacji:

Sprawdź pogodę

Wybierz kraj: Polska Wybierz miasto: Warszawa Sprawdź

Pogoda w Lublin, Polska

Temperatura: 12.36°C

Opis: overcast clouds

Wilgotność: 71%