



**POLITECHNIKA LUBELSKA
WYDZIAŁ ELEKTROTECHNIKI
I INFORMATYKI**

**KIERUNEK STUDIÓW
INFORMATYKA**

Przedmiot: Programowanie Aplikacji w Chmurze Obliczeniowej

Sprawozdanie – Zadanie 2 (Obowiązkowe)

Autorzy:
Svitlana Lusiuk, 97823

Lublin, 2025

W 2 zadaniu wykorzystałam pliki, które powstały w wyniku realizacji zadania nr 1 – [index.php](#), [data.php](#), [entrypoint.sh](#), [Dockerfile](#), [style.css](#) i katalog [logs](#).

Cel: zautomatyzowanie procesu budowania obrazu kontenera w GitHub Actions z uwzględnieniem:

- obsługi dwóch architektur (linux/amd64 i linux/arm64)
- wykorzystania cache z rejestrem DockerHub w trybie max
- przesyłania obrazu do ghcr.io tylko po pozytywnym przejściu testu bezpieczeństwa (CVE)
- opcjonalnego podpisu obrazu z wykorzystaniem cosign dla większego bezpieczeństwa łańcucha dostaw

1. Konfiguracja środowiska

- Zainstalowałam wymagane narzędzia:

```
sudo apt update
sudo apt install docker.io docker-buildx git gh curl jq -y
curl -sL https://raw.githubusercontent.com/aquasecurity/trivy/main/contrib/install.sh | sh -s -- -b /usr/local/bin
curl -sSL https://github.com/sigstore/cosign/releases/latest/download/cosign-linux-amd64 -o cosign
chmod +x cosign && sudo mv cosign /usr/local/bin
```

- Utworzyłam lokalny katalog projektu i repozytorium git

```
cd weather-app
```

- ! dodatkowo nadałam pełne prawa dostępu do folderu (czytanie, pisanie, wykonanie):

```
chmod 777 logs
```

```
git init
```

```
git config user.name "linkency"
```

```
git config user.email "svitllanalysiuk@gmail.com"
```

- Połączyłam repozytorium lokalne z publicznym repozytorium na GitHub

```
gh auth login
```

```
linkency@DESKTOP-4B4J9VK:~/weather-app$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations on this host? HTTPS
? Authenticate Git with your GitHub credentials? Yes
? How would you like to authenticate GitHub CLI? Paste an authentication token
Tip: you can generate a Personal Access Token here https://github.com/settings/tokens
The minimum required scopes are 'repo', 'read:org', 'workflow'.
? Paste your authentication token: *****
- gh config set -h github.com git_protocol https
✓ Configured git protocol
! Authentication credentials saved in plain text
✓ Logged in as linkency
```

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input checked="" type="checkbox"/> workflow	Update GitHub Action workflows
<input checked="" type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input checked="" type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input type="checkbox"/> admin:org	Full control of orgs and teams, read and write org projects
<input type="checkbox"/> write:org	Read and write org and team membership, read and write org projects
<input checked="" type="checkbox"/> read:org	Read org and team membership, read org projects
<input type="checkbox"/> manage_runners:org	Manage org runners and runner groups

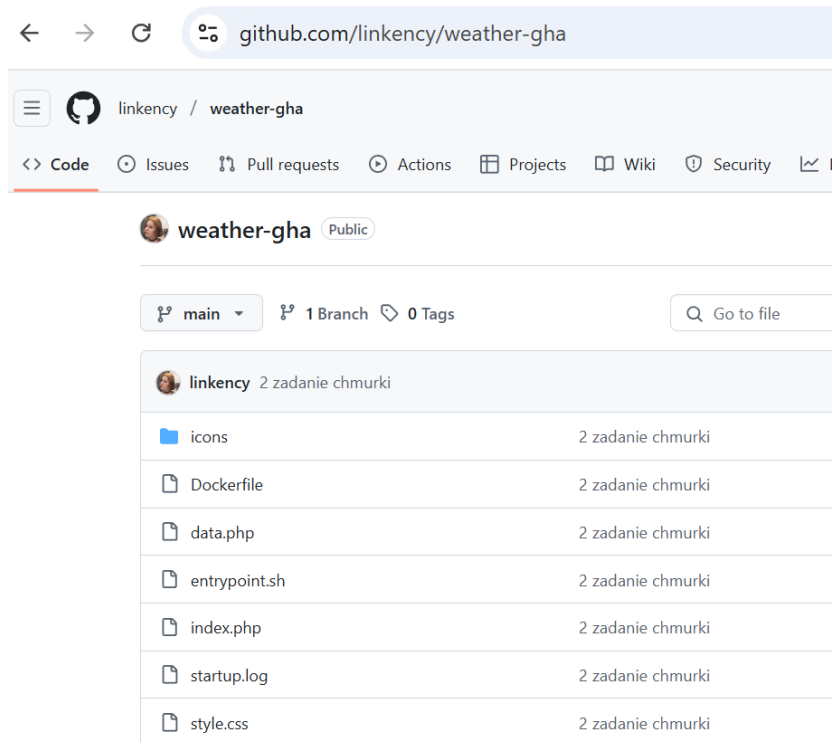
git add .

git commit -m "2 zadanie chmurki"

git branch -M main

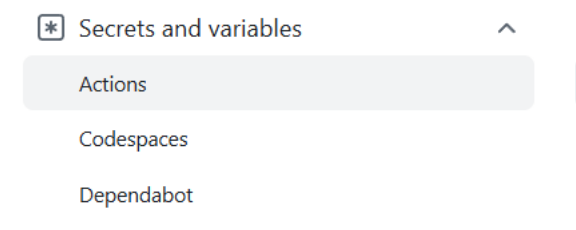
git remote add origin <https://github.com/linkency/weather-gha.git>

git push -u origin main



2. Przygotowanie tokenów i sekretów

- Wygenerowałam tokeny PAT: GHCR_TOKEN, DOCKERHUB_TOKEN
- Weszłam na: <https://github.com/settings/tokens>
- Utworzyłam token z uprawnieniami: write:packages, read:packages, workflow, repo
- Dodałam je do repozytorium w GitHub w sekcji Secrets jako: GHCR_USERNAME, GHCR_TOKEN, DOCKERHUB_TOKEN
- Settings/Secrets and variables/Actions












GHCR_USERNAME = linkency

GHCR_TOKEN = ***

DOCKERHUB_TOKEN = ***

Repository secrets

[New repository secret](#)

Name ↕	Last updated		
 DOCKERHUB_TOKEN	now		
 GHCR_TOKEN	33 minutes ago		
 GHCR_USERNAME	32 minutes ago		

3. Dostosowanie Dockerfile

Dostosowałam plik Dockerfile, aby:

- wykorzystywał minimalną ilość warstw (multi-stage),
- miał ustawiony bardziej dokładny LABEL z danymi autora zgodnie z OCI
- podmieniłam stary plik poleceniem `explorer.exe`.

```
Dockerfile x
1 #Obraz końcowy
2 FROM php:8.2-cli
3
4 # Dane autora zgodnie z OCI
5 LABEL org.opencontainers.image.title="Weather App" \
6       org.opencontainers.image.description="Prosta aplikacja pogodowa w PHP" \
7       org.opencontainers.image.authors="Svitlana Lysiuk <svitllanalsiuk@gmail.com>" \
8       org.opencontainers.image.version="1.0"
9
10 # Utworzenie katalogu aplikacji
11 WORKDIR /app
12
13 # Skopiowanie plików aplikacji
14 COPY index.php .
15 COPY data.php .
16 COPY entrypoint.sh .
17 COPY style.css .
18 COPY logs/ ./logs/
19
20 # Nadanie praw do skryptu uruchamiającego
21 RUN chmod +x entrypoint.sh && chmod 777 logs
22
23 # Otworzenie portu aplikacji
24 EXPOSE 8000
25
26 # Sprawdzenie zdrowia kontenera
27 HEALTHCHECK --interval=30s --timeout=5s --start-period=10s --retries=3 \
28   CMD curl --fail http://localhost:8000 || exit 1
29
30 # Polecenie startowe
31 ENTRYPOINT ["/entrypoint.sh"]
```

4. Budowa i test lokalny kontenera

```
- Zbudowałam obraz: docker buildx build --platform linux/amd64,linux/arm64 -t linkency/weather-app:latest .
```

```

[+] Building 44.6s (22/22) FINISHED
[+] [internal] load build definition from Dockerfile 0.0s
[+] => transferring dockerfile: 675B 0.0s
[+] [Linux/arm64 internal] load metadata for docker.io/library/php:8.2-cli 27.0s
[+] [Linux/arm64 internal] load metadata for docker.io/library/php:8.2-cli 27.2s
[+] [auth] Library/php:pull token for registry-1.docker.io 0.0s
[+] [internal] load definitions for docker.io/library/php:8.2-cli 0.0s
[+] => transferring context: 2B 0.0s
[+] [internal] load build context 0.1s
[+] => transferring context: 340B 4.4s
[+] [Linux/arm64 1/8] FROM docker.io/library/php:8.2-cli@sha256:f5a8ac239966729b239ea1f25039af1afeedd84c5005fd4f 16.5s
[+] => resolve docker.io/library/php:8.2-cli@sha256:f5a8ac239966729b239ea1f25039af1afeedd84c5005fd4f@F80e3dc959663 0.0s
[+] sha256:0571f4e9e2b0f9d6137793bb8dc3fd4a4e67c3f48917902cc739da0b4c1 246B / 2.4kB 0.2s
[+] sha256:1e70dc6c4b9bcaae33b3c5cf2cfac6f48c83300a13840 2.4kB / 2.4kB 0.2s
[+] sha256:ffc09f97d07d676389f3d8301035223c1c88089a2c5c5a0ad3d3599e2 35.68MB / 35.68MB 0.2s
[+] sha256:1382c79ac7aeb3046825da3421995549e99ac52c40652513ed08d04660e9b 488B / 488B 0.2s
[+] sha256:1867c97ab2a5c3669131d0b797193d0c3d66e9a68b 12.26MB / 12.26MB 0.2s
[+] sha256:68834d1bb8b67fae24c0a998a698a339da79238c122bdc12a0345e69976 226B / 226B 0.2s
[+] sha256:83d839999e4d23493e2641e8568f38799d3d35c67c5389ac6f7718b4b5349 194.33MB / 194.33MB 7.4s
[+] sha256:3023b130a2f1d1b8a1f0d26c99b61c0cf0b31384f9d 220B / 220B 0.2s
[+] extracting sha256:3023b130a2f1d1b8a1f0d26c99b61c0cf0b31384f9d 0.4s
[+] extracting sha256:83d839999e4d23493e2641e8568f38799d3d35c67c5389ac6f7718b4b5349 0.4s
[+] extracting sha256:68834d1bb8b67fae24c0a998a698a339da79238c122bdc12a0345e69976 4.4s
[+] extracting sha256:1867c97ab2a5c3669131d0b797193d0c3d66e9a68b 4.4s
[+] extracting sha256:1382c79ac7aeb3046825da3421995549e99ac52c40652513ed08d04660e9b 4.4s
[+] extracting sha256:ffc09f97d07d676389f3d8301035223c1c88089a2c5c5a0ad3d3599e2 1.1s
[+] extracting sha256:1e70dc6c4b9bcaae33b3c5cf2cfac6f48c83300a13840 0.0s
[+] extracting sha256:0571f4e9e2b0f9d6137793bb8dc3fd4a4e67c3f48917902cc739da0b4c1 0.0s
[+] [Linux/arm64 1/8] FROM docker.io/library/php:8.2-cli@sha256:f5a8ac239966729b239ea1f25039af1afeedd84c5005fd4f 6.8s
[+] => resolve docker.io/library/php:8.2-cli@sha256:f5a8ac239966729b239ea1f25039af1afeedd84c5005fd4f@F80e3dc959663 0.0s
[+] sha256:1ca197f784133e79f248b26c31c222da2e3a7713ec4c0d1bb6b 246B / 246B 0.2s
[+] sha256:48bdc0c7161675107b16f33d82b9a2f487f8b53a247230a2657f6d0782a5 35.49MB / 35.49MB 4.7s
[+] sha256:6882ba8a232b7f4b26d56fcd62084217b4b563578a5c9cf77032da8a2 2.4kB / 2.4kB 0.6s
[+] sha256:70a107b18519991e1bb4d7f60aa76523f16b3313e62574530311 12.26MB / 12.26MB 0.2s
[+] sha256:fdcaf65d8a7c7518576808d7fab67c26f40b2812b912d27e7550fa2161df1 488B / 488B 0.3s
[+] extracting sha256:70a107b18519991e1bb4d7f60aa76523f16b3313e62574530311 0.2s
[+] extracting sha256:fdcaf65d8a7c7518576808d7fab67c26f40b2812b912d27e7550fa2161df1 0.2s
[+] extracting sha256:6882ba8a232b7f4b26d56fcd62084217b4b563578a5c9cf77032da8a2 0.0s
[+] extracting sha256:48bdc0c7161675107b16f33d82b9a2f487f8b53a247230a2657f6d0782a5 0.0s
[+] extracting sha256:1ca197f784133e79f248b26c31c222da2e3a7713ec4c0d1bb6b 0.0s
[+] [Linux/arm64 2/8] WORKDIR /app 0.5s
[+] [Linux/arm64 3/8] COPY index.php . 0.2s
[+] [Linux/arm64 4/8] COPY data.php . 0.1s
[+] [Linux/arm64 5/8] COPY entrypoint.sh . 0.1s
[+] [Linux/arm64 6/8] COPY style.css . 0.1s
[+] [Linux/arm64 7/8] COPY logs /logs 0.1s
[+] [Linux/arm64 8/8] RUN chmod +x entrypoint.sh && chmod 777 logs 0.1s
[+] [Linux/arm64 2/8] WORKDIR /app 0.5s
[+] [Linux/arm64 3/8] COPY index.php . 0.1s
[+] [Linux/arm64 4/8] COPY data.php . 0.1s
[+] [Linux/arm64 5/8] COPY entrypoint.sh . 0.1s
[+] [Linux/arm64 6/8] COPY style.css . 0.1s
[+] [Linux/arm64 7/8] COPY logs /logs 0.1s
[+] [Linux/arm64 8/8] RUN chmod +x entrypoint.sh && chmod 777 logs 0.2s
WARNING: No output specified with DockerContainer driver. Build result will only remain in the build cache. To push result image into registry use --push or to load image into docker use --load

```

- Uruchomiłam kontener:

```
docker run -d -p 8000:8000 --name weather linkency/weather-app:latest
```

```
linkency@DESKTOP-4B4J9VK:~/weather-app$ docker run -d -p 8000:8000 --name weather linkency/weather-app:latest
3f0ef1cdaf4c4cfccff9fdb42895cdcbcb5a373a20324c73dac9418d81a9ed86f7
```

- Pobranie logów:

```
docker cp weather:/app/logs/startup.log . && cat startup.log
```

```
linkency@DESKTOP-4B4J9VK:~/weather-app$ docker cp weather:/app/logs/startup.log . && cat startup.log
Successfully copied 2.05kB to /home/linkency/weather-app/.
[Fri May 30 08:54:32 UTC 2025] Autor: Svitlana Lysiuk | Port: 8000
```

- Sprawdzenie obrazu:

```
docker image inspect linkency/weather-app:latest --format='{{.RootFS.Layers}}'
```

docker image ls

```
linkency@DESKTOP-4B4J9VK:~/weather-app$ docker image inspect linkency/weather-app:latest --format '{{.RootFS.Layers}}'
[[{"sha256:ace34d1d784c01e3f9d156687089e8f58f786e23ccd097bdfb337d6d28b3783", "sha256:c18fda44004a200996cfb160040b2aa26b945dd4de61668577abc6588dbe05f", "sha256:8379147501603beebba3460c6ad150a8093f38a9c4242d219c88f49ed90b7309", "sha256:d7cfd0e643e7c65fa491728b16e97fd830153edbcde82216dcb7c66acd3feb2d", "sha256:6d1bc4129b3d84d919ffacd4d0b44e225c91a06d55d548b8228e69bf249edaa", "sha256:41fdbf75b7cf936a3bab6a560d21cc0deda5efc77071ca6e6b419a41dc4eb092", "sha256:2e3f464e9b6d9c4597386aa09282a5aab264e09ae1527ceac9c57e9c2b9c5eb7", "sha256:5fd6430964617a0c85a7f8b281cc94d6e93658600909488c1fb7c61ee30284eb", "sha256:db0133fa89a3e9fc9117d38f6839eb49a690531064bf214437d8417b54c0c494", "sha256:2d414d5e6c31de19e94cde67a073ac978bb49a52eb860226c2d9c7b93cccfcd", "sha256:ed40e4e359979c54469a7b14d30aebb006dbabd1d9f102418879f8a246a3b0ae"]}]]
linkency@DESKTOP-4B4J9VK:~/weather-app$ docker image ls
REPOSITORY          TAG         IMAGE ID      CREATED       SIZE
linkency/weather-app latest      f55342d70fe0 12 hours ago 908MB
lab9-web             latest     421deb80a4d3 5 days ago   1.61GB
svitlana/weather-app latest     05afe90dee90 2 weeks ago  139MB
mongo               latest     9f67b6bafda0 4 weeks ago  1.19GB
moby/buildkit        buildx-stable-1 87afb62ed6a7 4 weeks ago  307MB
nginx                latest     fb39280b7b9e 6 weeks ago  279MB
node                 18        c6ae79e38498 2 months ago 1.57GB
```

5. Konfiguracja CI/CD w GitHub Actions

- Stworzyłam plik `.github/workflows/docker-ci.yml`, w którym zrealizowałam następujące kroki:

- checkout kodu,
- konfigurację buildera (docker/setup-buildx-action) i emulatora (qemu),
- logowanie do DockerHub i GHCR (docker/login-action),
- budowanie wieloplatformowe z cache'em:

`--cache-from=type=registry,ref=docker.io/linkency/weather-cache:latest`

`--cache-to=type=registry,ref=docker.io/linkency/weather-cache:latest,mode=max`

- test CVE przez Trivy (opis w punkcie 7),
- warunkowe publikowanie obrazu,
- podpis obrazu przez cosign.



Fundusze Europejskie
Wiedza Edukacja Rozwój



Rzeczpospolita
Polska

Unia Europejska
Europejski Fundusz Społeczny



```
.git\...\docker-ci.yml x
1 Build and Push Docker Image
2
3 on:
4   push:
5     branches: [main]
6
7 jobs:
8   build:
9     runs-on: ubuntu-latest
10
11   steps:
12     - name: Checkout repository
13       uses: actions/checkout@v3
14
15     - name: Set up Docker Buildx
16       uses: docker/setup-buildx-action@v2
17
18     - name: Log in to DockerHub (for caching)
19       uses: docker/login-action@v2
20       with:
21         username: ${ secrets.GHCR_USERNAME }
22         password: ${ secrets.DOCKERHUB_TOKEN }
23
24     - name: Log in to GitHub Container Registry
25       uses: docker/login-action@v2
26       with:
27         registry: ghcr.io
28         username: ${ secrets.GHCR_USERNAME }
29         password: ${ secrets.GHCR_TOKEN }
30
31     - name: Cache from DockerHub
32       uses: actions/cache@v3
33       with:
34         path: /tmp/.buildx-cache
```

```
35   key: ${ runner.os }}-buildx-${ github.sha }}
36   restore-keys: |
37     ${ runner.os }}-buildx-
38
39   - name: Build image with cache and multiarch
40     run: |
41       docker buildx build \
42         --platform linux/amd64,linux/arm64 \
43         --cache-from=type=registry,ref=docker.io/${ secrets.GHCR_USERNAME }}/weather-cache:cache \
44         --cache-to=type=registry,ref=docker.io/${ secrets.GHCR_USERNAME }}/weather-cache:cache,mode=max \
45         -t ghcr.io/${ secrets.GHCR_USERNAME }}/weather-app:latest \
46         --push .
47
48   - name: Install Trivy
49     run: |
50       curl -sL https://raw.githubusercontent.com/aquasecurity/trivy/main/contrib/install.sh | sh -s -- -b /usr/local/bin
51
52   - name: Scan image with Trivy (fail on HIGH or CRITICAL)
53     run: |
54       trivy image --exit-code 1 --severity HIGH,CRITICAL ghcr.io/${ secrets.GHCR_USERNAME }}/weather-app:latest
55
56   - name: Sign image with cosign using OIDC
57     run: |
58       cosign sign --yes ghcr.io/${ secrets.GHCR_USERNAME }}/weather-app:latest
59     env:
60       COSIGN_EXPERIMENTAL: true
```



Fundusze Europejskie
Wiedza Edukacja Rozwój



Rzeczpospolita
Polska

Unia Europejska
Europejski Fundusz Społeczny



8. Publikacja do GHCR

`docker push ghcr.io/linkency/weather-app:latest`

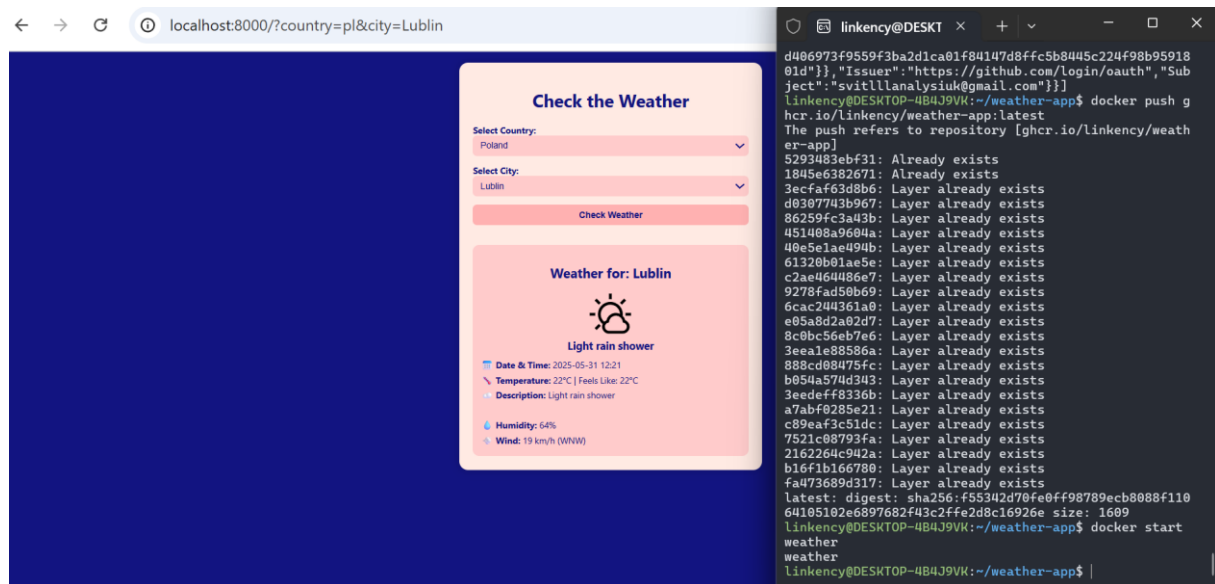
! Obraz trafia do publicznego repozytorium GHCR

```
linkency@DESKTOP-4B4J9VK:~/weather-app$ docker push ghcr.io/linkency/weather-app:latest
The push refers to repository [ghcr.io/linkency/weather-app]
5293483ebf31: Already exists
1845e6382671: Already exists
3ecfaf63d8b6: Layer already exists
d0307743b967: Layer already exists
86259fc3a43b: Layer already exists
451408a9604a: Layer already exists
40e5e1ae494b: Layer already exists
61320b01ae5e: Layer already exists
c2ae464486e7: Layer already exists
9278fad50b69: Layer already exists
6cac244361a0: Layer already exists
e05a8d2a02d7: Layer already exists
8c0bc56eb7e6: Layer already exists
3eea1e88586a: Layer already exists
888cd08475fc: Layer already exists
b054a574d343: Layer already exists
3eedeff8336b: Layer already exists
a7abf0285e21: Layer already exists
c89eaf3c51dc: Layer already exists
7521c08793fa: Layer already exists
2162264c942a: Layer already exists
b16f1b166780: Layer already exists
fa473689d317: Layer already exists
latest: digest: sha256:f55342d70fe0ff98789ecb8088f11064105102e6897682f43c2ffe2d8c16926e size: 1609
```

9. Test działania w przeglądarce

docker start weather

- Otworzyłam <http://localhost:8000/?country=pl&city=Lublin> i przetestowałam aplikację



Fundusze Europejskie
Wiedza Edukacja Rozwój



Rzeczpospolita
Polska

Unia Europejska
Europejski Fundusz Społeczny





Materiały zostały opracowane w ramach projektu
„*Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga*”,
umowa nr **POWR.000000Z060/1800**
w ramach Programu Operacyjnego Wiedza Edukacja Rozwój 20142020
współfinansowanego ze środków Europejskiego Funduszu Społecznego