



POLITECHNIKA
LUBELSKA
WYDZIAŁ ELEKTROTECHNIKI
I INFORMATYKI

Programowanie aplikacji w chmurze obliczeniowej

Autor: Adrian Sak vel Antoszak

grupa dziekańska: 6.8
numer albumu: 97727

Treść pliku Dockerfile:

```
# etap 1
# użycie obrazu bazowego metoda od podstaw
FROM scratch AS build

# zadeklarowanie zmiennej VERSION używanej w procesie budowania
ARG VERSION

# użycie obrazu bazowego z minimalnym systemem plików linuxa alpine
ADD alpine-minirootfs-3.19.1-x86_64.tar /

# określenie katalogu w którym będzie budowana aplikacja js
WORKDIR /usr/app

# aktualizacja pakietów, instalacja node i wyczyszczenia pamięci cache
RUN apk update && \
    apk add nodejs npm && \
    rm -rf /var/cache/apk/*

# przekopiowanie plików z używanymi bibliotekami js i ich instalacja
COPY ./package.json ./
RUN npm install

# skopiowanie plików aplikacji js do kontenera
COPY ./app.js ./

# etap 2
# kolejny etap budowy obrazu
# wykorzystanie obrazu alpine z nginx
# alpine posiada manager pakietów który pozwoli na instalację node,
# zwykły nginx nie posiada menadżera pakietów
FROM nginx:alpine

# przechwycenie zmiennej środowiskowej zadeklarowanej w poprzednim
# etapie budowy i zapisanie jej w zmiennej środowiskowej
ARG VERSION
ENV APP_VERSION=${VERSION:-v1}

# aktualizacja/instalacja node/czyszczenia cache.
RUN apk update && \
    apk add nodejs npm && \
    rm -rf /var/cache/apk/*

# skopiowanie plików z poprzedniego etapu do obrazu do obcego
```

```

COPY --from=build /usr/app /usr/share/nginx/html
# skopiowanie pliku konfiguracyjnego nginx
COPY ./default.conf /etc/nginx/conf.d

WORKDIR /usr/share/nginx/html

# wystawienie portu 80, na którym domyślnie działa nginx
EXPOSE 80

# ustawienie komendy sprawdzającej czy aplikacja działa poprawnie
HEALTHCHECK --interval=20s --timeout=3s --start-period=5s --retries=2 \
    CMD curl -f http://localhost:80 || exit 1

# uruchomienia serwera nginx i aplikacji js wewnątrz kontenera
CMD nginx -g "daemon off;" & node app.js

```

Polecenie do budowy obrazu oraz wynik jego działania:

```

[monio359@fedora Lab5]$ docker build --build-arg VERSION=1.0.0 -t zadanie5 .
[+] Building 10.7s (16/16) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 1.87kB
=> [internal] load metadata for docker.io/library/nginx:alpine
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load build context
=> => transferring context: 591B
=> CACHED [stage-1 1/5] FROM docker.io/library/nginx:alpine@sha256:31bad00311cb5eeb8a6648beadcf6727a175da89989f14727420a80e2e76742
=> [stage-1 2/5] RUN apk update 66      apk add nodejs npm 66      rm -rf /var/cache/apk/*
=> CACHED [build 1/6] ADD alpine-minirooftfs-3.19.1-x86_64.tar /
=> CACHED [build 2/6] WORKDIR /usr/app
=> [build 3/6] RUN apk update 66      apk add nodejs npm 66      rm -rf /var/cache/apk/*
=> [build 4/6] COPY ./package.json ./
=> [build 5/6] RUN npm install
=> [build 6/6] COPY ./app.js ./
=> [stage-1 3/5] COPY --from=build /usr/app /usr/share/nginx/html
=> [stage-1 4/5] COPY ./default.conf /etc/nginx/conf.d
=> [stage-1 5/5] WORKDIR /usr/share/nginx/html
=> exporting to image
=> => exporting layers
=> => writing image sha256:f91bdd37acb4bca5bb8151c9763e977d080cdc33056ddffaa97ea46f8e3ee6d
=> => naming to docker.io/library/zadanie5

View build details: docker-desktop://dashboard/build/default/default/xi7a6lp4dt2i8puv45m1054vk

What's Next?
1. Sign in to your Docker account → docker login
2. View a summary of image vulnerabilities and recommendations → docker scout quickview
[monio359@fedora Lab5]$

```

Polecenie uruchamiające kontener:

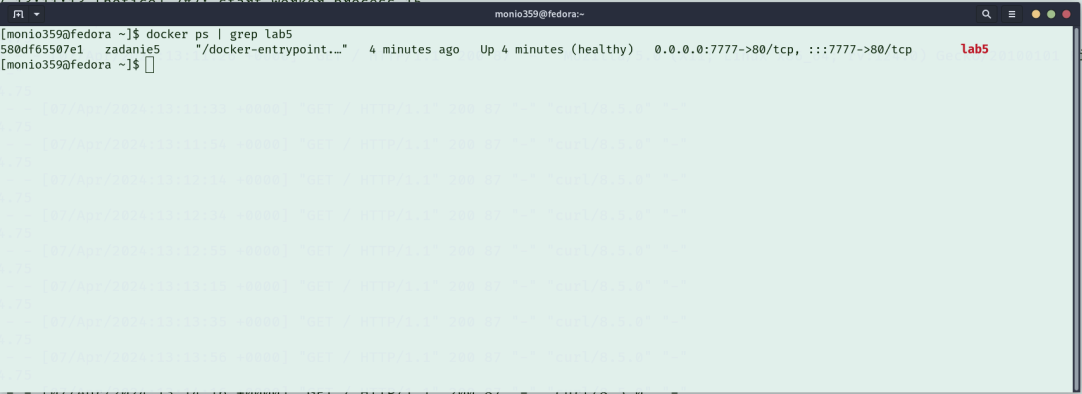
```

[monio359@fedora Lab5]$ docker run -it -p 7777:80 --name lab5 zadanie5
2024/04/07 13:11:13 [notice] 7#7: using the "epoll" event method
2024/04/07 13:11:13 [notice] 7#7: nginx/1.25.4
2024/04/07 13:11:13 [notice] 7#7: built by gcc 12.2.1 20220924 (Alpine 12.2.1_git20220924-r10)
2024/04/07 13:11:13 [notice] 7#7: OS: Linux 6.7.9-100.fc38.x86_64
2024/04/07 13:11:13 [notice] 7#7: getrlimit(RLIMIT_NOFILE): 1073741816:1073741816
2024/04/07 13:11:13 [notice] 7#7: start worker processes
2024/04/07 13:11:13 [notice] 7#7: start worker process 8
2024/04/07 13:11:13 [notice] 7#7: start worker process 9
2024/04/07 13:11:13 [notice] 7#7: start worker process 10
2024/04/07 13:11:13 [notice] 7#7: start worker process 11
2024/04/07 13:11:13 [notice] 7#7: start worker process 12
2024/04/07 13:11:13 [notice] 7#7: start worker process 13
2024/04/07 13:11:13 [notice] 7#7: start worker process 14
2024/04/07 13:11:13 [notice] 7#7: start worker process 15
Aplikacja działa na porcie 8080
77.253.174.75
172.17.0.1 - - [07/Apr/2024:13:11:26 +0000] "GET / HTTP/1.1" 200 87 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:124.0) Gecko/20100101 Firefo
x/124.0" "-"
77.253.174.75
127.0.0.1 - - [07/Apr/2024:13:11:33 +0000] "GET / HTTP/1.1" 200 87 "-" "curl/8.5.0" "-"
]

```

Polecenie sprawdzające czy wszystko w porządku z kontenerem:

```
2024/04/07 13:11:13 [notice] 7#7: start worker process 13
2024/04/07 13:11:13 [notice] 7#7: start worker process 14
2024/04/07 13:11:13 [notice] 7#7: start worker process 15
Applikacja
77.253.174.75 [monio359@fedora ~]$ docker ps | grep lab5
172.17.0.1 580df65507e1 zadanie5 "/docker-entrypoint..." 4 minutes ago Up 4 minutes (healthy) 0.0.0.0:7777->80/tcp Lab5
x/124.0" [monio359@fedora ~]$
```



The screenshot shows a terminal window with a dark background. The prompt is [monio359@fedora ~]. The user has run 'docker ps | grep lab5', which shows a container named 'zadanie5' with ID '580df65507e1' running '"/docker-entrypoint..."' 4 minutes ago, in a 'Up' state, healthy, and listening on '0.0.0.0:7777->80/tcp'. The window title is 'monio359@fedora:~'. To the right, a portion of a web browser window is visible, showing a red 'Lab5' label and the text 'irefo'.

Prezentacja działającej aplikacji z wykorzystaniem programu curl:

```
[monio359@fedora ~]$ curl http://localhost:7777
nazwa hosta: 580df65507e1<br> adres ip2: 77.253.174.75<br> wersja aplikacji: 1.0.0<br>[monio359@fedora ~]$
```