Cloud Computing

Vežbe 2

Primer 1.

Umesto preuzimanja gotove *hello-world* slike kontejnera, potrebno je kreirati svoju sliku koja funkcioniše jednako kao i već gotova slika.

Ograničenja za zadatak:

- Sliku kreirati tako da ne sadrži nikakav sloj operativnog sistema, dodatnih biblioteka, itd.
- Jedino što slika može da sadrži je izvršni fajl.

Pomoć za zadatak:

- U direktorijumu *primer1* postoji već gotov kod napisan u C programskom jeziku koji kada se izvrši daje isti rezultat kao pokretanje kontejnera na osnovu gotove *hello-world* slike.
- C kod se kompajlira sa komandom:
  `gcc hello-world.c -static -o hello-world`

```
Prilikom kompajliranja dodati i opciju -static

Dockerfile

FROM scratch
COPY hello-world /
CMD ["/hello-world"]
```

Primer 2.

Napraviti sliku kontejnera koja nakon pokretanja kontejnera predefinisano ispisuje poruku *"Pozdrav, <ime>!"* na terminal (naredba za ispis je `echo`).

1) upotrebiti *exec* formu za naredbu u Dockerfile-u
2) upotrebiti *shell* formu za naredbu u Dockerfile-u
3) pregaziti predefinisanu poruku novom porukom prilikom pokretanja kontejnera

```
Dockerfile:

FROM ubuntu:latest

#1) CMD [ "echo", "Pozdrav, Petar!"]
#2) CMD echo "Pozdrav, Petar!"
```

Terminal:

3) $ docker run <naziv_slike> echo "Pozdrav, Veljko!"

Primer 3.

Napraviti sliku kontejnera koja nakon pokretanja kontejnera predefinisano ispisuje poruku *"Nalazite se u <naziv_radnog_direktorijuma>"* na terminal (naziv radnog direktorijuma se čuva u `$PWD` environment varijabli).

Ograničenja za zadatak:

1) koristiti *alpine:latest* kao osnovu slike
2) podesiti u slici kontejnera da radni direktorijum za dati kontejner bude *cloud*
3) upotrebiti *exec* formu za naredbu u Dockerfile-u

```
Dockerfile:

FROM alpine:latest
WORKDIR /cloud
CMD [ "sh", "-c", "echo $PWD" ]
```

Primer 4.

Napraviti sliku kontejnera koja predefinisano pokreće python aplikaciju *countdown.py*, a omogućava i pokretanje druge python aplikacije *todo.py* prosleđivanjem SAMO naziva te druge aplikacije prilikom pokretanja

kontejnera.

Pomoć za zadatak:

- Koristiti i ENTRYPOINT u Dockerfile-u.
- Naredba za pokretanje python aplikacije `python countdown.py`

Dockerfile:

```
FROM python:3.9-slim
WORKDIR /app
ENV app=countdown.py
COPY countdown.py .
ENTRYPOINT ["python"]
CMD [$app]
```

Primer 5.

Napisati sliku kontejnera koja pokreće aplikaciju *server.py*.

Dodatno:

- Napraviti optimalnu varijantu slike kontejnera koja neće instalirati biblioteke svaki put kada se promeni kod u *server.py* aplikaciji.
- Izostaviti *README.md* iz build konteksta

Dockerfile:

```
FROM python:3.9-slim
WORKDIR /app
COPY server.py .
RUN pip install flask
ENTRYPOINT ["python"]
CMD ["server.py"]
```

.dockerignore:

```
README.md
```

Primer 6.

Napraviti sliku kontejnera za pokretanje python aplikacije *app6.py*.

Ograničenja za zadatak:

1) Omogućiti da predefinisana bazna slika bude python:3.9-slim, ali da je dozvoljeno da se slika kontejnera napravi i sa drugom verzijom bazne slike bez menjanja samog Dockerfile-a
2) Dodati 2 proizvoljne labele datoj slici kontejnera.
3) Program treba predefinisano da ispisuje poruku Hello, Docker!

Dodatno:

- Ispisati drugačiju poruku kao izlaz programa prosleđivanjem teksta poruke prilikom pokretanja kontejnera.

```
Dockerfile:

ARG PYTHON_VERSION=3.9-slim
FROM python:${PYTHON_VERSION}
ENV GREETING="Hello, Docker!"
LABEL verzija=v1
WORKDIR /app
COPY app6.py .
CMD ["python", "app6.py"]
```

Prosleđivanje drugog argumenta slici kontejnera:

Terminal:

$ docker build --build-arg PYTHON_VERSION="3.8" -t <naziv_slike> .

Ispisivanje drugačije poruke kao izlaz poruke:

Terminal:

$ docker run -e GREETING="Bye, Docker!" <naziv_slike>

Primer 7.

Napisati sliku kontejnera koja pokreće nginx web server, a zatim na svakih 13 sekundi proverava da li web server zaista radi. Ukoliko nema odgovora od web servera nakon 5 sekundi, smatrati da server nije funkcionalan.

Pomoć za zadatak:

- Nginx server se podiže predefinisano na portu 80, te se za proveru responzivnosti može koristiti curl naredba u sledećem obliku:

```
curl -f http://localhost/ || exit 1
```

Dockerfile:

```
FROM nginx:latest
HEALTHCHECK --interval=13s --timeout=5s CMD curl -f http://localhost/
|| exit 1
EXPOSE 80
```

Primer 8.

Napisati sliku kontejnera u nekoliko različitih varijanti. Podesiti radni direktorijum u kontejneru na /app.

a) test.txt kopirati na putanju .
b) test.txt kopirati na putanju /app
c) test.txt kopirati na putanju app
d) test.txt kopirati na putanju app/
e) test.txt kopirati na putanju /app/
f) probati varijacije gore navednih primera ali za kopiranje direktorijuma test umesto fajla test.txt. Koja rešenja prikazuju naziv direktorijuma test u kontejneru, a ne samo njegov sadržaj?

Dodatno: Insalirati ekstenziju za docker u okviru Visual Studio Code-a za lakše analiziranje zadatka.

Primer 9.

Kreirati sliku kontejnera na osnovu arhiviranog ubuntu OS-a.

Pomoć:

- Osnova slike mora biti prazna.
- Prilikom pokretanja kontejnera pokrenuti bash shell

Dodatno:

- Pokrenuti kontejner u interaktivnom režimu i kreirati novi fajl unutar kontejnera
- Instalirati docker ekstenziju sa Visual Studio Code, te pristupiti datom kontejneru preko ekstenzije i preuzeti kreirani fajl.

Dockerfile:

```
FROM scratch
ADD ubuntu-bionic-oci-amd64-root.tar.gz /
CMD ["/bin/bash"]
```