



Fakultet  
elektrotehnike i  
računarstva

DIPLOMSKI RAD br. 2124

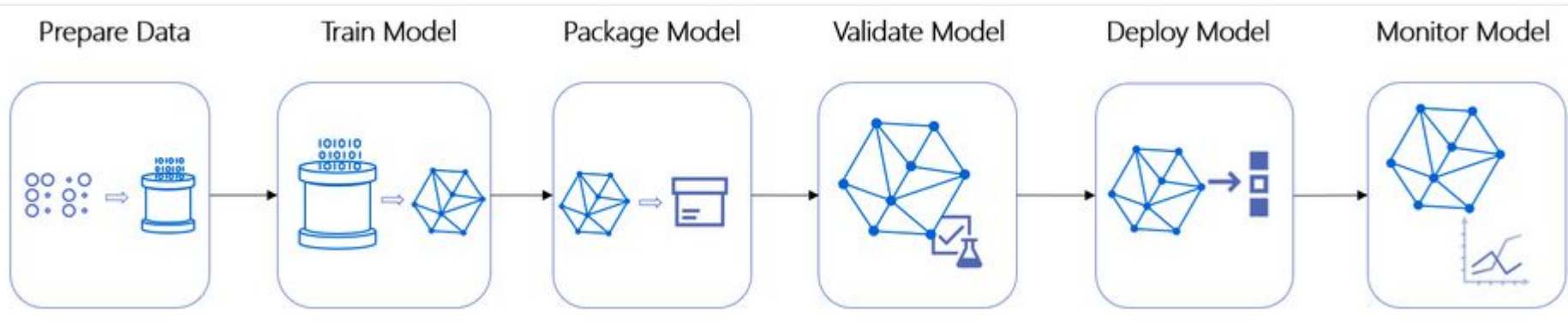
# Skalabilna arhitektura mikrousluga zasnovana na platformi Docker s primjenom na probleme strojnog učenja

Mentor: prof. dr. sc. Alan Jović

Student: Lucia Penić

Zagreb, lipanj 2020.

# Primjena kontejnerske tehnologije na probleme strojnog učenja



Slika 1. Klasični cjevovod strojnog učenja

- primjena moguća u različitim fazama:
  - priprema podataka
  - učenje modela
  - isporuka modela
- postoje već razvijena rješenja kroz web platforme kao Hadoop i Apache Spark

# Prednosti

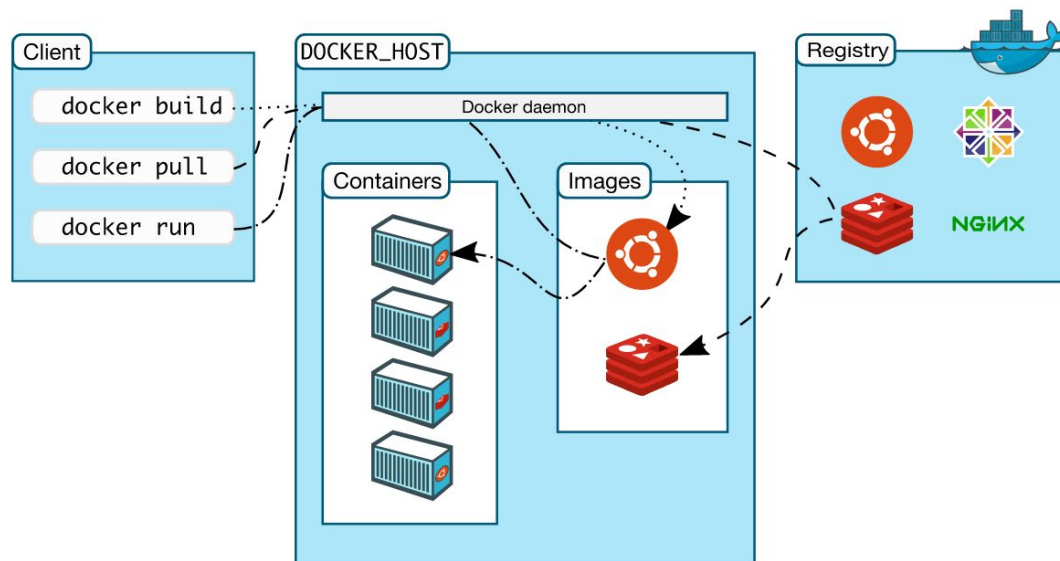
- izolacija
- prenosivost
- replikacija
- brzina
- bolje iskorištenje resursa
- ponovno korištenje
- paralelizacija
- fleksibilnost u izboru tehnologija

# Mane

- složena orkestracija
  - otežan nadzor mikrousluga
  - prilagodba algoritama
  - prilagodba komunikacije
  - otkrivanje pogreške
  - održavanje visoke raspoloživosti
-

# Docker

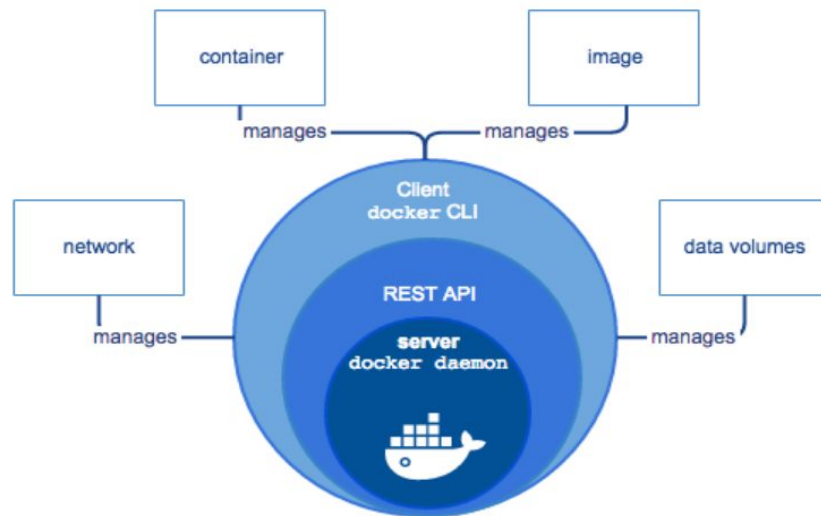
“Platforma Docker je sustav otvorenog koda koji pruža niz alata za razvoj, isporuku i pokretanje korisničkih programa”



Slika 2. Komponente sustava Docker

# Docker

“Platforma Docker je sustav otvorenog koda koji pruža niz alata za razvoj, isporuku i pokretanje korisničkih programa”



Slika 3. Komponente Docker Engine

# Dockerova slika

- *read-only* predložak
- naredbe za izgradnju kontejnera
- moguće korištenje postojećih ili stvaranje vlastite slike
- konfiguracijska datoteka - Dockerfile
- slojevita izgradnja (optimizacija)
- Docker-Hub

# Dockerov kontejner

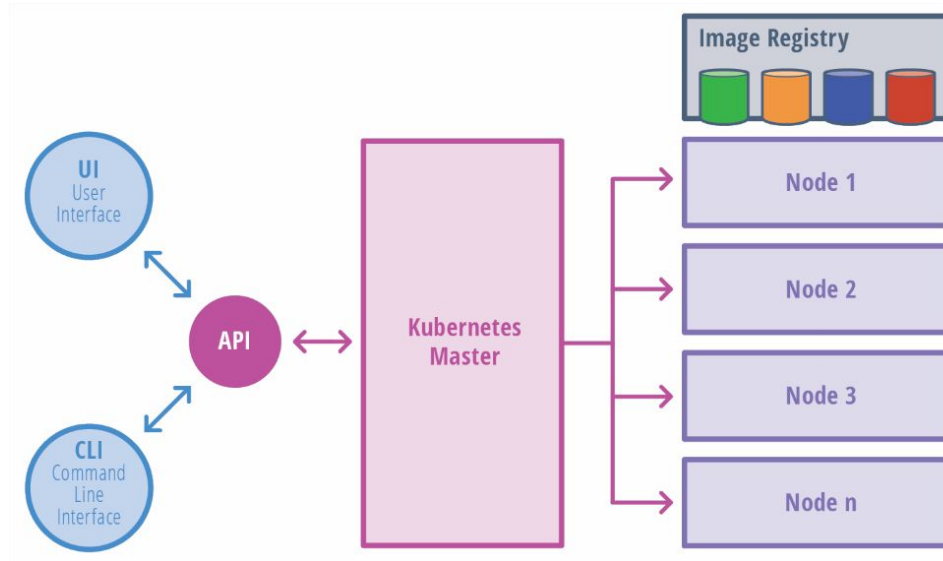
- pokrenuta instanca slike
  - sve izmjene nastale nakon izgradnje su privremene
  - izolacija od datotečnog sustava domaćina
  - ima vlastiti IP
-

# Primjer Dockerfilea

```
FROM python:3.6
WORKDIR /app
COPY requirements.txt /app/requirements.txt
RUN pip install -r requirements.txt
ADD . /app
ENTRYPOINT [ "python" ]
CMD [ "app.py" ]
```

# Orkestracija sustava - platforma Kubernetes

“Kubernetes (K8s) je sustav otvorenog koda koji omogućava korisnicima automatiziranu isporuku, skaliranje i upravljanje kontejniziranim aplikacijama”



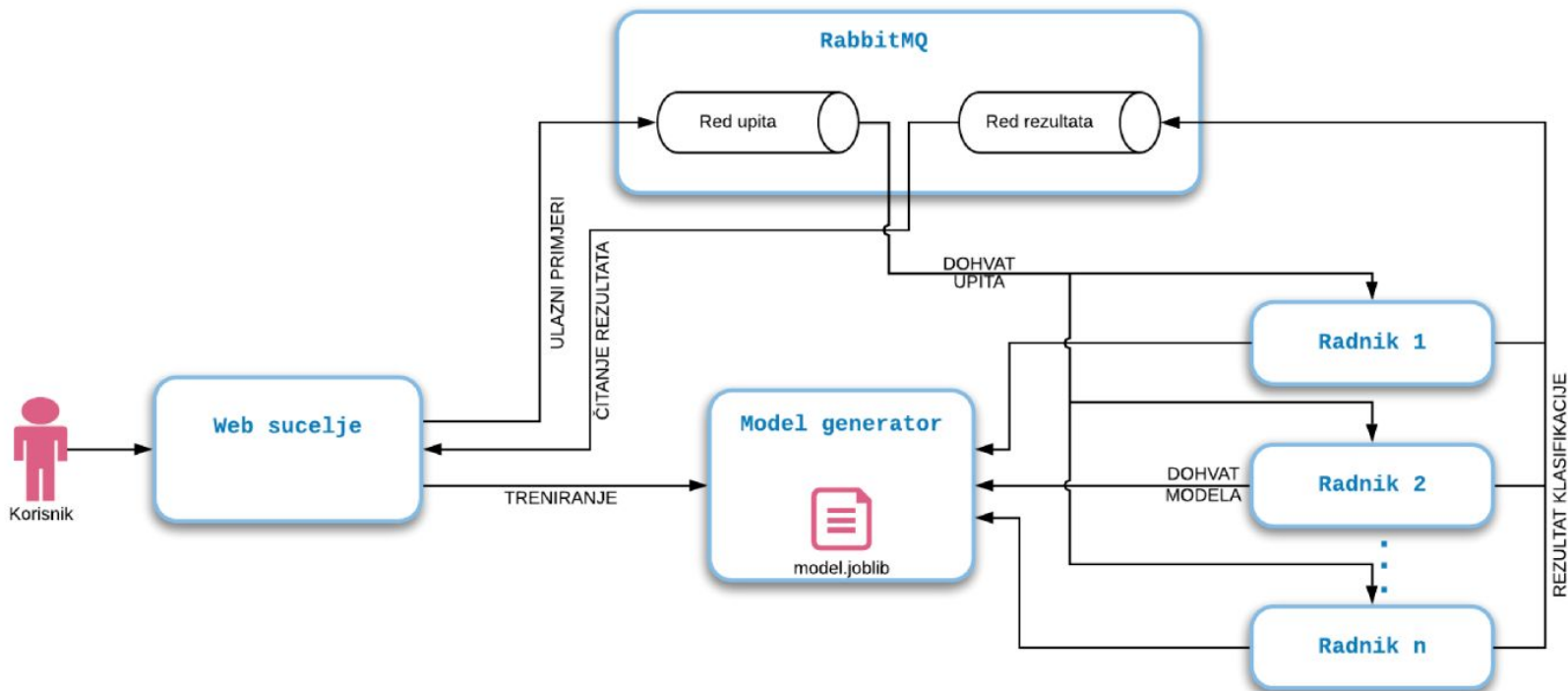
Slika 4. Organizacija sustava Kubernetes



# Orkestracija sustava - platforma Kubernetes

- organizacija računala u grozd (engl. *cluster*)
- jedan ili više master čvorova koji upravljaju grozdom (engl. *control plane*)
- više različitih apstrakcija / objekata: ljuska (engl. *pod*), imenski prostor (engl. *namespace*), usluga (engl. *service*), isporuka (engl. *deployment*) i mnogi drugi
- osnovna jedinica - ljuska
- deklarativni plan u obliku JSON ili YAML datoteke

# Praktični dio



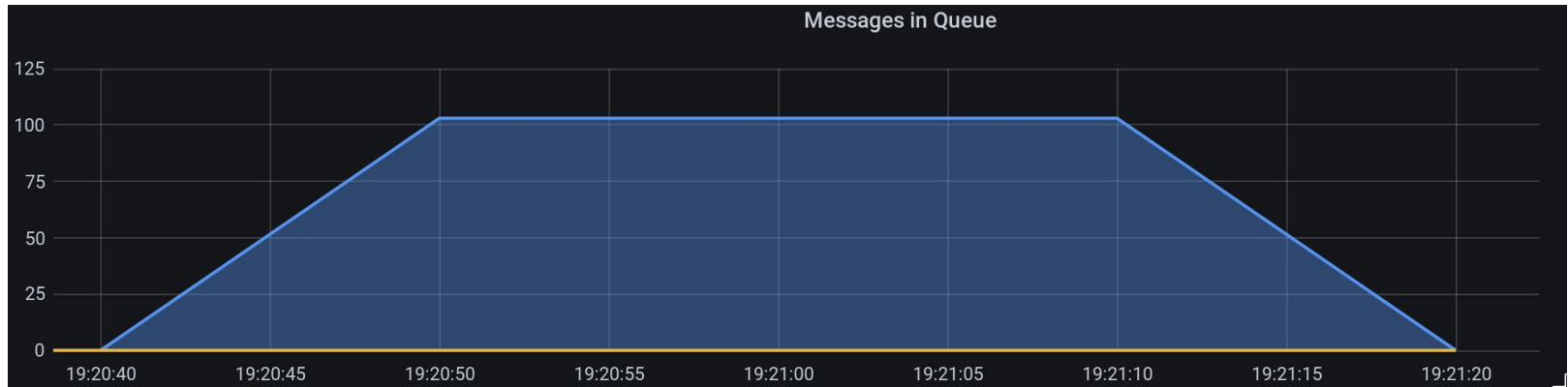
Slika 5. Grafički prikaz implementiranog sustava

- četiri usluge: RabbitMQ, web sučelje, dio za učenje modela (model-generator) i dio koji radi predviđanje (model-radnik)
- kontejnerizacija koristeći sustav Docker
- orkestracija koristeći sustav Kubernetes:
  - Operacijski sustav: Debian GNU/Linux 9
  - Procesor: 2 jezgre
  - Radna memorija: 4 GB
  - Fizička memorija: 80 GB SSD

# Vrednovanje programskog rješenja

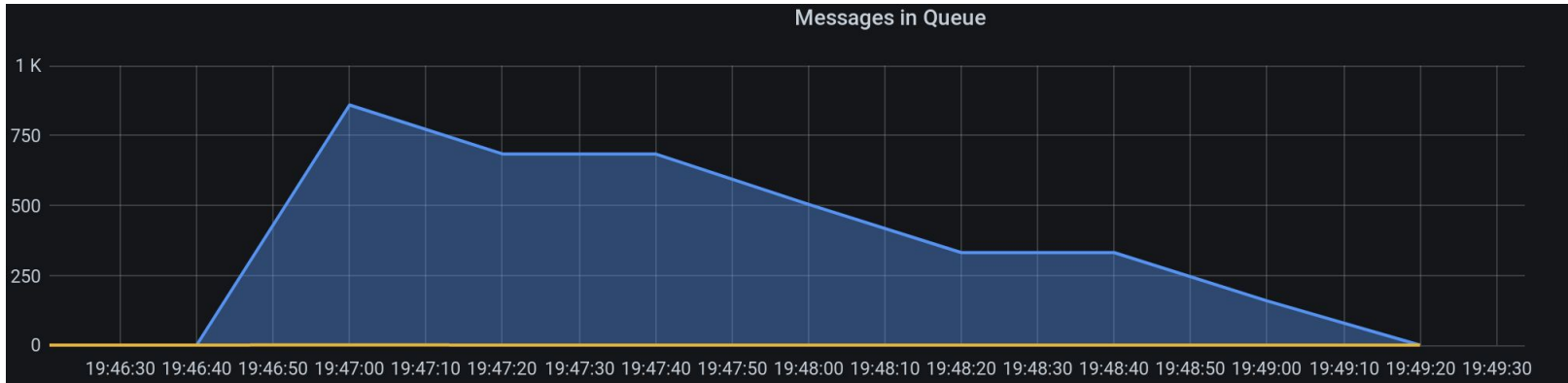
- prvi korak: priprema modela putem web sučelja
  - analiza sentimenta
  - točnost:75.22%
  - preciznost:75.73%
  - odziv: 74.10%
  - mjera F1: 75.54%
- test opterećenja s 1000 zahtjeva, odnosno poruka u redu poruka
- test s jednim radnikom koji se ponaša kao monolitna aplikacija i test sustava u kojem se broj radnika prilagođava opterećenju

- korišten samo jedan radnik
- na raspolaganju ima 500m (*milicorea*) procesorske snage
- vrijeme obrade: 30 sekundi



Slika 6. Test s jednim radnikom

- korišten Horizontal Pod Autoscaler (HPA) za skaliranje do pet radnika
- svaki na raspolaganju ima 100m (*milicorea*) procesorske snage
- vrijeme obrade: 170 sekundi
- razliku u vremenima obrade je moguće objasniti
  - vremenom potrebnim da se 4 dodatna kontejnera podignu i odrede sve potrebne konfiguracijske procese
  - vremenom potrebnim da se uoči povećano opterećenje sustava (HPA 15 sekundi)



Slika 7. Test s pet radnika



Hvala na  
pažnji!