Razvoj softvera II

Sadržaj

- Savremeno poslovanje
- Infrastruktura poslovanja - Old school
- Infrastruktura poslovanja - Hipervizor
- Infrastruktura poslovanja - Docker
- Arhitektura - Docker vs VM
- Docker – ključni pojmovi
- Docker komponente
- Docker komande

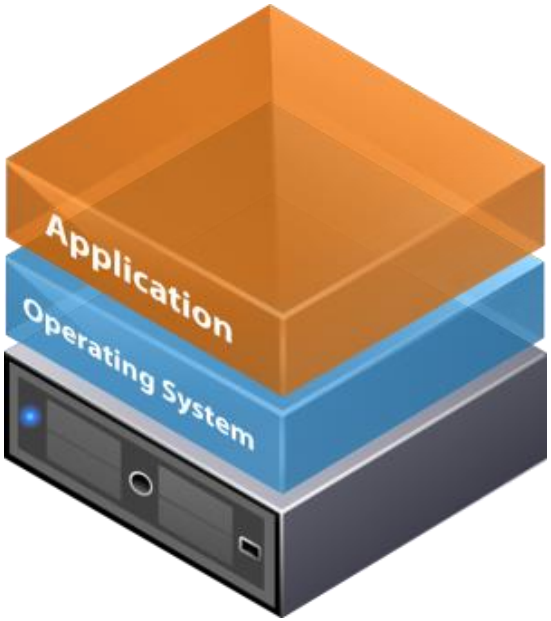
Savremeno poslovanje

- Savremeno poslovanje je gotovo nemoguće zamisliti bez adekvatnih aplikacija koje su u stanju podržati sve najznačajnije faze tog poslovanja. Drugim riječima, bez pretjerane skromnosti možemo reći da bez poslovnih aplikacija nema ni poslovanja
- U ne tako davnoj prošlosti svaka kompanija je posjedovala svoju serversku infrastrukturu koja je osiguravala dostupnost svih potrebnih aplikacija. Nerijetko se za potrebe svake značajnije aplikacije kupovala po jedna serverska konfiguracija koja je pored nezanemarivih troškova nabavke zahtijevala: troškove električne energije neophodne za rad i klimatizaciju, održavanje (nadogradnju, sigurnosne i mrežne postavke) i dr.

Infrastruktura poslovanja

Old school

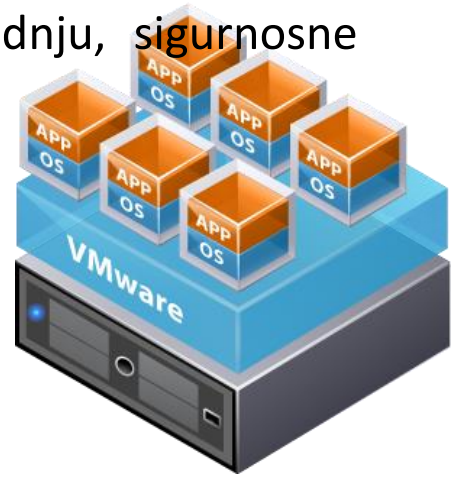
- Pristup: jedna aplikacija jedan server
- Negativni aspekti ovog pristupa je neiskorištenost resursa jer se gotovo uvijek nabavlja konfiguracija koja može zadovoljiti mnogo veće potrebe od stvarnih (često je iskoristivost kapaciteta iznosila najviše 30%)



Infrastruktura poslovanja

Hipervizor

- Pojava hipervizora koji je omogućio kreiranje i izvršavanje većeg broja virtualnih mašina u okviru jednog fizičkog uređaja je označila novu eru u iskoristivosti infrastrukturnih kapaciteta
- Pored velikog broja prednosti, ovaj pristup je imao i određene nedostatke:
  - Svaka virtualna mašina zahtijeva operativni sistem (licence, hardverski resursi i dr.)
  - Svaka virtualna mašina zahtijeva održavanje (nadogradnju, sigurnosne postavke, monitoring i dr.)



Infrastruktura poslovanja

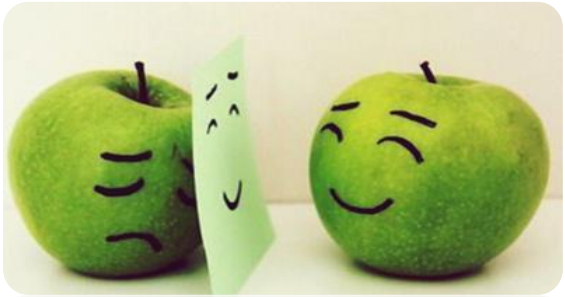
Docker

- Docker je open-source projekt koji automatizira isporuku aplikacija unutar softverskih *kontejnera*, pružajući dodatni sloj apstrakcije i automatizacije operativnog sistema
- Analogija je preuzeta od transportnih kontejnera koji su uvijek isti, bez obzira na sadržaj, te na taj način odgovaraju svim kamionima, dizalicama, brodovima i drugim sredstvima i uređajima u procesu.

Transport komponenti razvojnog okruženja ☺

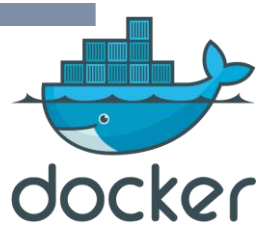
Old school vs Docker

- Transport sa i bez kontejnera se može doslovno preslikati u područje razvoja softvera (razvoj sa i bez korištenja kontejnera)



Infrastruktura poslovanja

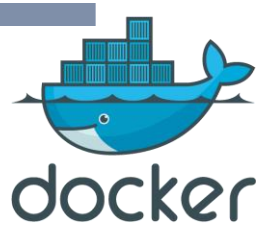
Docker



- Neke od prednosti koje docker sa sobom donosi u područje razvoja softvera su:
- *Postavljanje razvojnih okruženja je dosta brzo i unificirano* – postavljanje svih potrebnih komponenti (baze, web servera, sigurnosnih postavki i dr.) je jako teško uraditi za svakog programera, a posebno za članove tima koji rade na udaljenim lokacijama. Zahvaljujući docker-u, dovoljno je kreirati adekvatne kontejnere i učiniti ih dostupnim članovima tima
- *Minimizira potencijalne konflikte između aplikacija* - u slučaju da ne želimo migrirati na novu verziju framework-a ili koristiti neke specifične postavke koje zahtijevaju pojedine aplikacije, dovoljno je samo da promijenimo kontejner
- *Jednostavno kretanje softvera kroz različita okruženja* (razvojno, testno, produkcija) – minimiziraju se komentari: „kod mene ne radi...“
- *Brža isporuka kvalitetnog softvera*

Infrastruktura poslovanja

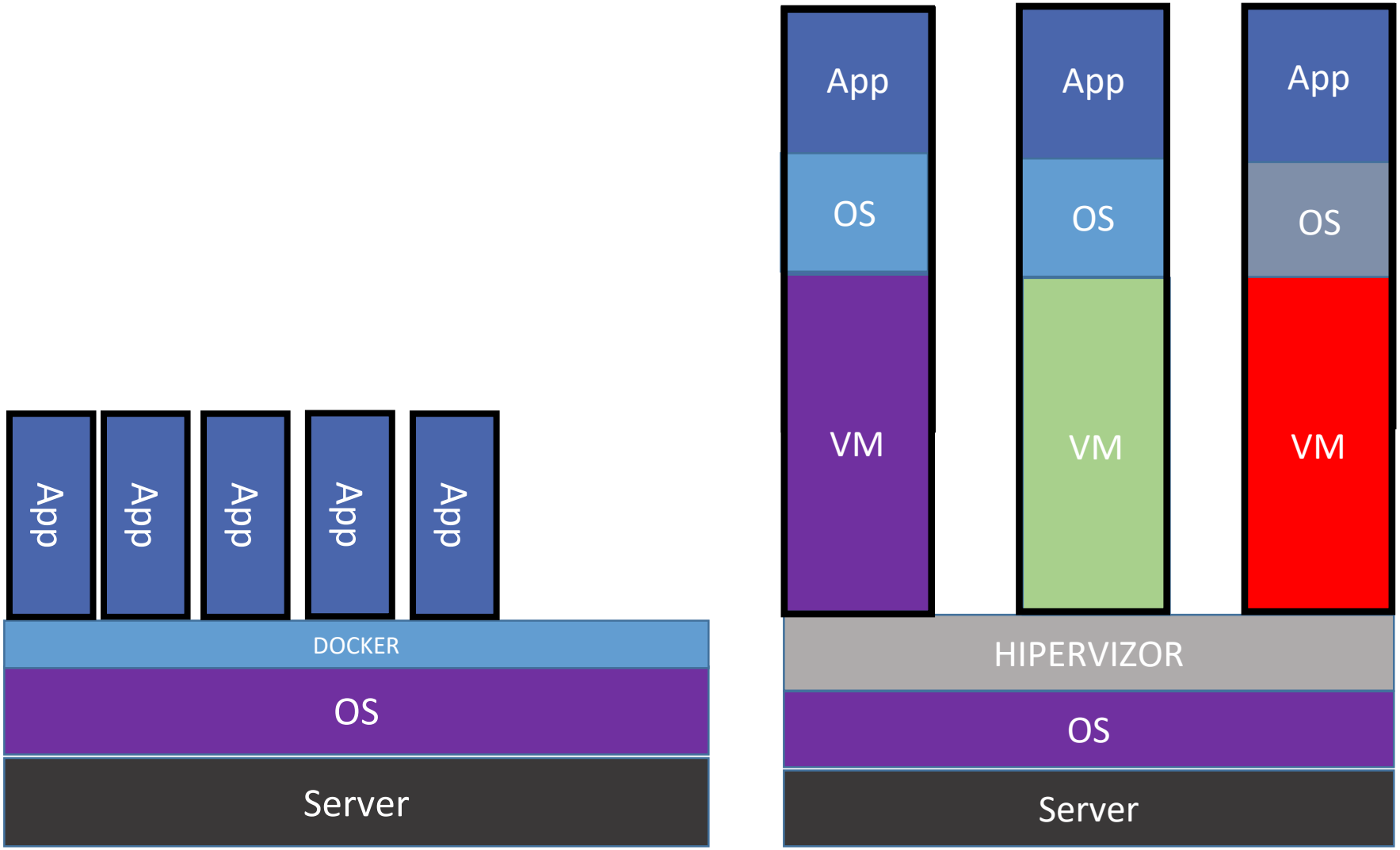
Docker



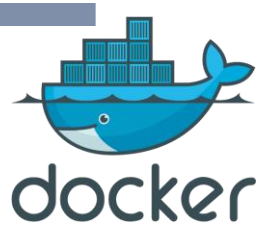
- Kontejneri su izolirani, ali dijele operativni sistem što omogućava:
  - Eliminisanje velikog broja licenci koje zahtijevaju operativni sistem
  - Eliminisanje potrebe za stalnim održavanjem i nadogradnjom velikog broja različitih operativnih sistema
  - Jednostavnije izvršavanje na različitim platformama i serverskim infrastrukturama (fizički i virtualni serveri, VM, Cloud ...)
  - Jednostavniju migraciju – prelazak na novu verziju zahtijeva samo par koraka
  - Ponovnu iskoristivost i prenosivost – kontejneri se na jednostavan način distribuiraju između pojedinaca, timova ili samih okruženja (razvojno, testno, produkcijsko)
  - Samostalnost okruženja – kontejneri minimiziraju zavisnost između pojedinih komponenti i/ili verzija
  - Jednostavno upravljanje – kontejnere je na jednostavan način moguće povezati, grupisati, te na taj način njima upravljati

Arhitektura

Docker vs VM

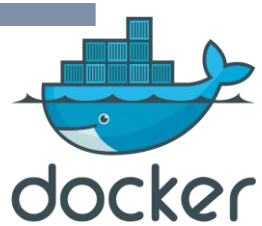


Docker – ključni pojmovi



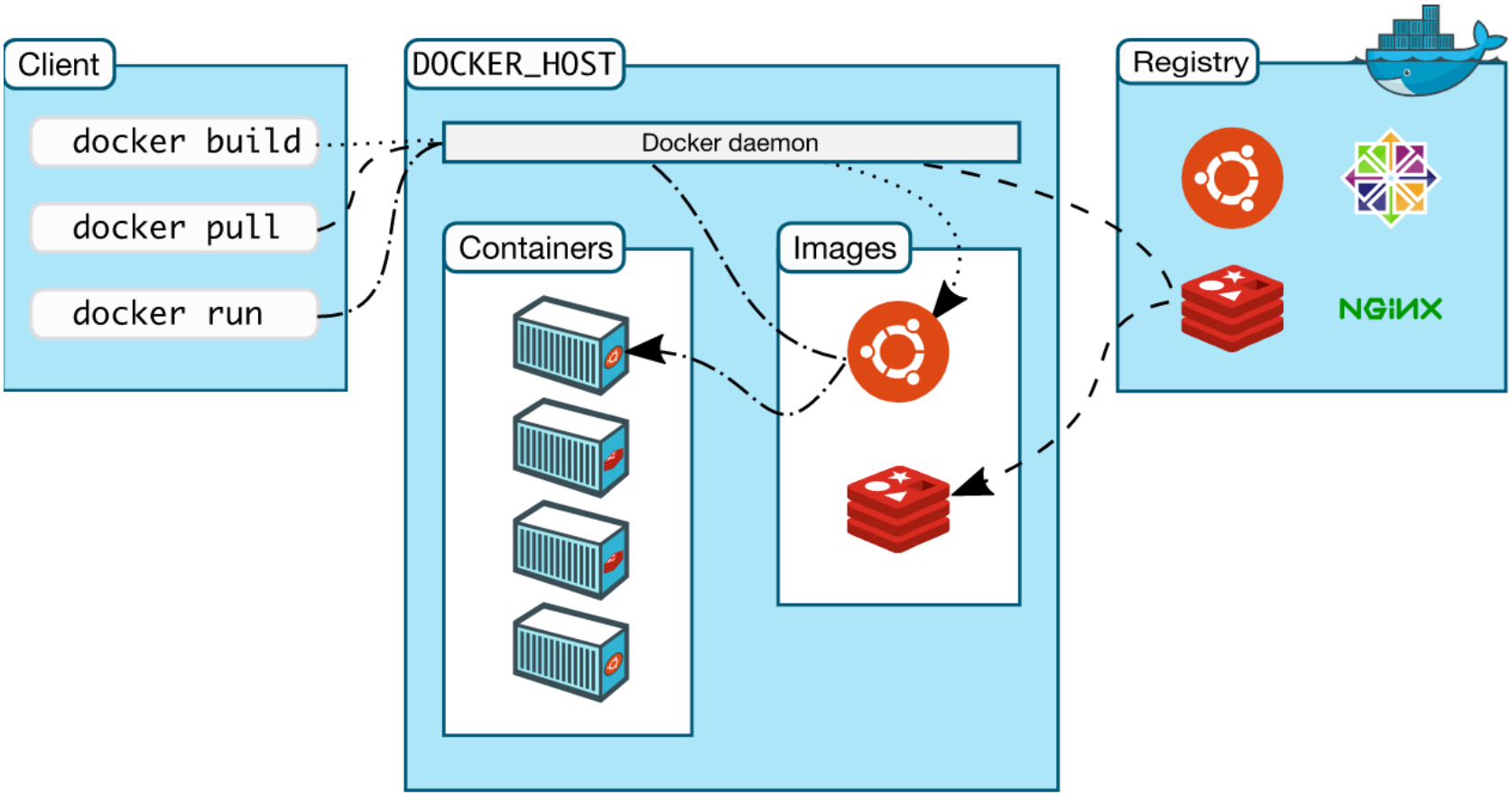
- Ključni pojmovi za docker su **image** i **kontejner**
- **Image** predstavlja kolekciju fajlova i meta podataka koji sačinjavaju root fajl sistem kontejnera, a obično su sastavljeni od više slojeva od kojih se neki mogu modifikovati
- **Kontejner** se može posmatrati kao set procesa koji se izvršavaju u okviru pomenutog fajl sistema. Drugim riječima, kontejner se može posmatrati kao pokrenuti/startani image.

Docker komponente

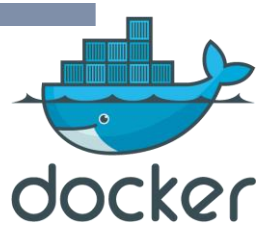


- Docker možemo posmatrati kroz tri komponente:
  - **Client** - omogućava interakciju sa komponentom Host-a koja se naziva Server ili Daemon. Pomenuta komponenta je zadužena za upravljanje sa image-ima i kontejnerima, te interakciju sa Hub-om
  - **Host** - u okviru ove komponente se izvršavaju kontejneri. Kada se preko klijenta zahtijeva pokretanje image-a, Server ili Daemon prvo provjerava da li taj image postoji u lokalnom repozitoriju, te ukoliko ne postoji vrši njegovo preuzimanje sa Hub-a (registry-a) i pokretanje u okviru kontejnera
  - **Hub** (hub.docker.com) - Predstavlja centralni repozitorij za publikovanje image-a. Pored javnog, posjeduje i privatne repozitorije

Docker komponente



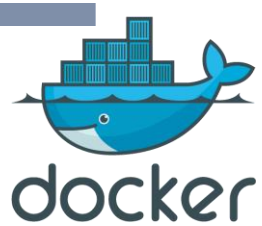
Docker



- Na koji način koristiti docker?
- Preuzeti verziju namijenjenu operativnom sistemu koji koristi računar na kome radite (npr. Docker for Windows -> https://www.docker.com/docker-windows)
- Nakon instalacije (potrebno je imati podršku za virtualizaciju), za interakciju sa Docker-om se može koristiti Command Prompt ili PowerShell
- Sve komande počinju sa ključnom riječju `docker`

```
PS C:\Users\RSII> docker version
Client:
 Version:      17.03.1-ce
 API version:  1.27
 Go version:   go1.7.5
 Git commit:   c6d412e
 Built:        Tue Mar 28 00:40:02 2017
 OS/Arch:      windows/amd64
Server:
 Version:      17.03.1-ce
 ....
```

Docker



Komanda	Opis
`docker version`	informacije o klijentu i serveru
`docker info`	informacije o broju image-a i kontejnera koji se trenutno izvršavaju ili su pauzirani ili stopirani, njihovim verzijama i dr.
`docker ps` `docker ps – a`	informacije o kontejnerima koji se trenutno izvršavaju ili su se ranije izvršavali na docker-u

```
PS C:\Users\RSII> docker info
Containers: 55                    <<<<< broj kontejnera
 Running: 0
 Paused: 0
 Stopped: 55
Images: 9                          <<<<< broj image-a
Server Version: 17.03.1-ce
...
```

Docker

Komanda	Opis
`docker run`	starta novi kontejner
`docker stop`	stopira izvršenje kontejnera
`docker pull`	kopira image (određenog naziva npr. mysql) na docker host
`docker images`	ispisuje listu image-a na docker host-u
`docker rmi`	uklanja image sa docker host-a

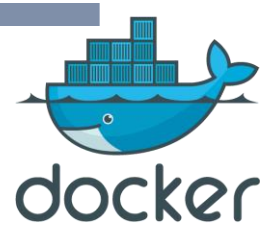
```
PS C:\Users\RSII> docker images
REPOSITORY                      TAG                 IMAGE ID            CREATED             SIZE
microsoft/dotnet-samples        latest              b4df79aa8780        12 days ago         251 MB
wordpress                       latest              ca96afcfa242        2 weeks ago         406 MB
ghost                           latest              142165321ade        2 weeks ago         334 MB
mysql                           latest              e799c7f9ae9c        3 weeks ago         407 MB
kitematic/hello-world-nginx     latest              03b4557ad7b9        24 months ago       7.91 MB
....
```

Docker

```
PS C:\Users\RSII> docker pull ubuntu:latest
latest: Pulling from library/ubuntu
bd97b43c27e3: Pull complete
6960dc1aba18: Pull complete
2b61829b0db5: Pull complete
1f88dc826b14: Pull complete
73b3859b1e43: Pull complete
Digest:
sha256:ea1d854d38be82f54d39efe2c67000bed1b03348bcc2f3dc094f260855df
f368
Status: Downloaded newer image for ubuntu:latest
PS C:\Users\RSII> docker run ubuntu
```

- Nakon naziva image-a može se navesti njegova konkretna verzija (npr. 1.0.8). U ovom slučaju je korištena ključna riječ *latest* koja označava posljednju verziju
- Kreiranjem naloga na sajtu hub.docker.com omogućen je pristup informacijama o svim verzijama image-a, te uočenim nedostacima za svaku od njih

Docker

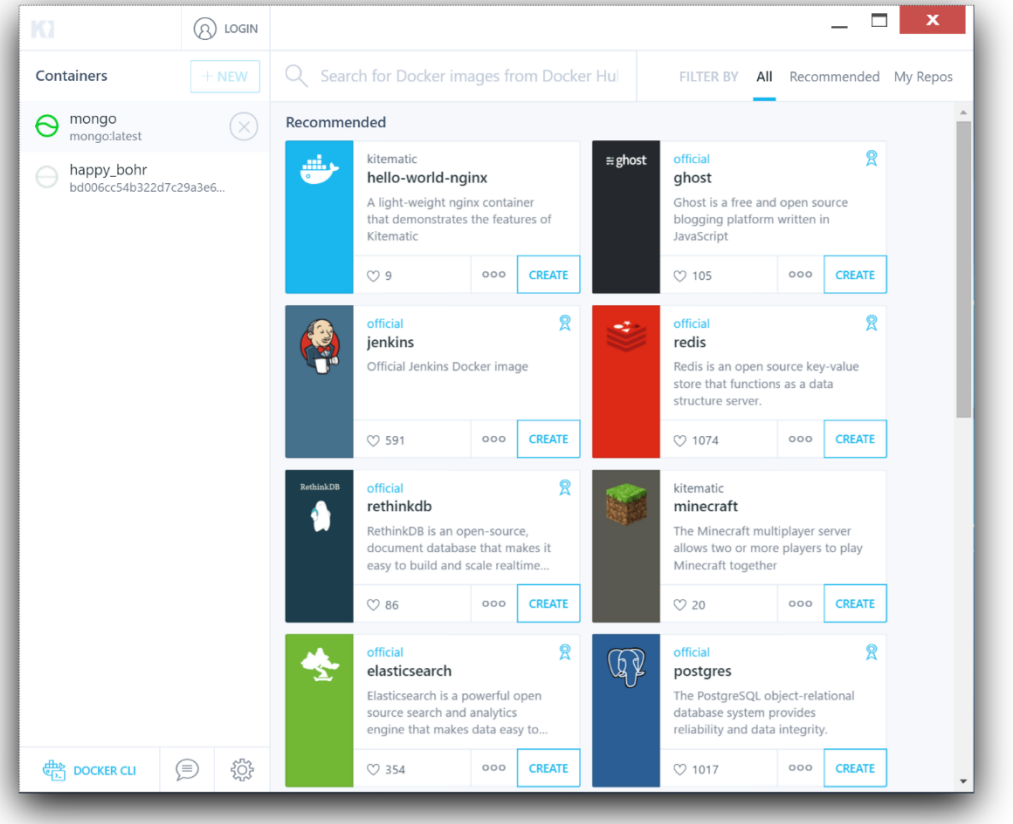



```
PS C:\Users\RSII> docker run -it --name wordpressFIT -p 80:8080 wordpress
WordPress not found in ... - copying now...
Complete! WordPress has been successfully copied to ...
...
[Mon May 29 22:54:22.436625 2017] [mpm_prefork:notice] [pid 1] AH00163:
Apache/2.4.10 (Debian) PHP/5.6.30
resuming normal operations
...
```

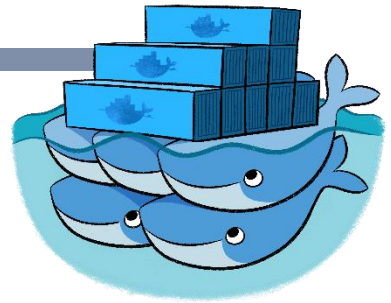
- **it** – označava da interakciju sa kontejnerom želimo obavljati kroz komandnu liniju u kojoj se trenutno nalazimo. Postoji i opcija –d koja označava detached mode
- **name** – omogućava definisanje naziva za izvršenje konkretnog kontejnera
- **p** – omogućava mapiranje portova između trenutne mašine i kontejnera
- **wordpress** – naziv image-a kojeg želimo startati

Docker

- Za sve one koji nisu ljubitelji konzolne interakcije, dostupan je alat Kitematic koji omogućava jednostavnu manipulaciju image-ima i kontejnerima



Docker swarm mode



- Docker engine je moguće grupisati u klaster, te se takav način rada naziva swarm i njime se osigurava skalabilnost
- Swarm mode definiše postojanje menadžer članova koji su zaduženi za raspodjelu zadataka ostalim članovima swarm-a (advertise-addr se koristi za definisanje adrese za komunikaciju sa swarm-om)

```
PS C:\Users\RSII> docker swarm init --advertise-addr 192.168.1.12:2377
Swarm initialized: current node (1e5vi0tazuyde4cj1ryegpoxx) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join \
    --token SWMTKN-1-4n2s2z4iziu1qlm4jsn0wkw1a22p0j5qvpdkmo823vebcbxiu5-
b5nx2o6k5e7tlr69ek4yu738n \
    192.168.1.12:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow
the instructions.
...
```

KRAJ PREZENTACIJE