SVEUČILIŠTE U RIJECI

FAKULTETE INFORMATIKE I DIGITALNIH TEHNOLOGIJA

DOKUMENTACIJA ZA SEMINARSKI RAD

Autor: Leo Vukoje

Kolegij: Mrežni i mobilni operacijski sustavi

Mentor: izv. prof. dr. sc. Božidar Kovačić

**Docker** je otvorena platforma za razvijanje, slanje i pokretanje aplikacija koja nam omogućava da aplikacije odvojimo od lokalne infrastrukture kako bi se proizvod mogao brže dostaviti. Korištenjem značajki koje nam Docker nudi možemo znatno smanjiti razmak između pisanja koda i pokretanja aplikacije u produkciji.

**Kontejner** je jedan proces na našoj mašini koji je izoliran od svih ostalih procesa na poslužiteljskoj mašini. Kontejner je jedna instanca slike (*engl. image*) koju možemo pokrenuti, zaustaviti, premještati i brisati. Može biti pokrenut na lokalnoj, virtualnoj mašini i na oblaku. Prenosiv je, što znači da može biti pokrenut na bilo kojem operacijskom sustavu. Izoliran je od ostalih kontejnera te pokreće svoje softvere i konfiguracije.

**Slika** nekog kontejnera pruža izolirani sustav datoteka kojeg koristi taj kontejner. Slika mora sadržavati sve što je kontejneru potrebno da može pokrenuti aplikaciju: konfiguracije, skripte, varijable okruženja, defaultne naredbe za pokretanje i ostale meta-podatke.

**Volumeni** su preferirani mehanizmi za spremanje podataka koje generiraju neki kontejneri i u potpunosti su upravljani od strane Dockera. Kada spremaju podatke ne povećavaju veličinu kontejnera koji ih koristi i njihov sadržaj postoji i van životnog ciklusa kontejnera.

Za ovaj seminarski rad bilo je potrebno u *docker-compose* datoteci navesti sve slike na temelju kojih sam htio izraditi kontejnere. Tako sam npr. za web poslužitelje iskoristio sliku distribucije *ubuntu:22.04* operacijskog sustava Linux, unutar koje sam preko naredbi u *Dockerfile-u* dohvatio Python te module Flask i Redis koji su mi bili potrebni za izradu minimalnog web servera sa jednim endpoint-om.

Docker može izgraditi slike čitajući naredbe iz **Dockerfile-a**. To je dokument koji sadrži sve naredbe koje se mogu pozvati za postavljanje neke slike. Tako sam za postavljanje web poslužitelja naveo naredbe kojima iz slike *ubuntu-a* prvo dohvaćam Python i instaler paketa Pip. Nakon toga sa pip-om instaliram module Flask i Redis i na kraju u komandnom sučelju pokrećem skriptu *hello.py*.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

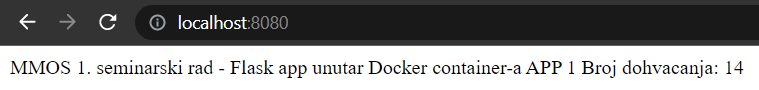
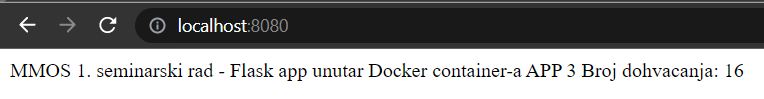
Da bih mogao koristiti Redis unutar Python aplikacije, morao sam u *docker-compose* datoteci i navesti sliku Redis-a u čijem kontejneru se pokretao server za predmemoriju. Tako sam u Python aplikacijama za ime Redis server naveo ime koje sam postavio za *service* u *docker-compose* datoteci.

Glavna **razlika između Docker-a i virtualne mašine** je u njihovim arhitekturama. Virtualne mašine imaju domaćinski OS i OS gosta unutar svake virtualne mašine, dok se Docker kontejneri pokreću na jednom fizičkom serveru kojeg dijele međusobno. Dijeljenje domaćinskog OS ih čini laganim i smanjuje vrijeme pokretanja. Docker kontejneri su korisni ako moramo pokrenuti više aplikacija nad jednim kernelom nekog OS-a, dok su virtualne mašine potrebne kada neke aplikacije zahtijevaju pokretanje na drukčijem operacijskom sustavu.

**Balanser opterećenja** je mrežno rješenje korišteno da rasprši promet preko nekoliko servera unutar jedne farme servera. Balanseri opterećenja poboljšavaju dostupnost aplikacije i preveniraju preopterećenost servera. Svaki balanser opterećenja se nalazi između klijentskih uređaja i backend servera, gdje prima i onda raspoređuje nadolazeće zahtjeve bilo kojem dostupnom serveru koji je u mogućnosti ispuniti taj zahtjev.

Balanser opterećenja u mom projektu sam iskoristio preko slike nginx-a. Unutar Dockerfile-a sam dohvatio sliku, te prebacio moju konfiguraciju u izolirani datotečni sustav, a u konfiguraciji sam naveo weight parametar koji utječe na odluku balansera, u mom slučaju weight parametar znači da će od 10 nadolazećih zahtjeva 4 sletjeti na prvu aplikaciju, a 3 na drugu i 3 na treću. Funkcioniranje balansera sam testirao kroz web preglednik odlaskom na njegov URL.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

U početku kontejneri su zamišljeni da budu **stateless**, bez pohrane ikakvog stanja. Sve baze podataka i spremanja podataka su prepušteni operacijskom sustavu. Danas tvrtke poput Dockera, Kubernetesa i drugih, pružaju način za rad i sa stateless i sa **stateful** kontejnerima.

**Terraform** je alat za zapisivanje infrastrukture u obliku koda koji pruža mogućnost upravljanja infrastrukturom uz pomoć konfiguracijskih datoteka umjesto grafičkim sučeljima. Dopušta gradnju, mijenjanje i upravljanje infrastrukturom na siguran, konzistentan način definiranjem resursnih konfiguracija koje se mogu verzionirati, reuse-ati i dijeliti.

Terraform služi kao alat za izgradnju infrastrukture, dok je Docker klasificiran kao virtualna mašina i kontejner koji pojednostavljuje izgradnju i dostavu distribuiranih aplikacija i uz to pruža razdvojenost unutar aplikacije i jeftinu skalabilnost. Što se tiče sigurnosti, podaci o kojima se Terraform brine su najveća briga. Terraform često radi backup podataka u produkciji i ima plan za oporavak u slučaju sigurnosnog ispada (*engl. outage*). Docker garantira da aplikacije koje se pokreću u kontejnerima su potpuno izolirane jedna od druge iz sigurnosne perspektive, dajući potpunu kontrolu nad upravljanjem prometom i tijekom.