&Bcy;&rcy;&ocy;&jsercy; &scy;&lcy;&ocy;&bcy;&ocy;&dcy;&ncy;&icy;&khcy; &mcy;&iecy;&scy;&tcy;&acy; &zcy;&acy; &scy;&iecy;&pcy;&tcy;&iecy;&mcy;&bcy;&acy;&rcy;&scy;&kcy;&icy; &ucy;&pcy;&icy;&scy;&ncy;&icy; &rcy;&ocy;&kcy; 2012**Univerzitet u Nišu, Elektronski fakultet**

**Katedra za računarstvo**

**Miloš Panić**

**Mikroservisna aplikacija arhitekture vođene događajima za pregled online sadržaja**

**Diplomski rad**

**Niš, 2022**

Sadržaj

[1. Uvod 3](#_Toc94959894)

[1.1. Definisanje problema 3](#_Toc94959895)

[Slika 1.1 – Mikroservisna arhitektura Netflix-a i Amazon-a 4](#_Toc94959896)

[1.2. Organizacija teze 4](#_Toc94959897)

[2. Mikroservisi i mikroservisna arhitektura 6](#_Toc94959898)

[2.1. Mikroservisi 6](#_Toc94959899)

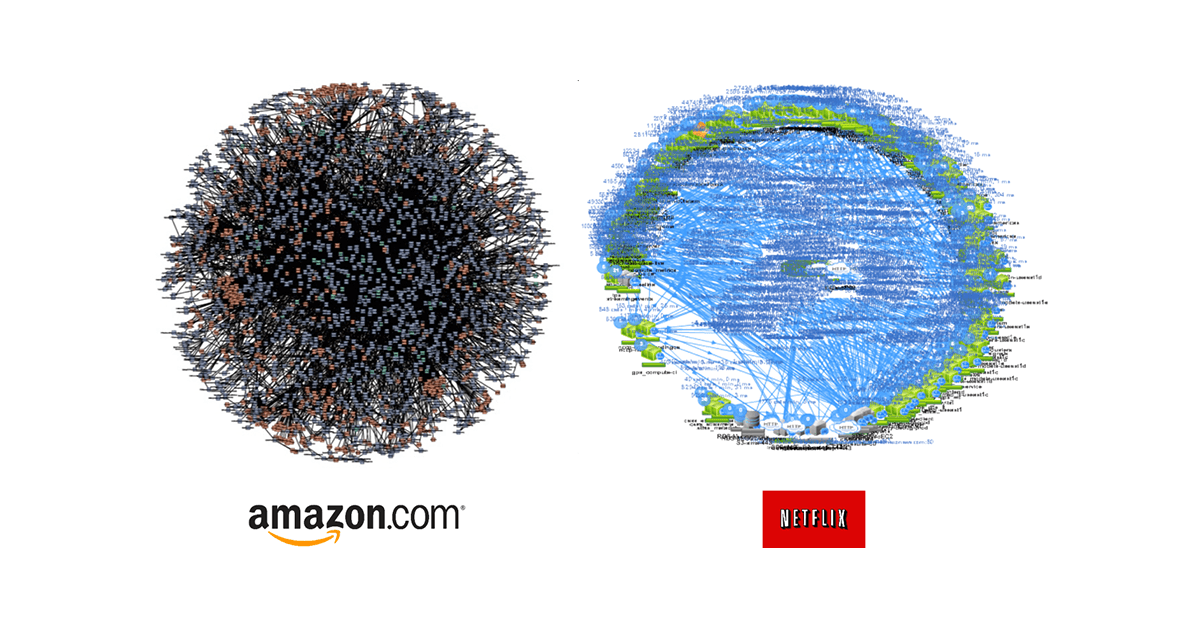
# Uvod

## Definisanje problema

Deljenje video sadržaja u realnom vremenu ili sama mogućnost da aplikacija poseduje opciju za deljenjem kao i pregledavanjem video sadržaja je oduvek bila aktuelna tema i verujem da se to neće menjati u daljem periodu. Počev od samih društvenih mreža, čiji je osnovni cilj upravo međusobno deljenje video sadržaja i slika između korisnika pa do nekih web aplikacija čija je edukaciona svrha, obezbeđivanje različitih video snimaka koje korisnici mogu pregledavati u svrhe učenja i izučavanja različitih oblasti. Sve te aplikacije možda pokrivaju različite domene, međutim sve one imaju jedan zajednički cilj, a to je obezbediti korisnicima mogućnost da pregledavaju različite video snimke bez ikakvog kašnjenja i čekanja da se video snimak učita. Možda to deluje kao naizgled očigledan zahtev i nešto oko čega ne bi trebalo biti problema u današnje vreme, u poređenju sa brzinama internet konekcija, međutim, ukoliko sama arhitektura web aplikacije nije isprojektovana da podnese veliki broj korisnika u realnom vremenu, koji preuzimaju video snimke, odmah se dolazi do problema.

U tom slučaju, na scenu stupa mikroservisna arhitektura koja bi sama po sebi, u skladu sa svojim prednostima i mana, trebala biti odličan izbor za rešavanje prethodno pomenutog problema. Mikroservisna arhitektura, čija popularnost je u naglom porastu, trebalo bi u potpunosti da podnese veliku količinu protoka podataka kroz samu arhitekturu između samih komponenti sistema ali i veliki broj korisnika koji zahtevaju podatke u realnom vremenu. Kako joj i samo ime kaže, mikroservisna arhitektura je sačinjena od velikog broja mikroservisa koji rade zajedno na obezbeđivanju različitih funkcionalnosti korisnicima, u zavisnosti od domena i potreba same aplikacije koja se razvija. U suštini, po meni, glavna prednost je u tome što se može napraviti veliki sistem, sačinjen od velikog broja komponenti između kojih se može vršiti balansiranje opterećenja, odnosno prosleđivanje zahteva ravnomerno mikroservisima kako se ne bi postiglo preopterećenje niti jedne komponente sistema. Tim pristupom odmah dobijamo odgovor zašto bi baš mikroservisna arhitektura bila pogodna za opsluživanje velikog broja korisnika koji zahtevaju veliku količinu podataka u realnom vremenu, kako preuzimanje tako i deljenje podataka.

Mikroservnisna arhitektura, tačnije takozvana mikroservisna arhitektura vođena dogajima(Event-driven architecture), je dosta korišćena danas i biva sve popularnija. Bez ikakve dalje priče, za primer bih najpre naveo Netflix kao kompaniju koja je u potpunosti implementirala mikroservisnu arhitekturu za manje više iste svrhe koje obezbeđuje i aplikacija razvijena za potrebe ovog rada. Njihova arhitektura je sačinjena od velikog broja mikroservisa(po nekim informacijama Netflix danas poseduje preko 1000 mikroservisa koji rade u pozadini i svaki je nadležan za odgovarajući deo aplikacije)[2].



### Slika 1.1 – Mikroservisna arhitektura Netflix-a i Amazon-a

Na slici 1.1 možemo videti na interesantan način reprezentovane arhitekture Netflix-a i Amazon-a. Na prvi pogled, deluje da je to neki preveliki nered u kojem se teško stvari mogu razumeti i koji uz toliko konekcija između komponenata prosto ne može da obezbedi odične performanse korisnicima ali stvarnost je naravno potpuno drugačija. Rezultati su naravno izvanredni i po pitanju Netflix-a prelazak na mikroservisnu arhitekturu sa stare monolitne arhitekture je bio pun pogodak. Broj korisnika je u stalnom porastu, a skalabilnost same mikroservisne arhitektura bi u potpunosti trebalo da podnese taj rast. Netflix trenutno broji preko 93 miliona korisnika.  
  
Distribuirani sistemi, paralelni sistemu su oduvek bili aktuelni i oni su meni bila glavna motivacija za učenjem i razvijanjem aplikacija kod kojih je u osnovi mikroservisna arhitektura. Sama paralelizacija, podela posla između komponenti sistema, samo razmišljanje na taj način da više delova sistema mogu paralelno obavljati poslove u cilju zadovoljenja potreba korisnika je meni oduvek bila fascinantna i privlačna za učenjem. Mikroservisi ciljaju na probleme monolitne arhitekture podržavanjem principa rada po imenu razdvajanje briga što bi bio grubi prevod(Separation of concerns)[1]. Taj princip podržava modularno razbijanje velikih programa u manje softverske komponente pri čemu svaka od tih softverskih komponenti poseduje sopstvenu enkapsulaciju podataka.

## Organizacija teze

U okviru ove podsekcije potrudiću se da ukratko predočim šta će ovaj rad sadržati u nastavku po poglavljima.   
U narednom poglavlju, poglavlju broj 2 pod imenom ,,Mikroservisi i mikroservisna arhitektura” ću se najviše dotaći teorije i pokušati da teoretski opravdam upotrebu mikroservisne arhitekture za potrebe deljenja video sadržaja ali naravno u okviru tog poglavlja biće reči i uopšte o samoj mikroservisnoj arhitekturi.

Treće poglavlje, pod imenom ,,Metode i tehnologije za razvoj mikroservisnih aplikacija” treba da se dotakne nekih dobrih praksi koje su se istakle kroz rad softverskih inženjera kao i tehnologija nastalih u okviru tog procesa i pokušaću da istaknem par tehnologija koje bi bile odlične za razvoj mikroservisnih aplikacija kao i tehnologije koje su danas najviše korišćenije.

Četvro poglavlje će biti u potpunosti posvećeno aplikaciji koja je razvijena za potrebe ovog rada i koja prati ovaj rad. U okviru tog poglavlja pokušaću da kroz različite dijagrame objasnim samu arhitekturu aplikacije, njene funkcionalne i nefunkcionalne zahteve ali i različite implementacione detalje.

U poglavlju nakon četvrtog poglavlja, priča se nastavlja o samoj aplikaciji i u okviru tog poglavlja ću pokušati da još detaljnije opišem funkcionalne zahteve te ću se i dotaći samog testiranja i evaluacije aplikacije.

Poslednje, odnosno šesto poglavlje bi trebalo da bude zaključno poglavlje i u okviru tog poglavlja pokušaću da iznesem neke svoje zaključke o samoj mikroservisnoj arhitekturi te i da sumiram sve stvari koje sam ja naučio u toku razvoja aplikacije.

# Mikroservisi i mikroservisna arhitektura

## Mikroservisi

Kod mikroservisne arhitekture, kao što joj i samo ime govori, osnovna gradivna jedinica, osnovni element u strukturi te arhitekture, je u pravo mikroservis. Mikroservis može da se posmatra kao nezavisna komponenta koja u saradnji sa ostalim mikroservisima obezbeđuje korisnicima različite funkcionalnosti u zavisnosti od potreba i domena same aplikacije za koju se razvija mikroservisna arhitektura.  
  
Mikroservisi su softverske komponte, nezavnisne jedinice deployment-a i razvoja koje sačinjavaju mikroservisnu arhitekturu i oni su u suštini osnovna gradivna jedinica te arhitekure. Mikroservisna arhitektura, odnosno, mikroservisi imaju za cilj da u potpunosti reše neke probleme sa kojima se susretala inicijalna monolitna arhitektura.

Osnovne karakteristike koje bi trebalo da ispunjava svaki mikroservis:

* Pogodan izuzetno za održavanje i testiranje
* Mikroservisi koji sačinjavaju mikroservisnu arhitekturu bi trebali biti slabo spregnuti
* Nezavisna jedinica deployment-a
* Nezavisna jedinica razvoja