

**PLATFORMA ZA STRIMOVANJE VIDEO SADRŽAJA ZASNOVANA NA
MIKROSERVISNOJ I SERVERLES ARHITEKTURI****VIDEO STREAMING PLATFORM BASED ON MICROSERVICE AND SERVERLESS
ARCHITECTURE**Milovan Milovanović, *Fakultet tehničkih nauka, Novi Sad***Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – Ovaj rad se bavi unapređenjem mikroservisne platforme za strimovanje video sadržaja implementirane u ranijem diplomskom radu. U radu je opisana primena serverles (engl. *serverless*) arhitekture i AWS servisa kao što su *Lambda*, *S3* i *CloudFront* sa ciljem da platforma postane globalno dostupna i skalabilna. U radu je takođe dat opis primene AWS CDK i *Serverless Framework* alata za definisanje infrastrukture putem koda.

Ključne reči: *serverless*, strimovanje videa, AWS, *Lambda*, *IaC*, CDK

Abstract – This paper focuses on improving the microservice platform for video streaming developed in the bachelor thesis. It describes the application of serverless architecture and AWS services such as *Lambda*, *S3*, and *CloudFront*, with the aim of making the platform globally accessible and scalable. The paper also provides an overview of the use of AWS CDK and *Serverless Framework* tools for defining infrastructure as code.

Keywords: *serverless*, video streaming, AWS, *Lambda*, *IaC*, CDK

1. UVOD

Sa porastom popularnosti video-sadržaja na internetu, javlja se potreba za platformama koje mogu efikasno upravljati i isporučivati multimedijalni sadržaj korisnicima širom sveta. Striming servisi kao što su *YouTube* i *Netflix* postali su integralni deo svakodnevnog života, postavljajući nove standarde za skalabilnost, pouzdanost i brzinu isporuke video-sadržaja. Međutim, razvoj ovakvih platformi nije trivijalan i zahteva rešavanje složenih problema poput upravljanja velikim količinama podataka, visoke dostupnosti i optimizacije troškova infrastrukture.

Cilj ovog rada jeste da pruži jedno moguće rešenje koje će odgovoriti na navedene probleme. Polazna tačka rešenja je diplomski rad pod naslovom „Platforma za gledanje i deljenje video-sadržaja bazirana na mikroservisnoj arhitekturi“, u okviru kojeg je razvijena osnovna aplikacija za upravljanje video-sadržajima. Ta aplikacija je, u svojoj inicijalnoj verziji, funkcionisala isključivo u lokalnom okruženju.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dušan Gajić, vanr. prof

U ovom radu, inicijalno rešenje je prošireno i nadograđeno kroz primenu AWS (*Amazon Web Services*) servisa tzv. računarstva u oblaku (engl. *cloud computing*) i serverles (engl. *serverless* – bez servera) arhitekture, odnosno arhitekture „bez servera“, što omogućava postavljanje aplikacije u oblak, sa potpunom podrškom za skalabilnost, automatsko upravljanje resursima i troškovnu efikasnost. Kroz integraciju servisa kao što su AWS *Lambda* za izvršavanje bez servera, *S3* za skladištenje, *RDS* za bazu podataka i *CloudFront* za isporuku sadržaja, platforma je transformisana u globalno dostupan sistem koji može da odgovori na izazove modernih video-servisa.

U okviru rešenja koristiće se i drugi AWS servisi kao što su *API Gateway* za upravljanje *API* zahtevima (*HTTP/REST API* i *WebSocket API*), *DynamoDB* za skladištenje podataka koji zahtevaju visoke performanse, *VPC* za mrežnu i bezbednosnu izolaciju resursa, i *Secrets Manager* za sigurno upravljanje osetljivim podacima. Pored toga, primenjuju se AWS CDK za infrastrukturu kao kod (engl. *infrastructure as code*; *IaC*), odnosno definisanje infrastrukture u kodu, kao i *Serverless Framework* za orkestraciju i postavljanje serverles funkcija u oblak.

2. PREGLED FUNKCIONALNOSTI SISTEMA

U ovom poglavlju dat je kratak pregled funkcionalnosti platforme za gledanje i deljenje video-sadržaja implementiranih u diplomskom radu [1], koji je polazna tačka ovog rada. U nastavku su opisani tipovi korisnika u sistemu, kao i same funkcionalnosti.

2.1. Tipovi korisnika

Sistem poznaje četiri tipa korisnika [1]: neregistrovan korisnik, registrovan korisnik, administrator i tehnička podrška.

2.2. Funkcionalnosti

U daljem tekstu su opisane dostupne funkcionalnosti po tipu korisnika u sistemu [1].

Neregistrovan korisnik: registracija na sistem, prijava na sistem, pretraga videa u sistemu i gledanje odn. strimovanje videa [1].

Registrovan korisnik: pretraga videa u sistemu, strimovanje videa, aploadovanje videa na sistem, brisanje sopstvenih videa sa sistema, ocenjivanje videa, pregled prosečne

ocene videa, pregled komentara na videu, postavljanje i brisanje sopstvenih komentara na videu, prijava komentara, prijava videa, pregled i izmena profila, komunikacija sa tehničkom podrškom i odjava sa sistema [1].

Administrator: sve funkcionalnosti registrovanog korisnika osim aploadovanja videa na sistem i komunikacije sa tehničkom podrškom, brisanje videa, brisanje komentara, uvid u prijavljene videe i komentare, kao i brisanje sadržaj uz blokiranje korisnika [1].

Tehnička podrška: prijava na sistem, pregled i izmena profila i komunikacija sa registrovanim korisnicima [1].

3. KORIŠĆENE TEHNOLOGIJE I KONCEPTI

Za proširenje platforme iz [1], kojim se ovaj rad bavi, relevantni su određeni koncepti i tehnologije koje se primenjuju kako bi platforma postala globalno dostupna, skalabilna i troškovno efikasna. Opis tih koncepata i tehnologija dat je u nastavku.

3.1. Računarstvo u oblaku

Računarstvo u oblaku (engl. *cloud computing*) predstavlja model računarskih usluga koji se zasniva na konceptu deljenja računarskih resursa, softvera i informacija na zahtev putem interneta. Kompanije ili pojedinci plaćaju kako bi pristupili virtuelnom fondu deljenih resursa, koji obuhvataju računarsku snagu, skladištenje i mrežne usluge. Ovi resursi su smešteni na udaljenim serverima koje poseduju i održavaju pružaoci *cloud* usluga [2].

3.2. Serverles arhitektura i AWS Lambda

Serverles arhitektura (u bukvalnom prevodu „arhitektura bez servera“) pristup je dizajnu softvera gde programeri mogu da razvijaju i upravljaju aplikacijama bez upravljanja osnovnom infrastrukturom. Uprkos možda zbunjujućem nazivu, serverles aplikacije se i dalje izvršavaju na serverima, ali je provajder odn. pružalac *cloud* usluga odgovoran za održavanje, obezbeđivanje, upravljanje i skaliranje celokupne infrastrukture u oblaku [3].

AWS Lambda je usluga serverles računarstva koju pruža *Amazon Web Services (AWS)*. Korisnici *AWS Lambda* servisa kreiraju funkcije – samostalne aplikacije napisane na jednom od podržanih jezika i okruženja, i otpremaju ih u *AWS Lambda*, koji te funkcije izvršava na efikasan i fleksibilan način [4].

Glavne prednosti *AWS Lambda* servisa u odnosu na održavanje sopstvenih servera u oblaku su: plaćanje po korišćenju, infrastruktura kojom upravlja provajder, automatsko skaliranje (što znači da nema nivoa skaliranja i drugih podešavanja o kojima treba brinuti – funkcije su dostupne kad god se opterećenje poveća ili smanji), kao i laka integracija sa drugim *AWS* servisima [4].

3.3. Amazon API Gateway

Amazon API Gateway je servis koji olakšava programerima da kreiraju, objavljuju, održavaju, nadgledaju i obezbeđuju *API*-je u *AWS* oblaku. Koristeći *API Gateway*, mogu se kreirati *HTTP/REST API*-ji, kao i *WebSocket API*-ji, koji omogućavaju potpuno dvostranu

komunikaciju u realnom vremenu (engl. *real-time*) između servera i klijenata [5].

API Gateway može da se kombinuje sa *AWS Lambda* servisom kako bi se realizovala serverles arhitektura. Kada se kombinuju, *API Gateway* postaje ulazna tačka za *Lambda* funkcije, omogućujući krajnjim korisnicima (ili aplikacijama i servisima) da šalju *HTTP(S)* ili *WebSocket* zahteve ka *Lambda* funkcijama.

3.4. Amazon S3

Amazon S3 (Amazon Simple Storage Service) je servis za skladištenje koji čuva podatke kao objekte unutar baketa (engl. *buckets*). Objekat predstavlja datoteku zajedno sa svim metapodacima koji opisuju tu datoteku. Bucket predstavlja kontejner za objekte [6].

3.5. Amazon RDS

Amazon Relational Database Service (RDS) predstavlja servisnu relacionu bazu podataka koju obezbeđuje *AWS*. *RDS* je zadužen za većinu administrativnih zadataka upravljanja bazom podataka [7].

3.6. Amazon DynamoDB

Amazon DynamoDB je *serverless, NoSQL* servisna baza podataka. *DynamoDB* pruža izuzetno brzo i perzistentno skladište podataka (engl. *datastore*) zasnovano na modelu ključ-vrednost [8].

3.7. Amazon VPC

Amazon Virtual Private Cloud (VPC) je servis koji omogućuje pokretanje *AWS* resursa u logički izolovanoj virtuelnoj mreži. Korisnici imaju potpunu kontrolu nad svojim virtuelnim mrežnim okruženjem, uključujući izbor sopstvenog opsega *IP* adresa, kreiranje podmreža (engl. *subnets*), kao i konfiguraciju tabela rutiranja i mrežnih prolaza (engl. *gateways*) [9].

3.8. Amazon CloudFront

Amazon CloudFront je veb-servis koji ubrzava distribuciju statičkog i dinamičkog veb-sadržaja aplikacije poput *.html*, *.css*, *.js* i slikovnih datoteka krajnjim korisnicima. *CloudFront* takav sadržaj isporučuje putem svetske mreže data centara, koji se nazivaju ivične lokacije (engl. *edge locations*). Kada korisnik zatraži sadržaj koji se servira preko *CloudFront*-a, zahtev se usmerava na ivičnu lokaciju koja pruža najmanju latenciju (odn. vremensko kašnjenje) kako bi se sadržaj isporučio sa što boljim performansama [10].

3.9. AWS Secrets Manager

AWS Secrets Manager je servis koji omogućuje upravljanje, preuzimanje i rotaciju kredencijala za baze podataka i aplikacije, *OAuth* tokena, *API* ključeva i drugih tajnih vrednosti tokom njihovog životnog ciklusa [11].

3.10. AWS IAM

AWS IAM (Identity and Access Management) je servis koji omogućava upravljanje pristupom *AWS* resursima. Osnovne komponente *IAM*-a su: korisnici (engl. *users*), grupe (engl. *groups*), uloge (engl. *roles*) i polise (engl. *policies*) [12].

3.11. AWS CDK

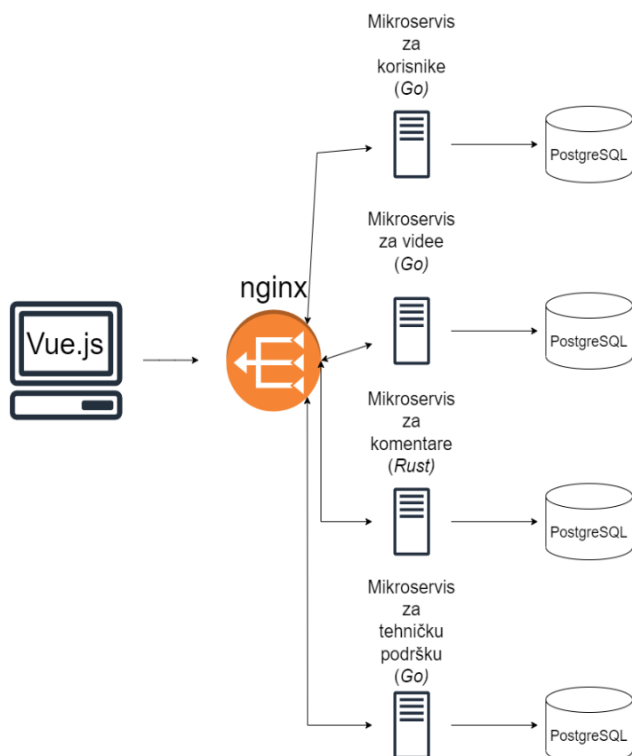
AWS Cloud Development Kit (AWS CDK) je radni okvir otvorenog koda (engl. *open-source framework*) za razvoj softvera koji omogućuje definisanje *cloud* infrastrukture putem koda (*IaC*) i njenu implementaciju preko *AWS CloudFormation* servisa. *CDK* pruža objektno-orijentisane apstrakcije visokog nivoa za definisanje *AWS* resursa i servisa na imperativan način, koristeći prednosti savremenih programskih jezika [13].

3.12. Serverless Framework

Serverless Framework je radni okvir odn. alat komandne linije koji pojednostavljuje postavku serverles aplikacija i njihove *cloud* infrastrukture kroz pristupačnu *YAML* sintaksu (deklarativni *IaC*) [14].

4. REŠENJE

Na slici 1. prikazana je arhitektura sistema u implementaciji platforme za strimovanje video-sadržaja iz [1]. Glavne komponente su mikroservisi za korisnike, videoe, komentare i tehničku podršku, *API Gateway* u vidu *nginx* servera, *PostgreSQL* baze podataka, kao i klijentska *Vue.js* aplikacija.

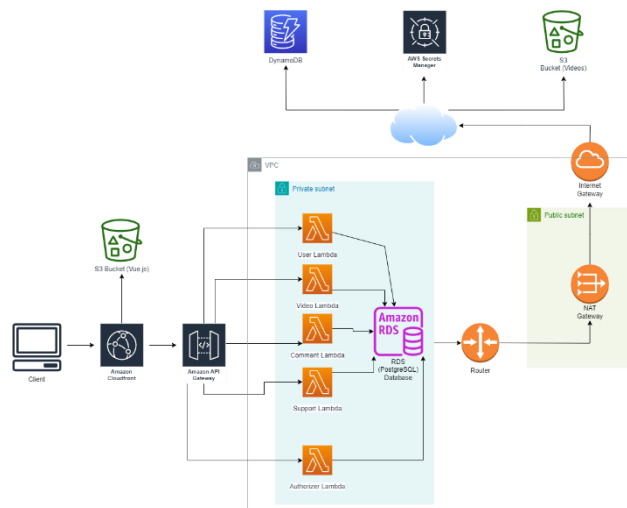


Slika 1. Arhitektura sistema [1]

Na slici 2. može se videti arhitektura nakon proširenja, odnosno arhitektura aplikacije na *AWS* infrastrukturi u oblaku, koja je znatno promenjena u odnosu na prethodno rešenje. Celokupan sistem sada je organizovan oko mikroservisne serverles arhitekture.

Glavne komponente nove arhitekture uključuju:

- **Mikroservisi za korisnike, videoe, komentare i tehničku podršku** – Svaki od ovih mikroservisa sada je implementiran kao *AWS Lambda* funkcija.



Slika 2. Arhitektura sistema u *AWS* oblaku

- ***AWS API Gateway*** – Za razliku od prethodne arhitekture, koja je koristila *nginx*, sada je primenjen *AWS API Gateway* servis, koji služi kao ulazna tačka za sve *HTTP* i *WebSocket* zahteve ka *Lambda* funkcijama. Ovaj servis takođe obezbeđuje rute pomoću *Lambda* funkcije za autorizaciju, koja je napravljena na osnovu mikroservisa za korisnike.
- **Skladištenje video-sadržaja na *S3*** – Umesto čuvanja na lokalnim diskovima, sada se svi video-zapisi (i *thumbnail*-ovi) čuvaju u *S3* baketu, što omogućuje visoku dostupnost i otpornost na kvarove.
- **Baza podataka** – Svi mikroservisi koriste *Amazon RDS* sa *PostgreSQL* bazom podataka, u kojoj postoji po tabela za svaki entitet kojem odgovara dati mikroservis.
- ***Amazon DynamoDB*** – Ovaj *NoSQL* servis se koristi za specifični slučaj upravljanja *WebSocket* konekcijama u mikroservisu za tehničku podršku, gde su potrebne performanse blizu realnom vremenu.
- ***AWS Secrets Manager*** – U ovom servisu se bezbedno čuvaju kredencijali za *RDS* bazu podataka, *API* ključ i tajni ključ za potpisivanje *JWT* tokena.
- ***VPC (Virtual Private Cloud)*** – Sistem je konfigurisan u okviru virtualne privatne mreže (*VPC*) koja obezbeđuje privatnost i bezbednost komunikacije *Lambda* funkcija sa ostalim servisima i sa internetom. Funkcije i *RDS* baza podataka su smešteni u privatne podmreže radi zabrane dolaznog saobraćaja sa interneta, dok se korišćenje usluga koje su izvan *VPC*-a poput *S3*, *Secrets Manager* i *DynamoDB* ostvaruje putem *NAT (Network Address Translation) Gateway*-a.
- ***Amazon CloudFront*** – Za distribuciju *Vue.js* aplikacije koristi se *CloudFront* mreža, koja je brzo i pouzdano isporučuje globalno.

Infrastruktura ovog sistema je definisana i postavljena u *AWS* oblak korišćenjem ranije opisanih *AWS CDK* i *Serverless Framework*-a, i jedne *Makefile* skripte.

4.1. *AWS CDK*

AWS CDK iskorišćen je za definisanje putem koda nekoliko delova infrastrukture u vidu *CDK* stekova (koji na *AWS*-u predstavljaju *CloudFormation* stekove): *DynamoDB*, *RDS*, *VPC*, *WebSocket API* stekovi, kao i stek za *deployment* klijentske aplikacije na *AWS* upotrebom *S3* i *CloudFront* servisa. Stekovi su definisani u programskom jeziku *TypeScript*.

Nakon što su definisani, stekovi su postavljeni na *AWS* jednostavnim pozivom komande u komandnoj liniji.

4.2. *Serverless Framework*

Upotrebom *Serverless Framework*-a, odnosno njegovog *YAML* konfiguracionog fajla (*serverless.yml*), definisane su *AWS Lambda* funkcije, njihova konfiguracija, varijable okruženja na osnovu izlaza iz *CloudFormation* stekova (koji postanu dostupni kad se završi *deployment CDK* stekova), uloga i dozvole (permisije), događaji (*HTTP* i *WebSocket*), autorizacija, *CORS* dozvole i slično. Uz *Lambda* funkcije, definisan je i sam *AWS API Gateway*, kao i njegov *API* ključ.

4.3. *Makefile* skripta

Napisana je *Makefile* skripta koja takođe igra ulogu u orkestraciji infrastrukture sistema. Naime, skripta najpre generiše novi tajni ključ za potpisivanje *JWT* tokena i čuva ga u *AWS Secrets Manager*-u upotrebom alata komandne linije za rad sa *AWS*-om: *AWS CLI*. Zatim kreira *S3* baket za video-sadržaj, nakon čega bilduje (engl. *build*) *Lambda* funkcije definisane u *serverless.yml*. Na kraju pokreće komandu za *deployment* konfiguracije *Serverless Framework*-a, što je poslednji korak u postavljanju čitave infrastrukture sistema u *AWS* oblak.

5. ZAKLJUČAK

Ovaj rad predstavlja dalji razvoj platforme za strimovanje video-sadržaja, čija je osnova postavljena u okviru diplomskog rada zasnovanog na mikroservisnoj arhitekturi. Dok je diplomski rad obrađivao mikroservisni pristup u lokalnom okruženju, ovaj rad se fokusira na primenu serverles arhitekture u *AWS* oblaku kao sledećeg koraka u razvoju sistema.

Uvođenje *AWS* servisa, kao što su *Lambda*, *API Gateway* i *S3*, rezultiralo je mogućnošću automatskog prilagođavanja kapaciteta sistema u zavisnosti od opterećenja, čime je sistem postao otporniji na velike fluktuacije u broju korisnika i zahteva. Ovaj pristup eliminiše potrebu za ručnim upravljanjem serverima, što zahteva dodatne resurse i vreme, dok serverles arhitektura omogućava plaćanje isključivo za korišćene resurse, čime se značajno smanjuju troškovi na duži rok u većini primena.

Pored toga, ovakav sistem je jednostavniji za proširivanje novim funkcionalnostima, jer serverles arhitektura podstiče upotrebu malih, nezavisnih komponenti koje se mogu lako integrisati u postojeći sistem bez značajnih izmena. Stoga uvođenje novih funkcija, poput editovanja video sadržaja i optimizacije video-zapisa upotrebom

kompresionih algoritama, postaje mnogo efikasnije i lakše za implementaciju, što ujedno predstavlja mogući dalji korak u unapređenju platforme.

6. LITERATURA

- [1] Milovan Milovanović, „Platforma za gledanje i deljenje video-sadržaja bazirana na mikroservisnoj arhitekturi“, 2022. , poslednji pristup 27.9.2024.
- [2] <https://cloud.google.com/learn/what-is-cloud-computing>, poslednji pristup 27.9.2024.
- [3] <https://cloud.google.com/discover/what-is-serverless-architecture>, poslednji pristup 27.9.2024.
- [4] <https://www.serverless.com/aws-lambda>, poslednji pristup 27.9.2024.
- [5] <https://aws.amazon.com/api-gateway/>, poslednji pristup 27.9.2024.
- [6] <https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html>, poslednji pristup 27.9.2024.
- [7] <https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Welcome.html>, poslednji pristup 27.9.2024.
- [8] https://en.wikipedia.org/wiki/Amazon_DynamoDB, poslednji pristup 27.9.2024.
- [9] <https://aws.amazon.com/vpc/features/>, poslednji pristup 27.9.2024.
- [10] <https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/Introduction.html>, poslednji pristup 27.9.2024.
- [11] <https://docs.aws.amazon.com/secretsmanager/latest/userguide/intro.html>, poslednji pristup 27.9.2024.
- [12] <https://www.datacamp.com/tutorial/aws-identity-and-access-management-iam-guide>, poslednji pristup 27.9.2024.
- [13] <https://github.com/aws/aws-cdk>, poslednji pristup 27.9.2024.
- [14] <https://github.com/serverless/serverless>, poslednji pristup 27.9.2024.

Kratka biografija:



Milovan Milovanović rođen je u Novom Sadu 1999. god. Diplomski rad na Fakultetu tehničkih nauka u Novom Sadu iz oblasti Elektrotehnike i računarstva – Softversko inženjerstvo i informacione tehnologije odbranio je 2022. god.

kontakt: milovanovicm309@gmail.com