Зборник радова Факултета техничких наука, Нови Сад

UDK: 4.41
DOI: https://doi.org/10.24867/30BE04Cosovic

CLOUD АПЛИКАЦИЈА ЗА ПРЕТРАГУ ФИЛМОВА БАЗИРАНА НА OPENSEARCH-У

CLOUD BASED APPLICATION FOR MOVIE SEARCH USING OPENSEARCH

Матеја Ћосовић, *Факултет техничких наука, Нови Сад*

Област – ПРИМЕЊЕНЕ РАЧУНАРСКЕ НАУКЕ И ИНФОРМАТИКА

Кратак садржај – *Рад се бави развојем апликације за претрагу филмова користећи AWS сервисе од којих је најзначајнији OpenSearch Service. У раду је дат теоријски опис свих коришћених AWS сервиси, програмских језика, радних оквира и алата. Такође, детаљно је описана архитектура, дизајн и модел података апликације, као и испуњени функционални и нефункционални захтеви. На крају су приказана три експеримента над апликацијом како би се измерile перформансе.*

Кључне речи: *AWS, cloud, филмови, serverless, OpenSearch, AWS CDK, Lambda*

Abstract – *The paper focuses on the development of a movie search application using AWS services, with the most significant being OpenSearch Service. The paper provides a theoretical description of all AWS services, programming languages, frameworks, and tools used during development. Additionally, the architecture, design, and data model of the application are described in detail, along with the fulfilled functional and non-functional requirements. Finally, three experiments on the application are presented to measure its performance.*

Keywords: *AWS, cloud, movies, serverless, OpenSearch, AWS CDK, Lambda*

1. УВОД

Љубав према филмовима је универзална и спаја људе широм света. Први филмови настали су пре више од сто година и снимани су широм земаљске кугле [1]. Број филмова експоненцијално расте из године у годину и гледаоцима је веома тешко пронаћи баш онај филм који им одговара у датом тренутку. Решавање овог проблема представља мотивацију која стоји иза овог рада.

Cloud і *serverless* технологије постају све популарније због својих предности у односу на традиционалне технологије које користе локалне ресурсе [2]. Пружају већу скалабилност, смањују трошкове одржавања и омогућавају брже развијање и имплементацију апликација. Амазонов AWS (*Amazon Web Services*) као лидер у клауд технологијама [3] нуди широк спектар *serverless* услуга које олакшавају развој апликација.

У раду ће детаљно бити описани кораци развоја апликације, укључујући дизајн система, избор технологија и имплементација, као и анализу перформанси решења.

Циљ је да се покаже како се *serverless* технологија може искористити за имплементацију ефикасног система за претрагу филмова.

2. ТЕОРИЈСКЕ ОСНОВЕ

2.1. Увод у *cloud* технологије

Клауд (енгл. *cloud*) технологије, или технологије у облаку, представљају приступ рачунарству где се ресурси као што су складиште података, процесорска снага и апликације не налазе локално на рачунару корисника, већ су доступни путем интернета.

Овај приступ омогућава корисницима да приступају ресурсима са било ког уређаја повезаног на интернет, чиме се побољшава флексибилност, скалабилност и ефикасност пословања [4].

Серверлес (енгл. *serverless*) технологије су технологије где пружалац клауд услуга аутоматски управља инфраструктуром потребном за покретање апликација, тако да корисници могу да се фокусирају на писање и имплементацију кода без бриге о управљању инфраструктуром.

Коришћење клауд и серверлес услуга доноси бројне предности које значајно унапређују ефикасност и конкурентност организација [5]. Првенствено, смањују се трошкови јер корисници плаћају само за ресурсе које користе, без потребе за великим почетним улагањима у хардвер и софтвер.

Коришћење клауд платформи омогућава брзу имплементацију нових технологија и сервиса, подржавајући иновације. Поред тога, клауд провајдери често користе енергетски ефикасне дата центре, чиме доприносе еколошкој одрживости [6].

2.2. AWS платформа

Amazon Web Services (AWS) је водећа платформа за клауд рачунарство, која пружа широк спектар услуга као што су процесирање, базе података, аналитику и многе друге. Основан 2006. године од стране Amazona, AWS омогућава организацијама свих величина да ефикасно управљају својим IT ресурсима због његове скалабилности, флексибилности и поузданости и постаје кључни алат за компаније [7].

2.3. Amazon Lambda

Amazon Lambda је серверлес рачунска услуга која омогућава корисницима да покрећу код као одговор на догађаје, без потребе за управљањем серверима.

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био др Драган Ивановић, ред. проф.

Ова услуга омогућава програмерима да се фокусирају на писање и оптимизацију кода, док се Amazon Lambda брине о основној инфраструктури. Код се извршава у оквиру високо доступног окружења, које аутоматски управља ресурсима потребним за покретање функција [8].
Једна од главних предности Lambda-е је способност да аутоматски скалира функције на основу броја долазних догађаја. Када дође до повећања оптерећења, Lambda аутоматски креира додатне инстанце како би обрадила повећани број захтева, а када оптерећење опадне, смањује се број инстанци.
Amazon Lambda представља моћан алат за модерну развојну праксу, омогућавајући брзу имплементацију, флексибилност и значајне уштеде. Њена способност аутоматског скалирања, интеграција са другим AWS услугама и робусна сигурносна архитектура чине је идеалним избором за широк спектар апликација, од једноставних до комплексних.

2.4. Amazon OpenSearch Service

Amazon OpenSearch Service је сервис који омогућава постављање, управљање и скалирање ElasticSearch тј. OpenSearch кластера на AWS инфраструктури. Намењен је за кориснике који желе моћне могућности претраге и анализе података, али желе да избегну сложености ручног управљања инфраструктурама и хостовања кластера самостално [9].
Amazon OpenSearch Service је дизајниран за високе перформансе, омогућавајући брзе претраге и анализе, чак и на великим скуповима података. Поред тога, модел наплате је заснован на стварној употреби ресурса, што омогућава оптимизацију трошкова. Корисници плаћају само за ресурсе које користе, укључујући инстанце, складиште и улазно-излазне операције, чиме се избегавају непотребни трошкови за неискоришћене ресурсе.

2.5. Amazon CloudFormation

Amazon CloudFormation је сервис за управљање инфраструктурама преко кода (Infrastructure as Code, IaC) који омогућава корисницима да дефинишу и аутоматски проводе ресурсе потребне за своје апликације на AWS-у. Коришћењем једноставних текстуалних датотека у JSON или YAML формату, корисници могу описати све AWS ресурсе које им апликација захтева.
CloudFormation аутоматски управља заузимањем и конфигурисањем ресурса, што смањује сложеност и ризик од људских грешака.
Једна од главних предности CloudFormation-а јесте конзистентност и поновљивост у постављању инфраструктуре. Када је инфраструктура дефинисана као код, лако се може делити, прегледати и контролисати верзије, што омогућава развојним тимовима да сарађују ефикасније [11].
Поред тога, промена инфраструктуре постаје једноставнија и мање ризична, јер се све промене могу тестирати и валидирати пре него што се примене у производном окружењу.
Ово доводи до бржег циклуса развоја и веће поузданости апликација.

2.6. Amazon CDK

AWS Cloud Development Kit (CDK) је *open-source* радни оквир који омогућава програмерима да дефинишу и заузму своју AWS инфраструктуру користећи познате програмске језике [12]. CDK омогућава писање инфраструктурног кода на исти начин као и апликативног кода, чиме се поједностављује процес развоја и управљања ресурсима у клауду.

3. СПЕЦИФИКАЦИЈА АПЛИКАЦИЈЕ

3.1. Функционални захтеви

Функционални захтеви за апликацију су специфичне карактеристике и функције које систем мора да испуњава како би задовољио потребе корисника и пословне циљеве. Ови захтеви дефинишу шта систем треба да ради, укључујући корисничке интеракције, обраду података, управљање подацима и друге оперативне функције [13].
Главна функционалност апликације за претрагу филмова је, може се наслутити, напредна претрага и филтрирање филмова над великим скупом података. Такође, постоје додатне функционалности попут напредне статистике, уноса појединачног филма и великог броја филмова, логовања и регистрација, као и брисања филмова.

3.2. Нефункционални захтеви

Нефункционални захтеви за апликацију су критеријуми који дефинишу како систем треба да функционише, а не шта систем треба да ради. Ови захтеви обухватају аспекте као што су перформансе, безбедност, употребљивост, поузданост, скалабилност и одрживост система. Одабрани нефункционални захтеви за апликацију су перформансе, безбедност и скалабилност.

3.3. Архитектура система

Архитектура апликације састоји се од клијентског и серверског дела. У оба дела коришћени су различити AWS сервиси. За израду клијентског дела коришћени су Amazon CloudFront и Amazon S3 и React радни оквир, док су за израду серверског дела коришћени Amazon Lambda, Amazon API Gateway, Amazon DynamoDB, Amazon SQS, Amazon S3, Amazon OpenSearch Service и Python програмски језик. Цела архитектура је закупљена уз помоћ AWS CDK радног оквира. На слици 3.2. приказана је архитектура система.

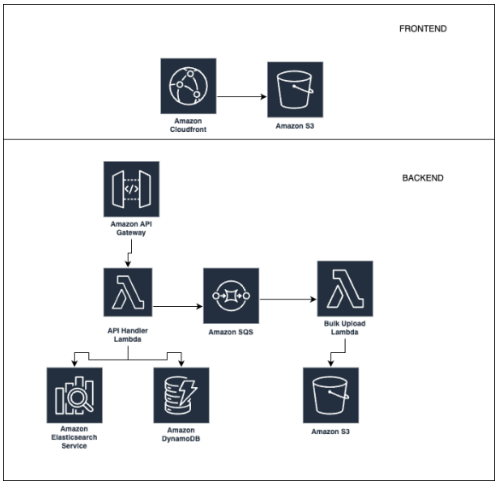
3.4. Дизајн система

За израду визуелно привлачног и респонзивног корисничког интерфејса коришћена је популарна MUI Material библиотека компоненти за React која пружа предефинисане стилске компоненте.
Апликација се састоји из следећих страница: Login, Registration, Search, Statistics и Upload.

3.5. Модел података

Модел података направљен је на основу случајева коришћења и представља структуру података који се

налазе у DynamoDB табели и OpenSearch домену. Ентитети система су корисник и филм.



Слика 3.2. Архитектура система

Како би што лакше унели велики број филмова у систем, а да притом филмови имају стварне вредности, одлучено је да се користи неки постојећи скуп података о филмовима са интернета. Изабран је *Full TMDB Movies Dataset 2024 (1M Movies)* са *Kaggle* интернет платформе.

4. ИМПЛЕМЕНТАЦИЈА РЕШЕЊА

4.1. Имплементација серверског дела апликације

Серверски део апликације написан је у Python програмском језику, конкретно користећи FastAPI радни оквир.

За организовање виртуалног окружења, изградњу пројекта за дистрибуцију и инсталацију пакета и зависности коришћен је poetry алат. Овај алат омогућава једноставно инсталирање свих потребних пакета, као и изградњу пројекта кроз покретање команди у терминалу и веома је лак за коришћење.

Апликација је организована по модулима, где је сваки модул задужен за одређене делове функционалности система и овај приступ организовања кода је традиционално виђен у пројектима заснованим у Python-у.

4.2. Имплементација клијентског дела апликације

Клијентска страна апликације развијена је користећи популарни React радни оквир. Према основном принципу React-а, примењивана је компонентна архитектура, односно апликација је подељена на компоненте како би мањи делови апликације били издвојени и лаки за одржавање.

4.3. Имплементација инфраструктуре апликације

За имплементацију инфраструктуре апликације је коришћен AWS Cloud Development Kit (CDK), моћан алат који омогућава програмерима да дефинишу cloud инфраструктуру користећи програмске језике.

У нашем случају, коришћен је Python програмски језик.

5. ЕКСПЕРИМЕНТИ И ПРИКАЗ РЕЗУЛТАТА

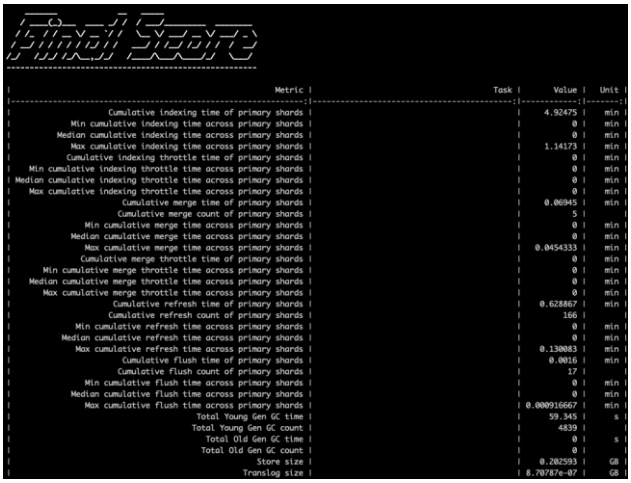
5.1. Први експеримент

У оквиру првог експеримента предложено је да се тестирају перформансе над различитим типовима инстанци и различитим бројем дата чворова. Конкретно, у раду су коришћене комбинације t3.small.search и t3.medium.search инстанци респективно са једним или пет чворова. Над сваком комбинацијом биће испробано индексирање од двеста хиљада докумената, затим претрага над њима, као и набављање детаљне статистике.

Забележени резултати слажу се са генералним претпоставкама о OpenSearch-у, броју дата чворова и типовима инстанци. Из резултата се може јасно видети да је претрага најбржа у случају 5 дата чворова t3.medium.search типа и просечно време извршавања износи 1.1 секунди, док је претрага најспорија случају једног чвора t3.small.search типа где просечно време износи 2.4 секунде.

5.2. Други експеримент

У оквиру другог експеримента предложено је да се над OpenSearch доменом покрене алат под именом OpenSearch Benchmark. Овај алат извршава тестирање над постојећим доменом са предефинисаним скупом података и прикупља детаљне информације о домену и његовим перформансама. На слици 5.1. може се видети резултат OpenSearch Benchmark-а.



Metric	Task	Value	Unit
Cumulative indexing time of primary shards		4.92475	min
Min cumulative indexing time across primary shards		0	min
Median cumulative indexing time across primary shards		0	min
Max cumulative indexing time across primary shards		1.14173	min
Cumulative indexing throttle time of primary shards		0	min
Min cumulative indexing throttle time across primary shards		0	min
Median cumulative indexing throttle time across primary shards		0	min
Max cumulative indexing throttle time across primary shards		0	min
Cumulative merge time of primary shards		0.00945	min
Cumulative merge count of primary shards		5	
Min cumulative merge time across primary shards		0	min
Median cumulative merge time across primary shards		0	min
Max cumulative merge time across primary shards		0.0454333	min
Cumulative merge throttle time of primary shards		0	min
Min cumulative merge throttle time across primary shards		0	min
Median cumulative merge throttle time across primary shards		0	min
Max cumulative merge throttle time across primary shards		0	min
Cumulative refresh time of primary shards		0.628867	min
Cumulative refresh count of primary shards		166	
Min cumulative refresh time across primary shards		0	min
Median cumulative refresh time across primary shards		0	min
Max cumulative refresh time across primary shards		0.130083	min
Cumulative flush time of primary shards		0.0016	min
Cumulative flush count of primary shards		17	
Min cumulative flush time across primary shards		0	min
Median cumulative flush time across primary shards		0	min
Max cumulative flush time across primary shards		0.000916667	min
Total Young Gen GC time		59.345	s
Total Young Gen GC count		4839	
Total Old Gen GC time		0	s
Total Old Gen GC count		0	
Store size		0.202593	GB
Translog size		8.70787e-07	GB

Слика 5.1. Резултат OpenSearch Benchmark-а

5.3. Трећи експеримент

У оквиру трећег експеримента предложено је да се прикаже како количина података утиче на перформансе. На табели 5.1. могу се видети резултати трећег експеримента.

Резултати забележени у трећем експерименту показују да што више података има у домену, то је спорије време извршавања упита, што је била и претпоставка пре извршавања експеримента.

Табела 5.1. Резултати трећег експеримента

Број докумената	**Претрага**	**Детаљна статистика**
10000 докумената	0.84 секунди	0.63 секунди
20000 докумената	0.78 секунди	0.59 секунди
50000 докумената	1.29 секунди	0.5 секунди
100000 докумената	2.3 секунди	0.62 секунди

6. ЗАКЉУЧАК

У раду је представљена cloud апликација за претрагу филмова која користи OpenSearch. Теоријски су описани све коришћене технологије и сервиси како би се разумело функционисање апликације.
Представљени су функционални и нефункционални захтеви које је апликације требало да задовољи и описани су имплементациони детаљи свих делова апликације. На крају је извршено три експеримента како би се упоредиле перформансе могућих верзија апликације.
Спроведени експерименти показују да је апликација успешно имплементирана, и осликавају њене перформансе. Такође, апликација је лако испоручива у било којем тренутку, то јест, може се наћи на интернету у веома кратком временском периоду. Једна од ствари о којој би се требало побринути у случају надоградње апликације је резервисање интернет домена путем AWS-овог Route53 сервиса. Додатно, скуп података би се могао проширити да укључује више од милион филмова, и претрага по више параметара него тренутно би могла бити омогућена. На крају, перформансе система би се могле побољшати тако што бисмо вертикално скалирали апликацију.

7. ЛИТЕРАТУРА

[1] Ishita Babbar. International Journal for Multidisciplinary Research (IJFMR) (2024). *Evolution of Cinema*, https://www.ijfmr.com/papers/2024/2/17578.pdf

[2] Zhang, Q., Cheng, L. & Boutaba, R. *Cloud computing: state-of-the-art and research challenges. J Internet Serv Appl* **1**, 7–18 (2010). https://doi.org/10.1007/s13174-010-0007-6

[3] Bandaru, Avinash. (2020). *Amazon Web Service*, https://www.researchgate.net/publication/347442916_AMAZON_WEB_SERVICES

[4] Rafia Islam, Vardhan Patamsetti, Aparna Gadhi, Ragha Madhavi Gondu, Chinna Manikanta Bandaru, Sai Chaitanya Kesani, Olatunde Abiona
Department of Computer Information Systems, Indiana University Northwest, (2023), *The Future of Cloud Computing: Benefits and Challenges* 10.4236/ijcns.2023.164004

[5] Sether, Ayob. (2016). *Cloud Computing Benefits* 10.13140/RG.2.1.1776.0880.

[6] Mittal, Sparsh. (2014). *Power Management Techniques for Data Centers: A Survey*

[7] Mukherjee, Sourav. (2019). *Benefits of AWS in Modern Cloud*. 10.5281/zenodo.2587217.

[8] Shashank Srivastava, et al. (2023). Execution of Serverless Functions Lambda in AWS Serverless Environment. *International Journal on Recent and Innovation Trends in Computing and Communication*, *11*(9), 1081–1086. https://doi.org/10.17762/ijritcc.v11i9.9014

[9] "What is Amazon OpenSearch Service", AWS Documentation, https://docs.aws.amazon.com/opensearch-service/latest/developerguide/what-is.html

[10] Chauhan, Sidhartha & Cuthbert, Dave & Devine, James & Halachmi, Alan & Lehwess, Matt & Matthews, Nick & Morad, Steve & Seymour, Steve & Walker, Dave. (2018). Amazon CloudFront. 10.1002/9781119549000.ch7.

[11] "5 Benefits of Infrastructure as Code (IaC) for Modern Businesses in the Cloud", Technology Advisors, https://www.techadv.com/blog/5-benefits-infrastructure-code-iac-modern-businesses-cloud

[12] "What is the AWS CDK?", AWS Documentation, https://docs.aws.amazon.com/cdk/v2/guide/home.html

[13] Malan, Ruth & Bredemeyer, Dana. (2001). Functional Requirements and Use Cases.

Кратка биографија:

Матеја Ћосовић рођен је 1999. године у Сомбору. У школској 2018/19. уписује студијски програм Софтверско инжењерство и информационе технологије на Факултету техничких наука у Новом Саду. Основне студије завршио је 2022. године. Након завршених основних студија, уписује мастер студије на истом студијском програму.
контакт:
matejacosovic@gmail.com