

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 3134

**GENERIČKA PLATFORMA U OBLAKU ZA OBJAVU
INFORMACIJA I OGLASA**

Zdravko Pandžić

Zagreb, lipanj 2023.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 3134

**GENERIČKA PLATFORMA U OBLAKU ZA OBJAVU
INFORMACIJA I OGLASA**

Zdravko Pandžić

Zagreb, lipanj 2023.

DIPLOMSKI ZADATAK br. 3134

Pristupnik: **Zdravko Pandžić (0036512112)**
Studij: Računarstvo
Profil: Programsko inženjerstvo i informacijski sustavi
Mentor: izv. prof. dr. sc. Boris Milašinović

Zadatak: **Generička platforma u oblaku za objavu informacija i oglasa**

Opis zadatka:

Tipična funkcionalnost portala namijenjenih za objavu korisnih informacija i informacija o događajima svodi se na mogućnost objave događaja i notifikaciju zainteresiranih korisnika, pri čemu nije neuobičajeno da takav portal ima određene elemente oglasnika. Takav koncept je primjenjiv na različite udruge, manja sportska udruženja, klubove obožavatelja, poljoprivredne zadruge i slično, a razlika će vjerojatno biti u vizualnom izgledu te kategorijama događaja i oglašanih proizvoda ili usluga. Ideja ovog rada je napraviti programsko rješenje u oblaku koje će predstavljati generičku platformu za uspostavu takvih informativnih portala s elementima oglasnika. Zainteresirani klijent bi se uz odgovarajući pretplatni model mogao registrirati na platformi, kreirati i administrirati vlastiti portal. Krajnji korisnik bi osim pregleda sadržaja portala mogao definirati postavke na temelju kojih bi dobivao obavijesti prilikom objave novih sadržaja. Rješenje je potrebno oblikovati korištenjem mikroservisa i sabirnice poruka, što bi omogućilo da se prilikom objave sadržaja aktiviraju različiti servisi, primjerice servis za automatsku provjeru sadržaja, servis za notifikacije i slično.

Rok za predaju rada: 23. lipnja 2023.

Zahvaljujem Bogu za snagu koju mi je dao, obitelji i djevojci za njihovu stalnu podršku, kao i prijateljima i rodbini koji su bili uz mene kroz svaki dio ovog putovanja.

Sadržaj

1.	Uvod	1
2.	Pregled sličnih rješenja	2
2.1.	Kriteriji za odabir sličnih rješenja.....	2
2.2.	Njuškalo	2
2.3.	Eventim.....	2
2.4.	MojPosao	3
2.5.	Usporedba s predloženom platformom.....	3
3.	Specifikacija programske potpore	4
3.1.	Funkcionalni zahtjevi.....	4
3.2.	Nefunkcionalni zahtjevi.....	5
4.	Arhitektura rješenja.....	6
4.1.	Model podataka	7
4.2.	Sekvencijski dijagram.....	9
5.	Implementacija.....	11
5.1.	Korištene tehnologije.....	11
5.1.1.	PostgreSQL.....	11
5.1.2.	Node.js	12
5.1.3.	React	13
5.1.4.	AWS (engl. <i>Amazon Web Services</i>).....	14
5.2.	Baza podataka	16
5.2.1.	RDS (engl. <i>Relational Database Service</i>)	16
5.2.2.	Struktura baze podataka.....	17
5.3.	Poslužitelj	20
5.3.1.	Struktura direktorija poslužitelja	20
5.3.2.	Programski kodovi.....	21
5.4.	AWS SQS	25
5.5.	AWS Lambda	26
5.5.1.	Kod Lambda funkcije	27
5.6.	Klijent	30
5.6.1.	Struktura klijentskog direktorija	30
5.6.2.	Programski kodovi.....	31

6.	Korisničke upute	34
6.1.	Prijava i registracija	36
6.2.	Kategorije	37
6.2.1.	Stranica kategorije i potkategorija	38
6.2.2.	Filtriranje i pregled objava.....	39
6.2.3.	Uređivanje, kreiranje i brisanje kategorije.....	40
6.3.	Moje objave	42
6.4.	Moji filteri.....	42
6.4.1.	Spremanje i Pregled Filtera.....	43
6.4.2.	Uređivanje i brisanje filtera	44
6.5.	Kreiranje nove objave i primanje obavijesti	45
6.6.	Prikaz portala	47
7.	Upute za instalaciju.....	48
7.1.	Kloniranje repozitorija.....	48
7.2.	Instalacija Docker-a	48
7.3.	Pokretanje aplikacije.....	48
7.4.	Pristup aplikaciji	48
	Zaključak	49
	Literatura.....	50
	Sažetak	51
	Summary.....	52

1. Uvod

U eri digitalizacije i brzog protoka informacija, postoji velika potreba za platformama koje mogu pružiti relevantne informacije i oglase zainteresiranim korisnicima. Iako postoje pojedinačne platforme koje ispunjavaju ove potrebe, postoji nedostatak sveobuhvatnog, prilagodljivog rješenja koje se može lako prilagoditi različitim potrebama i područjima interesa.

Ovaj rad predstavlja ideju o generičkoj platformi za objavu korisnih informacija i oglasa. Rješenje će kombinirati klasične karakteristike informativnih portala s elementima oglasnika, nudeći prilagodljivo rješenje koje će udovoljiti širokom spektru korisnika. Platforma je osmišljena tako da koristi mikroservise i sabirnice poruka, omogućujući tako fleksibilnost i skalabilnost, kao i uključivanje različitih dodatnih servisa i funkcionalnosti.

U okviru rada, analizirane su postojeće platforme koje se bave sličnim problemima, uključujući njihove prednosti i ograničenja. Na temelju ove analize, predložena je nova arhitektura koja nudi širi spektar mogućnosti i koja stavlja korisnika u središte iskustva, omogućujući mu da sam odredi što želi pretraživati i o kojim temama želi biti obaviješten.

Rad je organiziran u sedam ključnih poglavlja koja zajedno čine koherentnu strukturu. Drugo poglavlje fokusira se na pregled sličnih rješenja, odabirući ih po specifičnim kriterijima i analizirajući njihove prednosti i nedostatke. Treće poglavlje, koje se bavi specifikacijom programske potpore, razrađuje funkcionalne i nefunkcionalne zahtjeve predloženog rješenja. Četvrto poglavlje posvećeno je arhitekturi rješenja, uključujući model podataka i sekvencijske dijagrame. Peto poglavlje, koje se bavi implementacijom, detaljno opisuje korištene tehnologije i strukturu baze podataka, kao i poslužiteljske i klijentske komponente. Šesto poglavlje pruža korisničke upute, dok je sedmo poglavlje posvećeno uputama za instalaciju. Rad završava zaključkom koji sumira ključne nalaze i prednosti predloženog rješenja.

2. Pregled sličnih rješenja

Ovo poglavlje posvećeno je analizi i usporedbi platformi koje služe za objavu informacija i oglasa. Analiza je usmjerena na identifikaciju karakteristika koje ove platforme čine korisnima, kao i na ograničenja koja postavljaju.

2.1. Kriteriji za odabir sličnih rješenja

U procesu odabira platformi za usporedbu korišteni su sljedeći kriteriji:

- Mogućnost objave različitih vrsta informacija ili oglasa
- Postojanje sustava za filtriranje i pretragu
- Sustav notifikacija i obavijesti za korisnike

2.2. Njuškalo

Njuškalo je online platforma za oglase koja dominira tržištem unutar Hrvatske. Primarni fokus platforme je na objavama koje se tiču prodaje i potražnje različitih vrsta artikala i dobara.

- **Prednosti:** Jedna od glavnih prednosti Njuškala je njegova popularnost i široka upotreba u Hrvatskoj, što je rezultat ranog ulaska na tržište. Osim toga, platforma je intuitivna i prilagođena korisnicima.
- **Nedostaci:** Među ograničenjima Njuškala je što notifikacije o novim objavama dolaze jednom dnevno. Osim toga, platforma je isključivo orijentirana na prodaju i potražnju, bez mogućnosti objave drugih vrsta informacija.

2.3. Eventim

Eventim je platforma koja se fokusira na praćenje i objavu događaja. Osim informacija o događajima, platforma također nudi mogućnost online kupnje ulaznica.

- **Prednosti:** Glavna prednost Eventima je opcija za kupnju ulaznica, što korisnicima olakšava sudjelovanje u događajima. Platforma pruža pregled i informacije o različitim vrstama događaja, uključujući kulturu, glazbu i sport.

- **Nedostaci:** Platforma Eventim ne pruža kompletan popis dostupnih događaja, što je dijelom posljedica činjenice da samo administratori imaju mogućnost objavljivanja. Ovo ograničenje, zajedno s nedostatkom sistema za korisničke objave i notifikacije, smanjuje dinamičnost platforme i sužava spektar informacija koje su korisnicima dostupne.

2.4. MojPosao

MojPosao je platforma za objavu i pretragu poslovnih oglasa. Nudi napredne opcije filtriranja i pretrage, kao i mogućnost personalizacije kriterija za obavijesti o novim oglasima.

- **Prednosti:** Jedna od ključnih prednosti ove platforme je visoka razina prilagodljivosti i personalizacije, što korisnicima omogućuje da prilagode pretragu svojim specifičnim potrebama. Sistem notifikacija je također dobro dizajniran, pružajući korisnicima instant obavijesti o novim oglasima koji odgovaraju njihovim kriterijima.
- **Nedostaci:** Glavno ograničenje ove platforme je orijentiranost na poslovne oglase tj. ne pruža mogućnost objave ili pretrage informacija iz drugih kategorija.

2.5. Usporedba s predloženom platformom

Predložena platforma nastoji riješiti niz problema i nedostataka koji su identificirani u analiziranim platformama. Dizajnirana je da bude sveobuhvatna i prilagodljiva, pružajući korisnicima mogućnost objave i pretrage informacija i oglasa iz različitih kategorija. Sistem notifikacija je također optimiziran da pruži instant obavijesti, što je prednost u odnosu na druge analizirane platforme. Jedna od izrazitih prednosti predložene platforme je korisnička kontrola. Korisnici imaju slobodu da sami odrede što žele pretraživati i o kojim će temama biti obaviješteni, umjesto da su obavijesti i sadržaj nametnuti. Ovo pridonosi većem korisničkom zadovoljstvu i učinkovitosti platforme.

Na temelju analize, može se zaključiti da postoji prostor za platformu koja bi ponudila širi spektar mogućnosti za objavu i pretragu informacija i oglasa. Predložena platforma ima potencijal ispuniti ovu prazninu na tržištu, nudeći napredne opcije koje nisu dostupne u trenutnim rješenjima.

3. Specifikacija programske potpore

3.1. Funkcionalni zahtjevi

Postoje tri uloge: prijavljeni korisnik, neprijavljeni korisnik te administrator.

Osnovne funkcionalnosti (za sve korisnike):

1. Pregled kategorija:

- pregledavanje dostupnih kategorija unutar aplikacije

2. Pregled objava:

- pregledavanje svih objava unutar odabrane kategorije

3. Filtriranje objava:

- temeljene na odabranoj kategoriji
- temeljene na vrijednostima pojedinih polja povezanih s kategorijama

Neprijavljeni korisnik:

1. Prijava i registracija:

- pristup stranici za prijavu
- pristup formi za registraciju
- prelazak u ulogu „prijavljenog korisnika“ nakon uspješne registracije ili prijave

Prijavljeni korisnik:

1. Upravljanje objavama:

- kreiranje novih objava
- brisanje vlastitih objava

2. Upravljanje filterima za obavijesti:

- definiranje i spremanje filtera kako bi se primale obavijesti o objavama koje odgovaraju postavljenim kriterijima
- pregled, uređivanje i brisanje spremljenih filtera

3. Obavijesti:

- primanje obavijesti o novim objavama koje odgovaraju postavljenim filterima

4. Portal sa spremljenim filterima:

- pristup vlastitom portalu na kojem može pregledavati obavijesti spremljenih filtera

5. Odjava:

- Odjava iz sustava

Administrator:

1. Upravljanje kategorijama:

- kreiranje nove kategorije
- uređivanje postojeće kategorije
- brisanje kategorije

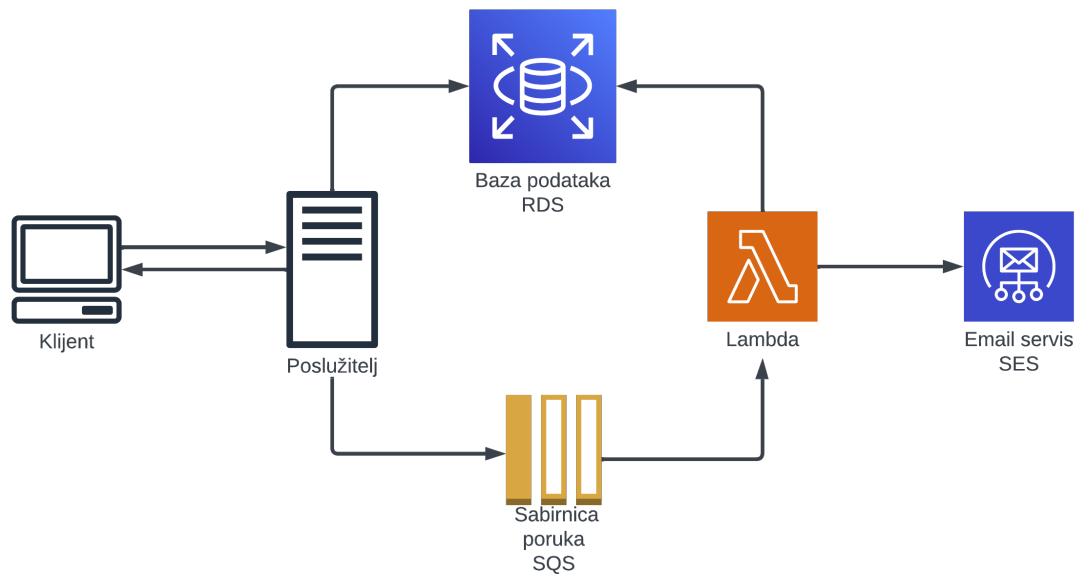
2. Upravljanje poljima:

- kreiranje novih polja unutar odabrane kategorije
- uređivanje postojećih polja povezana s kategorijama
- brisanje polja iz kategorija

3.2. Nefunkcionalni zahtjevi

Korisnici unutar sustava mogu kreirati i pregledavati objave prema određenim kategorijama. Osim korisnika, obavijesti o objavama automatski se šalju korisnicima koji su definirali odgovarajuće filtere. Rješenje mora biti izvedeno koristeći sabirnicu poruka.

4. Arhitektura rješenja



Slika 4.1: Arhitektura sustava

U središtu arhitekture ovog sustava je mikroservisna struktura potpomognuta AWS uslugama kako bi se osigurala skalabilnost, pouzdanost i efikasnost (Slika 4.1).

1. Klijent:

- predstavlja korisničko sučelje tj. frontend dio sustava
- komunikacija između klijenta i poslužitelja odvija se putem HTTP/HTTPS protokola

2. Poslužitelj:

- upravlja glavnim funkcionalnostima aplikacije
- obrađuje klijentove zahtjeve i komunicira s bazom podataka RDS, sabirnicom poruka SQS i AWS Lambda servisom

3. Baza podataka RDS (engl. *Amazon Relational Database Service*):

- RDS koristi se kao baza podataka gdje se pohranjuju svi podaci
- Poslužitelj dohvaća ili šalje podatke u bazu podataka prema potrebi
- Lambda funkcija izravno komunicira s bazom podataka kako bi dohvatila podatke potrebne za obradu spremljenih filtera

4. Sabirnica poruka SQS (engl. *Amazon Simple Queue Service*):

- SQS koristimo kao sabirnicu poruka za obradu novih objava

- Poslužitelj šalje detalje objave u obliku poruke u SQS koje Lambda funkcija kasnije obrađuje

5. AWS Lambda:

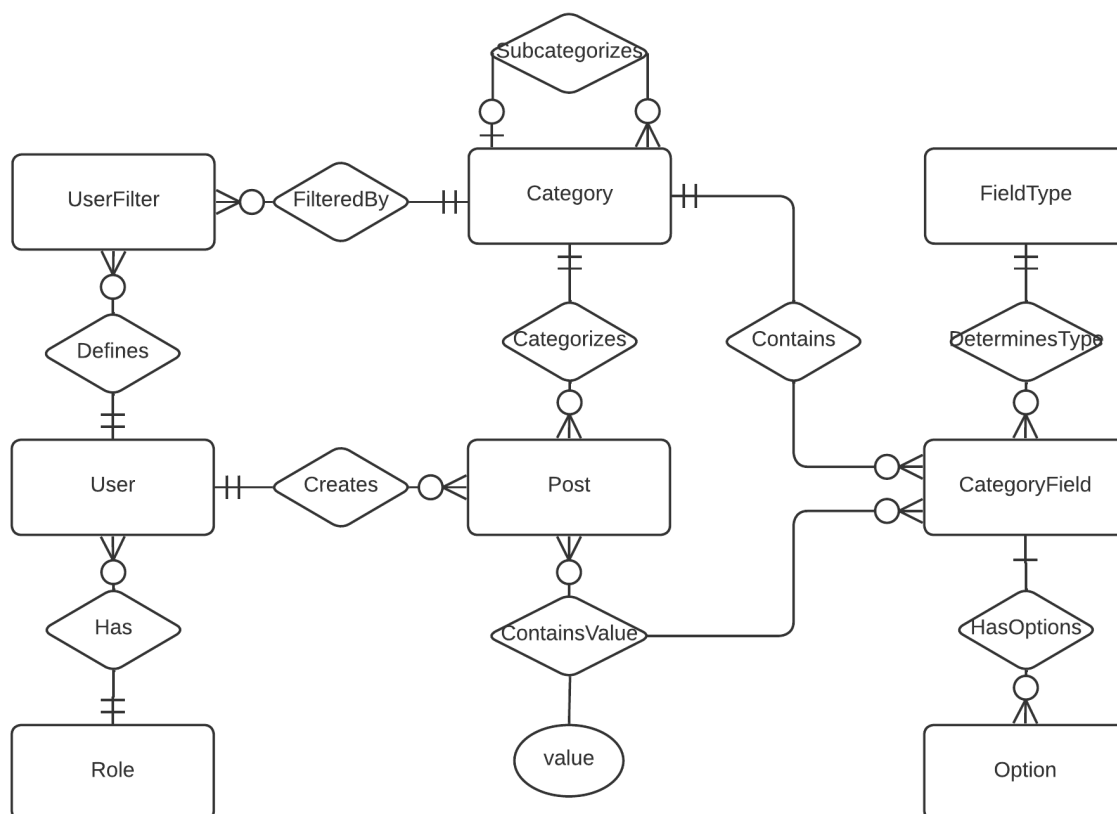
- Lambda funkcija konfigurirana je s okidačem (engl. *trigger*) za SQS, što znači da se automatski pokreće kada nova poruka stigne u red poruka SQS
- ova funkcija potom evaluira spremljene filtere iz baze i koristi Amazon SES za slanje e-mail notifikacija korisnicima

6. Email servis SES (engl. *Amazon Simple Email Service*):

- Nakon obrade u Lambda funkciji SES se koristi za slanje e-mailova korisnicima

4.1. Model podataka

Jedan od ključnih komponenata koji omogućava ispravno funkcioniranje aplikacije je model podataka. Model za izgradnju generičke platforme u oblaku za objavu informacija i oglasa predstavljen je ER dijagramom na Slika 4.2. Ovaj model ilustrira odnose između ključnih entiteta i njihove kardinalnosti.



Slika 4.2: ER Model podataka

Korisnik (**User**) predstavlja osnovu svake interakcije na platformi. Svaki korisnik ima (*Has*) dodijeljenu ulogu (entitet **Role**), koja definira njegove ovlasti i mogućnosti na portalu. Uloge omogućuju različite razine pristupa. Korisnik može imati samo jednu ulogu, a jednu ulogu može imati više korisnika.

Osnovna funkcionalnost koju korisnik koristi je kreiranje (*Creates*) i pregled objava (**Post**). Svaki korisnik može objaviti više objava. Svaka objava je kategorizirana (*Categorizes*) prema određenoj kategoriji (**Category**), koja služi kao tematski okvir za sadržaj. Kategorija može imati više objava dok svaka objava pripada samo jednoj kategoriji.

Kategorije se također mogu strukturirati hijerarhijski (*Subcategorizes*), gdje kategorije mogu biti podkategorije drugim kategorijama. Kategorija može imati više podkategorija, dok podkategorija može imati samo jednu nadređenu kategoriju.

Svaka kategorija može sadržavati (*Contains*) različita polja (**CategoryField**) koja dodatno opisuju i preciziraju sadržaj objave. Na primjer, kategorija „Glazbeni koncerti“ može imati polja poput „Mjesto“, „Vrijeme“ itd. Kategorija može sadržavati više polja, dok polje može pripadati samo jednoj kategoriji. Svako polje ima (*DeterminesType*) samo jedan tip polja (**FieldType**) koji može biti tekst, broj, datum, višestruki izbor i sl.

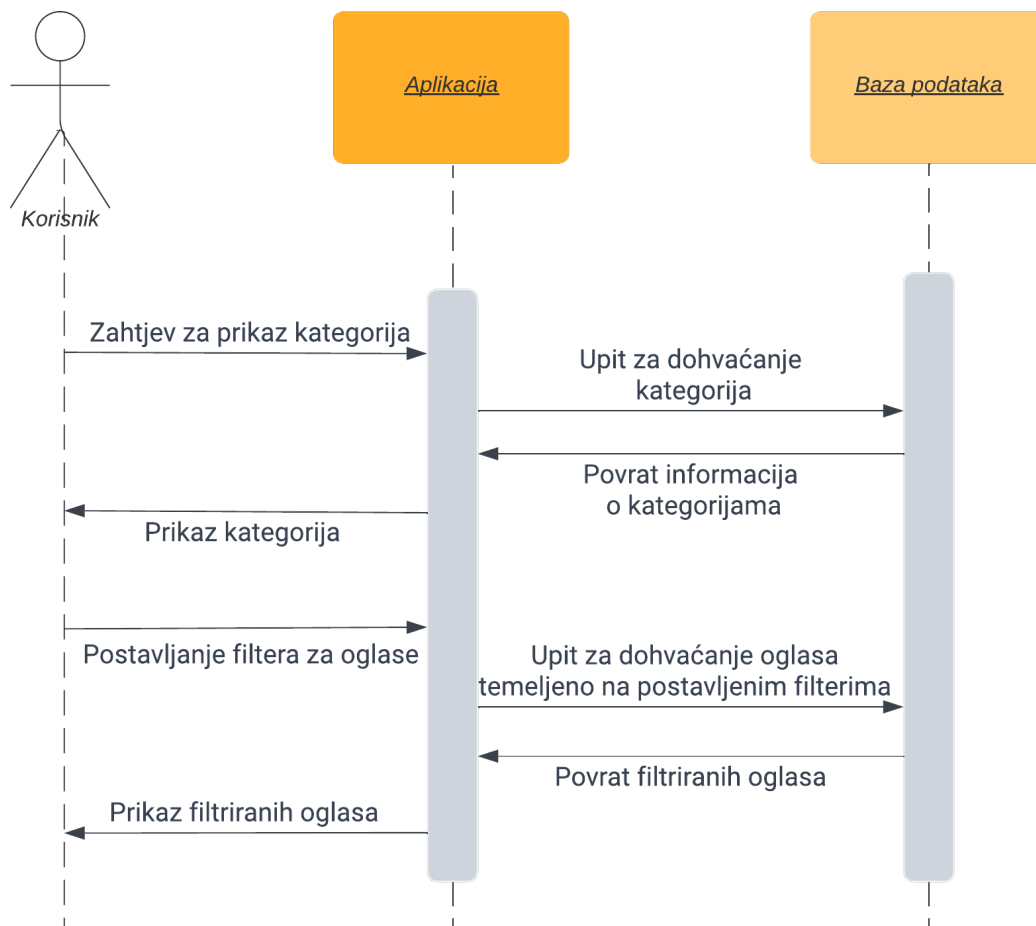
Za tip polja s višestrukim izborom, definira se (*HasOptions*) skup dopuštenih vrijednosti predstavljen entitetom **Option**.

Svaka objava pripada točno jednoj kategoriji i nasljeđuje polja te kategorije. S druge strane, svako polje kategorije je povezano sa svakom objavom koje pripadaju istoj kategoriji. Kardinalnost između objava i polja kategorija je *many-to-many* što znači da više objava može imati informacije (*ContainsValue*) temeljene na više različitih polja kategorija koje se spremaju u atributu *value*.

Za korisnike koji redovito posjećuju platformu i žele prilagoditi sadržaj prema svojim preferencijama, postoji mogućnost kreiranja (*Defines*) filtera (**UserFilter**). Filteri omogućuju korisnicima da personaliziraju prikaz objava na temelju određenih kriterija. Na primjer, korisnik može kreirati filter koji će mu prikazivati samo objave iz kategorije „Glazbeni koncerti“ u Zagrebu tijekom prosinca.

4.2. Sekvencijski dijagram

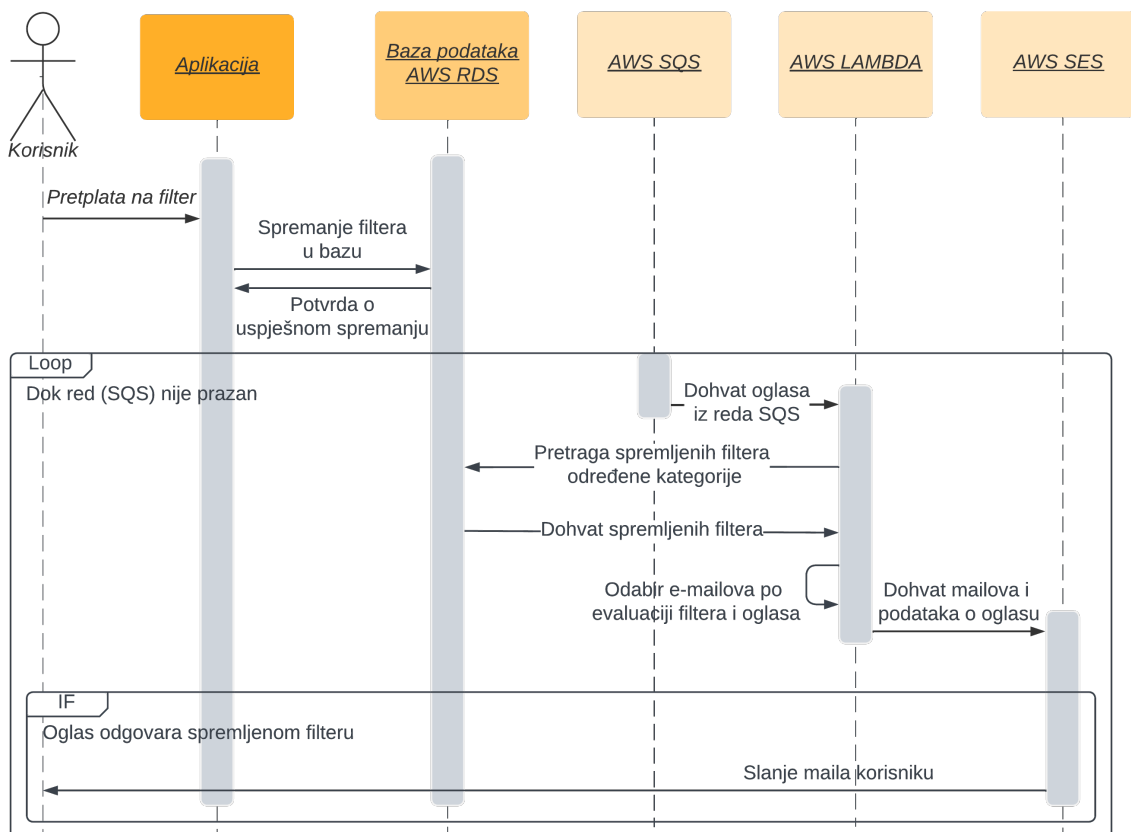
Sekvencijski dijagram prikazan na Slika 4.3 ilustrira proces kroz koji korisnik pristupa kategorijama i filtrira oglase unutar odabrane kategorije. Prvotno korisnik šalje zahtjev aplikaciji da vidi dostupne kategorije. Aplikacija potom šalje upit bazi podataka kako bi dohvatila popis svih kategorija. Kada baza podataka odgovori s informacijama o kategorijama, aplikacija ih prikazuje korisniku. Nakon što korisnik definira filtere i pošalje zahtjev, aplikacija šalje novi upit bazi podataka s ciljem dohvaćanja oglasa koji odgovaraju zadanim kriterijima. Baza podataka pretražuje i filtrira oglase prema postavljenim kriterijima te šalje rezultate natrag aplikaciji. Na kraju, aplikacija prikazuje korisniku oglase koji odgovaraju postavljenim filterima.



Slika 4.3: Prikaz kategorija i oglasa

Nakon odabira filtera za pretplatu (Slika 4.4), aplikacija filter pohranjuje u bazu podataka. Kada je filter uspješno spremljen, korisniku se šalje potvrda o uspješnoj pohrani. U pozadini, AWS Lambda funkcija konfigurirana je s okidačem za SQS te se automatski aktivira kada novi oglas stigne u red poruka SQS. Pojavom novog oglasa u SQS redu, započinje postupak evaluacije:

1. Lambda komunicira s bazom podataka da dohvati sve spremljene filtere koji su relevantni za kategoriju tog oglasa.
2. Nakon dohvaćanja filtera, Lambda funkcija također dohvaća e-mail adrese korisnika koji su se pretplatili na te filtere.
3. Ukoliko oglas odgovara nekom od filtera, Lambda funkcija koristi AWS SES za slanje e-mail-a korisnicima čiji filteri odgovaraju oglasu.



Slika 4.4: Pretplata korisnika i obavijesti o oglasima

5. Implementacija

5.1. Korištene tehnologije

5.1.1. PostgreSQL

PostgreSQL je *open-source* relacijska baza podataka koja je razvijena prije više od 30 godina na Sveučilištu Berkeley. Tijekom godina se prilagodio promjenjivim potrebama industrije što ga čini jednim od najdugovječnijih i najpouzdanijih sustava za upravljanje bazom podataka [1].

Prednosti:

1. **Visoka usklađenost sa standardima:** Postgres pruža široku podršku za SQL standard što olakšava prelazak iz drugih DBMS sustava.
2. **Napredni tipovi podataka:** Podržava JSON, XML i hstore što omogućuje fleksibilnost u pohranjivanju i upitu podataka.
3. **Proširivost:** Korisnici mogu definirati svoje tipove podataka, operatore i funkcije.
4. **Zajednica:** Budući da je *open-source* ima snažnu zajednicu koja pruža podršku, proširenja i dodatke.
5. **Povezanost s AWS-om:** Jednostavno je postaviti i upravljati Postgres instancama putem AWS RDS-a, što pruža skalabilnost, visoku dostupnost i sigurnosne značajke.

Nedostaci:

1. **Složenost:** Iako Postgres dolazi s raznovrsnim funkcionalnostima, to može biti i prednost i mana. Za one koji nisu upućeni u njegove specijalizirane mogućnosti, ovo može značiti povećanu složenost u upotrebi.
2. **Performanse:** Premda je Postgres vrlo brz, određene operacije mogu biti sporije u usporedbi s nekim komercijalnim DBMS-ima, posebno pri većim opterećenjima.

Za potrebe ovog projekta, Postgres je odabran zbog njegove praktičnosti, mogućnosti proširivosti i sposobnosti da efikasno upravlja velikim setovima podataka. Povezanost s AWS RDS-om dodatno je pojednostavila implementaciju i upravljanje bazom podataka.

5.1.2. Node.js

Node.js je *open-source* i *cross-platform* okruženje koje omogućava izvršavanje JavaScript-a na poslužiteljskoj strani. Dok se JavaScript tradicionalno koristi na klijentskoj strani unutar web preglednika, Node.js proširuje njegove sposobnosti na poslužitelj, čime omogućuje razvoj efikasnih i skalabilnih web aplikacija. Node.js se temelji na V8 JavaScript motoru, koji je ključna komponenta Google Chrome preglednika, osiguravajući time njegovu brzinu i performanse [2].

Prednosti Node.js [3]:

1. **Asinkronost:** Node.js koristi asinkroni, događajima vođen I/O pristup što ga čini idealnim za obradu više zahtjeva odjednom bez blokiranja.
2. **Jedan jezik:** Mogućnost pisanja klijentske i serverske strane aplikacije u istom jeziku (JavaScript) može značajno pojednostaviti razvoj i održavanje.
3. **Bogat ekosustav:** NPM (engl. *Node Package Manager*) najveći je *online* repozitorij biblioteka i alata što omogućuje brzo i lako dodavanje različitih funkcionalnosti aplikacije.
4. **Skalabilnost:** Pogodan je za mikroservisne arhitekture i lako se usklađuje s različitim bazama podataka i platformama.

Izazovi u radu s Node.js:

1. **Krivulja učenja:** Asinkroni kod može biti izazovan za razumijevanje i debugiranje (engl. *debugging*), posebno za one koji koriste sinkroni pristup.
2. **Nestabilne biblioteke:** Iako NPM ima ogroman broj paketa, neki od njih mogu biti nestabilni ili nedovoljno održavani.
3. **Performanse CPU-intenzivnih operacija:** Budući da je Node.js jednodretven, CPU-intenzivne operacije mogu blokirati cijelu aplikaciju. Za takve zadatke često su potrebne dodatne strategije.

Povezanost s drugim tehnologijama: Node.js često se koristi u kombinaciji s raznim bazama podataka, uključujući i PostgreSQL. Također, lako se integrira s platformama poput AWS-a.

5.1.3. React

React je razvila kompanija Facebook 2013. godine kao rješenje za izazove u izradi korisničkih sučelja. React je postao jedan od vodećih JavaScript biblioteka na svijetu s primjenom u velikim tvrtkama poput Airbnb-a, Netflix-a i Instagram-a [4].

React je temeljen na konceptu komponenti. Komponente predstavljaju osnovne građevne blokove svake React aplikacije. One su ponovno upotrebljive jedinice koda koje omogućavaju izgradnju složenih korisničkih sučelja na modularan način. Svaka komponenta je dobro izolirana, što olakšava njezino testiranje i održavanje. Ova modularnost ne samo da omogućava veću fleksibilnost u dizajnu, već i brzo ažuriranje sučelja bez potrebe za osvježavanjem cijele stranice.

Prednosti:

1. **Brze performanse zahvaljujući virtualnom DOM-u (engl. *Document Object Model*):** React koristi virtualni DOM kako bi smanjio potrebne manipulacije s pravim DOM-om. To omogućava aplikacijama da brzo ažuriraju sučelje bez značajnog utjecaja na performanse.
2. **Modularna arhitektura:** Kroz koncept komponenti, React omogućava programerima da kreiraju ponovno upotrebljive dijelove koda. Osim što olakšava razvoj, olakšava i održavanje aplikacija.
3. **Bogat ekosistem i zajednica:** Zbog svoje popularnosti, React ima ogromnu zajednicu koja pruža podršku, kao i mnoštvo dodatnih biblioteka i alata koji su kompatibilni s React-om.

Nedostaci:

1. **Potreba za drugim alatima:** Za kompletnu aplikaciju, često je potrebno kombinirati s drugim alatima i bibliotekama, poput Redux-a za upravljanje stanjem i sl.
2. **Brz razvoj i promjene:** Kako se React kontinuirano ažurira i razvija, programeri moraju redovito ažurirati svoje znanje i prilagoditi postojeći kod novim verzijama, što može dodatno otežati održavanje projekata.

5.1.4. AWS (engl. *Amazon Web Services*)

Amazon Web Services (AWS) je najveća i najraširenija *cloud* platforma na svijetu koju nudi Amazon. Pruža širok spektar infrastrukturnih usluga poput računalne snage, pohrane podataka i rješenja za povezivanje, sve dostupno na zahtjev.

5.1.4.1 RDS (engl. *Relation Database Service*)

Amazon RDS je upravljana usluga koja pojednostavljuje postupak postavljanja, korištenja i skaliranja relacijske baze podataka u oblaku. Podržava različite baze podataka poput PostgreSQL, MySQL, MariaDB, Oracle i Microsoft SQL Server.

Prednosti:

- Lakoća upravljanja: Automatsko sigurnosno kopiranje i obnova.
- Skalabilnost: Mogućnost lakog skaliranja performansi i kapaciteta.

5.1.4.2 SQS (engl. *Simple Queue Service*)

Amazon SQS je usluga poruka koja omogućava asinkronu obradu poruka korištenjem sigurnih redova poruka. Omogućava odvajanje (engl. *decoupling*) komponenata aplikacije.

Prednosti:

- Pouzdanost: Poruke se čuvaju do 14 dana.
- Skalabilnost: Automatsko skaliranje bez potrebe za intervencijom korisnika.
- Sigurnost: Podrška za enkripciju i IAM politike.

5.1.4.3 LAMBDA

AWS Lambda je računalna usluga koja omogućava pokretanje koda bez potrebe za upravljanjem poslužiteljima. Usluga automatski upravlja resursima potrebnim za izvršavanje koda u odgovoru na određene događaje

Prednosti:

- Bez poslužitelja: Nema potrebe za postavljanjem ili upravljanjem infrastrukturom.
- Automatsko skaliranje: Izvršava kod kao odziv na događaje, automatski skalirajući resurse.
- Fleksibilnost: Podržava više programskih jezika.

5.1.4.4 SES (engl. *Simple Email Service*)

Amazon SES je usluga za slanje e-mailova koja pruža mogućnosti za upravljanje različitim vrstama e-mail komunikacije. Može se koristiti za transakcijske, marketinške i masovne e-mailove. Kompatibilna je s nizom drugih AWS servisa i omogućuje automatizaciju raznih procesa.

Prednosti:

- Skalabilnost: Omogućava slanje velikog broja e-mailova bez potrebe za dodatnom infrastrukturom.
- Efikasnost troškova: Plaćate samo za ono što koristite, omogućujući tako efikasno upravljanje troškovima.

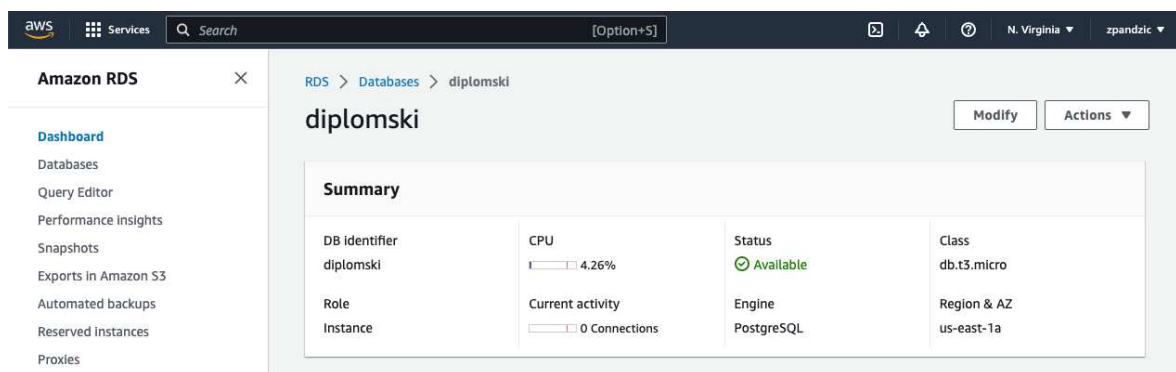
5.2. Baza podataka

U središtu svakog informacijskog sustava nalazi se baza podataka, koja služi kao osnovna komponenta za pohranu, dohvat i upravljanje podacima. Ovo poglavlje fokusira se na dizajn i strukturu baze podataka koja podržava generičku platformu za objavu informacija i oglasa.

5.2.1. RDS (engl. *Relational Database Service*)

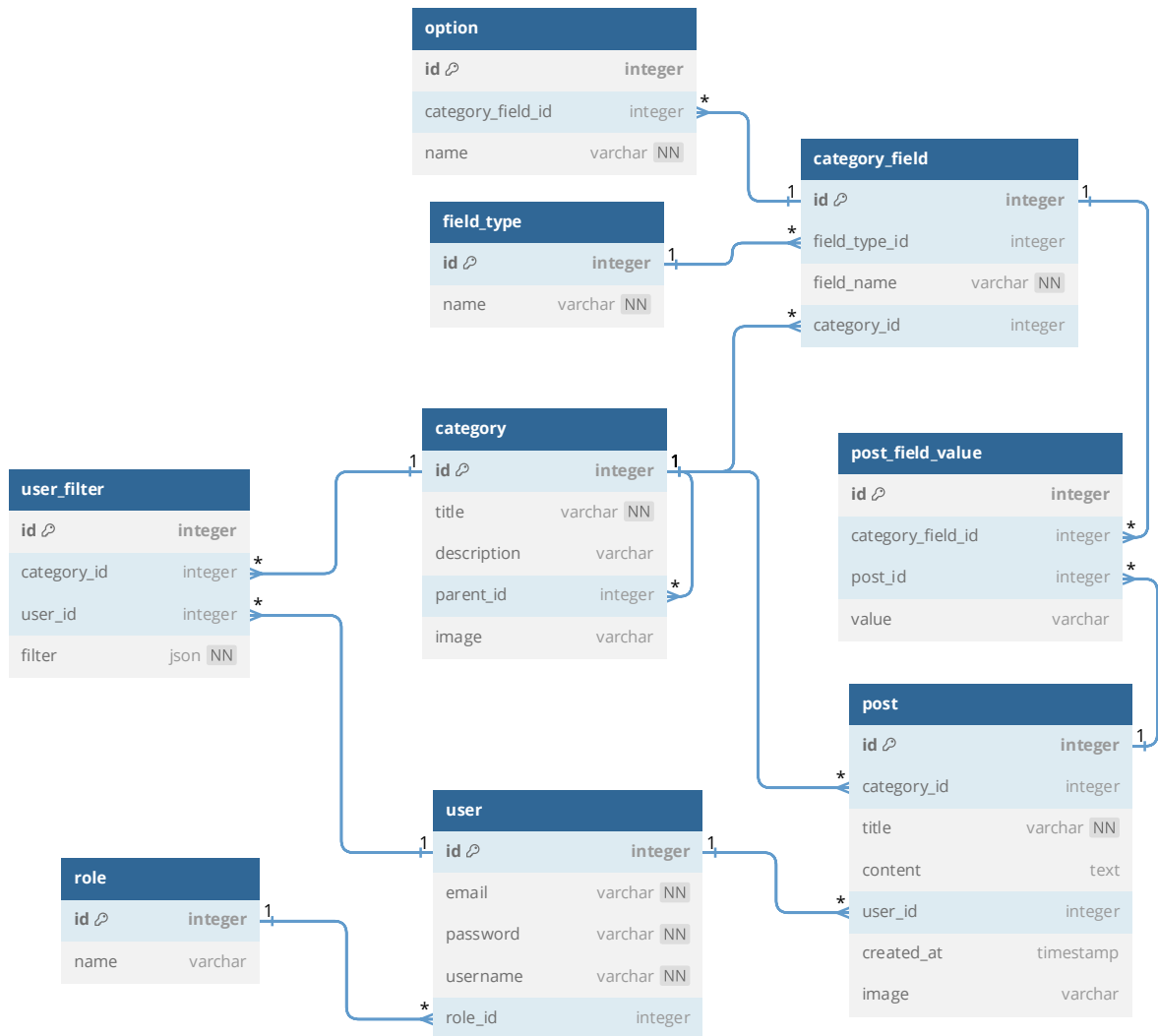
Za potrebe ovog projekta, odabrana je Amazon RDS platforma s PostgreSQL kao preferiranom relacijskom bazom podataka. Slika 5.1 prikazuje korisničko sučelje RDS-a s detaljima postavljene baze podataka. RDS nudi upravljane baze podataka optimizirane za performanse, skalabilnost i sigurnost. Specifikacije odabrane tijekom postavljanja su:

- **Tip baze podataka:** PostgreSQL
- **Identifikator:** diplomski
- **Klasa instance:** db.t3.micro - ovaj tip instance je odabran s obzirom na trenutačne potrebe u pogledu performansi i troškova
- **Region & AZ:** us-east-1a - odabrano zbog geografske prednosti i optimalne dostupnosti



Slika 5.1: Korisničko sučelje Amazon RDS-a

5.2.2. Struktura baze podataka



Slika 5.2 ER dijagram baze podataka

Kategorija (engl. *Category*)

Ova tablica čuva informacije vezane za kategorije oglasa. Svaka kategorija se karakterizira kroz sljedeće attribute:

- **Naslov:** oznaka kategorije
- **Opis:** detaljna informacija o čemu se kategorija bavi
- **ParentCategoryID (ID Roditeljske Kategorije):** Referenca na višu kategoriju u hijerarhijskom ustroju kategorija. Ova relacija omogućava stvaranje hijerarhije među kategorijama.

- **Slika:** vizualna reprezentacija kategorije

Polje Kategorije (engl. *CategoryField*)

Ova tablica pohranjuje informacije o specifičnim poljima koja se mogu pridružiti određenoj kategoriji. Svako polje je definirano kroz:

- **Naziv:** ime polja
- **CategoryID (ID Kategorije):** strani ključ koji upućuje na tablicu **Category**
- **FieldTypeID (ID Tipa Polja):** vrsta informacije koja se može unijeti u polje, čiji detalji se mogu pronaći u tablici **FieldType**
- **Opis:** dodatne informacije ili napomene o polju

Vrsta Polja (engl. *FieldType*)

Tablica sadrži informacije o različitim vrstama polja koje se mogu dodijeliti kategorijama.

- **FieldTypeID (ID Tipa Polja):** primarni ključ za identifikaciju tipa polja
- **Tip:** naziv tip polja (npr. text, number, select, boolean, date)

Opcija (engl. *Option*)

Ova tablica sadržava informacije o dostupnim opcijama za polja koja imaju višestruke odabire, kao što je „Select“ Za svaku opciju definirani su:

- **Naziv:** ime opcije
- **FieldTypeID (ID Tipa Polja):** strani ključ koji upućuje na **FieldType**
- **Vrijednost:** Konkretna vrijednost koja je povezana s imenom opcije. Na primjer, ako razmatramo polje „Boja“ s tipom polja „Select“, moguće vrijednosti mogu biti „Crvena“, „Plava“ itd.

Objava (engl. *Post*)

Ova tablica čuva informacije o događajima i oglasima koji su objavljeni na portalu. Svaka objava je karakterizirana kroz:

- **Naslov:** glavni naslov objave
- **Sadržaj:** tekstualni i glavni dio objave

- **Kategorija:** specifična kategorija kojoj objava pripada
- **UserID (ID Korisnika):** korisnik koji je kreirao objavu
- **Datum objave:** precizan datum kada je objava postavljena na portalu

Vrijednost Polja Objave (engl. *PostFieldValue*)

Tablica pohranjuje informacije o konkretnim vrijednostima polja koja su pridružena specifičnoj objavi. Za svaku vrijednost karakteristike su:

- **Naziv polja:** ime određenog polja koje se odnosi na objavu
- **PostID (ID Objave):** strani ključ na tablicu **Post**
- **FieldID (ID Polja):** strani ključ povezan s **CategoryField**
- **Vrijednost:** podatak koji je povezan s određenim poljem

Korisnik (engl. *User*)

Tablica sadržava informacije o korisnicima koji su registrirani na platformi:

- **E-mail adresa:** elektronička adresa korisnika
- **Lozinka:** tajna šifra korisnika
- **Korisničko ime:** jedinstvena oznaka svakog korisnika na platformi
- **RoleID (ID Uloge):** definira ovlasti i prava korisnika na platformi tablicom **Role**

Uloga (engl. *Role*)

Ova tablica pruža informacije o različitim ulogama koje korisnici mogu imati na platformi. Uloge su jasno definirane i uključuju:

- **Naziv:** ime uloge (administrator ili korisnik)

Korisnički Filter (engl. *UserFilter*)

Tablica pohranjuje informacije o specifičnim filterima koje korisnici mogu spremiti:

- **Kategorija:** određena kategorija na koju se filter primjenjuje

- **JSON objekt:** strukturirani skup podataka koji sadrži sve parametre filtera. Na primjer, mogu se pretraživati objave koje pripadaju kategoriji „Nogomet“ i imaju cijenu između 20 i 50 eura

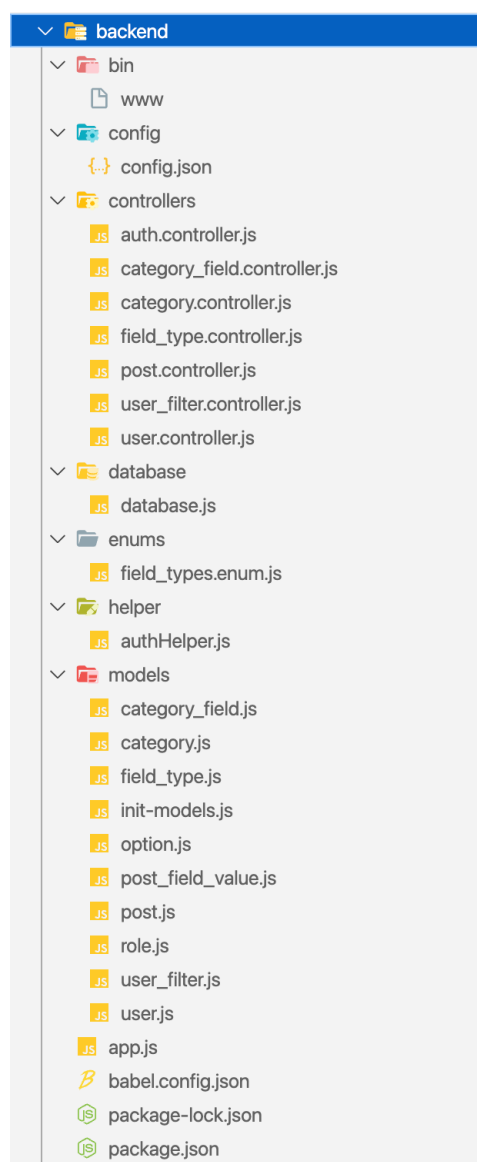
5.3. Poslužitelj

U ovom poglavlju detaljno će se osvrnuti na poslužiteljski dio aplikacije, odnosno backend, koji je ključan za obradu i upravljanje podacima, kao i za interakciju s klijentskim dijelom aplikacije.

5.3.1. Struktura direktorija poslužitelja

Backend aplikacije organiziran je kroz logički smislen i strukturiran način kako bi se omogućila jednostavna navigacija i održavanje (Slika 5.3):

- **app.js:** inicijalizira Express aplikaciju te definira putanje
- **bin/www:** skripta za pokretanje servera
- **config/config.json:** sadrži postavke za bazu podataka
- **controllers:** kontroleri koji upravljaju logikom aplikacije za različite segmente aplikacije (autentifikacija, kategorije, postovi, itd.)
- **database/database.js:** konfiguracija za inicijalizaciju konekcije s bazom podataka i modelima
- **enums/field_types.enum.js:** definira moguće tipove polja
- **helper/authHelper.js:** funkcija za rad s JWT-om
- **models:** sadrži modele koji reprezentiraju tablice u bazi podataka
- **package.json & package-lock.json:** definiraju ovisnosti projekta
- **node_modules:** instalirane npm ovisnosti



Slika 5.3: Struktura direktorija poslužitelja

Za pokretanje poslužiteljskog dijela aplikacije koriste se npm skripte specificirane u **package.json** datoteci. Skripta **start** definirana u **package.json**

izvršava naredbu **node ./bin/www**, koja pokreće skriptu unutar **bin/www** direktorija. Ova skripta potom inicijalizira i pokreće Express server koji prima HTTP zahtjeve.

5.3.2. Programski kodovi

5.3.2.1 Pretraživanje objava

Funkcija **search** (Programski kod 1) je funkcija za pretraživanje postova. Iz **req.body** dohvaća se nekoliko uvjeta poput **category_id**, **title**, **category_fields** i **content**. **whereClause** objekt se koristi za skladištenje svih uvjeta pretrage. Ukoliko određeni uvjeti pretrage postoje (npr. **title** ili **content**), oni se dodaju u **whereClause** objekt. Ako su definirana polja kategorije (**category_fields**), funkcija **buildAndConditions** generira podupit koji filtrira postove na temelju tih polja. Nakon što su svi uvjeti postavljeni, koristi se Sequelize-ova metoda **Post.findAll** za dohvaćanje svih postova koji zadovoljavaju date uvjete. Rezultat se potom vraća kao odgovor na upit.

```
const search = async (req, res) => {
  try {
    const { category_fields, category_id, title, content } = req.body;

    const whereClause = {
      ...(category_id && { category_id }),
      ...(title && { title: { [Op.iLike]: `%${title}%` } }),
      ...(content && { content: { [Op.iLike]: `%${content}%` } }),
      ...(category_fields?.length && {
        id: { [Op.and]: buildAndConditions(category_fields) },
      }),
    },
  );

  const posts = await Post.findAll({ where: whereClause });

  return posts;
} catch (err) {
  console.error(err);
  return [];
}
```

Programski kod 1: Funkcija pretraživanja objava

Funkcija **buildConditionForField** (Programski kod 2) je generira SQL uvjete za filtriranje na temelju tipa polja (**field_type_id**) i pripadajućih vrijednosti (**value**, **min_value**, **max_value**). Ova funkcija omogućuje visoku razinu dinamičnosti u filtriranju, uzimajući u obzir različite tipove podataka. Za tekstualne tipove koristi se operator „ILIKE“, za brojeve i datume provodi se filtriranje unutar određenog raspona, itd.

```
const buildConditionForField = (field_type_id, value, min_value, max_value) => {
  switch (field_type_id) {
    case FieldTypeEnum.TEXT:
      return ` AND value ILIKE '${value}%'`;
    case FieldTypeEnum.NUMBER:
      if (min_value && max_value) {
        return ` AND value::integer BETWEEN ${min_value} AND ${max_value}`;
      }
      return min_value ? ` AND value::integer >= ${min_value}` :
        ` AND value::integer <= ${max_value}`;
    case FieldTypeEnum.DATE:
      if (min_value && max_value) {
        return ` AND value::date BETWEEN '${min_value}' AND '${max_value}'`;
      }
      return min_value ? ` AND value::date >= '${min_value}'` :
        ` AND value::date <= '${max_value}'`;
    case FieldTypeEnum.SELECT:
    case FieldTypeEnum.BOOLEAN:
      return ` AND value = '${value}'`;
    default:
      return '';
  }
};
```

Programski kod 2: Dinamička konstrukcija SQL upita za pretragu objava

5.3.2.2 Autentifikacija

Funkcija **auth** (Programski kod 3) je *middleware* koji provjerava je li u HTTP zahtjevu prisutan i valjan JWT token za autentifikaciju korisnika. Ako token nije prisutan ili nije valjan, vraća se odgovarajuća poruka o grešci. Ako je sve u redu obrada zahtjeva se nastavlja sa sljedećom funkcijom u lancu.

```
function auth(req, res, next) {
  const token = req.header('Authorization')?.replace('Bearer ', '');
  if (!token) return res.status(401).send('Access denied. No token provided.');
```

```
  try {
    const decoded = jwt.verify(token, JWT_SECRET);
    req.user = decoded;
    next();
  } catch (ex) {
    res.status(400).send('Invalid token.');
```

```
  }
}
```

Programski kod 3: Autentifikacija korisnika

5.3.2.3 Slanje detalja objave u AWS *Simple Queue Service* (SQS).

Funkcija **sendSQS** (Programski kod 4) detalje objave pretvara u JSON format i šalje ih kao poruku SQS-u na adresu specifikaciju u **SQS_URL**. U slučaju greške prilikom slanja, greška se evidentira u konzoli.

```
const sendSQS = (data) => {
  const params = {
    MessageBody: JSON.stringify(data),
    QueueUrl: process.env.SQS_URL,
  };

  sqs.sendMessage(params, (err, result) => {
    if (err) {
      console.error(err);
      return;
    }
    console.log(result, params);
  });
};
```

Programski kod 4: Slanje detalja oglasa u SQS

5.3.2.4 Kreiranje objave

Programski kod 5 započinje s definicijom rute za kreiranje objave. Prvo se iz tijela zahtjeva izvlače potrebne informacije. Zatim se započinje transakcija baze podataka kako bi se osigurao konzistentan unos podataka. Ako se tijekom unosa pojavi bilo kakva greška, transakcija se poništava, čime se osigurava da baza ostane konzistentna. Nakon uspješnog unosa, objava se šalje u SQS red poruka pomoću **sendSQS** funkcije.

```
router.post('/', auth, async (req, res) => {
  const { title, content, category_id, post_fields } = req.body;
  const user_id = req.user.userId;
  const t = await sequelize.transaction();

  try {
    const post = await Post.create({ title, content, user_id, category_id },
      { transaction: t });

    if (post_fields) {
      await PostFieldValue.bulkCreate(post_fields.map((field) => ({
        post_id: post.id,
        category_field_id: field.category_field_id,
        value: field.value,
      })), { transaction: t });
    }
    await t.commit();

    sendSQS(post);
    res.status(201).json(post);
  } catch (error) {
    await t.rollback();
    res.status(400).json({ error: error.message });
  }
});
```

Programski kod 5: Kreiranje objave

5.3.2.5 Spremanje filtera

Programski kod 6 koristi middleware za autentifikaciju kako bi osigurao pristup samo ovlaštenim korisnicima. Iz zahtjeva se izdvaja **category_id** i **filter**, koji se potom konvertira u JSON format i pohranjuje u bazu podataka.

```
router.post('/save-filter', auth, async (req, res) => {
  const { category_id, filter } = req.body;
  try {
    const userFilter = await user_filter.create({
      user_id: req.user.userId,
      category_id,
      filter: JSON.stringify(filter),
    });
    res.status(201).json(userFilter);
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
});
```

Programski kod 6: Spremanje filtera

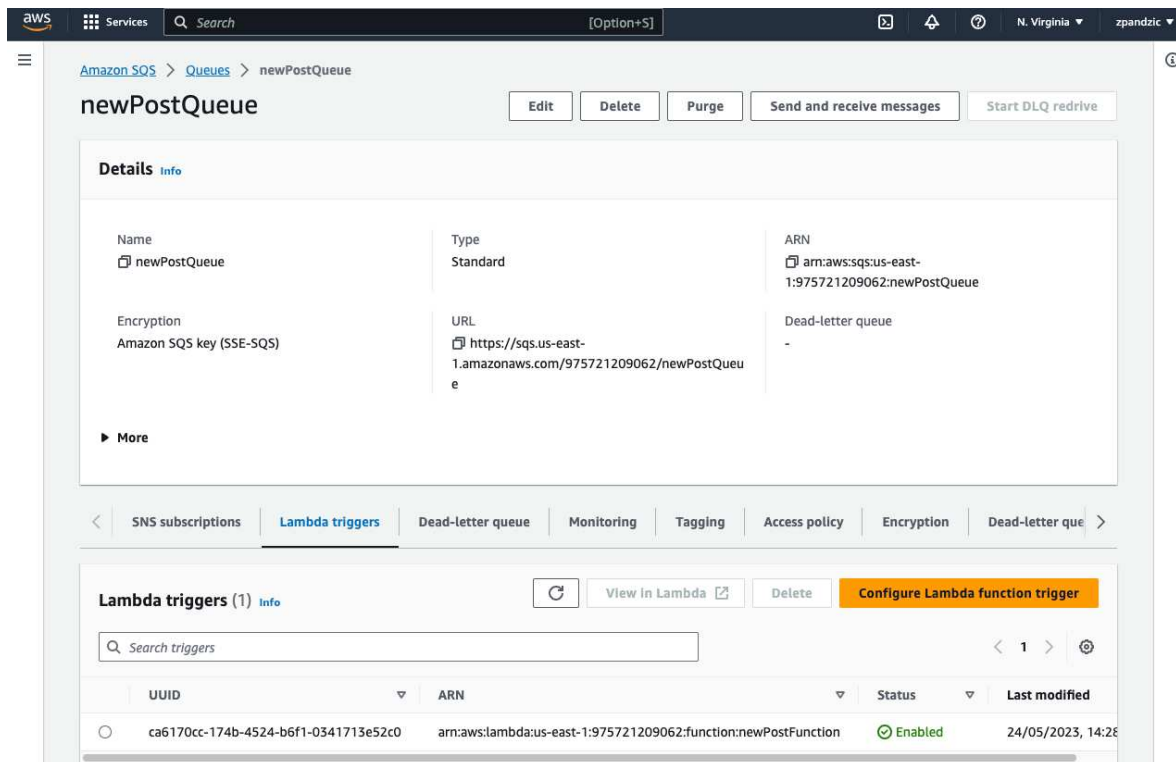
5.4. AWS SQS

Amazon Simple Queue Service (SQS) igra ključnu ulogu u aplikaciji te omogućava asinkronu obradu novih objava. Nakon što se objava uspješno stvori u bazi podataka, detalji nove objave šalju se SQS-u putem funkcije **sendSQS** koje je objašnjena u poglavlju „Slanje detalja objave u AWS Simple Queue Service (SQS)“. Slika 5.4 prikazuje korisničko sučelje SQS-a s detaljima postavljenog reda poruka.

Specifikacije reda poruka postavljene za ovaj projekt su:

- **Naziv reda:** **newPostQueue**
- **Vrsta reda:** Standard - ovaj tip reda omogućava visoku propusnost i jamči da će poruka biti dostavljena barem jednom
- **ARN:** **arn:aws:sqs:us-east-1:...:newPostQueue** Amazon Resource Name - identifikator za red
- **Enkripcija:** Amazon SQS key (SSE-SQS) - pruža dodatni sloj sigurnosti enkriptirajući poruke spremljene u red

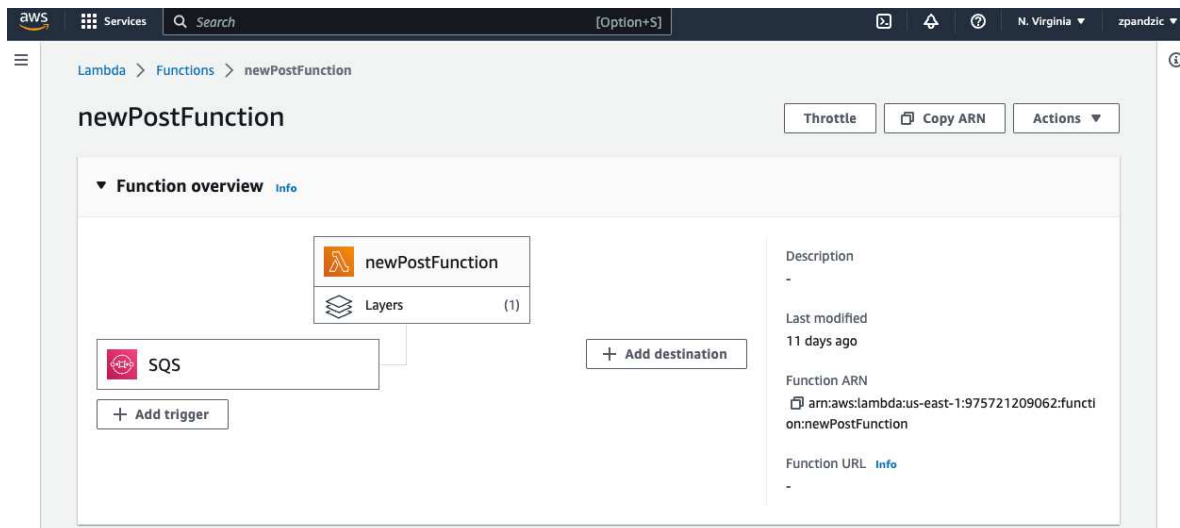
- **URL:** `https://.../newPostQueue` - Ova adresa funkcioniра kao krajnja točka (engl. *endpoint*) za sve akcije koje su usko vezane uz ovaj red poruka. U prikazanom Programski kod 4, varijabla `queueUrl` označava URL lokaciju navedenog reda poruka
- **Lambda Trigger:** Lambda funkcija `arn:aws:lambda:...:newPostFunction` automatski se aktivira svaki put kad stigne nova poruka u red



Slika 5.4: Korisničko sučelje Amazon SQS-a

5.5. AWS Lambda

AWS Lambda predstavlja ključnu komponentu aplikacije koja omogućava *serverless* izvršavanje koda. Slika 5.5 prikazuje korisničko sučelje AWS Lambda s detaljima funkcije **newPostFunction**. AWS Lambda automatski se aktivira nakon što se u redu poruka pojavi nova objava. Iako destinacija AWS SES-a, koja je zadužena za slanje e-mail obavijesti, nije direktno vidljiva na korisničkom sučelju (obzirom da je definirana unutar koda - Programski kod 12), ona je ključna za ispravno funkcioniranje cijelog sustava.



Slika 5.5: Korisničko sučelje Amazon Lambde

5.5.1. Kod Lambda funkcije

Programski kod 7 predstavlja proces uvoza potrebnih biblioteka i omogućava interakciju s AWS uslugama, uspostavlja vezu s PostgreSQL bazom podataka i postavlja inicijalne postavke.

```
import AWS from "aws-sdk";
import pkg from "pg";
const { Client } = pkg;

AWS.config.update({ region: 'us-east-1' });

const ses = new AWS.SES({ apiVersion: '2010-12-01' });
```

Programski kod 7: Uvođenje biblioteka i konfiguracija AWS-a

handler(): (Programski kod 8) je glavna ulazna funkcija koja se pokreće kada se događaj (engl. *event*) zaprimi. Kada je pozvana, inicijalizira konekciju prema bazi podataka, dohvaća detalje objave iz reda poruka (**event.Records**), traži odgovarajuće spremljene korisničke filtere u bazi po kategoriji dohvaćene objave i na kraju šalje e-mail obavijesti korisnicima koji zadovoljavaju uvjete filtera.

```

export const handler = async (event) => {
  const client = initializeDatabaseClient();
  let postDetails = JSON.parse(event.Records[0].body);

  const userFilters = await getUserFilters(client, postDetails.category_id);
  await client.end();

  const userEmails = extractUserEmails(postDetails, userFilters);
  sendEmailNotifications(userEmails, postDetails);
};

```

Programski kod 8: Ulazna handler funkcija

initializeDatabaseClient(): Ova funkcija (Programski kod 9) konfigurira i uspostavlja vezu s PostgreSQL bazom podataka te vraća objekt klijenta za komunikaciju s bazom.

```

function initializeDatabaseClient() {
  const client = new Client({
    // ...database config
  });
  await client.connect();
  return client;
}

```

Programski kod 9: Inicijalizacija klijenta baze podataka

getUserFilters(): Kroz ovu funkciju (Programski kod 10) vršimo upit prema bazi podataka kako bismo dohvatili sve spremljene filtere korisnika koji su spremljeni za tu određenu kategoriju.

```

async function getUserFilters(client, categoryId) {
  const res = await client.query(`
    SELECT "user".email, user_filter.filter
    FROM user_filter
    JOIN "user" ON user_filter.user_id = "user".id
    WHERE user_filter.category_id = $1
  `, [categoryId]);
  return res.rows;
}

```

Programski kod 10: Dohvat spremljenih filtera iz baze

extractUserEmails(): Ova funkcija (Programski kod 11) prolazi kroz listu dohvaćenih korisničkih filtera te provjerava odgovara li objava uvjetima svakog filtera. Ako objava

zadovoljava uvjete nekog filtera, e-mail adresa korisnika koji je postavio taj filter dodaje se na listu e-mail adresa za slanje obavijesti.

```
function extractUserEmails(postDetails, userFilters) {
  let userEmails = [];
  userFilters.forEach(filter => {
    if (checkIfPostMatchesFilter(postDetails, JSON.parse(filter.filter))) {
      userEmails.push(filter.email);
    }
  });
  return userEmails;
}
```

Programski kod 11: Provjera uvjeta filtera

sendEmailNotifications(): Nakon što se sastavi lista e-mail adresa korisnika koji trebaju primiti obavijest, ova funkcija (Programski kod 12) generira tijelo e-maila i šalje obavijesti svim e-mail adresama s liste. E-mailovi se šalju kroz AWS SES uslugu, što osigurava skalabilno i pouzdano slanje e-mailova.

```
function sendEmailNotifications(userEmails, postDetails) {
  let messageBody = `New post: ${postDetails.title}\n${postDetails.content}`;
  userEmails.forEach(email => {
    const params = {
      // ...email parameters
    };
    await ses.sendEmail(params).promise();
  });
}
```

Programski kod 12: Slanje E-mailova kroz SES

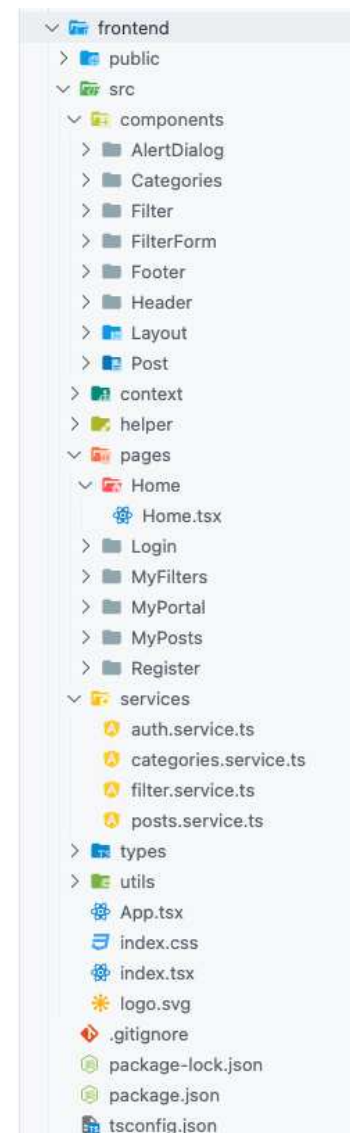
5.6. Klijent

Fokus ovog poglavlja je na klijentskom dijelu aplikacije, odnosno frontendu, koji je odgovoran za prikazivanje informacija korisnicima i omogućava interakciju s poslužiteljskim dijelom aplikacije. Klijentska aplikacija koristi React kao glavnu biblioteku za izgradnju korisničkog sučelja, dok TypeScript pruža strožu tipizaciju i omogućava lakše otkrivanje grešaka pri razvoju.

5.6.1. Struktura klijentskog direktorija

Klijentski dio aplikacije sastoji se od (Slika 5.6):

- **public:** direktorij u kojem se nalaze osnovne datoteke za javni dio aplikacije
- **src:** sadrži izvorni kod aplikacije
 - **App.tsx:** glavna komponenta aplikacije
 - **pages:** glavne stranice aplikacije: Home, MyFilters, MyPosts, Login itd.
 - **components:** direktorij za sve React komponente aplikacije kao što su:
 - **Header:** osnovne komponente za zaglavlje aplikacije
 - **Post:** upravljanje i prikaz objava
 - **Categories:** upravljanje i prikaz kategorija
 - **Filter i FilterForm:** komponente za kreiranje i upravljanje filterima
 - **AlertDialog:** prozor za upozorenja korisniku
 - **context:** kontekstualne komponente za globalno upravljanje stanjima
 - **helper:** pomoćne funkcije za obradu podataka
 - **services:** servisi za komunikaciju s poslužiteljom
 - **types:** definicije tipova i enumeracije
 - **utils:** dodatne funkcije za API pozive
- **package.json & package-lock.json:** sadrže informacije o projektu i njegovim ovisnostima
- **tsconfig.json:** konfiguracija za TypeScript



Slika 5.6: Struktura direktorija klijenta

5.6.2. Programski kodovi

5.6.2.1 Komunikacija s poslužiteljem

U aplikaciji je korištena biblioteka `axios` za izvođenje HTTP zahtjeva prema poslužitelju. Axios omogućava jednostavno izvođenje asinkronih zahtjeva i rukovanje odgovorima. Programski kod 13 predstavlja osnovnu konfiguraciju za **axios**. Postavljena je osnovna adresa API-ja i zaglavlje koje se šalje uz svaki zahtjev. Dodatno, implementiran je *interceptor* koji automatski dodaje JWT token iz lokalne pohrane pri svakom zahtjevu. Ovo omogućava da se korisnik autentificira kod svakog zahtjeva prema poslužitelju.

```
const api = axios.create({
  baseURL: 'http://localhost:3000',
  headers: {
    'Content-Type': 'application/json; charset=utf-8',
    'Access-Control-Allow-Origin': '*',
  },
});
api.interceptors.request.use((config) => {
  const token = localStorage.getItem('token');
  if (token) config.headers.Authorization = `Bearer ${token}`;
  return config;
});
```

Programski kod 13: Konfiguracija Axios-a za API komunikaciju

5.6.2.2 Autentifikacija

Implementacija sustava autentifikacije osigurava da samo ovlašteni korisnici mogu pristupiti određenim dijelovima aplikacije. Kroz Programski kod 14 prikazan je servis za autentifikaciju koji ima metode za prijavu i registraciju korisnika. Ove metode zahtjeve šalju poslužitelju na odgovarajući *endpoint* kako bi izvršile tražene operacije.

```
const authService = {
  login: (credentials: { email: string; password: string }) =>
    api.post('/auth/login', credentials),
  register: (data: { email: string; password: string; username: string }) =>
    api.post('/auth/register', data),
};

export default authService;
```

Programski kod 14: Implementacija servisa za autentifikaciju

5.6.2.3 Upravljanje globalnim stanjem

Kako bi se osigurala konzistentnost podataka kroz aplikaciju koristi se React Context za upravljanje globalnim stanjem. **UserContextProvider** (Programski kod 15) omogućava upravljanje podacima o autentificiranom korisniku kroz cijelu aplikaciju.

```
function UserContextProvider({ children }: any) {
  const [token, setToken] = useState<string | null>(null);
  const [userData, setUserData] = useState<{ email: string;
    userId: number; username: string } | null>(null);

  async function authenticate(jwt: string) {
    setToken(jwt);
    setUserData(jwt_decode<any>(jwt));
    localStorage.setItem('token', jwt);
  }
  // Ostatak koda
}
```

Programski kod 15: Konfiguracija React Context-a
za upravljanje globalnim korisničkim stanjem

5.6.2.4 Dohvaćanje podataka - Hookovi

React Hook-ovi omogućavaju rad sa stanjima u funkcionalnim komponentama bez potrebe za klasičnim React klasama. U primjeru koji je prikazan (Programski kod 16), **useEffect** hook dohvaća kategorije pri montiranju komponente i ažurira njeno stanje s tim podacima.

```
useEffect(() => {
  const fetchCategory = async () => {
    try {
      const categoryData = await
        categoriesService.getCategoryById(categoryId);
      setCategory(categoryData);
    } catch (error) {
      console.error(error);
    }
  };
  fetchCategory();
}, []);
```

Programski kod 16: Korištenje useEffect hook-a
za dohvaćanje podataka o kategoriji

5.6.2.5 Prikaz podataka korisnicima

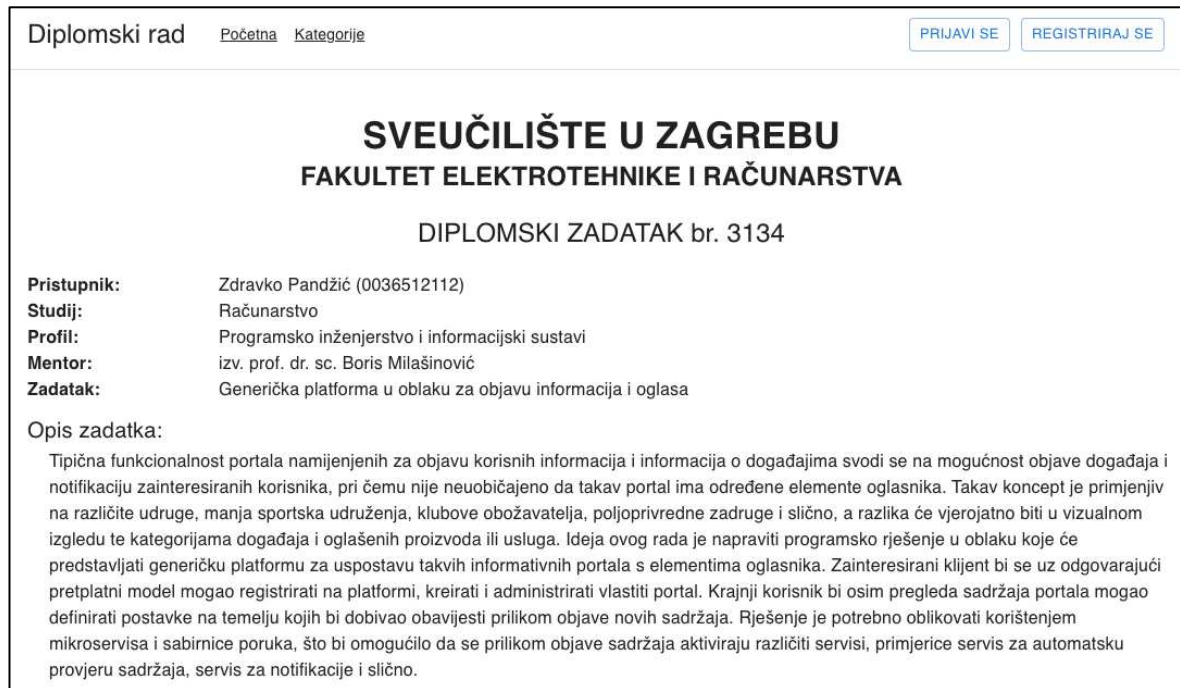
Kada se podaci dohvate s poslužitelja, važno je prikazati ih korisnicima na razumljiv i pregledan način. U navedenom kodu, komponenta **CategoryCardList** prima listu kategorija i funkciju koja se pokreće kada korisnik izabere neku od kategorija. Svaka kategorija se prikazuje kao **CategoryCard**, odnosno kao pojedinačna kartica koja vizualno predstavlja tu kategoriju.

```
export default function CategoryCardList({
  categories,
  changeCategoryHandler,
}: any) {
  return (
    <Grid container spacing={4}>
      {categories.map((category: Category) => (
        <CategoryCard
          key={category.id}
          category={category}
          changeCategory={() => changeCategoryHandler(category.id)}
        />
      ))}
    </Grid>
  );
}
```

Programski kod 17: Komponenta za prikaz kategorija

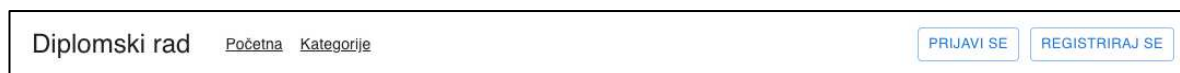
6. Korisničke upute

Platforma je dizajnirana da bude jednostavna za korištenje, s intuitivnim sučeljem koje omogućuje pregled kategorija, oglasa, spremljenih filtera i vlastitog portala.



Slika 6.1: Početna stranica

Slika 6.1 prikazuje početnu stranicu aplikacije, gdje korisnici mogu pregledavati različite kategorije i sadržaje. Sve opcije su dostupne kroz jednostavan i pregledan dizajn, što olakšava korisnicima pronalaženje željenog sadržaja. Na početnoj stranici korisnik može vidjeti navigacijsku traku i tekst zadatka diplomskog rada.

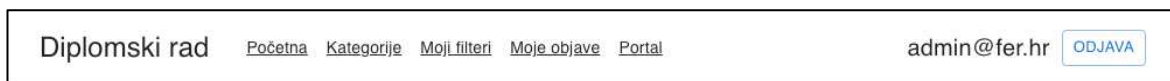


Slika 6.2: Navigacijska traka neprijavljenog korisnika

Slika 6.2 prikazuje navigacijsku traku za neprijavljenog korisnika. Sadrži sljedeće elemente:

1. **Početna:** Vodi korisnika na početnu stranicu
2. **Kategorije:** Vodi korisnika na stranicu kategorija
3. **Prijavi se:** Klikom na ovaj gumb, korisnik će biti preusmjeren na stranicu za prijavu

4. **Registriraj se:** Klikom na ovaj gumb, korisnik će biti preusmjeren na stranicu za registraciju



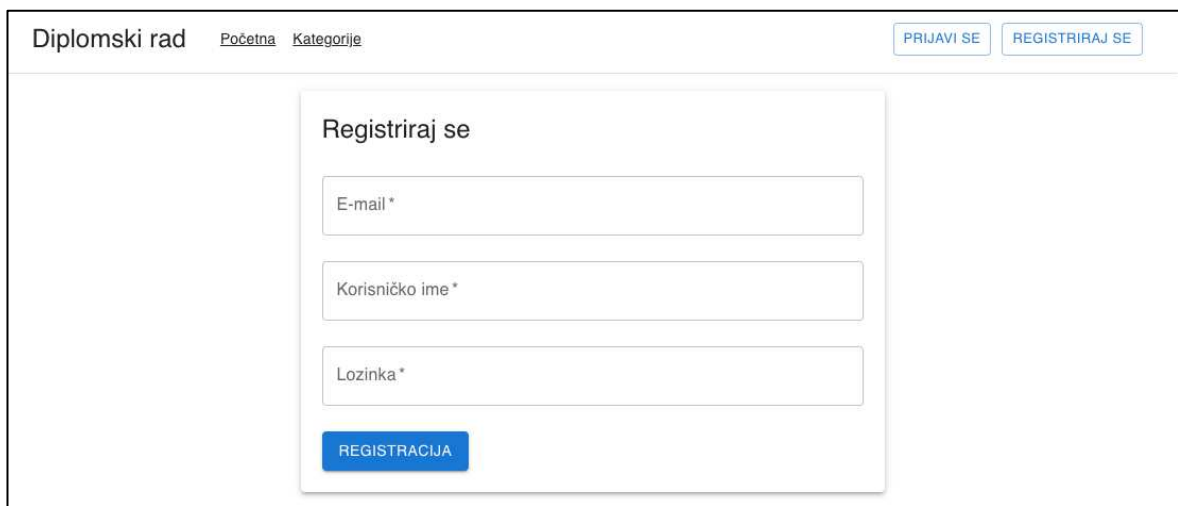
Slika 6.3: Navigacijska traka prijavljenog korisnika

Slika 6.3: Navigacijska traka prijavljenog korisnika prikazuje navigacijsku traku za prijavljenog korisnika. Sadrži sljedeće elemente:

1. **Početna:** vodi korisnika na početnu stranicu
2. **Kategorije:** vodi korisnika na stranicu kategorija
3. **Moji filteri:** omogućava korisnicima da pregledaju i upravljaju svojim filterima
4. **Portal:** prikazuje portal pretplata na spremljene filtere
5. **Odjava:** prijavljeni korisnik ima opciju za odjavu

6.1. Prijava i registracija

Kako bi se iskoristile sve mogućnosti platforme, potrebno je kreirati korisnički račun. Klikom na gumb „Registriraj se“, u navigacijskoj traci, prikazuje se forma za registraciju (Slika 6.4). Forma zahtjeva unos e-mail adrese, korisničkog imena i lozinke. Nakon što su podaci uneseni i validirani, korisnik može završiti proces registracije, klikom na gumb „Registracija“ i biti preusmjeren na formu za prijavu.



Diplomski rad Početna Kategorije

PRIJAVI SE REGISTRIRAJ SE

Registriraj se

E-mail *


Korisničko ime *

Lozinka *

REGISTRACIJA

Slika 6.4: Forma za registraciju

Na formu za prijavu (Slika 6.5) korisnik može doći klikom na gumb „Prijavi se“ ili biti preusmjeren nakon uspješne registracije. Forma zahtjeva unos e-mail adrese i lozinke koje su prethodno definirane tijekom registracije.



Diplomski rad Početna Kategorije

PRIJAVI SE REGISTRIRAJ SE

Prijavi se

E-mail *

admin@fer.hr

Lozinka *

PRIJAVA

Slika 6.5: Forma za prijavu

6.2. Kategorije

Klikom na „Kategorije“, u navigacijskoj traci, prikazat će se stranica koja je podijeljena u tri dijela:


1. **Kategorije:** Prvi dio prikazuje glavne kategorije uz naslov i opis kategorije te ispod gumb za kreiranje novih kategorija koji je vidljiv samo administratorima. Klikom na pojedinu kategoriju možete pristupiti potkategorijama ili objavama unutar te kategorije.
2. **Filteri:** Ovaj odjeljak sadrži dva osnovna polja za filtriranje objava: Naslov i Opis. Korisnici mogu koristiti ove filtere da bi lakše pronašli objave koje ih zanimaju.
3. **Objave:** Ovaj dio prikazuje objave iz svih kategorija uz detalje pojedine objave.

Diplomski rad Početna Kategorije Moji filteri Moje objave Portal admin@fer.hr ODJAVA

Kategorije


Događaji

Otkrijte i podijelite nadolazeće događaje iz različitih sfera interesa. Bilo da su to koncerti, radionice ili sportski susreti, ova kategorija omogućuje povezivanje organizatora i zainteresiranih sudionika.




Poslovi

Pronađite ili ponudite prilike za zaposlenje na različitim pozicijama i sektorima. Bilo da tražite stalni radni odnos, honorarni posao ili studentski posao, ova kategorija nudi širok spektar mogućnosti.




Oglasi

Pronađite ili ponudite različite artikle i usluge. Bilo da želite prodati neku staru stvar, tražite nekretninu za iznajmljivanje ili želite kupiti nešto novo, ova kategorija vam pruža platformu za povezivanje s prodavateljima i kupcima iz vaše zajednice.



Klubovi Obožavatelja

Mjesto za ljubitelje različitih interesa, poput glazbenih izvođača i filmskih zvijezda. Unutar ove kategorije, članovi mogu dijeliti sadržaje, diskutirati o omiljenim temama i povezivati se s istomišljenicima iz svoje zajednice.



+ KREIRAJ NOVU KATEGORIJU

Filter

PRETRAŽI

Objave

Created: 21/08/2023, 14:37:54

Zdravko Čolić u pratnji simfonijskog orkestra - Arena Pula

Doživite magičnu večer s Zdravkom Čolićem i simfonijskim orkestrom u povijesnom ambijentu Arene Pula

Mjesto: Pula

Kontakt informacije: info@eventim.hr

Cijena (€): 25

Created: 21/08/2023, 14:41:56

Prljavo kazalište u Osijeku

Dvorana Gradski vrt

Mjesto: Osijek

Kontakt informacije: info@eventim.hr

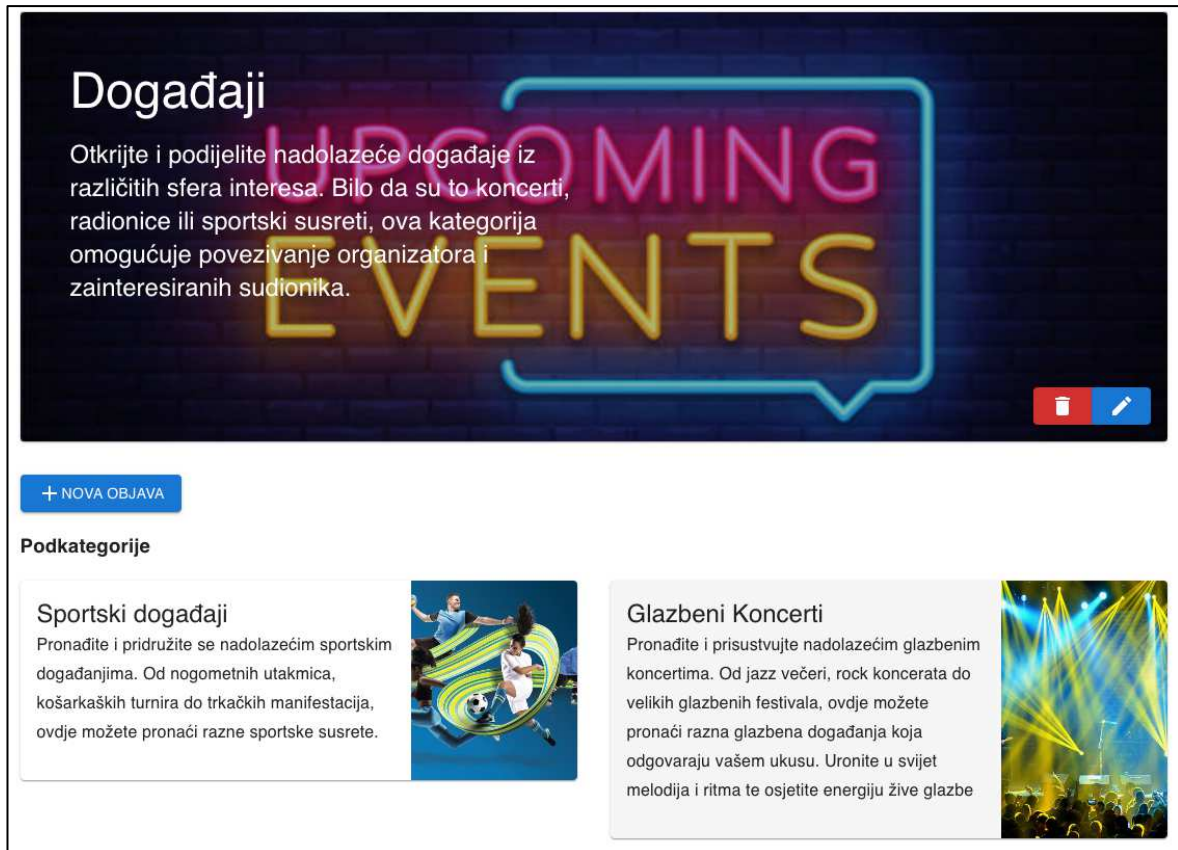
Cijena (€): 20

Datum: 22/12/2023

Slika 6.6: Pregled kategorija

6.2.1. Stranica kategorije i potkategorija

Odabirom kategorije „Događaji“ korisnik će biti preusmjeren na stranicu prikazuje tu kategoriju. Na vrhu stranice (Slika 6.7) prikazana je slika s naslovom i opisom kategorije. Ispod slike su prikazane potkategorije kategorije „Događaji“, a to su „Sportski događaji“ i „Glazbeni Koncerti“.



Slika 6.7: Pregled kategorije

Za kategoriju „Događaji,“ u filteru su nova dodatna polja:

- **Mjesto:** ovo polje omogućava korisnicima da filtriraju događaje prema lokaciji
- **Kontakt informacije:** korisnici mogu vidjeti kako kontaktirati organizatore događaja
- **Cijena:** filtriranje događaja na temelju cijene
- **Datum:** omogućuje korisnicima da odaberu datum na koji traže događaje

6.2.2. Filtriranje i pregled objava

Pregledom potkategorije „Glazbeni Koncerti“ nasljeđuju se polja roditeljske kategorije s dodatna dva polja „Žanr“ i „Izvođač“ (Slika 6.8). Ovi filteri omogućavaju korisnicima da ciljano traže koncerte koji odgovaraju njihovim glazbenim preferencijama.

Filter

Naslov

Opis

Mjesto

Kontakt informacije

Cijena (€) From

Cijena (€) To

Datum From

Datum To

Izvođač

Žanr

☆ SPREMI FILTER

PRETRAŽI

Slika 6.8: Filter potkategorije Glazbeni Koncerti

Slika 6.9 prikazuje objave potkategorije „Glazbeni koncerti“. Za svaku objavu se vidi vrijeme nastanka, naslov, opis i svi detalji polja te potkategorije.

Objave

Created: 21/08/2023, 14:37:54

Zdravko Čolić u pratnji simfonijskog orkestra - Arena Pula

Doživite magičnu večer s Zdravkom Čolićem i simfonijskim orkestrom u povijesnom ambijentu Arene Pula

Mjesto: Pula

Kontakt informacije: info@eventim.hr

Cijena (€): 35

Datum: 20/09/2023

Izvođač: Zdravko Čolić

Žanr: Pop

Created: 21/08/2023, 14:41:56

Prijava kazalište u Osijeku

Dvorana Gradski vrt

Mjesto: Osijek

Kontakt informacije: info@eventim.hr

Cijena (€): 20

Datum: 22/12/2023

Izvođač: Prijava kazalište

Žanr: Rock

Created: 21/08/2023, 14:50:31

Doris @ Arena Zagreb vol. 2 • 26.11.2023

ARENA ZAGREB

Mjesto: Zagreb

Kontakt informacije: info@eventim.hr

Cijena (€): 30

Datum: 26/11/2023

Izvođač: Doris Dragović

Žanr: Pop

Created: 21/08/2023, 17:25:48

VIBE MUSIC FESTIVAL

Spremi se za najjaču završnicu ljeta uz VIBE MUSIC FESTIVAL! U grad Split stižu ti najtraženiji izvođači regije! Prepusti se brutalnom vibeu 22.09.2023. u dvorani Gripe.

Mjesto: Split

Kontakt informacije: info@eventim.hr

Cijena (€): 20

Datum: 22/09/2023

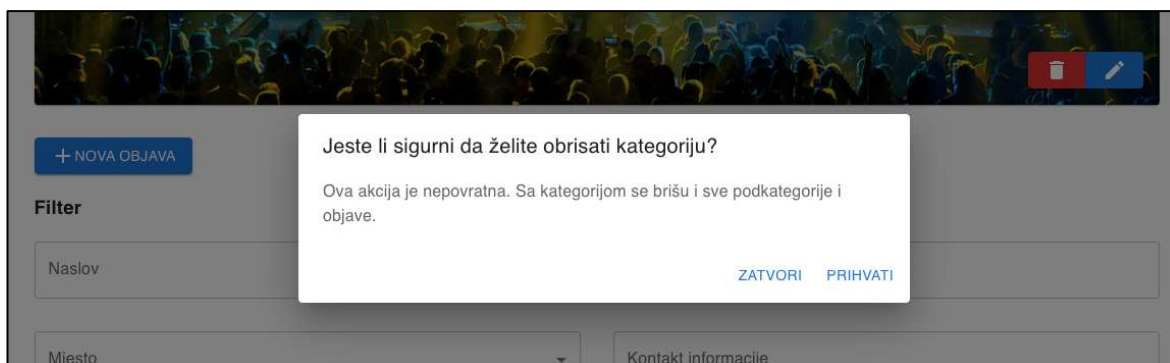
Izvođač: Gazda Paja, Nucci i Voyage

Žanr: Hip-hop i Rap

Slika 6.9: Pregled objava

6.2.3. Uređivanje, kreiranje i brisanje kategorije

Na stranici s detaljima kategorije (Slika 6.7), u donjem desnom kutu, prikazani su gumbi za uređivanje i brisanje kategorije. Klikom na gumb za brisanje, pojavi se prozor s upozorenjem koji ukazuje na to da je akcija nepovratna te da će se brisanjem kategorije izbrisati sve potkategorije i povezani postovi (Slika 6.10).



Slika 6.10: Prozor upozorenja o brisanju kategorije

Forma prikazana na Slika 6.11 služi za uređivanje postojećih kategorija i za stvaranje novih kategorija. Klikom na gumb za uređivanje, koji se nalazi odmah do gumba za brisanje kategorije, moguće je urediti postojeću kategoriju. Stvaranje nove kategorije moguće je klikom na gumb „+ Nova kategorija“ na stranici za pregled glavnih kategorija (Slika 6.6). Forma sadrži sljedeća polja:

1. **Naslov kategorije**
2. **Opis:** tekst koji objašnjava kategoriju
3. **Roditeljska kategorija:** odabir postojeće kategorije kao roditeljske nasljeđuju se polja kategorije pri pretraživanju i unosu nove objave
4. **Slika kategorije URL:** dodavanje URL-a slike koja će predstavljati kategoriju
5. **Polja kategorije** (Slika 6.12): omogućava dodavanje polja kategorije različitih tipova podataka kao što su: tekst, broj, datum, *boolean* vrijednost i tip polja višestrukog odabira gdje se opcije odabira unose kao tekst odvojen zarezom (npr. Ime polja Grad unosimo „Zagreb, Split,...“)

6.3. Moje objave

Klikom na „Moje objave“ u korisničkom izborniku prikazuje se stranica koja sadrži sve objave koje je korisnik stvorio. Ova stranica pruža pregled svih objava s mogućnostima za brisanje. Za svaku kategoriju prikazuje se kategorija oglasa, datum i vrijeme nastanka objave, naslov, opis i sve vrijednosti polja te kategorije.

Diplomski rad

Početna

Kategorije

Moji filteri

Moje objave

Profil

pandzic01@gmail.com

ODJAVA

Moje objave

Kategorija: Glazbeni Koncerti

Created: 21/08/2023, 14:37:54

Zdravko Čolić u pratnji simfonijskog orkestra - Arena Pula

Doživite magičnu večer s Zdravkom Čolićem i simfonijskim orkestrom u povijesnom ambijentu Arene Pula

Mjesto: Pula

Kontakt informacije: info@eventim.hr

Cijena (€): 35

Datum: 20/09/2023

Izvođač: Zdravko Čolić

Žanr: Pop

Kategorija: Glazbeni Koncerti

Created: 21/08/2023, 14:41:56

Prljavo kazalište u Osijeku

Dvorana Gradski vrt

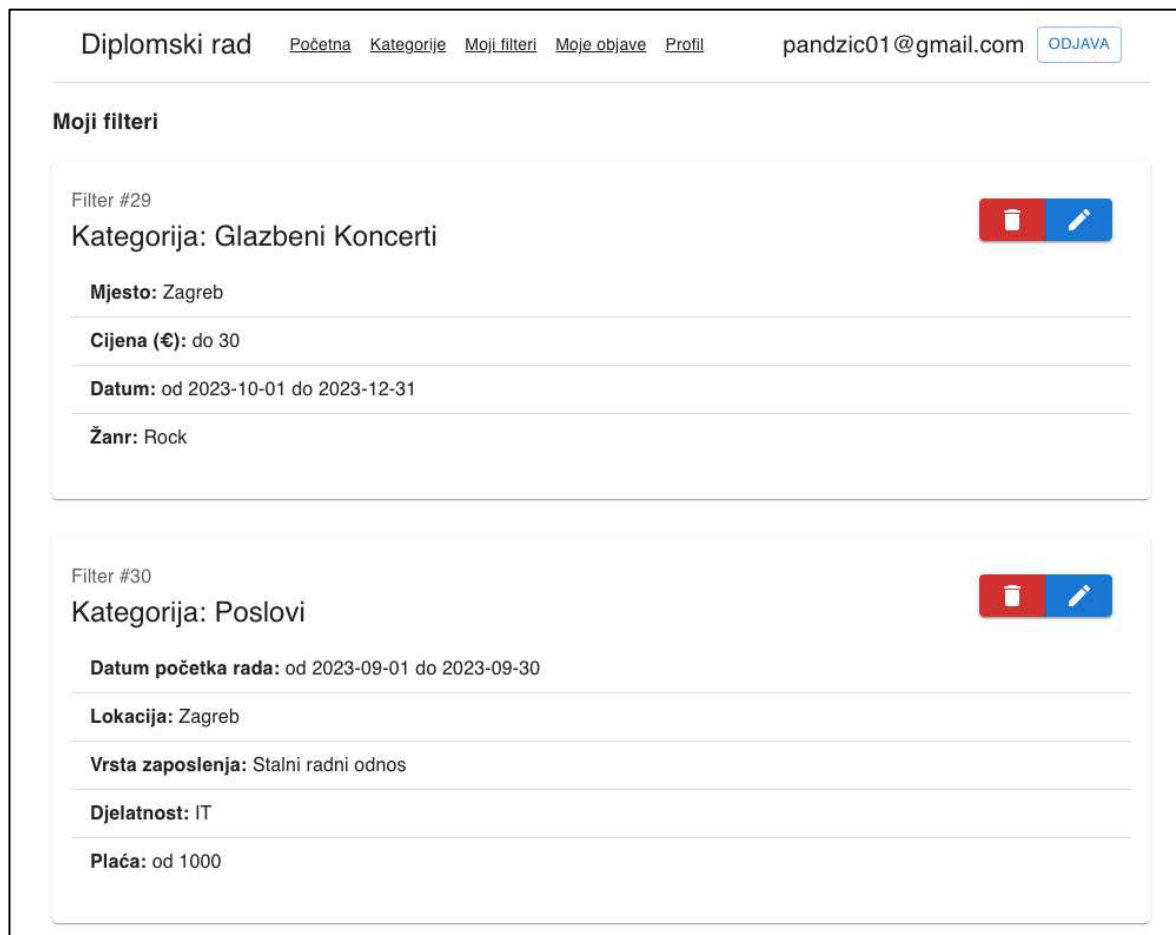
Slika 6.13: Pregled vlastitih objava

6.4. Moji filteri

U poglavlju Filtriranje i pregled objava prikazano je kako filtrirati obavijesti na osnovu vrijednosti polja. Za lakše praćenje i primanje obavijesti za oglase, korisnik može spremiti konfiguraciju filtera. Ova opcija ne samo da olakšava pregled, već omogućuje i automatsko primanje obavijesti pri novom unosu oglasa koji odgovara tim kriterijima.

6.4.1. Spremanje i Pregled Filtera

Klikom na gumb (Slika 6.8) „Spremi filter“ sprema se konfiguracija filtera, a potom se pojavljuje prozor koji obavještava korisnika o uspješnom spremanju. Sve spremljene filtere moguće je pregledavati na stranici „Moji filteri“.



Slika 6.14: Spremljeni filteri

Na primjer, na Slika 6.14 prikazana su dva spremljena filtera:

- Filter #29: Za pronalaženje glazbenih koncerata u Zagrebu, specifično u žanru rock, s cijenama ulaznica do 30 € i datumima od 2023-10-01 do 2023-12-31.
- Filter #30: Za pretragu poslovnih prilika u Zagrebu, za stalne radne odnose u IT sektoru, s početnim datumom rada od 2023-09-01 do 2023-09-30 i minimalnom plaćom od 1000 €.

6.4.2. Uređivanje i brisanje filtera

Svaki spremljeni filter može se jednostavno urediti ili izbrisati, pružajući korisnicima fleksibilnost u prilagodbi filtera prema svojim potrebama. Klikom na gumb za uređivanje, korisnik može pristupiti svakom polju filtera i izvršiti potrebne promjene (Slika 6.15). To može uključivati izmjenu lokacije, djelatnosti, plaće i drugih specifičnih kriterija koji su definirani poljima kategorije.

Diplomski rad Početna Kategorije Moji filteri Moje objave Profil pandzic01@gmail.com OBJAVA

Edit Filter

Naslov Opis

Datum početka rada From Datum početka rada To

01.09.2023. 30.09.2023.

Opis posla

Lokacija Vrsta zaposlenja

Zagreb Stalni radni odnos

Djelatnost Plaća From Plaća To

IT 1000

☆ SPREMI FILTER

Kategorija: Poslovi

Slika 6.15: Uređivanje filtera za posao

Ako korisnik više ne želi pratiti oglase koji odgovaraju određenom filteru, može jednostavno izbrisati filter klikom na gumb za brisanje (Slika 6.14). Time se filter trajno uklanja iz korisnikovog profila, a korisnik više neće primati obavijesti povezane s tim filterom.

6.5. Kreiranje nove objave i primanje obavijesti

Kako bi se kreirala objava u kategoriji potrebno je kliknut na gumb „+ Nova objava“ na stranici pregleda kategorije (Slika 6.7). Time je korisnik preusmjeren na stranicu ispunjavanja forme za objavu gdje se prikazuju sva polja te kategorije i polja naslijeđena od nadređene kategorije. Slika 6.16 prikazuje primjer ispunjavanja takve jedne forme za kategoriju poslovi.

Diplomski rad [Početna](#) [Kategorije](#) [Moji filteri](#) [Moje objave](#) [Profil](#) pandzic01@gmail.com [ODJAVA](#)

Kreiraj objavu

Naslov Objave *

React.js/Node.js (Full-stack Developer) - Softray Solutions

Opis *

Tražimo vještog i iskusnog React.js/Node.js Full Stack developera za pridruživanje našem timu. Osoba će biti odgovorna za raz

Datum početka rada

15.09.2023.

Opis posla

Razvoj i implementacija visoko responzivnih korisničkih sučelja; Pisanje testabilnog, ponovno upotrebljivog i učinkovitog koda

Lokacija

Zagreb

Vrsta zaposlenja

Stalni radni odnos

Djelatnost

IT

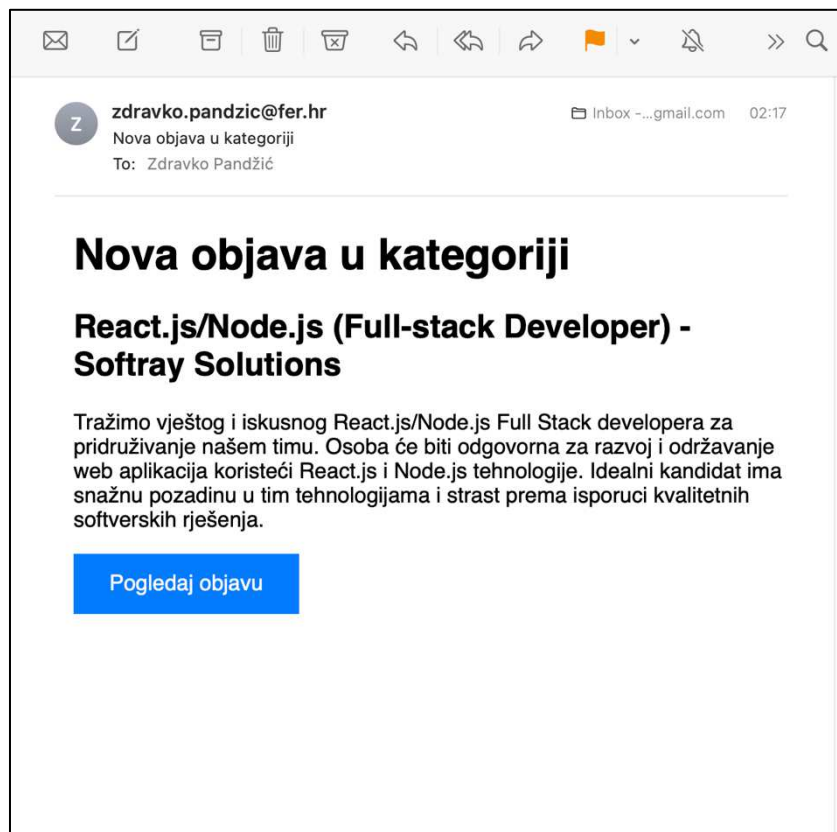
Plaća

1500

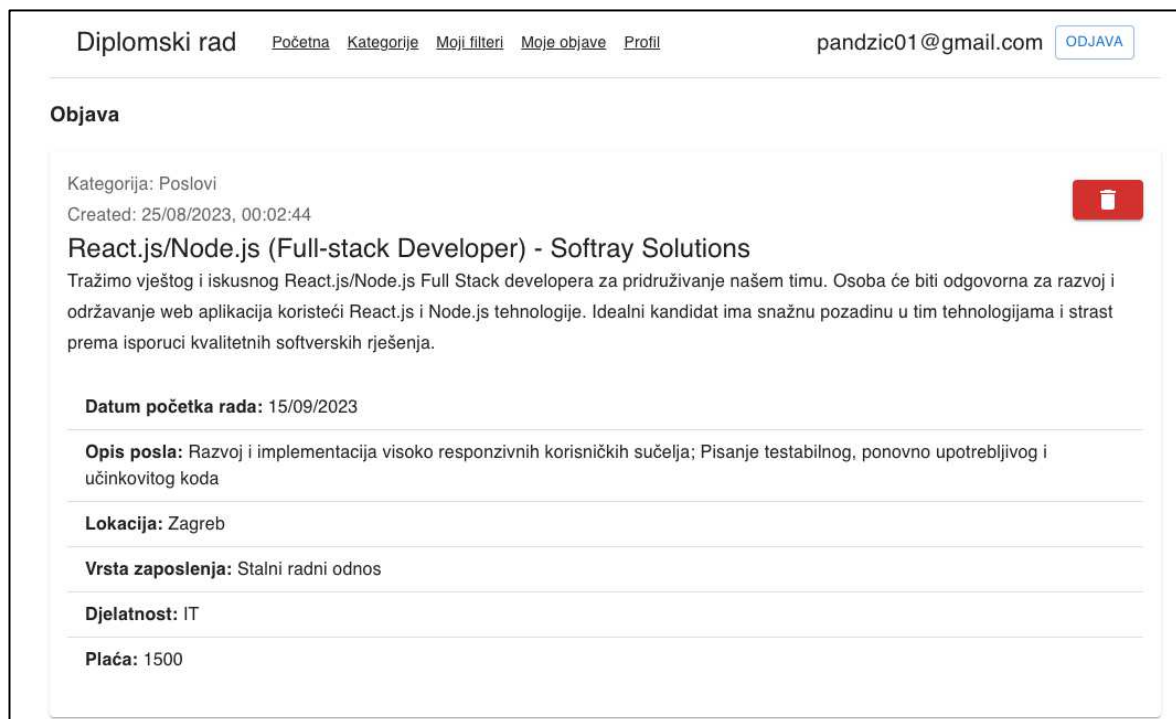
[SUBMIT](#)

Slika 6.16: Forma za kreiranje objave

Nakon objave oglasa, svi korisnici koji imaju spremljene filtere koje odgovaraju ovoj objavi primaju obavijesti na e-mail što je prikazano na Slika 6.17. Sadržaj e-maila je naslov i opis novog oglasa s URL adresom kojom se može pristupiti te se prikazuje cijeli sadržaj sa svim detaljima oglasa (Slika 6.18).



Slika 6.17: Prikaz e-maila nove objave



Slika 6.18: Pregled detalja oglasa dobivenog u e-mailu

6.6. Prikaz portala

Korisnik na stranici svog portala (Slika 6.19) može pregledavati objave koje su u skladu sa spremljenim filterima. Da bi se izvršila dodatna prilagodba, korisnik može aktivirati ili deaktivirati pojedine filtere. Na taj način, prikaz objava na korisnikovom profilu bit će prilagođen njegovim osobnim preferencijama.

Diplomski rad

Početna

Kategorije

Moji filteri

Moje objave

Profil

pandzic01@gmail.com

ODJAVA

Profil

Prikaži obavijesti za filtere:

☐ #30 Poslovi

☒ #29 Glazbeni Koncerti

Kategorija: Glazbeni Koncerti

Created: 21/08/2023, 14:50:31

Doris @ Arena Zagreb vol. 2 • 26.11.2023

ARENA ZAGREB

Mjesto: Zagreb

Kontakt informacije: info@eventim.hr

Cijena (€): 30

Datum: 26/11/2023

Izvođač: Doris Dragović

Žanr: Pop

Slika 6.19: Prikaz portala

7. Upute za instalaciju

Ovo poglavlje pružit će detaljne upute za instalaciju i pokretanje projekta koristeći Docker. Proces se sastoji od nekoliko ključnih koraka: kloniranje izvornog koda s GitHuba, instalacija Docker-a te pokretanje aplikacije.

7.1. Kloniranje repozitorija

Prvi korak je preuzimanje izvornog koda projekta s GitHub stranice. Potrebno je otvoriti naredbeni redak i unijeti sljedeću naredbu:

```
git clone https://github.com/zpandzic/master-thesis.git
```

7.2. Instalacija Docker-a

Za instalaciju Docker-a potrebno je instalirati Docker Desktop aplikaciju sa službene stranice (<https://docs.docker.com/get-docker/>). Aplikacija je kompatibilna s različitim operacijskim sustavima, uključujući Windows, macOS i različite distribucije Linuxa.

7.3. Pokretanje aplikacije

Nakon uspješne instalacije Docker-a i preuzimanja izvornog koda potrebno je otvoriti naredbeni redak i unijeti sljedeće naredbe:

```
cd master-thesis  
docker-compose -build
```

Ovom naredbom će se izgraditi i pokrenuti svi potrebni servisi kako je definirano u **docker-compose.yml** datoteci.

7.4. Pristup aplikaciji

Nakon uspješnog pokretanja, aplikacija se može pregledavati putem web preglednika na adresi `http://localhost:3001`

Zaključak

U okviru ovog diplomskog rada razvijena je generička platforma u oblaku koja omogućuje objavu informacija i oglasa. Istraživanje sličnih i postojećih rješenja, pokazalo je da, iako postoje pojedinačne platforme koje ispunjavaju te potrebe, nema univerzalnog rješenja koje je fleksibilno i skalabilno.

Prednost predloženog rješenja ogleda se u njegovoj prilagodljivosti i sveobuhvatnosti. Za razliku od sličnih rješenja koja su specifična za određene vrste informacija ili oglasa, predložena platforma pruža mogućnost objave i pretrage sadržaja iz različitih kategorija. Korištenjem AWS servisa, Node.js-a i Reacta, uz mikroservisnu arhitekturu, stvoren je sustav koji je lako nadogradiv i održavan. Također, uveden je sustav notifikacija koji omogućuje korisnicima da sami odluče o kojim će temama biti obaviješteni, pružajući visoki stupanj korisničke kontrole.

Iako je rad postigao svoje ciljeve, još uvijek postoji prostor za daljnje istraživanje i napredak. To uključuje povezanost s drugim platformama i servisima te širenje na nove geografske lokacije. U konačnici, ova platforma ne samo da rješava trenutačne izazove u objavi informacija i oglasa, već postavlja temelje za inovativna rješenja koja će oblikovati budućnost digitalnog oglašavanja i informiranja.

Literatura

- [1] P. G. D. Group, “PostgreSQL,” *PostgreSQL*, Aug. 26, 2023.
<https://www.postgresql.org/> (accessed Aug. 26, 2023).
- [2] “About,” *Node.js*. <https://nodejs.org/en/about> (accessed Aug. 16, 2023).
- [3] “5 Main Node.js Advantages and Disadvantages | EPAM Startups & SMBs.”
<https://anywhere.epam.com/business/node-js-pros-and-cons> (accessed Aug. 16, 2023).
- [4] “React (software),” *Wikipedia*. Aug. 16, 2023. Accessed: Aug. 17, 2023. [Online].
Available:
[https://en.wikipedia.org/w/index.php?title=React_\(software\)&oldid=1170657381](https://en.wikipedia.org/w/index.php?title=React_(software)&oldid=1170657381)

Sažetak

U današnjem digitalnom dobu, brzi protok informacija i oglasa postaje sve važniji. Postojeće platforme često su usmjerene na specifične vrste informacija, ostavljajući prazninu za sveobuhvatno i prilagodljivo rješenje. Ovaj diplomski rad predstavlja generičku platformu u oblaku za objavu informacija i oglasa. Razvijena s ciljem pružanja prilagodljivog i skalabilnog rješenja, platforma koristi mikroservisnu arhitekturu i sabirnice poruka, čime se postiže visok stupanj korisničke kontrole i prilagodljivosti.

Za potrebe ovog rada, analizirane su postojeće platforme poput Njuškala, Eventima i MojPosao, istražujući njihove prednosti i nedostatke. Temeljene na ovoj analizi, predložena je nova arhitektura koja korisnicima pruža širok spektar funkcionalnosti. Platforma je izgrađena korištenjem suvremenih tehnologija kao što su AWS servisi, Node.js i React, osiguravajući tako robusnost i održivost sustava.

Osim toga, jedan od ciljeva je osigurati intuitivno korisničko iskustvo koje omogućuje laku navigaciju i interakciju. Arhitektura je dodatno osmišljena da omogući lako skaliranje i nadogradnju, pružajući temelj za buduće proširenje i integraciju s drugim servisima.

Predložena platforma ne samo da rješava trenutačne izazove u objavi informacija i oglasa, već postavlja temelje za buduće inovativna rješenja u ovom području, uključujući mogućnost integracije s drugim platformama i geografskom ekspanzijom.

Ključne riječi: Generička platforma, Oblak, Mikroservisi, Sabirnica poruka, Informacijski portal, Oglasi, Notifikacije, React, Node.js, AWS

Summary

In today's digital age, the rapid flow of information and advertisements is becoming increasingly important. Existing platforms are often focused on specific types of information, leaving a gap for a comprehensive and customizable solution. This thesis presents a generic cloud platform for publishing information and advertisements. Developed with the goal of providing a customizable and scalable solution, the platform uses microservice architecture and message buses, which achieves a high degree of user control and adaptability.

For the purposes of this work, existing platforms such as Njuškalo, Eventima and MojPosao were analyzed, investigating their advantages and disadvantages. Based on this analysis, a new architecture is proposed that provides users with a wide range of functionality. The platform was built using modern technologies such as AWS services, Node.js and React, thus ensuring the robustness and sustainability of the system.

In addition, one of the goals is to provide an intuitive user experience that allows for easy navigation and interaction. The architecture is further designed to enable easy scaling and upgrading, providing a foundation for future expansion and integration with other services.

The proposed platform not only solves the current challenges in publishing information and ads, but also lays the foundation for future innovative solutions in this area, including the possibility of integration with other platforms and geographic expansion.

Keywords: Generic platform, Cloud, Microservices, Message bus, Information portal, Ads, Notifications, React, Node.js, AWS