Izrada Cloud aplikacije E-sandučić

Marić, Frane

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU
STRUČNI PRIJEDIPLOMSKI STUDIJ RAČUNARSTVO

Frane Marić, 0313023760

IZRADA CLOUD APLIKACIJE E-SANDUČIĆ

ZAVRŠNI RAD

Čakovec, rujan 2024.

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU
STRUČNI PRIJEDIPLOMSKI STUDIJ RAČUNARSTVO

Frane Marić, 0313023760

IZRADA CLOUD APLIKACIJE E-SANDUČIĆ

DEVELOPMENT OF A CLOUD APPLICATION E-INBOX

ZAVRŠNI RAD

Mentor:
dr. sc. Bruno Trestenjak, prof. struč. stud.

Čakovec, rujan 2024.

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU

PRIJAVA TEME I OBRANE ZAVRŠNOG/DIPLOMSKOG RADA

Stručni prijediplomski studij:

☒ Računarstvo ☐ Održivi razvoj ☐ Menadžment turizma i sporta

Stručni diplomski studij Menadžment turizma i sporta: ☐

Pristupnik: FRANE MARIĆ, JMBAG: 0313023760
(ime i prezime)

Kolegij: SLOŽENI APLIKACIJSKI PROGRAMI
(na kojem se piše rad)

Mentor: dr. sc. BRUNO TRSTENJAK, prof. struč. stud.
(ime i prezime, zvanje)

Naslov rada: IZRADA CLOUD APLIKACIJE E-SANDUČIĆ

Naslov rada na engleskom jeziku: DEVELOPMENT OF A CLOUD APPLICATION E-INBOX

Članovi povjerenstva:
1. dr.sc. Sanja Brekalo, prof. struč. stud., predsjednik
(ime i prezime, zvanje)
2. Jurica Trstenjak, v. pred., član
(ime i prezime, zvanje)
3. dr. sc. Bruno Trstenjak, prof. struč. stud., mentor
(ime i prezime, zvanje)
4. Marija Miščančuk, v. pred., zamjenski član
(ime i prezime, zvanje)

Broj zadatka: 2023-RAČ-8

Kratki opis zadatka: Izrada web aplikacije u oblaku koja će omogućavati prijem zaprimanje PDF dokumenata i računa u jedinstvenu bazu na serveru. U aplikaciji će se izraditi "pametan" algoritam prepoznavanja dokumenata. Na osnovi prepoznatih podataka u dokumentu, bar koda uplatnice ili nekog drugog podatka, aplikacija će sortirati dokumente i zapisivati u mape na serveru te podatke u jednostavnu relacijsku bazu. Aplikacija će biti izrađena u Spring Boot tehnologiji bazirano na Java objektnom programskom jeziku. U aplikaciji će bit implementirani administrativni dio koji će omogućiti provjeru zaprimljenih podataka i dokumenata.

Datum: 20.9.2024. Potpis mentora: dr.sc. Bruno Trstenjak, prof. struč. stud.

ZAHVALA

Ovaj rad posvećujem svojim roditeljima i sestri te im se zahvaljujem na beskrajnom strpljenju i podršci.

Hvala mentoru dr. sc. Bruni Trstenjaku prof. stručn. stud. na izvrsnom vodstvu u pisanju rada.

SAŽETAK

Ovaj rad prikazuje izradu e-sandučić aplikacije pomoću *Spring boot*-a. *Spring boot* je popularan *Java* okvir koji olakšava stvaranje samostalnih produkcijskih Spring aplikacija. Aplikacija se služi *PostgreSQL* bazom podataka, a to je objektno-relacijski sustav baze otvorenog koda koji ima snažnu reputaciju pouzdanosti, robusnosti značajki i performansi. Za čitanje i uređivanje baze podataka koristio se *pgAdmin*. *pgAdmin* je alat otvorenog koda za upravljanje *PostgreSQL* bazom. Može se pokrenuti kao web ili desktop aplikacija. Za objavu aplikacije na internetu koristi se *Heroku* platforma. *Heroku* je platforma kao usluga (PaaS) temeljena na kontejneru. Programeri koriste *Heroku* za implementaciju, upravljanje i skaliranje modernih aplikacija te kao jednostavan način za postavljanje njihovih aplikacija na tržište. Kako bi se mogli spremati dokumenti implementirao se amazonov *Amazon S3*. *Amazon S3* je usluga za pohranu objekata koja nudi dostupnost podataka, sigurnost i performanse.

Unutar aplikacije postoje 3 vrste korisnika: korisnik, pošiljatelj i administrator. Korisnik ima mogućnost prijave i pregled svog korisničkog sandučića. Pošiljatelj ima pravo na prijavu, pregled svog korisničkog sandučića te slanje dokumenata. Administrator ima sve mogućnosti kao korisnik i pošiljatelj, a uz to i pravo na pregled i uređivanje svih korisnika i pošiljatelja. Kako bi se korisnici prijavili moraju se prvo registrirati. Za provjeru točnosti informacija korisniku se šalje poruka na mail koja sadrži poveznicu s kojom se otključava njihov račun te im daje mogućnost prijave i ulaska u aplikaciju. Kod slanja dokumenata korisnicima, aplikacija pretvara te dokumente najprije u sliku, a zatim u binarnu bitmapu unutar koje traži *pdf417* barkod. Kada pronađe barkod, pročita ga, podjeli tekst u dijelove i pošalje cijeli dokument korisniku koji je upisan unutar tog barkoda. Kada korisnik zaprimi dokument, on ima mogućnost preuzimanja i brisanja tog dokumenta.

Ključne riječi: *Java, Spring boot, PostgreSQL, pgAdmin, Heroku, Amazon S3, pdf417*

ABSTRACT

This paper shows the creation of an e-mail application using *Spring boot*. *Spring boot* is a popular *Java* framework that makes it easy to create stand-alone production Spring applications. The application uses the *PostgreSQL* database, *PostgreSQL* is an open source object-relational database system that has a strong reputation for reliability, robustness of features and performance. *pgAdmin* was used to read and edit the database. pgAdmin is an open source tool for PostgreSQL database management. It can be run as a web or desktop application. The Heroku platform is used to publish the application to the Internet. Heroku is a container-based platform as a service (paaS). Developers use Heroku to deploy, manage and scale modern applications and provide an easy way to bring their applications to market. In order to be able to save documents, Amazon's Amazon S3 was implemented. Amazon S3 is an object storage service that offers data availability, security, and performance.

There are 3 types of users within the application: user, sender and administrator. The user has the option of logging in and viewing his user inbox. The sender has the right to log in, view his user inbox and send documents. The administrator has all the possibilities of users and senders with the right to view and edit all users and senders. In order to log in, users must first register. To check the accuracy of the information, a message is sent to the user's e-mail containing a link that unlocks their account and gives them the option of logging into the application. When sending documents to users, the application converts these documents into an image and from an image into a binary bitmap within which it searches for a pdf417 barcode within that bitmap. When it finds a barcode, it reads it, divides the text into parts and sends the entire document to the user who entered in that barcode. When the user receives a document, he has the option of downloading and deleting that document.

Keywords: *Java, Spring boot, PostgreSQL, pgAdmin, Heroku, Amazon S3, pdf417*

Popis korištenih kratica

SQL Strukturni jezik upita

OIB osobni identifikacijski broj

HTML jezik za označavanje hiperteksta

Amazon S3 Amazon jednostavni servis pohrane

CSS kaskadni listovi stilova

MVC model-pogled-kontroler

UTF-8 format unikoda transformacije 8

PaaS Platforma kao usluga

IP internet protokol

HTTP Protokol prijenosa hiperteksta

JVM Java Virtualna Mašina

SADRŽAJ

1. UVOD

U današnjem digitalnom dobu, sve veći broj poslovnih procesa prelazi u digitalan oblik, uključujući i slanje računa. Tradicionalan način slanja računa papirima podložan je kašnjenju i pogreškama zbog raznih tehničkih i ljudskih faktora kao što su pogreške u pošti, fizička oštećenja dokumenata, gubitak zbog neorganiziranosti i slično. Izradom aplikacije za slanje i pohranu računa bi se cijeli taj proces mogao automatizirati i ubrzati te bi se mogla smanjiti mogućnost ljudske pogreške pri ručnom unosu podataka.

Aplikacija koja ujedinjuje sve račune na jedno mjesto omogućuje pošiljateljima da brzo, jednostavno i sigurno pošalju sve dokumente te im daje uvid i potvrdu gdje, kada i jesu li ti dokumenti stigli na primateljevu adresu. Time se osigurava transparentnost i smanjuje rizik od gubitka važnih informacija.

S druge strane, primateljima prikazuje sve potrebne račune i dokumente na jednom mjestu što olakšava da ne budu zagubljeni ili nepročitani. Još jedan benefit ovakve aplikacije je bolja organizacija financija, lakše praćenje prethodno plaćenih računa te pristup povijesnim podatcima što olakšava bolje planiranje budućih obveza. Ujedno bi im omogućilo da primaju važne dokumente kada nisu kod kuće te da iste mogu otvarati i čitati bilo kada i bilo gdje.

Osim navedenih prednosti, prelazak na digitalno slanje i pohranu računa ima i značajan pozitivan utjecaj na očuvanje okoliša. Smanjenjem potrebe za tiskanim dokumentima smanjila bi se potrošnja papira kao što bi se uz manje poštanskih vozila u opticaju smanjila potrošnja goriva koje zagađuje okoliš.

Cilj ovog završnog rada je razvoj aplikacije čije će funkcionalnosti biti detaljnije opisane u naredim poglavljima.

2. CILJ ZAVRŠNOG RADA

Cilj ovog završnog rada je razvoj web aplikacije u oblaku za slanje i zaprimanje PDF dokumenata i računa u jedinstvenu bazu na serveru. U aplikaciji se izradio „pametan“ algoritam prepoznavanja dokumenata. Na osnovi prepoznatih podataka u dokumentu, bar koda uplatnice ili nekog drugog podatka, aplikacija sortira dokumente i zapisuje ih u mape na serveru te podatke u jednostavnu relacijsku bazu. Aplikacija je izrađena u Spring boot tehnologiji baziranoj na Java objektnom programskom jeziku. U aplikaciji je implementiran administrativni dio koji omogućuje provjeru zaprimljenih podataka i dokumenata.

U radu je detaljno objašnjen način razvoja aplikacije te korišteni alati za njenu izradu. Završni rad pruža uvid u tehničke aspekte razvoja, kao i potencijalne koristi koje ova aplikacija donosi u kontekstu modernizacije poslovnih procesa i poboljšanja korisničkog iskustva.

3. KORIŠTENE TEHNOLOGIJE

3.1. JAVA

Java je objektno orijentirani programski jezik koji se koristi u računarstvu, poslovnim sustavima i web razvoju, a čija mogućnost rada na bilo kojem sustavu osigurava da se kod napisan u Javi može izvršiti na širokom rasponu hardvera i operativnih sustava. To se postiže upotrebom JVM-a, koji pretvara kod u bajtni kod koji JVM interpretira ciljanom sustavu. Upotrebom JVM-a eliminira se potreba za prilagođavanjem aplikacija specifičnim platformama što znatno olakšava razvoj aplikacija. Java je prvi put objavljena 1995. godine od tvrtke Sun Microsystems te je kasnije prodana Oracle-u. Javina podrška za višenitnost omogućuje učinkovito rukovanje velikim skupovima podataka i istodobnim procesima, koji su često kritični u radu aplikacija. Javino automatsko upravljanje memorijom putem ugrađenog sustava prikupljanja smeća smanjuje rizik od curenja memorije te olakšava programerima da se ne moraju brinuti o oslobađanju memorije [6][19].

3.2. SPRING BOOT

Spring boot je okvir otvorenog koda koji pojednostavljuje razvoj aplikacija u Javi. Koristi se za izradu Spring aplikacija. Spring boot eliminira velik dio standardnog koda i

konfiguracije potrebne za tradicionalne Java aplikacije. Jedna od prednost Spring boot-a je njegova sposobnost da automatski konfigurira aplikaciju na temelju ovisnosti koje koristi. Sprong boot eliminira potrebu za zasebnom instalacijom web poslužitelja tako što ima ugrađene Tomcat i Jetty instance. Kako bi dodatno olakšao programerima, Spring boot automatski prepoznaje uobičajene scenarije (kao što je povezivanje s bazom) te koristi odgovarajuće postavke. Jedan od primarnih ciljeva Spring boot-a je pružiti brže i široko dostupno početno iskustvo za razvoj. Za brz i lagan početak razvoja Spring aplikacija, napravljena je stranica Spring initializr unutar koje se može jednostavno odabrati aplikacija koju se želi razvijati pa će generirati projekt s imenom i svim ovisnostima koje je korisnik odabrao [1].

3.3. SQL (STRUCTURED QUERY LANGUAGE)

SQL je programski jezik za pohranu i obradu informacija u bazu podataka. Pomoću SQL-a moguće je izvršiti širok spektar operacija nad podatcima. SQL se smatra temeljnim alatom za rad s bazama podataka u gotovo svim informatičkim sustavima.

Informacije unutar baze podataka se spremaju se u tablice koje se sastoje od stupaca i redova. Svaki stupac pokazuje određenu vrstu podataka, dok svaki red predstavlja pojedinačne zapise ili entitete.

Relacijska baza podataka sastoji se od više tablica koji su međusobno povezane ključevima. Primarni ključevi koriste za identifikaciju svakog retka dok se vanjski ključevi koriste za vezu između tablica.

SQL naredbe podijeljene su u dvije veće skupine:

- DDL (eng. *Data Definition Language*) je dio SQL jezika za definiranje struktura baza podataka. DDL-om možemo kreirati, mijenjati i brisati tablice, atribute, baze podataka.
- DML (eng. *Data Manipulation Language*) služi za manipulaciju podatcima koji se nalaze u strukturi kreiranog od strane DLL-a. DML omogućava umetanje podataka u tablice, ispravak podataka, brisanje podataka te projekciju podataka[9][18].

3.4. HTML

HTML je kod koji se koristi za strukturiranje web stranice i njenog sadržaja. HTML se sastoji od niza elemenata koji upućuju web preglednik na način kako će prikazati tekst koji

slijedi nakon tog elementa. Omogućava organizaciju sadržaja kao što su tekst, slike, videozapisi, veze, liste, tablice i drugi multimedijski elementi [12].

Glavni dijelovi elementa su:

- početna oznaka: sastoji se od naziva elementa umotanog u otvarajuće i zatvarajuće šiljaste zagrade. U nekim slučajevima, početna oznaka može sadržavati i dodatne atribute koji definiraju informacije o elementu, poput njegovog izgleda, ponašanja ili povezanosti s drugim elementima.
- završna oznaka: isto kao i početna, sastoji se od naziva elementa umotanog u otvarajuće i zatvarajuće šiljaste zagrade uz kosu crtu ispred naziva elementa Završna oznaka osugurava da preglednik zna gdje određeni element završava.
- sadržaj: sadrži sadržaj elementa. Najčešće samo tekst ili sliku. U slučaju slike, sadržaj može biti zamjenjen atributima koji definiraju izvor slike i alternativni tekst.

3.5. JAVASCRIPT

JavaScript je objektno orijentiran skriptni jezik za razvoj web aplikacija na klijentu. Nastao je 1995. godine te je temeljen na prototipu koji podržava objektno orijentirane stilove. Aplikacije napisane JavaScript-om odvijaju se na korisnikovom računalu unutar web preglednika (nemaju komunikaciju sa serverom). Dinamičke mogućnosti JavaScript-a uključuju konstrukciju objekata, propise varijabilnih parametara, varijable funkcija i oporavak izvornog koda. Popularnost JavaScript-a dolazi iz sposobnosti za stvaranjem dinamičkih korisničkih sučelja bez potrebe za ponovnim osvježavanjem stranice. Jedna od ključnih značajki JavaScript-a je njegova podrška za objektno orijentirano programiranje. Funkcije JavaScript-a tretiraju se kao objekti, što omogućava funkcijama da budu pohranjene u varijablama, prosljeđivanje drugim funkcijama ili vraćene kao rezultat iz funkcija. JavaScript omogućava dinamičke parametre , što znaci da funkcije mogu poprimiti različit broj argumenata prilikom poziva. Uz to, varijable nisu strogo tipizirane, pa mogu mijenjati svoje tipove tijekom izvođenja programa. [13].

3.6. AJAX

Ajax je asinkrona tehnologija koja služi za razvoj bržih, boljih i interaktivnih web aplikacija. Ajax je kombinacija HTML-a, CSS-a, JavaScript-a i XML-a. Moć Ajax-a dolazi iz

mogućnosti da se šalju i primaju podatci sa servera te prikazuju korisniku bez potrebe za osvježavanjem stranice. Ova funkcionalnost značajno poboljšava brzinu i korisničko iskustvo jer omogućava ažuriranje samo dijela stranice bez utjecaja na ostatak sadržaja. Korištenjem Ajax-a, ažuriranje korisničkih sučelja mogu se obaviti odmah nakon što dobije Ajax dobije povratnu informaciju od servera. Na primjer, ako korisnik promijeni odabir u padajućem izborniku, Ajax može odmah poslati zahtjev serveru i ažurirati samo relevantan dio stranice. Korištenjem Ajax-a s tehnologijama poput SessionStorage, moguće je privremeno pohraniti podatke na korisnikov uređaj bez potrebe za stalnom komunikacijom sa serverom.[14].

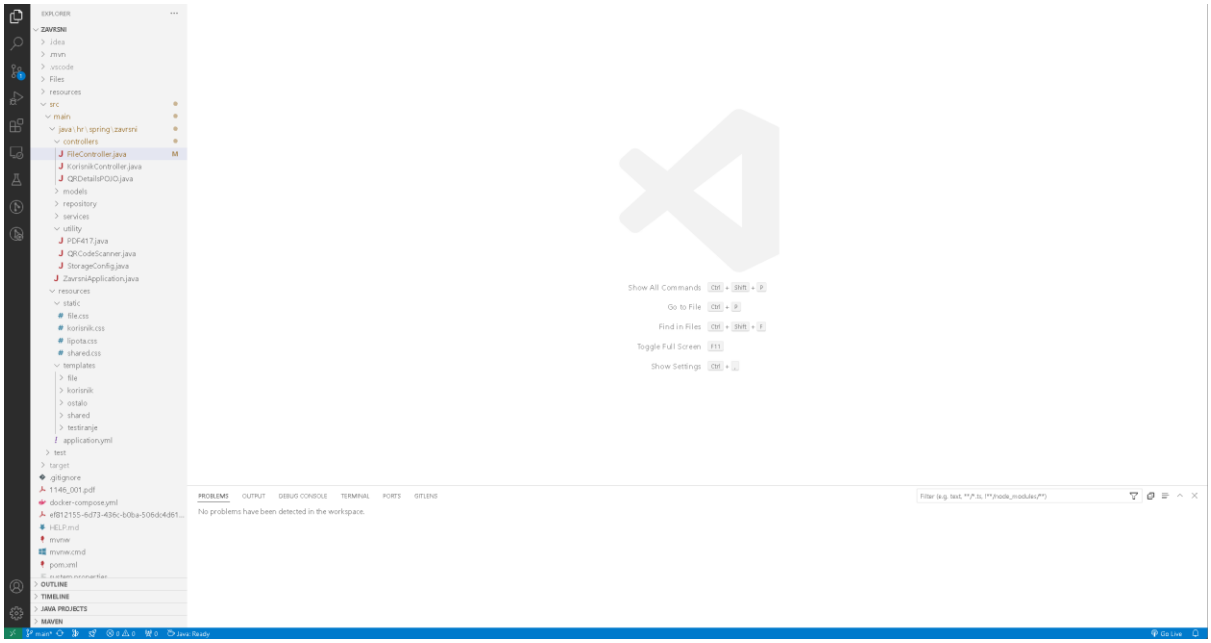
4. ALATI KORIŠTENI ZA IZRADU APLIKACIJE

4.1. VISUAL STUDIO CODE

Visual Studio Code je moćan uređivač koda koji je proizveo Microsoft. Dostupan je za Windows, macOS i Linux. Ima bogati sustav za ekstenzija te se može koristiti za mnoge programske jezike kao što su C#, C++, Java, Python i mnogi drugi. Ove ekstenzije ne pružaju samo osnovnu podršku za isticanje sintakse, već uključuju napradne značajke poput automatskog dovršavanja koda, integraciju s alatima za testiranje, analizu koda i podršku za upravljanje verzijama. Za razliku od Visual Studia, ne koristi toliko resursa pa manje opterećuje računalo te se može brzo i jednostavno pokrenuti [8].

Slika 1 prikazuje naslovnu stranicu Visual Studio Code-a. Na lijevoj strani su traka aktivnosti i preglednik datoteka. Pomoću trake aktivnosti korisnici mogu pristupiti izbornicima skinutih ekstenzija, dok se pomoću preglednika datoteka može intuitivno pregledavati i upravljati datotekama unutar otvorenog projekta.

Slika 1. Visual Studio Code



Izvor: Autor

4.2. AMAZON S3

Amazon Simple Storage Service služi za pohranu objekata koja nudi skalabilnost, dostupnost podataka, sigurnost i performanse. [5] Milijuni korisnika pohranjuju, upravljaju analiziraju i štite bilo koju količinu podataka za gotovo sve slučajeve upotrebe. Amazon S3 koristi model pohrane koji replicira podatke na više geografskih lokacija, čime se minimizira rizik od gubitka podataka.

Sigurnost je jedan od ključnih elemenata Amazon S3 usluge. Korisnici imaju mogućnost konfigurirati različite sigurnosne mjere poput enkripcije podataka i kontrole pristupa putem sustava AWS (Amazon Web Service).

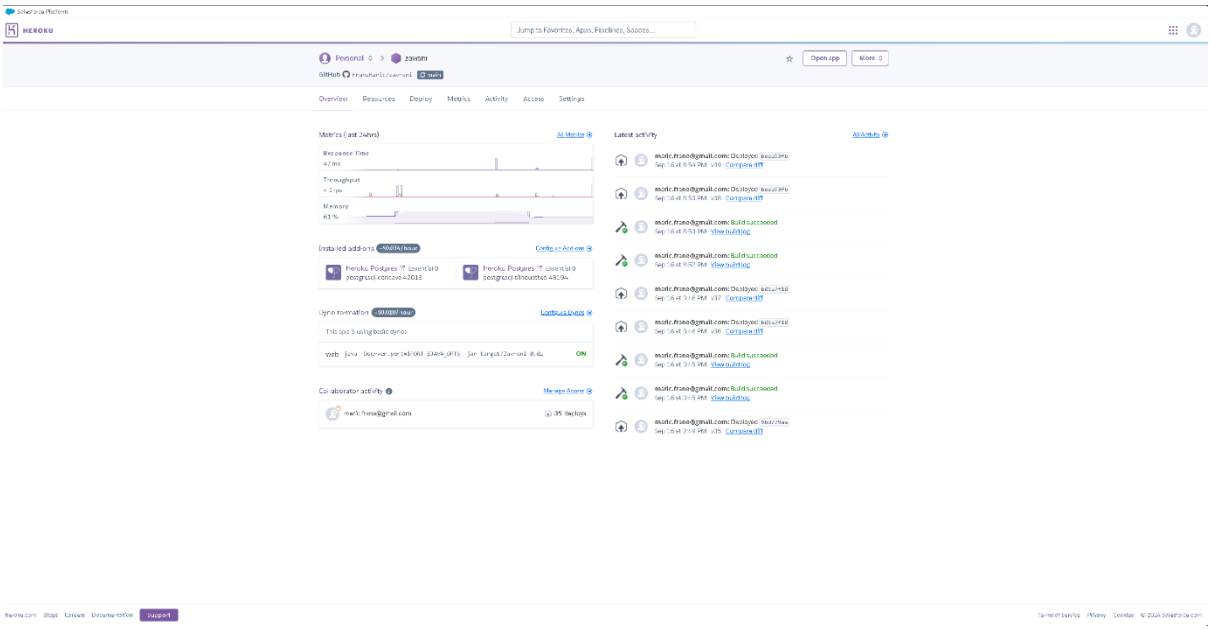
Za korisnike koji rade s velikim količinama podataka, Amazon S3 omogućava analizu podataka. Pomoću funkcionalnostima kao što su S3 Select i S3 Inventory, korisnici mogu izdvajati specifične podatke iz svojih objekata i analizirati ih [11].

4.3. HEROKU

Heroku je PaaS koja služi mnogim programerima za objavljivanje njihovih web aplikacija. Cilj Heroku-a je omogućiti programerima da se usredotoče na razvoj aplikacije bez briga oko održavanje servera, hardvera ili infrastrukture. Filozofija tima koji je razvio Heroku je da najbolje aplikacije dolaze od produktivnih programera te su se zbog toga usredotočili na izradu platforme koja će sa svojim alatima omogućiti programerima najbolje iskustvo pri razvoju aplikacija. Programeri koji koriste Heroku ne moraju učiti kako održavati server ili upravljati mrežom. Heroku uklanja prepreke programerima kako bi mogli bezbrižno razvijati aplikacije. Uz jednostavnost implementacije i proširenja, Heroku nudi automatsko skaliranje aplikacja ovisno o opterećenu. Pri naglim skokovima korisnika, Heroku može prilagoditi broj potrebnih resursa kako bi aplikacija mogla ispravno raditi bez prekida. Osim toga, Heroku nudi alate za nadzor i praćenje performansi aplikacija. Kroz svoju platformu, Heroku nudi programerima uvid u korištenje resursa, vrijeme odgovora aplikacija i detaljne logove. Pomoću svoje integracije sa sustavima za kontrolu verzija, Heroku omogućuje brzo i jednostavno vraćanje na prethodne verzije aplikacija [4].

Slika 2 prikazuje nadzornu ploču za ovu aplikaciju unutar Heroku platforme. Nadzorna ploča prikazuje aktivnost aplikacije (npr. kada i tko je promijenio verziju aplikacije) i metriku s kojom se može pratiti memorija i vrijeme odgovora aplikacije unutar zadnjih 24 sata.

Slika 2. Heroku nadzorna ploča



Izvor: Autor

4.4. pgAdmin4

pgAdmin4 je vodeći open source alat za upravljanje PostgreSQL baze podataka. Dizajniran je kako bi zadovoljio potrebe i početnika i iskusnih korisnika, pružajući moćno grafičko sučelje koje pojednostavljuje stvaranje, održavanje i korištenje objekata baze podataka [3]. Omogućuje praćenje i upravljanje više baza i EDB Advanced Server poslužitelja baza podataka, lokalnih i udaljenih, putem jednog grafičkog sučelja za jednostavno stvaranje i upravljanje objektima baze podataka [10].

Može se pokrenuti kao desktop ili web aplikacija. Desktop način rada instaliran je kao samostalna aplikacija koju koristi isti korisnik operativnog sustava, dok se poslužiteljskom načinu rada može pristupiti preko mreže, što omogućuje da ga koristi više korisnika. Obje implementacije načina rada slijede pristup arhitekture od 3 razine [15][16].

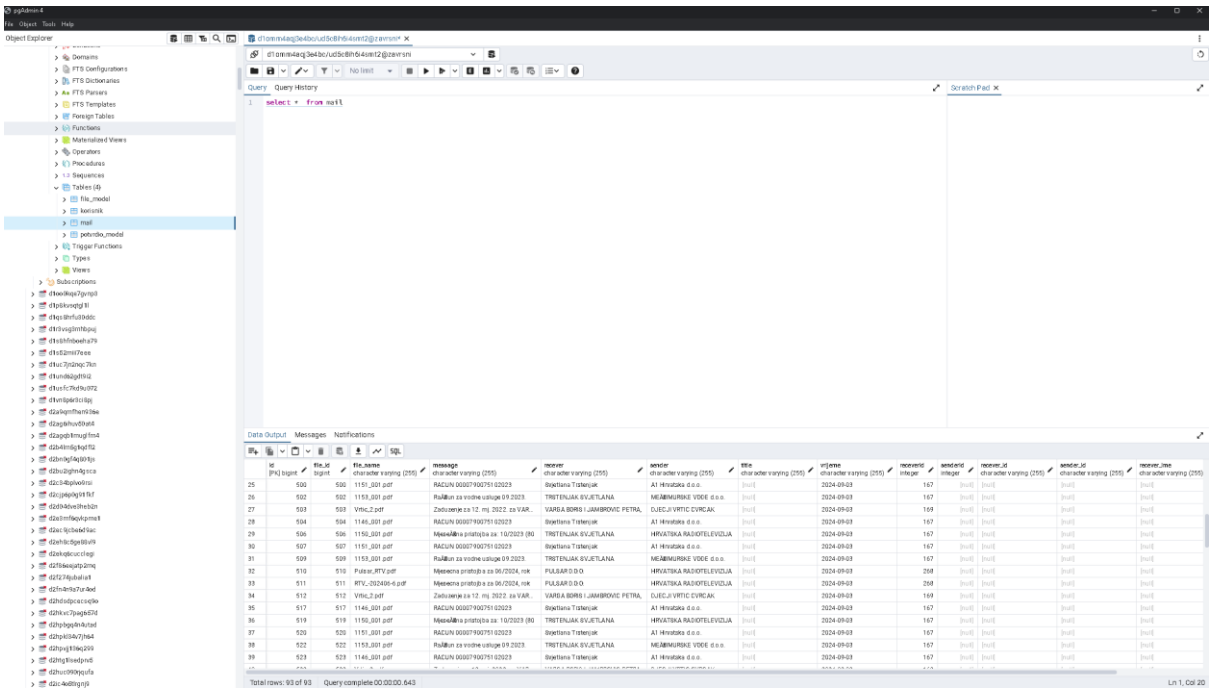
Aplikacija se može pokrenuti na više operacijskih sustava: Windows, Linux i macOS.

4.5. POSTGRESQL

PostgreSQL je besplatan ali moćan sustav otvorenog koda za baze podataka. Povijest postgreSQL-a traje već više od 35 godina te je zbog toga dobio snažnu reputaciju pouzdanosti i performansi. PostgreSQL dolazi s mnogo značajki koje imaju cilj pomoći razvojnim programerima u izradi aplikacija, administratorima u zaštiti integriteta podataka i izgradnji

okuženja otpornih na pogreške. Osim što je besplatan i otvorenog koda PostgreSQL je vrlo proširiv. Unutar postgreSQL-a se mogu definirati vlastiti tipovi podataka, izgraditi prilagođene funkcije, čak i pisati kod iz različitih programskih jezika. PostgreSQL se pokušava uskladiti sa SQL standardom gdje takva usklađenost nije u suprotnosti s tradicionalnim značajkama ili može dovesti do loših arhitektonskih odluka [2].

Slika 3. Prikaz alata za pregledavanje PostgreSQL



Izvor: Autor

4.6. GITHUB

GitHub je platforma napravljena za kontrolu verzije koda, dijeljenje i suradnju na kodu s drugima. Ključna prednost GitHub-a je mogućnost suradnje na projektima s više programera. GitHub omogućuje grananje (eng. *Branching*), koji omogućuje da svaki član tima radi na svojoj kopiji projekta. Na taj način promjene se ne sukobljavaju sve dok ne budu spremne za spajanje u glavnu verziju projekta. Uz grananje, GitHUb nudi razne alate poput GitHub Issues i GitHub Projecs pomoću kojih programeri mogu pratiti bugove, dodjeljivati zadatke i organizirati razvojni proces. Git je sustav kontrole verzije datoteka. Posebno je koristan kada se radi u grupi

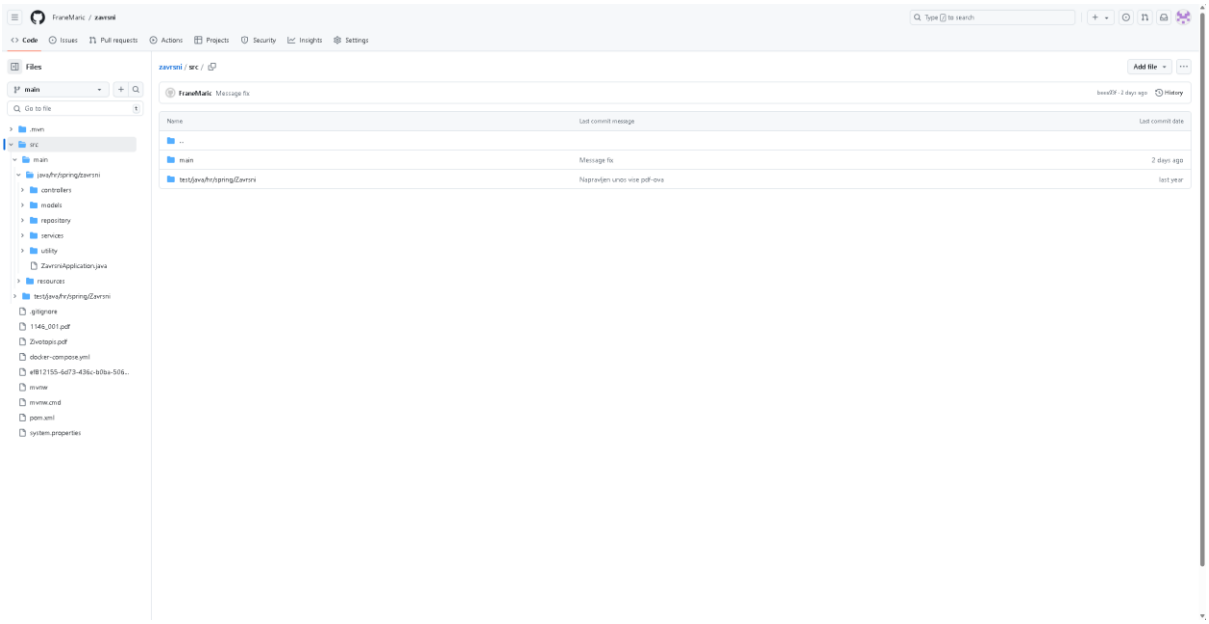
ljudi koja radi izmjene na istim datotekama u isto vrijeme. Spremanjem rada u repozitorij na GitHub-u omogućava korisnicima da:

- pokazuju i dijele svoj rad
- prate i upravljaju promjenama u svom kodu
- dopuste drugim korisnicima da pregledaju kod i daju svoje prijedloge za njegovo poboljšanje
- surađuju na zajedničkom projektu bez brige da će njihove promijene utjecati na rad suradnika [17].

Za ovu aplikaciju GitHub se koristio za kontroliranje verzije koda, dijeljenje repozitorija te lakšu objavu aplikacije na Heroku platformi.

Slika 5 prikazuje aplikaciju na javnom GitHub repozitoriju.

Slika 4. GitHub



Izvor: Autor

5. STRUKTURA APLIKACIJE

5.1. ARHHITEKTURA APLIKACIJE

Za izradu Spring boot aplikacije prvo treba otići na Spring initializr stranicu unutar koje se odabere vrsta projekta, programski jezik, verzija Spring boot-a i ovisnosti te se upišu informacije o aplikaciji koja se razvija (grupa, naziv, opis, ime paketa i verzija Jave). Spring initializr zatim generira prazan projekt spreman za pokretanje. S Microsoft-ovim alatom Visual Studio Code, koji nudi mnoge funkcionalnosti za razvoj, može se otvoriti taj projekt te krenuti s uređivanjem.

Aplikacija je napravljena MVC dizajnom koji odvaja aplikaciju na tri glavne komponente.

Prva komponenta zove se model i prikazuje podatke koji se koriste u bazi. Svaki model odgovara jednoj tablici u bazi podataka te omogućuje čitanje i spremanje te baze podataka.

Pogled (eng. View) je druga komponenta koja predstavlja dio koji korisnik vidi i služi za uređivanje izgleda aplikacije. Za uređivanje pogleda koristi se kombinacija HTML-a, CSS-a i JavaScrip-ta. HTML služi kako bi se dodale stvari koje korisnik vidi (npr. interaktivni gumbi koje korisnik stišće). Kako bi se uredili ti elementi koje korisnik vidi koristimo CSS. CSS-om se svi elementi iz HTML-a mogu urediti kako god programer poželi. JavaScript je skriptni jezik koji se izvršava u web pregledniku. Njime možemo izvršavati jednostavne skripte na pregledniku bez da šaljemo ili tražimo podatke sa servera. Pomoću pogleda možemo prikazivati i prikupljati podatke od korisnika.

Kako bi se obradili podatci koje prikupljamo i prikazujemo korisnicima izrađuju se kontroleri (eng. Controller). Oni su most između aplikacije i servera. Za slanje i dohvaćanje podataka koriste se REST (engl. *REpresentational State Transfer*) metode zahtjeva. REST definira arhitekturu podatkovnog tipa koji se šalje prema servisu. Kod svakog zahtjeva koji se šalje definirani su sljedeći elementi: krajnja točka, metoda zahtjeva, zaglavlje zahtjeva i podatci.

REST metode zahtjeva odnose se na slanje, izmjenjivanje i prikupljanje podataka. Koriste se standardne HTTP metode: GET, POST, PUT, DELETE [7].

5.2. OSNOVNE FUNKCIONALNOSTI APLIKACIJE

Aplikacije započinje s početnom stranicom za prijavu i registraciju korisnika. Registracija služi kako bi se napravili novi korisnički računi. Unutar registracije korisnici trebaju unijeti svoje informacije. Tijekom unosa korisničkih informacija, aplikacija pretražuje uneseno korisničko ime u bazi podataka te ako već postoji to korisničko ime, korisniku se prikazuje upozorenje i blokira gumb za registraciju dok ne unese drugo. Nakon unosa svih korisničkih podataka korisniku se šalje mail s verifikacijskim linkom na adresu koju je unio tijekom registracije. Pritiskom na link, korisnik potvrđuje svoj račun te ostvaruje pravo na prijavu u aplikaciju. Tijekom prijave, ako korisnik pogrešno upiše svoje korisničko ime i lozinku, aplikacija izbacuje prozorčić s porukom „Pogrešno korisničko ime ili lozinka“. Nakon uspješne prijave korisniku se prikazuje njegov korisnički sandučić. Unutar korisničkog sandučića korisnik može pregledavati, pretraživati, preuzimati i brisati svoje poruke. Kako bi se izbjeglo slučajno brisanje datoteka, pritiskom na brisanje, otvara se skočni prozor s pitanjem je li korisnik siguran da želi obrisati tu poruku.

Korisniku dolaze poruke od pošiljatelja. Pošiljatelji su korisnički računi koje može samo administrator napraviti i kojima je dodijeljeno pravo na slanje datoteka. Na navigacijskoj traci pošiljatelja, za razliku od običnog korisnika, nalazi se koverta. Ta ikona odvodi pošiljatelja na prozor za slanje datoteka. Za biranje datoteka koje pošiljatelj želi poslati, otvara se prozor na kojem pošiljatelj može odabrati datoteke koje želi poslati. Nakon odabira i slanja, aplikacija prolazi kroz svaki dokument u potrazi za barkodom. Ako aplikacija nađe barkod u dokumentu onda ga počne čitati i spremati informacije koje su upisane u njemu. Iz svakog barkoda koji je uspješno pročitan, aplikacija sprema ime i prezime osobe kojoj taj dokument treba stići, ime tvrtke ili osobe koja šalje taj dokument i poruku koja je spremljena. Uz te informacije aplikacija sprema trenutni datum kako bi korisnik mogao imati uvid kada mu je poslan taj dokument. Nakon obrađivanja svih dokumenata, pošiljatelju se prikazuje prozor s rezultatom tog slanja. Za svaki dokument piše je li uspješno poslan ili je imao nekih grešaka. Greške mogu biti npr. nije bilo barkoda na tom dokumentu, barkod je nađen ali nisu uspješno pročitane informacije u njemu ili korisnik kojem se šalje nema račun u aplikaciji. Kako bi pošiljatelj mogao lakše pratiti svoja slanja, unutar aplikacije ima uvid u sve poslane datoteke. Tamo može vidjeti sve poruke koje su poslane i provjeriti rezultat njihovog slanja.

Kod 1. Kod za slanje dokumenata

```
@PostMapping("/uploadPDF")
    public ResponseEntity<ErrorModel>
uploadFile(@RequestParam("files") List<MultipartFile> files,
            HttpSession session) {
        ErrorModel errorModel = new ErrorModel();
        for (MultipartFile file : files) {
            try {
                // Save the uploaded file
                byte[] pdfBytes = file.getBytes();

                // Process the PDF file to find and read PDF417
barcode
                String barcodeData = processPdf(pdfBytes);
                Random rnd = new Random();
                String newPath =
"C:\\Faks\\Zavrsni\\zavrsni\\Files\\pdf\\" + rnd.nextInt(1000000)
                        + file.getOriginalFilename();

                File newSavedFile = new File(newPath);
                file.transferTo(newSavedFile);
                s3Service.saveFile(newSavedFile);
                FileModel model = new FileModel();
                Mail mail = new Mail();
                model.setFileName(file.getOriginalFilename());
                model.setFilePath(newPath);
```

```
                model.setType(".pdf");
                model.setSender(session.getAttribute("username").toS
tring());
                String[] recever = barcodeData.split("\n");
                Boolean exsist = testUsername(recever[3], session);
                if (exsist == false) {
                    throw new userNameException(recever[3]);
                }
                model.setRecever(recever[6]);
                fileService.saveFile(model);
                mail.setSender(recever[6]);
                mail.setRecever(recever[3]);
                mail.setVrijeme(LocalDate.now().toString());
                mail.setMessage(recever[13]);
                mail.setFileId(model.getId());
                mail.setFileName(model.getFileName());
                mail.setReceverId(session.getAttribute("currUser").t
oString());
                mailService.saveMail(mail);

            } catch (userNameException ex) {
                errorModel.addUsername(ex.getMessage());
            } catch (Exception e) {
                errorModel.addFile(file.getOriginalFilename());
            }
            session.removeAttribute("currUser");
```

```
        }
        errorModel.removeDuplicateFiles();
        errorModel.removeDuplicateUsernames();
        return
ResponseEntity.status(HttpStatus.OK).body(errorModel);
    }
```

Izvor: Autor

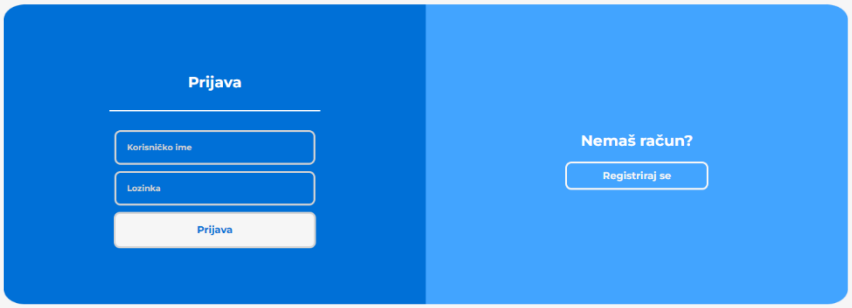
Priloženi kod služi za prikupljanje datoteka koje je pošiljatelj odabrao te procesiranje podataka koje se šalje na server. Pošiljatelj unese bilo koji broj datoteka koje želi poslati aplikacijom te ih ona sakupi i pošalje prema serveru. Server prima listu datoteka i sesiju. Aplikacija najprije uzme datoteku te ju pretvori u niz bitova. Nakon pretvorbe u bitove aplikacija ih pretvara u pdf datoteku. Kako bi se pomoglo aplikaciji u procesiranju te pdf datoteke, dodaju se savjeti za traženje barkoda. U priloženom kodu dodana su 2 savjeta. Prvi je da aplikacija optimizira preciznost traženja nauštrb potrošnje vremena. Ta pomoć služi da bi se povećala vjerojatnost pronalaska barkoda. Druga pomoć koja je u priloženome kodu, ograničava aplikaciju na čitanje samo barkodova koji su tipa pdf417. Time se osiguralo da aplikacija neće pročitati krivu informaciju ako je na dokumentu više tipova barkoda. Kada aplikacija dobije pomoći kreće u čitanje dokumenta. Svaku stranicu koja je na tom dokumentu pretvara u sliku te tu sliku u binarnu bitmapu. Nakon pretvorbe u bitmapu aplikacija pokušava pronaći barkod pomoću pomoći koje su gore spomenute. U slučaju kada barkod nije pronađen aplikacija šalje povratnu poruku da barkod nije nađen, izrađuje novi model za datoteku gdje sprema ime datoteke, tip datoteke i pošiljatelja te se nakon toga sprema taj model u bazu podataka. Uz model za datoteku izrađuje se i model za poštu unutar koje se sprema identifikacijski broj pošiljatelja, identifikacijski broj datoteke, ime datoteke, vrijeme slanja, neuspješnost slanja i poruka koja govori da barkod nije nađen u datoteci. U suprotnome kada je barkod pronađen, taj barkod se sprema u tekst. Kako bi se osigurala ispravnost čitanja dijakritičkih znakova taj tekst se pretvara u bitove te iz bitova ponovno u tekst tipa UTF-8. Nakon čitanja i spremanja teksta koji je u barkodu, aplikacija šalje i sprema poslanu datoteku na Amazon S3. Nakon toga radi se model za datoteku unutar kojeg se sprema ime datoteke (ime koje je datoteka imala na računalu pošiljatelja), tip datoteke (tip datoteke je uvijek pdf s obzirom da aplikacija radi konverziju) te pošiljatelj (sprema se ime pošiljatelja za lakše praćenje datoteka s istim imenom). Nakon toga tekst koji je bio zapisan u barkodu, aplikacija dijeli po redovima, u niz kako bi se lakše spremili

ti podatci u bazu podataka. Prije daljnjeg spremanja podataka u model, aplikacija provjerava postoji li korisnik u bazi podataka. U slučaju da ne postoji, aplikacija sprema sve gore navedene podatke uz ime primatelja te izrađuje novi model za poštu unutar koje se sprema primatelj, pošiljatelj, poruka koja je bila napisana u barkodu, identifikacijski broj pošiljatelja, identifikacijski broj datoteke (identifikacijski broj datoteke baza podataka pridjeljuje automatski prilikom spremanja modela za datoteku), ime datoteke, datum slanja poruke, neuspješnost slanja i poruka koja objašnjava da nema korisnika u bazi podataka. U suprotnome, ako korisnik postoji, spremaju se isti podatci u modele s razlikom što se u model za poštu sprema da je slanje izvršeno uspješno te poruka potvrde slanja korisniku i ime korisnika. Nakon prolaska kroz sve poslane datoteke i testiranja postoji li barkod na njima, aplikacija šalje listu s rezultatima nazad na pogled gdje će se ti rezultati smjestiti u tablicu i prikazati korisniku.
Treća vrsta korisnika je administrator. Trenutno je samo jedan administrator i on ima mogućnost pregleda i uređivanja svih korisnika koji su prijavljeni u aplikaciju. Na navigacijskoj traci ima dodatan izbor za prikaz svih korisnika ili pošiljatelja. Uz svakog korisnika i pošiljatelja, administrator ima izbor za brisanje ili uređivanje. Ako odabere brisanje, vidjet će skočni prozor s e-mailom korisnika koji ga pita želi li stvarno obrisati tog korisnika. Ako odabere na uređivanje, otvara se prozor unutar kojeg su sve informacije za tog korisnika i koje administrator može promijeniti.

5.3. NAČUN UPOTREBE APLIKACIJE

Prva stvar koju korisnik vidi kada uđe u aplikaciju je prozor za prijavu. Tu se korisniku daje na izbor hoće li se prijaviti u aplikaciju ili će otići na prozor za registraciju.

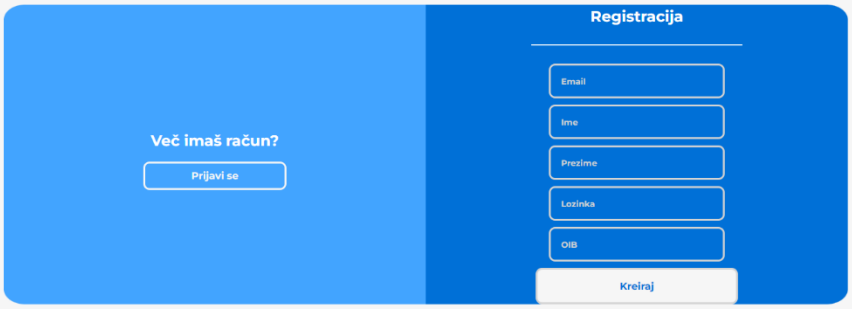
Slika 5. Prozor za prijavu



Izvor: Autor

Ako odluči otići na registraciju, otvara se prozor gdje korisnik unosi svoje podatke važne za registraciju u aplikaciju. Unutar tog prozora, korisnik treba upisati e-mail, ime, prezime, lozinku i OIB te stisnut pošalji.

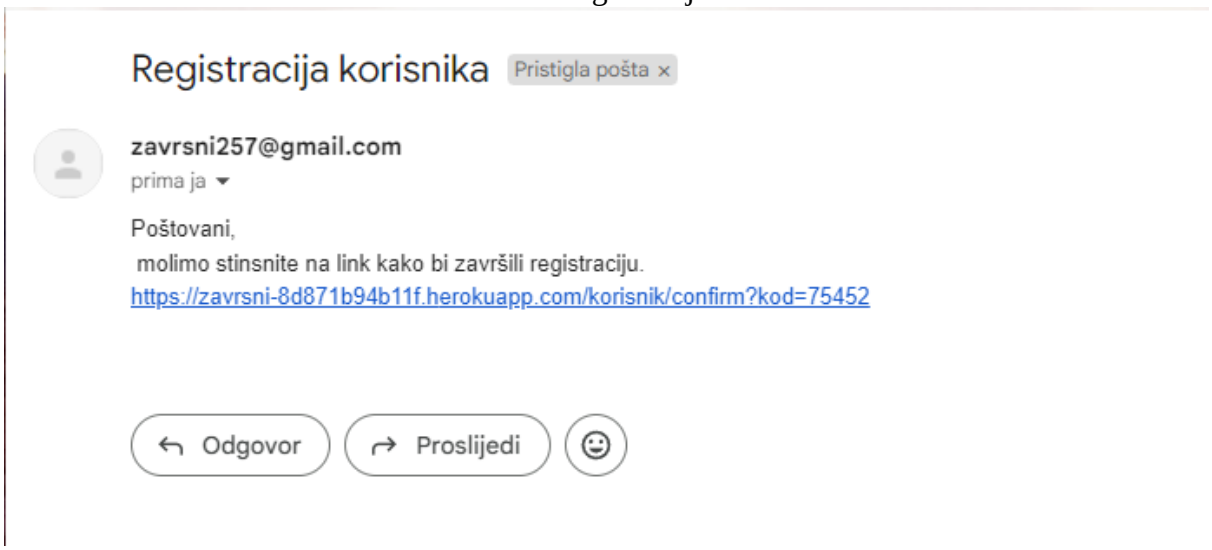
Slika 6. Prozor za registraciju korisnika



Izvor: Autor

Nakon uspješne registracije korisniku se šalje e-pošta s poveznicom za potvrdu računa na adresu koju je unio prilikom registracije.

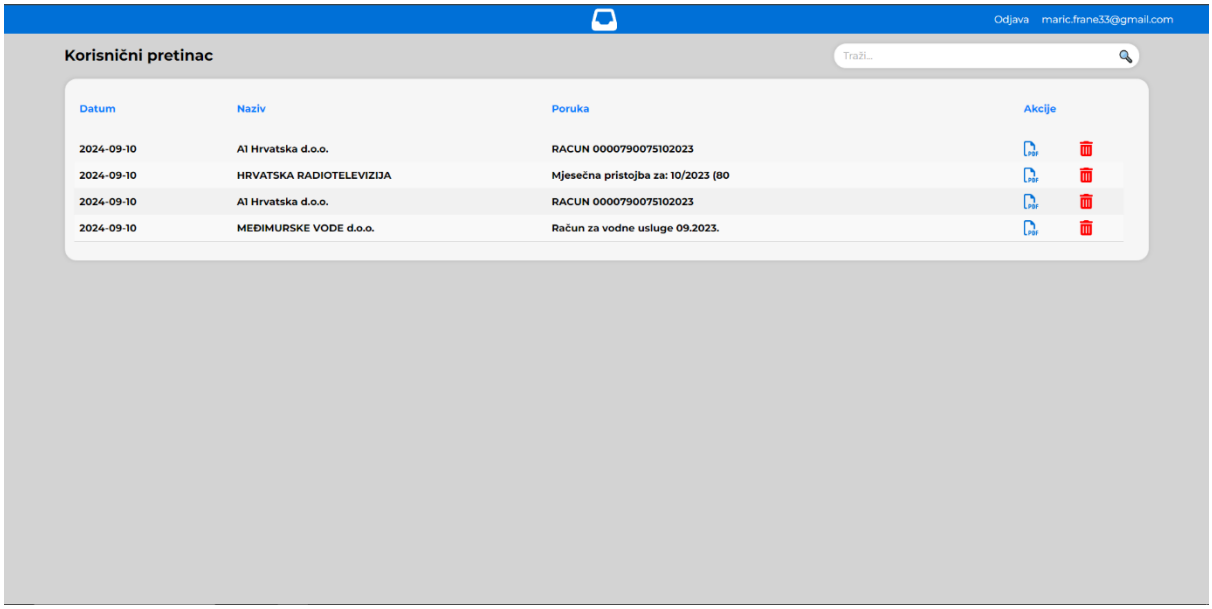
Slika 7. Mail za registraciju korisnika



Izvor: Autor

Kada korisnik stisne na tu poveznicu, preusmjeri ga se ponovno na prozor za prijavu gdje može unijeti svoju e-poštu i šifru te se tako prijaviti u aplikaciju. Nakon prijave korisniku se otvara njegov korisnički pretinac. Unutar tog pretinca, korisnik, ima uvid u sve svoje poruke i dokumente. Za svaki dokument koji je poslan korisniku, prikazuje se datum slanja tog dokumenta, pošiljatelj i poruka. Uz informacije o dokumentu, korisnik ima mogućnost preuzimanja i brisanja svih datoteka i poruka. Za lakšu navigaciju i traženje dokumenata, unutar prozora za korisnički pretinac postoji i traka za traženje. S njom korisnik može tražili bilo koju poruku ili pošiljatelja. Na vrhu korisničkog pretinca nalazi se navigacijska traka unutar koje su ikona za korisnički pretinac, gumb za odjavu i korisničko ime prijavljenog korisnika.

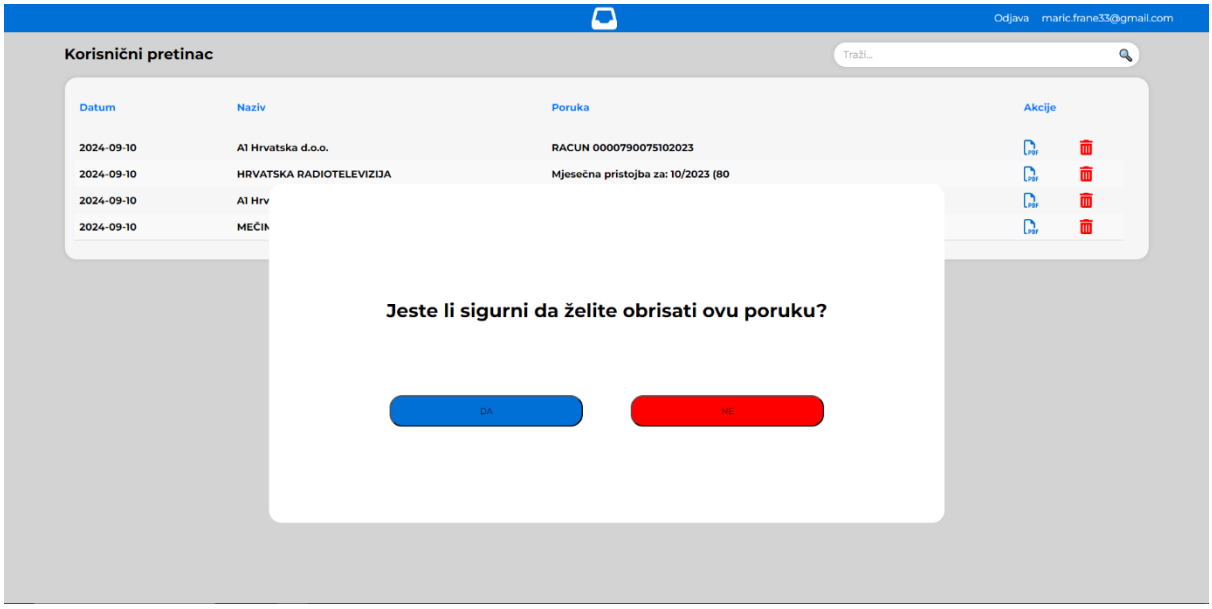
Slika 8. Korisnički pretinac



Izvor: Autor

Kako bi se osiguralo nenamjerno brisanje poruka prilikom korištenja aplikacije, za svaku poruku koju korisnik pokuša obrisati, na ekran dolazi mali prozor s pitanjem želi li korisnik stvarno obrisati tu poruku.

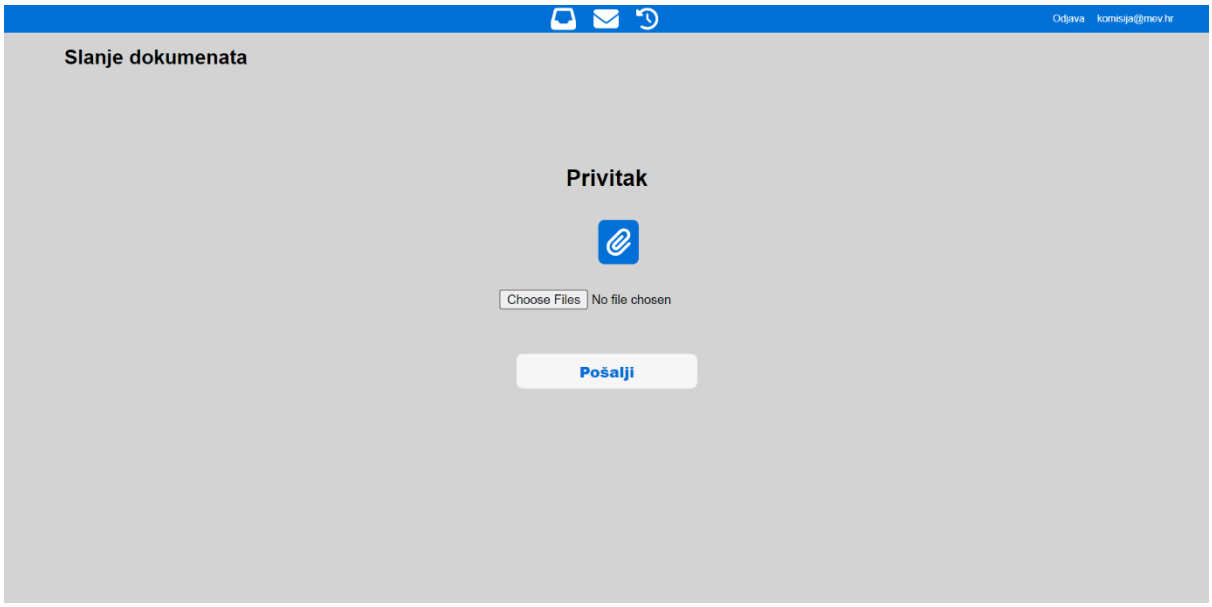
Slika 9. Skočni prozor za brisanje poruke



Izvor: Autor

Prijavom u aplikaciju s pošiljateljevim računom, na navigacijskoj traci se dodatno prikazuju ikona kuverte i povijesti slanja. Pritiskom na kuvertu pošiljatelju se otvara prozor za slanje dokumenata. Unutar tog prozora pošiljatelj ima 2 gumba. Prvi je za otvaranja izbornika za datoteke unutar kojeg pošiljatelj odabire jednu ili više datoteka koje želi poslati.

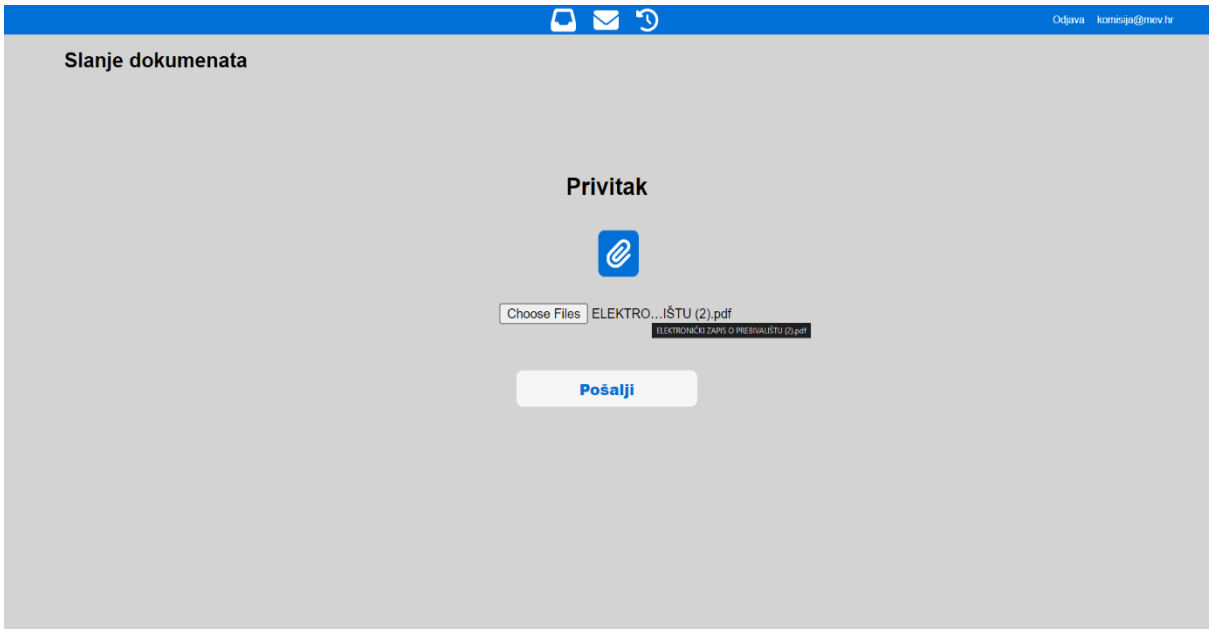
Slika 10. Prozor za slanje datoteka



Izvor: Autor

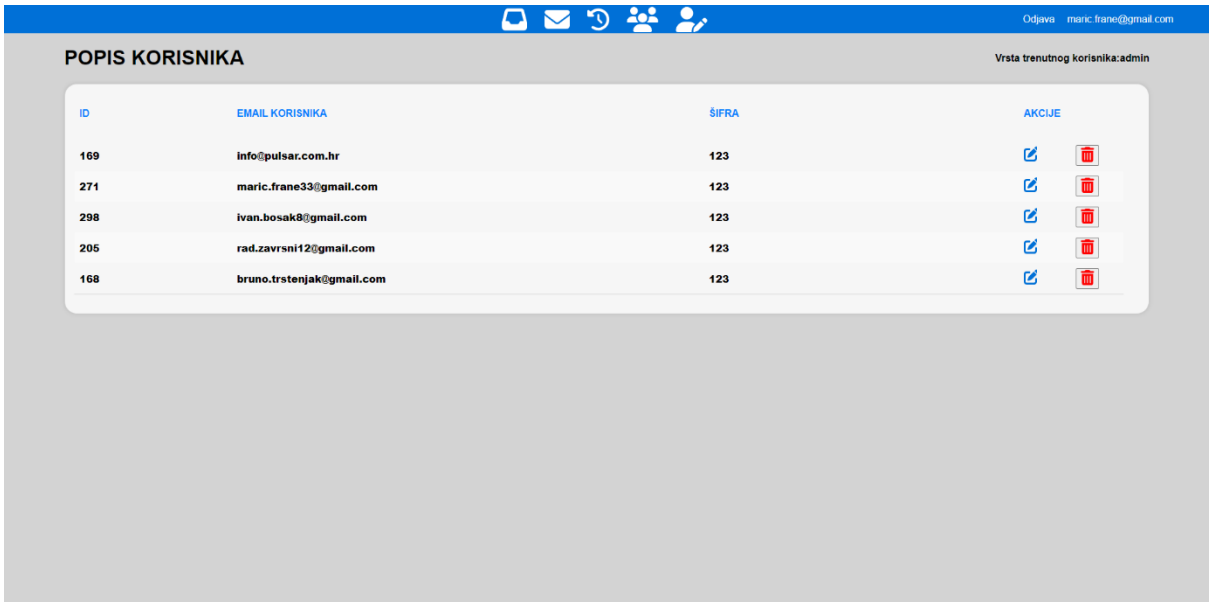
Kada pošiljatelj odabere datoteke koje želi poslati pokazuju mu se imena tih datoteka kako bi mogao, prije slanja, provjeriti koje je sve datoteke odabrao. Nakon odabira pošiljatelj mora stisnuti na drugi gumb, pošalji, kako bi se datoteke koje je odabrao poslale korisnicima.

Slika 11. Odabrani dokumenti



Izvor: Autor

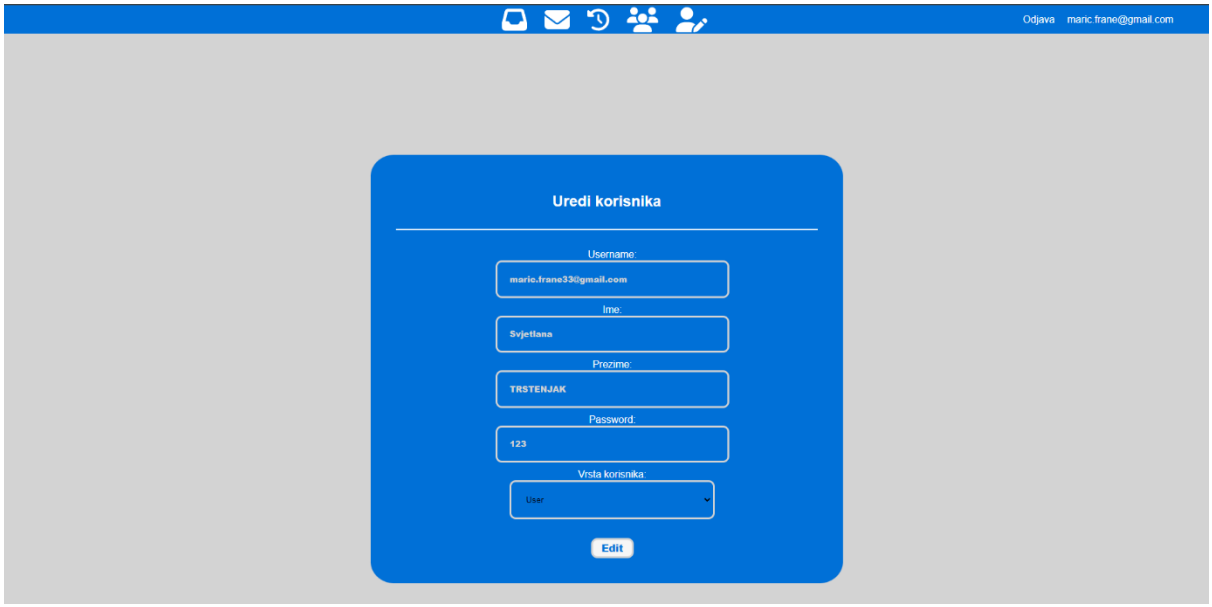
Slika 12. Prozor popisa korisnika



Izvor: Autor

Prijavom u aplikaciju s administratorom, na navigacijskoj traci se pokazuju sve ikone kao i kod pošiljatelja uz ikonu za popis svih korisnika i popis svih pošiljatelja. Pritiskom na ikonu za popis svih korisnika, administratoru se otvara prozor s tablicom unutar koje su prikazani id, e-mail, lozinka, ikona za uređivanje i ikona za brisanje za svakog korisnika.

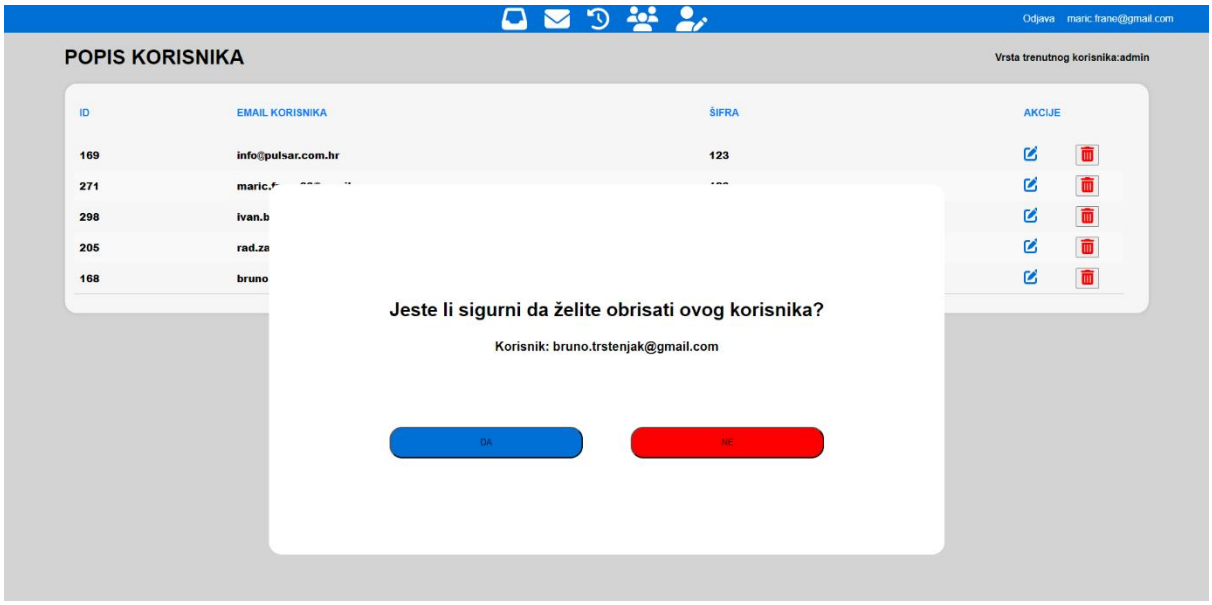
Slika 13. Prozor uređivanja korisnika



Izvor: Autor

Pritiskom na ikonu za uređivanje korisnika, otvara se prozor s formom. Unutar forme administrator ima mogućnost izmjene korisnikovo korisničko ime, ime, prezime, lozinku i vrstu korisnika. Svako polje koje korisnik može promijeniti već je ispunjeno s informacijama koje su trenutno postavljene za tog korisnika. U slučaju da korisnik stisne na brisanje korisnika, dolazi skočni prozor koji provjerava je li administrator zaista siguran da želi obrisati tog korisnika te ispisuje e-mail korisnika.

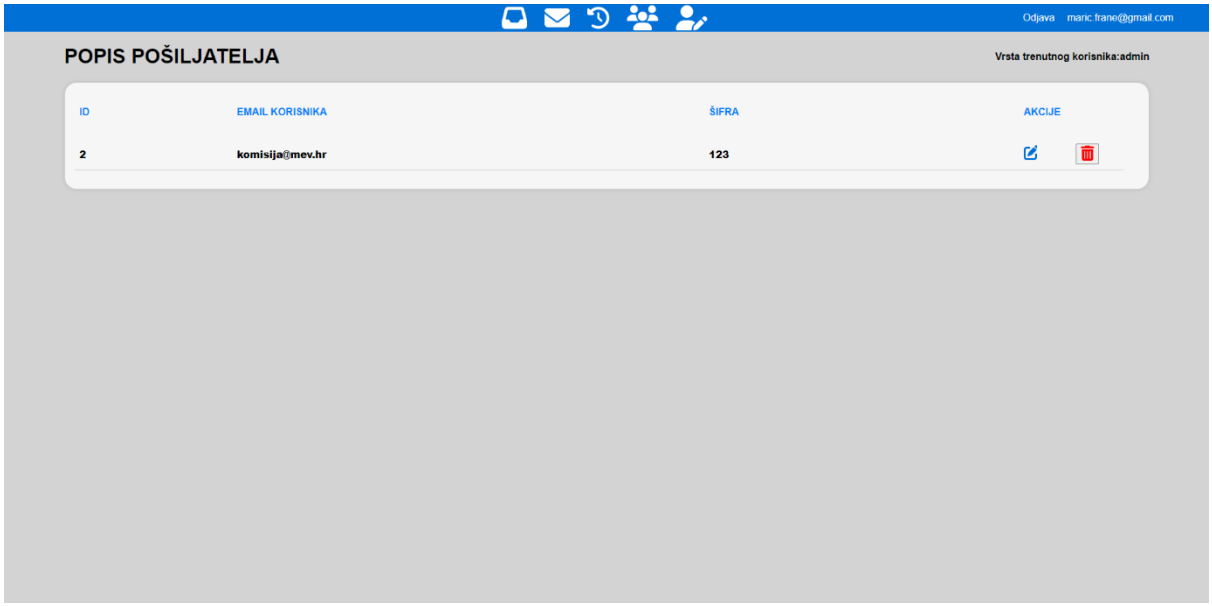
Slika 14. Skočni prozor za brisanje korisnika



Izvor: Autor

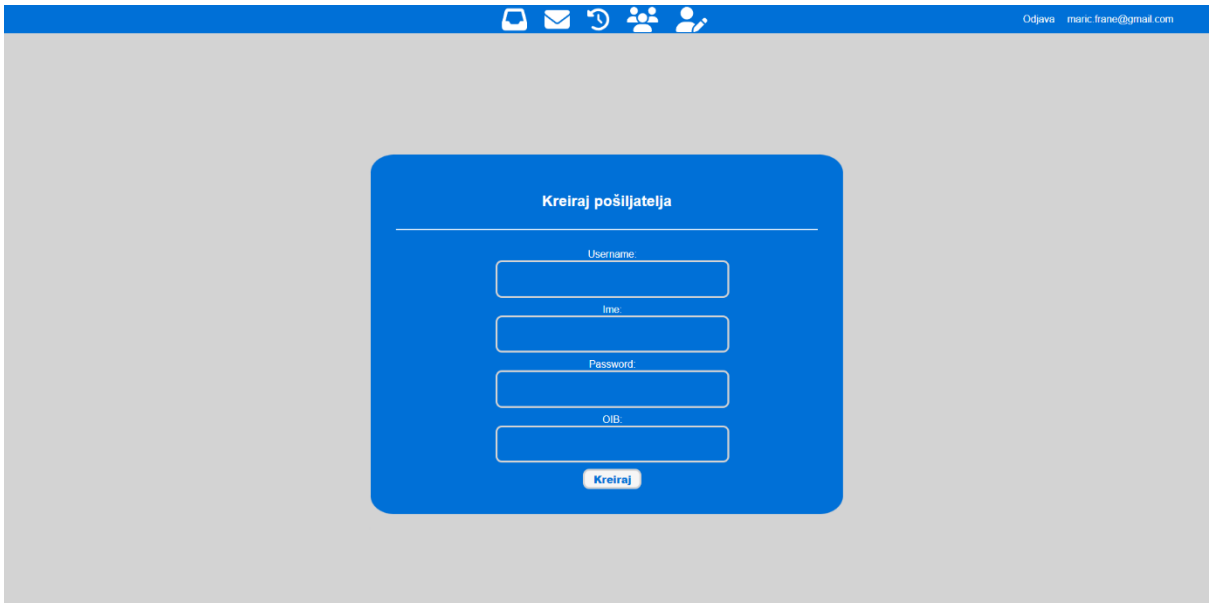
Zadnja ikona na navigacijskoj traci je za popis svih pošiljatelja. Pritiskom na nju preusmjerava se korisnika na prozor s tablicom svih pošiljatelja. Unutar tablice za popis pošiljatelja, prikazane su korisničke informacije o pošiljateljima i pošto pošiljatelje može raditi samo administrator, gumb za izradu novog pošiljatelja. Pritiskom na taj gumb otvara se prozor s formom za izradu novog pošiljatelja unutar kojeg administrator treba unijeti korisničko ime pošiljatelja,ime, lozinku i oib.

Slika 15. Prozor popisa pošiljatelja



Izvor: Autor

Slika 16. Prozor kreiranja korisnika



Izvor: Autor

6. BAZA PODATAKA

Baza podataka za ovu aplikaciju sastoji se od 4 tablice: file_model, korisnik, mail i potvrdio_model. Svaka od ovih tablica igra važnu ulogu u organizaciji i upravljanju podataka kojima se aplikacija služi.

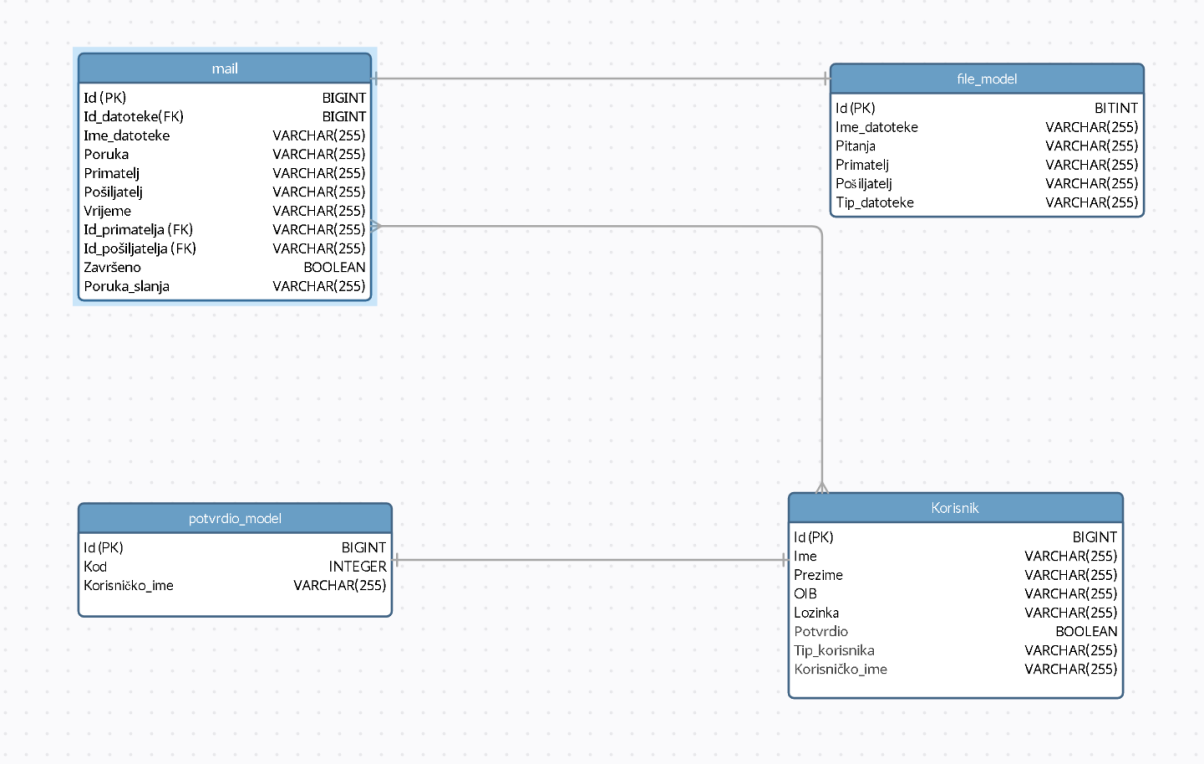
Prva tablica, file_model, koristi se za spremanje informacija o datotekama koje se šalju i primaju. Unutar nje spremaju se podatci o datotekama kao što su ime datoteke, primatelj, pošiljatelj i tip datoteke. Svaki novi zapis u tu tablicu dobiva svoj identifikacijski broj koji je unikatan za svaku datoteku te tako razlikuje datoteke koje imaju isto ime. Uz pomoć te tablice i uz pomoć informacija unutar nje, aplikacija zna dohvatiti datoteku koju korisnik želi preuzeti.

Druga tablica je korisnik. Unutar te tablice se spremaju informacije o korisnicima unutar aplikacije. Svi korisnici, potvrđeni i nepotvrđeni, su spremljeni kao i sve informacije koje su unijeli pri registraciji. Tablica korisnik sprema ime korisnika, oib, lozinku, informaciju je li korisnik potvrdio svoj korisnički račun, prezime, tip korisnika i korisničko ime. Pomoću tipa korisnika aplikacija dobiva informaciju koje ovlasti treba dati korisniku pri prijavi. Kao i kod file_model tablice svakom novom unosu je dodijeljen identifikacijski broj. Korištenjem identifikacijskog broja umjesto imena i prezimena prilikom uspoređivanja podataka, aplikacija može brže prolazili kroz korisnike.

Sljedeća tablica je mail tablica koja pohranjuje informacije o porukama koje su poslane unutar aplikacije. Pomoću nje se razvrstava pošta koja je poslana unutar aplikacije te osigurava točnost korisničkog sandučića svakog korisnika. Svaka poruka pohranjuje identifikacijski broj datoteke koja je poslana, nazivu te datoteke, poruci koja je upisana unutar barkoda, ime i prezime primatelja, ime i prezime ili ime tvrtke pošiljatelja, datum slanja poruke, identifikacijski broj primatelja te identifikacijski broj pošiljatelja. Također, pomoću mail tablice može se pratiti povijest komunikacije što omogućava administratorima bolje razumijevanje funkcionalnosti aplikacije vezane uz poruke.

Posljednja tablica je potvrdio_model. Unutar te tablice se spremaju informacije o registracijama. Kada se korisnik registrira unutar te tablice se spremi identifikacijski kod registracije i korisničko ime korisnika koji se registrirao. Informacije unutar potvrdio_model tablice čuvaju se dok korisnik ne potvrdi svoju registraciju te se nakon toga brišu kako bi se smanjila potrošnja pohrane.

Primaran ključ svih tablica je id (identifikacijski broj) koji se automatski dodjeljuje od strane PostgreSQL-a za svaku novo dodanu informaciju. Pomoću tih primarnih ključeva mogu se povezati tablice. Za svaku uspješnu i neuspješnu poruku koja se pošalje, spremaju se sve informacije koje su potrebne kako bi se ta poruka mogla prikazati korisniku i kasnije dohvatiti. Ova struktura podataka omogućava aplikaciji da učinkovito upravlja velikim količinama podataka i osigura visoku razinu organizacije istih.

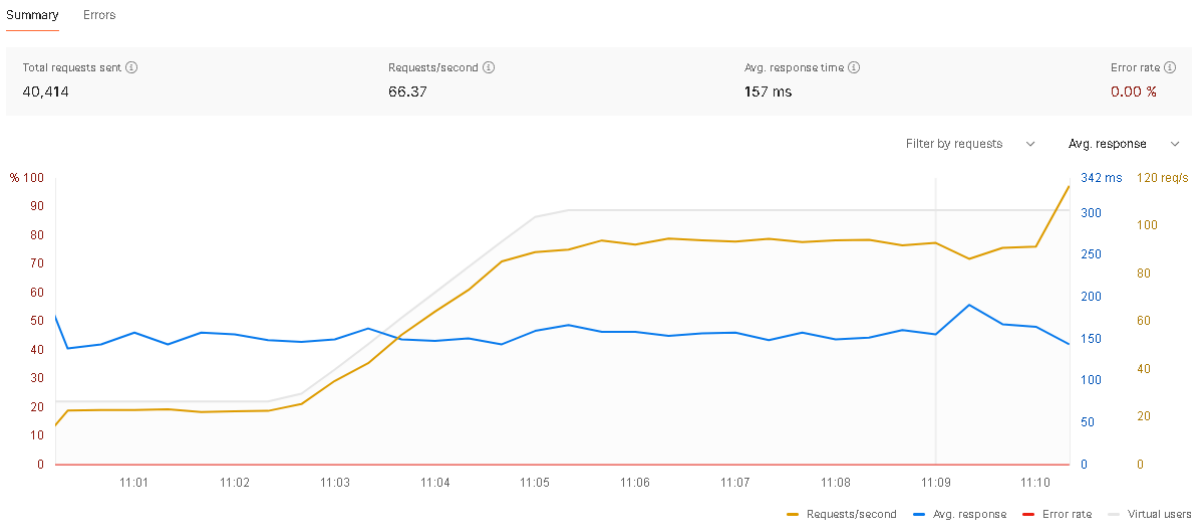


7. TESTIRANJE RADA APLIKACIJE

Kako bi se osigurala stabilnost, pouzdanost i funkcionalnost aplikacije, aplikaciju je testiralo više korisnika. Testirali su se razni dijelovi aplikacije te pokušali naći potencijalni problemi koji bi se mogli dogoditi u produkciji. Testni spektar sastojao se od mnogih dokumenata koji su imali različitu strukturu, položaj barkoda i tipove datoteka.

Svaki tester ručno je prošao kroz glavne funkcije aplikacije koje uključuju registraciju korisnika, prijavu, slanje računa, preuzimanje i brisanje istih, izradu pošiljatelja i uređivanje korisnika i pošiljatelja. Testeri su oponašali razne situacije koje bi se mogle dogoditi prilikom korištenja aplikacije u produkciji.

Testiranje aplikacije, također je uključivalo istovremeno povezivanje i korištenje aplikacije od više korisnika. Ovi testovi su rađeni kako bi se osiguralo da aplikacija i baza podataka mogu nesmetano raditi kada više korisnika istovremeno pošalje zahtjeve prema serveru.



Izvor: Autor

Slika prikazuje rezultate iz Postmanovog alata Runner unutar kojeg se simuliralo spajanje i korištenje aplikacije od 40 korisnika. Rezultati jasno pokazuju kako se povećanjem broja korisnika i zahtjeva prema serveru, aplikacija skalirala te održala slično vrijeme odgovora. Stres testiranje aplikacije je prošlo uspješno te su svi zahtjevi prošli bez greške.

Testiranje performansi kritičan je dio za aplikacije koje moraju izdržati visok promet zahtjeva korisnika te obraditi velik broj podataka. Svaki odrađeni test morao je biti procesiran u određenom vremenu kako bi se osiguralo brzo odvijanje aplikacije u produkciji.

Testiranje započinje testiranjem forme za registraciju korisnika. Kod registracije korisnika aplikacija treba zahtijevati da se ispune sva polja. Uz ispunu svih polja, za uspješnu registraciju korisnik treba unijeti e-mail adresu u polje korisničkog imena. Aplikacija provjerava ima li e-mail adresa prisutan znak @. Kako bi se izbjeglo pojavljivanje duplih unosa u bazu podataka prilikom upisa korisničkog imena, aplikacija provjerava da li već postoji uneseni e-mail.

Nakon registracije testira se prijava korisnika u aplikaciju. Aplikacija ne smije dopustiti korisniku da se prijavi prije nego li potvrdi svoju registraciju. Prilikom prijave lozinka mora biti skrivena. Ako korisnik unese krivo korisničko ime ili lozinku aplikacija treba napomenuti da su upisane krive informacije, ali ne i podatak koja je informacija kriva.

Sljedeće se testirao korisnički pretinac. Unutar korisničkog pretinca trebaju se prikazati sve poruke koje su poslane prijavljenome korisniku. Usporedbom informacija iz baze podataka provjeravalo se jesu li sve poruke prikazane unutar korisničkog pretinca te jesu li te poruke ispravno ispisane. Uz ispitivanje broja i prikaza poruka testiralo se i preuzimanje i brisanje poruka. Svaka datoteka koja je poslana korisniku mora imati mogućnost za preuzimanje i brisanje.

Nakon korisničkog pretinca, testeri su se prijavili kao pošiljatelj unutar aplikacije te slali razne datoteke. Kako bi se provjerila informacija koju aplikacija pročita iz barkoda (ako je barkod pronađen), dobiveni tekst se uspoređivao s vanjskom aplikacijom za čitanje barkodova. Svaka poruka koja je uspješno poslana morala je imati primatelja, pošiljatelja, poruku i vrijeme slanja.

Poslije testiranja svih funkcija pošiljatelja, testeri su se prijavili kao administrator. Provjeravalo se jesu li liste korisnika i pošiljatelja potpune i točne. Uz važnost da su svi korisnici na broju, provjeravale su se informacije o pojedinom korisniku koje administrator ima mogućnost pregledati i promijeniti.

Uz testiranje funkcionalnosti svake vrste korisnika, testeri su provjeravali navigacijsku traku. Svaki korisnik mora imati prilagođenu navigacijsku traku za svoje ovlasti.

8. SIGURNOST APLIKACIJE

Jedan od najvažnijih elemenata svake aplikacije je sigurnost, posebno kada se radi o pohrani osjetljivih podataka poput računa i dokumenata koji sadrže privatne ili poslovne informacije. Uvođenje sigurnosnih mjera ključno je za sigurnost korisnika i njihovih podataka. U ovoj aplikaciji, sigurnost ima dvostruku ulogu: zaštita korisničkih podataka i točnost dostavljenih dokumenata.

Kako bi se zaštitili korisnički podatci, za ulazak u aplikaciju je ugrađen sustav prijave koji zahtijeva od korisnika da unese svoje korisničko ime i lozinku. Ovaj sustav omogućava da samo ovlašteni korisnici mogu pristupiti svojim datotekama. Prilikom krivog upisa korisničkog imena i lozinke, korisniku neće pisati što je krivo unio kako bi se spriječila neovlaštena prijava korisnika pogađanjem njegovih informacija.

Zapisivanje datoteka na server zaštićeno je s Amazon S3 šifriranjem. Amazon S3 šifrira sve prijenose objekata u svim spremnicima. Također, kako bi se očuvala sigurnost dokumenata, Amazon S3 blokira javni pristup dokumentima te jedino aplikacija i administrator imaju pristup njima.

Kako bi se osigurala mogućnost spremanja datoteka svih veličina, datoteke koje se spremaju na server, spremaju se u više dijelova (eng. *Multipart*). Ako spremanje jednog dijela datoteke ne uspije, može se ponovno poslati taj dio bez utjecaja na druge dijelove. Nakon što se svi dijelovi učitaju Amazon S3 sastavlja te dijelove te radi objekt.

Ukoliko se na server pošalje istu datoteku više puta, spremiti će se više kopija iste. Kako bi aplikacija razlikovala te dokumente te mogla pratiti informacije o svakoj kopiji, unutar baze podataka je svako slanje spremljeno pod različitim identifikacijskim brojem.

Automatiziranjem slanja dokumenata uveliko se smanjuje mogućnost greške prilikom slanja dokumenata. Aplikacija detaljno uspoređuje primatelja unutar barkoda s korisnicima upisanima u bazu podataka te osigurava da dokument dođe pravoj osobi. Također, izbacivanjem ljudskog faktora smanjuje se rizik neovlaštenog pristupa osobnim podatcima pojedinaca ili tvrtke.

Kako bi se spriječilo otvaranje lažnih profila u tuđe ime, prilikom registracije je napravljena potvrda e-maila. Korisnik treba imati pristup e-mailu kako bi mogao pristupiti verifikacijskom kodu te ujedno i otključati svoj korisnički račun. Tom procedurom osigurava se da je korisnik

vlasnik tog e-maila čime se sprječava stvaranje računa bez autorizacije. Kako se ne bi mogla lažirati potvrda korisnika, na kraju verifikacijske poveznice je broj koji aplikacija sama i automatski dodjeljuje nasumično. Svaki korisnik dobiva jedinstveni nasumični verifikacijski kod.

Baza podataka, unutar koje se spremaju sve informacije o korisnicima i datotekama, zaštićena je posebnim sigurnosnim mjerama kako bi se spriječilo neovlašteno spajanje na istu. Spajanje na bazu dozvoljeno je samo jednom korisničkom računu koji je zaštićen korisničkim imenom i lozinkom. Kako bi se osiguralo neželjeno spajanje u bazu podataka, ograničio se broj veza koje ona može imati istovremeno. Informacije za pristup bazi podataka, dostupne su samo administratoru.

Svaki poziv funkcija unutar aplikacije bilježi se unutar Heroku platforme koja nudi detaljne zapisnike za praćenje svih aktivnosti rada aplikacije. Svaka korisnička prijava, kao i korištenje različitih funkcija aplikacije, evidentirana je u realnom vremenu što omogućava detaljnije praćenje svih događaja. Prilikom prijave korisnika ili pošiljatelja, evidentira se korisničko ime, identifikacijski broj korisnika i korisnikova IP adresa. Kombinacijom svih tih informacija, administrator može lako odrediti koji je korisnik kršio uvijete korištenja aplikacije. Zapisnici unutar Heroku platforme omogućavaju analizu rada aplikacije te administratoru daju mogućnost uvida kod nepredviđenog ponašanja aplikacije.

9. ZAKLJUČAK

Cilj ovo završnog rada je ostvaren. Korisnicima ove web aplikacije olakšati će se pristup važnim dokumentima, a pošiljateljima će se olakšati slanje tih dokumenata.

Tijekom razvoja ove aplikacije pojavilo se nekoliko problema. Jedan od problema je bio to što aplikacija nije prepoznala barkod u datoteci iako je on bio prisutan. Mijenjanjem rezolucije slike koju algoritam pretražuje osigurala se veća preciznost te učinila aplikaciju pouzdanijom. Drugi problem koji se pojavio je bio nemogućnost čitanja dijakritičkih znakova unutar barkoda. Problem se riješio tako što se pročitani tekst dekodiralo u bitove te nazad u tekst koji podržava dijakritičke znakove.

U daljnjem razvoju aplikacije moglo bi se ugraditi verifikacija korisnika putem OIB-a te tako dodatno povećati sigurnost i točnost korisničkih podataka. Također, enkripcijom korisničkih podataka te izradom uslova kod izrade lozinke bi se dodatno osigurala sigurnost čak i u sličaju sigurnosnih prijetnji. Procesiranje dokumenata u različitim nitima bi povećalo brzinu te tako poboljšalo iskustvo korisnika.

IZJAVA O AUTORSTVU

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU

Bana Josipa Jelačića 22/a, Čakovec

IZJAVA O AUTORSTVU

Završni/diplomski rad isključivo je autorsko djelo studenta te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, internetskih i drugih izvora) bez pravilnog citiranja. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom i nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, Frane Marić (ime i prezime studenta) pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog rada pod naslovom Cloud aplikacija E-sandučić

te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:

Frane Marić

(vlastoručni potpis)

10.POPIS LITERATURE

[1] Spring boot
Dostupno na: https://spring.io/projects/spring-boot (09.09.2024)
[2] PostgreSQL
Dostupno na: https://www.postgresql.org/ (11.09.2024)
[3] pgAdmin4
Dostupno na: https://www.pgadmin.org/faq/ (09.09.2024)
[4] Heroku
Dostupno na: https://www.heroku.com/about (12.09.2024)
[5] Amazon s3
Dostupno na: https://aws.amazon.com/pm/serv-s3/ (12.09.2024)
[6] Java
Dostupno na: https://www.java.com/en/download/help/whatis_java.html (12.09.2024)
[7] Trstenjak B. (2023.): Aplikacije u oblaku i mikroservisi
[8] Visual Studio Code
Dostupno na: https://code.visualstudio.com/docs (11.09.2024)
[9] SQL
Dostupno na: https://aws.amazon.com/what-is/sql/ (11.09.2024)
[10] pgAdmin4
Dostupno na: https://www.pgadmin.org/faq/#1 (12.09.2024)
[11] Amazon S3
Dostupno na: https://aws.amazon.com/s3/ (10.09.2024)
[12] HTML
Dostupno na: https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/HTML_basics (12.09.2024)
[13] JavaScript
Dostupno na: https://developer.mozilla.org/en-US/docs/Web/JavaScript (12.09.2024)
[14] Ajax
Dostupno na: https://www.freecodecamp.org/news/ajax-tutorial/ (14.09.2024.)
[15] pgAdmin4
Dostupno na: https://www.pgadmin.org/docs/pgadmin4/latest/index.html (14.09.2024)
[16] pgAdmin4
Dostupno na: https://www.enterprisedb.com/blog/understanding-pgadmin-4-architecture (14.09.2024.)
[17] GitHub
Dostupno na: https://docs.github.com/en/get-started/start-your-journey/about-github-and-git (15.09.2024.)

[18] Knok Ž. (2021): Prezentacija s predavanja – Uvod u baze podataka

[19] David J. Eck (2006.): Introduction to Programming Using Java, Hobar and William Smith Collegues

11.POPIS ILUSTRACIJA

12.POPIS KODOVA