

**SVEUČILIŠTE U SPLITU
FAKULTET ELEKTROTEHNIKE, STROJARSTVA I
BRODOGRADNJE**

ZAVRŠNI RAD

WEB APLIKACIJA PREUZMIME

Ivan Lukšić

Split, srpanj 2021.



Preddiplomski sveučilišni studij: **Računarstvo**

Smjer/Usmjerenje: /

Oznaka programa: 120

Akadska godina: 2020./2021.

Ime i prezime: **Ivan Lukšić**

Broj indeksa: 189-2018

ZADATAK ZAVRŠNOG RADA

Naslov: **Web aplikacija preuzmiMe**

Zadatak: Potrebno je izraditi web aplikaciju koja bi funkcionirala po principu oblaka. Omogućiti korisniku da postavi datoteku na web stranicu. Aplikacija treba imati svojstvo izrade dinamičkog linka na postavljenu datoteku i zaštitu pomoću lozinke. Koristiti tehnologije React, Node.js, Express.js te PostgreSQL baze podataka.

Datum obrane: 22.07.2021.

Mentor:

izv. prof. dr. sc. Damir Sedlar

IZJAVA

Ovom izjavom potvrđujem da sam završni rad s naslovom WEB APLIKACIJA PREUZMIME pod mentorstvom izv. prof. dr. sc. Damira Sedlara pisao samostalno, primijenivši znanja i vještine stečene tijekom studiranja na Fakultetu elektrotehnike, strojarstva i brodogradnje, kao i metodologiju znanstveno-istraživačkog rada, te uz korištenje literature koja je navedena u radu. Spoznaje, stavove, zaključke, teorije i zakonitosti drugih autora koje sam izravno ili parafrazirajući naveo/la u završnom radu citirao/la sam i povezao/la s korištenim bibliografskim jedinicama.

Student/ica



Ivan Lukšić

Sadržaj

1. UVOD	1
2. FUNKCIONALNOSTI I DIZAJN SUSTAVA	2
2.1. Funkcionalnosti i zahtjevi	2
2.1.1. Zahtjevi prilikom postavljanja datoteke	2
2.1.2. Zahtjevi prilikom preuzimanja datoteke	3
2.2. Dizajn sustava	3
2.2.1. Modeli sustava	3
2.2.2. Model podataka	5
3. TEHNOLOGIJE	6
3.1. React	6
3.1.1. Material-ui	7
3.2. Node.js	8
3.2.1. Bcrypt	9
3.2.2. Multer	9
3.2.3. Sequelize	10
3.3. Express.js	10
3.4. PostgreSQL	11
4. IMPLEMENTACIJA APLIKACIJE	12
4.1. Korisničko sučelje	12
4.2. Poslužiteljski dio aplikacije	20
ZAKLJUČAK	26
LITERATURA	27
POPIS OZNAKA I KRATICA	28
SAŽETAK	29
SUMMARY	30

1. UVOD

PreuzmiMe je web aplikacija rađena s ciljem da na jednostavan način omogući postavljanje datoteka te preuzimanje istih putem poveznica koju aplikacija generira prilikom postavljanja. Korisnicima je omogućena zaštita datoteka putem zaporke koja se mora upisati prije nego je omogućeno preuzimanje datoteke. Kontrola pristupa je također omogućena postavljanjem vremenskog ograničenja za preuzimanje te brojem preuzimanja datoteke s dobivenom poveznicom. U drugom poglavlju će se navesti funkcionalnosti aplikacije i dizajn samog sustava. Kroz treće poglavlje će se obrađivati tehnologije koje su se koristile za izradu aplikacije zajedno s izrescima koda koji pokazuju kako su se predmetne tehnologije koristile. Četvrto poglavlje će se fokusirati na implementaciju aplikacije i hodogram izrade same iste.

2. FUNKCIONALNOSTI I DIZAJN SUSTAVA

2.1. Funkcionalnosti i zahtjevi

Prilikom izrade aplikacije sljedeći zahtjevi trebaju biti ispunjeni i funkcionalnosti implementirane:

2.1.1. Zahtjevi prilikom postavljanja datoteke

- Korisnik može na intuitivan način povući datoteku u za to predviđen segment ili klikom na dugme otvoriti izbor datoteke za postavljanje. Nakon odabira željene datoteke korisnik dobiva povratnu informaciju u vidu prikaza imena datoteke u segmentu u koji se datoteka prenijela.
- Korisnik prenijetoj datoteci može postaviti zaštitu u vidu zaporku koju unosi u za to predviđeni okvir za unos teksta. Prilikom postavljanja datoteke na server ta zaporka se enkriptira tako da se smanji mogućnost curenja podataka u slučaju neovlaštenog napada na bazu podataka.
- Kontrola pristupa se također vrši tako da korisnik aplikacije može unjeti najveći mogući broj preuzimanja prenijete datoteke i tako ograničiti mogućnost preuzimanja datoteke na onolik broj preuzimanja koji on vjeruje da je shodan.
- Vremensko ograničenje trajanja poveznice je još jedna metoda kontrole pristupa implementirana u aplikaciju. Prilikom postavljanja datoteke može se odabrati vrijeme trajanja poveznice od jednog do dvadeset i četiri sata prilikom čega je početna vrijednost dvadeset i četiri. Ova značajka, iako ograničavajuća zbog kratkog trajanja poveznice, omogućava stabilniji rad aplikacije i optimalnije korištenje resursa diska zbog toga što onemogućava gomilanje datoteka s razmjerno kasnim vremenom isteka trajanja poveznice.
- Pritiskom na dugme prenesi se datoteka i pripadajuće postavke prenose na server te se unutar skočnog prozora prikaže poveznica koja se može kopirati klikom na dugme ili označivanjem okvira za tekst.
- Svi prethodno spomenuti elementi trebaju raditi jednako pouzdano neovisno o aspektu ekrana na raznovrsnim uređajima uključujući računala, tablete te telefone. Prijenos dato-

teka treba biti omogućen sa svih prethodno spomenutih uređaja.

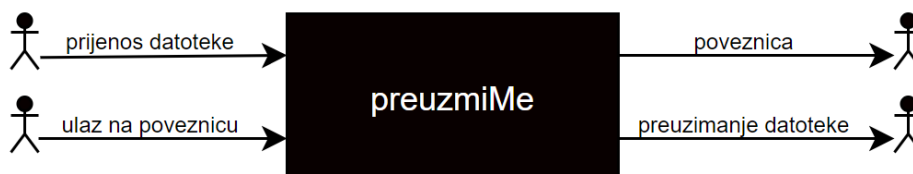
2.1.2. Zahtjevi prilikom preuzimanja datoteke

- Prilikom ulaza na poveznicu korisniku se unutar sekcije na sučelju ispisuje ime i ekstenzija(tip) datoteke ukoliko postoji datoteka povezana s tom poveznicom i nije zaštićena zaporkom. Ukoliko ne postoji ispisuje se pripadajuća poruka ovisno o tome je li poveznica istekla ili nije postojala.
- Ukoliko je datoteka zaštićena zaporkom pojavljuje se skočni prozor u kojem korisnik unosi zaporku te ako je zaporka ispravna se korisniku omogućava prikaz imena i ekstenzije te samo preuzimanje.
- Pritiskom na dugme preuzmi korisniku se željena datoteka sprema na uređaj.
- Svi prethodno spomenuti elementi trebaju biti funkcionalni neovisno o tome koji uređaj korisnik koristi: računalo, tablet ili telefon.

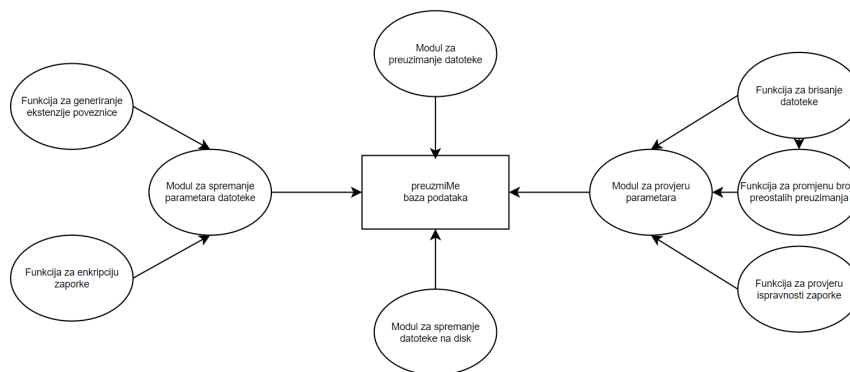
2.2. Dizajn sustava

2.2.1. Modeli sustava

Zbog jasne namjene aplikacije model crne kutije aplikacije preuzmiMe je prilično jednostavan što je razvidno iz slike 2.1. U taj model su radi prikaza namjene aplikacije ubačene samo dvije principijelne funkcionalnosti aplikacije kroz parove ulaz i izlaz za zadane funkcionalnosti, a spomenute funkcionalnosti su postavljanje i preuzimanje datoteke. Ostale funkcionalnosti aplikacije poput zaštite zaporkom se mogu smatrati dijelom osnovnih funkcionalnosti.



Slika 2.1: Model crne kutije za preuzmiMe.



Slika 2.2: Model bijele kutije za preuzmiMe.

Model bijele kutije je efektivniji u prikazivanju sustava i to ne iz pogleda korisnika za kojeg su važni isključivo ulaz i izlaz, već prikazuje od kojih se modula i funkcija sustav sastoji što je grafički prikazano na slici 2.2.

Moduli i funkcije sustava su:

- **Modul za spremanje parametara datoteke** - sprema informacije o datoteci te lokaciju na disku u kojem je spremljena u bazu podataka
- **Funkcija za enkripciju zaporku** - poziva ju modul za spremanje parametara datoteke kako bi unesenu zaporku enkriptirala prije spremanja u bazu podataka.
- **Funkcija za generiranje ekstenzije poveznice** - poziva ju modul za spremanje parametara datoteke u cilju generiranja ekstenzije poveznice na kojoj će se nalaziti datoteka za preuzimanje. Funkcija generira šesnaest alfanumeričkih znakova te provjerava da ista sekvenca ne postoji već u bazi podataka te ako postoji se poziva opet. Zbog količine znakova u ekstenziji prilično je mala šansa da se datoteci može pristupiti pogađajući ekstenziju.
- **Modul za spremanje datoteke na disk** - dodaje datoteci jedinstvenu oznaku kako bi onemogućili postojanje više datoteka s istim imenom na disku, sprema datoteku na disk te vraća ime i relativnu lokaciju datoteke.
- **Modul za provjeru parametara** - prilikom zahtjeva za preuzimanje na osnovu poveznice poslana u zahtjevu provjerava postoji li datoteka povezana uz tu ekstenziju poveznice, te na osnovu parametara poput broja preostalih preuzimanja, vremena isteka poveznice, zaporku poziva odgovarajuću funkciju, vraća vrijednost ili radi oboje.
- **Funkcija za provjeru ispravnosti zaporku** - provjerava odgovara li unesena zaporka postojećoj uz datoteku te u slučaju da odgovara vraća modulu za provjeru parametara potvrđan odgovor što omogućava preuzimanje datoteke korisniku.

- **Funkcija za brisanje datoteke** - poziva ju modul za provjeru parametra ako je vrijeme trajanja poveznice isteklo ili funkcija za promjenu broja preostalih preuzimanja ako je broj preostalih preuzimanja jednak nuli. Funkcija briše datoteku s diska te podatke o datoteci iz baze podataka.
- **Funkcija za promjenu broja preostalih preuzimanja** - poziva se kada je datoteka poslana korisniku, smanjuje broj preostalih preuzimanja za jedan, ako je nakon promjene broj preostalih preuzimanja jednak nuli poziva funkciju za brisanje datoteke.
- **Modul za preuzimanje datoteke** - dohvaća datoteku za slanje korisniku te u odgovoru zahtjeva šalje datoteku te njeno originalno ime.

2.2.2. Model podataka

Podaci o datoteci su u bazi podatka spremljeni u tablicu "*file_properties*" čiji atributi zajedno s tipom podatka i ograničenjima polja su vidljivi u tablici 2.1.

Tablica 2.1: Tablica atributa tablice *file_properties*

<i>file_properties</i>			
Atribut	Ključ	Tip	Ograničenja
id	PK	integer	≠ NULL
link		string	≠ NULL, UNIQUE
path		string	≠ NULL
type		string	
time_uploaded		dateTime	
time_expires		dateTime	
password		string	
num_dl_left		integer	

3. TEHNOLOGIJE

3.1. React

React je JavaScript biblioteka razvijena od strane Facebook-a za izradu korisničkih sučelja. Tri su odrednice React-a, a to su: deklarativnost koja omogućava jednostavno kreiranje interaktivnog korisničkog sučelja i implementaciju promjena u samo one komponente koji konzumiraju informacije koje su se promijenile, baziranost na komponentama što čini jednostavnim pisanje i snalaženje u kodu te najvažnije svaka komponenta sadrži svoje stanje koje je ne ovisi nužno o stanjima ostalih komponenti te zadnja odrednica interoperabilnost s mnoštvom drugih jezika prilikom izrade aplikacije.[1]

Primjenu React-a u aplikaciji je najbolje elaborirati kroz navođenje nekih koncepata koji čine React te dati primjer korištenja tih koncepata u aplikaciji.

Komponente u React-u mogu biti funkcije ili klase pri čemu je funkcijska komponenta doslovna JavaScript funkcija kao što se može vidjeti iz primjera koda 3.1. Kako je u primjeru komponenta jedna JavaScript funkcija koja vraća HTML kod, važno za napomenuti za React je da cijeli HTML kod komponente mora biti sadržan unutar jednog enkapsulirajućeg elementa. Komponente također primaju *prop*-ove koji predstavljaju podatke prosljeđene komponenti od strane komponentata roditelja, pri čemu je važno za napomenuti da se vrijednosti *props*-a unutar komponente ne smiju mijenjati [2].

Enkapsulacija je također jedna odrednica React-a koja omogućava prosljeđivanje HTML elemenata komponenti kao elementi djeca *prop*-a i stavljaju se između *HTMLtag*-ova komponente. Primjer enkapsulacije se može vidjeti u kodu 3.1. Enkapsulacija se koristi kada komponente ne znaju koji će im elementi biti elementi djeca, najčešće u slučaju kada komponenta ima neku općenitiju primjenu i koristi se više puta na različitim mjestima u kodu kao u danom primjeru [3].

Kako su se u izradi aplikacije koristile isključivo funkcijske komponente, stanja neće biti obrađena na klasičan način kao kod klasa što se može vidjeti u [5], već pomoću *useState hook*-a koji omogućava korištenje stanja *React* koja su u prijašnjim verzijama bila rezervirana isključivo

za klasne komponente, u funkcijskim komponentama [6]. Tako se stanje koristi na jednostavan način postavljanjem inicijalne vrijednosti, što se vidi u kodu 3.2.

Kod 3.1: Primjer komponente i enkapsulacije u React-u

```
export default function OptionWrapper(props) {
  const classes = useStyles();
  return (
    <div className={classes.root}>
      <Grid container>
        <Grid item xs={12}>
          <Paper className={classes.paper}>
            {props.children}
          </Paper>
        </Grid>
      </Grid>
    </div>
  )
}
```

Kod 3.2: Primjer korištenja stanja u funkcijskim komponentama

```
const [password, setPassword] = useState(null);
const [download, setDownload] = useState(null);
const [expire, setExpire] = useState(null);
```

3.1.1. Material-ui

Material-ui je biblioteka prilagodljivih komponenti koje olakšavaju izradu uniformnih te pristupačnih sučelja. Biblioteka se koristi principima dizajnerskog pravca *material design* kod oblikovanja izgleda komponenti. Međutim, biblioteka sadrži ne samo komponente koje poprimaju izgled, već i principe razmještanja i pisanja stilova, što se sve koristilo prilikom izrade aplikacije [10]. Primjer korištenja više aspekata biblioteke je vidljiv u kodu 3.3.

Kod 3.3: Primjer korištenja material-ui biblioteke

```
const useStyles = makeStyles((theme) => ({
  root: {
    flexGrow: 1
  },
  optionsGrid: {
```

```

        height: "50vh",
        backgroundColor: "#FFF2CC",
        borderRadius: "0.8em",
    },
    mainGrid:{
        flexGrow: 1,
        marginTop:"22.5vh"
    },
    ));

export default function ContentRow() {
    const [file , setFile] = useState(null);
    const classes = useStyles();

    return (
        <Grid className={classes.mainGrid} container
            direction="row" justify="center"
            alignItems="center" spacing={3}>
            <UploadOrDrag setFileUoD={ setFile }/>
            <OptionsContainer fileO={ file }/>
        </Grid>
    )
}

```

3.2. Node.js

Node.js nam omogućava programiranje poslužiteljske aplikacije u *JavaScript* okruženju. *Node* je okruženje otvorenog koda i nije vezano za niti jednu specifičnu platformu te pošto je namijenjen za rad na operacijskom sustavu uređaja, a ne u pregledniku, nema ograničenja koja imaju aplikacije vezane za preglednike poput primjerice rada s datotečnim sustavom. Upravo to što se radi o *JavaScript*-u je jedna od najvećih pogodnosti *node*-a jer je moguće pomoću samo jednog jezika napisati kompletnu aplikaciju uključujući i korisnički i poslužiteljski dio, kao i stotine tisuća paketa koji postoje da bi olakšali izradu aplikacije te cijela zajednica ljudi koja se oko *JavaScript*-a neprestano širi. *Node* se također izdvaja zbog svojih zavidnih performansi, lake optimizacije prilikom skaliranja i mnogih drugih benefita koji proizlaze iz toga što je jezik koji se za njega koristi mlad i često podlozan poboljšanjima [12].

3.2.1. Bcrypt

Bcrypt je paket za *Node.js* koji implementira istoimeni algoritam za enkripciju zaporki. *Bcrypt* koristi *Blowfish* algoritam te uvodi neke dodatne funkcionalnosti poput faktora rada te to u kombinaciji omogućava priličnu razinu sigurnosti za zaporku značajnu smanjujući mogućnost *brute-force* napada zbog sporosti enkripcije koja je i nekoliko redova veličine veća od sporih *hash* funkcija poput MD5 ili SHA1 [8]. Primjer korištenja *bcrypt*-a unutar *preuzmiMe* je dan u kodu 3.4

Kod 3.4: Primjer korištenja bcrypt-a

```
let salt = await bcrypt.genSalt(
  Number(config.bcrypt.noSaltRounds));

let encryptedPassword = await bcrypt.hash(password, salt);
```

3.2.2. Multer

Multer je *middleware* za *express* koji omogućava spremanje datoteka na zadanu lokaciju te dohvaćanje informacija o datoteci poput originalnog imena [9]. To omogućava spremanje datoteka na razmjerno lak način što je vidljivo iz koda 3.5.

Kod 3.5: Primjer korištenja multer-a

```
const storage = multer.diskStorage({
  destination: function (req, file, cb) {
    cb(null, config.fileRootPath);
  },
  filename: function (req, file, cb) {
    console.log(Date.now() + '-' + file.originalname);
    cb(null, Date.now() + '-' + file.originalname);
  }
});

const filter = (req, file, cb) => {
  if (file.mimetype ===
    'application/vnd.microsoft.portable-executable'
    || file.mimetype === 'application/octet-stream')
  {
```

```

        cb(new Error(file.mimetype + ' is not supported '), false)
    } else {
        cb(null, true);
    } ;
}
const upload = multer({
    storage: storage,
    limits: { fileSize: config.maxFileSize },
    fileFilter: filter
});

```

3.2.3. Sequelize

Sequelize je ORM koji omogućava komunikaciju između poslužitelja i odabrane baze podataka. *Sequelize* tablice u bazi podataka predstavlja kao modele što omogućava laku izmjenu bez direktne interakcije korisnika s bazom podataka. Predstavljanje tablica kao modela omogućava jednostavnu promjenu baza podataka. Pomoću poveznice na bazu podataka *sequelize* otvara konekciju te je iz *JavaScript*-a omogućeno dohvaćanje i manipulacija podacima u bazi podataka [11]. Primjer dohvaćanja retka tablice podataka po atributu je dan u kodu 3.6

Kod 3.6: Dohvaćanje podataka pomoću sequelize-a

```

let fileToDelete = await FileProperties.findOne({
    where: { link: link }
});

```

3.3. Express.js

Express.js služi kao nadogradnja *node* aplikacija zato što omogućava rukovanje s različitim HTTP zahtjevima na različitim rutama kao i omogućava postavljanje određenih portova koje aplikacije koriste za povezivanje. Iako sam relativno ograničen, *express* pruža odličnu podlogu za kojekakve *middleware* pakete koji su izrađeni za rješavanje specifičnih problema prilikom izrade aplikacija, te je vrijedno napomene to da se ti paketi mogu uključiti bilo gdje prilikom rukovanja zahtjevima. Jedna odrednica *express*-a je da ne postoji jedan ispravan način za napraviti nešto, već ista stvar može implementirati na više načina što s jedne strane daje mnogo veću fleksibilnost i primjenjivost dok s druge strane znači da je teže razumjeti i naći načine za implementaciju neke potrebne funkcionalnosti [12]. Primjer upravljanja rutama je dan u kodu 3.7

Kod 3.7: Upravljanje rutama pomoću *express-a*

```
const express = require('express');
const download = express.Router();
const downloadController =
require('../controllers/downloadController');

module.exports = function (router){

    router.use('/', download);
    download.get('/:unique_URL', downloadController.download);

}
```

3.4. PostgreSQL

PostgreSQL je relacijska baza podataka koja koristi i nadograđuje se na SQL jezik s mnogim značajkama koje omogućavaju sigurnost i skalabilnost čak i za velike baze podataka. *PostgreSQL* se koristi zbog svoje pouzdanosti, arhitekture, integriteta podataka, velikog seta funkcionalnosti uz to što podržava rad na svim većim operacijskim sustavima. Neke od funkcionalnosti uključuju mogućnosti definiranja vlastitih tipova podataka i izradu i prilagodbu funkcija. *PostgreSQL* se prilagođava svim SQL standardima koji ne dovode u pitanju arhitekturu same baze ili su oprećni nekim postojećim značajkama, pri čemu je većina standarda sadržana s eventualnim manjim razlikama u sintaksi [13].

4. IMPLEMENTACIJA APLIKACIJE

4.1. Korisničko sučelje

Implementaciji aplikacije je pristupljeno prvo kroz izradu korisničkog sučelja. Za početak je potrebno kreirati React aplikaciju, što je moguće pomoću prethodno instaliranog i na računalu globalno dostupnog Node.js paketa *create-react-app* [4]. Primjer kreiranja aplikacije pomoću spomenutog paketa je dan u kodu 4.1. *create-react-app* generira sve potrebne datoteke i povezuje ih na način da korisnik može odmah krenuti s programiranjem aplikacije.

Kod 4.1: Bash naredba za kreiranje React aplikacije

```
npx create-react-app preuzmime
```

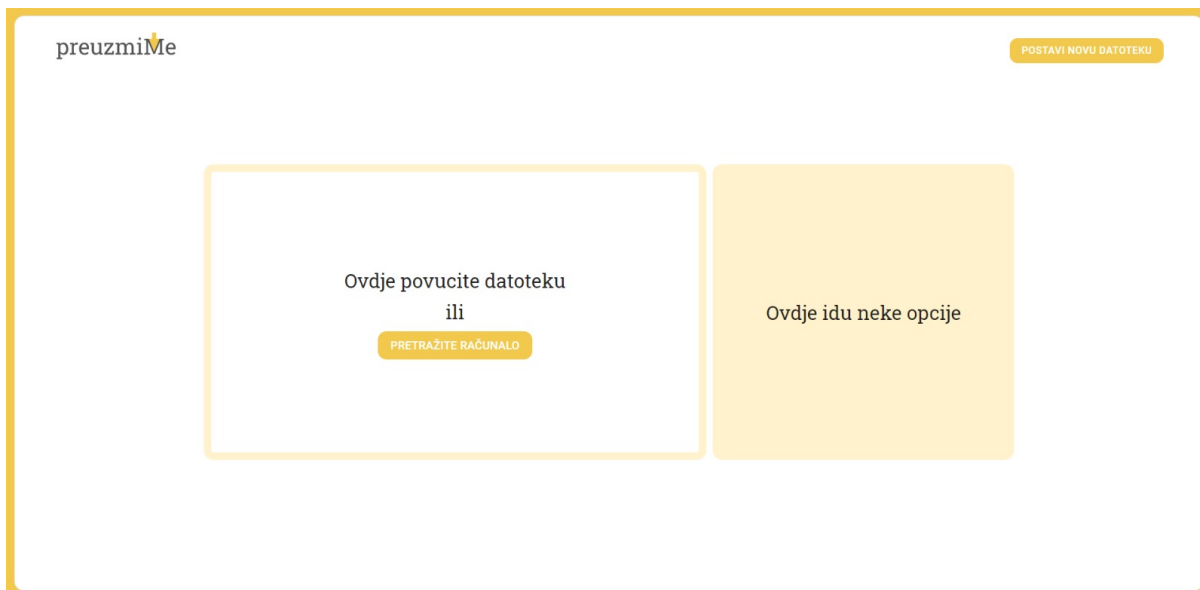
Nakon generiranja svih potrebnih datoteka može se pristupiti izradi aplikacije. Komponente se izrađuju te se postavljaju u druge datoteke imaju vezu s izvorišnom datotekom *App.js* ili su direktno dodane u izvorišnu datoteku. Pokretanje aplikacije je jednostavno pomoću prethodno spomenutog *create-react-app* paketa. Bash naredba je vidljiva u kodu 4.2

Kod 4.2: Bash naredba za pokretanje React aplikacije

```
npm start
```

Fokus prilikom izrade aplikacije *preuzmiMe* je bila jednostavnost i intuitivnost korištenja aplikacije od strane korisnika i zato je bio potreban dizajn sučelja koji odražava te dvije odrednice. Prilikom dizajna je također trebalo voditi računa o postojanju vizualnog identiteta aplikacije i zadržavanju istog kroz sve elemente sadržane u aplikaciji, bilo to kroz boje, oblike, stilove ili fontove. Ta jednolikost je bila glavna misao vodilja prilikom preliminarne izrade komponenata i njihovog pozicioniranja na sučelju što se može vidjeti na slici 4.1.

Na rubovima cijelog sučelja se nalazi obrub narančaste boje koja je uzeta za boju koja će biti svojstvena aplikaciji. Na gornjem dijelu sučelja se nalazi navigacijska traka s logom aplikacije i gumbom koji vodi na stranicu za postavljanje novog dokumenta s bilo koje stranice unutar aplikacije. Po sredini sučelja za postavljanje se nalaze dvije sekcije u kojima

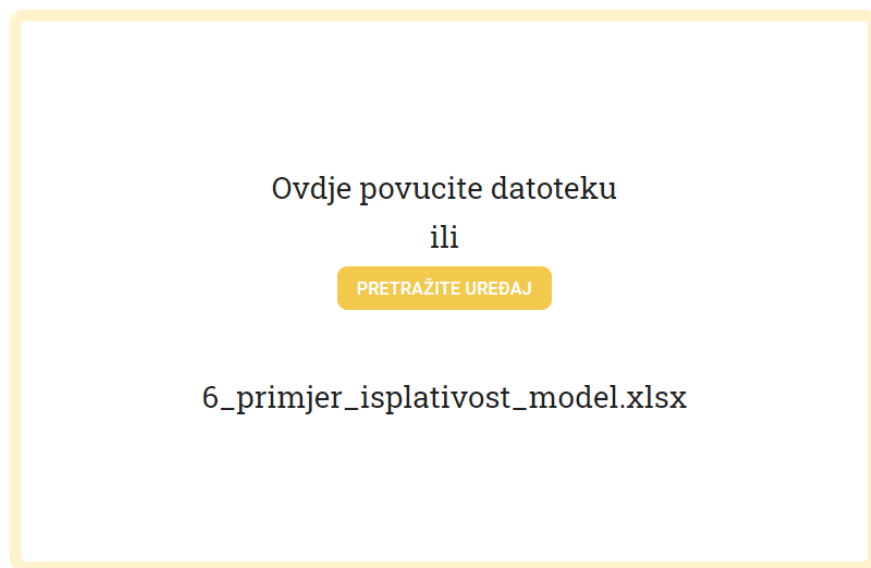


Slika 4.1: Pozicioniranje elemenata na sučelju

su sadržane sve bitne funkcionalnosti. U prvu sekciju korisnici mogu dovući datoteku iz bilo kojeg direktorija računala ili klikom na sekciju otvoriti izbornik datoteka te na taj način prenjeti datoteku. U drugoj sekciji će biti sadržane opcije nad datotekom i gumb koji služi za postavljanje datoteke na poslužitelj, međutim u inicijalnoj izradi je fokus bio na razmještanju elemenata na sučelju, a ne o svim funkcionalnostima gotove aplikacije. Također se vodilo računa o razmještanju komponenti na različitim uređajima te su na mobilnim uređajima dvije glavne sekcije razmještene jedna pod drugom umjesto jedna uz drugu.

Nakon osnovnog razmještanja glavnih sekcija sučelja i testiranja ponašanja na različitim uređajima, fokus se prebacio na dodavanje elemenata završne aplikacije, počevši od pogleda za postavljanje datoteke. Kod pogleda za postavljanje datoteke je potrebno dodati funkcionalnost za dovlačenje datoteke u jednu od glavnih sekcija te opcije za zaštitu datoteke zaporkom, broj mogućih preuzimanja datoteke te vrijeme trajanja datoteke kako bi se ispunili svi zahtjevi spomenuti u poglavlju 2.

U sekciju za prijenos datoteke su implementirane sve mogućnosti uz pomoć *react-dropzone* paketa koji omogućava prijenos datoteka povlačenjem te ograničavanje nekih parametara poput broja prenesenih datoteka, veličine datoteka, tipa datoteke među ostalim opcijama [7]. Prilikom prijenosa datoteke, ukoliko zadovoljava svim ograničenjima o veličini i broju datoteka, se pojavljuje ime prenesene datoteke kako bi korisnik znao da je datoteka uspješno prenjeta što se može vidjeti na primjeru slike 4.2. Time je kompletirana sekcija za prijenos datoteke izuzev propagacije podataka unutar React stabla komponenata, što će biti napravljeno u fazi povezivanja, nakon implementacije serverskog dijela aplikacije.



Slika 4.2: Sekcija za prijenos datoteke

U sekciju za opcije je potrebno postaviti tri komponente za upis vrijednosti opcija, gumb kojim se označava prihvatanje uvjeta korištenja koji se pišu u slučaju da je aplikacija postavljena na Internet za proizvoljne korisnike te gumb za prijenos datoteke i opcija na poslužitelj. Za opcije su rađene tri zasebne komponente iz dvojakog razloga od kojih prvi ima veze s ujednačenosti dizajna zbog obruba kojeg svaka komponenta ima te drugog, tehnički važnijeg razloga, da se stanje određene komponente mijenja samo kada se mijenjaju vrijednosti u polju određene opcije. Prva opcija je zaporka, u nju se unosi zaporka koja nije ograničena ni dužinom niti setom znakova iz razloga što se razina sigurnosti zaporke ostavlja korisniku na diskreciju te nije od važnosti kao primjerice zaporka korisničkog računa u nekim konvencionalnim aplikacijama. Druga opcija definira broj mogućih preuzimanja datoteke s iste poveznice te kao i zaporka ne mora poprimiti vrijednost, te je jedino ograničenje da unesena vrijednost mora biti broj. Treća opcija je vrijeme trajanja poveznice koje je unaprijed postavljeno na maksimalnu vrijednost od dvadesetčetiri sata te se iz izbornika mogu samo odabrati vrijednosti od jednog sata do dvadesetčetiri sata iz razloga spomenutih u zahtjevima poglavlja 2. Ispod opcija je gumb u vidu kvačice koji onemogućava prijenos datoteke na poslužitelj ukoliko nisu prihvaćeni prethodno spomenuti uvjeti korištenja, što je vidljivo iz slike 4.3. Ako su uvjeti prihvaćeni omogućen je prijenos datoteke na poslužitelj što se radi pomoću posljednjeg gumba sekcije za opcije te je primjer proizvoljno popunjene sekcije za opcije dan na slici 4.4. Propagacija podataka je rješena na način da se prilikom promjene stanja opcije ta stanja šalju elementu roditelju iz kojeg gumb za prijenos datoteke na poslužitelj preuzima podatke kada šalje zahtjev. Implementacija spomenutog zahtjeva je ostavljena za fazu povezivanja.

Zadnji element kod pogleda za prijenos datoteke je skočni prozor koji se otvori kada se

Opcije (*neobavezno*):

Zaporka:

Broj preuzimanja:

Vrijeme trajanja(0-24h):

☐ Prihvaćam opće uvjete korištenja

PRENESI DATOTEKU

Slika 4.3: Sekcija za opcije s onemogućenim prijenosom

Opcije (*neobavezno*):

Zaporka:

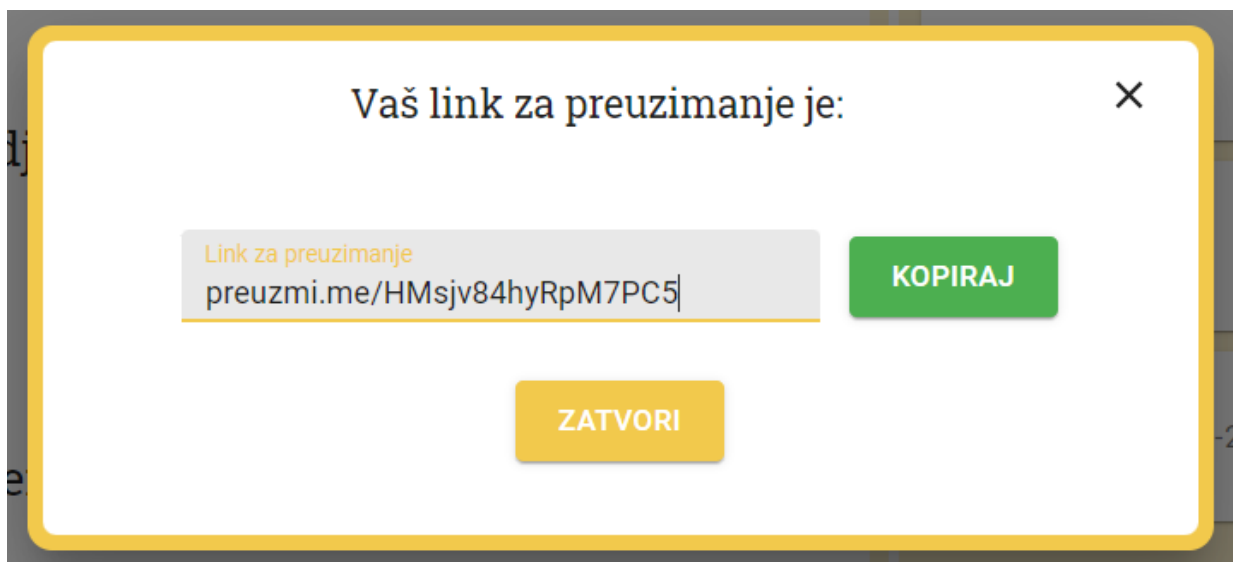
Broj preuzimanja:

Vrijeme trajanja(0-24h):

☒ Prihvaćam opće uvjete korištenja

PRENESI DATOTEKU

Slika 4.4: Sekcija za opcije s omogućenim prijenosom



Slika 4.5: Skočni prozor s poveznicom

datoteka uspješno postavi na poslužitelj. Taj skočni prozor sadrži poveznicu pomoću koje se može preuzeti datoteka. Poveznicu se može kopirati u međuspremnik klikom na gumb desno od polja u kojem se nalazi poveznica ili označivanjem poveznice i kopiranjem pomoću nekog prečaca. Skočni prozor također sadrži gumb za zatvaranje prozora, prozor se alternativno može zatvoriti pritiskom bilo gdje unutar stranice van samog prozora ili na gumb u gornjem desnom kutu. Izgled prozora je vidljiv iz slike 4.5. Potpuni izgled sučelja je vidljiv na slici 4.6 za računala te slici 4.7 za mobilne uređaje.

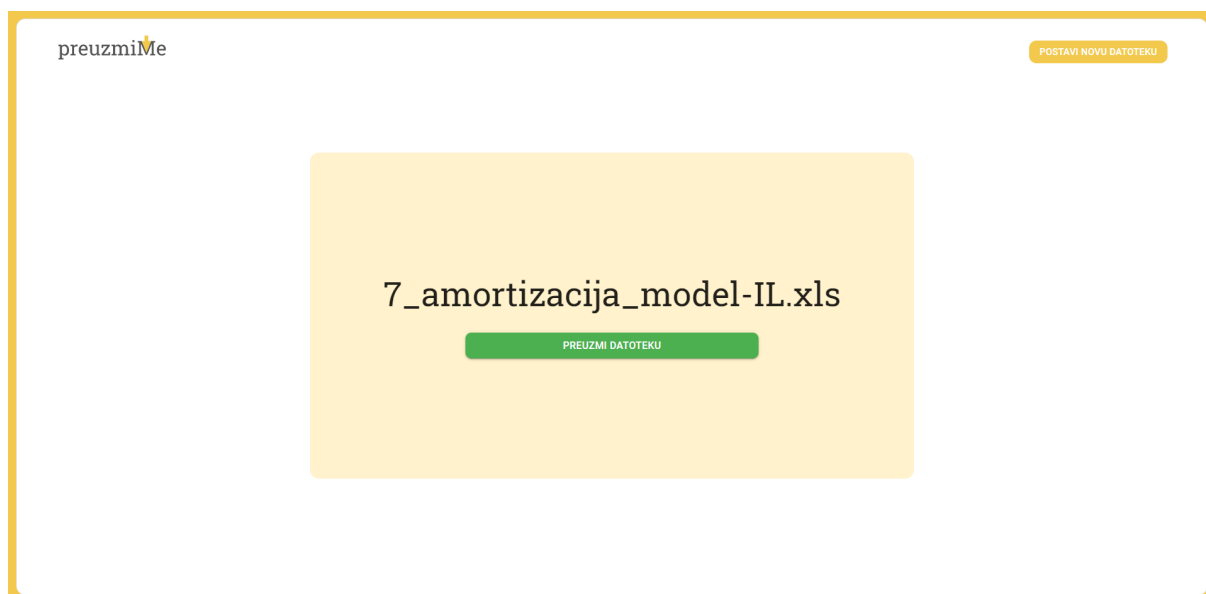
Kod pogleda za preuzimanje datoteke postoji jedna glavna sekcija unutar koje se nalazi ime datoteke zajedno s ekstenzijom tipa datoteke te gumb za preuzimanje datoteke kao što je vidljivo iz slike 4.8. Umjesto imena datoteke je moguće dobiti poruku ako poveznica nije valjana ili je istekla vidljivo na slici 4.9. U slučaju da je datoteka zaštićena zaporkom pojavljuje se skočni prozor nalik onom koji se pojavljuje kod pogleda za prijenos datoteke. Skočni prozor sadrži element za unos zaporke, gumb za provjeru ispravnosti zaporke te gumbe za zatvaranje skočnog prozora. Izgled skočnog prozora se može vidjeti u slici 4.10. Ako je zaporka ispravna omogućava se preuzimanje datoteke. Zahtjev za datotekom se implementira u fazi povezivanja.

The desktop interface features a header with the 'preuzmiMe' logo on the left and a 'POSTAVI NOVU DATOTEKU' button on the right. The main content area is divided into two sections. The left section contains the text 'Ovdje povucite datoteku ili' followed by a yellow button labeled 'PRETRAŽITE UREDAJ'. The right section, titled 'Opcije (neobavezno):', includes three input fields: 'Zaporka:' with a 'Unesite zaporku' button, 'Broj preuzimanja:' with a 'Unesite broj' button, and 'Vrijeme trajanja(0-24h):' with a dropdown menu set to '24h'. Below these is a checkbox for 'Prihvaćam opće uvjete korištenja' and a 'PRENESI DATOTEKU' button.

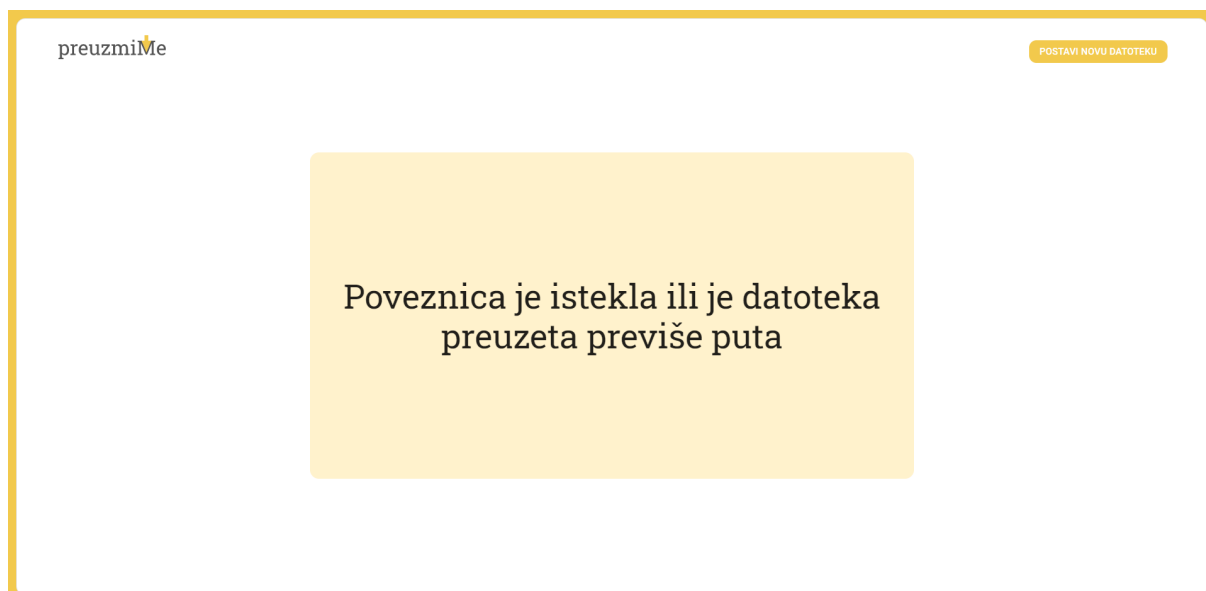
Slika 4.6: Izgled pogleda za prijenos

The mobile interface is a vertical layout. At the top, it shows the 'preuzmiMe' logo and a 'POSTAVI NOVU DATOTEKU' button. Below this is a large white box with the text 'Ovdje povucite datoteku ili' and a yellow 'PRETRAŽITE UREDAJ' button. At the bottom, there is a section titled 'Opcije (neobavezno):' containing a 'Zaporka:' input field with a 'Unesite zaporku' button.

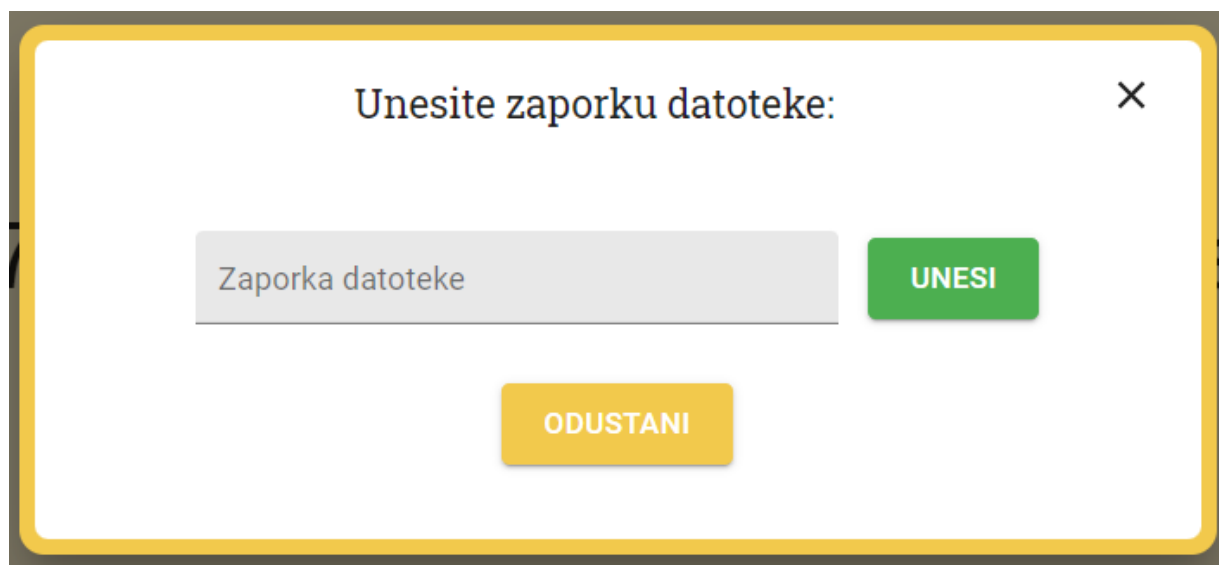
Slika 4.7: Izgled pogleda za prijenos na mobilnim uređajima



Slika 4.8: Izgled pogleda za preuzimanje datoteke



Slika 4.9: Pogled za prijenos kada poveznica nije valjana



A screenshot of a file password dialog box. The dialog has a white background with a yellow border. At the top, the text "Unesite zaporku datoteke:" is displayed in a dark blue font, followed by a close button (X) in the top right corner. Below the text is a light gray rectangular input field containing the placeholder text "Zaporka datoteke". To the right of the input field is a green button with the text "UNESI" in white. Below these elements is a yellow button with the text "ODUSTANI" in black.

Slika 4.10: Skočni prozor za unos zaporke

4.2. Poslužiteljski dio aplikacije

Izrada poslužiteljskog dijela aplikacije počinje izradom instance PostgreSQL baze podataka te dohvaćanja poveznice preko koje se poslužitelj može povezati na bazu podataka. Idući korak za postavljanje baze podataka je izrada tablice *FileProperties* pomoću *Sequelize ORM*-a nakon inicijalizacije samog *ORM*-a. U tablicu se dodaju svi atributi spomenuti u poglavlju 2. Kod za izradu tablice se može vidjeti u kodu 4.3. Sljedeći korak je migracija tog koda u bazu podataka, to jest kreiranje same tablice u bazi podataka. To se radi pomoću naredbe dane u kodu 4.4.

Kod 4.3: Kod za izradu FileProperties tablice

```
module.exports = (sequelize, DataTypes) => {  
  
  FileProperties.init({  
    link: DataTypes.STRING,  
    path: DataTypes.STRING,  
    type: DataTypes.STRING,  
    time_uploaded: DataTypes.DATE,  
    time_expires: DataTypes.DATE,  
    password: DataTypes.STRING,  
    num_dl_left: DataTypes.INTEGER,  
  }, {  
    sequelize,  
    modelName: 'FileProperties',  
  });  
  return FileProperties;  
};
```

Kod 4.4: Naredba za migraciju modela u bazu podataka

```
sequelize db:migrate
```

Jednom kada je tablica vidljiva u bazi podataka, potrebno je povezati bazu podataka s poslužiteljem tako da u normalnom radu datoteke poslužitelja mogu pristupiti podacima u bazi podataka. Ta konekcija se otvara iz korijenske datoteke poslužitelja *app.js* koja poziva *index.js* datoteku mape *loaders* koja pak poziva *sequelize.js* datoteku koja otvara samu konekciju. Kod otvaranja je vidljiv u kodu 4.5. Iako je struktura pozivanja od korijenske datoteke do datoteke koja otvara samu konekciju naizgled zamršena, na taj način se postiže čistoća i čitkost koda te olakšavaju bilo kakve eventualne izmjene, jer je svaka funkcionalnost zasebno

odvojena, slično kako se stvaraju zasebne komponente na korisničkom sučelju. Na isti se način, učitava konfiguracija *express*-a te tzv. *logger*-i koji služe kao pomoć za otklanjanje grešaka ispisivanjem opisa greške u konzolu. Tako da korijenska datoteka pozivom jedne funkcije unutar *index* datoteke *loaders* mape učitava sve potrebno za daljni rad aplikacije.

Kod 4.5: Otvaranje konekcije s bazom podataka

```
module.exports = {
  SequelizeDBConnection: async function () {
    try {
      await sequelize.authenticate();
      console.log("Connected to database.");
    } catch (error) {
      console.error("Unable to connect to the database:", error);
      process.exit(1);
    }
  }
}
```

Nakon što je omogućeno pokretanje aplikacije, servisi su sljedeće bitne komponente aplikacije. Servisi sadrže svu logiku aplikacije, komunikaciju s bazom podataka te se sva manipulacija podacima tu događa. Kod preuzmiMe servisi mapa sadrži *index.js* datoteku koja služi za instanciranje dvije glavne klase, koje se nalaze u istoimenim datotekama, a to su *upload* i *download* klase odnosno servisi koji pozivaju pomoćne servise za neku funkcionalnost.

Upload servis se brine o generiranju nasumične ekstenzije poveznice, enkriptiranju zaporce ako ista postoji, spremanju parametara datoteke, davanju jedinstvenog imena te spremanje datoteke na poslužitelj. *Upload* servis također brine o ispravnosti pristiglih podataka, koji bi trebali pristići s korisničke strane unutar zadanih ograničenja poput veličine datoteke te vremena isteka manjeg od jednog dana, međutim kako bi se izbjegla svaka moguća greška se vrše provjere ispravnosti. *Upload* servis poziva pomoćni *generatelink* servis koji generira i vraća ekstenziju poveznice te provjerava da ne postoji ista, vidljivo u kodu 4.6. Ukoliko postoji, zaporka se šalje pomoćnom servisu *encrypt.js* koji pomoću *bcrypt* paketa enkriptira zaporku te je vraća *upload* servisu koji je potom sprema u bazu podataka. Kod enkripcije zaporce u *encrypt* servisu je dan u kodu 4.7. Pomoću *multer* servisa se datoteka sprema na poslužitelj te joj se daje jedinstveno ime koje se sprema u bazu podataka kao lokacija na kojoj se nalazi datoteka. Nakon što su sve prethodne radnje obavljene, *upload* servis sprema parametre datoteke u bazu podataka te vraća poveznicu na datoteku.

Kod 4.6: Generiranje ekstenzije poveznice

```
module.exports = async function(){

  try {

    let linkSearch = 1;
    let link;

    while(linkSearch){

      link = randomString.generate(16);

      linkSearch = await FileProperties.findOne({
        where: {link: link}
      })

    }

    logger.info("Link extension has been generated: "+ link)
    return link;

  } catch (error) {

    logger.error("Error in generating link: "+ error);
    throw(error)

  }

}
```

Kod 4.7: Enkripcija zaporke

```
module.exports = async function(password){

  try {

    let salt = await bcrypt.genSalt(
      Number(config.bcrypt.noSaltRounds));
```

```

    let encryptedPassword = await bcrypt.hash(password, salt);

    logger.info("Password has been encrypted: "
        + encryptedPassword);
    return encryptedPassword;

} catch (error) {

    logger.error("Error in encrypting password: "+ error);
    throw(error)

}

}

```

Download servis prima ekstenziju poveznice s koje je poslan zahtjev te provjerava parametre datoteke te vraća odgovor na osnovu parametara. Također smanjuje broj mogućih preuzimanja prilikom svakog slanja datoteke, provjerava ispravnost zaporka te briše datoteke s isteklom poveznicom i datoteke koje su preuzete maksimalan broj puta. Ako datoteka s primljenom ekstenzijom poveznice ne postoji vraća poruku da ne postoji da poveznica. Ukoliko postoji zaporka za datoteku, a nije poslana u zahtjevu, servis će tražiti zaporku, te na osnovu odgovora vratiti datoteku ili onemogućiti preuzimanje iste. Ako je broj preuzimanja nakon smanjenja mogućih preuzimanja jednak nula ili je datum isteka poveznice manji od trenutnog datuma, datoteka i informacije o datoteci u bazi podataka se brišu pomoću pomoćnog servisa *delete.js* čiji je kod dan u kodu [4.8](#).

Kod 4.8: Brisanje datoteke

```

module.exports = async function(link){

    try {

        let fileToDelete = await FileProperties.findOne({
            where: {link: link}
        });

        fs.unlinkSync(fileToDelete.path)
        await fileToDelete.destroy();

    }
}

```

```

        logger.info("File has been successfully deleted");

    } catch (error) {

        logger.error("Error in deleting file");
        throw(error)

    }

}

```

Config mapa sadrži datoteke koje služe za dohvaćanje varijabli okruženja poput poveznice za povezivanje na bazu podataka i slične podatke koje ne želimo da su vidljive u kodu od strane bilo koga. Kada vanjske datoteke trebaju pristup nekoj od spomenutih varijabli, pozivaju odgovarajuću datoteku iz *config* mape.

API mapa služi za komunikaciju poslužitelja s korisničkim dijelom aplikacije te se sastoji od podmapa *routers* i *controllers*.

Datoteke *routers* mape su *upload.js* i *download.js* i one definiraju rute na poslužitelje na koje mogu stići zahtjevi da bi bili ispravno prosljeđeni odgovarajućim kontrolerima koji dalje šalju podatke servisima. *Upload* definira rute na koje se mogu slati zahtjevi za postavljanjem datoteke te je kod datoteke dan u kodu 4.9. *Download* definira što se šalje kontrolerima kada dolazi zahtjev s proizvoljnim ekstenzijama poveznice što su zapravo zahtjevi za preuzimanje, a ekstenzije poveznica su najčešće ekstenzije na kojima se nalaze datoteke.

Kod 4.9: Definiranje ruta

```

module.exports = function (router){

    router.use('/', upload);
    upload.post('/upload',
        uploadFile.single('file'),
        uploadController.upload);

}

```

U mapi *controllers* postoje datoteke *uploadController* i *downloadController* koji služe za pozivanje i prosljeđivanje podataka odgovarajućim servisima. Kontroler za postavljanje datoteke prima zahtjev, te šalje u *upload* servis sve podatke koji se nalaze u tom zahtjevu. Kod tog

kontrolera je vidljiv u kodu [4.10](#). Kontroler za preuzimanje datoteke poziva *download* servis te na osnovu odgovora dohvaća datoteku s lokacije na kojoj se nalazi ili vraća odgovor zašto je datoteka nedostupna.

Kod 4.10: Kontroler za postavljanje datoteke

```
module.exports = {  
  
  upload: async (req, res, next) => {  
  
    try {  
  
      let link = await uploadClassInstance.saveFileParameters(  
        req.file.mimetype, req.file.path,  
        req.body.timeUploaded, req.body.timeExpires,  
        req.body.password, req.body.dlNumber)  
  
        res.json(link);  
  
      } catch (error) {  
  
        logger.error('Error in upload controller');  
        next(error);  
  
      }  
  
    }  
  
  }  
  
}
```

ZAKLJUČAK

Kroz ovaj završni rad je izrađena i obrađena aplikacija koja služi kao oblak omogućavajući postavljanje i preuzimanje datoteka s poslužitelja. Aplikacija ne zahtijeva prijavu ni registraciju. Rađena je fokusom na intuitivnost i jednostavnost. Aplikacija sadrži mogućnosti postavljanja datoteke uz opcije postavljanja zaporke i maksimalnog broja mogućih preuzimanja te ograničavanje vremena trajanja poveznice dok je postavljanje jednostavno uz mogućnost odabira datoteke iz datotečnog sustava ili povlačenja s proizvoljne lokacije na računalu. Nakon postavljanja, aplikacija izrađuje dinamičnu poveznicu preko koje se datoteka može preuzeti. Kroz rad je opisana izrada funkcionalnosti i dizajn arhitekture sustava, spomenute su korištene web tehnologije i njihovi dodatci, te izrada aplikacije od početka do kraja kako korisničkog dijela aplikacije tako i poslužiteljskog dijela aplikacije.

LITERATURA

- [1] Facebook, „React”, s Interneta, <https://github.com/facebook/react>, 16.lipnja, 2021.
- [2] Facebook, „Components and props”,s Interneta <https://reactjs.org/docs/components-and-props.html>, 16.lipnja, 2021.
- [3] Facebook, „Composition vs Inheritance”, s Interneta <https://reactjs.org/docs/composition-vs-inheritance.html>, 18. lipnja, 2021.
- [4] Facebook, „create-react-app”, s Interneta <https://github.com/facebook/create-react-app>, 22. lipnja, 2021.
- [5] Facebook, „State and Lifecycle”, s Interneta <https://reactjs.org/docs/state-and-lifecycle.html>, 22. lipnja, 2021.
- [6] Facebook, „Using the State Hook”, s Interneta <https://reactjs.org/docs/hooks-state.html>. 22. lipnja, 2021.
- [7] react-dropzone, „react-dropzone”, s Interneta <https://react-dropzone.js.org/>, 22. lipnja, 2021.
- [8] Coda Hale, „How To Safely Store A Password”, s Interneta <https://codahale.com/how-to-safely-store-a-password/>, 27. lipnja,2021.
- [9] Dillion Megida, „Uploading files using Node.js and Multer”, s Interneta <https://blog.logrocket.com/uploading-files-using-multer-and-node-js/>, 27. lipnja 2021.
- [10] mui-org, „material-ui”, s Interneta <https://github.com/mui-org/material-ui>, 27. lipnja 2021.
- [11] educative, „What is sequelize.js?”, s Interneta <https://www.educative.io/edpresso/what-is-sequelizejs>, 4. srpnja 2021.
- [12] Mozilla, „Express/Node introduction”, s Interneta https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction, 7. srpnja 2021.
- [13] PostgreSQL, „About", s Interneta <https://www.postgresql.org/about/>, 10. srpnja 2021.

POPIS OZNAKA I KRATICA

Popis oznaka i kratica navodi se abecednim redom.

API	<i>Aplication programming interface</i>
HTTP	<i>Hyper text transfer protocol</i>
ORM	<i>Object-relational mapping</i>
<i>props</i>	<i>properties</i>
SQL	<i>Structured query language</i>

SAŽETAK

Sažetak:

Tema ovog završnog rada je bila izrada aplikacija koja omogućava postavljanje datoteke te preuzimanje iste pomoću poveznice koju aplikacija generira uz mogućnost zaštite datoteke pomoću opcija. Kod izrade korisničkog dijela aplikacije koristile su se *React* i *Material-ui* tehnologije, dok su se za izradu poslužiteljskog dijela aplikacije od tehnologija koristili *Node.js*, *Express.js*, *PostgreSQL*. U drugom poglavlju su opisane funkcionalnosti i dizajn sustava, treće poglavlje je dalo pregled korištenih tehnologija, a četvrto poglavlje je prikazalo izradu same aplikacije kroz implementaciju funkcionalnosti pomoću odabranih tehnologija.

Ključne riječi:

postavljanje, preuzimanje, datoteka, poveznica, aplikacija, zaštita

SUMMARY

Title: WEB APPLICATION PREUZMIME

Summary:

The topic of this final paper was the creation of an application that allows you to upload a file and download it using the link that the application generates with the ability to protect the file with options. React and Material-ui technologies were used to create the user part of the application, while Node.js, Express.js, PostgreSQL technologies were used to create the server part of the application. The second chapter describes the functionalities and design of the system, the third chapter gives an overview of the technologies used, and the fourth chapter shows the development of the application itself through the implementation of functionality using selected technologies.

Key words:

upload, download, file, link, application, security