АКАДЕМИЈА ТЕХНИЧКО-ВАСПИТАЧКИХ СТРУКОВНИХ СТУДИЈА

Rezervacija autobuskih karata

Predmet: Klijent Server Sistemi

Profesor
dr Mirko Kosanović
Miloš Kosanović

Student:
Nemanja Mihajlović
REr 3/19

10.01.2022.

Sadržaj

1. Uvod

U ovom projektu je urađena veb aplikacija za pretragu reda vožnje i rezervaciju autobuskih karata.Tehnologije korišnjenje na klijentskoj strani su HTML5,SCSS,Boostrap i JS. Na serverskoj strani je korišnjen Node JS sa Express frejmvorkom i Sequelizom koji je korišnjen za rad sa MySql bazom podataka. Aplikacija omogućava registraciju i prijavu korisnika,pretragu reda vožnje rezervaciju karata za izabrani termin. Aplikacija podržava više koristničkih rola.Obični korisnici mogu samo da pretražuju red vožnje i vide cene karata.Registrovani korisnici imaju mogućnost rezervacije/kupovine karata. Administratori imaju mogućnost dodavanja novih prevoznika,destinacija,kreiranja novih redova vožnje.kreiranja novih korisnika (prodavaca i konduktera). Prodavci (šalterski radnici) imaju mogućnost prodaje karata. Kondukteri mogu da očitaju kartu putnika.

2. Instalacija i podešavanje projekta

Kako bi se aplikacija pokrenula potrebno je da se instalira NodeJS i MySql baza podataka, zatim je potrebno dodati bazu sa nazivom „moja_rezervacija_dev“ i pokrenuti migracije komandom `„npx sequelize-cli db:migrate”.` Ova komanda će kreirati sve potrebne tabele.Zatim je potrebno izvršiti komandu `npx sequelize-cli db:seed:all` .Ova komanda će dodati jednog novog korisnika u bazu.Korisnik će biti admin, njegov e-mail će biti “admin@admin.com”, a loznika “mojBus123”.

2.1 Instaliranje modula

Projekat sadrži fajl “package.json”.Ovaj fajl sadrži spisak svih modula koje projekat koristi.Pošto uz projekat ne dolaze moduli, korisnik ih treba instalirati na svom računaru pre pokretanja.To radi pomoću komande “npm install”.Projekat koristi sledeće module:

- bcrypt (korišnjen je za hešovanje korisničkih lozinki)
- express
- express-session (korišćen je za čuvanje podataka o prijavljenom korisniku)
- jade (pug je korišnjen kao template engine,ali je on nastao iz jadea tj. jade je preimenovan u pug)
- multer (korišćen je za upload fajlova preko forme)
- mysql2 (za pristup bazi)
- pug (template engine)
- sequelize (ORM korišnjen je za rad sa bazom podataka)
- uuid (korišnjen je za genersianje random imena fajlovima pri uploadu)
- sequelize-cli (korišćen je za kreiranje modela i migracija preko komandnog interfejsa)

3. Arhitektura aplikacije

Prilikom izrade aplikacije korišnjen je MVC dizajn patern.Upotreblja se i Service layer,ali samo

u nekim delovima aplikacije,kako bi se sprečilo ponavljanje koda koji se koristi u više kontrolera i kako bi se pojedini kontroleri učinili preglednijim,prebacivanjem dela logike unutar servisa. Aplikacija sadrži sledeće foldere:

- node_modules - sadrži sve module koji su korišćeni
- routes – sadrži sve rute grupisane po folderima
- helpers – sadrži pomoćne metode koje koriste kontroleri i viewovi
- config sadrži fajl za konfiguraciju sequeliza
- public sadrži fajlove koji su javno dostupni
- services - sadrži servise koje upotrebljavaju kontroleri
- models – sadrži modele pošto projekat koristi MVC dizajn patern i koristi sequelize ORM
- controllers - sadrži kontrolere koji rade direktno sa modelima ili se servisima,zatim dobijene podatke prikazuju korisinku koristići određeni view kojem prosleđuje podatke ili podatke dobijene od korisnika upisuje u bazu koristeći sequelize
- views-Sadrži viewove,kontroler za određenu rutu prikazuje view u većini slučajeva viewu prosleđuje i neke podatke.Viewovi su rađeni u pug template enginu.
- migrations – sadrži fajlove koji izvršavaju migracije tj. rade određene promene u bazi, za svaku izmenu u bazi podataka je kreirana migracija koja će tu promenu izvršiti nad bazom
- seeders-Sadrži podatke koje treba inicijalno ubaciti u bazu komandom `npx sequelize-cli db:seed:all`, posle pokretanja migracija

3.1 Serverski deo

Na serverskoj strani se koristi Node JS sa express frameworkom i sequelizom.Kao template engine se koristi pug.
Pug omogućava izradu dinamičkih stranica.
Sequelize je ORM. ORM je skraćenica za objektno orijenisano mapiranje. ORM omogućava rad sa tabelama kao objektima. Svaka tabela u bazi je predstavljena kao model (kao klasa), polja te klase su zapravo atributi u tabeli.Red u tabeli će biti instanca određene klase.Relacije između tabela su predstavljene kao polja objekta.Rezervacija ima sledeća polja i relacije.

```
Rezervacija.init({
    polazak_id: DataTypes.INTEGER,//ID polaska za koji je vezana rezervacija
    korisnik_id: DataTypes.INTEGER,//id korisnika koji je izvrsio rezervaciju
    pocetna_destinacija_id:DataTypes.INTEGER,//id pocetne destinacije rezervisanog polaska
    krajnja_destinacija_id:DataTypes.INTEGER,//id krajnje destinacije rezervisano polaska
    platio:DataTypes.BOOLEAN,//da li je korisnik platio kartu ili treba da je plati pri
polasku

  },
 static associate(models) {
      this.belongsTo(models.Destinacija,{//relacija sa tabelom destinacije(pocetna
destinacija)
        foreignKey:'pocetna_destinacija_id',
        as:'pocetna_destinacija'
```

```
      });
      this.belongsTo(models.Destinacija,{//relacija sa tabelom destinacije(krajnja
destinacija)
        foreignKey:'krajnja_destinacija_id',
        as:'krajnja_destinacija'
      });
      this.belongsTo(models.Korisnik,{//relacija sa tabelom korisnik
        foreignKey:'korisnik_id',
        as:'korisnik'
      });
      this.belongsTo(models.Polazak,{//relacija sa tabelom polasci
        foreignKey:'polazak_id',
        as:'polazak'
      });
      this.hasMany(models.RezervisanoSediste,{//relacija sa tabelom rezervisana_sedista
        foreignKey:'rezervacija_id',
        as:'rezervisana_sedista'
      });
    ...
    }
```

3.2 Klijentski deo

Na klijentskom delu je korišćen HTML,CSS(kompajlovan iz SCSSa) i Bootstrap framework.Bootstrap framework je dizajniran sa namerom da omogući brži razvnoj dinamičkih veb sajtova. Bootstrap sadrži već gotove komponete kao što su kartice ,dugmići, navigacioni meniji itd. Ovo značajno ubrzava razvoj web sajtova. Takođe boostrap podržava responsivni dizajn i omogućava laku izradu veb sajtova koji se mogu prilagoditi različitim veličinama ekrana.

3.3 Baza podataka

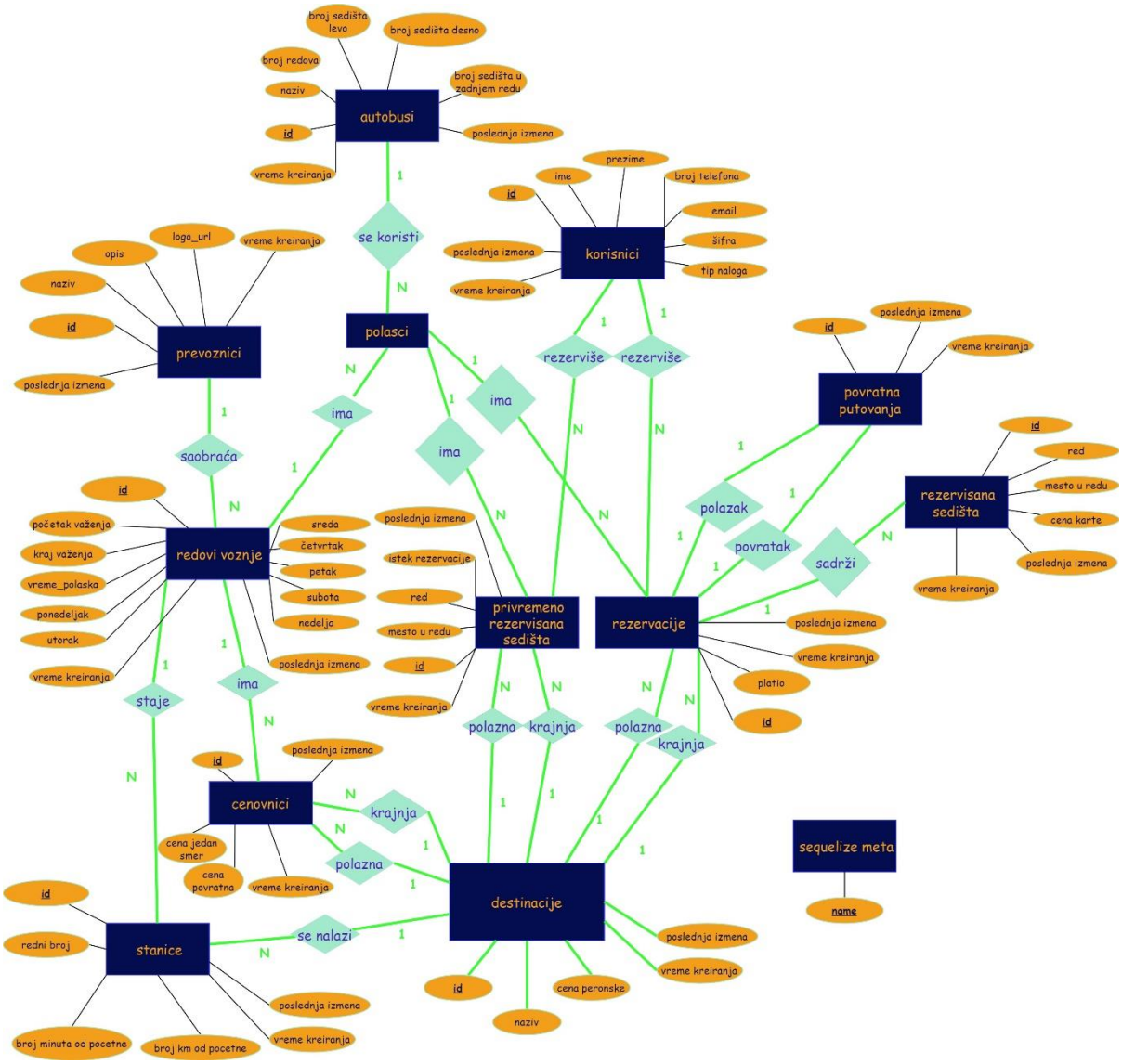
Za čuvanje podataka korišnjenja je MySql baza podatak kojoj se pristupa preko Sequeliza.Baza u kojoj se čuvaju podaci se lako može zamenititi pošto Sequelize podržava i SQLite,Postrges i Microsoft SQL server.Pri tome potrebno je izmeniti samo neke sirove (raw) upite,dok ostale/većinu upita koji su kreirani pomoću Sequliza nije potrebno menjati.Baza se sastoji od 13 tabela:

3.3.1 Opis tabela u bazi podataka

- korisnici – sadrži informacije o korisnicima
- autobusi – sadrži informacije o različitim tipovima autobusa,autobusi se razlikuju po broju redova i broju sedišta u redovima pa ova tabela sadrži te informacije za svaki autobus
- stanice – sadrži informacije o stanicama na kojima pojedini autobusi staju i cenu peronske karte za tu stanicu
- prevoznici – sadrži informacije o prevoznicima koji koriste ovaj sistem
- redovi_voznje – sadrži informacije o redovima vožnje

- destinacije – sadrži informacije o stanicama na koju autobusi koji idu određenom rutom staju
- cenovnici – sadrži cenu karte i cenu povratne karte između 2 destinacije za određeni red vožnje
- polasci – sadrži satnicu svakog polaska za određenu rutu (red vožnje)
- rezervacije – sadži informacije o rezervacijama
- rezervisana_sedista – informacije o rezervisanim sedištima za određenu rezervaciju
- privremeno_rezervisana_sedista – sadrži sedišta koja su rezervisana,ali rezervacija još uvek nije završena tj. rezervacija je još uvek u toku,zbog toga će ova sedišta biti rezervisana samo određeni (kratki) vremenski period. Kada se rezervacija završi ova sedišta će biti trajno rezervisana, a ako dođe do prekida rezervacije tj rezervacija se ne završi u određenom periodu ona više neće biti rezervisana
- povratna_putovanja – trenutno nema neku upotrebu,ali se u njoj čuvaju informacije o tome koja rezervacija je povezana sa drugom rezevacijom kao povratnim putovanjem
- sequelizemeta – tabela je automaski kreirana od strane Sequeliza i onda pamti informacije o izvršenim migracijama baze podataka

3.3.2 ER dijagram



** u prilogu se nalazi master slike (koji je moguće uveličati)*

3.4 Komunikacija

Tipovi naloga: N-Neprijavljen,U-Prijavljen korisnik,K-kondukter,P-Prodavac,A-admin
Pravo pristupa: D-ima pravo pristupa,N-nema pravo pristupa,R-samo ako je on vlasnik resursa

Adresa rute	HTTP Metod	POST Parametetri	OPIS	PRAVO PRISTUPA				
				N	U	K	P	A
/			Početna strana/strana za pretragu reda vožnje	D	D	D	D	D
/prijava	GET		Prikazuje stranicu za prijavu korisnika	D	N	N	N	N
/prijava	POST	email,lozinka,zahtevan_url	Vrši prijavu korisinika na osnovu unetih podataka	D	N	N	N	N
/registracija	GET		Prikazuje stranicu za	D	N	N	N	N

Adresa rute	HTTP Metod	POST Parametetri	OPIS	PRAVO PRISTUPA				
				N	U	K	P	A
			registraciju					
/registracija	POST	ime,prezime,broj_telefona,email,sifra	Registruje korisnika sa unetim podacima	D	N	N	N	N
/odjava	POST		Odjavljuje prijavljenog korisnika	N	D	D	D	D
/redvoznje/pretraga	GET		Stranica za pretragu reda vožnje(ista kao početna)	D	D	D	D	D
/redvoznje/pretraga	POST	pocetna_lokacija, krajnja_lokacija,datum_polaska,broj_putnika	Za izabrane destinacije i datum prikazuje autobuse koji saobrađaju tom linijom.	D	D	D	D	D
/rezervacija/prikaz/:red_voznje_id/:pocetna_destinacija/:krajnja_destinacija/:broj_putnika/:enkodovano_vreme/:tip_karte	GET		Prikazuje stranicu za rezervaciju autobuskih mesta za odabrane destinacije i datum	N	D	D	D	D
/termini/:pocetna_destinacija_id/:krajnja_destinacija_id/:enkodovan_datum/:prevoznik_id	GET		Ovo je asinhrona ruta.Njen zadatak je da vrati sve polaske (vremena polaska) za odabrane destinacije,datum i za određenog prevoznika.Koristi se prilikom biranja termina za rezervaciju povratne karte.	N	D	D	D	D
/rezervacija/autobus/:red_voznje_id/:broj_putnika/:pocetna_destinacija_id/:krajnja_destinacija_id/:datum_enkodovan/:vreme_enkodovano/:naziv_putovanja	GET		Ova ruta je asinhrona.Njen zadatak je da vrati HTML koji sadrži autobus.Koristi se prilikom rezervacije za crtanje autobusa za određeni polazak.	N	D	D	D	D
/rezervacija/kreiraj/:tip_putovanja	POST	nazivi_putovanja[],placanje,prvismer_pocetna_destinacija_id,prvi_smer_krajnja_destinacija_id,prvi_smer_polazak_id,prvi_smer_rezervacija,	Ova ruta se poziva prilikom kreiranja nove rezervacije.Njen zadatak je da kreira rezervaciju za izabrani termin polaska (i povratka) i izabrana sedišta.	N	D	D	D	D

Adresa rute	HTTP Metod	POST Parametetri	OPIS	PRAVO PRISTUPA				
				N	U	K	P	A
		opciono: povratak_pocetna_destinacija_id,povratak_krajnja_destinacija_id,povratak_polazak_id,povratak_rezervacija						
/rezervacija/rezervisanostsedista/:polazak_id/:pocetna_destinacija_id/:krajnja_destinacija_id/:red/:mesto_u_redu	GET		Asinhrona ruta,proverava da li je sedište koje je korisnik izabrao zauzeto.Može se dogoditi da tokom pravljenja rezervacije neki drugi korisnik odabere isto to sedište zbog toga je potrebno proveriti da li je sedište u međuvremenu zauzeto pre nego što ga korisnik rezerviše.	N	N	N	N	N
/rezervacija/privremenorezervisi/:polazak_id/:pocetna_destinacija_id/:krajnja_destinacija_id/:red/:mesto_u_redu/:vreme_isteka	POST		Asinhrona ruta.Kako bi se sprečilo da drugi korisnici rezevišu sedište koje je neki korisnik izabrao,a koji još uvek nije kompletirao rezervaciju,ova ruta će privremeno rezervisati to sedište (na 10 minuta) kako bi on imao vremena da kompletira rezervaciju bez bojazni da će neko drugi u međuvremenu to sedište rezervisati.	N	D	D	D	D
/rezervacija/obrisiprivremeno/:privremeno_sediste_id	POST		Za slučaj da korisnik promeni odluku o rezervaciji sedišta (deselektuje) izabrano sedište ova ruta se poziva i briše to sedište iz tabele privremeno rezervisanih sedišta	N	D	D	D	D
/rezervacija/moje	GET		Prikazuje sve rezervacije za prijavljenog korisnika	N	D	D	D	D
/rezervacija/prikazirezervisana/:rezervacija	GET		Prikazuje listu rezervisanih sedišta (karte) za određenu	N	R	D	N	D

Adresa rute	HTTP Metod	POST Parametetri	OPIS	PRAVO PRISTUPA				
				N	U	K	P	A
_id,			rezervaciju					
/rezervacija/prikazi/:id_karte/:sifra_karte	GET		Prikazuje samo pojedinačnu kartu na osnovu broja karte(IDija) i šifre karte(random broja koji je generisan kao šifra za oređenu kartu)	N	R	D	N	D
/rezervacija/citac	GET		Prikazuje stranicu za čitanje autobuske karte	N	N	D	N	D
/rezervacija/citac	POST	broj_karte,sifra_karte	Prikazuje kartu na osnovu IDija i šifre karte.	N	N	D	N	D
/rezervacija/ocitaj/:id_karte/:sifra_karte	POST		Vrši očitavanje autobuske karte.	N	N	D	N	D
/korisnik/kreiraj	GET		Prikazuje stranicu za kreiranje korisnika	N	N	N	N	D
/korisnik/kreiraj	POST	ime,prezime,broj_telefona,email,sifra,tip_naloga	Kreira novog korisnika,samo admin ima pristup.	N	N	N	N	D
/autobus/kreiraj	GET		Prikazuje stranicu za kreiranje novog autobusa	N	N	N	N	D
/autobus/kreiraj	POST	naziv,broj_sedista_levo,broj_sedista_desno,broj_redova,broj_sedista_u_zadnjem_redu	Na osnovu prosleđenih podataka kreira novi autobus.	N	N	N	N	D
/destinacija/kreiraj	GET		Prikazuje stranicu za kreiranje nove destinacije	N	N	N	N	D
/destinacija/kreiraj	POST	naziv,cena_peronske	Kreira novu destinaciju na osnovu prosleđenih podataka	N	N	N	N	D
/destinacija/lista	GET		Prikazuje sve kreirane destinacije	N	N	N	N	D
/destinacija/:id /izmeni	GET		Prikazuje stranicu za izmenu destinacije sa ID-ijem koje je prosleđen kao parametar	N	N	N	N	D
/destinacija/:id /izmeni	POST	naziv,cena_peronske	Čuva izmenjenu destinaciju	N	N	N	N	D
/prevoznik/kreiraj	GET		Prikazuje stranicu za dodavanje novog prevoznika	N	N	N	N	D
/prevoznik/kreiraj	POST	naziv,opis,logo	Kreira novog prevoznika	N	N	N	N	D
/prevoznik/lista	GET		Prikazuje sve prevoznike	N	N	N	N	D
/prevoznik/:id/izmeni	GET		Prikazuje stranicu za izmenu određenog prevoznika	N	N	N	N	D
/prevoznik/:id/izmeni	POST	naziv,opis,logo	Čuva nove podatke o	N	N	N	N	D

Adresa rute	HTTP Metod	POST Parametetri	OPIS	PRAVO PRISTUPA				
				N	U	K	P	A
			prevozniku					
/redvoznje/kreiraj	GET		Prikazuje stranicu za kreiranje reda vožnje	N	N	N	N	D
/redvoznje/kreiraj	POST	naziv,ponedeljak-nedelja,vreme_polaska,autobus,prevoznik,poceak_vazenja,rok_vazenja,stanice[],vreme[],kilometri[],karata[*pocetna_destinacija*][*krajnja_destinacija*][tip]	Kreira novi red vožnje,stanice za taj red vožnje,cenovnike za moguće rute za taj red vožnje,kreira sve polaske koji su u opsegu važenja reda vožnje	N	N	N	N	D
/redvoznje/lista	GET		Prikazuje sve kreirane redove vožnje	N	N	N	N	D
/redvoznje/:id/kopiraj	GET		Prikazuje stranicu za kopiranje reda vožnje.	N	N	N	N	D
/redvoznje/:id/kopiraj	POST	naziv,pocetak_vazenja,kraj_vazenja,vreme_polaska,autobus,obrni_smer	Kreira novi red vožnje koji ima drugacičiji naziv, rok važenja,autobus i vreme polaska i možda smer, ali staje na iste stanice i važe iste cene kao i za red vožnje čiji je kopija.	N	N	N	N	D

4. Rad aplikacije

4.1 Opis implementacije

4.1.1 Autentifikacije

Pošto aplikacija podržava registraciju potrebno je omogućiti i prijavu korisnicima. Kako korisnici ne bi morali da kucaju svoju loziku prilikom svake interakcije sa aplikacijom, potrebno je negde upamtiti koji korisnik je trenutno prijavljen na određenom računaru. To se radi pomoću sesije. Prilikom uspešne prijave korisnika na serverskoj strani se u sesiji pamti id prijavljenog korisnika i tip naloga (kako ne bi svaki put povlačili tip naloga korisnika iz baze na ovaj način je

"keširan" pošto se tip naloga korisnika gotovo nikada ne menja). Na klijenskoj strani je zapamćen ključ sesije koji se automatski šalje serveru prilikom svakog zahteva. Ovo omogućava express sesssion modul.

4.1.2 Način čuvanja korisničkih lozinki u bazi podataka

Čuvanje lozinke kao običan tekst u tabeli nije baš najbolje rešenje. U slučaju neovlašnjenog pristupa maliciozni korisnik će imati pristup lozinkama svih korisnika. Zbog toga svaka moderna aplikacija šifre korisnika čuva u hešovanom obliku. Heš funkcija je jednosmerna funkcija, iz heša nije moguće dirketno izvući šifru, ali je za svaku šifru moguće generisati heš. Postoje različiti alogiritmi za heširanje podataka, vremenom su nađeni propusti u nekim heš alogirmtima kao što je MD5, zbog toga je preporučeno da se on ne koristi. Za hešovanje šifri se koristi bcypt modul. On omogućava dodavanje "salta" na unetu šifru na taj način se bezbednost još više podiže zato što se dodavanjem "salta" sprečava "rainbow attack". Takođe bcyrpt omogućava da se šifra pre čuvanja u bazu hešira više puta uzastopno, to naznačajno usporava proces prijave korisnika, ali podiže bezbednost zato što bi u slučaju da haker dobije pristup bazi njem bilo potrebno više vremena da „pronađe" heš za određenu šifru.

4.1.3 Autorizacija

Aplikacija podržava više tipova korisnika: običnog korisnika, konduktera, prodavca, admina i vozača (ovaj tip naloga još uvek nema primenu). Potrebno je određenim korisnicima zabraniti pristup određenim delovima aplikacije. U aplikaciji su primenjanja dva tipa autorizacije:

- Autorizacija bazirana na tipu naloga – samo korisnik sa određenim tipom naloga ima pristup nekim delovima sistema
- Autorizacija bazirana na vlasništvu resursa – samo vlasnik resursa ima pristup određenom reserusu. Primer ove autorizacije je pregled karata za određenu rezervaciju ruta `/prikazirezervisana/:rezervacija_id` omogućava da pomoću IDija rezervacije korisnik vidi sedišta koja su u sklopu neke rezervacije,potrebno je sprečiti korisnike koji nisu napravili određenu rezervaciju da pregledaju tu rezervaciju. Za to se korisiti ova vrsta autoriacije.

4.1.3a Autoritzacija bazirana na tipu naloga

Ova autorizacija je urađena pomoću middlewarea.Postoje tri middlewarea koji se koriste za ovu svrhu,njihov kod je smešten unutar servisa autorizacijaService.

```
module.exports.autorizacijaZaTipNaloga=(...tip_naloga)=>
{
    return function(req,res,next)
    {
        let tipovi_naloga= Array.from(tip_naloga)
        if(typeof (req.session.korisnik_id)=='undefined' ||
req.session.korisnik_id==null)

        {
            var zahtevan_url=encodeURI(req.originalUrl);
```

```
            res.render('korisnik/prijava',{zahtevan_url});
        }
        else if(tipovi_naloga.includes(req.session.tip_naloga))
        {
            next();
        }
        else
        {
            res.render('403')
        }
    }
}
```

Ovaj middleware će omogućiti pristup samo prijavljenim korisnicima čiji tip naloga odgovara nekom od tipova naloga koji su prosleđeni kao parametar ovom middlewaru.U slučaju da korisnik nije prijavljen biće preusmeren na stranicu za prijavu,u slučaju da nema pristup biće mu prikazana stranica 403 (zabranjen pristup), a u slučaju da mu je pristup dozvoljen,njegov zahtev će biti prosleđen sledećem middlewareu.Pored ovog middlewara postoje još dva slična middlewarea:

- autorizacijaPrijavljen – blokira pristup neprijavljenim korisnicima
- autorizacijaOdjavljen – blokira pristup prijavljenim korisnicima

Primer upotrebe middlewarea:

```
router.post('/ocitaj/:id_karte/:sifra_karte',
autorizacijaZaTipNaloga(TipNaloga.admin,TipNaloga.kondukter)
,rezervacijaController.ocitajKartu);
```

Ruta za očitavanje karte je zaštićena i samo admin i kondukter mogu da očitaju kartu.Ako je korisnik admin ili kondukter zahtev će biti prosleđen kontroleru zaduženom za ovu rutu.

4.1.3b Autorizacija bazirana na vlasništvu resursa

Kao što je već rečeno nekada je potrebno dozvoliti samo vlasniku nekog resursa pristup tom resursu u ovom slučaju jedina implementacija takve vrste autorizacije je prilikom pregleda detalja o rezervaciji tj. prlikom pregleda karata za određenu rezervaciju. Za ovaj tip autorizacije postoji metoda. Autorizacija na bazi vlasništva na primeru metode za prikaz rezervisanih sedišta:

```
 module.exports.prikaziRezervisanaSedista=async(req,res)=>
{
    var rezervacija_id=req.params.rezervacija_id;
    var rezervacija=await
rezervacijaService.nadjiRezervacijuSaSedistima(rezervacija_id)
    var vlasnik_resursa_id=rezervacija.korisnik_id;
    if(!imaPristupResursu(vlasnik_resursa_id,req.session.korisnik_id,req.session.
tip_naloga,TipNaloga.admin,TipNaloga.kondukter,TipNaloga.prodavac))
```

```
    {
        res.render('403');
        return;
    }
...
}
```

Prvo se iz baze podataka nalazi rezererervacija sa sve rezerzervisanim sedištima.Zatim se pomoću metode `imaPristupResursu` servisa "autorizacijaService" vrši provera da li korisnik ima pristup tom resursu(rezervaciji).On će imati pristup samo u slučaju da je on kreirao rezervaciju ili je admin ili kondukter,u ostalim slučajevima pritup će mu biti zabranjen.

```
module.exports.imaPristupResursu=(vlasnik_resursa_id,korisnik_id,korisnik_tip_nal
oga,...tip_naloga)=>
{
    let tipovi_naloga= Array.from(tip_naloga)
    if(korisnik_tip_naloga=='admin' ||
tipovi_naloga.includes(korisnik_tip_naloga) || korisnik_id==vlasnik_resursa_id)
    {
        return true;
    }
    else
    {
        return false;
    }
}
```

Nekada je potrebno da nekom resursu,pored vlasnika tog resursa,pristup omogućimo još nekim tipovima korisnika (npr. admin treba da ima pristup svim resursima). Zbog toga pored IDija korisnika kome je dozvoljen pristup ova metoda prima i tipove naloga koji takođe imaju pristup.Ova metoda proverava da li korisnik admin ili je tipa njegov tip naloga neki od tipova naloga kojima je dozvoljen pristup ili je korisnik vlasnik resursa,ako je neki od ova tri uslova ispunjen korisniku će biti dozvoljen pristup.U ostalim slučajevima korisniku neće biti dozvoljen pristup tj. metoda će vratiti false.

4.1.4 Preusmeravanje

Često se dešava da je potrebno korisnika,kada završi popunjavanje neke forme, preusmeriti na neku drugu stranicu. Često mu je potrebno prikazati i neku poruku prilikom tog preusmravanja,recimo da je podatak uspešno upisan u bazu ili da je forma pogrešno popunjena.Zbog toga je kreirana posebna Redirect biblioteka metoda u okviru helper foldera. Ova biblioteka sadrži metode za preusmeravanje korisnika sa porukom:

- Redirect.backWithValidationErrors(req,res,err)
    - Sequlize poseduje metode za validaciju podataka pre samog upisa u bazu(logika za validaciju je smeštena u okviru modela),u slučaju da su podaci u lošem formatu

tj. da nisu prošli validaciju sequlize će „baciti“ grešku ta greška se zatim „hvata“ i može se pozvati ova metoda. Ona će te greške “prepakovati” i prikazati ih korisniku. Glavni pug layout koji nasleđuju svi viewovi sadrži html elemente u kojima se može prikazi poruka o grešci ili o uspešnom izvršenju neke radnje.

- Redirect.backWithSuccess(req,res,msg)
  - Ova metoda će vratiti korisnika na prethodnu stranu i pri tome mu prikazati poruku o uspešnom izvršenju neke radnje (tekst poruke se prosleđuje kao treći parametar)
- Redirect.toRouteWithSuccess(req,res,route,err)
  - Redi isto što i prethodna metoda samo korisnika ne vraća na prethodnu stranicu neko na rutu koja je prosleđena kao treći parametar.
- Redirect.backWithError(req,res,msg)
  - Radi isto što i prethodna metoda samo korisniku prikazuje poruku o grešci.
- Redirect.toRouteWithError(req,res,route,err)
  - Redi isto što i prethodna metoda samo korisnika ne vraća na prethodnu stranicu neko na rutu koja je prosleđena kao treći parametar
- Redirect.backIfUndefinedOrEmpty=(req,res,...data)
  - Ova metoda se poziva i njoj se prosleđuju podaci koje je korisnik morao da pošalje serveru kako bi server ispunio neki zahtev,za slučaj da korisnik nije poslao sve potrebne podatke,biće vraćen na prethodnu stranicu i biće mu prikazana poruka da je potrebno da unese sve potrebne podatke

4.1.4a Prikaz prethodno unetih podataka prilikom greške

Često u slučaju da korisnik unese pogrešne podatke potrebno ga je vratiti na stranicu na kojoj se nalazio (što prethodne metode i rade),ali je potrebno korisniku prikazati i podatke koje je uneo, kako ne bi morao celu formu da popunjava od početka.To je urađeno na sledeći način. Napravljen je middleware koji presreće svaki zahtev koji korisnik uputi serveru,proverava da li je zahtev tipa “POST” za slučaj da jeste sve podatke koje je korisnik uneo smešta u korisnikovu sesiju `req.session.old_data=req.body`.Ovaj middleware zatim objektuu `res.locals` dodaje metodu `old` ,ova metoda kao parametar prima naziv parametra POST zahteva čiju je vrednost potrebno pronaći,a kao drugi opcioni parametar prima podrazumevanu vrednost za slučaj da staru vrednost ne pronađe.Ova metoda je smeštena u objekat `res.locals` zato što njemu ima pristup pug template engine. Primena metode old prilikom izmene prevoznika

```
input#naziv.form-control(type='text' name='naziv'
value=locals.old('naziv',naziv))
```

Za slučaj da stara vrednost parametra naziv ne postoji korisniku će biti prikazana vrednost ovog parametra iz baze.Pošto se vrši izmena podrazumeva se da je u bazi već upisan neki podatak.

4.1.5. Opis nekih Sequelize upita

```
var rezervacije=Rezervacija.findAll(
  {
    where:
    {
```

```
      korisnik_id:id_korisnika //od svih rezervacija prikazace samo one koje je
napravio odredjeni korisnik
    },
    attributes: {
      include: [
        [sequelize.fn('COUNT', sequelize.col('rezervisana_sedista.id')),
'broj_sedista']
        //pored svih atributa modela potrebno je prikazati i broj rezervisanih
sedista za svaku rezervaciju zbog toga korisntimo sequlize funkciju
        //kao prvi parametar se prosledjuje naziv funckije,kao drugi parametar
kolona nad kojom se funckija primenjuje,a kao treci parametar kako ce se zvati
izlaz iz funckije tj. naziv atributa/alijas
      ]
    },
    include://ukljucivanje asocijacija
    [
      {
          model:Destinacija,//pored informacija o rezervaciji zelimo da prikazemo
i naziv pocetne destinacije zbog toga moramo ukljucti asocijaciju Destinacija
          as:'pocetna_destinacija', //posto 2x ukljucajemo model Destinacija za
pocetnu i krajnju destinaciju moramo koristiti alijas koji smo specificirali u
modelu
      },
      {
          model:Destinacija,
          as:'krajnja_destinacija',
      },
      {
        model:RezervisanoSediste,
        as:'rezervisana_sedista',
      },
      {//zelimo da prikazemo i naziv prevoznika kod koga je izvrsena
rezervacija,ali da bi dosli do tog podataka potrebno je proci kroz vise
asocijacija (odnosno tabela u bazi)
          model:Polazak,
          as:'polazak',
          include:
          [
              {
                  model:RedVoznje,//nalazimo red voznje kojem pripada polazak
                  as:'red_voznje',
                  include:
                  [
                      {
```

```
                        model:Prevoznik,//posto red voznje sadrzi relaciju sa
prevoznikom tako dobijamo prevoznika
                        as:'prevoznik'
                    },
                ]
            }
        ]
    }
    ],
    group: ['Rezervacija.id']//grupisemo po idiju rezervacije(zbog prebrojavanja
broja sedista po rezervaciji)
  })
```

Sequlize će zatim zatim ovaj upit pretvoriti u SQL upit pošto je podešen za rad sa MySql bazom. Dobijene podatke od baze će zatim pretvoriti u niz JS objekata,asocijacije(relacije) će biti ugnježdene.Tako da ako je potrebno pristupiti recimo nazivu početne destinacije prve rezervacije to bi bilo urađeno na sledeći način `rezervacije[0].pocetna_destinacija.ime`

4.1.5.1 Sirovi upiti

Nekada je potrebno napisati sirov(raw/običan) SQL upit, bez korišnjenja sequlize ORMa.Razlozi za to su mnogobrojni,a neki od njih su

- Nekada je zbog brzine potrebno napraviti optimizovaniji upit od onog kojeg pravi sequlize
- Neki upiti bi bili previše kompleksni da su rađeni pomoću ORMa,pa ih je jednostavnije napisati kao običan SQL upit
- Programer ne zna da napravi određeni upit pomouću ORMa
- ORM može usporiti aplikaciju pošto vrši mapiranje dobijenih podataka od baze

Sirovi upiti u su ovaj aplikaciji mahom korišćeni za prikazivanje zauzetih sedišta u okviru `rezervacijaService/prikaziZauzetaSedista` metode. Bilo je mnogo praktičnije napisati sirove upite nego koristi ORM.Upiti su bili poprilično kompleksni,zahtevali su povezivanje dosta tabela sa više uslova, a izvlačenje relativno malog broja podataka.

4.1.5.2 Transakcije

Nekad je potrebno izvršiti više povezanih SQL upita, u slučaju da dođe do greške prilikom izvršenja jednog upita, potrebno je poništiti i prethodne (povezane) upite, to se naziva transakcija.Sequlize ima podršku za transakcije.Prilikom kreiranja novog reda vožnje potrebno je ubaciti nove podatke u više tabela. Prvo je potrebno kreirati red vožnje,zatim stanice na kojima autobus koji se kreće tom linijom staje, zatim cenovnike za taj red vožnje i polaske za njega.Ne želimo da dođemo u situaciju da se red vožnje kreira bez cenovnika ili nešto slično.Zbog toga su svi ovi upiti povezani u jednu transakciju.

```
try
   {
    const rezultat= await sequelize.transaction(async(t)=>
    {
```

```
        var redvoznje=await RedVoznje.create(red_voznje_model,{transaction: t})
  . . .
Logika za kreiranje stanica i cenovnika.Ona se izvršava tek posle dodavanja novog
reda vožnje pošto tada znamo njegov ID u bazi
  . . .
 . . .
            var stanice=await
Stanica.bulkCreate(stanica_modeli,{validate:true,transaction: t})
            var cenovnici=await
Cenovnik.bulkCreate(cenovnik_modeli,{validate:true,transaction: t})
            var
polasci_modeli=polazakService.kreirajPolaskeZaRedVoznje(redvoznje.id,autobus_id,v
reme_polaska,pocetak_vazenja,rok_vazenja,ponedeljak,utorak,sreda,cetvrtak,petak,s
ubota,nedelja)
            var polasci=await
Polazak.bulkCreate(polasci_modeli,{validate:true,transaction: t})
    })
   }

catch(err)
    {
        Redirect.backWithValidationErrors(req,res,err);
        return;
    }
```

U slučaju greške prilikom bilo kog od ovog upita sve što je ubačeno u bazu će biti obrisano i korisnik će biti vraćen na prethodnu stranicu i biće mu prikazana poruka o grešci.

4.1.5.3 Asinhroni http zahtevi

Asinhorni http zahtevi su zahtevi upućeni serveru koji se izvršavaju u pozadini bez osvežavanja stranice.Jedan od razloga za ovake zahteve je bolje korisničko iskustvo,pošto ne želimo da korisniku celu stranicu ponovo učitamo zbog nekog malog zahteva koji će izvršiti minimalnu promenu na stranici. U ovoj aplikaciji asinhroni zahtevi su implementirani prilikom rezevaracije mesta.Logika za crtanje autobusa je smeštena na serveru.Kada se korisniku učita stranica za kreiranje rezervacije, asinhrono će biti upućen zahtev koji će od servera tražiti prikaz autobusa za određeni polazak i destinacije, server će mu zatim u HTML obliku vratiti deo stranice koji će sadržati autobus.Taj HTML kod će zatim biti dodat na stranicu.Takođe u slučaju da je korisnik odabrao povratno putovanje,on će imati mogućnost da odabere datum povratka,kada korisnik odabere datum,desiće se još jedan asinhroni zahtev,ali će ovaj put server u JSON formatu vratiti sve podatke (Slika 1).

```
▼[{red_voznje_id: 1, vreme_polaska_sa_prve_stanice: "08:00:00", vreme: "08:55"},…]
  ▼0: {red_voznje_id: 1, vreme_polaska_sa_prve_stanice: "08:00:00", vreme: "08:55"}
     red_voznje_id: 1
     vreme: "08:55"
     vreme_polaska_sa_prve_stanice: "08:00:00"
  ▶1: {red_voznje_id: 5, vreme_polaska_sa_prve_stanice: "12:00:00", vreme: "12:55"}
  ▶2: {red_voznje_id: 3, vreme_polaska_sa_prve_stanice: "14:00:00", vreme: "14:55"}
```

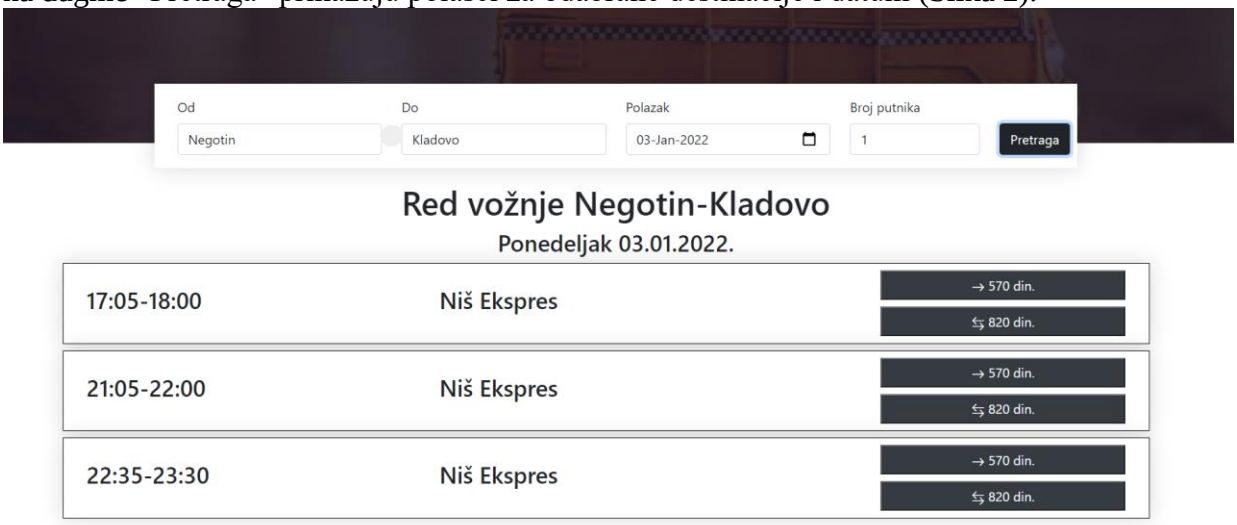

Slika 1

Korisnik će zatim imati mogućnost da odabere i vreme povratka.Prilikom odabira vremena povratka,opet će se desiti asinhroni zahtev.Ovaj zahtev će biti identičan onom prvom za crtanje autobusa,samo što će sada vratiti nacrtan autobus za drugi polazak. Zbog toga je i najviše i onaj prvi zahtev bio asinhron,kako se logika za crtanje autobusa ne bi duplirala.Rezultat ovog zahteva će opet biti HTML kod.

4.2 Opis funkcionalnosti – korisničko uputstvo

4.2.1 Pregled reda vožnje

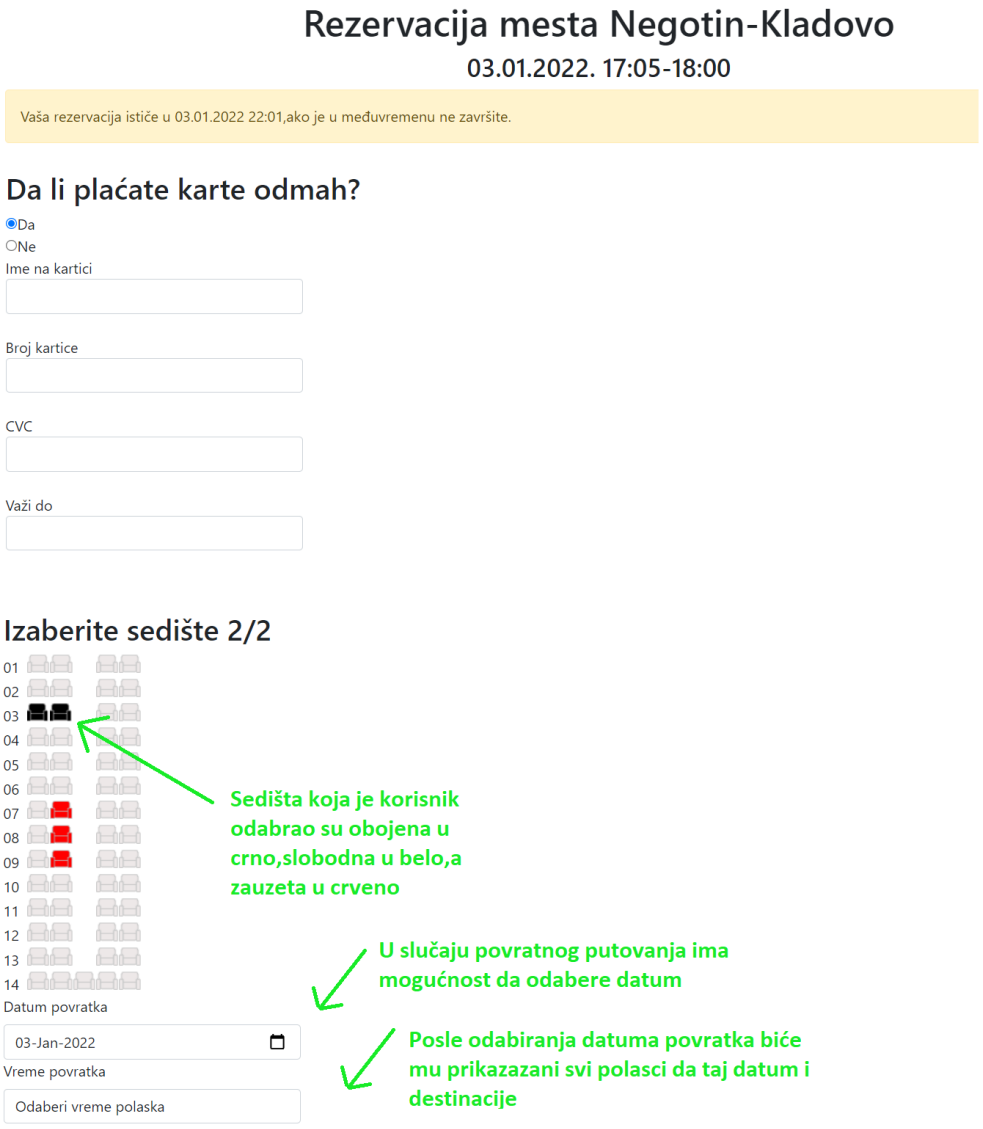
Korisnik na početnoj stranici ima mogućnost pregleda reda vožnje,u slučaju da je prijavljen ima i mogućnost rezervacije,u slučaju da nije, prilikom pokušaja rezervacije biće preusmeren na stranicu za prijavu.Korisnik bira početnu i krajnju destinaciju i broj putnika zatim mu se klikom na dugme “Pretraga” prikazuju polasci za odabrane destinacije i datum (Slika 2).



Slika 2

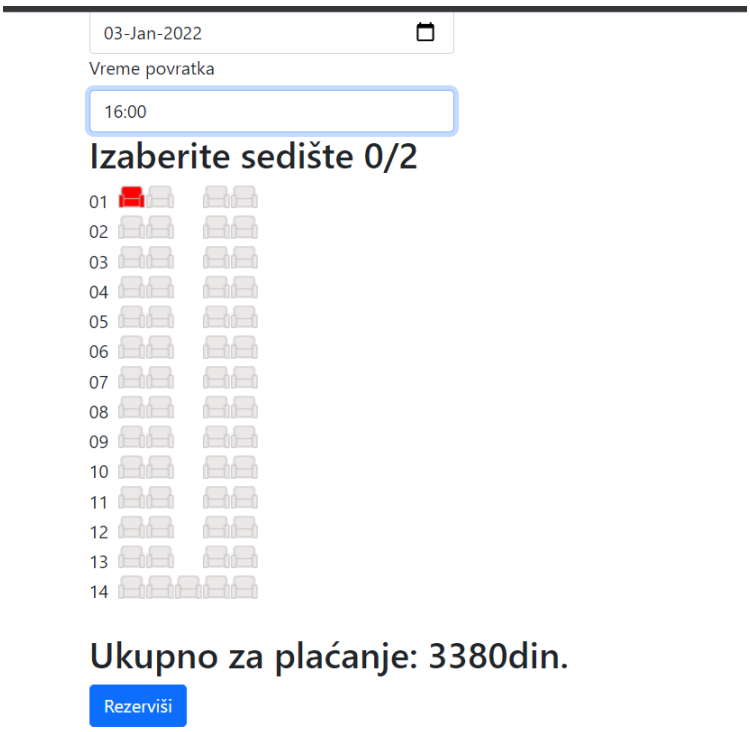
4.2.2 Rezervacija karata

Klikom na kartu za željeni polazak korisnik će biti preusmeren na stranicu za rezervaciju za odabran polazak. Na stranici za rezervaciju će imati mogućnost da izabere da li kartu plaća odmah (skidanje novca sa kartice još uvek nije implemntirano, samo je pokaznog karaktera). Zatim će imati mogućnost da odabere sedišta. U slučaju da je odabrao povratnu kartu imaće i mogućnost da izabere datum povratka, a zatim će mu biti prikazana satica svih povrataka za odabrani datum,ali samo za prevoznika kojim putuje u prvom smeru. (Slika 4)



Slika 4

Posle biranja datuma i vremena povratka korisniku će se pojaviti autobus za taj povratak.Na slici autobusa će biti označena zauzeta mesta i korisnik će imati mogućnost da odabere neka od dostupnih mesta.Takođe će mu biti prikazana i cifra koju treba da plati (Slika 5).Prodavci karata (šalterski radnici) imaju istu ovu mogućnost samo što oni nemaju opciju da biraju da li se karta odma plaća ili kasnije,podrazumeva se da korisnik koji kupuje kartu odmah šalterskom službeniku daje novac za kupljenu kartu tj. da je karta odmah plaćena.



Slika 5

4.2.3 Pregled rezervacija

Korisnik takođe u navigacionom meniju ima opciju za pregled svojih rezervacija.Klikom na opciju za prikaz,biće mu prikazane sve njegove rezervacije. (Slika 6).

#	Relacija	Datum polaska	Prevoznik	Broj karata	Plaćeno	Akcija
1	Negotin-Zaječar	03.01.2022.	Niš Ekspres	2	Ne	Detalji
2	Zaječar-Negotin	03.01.2022.	Niš Ekspres	1	Ne	Detalji
3	Negotin-Zaječar	03.01.2022.	Niš Ekspres	2	Ne	Detalji
4	Zaječar-Negotin	03.01.2022.	Niš Ekspres	2	Ne	Detalji

Slika 6

Klikom na dugme “Detalji” biće mu prikazani detalji za odabranu rezervaciju.Odnosno biće mu prikazane karte koje će moći da odštampa(Slika 7). Klikom na QR kod ili skeniranjem QR koda karte biće preusmeren na stranicu koja će prikazati informacije samo za tu kartu.

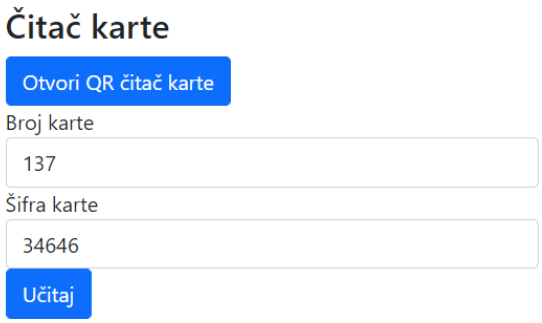
Slika 7

Prodavci karata(šalterski radnici) pored liste svih rezervacija koje su napravili za druge korisnike vide i statistiku o prodaji karata(Slika 8).

Slika 8

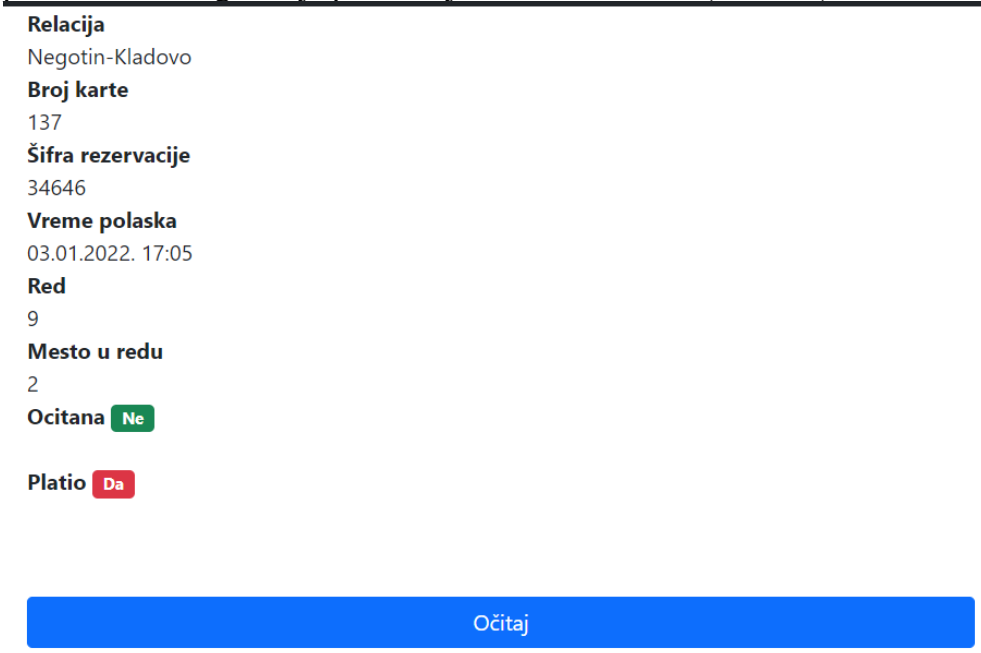
4.2.4 Očitavanje karata

Kondukter ima mogućnost da očita kartu.Čitač karte otvara iz navigacionog menija.
Čitač karte sadrži dugme za otvaranje QR čitača kartice,na android telefonu će ovo pokrenuti eksternu aplikaciju (napravljenu od strane treće strane) koja ima mogućnost čitanja QR koda.
Svaka karta sadrži QR kod,QR kod sadrži link ka ruti koja prikazuje podatke o karti.U slučaju da je korisnik prijavljen na sistem kao kondukter on će imati mogućnost da očita tu kartu.Kondukter može i ručno uneti broj karte i šifru karte i na taj način prikazati detalje za kartu. (Slika 9).



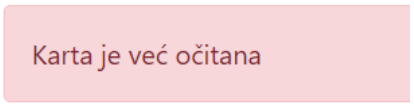
Slika 9

Kada kondukter očita kartu on će videti detalje za tu kartu i imati mogućnost da je očita (digitalno "pocepa") i na taj način spreči još nekog korisnika da mu pokaže istu kartu,pošto će prilikom sledećeg čitanja pisati da je karta već očitana.(Slika 10)



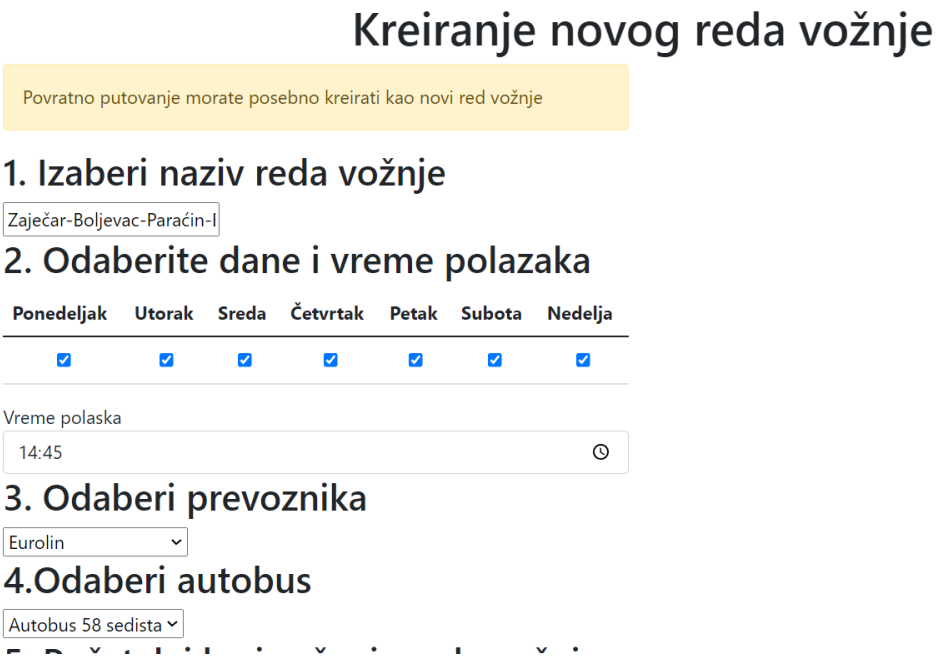
Slika 10

Sledeći put kada pokuša da očita kartu sa istim brojem biće mu prikazana poruka da je karta očitana umesto dugmeta za očitavanje karte(Slika 11).



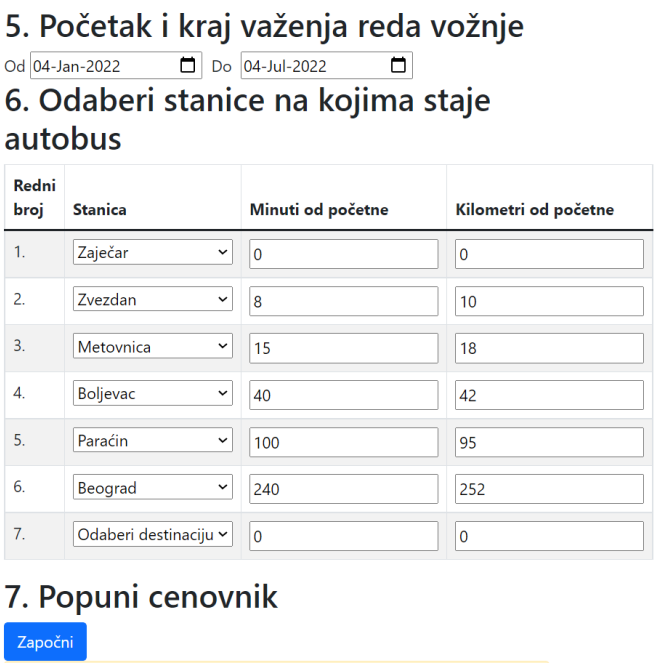
Slika 11

4.2.5 Kreiranje novog reda vožnje



Admin ima pristup svim funkcionalnostima aplikacije,jedna od funckcionalnosti je i kreiranje novog reda vožnje/rute.Kreiranju novog reda vožnje se pristupa iz navigacionog menija. Admin daje naziv redu vožnje bira datum i vreme polaska i tip autobusa (Slika 12)

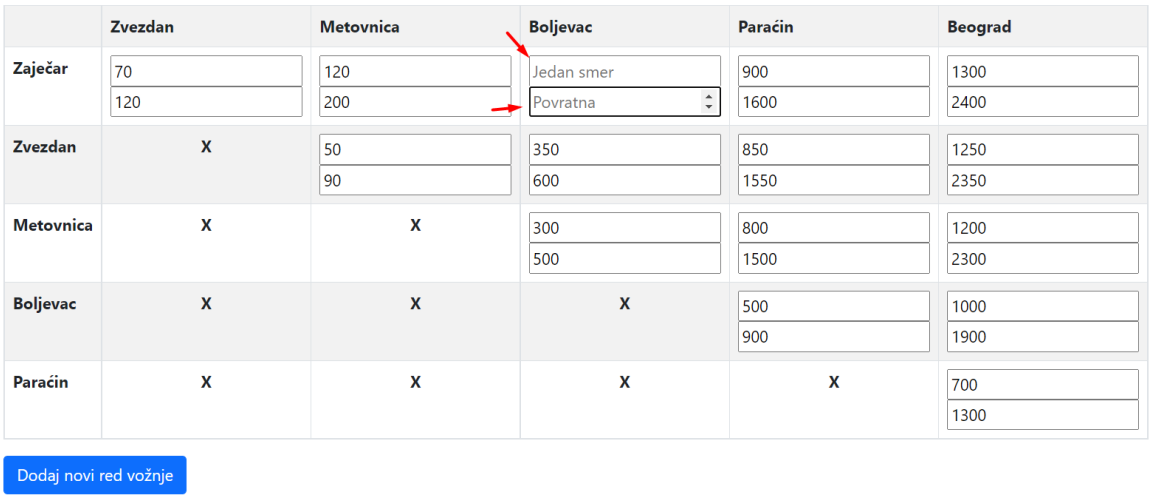
Slika 12



Slika 13

Zatim određuje trajanje reda vožnje. Posle toga bira stanice na kojima autobus staje, upisuje udaljenost i vremensku udaljenost svake stanice od početne stanice(Slika 13). Posle se klikom na dugme “Započni” iscrtava tabela za prikazivanje cenovnika.

Admin unosi cenu karte u jednom smeru i cenu povratne karte za svaku moguću “podrutu” (slika 14)



Slika 14

Klikom na dugme „Dodaj novi red vožnje “ admin dodaje novi red vožnje u bazu,pored samog reda vožnje u bazu će biti upisane i sve stanice na kojima autobus staje,cenovnici i biće generisani polasci koji su u okviru važenja reda vožnje.

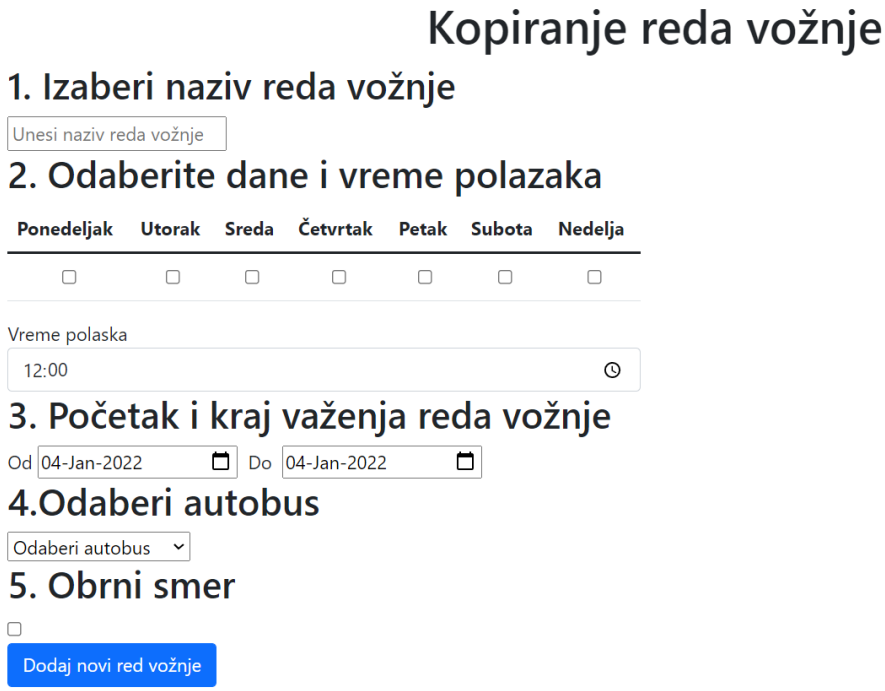
4.2.6 Kopiranje reda vožnje

Nekada je potrebno kreirati sličan red vožnje tj. red vožnje koji će se malo razlikovati od nekog: po smeru,roku važenja ili po danima i satnici polazka. Zbog toga je adminima omogućeno da kopiraju željeni red vožnje.To mogu uraditi klikom na “Redovi vožnje” u navigacionom baru. Time će prikazati listu svih redova vožnje. (Slika 15)

#	Naziv	Rok važenja	Dani polaska	Vreme polaska sa prve	Prevoznik	Akcija
1	Kladovo-Nis Nis Ekspres 8h svaki dan	18.12.2021.-01.05.2022.	ponedeljak,utorak,sreda,četvrtak,petak,subota,nedelja	08:00:00	Niš Ekspres	Kopiraj
2	Nis-Kladovo 12h Nis ekspress svaki dan	18.12.2021.-31.05.2022.	ponedeljak,utorak,sreda,četvrtak,petak,subota,nedelja	13:30:00	Niš Ekspres	Kopiraj
3	Kladovo Nis-Nis ekspress svaki dan 14h	18.12.2021.-30.05.2022.	ponedeljak,utorak,sreda,četvrtak,petak,subota,nedelja	14:00:00	Niš Ekspres	Kopiraj
4	Nis-Kladovo Nis ekspress svaki dan 19h	18.12.2021.-30.05.2022.	ponedeljak,utorak,sreda,četvrtak,petak,subota,nedelja	19:00:00	Niš Ekspres	Kopiraj

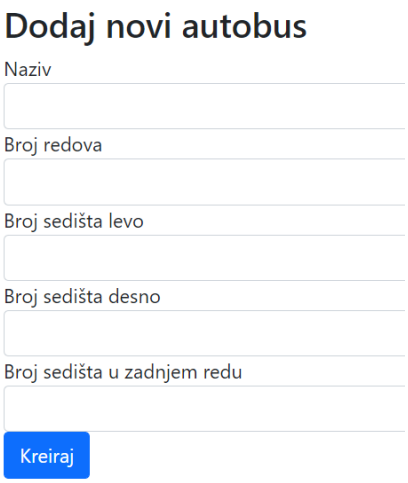
Slika 15

Klikom na dugme “Kopiraj” adminu će se otvoriti dijalog za kopiranje reda vožnje.U dijalogu će imati mogućnost da odabere druge dane i satnicu polazaka,da obrne smer putovanja i da promeni autobus kojim se putuje (Slika 16)



Slika 16

4.2.7 Dodavanje novih autobusa



Admin ima mogućnost dodavanja novog autobusa.Klikom na dugme “Dodaj autobus” u navigacionom meniju(Slika 17). Bira naziv, broj redova, broj sedišta sa leve strane u jednom redu i broj sedišta sa desne strane u jednom redu.Pošto se broj sedišta u zadnjem redu u većini autobusa razlikuje u odnosu na broj sedišta u prethodnim redovima on to unosi kao poseban podatak.Pored toga admin ima mogućnost dodavanja novih prevoznika i destinacija na sličan način.Admin takođe ima i mogućnost kreiranja novih korisnika konduktera i prodavaca.

Slika 17

5. Literatura

https://momentjs.com/docs/
https://sequelize.org/
http://test.8bit.rs/Images/seats.png - Slike sedišta koje koristi bioskop Vilin grad
https://stackoverflow.com/