

Platforma za rezervaciju voznih karata

Predmet: Klijent Server Sistemi

Predmetni profesor:
Dr Mirko Kosanović

Asistent:
Miloš Kosanović

Student:
Jovan Nedeljković Rer 60/17

Datum:
30.12.2019. godine

SADRŽAJ

1. Uvod	3
2. Instalacija i podešavanje projekta.....	3
2.1.1 O NodeJS-u	3
2.1.2 Instalacija NodeJS-a	4
2.1.3 O MongoDB	4
2.1.4 Instalacija MongoDB Compass aplikacije	5
3. Pokretanje aplikacije „Platforma za rezervaciju voznih karata“	6
4. Rad aplikacije	9
5. Editovanje baze promenom podataka u aplikaciji i obratno.....	15
6. Literatura.....	17

1. Uvod

U ovom projektu izrađena je aplikacija u vidu Platforme za online rezervaciju vozne karte. Na klijentskoj strani, od tehnologija su korišćeni HTML5 (koji se sastoji od HTML-a, CSS-a i JavaScripta-a) i Bootstrap framework, dok su na serverskoj strani korišćeni NodeJS sa Express Framework-om i MongoDB (NoSQL baza podataka).

Alati koji su korišćeni prilikom izrade aplikacije su Opera pretraživač (za grafički prikaz aplikacije) i Visual Studio Code (za pisanje aplikacije).

Aplikacija sadrži nekoliko povezanih stranica (template-a). Na početnoj stranici nalazi se tekst sa kratkim objašnjenjem korisniku o kakvoj se aplikaciji radi i dva dugmeta (Buttons), preko kojih korisnik može da se registruje u okviru aplikacije i nakon toga pristupi rezervaciji karte po želji.

Nakon klika na dugme Prijava na početnoj stranici aplikacije (<http://localhost:3000>), otvara se nova stranica (template), gde korisnik može videti tabelu u kojoj se nalaze podaci o određenom vozu (Broj voza, Destinacija voza, Datum i vreme polaska voza, kao i Broj slobodnih mesta za taj voz). Za svaki voz postoji mogućnost online rezervacije vozne karte klikom na dugme Rezerviši.

Nakon što korisnik rezerviše kartu, otvoriće se nova stranica sa obaveštenjem korisniku da je uspešno rezervisao kartu, kao i dugmetom za povratak na stranicu sa tabelom vozova. Rezervisanjem karte, automatski će se smanjiti broj slobodnih mesta za 1 u vozu za koji je korisnik rezervisao kartu, što će biti i prikazano u tabeli u koloni Broj slobodnih mesta, nakon što se korisnik vrati na stranicu za naručivanje karte.

2. Instalacija i podešavanje projekta

Da bi pokrenuli aplikaciju Platforma za rezervaciju karata, potrebno nam je da instaliramo NodeJS i MongoDB Compass i da na računaru imamo neki internet pretraživač i konzolu za pokretanje aplikacije (Command Prompt)

2.1.1 O NodeJS-u

Node.js je radno okruženje JavaScript programskog jezika koje se koristi na strani web servera. JavaScript se tradicionalno koristi na strani klijenta – tzv. front-end razvoj, a Node je omogućio da se JavaScript koristi i za back-end razvoj odnosno serversko izvršavanje koda.

Node.js ima veliki broj modula – JavaScript biblioteka ili funkcija koje obavljaju određene specifične zadatke i koje možemo pridodati našoj aplikaciji. Postoje neki ugrađeni moduli, dok su drugi dostupni pre svega putem NPM repozitorijuma – **N**ode **P**ackage **M**anager. NPM nije ograničen samo na Node.js – sada je to univerzalni JavaScript repozitorijum softvera i dostupni su moduli i za back-end i za front-end razvoj, React, Angular, Vue.js itd.

2.1.2 Instalacija NodeJS-a

NodeJS Framework možemo instalirati kao klasičnu Windows aplikaciju. Potrebno je da preko internet pretraživača odemo na sledeću stranicu:

<https://nodejs.org/en/download/>

Tu možemo izabrati da li ćemo koristiti poslednju stabilnu verziju (LTE recommended for most users ili Current latest features). Preporuka je da se selektuje LTE i u zavisnosti od verzije našeg operativnog sistema izabere datoteta za preuzimanje i instalaciju nakon toga. Izabraćemo dugme Windows installer kliknuvši na njega, nakon čega ćemo preuzeti instalacionu datoteku NodeJS-a (node-v12.14.0-x64).

Instalaciju pokrećemo dvoklikom na preuzetu datoteku, nakon čega sledimo uputstva na wizardu za instalaciju. Ono što je bitno je da u toku instalacije izaberemo sve komponente aplikacije, posebno opciju Add to PATH.

Nakon što instaliramo NodeJS, potrebno je instalirati dodatne NodeJS module.

Express je Node.js web framework – na srpskom bi se moglo reći da to znači Node.js okvir. Koristi se za programiranje web aplikacija. Express olakšava i ubrzava mnoge radnje i spašava nas pisanja suvišnih linija koda prilikom kreiranja Node.js aplikacija.

Da bi instalirali Express, potrebno je prvo otvoriti Command Prompt i navesti putanju do foldera u kome se nalazi naša aplikacija. Pošto naša aplikacija već poseduje *package.json* fajl, koji sadrži određene informacije u vezi sa našom aplikacijom, kao i o NPM modulima koje koristimo u ovoj aplikaciji, preskočićemo komandu `npm init`.

Kako funkcionalnost naše aplikacije zavisi od modula koje koristimo prilikom razvoja iste, te informacije je potrebno negde pohraniti. Zbog čega? Pre svega da bismo sami lakše pratili module koje koristimo, ali i da bi drugi programeri sa kojima delimo kod mogli da bez napora utvrde šta je korišćeno od modula u aplikaciji. Takođe, ako bismo želeli da našu Node.js aplikaciju postavimo javno na internet, ovaj fajl će biti potreban. Naime, vrlo verovatno je da će platforma koja bi trebalo da hostuje aplikaciju (npr. Heroku) upravo iz package.json fajla pročitati informacije o tome koje module bi trebalo instalirati zajedno sa aplikacijom kako bi ona funkcionisala.

Express je verovatno najpopularniji Node.js modul, ali nije preinstaliran, pa ga moramo preuzeti i instalirati iz NPM softverskog skladišta. To ćemo učiniti uz pomoć komande:

`npm install express --save`.

```
PS C:\Users\██████\Documents\node> npm install express --save
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN node@1.0.0 No description
npm WARN node@1.0.0 No repository field.

+ express@4.16.3
added 49 packages in 12.473s
```

Dakle, upravo smo napisali da želimo da Express bude instaliran i sačuvan u *package.json* fajlu kao modul od kojeg zavisi rad naše aplikacije.

2.1.3 O MongoDB

MongoDB je vodeća NoSQL baza podataka, ne koristi SQL za povezivanje, nerelaciona je, distribuirana, otvorenog koda i horizontalno skalabilna. Napisana je u C++ jeziku (kao i NodeJS) i otvorenog je koda.

MongoDB čuva podatke kao JSON dokumente sa dinamičkim šemama. JSON (*JavaScript Object Notation*) je otvoreni standard zasnovan na tekstu, osmišljen za razmenu podataka koji su pogodni za čitanje ljudima. MongoDB je prihvaćen kao backend softver brojnih značajnih web-sajtova i servisa.

MongoDB je baza podataka za opštu upotrebu. Postoji mnogo projekata koji danas koriste MongoDB. Njena dinamička šema i objektno-orijentisana struktura, čine je pravim izborom za analitiku u realnom vremenu, kao i za e-komerc, mobilne aplikacije, arhiviranje i slično. Poznati slučajevi korišćenja MongoDB obuhvataju “*big data*” podatke, upravljanje sadržajem, mobilnu i društvenu infrastrukturu i mnoge druge. Takođe, pojavljuju se izazovi za korišćenje MongoDB za *Business Intelligence modele*.

2.1.4 Instalacija MongoDB Compass aplikacije

MongoDB Compass aplikacija koja je pruža najlakši način da istražimo i upravljamo našom Mongo bazom podataka. Instalacija je vrlo jednostavna. Na stranici:

<https://www.mongodb.com/download-center/community>

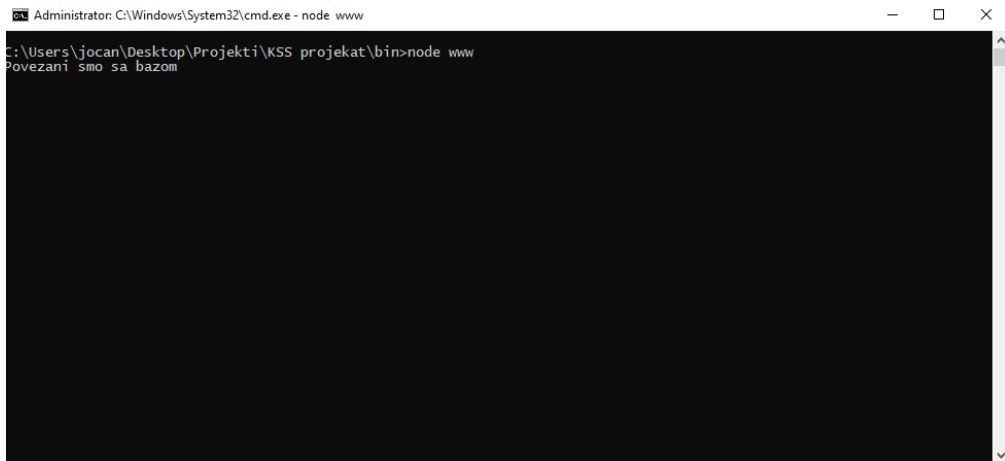
potrebno je izabrati platformu (Windows u našem slučaju) i preuzeti aplikaciju klikom na dugme Download. Nakon što preuzmemo aplikaciju, potrebno je pokrenuti instalaciju iste dvoklikom na preuzeti fajl. U toku instalacije izabrati Full install.

Sada imamo sve što je potrebno da bi pokrenuli našu aplikaciju „*Platforma za rezervaciju voznih karata*“.

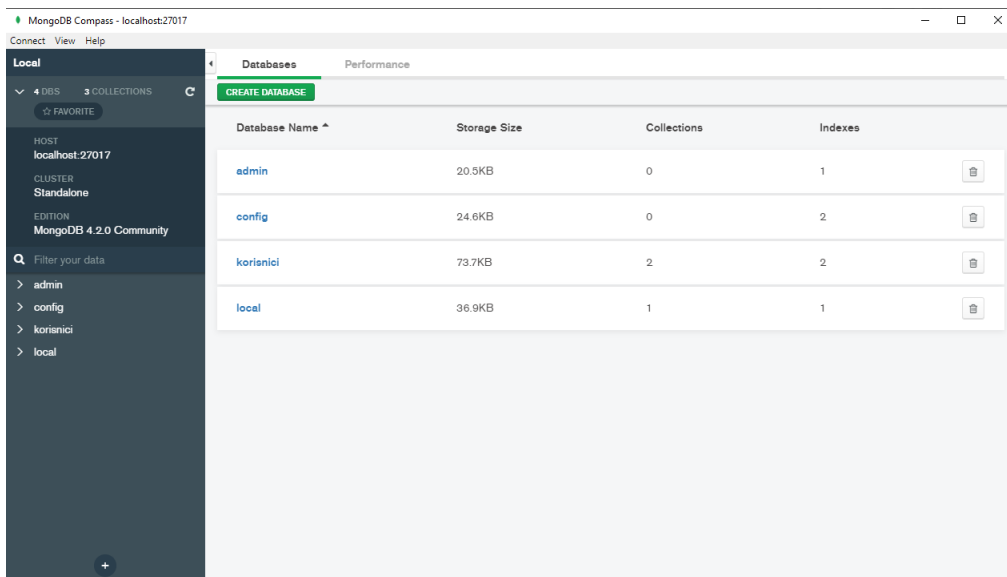
3. Pokretanje aplikacije „Platforma za rezervaciju voznih karata“

Prvo što treba da uradimo je da pokrenemo Command Prompt i navedeno putanju do foldera gde se nalazi naša aplikacija. Nakon toga kucati **npm start**. Ovime smo pokrenuli našu aplikaciju (slika 2).

Drugo što treba da uradimo je da pokrenemo MongoDB Compass aplikaciju. Nakon što se aplikacija pokrene, dovoljno je kliknuti na Connect. Ako je sve urađeno kako treba, treba da dobijete ovakve rezultate (slika 3):



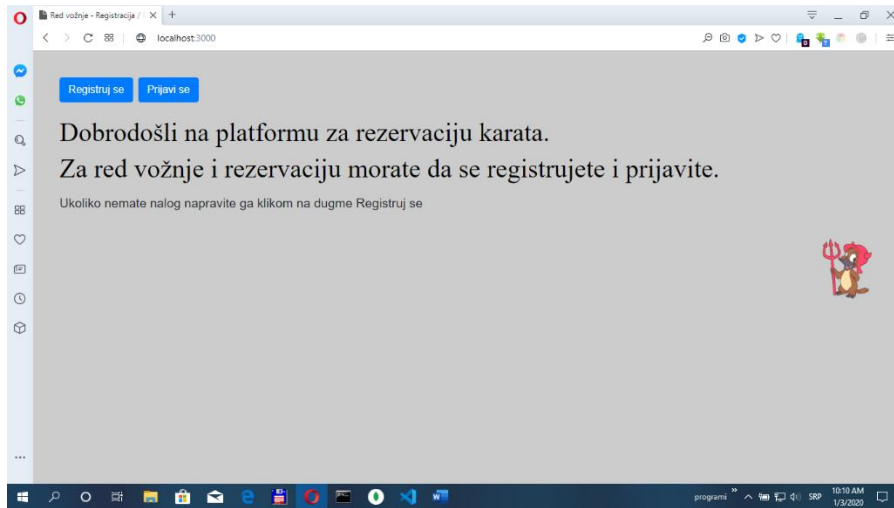
Slika 2



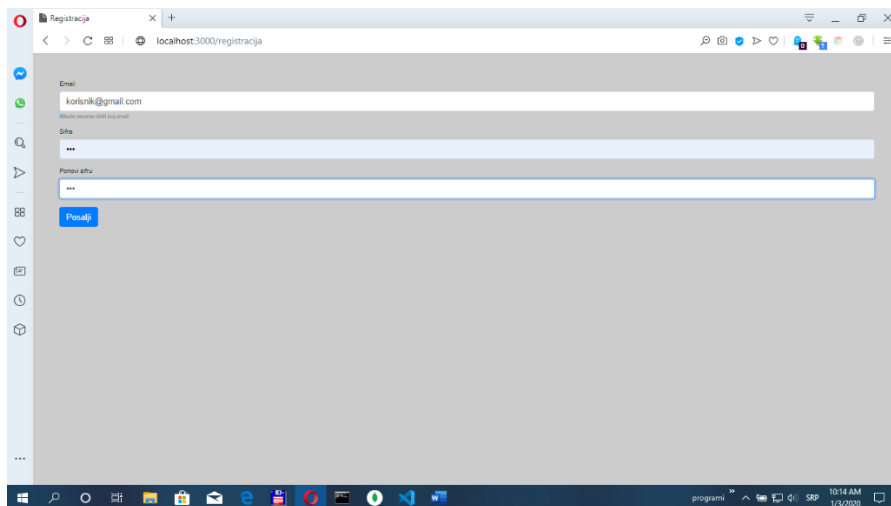
Slika 3

Sada je naša aplikacija pokrenuta na localhost-u na portu 3000.

Da bi videli kako aplikacija radi, potrebno je u pretraživaču kucati **localhost:3000**. Otvoriće se prozor kao na slici 4:



Tu možemo da se registrujemo da bi mogli da online rezervišemo kartu. Tako da prvo treba kliknuti na dugme Registruj se, nakon čega se otvara stranica kao na slici 5:



Nakon što upišemo svoj email, zatim šifru, pa ponovimo upis šifre, kliknućemo na dugme Pošalji. Ovime smo se registrovali i možemo pristupiti aplikaciji. Nakon što kliknemo na dugme Pošalji, naši korisnički podaci biće upisani u MongoDB bazi koja radi u pozadini sa našom aplikacijom i vратиćemo se na početnu stranicu gde ćemo kada kliknemo na dugme Prijavi se, dobiti stranicu na kojoj ćemo upisati malopredašnje podatke koje smo uneli za registraciju i klikom na dugme Pošalji, otvoriće nam se glavni prozor aplikacije gde možemo rezervisati kartu (slika 6):

Red vožnje

Broj voza	Destinacija	Datum polaska	Vreme polaska	Broj slobodnih mesta	
790	Beograd	23.12.2019	06:40	14	Rezerviši
2902	Beograd	07.12.2019	07:30	27	Rezerviši
2904	Beograd	07.12.2019	12:10	30	Rezerviši
2906	Beograd	07.12.2019	15:50	29	Rezerviši
792	Beograd	07.12.2019	17:20	39	Rezerviši
4901	Preševo	07.12.2019	07:05	40	Rezerviši
4905	Preševo	07.12.2019	19:00	40	Rezerviši
5901	Dimitrovgrad	07.12.2019	07:00	25	Rezerviši
5903	Dimitrovgrad	07.12.2019	11:10	25	Rezerviši
5905	Dimitrovgrad	07.12.2019	15:55	25	Rezerviši
7821	Kuršumlija	07.12.2019	07:14	24	Rezerviši

Sada ćemo izabrati neki od ponuđenih vozova i kliknućemo na dugme Rezerviši, nakon čega nam se otvara stranica kao na slici 7, gde možemo videti relevantne podatke o karti koju smo rezervisali:

Uspešno ste rezervisali kartu

Broj voza: 2902	Destinacija: Beograd
Datum polaska: 07.12.2019	Vreme: 07:30

Povratak na: [Red vožnje](#)

Nakon klika na dugme Red vožnje, vraćamo se na prethodnu stranicu, gde možemo videti da se broj slobodnih mesta za naš voz smanjio za 1.

Red vožnje

Broj voza	Destinacija	Datum polaska	Vreme polaska	Broj slobodnih mesta	
790	Beograd	23.12.2019	06:40	14	Rezerviši
2902	Beograd	07.12.2019	07:30	26	Rezerviši
2904	Beograd	07.12.2019	12:10	30	Rezerviši
2906	Beograd	07.12.2019	15:50	29	Rezerviši
792	Beograd	07.12.2019	17:20	39	Rezerviši
4901	Preševo	07.12.2019	07:05	40	Rezerviši
4905	Preševo	07.12.2019	19:00	40	Rezerviši
5901	Dimitrovgrad	07.12.2019	07:00	25	Rezerviši
5903	Dimitrovgrad	07.12.2019	11:10	25	Rezerviši
5905	Dimitrovgrad	07.12.2019	15:55	25	Rezerviši
7821	Kuršumlija	07.12.2019	07:14	24	Rezerviši

Ovime smo završili sa objašnjenjem o korisničkom delu aplikacije, odnosno o tome kako krajnji korisnik može da je koristi. Sada ćemo da opišemo šta aplikacija radi u pozadini.

4. O kodu aplikacije

Prvo ćemo krenuti od fajla **app.js** koja izgleda ovako:

```
var createError = require('http-errors');
var express = require('express');
var path = require('path');
var cookieParser = require('cookie-parser');
var logger = require('morgan');
var bodyParser = require('body-parser')

var indexRouter = require('./routes/index');
var usersRouter = require('./routes/users');
var registracijaRouter=require('./routes/registracija')
var prijavaRouter=require('./routes/prijava');
var kartaRouter=require('./routes/karta');
var redVoznjeRouter=require('./routes/redvoznje')
var redVoznjeDBRouter=require('./routes/redvoznjeDB')
var app = express();

// view engine setup
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'ejs');

app.use(logger('dev'));
app.use(express.json());
app.use(express.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(express.static(path.join(__dirname, 'public')));

app.use(bodyParser.urlencoded({ extended: false }))

app.use('/', indexRouter);
app.use('/users', usersRouter);
app.use('/',registracijaRouter);
app.use('/',prijavaRouter);
app.use('/',kartaRouter);
app.use('/',redVoznjeRouter.router);
app.use('/',redVoznjeDBRouter)

// catch 404 and forward to error handler
app.use(function(req, res, next) {
  next(createError(404));
});

// error handler
app.use(function(err, req, res, next) {
  // set locals, only providing error in development
  res.locals.message = err.message;
  res.locals.error = req.app.get('env') === 'development' ? err : {};

  // render the error page
  res.status(err.status || 500);
  res.render('error');
});

module.exports = app;
```

Zatim čemo prikazati kako izgledaju kodovi za naredne templejte:

- error.ejs

```
<h1><%= message %></h1>
<h2><%= error.status %></h2>
<pre><%= error.stack %></pre>
```

- index.ejs

[illegible]

- karta.ejs

```

<h1>Broj slobodnih mesta</h1>
<h1><%= broj %></h1>
<h1><%= destinacija %></h1>
<h1><%= datum %></h1>
<h1><%= vreme %></h1>
<h1><%= brojMesta %></h1>

```

- `kupljenaKarta.ejs`

```
<h1>Uspješno ste rezervisali kartu</h1>
<div class="container">
  <div class="row border border-primary bg-primary">
    <div class="col">
      <h3>Broj voza: <%= voz.brojVoza %></h3>
    </div>
    <div class="col">
      <h3>Destinacija: <%= voz.destinacija %></h3>
    </div>
  </div>
  <div class="row border border-primary">
    <div class="col">
      <h3>Datum polaska: <%= voz.datumPolazka %></h3>
    </div>
    <div class="col">
      <h3>Vreme: <%= voz.vreme %></h3>
    </div>
  </div>
</div>

<h4>Povratak na: <a href="/redvoznje" class="btn btn-success">Red vožnje</a></h4>
```

- prijava.ejs

```
<body>
<form action="/prijava" method="POST">
  <div class="form-group">
    <label for="exampleInputEmail1">Email</label>
    <input type="email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp" name="email">
    <small id="emailHelp" class="form-text text-muted">Nikada necemo deliti tvoj email</small>
  </div>
  <div class="form-group">
    <label for="exampleInputPassword1">Sifra</label>
    <input type="password" class="form-control" id="exampleInputPassword1" name="sifra">
  </div>
  <button type="submit" class="btn btn-primary">Posalji</button>
</form>
```

- redvoznje.ejs

```
<h1>Red vožnje</h1>
<table class="table table-striped">
  <thead>
    <tr>
      <th scope="col">Broj voza</th>
      <th scope="col">Destinacija</th>
      <th scope="col">Datum polaska</th>
      <th scope="col">Vreme polaska</th>
      <th scope="col">Broj slobodnih mesta</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <% for(var i=0; i < vozovi.length; i++) { %>
        <tr>
          <td><%= vozovi[i].brojVoza %></td>
          <td><%= vozovi[i].destinacija %></td>
          <td><%= vozovi[i].datumPolazka %></td>
          <td><%= vozovi[i].vreme %></td>
          <td><%= vozovi[i].brojSlobodnihMesta %></td>
          <td>
            <a class="btn btn-primary" href=<%= vozovi[i].link %> role="button">Rezerviši</a>
          </td>
        </tr>
      <% } %>
    </tbody>
  </table>
```

- redvoznjeDB.ejs

```
<body>
<form action="/redvoznjeDB" method="post">
  Broj voza <input type="text" name="brojvoza" required>
  Destinacija <input type="text" name="destinacija" required>
  Datum <input type="text" name="datum" required>
  Vreme <input type="text" name="vreme" required>
  Broj mesta <input type="text" name="brojmesta" required>
  <input type="submit" value="Unesi" required>
</form>
```

- registracija.ejs

```
<form action="/registracija" method="POST">
  <div class="form-group">
    <label for="exampleInputEmail1">Email</label>
    <input type="email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp" name="email" required>
    <small id="emailHelp" class="form-text text-muted">Nikada necemo deliti tvoj email</small>
  </div>
  <div class="form-group">
    <label for="exampleInputPassword1">Sifra</label>
    <input type="password" class="form-control" id="exampleInputPassword1" name="sifra" required>
  </div>
  <div class="form-group">
    <label for="exampleInputPassword1">Ponovi sifru</label>
    <input type="password" class="form-control" id="exampleInputPassword1" name="sifraPonovljena" required>
  </div>
  <button type="submit" class="btn btn-primary">Posalji</button>
</form>
```

A nakon toga i kako izgledaju naredne rute za templejte:

- baza.js

```
var mongoose = require('mongoose');
mongoose.connect('mongodb://localhost:27017/korisnici', {useNewUrlParser: true,useUnifiedTopology: true});
var db = mongoose.connection;
db.on('error', console.error.bind(console, 'connection error:'));
db.once('open', function() {
  // povezani smo
  console.log('Povezani smo sa bazom')
});

var korisniciSchema = new mongoose.Schema({
  email:{
    type:String,
    required:true
  },
  sifra:{
    type:String,
    required:true
  }
});

var korisnikModel = mongoose.model('korisnikModel', korisniciSchema);//model korisnika

var vozoviSchema = new mongoose.Schema({
  brojVoza: {type:String,required:true},
  destinacija: {type:String,required:true},
  datumPolazka: {type:String,required:true},
  vreme: { type:String,required:true},
  brojSlobodnihMesta: {type:Number,required:true},
  link: {type:String,required:true},
});

var vozoviModel=mongoose.model('vozoviModel',vozoviSchema)//model voza

module.exports={korisnikModel:korisnikModel,vozoviModel:vozoviModel};
```

- index.js

```
var express = require('express');
var router = express.Router();

router.get('/', function(req, res, next) {
  res.render('index', { title: 'Express' });
});

module.exports = router;
```

- karta.js

```
var express = require('express');
var router = express.Router();
var vozovi=require('./redvoznje');
var Model=require('./baza');

router.get('/karta/:broj', function(req, res, next) {
  console.log(req.params.broj);
  var BSM;
  console.log(typeof(req.params.broj))
  Model.vozoviModel.findOne({ brojVoza:req.params.broj },function(err,data){
    BSM=data.brojSlobodnihMesta-1;
    if(BSM>1){
      Model.vozoviModel.updateOne({ brojVoza:req.params.broj },
        {$set:{ brojSlobodnihMesta:BSM } },function(err){ }).then(res.render('kupljenaKarta',{ voz:data}));
    }
    else{
      res.send('Nema dovoljno karata');
    }
  })
})

router.post('karta',(req,res,next)=>{ //Završi bazu
})
module.exports = router;
```

- prijava.js

```
var express = require('express');
var router = express.Router();
var mongoose = require('mongoose');
var Model=require('./baza');

//model korisnika
router.get('/prijava', function(req, res, next) {
  res.render('prijava', { title: 'Express' });
});

router.post('/prijava',(req,res,next)=>{
  emailR=req.body.email;
  sifraR=req.body.sifra;

  Model.korisnikModel.findOne({ email:emailR },(err,docs)=>{
    console.log(docs);

    if(docs!=null){
      if(docs.sifra==sifraR)
        res.redirect('/redvoznje')
      else{
        res.redirect('/prijava')
      }
    }else{
      res.redirect('/prijava')
    }
  })
  console.log(req.body)
})

module.exports = router;
```

- redvoznje.js

```
var express = require('express');
var router = express.Router();
var Model=require('./baza');

router.get('/redvoznje', function(req, res, next) {
  Model.vozoviModel.find({ },function(err,data){
    res.render('redvoznje', { vozovi:data })
  })
});
module.exports = {router};
```

- redvoznjeDB.js

```
var express = require('express');
var router = express.Router();
var Model=require('./baza');

router.get('/redvoznjeDB', function(req, res, next) {
  res.render('redvoznjeDB', { title: 'Express' });
});
router.post('/redvoznjeDB', function(req, res, next) {
  brojVoza1=req.body.brojvoza;
  destinacija1=req.body.destinacija;
  datumPolzka1=req.body.datum
  vreme1=req.body.vreme
  brojSlobodnihMesta1=req.body.brojmesta
  link1='/karta/'+ req.body.brojvoza
  console.log(req.body.destinacija)
  var Voz = new Model.vozoviModel(
    {
      brojVoza:brojVoza1,
      destinacija:destinacija1,
      datumPolazka:datumPolzka1,
      vreme:vreme1,
      brojSlobodnihMesta:brojSlobodnihMesta1,
      link:link1
    }
  ).save().then(res.send('Uspesno upisano'));
});
module.exports = router;
```

- registracija.js

```
var express = require('express');
var router = express.Router();
var Model=require('./baza');

router.get('/registracija',(req,res,next)=>{
  res.render('registracija', { title: 'Express' });
})
router.post('/registracija',(req,res,next)=>{
  console.log(req.body);
  var sifra1=req.body.sifra;
  var sifra2=req.body.sifraPonovljena;

  if(sifra1===sifra2){ //rad sa bazom podataka
    var korisnikM = new Model.korisnikModel(
      { email: req.body.email,
        sifra:sifra1
      });
    korisnikM.save();
    res.redirect('/');
  })
module.exports = router;
```

- users.js

```
var express = require('express');
var router = express.Router();

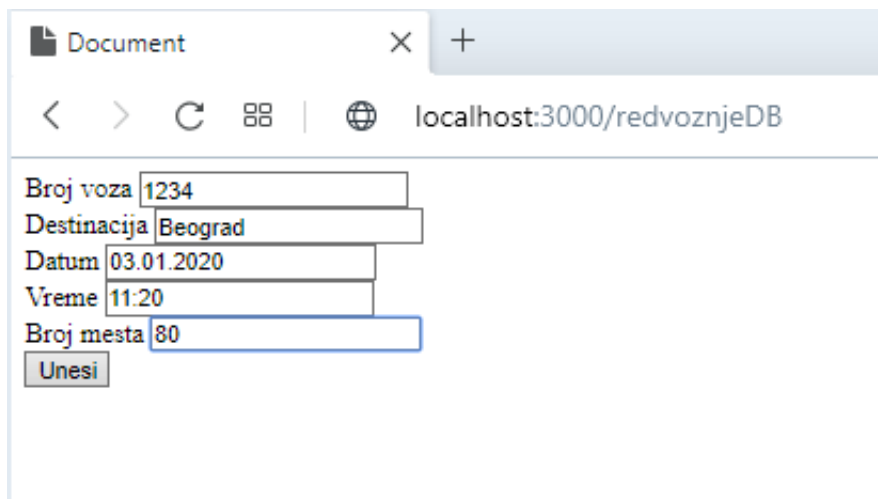
/* GET users listing. */
router.get('/', function(req, res, next) {
  res.send('respond with a resource');
});

module.exports = router;
```

5. Editovanje baze promenom podataka u aplikaciji i obratno

Upis vozova u stranicu „Red vožnje“ vrši se upisivanjem sledećeg linka u pretraživač:
<http://localhost:3000/redvoznjeDB>

Kao što možemo da vidimo na slici 9, nakon što u poljima unesemo podatke o vozu i kliknemo na dugme Unesi, pojaviće nam se obaveštenje „Uspesno upisano“ i na stranici „Red vožnje“, doći će do promene i biće prikazan novi voz.



Document X +

< > C || | localhost:3000/redvoznjeDB

Broj voza

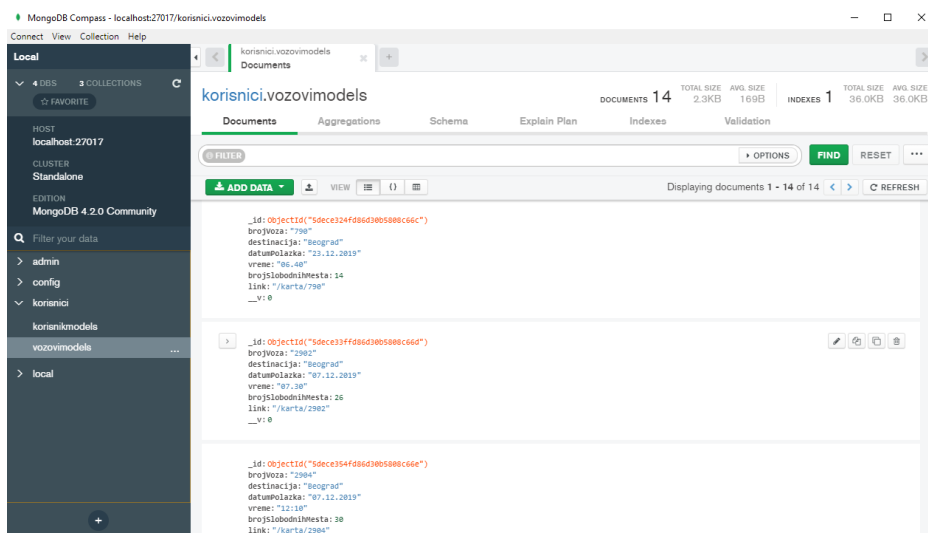
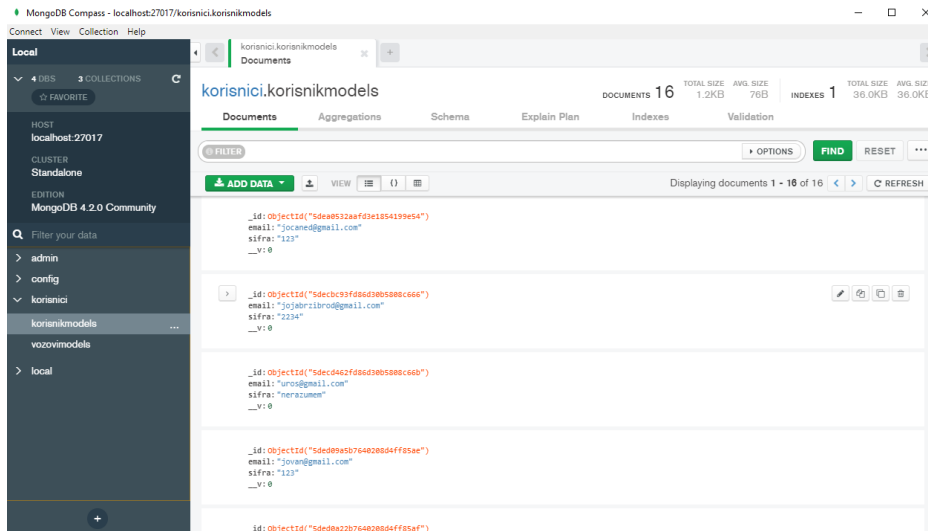
Destinacija

Datum

Vreme

Broj mesta

Naravno i sve promene koje napravimo u aplikaciji prilikom registrovanja korisnika, rezervacije karte i upisa voza u bazu, prikazaće se automatski i u našoj MongoDB bazi, kao na slikama 10 i 11, odakle takođe promene koje tu izvršimo mogu da se odraze i u našoj aplikaciji:



Literatura

Uvod u NodeJS..... <https://www.youtube.com/watch?v=VjwoppW99r0&list=PLXNTNfUTUEe6vwDz9gICfaJ4tTwAatDEv>
Learn NodeJS.....<https://www.codecademy.com/learn/learn-node-js>
NodeJS MongoDB i AngularBrad Dayley, Brendan Dayley, Caleb Dayley