VISOKA TEHNIČKA ŠKOLA
STRUKOVNIH STUDIJA

Novi Automobili

Predmet: Klijent Server Sistemi

Profesor:

dr Mirko Kosanović

Miloš Kosanović

Student:

Dimitrije Tatić

Ser 72/17

Februar 2021.

SADRŽAJ

1. Uvod

U ovom projektu obrađena je izrada web aplikacije koja će predstavljati oglasnu tablu novih automobila kojoj je moguće pristupiti bez prethodnog prijavljivanja na istu. Samo uz prethodno registrovani nalog moguće je objavljivati oglase, izmenjivati ih i brisati. Tehnologije koje su korišćene na klijentskoj strani su HTML5, CSS sa Bootstrap framework-om i JavaScript, dok su na serverskoj strani korišćeni NodeJS sa Express framework-om. Za usluživanje HTML stranica klijentu koristili smo EJS templetski jezik. Alati koji su korišćeni prilikom izrade aplikacije su Google Chrome pretraživač i Visual Studio Code kao text editor.

Aplikacija se sastoji iz dva dela. Prvi deo aplikacije jeste oglasna tabla. Kada joj korisnik pristupi dobija mogućnost da čita postojeće oglase i pristupi svakom od njih pojedinačno. Prilikom pristupa stranici nekog konkretnog oglasa, korisnik može pročitati karakteristike automobila i videti slike u okviru oglasa ili pristupiti svakoj pojedinačno. Drugi deo sadrži formu za registraciju korisnika koji tom prilikom bivaju upisani u bazu, i (ili) prijavu istih koji već imaju kreiran nalog**(Ovaj deo aplikacije je realizovan po ugledu na aplikaciju koju sam naveo u literaturi pod rednim brojem dva, sa pojedinim izmenama)**, koji zatim mogu da objavljuju oglase sa slikama, izmenjuju ih ili brišu.

2. Instalacija i podešavanje projekta

Da bismo pokrenuli NodeJs aplikaciju potrebno je da instaliramo NodeJs, a za ovaj projekat koristimo i MongoDB Atlas, pa je potrebno i pristupiti bazi podataka na cloud-u ako bismo hteli da vidimo kako izgleda unos korisnika prilikom registracije, ili objavljivanje oglasa u bazu podataka.

2.1. Instaliranje modula

U ovom projektu nalazi se **package.json** fajl, to je fajl koji se inicijalizuje sa projektom, naravno na zahtev programera, i ukoliko popunimo sve informacije ispravno biće kreiran fajl. Zatim kad krenemo da instaliramo nove module i ukoliko upotrebimo neke ključne reči (-S , -save) prilikom instalacije u ovom fajlu biće nam upisani svi moduli koje koristimo za našu aplikaciju. U koliko je sve to ispravno kreirano, da bi smo na nekom drugom računaru pokrenuli aplikaciju, potrebni su nam moduli koji su definisani unutar ovog fajla, pa kucamo sledeću

naredbu u konzoli **npm install,** ova naredba prvo pretražuje **package.json** fajl i u njemu traži i instalira sve dependence(tj. Module ili biblioteke) koji su potrebni za ovaj projekat.

3. Arhitektura aplikacije

Aplikacija u sebi sadrži korenski (engl. Root) direktorijum '/novi_automobili' koji u sebi sadrži datoteku '/package.json' koja predstavlja JSON dokument koji opisuje samu aplikaciju i sadrži spisak modula od kojih je serverski deo aplikacije zavistan i app.js datoteku u kojoj je kod NodeJS servera. Nakon instalacije modula iz JSON dokumenta, u korenskom direktorijumu se kreira '/node_modules' direktorijum koji sadrži module koj i su potrebni samoj aplikaciji. U korenskom direktorijumu se takođe nalaze '/config' direktorijum u kome se nalaze podešavanja modula koje koristimo, '/views' direktorijum u kome je smešten klijentski deo aplikacije, '/routes' direktorijum u kome se nalaze fajlovi 'index.js' i 'users.js' kojima su opisane rute , '/models' sa dva fajla 'User.js' i 'Oglas.js' koji opisuju MongoDB šeme za čuvanje korisnika i oglasa u bazi. U '/views' direktorijumu se nalazi 'layout.ejs' dokument koji predstavlja osnovni EJS dokument u kome će se dal je ostali .ejs fajlovi prikazivati kada ih vraćamo klijentu putem hendlerskih funkcija ruta, i 'partials' direktorijuma u kome se nalazi ‘messages.ejs’ fajl pomoću kojeg ćemo predstavljati poruke klijentu(Uspešna registracija, Pogrešna šifra, Nisu popunjena sva polja …).

3.1 Serverski deo

Za realizaciju serverskog dela (back-end) koristićemo Node.js za inicijalizaciju servera pomoću Express modula uz korišćenje drajvera Mongoose za povezivanje sa bazom podataka, koja je u našem slučaju na cloud-u, a to je MongoDB Atlas. Kao pripomoć u korišćenju HTTP metoda za brisanje i izmenu oglasa koristimo *modul Method-Override*. Zamena ID-a u URL-u rute oglasa za “*slug*”(Pseudo ime) je omogućena pomoću modula *Slugify*. Upload slika realizovan je uz pomoć modula *Multer*. Za usluživanje HTML stranica koristili smo templejtski jezik EJS. Za autentifikaciju korisnika smo iskoristli middleware pod nazivom Passport, koji nam omogućava lakšu proveru korisnika prilikom registracije i prijavljivanja na sajt. Za enkripciju šifre koju korisnik odabere prilikom registracije koristili smo modul Bcrypt koji nam omogućava lakše hash-ovanje šifre, a za prikaz poruka o grešci ili uspešnosti (Registracije, Prijave, Komentarisanja...) smo iskoristili *modul Flash*.

Node.js

Node.js je programski jezik zasnovan na JavaScript jeziku. On je ne-blokirajući, event driven, lightweight, efikasan jezik čija je glavna namena da se koristi kod distribuiranih aplikacija koje rade na različitim platformama i koje imaju potrebu da rade sa velikim količinama zahteva ili podataka u realnom vremenu. Node.js je naročito pogodan za aplikacije koje moraju da održavaju perzistentnu konekciju sa serverom. Mrežne aplikacije koje zahtevaju brzinu, skalabilnost i podržavaju veliki broj istovremenih konekcija se razvijaju u ovom programskom jeziku.

Više o NodeJS-u: https://nodejs.org/en/

Express modul

Express modul je najpopularniji i najrasprostranjeniji modul koji se koristi u Node.js aplikacijama. Omogućava ubrzanje razvoja i koristi se slično kao http i https moduli, samo što ima dodatne mogućnosti. Kako Express nije glavni modul on se mora instalirati lokalno uz pomoć komande npm-a (Node package manager).

Passport

Passport jeste middleware koji nam omogućava autentifikaciju korisnika u Node.js aplikacijama. Izuzetno je fleksibilan I modularan, pa se tako može I neprimetno ubaciti u bilo koju aplikaciju zasnovanu na Express-u. Sveobuhvatan skup strategija podržava autentifikaciju pomoću korisničkog imena I lozinke, Facebook-a, Twitter-a I još mnogo toga.

Više o Passport-u: http://www.passportjs.org/

3.2 Klijentski deo

Za realizaciju klijentskog dela (front-end) koristićemo HTML5, CSS tehnologije uz CSS Framework pod nazivom Bootstrap. Za ikonice koje smo umetnuli u dizajn, koristili smo se jednim od najboljih besplatnih open-source GitHub projekata pod nazivom FontAwesome. Da bi koristili besplatan kit FontAwesome-a, moramo kreirati nalog i potom se prijaviti da bi pristupili biblioteci ikonica koju imaju na usluzi. Klijentu ćemo uslužiti stranice koje zahteva od servera putem EJS templejtskog jezika.

3.3 Baza podataka

MongoDB

Da bi govorili o MongoDB Atlas-u, moramo prvo reći sta je MongoDB. MongoDB je NoSQL baza podataka. NoSQL znači u stvari Not Only SQL i označava skup baza podataka koje ne funkcionišu na način kao što to rade relacione baze podataka. MongoDB je po strukturi dokumentovano orijentisana baza podataka. Gde se podaci smeštaju u obliku ključ-vrednost parova. Struktura dokumenata je slična JSON formatu, a skup dokumenata koji imaju istu namenu naziva se kolekcija. Kolekcija je nešto slično tabeli u relacionom modelu baze podataka, ali dokumenti u okviru kolekcije mogu ali i ne moraju da imaju iste atribute**.**

Više o MongoDB-u: https://www.mongodb.com/

MongoDB Atlas jeste baza podataka na cloud-u kojom može u potpunosti da se upravlja, a razvili su je isti ljudi koji su izgradili MongoDB. I to je ustvari web platforma koju iznajmljuje MongoDB I na njoj možemo skladištiti baze podataka gde su replike I particije već predefinisane a mogu se dodatno podešavati prostim klikom na dugme. Prilikom korišćenja MongoDB Atlas-a nije potrebno instalirati MongoDB na lokalnom racunaru već možemo pristupiti bazi podataka koja je na cloud-u.

Više o MongoDB Atlas-u: https://www.mongodb.com/cloud/atlas

U ovom projektu smo se koristili ovom bazom podataka, a to je moguće na sledeći način:

- Pristup stranici https://www.mongodb.com/cloud/atlas/signup i kreirati nalog
- Pošto smo kreirali nalog i ulogovali se na sajt potrebno je napraviti klaster u našem slučaju će to biti besplatan klaster
- Pri kreaciji klastera moraćemo i da upišemo IP adresu sa koje ćemo pristupati istom, klikom na dugme ‘Connect’ nailazimo na polje koje služi toj svrsi, u našem slučaju će adresa biti 0.0.0.0 (tj. sa svih adresa)
- Zatim trebamo kreirati korisnika koji će koristiti klaster klikom na isto dugme ‘Connect’ i pristupom dialogu za unos korisničkog imena i lozinke
- Zatim se trebamo konektovati na kreirani klaster iz naše aplikacije

Uputstvo možete naći i na sledećem linku: https://docs.atlas.mongodb.com/getting-started/

Pristup bazi podataka koja je korišćena u realizaciji ovog projekta možemo naći na sledećem linku: noviAutomobili Overview | Atlas: MongoDB Atlas

- Email: taticdimitrije3@gmail.com
- Lozinka: Novasifra12

3.4 Komunikacija

Unutar korenskog direktorijuma nalazi se direktorijum pod nazivom ‘routes’ u kome se nalaze ‘index.js’ i ‘user.js’ fajlovi koji opisuju rute putem kojih na osnovu zahteva klijenta server vraća zahtevane stranice generisane putem EJS templejtskog jezika.

U nastavku ćemo videti tabelu ruta i kratak opis istih.

TABELA RUTA:

RUTA:	OPIS:
GET: http://localhost:5000/oglasi	Vraća oglasnu tablu sajta
GET: http://localhost:5000/users/register	Vraća stranicu forme za registraciju korisnika
GET: http://localhost:5000/users/login	Vraća stranicu za prijavu korisnika
GET: http://localhost:5000/oglasi/novi	Vraća stranicu za kreiranje novog oglasa
GET:http://localhost:5000/oglasi/izmeni/:id	Vraća stranicu za izmenu oglasa
GET:http://localhost:5000/oglasi/moji/:slug	Vraća stranicu oglasa prijavljenog korisnika
GET: http://localhost:5000/oglasi/:slug	Vraća stranicu traženog oglasa
PUT:http://localhost:5000/oglasi/izmeni/:id	Šalje podatke iz forme za izmenu oglasa
DELETE:http://localhost:5000/oglasi/izbrisi/:id	Brisanje oglasa
POST: http://localhost:5000/oglasi/novi	Šalje podatke iz forme za kreiranje oglasa

4. Rad aplikacije

4.1 Opis implementacije

Dakle, implementacija prethodno instaliranih modula se vrši na sledeći način u app.js fajlu:

```
const express = require('express');
const expressLayouts = require('express-ejs-layouts');
const mongoose = require('mongoose');
const passport = require('passport');
const flash = require('connect-flash');
const session = require('express-session');
const methodOverride = require('method-override');
const multer = require('multer');
const fs = require('fs');
```

EJS template language(templejtski jezik) ćemo koristiti za renderovanje HTML stranica koje klijent potražuje. Po defoltu EJS pri instalaciji nema layout, pa smo morali dodatno da ga implementiramo.

Mongoose ćemo iskoristi kao drajver koji nam omogućava komunikaciju između baze podataka MongoDB, koja je u našem slučaju na cloud-u (MongoDB Atlas), i aplikacije kojom bi trebali da manipulišemo tim podacima u cilju kreiranja i provere legitimnosti korisnika.

Flash modul će nam pomoći prilikom prikazivanja poruka o uspehu ili greškama prilikom registracije, prijavljivanja na sajt ili pak komentarisanja. Za to će nam biti potreban modul express-session jer Flash zavisi od sesije koju korisnik koristi prilikom posete sajtu.

Passport middleware koristićemo za autentifikaciju, i uz njega ćemo implementirati strategiju koju želimo da koristimo u posebnom fajlu, i prilikom implementiranja strategije ona biva sagledana kao odvojeni modul, u našem slučaju će to biti lokalna strategija na osnovu koje ćemo proveravati da li zahtev za resurse dolazi od strane verifikovanog korisnika.

MethodOverride modul služi nam kako bi mogli da koristimo HTTP metode kao što su PUT i DELETE, na mestima gde klijent to ne dozvoljava(HTML5 forme podržavaju metode GET i POST).

Slugify modul koristimo u cilju promene izgleda URL-a konkretnog oglasa i oglasa konkretnog korisnika, kako bi umesto ID-eva oglasa ili korisnika mogli da pristupimo tim stranicama putem ruta koje koriste njihova pseudo imena(*slug*).

Multer middleware omogućava da uploud-ujemo sliku ili više slika u okviru oglasa tako što sa podacima iz forme pored *req.body* parametra dodaje i *req.files* parametar putem kojeg možemo da pristupimo informacijama fajla i prebacimo ga u željeni direktorijum,menjamo ime...

Implementacija Lokalne Strategije za Passport autentifikaciju I bcrypt modula za enkripciju lozinki u 'passport.js' fajlu:

```
const LocalStrategy = require('passport-local').Strategy;
const bcrypt = require('bcryptjs');
```

Bcrypt modul jeste funkcija za hash-ovanje(prikrivanje) lozinke koja se služi salt-om(U kriptografiji salt ili na srpskom so je random podatak koji se koristi kao dodatni ulaz jednosmernoj funkciji koja ima lozinku. Salt se koristi za zaštitu lozinki u skladištu.) da bi zaštitila lozinku od takozvanih “Rainbow table” napada.

4.2 Opis funkcionalnosti – korisničko uputstvo

Da bi pokrenuli aplikaciju (naravno ako smo pre toga instalirali potrebne module iz fajla "package.json") u konzoli koju smo otvorili u korenskom direktorijumu treba pokrenuti komandu "**npm start**", a ako želimo da pravimo izmene na serveru prilikom rada aplikacije a da te izmene budu vidljive u aplikaciji, aplikaciju treba pokrenuti komandom „**npm run dev"** koja pokreće aplikaciju putem *nodemon modula* i u slučaju izmene restartuje server kako bi promena bila vidljiva.

Nakon pokretanja aplikacije, pristup početnoj stranici vršimo putem rute **http://localhost:5000/**. Prilikom pristupa početnoj stranici korisniku će biti omogućeno listanje oglasa, ali ne i objavljivanje dok ne napravi nalog. U slučaju da nema nalog kojim bi mu bila omogućena objava oglasa, korisnik bi izabrao registraciju i bivao preusmeren na stranicu registracije gde bi naišao na formu sa obaveznim poljima u koje bi trebao da unese svoje podatke ograničene odredjenim kriterijumima koji trebaju biti zadovoljeni. U slučaju neuspešne registracije biće obavešten od strane servera o greškama i o tome šta bi trebao ispraviti u popunjavanju navedenih polja. U slučaju uspešne registracije biće preusmeren na stranicu za prijavu gde bi trebao da unese email i šifru svog naloga. Ako pak načini neku grešku prilikom prijave, o tome će biti obavešten konkretnom porukom. U slučaju uspešne prijave, klijent ponovo biva preusmeren na početnu stranicu oglasne table gde može čitati oglase ali takođe ih i objavljivati. Putem navigacije može odabrati da doda oglas, da pogleda oglase koji su objavljeni sa njegovog naloga i izlogovati se. Prilikom dodavanja oglasa korisnik će odabrati naslov oglasa, model automobila, pogon, grad u kome se nalazi, cenu, godište, oblik karoserije i naravno opisati ponudu automobila koji nudi uz objavu slika(Maksimum 10, pod uslovom da svaka ne prelazi veličinu od 5MB). Prilikom pristupa stranici sopstvenih oglasa, ili pristupa pojedinačnom oglasu(u slučaju da je oglas objavljen od strane istog korisnika), korisnik će dobiti mogućnost izmene ili brisanja oglasa. Prilikom izmene oglasa korisnik će moći da izmeni sve informacije o automobilu sem slika. Ako bi to želeo, treba bi da izbriše oglas i ponovo ga postavi na oglasnu tablu.

Za testiranje aplikacije možemo iskoristiti sledeće podatke za prijavu na sajt:

- Email: taticdimitrije3@gmail.com
- Lozinka: 123456

5. Literatura

1. Klijent server sistemi, M. Kosanović, interna skripta: https://vtsnis.edu.rs/wp-content/plugins/vts-predmeti/uploads/2019_Praktikum_V1.pdf
2. bradtraversy/node_passport_login: Node.js login, registration and access control using Express and Passport (github.com)
3. WebDevSimplified/Markdown-Blog (github.com)
4. https://nodejs.org/docs/latest-v11.x/api/
5. http://www.passportjs.org/docs/
6. https://docs.atlas.mongodb.com/
7. https://www.npmjs.com/package/bcrypt
8. https://github.com/expressjs/flash
9. slugify - npm (npmjs.com)
10. multer - npm (npmjs.com)
11. method-override - npm (npmjs.com)