

SEMINARSKI RAD

PROGRAMIRANJE MOBILNIH APLIKACIJA

SportApp

Student: Josip Goreta

Datum: 15.02.2019

Sadržaj

1	UVOD	1
2	STRUKTURA APLIKACIJE	2
2.1	Config	2
2.2	Models	3
2.3	Public	3
2.4	Routes	3
2.5	Views	3
2.6	Node_modules	3
3	OPIS KORIŠTENIH TEHNOLOGIJA	4
3.1	Pug	4
3.2	MongoDB	5
3.2.1	Unos podataka u bazu – „Create“	6
3.2.2	Brisanje podataka iz baze – „Delete“	12
3.2.3	Ažuriranje podataka u bazi – „Update“	14
3.3	Registracija korisnika – „Passport.js“	15
3.4	Kriptiranje lozinke – „bcryptjs“	16
3.5	Autoincrement ID	16

1 UVOD

Aplikacija je osmišljena tako da krajnjem korisniku prikazuje sportske rezultate koje s druge strane uređuju administratori te aplikacije. Krajnji korisnik može vidjeti samo naslovnu stranu i listati po njoj.

Aplikacija je smještena na Heroku server na linku: <https://sport-app-jg.herokuapp.com/>

Za izgled aplikacije zaslužan je Bootstrap front-end Web framework.

Administratori aplikacije imaju dodatne mogućnosti. Jedan glavni administrator je prilikom kreiranja baze podataka ubačen u bazu i on se može prijaviti u aplikaciju i onda uređivati mečeve, dodavati sportove, klubove, lige i registrirati nove administratore.

Registrirati nove administratore može bilo koji registrirani administrator. Krajnjem korisniku je zabranjeno pristupiti svim dijelovima aplikacije osim naslovnoj strani gdje mogu pratiti rezultate koje za njih unose administratori. To je omogućeno izradom ove metode koja se dalje poziva kao parametar u funkcijama pri pristupanju stranicama:

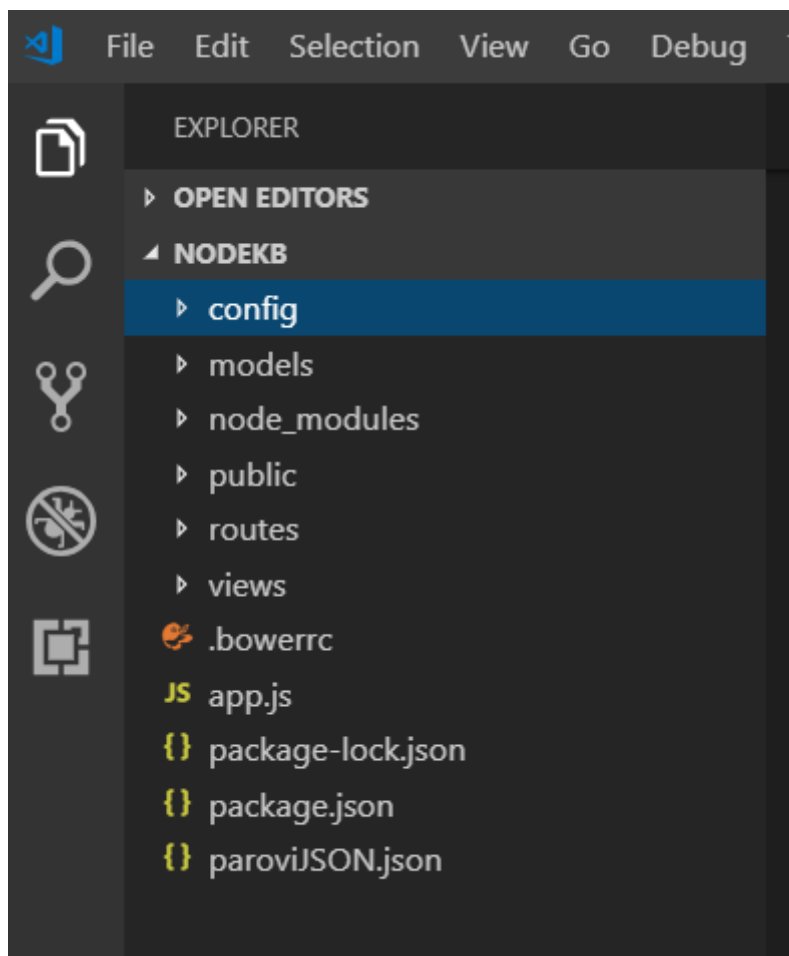
```
//Access Control
function ensureAuthenticated(req, res, next){
  if(req.isAuthenticated()){
    return next();
  }else{
    req.flash('danger', 'Please login');
    res.redirect('/users/login');
  }
}
```

```
//Add route
router.get('/add', ensureAuthenticated, function(req, res){
  res.render('add_league',{
    title: 'Add League'
  });
});
```

Korištene tehnologije i jezici: JavaScript, Pug, CSS, Node.js, Express.js, Bootstrap, MongoDB, jQuery.

2 STRUKTURA APLIKACIJE

Izgled strukture aplikacije:



2.1 Config

Unutar „config“ nalaze se dvije datoteke, a to su database.js i passport.js.

Database.js sadrži samo putanju do baze podataka. Ovaj modul je „exportan“ tako da je poslije u aplikaciji omogućeno da umjesto da se piše cijela putanja, napiše se samo „database“.

Passport.js sadrži neke potrebne stvari u vezi autentifikacije, prijave i registracije korisnika.

2.2 Models

Unutar „models“ opisani su atributi entiteta koji se koriste u aplikaciji kao što su „club“, „league“, „match“, „sport“, „user“. Svaki od tih modela ima svoju zasebnu kolekciju u bazi podataka.

2.3 Public

Unutar „public“ nalaze se datoteke za Bootstrap, jQuery, CSS te main.js u kojem je korišten i ajax koji je omogućio brisanje elemenata iz aplikacije.

2.4 Routes

Unutar „routes“ se nalaze zasebne datoteke za svaki model koji je opisan u „models“. U svakoj datoteci su napisane rute za pristup dijelovima aplikacije ali baš za taj model. Ove rute su razdvojene u različite datoteke radi preglednosti koda.

2.5 Views

Unutar „views“ nalaze se .pug datoteke u kojima se uređuje izgled aplikacije. Tako tu imamo datoteke koje uređuju stranice za dodavanje novih modela, ažuriranje modela, prijavu, registraciju...

2.6 Node_modules

Unutar „node_modules“ nalaze se datoteke potrebne za rad servera. Datoteke servera su još i app.js te package.json.

3 OPIS KORIŠTENIH TEHNOLOGIJA

3.1 Pug

Da bismo razumjeli što je Pug, moramo znati da preglednik čita HTML i CSS, a zatim prikazuje formatirane slike i tekst klijentu na temelju onoga što mu oni govore.

U svemu tome Pug je posrednik. On je „template engine“ za Node.js. Ukratko, za vrijeme izvođenja Pug zamjenjuje varijable u našoj datoteci sa stvarnim vrijednostima, a zatim pošalje rezultirajući HTML string klijentu.

U sklopu Pug-a su korišteni „mixins“. Mixins nam omogućuju stvaranje blokova za Pug koji se mogu ponovno koristiti.

Primjer mixina u Pug-u:

```
mixin comment(commentData)
  li.comment
    .league= commentData.league
    .date= commentData.date
    .time= commentData.time
    .host= commentData.host
    .host_goal= commentData.host_goal
    .guest_goal= commentData.guest_goal
    .guest= commentData.guest
    .postedByAdmin= commentData.postedByAdmin
```

U ovom radu korišten je Pug jer naspram HTML-a omogućuje rad sa varijablama, nizovima i ostalim tipovima podataka što možemo vidjeti u sljedećem kodu:

```
//-dohvaćamo u niz mySports sve različite sportove iz mečeva zbog
sortiranja mečeva po sportovima
- var mySports= []
  each match, i in matches
    if !mySports.includes(match.sport_id)
      -mySports.push(match.sport_id)
```

```
//-ovaj niz nam treba zbog ispisa na zaslon, u njega dohvaćamo JSON oblik svakog meča zasebno i onda možemo jednostavno dohvaćati attribute iz njegovih elemenata
```

```
- var allMatches= []  
- for (var i = 0; i < matches.length; ++i) {  
    -allMatches.push(matches[i])  
- }  
  
- for (var i = 0; i < mySports.length; ++i) {  
    h1= mySports[i]  
    ul.list-group  
        - for (var j = 0; j < allMatches.length; ++j) {  
            if allMatches[j].sport_id==mySports[i]  
                - const c1 = { postedByAdmin:allMatches[j].duration,  
date: allMatches[j].date_of_play, time: allMatches[j].playing_time,  
host:allMatches[j].host_id, host_goal:allMatches[j].host_goals+" - ",  
guest:allMatches[j].guest_id, guest_goal:allMatches[j].guest_goals,  
league:allMatches[j].league_id}  
                +comment(c1)  
            - }  
        - }  
- }
```

3.2 MongoDB

MongoDB je nerelacijska baza podataka koja nam omogućuje ogromnu skalabilnost i fleksibilnost u vezi upita. Pohranjuje podatke u fleksibilne JSON dokumente, a struktura podataka se može mijenjati tijekom vremena. Zbog svih tih svojih prednosti korištena je u izradi ove aplikacije. Baza nije lokalna već se nalazi na internetu tako da joj se može pristupiti sa bilo kojeg uređaja koji je spojen na Internet. Ova baza radi sa kolekcijama, što su sinonimi za tablice u relacijskim bazama podataka. Ovako izgleda popis kolekcija u bazi:

Collections	Users	Stats	Backups	Tools
<div> Delete all collections Add collection </div>				
NAME	DOCUMENTS	CAPPED?	SIZE	
clubs	51	false	19.94 KB	
leagues	14	false	11.02 KB	
matches	10	false	12.83 KB	
sports	3	false	8.31 KB	
users	11	false	10.56 KB	

NAME	DOCUMENTS	SIZE
system.indexes	7	0.77 KB

Kad se uđe u pojedinu kolekciju dobijemo ovakav pregled baze:

Display mode: ☒ list ☐ table (edit table view)

records / page 10 [1 - 10 of 10]

ID	Sport	League	Host	Guest	Date	Time	Host Goals	Guest Goals	Status
3	Rukomet	Danska 1	Ribe-Esbjerg	Kolding	2019-02-08	19.00	22	20	Kraj
4	Nogomet	Italija 1	Chievo	Roma	2019-02-08	20.30	0	3	Kraj
5	Nogomet	Engleska 1	Liverpool	Bournemouth	2019-02-09	16.00	3	0	Kraj
6	Nogomet	Engleska 1	Fulham	Manchester Utd	2019-02-09	13.30	0	3	Kraj
7	Nogomet	Engleska 1	Crystal Palace	West Ham	2019-02-09	16.00	1	1	Kraj
8	Nogomet	Engleska 1	Huddersfield	Arsenal	2019-02-09	16.00	1	2	Kraj
9	Nogomet	Engleska 1	Southampton	Cardiff	2019-02-09	16.00	1	2	Kraj
10	Nogomet	Engleska 1	Watford	Everton	2019-02-09	16.00	1	0	Kraj
11	Nogomet	Engleska 1	Brighton	Burnley	2019-02-09	18.30	0	0	Nije pöelo
12	Rukomet	SEHA Liga	Nexe	Vardar	2019-02-09	20.00	0	0	Nije pöelo

records / page 10 [1 - 10 of 10]

3.2.1 Unos podataka u bazu – „Create“

Primjer koda kojim aplikacija unosi podatke u bazu:

```
//Add Submit Post Route
router.post('/add', function(req, res){
  req.checkBody('id', 'Id is required').notEmpty();
  req.checkBody('name', 'Name is required').notEmpty();
  req.checkBody('sport_id', 'Sport ID is required').notEmpty();

  //Get Errors
  let errors = req.validationErrors();

  if(errors) {
    res.render('add_league', {
      title:'Add League',
```



```

        errors:errors
    });
}else{
    let league = new League();
    league.id = req.body.id;
    league.author = req.user._id;
    league.name = req.body.name;
    league.sport_id = req.body.sport_id;

    league.save(function(err){
        if(err){
            console.log(err);
            return;
        }else{
            req.flash('success', 'League Added');
            res.redirect('/lige');
        }
    });
}
});

```

Primjer koda kojim aplikacija čita podatke iz baze:

```

//Route fot add new Match
app.get('/matches/add',ensureAuthenticated, function(req, res){
    Match.find({}, function(err, matches){
        Club.find({}, function(err, clubs){
            League.find({}, function(err, leagues){
                Sport.find({}, function(err, sport){
                    if(err){
                        console.log(err);
                    }
                    else{
                        res.render('add_match', {
                            title: 'Add Match',

```

```

    matches : matches,
    clubs : clubs,
    leagues : leagues,
    sport : sport...

```

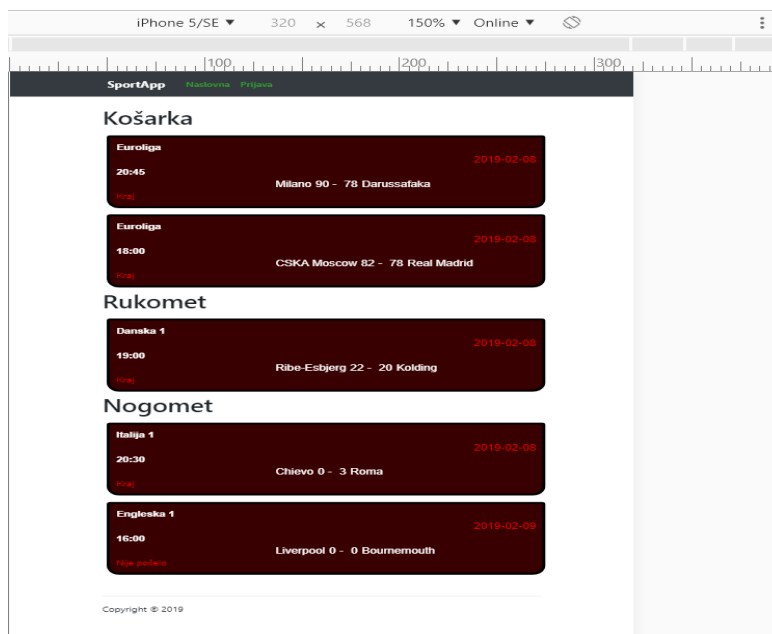
Kod poviše omogućuje da se na putanji „/matches/add“ može doći do podataka o kolekcijama „matches, clubs, leagues i sport“. Ti podaci se poslije ispisuju u Pug-u npr. ovim kodom:

```

block content
  h1 #{title}
  br
  h1 Dodani klubovi
  br
  ul.list-group
    each club, i in clubs
      li.list-group-item
        a(href="/clubs/"+club._id)=club.name + " (" +club.sport_id+")"

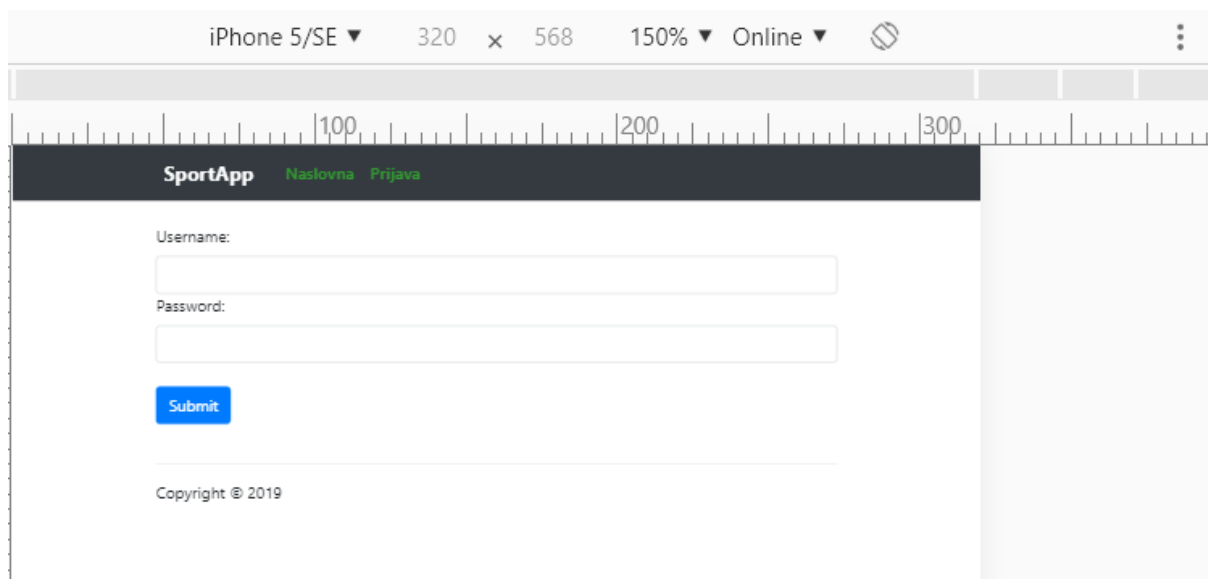
```

Naslovna stranica kada se učita izgleda ovako:



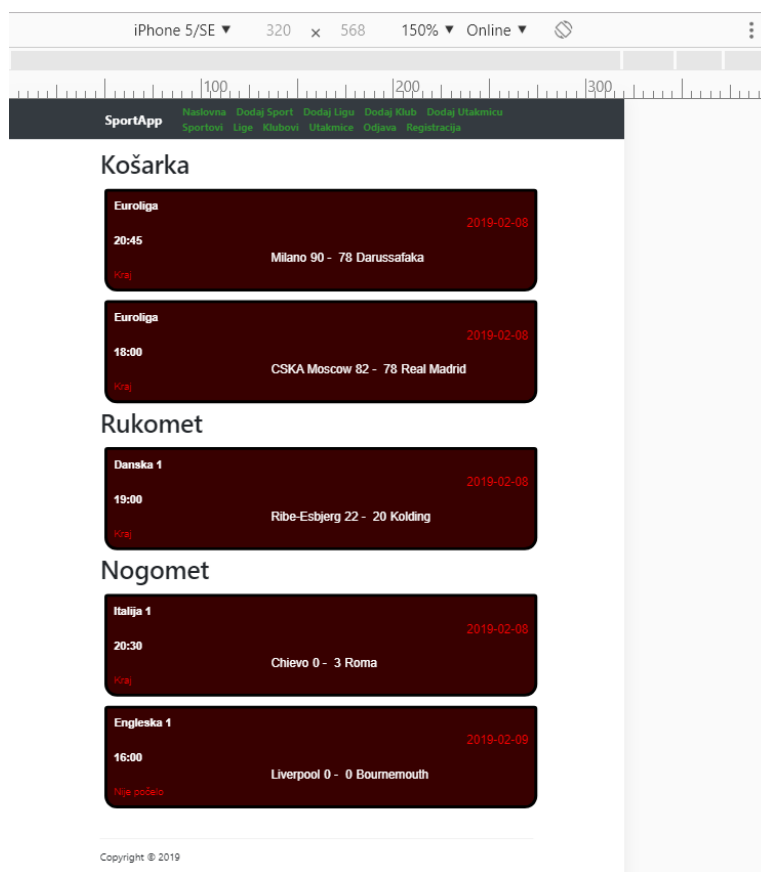
Krajnji korisnik može vidjeti samo ovu stranicu i ne može otići na druge stranice koje koriste administratori niti preko adresne trake jer su sve rute zaštićene od neovlaštenog pristupa, tj. mogu im pristupiti samo logirani korisnici.

Klikom na „Prijava“ otvara se prozor:



The screenshot shows a mobile browser window with the address bar displaying 'iPhone 5/SE', '320 x 568', '150%', 'Online', and a lock icon. The page has a dark header with 'SportApp' and two green links: 'Naslovna' and 'Prijava'. Below the header is a login form with two input fields labeled 'Username:' and 'Password:', followed by a blue 'Submit' button. At the bottom, there is a copyright notice 'Copyright © 2019'.

Tu se registrirani administrator može prijaviti i onda mu se otvara aplikacija sa dodatnim mogućnostima kao na sljedećoj slici:



Tu se administratoru ponude dodatne opcije za uređivanje baze podataka sa koje aplikacija čita podatke.

Klikom na „**Dodaj** Sport, Ligu, Klub, Utakmicu“ u bazu se dodaje novi sport, liga, klub, utakmica:



SportApp
Naslovna
Dodaj Sport
Dodaj Ligu
Dodaj Klub
Dodaj Utakmicu
Sportovi
Lige
Klubovi
Utakmice
Odjava
Registracija

Dodaj utakmicu

ID utakmice:

6

Odaberi sport kojem utakmica pripada:

Nogomet

Košarka

Rukomet

Nogomet -> Italija 1 -> AC Milan

Odaberi gosta:

Nogomet -> Italija 1 -> AC Milan

Datum igranja:

dd.mm.gggg.

Vrijeme igranja:

--:--

Golovi domaćina:

Golovi gosta:

Status utakmice:

Igra se

Submit

Klikom na „**Sportovi, Lige, Klubovi, Utakmice**“ administratoru se prikazuju svi dodani elementi ovisno na što je kliknuo:

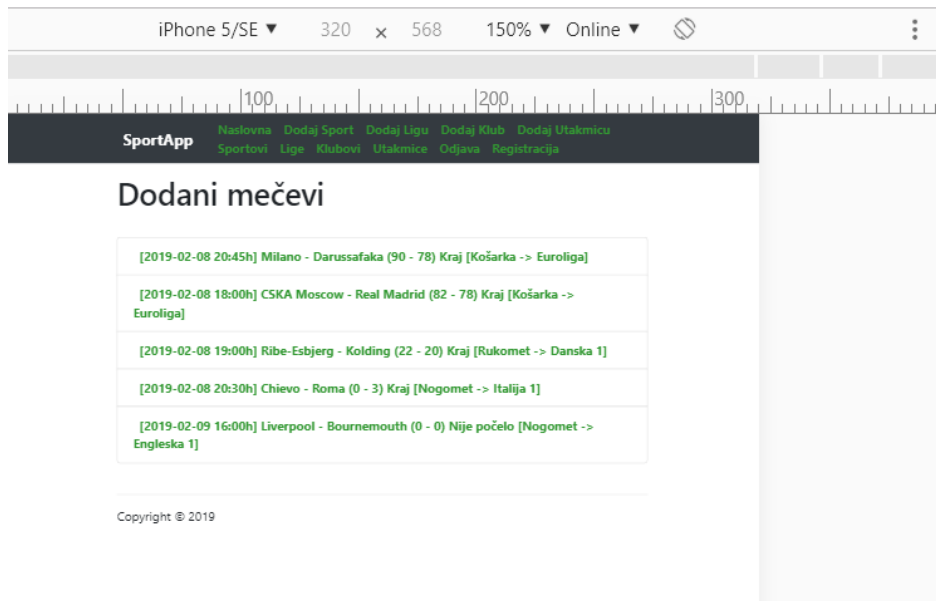
iPhone 5/SE
320
x
568
150%
Online

SportApp
Naslovna
Dodaj Sport
Dodaj Ligu
Dodaj Klub
Dodaj Utakmicu
Sportovi
Lige
Klubovi
Utakmice
Odjava
Registracija

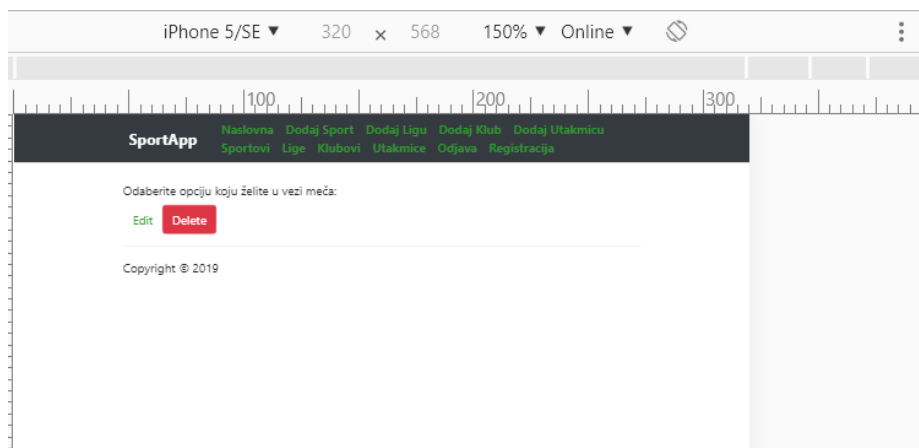
Dodane lige

Italija 1 (Nogomet)
NBA (Košarka)
Španjolska 1 (Nogomet)
SEHA Liga (Rukomet)
Engleska 1 (Nogomet)
Liga Prvaka (Nogomet)
Liga Prvaka (Rukomet)
Hrvatska 1 (Nogomet)
Nizozemska 1 (Nogomet)
Danska 1 (Rukomet)
Euroliga (Košarka)

Copyright © 2019



Na svaki od ovih elemenata koji je ispisan može se kliknuti i onda se pojavi prozor sa 2 opcije:



3.2.2 Brisanje podataka iz baze – „Delete“

Klikom na „Delete“ taj element se briše iz baze podataka što je omogućemo ovim dijelom koda:

```
$(document).ready(function() {
    $('.delete-league').on('click', function(e) {
        $target = $(e.target);
        const id = $target.attr('data-id');

        $.ajax({
            type: 'DELETE',
```

```

        url: '/leagues/'+id,
        success: function(response){
            alert('Deleting League');
            window.location.href='/lige';
        },
        error: function(err){
            console.log(err);
        }
    });
});
});

```

```

//Delete League
router.delete('/:id', function(req, res){

    if(!req.user._id){
        res.status(500).send();
    }

    let query = { _id:req.params.id }

    League.findById(req.params.id, function(err, league){
        //if(league.author != req.user._id){
            // res.status(500).send();
        //}else{
            League.remove(query, function(err){
                if(err){
                    console.log(err)
                }
                res.send('Success');
            });
        //}
    });
});
});

```

3.2.3 Ažuriranje podataka u bazi – „Update“

Klikom na „Edit“ može se ažurirati kolekcije što je omogućemo ovim dijelom koda:

```
//Update Submit post route
router.post('/edit/:id', function(req, res){
    let league = {};
    league.id = req.body.id;
    league.name = req.body.name;
    let query = {_id:req.params.id}
    League.update(query, league, function(err){
        if(err){
            console.log(err);
            return;
        }else{
            req.flash('success', 'League Updated');
            res.redirect('/');
        }
    });
});
```

Zaslon za ažuriranje meča izgleda ovako:

The screenshot displays a mobile web interface for a sports application. The browser's status bar at the top indicates 'iPhone 5/SE' with a resolution of 320x568 and a 150% zoom level. The application's header is dark with the 'SportApp' logo and a series of navigation links: 'Naslovna', 'Dodaj Sport', 'Dodaj Ligu', 'Dodaj Klub', 'Dodaj Utakmicu', 'Sportovi', 'Lige', 'Klubovi', 'Utakmice', 'Odjava', and 'Registracija'. The main section is titled 'Edit Match' and contains a form for updating match details. The form fields are as follows:

- Id meča:
- Autor meča:
- Sport:
- Liga:
- Domaćin:
- Gost:
- Datum igranja:
- Vrijeme igranja:
- Golovi domaćina:
- Golovi gosta:
- Status meča:

A blue 'Submit' button is located at the bottom of the form. The footer of the page shows 'Copyright © 2019'.

3.3 Registracija korisnika – „Passport.js“

Za registraciju korisnika u bazu MongoDB korišten je Passport.js. To je middleware za provjeru autentičnosti za Node.js. Fleksibilan je i modularan i može se ubaciti u bilo koju web aplikaciju temeljenu na Expressu. Sveobuhvatan skup strategija podržava provjeru autentičnosti pomoću korisničkog imena i zaporke. Njegov kod se može pronaći na ovom linku: <http://www.passport.org/docs/>.

Primjer koda dodanog u aplikaciju:

```
const LocalStrategy = require('passport-local').Strategy;
module.exports = function(passport) {
  //Local Strategy - implementation
  passport.use(new LocalStrategy(function(username, password, done) {
    //Match Username
    let query = {username:username};

    User.findOne(query, function(err, user) {
      if(err) throw err;
      if(!user) {
        return done(null, false, {message: 'No user found'});
      }
      bcrypt.compare(password, user.password, function(err,
isMatch) {
        if(err) throw err;
        if(isMatch) {
          return done(null, user);
        } else {
          return done(null, false, {message: 'Wrong
password'});
        }
      });
    });
  }));
});
```

3.4 Kriptiranje lozinke – „bcryptjs“

Zaporke koje se spremaju u bazu ove aplikacije su kriptirane. A to je omogućeno sa ovim dijelom koda:

```
const bcrypt = require('bcryptjs');

bcrypt.genSalt(10, function(err, salt){
    bcrypt.hash(newUser.password, salt, function(err, hash){
        if(err){
            console.log(err);
        }
        newUser.password = hash;

        newUser.save(function(err){
            if(err){
                console.log(err);
                return;
            }else{
                req.flash('success', 'You are now registered and
can log in')

                res.redirect('/users/login');
            }
        });
    });
});
```

3.5 Autoincrement ID

U dodavanju sporta, lige, kluba ili meča ID se automatski inkrementira s obzirom na bazu. Gleda se zadnji ID u bazi, a to je ujedno i najveći ID pa kad želimo dodati sljedeći element on se inkrementira uvijek za 1 s obzirom na zadnji ID u bazi. To se dobije ovim kodom:

```
block content
  h1 #{title}
```

```
- var maxi = 0;

each league, i in leagues
    if league.id > maxi, maxi=league.id
br
form(method='POST', action='/leagues/add')
    br
    h1 Dodaj ligu
    br
    #form-group
        label ID lige:
        input.form-control(name='id', type='text', value= maxi+1)
    #form-group
        label Ime Lige:
        input.form-control(name='name', type='text')
```