

Unixprom dooo

Web orijentisane tehnologije i sistemi
Aleksa Atanacković IT14/2016

Opis:

Web aplikacija, Unixprom doo, internet prodavnica i u kojoj korisnik može dodavati proizvode u svoju korpu, takođe videti osnovne podatke o firmi kao i poslati svoje podatke odnosno aplicirati za posao u firmi na čijem se sajtu nalazi.

Tehnologije: Angular, Express, Mongoose, NodeJs, Bootstrap

Autentifikacija

Registracija novog korisnika

Unixprom d.o.o.

[Home](#) [O nama](#) [Karijera](#) [Meni](#) ▼

[Prijava](#) [Registracija](#)

First Name

Last Name

Mail

Password

Submit

```

TS registration.component.ts • TS auth.service.ts <> header.component.html <> app.component.ts
14 providers: [AuthService]
15 })
16 export class RegistrationComponent implements OnInit {
17
18     myForm: FormGroup;
19
20     constructor(private authService: AuthService, private router: Router) { }
21
22
23     registration(){
24
25         let newUser: User = {
26             firstName: this.myForm.value.firstName,
27             lastName: this.myForm.value.lastName,
28             isAdmin: false,
29             email: this.myForm.value.email,
30             password: this.myForm.value.password
31         };
32
33
34         this.authService.registerUser(newUser)
35             .subscribe(data => {
36                 if (data.msg == 'Uspesno logovanje!'){
37                     localStorage.setItem('token', data.token);
38                     localStorage.setItem('userId', data.userId);
39                     this.router.navigateByUrl('/');
40                 }
41                 else if (data.msg == 'Losa lozinka'){
42                     alert('Lozinka pogresna!');
43                 }
44                 else if (data.msg == 'Ne postoji'){
45                     alert('Korisnik sa tim emailom vec postoji u bazi');
46                 }
47                 else{
48                     alert('Neuspesno logovanje');
49                 }
50
51             });
52
53     }

```

```

registerUser(newUser){
    let headers = new Headers();
    headers.append('content-Type', 'application/json');
    return this.http
        .post('http://localhost:3000/api/user', newUser, {headers: headers}).pipe(
            map(res => res.json()));
}

```

Dugme Submit hendluje metodu registration() u registration.component.ts gde se uzimaju podaci iz forme i šalju preko auth.service-a u user.js (routes foder). Tu vršimo proveru da li korisnik sa tim jedinstvenim mejlom već postoji u bazi. Ako se radi o novom korisniku, ubacujemo ga u bazu, kriptujemo mu lozinku, dodeljujemo mu token i logujemo.

```
router.post('/user', (req, res, next) => {
  User.findOne({email: req.body.email}, function(err, user){
    if(err){
      return res.json({msg: 'Lose'});
    }
    else if (user){
      return res.json({msg: 'Ne postoji'});
    }
  });
  let newUser = new User({
    firstName: req.body.firstName,
    lastName: req.body.lastName,
    isAdmin: req.body.isAdmin,
    email: req.body.email,
    password: bcrypt.hashSync(req.body.password,10)
  });

  newUser.save((err, user)=>{
    if(err)
      return res.json(err);
    else{
      var token = jwt.sign({user: user}, 'secret', {expiresIn: 5000});
      return res.json({msg: 'Uspesno logovanje!',
        token: token,
        userId: user._id});
    }
  });
});
```

Login

Unixprom doo

[Home](#) [O nama](#) [Karijera](#) [Meni](#) ▾

[Prijava](#) [Registracija](#)

E-Mail

Password

Prijavi se

```
TS auth.service.ts • TS login.component.ts x <> header.component.html <> app.component.html
20 ngOnInit() {
21
22     this.myForm = new FormGroup({
23       email: new FormControl(null, [
24         Validators.required,
25         Validators.pattern("[a-z0-9!#$%&*+/?^_`{|}~-]+(?:\\. [a-z0-9!#$%&*+/?^_`{|}~-]+)*@(?:[a-z0-9](?:[a-z0-9-]*[a-z0-9])?\\.)+[a-z0-9](?:[a-z0-9-]*[a-z0-9])?")
26       ]),
27       password: new FormControl(null, Validators.required)
28     });
29   }
30
31   onSubmit(){
32     let user: User={
33       email: this.myForm.value.email,
34       password: this.myForm.value.password
35     };
36
37     this.authService.signin(user)
38       .subscribe(data => {
39         if (data.msg == 'Uspesno logovanje!'){
40           localStorage.setItem('token', data.token);
41           localStorage.setItem('userId', data.userId);
42           this.router.navigateByUrl('/');
43         }
44         else if (data.msg == 'Losa lozinka'){
45           alert('Lozinka pogresna!');
46         }
47         else if (data.msg == 'Ne postoji'){
48           alert('Nepostojeci korisnik');
49         }
50         else{
51           alert('Neuspesno logovanje');
52         }
53       })
54   }
```

```

router.post('/signin', (req, res, next)=>{
  User.findOne({email: req.body.email}, function(err, user){
    if(err){
      res.json({msg: 'Lose'});
    }
    else if (!user){
      res.json({msg: 'Ne postoji'});
    }

    else if (!bcrypt.compareSync(req.body.password, user.password)){
      res.json({msg: 'Losa lozinka'});
    }

    else{
      var token = jwt.sign({user: user}, 'secret', {expiresIn: 5000});
      res.json({msg: 'Uspesno logovanje!',
        token: token,
        userId: user._id});
    }
  });
});

```

Dugme Prijavi se hendluje metodu onSubmit() u login.component.ts gde se uzimaju podaci iz forme i šalju preko auth.service-a u user.js (routes fodler). Tu vršimo proveru da li korisnik sa tim jedinstvenim mejlom već postoji u bazi. Ako ne postoji vratice se alert sa greskom koja je nastala (pogresan mejl ili lozinka). U suprotnom korisnik će se uspešno ulogovati.

CRUD

Create

Delete	Izmeni
--------	--------

Delete	Izmeni
--------	--------

Delete	Izmeni
--------	--------

Add item

Name:

Quantity:

Price: (rsd)

Category:

Image:

Add

U slučaju da smo se ulogovali kao admin klikom na dugme Add možemo dodati novi proizvod u ponudu (bazu).

```
// adminovo dodavanje
addItem(form){
  let newItem: Item = {
    name: form.value.name,
    quantity: form.value.quantity,
    price: form.value.price,
    bought: false,
    category: form.value.category,
    image: form.value.image
  }
  this.dataservice.addShoppingItem(newItem)
  .subscribe(item =>{
    this.getItems();
  });
  form.reset();
}
```

Pravimo novi model klase odnosno objekat klase Item i saljemo ga metodi addShoppingItem klase DataService.

```

addShoppingItem(newItem){
  let headers = new Headers();
  headers.append('content-Type', 'application/json');
  return this.http
    .post('http://localhost:3000/api/item', newItem, {headers: headers}).pipe(
      map(res => res.json())
    );
}

```

```


26 router.post('/item', (req, res, next) => {
27   let newShoppingItem = new Item({
28     name: req.body.name,
29     quantity: req.body.quantity,
30     price: req.body.price,
31     bought: req.body.bought,
32     category: req.body.category,
33     image: req.body.image
34   });
35
36   newShoppingItem.save((err, item)=>{
37     if(err)
38       res.json(err);
39     else
40       res.json({msg: 'Item added to db :' + item});
41   });
42 });

```


Na request argument koji je prosledjen u app.js skladištimo prosleđene podatke u newShoppingItem i memorišemo novi proizvod u bazu.

Read

Kompletna ponuda
Keramika
Rosfraj
Staklo




Akcijska ponuda
Širok asortiman keramičkih i staklenih proizvoda




Čašica
Kategorija: staklo
89,00 RSD

Delete Izmeni



Šerpa
Kategorija: rosfraj
1279,00 RSD

Delete Izmeni



Soljica
Kategorija: keramika
127,00 RSD

Delete Izmeni

```

63
64
65  getItem(){
66      this.dataservice.getShoppingItems()
67          .subscribe(items => {
68              this.shoppingItemList = items;
69          });
70  }
71

```

```

48  ngOnInit() {
49      this.getItem();
50      this.getItemFromCart();
51  }
52

```

```

@Injectables()
export class DataService {

  constructor(private http: Http) { }

  getShoppingItems(){
    return this.http.get('http://localhost:3000/api/items')
      .pipe(map(res => res.json()));
  }
}








```

```

router.get('/items', function(req, res, next) {
  Item.find((err, items) => {
    if(err){
      res.json();
    } else {
      res.json(items);
    }
  });
});

```

Akcija getitems vraća dva view-a. Prvi se odnosi na stranicu koja prikazuje korisniku to su proizvodi koje može dodati u korpu, a drugi se odnosi na home stranicu koja se prikazuje adminu, kompletna ponuda iz proizvoda koje on može brisati ili izmeniti.

<div>Čašica</div> <div>Kategorija: staklo</div> <div>89,00 RSD</div> <div>Add</div> <div></div>	<div>Šerpa</div> <div>Kategorija: rosfraj</div> <div>1279,00 RSD</div> <div>Add</div> <div></div>	<div>Soljica</div> <div>Kategorija: keramika</div> <div>127,00 RSD</div> <div>Add</div> <div></div>	<div>Korpa </div> <div>Naziv: Kolicina Cena</div> <div>Iznos racuna: 0 rsd</div> <div>Odustani Poruči</div>
<div>Krigla</div> <div>Kategorija: staklo</div> <div>153,00 RSD</div> <div>Add</div> <div></div>	<div>Noz</div> <div>Kategorija: keramika</div> <div>799,00 RSD</div> <div>Add</div> <div></div>	<div>Cediljka</div> <div>Kategorija: rosfraj</div> <div>299,00 RSD</div> <div>Add</div> <div></div>	

Update

Delete

Izmeni

Delete

Izmeni

Delete

Izmeni

Edit item

Name:

Quantity:

Price: (rsd)

Category:

Image:

Save

Odustani

Svaki admin klikom na dugme Izmeni ima mogućnost izmene svojih proizvoda. Poziva se funkcija editItem gde se vrednostima iz forme popunjava promenljiva newItem koja se prosledjuje funkciji updateShoppingItem() pomocu objekta klase DataService. U slučaju da se klikne na dugme odustani dugme izmeni će ponovo biti omogućeno za klikanje a forma će se isprazniti sa podacima prethodno označenog, dok prethodno označeni proizvod neće biti imenjen u bazi.

```

216   editItem(form){
217     let newItem: Item = {
218       _id: this.selectedItem._id,
219       name: form.value.name,
220       quantity: form.value.quantity,
221       price: form.value.price,
222       image: form.value.image,
223       category: form.value.category,
224       bought: false
225     }
226     this.dataservice.updateShoppingItem(newItem)
227       .subscribe(result => {
228         this.getItems();
229       });
230
231     this.toggleForm = !this.toggleForm;
232   }
233
234   showEditForm(item){
235     this.selectedItem = item;
236     this.toggleForm = !this.toggleForm;
237   }
238
239   onOdustani(form){
240
241     this.toggleForm = !this.toggleForm;
242     form.reset();
243   }
244
245

```

```

32   updateShoppingItem(newItem){
33     let headers = new Headers();
34     headers.append('content-Type', 'application/json');
35     return this.http
36       .put('http://localhost:3000/api/item/' + newItem._id, newItem, {headers: headers}).pipe(
37         map(res => res.json())
38       );
39   }
40

```



```

44
45 router.put('/item/:id', (req, res, next)=>{
46   Item.findOneAndUpdate({_id: req.params.id}, {
47     $set:{
48       name: req.body.name,
49       quantity: req.body.quantity,
50       bought: req.body.bought,
51       category: req.body.category,
52       image: req.body.image
53     }
54   },
55
56   function(err, result){
57     if(err){
58       res.json(err);
59     }
60     else
61       res.json(result);
62   });
63
64 });
65

```

Pomoću mongoose funkcije `findOneAndUpdate` tražimo proizvod i setujemo njegove promenljive na vrednosti prosleđene promenljive.

Delete(Cancel)

Implementirano je i brisanje proizvoda iz baze. Ova funkcionalnost je omogućena samo adminu.



Klikom na dugme delete pozvacemo u pozadini funkciju deleteltem.

```

203 deleteItem(id){
204   this.dataservice.deleteShopingItem(id)
205   .subscribe(data =>{
206     if (data.n == 1){
207       for(var i=0; i<this.shoppingItemList.length;i++){
208         if (id== this.shoppingItemList[i]._id){
209           this.shoppingItemList.splice(i,1);
210         }
211       }
212     }
213   });
214 }
215

```

Funkciji se prosledjuje id proizvoda koji se briše koji se prosleđuje dalje servisu i app.js-u, na osnovu koga će proizvod biti nađen u bazi i krajnje izmenjen (slike ispod).

```

26
27 deleteShopingItem(id){
28   return this.http.delete('http://localhost:3000/api/item/' + id)
29   .pipe(map(res => res.json()));
30 }
31

```

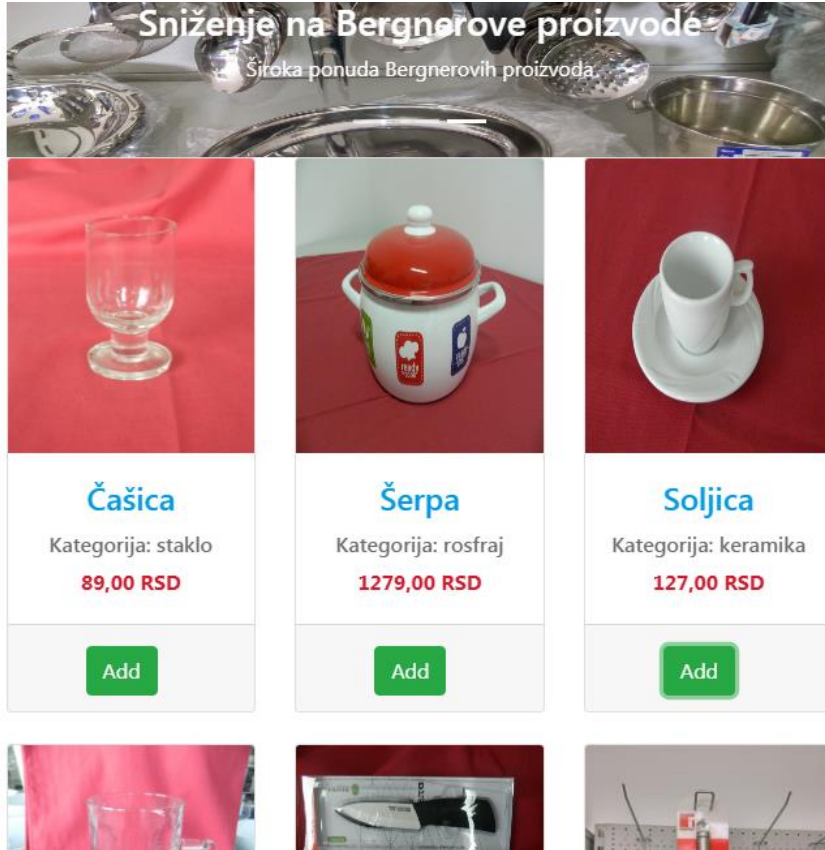
```

66 router.delete('/:id', (req, res, next)=>{
67   Item.remove({_id: req.params.id}, function(err, result){
68     if(err){
69       res.json(err);
70     }
71     else
72       res.json(result);
73   });
74 });
75

```

Add funkcionalnost

Keramika
Rosfraj
Staklo



Klikom na Add dugme postoje 2 moguća događaja. Ako korisnik nije prijavljen poslaće ga na stranicu Prijava (login).

U slučaju da je korisnik prijavljen pozvaće se funkcija `addItemToCart(item)` gde se kao argument prosledjuje proizvod na koji je kliknuto.

E-Mail

Password

Prijavi se

```

96
97   addItemToCart(item){
98     this.dataservice.getPr(item)
99     .subscribe(poruka =>{
100       if (poruka.msg == 'Not found'){
101         this.dataservice.addProizvoda(item)
102         .subscribe(item =>{
103           console.log(item);
104           this.getItemFromCart();
105         });
106       } else {
107         alert("Proizvod ste vec dodali! Kolicinu mozete povecati direktno u vashoj korpi.");
108       }
109     });
110
111   }

```

```

46
47   addProizvoda(newItem){
48     let headers = new Headers();
49     headers.append('content-Type', 'application/json');
50     return this.http
51       .post('http://localhost:3000/api/proizvod', newItem, {headers: headers}).pipe(
52         map(res => res.json())
53       );
54   }
55

```

```

31 //dodaj u korpu
32 router.post('/proizvod', (req, res, next) => {
33   let newItemInChart = new Proizvodi({
34     _id: req.body._id,
35     name: req.body.name,
36     bought: req.body.bought,
37     quantity: req.body.quantity,
38     price: req.body.price,
39   });
40
41   newItemInChart.save((err, item) => {
42     if(err)
43       res.json(err);
44     else
45       res.json({msg: 'Item added to cart:' + item});
46   });
47 });

```

U kolekciju proizvodi dodaje se novi proizvod koji se nalazi u korpu. Proizvodi predstavlja privremenu kolekciju pomoću koje predstavljamo proizvode dodate u korpu. Privremena joj je epitet zato što nakon potvrde korpe svi privremeni članovi ove kolekcije se brišu.

Funkcionalnos + (povećaj količinu proizvoda u korpi)

Klikom na dugme + (plus) dodajemo količinu proizvoda za koji je kliknuto dato dugme.

Korpa

Naziv:

Kolicina

Cena

Soljica

5

635 rsd

-

+

Serpa

1

1279 rsd

-

+

Čašica

1

89 rsd

-

+

Iznos racuna: 2003 rsd

Odustani

Poruči

Klikom na dugme + poziva se funkcija `changeItemFromCart()` gde se prosledjenom proizvodu (item) povecavava properti `quantity` za jedan. Dok se pri kliktanju dugmeta – (minus) poziva funkcija `deleteItemFromCart` kojoj se takodje prosleđuje proizvod (item) gde se vrši provera u odnosu na koju dugme može imati ulogu da izbriše ili smanji količinu proizvoda.

```

112
113     deleteItemFromCart(item){
114
115         if (item.quantity > 1){
116             var br;
117             br = parseInt(item.quantity)-1;
118
119             let newItem: Proizvodi = {
120                 _id: item._id,
121                 quantity: br
122             }
123             this.dataservice.updateProizvoda(newItem)
124                 .subscribe(result => {
125                 this.getItemFromCart();
126             });
127         }
128         else{
129             this.dataservice.deleteProizvoda(item._id)
130                 .subscribe(data => {
131                 this.getItemFromCart();
132             });
133         }
134     }

```

```

135
136     changeItemFromCart(item){
137         var br;
138         br = parseInt(item.quantity)+1;
139
140         let newItem: Proizvodi = {
141             _id: item._id,
142             quantity: br
143         }
144         this.dataservice.updateProizvoda(newItem)
145             .subscribe(result => {
146             this.getItemFromCart();
147         });
148     }

```


Odustani funkcionalnost

Korpa 

Naziv: Kolicina Cena

Soljica 5 635 rsd

-

+

Šerpa 1 1279 rsd

-

+

Čašica 1 89 rsd

-

+

Iznos racuna: 2003 rsd

Odustani

Poruči

Klikom na Odustani dugme poziva se funkcija onOdustaniOdKupovine() gde se takođe vrši provera da li postoji neka stavka u korpi, ako postoji poziva se funkcija deleteAllTempProizvoda() iz dataservice-a, a niz pomoću koga izlistavamo proizvode stavljamo na null.

```
83  
84   onOdustaniOdKupovine(){  
85       if (this.chartList.length<1){  
86           return alert('Korpa je prazna!');  
87       }  
88       this.dataservice.deleteAllTempProizvoda()  
89       .subscribe(data =>{  
90           this.chartList=[];  
91           this.getItemFromCart();  
92       });  
93       this.chartList=[];  
94       this.getItemFromCart();  
95   }  
96
```

```
TS data.service.ts    TS asortiman-list.component.ts    JS proizvodi.js x
48
49 //izbrisi iz korpe
50 router.delete('/proizvod/:id', (req, res, next)=>{
51   Proizvodi.remove({_id: req.params.id}, function(err, result){
52     if(err){
53       res.json(err);
54     }
55     else
56       res.json(result);
57   });
58 });
59
60 router.delete('/proizvodi', (req, res, next)=>{
61   Proizvodi.remove({}, function(err, result){
62     if(err){
63       res.json(err);
64     }
65     else
66       res.json(result);
67   });
68 });
69
70 router.put('/proizvod/:id', (req, res, next)=>{
71   Proizvodi.findOneAndUpdate({_id: req.params.id}, {
72     $set:{
73       quantity: req.body.quantity ,
74     }
75   },
76   function(err, result){
77     if(err){
78       res.json(err);
79     }
80     else
81       res.json(result);
82   });
83 });
84
85 });
86
```

Poruči funkcionalnost

Klikom na Poruči dugme poziva se funkcija `onPoruči()` gde se takođe vrši provera da li postoji neka stavka u korpi, ako postoji poziva se funkcija `addCart()` iz `dataservice-a`, podešavamo promenljivu `event` na trenutno vreme u Srbiji, a u niz `itemsFromCart` smeštamo proizvode iz niza po imenu `chartList`. Pravimo dakle promenljivu tipa `Cart` i prosledjujemo je kao argument funkciji `addCart()`. Nakon uspesne kupovine brišemo sve proizvode iz privremene kolekcije `Proizvodi` i dodajemo u korisnikov niz `cartova` id kupovine koju je obavio.

```

149
150     onPoruci(form){
151         if (this.chartList.length<1){
152             return alert('Korpa je prazna!');
153         }
154         this.event.setUTCHours(this.event.getUTCHours()+2);
155
156         this.chartList.forEach(e1 => {
157             this.itemsFromCart.push(e1);
158         });
159
160
161         let newCart: Cart = {
162             userId: this.dataservice.getUserId(),
163             items: this.itemsFromCart,
164             napomena: 'Proslo',
165             date: (this.event.toLocaleString('sr-Latn', { timeZone: 'UTC' }))
166         }
167
168         this.dataservice.addCart(newCart)
169         .subscribe(data =>{
170             if (data.msg == 'Cart added to db: '){
171                 this.onOdustaniOdKupovine();
172                 alert('Uspesna kupovina');
173             }
174             else{
175                 alert('Neuspesna kupovina');
176             }
177         });
178
179     }

```

```

router.post('/cart', (req, res, next) => {

  User.findById(req.body.userId, (err, user) => {
    if(err){
      return res.json(err);
    }
    else if(!user){
      return res.json({msg:"User not found!!"});
    }

    var itemz = [];
    var niz = req.body.items;

    for(var i=0;i<niz.length;i++){
      itemz.push(niz[i].name + ', kolicina ' + niz[i].quantity + ', cena/komad: ' + niz[i].price);
    }

    var newCart = new Cart({
      user: user,
      items: itemz,
      napomena: req.body.napomena,
      date: req.body.date,
    });

    newCart.save((err, cart)=>{
      if(err)
        return res.json(err);
      else{
        user.carts.push(cart);
        user.save();
        return res.json({msg: 'Cart added to db: '});
      }
    });

  });
});

```


Apliciraj za posao

U slučaju da ste nezaposleni pripremili smo i funkcionalnost da konkurišete za posao u našem kolektivu.

```
28 openFormModal() {
29
30   if(this.authService.isLoggedIn()){
31     const modalRef = this.modalService.open(FormModalComponent);
32
33     modalRef.result.then((result) => {
34
35       this.dataservice.addJob(result)
36       .subscribe(item =>{
37         console.log('Job sent' + item);
38       });
39     }).catch((error) => {
40       console.log(error);
41     });
42   }
43   else
```

Posao za studente

Jedan od najefikasnijih načina reklamiranja, podelom flajera, koji sigurno dospevaju u ruke naših građana, obaveštavajući sadašnje i buduće klijente o onome čime se naša firma bavi. Novim projektima ili proizvodima, kao i raznim pogodnostima poput popusta i akcijskih predloga. Studenti koji žele da zarade klik na prijavi



Apliciraj

Prezime

Email

Pol:

☐ Musko

☐ Žensko

☐ Neopredeljen/a/o

Godina rođenja

Strucna sprema

Motivaciona poruka

*Nepotpune prijave necemo uzimati u obrazlozenje

Odustani Save

Posao za vozače

ercijala je po pravilu jedan jako dinamičan, rovan ali istovremeno i odgovoran posao. gim rečima, to je posrednik između svoje ne firme koju predstavlja, i druge firme sa njom ugovara prodaju. Takođe, radi na onalaženju novih klijenata, obezbeđuje kvalitetan transport robe.



Apliciraj

Nakon unosa podataka postoje 2 opcije, odustani i save, gde se klikom na odustani zatvara modalni dialog, a klikom na save salje forma save dugme je dostupno za klik tek popunjavanjem svih polja.

```
33   closeModal() {
34     | this.activeModal.close('Modal Closed');
35   }
36   eventHandler(event: any){
37     | this.vrednostGodista = event.target.value;
38   }
39   genderHandler(event: any){
40     | this.pol = event.target.value;
41   }
42   fakultetChoice(event: any){
43     | this.fakultet = event.target.value;
44   }
45   submitForm(form) {
46     | let newJob: Job = {
47       | firstName: form.value.nameStud,
48       | lastName: form.value.surnameStud,
49       | email: form.value.email,
50       | gender: this.pol,
51       | faculty: this.fakultet,
52       | birth: this.vrednostGodista ,
53       | message: form.value.message
54     | }
55     | this.activeModal.close(newJob);
56   }
57
58   getElements(){
59     | this.godine.push('Izaberite..');
60     | for (var i = 1955 ; i < 2005; i++){
61     |   |
62     |   | this.godine.push(i.toString());
63     | }
64 }
```



```

8   router.get('/jobs', function(req, res, next) {
9       Job.find((err, users) => {
10           if(err){
11               res.json();
12           } else {
13               res.json(users);
14           }
15       });
16   });
17
18   router.post('/job', (req, res, next) => {
19       let newJob = new Job({
20           firstName: req.body.firstName,
21           lastName: req.body.lastName,
22           email: req.body.email,
23           gender: req.body.gender,
24           birth: req.body.birth,
25           faculty: req.body.faculty,
26           message: req.body.message,
27       });
28
29       newJob.save((err, user) => {
30           if(err)
31               res.json(err);
32           else
33               res.json({msg: 'Job added to db: ' + user});
34       });
35   });
36
37   router.delete('/job/:id', (req, res, next) => {
38       Job.remove({_id: req.params.id}, function(err, result){
39           if(err){
40               return res.json(err);
41           }
42           else
43               return res.json(result);
44       });
45   });

```

Admin nema mogućnost da konkuriše za posao ali ima mogućnost da obriše kandidata čiju aplikaciju ne smatra za adekvatnom.

Struktura fajlova

