Elektrotehnički fakultet u Sarajevu

Računarstvo i informatika

Web tehnologije

Ak. god 2023/24

Projekat - Spirala 3

**Zadatak 1. [4B]**

Reorganizirajte projekat tako da se sve statičke datoteke nalaze u *public* folderu. Svaka stranica do sada se treba moći otvoriti kroz *express* server kojeg ćete implementirati u datoteci *index.js*. Datoteku *index.js* postavite u početni direktorij projekta. Svaka stranica na projektu se mora otvarati isključivo preko linka <http://localhost:3000/nazivStranice.html>. Svaka stranica mora imati stil kao i do sada, a funkcionalnosti sa prethodnih spirala i dalje trebaju raditi.

Podaci koji su potrebni web aplikaciji su smješteni u datotekama: *korisnici.json* i *nekretnine.json*. Ove dvije datoteke se nalaze u folderu *data* i imaju sljedeći izgled:

**korisnici.json**

| [  {  "id": 1,  "ime": "Neko",  "prezime": "Nekic",  "username": "username1",  "password": "hashPassworda1"  },  {  "id": 2,  "ime": "Neko2",  "prezime": "Nekic2",  "username": "username2",  "password": "hashPassworda2"  } ] |
| --- |

**nekretnine.json**

| [  {  "id": 1,  "tip\_nekretnine": "Stan",  "naziv": "Useljiv stan Sarajevo",  "kvadratura": 58,  "cijena": 232000,  "tip\_grijanja": "plin",  "lokacija": "Novo Sarajevo",  "godina\_izgradnje": 2019,  "datum\_objave": "01.10.2023.",  "opis": "Sociis natoque penatibus.",  "upiti": [  {  "korisnik\_id": 1,  "tekst\_upita": "Nullam eu pede mollis pretium."  },  {  "korisnik\_id": 2,  "tekst\_upita": "Phasellus viverra nulla."  }  ]  },  {  "id": 2,  "tip\_nekretnine": "Poslovni prostor",  "naziv": "Mali poslovni prostor",  "kvadratura": 20,  "cijena": 70000,  "tip\_grijanja": "struja",  "lokacija": "Centar",  "godina\_izgradnje": 2005,  "datum\_objave": "20.08.2023.",  "opis": "Magnis dis parturient montes.",  "upiti": [  {  "korisnik\_id": 2,  "tekst\_upita": "Integer tincidunt."  }  ]  } ] |
| --- |

Obratite pažnju da su šifre korisnika (*password*) spremljene kao *hash* vrijednosti. Za kreiranje *hash*-a *password*-a, koristite *bcrypt* paket:

| bcrypt.hash(plainTextPassword, 10, function(err, hash) {     // hash šifre imate ovdje }); |
| --- |

Modificirajte stranice sa prethodnih spirala na sljedeći način:

* Na stranicu *prijava.html* (koja je, do sada, bila prazna) dodajte formu sa poljima *username* i *password* i dugme “Prijava”. Klikom na ovo dugme se korisnik pokušava prijaviti u web aplikaciju.
* Kada je korisnik prijavljen u aplikaciji, tada umjesto “Prijava” stranice u glavnom meniju treba biti prikazano dugme “Odjava” (izgled nije bitan, već je samo bitno da je riječ o dugmetu).
* Kada je korisnik prijavljen u aplikaciji, tada u meniju trebaju biti opcije “Profil”, “Nekretnine”, “Detalji” i “Odjava”. Ukoliko korisnik nije prijavljen, tada u meniju trebaju biti opcije “Nekretnine”, “Detalji” i “Prijava”.
* Podaci na stranici “Detalji” nisu bitni na ovoj spirali (tj. na ovoj stranici možete ostaviti hardkodirane bilo kakve podatke).

Implementirajte sljedeće rute na web serveru:

* **Ruta**:/login
  + **Metoda**: POST
  + **Format tijela zahtjeva**: JSON
  + **Dijelovi tijela zahtjeva**:
    - username (tipa string) - *username* korisnika koji se pokušava prijaviti u aplikaciju.
    - password (tipa string) - *password* korisnika koji se pokušava prijaviti u aplikaciju.
  + **Opis rute**: provjerava da li korisnik postoji i da li je ispravna šifra na osnovu podataka u datoteci *korisnici.json*. Ukoliko je korisnik ispravno unio podatke, tada u sesiju upišite *username* korisnika i kao odgovor vratite {“poruka”:”Uspješna prijava”}. Ako prijava nije bila uspješna, tada vratite JSON {“greska”:”Neuspješna prijava”}. Odgovor uvijek mora biti u JSON formatu.
  + **Odgovori**:
    - **Statusni kod**: 200
      * **Tijelo odgovora**: {“poruka”:”Uspješna prijava”}
    - **Statusni kod**: 401
      * **Tijelo odgovora**: {“greska”:”Neuspješna prijava”}
* **Ruta**:/logout
  + **Metoda**: POST
  + **Format tijela zahtjeva**: zahtjev nema tijelo
  + **Opis rute**: poziv ove rute briše sve informacije iz sesije.
  + **Odgovori**:
    - **Statusni kod**: 200
      * **TIjelo odgovora**: {“poruka”:”Uspješno ste se odjavili”}
    - **Statusni kod**: 401
      * **Tijelo odgovora**: {“greska”:”Neautorizovan pristup”}
* **Ruta**:/korisnik
  + **Metoda**: GET
  + **Opis rute**: ova ruta vraća podatke o loginovanom korisniku. Podaci se vraćaju u JSON formatu. Ukoliko korisnik nije loginovan, vratite JSON {“greska”:”Neautorizovan pristup”}.
  + **Odgovori**:
    - **Statusni kod**: 200
      * **TIjelo odgovora**: JSON objekat sa svim podacima o korisniku (*id, ime, prezime, username, password~~, slika~~*).
    - **Statusni kod**: 401
      * **Tijelo odgovora**: {“greska”:”Neautorizovan pristup”}
* **Ruta**:/upit
  + **Metoda**: POST
  + **Format tijela zahtjeva**: JSON
  + **Dijelovi tijela zahtjeva**:
    - nekretnina\_id (tipa integer) - *id* nekretnine za koju korisnik želi kreirati upit.
    - tekst\_upita (tipa string) - sadržaj korisničkog upita
  + **Opis rute**: omogućava loginovanom korisniku da postavi upit za određenu nekretninu. Ukoliko korisnik nije loginovan vratite {“greska”:”Neautorizovan pristup”}. Ako je korisnik loginovan, tada u datoteci *korisnici.json* pronađite *id* loginovanog korisnika a u *nekretnine.json* pronađite nekretninu čiji je *id* jednak *nekretnina\_id* iz tijela zahtjeva. Ukoliko takva nekretnina ne postoji, onda vratite {“greska”:”Nekretnina sa id-em ${nekretnina\_id} ne postoji”}. Ako je nekretnina uspješno pronađena, u niz *upiti* pronađene nekretnine, dodajte novi upit sa podacima o *id*-u korisnika koji je postavio taj upit i tekstom koji je sadržan u *tekst\_upita* u tijelu zahtjeva. Kada je zahtjev uspješno završen, vratite poruku o tome {“poruka”:”Upit je uspješno dodan”}.
  + **Odgovori**:
    - **Statusni kod**: 200
      * **TIjelo odgovora**: {“poruka”:”Upit je uspješno dodan”}
    - **Statusni kod**: 400
      * **Tijelo odgovora**: {“greska”:”Nekretnina sa id-em ${nekretnina\_id} ne postoji”}
    - **Statusni kod**: 401
      * **Tijelo odgovora**: {“greska”:”Neautorizovan pristup”}
* **Ruta**:/korisnik
  + **Metoda**: PUT
  + **Format tijela zahtjeva**: JSON
  + **Dijelovi tijela zahtjeva**:
    - ime (tipa string) - ažurirano ime korisnika.
    - prezime (tipa string) - ažuriran prezime korisnika
    - username (tipa string) - ažurirani *username* korisnika
    - password (tipa string) - ažurirani *password* korisnika
  + **Opis rute**: omogućava loginovanom korisniku da promijeni podatke svog profila. Ukoliko korisnik nije loginovan vratite {“greska”:”Neautorizovan pristup”}. Ako je korisnik loginovan, datoteku *korisnici.json* ažurirajte sa novim podacima korisnika. Obratite pažnju da tijelo zahtjeva ne mora uvijek imati sva svojstva, odnosno mogu se desiti situacije kada korisnik želi ažurirati samo dio svojih ličnih podataka a ne sve. Kada je zahtjev uspješno završen, vratite poruku o tome {“poruka”:”Podaci su uspješno ažurirani”}
  + **Odgovori**:
    - **Statusni kod**: 200
      * **TIjelo odgovora**: {“poruka”:”Podaci su uspješno ažurirani”}
    - **Statusni kod**: 401
      * **Tijelo odgovora**: {“greska”:”Neautorizovan pristup”}
* **Ruta**:/nekretnine
  + **Metoda**: GET
  + **Opis rute**: vraća sve nekretnine iz datoteke *nekretnine.json*.
  + **Odgovori**:
    - **Statusni kod**: 200
      * **TIjelo odgovora**: JSON sa svim nekretninama iz datoteke *nekretnine.json*.

Za sve rute koje imaju tijelo, pretpostavite da je ono u ispravnom obliku, odnosno nije potrebno validirati tijelo zahtjeva.

**Zadatak 2. [3B]**

Modul *PoziviAjax* se koristi za spajanje *frontend*-a i *backend*-a aplikacije. Ovaj modul ima sljedeći izgled:

| const PoziviAjax = (() => {   // fnCallback se u svim metodama poziva kada stigne  // odgovor sa servera putem Ajax-a  // svaki callback kao parametre ima error i data,  // error je null ako je status 200 i data je tijelo odgovora  // ako postoji greška, poruka se prosljeđuje u error parametru  // callback-a, a data je tada null   // vraća korisnika koji je trenutno prijavljen na sistem  function impl\_getKorisnik(fnCallback) {   }   // ažurira podatke loginovanog korisnika  function impl\_putKorisnik(noviPodaci, fnCallback) {   }   // dodaje novi upit za trenutno loginovanog korisnika  function impl\_postUpit(nekretnina\_id, tekst\_upita, fnCallback) {   }   function impl\_getNekretnine(fnCallback) {   }   function impl\_postLogin(username, password, fnCallback) {   }   function impl\_postLogout(fnCallback) {   }   return {  postLogin: impl\_postLogin,  postLogout: impl\_postLogout,  getKorisnik: impl\_getKorisnik,  putKorisnik: impl\_putKorisnik,  postUpit: impl\_postUpit,  getNekretnine: impl\_getNekretnine  }; })(); |
| --- |

U ovoj datoteci ne smije biti nikakav drugi kod osim koda modula.

Spojite modul sa HTML stranicama na način koji je sličan načinu na koji ste to uradili na prošloj spirali. Ishod svega bi trebao biti da se sada podaci prikupljaju sa servera, tj. da nisu hardkodirani kao što je to bio slučaj na prethodnoj spirali. Sve funkcionalnosti prošle spirale trebaju raditi, ali ćete sada umjesto hardkodiranog spiska nekretnina koristiti spisak nekretnina sa servera, tj. iz datoteke *nekretnine.json* itd. Isto vrijedi i za ostale funkcionalnosti.

Važno je da sve statičke datoteke budu u *public* folderu razdvojene od serverskih datoteka. Povedite računa da putanje do resursa u okviru HTML-a budu relativne.

Rute na serveru moraju biti implementirane na način koji je opisan u postavci. Osim toga, **nije dozvoljeno korištenje sinhronih metoda**. Komunikacija HTML stranica sa web serverom mora biti ostvarena korištenjem modula *PoziviAjax*.

**Zadatak 3. [3B]**

Na početak (vrh) stranice *nekretnine.html* dodajte polja za pretragu po minimalnoj, maksimalnoj cijeni, minimalnoj i maksimalnoj kvadraturi nekretnine. Pored ovih polja dodajte dugme za pretragu. Izgled ovih elemenata odredite sami. Kada korisnik unese vrijednosti u polja i klikne na dugme za filtriranje, uradite filtriranje prema kriterijima koji su proslijeđeni. Ukoliko su sva polja prazna, onda se vraćaju sve nekretnine, tj. radi se filtriranje samo po tipu nekretnine zbog prikaza. U zavisnosti od proslijeđenih kriterija, uradite poziv metode *filtrirajNekretnine* sa prošle spirale sa odgovarajućim parametrom kriterija.

Implementirajte modul **MarketingAjax** i rute **/marketing** koje omogućavaju praćenje preferencija korisnika prilikom korištenja aplikacije. Nakon svakog filtriranja (tj. pretrage) nekretnina, listu id-eva nekretnina koje su rezultat filtriranja pošaljite na rutu **POST /marketing/nekretnine**, a nakon svakog klika na dugme “Detalji” neke od nekretnina, pošaljite zahtjev na rutu **POST /marketing/nekretnina/:id**. Klik na dugme “Detalji” ne otvara detalje date nekretnine, jer taj dio ne trebate implementirati u sklopu ove spirale.

Metode:

* **MarketingAjax.osvjeziPretrage(divNekretnine)**
* **MarketingAjax.osvjeziKlikove(divNekretnine)**

periodično (svakih 500ms) ažuriraju broj pretraga/klikova za svaku od nekretnina koje su prikazane. Broj pretraga i broj klikova se prikazuju u div elementu svake nekretnine. Id-evi tih div-ova su *pretrage-idNekretnine* i *klikovi-idNekretnine* (ispravite funkciju sa spirale 2 za iscrtavanje nekretnina tako da uključuje ove div elemente; stil i izgled odredite sami). Referenca **divNekretnine** je div u kojem se prikazuju sve trenutne nekretnine.

Ako se ne može vratiti broj pretraga/klikova za neku od nekretnina, vrijednosti pretraga/klikova na stranici prikazujte kao što je bilo u prethodnom uspješnom pozivu. Ukoliko nikada nije bio uspješan poziv, onda u tim nekretninama nikako ne prikazujte div-ove *#pretrage-idNekretnine* i *#klikovi-idNekretnine*(dakle, u ovom slučaju ovi div-ovi postoje ali nisu vidljivi). Metode *osvjeziPretrage* i *osvjeziKlikove* samo prvi put na server šalju niz id-eva nekretnina za koje trebate osvježiti pretrage/klikove, a svaki sljedeći put koristite sesiju korisnika da uradite isto.

Ukoliko se izvrši novo filtriranje (tj. ukoliko se promijeni lista nekretnina), trebate pozvati metodu:

**MarketingAjax.novoFiltriranje(listaFiltriranihNekretnina)**

Ovom metodom govorite metodi *osvjeziPretrage* da treba tražiti informacije o pretragama za novu listu nekretnina (parametar metode je lista koja se dobije kao rezultat metode *filtrirajNekretnine* iz modula sa prethodne spirale). Također, na rutu *POST /marketing/nekretnine* šaljete zahtjev i informaciju o tome da je obavljeno jedno filtriranje sa listom id-eva nekretnina, čime se broj pretraga za svaku nekretninu iz liste uvećava za jedan.

Ukoliko se klikne na neku nekretninu (tj. na dugme “Detalji” neke od nekretnina), širinu nekretnine postavite tako da iznosi 500px (a ne 300px kako je prethodno bilo) i pozovite:

**MarketingAjax.klikNekretnina(idNekretnine)**

Pozivom ove metode zaustavlja se osvježavanje za ostale nekretnine, a pokreće osvježavanje samo za trenutnu nekretninu. Također, šalje se zahtjev na rutu POST /marketing/nekretina/:id čime se uvećava broj klikova za tu nekretninu za jedan.

Rute potrebne za ovaj zadatak su:

* **Ruta:** /marketing/nekretnine
  + **Metoda:** POST
  + **Format tijela zahtjeva:** JSON
  + **Primjer tijela zahtjeva:** {"nizNekretnina":[1,5,6,7,12]}
  + **Odgovor:**
    - **Statusni kod**: 200
      * **TIjelo odgovora**: bez tijela odgovora
* **Ruta:** /marketing/nekretnina/:id
  + **Metoda:** POST
  + **Format tijela zahtjeva:** bez tijela zahtjeva
  + **Odgovor:**
    - **Statusni kod**: 200
      * **TIjelo odgovora**: bez tijela odgovora
* **Ruta:** /marketing/osvjezi
  + **Metoda:** POST
  + **Format tijela zahtjeva:** JSON
  + **Primjer tijela zahtjeva:** {"nizNekretnina":[1,5,6,7,12]} - za prvi zahtjev. U slučaju osvježavanja za iste nekretnine, tada se ne šalje tijelo u zahtjevu.
  + **Odgovori:**
    - **Statusni kod**: 200
      * **Tijelo odgovora:** JSON. Primjer:  
        {"nizNekretnina":[{"id":1,"klikovi":100,"pretrage":20},{"id":5,"klikovi":40,"pretrage":2}]} - vraćaju se podaci samo za nekretnine za koje postoji promjena u odnosu na prethodni poziv

Modul implementirajte kao i modul u prošlom zadatku. Datoteka modula neka ima isti naziv kao i sam modul. Sve metode definisane u ovom zadatku moraju biti *public* metode modula. Modul isključivo sadrži metode koje su definisane u zadatku. Povezivanje modula sa stranicom, *event*-e na klik i sl. realizujte u datoteci *nekretnine.js*.

Pitanja vezana za projekat postavljajte na [Piazza forum](https://piazza.com/etf.unsa.ba/fall2023/ri31) folder spirala\_3.

Projekat nastavite raditi u repozitoriju koji ste napravili za prethodne spirale. Rok za rad na spirali je **31.12.2023 do 23:59**. Prethodnu spiralu postavite na *branch* **spirala2**. Aktuelnu spiralu radite na *branch*-u **master**.