

Programsko inženjerstvo
Ak. god. 2020./2021.

WebGym
Dokumentacija, Rev. 1

Grupa: *bugBusters*
Voditelj: *Luka Merćep*

Datum predaje: 13. 11. 2020.

Nastavnik: *Eugen Vušak*

Sadržaj

1 Dnevnik promjena dokumentacije	3
2 Opis projektnog zadatka	4
3 Specifikacija programske potpore	9
3.1 Funkcionalni zahtjevi	9
3.1.1 Obrasci uporabe	12
3.1.2 Sekvencijski dijagrami	26
3.2 Ostali zahtjevi	30
4 Arhitektura i dizajn sustava	31
4.1 Baza podataka	33
4.1.1 Opis tablica	34
4.1.2 Dijagram baze podataka	39
4.2 Dijagram razreda	40
4.3 Dijagram stanja	44
4.4 Dijagram aktivnosti	46
4.5 Dijagram komponenti	48
5 Implementacija i korisničko sučelje	49
5.1 Korištene tehnologije i alati	49
5.2 Ispitivanje programskog rješenja	51
5.2.1 Ispitivanje komponenti	51
5.2.2 Ispitivanje sustava	53
5.3 Dijagram razmještaja	59
5.4 Upute za puštanje u pogon	60
6 Zaključak i budući rad	63
Popis literature	65
Indeks slika i dijagonama	66

Dodatak: Prikaz aktivnosti grupe

67

1. Dnevnik promjena dokumentacije

Kontinuirano osvježavanje

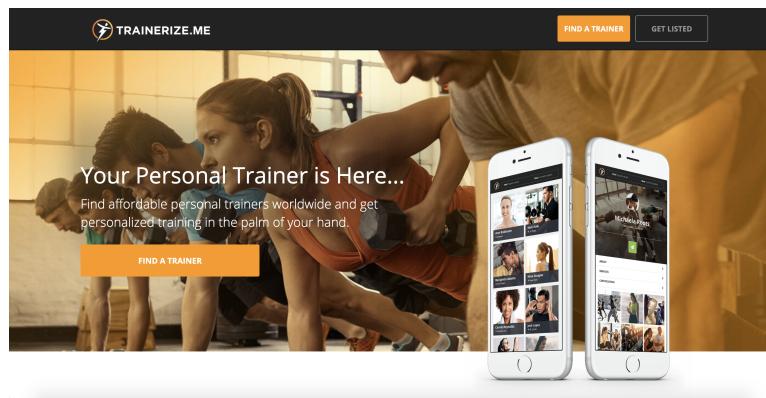
Rev.	Opis promjene/dodataka	Autori	Datum
0.1	Funkcionalni zahtjevi	Sodić	27.10.2020.
0.2	Opisi UC1-UC10 obrazaca uporabe	Milić	31.10.2020.
0.3	Opis projektnog zadatka	Merćep	31.10.2020.
0.4	Opisi UC11-UC20 obrazaca uporabe	Kniewald	1.11.2020
0.5	Opisi UC21-UC31 obrazaca uporabe	Lovrenčić	1.11.2020
0.7	Ostali zahtjevi	Kniewald	4.11.2020
0.8	Sekvencijski dijagrami	Dević	5.11.2020
0.9	Arhitektura i dizajn sustava	Dević	9.11.2020
0.10	Baza podataka	Dević, Merćep, Lovrenčić	9.11.2020
0.11	Dijagram baze podataka	Kniewald	9.11.2020
0.12	Arhitektura i dizajn sustava	Dević	10.11.2020
0.13	Sekvencijski dijagrami	Dević	10.11.2020
0.14	Dijagram razreda	Lovrenčić, Merćep	10.11.2020

2. Opis projektnog zadatka

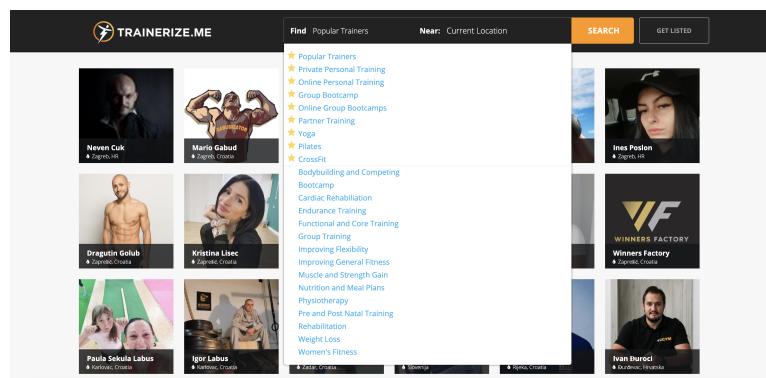
"*WebGym*" je web aplikacija namijenjena svim ljudima željnim organizacije svog vježbanja u teretanama. Korisnicima aplikacije bit će moguće pregledavanje dostupnih teretana, njihovih cijena, pregled trenera koji nude privatne ili grupne treninge, pregled ponuda planova vježbanja i planova prehrane trenera kao i mogućnost zadavanja vlastitih ciljeva te vođenje evidencije o njihovom ostvarivanju. Također ova web aplikacija omogućiće trenerima jednostavnije spajanje s klijentima, komunikaciju s teretanama te će im potencijalno proširiti tržište. Teretanama će pak ovo omogućiti odlično mjesto za prezentaciju svoje ponude jer će ovo biti platforma na kojoj će svi jednostavno moći pregledavati njihovu ponudu, te će onim teretanama s najboljim omjerom cijene i kvalitete potencijalno povećati broj korisnika.

Na internetu nismo uspjeli pronaći pandan našoj platformi niti platformu koja nudi veći opseg od naše platforme, ali neka slična rješenja postoje te će u nastavku biti detaljnije opisana.

Stranica Trainerize.me (<https://www.trainerize.me/>) nudi platformu na kojoj je vrlo jednostavno moguće pronaći osobnog trenera, vidjeti njegov opis te ga je moguće kontaktirati putem poruke. Trenere je moguće pretraživati po lokaciji, popularnosti, vrsti vježbanja koje nude te još mnogo toga. Treneri također mogu nuditi online treninge kao i treninge u teretanama. Stranica trenerima daje mjesto na kojem se oni mogu oglašavati te ih čini puno vidljivijima na tržištu. Također im zbog mogućnosti online treninga znatno proširuje tržište. Stranica uz sve već navedeno nudi i mogućnost objavljivanja te čitanja članaka vezanih uz treniranje.

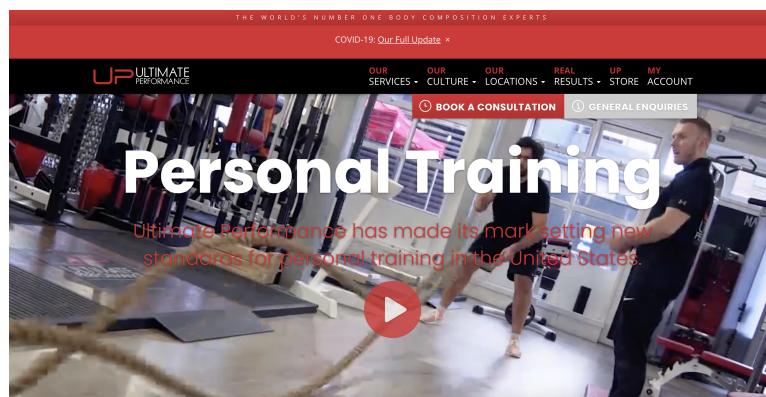


Slika 2.1: Početna stranica <https://www.trainerize.me/>



Slika 2.2: Pretraživanje trenera <https://www.trainerize.me/>

Stranica Ultimate Performance (<https://ultimateperformance.com/>) nudi go-tove planove vježbanja za određene svrhe, kao što su gubljenje kilograma, dobivanje mišićne mase i slično, također su programi podijeljeni po spolovima. Također je moguće dogovoriti konzultacije s osobnim trenerima te naručivati personalizirane planove vježbanja.



Slika 2.3: Početna stranica <https://ultimateperformance.com/>

Cilj ovog projekta je razviti programsku podršku za stvaranje web aplikacije "WebGym" koja će svojim korisnicima uvelike olakšati administrativne poslove vezane uz njihove odlaske u teretanu. Cilj platforme "WebGym" je poboljšavanje ukupnog doživljaja teretane, a namijenjena je neregistriranim korisnicima, registriranim korisnicima, trenerima u teretani i voditeljima svake od teretana (jedan voditelj može biti zadužen za više teretana i za svaku teretanu može biti zaduženo više voditelja). Za registraciju bilo kojeg od korisnika potrebno je unijeti ime, prezime, email adresu i osobne podatke bitne za trenere (visina, težina...) te je PayPal račun opcionalan.

Prilikom pokretanja sustava (odlaska na početnu stranicu) ukoliko korisnik nije trenutno prijavljen pokazuje se općenita početna stranica. Na toj početnoj stranici u gornjem desnom kutu nalaze se polja za unos korisničkog imena i lozinke te gumb za registraciju korisnika koji se do sada nisu registrirali. Na početnoj stranici se za neregistrirane korisnike nalazi popis najpopularnijih teretana, a svaka teretana u tom popisu ima prikazanu svoju profilnu sliku, ime i adresu teretane.

Za kreiranje novog računa potrebni su sljedeći podaci:

- Korisničko ime
- Ime
- Prezime
- Broj mobitela
- e-mail

Također optionalno se mogu unijeti i sljedeći podatci:

- PayPal račun
- Visina
- Težina

Pri izradi korisničkog računa moguće je odabrati jednu od tri opcije:

- Korisnik teretana (u nastavku dokumenta Klijent)
- Voditelj teretane
- Trener

Običan registrirani korisnik je osoba koja je izradila račun s namjerom pohađanja teretane radi vježbanja. Korisnik može pregledavati teretane, plaćati članarine u njima putem interneta, vidjeti u kojim sve teretanama može vježbati s već uplaćenim

članarinama (jedan lanac teretana može imati više teretana i korisnik može plaćati članarinu u više teretana), također za svaku teretanu se vodi i datum isteka trenutne članarine. Korisnik može također trenerima plaćati planove prehrane ili vježbanja te privatne i grupne treninge. Korisnik uz sve ovo može imati i svoj plan vježbanja te na samoj stranici može voditi napredak u svom planu.

Treneri su korisnici koji su pri izradi korisničkog računa kliknuli na opciju izrade trenerskog računa. Treneri na svojoj stranici mogu nuditi planove treninga i prehrane, kao i individualno ili grupno vježbanje, a sve ranije opcije mogu biti besplatne ili ih treneri mogu naplaćivati. Treneri mogu organizirati i grupno vježbanje u teretanama u kojima im je voditelj dao tu ovlast, odnosno stavio ih na popis trenera te teretane.

Treneri i obični korisnici mogu pregledati sve izvršene transakcije u kojima su sudjelovali, a voditelji mogu vidjeti sve transakcije koje su se izvršile na aplikaciji u sklopu teretana koje vode.

Administrator je korisnik koji je pri izradi svojeg korisničkog računa kliknuo na izradu administratorskog računa. Administrator ima sve voditeljske privilegije nad svim teretanama te može vidjeti sve izvršene transakcije. On može brisati i pregledavati i sve teretane i sve račune korisnika.

Voditelj teretane je korisnik koji je izradio svoj korisnički račun i pri tome kliknuo da izrađuje voditeljski račun. Svaki voditelj može stvarati nove teretane u sustavu te on može davati dozvolu drugim voditeljima da vode neku od teretana za koje je on voditelj. Uloga voditelja je postavljanje teretane u sustav te promjene bitnih informacija o toj teretani (promjena lokacije, radnog vremena i sl.). Voditelj također može dodavati registrirane trenere u svoju teretanu, odnosno može im dopustiti rad u svojoj teretani te će se trener nakon toga pokazati na popisu trenera u toj teretani.

Mogućnosti nadogradnje ovog projektnog zadatka su mnogobrojne. Moguće je uvesti chat u kojem bi se trenerima i njihovim polaznicima treninga pružila mogućnost izravnog dopisivanja o planu vježbanja, potencijalnim novim ponudama i još mnogo toga. Chat bi se također mogao koristiti između više voditelja iste teretane kako bi mogli koordinirati svoje akcije na platformi ili općenito veze uz poslovanje teretane. Također bi i komunikacija trenera i teretana uvođenjem ove opcije bila uvelike olakšana te bi bila mnogo kvalitetnija.

Druga potencijalna mogućnost nadogradnje bi bila uvedene online dućane u kojemu bi svaka teretana mogla svojim klijentima nuditi proizvode kao što su su-

plementi za teretanu, opremu za treniranje i slično. Također bi se mogao uvesti jedinstveni dućan na razini platforme u kojemu bi vlasnici platforme mogli nuditi proizvode ili bi taj dućan mogao biti izведен kao mjesto na kojem dućani koji prodaju suplemente ili opremu za treniranje mogu nuditi svoje proizvode. Ukoliko se dućan odluči izvesti na drugi način u sustavu bi se trebala dodati mogućnost stvaranja dućana te njihovih zaposlenika (to bi bilo izvedeno vrlo slično kao i za teretanu te za voditelje teretana).

3. Specifikacija programske potpore

3.1 Funkcionalni zahtjevi

Dionici:

1. Neregistrirani korisnik
2. Klijent
3. Trener
4. Voditelj teretane
5. Administrator
6. Razvojni tim

Aktori i njihovi funkcionalni zahtjevi:

1. Neregistrirani/neprijavljeni korisnik (inicijator) može:
 - (a) pregledati popis svih teretana na platformi
 - (b) sortirati spomenuti popis prema sljedećim kriterijima: ime teretane, lokacija, trener
 - (c) otvoriti početnu stranicu svake teretane na kojoj se nalaze osnovne informacije (radno vrijeme, lokacija, cijena članarine...)
 - (d) izraditi administratorski, voditeljski, trenerski ili klijentski račun s nامjером treniranja u teretani za koje je potrebno navesti ime, prezime i email adresu, dok se može, ali ne mora dodati PayPal račun te je za izradu trenerskog korisničkog računa posebno potrebno navesti posebne podatke poput visine i težine
2. Klijent (inicijator) može:
 - (a) pregledavati i sortirati popis registriranih teretana
 - (b) pregledavati i mijenjati osobne podatke
 - (c) izbrisati svoj korisnički račun
 - (d) plaćati članarine u teretanama putem interneta
 - (e) pregledavati sve izvršene transakcije u kojima su sudjelovali

(f) pregledavati popis teretana u kojima smiju vježbati, odnosno u kojima su platili članarinu

(g) kupovati planove prehrane i vježbanja od trenera

(h) ugоварати privatne ili grupne treninge

(i) voditi i pratiti napredak u vlastitom planu vježbanja

3. Trener (inicijator) može:

(a) pregledavati i sortirati popis registriranih teretana

(b) pregledavati i mijenjati osobne podatke

(c) izbrisati svoj korisnički račun

(d) objavljivati ponude planova treninga i/ili vježbanja

(e) objavljivati i ugоварati termine privatnih i grupnih treninga u teretanama gdje imaju te ovlasti

(f) pregledavati sve izvršene transakcije u kojima su sudjelovali

(g) pregledavati popis teretana u kojima smiju djelovati, odnosno raditi (vodići treninge, planovi prehrane i sl.)

(h) nuditi usluge treniranja teretanama

4. Voditelj teretane (inicijator) može:

(a) pregledavati i sortirati popis registriranih teretana

(b) pregledavati i mijenjati osobne podatke

(c) izbrisati svoj korisnički račun

(d) stvarati nove teretane u sustavu

(e) davati dozvolu drugim voditeljima da vode neke njegove teretane

(f) mijenjati važne informacije o teretanama (radno vrijeme, lokacija i sl.)

(g) dopuštati registriranim trenerima rad u teretanama koje vodi

(h) vidjeti sve izvršene transakcije na aplikaciji unutar vlastite teretane

5. Administrator (inicijator) može:

(a) pregledavati i sortirati popis registriranih teretana

(b) pregledavati i mijenjati osobne podatke

(c) vidjeti sve korisničke račune

(d) izbrisati svoj korisnički račun

(e) stvarati nove i brisati postojeće teretane u sustavu

(f) pregledati sve izvršene transakcije u aplikaciji

(g) davati dozvolu voditeljima da vode pojedine teretane

- (h) mijenjati važne informacije o teretanama (radno vrijeme, lokacija i sl.)
- (i) dopuštati registriranim trenerima rad u teretanama

6. Baza podataka (sudionik):

- (a) pohranjuje sve podatke o korisnicima
- (b) čuva informacije o ulogama pojedinih korisnika
- (c) pohranjuje podatke o svim teretanama, njihovim voditeljima, trenerima i članovima
- (d) pohranjuje izvršene transakcije

3.1.1 Obrasci uporabe

Opis obrazaca uporabe

UC1 - Pregled teretana

- **Glavni sudionik:** neregistrirani korisnik, klijent, trener, voditelj teretane i administrator
- **Cilj:** Pregled teretana
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Prikazan popis teretana
 2. Korisnik može tražiti teretane (prema nekim kriterijima - search)
 3. Korisnik može birati teretanu o kojoj će dobiti informacije

UC2 - Registracija korisnika

- **Glavni sudionik:** Neregistrirani korisnik
- **Cilj:** Izrada korisničkog računa kojim korisnik dobiva dodatne funkcionalnosti sustava
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Neprijavljeni korisnik odabire opciju registracije
 2. Neprijavljeni korisnik unosi potrebne podatke
 3. Korisnik prima obavijest o uspješnoj registraciji
- **Opis mogućih odstupanja:**
 - Odabir zauzetog korisničkog imena ili e-maila, odabir nepostojećeg e-maila ili nedozvoljen format unosa nekog od podataka
 1. Sustav neprijavljenom korisniku šalje objavu o neuspješnoj registraciji te ga vraća na početnu stranicu za registraciju
 2. Korisnik mijenja neispravne podatke i završava unos ili odustaje od registriranja

UC3 - Prijava u sustav

- **Glavni sudionik:** Klijent
- **Cilj:** Dobiti pristup korisničkom sučelju
- **Sudionici:** Baza podataka

- **Preduvjet:** Registracija
- **Opis osnovnog tijeka:**
 1. Unos korisničkog imena i lozinke
 2. Potvrda o ispravnosti unesenih podataka
 3. Pristup korisničkim funkcionalnostima
- **Opis mogućih odstupanja:**
 - Neispravno korisničko ime i/ili lozinka
 1. Sustav obaveštava korisnika o neuspješnom upisu i vraća ga na stranicu za prijavu

UC4 - Pregled osobnih podataka

- **Glavni sudionik:** Klijent
- **Cilj:** Pregledati osobne podatke korisnika
- **Sudionici:** Baza podataka
- **Preduvjet:** Klijent je prijavljen
- **Opis osnovnog tijeka:**
 1. Klijent odabire opciju "Osobni podaci"
 2. Aplikacija prikazuje osobne podatke korisnika

UC5 - Promjena osobnih podataka

- **Glavni sudionik:** Klijent, trener, voditelj, administrator
- **Cilj:** Promijeniti osobne podatke
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
 1. Korisnik odabire opciju promjene osobnih podataka
 2. Korisnik mijenja svoje osobne podatke
 - Ako je korisnik prijavljen kao klijent, on može postaviti svoje ciljeve i rezultate
 - Ako je korisnik prijavljen kao trener, on može uređivati svoju stranicu
 3. Korisnik bira opciju "Spremi promjenu"
 4. Ažuriranje baze podataka
- **Opis mogućih odstupanja:**
 - Korisnik promijeni podatke, ali ne odabere opciju "Spremi promjenu"

1. Sustav obavještava korisnika da nije spremio podatke prije izlaska iz prozora

UC6 - Brisanje korisničkog računa

- **Glavni sudionik:** Klijent, trener, voditelj, administrator
- **Cilj:** Izbrisati svoj korisnički račun
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
 1. Korisnik pregledava osobne podatke
 2. Korisnik bira opciju "Obriši račun"
 3. Korisnik briše račun
 4. Korisnikov račun se briše iz baze podataka
 5. Otvara se početna stranica

UC7 - Pregled specifične teretane

- **Glavni sudionik:** Neregistrirani korisnik, klijent, trener, voditelj, administrator
- **Cilj:** Vidjeti osnovne podatke o teretani i trenere u toj teretani
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Korisnik odabire željenu teretanu
 2. Prikazuju se voditelji teretane, treneri koji su dio te teretane, lokacija i radno vrijeme te ponuda članarine

UC8 - Pregled transakcija

- **Glavni sudionik:** Klijent, trener, voditelj
- **Cilj:** Pregled transakcija u kojima je korisnik do sada sudjelovao
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
 1. Korisnik odabire opciju pregleda osobnih podataka
 2. Korisnik odabire opciju pregleda transakcija
 3. Korisnik dobiva prikaz svih transakcija u kojima je sudjelovao

UC9 - Pregled odredene transakcije

- **Glavni sudionik:** Klijent, trener, voditelj
- **Cilj:** Pregled sudionika u transakciji, opis usluge, iznos plaćanja u kunama i datum izvršenja transakcije
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
 1. Korisnik odabire opciju pregleda transakcija
 2. Korisnik odabire opciju pregleda određene transakcije
 3. Pregled detalja odabrane transakcije

UC10 - Učlanjivanje korisnika u određenu teretanu

- **Glavni sudionik:** Klijent
- **Cilj:** Učlanjivanje korisnika u određenu teretanu (ili lanac teretana)
- **Sudionici:** Baza podataka
- **Preduvjet:** Klijent je prijavljen
- **Opis osnovnog tijeka:**
 1. Klijent odabire određenu teretanu
 2. Klijentu se prikazuje ponuda vrsta članarina
 3. Klijent odabire opciju članarine koju želi platiti
- **Opis mogućih odstupanja:**
 - Klijent pokušava kupiti članarinu u teretani u kojoj već ima aktivnu članarinu istog tipa
 1. Sustav obaveštava klijenta da već postoji aktivna članarina tog tipa
 2. Vraća ga na stranicu s popisom članarina te teretane

UC11 - Plaćanje narudžbe

- **Glavni sudionik:** Klijent
- **Cilj:** Platiti željenu narudžbu
- **Sudionici:** Baza podataka
- **Preduvjet:** Klijent je prijavljen i napravio je narudžbu
- **Opis osnovnog tijeka:**
 1. Prikazuju se podaci o narudžbi
 2. Klijent odabire opciju dovrši narudžbu
- **Opis mogućih odstupanja:**
 - Nedovoljno stanje računa klijenta
 1. Sustav obaveštava klijenta da nema dovoljno sredstava na računu

UC12 - Pregled informacija o određenom treneru

- **Glavni sudionik:** Registrirani i neregistrirani korisnici
- **Cilj:** Dobiti informacije odabranog trenera
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Odabrat određenog trenera
 2. Prikaz osnovnih informacija (ime, prezime, popis teretana u kojima radi)
 3. Prikaz ponuda treninga i plana prehrane

UC13 - Odabir trenerovog programa treniranja ili plana prehrane

- **Glavni sudionik:** Klijent
- **Cilj:** Odabrat trenerov programa treniranja ili plana prehrane
- **Sudionici:** Baza podataka, Trener
- **Preduvjet:** Klijent je prijavljen
- **Opis osnovnog tijeka:**
 1. Klijent odabire određenu ponudu treninga ili plana prehrane
 2. Ima opciju izrade narudžbe
 3. Klijent potvrđuje narudžbu
- **Opis mogućih odstupanja:**
 - Klijent pokušava rezervirati program treninga koji je popunjen
 - 1. Sustav javlja da je termin popunjen i nudi mu slobodne termine

UC14 - Pregled, uređivanje i postavljanje vlastitih ciljeva

- **Glavni sudionik:** Klijent
- **Cilj:** Pregled, uređivanje i postavljanje vlastitih ciljeva
- **Sudionici:** Baza podataka
- **Preduvjet:** Klijent je prijavljen i odabrao je promjenu svojih podataka
- **Opis osnovnog tijeka:**
 1. Odabire opciju "Moji ciljevi"
 2. Pregled postojećih ciljeva
 3. Postavljanje novih ciljeva i izmjena postojećih (može ga se izmijeniti ili postaviti kao odrađenog)

UC15 - Uređivanje trenerove ponude za treninge

- **Glavni sudionik:** Trener

- **Cilj:** Uređivanje ponude za treninge
- **Sudionici:** Baza podataka
- **Preduvjet:** Trener je prijavljen i odabrao je promjenu svojih podataka
- **Opis osnovnog tijeka:**
 1. Trener uređuje svoju ponudu
 2. Izmjena se pohrani
 3. Baza podataka se ažurira

UC16 - Uređivanje trenerove ponude za plan prehrane

- **Glavni sudionik:** Trener
- **Cilj:** Uređivanje ponude za plan prehrane
- **Sudionici:** Baza podataka
- **Preduvjet:** Trener je prijavljen i odabrao je promjenu svojih podataka
- **Opis osnovnog tijeka:**
 1. Trener uređuje svoju ponudu
 2. Izmjena se pohrani
 3. Baza podataka se ažurira

UC17 - Dozvola za rad trenera u teretani

- **Glavni sudionik:** Trener
- **Cilj:** Dobiti dozvolu za rad u određenoj teretani
- **Sudionici:** Baza podataka, Voditelji odabrane teretane
- **Preduvjet:** Trener je prijavljen, odabrana željena teretana
- **Opis osnovnog tijeka:**
 1. Trener odabire opciju za pisanje zamolbe
 2. Trener piše zamolbu
 3. Trener šalje zamolbu za dozvolu za rad u toj teretani
- **Opis mogućih odstupanja:**
 - Trener je već zaposlen u toj teretani
 1. Sustav mu javlja da je već zaposlen u toj teretani

UC18 - Pregled postojećih klijenata

- **Glavni sudionik:** Trener
- **Cilj:** Pregled postojećih klijenata i popis zajedničkih termina treninga
- **Sudionici:** Baza podataka
- **Preduvjet:** Trener je prijavljen, odabran pregled podataka

- **Opis osnovnog tijeka:**

1. Trener odabire opciju za popis klijenata
2. Treneru se prikazuje cijeli popis klijenata i za svakog popis zajedničkih termina treninga

UC19 - Pregled podataka određenog klijenta

- **Glavni sudionik:** Trener

- **Cilj:** Pogledati podatke određenog klijenta

- **Sudionici:** Baza podataka

- **Preduvjet:** Trener je prijavljen, odabran pregled podataka

- **Opis osnovnog tijeka:**

1. Trener odabire određenog klijenta
2. Treneru se prikazuju osobni podatci klijenta

UC20 - Pregled trenerove ponude

- **Glavni sudionik:** Trener

- **Cilj:** Pregledati trenerovu ponudu programa i plana prehrane

- **Sudionici:** Baza podataka

- **Preduvjet:** Prijavljen je trener

- **Opis osnovnog tijeka:**

1. Trener odabire pregled svoje ponude
2. Prikazuje mu se njegova ponuda treninga i plana prehrane

UC21 - Stvaranje nove teretane

- **Glavni sudionik:** Voditelj, admin

- **Cilj:** Stvoriti novu teretanu u sustavu

- **Sudionici:** Baza podataka

- **Preduvjet:** Prijavljen je voditelj ili admin

- **Opis osnovnog tijeka:**

1. Voditelj ili admin je odabrao opciju stvaranje nove teretane
2. Voditelj ili admin postavlja podatke teretane i stvara ju

- **Opis mogućih odstupanja:**

- Teretana takvog imena već postoji u sustavu
 1. Sustav šalje poruku da je ime teretane već zauzeto
 2. Voditelj ili admin odabire novo ime teretane

UC22 - Pregled voditeljevih teretana

- **Glavni sudsionik:** Voditelj
- **Cilj:** Pregledati popis teretana kojima je on voditelj
- **Sudsionici:** Baza podataka
- **Preduvjet:**
 1. Prijavljen je voditelj
 2. Voditelj je odabrao pregled svojih teretana
- **Opis osnovnog tijeka:**
 1. Voditelj je odabrao opciju "Moje teretane"
 2. Prikazuje se popis teretana tog voditelja

UC23 - Micanje teretane s voditeljevog popisa

- **Glavni sudsionik:** Voditelj
- **Cilj:** Maknuti teretanu s popisa svojih teretana
- **Sudsionici:** Baza podataka
- **Preduvjet:**
 1. Prijavljen je voditelj
 2. Voditelj je odabrao pregled svojih teretana
- **Opis osnovnog tijeka:**
 1. Voditelj odabire teretanu koju želi maknuti s popisa
 2. Voditelj potvrđuje odabir
 3. Baza podataka se ažurira
- **Opis mogućih odstupanja:**
 - Ako voditelj pokuša maknuti teretanu sa svog popisa i on je jedini voditelj
 1. Sustav obavještava voditelja da je on jedini voditelj

UC24 - Brisanje teretane iz sustava

- **Glavni sudsionik:** Voditelj, admin
- **Cilj:** Obrisati teretanu iz sustava
- **Sudsionici:** Baza podataka
- **Preduvjet:**
 1. Prijavljen je voditelj ili admin
 2. Ako je prijavljen voditelj, on odabire pregled svojih teretana
- **Opis osnovnog tijeka:**
 1. Voditelj ili admin odabire teretanu koju želi maknuti s popisa
 2. Voditelj ili admin potvrđuje odabir

3. Baza podataka se ažurira

UC25 - Izmjena podataka teretane od strane voditelja

- **Glavni sudionik:** Voditelj
- **Cilj:** Izmjena podataka određene teretane voditelja
- **Sudionici:** Baza podataka
- **Preduvjet:**
 1. Prijavljen je voditelj
 2. Voditelj odabire pregled svojih teretana
- **Opis osnovnog tijeka:**
 1. Voditelj odabire teretatu kojoj želi izmjeniti podatke
 2. Voditelj potvrđuje izmjenu podataka
 3. Baza podataka se ažurira

UC26 - Dodavanje trenera u određenu teretanu

- **Glavni sudionik:** Voditelj
- **Cilj:** Davanje treneru dozvolu za rad u toj teretani
- **Sudionici:** Baza podataka,trener
- **Preduvjet:**
 1. Prijavljen je voditelj
 2. Trener se prijavio za rad u toj teretani
- **Opis osnovnog tijeka:**
 1. Voditelj otvara zamolbu trenera za dozvolu za rad
 2. Voditelj potvrđuje trenera za rad u teretani
- **Opis mogućih odstupanja:**
 - Trener je već zaposlen u teretani
 - 1. Sustav obaveštava voditelja da je trener već zaposlen u teretani

UC27 - Dodavanje voditelja u svoju teretanu

- **Glavni sudionik:** Voditelj
- **Cilj:** Davanje voditelju dozvolu za rad u toj teretani
- **Sudionici:** Baza podataka,voditelj
- **Preduvjet:** Prijavljen je voditelj
- **Opis osnovnog tijeka:**
 1. Voditelju se prikazuje popis svih voditelja
 2. Voditelj odabire voditelja kojeg želi

3. Voditelj dodaje voditelja u svoju teretanu
 4. Baza podataka se ažurira
- **Opis mogućih odstupanja:**
 - Voditelj je već zaposlen u teretani
 - 1. Sustav obavještava voditelja da je voditelj već zaposlen u teretani

UC28 - Pregled svih korisničkih računa

- **Glavni sudionik:** Admin
- **Cilj:** Pregledati sve korisniče račune
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen je admin
- **Opis osnovnog tijeka:**
 1. Admin odabire pregled svih korisničkih računa
 2. Adminu se prikazuju svi korisnički računi

UC29 - Pregled svih transakcija

- **Glavni sudionik:** Admin
- **Cilj:** Pregledati sve transakcije
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen je admin
- **Opis osnovnog tijeka:**
 1. Admin odabire pregled svih transakcija
 2. Adminu se prikazuju sve transakcije

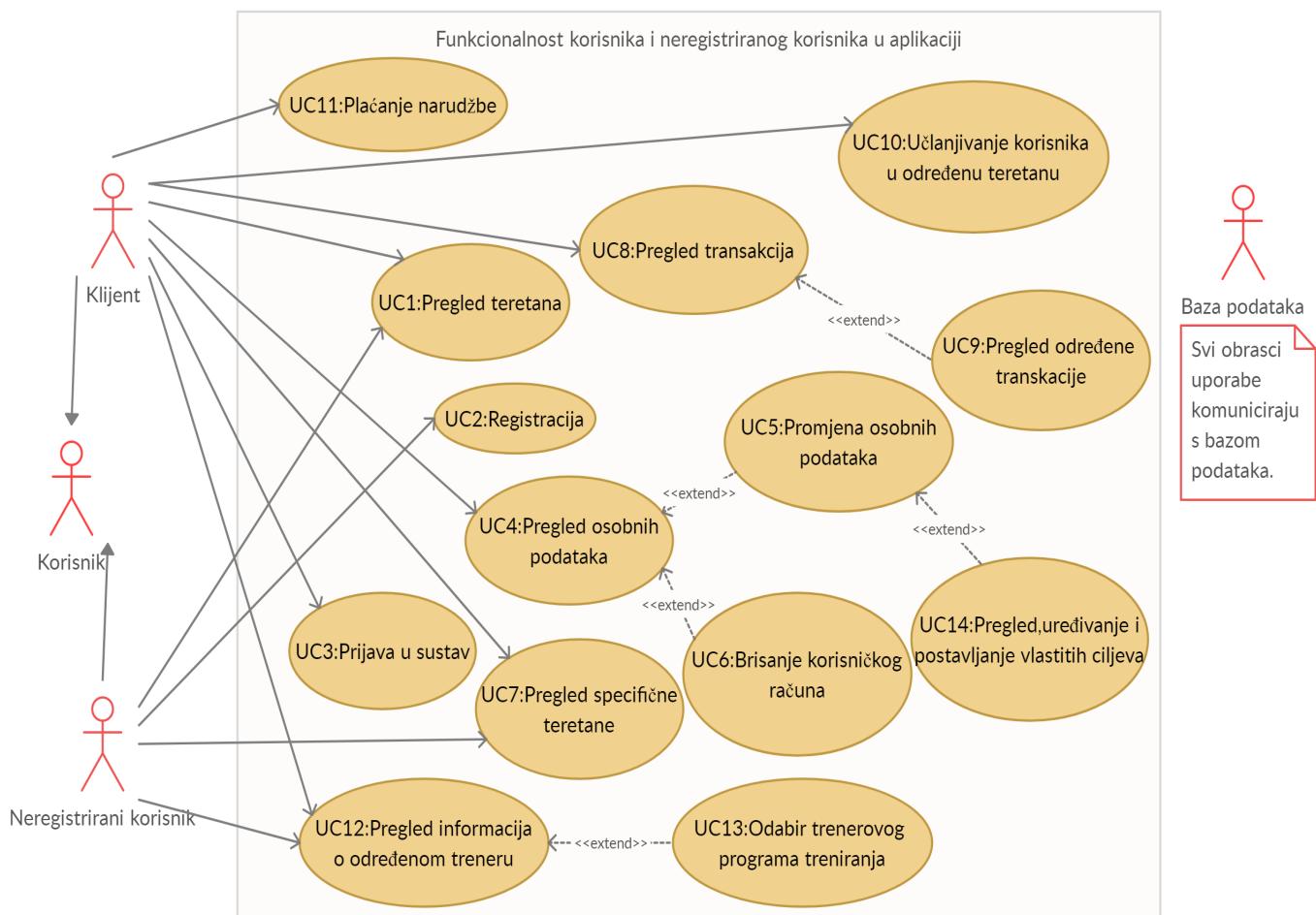
UC30 - Dodavanje voditelja u bilo koju teretanu

- **Glavni sudionik:** Admin
- **Cilj:** Davanje voditelju dozvolu za rad u nekoj teretani
- **Sudionici:** Baza podataka, voditelj
- **Preduvjet:** Prijavljen je admin
- **Opis osnovnog tijeka:**
 1. Admin dodaje voditelja u tu teretanu
 2. Baza podataka se ažurira
- **Opis mogućih odstupanja:**
 - Voditelj je već zaposlen u teretani
 - 1. Sustav obavještava admina da je voditelj već zaposlen u teretani

UC31 - Izmjena podataka bilo koje teretane od strane admina

- **Glavni sudionik:** Admin
- **Cilj:** Izmjena podataka neke teretane
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen je admin
- **Opis osnovnog tijeka:**
 1. Admin odabire teretatnu kojoj želi izmjeniti podatke
 2. Admin potvrđuje izmjenu podataka
 3. Baza podataka se ažurira

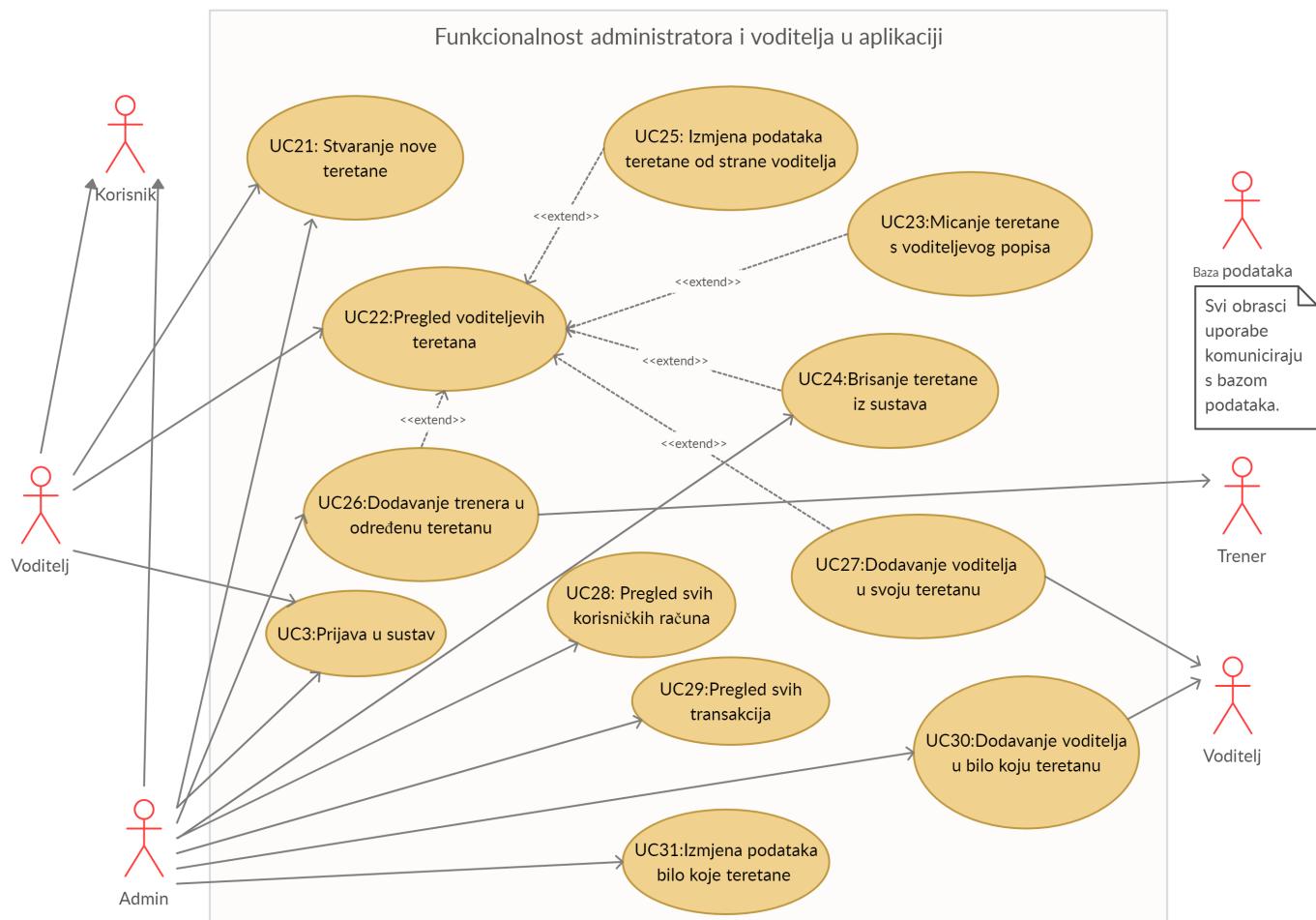
Dijagrami obrazaca uporabe



Slika 3.1: Dijagram obrasca uporabe, funkcionalnost korisnika i neregistriranog korisnika u aplikaciji



Slika 3.2: Dijagram obrasca uporabe, funkcionalnost korisnika i trenera u aplikaciji

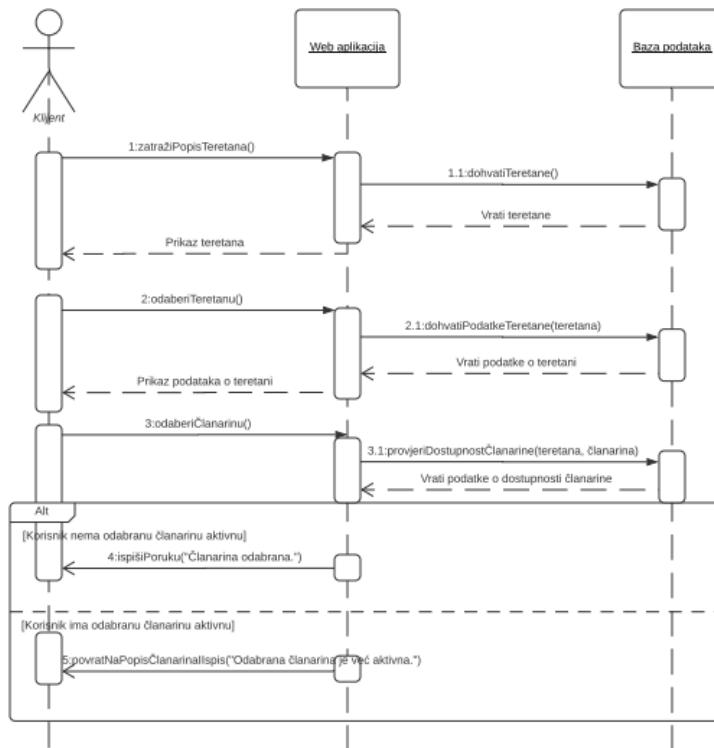


Slika 3.3: Dijagram obrasca uporabe, Funkcionalnost korisnika i neregistriranog korisnika u aplikaciji

3.1.2 Sekvencijski dijagrami

Obrazac uporabe UC10 - Učlanjivanje korisnika u određenu teretanu (ili lanac teretana)

Klijent šalje zahtjev za popis svih teretana te web aplikacija dohvaća popis teretana iz baze podataka te prikazuje klijentu. Klijent potom odabire teretanu te mu se izlistava popis članarina unutar odabrane teretane, tada klijent odabire članarinu koju želi te ako trenutno nema odabranu članarinu aktivnu dobiva poruku kako je članarina odabrana, dok ga u suprotnom Web aplikacija preusmjerava natrag na popis članarina za odabratи iz odabrane teretane.

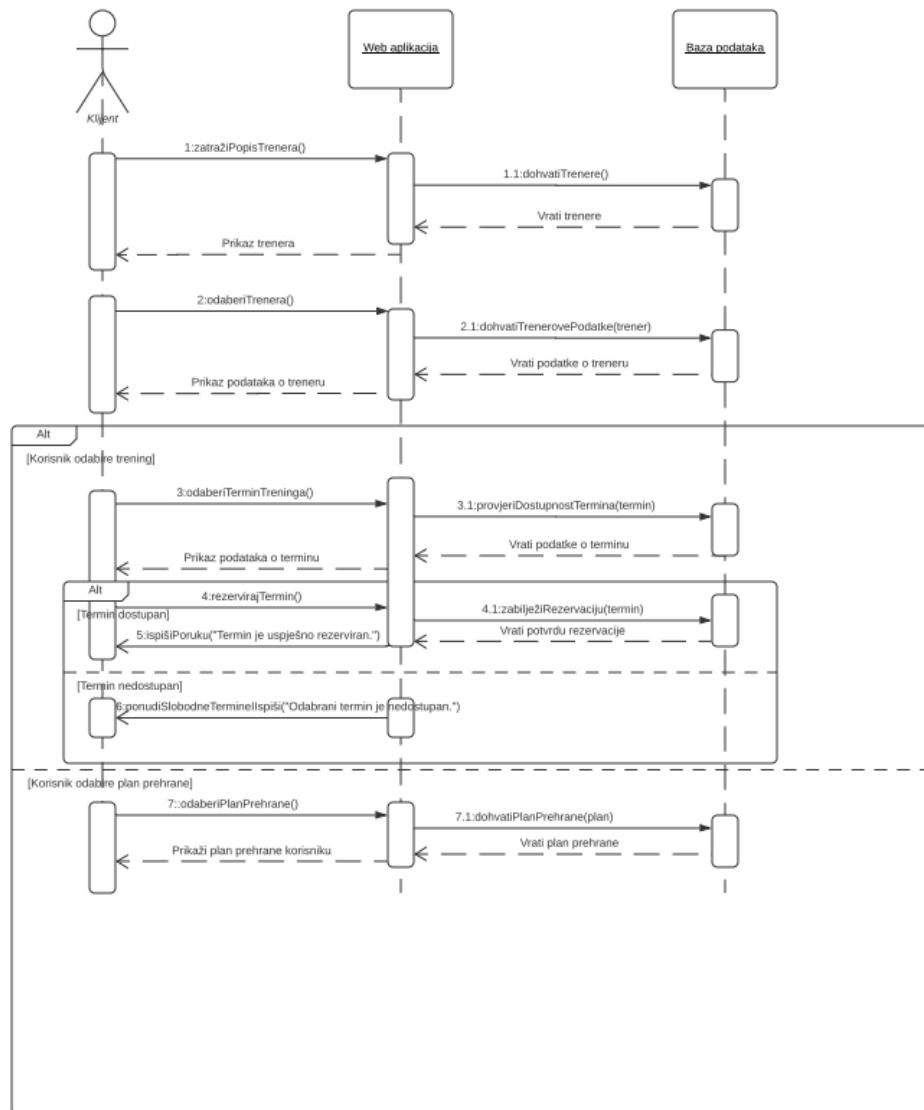


Slika 3.1: Sekvencijski dijagram za UC10

Obrazac uporabe UC13 - Odabir trenerovog programa treniranja ili plana prehrane

Klijent šalje zahtjev za popis svih trenera te web aplikacija dohvaća popis trenera iz baze podataka te ih sve prikazuje klijentu. Klijent potom odabire trenera. Tada klijent ima opciju odabratи trening ili plan prehrane kod odabranog trenera. U slučaju odabira plana prehrane se pokazuje korisniku. U slučaju

odabira treninga, korisnik uz trening odabire i termin treninga. Ako je termin slobodan korisnik dobiva tu informaciju dok u suprotnom dobiva popis slobodnih termina treninga kod odabranog trenera iz kojih potom bira novi termin.

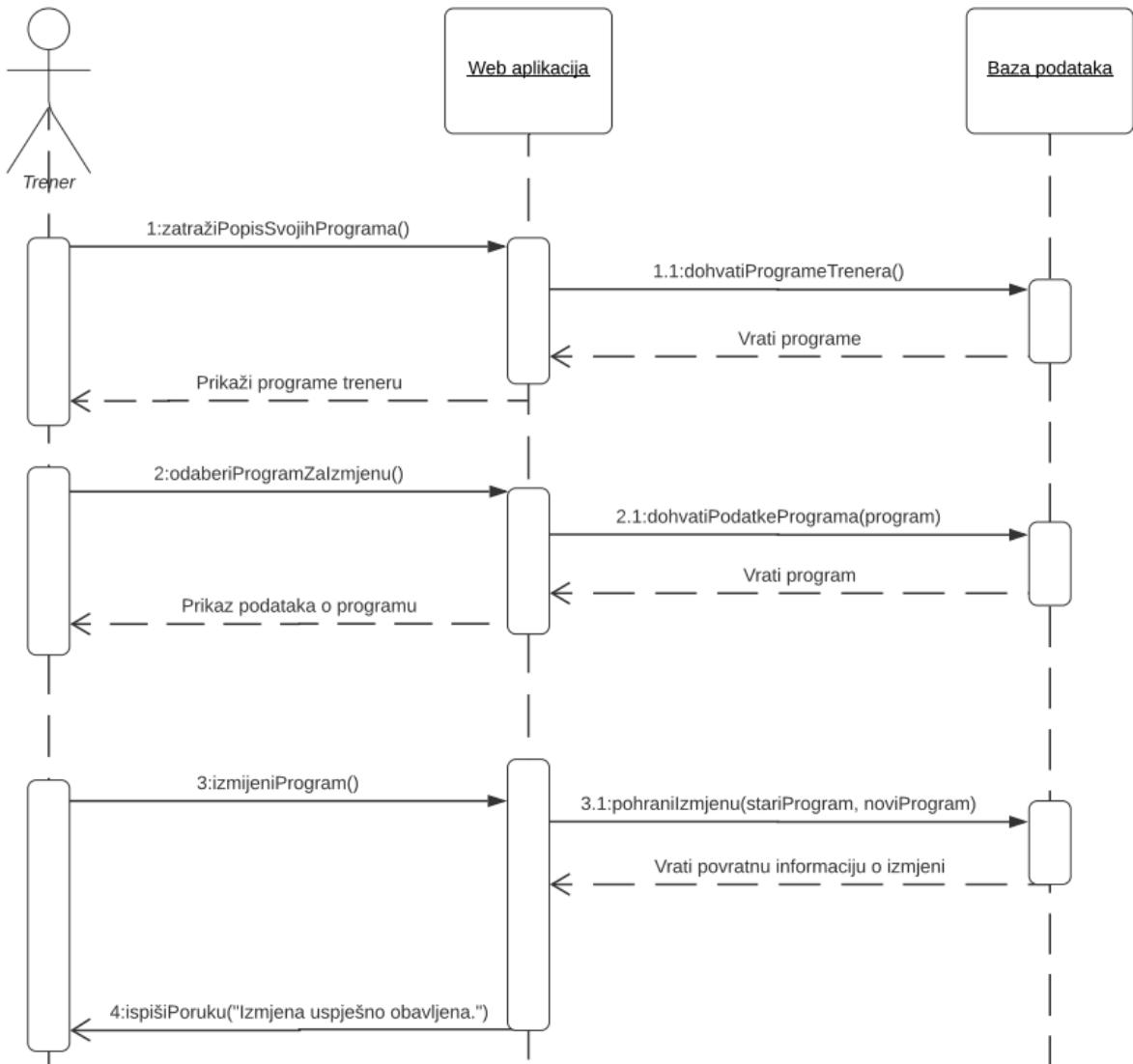


Slika 3.2: Sekvencijski dijagram za UC13

Obrazac uporabe UC15 - Uređivanje trenerove ponude za treninge

Trener šalje zahtjev za popis svih njegovih programa te web aplikacija dohvata popis iz baze podataka te ih prikazuje treneru. Trener potom odabire jedan od ponuđenih programa te web aplikacija iz baze podataka povlači informacije o programu te ih prikazuje treneru. Trener potom može izmjeniti program te se nak-

nadno ta promjena zabilježava u bazu podataka za buduće izmjene ili potražnje korisnika za tim programom.

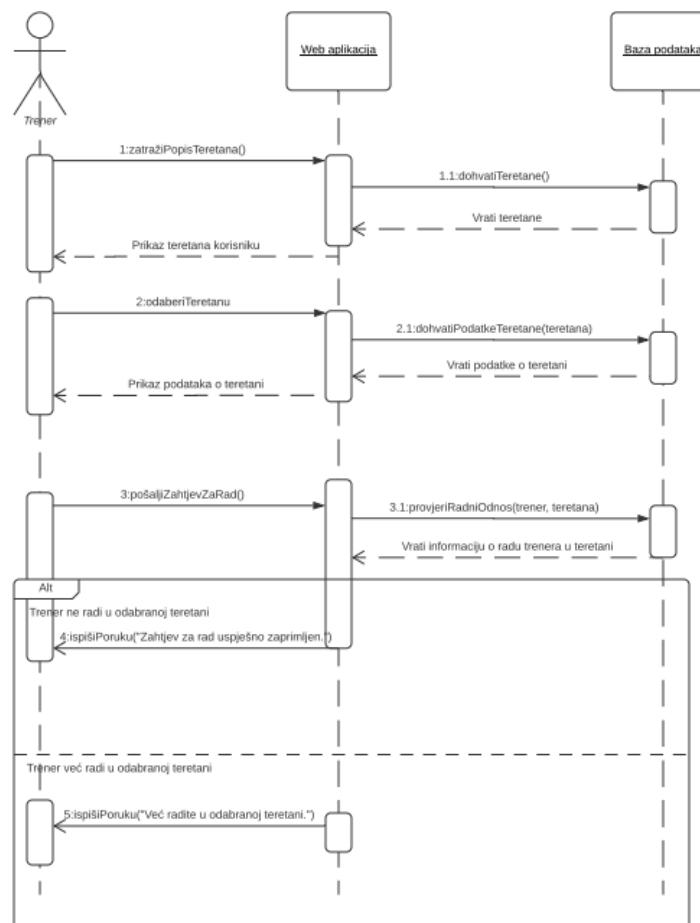


Slika 3.3: Sekvencijski dijagram za UC15

Obrazac uporabe UC17 - Dozvola za rad trenera u teretani

Trener šalje zahtjev za popis svih teretana te web aplikacija dohvaca popis iz baze podataka te ih prikazuje treneru. Trener potom odabire teretanu za koju je zainteresiran, a podaci o teretani se ponovno dohvacaju iz baze podataka. Tada trener šalje zahtjev za rad u teretanu, a web aplikacija u bazi provjerava radi li već trener u odabranoj teretani. U slučaju da ne radi u odabranoj teretani dobiva poruku

kako je zahtjev uspješno zaprimljen te zahtjev čeka odobrenje voditelja teretane, a u slučaju da trener ondje već radi dobiva poruku koja o tome obavještava.



Slika 3.4: Sekvencijski dijagram za UC17

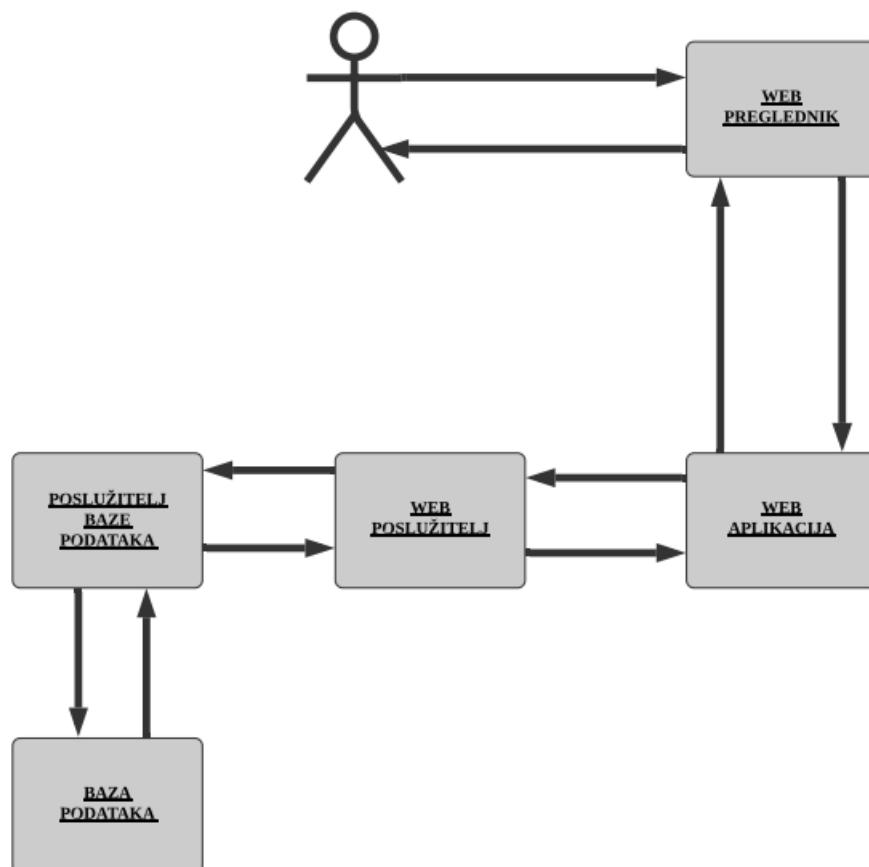
3.2 Ostali zahtjevi

- Sustav treba podržati rad više korisnika u stvarnom vremenu bez gubitaka funkcionalnosti
- Prikaz i tekstualni unos mora podržavati hrvatske dijakritičke znakove
- Sustav treba biti jednostavan za korištenje, korisnici se moraju znati koristiti web sučeljem i sustavom bez opširnih uputa
- Sustav bi trebao jamčiti točnost i pouzdanost informacija u aplikaciji
- Ažuriranje i nadogradnja sustava ne smije narušavati postojeće funkcionalnosti sustava
- Baza podataka mora biti brza i zaštićena od bilo kakvih vanjskih utjecaja
- Odgovor na korisnikov upit koji je povezan sa bazom podataka trebao bi trajati najduže par sekundi
- Nepravilne akcije korisnika, trenera ili voditelja u korisničkom sučelju ne smiju narušiti postojeće funkcionalnosti sustava
- Sustav kao valutu dopušta hrvatsku kunu
- Sustavu se treba moći pristupiti iz javne mreže sigurnim protokolom (HT-TPS)

4. Arhitektura i dizajn sustava

Arhitektura korištena pri izradi WebGym web aplikacije sastoji se od:

- Web poslužitelj
- Web aplikacija
- Baza podataka
- Poslužitelj baze podataka



Slika 4.1: Arhitektura sustava

Na samom početku programiranja sustava bilo je neophodno dogovoriti se oko arhitekture i tehnologija koje će se koristiti za izradu programskog rješenja. Preciznije, dogovaralo se o pojedinostima kao što su programski jezici, razvojna okruženja i koncepti za razvoj aplikacije. Arhitekturu smo odabrali kako bi zadovoljavala mnoge značajke kao što su fleksibilnost, mogućnost nadogradnje i jeftinije održavanje. Web aplikacija je razuman odabir zato što ovisi samo o veličini ekrana uređaja na kojoj se koristi, a taj problem je jednostavniji za riješiti nego problem specifične arhitekture pojedinih operacijskih sustava. Također, odabrana arhitektura od korisnika naše web aplikacije zahtijeva samo pristup internetu te web preglednik.

Pregled web-stranice, svih podataka i zatraženih medija odvija se pomoću web preglednika. Web preglednik ima nekoliko uloga, jedna od njih je prikaz stranice na način koji je čitljiv čovjeku. Stoga, možemo reći da web preglednik ima ulogu, odnosno funkciju prevoditelja. Također, upravo je web preglednik alat pomoću kojeg korisnik šalje zahtjeve web poslužitelju.

Iduća funkcionalnost je komunikacija korisnika s aplikacijom. Tome služi web poslužitelj koji je svojevrsni motor web aplikacije. Sama komunikacija odvija se pomoću HTTP protokola koji služi za prijenos informacija i komunikaciju na internetu.

Web aplikacija se pokreće pomoću web poslužitelja kojem šalje zahtjeve, a web poslužitelj služi za obradu tih zahtjeva. Za prikaz je zaslužan već prije spomenuti web poslužitelj koji vraća korisniku HTML dokument kao odgovor, a čitljivi prikaz korisniku osiguran je pomoću web preglednika. Također, web poslužitelj zahtjeve vezane uz podatke u bazi podataka prosljeđuje prema poslužitelju baze podataka.

Poslužitelj baze podataka ispunjava ulogu komunikacije s bazom podataka. Dakle, on zaprima i obrađuje zahtjeve za podacima poslane iz web aplikacije koje mu prosljeđuje web poslužitelj. Ovisno o vrsti zathjeva, u odgovorima se mogu nalaziti zatraženi podatci ili informacije o uspješnosti nekih operacija, na primjer autentifikacije ili umetanja novih podataka u bazu podataka.

Aplikacija se sastoji od front-end-a i back-end-a. S jedne strane, tehnologija koju smo odabrali za izradu back-end-a je Java Spring Boot. Konkretno, back-end obuhvaća komunikaciju s bazom podataka (poslužitelj baze podataka). S druge strane, za front-end smo odabrali React.js. Front-end obuhvaća prikaz korisničkog sučelja aplikacije (web aplikacija i web poslužitelj) i slanje zahtjeva za podacima prema back-endu. Arhitektura sustava za koju smo se odlučili je MVC odnosno (Model-View-Controller) koncept.

Glavna korist MVC kocepta je ta što dijelovi aplikacije nisu usko povezani i omogućavaju se jednostavnije testiranje, a pogreška u jednom od dijelova neće utjecati na rad ostalih.

MVC koncept sastoji se od:

- **Model** - klase koje opisuju same podatke koji se koriste u aplikaciji. Uz to, klase sadrže i logiku vezanu za te podatke. Podaci u modelu ne ovise o korisničkom sučelju. Model prima ulazne podatke od Controllera.
- **View** - omogućuje komunikaciju s korisnikom te definira kako će korisničko sučelje biti prikazano korisniku.
- **Controller** - klase koje kontroliraju komunikaciju korisnika sa aplikacijom. To je sloj aplikacije koji obrađuje korisničke zahtjeve. Controller prihvata sve ulaze i pretvara ih u naredbe za Model i View.

4.1 Baza podataka

Za potrebe WebGym sustava koristit ćemo relacijsku bazu podataka koja nam pojednostavljuje modeliranje odnosa i scenarija koji se u realnosti događaju. Svaka je tablica definirana svojim imenom te skupom atributa koji su nam potrebni za punu funkcionalnost pojma ili odnosa opisanim tablicom. Brza i jednostavna pohrana te pristup samim podacima je zadaća ovog sustava pohranjivanja podataka u bazu. Baza podataka sastoji se od sljedećih entiteta:

- Korisnik
- Teretana
- TeretanaLokacija
- Ciljevi
- PlanTreningaIPrehrane
- Članarina
- ČlanarinaKlijent
- Zamolba
- PlanKlijent

- TeretanaKorisnik

4.1.1 Opis tablica

Korisnik Ovaj entitet sadržava sve bitne informacije o korisniku WebGym web aplikacije. Sadrži atribute kao što su: korisničko ime, ime, prezime, email, hashirana lozinku, broj mobitela, broj PayPala, visinu, težinu te ulogu u sustavu. Primarni ključ ovog entiteta je korisničkoIme te je stoga taj atribut boldan u tablici. Uloga definira ovlasti koje korisnik u sustavu i u odnosima ima. Ovaj entitet je u vezi *One-to-Many* s entitetom Ciljevi preko korisničkog imena klijenta koji je i sam korisnik, u vezi s *Many-to-Many* entitetom PlanTreningaIPrehrane preko korisničkog imena trenera koji je i sam korisnik, u vezi *One-to-Many* s entitetom ČlanarinaKlijent preko korisničkog imena klijenta, u vezi *One-to-Many* s entitetom Zamolba korisničkog imena trenera, u vezi *Many-to-Many* s entitetom PlanKlijent preko korisničkog imena klijenta te u vezi *Many-to-Many* s entitetom TeretanaKorisnik preko korisničkog imena trenera ili voditelja koji su i sami korisnici.

Korisnik		
korisničkoIme	VARCHAR	jedinstveno ime korisnika
ime	VARCHAR	ime korisnika
prezime	VARCHAR	prezime korisnika
email	VARCHAR	email korisnika s kojim je napravio račun
lozinka	VARCHAR	hash korisnikove lozinke za prijavu
brojMobitela	VARCHAR	korisnikov broj mobitela
PayPal	VARCHAR	korisnikov PayPal za plaćanje usluga
visina	INT	korisnikova težina
težina	INT	korisnikova visina
uloga	VARCHAR	korisnikova uloga u sustavu

Teretana Ovaj entitet sadrži sve bitne informacije o lancu teretani. Sadrži atribute: id, ime, opis te email, a ovaj je entitet u vezi *One-to-Many* s entitetom TeretanaLokacija preko jedinstvenog brojčanog identifikatora teretane te u vezi *One-to-Many* s entitetom članarina preko jedinstvenog brojčanog identifikatora teretane. Primarni ključ ovog entiteta je id te je stoga taj atribut boldan u tablici.

Teretana		
id	INT	jedinstveni brojčani identifikator teretane
ime	VARCHAR	ime teretane
opis	VARCHAR	opis teretane
email	VARCHAR	e mail teretane

TeretanaLokacija Ovaj entitet sadrži sve bitne informacije o pojedinoj teretani. Sadrži atribute: id, id lanca teretani, država u kojoj se nalazi, grad u kojem se nalazi, ulica u kojoj se nalazi, početak radnog vremena teretane, kraj radnog vremena teretane te broj telefona teretane, a ovaj je entitet u vezi *Many-to-One* s entitetom Teretana preko jedinstvenog brojčanog identifikatora teretane, u vezi *One-to-Many* s entitetom Zamolba preko jedinstvenog brojčanog identifikatora pojedine teretane te je u vezi *One-to-Many* s entitetom TeretanaKorisnik preko jedinstvenog brojčanog identifikatora pojedine teretane. Primarni ključ ovog entiteta je id te je stoga taj atribut boldan u tablici, dok je idTeretana strani ključ te je stoga taj atribut pisan formatom italic u tablici.

TeretanaLokacija		
id	INT	jedinstveni brojčani identifikator pojedine teretane
<i>idTeretana</i>	INT	jedinstveni brojčani identifikator lanca teretani (Teretana.id)
država	VARCHAR	država u kojoj se teretana nalazi
grad	VARCHAR	grad u kojem se teretana nalazi
ulica	VARCHAR	ulica u kojoj se teretana nalazi
radnoVrijemePočetak	TIME	vrijeme početka rada teretane
radnoVrijemeKraj	TIME	vrijeme kraja rada teretane
telefon	VARCHAR	broj na koji se teretana može nazvati

Ciljevi Ovaj entitet sadrži sve važne informacije o ciljevima samog klijenta. Sadrži atribute: id, korisničko ime klijenta te opis cilja, a ovaj je entitet u vezi *Many-to-One* s entitetom Korisnik preko korisničkog imena. Primarni ključ ovog entiteta je id te je stoga taj atribut boldan u tablici, dok je korisničkoImeKlijent strani ključ te je stoga taj atribut pisan formatom italic u tablici.

Ciljevi		
id	INT	jedinstveni brojčani identifikator cilja
<i>korisničkoImeKlijent</i>	VARCHAR	jedinstveni identifikator klijenta koji cilj obavlja (Korisnik.korisničkoIme)
opis	VARCHAR	opis cilja
obavljen	BOOLEAN	je li cilj obavljen

PlanTreningaIPrehrane Ovaj entitet sadrži sve važne informacije o planu treninga i prehrane. Korisnik može kupiti trening te onda ima individualni pristup trenera, ili može samo kupiti plan te dobiti gotov file sa treninzima i preporučenim načinom prehrane. Sadrži atribute: id, korisničko ime trenera, opis plana, datum početka plana, datum isteka plana te atribut koji nam govori je li trening u pitanju, a ovaj je entitet u vezi *Many-to-Many* s entitetom Korisnik preko korisničkog imena. Primarni ključ ovog entiteta je id te je stoga taj atribut boldan u tablici, dok je korisničkoImeTrener strani ključ te je stoga taj atribut pisan formatom italic u tablici.

PlanTreningaIPrehrane		
id	INT	jedinstveni brojčani identifikator plana treninga i prehrane
<i>korisničkoImeTrener</i>	VARCHAR	jedinstveni identifikator trenera čiji je plan (Korisnik.korisničkoIme)
opis	VARCHAR	opis plana treninga i prehrane
datumPočetka	TIMESTAMP	vrijeme početka plana
datumIsteka	TIMESTAMP	vrijeme kraja plana
cijena	DECIMAL	cijena plana
jeLiTrening	BOOLEAN	je li plan individualni plan, inače je samo kupljeni gotovi plan treninga i prehrane

Članarina Ovaj entitet sadrži sve važne informacije o članarini u teretani. Sadrži atribute: id, id pojedine teretane, cijena članarine, opis te trajanje, a ovaj je entitet u vezi *Many-to-One* s entitetom Teretana preko jedinstvenog brojčanog identifikatora teretane te je u vezi *One-to-Many* s entitetom ČlanarinaKlijent preko jedinstvenog brojčanog identifikatora članarine. Primarni ključ ovog entiteta je id te je stoga taj atribut boldan u tablici, dok je idTeretana strani ključ te je stoga taj atribut pisan formatom italic u tablici.

Članarina		
id	INT	jedinstveni brojčani identifikator članarine
<i>idTeretana</i>	INT	jedinstveni brojčani identifikator trenere koja nudi članarinu (Teretana.id)
cijena	DECIMAL	cijena članarine
opis	VARCHAR	opis članarine (što sve članarina uključuje)
trajanje	INTERVAL	trajanje članarine

ČlanarinaKlijent Ovaj entitet sadrži sve važne informacije o članarini koju klijent posjeduje. Sadrži atribute: id, korisničko ime klijenta, id članarine koju teretana nudi, datum početka te datum isteka trajanja članarine, a ovaj je entitet u vezi *Many-to-One* s entitetom Korisnik preko korisničkog imena klijenta te je u vezi *Many-to-One* s entitetom Članarina preko jedinstvenog brojčanog identifikatora članarine. Primarni ključ ovog entiteta je id te je stoga taj atribut boldan u tablici, dok su korisničkoImeKlijent i idČlanarina strani ključevi te su stoga ti atributi pisani formatom italic u tablici.

ČlanarinaKlijent		
id	INT	jedinstveni brojčani identifikator članarine koju klijent posjeduje
<i>korisničkoImeKlijent</i>	VARCHAR	korisničko ime klijenta koji posjeduje članarinu (Korisnik.korisničkoIme)
<i>idČlanarina</i>	INT	jedinstveni brojčani identifikator članarine (Članarina.id)
datumPočetka	TIMESTAMP	datum početka trajanja članarine
datumIsteka	TIMESTAMP	datum kraja trajanja članarine

Zamolba Ovaj entitet sadrži sve važne informacije o prijavi za posao koju trener šalje pojedinoj teretani. Sadrži atribute: id, korisničko ime trenera, id pojedine teretane u kojoj se trener za posao prijavljuje, te opis prijave, a ovaj je entitet u vezi *Many-to-One* s entitetom Korisnik preko korisničkog imena trenera te je u vezi *Many-to-One* s entitetom Teretana preko jedinstvenog brojčanog identifikatora teretane. Primarni ključ ovog entiteta je id te je stoga taj atribut boldan u tablici, dok su korisničkoImeTrener i idTeretana strani ključevi te su stoga ti atributi pisani formatom italic u tablici.

Zamolba		
id	INT	jedinstveni brojčani identifikator prijave
<i>korisničkoImeTrener</i>	VARCHAR	korisničko ime trenera koji se prijavljuje za posao u teretani (Korisnik.korisničkoIme)
<i>idTeretana</i>	INT	jedinstveni identifikator teretane u koju se prijavljuje (TeretanaLokacija.id)
opis	VARCHAR	opis prijave za posao
dozvola	BOOLEAN	je li treneru odobren rad u teretani

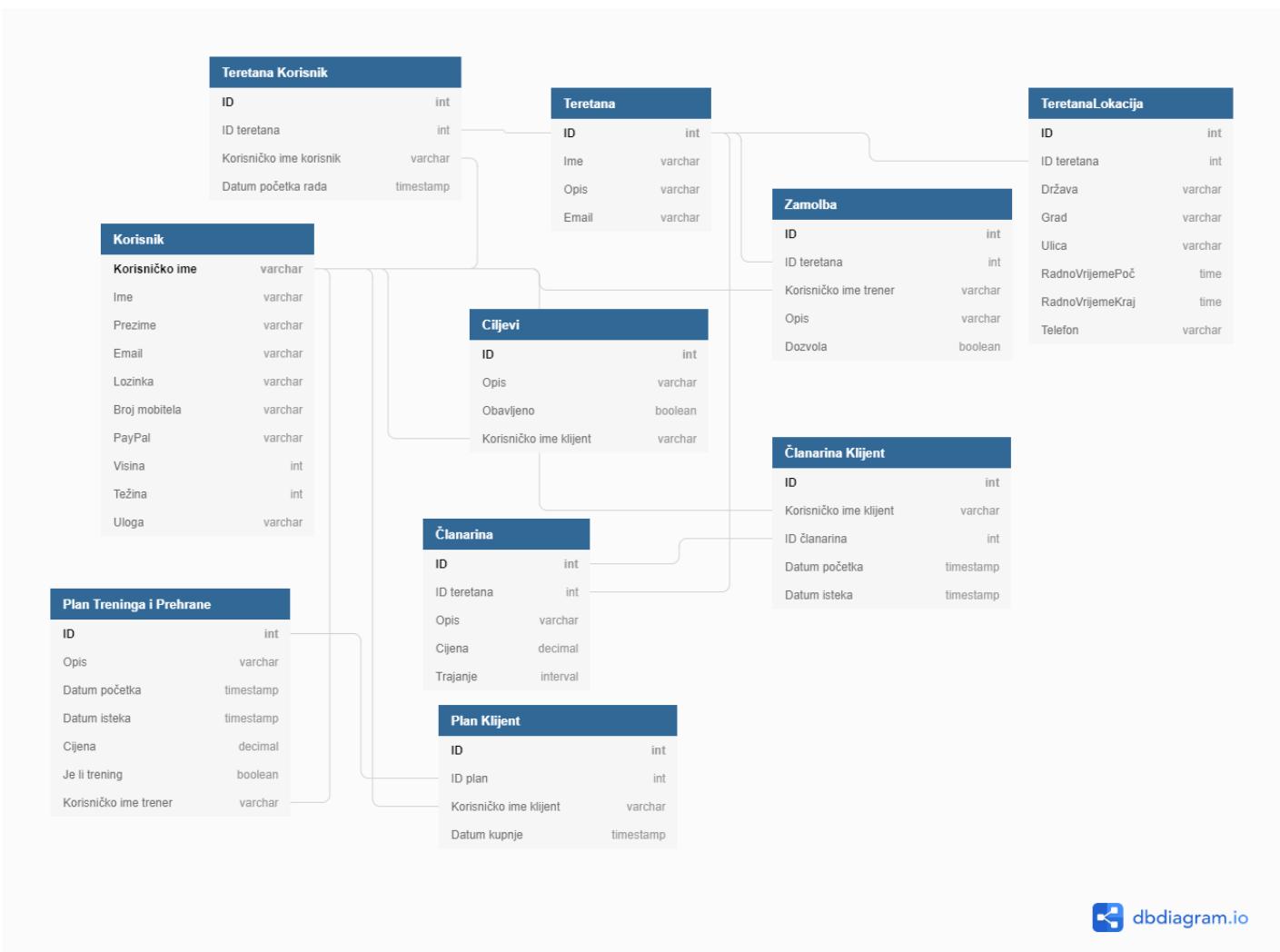
PlanKlijent Ovaj entitet sadrži sve važne informacije o planu prehrane i treninga kojeg klijent posjeduje. Sadrži atribute: id, id plana, korisničko ime klijenta, te datum kupnje, a ovaj je entitet u vezi *Many-to-One* s entitetom Korisnik preko korisničkog imena klijenta te je u vezi *Many-to-One* s entitetom PlanTreningaIPrehrane preko jedinstvenog brojčanog identifikatora plana. Primarni ključ ovog entiteta je id te je stoga taj atribut boldan u tablici, dok su korisničkoImeKlijent i idPlan strani ključevi te su stoga ti atributi pisani formatom italic u tablici.

PlanKlijent		
id	INT	jedinstveni brojčani identifikator plana kojeg klijent posjeduje
<i>idPlan</i>	INT	jedinstveni brojčani identifikator plana treninga i prehrane (PlanTreningaIPrehrane.id)
<i>korisničkoImeKlijent</i>	VARCHAR	korisničko ime klijenta koji plan posjeduje (Korisnik.korisničkoIme)
datumKupnje	TIMESTAMP	datum kupnje plana

TeretanaKorisnik Ovaj entitet sadrži sve važne informacije o korisniku koji radi u teretani kao voditelj ili kao trener. Sadrži atribute: id, id pojedine teretane, korisničko ime korisnika, te datum početka rada, a ovaj je entitet u vezi *Many-to-One* s entitetom Korisnik preko korisničkog imena korisnika te je u vezi *Many-to-One* s entitetom TeretanaLokacija preko jedinstvenog brojčanog identifikatora pojedine teretane. Primarni ključ ovog entiteta je id te je stoga taj atribut boldan u tablici, dok su idTeretana i korisničkoImeKorisnik strani ključevi te su stoga ti atributi pisani formatom italic u tablici.

TeretanaKorisnik		
id	INT	jedinstveni brojčani identifikator odnosa korisnika i teretane
<i>idTeretana</i>	INT	jedinstveni brojčani identifikator plana treninga i prehrane (TeretanaLokacija.id)
<i>korisničkoImeKorisnik</i>	VARCHAR	korisničko ime trenera ili voditelja u teretani (Korisnik.korisničkoIme)
<i>datumPočetkaRada</i>	TIMESTAMP	datum zaposlenja trenera u teretani

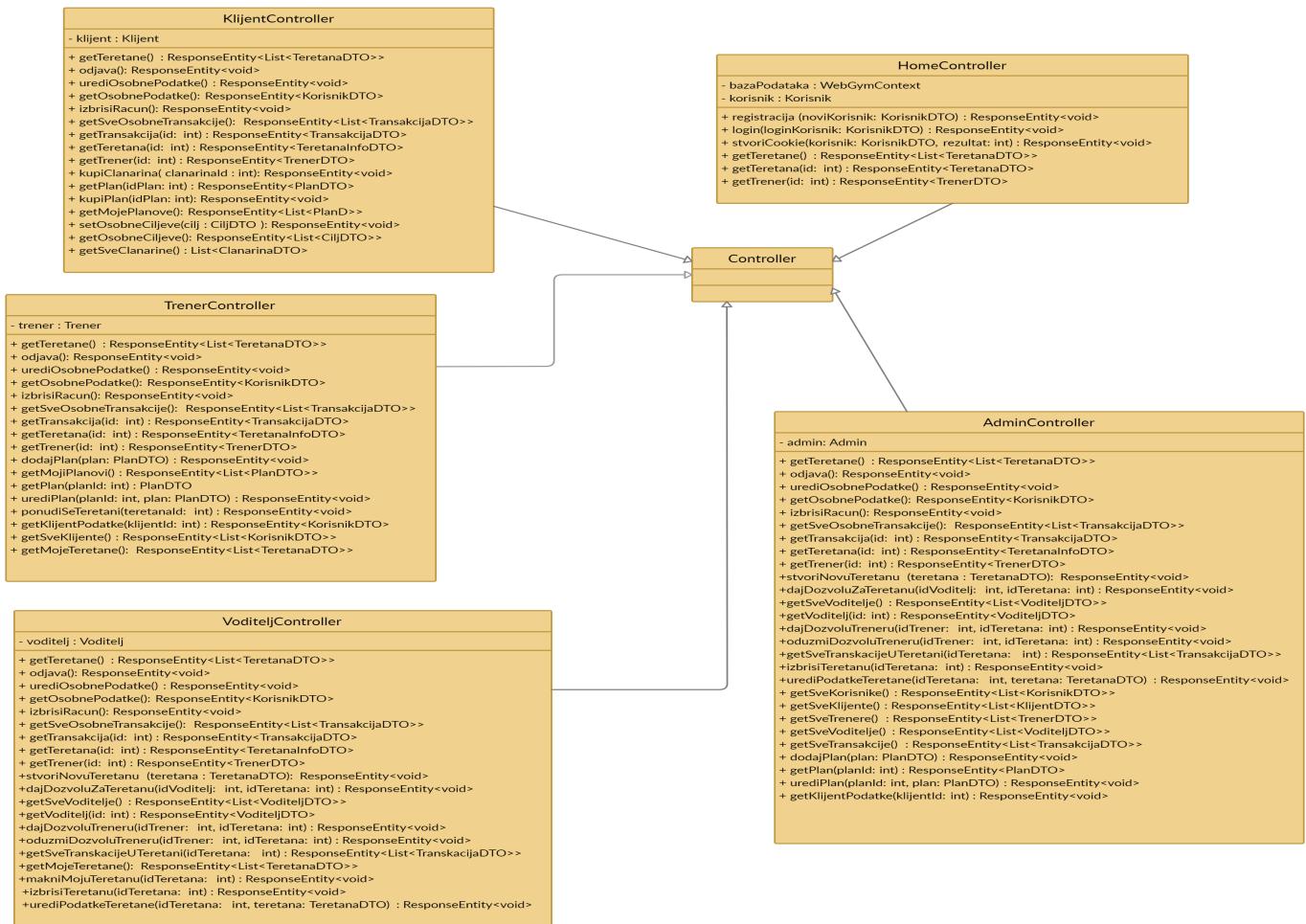
4.1.2 Dijagram baze podataka



Slika 4.2: E-R dijagram baze podataka

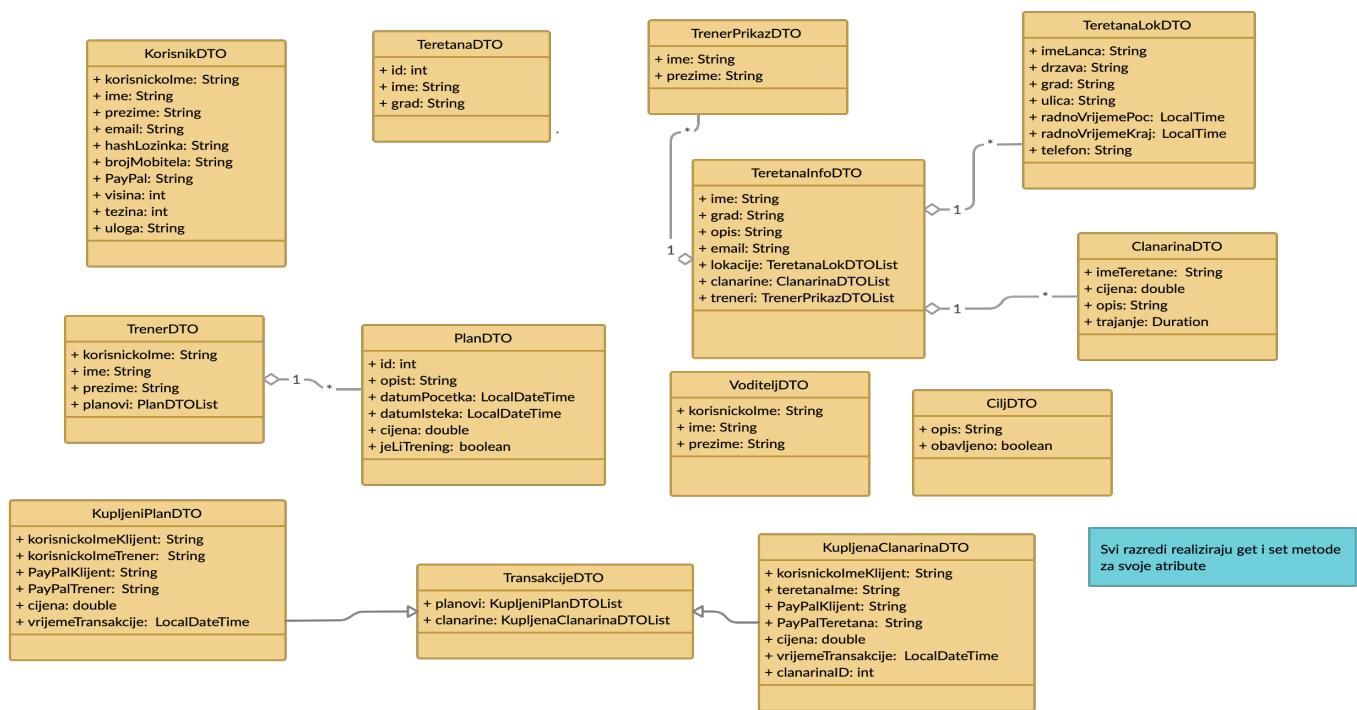
4.2 Dijagram razreda

Na slikama 4.3, 4.4 i 4.5 su prikazani razredi koji pripadaju Controller i Model dijelovima MVC arhitekture. Razredi prikazani na slici 4.3 prikazuju razrede koji nasljeđuju Controller. Unutar implementiranih controllera koriste se metode pomoću kojih se manipulira i "razgovara" s DTO (data transfer object). Funkcionalnost metoda unutar pojedinih controllera isprogramirana je da vraća JSON datoteke s html status kodom. Razredi su podijeljeni logički po imenima aktora u našoj web aplikaciji te implementirani s pripadajućim metodama, ovisno o ovlasti pojedinačnog aktora. Razredi prikazani na slici 4.4 predstavljaju DTO (data transfer object) razred. U području programiranja DTO je objekt koji prenosi podatke između procesa. Motivacija za njegovu upotrebu je ta da se komunikacija između procesa obično vrši pribjegavanjem udaljenim sučeljima (npr. Web uslugama), gdje je svaki poziv skupa operacija. Rješenje za smanjenje poziva je enkapsulacija podataka u vrijednosni objekt koji se može prenositi preko mreže, odnosno korištenje DTO-a.

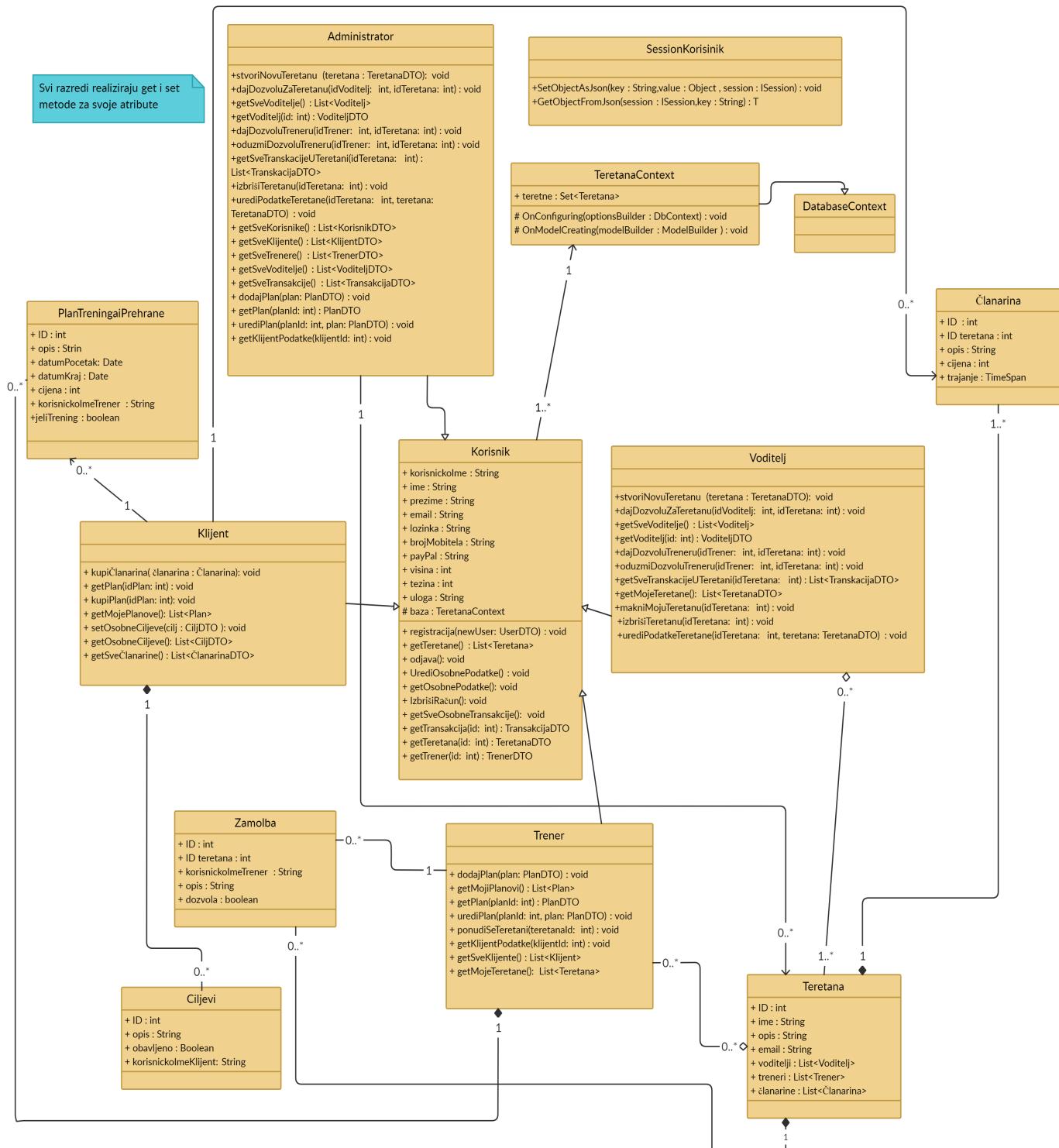


Slika 4.3: Dijagram razreda - dio Controllers

Model razredi preslikavaju strukturu baze podataka u aplikaciji. Implementirane metode direktno komuniciraju s bazom podataka te vraćaju tražene podatke. Razred Korisnik predstavlja uniju svih različitih entiteta koji se mogu registrirati u aplikaciji. Razred klijent predstavlja registriranog korisnika te on može koristiti sve rezervirane funkcionalnosti. Razred Administrator predstavlja administratora koji može koristiti više manje sve metode i ima najveće ovlasti te u suštini on upravlja cijelom aplikacijom. Razred Trener predstavlja trenera koji može raditi u nekoj od ponuđenih teretana te koristiti teretanu kao platformu za prodavanje svojih planova treninga i prehrane. Razred Voditelj predstavlja voditelja teretane, voditelj koristi svoje ovlasti kako bi osigurao pravilnu povezanost teretane, trenera i klijenata. Razred Zamolba predstavlja trenerov upit odnosno zamolbu za rad u određenoj teretani. Razred Ciljevi predstavlja određeni cilj pojedinog klijenta. Razred PlanTreningaIPrehrane predstavlja trenerov proizvod koji se nalazi u teretani u kojoj trener radi te ga mogu kupiti trenerovi klijenti.



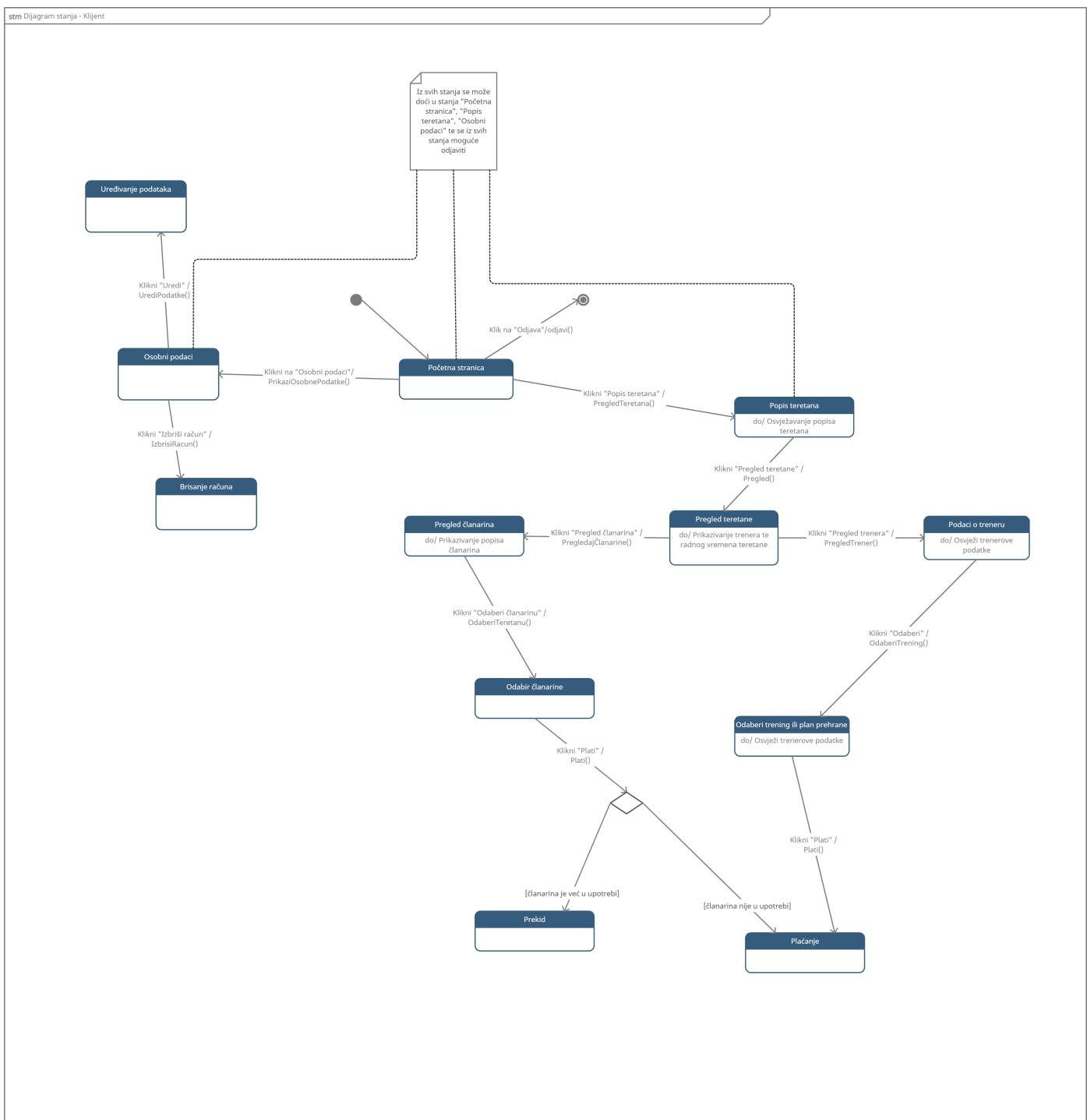
Slika 4.4: Dijagram razreda - dio Data transfer objects



Slika 4.5: Dijagram razreda - dio Models

4.3 Dijagram stanja

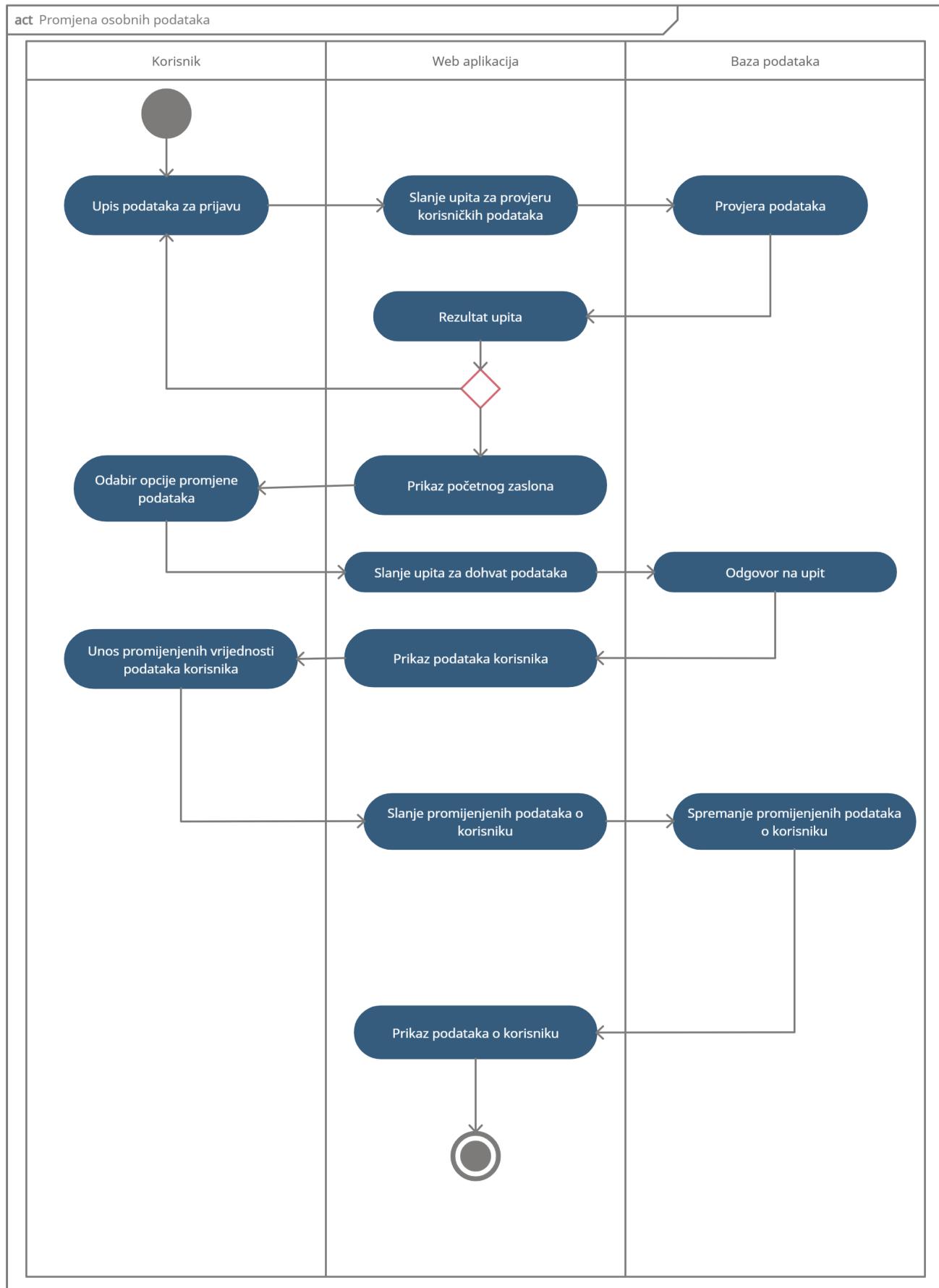
Dijagram stanja prikazuje stanje te sve prijelaze u ostala stanja uzrokovana događajima. Na slici 4.6 prikazan je dijagram stanja za registriranog korisnika. Kao što vidimo na dijagramu registriranog i prijavljenog korisnika na početnoj stranici čeka mogućnost pregleda svih teretana kao i pregled osobnih podataka. Ako korisnik izvrši akciju odlaska na stranicu pregleda osobnih podataka, ondje ima mogućnost izmijeniti svoje osobne podatke ili izbrisati račun na kojem je trenutno prijavljen. Ako korisnik odabere pregled teretana korisniku se prikazuju sve teretane. Korisnik tada ima mogućnost odabrati jednu od ponuđenih teretana nakon čega mu se prikaže popis trenera te teretane kao i radno vrijeme same teretane. Korisnik na odabranoj teretani također može odabrati i jednu od ponuđenih članarina koje se prikažu pritiskom na "pregled članarina" te se potom korisnika pritiskom na "plati" preusmjerava na plaćanje. U slučaju da korisnik već ima odabranu članarinu korisniku se dojavljuje da je odabранa članarina već u tijeku te se odlazi u stanje prekida u kojem će se provjeriti želi li se korisnik vratiti i odabrati novu članarinu. Ako se korisnik ipak pri odabranoj teretani odluči odabrati jednog od trenera pritiskom na "pregled trenera" korisniku se dohvaćaju trenerovi programi koje može odabrati. Pritiskom na "plati" na odabranom programu korisnika se usmjerava na plaćanje samog odabranog programa.



Slika 4.6: Dijagram stanja

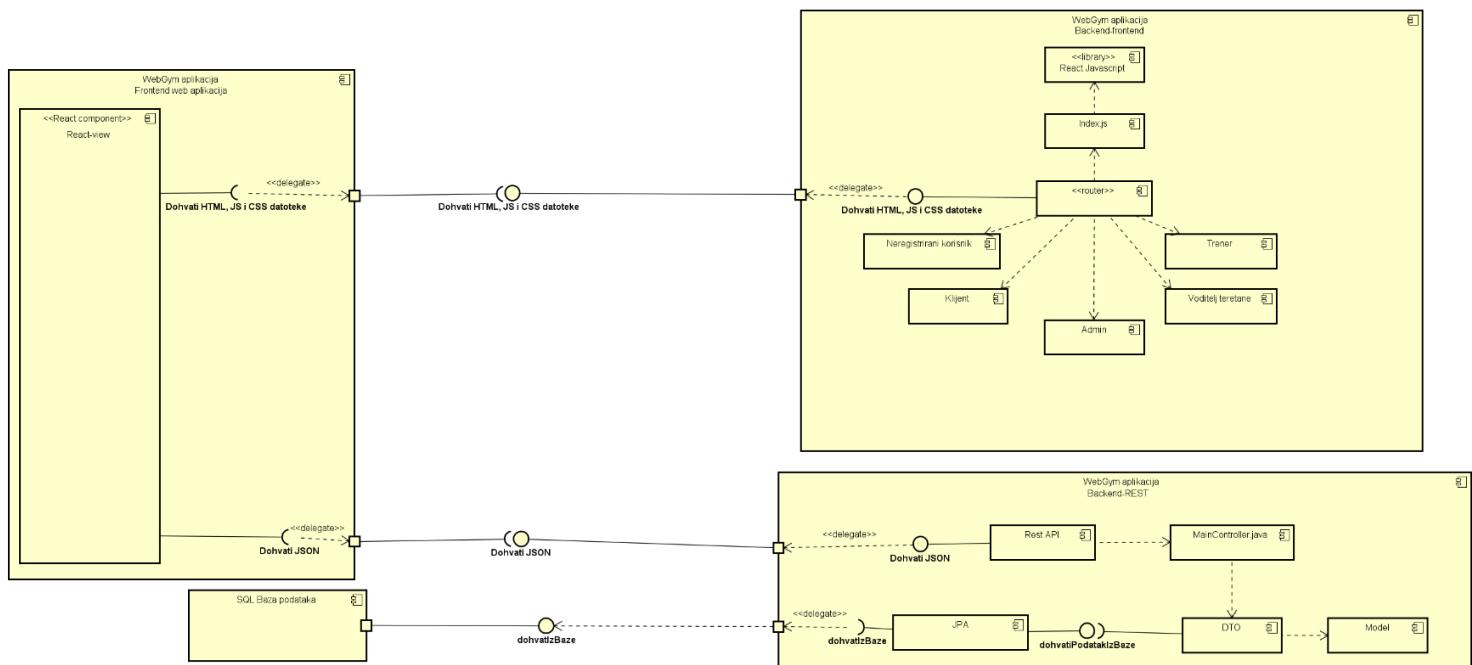
4.4 Dijagram aktivnosti

Dijagram aktivnosti pomaže nam u opisivanju toka upravljanja odnosno puta podataka. U modeliranju dijagrama aktivnosti tok upravljanja funkcioniра tako da se svaki novi korak izvršava nakon završenog prethodnog te je cilj prikazati tok tih koraka što jednostavnije. Na slici 4.7 prikazan je proces promjene podataka korisnika. Korisnik se prijavljuje u sustav, prikazuje mu se opcija pregleda osobnih podataka koju i odabire. Odabire opciju promjene osobnih podataka, unosi nove podatke o sebi te ih spremi. Spremljeni podaci šalju se od korisnika do web aplikacije, te ih web aplikacija šalje u bazu podataka gdje se spremaju. Ako je korisnik u početku unio krive osobne podatke za prijavu, korisnika se vraća na početnu stranicu.



4.5 Dijagram komponenti

Sljedeći dijagram prikazan na slici 4.8 jest dijagram komponenti koji prikazuje međuovisnost i organizaciju komponenti. Sustavu se pristupa preko nekoliko sučelja sa različitim ulogama. Prvo sučelje naziva se "Dohvati HTML, JS i CSS datoteke" kao što i samo ime sučelja govori, ono služi za dohvaćanje datoteka potrebnih frontendu kako bi korisniku prikazao potrebno. Router komponenta služi kako bi frontend znao kakav će prikaz biti, u ovom slučaju ovisno o ulozi korisnika u sustavu koju backend dohvaća iz baze. U frontend dijelu nalaze se JavaScript datoteke logički raspoređene te nazvane po ulogama u sustavu. Sve navedene JavaScript datoteke ovise o React biblioteci. Sučelje "Dohvati JSON", kao što nam i samo ime sučelja govori, služi za dohvrat JSON-a. Rest API poslužuje Backend-REST komponentu dok JPA služi za dohvrat podataka pomoću SQL upita. To se ostvaruje pomoću sučelja za dohvrat podataka iz baze. Pristigli podaci iz baze se oblikuju u DTO (Data transfer object) definirane u modelima u bakcend komponenti sustava. React pomoću svih navedenih sučelja kontinuirano komunicira sa backendom te tako prikazuje potrebno korisniku.



Slika 4.8: Dijagram komponenti

5. Implementacija i korisničko sučelje

5.1 Korištene tehnologije i alati

dio 2. revizije

Detaljno navesti sve tehnologije i alate koji su primijenjeni pri izradi dokumentacije i aplikacije. Ukratko ih opisati, te navesti njihovo značenje i mjesto primjene. Za svaki navedeni alat i tehnologiju je potrebno **navesti internet poveznicu** gdje se mogu preuzeti ili više saznati o njima.

Za svakodnevnu komunikaciju je korištena aplikacija WhatsApp¹ dok smo za sastanke i videopozive koristili platformu MS Teams². Za izradu UML dijagrama i svih tablica je korištena web aplikacija Creatly³. Dokumentacija je pisana pomoću online Latex editora Overleaf⁴. Kao sustav za upravljanje izvornim kodom korišten je Git⁵ dok se udaljeni repozitorij projekta nalazi na web platformi GitLab⁶.

Kao razvojno okruženje za backend je korišten IntelliJ⁷ - integrirano razvojno okruženje (IDE) napisano u Javi za razvoj računalnog softvera. Razvio ga je JetBrains (prije poznat kao IntelliJ) i dostupan je pod licencom Apache 2. Može se koristit na svim operacijskim sustavima, koristi se za razvoj web-stranica i web-aplikacija. Frontend je napisan u Visual Studio Code⁸ - source-code editor koji je razvijen od tvrtke Microsoft. Zaštićen je pod MIT licencom. VSC podržava programiranje u raznim jezicima kao što su Java, JavaScript, Go, Node.js and C++. Velika prednost je što se mogu instalirati proširenja koja mogu pružiti podršku za nove jezike, teme ili debuggere. Također podržava rad sa JSON, CSS i HTML tipovima podataka što omogućuje razvoj web-stranica i web-aplikacija.

Aplikacija je napisana koristeći radni okvir Spring Framework⁹ i jezik Java¹⁰

¹<https://www.whatsapp.com/>

²<https://www.microsoft.com/en-ww/microsoft-365/microsoft-teams/group-chat-software>

³<https://www.creatly.com/>

⁴<https://www.overleaf.com/>

⁵<https://git-scm.com/>

⁶<https://gitlab.com/>

⁷<https://www.jetbrains.com/idea/>

⁸<https://code.visualstudio.com/>

⁹<https://spring.io/>

¹⁰<https://www.java.com/en/>

za izradu backenda dok je za frontend korišten React¹¹ i jezik JavaScript¹². Spring Framework je otvorenog koda i sadrži gotove funkcionalnosti koji se mogu koristiti u svakoj Java aplikaciji, ali postoje i proširenja za izradu same aplikacije. React je JavaScript biblioteka koja se koristi za razvoj korisničkih sučelja i UI komponenti. Održavan je od strane Facebooka i zajednici programera i kompanija. Može se koristiti kao baza za jednostavne stranice ili aplikacije, ali za složeniju upotrebu je potrebno koristiti dodatne biblioteke kao što su Redux i React Router, jer se React zapravo samo bavi prikazom podataka u DOM-u.

Baza podataka se nalazi na poslužitelju u oblaku Heroku¹³.

¹¹<https://reactjs.org/>

¹²<https://developer.mozilla.org/en-US/docs/Web/JavaScript>

¹³<https://www.heroku.com/>

5.2 Ispitivanje programskog rješenja

U ovom poglavlju je potrebno opisati provedbu ispitivanja implementiranih funkcionalnosti na razini komponenti i na razini cijelog sustava s prikazom odabralih ispitnih slučajeva. Studenti trebaju ispitati temeljnu funkcionalnost i rubne uvjete.

5.2.1 Ispitivanje komponenti

Potrebno je provesti ispitivanje jedinica (engl. *unit testing*) nad razredima koji implementiraju temeljne funkcionalnosti. Razraditi **minimalno 6 ispitnih slučajeva** u kojima će se ispitati redovni slučajevi, rubni uvjeti te izazivanje pogreške (engl. *exception throwing*). Poželjno je stvoriti i ispitni slučaj koji koristi funkcionalnosti koje nisu implementirane. Potrebno je priložiti izvorni kôd svih ispitnih slučajeva te prikaz rezultata izvođenja ispita u razvojnem okruženju (*prolaz/pad ispita*).

Za ispitivanje razrednih funkcija koje implementiraju temeljne funkcionalnosti koristili smo MockMvc. MockMvc je razred definiran u javi pomoću kojeg možemo kontrolerima poslati lažne HTTP zahteve i testirati njihovo ponašanje bez pokretanja kontrolera unutar poslužitelja. U nastavku će biti prikazani neke od metoda koje smo testirali i odlučili prikazati.

1. Dohvati sve teretane

Ovdje smo testirali pokušaj dohvatanja teretana odlaskom na /getGyms.

```
@Test
public void getGymsTest() throws Exception {
    Gym gym = new Gym("Gyms4you", "opis", "gym4you@gmail.com");
    gymList.add(gym);

    when(gymService.listGyms()).thenReturn(gymList);

    mockMvc.perform( MockMvcRequestBuilders
        .get("/gymList")
        .accept(MediaType.APPLICATION_JSON)
        .andDo(print())
        .andExpect(status().isOk())
        .andExpect(MockMvcResultMatchers.jsonPath("$.exists()"))
        .andExpect(MockMvcResultMatchers.jsonPath("$[0].name").value("Gyms4you")));
}
```

2. Dodaj novu teretanu

Ovdje smo testirali pokušaj dodavanja nove teretane odlaskom na /addGym dok smo ulogirani kao role owner.

```
@Test
public void addNewGymTest() throws Exception {
    this.mockMvc.perform(post("/addGym")
        .contentType(MediaType.APPLICATION_JSON)
        .cookie(new Cookie("username","jJel"))
        .cookie(new Cookie("role","OWNER"))
        .content("{\"name\":\"Gyms4you\", \"description\": \"opis\", \"email\":\"gyms4you@gmail.com\"}")
        .andExpect(status().isOk()));
}
```

3. Dohvati trenerovu teretanu

Ovdje smo testirali pokušaj dohvaćanja teretane u kojoj radi određeni trener odlaskom na /myGyms sa trenutnim ulogiranim trenerom.

```
@Test
public void getCoachGym() throws Exception {
    GymDto gym = new GymDto();
    gym.setDescription("opis");
    gym.setEmail("gym4you@gmail.com");
    gym.setName("Gyms4you");
    gym.setId(1L);
    List<GymDto> gyms = new ArrayList<>();
    gyms.add(gym);

    when(gymService.getMyGyms("tLov")).thenReturn(gyms);

    this.mockMvc.perform(MockMvcRequestBuilders
        .get("/myGyms")
        .cookie(new Cookie("username","tLov"))
        .cookie(new Cookie("role","coach"))
        .accept(MediaType.APPLICATION_JSON))
        .andDo(print())
        .andExpect(MockMvcResultMatchers.jsonPath("[0].name").value("Gyms4you"))
        .andExpect(MockMvcResultMatchers.jsonPath("[0].description").value("opis"));

}
```

4. Dohvati informacije o teretani

Ovdje smo testirali pokušaj dohvaćanja informacija o određeneoj teretani odlaskom na /gymInfo sa pripadnjim parametrom id koji se šalje preko URL-a.

```
@Test
public void getGymsTest() throws Exception {
    Gym gym = new Gym("Gyms4you","opis","gym4you@gmail.com");
    gymList.add(gym);

    when(gymService.listGyms()).thenReturn(gymList);

    mockMvc.perform( MockMvcRequestBuilders
        .get("/gymList")
        .accept(MediaType.APPLICATION_JSON))
        .andDo(print())
        .andExpect(status().isOk())
        .andExpect(MockMvcResultMatchers.jsonPath("$.exists()"))
        .andExpect(MockMvcResultMatchers.jsonPath("[0].name").value("Gyms4you"));

}
```

5. Pronađi trenera

Ovdje smo testirali pokušaj dohvaćanja trenera odlaskom na /coach sa pripadnjim parametrom username koji se šalje preko URL-a.

```

@Test
public void findCoachById() throws Exception {
    CoachResponseDto response = new CoachResponseDto();
    UserDto user = new UserDto();
    user.setName("tomislav");
    response.setUser(user);

    when(coachService.getCoach("tLov")).thenReturn(response);

    this.mockMvc.perform(MockMvcRequestBuilders
        .get("/coach")
        .param("username", "tLov")
        .accept(MediaType.APPLICATION_JSON))
        .andDo(print())
        .andExpect(MockMvcResultMatchers.jsonPath("$.user.name").value("tomislav"));

}

```

6. Pronađi trenera (error)

Ovdje smo testirali pokušaj dohvaćanja trenera odlaskom na /coach bez parametra username kako bi dobili response 4xx koji javlja da je došlo do pogreške.

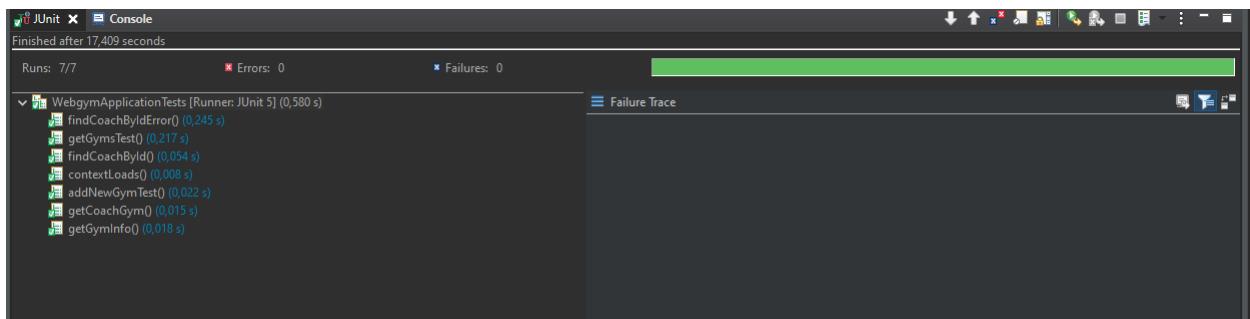
```

@Test
public void findCoachByIdError() throws Exception {
    this.mockMvc.perform(MockMvcRequestBuilders
        .get("/coach")
        .accept(MediaType.APPLICATION_JSON))
        .andDo(print())
        .andExpect(status().is4xxClientError());

}

```

Prikaz prolaza svih ispita



5.2.2 Ispitivanje sustava

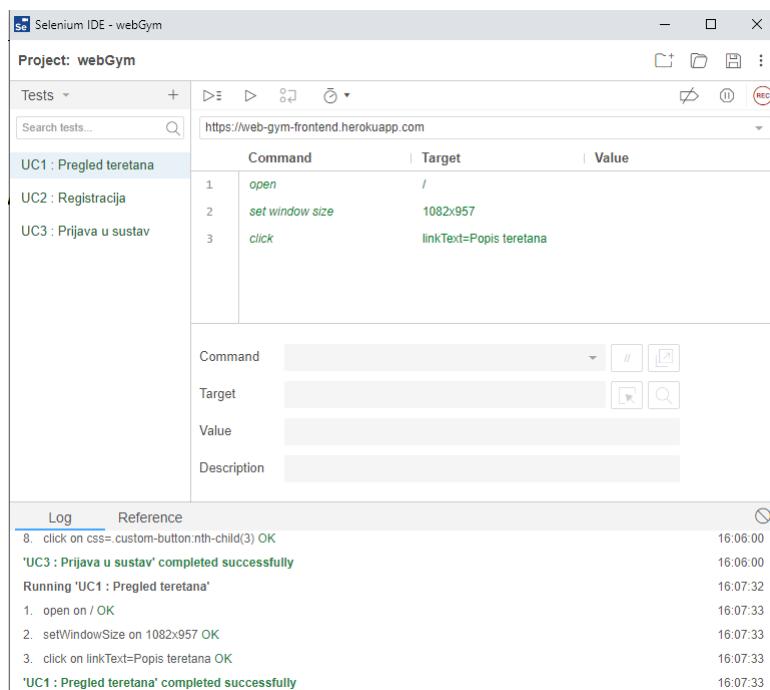
Potrebno je provesti i opisati ispitivanje sustava koristeći radni okvir. Razraditi **minimalno 4 ispitna slučaja** u kojima će se ispitati redovni slučajevi, rubni uvjeti te poziv funkcionalnosti koja nije implementirana/izaziva pogrešku kako bi se vidjelo na koji način sustav reagira kada nešto nije u potpunosti ostvareno. Ispitni slučaj se treba sastojati od ulaza (npr. korisničko ime i lozinka), očekivanog izlaza ili rezultata, koraka ispitivanja i dobivenog izlaza ili rezultata.

Za ispitivanje funkcionalnosti sustava koristili smo Selenium IDE. Testirali smo sve obrazce uporabe te smo prolazili kroz aplikaciju i gledali općenito ponašanje

u nadi da pronađemo neko neočekivano hazardno ponašanje u svrhu otklonuća problema. Neke od testiranih obrazaca uporabe smo prikazali u nastavku. (UC1, UC2, UC3, UC7).

1. Ispitni slučaj (UC1 : Pregled teretana)

- **Ulaz :**
 1. korisnik je pritisnuo na "popis teretana"
- **Očekivani rezultat:**
 1. korisnik je dobio prikaz svih teretana
- **rezultat :**
 1. rezultat je jednak očekivanom rezultatu.



2. Ispitni slučaj (UC2 : Registracija korisnika)

- **Ulaz :**
 1. korisnik je pritisnuo na "prijava"
 2. korisnik je ispunio obrazac "registracija"
 3. korisnik je pritisnuo "registriraj"
- **Očekivani rezultat:**
 1. korisnik se uspešno registrirao i spremio u bazu podataka
- **rezultat :**
 1. rezultat je jednak očekivanom rezultatu.

The screenshot shows the Selenium IDE interface with the project set to 'webGym'. A test named 'UC2 : Registracija' is currently selected. The test log displays the following steps:

	Command	Target	Value
13	click	css= username-and-password-registration	
14	click	name=passwordRegistration	
15	type	name=passwordRegistration	peroslav
16	click	css= custom-button:nth-child(5)	

Below the table, there are input fields for Command, Target, Value, and Description. The Log section shows the execution results for each step, indicating they were successful. The final message in the log is "'UC2 : Registracija' completed successfully".

3. Ispitni slučaj (UC3 : Prijava u sustav)

- **Ulaz :**
 1. korisnik je pritisnuo na "prijava"
 2. korisnik je ispunio obrazac "prijava"
 3. korisnik je pritisnuo "prijava"
- **Očekivani rezultat:**
 1. korisnik se uspješno prijavio u sustav
- **rezultat :**
 1. rezultat je jednak očekivanom rezultatu.

Selenium IDE - webGym

Project: webGym

Tests + D E Run current test Ctrl+R d.herokuapp.com

	Command	Target	Value
UC1 : Pregled teretana	click	name=usernameLogin	
UC2 : Registracija	type	name=usernameLogin	tLov
UC3 : Prijava u sustav	click	name=passwordLogin	
	type	name=passwordLogin	456
	click	css=custom-button:nth-child(3)	

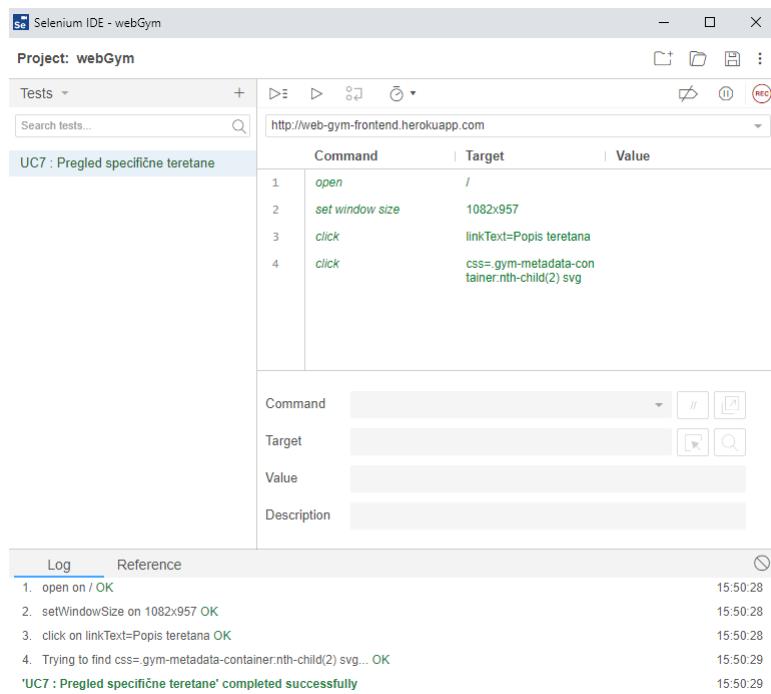
Command Target Value Description

Log Reference

3. click on linkText=Prijava OK 16:09:28
4. click on name=usernameLogin OK 16:09:29
5. type on name=usernameLogin with value tLov OK 16:09:29
6. click on name=passwordLogin OK 16:09:29
7. type on name=passwordLogin with value 456 OK 16:09:29
8. click on css=custom-button:nth-child(3) OK 16:09:29
'UC3 : Prijava u sustav' completed successfully 16:09:30

4. Ispitni slučaj (UC7 : Pregled specifične teretane)

- **Ulaz :**
 1. korisnik je pritisnuo na "Popis teretana"
 2. korisnik je pritisnuo na gumb informacije
- **Očekivani rezultat:**
 1. korisnik je dobio informacije o odabranoj teretani
- **rezultat :**
 1. rezultat je jednak očekivanom rezultatu.



5. Ispitni slučaj (UC20 : Pregled trenereove ponude)

- **Ulaz :**
 1. trener je pritisnuo na "Moji planovi"
- **Očekivani rezultat:**
 1. trener je dobio prikaz svih svojih planova treninga
- **rezultat :**
 1. došlo je do pogreške te se nisu učitali trenerovi planovi.
 - Test je napravljen prije implementiranja mogućnosti trenera da pregleda svoje planove treninga i prehrane , do greške je došlo zbog problema u pamćenju cookie-a koje smo naknadno riješili.

The screenshot shows the Selenium IDE interface with the following details:

Project: webGym*

Test Name: UC20 : Pregled trenerove ponude*

URL: http://web-gym-frontend.herokuapp.com

Script Content:

Command	Target	Value
5 type	name=usernameLogin	tLov
6 click	name=passwordLogin	
7 type	name=passwordLogin	456
8 click	css=.custom-button:nt h-child(3)	
9 click	linkText=Moji planovi	
10 assert alert	Došlo je do pogreške	
11 assert alert	Došlo je do pogreške	

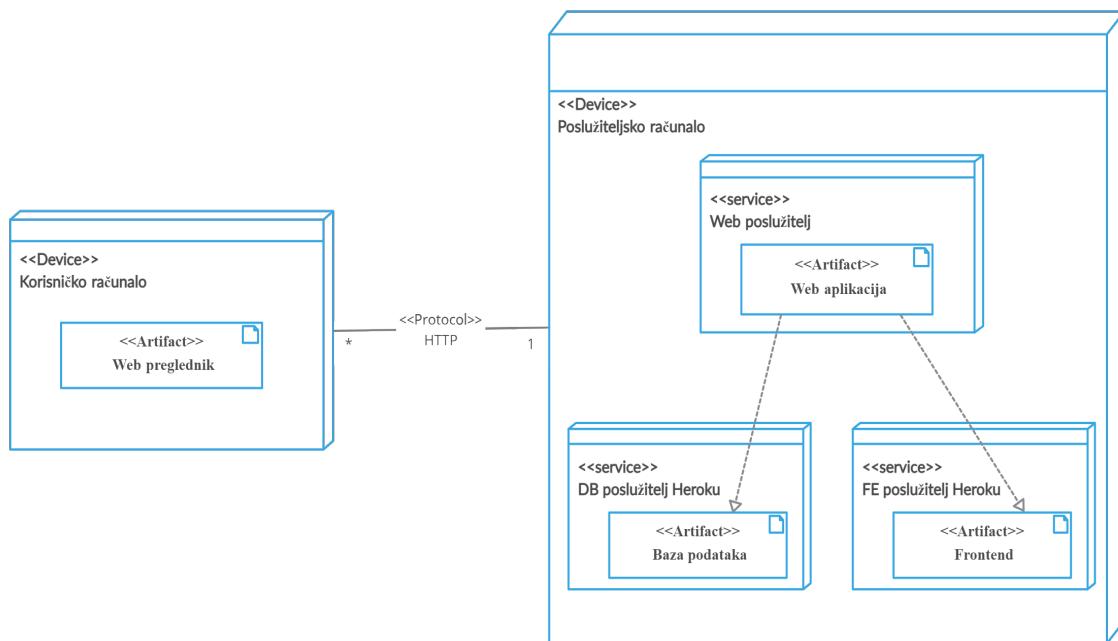
Log:

- 9. Trying to find linkText=Moji planovi... OK 15:21:44
- 10. assertAlert on Došlo je do pogreške OK 15:21:45
- 11. assertAlert on Došlo je do pogreške Undetermined 15:21:45
Aborting...
- 'UC20 : Pregled trenerove ponude' was aborted 15:23:03

5.3 Dijagram razmještaja

Potrebno je umetnuti **specifikacijski dijagram razmještaja** i opisati ga. Moguće je umjesto specifikacijskog dijagrama razmještaja umetnuti dijagram razmještaja instanci, pod uvjetom da taj dijagram bolje opisuje neki važniji dio sustava.

Dijagram razmještaja prikazuje generalnu topologiju sustava koji se koristi kako bi resursi bili optimalno raspoređeni. Predstavlja statički pogled na razmještaj sklopovskih i programskih komponenata. Na poslužiteljskom računalu se nalaze web poslužitelj, poslužitelj baze podataka i poslužitelj za frontend. Klijenti koriste web preglednik kako bi pristupili web aplikaciji. U bazi podataka sadržane su sve potrebne informacije i datoteke koje korisnik može zahtijevati. Sustav je baziran na arhitekturi "klijent – poslužitelj". Komunikacija između poslužiteljskog računala i klijentskog računala se omogućuje HTTP protokolom.



Slika 5.1: Dijagram razmještaja

5.4 Upute za puštanje u pogon

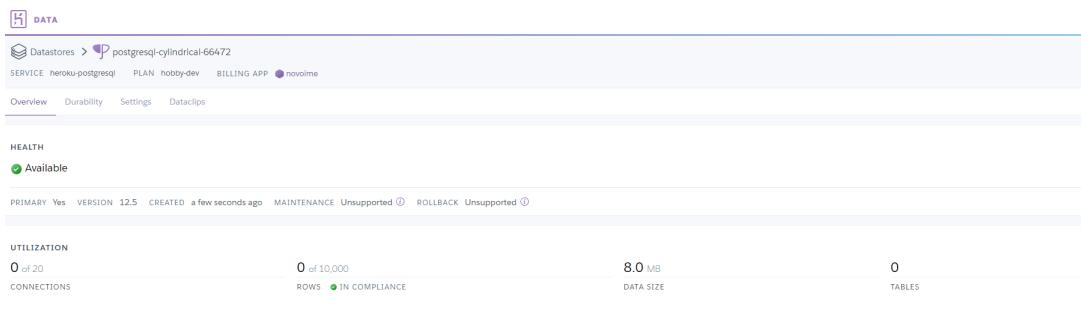
- **1. Stvaranje backend heroku servera**
 1. Stvaranje account-a na <https://dashboard.heroku.com/>
 2. Instalacija Heroku CLI : <https://devcenter.heroku.com/articles/heroku-cli>
 3. Otvaranje command prompt-a
 4. Upiši "heroku create "NAME-OF-APP" (webGym) ili samo "heroku create"
 - U slučaju da se ne navede ime aplikacije , i dalje će se stvoriti server ali s random imenom koji je kasnije moguće preimenovati u željeno ime.
 5. Stvoren je server.

```
C:\Users\Tomislav Lovrencic>heroku create
Creating app... done, ⚡ mysterious-sierra-72296
https://mysterious-sierra-72296.herokuapp.com/ | https://git.heroku.com/mysterious-sierra-72296.git

C:\Users\Tomislav Lovrencic>heroku apps:rename novoime --app mysterious-sierra-72296
Renaming mysterious-sierra-72296 to novoime... done
https://novoime.herokuapp.com/ | https://git.heroku.com/novoime.git
!   Don't forget to update git remotes for all other local checkouts of the app.

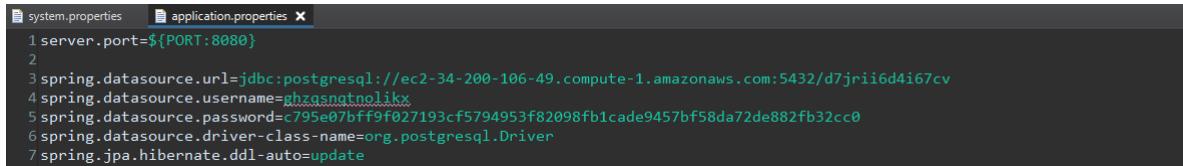
C:\Users\Tomislav Lovrencic>
```

- **2. Konfiguracija baze podataka**
 1. Odlazak na stranicu <https://dashboard.heroku.com/>
 2. Klik na stvoreni server. (novoime)
 3. Klik na resources
 4. Pronaći pod add-ons Heroku POSTGRES
 - Odabratи Hobby dev - FREE
 5. Stisnuti na SUBMIT FORM.
 6. Stvorena je baza podataka



- **3. Spajanje backend server na stvorenu bazu u heroku**
 1. Odlazak na stranicu <https://dashboard.heroku.com/>
 2. Klik na stvoreni server. (novoime)

3. Klik na resources
4. Klik na Heroku POSTGRES
5. Klik na settings
6. Klik na view credentials
7. Otvori backend kod (java)
8. Odi u src/main/resources/application.properties
9. Podesi :
 - Potrebno je podesiti backend tako da se povezuje sa bazom koju smo u prethodnom koraku stvorili na heroku.
 - U tom naumu ćemo morati podesiti spring.datasource.url , spring.datasource.username te spring.datasource.password na način da ćemo pročitati vrijednosti koje nam je dao heroku credentials i nadodati ih na spomenute vrijednosti.
 - Zadnja napomena je dodavanje "jdbc:" na početak URI-a koje nam je generirao heroku.



```

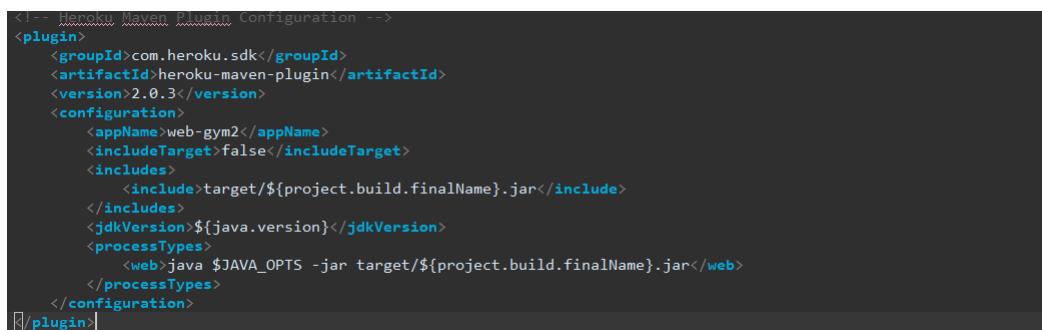
system.properties
application.properties x

1 server.port=${PORT:8080}
2
3 spring.datasource.url=jdbc:postgresql://ec2-34-200-106-49.compute-1.amazonaws.com:5432/d7jrri6d4i67cv
4 spring.datasource.username=ghzsgntnolikx
5 spring.datasource.password=c795e07bff9f027193cf5794953f82098fb1cade9457bf58da72de882fb32cc0
6 spring.datasource.driver-class-name=org.postgresql.Driver
7 spring.jpa.hibernate.ddl-auto=update

```

- 4. deploy backend-a

1. Otvoriti backend project.
2. Otvoriti pom.xml te dodati ovaj plugin.



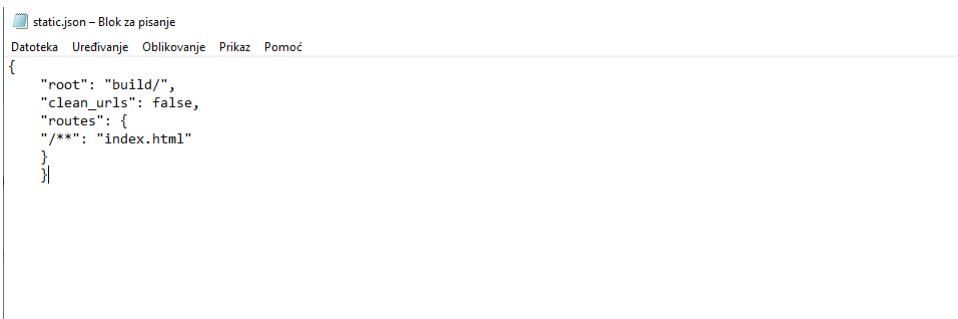
```

<!-- Heroku Maven Plugin Configuration -->
<plugin>
  <groupId>com.heroku.sdk</groupId>
  <artifactId>heroku-maven-plugin</artifactId>
  <version>2.0.3</version>
  <configuration>
    <appName>web-gym2</appName>
    <includeTarget>false</includeTarget>
    <includes>
      <include>target/${project.build.finalName}.jar</include>
    </includes>
    <jdkVersion>${java.version}</jdkVersion>
    <processTypes>
      <web>java $JAVA_OPTS -jar target/${project.build.finalName}.jar</web>
    </processTypes>
  </configuration>
</plugin>

```

- Napomena : u našem slučaju prateći sve ove korake bi treabli napisati pod appName "novoime", dok nam je za projekt ime aplikacije ,kao što možemo vidjeti na slici "web-gym2".
3. Otvoriti cmd te se pozicionirati u root backend projekta
 4. Napisati "mvn clean heroku:deploy"

- **5. Stvaranje frontend servera te deploy frontend-a**
 1. Odlazak na stranicu <https://dashboard.heroku.com/>
 2. Otvoriti cmd te upisati "heroku create NAME-OF-FRONTEND-APP -- buildpack <https://github.com/mars/create-react-app-buildpack.git>"
 3. Napraviti prazan folder , unutar foldera u cmd-u napisati "git init"
 4. U cmd-u napisati "heroku git:remote -a NAME-OF-FRONTEND-APP"
 5. Prekopirati root frontend projekta u stvoreni folder
 6. Dodati static.json datoteku u root frontend projekta



The screenshot shows a code editor window titled "static.json – Blok za pisanje". The menu bar includes "Datoteka", "Uređivanje", "Oblikovanje", "Prikaz", and "Pomoć". The code content is as follows:

```
{  
  "root": "build/",  
  "clean_urls": false,  
  "routes": {  
    "/*": "index.html"  
  }  
}
```

7. prije deploja frontenda , uči u src/App.js te promjeniti backendURL i postaviti ga na server backenda.

```
constructor(){  
  super()  
  this.state = {  
    backendURL: "https://web-gym2.herokuapp.com/",
```

8. U cmd- napisati "git add ."
9. U cmd- napisati "git add -m "initial commit""
10. U cmd- napisati "git push heroku master"

6. Zaključak i budući rad

Zadatak naše grupe bio je napraviti web aplikaciju za pronađazak teretana i njihovih informacija u svrhu učlanjivanja ili pronađenja trenera i trenerovih ponuda za programe treniranja i planova prehrana. Mogućnosti ove aplikacije su velike te mogu značajno olakšati odabir teretane po subjektivnim kriterijima. Na izradi aplikacije sudjelovalo je 7 osoba u trajanju od 17 tjedana. Za pravilnu i sistematičnu podjelu posla zaslužan je naš voditelj koji je stvorio tri podtimu koja su djelovali kao homogena zajednica i uz kontinuiranu međusobnu komunikaciju stvorili završni proizvod. Rad na projektu možemo opisati kroz dvije veće faze , u kojima je bilo manjih, povezanih faza. U prvoj fazi projekta tim se dogovarao o tehnologijama i tehničkoj podrški koju ćemo koristiti za izradu ove aplikacije , uz dogovore potrošen je dio vremena za individualno učenje i usavršavanje spomenutih tehnologija. Nakon detaljnih upoznavanja , kako sa tehnologijom tako i sami tim međusobno , krenuli smo u stvaranje aplikacije. Glavni cilj prve faze bio nam je stvoriti kostur projekta odnosno napraviti dobru dokumentaciju po kojoj ćemo onda graditi našu aplikaciju. U drugoj fazi projekta dolazi do izražaja pravilan raspored timova na određenim područjima , tako smo imali podtim za backend , za frontend i za dovršavanje dokumentacije. Unatoč velikoj organiziranosti i pravilnoj podjeli posla u drugoj fazi projekta nailazili smo na dosta problema vezane uz tehnologiju , te je u tim trenutcima do izražaja došao samostalni rad na učenje dodatnih materijala i alata kako bi riješili ove probleme. Najčešći problemi na koje smo nailazili bili su vezani za radni okvir react s kojim se tek upoznali i krenuli učiti za ovaj projekt. Tijekom izrade projekta shvatili smo da je ključan dio izrade tehničke aplikacije bio pravilna i transparentna dokumentacija te kontinuirana komunikacija između podtimova. Komunikaciju smo vršili preko discorda koji je omogućio lagane i brze sastanke za probleme koje je trebalo riješiti. TODO Sudjelovanje i rad na ovom projektu stvorilo je nove poglede na rad u timu te je stečeno vrijedno radno iskustvo u stvaranju web aplikacije. Za kraj možemo reći da je ovo bilo brdovito putovanje na koje smo krenuli nepremni ali znatiželjni i uporni te smo prilično zadovoljni završnom aplikacijom no smatramo da postoji

puno prostora za unapređenje i poboljšanje aplikacije.

Popis literature

Kontinuirano osvježavanje

Popisati sve reference i literaturu koja je pomogla pri ostvarivanju projekta.

1. Programsко inženjerstvo, FER ZEMRIS, <http://www.fer.hr/predmet/proinz>
2. I. Sommerville, "Software engineering", 8th ed, Addison Wesley, 2007.
3. T.C.Lethbridge, R.Langaniere, "Object-Oriented Software Engineering", 2nd ed. McGraw-Hill, 2005.
4. I. Marsic, Software engineering book“, Department of Electrical and Computer Engineering, Rutgers University, <http://www.ece.rutgers.edu/~marsic/books/SE>
5. The Unified Modeling Language, <https://www.uml-diagrams.org/>
6. Astah Community, <http://astah.net/editions/uml-new>

Indeks slika i dijagrama

2.1 Početna stranica https://www.trainerize.me/	5
2.2 Pretraživanje trenera https://www.trainerize.me/	5
2.3 Početna stranica https://ultimateperformance.com/	5
3.1 Sekvencijski dijagram za UC10	26
3.2 Sekvencijski dijagram za UC13	27
3.3 Sekvencijski dijagram za UC15	28
3.4 Sekvencijski dijagram za UC17	29
4.1 Arhitektura sustava	31
4.2 E-R dijagram baze podataka	39
4.3 Dijagram razreda - dio Controllers	41
4.4 Dijagram razreda - dio Data transfer objects	42
4.5 Dijagram razreda - dio Models	43
4.6 Dijagram stanja	45
4.7 Dijagram aktivnosti	47
4.8 Dijagram komponenti	48

Dodatak: Prikaz aktivnosti grupe

Dnevnik sastajanja

Kontinuirano osvježavanje

U ovom dijelu potrebno je redovito osvježavati dnevnik sastajanja prema predlošku.

1. sastanak

- 7.listopada 2020.
- Prisustvovali: L.Merćep, J.Milić, E.Đuras, F.Sodić, D.Kniewald, T.Lovrenčić, B.Dević
- Teme sastanka:
 - Inicijalna pitanja asistentu za projekt

2. sastanak

- 14.listopada 2020.
- Prisustvovali: L.Merćep, J.Milić, E.Đuras, F.Sodić, D.Kniewald, T.Lovrenčić, B.Dević
- Teme sastanka:
 - Podjela tima na backend i frontend

3. sastanak

- 20.listopada 2020.
- Prisustvovali: E.Đuras, F.Sodić, D.Kniewald
- Teme sastanka:
 - Frontend
 - Istraživanje react tehnologije

4. sastanak

- 24.listopada 2020.
- Prisustvovali: L.Merćep, J.Milić, T.Lovrenčić, B.Dević
- Teme sastanka:
 - Backend
 - Istraživanje Spring Boot i Heroku tehnologija

5. sastanak

- 30.listopada 2020.
- Prisustvovali: L.Merćep, J.Milić, D.Kniewald, T.Lovrenčić
- Teme sastanka:
 - Dokumentacija
 - Revizija Funkcionalnih zahtjeva koje je F.Sodić napisao
 - Obrasci uporabe

Tablica aktivnosti

Kontinuirano osvježavanje

Napomena: Doprinose u aktivnostima treba navesti u satima po članovima grupe po aktivnosti.

	Luka Merćep	Jura Milić	Eduard Đuras	Fran Sodić	Dominik Kniewald	Tomislav Lovrenčić	Bruno Dević
Upravljanje projektom	7		3				
Opis projektnog zadatka	5						
Funkcionalni zahtjevi				5			
Opis pojedinih obrazaca	3	3			5	5	
Dijagram obrazaca						4	
Sekvencijski dijagrami							4
Opis ostalih zahtjeva				2			
Arhitektura i dizajn sustava			2				3
Baza podataka	3			2	2	4	4
Dijagram razreda	2					5	
Dijagram stanja							
Dijagram aktivnosti							
Dijagram komponenti							
Korištene tehnologije i alati							
Ispitivanje programskog rješenja							
Dijagram razmještaja							
Upute za puštanje u pogon							
Dnevnik sastajanja							
Zaključak i budući rad							
Popis literature							

	Lukka Merćep	Jura Milić	Eduard Duras	Fran Sodić	Dominik Kniewald	Tomislav Lovrenčić	Bruno Dević
Dodatne stavke kako ste podijelili izradu aplikacije							
Izrada početne stranice			15	7			
Izrada baze podataka		5					
Spajanje s bazom podataka		4				2	
Back end		5					
Čitanje i provjera dokumentacije				5	2		

Dijagrami pregleda promjena

dio 2. revizije

Prenijeti dijagram pregleda promjena nad datotekama projekta. Potrebno je na kraju projekta generirane grafove s gitlaba prenijeti u ovo poglavlje dokumentacije. Dijagrami za vlastiti projekt se mogu preuzeti s gitlab.com stranice, u izborniku Repository, pritiskom na stavku Contributors.