Sadržaj

Uvod

U današnje vrijeme sjedilačkog načina života sport je jako važan za zdravlje ljudi. U zadnjih nekoliko godina velik je naglasak na promoviranje fitnessa kao oblika rekreacije. Osim toga što se i sam bavim sportom cijeli život, to me potaklo da prikažem svoje znanje stečeno u prethodne tri godine studija u obliku web-aplikacije za trenažni proces u teretani.

Aplikacija je specijalizirana za pohranu i prikaz podataka o odrađenim treninzima, vježbama u treninzima, postignutim rekordima i izmjerenim mjerama kao što su visina, težina, opseg struka itd. Osim podataka o treningu, aplikacija omogućava pregled svih pohranjenih teretana preko kojih korisnik može dobiti uvid koji trener radi u kojoj teretani.

Aplikacija sustavno sadrži dvije uloge: klijent i administrator. Klijent može upravljati svojim treninzima, mjerama i rekordima, dok administrator može dodavati uređivati i brisati one podatke koje klijent ne može.

Prvo poglavlje bavi se specifikacijom zahtjeva. Objasnit će se koje uloge postoje u aplikaciji i koje su njihove mogućnosti, što će se ilustrirati dijagramima obrazaca uporabe. Drugo poglavlje govori o bazi podataka uz priloženi ER i relacijski model. Objasnit će se struktura baze podataka, njezine tablice, model i ograničenja. U trećem i posljednjem poglavlju slijedi opis cjelokupne izrade ovog projekta, nakon čega su objašnjene funkcionalnosti aplikacije uz priloženi vodič kroz cijelu web-aplikaciju. Nakon toga u zaključku je dana završna riječ, nakon čega slijedi sažetak projekta.

1. Specifikacija zahtjeva

1.1. Analiza zahtjeva

Na početku oblikovanja ovakve web-aplikacije moramo ustanoviti kome je ona namijenjena i tko će ju koristiti. Osim toga, moramo specificirati skupine korisnika i definirati njihove zahtjeve i mogućnosti. U ovoj web-aplikaciji postoje dvije skupine korisnika: administrator i klijent.

Klijent je primarni korisnik sustava. Nakon što obavi registraciju i prijavu u sustav, on može:

- Dodavati, uređivati i brisati trening
- Dodavati i brisati vježbe u određenom treningu
- Pregledati sve svoje prijašnje treninge
- Dodavati i brisati svoje rekorde u pojedinoj vježbi
- Dodavati, uređivati i brisati svoje mjere (težina, opseg struka, opseg ruke…)
- Uređivati osobne podatke
- Pregledati sve dostupne teretane u aplikaciji te vidjeti koji trener radi u kojoj teretani

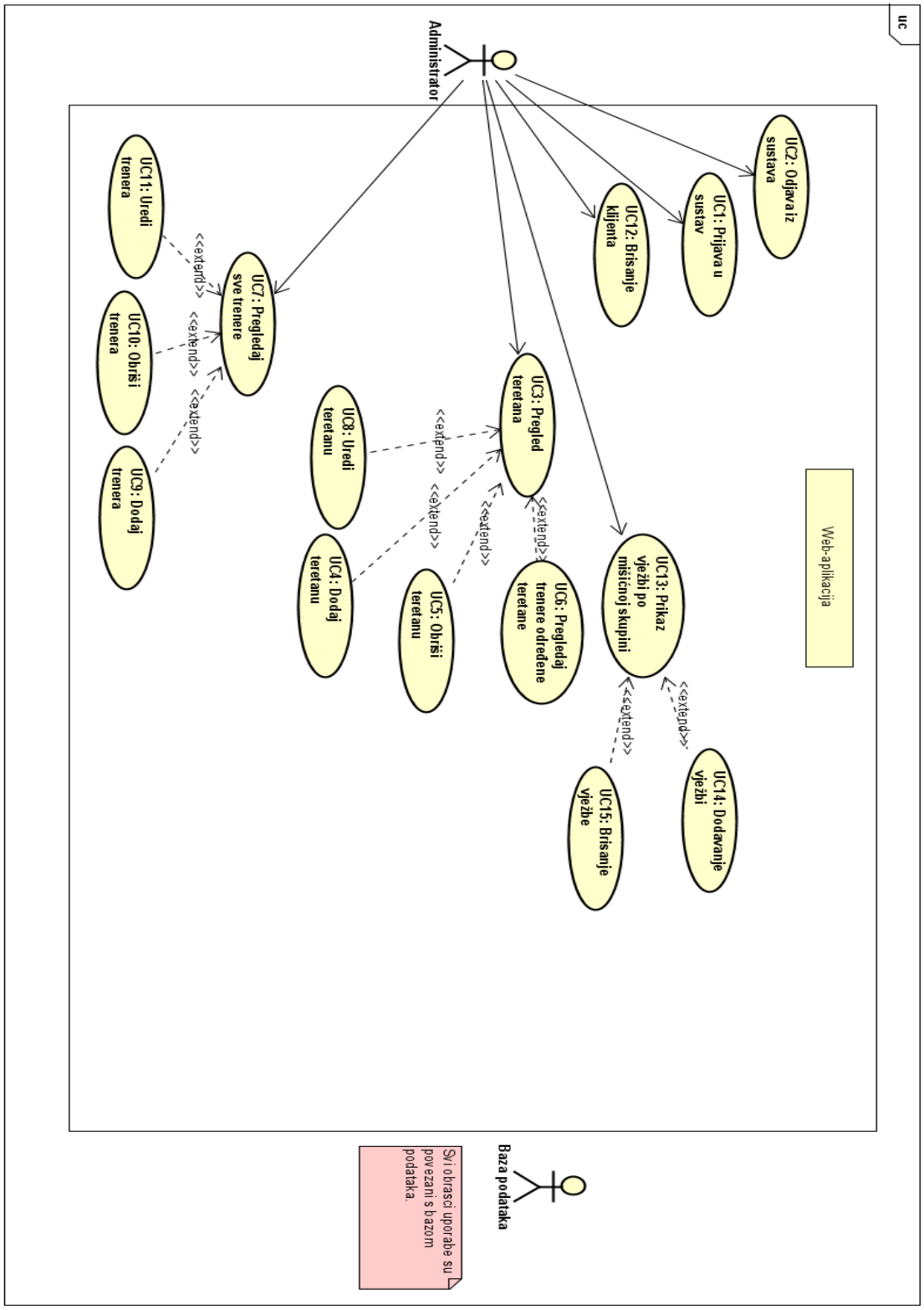
S druge strane, administrator ima opcije:

- Dodavanja teretana, trenera i vježbi
- Brisanja klijenata, teretana, trenera i vježbi
- Uređivanja teretana i trenera
- Prikaza koji trener radi u kojoj teretani

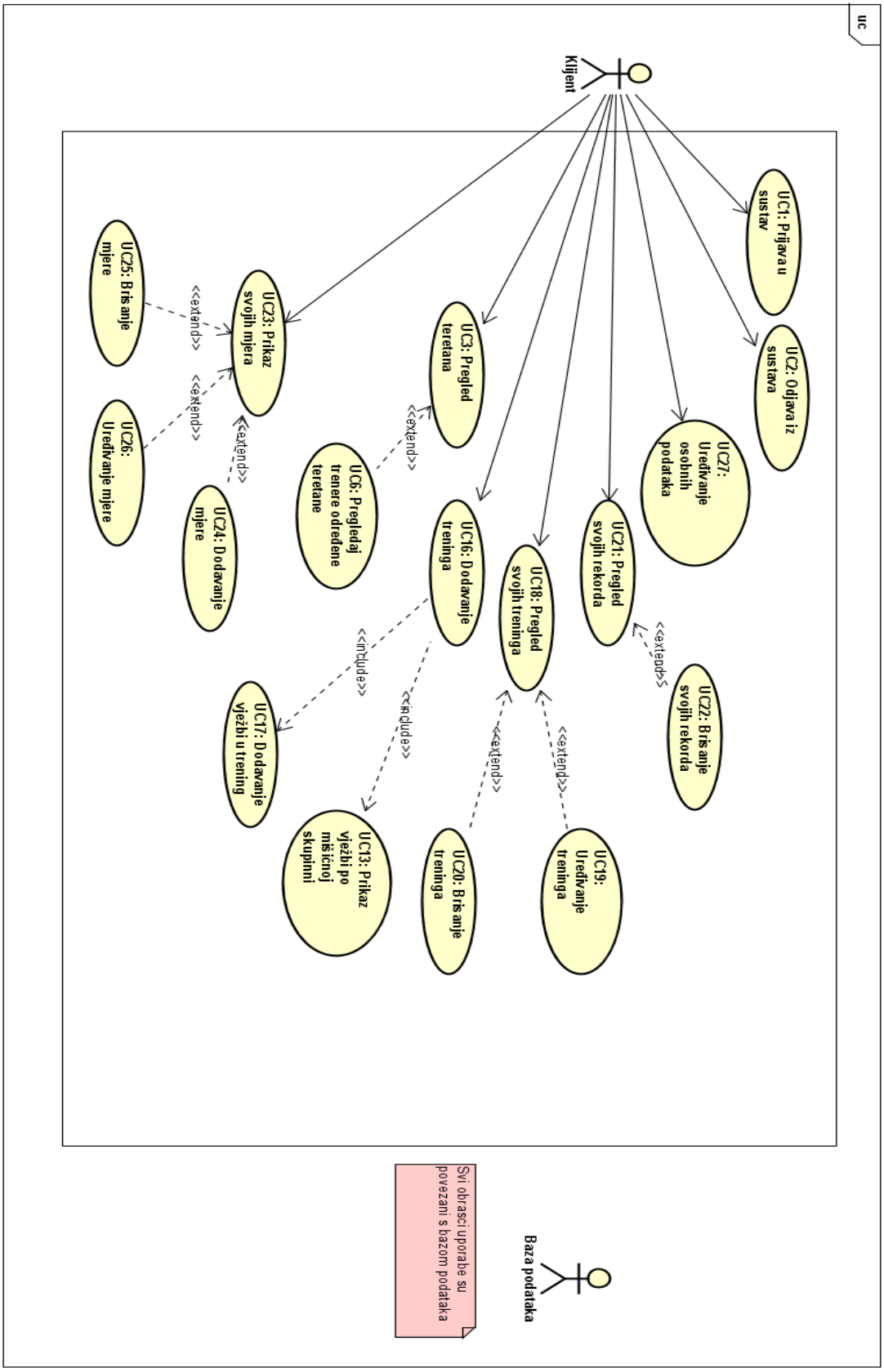
1.2. Obrasci uporabe

Za prikaz funkcionalnih zahtjeva pojedine uloge u web-aplikaciji često se koriste dijagrami obrazaca uporabe (engl. *Use case diagram*) koji pripadaju skupini statičkih UML

dijagrama (engl. *Unified Modeling Language*). Na Slika 1.1 i Slika 1.2 prikazani su dijagrami obrazaca uporabe. Radi čitljivijeg prikaza dijagram sam razdijelio u dva poddijagrama, svaki za pojedinu ulogu u web-aplikaciji.



Slika 1.1 – Dijagram obrazaca uporabe za administratora



Slika 1.2 – Dijagram obrazaca uporabe za klijenta

1.3. Opis obrazaca uporabe

UC1: Prijava u sustav

- **Glavni sudionik:** bilo koji registrirani korisnik (klijent, administrator)
- **Cilj:** dobiti pristup korisničkom sučelju web-aplikacije
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti registriran u sustav
- **Željeni scenarij:** korisnik unese korisničko ime i zaporku te mu se prikaže korisničko sučelje aplikacije ovisno o ulozi korisnika (klijent, administrator)
- **Mogući ostali scenariji:** greška u prijavi, u slučaju da korisnik nije registriran

UC2: Odjava iz sustava

- **Glavni sudionik:** bilo koji prijavljeni korisnik (klijent, administrator)
- **Cilj:** odjava iz korisničkog sučelja
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav
- **Željeni scenarij:** pritiskom na dugme za odjavu korisnik se odjavljuje iz korisničkog sučelja
- **Mogući ostali scenariji:** nema ostalih mogućih scenarija

UC3: Pregled teretana

- **Glavni sudionik:** bilo koji prijavljeni korisnik (klijent, administrator)
- **Cilj:** formatirani prikaz svih pohranjenih teretana
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav
- **Željeni scenarij:** pritiskom na određeni gumb korisniku se prikazuje lista teretana
- **Mogući ostali scenariji:** ako nema nijedne teretane korisniku se ne prikazuje ništa

UC4: Dodaj teretanu

- **Glavni sudionik:** administrator
- **Cilj:** dodavanje nove teretane u bazu podataka

- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu administratora
- **Željeni scenarij:** korisnik ispuni formu za dodavanje teretane nakon čega se pohranjuje nova teretana
- **Mogući ostali scenariji:** greška pri ispunjavanju forme ili pohrani u bazu podataka, sustav dojavljuje poruku o pogrešci

UC5: Obriši teretanu

- **Glavni sudionik:** administrator
- **Cilj:** brisanje teretane iz baze podataka
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu administratora
- **Željeni scenarij:** pritiskom na dugme obriši administrator briše određenu teretanu iz baze podataka
- **Mogući ostali scenariji:** ako je neuspješno brisanje iz baze podataka aplikacija ispisuje grešku

UC6: Pregledaj trenere određene teretane

- **Glavni sudionik:** bilo koji prijavljeni korisnik (klijent, administrator)
- **Cilj:** formatirani prikaz svih trenera pojedine teretane
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav
- **Željeni scenarij:** pritiskom na gumb ispod određene teretane korisnik dobiva formatirani prikaz liste svih trenera pojedine teretane
- **Mogući ostali scenariji:** ako nema nijednog trenera za tu teretanu korisniku se ispisuje poruka da nema dostupnih trenera za tu teretanu

UC7: Pregledaj sve trenere

- **Glavni sudionik:** administrator
- **Cilj:** formatirani prikaz svih trenera iz baze podataka
- **Ostali sudionici:** baza podataka

- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu administratora
- **Željeni scenarij:** nakon pritiska određenog gumba korisnik dobiva formatirani prikaz svih dostupnih trenera
- **Mogući ostali scenariji:** ako nema nijednog trenera korisniku se ispisuje poruka da nema dostupnih trenera

UC8: Uredi teretanu

- **Glavni sudionik:** administrator
- **Cilj:** izmjena podataka o teretani
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu administratora
- **Željeni scenarij:** nakon ispunjavanja forme za uređivanje izmjenjuju se podaci o teretani u bazi podataka
- **Mogući ostali scenariji:** greška pri ispunjavanju forme ili pohrani u bazu podataka, sustav dojavljuje poruku o pogrešci

UC9: Dodaj trenera

- **Glavni sudionik:** administrator
- **Cilj:** dodavanje novog trenera u bazu podataka
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu administratora
- **Željeni scenarij:** nakon ispunjavanja forme za dodavanje trenera dodaje se novi trener u bazu podataka
- **Mogući ostali scenariji:** greška pri ispunjavanju forme ili pohrani u bazu podataka, sustav dojavljuje poruku o pogrešci

UC10: Obriši trenera

- **Glavni sudionik:** administrator
- **Cilj:** brisanje trenera iz baze podataka
- **Ostali sudionici:** baza podataka

- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu administratora
- **Željeni scenarij:** pritiskom na dugme obriši administrator briše određenog trenera iz baze podataka
- **Mogući ostali scenariji:** ako je neuspješno brisanje iz baze podataka aplikacija ispisuje grešku

UC11: Uredi trenera

- **Glavni sudionik:** administrator
- **Cilj:** uređivanje podataka o treneru
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu administratora
- **Željeni scenarij:** nakon ispunjavanja forme za uređivanje izmjenjuju se podaci o treneru u bazi podataka
- **Mogući ostali scenariji:** greška pri ispunjavanju forme ili pohrani u bazu podataka, sustav dojavljuje poruku o pogrešci

UC12: Brisanje klijenta

- **Glavni sudionik:** administrator
- **Cilj:** brisanje klijenta iz baze podataka
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu administratora
- **Željeni scenarij:** pritiskom na dugme obriši administrator briše određenog klijenta iz baze podataka
- **Mogući ostali scenariji:** ako je neuspješno brisanje iz baze podataka aplikacija ispisuje grešku

UC13: Prikaz vježbi po mišićnoj skupini

- **Glavni sudionik:** bilo koji prijavljeni korisnik (klijent, administrator)
- **Cilj:** formatirani prikaz svih vježbi iz baze podataka grupiranih po mišićnoj skupini

- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav
- **Željeni scenarij:** nakon što je to zatražio korisniku se prikazuju sve dostupne vježbe grupirane po mišićnoj skupini
- **Mogući ostali scenariji:** ako ne postoji nijedna vježba za određenu mišićnu skupinu pod tom mišićnom skupinom se neće prikazivati ništa

UC14: Dodavanje vježbi

- **Glavni sudionik:** administrator
- **Cilj:** dodavanje nove vježbe u bazu podataka
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu administratora
- **Željeni scenarij:** nakon što mu se prikažu sve vježbe po mišićnoj skupini, administrator može dodati vježbu za određenu mišićnu skupinu. Nakon ispunjavanja forme za dodavanje vježbe dodaje se nova vježba u bazu podataka
- **Mogući ostali scenariji:** greška pri ispunjavanju forme ili pohrani u bazu podataka, sustav dojavljuje poruku o pogrešci

UC15: Brisanje vježbe

- **Glavni sudionik:** administrator
- **Cilj:** brisanje vježbe iz baze podataka
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu administratora
- **Željeni scenarij:** pritiskom na dugme obriši administrator briše određenu vježbu iz baze podataka
- **Mogući ostali scenariji:** ako je neuspješno brisanje iz baze podataka aplikacija ispisuje grešku

UC16: Dodavanje treninga

- **Glavni sudionik:** klijent
- **Cilj:** dodavanje novog treninga u bazu podataka

- **Ostali sudionici:** baza podatka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu klijenta
- **Željeni scenarij:** nakon ispunjavanja forme za dodavanje treninga dodaje se novi trening u bazu podataka te se klijenta preusmjerava na stranicu za dodavanje vježbi u taj trening
- **Mogući ostali scenariji:** greška pri ispunjavanju forme ili pohrani u bazu podataka, sustav dojavljuje poruku o pogrešci

UC17: Dodavanje vježbi u trening

- **Glavni sudionik:** klijent
- **Cilj:** dodavanje novih vježbi u postojeći trening
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu klijenta
- **Željeni scenarij:** klijent odabire koju će vježbu dodati u trening i ispunjava formu o dodatnim informacijama (broj setova, broj ponavljanja…). Nakon toga vježba se pohranjuje u kontekstu određenog treninga
- **Mogući ostali scenariji:** greška pri ispunjavanju forme ili pohrani u bazu podataka, sustav dojavljuje poruku o pogrešci

UC18: Pregled svojih treninga

- **Glavni sudionik:** klijent
- **Cilj:** formatirani prikaz svih treninga klijenta
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu klijenta
- **Željeni scenarij:** pritiskom na određeno dugme klijentu se prikazuje formatirana lista svih njegovih treninga
- **Mogući ostali scenariji:** ako nema nijednog treninga klijentu se ispisuje prazna stranica

UC19: Uređivanje treninga

- **Glavni sudionik:** klijent
- **Cilj:** uređivanje određenog treninga
- **Ostali sudionici:** baza podataka

- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu klijenta
- **Željeni scenarij:** nakon pritiska dugmeta za uređivanje treninga klijentu se prikazuje stranica za uređivanje treninga i forma za dodavanje vježbe u trening
- **Mogući ostali scenariji:** greška pri ispunjavanju forme ili pohrani u bazu podataka, sustav dojavljuje poruku o pogrešci

UC20: Brisanje treninga

- **Glavni sudionik:** klijent
- **Cilj:** brisanje određenog treninga iz baze podataka
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu klijenta
- **Željeni scenarij:** pritiskom na dugme obriši administrator briše određeni trening iz baze podataka
- **Mogući ostali scenariji:** ako je neuspješno brisanje iz baze podataka aplikacija ispisuje grešku

UC21: Pregled svojih rekorda

- **Glavni sudionik:** klijent
- **Cilj:** prikaz svih rekorda klijenta
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu klijenta
- **Željeni scenarij:** nakon što to zatraži, klijentu se ispisuje lista svih njegovih rekorda za svaku vježbu koju je radio u treninzima
- **Mogući ostali scenariji:** ako nema nijednog rekorda tj. klijent nije napravio nijedan trening ispisuje se prazna stranica

UC22: Brisanje svojih rekorda

- **Glavni sudionik:** klijent
- **Cilj:** brisanje svog rekorda iz baze podataka
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu klijenta
- **Željeni scenarij:** pritiskom na dugme obriši klijent briše određeni rekord iz baze podataka

- **Mogući ostali scenariji:** ako je neuspješno brisanje iz baze podataka aplikacija ispisuje grešku

UC23: Prikaz svojih mjera

- **Glavni sudionik:** klijent
- **Cilj:** prikaz svih klijentovih mjera iz baze podataka
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu klijenta
- **Željeni scenarij:** pritiskom na određeni gumb klijentu se prikazuju sve njegove prethodno unesene mjere
- **Mogući ostali scenariji:** ako klijent nema nijedne mjere ispisuje se prazna stranica

UC24: Dodavanje mjere

- **Glavni sudionik:** klijent
- **Cilj:** dodavanje nove mjere u bazu podataka
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu klijenta
- **Željeni scenarij:** nakon ispunjavanja forme za dodavanje nove mjere dodaje se nova mjera u bazu podataka
- **Mogući ostali scenariji:** greška pri ispunjavanju forme ili pohrani u bazu podataka, sustav dojavljuje poruku o pogrešci

UC25: Brisanje mjere

- **Glavni sudionik:** klijent
- **Cilj:** brisanje određene klijentove mjere iz baze podataka
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu klijenta
- **Željeni scenarij:** pritiskom na dugme obriši klijent briše određenu svoju mjeru iz baze podataka
- **Mogući ostali scenariji:** ako je neuspješno brisanje iz baze podataka aplikacija ispisuje grešku

UC26: Uređivanje mjere

- **Glavni sudionik:** klijent
- **Cilj:** uređivanje podataka o postojećoj mjeri klijenta
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu klijenta
- **Željeni scenarij:** nakon ispunjavanja forme za uređivanje izmjenjuju se podaci o određenoj mjeri klijenta u bazi podataka
- **Mogući ostali scenariji:** greška pri ispunjavanju forme ili pohrani u bazu podataka, sustav dojavljuje poruku o pogrešci

UC27: Uređivanje osobnih podataka

- **Glavni sudionik:** klijent
- **Cilj:** promjena vlastitih osobnih podataka u bazi podataka
- **Ostali sudionici:** baza podataka
- **Preduvjeti:** korisnik mora biti prijavljen u sustav i mora imati ulogu klijenta
- **Željeni scenarij:** nakon ispunjavanja forme za uređivanje izmjenjuju se osobni podaci klijenta u bazi podataka
- **Mogući ostali scenariji:** greška pri ispunjavanju forme ili pohrani u bazu podataka, sustav dojavljuje poruku o pogrešci

2. Baza podataka

Da bi izradili kvalitetnu bazu podataka, moramo gledati na nju kao zasebnu cjelinu. Takvim načinom izrade mislit ćemo na sva ograničenja koja baza podataka mora implementirati ne vezano za web-aplikaciju u kojoj se koristi. Ovo omogućava da nam je baza podataka portabilna i primjenjiva na više različitih web-aplikacija uz minimalne ili čak nikakve izmjene.

Baza podataka za ovu web-aplikaciju izrađena je pomoću PostgreSQL-a koristeći alat pgAdmin 4.

2.1. ER model

ER model (engl. *Entity-Relationship model*) jednostavan je grafički prikaz modela baze podataka. Pomoću ER modela dobivamo uvid u odnose entiteta i veza između njih u bazi podataka. Naša baza podataka se koristi za pohranu teretana, trenera, klijenata, njihovog napretka i trenažnog procesa.

Za svaku teretanu bilježi se ime, adresa, radno vrijeme, mjesečna članarina te opcionalni opis teretane(npr. kojim danima tjedno radi, neke dodatne informacije o radu i članarini). Također za teretanu se bilježi u kojem se gradu nalazi. Grad obilježava poštanski broj, ime grada te država u kojoj se nalazi.

Svaka teretana ima više trenera za kojeg bilježimo ime, prezime, cijenu jednog individualnog treninga, te je moguće još pohraniti link na njegovu sliku, Instagram profil i broj mobitela. Jedan trener može biti dostupan u jednoj ili više teretana istovremeno.

Klijent u istom periodu može ići samo u jednu teretanu, čime svjesno postavljamo blago ograničenje u aplikaciju. Klijenta definira ime, prezime, datum rođenja, mail adresa i uloga koja može biti klijent ili administrator. Svaki klijent može odraditi 0 ili više treninga.

Trening opisujemo datumom kad je izveden i trajanjem u minutama. Na treningu se izvode jedna ili više vježbi. Isto tako neka vježba se može pojavljivati u više različitih treninga.

Vježbu definira ime, opis vježbe te koja mišićna skupina je pogođena tom vježbom. Svaka vježba se može klasificirati tako da pogađa točno jednu mišićnu skupinu.

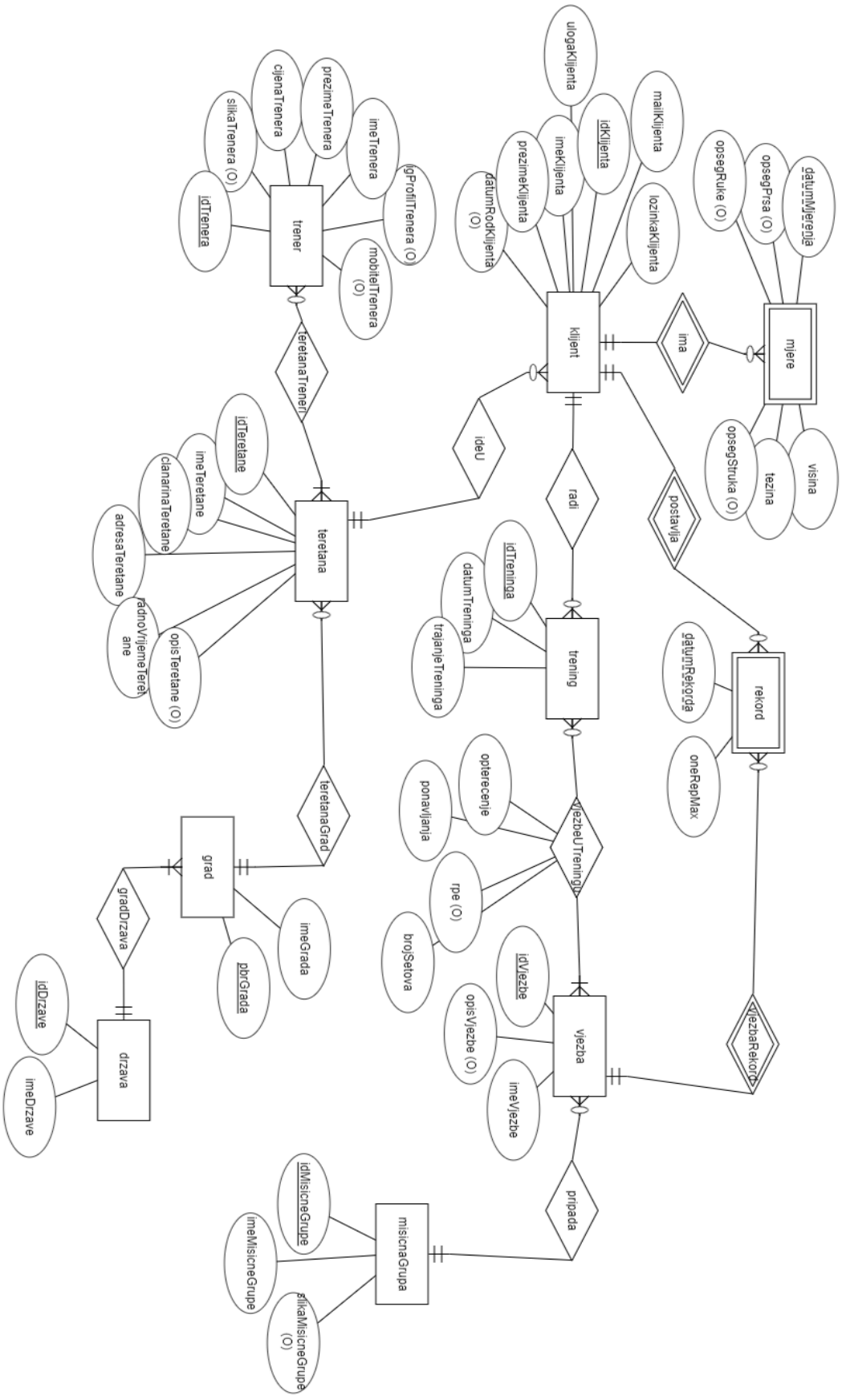
Mišićne skupine (leđa, noge, biceps, ramena..) definiramo njezinim imenom i linkom na sliku te mišićne skupine.

Prilikom treniranja određene vježbe u bazu se zapisuje opterećenje u kilogramima, broj odrađenih setova, broj ponavljanja i opcionalno RPE (engl. *rate of perceived exertion* - osobni dojam klijenta koliko su mu teški bili setovi određene vježbe s određenim opterećenjem, skalira se od 1 do 10).

Klijentu se za svaku odrađenu vježbu pohranjuje osobni rekord. Svaki rekord pripada točno jednom klijentu i točno jednoj vježbi te nema smisla da postoji bez klijenta i vježbe. Definira ga datum kad je izvedena vježba kao primarni ključ te funkcijom izračunat one rep max. Nakon svakog izvođenja vježbe provjerava se je li postignut novi one rep max klijenta za tu vježbu. Ako je, tada je osvježava tablica osobnih rekorda.

Klijent u aplikaciji može pohranjivati i svoje mjere koje će također biti egzistencijalno slab entitet. Njih definira datum mjerenja kao primarni ključ, visina, težina, te opcionalno opseg struka, prsa i nadlaktice.

Slika 2.1 prikazuje ER model opisane baze podataka.



Slika 2.1 – ER dijagram baze podataka

2.1.1. Popis entiteta

Popis svih entiteta iz ER modela baze podataka s pripadajućim atributima:

- KLIJENT - idKlijenta, imeKlijenta, prezimeKlijenta, mailKlijenta, lozinkaKlijenta, ulogaKlijenta, datumRodKlijenta
- TERETANA - idTeretane, imeTeretane, clanarinaTeretane, adresaTeretane, radnoVrijemeTeretane, opisTeretane
- TRENER - idTrenera, imeTrenera, prezimeTrenera, cijenaTrenera, slikaTrenera, igProfilTrenera, mobitelTrenera
- GRAD - pbrGrada, imeGrada
- DRŽAVA - idDrzave, imeDrzave
- TRENING - idTreninga, datumTreninga, trajanjeTreninga
- VJEŽBA - idVjezbe, opisVjezbe, imeVjezbe
- MISIĆNA GRUPA - idMisicneGrupe, imeMisicneGrupe, slikaMisicneGrupe
- REKORD - datumRekorda, idVjezbe, idKlijenta, oneRepMax
- MJERE - datumMjerenja, idKlijenta, visina, tezina, opsegStruka, opsegRuke, opsegPrsa

2.1.2. Popis veza

Popis svih veza iz ER modela baze podataka s pripadajućim atributima:

- IMA - datumMjerenja, idKlijenta
- POSTAVLJA - datumRekorda, idKlijenta, idVjezbe
- VJEŽBA REKORD - datumRekorda, idKlijenta, idVjezbe
- PRIPADA - idVjezbe, idMisicneGrupe

- VJEŽBE U TRENINGU - idTreninga, idVjezbe, opterecenje, ponavljanja, brojSetova, rpe
- RADI - idTreninga, idKlijenta
- IDE U - idKlijenta, idTeretane
- TERETANA TRENERI - idTrenera, idTeretane
- TERETANA GRAD - idTeretane, pbrGrada
- GRAD DRŽAVA - pbrGrada, idDrzave

2.2. Relacijski model

Relacijski model baze podataka dobiva se direktno iz ER modela i ključan je za dobivanje relacijske baze podataka. Relacijski model prikazuje relacije te iz njega direktno implementiramo našu bazu podataka. Iz poviše spomenutog ER modela dobivamo sljedeće relacije i relacijski model (Slika 2.2):

- KLIJENT – idKlijenta, imeKlijenta, prezimeKlijenta, mailKlijenta, lozinkaKlijenta, datumRodKlijenta, ulogaKlijenta, idTeretane

Relacija klijent bilježi podatke o korisniku. Primarni ključ je atribut 'idKlijenta'.Atribut 'ulogaKlijenta' razvrstava korisnike na klijente i administratore a atribut 'datumRodKlijenta' je opcionalan i može poprimiti NULL vrijednost. Atribut 'idTeretane' je strani ključ koji se referencira na relaciju 'Teretana'.

- REKORD – datumRekorda, idKlijenta, idVjezbe, oneRepMax

Relacija rekord bilježi sve novonastale rekorde pojedinog klijenta u pojedinoj vježbi. Primarni ključ su atributi ' datumRekorda', ' idKlijenta' i ' idVjezbe', a 'oneRepMax' se izračunava na temelju posebne funkcije.

- VJEŽBA – idVjezbe, opisVjezbe, imeVjezbe, idMisicneGrupe

Relacija vježba opisuje svaku vježbu. Primarni ključ joj je 'idVjezbe', atribut 'opisVjezbe' je opcionalan a 'idMisicneGrupe' je strani ključ koji se referencira na relaciju 'Mišićna grupa'.

- MIŠIĆNA GRUPA – idMisicneGrupe, imeMisicneGrupe, slikaMisicneGrupe

Svaka mišićna grupa ima jedinstven ' idMisicneGrupe', a u atribut ' slikaMisicneGrupe' se pohranjuje poveznica na sliku mišićne grupe koja se pohranjuje lokalno.

- TRENING – idTreninga, datumTreninga, trajanjeTreninga, idKlijenta

Relacija trening opisuje trening koji klijent izvodi, primarni ključ joj je 'idTreninga', trajanje treninga izraženo je u minutama a atribut 'idKlijenta' je strani ključ koji se referencija na relaciju 'Klijent'.

- VJEŽBE U TRENINGU – idTreninga, idVjezbe, opterecenje, ponavljanja, brojSetova, rpe

Relacija 'Vježbe u treningu' govori nam koju vježbu je klijent radio u kojem treningu, te koliko puta i pod kojim opterećenjem (broj setova, broj ponavljanja, opterećenje, rpe). Primarni ključ sastoji se od atributa 'idTreninga' (koji se referencira na relaciju 'Teretana') i atributa 'idVjezbe' (koji se referencira na relaciju 'Vježba'). Atribut 'rpe' je opcionalan.

- MJERE – datumMjerenja, idKljenta, visina, tezina, opsegStruka, opsegRuke, opsegPrsa

Relacija mjere pohranjuje mjerenja klijenta. Primarni ključ su atributi 'datumMjerenja' i 'idKlijenta' (koji se referencira na relaciju 'Klijent'). 'opsegStruka', 'opsegRuke' i 'opsegPrsa' su opcionalni atributi.

- TRENER – idTrenera, imeTrenera, prezimeTrenera, cijenaTrenera, mobitelTrenera, igProfilTrenera, slikaTrenera

Relacija trener daje nam podatke o treneru. Svaki trener jednoznačno je označen putem atributa 'idTrenera', a atributi 'mobitelTrenera', 'igProfilTrenera' i 'slikaTrenera' su opcionalni.

- TERETANA TRENERI – idTrenera, idTeretane

Relacija 'Teretana treneri' govori nam koji treneri rade u kojoj teretani. Primarni ključevi se referenciraju na relacije 'Trener' i 'Teretana'.

- TERETANA – idTeretane, imeTeretane, adresaTeretane, radnoVrijemeTeretane, clanarinaTeretane, opisTeretane, pbrGrada

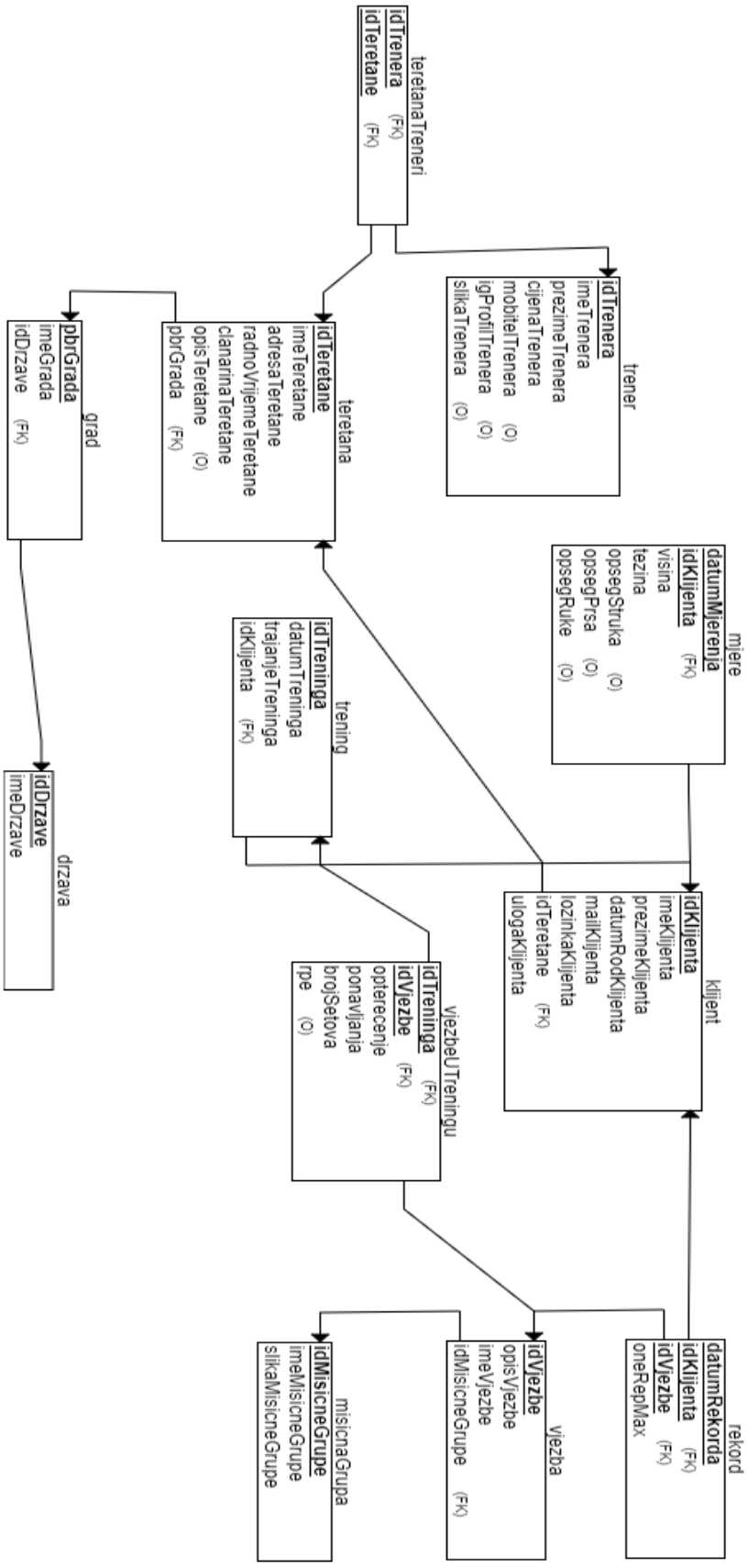
Relacija teretana pohranjuje podatke o svakoj teretani. Primarni ključ je 'idTeretane'. Atribut 'opisTeretane' ja opcionalan a atribut 'pbrGrada' je strani ključ koji se referencira na relaciju 'Grad'.

- GRAD – pbrGrada, imeGrada, idDrzave

Relacija grad pohranjuje gradove u kojima se teretane nalaze. Primarni ključ je poštanski broj grada, a atribut 'idDrzave' referencira se na relaciju 'Država'.

- DRŽAVA – idDrzave, imeDrzave

Relacija država pohranjuje države. Svaku državu jedinstveno određuje 'idDrzave'.



Slika 2.2 – relacijski model baze podataka

2.3. Programski kôd za kreiranje baze podataka

U nastavku prilažem SQL kôd za kreiranje tablica baze podataka te neka dodatna ograničenja za pojedine atribute (Slika 2.3, Slika 2.4, Slika 2.5, sSlika 2.6, Slika 2.7, Slika 2.8).

```
CREATE TABLE trener
(
  igProfilTrenera VARCHAR(50),
  mobitelTrenera VARCHAR(20),
  slikaTrenera VARCHAR(255),
  cijenaTrenera INT NOT NULL,
  imeTrenera VARCHAR(50) NOT NULL,
  prezimeTrenera VARCHAR(50) NOT NULL,
  idTrenera INT NOT NULL,
  PRIMARY KEY (idTrenera)
);

CREATE TABLE misicnaGrupa
(
  imeMisicneGrupe VARCHAR(100) NOT NULL,
  idMisicneGrupe INT NOT NULL,
  slikaMisicneGrupe VARCHAR(255),
  PRIMARY KEY (idMisicneGrupe)
);

CREATE TABLE drzava
(
  idDrzave INT NOT NULL,
  imeDrzave VARCHAR(50) NOT NULL,
  PRIMARY KEY (idDrzave)
);

CREATE TABLE vjezba
(
  opisVjezbe VARCHAR(500),
  idVjezbe INT NOT NULL,
  imeVjezbe VARCHAR(50) NOT NULL,
  idMisicneGrupe INT,
  PRIMARY KEY (idVjezbe),
  FOREIGN KEY (idMisicneGrupe) REFERENCES misicnaGrupa(idMisicneGrupe)
      ON DELETE SET NULL
      ON UPDATE CASCADE
);
```

Slika 2.3 – naredbe za kreiranje tablica baze podataka

```
CREATE TABLE grad
(
  pbrGrada INT NOT NULL,
  imeGrada VARCHAR(50) NOT NULL,
  idDrzave INT,
  PRIMARY KEY (pbrGrada),
  FOREIGN KEY (idDrzave) REFERENCES drzava(idDrzave)
      ON DELETE SET NULL
      ON UPDATE CASCADE,
  CHECK(pbrGrada BETWEEN 10000 AND 99999)
);

CREATE TABLE teretana
(
  adresaTeretane VARCHAR(100) NOT NULL,
  imeTeretane VARCHAR(100) NOT NULL,
  clanarinaTeretane INT NOT NULL,
  radnoVrijemeTeretane VARCHAR(50) NOT NULL,
  opisTeretane VARCHAR(500),
  idTeretane INT NOT NULL,
  pbrGrada INT,
  PRIMARY KEY (idTeretane),
  FOREIGN KEY (pbrGrada) REFERENCES grad(pbrGrada)
      ON DELETE SET NULL
      ON UPDATE CASCADE
);

CREATE TABLE teretanaTreneri
(
  idTrenera INT NOT NULL,
  idTeretane INT NOT NULL,
  PRIMARY KEY (idTrenera, idTeretane),
  FOREIGN KEY (idTrenera) REFERENCES trener(idTrenera)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
  FOREIGN KEY (idTeretane) REFERENCES teretana(idTeretane)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);
```

Slika 2.4 - naredbe za kreiranje tablica baze podataka

```
CREATE TABLE klijent
(
  datumRodKlijenta DATE,
  imeKlijenta VARCHAR(50) NOT NULL,
  prezimeKlijenta VARCHAR(50) NOT NULL,
  mailKlijenta VARCHAR(50) NOT NULL,
  idKlijenta INT NOT NULL,
  lozinkaKlijenta VARCHAR(255) NOT NULL,
  ulogaKlijenta VARCHAR(255) NOT NULL,
  idTeretane INT,
  PRIMARY KEY (idKlijenta),
  UNIQUE (mailKlijenta),
  FOREIGN KEY (idTeretane) REFERENCES teretana(idTeretane)
      ON DELETE SET NULL
      ON UPDATE CASCADE
);

CREATE TABLE trening
(
  datumTreninga DATE NOT NULL,
  trajanjeTreninga INT NOT NULL,
  idTreninga INT NOT NULL,
  idKlijenta INT NOT NULL,
  PRIMARY KEY (idTreninga),
  FOREIGN KEY (idKlijenta) REFERENCES klijent(idKlijenta)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);
```

Slika 2.5 - naredbe za kreiranje tablica baze podataka

```
CREATE TABLE rekord
(
  oneRepMax INT NOT NULL,
  datumRekorda DATE NOT NULL,
  idKlijenta INT NOT NULL,
  idVjezbe INT NOT NULL,
  PRIMARY KEY (datumRekorda, idKlijenta, idVjezbe),
  FOREIGN KEY (idKlijenta) REFERENCES klijent(idKlijenta)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
  FOREIGN KEY (idVjezbe) REFERENCES vjezba(idVjezbe)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE mjere
(
  visina INT NOT NULL,
  tezina FLOAT NOT NULL,
  opsegStruka FLOAT,
  opsegPrsa FLOAT,
  opsegRuke FLOAT,
  datumMjerenja DATE NOT NULL,
  idKlijenta INT NOT NULL,
  PRIMARY KEY (datumMjerenja, idKlijenta),
  FOREIGN KEY (idKlijenta) REFERENCES klijent(idKlijenta)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
  CHECK (visina BETWEEN 50 AND 300),
  CHECK (tezina BETWEEN 20 AND 300)
);
```

Slika 2.6 - naredbe za kreiranje tablica baze podataka

```
CREATE TABLE vjezbeUTreningu
(
  opterecenje INT NOT NULL,
  ponavljanja INT NOT NULL,
  rpe INT,
  brojSetova INT NOT NULL,
  idTreninga INT NOT NULL,
  idVjezbe INT NOT NULL,
  PRIMARY KEY (idTreninga, idVjezbe),
  FOREIGN KEY (idTreninga) REFERENCES trening(idTreninga)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
  FOREIGN KEY (idVjezbe) REFERENCES vjezba(idVjezbe)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
  CHECK (rpe BETWEEN 1 AND 10)
);
```

Slika 2.7 - naredbe za kreiranje tablica baze podataka

```
ALTER TABLE klijent
ADD CONSTRAINT chkDatumRod CHECK (datumrodklijenta < CURRENT_DATE);

ALTER TABLE klijent
ADD CONSTRAINT chkEmail CHECK (mailklijenta LIKE '%@%');

ALTER TABLE mjere
ADD CONSTRAINT chkDatumMjerenja CHECK (datummjerenja <= CURRENT_DATE);

ALTER TABLE rekord
ADD CONSTRAINT chkDatumRekord CHECK (datumrekorda <= CURRENT_DATE);

ALTER TABLE trening
ADD CONSTRAINT chkDatumTreninga CHECK (datumtreninga <= CURRENT_DATE);

ALTER TABLE teretana
ADD CONSTRAINT uniqueIme UNIQUE(imeteretane);
```

Slika 2.8 – neka naknadno dodana ograničenja

3. Web-aplikacija

3.1. Korišteni alati i tehnologije

Aplikacija je izrađena korištenjem radnog okvira Spring Boot. Spring Boot je najpoznatiji radni okvir za razvoj web-aplikacija pisanih u programskom jeziku Java te je u njemu napravljen kompletni backend ove web-aplikacije. Za izradu frontenda (dijela koji se šalje i prikazuje klijentu) koristio sam označni jezik HTML, JavaScript, CSS i njegov radni okvir Bootstrap. Za posluživanje informacija (modela) klijentu koristio sam Thymeleaf. Aplikacija je izrađena u IntelliJ IDEA okolini. U nastavku donosim upute za izradu Spring Boot projekta, instalaciju IntelliJa te kratak opis strukture ovog projekta.

3.1.1. Instalacija IntelliJ IDEA

IntelliJ IDEA je jedan od najboljih okolina za izradu Java aplikacija. Izrađen je od strane JetBrainsa koji pružaju još niz okolina za rad u raznim programskim jezicima (SQL, C, C++, Ruby...). Konkretno za instalaciju okoline za programski jezik Java trebamo posjetiti link https://www.jetbrains.com/idea/ te klikom na gumb download odabrati verziju koja odgovara našem računalu. Nakon instalacije moguće je izvoditi sve Java projekte u ovoj okolini.

3.1.2. Izrada Spring Boot projekta

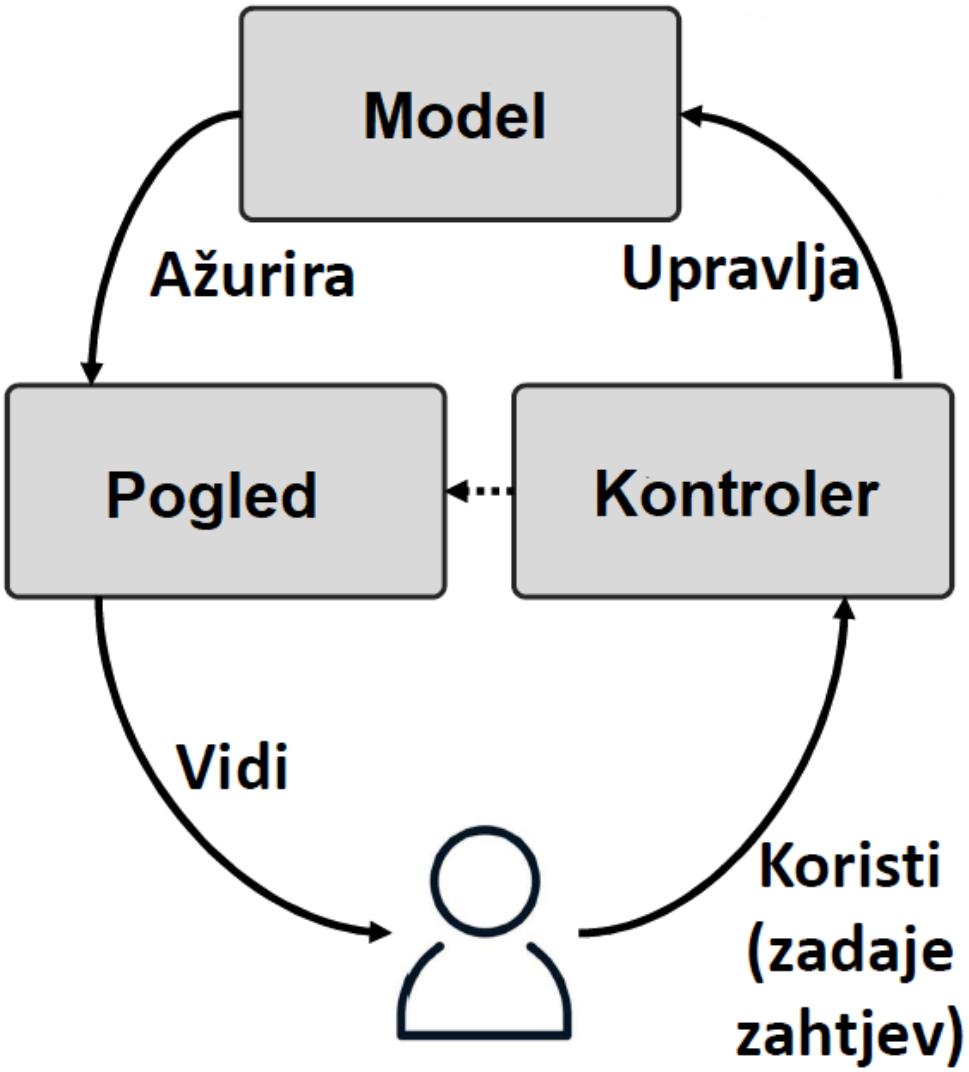
Spring Boot projekt najlakše je izraditi putem Spring Initializera (https://start.spring.io/) kroz koji možemo pridružiti ovisnosti (engl. *dependency*) svom projektu. Za osnovnu Spring Boot aplikaciju koja se povezuje na PostgreSQL bazu podataka potrebno je odabrati Spring web, Spring Dana JPA i PostgreSQL Driver dependency. Ako koristimo Thymeleaf na frontendu izabrat ćemo i dependency Thymeleaf. Nakon dodanih ovisnosti i inicijalizacije pohranjujemo zip datoteku koju raspakiramo i spremni smo za rad na novom Spring Boot projektu.

3.1.3. Arhitektura sustava

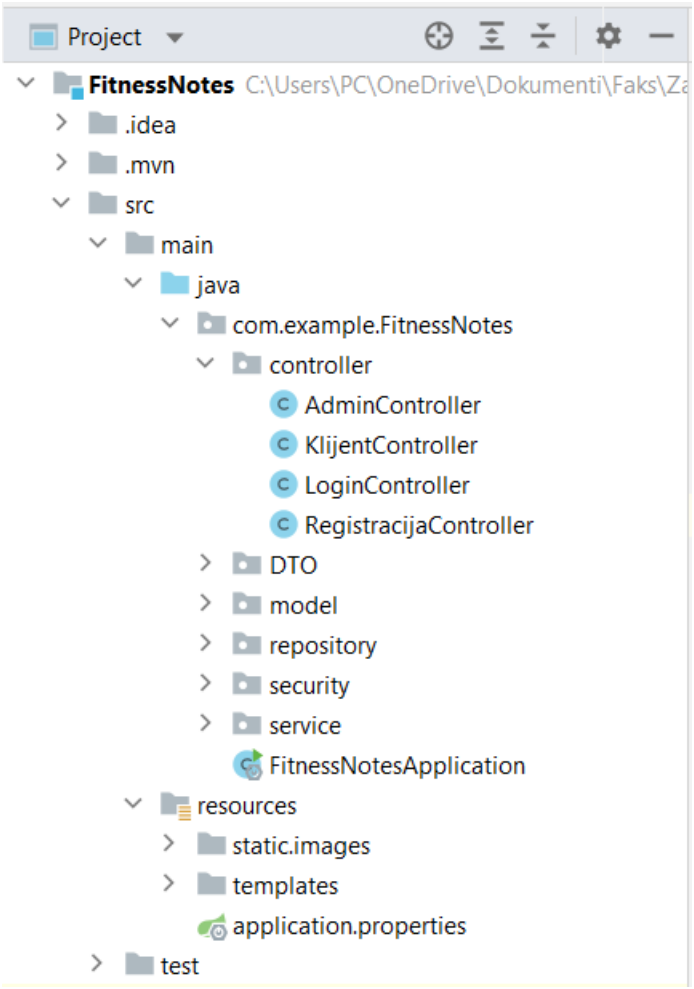
MVC (engl. *Model View Controller*) arhitektura je tipična arhitektura za stvaranje web-aplikacija. Sastoji se od tri komponente:

- **Model –** predstavlja podatke s kojim radimo i metode koje koristimo kao poslovnu logiku. U mom slučaju Spring Boota, u model spadaju sve datoteke iz direktorija 'model' (preslika tablica iz baze podataka), 'service' (poslovna logika aplikacije), 'DTO' (engl. *Dana Transfer Object*, format podataka koji šaljemo na pogled) i 'repository' (direktna poveznica između modela i baze podataka). Podacima iz modela manipulira kontroler te ih spaja sa pogledom koji se šalje korisniku.
- **Kontroler** (engl. *Controller*) **–** predstavlja funkcije koje se aktiviraju kad korisnik zatraži url za koji je ta funkcija predviđena. Nakon što korisnik zatraži pristup određenoj stranici, kontroler prima taj zahtjev i poziva sve potrebne funkcije iz modela da bi taj zahtjev obradio kako je predviđeno. Na kraju kontroler korisniku vraća pogled ažuriran podacima iz modela. U mojoj aplikaciji u kontroler spadaju sve datoteke iz direktorija 'controller'.
- **Pogled** (engl. *View*) **–** predstavlja podatke koje aplikacija šalje klijentu i koje klijent može vidjeti. Kroz kontroler pogled dobiva sve potrebne podatke za ažurirani prikaz, a to su zapravo podaci iz modela. U mojoj aplikaciji u pogled spadaju sve datoteke iz direktorija 'templates', a to su zapravo HTML datoteke koje prikazuju podatke iz modela koristeći Thymeleaf.

Slika 3.1 [2] prikazuje ilustraciju MVC arhitekture, a Slika 3.2 prikazuje radne direktorije ovog projekta. Direktorij 'security' služi za podešavanje Spring Securityja koji omogućuje login, autentifikaciju i autorizaciju korisnika. U direktoriju 'static.images' nalaze se slike koje su bile potrebne za ovaj projekt. Slika 3.3 prikazuje primjer jednog kontrolera iz aplikacije. Svaki kontroler počinje oznakom '@GetMapping' nakon čega slijedi url na koji će taj kontroler odgovarati.



Slika 3.1 – ilustracija MVC arhitekture



Slika 3.2 – radni direktoriji ovog projekta

```
@GetMapping("/dodajVjezbe/{id}")
public ModelAndView misicneGrupe(Authentication auth,
                                 @PathVariable(value = "id") Integer id) {
    if (auth == null) {
        return new ModelAndView( viewName: "redirect:/login/router");
    }
    List<VjezbaUTreninguDTO> vjezbe = treningService.dohvatiAktivnostUTreningu(id);
    Trening trening = treningService.dohvatiTreningPrekoId(id);
    List<MisicnaGrupaIVjezbeDTO> lista = vjezbaService.prikaziVjezbePoMisicnojGrupi();
    ModelAndView mav = new ModelAndView( viewName: "dodaj_vjezbe_u_trening");
    mav.addObject( attributeName: "vjezbe", vjezbe);
    mav.addObject( attributeName: "trening", trening);
    mav.addObject( attributeName: "lista", lista);
    return mav;
}
```

Slika 3.3 – primjer jednog kontrolera

3.1.4. Implementacija sustava

Implementaciju sam započeo izradom ER i relacijskog modela baze podataka. Nakon što sam bio siguran da je baza podataka legitimno i ispravno izrađena, kreirao sam Spring Boot projekt i krenuo s izradom serverske strane (engl. *backend*) ove web-aplikacije. Prvo sam morao izraditi model koji će preslikavati bazu podataka. Slika 3.4 prikazuje primjer modela u aplikaciji za tablicu trener iz baze podataka. Izostavio sam Gettere i Settere za svaki atribut koji se također nalaze u klasi.

```java
package com.example.FitnessNotes.model;

import javax.persistence.*;

@Entity
@Table(name = "trener")
public class Trener {
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Id
    @Column(name = "idtrenera", nullable = false)
    private Integer id;

    @Column(name = "igprofiltrenera", length = 50)
    private String igprofiltrenera;

    @Column(name = "mobiteltrenera", length = 20)
    private String mobiteltrenera;

    @Column(name = "slikatrenera")
    private String slikatrenera;

    @Column(name = "cijenatrenera", nullable = false)
    private Integer cijenatrenera;

    @Column(name = "imetrenera", nullable = false, length = 50)
    private String imetrenera;

    @Column(name = "prezimetrenera", nullable = false, length = 50)
    private String prezimetrenera;
```

Slika 3.4 – model za tablicu trener

Nakon formiranog modela, prebacio sam se na izradu kontrolera koji će prihvaćati zahtjeve klijenta kao što je opisano u poglavlju Arhitektura sustava. Dok sam izrađivao kontrolere paralelno sam radio i poslovnu logiku u direktoriju service. Gotovo svaka tablica iz baze podataka je imala svoju service datoteku. Slika 3.5 i Slika 3.6 prikazuju par metoda service-a za tablicu klijent. Dok sam radio kontrolere paralelno sam radio i klijentsku stranu (engl. *frontend*). Tako mi je bilo lakše zato što bih u tom trenutku taman završio kontroler pa bi mi bilo 'svježe' koje podatke prenosim na klijentsku stranu, što je možda malo ubrzalo izradu aplikacije.

```
public void azurirajKlijenta(Klijent stari, OsobniPodaciDTO osobniPodaci) {
    if (osobniPodaci != null && klijentRepository.existsById(stari.getId())) {
        Klijent azuriran = new Klijent();
        azuriran.setId(stari.getId());
        azuriran.setImeklijenta(osobniPodaci.getIme());
        azuriran.setPrezimeklijenta(osobniPodaci.getPrezime());
        azuriran.setDatumrodklijenta(osobniPodaci.getDatumRod());
        azuriran.setMailklijenta(stari.getMailklijenta());
        azuriran.setLozinkaklijenta(stari.getLozinkaklijenta());
        azuriran.setUloga(stari.getUloga());
        if (teretanaRepository.existsById(osobniPodaci.getTeretanaId())) {
            Teretana teretana = teretanaRepository.getById(osobniPodaci.getTeretanaId());
            azuriran.setIdteretane(teretana);
            klijentRepository.save(azuriran);
            return;
        } else {
            System.out.println("Teretana ne postoji.");
            throw new IllegalArgumentException("Teretana ne postoji.");
        }
    }
    System.out.println("Podaci za promjenu su neispravni ili korisnik ne postoji.");
    throw new IllegalArgumentException("Podaci za promjenu su neispravni ili korisnik ne postoji.");
}
```

Slika 3.5 – metoda 'ažuriraj klijenta' iz klijentovog service-a

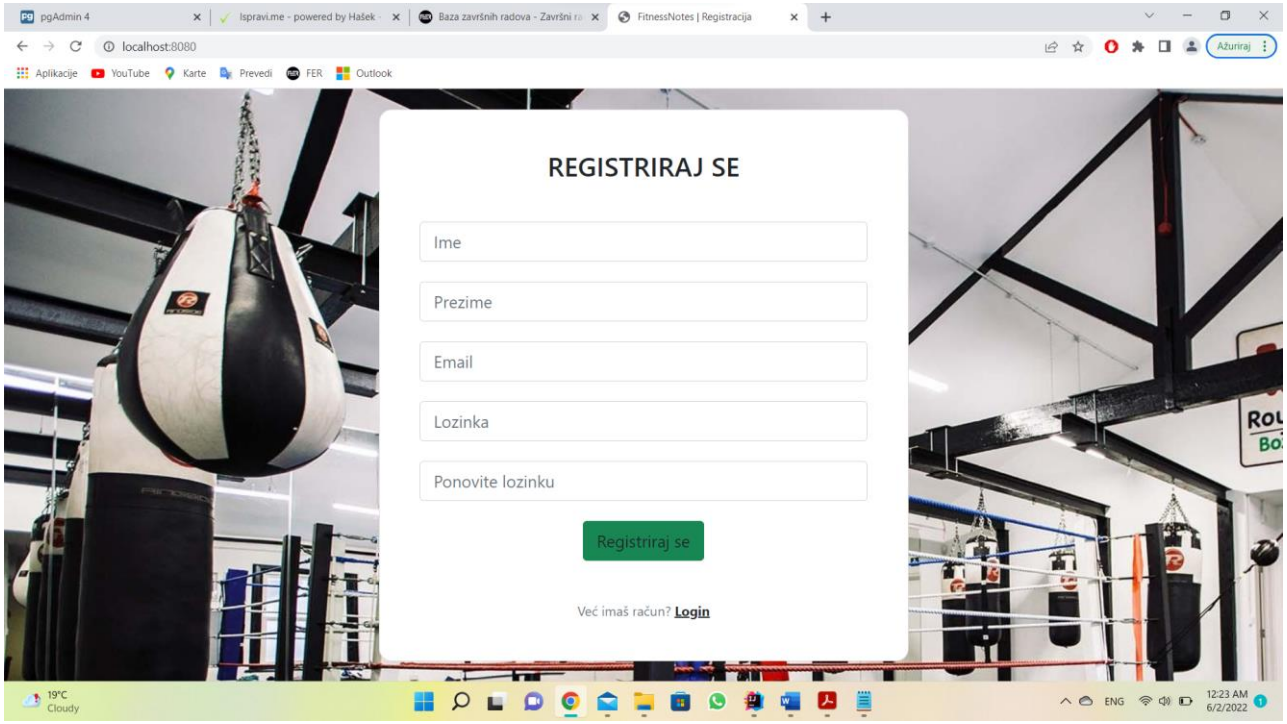
```java
public List<RekordDTO> dohvatiRekordeKlijenta(Klijent klijent) {
    if (klijentRepository.existsById(klijent.getId())) {
        List<Rekord> rekordi = rekordRepository.findAllByIdKlijenta(klijent.getId());
        List<RekordDTO> rekordiDto = new ArrayList<>();
        int i = 1;
        for (var rekord: rekordi) {
            Optional<Vjezba> vjezba = vjezbaRepository.findById(rekord.getId().getIdvjezbe());
            if(vjezba.isPresent()) {
                rekordiDto.add(new RekordDTO(i, rekord, vjezba.get().getImevjezbe()));
            } else {
                rekordiDto.add(new RekordDTO(i, rekord,  imeVjezbe: "Nepoznata vježba"));
            }
            i++;
        }
        Collections.sort(rekordiDto, new Comparator<RekordDTO>() {
            @Override
            public int compare(RekordDTO o1, RekordDTO o2) {
                return o2.getRekord().getId().getDatumrekorda().compareTo(o1.getRekord().getId().getDatumrekorda());
            }
        });
        return rekordiDto;
    } else {
        System.out.println("Klijent ne postoji");
        throw new IllegalArgumentException("Klijent ne postoji");
    }
}
```

Slika 3.6 – metoda 'dohvati rekorde klijenta' iz klijentovog service-a

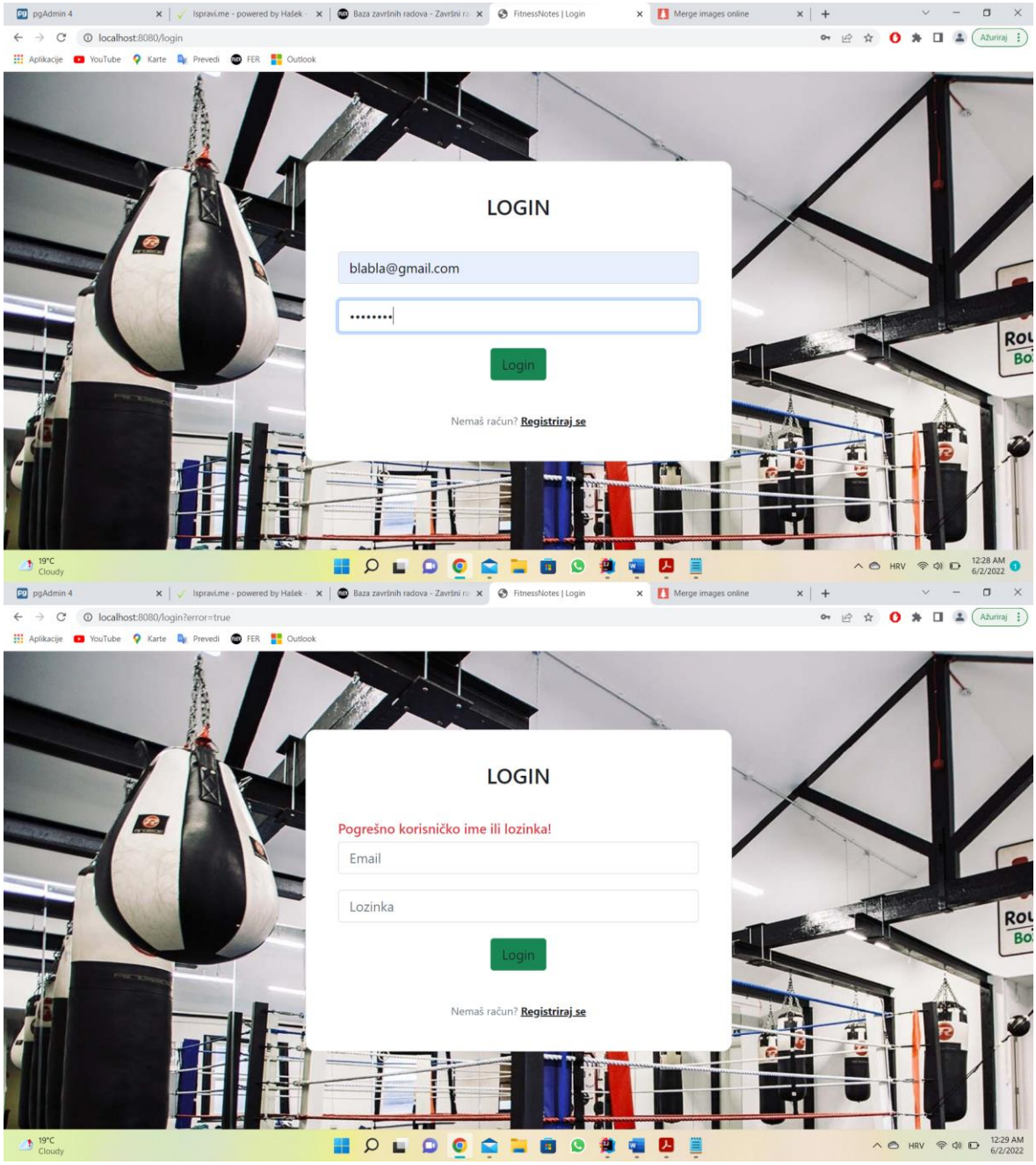
3.2. Upute za aplikaciju

3.2.1. Registracija i prijava

Aplikacija započinje formom za registraciju i prijavu. Formu za registraciju vidimo na Slika 3.7. Ako korisnik nije registriran mora najprije obaviti registraciju inače neće imati pristup korisničkom sučelju web-aplikacije (Slika 3.8). Kad je registracija obavljena korisnik dobiva pristup korisničkom sučelju aplikacije namijenjenom klijentu. Administrator već postoji i dodan je direktno u bazu podataka te se novi korisnik ne može registrirati putem aplikacije pod ulogom administratora.



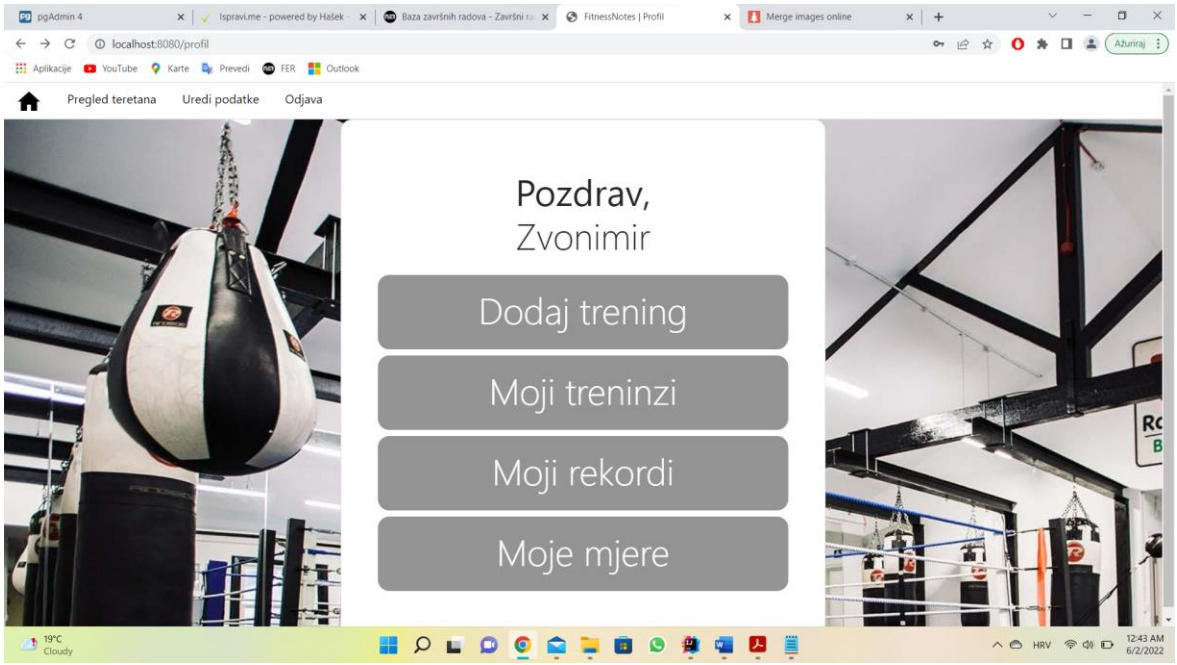
Slika 3.7 – forma za registraciju



Slika 3.8 - pokušaj prijave od strane neregistriranog korisnika

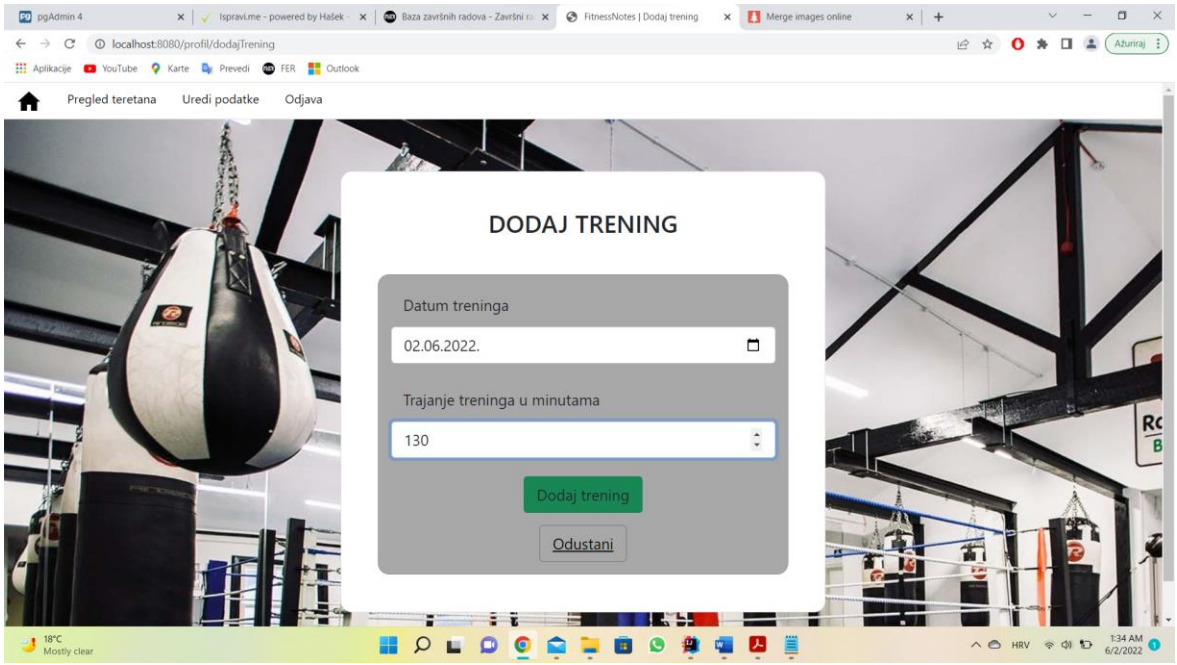
3.2.2. Vodič za klijenta

Nakon prijave klijentu se prikazuje korisničko sučelje aplikacije u kojem ima više mogućnosti (Slika 3.9). U nastavku prilažem snimke zaslona složenijih i interesantnijih funkcionalnosti za klijenta jer su sve ostale funkcionalnosti jako intuitivne.

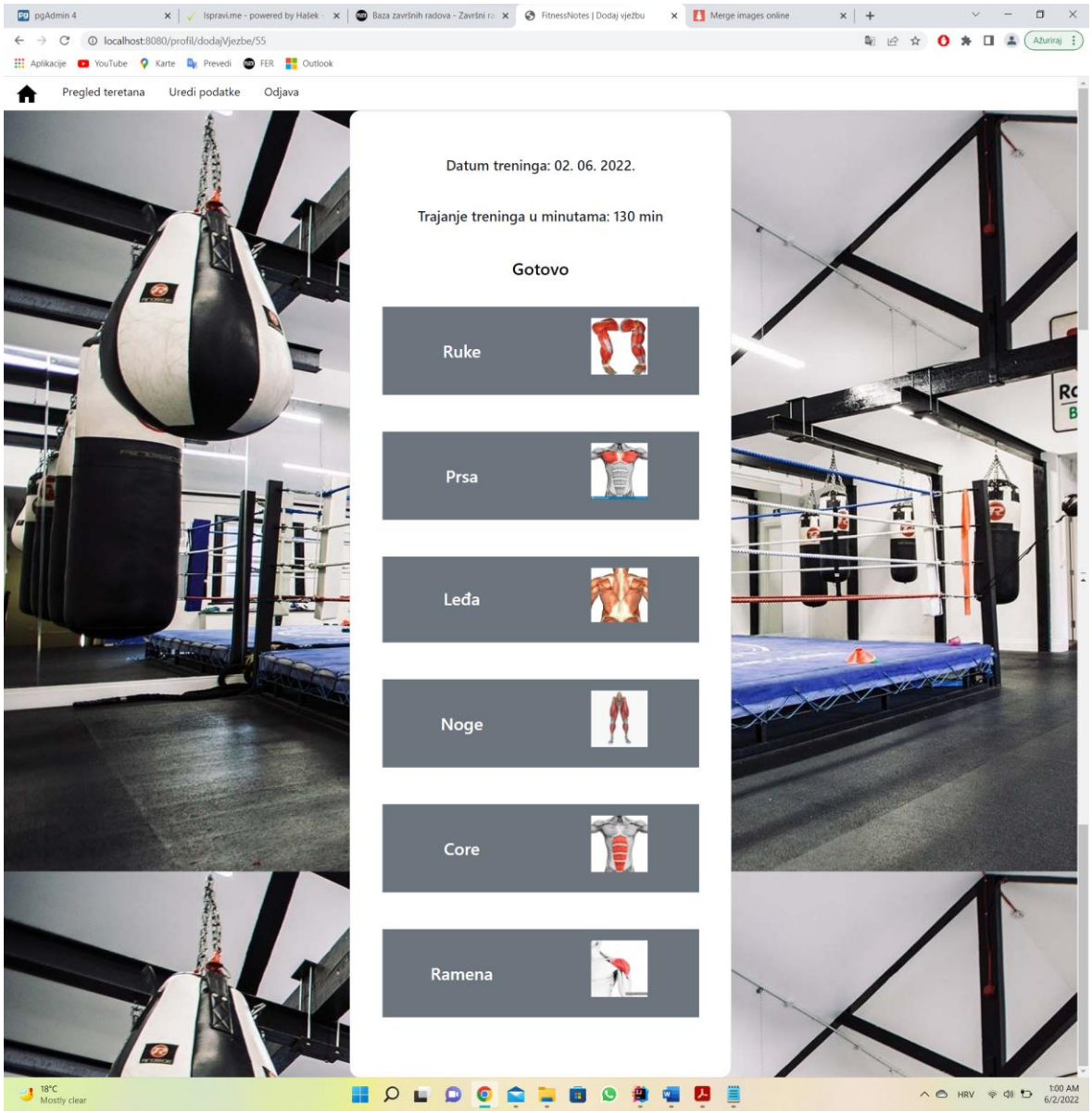


Slika 3.9 - početna stranica nakon prijave klijenta

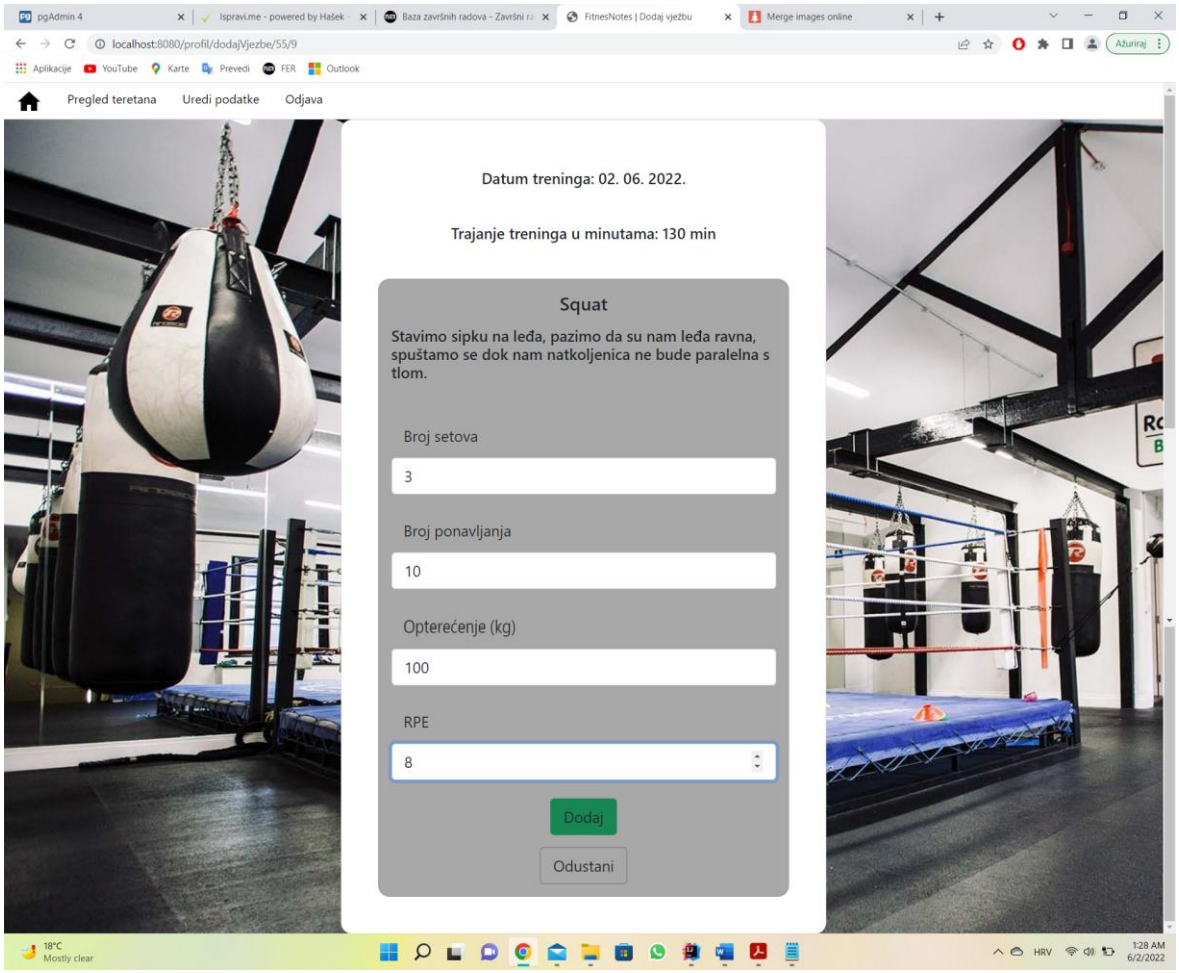
Pritiskom na gumb 'Dodaj trening' klijent stvara novi trening (Slika 3.10) te nakon toga odabire koju vježbu će dodati u svoj trening, koliko je bilo opterećenje na toj vježbi, koliko je ponavljanja i serija napravio te koliki je RPE bio na toj vježbi. Slika 3.11 prikazuje listu vježbi poredanih po mišićnoj skupini. Klikom na mišićnu skupinu otvara se lista vježbi te mišićne skupine. Nakon odabira određene vježbe klijent ispunjava formu o dodatnim informacijama o toj vježbi u svom treningu (broj setova, broj ponavljanja, opterećenje, rpe). Formu za dodavanje određene vježbe u trening možemo vidjeti na Slika 3.12.



Slika 3.10 - dodavanje novog treninga

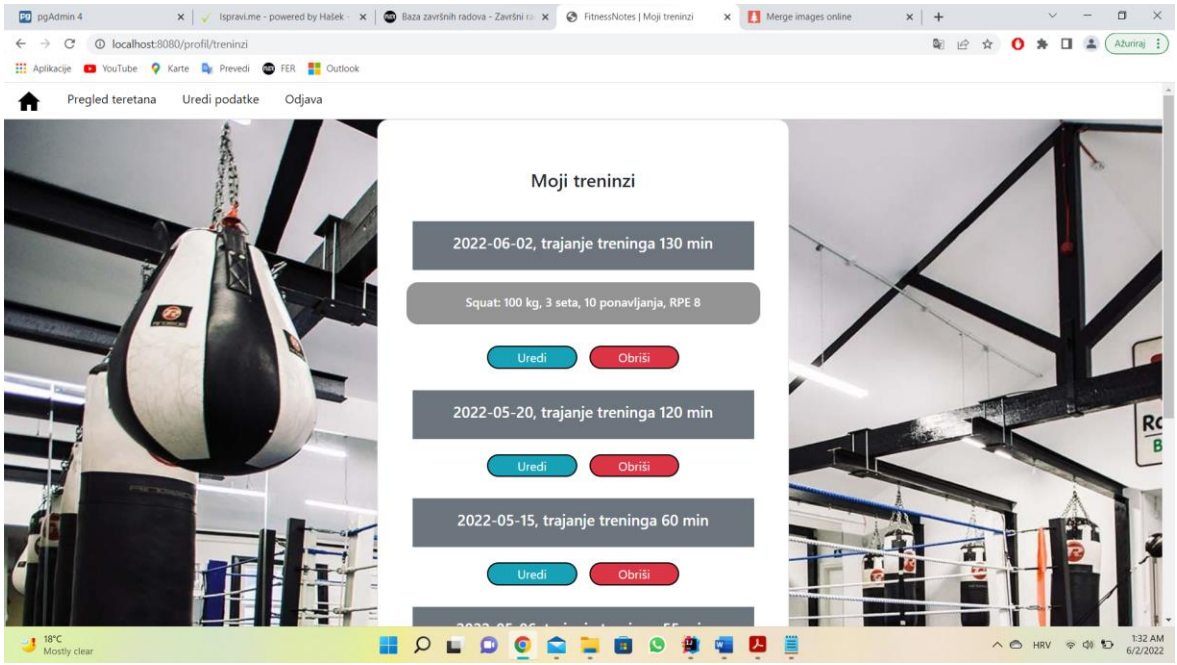


Slika 3.11 - klijent odabire koju vježbu će dodati u trening (klikom na mišićnu skupinu prikazuju se sve vježbe te mišićne skupine)



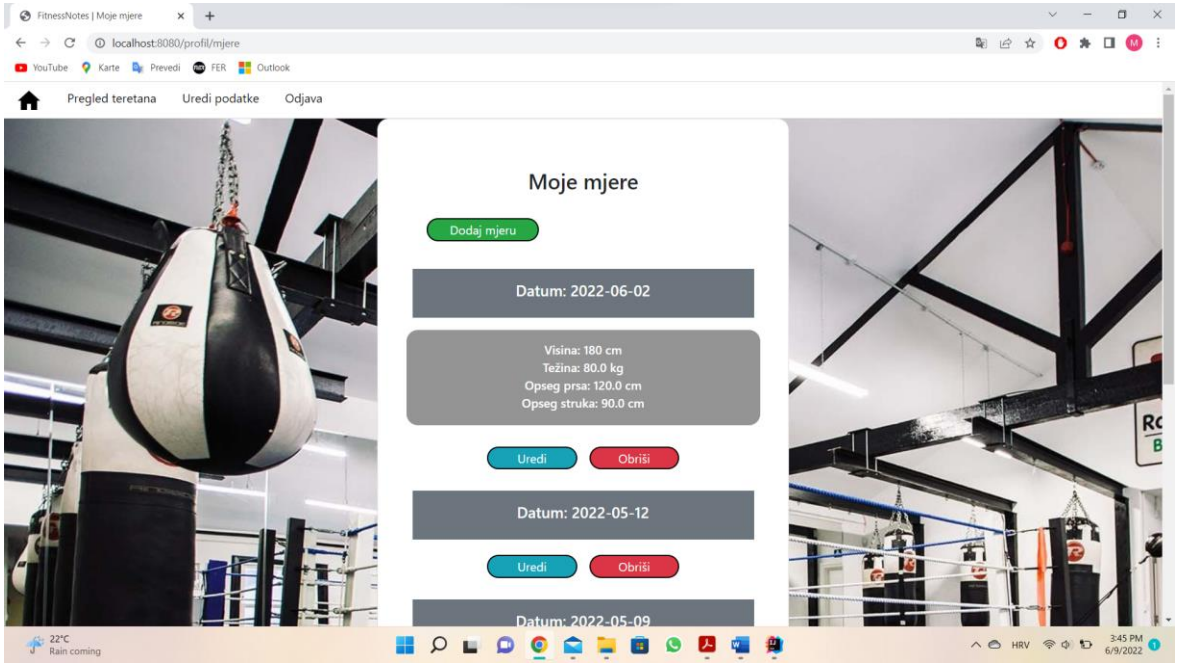
Slika 3.12 - forma za dodavanje odabrane vježbe u trening

Nakon što doda novi trening i vježbe u njega, klijentu je sažetak tog treninga dostupan u odjeljku 'Moji treninzi'. Radi demonstracije, pretpostavimo da smo vježbu sa Slika 3.12 dodali u naš trening. Na Slika 3.13 vidimo arhivu treninga nakon dodavanja treninga sa Slika 3.10 i vježbe sa Slika 3.12.



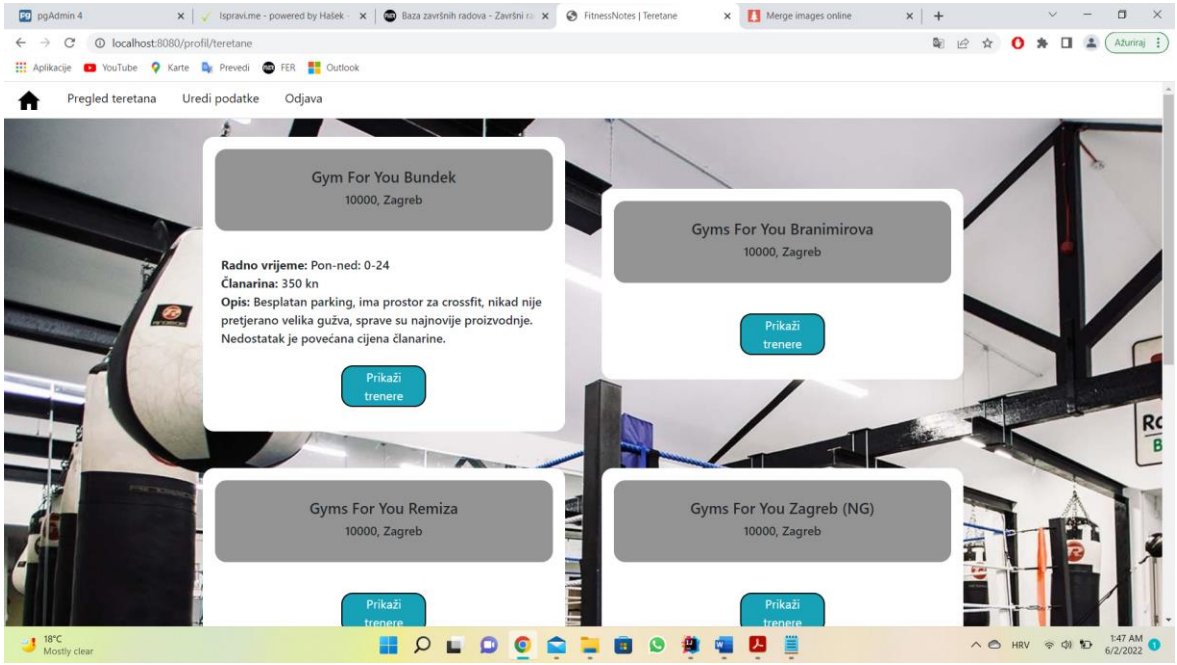
Slika 3.13 – prikaz arhive treninga nakon dodavanja gornjeg treninga

Klijent također može dodavati svoje mjere i tako voditi evidenciju o njima. Stranica za prikaz mjera je prikazana na Slika 3.14.

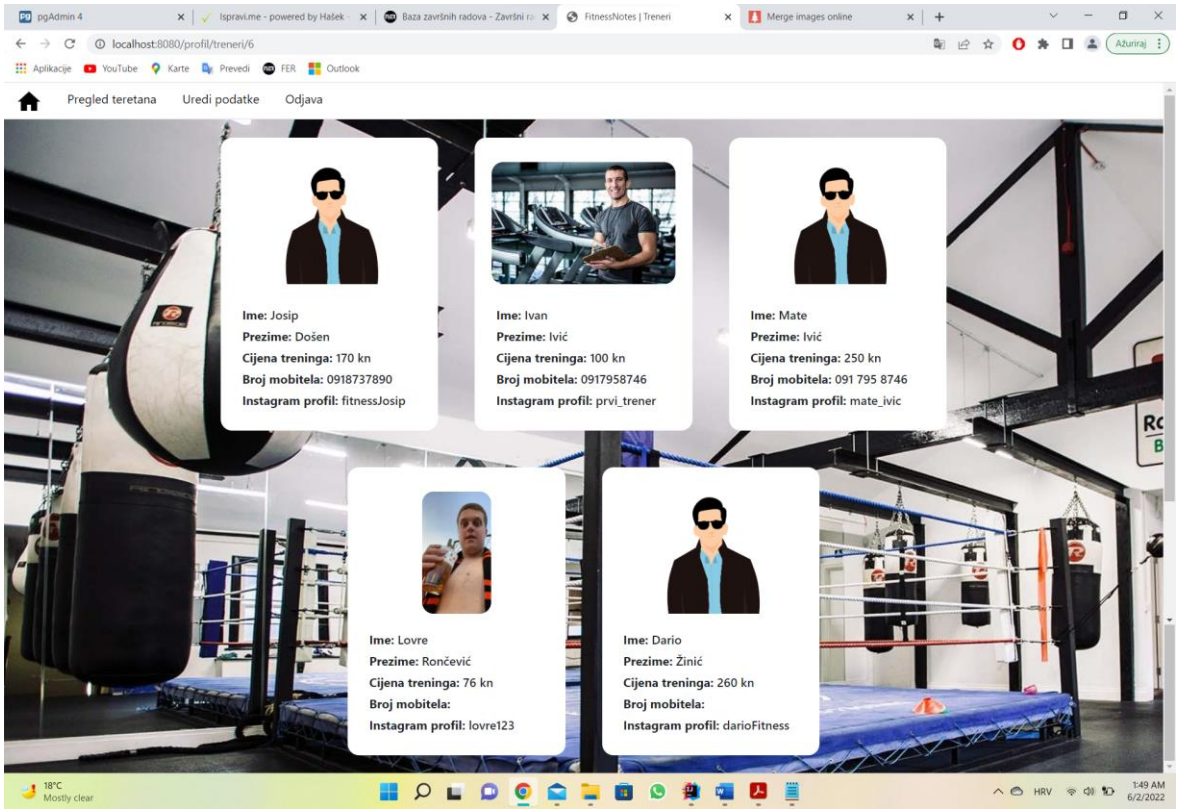


Slika 3.14 – prikaz klijentovih mjera

U svakom trenutku klijent može pregledati sve teretane klikom na gumb 'Pregled teretana', nakon čega ima opciju pogledati sve aktivne trenere pojedine teretane. Klikom na teretanu klijent dobiva dodatne informacije o teretani (Slika 3.15). Klikom na gumb 'Prikaži trenere' klijentu se prikazuje slika svih trenera koji su aktivni u toj teretani (Slika 3.16).



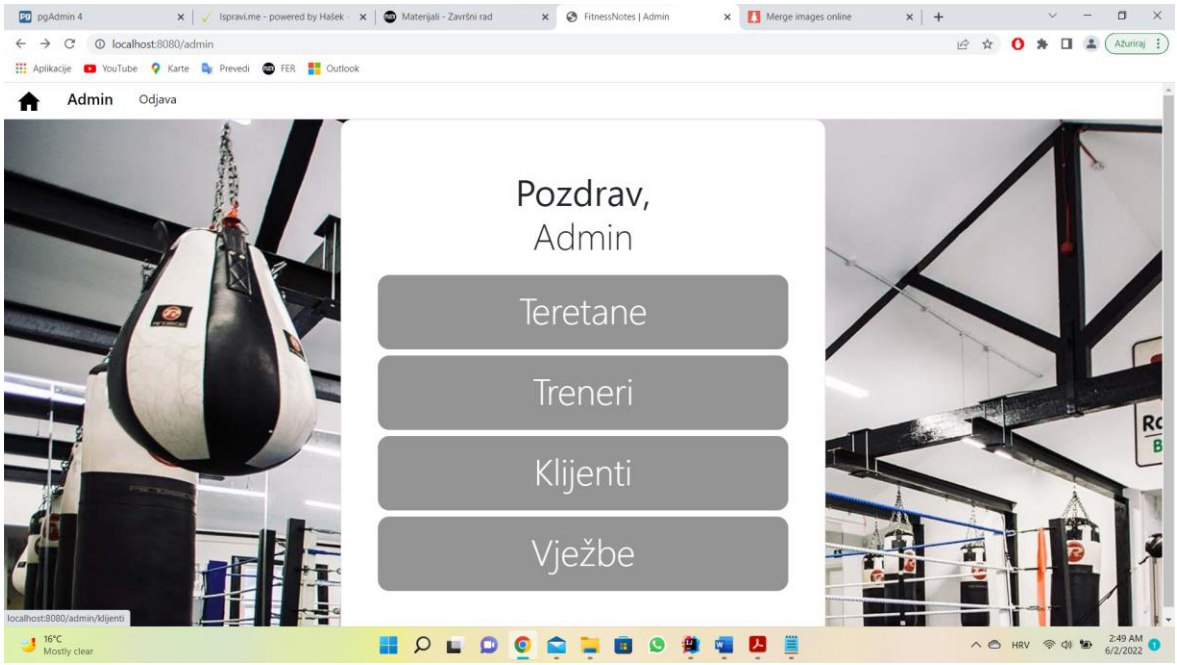
Slika 3.15 – pregled teretana, korisnik je kliknuo na Gym For You Bundek



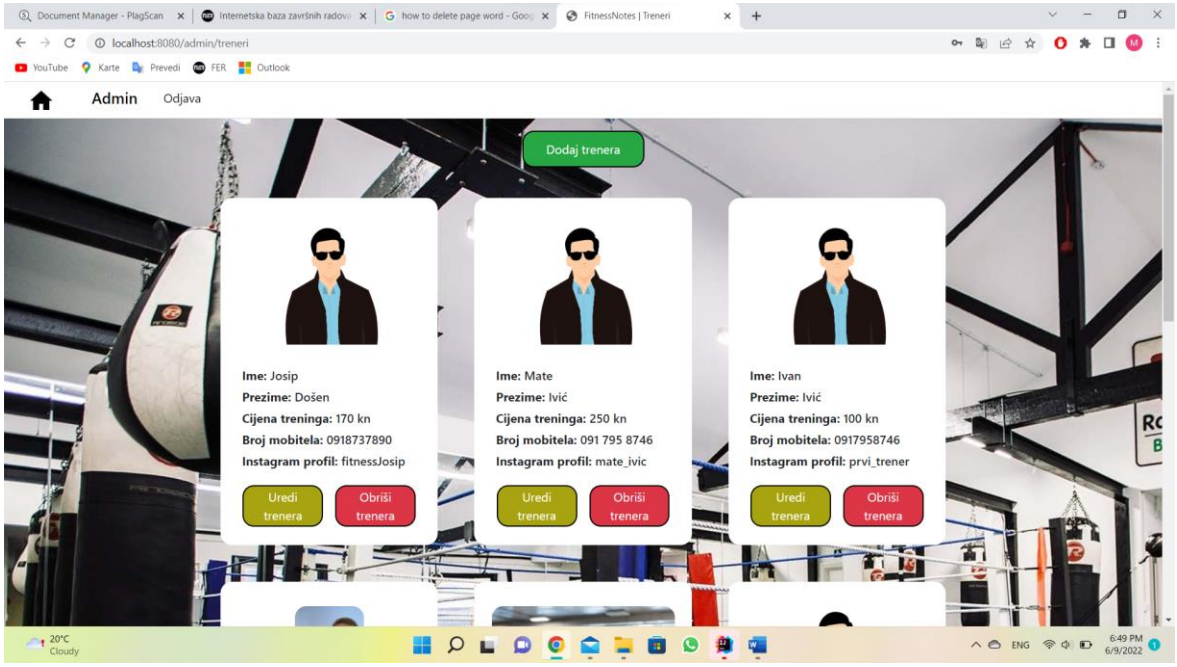
Slika 3.16 – prikaz svih trenera teretane Gym For You Bundek

3.2.3. Vodič za administratora

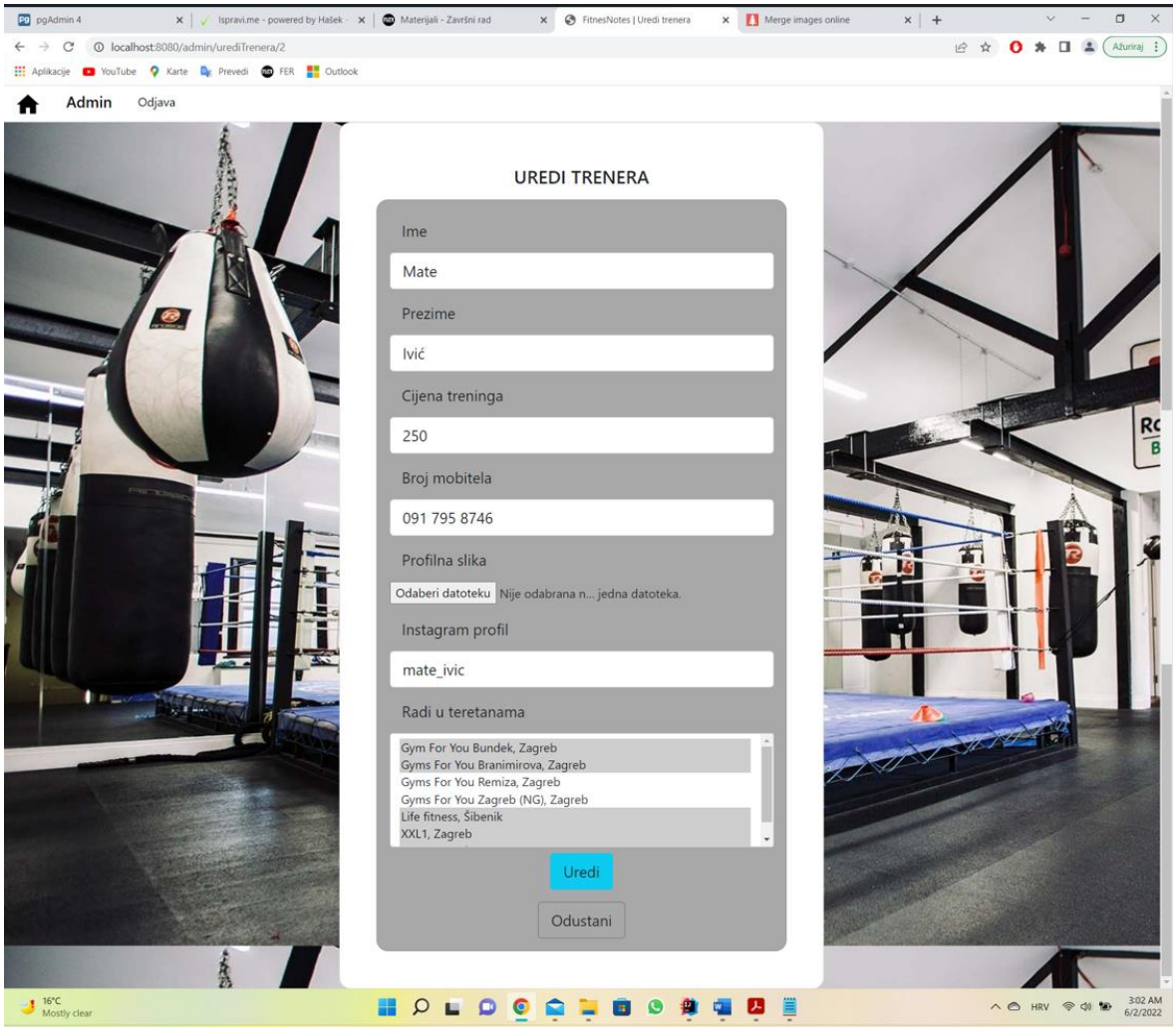
Nakon prijave u sustav, administratoru se prikazuje njegovo korisničko sučelje (Slika 3.17) iz kojeg su mu dostupne sve funkcionalnosti iz poglavlja 1.1 - Analiza zahtjeva. Pritiskom na gumb 'Treneri' administratoru se prikazuju svi treneri koji se mogu uređivati, brisati, te je moguće stvoriti novog trenera (Slika 3.18). Slika 3.19 prikazuje formu pomoću koje možemo urediti trenera. Ista logika vrijedi i za gumb 'Teretane'. Pritiskom na gumb 'Klijenti' administrator dobiva listu svih registriranih klijenata te ih može obrisati po želji. Pritiskom na gumb 'Vježbe' administratoru se prikazuju sve vježbe grupirane po mišićnim skupinama (Slika 3.20). Radi demonstracije dodat ćemo novu vježbu imena 'Biceps curl' pod mišićnu grupu 'Ruke' (Slika 3.21 i Slika 3.22).



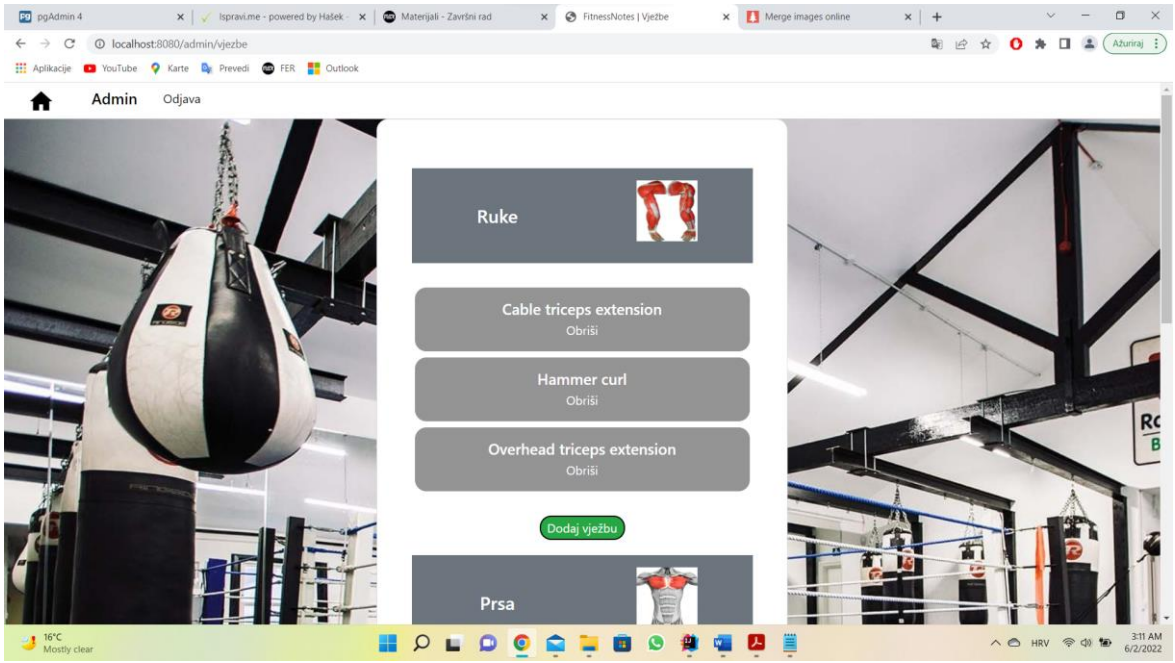
Slika 3.17 - početna stranica administratora



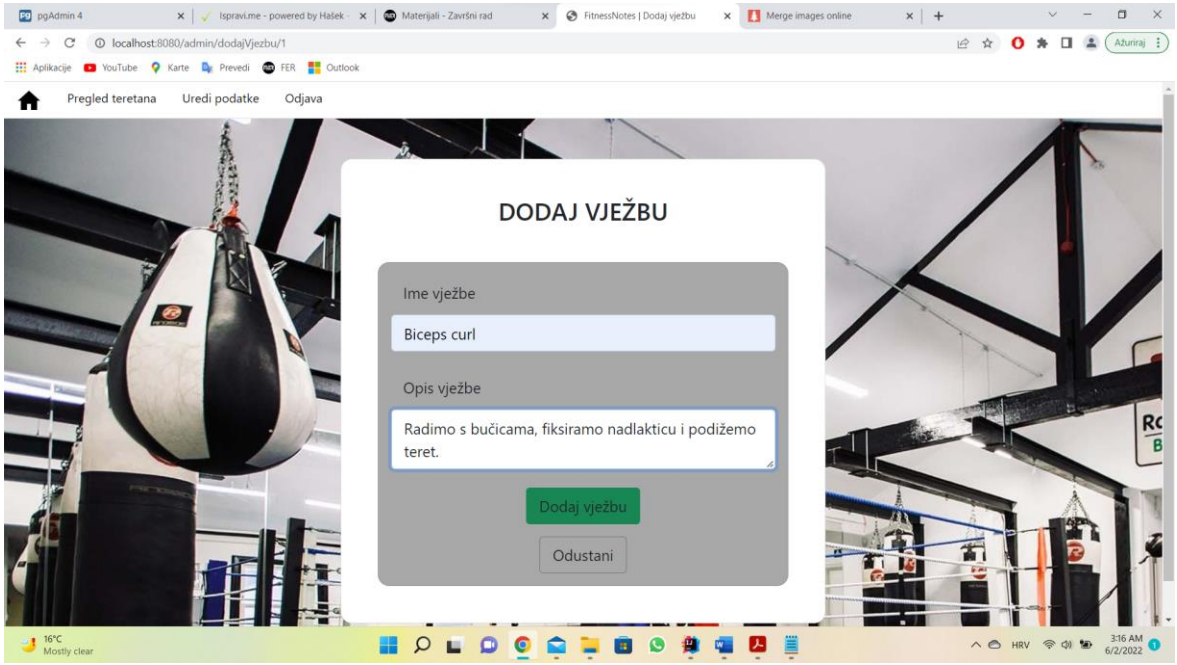
Slika 3.18 – prikaz svih trenera



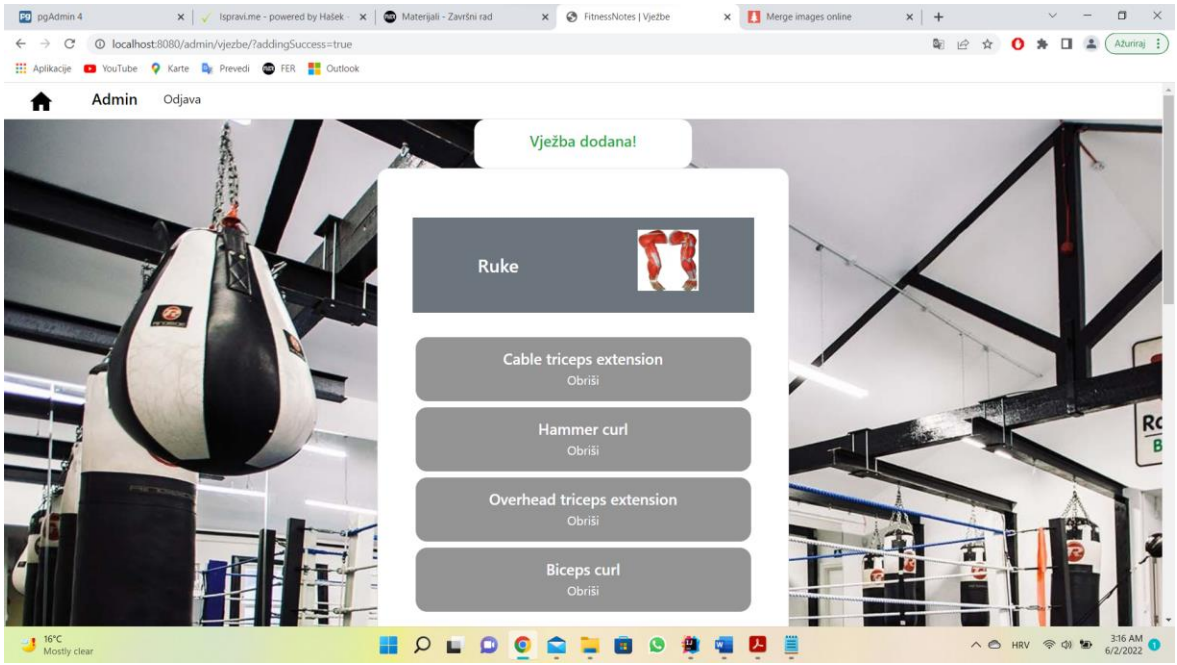
Slika 3.19 – forma za uređivanje trenera Mate Ivić



Slika 3.20 – prikaz vježbi prije dodavanja vježbe biceps curl



Slika 3.21 – dodavanje nove vježbe



Slika 3.22 - prikaz vježbi nakon dodavanja vježbe biceps curl

Zaključak

Korištenjem gore spomenutih alata i tehnologija uspješno je odrađen zadatak izrade baze podataka i web-aplikacije za trenažni proces u teretani. Kroz vrijeme od jednog semestra završena je cijela web-aplikacija.

Nedvojbeno je da aplikacija ima prostora za napredak, ali s obzirom na to da mi je ovo prva samostalna izrada web-aplikacije, zadovoljan sam sa svojim radom i uspjehom.

Kroz izradu ove aplikacije pobliže sam se upoznao s mnogim alatima i tehnologijama za izradu web-aplikacija kao što su Spring Boot, HTML/CSS/JS, Thymeleaf, Bootstrap. Također sam ponovio i nadam se utvrdio postupak izrade relacijske baze podataka. Uvelike mi je pomoglo gradivo koje sam naučio u prethodnim godinama studiranja, te smatram da bi ovaj rad bilo nemoguće napraviti bez znanja koje sam dobio na studiju. Naravno dio znanja potrebnog za ovaj završni rad sam morao sam steći, i drago mi je što jesam jer smatram da će mi sigurno olakšati kasniji studij i rad u struci.

Literatura

[1] Nastavni materijali kolegija Baze podataka, Fakultet elektrotehnike i računarstva, 2021.

[2] Nastavni materijali kolegija Razvoj programske potpore za Web, Fakultet elektrotehnike i računarstva, 2021.

[3] W3Schools, https://www.w3schools.com/, 20.05.2022.

[4] Java Brains Spring Security tutorials, https://www.youtube.com/playlist?list=PLqq-6Pq4lTTYTEooakHchTGglSvkZAjnE, 15.05.2021.

[5] Bootstrap, https://getbootstrap.com/, 17.05.2022.

[6] Stack Overflow, https://stackoverflow.com/, 16.05.2022.

[7] Thymeleaf tutorials, https://www.thymeleaf.org/doc/tutorials/, 19.05.2022.

[8] Baeldung Spring tutorials, https://www.baeldung.com/, 12.05.2022.

Sažetak

Baza podataka i web-aplikacija za trenažni proces u teretani

Tema ovog završnog rada je izrada baze podataka i web-aplikacije za trenažni proces u teretani. Sustav pohranjuje podatke o odrađenim treninzima, vježbama u treninzima i postignutim rekordima. Sustav također omogućava pohranu, izmjenu i brisanje podataka bitnih za evidenciju odrađenih treninga i praćenje napretka u treningu. U bazu podataka su pohranjeni treneri i teretane koje klijent može samo pregledati, dok administrator ima potpunu manipulaciju nad tim podacima.

Baza podataka je izrađena koristeći sustav za upravljanje bazama podataka PostgreSQL, a aplikacija koristeći radni okvir Spring Boot i Thymeleaf.

Ključne riječi: Web-aplikacija, PostgreSQL baza podataka, Spring Boot projekt, fitness notes, trenažni proces, Java

Summary

Database and web application for the gym training process

The topic of this final paper is the creation of a database and web application for the training process in the gym. The system stores data on completed trainings, training sessions and achieved records. The system also allows the storage, modification and deletion of data relevant to the records of training and monitoring of training progress. The database stores coaches and gyms that the client can only view, while the administrator has a complete manipulation of this data.

The database was created using the PostgreSQL database management system, and the application using the Spring Boot and Thymeleaf frameworks.

Keywords: Web application, PostgreSQL database, Spring Boot project, fitness notes, training process, Java