

# Programsko inženjerstvo

Ak. god. 2021./2022.

## CodeShark

Dokumentacija, Rev. 2. 0

Grupa: DomeFanClub  
Voditelj: Marko Damjanić

Datum predaje: 14. 1. 2022.

Nastavnik: Hrvoje Nuić

# Sadržaj

<b>1</b>	<b>Dnevnik promjena dokumentacije</b>	<b>3</b>
<b>2</b>	<b>Opis projektnog zadatka</b>	<b>5</b>
<b>3</b>	<b>Specifikacija programske potpore</b>	<b>9</b>
3.1	Funkcionalni zahtjevi . . . . .	9
3.1.1	Obrasci uporabe . . . . .	11
3.1.2	Sekvencijski dijagrami . . . . .	23
3.2	Ostali zahtjevi . . . . .	26
<b>4</b>	<b>Arhitektura i dizajn sustava</b>	<b>27</b>
4.1	Baza podataka . . . . .	28
4.1.1	Opis Tablica . . . . .	28
4.1.2	Dijagram baze podataka . . . . .	34
4.2	Dijagram razreda . . . . .	35
4.3	Dijagram stanja . . . . .	37
4.4	Dijagram aktivnosti . . . . .	38
4.5	Dijagram komponenti . . . . .	39
<b>5</b>	<b>Implementacija i korisničko sučelje</b>	<b>40</b>
5.1	Korištene tehnologije i alati . . . . .	40
5.2	Ispitivanje programskog rješenja . . . . .	41
5.2.1	Ispitivanje komponenti . . . . .	41
5.2.2	Ispitivanje sustava . . . . .	43
5.3	Dijagram razmještaja . . . . .	47
5.4	Upute za puštanje u pogon . . . . .	48
<b>6</b>	<b>Zaključak i budući rad</b>	<b>52</b>
	<b>Popis literature</b>	<b>53</b>
	<b>Indeks slika i dijagrama</b>	<b>54</b>

**Dodatak: Prikaz aktivnosti grupe**

55

# 1. Dnevnik promjena dokumentacije

Rev.	Opis promjene/dodatka	Autori	Datum
0.1	Napravljen predložak.	Marko Damjanić	26.10.2021.
0.2	Napravljen opis projekta	Marko Damjanić	7.11.2021.
0.3	Dodani dionici i aktori	Fran Jelavić	16.11.2021.
0.4	Dodan prvi dio obrazaca uporabe	Luka Maros	16.11.2021.
0.4.1	Dodan drugi dio obrazaca uporabe	Antonio Griparić	16.11.2021.
0.5	Dodana arhitektura i dizajn sustava	Fran Jelavić	16.11.2021.
0.6	Uspostavljena Baza podataka	Marko Damjanić	16.11.2021.
0.7	Dodani Sekvencijski dijagrami	Luka Maros	18.11.2021.
0.7.1	Dodani ostali zahtjevi	Marko Damjanić	18.11.2021.
0.7.2	Popravljen baza podataka	Marko Damjanić	18.11.2021.
0.7.3	Dodani dijagrami obrazaca uporabe	Antonio Griparić	19.11.2021.

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Rev.	Opis promjene/dodatka	Autori	Datum
0.8	Dodani dijagrami razreda	Roko Mihalić	19.11.2021.
0.9	Korigiranje teksta	Marko Damjanić	19.11.2021.
1.0	Provjera dokumentacije	Marko Damjanić	19.11.2021.
1.1	Dodan dijagram stanja	Marko Damjanić	13.1.2022.
1.2	Dodan dijagram aktivnosti	Marko Damjanić	13.1.2022.
1.3	Dodan dnevnik sastanaka	Marko Damjanić	14.1.2022.
1.4	Dodana tablica aktivnosti	Marko Damjanić	14.1.2022.
1.5	Popravljen dokumentacija baze	Marko Damjanić	14.1.2022.
1.6	Dodan Zaključak i budući rad	Marko Damjanić	14.1.2022.
1.7	Dodane korištene tehnologije i alati	Marko Damjanić	14.1.2022.
1.8	Dodan dijagram razmještaja	Marko Damjanić	14.1.2022.
1.8.1	Dodano ispitivanje sustava	Marko Damjanić	14.1.2022.
2.0	Dodano puštanje u pogon	Marko Damjanić	14.1.2022.

## 2. Opis projektnog zadatka

Rješavanjem programskih zadataka i kompetitivnim programiranjem današnji programeri razvijaju svoje logičke vještine, ubrzavaju brzinu razvoja svoga koda, pišu kvalitetniji kod i lakše briljiraju kod firmi koje cijene visoke kompetitivnosti programera.

Cilj ovog projekta je razviti programsku podršku za stvaranje web aplikacije "CodeShark" koja će korisniku omogućiti da rješava programske zadatke prik ladne njegovim interesima i razini znanja, te da se natječe na natjecanjima organiziranih od strane voditelja. Zadatak voditeljima je stvarati zadatke korisnicima te organizirati natjecanja koja će najbolje natjecatelje nagraditi trofejima prikazanim na profilu korisnika.

Korisnici na stranici mogu pregledavati i rješavati zadatke, natjecati se u natjecanjima koja su dostupna u kalendaru, te pratiti profile ostalih natjecatelja i voditelja natjecanja. Neregistriranim korisnicima se prilikom pokretanja zadataka ili tijekom pokušaja prijave na natjecanje otvara stranica za log-in pomoću već registriranog profila (potrebno upisati korisničko ime i lozinku) te ako nemaju već registriran profil imaju opciju registracije. Za registraciju novog profila potrebni su sljedeći podaci:

- korisničko ime
- fotografija
- lozinka
- ime
- prezime
- email adresa
- željena uloga na koju se prijavljuje (voditelj natjecanja ili natjecatelj)

Registracija se završava potvrdom preko email adrese. Automatski se korisniku dodjeljuju prava natjecatelja a za ulogu i prava voditelja mora dodatno potvrditi i administrator. Registrirani korisnik može pregledavati, mijenjati osobne podatke i izbrisati svoj korisnički račun.

*Natjecatelj* može pristupiti stranici s zadacima za vježbanje, na kojoj je prikazan popis svih postojećih zadataka koji su već bili iskorišteni na jednom od natjecanja (javni zadatci) te ostalih zadataka koje su voditelji dodali. Zadaci se mogu pretraživati po korisničkom imenu autora, sortirati po težini, statusu riješenosti te raznim drugim kategorijama podjele. Pritiskom na zadatak otvara se stranica tog zadatka gdje mu je omogućeno učitavanje programskog rješenja u aplikaciju. Učitani program se ovisno o potrebi prevodi u računalni kod, te potom provjerava na temelju zadanih primjera. Izlaz programa mora biti jednak rješenju i izvesti se unutar zadanog vremena. Broj bodova koje je natjecatelj ostvario se dodjeljuje proporcionalno broju točno riješenih primjera.

Natjecatelj također može pristupiti stranici s natjecanjima. Na njoj je prikazan interaktivni kalendar s svim budućim, trenutnim i prošlim natjecanjima. Također sadrži i uređenu listu natjecanja koja se može sortirati po autoru, državi za koju je namijenjeno te klasi natjecanja (za početnike, veterane i otvoren upad). Pritiskom na natjecanje otvara se stranica tog natjecanja s svim bitnim detaljima. Ako se to natjecanje još nije održalo, natjecatelj se može na njega prijaviti a kada bude vrijeme početka natjecanje se može pridružiti.

U trenutku početka natjecanja svi zadatci postaju vidljivi aktivnim natjecateljima. Za svaki zadatak, natjecatelj može poslati datoteku s programskim kodom. Završetkom natjecanja objavljuje se rang lista svih natjecatelja po ostvarenom broju bodova. Natjecateljima se na temelju postignuća za prva tri mjesta dodjeljuje pehar. Za izračun osvojenih bodova na natjecanju potrebno je uzeti u obzir provedeno vrijeme za rješavanje zadatka i postotak točnih primjera.

Natjecatelj nakon natjecanja može vidjeti i popis svih učitanih rješenja od nekog drugog natjecatelja. Slično, svaki zadatak ima popis svih natjecatelja koji su učitali neko rješenje za taj zadatak, broj točnih primjera po najboljem učitavanju od natjecatelja, prosječnom vremenu izvršavanja po primjeru i gumb za dohvat učitano rješenja koji se aktivira samo onim natjecateljima koji su već potpuno točno riješili zadatak.

Ako se natjecanje već održalo natjecatelj može sebi pokrenuti virtualno natjecanje. Virtualno natjecanje je aktivno samo za natjecatelja koji ga je pokrenuo. Pri isteku ograničenog vremena ili po želji natjecatelja virtualno natjecanje završava. Pri završetku se natjecatelja rangira u usporedbi s službenim rezultatima korisnika koji su prisustvovali tom natjecanju.

Natjecatelj isto tako može napraviti virtualno natjecanje gdje aplikacija nasumično odabere zadatke, ali tako da ravnomjerno rasporedi težine zadataka prema količini bodova pojedinog zadatka.

Postoje još 2 vrste korisnika, oboje s većim pravima od natjecatelja, a to su:

- voditelj
- administrator

Voditelj ima mogućnosti učitati nove zadatke u aplikaciju i organizirati natjecanja. Za postaviti novi zadatak potrebno je:

- naziv zadatka
- broj bodova (težina 1-5)
- vremensko ograničenje izvršavanja programa
- tekst zadatka
- primjeri za evaluaciju (ulaz i izlaz programa)
- privatnost zadatka

Privatni zadatak automatski postaje javan završetkom natjecanja.

Voditelj može izraditi natjecanje i ono svima postaje vidljivo u kalendaru natjecanja. Za organizaciju natjecanja su mu potrebni:

- naziv natjecanja
- tekst natjecanja
- vrijeme početka i završetka natjecanja (max 48 sati)
- broj zadataka
- odabrani privatni zadaci koje je voditelj učitao
- sličica pehara
- država natjecanja

Voditelj može uređivati vlastito objavljene zadatke i natjecanje. Uređivanjem zadatka se ne mijenjaju prethodno ostvareni rezultati na tom zadatku.

Administrator sustava ima najveće ovlasti. On može vidjeti popis svih registriranih korisnika i njihovih osobnih podataka te im mijenjati dodijeljena prava i osobne podatke. Ima potpun pristup bazi zadataka i natjecanja te ih ima pravo obrisati.



Uz vlastiti profil, u web aplikaciji postoji stranica s listom profila preko koje se mogu pogledati statistike i trofeji drugih korisnika. Na profilu natjecatelja su ispisane statistike o broju točno riješenih i broju isprobanih zadataka. Na profilima je također kalendar natjecanje na kojem je korisnik prisustvovao, a za voditelje i ona natjecanja koja su organizirali. Profili voditelja također sadrže popis njihovih učitanih zadataka s mogućnošću sortiranja.

Od sličnih rješenja istaknut je <https://leetcode.com/>. Za razliku od leetcode-a CodeShark ima mogućnosti virtualnih natjecanja koja pridonose boljoj pripremi korisnika. CodeShark se također više fokusira na svojom jednostavnosti i interaktivnosti.

The screenshot shows the LeetCode website interface. At the top, there are navigation tabs: Explore, Problems (selected), Contest, Discuss, Interview, and Store. To the right, there are links for Premium, Register, and Sign in. Below the navigation, there are several promotional banners: 'LeetCode Interview', 'Detailed Explanation of Graph', and 'Weekly Contest 267'. To the right of these banners is a calendar for the weekly contest, showing the current date (Day 8) and the contest schedule (Sunday, November 14, 2:30 - 4:00AM UTC). Below the banners, there are three study plans: '14 Days Study Plan to Crack Algo', '2 Weeks Study Plan to Tackle DS', and 'Ultimate DP Study Plan'. To the right of these study plans is a 'Study Plan' section with a 'Complete and win badges' button. Below the study plans, there is a 'LeetCode's Pick' section with a 'Start Creating' button. At the bottom, there is a 'Featured Lists' section with links to 'LeetCode Curated Algo 170', 'LeetCode Curated SQL 70', 'Top 100 Liked Questions', and 'Top Amazon Questions'. The main content area shows a list of problems with columns for Status, Title, Solution, Acceptance, Difficulty, and Frequency. The first problem listed is '96. Unique Binary Search Trees' with a solution link, 57.1% acceptance, and Medium difficulty.

Status	Title	Solution	Acceptance	Difficulty	Frequency
📅	96. Unique Binary Search Trees	<a href="#">Solution</a>	57.1%	Medium	<a href="#">Frequency</a>
—	1. Two Sum	<a href="#">Solution</a>	47.8%	Easy	<a href="#">Frequency</a>
—	2. Add Two Numbers	<a href="#">Solution</a>	37.2%	Medium	<a href="#">Frequency</a>
—	3. Longest Substring Without Repeating Chara...	<a href="#">Solution</a>	32.4%	Medium	<a href="#">Frequency</a>

Slika 2.1: <https://leetcode.com/problemset/all/>

## 3. Specifikacija programske potpore

### 3.1 Funkcionalni zahtjevi

Dionici:

1. Natjecatelj
2. Voditelj
3. Administrator
4. Razvojni tim

Aktori i njihovi funkcionalni zahtjevi:

1. Neregistrirani/neprijavljeni korisnik (inicijator) može:
  - (a) pretraživati natjecanja i pristupiti popisu zadataka s natjecanja i sudionika natjecanja
  - (b) pretraživati i čitati zadatke s natjecanja
  - (c) pristupiti informacijama tuđeg korisničkog profila:
    - i. korisničko ime, ime i prezime, i osvojene nagrade
    - ii. statistike u što se ubraja broj točno riješenih i broj isprobanih zadataka
    - iii. popis natjecanja na kojima je korisnik prisustvovao ili natjecanja koja je održavao
  - (d) se registrirati u sustav, stvoriti novi korisnički račun za koji:
    - i. su mu potrebni korisničko ime, lozinka, ime i prezime te e-mail adresa
    - ii. mora izabrati razinu pristupa "natjecatelj" ili "voditelj"
2. Natjecatelj (inicijator) može:
  - (a) pregledavati i mijenjati osobne podatke
  - (b) izbrisati svoj korisnički račun
  - (c) pristupiti zadatcima za vježbu i rješavati ih
  - (d) prijaviti i odjaviti se na natjecanja i pristupiti natjecanjima

- (e) osvojiti nagrade
- (f) pokrenuti virtualno natjecanje

3. Voditelj (inicijator) može:

- (a) pregledavati i mijenjati osobne podatke
- (b) izbrisati svoj korisnički račun
- (c) pristupiti zadatcima za vježbu i rješavati ih
- (d) prijaviti i odjaviti se na natjecanja i pristupiti natjecanjima
- (e) osvojiti nagrade
- (f) pokrenuti virtualno natjecanje
- (g) pokrenuti i voditi natjecanje
- (h) kreirati zadatke
- (i) mijenjati i brisati vlastite zadatke

4. Administrator (inicijator) može:

- (a) pristupiti zadatcima za vježbu i rješavati ih
- (b) prijaviti i odjaviti se na natjecanja i pristupiti natjecanjima
- (c) osvojiti nagrade
- (d) pokrenuti virtualno natjecanje
- (e) pokrenuti i voditi natjecanje
- (f) kreirati zadatke
- (g) mijenjati vlastite zadatke
- (h) vidjeti popis svih registriranih korisnika i njihovih osobnih podataka
- (i) korisnike brisati i mijenjati im razinu pristupa aplikaciji (natjecatelj, voditelj, administrator)
- (j) pristupiti statistici

### 3.1.1 Obrasci uporabe

#### Opis obrazaca uporabe

##### UC1 - Registracija

- **Glavni sudionik:** Korisnik
- **Cilj:** Registracija u sustav
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za registraciju
  2. Korisnik unosi potrebne korisnicke podatke
  3. Korisnik prima obavijest o uspješnoj registraciji
- **Opis mogućih odstupanja:**
  - 2.a Odabir već zauzetog korisničkog imena i/ili e-maila, unos korisničkog podatka u nedozvoljenom formatu ili pružanje neispravnoga e-maila
    1. Sustav obavještava korisnika o neuspjelom upisu i vraća ga na stranicu za registraciju
    2. Korisnik mijenja potrebne podatke te završava unos ili odustaje od registracije

##### UC2 - Prijava

- **Glavni sudionik:** Korisnik
- **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 funkcijama
- **Opis mogućih odstupanja:**
  - 2.a Neispravno korisničko ime/lozinka
    1. Sustav obavještava korisnika o neuspjelom upisu i vraća ga na stranicu za registraciju

##### UC3 - Pregled mogeg profila

- **Glavni sudionik:** Korisnik

- **Cilj:** Pregledati osobne podatke
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju "Moj Profil"
  2. Aplikacija prikazuje osobne podatke korisnika

#### UC4 - Promjena podataka na mojem profilu

- **Glavni sudionik:** Korisnik
- **Cilj:** Promjeniti osobne podatke
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju "Uredi profil"
  2. Korisnik mijenja svoje osobne podatke
  3. Korisnik sprema promjene
  4. Baza podataka se ažurira
- **Opis mogućih odstupanja:**
  - 2.a Korisnik promjeni svoje osobne podatke, ali ne odabire opciju "Spremi"
    1. Sustav obavještava korisnika da nije spremio podatke prije izlaza iz prozora.

#### UC5 - Odjava iz sustava

- **Glavni sudionik:** Korisnik
- **Cilj:** Odjava iz sustava
- **Sudionici:** Korisnik
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju "Odjava"
  2. Sustav vraća korisnika na početnu stranicu

#### UC6 - Brisanje mog profila

- **Glavni sudionik:** Korisnik
- **Cilj:** Brisanje korisničkog profila
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen

- **Opis osnovnog tijeka:**

1. Korisnik odabire opciju "Izbriši Moj Profil"
2. Sustav odjavljuje korisnika i vraća ga na početnu stranicu
3. Baza Podataka se ažurira

### UC7 - Pregled zadataka

- **Glavni sudionik:** Korisnik
- **Cilj:** Pregledati sve dostupne zadatke za vježbu
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju "Zadaci"
  2. Korisniku se prikažu svi dostupni zadaci za vježbu

### UC8 - Rješavanje zadataka za vježbu

- **Glavni sudionik:** Korisnik
- **Cilj:** Pokretanje i rješavanje zadataka za vježbu
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire jedan od zadataka za vježbu
  2. Sustav prebacuje korisnika na stranicu za rješavanje zadataka

### UC9 - Dodavanje zadataka

- **Glavni sudionik:** Voditelj
- **Cilj:** Dodati novi zadatak u bazu zadataka za vježbu
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik mora imati ovlasti "Voditelj"
- **Opis osnovnog tijeka:**
  1. Voditelj odabire opciju "Dodaj novi zadatak"
  2. Voditelj upisuje tekst, težinu, rješenje i vremensko ograničenje za zadatak
  3. Voditelj odabire opciju "Završi"
  4. Baza podataka se ažurira
- **Opis mogućih odstupanja:**
  - 2.a Voditelj unosi podatke u nedozvoljenom formatu
    1. Sustav obavještava voditelja o neuspjelom upisu podataka

## 2. Voditelj popravljaj unose

### UC10 - Pregled sudionika koji su rješavali određeni zadatak

- **Glavni sudionik:** Natjecatelj
- **Cilj:** Pregledati sudionike koji su rješavali određeni zadatak
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. Natjecatelj odabire jedan od zadataka
  2. Natjecatelju se otvara lista svih koji su ga rješavali

### UC11 - Brisanje zadataka

- **Glavni sudionik:** Voditelj
- **Cilj:** Izbrisati zadatak iz baze zadataka za vježbu
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik mora imati ovlasti "Voditelj"
- **Opis osnovnog tijeka:**
  1. Voditelj odabire jedan od zadataka za vježbu
  2. Voditelj odabire opciju "Izbriši zadatak"
  3. Baza podataka se ažurira

### UC12 - Uređivanje zadataka za vježbu

- **Glavni sudionik:** Voditelj
- **Cilj:** Urediti zadatak iz baze zadataka za vježbu
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik mora imati ovlasti "Voditelj"
- **Opis osnovnog tijeka:**
  1. Voditelj odabire jedan od zadataka za vježbu
  2. Voditelj odabire opciju "Uredi zadatak"
  3. Voditelj upisuje tekst, težinu, rješenje i vremensko ograničenje za zadatak
  4. Voditelj odabire opciju "Završi"
  5. Baza podataka se ažurira
- **Opis mogućih odstupanja:**
  - 2.a Voditelj unosi podatke u nedozvoljenom formatu
    1. Sustav obavještava voditelja o neuspjelom upisu podataka
    2. Voditelj popravljaj unose

**UC13 - Pregled natjecanja**

- **Glavni sudionik:** Natjecatelj
- **Cilj:** Pregledati listu svih natjecanja
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju "Natjecanja"
  2. Sustav prenosi korisnika na stranicu s kalendarom i listom natjecanja

**UC14 - Pokretanje virtualnog natjecanja**

- **Glavni sudionik:** Natjecatelj
- **Cilj:** Vježba putem virtualnih natjecanja
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. Voditelj odabire opciju "Natjecanja"
  2. Korisnik odabire "Novo virtualno natjecanje"
  3. Korisnik započinje rješavanje natjecanja

**UC15 - Dodavanje natjecanja**

- **Glavni sudionik:** Voditelj
- **Cilj:** Dodati natjecanje
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik mora biti prijavljen i imati ulogu voditelja
- **Opis osnovnog tijeka:**
  1. Voditelj odabire karticu "Natjecanja"
  2. Voditelj dodaje novo natjecanje
  3. Voditelj unosi potrebne podatke o natjecanju
  4. Promjene se upisuju u bazu podataka
- **Opis mogućih odstupanja:**
  - 3.a Voditelj unosi podatke u nedozvoljenom formatu
    1. Sustav obavještava voditelja o neuspjelom upisu podataka
    2. Voditelj popravljiva neispravne unose i stvara natjecanje ili odustaje od stvaranja natjecanja

**UC16 - Pokretanje natjecanja**



- **Glavni sudionik:** Voditelj
- **Cilj:** Pokrenuti natjecanje
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik mora biti prijavljen i imati ulogu voditelja
- **Opis osnovnog tijeka:**
  1. Voditelj odabire karticu "Natjecanja"
  2. Voditelj odabire jedno od svojih natjecanja
  3. Voditelj pokreće natjecanje

#### UC17 - Prijava na natjecanje

- **Glavni sudionik:** Natjecatelj
- **Cilj:** Prijaviti se na nadolazeće natjecanje
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik mora biti prijavljen
- **Opis osnovnog tijeka:**
  1. Natjecatelj odabire karticu "Natjecanja"
  2. Natjecatelj odabire jedno od nadolazećih natjecanja
  3. Natjecatelj se prijavljuje na odabrano natjecanje

#### UC18 - Pristupanje natjecanju

- **Glavni sudionik:** Natjecatelj
- **Cilj:** Pristupiti natjecanju
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik mora biti prijavljen u sustav i na natjecanje koje je trenutno aktivno
- **Opis osnovnog tijeka:**
  1. Natjecatelj odabire karticu "Natjecanja"
  2. Natjecatelj odabire jedno od trenutno aktivnih natjecanja na koje je prijavljen
  3. Natjecatelj pristupa natjecanju

#### UC19 - Odjava s natjecanja

- **Glavni sudionik:** Natjecatelj
- **Cilj:** Odjaviti se s natjecanja
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik mora biti prijavljen u sustav i na natjecanje

- **Opis osnovnog tijeka:**

1. Natjecatelj odabire karticu "Natjecanja"
2. Natjecatelj odabire jedno od nadolazećih natjecanja na koje je prijavljen
3. Natjecatelj se odjavljuje s odabranog natjecanja

#### UC20 - Uređivanje natjecanja

- **Glavni sudionik:** Voditelj

- **Cilj:** Urediti natjecanje

- **Sudionici:** Baza podataka

- **Preduvjet:** Korisnik mora biti prijavljen i imati ulogu voditelja

- **Opis osnovnog tijeka:**

1. Voditelj odabire karticu "Natjecanja"
2. Voditelj odabire jedno od svojih natjecanja
3. Voditelj uređuje pojedinosti svog natjecanja
4. Promjene se upisuju u bazu podataka

- **Opis mogućih odstupanja:**

- 3.a Voditelj unosi podatke u nedozvoljenom formatu

1. Sustav obavještava voditelja o neuspjelom upisu podataka
2. Voditelj popravljajući neispravne unose i uspješno uređuje natjecanje ili odustaje od uređivanja natjecanja

#### UC21 - Pregled sudionika na natjecanju

- **Glavni sudionik:** Korisnik

- **Cilj:** Pregledati natjecatelje koji su sudjelovali na natjecanju

- **Sudionici:** Baza podataka

- **Preduvjet:** -

- **Opis osnovnog tijeka:**

1. Korisnik odabire karticu "Natjecanja"
2. Korisnik odabire jedno od prošlih natjecanja
3. Prikazuje se lista natjecatelja koji su sudjelovali na odabranom natjecanju

#### UC22 - Brisanje natjecanja

- **Glavni sudionik:** Administrator

- **Cilj:** Obrisati postojeće natjecanje

- **Sudionici:** Baza podataka

- **Preduvjet:** Korisnik mora biti prijavljen i imati ulogu administratora
- **Opis osnovnog tijeka:**
  1. Administrator odabire karticu "Natjecanja"
  2. Administrator odabire jedno od natjecanja
  3. Administrator briše odabrano natjecanje
  4. Natjecanje se briše iz baze podataka

#### UC23 - Dohvat učitano g rješenja zadatka s natjecanja

- **Glavni sudionik:** Natjecatelj
- **Cilj:** Dohvatiti učitano rješenje zadatka nekog od natjecatelja koji je sudjelovao na natjecanju
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik mora biti prijavljen i imati potpuno točno riješen konkretan zadatak
- **Opis osnovnog tijeka:**
  1. Natjecatelj odabire karticu "Natjecanja"
  2. Natjecatelj odabire jedno od prošlih natjecanja
  3. Natjecatelj odabire jedan od zadataka s odabranog natjecanja
  4. Natjecatelj dohvaća rješenje zadatka

#### UC24 - Završetak natjecanja

- **Glavni sudionik:** Natjecatelj
- **Cilj:** Završiti s natjecanjem
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik mora biti prijavljen i morao je pristupiti natjecanju
- **Opis osnovnog tijeka:**
  1. Natjecatelj odabire karticu "Natjecanja"
  2. Natjecatelj odabire jedno od trenutno aktivnih natjecanja na koje je prijavljen
  3. Natjecatelj pristupa natjecanju
  4. Nakon rješavanja zadataka, a prije isteka vremena, natjecatelj završava natjecanje

#### UC25 - Pregled ostalih korisnika

- **Glavni sudionik:** Korisnik
- **Cilj:** Pregled profila ostalih korisnika

- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
  1. Korisnik odabire karticu "Korisnici"
  2. Prikazuje se lista svih registriranih natjecatelja i voditelja

#### UC26 - Promjena ovlasti korisniku

- **Glavni sudionik:** Administrator
- **Cilj:** Promjeniti ovlasti korisniku
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik mora biti prijavljen i imati ulogu administratora
- **Opis osnovnog tijeka:**
  1. Administrator odabire karticu "Korisnici"
  2. Administrator odabire korisnika
  3. Administrator mijenja odabranom korisniku ovlasti
  4. Promjene se upisuju u bazu podataka

#### UC27 - Pregled statistike pojedinog korisnika

- **Glavni sudionik:** Korisnik
- **Cilj:** Pregledati statistiku pojedinog korisnika
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
  1. Korisnik odabire karticu "Korisnici"
  2. Korisnik odabire određenog korisnika s liste
  3. Prikazuje se statistika odabranog korisnika

#### UC28 - Pregled svih učitanih rješenja nekog natjecatelja

- **Glavni sudionik:** Natjecatelj
- **Cilj:** Pregledati sva učitana rješenja natjecatelja koji je sudjelovao na natjecanju
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik mora biti prijavljen
- **Opis osnovnog tijeka:**
  1. Natjecatelj odabire karticu "Natjecanja"
  2. Natjecatelj odabire jedno od prošlih natjecanja

3. Natjecatelj odabire jednog od natjecatelja koji je sudjelovao na natjecanju
4. Prikazuju se sva učitana rješenja odabranog natjecatelja

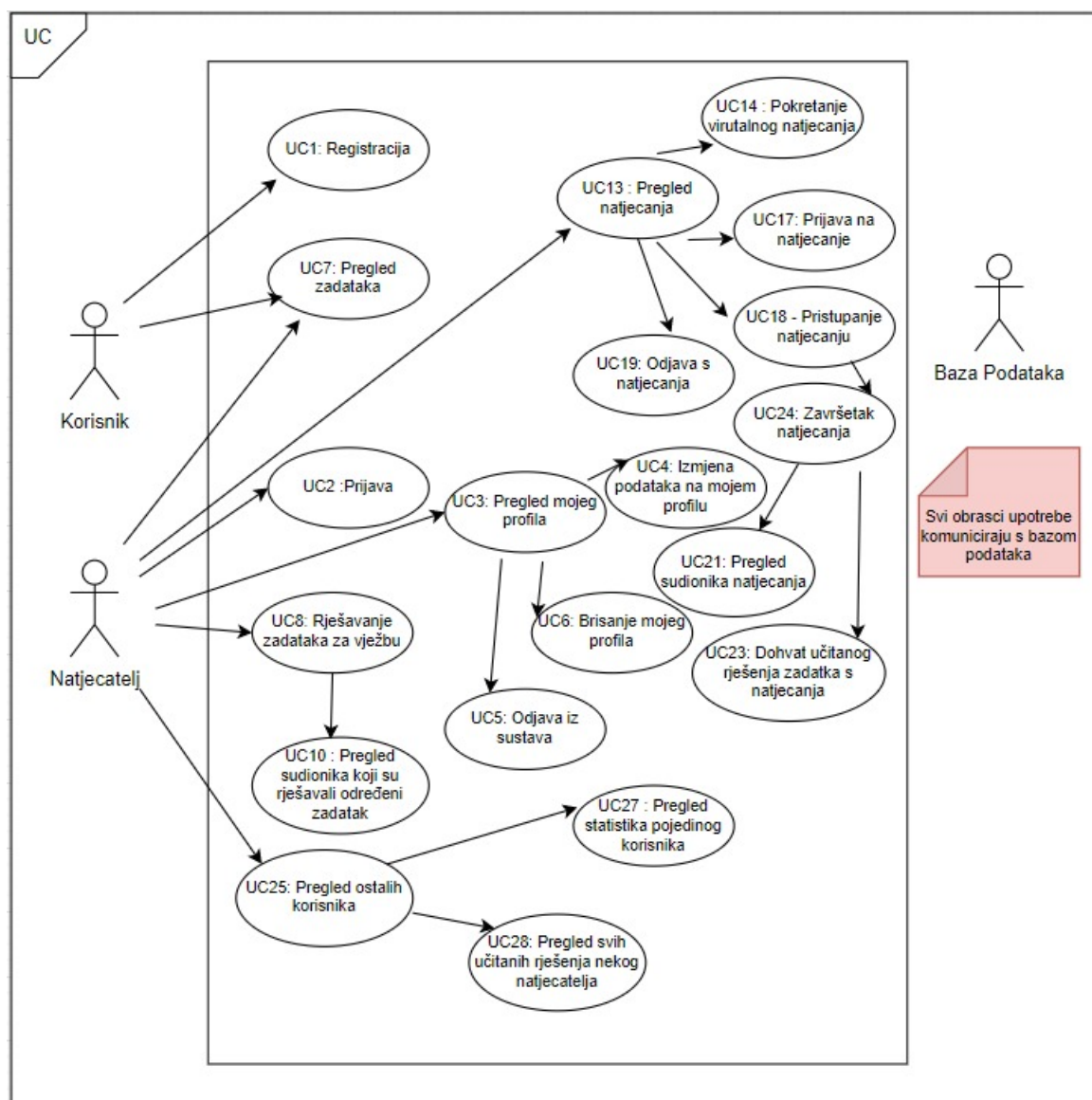
#### UC29 - Promjena osobnih podataka korisniku

- **Glavni sudionik:** Administrator
- **Cilj:** Promijeniti osobne podatke korisniku
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik mora biti prijavljen i imati ulogu administratora
- **Opis osnovnog tijeka:**
  1. Administrator odabire karticu "Korisnici"
  2. Administrator odabire određenog korisnika
  3. Administrator mijenja željene osobne podatke korisniku
  4. Promjene se upisuju u bazu podataka
- **Opis mogućih odstupanja:**
  - 3.a Voditelj unosi podatke u nedozvoljenom formatu
    1. Sustav obavještava administratora o neuspjelom upisu podataka
    2. Administrator popravljajući neispravne unose i uspješno mijenja osobne podatke odabranog korisnika ili odustaje od promjena

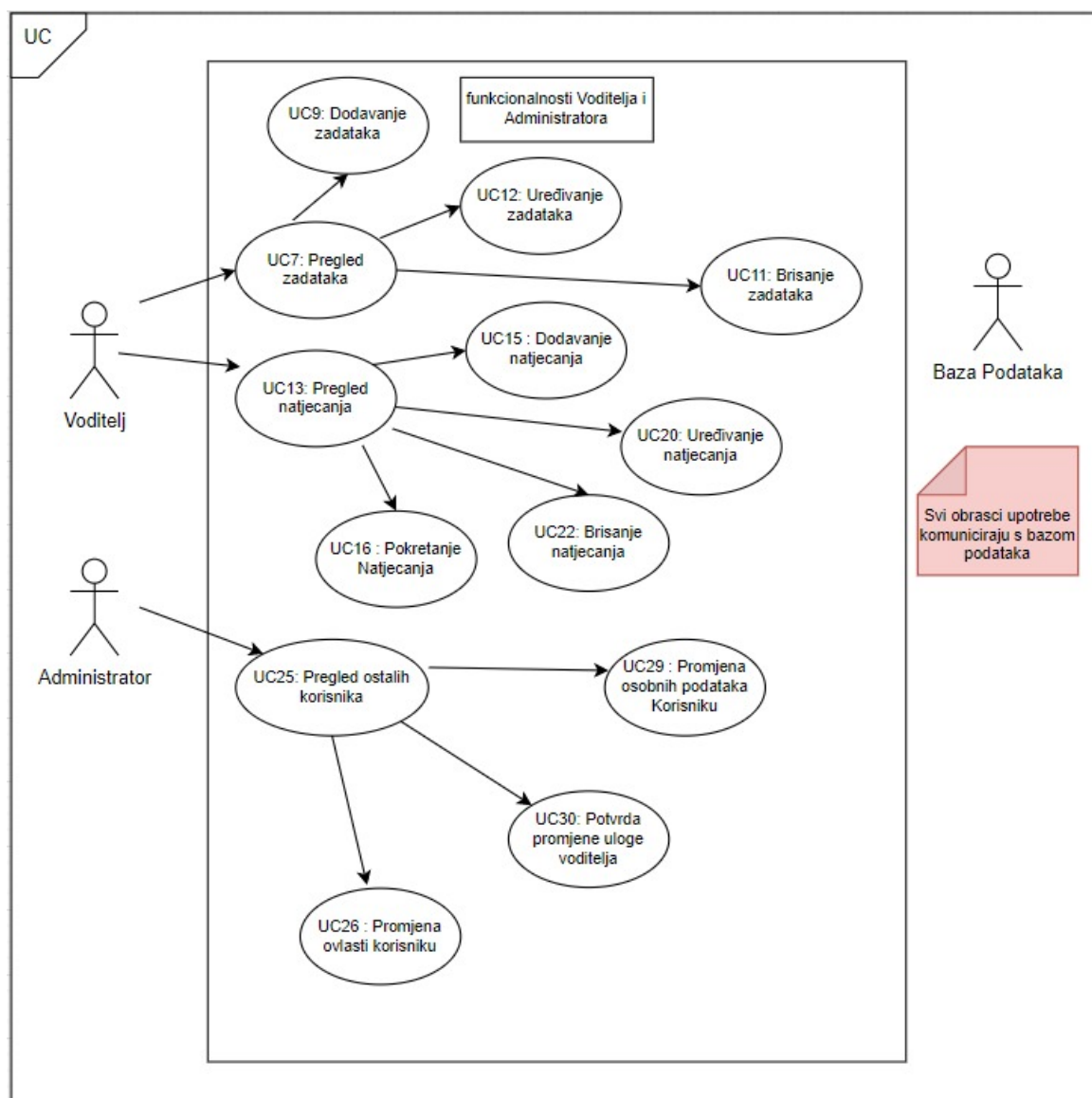
#### UC30 - Potvrda promjene uloge voditelja

- **Glavni sudionik:** Administrator
- **Cilj:** Potvrditi promjenu uloge voditelju
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik mora biti prijavljen i imati ulogu administratora
- **Opis osnovnog tijeka:**
  1. Administrator odabire karticu "Korisnici"
  2. Administrator odabire određenog korisnika
  3. Administrator potvrđuje korisniku promjenu uloge u voditelja natjecanja
  4. Promjene se upisuju u bazu podataka

## Dijagrami obrazaca uporabe



Slika 3.1: Dijagram obrasca uporabe, funkcionalnost korisnika i natjecatelja

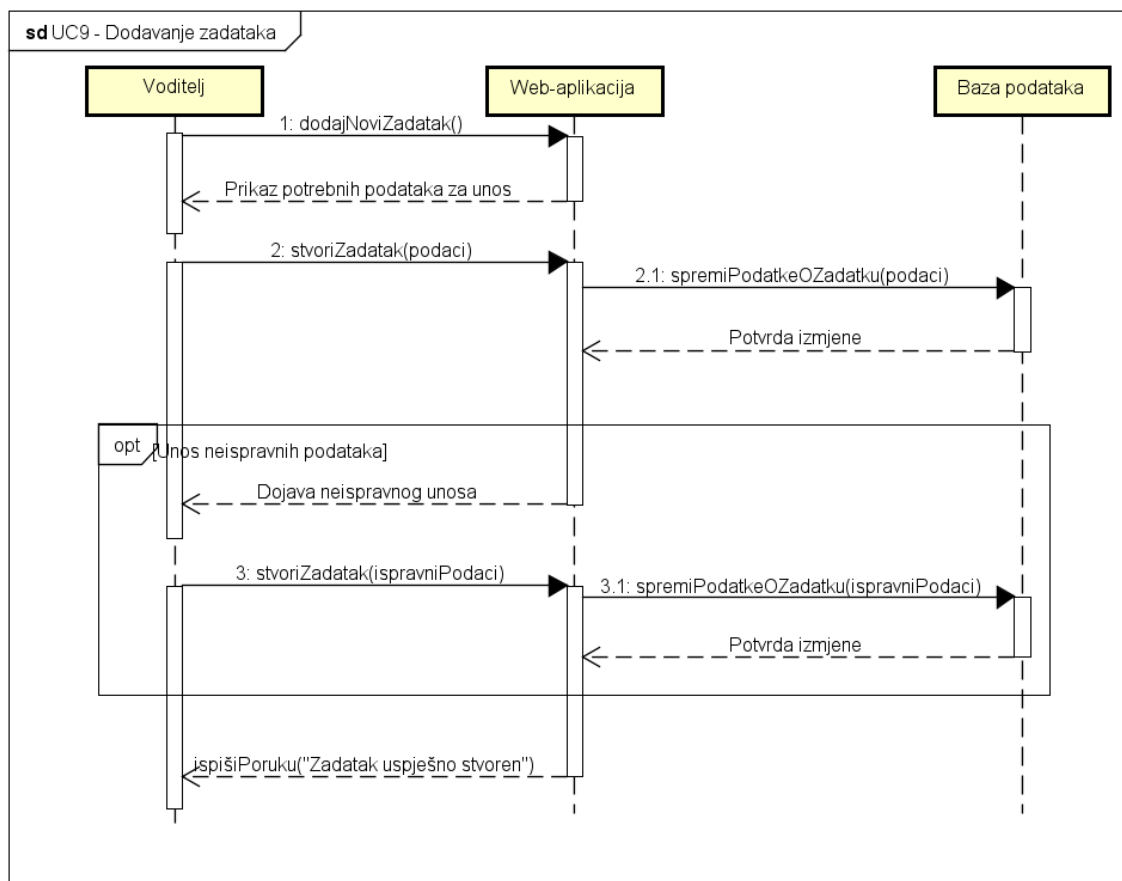


Slika 3.2: Dijagram obrasca uporabe, funkcionalnost voditelja i administratora

### 3.1.2 Sekvencijski dijagrami

#### Obrazac uporabe UC9 - Dodavanje zadatka

Voditelj šalje zahtjev za dodavanjem novog zadatka te mu se prikazuje stranica s poljima koja treba popuniti. Ta polja su naziv zadatka, broj bodova (tj. težina 1-5), vremensko ograničenje izvršavanja programa, tekst zadatka, primjeri za evaluaciju (ulaz i očekivani izlaz programa) i privatnost zadatka. Voditelj upisuje tražene podatke i sprema promjene. Ukoliko su svi upisani podaci ispravno napisani, voditelj dobiva potvrdu u obliku poruke da je uspješno stvorio zadatak, a ako su upisani podaci na neki način neispravni, voditelju se dojavljuje greška i nudi da ispravi podatke.

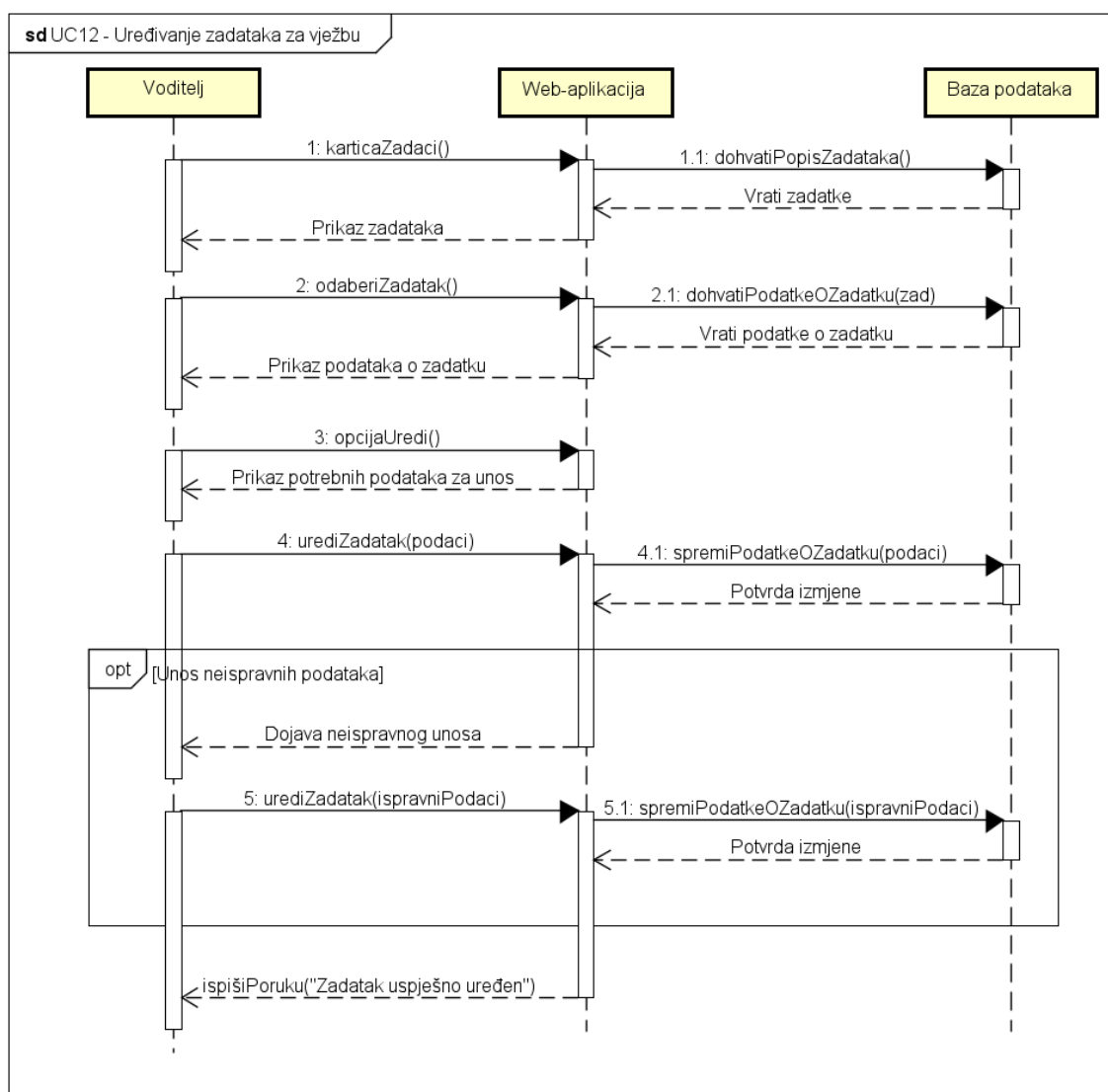


Slika 3.3: Sekvencijski dijagram za UC9



## Obrazac uporabe UC12 - Uređivanje zadataka za vježbu

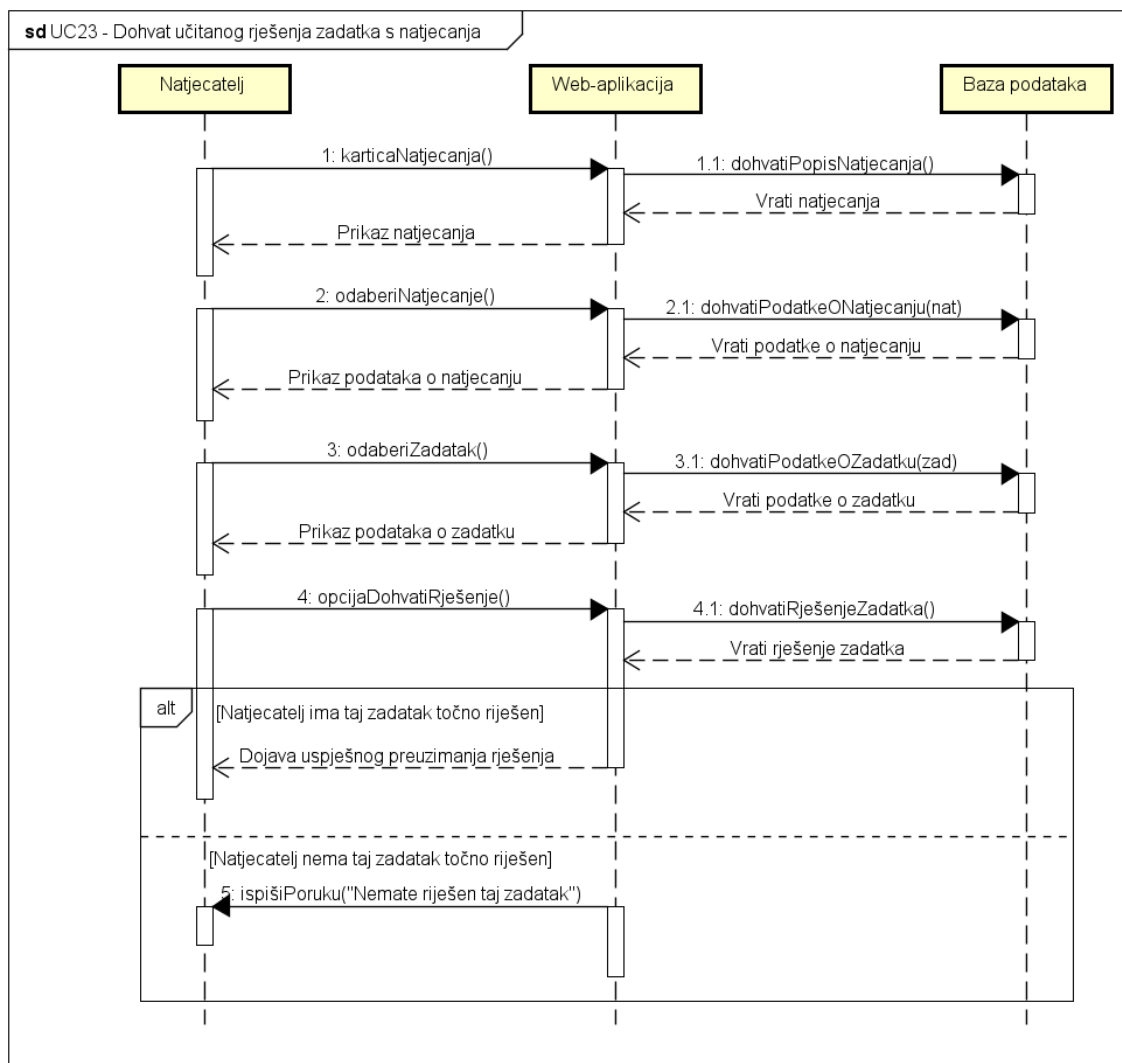
Voditelj najprije odabire karticu sa zadacima te mu se prikazuje popis zadataka. Zatim odabire određeni zadatak s popisa i dobiva sve informacije o njemu. Zatim odabire opciju za uređivanje zadataka i, slično kao i u prethodnom dijagramu, prikazuje mu se stranica s istim poljima koja treba popuniti. Voditelj uređuje željene podatke, sprema promjene i, ako su uređeni podaci ispravni, dobiva potvrdu o uspješnom uređivanju zadatka, inače dobiva poruku greške i nudi mu se da ispravi podatke.



Slika 3.4: Sekvencijski dijagram za UC12

### Obrazac uporabe UC23 - Dohvat učitano g rješenja zadatka s natjecanja

Natjecatelj najprije odabire karticu s natjecanjima i dobiva popis natjecanja. Nakon toga bira željeno natjecanje i dobiva sve informacije o njemu. Zatim odabire zadatak čije rješenje želi dohvatiti i prikazuju mu se podaci o zadatku. Na ovoj stranici odabire opciju za dohvat rješenja i prima potvrdu o njegovom uspješnom preuzimanju samo ako je natjecatelj već riješio taj zadatak, i to potpuno točno, a inače dobiva poruku da ne može preuzeti rješenje tog zadatka.



Slika 3.5: Sekvencijski dijagram za UC23

## 3.2 Ostali zahtjevi

- Sustav treba omogućiti rad više korisnika u stvarnom vremenu
- Korisničko sučelje i sustav moraju podržavati hrvatsku abecedu (dijakritičke znakove) pri unosu i prikazu tekstualnog sadržaja
- Izvršavanje dijela programa u kojem se pristupa bazi podataka ne smije trajati duže od nekoliko sekundi
- Sustav treba biti implementiran kao web aplikacija koristeći objektno-orijentirane jezike
- Neispravno korištenje korisničkog sučelja ne smije narušiti funkcionalnost i rad sustava
- Sustav treba biti jednostavan za korištenje, korisnici se moraju znati koristiti sučeljem bez opširnih uputa
- Sustav mora validirati podatke na klijentskoj i poslužiteljskoj strani
- Veza s bazom podataka mora biti kvalitetno zaštićena, brza i otporna na vanjske greške
- Pristup sustavu mora biti omogućen iz javne mreže pomoću HTTPS.
- Sustav je responzivan na mobilnim uređajima

## 4. Arhitektura i dizajn sustava

Arhitekturu je moguće podijeliti na tri podsustava:

- Web poslužitelj
- Web aplikacija
- Baza podataka

Web preglednik je program koji korisniku omogućuje pregled web-stranica i multimedijskog sadržaja vezanog uz njih. Korisnik putem web preglednika šalje zahtjev web poslužitelju te mu web preglednik kao interpreter prevodi web-stranicu i njezin sadržaj u format koji je korisniku razumljiv.

Web poslužitelj temelj je rada web aplikacije te je njegov glavni zadatak omogućiti komunikaciju klijenta s aplikacijom. Komunikacija se ostvaruje preko protokola HTTP (engl. *Hyper Text Transfer Protocol*), koji služi za prijenos informacija na webu. Web aplikacija se pokreće preko poslužitelja koji joj prosljeđuje zahtjev od strane korisnika.

Web aplikacija služi za obradu korisničkih zahtjeva. Ovisno o zahtjevu, web aplikacija tijekom obrade zahtjeva pristupa bazi podataka te korisniku vraća odgovor u obliku HTML (engl. *HyperText Markup Language*) dokumenta koji se prikazuje preko web preglednika.

Codeshark web aplikacija bazirana je na programskom jeziku Python-u za razvoj *backend*-a, zajedno s JavaScript-om uz biblioteku React za razvoj *frontend*-a. Kao razvojno okruženje koristio se Microsoft Visual Studio.

Arhitektura sustava temelji se na konceptu MVC-a (Model-View-Controller). Karakteristika MVC koncepta je nezavisan razvoj pojedinih dijelova aplikacije što za posljedicu ima jednostavnije ispitivanje kao i jednostavno razvijanje i dodavanje novih svojstava u sustav.

MVC koncept sastoji se od:

- **Model** - Središnja komponenta sustava. Predstavlja dinamičke strukture podataka, neovisne o korisničkom sučelju. Izravno upravlja podacima, logikom i pravilima aplikacije. Ujedno i prima ulazne podatke od Controller-a.

- **View** - Bilo kakav prikaz podataka, poput grafa. Mogući su različiti prikazi iste informacije poput grafičkog ili tabličnog prikaza podataka.
- **Controller** - Prima ulaze i prilagođava ih za prosljeđivanje Model-u ili View-u. Upravlja korisničkim zahtjevima i temeljem njih izvodi daljnju interakciju s ostalim elementima sustava.

## 4.1 Baza podataka

Za potrebe našeg sustava koristit ćemo relacijsku bazu podataka koja svojom strukturom olakšava baratanju potrebnih podataka. Gradivna jedinka baze je relacija, odnosno tablica koja je definirana svojim imenom i skupom atributa. Zadaća baze podataka je brza i jednostavna pohrana, izmjena i dohvat podataka za daljnju obradu. Baza podataka ove aplikacije sastoji se od sljedećih entiteta:

- Korisnik
- Natjecanje
- Zadatak
- Test-Primjeri
- Upload-Rjesenja
- Virtualno-Natjecanje
- Trofej
- Sudjeluje-Na
- Je-Osvojio
- Klasa-Natjecanja
- Sesija

### 4.1.1 Opis Tablica

**Korisnik** Ovaj entitet sadržava sve važne informacije o korisniku aplikacije. Sadrži attribute: Korisnikov ID, Korisničko ime, lozinku, ime, prezime, sliku profila, email, titulu, nivou prava, token, vrijeme generacije tokena i stanje aktivacije korisnika. Ovaj entitet u vezi je One-to-Many s entitetom Natjecanje preko ID-a korisnika (AutorID), u vezi One-to-Many s entitetom Zadatak preko ID-a korisnika (Auto-rID), u vezi One-to-Many s entitetom Sesija preko ID-a korisnika (KorisnikId), u vezi Many-to-Many s Je-Osvojio preko ID-a korisnika, u vezi Many-to-Many s Sudjeluje-na preko ID-a korisnika, u vezi One-to-Many s entitetom Upload-Rjesenja

preko ID-a korisnika te u vezi One-to-Many s entitetom VirtNatjecanje preko ID-a korisnika.

Korisnik		
KorisnikId	SERIAL	jedinstveni indikator korisnika
KorisnickoIme	VARCHAR	identificirajuće ime korisnika
Lozinka	VARCHAR	hash lozinke
SlikaProfila	VARCHAR	slika profila korisnika
Ime	VARCHAR	ime korisnika
Prezime	VARCHAR	prezime korisnika
Email	VARCHAR	email korisnika
Titula	VARCHAR	prilagođena titula korisnika
NivouPrava	INT	razina ovlasti korisnika
Token	VARCHAR	token napravljen za korisnika
Token Generiran	TIMESTAMP	vrijeme kada je token generiran
Aktivan	BOOLEAN	stanje verifikacije korisnika

**Natjecanje** Ovaj entitet sadržava sve važne informacije o održavanju natjecanja. Sadrži attribute: ID Natjecanja, ime natjecanja, tekst natjecanja, vrijeme kraja natjecanja, vrijeme početka natjecanja, sliku trofeja, broj zadataka, ID autora natjecanja, ID klase natjecanja, ID trofeja. Ovaj entitet u vezi je Many-to-One s entitetom Korisnik preko ID-a korisnika (AutorId), u vezi One-to-Many s entitetom Zadatak preko ID-a natjecanja, u vezi One-to-Many s VirtNatjecanje preko ID-a natjecanja, u vezi One-to-One s entitetom Trofej preko ID-a trofeja te u vezi Many-to-One s entitetom IDKlasaNatjecanja preko ID-a klase natjecanja.

Natjecanje		
NatjecanjeId	SERIAL	jedinstveni indikator natjecanja
ImeNatjecanja	VARCHAR	identificirajuće ime natjecanja

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Natjecanje		
Tekst Natjecanja	VARCHAR	sadržaj teksta natjecanja
VrijemeKraj	TIMESTAMP	vrijeme završetka natjecanja
VrijemePoc	TIMESTAMP	vrijeme početka natjecanje
SlikaTrofeja	VARCHAR	sličica trofeja
BrojZadataka	INT	broj zadataka u natjecanju
Slug	VARCHAR	slug verzija imena natjecanja
AutorId	INT	autorov Korisnik ID (korisnik.KorisnikID)
ID Klase Natjecanja	INT	ID Klase Natjecanja (klasanatjecanja.IdKlaseNatjecanja)
TrofejID	INT	Trofej ID (trofej.TrofejID)

**Zadatak** Ovaj entitet sadržava sve važne informacije o zadacima. Sadrži attribute: Zadatak ID, ime zadatka, tekst zadatka, bodovi zadatka, maksimalno vrijeme izvršavanja zadatka, privatnost zadatka, slug naziv zadatka, id autora , id natjecanja(opcionalno). Ovaj entitet u vezi je Many-to-One s entitetom Korisnik preko ID-a korisnika (AutorID), u vezi Many-to-One s entitetom Natjecanje preko ID-a natjecanja , u vezi One-to-Many s TestPrimjeri preko ID-a zadatka te u vezi One-to-Many s entitetom UploadRjesenja preko ID-a zadatka.

Zadatak		
ZadatakId	SERIAL	jedinstveni indikator zadatka
ImeZadatka	VARCHAR	identificirajući naziv zadatka
Bodovi	INT	broj bodova(težina 1-5)
MaxVrijeme Izvrs	NUMERIC	maksimalno vrijeme izvršavanja programa
TekstZadatka	VARCHAR	sadržaj teksta zadatka
Privatnost	BOOLEAN	stanje privatnosti zadatka

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Zadatak		
Slug	VARCHAR	slug verzija naziva zadatka
AutorId	INT	autorov Korisnik ID (korisnik.KorisnikId)
NatjecanjeId	INT	(opcionarno) Natjecanje ID (natjecanje.NatjecanjeId)

**Test Primjeri** Ovaj entitet sadržava sve važne informacije o testnim primjerima za zasebne zadatke. Sadrži attribute: Zadatak ID, ulaz testa, izlaz testa. Ovaj entitet u vezi je Many-to-One s entitetom Zadatak preko ID-a zadatka.

TestPrimjeri		
Ulaz	VARCHAR	ulaz testa
ZadatakId	INT	Zadatak ID (zadatak.ZadatakId)
Izlaz	VARCHAR	željeni izlaz testa

**Upload Rjesenja** Ovaj entitet sadržava sve informacije koje su bitne oko uploada rješenja na zadatak. Sadrži attribute: Zadatak ID, korisnikov id, vrijeme predaje rješenja, predano rješenje, prolaznost, prosječno vrijeme izvršavanja, aktivnost natjecanja. Ovaj entitet u vezi je Many-to-One s entitetom Korisnik preko ID-a korisnika te u vezi Many-to-One s entitetom Zadatak preko ID-a zadatka.

UploadRjesenja		
VrijemePredaje	TIMESTAMP	vrijeme predaje rješenja
KorisnikId	INT	Korisnik ID (korisnik.KorisnikId)
ZadatakId	INT	Zadatak ID (zadatak.ZadatakId)
Predano Rjesenje	VARCHAR	datoteka koju je predao korisnik
Prolaznost	NUMERIC	posto riješenosti primjera zadataka

Nastavljeno na idućoj stranici



Nastavljeno od prethodne stranice

UploadRjesenja		
ProsjVrijeme Izvrš	NUMERIC	prosječno vrijeme izvršavanja po primjeru

**Virtualno Natjecanje** Ovaj entitet sadržava sve važne informacije o Virtualnim natjecanjima. Sadrži attribute: Korisnikov ID, ID natjecanja, ID random zadatka težine 2, ID random zadatka težine 3, ID random zadatka težine 4 te ID random zadatka težine 5. Ovaj entitet u vezi je Many-to-One s entitetom Korisnik preko ID-a korisnika, u vezi Many-to-One s entitetom Natjecanje preko ID-a natjecanja te u 4 veze Many-to-One s entitetom Zadatak preko ID-ova zadataka.

VirtNatjecanje		
VirtNatjecanje Id	SERIAL	identifikacijski broj virtualnog natjecanja
Vrijeme Kreacije	TIMESTAMP	vrijeme kreacije virtualnog natjecanja
Zadaci	INTEGER[]	(Opcionalno) array svih random odaberenih zadataka
NatjecanjeId	INT	(Opcionalno) Natjecanje ID (natjecanje.NatjecanjeId)
KorisnikId	INT	Korisnik ID (korisnik.KorisnikId)

**Trofej** Ovaj entitet sadržava sve važne informacije o trofejima. Sadrži attribute: Trofej ID, ime trofeja, slika trofeja. Ovaj entitet u vezi je Many-to-One s entitetom Korisnik preko ID-a korisnika, u vezi One-to-Many s entitetom Natjecanje preko ID-a trofeja te u vezi Many-to-Many s Je-Osvojio preko ID-a trofeja.

Trofej		
TrofejID	SERIAL	jedinstveni indikator trofeja
ImeTrofeja	VARCHAR	ime trofeja

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Trofej		
SlikaTrofeja	VARCHAR	slika trofeja

**Sudjeluje Na** Ovaj entitet sadržavi listu svih sudjelovanja korisnika na natjecanjima. Sadrži attribute: Korisnik ID, Natjecanje ID.

SudjelujeNa		
KorisnikId	INT	Korisnik ID (korisnik.KorisnikId)
NatjecanjeId	INT	Natjecanje ID (natjecanje.NatjecanjeId)

**Je Osvojio** Ovaj entitet sadržava listu svih korisnika s osvojenim peharima. Sadrži attribute: Korisnik ID, Trofej ID. Ovaj entitet u vezi je One-To-Many s entitetom Natjecanje preko ID-a klase natjecanja.

JeOsvojio		
KorisnikId	INT	Korisnik ID (korisnik.KorisnikId)
TrofejId	INT	Trofej ID (trofej.TrofejId)

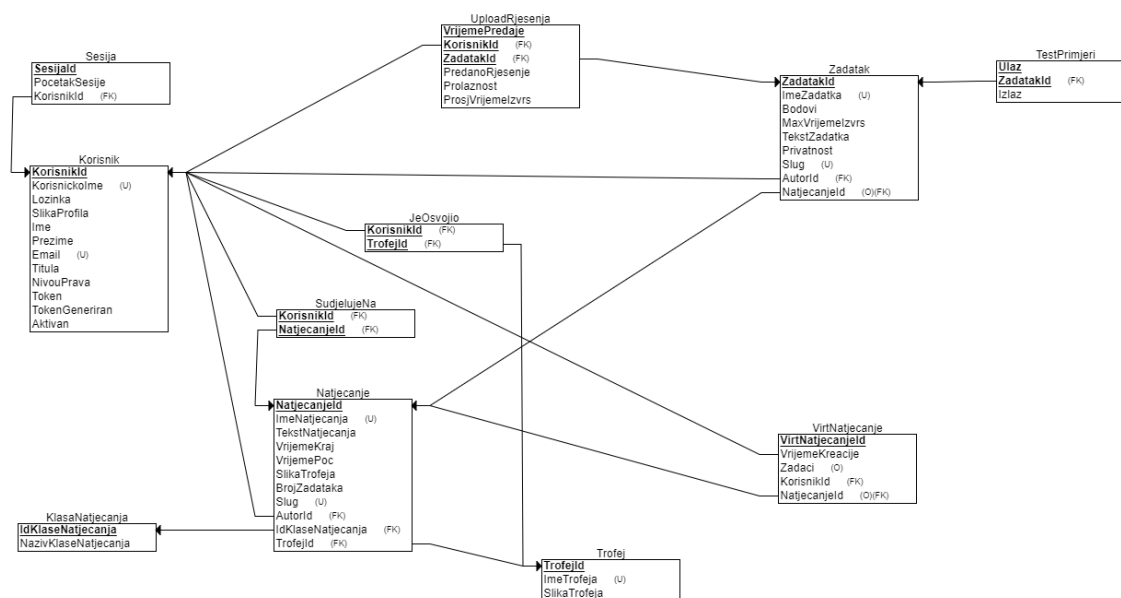
**Klasa Natjecanja** Ovaj entitet sadržava sve informacije o klasi natjecanja. Sadrži attribute: ID klase natjecanja, naziv klase natjecanja.

KlasaNatjecanja		
IdKlase Natjecanja	INT	jedinstveni indikator klase natjecanje
NazivKlase Natjecanja	VARCHAR	naziv klase natjecanja

**Sesija** Ovaj entitet sadržava sve informacije o korisničkoj sesiji. Sadrži attribute: Sesija ID, Pocetak sesije, korisnikov ID.

Sesija		
SesijaId	VARCHAR	jedinstveni indikator sesije
PocetakSesije	TIMESTAMP	timestamp pocetka sesije
KorisnikId	INT	id vlasnika sesije (Korisnik.KorisnikId)

### 4.1.2 Dijagram baze podataka

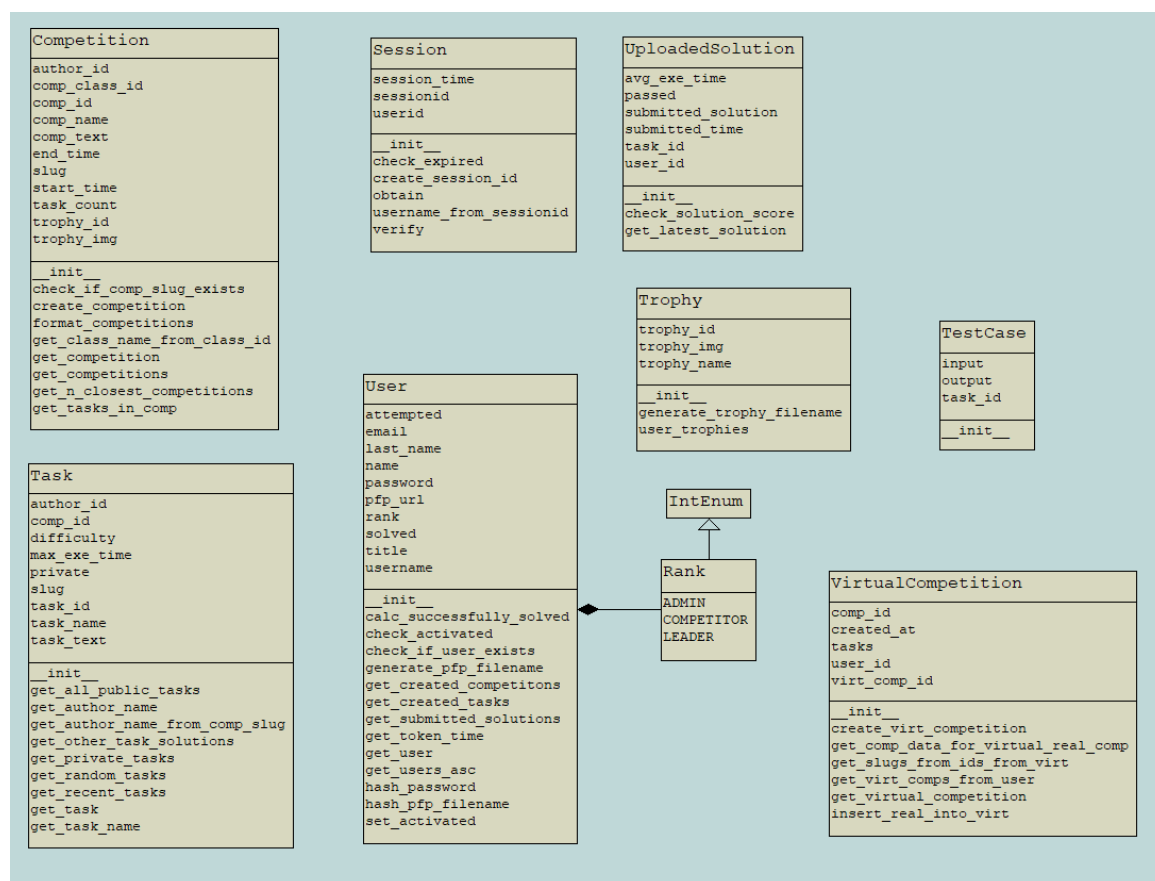


Slika 4.1: E-R dijagram baze podataka

## 4.2 Dijagram razreda

Na slici 4.2. vidimo osnovni model UML dijagrama koji prikazuje osnovne klase koje koristimo na backendu.

Trenutni dijagram razreda opisuje izgled klasa i njihovih metoda koje će nam biti potrebne u projektu. Trenutno pošto imamo registraciju, ulogiravanje i validaciju, koristimo samo klasu Korisnik te njene metode. Imamo privatnu metodu koja se zove `__get_id()` preko koje dobivamo korisnikid iz baze podataka, zato što je to metoda koja dohvaća privatni ključ te ne želimo da korisnik ikad ima pristup tome. Tu metodu trenutno ne koristimo previše već će biti jako korisna kasnije u konekcijama s instancama ostalih klasa prilikom dobivanja informacija iz ostalih tablica baze podataka. Kao što se vidi, svaka klasa ima konstruktor `__init__()` te primaju informacije koje su identične onima iz njihovih modela iz baze podataka.



Slika 4.2: Dijagram razreda

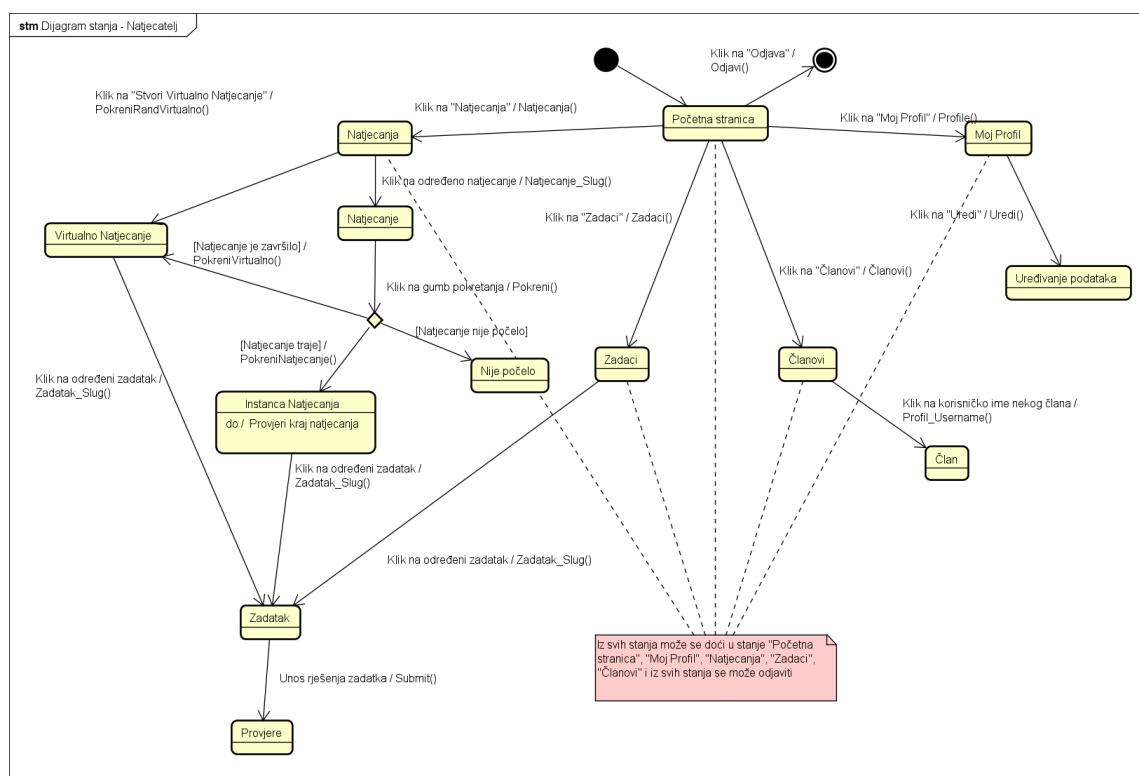
Budući da radimo u programskom jeziku Python, ne možemo lako prikazati

Data Transfer objects. Međutim, prilikom registracije, sve informacije koje dobivamo s frontenda prosljeđujemo u konstruktor Korisnik. Kasnije, prilikom login-a, preko korisničkog imena dobivamo podatke iz baze te izrađujemo instancu Korisnik. S tim Korisnikom možemo baratati pomoću njegovih podataka. Tražimo iz baze podataka je li aktiviran te odgovara li hash unesene lozinke hashu lozinke koja je spremljena u bazu podataka, tj. u instanci Korisnik.

Naša klasa Korisnik se zapravo referira i na voditelja i na administratora. Oni svi imaju identične podatke te duplikacija koda nije potrebna. Velika razlika je samo u njihovom podatku NivouPrava o kojem će ovisiti kakve ovlasti na stranici ima. To će također biti funkcija koju će biti privatna. Voditelj će za razliku od natjecatelja imati pristup stranicama za kreiranje zadataka te stvaranju natjecanja. Administrator će uz to sve moći uređivati ovlasti svih korisnika te će pomoću toga imati mogućnost potvrditi voditelja što će biti na posebnoj stranici.

## 4.3 Dijagram stanja

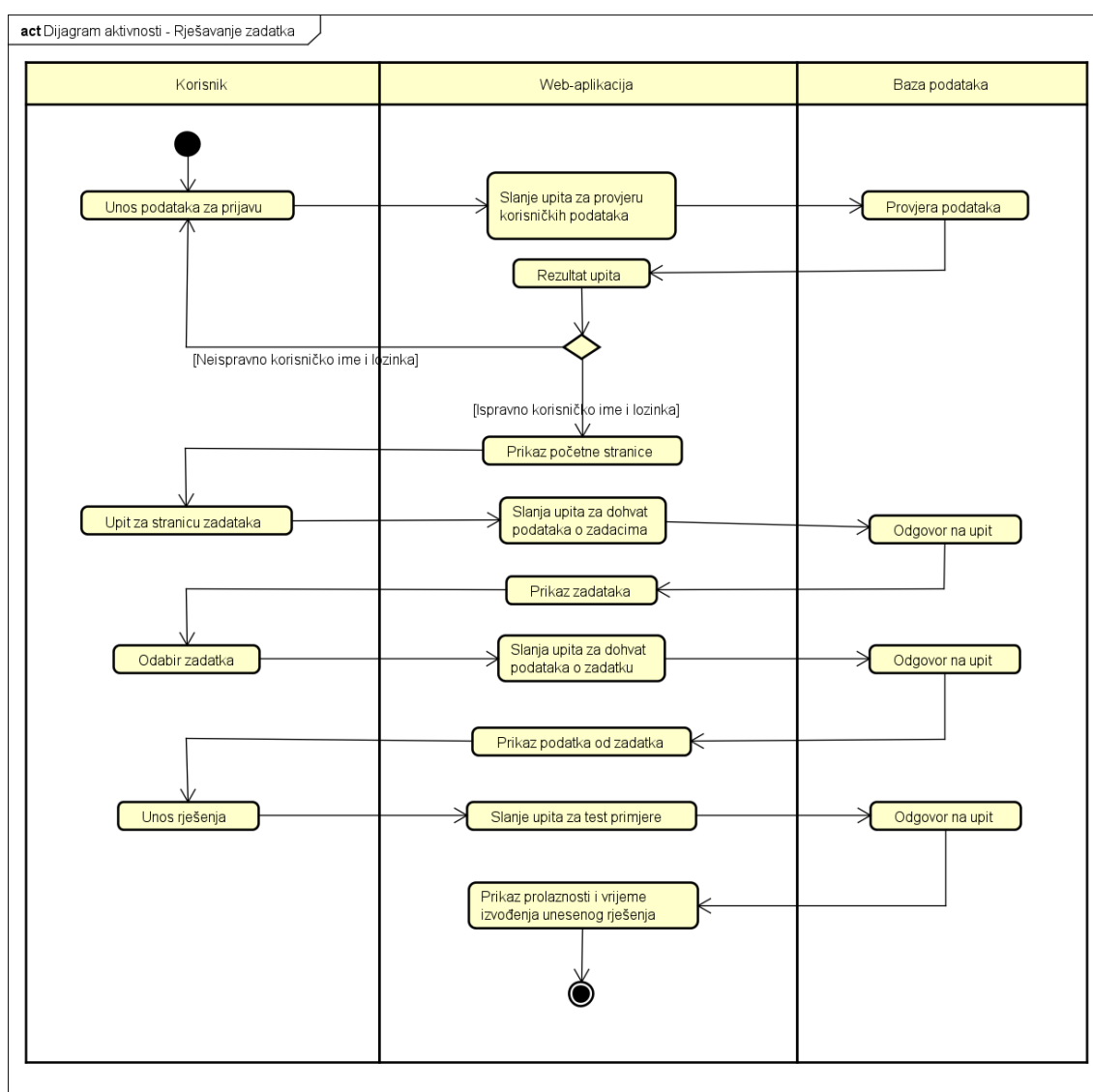
Dijagram stanja prikazuje stanja objekta te prijelaze iz jednog stanja u drugo temeljene na događajima. Na slici 4.3 prikazan je dijagram stanja za registriranog natjecatelja. Nakon prijave, natjecatelju se prikazuje početna stranica na kojoj može otići na stranicu zadataka. Klikom na odabrani zadatak on unosi svoj programski kod koji se šalje na provjeru te dobiva rezultate. Također, može otići na stranicu natjecanja gdje mi se nudi sva bivša, trenutna i buduća natjecanja. Klikom na natjecanje ga vodi na stranicu tog natjecanje gdje ga ima opciju pokrenuti. Ovisno o statusu trajanja natjecanje, pokreće se različita instanca natjecanja. Na toj instanci se prikazuju zadaci koje je moguće riješiti u ograničenom vremenu. Uz stranice zadataka i natjecanja, natjecatelj uvijek može pristupiti stranici vlastitog profila na kojoj može uređivati svoje podatke, te stranici ostalih članova s koje može gledati profile ostalih korisnika.



Slika 4.3: Dijagram stanja

## 4.4 Dijagram aktivnosti

Dijagram aktivnosti primjenjuje se za opis modela toka upravljanja ili toka podataka. Ne upotrebljava se za modeliranje događajima poticanog ponašanja. U modeliranju toka upravljanja svaki novi korak poduzima se nakon završenog prethodnog, a naglasak je na jednostavnosti. Na dijagramu aktivnosti 4.4 prikazan je proces rješavanja zadatka. Korisnik se prijavi u sustav, odabere stranicu sa zadacima te odabere željeni zadatak. Prikažu mu se podaci od odabranog zadatka te prostor za unos rješenja. Nakon unosa svojeg rješenja prikaže mu se njegova prolaznost i vrijeme izvođenja.

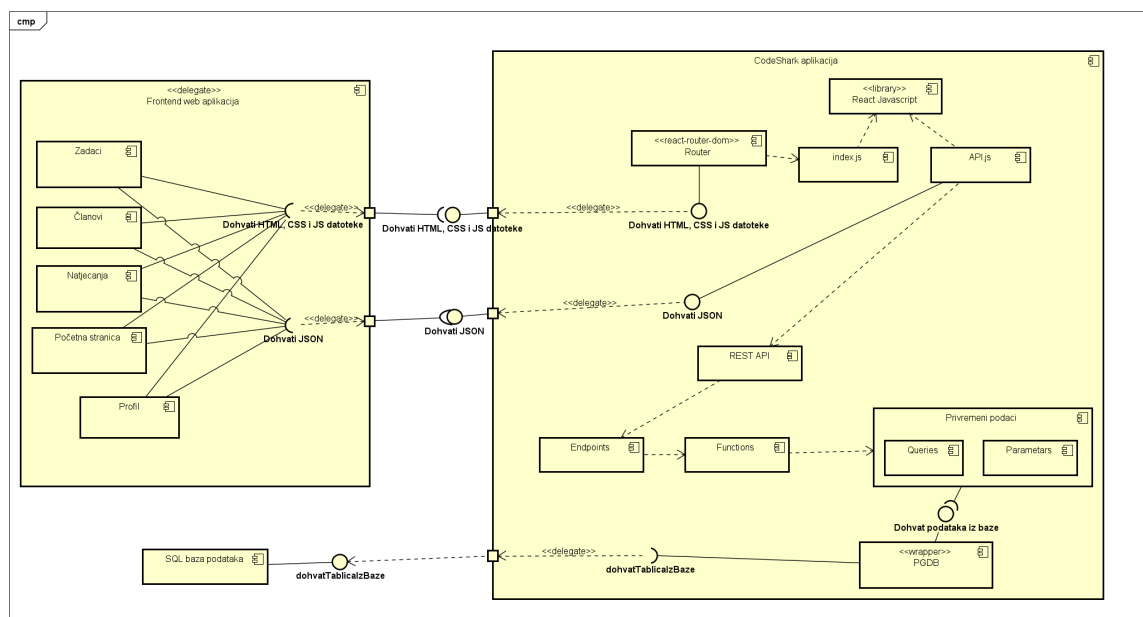


Slika 4.4: Dijagram aktivnosti

## 4.5 Dijagram komponenti

Dijagram komponenti prikazan na slici 4.5 opisuje organizaciju i međuovisnost komponenti, interne strukture i odnose prema okolini. Sustavu se pristupa preko dva različita sučelja. Preko sučelja za dohvat HTML, CSS i JS datoteka poslužuju se datoteke koje pripadaju frontend dijelu aplikacije. Router je komponenta koja na upit prema specifičnom URL-u određuje koje komponente će se poslužiti na sučelje. Frontend aplikacije sastoji se od detaljno rastavljenih komponenata koje su hijerarhijski organizirane kako bi omogućile jednostavno ponovno iskorištavanje postojećih komponenti te čitljivost koda. Sve JavaScript datoteke ovise o React biblioteci koju koriste za interakciju sa korisničkim preglednikom

Preko REST API se šalju zahtjevi na endpointove, koji pozivaju posebne klasne funkcije. Većina tih funkcija dohvaća ili šalje SQL upite. Posebni wrapper PGDB dobiva od funkcija kombinaciju querya i podataka, priprema ih i šalje zahtjev u SQL bazu podataka. Od baze dobiva podatke, PGDB wrapper ih otpakirava iz tuple-ova i šalje nazad funkcijama koje ih vraćaju REST API-u.



Slika 4.5: Dijagram komponenti



## 5. Implementacija i korisničko sučelje

### 5.1 Korištene tehnologije i alati

Komunikacija u timu realizirana je korištenjem aplikacija [WhatsApp](https://www.whatsapp.com/)<sup>1</sup> i [Discord](https://discord.com/)<sup>2</sup>. Za izradu UML dijagrama korišten je alat [Astah UML](https://astah.net/products/astah-uml/)<sup>3</sup>, a kao sustav za upravljanje izvornim kodom [Git](https://git-scm.com/)<sup>4</sup>. Udaljeni repozitorij projekta je dostupan na web platformi [GitLab](https://gitlab.com/)<sup>5</sup>. Kao razvojno okruženje korišten je [Visual Studio Code](https://code.visualstudio.com/)<sup>6</sup>. Prvenstveno se koristi kao uređivač teksta pri razvoju računalnih programa za operacijski sustav Windows, kao i za web-stranice, web-aplikacije, web-usluge i mobilne aplikacije. Aplikacija je napisana koristeći radni okvir [Flask](https://flask.palletsprojects.com/en/2.0.x/)<sup>7</sup> i web poslužitelj [Gunicorn](https://gunicorn.org/)<sup>8</sup> u jeziku [Python 3.8](https://www.python.org/)<sup>9</sup> za izradu backenda. Za izradu frontenda su se koristili razvojni okvir [React](https://reactjs.org/)<sup>10</sup> i jezik [JavaScript](https://www.javascript.com/)<sup>11</sup>. React, također poznat kao React.js ili ReactJS, je biblioteka u JavaScriptu za izgradnju korisničkih sučelja. React se najčešće koristi kao osnova u razvoju web ili mobilnih aplikacija. Složene aplikacije u Reactu obično zahtijevaju korištenje dodatnih biblioteka za interakciju s API-jem. Web poslužitelj Gunicorn nadograđuje radni okvir Flask kako bi mogao posluživati više zahtjeva u isto vrijeme. Naša cijela infrastruktura se nalazi u oblaku na privatnom VPS poslužitelju, poslužuje zahtjeve pomoću [NGINX](https://nginx.org/en/)<sup>12</sup> web poslužitelja na linux ubuntu sustavu. Mrežna potpora našoj arhitekturi je [CloudFlare](https://www.cloudflare.com/)<sup>13</sup> koja nam nudi usluge poput predmemorije (engl. cache) i DDoS zaštite. Bitan dio naše infrastrukture je baza podataka realizirana kroz [PostgreSQL RDBMS](https://www.postgresql.org/)<sup>14</sup>.

---

<sup>1</sup><https://www.whatsapp.com/>

<sup>2</sup><https://discord.com/>

<sup>3</sup><https://astah.net/products/astah-uml/>

<sup>4</sup><https://git-scm.com/>

<sup>5</sup><https://gitlab.com/>

<sup>6</sup><https://code.visualstudio.com/>

<sup>7</sup><https://flask.palletsprojects.com/en/2.0.x/>

<sup>8</sup><https://gunicorn.org/>

<sup>9</sup><https://www.python.org/>

<sup>10</sup><https://reactjs.org/>

<sup>11</sup><https://www.javascript.com/>

<sup>12</sup><https://nginx.org/en/>

<sup>13</sup><https://www.cloudflare.com/>

<sup>14</sup><https://www.postgresql.org/>

## 5.2 Ispitivanje programskog rješenja

Svi testovi izvršeni su pomoću skripte u Pythonu koristeći Selenium WebDriver i pytest. Ispitivanje se radilo po obrascima uporabe kako bi se provjerila osnovna funkcionalnost sustava, ali i nasumičnim kretanjima po aplikaciji kako bi se pronašle neočekivane greške("bugovi") ili nepredviđena ponašanja. Svaki dio sustava je ispitivan, no zbog jednostavnosti u dokumentaciji će biti prikazan samo dio ispitivanja.

### 5.2.1 Ispitivanje komponenti

#### Ispitni slučajevi:

Provedeno je ispitivanje jedinca za 6 ispitnih slučajeva. Ti su slučajevi redom:

1. Konekcija na bazu podataka
2. Provjera postojanja korisnika
3. Dohvaćanje imena klase natjecanja
4. Dohvaćanje zadatka
5. Dohvaćanje imena zadatka
6. Dohvaćanje virtualnog natjecanja

#### Očekivani rezultati:

1. Uspješna spajanje s bazom podataka
2. Uspješno/Neuspješno postojanje korisnika ovisno o parametrima
3. Uspješno dohvaćanje imena klase natjecanja
4. Uspješno dohvaćanje zadatka
5. Uspješno dohvaćanje punog imena zadatka
6. Uspješno dohvaćanje virtualnog natjecanja

```

import pytest
import psycopg2.extensions

from codeshark_backend import connect_to_db
from classes import User, Competition, Task, VirtualCompetition

@pytest.mark.parametrize('expected_type', [psycopg2.extensions.connection])
def test_db_connection(expected_type):
    conn, _ = connect_to_db()
    assert isinstance(conn, expected_type)
    conn.close()

@pytest.mark.parametrize('username, email, expected_result',
    [('obicansmrtnik1', 'obicansmrtnik@ffzg.hr', True),
     ('incorrect_username', 'incorrect_username@fer.hr', False)])
def test_check_if_user_exists(username, email, expected_result):
    existence = User.check_if_user_exists(username, email)
    assert existence[0] == expected_result

@pytest.mark.parametrize('class_id, expected_result', [(1, 'amater'), (2, 'professional')])
def test_get_class_name(class_id, expected_result):
    name = Competition.get_class_name_from_class_id(class_id)
    # This shouldn't be done when comparing with None (should be "is None")
    # but to simplify this example
    assert name == expected_result

@pytest.mark.parametrize('slug, expected_result_type', [('area-of-a-sphere', Task), ('wrong-slug',
type(None))])
def test_get_task(slug, expected_result_type):
    task, _ = Task.get_task(slug)
    # we can't use isinstance in this case as it can't compare when task is None
    assert type(task) == expected_result_type

@pytest.mark.parametrize('id, expected_task_name', [(8, 'Area of a sphere')])
def test_get_task_name(id, expected_task_name):
    task_name = Task.get_task_name(id)
    assert task_name == expected_task_name

@pytest.mark.parametrize('virt_id, expected_result_type', [(31, VirtualCompetition)])
def test_get_virtual_competition(virt_id, expected_result_type):
    virt_comp = VirtualCompetition.get_virtual_competition(virt_id)
    assert isinstance(virt_comp[0], expected_result_type)

```

Slika 5.1: Ispitivanje komponenti

```

PS \codeshark-backend\IzvorniKod> pytest .\pytest_unit_testing.py
===== test session starts =====
platform win32 -- Python 3.9.4, pytest-6.2.5, py-1.11.0, pluggy-1.0.0
rootdir: \codeshark-backend\IzvorniKod
collected 9 items

pytest_unit_testing.py ..... [100%]

===== 9 passed in 9.04s =====

```

Slika 5.2: Rezultat ispitivanja komponenti

**Rezultat:**

Svi očekivani rezultati su zadovoljeni.

## 5.2.2 Ispitivanje sustava

### Ispitni slučajevi:

Provedeno je ispitivanje jedinca za 6 ispitnih slučajeva. Ti su slučajevi redom:

1. Dohvaćanje početne stranice
2. Dohvaćanje popisa korisnika od strane neregistriranog korisnika
3. Dohvaćanje popisa korisnika od strane registriranog korisnika
4. Testiranje prijave
5. Dohvaćanje profilne stranice od strane neregistriranog korisnika
6. Dohvaćanje profilne stranice od strane registriranog korisnika

### Očekivani rezultati:

1. Stranica je uspješno dohvaćena
2. Stranica preusmjeruje na stranicu prijave
3. Stranica je uspješno dohvaćena
4. Uspješna/Neuspješna prijava ovisno o parametrima
5. Stranica preusmjeruje na stranicu prijave
6. Stranica je uspješno dohvaćena

```
import pytest
from selenium import webdriver
from selenium.webdriver.common.by import By
import time
from selenium.common.exceptions import NoSuchElementException
import warnings

warnings.filterwarnings("ignore", category=DeprecationWarning)
chrome_options = webdriver.ChromeOptions()
chrome_options.add_argument("--incognito")
chrome_options.add_argument("--log-level=3")
driver = webdriver.Chrome(executable_path=r"path_to_chromedriver.exe", options=chrome_options)
base_url = 'https://domefan.club'

@pytest.mark.parametrize('url', [base_url])
def test_home_page(url):
    driver.get(url)
    try:
        time.sleep(1)
        element = driver.find_element(By.CLASS_NAME, 'banner-description')
        time.sleep(1)
        assert True
    except:
        assert False

@pytest.mark.parametrize('url', [f"{base_url}/members"])
def test_memebers_page_not_logged_in(url):
    driver.get(url)
    # test should fail because it can't reach members page if not logged in
    # it should find the the login form
    try:
        time.sleep(1)
        element = driver.find_element(By.NAME, "username")
        time.sleep(1)
        assert True
    except NoSuchElementException:
        assert False

@pytest.mark.parametrize('url', [f"{base_url}/members"])
def test_memebers_page_logged_in(url):
    # first login
    driver.get(f'{base_url}/login')
    time.sleep(1)
    element = driver.find_element(By.NAME, "username")
    element.send_keys('obicansmrtnik1')
    time.sleep(.1)
    element = driver.find_element(By.NAME, "password")
    element.send_keys('smrtnik')
    driver.find_element(By.ID, "login-button").click()
    time.sleep(2)
    driver.find_element(By.CLASS_NAME, "swal2-confirm").click()
    time.sleep(2)

    # try to get members page
    driver.get(url)
    try:
        time.sleep(1)
        element = driver.find_element(By.CLASS_NAME, 'table-dark')
        time.sleep(1)
        assert True
    except NoSuchElementException:
        assert False

    time.sleep(1)
    driver.get(f'{base_url}/logout')
```

Slika 5.3: Ispitivanje sustava prvi dio

```
@pytest.mark.parametrize('username , password, expected_result',
    [('obicansmrtnik1', 'smrtnik', 'pass'),
     ('obicansmrtnik2', 'smrtnik', 'pass'),
     ('obicansmrtnik3', 'smrtnik', 'pass'),
     ('AlphaMale', 'alpha', 'pass'),
     ('SigmaMale', 'sigma', 'pass'),
     ('Ajvar', 'ajvar', 'pass'),
     ('incorrect_username', 'whatever', 'fail'),
     ('obicansmrtnik3', 'wrong_password', 'fail')])
def test_login_page(username, password, expected_result):

    driver.get(f'{base_url}/login')
    time.sleep(1)
    element = driver.find_element(By.NAME, "username")
    element.send_keys(username)
    time.sleep(.1)
    element = driver.find_element(By.NAME, "password")
    element.send_keys(password)
    driver.find_element(By.ID, "login-button").click()
    time.sleep(1)
    text = (driver.find_element(By.XPATH, '/html/body/div[2]/div/h2/p')).text
    time.sleep(1)

    # sometimes the expected result is to fail
    if text == 'Successfully signed in!':
        assert expected_result == 'pass'
    elif text == 'Could not sign in!':
        assert expected_result == 'fail'
    else:
        assert False

    driver.get(f'{base_url}/logout')

@pytest.mark.parametrize('url', [f'{base_url}/profile'])
def test_profile_page_not_logged_in(url):
    driver.get(url)
    # the page should redirect to the login form
    try:
        time.sleep(1)
        element = driver.find_element(By.NAME, "username")
        time.sleep(1)
        assert True
    except NoSuchElementException:
        assert False
```

Slika 5.4: Ispitivanje sustava drugi dio

```
@pytest.mark.parametrize('url', [f"{base_url}/profile"])
def test_profile_page_logged_in(url):
    # first login
    driver.get(f'{base_url}/login')
    time.sleep(1)
    element = driver.find_element(By.NAME, "username")
    element.send_keys('obicansmrtnik1')
    time.sleep(.1)
    element = driver.find_element(By.NAME, "password")
    element.send_keys('smrtnik')
    driver.find_element(By.ID, "login-button").click()
    time.sleep(2)
    driver.find_element(By.CLASS_NAME, "swal2-confirm").click()
    time.sleep(2)

    driver.get(url)
    try:
        time.sleep(1)
        element = driver.find_element(By.ID, 'avatar-profile')
        time.sleep(1)
        assert True
    except NoSuchElementException:
        assert False

    driver.get(f'{base_url}/logout')
```

Slika 5.5: Ispitivanje sustava treći dio

```
PS \codeshark-backend\Izvornikod> pytest .\pytest_sys_testing.py
===== test session starts =====
platform win32 -- Python 3.9.4, pytest-6.2.5, py-1.11.0, pluggy-1.0.0
rootdir: \codeshark-backend\Izvornikod
collecting ...
DevTools listening on ws://127.0.0.1:51901/devtools/browser/
collected 13 items

pytest_sys_testing.py ..... [100%]

===== 13 passed in 95.42s (0:01:35) =====
```

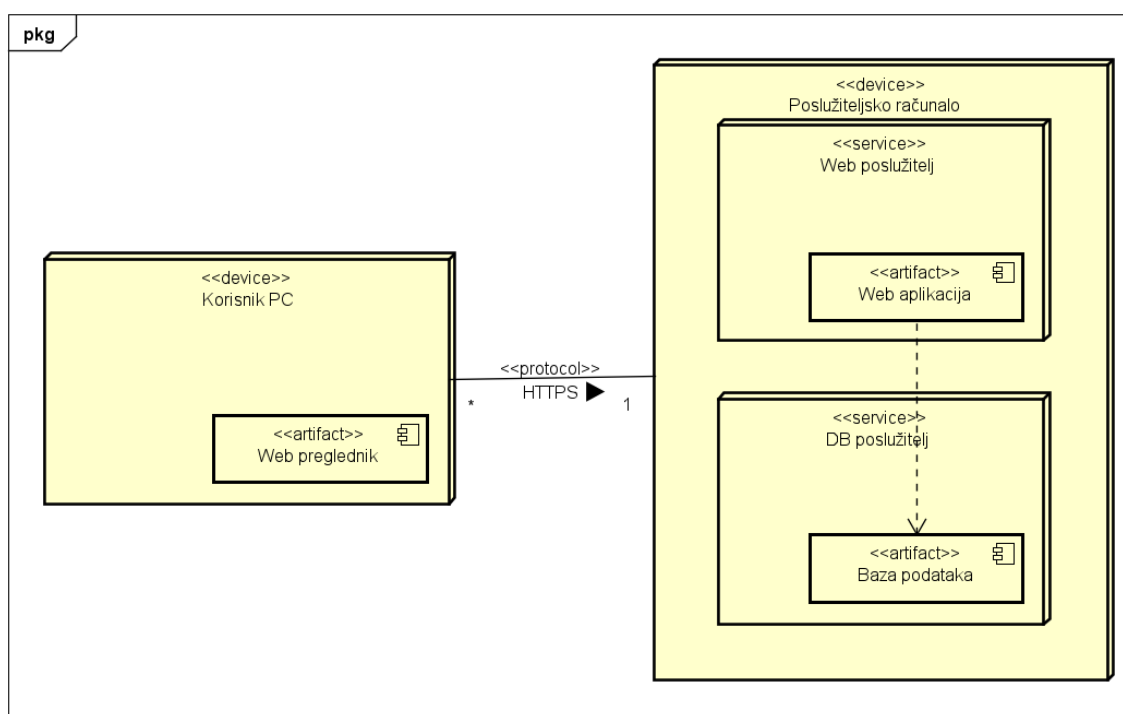
Slika 5.6: Rezultat ispitivanja sustava

**Rezultat:**

Svi očekivani rezultati su zadovoljeni.

## 5.3 Dijagram razmještaja

Dijagrami razmještaja opisuju topologiju sklopovlja i programsku potporu koja se koristi u implementaciji sustava u njegovom radnom okruženju. Na poslužiteljskom računalu se nalaze web poslužitelj i poslužitelj baze podataka. Klijenti koriste web preglednik kako bi pristupili web aplikaciji. Sustav je baziran na arhitekturi "klijent – poslužitelj", a komunikacija između računala korisnika (klijent, zaposlenik, vlasnik, administrator) i poslužitelja odvija se preko HTTPS veze.



Slika 5.7: Dijagram Razmještaja



## 5.4 Upute za puštanje u pogon

Ove upute su prilagođene očekivanom produkcijskom okruženju softverskih aplikacija. U našem slučaju to znači izvođenje na Linux platformi.

### Instalacija poslužitelja baze podataka

Potrebno je preuzeti PostgreSQL server za Linux ubuntu platformu. Nakon preuzimanja softverske podrške za trenutnu platformu potrebno je postaviti korisnika te bazu podataka na koju taj korisnik ima pristup.

### Konfiguracija DBMS poslužitelja

Nakon uspješne instalacije DBMS poslužitelja potrebno je postaviti host i port. Preporučeno je zatvoriti defaultni Postgres port (5432) na razini sustava kako nitko izvana ne bi mogao pristupiti bazi. Dapače moguće je i otvoriti port Postgres baze u sustavskom vatrozidu tijekom razvoja aplikacije kako bi programeri imali olakšani pristup sustavu. Poslužitelj baze podataka radi kao servis kada ga instaliramo u CLI okruženju (što je standardan postupak) te ga ovisno o distribuciji servera možemo ponovno pokrenuti ili zaustaviti pomoću jedne jednostavne naredbe (npr. *service postgres start*).

### Punjenje baze podataka

Punjenje baze podataka jest doista jednostavno. Uz preuzetu datoteku `dump.sql` možemo izvršiti jednostavnu naredbu u terminalu `psql -U username dbname ; dbexport.pgsql`

### Web poslužitelj NGINX

U našoj implementaciji koristimo web poslužitelj NGINX kojime obrađujemo dolazne HTTP zahtjeve i usmjerujemo ih ka pravom odredištu. Ovo nije potpuno nužno za pokretanje aplikacije no preporučeno je kako bi se osigurali stilizirani linkovi na samu aplikaciju. Kod postavljanja tzv. Server blockova za poslužitelj moramo obratiti pozornost da dolazne zahtjeve preusmjerimo na ispravan port. NGINX Web poslužitelj je vrlo često instaliran već na poslužiteljima baziranim na

linux operacijskom sustavu. Alternativno NGINX-u postoji Apache web poslužitelj koji ispunjava istu ulogu.

Unutar direktorija `/var/www/` se tipično nalaze datoteke koje web poslužitelj poslužuje na vanjske zahtjeve. Svaka domena na sustavu ima svoju pod-mapu.

Frontend naše aplikacije izgrađen je u React razvojnom okruženju što nam osigurava mogućnost građenja statičkih minimiziranih datoteka koje su spremne za posluživanje. To osigurava dugotrajnost i stabilnost aplikacije. Naredbom `"npm run build"` možemo izgraditi verziju frontenda spremnu za produkcijsko okruženje. Bitna napomena kod ovog koraka pokretanja aplikacije jesu konfiguracijske varijable, `".env.production"` te `".env.development"`. One sadrže bitne informacije poput adrese poslužitelja te poveznica do određenih resursa.

Tijekom lokalnog razvoja moguće je pokrenuti naredbu `"npm start"` kojom aplikaciju dižemo na lokalnom računalu u svrhu razvoja.

## Backend sustav

Backend infrastruktura naše aplikacije bazirana je na Flask i Gunicorn razvojnim okruženjima u jeziku python. Prije pokretanja je bitno prilagoditi konfiguracijske varijable u datoteci `"codeshark.cfg"`. Tamo se nalaze informacije o putanjama na disku za pohranu, podaci za pristup bazi podataka itd. Nakon ispravne konfiguracije te datoteke moguće je pokrenuti backend naredbom `"gunicorn -config gunicorn_config.py"`.

```
{
  "debug_main":          false,
  "debug_ssl":           false,
  "certfile":            "/etc/letsencrypt/live/domain.com/fullchain.pem",
  "keyfile":             "/etc/letsencrypt/live/domain.com/privkey.pem",
  "flask_host":          "0.0.0.0",
  "flask_port":          "5000",

  "postgres_name":       "codeshark",
  "postgres_host":       "127.0.0.1",
  "postgres_user":       "codeshark_db_user",
  "postgres_pass":       "password",

  "img_upload_dir":      "/var/www/domain.com/images",
  "trophy_upload_dir":   "/var/www/domain.com/trophy",
  "max_content_length":  1048576,

  "debug_mail":          false,
  "mail_url":            "https://domain.com/validate/",
  "rank_upgrade_url":    "https://domain.com/members/",
  "mail_username":       "codeshark@mail.com",
  "admin_mail":          "adm.codeshark@mail.com",
  "mail_password":       "password",
  "smtp_server":         "smtp.mail.com",
  "smtp_port":           587,

  "solutions_dir":       "/var/www/domain.com/code_uploads",
  "user_account_name":   "codeshark_runner",
  "compile_timeout":     10,

  "python_interpreter":  "python3",
  "c++_compiler_version": "c++11",
  "c_compiler_version":  "c11",

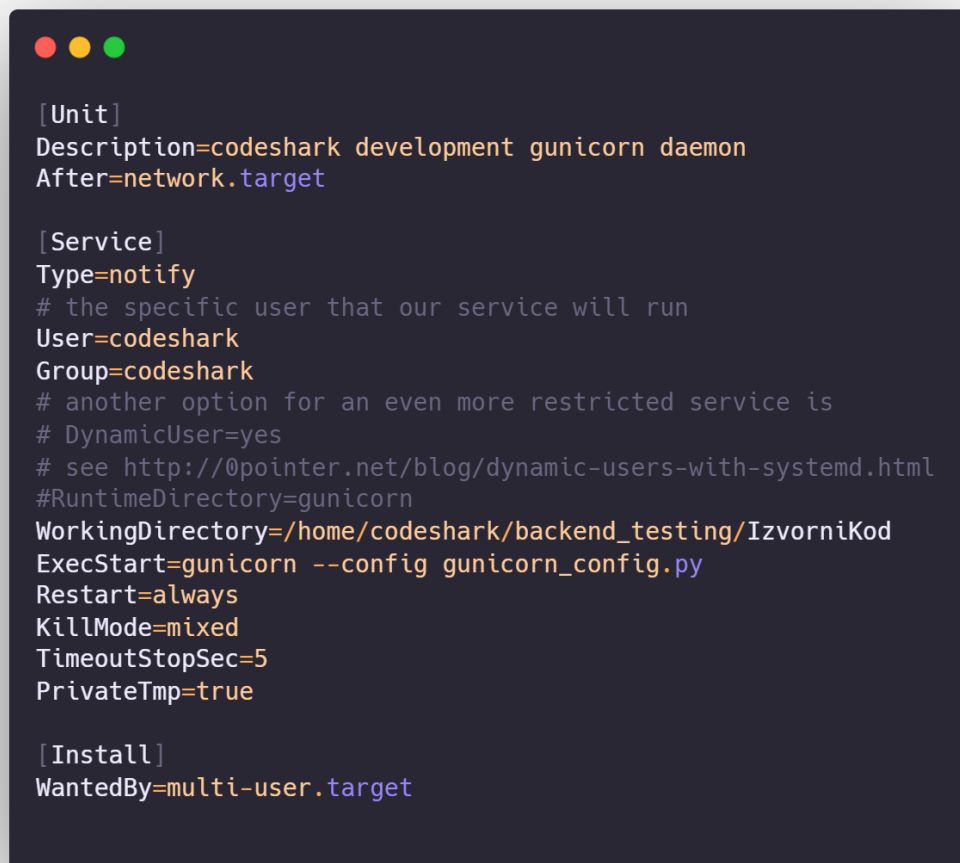
  "avatar_gen_url":      "https://avatars.dicebear.com/api/jdenticon/<seed>"
}
```

Slika 5.8: Primjer konfiguracije backend poslužitelja aplikacije

## Postavljanje servisa na linux platformi

Održavanje ovakvih aplikacijskih sustava uvelike olakšava mogućnost stvaranja Linux servisa za backend procese. Oni nam omogućavaju jednostavno pokretanje i nadzor nad našim kodom. Unutar servisa moguće je specificirati točno na koji način se aplikacija ponaša. Ovdje je bitno obratiti pozornost na korisnika koji izvršava naš kod, njegove dozvole na Linux poslužitelju te putanju do same aplikacije. Ovaj dio koda nam omogućava da se čak i u slučaju greške backend sam auto-

matski može ponovno pokrenut. Dodatne pomoćne naredbe bi bile "journalctl -xe -f -u codeshark-testing" pomoću koje u pravom vremenu možemo promatrati izlaz aplikacije, te tako možemo vidjeti trenutno stanje i potencijalne greške tijekom izvođenja.



```
[Unit]
Description=codeshark development unicorn daemon
After=network.target

[Service]
Type=notify
# the specific user that our service will run
User=codeshark
Group=codeshark
# another option for an even more restricted service is
# DynamicUser=yes
# see http://0pointer.net/blog/dynamic-users-with-systemd.html
#RuntimeDirectory=unicorn
WorkingDirectory=/home/codeshark/backend_testing/IzvorniKod
ExecStart=unicorn --config unicorn_config.py
Restart=always
KillMode=mixed
TimeoutStopSec=5
PrivateTmp=true

[Install]
WantedBy=multi-user.target
```

Slika 5.9: Primjer konfiguracije servisa za backend poslužitelja

Ovaj postupak nije potreban za frontend naše aplikacije pošto je naše web sučelje statički izgrađeno - sav potreban JavaScript kod se izvršava kod klijenta u pregledniku te ne postoji potreba za aktivnim procesom na našem poslužitelju.

## 6. Zaključak i budući rad

Zadatak naše grupe bio je razvoj web aplikacije koja omogućuje provjeru riješenih programskih zadataka i sudjelovanje u natjecanjima. Nakon 17 tjedana rada u timu i razvoja, ostvarili smo zadani cilj. Rad na projektu se vršio kroz dvije faze.

Prva faza projekta je započela okupljanjem tima za razvoj aplikacije, dodjelu projektnog zadatka, koncipiranje izvedbe projektnog zadatka te dokumentiranju zahtjeva. Kvalitetna komunikacija u prvoj fazi uvelike je olakšala daljnji rad pri realizaciji osmišljenog sustava. Rani fokus na izradu vizualnih prikaza svih idejnih rješenja zadanog zadatka je omogućila članovima projekta lakšu koordinaciju i smanjila je pojave raznih nedoumica koje su se mogle pojaviti.

Druga faza projekta, iako nešto kraća od prve, bila je puno intenzivnija po pitanju samostalnog rada članova. Manjak iskustva članova u izradi sličnih implementacijskih rješenja primorao je članove na samostalno učenje odabranih alata i programskih jezika kako bi ispunili dogovorene ciljeve. Nasreću, projektni tim je imao ljude koji su vještiji s odabranim tehnologijama, pa se proces učenja kod ostalih znatno ubrzao. Osim implementacije svih zahtjeva, dio druge faze se isto fokusirao na izvedbe UML dijagrama i popratne dokumentacije kako bi budućim korisnicima bilo lakše koristiti ili vršiti preinake na sustavu.

Komunikacija među članovima tima je bila pretežito uživo kako bi dočarali osjećaj timskog rada i zajedništva. Tome su naravno pomogli redovni "Team-buildinzi" svakakvih vrsta. Ostala komunikacije se vršila putem WhatsAppa i Discorda. Jedno od mogućih proširenja sustava je uvođenje funkcije slanje poruka između korisnika te podizanje foruma što bi uvelike proširilo ciljani fokus na unapređenju pojedinaca kroz rad.

Sudjelovanje na ovakvom projektu bilo je vrijedno iskustvo svim članovima tima jer je to većini prvi ozbiljniji grupni projekt na kojem se moralo u kratkom roku intenzivno raditi. Također se iskusila važnost komunikacije i koordiniranosti između članova tima. Cijeli tim je zadovoljan postignutim i smatra da će ovakvo iskustvo biti vrlo korisno kroz daljnje fakultativno i poslovno obrazovanje.

# Popis literature

## *Kontinuirano osvježavanje*

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

1. Programsko 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>
7. PostgreSQL Community, <https://www.postgresql.org/docs/14/>

# Indeks slika i dijagrama

2.1	<a href="https://leetcode.com/problemset/all/">https://leetcode.com/problemset/all/</a> . . . . .	8
3.1	Dijagram obrasca uporabe, funkcionalnost korisnika i natjecatelja .	21
3.2	Dijagram obrasca uporabe, funkcionalnost voditelja i administratora	22
3.3	Sekvencijski dijagram za UC9 . . . . .	23
3.4	Sekvencijski dijagram za UC12 . . . . .	24
3.5	Sekvencijski dijagram za UC23 . . . . .	25
4.1	E-R dijagram baze podataka . . . . .	34
4.2	Dijagram razreda . . . . .	35
4.3	Dijagram stanja . . . . .	37
4.4	Dijagram aktivnosti . . . . .	38
4.5	Dijagram komponenti . . . . .	39
5.1	Ispitivanje komponenti . . . . .	42
5.2	Rezultat ispitivanja komponenti . . . . .	42
5.3	Ispitivanje sustava prvi dio . . . . .	44
5.4	Ispitivanje sustava drugi dio . . . . .	45
5.5	Ispitivanje sustava treći dio . . . . .	46
5.6	Rezultat ispitivanja sustava . . . . .	46
5.7	Dijagram Razmještaja . . . . .	47
5.8	Primjer konfiguracije backend poslužitelja aplikacije . . . . .	50
5.9	Primjer konfiguracije servisa za backend poslužitelja . . . . .	51
6.1	Prikaz Aktivnosti Na Repozitoriju Prvi Dio . . . . .	59
6.2	Prikaz Aktivnosti Na Repozitoriju Drugi Dio . . . . .	60

# Dodatak: Prikaz aktivnosti grupe

## Dnevnik sastajanja

### 1. sastanak

- Datum: 12. listopada 2021.
- Prisustvovali: M.Damjanić, F.Tomljenović, I.Jeržabek, A.Griparić, R.Mihalić
- Teme sastanka:
  - Moguće alternativne ideje za projekt
  - Početne ideje za tehnologije

### 2. sastanak

- Datum: 14. listopada 2021.
- Prisustvovali: M.Damjanić, F.Tomljenović, I.Jeržabek, A.Griparić, R.Mihalić, L.Maros, F.Jelavić
- Teme sastanka:
  - Dogovorene korištene tehnologije

### 3. sastanak

- Datum: 28. listopada 2021.
- Prisustvovali: M.Damjanić, F.Tomljenović, I.Jeržabek, A.Griparić, R.Mihalić, L.Maros, F.Jelavić
- Teme sastanka:
  - Predložen koncept cijelog projekta
  - Podjela posla po članovima za 1.ciklus

### 4. sastanak

- Datum: 12. studeni 2021.
- Prisustvovali: M.Damjanić, F.Tomljenović, I.Jeržabek, A.Griparić, R.Mihalić, L.Maros, F.Jelavić
- Teme sastanka:
  - Jeržabek je održao predavanje o Reactu
  - Dane upute o spajanju na bazu i pisanju dokumentacije



## 5. sastanak

- Datum: 17. studeni 2021.
- Prisustvovali: M.Damjanić, I.Jeržabek, A.Griparić, R.Mihalić
- Teme sastanka:
  - Sastanak s asistentom i demosom - evaluacija dosadašnjeg rada

## 6. sastanak

- Datum: 8. prosinca 2021.
- Prisustvovali: M.Damjanić, F.Tomljenović, I.Jeržabek, A.Griparić, R.Mihalić, L.Maros, F.Jelavić
- Teme sastanka:
  - Prolaženje kroz svih dijelova projekta sa svim članovima

## 7. sastanak

- Datum: 16. prosinca 2021.
- Prisustvovali: M.Damjanić, F.Tomljenović, I.Jeržabek, A.Griparić, R.Mihalić, L.Maros
- Teme sastanka:
  - Dogovorene raspodjela poslova za 2.ciklus

## 8. sastanak

- Datum: 21. prosinac 2021.
- Prisustvovali: F.Tomljenović, I.Jeržabek, A.Griparić, R.Mihalić, L.Maros
- Teme sastanka:
  - sastanak s asistentom i demosom - demonstracija alfa inačice

## 9. sastanak

- Datum: 13. siječnja 2022.
- Prisustvovali: M.Damjanić, I.Jeržabek, R.Mihalić
- Teme sastanka:
  - sastanak s asistentom za zadnje implementacije i izvođenje buduće prezentacije

## Tablica aktivnosti

	Marko Damjanić	Ivan Jeržabek	Roko Mihalić	Fran Tomljenović	Fran Jelavić	Antonio Griparić	Luka Maros
Upravljanje projektom	35						
Opis projektnog zadatka	6						
Funkcionalni zahtjevi					5		
Opis pojedinih obrazaca						10	10
Dijagram obrazaca						4	
Sekvencijski dijagrami							4
Opis ostalih zahtjeva	2						
Arhitektura i dizajn sustava					5		
Baza podataka	15		5	5			
Dijagram razreda			3				
Dijagram stanja	3						
Dijagram aktivnosti	2						
Dijagram komponenti	2	1	1				
Korištene tehnologije i alati	1	1	1				
Ispitivanje programskog rješenja	3		8				
Dijagram razmještaja	1						
Upute za puštanje u pogon	2	2	2				
Dnevnik sastajanja	2						
Zaključak i budući rad	1						

Nastavljeno na idućoj stranici

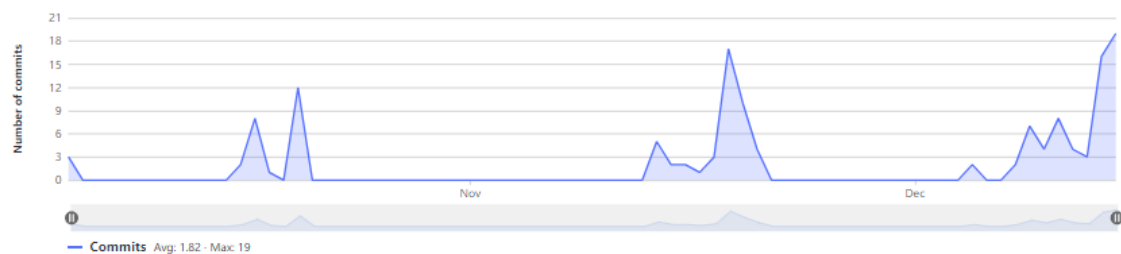
Nastavljeno od prethodne stranice

	<b>Marko Damjanić</b>	<b>Ivan Jeržabek</b>	<b>Roko Mihalić</b>	<b>Fran Tomljenović</b>	<b>Fran Jelavić</b>	<b>Antonio Griparić</b>	<b>Luka Maros</b>
Popis literature	1						
Izrada početne stranice					15		
Izrada stranice s zadacima		10				15	
Izrada stranice odabranog zadatka		15				25	
Izrada stranice natjecanja		55					
Izrada stranice korisničkih profila		4					35
Izrada stranice članova		2	1		15		
Izrada baze podataka	20						
Izrada vizualnog identiteta					6		
Back end			130	110			
Izvršavanje zadataka				20			
Puštanje u pogon		35		5			

## Dijagrami pregleda promjena

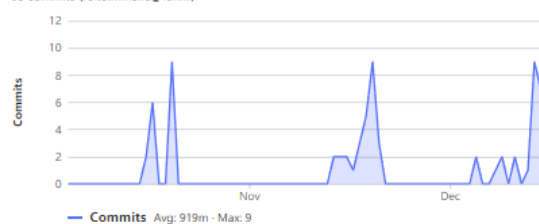
### Commits to main

Excluding merge commits. Limited to 6,000 commits.



### Roko

68 commits (roko.mihalic@fer.hr)



### Fran Tomljenovic

59 commits (fran.tomljenovic@fer.hr)



### Ivan

8 commits (ijerzabek.ivan@gmail.com)



Slika 6.1: Prikaz Aktivnosti Na Repozitoriju Prvi Dio



Slika 6.2: Prikaz Aktivnosti Na Repozitoriju Drugi Dio