

# Programsko inženjerstvo

Ak. god. 2023./2024.

*BytePit*

Dokumentacija, Rev. 2

Grupa: *Koder kolege*

Voditelj: *Petra Kelković*

Datum predaje: *17. studenoga 2023.*

Nastavnik: *Hrvoje Nuić*

# Sadržaj

<b>1 Dnevnik promjena dokumentacije</b>	<b>3</b>
<b>2 Opis projektnog zadatka</b>	<b>5</b>
<b>3 Specifikacija programske potpore</b>	<b>11</b>
3.1 Funkcionalni zahtjevi . . . . .	11
3.1.1 Obrasci uporabe . . . . .	14
3.1.2 Sekvencijski dijagrami . . . . .	23
3.2 Ostali zahtjevi . . . . .	25
<b>4 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 . . . . .	32
4.2 Dijagram razreda . . . . .	34
4.3 Dijagram stanja . . . . .	36
4.4 Dijagram aktivnosti . . . . .	37
4.5 Dijagram komponenti . . . . .	39
<b>5 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 . . . . .	47
5.3 Dijagram razmještaja . . . . .	49
5.4 Upute za puštanje u pogon . . . . .	50
<b>6 Zaključak i budući rad</b>	<b>57</b>
<b>Popis literature</b>	<b>58</b>
<b>Indeks slika i dijagonama</b>	<b>60</b>

**Dodatak: Prikaz aktivnosti grupe**

**61**

# 1. Dnevnik promjena dokumentacije

Rev.	Opis promjene/dodataka	Autori	Datum
0.1	Personalizirana naslovna stranica te header i footer.	Dora Bilić-Pavlinović	28.10.2023.
0.2.1	Opis projektnog zadatka - opći opis	Dora Bilić-Pavlinović	28.10.2023.
0.2.2	Opis projektnog zadatka - slične aplikacije i druge primjene	Matea Cvetković	29.10.2023.
0.2.3	Opis projektnog zadatka - final touches	Filip Mohler	31.10.2023.
0.3.1	Funkcijski zahtjevi	Petra Kelković, Mislav Korotaj	30.10.2023.
0.4.1	Obrasci upotrebe (UC dijagrami) - prvi dio	Mislav Korotaj	31.10.2023.
0.4.2	Obrasci upotrebe - drugi dio	Mislav Korotaj, Nives Ostojić, Filip Mohler	31.10.2023.

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

<b>Rev.</b>	<b>Opis promjene/dodataka</b>	<b>Autori</b>	<b>Datum</b>
0.4.3	Obrasci upotrebe - treći dio	Filip Mohler	3.11.2023.
0.4.4	Obrasci upotrebe - dijagrami	Filip Mohler	6.11.2023.
0.5.1	Sekvencijski dijagrami	Petra Buršić	6.11.2023.
0.5.2	Sekvencijski dijagrami - nastavak	Petra Buršić	7.11.2023.
0.5.3.	Sekvencijski dijagrami - finalno	Petra Buršić	10.11.2023.
0.6	Unos sastanka: podjela uloga Plan daljnog rada	svi	20.10.2023.
0.7	Opis arhitekture sustava i baze podataka	Nives Ostojić	10.11.2023.
0.8	Ispravak sekvencijskih dijagrama Dodani ostali zahtjevi	Filip Mohler	13.11.2023.
0.9	Popravljena dokumentacija na sastanku	svi	14.11.2023.
0.10	Dijagrami razreda + opis	Matea Cvetković	15.11.2023.
0.11	ER dijagram baze i opis	Filip Mohler	16.11.2023.
<b>1.0</b>	Verzija samo s bitnim dijelovima za 1. ciklus	*	16.11.2023.

## 2. Opis projektnog zadatka

Tema našeg projektnog rada je izrada web aplikacije "BytePit" koja omogućuje korisnicima sudjelovanje u programerskim natjecanjima i provjeru riješenih zadataka. Ideja je da naša stranica ima sve potrebno za obavljanje natjecanja poput registracije korisnika, uključivanje u natjecanje, pribavljanje zadataka, vrednovanje priloženih rješenja, prikaz dosadašnjih uspjeha natjecatelja i još mnogo toga.

Neregistrirani korisnik može se registrirati definirajući registrira li se kao **voditelj** ili **natjecatelj**. Za registraciju korisnika potrebno je unijeti

- korisničko ime
- fotografiju
- lozinku
- ime
- prezime
- email adresu

Uspješnost registracije potvrđuje se preko email adrese dok voditelja dodatno potvrđuje i administrator.

*Neregistrirani korisnik* na web stranici može vidjeti kalendar s natjecanjima te pregledati rezultate prethodnih natjecanja - rang listu te sve zadatke koje su korisnici predali na tom natjecanju. Svi registrirani korisnici automatski nasljeđuju sve mogućnosti koje neregistrirani korisnici imaju.

*Registrirani korisnik* može vidjeti kalendar s aktualnim natjecanjima kojima može pristupiti te virtualnim natjecanjima koja može rješavati za vježbu. Prilikom sudjelovanja u natjecanju korisnik može koristiti playground pomoću kojeg provjerava točnost svog rješenja, a zatim može učitati datoteku s programskim kodom koja predstavlja konačno rješenje u aplikaciju. Također može pristupiti stranici za vježbu na kojoj bira želi li rješavati zadatke pojedinačno ili pokrenuti opciju virtualnog natjecanja. Na stranici "korisnici" može pregledavati profile svih registriranih korisnika te uređivati svoj profil. Na stranici "rezultati" registriranom korisniku omogućuje se opcija preuzimanja svih predanih rješenja zadatka ukoliko ga je riješio u potpunosti točno.

*Natjecatelj* je registrirani korisnik koji sudjeluje u natjecanjima. Na njegovom profilu prikazana je statistika s podacima o broju predanih i točnih programske rješenja. Prikazani su i pehari za ona natjecanja na kojima je sudjelovao i osvojio jedno od prva tri mesta.

*Voditelj* je registrirani korisnik koji ima ovlasti kreiranja sadržaja. Moguće je kreirati pojedinačni zadatak ili novo natjecanje. Nakon kreiranja, novi je sadržaj moguće uređivati. Na njegovom profilu vidljivi su svi javni zadatci koje je kreirao i kalendar sa svim natjecanjima koja je izradio.

*Administrator* nasljeđuje sve ovlasti registriranih korisnika. Dodatno mu je omogućeno uređivanje svih korisničkih profila, svih zadataka i svih budućih natjecanja (aktualna i prošla natjecanja ne mogu se više uređivati). Njegov zadatak je i potvrđivanje novoregistriranih voditelja natjecanja što može napraviti klikom na gumb na profilu novog voditelja.

## Provedba natjecanja

Kada dođe vrijeme koje je voditelj postavio kao početak natjecanja, zadatci ispita postaju vidljivi aktivnim natjecateljima. Za svaki zadatak natjecatelji mogu testirati točnost svog rješenja pomoću prozora za testiranje te priložiti datoteku s programskim kodom u jeziku Java. Pristupanje natjecanju omogućeno je u periodu trajanja natjecanja, a jednom kada mu se pristupi može ga se rješavati onoliko vremena koliko je vremensko ograničenje rješavanja svih zadataka. Natjecanju je moguće pristupiti samo jednom. Korisnik može pristupiti rezultatima tek kad natjecanje više nije aktivno. Rezultati se prikazuju oblikom rang liste svih sudionika poredanih silazno po prikupljenom broju bodova. Vidljiv je i popis i statistika svih rješenja koja su korisnici predali na natjecanju, po zadatcima i po korisnicima. Pri kalkulaciji broja bodova uzima se u obzir postotak točnosti i isteklo vrijeme. Onima koji su se plasirali na prva tri mesta pridodaje se slika pehara na njihovom profilu. Po završetku natjecanja svi zadatci koji su bili privatni postaju javni i vidljivi su na stranici za vježbu, a natjecanje postaje virtualno.

## Virtualno natjecanje

Virtualno natjecanje je koncept osmišljen kako bi natjecatelji mogli provjeriti koliko su se dobro pripremili za nadolazeće natjecanje. Virtualnom natjecanju se može pristupiti preko kalendara i preko stranice za vježbu. Postoje dvije vrste virtualnog natjecanja: nasumično generirano natjecanje, u kojem aplikacija na-

sumično odabire pet zadataka iz baze, te simulacija prethodno održanog natjecanja koje je potrebno izabrati iz kalendarja. Po završetku korisniku se prikazuje rang lista s ukupno ostvarenim brojem bodova, a ukoliko već postoji rang lista za to natjecanje prikazuje mu se i njegov virtualni rang.

## Slične aplikacije

Već postojeća aplikacija vrlo slična ovoj je Edgar koji se koristi na FER-u za provođenje ispita i laboratorijskih vježbi. S obzirom na to da je svrha te aplikacije ipak drugačija od naše, postoje neke značajne razlike. Dok se za registrirani pristup našoj aplikaciji korisnik sam prijavljuje i čeka potvrdu administratora, u Edgaru to čini administrator samostalno dodavajući korisnike (kojima se kasnije dodijele njihovi pristupni podaci). Zbog same razlike u namjeni, predana rješenja se drugačije budu (nekim stalnim brojem bodova, bez ovisnosti o vremenu). Također, studentu prijavljenom u sustav nije omogućen pregled tuđih rješenja kao što je to slučaj u našoj aplikaciji, kao ni pristup pojedinačnim zadatcima: moguće je samo pokrenuti probni ispit ili vježbu, bez mogućnosti odabira pojedinog zadatka (slika 2.1).

The screenshot shows the Edgar application interface. At the top, there's a navigation bar with links for 'My previous exams', 'My tickets', 'My stats', 'Playground', 'Code theme', and a dropdown for 'B\_BP' and '2022/2023'. On the right, it shows the user's name 'Ime Prezime Student' and a 'Logout' button. Below the navigation, there's a search bar with 'Password...' placeholder and a 'Start exam' button. A message says 'You have unsubmitted private exams:' followed by a table with one row. The table has columns for '#', 'Title', 'Started at', 'Available until', and 'Can continue?'. The row shows entry #1 for '3. laboratorijska vježba' started at 2023-05-25 19:03:52, available until 2023-06-02 11:00:00, and 'Can't continue: expired.' Below this, there's a section titled 'Public exams:' with a table listing five exams from different academic years (2022/2023, 2023/2024, 2019/2020, 2020/2021, 2020/2021) with their titles, start times, durations, and 'Start' buttons.

#	Title	Started at	Available until	Can continue?
1	3. laboratorijska vježba	2023-05-25 19:03:52	2023-06-02 11:00:00	Can't continue: expired.

#	Academic year	Exam title, questions no	Runs/Max	Duration	Score ignored	Forward only	Used in stats	Global	Public	Available for	Start exam
1	2022/2023	Vježba: Završni 22/23 9 question(s)	0/0	3 hours	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	9 years expires: 2023-09-27 12:00:00	<button>Start</button>
2	2022/2023	Vježba: Meduispit 22/23 8 question(s)	0/0	2 hours	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	9 years expires: 2023-09-27 12:00:00	<button>Start</button>
3	2019/2020	Vježba: Završni ispit 19/20 8 question(s)	0/0	3 hours	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	9 years expires: 2023-09-27 12:00:00	<button>Start</button>
4	2020/2021	Vježba: Dekanski rok 20/21 10 question(s)	0/0	2 hours	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	9 years expires: 2023-09-27 12:00:00	<button>Start</button>
5	2020/2021	Vježba: Jesenski rok 20/21	0/0	3 hours	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	9 years	<button>Start</button>

Slika 2.1: Edgar: početna stranica i popis vježbi za ispite

Ostale funkcionalnosti BytePita vrlo su slične Edgaru: uloge natjecatelja i studenta su slične, oni mogu učitavati i provjeravati svoj kod, pokrenuti probni ispit (slika 2.2) (u BytePitu virtualno natjecanje) kao i pristupiti ispitu (odnosno natjecanju). Koncept natjecanja i ispita vrlo je sličan - korisnicima su dostupni svi ispitni zadaci istovremeno, a po završetku se ti zadatci objavljaju na stranici za vježbu. Ono

što u BytePitu predstavlja uloga voditelja, u Edaru je asistent/profesor koji ima ovlasti objavljivanja tj. izrade zadataka i organizacije ispita (odabir zadataka, trajanja). U Edaru čak postoji i stranica sa statistikom koja prikazuje uspješnost u odnosu na druge studente, postotak točno riješenih zadataka i sl.(slika 2.3). BytePit ima stranicu slične namjene, ali ipak s drugačijim podatcima: na njoj natjecatelj može vidjeti tuđa rješenja i njihovu uspješnost, kao i svoj rang.

The screenshot shows a question from an online exam. The question is: "Zadatak 3. (4 bodova) Streamflix želi stvoriti novu ulogu MOVIE\_CLEANER koja će obavljati slijedeće poslove: - pregled svih stupaca iz tablice TRACK - ažuriranje svih nekućnih atributa iz tablice TRACK osim tipa sadržaja - brisanje redaka iz tablice TRACK". Below the question, there are four answer options: 1. (-), 2. (-), 3. (-) [selected], 4. (-), 5. (-), 6. (-), 7. (-), 8. (-), 9. (-). The interface includes a user profile icon, a timer showing 02 : 29 : 48, and a "Submit" button. At the bottom, it says "NAPOMENA: U zadatu se ne smiju koristiti procedure i okidači."

Slika 2.2: Edgar: probni ispit

The screenshot shows a student's performance statistics. It includes a summary card with the student's name (Ime Prezime), rank (#104 / 567), points (81.98 / 275), and percentile rank (82%). Below this are three circular progress indicators: Percentage (30%), Percentage attempted (86%), and Average score (89%). A section titled "Exams taken: 16 / 30" provides overall statistics for those exams. The bottom section, "Selected exam statistics", contains two charts: "Exam score percentage distribution (n = 567)" (a bar chart showing the number of students vs score percentage) and "Per Question" (a bar chart showing the percentage of students who answered each question correctly). On the right, there is a summary card for the current exam, showing rank (#13 / 567), percentile rank (97%), and score (27 / 30).

Slika 2.3: Edgar: stranica sa statistikom

Osim Edgara, postoji još niz aplikacija sličnih BytePitu, a jedna od njih je Codeforces, web aplikacija koja omogućuje sudjelovanje u online natjecanjima. Gotovo i da

nema razlike među ovim aplikacijama: na profilima korisnika vidljiva je njihova statistika, omogućen je pristup virtualnim natjecanjima koja simuliraju prava, vidljiva je lista zadataka kao i njihovih rješenja koja su učitali korisnici (slika 2.4)... Ta rješenja nisu uvijek vidljiva, vidljivost ovisi o postavkama natjecanja tako je da ovisno o sudjelovanju nekim korisnicima onemogućen pregled predanih rješenja. Nasuprot tomu, u BytePitu rješenja može dohvatiti samo natjecatelj koji je i sam točno riješio zadatak. Bitna razlika u ovom je slučaju također i to što Codeforces omogućava svim korisnicima da učitaju zadatke, koji potom prolaze dodatne provjere da bi se utvrdila njihova ispravnost, dok je u BytePitu ta mogućnost otvorena samo voditeljima, i to bez dodatnih provjera nakon objave zadatka. Na profilima korisnika koji su zadatke učitali ti zadatci nisu vidljivi (u BytePitu se oni nalaze na profilima voditelja). Slično kao u našoj aplikaciji, nakon natjecanja moguće je na profilima korisnika vidjeti njihova rješenja i rezultate testova ali čak i bez registracije: svaki korisnik vidi rješenja svakog korisnika te nije potrebna registracija. Ono što registracija omogućuje je, dakako, sudjelovanje u natjecanjima i virtualnim natjecanjima te izvršavanje i predaja koda za riješene zadatake za vježbu (slika 2.5) (neregistrirani korisnik može vidjeti tekst zadatka, ali ne može izvršiti kod i time provjeriti točnost rješenja). Ono što BytePit omogućuje, a Codeforces ne je mogućnost virtualnog natjecanja koje se sastoji od nasumičnih zadataka: u potonjem se nude samo replike stvarnih natjecanja koje se mogu pokrenuti.

Problem 1890B - Qingshan Loves Strings							
Contest status							
#	When	Who	Problem	Lang	Verdict	Time	Memory
230466190	Oct/30/2023 09:20 UTC+1	eugalt	1890B - Qingshan Loves Strings	Python 3	Accepted	187 ms	41700 KB
230465212	Oct/30/2023 09:10 UTC+1	eugalt	1890B - Qingshan Loves Strings	Python 3	Accepted	62 ms	0 KB
230305402	Oct/29/2023 04:29 UTC+1	eugalt	1890B - Qingshan Loves Strings	Python 3	Accepted	62 ms	100 KB
230492736	Oct/30/2023 13:15 UTC+1	rajvirsingh192002	1890B - Qingshan Loves Strings	Python 3	Accepted	170 ms	41700 KB
230379325	Oct/29/2023 15:18 UTC+1	Sparkle_Twilight	1890B - Qingshan Loves Strings	Python 3	Accepted	46 ms	100 KB
230279886	Oct/28/2023 20:14 UTC+1	eugalt	1890B - Qingshan Loves Strings	Python 3	Accepted	46 ms	0 KB
230215478	Oct/28/2023 16:18 UTC+1	eugalt	1890B - Qingshan Loves Strings	Python 3	Accepted	62 ms	0 KB
230383099	Oct/29/2023 15:48 UTC+1	Leo25Darklight	1890B - Qingshan Loves Strings	PyPy 3-64	Accepted	202 ms	7400 KB
230222712	Oct/28/2023 16:31 UTC+1	_M_H_M_	1890B - Qingshan Loves Strings	Python 3	Accepted	77 ms	0 KB
230192760	Oct/28/2023 15:48 UTC+1	emrakul	1890B - Qingshan Loves Strings	PyPy 3	Accepted	249 ms	5600 KB
230274516	Oct/28/2023 19:29 UTC+1	MdNazmulHossain	1890B - Qingshan Loves Strings	Python 3	Accepted	61 ms	0 KB
230211866	Oct/28/2023 16:12 UTC+1	just_sai	1890B - Qingshan Loves Strings	PyPy 3-64	Accepted	670 ms	10900 KB
230377483	Oct/29/2023 15:03 UTC+1	Aldibek	1890B - Qingshan Loves Strings	PyPy 3-64	Accepted	670 ms	10900 KB

Slika 2.4: Codeforces: rješenja različitih korisnika



Slika 2.5: Codeforces: mogućnost izvršavanja koda

BytePit je aplikacija koja svakako može imati širu primjenu od ovdje opisane: dok je trenutna verzija aplikacije pogodna uglavnom za programerska natjecanja, s manjim preinakama ona bi se mogla koristiti u razne svrhe. Svakako bi bila dobra ideja koristiti aplikaciju kao svojevrsni test pri zapošljavanju programera odnosno za selekciju najboljih kandidata: kandidati bi dobili podatke za pristup i od njih bi se tražilo da riješe određen broj zadataka (naravno, drugačije vrste od natjecateljskih). Tako bi se lakše probralo bolje kandidate koji ulaze u uži izbor za određenu poziciju. Prilagodbom težine zadataka, BytePit bi mogao postati i platforma za vježbu i učenje programiranja. Naravno, u tom bi slučaju bilo potrebno osmisliti i kratke tečajeve programiranja, kao što to postoji na npr. Codecademy-u. Aplikacija bi se mogla proširiti i na način da bude slična gore predstavljenoj aplikaciji Edgar: mogla bi biti platforma za ispite iz programiranja, kako na fakultetu, tako i u osnovnim i srednjim školama.

# 3. Specifikacija programske potpore

## 3.1 Funkcionalni zahtjevi

Dionici:

1. Naručitelj
2. Voditelj natjecanja
3. Natjecatelj
4. Administrator
5. Razvojni tim

Aktori i njihovi funkcionalni zahtjevi:

1. Neregistrirani korisnik (inicijator) može:
  - (a) vidjeti kalendar s dostupnim natjecanjima
  - (b) poslati zahtjev za registracijom
  - (c) vidjeti rezultate prethodno održanih natjecanja
2. Registrirani korisnik (inicijator) može:
  - (a) pregledavati programske zadatke objavljene na stranici
  - (b) pristupiti aktivnim i virtualnim natjecanjima
  - (c) uređivati svoj profil
  - (d) pregledavati profile drugih korisnika (natjecatelja i voditelja natjecanja)
  - (e) za vrijeme trajanja natjecanja:
    - i. vidjeti aktualne zadatke
    - ii. poslati datoteku s programskim kodom za svaki zadatak
  - (f) nakon natjecanja:
    - i. vidjeti popis učitanih rješenja drugih natjecatelja
    - ii. za svaki pojedini zadatak vidjeti popis svih natjecatelja koji su učitali rješenje za taj zadatak, broj točnih primjera po najboljem učitavanju od natjecatelja i prosječno vrijeme izvršavanja po primjeru

iii. dohvatiti učitano rješenje za pojedini zadatak ukoliko je on u potpunosti točno riješen

(g) vježbati prethodno objavljene zadatke

i. učitati rješenje zadatka u aplikaciju

(h) pokrenuti virtualno natjecanje:

i. odabirom prošlog natjecanja iz kalendara

ii. odabirom opcije rješavanja nasumičnih zadataka iz prethodnih natjecanja

3. Voditelj natjecanja (inicijator) može:

(a) učitati nove zadatke u aplikaciju

(b) organizirati natjecanje:

i. odabire vrijeme početka i završetka

ii. određuje broj zadataka

iii. odlučuje koji su zadaci aktivni

iv. po želji učitava sličicu pehara

(c) izraditi zadatak

(d) uređivati vlastito objavljene zadatke i natjecanja (to ne mijenja prijašnje rezultate)

4. Administrator (inicijator) može:

(a) vidjeti popis svih registriranih korisnika (potvrđenih i nepotvrđenih) i njihovih osobnih podataka

(b) mijenjati dodijeljena prava registriranim korisnicima

(c) mijenjati osobne podatke registriranih korisnika

(d) potvrditi/odbiti registracijski zahtjev za ulogu voditelja

(e) uređivati sve zadatke i natjecanja

5. Baza podataka (sudionik):

(a) pohranjuje sve podatke o registriranim korisnicima

(b) briše podatke o korisnicima koji nisu potvrdili registracijski mail unutar 24 sata

(c) briše podatke o voditeljima koje nije potvrdio administrator unutar 7 dana

(d) pohranjuje opise svih zadataka i njihova rješenja

(e) pohranjuje sve podatke o natjecanjima i njihove rang liste

- (f) pohranjuje sva rješenja koja su korisnici učitali tijekom pravog natjecanja
- (g) pohranjuje sve statistike vezane uz korisnike i natjecanja

### 3.1.1 Obrasci uporabe

#### Opis obrazaca uporabe

##### UC1 - Registracija

- **Glavni sudionik:** Neregistrirani korisnik
- **Cilj:** Registracija novog korisnika
- **Sudionici:** Baza podataka, Administrator
- **Preduvjet:** /
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za registraciju
  2. Otvara se obrazac u koji upisuje podatke:
    - (a) korisničko ime
    - (b) fotografija
    - (c) lozinka
    - (d) ime i prezime
    - (e) email adresa
    - (f) odabire: voditelj natjecanja / natjecatelj
  3. Upisuje i odabire potrebne podatke
  4. Korisnik dobiva poruku da potvrdi registraciju putem email adrese
  5. Korisniku se šalje mail za potvrdu registracije
  6. Korisnik potvrđuje registraciju putem linka te se otvara stranica s porukom dobrodošlice
- **Opis mogućih odstupanja:**
  - 2.a Email/korisničko ime su već zauzeti
    1. Korisnik dobiva poruku da je mail/korisničko ime u upotrebi
    2. Traži se ponovni upis podataka
  - 2.f Korisnik se registrira kao "voditelj natjecanja"
    1. Administratoru je omogućena potvrda korisnika na njegovom profilu
  - 5.a Korisnik ne potvrđuje email unutar 24 sata
    1. Korisnikov profil se briše iz baze podataka

##### UC2 - Prijava u sustav

- **Glavni sudionik:** Neprijavljeni korisnik
- **Cilj:** Dobiti pristup korisničkom sustavu

- **Sudionici:** Baza podataka
- **Preduvjet:** Registracija
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju prijava
  2. Unos korisničkog imena i lozinke
  3. Potvrda od sustava o ispravnim podatcima
  4. Pristup korisničkim opcijama
- **Opis mogućih odstupanja:**
  - 3.a Neispravno korisničko ime ili lozinka
    1. Korisnik dobiva poruku o neispravnom korisničkom imenu ili lozinki

#### UC3 - Pregled zadataka za vježbu

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Pregled svih javnih zadataka
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju Vježba -> Zadaci za vježbu
  2. Otvara se stranica s popisom svih zadataka
  3. Korisnik odabire zadatak
  4. Otvara se stranica sa odabranim zadatkom te se prikazuje tekst zadatka

#### UC4 - Pregled rezultata natjecanja

- **Glavni sudionik:** Korisnik
- **Cilj:** Pregled rezultata prethodno održanog natjecanja
- **Sudionici:** Baza podataka
- **Preduvjet:**/
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju rezultati
  2. Korisniku je prikazana lista prethodnih natjecanja s opcijama "Pogledaj ljestvicu!" i "Pogledaj rješenja!"

#### UC5 - Pregled rang liste natjecanja

- **Glavni sudionik:** Korisnik
- **Cilj:** Pregled rang liste prethodno održanog natjecanja

- **Sudionici:** Baza podataka
- **Preduvjet:**
- **Opis osnovnog tijeka:**
  1. UC4 - Pregled rezultata natjecanja
  2. Korisnik odabire opciju "Pogledaj ljestvicu" za odabranou natjecanje
  3. Korisnik dobije listu s bodovima, rangom te imenima svih sudionika

### **UC6 - Pregled rješenja sudionika natjecanja**

- **Glavni sudionik:** Korisnik
- **Cilj:** Pregled rješenja i statistika svih natjecatelja koji su sudjelovali u odabranom natjecanju
- **Sudionici:** Baza podataka
- **Preduvjet:**
- **Opis osnovnog tijeka:**
  1. UC4 - Pregled rezultata natjecanja
  2. Korisnik odabire opciju "Pogledaj rješenja!" za odabranou natjecanje
  3. Prikazuju se sljedeći podatci:
    - (a) klikom na zadatak
      - i. Svi natjecatelji koji su učitali neko rješenje za pojedini zadatak
      - ii. Broj točnih primjera za svakog natjecatelja
      - iii. Vrijeme rješavanja po natjecatelju
    - (b) klikom na "Po korisnicima"
      - i. Popis svih natjecatelja koji su učitali bar neko rješenje na natjecanju
      - ii. popis svih zadataka koje je pojedini korisnik učitao

### **UC7 - Pregled kalendara**

- **Glavni sudionik:** Korisnik
- **Cilj:** Pregled kalendara sa svim natjecanjima
- **Sudionici:** Baza podataka
- **Preduvjet:** /
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju kalendar
  2. Otvara se stranica sa prikazom mjesečnog kalendara u kojem su označena natjecanja

### **UC8 - Pregled profila korisnika**

---

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Pregled pojedinog korisnika
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju korisnici
  2. Korisniku je prikazana lista profila svih korisnika

#### **UC9 - Pregled profila natjecatelja**

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Pregled profila natjecatelja
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. UC8 - Pregled profila korisnika
  2. Korisnik odabire profil korisnika s ulogom natjecatelja
  3. Prikazuje se profil natjecatelja:
    - (a) Statistika o broju točno riješenih zadataka
    - (b) Statistika o broju isprobanih zadataka
    - (c) Pehari za osvojena natjecanja

#### **UC10 - Pregled profila voditelja**

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Pregled profila korisnika s ulogom voditelja
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. UC8 - Pregled profila korisnika
  2. Korisnik odabire profil korisnika s ulogom voditelja
  3. Prikazuje se profil voditelju:
    - (a) Svi objavljeni zadatci voditelja
    - (b) Kalendar sa svim natjecanjima voditelja

#### **UC11 - Sudjelovanje u natjecanju**

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Natjecanje

- **Sudionici:** Baza podataka
- **Preduvjet:** korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. Natjecatelj odabire aktivno natjecanje na početnoj stranici
  2. Otvara se stranica s natjecanjem
  3. U trenutku početka natjecanja, svi zadaci postaju vidljivi
  4. Natjecatelj rješava zadatke unutar vremenskog ograničenja
  5. Natjecatelj predaje rješenja zadataka
- **Opis mogućih odstupanja:**
  - 3.a Pokušaj pristupa aktivnom natjecanju nakon prvog sudjelovanja
    1. Korisnik dobiva poruku o nemogućnosti pristupa natjecanju više od jednom.

### UC12 - Virtualno natjecanje

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Simulacija natjecanja
- **Sudionici:** Baza podataka
- **Preduvjet:** korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. Natjecatelj odabire opciju vježba
  2. Natjecatelj odabire opciju Virtualno natjecanje
  3. Natjecatelju se otvaraju dvije opcije:
    - (a) Odabir prošlog natjecanja
    - (b) Nasumično generirano natjecanje

### UC13 - Prethodno natjecanje

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Simulacija prethodnog natjecanja
- **Sudionici:** Baza podataka
- **Preduvjet:** korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. UC12 - Virtualno natjecanje
  2. Korisnik odabire opciju Odabir prošlog natjecanja
  3. Otvara se kalendar
  4. Korisnik bira natjecanje

5. Korisnik se po završetku natjecanja rangira u odnosu na službene retul-tate originalnog natjecanja

#### UC14 - Nasumično generirano natjecanje

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Simulacija natjecanja
- **Sudionici:** Baza podataka
- **Preduvjet:** korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. UC12 - Virtualno natjecanje
  2. Natjecatelj odabire opciju Nasumično generirano natjecanje
  3. Aplikacija nasumično odabire pet zadataka
  4. Korisniku se po završetku natjecanja prikazuje broj bodova koji je ostva-rio

#### UC15 - Rješavanje zadataka za vježbu

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Vježbanje zadataka za natjecanje
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju vježba
  2. Korisnik odabire opciju Zadaci za vježbu
  3. Otvara se stranica s popisom svih zadataka
  4. Korisnik odabire zadatak
  5. Otvara se stranica sa zadatkom
  6. Korisnik učitava svoje rješenje
  7. Korisnik odabire opciju Predaj te dobiva povratnu informaciju o točnosti svog rješenja

#### UC16 - Izrada natjecanja

- **Glavni sudionik:** Voditelj
- **Cilj:** Napraviti natjecanje za korisnike
- **Sudionici:** Baza podataka
- **Preduvjet:** korisnik je prijavljen kao voditelj
- **Opis osnovnog tijeka:**

1. Voditelj odabire opciju Kreiraj sadržaj
2. Voditelj odabire opciju Kreiraj natjecanje
3. Odabire vrijeme početka i završetka natjecanja
4. Bira broj zadataka
5. Unosi tekst zadatka
6. Bira broj bodova za zadatak
7. Odabire po želji sličicu pehara

### UC17 - Uređivanje svojih natjecanja

- **Glavni sudionik:** Voditelj
- **Cilj:** Ispraviti pogreške ili dodati zadatke
- **Sudionici:** Baza podataka
- **Preduvjet:** korisnik je prijavljen kao voditelj
- **Opis osnovnog tijeka:**
  1. UC10 - Pregled profila voditelja
  2. Voditelj odabire opciju "Uredi natjecanje" na svom profilu pored odbaranog natjecanja
  3. Voditelj radi promjene
  4. Sprema promjene

### UC18 - Uređivanje korisničkih podataka

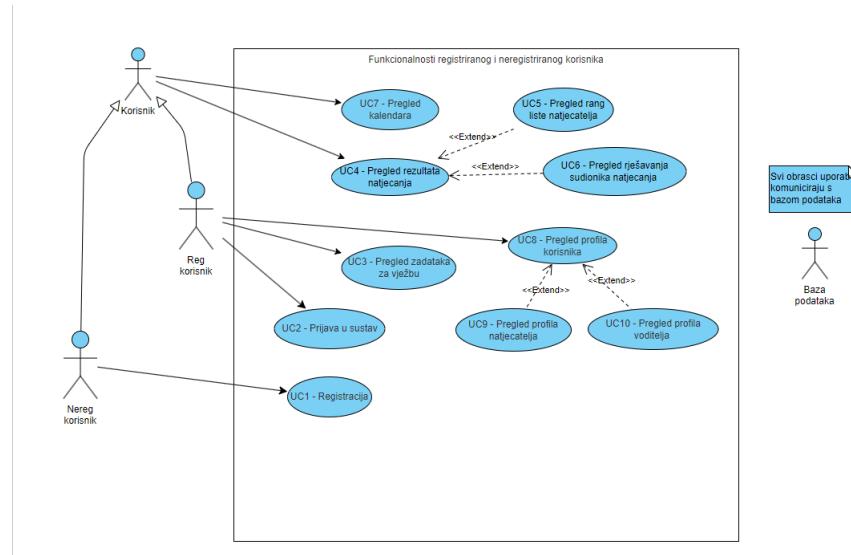
- **Glavni sudionik:** Administrator
- **Cilj:** Učinkovita administracija i održavanje sustava
- **Sudionici:** Baza podataka
- **Preduvjet:** korisnik je prijavljen kao administrator
- **Opis osnovnog tijeka:**
  1. UC8 - Pregled profila korisnika
  2. Administrator odabire opciju uredi korisnika
  3. Administrator mijenja podatke ili prava korisnika
  4. Sprema promjene
  5. Korisniku se šalje mail o promijeni njegovih podataka.

### UC19 - Uređivanje svih natjecanja/zadataka

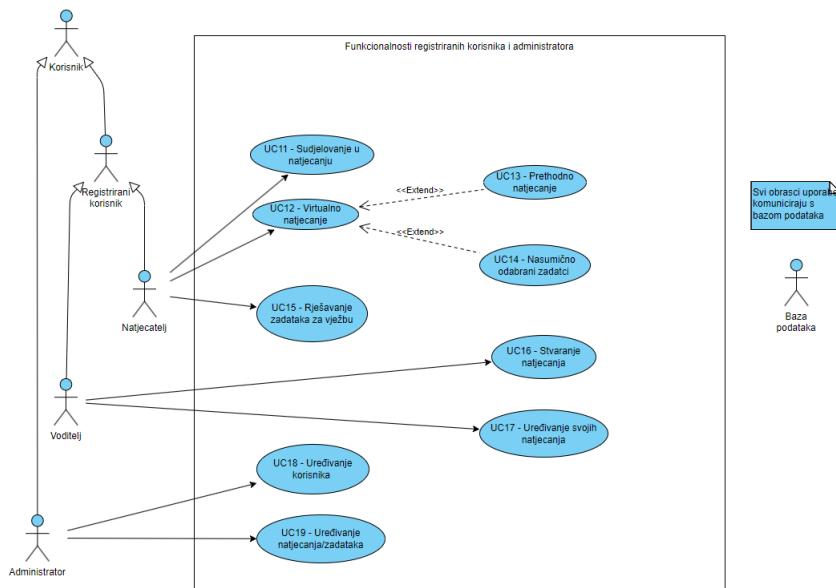
- **Glavni sudionik:** Administrator
- **Cilj:** Ispravljanje grešaka ili nesporazuma u natjecanju/zadatku
- **Sudionici:** Baza podataka

- **Preduvjet:** korisnik je prijavljen kao administrator
- **Opis osnovnog tijeka:**
  1. UC8 - Pregled profila voditelja
  2. Odabir polja uredi zadatak ili uredi natjecanje
  3. Administrator odabire zadatak/natjecanje
  4. Administrator uređuje zadatak/natjecanje
  5. Administrator sprema promjene

### Dijagrami obrazaca uporabe



Slika 3.1: Dijagram obrasca uporabe, funkcionalnost reg. i nereg. korisnika



Slika 3.2: Dijagram obrasca uporabe, funkcionalnost voditelja, natjecatelja i administratora

### 3.1.2 Sekvencijski dijagrami

#### 1) Registracija korisnika

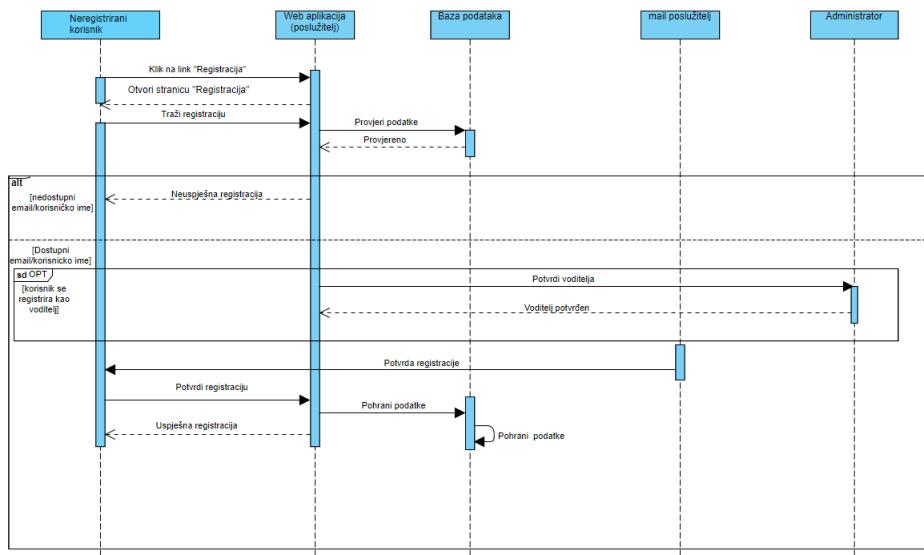
**Opis dijagrama:** Korisnik započinje registraciju odabirom opcije "Registracija" na korisničkom sučelju. Nakon što korisnik odabere tu opciju, poslužitelj aplikacije šalje zahtjev korisniku da unese sljedeće podatke:

- (a) korisničko ime
- (b) fotografiju
- (c) lozinku
- (d) ime i prezime
- (e) email adresu
- (f) odabir: voditelj natjecanja / natjecatelj

Nakon što korisnik unese navedene podatke, šalje zahtjev za registraciju putem korisničkog sučelja. Nakon toga, poslužitelj provjerava unesene podatke. Provjerava se dostupnost e-mail adrese i korisničkog imena u bazi podataka. U slučaju da su e-mail adresa ili korisničko ime već zauzeti, poslužitelj obavještava korisnika da registracija nije uspjela. U slučaju da su e-mail ili korisničko ime adresa dostupni, korisniku se šalje zahtjev za potvrdom registracije putem e-maila. Korisnik potvrđuje registraciju - podaci se trajno pohranjuju u bazu podataka, a poslužitelj obavještava korisnika da je registracija uspješna.

U slučaju da se korisnik registrira kao "voditelj natjecanja," tada administrator mora u aplikaciji potvrditi novog voditelja.

**Slika dijagrama:**

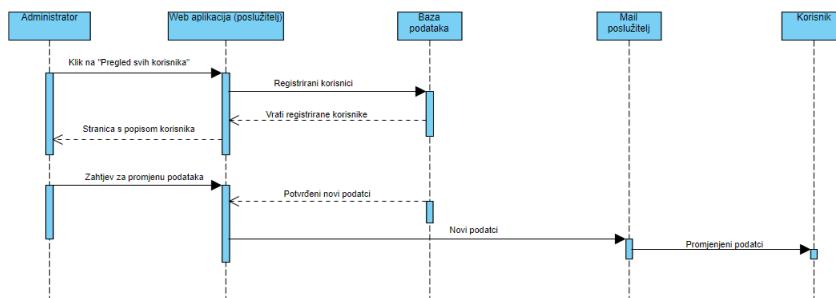


Slika 3.3: Sekvencijski dijagram registracije novog korisnika

## 2) Pregled i uređivanje korisnika

**Opis dijagrama:** Administrator šalje zahtjev za pregled korisnika i njihovih podataka tako da klikne na link "Pregled svih korisnika". Poslužitelj šalje upit bazi podataka za podatke registriranih korisnika, a baza podataka vraća podatke o registriranim korisnicima. Otvara se stranica s popisom registriranih korisnika i njihovim podacima koje je moguće urediti ili izbrisati. Kada administrator pošalje zahtjev za izmjenom podataka, podatci se uspješno pohranjuju u bazi podataka te se prikazuju na stranici.

**Slika dijagrama:**



Slika 3.4: Sekvencijski dijagram pregleda i uređivanja korisnika

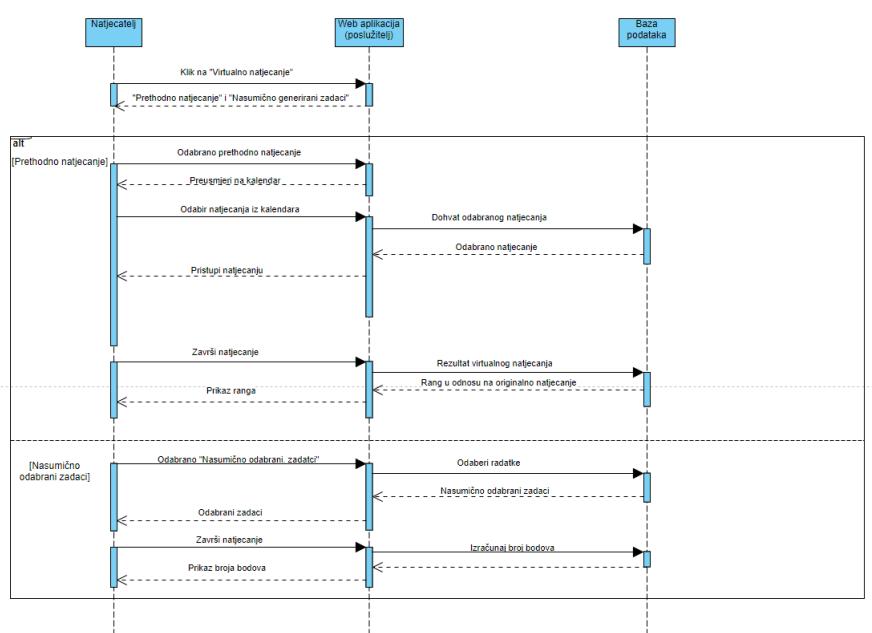
### 3) Virtualno natjecanje

**Opis dijagrama:** Kada natjecatelj klikne na link/gumb "Virtualno natjecanje", poslužitelj mu nudi mogućnost odabira između "Prethodno natjecanje" i "Nasumično odabrani zadaci".

Ako natjecatelj odabere "Prethodno natjecanje", poslužitelj ga preusmjerava na kalendar natjecanja, gdje natjecatelj može birati prethodna natjecanja.

Ako natjecatelj odabere opciju "Nasumično odabrani zadaci", poslužitelj iz baze podataka dohvaća zadatke i odabire ih ravnomjerno prema težini. Nakon odabiranja zadataka, zadaci se prikazuju natjecatelju.

**Slika dijagrama:**



Slika 3.5: Sekvencijski dijagram virtualnog natjecanja

## 3.2 Ostali zahtjevi

- Sustav treba biti funkcionalan na bilo kojem web pregledniku
- Dohvat zadataka ili korisnika iz baze podataka mora se obaviti u konačnom vremenu (manji od 60 sekundi)
- Sustav mora podržavati hrvatsku abecedu pri unosu i prikazu tekstualnog sadržaja
- Sustav mora podržavati rad jednog ili više korisnika istovremeno

- Pristup sustavu mora biti omogućen preko javne mreže pomoću HTTPS
- Nadogradnja sustava ne smije narušiti postojeće funkcionalnosti sustava
- Sustav podržava format slike jpeg (maksimalna veličina 1048576 bajtova)
- Sustav mora podržavati programska rješenja u jeziku Java.

## 4. Arhitektura i dizajn sustava

*Arhitektura ima 3 podsustava:*

- *Web preglednik*
- *Web aplikacija na web poslužitelju*
- *Baza podataka - PostgreSQL*

Web preglednik je ujedino i prevoditelj koda koji korisniku omogućuje pregleđavanje sadržaja web aplikacije i interakciju s istim.

Web poslužitelj prima HTTP (engl. Hyper Text Transfer Protocol) zahtjeve od klijenta (preglednika) koji sadrže informacije o tome što klijent traži, kao što su primjerice URL, GET, POST... i vraća dohvaćeni resurs ako ga ima te vraća statusni kod koji daje informacije o uspješnosti zahtjeva.

Tehnologije korištene u našoj aplikaciji jesu Spring Boot i React. Aplikacija se sastoji od serverske komponente napisane u Javi (Spring Boot) i klijentske komponente napisane u JavaScriptu (React). Za razvojno okruženje koristimo IntelliJ, a baza koju koristimo za spremanje podataka o registriranim korisnicima i sve informacije o natjecanjima je PostgreSQL.

Web aplikacija temelji se na arhitekturi Model-View-Controller (MVC). Ova arhitektura omogućuje organizaciju aplikacije u tri ključne komponente:

- **Model:** Ova komponenta predstavlja poslovnu logiku i podatke aplikacije. Model je implementiran u Java programskom jeziku (Spring Boot) i odgovoran je za upravljanje podacima, komunikaciju s bazom podataka te izračune i obrade podataka.
- **View:** Komponenta za prikaz podataka korisnicima. Prikazi se generiraju u Reactu, a omogućuju korisnicima interakciju s aplikacijom putem web preglednika. Prikazi se oblikuju pomoću HTML-a, CSS-a i JavaScripta.
- **Controller:** Kontroler je posrednik između Modela i Viewa. Ova komponenta upravlja korisničkim zahtjevima, prima ulazne podatke od korisnika te izvršava odgovarajuće akcije u Modelu. Kontroler također određuje koji

prikaz treba biti poslan korisnicima.

## 4.1 Baza podataka

Koristimo relacijsku bazu podataka čije su gradivne jedinke tablice definirane imenom i skupom atributa za jednostavno upravljanje podacima. Baza podataka ove aplikacije sastoji se od sljedećih entiteta:

- *Korisnik*
- *Natjecanje*
- *Zadatak*
- *Zadaci na natjecanju*
- *Primjeri za evaluaciju*
- *Slike*

### 4.1.1 Opis tablica

**Korisnik** - entitet sadržava sve važne informacije o korisniku: Korisničko ime, lozinku, ime, prezime, identifikator slike, email, tip korisnika (natjecatelj/voditelj), individualni sažetak, potvrda registracije, potvrda administratora. U vezi je One-To-One sa entitetom slika

Korisnik		
id	INT	jedinstveni identifikator korisnika
username	VARCHAR	izabrano korisničko ime
email	VARCHAR	e-mail adresa korisnika
image-id	INT	identifikator slike korisnika
confirmation-hash	VARCHAR	individualni sažetak za registraciju
name	VARCHAR	ime korisnika

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Korisnik		
lastname	VARCHAR	prezime korisnika
user-type	VARCHAR	natjecatelj ili voditelj natjecanja
password	VARCHAR	šifra korisnika
confirmed	BOOLEAN	potvrđena registracija preko korisničkog emaila
confirmed-by-admin	BOOLEAN	potvrđena registracija od strane administratora

**Natjecanje** - entitet sadržava sve važne informacije o natjecanju: Id natjecanja, vrijeme početka, vrijeme završetka, broj zadataka, id voditelja natjecanja, identifikator pehara, ime natjecanja i je li virtualno. Veze s drugim entitetima: Many-To-One s entitetom korisnik preko atributa tvorac natjecanja (competition-maker-id), One-To-One sa entitetom slika, Many-to-Many s entitetom problema.

Natjecanje		
id	INT	jedinstveni identifikator natjecanja
image-id	INT	identifikator slike pehara
date-time-of-beginning	TIMESTAMP	vrijeme početka natjecanja
date-time-of-ending	TIMESTAMP	vrijeme završetka natjecanja
competition-maker-id	INT	id voditelja natjecanja
number-of-problems	INT	broj zadataka u natjecanju
name	VARCHAR	ime natjecanja
isVirtual	BOOLEAN	virtualno natjecanja

**Zadatak** - entitet sadržava sve važne informacije o zadatku. Sadrži atribute id, trajanje zadatka, booleanski atribut is-private, broj bodova koje je moguće ostvariti,

tip problema (težinu zadatka), tekst zadatka, naslov i id korisnika koji je napravio zadatak. S entitetom Korisnik je u odnosu Many-To-One preko atributa problem-maker-id te u odnosi Many-to-Many s entitetom natjecanje.

Zadatak		
id	INT	jedinstveni identifikator zadatka
problem-maker-id	INT	identifikator vlasnika zadatka
duration	NUMERIC	trajanje zadatka
is-private	BOOLEAN	provjerava je li zadatak objavljen
problem-type	INT	težina zadatka
text	VARCHAR	tekst zadatka
title	NUMERIC	naslov zadatka
points	INT	broj bodova zadatka

**Zadaci na natjecanju** - entitet sadržava informacije o tome koje natjecanje ima koje zadatke. Atributi: id natjecanja i id zadatka. On predstavlja vezu Many-To-Many natjecanja i zadatka

Zadaci na natjecanju		
competition-id	INT	identifikator natjecanja (natjecanje.id)
problem-id	INT	identifikator zadatka (problem.id)

**Primjeri za evaluaciju** - entitet sadržava id zadatka te vrijednost i ključ, odnosno na ovaj način se mapiraju rješenja zadataka i služi za evaluaciju rješenja korisnika.

Primjeri za evaluaciju		
problem-id	INT	identifikator zadatka
vrijednost	VARCHAR	očekivani izlaz programa

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Primjeri za evaluaciju		
ključ	VARCHAR	ulaz programa

**Slike** - entitet sadržava id slike te bajt zapis slike. Na ovaj način spremamo slike korisnika i pehara.

Slike		
id	INT	jedinstveni identifikator slike
data	BYTEA	binarni kod slike

**Tablica sudionika natjecanja** - entitet sadržava id korisnika i id natjecatelja te informaciju do kada korisnik može rješavati natjecanje. Ova tablica predstavlja Many-To-Many vezu natjecanja i korisnika.

Tablica sudionika natjecanja		
competition_id	INT	identifikator natjecanja (natjecanje.id)
user_id	INT	identifikator korisnika
access_time	TIMESTAMP	rok završetka rješavanja natjecanja za korisnika

**Predaje natjecanja** - entitet predstavlja zapis već predanog natjecanja. Sadržava jedinstveni id predanog natjecanja, korisničko ime, id natjecanja i broj bodova koje je korisnik ostvario. Ova tablica je u odnosu Many-to-One s natjecanjem preko njegovog id-a te također u odnosu Many-to-One s entitetom korisnika. Na temelju predanih natjecanja SQL naredbama direktno se generira rang.

Predaje natjecanja		
id	INT	jedinstveni identifikator predanog natjecanja
points	NUMERIC	broj ostvarenih bodova
competition_id	INT	identifikator natjecanja (natjecanje.id)
username	VARCHAR	korisničko ime

**Plasman** - entitet sadržava id natjecatelja, plasman te id natjecanja na kojem je prisustvovao. Ova tablica je u odnosu Many-To-Many s korisnikom te u odnosu Many-to-Many s natjecanjem. Ovaj entitet pomaže nagrađivanju te formiranju korisnikove statistike koja se prikazuje na njegovom profilu.

<b>Plasman</b>		
id natjecatelja	INT	identifikator natjecatelja
competition_id	INT	identifikator natjecanja
placement	INT	plasman

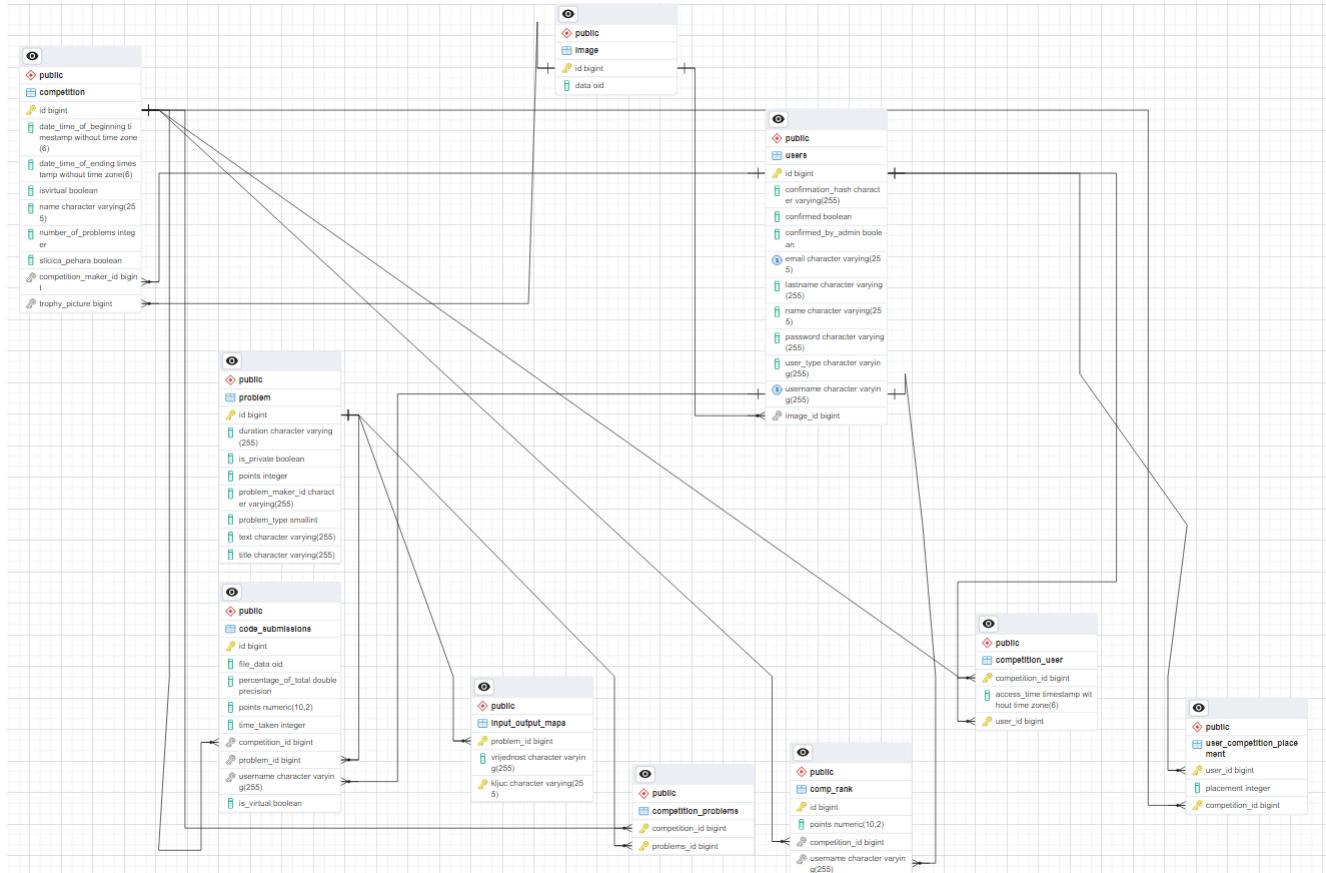
**Učitana programska rješenja** - entitet sadržava jedinstven identifikator rješenja, bitove pohranjenog rješenja, vrijeme od početka rješavanja do predaje, korisničko ime, identifikator zadatka, ostvarene bodove, identifikator natjecanja, postotak točnosti i atribut is\\_virtual koji govori radi li se o predaji koja je s virtualnog ili običnog natjecanja. Ova tablica je u odnosu Many-To-One s entitetima natjecanja, korisnika i problema.

<b>Učitana programska rješenja</b>		
id	INT	jedinstven identifikator
file_data	BYTEA	binarni kod rješenja
time_taken	INT	vrijeme od početka rješavanja do predaje
problem_id	INT	identifikator zadatka
username	VARCHAR	izabrano korisničko ime
points	NUMERIC	ostvareni bodovi
competition_id	INT	identifikator natjecanja (natjecanje.id)
percentage_of_total	DOUBLE	postotak točnosti
is_virtual	BOOLEAN	virtualno učitanje

#### 4.1.2 Dijagram baze podataka

ER dijagram baze podataka prikazan na slici 4.1 sadrži vizualne slike ključeva: primarni ključ prikazan je ikonom ključa žute boje, strani ključ sivom bojom, a

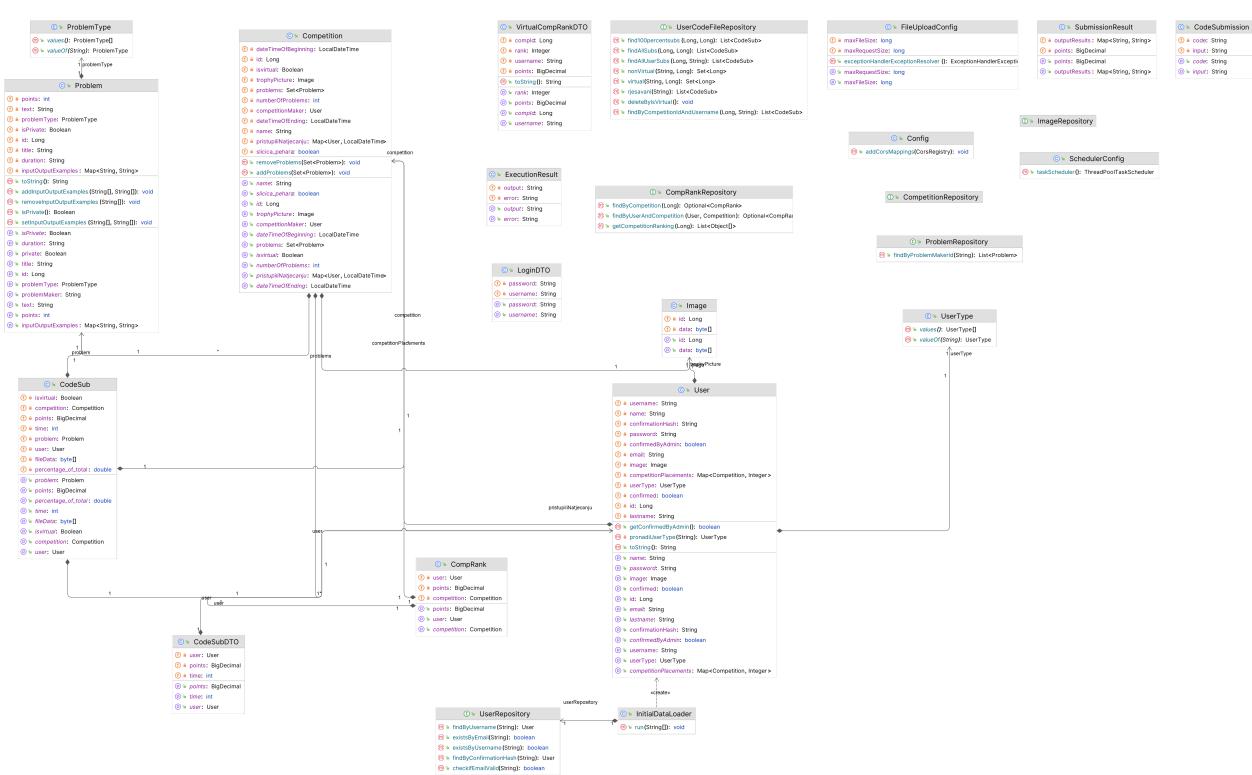
atributi označeni brojem 1 su alternativni ključevi. Prvi redak svake tablice entiteta predstavlja vidljivost tablice u bazi podataka, u drugom se retku nalazi ime tablice, a u preostalim retcima su ispisani atributi.



Slika 4.1: Er dijagram baze podataka

## 4.2 Dijagram razreda

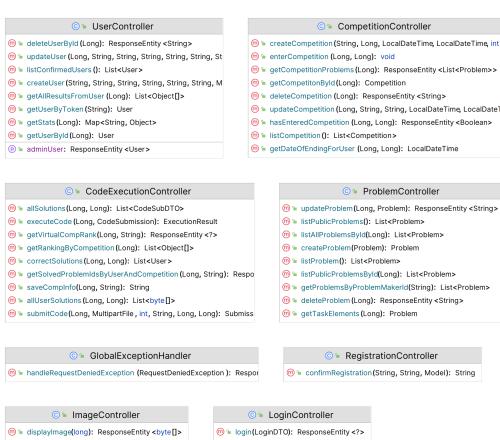
U nastavku su prikazani razredi na kojima je zasnovan backend aplikacije. Lako je uočljivo da većina njih u nazivu sadrži naziv entiteta iz baze, te riječi Service i Controller.



Slika 4.2: Dijagram razreda: razredi koji predstavljaju strukturu baze

Razredi User, Competition, Image, CodeSub i Problem sa slike 4.2 preslikavaju atribute svakog od entiteta iz baze. Razred User predstavlja registriranog korisnika, čiji UserType može biti COMPETITOR, COMPETITION LEADER i ADMIN (samo jedan, glavni korisnik), a koji pri registraciji unosi podatke prikazane na dijagramu (koji su vlastiti atributi razreda User). Razred Problem predstavlja zadatak koji je korisnik s tipom voditelj unio u sustav: na dijagramu se vidi ta poveznica. Jedna ili više instanci razreda Problem je dio Natjecanja tj. razreda Competition. Jedan korisnik može sudjelovati na više natjecanja, kao što i na jednom natjecanju sudjeluje jedan ili više korisnika. Tu je i razred Image koji služi za pohranu slike u bazu podataka, te razred CodeSub koji prati predaju rješenja zadataka te njihovo bodovanje.

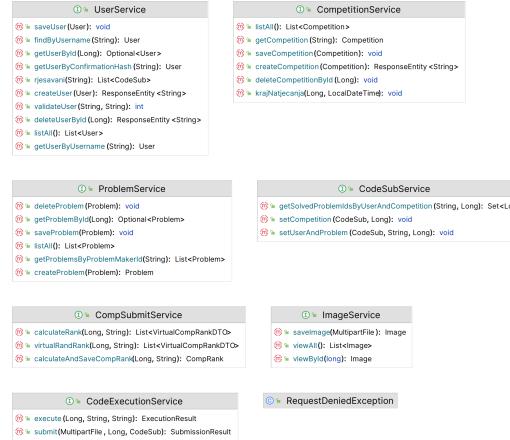
Razredi imena Service (slika 4.4) služe za baratanje objektima razreda User,



Slika 4.3: Dijagram razreda: razredi Controller

Competition itd. Razredi imena Controller (slika 4.3) obrađuju HTTP zahtjeve poslane na server i vraćaju zatražene podatke u .json obliku.

Strelice sa slike na jednostavan i intuitivan način prikazuju poveznice između razreda User, UserController i UserService. Metode koje su u razredima implementirane služe za baratanje entitetima u bazi: primarno su to razredi koji u svome nazivu sadrže 'Service' kao što je prije rečeno. Razred UserController oslanja se na ta dva prethodno navedena razreda da bi mogao ispuniti zaprimljene HTTP zahtjeve, bilo da je to samo informacija o korisnicima ili izmjena podataka. Prikazan je i TokenService razred koji omogućava autentifikaciju korisnika, a time sadrži i informacije o njegovim ovlastima u aplikaciji.



Slika 4.4: Dijagram razreda: razredi Service

## 4.3 Dijagram stanja

UML-dijagram stanja je ponašajni UML-dijagram kojim se prikazuje diskretno ponašanje objekta ili sustava putem prelazaka između konačnog broja stanja, a često se za cjelokupni model stanja i prijelaza između stanja koristi izraz stroj stanja. Ova vrsta dijagrama se koristi za modeliranje ponašanja entiteta tijekom vremena, naglašavajući odgovor na dogadaje i okidače. Na dijagramu stanja, slika 4.5 prikazane su sve funkcionalnosti kojima natjecatelj može pristupiti.

Proces kreće prijavom korisnika ulogom natjecatelja. Na početnoj stranici natjecatelj može pristupiti aktivnom natjecanju, otići na kalendar natjecanja, pregled profila korisnika, stranicu za vježbu ili stranicu s rezultatima. U bilo kojem slučaju korisnik može doći u stanje "Početna stranica", "Kalendar natjecanja", "Korisnici", "Vježba", "Rezultati". Iz svih se stanja može odjaviti.

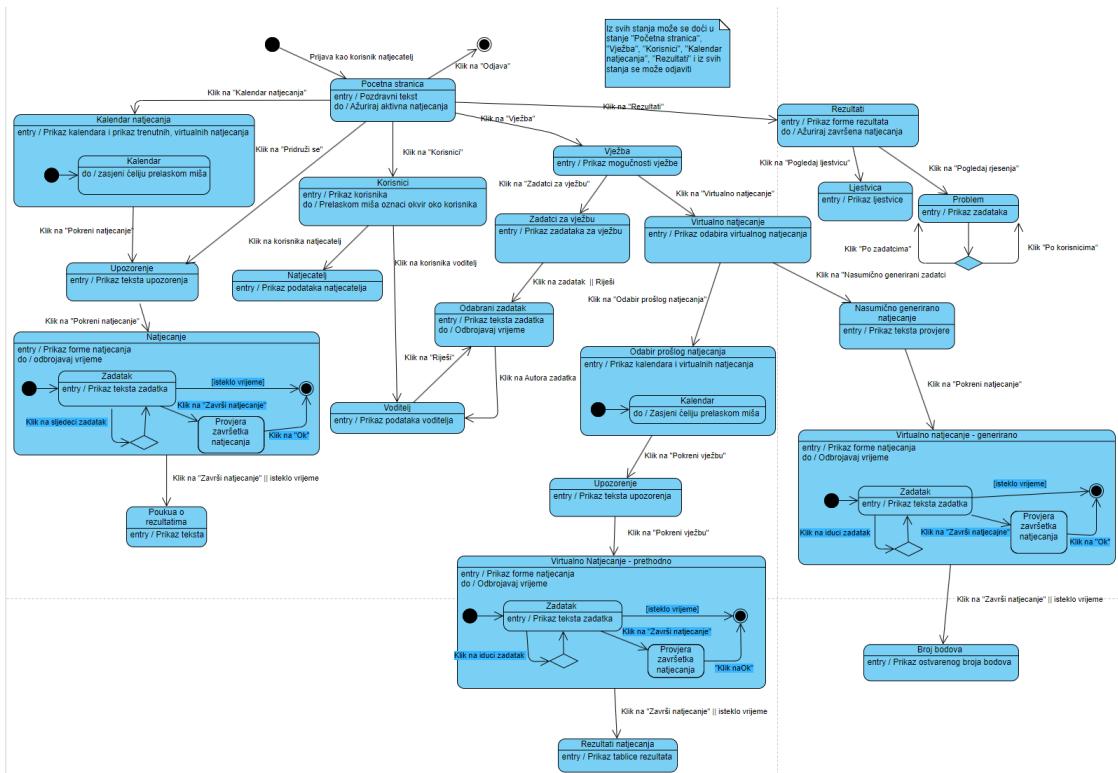
Odabirom kalendarstranica preusmjerava na kalendar s prikazanim natjecanjima, ukoliko su aktivni. U slučaju aktivnog natjecanja, natjecatelj može pristupiti natjecanju odabirom gumba "Pokreni natjecanje". Tim odabirom javlja se poruka upozorenja koja naglašava kako se korisnik ne bi smio koristiti nedopuštenim sredstvima. Odabirom gumba "Pokreni natjecanje", natjecanje se aktivira te počinje odbrojavanje vremena. Korisnik može vidjeti tekst svakog zadatka, preći na sljedeći zadatak, testirati rješenje, učitati svoj kod te završiti natjecanje. Završetkom natjecanja natjecatelj vidi rang listu svih natjecatelja koji su pristupili tom natjecanju na stranici rezultati.

Odabirom poveznice Korisnici, stranica prikazuje profile svih registriranih korisnika (potvrđenih od administratora). Klikom na bilo kojeg korisnika prikazuju se podatci o njegovom profilu. Za natjecatelja se prikazuju podatci o broju točno riješenih zadataka, broju isprobanih zadataka te su za sva natjecanja na kojima je imao dobar plasman iscrtani pehari. S druge strane, profili voditelja sadrže popis učitanih zadataka s mogućnošću sortiranja i kalendar s popisom objavljenih natjecanja. Odabirom na zadatak pojedinog voditelja, moguće je vidjeti i rješiti taj zadatak.

Odabirom poveznice Vježba stranica nudi dva načina vježbe: Virtualno natjecanje i Zadatci za vježbu. Odabirom zadataka za vježbu prikazuju se javno objavljeni zadatci svih voditelja. Klikom na ime zadatka ili na gumb Riješi započinje odbrojavanje vremena te natjecatelj može pokušati rješiti zadatak i testirati točnost svog koda. Odabirom virtualnog natjecanja stranica nudi dvije opcije:

odabir pošlog natjecanja ili nasumično generiranog natjecanja. U slučaju nasumično generiranog natjecanja stranica poručuje kako je vrijeme ograničeno te klikom na pokreni natjecanje, natjecatelj započinje natjecanje. Odabirom prošlog natjecanja, prikazuje se kalendar u kojem korisnik može odabrati prijašnje natjecanje. Postupak provedbe natjecanja za oba slučaja virtualnog natjecanja je isti kao i za normalno natjecanje.

Proces završava odjavom natjecatelja.



Slika 4.5: Dijagram stanja

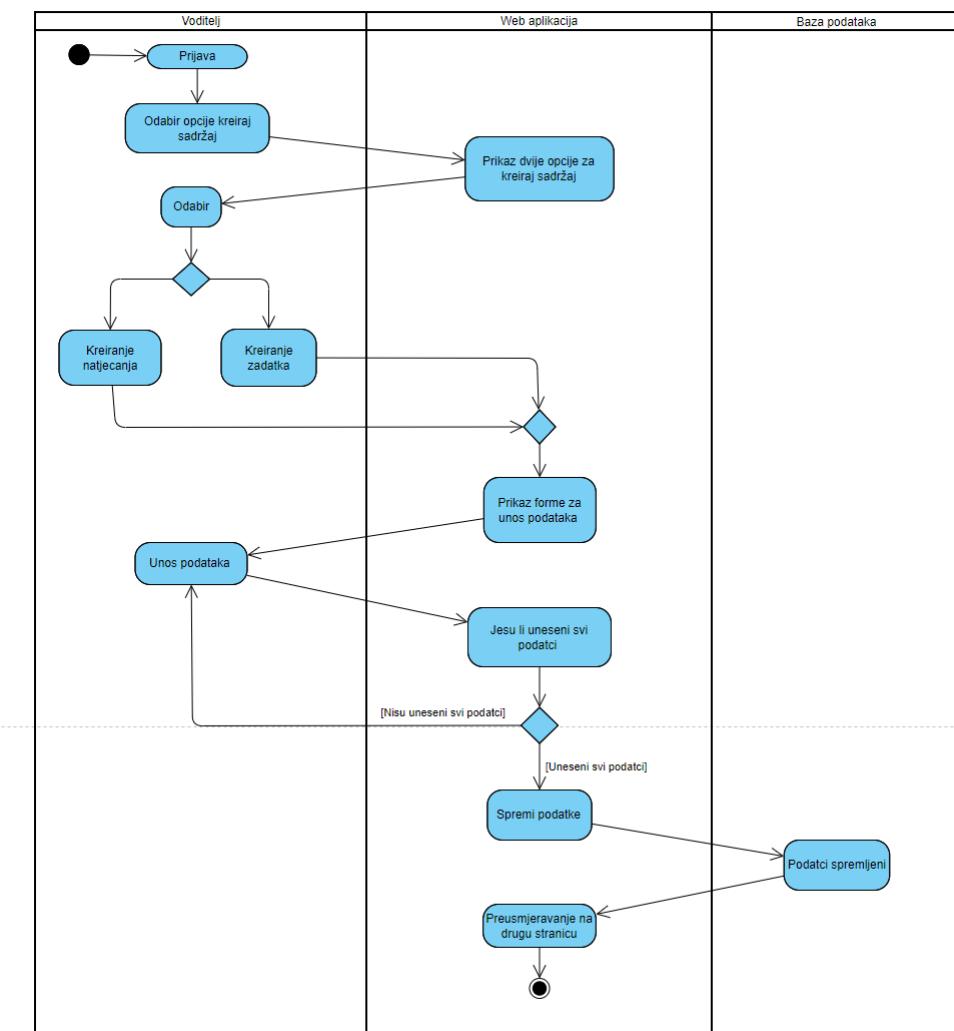
## 4.4 Dijagram aktivnosti

Dijagram aktivnosti je vrsta UML-dijagrama koja se u programskom inženjerstvu koristi za modeliranje i grafički prikaz dinamičkog ponašanja sustava. Na njemu se prikazuje izvođenje aktivnosti kroz niz akcija koje čine upravljačke tokove i tokove objekata, s naglaskom na slijed i uvjete toka. Na dijagramu aktivnosti (slika 4.6) prikazan je proces kojim voditelj stvara zadatku za vježbu ili natjecanje.

Proces započinje prijavom korisnika u sustav kao voditelj. Na početnoj stranici, klikom na "Kreiraj sadržaj", web aplikacija prikazuje 2 moguća odabira: kreiranje zadatka ili kreiranje natjecanja. Voditelj odabire što želi kreirati, te nakon odabira

web aplikacija prikazuje formu za unos potrebnih podataka. Forma za stvaranje zadataka razlikuje se od forme za kreiranje natjecanja, ali bez obzira na odabранo, web aplikacija izvršava isti postupak.

Nakon što voditelj klikne na gumb Izradi, web aplikacija provjerava jesu li uneseni svi podaci. Ako nisu, web aplikacija ne dopušta stvaranje te javlja voditelju da mora dopuniti izostavljene podatke. Kada su svi podaci uneseni, web aplikacija sprema kreirani sadržaj u bazu podataka te preusmjerava korisnika na drugu stranicu.

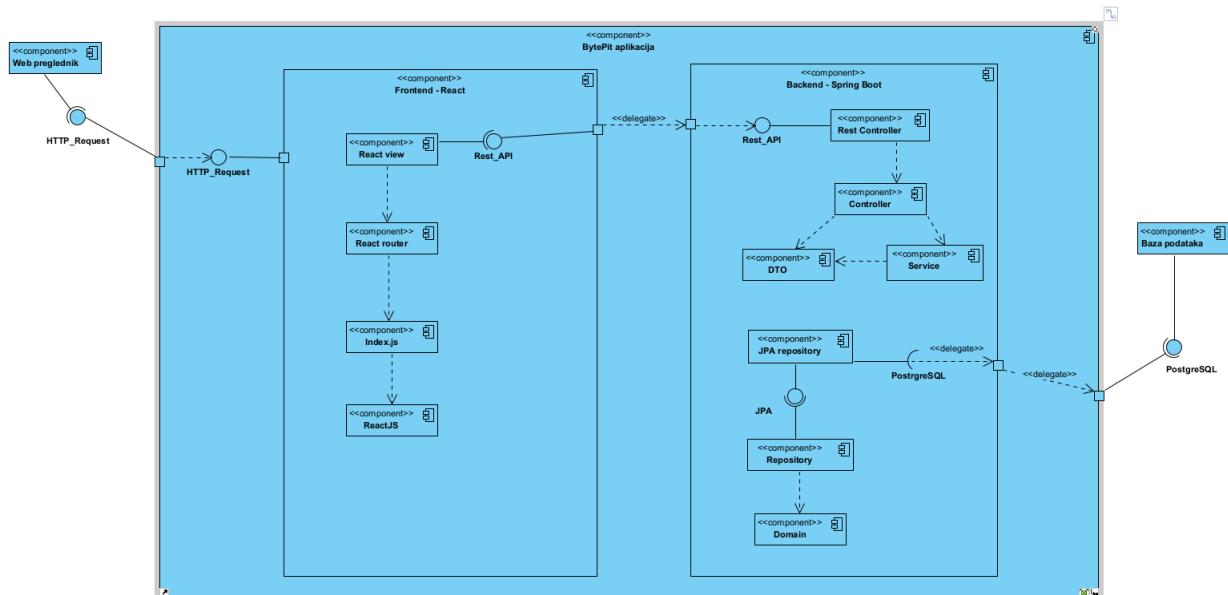


Slika 4.6: Dijagram aktivnosti

## 4.5 Dijagram komponenti

UML-dijagrami komponenti su vrsta strukturnih UML-dijagrama koji prikazuju organizaciju i odnose komponenti koji čine programsku potporu. Pružaju vizualni prikaz arhitekture sustava, naglašavajući modularnu strukturu i interakcije između komponenti. Dijagram komponenti na slici 4.7 opisuje organizaciju i međuvisnost komponenti, interne strukture i odnose prema okolini.

U web aplikaciji BytePit postoje dvije velike komponente: Frontend - React i Backend - Spring Boot. Te dvije komponente komuniciraju preko sučelja REST\_API koje je zaduženo za komunikaciju frontend dijela aplikacije s backend dijelom aplikacije. U React komponenti sučelje REST\_API spaja se na komponentu React view koja komunicira s ostalim Frontend komponentama. Sve frontend komponente ovise o komponenti ReactJS koja je zadužena za dohvaćanje React biblioteka. S druge strane, u komponenti Spring Boot REST\_API je spojeno na komponentu Rest Controller koji je glavna komponenta za backend dio aplikacije. Sučelje JPA s vezom "ball and soccet" povezuje komponente Repository i JPA repository, JPA repository omogućuje komunikaciju s bazom podataka. Web aplikacija se spaja na vanjske komponente: web preglednik preko sučelja HTTP\_Request koji podržava HTTP protokol i sve njegove operacije (Get, Post, Put, Delete), također na bazu podataka preko sučelja postgresSQL.



Slika 4.7: Dijagram komponenti

# 5. Implementacija i korisničko sučelje

## 5.1 Korištene tehnologije i alati

U nastavku su navedene tehnologije i alati koje smo koristili za izradu dokumentacije i aplikacije.

- **PgAdmin**<sup>1</sup> je open-source alat za upravljanje bazama podataka PostgreSQL. Koristi se za sve osnovne operacije na bazi podataka, kao što su kreiranje, ažuriranje i brisanje tablica te postavljanje SQL upita. Ovaj alata omogućuje i napredne operacije, poput izvoza i uvoza podataka, upravljanja sigurnosnim ovlastima te optimizacije performansi.
- **React**<sup>2</sup> je JavaScript biblioteka za izradu web sučelja (frontend razvoj). Koristi se za izgradnju dinamičnih i interaktivnih web stranica i aplikacija koje reagiraju na korisničke interakcije. React je jednostavan za učenje i korištenje, a također je vrlo fleksibilan.
- **Spring Boot**<sup>3</sup> je open-source framework za Java backend razvoj. Koristi se za izgradnju RESTful web servisa i mikroservisa, kao i za automatizaciju mnogih zadataka koji su inače potrebni za izgradnju web servisa (kao što su konfiguracija, upravljanje ovisnostima i sigurnost). Spring Boot je iznimno popularan zbog svoje jednostavnosti i brzine.
- **GitHub**<sup>4</sup> je web platforma za upravljanje izvornim kodom. Koristi se za spremanje, dijeljenje i suradnju na kodu.
- **WhatsApp**<sup>5</sup> je popularna mobilna aplikacija za razmjenu poruka. Koristi se za komunikaciju unutar tima pomoću tekstualnih, glasovnih ili video poruka. WhatsApp je jednostavna moderna aplikacija koja na pouzdan način omogućava komunikaciju u realnom vremenu.

<sup>1</sup><https://www.pgadmin.org/>

<sup>2</sup><https://reactjs.org/>

<sup>3</sup><https://spring.io/projects/spring-boot>

<sup>4</sup><https://github.com/>

<sup>5</sup><https://www.whatsapp.com/>

- **Visual Paradigm Online**<sup>6</sup> je online alat za izradu UML dijagrama. Koristi se za modeliranje softverskih i drugih sustava. Ovaj je alat jednostavan za korištenje i ima široku paletu funkcionalnosti.
- **Latex**<sup>7</sup> je programski jezik za pisanje strukturiranih tekstova i njihov automatski slog i prijelom u dokumente profesionalne kvalitete spremne za tisk. Omogućuje precizno kontroliranje izgleda dokumenta, uključujući veličinu i oblik slova, razmake, margine i druge elemente. LaTeX je popularan među znanstvenicima i inženjerima za pisanje tehničkih dokumenata.

### Zaključak

U izradi dokumentacije i aplikacije korištene su suvremene tehnologije i alati koji su omogućili izgradnju kvalitetnog i funkcionalnog proizvoda.

## 5.2 Ispitivanje programskog rješenja

### 5.2.1 Ispitivanje komponenti

#### Test 1: Osnovne funkcije razreda User.

U ovom testu provjeravamo hoće li naš razred u konstruktoru dobro "zapamtitи" vrijednosti parametara, koje kasnije dohvaćamo get metodama. Primjer korištenja je pri registraciji.

---

```
@Test
void getterTest(){
    User user = new User("korisnik1", "ime", "prezime",
        "lozinka", "ime.prezime@gmail.com", null, "COMPETITOR");
    assertEquals("korisnik1", user.getUsername());
    assertEquals("ime", user.getName());
    assertEquals("prezime", user.getLastname());
    assertEquals("lozinka", user.getPassword());
    assertEquals("ime.prezime@gmail.com", user.getEmail());
    assertEquals(null, user.getImage());
    assertEquals(UserType.COMPETITOR, user.getUserType());
}
```

---

#### Test 2: Osnove funkcije razreda Problem i Competition

<sup>6</sup><https://online.visual-paradigm.com/>

<sup>7</sup><https://www.latex-project.org/>

U ovom primjeru provjeravamo možemo li stvoriti novi problem (zadatak), novi competition (natjecanje) te možemo li dodati problem u competition. Primjer korištenja je pri stvaranju natjecanja, odabiremo koji zadatci će biti dio tog natjecanja.

---

```
@Test
void competitionGetProblemsTest(){
    Map<String, String> mapa = new HashMap<>();
    mapa.put("ulaz", "izlaz");

    Problem problem = new Problem("1", "naslov", 5, "10:00",
        "text", mapa, false, ProblemType.HARD);
    User user = new User("korisnik1", "ime", "prezime",
        "lozinka", "ime.prezime@gmail.com", null,
        "COMPETITION_LEADER");
    Set<Problem> set = new HashSet<>();
    set.add(problem);
    Competition competition = new Competition(user,
        LocalDateTime.now(), LocalDateTime.now(), set, null,
        false, false);
    assertEquals(set, competition.getProblems());
}
```

---

### Test 3: Provjera sličice pehara.

U ovom primjeru provjeravamo slučaj kada neko natjecanje nema sličicu trofeja. Primjer korištenja je pri prikaza statistike natjecatelja. U slučaju da nema sličice pehara, prikazuje se "default" ikona pehara.

---

```
@Test
void competitionNoTrophyPictureThrowsTest(){
    Map<String, String> mapa = new HashMap<>();
    mapa.put("ulaz", "izlaz");

    Problem problem = new Problem("1", "naslov", 5, "10:00",
        "text", mapa, false, ProblemType.HARD);
    User user = new User("korisnik1", "ime", "prezime",
        "lozinka", "ime.prezime@gmail.com", null,
        "COMPETITION_LEADER");
    Set<Problem> set = new HashSet<>();
```

```
    set.add(problem);
    Competition competition = new Competition(user,
        LocalDateTime.now(), LocalDateTime.now(), set, null,
        false, false);
    assertThrows(NullPointerException.class, () ->
        competition.getTrophyPicture().getData());
}
```

---

#### Test 4: Provjera kompleksnijih funkcija sustava.

U ovom primjeru provjeravamo može li baza podataka dohvatiti nekog korisnika.

Provjeru radimo uspoređivanjem savedUser.id (nalazi se u bazi podataka) i jsonNode.get("id"). JsonNode je objekt koji predstavlja odgovor na naš get zahtjev na server, ta metoda bi trebala iz baze podataka dohvatiti korisnika s id-em.

Nakon dohvata korisnika provjeravamo je li get metoda vratila točan podatak.

---

```
@Test
@DirtiesContext
void testGetUserById() throws Exception{
    User user = new User("korisnik1", "ime", "prezime",
        "lozinka", "ime.prezime@gmail.com", null,
        "COMPETITION_LEADER");
    userRepository.delete(user);
    User savedUser = userRepository.save(user);
    MvcResult result = mockMvc.perform(get("/users/byId/{id}",
        savedUser.getId()))
        .andExpect(status().isOk())
        .andReturn();
    userRepository.deleteById(savedUser.getId());

    ObjectMapper objectMapper = new ObjectMapper();
    JsonNode jsonNode =
        objectMapper.readTree(result.getResponse().getContentAsString());

    assertEquals(savedUser.getId(), jsonNode.get("id").asLong());
}
```

---

#### Test 5: Provjera put metode pri stvaranju korisnika.

U ovom primjeru provjeravamo, slično kao u prošlom primjeru, hoće li get

metoda vratiti dobrog korisnika. Korisnika ne upisujemo direktno u bazu v

---

```
@Test
@DirtiesContext
void createUserTest() throws Exception{
    MockMultipartFile file = new MockMultipartFile(
        "image",           // Parametar "image" iz metode
        "filename.jpg",   // Naziv datoteke
        MediaType.IMAGE_JPEG_VALUE, // Tip datoteke
        "file content".getBytes()
    );

    MvcResult resultFromCreateUser =
        mockMvc.perform(MockMvcRequestBuilders.multipart("/users")
            .file(file)
            .param("name", "ime2")
            .param("lastname", "prezime2")
            .param("username", "korisnik2")
            .param("email", "ime.prezime2@gmail.com")
            .param("password", "lozinka2")
            .param("userType", "COMPETITION_LEADER"))
            .andExpect(status().isOk())
            .andReturn();

    User userFromDatabase =
        userRepository.findByUsername("korisnik2");

    MvcResult resultFrom GetUser =
        mockMvc.perform(get("/users/byId/{id}",
            userFromDatabase.getId()))
            .andExpect(status().isOk())
            .andReturn();

    ObjectMapper objectMapper = new ObjectMapper();
    JsonNode jsonNode =
        objectMapper.readTree(resultFrom GetUser.getResponse().getContentAsString());
```

```
    userRepository.delete(userFromDatabase);
    assertEquals(userFromDatabase.getId(),
        jsonNode.get("id").asLong());
}
```

---

#### Test 6: Provjera uvjete za stvaranje korisnika.

U ovom primjeru pokušavamo stvoriti dva korisnika s istim korisničkim imenom (što narušava integritet baze podataka) te provjeravamo hoće li nam server to dopustiti ili vratiti pogrešku. Primjer korištenja je pri prijavi korisnika.

---

```
@Test
void uniqueUsernameTest() throws Exception{
    MockMultipartFile file = new MockMultipartFile(
        "image",           // Parametar "image" iz metode
        "filename.jpg",   // Naziv datoteke
        MediaType.IMAGE_JPEG_VALUE, // Tip datoteke
        "file content".getBytes()
    );

    MvcResult resultFromCreateUser =
        mockMvc.perform(MockMvcRequestBuilders.multipart("/users")
            .file(file)
            .param("name", "ime3")
            .param("lastname", "prezime3")
            .param("username", "korisnik3")
            .param("email", "ime.prezime3@gmail.com")
            .param("password", "lozinka3")
            .param("userType", "COMPETITION_LEADER"))
            .andExpect(status().isOk())
            .andReturn();
    MvcResult resultFromCreateUser2 =
        mockMvc.perform(MockMvcRequestBuilders.multipart("/users")
            .file(file)
            .param("name", "ime3")
            .param("lastname", "prezime3")
            .param("username", "korisnik3")
            .param("email", "ime.prezime4@gmail.com"))
```

```

    .param("password", "lozinka3")
    .param("userType", "COMPETITION_LEADER"))
    .andReturn();
ObjectMapper objectMapper = new ObjectMapper();
User userKojegBrisem =
    userRepository.findByUsername("korisnik3");
userRepository.deleteById(userKojegBrisem.getId());
assertTrue(resultFromCreateUser2.getResponse().getStatus() ==
400);

}

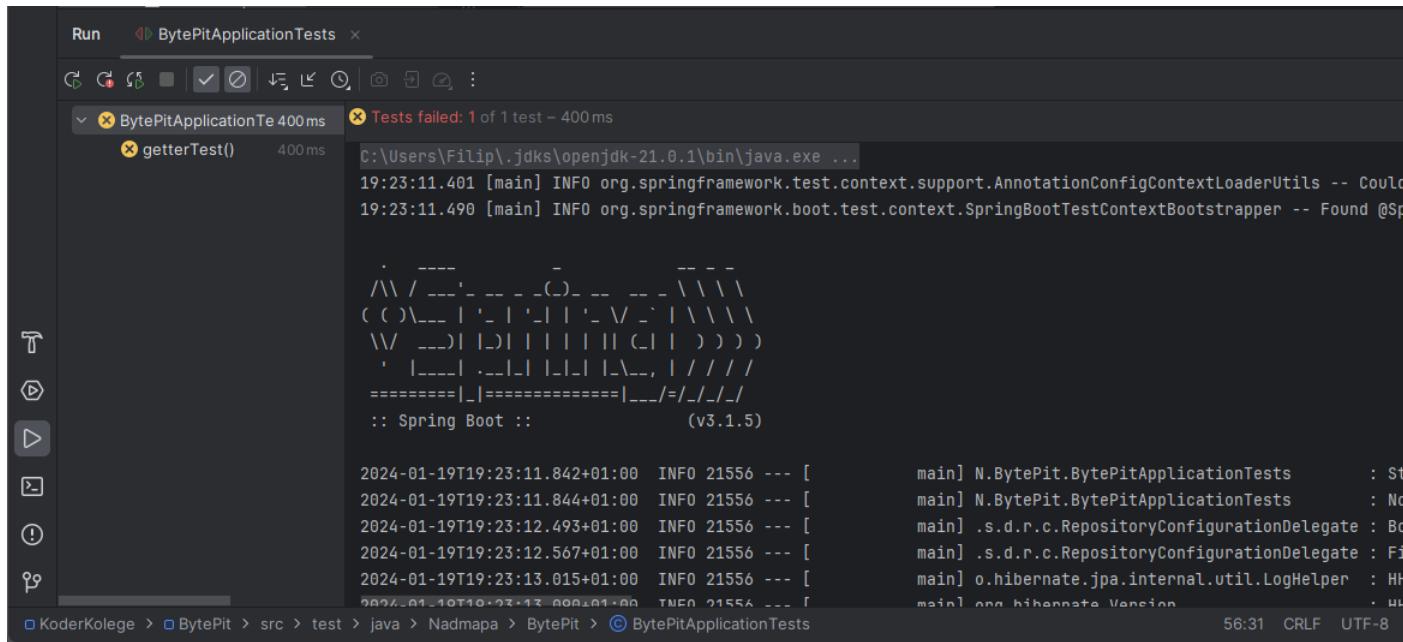
```

Primjer svih uspješno izvršenih testova:

The screenshot shows a Java IDE interface with a 'Run' tab selected. Under the 'Run' tab, there is a tree view of test classes and methods. A green checkmark indicates that all 7 tests have passed. The total execution time is 2 seconds and 881 milliseconds. The log pane below shows the command used to run the tests (java.exe), the Spring framework logs, and the output of the tests themselves. The output includes ASCII art representing a spring boot logo and various log entries from the application.

Slika 5.1: Uspješno izvedeni testovi

Uzmimo za primjer Test 1, u slučaju da želimo simulirati pogrešno izršen test, umjesto koriskičkog imena "korisnik1" postat će "korisnik" Primjer neuspješno izvršenog testa:



Slika 5.2: Ne uspješno izvedeni testovi

```

@Test
void getterTest(){
    User user = new User("korisnik1", "ime", "prezime",
        "lozinka", "ime.prezime@gmail.com", null, "COMPETITOR");
    assertEquals("korisnik", user.getUsername());
    assertEquals("ime", user.getName());
    assertEquals("prezime", user.getLastname());
    assertEquals("lozinka", user.getPassword());
    assertEquals("ime.prezime@gmail.com", user.getEmail());
    assertEquals(null, user.getImage());
    assertEquals(UserType.COMPETITOR, user.getUserType());
}

```

## 5.2.2 Ispitivanje sustava

### Test 1: Ispravan Login

**Ulaz:** "fm1234" "1234" **Očekivani izlaz:** preusmjeravanje na početnu stranicu

Command	Target	Value
1 ✓ open	2570:1400	
2 ✓ set window size	BytePit-PRIVACY	
3 ✓ click	css=AutocompleteInput + input	
4 ✓ type	css=AutocompleteInput + input	fm
5 ✓ click	css=AutocompleteInput + input	
6 ✓ type	css=AutocompleteInput + input	1234
7 ✓ click	css=AutocompleteInput + input	
8 ✓ click	css=submitForm	

Log - Rekord: 1  
1 click on css=AutocompleteInput OK  
2 type on css=AutocompleteInput + input OK  
3 type on css=AutocompleteInput + input with value fm OK  
4 click on css=AutocompleteInput + input OK  
5 type on css=AutocompleteInput + input with value 1234 OK  
6 click on css=AutocompleteInput + input OK  
7 type on css=AutocompleteInput + input with value 1234 OK  
8 click on css=submitForm OK  
\*Logout Log completed successfully

Slika 5.3: Ispravan Login

## Test 2: Neispravan Login

**Ulaz:** "fm" "1234" **Očekivani izlaz:** neispravno korisnicko ime ili lozinka.

Command	Target	Value
1 ✓ open	1000x1200	
2 ✓ set window size	BytePit-PRIVACY	
3 ✓ click	css=AutocompleteInput + input	fm
4 ✓ type	css=AutocompleteInput + input	
5 ✓ click	css=AutocompleteInput + input	
6 ✓ type	css=AutocompleteInput + input	1234
7 ✓ click	css=AutocompleteInput + input	
8 ✓ click	css=submitForm	

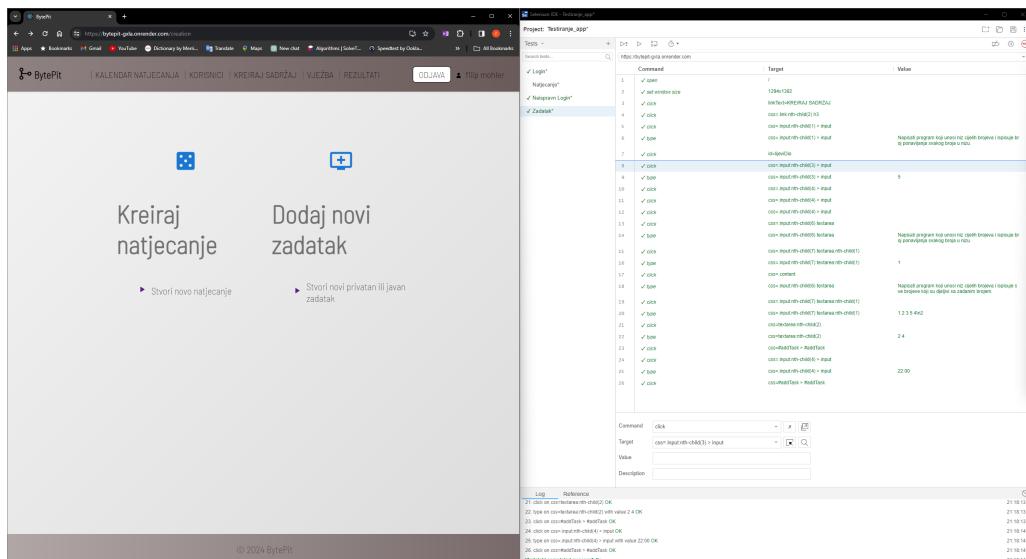
Log - Rekord: 1  
1 click on css=AutocompleteInput OK  
2 type on css=AutocompleteInput + input OK  
3 type on css=AutocompleteInput + input with value fm OK  
4 click on css=AutocompleteInput + input OK  
5 type on css=AutocompleteInput + input OK  
6 click on css=AutocompleteInput + input OK  
7 type on css=AutocompleteInput + input with value 1234 OK  
8 click on css=submitForm OK  
\*Logout Log completed successfully

Slika 5.4: Ispravan Login

## Test 3: Stvaranje zadatka za vježbu

**Ulaz:** "Zadatak 1" 5 "30:00 PM" "Napisati program koji unosi niz cijelih brojeva i ispisuje sve brojeve koji su djeljivi sa zadanim brojem." "1 2 3 4 5 2" "2 4"

**Očekivani izlaz:** Preusmjeravanje stranicu za vježbu.



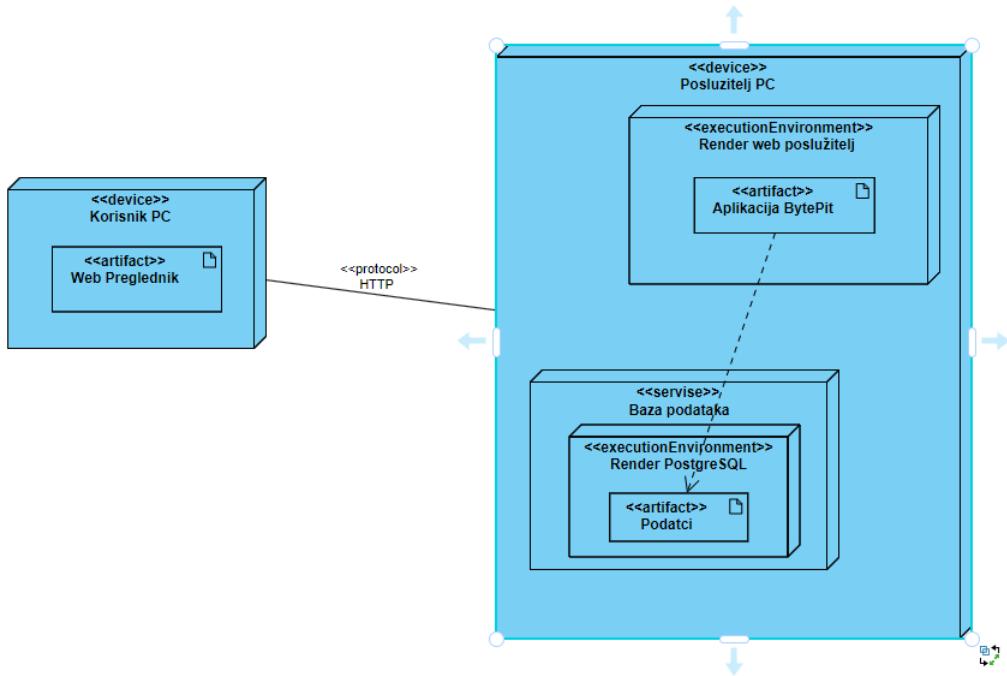
Slika 5.5: Kreiranje zadatka za vježbu

## 5.3 Dijagram razmještaja

UML-dijagrami razmještaja su vrsta strukturnih UML-dijagrama koji prikazuju fizičku arhitekturu i konfiguraciju razmještaja programskog sustava.

U ovoj aplikaciji korisnikovo računalo preko web preglednika pristupa poslužiteljovom računalu pomoću HTTP protokola. Na poslužiteljskom računalu se nalazi naša aplikacija BytePit i baza podataka s kojom aplikacija komunicira.

Sustav je baziran na arhitekturi "klijent – poslužitelj"



Slika 5.6: Dijagram razmještaja

## 5.4 Upute za puštanje u pogon

Postavljanje aplikacije u pogon putem usluge Render zahtijeva pažljivo provedene korake uključujući kreiranje baze podataka, kreiranje backend-a te kreiranje frontenda, čime se osigurava siguran i učinkovit deploy.

Prije samog puštanja aplikacije u pogon potrebno je provesti tri ključna koraka:

### 1. Konfiguracija baze podataka

- Ovaj proces uključuje definiranje environmental varijabli unutar konfiguracije razvojnog okruženja (IDE). U našem slučaju, konfiguracija se nalazi u datoteci `src/main/resources/application.properties`. U ovoj datoteci specificiramo parametre kao što su korisničko ime, lozinka, URL baze podataka te određujemo port poslužitelja. Primjer konfiguracije:

```
# Port na kojem će se vrtiti API
# Obavezno izloziti, ovu varijablu koristi Render
server.port=${PORT:8080}

# Korijenska putanja ("prefiks") za sve zahtjeve na backend - preporuča se postaviti ovo zbog proxy konfiguracije
# Ako je npr. u controlleru navedena putanja /test, moći će joj se pristupiti pomoću putanje /api/test
server.servlet.context-path=/api

# Koristi se samo kao primjer koristenja environment varijable unutar TestController klase
# SERVER_MESSAGE je sada environment varijabla koja će se mapirati na property "message", ako nije postavljena uzima se default vrijednost "Hello from backend! "
message=${SERVER_MESSAGE:Hello from backend! }

# Lokacija Liquibase master chagelog-a
spring.liquibase.change-log=classpath:/db/changelog/changelog-master.xml

# Konfiguracija baze podataka
# Izlaganje environment varijabli je nuzno da bismo mogli postaviti adresu, korisnicko ime i lozinku baze podataka na produkciji
# Stavljanje credentialsa proizvodnje baze podataka direktno u kod je jako losa praksa!
spring.datasource.password=${DB_PASS:password}
spring.datasource.username=${DB_USERNAME:username}
spring.datasource.url=${DB_URL:jdbc:postgresql://localhost:5432/db}
spring.datasource.driverClassName=${DB_DRIVER:org.postgresql.Driver}
```

Slika 5.7: Environment Varijable u IDE-u

## 2. Priprema backend-a za deploy

- Nakon postavljanja baze podataka, slijedi priprema backend-a za deploy. Ovaj korak uključuje dodavanje Dockerfile skripte za izgradnju i pokretanje aplikacije. U našem slučaju, koristimo Maven, pa je Dockerfile konfiguriran na sljedeći način:

```
# Container za izgradnju (build) aplikacije
FROM openjdk:17-alpine AS builder

# Kopiranje izvornog koda u container
COPY ../../mvnw .mvnw
COPY ../../mvnw .
COPY ../../pom.xml .
COPY ../../src src
RUN chmod +x mvnw

# Pokretanje builda
RUN ./mvnw clean package

FROM openjdk:17-alpine
```

```
COPY --from=builder target/*.jar /app.jar
```

```
EXPOSE 8080
```

```
ENTRYPOINT ["java", "-jar", "/app.jar"]
```

---

- Ukoliko se mijenja lokacija Dockerfilea paziti na putanje unutar COPY naredbi u Dockerfile skripti. U nasem slucaju Docker file se nalazi u root direktoriju Docker/Maven/Dockerfile te su po toj putanji pisane naredbe COPY.

### 3. Priprema frontenda za deploy

- Potrebno je dodati potrebne dependencije u package.json datoteku. U terminalu je potrebno izvršiti sljedeće naredbe:

```
npm install http-proxy-middleware  
npm install dotenv  
npm install express
```

---

- Potrebno je kreirati Proxy.js, u src direktorij, koji služi kao proxy server za lokalni development (preusmjeruje api pozive na localhost:8080). Ovo je primjer koda:

```
const { createProxyMiddleware } =  
  require("http-proxy-middleware");  
  
module.exports = function (app) {  
  app.use(  
    "/api",  
    createProxyMiddleware({  
      target: "http://localhost:8080/",  
      changeOrigin: true,  
    })  
  );  
};
```

---

- Stvoriti app.js, u root direktorij, koja sadrži Express server za produkcijski proxy i posluživanje frontenda. Ovo je primjer koda:

```
const express = require("express");
const { createProxyMiddleware } =
    require("http-proxy-middleware");
require("dotenv").config();
const path = require("path");

const app = express();

// Configuration
const { PORT, HOST, API_BASE_URL } = process.env;

// Proxy
app.use(
  "/api",
  createProxyMiddleware({
    target: API_BASE_URL,
    changeOrigin: true,
  })
);

app.use(express.static(path.join(__dirname,
  'build')));

app.listen(PORT, HOST, () => {
  console.log(`Starting Proxy at
${HOST}:${PORT}`);
});

app.get("*", async (req, res) => {
  res.sendFile(path.join(__dirname, 'build',
    'index.html'))
});


---


```

- Izmjeniti package.json skripte i dodati specifične konfiguracije:

```
"build": "yarn install && react-scripts build",
"start-prod": "node app.js",
"engines": {
```

```
"node": ">=18.18.0 <19.0.0"
}
```

Sada, nakon završene pripreme, možemo započeti deploy.

## 1. Kreiranje baze podataka U rander dashboardu:

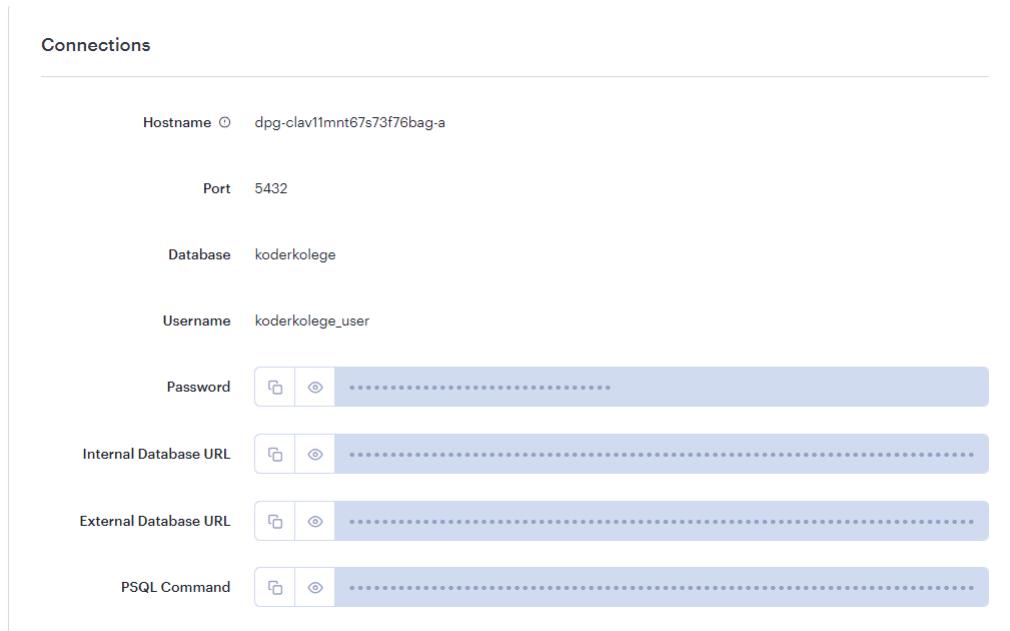
- New →PostgreSQL
- Postaviti ime baze i opcionalno username za korisnika baze (password je automatski generiran)
- Region Frankfurt
- Create Database

## 2. Kreiranje backend-a U Render dashboardu:

- New →Web Service
- Povezati GitHub račun, nakon čega su za odabir dostupni svi projekti na koje imate prava pristupa
- Stisnuti connect pored odgovarajućeg projekta
- Postaviti ime za servis (postat će dio web adrese)
- Postaviti root directory (u nasem slučaju: dev)
- Environment Docker
- Region Frankfurt
- Na dnu proširiti *advanced*
- Dodati potrebne environment varijable (vidi sliku 5.7), kopirati vrijednosti iz postavki baze na renderu (vidi sliku 5.9). Također dodati kopirane environment varijable u IDE okruzenje (vidi sliku 5.7)



Slika 5.8: Environment Varijable za backend



Slika 5.9: Postavke baze podataka na renderu

- Postaviti putanju za Dockerfile ovisno koji se package manager koristi (u ovom slučaju ./docker/maven/Dockerfile, vidi sliku 5.10)



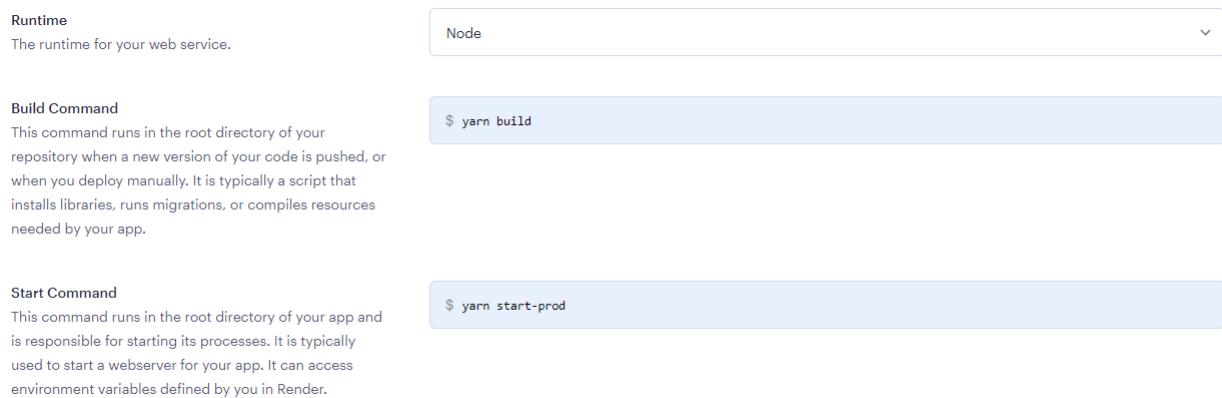
Slika 5.10: Putanja za Dockerfile

- Stisnuti Create Web Service

### 3. Kreiranje frontenda

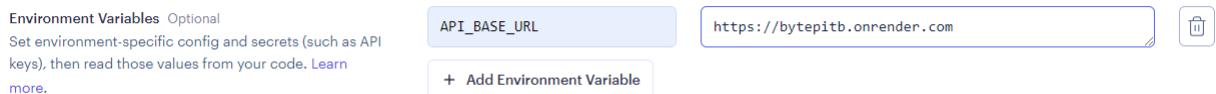
- New → Web Service
- Povezati GitLab racun, nakon cega su za odabir dostupni svi projekti na koje imate prava pristupa
- Stisnuti connect pored odgovarajućeg projekta
- Postaviti ime za servis (postat će dio web adrese)
- Postaviti root directory (u nasem slučaju: dev)
- Environment Node
- Region Frankfurt

- Build Command postaviti na yarn build, a Start Command yarn start-prod (vidi sliku 6.1)



Slika 5.11: Build Command za Node environment

- Na dnu proširiti *advanced*
- Dodati potrebne environment varijable - API BASE URL postaviti na adresu deployanog backenda aplikacije dostupnu na Render dashboardu (vidi sliku 6.1)



Slika 5.12: Environment Varijable za frontend

- Stisnuti Create Web Service

Nakon uspješnog deploja, naša aplikacija sada bezbrižno operira na Render poslužitelju.

## 6. Zaključak i budući rad

Timski rad na projektu Bytepit predstavlja je izazov, ali i priliku za učenje i razvoj. Tijekom vremena izrade projektnog zadatka, suočili smo se s raznim tehničkim izazovima, ali smo istovremeno stekli važna znanja i vještine. U ovom osvrtu, razmatrat ćemo ključne aspekte projekta, uočene izazove, rješenja koja smo primjenili, stečena znanja te preostale mogućnosti za poboljšanja i budući rad.

### Vrijeme izrade projekta

Razdoblje izrade projekta bilo je intenzivno, no istovremeno je pružilo priliku za suradnju i učenje. Rad u timu od sedam članova zahtijevao je učinkovitu komunikaciju i koordinaciju, što smo postigli redovitom komunikacijom preko WhatsApp-a te sastancima putem Google Meet-a.

### Tehnički izazovi

Tijekom razvoja aplikacije, identificirali smo nekoliko tehničkih izazova, uključujući optimalnu integraciju korisničkog sučelja, upravljanje korisničkim podacima i sigurnosne aspekte vezane uz evaluaciju programske zadatke te provedbu natjecanja.

### Stečena znanja

Izrada projekta Bytepit omogućila nam je široko stjecanje znanja iz područja web razvoja, baza podataka, tehnologija ocjenjivanja programskih zadatka i upravljanja korisnicima. Svaki član tima poboljšao je svoje programerske vještine, stekao iskustvo u radu s tehnologijama poput React i Spring Boot te se upoznao s konceptima sigurnog rukovanja podacima korisnika.

### Perspektive za nastavak rada

Daljnji razvoj Bytepita može uključivati proširenje funkcionalnosti, poboljšanje performansi, dodatne sigurnosne aspekte te integracije s drugim platformama recimo Android i IOS.

### Zaključak

Projekt Bytepit predstavlja je izazovno, ali izuzetno poučno iskustvo. Kroz suradnju s timom, rješavanje tehničkih izazova te stjecanje raznolikih vještina, svaki član tima doprinio je uspjehu projekta.

# 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>

# Indeks slike i dijagrama

2.1 Edgar: početna stranica i popis vježbi za ispite . . . . .	7
2.2 Edgar: probni ispit . . . . .	8
2.3 Edgar: stranica sa statistikom . . . . .	8
2.4 Codeforces: rješenja različitih korisnika . . . . .	9
2.5 Codeforces: mogućnost izvršavanja koda . . . . .	10
3.1 Dijagram obrasca uporabe, funkcionalnost reg. i nereg. korisnika . .	21
3.2 Dijagram obrasca uporabe, funkcionalnost voditelja, natjecatelja i administratora . . . . .	22
3.3 Sekvencijski dijagram registracije novog korisnika . . . . .	24
3.4 Sekvencijski dijagram pregleda i uređivanja korisnika . . . . .	24
3.5 Sekvencijski dijagram virtualnog natjecanja . . . . .	25
4.1 Er dijagram baze podataka . . . . .	33
4.2 Dijagram razreda: razredi koji predstavljaju strukturu baze . . . .	34
4.3 Dijagram razreda: razredi Controller . . . . .	35
4.4 Dijagram razreda: razredi Service . . . . .	35
4.5 Dijagram stanja . . . . .	37
4.6 Dijagram aktivnosti . . . . .	38
4.7 Dijagram komponenti . . . . .	39
5.1 Uspješno izvedeni testovi . . . . .	46
5.2 Ne uspješno izvedeni testovi . . . . .	47
5.3 Ispravan Login . . . . .	48
5.4 Ispravan Login . . . . .	48
5.5 Kreiranje zadatka za vježbu . . . . .	49
5.6 Dijagram razmještaja . . . . .	50
5.7 Environment Variable u IDE-u . . . . .	51
5.8 Environment Variable za backend . . . . .	54
5.9 Postavke baze podataka na renderu . . . . .	55
5.10 Putanja za Dockerfile . . . . .	55

5.11 Build Command za Node environment . . . . .	56
5.12 Environment Varijable za frontend . . . . .	56
6.1 Dijagram pregleda promjena . . . . .	65

# Dodatak: Prikaz aktivnosti grupe

## Dnevnik sastajanja

### 1. sastanak

- Datum: 20. listopada 2023.
- Prisustvovali: Petra Buršić, Filip Mohler, Matea Cvetković, Dora Bilić-Pavlinović, Petra Kelković, Mislav Korotaj, Nives Ostojić
- Teme sastanka:
  - Prvotno okupljanje tima
  - Familiarizacija s temom projekta

### 2. sastanak

- Datum: 26. listopada 2023.
- Prisustvovali: Petra Buršić, Filip Mohler, Matea Cvetković, Dora Bilić-Pavlinović, Petra Kelković, Mislav Korotaj, Nives Ostojić
- Teme sastanka:
  - Podjela uloga među članovima tima

### 3. sastanak

- Datum: 14. studenoga 2023.
- Prisustvovali: Petra Buršić, Filip Mohler, Matea Cvetković, Dora Bilić-Pavlinović, Petra Kelković, Mislav Korotaj, Nives Ostojić
- Teme sastanka:
  - Pregled dokumentacije, podjela ostatka posla

### 4. sastanak

- Datum: 16. studenoga 2023.
- Prisustvovali: Petra Buršić, Filip Mohler, Matea Cvetković, Dora Bilić-Pavlinović, Petra Kelković, Mislav Korotaj, Nives Ostojić
- Teme sastanka:
  - Deploy aplikacije

### 5. sastanak

- Datum: 3. siječanj 2024.

- Prisustvovali: Filip Mohler i Petra Kelković
- Teme sastanka:
  - rasprava o strukturi stranice i dijagramima
  - podjela posla

#### 6. sastanak

- Datum: 14. siječanj 2024.
- Prisustvovali: Petra Buršić, Filip Mohler, Matea Cvetković, Petra Kelković, Mislav Korotaj
- Teme sastanka:
  - dogovor oko podjele ostatka posla

#### 7. sastanak

- Datum: 18. siječanj 2024.
- Prisustvovali: Petra Buršić, Filip Mohler, Matea Cvetković, Dora Bilić-Pavlinović, Petra Kelković, Mislav Korotaj, Nives Ostojić
- Teme sastanka:
  - Deploy aplikacije i provjera

**Tablica****aktivnosti***Kontinuirano osvježavanje*

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

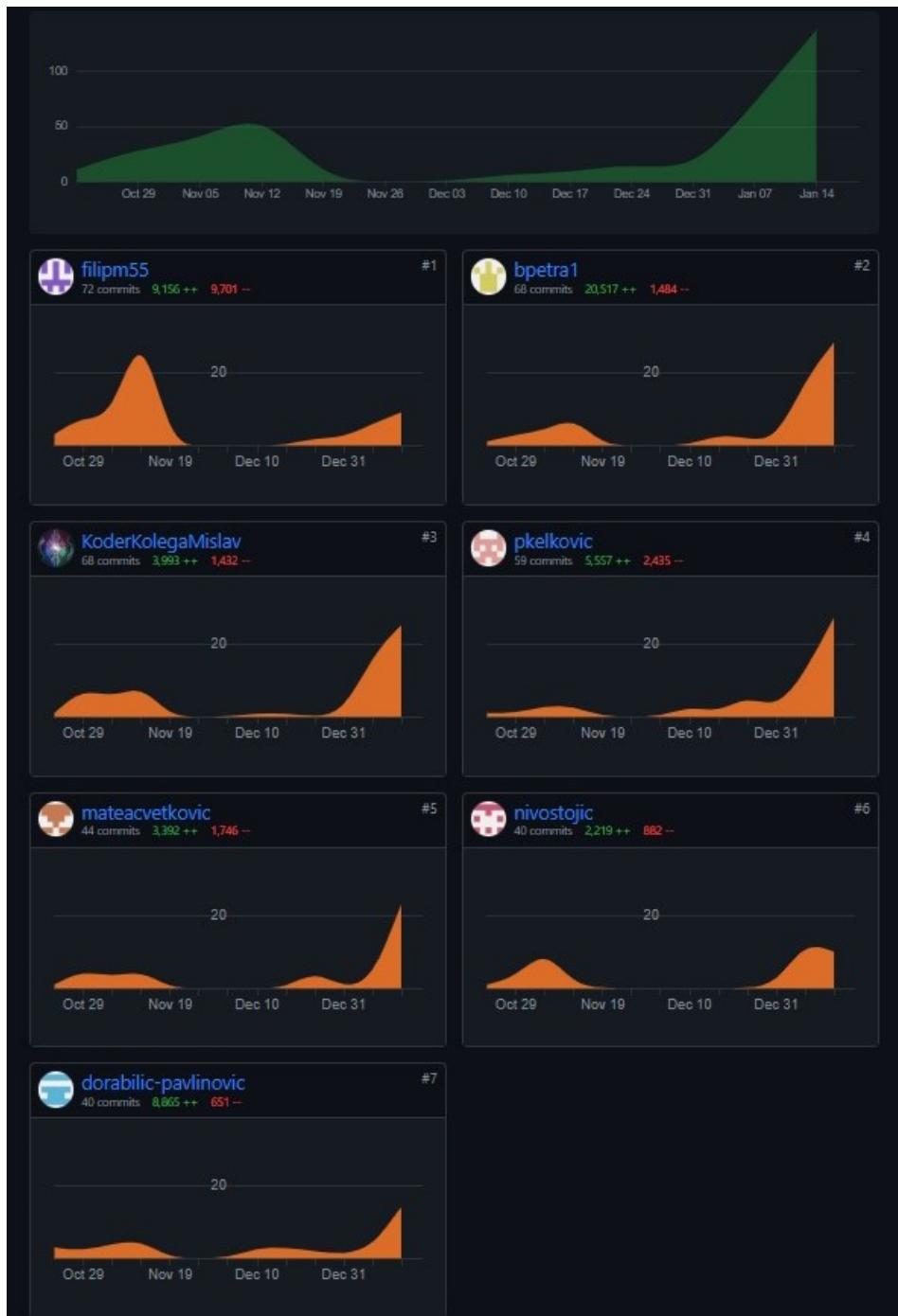
	Petra Kelković	Petra Buršić	Nives Ostojić	Matea Cvetković	Dora Bilić-Pavlinović	Mislav Korotaj	Filip Mohler
Upravljanje projektom	3	3	3	5	6	3	3
Opis projektnog zadatka				5	6		
Funkcionalni zahtjevi	3					3	1
Opis pojedinih obrazaca						3	6
Dijagram obrazaca							2
Sekvencijski dijagrami		6					2
Opis ostalih zahtjeva							2
Arhitektura i dizajn sustava			5		2	2	1
Baza podataka			7			3	1
Dijagram razreda				2			3
Dijagram stanja							
Dijagram aktivnosti							
Dijagram komponenti							
Korištene tehnologije i alati						6	
Ispitivanje programskog rješenja	2		4		5	5	
Dijagram razmještaja							
Upute za puštanje u pogon	10	10	10	10	10	10	10

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

	Petra Kelković	Petra Buršić	Nives Ostojić	Matea Cvetković	Dora Bilić-Pavlincović	Mislav Korotaj	Filip Mohler
Dnevnik sastajanja					1		
Zaključak i budući rad							
Popis literature							
<i>upoznavanje s korištenim tehnologijama</i>	7	6	6	7	6	6	6
<i>izrada osnovnih prikaza</i>	50	45		45			
<i>izrada baze podataka</i>			3		3	5	
<i>spajanje s bazom podataka</i>			13	5	15	10	
<i>spajanje fronta i backenda</i>	25	30	10	30	10	10	
<i>back end</i>		1	50		50	50	

# Dijagrami pregleda promjena



Slika 6.1: Dijagram pregleda promjena