

Home

FSturlic edited this page on Nov 15, 2024 · 4 revisions

[Edit](#) [New page](#)

Programsko inženjerstvo ak. god. 2024./2025.

PlayPadel

Grupa: <G18.2>

Tim: Tim182

Nastavnik: Vlado Sruk

Tim182

Pages 13
<input type="text"/> Find a page...
Home
Programsko inženjerstvo ak. god. 2024./2025.
PlayPadel
Grupa: <G18.2>
Tim: Tim182
Nastavnik: Vlado Sruk
1. Opis projektnog zadatka
2. Analiza zahtjeva
3. Specifikacija programske potpore
4. Arhitektura i dizajn sustava
5. Arhitektura komponenta i razmješ...
6. Ispitivanje programskog rješenja
7. Tehnologije za implementaciju apli...
8. Upute za puštanje u pogon
9. Zaključak i budući rad
A. Popis literature
B. Dnevnik promjena dokumentacije
C. Prikaz aktivnosti grupe

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/franogrinsak/Tim182>



© 2025 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact](#) [Manage cookies](#) [Do not share my personal information](#)

1. Opis projektnog zadatka

Filip Šturić edited this page 3 minutes ago [12 revisions](#)

Cilj ovog projekta je razviti aplikaciju za rezervaciju i promociju padela koja će pojednostaviti pristup informacijama i rezervaciju termina za igru, te olakšati organizaciju i sudjelovanje na turnirima. Aplikacija će omogućiti igračima brži i lakši način pronaleta slobodnih termina, a vlasnicima terena centralizirano mjesto za upravljanje terminima i turnirima. Uz to, aplikacija će omogućiti promociju padela kao sporta kroz sustav obavijesti o turnirima i praćenje rezultata. Tržišta za rezervaciju i organizaciju sportskih događanja trenutačno su fragmentirana, s brojnim platformama koje pružaju ograničene mogućnosti i nisu međusobno povezane. Većina aplikacija nudi osnovne opcije, kao što su pregled slobodnih termina ili jednostavna rezervacija, no izostavljaju sveobuhvatne mogućnosti za organizaciju turnira, što predstavlja izazov za komunikaciju i koordinaciju među igračima i organizatorima. Korisnici često moraju koristiti različite aplikacije kako bi pronašli sve potrebne informacije o sportskim događanjima, što zahtijeva dodatno vrijeme i može smanjiti njihovu motivaciju za prijavu i sudjelovanje. Nedostatak povezanih podataka o korisnicima i njihovim aktivnostima dodatno otežava praćenje sudjelovanja i postignuća na događajima. Integrirana platforma koja je prilagođena padelu rješila bi ove nedostatke te olakšala povezivanje i komunikaciju unutar padel zajednice.

Potencijalni korisnici i njihovi problemi

1. Igrači padela:

Problem: Trenutačno igrači nemaju jednostavan način pronaleta slobodnog terena, što otežava planiranje i organizaciju igara. Također, informacija o turnirima nije uvek lako dostupna, a prijava na iste može biti administrativno zahtjevna.
Potrebe: Korisnici traže platformu gdje mogu lako pregledavati i rezervirati slobodne termine, prijaviti se na turnire te pratiti rezultate završenih događaja.

2. Vlasnici terena:

Problem: Vlasnici terena često moraju osobno komunicirati s igračima radi rezervacija, a organizacija turnira zahtjeva koordinaciju i administrativne napore.
Potrebe: Vlasnici trebaju centralizirani sustav koji im omogućava upravljanje dostupnošću terena, organizaciju turnira i jednostavan način pregleda rezervacija i prijava.

Aplikacija omogućava korisnicima niz pogodnosti

• Igračima:

Jednostavno pretraživanje slobodnih termina po dostupnosti i vrsti terena
Brza rezervacija i različite opcije plaćanja, uključujući PayPal i kreditne kartice
Pregled i prijava na turnire s filtriranjem prema kategorijama, poput kotizacije ili razine igrača
Primanje obavijesti o novim turnirima, te mogućnost praćenja rezultata i komentiranja

• Vlasnicima terena:

Središnji profil s prikazom dostupnih terena i osnovnim informacijama
Postavljanje i uređivanje termina terena i turnira, uključujući cijene i dostupnost
Lakša koordinacija prijava na turnire i ažuriranje rezultata te slike događanja

Potencijalna korist projekta

Razvoj ove aplikacije donosi brojne koristi za sve korisnike i za padel zajednicu u cjelini:

1. Povećanje dostupnosti sporta: Digitalizacijom procesa rezervacije terena i organizacije turnira, aplikacija će pridonijeti širenju padela kao sporta te ga učiniti dostupnijim širem krugu korisnika.
2. Smanjenje administrativnih opterećenja: Automatizacijom rezervacija i prijava na turnire, vlasnici terena će se manje baviti logistikom, a više moći posvetiti samom upravljanju terenima i unapređenju usluga.
3. Poboljšana povezanost zajednice: Kroz obavijesti o novim događajima i mogućnost komentiranja, igrači i vlasnici će moći bolje međusobno komunicirati i graditi povezanost unutar padel zajednice.
4. Unaprjeđena organizacija i promocija sportskih događaja: Jednostavniji proces prijava i upravljanja turnirima može potaknuti organizaciju više lokalnih natjecanja, što će doprinjeti popularizaciji padela.

Uz ovaku aplikaciju, svi korisnici imaju jednostavniji pristup ključnim informacijama, dok sport dobiva važnu digitalnu podršku za daljnji razvoj i popularizaciju.

U ovoj aplikaciji za rezervaciju i organizaciju padel terena i turnira korisnik ima tri moguće uloge:

• Igrač

Pregled i rezervacija terena: Igrači mogu pregledati slobodne termine za određeni teren, kao i napraviti rezervaciju putem aplikacije.
Plaćanje termina: mogu birati način plaćanja (gotovina prilikom dolaska, PayPal, ili kreditna kartica).
Prijava na turnire: Igrači imaju mogućnost pregledavanja dostupnih turnira prema kategorijama (razina igre, cijena, kotizacije, nagrada) i prijave na odabrane. Nakon prijave, čeka se odobrenje vlasnika terena.
Pregled rezultata, fotografija i komentara: Mogu pregledavati rezultate završenih turnira i objavljene fotografije turnira.
Komentiranje i slike: Igrači mogu komentirati odigrane mečeve i postavljati slike istih.
Preplata na obavijesti: Kako bi bili u toku s novim turnirima, igrači se mogu preplatiti na obavijesti o nadolazećim turnirima.

• Vlasnik

Uređivanje profila vlasnika: Mogu uređivati osnovne podatke na svom profilu (naziv, adresa, kontakt, popis svih terena i turnira).
Postavljanje informacija o terenu: Vlasnici mogu uređivati podatke o terenima na svom profilu (lokacija, slika i tip terena (unutarnji ili vanjski)).
Definiranje slobodnih termina: Mogu kreirati termine za rezervaciju (datum, vrijeme i cijena) i upravljati njima.
Organizacija turnira: Vlasnici mogu organizirati turnire, dodavati podatke o svakom turniru (naziv, datum, lokacija, cijena, kotizacije, nagrada i opis), te objavljivati rezultate i fotografije završetka turnira.
Upravljanje prijavama: Svaku prijavu igrača na turnir vlasnik mora odobriti ili odbiti.

Pages (13)
<input type="text"/> Find a page...
» Home
» 1. Opis projektnog zadatka
Potencijalni korisnici i njihovi problemi
Aplikacija omogućava korisnicima niz pogodnosti
Potencijalna korist projekta
Primjeri sličnih rješenja
» 2. Analiza zahtjeva
» 3. Specifikacija programske potpore
» 4. Arhitektura i dizajn sustava
» 5. Arhitektura komponenata i razmješ...
» 6. Ispitivanje programskog rješenja
» 7. Tehnologije za implementaciju apli...
» 8. Upute za puštanje u pogon
» 9. Zaključak i budući rad
» A. Popis literature
» B. Dnevnik promjena dokumentacije
» C. Prikaz aktivnosti grupe

+ Add a custom sidebar

Clone this wiki locally

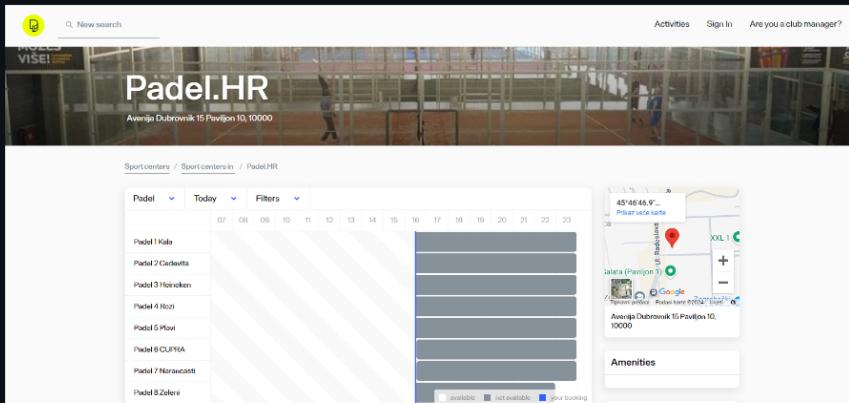
<https://github.com/franogrinsak/Tim1>

- **Administrator sustava**
Upravljanje korisnicima: Administratori mogu upravljati svim korisnicima aplikacije, uključujući i igrače i vlasnike terena, te imaju mogućnost brisanja računa ako je potrebno.
Postavljanje cijena članstva: Administratori definiraju iznos godišnje članarine za vlasnike terena.
Nadzor nad platformom: Administratori imaju uvid u sve podatke unutar sustava i mogu nadgledati aktivnosti, ali se ne bave organizacijom terena i turnira.

Ove uloge omogućavaju organiziran pristup platformi i olakšavaju kontrolu nad podacima i funkcijama prema specifičnim potrebama svakog korisnika, stvarajući sveobuhvatno korisničko iskustvo. Projekt će biti prilagođen za web aplikaciju, optimiziranu za rad na različitim uredajima, što će omogućiti korisnicima jednostavan pristup bez obzira na platformu. S obzirom na postavljene rokove projekta, koji uključuju dva termina predaje, nužno je zadržati fokus na ključnim funkcionalnostima. Zbog ograničenog vremena, napredne opcije poput analitike korisničkog ponašanja, prilagođenih notifikacija na temelju aktivnosti korisnika i automatiziranih marketinških kampanja bit će izostavljene u ovoj fazi razvoja. Projekt će također uključivati fazu testiranja kako bi se osigurala visoka kvaliteta softvera. U toj fazi provest će se osnovna testiranja korisničkog sučelja, funkcionalnosti i sigurnosnih aspekata aplikacije. Izraditi ćemo i upute za puštanje aplikacije u pogon, kao i tehničku dokumentaciju koja će biti osnova za daljnji razvoj i održavanje aplikacije.

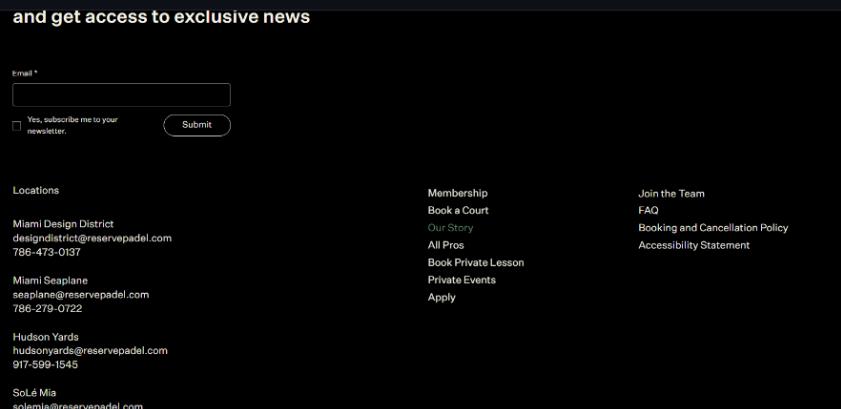
Primjeri sličnih rješenja

- [Playtomic](#) Ime mogućnost prijave na turnire na kojima je navedeno koji spolovi natjecatelja su dopušteni za određeni turnir, teren na kojem se igra, koliko traje i za koji sport (po tim kategorijama je moguće filtrirati turnire). Na prvoj stranici ima navedeno radno vrijeme po danima u tjednu, vidljiva je integracija API karata za lakši prikaz lokacije te integracija kalendara, vidimo opis kluba te opis kluba uz slike terena. Postoji verzija za web i mobilna aplikacija, te je zasebno aplikacija za igrače i za vlasnike. Možemo pročitati politiku privatnosti i kolačića. Dostupni su tereni za više sportova, ne samo za padel. Također možemo pregledati statistiku o popularnosti termina i trendovima korištenja.



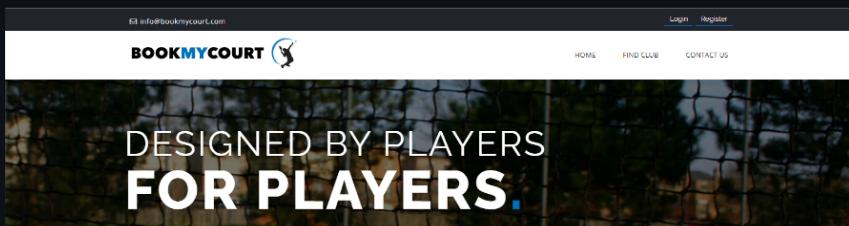
Slika 1.1: Aplikacija Playtomic.io

- [reservepadel](#) Na početnoj stranici imamo mogućnost prijave na newsletter, pregled svih lokacija i kratki opis aplikacije. Postoji stranica za najčešća pitanja, za politiku i pravila rezerviranja i otkazivanja rezervacije. Imamo pregled svih profesionalaca koji su dostupni na treniranje, mogućnost privatnih treninga, rezervacija terena i privatnih dogadaja. Postoje linkovi za profile na društvenim mrežama te link za prijavu za rad. Omogućena je brza rezervacija za povremen korisnike bez registracije.



Slika 1.2: Aplikacija reservepadel

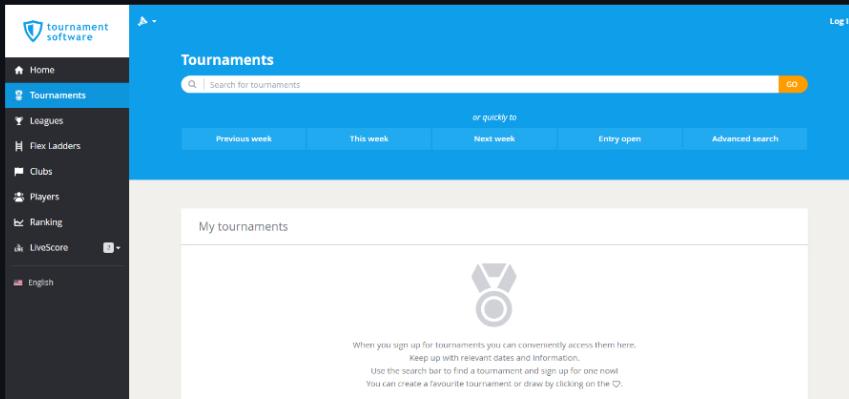
- [BookMyCourt](#) Na početnoj stranici imamo linkove za kontakt, prijavu i registraciju. Možemo pročitati kratak opis aplikacije, . Za razliku od našeg projekta, ovdje igrači sami unose rezultate mečeva, nakon čega se bodovi izračunaju automatski uz pomoć ugrađenih pravila. Nakon toga se ažuriraju pozicije na listi igrača. Svaki igrač ima mogućnost pregleda prošlih mečeva protiv odabranog protivnika, mogućnost prilagodavanja obavijesti kako bi lakše pronašli protivnike sličnih sposobnosti. Za vlasnike je dostupan izvještaj o korištenju aplikacije. Dostupna je za pet sportova, moguće ju je besplatno isprobati 60 dana i registrirati se za pristup.





Slika 1.3: Aplikacija BookMyCourt

- **Tournament Software** Na početnoj stranici možemo odabratи jedan od nekoliko sportova, pregledati tablicu, trenutačne lige i dostupne turnire, sve registrirane igračе i njihov poredak te je dostupan pregled rezultata uživo. Još je dostupna i promjena jezika.



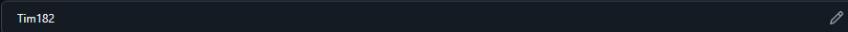
Slika 1.4: Aplikacija Tournament Software

Na temelju ovih aplikacija, dodatne funkcije koje bi korisnici mogli imati su:

- Igračи
Opcija formiranja timova: Igračи se mogu povezati i formirati ekipe za grupne prijave na turnire
Povijest rezultata i statistika: Prikaz statistika poput pobjeda i poraza, broj osvojenih setova i slično
Personalizirane notifikacije: Obavijesti za omiljene terene i slobodne termine
- Vlasnici terena
Automatizirani popusti: Mogućnost postavljanja popusta na termine tijekom manje zauzetih perioda
Izvještavanje o trendovima korištenja: Analize i izvještaji o popularnosti termina i vremena najveće posjećenosti
Prikaz povijesti turnira: Detaljan prikaz prijašnjih turnira i rangiranja na profilima terena
- Administratori
Analitika korisnika: Praćenje aktivnosti korisnika, pregled aktivnih i neaktivnih korisnika, najpopularnije funkcionalnosti
Dodatne postavke članstva: Prilagodba modela članstva s različitim pogodnostima ili planovima, uključujući probne verzije
Praćenje trendova i popularnosti turnira: Izvještaji i statistike o broju prijavljenih igračа i sudjelovanju na turnirima

Ove dodatne funkcionalnosti pružaju bogatije mogućnosti za korisnike, podržavaju personaliziranja iskustva, automatizirano obavještavanje i dublje analize, što može dodatno unaprijediti korisničko iskustvo i pojednostaviti organizaciju turnira i upravljanje rezervacijama.

Također možemo dodati napredno pretraživanje, automatsko generiranje izvještaja ili mogućnost dodatnih opcija za analizu podataka. Moguće je dodati i opciju za više jezika.



NF-1.1	Autentifikacija i autorizacija: Korisnici moraju biti autenticirani sigurnim metodama; lozinke su šifrirane, a autorizacija je potrebna za specifične funkcionalnosti (Google Auth).	Visok
NF-1.2	Sigurnost transakcija: Plaćanja putem PayPala ili kreditnih kartica moraju biti zaštićena kroz sigurne vanjske servise (Stripe).	Visok
NF-3.1	Zaštita podataka: Sve osjetljive informacije moraju biti pohranjene u skladu s GDPR zahtjevima i pravilima privatnosti.	Visok
NF-3.2	Komunikacijski protokoli: Komunikacija između klijentskog sučelja i poslužitelja mora koristiti HTTPS protokol za sigurnost.	Visok
NF-6.1	Održavanje: Sustav treba biti dizajniran na način koji omogućuje lako održavanje i ažuriranje koda te baze podataka (MVC arhitektura).	Visok
NF-6.2	Dokumentacija: Sustav mora biti popraćen tehničkom dokumentacijom koja uključuje opise sustava i vodiće za implementaciju i održavanje (Github Wiki).	Visok
NF-6.3	Testiranje: Sustav treba biti razvijen u skladu s metodama automatiziranog testiranja, uključujući funkcionalna, sigurnosna i integracijska testiranja (Selenium1).	Visok
NF-1.3	Integracija s vanjskim sustavima: Sustav mora podržavati besprekorno integraciju s vanjskim kalendarima (npr. Google Calendar, Stripe) bez smanjenja performansi.	Srednji
NF-4.1	Skalabilnost: Sustav treba biti dizajniran modularno kako bi omogućio jednostavno proširenje funkcionalnosti bez većih promjena postojeće infrastrukture.	Srednji
NF-4.2	Dostupnost: Sustav treba imati najmanje 99.5% dostupnost na godišnjoj razini, uključujući planirane prekide.	Srednji

Dionici

1. Razvojni tim
2. Administrator
3. Naručitelj
4. Korisnici

Aktori i njihovi funkcionalni zahtjevi:

1. Anonimni korisnik (inicijator) može:

prijaviti se u sustav (F-1)
izvršiti registraciju (F-2)

2. Igrač (inicijator) može:

pregledavati i rezervirati termine (F-5)
|| plaćati termin (F-11)
otkazivati rezervacije (F-6)
pregledati rezultate prošlih turnira (F-9)
prijaviti se na turnire (F-14)
preplatići se na sadržaj o turnirima (F-15)
postavljati slike i komentare odigranih mečeva za odigrane turnire (F-18)

3. Vlasnik (inicijator) može:

pregledavati i uredavati svoj korisnički profil (F-10)
dodati nove terene i uredavati podatke o terenu (F-3)
definirati terminе za terene (F-4)
dodavati turnire (F-7)
unositi podatke o otvorenom turniru (F-8)
|| odrediti završetak turnira i rezultate
kupiti i obnavljati članstvo (F-13)
odobriti prijave na turnir (F-11)

4. Administrator (inicijator) može:

postaviti cijenu članstva (F-16)
upravljati korisnicima
|| brisanje, uredivanje i dodavanje (F-17)

5. Baza podataka (sudionik) može:

pohranjivati podatke o registriranim korisnicima
pohranjivati podatke o terenima, terminima i turnirima
pohranjivati podatke potrebe za funkcionalnost usluga



franogrinsak / Tim182

Type to search

Code Issues Pull requests Actions Projects Wiki Security Insights

3. Specifikacija programske potpore

Fran Ogrinšak edited this page 6 minutes ago · 9 revisions

Obrasci uporabe

Dijagrami obrazaca uporabe

Napomena: Aktori vlasnik, igrač i administrator nasljeđuju registriranog korisnika.

1. Visokorazinski dijagram obrazaca uporabe cijelog sustava

uc Visokorazinski

Igrač → UC5 - Rezerviranje termina
Igrač → UC10 - Prijava na turnir
Vlasnik → UC3 - Postavljanje terena
Vlasnik → UC9 - Organizacija turnira
Administrator → UC15 - Dodavanje korisnika

Baza podataka je pasivni aktor sa svim obrascima uporabe

2. dijagram obrazaca uporabe za ključne značajke

| detaljnije dijagrame koji se odnose na specifične značajke, poput procesa prijave korisnika, upravljanja podacima, ... obrade transakcija.

uc Ključne značajke

Igrač → UC20 - Pregled terena
Igrač → UC5 - Rezerviranje termina
Igrač → UC1 - Prijava
Igrač → UC7 - Plaćanje rezervacije
Vlasnik → UC20 - Pregled terena
Vlasnik → UC3 - Postavljanje terena
Vlasnik → UC18 - Ažuriranje godišnje članarine
Administrator → UC18 - Ažuriranje godišnje članarine

Baza podataka je pasivni aktor sa svim obrascima uporabe

PlayPadel

UC20 - Pregled terena, UC5 - Rezerviranje termina, UC1 - Prijava, UC7 - Plaćanje rezervacije, UC3 - Postavljanje terena, UC18 - Ažuriranje godišnje članarine

Stripe, Google

Sekvenčni dijagrami
Provjera uključenosti ključnih funkcionalnosti u obrascima uporabe

Pages 13

Find a page...

Home

1. Opis projektnog zadatka

2. Analiza zahtjeva

3. Specifikacija programske potpore

Obrasci uporabe

Dijagrami obrazaca uporabe

1. Visokorazinski dijagram obrazaca uporabe cijelog sustava
2. dijagram obrazaca uporabe za ključne značajke
3. dijagram obrazaca uporabe za korisničke uloge
4. dijagram obrazaca uporabe za osnovne poslovne procese
5. dijagram obrazaca uporabe za kritične sustave i integracije

Opis obrazaca uporabe

- UC1 - Registracija
- UC2 - Prijava
- UC3 - Postavljanje terena
- UC4 - Postavljanje termina
- UC5 - Rezerviranje termina
- UC6 - Otakzivanje rezervacije
- UC7 - Plaćanje rezervacije
- UC8 - Plaćanje godišnje članarine
- UC9 - Organiziranje turnira
- UC10 - Prijava na turnir
- UC11 - Pregled turnira s pomoću kategorija
- UC12 - Odobravanje prijave na turnir
- UC13 - Preplata na sadržaje o turnirima
- UC14 - Postavljanje slika i komentara za odigrane turnire
- UC15 - Dodavanje korisnika
- UC16 - Uređivanje korisnika
- UC17 - Brisanje korisnika
- UC18 - Ažuriranje godišnje članarine
- UC19 - Uređivanje profila vlasnika
- UC20 - Pregled terena
- UC21 - Pregled svih korisnika

4. Arhitektura i dizajn sustava

5. Arhitektura komponenta i razmješ...

6. Ispitivanje programskog rješenja

7. Tehnologije za implementaciju apli...

8. Upute za puštanje u pogon

9. Zaključak i budući rad

A. Popis literature

B. Dnevnik promjena dokumentacije

C. Prikaz aktivnosti grupe

+ Add a custom sidebar

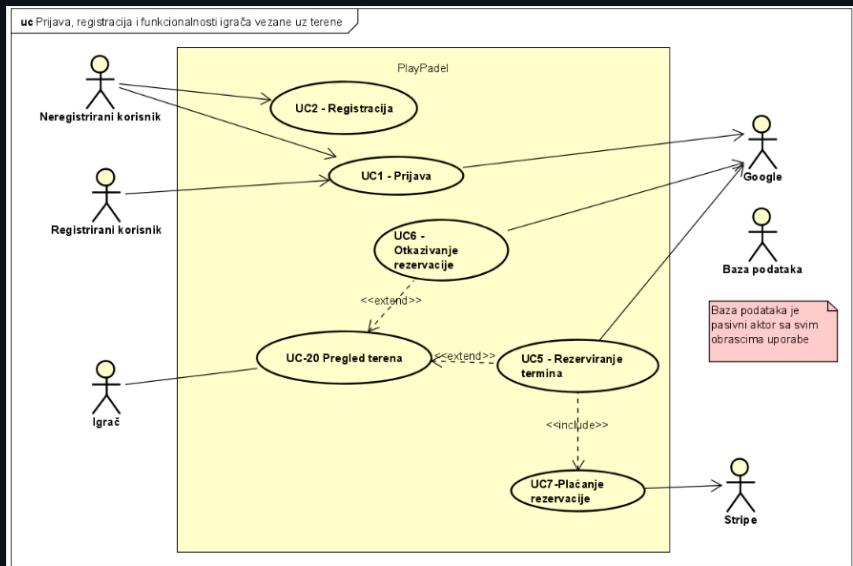
Clone this wiki locally

<https://github.com/franogrinsak/Tim182>

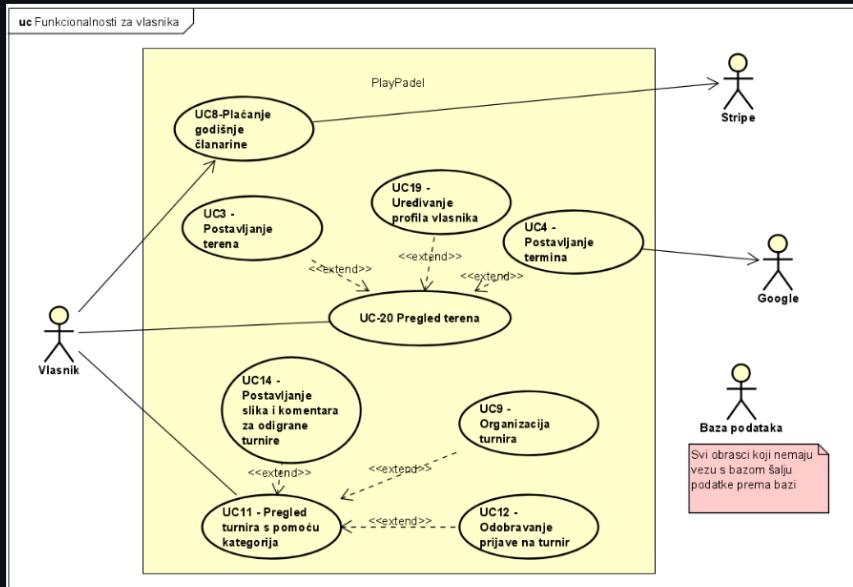
U ranim fazama projekta, važno je jasno definirati koje su osnovne funkcionalnosti sustava, omogućava preciznije planiranje

razvoj i podjele funkcionalnosti na manje zadatke.

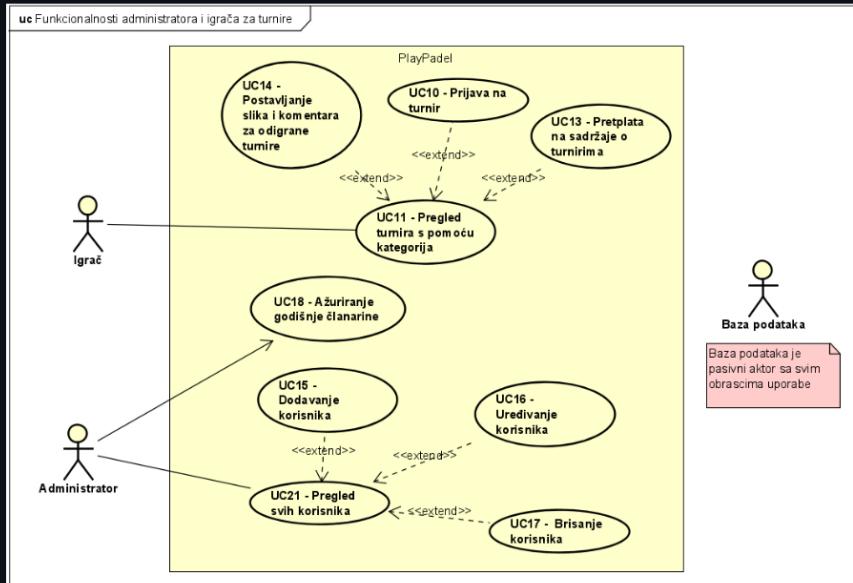
3. dijagram obrazaca uporabe za korisničke uloge



Slika 3.3: Dijagram za prijavu, registraciju i funkcionalnosti igrača vezane uz terene

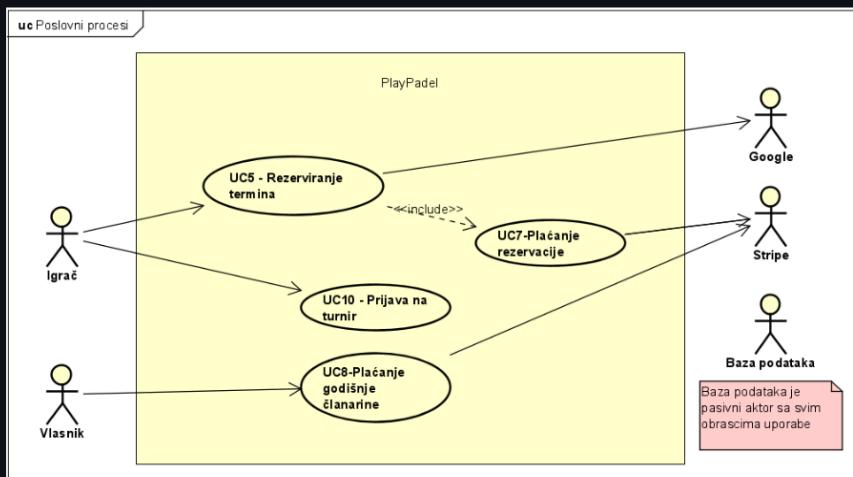


Slika 3.4: Dijagram za funkcionalnosti vezane uz vlasnika terena



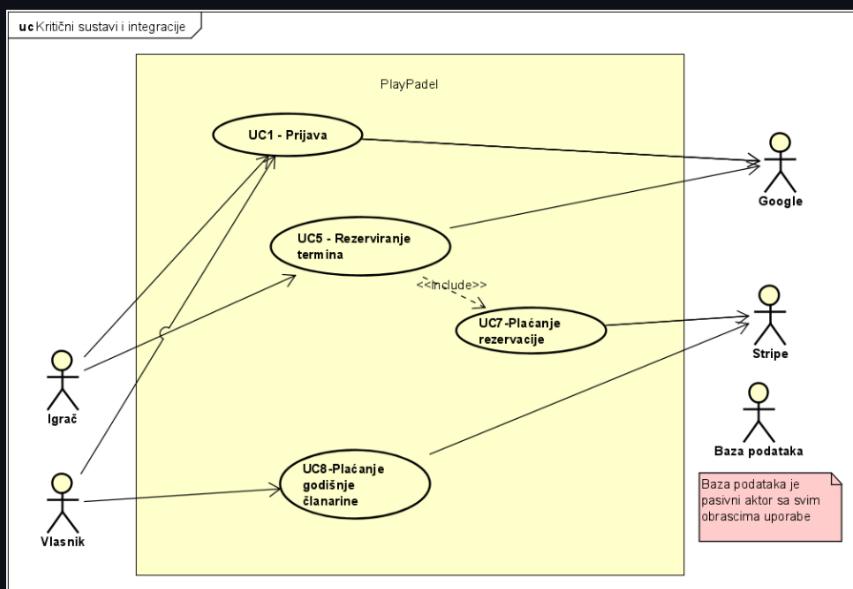
Slika 3.5: Dijagram za funkcionalnosti vezane uz administratora i igrača vezane uz turnire

4. dijagram obrazaca uporabe za osnovne poslovne procese



Slika 3.6: Dijagram za osnovne poslovne procese

5. dijagram obrazaca uporabe za kritične sustave i integracije



Slika 3.7: Dijagram za kritične sustave i funkcionalnosti

Opis obrazaca uporabe

UC1 - Registracija

- Glavni sudionik: Neregistrirani korisnik
- Cilj: Kreacija korisničkog računa
- Sudionici: Baza podataka
- Preduvjet: -
- Opis osnovnog tijeka:

1. Odabir "Register" u sučelju
2. Korisnik unosi potrebne podatke (ime, prezime, optionalno telefonski broj)
3. Sustav unesene podatke validira i sprema u bazu podataka

- Opis mogućih odstupanja:

- 2.a Neispravni podaci
1. Korisnika se obavještava

UC2 - Prijava

- Glavni sudionik: Registrirani korisnik ili neregistrirani korisnik
- Cilj: Prijava u sustav
- Sudionici: Baza podataka, Google
- Preduvjet: Kreirani korisnički račun
- Opis osnovnog tijeka:

1. Odabir "Sign in with Google" u sučelju
2. Korisnik se preusmjeri na vanjski Google servis unosi podatke (email, lozinka)

3. Sustav provjerava unesene podatke u bazi i javlja informacije o uspješnosti

- **Opis mogućih odstupanja:**

2.a Korisnik je unio neispravni e-mail/lozinku

1. Sustav daje upozorenje o neispravnosti unesenih podataka

UC3 - Postavljanje terena

- **Glavni sudionik:** Vlasnik

- **Cilj:** Dodavanje podataka o terenu

- **Sudionici:** Baza podataka

- **Preduvjet:** Registrirani korisnik koji je platio godišnju članarinu, tj vlasnik je

- **Opis osnovnog tijeka:**

1. Vlasnik odabire opciju "Add court"

2. Otvara se stranica s obrascem za unos podataka o terenu (lokacija, slika, tip terena) kojega korisnik ispunjava

3. Sustav validira unesene podatke o terenu i vraća obavijest o uspješnosti kreacije istog

- **Opis mogućih odstupanja:**

2.a Teren već postoji

1. Upozorenje o tome da već postoji teren s istim imenom 3.a Nedostaju neki podaci

2. Nemogućnosti nastavka procesa dodavanja terena

UC4- Postavljanje termina

- **Glavni sudionik:** Vlasnik

- **Cilj:** Postaviti novi teren s detaljima

- **Sudionici:** Baza podataka, Google

- **Preduvjet:** Korisnik je prijavljen u sustav i ima valjanu članarinu

- **Opis osnovnog tijeka:**

1. Vlasnik terena odabire opciju "Add a padel session" na odabranom terenu

2. Unosi podatke o terminu na integrirani Google kalendar (datum, vrijeme, cijena)

3. Sustav validira podatke te ih upisuje u bazu

- **Opis mogućih odstupanja:**

2.a Termin na tom terenu u odabranu vrijeme već postoji

1. Nemogućnost odabira termina 2. Sustav omogućava ponovni odabir

UC5 - Rezerviranje termina

- **Glavni sudionik:** Igrač

- **Cilj:** Rezervirati slobodan termin na terenu za igranje padela

- **Sudionici:** Baza podataka, Google

- **Preduvjet:** Igrač je registrovan i prijavljen u aplikaciji

- **Opis osnovnog tijeka:**

1. Igrač odabire željeni teren u integriranom Google kalendaru i pregledava dostupne termine

2. Igrač odabire željeni termin i sustav ga obavještava o uspješnosti rezervacije

- **Opis mogućih odstupanja:**

2.a Termin je rezerviran

1. Obavijest o rezerviranosti terena u tom trenutku

UC6 - Otkazivanje rezervacije

- **Glavni sudionik:** Igrač

- **Cilj:** Otkazati prethodno rezervirani termin na terenu

- **Sudionici:** Baza podataka, Google

- **Preduvjet:** Igrač ima aktivnu rezervaciju termina

- **Opis osnovnog tijeka:**

1. Igrač otvara popis svojih rezervacija i odabire termin u Google kalendaru koji želi otkazati

2. Sustav traži potvrdu otkazivanja rezervacije

3. Sustav briše rezervaciju

- **Opis mogućih odstupanja:**

2.a Otkazivanje nije moguće jer je manje od 24 sata do termina:

1. Rezervacija se ne može otkazati jer je unutar perioda kada su otkazivanja onemogućena

2. Prikazuje se obavijest da je otkazivanje tog termina onemogućeno 3.a Rezervacija nije pronađena

3. Igračeva rezervacija nije pronadena u bazi podataka (moguće je da je već otkazana ili promijenjena).

4. Prikazuje se obavijest da odabrana rezervacija više nije aktivna.

UC7 - Plaćanje rezervacije

- **Glavni sudionik:** Igrač

- **Cilj:** Uplati rezervaciju termina

- **Sudionici:** Baza podataka, Stripe

- **Preduvjet:** Igrač je prijavljen te je odabrao dostupan termin

- **Opis osnovnog tijeka:**

- 1. Administratoru se prikazuje obrazac s odabirom vrste plaćanja
- 2. Pokreće se tijek plaćanja s vanjskom uslugom Stripe.
- 3. Nakon uspješnog plaćanja u bazi podatka se evidentira podatak o članstvu vlasnika

- **Opis mogućih odstupanja:**

2.a Plaćanje nije uspјelo:

- 1. Obavještava se vlasnika i vraća ga se na opciju iz koraka 1

UC8 - Plaćanje godišnje članarine

- Glavni sudionik: Vlasnik
- Cilj: Uplatiti godišnju članarinu
- Sudionici: Baza podataka, Stripe
- Preduvjet: Vlasnik je prijavljen te je članarina istekla ili se prvi put plaća
- Opis osnovnog tijeka:

- 1. Vlasniku se prikazuje obrazac s odabirom vrste plaćanja
- 2. Pokreće se tijek plaćanja s vanjskom uslugom Stripe.
- 3. Nakon uspješnog plaćanja u bazi podatka se evidentira podatak o članstvu vlasnika

- **Opis mogućih odstupanja:**

2.a Plaćanje nije uspјelo:

- 1. Obavještava se vlasnika i vraća ga se na opciju iz koraka 1.

UC9 - Organiziranje turnira

- Glavni sudionik: Vlasnik
- Sudionici: Baza podataka
- Preduvjet: Vlasnik je prijavljen
- Opis osnovnog tijeka:

- 1. Vlasnik popunjava različite kategorije za svoj turnir te tako stvara pojedini turnir
- 2. U bazu podataka se spremaju promjene

- **Opis mogućih odstupanja:**

2.a Neispravni ili nepotpuni podaci

- 1. Vlasniku se šalje obavijest o ponovnom pokušaju unosa podataka

UC10 - Prijava na turnir

- Glavni sudionik: Igrač
- Sudionici: Baza podataka
- Preduvjet: Igrač je prijavljen
- Opis osnovnog tijeka:

- 1. Igrač bira neki od otvorenih turnira te odabire opciju prijave na turnir
- 2. Baza podataka spremi podatke o prijavi

- **Opis moguća odstupanja:**

2.a Turnir više nije dostupan za prijavu:

- 1. Turnir je zatvoren ili je dosegnut maksimalan broj prijava.

- 2. Igraču se prikazuje obavijest da prijava više nije moguća za taj turnir.

UC11 - Pregled turnira s pomoću kategorija

- Glavni sudionik: Igrač ili Vlasnik
- Cilj: Pregledati turnire s pomoću različitih kategorija
- Sudionici: Baza podataka
- Preduvjet: Igrač ili vlasnik je prijavljen
- Opis osnovnog tijeka:

- 1. Igrač ili Vlasnik bira različite kategorije, primjerice razina igrača, cijena kotizacije, iznos nagrade
- 2. Na osnovu odabranih kategorija korisniku se prikazuju adekvatni turniri

- **Opis mogućih odstupanja:**

2.a Nema turnira koji odgovaraju odabranim kategorijama:

- 1. Sustav prikazuje obavijest da nema dostupnih turnira koji odgovaraju odabranim kriterijima.

- 2. Igraču se predlaže da pokuša s drugim kriterijima.

UC12 - Odobravanje prijave na turnir

- Glavni sudionik: Vlasnik
- Sudionici: Baza podataka
- Preduvjet: Vlasnik je prijavljen
- Opis osnovnog tijeka:

- 1. Vlasnik otvara listu prijava

- 2. Vlasnik odobrava prijavu

- 3. U bazu podataka se spremaju promjene

- **Opis mogućih odstupanja:**

- 2.a Prijava nije moguća za odobrenje:
1. Prijava je možda već ranije odobrena ili odbijena, ili turnir više ne prima prijave.
 2. Vlasniku se prikazuje obavijest da je prijava već obradena.

UC13 - Preplata na sadržaje o turnirima

- Glavni sudionik: Igrač
- Sudionici: Baza podataka
- Preduvjet: Igrač je prijavljen
- Opis osnovnog tijeka:
 1. Igrač se pretplaćuje na sadržaj turnira
 2. Navedena promjena se sprema u bazu podataka
- Opis mogućih odstupanja:
 - 1.a Igrač je već preplaćen:
 1. Ako je igrač već preplaćen na sadržaje o turnirima, prikazuje se obavijest da je preplata već aktivna.

UC14 - Postavljanje slika i komentara za odigrane turnire

- Glavni sudionik: Igrač
- Sudionici: Baza podataka
- Preduvjet: Igrač je prijavljen
- Opis osnovnog tijeka:
 1. Igrač izabere opciju prikaza odigranih turnira
 2. Igrač ima mogućnost postaviti sliku ili komentar za odabrani odigrani turnir
- Opis mogućih odstupanja:
 - 2.a Turnir nije pronađen ili je nedostupan za komentiranje/slikanje:
 1. Ako turnir više nije dostupan ili ga igrač nije odigrao, prikazuje se obavijest da nije moguće dodati sadržaj za taj turnir.

UC15 - Dodavanje korisnika

- Glavni sudionik: Administrator
- Cilj: Dodati novog korisnika
- Sudionici: Baza podataka
- Preduvjet: Administrator je prijavljen
- Opis osnovnog tijeka:
 1. Administrator odabire opciju dodaj korisnika
 2. Sustav šalje administratoru obrazac
 3. Administrator popunjava obrazac s potrebnim podatcima o korisniku
 4. Administrator šalje obrazac te se u bazi podataka pojavljuje novi korisnik
 5. Administrator dobiva poruku o uspješnosti dodavanja
- Opis mogućih odstupanja:
 - 3.a Obrazac je nepotpun
 1. Obavještava se administratora 4.a Dodavanje nije uspjelo
 2. Obavještava se administratora

UC16 - Uređivanje korisnika

- Glavni sudionik: Administrator
- Cilj: Ažurirati podatke o postojećim korisnicima
- Sudionici: Baza podataka
- Preduvjet: Administrator je prijavljen
- Opis osnovnog tijeka:
 1. Administrator za odgovarajućeg korisnika odabire opciju ažuriraj podatke
 2. Otvara obrazac s promjenom (UC15)
 3. Administrator ažurira obrazac s novim podatcima o korisniku
 4. Administrator šalje obrazac te se u bazi podataka ažuriraju podaci o korisniku
 5. Administrator dobiva poruku o uspješnosti ažuriranja
- Opis mogućih odstupanja:
 - 3.a Obrazac je nepotpun
 1. Obavještava se administratora 4.a Ažuriranje nije uspjelo
 2. Obavještava se administratora

UC17 - Brisanje korisnika

- Glavni sudionik: Administrator
- Cilj: Obrisati postojećeg korisnika
- Sudionici: Baza podataka
- Preduvjet: Administrator je prijavljen
- Opis osnovnog tijeka:
 1. Administrator za odgovarajućeg korisnika odabire opciju izbriši korisnika

- 2. Sustav javlja administratoru o uspješnosti brisanja

- Opis mogućih odstupanja:

3.2 Brisanje nije uspjelo 1. Obaveštava se administratora

UC18 - Ažuriranje godišnje članarine

- Glavni sudionik: Administrator
- Cilj: Promjeniti iznos godišnje članarine
- Sudionici: Baza podataka
- Preduvjet: Administrator je prijavljen
- Opis osnovnog tijeka:

1. Administrator odabire opciju godišnja članarina
2. Administrator mijenja iznos trenutačne cijene u obrascu.
3. Sustav javlja administratoru o uspješnosti ažuriranja cijene

- Opis mogućih odstupanja:

2.a Administrator nije unio cijenu ili je izvan intervala [0€, 1000€]

1. Prikazuje se poruka o odstupanju uvjeta 3.a Ažuriranje nije uspjelo
2. Obaveštava se administratora

UC19 - Uređivanje profila vlasnika

- Glavni sudionik: Vlasnik
- Cilj: Ažurirati podatke o korisničkom profilu vlasnika
- Sudionici: Baza podataka
- Preduvjet: Vlasnik je prijavljen
- Opis osnovnog tijeka:

1. Vlasnik otvara stranicu profila i odabire opciju "Uredi profil"
2. Sustav prikazuje obrazac s trenutačnim podacima profila (npr. ime, kontakt podaci, slika profila)
3. Vlasnik ažurira željene informacije u obrascu
4. Sustav provjerava validnost novih podataka i sprema ih u bazu podataka

- Opis mogućih odstupanja:

3.a Ažurirani podaci su nepotpuni ili neispravni

1. Sustav prikazuje upozorenje o potrebnim popravcima i vraća vlasnika na obrazac.

UC20 - Pregled terena

- Glavni sudionik: Vlasnik ili Igrač
- Cilj: Prikazati popis objavljenih terena
- Sudionici: Baza podataka
- Preduvjet: Vlasnik ili Igrač je prijavljen
- Opis osnovnog tijeka:

1. Korisnik otvara stranicu s popisom terena
2. Sustav prikazuje korisniku potrebne podatke

- Opis mogućih odstupanja:

2.a Tereni nisu postavljeni

1. Korisniku se prikazuje informacija o nepostojanju terena

UC21 - Pregled svih korisnika

- Glavni sudionik: Administrator
- Cilj: Prikazati popis objavljenih terena
- Sudionici: Baza podataka
- Preduvjet: Administrator je prijavljen
- Opis osnovnog tijeka:

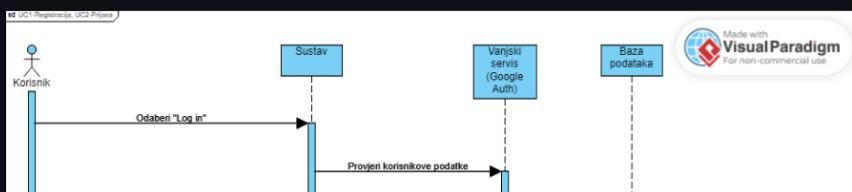
1. Korisnik otvara stranicu s popisom svih korisnika, uključujući trenutačnog

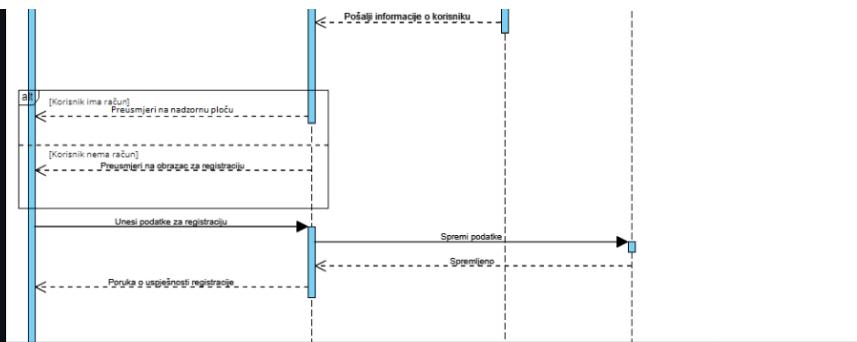
2. Sustav prikazuje tablicu s popisom

Sekvenčijski dijagrami

- Opis sekvenčnog dijagrama za UC1 - Registracija i UC2 - Prijava

Korisnik odabire "Log in". Sustav komunicira sa vanjskim servisom za autentifikaciju koji provjerava korisnikove podatke. Kao odgovor šalje informaciju o tome ima li taj korisnik račun ili je novi. Ako se radi o postojećem korisniku, prijava je završena i korisnika se preusmjerava na nadzornu ploču aplikacije. U slučaju da se radi o novom korisniku, preusmjerava ga se na obrazac za registraciju. On unosi tražene podatke. Ti podaci se spremaju u bazu podataka. Na kraju se šalje obavijest o uspješnosti registracije.

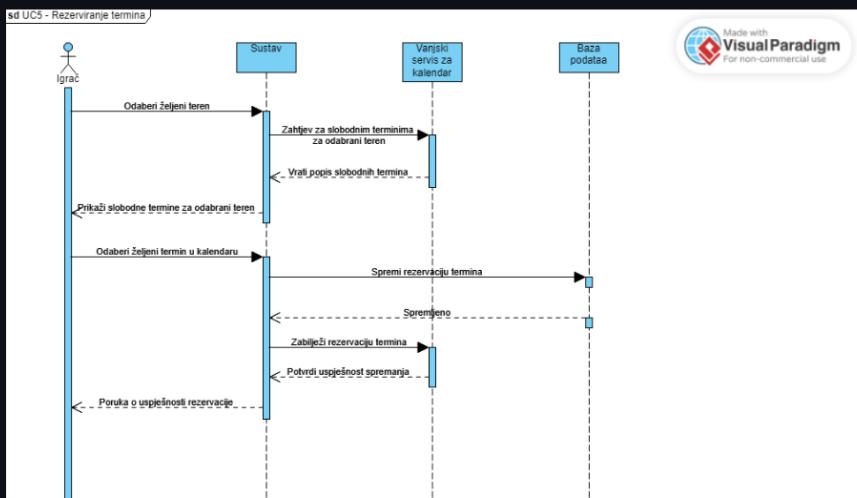




Slika 2.5: Sekvenčni dijagram za UC1 i UC2

- Opis sekvenčnog dijagrama za UC5 - Rezerviranje termina

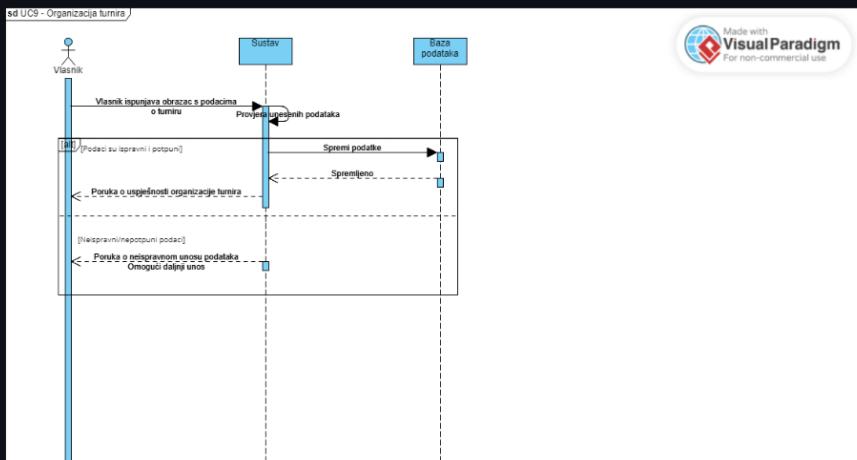
Proces rezervacije termina za padel teren započinje kada igrač odabere željeni teren u aplikaciji. Tada se sustav povezuje s vanjskim servisom za kalendar kako bi prikupio slobodne termine za taj teren i prikazao ih igraču. Popis slobodnih termina se šalje u aplikaciju gdje se oni igraču prikazuju na kalendaru. Nakon što igrač odabere termin, sustav pohranjuje rezervaciju u svoju bazu podataka, čime osigurava da je rezervacija evidentirana u glavnom sustavu. Nakon potvrde uspješnog spremanja u bazi, sustav zatim šalje podatke o rezervaciji i vanjskom servisu radi dodatne sinkronizacije. Kada vanjski servis potvrdi spremanje, sustav obavještava igrača o uspješnoj rezervaciji, čime je postupak završen.



Slika 2.6: Sekvenčni dijagram za UC5

- Opis sekvenčnog dijagrama UC9 - Organizacija turnira

Na početku procesa, vlasnik ispunjava obrazac sa podacima o turniru (naziv, lokacija, datum, cijena kotizacije, nagrada i opis). Nakon što vlasnik završi s unosom podataka, sustav provodi provjeru kako bi osigurao da su svi podaci ispravni i potpuni. Ako sustav potvrdi da su podaci ispravni, pohranjuje ih u svoju bazu podataka. Nakon uspješne pohrane podataka, sustav vraća potvrdu vlasniku da je turnir uspješno kreiran. U slučaju da sustav otkrije pogreške ili nedostatke u podacima, prikazuje vlasniku obavijest u kojoj ga obavještava da mora ispraviti unesene podatke prije nego što može nastaviti s kreiranjem turnira.



Slika 2.6: Sekvenčni dijagram za UC9

Provjera uključenosti ključnih funkcionalnosti u obrasce uporabe

Obrazac	Funkcionalni zahtjev
UC1 - Registracija	F-2 - Registracija kao igrač ili vlasnik

UC2 - Prijava	F-1 - Prijava u sustav kao administrator, igrač ili vlasnik
UC3 - Postavljanje terena	F-3 - Dodavanje novih terena i uređivanje podataka o terenu
UC4 - Postavljanje termina	F-4 - Organizacija termina
UC5 - Rezervacija termina	F-5 - Rezerviranje termina
UC6 - Otkazivanje rezervacije	F-6 - Otkazivanje rezervacija
UC7 - Plaćanje rezervacije	F-12 - Plaćanje termina
UC8 - Plaćanje godišnje članarine	F-13 - Plaćanje članstva
UC9 - Organiziranje turnira	F-7 - Dodavanje turnira
	F-8 - Unos podataka o turniru
UC10 - Prijava na turnir	F-14 - Prijave na turnire
UC11 - Pregled turnira pomoću kategorija	F-9 - Pregled turnira prema kategorijama
UC12 - Odobravanje prijave na turnir	F-11- Odobravanje prijave igrača na turnir
UC13 - Pretplata na obavijesti o turnirima	F-15 - Podsustav obavještavanja
UC14 - Postavljanje slika i komentara za odigrane turnire	F-17 - Postavljanje slika i komentara
UC15 - Dodavanje korisnika	F-10 - Pregled i uređivanje profila vlasnika
	F-17 - Upravljanje korisnicima
UC16 - Uređivanje korisnika	F-10 - Pregled i uređivanje profila vlasnika
	F-17 - Upravljanje korisnicima
UC17 - Brisanje korisnika	F-10 - Pregled i uređivanje profila vlasnika
	F-17 - Upravljanje korisnicima
UC18 - Ažuriranje godišnje članarine	F-16 - Postavljanje članstva
UC19 - Uređivanje profila vlasnika	F-10 - Pregled i uređivanje profila vlasnika
UC20 - Pregled terena	F-3 - Dodavanje novih terena i uređivanje podataka o terenu
UC21 - Pregled svih korisnika	F-17 - Upravljanje korisnicima

Tim182



© 2025 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact](#) [Manage cookies](#) [Do not share my personal information](#)

4. Arhitektura i dizajn sustava

Fran Ogrinšak edited this page 1 hour ago · 21 revisions

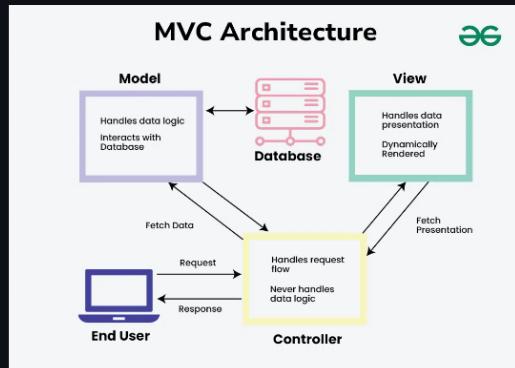
Edit New page

Arhitektura sustava

Opis arhitekture

Stil arhitekture

Odabrali smo MVC arhitekturu za platformu koja omogućava rezervaciju i plaćanje terena za igru padela, kao i praćenje turnira, zbog njene sposobnosti da odvoji poslovnu logiku, prezentaciju i upravljanje korisničkim interakcijama. Ova struktura ne samo da pojednostavljuje razvoj i održavanje sustava, već i poboljšava organizaciju koda, što je od esencijalne važnosti za uspešnu implementaciju i buduće nadogradnje [8].



Slika 3.1: MVC model arhitekture

Objašnjenje komponenata i njihove povezanosti:

- Model**
Opis: Model predstavlja poslovnu logiku aplikacije i upravlja podacima koji dolaze iz baze. Model ne sadrži pogled podataka, već samo manipulira podacima koji će se prikazati.
Povezanost: Prima zahtjeve od nadglednika za ažuriranje podataka, a promjene šalje pogledu (View) koji ih koristi za ažuriranje sučelja.
- View (Pogled)**
Opis: Pogled prikazuje podatke korisnicima putem korisničkog sučelja. Ova komponenta predstavlja vizualni dio aplikacije s kojim korisnici izravno komuniciraju.
Povezanost: Komunicira s nadglednikom radi primanja ažuriranja u vezi s podacima i prikazuje promjene iz modela kada se podaci promijene.
- Controller (Nadglednik)**
Opis: Nadglednik obraduje korisničke zahtjeve i ulazne podatke, šalje upute modelu za ažuriranje podataka i pogledu za prikaz novih informacija.
Povezanost: Nadglednik služi kao posrednik između pogleda (View) i modela, preusmjerava korisničke zahtjeve modelu i ažurira pogled na temelju promjena podataka.

Podsistavi

- Korisnički sučelje (UI):**
Odgovorno za interakciju s korisnicima. Razvijeno s pomoću tehnologije React.
- API Gateway:**
Servis koji upravlja svim API pozivima i usmjerava ih prema odgovarajućim mikroservisima.
- Mikroservisi:**
Različiti servisi za specifične funkcionalnosti (npr. autentifikacija: Google Auth, obrada plaćanja: Stripe, kalendar: Google kalendar).
- Baza podataka:**
Različite baze podataka (relacijske, NoSQL) za pohranu podataka.

Preslikavanje na radnu platformu

- Klijent:** Korisnički uređaji, poput mobilnih telefona, tableta i računala, koriste web preglednik ili aplikaciju za pristup korisničkom sučelju (UI). Sučelje omogućuje unos podataka i prikaz informacija sa servera, čime korisnici ostvaruju interakciju sa sustavom putem mreže.
- Server:** Sadrži poslovnu logiku, upravlja autentifikacijom korisnika i API-jem koji omogućuje integraciju s vanjskim uslugama te bazom podataka. Time server centralizira ključne operacije, čuva podatke, provjerava pristup korisniku i posreduje komunikaciju s klijentima putem API-ja.

Spremišta podataka

Centralizirana baza podataka (SQL) koja pohranjuje:

Pages 13
Find a page...
Home
1. Opis projektnog zadatka
2. Analiza zahtjeva
3. Specifikacija programske potpore
4. Arhitektura i dizajn sustava
Arhitektura sustava
Opis arhitekture
Stil arhitekture
Podsistavi
Preslikavanje na radnu platformu
Spremišta podataka
Mrežni protokoli
Globalni upravljački tok
Sklopovskoprogamski zahtjevi
Razlozi za odabir MVC arhitekture:
Organizacija sustava s najviše razine apstrakcije
Klijent-postužitelj
Baza podataka
Datotečni sustav
Grafičko sučelje
Organizacija aplikacije
Baza podataka
Opis tablica
user_roles
users
owners
players
courts
tournaments
bookings
images
comments
notifications
participateIn
membership_price
Dijagram baze podataka
Dijagram razreda
Dinamičko ponašanje aplikacije
UML dijagrami stanja
UML dijagrami aktivnosti
5. Arhitektura komponenata i razmješ...
6. Ispitivanje programskog rješenja
7. Tehnologije za implementaciju apli...
8. Upute za puštanje u pogon
9. Zaključak i budući rad
A. Popis literature
B. Dnevnik promjena dokumentacije
C. Prikaz aktivnosti grupe

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/franogrinsak/Tim1>

Podatke o terenima za igru padela
Informacije o igračima i vlasnicima dvorana
Detalje o rezervacijama, plaćanjima i turnirima

Mrežni protokoli

API protokoli (REST, GraphQL): Omogućavaju upite i manipulaciju podacima
HTTP/HTTPS: Koriste se za komunikaciju između klijenta i poslužitelja

Globalni upravljački tok

Korisnici šalju zahtjeve API Gateway putem korisničkog sučelja
API Gateway prosledjuje zahtjeve mikroservisima
Mikroservisi obrađuju zahtjeve i komuniciraju s bazama podataka po potrebi
Rezultati se vraćaju korisnicima putem API Gatewaya

Sklopovskoprogramski zahtjevi

- Hardverski zahtjevi:
 - Server s potrebnim resursima (Render: disk 1 GB, RAM 512 MB, CPU 100 millicores)
- Softverski zahtjevi:
 - Podržava sve operacijske sustave koji podržavaju jdk, node i postgresql, Render za upravljanje bazom podataka, frameworkovi i alati za razvoj (npr. Node.js, Spring Boot). Korisnici moraju imati samo moderan browser.
- Sigurnosni zahtjevi:
 - Implementacija autentifikacije i autorizacije (npr. OAuth 2.0)
- Performance:
 - Sustav treba biti sposoban obraditi zahtjev do desetak sekundi

Razlozi za odabir MVC arhitekture:

- Razdvojene odgovornosti: MVC jasno dijeli model (poslovna logika i podaci), pogled (korisničko sučelje) i nadglednik (upravljanje interakcijama), što omogućava timovima da se fokusiraju na specifične aspekte bez ometanja
- Jednostavna prilagodba: Moguće je lako mijenjati ili zamjeniti jedan dio sustava bez utjecaja na ostale, čime se povećava fleksibilnost aplikacije
- Ponovna upotreba koda: Razdvajanjem komponenti omogućuje se ponovna upotreba i testiranje koda, što smanjuje vrijeme razvoja i osigurava dosljednost
- Bolje upravljanje složenosti: MVC organizira složene aplikacije, olakšavajući praćenje logike i interakcija, što je posebno korisno za veće projekte
- Podrška za timski rad: Odvojene komponente omogućavaju članovima tima rad na različitim dijelovima paralelno, povećavajući produktivnost
- Jednostavno testiranje: MVC olakšava testiranje jer se model, pogled i nadglednik mogu testirati neovisno, što pomaže bržem otkrivanju problema

Organizacija sustava s najviše razine apstrakcije

Klijent-poslužitelj

- Klijent
Klijent je dio aplikacije koji koristi korisnik (igrač ili vlasnik terena). Korisnici se prijavljuju u aplikaciju putem web preglednika ili mobilne aplikacije. Korisnici mogu registrirati profil, pretraživati dostupne terene i termine, rezervirati ih uz plaćanje putem integriranih sustava, te pratiti turnire i rezultate uz interakciju s profilima vlasnika dvorana.
- Poslužitelj
Poslužitelj je centralni dio aplikacije koji obrađuje zahtjeve klijenata, upravlja podacima i provodi logiku aplikacije. Poslužitelj obrađuje zahtjeve klijenata, upravlja podacima u bazi, osigurava autentifikaciju i autorizaciju korisnika te implementira poslovnu logiku za rezervacije, plaćanja i upravljanje korisnicima.

Baza podataka

Baza podataka pohranjuje sve relevantne informacije potrebne za funkcioniranje platforme. Ona omogućava pohranu, pretraživanje i upravljanje podacima u strukturalnom formatu

Podaci koje pohranjuje:

- Podaci o korisnicima: Informacije o igračima i vlasnicima dvorana (ime, prezime, kontakt informacije, tip korisnika)
- Podaci o terenima: Detalji o svakom terenu (naziv, lokacija, tip terena, slike, dostupni termini i cijene)
- Podaci o turnirima: Informacije o organiziranim (naziv, lokacija, datum, cijena kotizacije, nagrade, opisi) i završenim turnirima (rezultati i fotografije)
- Podaci o rezervacijama: Informacije o rezerviranim terminima, uključujući korisnika, teren, datum, vrijeme i status rezervacije

Datotečni sustav

Datotečni sustav služi za pohranu podataka koji nisu izravno povezani s bazom podataka, kao što su medijski sadržaji ili datoteke koje se koriste u aplikaciji

- Podaci koje pohranjuje: slike terena i rezultate turnira, log datoteke (zapisnici o aktivnostima korisnika i sustava, koji se koriste za praćenje i dijagnostiku)

Grafičko sučelje

Grafičko sučelje (Graphical User Interface, GUI) predstavlja komponentu aplikacije s kojom korisnici komuniciraju. Klučno je za korisničko iskustvo jer omogućava vizualnu percepцију i olakšava interakciju s aplikacijom. U ovom slučaju se radi o web

aplikaciji. Suceđe povezano s poslužiteljem koji obraduje zahtjeve i komunicira s bazom podataka za pohranu podataka te s datotečnim sustavom za pohranu datoteka.

Organizacija aplikacije

- Frontend i Backend slojevi: Frontend sloj zadužen je za prikaz korisničkog sučelja, prikupljanje podataka od korisnika i proslijedivanje zahtjeva backenu putem API-ja. Backend sloj odgovoran je za obradu tih zahtjeva, upravljanje bazom podataka, provedbu poslovne logike i osiguravanje sigurnosti. Komunikacija između slojeva odvija se putem HTTP zahtjeva, pri čemu frontend šalje podatke backendu, a zauzvrat dobiva obrađene informacije, obično u JSON formatu, koje koristi za ažuriranje sučelja.
- MVC arhitekturu smo implementirali tako da model sadrži ORM za rad s bazom podataka, gdje su definirane klase za korisnike, terene i turnire. Pogled je sloj koji je zadužen za prikaz podataka korisniku. U našem slučaju je to frontend aplikacije koja koristi react.js i vanjski servis Google calendar za prikaz kalendarja termina. S backendom komunicira uz pomoć API-ja. Nadglednik je posrednik između modela i pogleda u kojem kontroleri mapiraju rute na odgovarajuće funkcije. Te funkcije obradjuju ulazne podatke uz pomoć metoda definiranih u modelu te vraćaju odgovore korisniku.

Baza podataka

Pri odabiru baze podataka nismo uočili specifične funkcionalnosti koje nude samo određene tehnologije. Imajući to na umu, odlučili smo se koristiti PostgreSQL kao sustav za upravljanje bazama podataka jer su svi članovi razvojnog tima upoznati s njim. Baze podataka u ovoj tehnologiji temelje se na relacijskoj algebri. Podaci su organizirani u relacije koje sadrže atribute i naziv. Dohvat podataka temelji se na operacijama relacijske algebre. Glavna zadaća baze podataka u našem sustavu jest praćenje stanja. Bitno je da su omogućeni dohvati, pohrana, izmjena i brisanje podataka. U PostgreSQL-u koriste se komponente kao što su tablice(relacije), indeksi, pogledi i pohranjene procedure. Tablice služe za pohranu podataka, indeksi omogućuju brži pristup podacima, dok pogledi i pohranjene procedure omogućuju složenije akcije pri promjeni podataka. Također se nad tablicama mogu definirati i ograničenja kako bi ostao očuvan integritet baze podataka.

Opis tablica

user_roles

Tablica sadrži sve definirane uloge.

Atribut	Tip podatka	Opis varijable
roleId(PK)	SERIAL	Jedinstveni identifikator uloge
roleName	VARCHAR(20)	Jedinstveno ime uloge

users

Sadrži najopćenitiji pogled na korisnika u sustavu. Svi korisnici imaju podatke u ovoj tablici. Za administratora je dovoljna samo ova tablica.

Atribut	Tip podatka	Opis varijable
userId(PK)	SERIAL	Jedinstveni identifikator korisnika
email	VARCHAR(50)	Jedinstvena e-mail adresa korisnika
firstName	VARCHAR(30)	Ime korisnika
lastName	VARCHAR(30)	Prezime korisnika
roleId(FK)	INT	Jedinstveni identifikator uloge

owners

Tablica sadrži specifične podatke o vlasnicima terena. Tablice koje se odnose isključivo na vlasnika a ne na općeg korisnika se referenciraju na owners tablicu.

Atribut	Tip podatka	Opis varijable
userId(PK,FK)	INT	Jedinstveni identifikator korisnika
phoneNumber	VARCHAR(50)	Telefonski broj vlasnika terena
membershipExpirationDate	DATE	Datum isteka članarine

players

Tablica sadrži specifične podatke o igračima. Tablice koje se odnose isključivo na igrača a ne na općeg korisnika se referenciraju na players tablicu. U 2. reviziji kada se kreće raditi sustav za obavijesti ova tablica bi mogla biti potrebna, trenutačno se samo koristi pri registraciji igrača.

Atribut	Tip podatka	Opis varijable
userId(PK,FK)	INT	Jedinstveni identifikator korisnika
isSubscribedToTournaments	BOOLEAN	Prijavljenost korisnika na obavijesti o novim turnirima

courts

Sadrži podatke o terenima te sve tablice koje trebaju podatke o terenima se referenciraju na nju.

Atribut	Tip podatka	Opis varijable
courtId(PK)	SERIAL	Jedinstveni identifikator terena
courtName	VARCHAR(50)	Jedinstveno ime terena
location	VARCHAR(50)	Lokacija terena
isLeased	BOOLEAN	Opisuje li teren u dnevnom ili je vanjski

isIndoor	BOOLEAN	Opisuje je li teren u dvorani ili je vanjski
image	TEXT	Slika terena u Base64 formatu
userId(FK)	INT	Jedinstveni identifikator korisnika(vlasnika)

tournaments

Sadrži sve podatke o turniru trenutačno je samo konceptualna tablica.

Atribut	Tip podatka	Opis varijable
tournamentId(PK)	SERIAL	Jedinstveni identifikator turnira
tournamentName	VARCHAR(50)	Jedinstveno ime turnira
date	DATE	Datum održavanja turnira
registrationFee	NUMERIC(5,2)	Cijena kotizacije turnira
reward	NUMERIC(5,2)	Nagrada za pobednika u eurima
playerLevel	INT	Razina igrača koji bi trebali sudjelovati u turniru
description	VARCHAR(500)	Opis turnira
isOpen	BOOLEAN	Opisuje je li turnir otvoren ili ne(završio)
results	VARCHAR(1000)	Zapis rezultata turnira po volji vlasnika
image	TEXT	Slika terena u Base64 formatu
userId(FK)	INT	Jedinstveni identifikator korisnika(vlasnika)
courtId(FK)	INT	Jedinstveni identifikator terena

bookings

Sadrži podatke vezane uz termine. Još nije određeno treba li postojati ova tablica.

Atribut	Tip podatka	Opis varijable
bookingId(PK)	SERIAL	Jedinstveni identifikator termina
startTimestamp	TIMESTAMP	Početak termina
endTimestamp	TIMESTAMP	Kraj termina
price	NUMERIC(5,2)	Cijena rezervacije termina
userId(FK)	INT	Jedinstveni identifikator korisnika(igrača) koji je rezervirao termin
courtId(FK)	INT	Jedinstveni identifikator terena

images

Sadrži podatke vezane uz slike koje se stavljuju na završene turnire.

Atribut	Tip podatka	Opis varijable
imageId(PK)	SERIAL	Jedinstveni identifikator slike
imageContent	BYTEA	Binarni sadržaj slike
uploadTime	TIMESTAMP	Vrijeme postavljanja slike
userId(FK)	INT	Jedinstveni identifikator korisnika(igrača ili vlasnika) koji je postavio sliku na turnir
tournamentId(FK)	INT	Jedinstveni identifikator turnira na kojem je slika

comments

Sadrži podatke vezane uz komentare koje se stavljuju na završene turnire.

Atribut	Tip podatka	Opis varijable
commentId(PK)	SERIAL	Jedinstveni identifikator komentara
content	VARCHAR(500)	Binarni sadržaj slike
uploadTime	TIMESTAMP	Vrijeme postavljanja komentara
userId(FK)	INT	Jedinstveni identifikator korisnika(igrača) koji je postavio komentar na turnir
tournamentId(FK)	INT	Jedinstveni identifikator turnira na kojem je slika

notifications

Tablica modelira obavijesti tako da grupira igrače koje su se preplatili na turnire te turnir koji je objavljen nakon što su se igrači preplatili na obavijesti, moguće da je potrebno dodati i vremenski zapis o vremenu objave turnira da igrač može vidjeti kada je objavljen turnir.

Atribut	Tip podatka	Opis varijable
notificationId(PK)	SERIAL	Jedinstveni identifikator obavijesti
isRead	BOOLEAN	Opisuje je li korisnik pročitao obavijest
creationTime	TIMESTAMP	Vrijeme stvaranja obavijesti
userId(FK)	INT	Jedinstveni identifikator korisnika(igrača) koji se preplatio na obavijesti
tournamentId(FK)	INT	Jedinstveni identifikator turnira koji je objavljen

participateIn

Sadrži podatke vezane uz prijave na turnire. Moguće da je potrebno dodati i vremenski zapis o vremenu prijave na turnir da vlasnici vide redoslijed prijava.

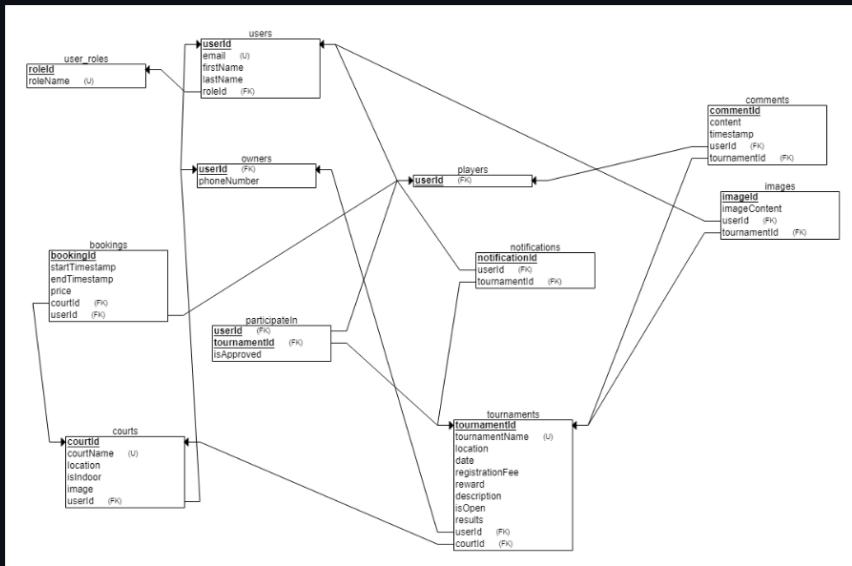
Atribut	Tip podatka	Opis varijable
isApproved	BOOLEAN	Status odobravanja prijave
isDenied	BOOLEAN	Status odbijanja prijave
signUpTime	BOOLEAN	Vrijeme prijave
userId(PK, FK)	INT	Jedinstveni identifikator korisnika(igraca) koji se prijavio na turnir
tournamentId(PK, FK)	INT	Jedinstveni identifikator turnira na koji se igrač prijavio

membership_price

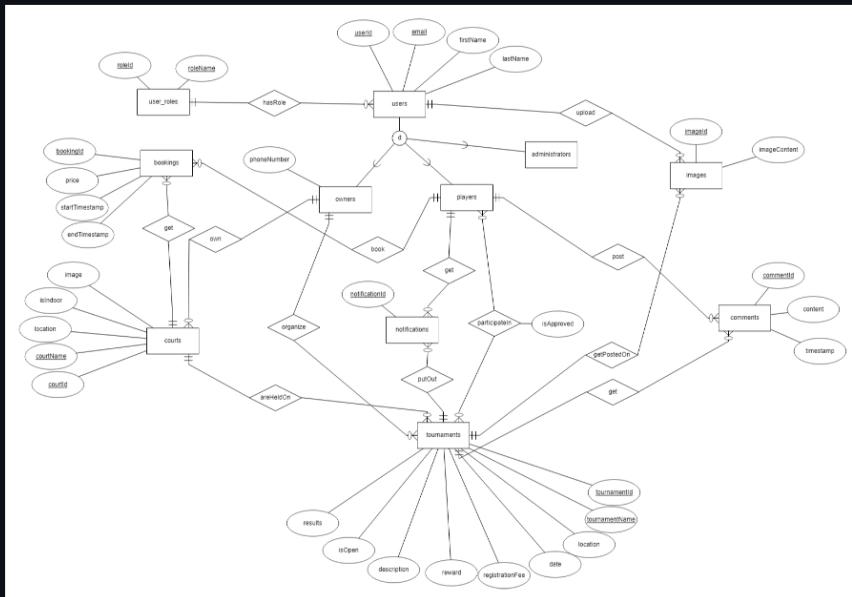
Sadrži cijenu članstva.

Atribut	Tip podatka	Opis varijable
membershipPrice	DECIMAL(6,2)	Cijena članstva

Dijagram baze podataka



Slika 3.2: REL dijagram baze podataka



Slika 3.3: ER dijagram baze podataka

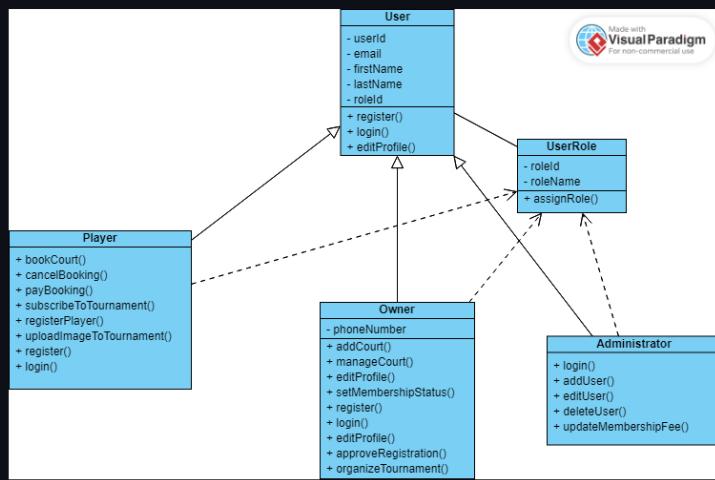
Dijagram razreda

Prvi dijagram (Slika 3.4) prikazuje upravljanje korisnicima u sustavu. Klasa User sadrži osnovne podatke o korisnicima, dok se specifične uloge, poput administratora(Administrator), igrača(Player) i vlasnika(Owner), dodaju korisnicima kroz klasu UserRole. Administrator ima proširena prava za upravljanje korisnicima, sadržajem i funkcijama sustava, dok Player i Owner imaju specifične funkcionalnosti vezane uz sudjelovanje u turnirima i upravljanje terenima. Povezana klasa ostvaruje se putem

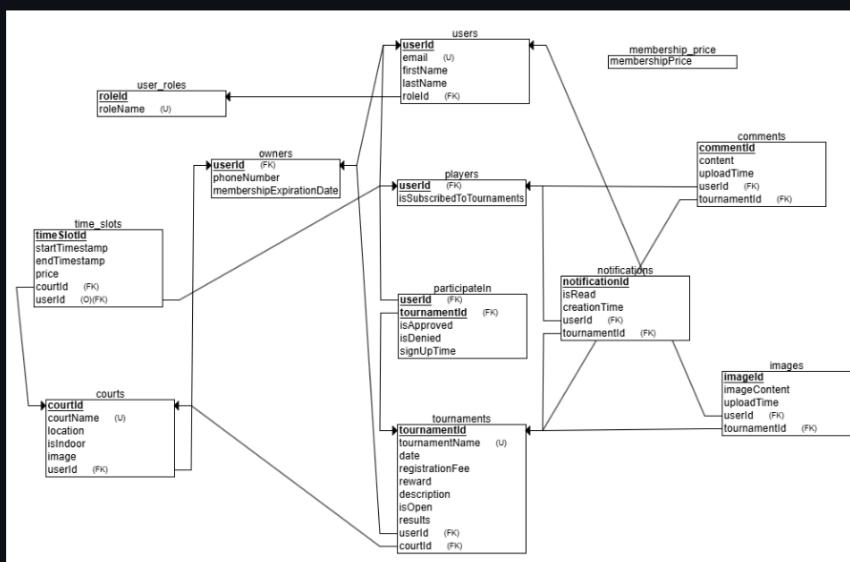
naslijedivanja, asocijacija i ovisnosti.

Druzi dijagram (Slika 3.5) fokusira se na upravljanje sadržajem unutar aplikacije, uključujući Court (terene), Tournament (turnire), rezervacije (Booking), komentare (Comment), obavijesti (Notification) i slike (Image). Klase Booking i Tournament omogućavaju korisnicima rezervaciju terena, dok klase Court i Tournament omogućuje rezervaciju turnira. Comments omogućuju korisnicima ostavljanje povratnih informacija, a Notifications služe za obaveštanje korisnika o novim dogadanjima. Images su povezane s terenima (Court) za prikaz slika terena.

Veza između dijagrama uspostavljena je kroz uloge korisnika. Igrač (Player) može rezervirati teren (Court) putem klase Booking i sudjelovati u turnirima (Tournament), dok vlasnik (Owner) može upravljati terenima (Court), turnirima (Tournament) i obavijestima (Notifications). Administrator nadzire sve aspekte ova dijagrama, uključujući korisnike, terene, turnire i rezervacije te ima ovlasti uređivanja/dodavanja/brisanja korisnika.



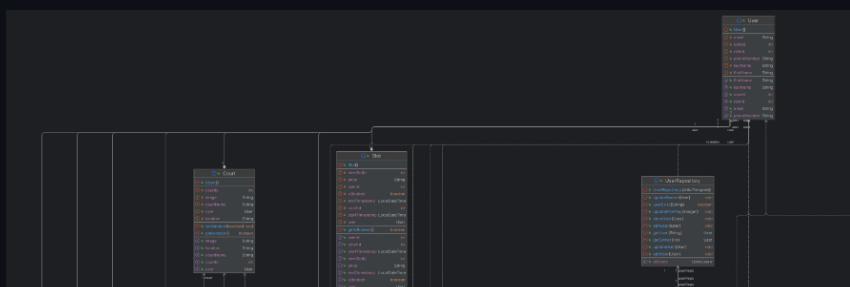
Slika 3.4: Dijagram razreda za upravljanje korisnicima i osnovnim funkcijama

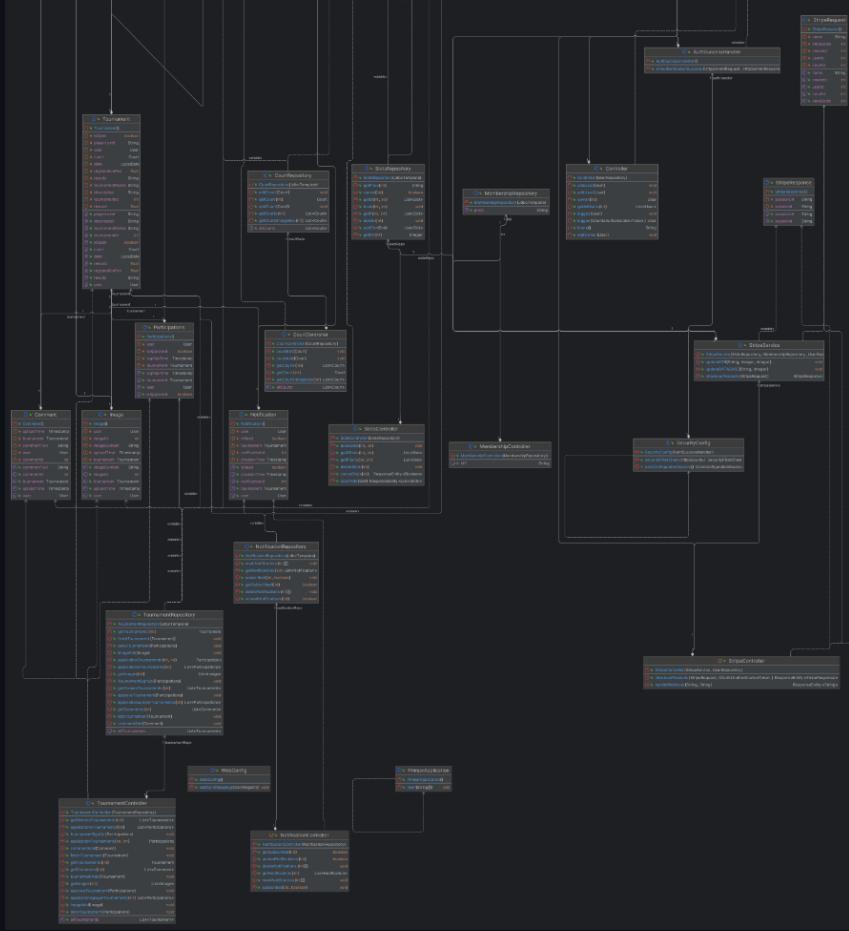


Slika 3.5: Dijagram razreda za upravljanje sadržajem aplikacije

dio 2. revizije

Implementacijski dijagram razreda (Slika 3.6) prikazuje arhitekturu sustava temeljenog na klijent-poslužitelj modelu, obuhvaćajući ključne klase, njihove atribute, metode i odnose među njima. U središtu sustava nalazi se klasa User, koja predstavlja korisnika s atributima poput imena, prezimena, emaila i ID-a, dok klasa UserRepository omogućuje upravljanje podacima o korisnicima kroz metode za spremanje, dohvatanje i ažuriranje. Klase povezane s turnirima, poput Tournament i Participations, omogućuju organizaciju i sudjelovanje u turnirima, dok pripadajući kontroleri i rezpositorij pružaju funkcionalnosti za dodavanje turnira, prijavu sudionika te upravljanje komentarama i slikama. Za upravljanje terenima i rezervacijama, klase Court i Slot predstavljaju sportske terene i vremenske termine, dok kontroleri i rezpositoriji omogućuju njihovo dodavanje, urediranje i dohvatanje. Klase Notification i Comment omogućuju obaveštanje korisnika te upravljanje komentara vezanim uz turnire. Sustav je integriran s vanjskim servisima putem klase StripeService i StripeController, koje upravljaju funkcionalnostima vezanim uz plaćanja. Sve komponente povezane su kroz jasno definirane odnose i metode koje osiguravaju pravilno funkcioniranje sustava.





Slika 3.6: Implementacijski dijagram razreda

Dinamičko ponašanje aplikacije

Dinamičko ponašanje aplikacije odnosi se na način na koji objekti u sustavu evoluiraju kroz vrijeme, uključujući prijelaze između različitih stanja. To uključuje aktivnosti, događaje, odluke i interakcije unutar aplikacije. UML dijagrami stanja omogućuju vizualizaciju tih promjena i olakšavaju razumijevanje dinamike sustava.

Razumijevanje promjena stanja neophodno je za pravilno funkcioniranje aplikacije jer pruža uvid u interakcije među objektima, komponentama i korisnicima tijekom rada sustava. Korištenjem UML dijagrama stanja i aktivnosti moguće je vizualizirati prijelaze i stanja objekata, identificirati potencijalne probleme, osigurati točnu implementaciju te poboljšati komunikaciju među članovima tima.

UML dijagrami stanja

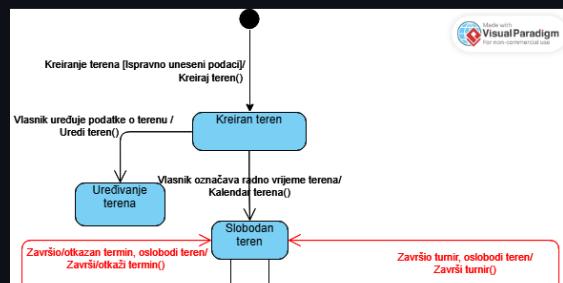
UML dijagrami stanja nužni su za razumijevanje dinamičkog ponašanja sustava. Oni jasno prikazuju promjene stanja objekata tijekom vremena ovisno o događajima i uvjetima.

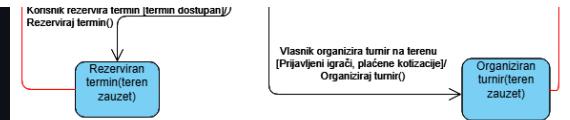
Ovaj dijagram (Slika 3.7) opisuje proces upravljanja terenima, počevši od početnog stanja "Nepostojeći teren". Kada korisnik odluči kreirati novi teren, proces prelazi u stanje "Kreiran teren". Ovo stanje omogućava osnovno postavljanje terena i služi kao polazišna točka za daljnje upravljanje.

Iz stanja "Kreiran teren" moguće su tri prijelaza. Prvi prijelaz vodi u stanje "Uređivanje terena", koje omogućava izmjene podataka o terenu, poput naziva, veličine ili dodatnih informacija. Drugi prijelaz, izrada kalendara za teren, omogućuje prelazak u stanje "Slobodan teren". Ovo stanje označava teren koji je spremан за rezervaciju termina ili organizaciju turnira. Treći prijelaz iz stanja "Kreiran teren" vodi prema završetku "Teren obrisan", što označava brisanje terena iz sustava.

U stanju "Slobodan teren" omogućena je interakcija korisnika s terenom. Korisnik može rezervirati termin, čime teren prelazi u stanje "Rezerviran termin", ili organizirati turnir, čime teren prelazi u stanje "Organiziran turnir". Oba stanja predstavljaju situacije u kojima je teren zauzet i nije dostupan za druge aktivnosti.

Nakon završetka turnira ili termina ili otkazivanja termina, teren se vraća u stanje "Slobodan teren", čime se ponovo omogućava njegova dostupnost za buduće rezervacije ili događaje. Na taj način dijagram prikazuje cijeli životni ciklus upravljanja terenom, od njegovog kreiranja i prilagodbe, preko rezervacija i organizacije događaja, do eventualnog završetka ili brisanja iz sustava.



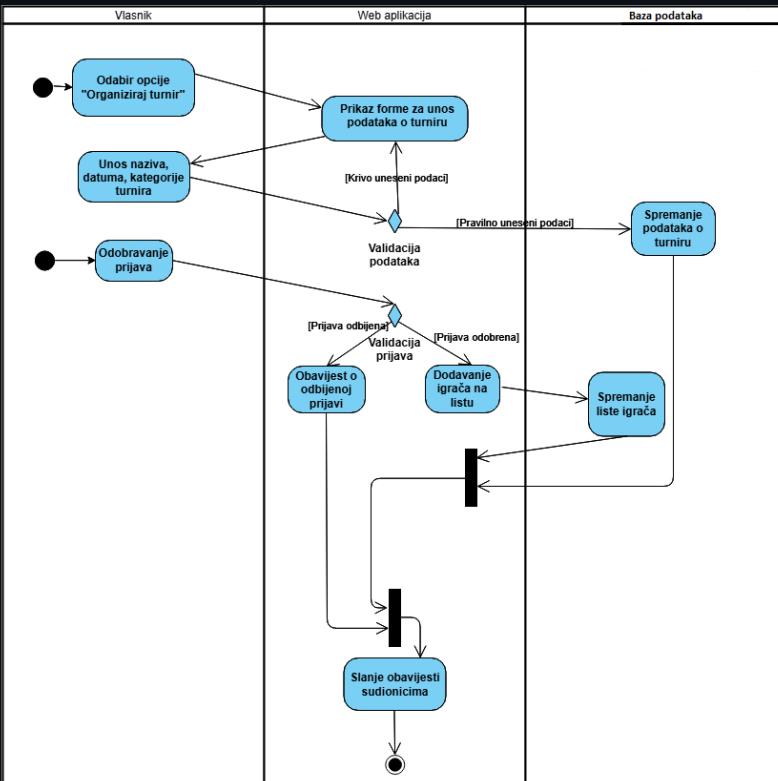


Slika 3.7: Dijagram stanja

UML dijagrami aktivnosti

Dijagram aktivnosti prikazuje tijek izvršavanja određenog procesa. Osim za razumijevanje toka podataka unutar aplikacije, koristi se za analizu poslovnih procesa.

Dijagram aktivnosti na Slici 3.8 prikazuje proces organiziranja turnira u aplikaciji Playpadel, podijeljen na dva glavna toka. U prvom toku, proces započinje s odabirom opcije "Organiziraj turnir", gdje se korisniku prikazuje forma za unos podataka o turniru (naziv, datum, kategorija). Nakon unosa podataka slijedi upravljački čvor: ako su podaci pravilno uneseni, spremaju se u bazu, dok se u slučaju greške korisniku vratiča forma za ispravak. U drugom toku, odobravanje prijava igrača uključuje upravljački čvor: prijava može biti odbijena, uz obavijest prijavitelju, ili odobrena, čime se igrač dodaje na listu i obaveštava, te se lista zatim sprema u bazu. Naposjetku proces organiziranja turnira završava završnim tokom.



Slika 3.8: Dijagram aktivnosti

franogrinsak / Tim182

Code Issues Pull requests Actions Projects Wiki Security Insights

5. Arhitektura komponenata i razmještaja

Filip Šturić edited this page last week · 2 revisions

Arhitektura sustava predstavlja temeljni okvir za razumijevanje i implementaciju svih njegovih funkcionalnosti. U kontekstu razvojne dokumentacije aplikacija, dijagrami komponenata i razmještaja odlučujući su za prikaz povezanosti i rasporeda različitih komponenata sustava. Ovi dijagrami omogućuju sudionicima projekta razumijevanje i vizualizaciju fizičkog i logičkog dizajna sustava, uključujući interakcije između dijelova aplikacije, što je odlučujuće za efikasnu implementaciju i dugoročnu održivost sustava.

Arhitektura sustava, u kontekstu dijagrama komponenata i razmještaja, pruža uvid u strukturu i raspored ključnih dijelova aplikacije. Ovi dijagrami nisu korisni samo tijekom faza oblikovanja i implementacije, već služe i kao alati za održavanje i optimizaciju sustava u budućnosti.

Kao dio razvoja aplikacije, važno je osmislit i dokumentirati arhitekturu sustava s naglaskom na dijagramu komponenata i razmještaja. Vaš zadatak je izraditi **dijagram komponenata** koji će jasno prikazivati ključne funkcionalne komponente aplikacije, njihovu međusobnu povezanost te sučelja za komunikaciju. Također, trebate izraditi **dijagram razmještaja** komponenata, koji treba detaljno prikazivati kako su te komponente raspoređene u infrastrukturi sustava, uključujući fizičke i virtualne resurse poput poslužitelja ili uredaja krajnjih korisnika.

Dijagram komponenata

Komponente sustava predstavljaju bitne dijelove aplikacije koji obavljaju specifične funkcije. Svaka komponenta je autonomna jedinica sa vlastitim odgovornostima, ali je povezana s drugim komponentama kako bi sustav u cijelini funkcioniраo. Komponente mogu biti elementi poput modula, servisa, razreda ili paketa, te komuniciraju putem jasno definiranih sučelja. Naš dijagram komponenata, vidljiv na slici 4.1, prikazuje četiri ključne komponente sustava: Frontend Web Prikaz, Web Aplikacija PlayPadel, React API i PostgreSQL Baza podataka. Frontend Web Prikaz s komponentom "View" implementiranom u Reactu pruža HTML, CSS i JS za prikaz te JSON za backend komunikaciju, povezivanjem s PlayPadel aplikacijom putem REST API-ja. Web Aplikacija PlayPadel služi kao centralni dio sustava s navigacijom kroz React Router i modulima poput Register, EditProfile i Membership, spajajući se na React API za autentifikaciju (Google Auth), upravljanje terminima (Google Calendar) i plaćanja (Stripe), dok dohvaća i sprema podatke u PostgreSQL bazu podataka putem Repositories i Controllers. Ova arhitektura omogućava učinkovitu integraciju korisničkog sučelja, backend logike, vanjskih servisa i baze podataka.

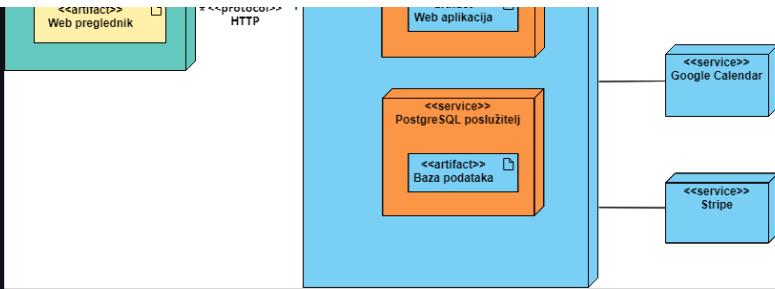
Slika 4.1: Dijagram komponenata

Dijagram razmještaja

UML dijagram razmještaja prikazuje fizičku ili virtualnu raspodjelu komponenata sustava unutar infrastrukture. Cilj je prikazati kako su komponente raspoređene (npr. na poslužiteljima, u okruženjima oblaka ili na uredajima krajnjih korisnika) te način komunikacije, API-ja ili drugih komunikacijskih protokola.

U našoj aplikaciji sustav koristi arhitekturu "klijent - poslužitelj" i sastoji se od klijentskog i poslužiteljskog računala. Na klijentskom računalu nalazi se web preglednik koji korisnicima, poput igrača, vlasnika i administratora, omogućava pristup web aplikaciji. Klijentsko računalo komunicira s poslužiteljskim računalom putem HTTP protokola. Na poslužiteljskom računalu nalaze se web poslužitelj, koji pokreće web aplikaciju, i poslužitelj baze podataka (PostgreSQL), zadužen za pohranu i dohvat podataka. Web poslužitelj i baza podataka suradjuju kako bi osigurali nesmetano funkcioniranje sustava.

Dodatano, poslužiteljsko računalo koristi vanjske servise, kao što su Google Auth za autentifikaciju korisnika, Google Calendar za upravljanje terminima te Stripe za procesiranje plaćanja. Ovi servisi su integrirani s web poslužiteljem, koji posreduje između njih i klijentskog računala. Web poslužitelj koristi API-je ovih servisa za sigurno slanje zahtjeva i obradu odgovora te omogućava korisnicima pristup funkcionalnostima vanjskih servisa putem web aplikacije.



Slika 4.2: Dijagram razmještaja

Tim182



© 2025 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact](#) [Manage cookies](#) [Do not share my personal information](#)

6. Ispitivanje programskog rješenja

Fran Ogrinšak edited this page 53 minutes ago · 17 revisions

[Edit](#) [New page](#)

Ispitivanje komponenti

1. Ispitni slučaj:Dodavanje terena

Funkcionalnost: Ispituјemo funkcionalnost dodavanja terena, odnosno funkciju addCourt klase CourtRepository

- Ime i klasa testa u kodu: `Ime - testaddCourt`, Klasa - `CourtRepoTests`

- Ulazni podaci:

Teren (`court`) s atributima:

- `courtName = "teren1"`
- `location = "lokacija1"`
- `isIndoor = true`
- `userId = 1`

- Očekivani rezultati:

Dohvaćeni teren (`court`) s istim atributima.

- Dobiveni rezultati:

Dohvaćeni teren s očekivanim atributima – prolaz testa.

- Tijek ispitivanja:

- i. Stvaramo objekt `court` s navedenim atributima.
- ii. Pomoću funkcije `courtRepo.addCourt(court)` dodajemo teren.
- iii. Dohvaćamo teren funkcijom `courtRepo.getCourtByName("teren1")`.
- iv. Provjeravamo je li ime dohvaćenog terena jednako dodanom terenu pomoću `assertEquals`.
- v. Ako su ista, test je prolazan, ako nisu test je ne prolazan.

Pages 13
<input type="text"/> Find a page...
▶ Home
▶ 1. Opis projektnog zadatka
▶ 2. Analiza zahtjeva
▶ 3. Specifikacija programske potpore
▶ 4. Arhitektura i dizajn sustava
▶ 5. Arhitektura komponenata i razmješ...
▶ 6. Ispitivanje programskog rješenja
Ispitivanje komponenti
1. Ispitni slučaj:Dodavanje terena
2. Ispitni slučaj: Uređivanje terena
3. Ispitni slučaj: Nepostojeća funkcionalnost
4. Ispitni slučaj: Dodavanje termina
5. Ispitni slučaj: Rubni uvjeti
6. Ispitni slučaj: Rezervacija termina
7. Ispitni slučaj: Otkazivanje termina
8. Ispitni slučaj: Izazivanje pogreške
9. Ispitni slučaj: Nepostojeća funkcionalnost
Slike rezultata
Ispitivanje sustava
Slike rezultata
▶ 7. Tehnologije za implementaciju apli...
▶ 8. Upute za puštanje u pogon
▶ 9. Zaključak i budući rad
▶ A. Popis literature
▶ B. Dnevnik promjena dokumentacije
▶ C. Prikaz aktivnosti grupe

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/Franogrinsak/Tim1>

2. Ispitni slučaj: Uređivanje terena

Funkcionalnost: Ispituјemo funkcionalnost uređivanja terena, odnosno funkciju editCourt klase CourtRepository

- Ime i klasa testa u kodu: `Ime - testEditCourt`, Klasa - `CourtRepoTests`

- Ulazni podaci:

Teren (`court`) s atributima:

- `courtId = 10`
- `courtName = "terenEdit1234"`
- `location = "lokacijaEdit1234"`
- `image = "aaaaaa"`
- `isIndoor = true`
- `userId = 1`

- Očekivani rezultati:

Dohvaćeni teren s istim atributima.

- Dobiveni rezultati:

Dohvaćeni teren s očekivanim atributima – prolaz testa.

- Tijek ispitivanja:

- i. Stvaramo objekt `court` s navedenim atributima.
- ii. Pomoću funkcije `courtRepo.editCourt(court)` uređujemo teren.
- iii. Dohvaćamo teren funkcijom `courtRepo.getCourt(10)`.
- iv. Provjeravamo atribute dohvaćenog terena s atributima stvorenog objekta pomoću `assertEquals`.
- v. Ako su isti, test je prolazan, ako nisu test je ne prolazan.

3. Ispitni slučaj: Nepostojeća funkcionalnost

Funkcionalnost: Ispituјemo slučaj nepostojeće funkcionalnosti, odnosno funkciju deleteCourt klase CourtRepository

- Ime i klasa testa u kodu: `Ime - testDeleteCourt`, Klasa - `CourtRepoTests`

- Ulazni podaci: Nema.

- Očekivani rezultati: `UnsupportedOperationException` greška.

- Dobiveni rezultati: `UnsupportedOperationException` greška – prolaz testa.

- Tijek ispitivanja:

- i. Definiramo funkciju `deleteCourt` koja vraća `UnsupportedOperationException`.
- ii. Pozivamo funkciju `deleteCourt`.
- iii. Provjeravamo je li funkcija vratila grešku pomoću `assertThrows`.

4. Ispitni slučaj: Dodavanje termina

Funkcionalnost: Ispitujuemo funkcionalnost dodavanja termina terena, odnosno funkciju addSlot klase SlotsRepository.

- Ime i klasa testa u kodu: **Ime** - testAddSlot, **Klasa** - SlotRepoTest
- Ulazni podaci:
Termin (`slot`) s atributima:
 - `startTimestamp` = "2025-03-25 08:30:00"
 - `endTimestamp` = "2025-03-25 09:30:00"
 - `courtId` = 1
 - `price` = 5.0
- Očekivani rezultati:
Dohvaćeni termin s istim atributima.
- Dobiveni rezultati:
Dohvaćeni termin s očekivanim atributima – **prolaz testa**.
- Tijek ispitivanja:
 - i. Stvaramo objekt `slot` s navedenim atributima.
 - ii. Dodajemo termin pomoću `slotRepo.addSlot(slot)`.
 - iii. Dohvaćamo termin funkcijom `slotRepo.getSlot`.
 - iv. Provjeravamo atribute dohvaćenog termina s atributima stvorenog termina pomoću `assertEquals`.
 - v. Ako su isti, test je prolazan, ako nisu test je ne prolazan.

5. Ispitni slučaj: Rubni uvjeti

Funkcionalnost: Ispitujuemo rubne uvjete na primjeru dodavanja termina terena, odnosno funkciju addSlot klase SlotsRepository.

- Ime i klasa testa u kodu: **Ime** - testEdgeAddSlot, **Klasa** - SlotRepoTest
- Ulazni podaci:
Termin (`slot`) s atributima:
 - `startTimestamp` = "2025-03-18 12:00:00"
 - `endTimestamp` = "2025-03-18 12:30:00"
 - `courtId` = 1
 - `price` = 5.0
- Očekivani rezultati:
Dohvaćeni termin s istim atributima.
- Dobiveni rezultati:
Dohvaćeni termin s očekivanim atributima – **prolaz testa**.
- Tijek ispitivanja:
 - i. U bazi postoji termin: ('2025-03-18 08:30:00', '2025-03-18 12:00:00', 12.00, 1, NULL).
 - ii. Dodajemo novi termin pomoću `slotRepo.addSlot(slot)`.
 - iii. Dohvaćamo termin funkcijom `slotRepo.getSlot`.
 - iv. Provjeravamo atribute dohvaćenog termina s atributima stvorenog termina pomoću `assertEquals`.
 - v. Ako su isti, test je prolazan, ako nisu test je ne prolazan.

6. Ispitni slučaj: Rezervacija termina

Funkcionalnost: Ispitujuemo funkcionalnost rezervacije termina, odnosno funkciju book klase SlotsRepository.

- Ime i klasa testa u kodu: **Ime** - testBookSlot, **Klasa** - SlotRepoTest
- Ulazni podaci:
 - `timeSlotId` = 5
 - `userId` = 3
- Očekivani rezultati:
`timeSlotId` = 5
- Dobiveni rezultati:
`timeSlotId` = 5 – **prolaz testa**.
- Tijek ispitivanja:
 - i. Rezerviramo termin pomoću `slotRepo.book(timeSlotId, userId)`.
 - ii. Dohvaćamo termin pomoću `slotRepo.getSlot(timeSlotId)`.
 - iii. Provjeravamo je li `timeSlotId` dohvaćenog termina jednak očekivanom pomoću `assertEquals`.
 - iv. Ako su isti, test je prolazan, ako nisu test je ne prolazan.

7. Ispitni slučaj: Otkazivanje termina

Funkcionalnost: Ispitujuemo funkcionalnost otkazivanja termina, odnosno funkciju cancel klase SlotsRepository.

- Ime i klasa testa u kodu: Ime - testCancelSlot, Klasa - SlotRepoTest
- Ulazni podaci:
 - timeSlotId = 6
- Očekivani rezultati:
 - userId = 0
- Dobiveni rezultati:
 - userId = 0 – prolaz testa.
- Tijek ispitivanja:
 - i. Otkazujemo termin pomoću slotRepo.cancel(timeSlotId).
 - ii. Dohvaćamo termin pomoću slotRepo.getSlot(timeSlotId).
 - iii. Provjeravamo je li userId jednak 0 pomoći assertEquals.
 - iv. Ako su isti, test je prolazan, ako nisu test je ne prolazan.

8. Ispitni slučaj: Izazivanje pogreške

Funkcionalnost: Ispitujemo izazivanje pogreške na primjeru funkcionalnosti otkazivanja termina, odnosno funkciju cancel klase SlotsRepository.

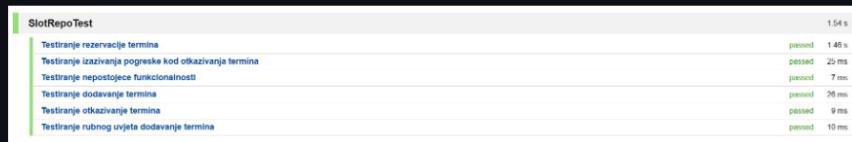
- Ime i klasa testa u kodu: Ime - testCancelErrorSlot, Klasa - SlotRepoTest
- Ulazni podaci:
 - timeSlotId = 4
- Očekivani rezultati:
 - UnsupportedOperationException greška.
- Dobiveni rezultati:
 - UnsupportedOperationException greška – prolaz testa.
- Tijek ispitivanja:
 - i. Termin u bazi: (CURRENT_TIMESTAMP + INTERVAL '10' HOUR, CURRENT_TIMESTAMP + INTERVAL '13' HOUR, 12.00, 1, 3)-do početka termina je manje od 24 sata.
 - ii. Pokušaj otkazivanja termina pomoću slotRepo.cancel(timeSlotId).
 - iii. Provjera je li funkcija vratila grešku UnsupportedOperationException pomoći assertThrows - zbog toga što se ne može oktazat unutar 24 sata prije.
 - iv. Ako je vratila grešku test je prolazan, ako nije onda je ne prolazan.

9. Ispitni slučaj: Nepostojeća funkcionalnost

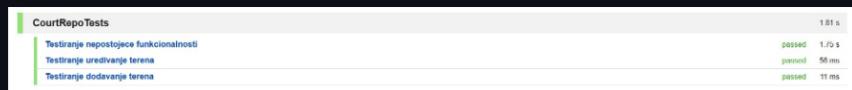
Funkcionalnost: Ispitujemo slučaj nepostojeće funkcionalnosti, odnosno funkciju brojTermina klase CourtRepository.

- Ime i klasa testa u kodu: Ime - testUnknownSlot, Klasa - SlotRepoTest
- Ulazni podaci: Nema.
- Očekivani rezultati:
 - UnsupportedOperationException greška.
- Dobiveni rezultati:
 - UnsupportedOperationException greška – prolaz testa.
- Tijek ispitivanja:
 - i. Definiramo funkciju brojTermina koja vraća UnsupportedOperationException.
 - ii. Provjeravamo je li funkcija vratila grešku pomoći assertThrows .
 - iii. Ako je vratila grešku test je prolazan, ako nije onda je ne prolazan.

Slike rezultata



Slika 6.1: Rezultati ispitivanja termina



Slika 6.2: Rezultati ispitivanja terena

Ispitivanje sustava

Upute za pokretanje testova na operacijskom sustavu Windows: Potrebno je instalirati ChromeDriver. Nakon toga otvoriti cmd te pozicionirati se u lokaciju gdje je chrome.exe smješten. Pokrenuti naredbu: chrome.exe --remote-debugging-port=8989 --user-data-dir="putanja". Putanja može biti bilo koji direktorij koji će spremati podatke. U novo otvoreni Chrome prozor potrebno je ulogirati se u Google račune od vlasnika i administratora. Nakon toga se mogu pokrenuti JUnit Selenium testovi.

1. Dodavanje terena:

- o JUnit ime: `testAddCourt`

- o Ulaz: slika terena, Court Name(slučajno generirano), Location = "location", Setting = "Outdoor"

- o Koraci:

- Prijaviti se u aplikaciju kao vlasnik terena.
- Pozicionirati se u dodavanje terena u korisničkom sučelju.
- Unijeti podatke u formu za dodavanje terena.
- Provjeriti je li vlasnik preusmjeren na popis njegovih terena.

- o Očekivani izlaz: domenaAplikacije + "app/courts/1"

2. Rubni uvjet - dodavanje terena s postojećim imenom:

- o JUnit ime: `testAddDuplicateCourtName`

- o Ulaz: slika terena, Court Name(slučajno generirano), Location = "location", Setting = "Outdoor"

- o Koraci:

- Obaviti korake kao u testu s dodavanjem terena s slučajno generiranim imenom terena.
- Ponoviti korake za dodavanje ali s istim slučajnim imenom.
- Provjeriti je li prikazana poruka o grešci.

- o Očekivani izlaz: "Failed to organize tournament, reason: That tournament name alrady exists"

3. Nepostojeća funkcionalnost - pretraživanje nepostojećeg identifikatora terena upisivanjem u adresnu traku:

- o JUnit ime: `testNonExistentCourt`

- o Ulaz: domenaAplikacije + "app/courts/1/1000"

- o Koraci:

- Prijaviti se u aplikaciju kao vlasnik.
- Upisati u adresnu traku ulazni podatak.
- Provjeriti je li prikazana poruka o grešci.

- o Očekivani izlaz: "Error"

4. Organiziranje turnira:

- o JUnit ime: `organizeTournament`

- o Ulaz: Tournament name(slučajno generirano), Player Level = "Beginner", Description = "description", Registration fee = "5.00", Reward = "5.00", courtId = "2", date(1 dan u budućnosti od datuma pokretanja testa)

- o Koraci:

- Prijaviti se u aplikaciju kao vlasnik.
- Pozicionirati se u organiziranje turnira u korisničkom sučelju.
- Unijeti podatke u formu za organiziranje turnira.
- Provjeriti je li vlasnik preusmjeren na popis njegovih turnira.

- o Očekivani izlaz: domenaAplikacije + "app/tournaments/1"

5. Rubni uvjet - organiziranje turnira s istim imenom:

- o JUnit ime: `organizeTournamentDuplicateName`

- o Ulaz: Tournament name(slučajno generirano), Player Level = "Beginner", Description = "description", Registration fee = "5.00", Reward = "5.00", courtId = "3", date(1 dan u budućnosti od datuma pokretanja testa)

- o Koraci:

- Obaviti korake kao u testu s organiziranjem turnira s slučajno generiranim imenom turnira.
- Ponoviti korake za organiziranje, ali s istim slučajnim imenom i courtId = "4".
- Provjeriti je li prikazana poruka o grešci.

- o Očekivani izlaz: "Failed to organize tournament, reason: That tournament name alrady exists"

6. Rubni uvjet - organiziranje turnira s datumom održavanja u prošlosti:

- o JUnit ime: `organizeTournamentDateInThePast`

- o Ulaz: Tournament name(slučajno generirano), Player Level = "Beginner", Description = "description", Registration fee = "5.00", Reward = "5.00", courtId = "5", date(1 dan u prošlosti od datuma pokretanja testa)

- o Koraci:

- Prijaviti se u aplikaciju kao vlasnik.
- Pozicionirati se u organiziranje turnira u korisničkom sučelju.
- Unijeti podatke u formu za organiziranje turnira.
- Provjeriti je li prikazana poruka o grešci.

- o Očekivani izlaz: "Date has to be in the future."

7. Dodavanje korisnika:

- JUnit ime: `testAddUser`
- Ulaz: First Name = "John", Last Name = "Tester", E-mail(slučajno generiran) Court Name(slučajno generirano), roleId = "2"
- Koraci:
 - a. Prijaviti se u aplikaciju kao administrator.
 - b. Pozicionirati se u dodavanje korisnika u korisničkom sučelju.
 - c. Unijeti podatke u formu za dodavanje korisnika.
 - d. Provjeriti je li administrator preusmjeren na popis svih korisnika.
- Očekivani izlaz: domenaAplikacije + "app/users"

8. Rubni uvjet - dodavanje korisnika s postojećim E-mailom:

- JUnit ime: `testAddUserDuplicateEmail`
- Ulaz: First Name = "John", Last Name = "Tester", E-mail(slučajno generiran) Court Name(slučajno generirano), roleId = "2"
- Koraci:
 - a. Obaviti korake kao u testu s dodavanjem korisnika sa slučajno generiranim E-mailom.
 - b. Ponoviti korake za dodavanje ali s istim slučajnim E-mailom.
 - c. Provjeriti je li prikazana poruka o grešci.
- Očekivani izlaz: "Failed to add the user: That email is already being used"

9. Postavljanje termina:

- JUnit ime: `testAddTimeSlot`
- Ulaz: startDate(2 dana u budućnosti od datuma pokretanja testa), startTime = "03:00PM", endDate = startDate endTime = "04:00PM", Price = "5.00"
- Koraci:
 - a. Prijaviti se u aplikaciju kao vlasnik.
 - b. Pozicionirati se u postavljanje termina u korisničkom sučelju.
 - c. Unijeti podatke u formu za postavljanje termina.
 - d. Provjeriti je li zatvoren obrazac za postavljanje termina.
- Očekivani izlaz: "success"

10. Rubni uvjet - postavljanje termina koji počinje u prošlosti:

- JUnit ime: `testPastDate`
- Ulaz: startDate(1 dan u prošlosti od datuma pokretanja testa), startTime = "03:00PM", endDate = startDate endTime = "04:00PM", Price = "5.00"
- Koraci:
 - i. Prijaviti se u aplikaciju kao vlasnik.
 - ii. Pozicionirati se u postavljanje termina u korisničkom sučelju.
 - iii. Unijeti podatke u formu za postavljanje termina.
 - iv. Provjeriti je li prikazana poruka o grešci.
- Očekivani izlaz: "Starting date and time has to be at least 30 minutes after the current time."

11. Rubni uvjet - postavljanje termina s kratkim trajanjem:

- JUnit ime: `testShortSlot`
- Ulaz: startDate(2 dana u budućnosti od datuma pokretanja testa), startTime = "03:00PM", endDate = startDate endTime = "03:02PM", Price = "5.00"
- Koraci:
 - i. Prijaviti se u aplikaciju kao vlasnik.
 - ii. Pozicionirati se u postavljanje termina u korisničkom sučelju.
 - iii. Unijeti podatke u formu za postavljanje termina.
 - iv. Provjeriti je li prikazana poruka o grešci.
- Očekivani izlaz: "Starting date and time has to be at least 30 minutes after the current time."

12. Rubni uvjet - postavljanje termina s dugim trajanjem:

- JUnit ime: `testAddTimeSlotBig`
- Ulaz: startDate(2 dana u budućnosti od datuma pokretanja testa), startTime = "03:00PM", endDate(4 dana u budućnosti od datuma pokretanja testa) endTime = "04:00PM", Price = "5.00"
- Koraci:
 - i. Prijaviti se u aplikaciju kao vlasnik.
 - ii. Pozicionirati se u postavljanje termina u korisničkom sučelju.

iii. Unijeti podatke u formu za postavljanje termina.

iv. Provjeriti je li prikazana poruka o grešci.

o **Očekivani izlaz:** "Start and end times have to be 24 hours apart at most."

13. Rubni uvjet - postavljane termina s vremenom početka poslije vremena završetka:

o **JUnit ime:** testPastDate

o **Ulaz:** startDate(2 dana u budućnosti od datuma pokretanja testa), startTime = "04:00PM", endDate = startDate
endTime = "03:00PM", Price = "5.00"

o **Koraci:**

i. Prijaviti se u aplikaciju kao vlasnik.

ii. Poszionirati se u postavljanje termina u korisničkom sučelju.

iii. Unijeti podatke u formu za postavljanje termina.

iv. Provjeriti je li prikazana poruka o grešci.

o **Očekivani izlaz:** "Start date and time has to be earlier than end date and time."

14. Rubni uvjet - postavljane termina koji se preklapa s drugim terminom:

o **JUnit ime:** testPastDate

o **Ulaz:** startDate(3 dana u budućnosti od datuma pokretanja testa), startTime = "02:00PM", endDate = startDate
endTime = "04:00PM", Price = "5.00"

o **Koraci:**

i. Prijaviti se u aplikaciju kao vlasnik.

ii. Poszionirati se u postavljanje termina u korisničkom sučelju.

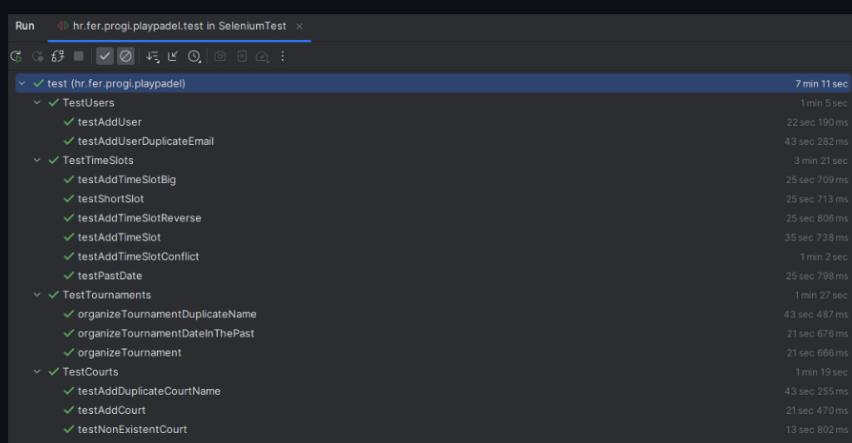
iii. Unijeti podatke u formu za postavljanje termina, ali startTime = "03:00PM".

iv. Ponoviti korake 1-3 ali koristiti startTime s ulaza.

v. Provjeriti je li prikazana poruka o grešci.

o **Očekivani izlaz:** "Failed to add a time slot: there is a conflict with a slot starting at "
startDate + " 15:00 PM"

Slike rezultata



Slika 6.3: Rezultati ispitivanja sustava

Tim182



© 2025 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

7. Tehnologije za implementaciju aplikacije

Filip Šturić edited this page now · [4 revisions](#)

[Edit](#)[New page](#)

Korištenje tehnologije i alati

Komunikacija tima odvijala se uživo te preko aplikacija Microsoft Teams i WhatsApp.

UML dijagrami izrađeni su korištenjem alata Astah (<https://astah.net>) te Visual Paradigm (<https://online.visual-paradigm.com>), koji omogućuju jednostavno modeliranje sustava pomoću dijagrama. Dijagram baze podataka je izrađen u online alatu ERDPlus (<https://erdplus.com>). Kao razvojna okruženja korišteni su Inteliž IDEA (<https://www.jetbrains.com/idea>) i Visual Studio Code (<https://code.visualstudio.com>). Inteliž IDEA (verzija 2023.3.2), napredni IDE za razvoj u Java, omogućuje autokompletiranje, debugiranje i rad s frameworkovima poput Spring Boota. Visual Studio Code (verzija 1.96.4), iako nije punokrni IDE, zbog svojih ekstenzija pruža podršku za razne jezike i latoe poput debuggera i Intellisense-a.

Sustav za kontrolu verzija bio je Git (verzija 2.43.0), a udaljeni rezpozitorij projekta nalazi se na GitHubu. Baza podataka izrađena je u sustavu PostgreSQL (verzija 17.2) (<https://www.postgresql.org>), poznat po svojoj pouzdanosti i bogatim funkcionalnostima.

Backend aplikacije razvijen je korištenjem Spring Boota (verzija 3) (<https://spring.io/projects/spring-boot>), što omogućuje brzo postavljanje i konfiguraciju aplikacije uz princip "konvencija ispred konfiguracije". Frontend je izrađen kombinacijom HTML, CSS i JavaScript, uz korištenje biblioteke React (verzija 18) (<https://react.dev>), koja omogućuje modularni i učinkoviti razvoj korisničkog sučelja. Također React u ovom projektu koristi dodatne biblioteke: PrimeReact (<https://primereact.org>), Material Tailwind (<https://www.material-tailwind.com>) i tailwindcss (<https://tailwindcss.com>).

Aplikacija je deployana na poslužitelju Render (<https://render.com>), koji omogućuje jednostavnu implementaciju aplikacija u oblaku.

Za ispitivanje aplikacije upotrijebljen je Selenium (verzija 4) (<https://www.selenium.dev>), koji omogućuje automatizirano testiranje korisničkog sučelja, te JUnit (verzija 5) (<https://junit.org/junit5>) za jedinično testiranje u Java okruženju.

Također su korišteni vanjski servisi:

- Stripe (<https://stripe.com/en-hr>) za obradu plaćanja, čime se omogućava sigurno i pouzdano upravljanje transakcijama.
- Google OAuth2 za autentifikaciju korisnika putem njihovih Google računa, čime se pojednostavljuje proces prijave i osigurava dodatna sigurnost.

[Tim182](#)

Pages 13
Find a page...
Home
1. Opis projektnog zadatka
2. Analiza zahljeva
3. Specifikacija programske potpore
4. Arhitektura i dizajn sustava
5. Arhitektura komponenata i razmješ...
6. Ispitivanje programskog rješenja
7. Tehnologije za implementaciju apl...
Korištenje tehnologije i alati
8. Upute za puštanje u pogon
9. Zaključak i budući rad
A. Popis literature
B. Dnevnik promjena dokumentacije
C. Prikaz aktivnosti grupe

[+ Add a custom sidebar](#)[Clone this wiki locally](#)<https://github.com/franogrinsak/Tim182>

franogrinsak / Tim182

Type to search

Code Issues Pull requests Actions Projects 1 Wiki Security Insights

8. Upute za puštanje u pogon

Filip Šturić edited this page 6 minutes ago · 4 revisions

[Edit](#) [New page](#)

1. Instalacija

Predvjeti: Node.js, IntelliJ IDEA, VS Code (preporučujemo najnovije verzije)

- Preuzimanje:

Potrebito je klonirati repozitorij naredbom:

```
git clone https://github.com/franogrinsak/Tim182.git
```

Ulazak u repozitorij sa:

```
cd Develop/frontend
```

Instalacija ovisnosti:

```
npm install
```

2. Postavke

Detaljne upute za konfiguraciju aplikacije:

Prilikom pokretanja programa u Intelliju (datoteka PrimjerApplicationTests.java -> Edit Config -> Spring Boot -> Environment variables) potrebito je dodati varijable:

```
VITE_BACKEND_URL=https://localhost:8080/api
```

Backend ne zna lokaciju frontenda pa koristimo varijable za prikazivanje URL-a:

```
FRONTEND_DOMAIN=http://localhost:5137/
frontendUrl=http://localhost:5137/app
```

- Za pokretanje usluge plaćanja (stripe) lokalno koristimo naredbu

```
stripe listen --forward-to http://localhost:8080/api/webhook
```

- Postavke baze podataka:

Unutar direktorija db se nalazi datoteka insert.sql sa testnim podacima te create.sql za inicijalizaciju baze

3. Pokretanje aplikacije

Upute za pokretanje aplikacije u različitim okruženjima:

- Razvojno okruženje:

```
npm run dev
```

- Provjera rada:

```
URL = http://localhost:5137/
```

4. Upute za administratore

Smjernice za administratora aplikacije nakon puštanja u pogon:

- Pristup administratorskom sučelju: Dodajemo korisnika sa administratorskim ovlastima u bazu podataka uz pomoć naredbe:

```
INSERT INTO users (userId, email, firstName, lastName, roleid) VALUES [dodajte željene vrijednosti] --admin
```

- Redovito održavanje:

- Ažuriranje aplikacije

Ažuriramo datoteke

```
git pull origin main
```

Ponovno instaliramo i pokrenemo aplikaciju

```
npm install
npm run build
```

5. Primjer za Render platformu (Cloud Deploy)

Render je popularna cloud platforma za jednostavno smještanje aplikacija.

- Priprema repozitorija:

- U repozitoriju unutar direktorija backend se nalazi Dockerfile za konfiguraciju deploya.

- Postavljanje na Render:

- Prijavljujemo se na [Render](#).
- Kreiramo novi Web Service i povezujemo ga s našim repozitorijem.
- Dodajemo environment varijable:

VITE_BACKEND_URL za generiranje backend URL-a

datasource_password i datasource_url te datasource_username za povezivanje baze

FRONTEND_DOMAIN i frontendUrl za prikaz URL-a frontenda backendu

google_client_id i google_client-secret za korištenje vanjskog servisa za autentifikaciju

STRIPE_SECRET_KEY i STRIPE_WEBHOOK_KEY za omogućavanje plaćanja

- **Pokretanje aplikacije:**

Render će automatski preuzeti repozitorij, instalirati ovisnosti i pokrenuti aplikaciju. Nakon deploja, aplikaciji možemo pristupiti putem generiranog URL-a (<https://playpaddle.onrender.com/>).

Opis pristupa aplikaciji na javnom poslužitelju

Pristup aplikaciji

- Ograničenja: koristiti Google Chrome preglednik, ako ostali preglednici imaju greške pri radu aplikacije
- Korisnici pristupaju aplikaciji unosom adrese playpadel.tech u web preglednik

Tim182



© 2025 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact](#) [Manage cookies](#) [Do not share my personal information](#)

9. Zaključak i budući rad

Filip Šturić edited this page 2 days ago · 3 revisions

Projektni zadatak izrade web aplikacije za rezervaciju i promociju padel sporta uspješno je realiziran. Osim što smo razvili funkcionalnu platformu, koja omoguće jednostavnu rezervaciju termina i promociju padela, stekli smo vrijedna iskustva iz timskog rada, upravljanja projektom te korištenja modernih tehnologija i alata.

Projekt je bio organiziran u dvije glavne faze. U prvoj fazi fokusirali smo se na analizu zahtjeva, definiranje funkcionalnih i nefunkcionalnih specifikacija te izradu početne dokumentacije. Dokumentacija je obuhvaćala sve ključne aspekte projekta, poput UML dijagrama, tehničke arhitekture te inicijalne funkcionalnosti sustava, uključujući registraciju i prijavu korisnika te kreiranje terena. Ova faza nam je omogućila temeljitu pripremu za tehničku implementaciju.

U drugoj fazi prešli smo na razvoj i implementaciju funkcionalnosti platforme. Kroz koordinirani timski rad, implementirane su ostale funkcionalnosti: pregled i rezervacija termina, organizacija, prijava te odobravanje/odbijanje prijava na turnire, upravljanje plaćanjima putem vanjskih servisa (PayPal i kreditne kartice) te pregled rezultata i sadržaja turnira. Dodatno, vlasnicima dvorana omogućen je unos rezultata o turnirima, dok administratori mogu upravljati korisnicima i postavljati cijenu članstva. Također smo koristili alate za ispitivanje implementiranih rješenja te sve te postupke dokumentirali prema predlošcima.

Tijekom projekta suočili smo se s izazovima, poput uskladivanja rokova i implementacije složenih funkcionalnosti, no kroz redovite sastanke i prilagodbe uspjeli smo ih uspješno savladati. Naučili smo važnost dobre komunikacije i međusobne podrške unutar tima, što je rezultiralo uspješnim završetkom projekta.

Zaključno, sudjelovanje u ovom projektu donijelo nam je nova tehnička znanja, poput korištenja tehnologija za frontend i backend razvoj, integracije vanjskih servisa te izrade kvalitetne dokumentacije. Ova iskustva značajno su nas pripremila za buduće projekte i karijeru u razvoju softverskih rješenja. Uvjereni smo da bismo, uz stečeno iskustvo, sada projekt mogli realizirati još učinkovitije i kvalitetnije.

Tim182

Pages 13
Find a page...
Home
1. Opis projektnog zadatka
2. Analiza zahtjeva
3. Specifikacija programske potpore
4. Arhitektura i dizajn sustava
5. Arhitektura komponenata i razmjješ...
6. Ispitivanje programskog rješenja
7. Tehnologije za implementaciju apl...
8. Upute za puštanje u pogon
9. Zaključak i budući rad
A. Popis literature
B. Dnevnik promjena dokumentacije
C. Prikaz aktivnosti grupe

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/franogrinsak/Tim1>



A. Popis literature

Filip Šturić edited this page 3 weeks ago · [1 revision](#)

[Edit](#)[New page](#)

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. Visual Paradigm Online, <https://online.visual-paradigm.com/>
8. SocraticsSol. (2020, December 14). Why MVC architecture. Medium. <https://medium.com/@socraticsol/why-mvc-architecture-e833e28e0c76>
9. Izvor slike 3.1, <https://www.geeksforgeeks.org/mvc-architecture-system-design/>
10. Primjeri sličnih aplikacija:
 - 10.1. Playtomic (<https://playtomic.io/padelhr/da7877d5-43b3-11e8-8674-52540049669c?q=PADEL~2024-11-15~~~>)
 - 10.2. reservepadel (<https://www.reservepadel.com/>)
 - 10.3. BookMyCourt (<http://www.bookmycourt.com/default.aspx>)
 - 10.4. Tournament Software (<https://www.tournamentsoftware.com/>)

Tim182

Pages 13
<input type="text"/> Find a page...
▶ Home
▶ 1. Opis projektnog zadatka
▶ 2. Analiza zahljeva
▶ 3. Specifikacija programske potpore
▶ 4. Arhitektura i dizajn sustava
▶ 5. Arhitektura komponenata i razmjješ...
▶ 6. Ispitivanje programskog rješenja
▶ 7. Tehnologije za implementaciju apli...
▶ 8. Upute za puštanje u pogon
▶ 9. Zaključak i budući rad
▶ A. Popis literature
▶ B. Dnevnik promjena dokumentacije
▶ C. Prikaz aktivnosti grupe

[+ Add a custom sidebar](#)[Clone this wiki locally](#)<https://github.com/franogrinsak/Tim182> 

B. Dnevnik promjena dokumentacije

Fran Ogrinšak edited this page 1 hour ago · 5 revisions

[Edit](#)
[New page](#)

Rev.	Opis promjene/dodataka	Autori	Datum
0.1	Napravljen predložak	Fran Ogrinšak	19.10.2024.
0.2.1	Dodani funkcionalni zahtjevi	Fran Ogrinšak	24.10.2024.
0.2.2	Dodani nefunkcionalni zahtjevi	Filip Šturić	24.10.2024.
0.3.1	Dodani obrasci i dijagrami obrazaca uporabe	Filip Šturić	31.10.2024.
0.3.2	Dodan opis arhitekture	Filip Šturić	31.10.2024.
0.4	Dodan opis projektnog zadatka	Filip Šturić	12.11.2024.
0.5	Dodana zasebna stranica za zahtjeve	Filip Šturić	13.11.2024.
0.6.1	Ažuriran opis projektnog zadatka	Filip Šturić	13.11.2024.
0.6.2	Ažurirani funk. i nefunkc. zahtjevi	Filip Šturić	13.11.2024.
0.6.3	Dodani sekvencijski dijagrami	Filip Šturić	13.11.2024.
0.6.4	Dodan opis i dijagrami baze podataka te opis tablica	Fran Ogrinšak	13.11.2024.
0.7.1	Ažuriran Opis projektnog zadatka	Filip Šturić	15.11.2024.
0.7.2	Ažuriran Opis arhitekture	Filip Šturić	15.11.2024.
0.7.3	Ažurirani Sekvencijski dijagrami	Filip Šturić	15.11.2024.
0.7.4	Ažurirani NF zahtjevi	Filip Šturić	13.11.2024.
0.7.5	Dodan dijagram razreda	Filip Šturić	15.11.2024.
0.7.6	Ažurirani dijagrami razreda	Filip Šturić	15.11.2024.
0.7.7	Ažurirani dijagrami obrazaca uporabe i obrasci uporabe	Fran Ogrinšak	15.11.2024.
1.0	Ažuriran prikaz aktivnosti grupe za kraj 1. revizije	Fran Ogrinšak	15.11.2024.
1.1	Uskladivanje predloška dokumentacije za 2. reviziju	Filip Šturić	2.1.2025.
1.2	Dodan implementacijski dijagram razreda	Filip Šturić	15.1.2025.
1.3.1	Dodan dijagram stanja	Filip Šturić	15.1.2025.
1.3.2	Dodan dijagram komponenata	Filip Šturić	15.1.2025.
1.4.1	Dodan dijagram razmještaja	Filip Šturić	15.1.2025.
1.4.2	Dodan dijagram aktivnosti	Filip Šturić	16.1.2025.
1.5	Ažurirani dijagrami stanja i aktivnosti	Filip Šturić	22.1.2025.
1.6	Dodane korištene tehnologije i alati	Filip Šturić	22.1.2025.
1.7	Dodan zaključak	Filip Šturić	22.1.2025.
1.8	Dodane upute za puštanje u pogon	Filip Šturić	24.1.2025.
1.9	Dodano ispitivanje komponenti	Patrik Pašić	24.1.2025.
1.10	Dodano ispitivanje sustava	Fran Ogrinšak	24.1.2025.
2.0	Ažuriranje prikaza aktivnosti grupe za kraj 2. revizije	Fran Ogrinšak	24.1.2025.

Pages 13
Find a page...
Home
1. Opis projektnog zadatka
2. Analiza zahtjeva
3. Specifikacija programske potpore
4. Arhitektura i dizajn sustava
5. Arhitektura komponenata i razmješ...
6. Ispitivanje programskog rješenja
7. Tehnologije za implementaciju apli...
8. Upute za puštanje u pogon
9. Zaključak i budući rad
A. Popis literature
B. Dnevnik promjena dokumentacije
C. Prikaz aktivnosti grupe

[+ Add a custom sidebar](#)
[Clone this wiki locally](#)
<https://github.com/franogrinsak/Tim1>

C. Prikaz aktivnosti grupe

Filip Šturić edited this page 35 minutes ago · 9 revisions

[Edit](#) [New page](#)

#Dnevnik sastajanja

1. sastanak

- Datum: 17. listopada 2024.
- Prisustvovali: L. Matić, F. Ogrinšak, P. Pašić, Š. Filip, Z. Luka
- Teme sastanka:
 - Upoznavanje članova tima
 - Raspodjela u članova u timove prema vrsti posla (frontend, backend, dokumentacija)
 - Odabir tehnologija i alata koji će se koristiti na projektu

2. sastanak

- Datum: u ovom formatu: 21. listopada 2024.
- Prisustvovali: L. Matić, F. Ogrinšak, P. Pašić, Š. Filip, Z. Luka
- Teme sastanka:
 - Zajedničko definiranje funkcionalnih zahtjeva
 - Podjela rada za izradu obrazaca uporabe

3. sastanak

- Datum: u ovom formatu: 24. listopada 2024.
- Prisustvovali: L. Matić, F. Ogrinšak, P. Pašić, Š. Filip, Z. Luka
- Teme sastanka:
 - Prezentacija funkcionalnih zahtjeva
 - Provjera izrađenih obrazaca uporabe

4. sastanak

- Datum: u ovom formatu: 28. listopada 2024.
- Prisustvovali: L. Matić, F. Ogrinšak, P. Pašić, Š. Filip
- Teme sastanka:
 - Sastavljanje napravljenih obrazaca uporabe i rad na bazi podataka
 - Raspodjela poslova za implementaciju registracije i prijave

5. sastanak

- Datum: 7. studenog 2024.
- Prisustvovali: F. Ogrinšak, P. Pašić, Z. Luka
- Teme sastanka:
 - Predstavljanje logina i registracije
 - Raspodjela posla za stvaranje terena i administratorske funkcionalnosti

6. sastanak

- Datum: 14. studenog 2024.
- Prisustvovali: L. Matić, F. Ogrinšak, P. Pašić, Š. Filip
- Teme sastanka:
 - Predstavljanje implementacije pregleda terena i administratorskih funkcionalnosti
 - Provjera dokumentacije te raspodjela zadataka za ispravak pogrešaka u dokumentaciji

7. sastanak

- Datum: 16. siječnja 2025.
- Prisustvovali: L. Matić, F. Ogrinšak, P. Pašić, Š. Filip
- Teme sastanka:
 - Prezentacija alfa inačice
 - Podjela zadataka za testiranje i završetak dokumentacije

Tablica aktivnosti

- Dokumentacija:
- Upravljanje projektom - F. Ogrinšak (10h)
- Opis projektnog zadatka - F. Šturić (2h)
- Funkcionalni zahtjevi - F. Ogrinšak (2h), F. Šturić (2h), L. Matić (2h), L. Matić (2h)
- Opis pojedinih obrazaca - F. Ogrinšak(2h), F. Šturić (4h), L. Matić (2h)
- Dijagram obrazaca - F. Ogrinšak (2h), F. Šturić (2h)
- Sekvencijski dijagrami - F. Šturić (4h)
- Opis ostalih zahtjeva - F. Šturić (1h)
- Arhitektura i dizajn sustava - F. Šturić (2h)
- Baza podataka - F. Ogrinšak (1h)
- Dijagram razreda - F. Šturić (2h)

Pages 13
<input type="text" value="Find a page..."/>
▶ Home
▶ 1. Opis projektnog zadatka
▶ 2. Analiza zahtjeva
▶ 3. Specifikacija programske potpore
▶ 4. Arhitektura i dizajn sustava
▶ 5. Arhitektura komponenata i razmješ...
▶ 6. Ispitivanje programskog rješenja
▶ 7. Tehnologije za implementaciju apli...
▶ 8. Upute za puštanje u pogon
▶ 9. Zaključak i budući rad
▶ A. Popis literature
▶ B. Dnevnik promjena dokumentacije
▶ C. Prikaz aktivnosti grupe
Tablica aktivnosti
Dijagram pregleda promjena
Ključni izazovi i rješenja
Sinkronizacija članova tima: više članova timova radi na istoj funkcionalnosti
Izlučivanje zahtjeva iz zadatka: manjak preciznog opisa zadatka

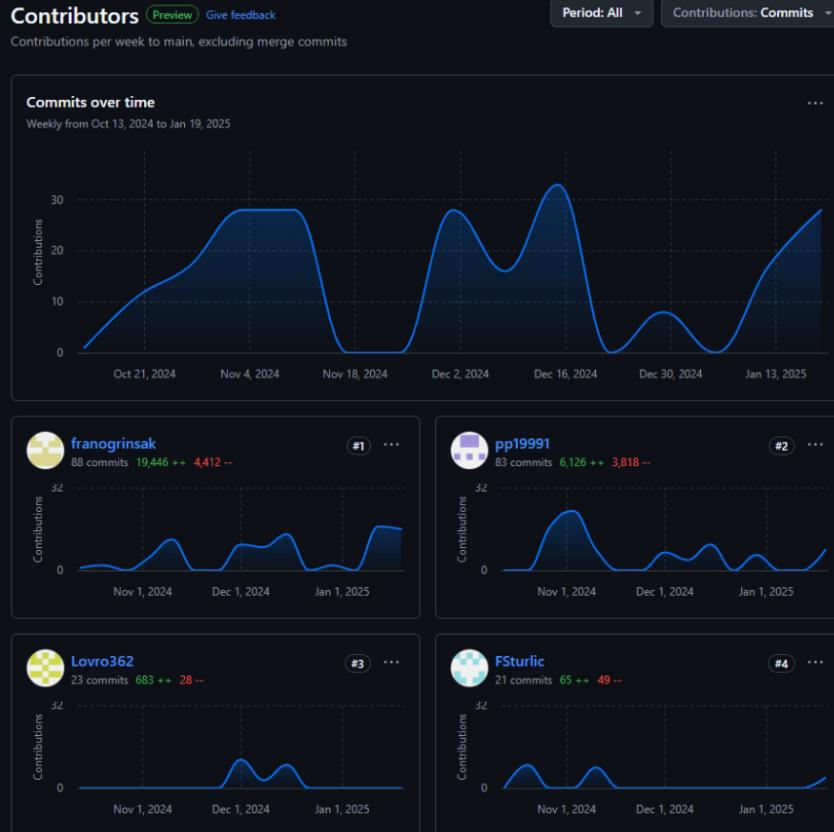
+ Add a custom sidebar

Clone this wiki locally

<https://github.com/franogrinsak/Tim1> 

- Dijagram stanja - F. Šturić (1h)
- Dijagram aktivnosti - F. Šturić (2h)
- Dijagram komponenti - F. Šturić (2h)
- Korištenje tehnologije i alati - F. Šturić (1h)
- Ispitivanje programskog rješenja - F. Ogrinšak (5h), P. Pašić (5h)
- Dijagram raznještaja - F. Šturić (0.5h)
- Upute za puštanje u pogon - F. Šturić (2h)
- Dnevnik sastajanja - F. Ogrinšak (0.5h)
- Zaključak i budući rad - F. Šturić (1h)
- Popis literature - F. Šturić (1h)
- Izrada prezentacije - F. Šturić (1h)
- Izrada aplikacije:
- izrada baze podataka - F. Ogrinšak (2h)
- Pojedine backend i frontend aktivnosti:
- spajanje s bazom podataka - L. Matić (2h), P. Pašić (2h)
- Izrada prijave i registracije - P. Pašić (15h), F. Ogrinšak (10h)
- Izrada terena - P. Pašić (5h), F. Ogrinšak (5h)
- Izrada administratorskih funkcionalnosti - L. Matić (8h), F. Ogrinšak (3h)
- Izrada turnira - L. Matić (20h), F. Ogrinšak (20h)
- Izrada obavijesti - L. Matić (10h), F. Ogrinšak (5h)
- Izrada termina - P. Pašić (20h), F. Ogrinšak (15h)
- Izrada plaćanja - P. Pašić (10h), F. Ogrinšak (5h)

Dijagram pregleda promjena



Slika 12.1: Dijagram pregleda promjena

Ključni izazovi i rješenja

Sinkronizacija članova tima: više članova timova radi na istoj funkcionalnosti

- Rješenje: koristi git grane za vrijeme razvoja

Izlučivanje zahtjeva iz zadatka: manjak preciznog opisa zadatka

- Rješenje: redovna komunikacija s asistentima

Tim182



