

Programsko inženjerstvo ak.god 2025./2026.

Sveučilište u Zagrebu

Fakultet elektrotehnike i računarstva

PlayTrade

Tim: TG04.3

Ime tima: error808

Nastavnik: Vlado Sruk

Članovi tima:

- Marko Bošnjak - Backend
- Marin Čikotić - Fullstack
- Frane Ćevid - Baze podataka
- Toni Kapućija - Testiranje
- Niko Knežević - Voditelj
- Mihael Rošić - Frontend
- Ivan Žalac - DevOps

Poveznica na GitHub projekta:

<https://github.com/tim-error808/error808>

PlayTrade je web aplikacija osmišljena za sve ljubitelje društvenih igara kako bi mogli iskusiti veći opseg različitih igara.

Mnogi ljudi koji uživaju u društvenim igrama često kupe igru koja im se svidi, ali nakon vremena mnoge postanu monotone. Naime, vrlo često se dogodi kod društvenih igara da njihovi igrači nauče komponente igre kroz iskustvo i nakon nekog vremena igra postaje predvidljiva, stoji na polici i skuplja prašinu. Tu bi od velike koristi moglo biti zamjeniti igru koju vlasnik više neće koristiti za neku novu, jer ne samo da se neće povećavati broj igara u kolekciji, nego se i neće morati trošiti novce na nove. Stoga PlayTrade omogućava korisnicima upravo to.

Aplikacija omogućuje svim korisnicima da pregledavaju objave igara koje su dostupne za zamjenu i traže neku koju imaju na umu, a onim registriranim omogućuje da objave svoje igre koje žele zamijeniti i da vrše zamjene s drugima. Na taj način svaki korisnik doprinosi ukupnoj bazi igara i povećava mrežu, a time i mogućnost da svatko pronađe neku novu, uzbudljivu igru.

Potencijalna korist ovog projekta

Aplikacija PlayTrade ima više potencijalnih koristi za igrače diljem svijeta, ali i okoliš:

Umanjuje troškove na društvene igre: mnogi ljudi bi voljeli probati različite društvene igre, ali često odustanu od toga iz finansijskih razloga. Međutim, na PlayTrade-u korisnici mogu koristiti svoju malu bazu igara i mijenjati ih za koju god žele bez dodatnih troškova.

Pomaže proširenju iskustva: entuzijasti vole iskusiti različite društvene igre - različitim zahtjevnostima, profila i s različitim brojem igrača, ali iz različitih razloga to nije uvijek lako. Primjerice, moguće je da se određena igra više ne proizvodi ili prodaje, a isto tako je moguće da više ne postoji mjesto za novu igru. Uz pomoć PlayTrade-a korisnici mogu relativno lako pronaći željenu igru i zamijeniti se za nju, te time oslobođiti mjesto na polici i iskusiti nešto novo.

Pomaže povezivanju ljudi: Putem aplikacije korisnici se mogu susresti s ljudima koji imaju slične interese kao i oni i na taj način povećati mrežu svojih prijateljstava, a možda i pronaći novog suigrača.

Pomaže u očuvanju okoliša i štednji energije i materijala: Zamjenom jedne društvene igre za drugu između dva korisnika umanjuje se potreba da ljudi kupe igru koju žele. Samim time stare igre se recikliraju umjesto da se kupuju nove i na taj način se smanjuje otpad koji nastaje jednom kad završi životni vijek igre, ali se i uštedi na energiji i materijalima koji bi bili potrebni u proizvodnim procesima.

Postojeća slična rješenja

BoardGamesSwap

Slika 1.1 Korisničko sučelje stranice BoardGamesSwap



Search products

Getting started

Learn how the platform works in a few minutes

Sell now

With integrated shipping & payment

Boardgames

Boardgame Accessories

Trading Card Games

Role Playing Games

TT miniature wargames

Puzzles

BGS Merchandise

Rješenje slično našem je stranica naziva **BoardGamesSwap**.

Na njoj korisnici mogu objaviti svoju društvenu igru s cijenom i kupiti nečiju drugu društvenu igru. Uz pomoć BoardGamesSwap neregistrirani korisnici mogu kupiti nečiju igru, a registrirani mogu postaviti i svoju igru na prodaju.

Korisnici mogu tražiti neku igru direktno ili listati sve objave iz neke kategorije direktno (Boardgames, Boardgame Accessories, Trading Card Games, TT miniature wargames, Puzzles, BGS Merchandise). Odnosno moguće je kupiti i druge stvari vezane uz igre, a ne igre isključivo.

Korisnici mogu filtrirati ponude po različitim kategorijama: jezik, stanje, godina izdanja, starost, minimalna dob, broj igrača, trajanje igre...

Naš sustav dijeli neke sličnosti i razlike. Za početak, naš sustav nije platforma za prodaju igara već se samo vrše direktnе zamjene. Također PlayTrade se fokusira na društvene igre, odnosno ne uključuje stvari kao što su dodatci za igre.

Na PlayTrade-u se igre mogu filtrirati po sličnim kategorijama: očuvanost, broj igrača, zahtjevnost, izdavač, godina izdanja...

Facebook i Reddit grupe

Slika 1.2 Primjer jedne Facebook grupe namijenjene razmjeni društvenih igara

Board Game Exchange - The Original “Buy, Sell & Trade” Boardgame Group

🔒 Private group · 57.1K members

 Join group



About Discussion



About this group

Welcome to the original board game exchange; "buy, sell, and trade" group! We are the most active community on Facebook dedicated to the buying,... [See more](#)



Private

Only members can see who's in the group and what they post.



Visible

Anyone can find this group.



History

Group created on September 7, 2015. Name last changed on April 30, 2020.

[See more](#)



Tags

Board Games

Na Facebooku i Redditu su osnovane različite grupe za razmjenu društvenih igara i na njima je osim zamjene moguće vršiti i prodaju. Za razliku od toga, na PlayTrade-u se vrši isključivo zamjena. Na Facebooku i Redditu ne postoje tako organizirani filteri kojima ljudi mogu naći točno ono što žele, kao na PlayTrade-u.

BoardGameGeek

Slika 1.3 Guide to BoardGameGeek

THE HOTNESS

GAMES ▾


Miskatonic Tales: Journey to...
2025
Speakeasy
2026
The Old King's Crown
2025
Winnie the Pooh: Serious...
2026
Ikusa
1986
The Lord of the Rings: Fate...
2025
Covenant
2025
Ortoj: The Prague...
2025
The Druids of Edora
2025
SETI: Search for...
2024
Ayar: Children of the Sun
2025Search: Titles Only: Go[Index](#) | [All](#) | [Recent](#) | [Guidelines](#)[Article](#) [Edit](#) [History](#) [Editors](#)Guide To BoardGameGeek 

This page is for board games. For RPGs, see [RPGGeek](#). For video games, see [VideoGameGeek](#).

[BoardGameGeek](#) is a website dedicated to physical board games, with an extensive database of more than 120,000 board games (as of October 2020) as well as an active community of users who discuss, argue, buy, sell, trade and play board games. Each game has its own [game entry](#) with information about a game, user ratings, forums for discussion, and a great deal more.

The [forums](#) are a good place to explore to involve yourself in the community. If you are completely new to the site, check out the [Getting Started](#) page and the [Welcome to BoardGameGeek](#) page. To start exploring the wiki, have a look at the [Wiki Index](#), and if you want to learn more about what the wiki is and how to add content to it, go to [About the BoardGameGeek Wiki](#).

Major Content Areas of BGG

- [Front Page](#)
- [Dashboard](#) - A customizable compilation of what's new on BGG
- [Games](#) - Description of the sections on a game page
- [My Geek](#) - Sections found on your personal page
- [Forums](#) - Info and FAQ about the forums, plus a list of Forums
- [Blogs](#) - Info about creating and contributing to a Blog
- [GeekLists](#) - What GeekLists are and options for their use
- [Shopping](#) - Place to trade and buy games
- [Misc](#) - Guilds, Stats, Cameras, etc. - Everything under the Misc menu above
- [Wiki Index](#) - Top level index of this wiki

Table of Contents

- [Major Content Areas of BGG](#)
- [Contribute to the BGG Database](#)
- [Buying, Selling and Trading Games](#)
- [More Ways to Interact with BGG](#)
- [Become a financial supporter](#)
- [Text Editors at BGG](#)

Contribute to the BGG Database

- [Content Rules](#)
- [Submit game listings](#)

[BoardGameGeek](#) je vrlo popularna stranica među ljubiteljima društvenih igara. Ondje je za izuzetno velik broj igara moguće pronaći informaciju o tome na kojoj se platformi i po kolikoj cijeni mogu kupiti društvene igre, moguće je vršiti zamjene igara, moguće je prodavati igre, moguće se upoznavati sa zajednicom, sudjelovati u forumima i mnogo drugih stvari.

S druge strane, PlayTrade nije toliko širok, ali je specijaliziran za zamjenu igara s drugim korisnicima kako bi to iskustvo bilo što efikasnije i ugodnije.

Skup korisnika zainteresiran za ostvareno rješenje

Skup korisnika koji bi mogao biti zainteresiran za aplikaciju PlayTrade uključuje svakog tko ima ikakav interes za društvene igre.

Aplikacija bi mogla privući **entuzijaste** s ogromnim iskustvom i željom za proširenje vidika.

Jednako tako, mogla bi privući i **hobiste**, one koji vole svoje druženje ispuniti društvenim igrama i koji traže nešto novo.

Ali PlayTrade bio bi pogodan i za **početnike**, one koji još nisu sigurni želete li dublje ući u svijet društvenih igara pa istražuju mogućnosti bez troškova.

Mogućnost prilagodbe rješenja

Aplikacija PlayTrade osmišljena je tako da bude **fleksibilna** i **prilagodljiva** potrebama različitih korisnika i potencijalnih tržišta, ali i da bude responzivna, odnosno da se korisničko sučelje prilagodi različitim vrstama i oblicima ekrana.

Korisnici će moći prilagoditi prikaz i način pretraživanja igara prema vlastitim preferencijama. Primjerice, moguće je definirati personalizirane obavijesti o novim objavama igara ili potencijalnim zamjenama prema korisnikovim prethodnim interesima.

Administratorski dio sustava također je dizajniran s mogućnošću prilagodbe. Sustav može biti konfiguriran za različite scenarije, uključujući dodavanje novih kategorija igara, upravljanje korisničkim računima te provjeravanje oglasa.

Opseg projektnog zadatka

Opseg projektnog zadatka obuhvaća sve estetske, funkcionalne, sigurnosne i tehničke aspekte:

- Izrada korisničkog sučelja:** Aplikacija će imati jednostavno korisničko sučelje na kojem će korisnici moći pristupiti registraciji, pristupiti bazi objava i kretati se kroz njih, a jednom kad se registriraju preko korisničkog sučelja moći će pristupiti zamjenama, odvijati interakciju s drugim korisnicima i pristupiti arhivi svojih zamjena i želja.
- Kreiranje filtera za objavljene igre:** Aplikacija će omogućavati korisnicima da pregledaju igre po određenim kategorijama, odnosno da pronađu igre ovisno o tome koliko su zahtjevne, da ih pronađu po tome za koliko igrača su namijenjene, da listaju cijelu kolekciju igara bez filtera ili da pronađu neku igru direktno.
- Kreiranje sustava za objave:** Korisnicima se mora omogućiti da mogu postaviti objavu igre koju žele zamijeniti i pritom popuniti potrebne podatke: naziv, žanr, izdavač, godinu izdanja, ocjenu očuvanosti igre, broj igrača, vrijeme igranja, procjenu težine igre, fotografiju igre te dodatan opis.
- Kreiranje sustava koji podržava interakciju s drugim korisnicima:** Aplikacija mora nuditi korisnicima mogućnost da zatraže zamjenu koja se nudi prije nego ju izvrše
- Kreiranje sustava za sigurnu registraciju:** Korisnici se na aplikaciji moraju registrirati, a ta registracija mora biti sigurna, kao i ostali podaci korisnika.
- Sustav za administratora:** sustavski administratori moraju moći upravljati oglasima i korisnicima.

Moguće nadogradnje projektnog zadatka

Aplikaciju PlayTrade u budućnosti će biti moguće nadograditi kako bi se obogatilo korisničko iskustvo i privukao širi spektar korisnika.

Aplikacija bi u budućnosti mogla omogućavati i zamjene drugih stvari vezane uz igre, primjerice određene figurice u igri.

Osim toga, aplikacija bi mogla nuditi korisnicima i stvaranje igračih grupa s drugima čija je lokacija u blizini, mogla bi pomoći u organiziranju i vođenju natjecanja u nekim igrama.

Također, aplikacija bi mogla nuditi tutoriale za određene igre kako bi pomogla igračima prisjetiti se koja su pravila igre, u slučaju da su svoja pravila izgubili. Ukratko, aplikacija ima širok prostor za napredak koji bi mogao pomoći povećanju njene korisničke baze i uspješnosti.

Cilj izrade projekta

Postoje različiti ciljevi ovog projekta:

- Stvaranje konkretnog rješenja koje bi se moglo plasirati u stvarni svijet
- Stjecanje znanja o razvoju softvera koji je prilagođen tako da pruža ugodno korisničko iskustvo, a istovremeno se brine o sigurnosti podataka i pravilnom radu sustava u pozadini
- Stjecanje iskustva rada u timu kako bi sudionici naučili koje se poteškoće i prilike iz toga mogu javiti i kako ih pretvoriti u snage### Funkcionalni zahtjevi:
- opisuju što sustav treba raditi ili koje funkcionalnosti mora pružiti korisnicima

Nefunkcionalni zahtjevi

- definiraju kako sustav treba raditi, uključujući performanse, pouzdanost, sigurnost i upotrebljivost.

Funkcionalni zahtjevi

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
F-01	Korisnik mora moći pristupiti sustavu.	Visok	Zahtjev dionika	Potencijalni korisnici moraju moći pristupiti stranici na efikasan, konstantan i jednostavan način.
F-02	Korisnik mora moći odabrati svoju lokaciju	Visok	Zahtjev dionika	Korisnik prilikom prijave ili uređivanja profila mora moći odabrati/označiti svoju lokaciju koja će biti vezana uz njegov profil.
F-03	Sustav mora reagirati na neispravnosti koje uzrokuje korisnik	Visok	Zahtjev dionika	U slučaju da korisnik uneše pogrešne podatke ili dođe do neke druge greške na stranici sustav mora pravilno reagirati prema korisniku.
F-04	Sustav mora moći provjeravati neispravnosti	Visok	Zahtjev dionika	Sustav mora moći provjeriti podatke kao što su: valjanost e-maila korisnika, ispravnost lozinke, popunjenošt određenih polja i ostalo.
F-05	Sustav omogućuje korisniku sučelje za objavu igre	Visok	Zahtjev dionika	Korisnik mora moći pristupiti sučelju za objavu svoje igre i jednom kad popuni što treba da se njegova igra postavi i bude vidljiva ostalim korisnicima.
F-06	Odabir lokacije za korisnika treba biti intuitivan i jednostavan	Srednji	Zahtjev dionika	Sučelje kojim korisnik odabire svoju lokaciju treba biti jednostavno i user-friendly.
	Sustav omogućuje korisnicima		Zahtjev	

F-07 ID zahtjeva	kreiranje računa pomoću e-mail opise.	Visok Prioritet	dionika Izvor	Korisnik se može registrirati e-mailom. Kriteriji prihvaćanja
F-08	Sustav provodi autorizaciju	Visok	Zahtjev dionika	Autorizacija se odvija putem OAuth 2.0 protokola koristeći tokene.
F-09	Sustav omogućuje korisnicima pregled objavljenih igara.	Visok	Zahtjev dionika	Čak i neregistrirani korisnik može pristupiti listi objavljenih igara i razgledati je
F-10	Sustav omogućuje korisnicima pretraživanje objavljenih igara.	Visok	Zahtjev dionika	Čak i neregistrirani korisnik može filtrirati igre koje gleda ili tražiti neku igru direktno.
F-11	Sustav omogućuje korisniku upravljanje profilom	Visok	Zahtjev dionika	Registrirani korisnik može objaviti svoju fotografiju i navesti kratak opis te birati kategorije igara po svom interesu.
F-12	Obavezno je da svaki profil ima pripadnu lokaciju.	Visok	Poslovno pravilo	Obavezno je da korisnicima jedna od stavki njihova profila bude njihova lokacija, za što se koristi vanjska usluga-OpenStreetMap.
F-13	Sustav omogućava korisniku da pristupi arhivi svojih zamjena	Srednji	Zahtjev dionika	Korisnik treba moći na stranici pristupiti svojim bivšim zamjenama i pregledati ih.
F-14	Sustav omogućuje korisniku unošenje detalja o igri.	Visok	Zahtjev dionika	Registrirani korisnik za svaku igru koju objavljuje navodi žanr, izdavača, godinu izdanja, ocjenu očuvanosti, broj igrača, vrijeme igranja, procjenu težine, fotografiju igre i dodatan opis ako je potreban.
F-15	Sustav omogućuje korisniku uređivanje svojih oglasa.	Visok	Zahtjev dionika	Registrirani korisnik može pristupiti svim svojim objavama i uređivati ih na pregledu "Moje igre".
F-16	Sustav omogućuje korisniku ponudu zamjene.	Visok	Zahtjev dionika	Registrirani korisnik može za željenu igru ili igre ponuditi zamjenu gumbom "Ponudi zamjenu".
F-17	Sustav mora javiti korisniku kad je zamjena dostupna i ponuditi mu njeno prihvatanje, odbijanje ili izmjenu.	Visok	Zahtjev dionika	Registrirani korisnik na e-mail mora dobiti obavijest o tome kad je zamjena dostupna i onda ju može prihvati, odbiti ili dodatno urediti.
F-18	Sistemski administratori održavaju platformu.	Visok	Administratori sustava	Sistemski administratori mogu upravljati korisnicima i oglasima (brisati ih i deaktivirati).

Ostali zahtjevi

Zahtjevi za performanse

ID zahtjeva	Opis	Prioritet
NF-1.1	Sustav treba raditi dobro bez obzira na broj korisnika.	Visok
NF-1.2	Sustav treba omogućiti brzu pretragu igara.	Srednji

Zahtjevi za iskustvo korisnika

ID zahtjeva	Opis	Prioritet
NF-2.1	Sustav mora biti razvijen kao web-aplikacija.	Visok
NF-2.2	Web-aplikacija mora biti responzivna.	Srednji

Zahtjevi za održavanje

ID zahtjeva	Opis	Prioritet
NF-3.1	Sustav treba biti intuitivno organiziran i pregledan.	Srednji
NF-3.2	Sustav treba imati dovoljnu dokumentaciju.	Visok

Zahtjevi za sigurnost i skalabilnost

ID zahtjeva	Opis	Prioritet
NF-4.1	Aplikacija treba osigurati sigurnost privatnih podataka korisnika.	Visok
NF-4.2	Sustav treba biti skalabilan i podržavati proširenje za nove objave.	Visok
NF-4.3	Sustav treba biti responzivan kako bi se mogao koristiti na različitim uređajima.	Visok
NF-4.4	Sustav treba biti pravilno i jednostavno integriran s vanjskim sustavima.	Srednji

Dionici

- Administrator (F-18)
- Registrirani korisnici (F-01, F-02, F-05, F-07, F-09, F-10, F-11, F-13, F-14, F-15, F-16, F-17)
- Neregistrirani korisnici (F-01, F-09, F-10)
- Prijavljeni korisnik (F-01, F-02, F-05, F-07, F-09, F-10, F-11, F-13, F-14, F-15, F-16, F-17)
- Vanjska usluga geolokacije (F-02, F-12)
- Baza podataka

Dionici i njihovi funkcionalni zahtjevi:

Administrator može:

- upravljati oglasima i korisnicima, tj. brisati oglase i blokirati korisnike (F-18)

Neregistrirani korisnik može:

- koristiti ograničene funkcionalnosti web-aplikacije, odnosno smije pretraživati i pregledavati objavljene igre (F-01, F-09, F-10)

Registrirani korisnik može:

- sve što i neregistrirani korisnik i dodatno:
 - primati obavijest o ponudi zamjene (F-16, F-17)
 - primati obavijest o dostupnosti igre s liste želja (F-17)

Prijavljeni korisnik može:

- sve što i registrirani korisnik i dodatno:
 - objavljivati igre (F-05, F-14)
 - nuditi zamjene (F-16)
 - uređivati svoj profil (F-11, F-02)
 - pristupati arhivi ponuda i zamjena (F-13)
 - uređivati svoje objave (F-15)

Baza podataka:

- služi za pohranu svih korisničkih podataka
- na zahtjev poslužitelja daje potrebne podatke

Vanjska usluga geolokacije:

- služi za lociranje osoba za vršenje zamjena (F-02, F-12)

Obrasci uporabe

Oznaka UC	Naziv obrasca uporabe	Povezani funkcionalni zahtjevi

Učinkova UC	Naziv obrasca uporabe	Povezani funkcionalni zahtjevi
UC-01	Registracija korisnika	F-07, F-02, F-04, F-12
UC-02	Prijava korisnika	F-01, F-08, F-04
UC-03	Uređivanje profila	F-11, F-02, F-12
UC-04	Pregled svih igara	F-09
UC-05	Objava nove igre	F-05, F-14
UC-06	Uređivanje i brisanje svojih objava	F-15
UC-07	Pretraživanje igara	F-10
UC-08	Ponuda zamjene	F-16, F-17
UC-09	Obavijest o novoj ponudi	F-17
UC-10	Prihvatanje ili izmjena ponude	F-17
UC-11	Pregled ponuda	F-16, F-17
UC-12	Stvaranje popisa želja	F-09, F-10
UC-13	Obavijest o dostupnosti igara s popisa želja	F-17, F-09
UC-14	Arhiviranje izvršenih zamjena	F-13, F-17
UC-15	Upravljanje korisnicima i oglasima	F-18
UC-16	Deaktivacija korisnika i oglasa zbog kršenja pravila	F-18

Opis obrazaca uporabe

UC-01: Registracija

Opis:

Korisnik unosi valjanu e-mail adresu, lozinku za web-aplikaciju, bira svoju lokaciju i registrira se.

Povezani funkcionalni zahtjevi: F-07, F-02, F-04, F-12

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Neregistrirani korisnik
- **Cilj:**
 - Stvaranje novog korisničkog računa
- **Sudionici:**
 - Neregistrirani korisnik, vanjska usluga geolokacije
- **Preduvjet:**
 - Korisnik nema postojeći račun
- **Osnovni tijek:**
 1. Korisnik otvara aplikaciju i odabire opciju 'Registriraj se' ili to čini putem Googlea (F-07 ili F-08)
 2. Unosi valjanu e-mail adresu (F-07)
 3. Odabire i unosi lozinku za aplikaciju (F-07)
 4. Koristeći OpenStreetMap na sustavu odabire svoju lokaciju (F-02, F-12)
 5. Provjerava se valjanost e-mail adrese i da korisnik s tom adresom već ne postoji (F-04)
 6. Ako je sve u redu, kreira se novi korisnički račun kojem su ostali podatci ispunjeni (F-07)
- **Moguća odstupanja:**
 - **Nevaljana e-mail adresa:** Sustav prikazuje obavijest o greški. (F-04)
 - **Korisnik već postoji:** Sustav obavještava da korisnik s tom e-mail adresom već postoji. (F-04)

UC-02: Prijava korisnika

Opis:

Korisnik unosi valjanu e-mail adresu i lozinku te se prijavljuje na stranicu ili se prijavljuje putem Googlea

Povezani funkcionalni zahtjevi: F-01, F-08, F-04

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Registrirani korisnik
- **Cilj:**
 - Prijava na stranicu i omogućavanje proširenih aktivnosti korisniku
- **Sudionici:**
 - Registrirani korisnik
- **Preduvjet:**
 - Korisnik se prethodno registrirao
- **Osnovni tijek:**
 1. Korisnik otvara aplikaciju i odabire opciju 'Prijava' ili se prijavljuje putem Googlea (F-01 ili F-08)
 2. Unosi valjanu e-mail adresu (F-04)
 3. Unosi svoju lozinku (F-04)
 4. Sustav provjerava valjanost podataka i vrši autorizaciju (F-08)
 5. Ako su podaci valjni, korisnik je prijavljen (F-01)
- **Moguća odstupanja:**
 - **Ne postoji račun s tom e-mail adresom:** Sustav prikazuje obavijest o greški. (F-04)
 - **Unesena neispravna lozinka:** Sustav obaveštava da je lozinka neispravna. (F-04)

UC-03: Uređivanje profila

Opis:

Korisnik može mijenjati podatke na svom profilu, konkretno: fotografiju profila, opis profila, kategorije igara od interesa, lokaciju.

Povezani funkcionalni zahtjevi: F-11, F-02, F-12

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Prijavljeni korisnik
- **Cilj:**
 - Korisnik može urediti svoj korisnički profil po svojim željama
- **Sudionici:**
 - Prijavljeni korisnik, vanjska usluga geolokacije
- **Preduvjet:**
 - Korisnik se prethodno registrirao i prijavio
- **Osnovni tijek:**
 1. Prijavljeni korisnik uđe u web-aplikaciju i klikne na ikonu svog profila (F-11)
 2. Odabire i mijenja elemente profila (F-11)
 3. Ažurira ili mijenja lokaciju pomoću kartografske usluge (F-02, F-12)
- **Moguća odstupanja:**
 - **Prijavljeni korisnik svojim promjenama prekrši pravila ponašanja:** Sustavski administratori uređuju ili brišu njegov profil. (F-18)

UC-04: Pregled svih igara

Opis:

Korisnik može pregledavati igre koje su objavili drugi korisnici.

Povezani funkcionalni zahtjevi: F-09

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Korisnik
- **Cilj:**
 - Korisnik može listati kroz igre koje su objavili drugi korisnici
- **Sudionici:**
 - Korisnik
- **Preduvjet:**
 - Korisnik je uspješno pristupio web-aplikaciji
- **Osnovni tijek:**
 1. Korisnik otvara aplikaciju (F-09)
 2. Ako želi, korisnik se može registrirati i/ili prijaviti (F-09)
 3. Odabire stranicu za pregled objava (F-09)
 4. Korisnik pregledava objavljene igre (F-09)
- **Moguća odstupanja:**
 - **Nema nijedna objava:** Sustav obaveštava korisnika o tome da još nema nijedna objavljena društvena igra. (F-09)

UC-05: Objava nove igre

Opis:

Korisnik može objaviti novu igru na način da odabere 'Objavi igru'.

Povezani funkcionalni zahtjevi: F-05, F-14

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Prijavljeni korisnik
- **Cilj:**
 - Korisnik može ponuditi neku svoju igru na zamjenu tako da je vide svi drugi korisnici
- **Sudionici:**
 - Prijavljeni korisnik
- **Preduvjet:**
 - Korisnik se prethodno registrirao i prijavio
- **Osnovni tijek:**
 1. Prijavljeni korisnik otvara aplikaciju (F-05)
 2. Odabire stranicu za objavu igara (F-05)
 3. Korisnik popunjava potrebne podatke o igri (F-14)
 4. Ako su podaci popunjeni korisnik stisne gumb 'Objavi' (F-05)
 5. Igra se objavljuje i postaje vidljiva drugim korisnicima (F-05)
- **Moguća odstupanja:**
 - **Podaci krše pravila ponašanja:** Administratori reagiraju na nepoštivanje pravila. (F-18)
 - **Nisu popunjeni svi podaci:** Korisnik ne može objaviti igru dok ne ispuni sva polja. (F-14)

UC-06: Uređivanje i brisanje svojih objava

Opis:

Korisnik može pristupiti svojim objavljenim igram na stranici za uređivanje i brisanje.

Povezani funkcionalni zahtjevi: F-15

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Prijavljeni korisnik
- **Cilj:**

- Korisnik može upravljati svojim već objavljenim igrama po želji
- **Sudionici:**
 - Prijavljeni korisnik
- **Preduvjet:**
 - Korisnik se prethodno registrirao i prijavio
- **Osnovni tijek:**
 1. Prijavljeni korisnik otvara aplikaciju (F-15)
 2. Odabire stranicu 'Moje igre' (F-15)
 3. Odabire gumb 'Uredi' ili 'Obriši' (F-15)
 4. Uređuje ili briše objavu (F-15)
- **Moguća odstupanja:**
 - **Nema objava:** Sustav obaveštava korisnika. (F-15)
 - **Podaci krše pravila ponašanja:** Administratori reagiraju. (F-18)

UC-07: Pretraživanje igara

Opis:

Korisnik može filtrirati igre koje su objavili drugi korisnici ili tražiti neku konkretnu igru.

Povezani funkcionalni zahtjevi: F-10

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Korisnik
- **Cilj:**
 - Korisnik može koristiti filtere za pretragu igara ili tražilicom tražiti specifičnu igru koju ima na umu
- **Sudionici:**
 - Korisnik
- **Preduvjet:**
 - Korisnik može pristupiti web-aplikaciji
- **Osnovni tijek:**
 1. Korisnik otvara aplikaciju (F-10)
 2. Odabire stranicu za pregled igara (F-10)
 3. Filtrira igre ili koristi tražilicu (F-10)
- **Moguća odstupanja:**
 - **Nema objava koje zadovoljavaju filter:** Sustav obaveštava korisnika. (F-10)

UC-08: Ponuda zamjene

Opis:

Korisnik može za određenu igru drugog korisnika ponuditi zamjenu.

Povezani funkcionalni zahtjevi: F-16, F-17

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Prijavljeni korisnik
- **Cilj:**
 - Korisnik može korisniku koji je vlasnik igre koju želi poslati zahtjev za ponudu s detaljima ponude
- **Sudionici:**
 - Prijavljeni korisnik
- **Preduvjet:**
 - Korisnik se prethodno registrirao i prijavio
- **Osnovni tijek:**

1. Korisnik otvara aplikaciju (F-16)
2. Pregledava objavljene igre (F-16)
3. Odabire gumb 'Ponudi zamjenu' (F-16)
4. Unosi igre koje nudi (F-16)
5. Klikne gumb 'Ponudi' (F-16)
6. Vlasniku igre stiže obavijest (F-17)

- **Moguća odstupanja:**

- **Nema dostupnih igara:** Sustav obaveštava korisnika. (F-16)

UC-09: Obavijest o novoj ponudi

Opis:

Korisniku za čiju je igru ponuđena zamjena dolazi obavijest u sustav i na e-mail.

Povezani funkcionalni zahtjevi: F-17

Detalji obrasca uporabe

- **Glavni sudionik:**

- Registrirani korisnik

- **Cilj:**

- Korisnik prima obavijest o tome da netko želi njegovu igru s objave

- **Sudionici:**

- Registrirani korisnik

- **Preduvjet:**

- Korisnik se prethodno registrirao i ima objavljenu igru koju netko može zatražiti

- **Osnovni tijek:**

1. Neki korisnik ponudi zamjenu (F-17)
2. Vlasniku igre stiže e-mail obavijest (F-17)
3. Prikazuje se i obavijest u aplikaciji (F-17)

- **Moguća odstupanja:**

- **Korisnik nije registriran:** Ne može praviti objave, pa time ni primati ponude. (F-17)

UC-10: Prihvatanje ili izmjena ponude

Opis:

Korisnik može ponudu prihvatiti, odbiti ili ponuditi kontra-ponudu.

Povezani funkcionalni zahtjevi: F-17

Detalji obrasca uporabe

- **Glavni sudionik:**

- Prijavljeni korisnik

- **Cilj:**

- Korisnik može uređivati uvjete zamjene, prihvatiti je ili odbiti

- **Sudionici:**

- Prijavljeni korisnik

- **Preduvjet:**

- Korisnik se prethodno registrirao i prijavio

- **Osnovni tijek:**

1. Prijavljeni korisnik otvara aplikaciju (F-17)
2. Pristupa ponudi koju je dobio (F-17)
3. Prihvata, odbija ili uređuje ponudu (F-17)
4. Drugoj strani stiže obavijest (F-17)

- **Moguća odstupanja:**

- **Korisnik nije prijavljen:** Ne može upravljati ponudama dok se ne prijavi na stranici. (F-17)

UC-11: Pregled ponuda

Opis:

Na stranici 'Ponude' registriranog korisnika može se pristupiti svim primljenim ponudama.

Povezani funkcionalni zahtjevi: F-16, F-17

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Prijavljeni korisnik
- **Cilj:**
 - Korisnik može pristupiti svim ponudama koje trenutno ima na jednom mjestu
- **Sudionici:**
 - Prijavljeni korisnik
- **Preduvjet:**
 - Korisnik se prethodno registrirao i prijavio
- **Osnovni tijek:**
 1. Prijavljeni korisnik otvara aplikaciju (F-16)
 2. Pristupa stranici 'Ponude' (F-16)
 3. Pregledava ponude koje je dobio (F-17)
- **Moguća odstupanja:**
 - **Nema ponuda:** Sustav obavještava korisnika. (F-16)
 - **Korisnik nije prijavljen:** Ne može pristupiti ponudama. (F-17)

UC-12: Stvaranje popisa želja

Opis:

Korisnik može stvoriti popis igara koje želi dobiti zamjenom.

Povezani funkcionalni zahtjevi: F-09, F-10

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Prijavljeni korisnik
- **Cilj:**
 - Korisnik može stvoriti listu igara koje želi dobiti zamjenom, a ne može ih naći među objavama
- **Sudionici:**
 - Prijavljeni korisnik
- **Preduvjet:**
 - Korisnik se prethodno registrirao i prijavio
- **Osnovni tijek:**
 1. Prijavljeni korisnik otvara aplikaciju (F-09)
 2. Pristupa profilu i opciji 'Moje želje' (F-10)
 3. Odabire igre koje želi (F-10)
 4. Sprema promjene (F-09)
- **Moguća odstupanja:**
 - **Nepostojeća igra:** Sustav dopušta izbor samo postojećih/sustavu poznatih igara. (F-09)
 - **Korisnik nije prijavljen:** Ne može pristupiti profilu. (F-09)

UC-13: Obavijest o dostupnosti igara s popisa želja

Opis:

Korisniku dolazi obavijest kad netko objavi igru koja je na njegovu popisu želja.

Povezani funkcionalni zahtjevi: F-17, F-09

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Registrirani korisnik
- **Cilj:**
 - Korisniku se javlja kad je njegova igra s popisa želja dostupna za zamjenu kako bi mogao pravovremeno reagirati
- **Sudionici:**
 - Registrirani korisnik
- **Preduvjet:**
 - Korisnik se prethodno registrirao
- **Osnovni tijek:**
 1. Neki korisnik objavi novu igru (F-09)
 2. Sustav šalje obavijest korisnicima koji je imaju na popisu (F-17)
 3. Obavijest se prikazuje i na aplikaciji (F-17)
- **Moguća odstupanja:**
 - **Korisnik nije prijavljen:** Ne može reagirati na obavijest. (F-17)
 - **Korisnik nije registriran:** Nema korisnički profil, pa ne može ni praviti listu želja. (F-17)

UC-14: Arhiviranje izvršenih zamjena

Opis:

Korisnik može vidjeti koje je zamjene već obavio.

Povezani funkcionalni zahtjevi: F-13, F-17

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Prijavljeni korisnik
- **Cilj:**
 - Korisniku može pristupiti obavljenim zamjenama kako bi ih mogao analizirati ako bude potrebno
- **Sudionici:**
 - Prijavljeni korisnik
- **Preduvjet:**
 - Korisnik se prethodno registrirao i prijavio
- **Osnovni tijek:**
 1. Prijavljeni korisnik otvara aplikaciju (F-17)
 2. Odabire stranicu 'Moje zamjene' (F-13)
 3. Pregledava arhivirane zamjene (F-13)
- **Moguća odstupanja:**
 - **Nema arhiviranih zamjena:** Sustav obavještava korisnika. (F-13)
 - **Korisnik nije prijavljen:** Ne može pristupiti arhivi. (F-17)

UC-15: Upravljanje korisnicima i oglasima

Opis:

Sistemski administratori mogu upravljati korisnicima i oglasima.

Povezani funkcionalni zahtjevi: F-18

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Administrator
- **Cilj:**
 - Administratori mogu upravljati korisnicima, njihovim profilima i objavama ako smatraju da je potrebno
- **Sudionici:**
 - Administratori

- Administrator
- **Preduvjet:**
 - Administrator je koristio svoje administratorske podatke za prijavu u sustav
- **Osnovni tijek:**
 1. Administrator se prijavljuje u sustav (F-18)
 2. Otvara administratorsko sučelje (F-18)
 3. Pregledava korisnike i oglase (F-18)
 4. Upravlja ili uređuje zapise (F-18)
- **Moguća odstupanja:**
 - **Nema podataka:** Sustav prikazuje poruku o neaktivnosti. (F-18)
 - **Administrator nije prijavljen:** Nema pristup funkcijama. (F-18)

UC-16: Deaktivacija korisnika i oglasa koji krše pravila

Opis:

Sistemski administratori mogu uklanjati korisnike i oglase.

Povezani funkcionalni zahtjevi: F-18

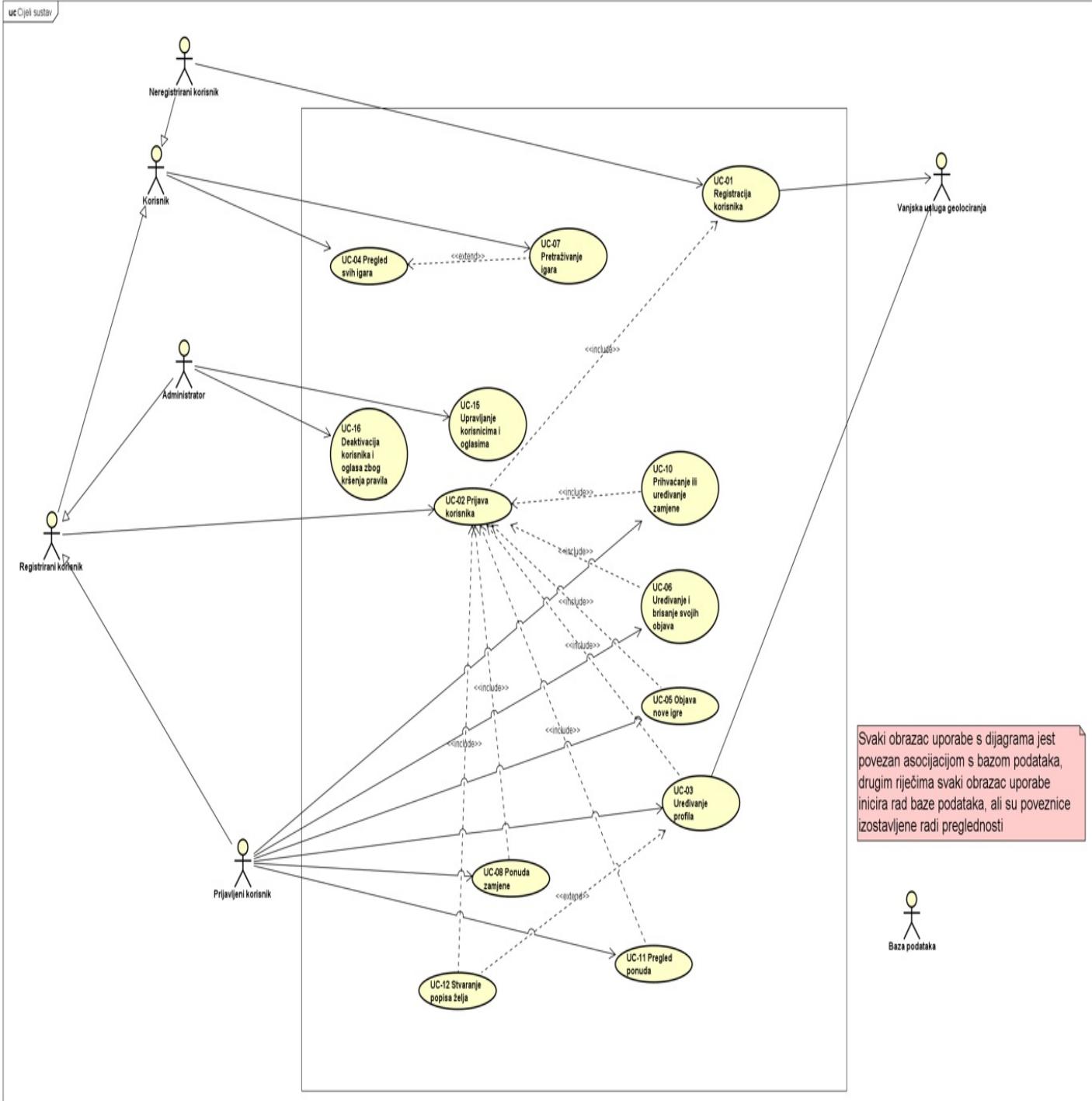
Detalji obrasca uporabe

- **Glavni sudionik:**
 - Administrator
- **Cilj:**
 - Administrator uklanja korisnike i/ili objave ako krše pravila ponašanja
- **Sudionici:**
 - Administrator
- **Preduvjet:**
 - Administrator je koristio svoje administratorske podatke za prijavu i upoznat je s pravilima ponašanja
- **Osnovni tijek:**
 1. Administrator se prijavljuje (F-18)
 2. Pristupa sučelju za upravljanje korisnicima (F-18)
 3. Odabire korisnika ili oglas za deaktivaciju (F-18)
 4. Potvrđuje deaktivaciju (F-18)
- **Moguća odstupanja:**
 - **Administrator nije prijavljen:** Nema pristup funkciji. (F-18)

Dijagrami obrazaca uporabe

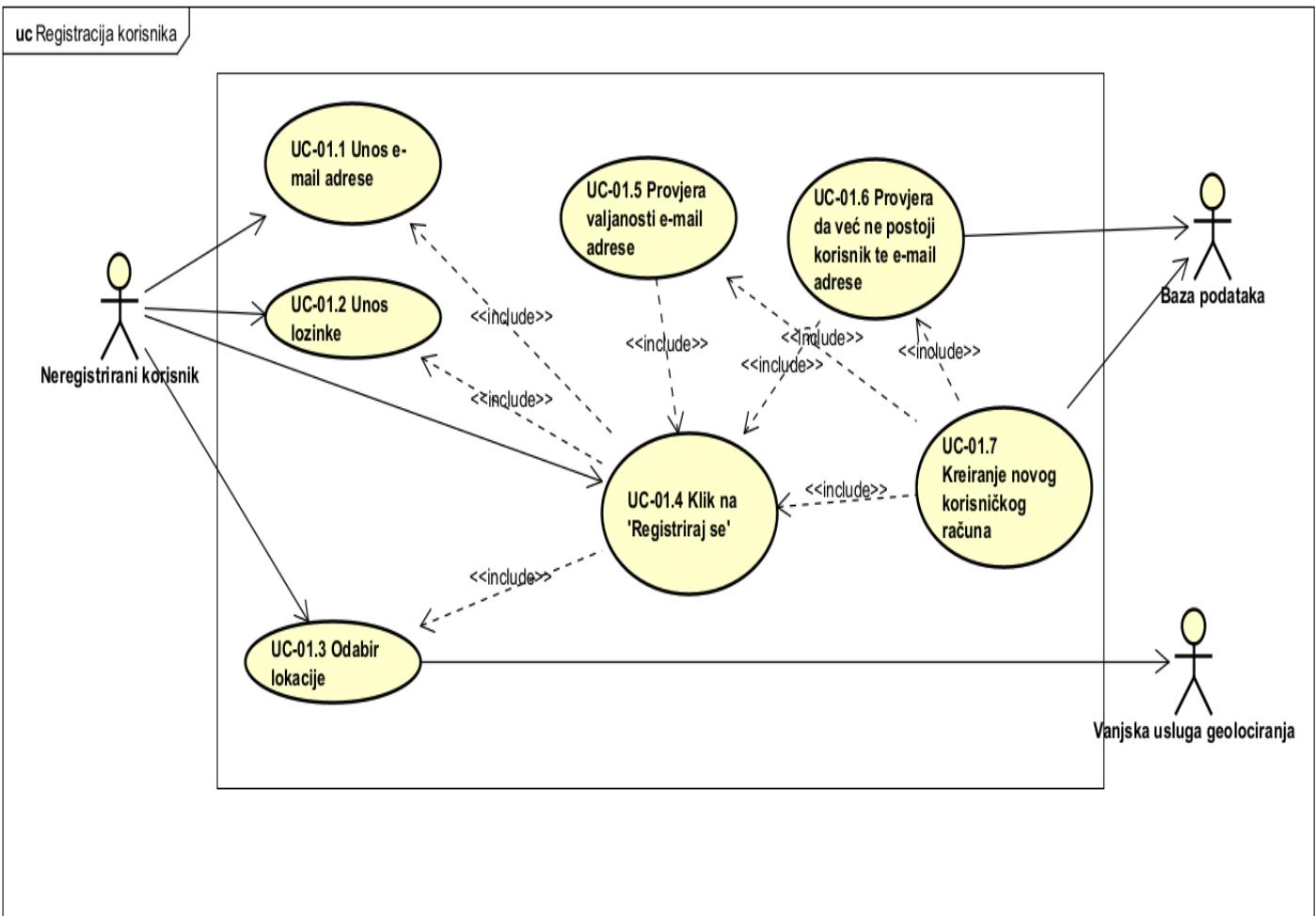
1. Visokorazinski dijagram obrazaca uporabe cijelog sustava

Slika 3.1 Dijagram osnovnih funkcionalnosti cijelog sustava

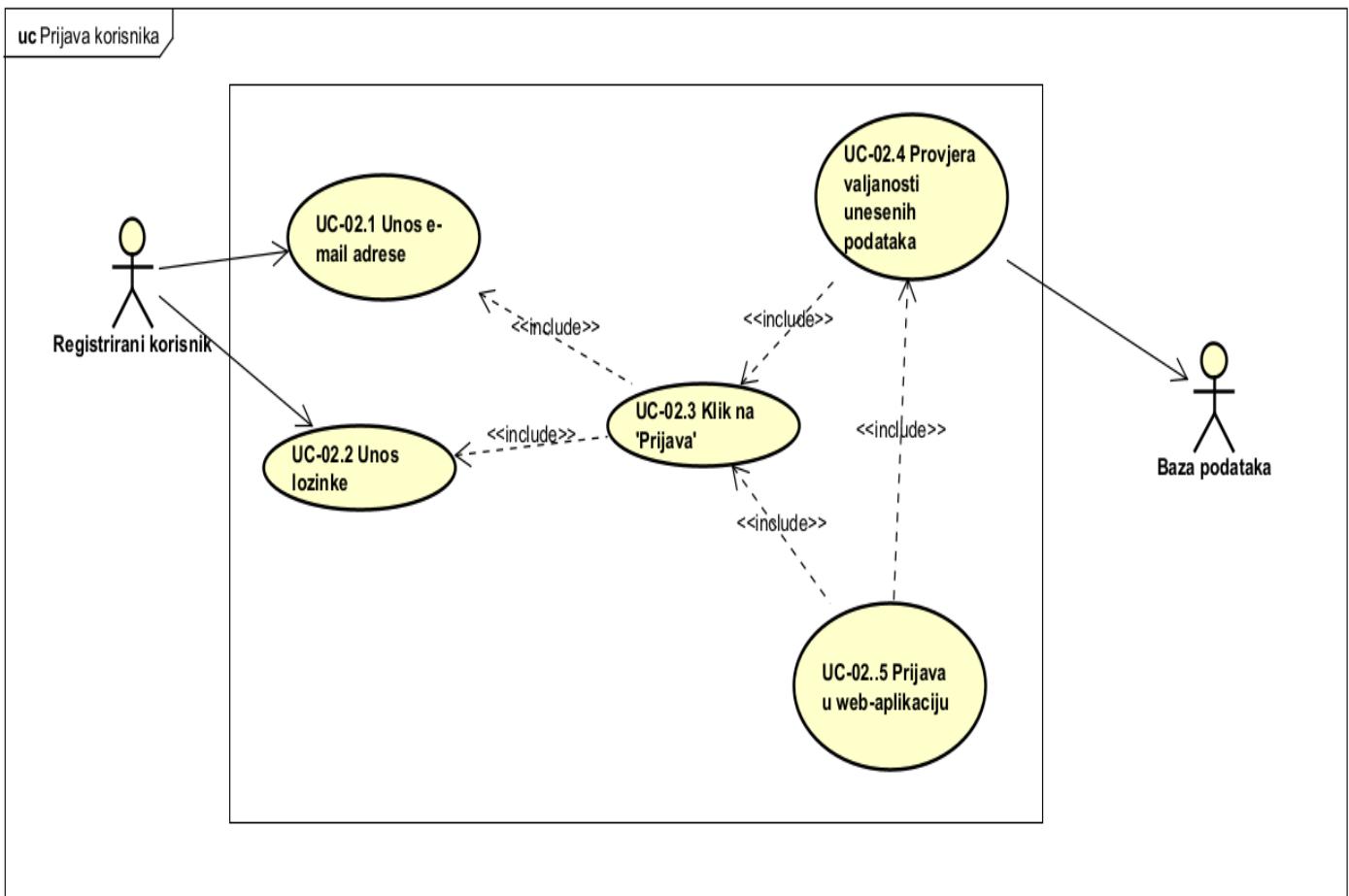


2. dijagram obrazaca uporabe za ključne značajke

Slika 3.2 Dijagram funkcionalnosti registracije

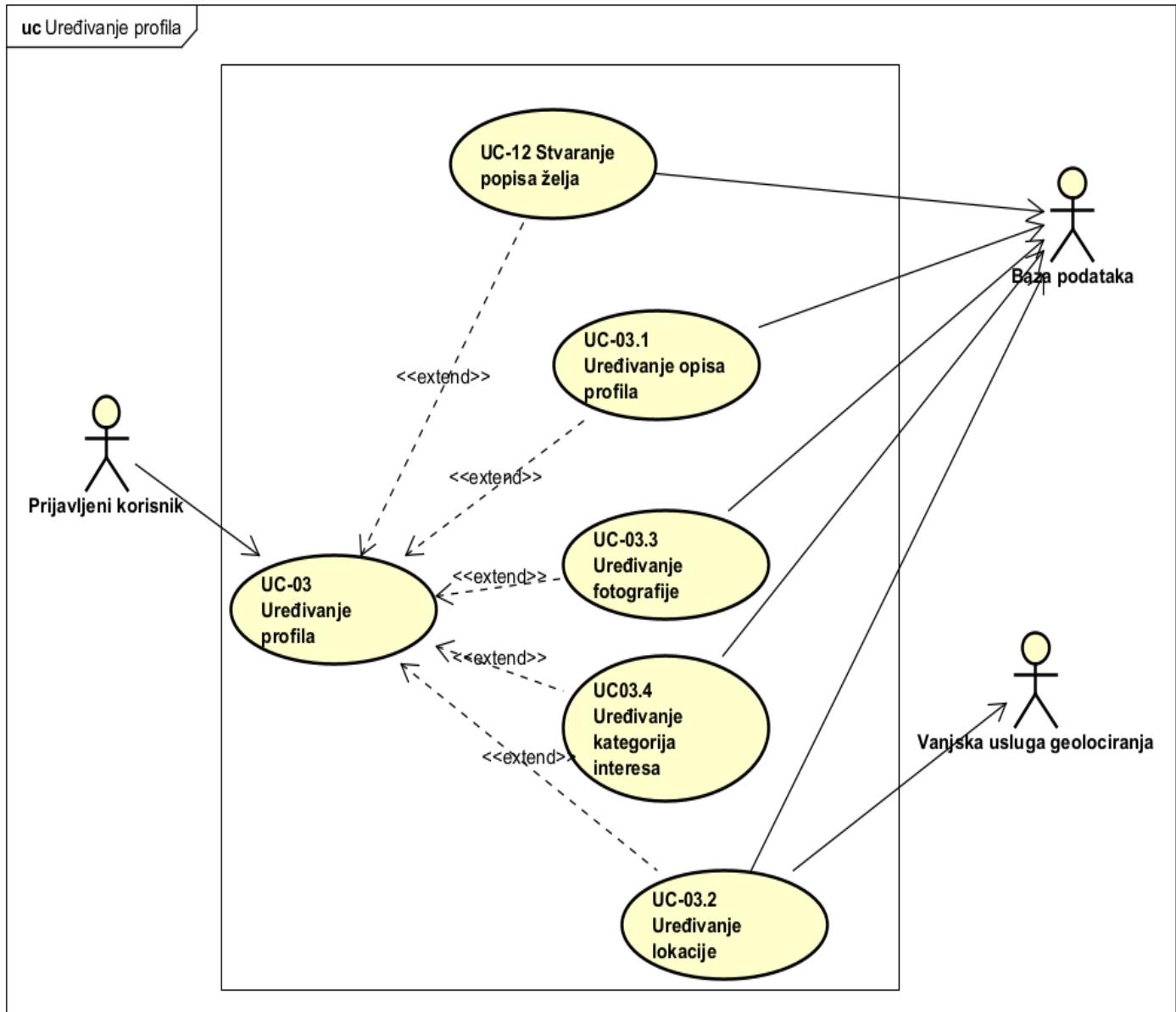


Slika 3.3 Dijagram funkcionalnosti prijave korisnika



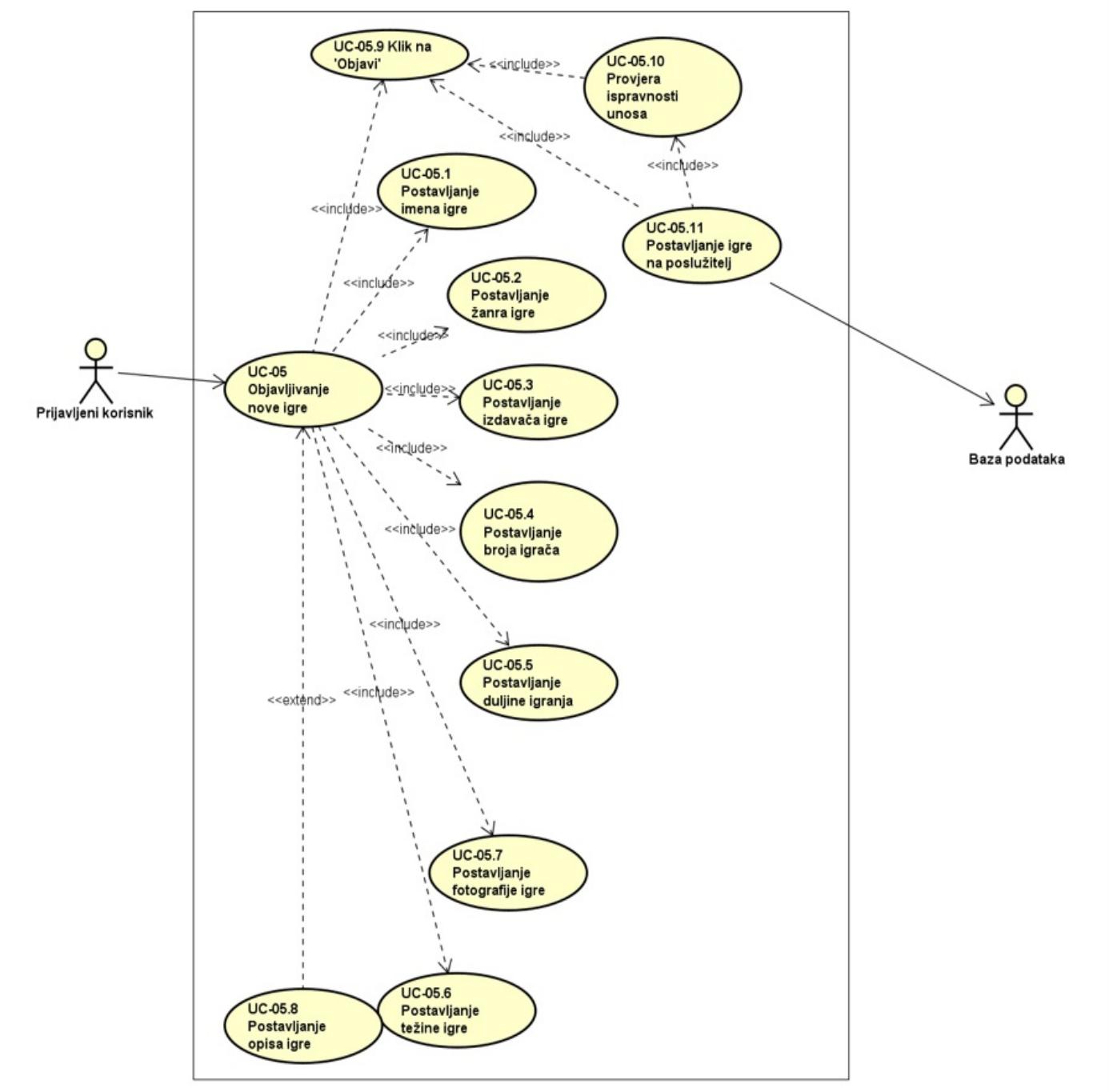
3. dijagram obrazaca uporabe za korisničke uloge

Slika 3.4 Dijagram funkcionalnosti uređivanja profila

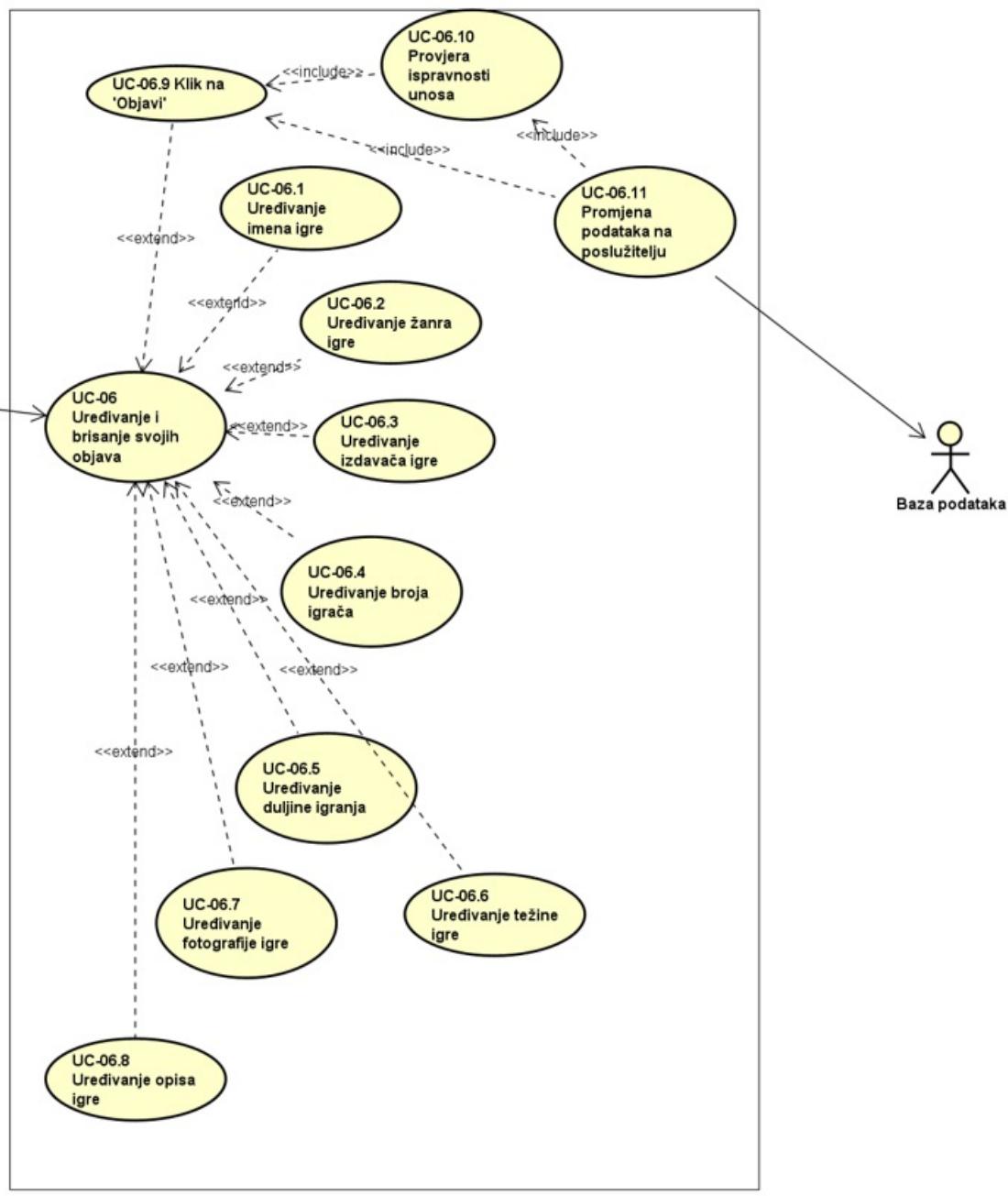


Slika 3.5 Dijagram objave nove igre

uc Objava nove igre

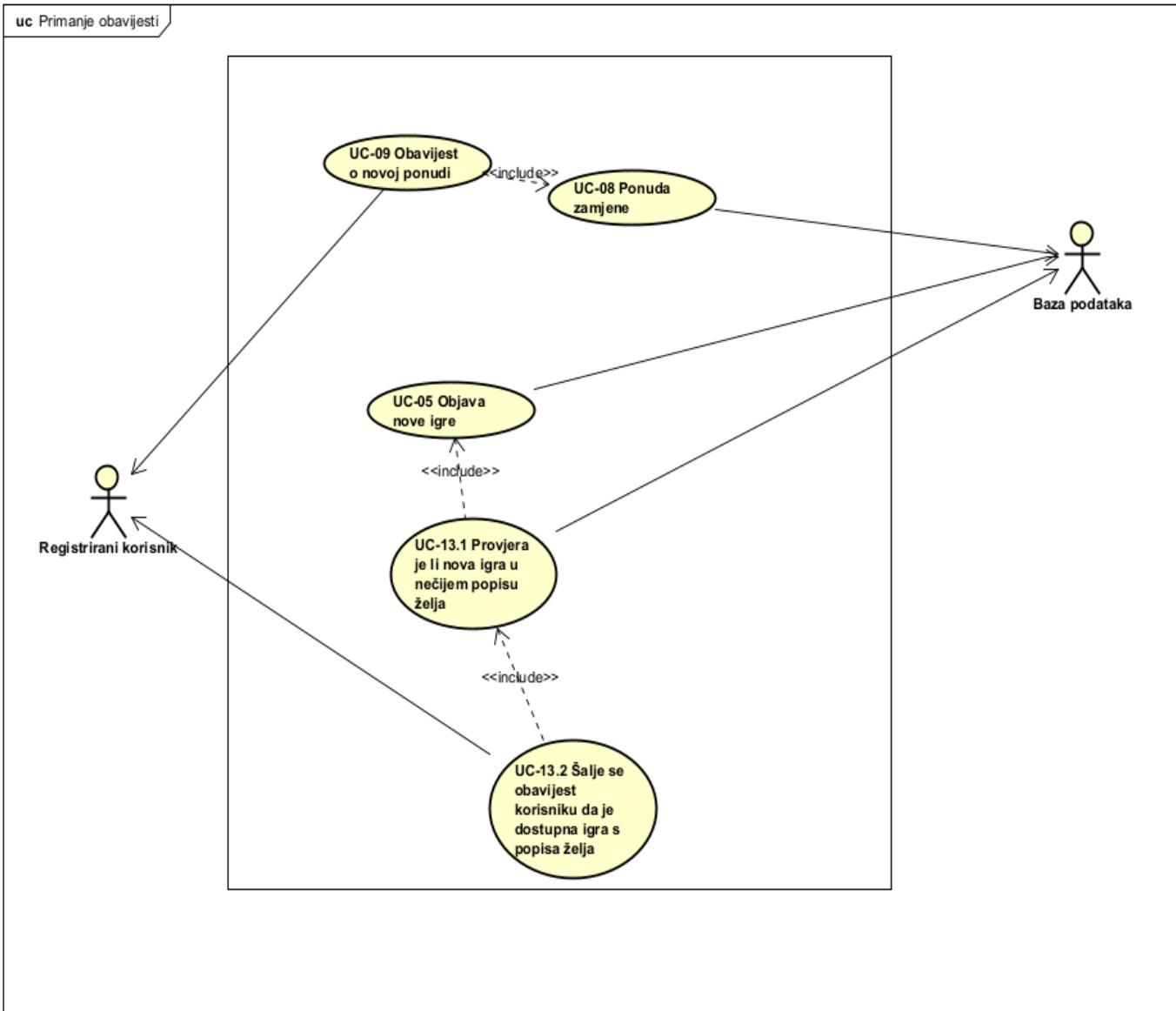


Slika 3.6 Dijagram uređivanja svoje objave igre

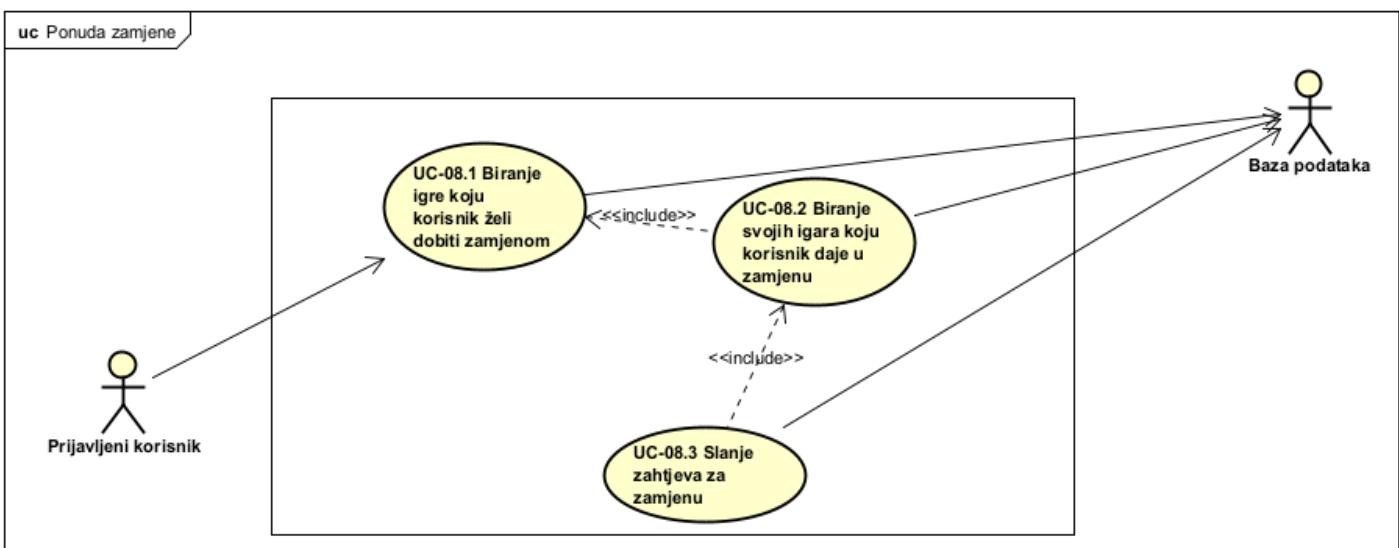


4. dijagram obrazaca uporabe za osnovne poslovne procese

Slika 3.7 Dijagram primanja i slanja obavijesti



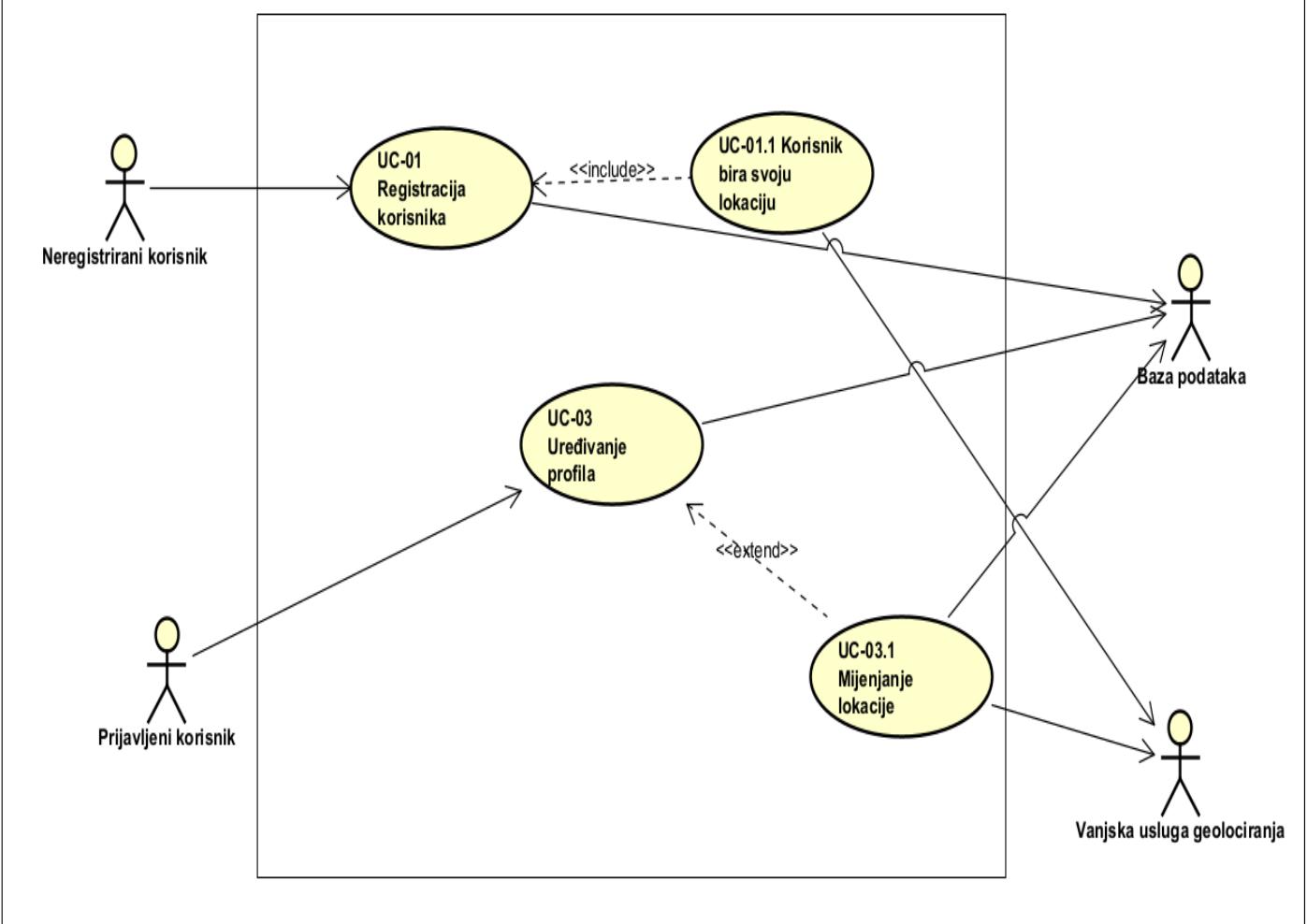
Slika 3.8 Dijagram ponude zamjene



5. dijagram obrazaca uporabe za kritične sustave i integracije

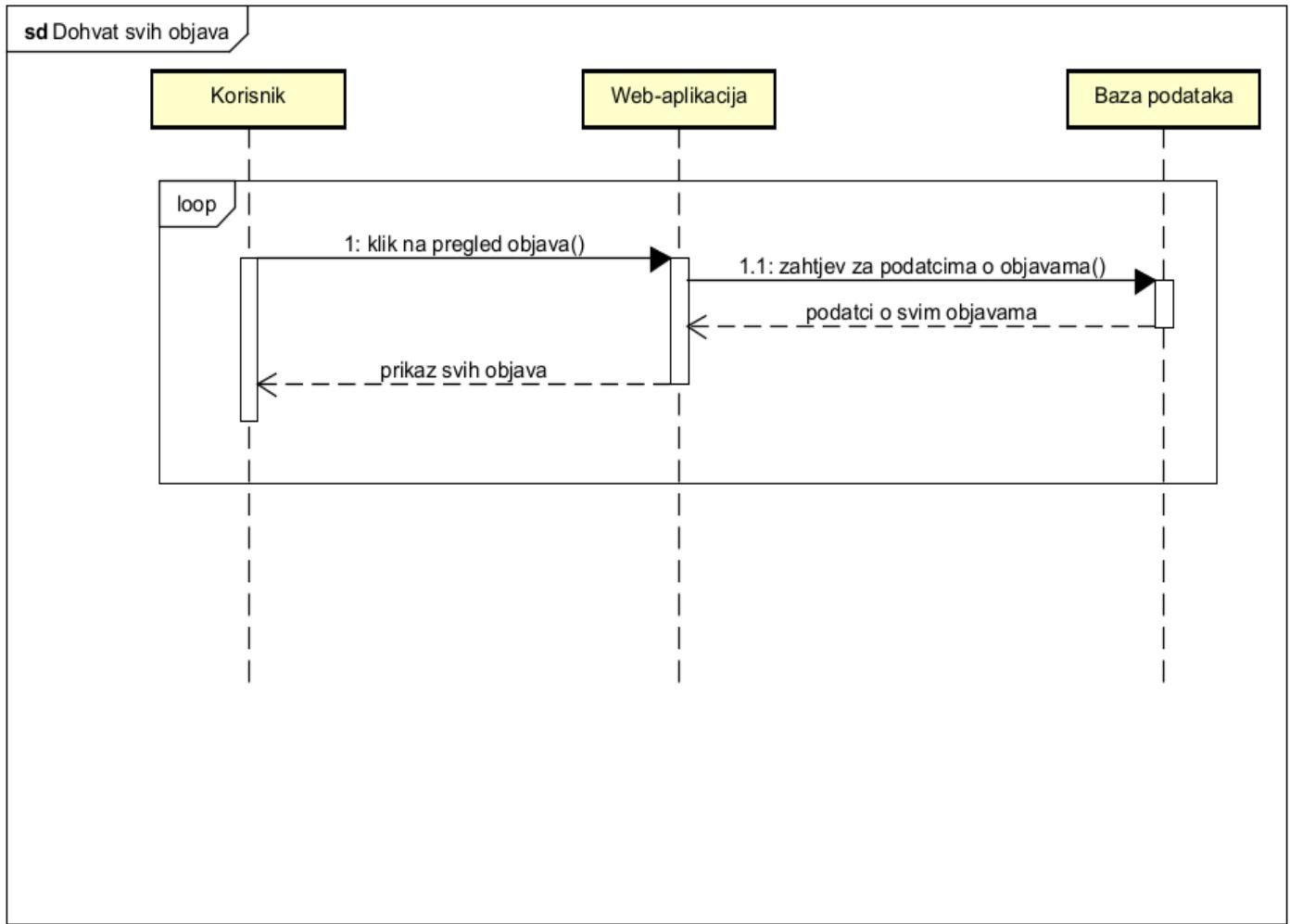
Slika 3.9 Dijagram korištenja vanjske usluge

ucKorištenje vanjske usluge

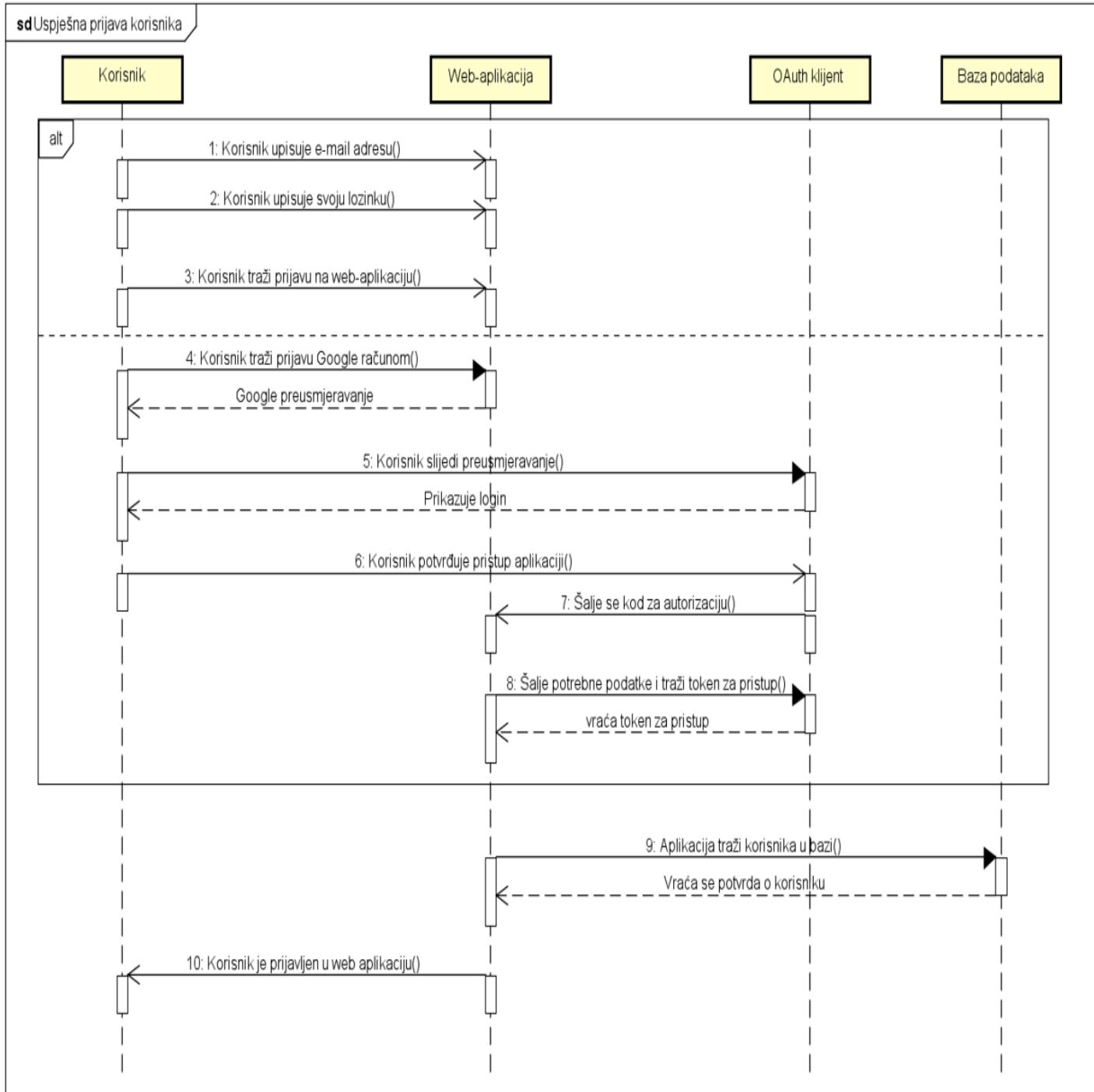


Sekvencijski dijagrami

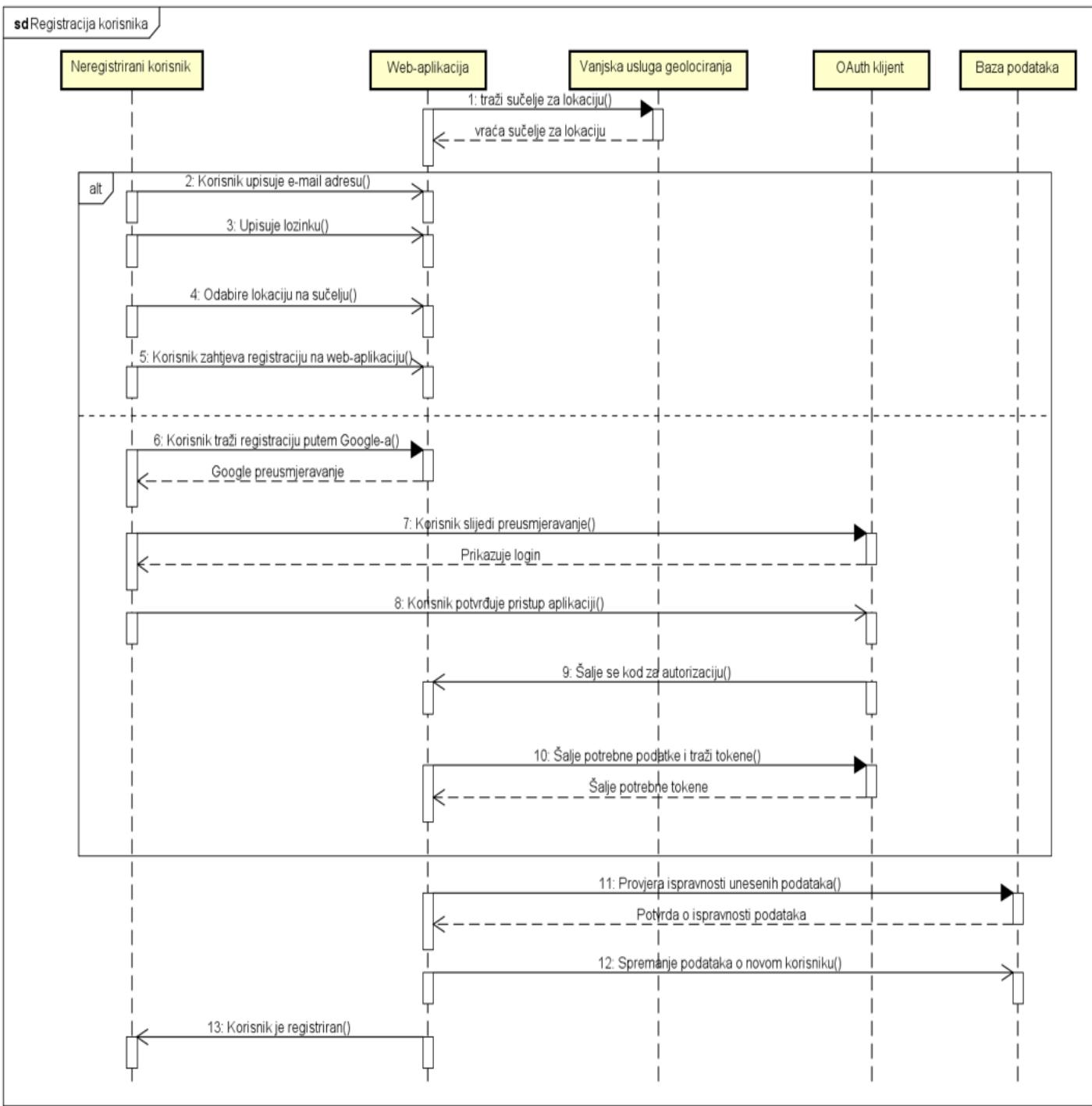
Slika 3.10 Sekvencijski dijagram dohvata svih objava



Slika 3.11 Sekvencijski dijagram uspješne prijave korisnika



Slika 3.12 Sekvencijski dijagram uspješne registracije korisnika



Arhitektura sustava

Stil arhitekture

Za ovaj je sustav odabrana **klijent-poslužitelj** arhitektura u kojoj su frontend i backend jasno odvojeni i međusobno komuniciraju putem RESTful API-ja.

Frontend je implementiran u Reactu, dok je backend razvijen u Node.js okruženju koristeći Express okvir.

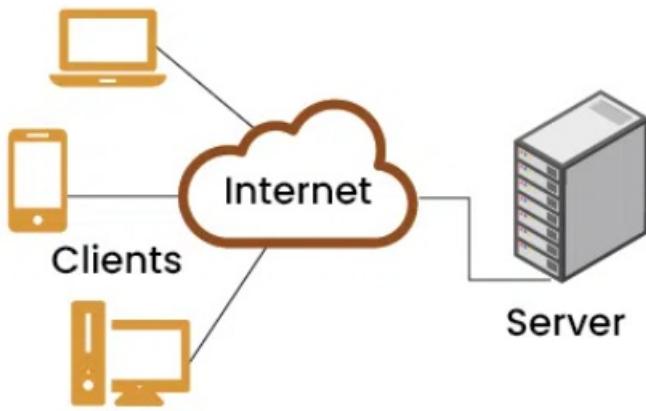
Podatci se razmjenjuju u JSON formatu putem HTTP protokola.

Ovaj arhitektonski stil odabran je jer omogućuje:

- odvajanje odgovornosti između korisničkog sučelja i poslovne logike
- neovisnost razvoja frontend i backend dijelova
- fleksibilnost i skalabilnost sustava
- jednostavno održavanje i buduće proširenje funkcionalnosti

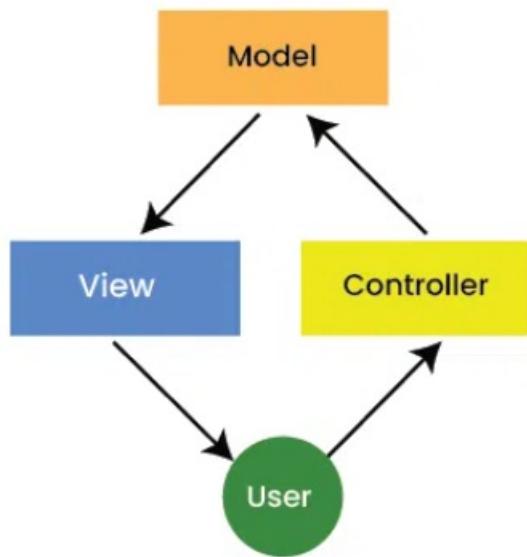
Klijent-poslužitelj arhitektura je najprikladnija jer sustav zahtjeva interakciju više korisnika u stvarnom vremenu te pouzdan prijenos podataka između različitih komponenti (frontend, backend, baza podataka).

Slika 4.1 Klijent-poslužitelj arhitektura



Osim toga, sustav slijedi i načela MVC arhitekture, u kojoj je klijent (frontend) odgovoran za View sloj, dok je poslužitelj (backend) implementiran kroz Model i Controller sloj.
Ovakav pristup omogućuje jasnu podjelu između prikaza, poslovne logike i upravljanja podacima, čime se dodatno povećava čitljivost i održivost koda. Naš model stoga kombinira klijent-poslužitelj stil i MVC obrazac, gdje React predstavlja View razinu, dok Node.js/Express backend sadrži Model (rad s bazom podataka) i Controller (obrada zahtjeva).

Slika 4.2 MVC arhitektura

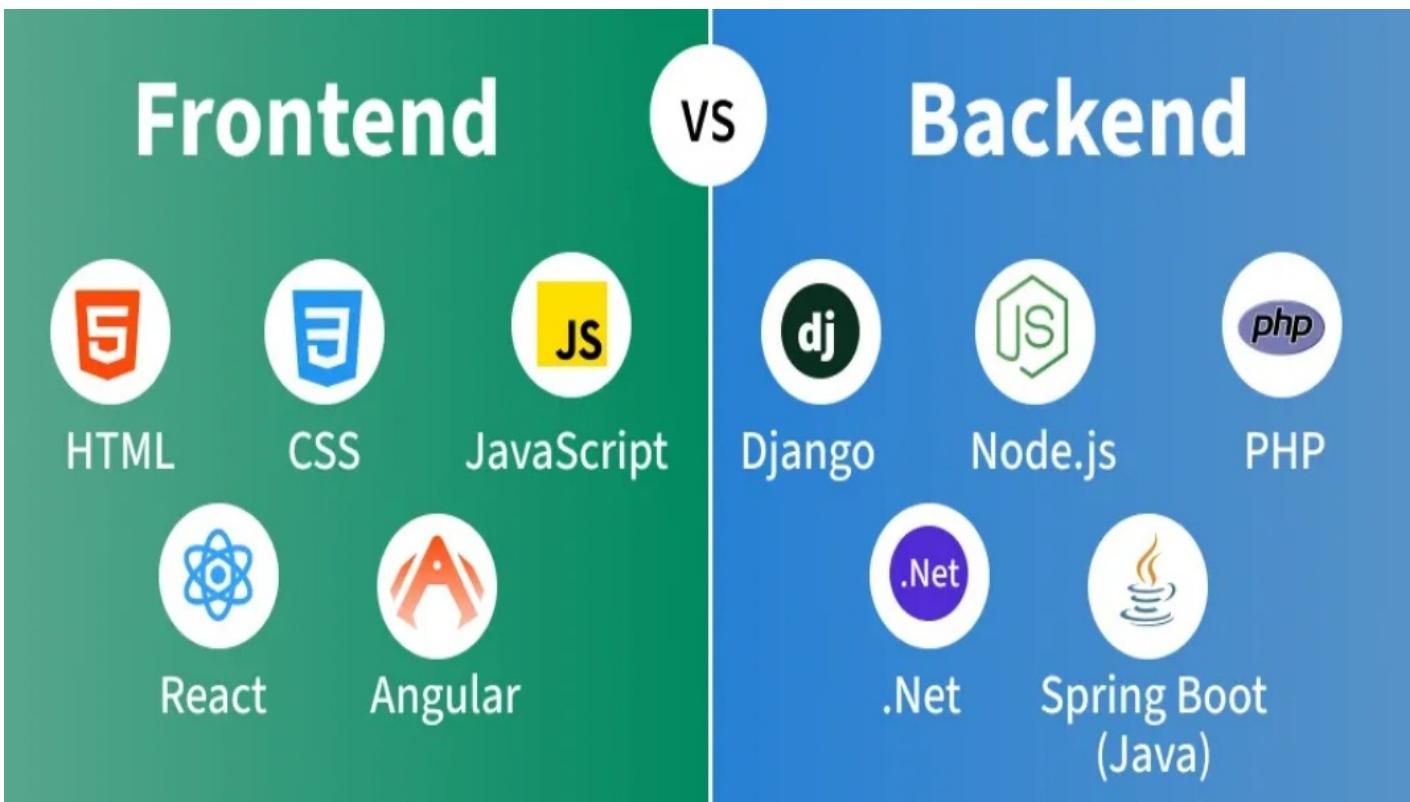


Podsustavi

Sustav je podijeljen u tri glavna podsustava:

1. Frontend podsustav (React)
 - odgovoran za prikaz korisničkog sučelja i interakciju s korisnikom
 - koristi HTTP zahtjeve za komunikaciju s backendom
 - implementira logiku prikaza, navigaciju i prikaz povratnih informacija
 - responsivan dizajn omogućuje i korištenje aplikacije na mobilnim uređajima
2. Backend podsustav (Node.js/Express)
 - implementira poslovnu logiku sustava i upravlja zahtjevima koje šalje frontend
 - koristi RESTful API za pružanje podataka i funkcionalnosti
 - obrada korisničkih zahtjeva uključuje autentifikaciju (OAuth 2.0), upravljanje korisnicima, igrama, ponudama i zamjenama
3. Baza podataka (MongoDB)
 - NoSQL baza podataka namijenjena za pohranu korisničkih podataka, oglasa igara, ponuda za zamjenu i arhive zamjena
 - podatci se pohranjuju u JSON-like dokumentima što omogućuje fleksibilnost i brzo dohvaćanje podataka

Slika 4.3 Prikaz alata često korištenih u razvoju frontenda i backenda



Preslikavanje na radnu platformu

Cijeli sustav bit će hostan na Microsoft Azure cloud platformi.

Frontend će biti distribuiran putem Azure Static Web Apps servisa, dok će backend biti implementiran i pokretan u Azure App Service okruženju.

Odabir Azure-a temelji se na:

- pouzdanosti i visokoj dostupnosti cloud arhitekture
- podršci za Node.js i React
- jednostavnoj integraciji OAuth 2.0 autentifikacije
- mogućnosti automatskog skaliranja i nadzora performansi

Spremišta podataka

Za pohranu podataka koristi se NoSQL baza podataka MongoDB.

MongoDB omogućuje fleksibilno modeliranje podataka pomoću JSON dokumenta, što olakšava kompleksnih struktura poput oglasa, ponuda i korisničkih profila.

Mrežni protokoli

Komunikacija između elemenata podsustava odvija se putem **HTTP(S)** protokola.

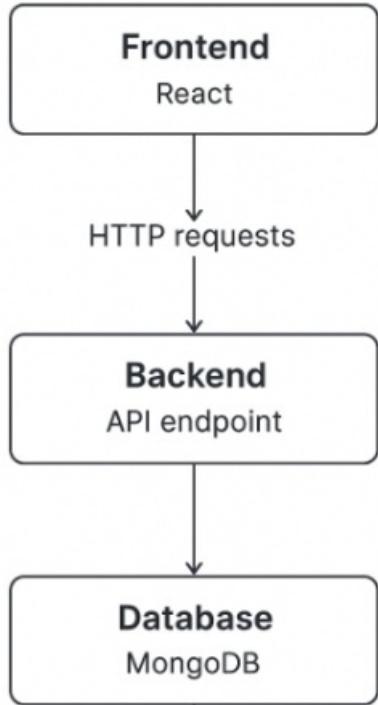
Frontend šalje zahtjeve prema backendu (npr. **GET**), a backend odgovara JSON objektima.

Sigurna autentifikacija korisnika odvija se putem **OAuth 2.0** protokola korištenjem tokena.

Globalni upravljački tok

1. Korisnik pristupa aplikaciji korištenjem preglednika (frontend)
2. React aplikacija šalje HTTP zahtjeve prema backendu
3. Backend provjerava korisničku autentifikaciju i obrađuje zahtjev prema odgovarajućem API endpointu
4. Backend komunicira s bazom podataka (MongoDB) i dohvaca ili mijenja potrebne podatke
5. Rezultat (u JSON formatu) se vraća frontend dijelu, koji ažurira prikaz korisniku
6. Ciklus se ponavlja ovisno o interakciji korisnika (npr. pregled oglasa, ponuda zamjene)

Slika 4.4 Pojednostavljeni prikaz upravljačkog toka



Sklopoškoprogramski zahtjevi

Frontend: podrška za moderne preglednike (Chrome, Edge, Firefox, Safari)

Backend: Node.js okruženje s Express frameworkom

Baza podataka: MongoDB Atlas instance

Operativni sustav: Windows, Linux ili macOS; Azure cloud za produkcijsko okruženje

Minimalni resursi (procjena): 4GB RAM-a za backend; 512MB RAM-a za frontend host

Obrazloženje odabira arhitekture

Odabir klijent-poslužitelj arhitekture temelji se na potrebi za **modularnim, skalabilnim i distribuiranim sustavom** koji omogućuje razvoj više komponenti paralelno.

Ključni razlog zbog kojeg smo se odlučili za zadalu arhitekturu jest: frontend i backend su **neovisni**, što olakšava timski rad i testiranje. Ali, osim toga, sustav temeljen na klijent-poslužitelj arhitekturi jest lakši za održavanje i upravljanje. Međutim morali smo napraviti kompromis što se tiče rizika, jer pad poslužitelja može paralizirati sustav, ali može doći i do zagušenja sustava jer se sav promet odvija kroz poslužitelja. Iz tog razloga smo se potrudili te rizike svesti na minimum. Prednosti odabrane arhitekture:

- visoka kohezija i niska povezanost između sustava
- skalabilnost - omogućuje horizontalno proširenje sustava dodavanjem novih servera
- modularnost - svaki dio sustava se može proširivati bez utjecaja na druge

Izbor arhitekture temeljen na principima oblikovanja

Principi koji pomažu u ostvarenju ciljeva projekta i način na koji pomažu:

- visoka **kohezija** i niska povezanost između sustava
- **fleksibilnost** u razvoju i održavanju
- **skalabilnost** - omogućuje horizontalno proširenje sustava dodavanjem novih servera
- **sigurnost** kroz OAuth 2.0 autentifikaciju i HTTPS komunikaciju
- **modularnost** - svaki dio sustava se može proširivati bez utjecaja na druge

Razmatrane alternative

Razmatrane alternative:

- **Monolitna arhitektura:** odbijena, jer bi otežavala održavanje i skaliranje zbog spajanja frontenda i backenda u jedinstvenu aplikaciju
- **Mikroslužba arhitektura:** odbačena zbog kompleksnosti i ograničenih resursa projekta; nije potrebna za trenutni opseg funkcionalnosti

Organizacija sustava na visokoj razini

1. Klijent-poslužitelj
 - React frontend (klijent) šalje zahtjeve Node.js backendu (poslužitelju), koji obrađuje podatke i komunicira s MongoDB bazom
2. Baza podataka
 - MongoDB kao NoSQL baza podataka omogućuje jednostavno dohvaćanje podataka po ključu i fleksibilnu pohranu

3. Grafičko sučelje:

- Web aplikacija izrađena u Reactu, responzivna i optimizirana za mobilne uređaje

Organizacija aplikacije

Aplikacija je organizirana prema klijent-poslužitelj arhitekturi, s jasnim razdvajanjem između frontend i backend slojeva. Ovakva struktura omogućuje neovisni razvoj, održavanje i skaliranje svake komponente sustava. Komunikacija između slojeva odvija se putem RESTful API-ja uz razmjenu podataka u JSON formatu.

Frontend sloj (React)

Frontend podsustav implementiran je u React okruženju i predstavlja korisnički prikaz aplikacije (engl. View layer). Njegove su osnovne odgovornosti:

- prikaz korisničkog sučelja i rukovanje interakcijama korisnika
- slanje HTTP zahtjeva prema backendu putem REST API-ja
- prikaz rezultata i povratnih informacija u stvarnom vremenu
- osiguravanje responzivnosti i prilagodljivosti sučelja na različitim uređajima

Frontend je organiziran prema komponentnom pristupu, gdje je svaka komponenta zadužena za određeni dio sučelja (npr. prikaz profila, oglas, ponuda zamjene). Takva struktura povećava ponovnu iskoristivost koda i olakšava održavanje. Za upravljanje stanjem aplikacije koristi se React-ov kontekstni sustav i lokalni storage, čime se omogućuje sinkronizacija podataka između različitih dijelova aplikacije.

Backend sloj (Node.js)

Backend podsustav razvijen je u Node.js okruženju koristeći Express framework. Njegova je primarna uloga implementacija poslovne logike i upravljanje zahtjevima koje dolaze s frontend sloja. Backend:

- prima i obrađuje HTTP zahtjeve korisnika
- provodi autentifikaciju i autorizaciju pomoću OAuth 2.0 protokola
- komunicira s bazom podataka (MongoDB) kroz modelne slojeve
- vraća odgovore u JSON formatu prema frontend aplikaciji

Ovaj sloj je organiziran prema MVC arhitekturnom obrascu, gdje se odgovornosti dijele na tri razine:

- Model (M) – predstavlja strukturu podataka i povezan je s bazom podataka. Modeli definiraju entitete poput korisnika, igara, oglasa i ponuda te pružaju metode za njihovo dohvaćanje, spremanje i ažuriranje
- View (V) – u našem slučaju, View sloj fizički se nalazi u frontend dijelu aplikacije (React), koji prikazuje podatke korisniku
- Controller (C) – sadrži logiku obrade zahtjeva. Controlleri primaju zahtjeve od frontenda, koriste modele za pristup podacima i vraćaju odgovarajuće odgovore prema View sloju

Baza podataka

Za pohranu podataka koristi se MongoDB, NoSQL baza podataka koja pohranjuje podatke u JSON-like dokumentima. Prednosti ovog pristupa su:

- fleksibilna struktura podataka
- brzo dohvaćanje i filtriranje
- lako skaliranje baze podataka

U bazi se pohranjuju podaci o korisnicima, oglasima, igram, ponudama zamjene i povijesti izvršenih transakcija.

Interakcija slojeva

1. Korisnik putem preglednika pristupa frontend aplikaciji (React).
2. Frontend šalje REST zahtjeve (GET, POST, PUT, DELETE) prema backendu.
3. Backend (Express) prima zahtjev, provjerava korisničke vjerodajnice (OAuth 2.0) i proslijeđuje zahtjev odgovarajućem kontroleru.
4. Kontroler koristi modele za komunikaciju s MongoDB bazom podataka.
5. Dobiveni podaci vraćaju se kao JSON objekt prema frontendu.
6. Frontend ažurira prikaz korisničkog sučelja prema dobivenim podacima.

Ovaj tok omogućuje jasno odvajanje odgovornosti i brzu dvosmjernu komunikaciju između korisničkog sučelja i poslužiteljske logike.

Zašto MVC u backendu

Backend dio arhitekture slijedi MVC arhitekturu jer:

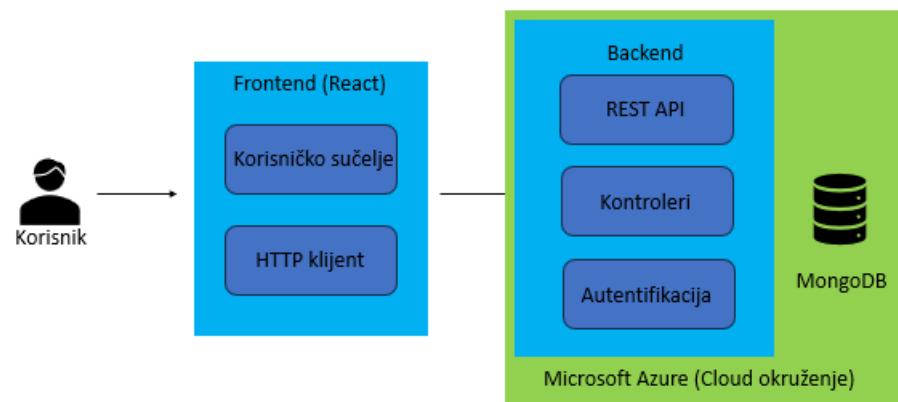
- Model sloj sadrži definicije podataka i interakcije s bazom (MongoDB kolekcije i Mongoose modeli),
- Controller sloj upravlja zahtjevima, validira podatke i kontrolira tok između Modela i frontend-a,
- View sloj je vanjski (React aplikacija), što znači da je vizualna prezentacija fizički odvojena od poslužiteljske logike.

Takva organizacija omogućuje:

- jednostavno testiranje i ponovno korištenje kontrolera i modela
- izolaciju grešaka
- olakšanu nadogradnju funkcionalnosti bez utjecaja na druge dijelove sustava

Na taj način sustav kombinira klijent-poslužitelj arhitekturu s MVC obrascem

Dijagram visoke razine (skica)



Slika 4.5 Dijagram visoke razine (skica)

Reference

- Programsko inženjerstvo, FER ZEMRIS, <http://www.fer.hr/predmet/proinz>
- Client-Server Architecture – System Design | GeeksforGeeks. Dostupno: <https://www.geeksforgeeks.org/system-design/client-server-architecture-system-design/> (10. studeni 2025.).
- MVC Architecture – System Design | GeeksforGeeks. Dostupno: <https://www.geeksforgeeks.org/system-design/mvc-architecture-system-design/> (10. studeni 2025.).
- Frontend vs Back-end Development | GeeksforGeeks. Dostupno: <https://www.geeksforgeeks.org/blogs/frontend-vs-backend/> (10. studeni 2025.).
- MongoDB Documentation. Dostupno: <https://www.mongodb.com/docs/> (7. studeni 2025.).
- React – Learn React. Dostupno: <https://react.dev/learn> (7. studeni 2025.).
- Node.js – Introduction to Node.js. Dostupno: <https://nodejs.org/en/learn/getting-started/introduction-to-nodejs> (7. studeni 2025.).

Baza podataka

Odabrali smo MongoDB kao sustav za upravljanje bazom podataka jer se izvrsno uklapa s našim Node.js backendom, budući da koristi BSON (Binary JSON) dokumente čija je struktura vrlo slična JSON formatu s kojim Node.js prirodno radi.

MongoDB je NoSQL dokumentno orijentirani sustav, što znači da podatke pohranjuje u obliku fleksibilnih dokumenata unutar kolekcija, a ne u tablicama kao relacijske baze. Svaki dokument može sadržavati različita polja i ugniježđene strukture, što omogućuje veću prilagodljivost u radu s dinamičnim podacima.

Za razliku od relacijskih baza podataka, MongoDB ne koristi primarne i strane ključeve niti stroge relacijske veze između entiteta. Umjesto toga, veze između kolekcija ostvaruju se putem referenci (ObjectId) ili ugnježđivanjem dokumenata, ovisno o potrebama sustava.

Ovakav pristup omogućuje brže dohvaćanje podataka, jednostavnije skaliranje i lakšu prilagodbu strukture baze tijekom razvoja aplikacije.

Opis tablica

users

Atribut	Tip podatka	Opis variabile
_id	ObjectId	Jedinstveni identifikator
email	String	Email korisnika
passwordHash	String	Hashirana lozinka
username	String	Username korisnika
scope	String	Uloga korisnika
location	Object	Objekt za pohranu lokacije
profile	Object	Objekt za pohranu detalja profila
createdAt	Date	Datum registracije korisnika
isActive	Boolean	Označava je li korisnik aktivan

Kolekcija users sadrži sve registrirane korisnike aplikacije. Svaki korisnik ima osnovne podatke, kategorije igara koje ga interesiraju koje se nalaze unutar objekta profile, unutar objekta profile se još nalazi opis koji korisnik može napisati i slika koju može objaviti te korisnik ima lokaciju dobivenu putem vanjske geolokacijske usluge (OpenStreetMap) čiji se detalji nalaze unutar objekta location.

games

Atribut	Tip podatka	Opis varijable
_id	ObjectId	Jedinstveni identifikator
name	String	Naziv igre koju je korisnik objavio
genre	String	Naziv žanra igre
publisher	String	Naziv izdavača igre
releaseYear	Int	Godina izlaska igre
minPlayers	Int	Najmanji broj igrača za igranje igre.
maxPlayers	Int	Najveći broj igrača za igranje igre.
playtime	Int	Duljina prosječne partije
difficulty	Int	Težina igre
imageUrl	String	Slika igre

Kolekcija games predstavlja sve igre koje korisnici objave kao dostupne za razmjenu.

listings

Atribut	Tip podatka	Opis varijable
_id	ObjectId	Jedinstveni identifikator
userId	ObjectId	Referenca na korisnika koji postavlja igru na web-stranicu
gameId	ObjectId	Referenca na igru koju je korisnik postavio u oglasu
condition	String	Ocjena očuvanosti igre
available	Boolean	Je li igra dostupna za zamjenu
description	String	Dodatni opis oglasa
createdAt	Date	Datum objave oglasa

Kolekcija listings predstavlja oglase koje korisnici objavljaju za igre koje žele zamijeniti. Svaki oglas je povezan s korisnikom i određenom igrom.

trades

Atribut	Tip podatka	Opis varijable
_id	ObjectId	Jedinstveni identifikator
initiatorId	ObjectId	Referenca na korisnika koji nudi zamjenu
receiverId	ObjectId	Referenca na korisnika koji prima ponudu
offeredListing	Array	Oglas koji korisnik nudi u zamjeni
requestedListing	Array	Oglas koji korisnik traži u zamjeni
status	String	Stanje zamjene
messages	Array	Poruke koje korisnik šalje

createdAt	Date	Tip podatka	Datum inicijacije zamjene
Atribut			Opis varijable

Kolekcija trades bilježi sve zamjene između korisnika. Svaka zamjena ima inicijatora, primatelja, listu ponuđenih i traženih oglasa te trenutni status. U messages se nalaze objekti koji u sebi sadrže senderId, text i vrijeme slanja poruke.

wishlist

Atribut	Tip podatka	Opis varijable
_id	ObjectId	Jedinstveni identifikator
userId	ObjectId	Referenca na korisnika koji je kreirao svoj wishlist
games	Array	Sve igre koje korisnik želi imati
notificationsEnabled	Boolean	Ima li korisnik uključene notifikacije

Kolekcija wishlist sadrži popise željenih igara svakog korisnika. Koristi se za obavještanje korisnika kada neka od željenih igara postane dostupna u oglasima.

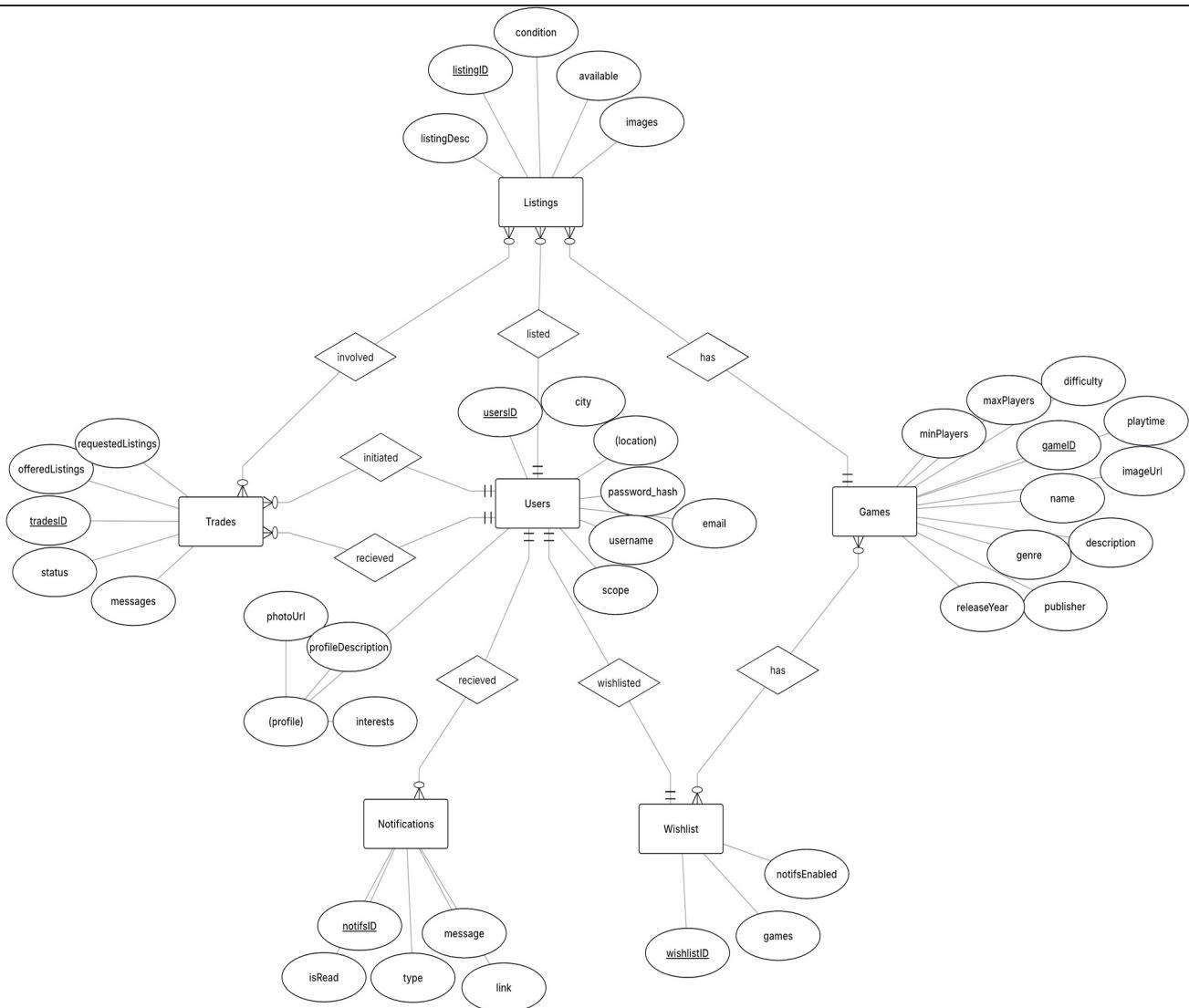
notifications

Atribut	Tip podatka	Opis varijable
_id	ObjectId	Jedinstveni identifikator
userId	ObjectId	Referenca na korisnika koji prima notifikaciju
type	String	Vrsta notifikacije
message	String	Sadržaj notifikacije
link	String	link notifikacije
isRead	Boolean	Je li notifikacija pročitana
createdAt	Date	Datum primanja notifikacije

Kolekcija notifications pohranjuje sve obavijesti koje korisnici primaju (npr. o novim ponudama, odgovorima na zamjenu ili dostupnim igramu s wishliste).

Dijagram baze podataka

Slika 4.6 Dijagram baze podataka



Budući da je baza podataka izrađena u NoSQL sustavu, ER dijagram nema izravnu funkcionalnu ulogu. Ipak, smatramo da ga je korisno priložiti kako bi se jasno prikazale sve kolekcije i odnosi među njima.

Dijagram razreda

Slika 4.7 Dijagram razreda Napomena: zbog veličine dijagrama tip slike je svg format. Da bi se mogla vidjeti detalji u dijagramu razreda potrebno je otvoriti sluku direktno(te proizvoljno smanjiti veličinu) preko linka: [Dijagram razreda](#)

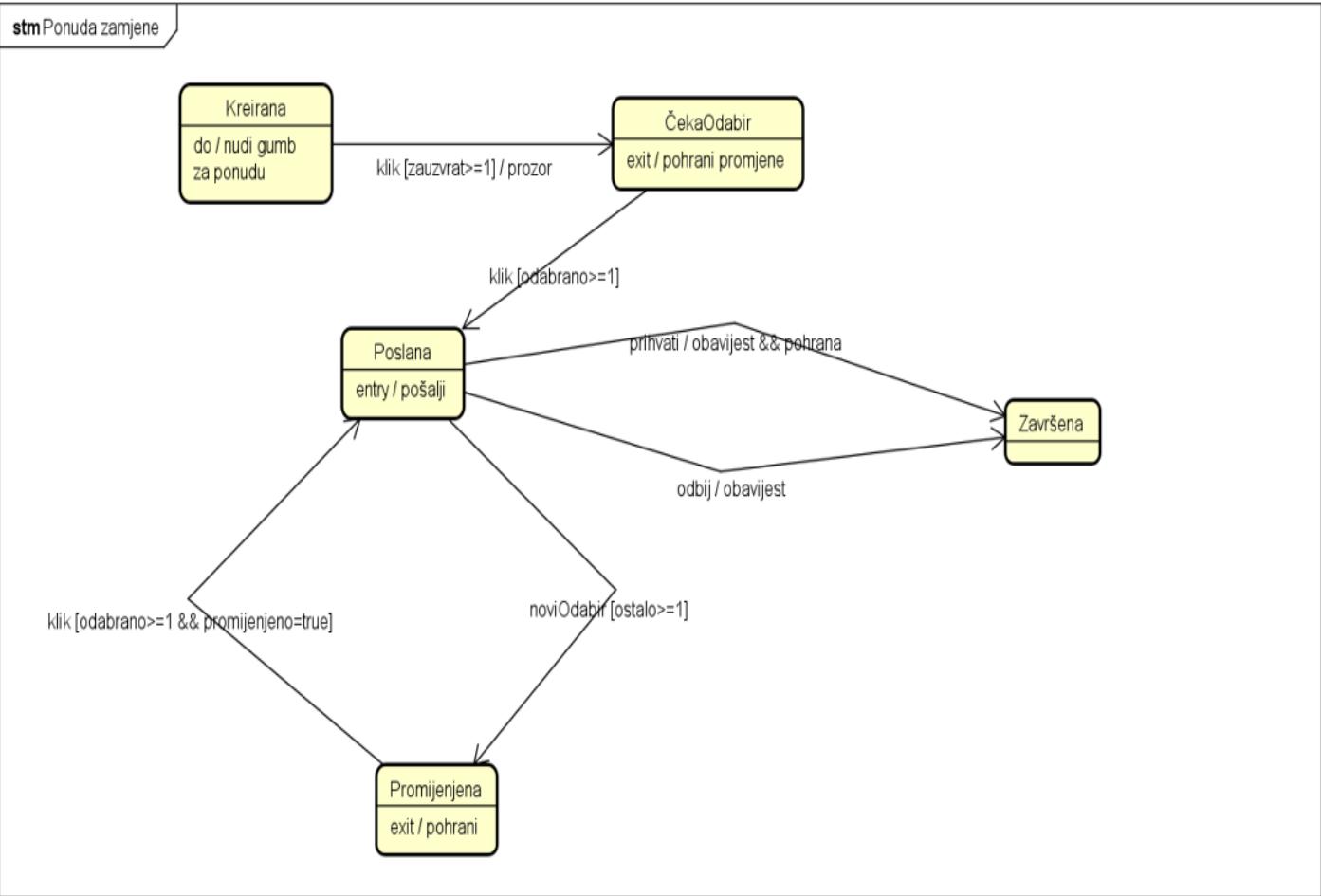
Dinamičko ponašanje aplikacije

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.

U nastavku su prikazani dijagrami stanja i dijagrami aktivnosti koji opisuju pojedine elemente sustava tij njihovu međusobnu dinamiku.

UML dijagrami stanja

Slika 4.8 Dijagram stanja za proces zamjene



Stanja :

- Kreirana - objavu igre kreirao je neki korisnik i ona je vidljiva drugim korisnicima i spremna za ponudu (POČETNO STANJE)
- ČekaOdabir - u prozoru za ponude korisnik koji traži zamjenu odabire što daje zauzvrat za traženu igru
- Poslana - ponuda je poslana prema nekom korisniku, čeka se njegova reakcija na nju
- Promijenjena - neki korisnik je u procesu mijenjanja uvjeta zamjene
- Završena - jedan od korisnika je konačno odbio ili prihvatio ponudu (KONAČNO STANJE)

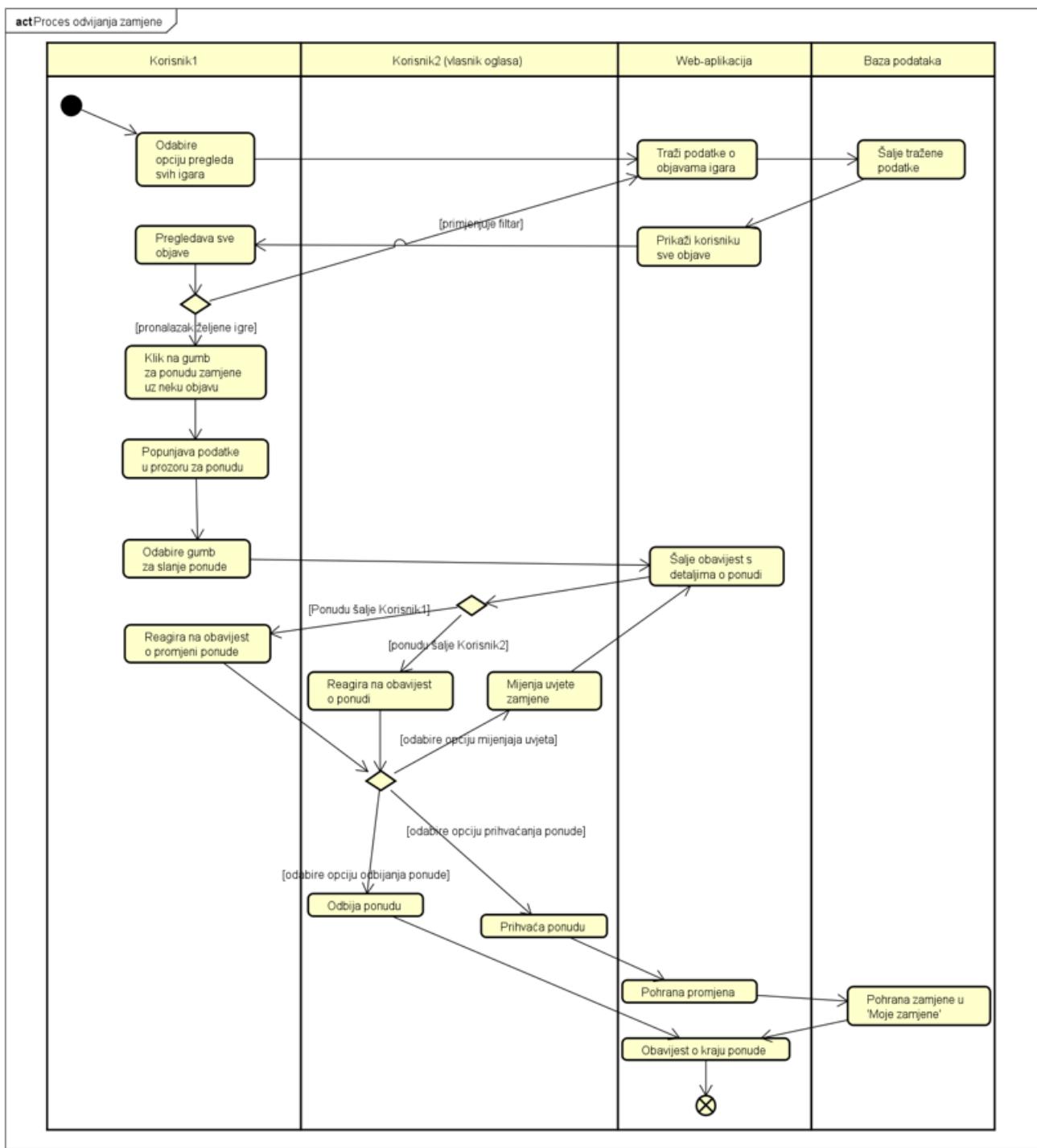
Entry/ Do/ Exit :

- nudi gumb za ponudu - nećija objava kraj sebe ima gumb kojim ostali korisnici mogu zatražiti zamjenu
- pohrani promjene - podatci o ponudi spremaju se u bazu podataka
- pošalji - korisniku s druge strane ponude šalje se obavijest o detaljima ponude na stranicu i na e-mail
- pohrani - ažuriraju se podatci o promjeni uvjeta zamjene

Prijelazi

- klik [zauzvrat>=1] / prozor - prijelaz se pokreće klikom na gumb za davanje ponude, prijelaz se odvija samo ako ponuditelj ima barem jednu svoju objavljenu igru koju može nuditi za zamjenu, akcija je otvaranje prozora za ponudu* klik [odabran>=1] - prijelaz se pokreće kad se klikne na gumb za slanje ponude, a odvija se samo ako je korisnik odabrao barem jednu svoju igru koju nudi za zamjenu
- noviOdabir [ostalo>=1] - prijelaz se pokreće kad korisnik zatraži promjenu uvjeta zamjene, a odvija se samo ako korisnik s druge strane zamjene ima više od jedne objavljene igre, odnosno, ako se ima išta za mijenjati u uvjetima zamjene
- klik [odabran>=1 && promijenjeno=true] - prijelaz se pokreće kad korisnik klikne na gumb za slanje ponude s novim uvjetima zamjene, a prijelaz se odvija samo ako su uvjeti zamjene mijenjani na neki način i ako broj igara koji se traži nije jednak 0
- odbij / obavijest - prijelaz se pokreće kad jedan od korisnika odbije ponudu zamjene, akcija je slanje obavijesti o odbijanju drugoj strani zamjene o odbijanju ponude
- prihvati / obavijest && pohrana - prijelaz se pokreće kad jedan od korisnika prihvati ponudu zamjene, akcije su slanje obavijesti drugoj strani zamjene o prihvaćanju ponude i pohrana uspješne zamjene u kategoriju 'Moje zamjene' svakog korisnika

UML dijagrami aktivnosti



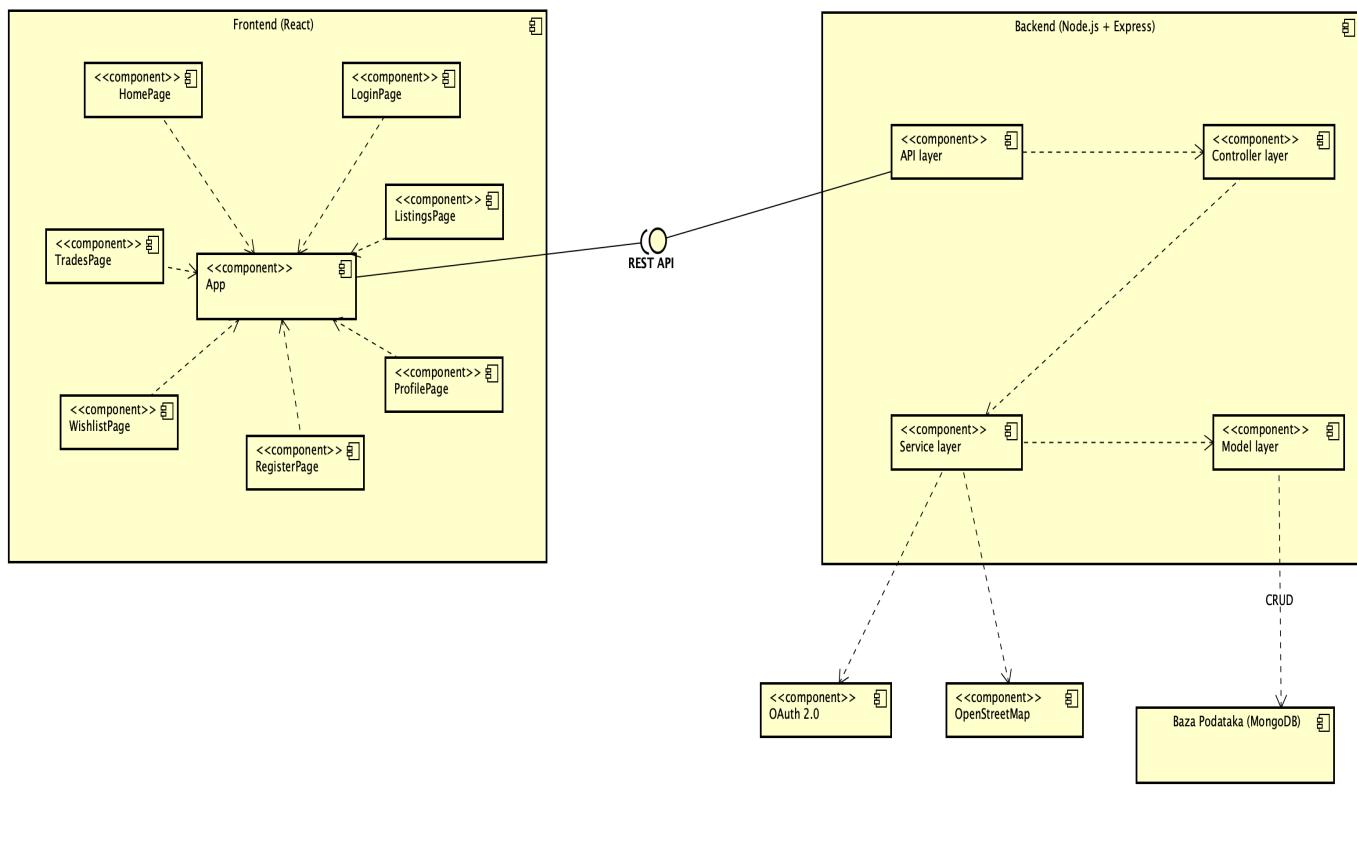
Particije :

- Korisnik1 - onaj korisnik sustava koji prvi daje ponudu za neku objavu
- Korisnik2 - onaj korisnik koji je vlasnik oglasa za kojeg Korisnik1 daje ponudu
- Web-aplikacija - PlayTrade
- Baza podataka - zadužena za spremanje i pružanje svih podataka sustava

Opis tijeka aktivnosti :

1. Korisnik1 pristupa sustavu i zahtijeva pristup svim objavama drugih korisnika (prepostavka je da je registriran i prijavljen)
2. Web-aplikacija od baze podataka traži i dobije podatke o svim objavama i pritom ih prikaze korisniku
3. Korisnik1 pregledava sve dobivene igre ili primjenjuje filter prilikom čega se proces ponavlja od točke 2.
4. Korisnik1 pronađe objavu igre koju želi dobiti zamjenom i odabire opciju za davanje ponude vlasniku te objave (Korisnik2)
5. Korisnik1 odabire koje svoje igre daje zauzvrat u zamjenu za traženu
6. Korisnik1 potvrđuje slanje ponude
7. Web-aplikacija šalje obavijest o ponudi drugoj strani ponude
8. Druga strana ponude reagira na ponudu na jedan od tri načina: prihvata ju (nastavak na korak 9.), odbija ju (skok na korak 10.), ili mijenja uvjete ponude (povratak na korak 7.)
9. Ako je ponuda prihvaćena web-aplikacija sprema u bazu podataka podatke o izvršenoj zamjeni (u kategoriji 'Moje zamjene' obaju korisnika)
10. Šalje se obavijest o zatvaranju ponude

Dijagram komponenata



Dijagram komponenata prikazuje logičku strukturu aplikacije kroz skup međusobno povezanih funkcionalnih komponenti. Svaka komponenta predstavlja zasebnu cjelinu s jasno definiranim odgovornostima, dok se komunikacija između komponenti odvija putem definiranih sučelja.

Frontend sloj (React)

Frontend dio aplikacije implementiran je pomoću React biblioteke te je organiziran oko središnje komponente App, koja služi kao glavna ulazna točka aplikacije i upravlja navigacijom između pojedinih stranica.

Unutar frontend sloja nalaze se sljedeće komponente: HomePage, LoginPage, RegisterPage, ListingsPage, TradesPage, WishlistPage, ProfilePage.

Navedene komponente predstavljaju pojedine stranice aplikacije te su ovisne o komponenti App, koja koordinira njihovo prikazivanje i upravljanje stanjem aplikacije. Frontend komunicira s backend dijelom sustava putem REST API sučelja, koristeći HTTP(S) protokol.

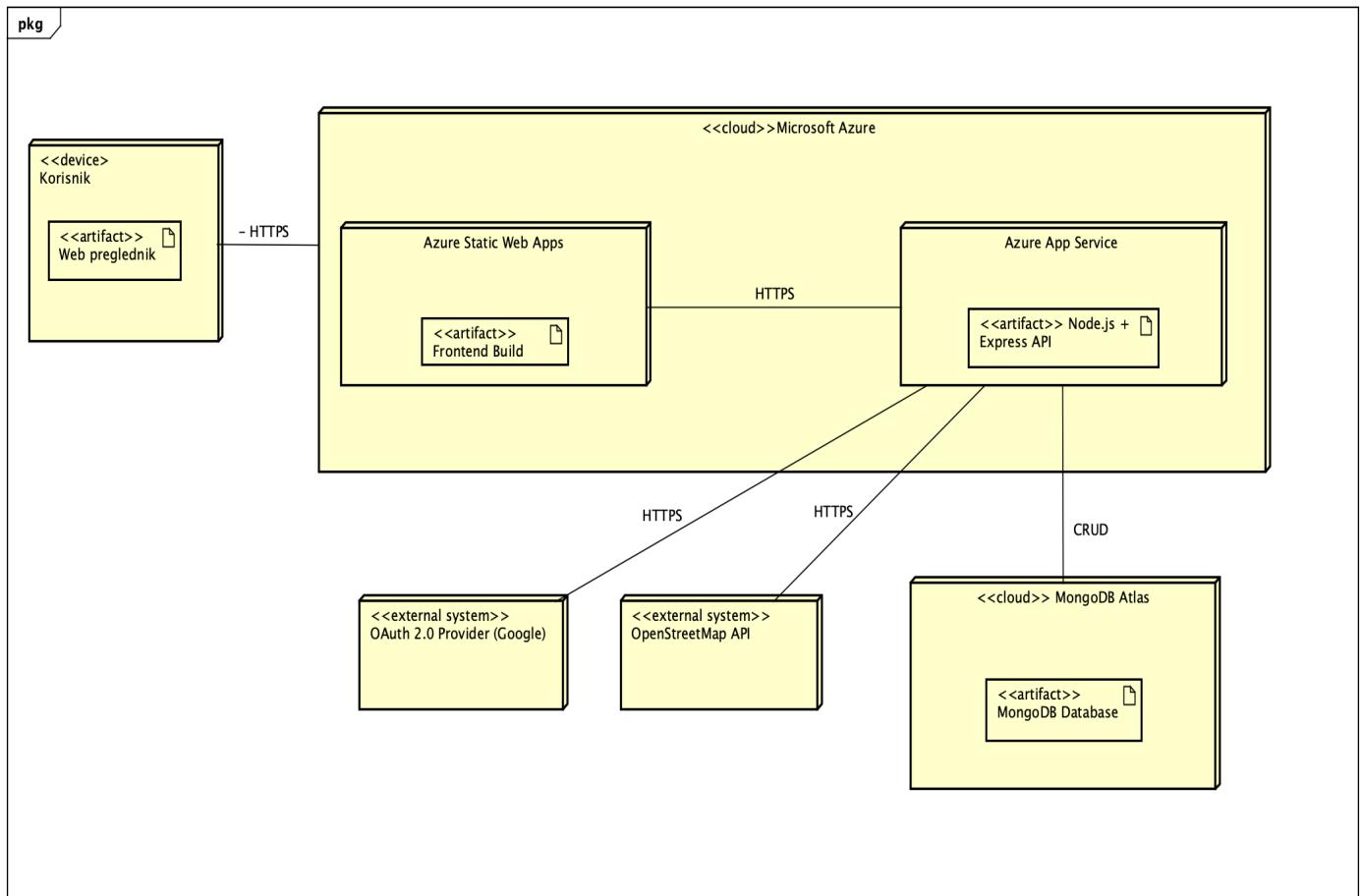
Backend sloj (Node.js + Express)

Backend dio aplikacije implementiran je korištenjem Node.js platforme i Express okvira te je organiziran prema slojevitoj arhitekturi.

Backend se sastoji od sljedećih komponenti: API / Routes sloj, koji definira dostupne REST rute i prima zahteve s frontend strane, Controller sloj, koji obrađuje dolazne zahteve i koordinira daljnju logiku obrade, Service sloj, koji sadrži poslovnu logiku aplikacije, Model sloj, koji predstavlja podatkovni sloj aplikacije i omogućuje pristup bazi podataka.

Service sloj koristi vanjske servise poput OAuth 2.0 sustava za autentikaciju korisnika te OpenStreetMap API-ja za dohvati i obradu geografskih podataka. Model sloj zadužen je za trajnu pohranu podataka i komunikaciju s bazom podataka MongoDB.

Dijagram razmještaja



Dijagram razmještaja prikazuje fizičku i virtualnu raspodjelu komponenti sustava unutar infrastrukturnog okruženja. Dijagram je izrađen u implementacijskom obliku te prikazuje stvarne servise, čvorove i artefakte na kojima se aplikacija izvršava.

Klijentski sloj

Korisnik pristupa aplikaciji putem uređaja (npr. osobnog računala ili mobilnog uređaja) koristeći web preglednik. Komunikacija između korisnika i frontend dijela aplikacije odvija se putem sigurnog HTTPS protokola.

Cloud infrastruktura – Microsoft Azure

Aplikacija je implementirana unutar Microsoft Azure okruženja, koje sadrži sljedeće servise: Azure Static Web Apps, koji hosta frontend dio aplikacije. Na ovom servisu nalazi se izgrađeni React frontend (Frontend Build) koji se isporučuje korisnicima. Azure App Service, na kojem je implementiran backend aplikacije u obliku Node.js + Express API-ja. Ovaj servis obrađuje dolazne zahtjeve, provodi poslovnu logiku i upravlja pristupom podacima.

Frontend i backend komuniciraju međusobno putem HTTPS REST API poziva.

Baza podataka – MongoDB Atlas

Za trajnu pohranu podataka koristi se MongoDB Atlas, koji predstavlja cloud bazu podataka izvan Azure okruženja. Backend aplikacija komunicira s bazom podataka putem MongoDB drivera, koristeći CRUD operacije za upravljanje podacima.

Vanjski servisi

Sustav koristi i vanjske servise koji nisu dio glavne infrastrukture: OAuth 2.0 Provider, koji omogućuje autentikaciju i autorizaciju korisnika putem sigurnog protokola i OpenStreetMap API, koji se koristi za dohvrat geografskih i kartografskih podataka.

Komunikacija s navedenim servisima odvija se putem HTTPS protokola, a integracija je ostvarena unutar backend sloja aplikacije. # Ispitivanje komponenti Ovo poglavљje treba opisati provedena ispitivanja implementiranih funkcionalnosti na razini komponenti i sustava. Fokus je na odabiru i izvedbi ispitnih slučajeva koji obuhvaćaju redovne, rubne uvjete i testiranje grešaka, kao i upotrebu odgovarajućih alata za provedbu testiranja.

Postupak testiranja provodi se Jest-om. Test se pokreće nardbom npm test.

```
C:\PROGI\error808\backend>npm test
> error808-backend@0.0.0 test
> jest

PASS  tests/controllers/auth/registerController.test.js
PASS  tests/controllers/auth/loginController.test.js
PASS  tests/controllers/listingsController.test.js

Test Suites: 3 passed, 3 total
Tests:       15 passed, 15 total
Snapshots:   0 total
Time:        1.273 s
Ran all test suites.
```

slika 6.1.: Jest test

Testiranje filtera Difficulty i Number Of Players

Mock-ani podatci		
Opis	Očekivani podatci	Dobiveni rezultat
Postavljanje filtera - Difficulty: easy	{ name: "Easy Game", difficulty: 1, maxPlayers: 2}	{ name: "Easy Game", difficulty: 1, maxPlayers: 2}
Postavljanje filtera - Number Of Players: 2-4 players	{ name: "Medium Party Game", difficulty: 2, maxPlayers: 4}	{ name: "Medium Party Game", difficulty: 2, maxPlayers: 4}
Postavljanje filtera - Difficulty: hard && Number Of Players: 4+ palyers	{ name: "Hard Strategy Game", difficulty: 4, maxPlayers: 6}	{ name: "Hard Strategy Game", difficulty: 4, maxPlayers: 6}

link: [Filter-Search Unit Test](#)

Testiranje search funkcionalnosti

Opis	Ulagani podatak	Očekivani podatci	Dobiveni podatci
Search funkcionalnosti	"Party"	{ name: "Medium Party Game", difficulty: 2, maxPlayers: 4}	{ name: "Medium Party Game", difficulty: 2, maxPlayers: 4}
Osjetljivosti na velika i mala slova	"PARTY"	{ name: "Medium Party Game", difficulty: 2, maxPlayers: 4}	{ name: "Medium Party Game", difficulty: 2, maxPlayers: 4}
Pretraživanja nepostojeće igre	"Nonexistent"	[]	[]
Bez pretraživanja (prazan search)	""	{ name: "Easy Game", difficulty: 1, maxPlayers: 2}, { name: "Medium Party Game", difficulty: 2, maxPlayers: 4}, { name: "Hard Strategy Game", difficulty: 4, maxPlayers: 6}	{ name: "Easy Game", difficulty: 1, maxPlayers: 2}, { name: "Medium Party Game", difficulty: 2, maxPlayers: 4}, { name: "Hard Strategy Game", difficulty: 4, maxPlayers: 6}

link: [Filter-Search Unit Test](#)

Testiranje sign up funkcionalnosti

Opis	Očekivani podatci	Dobiveni podatci

Opis	Očekivani podaci	Dobiveni podaci
Korisnik već postoji	409, "Username or email already in use"	409, "Username or email already in use"
Uspješna registracija	201, "Registration successful!"	201, "Registration successful!"

link: [Sgin up Unit Test](#)

Testiranje log in funkcionalnosti

Opis	Očekivani podaci	Dobiveni podaci
Nedostaje identifikator ili lozinka	400, "All fields are required"	400, "All fields are required"
Nije pronađen korisnik	401, "User not Found!"	401, "User not Found!"
Lozinka se ne podudara	401, "Wrong Password!"	401, "Wrong Password!"
Uspješna prijava	200, "Login successful!"	200, "Login successful!"

link: [Log in Unit Test](#)

Ispitivanje sustava

Cilj ispitivanja sustava je testiranje ponašanja cijelog sustava u uvjetima stvarnog korištenja, uz posebnu pažnju na međusobnu povezanost svih komponenti. Ispitivanje treba obuhvatiti sve aspekte sustava i njegovu interakciju s korisnicima.

Sljedeći testovi su napravljeni pomoću alata Selenium

Sign up test

Koraci:

1. Otvoriti stranicu
2. Pritisnuti na gumb Log In -> korisnik je preusmjeren na stranicu za log in
3. Pritisnuti na "New to here? Sign Up"
4. Unijeti Email: "selenium.testing @ testing.test"
5. Unijeti Username: "Selenium Testing"
6. Unijeti Password: "test123"
7. Pritisnuti gumb Sign Up -> korisnik je registriran i preusmjeren na početni zaslon

Očekivani rezultat: Korisnik je uspješno registriran i preusmjeren na početnu stranicu.

Dobiveni rezultat: Test je uspješno izvršen, korisnik je registriran i preusmjeren na početnu stranicu.

https://frontend.err808.xyz

	Command	Target	Value
1	✓ open	/	
2	✓ set window size	1536x824	
3	✓ click	linkText=Log In	
4	✓ click	css=.flip-text:nth-child(5)	
5	✓ click	id=signup_email	
6	✓ type	id=signup_email	selenium.testing@testing.test
7	✓ click	id=signup_username	
8	✓ type	id=signup_username	Selenium Testing
9	✓ click	id=signup_password	
10	✓ type	id=signup_password	test123
11	✓ click	id=signup_repeat_password	
12	✓ type	id=signup_repeat_password	test123
13	✓ click	css=.login_buttons:nth-child(10) > .form__submit-button	
14	✓ click	css=.public-browse-link	

slika 6.2.: SginUp_test

Rubni slučaj: Korisnik unese Email ili Username koji već postoji -> iskače prozor koji obavješta korisnika o tome da je Email ili Username već u uporabi, korisnik nije preusmjeren na početnu stranicu.

https://frontend.err808.xyz

	Command	Target	Value
1	✓ open	/	
2	✓ set window size	1536x824	
3	✓ click	linkText=Log In	
4	✓ click	css=.flip-text:nth-child(5)	
5	✓ click	id=signup_email	
6	✓ type	id=signup_email	selenium.testing@testing.test
7	✓ click	id=signup_username	
8	✓ type	id=signup_username	Test
9	✓ click	id=signup_password	
10	✓ type	id=signup_password	test123
11	✓ click	id=signup_repeat_password	
12	✓ type	id=signup_repeat_password	test123
13	✓ click	css=.login_buttons:nth-child(10) > .form__submit-button	
14	X click	css=.public-browse-link	

slika 6.3.: SginUp_test_fail

Log in test

Koraci:

1. Otvoriti stranicu
2. Pritisnuti na gumb Log In -> korisnik je preusmjeren na stranicu za log in
3. Unijeti Username: "Selenium Testing"
4. Unijeti Password: "test123"
5. Pritisnuti gumb Log In -> korisniku iskače prozor s porukom "Login successful!" i gumbom za preusmjeravanje na početni zaslon
6. Pritisnuti gumb "Go To Front Page" -> korisnik je preusmjeren na početni zaslon

Očekivani rezultat: Korisnik je uspješno prijavljen i preusmjeren na početnu stranicu.

Dobiveni rezultat: Test je uspješno izvršen, korisnik je prijavljen i preusmjeren na početnu stranicu.

https://frontend.err808.xyz			
	Command	Target	
		Value	
1	✓ open	/	
2	✓ set window size	1263x778	
3	✓ click	css=.public	
4	✓ click	linkText=Log In	
5	✓ click	id=login_email	
6	✓ type	id=login_email	Selenium Testing
7	✓ click	id=login_password	
8	✓ type	id=login_password	test123
9	✓ click	css=.login_buttons:nth-child(6) > .form__submit-button	
10	✓ click	css=.auth-done-btn	
11	✓ click	css=.public-browse-link	

slika 6.4: Login_test

Rubni slučaj: Korisnik unese pogrešan Email/Username ili Password -> iskače prozor koji obavješta korisnika o tome da je unio pogrešan Email/Username ili Password, korisnik nije preusmjeren na početnu stranicu.

https://frontend.err808.xyz			
	Command	Target	
		Value	
1	✓ open	/	
2	✓ set window size	1263x778	
3	✓ click	css=.public	
4	✓ click	linkText=Log In	
5	✓ click	id=login_email	
6	✓ type	id=login_email	Selenium Testing
7	✓ click	id=login_password	
8	✓ type	id=login_password	wrongPass
9	✓ click	css=.login_buttons:nth-child(6) > .form__submit-button	
10	✓ click	css=.auth-done-btn	
11	X click	css=.public-browse-link	

slika 6.5: Login_test_fail_wrongPass

<https://frontend.err808.xyz>

	Command	Target	Value
1	✓ <code>open</code>	/	
2	✓ <code>set window size</code>	1263x778	
3	✓ <code>click</code>	css=.public	
4	✓ <code>click</code>	linkText=Log In	
5	✓ <code>click</code>	id=login_email	
6	✓ <code>type</code>	id=login_email	wrongName
7	✓ <code>click</code>	id=login_password	
8	✓ <code>type</code>	id=login_password	test123
9	✓ <code>click</code>	css=.login_buttons:nth-child(6) > .form__submit-button	
10	✓ <code>click</code>	css=.auth-done-btn	
11	X <code>click</code>	css=.public-browse-link	

slika 6.6: Login_test_fail_wrongName

NewListing_test

Koraci:

1. Otvoriti stranicu
2. Pritisnuti na gumb Make New Listing -> korisnik je preusmjeren na stranicu za stvaranje novog Listing-a
3. Unos podataka o igri
4. Pritisnuti gumb Submit -> iskače prozor s porukom "Listing added successfully" i gumbom "OK"
5. Pritisnuti gumb OK -> korisnik je preusmjeren na stranicu My Games

Očekivani rezultat: Korisnik je uspješno objavio oglas za igru te je preusmjeren na stranicu My Games.

Dobiveni rezultat: Test je uspješno izvršen, korisnik je objavio oglas za igru te je preusmjeren na stranicu My Games.

<https://frontend.err808.xyz>

	Command	Target	Value
1	✓ open	/	
2	✓ set window size		1263x778
3	✓ click	linkText=Make New Listing	
4	✓ click	name=name	
5	✓ type	name=name	Selenium Test Listing
6	✓ click	name=publisher	
7	✓ type	name=publisher	Selenium
8	✓ click	name=genre	
9	✓ type	name=genre	Test
10	✓ click	name=releaseYear	
11	✓ type	name=releaseYear	2000
12	✓ click	name=condition	
13	✓ select	name=condition	label>New
14	✓ click	css=option:nth-child(2)	
15	✓ click	name=minPlayers	
16	✓ type	name=minPlayers	2
17	✓ click	name=maxPlayers	
18	✓ type	name=maxPlayers	4
19	✓ click	name=playTime	
20	✓ type	name=playTime	5
21	✓ click	name=difficulty	
22	✓ type	name=difficulty	1
23	✓ click	css=.form-group:nth-child(9) > label	
24	✓ click	css=textarea	
25	✓ click	css=.full-width > input	
26	✓ type	css=.full-width > input	C:\Users\Toni\Pictures\test.jpg
27	✓ click	css=textarea	
28	✓ type	css=textarea	...
29	✓ click	css=.primary-button	
30	✓ click	css=.auth-done-btn	

slika 6.7.: NewListing_test

Rubni slučaj: Korisnik pokuša napraviti oglas bez da ima postavljenu lokaciju na profilu, korisnik nije ispunio sve obavezne podatke o igri.

<https://frontend.err808.xyz>

	Command	Target	Value
1	✓ <code>open</code>	/	
2	✓ <code>set window size</code>	1263x778	
3	✓ <code>click</code>	linkText=Make New Listing	
4	X <code>click</code>	name=name	
5	<code>type</code>	name=name	Selenium Test Listing
6	<code>click</code>	name=publisher	
7	<code>type</code>	name=publisher	Selenium
8	<code>click</code>	name=genre	
9	<code>type</code>	name=genre	Test
10	<code>click</code>	name=releaseYear	
11	<code>type</code>	name=releaseYear	2000
12	<code>click</code>	name=condition	
13	<code>select</code>	name=condition	label>New
14	<code>click</code>	css=option:nth-child(2)	
15	<code>click</code>	name=minPlayers	
16	<code>type</code>	name=minPlayers	2
17	<code>click</code>	name=maxPlayers	
18	<code>type</code>	name=maxPlayers	4
19	<code>click</code>	name=playTime	
20	<code>type</code>	name=playTime	5
21	<code>click</code>	name=difficulty	
22	<code>type</code>	name=difficulty	1
23	<code>click</code>	css=.form-group:nth-child(9) > label	
24	<code>click</code>	css=textarea	
25	<code>click</code>	css=.full-width > input	
26	<code>type</code>	css=.full-width > input	C:\Users\Toni\Pictures\test.jpg
27	<code>click</code>	css=textarea	
28	<code>type</code>	css=textarea	...
29	<code>click</code>	css=.primary-button	
30	<code>click</code>	css=.auth-done-btn	

slika 6.8.: `NewListing_test_fail_noProfileLocation`

	Command	Target	Value
1	✓ open	/	
2	✓ set window size	1263x778	
3	✓ click	linkText=Make New Listing	
4	✓ click	name=name	
5	✓ type	name=name	Selenium Test Listing
6	✓ click	name=publisher	
7	✓ click	name=genre	
8	✓ type	name=genre	Test
9	✓ click	name=releaseYear	
10	✓ type	name=releaseYear	2000
11	✓ click	name=condition	
12	✓ click	css=option:nth-child(2)	
13	✓ click	name=minPlayers	
14	✓ type	name=minPlayers	2
15	✓ click	name=maxPlayers	
16	✓ type	name=maxPlayers	4
17	✓ click	name=playTime	
18	✓ type	name=playTime	5
19	✓ click	name=difficulty	
20	✓ type	name=difficulty	1
21	✓ click	css=.form-group:nth-child(9) > label	
22	✓ click	css=textarea	
23	✓ click	css=.full-width > input	
24	✓ type	css=.full-width > input	C:\Users\Toni\Pictures\test.jpg
25	✓ click	css=textarea	
26	✓ type	css=textarea	...
27	✓ click	css=.primary-button	
28	X click	css=.auth-done-btn	

slika 6.9.: NewListing_test_fail_missingDetail

OfferExchange_test

Koraci:

1. Otvoriti stranicu
2. Pritisnuti na gumb Browse All Games -> korisnik je preusmjeren na stranicu sa svim oglasima
3. Pritisnuti na oglas -> korisnik je preusmjeren na stranicu oglasa
4. Pritisnuti na gumb Offer Exchange -> iskače prozor sa listom igara koje korisnik može ponuditi za zamijenu
5. Označi igru kao ponudu za zamijenu
6. Pritisnuti gumb Offer
7. Pritisnuti na gumb My Offers -> korisnik je preusmijeren na stranicu My Offers gdje je vidljiva ponuda za zamijenu

Očekivani rezultat: Korisnik je uspješno napravio trade offer te je preusmjeren na stranicu My Offers**Dobiveni rezultat:** Test je uspješno izvršen, korisnik je napravio trade offer te je preusmjeren na stranicu My Offers.

<https://frontend.err808.xyz>

Command		Target	Value
1	✓ open	/	
2	✓ set window size		1263x778
3	✓ click		linkText=Browse All Games
4	✓ click		css=.game-card:nth-child(4) .game-card-details
5	✓ run script		window.scrollTo(0,0)
6	✓ click		css=.primary-button
7	✓ click		css=input
8	✓ click		css=.trade-actions > .primary-button
9	✓ click		linkText=My Offers
10	✓ run script		window.scrollTo(0,0)

slika 6.10: OfferExchange_test

Rubni slučaj: Korisnik koji nema nijedan oglas za igru pokuša napraviti trade offer.

<https://frontend.err808.xyz>

Command		Target	Value
1	✓ open	/	
2	✓ set window size		1263x778
3	✓ click		linkText=Browse All Games
4	✓ click		css=.game-card:nth-child(4) .game-card-details
5	✓ run script		window.scrollTo(0,0)
6	✓ click		css=.primary-button
7	X click		css=input
8	click		css=.trade-actions > .primary-button
9	click		linkText=My Offers
10	run script		window.scrollTo(0,0)

slika 6.11: OfferExchange_test_fail_noListings# Korištene tehnologije i alati

Redni broj	Tehnologija / Alat	Verzija	Kategorija
1	JavaScript	ES6+	programski jezik
2	HTML	HTML5	programski jezik
3	CSS	CSS3	programski jezik
4	Node.js	20.19.2	radni okvir
5	Express.js	5.1.0	radni okvir
6	React	19.2.0	radni okvir
7	Passport.js	0.7.0	radni okvir
8	Mongoose	8.19.3	radni okvir
9	Nodemailer	7.0.12	radni okvir
10	MongoDB	8.2	baza podataka
11	WebStorm	2025.3	razvojni alat

Redni broj	Technologija / Alat	Verzija	Kategorija
13	Neovim	-	razvojni alat
14	Git	2.52.0	razvojni alat
15	GitHub Actions	-	alat za razmještaj
16	Azure	-	razvojni alat
17	Selenium	-	alat za ispitivanje
18	Jest	-	alat za ispitivanje
19	OpenStreetMap API	0.6	razvojni alat

Programski jezici

JavaScript

JavaScript je programski jezik koji se koristi za razvoj web aplikacija na klijentskoj i poslužiteljskoj strani. U našem projektu JavaScript je osnovni jezik za razvoj frontenda i backenda aplikacije. Korištenje ES6+ standarda omogućuje modernu sintaksu, bolju čitljivost koda i kompatibilnost s bibliotekama i alatima.

HTML

HTML je standardni jezik koji se koristi za definiranje strukture web stranica. U projektu ga koristimo za izradu osnovne strukture korisničkog sučelja. Verzija HTML5 donosi semantičke elemente i bolju podršku za multimedijalne sadržaje.

CSS

CSS je jezik namijenjen opisivanju izgleda i stiliziranju web stranica. U projektu se koristi za definiranje vizualnog identiteta korisničkog sučelja te za osiguravanje responsivnog prikaza na različitim uređajima.

Radni okviri i biblioteke

Node.js

Node.js je JavaScript runtime koji omogućuje izvođenje koda izvan web preglednika i često se koristi za razvoj poslužiteljskog dijela aplikacija. U projektu se koristi verzija 20.19.2, koja pripada LTS grani i donosi stabilnost, sigurnosne nadogradnje te kompatibilnost s modernim backend bibliotekama.

Express.js

Express.js je radni okvir za izradu web aplikacija u Node.js okruženju. U projektu se koristi za implementaciju REST API-ja i obradu HTTP zahtjeva. Verzija 5.1.0 pruža poboljšanu sigurnost i stabilnost.

React

React je radni okvir za izradu korisničkih sučelja temeljen na komponentnom pristupu. U projektu se koristi za razvoj frontenda aplikacije, omogućujući dinamično i responsivno korisničko sučelje. Verzija 19.2.0 osigurava podršku za najnovije značajke i optimizacije.

Passport.js

Passport.js je radni okvir za autentikaciju korisnika. U projektu se koristi za implementaciju sustava prijave i autentikacije korisnika. Njegova modularnost omogućuje jednostavnu integraciju različitih autentikacijskih strategija.

Mongoose

Mongoose je radni okvir (ODM) za rad s MongoDB bazom podataka. Omogućuje definiranje shema i modela te strukturirani pristup podacima. U projektu ga koristimo za upravljanje podacima i validaciju unosa korisnika.

Nodemailer

Nodemailer je radni okvir za slanje elektroničke pošte iz Node.js aplikacija. U projektu se koristi za slanje automatiziranih e-mail poruka, primjerice za potvrdu registracije ili obavijesti korisnicima.

Baza podataka

MongoDB

MongoDB je NoSQL baza podataka koja pohranjuje podatke u obliku dokumenata. U projektu se koristi za pohranu korisničkih podataka i ostalih informacija aplikacije. Verzija 8.2 omogućuje visoku skalabilnost i fleksibilnu strukturu podataka.

Razvojni alati

WebStorm

WebStorm je integrirano razvojno okruženje (IDE) namijenjeno razvoju JavaScript aplikacija. U projektu se koristi za razvoj backenda i frontenda zbog naprednih alata za analizu i pisanje koda.

Visual Studio Code

Visual Studio Code je jednostavan i fleksibilan editor koda koji podržava velik broj programskih jezika i proširenja. U projektu se koristi za razvoj i uređivanje izvornog koda.

Neovim

Neovim je editor koda, dizajniran za brzinu, proširivost i učinkovit rad bez grafičkog sučelja. Omogućuje uređivanje datoteka kroz prečace, skriptiranje i širok ekosustav dodataka. U projektu se koristi kao alternativni razvojni alat za rad u terminalskom okruženju.

Git

Git je distribuirani sustav za kontrolu verzija koji omogućuje praćenje promjena u kodu i timsku suradnju. U projektu se koristi za upravljanje izvornim kodom i verzijama aplikacije.

Azure

Azure je cloud platforma koja se u projektu koristi za razmještaj i hosting aplikacije. Omogućuje pokretanje aplikacije u oblaku bez potrebe za vlastitom infrastrukturom, čime se pojednostavljuje održavanje sustava. Azure pruža visoku dostupnost, mogućnost skaliranja resursa prema opterećenju te centralizirano upravljanje aplikacijskim komponentama.

OpenStreetMap API

OpenStreetMap API omogućuje pristup geolokacijskim podacima. U projektu se koristi za postavljanje i odabir lokacije, čime se omogućuje precizno definiranje lokacije oglasa unutar aplikacije.

Alati za ispitivanje

Selenium

Selenium je alat za automatizirano testiranje web aplikacija. U našem projektu se koristi za testiranje korisničkog sučelja simulacijom interakcije korisnika s aplikacijom.

Jest

Jest je alat za jedinično testiranje JavaScript aplikacija. U projektu se koristi za provjeru ispravnosti funkcionalnosti pojedinih dijelova koda, čime se osigurava stabilnost i pouzdanost aplikacije.

Alati za razmještaj

GitHub Actions

GitHub Actions je alat za automatizaciju procesa razvoja softvera. U ovom projektu koristi se za razmještaj aplikacije, odnosno za automatizirano postavljanje aplikacije na proizvodjsko okruženje nakon promjena u repozitoriju. Time se osigurava dosljedan i ponovljiv proces razmještaja.

Dodatni alati

Za komunikaciju i sastanke smo koristili:

1. WhatsApp
2. Discord

- **Preduvjeti:**

1. Node.js 20
2. npm [LATEST]

- **Preuzimanje:**

```
$ git clone https://github.com/tim-error808/error808.git
```

Instalacija ovisnosti:

```
$ cd error808
$ cd backend
$ npm install
$ cd ../frontend
$ npm install
```

2. Postavke

Detaljne upute za konfiguraciju aplikacije:

- **Konfiguracijske datoteke:**

1. frontend

Potrebno je postaviti adresu backend API. Konfiguracijska datoteka se nalazi u frontend/src/config/ModeConfig.js

2. backend

Potrebno je postaviti environment varijable:

```
FRONTEND_URL
GOOGLE_CLIENT_ID - oauth
GOOGLE_CLIENT_SECRET - oauth
JWT_SECRET
LOCAL_TEST
MONGODB_URI
REFRESH_SECRET
REST_API_PORT
EMAIL_SENDER - "from@someone.domain"
EMAIL_SMTP_HOST
EMAIL_PASSWORD
```

Kontekst pojedine varijable je jasan iz naziva varijabli. Ovisno o okruženju postavljanje varijabli se razlikuje, stoga je na korisniku da poznaje alate svoga okruženja.

- **Postavke baze podataka:**

Baza podataka se postavlja na temelju modela koji se nalaze u backend/models direktoriju i na wiki stranicama.

3. Pokretanje aplikacije

```
$ cd backend
$ npm run dev-start
$ cd ../frontend
$ npm start
```

4. Upute za administratore

Smjernice za administratore aplikacije nakon puštanja u pogon:

- **Pristup administratorskom sučelju:**

- URL/admin
- Administrator treba biti postavljen ili dodijeljen od strane drugog administratora

- **Redovito održavanje:**

- Redovito pregledati stanje web aplikacije na servisu koji se koristi za hosting.

5. Puštanje u pogon na Azure platformi

Azure je platforma koja je korištena za ovaj projekt kako je dostupna studentima za razvoj projekata bez dodatnih troškova.

- Za frontend koristi se Static Web App

1. Na portalu Azure usluge odabrati izradu Static Web App
2. Povezati s github repozitorijem projekta, tj folderom frontend
3. Azure automatski postavlja skriptu za deploy frontend projekta
4. Postaviti simboličku adresu po uputama na platformi
5. Postaviti potrebne environment varijable na sučelju kreirane Static Web App usluge

- Za backend koristi se App Service

1. Na portalu Azure usluge odabrati izradu App Service
2. Povezati s github repozitorijem projekta, tj folderom backend
3. Azure automatski postavlja skriptu za deploy backend projekta
4. Postaviti simboličku adresu po uputama na platformi
5. Postaviti potrebne environment varijable na sučelju kreirane App Service usluge
6. Postaviti CORS postavke na sučelju kreirane App Service usluge

Aplikaciji se pristupa pomoću postavljene simboličke adrese.

Osvrt na vrijeme izrade projektnog zadatka

Izrada projektnog zadatka trajala je oko 12 tjdana (nakon što se oduzme vrijeme u kojem se svaki član tima posvetio drugim obavezama).

Izrada je počela u prvoj polovici listopada 2025., a završila je krajem siječnja 2026. godine.

Tijekom tog vremena pokušalo se što bolje koristiti uputama za izvođenje projekta na koje nas je usmjeravao asistent, kako bi rad na projektu prošao bez velikih prepreka, odgađanja i nagomilavanja zadatka prije važnih rokova.

Najvažniji rokovi s kojima smo se susretali na projektu bili su:

- prezentacija osnovne organizacije i plana projekta (21.10.2025.)
- predaja projekta na GitHub za prvi ciklus (14.11.2025.)
- demonstracija alfa verzije aplikacije (13.01.2026.)
- predaja projekta na GitHub za drugi ciklus (23.01.2026.)

Najveći dio vremena rada na projektu bio je posvećen uskladišvanju članova tima, razradi funkcijskih i nefunkcijskih zahtjeva, ali i samoj implementaciji.

Prepoznati tehnički izazovi i njihova rješenja

Primjeri izazova:

- arhitektura sustava - kako odrediti optimalnu arhitekturu na kojoj će se temeljiti aplikacija koja ispunjava sve zahtjeve?
- rad s bazom podataka - kako izraditi bazu podataka u novom sustavu s kojim članovi tima još nemaju iskustva i kako tu bazu povezati s ostalim komponentama?
- autentifikacija korisnika - kako svelatiti i implementirati OAuth 2.0 sustav s kojim članovi tima nemaju iskustva?
- integracija vanjskih sustava - na koji način iskoristiti vanjski sustav za optimalnu integraciju s aplikacijom?
- sinkronizacija rada u timu - na koji način uskladiti rad svih članova tima tako da svatko dobije zadatku s kojim se može nositi, a da pritom doprinese maksimalno radu tima?

Rješenja izazova:

- problem arhitekture je riješen na način da je tim organizirao sastanak na kojem se prošlo kroz različite organizacije sustava kako bi se utvrdilo koji od njih najbolje odgovara zahtjevima projekta
- izazov stvaranja i integracije baze podataka je riješen tako da su članovi tima koji su bili zaduženi za njeno funkcioniranje prošli kroz dokumentaciju kako bi shvatili na koji način sustav radi i iskoristili ga za konstruiranje baze koja zadovoljava zahtjeve projekta
- autentifikaciju korisnika preuzeli su članovi tima koji imaju najveće iskustvo potrebitno za njeno funkcioniranje, ali i oni su trebali proći kroz dokumentaciju kako bi se upoznali s novim sustavom
- integracija vanjskih sustava je riješena tako da su članovi tima pročitali službene upute za njenu integraciju i primjenili znanje na njeno implementiranje u sustav
- sinkronizacija rada članova tima je riješena kontinuiranom komunikacijom, sastancima i kvalitetnom podjelom zadatka

Stečena znanja

Kratak opis znanja i vještina koje smo stekli radom na projektu:

Timski rad

Tijekom razvoja projekta stekli smo iskustvo rada u timu i vidjeli kako otrpilike izgleda programsko okruženje koje zajedničkim snagama razvija neki sustav. Stekli smo iskustvo kako je to kombinirati različita znanja s kojima dolaze različiti članovi tima i uvidjeli da programiranje nije samo pisanje programskog koda nego sveobuhvatan proces razvoja programske potpore koji uključuje stalnu komunikaciju, revizije, dokumentaciju, ispravljanje grešaka...

Planiranje

Stekli smo uvid u to o kojim stvarima se treba pobrinuti tijekom razvoja programske potpore i riješiti ih na vrijeme. Primjerice, naučili smo kako stvari ne treba rješavati u zadnji tren jer neочекivani problemi znaju iskrsnuti. Isto tako naučili smo kako treba pomno pratiti sve rokove i kod složenijih problema voditi uredne bilješke o rokovima i radu. Iskusili smo kako je to prilagođavati se radu ostalih članova tima imajući na umu da je ključna stvar razviti kvalitetan projekt.

Debugiranje

Naučili smo kako će kod razvoja programske potpore često trebati zastati, vratiti se unatrag i ispraviti pogrešku u kodu za koji smo vjerovali da je u redu. Na prvu se može činiti kao da sporo čitanje i testiranje dijela sustava koji ne radi ispravno usporava rad na projektu, ali zapravo treba razumjeti da se takve stvari treba očekivati i izdvojiti unaprijed vremena za nošenje s njima.

Dokumentacija

Shvatili smo kako dokumentacija nije samo onaj dio projekta koji se treba napraviti da se zadovolji forma, nego vrlo često služi za osvrt napravljenog članovima tima, ali i svima onima koji bi mogli biti zainteresirani za detalje implementacije i razvoja sustava.

Nove tehnologije

Radom na projektu morali smo se prilagoditi novim sustavima i tehnologijama, kao što su MS Azure, MongoDB, OAuth 2.0, Postman... i time ne samo da smo omogućili potrebnu funkcionalnost aplikacije, već i stekli znanje koje ostaje i za vrijeme nakon projekta.

Znanja potrebna za bržu i kvalitetniju izradu projekta

Za bržu i kvalitetniju izradu projekta trebala bi nam šira znanja i unaprijed poznavanje određenih sustava jer tada ne bismo trebali čitati opsežnu dokumentaciju kako bismo razumijeli kako sustav radi. Osim toga bila bi nam potrebna šira znanja o lakšoj i bržoj komunikaciji i integraciji s drugim članovima tima kako bi zajednički rad tekao što kvalitetnije.

Perspektive za nastavak rada u projektnoj grupi

Projekt ima potencijala za daljnji razvoj. Članovi tima ostvarili su dobru suradnju i s lakoćom bi mogli nastaviti surađivati na projektu i nadograđivati ga različitim funkcionalnostima te ga usavršiti kako bi bio potpuno prilagođen za javnu uporabu. Primjeri funkcionalnosti koji bi se mogli dodati u aplikaciju: omogućavanje komunikacije korisnika izravno unutar aplikacije, zamjena ne samo društvenih igara, već i ostalih povezanih komponenti, kao što su figurice, kockice i ostalo, povezivanje korisnika u grupe sličnih interesa i/ili sličnih lokacija kako bi mogli nastaviti suradnju i komunikaciju uživo...

Ograničenja

Ograničenja s kojima se susretalo tijekom razvoja su finansijske i vremenske prirode. Mnoge već napravljene stvari smo mogli poboljšati da smo imali više vremena za razvoj, ali osim toga, mnogli smo dodati i neke dodatne, napredne funkcionalnosti. Što se financija tiče, zbog ograničenog budžeta koristili smo besplatne ili jeftine verzije nekih sustava i time ograničili funkcionalnosti našeg sustava (primjerice, prostor u bazi podataka).

Funkcionalnosti koje nisu implementirane

Naš sustav pokrio je sve funkcionalnosti opisane zahtjevima projekta, međutim postoje područja za daljnji razvoj, kao što je poboljšanje performansa, dodavanje novih funkcionalnosti za korisnika i slično.1. Programsko inženjerstvo, FER ZEMRIS, <http://www.fer.hr/predmet/proinz>

2. The Original "Buy, Sell & Trade" Boardgame Group – Facebook. Dostupno: <https://www.facebook.com/groups/boardgameexchange/> (20. listopad 2025.).
3. BoardGameGeek | Gaming Unplugged Since 2000. Dostupno: <https://boardgamegeek.com/> (20. listopad 2025.).
4. BoardGamesSwap – Buy, Sell and Trade Board Games. Dostupno: <https://www.boardgamesswap.com/> 20. listopad 2025.).
5. MongoDB Documentation. Dostupno: <https://www.mongodb.com/docs/> (7. studeni 2025.).
6. React – Learn React. Dostupno: <https://react.dev/learn> (7. studeni 2025.).
7. Node.js – Introduction to Node.js. Dostupno: <https://nodejs.org/en/learn/getting-started/introduction-to-nodejs> (7. studeni 2025.).
8. Client-Server Architecture – System Design | GeeksforGeeks. Dostupno: <https://www.geeksforgeeks.org/system-design/client-server-architecture-system-design/> (10. studeni 2025.).
9. MVC Architecture – System Design | GeeksforGeeks. Dostupno: <https://www.geeksforgeeks.org/system-design/mvc-architecture-system-design/> (10. studeni 2025.).
10. Frontend vs Back-end Development | GeeksforGeeks. Dostupno: <https://www.geeksforgeeks.org/blogs/frontend-vs-backend/> (10. studeni 2025.).

Rev.	Opis promjene/dodataka	Autori	Datum
0.1	Napravljen home page	Niko Knežević	20.10.2025.
0.2	Uređen home page	Niko Knežević	20.10.2025.
0.3	Kloniran predložak dokumentacije	Niko Knežević	20.10.2025.
1.1	Popunjena stranica dokumentacije	Niko Knežević	20.10.2025.
2.1	Popunjen dio funkcijskih i nefunkcijskih zahtjeva na stranici	Niko Knežević	20.10.2025.
B.1	Dodane informacije o sastancima	Niko Knežević	20.10.2025.
2.2	Dovršena stranica dokumentacije	Niko Knežević	2.11.2025.
B.2	Dodane informacije o sastancima	Niko Knežević	2.11.2025.
B.3	Dodane informacije o sastancima	Niko Knežević	6.11.2025.
B.4	Napravljena i ažurirana tablica aktivnosti	Niko Knežević	6.11.2025.
B.5	Napravljena i ažurirana tablica plana rada	Niko Knežević	6.11.2025.
2.3	Ažurirani podatci o dionicima	Niko Knežević	6.11.2025.
3.1	Izrađena tablica obrazaca uporabe	Niko Knežević	6.11.2025.
3.2	Ispisani detaljni podatci o svakom obrascu uporabe	Niko Knežević	6.11.2025.
4.1	Ispisana većina stvari o arhitekturi sustava i korištenim tehnologijama	Niko Knežević	7.11.2025.
3.3	Napravljeni dijagrami obrazaca uporabe	Frane Ćovid, Niko Knežević	8.11.2025.
3.4	Napravljen prvi sekvencijski dijagram	Frane Ćovid, Niko Knežević	8.11.2025.
3.5	Uređeni i popravljeni dijagrami obrazaca uporabe	Niko Knežević	9.11.2025.
3.6	Uređeni i popravljeni opisi obrazaca uporabe i njihove veze s funkcijskim zahtjevima	Niko Knežević	9.11.2025.
2.4	Nadodani i ispravljeni funkcijski zahtjevi	Niko Knežević	9.11.2025.
4.2	Opisana arhitektura baze podataka	Frane Ćovid	9.11.2025.
4.3	Opisan ostatak arhitekture	Niko Knežević	9.11.2025.
README.1	Napisan je predložak za README	Niko Knežević	9.11.2025.
A.1	Postavljena literatura korištена u izradi dokumentacije	Niko Knežević	10.11.2025.
3.7	Napravljen ostatak sekvencijskih dijagrama	Niko Knežević, Ivan Žalac	10.11.2025.

B.6 Rev.	Opisani problemi rada na projektu Opis promjene/dodataka	Niko Knežević Autori	Datum
4.4	Izrađen dijagram stanja	Niko Knežević	09.01.2026.
4.5	Izrađen dijagram aktivnosti	Niko Knežević	10.01.2026.
5.1	Izrađen dijagram komponenata i dijagram razmještaja	Frane Ćevid	11.01.2026.
8.1	Izrađen glavni dio uputa za puštanje u pogon	Ivan Žalac	12.01.2026.
6.1	Izrađen dio wiki dokumentacije o testovima	Toni Kapučija	12.01.2026
B.6	Ažuriranje aktivnosti i postavljanje grafova	Niko Knežević	23.01.2026.
6.2	Ažuriranje wiki dokumentacije o testovima	Toni Kapučija	23.01.2026.
4.6	Dodan dijagram razreda	Marko Bošnjak	23.01.2026.

Napomena : u tablici nisu navedene promjene dokumentacije nasatale u ovom poglavju # Dnevnik sastajanja

1. sastanak

- Datum: 9. listopada 2025.
- Prisustvovali: Marko Bošnjak, Marin Čikotić, Frane Ćevid, Toni Kapučija, Niko Knežević, Mihael Rošić
- Teme sastanka:
 - konačna odluka o temi projekta
 - dodjela uloge svakom članu tima
 - stvaranje GitHub repozitorija

2. sastanak

- Datum: 18. listopada 2025.
- Prisustvovali: Marko Bošnjak, Marin Čikotić, Frane Ćevid, Toni Kapučija, Niko Knežević, Mihael Rošić, Ivan Žalac
- Teme sastanka:
 - odluka o korištenim tehnologijama
 - detaljniji opis uloga i prvih zadatka
 - stvaranje PowerPoint prezentacije za predstavljanje projekta

3. sastanak

- Datum: 26. listopada 2025.
- Prisustvovali: Niko Knežević, Ivan Žalac
- Teme sastanka:
 - detaljno raspisivanje svih zahtjeva i funkcionalnosti koje treba pokriti projekt
 - dogovor oko daljnjih aktivnosti s ostatkom tima
 - dogovor oko načina daljnjih doprinosa na projektu

4. sastanak

- Datum: 1. studenog 2025.
- Prisustvovali: Marko Bošnjak, Marin Čikotić, Frane Ćevid, Toni Kapučija, Niko Knežević, Mihael Rošić, Ivan Žalac
- Teme sastanka:
 - osvrt na do sad napravljene stvari
 - usporedba napravljenog s očekivanim ciljevima
 - razgovor o detaljima implementacije
 - podjela poslova među članovima grupe

5. sastanak

- Datum: 2. studenog 2025.
- Prisustvovali: Mihael Rošić, Ivan Žalac
- Teme sastanka:
 - dogovor oko izgleda korisničkog sučelja
 - raspisivanje prvih zahtjeva

6. sastanak

- Datum: 6. studenog 2025.
- Prisustvovali: Marko Bošnjak, Marin Čikotić, Mihael Rošić
- Teme sastanka:
 - detaljniji dogovor oko dodijeljenih zadataka i njihova podjela
 - pisanje dijela dokumentacije o arhitekturi
 - napisani issues potrebeni za prvu predaju projekta
 - rad na backendu

7. sastanak

- Datum: 8. studenog 2025.
- Prisustvovali: Frane Ćevid, Niko Knežević
- Teme sastanka:
 - izrada dijagrama obrazaca uporabe
 - izrada jednog sekvencijskog dijagrama
 - dogovor oko daljnjih aktivnosti

8. sastanak

- Datum: 10. studenog 2025.
- Prisustvovali: Marko Bošnjak, Toni Kapučija, Mihael Rošić
- Teme sastanka:
 - povezivanje dijelova sustava
 - testiranje funkcionalnosti
 - priprema pitanja za termin laboratorijske vježbe

9. sastanak

- Datum: 14. studenog 2025.
- Prisustvovali: Marko Bošnjak, Toni Kapučija
- Teme sastanka:
 - testiranje implementacije
 - popravljanje korištenja OAtuh 2.0-a

10. sastanak

- Datum: 10. prosinca 2025.
- Prisustvovali: Marko Bošnjak, Marin Čikotić, Toni Kapučija, Niko Knežević, Mihael Rošić
- Teme sastanka:
 - osvrt na odradene segmente
 - planiranje daljnog rada
 - podjela zadataka drugog ciklusa članovima tima

Plan rada

Zadatak	Rok	Zaduženi	Status
Tema projekta	10.10.2025.	Cijeli tim	Odrađeno
Dijagrami obrazaca uporabe	9.11.2025.	Niko Knežević, Frane Ćevid, Ivan Žalac	Odrađeno
Nabranje funkcionalnih zahtjeva	28.10.2025.	Niko Knežević	Odrađeno
Planiranje dizajna	28.10.2025.	Mihael Rošić	Odrađeno
Sekvencijski dijagrami	9.11.2025.	Marko Bošnjak, Niko Knežević, Frane Ćevid, Ivan Žalac	Odrađeno
Izrada osnove baze podataka sustava	9.11.2025.	Frane Ćevid	Odrađeno
Priprema za prezentiranje napretka	11.11.2025.	Cijeli tim	Odrađeno
Izrada početne stranice	9.11.2025.	Mihael Rošić	Odrađeno
Spajanje s bazom podataka	14.11.2025.	Marko Bošnjak, Frane Ćevid	Odrađeno
Razvoj registracije korisnika	14.11.2025.	Marko Bošnjak	Odrađeno
Testiranje osnovne verzije sustava	14.11.2025.	Toni Kapučija	Odrađeno
Postavljanje aplikacije na poslužitelj	11.11.2025.	Ivan Žalac, Marko Bošnjak	Odrađeno
Uspostava sustava autentifikacije	14.11.2025.	Marko Bošnjak, Marin Čikotić	Odrađeno
Kreiranje sustava za objave društvenih igara	nedefinirano	Marko Bošnjak, Marin Čikotić	Odrađeno
Funkcionalnost filtera igara	14.11.2025.	Marko Bošnjak, Toni Čikotić	Odrađeno
Moderatorski sustav	13.01.2026.	Marko Bošnjak, Marin Čikotić	Odrađeno
Testiranje alfa verzije aplikacije	13.01.2026.	Cijeli tim	Odrađeno
Implementacija osnovnih funkcionalnosti alfa verzije	13.01.2026.	Cijeli tim	Odrađeno
Implementacija sustava za pretragu objava po filterima	13.01.2026.	Marko Bošnjak	Odrađeno
Testiranje dodatnih funkcionalnosti	13.01.2026.	Toni Kapučija	Odrađeno
Testiranje finalne verzije aplikacije	23.01.2026.	Cijeli tim	Odrađeno

Tablica aktivnosti

aktivnosti	Marko Bošnjak	Marin Čikotić	Frane Ćevid	Toni Kapučija	Niko Knežević	Mihael Rošić	Ivan Žalac
Upravljanje projektom	5	5	5	5	5	5	5
Opis projektnog zadatka	3	3	3	3	3	3	3
Funkcionalni zahtjevi	1	1	1	1	2	1	5
Dijagram obrazaca	0	0	3	0	3	0	1
Sekvencijski dijagrami	0	0	1	0	2	0	2
Opis ostalih zahtjeva	0	0	0	0	1	0	2
Opis projekta	3	3	3	3	3	3	3
Arhitektura i dizajn sustava	18	3	3	3	0	12	7
Baza podataka	2	1	6	0	0	0	0
Opis baze podataka	0	0	4	0	0	0	0
Dijagram razreda	0	0	0	0	0	0	0
Dijagram stanja	0	0	0	0	2	0	0
Dijagram aktivnosti	0	0	0	0	2	0	0
Dijagram komponenti	0	0	2	0	0	0	0
Korištene tehnologije i alati	4	5	4	4	0	4	5
Ispitivanje programskog rješenja	3	2	0	10	0	0	2
Dijagram razmještaja	0	0	2	0	0	0	0
Upute za puštanje u pogon	2	0	0	0	0	0	6
Dnevnik sastajanja	0	0	0	0	3	0	0
Zaključak i budući rad	0	0	0	0	2	0	0
Popis literature	0	0	0	0	1	0	0
Izrada aplikacije	15	5	5	5	0	15	5
Izrada baze podataka	0	0	5	0	0	0	0
Spajanje s bazom podataka	3	0	3	0	0	3	0
Dokumentiranje plana projekta i uključenosti	0	0	0	0	2	0	0
Dokumentacija specifikacije obrazaca uporabe	0	0	0	0	5	0	0

Dijagram pregleda promjena

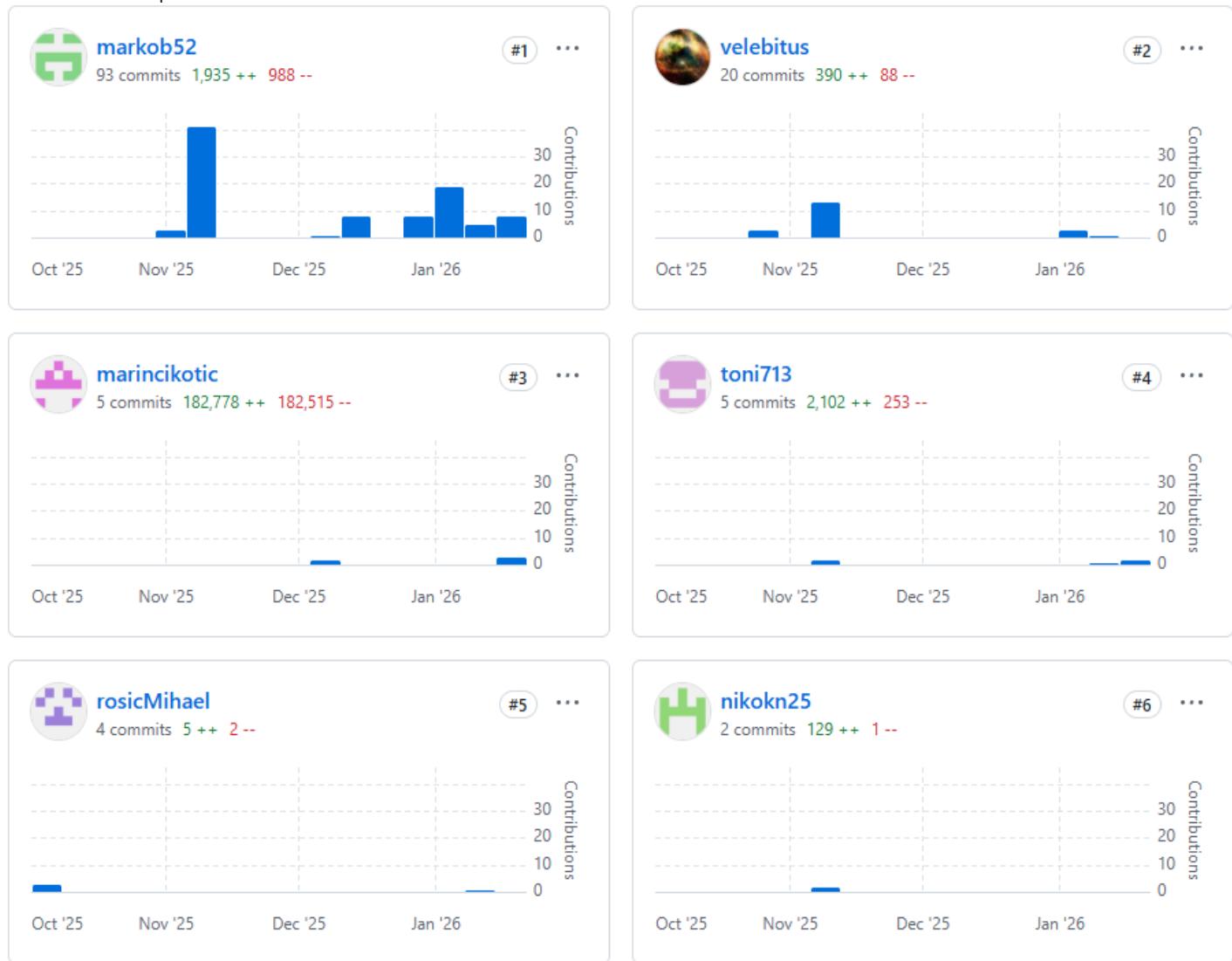
Sljka B.1 Commitovi tijekom vremena

Commits over time

Weekly from Oct 5, 2025 to Jan 18, 2026



Slika B.2 Commitovi po članu tima



Kjučni izazovi i rješenja

Tijekom rada na projektu pojavili su se neki problemi od kojih su neki bili očekivani, a neki pomalo neočekivani. Ovo je opis najznačajnijih problema kod kojih se naišlo tijekom razvoja projekta i načina na koji su riješeni:

Rokovi

Opis: najveći problem koji se pojavljivao tijekom razvoja bio je praćenje rokova i sustizanje potrebnih zadataka na vrijeme.

Primjeri: razvoj potrebnih funkcionalnosti do prve predaje projekta i razvoj funkcionalnosti te testiranje sustava do roka za predstavljanje alfa verzije aplikacije.

Rješenja: pokušaj što kvalitetnije podjele zadataka među članovima tima kako bi se sve dovršilo na vrijeme tako da svatko preuzeme odgovornost za svoju ulogu

Pronalazak vremena za rad

Opis: svi članovi tima imaju obaveza nevezanih za projekt i ponekad je bilo teško pronaći dovoljno vremena za razvoj aplikacije.

Primjeri: balansiranje uloženog vremena rada na projektu s učenjem, rješavanjem laboratorijskih vježbi i pisanjem drugih projekata (projekt R i slično).

Rješenja: kontinuirana komunikacija sa svim članovima tima kako bi se znalo koliki se napredak može očekivati od pojedinca u određenom vremenskom periodu.

Greške u odraćenom poslu

Opis: prilikom bilo kakvog rada znaju se dogoditi neočekivane greške koje zahtijevaju ispravak nečeg već napravljenog.

Primjer: greška prilikom spajanja baze podataka s backendom.

Rješenje: ponovni rad na stvari koja je izazvala problem i detaljan pregled greške kako bi se ustanovilo što ju uzrokuje.

Razumijevanje zahtjeva

Opis: tijekom rada pojavljivali se se problemi razmijevanja zahtjeva aplikacije, što uzrokuje zastoj u radu jer se ne može nastaviti dok se ne razjasni što se očekuje od pojedinog segmenta sustava.

Primjer: nedovoljno razumljiv opis načina na koji treba raditi sustav geolociranja.

Rješenje: kontaktacije s asistentom uživo ili putem maila s unaprijed pripremljenim pitanjima svih članova grupe, ali samo kad se utvrdi da niti jedan član tima ne razumije zahtjeve.

Programsko inženjerstvo ak.god 2025./2026.

Sveučilište u Zagrebu

Fakultet elektrotehnike i računarstva

PlayTrade

Tim: TG04.3

Ime tima: error808

Nastavnik: Vlado Sruk

Članovi tima:

- Marko Bošnjak - Backend
- Marin Čikotić - Fullstack
- Frane Ćevid - Baze podataka
- Toni Kapučija - Testiranje
- Niko Knežević - Voditelj
- Mihail Rošić - Frontend
- Ivan Žalac - DevOps

Poveznica na GitHub projekta:

<https://github.com/tim-error808/error808>

PlayTrade je web aplikacija osmišljena za sve ljubitelje društvenih igara kako bi mogli iskusiti veći opseg različitih igara.

Mnogi ljudi koji uživaju u društvenim igrama često kupe igru koja im se svidi, ali nakon vremena mnoge postanu monotone. Naime, vrlo često se dogodi kod društvenih igara da njihovi igrači nauče komponente igre kroz iskustvo i nakon nekog vremena igra postaje predvidljiva, stoji na polici i skuplja prašinu. Tu bi od velike koristi moglo biti zamjeniti igru koju vlasnik više neće koristiti za neku novu, jer ne samo da se neće povećavati broj igara u kolekciji, nego se i neće morati trošiti novci na nove. Stoga PlayTrade omogućava korisnicima upravo to.

Aplikacija omogućuje svim korisnicima da pregledavaju objave igara koje su dostupne za zamjenu i traže neku koju imaju na umu, a onim registriranim omogućuje da objave svoje igre koje žele zamjeniti i da vrše zamjene s drugima. Na taj način svaki korisnik doprinosi ukupnoj bazi igara i povećava mrežu, a time i mogućnost da svatko pronađe neku novu, uzbudljivu igru.

Potencijalna korist ovog projekta

Aplikacija PlayTrade ima više potencijalnih koristi za igrače diljem svijeta, ali i okoliš:

Umanjuje troškove na društvene igre: mnogi ljudi bi voljeli probati različite društvene igre, ali često odustanu od toga iz finansijskih razloga. Međutim, na PlayTrade-u korisnici mogu koristiti svoju malu bazu igara i mijenjati ih za koju god žele bez dodatnih troškova.

Pomaže proširenju iskustva: entuzijasti vole iskusiti različite društvene igre - različitim zahtjevnostima, profila i s različitim brojem igrača, ali iz različitih razloga to nije uvek lako. Primjerice, moguće je da se određena igra više ne proizvodi ili prodaje, a isto tako je moguće da više ne postoji mjesto za novu igru. Uz pomoć PlayTrade-a

korisnici mogu relativno lako pronaći željenu igru i zamijeniti se za nju, te time osloboditi mjesto na polici i iskusiti nešto novo.

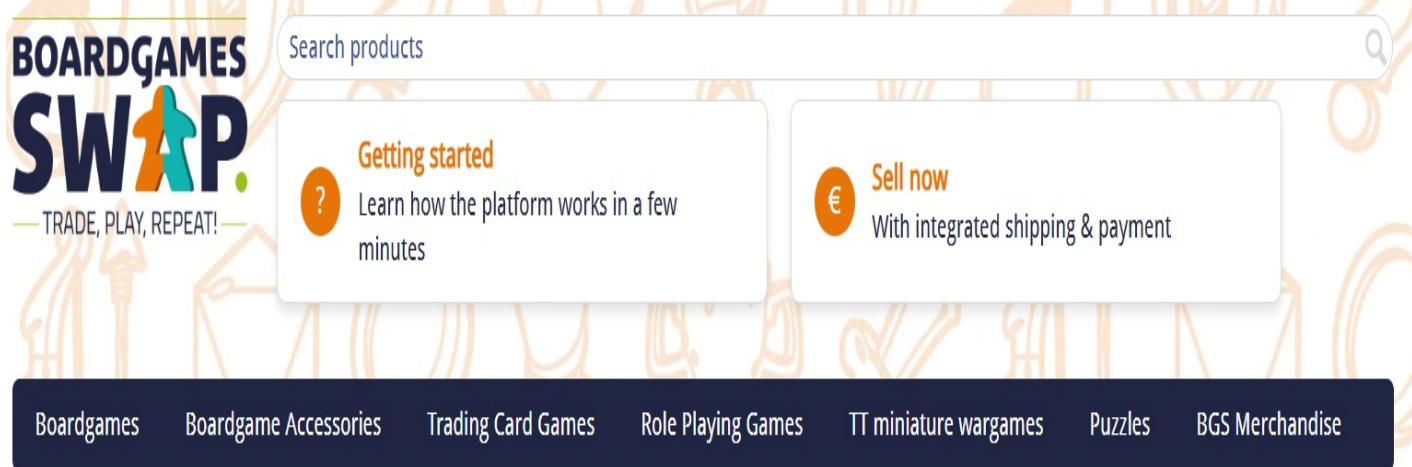
Pomaže povezivanju ljudi: Putem aplikacije korisnici se mogu susresti s ljudima koji imaju slične interese kao i oni i na taj način povećati mrežu svojih prijateljstava, a možda i pronaći novog sugrača.

Pomaže u očuvanju okoliša i štednji energije i materijala: Zamjenom jedne društvene igre za drugu između dva korisnika umanjuje se potreba da ljudi kupu igru koju žele. Samim time stare igre se recikliraju umjesto da se kupuju nove i na taj način se smanjuje otpad koji nastaje jednom kad završi životni vijek igre, ali se i uštedi na energiji i materijalima koji bi bili potrebni u proizvodnim procesima.

Postojeća slična rješenja

BoardGamesSwap

Slika 1.1 Korisničko sučelje stranice BoardGamesSwap



Rješenje slično našem je stranica naziva **BoardGamesSwap**.

Na njoj korisnici mogu objaviti svoju društvenu igru s cijenom i kupiti nečiju drugu društvenu igru. Uz pomoć BoardGamesSwap neregistrirani korisnici mogu kupiti nečiju igru, a registrirani mogu postaviti i svoju igru na prodaju.

Korisnici mogu tražiti neku igru direktno ili listati sve objave iz neke kategorije direktno (Boardgames, Boardgame Accessories, Trading Card Games, TT miniature wargames, Puzzles, BGS Merchandise). Odnosno moguće je kupiti i druge stvari vezane uz igre, a ne igre isključivo.

Korisnici mogu filtrirati ponude po različitim kategorijama: jezik, stanje, godina izdanja, starost, minimalna dob, broj igrača, trajanje igre...

Naš sustav dijeli neke sličnosti i razlike. Za početak, naš sustav nije platforma za prodaju igara već se samo vrše direktnе zamjene. Također PlayTrade se fokusira na društvene igre, odnosno ne uključuje stvari kao što su dodaci za igre.

Na PlayTrade-u se igre mogu filtrirati po sličnim kategorijama: očuvanost, broj igrača, zahtjevnost, izdavač, godina izdanja...

Facebook i Reddit grupe

Slika 1.2 Primjer jedne Facebook grupe namijenjene razmjeni društvenih igara

Board Game Exchange - The Original “Buy, Sell & Trade” Boardgame Group

🔒 Private group · 57.1K members

 Join group



About Discussion



About this group

Welcome to the original board game exchange; "buy, sell, and trade" group! We are the most active community on Facebook dedicated to the buying,... [See more](#)



Private

Only members can see who's in the group and what they post.



Visible

Anyone can find this group.



History

Group created on September 7, 2015. Name last changed on April 30, 2020.

[See more](#)



Tags

Board Games

Na Facebooku i Redditu su osnovane različite grupe za razmjenu društvenih igara i na njima je osim zamjene moguće vršiti i prodaju. Za razliku od toga, na PlayTrade-u se vrši isključivo zamjena. Na Facebooku i Redditu ne postoje tako organizirani filteri kojima ljudi mogu naći točno ono što žele, kao na PlayTrade-u.

BoardGameGeek

Slika 1.3 Guide to BoardGameGeek

THE HOTNESS

GAMES ▾


Miskatonic Tales: Journey to...
2025
Speakeasy
2026
The Old King's Crown
2025
Winnie the Pooh: Serious...
2026
Ikusa
1986
The Lord of the Rings: Fate...
2025
Covenant
2025
Ortoj: The Prague...
2025
The Druids of Edora
2025
SETI: Search for...
2024
Ayar: Children of the Sun
2025Search: Titles Only: Go[Index](#) | [All](#) | [Recent](#) | [Guidelines](#)[Article](#) [Edit](#) | [History](#) | [Editors](#)Guide To BoardGameGeek 

This page is for board games. For RPGs, see [RPGGeek](#). For video games, see [VideoGameGeek](#).

[BoardGameGeek](#) is a website dedicated to physical board games, with an extensive database of more than 120,000 board games (as of October 2020) as well as an active community of users who discuss, argue, buy, sell, trade and play board games. Each game has its own [game entry](#) with information about a game, user ratings, forums for discussion, and a great deal more.

The [forums](#) are a good place to explore to involve yourself in the community. If you are completely new to the site, check out the [Getting Started](#) page and the [Welcome to BoardGameGeek](#) page. To start exploring the wiki, have a look at the [Wiki Index](#), and if you want to learn more about what the wiki is and how to add content to it, go to [About the BoardGameGeek Wiki](#).

Major Content Areas of BGG

- [Front Page](#)
- [Dashboard](#) - A customizable compilation of what's new on BGG
- [Games](#) - Description of the sections on a game page
- [My Geek](#) - Sections found on your personal page
- [Forums](#) - Info and FAQ about the forums, plus a list of Forums
- [Blogs](#) - Info about creating and contributing to a Blog
- [GeekLists](#) - What GeekLists are and options for their use
- [Shopping](#) - Place to trade and buy games
- [Misc](#) - Guilds, Stats, Cameras, etc. - Everything under the Misc menu above
- [Wiki Index](#) - Top level index of this wiki

Table of Contents

- [Major Content Areas of BGG](#)
- [Contribute to the BGG Database](#)
- [Buying, Selling and Trading Games](#)
- [More Ways to Interact with BGG](#)
- [Become a financial supporter](#)
- [Text Editors at BGG](#)

Contribute to the BGG Database

- [Content Rules](#)
- [Submit game listings](#)

[BoardGameGeek](#) je vrlo popularna stranica među ljubiteljima društvenih igara. Ondje je za izuzetno velik broj igara moguće pronaći informaciju o tome na kojoj se platformi i po kolikoj cijeni mogu kupiti društvene igre, moguće je vršiti zamjene igara, moguće je prodavati igre, moguće se upoznavati sa zajednicom, sudjelovati u forumima i mnogo drugih stvari.

S druge strane, PlayTrade nije toliko širok, ali je specijaliziran za zamjenu igara s drugim korisnicima kako bi to iskustvo bilo što efikasnije i ugodnije.

Skup korisnika zainteresiran za ostvareno rješenje

Skup korisnika koji bi mogao biti zainteresiran za aplikaciju PlayTrade uključuje svakog tko ima ikakav interes za društvene igre.

Aplikacija bi mogla privući **entuzijaste** s ogromnim iskustvom i željom za proširenje vidika.

Jednako tako, mogla bi privući i **hobiste**, one koji vole svoje druženje ispuniti društvenim igrama i koji traže nešto novo.

Ali PlayTrade bio bi pogodan i za **početnike**, one koji još nisu sigurni želete li dublje ući u svijet društvenih igara pa istražuju mogućnosti bez troškova.

Mogućnost prilagodbe rješenja

Aplikacija PlayTrade osmišljena je tako da bude **fleksibilna** i **prilagodljiva** potrebama različitih korisnika i potencijalnih tržišta, ali i da bude responzivna, odnosno da se korisničko sučelje prilagodi različitim vrstama i oblicima ekrana.

Korisnici će moći prilagoditi prikaz i način pretraživanja igara prema vlastitim preferencijama. Primjerice, moguće je definirati personalizirane obavijesti o novim objavama igara ili potencijalnim zamjenama prema korisnikovim prethodnim interesima.

Administratorski dio sustava također je dizajniran s mogućnošću prilagodbe. Sustav može biti konfiguriran za različite scenarije, uključujući dodavanje novih kategorija igara, upravljanje korisničkim računima te provjeravanje oglasa.

Opseg projektnog zadatka

Opseg projektnog zadatka obuhvaća sve estetske, funkcionalne, sigurnosne i tehničke aspekte:

- Izrada korisničkog sučelja:** Aplikacija će imati jednostavno korisničko sučelje na kojem će korisnici moći pristupiti registraciji, pristupiti bazi objava i kretati se kroz njih, a jednom kad se registriraju preko korisničkog sučelja moći će pristupiti zamjenama, odvijati interakciju s drugim korisnicima i pristupiti arhivi svojih zamjena i želja.
- Kreiranje filtera za objavljene igre:** Aplikacija će omogućavati korisnicima da pregledaju igre po određenim kategorijama, odnosno da pronađu igre ovisno o tome koliko su zahtjevne, da ih pronađu po tome za koliko igrača su namijenjene, da listaju cijelu kolekciju igara bez filtera ili da pronađu neku igru direktno.
- Kreiranje sustava za objave:** Korisnicima se mora omogućiti da mogu postaviti objavu igre koju žele zamijeniti i pritom popuniti potrebne podatke: naziv, žanr, izdavač, godinu izdanja, ocjenu očuvanosti igre, broj igrača, vrijemeigranja, procjenu težine igre, fotografiju igre te dodatan opis.
- Kreiranje sustava koji podržava interakciju s drugim korisnicima:** Aplikacija mora nuditi korisnicima mogućnost da zatraže zamjenu koja se nudi prije nego ju izvrše
- Kreiranje sustava za sigurnu registraciju:** Korisnici se na aplikaciji moraju registrirati, a ta registracija mora biti sigurna, kao i ostali podaci korisnika.
- Sustav za administratora:** sustavski administratori moraju moći upravljati oglasima i korisnicima.

Moguće nadogradnje projektnog zadatka

Aplikaciju PlayTrade u budućnosti će biti moguće nadograditi kako bi se obogatilo korisničko iskustvo i privukao širi spektar korisnika.

Aplikacija bi u budućnosti mogla omogućavati i zamjene drugih stvari vezane uz igre, primjerice određene figurice u igri.

Osim toga, aplikacija bi mogla nuditi korisnicima i stvaranje igračih grupa s drugima čija je lokacija u blizini, mogla bi pomoći u organiziranju i vođenju natjecanja u nekim igrama.

Također, aplikacija bi mogla nuditi tutoriale za određene igre kako bi pomogla igračima prisjetiti se koja su pravila igre, u slučaju da su svoja pravila izgubili. Ukratko, aplikacija ima širok prostor za napredak koji bi mogao pomoći povećanju njene korisničke baze i uspješnosti.

Cilj izrade projekta

Postoje različiti ciljevi ovog projekta:

- Stvaranje konkretnog rješenja koje bi se moglo plasirati u stvarni svijet
- Stjecanje znanja o razvoju softvera koji je prilagođen tako da pruža ugodno korisničko iskustvo, a istovremeno se brine o sigurnosti podataka i pravilnom radu sustava u pozadini
- Stjecanje iskustva rada u timu kako bi sudionici naučili koje se poteškoće i prilike iz toga mogu javiti i kako ih pretvoriti u snage### Funkcionalni zahtjevi:
- opisuju što sustav treba raditi ili koje funkcionalnosti mora pružiti korisnicima

Nefunkcionalni zahtjevi

- definiraju kako sustav treba raditi, uključujući performanse, pouzdanost, sigurnost i upotrebljivost.

Funkcionalni zahtjevi

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
F-01	Korisnik mora moći pristupiti sustavu.	Visok	Zahtjev dionika	Potencijalni korisnici moraju moći pristupiti stranici na efikasan, konstantan i jednostavan način.
F-02	Korisnik mora moći odabrati svoju lokaciju	Visok	Zahtjev dionika	Korisnik prilikom prijave ili uređivanja profila mora moći odabrati/označiti svoju lokaciju koja će biti vezana uz njegov profil.
F-03	Sustav mora reagirati na neispravnosti koje uzrokuje korisnik	Visok	Zahtjev dionika	U slučaju da korisnik uneše pogrešne podatke ili dođe do neke druge greške na stranici sustav mora pravilno reagirati prema korisniku.
F-04	Sustav mora moći provjeravati neispravnosti	Visok	Zahtjev dionika	Sustav mora moći provjeriti podatke kao što su: valjanost e-maila korisnika, ispravnost lozinke, popunjenošt određenih polja i ostalo.
F-05	Sustav omogućuje korisniku sučelje za objavu igre	Visok	Zahtjev dionika	Korisnik mora moći pristupiti sučelju za objavu svoje igre i jednom kad popuni što treba da se njegova igra postavi i bude vidljiva ostalim korisnicima.
F-06	Odabir lokacije za korisnika treba biti intuitivan i jednostavan	Srednji	Zahtjev dionika	Sučelje kojim korisnik odabire svoju lokaciju treba biti jednostavno i user-friendly.
	Sustav omogućuje korisnicima		Zahtjev	

F-07 ID zahtjeva	kreiranje računa pomoću e-mail opise.	Visok Prioritet	dionika Izvor	Korisnik se može registrirati e-mailom. Kriteriji prihvaćanja
F-08	Sustav provodi autorizaciju	Visok	Zahtjev dionika	Autorizacija se odvija putem OAuth 2.0 protokola koristeći tokene.
F-09	Sustav omogućuje korisnicima pregled objavljenih igara.	Visok	Zahtjev dionika	Čak i neregistrirani korisnik može pristupiti listi objavljenih igara i razgledati je
F-10	Sustav omogućuje korisnicima pretraživanje objavljenih igara.	Visok	Zahtjev dionika	Čak i neregistrirani korisnik može filtrirati igre koje gleda ili tražiti neku igru direktno.
F-11	Sustav omogućuje korisniku upravljanje profilom	Visok	Zahtjev dionika	Registrirani korisnik može objaviti svoju fotografiju i navesti kratak opis te birati kategorije igara po svom interesu.
F-12	Obavezno je da svaki profil ima pripadnu lokaciju.	Visok	Poslovno pravilo	Obavezno je da korisnicima jedna od stavki njihova profila bude njihova lokacija, za što se koristi vanjska usluga-OpenStreetMap.
F-13	Sustav omogućava korisniku da pristupi arhivi svojih zamjena	Srednji	Zahtjev dionika	Korisnik treba moći na stranici pristupiti svojim bivšim zamjenama i pregledati ih.
F-14	Sustav omogućuje korisniku unošenje detalja o igri.	Visok	Zahtjev dionika	Registrirani korisnik za svaku igru koju objavljuje navodi žanr, izdavača, godinu izdanja, ocjenu očuvanosti, broj igrača, vrijeme igranja, procjenu težine, fotografiju igre i dodatan opis ako je potreban.
F-15	Sustav omogućuje korisniku uređivanje svojih oglasa.	Visok	Zahtjev dionika	Registrirani korisnik može pristupiti svim svojim objavama i uređivati ih na pregledu "Moje igre".
F-16	Sustav omogućuje korisniku ponudu zamjene.	Visok	Zahtjev dionika	Registrirani korisnik može za željenu igru ili igre ponuditi zamjenu gumbom "Ponudi zamjenu".
F-17	Sustav mora javiti korisniku kad je zamjena dostupna i ponuditi mu njeno prihvatanje, odbijanje ili izmjenu.	Visok	Zahtjev dionika	Registrirani korisnik na e-mail mora dobiti obavijest o tome kad je zamjena dostupna i onda ju može prihvati, odbiti ili dodatno urediti.
F-18	Sistemski administratori održavaju platformu.	Visok	Administratori sustava	Sistemski administratori mogu upravljati korisnicima i oglasima (brisati ih i deaktivirati).

Ostali zahtjevi

Zahtjevi za performanse

ID zahtjeva	Opis	Prioritet
NF-1.1	Sustav treba raditi dobro bez obzira na broj korisnika.	Visok
NF-1.2	Sustav treba omogućiti brzu pretragu igara.	Srednji

Zahtjevi za iskustvo korisnika

ID zahtjeva	Opis	Prioritet
NF-2.1	Sustav mora biti razvijen kao web-aplikacija.	Visok
NF-2.2	Web-aplikacija mora biti responzivna.	Srednji

Zahtjevi za održavanje

ID zahtjeva	Opis	Prioritet
NF-3.1	Sustav treba biti intuitivno organiziran i pregledan.	Srednji
NF-3.2	Sustav treba imati dovoljnu dokumentaciju.	Visok

Zahtjevi za sigurnost i skalabilnost

ID zahtjeva	Opis	Prioritet
NF-4.1	Aplikacija treba osigurati sigurnost privatnih podataka korisnika.	Visok
NF-4.2	Sustav treba biti skalabilan i podržavati proširenje za nove objave.	Visok
NF-4.3	Sustav treba biti responzivan kako bi se mogao koristiti na različitim uređajima.	Visok
NF-4.4	Sustav treba biti pravilno i jednostavno integriran s vanjskim sustavima.	Srednji

Dionici

- Administrator (F-18)
- Registrirani korisnici (F-01, F-02, F-05, F-07, F-09, F-10, F-11, F-13, F-14, F-15, F-16, F-17)
- Neregistrirani korisnici (F-01, F-09, F-10)
- Prijavljeni korisnik (F-01, F-02, F-05, F-07, F-09, F-10, F-11, F-13, F-14, F-15, F-16, F-17)
- Vanjska usluga geolokacije (F-02, F-12)
- Baza podataka

Dionici i njihovi funkcionalni zahtjevi:

Administrator može:

- upravljati oglasima i korisnicima, tj. brisati oglase i blokirati korisnike (F-18)

Neregistrirani korisnik može:

- koristiti ograničene funkcionalnosti web-aplikacije, odnosno smije pretraživati i pregledavati objavljene igre (F-01, F-09, F-10)

Registrirani korisnik može:

- sve što i neregistrirani korisnik i dodatno:
 - primati obavijest o ponudi zamjene (F-16, F-17)
 - primati obavijest o dostupnosti igre s liste želja (F-17)

Prijavljeni korisnik može:

- sve što i registrirani korisnik i dodatno:
 - objavljivati igre (F-05, F-14)
 - nuditi zamjene (F-16)
 - uređivati svoj profil (F-11, F-02)
 - pristupati arhivi ponuda i zamjena (F-13)
 - uređivati svoje objave (F-15)

Baza podataka:

- služi za pohranu svih korisničkih podataka
- na zahtjev poslužitelja daje potrebne podatke

Vanjska usluga geolokacije:

- služi za lociranje osoba za vršenje zamjena (F-02, F-12)

Obrasci uporabe

Oznaka UC	Naziv obrasca uporabe	Povezani funkcionalni zahtjevi

Uč-01 UC UC-02	Registracija korisnika Naziv obrazca uporabe	F-07, F-02, F-04, F-12 Povezani funkcionalni zahtjevi
UC-02	Prijava korisnika	F-01, F-08, F-04
UC-03	Uređivanje profila	F-11, F-02, F-12
UC-04	Pregled svih igara	F-09
UC-05	Objava nove igre	F-05, F-14
UC-06	Uređivanje i brisanje svojih objava	F-15
UC-07	Pretraživanje igara	F-10
UC-08	Ponuda zamjene	F-16, F-17
UC-09	Obavijest o novoj ponudi	F-17
UC-10	Prihvatanje ili izmjena ponude	F-17
UC-11	Pregled ponuda	F-16, F-17
UC-12	Stvaranje popisa želja	F-09, F-10
UC-13	Obavijest o dostupnosti igara s popisa želja	F-17, F-09
UC-14	Arhiviranje izvršenih zamjena	F-13, F-17
UC-15	Upravljanje korisnicima i oglasima	F-18
UC-16	Deaktivacija korisnika i oglasa zbog kršenja pravila	F-18

Opis obrazaca uporabe

UC-01: Registracija

Opis:

Korisnik unosi valjanu e-mail adresu, lozinku za web-aplikaciju, bira svoju lokaciju i registrira se.

Povezani funkcionalni zahtjevi: F-07, F-02, F-04, F-12

Detalji obrazca uporabe

- **Glavni sudionik:**
 - Neregistrirani korisnik
- **Cilj:**
 - Stvaranje novog korisničkog računa
- **Sudionici:**
 - Neregistrirani korisnik, vanjska usluga geolokacije
- **Preduvjet:**
 - Korisnik nema postojeći račun
- **Osnovni tijek:**
 1. Korisnik otvara aplikaciju i odabire opciju 'Registriraj se' ili to čini putem Googlea (F-07 ili F-08)
 2. Unosi valjanu e-mail adresu (F-07)
 3. Odabire i unosi lozinku za aplikaciju (F-07)
 4. Koristeći OpenStreetMap na sustavu odabire svoju lokaciju (F-02, F-12)
 5. Provjerava se valjanost e-mail adrese i da korisnik s tom adresom već ne postoji (F-04)
 6. Ako je sve u redu, kreira se novi korisnički račun kojem su ostali podatci ispunjeni (F-07)
- **Moguća odstupanja:**
 - **Nevaljana e-mail adresa:** Sustav prikazuje obavijest o greški. (F-04)
 - **Korisnik već postoji:** Sustav obavještava da korisnik s tom e-mail adresom već postoji. (F-04)

UC-02: Prijava korisnika

Opis:

Korisnik unosi valjanu e-mail adresu i lozinku te se prijavljuje na stranicu ili se prijavljuje putem Googlea

Povezani funkcionalni zahtjevi: F-01, F-08, F-04

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Registrirani korisnik
- **Cilj:**
 - Prijava na stranicu i omogućavanje proširenih aktivnosti korisniku
- **Sudionici:**
 - Registrirani korisnik
- **Preduvjet:**
 - Korisnik se prethodno registrirao
- **Osnovni tijek:**
 1. Korisnik otvara aplikaciju i odabire opciju 'Prijava' ili se prijavljuje putem Googlea (F-01 ili F-08)
 2. Unosi valjanu e-mail adresu (F-04)
 3. Unosi svoju lozinku (F-04)
 4. Sustav provjerava valjanost podataka i vrši autorizaciju (F-08)
 5. Ako su podaci valjni, korisnik je prijavljen (F-01)
- **Moguća odstupanja:**
 - **Ne postoji račun s tom e-mail adresom:** Sustav prikazuje obavijest o greški. (F-04)
 - **Unesena neispravna lozinka:** Sustav obaveštava da je lozinka neispravna. (F-04)

UC-03: Uređivanje profila

Opis:

Korisnik može mijenjati podatke na svom profilu, konkretno: fotografiju profila, opis profila, kategorije igara od interesa, lokaciju.

Povezani funkcionalni zahtjevi: F-11, F-02, F-12

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Prijavljeni korisnik
- **Cilj:**
 - Korisnik može urediti svoj korisnički profil po svojim željama
- **Sudionici:**
 - Prijavljeni korisnik, vanjska usluga geolokacije
- **Preduvjet:**
 - Korisnik se prethodno registrirao i prijavio
- **Osnovni tijek:**
 1. Prijavljeni korisnik uđe u web-aplikaciju i klikne na ikonu svog profila (F-11)
 2. Odabire i mijenja elemente profila (F-11)
 3. Ažurira ili mijenja lokaciju pomoću kartografske usluge (F-02, F-12)
- **Moguća odstupanja:**
 - **Prijavljeni korisnik svojim promjenama prekrši pravila ponašanja:** Sustavski administratori uređuju ili brišu njegov profil. (F-18)

UC-04: Pregled svih igara

Opis:

Korisnik može pregledavati igre koje su objavili drugi korisnici.

Povezani funkcionalni zahtjevi: F-09

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Korisnik
- **Cilj:**
 - Korisnik može listati kroz igre koje su objavili drugi korisnici
- **Sudionici:**
 - Korisnik
- **Preduvjet:**
 - Korisnik je uspješno pristupio web-aplikaciji
- **Osnovni tijek:**
 1. Korisnik otvara aplikaciju (F-09)
 2. Ako želi, korisnik se može registrirati i/ili prijaviti (F-09)
 3. Odabire stranicu za pregled objava (F-09)
 4. Korisnik pregledava objavljene igre (F-09)
- **Moguća odstupanja:**
 - **Nema nijedna objava:** Sustav obaveštava korisnika o tome da još nema nijedna objavljena društvena igra. (F-09)

UC-05: Objava nove igre

Opis:

Korisnik može objaviti novu igru na način da odabere 'Objavi igru'.

Povezani funkcionalni zahtjevi: F-05, F-14

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Prijavljeni korisnik
- **Cilj:**
 - Korisnik može ponuditi neku svoju igru na zamjenu tako da je vide svi drugi korisnici
- **Sudionici:**
 - Prijavljeni korisnik
- **Preduvjet:**
 - Korisnik se prethodno registrirao i prijavio
- **Osnovni tijek:**
 1. Prijavljeni korisnik otvara aplikaciju (F-05)
 2. Odabire stranicu za objavu igara (F-05)
 3. Korisnik popunjava potrebne podatke o igri (F-14)
 4. Ako su podaci popunjeni korisnik stisne gumb 'Objavi' (F-05)
 5. Igra se objavljuje i postaje vidljiva drugim korisnicima (F-05)
- **Moguća odstupanja:**
 - **Podaci krše pravila ponašanja:** Administratori reagiraju na nepoštivanje pravila. (F-18)
 - **Nisu popunjeni svi podaci:** Korisnik ne može objaviti igru dok ne ispuni sva polja. (F-14)

UC-06: Uređivanje i brisanje svojih objava

Opis:

Korisnik može pristupiti svojim objavljenim igram na stranici za uređivanje i brisanje.

Povezani funkcionalni zahtjevi: F-15

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Prijavljeni korisnik
- **Cilj:**

- Korisnik može upravljati svojim već objavljenim igrama po želji
- **Sudionici:**
 - Prijavljeni korisnik
- **Preduvjet:**
 - Korisnik se prethodno registrirao i prijavio
- **Osnovni tijek:**
 1. Prijavljeni korisnik otvara aplikaciju (F-15)
 2. Odabire stranicu 'Moje igre' (F-15)
 3. Odabire gumb 'Uredi' ili 'Obriši' (F-15)
 4. Uređuje ili briše objavu (F-15)
- **Moguća odstupanja:**
 - **Nema objava:** Sustav obaveštava korisnika. (F-15)
 - **Podaci krše pravila ponašanja:** Administratori reagiraju. (F-18)

UC-07: Pretraživanje igara

Opis:

Korisnik može filtrirati igre koje su objavili drugi korisnici ili tražiti neku konkretnu igru.

Povezani funkcionalni zahtjevi: F-10

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Korisnik
- **Cilj:**
 - Korisnik može koristiti filtere za pretragu igara ili tražilicom tražiti specifičnu igru koju ima na umu
- **Sudionici:**
 - Korisnik
- **Preduvjet:**
 - Korisnik može pristupiti web-aplikaciji
- **Osnovni tijek:**
 1. Korisnik otvara aplikaciju (F-10)
 2. Odabire stranicu za pregled igara (F-10)
 3. Filtrira igre ili koristi tražilicu (F-10)
- **Moguća odstupanja:**
 - **Nema objava koje zadovoljavaju filter:** Sustav obaveštava korisnika. (F-10)

UC-08: Ponuda zamjene

Opis:

Korisnik može za određenu igru drugog korisnika ponuditi zamjenu.

Povezani funkcionalni zahtjevi: F-16, F-17

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Prijavljeni korisnik
- **Cilj:**
 - Korisnik može korisniku koji je vlasnik igre koju želi poslati zahtjev za ponudu s detaljima ponude
- **Sudionici:**
 - Prijavljeni korisnik
- **Preduvjet:**
 - Korisnik se prethodno registrirao i prijavio
- **Osnovni tijek:**

1. Korisnik otvara aplikaciju (F-16)
2. Pregledava objavljene igre (F-16)
3. Odabire gumb 'Ponudi zamjenu' (F-16)
4. Unosi igre koje nudi (F-16)
5. Klikne gumb 'Ponudi' (F-16)
6. Vlasniku igre stiže obavijest (F-17)

- **Moguća odstupanja:**

- **Nema dostupnih igara:** Sustav obaveštava korisnika. (F-16)

UC-09: Obavijest o novoj ponudi

Opis:

Korisniku za čiju je igru ponuđena zamjena dolazi obavijest u sustav i na e-mail.

Povezani funkcionalni zahtjevi: F-17

Detalji obrasca uporabe

- **Glavni sudionik:**

- Registrirani korisnik

- **Cilj:**

- Korisnik prima obavijest o tome da netko želi njegovu igru s objave

- **Sudionici:**

- Registrirani korisnik

- **Preduvjet:**

- Korisnik se prethodno registrirao i ima objavljenu igru koju netko može zatražiti

- **Osnovni tijek:**

1. Neki korisnik ponudi zamjenu (F-17)
2. Vlasniku igre stiže e-mail obavijest (F-17)
3. Prikazuje se i obavijest u aplikaciji (F-17)

- **Moguća odstupanja:**

- **Korisnik nije registriran:** Ne može praviti objave, pa time ni primati ponude. (F-17)

UC-10: Prihvatanje ili izmjena ponude

Opis:

Korisnik može ponudu prihvatiti, odbiti ili ponuditi kontra-ponudu.

Povezani funkcionalni zahtjevi: F-17

Detalji obrasca uporabe

- **Glavni sudionik:**

- Prijavljeni korisnik

- **Cilj:**

- Korisnik može uređivati uvjete zamjene, prihvatiti je ili odbiti

- **Sudionici:**

- Prijavljeni korisnik

- **Preduvjet:**

- Korisnik se prethodno registrirao i prijavio

- **Osnovni tijek:**

1. Prijavljeni korisnik otvara aplikaciju (F-17)
2. Pristupa ponudi koju je dobio (F-17)
3. Prihvata, odbija ili uređuje ponudu (F-17)
4. Drugoj strani stiže obavijest (F-17)

- **Moguća odstupanja:**

- **Korisnik nije prijavljen:** Ne može upravljati ponudama dok se ne prijavi na stranici. (F-17)

UC-11: Pregled ponuda

Opis:

Na stranici 'Ponude' registriranog korisnika može se pristupiti svim primljenim ponudama.

Povezani funkcionalni zahtjevi: F-16, F-17

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Prijavljeni korisnik
- **Cilj:**
 - Korisnik može pristupiti svim ponudama koje trenutno ima na jednom mjestu
- **Sudionici:**
 - Prijavljeni korisnik
- **Preduvjet:**
 - Korisnik se prethodno registrirao i prijavio
- **Osnovni tijek:**
 1. Prijavljeni korisnik otvara aplikaciju (F-16)
 2. Pristupa stranici 'Ponude' (F-16)
 3. Pregledava ponude koje je dobio (F-17)
- **Moguća odstupanja:**
 - **Nema ponuda:** Sustav obavještava korisnika. (F-16)
 - **Korisnik nije prijavljen:** Ne može pristupiti ponudama. (F-17)

UC-12: Stvaranje popisa želja

Opis:

Korisnik može stvoriti popis igara koje želi dobiti zamjenom.

Povezani funkcionalni zahtjevi: F-09, F-10

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Prijavljeni korisnik
- **Cilj:**
 - Korisnik može stvoriti listu igara koje želi dobiti zamjenom, a ne može ih naći među objavama
- **Sudionici:**
 - Prijavljeni korisnik
- **Preduvjet:**
 - Korisnik se prethodno registrirao i prijavio
- **Osnovni tijek:**
 1. Prijavljeni korisnik otvara aplikaciju (F-09)
 2. Pristupa profilu i opciji 'Moje želje' (F-10)
 3. Odabire igre koje želi (F-10)
 4. Sprema promjene (F-09)
- **Moguća odstupanja:**
 - **Nepostojeća igra:** Sustav dopušta izbor samo postojećih/sustavu poznatih igara. (F-09)
 - **Korisnik nije prijavljen:** Ne može pristupiti profilu. (F-09)

UC-13: Obavijest o dostupnosti igara s popisa želja

Opis:

Korisniku dolazi obavijest kad netko objavi igru koja je na njegovu popisu želja.

Povezani funkcionalni zahtjevi: F-17, F-09

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Registrirani korisnik
- **Cilj:**
 - Korisniku se javlja kad je njegova igra s popisa želja dostupna za zamjenu kako bi mogao pravovremeno reagirati
- **Sudionici:**
 - Registrirani korisnik
- **Preduvjet:**
 - Korisnik se prethodno registrirao
- **Osnovni tijek:**
 1. Neki korisnik objavi novu igru (F-09)
 2. Sustav šalje obavijest korisnicima koji je imaju na popisu (F-17)
 3. Obavijest se prikazuje i na aplikaciji (F-17)
- **Moguća odstupanja:**
 - **Korisnik nije prijavljen:** Ne može reagirati na obavijest. (F-17)
 - **Korisnik nije registriran:** Nema korisnički profil, pa ne može ni praviti listu želja. (F-17)

UC-14: Arhiviranje izvršenih zamjena

Opis:

Korisnik može vidjeti koje je zamjene već obavio.

Povezani funkcionalni zahtjevi: F-13, F-17

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Prijavljeni korisnik
- **Cilj:**
 - Korisniku može pristupiti obavljenim zamjenama kako bi ih mogao analizirati ako bude potrebno
- **Sudionici:**
 - Prijavljeni korisnik
- **Preduvjet:**
 - Korisnik se prethodno registrirao i prijavio
- **Osnovni tijek:**
 1. Prijavljeni korisnik otvara aplikaciju (F-17)
 2. Odabire stranicu 'Moje zamjene' (F-13)
 3. Pregledava arhivirane zamjene (F-13)
- **Moguća odstupanja:**
 - **Nema arhiviranih zamjena:** Sustav obavještava korisnika. (F-13)
 - **Korisnik nije prijavljen:** Ne može pristupiti arhivi. (F-17)

UC-15: Upravljanje korisnicima i oglasima

Opis:

Sistemski administratori mogu upravljati korisnicima i oglasima.

Povezani funkcionalni zahtjevi: F-18

Detalji obrasca uporabe

- **Glavni sudionik:**
 - Administrator
- **Cilj:**
 - Administratori mogu upravljati korisnicima, njihovim profilima i objavama ako smatraju da je potrebno
- **Sudionici:**
 - Administratori

- Administrator
- **Preduvjet:**
 - Administrator je koristio svoje administratorske podatke za prijavu u sustav
- **Osnovni tijek:**
 1. Administrator se prijavljuje u sustav (F-18)
 2. Otvara administratorsko sučelje (F-18)
 3. Pregledava korisnike i oglase (F-18)
 4. Upravlja ili uređuje zapise (F-18)
- **Moguća odstupanja:**
 - **Nema podataka:** Sustav prikazuje poruku o neaktivnosti. (F-18)
 - **Administrator nije prijavljen:** Nema pristup funkcijama. (F-18)

UC-16: Deaktivacija korisnika i oglasa koji krše pravila

Opis:

Sistemski administratori mogu uklanjati korisnike i oglase.

Povezani funkcionalni zahtjevi: F-18

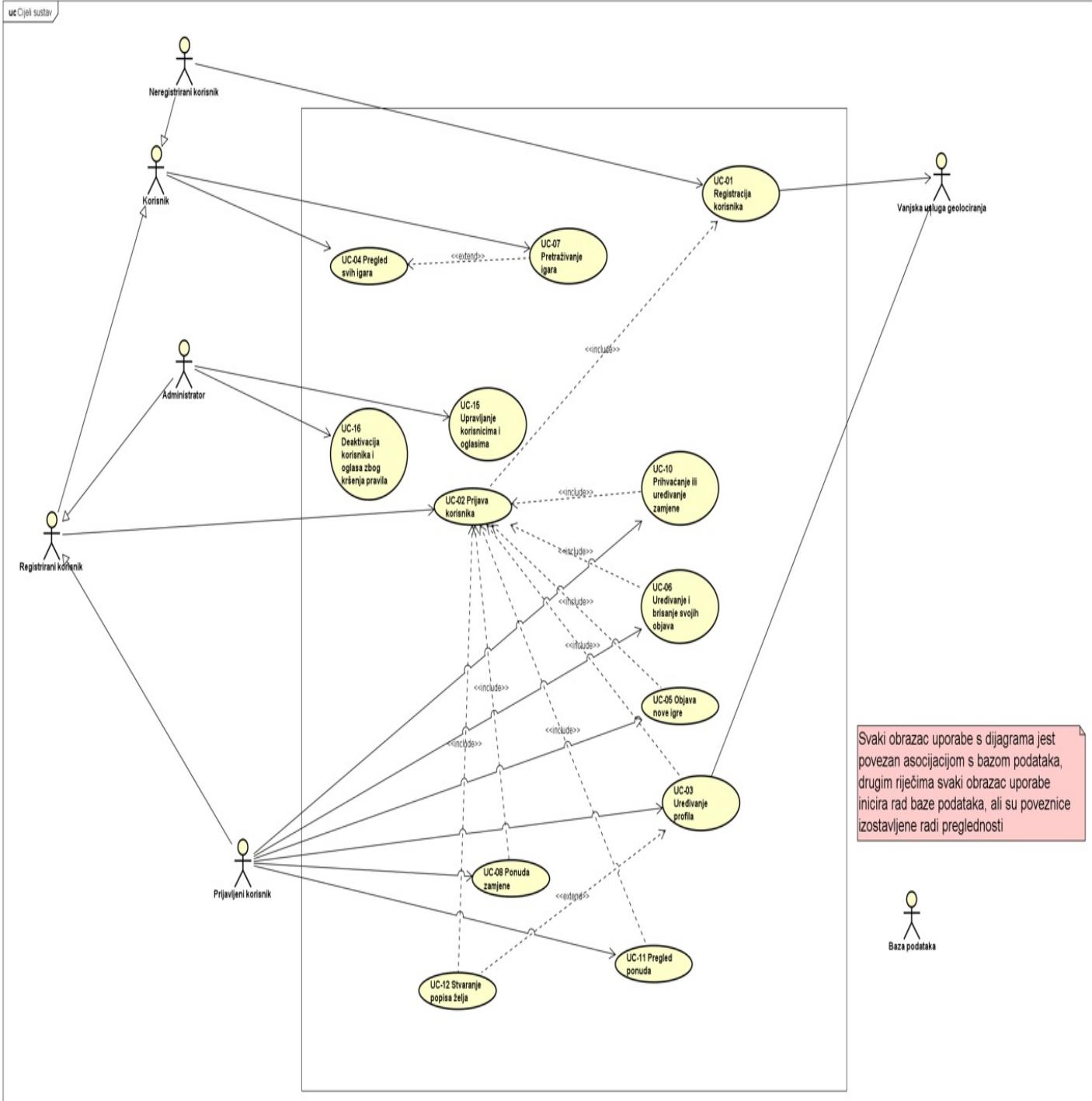
Detalji obrasca uporabe

- **Glavni sudionik:**
 - Administrator
- **Cilj:**
 - Administrator uklanja korisnike i/ili objave ako krše pravila ponašanja
- **Sudionici:**
 - Administrator
- **Preduvjet:**
 - Administrator je koristio svoje administratorske podatke za prijavu i upoznat je s pravilima ponašanja
- **Osnovni tijek:**
 1. Administrator se prijavljuje (F-18)
 2. Pristupa sučelju za upravljanje korisnicima (F-18)
 3. Odabire korisnika ili oglas za deaktivaciju (F-18)
 4. Potvrđuje deaktivaciju (F-18)
- **Moguća odstupanja:**
 - **Administrator nije prijavljen:** Nema pristup funkciji. (F-18)

Dijagrami obrazaca uporabe

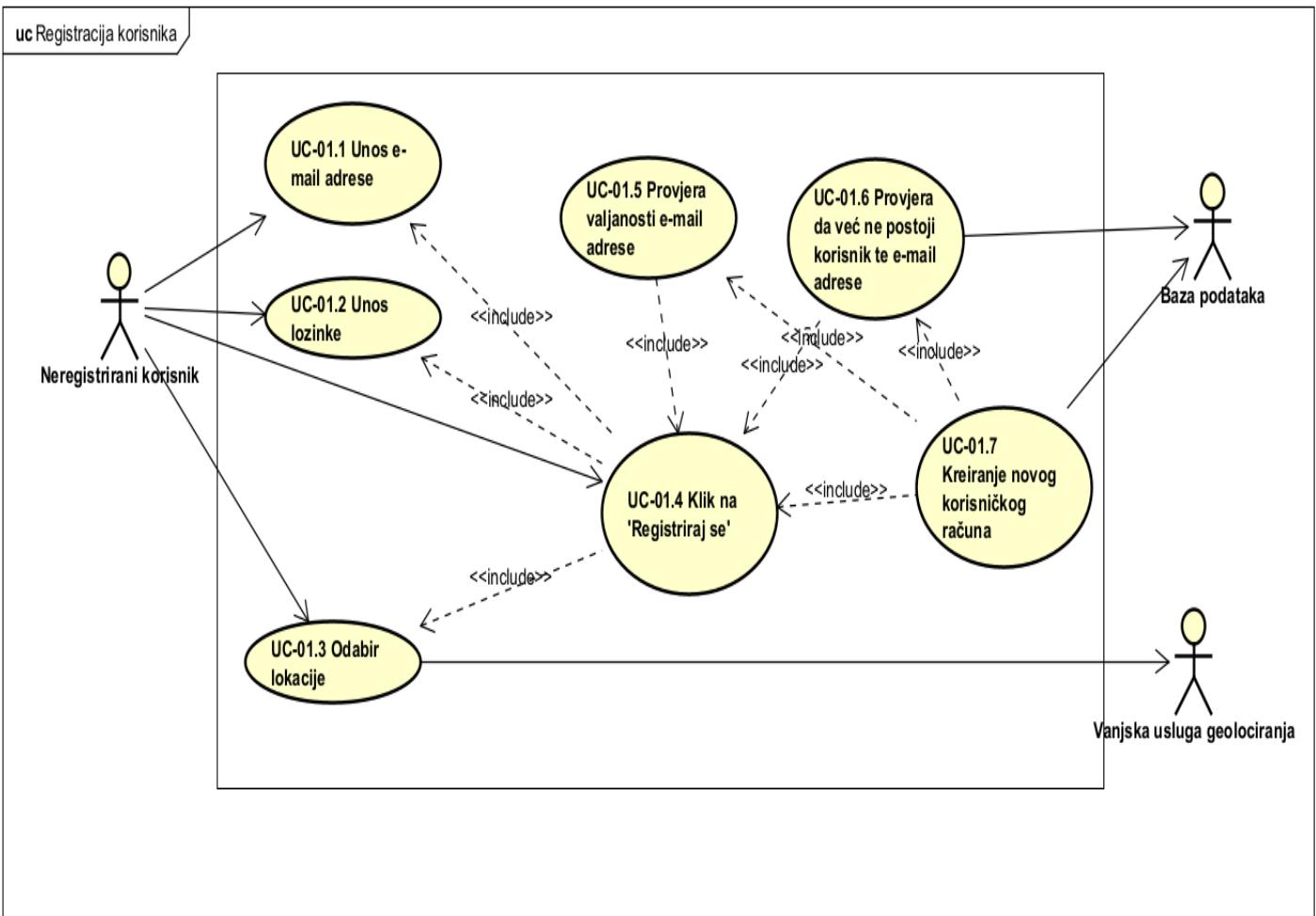
1. Visokorazinski dijagram obrazaca uporabe cijelog sustava

Slika 3.1 Dijagram osnovnih funkcionalnosti cijelog sustava

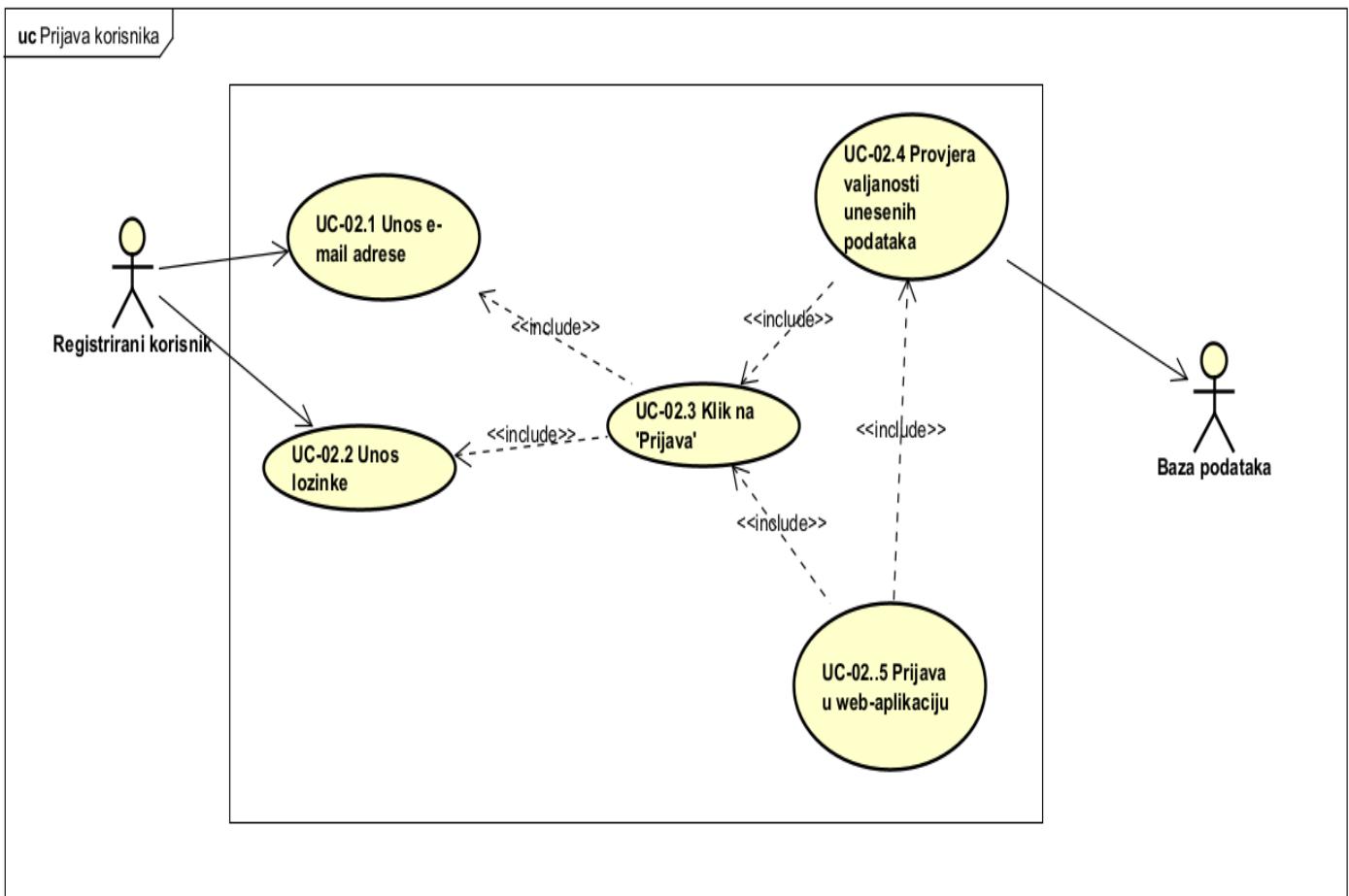


2. dijagram obrazaca uporabe za ključne značajke

Slika 3.2 Dijagram funkcionalnosti registracije

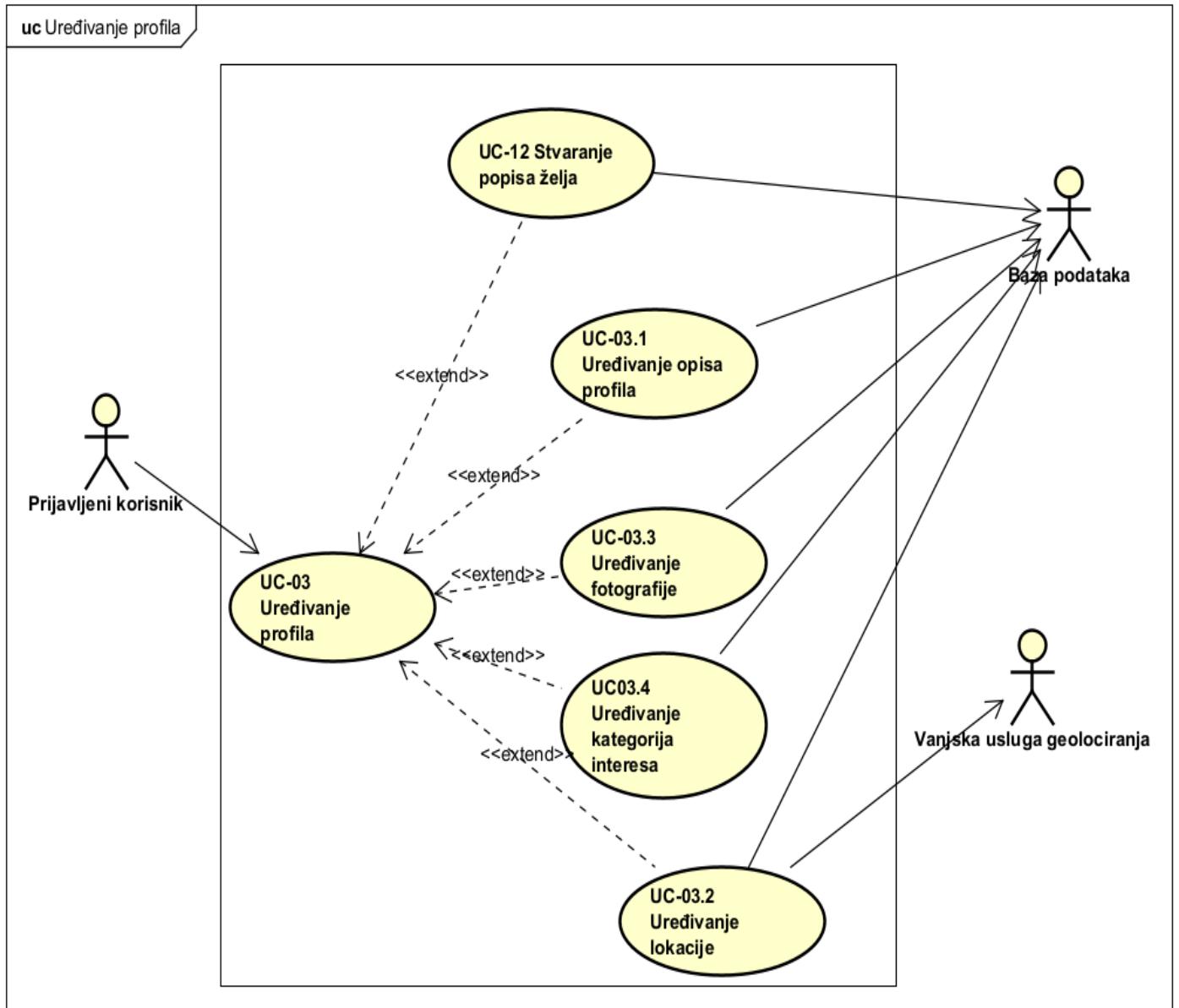


Slika 3.3 Dijagram funkcionalnosti prijave korisnika



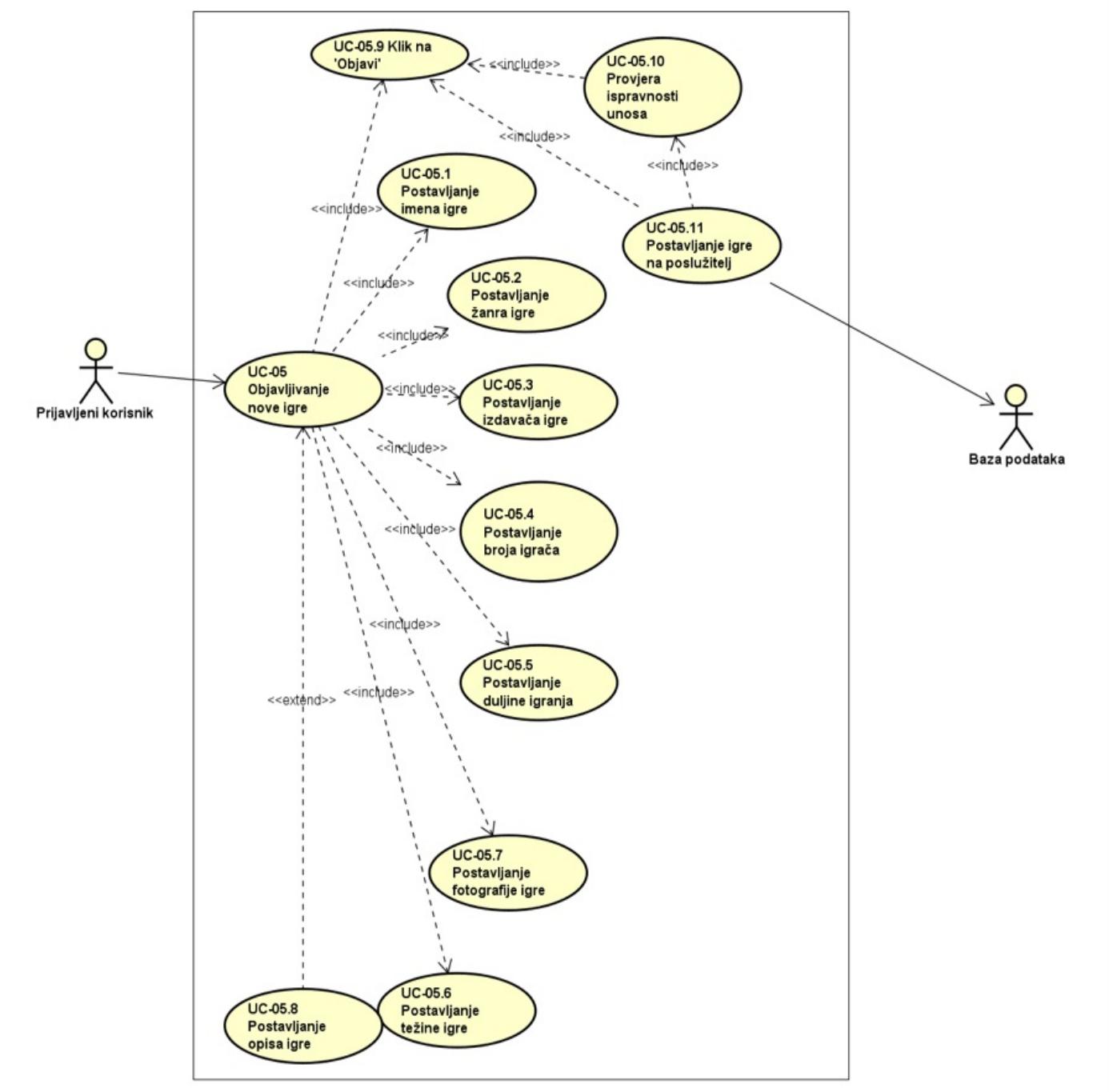
3. dijagram obrazaca uporabe za korisničke uloge

Slika 3.4 Dijagram funkcionalnosti uređivanja profila

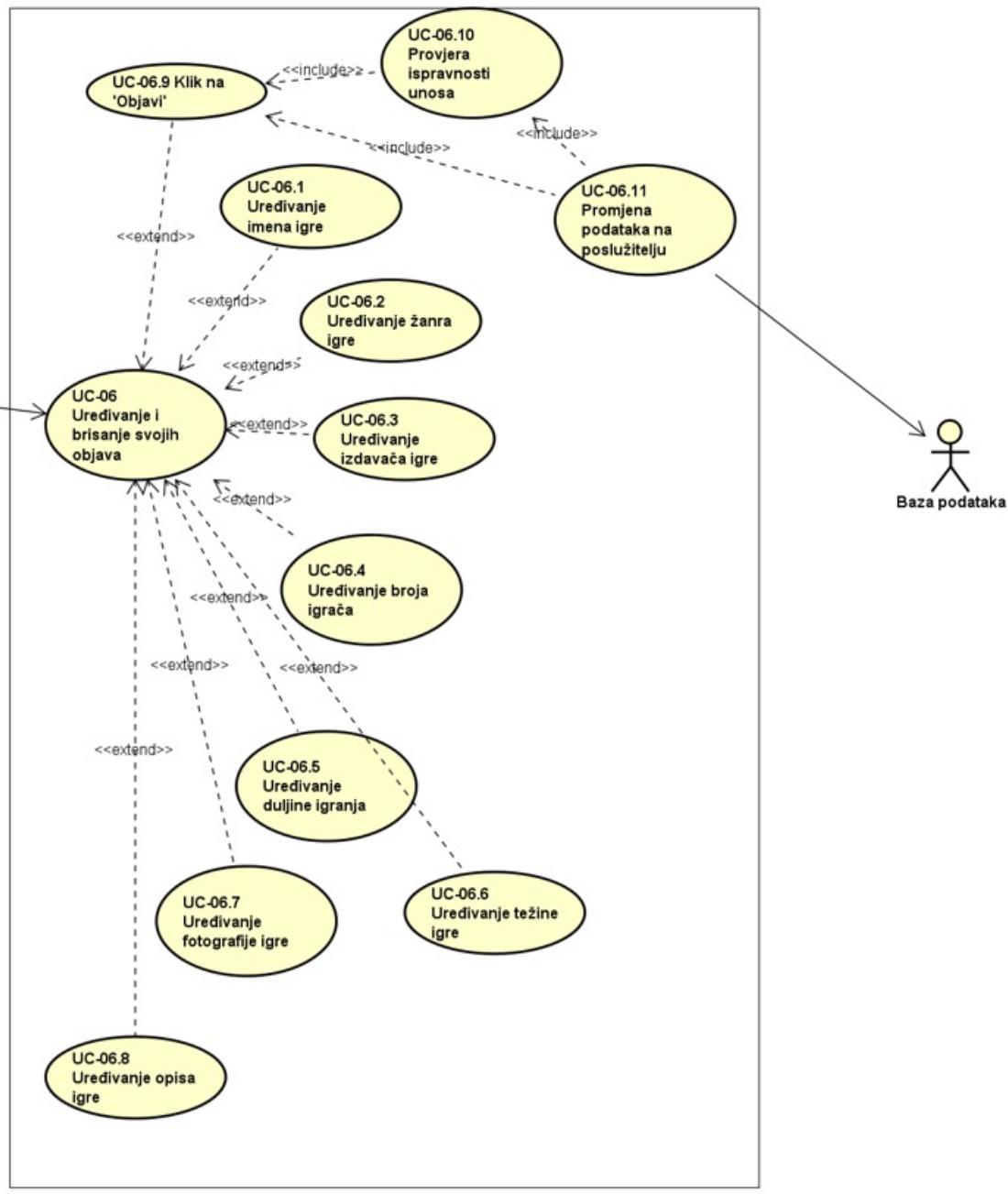


Slika 3.5 Dijagram objave nove igre

uc Objava nove igre

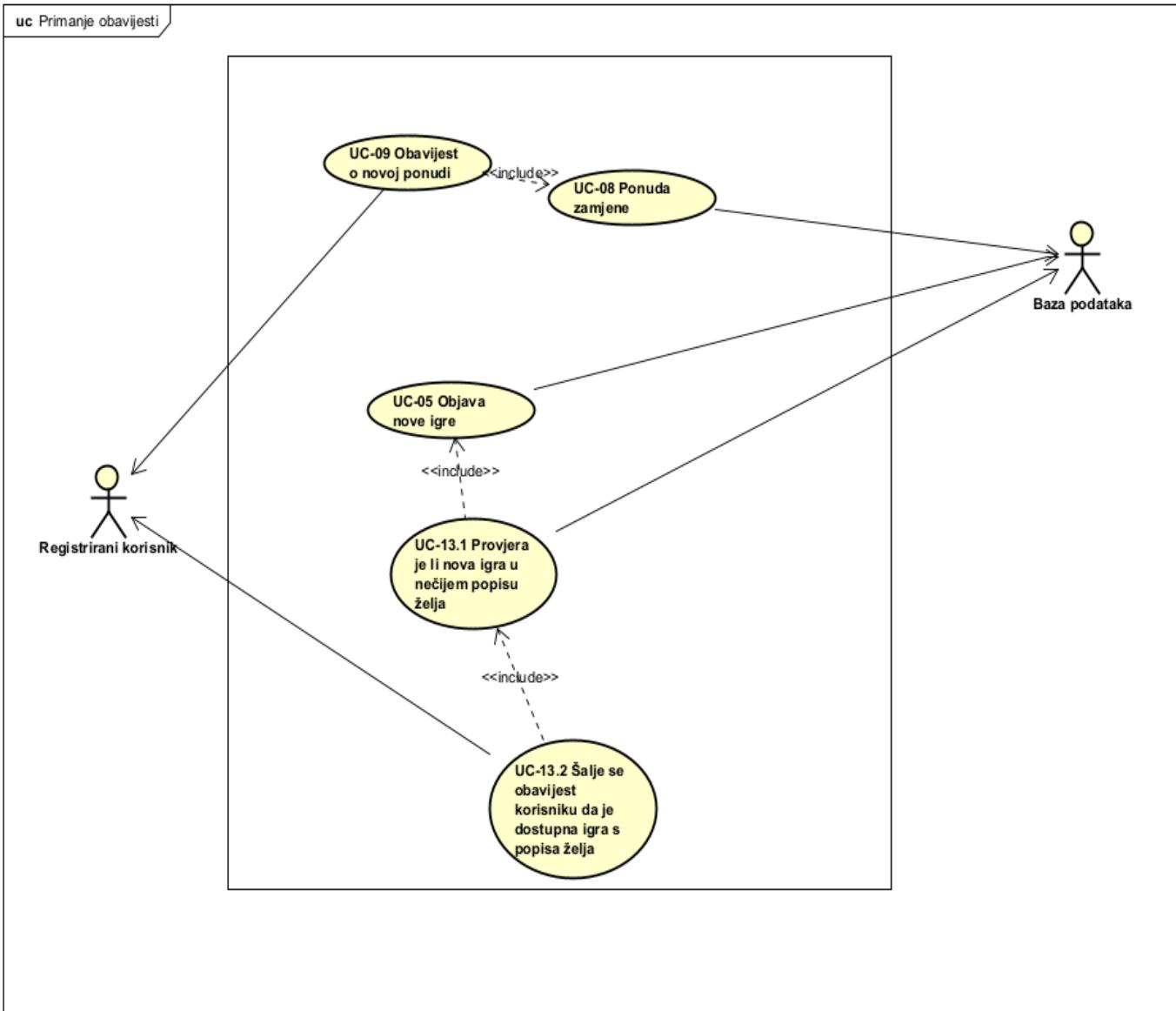


Slika 3.6 Dijagram uređivanja svoje objave igre

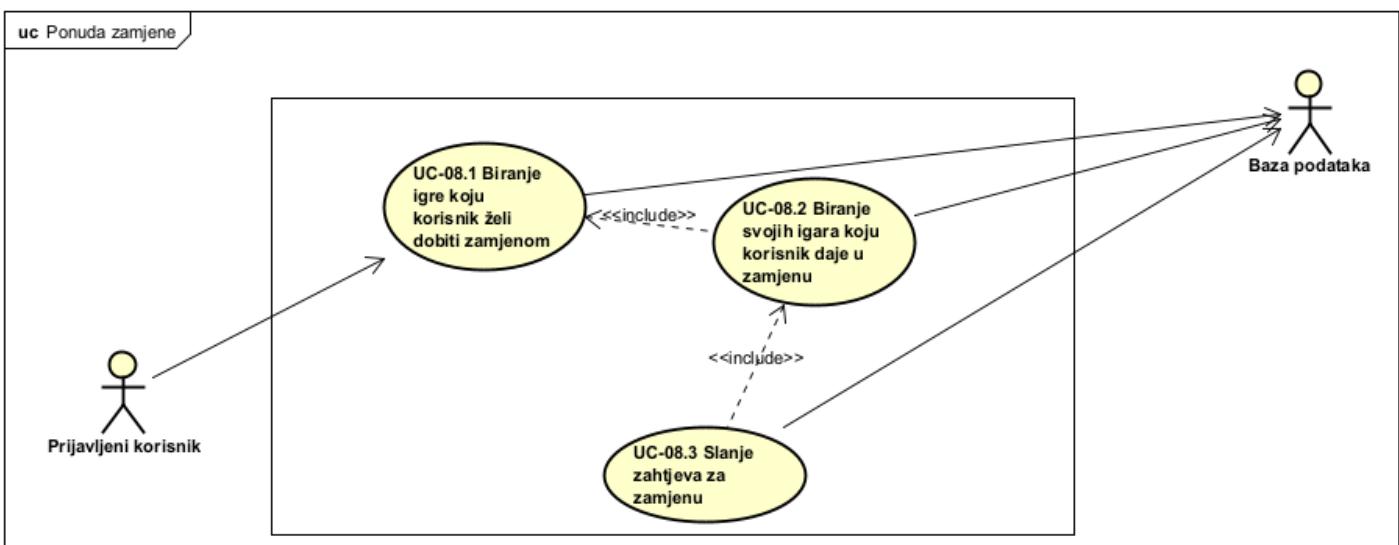


4. dijagram obrazaca uporabe za osnovne poslovne procese

Slika 3.7 Dijagram primanja i slanja obavijesti



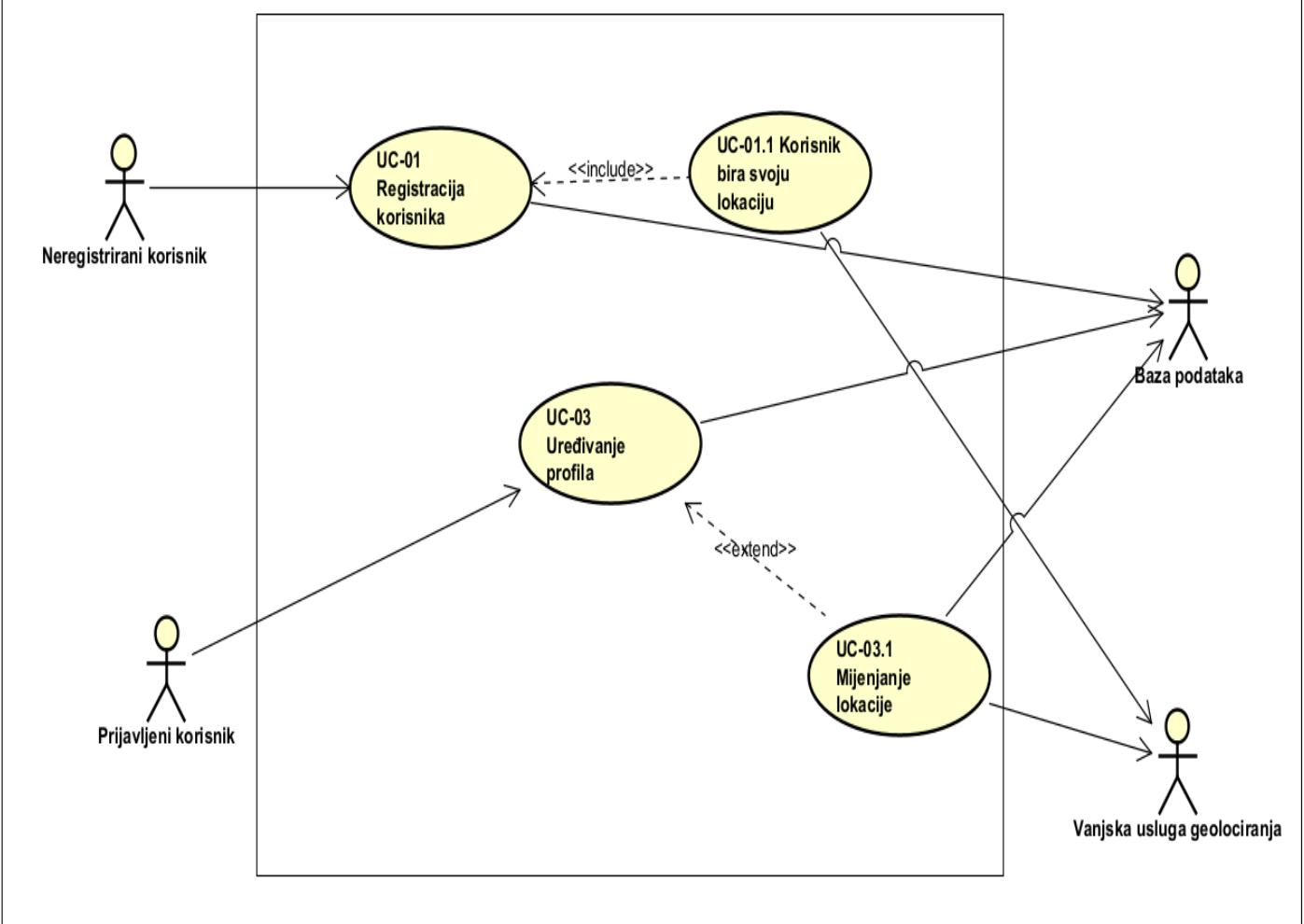
Slika 3.8 Dijagram ponude zamjene



5. dijagram obrazaca uporabe za kritične sustave i integracije

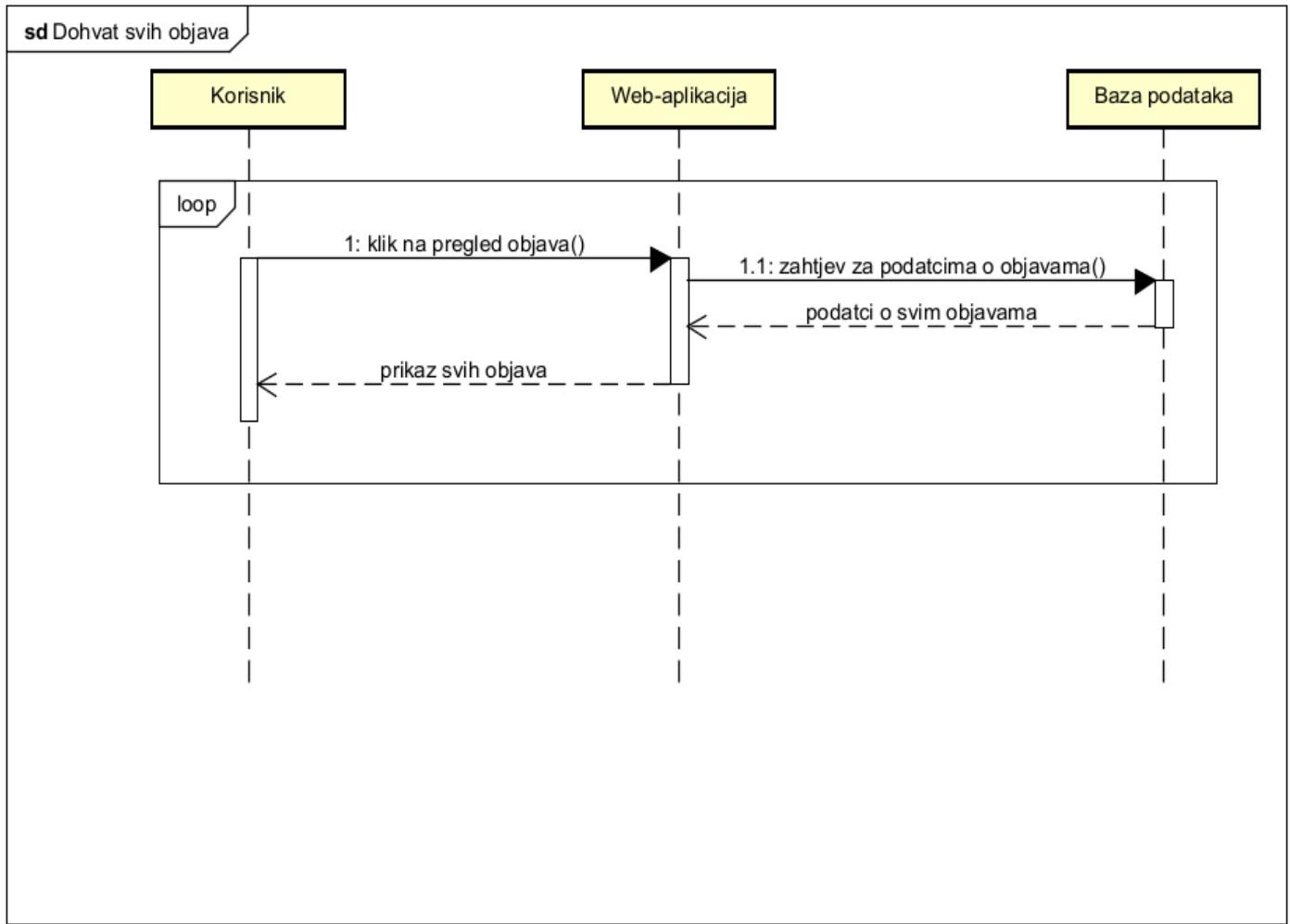
Slika 3.9 Dijagram korištenja vanjske usluge

ucKorištenje vanjske usluge

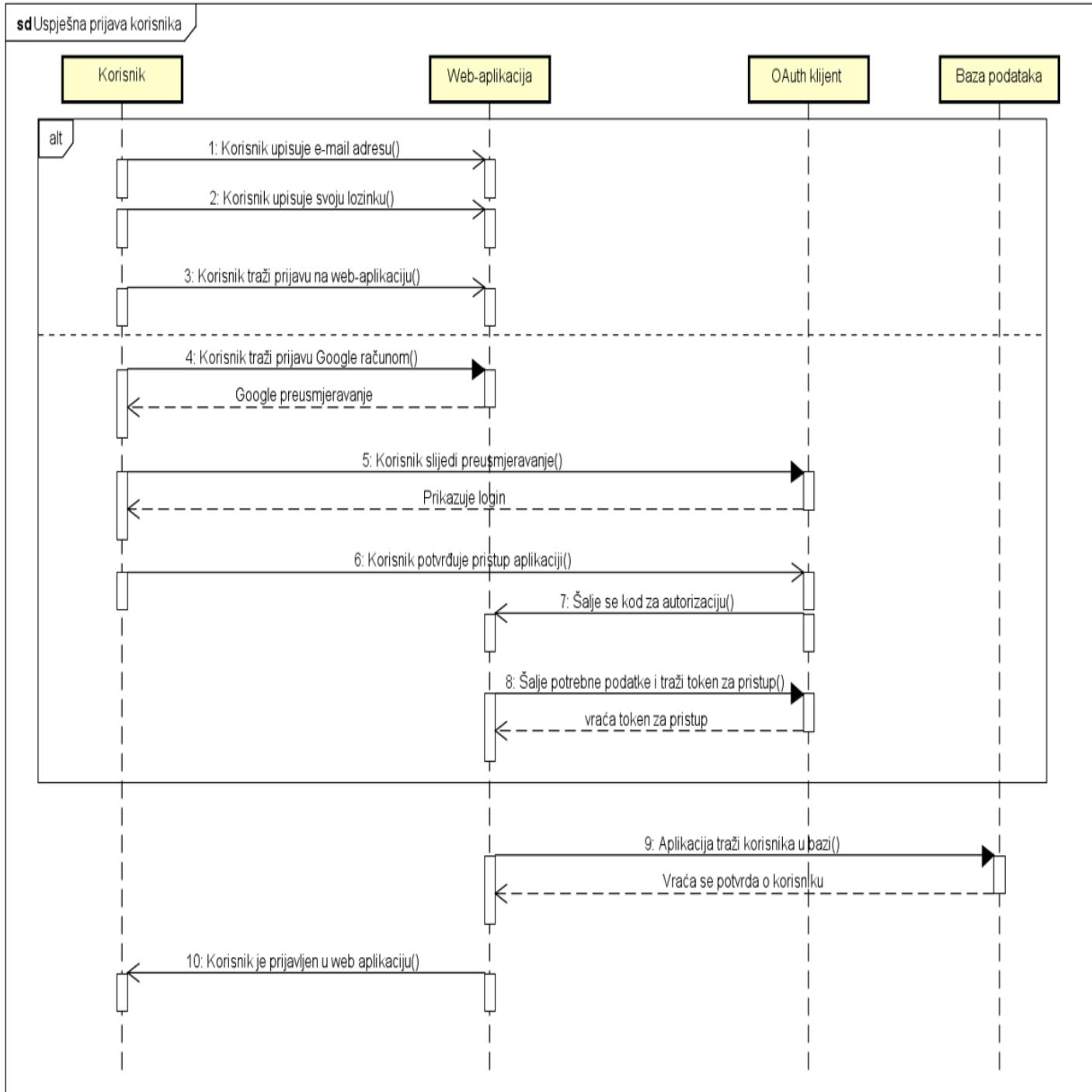


Sekvencijski dijagrami

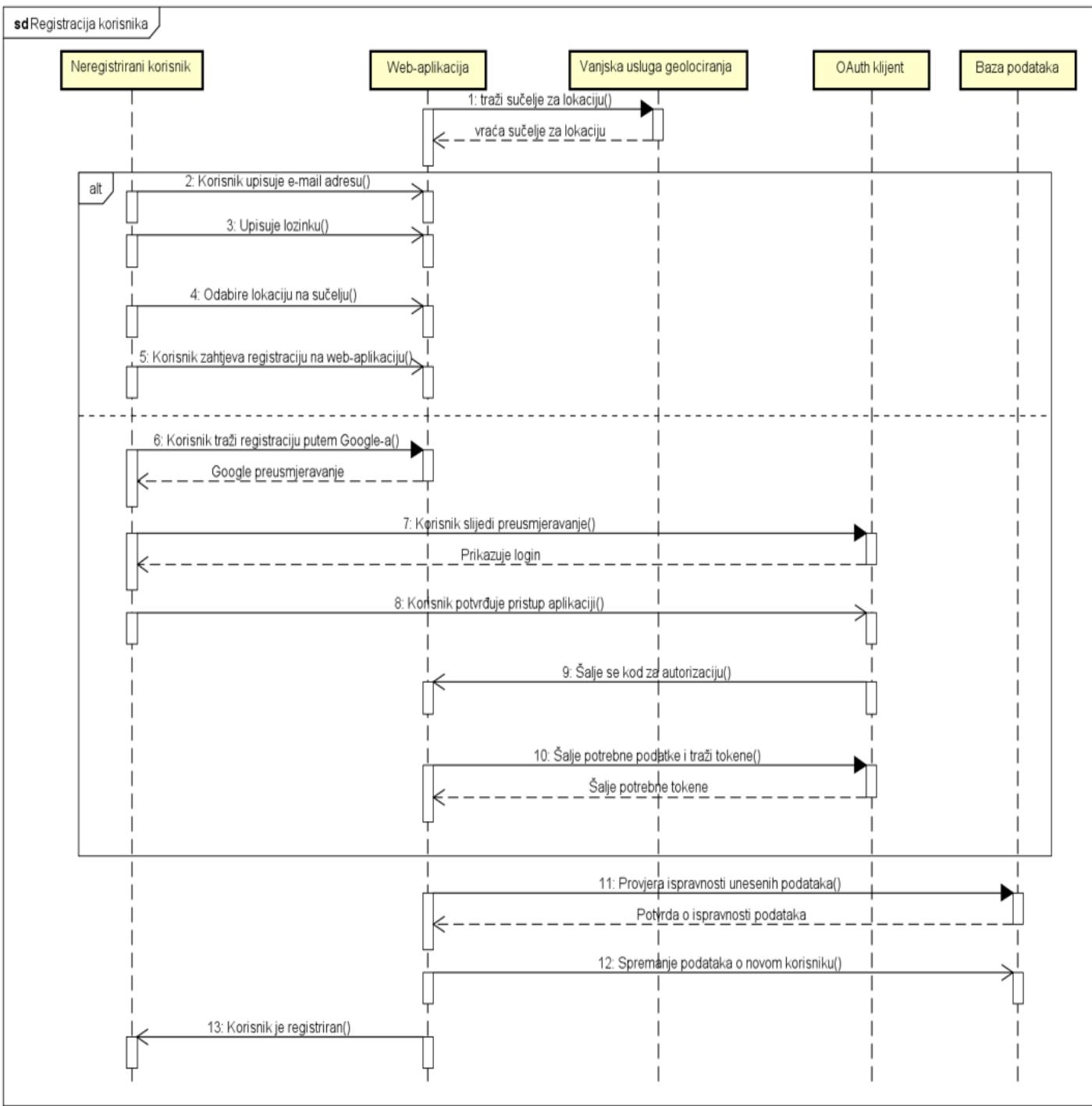
Slika 3.10 Sekvencijski dijagram dohvata svih objava



Slika 3.11 Sekvencijski dijagram uspješne prijave korisnika



Slika 3.12 Sekvencijski dijagram uspješne registracije korisnika



Arhitektura sustava

Stil arhitekture

Za ovaj je sustav odabrana **klijent-poslužitelj** arhitektura u kojoj su frontend i backend jasno odvojeni i međusobno komuniciraju putem RESTful API-ja.

Frontend je implementiran u Reactu, dok je backend razvijen u Node.js okruženju koristeći Express okvir.

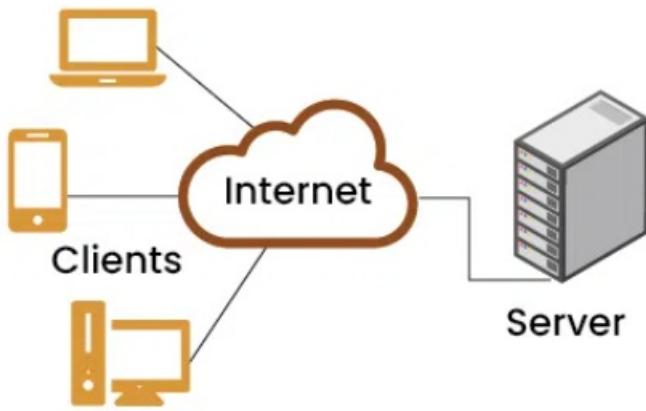
Podatci se razmjenjuju u JSON formatu putem HTTP protokola.

Ovaj arhitektonski stil odabran je jer omogućuje:

- odvajanje odgovornosti između korisničkog sučelja i poslovne logike
- neovisnost razvoja frontend i backend dijelova
- fleksibilnost i skalabilnost sustava
- jednostavno održavanje i buduće proširenje funkcionalnosti

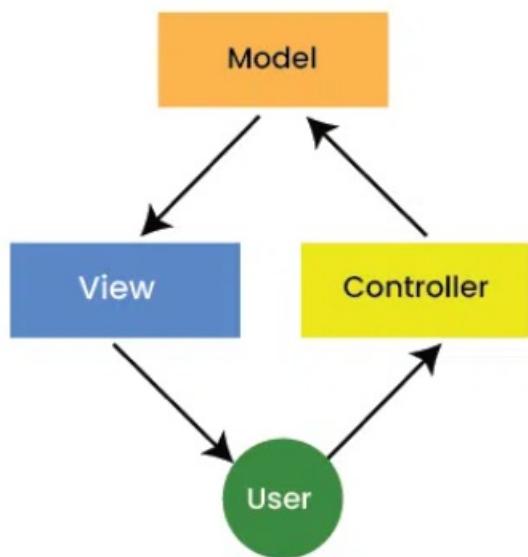
Klijent-poslužitelj arhitektura je najprikladnija jer sustav zahtjeva interakciju više korisnika u stvarnom vremenu te pouzdan prijenos podataka između različitih komponenti (frontend, backend, baza podataka).

Slika 4.1 Klijent-poslužitelj arhitektura



Osim toga, sustav slijedi i načela MVC arhitekture, u kojoj je klijent (frontend) odgovoran za View sloj, dok je poslužitelj (backend) implementiran kroz Model i Controller sloj.
Ovakav pristup omogućuje jasnu podjelu između prikaza, poslovne logike i upravljanja podacima, čime se dodatno povećava čitljivost i održivost koda. Naš model stoga kombinira klijent-poslužitelj stil i MVC obrazac, gdje React predstavlja View razinu, dok Node.js/Express backend sadrži Model (rad s bazom podataka) i Controller (obrada zahtjeva).

Slika 4.2 MVC arhitektura

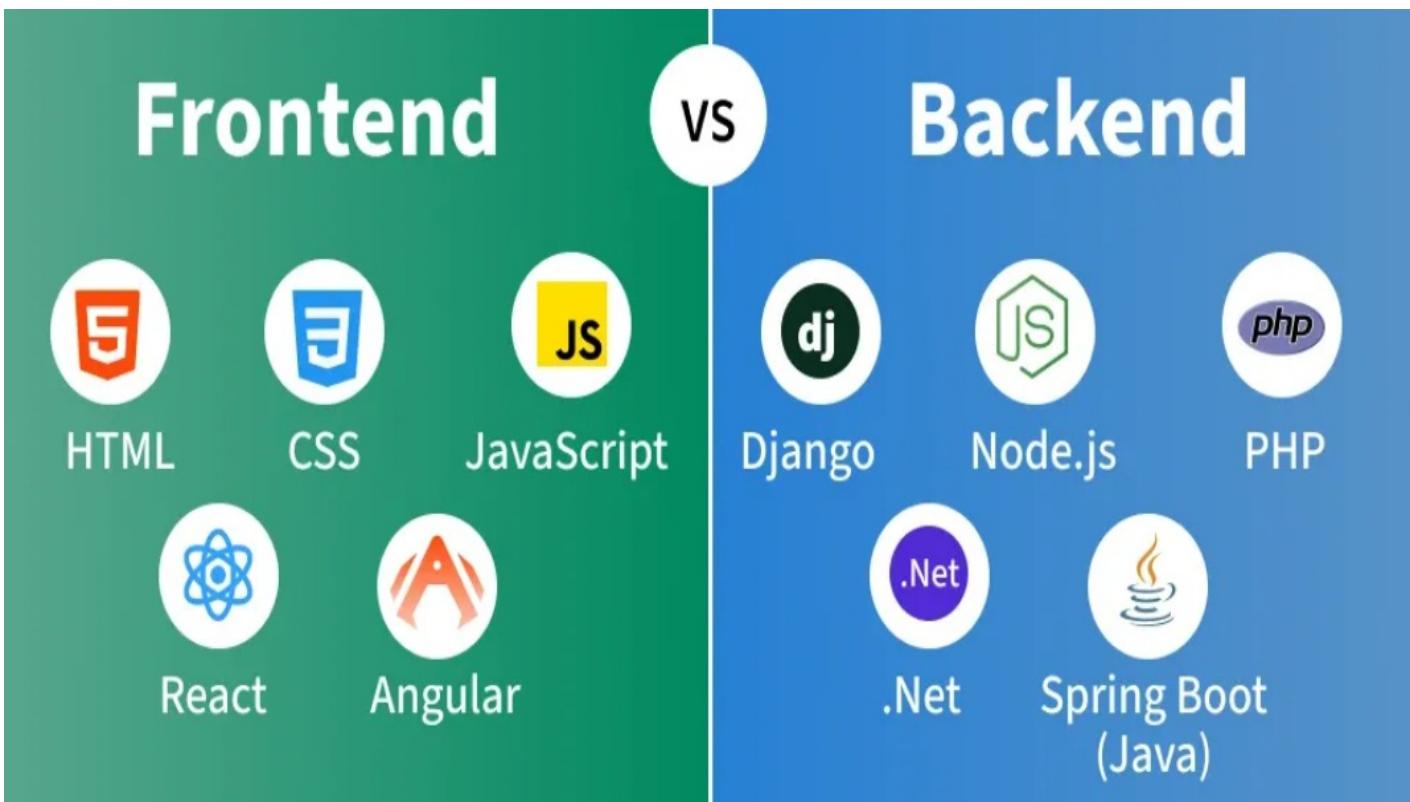


Podsustavi

Sustav je podijeljen u tri glavna podsustava:

1. Frontend podsustav (React)
 - odgovoran za prikaz korisničkog sučelja i interakciju s korisnikom
 - koristi HTTP zahtjeve za komunikaciju s backendom
 - implementira logiku prikaza, navigaciju i prikaz povratnih informacija
 - responsivan dizajn omogućuje i korištenje aplikacije na mobilnim uređajima
2. Backend podsustav (Node.js/Express)
 - implementira poslovnu logiku sustava i upravlja zahtjevima koje šalje frontend
 - koristi RESTful API za pružanje podataka i funkcionalnosti
 - obrada korisničkih zahtjeva uključuje autentifikaciju (OAuth 2.0), upravljanje korisnicima, igrama, ponudama i zamjenama
3. Baza podataka (MongoDB)
 - NoSQL baza podataka namijenjena za pohranu korisničkih podataka, oglasa igara, ponuda za zamjenu i arhive zamjena
 - podatci se pohranjuju u JSON-like dokumentima što omogućuje fleksibilnost i brzo dohvaćanje podataka

Slika 4.3 Prikaz alata često korištenih u razvoju frontenda i backenda



Preslikavanje na radnu platformu

Cijeli sustav bit će hostan na Microsoft Azure cloud platformi.

Frontend će biti distribuiran putem Azure Static Web Apps servisa, dok će backend biti implementiran i pokretan u Azure App Service okruženju.

Odabir Azure-a temelji se na:

- pouzdanosti i visokoj dostupnosti cloud arhitekture
- podršci za Node.js i React
- jednostavnoj integraciji OAuth 2.0 autentifikacije
- mogućnosti automatskog skaliranja i nadzora performansi

Spremišta podataka

Za pohranu podataka koristi se NoSQL baza podataka MongoDB.

MongoDB omogućuje fleksibilno modeliranje podataka pomoću JSON dokumenta, što olakšava kompleksnih struktura poput oglasa, ponuda i korisničkih profila.

Mrežni protokoli

Komunikacija između elemenata podsustava odvija se putem **HTTP(S)** protokola.

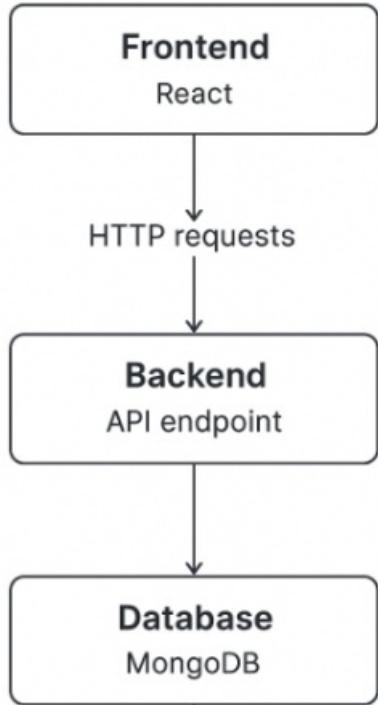
Frontend šalje zahtjeve prema backendu (npr. **GET**), a backend odgovara JSON objektima.

Sigurna autentifikacija korisnika odvija se putem **OAuth 2.0** protokola korištenjem tokena.

Globalni upravljački tok

1. Korisnik pristupa aplikaciji korištenjem preglednika (frontend)
2. React aplikacija šalje HTTP zahtjeve prema backendu
3. Backend provjerava korisničku autentifikaciju i obrađuje zahtjev prema odgovarajućem API endpointu
4. Backend komunicira s bazom podataka (MongoDB) i dohvaca ili mijenja potrebne podatke
5. Rezultat (u JSON formatu) se vraća frontend dijelu, koji ažurira prikaz korisniku
6. Ciklus se ponavlja ovisno o interakciji korisnika (npr. pregled oglasa, ponuda zamjene)

Slika 4.4 Pojednostavljeni prikaz upravljačkog toka



Sklopoškoprogramski zahtjevi

Frontend: podrška za moderne preglednike (Chrome, Edge, Firefox, Safari)

Backend: Node.js okruženje s Express frameworkom

Baza podataka: MongoDB Atlas instance

Operativni sustav: Windows, Linux ili macOS; Azure cloud za produkcijsko okruženje

Minimalni resursi (procjena): 4GB RAM-a za backend; 512MB RAM-a za frontend host

Obrazloženje odabira arhitekture

Odabir klijent-poslužitelj arhitekture temelji se na potrebi za **modularnim, skalabilnim i distribuiranim sustavom** koji omogućuje razvoj više komponenti paralelno.

Ključni razlog zbog kojeg smo se odlučili za zadanu arhitekturu jest: frontend i backend su **neovisni**, što olakšava timski rad i testiranje. Ali, osim toga, sustav temeljen na klijent-poslužitelj arhitekturi jest lakši za održavanje i upravljanje. Međutim morali smo napraviti kompromis što se tiče rizika, jer pad poslužitelja može paralizirati sustav, ali može doći i do zagruženja sustava jer se sav promet odvija kroz poslužitelja. Iz tog razloga smo se potrudili te rizike svesti na minimum. Prednosti odabrane arhitekture:

- visoka kohezija i niska povezanost između sustava
- skalabilnost - omogućuje horizontalno proširenje sustava dodavanjem novih servera
- modularnost - svaki dio sustava se može proširivati bez utjecaja na druge

Izbor arhitekture temeljen na principima oblikovanja

Principi koji pomažu u ostvarenju ciljeva projekta i način na koji pomažu:

- visoka **kohezija** i niska povezanost između sustava
- **fleksibilnost** u razvoju i održavanju
- **skalabilnost** - omogućuje horizontalno proširenje sustava dodavanjem novih servera
- **sigurnost** kroz OAuth 2.0 autentifikaciju i HTTPS komunikaciju
- **modularnost** - svaki dio sustava se može proširivati bez utjecaja na druge

Razmatrane alternative

Razmatrane alternative:

- **Monolitna arhitektura:** odbijena, jer bi otežavala održavanje i skaliranje zbog spajanja frontenda i backenda u jedinstvenu aplikaciju
- **Mikroslužba arhitektura:** odbačena zbog kompleksnosti i ograničenih resursa projekta; nije potrebna za trenutni opseg funkcionalnosti

Organizacija sustava na visokoj razini

1. Klijent-poslužitelj
 - React frontend (klijent) šalje zahtjeve Node.js backendu (poslužitelju), koji obrađuje podatke i komunicira s MongoDB bazom
2. Baza podataka
 - MongoDB kao NoSQL baza podataka omogućuje jednostavno dohvaćanje podataka po ključu i fleksibilnu pohranu

3. Grafičko sučelje:

- Web aplikacija izrađena u Reactu, responzivna i optimizirana za mobilne uređaje

Organizacija aplikacije

Aplikacija je organizirana prema klijent-poslužitelj arhitekturi, s jasnim razdvajanjem između frontend i backend slojeva. Ovakva struktura omogućuje neovisni razvoj, održavanje i skaliranje svake komponente sustava. Komunikacija između slojeva odvija se putem RESTful API-ja uz razmjenu podataka u JSON formatu.

Frontend sloj (React)

Frontend podsustav implementiran je u React okruženju i predstavlja korisnički prikaz aplikacije (engl. View layer). Njegove su osnovne odgovornosti:

- prikaz korisničkog sučelja i rukovanje interakcijama korisnika
- slanje HTTP zahtjeva prema backendu putem REST API-ja
- prikaz rezultata i povratnih informacija u stvarnom vremenu
- osiguravanje responzivnosti i prilagodljivosti sučelja na različitim uređajima

Frontend je organiziran prema komponentnom pristupu, gdje je svaka komponenta zadužena za određeni dio sučelja (npr. prikaz profila, oglas, ponuda zamjene). Takva struktura povećava ponovnu iskoristivost koda i olakšava održavanje. Za upravljanje stanjem aplikacije koristi se React-ov kontekstni sustav i lokalni storage, čime se omogućuje sinkronizacija podataka između različitih dijelova aplikacije.

Backend sloj (Node.js)

Backend podsustav razvijen je u Node.js okruženju koristeći Express framework. Njegova je primarna uloga implementacija poslovne logike i upravljanje zahtjevima koje dolaze s frontend sloja. Backend:

- prima i obrađuje HTTP zahtjeve korisnika
- provodi autentifikaciju i autorizaciju pomoću OAuth 2.0 protokola
- komunicira s bazom podataka (MongoDB) kroz modelne slojeve
- vraća odgovore u JSON formatu prema frontend aplikaciji

Ovaj sloj je organiziran prema MVC arhitekturnom obrascu, gdje se odgovornosti dijele na tri razine:

- Model (M) – predstavlja strukturu podataka i povezan je s bazom podataka. Modeli definiraju entitete poput korisnika, igara, oglasa i ponuda te pružaju metode za njihovo dohvaćanje, spremanje i ažuriranje
- View (V) – u našem slučaju, View sloj fizički se nalazi u frontend dijelu aplikacije (React), koji prikazuje podatke korisniku
- Controller (C) – sadrži logiku obrade zahtjeva. Controlleri primaju zahtjeve od frontenda, koriste modele za pristup podacima i vraćaju odgovarajuće odgovore prema View sloju

Baza podataka

Za pohranu podataka koristi se MongoDB, NoSQL baza podataka koja pohranjuje podatke u JSON-like dokumentima. Prednosti ovog pristupa su:

- fleksibilna struktura podataka
- brzo dohvaćanje i filtriranje
- lako skaliranje baze podataka

U bazi se pohranjuju podaci o korisnicima, oglasima, igram, ponudama zamjene i povijesti izvršenih transakcija.

Interakcija slojeva

1. Korisnik putem preglednika pristupa frontend aplikaciji (React).
2. Frontend šalje REST zahtjeve (GET, POST, PUT, DELETE) prema backendu.
3. Backend (Express) prima zahtjev, provjerava korisničke vjerodajnice (OAuth 2.0) i proslijeđuje zahtjev odgovarajućem kontroleru.
4. Kontroler koristi modele za komunikaciju s MongoDB bazom podataka.
5. Dobiveni podaci vraćaju se kao JSON objekt prema frontendu.
6. Frontend ažurira prikaz korisničkog sučelja prema dobivenim podacima.

Ovaj tok omogućuje jasno odvajanje odgovornosti i brzu dvosmjernu komunikaciju između korisničkog sučelja i poslužiteljske logike.

Zašto MVC u backendu

Backend dio arhitekture slijedi MVC arhitekturu jer:

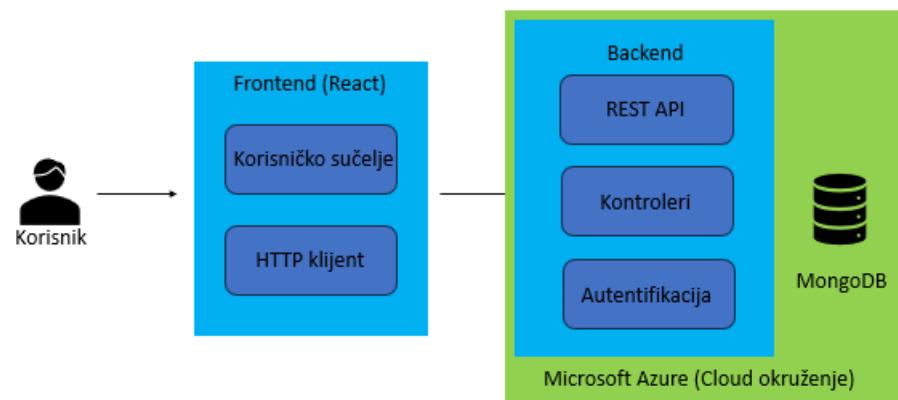
- Model sloj sadrži definicije podataka i interakcije s bazom (MongoDB kolekcije i Mongoose modeli),
- Controller sloj upravlja zahtjevima, validira podatke i kontrolira tok između Modela i frontend-a,
- View sloj je vanjski (React aplikacija), što znači da je vizualna prezentacija fizički odvojena od poslužiteljske logike.

Takva organizacija omogućuje:

- jednostavno testiranje i ponovno korištenje kontrolera i modela
- izolaciju grešaka
- olakšanu nadogradnju funkcionalnosti bez utjecaja na druge dijelove sustava

Na taj način sustav kombinira klijent-poslužitelj arhitekturu s MVC obrascem

Dijagram visoke razine (skica)



Slika 4.5 Dijagram visoke razine (skica)

Reference

- Programsko inženjerstvo, FER ZEMRIS, <http://www.fer.hr/predmet/proinz>
- Client-Server Architecture – System Design | GeeksforGeeks. Dostupno: <https://www.geeksforgeeks.org/system-design/client-server-architecture-system-design/> (10. studeni 2025.).
- MVC Architecture – System Design | GeeksforGeeks. Dostupno: <https://www.geeksforgeeks.org/system-design/mvc-architecture-system-design/> (10. studeni 2025.).
- Frontend vs Back-end Development | GeeksforGeeks. Dostupno: <https://www.geeksforgeeks.org/blogs/frontend-vs-backend/> (10. studeni 2025.).
- MongoDB Documentation. Dostupno: <https://www.mongodb.com/docs/> (7. studeni 2025.).
- React – Learn React. Dostupno: <https://react.dev/learn> (7. studeni 2025.).
- Node.js – Introduction to Node.js. Dostupno: <https://nodejs.org/en/learn/getting-started/introduction-to-nodejs> (7. studeni 2025.).

Baza podataka

Odabrali smo MongoDB kao sustav za upravljanje bazom podataka jer se izvrsno uklapa s našim Node.js backendom, budući da koristi BSON (Binary JSON) dokumente čija je struktura vrlo slična JSON formatu s kojim Node.js prirodno radi.

MongoDB je NoSQL dokumentno orijentirani sustav, što znači da podatke pohranjuje u obliku fleksibilnih dokumenata unutar kolekcija, a ne u tablicama kao relacijske baze. Svaki dokument može sadržavati različita polja i ugniježđene strukture, što omogućuje veću prilagodljivost u radu s dinamičnim podacima.

Za razliku od relacijskih baza podataka, MongoDB ne koristi primarne i strane ključeve niti stroge relacijske veze između entiteta. Umjesto toga, veze između kolekcija ostvaruju se putem referenci (ObjectId) ili ugnježđivanjem dokumenata, ovisno o potrebama sustava.

Ovakav pristup omogućuje brže dohvaćanje podataka, jednostavnije skaliranje i lakšu prilagodbu strukture baze tijekom razvoja aplikacije.

Opis tablica

users

Atribut	Tip podatka	Opis variabile
_id	ObjectId	Jedinstveni identifikator
email	String	Email korisnika
passwordHash	String	Hashirana lozinka
username	String	Username korisnika
scope	String	Uloga korisnika
location	Object	Objekt za pohranu lokacije
profile	Object	Objekt za pohranu detalja profila
createdAt	Date	Datum registracije korisnika
isActive	Boolean	Označava je li korisnik aktivan

Kolekcija users sadrži sve registrirane korisnike aplikacije. Svaki korisnik ima osnovne podatke, kategorije igara koje ga interesiraju koje se nalaze unutar objekta profile, unutar objekta profile se još nalazi opis koji korisnik može napisati i slika koju može objaviti te korisnik ima lokaciju dobivenu putem vanjske geolokacijske usluge (OpenStreetMap) čiji se detalji nalaze unutar objekta location.

games

Atribut	Tip podatka	Opis varijable
_id	ObjectId	Jedinstveni identifikator
name	String	Naziv igre koju je korisnik objavio
genre	String	Naziv žanra igre
publisher	String	Naziv izdavača igre
releaseYear	Int	Godina izlaska igre
minPlayers	Int	Najmanji broj igrača za igranje igre.
maxPlayers	Int	Najveći broj igrača za igranje igre.
playtime	Int	Duljina prosječne partije
difficulty	Int	Težina igre
imageUrl	String	Slika igre

Kolekcija games predstavlja sve igre koje korisnici objave kao dostupne za razmjenu.

listings

Atribut	Tip podatka	Opis varijable
_id	ObjectId	Jedinstveni identifikator
userId	ObjectId	Referenca na korisnika koji postavlja igru na web-stranicu
gameId	ObjectId	Referenca na igru koju je korisnik postavio u oglasu
condition	String	Ocjena očuvanosti igre
available	Boolean	Je li igra dostupna za zamjenu
description	String	Dodatni opis oglasa
createdAt	Date	Datum objave oglasa

Kolekcija listings predstavlja oglase koje korisnici objavljaju za igre koje žele zamijeniti. Svaki oglas je povezan s korisnikom i određenom igrom.

trades

Atribut	Tip podatka	Opis varijable
_id	ObjectId	Jedinstveni identifikator
initiatorId	ObjectId	Referenca na korisnika koji nudi zamjenu
receiverId	ObjectId	Referenca na korisnika koji prima ponudu
offeredListing	Array	Oglas koji korisnik nudi u zamjeni
requestedListing	Array	Oglas koji korisnik traži u zamjeni
status	String	Stanje zamjene
messages	Array	Poruke koje korisnik šalje

createdAt	Date	Tip podatka	Datum inicijacije zamjene
Atribut			Opis varijable

Kolekcija trades bilježi sve zamjene između korisnika. Svaka zamjena ima inicijatora, primatelja, listu ponuđenih i traženih oglasa te trenutni status. U messages se nalaze objekti koji u sebi sadrže senderId, text i vrijeme slanja poruke.

wishlist

Atribut	Tip podatka	Opis varijable
_id	ObjectId	Jedinstveni identifikator
userId	ObjectId	Referenca na korisnika koji je kreirao svoj wishlist
games	Array	Sve igre koje korisnik želi imati
notificationsEnabled	Boolean	Ima li korisnik uključene notifikacije

Kolekcija wishlist sadrži popise željenih igara svakog korisnika. Koristi se za obavještanje korisnika kada neka od željenih igara postane dostupna u oglasima.

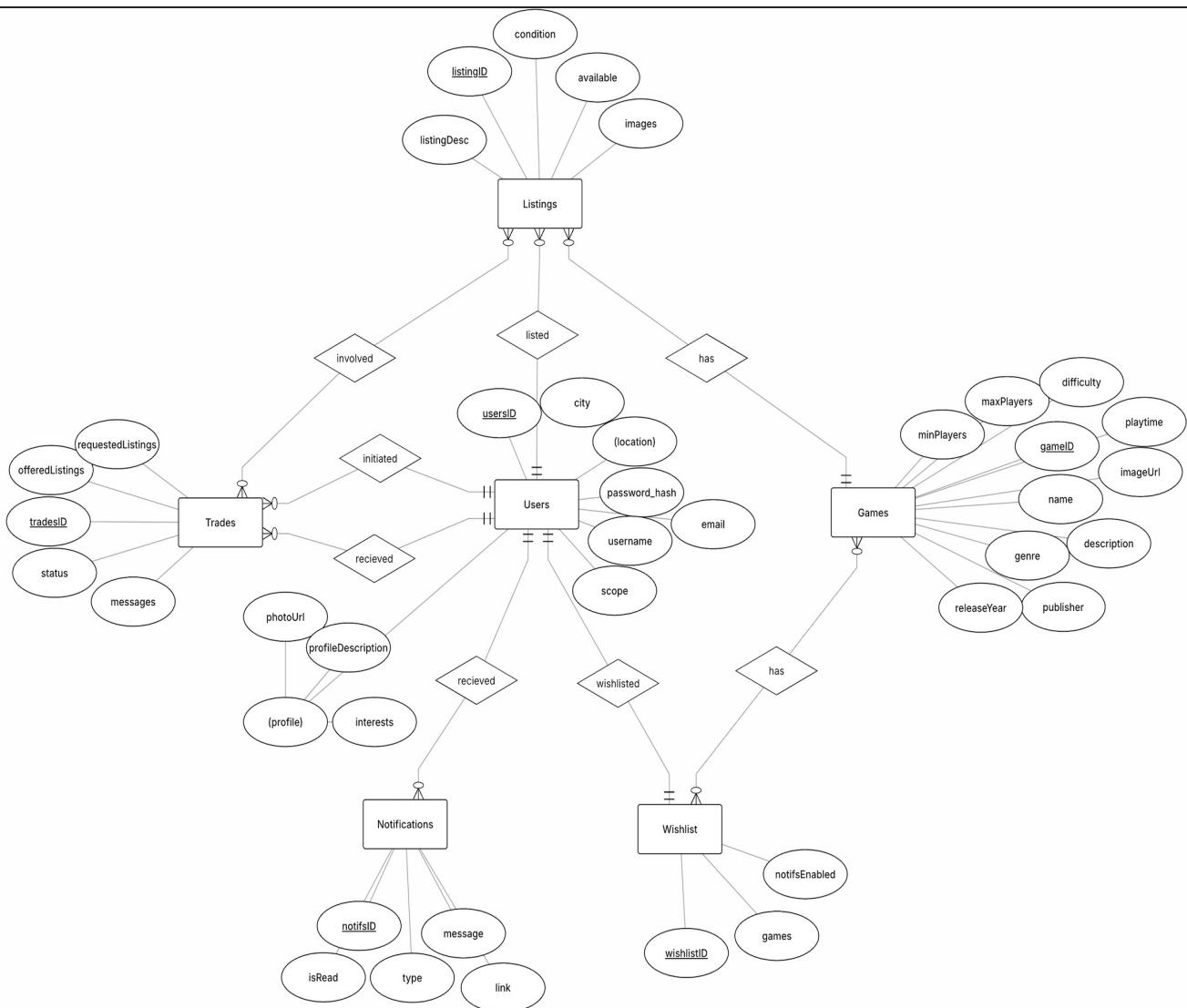
notifications

Atribut	Tip podatka	Opis varijable
_id	ObjectId	Jedinstveni identifikator
userId	ObjectId	Referenca na korisnika koji prima notifikaciju
type	String	Vrsta notifikacije
message	String	Sadržaj notifikacije
link	String	link notifikacije
isRead	Boolean	Je li notifikacija pročitana
createdAt	Date	Datum primanja notifikacije

Kolekcija notifications pohranjuje sve obavijesti koje korisnici primaju (npr. o novim ponudama, odgovorima na zamjenu ili dostupnim igramu s wishliste).

Dijagram baze podataka

Slika 4.6 Dijagram baze podataka



Budući da je baza podataka izrađena u NoSQL sustavu, ER dijagram nema izravnu funkcionalnu ulogu. Ipak, smatramo da ga je korisno priložiti kako bi se jasno prikazale sve kolekcije i odnosi među njima.

Dijagram razreda

Slika 4.7 Dijagram razreda [] Napomena: zbog veličine dijagrama tip slike je svg format. Da bi se mogla vidjeti detalji u dijagramu razreda potrebno je otvoriti sluku direktno(te proizvoljno smanjiti veličinu) preko linka: [Dijagram razreda](#)

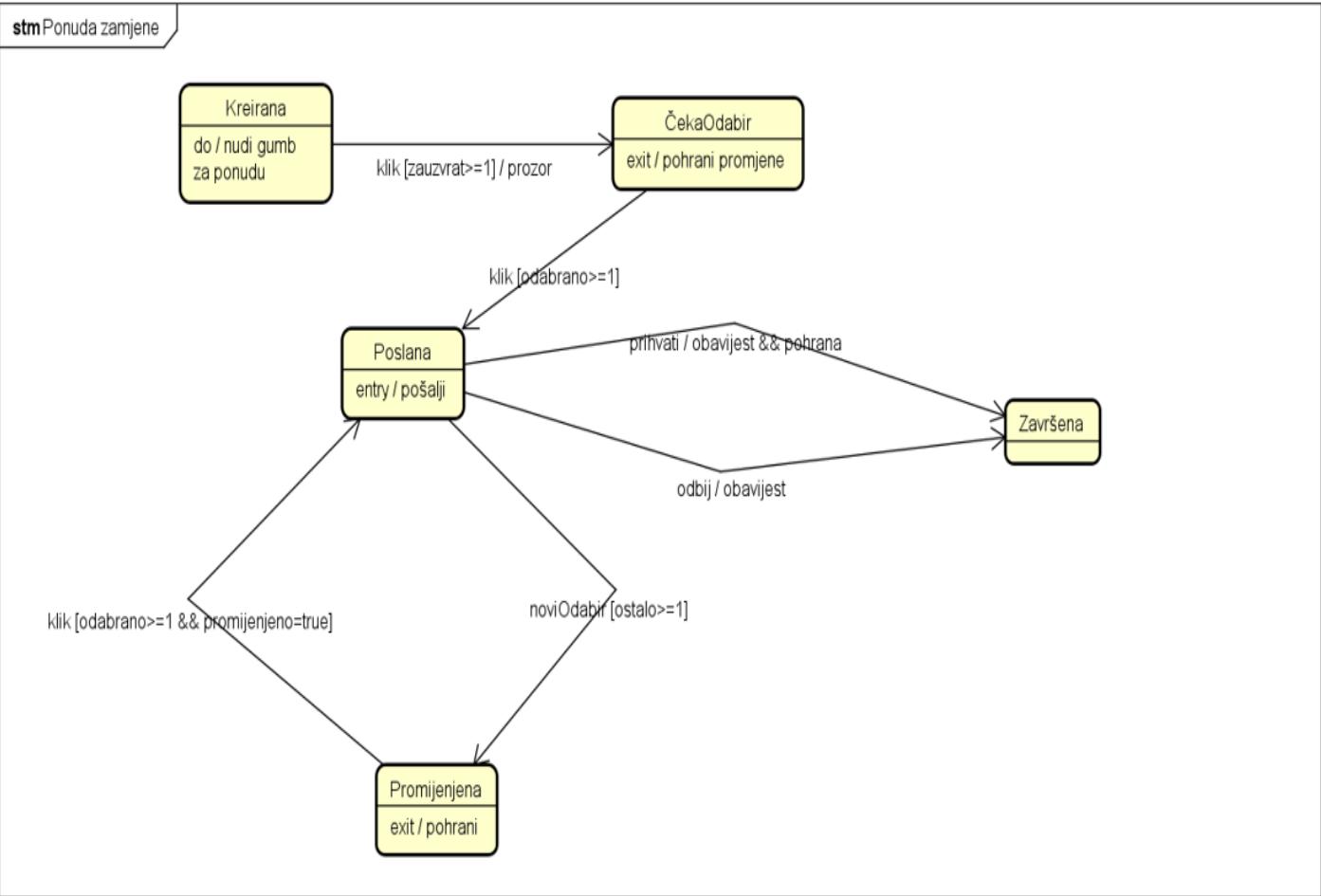
Dinamičko ponašanje aplikacije

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.

U nastavku su prikazani dijagrami stanja i dijagrami aktivnosti koji opisuju pojedine elemente sustava tj njihovu međusobnu dinamiku.

UML dijagrami stanja

Slika 4.8 Dijagram stanja za proces ponude zamjene



Stanja :

- Kreirana - objavu igre kreirao je neki korisnik i ona je vidljiva drugim korisnicima i spremna za ponudu (POČETNO STANJE)
- ČekaOdabir - u prozoru za ponude korisnik koji traži zamjenu odabire što daje zauzvrat za traženu igru
- Poslana - ponuda je poslana prema nekom korisniku, čeka se njegova reakcija na nju
- Promijenjena - neki korisnik je u procesu mijenjanja uvjeta zamjene
- Završena - jedan od korisnika je konačno odbio ili prihvatio ponudu (KONAČNO STANJE)

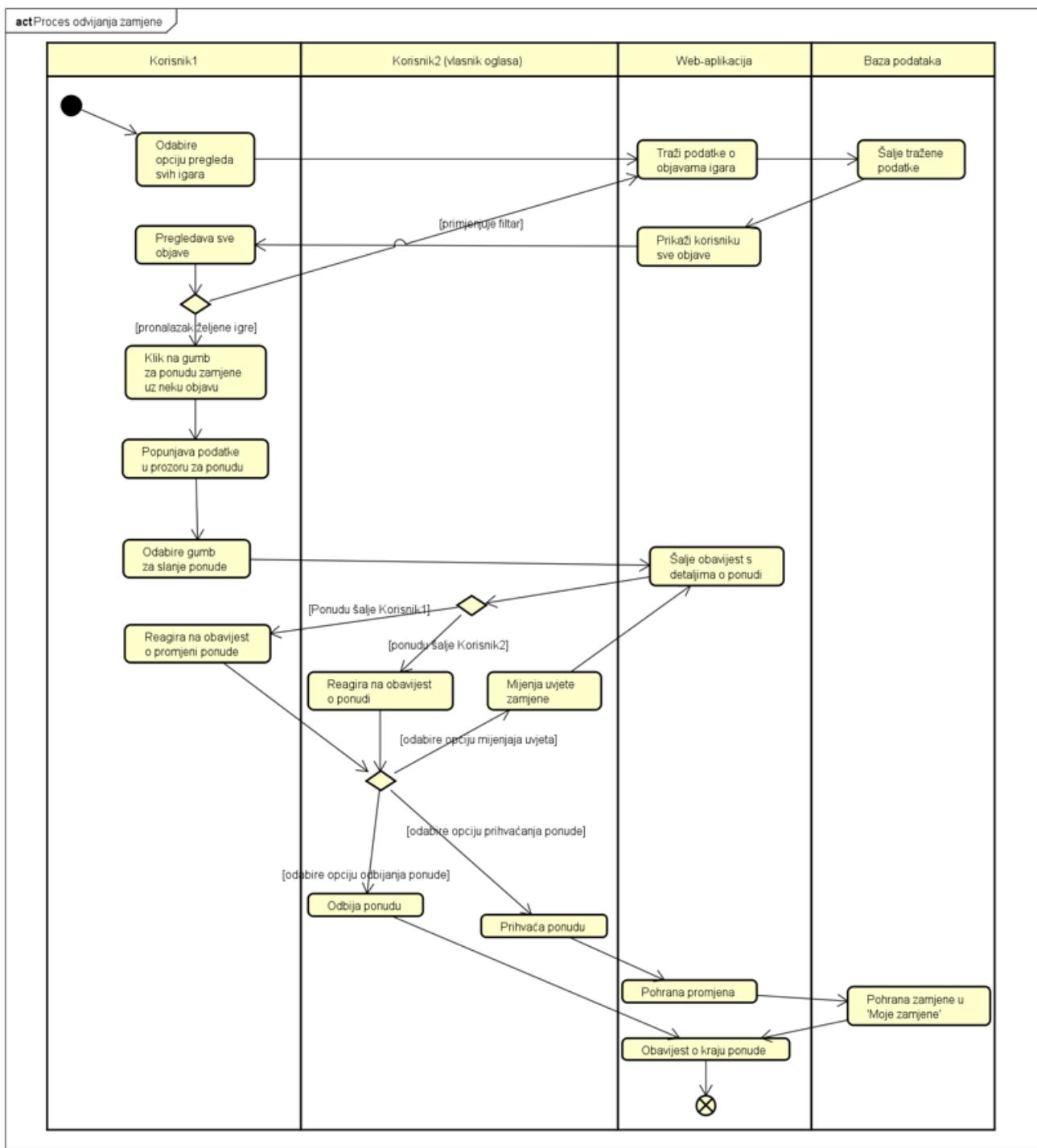
Entry/ Do/ Exit :

- nudi gumb za ponudu - nećija objava kraj sebe ima gumb kojim ostali korisnici mogu zatražiti zamjenu
- pohrani promjene - podatci o ponudi spremaju se u bazu podataka
- pošalji - korisniku s druge strane ponude šalje se obavijest o detaljima ponude na stranicu i na e-mail
- pohrani - ažuriraju se podatci o promjeni uvjeta zamjene

Prijelazi

- klik [zauzvrat>=1] / prozor - prijelaz se pokreće klikom na gumb za davanje ponude, prijelaz se odvija samo ako ponuditelj ima barem jednu svoju objavljenu igru koju može nuditi za zamjenu, akcija je otvaranje prozora za ponudu* klik [odabran>=1] - prijelaz se pokreće kad se klikne na gumb za slanje ponude, a odvija se samo ako je korisnik odabrao barem jednu svoju igru koju nudi za zamjenu
- noviOdabir [ostalo>=1] - prijelaz se pokreće kad korisnik zatraži promjenu uvjeta zamjene, a odvija se samo ako korisnik s druge strane zamjene ima više od jedne objavljene igre, odnosno, ako se ima išta za mijenjati u uvjetima zamjene
- klik [odabran>=1 && promijenjeno=true] - prijelaz se pokreće kad korisnik klikne na gumb za slanje ponude s novim uvjetima zamjene, a prijelaz se odvija samo ako su uvjeti zamjene mijenjani na neki način i ako broj igara koji se traži nije jednak 0
- odbij / obavijest - prijelaz se pokreće kad jedan od korisnika odbije ponudu zamjene, akcija je slanje obavijesti o odbijanju drugoj strani zamjene o odbijanju ponude
- prihvati / obavijest && pohrana - prijelaz se pokreće kad jedan od korisnika prihvati ponudu zamjene, akcije su slanje obavijesti drugoj strani zamjene o prihvatanju ponude i pohrana uspješne zamjene u kategoriju 'Moje zamjene' svakog korisnika

UML dijagrami aktivnosti



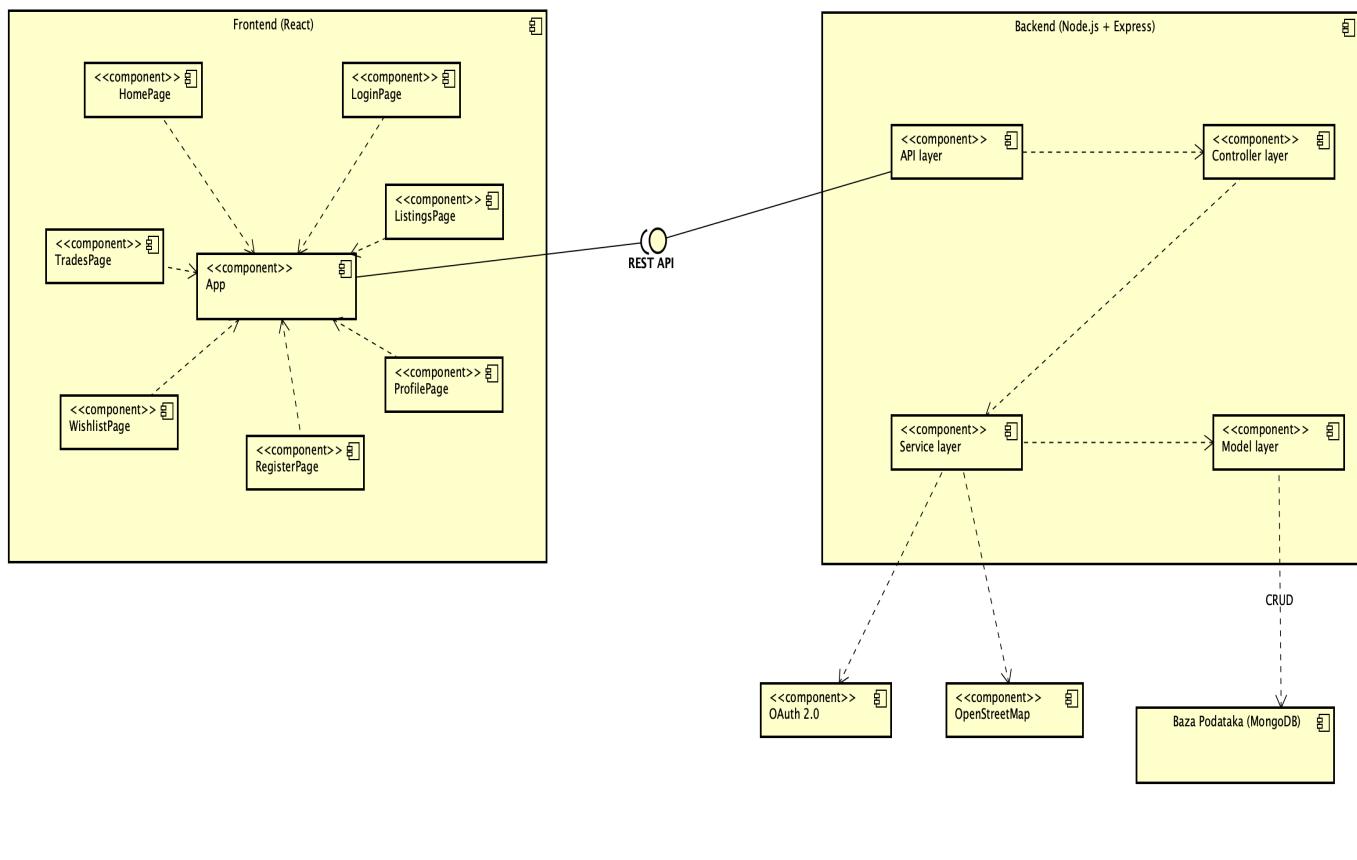
Particije :

- Korisnik1 - onaj korisnik sustava koji prvi daje ponudu za neku objavu
 - Korisnik2 - onaj korisnik koji je vlasnik oglasa za kojeg Korisnik1 daje ponudu
 - Web-aplikacija - PlayTrade
 - Baza podataka - zadužena za spremanje i pružanje svih podataka sustava

Opis tijeka aktivnosti :

1. Korisnik1 pristupa sustavu i zahtijeva pristup svim objavama drugih korisnika (prepostavka je da je registriran i prijavljen)
 2. Web-aplikacija od baze podataka traži i dobije podatke o svim objavama i pritom ih prikaze korisniku
 3. Korisnik1 pregledava sve dobivene igre ili primjenjuje filter prilikom čega se proces ponavlja od točke 2.
 4. Korisnik1 pronađe objavu igre koju želi dobiti zamjenom i odabire opciju za davanje ponude vlasniku te objave (Korisnik2)
 5. Korisnik1 odabire koje svoje igre daje zauzvrat u zamjenu za traženu
 6. Korisnik1 potvrđuje slanje ponude
 7. Web-aplikacija šalje obavijest o ponudi drugoj strani ponude
 8. Druga strana ponude reagira na ponudu na jedan od tri načina: prihvata ju (nastavak na korak 9.), odbija ju (skok na korak 10.), ili mijenja uvjete ponude (povratak na korak 7.)
 9. Ako je ponuda prihvaćena web-aplikacija sprema u bazu podataka podatke o izvršenoj zamjeni (u kategoriji 'Moje zamjene' obaju korisnika)
 10. Šalje se obavijest o zatvaranju ponude

Dijagram komponenata



Dijagram komponenata prikazuje logičku strukturu aplikacije kroz skup međusobno povezanih funkcionalnih komponenti. Svaka komponenta predstavlja zasebnu cjelinu s jasno definiranim odgovornostima, dok se komunikacija između komponenti odvija putem definiranih sučelja.

Frontend sloj (React)

Frontend dio aplikacije implementiran je pomoću React biblioteke te je organiziran oko središnje komponente App, koja služi kao glavna ulazna točka aplikacije i upravlja navigacijom između pojedinih stranica.

Unutar frontend sloja nalaze se sljedeće komponente: HomePage, LoginPage, RegisterPage, ListingsPage, TradesPage, WishlistPage, ProfilePage.

Navedene komponente predstavljaju pojedine stranice aplikacije te su ovisne o komponenti App, koja koordinira njihovo prikazivanje i upravljanje stanjem aplikacije. Frontend komunicira s backend dijelom sustava putem REST API sučelja, koristeći HTTP(S) protokol.

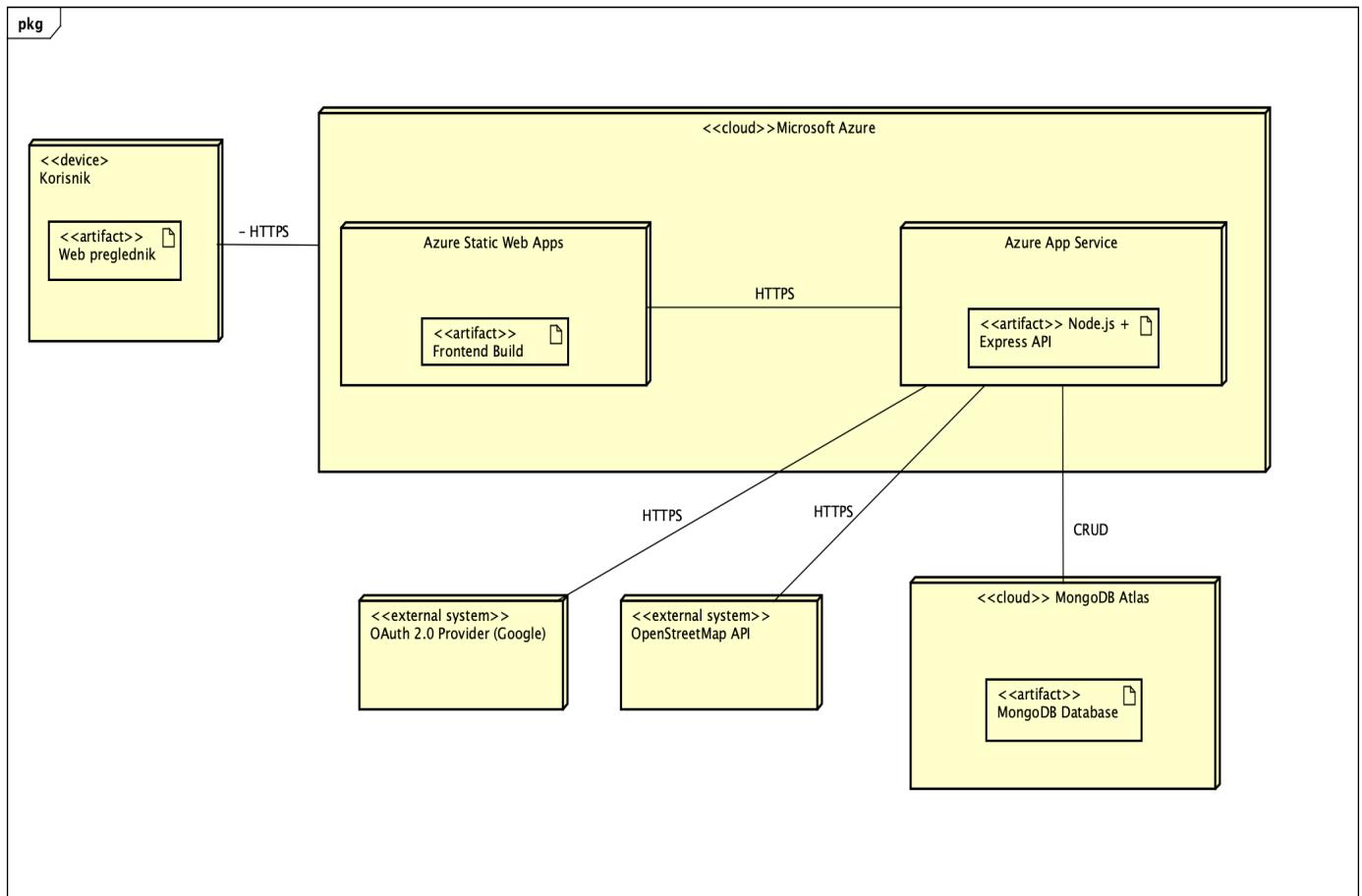
Backend sloj (Node.js + Express)

Backend dio aplikacije implementiran je korištenjem Node.js platforme i Express okvira te je organiziran prema slojevitoj arhitekturi.

Backend se sastoji od sljedećih komponenti: API / Routes sloj, koji definira dostupne REST rute i prima zahteve s frontend strane, Controller sloj, koji obrađuje dolazne zahteve i koordinira daljnju logiku obrade, Service sloj, koji sadrži poslovnu logiku aplikacije, Model sloj, koji predstavlja podatkovni sloj aplikacije i omogućuje pristup bazi podataka.

Service sloj koristi vanjske servise poput OAuth 2.0 sustava za autentikaciju korisnika te OpenStreetMap API-ja za dohvati i obradu geografskih podataka. Model sloj zadužen je za trajnu pohranu podataka i komunikaciju s bazom podataka MongoDB.

Dijagram razmještaja



Dijagram razmještaja prikazuje fizičku i virtualnu raspodjelu komponenti sustava unutar infrastrukturnog okruženja. Dijagram je izrađen u implementacijskom obliku te prikazuje stvarne servise, čvorove i artefakte na kojima se aplikacija izvršava.

Klijentski sloj

Korisnik pristupa aplikaciji putem uređaja (npr. osobnog računala ili mobilnog uređaja) koristeći web preglednik. Komunikacija između korisnika i frontend dijela aplikacije odvija se putem sigurnog HTTPS protokola.

Cloud infrastruktura – Microsoft Azure

Aplikacija je implementirana unutar Microsoft Azure okruženja, koje sadrži sljedeće servise: Azure Static Web Apps, koji hosta frontend dio aplikacije. Na ovom servisu nalazi se izgrađeni React frontend (Frontend Build) koji se isporučuje korisnicima. Azure App Service, na kojem je implementiran backend aplikacije u obliku Node.js + Express API-ja. Ovaj servis obrađuje dolazne zahtjeve, provodi poslovnu logiku i upravlja pristupom podacima.

Frontend i backend komuniciraju međusobno putem HTTPS REST API poziva.

Baza podataka – MongoDB Atlas

Za trajnu pohranu podataka koristi se MongoDB Atlas, koji predstavlja cloud bazu podataka izvan Azure okruženja. Backend aplikacija komunicira s bazom podataka putem MongoDB drivera, koristeći CRUD operacije za upravljanje podacima.

Vanjski servisi

Sustav koristi i vanjske servise koji nisu dio glavne infrastrukture: OAuth 2.0 Provider, koji omogućuje autentikaciju i autorizaciju korisnika putem sigurnog protokola i OpenStreetMap API, koji se koristi za dohvrat geografskih i kartografskih podataka.

Komunikacija s navedenim servisima odvija se putem HTTPS protokola, a integracija je ostvarena unutar backend sloja aplikacije. # Ispitivanje komponenti Ovo poglavљje treba opisati provedena ispitivanja implementiranih funkcionalnosti na razini komponenti i sustava. Fokus je na odabiru i izvedbi ispitnih slučajeva koji obuhvaćaju redovne, rubne uvjete i testiranje grešaka, kao i upotrebu odgovarajućih alata za provedbu testiranja.

Postupak testiranja provodi se Jest-om. Test se pokreće nardbom npm test.

```
C:\PROGI\error808\backend>npm test
> error808-backend@0.0.0 test
> jest

PASS  tests/controllers/auth/registerController.test.js
PASS  tests/controllers/auth/loginController.test.js
PASS  tests/controllers/listingsController.test.js

Test Suites: 3 passed, 3 total
Tests:       15 passed, 15 total
Snapshots:   0 total
Time:        1.273 s
Ran all test suites.
```

slika 6.1.: Jest test

Testiranje filtera Difficulty i Number Of Players

Mock-ani podatci		
Opis	Očekivani podatci	Dobiveni rezultat
Postavljanje filtera - Difficulty: easy	{ name: "Easy Game", difficulty: 1, maxPlayers: 2}	{ name: "Easy Game", difficulty: 1, maxPlayers: 2}
Postavljanje filtera - Number Of Players: 2-4 players	{ name: "Medium Party Game", difficulty: 2, maxPlayers: 4}	{ name: "Medium Party Game", difficulty: 2, maxPlayers: 4}
Postavljanje filtera - Difficulty: hard && Number Of Players: 4+ palyers	{ name: "Hard Strategy Game", difficulty: 4, maxPlayers: 6}	{ name: "Hard Strategy Game", difficulty: 4, maxPlayers: 6}

link: [Filter-Search Unit Test](#)

Testiranje search funkcionalnosti

Opis	Ulagani podatak	Očekivani podatci	Dobiveni podatci
Search funkcionalnosti	"Party"	{ name: "Medium Party Game", difficulty: 2, maxPlayers: 4}	{ name: "Medium Party Game", difficulty: 2, maxPlayers: 4}
Osjetljivosti na velika i mala slova	"PARTY"	{ name: "Medium Party Game", difficulty: 2, maxPlayers: 4}	{ name: "Medium Party Game", difficulty: 2, maxPlayers: 4}
Pretraživanja nepostojeće igre	"Nonexistent"	[]	[]
Bez pretraživanja (prazan search)	""	{ name: "Easy Game", difficulty: 1, maxPlayers: 2}, { name: "Medium Party Game", difficulty: 2, maxPlayers: 4}, { name: "Hard Strategy Game", difficulty: 4, maxPlayers: 6}	{ name: "Easy Game", difficulty: 1, maxPlayers: 2}, { name: "Medium Party Game", difficulty: 2, maxPlayers: 4}, { name: "Hard Strategy Game", difficulty: 4, maxPlayers: 6}

link: [Filter-Search Unit Test](#)

Testiranje sign up funkcionalnosti

Opis	Očekivani podatci	Dobiveni podatci

Opis	Očekivani podaci	Dobiveni podaci
Korisnik već postoji	409, "Username or email already in use"	409, "Username or email already in use"
Uspješna registracija	201, "Registration successful!"	201, "Registration successful!"

link: [Sgin up Unit Test](#)

Testiranje log in funkcionalnosti

Opis	Očekivani podaci	Dobiveni podaci
Nedostaje identifikator ili lozinka	400, "All fields are required"	400, "All fields are required"
Nije pronađen korisnik	401, "User not Found!"	401, "User not Found!"
Lozinka se ne podudara	401, "Wrong Password!"	401, "Wrong Password!"
Uspješna prijava	200, "Login successful!"	200, "Login successful!"

link: [Log in Unit Test](#)

Ispitivanje sustava

Cilj ispitivanja sustava je testiranje ponašanja cijelog sustava u uvjetima stvarnog korištenja, uz posebnu pažnju na međusobnu povezanost svih komponenti. Ispitivanje treba obuhvatiti sve aspekte sustava i njegovu interakciju s korisnicima.

Sljedeći testovi su napravljeni pomoću alata Selenium

Sign up test

Koraci:

1. Otvoriti stranicu
2. Pritisnuti na gumb Log In -> korisnik je preusmjeren na stranicu za log in
3. Pritisnuti na "New to here? Sign Up"
4. Unijeti Email: "selenium.testing @ testing.test"
5. Unijeti Username: "Selenium Testing"
6. Unijeti Password: "test123"
7. Pritisnuti gumb Sign Up -> korisnik je registriran i preusmjeren na početni zaslon

Očekivani rezultat: Korisnik je uspješno registriran i preusmjeren na početnu stranicu.

Dobiveni rezultat: Test je uspješno izvršen, korisnik je registriran i preusmjeren na početnu stranicu.

https://frontend.err808.xyz

	Command	Target	Value
1	✓ open	/	
2	✓ set window size	1536x824	
3	✓ click	linkText=Log In	
4	✓ click	css=.flip-text:nth-child(5)	
5	✓ click	id=signup_email	
6	✓ type	id=signup_email	selenium.testing@testing.test
7	✓ click	id=signup_username	
8	✓ type	id=signup_username	Selenium Testing
9	✓ click	id=signup_password	
10	✓ type	id=signup_password	test123
11	✓ click	id=signup_repeat_password	
12	✓ type	id=signup_repeat_password	test123
13	✓ click	css=.login_buttons:nth-child(10) > .form__submit-button	
14	✓ click	css=.public-browse-link	

slika 6.2.: SginUp_test

Rubni slučaj: Korisnik unese Email ili Username koji već postoji -> iskače prozor koji obavješta korisnika o tome da je Email ili Username već u uporabi, korisnik nije preusmjeren na početnu stranicu.

https://frontend.err808.xyz

	Command	Target	Value
1	✓ open	/	
2	✓ set window size	1536x824	
3	✓ click	linkText=Log In	
4	✓ click	css=.flip-text:nth-child(5)	
5	✓ click	id=signup_email	
6	✓ type	id=signup_email	selenium.testing@testing.test
7	✓ click	id=signup_username	
8	✓ type	id=signup_username	Test
9	✓ click	id=signup_password	
10	✓ type	id=signup_password	test123
11	✓ click	id=signup_repeat_password	
12	✓ type	id=signup_repeat_password	test123
13	✓ click	css=.login_buttons:nth-child(10) > .form__submit-button	
14	X click	css=.public-browse-link	

slika 6.3.: SginUp_test_fail

Log in test

Koraci:

1. Otvoriti stranicu
2. Pritisnuti na gumb Log In -> korisnik je preusmjeren na stranicu za log in
3. Unijeti Username: "Selenium Testing"
4. Unijeti Password: "test123"
5. Pritisnuti gumb Log In -> korisniku iskače prozor s porukom "Login successful!" i gumbom za preusmjeravanje na početni zaslon
6. Pritisnuti gumb "Go To Front Page" -> korisnik je preusmjeren na početni zaslon

Očekivani rezultat: Korisnik je uspješno prijavljen i preusmjeren na početnu stranicu.

Dobiveni rezultat: Test je uspješno izvršen, korisnik je prijavljen i preusmjeren na početnu stranicu.

https://frontend.err808.xyz			
	Command	Target	
		Value	
1	✓ open	/	
2	✓ set window size	1263x778	
3	✓ click	css=.public	
4	✓ click	linkText=Log In	
5	✓ click	id=login_email	
6	✓ type	id=login_email	Selenium Testing
7	✓ click	id=login_password	
8	✓ type	id=login_password	test123
9	✓ click	css=.login_buttons:nth-child(6) > .form__submit-button	
10	✓ click	css=.auth-done-btn	
11	✓ click	css=.public-browse-link	

slika 6.4: Login_test

Rubni slučaj: Korisnik unese pogrešan Email/Username ili Password -> iskače prozor koji obavješta korisnika o tome da je unio pogrešan Email/Username ili Password, korisnik nije preusmjeren na početnu stranicu.

https://frontend.err808.xyz			
	Command	Target	
		Value	
1	✓ open	/	
2	✓ set window size	1263x778	
3	✓ click	css=.public	
4	✓ click	linkText=Log In	
5	✓ click	id=login_email	
6	✓ type	id=login_email	Selenium Testing
7	✓ click	id=login_password	
8	✓ type	id=login_password	wrongPass
9	✓ click	css=.login_buttons:nth-child(6) > .form__submit-button	
10	✓ click	css=.auth-done-btn	
11	X click	css=.public-browse-link	

slika 6.5: Login_test_fail_wrongPass

<https://frontend.err808.xyz>

	Command	Target	Value
1	✓ <code>open</code>	/	
2	✓ <code>set window size</code>	1263x778	
3	✓ <code>click</code>	css=.public	
4	✓ <code>click</code>	linkText=Log In	
5	✓ <code>click</code>	id=login_email	
6	✓ <code>type</code>	id=login_email	wrongName
7	✓ <code>click</code>	id=login_password	
8	✓ <code>type</code>	id=login_password	test123
9	✓ <code>click</code>	css=.login_buttons:nth-child(6) > .form__submit-button	
10	✓ <code>click</code>	css=.auth-done-btn	
11	X <code>click</code>	css=.public-browse-link	

slika 6.6: Login_test_fail_wrongName

NewListing_test

Koraci:

1. Otvoriti stranicu
2. Pritisnuti na gumb Make New Listing -> korisnik je preusmjeren na stranicu za stvaranje novog Listing-a
3. Unos podataka o igri
4. Pritisnuti gumb Submit -> iskače prozor s porukom "Listing added successfully" i gumbom "OK"
5. Pritisnuti gumb OK -> korisnik je preusmjeren na stranicu My Games

Očekivani rezultat: Korisnik je uspješno objavio oglas za igru te je preusmjeren na stranicu My Games.

Dobiveni rezultat: Test je uspješno izvršen, korisnik je objavio oglas za igru te je preusmjeren na stranicu My Games.

<https://frontend.err808.xyz>

	Command	Target	Value
1	✓ open	/	
2	✓ set window size		1263x778
3	✓ click	linkText=Make New Listing	
4	✓ click	name=name	
5	✓ type	name=name	Selenium Test Listing
6	✓ click	name=publisher	
7	✓ type	name=publisher	Selenium
8	✓ click	name=genre	
9	✓ type	name=genre	Test
10	✓ click	name=releaseYear	
11	✓ type	name=releaseYear	2000
12	✓ click	name=condition	
13	✓ select	name=condition	label>New
14	✓ click	css=option:nth-child(2)	
15	✓ click	name=minPlayers	
16	✓ type	name=minPlayers	2
17	✓ click	name=maxPlayers	
18	✓ type	name=maxPlayers	4
19	✓ click	name=playTime	
20	✓ type	name=playTime	5
21	✓ click	name=difficulty	
22	✓ type	name=difficulty	1
23	✓ click	css=.form-group:nth-child(9) > label	
24	✓ click	css=textarea	
25	✓ click	css=.full-width > input	
26	✓ type	css=.full-width > input	C:\Users\Toni\Pictures\test.jpg
27	✓ click	css=textarea	
28	✓ type	css=textarea	...
29	✓ click	css=.primary-button	
30	✓ click	css=.auth-done-btn	

slika 6.7.: NewListing_test

Rubni slučaj: Korisnik pokuša napraviti oglas bez da ima postavljenu lokaciju na profilu, korisnik nije ispunio sve obavezne podatke o igri.

<https://frontend.err808.xyz>

	Command	Target	Value
1	✓ <code>open</code>	/	
2	✓ <code>set window size</code>	1263x778	
3	✓ <code>click</code>	linkText=Make New Listing	
4	X <code>click</code>	name=name	
5	<code>type</code>	name=name	Selenium Test Listing
6	<code>click</code>	name=publisher	
7	<code>type</code>	name=publisher	Selenium
8	<code>click</code>	name=genre	
9	<code>type</code>	name=genre	Test
10	<code>click</code>	name=releaseYear	
11	<code>type</code>	name=releaseYear	2000
12	<code>click</code>	name=condition	
13	<code>select</code>	name=condition	label>New
14	<code>click</code>	css=option:nth-child(2)	
15	<code>click</code>	name=minPlayers	
16	<code>type</code>	name=minPlayers	2
17	<code>click</code>	name=maxPlayers	
18	<code>type</code>	name=maxPlayers	4
19	<code>click</code>	name=playTime	
20	<code>type</code>	name=playTime	5
21	<code>click</code>	name=difficulty	
22	<code>type</code>	name=difficulty	1
23	<code>click</code>	css=.form-group:nth-child(9) > label	
24	<code>click</code>	css=textarea	
25	<code>click</code>	css=.full-width > input	
26	<code>type</code>	css=.full-width > input	C:\Users\Toni\Pictures\test.jpg
27	<code>click</code>	css=textarea	
28	<code>type</code>	css=textarea	...
29	<code>click</code>	css=.primary-button	
30	<code>click</code>	css=.auth-done-btn	

slika 6.8.: `NewListing_test_fail_noProfileLocation`

	Command	Target	Value
1	✓ open	/	
2	✓ set window size	1263x778	
3	✓ click	linkText=Make New Listing	
4	✓ click	name=name	
5	✓ type	name=name	Selenium Test Listing
6	✓ click	name=publisher	
7	✓ click	name=genre	
8	✓ type	name=genre	Test
9	✓ click	name=releaseYear	
10	✓ type	name=releaseYear	2000
11	✓ click	name=condition	
12	✓ click	css=option:nth-child(2)	
13	✓ click	name=minPlayers	
14	✓ type	name=minPlayers	2
15	✓ click	name=maxPlayers	
16	✓ type	name=maxPlayers	4
17	✓ click	name=playTime	
18	✓ type	name=playTime	5
19	✓ click	name=difficulty	
20	✓ type	name=difficulty	1
21	✓ click	css=.form-group:nth-child(9) > label	
22	✓ click	css=textarea	
23	✓ click	css=.full-width > input	
24	✓ type	css=.full-width > input	C:\Users\Toni\Pictures\test.jpg
25	✓ click	css=textarea	
26	✓ type	css=textarea	...
27	✓ click	css=.primary-button	
28	X click	css=.auth-done-btn	

slika 6.9.: NewListing_test_fail_missingDetail

OfferExchange_test

Koraci:

1. Otvoriti stranicu
2. Pritisnuti na gumb Browse All Games -> korisnik je preusmjeren na stranicu sa svim oglasima
3. Pritisnuti na oglas -> korisnik je preusmjeren na stranicu oglasa
4. Pritisnuti na gumb Offer Exchange -> iskače prozor sa listom igara koje korisnik može ponuditi za zamijenu
5. Označi igru kao ponudu za zamijenu
6. Pritisnuti gumb Offer
7. Pritisnuti na gumb My Offers -> korisnik je preusmijeren na stranicu My Offers gdje je vidljiva ponuda za zamijenu

Očekivani rezultat: Korisnik je uspješno napravio trade offer te je preusmjeren na stranicu My Offers**Dobiveni rezultat:** Test je uspješno izvršen, korisnik je napravio trade offer te je preusmjeren na stranicu My Offers.

Command		Target	Value
1	✓ open	/	
2	✓ set window size		1263x778
3	✓ click		linkText=Browse All Games
4	✓ click		css=.game-card:nth-child(4) .game-card-details
5	✓ run script		window.scrollTo(0,0)
6	✓ click		css=.primary-button
7	✓ click		css=input
8	✓ click		css=.trade-actions > .primary-button
9	✓ click		linkText=My Offers
10	✓ run script		window.scrollTo(0,0)

slika 6.10: OfferExchange_test

Rubni slučaj: Korisnik koji nema nijedan oglas za igru pokuša napraviti trade offer.

Command		Target	Value
1	✓ open	/	
2	✓ set window size		1263x778
3	✓ click		linkText=Browse All Games
4	✓ click		css=.game-card:nth-child(4) .game-card-details
5	✓ run script		window.scrollTo(0,0)
6	✓ click		css=.primary-button
7	X click		css=input
8	click		css=.trade-actions > .primary-button
9	click		linkText=My Offers
10	run script		window.scrollTo(0,0)

slika 6.11: OfferExchange_test_fail_noListings# Korištene tehnologije i alati

Redni broj	Tehnologija / Alat	Verzija	Kategorija
1	JavaScript	ES6+	programski jezik
2	HTML	HTML5	programski jezik
3	CSS	CSS3	programski jezik
4	Node.js	20.19.2	radni okvir
5	Express.js	5.1.0	radni okvir
6	React	19.2.0	radni okvir
7	Passport.js	0.7.0	radni okvir
8	Mongoose	8.19.3	radni okvir
9	Nodemailer	7.0.12	radni okvir
10	MongoDB	8.2	baza podataka
11	WebStorm	2025.3	razvojni alat

Rедни број	Технологија / Алат	Верзија	Категорија
13	Neovim	-	развојни алат
14	Git	2.52.0	развојни алат
15	GitHub Actions	-	алат за размјештај
16	Azure	-	развојни алат
17	Selenium	-	алат за испитивање
18	Jest	-	алат за испитивање
19	OpenStreetMap API	0.6	развојни алат

Programski jezici

JavaScript

JavaScript је програмски језик који се користи за развој веб апликација на клијентској и послуžитељској страни. У нашем пројекту JavaScript је основни језик за развој фронтенда и бекенда апликације. Кorištenje ES6+ стандарда омогућује модерну синтаксу, бољу читљивост кода и компатibilnost s библиотекама i алатима.

HTML

HTML је стандардни језик који се користи за дефинирање структуре веб страница. У пројекту га користимо за израду основне структуре корисничког сеџела. Верзија HTML5 доноси семантичке елементе и бољу подршку за мултимедијалне садржаве.

CSS

CSS је језик намјенjen описивању изгледа и стилизирању веб страница. У пројекту се користи за дефинирање визуалног идентитета корисничког сеџела te за осигуравање респонзивног приказа на разлиčitim uređajima.

Radni okviri i biblioteke

Node.js

Node.js је JavaScript runtime који омогућује извођење кода изван веб preglednika i често се користи за развој послуžитељског dijela апликација. У пројекту се користи верзија 20.19.2, која припада LTS grani i доноси стабилност, sigurnosne nadogradnje te kompatibilnost s modernim backend библиотекама.

Express.js

Express.js је радни оквир за израду веб апликација у Node.js окружењу. У пројекту се користи за implementацију REST API-ja i obradu HTTP zahtjeva. Верзија 5.1.0 pruža poboljšanu sigurnost i stabilnost.

React

React је радни оквир за израду корисничких сеџела темељен на компонентном приступу. У пројекту се користи за развој фронтенда апликације, омогућујући динамиично i респонзивно корисничко сеџеље. Верзија 19.2.0 осигурује подршку за најновије значајке i оптимизације.

Passport.js

Passport.js је радни оквир за аутентификацију корисника. У пројекту се користи за implementацију sustava prijave i autentikacije korisnika. Njegova modularnost омогућује jednostavnu integraciju različitih autentikacijskih strategija.

Mongoose

Mongoose је радни оквир (ODM) за rad s MongoDB bazom podataka. Omogućuje definiranje shema i modela te структурирани приступ подацима. У пројекту га користимо за управљање подацима i validaciju unosa korisnika.

Nodemailer

Nodemailer је радни оквир за slanje elektroničke pošte iz Node.js апликација. У пројекту се користи за slanje automatiziranih e-mail poruka, primjerice за потврду registracije ili obavijesti korisnicima.

Baza podataka

MongoDB

MongoDB је NoSQL база података која пohranjuje податке u облику документа. У пројекту се користи за pohranu корисничких података i осталих информација апликације. Верзија 8.2 омогућује visoku skalabilnost i fleksibilnu strukturu podataka.

Razvojni alati

WebStorm

WebStorm је integrirano razvojno okruženje (IDE) namijenjeno razvoju JavaScript апликација. У пројекту се користи за развој backenda i frontenda zbog naprednih alata za analizu i pisanje koda.

Visual Studio Code

Visual Studio Code je jednostavan i fleksibilan editor koda koji podržava velik broj programskih jezika i proširenja. U projektu se koristi za razvoj i uređivanje izvornog koda.

Neovim

Neovim je editor koda, dizajniran za brzinu, proširivost i učinkovit rad bez grafičkog sučelja. Omogućuje uređivanje datoteka kroz prečace, skriptiranje i širok ekosustav dodataka. U projektu se koristi kao alternativni razvojni alat za rad u terminalskom okruženju.

Git

Git je distribuirani sustav za kontrolu verzija koji omogućuje praćenje promjena u kodu i timsku suradnju. U projektu se koristi za upravljanje izvornim kodom i verzijama aplikacije.

Azure

Azure je cloud platforma koja se u projektu koristi za razmještaj i hosting aplikacije. Omogućuje pokretanje aplikacije u oblaku bez potrebe za vlastitom infrastrukturom, čime se pojednostavljuje održavanje sustava. Azure pruža visoku dostupnost, mogućnost skaliranja resursa prema opterećenju te centralizirano upravljanje aplikacijskim komponentama.

OpenStreetMap API

OpenStreetMap API omogućuje pristup geolokacijskim podacima. U projektu se koristi za postavljanje i odabir lokacije, čime se omogućuje precizno definiranje lokacije oglasa unutar aplikacije.

Alati za ispitivanje

Selenium

Selenium je alat za automatizirano testiranje web aplikacija. U našem projektu se koristi za testiranje korisničkog sučelja simulacijom interakcije korisnika s aplikacijom.

Jest

Jest je alat za jedinično testiranje JavaScript aplikacija. U projektu se koristi za provjeru ispravnosti funkcionalnosti pojedinih dijelova koda, čime se osigurava stabilnost i pouzdanost aplikacije.

Alati za razmještaj

GitHub Actions

GitHub Actions je alat za automatizaciju procesa razvoja softvera. U ovom projektu koristi se za razmještaj aplikacije, odnosno za automatizirano postavljanje aplikacije na proizvodjsko okruženje nakon promjena u repozitoriju. Time se osigurava dosljedan i ponovljiv proces razmještaja.

Dodatni alati

Za komunikaciju i sastanke smo koristili:

1. WhatsApp
2. Discord

- **Preduvjeti:**

1. Node.js 20
2. npm [LATEST]

- **Preuzimanje:**

```
$ git clone https://github.com/tim-error808/error808.git
```

Instalacija ovisnosti:

```
$ cd error808
$ cd backend
$ npm install
$ cd ../frontend
$ npm install
```

2. Postavke

Detaljne upute za konfiguraciju aplikacije:

- **Konfiguracijske datoteke:**

1. frontend

Potrebno je postaviti adresu backend API. Konfiguracijska datoteka se nalazi u frontend/src/config/ModeConfig.js

2. backend

Potrebno je postaviti environment varijable:

```
FRONTEND_URL
GOOGLE_CLIENT_ID - oauth
GOOGLE_CLIENT_SECRET - oauth
JWT_SECRET
LOCAL_TEST
MONGODB_URI
REFRESH_SECRET
REST_API_PORT
EMAIL_SENDER - "from@someone.domain"
EMAIL_SMTP_HOST
EMAIL_PASSWORD
```

Kontekst pojedine varijable je jasan iz naziva varijabli. Ovisno o okruženju postavljanje varijabli se razlikuje, stoga je na korisniku da poznaje alate svoga okruženja.

- **Postavke baze podataka:**

Baza podataka se postavlja na temelju modela koji se nalaze u backend/models direktoriju i na wiki stranicama.

3. Pokretanje aplikacije

```
$ cd backend
$ npm run dev-start
$ cd ../frontend
$ npm start
```

4. Upute za administratore

Smjernice za administratore aplikacije nakon puštanja u pogon:

- **Pristup administratorskom sučelju:**

- URL/admin
- Administrator treba biti postavljen ili dodijeljen od strane drugog administratora

- **Redovito održavanje:**

- Redovito pregledati stanje web aplikacije na servisu koji se koristi za hosting.

5. Puštanje u pogon na Azure platformi

Azure je platforma koja je korištena za ovaj projekt kako je dostupna studentima za razvoj projekata bez dodatnih troškova.

- Za frontend koristi se Static Web App

1. Na portalu Azure usluge odabrati izradu Static Web App
2. Povezati s github repozitorijem projekta, tj folderom frontend
3. Azure automatski postavlja skriptu za deploy frontend projekta
4. Postaviti simboličku adresu po uputama na platformi
5. Postaviti potrebne environment varijable na sučelju kreirane Static Web App usluge

- Za backend koristi se App Service

1. Na portalu Azure usluge odabrati izradu App Service
2. Povezati s github repozitorijem projekta, tj folderom backend
3. Azure automatski postavlja skriptu za deploy backend projekta
4. Postaviti simboličku adresu po uputama na platformi
5. Postaviti potrebne environment varijable na sučelju kreirane App Service usluge
6. Postaviti CORS postavke na sučelju kreirane App Service usluge

Aplikaciji se pristupa pomoću postavljene simboličke adrese.

Osvrt na vrijeme izrade projektnog zadatka

Izrada projektnog zadatka trajala je oko 12 tjdana (nakon što se oduzme vrijeme u kojem se svaki član tima posvetio drugim obavezama).

Izrada je počela u prvoj polovici listopada 2025., a završila je krajem siječnja 2026. godine.

Tijekom tog vremena pokušalo se što bolje koristiti uputama za izvođenje projekta na koje nas je usmjeravao asistent, kako bi rad na projektu prošao bez velikih prepreka, odgađanja i nagomilavanja zadatka prije važnih rokova.

Najvažniji rokovi s kojima smo se susretali na projektu bili su:

- prezentacija osnovne organizacije i plana projekta (21.10.2025.)
- predaja projekta na GitHub za prvi ciklus (14.11.2025.)
- demonstracija alfa verzije aplikacije (13.01.2026.)
- predaja projekta na GitHub za drugi ciklus (23.01.2026.)

Najveći dio vremena rada na projektu bio je posvećen uskladišvanju članova tima, razradi funkcijskih i nefunkcijskih zahtjeva, ali i samoj implementaciji.

Prepoznati tehnički izazovi i njihova rješenja

Primjeri izazova:

- arhitektura sustava - kako odrediti optimalnu arhitekturu na kojoj će se temeljiti aplikacija koja ispunjava sve zahtjeve?
- rad s bazom podataka - kako izraditi bazu podataka u novom sustavu s kojim članovi tima još nemaju iskustva i kako tu bazu povezati s ostalim komponentama?
- autentifikacija korisnika - kako svelatiti i implementirati OAuth 2.0 sustav s kojim članovi tima nemaju iskustva?
- integracija vanjskih sustava - na koji način iskoristiti vanjski sustav za optimalnu integraciju s aplikacijom?
- sinkronizacija rada u timu - na koji način uskladiti rad svih članova tima tako da svatko dobije zadatku s kojim se može nositi, a da pritom doprinese maksimalno radu tima?

Rješenja izazova:

- problem arhitekture je riješen na način da je tim organizirao sastanak na kojem se prošlo kroz različite organizacije sustava kako bi se utvrdilo koji od njih najbolje odgovara zahtjevima projekta
- izazov stvaranja i integracije baze podataka je riješen tako da su članovi tima koji su bili zaduženi za njeno funkcioniranje prošli kroz dokumentaciju kako bi shvatili na koji način sustav radi i iskoristili ga za konstruiranje baze koja zadovoljava zahtjeve projekta
- autentifikaciju korisnika preuzeli su članovi tima koji imaju najveće iskustvo potrebitno za njeno funkcioniranje, ali i oni su trebali proći kroz dokumentaciju kako bi se upoznali s novim sustavom
- integracija vanjskih sustava je riješena tako da su članovi tima pročitali službene upute za njenu integraciju i primjenili znanje na njeno implementiranje u sustav
- sinkronizacija rada članova tima je riješena kontinuiranom komunikacijom, sastancima i kvalitetnom podjelom zadatka

Stečena znanja

Kratak opis znanja i vještina koje smo stekli radom na projektu:

Timski rad

Tijekom razvoja projekta stekli smo iskustvo rada u timu i vidjeli kako otrilike izgleda programsko okruženje koje zajedničkim snagama razvija neki sustav. Stekli smo iskustvo kako je to kombinirati različita znanja s kojima dolaze različiti članovi tima i uvidjeli da programiranje nije samo pisanje programskog koda nego sveobuhvatan proces razvoja programske potpore koji uključuje stalnu komunikaciju, revizije, dokumentaciju, ispravljanje grešaka...

Planiranje

Stekli smo uvid u to o kojim stvarima se treba pobrinuti tijekom razvoja programske potpore i riješiti ih na vrijeme. Primjerice, naučili smo kako stvari ne treba rješavati u zadnji tren jer neочекivani problemi znaju iskrsnuti. Isto tako naučili smo kako treba pomno pratiti sve rokove i kod složenijih problema voditi uredne bilješke o rokovima i radu. Iskusili smo kako je to prilagođavati se radu ostalih članova tima imajući na umu da je ključna stvar razviti kvalitetan projekt.

Debugiranje

Naučili smo kako će kod razvoja programske potpore često trebati zastati, vratiti se unatrag i ispraviti pogrešku u kodu za koji smo vjerovali da je u redu. Na prvu se može činiti kao da sporo čitanje i testiranje dijela sustava koji ne radi ispravno usporava rad na projektu, ali zapravo treba razumjeti da se takve stvari treba očekivati i izdvojiti unaprijed vremena za nošenje s njima.

Dokumentacija

Shvatili smo kako dokumentacija nije samo onaj dio projekta koji se treba napraviti da se zadovolji forma, nego vrlo često služi za osvrt napravljenog članovima tima, ali i svima onima koji bi mogli biti zainteresirani za detalje implementacije i razvoja sustava.

Nove tehnologije

Radom na projektu morali smo se prilagoditi novim sustavima i tehnologijama, kao što su MS Azure, MongoDB, OAuth 2.0, Postman... i time ne samo da smo omogućili potrebnu funkcionalnost aplikacije, već i stekli znanje koje ostaje i za vrijeme nakon projekta.

Znanja potrebna za bržu i kvalitetniju izradu projekta

Za bržu i kvalitetniju izradu projekta trebala bi nam šira znanja i unaprijed poznavanje određenih sustava jer tada ne bismo trebali čitati opsežnu dokumentaciju kako bismo razumijeli kako sustav radi. Osim toga bila bi nam potrebna šira znanja o lakšoj i bržoj komunikaciji i integraciji s drugim članovima tima kako bi zajednički rad tekao što kvalitetnije.

Perspektive za nastavak rada u projektnoj grupi

Projekt ima potencijala za daljnji razvoj. Članovi tima ostvarili su dobru suradnju i s lakoćom bi mogli nastaviti surađivati na projektu i nadograđivati ga različitim funkcionalnostima te ga usavršiti kako bi bio potpuno prilagođen za javnu uporabu. Primjeri funkcionalnosti koji bi se mogli dodati u aplikaciju: omogućavanje komunikacije korisnika izravno unutar aplikacije, zamjena ne samo društvenih igara, već i ostalih povezanih komponenti, kao što su figurice, kockice i ostalo, povezivanje korisnika u grupe sličnih interesa i/ili sličnih lokacija kako bi mogli nastaviti suradnju i komunikaciju uživo...

Ograničenja

Ograničenja s kojima se susretalo tijekom razvoja su finansijske i vremenske prirode. Mnoge već napravljene stvari smo mogli poboljšati da smo imali više vremena za razvoj, ali osim toga, mnogli smo dodati i neke dodatne, napredne funkcionalnosti. Što se financija tiče, zbog ograničenog budžeta koristili smo besplatne ili jeftine verzije nekih sustava i time ograničili funkcionalnosti našeg sustava (primjerice, prostor u bazi podataka).

Funkcionalnosti koje nisu implementirane

Naš sustav pokrio je sve funkcionalnosti opisane zahtjevima projekta, međutim postoje područja za daljnji razvoj, kao što je poboljšanje performansa, dodavanje novih funkcionalnosti za korisnika i slično.1. Programsko inženjerstvo, FER ZEMRIS, <http://www.fer.hr/predmet/proinz>

2. The Original "Buy, Sell & Trade" Boardgame Group – Facebook. Dostupno: <https://www.facebook.com/groups/boardgameexchange/> (20. listopad 2025.).
3. BoardGameGeek | Gaming Unplugged Since 2000. Dostupno: <https://boardgamegeek.com/> (20. listopad 2025.).
4. BoardGamesSwap – Buy, Sell and Trade Board Games. Dostupno: <https://www.boardgamesswap.com/> 20. listopad 2025.).
5. MongoDB Documentation. Dostupno: <https://www.mongodb.com/docs/> (7. studeni 2025.).
6. React – Learn React. Dostupno: <https://react.dev/learn> (7. studeni 2025.).
7. Node.js – Introduction to Node.js. Dostupno: <https://nodejs.org/en/learn/getting-started/introduction-to-nodejs> (7. studeni 2025.).
8. Client-Server Architecture – System Design | GeeksforGeeks. Dostupno: <https://www.geeksforgeeks.org/system-design/client-server-architecture-system-design/> (10. studeni 2025.).
9. MVC Architecture – System Design | GeeksforGeeks. Dostupno: <https://www.geeksforgeeks.org/system-design/mvc-architecture-system-design/> (10. studeni 2025.).
10. Frontend vs Back-end Development | GeeksforGeeks. Dostupno: <https://www.geeksforgeeks.org/blogs/frontend-vs-backend/> (10. studeni 2025.).

Rev.	Opis promjene/dodataka	Autori	Datum
0.1	Napravljen home page	Niko Knežević	20.10.2025.
0.2	Uređen home page	Niko Knežević	20.10.2025.
0.3	Kloniran predložak dokumentacije	Niko Knežević	20.10.2025.
1.1	Popunjena stranica dokumentacije	Niko Knežević	20.10.2025.
2.1	Popunjen dio funkcijskih i nefunkcijskih zahtjeva na stranici	Niko Knežević	20.10.2025.
B.1	Dodane informacije o sastancima	Niko Knežević	20.10.2025.
2.2	Dovršena stranica dokumentacije	Niko Knežević	2.11.2025.
B.2	Dodane informacije o sastancima	Niko Knežević	2.11.2025.
B.3	Dodane informacije o sastancima	Niko Knežević	6.11.2025.
B.4	Napravljena i ažurirana tablica aktivnosti	Niko Knežević	6.11.2025.
B.5	Napravljena i ažurirana tablica plana rada	Niko Knežević	6.11.2025.
2.3	Ažurirani podatci o dionicima	Niko Knežević	6.11.2025.
3.1	Izrađena tablica obrazaca uporabe	Niko Knežević	6.11.2025.
3.2	Ispisani detaljni podatci o svakom obrascu uporabe	Niko Knežević	6.11.2025.
4.1	Ispisana većina stvari o arhitekturi sustava i korištenim tehnologijama	Niko Knežević	7.11.2025.
3.3	Napravljeni dijagrami obrazaca uporabe	Frane Ćovid, Niko Knežević	8.11.2025.
3.4	Napravljen prvi sekvencijski dijagram	Frane Ćovid, Niko Knežević	8.11.2025.
3.5	Uređeni i popravljeni dijagrami obrazaca uporabe	Niko Knežević	9.11.2025.
3.6	Uređeni i popravljeni opisi obrazaca uporabe i njihove veze s funkcijskim zahtjevima	Niko Knežević	9.11.2025.
2.4	Nadodani i ispravljeni funkcijski zahtjevi	Niko Knežević	9.11.2025.
4.2	Opisana arhitektura baze podataka	Frane Ćovid	9.11.2025.
4.3	Opisan ostatak arhitekture	Niko Knežević	9.11.2025.
README.1	Napisan je predložak za README	Niko Knežević	9.11.2025.
A.1	Postavljena literatura korištена u izradi dokumentacije	Niko Knežević	10.11.2025.
3.7	Napravljen ostatak sekvencijskih dijagrama	Niko Knežević, Ivan Žalac	10.11.2025.

B.6 Rev.	Opis problemi rada na projektu Opis promjene/dodataka	Niko Knežević Autori	Datum
4.4	Izrađen dijagram stanja	Niko Knežević	09.01.2026.
4.5	Izrađen dijagram aktivnosti	Niko Knežević	10.01.2026.
5.1	Izrađen dijagram komponenata i dijagram razmještaja	Frane Ćevid	11.01.2026.
8.1	Izrađen glavni dio uputa za puštanje u pogon	Ivan Žalac	12.01.2026.
6.1	Izrađen dio wiki dokumentacije o testovima	Toni Kapučija	12.01.2026
B.6	Ažuriranje aktivnosti i postavljanje grafova	Niko Knežević	23.01.2026.
6.2	Ažuriranje wiki dokumentacije o testovima	Toni Kapučija	23.01.2026.
4.6	Dodan dijagram razreda	Marko Bošnjak	23.01.2026.

Napomena : u tablici nisu navedene promjene dokumentacije nasatale u ovom poglavju # Dnevnik sastajanja

1. sastanak

- Datum: 9. listopada 2025.
- Prisustvovali: Marko Bošnjak, Marin Čikotić, Frane Ćevid, Toni Kapučija, Niko Knežević, Mihael Rošić
- Teme sastanka:
 - konačna odluka o temi projekta
 - dodjela uloge svakom članu tima
 - stvaranje GitHub repozitorija

2. sastanak

- Datum: 18. listopada 2025.
- Prisustvovali: Marko Bošnjak, Marin Čikotić, Frane Ćevid, Toni Kapučija, Niko Knežević, Mihael Rošić, Ivan Žalac
- Teme sastanka:
 - odluka o korištenim tehnologijama
 - detaljniji opis uloga i prvih zadatka
 - stvaranje PowerPoint prezentacije za predstavljanje projekta

3. sastanak

- Datum: 26. listopada 2025.
- Prisustvovali: Niko Knežević, Ivan Žalac
- Teme sastanka:
 - detaljno raspisivanje svih zahtjeva i funkcionalnosti koje treba pokriti projekt
 - dogovor oko daljnjih aktivnosti s ostatkom tima
 - dogovor oko načina daljnjih doprinosa na projektu

4. sastanak

- Datum: 1. studenog 2025.
- Prisustvovali: Marko Bošnjak, Marin Čikotić, Frane Ćevid, Toni Kapučija, Niko Knežević, Mihael Rošić, Ivan Žalac
- Teme sastanka:
 - osvrt na do sad napravljene stvari
 - usporedba napravljenog s očekivanim ciljevima
 - razgovor o detaljima implementacije
 - podjela poslova među članovima grupe

5. sastanak

- Datum: 2. studenog 2025.
- Prisustvovali: Mihael Rošić, Ivan Žalac
- Teme sastanka:
 - dogovor oko izgleda korisničkog sučelja
 - raspisivanje prvih zahtjeva

6. sastanak

- Datum: 6. studenog 2025.
- Prisustvovali: Marko Bošnjak, Marin Čikotić, Mihael Rošić
- Teme sastanka:
 - detaljniji dogovor oko dodijeljenih zadataka i njihova podjela
 - pisanje dijela dokumentacije o arhitekturi
 - napisani issues potrebeni za prvu predaju projekta
 - rad na backendu

7. sastanak

- Datum: 8. studenog 2025.
- Prisustvovali: Frane Ćevid, Niko Knežević
- Teme sastanka:
 - izrada dijagrama obrazaca uporabe
 - izrada jednog sekvencijskog dijagrama
 - dogovor oko daljnjih aktivnosti

8. sastanak

- Datum: 10. studenog 2025.
- Prisustvovali: Marko Bošnjak, Toni Kapučija, Mihael Rošić
- Teme sastanka:
 - povezivanje dijelova sustava
 - testiranje funkcionalnosti
 - priprema pitanja za termin laboratorijske vježbe

9. sastanak

- Datum: 14. studenog 2025.
- Prisustvovali: Marko Bošnjak, Toni Kapučija
- Teme sastanka:
 - testiranje implementacije
 - popravljanje korištenja OAtuh 2.0-a

10. sastanak

- Datum: 10. prosinca 2025.
- Prisustvovali: Marko Bošnjak, Marin Čikotić, Toni Kapučija, Niko Knežević, Mihael Rošić
- Teme sastanka:
 - osvrt na odradene segmente
 - planiranje daljnog rada
 - podjela zadataka drugog ciklusa članovima tima

Plan rada

Zadatak	Rok	Zaduženi	Status
Tema projekta	10.10.2025.	Cijeli tim	Odrađeno
Dijagrami obrazaca uporabe	9.11.2025.	Niko Knežević, Frane Ćevid, Ivan Žalac	Odrađeno
Nabranje funkcionalnih zahtjeva	28.10.2025.	Niko Knežević	Odrađeno
Planiranje dizajna	28.10.2025.	Mihael Rošić	Odrađeno
Sekvencijski dijagrami	9.11.2025.	Marko Bošnjak, Niko Knežević, Frane Ćevid, Ivan Žalac	Odrađeno
Izrada osnove baze podataka sustava	9.11.2025.	Frane Ćevid	Odrađeno
Priprema za prezentiranje napretka	11.11.2025.	Cijeli tim	Odrađeno
Izrada početne stranice	9.11.2025.	Mihael Rošić	Odrađeno
Spajanje s bazom podataka	14.11.2025.	Marko Bošnjak, Frane Ćevid	Odrađeno
Razvoj registracije korisnika	14.11.2025.	Marko Bošnjak	Odrađeno
Testiranje osnovne verzije sustava	14.11.2025.	Toni Kapučija	Odrađeno
Postavljanje aplikacije na poslužitelj	11.11.2025.	Ivan Žalac, Marko Bošnjak	Odrađeno
Uspostava sustava autentifikacije	14.11.2025.	Marko Bošnjak, Marin Čikotić	Odrađeno
Kreiranje sustava za objave društvenih igara	nedefinirano	Marko Bošnjak, Marin Čikotić	Odrađeno
Funkcionalnost filtera igara	14.11.2025.	Marko Bošnjak, Toni Čikotić	Odrađeno
Moderatorski sustav	13.01.2026.	Marko Bošnjak, Marin Čikotić	Odrađeno
Testiranje alfa verzije aplikacije	13.01.2026.	Cijeli tim	Odrađeno
Implementacija osnovnih funkcionalnosti alfa verzije	13.01.2026.	Cijeli tim	Odrađeno
Implementacija sustava za pretragu objava po filterima	13.01.2026.	Marko Bošnjak	Odrađeno
Testiranje dodatnih funkcionalnosti	13.01.2026.	Toni Kapučija	Odrađeno
Testiranje finalne verzije aplikacije	23.01.2026.	Cijeli tim	Odrađeno

Tablica aktivnosti

aktivnosti	Marko Bošnjak	Marin Čikotić	Frane Ćevid	Toni Kapučija	Niko Knežević	Mihael Rošić	Ivan Žalac
Upravljanje projektom	5	5	5	5	5	5	5
Opis projektnog zadatka	3	3	3	3	3	3	3
Funkcionalni zahtjevi	1	1	1	1	2	1	5
Dijagram obrazaca	0	0	3	0	3	0	1
Sekvencijski dijagrami	0	0	1	0	2	0	2
Opis ostalih zahtjeva	0	0	0	0	1	0	2
Opis projekta	3	3	3	3	3	3	3
Arhitektura i dizajn sustava	18	3	3	3	0	12	7
Baza podataka	2	1	6	0	0	0	0
Opis baze podataka	0	0	4	0	0	0	0
Dijagram razreda	0	0	0	0	0	0	0
Dijagram stanja	0	0	0	0	2	0	0
Dijagram aktivnosti	0	0	0	0	2	0	0
Dijagram komponenti	0	0	2	0	0	0	0
Korištene tehnologije i alati	4	5	4	4	0	4	5
Ispitivanje programskog rješenja	3	2	0	10	0	0	2
Dijagram razmještaja	0	0	2	0	0	0	0
Upute za puštanje u pogon	2	0	0	0	0	0	6
Dnevnik sastajanja	0	0	0	0	3	0	0
Zaključak i budući rad	0	0	0	0	2	0	0
Popis literature	0	0	0	0	1	0	0
Izrada aplikacije	15	5	5	5	0	15	5
Izrada baze podataka	0	0	5	0	0	0	0
Spajanje s bazom podataka	3	0	3	0	0	3	0
Dokumentiranje plana projekta i uključenosti	0	0	0	0	2	0	0
Dokumentacija specifikacije obrazaca uporabe	0	0	0	0	5	0	0

Dijagram pregleda promjena

Sljka B.1 Commitovi tijekom vremena

Commits over time

Weekly from Oct 5, 2025 to Jan 18, 2026



Slika B.2 Commitovi po članu tima



Kjučni izazovi i rješenja

Tijekom rada na projektu pojavili su se neki problemi od kojih su neki bili očekivani, a neki pomalo neočekivani. Ovo je opis najznačajnijih problema kod kojih se naišlo tijekom razvoja projekta i načina na koji su riješeni:

Rokovi

Opis: najveći problem koji se pojavljivao tijekom razvoja bio je praćenje rokova i sustizanje potrebnih zadataka na vrijeme.

Primjeri: razvoj potrebnih funkcionalnosti do prve predaje projekta i razvoj funkcionalnosti te testiranje sustava do roka za predstavljanje alfa verzije aplikacije.

Rješenja: pokušaj što kvalitetnije podjele zadataka među članovima tima kako bi se sve dovršilo na vrijeme tako da svatko preuzeme odgovornost za svoju ulogu

Pronalazak vremena za rad

Opis: svi članovi tima imaju obaveza nevezanih za projekt i ponekad je bilo teško pronaći dovoljno vremena za razvoj aplikacije.

Primjeri: balansiranje uloženog vremena rada na projektu s učenjem, rješavanjem laboratorijskih vježbi i pisanjem drugih projekata (projekt R i slično).

Rješenja: kontinuirana komunikacija sa svim članovima tima kako bi se znalo koliki se napredak može očekivati od pojedinca u određenom vremenskom periodu.

Greške u odraćenom poslu

Opis: prilikom bilo kakvog rada znaju se dogoditi neočekivane greške koje zahtijevaju ispravak nečeg već napravljenog.

Primjer: greška prilikom spajanja baze podataka s backendom.

Rješenje: ponovni rad na stvari koja je izazvala problem i detaljan pregled greške kako bi se ustanovilo što ju uzrokuje.

Razumijevanje zahtjeva

Opis: tijekom rada pojavljivali se se problemi razmijevanja zahtjeva aplikacije, što uzrokuje zastoj u radu jer se ne može nastaviti dok se ne razjasni što se očekuje od pojedinog segmenta sustava.

Primjer: nedovoljno razumljiv opis načina na koji treba raditi sustav geolociranja.

Rješenje: kontaktiranje s asistentom uživo ili putem maila s unaprijed pripremljenim pitanjima svih članova grupe, ali samo kad se utvrdi da niti jedan član tima ne razumije zahtjeve.

Programsko inženjerstvo ak.god 2025./2026.

Sveučilište u Zagrebu

Fakultet elektrotehnike i računarske 808