> Dobrodošli na Programsko-inzenjerstvo wiki!

Programsko inženjerstvo ak.god 2025./2026.

Sveučilište u Zagrebu

Fakultet elektrotehnike i računarstva

StanPlan

Tim: <TG09.4>

- Jakov Jauk - Backend
- Borna Matković - Frontend
- Ivano Markić - Backend
- Luka Matešković - Backend
- Tea Poplašen - Backend
- Dorotea Perković - Frontend / Voditelj

Ime tima: Nexify

Nastavnik: Vlado Sruk

Cilj projekta

Cilj projekta "StanPlan" je uspostaviti moderno, digitalno rješenje namijenjeno proceduralnom upravljanju, formalizaciji i arhiviranju odluka unutar višestambenih zgrada. Sustav omogućuje predstavniku suvlasnika nadzor svih koraka u procesu odlučivanja: od organizacije i objave sastanaka do generiranja pravno valjanih zaključaka. Implementacijom automatiziranog tijeka rada, smanjuje se administrativna kompleksnost i osigurava pravna utemeljenost akata. "StanPlan" izravno podržava primjenu novih zakonskih regulativa i povećava transparentnost. Kroz integraciju s aplikacijom "StanBlog", uspostavlja se veza između neformalne komunikacije i formalnog donošenja odluka.

Krajnji cilj je isporučiti integriranu platformu koja omogućuje precizno i transparentno vođenje proceduralnih aspekata zajedničkog vlasništva, što je ključno za postizanje potpune usklađenosti s pravnim okvirom upravljanja stambenim zgradama.

Prednosti projekta

Aplikacija "StanPlan" donosi niz ključnih prednosti za predstavnike stanara, suvlasnike i upravitelje zgrada:

1. **Povećana organiziranost**
   Centralizacijom svih sastanaka, dnevnih redova i zaključaka u jedinstvenom sustavu, aplikacija znatno smanjuje potrebu za ručnim vođenjem bilješki, komunikacijom putem e-pošte i oglasnih ploča. Automatizirani prijelazi između stanja sastanka omogućuju predstavniku jasnu kontrolu nad procesom.

2. **Bolja komunikacija među suvlasnicima**
   "StanPlan" omogućuje suvlasnicima pravovremeni uvid u sve objavljene sastanke, dnevne redove i zaključke, uz automatsko obavještavanje putem e-pošte. Time se potiče veća uključenost suvlasnika i smanjuje nesporazum koji nastaje zbog zakašnjelih informacija ili nejasne komunikacije.

3. **Poboljšana transparentnost odluka**
   Zahvaljujući jasno definiranom procesu glasanja i evidenciji zaključaka s pravnim učinkom, aplikacija osigurava potpunu transparentnost svih odluka donesenih na sastancima. Svaki suvlasnik ima uvid u rezultate glasanja i arhivirane zapisnike.

4. **Jednostavnije praćenje i arhiviranje**
   Digitalno spremanje svih sastanaka i njihovih zaključaka omogućuje dugoročnu pohranu i

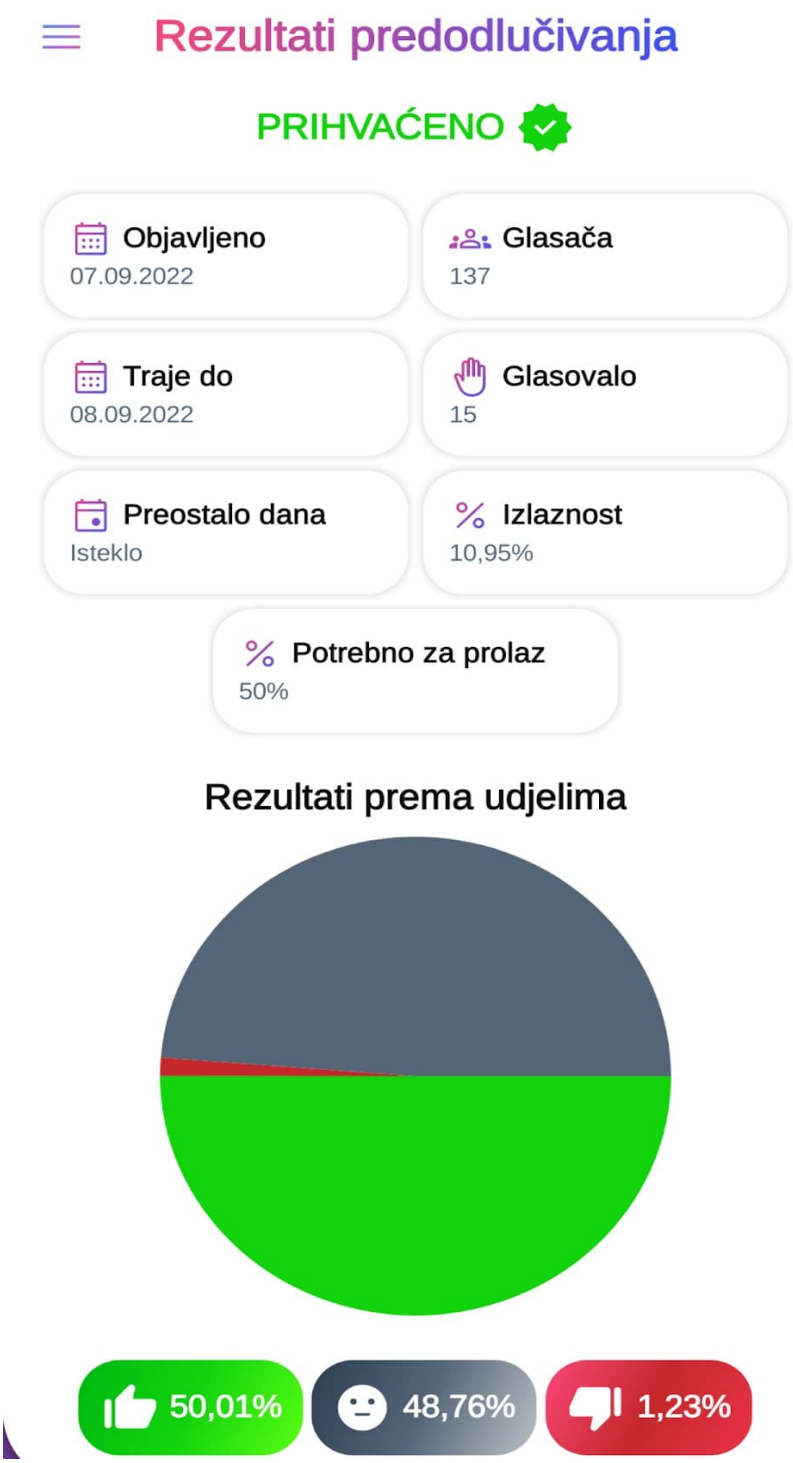
jednostavno pretraživanje povijesnih podataka. Predstavnici i suvlasnici lako mogu pronaći ranije odluke ili prijedloge, što olakšava planiranje budućih aktivnosti.

5. **Integracija s drugim sustavima**
   Kroz integraciju s aplikacijom "StanBlog", "StanPlan" omogućuje povezivanje točaka dnevnog reda s raspravama i prijedlozima iz drugih digitalnih izvora. Time se stvara povezana i informacijski bogata platforma koja olakšava donošenje odluka na temelju prethodnih diskusija

Postojeća slična rješenja

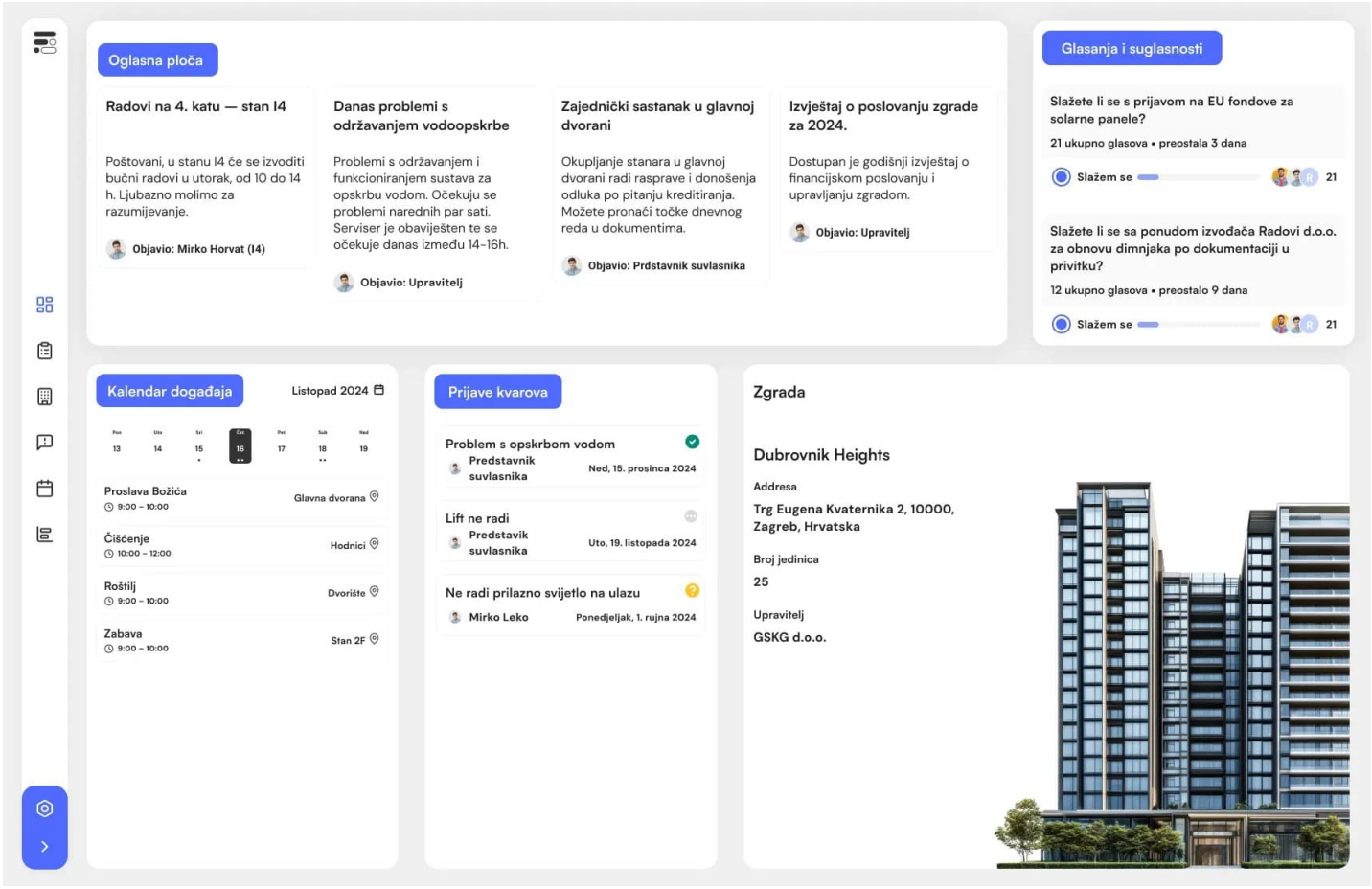
mSuvlasnik

mSuvlasnik je digitalni sustav koji korisnicima omogućuje uvid u tekuće obveze, dugove, pretplate i buduća zaduženja povezana s upravljanjem zajedničkim prostorima u stambenim zgradama. Sustav omogućuje predstavniku suvlasnika slanje relevantnih obavijesti te pokretanje anketa među suvlasnicima.
Međutim, takav način upravljanja može dovesti do nesporazuma jer ne omogućuje izravnu i dvosmjernu komunikaciju među suvlasnicima prilikom glasanja i ostaje na razini neformalnijih anketa.



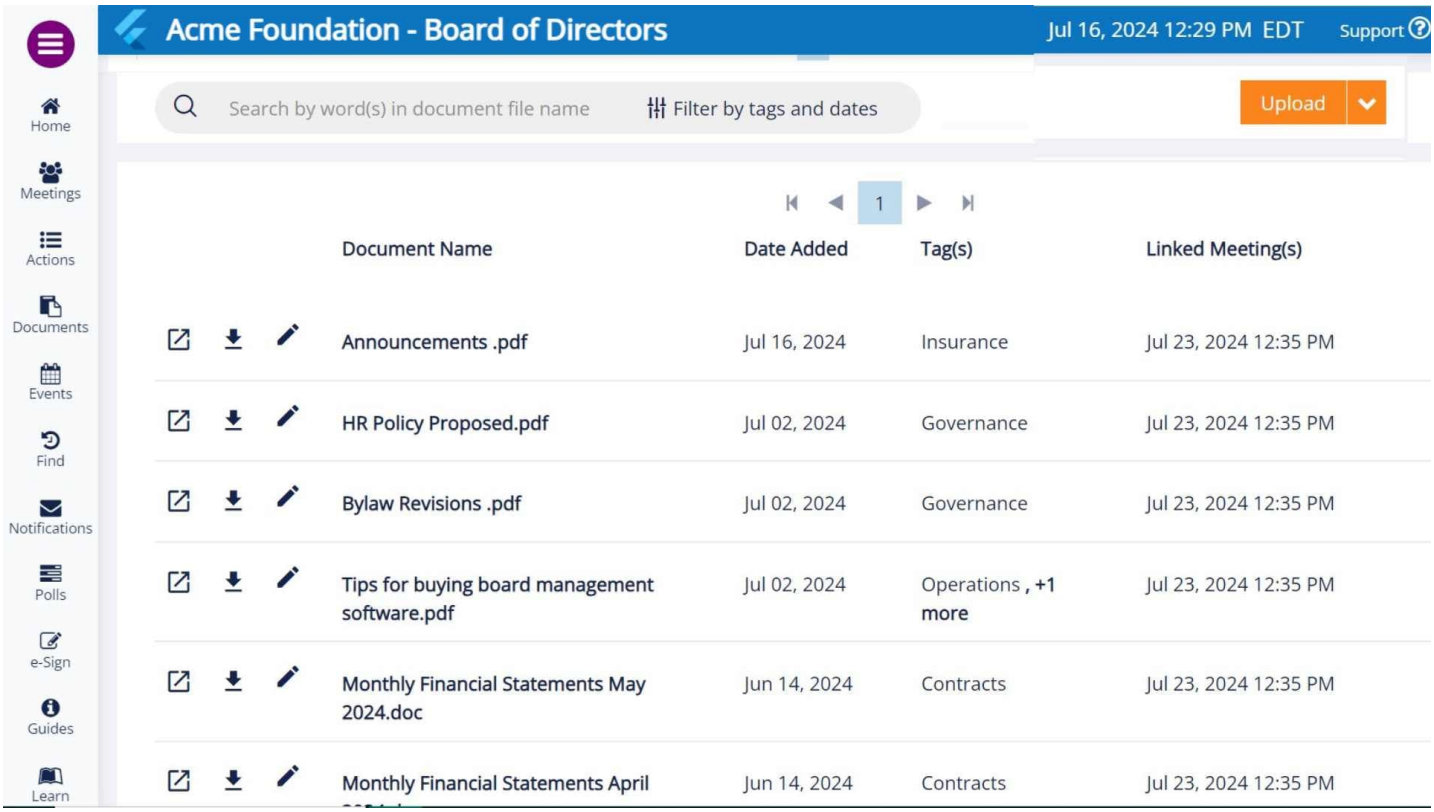
Flatie

Flatie je web-aplikacija koja korisnicima omogućuje uvid u financijsko poslovanje stambene zgrade, te pruža digitalnu oglasnu ploču, mogućnost prijave kvarova i slanje važnih obavijesti. Iako Flatie učinkovito pokriva financijski i informativni aspekt upravljanja, nedostaje mu naglašeniji društveni i interaktivni element koji bi poticao i olakšao izravnu razmjenu mišljenja i zajedničko donošenje odluka.



BoardSpace

BoardSpace je online alat namijenjen upraviteljima i odborima stambenih zajednica za organizaciju sastanaka, pohranu dokumenata i praćenje zadataka. Omogućuje dijeljenje materijala i automatske obavijesti članovima odbora. Ipak, aplikacija je više prilagođena formalnom radu uprave nego svakodnevnoj komunikaciji i uključivanju svih suvlasnika u odlučivanje.



Ključne razlike

"StanPlan" objedinjuje ključne procese upravljanja stambenim zgradama – od planiranja i vođenja sastanaka, dodavanja točaka dnevnog reda i bilježenja zaključaka do automatskog obavještavanja suvlasnika. Za razliku od postojećih rješenja koja se uglavnom fokusiraju na financijsko praćenje ili osnovne obavijesti, "StanPlan" naglašava formalno vođenje sastanaka s pravnim učinkom te transparentnu evidenciju odluka. Istovremeno potiče aktivnu komunikaciju i suradnju među suvlasnicima, čime se smanjuju nesporazumi i povećava učinkovitost zajedničkog odlučivanja.

Skup korisnika

Skup korisnika aplikacije **StanPlan** obuhvaća:

- **Predstavnik suvlasnika** - glavni korisnik aplikacije koji organizira i planira sastanke, vodi dnevni red i objavljuje zaključke
- **Suvlasnici** - korisnici koji pregledavaju sastanke i zaključke te potvrđuju sudjelovanje
- **Administrator sustava** - dodjeljuje korisničke račune, dodjeljuje pristupne popdatke, održava sustav te konfigurira integraciju s drugim aplikacijama

Zainteresirane skupine

Zainteresirane skupine za aplikaciju **StanPlan** obuhvaćaju pojedince i organizacije uključene u upravljanje, održavanje i suradnju unutar stambenih zgrada:

- **Stambene zajednice** - zgrade s više suvlasnika koje traže učinkovitiji način komunikacije, organizacije i donošenja odluka
- **Predstavnici suvlasnika** - osobe koje koordiniraju aktivnosti suvlasnika
- **Tvrtke za upravljanje nekretninama** - profesionalni upravitelji koji nadziru više zgrada i koriste aplikaciju za organizaciju poslova
- **Servisne i održavateljske tvrtke** - poduzeća koja izvode radove na zgradama te mogu biti uključena u procese planiranja i prijave održavanja
- **Pravne službe i administratori** - stručnjaci koji nadziru ispravnost donošenja odluke
- **Stanari i suvlasnici** - krajnji korisnici
- **Lokalne uprave i komunalna poduzeća** - potencijalni partneri kod digitalizacije sustava upravljanja višestambenim objektima
- **IT firme** - tehnološki partneri koji mogu razvijati dodatne module ili integracije

Mogućnost prilagodbe rješenja

Aplikacija **StanPlan** osmišljena je tako da bude fleksibilna i mogućnosti njene nadogradnje su brojne:

- **Proširenje korisničkih uloga** - sustav može biti proširen novim vrstama korisnika (npr. serviseri, upravitelji zgrada itd.) ovisno o potrebama pojedine zgrade ili stambene zajednice
- **Integracija s dodatnim aplikacijama** - zahvaljujući definiranim API-ima, moguće je jednostavno povezivanje s drugim servisima(npr. aplikacija za upravljanje financijama zgrade)
- **Podrška za različite oblike autentifikacije** - sustav podržava integraciju s vanjskim servisima za prijavu(npr. OAuth 2.0), čime se omogučuje jednostavnije i sigurnije upravljanje pristupom korisnika
- **Prilagodba jezika i izgleda** - sučelje aplikacije može se lokalizirati na više jezika i dizajnirati prema potrebama krajnjih korisnika

Opseg projekta

Opseg projektnog zadatka obuhvaća razvoj aplikacije za digitalnu podršku u radu predstavnika stanara i koordinaciji suvlasnika u stambenim zgradama. Aplikacija omogućava učinkovito planiranje i praćenje sastanaka, glasanja te integraciju s vanjskim servisom StanBlog za diskusije. Projekt obuhvaća razvoj aplikacije koja podržava sljedeće funkcionalnosti:

Upravljanje korisnicima

- administrator ima mogućnost kreirati nove korisnike, definirati njihove podatke (korisničko ime te lozinka) i adrese elektroničke pošte te dodati URL adresu vanjskog servisa (StanBlog)
- korisnici se mogu prijaviti u sustav putem jednostavne autentifikacije te promijeniti svoju inicijalnu lozinku koristeći prethodnu lozinku

Upravljanje sastancima

Predstavnik stanara ima mogućnost:

- kreirati novi sastanak te definirati naslov, sažetak, vrijeme i mjesto održavanja tog sastanka
- dodavati točke dnevnog reda (s ili bez pravnog učinka)
- označiti točke na kojima će se provoditi glasovanje
- mijenjati stanje sastanka između faza: „Planiran", „Objavljen", „Obavljen" i „Arhiviran", ovisno o njegovom tijeku i statusu izvršenja
- unositi zaključke i označiti ishod glasanja kada je sastanak „Obavljen"
- povezivati točke s diskusijama iz aplikacije StanBlog

Interakcija suvlasnika

Suvlasnici mogu:

- pregledavati sastanke u stanjima „Objavljen" i „Arhiviran"
- potvrditi sudjelovanje na nadolazećem sastanku
- primati obavijesti o objavi i arhiviranju sastanaka putem elektroničke pošte

Interakcija sa StanBlog aplikacijom

- omogućuje se preuzimanje liste diskusija iz StanBlog aplikacije
- povezivanje pojedine točke dnevnog reda s postojećom diskusijom
- StanBlog aplikacija može automatski kreirati sastanak s jednom točkom dnevnog reda

Moguće nadogradnje projektnog zadatka

U osnovnoj verziji ova aplikacija omogućava samo osnovne funkcionalnosti vođenja zgrade. Moguće su brojne nadogradnje kojima bi proširili funkcionalnosti i približili se potpunom digitalnom upravljanju zgradama. Neke od njih su:

- Sustav elektroničkog glasanja - omogućava suvlasnicima glasanje putem korisničkog sučelja ako ne mogu pristupiti sastanku uživo, čime se povećava sudjelovanje i transparentnost odluka

- Financijsko upravljanje - omogućilo bi pregled i praćenje financijskih tokova pričuve u stvarnom vremenu, čime bi se smanjila potreba za ručnim izvještavanjem
- Interakcija s dodatnim aplikacijama - uz postojeću interakciju sa StanBlog aplikacijom bile bi moguće i interakcije sa sustavima za online plaćanjem, upraviteljskim servisima zgrada i sličnim aplikacijama

Funkcionalni zahtjevi

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
F-001	Prijava korisnika prema ulozi (predstavnik suvlasnika, suvlasnik, administrator)	Visok	Zahtjev dionika	Svaki dionik može se prijaviti vlastitim korisničkim imenom i lozinkom; sustav dodjeljuje rolu pri prijavi.
F-002	Kreiranje sastanka s naslovom, sažetkom, vremenom i mjestom (sastanak je tada u stanju "Planiran")	Visok	Dokument zahtjeva	Predstavnik suvlasnika može popuniti sve podatke i kreirati sastanak; sastanak je odmah vidljiv u sustavu.
F-003	Dodavanje i uređivanje točaka dnevnog reda sastanku koji je u stanju "Planiran"	Srednji	Dokument zahtjeva	Predstavnik suvlasnika može dodati ili urediti točke dnevnog reda za svaki sastanak koji je u stanju "Planiran".
F-004	Označavanje točaka dnevnog reda s pravnim učinkom (ili bez njega)	Srednji	Zahtjev dionika	Predstavnik suvlasnika u aplikaciji StanBlog može označiti svaku točku kao pravno relevantnu; oznaka je vidljiva u pregledu sastanka.
F-005	Objavljivanje sastanka korisnicima nakon pripreme	Visok	Zahtjev dionika	Predstavnik suvlasnika može objaviti sastanak; korisnici primaju obavijest o novom objavljenom sastanku.
F-006	Pregled broja potvrđenih sudionika za sastanak	Srednji	Povratne informacije korisnika	Predstavnik suvlasnika može pregledati trenutni broj potvrđenih sudionika za svaki sastanak.
F-007	Prevođenje sastanka u stanje „Obavljen“ nakon završetka održavanja	Srednji	Dokument zahtjeva	Predstavnik suvlasnika može promijeniti status sastanka iz „Objavljen“ u „Obavljen“; status je ažuriran u prikazu.
F-008	Unos zaključaka za svaku točku dnevnog reda	Srednji	Dokument zahtjeva	Predstavnik suvlasnika može unijeti zaključak za svaku točku dnevnog reda; zaključci su dostupni svim sudionicima.

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
F-009	Arhiviranje sastanka kad su svi zaključci dodani za pravno relevantne točke	Srednji	Postojeći sustav	Sastanak se može arhivirati automatski ili ručno kad svi zaključci za pravne točke postoje; arhiva je dostupna za pregled.
F-010	Primanje elektroničkih obavijesti o promjenama statusa sastanka	Srednji	Povratne informacije korisnika	Kad predstavnik suvlasnika promijeni status sastanka, svi relevantni korisnici dobivaju email ili notifikaciju u aplikaciji.
F-011	Povezivanje točaka dnevnog reda s raspravama iz aplikacije StanBlog	Srednji	Zahtjev dionika	Predstavnik suvlasnika i aplikacija StanBlog mogu svakom sastanku i točki dodati poveznicu na raspravu iz StanBlog-a; vidljivo u detaljima.
F-012	Pregled objavljenih sastanaka i pregled zaključaka sastanaka	Srednji	Povratne informacije korisnika	Suvlasnik može pregledavati objavljene sastanke i vidjeti zaključke arhiviranih sastanaka.
F-013	Kreiranje i upravljanje korisničkim računima	Visok	Zahtjev dionika	Administrator može dodavati, mijenjati i brisati korisničke račune te postavljati uloge; promjene se odmah primjenjuju.
F-014	Dodjeljivanje korisničkih imena, lozinki i e-mail adresa	Srednji	Postojeći sustav	Administrator može dodati i mijenjati korisnička imena, lozinke i e-mail; korisnici se mogu prijaviti prema navedenim podacima.
F-015	Omogućavanje korisnicima promjenu lozinke	Srednji	Povratne informacije korisnika	Administrator omogućuje promjenu inicijalne lozinke; korisnici mogu odmah unijeti novu lozinku kroz aplikaciju.
F-016	Konfiguracija adrese poslužitelja aplikacije StanBlog radi integracije	Srednji	Povratne informacije korisnika	Administrator može unijeti i izmijeniti adresu StanBlog poslužitelja; adresa se koristi za API integraciju.
F-017	Dohvat popisa rasprava povezanih s dnevnim redom putem API-ja	Srednji	Postojeći sustav	StanBlog aplikacija automatski dohvaća i prikazuje rasprave vezane uz točke dnevnog reda kroz API; popis je ažuriran.
F-018	Pohrana podataka o korisnicima, sastancima, točkama, zaključcima, raspravama	Visok	Postojeći sustav	Baza podataka automatski bilježi i čuva sve podatke; ovlašteni korisnici mogu pristupiti relevantnim zapisima.

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
F-019	Potvrda sudjelovanja na nadolazećem sastanku	Srednji	Povratne informacije korisnika	Suvlasnici mogu označiti da će sudjelovati na sastanku koji je u stanju "Objavljen".

Ostali zahtjevi

Performanse

ID ZAHTJEVA	OPIS	PRIORITET
NF-PERF-001	Sustav mora podržavati minimalno 50 korisnika.	Visok
NF-PERF-002	Sustav mora podržavati minimalno 120 sastanaka godišnje.	Visok
NF-PERF-003	Sustav mora odgovarati na korisničke zahtjeve u prihvatljivom vremenu, posebno kod učitavanja sastanaka i slanja obavijesti.	Visok

---

Korisničko iskustvo

ID ZAHTJEVA	OPIS	PRIORITET
NF-UX-001	Sustav mora biti responzivan i pružati dosljedne performanse na svim uređajima, uključujući stolna računala, tablete i pametne telefone.	Visok
NF-UX-002	Sustav mora korisnicima pružati jasne poruke o pogreškama i smjernice kada naiđu na probleme (npr. krivi unosi).	Srednji
NF-UX-003	Sustav mora imati intuitivno i jednostavno korisničko sučelje koje minimizira krivulju učenja za sve korisničke uloge.	Visok

---

Pouzdanost

ID ZAHTJEVA	OPIS	PRIORITET

ID ZAHTJEVA	OPIS	PRIORITET
NF-REL-001	Sustav mora biti dostupan većinu vremena, s minimalnim prekidima zbog održavanja.	Visok
NF-REL-002	Sustav mora omogućiti automatski oporavak od nepredviđenih grešaka i minimalan gubitak podataka.	Visok
NF-REL-003	Sustav mora uključivati automatske sigurnosne kopije podataka svakih 24 sata kako bi se spriječio gubitak podataka.	Visok
NF-REL-004	Sustav se mora oporaviti od kvarova (npr. pada poslužitelja) unutar 10 minuta putem automatiziranih mehanizama oporavka.	Visok

---

Standardi kvalitete

ID ZAHTJEVA	OPIS	PRIORITET
NF-QUAL-001	Sustav će biti razvijen korištenjem standardiziranih okvira i biblioteka kako bi se osigurala konzistentnost i održivost.	Srednji
NF-QUAL-002	Sustav mora biti u skladu sa standardima ISO/IEC 25010 za kvalitetu softvera.	Visok
NF-QUAL-003	Sustav mora osigurati dvosmjernu komunikaciju sa StanBlog aplikacijom.	Visok

---

Sigurnost

ID ZAHTJEVA	OPIS	PRIORITET
NF-SEC-001	Svi korisnici (administrator, predstavnik, suvlasnik) moraju se prijaviti u sustav koristeći sigurnu autentifikaciju putem OAuth 2.0 protokola. Sustav ne pohranjuje lozinke u otvorenom obliku.	Visok
NF-SEC-002	Korisnicima su dostupne samo funkcionalnosti koje odgovaraju njihovoj ulozi (npr. suvlasnik ne može kreirati sastanke niti uređivati zaključke).	Visok
NF-SEC-003	Sav promet između klijenata, poslužitelja i vanjskih API servisa mora biti šifriran (HTTPS/TLS). Podaci o sastancima i korisnicima osigurani su modernim standardima šifriranja.	Visok

Zahtjevi za održavanje

ID zahtjeva	Opis	Prioritet
NF-3.1	Sustav treba biti oblikovan tako da omogućuje jednostavno održavanje.	Visok
NF-3.1.7	Sustav treba imati dovoljnu dokumentaciju.	Visok
NF-3.1.7.1	Kôd sustava treba biti dokumentiran prema "Code Conventions for the Java Programming Language" dostupnim na Oracle.	Visok
NF-3.1.7.2	Sustav treba biti opisan putem dokumenta oblikovanja /SRS/.	Visok
NF-3.1.7.3	Sustav treba biti popraćen "Priručnikom za rad" koji opisuje pravilnu upotrebu sustava.	Visok
NF-3.1.7.4	Sustav treba imati "Plan implementacije" za pravilno postavljanje sustava.	Visok
NF-3.1.7.5	Razvoj, održavanje i nadogradnja baze podataka i API funkcionalnosti	Visok

Dionici

- Predstavnik suvlasnika
- Suvlasnik
- Administrator
- Aplikacija StanBlog
- Baza podataka
- Razvojni tim

Aktori i njihovi funkcionalni zahtjevi:

A-1 Predstavnik suvlasnika (inicijator) može:

- prijaviti se u sustav prema svojoj ulozi (F-001)
- kreirati novi sastanak s naslovom, sažetkom, vremenom i mjestom (F-002)
- dodavati točke dnevnog reda sastanku (F-003)
- označiti točke dnevnog reda kao s pravnim učinkom ili bez njega (F-004)
- objaviti sastanak kada je priprema završena (postavlja stanje na „Objavljen“) (F-005)
- pregledati broj potvrđenih sudionika za sastanak (F-006)
- prevesti sastanak u stanje „Obavljen“ nakon isteka vremena održavanja (F-007)
- unositi zaključke za svaku točku dnevnog reda (F-008)
- arhivirati sastanak ako su dodani svi zaključci za točke s pravnim učinkom (F-009)
- povezati točke dnevnog reda s raspravama iz aplikacije StanBlog (F-011)
- primati obavijesti elektroničkom poštom o promjenama statusa sastanka (F-010)

A-2 Suvlasnik (sudionik) može:

- prijaviti se u sustav prema svojoj ulozi (F-001)
- pregledavati objavljene sastanke i njihove detalje (F-012)
- potvrditi sudjelovanje na objavljenom sastanku (F-020)
- pregledavati zaključke nakon što je sastanak arhiviran (F-012)
- primati obavijesti elektroničkom poštom o objavi i arhiviranju sastanaka (F-010)

A-3 Administrator (inicijator) može:

- prijaviti se u sustav prema svojoj ulozi (F-001)
- kreirati i upravljati korisničkim računima predstavnika i suvlasnika (F-013)
- dodjeljivati korisnička imena, lozinke i adrese elektroničke pošte (F-014)
- omogućiti korisnicima promjenu inicijalne lozinke (F-015)
- konfigurirati adresu poslužitelja aplikacije StanBlog za integraciju (F-016)

A-4 Aplikacija StanBlog (inicijator/sudionik) može:

- dohvatiti popis rasprava povezanih s točkama dnevnog reda putem API sučelja (F-017)
- kreirati sastanak s jednom točkom dnevnog reda na temelju postojeće rasprave (F-002)
- označiti točku kao onu s pravnim učinkom i definirati potrebu za glasovanjem (F-004)

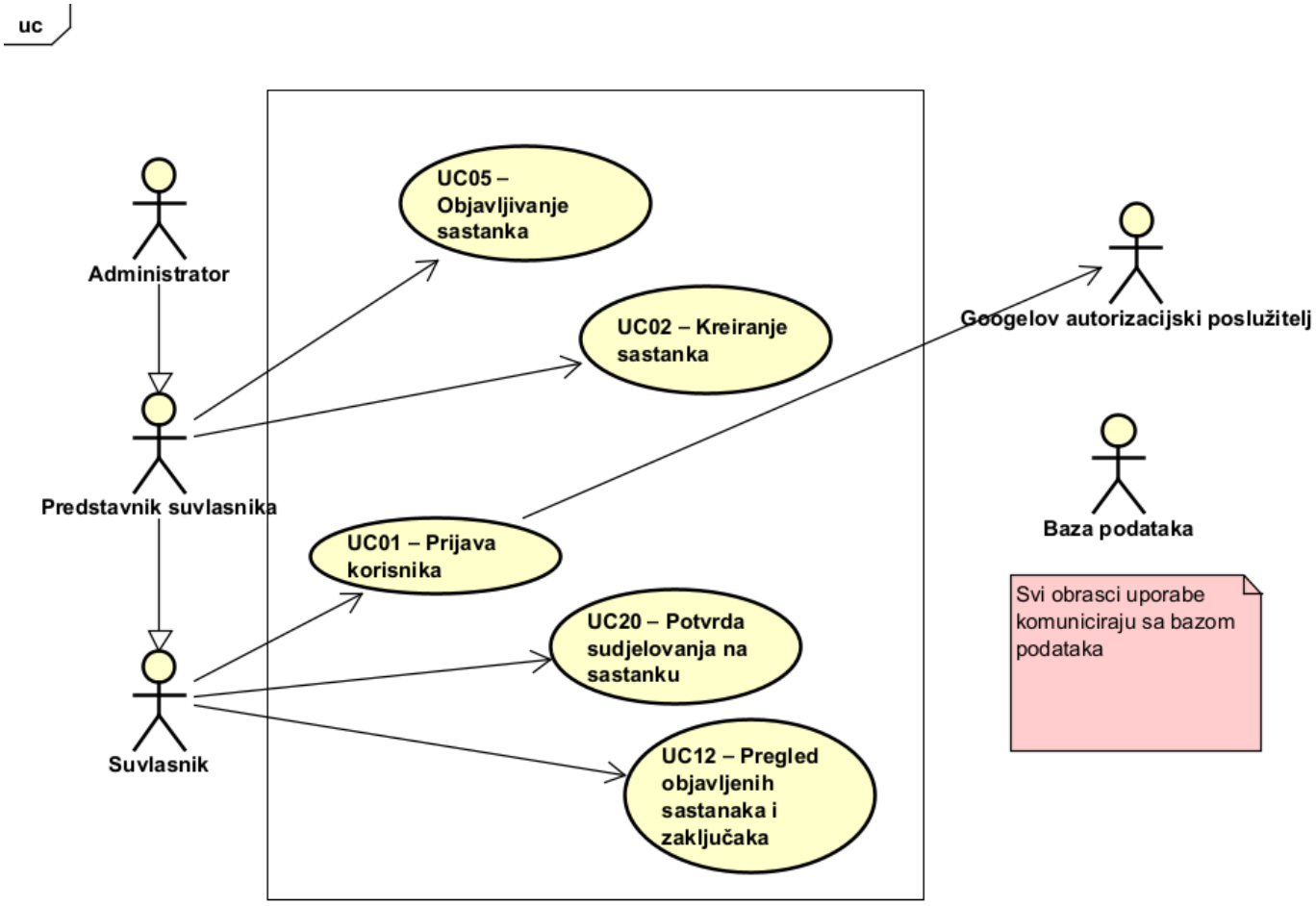
A-5 Baza podataka (sudionik) može:

- pohraniti podatke o svim korisnicima i njihovim ulogama (F-018)
- pohraniti podatke o svim sastancima i njihovim stanjima (F-018)
- pohraniti podatke o svim točkama dnevnog reda i zaključcima (F-018)
- pohraniti poveznice na rasprave iz aplikacije StanBlog (F-018)

Obrasci uporabe

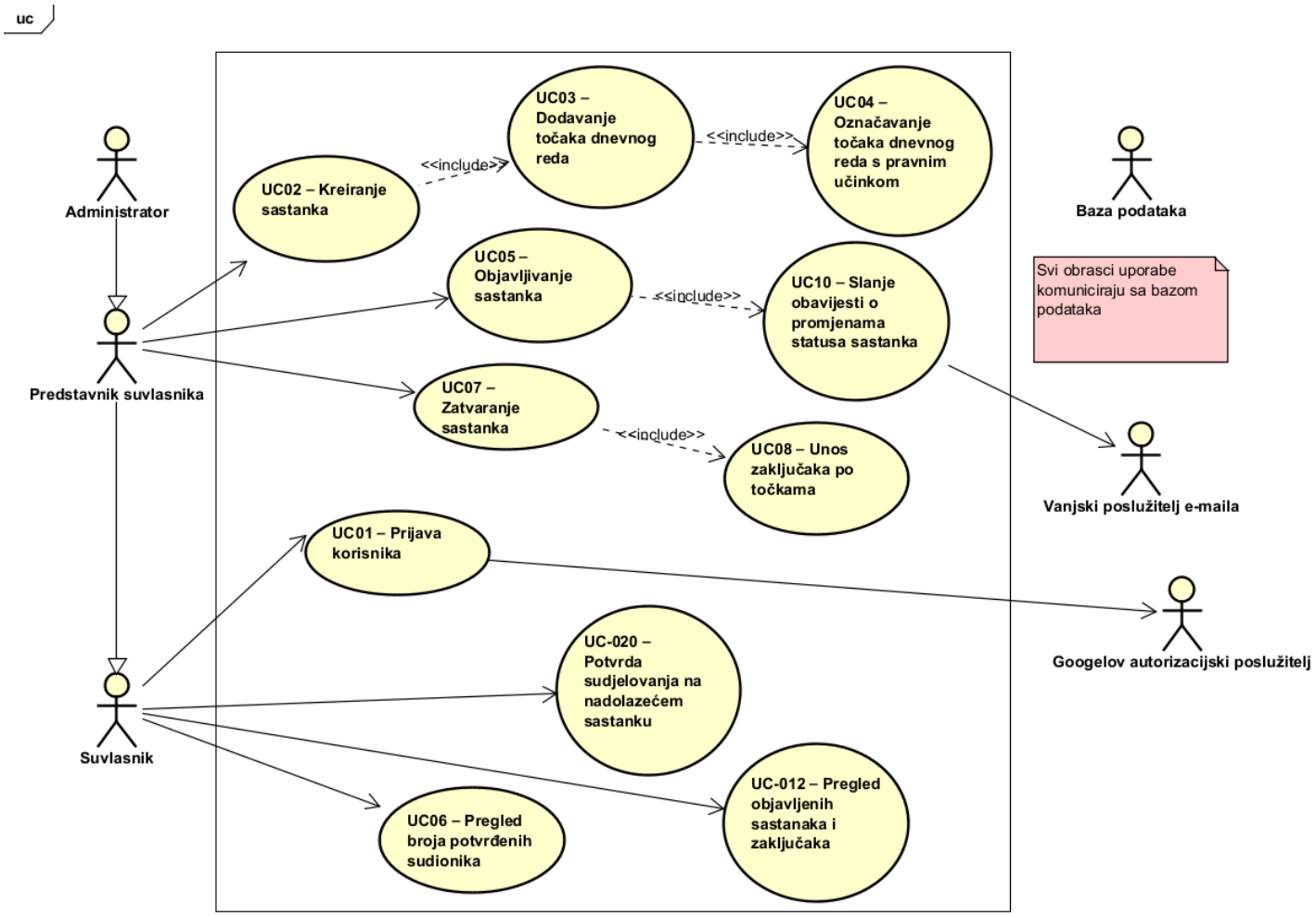
Dijagrami obrazaca uporabe

1. Visokorazinski dijagram obrazaca uporabe cijelog sustava

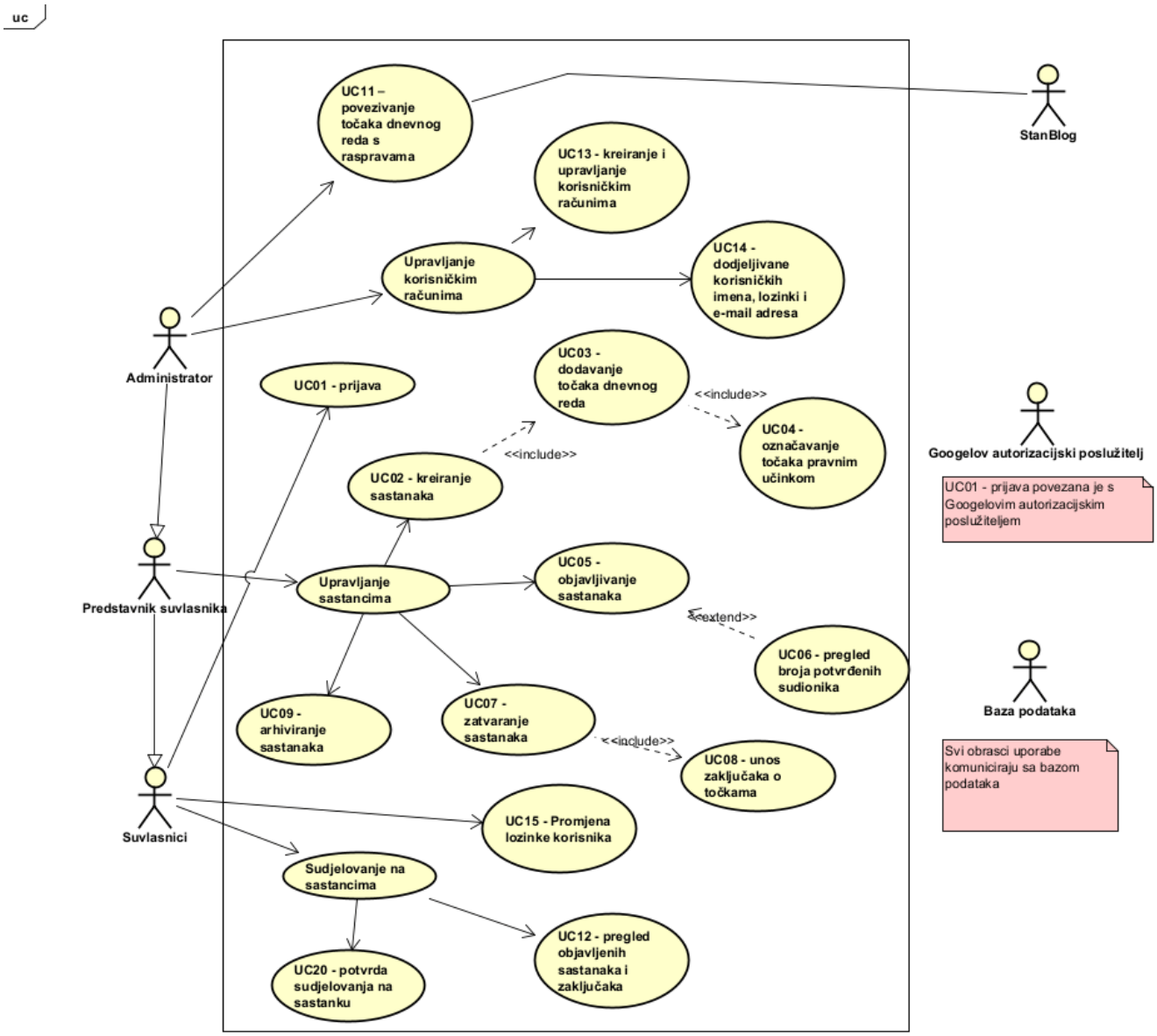


---

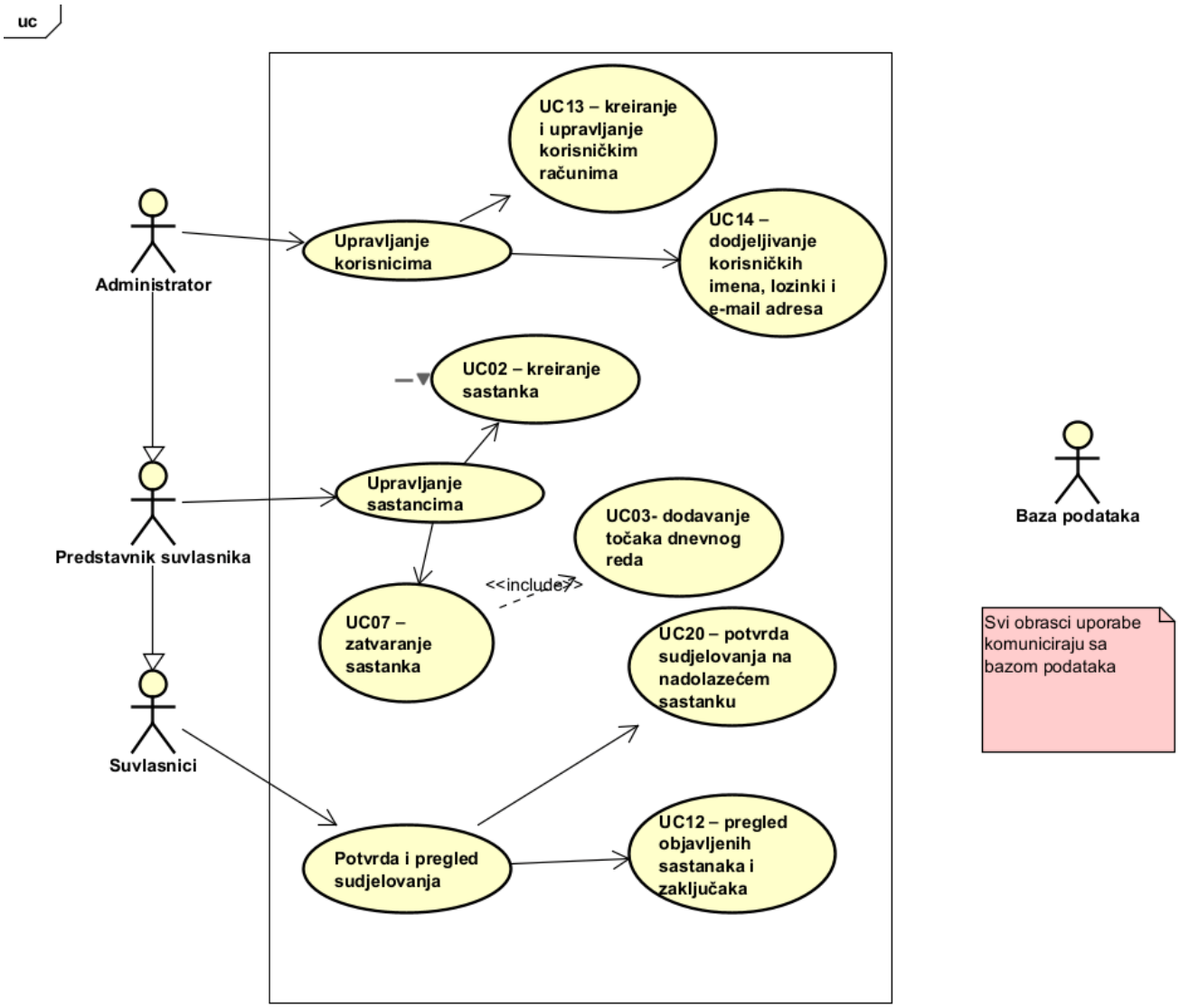
2. dijagram obrazaca uporabe za ključne značajke



3. dijagram obrazaca uporabe za korisničke uloge



4. dijagram obrazaca uporabe za osnovne poslovne procese



5. dijagram obrazaca uporabe za kritične sustave i integracije

Opis obrazaca uporabe

UC01 – Prijava korisnika

- Glavni sudionik: Predstavnik suvlasnika / Suvlasnik / Administrator
- Cilj: Omogućiti korisnicima prijavu u sustav sa svojom ulogom.
- Sudionici: Baza podataka
- Preduvjet: Korisnik ima dodijeljeno korisničko ime, lozinku i aktiviran račun.
- Opis osnovnog tijeka:

> 1. Korisnik otvara stranicu za prijavu. (F-001)

2. Upisuje korisničko ime i lozinku.
3. Sustav provjerava ispravnost podataka.
4. Ako su valjani, sustav dozvoljava ulazak u stranicu.
5. Korisnik je preusmjeren na početnu stranicu odgovarajući svojoj ulozi.

Opis mogućih odstupanja:

> • Lozinka ili korisničko ime nisu točni → Sustav prikazuje poruku o pogrešnim podacima i nudi ponovno upisivanje.

UC02 – Kreiranje sastanka

- Glavni sudionik: Predstavnik suvlasnika
- Cilj: Omogućiti kreiranje sastanka s osnovnim informacijama.
- Sudionici: Baza podataka
- Preduvjet: Korisnik je prijavljen u sustav.
- Opis osnovnog tijeka:

1. Predstavnik suvlasnika otvara obrazac „Novi sastanak“. (F-002)
2. Unosi naslov, sažetak, vrijeme i mjesto sastanka.
3. Sustav provjerava ispravnost unesenih polja.
4. Klikom na „Spremi“, sustav pohranjuje podatke u bazu.
5. Novi sastanak prikazuje se na popisu svih (planiranih) sastanaka.

Opis mogućih odstupanja:

> • Ako nedostaje obvezno polje (npr. vrijeme), sustav traži dopunu.

UC03 – Dodavanje točaka dnevnog reda

- Glavni sudionik: Predstavnik suvlasnika
- Cilj: Dodati točke dnevnog reda postojećem sastanku.
- Sudionici: Baza podataka
- Preduvjet: Sastanak je kreiran i korisnik prijavljen.
- Opis osnovnog tijeka:

1. Korisnik odabire postojeći sastanak. (F-003)
2. Klikne „Dodaj točku dnevnog reda“.
3. Unosi opis točke.
4. Sustav pohranjuje novu točku i prikazuje ju u popisu točaka.

UC04 – Označavanje točaka dnevnog reda s pravnim učinkom

- Glavni sudionik: Predstavnik suvlasnika
- Cilj: Označiti točke dnevnog reda koje imaju pravni učinak.
- Sudionici: Baza podataka
- Preduvjet: Točke dnevnog reda su dodane.
- Opis osnovnog tijeka:

> 1. Predstavnik suvlasnika otvara popis točaka. (F-004)
> 2. Označava željene točke kao pravno relevantne.
> 3. Sustav sprema oznaku i prikazuje status uz točku.
> 4. Oznaka je vidljiva svim korisnicima koji pregledavaju sastanak.

UC05 – Objavljivanje sastanka

- Glavni sudionik: Predstavnik suvlasnika
- Cilj: Objaviti sastanak nakon pripreme i učiniti ga vidljivim svim korisnicima.
- Sudionici: Baza podataka, Sustav obavijesti
- Preduvjet: Sastanak ima sve potrebne točke i podatke.
- Opis osnovnog tijeka:

> 1. Predstavnik suvlasnika otvara stranicu s detaljima sastanka. (F-005)
> 2. Klikne „Objavi sastanak“.
> 3. Sustav postavlja status sastanka na „Objavljen“.
> 4. Sustav automatski šalje obavijest svim korisnicima.
> 5. Sastanak postaje vidljiv suvlasnicima u aplikaciji.

UC06 – Pregled broja potvrđenih sudionika

- Glavni sudionik: Predstavnik suvlasnika
- Cilj: Pregledati koliko je korisnika potvrdilo sudjelovanje.
- Sudionici: Baza podataka
- Preduvjet: Sastanak je objavljen.
- Opis osnovnog tijeka:

> 1. Predstavnik suvlasnika otvara detalje sastanka. (F-006)
> 2. Sustav dohvaća broj potvrda iz baze podataka.
> 3. Broj potvrđenih sudionika prikazuje se na ekranu.

UC07 – Zatvaranje sastanka („Obavljen“ status)

- Glavni sudionik: Predstavnik suvlasnika
- Cilj: Promijeniti status završenog sastanka u „Obavljen“.
- Sudionici: Baza podataka
- Preduvjet: Sastanak je završio i ima unesene točke.
- Opis osnovnog tijeka:

> 1. Predstavnik suvlasnika otvara sastanak. (F-007)
> 2. Klikne opciju „Završi sastanak“.
> 3. Sustav mijenja status sastanka u „Obavljen“.
> 4. Završen sastanak više nije moguće uređivati.

Opis mogućih odstupanja:

> • Ako korisnik pokušava zatvoriti sastanak prije zakazanog vremena, sustav prikazuje upozorenje.

UC08 – Unos zaključaka po točkama

- Glavni sudionik: Predstavnik suvlasnika
- Cilj: Unijeti zaključke za svaku točku dnevnog reda.
- Sudionici: Baza podataka
- Preduvjet: Sastanak ima točke dnevnog reda i status „Obavljen“.
- Opis osnovnog tijeka:

> 1. Predstavnik suvlasnika otvara točke dnevnog reda. (F-008)
> 2. Odabire određenu točku i unosi zaključak.
> 3. Sustav pohranjuje zaključak.

Opis mogućih odstupanja:

> • Zaključci točaka dnevnog reda koji predlažu glasanje postavljaju stanje zaključka u „Izglasan“ ili „Odbijen“

UC09 – Arhiviranje sastanka

- Glavni sudionik: Predstavnik suvlasnika
- Cilj: Arhivirati sastanak nakon unosa svih zaključaka.
- Sudionici: Baza podataka
- Preduvjet: Sve točke s pravnim učinkom imaju zaključke.
- Opis osnovnog tijeka:

> 1. Predstavnik suvlasnika otvara završeni sastanak. (F-009)

2. Klikne „Arhiviraj sastanak“.
3. Sustav provjerava ispunjenost uvjeta (svi potrebni zaključci uneseni).
4. Sustav mijenja status sastanka u „Arhiviran“.
5. Sastanak prelazi na popis arhiviranih sastanaka.

Opis mogućih odstupanja:

> • Ako nedostaje zaključak za neku točku s pravnim učinkom, sustav onemogućuje arhiviranje i prikazuje koje točke nedostaju.

UC10 – Slanje obavijesti o promjenama statusa sastanka

- Glavni sudionik: Sustav
- Cilj: Automatski obavijestiti korisnike o statusnim promjenama sastanaka.
- Sudionici: Baza podataka
- Preduvjet: Postoji barem jedan objavljen sastanak.
- Opis osnovnog tijeka:

1. Predstavnik suvlasnika mijenja status sastanka. (F-010)
2. Sustav detektira promjenu statusa.
3. Sustav automatski generira i šalje obavijesti putem e-pošte i notifikacija.
4. Suvlasnici primaju obavijest i mogu otvoriti sastanak iz poruke.

UC11 – Povezivanje točaka dnevnog reda s raspravama

- Glavni sudionik: Administrator
- Cilj: Povezati točku dnevnog reda s odgovarajućom raspravom u sustavu StanBlog.
- Sudionici: Administrator, Sustav, StanBlog API
- Preduvjet: Administrator je prijavljen i adresa poslužitelja StanBlog je konfigurirana (UC-016).
- Opis osnovnog tijeka:

1. Administrator odabire sastanak u stanju „Planiran“. (F-011)
2. Sustav dohvaća popis rasprava iz StanBlog API-ja. (F-017)
3. Administrator odabire rasprave koje želi povezati s točkom dnevnog reda.
4. Sustav pohranjuje veze u bazu podataka. (F-018)
5. Sustav potvrđuje uspješno povezivanje i vraća pregled povezanih rasprava.

Opis mogućih odstupanja:

> • API ne odgovara / veza prekinuta → Sustav prikazuje poruku o grešci i nudi opciju „Pokušaj ponovno“.
> • Nisu pronađene rasprave → Sustav obavještava „Nema dostupnih rasprava za povezivanje“.

UC12 – Pregled objavljenih sastanaka i zaključaka

- Glavni sudionik: Suvlasnik
- Cilj: Omogućiti suvlasniku pregled objavljenih sastanaka i pripadajućih zaključaka.
- Sudionici: Suvlasnik, Sustav
- Preduvjet: Suvlasnik je prijavljen i ima pristup modulu za sastanke.
- Opis osnovnog tijeka:

> 1. Suvlasnik otvara listu objavljenih sastanaka. (F-012)
> 2. Sustav prikazuje sastanke s osnovnim podacima.
> 3. Suvlasnik odabire sastanak i otvara prikaz zaključaka.
> 4. Sustav prikazuje zaključke i povezane rasprave.
> 5. Suvlasnik zatvara pregled ili odabire drugi sastanak.

Opis mogućih odstupanja:

> • Nema objavljenih sastanaka → Sustav prikazuje poruku „Trenutno nema objavljenih sastanaka".
> • Podaci nisu dohvatljivi → Sustav prikazuje upozorenje „Greška pri dohvaćanju podataka".

UC13 – Kreiranje i upravljanje korisničkim računima

- Glavni sudionik: Administrator
- Cilj: Kreirati, uređivati i brisati korisničke račune u sustavu.
- Sudionici: Administrator, Sustav
- Preduvjet: Administrator je prijavljen.
- Opis osnovnog tijeka:

> 1. Administrator otvara modul „Korisnici". (F-013)
> 2. Odabire opciju „Dodaj korisnika".
> 3. Unosi korisničke podatke (ime, e-mail, uloga). (F-014)
> 4. Sustav provjerava valjanost podataka.
> 5. Sustav sprema novi korisnički račun u bazu. (F-018)
> 6. Sustav potvrđuje uspješno kreiranje.

Opis mogućih odstupanja:

> • Uneseni e-mail već postoji → Sustav prikazuje upozorenje i vraća administratora na uređivanje.
> • Podaci ne zadovoljavaju format → Sustav prikazuje poruku o pogrešci i traži ispravak.

UC14 – Dodjeljivanje korisničkih imena, lozinki i e-mail adresa

- Glavni sudionik: Administrator
- Cilj: Dodijeliti korisnicima pristupne podatke.
- Sudionici: Administrator, Sustav
- Preduvjet: Korisnički račun je kreiran (UC-013).
- Opis osnovnog tijeka:

> 1. Administrator otvara profil korisnika. (F-014)
> 2. Unosi korisničko ime i e-mail.
> 3. Sustav generira privremenu lozinku.
> 4. Administrator potvrđuje i pohranjuje podatke.
> 5. Sustav šalje korisniku pristupne podatke e-poštom.

Opis mogućih odstupanja:

> • E-mail adresa nije valjana → Sustav prikazuje poruku „Neispravan format e-mail adrese".
> • Slanje e-pošte nije uspjelo → Sustav bilježi pogrešku i obavještava administratora.

UC15 – Promjena lozinke korisnika

- Glavni sudionik: Korisnik
- Cilj: Omogućiti korisniku promjenu lozinke radi sigurnosti.
- Sudionici: Korisnik, Sustav
- Preduvjet: Korisnik je prijavljen u sustav.
- Opis osnovnog tijeka:

> 1. Korisnik otvara „Postavke računa". (F-015)
> 2. Odabire opciju „Promijeni lozinku".
> 3. Unosi staru i novu lozinku.
> 4. Sustav provjerava staru i validira novu lozinku.
> 5. Sustav sprema novu lozinku i potvrđuje promjenu.

Opis mogućih odstupanja:

> • Stara lozinka nije ispravna → Sustav odbija promjenu i prikazuje upozorenje.
> • Nova lozinka ne zadovoljava kriterije → Sustav prikazuje objašnjenje i traži unos jače lozinke.

UC16 – Konfiguracija adrese poslužitelja StanBlog

- Glavni sudionik: Administrator
- Cilj: Postaviti adresu poslužitelja aplikacije StanBlog radi integracije sustava.

- Sudionici: Administrator, Sustav
- Preduvjet: Administrator ima pristup postavkama sustava.
- Opis osnovnog tijeka:

> 1. Administrator otvara „Postavke integracije“. (F-016)
> 2. Unosi URL poslužitelja StanBlog.
> 3. Sustav provjerava valjanost unosa i dostupnost poslužitelja.
> 4. Sustav pohranjuje konfiguraciju.
> 5. Prikazuje se potvrda o uspješnom povezivanju.

Opis mogućih odstupanja:

> • URL nije dostupan → Sustav prikazuje poruku „Server nedostupan“.
> • API ključ nije valjan → Sustav traži unos novog ključa.

UC17 – Dohvat popisa rasprava povezanih s dnevnim redom

- Glavni sudionik: Sustav
- Cilj: Dohvatiti popis rasprava iz aplikacije StanBlog povezanih s točkama dnevnog reda.
- Sudionici: Sustav, StanBlog API
- Preduvjet: Konfigurirana i dostupna adresa API-ja (UC-016).
- Opis osnovnog tijeka:

> 1. Sustav šalje zahtjev API-ju StanBlog. (F-017)
> 2. API vraća popis rasprava.
> 3. Sustav parsira i filtrira relevantne podatke.
> 4. Sustav pohranjuje rasprave u bazu podataka. (F-018)
> 5. Sustav ažurira prikaz povezanih rasprava u sučelju.

Opis mogućih odstupanja:

> • API ne odgovara → Sustav ponavlja zahtjev ili obavještava administratora.
> • Format podataka nije valjan → Sustav bilježi grešku i ne prikazuje rezultate.

UC18 – Pohrana podataka o korisnicima, sastancima i raspravama

- Glavni sudionik: Sustav
- Cilj: Osigurati trajnu pohranu svih relevantnih podataka u bazu.
- Sudionici: Sustav, Baza podataka
- Preduvjet: Sustav ima aktivnu vezu s bazom.
- Opis osnovnog tijeka:

1. Sustav prima podatke iz korisničkog sučelja ili API-ja. (F-018)
2. Provjerava format i integritet podataka.
3. Upisuje podatke u odgovarajuće tablice baze.
4. Potvrđuje uspješno spremanje.
5. Generira zapis u dnevniku aktivnosti.

Opis mogućih odstupanja:

• Povezanost s bazom prekinuta → Sustav sprema podatke lokalno i pokušava ponovno.
• Kršenje integriteta podataka → Sustav poništava transakciju i obavještava korisnika.

UC19 – Potvrda sudjelovanja na nadolazećem sastanku

- Glavni sudionik: Suvlasnik
- Cilj: Potvrditi dolazak na planirani sastanak.
- Sudionici: Suvlasnik, Sustav
- Preduvjet: Sastanak je objavljen.
- Opis osnovnog tijeka:

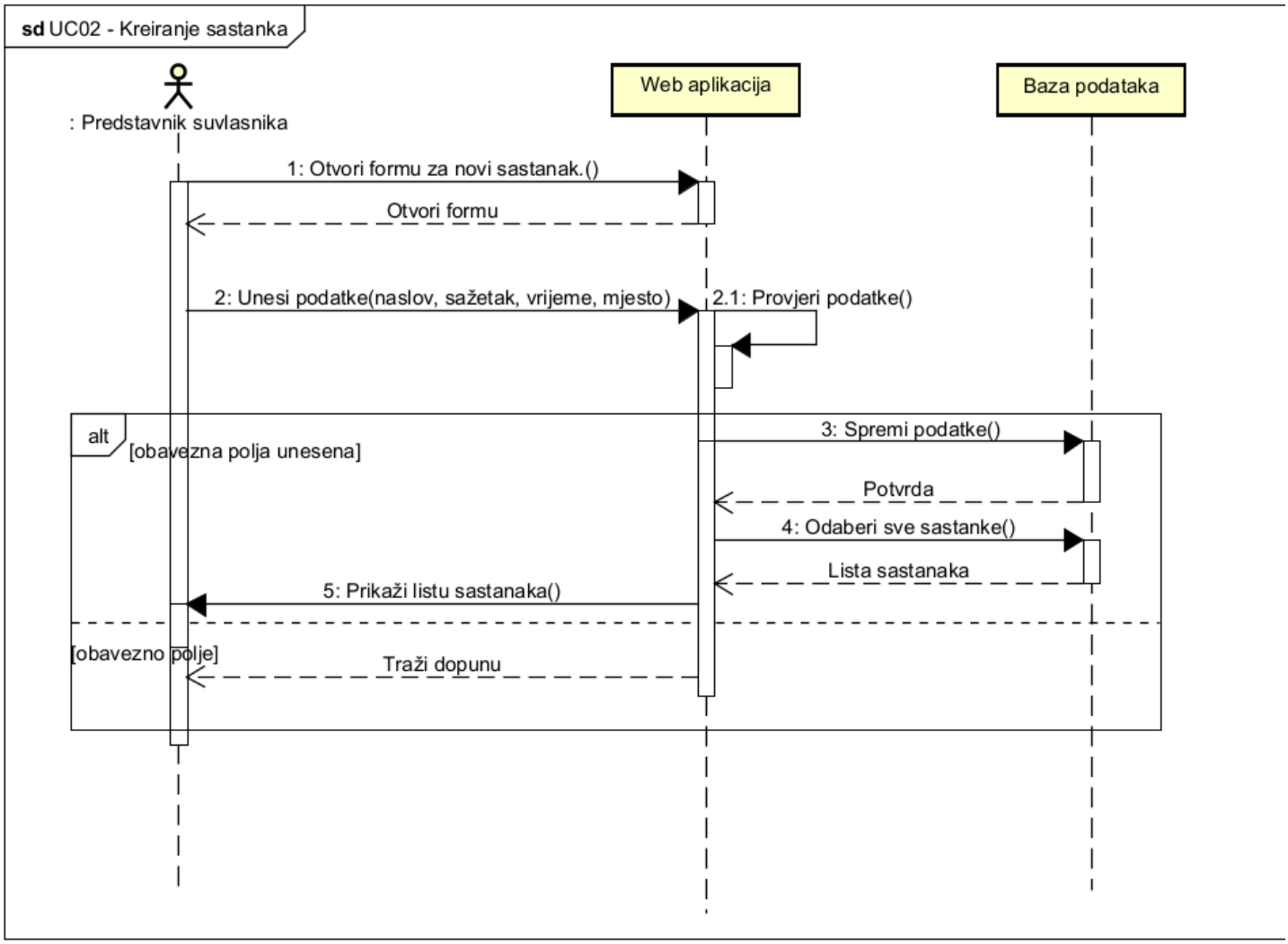
1. Suvlasnik otvara popis nadolazećih sastanaka. (F-020)
2. Odabire sastanak i klikne „Potvrdi sudjelovanje“.
3. Sustav bilježi potvrdu u bazu podataka. (F-018)
4. Sustav ažurira broj prijavljenih sudionika. (F-006)
5. Prikazuje se poruka „Sudjelovanje potvrđeno“.

Opis mogućih odstupanja:

• Korisnik je već potvrdio sudjelovanje → Sustav prikazuje poruku „Već ste potvrdili sudjelovanje“.
• Sastanak više nije aktivan → Sustav ne dopušta potvrdu i prikazuje upozorenje.

Sekvencijski dijagrami

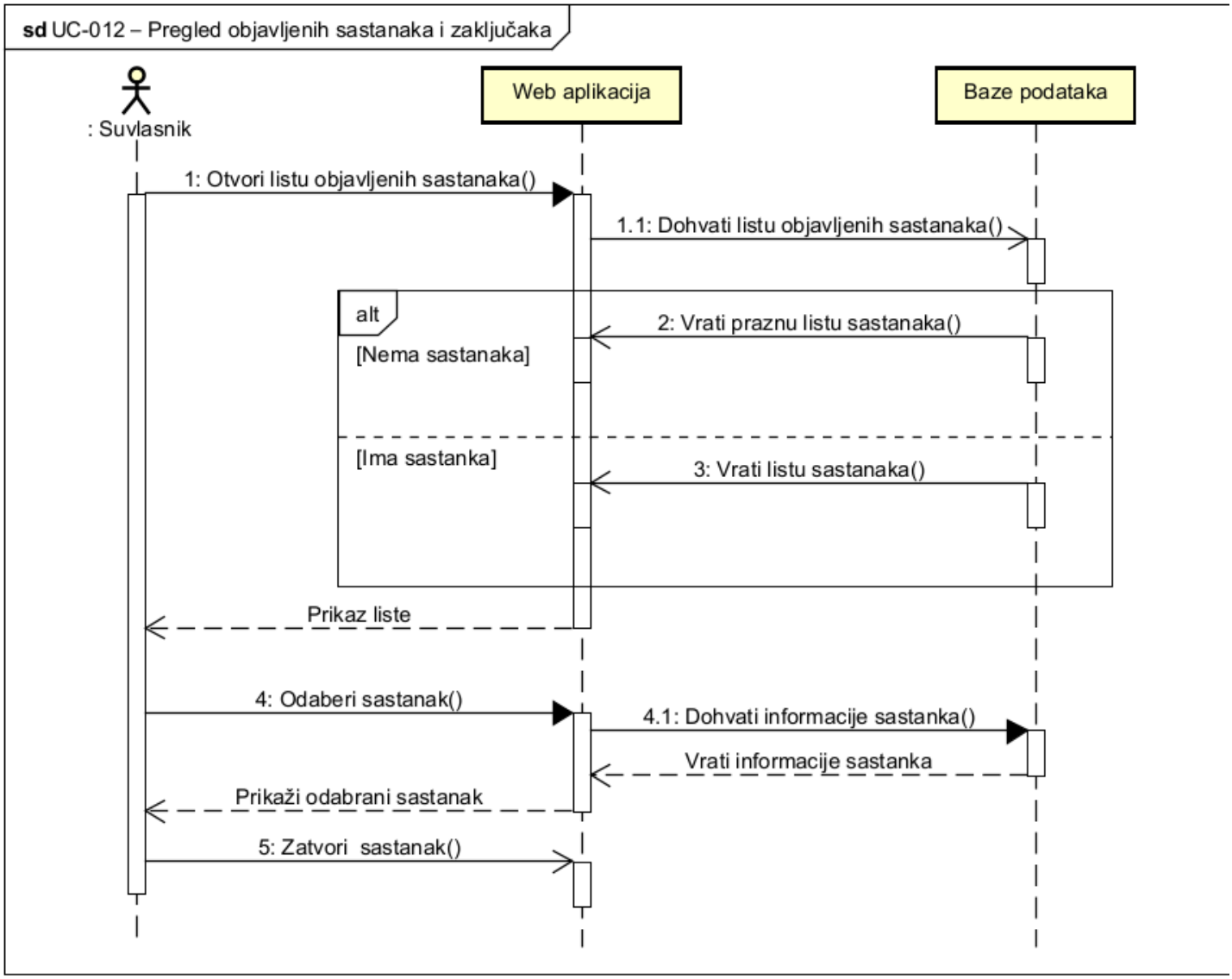
1. Sekvencijski dijagram kreiranja sastanka



Opis dijagrama

Ovaj sekvencijski dijagram bilježi postupak kreiranja novog sastanka koji inicira predstavnik suvlasnika putem web aplikacije. Predstavnik započinje odlaskom na formu "Novi sastanak" i unosi sve detalje o sastanku, kao što su naslov i vrijeme, nakon čega aplikacija odmah provjerava svaki unos kako bi se osigurala potpunost. Ako su sva obavezna polja ispunjena, sustav pohranjuje zahtjev u bazu podataka, potvrđuje uspjeh i preusmjerava korisnika na ažurirani prikaz liste svih sastanaka. Ako podaci nisu potpuni, aplikacija traži dopunu. Ovaj slijed naglašava validaciju unosa i osigurava uspješnu pohranu samo potpunih zahtjeva, pružajući korisniku neposrednu povratnu informaciju.

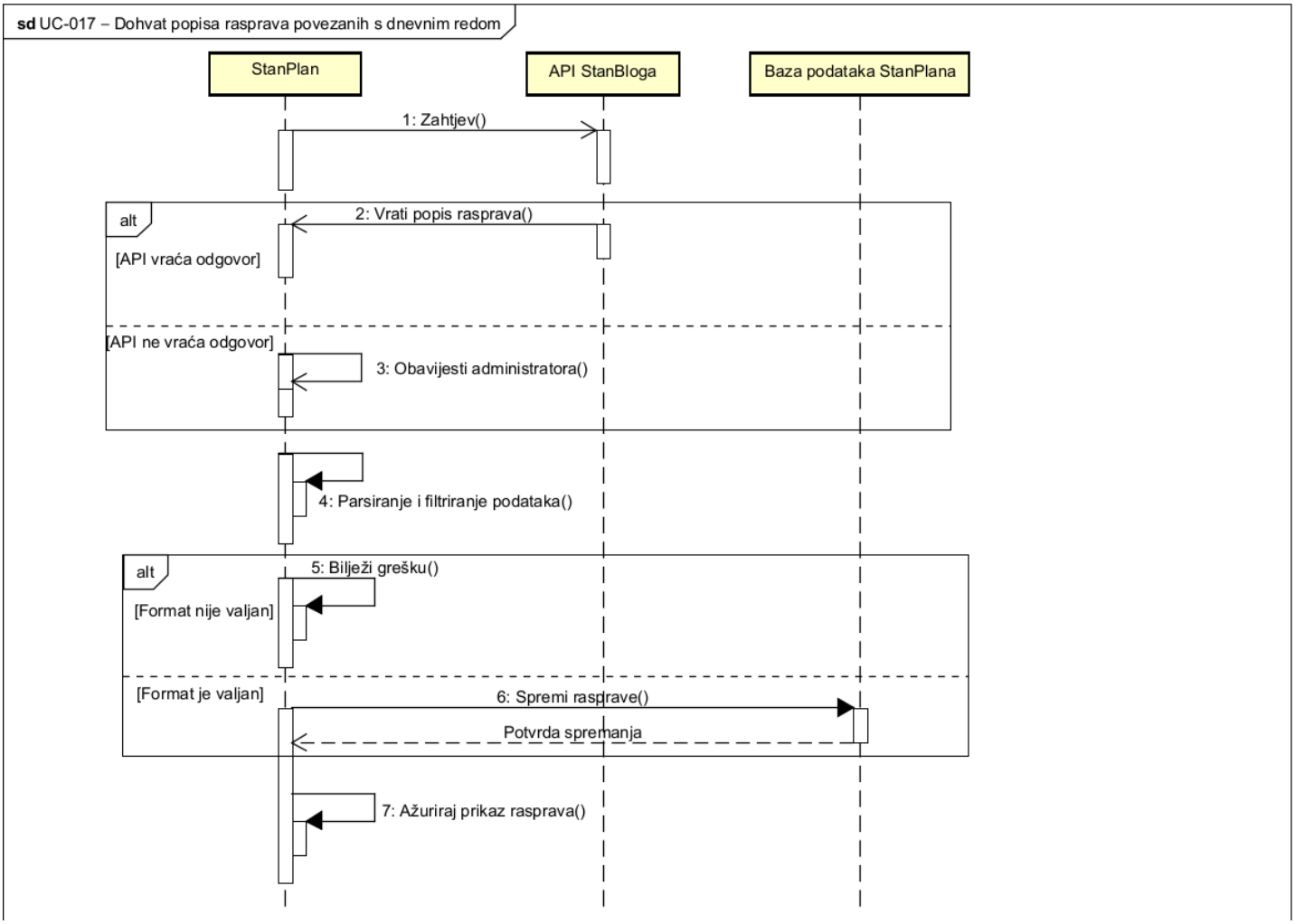
2. Sekvencijski dijagram pregleda objavljenih sastanaka



Opis dijagrama:

Ovaj dijagram prikazuje proces u kojem suvlasnik pregledava objavljene sastanke i njihove zaključke putem Web aplikacije StanPlan. Suvlasnik započinje otvaranjem stranice s listom objavljenih sastanaka i odabirom sastanka kojeg želi pregledati. Web aplikacija dohvaća podatke o sastancima iz baze i prikazuje osnovne informacije, poput datuma, vremena i naslova sastanka. Nakon što suvlasnik odabere sastanak, web aplikacija prikazuje pripadajuće zaključke i povezana raspravljanja. Ako nema objavljenih sastanaka ili podaci nisu dohvatljivi, na stranici se prikazuju odgovarajuće poruke upozorenja. Ovaj sekvencijski dijagram ilustrira ključne interakcije između suvlasnika i web aplikacije, osigurava točnost podataka i omogućuje pregledan prikaz informacija za jednostavnu i sigurnu upotrebu.

3. Sekvencijski dijagram dohvata popisa rasprava sa "StanBlog" aplikacije



Opis dijagrama:

Ovaj dijagram prikazuje proces u kojem aplikacija StanPlan dohvaća popis rasprava iz aplikacije StanBlog povezanih s točkama dnevnog reda. Web aplikacija započinje slanjem zahtjeva prema StanBlog API-ju kako bi dobio popis rasprava. API vraća podatke, a aplikacija ih parsira i filtrira relevantne informacije povezane s dnevnim redom. Nakon obrade, aplikacija StanPlan pohranjuje rasprave u svoju bazu podataka i ažurira prikaz u korisničkom sučelju, omogućujući pregled povezanih rasprava. U slučaju da API ne odgovara aplikacija obavještava administratora, ili ako su podaci neispravnog formata, sustav bilježi greške i prikazuje odgovarajuće poruke upozorenja. Dijagram jasno ilustrira ključne interakcije između aplikacija StanPlan i StanBlog, osigurava točnost podataka i omogućuje pregledan prikaz informacija korisniku.

OBRASCI UPORABE	FUNKCIONALNI ZAHTJEVI
UC01 — prijava korisnika	F-001
UC02 — kreiranje sastanka	F-002
UC03 — dodavanje točaka dnevnog reda	F-003
UC04 — označavanje točaka dnevnog reda s pravnim učinkom	F-004
UC05 — objavljivanje sastanka	F-005

OBRASCI UPORABE	FUNKCIONALNI ZAHTJEVI
UC06 — pregled broja potvrđenih sudionika	F-006
UC07 — zatvaranje sastanka („Obavljen" status)	F-007
UC08 — unos zaključaka po točkama	F-008
UC09 - arhiviranje sastanka	F-009
UC10 - slanje obavijesti o promjenama statusa sastanka	F-010
UC11 - povezivanje točaka dnevnog reda s raspravama	F-011, F-017, F-018
UC12 - pregled objavljenih sastanaka i zaključaka	F-012
UC13 - kreiranje i upravljanje korisničkim računima	F-013, F-014, F-018
UC14 - dodjeljivanje korisničkih imena, lozinki i e-mail adresa	F-014
UC15 - promjena lozinke korisnika	F-015
UC16 - konfiguracija adrese poslužitelja StanBlog	F-016
UC17 - dohvat popisa rasprava povezanih s dnevnim redom	F-017, F-018
UC18 - pohrana podataka o korisnicima, sastancima i raspravama	F-018
UC19 - potvrda sudjelovanja na nadolazećem sastanku	F-006, F-018,F-019

Arhitektura sustava

Opis arhitekture

Stil arhitekture

Odabrani arhitektonski stil za ovaj sustav je višeslojna klijent–poslužitelj arhitektura temeljena na obrascu Model–View–Controller (MVC). Ovaj stil omogućuje jasnu podjelu između različitih dijelova aplikacije te pojednostavljuje razvoj, održavanje i nadogradnju sustava. Klijent–poslužiteljski pristup temelji se na komunikaciji između korisničkog sučelja (klijenta) i poslužitelja koji obrađuje zahtjeve i upravlja podacima. Takva struktura omogućuje neovisno razvijanje i ažuriranje dijelova sustava, što je u skladu s principima separacije odgovornosti i slabe povezanosti koje smo učili na predavanjima.

U okviru MVC obrasca, Model predstavlja sloj koji upravlja poslovnom logikom i komunikacijom s bazom podataka, čime je odvojen od prikaza i korisničke interakcije. View (pogled) je zadužen za prikaz informacija korisniku i oblikovanje korisničkog sučelja, dok Controller (kontroler) povezuje model i pogled, obrađuje korisničke zahtjeve te prosljeđuje podatke između slojeva. Na ovaj način se postiže visoka kohezija unutar slojeva i slaba povezanost između njih, što povećava čitljivost i kvalitetu koda.

Ovaj arhitektonski stil je odabran jer omogućuje fleksibilnost, jednostavno testiranje i održavanje te podržava iterativan razvoj sustava, što je posebno važno u timskom radu. Uz to, višeslojna struktura olakšava primjenu principa programskog inženjerstva poput apstrakcije, modularnosti i ponovne iskoristivosti, čime sustav postaje dugoročno održiv i prilagodljiv promjenama zahtjeva korisnika.
• Podsustavi: Identificirajte glavne podsustave unutar sustava i objasnite njihove odgovornosti.
• Preslikavanje na radnu platformu: Objasnite kako će arhitektura biti implementirana na konkretnoj radnoj platformi (npr. cloud, lokalni serveri, hibridni pristup) i zašto je ta platforma odabrana.

Spremište podataka

Podatci se spremaju u relacijsku bazu podataka jer omogućuje strukturirano pohranjivanje podataka u tablice i definiranje odnosa između entiteta. Takav pristup osigurava pouzdanost, konzistentnost i jednostavno održavanje podataka. Kao tehnologija smo koristili PostgreSQL, koji je prikladan za web sustave temeljene na klijent–poslužitelj arhitekturi i lako se povezuje s poslužiteljskim dijelom aplikacije. Baza podataka je hostana na platformi Supabase, koja pruža cloud infrastrukturu za PostgreSQL baze te omogućuje jednostavno upravljanje, autentifikaciju i sigurnu pohranu podataka.

Mrežni protokoli

U ovom projektu komunikacija između klijentskog i poslužiteljskog dijela odvija se putem HTTP/HTTPS protokola i RESTful API-ja. Frontend aplikacija razvijena u Reactu komunicira s backend servisima i bazom podataka putem standardnih HTTP zahtjeva, pri čemu se podaci razmjenjuju u JSON formatu. Budući da je baza podataka smještena na Supabase platformi, pristup joj se također ostvaruje putem sigurnog HTTPS REST API-ja, koji omogućuje dohvat, spremanje i ažuriranje podataka u oblaku. Ovakav način komunikacije osigurava sigurnost, pouzdanost i jednostavnu integraciju svih dijelova sustava.

Globalni upravljački tok

Globalni upravljački tok sustava StanPlan započinje prijavom korisnika u aplikaciju. Administrator kreira korisničke račune za predstavnika suvlasnika i suvlasnike te im dodjeljuje adrese e-pošte koje se koriste za automatsko slanje obavijesti. Nakon prijave, predstavnik suvlasnika kreira novi sastanak koji je u početku u stanju „Planiran". U tom koraku unosi osnovne informacije poput naslova, opisa, vremena i mjesta sastanka te dodaje točke dnevnog reda. Kada je sastanak spreman, prelazi u stanje „Objavljen", a sustav automatski šalje e-mail obavijesti svim suvlasnicima s informacijama o održavanju sastanka.

Suvlasnici putem aplikacije mogu pregledavati objavljene sastanke, vidjeti dnevni red i označiti svoje sudjelovanje. Nakon što se sastanak održi, predstavnik ga prevodi u stanje „Obavljen" i unosi zaključke za svaku točku dnevnog reda. Kada su svi zaključci uneseni, sastanak prelazi u stanje „Arhiviran", a sustav ponovno šalje obavijest putem e-pošte svim suvlasnicima s obavještenjem da su zaključci dostupni za pregled.

Tijekom rada, aplikacija StanPlan ostvaruje komunikaciju s aplikacijom StanBlog putem vanjskog aplikacijskog sučelja.

Sklopovskoprogramski zahtjevi

Aplikacija se može pokretati na bilo kojem modernom operativnom sustavu (Windows, macOS, Linux) koji podržava web preglednik poput Chromea, Firefoxa ili Edgea. Preporučeni minimalni zahtjevi za korisničko računalo su dvojezgreni procesor, 4 GB RAM-a i stabilna internetska veza.

Za razvojni i poslužiteljski dio sustava potrebna je instalacija Node.js okruženja za pokretanje React aplikacije te podrška za REST API i pristup Supabase bazi. Budući da se baza podataka nalazi u oblaku, lokalni poslužitelj ne mora imati veliku pohranu — dovoljno je nekoliko gigabajta prostora za lokalne konfiguracijske datoteke i cache. Sustav se može jednostavno hostati na bilo kojoj cloud platformi (npr. Vercel, Netlify, Render), što ga čini lako prenosivim i skalabilnim.

Obrazloženje odabira arhitekture

Višeslojna MVC arhitektura

Višeslojni Model-View-Controller (MVC) arhitektonski dizajn dijeli aplikaciju na tri logički odvojena i funkcionalno neovisna segmenta čime se postiže ključna separacija briga koja ovaj obrazac čini tako robusnim i snažnim. Svaki sloj ima jasno definiranu ulogu: Model se brine za podatke i interakcije s bazom, View je isključivo zadužen za korisničko sučelje i prezentaciju, dok Controller djeluje kao posrednik i orkestrator koji povezuje i usmjerava ostale slojeve. Ovo razdvajanje omogućava razvojnim programerima da samostalno pristupaju, modificiraju i testiraju svaki sloj, što značajno pojednostavljuje implementaciju funkcionalnosti i otklanjanje pogrešaka. Najveća prednost je ta što greške ili promjene unutar jednog sloja ne remete funkcionalnost drugog, čime se znatno olakšava ažuriranje, održavanje i skalabilnost cijelog softverskog sustava.

Figma

Figma je odabrana kao alat za dizajn korisničkog sučelja zbog svoje pristupačnosti, suradničkih mogućnosti i jednostavnosti korištenja. Kao web-bazirani alat, Figma omogućuje više korisnika da istovremeno rade na istom projektu, što značajno olakšava timsku suradnju i komunikaciju između dizajnera i developera. Njezino intuitivno sučelje i bogat skup alata omogućuju brzo stvaranje prototipova, testiranje ideja i izradu vizualno privlačnih dizajna. Prednost Figure je i u tome što omogućuje jednostavno dijeljenje projekata te brzu izmjenu i prilagodbu elemenata bez potrebe za dodatnim softverom. Zbog svoje fleksibilnosti, brzine i učinkovitosti, Figma se pokazala idealnim rješenjem za dizajn modernih, responzivnih korisničkih sučelja.

React

React.js je odabran kao frontend tehnologija zbog svoje jednostavnosti, brzine i visoke prilagodljivosti pri izradi modernih web aplikacija. Njegova komponentna arhitektura omogućuje lako stvaranje i ponovnu upotrebu elemenata korisničkog sučelja, što znatno ubrzava razvoj i održavanje koda. Zahvaljujući virtualnom DOM-u, React učinkovito upravlja promjenama u sučelju, ažurirajući samo one dijelove koji su se promijenili, čime osigurava brzo i glatko korisničko iskustvo. Također, njegova modularnost i bogat ekosustav biblioteka omogućuju jednostavno proširivanje funkcionalnosti i prilagodbu različitim potrebama projekta. React.js se pokazao kao stabilan i dugoročno održiv izbor za razvoj dinamičnih web sučelja.

PostgreSQL

Shema relacijske baze podataka omogućuje učinkovito i dinamično upravljanje podacima, uz podršku za česta umetanja, izmjene i brisanja zapisa. Podaci su organizirani u tablice s unaprijed definiranom strukturom, što osigurava njihovu dosljednost i integritet. Takav način organizacije omogućuje brzo dohvaćanje i manipulaciju podacima unutar sustava. PostgreSQL koristi SQL, standardizirani i vrlo moćan jezik za rad s bazama podataka, koji omogućuje jednostavno oblikovanje složenih upita. Pritom se mogu koristiti mogućnosti poput spajanja tablica, filtriranja i agregiranja, što olakšava pristup i obradu podataka iz više izvora.

Django (Python)

Django je odabran za backend zbog svoje robusnosti, sigurnosti i brzog razvoja. Omogućuje jasnu organizaciju koda i jednostavno održavanje. Ugrađene funkcionalnosti poput autentifikacije, ORM-a i administracijskog sučelja ubrzavaju razvoj, dok snažan fokus na sigurnost štiti aplikaciju od najčešćih web ranjivosti. Django u kombinaciji s PostgreSQL-om osigurava stabilan, skalabilan i pouzdan backend sustav.

HTTP/HTTPS

HTTP je standardni web protokol koji omogućuje jednostavnu i učinkovitu komunikaciju između klijenta i poslužitelja. Njegov jasan model zahtjeva i odgovora osigurava kompatibilnost s različitim tehnologijama i okvirima. HTTPS, kao sigurna verzija protokola, koristi enkripciju za zaštitu razmjene podataka, čime značajno povećava sigurnost i povjerenje korisnika u aplikaciju.

JavaScript

JavaScript je ključna tehnologija za razvoj interaktivnih web aplikacija. U kombinaciji s Reactom omogućuje dinamično ažuriranje korisničkog sučelja bez ponovnog učitavanja stranice. Zahvaljujući svojoj fleksibilnosti i širokoj podršci kroz različite biblioteke i okvire, JavaScript omogućuje brzo razvijanje responzivnih i prilagodljivih web rješenja.

Supabase

Supabase je korišten kao platforma za hostanje baze podataka i upravljanje backend funkcionalnostima. Temelji se na PostgreSQL-u i pruža jednostavnu integraciju s Django aplikacijom. Omogućuje autentifikaciju korisnika, pohranu podataka u oblaku i sigurnu komunikaciju putem RESTful API-ja. Zahvaljujući svojoj pouzdanosti i jednostavnosti, Supabase osigurava stabilno i skalabilno okruženje za rad aplikacije.

Organizacija sustava na visokoj razini

1. Klijent-poslužitelj

Značajka	Opis

Značajka	Opis
Klijent	Klijent je web aplikacija StanPlan izgrađena u Reactu. Aplikacija omogućava grafičko korisničko sučelje koje korisnicima omogućava izvođenje funkcionalnosti ovisno o njihovoj ulozi. Klijent komunicira s poslužiteljem koristeći REST API. Zahtjevi se šalju preko HTTP protokola, a odgovori se vraćaju u JSON formatu, što omogućuje jednostavnu razmjenu podataka i jasnu separaciju između frontenda i backenda. Ovaj pristup omogućuje fleksibilnu integraciju novih funkcionalnosti u aplikaciju te osigurava da korisnici imaju pristup podacima i opcijama koje odgovaraju njihovoj ulozi.
Poslužitelj	Poslužitelj obrađuje zahtjeve koje šalje klijent StanPlan, implementira poslovnu logiku i izvršava operacije poput upravljanja korisnicima i njihovim ulogama, kreiranja i pregleda sastanaka, autentikacije korisnika te slanja odgovarajućih podataka u JSON formatu prema frontendu.

2. Baza podataka

Značajka	Opis
Vrsta	Relacijska baza podataka (PostgreSQL)
Svrha	Trajno pohranjivanje strukturiranih podataka uz osiguravanje integriteta i konzistentnosti, te omogućavanje izvođenja složenih SQL upita nad više povezanih tablica.
Pohranjeni podatci	Podaci o korisnicima (npr. korisnička imena, lozinke, e-mail adrese, uloge), sastancima (npr. naslovi, datumi, lokacije, statusi), obavijestima, točkama dnevnog reda, raspravama i zaključcima. Baza također pohranjuje odnose između korisnika i sastanaka putem tablice Sudjeluje, čime se omogućuje praćenje prisustva i potvrda sudjelovanja.

3. Datotečni sustav

Značajka	Opis
Značajka	Sprema datoteke povezane sa sastancima ili korisnicima, kao što su prilozi ili dokumenti vezani uz sastanke.
Integracija	Backend komunicira s datotečnim sustavom radi prijenosa, dohvaćanja i upravljanja tim datotekama. Metapodaci, poput putanja do datoteka ili URL-ova, pohranjuju se u bazu podataka.

4. Grafičko sučelje

Značajka	Opis
Vrsta	Web aplikacija napisana s pomoću React.js

Značajka	Opis
Značajke	Aplikacija omogućuje: • Unos korisnika • Stvaranje novih sastanaka • Pregled svih vrsta sastanaka • Komunikaciju s aplikacijom StanBlog
Integracija	Komunicira s backendom putem RESTful API-ja. Kada korisnik izvrši neku akciju, klijent šalje HTTP zahtjev poslužitelju, koji ga obrađuje i vraća odgovor za prikaz na sučelju.

Organizacija aplikacije

Aplikacija StanPlan organizirana je u više slojeva, što omogućuje jasno razdvajanje odgovornosti i olakšava održavanje.

Frontend je web aplikacija izrađena u Reactu i predstavlja sloj za prikaz korisničkog sučelja. Njegove glavne odgovornosti uključuju pružanje intuitivnog i responzivnog korisničkog sučelja, obradu korisničkih interakcija poput popunjavanja obrazaca i navigacije kroz aplikaciju, slanje HTTP zahtjeva backendu putem REST API-ja te prikaz podataka dobivenih od backenda, primjerice informacije o sastancima ili korisnicima. Frontend omogućuje korisnicima nesmetanu interakciju s aplikacijom, fokusirajući se na korisničko iskustvo.

Backend je izrađen u Djangou i predstavlja sloj poslovne logike i obrade podataka. Njegove odgovornosti uključuju obradu zahtjeva koje šalje frontend i njihovo usmjeravanje prema odgovarajućim servisima, implementaciju poslovnih pravila poput kreiranja sastanaka, dodjeljivanja uloga i autentikacije korisnika, interakciju s bazom podataka, upravljanje autentikacijom i autorizacijom korisnika te pružanje RESTful API endpointa za frontend. Backend funkcionira kao okosnica aplikacije, koordinirajući interakciju između frontenda i drugih komponenti poput baze podataka i datotečnog sustava.

Frontend šalje HTTP zahtjeve backendu za akcije poput kreiranja sastanka, pregleda sastanaka ili dodavanja korisnika. Backend kontroleri obrađuju zahtjeve, koriste servise za obradu poslovne logike i dohvaćaju ili ažuriraju podatke putem modela. Dobiveni rezultati vraćaju se frontendu u obliku strukturiranog JSON odgovora, što omogućuje dinamičko ažuriranje sučelja. Frontend potom prikazuje podatke na intuitivan način, reflektirajući odgovore backenda u stvarnom vremenu.

MVC arhitektura

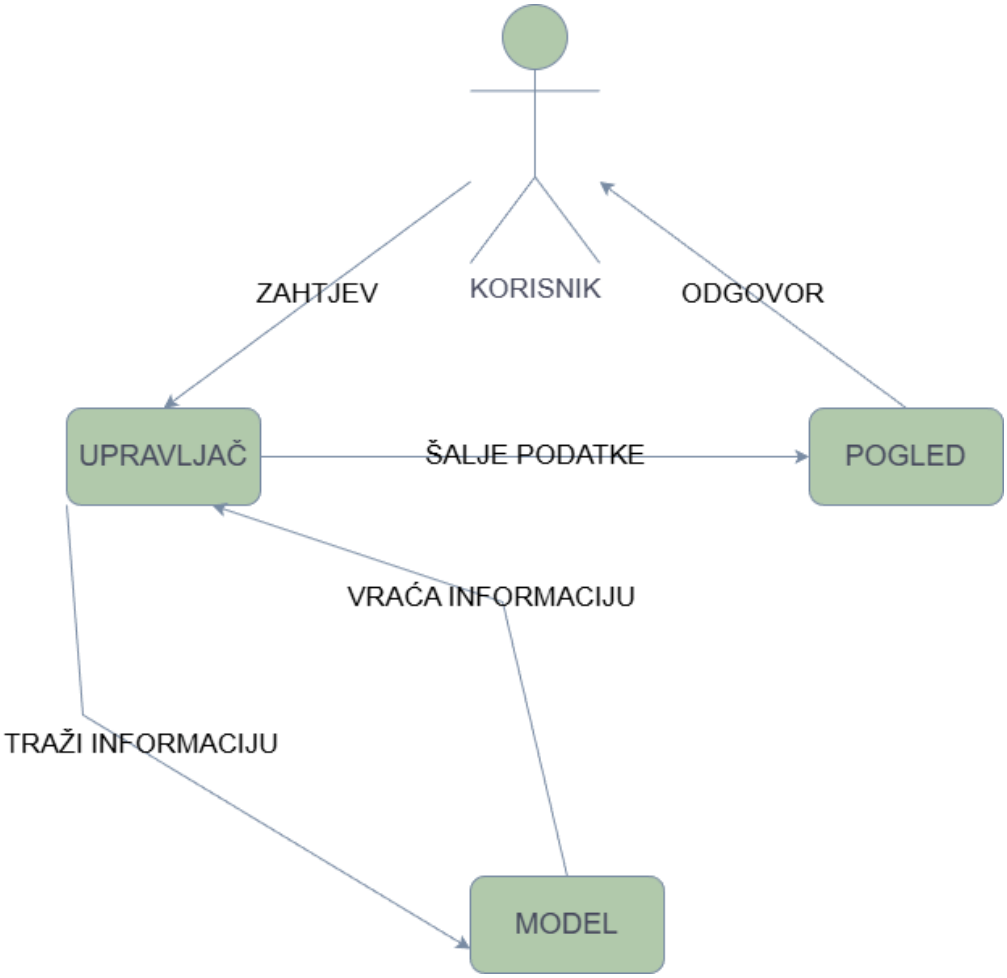
Backend prati MVC (Model-View-Controller) arhitekturu koja razdvaja logiku aplikacije u odvojene slojeve.
Model predstavlja podatke i poslovnu logiku. Uključuje entitete koji se izravno mapiraju na tablice u bazi podataka, sučelja za interakciju s bazom podataka te servise koji implementiraju poslovnu logiku, primjerice dodjeljivanje uloga ili kreiranje sastanaka.
View sloj je primarno implementiran u frontendu. Backend šalje podatke frontend-u u obliku JSON

odgovora putem REST API-ja, a frontend koristi te podatke za dinamičko prikazivanje sučelja. Controller sloj obrađuje HTTP zahtjeve i koordinira rad Modela i View-a. Njegove funkcije uključuju primanje API poziva s frontenda, validaciju podataka iz zahtjeva, pozivanje odgovarajućih servisnih metoda za obradu podataka te slanje odgovora natrag frontend-u. Ova struktura osigurava jasno razdvajanje odgovornosti između slojeva.

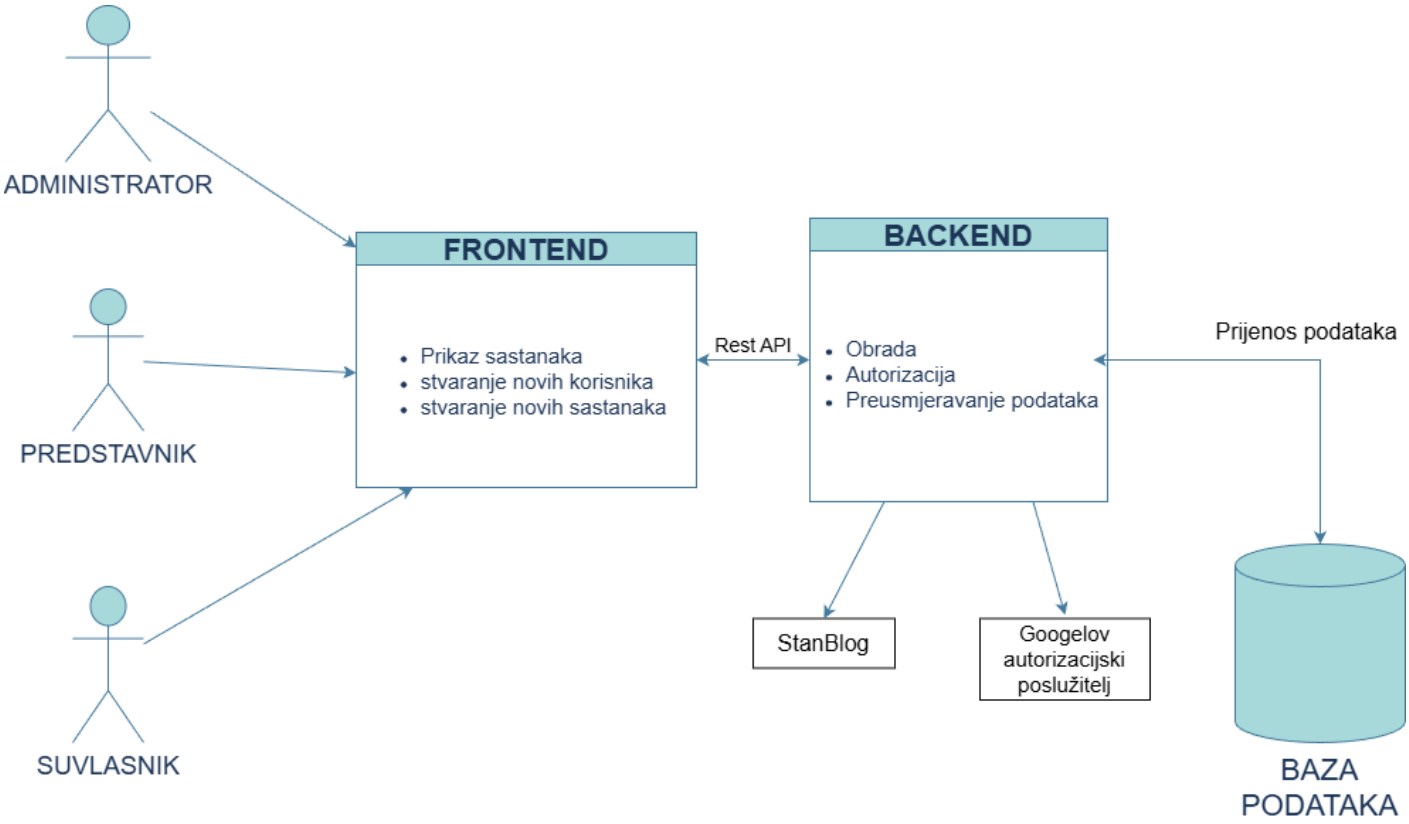
Prednosti MVC arhitekture

MVC arhitektura omogućuje jednostavnu organizaciju i održavanje velikih web aplikacija zahvaljujući jasnom razdvajanju koda na tri sloja – Model, View i Controller. Takva struktura olakšava podjelu posla među većim timovima, omogućuje brzo pronalaženje dijelova koda i jednostavno dodavanje novih funkcionalnosti.
Metodologija MVC-a povećava fleksibilnost i skalabilnost sustava, jer promjene u jednom dijelu aplikacije (npr. u pogledu) ne utječu na ostale dijelove. Razvijanje aplikacija je brže i učinkovitije, budući da više razvojnih inženjera može istodobno raditi na različitim slojevima sustava.
Također, MVC pristup olakšava planiranje i održavanje jer daje jasan okvir za strukturiranje ideja i sprječava dupliciranje koda. Omogućuje i stvaranje više različitih prikaza istih podataka, čime se dodatno povećava fleksibilnost korisničkog sučelja.

MVC dijagram



Dijagram visoke razine



Referenca

Ovaj pristup je u skladu s načelima MVC arhitekture kako ih opisuje članak Benefit of Using MVC (GeeksforGeeks, 2023), koji naglašava prednosti poput lakšeg održavanja, bržeg razvoja te veće fleksibilnosti i skalabilnosti aplikacija.

Baza podataka

Sustav se temelji na korištenju relacijske baze podataka implementirane u PostgreSQL-u, pri čemu su entiteti modelirani kao tablice s jedinstvenim nazivima i pripadajućim atributima.
Odluka o korištenju relacijskog modela proizlazi iz potrebe za učinkovitim upravljanjem podacima koji uključuju korisnike, sastanke, obavijesti, točke dnevnog reda, rasprave i zaključke.
Relacijska struktura omogućuje jednostavno modeliranje stvarnih procesa u aplikaciji StanPlan, čime se postiže bolja organizacija, praćenje i izvještavanje o aktivnostima predstavnika stanara i suvlasnika.
Baza podataka osigurava sigurnost, integritet i konzistentnost podataka, kao i učinkovito dohvaćanje, pohranu, izmjenu i brisanje informacija potrebnih za rad sustava.

Entiteti uključeni u bazu podataka:

- Uloga
- Korisnik
- Status_sastanka

- Sastanak
- Obavijesti
- Sudjeluje
- TockeDnevReda
- Rasprava
- Zakljucak

Opis tablica

Uloga

Atribut	Tip podataka	Opis varijable
id_uloge	Primarni ključ	Jedinstveni identifikator uloge.
naziv_uloge	VARCHAR(50)	Naziv uloge (npr. Administrator, Predstavnik, Suvlasnik).

Korisnik

Atribut	Tip podataka	Opis varijable
id_korisnik	Primarni ključ	Jedinstveni identifikator korisnika.
korisnicko_ime	VARCHAR(50)	Jedinstveno korisničko ime za prijavu.
email	VARCHAR(100)	Adresa e-pošte korisnika.
aktivan	BOOLEAN	Oznaka je li korisnički račun aktivan.
registriran_od	TIMESTAMP	Datum i vrijeme registracije korisnika.
password_hash	VARCHAR(255)	Enkriptirana lozinka korisnika.
id_uloge	Vanjski ključ	Povezuje korisnika s tablicom Uloga.

Status_sastanka

Atribut	Tip podataka	Opis varijable
id_status	Primarni ključ	Jedinstveni identifikator statusa.
naziv_status	VARCHAR(30)	Naziv statusa (npr. Planiran, Objavljen, Obavljen, Arhiviran).

Sastanak

Atribut	Tip podatka	Opis varijable
id_sastanak	Primarni ključ	Jedinstveni identifikator sastanka.
naslov	VARCHAR(200)	Naslov sastanka.
napravljen_od	TIMESTAMP	Vrijeme kreiranja sastanka.
lokacija	VARCHAR(200)	Lokacija održavanja sastanka.
datum_vrijeme	TIMESTAMP	Datum i vrijeme sastanka.
sazetak	TEXT	Kratki opis ili namjera sastanka.
id_korisnik	Vanjski ključ	Korisnik koji je kreirao sastanak.
id_status	Vanjski ključ	Povezuje sastanak s tablicom Status_sastanka.

Obavijesti

Atribut	Tip podatka	Opis varijable
id_obavijesti	Primarni ključ	Jedinstveni identifikator obavijesti.
poruka	TEXT	Sadržaj obavijesti.
poslano_u	TIMESTAMP	Vrijeme slanja obavijesti.
status_mail	VARCHAR(30)	Status slanja e-pošte.
id_sastanak	Vanjski ključ	Povezuje obavijest s određenim sastankom.

Sudjeluje

Atribut	Tip podatka	Opis varijable
id_sudjelovanja	Primarni ključ	Jedinstveni identifikator sudjelovanja.
vrijeme_potvrde	TIMESTAMP	Vrijeme potvrde sudjelovanja.
potvrda	BOOLEAN	Oznaka je li korisnik potvrdio dolazak.
id_korisnik	Vanjski ključ	Povezuje korisnika sa sastankom.
id_sastanak	Vanjski ključ	Povezuje sastanak s korisnikom.
(id_korisnik, id_sastanak)	Jedinstveni par	Osigurava da isti korisnik ne potvrdi isti sastanak više puta.

TockeDnevReda

Atribut	Tip podatka	Opis varijable
id_tocke	Primarni ključ	Jedinstveni identifikator točke dnevnog reda.
broj_tocke	INT	Redni broj točke na sastanku.
naziv	VARCHAR(200)	Naziv točke dnevnog reda.
opis	TEXT	Opis teme točke.
pravni_ucinak	BOOLEAN	Oznaka ima li točka pravni učinak.
id_sastanak	Vanjski ključ	Povezuje točku s određenim sastankom.

Rasprava

Atribut	Tip podatka	Opis varijable
id_rasprave	Primarni ključ	Jedinstveni identifikator rasprave.
url_StanBlog	VARCHAR(255)	Poveznica na odgovarajuću raspravu u aplikaciji StanBlog.
id_tocke	Vanjski ključ	Povezuje raspravu s određenom točkom dnevnog reda.

Zakljucak

Atribut	Tip podatka	Opis varijable
id_zakljucak	Primarni ključ	Jedinstveni identifikator zaključka.
tekst	TEXT	Tekst zaključka.
status	VARCHAR(30)	Status zaključka (npr. Izglasan, Odbijen).
unesen_u	TIMESTAMP	Vrijeme unosa zaključka.
id_korisnik	Vanjski ključ	Korisnik koji je unio zaključak.
id_tocke	Vanjski ključ	Povezuje zaključak s točkom dnevnog reda.

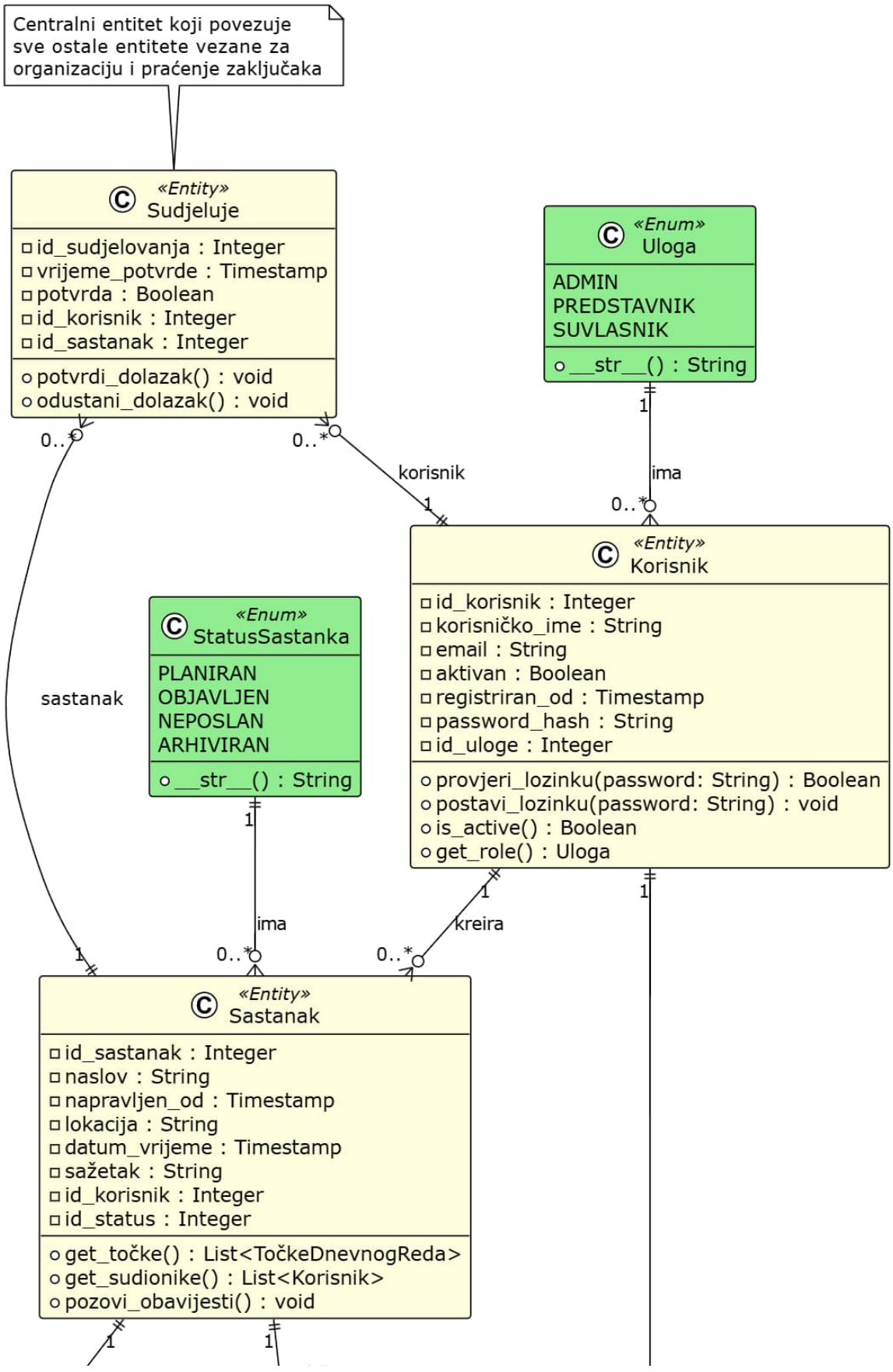
Dijagram baze podataka

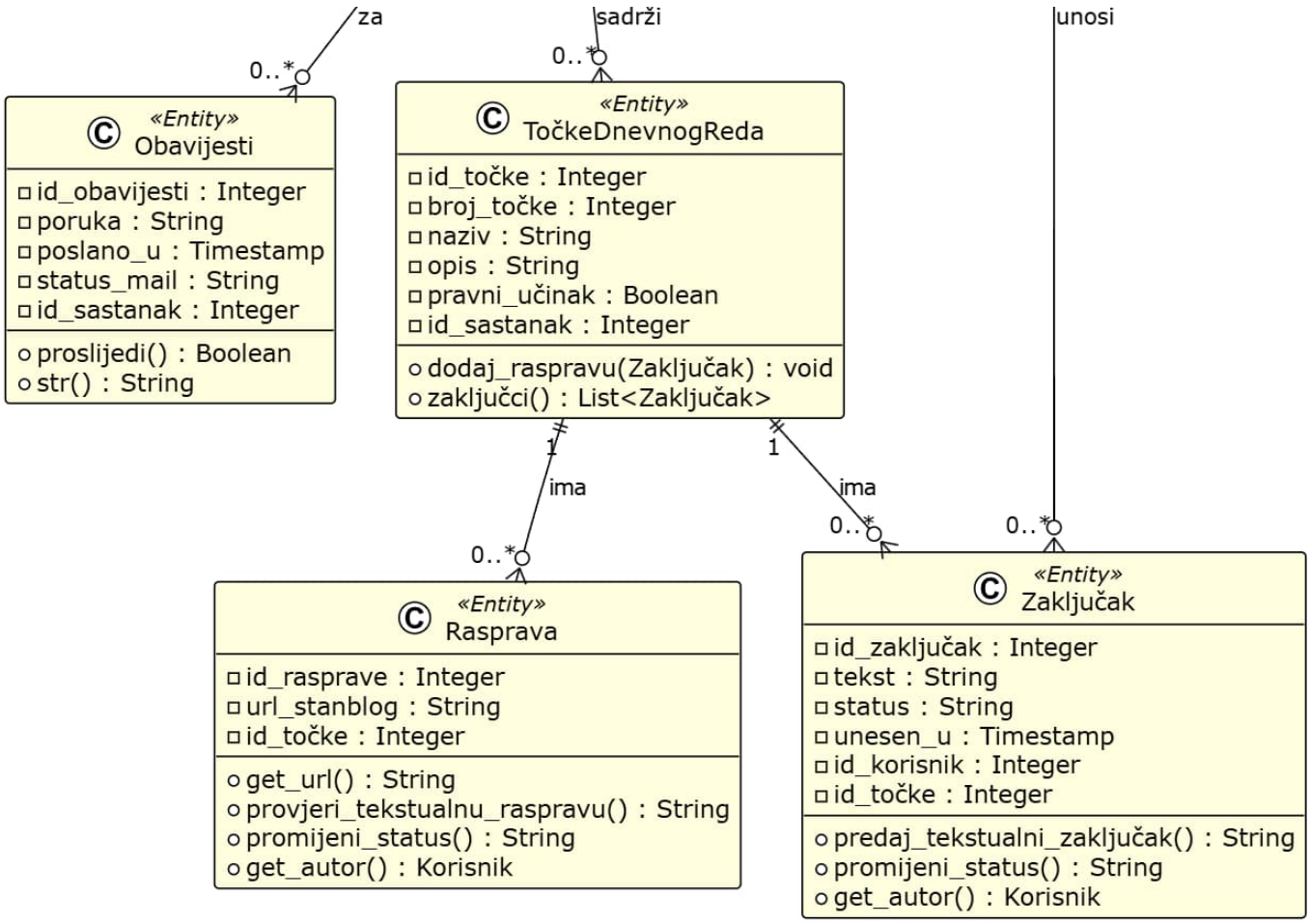
Relacijska schema

ER dijagram

Dijagram razreda

Dijagram razreda - model (baza podataka)





Opis

Ovaj UML dijagram razreda prikazuje **generičku funkcionalnost sustava** za upravljanje sastancima stanara. Dijagram modelira potpunu strukturu podataka s jasno definiranim vezama između entiteta, kardinalnostima i enumeracijskim tipovima.

Ključni entiteti i funkcionalnosti

Korisničko upravljanje:

- `Korisnik` - Centralni entitet s atributima za autentifikaciju i profilne podatke
- `Uloga` - Enumeracija tipova korisnika (ADMIN, PREDSTAVNIK, SUVLASNIK)

Upravljanje sastancima:

- `Sastanak` - Glavni entitet s vremenom, lokacijom i statusom
- `StatusSastanka` - Enumeracija stanja (PLANIRAN, OBJAVLJEN, NEPOSLAN, ARHIVIRAN)
- `Sudjeluje` - Asocijacijska klasa koja povezuje korisnike sa sastancima

Struktura dnevnog reda:

- `TočkeDnevnogReda` - Organizacija sadržaja sastanka
- `Rasprava` - Povezivanje s vanjskim resursima za diskusiju

- `Zaključak` - Formalni ishodi odlučivanja

Komunikacija:

- `Obavijesti` - Upravljanje porukama i notifikacijama

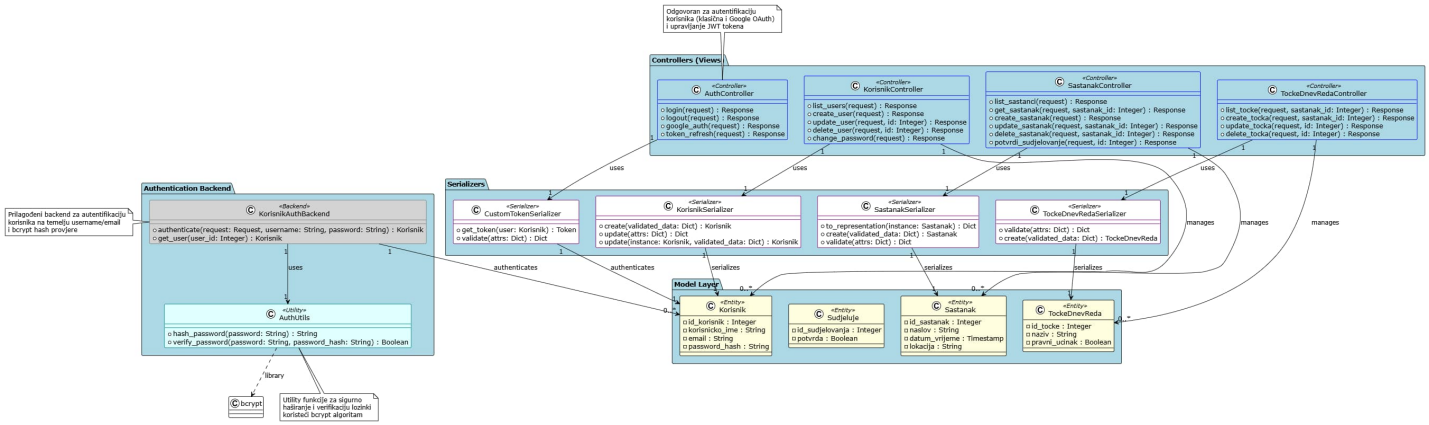
Modelirane veze i ograničenja

- **1:N veze** između korisnika i kreiranih sastanaka
- **M:N veze** između korisnika i sastanaka preko entiteta `Sudjeluje`
- **Hijerarhijske veze** sastanak → točke dnevnog reda → zaključci
- **Enumeracijske ograničenja** za tipove korisnika i statuse sastanaka
- **Audit trail** preko timestamp atributa i korisničkih referenci

Razina apstrakcije

Dijagram predstavlja **podatkovni model** (data layer) koji definira strukturu baze podataka i osnovne poslovne entitete potrebne za funkcionalnost sustava upravljanja sastancima.

Dijagram razreda - backend-API



Opis

Ovaj UML dijagram razreda prikazuje **arhitekturu backend sustava** izgrađenu na Django REST Framework-u. Dijagram modelira slojeve aplikacije grupirane prema srodnim funkcionalnostima i razinama apstrakcije.

Modelirane komponente po slojevima

Serializers (Serijalizacijski sloj):

- `KorisnikSerializer` - Transformacija i validacija korisničkih podataka

- `SastanakSerializer` - Obrada podataka sastanaka s nested objektima
- `TočkeDnevnogRedaSerializer` - Serijalizacija strukturiranog dnevnog reda
- `CustomTokenSerializer` - JWT token generiranje i validacija

Controllers (Kontrolerski sloj):

- `AuthController` - Autentifikacija (lokalna + Google OAuth)
- `KorisnikController` - CRUD operacije za korisnike
- `SastanakController` - Upravljanje životnim ciklusom sastanaka
- `TočkeDnevnogRedaController` - Organizacija sadržaja sastanaka

Authentication Backend (Sigurnosni sloj):

- `KorisnikAuthBackend` - Prilagođeni Django authentication backend
- `AuthUtils` - Utility funkcije za sigurno rukovanje lozinkama (bcrypt)

Model Layer (Podatkovni sloj):

- Referentni entiteti iz baze podataka

Modelirane veze i pristup metodama

- **Javni pristup** (+) za API endpoint metode
- **1:1 veze** između serializera i entiteta
- **1:N veze** između kontrolera i upravljanih resursa
- **Dependency injection** između slojeva
- **Utility pattern** za cross-cutting concerns

Razina apstrakcije

Dijagram predstavlja **aplikacijski sloj** (application layer) koji enkapsulira poslovnu logiku, API definicije i sigurnosne mehanizme potrebne za funkcionalnost web službi sustava.

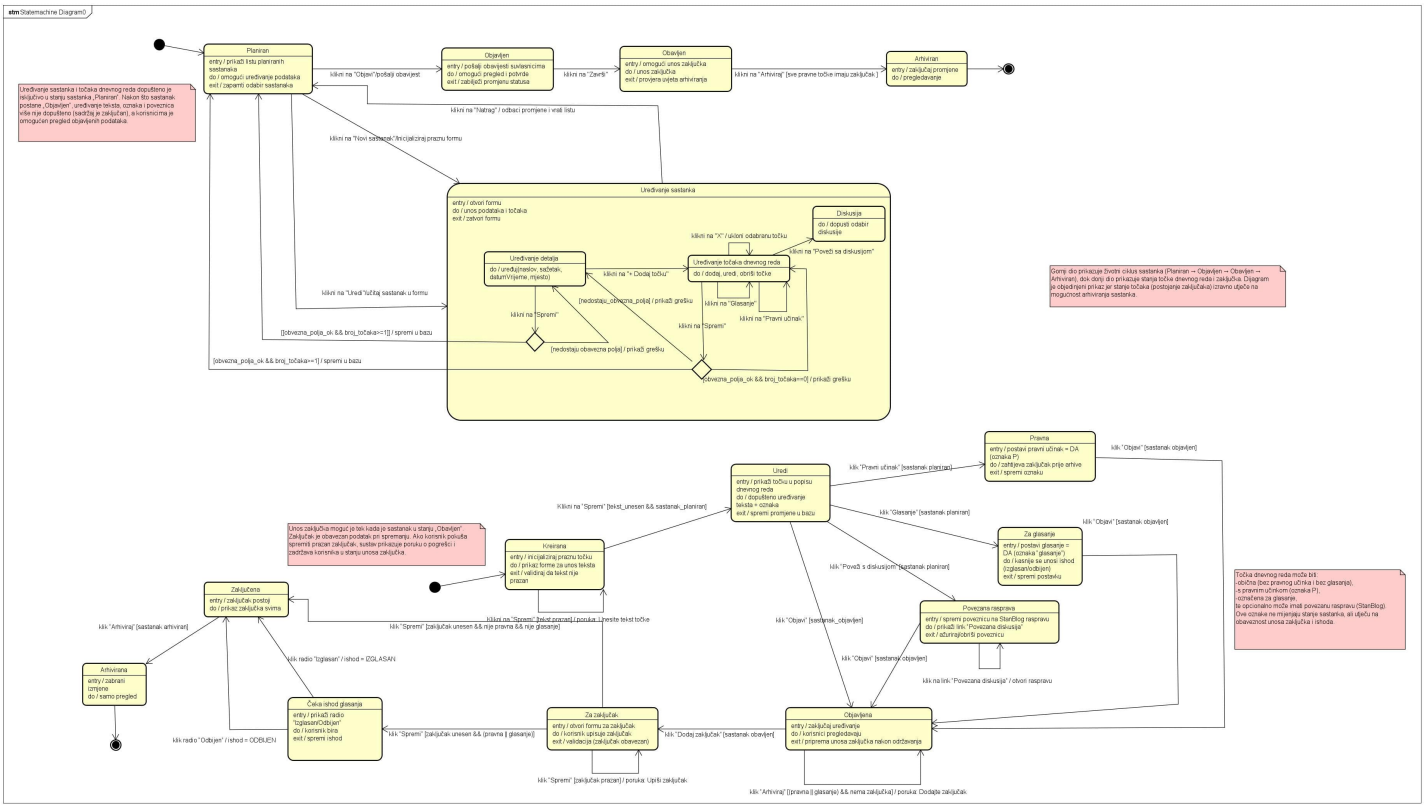
Dinamičko ponašanje aplikacije

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

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

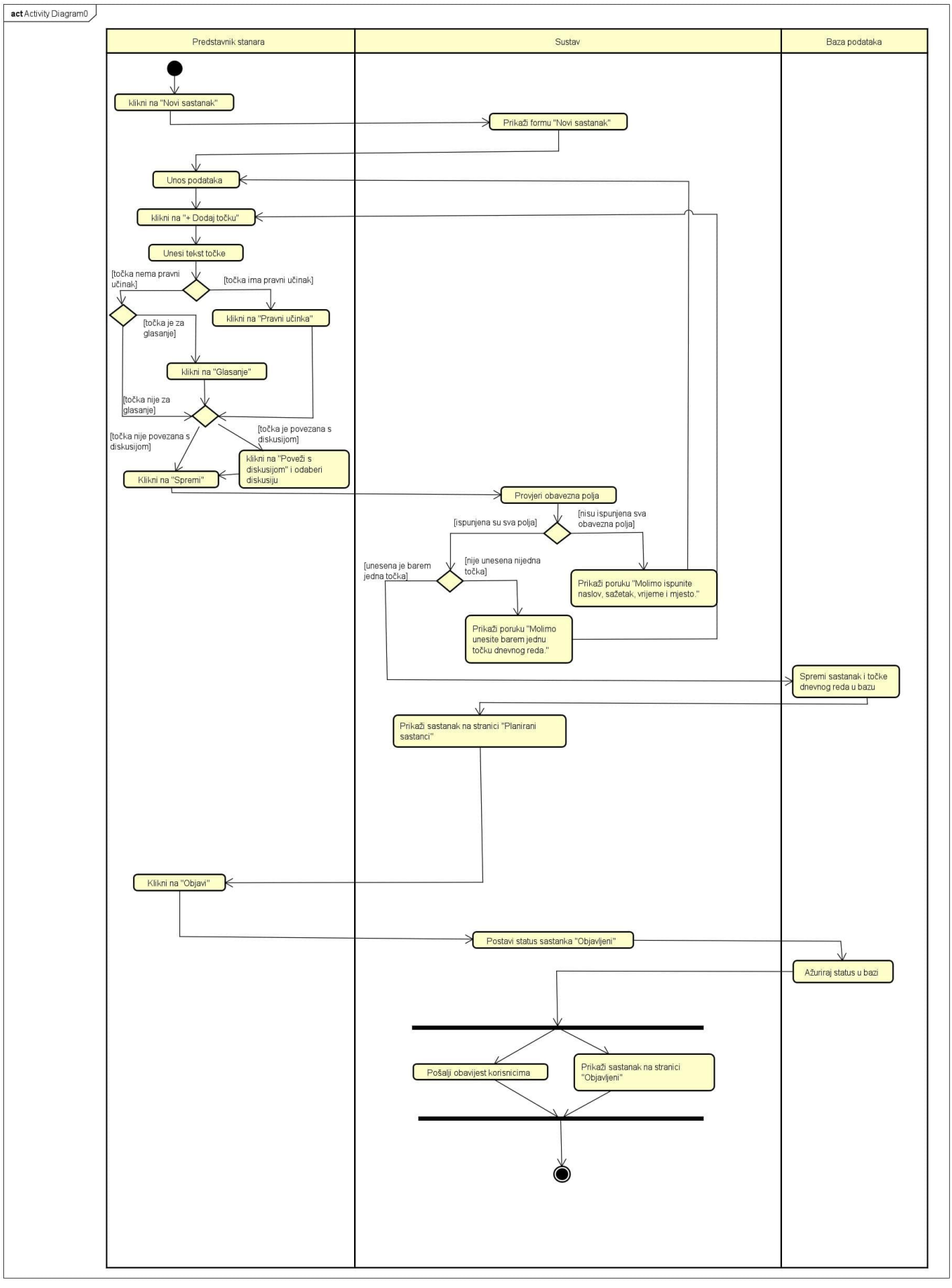
UML dijagrami stanja

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

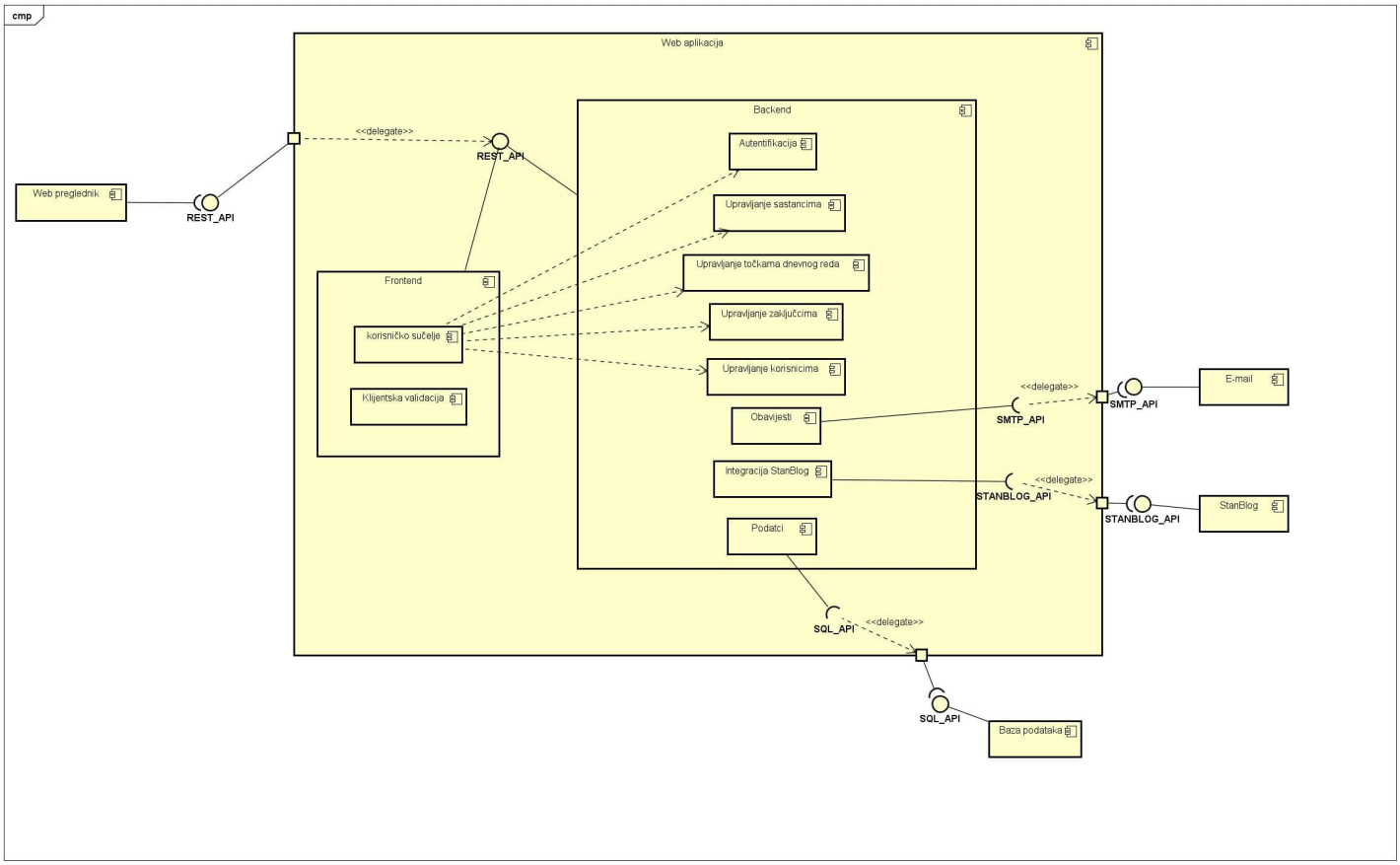


UML dijagrami aktivnosti

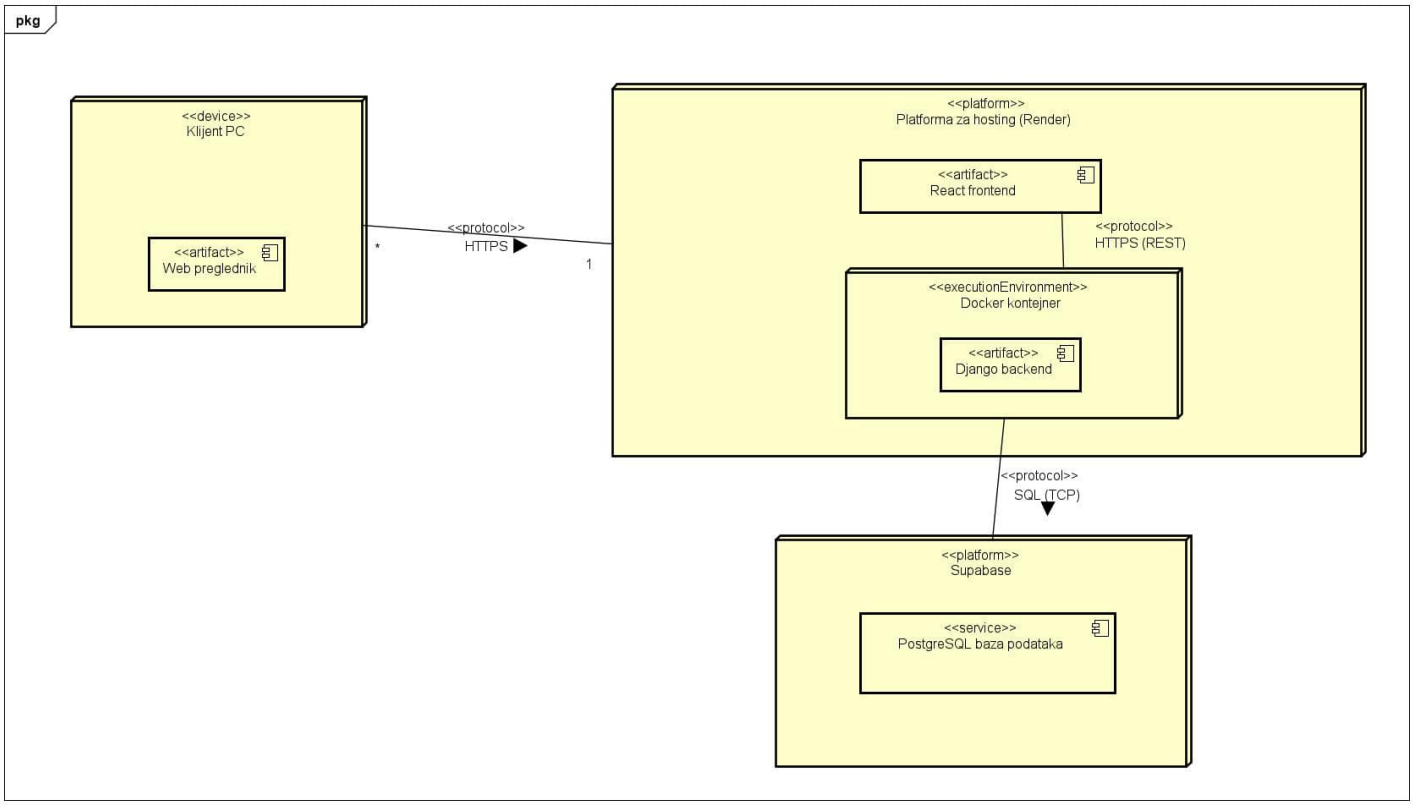
Dijagram aktivnosti prikazuje tijek izvršavanja određenog procesa. Osim za razumijevanje toka podataka unutar aplikacije, koristi se za analizu poslovnih procesa.


act Activity Diagram0
Predstavnik stanara
Sustav
Baza podataka
klikni na "Novi sastanak"
Prikaži formu "Novi sastanak"
Unos podataka
klikni na "+ Dodaj točku"
Unesi tekst točke
[točka nema pravni učinak]
[točka ima pravni učinak]
klikni na "Pravni učinka"
[točka je za glasanje]
klikni na "Glasanje"
[točka nije za glasanje]
[točka nije povezana s diskusijom]
[točka je povezana s diskusijom]
klikni na "Poveži s diskusijom" i odaberi diskusiju
Klikni na "Spremi"
Provjeri obavezna polja
[ispunjena su sva polja]
[nisu ispunjena sva obavezna polja]
[unesena je barem jedna točka]
[nije unesena nijedna točka]
Prikaži poruku "Molimo ispunite naslov, sažetak, vrijeme i mjesto."
Prikaži poruku "Molimo unesite barem jednu točku dnevnog reda."
Spremi sastanak i točke dnevnog reda u bazu
Prikaži sastanak na stranici "Planirani sastanci"
Klikni na "Objavi"
Postavi status sastanka "Objavljeni"
Ažuriraj status u bazi
Pošalji obavijest korisnicima
Prikaži sastanak na stranici "Objavljeni"

Dijagram komponenata



Dijagram razmještaja



ISPITIVANJE PROGRAMSKOG RJEŠENJA

Ispitivanje sustava i njegovih komponenti proveli smo koristeći kombinaciju unit testova u Django-u i Selenium testova za korisničko sučelje.

- Ispitivanje komponenti:
  Komponente sustava, uključujući modele, validaciju podataka i poslovnu logiku, testirane su koristeći kombinaciju Django i Selenium testova. Ovi testovi omogućili su provjeru pravilnog funkcioniranja funkcionalnosti kao što su kreiranje sastanaka, validacija korisnika, upravljanje statusima i rukovanje iznimkama.
- Ispitivanje sustava:
  Cjelokupni sustav, uključujući integraciju komponenti i korisničko sučelje, testiran je pomoću Selenium testova. Korištenjem preglednika automatizirano smo provjeravali funkcionalnosti sustava.
- Lokacija testova:
  Svi Django testovi nalaze se u direktoriju backend/api/tests/test_sastanak.py, dok se Selenium testovi nalaze u /ispitivanje/"ispitivanje sustava" ili /ispitivanje/"ispitivanje komponenti".

Za pokretanje Django testova potrebno se pozicionirati u /backend te izvršiti sljedeću naredbu `python manage.py test api.tests.test_sastanak` , a za Selenium testove potrebno se pozicionirati u /ispitivanje, onda ovisno o željenom ispitu, u jedan od dva foldera /"ispitivanje sustava" ili /"ispitivanje komponenti" te izvesti sljddeću naredbu `python testOdjavaKorisnika.py` (npr. \ispitivanje\Ispitivanje komponenti>python testOdjavaKorisnika.py).

> Napomena: ako želite pokrenuti Selenium testove, prvo se prijavite na našu stranicu kako bi je pokrenuli, jer s cold startom Selenium test neće radit ako shvati da stranica nije aktivna.

Ispitivanje komponenti

Django testovi:

1. Test: Predstavnik može kreirati planirani sastanak - Redovan slučaj

- Funkcionalnost: Kreiranje sastanka u sustavu. Predstavnik može kreirati planirani sastanak.
- Ispitni slučaj:
  - Ulazni podaci: Korisnik: iva, uloga: Predstavnik; status sastanka: Planiran; naslov: Planirani sastanak; lokacija: Online.

    - Očekivani rezultat: Sastanak se kreira bez pogreške; id_korisnik.get_role() = Predstavnik; id_status.naziv_status = Planiran.
    - Dobiveni rezultat: Test prolazi (OK).
- Postupak ispitivanja: Kreiran je objekt Korisnik s ulogom Predstavnik i objekt Sastanak s planiranim statusom. Provjerava se atribut id_korisnik.get_role() i status sastanka.

2. Test: Suvlasnik ne može kreirati sastanak - izazivanje pogreške

- Funkcionalnost: Sprečavanje neovlaštenog kreiranja planiranog sastanka.
- Ispitni slučaj:
    - Ulazni podaci: Korisnik: Iani, uloga: Suvlasnik; status sastanka: Planiran.
    - Očekivani rezultat: Podizanje PermissionError s porukom "Suvlasnik ne može kreirati sastanak".
    - Dobiveni rezultat: Test prolazi; exception je podignut i ispisuje se poruka u konzoli.
- Postupak ispitivanja: Provjerava se role korisnika. Ako korisnik nije Predstavnik i status je Planiran, ručno se podiže PermissionError i hvata u try/except bloku. Poruka exception-a se ispisuje, a test potvrđuje da je exception bačen.

3. Test: Sastanak bez statusa - izazivanje pogreške

- Funkcionalnost: Provjera što se događa kada se kreira sastanak bez definiranog statusa.
- Ispitni slučaj:
    - Ulazni podaci: Korisnik: vita, uloga: Korisnik; status sastanka: None; naslov: Sastanak bez statusa.
    - Očekivani rezultat: id_status je None.
    - Dobiveni rezultat: Test prolazi (OK).
- Postupak ispitivanja: Kreira se sastanak bez statusa. Provjerava se atribut id_status da bude None. Printom se prikazuje kreirani sastanak i status.

4. Test: Sastanak bez korisnika - izazivanje pogreške

- Funkcionalnost: Provjera ponašanja kada se sastanak kreira bez korisnika.
- Ispitni slučaj:
    - Ulazni podaci: Korisnik: None; status sastanka: Planiran; naslov: Sastanak bez korisnika.
    - Očekivani rezultat: id_korisnik je None.
    - Dobiveni rezultat: Test prolazi (OK).
- Postupak ispitivanja: Kreira se sastanak s id_korisnik=None. Provjerava se atribut id_korisnik da bude None. Printom se prikazuje kreirani sastanak i korisnik.

5. Test: Duplikat username - rubni slučaj

- Funkcionalnost: Validacija jedinstvenosti korisničkog imena.
- Ispitni slučaj:
    - Ulazni podaci: Dva korisnika: dupli i dupli (isti username).

    - Očekivani rezultat: Podizanje ValueError s porukom "Username mora biti jedinstven".
    - Dobiveni rezultat: Test prolazi; exception je podignut i ispisuje se poruka.
- Postupak ispitivanja: Pokušava se kreirati dva korisnika s istim usernameom. Ako se username poklapa, podiže se ValueError. Printom se prikazuje poruka exception-a, a test provjerava da je exception bačen.

6. Test: Sastanak u prošlosti - rubni slučaj

- Funkcionalnost: Validacija datuma sastanka da ne bude u prošlosti.
- Ispitni slučaj:
    - Ulazni podaci: Datum sastanka: 1.1.2020; korisnik: eva.
    - Očekivani rezultat: Podizanje ValueError s porukom "Datum sastanka ne smije biti u prošlosti".
    - Dobiveni rezultat: Test prolazi; exception je podignut i ispisuje se poruka.
- Postupak ispitivanja: Kreira se sastanak s datumom u prošlosti. Ako je datum manji od datetime.now(), podiže se exception i printa poruka. Test provjerava da je exception podignut.

7. Test: Poziv nepostojeće metode - nepostojeća funkcionalnost

- Funkcionalnost: Provjera reakcije sustava kada se pozove nepostojeća metoda ili atribut.
- Ispitni slučaj:
    - Ulazni podaci: Korisnik: ivan; pokušaj pristupa nepostojećem atributu/metodi: nadimak ili nepostojeća_metoda().
    - Očekivani rezultat: Podizanje AttributeError s porukom 'Korisnik' object has no attribute 'nadimak'.
    - Dobiveni rezultat: Test prolazi; exception je podignut i ispisuje se poruka u konzoli.
- Postupak ispitivanja: Pokušava se pristupiti atributu/metodi koja ne postoji. Hvata se AttributeError, printa poruka exception-a, a test provjerava da je exception bačen.

Selenium testovi:

8. Test nevažećeg korisnika [testNevazeciKorisnik.py]

Ulazni podaci:

- Korisničko ime: nepostojeci_korisnik_123
- Lozinka: kriva_lozinka

Koraci:

- Unos podataka za prijavu.
- Klik na prijava.
- Čekamo prikaz pogreške

Očekivani izlaz:

PASS: Nepostojeći korisnik: greška ispravno prikazana

Dobiveni izlaz:

```
PASS: Nepostojeći korisnik: greška ispravno prikazana
```

9. Test odjave korisnika [testOdjavaKorisnika.py]

Ulazni podaci:

- Korisničko ime: stanar
- Lozinka: stanar123

Koraci:

- Unos podataka za prijavu.
- Klik na prijava.
- Odlazak na profil
- Odjava klikom na „Odjavi se"

Očekivani izlaz:

PASS: Prijava uspješna

PASS: Odjava uspješna - login ponovno vidljiv

Dobiveni izlaz:

```
PASS: Prijava uspješna
PASS: Odjava uspješna - login ponovno vidljiv
```

10. Test pogrešne lozinke [testPogresnaLozinka.py]

Ulazni podaci:

- Korisničko ime: stanar
- Lozinka: kriva123

Koraci:

- Unos podataka za prijavu.
- Klik na prijava.
- Čekamo prikaz pogreške

Očekivani izlaz:

PASS: Pogrešna lozinka: greška ispravno prikazana

Dobiveni izlaz:

```
PASS: Pogrešna lozinka: greška ispravno prikazana
```

11. Test pokušaj brisanja sastanka [testPokusajBrisanjaSastanaka.py]

Ulazni podaci:

- Korisničko ime: boki
- Lozinka: luk123

Koraci:

- Unos podataka za prijavu.
- Klik na prijava.
- Prelazak na planirane sastanke
- Provjera postoji li gumb "obriši" negdje na stranici planiranih sastanaka
- Ponavljamo postupak za sve sastanke

Očekivani izlaz:

PASS: Nepostojeća funkcionalnost: gumb 'Obriši' ne postoji na stranici planiranih sastanaka

PASS: Nepostojeća funkcionalnost: gumb 'Obriši' ne postoji na stranici objavljenih sastanaka

PASS: Nepostojeća funkcionalnost: gumb 'Obriši' ne postoji stranici obavljenih sastanaka

PASS: Nepostojeća funkcionalnost: gumb 'Obriši' ne postoji na stranici arhiviranih sastanaka

Dobiveni izlaz:

```
PASS: Nepostojeća funkcionalnost: gumb 'Obriši' ne postoji na stranici planiranih sastanaka
PASS: Nepostojeća funkcionalnost: gumb 'Obriši' ne postoji na stranici objavljenih sastanaka
PASS: Nepostojeća funkcionalnost: gumb 'Obriši' ne postoji stranici obavljenih sastanaka
PASS: Nepostojeća funkcionalnost: gumb 'Obriši' ne postoji na stranici arhiviranih sastanaka
```

12. Test praznih polja na prijavi [testPraznaPoljaNaPrijavi.py]

Ulazni podaci:

- Korisničko ime: stanar
- Lozinka: {prazno polje}

Koraci:

- Unos korisničkog imena
- Lozinku namjerno ostavljamo praznom
- Čekamo prikaz pogreške

Očekivani izlaz:

PASS: Prazna polja: korisnik nije ulogiran

PASS: Prazna polja: nije preusmjeren na /suvlasnici

Dobiveni izlaz:

```
PASS: Prazna polja: korisnik nije ulogiran
PASS: Prazna polja: nije preusmjeren na /suvlasnici
```

13. Test prijava [testPrijeva.py]

Ulazni podaci:

- Korisničko ime: stanar
- Lozinka: stanar123

Koraci:

- Unos podataka za prijavu.
- Čekamo preusmjerenje na stranicu.

Očekivani izlaz:

PASS: Prijava uspješna i preusmjerenje na /suvlasnici

Dobiveni izlaz:

```
PASS: Prijava uspješna i preusmjerenje na /suvlasnici
```

Ispitivanje sustava

Test promjene lozinke [testLozinkaStanar.py] - Redovan slučaj

Ulazni podaci:

- Korisničko ime: stanar
- Lozinka: stanar123

Koraci:

- Unos podataka za prijavu.
- Klik na prijava.
- Prelazak na stranicu proflia klikom na ikonu profila.
- Promjena lozinke:
    - Stara lozinka: stanar123
    - Nova lozinka: stanar
    - Potvrda nove lozinke: stanar

    - Klik na promjeni lozinku
- Odjava klikom na „Odjavi se"
- Pokušaj prijave starom lozinkom (Korisničko ime: stanar, Lozinka: stanar123) - izazivanje pogreške
- Brisanje stare te upis nove lozinke
- Prelazak na stranicu proflia klikom na ikonu profila.
- Promjena lozinke:
    - Stara lozinka: stanar
    - Nova lozinka: stanar123
    - Potvrda nove lozinke: stanar123
    - Klik na promjeni lozinku
- Odjava klikom na „Odjavi se"

Očekivani izlaz:

> PASS: Prijava uspješna i preusmjerenje na /suvlasnici

> PASS: Lozinka promijenjena

> PASS: Korisnik odjavljen

> PASS: Pogrešna lozinka ispravno detektirana

> PASS: Lozinka vraćena na početnu

Dobiveni izlaz:

```
PASS: Prijava uspješna i preusmjerenje na /suvlasnici
PASS: Lozinka promijenjena
PASS: Korisnik odjavljen
PASS: Pogrešna lozinka ispravno detektirana
PASS: Lozinka vraćena na početnu
```

Otkrivene greške: 0

Test navigacije kroz sustav [testNavigacijaSustava.py] - Redovan slučaj

Ulazni podaci:

- Korisničko ime: stanar
- Lozinka: stanar123

Koraci:

- Unos podataka za prijavu.
- Klik na prijava.
- Prelazak na stranicu proflia klikom na ikonu profila.
- Povratak na sastanke:

    - Hover preko filter ikone te odabir objavljenih sastanaka
    - Hover preko filter ikone te odabir arhiviranih sastanaka
- Prelazak na stranicu proflia klikom na ikonu profila.
- Odjava klikom na „Odjavi se"

Očekivani izlaz:

> PASS: Login uspješan
>
> PASS: Otvoren profil
>
> PASS: Hover menu otvoren
>
> PASS: Povratak na /suvlasnici/objavljeni uspješan
>
> PASS: Hover menu otvoren
>
> PASS: Povratak na /suvlasnici/arhivirani uspješan
>
> PASS: Uspješna odjava

Dobiveni izlaz:

```
PASS: Login uspješan
PASS: Otvoren profil
PASS: Hover menu otvoren
PASS: Povratak na /suvlasnici/objavljeni uspješan
PASS: Hover menu otvoren
PASS: Povratak na /suvlasnici/arhivirani uspješan
PASS: Uspješna odjava
```

Otkrivene greške: 0

Test odjave i refresha nakon odjave [testOdjavaRefresh.py] - Rubni slučaj

Ulazni podaci:

- Korisničko ime: stanar
- Lozinka: stanar123

Koraci:

- Unos podataka za prijavu.
- Klik na prijava.
- Prelazak na stranicu proflia klikom na ikonu profila.
- Odjava klikom na „Odjavi se"
- Refresh stranice - provjera je li korisnik odjavljen

Očekivani izlaz:

> PASS: Odjava uspješna - login forma vidljiva

> PASS: Nakon refresh-a korisnik ostaje odjavljen

Dobiveni izlaz:

```
PASS: Odjava uspješna - login forma vidljiva
PASS: Nakon refresh-a korisnik ostaje odjavljen
```

Otkrivene greške: 0

Test refresha [testRefresh.py] - Redovan slučaj

Ulazni podaci:

- Korisničko ime: stanar
- Lozinka: stanar123

Koraci:

- Unos podataka za prijavu.
- Klik na prijava.
- Pokušaj refresha.

Očekivani izlaz:

> PASS: Prijava uspješna i preusmjerenje na /suvlasnici

> PASS: Nakon refresh-a korisnik ostaje na /suvlasnici

Dobiveni izlaz:

```
PASS: Prijava uspješna i preusmjerenje na /suvlasnici
PASS: Nakon refresh-a korisnik ostaje na /suvlasnici
```

Otkrivene greške: 0

Test može li suvlasnik vidjeti planirane sastanke [testSuvlasnikPlanirani.py] - Neispravna funkcijonalnost

Ulazni podaci:

- Korisničko ime: stanar
- Lozinka: stanar123

Koraci:

- Unos podataka za prijavu.
- Klik na prijava.
- Hover preko filter ikone te „odabir" planiranih sastanaka
- Hover preko filter ikone te „odabir" obavljenih sastanaka
- Hover preko filter ikone te „odabir" Novog sastanaka
- Hover preko filter ikone te „odabir" Dodavanja korisnika

Očekivani izlaz:

> PASS: Prijava uspješna i učitana stranica

> PASS: Hover menu otvoren

> PASS: Suvlasnik ne vidi 'Planirani' u meniju

> PASS: Suvlasnik ne vidi 'Obavljeni' u meniju

> PASS: Suvlasnik ne vidi 'Novi sastanak' u meniju

> PASS: Suvlasnik ne vidi 'Dodavanje korisnika' u meniju

Dobiveni izlaz:

```
PASS: Prijava uspješna i učitana stranica
PASS: Hover menu otvoren
PASS: Suvlasnik ne vidi 'Planirani' u meniju
PASS: Suvlasnik ne vidi 'Obavljeni' u meniju
PASS: Suvlasnik ne vidi 'Novi sastanak' u meniju
PASS: Suvlasnik ne vidi 'Dodavanje korisnika' u meniju
```

Otkrivene greške: 0

Korištene tehnologije i alati

1. **Programski jezici**:

- **JavaScript (ES2024)**: Koristi se za frontend razvoj. Novije verzije JavaScripta omogućuju moderan, deklarativan pristup razvoju korisničkog sučelja uz podršku za asinkroni kod, arrow funkcije i modularnu arhitekturu.

- **Python (3.9+)**: Odabran je za backend razvoj zbog svoje čitljivosti, bogatog ekosustava biblioteka i brze izrade prototipova. Verzija Python 3.9+ omogućuje modernu tipizaciju i performanse potrebne za poslužiteljsku logiku.

2. **Radni okviri i biblioteke**:

- **React (19.1.1)**: Popularna biblioteka za izradu interaktivnih korisničkih sučelja temeljenih na komponentama. React omogućuje stvaranje samostalnih, ponovno iskoristivih komponenti s reaktivnim ažuriranjem korisničkog sučelja. Koristi koncept virtualnog DOM-a za optimizaciju performansi pri renderiranju.

- **Vite (7.1.7)**: Moderan alat za izgradnju aplikacija i razvojni poslužitelj koji omogućuje brzo osvježavanje tijekom razvoja (Hot Module Replacement). Vite je odabran jer je brži od tradicionalnih alata poput Webpacka, uz bolje razvojno iskustvo i manju veličinu paketa.

- **Django (5.2.8)**: Visokorazinski web okvir za Python koji potiče brz razvoj i čist, pragmatičan dizajn. Django pruža ORM za rad s bazom podataka, **sustav** autentifikacije, administratorsko sučelje i sigurnosne mehanizme.

3. **Baza podataka**:

- **PostgreSQL**: Relacijska baza podataka odabrana za pohranu podataka aplikacije. PostgreSQL pruža:
    - pouzdana ACID jamstva za integritet podataka
    - složene upite i JOIN operacije za povezane podatke
    - skalabilnost za buduće proširenje aplikacije
    - sigurnosne mehanizme i enkripciju podataka

4. **Razvojni alati**:

- **Visual Studio Code**: Preporučeni uređivač koda za brz i fleksibilan razvoj s bogatim ekosustavom proširenja za JavaScript i Python.
- **PyCharm**: Snažan integrirani razvojni okoliš za Python s ugrađenim alatom za ispravljanje pogrešaka i analizom koda.

5. **Alati za ispitivanje**:

- **Selenium (4.0)**: Alat za automatizirano testiranje korisničkog sučelja web aplikacija. Selenium omogućuje simulaciju ponašanja korisnika u pregledniku, testiranje funkcionalnosti kroz stvarne scenarije korištenja te provjeru kompatibilnosti aplikacije s različitim web preglednicima.
- **Django Test Framework**: Django ugrađeni alat za testiranje komponenti web aplikacije. Omogućuje pisanje i izvođenje unit testova za modele, prikaze (views), forme i poslovnu logiku. Django testovi pomažu provjeriti ispravnost funkcionalnosti pojedinačnih komponenti, validaciju podataka, rukovanje iznimkama te integraciju između različitih dijelova sustava. Testovi se mogu automatizirano pokretati i integrirati u CI/CD procese, čime se osigurava stabilnost i kvalitetu koda.

6. **Alati za razmještaj**:

- **Docker (opcionalno)**: Iako nije korišten u trenutačnoj konfiguraciji, Docker se može koristiti za kontejnerizaciju aplikacije i osiguranje dosljednog okruženja između razvoja i produkcije.

7. **Cloud platforma**:

- **Render**: Koristi se za hosting cijele aplikacije (frontend i backend) na javnom internetu.
    - **URL aplikacije**: `https://programsko-inzenjerstvo-projekt-1.onrender.com`
    - **Prednosti Rendera**: jednostavno postavljanje, automatski deploy iz Git repozitorija, SSL certifikati i **besplatna** razina za **manje** projekte.
- **Supabase (hosting baze podataka u oblaku)**: PostgreSQL baza podataka hostana je na Supabaseu, koji koristi AWS infrastrukturu (EU North 1 regija).
    - Omogućuje automatske sigurnosne kopije, nadzor i sigurnosne mehanizme.

UPUTE ZA PUŠTANJE U POGON

1. Instalacija

Potrebne komponente za instalaciju:

- Node.js 22.16.0
- Python 3.13.5
- Pip 25.3
- Django 5.2.8

Sve ostalo dostupno je u requirements.txt: `pip install -r requirements.txt`

Preuzimanje izvornog koda:

Izvorni kod aplikacije dostupan je u Git repozitoriju. Repozitorij se može preuzeti kloniranjem s pomoću alata Git.

1. Potrebno je imati instaliran Git.
2. U naredbenom retku (Command Prompt, PowerShell ili Terminal) pozicionirati se u željeni direktorij.
3. Izvršiti sljedeću naredbu za kloniranje repozitorija: `git clone <URL_GIT_REPOZITORIJA>`
4. Nakon uspješnog kloniranja, ući u direktorij projekta: `cd <IME_PROJEKTA>`

Instalacija ovisnosti

Nakon preuzimanja izvornog koda i ulaska u direktorij projekta, potrebno je instalirati sve potrebne ovisnosti. Instalacija ovisnosti provodi se s pomoću alata npm sljedećom naredbom: `npm install`
Naredba će automatski preuzeti i instalirati sve Node.js pakete definirane u datoteci package.json.

2. Postavke

Konfiguracijske datoteke

Konfiguracija aplikacije provodi se s pomoću konfiguracijskih datoteka koje se nalaze u korijenskom direktoriju projekta. Glavna konfiguracijska datoteka je .env, koja sadrži varijable okruženja potrebne za ispravan rad aplikacije.

Prije pokretanja aplikacije potrebno je kreirati .env datoteku na temelju dostupnog predloška te prilagoditi vrijednosti varijabli prema vlastitom okruženju.

Postavke baze podataka

Baza podataka je već inicijalizirana i sadrži sve potrebne tablice i početne podatke.

Ako je potrebno postaviti bazu iznova (npr. u novom okruženju), koriste se sljedeće naredbe: `npm run db:migrate` - migracija baze podataka i `npm run db:seed` - postavljanje inicijalnih podataka

3. Pokretanje aplikacije

Projekt se sastoji od frontend i backend dijela, koji se pokreću zasebno.

Za pokretanje frontenda, potrebno je biti u root direktoriju projekta i pokrenuti: `npm run dev`

Za pokretanje backenda (Python/Django), potrebno je pozicionirati se u direktorij backend: `cd backend` , zatim pokrenuti razvojni poslužitelj naredbom : `python manage.py runserver`

Backend će tada biti dostupan na lokalnom URL-u, ovisno o konfiguraciji.

4. Upute za administratore

Pristup administratorskom sučelju

Administratori se prijavljuju s pomoću svojih postojećih podataka za prijavu.
Nakon uspješne prijave, aplikacija ih automatski preusmjerava na stranicu admina.

Administratori su odgovorni za:

- Arhiviranje baze podataka:
  Redovito izrađujte sigurnosne kopije baze kako bi se spriječio gubitak podataka.
- Održavanje servera s pomoću Rendera.
- Dodavanje novih korisnika.

---

5. Render platforma

Aplikacija je postavljena na Render cloud platformu, koja omogućuje jednostavno smještanje web aplikacija.
Za više detalja o deployu i konfiguraciji repozitorija, pogledajte našu stranicu:
`https://programsko-inzenjerstvo-projekt-1.onrender.com`

> Napomena: Render besplatni paket koristi tzv. cold start, što znači da se aplikacija može prvo malo sporije pokrenuti nakon perioda neaktivnosti.

Opis prisutpa aplikaciji na javnom poslužitelju

Aplikacija je dostupna na javnom poslužitelju i može joj se pristupiti putem internetskog preglednika.

Pristup korisnicima:

- Otvorite preglednik po izboru (Chrome, Firefox, Edge, itd.).
- U adresnu traku unesite URL aplikacije: `https://programsko-inzenjerstvo-projekt-1.onrender.com`
- Prijavite se sa svojim korisničkim podacima.

Ograničenja

Aplikacija je postavljena na besplatni paket Render platforme, pa se može javiti cold start nakon duljeg perioda neaktivnosti.

Maksimalni broj istovremenih korisnika može biti ograničen prema resursima besplatnog paketa.

Za optimalan rad preporučuje se korištenje modernih preglednika (Chrome, Firefox, Edge).

ZAKLJUČAK I BUDUĆI RAD

Tijekom izrade projekta StanPlan stečeno je vrijedno iskustvo u timskom razvoju softverskog rješenja prema načelima programskog inženjerstva. Projekt je obuhvatio sve faze razvoja, od analize zahtjeva, preko projektiranja i implementacije, do testiranja i dokumentiranja sustava. Aplikacija omogućuje administratorima kreiranje korisnika i upravljanje StanBlog integracijom, predstavnicima suvlasnika jednostavno upravljanje sastancima i objavama, te suvlasnicima pregled objavljenih sastanaka, potvrdu sudjelovanja i primanje e-mail obavijesti, čime se ostvaruje digitalna podrška propisanim procesima upravljanja zgradama.

Tijekom razvoja tim se suočio s tehničkim izazovima poput implementacije sustava za upravljanje stanjima sastanka (Planiran, Objavljen, Obavljen, Arhiviran), pravilnog slanja obavijesti putem elektroničke pošte, autentifikacije korisnika pomoću vanjskih servisa te integracije s aplikacijom StanBlog putem REST API sučelja. Svi identificirani izazovi uspješno su riješeni, a konačna verzija aplikacije ostvaruje svu predviđenu funkcionalnost.

Članovi tima pritom su usvojili nova znanja iz razvoja web aplikacija, dizajna baza podataka, oblikovanja korisničkih sučelja te uporabe suvremenih razvojnih alata i tehnologija.

U budućnosti bi se projekt mogao dodatno unaprijediti kroz optimizaciju performansi, poboljšanje korisničkog iskustva i eventualni razvoj mobilne verzije aplikacije. Također, moguće je proširenje funkcionalnosti sustavom za detaljnije izvještavanje i analitiku aktivnosti unutar zgrade.

Projekt StanPlan predstavlja potpuno funkcionalno rješenje te dobru osnovu za daljnji razvoj digitalnog sustava kojim se poboljšava transparentnost, učinkovitost i suradnja među suvlasnicima stambenih jedinica.

POPIS LITERATURE

1. Programsko inženjerstvo, FER ZEMRIS, http://www.fer.hr/predmet/proinz
2. Astah Community, http://astah.net/editions/uml-new
3. GeeksforGeeks, Benefit of using MVC, dostupno: https://www.geeksforgeeks.org/software-engineering/benefit-of-using-mvc/ (posjećeno: 11. studeni 2025.)
4. Bro code, React Full Course for Beginners [Video](16. siječnja 2024.). dostupno: https://www.youtube.com/watch?v=CgkZ7MvWUAA
5. SeleniumHQ, Selenium Documentation, dostupno: https://www.selenium.dev/documentation/ (posjećeno: 17. siječnja 2026.)

Dnevnik promjena dokumentacije

Rev.	Opis promjene/dodatka	Autori	Datum
0.1	Dodan predložak	Dorotea Perković	11.10.2025.
0.2	Dodavanje opsega projektnog zadatka i moguće nadogradnje	Dorotea Perković	17.10.2025.
0.3	Cilj projekta i slična rješenja	Tea Poplašen	17.10.2025.
0.4	Ispunjavanje dionika u analizi zahtjeva	Borna Matković	18.10.2025.
0.5	Ispunjavanje tablice funkcionalnih zahtjeva	Ivano Markić	18.10.2025.
0.6	Ispunjavanje tablice nefunkcionalnih zahtjeva	Jakov Jauk	18.10.2025.
0.7	Zainteresirane skupine i mogućnost prilagodbe rješenja	Luka Matešković	19.10.2025.
0.8	Dijagrami obrazaca uporabe broj 1,2 i 5	Borna Matković	26.10.2025.
0.9	Sekvencijski dijagrami (2. i 3.)	Dorotea Perković	26.10.2025.
1.0	Opis obrazaca uporabe (1. - 10.)	Ivano Markić	26.10.2025.
1.1	Sekvencijski dijagrami (1.)	Tea Poplašen	26.10.2025.
1.2	Opis obrazaca uporabe (11. - 20.)	Jakov Jauk	26.10.2025.
1.3	Dijagrami obrazaca uporabe broj 3 i 4	Luka Matešković	26.10.2025.
1.4	Baza podataka	Tea Poplašen	3.11.2025.
1.5	Arh. i dizajn sustava (Organizacija sustava na visokoj razini, Organizacija aplikacije)	Dorotea Perković	11.11.2025.
1.6	Arhitektura i dizajn sustava	Tea Poplašen	13.11.2025.
1.7	Promjena i dovršavanje dijagrama razreda	Ivano Markić	14.11.2025.
1.8	Tehnologije za implementaciju aplikacije	Borna Matković	15.01.2026.
1.9	Ispitivanje programskog rješenja prt1. i upute za puštanje u pogon	Dorotea Perković	17.01.2026.

Rev.	Opis promjene/dodatka	Autori	Datum
2.0	Ispitivanje programskog rješenja prt2.	Dorotea Perković	18.01.2026.
2.1	Dijagram stanja, aktivnosti, komponenti i razmještaja	Tea Poplašen	20.01.2026.

Prikaz aktivnosti grupe

1. sastanak

- Datum: 9. listopada 2025.
- Prisustvovali: Tea Poplašen, Jakov Jauk, Ivano Markić, Borna Matković, Luka Matešković, Dorotea Perković
- Teme sastanka:

  - detaljan opis načina rada
  - opis zadatka

2. sastanak

- Datum: 15. listopada 2025.
- Prisustvovali: Tea Poplašen, Jakov Jauk, Ivano Markić, Borna Matković, Luka Matešković, Dorotea Perković
- Teme sastanka:

  - uspostavljena WhatsApp grupa
  - podjela uloga

3. sastanak

- Datum: 23. listopada 2025.
- Prisustvovali: Tea Poplašen, Jakov Jauk, Ivano Markić, Borna Matković, Luka Matešković, Dorotea Perković
- Teme sastanka:

  - prezentiranje obavljenih zadataka asistentu
  - rješavanje nedoumica
  - rasprava o sljedećim zadacima

4. sastanak

- Datum: 3. studenoga 2025.
- Prisustvovali: Tea Poplašen, Jakov Jauk, Ivano Markić, Borna Matković, Luka Matešković, Dorotea Perković
- Teme sastanka:

  - podjela uloga
  - detaljna rasprava o daljnjem napretku rasprave

5. sastanak

- Datum: 13. studenoga 2025.
- Prisustvovali: Tea Poplašen, Jakov Jauk, Ivano Markić, Borna Matković, Luka Matešković, Dorotea Perković
- Teme sastanka:

  > - prezentiranje obavljenih zadataka asistentu prije prve predaje
  > - rješavanje nedoumica

Plan rada

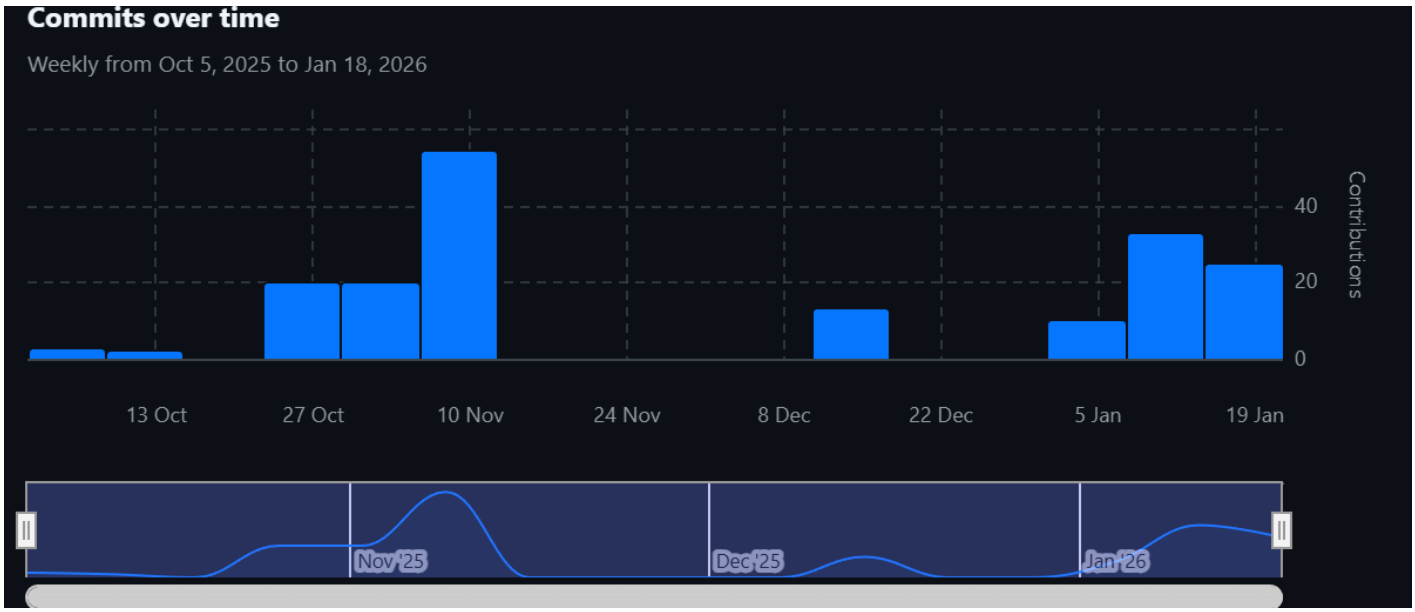
Tablica aktivnosti

Aktivnost	Dorotea Perković	Jakov Jauk	Luka Matešković	Borna Matković	Ivano Markić	Tea Poplašen
Upravljanje projektom	V	S/K	S/K	S/K	S/K	S/K
Opis projektnog zadatka	S	-	S	-	-	S
Funkcionalni zahtjevi	-	S	-	S	S	-
Opis pojedinih obrazaca	K	S	K	K	S	K
Dijagram obrazaca	K	K	S	S	K	K
Sekvencijski dijagrami	S	K	K	K	K	S
Opis ostalih zahtjeva	S	S	S	S	S	S
Arhitektura i dizajn sustava	S	K	K	K	S	S
Baza podataka	-	S	S	-	K	V
Dijagram razreda	-	K	K	-	V	-
Dijagram aktivnosti	-	K	K	-	K	V
Dijagram komponenti	-	K	K	-	K	V
Korištene tehnologije i alati	K	K	K	V	K	K
Ispitivanje programskog rješenja	S	-	-	-	-	S

Aktivnost	Dorotea Perković	Jakov Jauk	Luka Matešković	Borna Matković	Ivano Markić	Tea Poplašen
Dijagram razmještaja	-	K	K	-	K	V
Upute za puštanje u pogon	V	-	-	-	-	-
Dnevnik sastajanja	S	-	-	-	-	S
Zaključak i budući rad	-	-	-	V	-	-
Popis literature	S	K	K	K	K	K

Legenda:
V (Vodi),
S (Sudjeluje),
K (Konzultira se),
- (Nije direktno uključen)

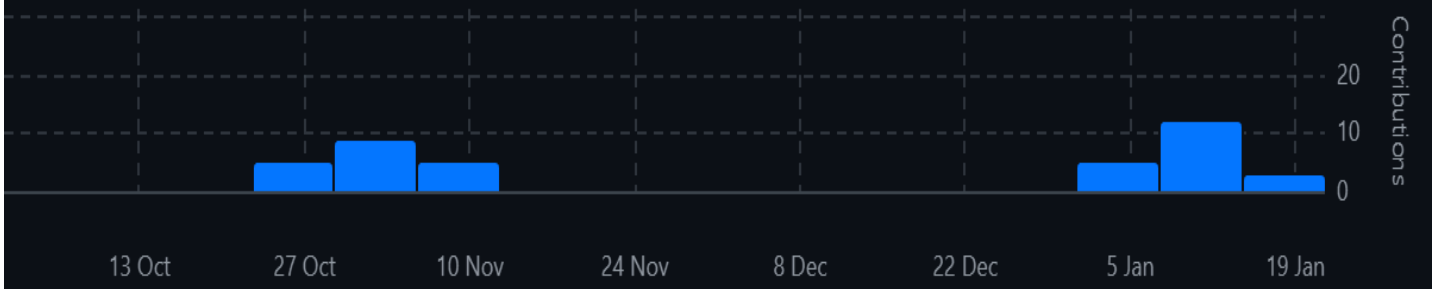
Dijagram pregleda promjena



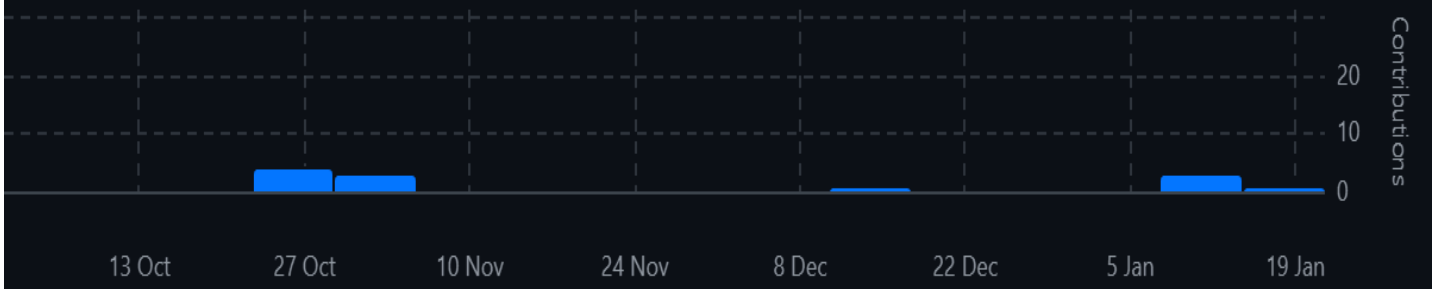
teapoplasen's Commits

jakovjauk's Commits

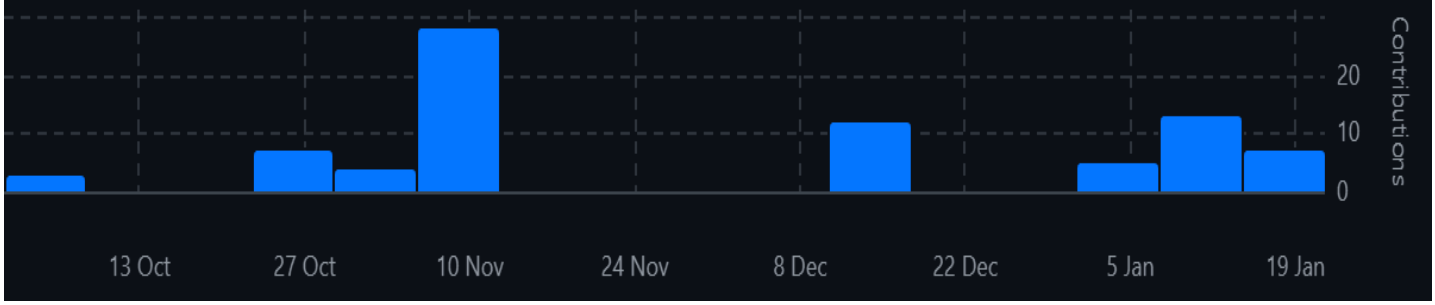
LukaMateskovic1's Commits



BornaMatkovic's Commits



DoroteaPerkovic's Commits



Ispravan broj commitova svake osobe: zbog promjene imena/računa na grafovima se ne prikazuje pravilan broj commitova.

```
80  Dorotea Perković
50  LukaMateskovic1
22  teapoplasen
20  Ivano
12  Jakov Jauk
 7  BornaMatkovic
 6  Borna Matkovic
 5  Ivano Markić
```

Kjučni izazovi i rješenja

Izazovi prve predaje

Opis izazova:
Tijekom razvoja prve verzije projekta susreli smo se s izazovima u implementaciji osnovnih funkcionalnosti backend-a i povezivanju s frontend-om, posebno oko autentifikacije korisnika. Pojavljivali su se i manji Git konflikti prilikom spajanja promjena među članovima tima.

Rješenja i naučene lekcije:
Izazovi su prevladani prioritiziranjem osnovnih funkcionalnosti te redovitim koordiniranjem rada u Git-u. Naučili smo koliko je važno planirati razvoj i testirati integraciju frontenda i backend-a već u ranim fazama, što je doprinijelo boljoj organizaciji i uspješnoj prvoj verziji projekta.

Izazovi druge predaje

Opis izazova:
Tijekom druge faze razvoja projekta fokus je bio na proširivanju postojećih funkcionalnosti, implementaciji složenije poslovne logike te poboljšanju korisničkog sučelja. Kao izazovi su se pojavili usklađivanje frontenda s promjenama u backend API-ju, obrada rubnih slučajeva (npr. validacija podataka i statusi sastanaka) te osiguravanje ispravnog tijeka rada kroz aplikaciju. Također, bilo je potrebno dodatno uskladiti rad članova našeg tima s radom članova drugih timova.

Rješenja i naučene lekcije:
Izazovi su riješeni jasnijom podjelom zadataka, češćom komunikacijom unutar tima i postupnim testiranjem funkcionalnosti nakon svake veće promjene. Naučili smo koliko je važno dosljedno definirati API ugovore između frontenda i backend-a, kao i pravovremeno rješavati potencijalne konflikte u Git-u.

Ova faza razvoja doprinijela je stabilnijoj aplikaciji, boljem korisničkom iskustvu i kvalitetnijoj pripremi projekta za daljnji razvoj.