1. Opis projektnog zadatka

Opis

Apartmeet predstavlja aplikaciju razvijenu s ciljem unapređenja i olakšavanja komunikacije, planiranja i evidencije među suvlasnicima stambenih zgrada. Fokus aplikacije je pružiti jednostavno rješenje za organizaciju sastanaka, razmjenu informacija, praćenje pravnih zaključaka i povezivanje s dodatnim platformama za komunikaciju između suvlasnika. Aplikacija nudi tri osnovne uloge korisnika s različitim razinama pristupa i mogućnosti, a to su: **Administrator**, **Predstavnik suvlasnika** i **Suvlasnik**.

Osnovni funkcijski zahtjevi

1. **Pristup i registracija korisnika**
    - **Administrator** je odgovoran za inicijalno kreiranje računa svih korisnika i dodjeljivanje početnih lozinki. Prilikom prvog prijavljivanja u aplikaciju, korisnici se prijavljuju pomoću dodijeljene početne lozinke, nakon čega im je omogućena izmjena lozinke radi veće sigurnosti. Prijava u Apartmeet aplikaciju odvija se koristeći vanjski sigurnosni servis **OAuth 2.0** radi dodatne zaštite podataka korisnika.
2. **Statusi sastanaka**
    - Sastanci unutar Apartmeet aplikacije prolaze kroz četiri osnovna stanja, koji definiraju status sastanka i mogućnost njegovog uređivanja:
        - **Planiran**: Nakon kreiranja, sastanak je u stanju "Planiran" i sadrži sve osnovne informacije, uključujući naslov, vrijeme održavanja, lokaciju i sažetak sastanka (točke dnevnog reda). **Predstavnik suvlasnika** ima isključivo pravo kreiranja sastanaka, a kao obvezan uvjet sastanak mora uključivati barem jednu točku dnevnog reda kako bi on mogao biti kreiran. Time se osigurava struktura sastanka i omogućuje precizno praćenje tema.
        - **Objavljen**: Kada sastanak postane vidljiv svim suvlasnicima, automatski se šalje obavijest putem emaila, kako bi svi registrirani korisnici bili informirani o nadolazećem događaju. Suvlasnici sada imaju pristup sadržaju sastanka te mogu pregledati točke dnevnog reda i odlučiti o svom sudjelovanju, omogućujući im da na jednostavan način izraze svoje prisustvo na budućem sastanku.
        - **Obavljen**: Nakon održanog sastanka, dolazi se do izrade zaključaka, pri čemu se svakoj točki dnevnog reda može dodijeliti zaključak. Sve točke koje su označene kao pravno važne **moraju** sadržavati zaključak, čime se povećava pravna transparentnost.
        - **Arhiviran**: Ovaj status omogućava automatsko slanje email obavijesti svim suvlasnicima s informacijom o završetku sastanka i njegovom arhiviranju. U ovom statusu više nije moguće uređivati sadržaj sastanka, a točke koje su označene kao pravno obvezujuće označavaju se kao "Izglasane" ili "Odbijene", pružajući jasan pregled rezultata sastanka za sve sudionike.
3. **Povezivanje s platformom Stanblog**
    - Platforma Apartmeet omogućava povezivanje s aplikacijom **Stanblog** tijekom kreiranja točaka dnevnog reda, što korisnicima omogućuje jednostavno povezivanje točaka dnevnog reda s relevantnim diskusijama na platformi Stanblog. Korisnici mogu ubaciti rasprave iz Stanblog aplikacije unutar dnevnog reda sastanka, čime se pruža dodatni kontekst i omogućuje suvlasnicima lakši uvid u prethodne diskusije o specifičnim temama. Na taj način Apartmeet pruža mogućnost povezivanja sa drugim aplikacijama radi razmjene informacija.
4. **Upravljanje ulogama i ograničenja**

- **Administrator**: Ima ulogu kreiranja računa za sve korisnike i dodjeljivanja početnih lozinki. Također, administrator može pregledavati sve registrirane korisnike te dodijeliti uloge i potrebne ovlasti korisnicima unutar aplikacije. Administratori ne sudjeluju u sastancima, ali imaju pristup kompletnom sadržaju aplikacije radi održavanja i sigurnosti.
- **Predstavnik suvlasnika**: Osoba zadužena za zakazivanje sastanaka, jedini korisnik aplikacije koji ima mogućnost kreiranja sastanka. Osim kreiranja, predstavnici suvlasnika mogu ažurirati podatke o sastanku prije nego što sastanak prijeđe u status "Objavljen".
- **Suvlasnik**: Svaki suvlasnik ima pravo pristupa objavljenim sastancima, uvida u dnevni red i mogućnost izražavanja sudjelovanja na sastanku. Prilikom prelaska sastanka u status "Obavljen", suvlasnici mogu pregledavati zaključke, dok u statusu "Arhiviran" imaju uvid u rezultate glasovanja za točke s pravnim učinkom.

5. **Pregled rezultata sastanka**

   - Nakon svakog sastanka koji dobije status "Arhiviran," suvlasnici dobivaju uvid u sve zaključke i rezultate glasovanja, osobito za točke s pravnim učinkom koje se bilježe kao "Izglasane" ili "Odbijene." Ovaj jasan prikaz rezultata pomaže suvlasnicima u praćenju važnih odluka koje utječu na zajedničke interese i održavanje zgrade.

6. **Automatske obavijesti i podsjetnici**

   - **Apartmeet** omogućava slanje automatskih obavijesti putem emaila za svaki važan korak u organizaciji sastanka, uključujući objavu sastanka, promjene u dnevnom redu, te obavijesti o održavanju i arhiviranju sastanka. Osim toga, aplikacija šalje podsjetnike neposredno prije sastanka kako bi se osiguralo da suvlasnici budu pravovremeno obaviješteni.

7. **Sigurnost podataka i transparentnost**

   - Korištenjem **OAuth 2.0** sustava za prijavu, Apartmeet osigurava visoku razinu zaštite podataka korisnika. Osim toga, aplikacija je dizajnirana s naglaskom na transparentnost, omogućujući suvlasnicima stalan i jednostavan uvid u sve aktivnosti i odluke koje se donose unutar zgrade. Time se smanjuje mogućnost nesporazuma.

8. **Označavanje prisutnosti na sastanku**
   -Suvlasnici imaju mogućnost označiti svoje sudjelovanje na sastanku dok je u stanju "Objavljen," što predstavlja potvrdu njihovog prisustva i olakšava organizaciju sastanka.

Osnovni nefunkcijski zahtjevi

1. **Zaštita korisničkih podataka**

   - Aplikacija mora osigurati visoku razinu zaštite korisničkih podataka, uključujući lozinke i osobne informacije, kako bi se zaštitila privatnost korisnika.

2. **Upravljanje ulogama korisnika**

   - Administrator mora imati mogućnost upravljanja ulogama korisnika, što uključuje dodjeljivanje i oduzimanje ovlasti suvlasnicima i predstavnicima

3. **Kompatibilnost s uređajima**

   - Aplikacija mora biti kompatibilna s različitim uređajima, poput desktop računala i pametnih telefona, kako bi korisnici mogli pristupiti aplikaciji s bilo kojeg uređaja.

4. **Jednostavnost korištenja**

   - Aplikacija mora biti jednostavna za korištenje, s intuitivnim korisničkim sučeljem koje ne zahtijeva posebnu obuku za korisnike.

Mogući funkcijski zahtjevi

1. **Kreiranje glasanja**
    - Predstavnik suvlasnika mora moći kreirati glasanje za određene točke dnevnog reda koje zahtijevaju donošenje odluke.
2. **Bilježenje i prikaz rezultata glasanja**
    - Sustav mora bilježiti rezultate glasanja i automatski prikazivati zaključke kao "Izglasano" ili "Odbijeno".
3. **Prikaz detaljnih rezultata nakon glasanja**
    - Nakon završetka glasanja, sustav mora prikazati rezultate (broj glasova za, protiv i suzdržanih) svim korisnicima.
4. **Vidljivost tijekom glasanja**
    - Tijekom glasanja, korisnici moraju moći vidjeti koliko je osoba već glasalo.

Mogući nefunkcijski zahtjevi

1. **Automatska provjera kvoruma**
    - Sustav mora automatski provjeriti i potvrditi da je postignut potreban kvorum za valjanost glasanja.
2. **Ograničenje glasanja**
    - Sustav mora osigurati da samo ovlašteni suvlasnici mogu sudjelovati u glasanju i da svaki suvlasnik može glasati samo jednom.

Tehnologije

Frotnend - React, Figma
Backend - ASP.NET Core

Zaključak

Apartmeet je osmišljen kao rješenje za upravljanje komunikacijom i organizacijom u stambenim zgradama. Aplikacija omogućuje transparentnost, lakše praćenje pravno važnih odluka i nudi fleksibilnost kroz povezivanje s dodatnim platformama. Na taj način, Apartmeet olakšava svakodnevno funkcioniranje zajednice, smanjuje nesporazume i osigurava pravnu usklađenost svih aktivnosti unutar zajednice.

Moguće nadogradnje

Izrada mobilne aplikacije – omogućilo bi suvlasnicima lakši pristup aplikaciji, također dobivali bi obavijesti u stvarnom vremenu umjesto putem e-maila, integracija sa kalendarom (automatsko ubacivanje termina sastanka)
Glasovnja za točke dnevnog reda - omogućilo bi uštedu vremena suvlasnicima oko točaka dnevnog reda za koje nije potrebna diskusija već samo njihova odluka ("Za ili protiv")

Chat za suvlasnike - omogućilo bi bolju povezanost i komunikaciju između suvlasnika
Upravljanje financijama – Proširenje aplikacije na područje financijskog upravljanja zgradom

Postojeća rješenja

Postoje aplikacije sličnih funkcionalnosti poput mSuvlasnik, Moja Zgrada i eUpravitelj.
Moguća nadogradnja naše aplikacije, a moguća je u mSuvlasnik-u je glasanje unutar aplikacije eUpravitelj uz evidenciju sastanaka nudi mogućnosti poput prijave štete, traženje majstora, uvid u financije zgrade i druge.

2. Analiza zahtjeva

Funkcijski zahtjevi

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
F-01	Administrator mora imati mogućnost kreiranja svih korisničkih računa unutar aplikacije i dodjeljivanja inicijalnih lozinki.	Visok	Zahtjev dionika	Administrator uspješno kreira nove korisničke račune i dodjeljuje inicijalne lozinke.
F-02	Korisnici moraju moći promijeniti inicijalnu lozinku nakon prve prijave.	Visok	Zahtjev dionika	Korisnici uspješno mijenjaju inicijalnu lozinku prilikom prve prijave.
F-03	Predstavnik suvlasnika mora imati mogućnost zakazivanja sastanaka, definiranja dnevnog reda i upravljanja zaključcima.	Visok	Zahtjev dionika	Predstavnik uspješno zakazuje sastanke, definira dnevni red i upravlja zaključcima.
F-04	Aplikacija mora omogućiti prijavu korisnika putem vanjskog servisa za autentifikaciju Oauth 2.0.	Srednji	Zahtjev dionika	Korisnici se uspješno prijavljuju putem Oauth 2.0 autentifikacijskog servisa.
F-05	Suvlasnici moraju imati mogućnost prijave, pregleda zakazanih sastanaka	Visok	Zahtjev dionika	Suvlasnici se uspješno prijavljuju, pregledavaju zakazane sastanke.
F-06	Predstavnik mora moći kreirati nove sastanke sa definiranim dnevnim redom i točkama koje mogu, ali i ne moraju imati pravni učinak.	Visok	Zahtjev dionika	Predstavnik kreira sastanak s definiranim dnevnim redom i točkama koje mogu, ali i ne moraju imati pravni učinak.
F-07	Predstavnik kreira sastanak koji se nalazi u stanju 'Planiran' sve dok se ne navede naslov, mjesto, vrijeme i točke dnevnog reda budućeg sastanka.	Visok	Zahtjev dionika	Predstavnik uspješno kreira sastanak koji se nalazi u stanju 'Planiran' sve dok se ne navedu nužni podaci.
F-08	Kada su ispunjene sve informacije o sastanku (mjesto, vrijeme, točke dnevnog reda), predstavnik ga prebacuje u stanje 'Objavljen', tada postaje vidljiv svim suvlasnicima.	Visok	Zahtjev dionika	Sastanak prelazi u stanje 'Objavljen'.
F-09	Aplikacija mora automatski slati e-mail podsjetnike korisnicima za nadolazeće sastanke.	Srednji	Zahtjev dionika	Korisnici primaju podsjetnike za nadolazeće sastanke.
F-10	Suvlasnici mogu pregledavati točke dnevnog reda sastanka dok je u stanju 'Objavljen'.	Srednji	Zahtjev dionika	Suvlasnici pregledavaju točke dnevnog reda za sastanke u stanju 'Objavljen'.

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
F-11	Suvlasnici mogu označiti da će sudjelovati na sastanku putem aplikacije.	Srednji	Zahtjev dionika	Suvlasnici uspješno označavaju sudjelovanje putem aplikacije.
F-12	Nakon što je sastanak završio, Predstavnik prebacuje stanje sastanka u 'Obavljen'.	Srednji	Zahtjev dionika	Sastanak prelazi u stanje 'Obavljen'.
F-13	Predstavnik može dodavati zaključke za svaku točku dnevnog reda nakon što je sastanak prešao u stanje 'Obavljen'.	Visok	Zahtjev dionika	Predstavnik dodaje zaključke nakon što sastanak pređe u stanje 'Obavljen'.
F-14	Točke dnevnog reda s pravnim učinkom moraju imati zaključak kojeg unosi Predstavnik.	Visok	Zahtjev dionika	Zaključci za točke s pravnim učinkom su dodani od strane Predstavnika prije arhiviranja.
F-15	Nakon obrade zaključaka, Predstavnik prebacuje stanje sastanka u 'Arhiviran', a suvlasnici primaju obavijest putem e-maila.	Srednji	Zahtjev dionika	Sastanak prelazi u stanje 'Arhiviran', a obavijest se šalje e-mailom.
F-16	Nakon arhiviranja, sadržaj sastanka i zaključci se više ne mogu mijenjati.	Srednji	Zahtjev dionika	Sadržaj i zaključci arhiviranog sastanka nisu promjenjivi.
F-17	Aplikacija mora omogućiti povezivanje točaka dnevnog reda sa specifičnim diskusijama na aplikaciji StanBlog.	Srednji	Zahtjev dionika	Točke dnevnog reda povezane su s diskusijama na StanBlogu.
F-18	Suvlasnici moraju moći pregledavati povijest sastanaka i njihovih zaključaka nakon što su arhivirani.	Nizak	Zahtjev dionika	Suvlasnici pregledavaju povijest sastanaka i zaključaka.
F-19	Aplikacija mora moći mijenjati status sastanka, od "Planiran" do "Objavljen", "Obavljen" i "Arhiviran".	Visok	Zahtjev dionika	Status sastanka se može mijenjati u skladu s pravilima aplikacije.

Ostali zahtjevi

Nefunkcijski zahtjevi

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
NF-01	Aplikacija mora imati zaštitu za korisničke podatke, poput lozinki i osobnih informacija.	Visok	Sigurnosni zahtjev	Korisnički podaci su zaštićeni enkripcijom i sigurnosnim protokolima; aplikacija prolazi sigurnosne provjere.
NF-02	Administrator mora moći upravljati ulogama korisnika, poput dodjeljivanja ili oduzimanja ovlasti suvlasnicima i predstavnicima.	Visok	Administrativni zahtjev	Administrator uspješno dodjeljuje i oduzima ovlasti korisnicima bez grešaka.
NF-03	Aplikacija mora biti kompatibilna s različitim uređajima (desktop, pametni telefoni).	Srednji	Tehnički zahtjev	Aplikacija funkcionira bez problema na desktopu i pametnim telefonima; provjera kroz testiranje na više uređaja.
NF-04	Aplikacija mora biti jednostavna za korištenje, s intuitivnim korisničkim sučeljem koje ne zahtijeva posebnu obuku.	Srednji	Korisnički zahtjev	Korisnici mogu koristiti aplikaciju bez potrebe za dodatnom obukom, što se potvrđuje kroz testove upotrebljivosti.

Dionici

	Dionik
1	Vlasnik
2	Predstavnik (korisnik)
3	Suvlasnik (korisnik)
4	Administrator
5	Razvojni Tim

Aktori i njihovi funkcijski zahtjevi

ID	Dionik (uloga)	ID zahtjeva
A-1	Predstavnik (inicijator)	F-03, F-06, F-07, F-13, F-14
A-2	Predstavnik (sudionik)	F-02, F-08, F-09, F-12, F-15, F-16, F-19
A-3	Suvlasnik (inicijator)	F-11
A-4	Suvlasnik (sudionik)	F-02, F-05, F-08, F-09, F-10, F-12, F-15, F-16, F-18, F-19
A-5	Administrator (inicijator)	F-01
A-6	StanBlog API (sudionik)	F-17
A-7	OAuth2.0 API (inicijator)	F-04

3. Specifikacija zahtjeva sustava

Dijagram obrazaca uporabe za kritične sustave i integracija

UC1 - Kreiranje korisničkih računa

- **Glavni sudionik**: Administrator
- **Cilj**: Omogućiti administratoru kreiranje korisničkih računa unutar aplikacije.
- **Sudionici**: Baza podataka
- **Preduvjet**: Administrator je prijavljen.
- **Opis osnovnog tijeka**:
    1. Administrator odabire opciju za kreiranje korisničkog računa. (F-01)
    2. Aplikacija prikazuje obrazac za unos podataka.
    3. Administrator unosi potrebne podatke i inicijalnu lozinku.
    4. Administrator potvrđuje unos.
    5. Sustav pohranjuje podatke i prikazuje potvrdu o uspješnom kreiranju računa.
- **Opis mogućih odstupanja**:
  iii.a Ako podaci nisu uneseni ispravno, sustav prikazuje poruku o grešci i vraća administratora na obrazac.

UC2 - Promjena inicijalne lozinke

- **Glavni sudionik**: Korisnik
- **Cilj**: Omogućiti korisnicima promjenu inicijalne lozinke nakon prve prijave.
- **Sudionici**: Sustav za autentifikaciju
- **Preduvjet**: Korisnik je prijavljen.
- **Opis osnovnog tijeka**:

1. Korisnik odabire opciju za promjenu lozinke. (F-02)
2. Sustav prikazuje obrazac za unos trenutne i nove lozinke.
3. Korisnik unosi staru lozinku i novu lozinku.
4. Korisnik potvrđuje promjenu.
5. Sustav ažurira lozinku i obavještava korisnika o uspješnoj promjeni.

- **Opis mogućih odstupanja**:
  iii.a Ako unesena lozinka nije točna, sustav prikazuje poruku o grešci i traži ponovni unos.

UC3 - Zakazivanje sastanka

- **Glavni sudionik**: Predstavnik
- **Cilj**: Omogućiti predstavniku zakazivanje sastanka, definiranje dnevnog reda i upravljanje zaključcima.
- **Sudionici**: Baza podataka
- **Preduvjet**: Predstavnik je prijavljen.
- **Opis osnovnog tijeka**:
  1. Predstavnik odabire opciju za zakazivanje sastanka. (F-03)
  2. Aplikacija prikazuje obrazac za unos podataka.
  3. Predstavnik unosi sve potrebne informacije.
  4. Predstavnik potvrđuje unos.
  5. Sustav pohranjuje sastanak i prikazuje ga kao "Planiran".
- **Opis mogućih odstupanja**:
  iii.a Ako podaci nisu potpuni, sustav prikazuje poruku o grešci i traži dopunu.

UC4 - Prijava putem Oauth 2.0

- **Glavni sudionik**: Korisnik
- **Cilj**: Omogućiti korisniku prijavu pomoću Oauth 2.0 servisa za autentifikaciju.
- **Sudionici**: Vanjski servis za autentifikaciju
- **Preduvjet**: Korisnik je odabrao opciju prijave putem Oauth 2.0.
- **Opis osnovnog tijeka**:
  1. Korisnik klikne na opciju prijave putem Oauth 2.0. (F-04)
  2. Sustav preusmjerava korisnika na stranicu vanjskog servisa za prijavu.
  3. Korisnik unosi svoje korisničke podatke.
  4. Vanjski servis provjerava korisnikove podatke i šalje potvrdu natrag aplikaciji.
  5. Sustav prijavljuje korisnika i preusmjerava ga na aplikaciju.
- **Opis mogućih odstupanja**:
  iii.a Ako vanjski servis trenutno nije dostupan, aplikacija obavještava korisnika o problemu.

UC5 - Pregled i označavanje sudjelovanja na sastanku

- **Glavni sudionik**: Suvlasnik
- **Cilj**: Omogućiti suvlasnicima pregled i označavanje sudjelovanja na sastanku.
- **Sudionici**: Baza podataka
- **Preduvjet**: Suvlasnik je prijavljen.
- **Opis osnovnog tijeka**:
  1. Suvlasnik odabire opciju za pregled sastanaka. (F-05)
  2. Aplikacija prikazuje listu dostupnih sastanaka.
  3. Suvlasnik odabire sastanak i označava sudjelovanje.
  4. Sustav bilježi promjenu.
  5. Aplikacija prikazuje potvrdu.
- **Opis mogućih odstupanja**:
  iii.a Ako postoji greška pri označavanju, aplikacija prikazuje poruku.

UC6 - Kreiranje sastanka s pravnim učinkom

- **Glavni sudionik**: Predstavnik

- **Cilj**: Omogućiti kreiranje sastanaka s dnevnim redom koji može imati pravni učinak.
- **Sudionici**: Baza podataka
- **Preduvjet**: Predstavnik je prijavljen.
- **Opis osnovnog tijeka**:
    1. Predstavnik odabire opciju za kreiranje sastanka. (F-06)
    2. Predstavnik unosi točke dnevnog reda koje mogu imati pravni učinak.
    3. Predstavnik potvrđuje unos.
    4. Sustav pohranjuje sastanak.
    5. Sastanak se prikazuje kao "Planiran".
- **Opis mogućih odstupanja**:
  ii.a Ako podaci nisu uneseni ispravno, sustav vraća korisnika na obrazac za unos i prikazuje poruku o grešci.

UC7 - Početno stanje kreiranog sastanka - 'Planiran'

- **Glavni sudionik**: Predstavnik
- **Cilj**: Postaviti početno stanje sastanka u 'Planiran'.
- **Sudionici**: Baza podataka
- **Preduvjet**: Predstavnik je prijavljen.
- **Opis osnovnog tijeka**:
    1. Predstavnik odabire opciju za kreiranje sastanka. (F-07)
    2. Unosi naslov, mjesto, vrijeme i sažetak sastanka.
    3. Sustav pohranjuje sastanak u stanju 'Planiran'.
    4. Aplikacija prikazuje potvrdu o kreiranju.
- **Opis mogućih odstupanja**:
  ii.a Ako nije unesen neki obavezni podatak, sustav prikazuje poruku o grešci i vraća korisnika na obrazac za unos.

UC8 - Objavljivanje sastanka

- **Glavni sudionik**: Predstavnik
- **Cilj**: Omogućiti predstavniku objavljivanje sastanka kako bi postao vidljiv suvlasnicima, uz automatsko obavještavanje.
- **Sudionici**: Baza podataka
- **Preduvjet**: Sastanak se nalazi u statusu 'Planiran'.
- **Opis osnovnog tijeka**:
    1. Predstavnik pregledava listu sastanaka i odabire onaj s oznakom 'Planiran'. (F-08)
    2. Predstavnik klikne na gumb za objavu sastanka.
    3. Sustav vrši provjeru kako bi osigurao da su svi potrebni detalji sastanka ispunjeni.
    4. Nakon uspješne provjere, sustav mijenja status sastanka na 'Objavljen'.
    5. Sustav šalje automatsku e-mail obavijest svim suvlasnicima.
- **Opis mogućih odstupanja**:
  iii.a Ako sastanak nije ispravno popunjen sustav prikazuje poruku o grešci te nudi mogućnost vraćanja na stranicu za uređivanje kako bi se dopunili potrebni podaci.

UC9 - Slanje podsjetnika za sastanke

- **Glavni sudionik**: Sustav
- **Cilj**: Automatski slati podsjetnike korisnicima za nadolazeće sastanke.
- **Sudionici**: Baza podataka, sustav za slanje e-maila
- **Preduvjet**: Sastanak je zakazan i približava se datum održavanja.
- **Opis osnovnog tijeka**:
    1. Sustav prepoznaje nadolazeći sastanak. (F-09)
    2. Sustav generira e-mail podsjetnike.
    3. Sustav šalje podsjetnike korisnicima.
    4. Korisnici primaju obavijesti na svoje e-mail adrese.

- **Opis mogućih odstupanja**:
  iii.a Ako sustav za slanje e-maila nije dostupan, podsjetnici se ne šalju i sustav bilježi grešku.

UC10 - Pregled o informacijama sastanka u stanju 'Objavljen'

- **Glavni sudionik**: Suvlasnik
- **Cilj**: Omogućiti pregled bitnih informacija za objavljeni sastanak (vrijeme, mjesto, točke dnevnog reda, itd...)
- **Sudionici**: Baza podataka
- **Preduvjet**: Suvlasnik je prijavljen.
- **Opis osnovnog tijeka**:
  1. Suvlasnik odabire opciju za pregled zakazanih sastanaka. (F-10)
  2. Aplikacija prikazuje listu sastanaka.
  3. Suvlasnik odabire sastanak u stanju 'Objavljen'.
  4. Sustav prikazuje točke dnevnog reda i ostale informacija.
- **Opis mogućih odstupanja**:
  ii.a Ako ne postoje dostupni sastanci u stanju 'Objavljen', aplikacija prikazuje poruku o nedostupnosti sastanaka.

UC11 - Označavanje sudjelovanja na sastanku

- **Glavni sudionik**: Suvlasnik
- **Cilj**: Omogućiti korisniku da označi svoje sudjelovanje na sastanku.
- **Sudionici**: Baza podataka
- **Preduvjet**: Suvlasnik je prijavljen i sastanak je u stanju 'Objavljen'.
- **Opis osnovnog tijeka**:
  1. Suvlasnik odabire opciju za pregled sastanaka. (F-11)
  2. Aplikacija prikazuje listu sastanaka.
  3. Suvlasnik odabire sastanak i označava sudjelovanje.
  4. Sustav bilježi promjenu sudjelovanja.
  5. Aplikacija prikazuje potvrdu o sudjelovanju.
- **Opis mogućih odstupanja**:
  iii.a Ako korisnik već ima označeno sudjelovanje, aplikacija prikazuje obavijest.

UC12 - Prelazak sastanka u stanje 'Obavljen'

- **Glavni sudionik**: Predstavnik
- **Cilj**: Omogućiti prebacivanje sastanka u stanje 'Obavljen' nakon završetka.
- **Sudionici**: Baza podataka
- **Preduvjet**: Predstavnik je prijavljen i sastanak je završen.
- **Opis osnovnog tijeka**:
  1. Predstavnik odabire sastanak koji je završen. (F-12)
  2. Predstavnik odabire opciju za prelazak u stanje 'Obavljen'.
  3. Sustav provjerava uvjete i mijenja status sastanka.
  4. Sustav pohranjuje promjenu i prikazuje potvrdu.
- **Opis mogućih odstupanja**:
  ii.a Ako sastanak nije završen ili nedostaju podaci, sustav prikazuje poruku o grešci.

UC13 - Dodavanje zaključaka na točke dnevnog reda

- **Glavni sudionik**: Predstavnik
- **Cilj**: Omogućiti dodavanje zaključaka za svaku točku dnevnog reda nakon prelaska sastanka u stanje 'Obavljen'.
- **Sudionici**: Baza podataka
- **Preduvjet**: Sastanak je u stanju 'Obavljen'.
- **Opis osnovnog tijeka**:
  1. Predstavnik odabire sastanak u stanju 'Obavljen'. (F-13)

2. Predstavnik dodaje zaključke za točke dnevnog reda.
3. Predstavnik potvrđuje unos.
4. Sustav pohranjuje zaključke i prikazuje potvrdu.

- **Opis mogućih odstupanja**:
  ii.a Ako se zaključci ne unesu ispravno, aplikacija prikazuje poruku o grešci i traži ponovni unos.

UC14 - Dodavanje zaključaka za točke s pravnim učinkom

- **Glavni sudionik**: Predstavnik
- **Cilj**: Omogućiti unos zaključaka za točke dnevnog reda s pravnim učinkom prije arhiviranja sastanka.
- **Sudionici**: Baza podataka
- **Preduvjet**: Sastanak je u stanju 'Obavljen' i točke imaju pravni učinak.
- **Opis osnovnog tijeka**:
  1. Predstavnik odabire sastanak. (F-14)
  2. Predstavnik dodaje zaključke za točke s pravnim učinkom.
  3. Predstavnik potvrđuje unos.
  4. Sustav pohranjuje podatke i prikazuje potvrdu.
- **Opis mogućih odstupanja**:
  ii.a Ako zaključci nisu uneseni, sustav ne dopušta arhiviranje sastanka i prikazuje upozorenje.

UC15 - Arhiviranje sastanka

- **Glavni sudionik**: Predstavnik
- **Cilj**: Omogućiti arhiviranje sastanka i slanje obavijesti suvlasnicima.
- **Sudionici**: Baza podataka, sustav za slanje e-maila
- **Preduvjet**: Sastanak je u stanju 'Obavljen' i svi zaključci su dodani.
- **Opis osnovnog tijeka**:
  1. Predstavnik odabire opciju za arhiviranje sastanka. (F-15)
  2. Sustav provjerava status sastanka.
  3. Sustav arhivira sastanak.
  4. Sustav automatski šalje obavijesti e-mailom suvlasnicima.
- **Opis mogućih odstupanja**:
  i.a Ako sastanak nije spreman za arhiviranje, sustav prikazuje obavijest o grešci i traži dodatne korake.

UC16 - Ograničenje izmjena arhiviranog sastanka

- **Glavni sudionik**: Sustav
- **Cilj**: Osigurati da se arhivirani sastanak i njegovi zaključci ne mogu mijenjati.
- **Sudionici**: Baza podataka
- **Preduvjet**: Sastanak je u stanju 'Arhiviran'.
- **Opis osnovnog tijeka**:
  1. Sustav zaključava sadržaj sastanka nakon arhiviranja. (F-16)
  2. Korisnici pokušavaju pristupiti i mijenjati sadržaj.
  3. Sustav odbija pokušaj izmjene i prikazuje obavijest o zabrani.
- **Opis mogućih odstupanja**:
  ii.a Sustav automatski sprečava sve pokušaje izmjena arhiviranih podataka.

UC17 - Povezivanje točaka dnevnog reda s diskusijama na StanBlogu

- **Glavni sudionik**: Predstavnik
- **Cilj**: Omogućiti povezivanje točaka dnevnog reda s relevantnim diskusijama na StanBlogu.
- **Sudionici**: Baza podataka, StanBlog sustav
- **Preduvjet**: Predstavnik je prijavljen i sastanak je u stanju 'Objavljen'.
- **Opis osnovnog tijeka**:
  1. Predstavnik odabire sastanak u stanju 'Objavljen'. (F-17)
  2. Odabire točku dnevnog reda za povezivanje.
  3. Predstavnik unosi poveznicu na diskusiju na StanBlogu.

4. Predstavnik potvrđuje povezivanje.
5. Sustav pohranjuje poveznicu i prikazuje potvrdu.

- **Opis mogućih odstupanja**:
  iii.a Ako StanBlog sustav nije dostupan, aplikacija prikazuje obavijest o grešci.

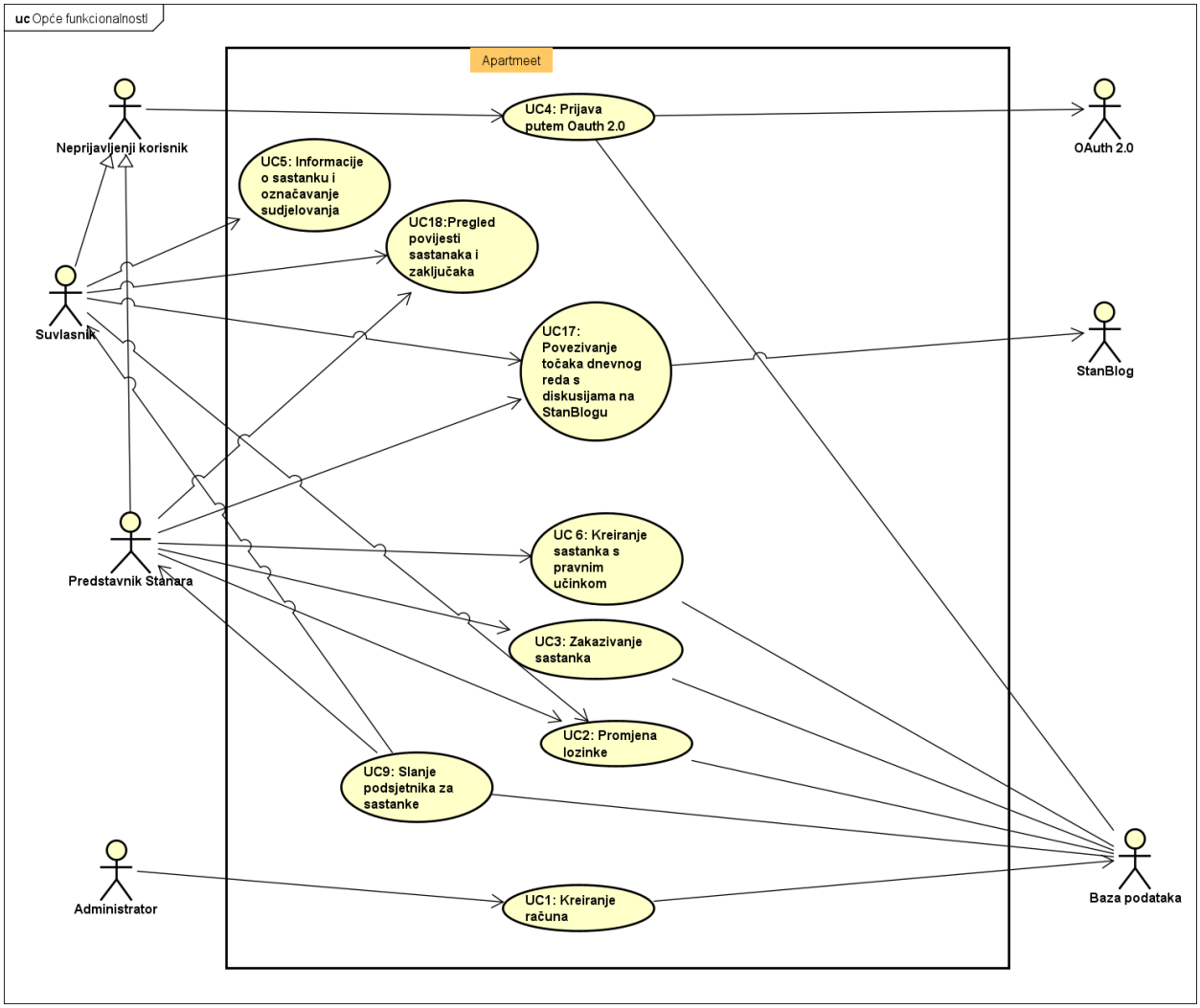
UC18 - Pregled povijesti sastanaka i zaključaka

- **Glavni sudionik**: Suvlasnik
- **Cilj**: Omogućiti pregled povijesti arhiviranih sastanaka i njihovih zaključaka.
- **Sudionici**: Baza podataka
- **Preduvjet**: Suvlasnik je prijavljen.
- **Opis osnovnog tijeka**:
  1. Suvlasnik odabire opciju 'Povijest sastanaka'. (F-18)
  2. Aplikacija prikazuje popis arhiviranih sastanaka.
  3. Suvlasnik odabire sastanak za pregled.
  4. Sustav prikazuje povijest zaključaka.
- **Opis mogućih odstupanja**:
  ii.a Ako nema arhiviranih sastanaka, sustav prikazuje poruku o nedostupnosti podataka.

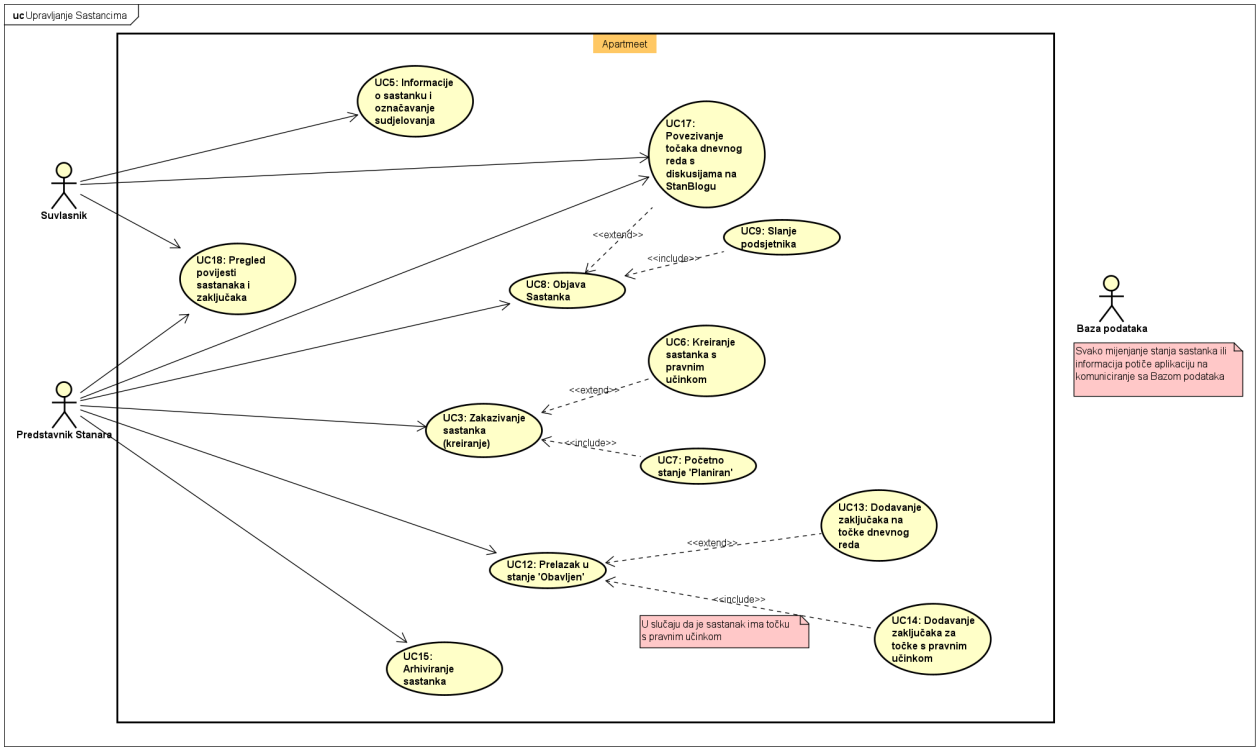
UC19 - Mijenjanje statusa sastanka

- **Glavni sudionik**: Predstavnik
- **Cilj**: Omogućiti mijenjanje statusa sastanka između 'Planiran', 'Objavljen', 'Obavljen' i 'Arhiviran'.
- **Sudionici**: Baza podataka
- **Preduvjet**: Predstavnik je prijavljen.
- **Opis osnovnog tijeka**:
  1. Predstavnik odabire sastanak za izmjenu statusa. (F-19)
  2. Predstavnik odabire novi status.
  3. Sustav provjerava uvjete za promjenu.
  4. Sustav mijenja status sastanka.
  5. Aplikacija prikazuje potvrdu o promjeni.
- **Opis mogućih odstupanja**:
  ii.a Ako uvjeti za promjenu statusa nisu zadovoljeni, sustav prikazuje poruku o grešci.

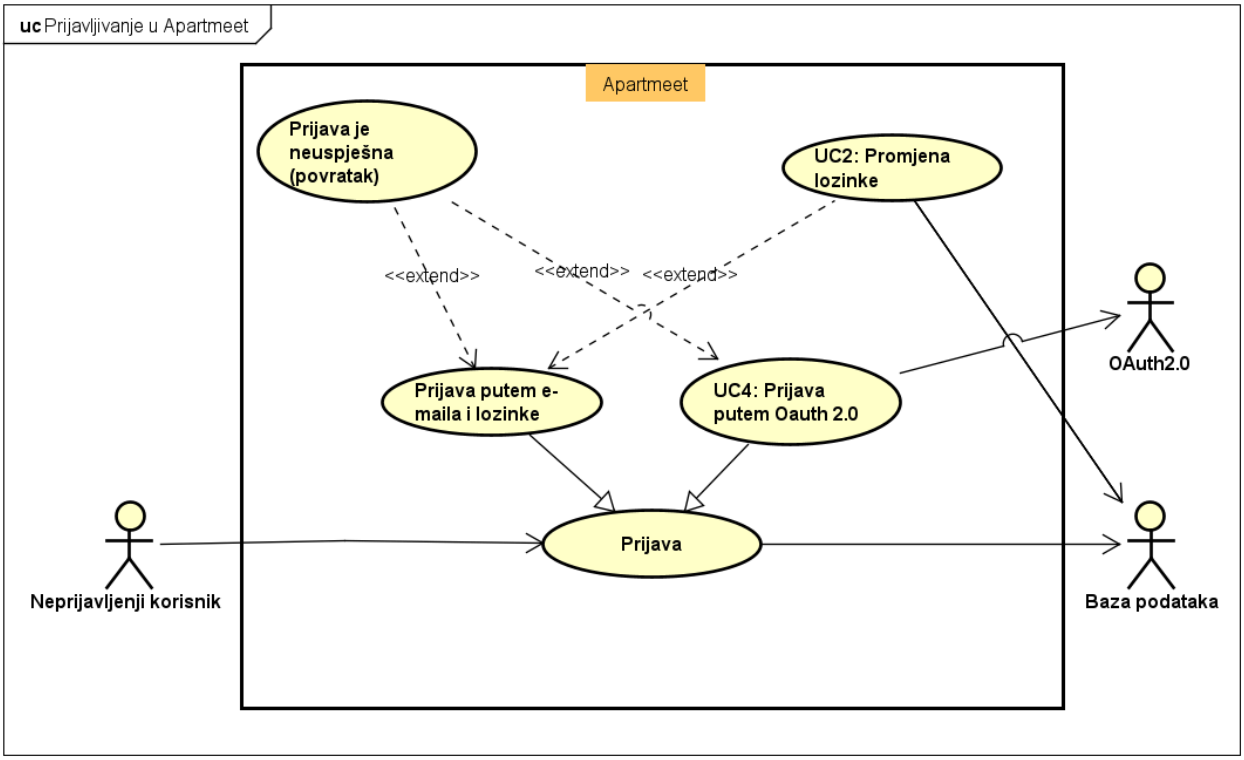
Dijagrami obrazaca



Slika 3.1 Dijagram obrazaca prikazuje glavne funkcionalnosti aplikacije Apartmeet i interakcije korisnika s njom. **Korisnik bez statusa** može koristiti osnovne funkcije poput prijave uz pomoć vanjskog servisa **OAuth 2.0**. **Administrator** upravlja kreiranjem korisničkih računa, dok **Predstavnik Stanara** organizira sastanke. **Suvlasnici** i **Predstavnik Stanara** mogu pregledavati sastanke i potvrđivati svoje sudjelovanje, također mogu povezati točke dnevnog reda sa diskusijama na platformi **StanBlog**. Aplikacija također omogućava slanje podsjetnika korisnicima i integraciju s vanjskim servisima kako bi se poboljšala koordinacija i komunikacija među korisnicima.



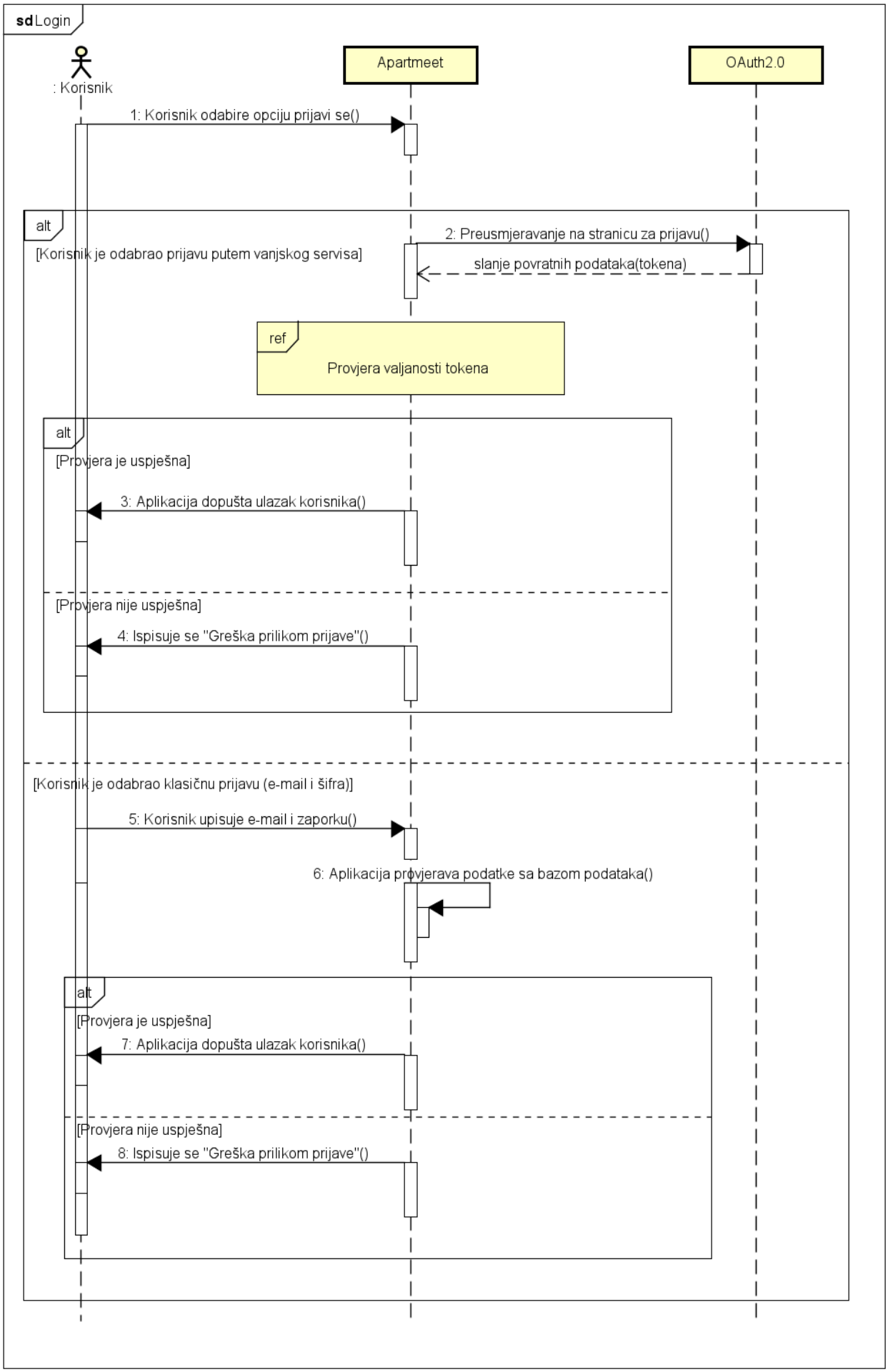
Slika 3.2 Dijagram obrazaca prikazuje kako aplikacija Apartmeet upravlja sastancima. **Predstavnik Stanara** ima mogućnost kreiranja sastanaka (**UC3**) i dodavanja točaka s pravnim učinkom (**UC6**), dok se sastanak nakon toga stavlja u stanje 'Planiran' (**UC7**). Kada je spreman, sastanak se objavljuje (**UC8**), a korisnicima se šalju podsjetnici (**UC9**). **Suvlasnici** mogu pregledati objavljene sastanke i označiti svoje sudjelovanje (**UC5**). Nakon završetka, sastanak prelazi u stanje 'Obavljen' (**UC12**), a predstavnik može dodati zaključke na točke dnevnog reda (**UC13**, **UC14**). Za točke dnevnog reda koje imaju pravni učinak **Predstavnik** je dužan napisati zaključak vezanu za tu točku. Završeni sastanci se arhiviraju (**UC15**), a korisnici imaju pristup povijesti sastanaka i zaključaka (**UC18**). Svaka promjena ili ažuriranje komunicira se s **Bazom podataka**, kako bi podaci bili ažurirani i pohranjeni.



Slika 3.3 Dijagram obrazaca prikazuje proces prijavljivanja korisnika u aplikaciju Apartmeet. **Neprijavljeni korisnik** započinje proces prijave. Proces prijave je moguće obaviti uz pomoć vanjskog servisa **OAuth 2.0** ili klasičnim putem (e-mail i lozinka). Ako je prijava uspješna, korisniku se dodjeljuje status **Suvlasnika** ili

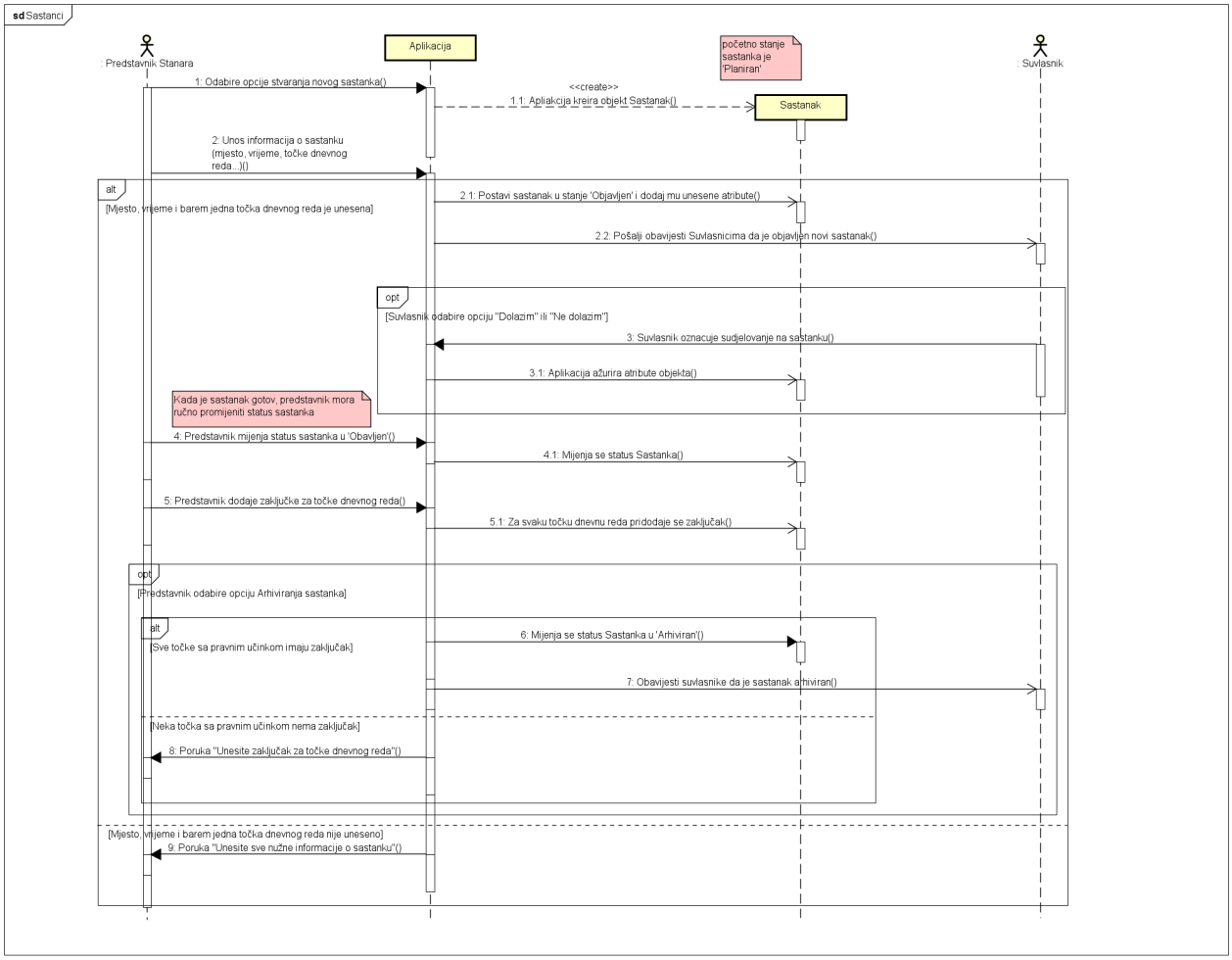
Predstavnika. Nakon uspješne provjere **Korisnik** može iskoristiti funkciju promjene zaporke. U slučaju da prijava nije uspješna, aplikacija prikazuje obavijest o grešci korisniku i vraća ga na početni zaslon. Svi ovi procesi komuniciraju s **Bazom podataka** radi provjere i ažuriranja podataka.

Sekvencijski dijagrami

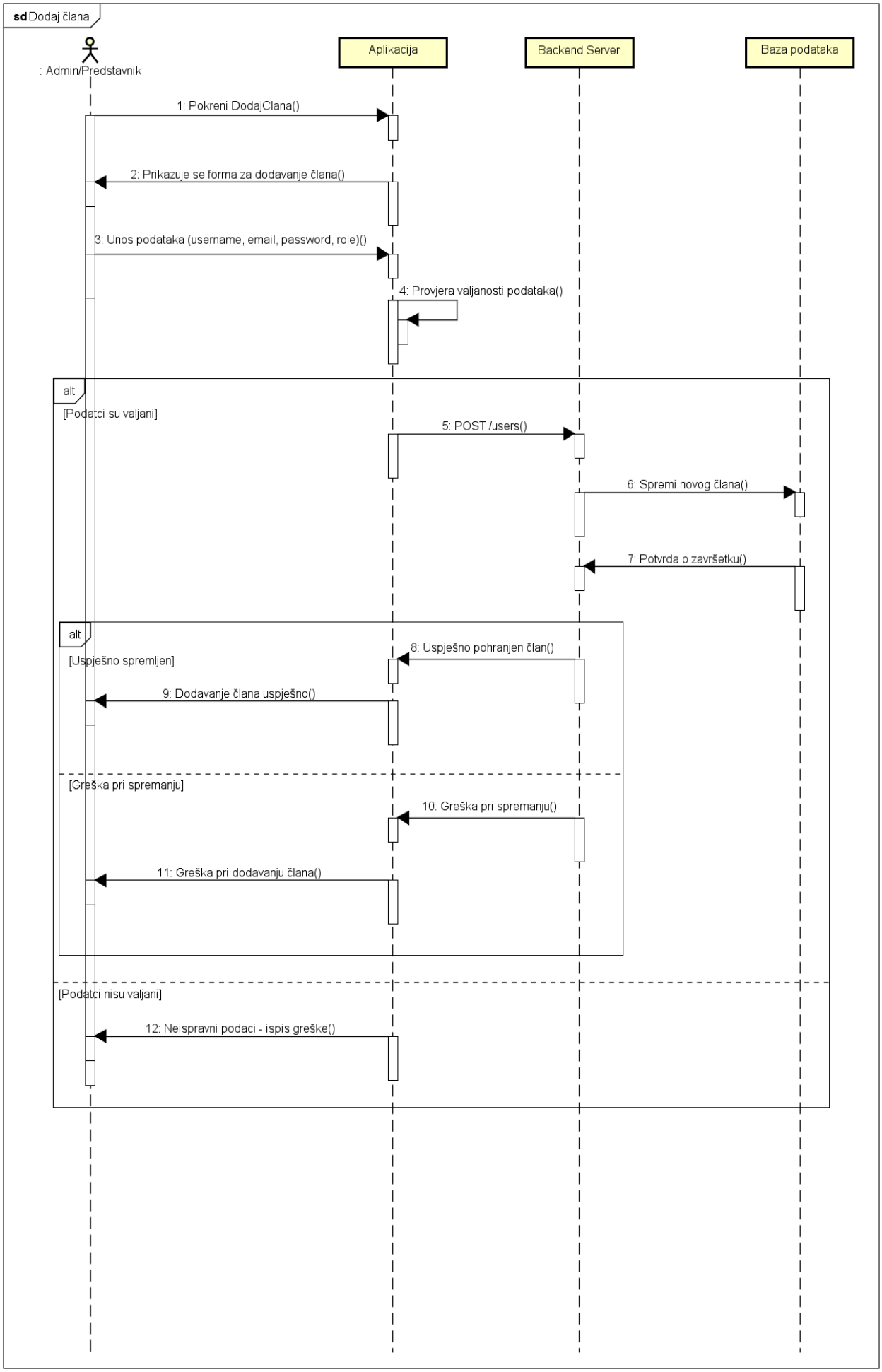


Slika 3.4: Sekvencijski dijagram prikazuje proces prijave korisnika. Korisnik može odabrati prijavu putem vanjskog servisa (kao što je OAuth 2.0(Google)) ili putem klasične prijave s e-mail adresom i lozinkom. Ako korisnik odabere vanjski servis za prijavu, aplikacija ga preusmjerava na stranicu za prijavu gdje se vrši prijava putem Google računa. U slučaju uspješne prijave, aplikacija dopušta korisniku pristup. U slučaju da

korisnik odabere klasičnu prijavu, unosi svoj e-mail i lozinku koje aplikacija provjerava u bazi podataka. Ako su podaci ispravni, korisnik dobiva pristup; inače, sustav prikazuje poruku o pogrešci.



Slika 3.5: Sekvencijski dijagram prikazuje postupak kreiranja i upravljanja sastankom. Proces započinje kada predstavnik stanara odabere opciju za stvaranje novog sastanka. Unose se osnovne informacije o sastanku, poput mjesta, vremena i točaka dnevnog reda. Nakon toga, aplikacija postavlja sastanak u status "Objavljen" i šalje obavijesti suvlasnicima. Suvlasnici zatim mogu označiti hoće li sudjelovati na sastanku, što se bilježi u sustavu. Nakon održavanja sastanka, predstavnik može promijeniti status sastanka na "Obavljen" i unijeti zaključke za svaku točku dnevnog reda. Na kraju, sastanak prelazi u status "Arhiviran", čime se zaključava njegov sadržaj i šalje se završna obavijest suvlasnicima. Sastanak ne može biti arhiviran dok sve točke sa pravnim učinkom nemaju zaključak.



Slika 3.6: Sekvencijski dijagram prikazuje proces dodavanja novog člana. Proces započinje kada korisnik pokrene funkciju dodavanja člana. Nakon što je administrator započeo proces dodavanje člana, aplikacija prikazuje formu za unos podataka (korisničko ime, email, lozinka, uloga). Slijedi provjera valjanosti

unesenih podataka; ako su podaci ispravni, šalju se serveru putem POST zahtjeva, gdje se pohranjuju u bazu podataka. U slučaju uspješne pohrane, korisniku se šalje potvrda o uspjehu. Ako dođe do greške, aplikacija vraća poruku o neuspjehu, ispisujući pogrešku u pohranjivanju ili validaciji podataka.

4. Arhitektura i dizajn sustava

Arhitektura sustava

Opis arhitekture

Stil arhitekture

Odabrali smo monolitni arhitektonski stil s pristupom klijent-poslužitelj. S obzirom na to da se radi o relativno malom timu i jednostavnom aplikaciji, ovim pristupom možemo lako podijeliti posao i testirati funkcionalnosti jer svi dijelovi sustava rade zajedno u jednoj cjelini.

Podsustavi

- Baza podataka služi za sigurnu potrebnu svih podataka potrebnih za ispravan rad aplikacije.
- Web poslužitelj je softver povezan na klijent i bazu podataka koji ovisno o HTTP zahtjevima koje primi uzima ili stavlja podatke u bazu podataka i klijentu šalje informacije potrebne za prikladan prikaz web stranice.
- Web preglednik je program koji omogućava korisniku pregled web-stranica. Putem njega korisnik može slati zahtjeve web poslužitelju na osnovu kojih poslužitelj šalje podatke klijentu koji ih prikazuje korisniku.

Spremišta podataka

Podatci su pohranjeni u SQLite relacijskoj bazi podataka koju smo odabrali zbog jednostavnosti korištenja i postojeće programske podrške za tu bazu dostupnu u Visual Studio Code-u i .NET bibliotekama.

Mrežni protokoli

Za komunikaciju između frontend i backend sustava koristi se HTTP protokol. Korisnici tijekom uporabe aplikacije koriste klijentsku stranu aplikacije koja šalje zahtjeve poslužiteljskoj strani aplikacije. Poslužiteljska strana obrađuje te zahtjeve i izmjenjuje podatke s bazom podataka. Nakon završene obrade, poslužiteljska strana šalje potrebne podatke klijentskoj koja ih prikazuje korisniku.

Sklopovsko-programski zahtjevi

Za implementaciju aplikacije smo koristili programski jezik visoke razine pa nemamo specifične zahtjeve za sklopovlje. Kao programsku podršku smo koristili biblioteke za komunikaciju s bazom podataka, autorizaciju, izgradnju korisničkog sučelja i komunikaciju između frontenda i backenda.

Organizacija sustava na visokoj razini

- Klijent-poslužitelj: Klijent (korisnici sustava poput suvlasnika, predstavnika suvlasnika i administratora) povezuje se s poslužiteljem putem interneta. Klijent može pristupiti aplikaciji koristeći web preglednik koja se povezuje s glavnim poslužiteljem koji upravlja prijavom, obradom podataka i planiranjem sastanaka. Poslužitelj koristi OAuth 2.0 protokol za sigurnu autentifikaciju korisnika te

Login pomoću Gmail računa, čime se osigurava kontrolirani i siguran pristup aplikaciji. Poslužitelj posreduje između korisnika i baze podataka, pružajući siguran pristup informacijama o sastancima i korisnicima.

- Baza podataka: Sustav koristi relacijsku bazu podataka za pohranu strukturiranih podataka, uključujući podatke o korisnicima, sastancima, dnevnom redu sastanaka, zaključcima i statusima prisutnosti korisnika. Relacijska baza omogućava jednostavno upravljanje složenim povezanim podacima putem tablica i relacija, što olakšava organizaciju i dohvat podataka. Baza je osmišljena tako da podržava integritet i konzistentnost podataka potrebnih za funkcionalnost sustava.

  Meeting - identifikator sastanka, naslov, zaključci, status, vrijeme i mjesto

  AgendaPoints – identifikator točke dnevnog reda, sažetak, ima li pravni učinak, ishod ili zaključak točke dnevnog reda i identifikator sastanka kojem točka dnevnog reda pripada

  User – identifikator korisnika, korisničko ime, e-mail, zaporka i uloga korisnika

  UserMeeting – identifikator korisnika i identifikator sastanka

- Datotečni sustav: nije implementiran u aplikaciji

- Grafičko sučelje: Korisničko sučelje Apartmeet aplikacije je dizajnirano kao web aplikacija s intuitivnim i preglednim dizajnom. Sučelje pruža jednostavan uvid u točke dnevnog reda sastanka, status sastanaka, mogućnost pregledavanja zaključaka te označavanje prisutnosti na sastancima. Ova web aplikacija komunicira s poslužiteljem putem HTTP zahtjeva, koji obrađuju korisničke akcije i dohvaćaju relevantne podatke iz baze.

Baza podataka

Koristili smo se relacijskom bazom podataka SQLite. Ukupno imamo 4 entiteta: AgendaPoint, Meeting, UserMeeting i User

Opis tablica

AgendaPoint

Pojedinačna točka dnevnog reda na sastanku, koja može imati pravni učinak ili biti informativna. Nakon sastanka, svaka točka može dobiti zaključak koji se označava kao „Izglasan“ ili „Odbijen“. Povezana je s entitetom Meeting, jer svaka točka pripada određenom sastanku.

Atribut	Tip	Opis
Id	int	Jedinstveni identifikator točke dnevnog reda.
Description	string	Tekstualni opis točke dnevnog reda.
HasLegalEffect	bool	Pokazuje ima li točka pravni učinak.
Outcome	string	Rezultat ili zaključak točke dnevnog reda.
MeetingId	int	Referenca na povezani sastanak.

Meeting

Predstavlja sastanak organiziran od strane predstavnika suvlasnika, s detaljima poput naslova, sažetka, statusa, vremena i mjesta održavanja. Povezan je s entitetom AgendaPoint, gdje svaki sastanak može imati više točaka dnevnog reda, i s entitetom UserMeeting, što omogućava praćenje sudjelovanja korisnika na sastancima.

Atribut	Tip	Opis
Id	int	Jedinstveni identifikator sastanka.
Title	string	Naslov sastanka.
Summary	string	Kratki sažetak sastanka.
Status	string	Trenutni status sastanka.
ScheduledDate	DateTime	Datum i vrijeme održavanja sastanka.
Place	string	Lokacija na kojoj se sastanak održava.

UserMeeting

Ovaj entitet povezuje korisnike sa sastancima, označavajući njihovo sudjelovanje. Na taj način omogućuje se praćenje koji korisnici sudjeluju na kojem sastanku. Povezan je s entitetima User i Meeting, što omogućava evidentiranje prisutnosti korisnika na sastancima.

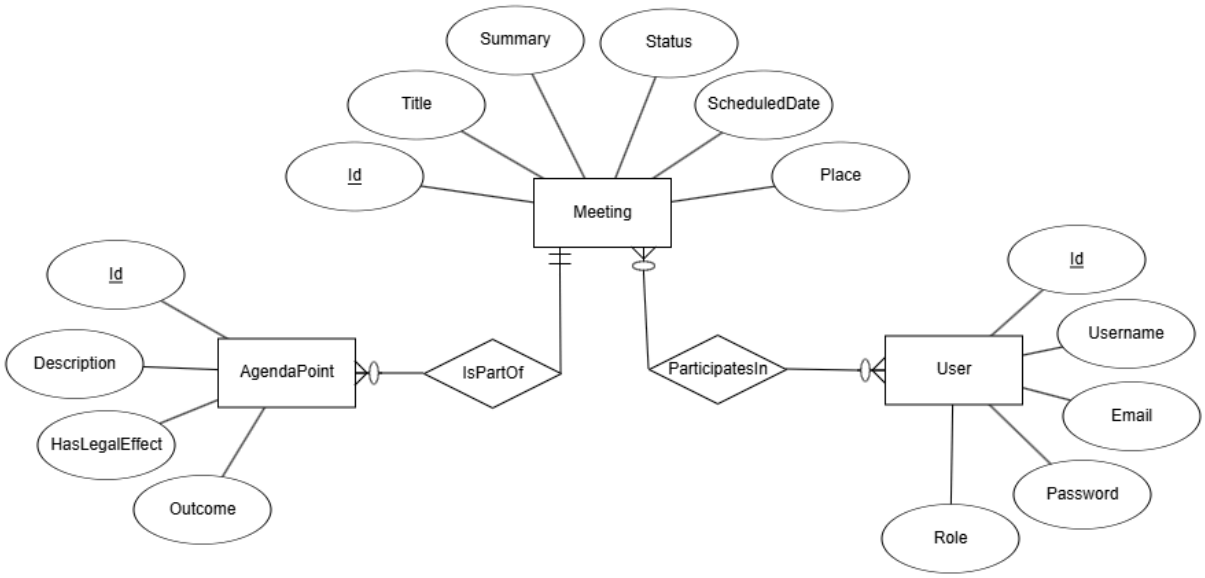
Atribut	Tip	Opis
UserId	int	Identifikator korisnika.
MeetingId	int	Identifikator sastanka.

User

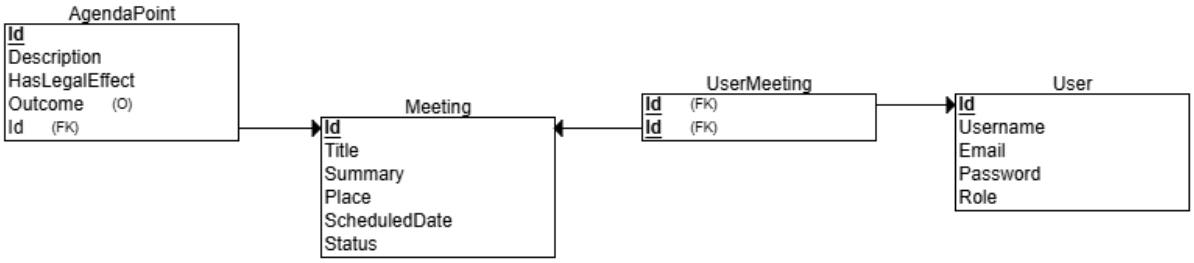
Predstavlja korisnika sustava, uključujući administratore, predstavnike suvlasnika i same suvlasnike. Korisnici imaju mogućnost pregleda sastanaka, označavanja prisutnosti te upravljanja inicijalnim postavkama računa.

Atribut	Tip	Opis
Id	int	Jedinstveni identifikator korisnika.
Username	string	Korisničko ime za prijavu.
Email	string	Email adresa korisnika.
Password	string	Lozinka korisnika.
Role	string	Uloga korisnika (npr. administrator, suvlasnik).

Dijagram baze podataka

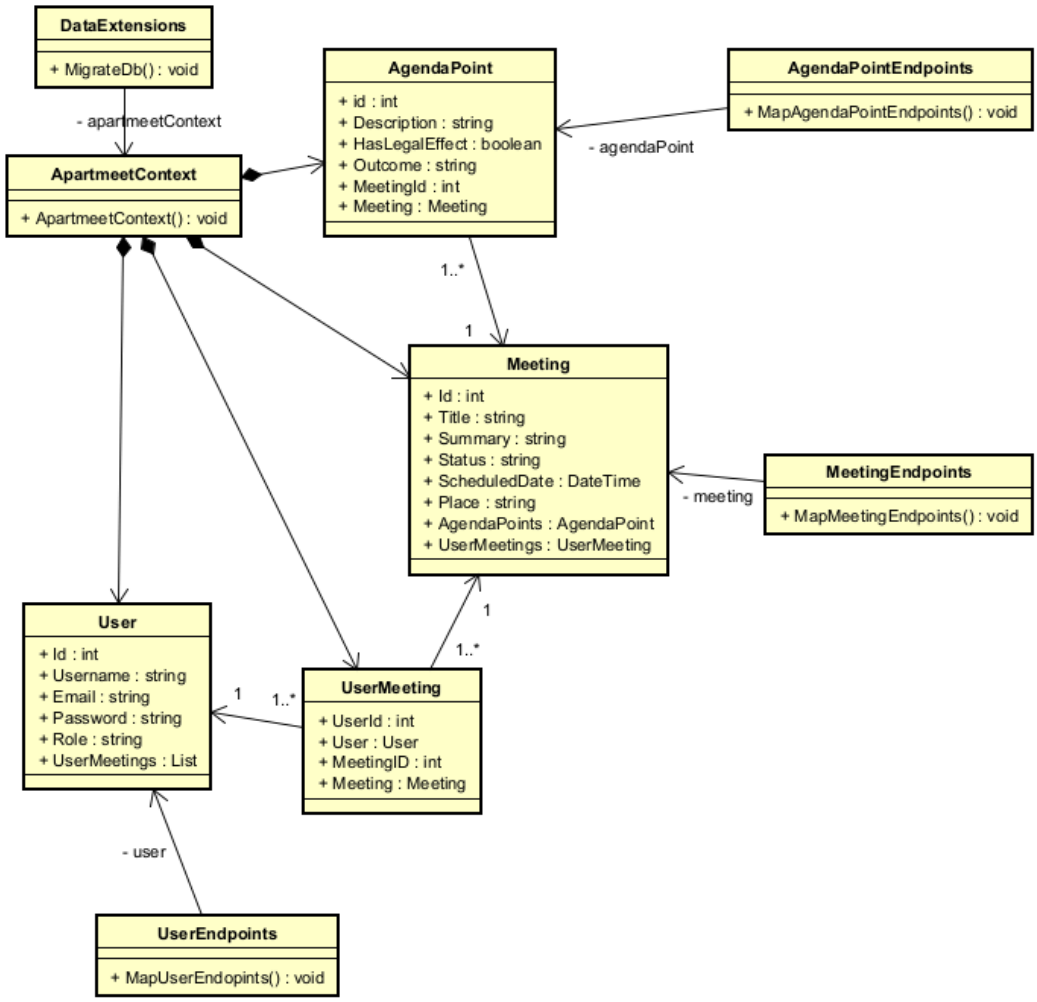


Slika 4.1 ER dijagram baze podataka



Slika 4.2 Relacijska shema baze podataka

Dijagram razreda



Slika 4.3 Dijagram razreda

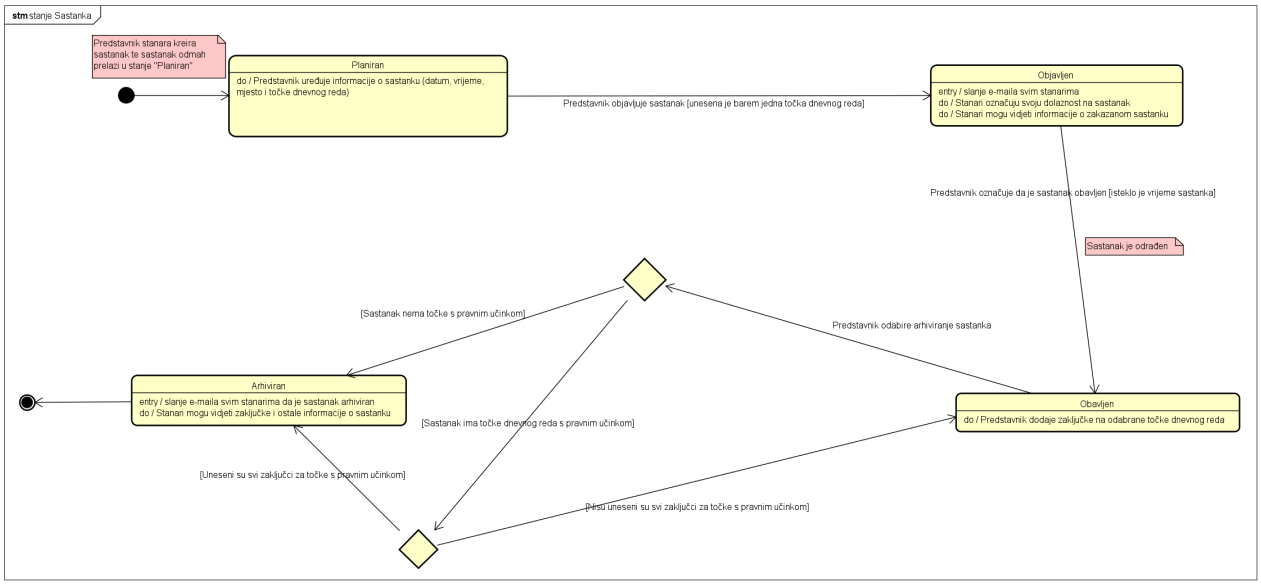
Klasni dijagram prikazuje strukturu aplikacije za upravljanje sastancima. Klasa **ApartmeetContext** predstavlja klasu koja povezuje bazu podataka s drugim dijelovima aplikacije, omogućavajući rad s podacima o sastancima, korisnicima, točkama dnevnog reda i korisničkim ulogama. **DataExtensions** klasa upravlja prijenosima podataka baze podataka i nudi metode za ažuriranje ili inicijalizaciju podataka.

Klase **Meeting**, **AgendaPoint**, **User** i **UserMeeting** definiraju osnovne entitete aplikacije. **Meeting** klasa sadrži informacije o sastancima, uključujući naziv, opis, status i datum održavanja, dok **AgendaPoint** predstavlja pojedinačne točke dnevnog reda unutar svakog sastanka, što je označeno odnosom 1..* prema **Meeting** klasi (jedan sastanak može imati više točaka). **UserMeeting** služi kao posrednička klasa između korisnika (**User**) i sastanaka (**Meeting**), što omogućava da jedan korisnik može biti povezan s više sastanaka, a također više korisnika može sudjelovati u jednom sastanku.

Endpoint klase, poput **UserEndpoints**, **MeetingEndpoints** i **AgendaPointEndpoints**, služe za upravljanje specifičnim funkcijama poput mapiranja krajnjih točaka (endpoints) za rad s podacima o korisnicima, sastancima i točkama dnevnog reda, omogućujući lakši pristup i manipulaciju podacima putem API-ja.

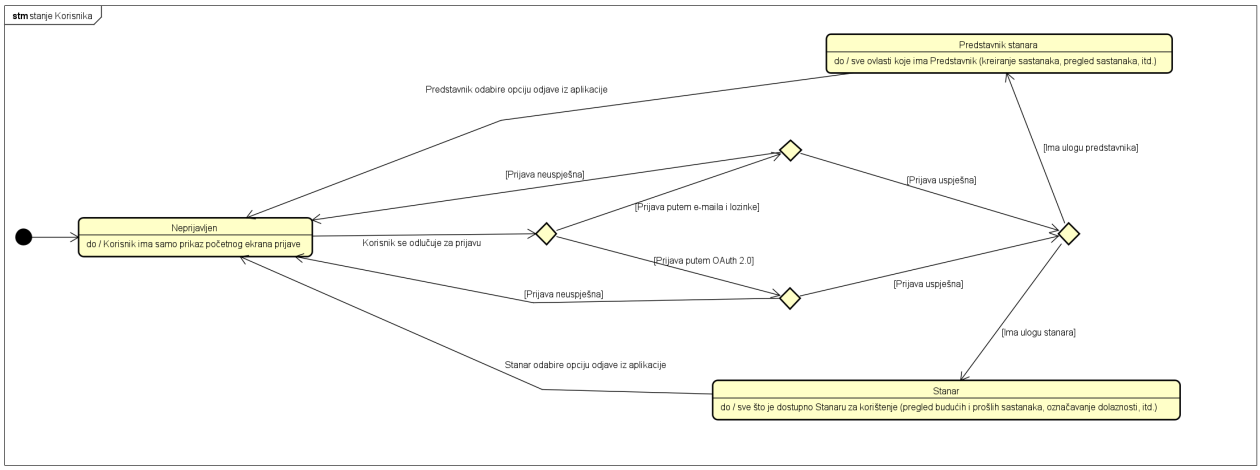
Dinamičko ponašanje aplikacija

UML Dijagrami stanja



Slika 4.4 Dijagram stanja sastanka

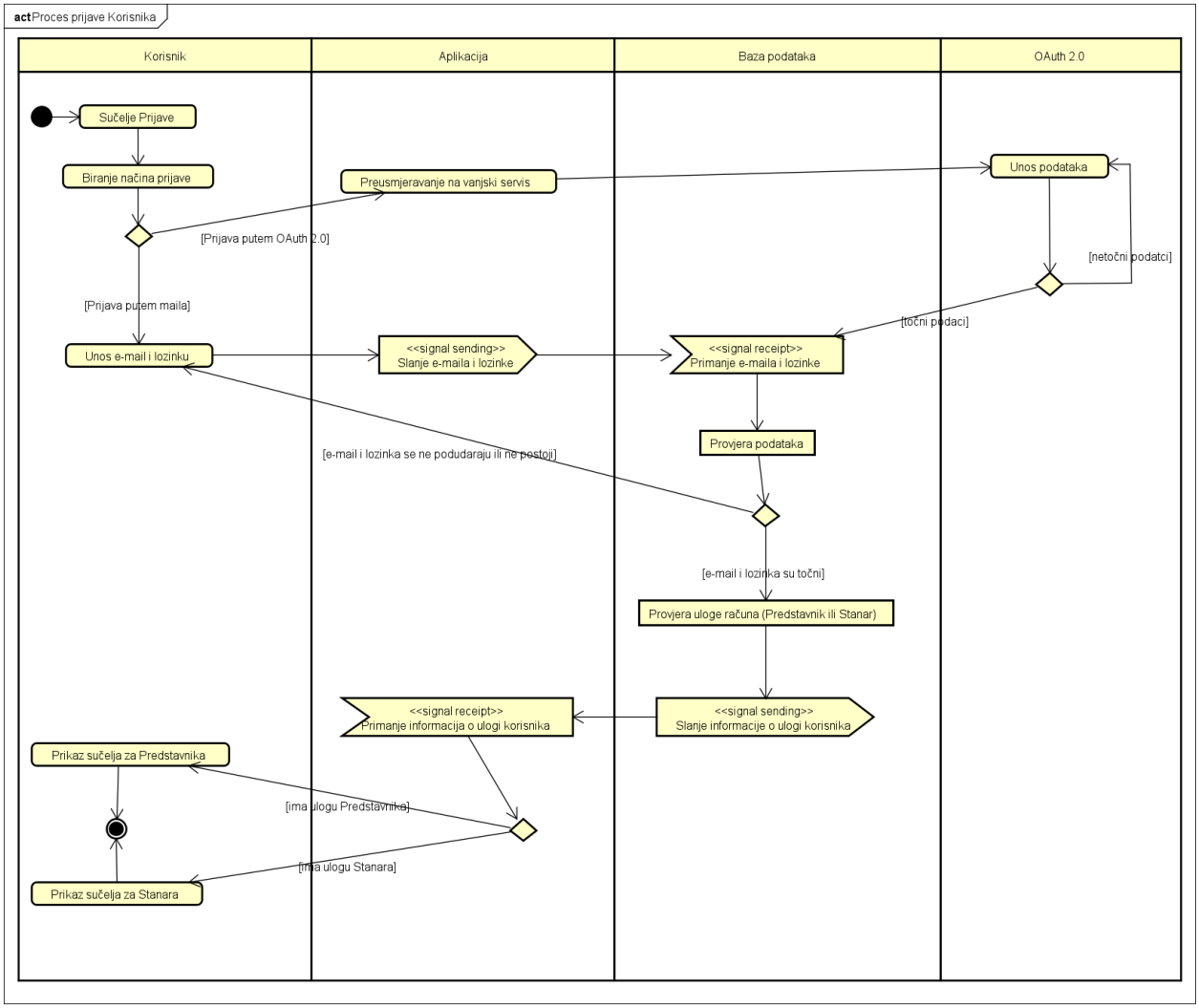
Dijagram stanja prikazuje tijek stanja kroz koja prolazi sastanak, počevši od kreiranja pa sve do arhiviranja. Sastanak započinje u stanju **Planiran**, gdje predstavnik stanara uređuje osnovne informacije poput datuma, vremena i točaka dnevnog reda. Kada se sastanak objavi, prelazi u stanje **Objavljen**, gdje postaje vidljiv svim stanarima, a sustav automatski šalje e-mail obavijesti svim stanarima. Nakon što sastanak završi, predstavnik ga označava kao **Obavljen**, čime mu se omogućuje dodavanje zaključaka za pojedine točke dnevnog reda. Ako sastanak **nema točke s pravnim učinkom**, može se odmah arhivirati, dok se u suprotnom mora osigurati da su svi zaključci za točke s pravnim učinkom uneseni prije nego što sastanak može preći u stanje **Arhiviran**. Tijekom arhiviranja, sustav ponovno šalje obavijesti svim stanarima i omogućuje pregled zaključaka.



Slika 4.5 Dijagram stanja korisnika

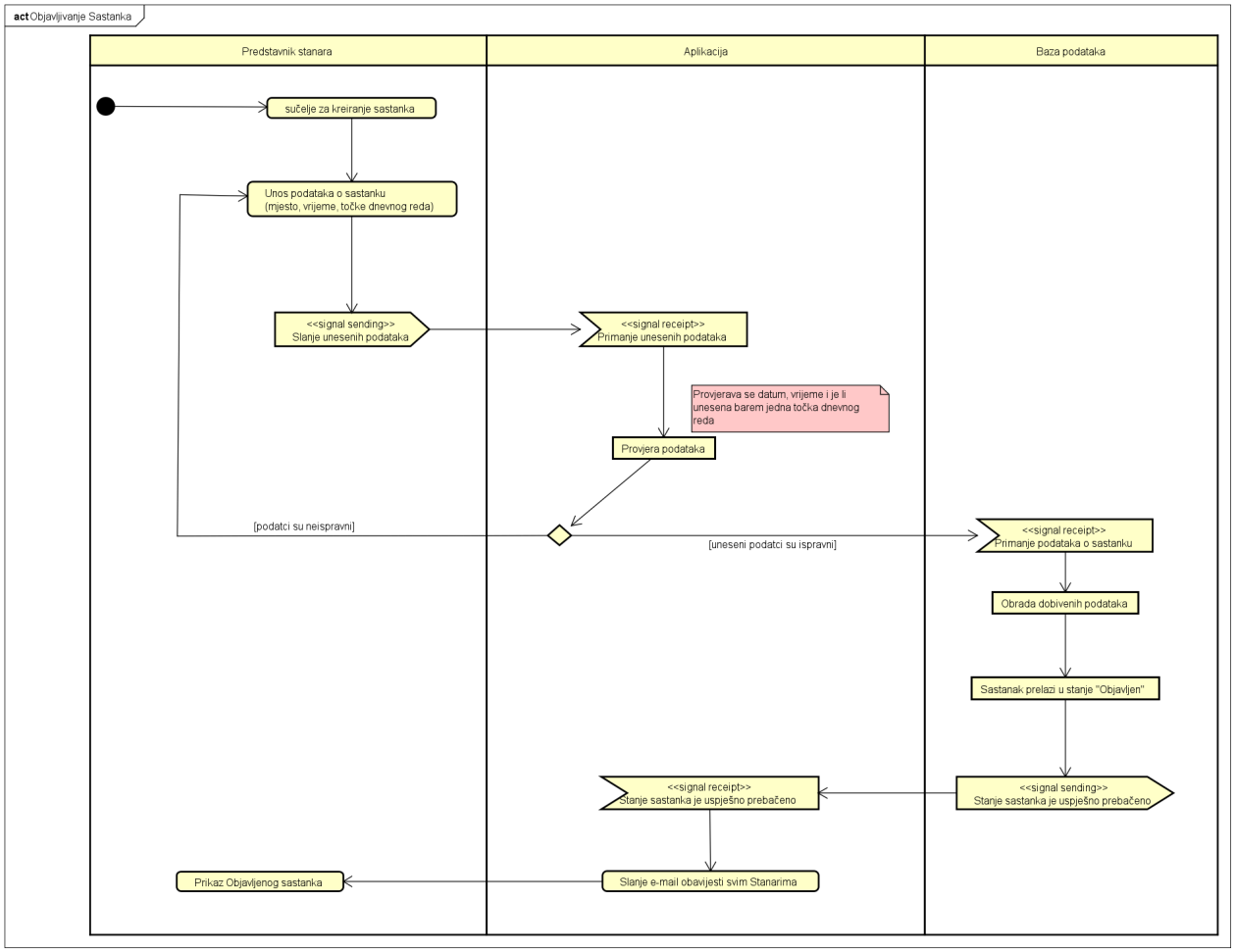
Dijagram stanja prikazuje promjene stanja korisnika, počevši od početnog stanja Neprijavljen, gdje korisnik može vidjeti samo ekran prijave. Korisnik se tada odlučuje za prijavu te bira između klasične autentifikacije putem e-maila i lozinke ili OAuth 2.0 autentifikacije. Ako je prijava neuspješna, korisnik se vraća u stanje Neprijavljen, dok uspješna prijava vodi u izbor uloge korisnika. Ako korisnik ima ulogu Predstavnika stanara, dobiva ovlasti Predstavnika, a ako nema dobiva ulogu Stanara. U bilo kojem trenutku korisnik može odabrati odjavu, čime se vraća u početno stanje Neprijavljen.

UML dijagrami aktivnosti



Slika 4.6 Dijagram Aktinosti Prijave Korisnika

Ovaj dijagram prikazuje postupak autentifikacije korisnika u aplikaciji. Korisnik započinje proces prijave odabirom između OAuth 2.0 ili klasične prijave putem e-maila i lozinke. Ako odabere OAuth 2.0, preusmjerava se na vanjski autentifikacijski servis, gdje se provjerava valjanost podataka. U slučaju klasične prijave, korisnik unosi svoje podatke, a aplikacija ih šalje bazi podataka na provjeru. Ako su podaci točni, baza podataka određuje korisnikovu ulogu (stanar ili predstavnik stanara) i vraća tu informaciju aplikaciji. Nakon uspješne autentifikacije, korisnik dobiva pristup sučelju aplikacije prema svojoj dodijeljenoj ulozi. Ako podaci nisu ispravni, korisnik dobiva poruku o pogrešci i može pokušati ponovno.

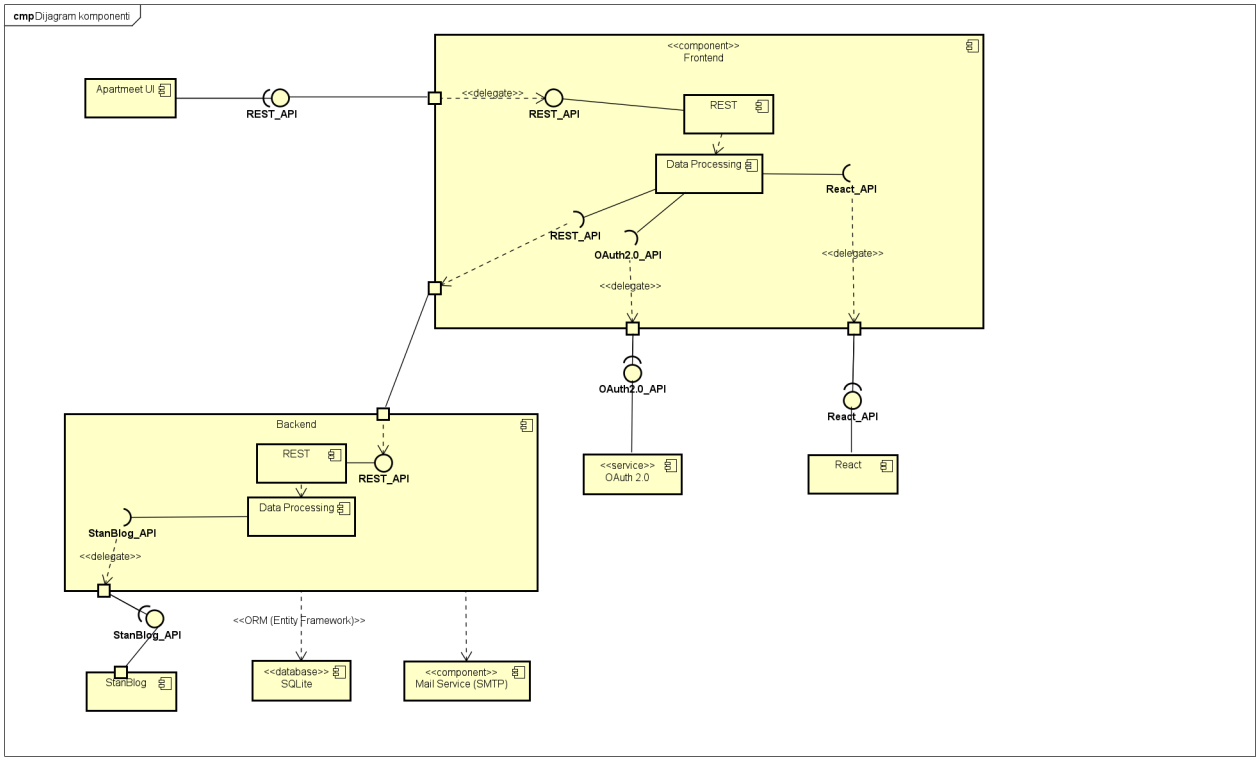


Slika 4.7 Dijagram Aktinosti Objavljivanja sastanka

Dijagram prikazuje proces kreiranja i objavljivanja sastanka u aplikaciji. Predstavnik stanara započinje unosom podataka o sastanku (datuma, vrijeme i točke dnevnog reda). Nakon što unese potrebne podatke, aplikacija ih šalje bazi podataka na provjeru. Ako neki podaci nedostaju, aplikacija vraća poruku o grešci i korisnik može ispraviti unos. Kada su podaci valjani, baza ih obrađuje i sastanak prelazi u stanje "Objavljen". Nakon toga, aplikacija generira e-mail obavijesti koje se šalju svim stanarima. Proces završava prikazivanjem objavljenog sastanka u aplikaciji, gdje korisnici mogu vidjeti detalje i potvrditi dolazak.

5. Arhitektura komponenata i razmještaja

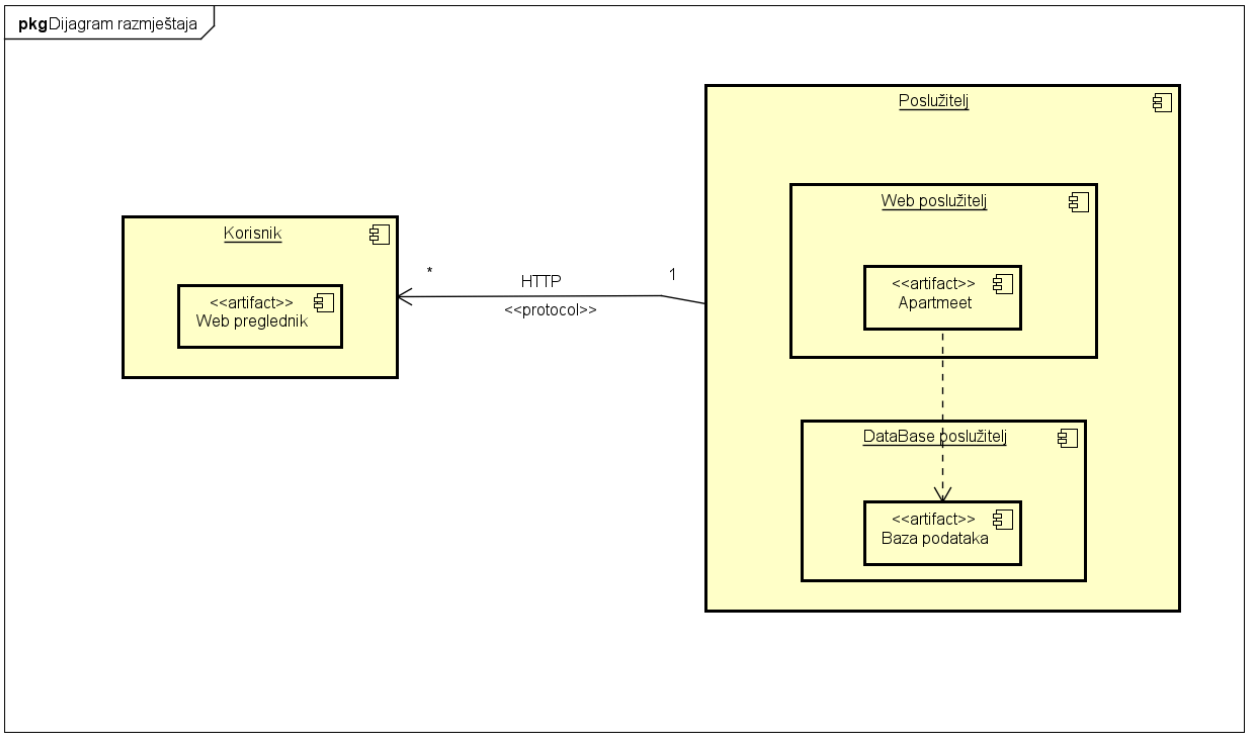
Dijagram komponenata



Slika 5.1 Dijgram komponenata sustava

Dijagram komponenti prikazuje podjelu sustava Apartmeet na frontend i backend, gdje frontend koristi React za korisničko sučelje i komunicira s backendom putem REST API-ja. Backend obrađuje podatke, pristupa SQLite bazi podataka preko Entity Framework ORM-a te koristi OAuth 2.0 za autentifikaciju. Mail Service (SMTP) omogućava slanje e-mailova, a dodatno postoji i StanBlog API povezan s StanBlog. Veze između komponenti su jasno definirane kroz API-jeve i ORM pristupe.

Dijagram razmještaja



Slika 5.2 Dijgram razmještaja sustava

Dijagram razmještaja prikazuje kako korisnici pristupaju aplikaciji putem web preglednika koristeći HTTP protokol. Web poslužitelj pokreće aplikaciju Apartmeet i komunicira s Database poslužiteljem, gdje je smještena baza podataka. Dijagram jasno odvaja klijentsku i serversku stranu te prikazuje kako korisnici istovremeno mogu koristiti sustav.

6. Ispitivanje programskog rješenja

Ispitivanje komponenti

1. Ispitivanje - Brisanje člana

1. Ulazni podaci:

- username: ivan ivanic

---

2. Koraci testiranja:

- Kliknuti na izbornik (tri crtice u gornjem lijevom kutu)
- Kliknuti "Obriši člana"
- Unijeti korisničko ime člana
- Kliknuti "Obriši"

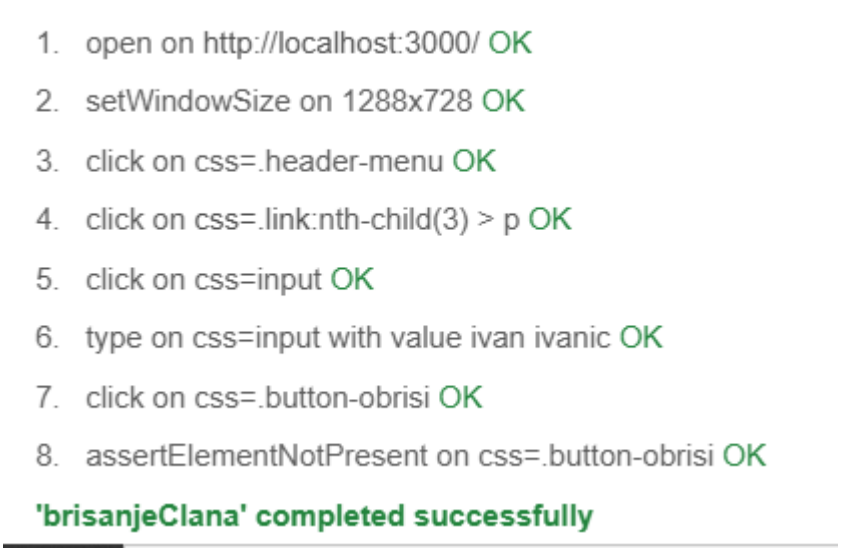
---

3. Očekivani izlaz

- Usmjerenje na početnu stranicu aplikacije
- Gumb "Obriši" više nije prisutan

---

4. Dobiveni izlaz



Slika 6.1 Dobiveni izlaz za Testiranje brisanja člana

---

5. Izvorni kod ispitnog slučaja

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Threading;
using OpenQA.Selenium;
using OpenQA.Selenium.Chrome;
using OpenQA.Selenium.Firefox;
using OpenQA.Selenium.Remote;
using OpenQA.Selenium.Support.UI;
using OpenQA.Selenium.Interactions;
using NUnit.Framework;
[TestFixture]
0 references
public class BrisanjeClanaTest {
  11 references
  private IWebDriver driver;
  1 reference
  public IDictionary<string, object> vars {get; private set;}
  1 reference
  private IJavaScriptExecutor js;
  [SetUp]
  1 reference
  public void SetUp() {
    driver = new ChromeDriver();
    js = (IJavaScriptExecutor)driver;
    vars = new Dictionary<string, object>();
  }
  [TearDown]
  1 reference
  protected void TearDown() {
    driver.Quit();
  }
  [Test]
  0 references
  public void brisanjeClana() {
    driver.Navigate().GoToUrl("https://apartmeet.onrender.com/");
    driver.Manage().Window.Size = new System.Drawing.Size(1288, 728);
    driver.FindElement(By.CssSelector(".header-menu")).Click();
    driver.FindElement(By.CssSelector(".link:nth-child(3) > p")).Click();
    driver.FindElement(By.CssSelector("input")).Click();
    driver.FindElement(By.CssSelector("input")).SendKeys("ivan ivanic");
    driver.FindElement(By.CssSelector(".button-obrisi")).Click();
    {
      var elements = driver.FindElements(By.CssSelector(".button-obrisi"));
      Assert.True(elements.Count == 0);
    }
  }
}
```

Slika 6.2 Kod ispitnog slučaja Testiranje brisanja člana

Test brisanja člana provjerava mogućnost uklanjanja korisnika iz sustava. Skripta u Seleniumu otvara aplikaciju, navigira do odgovarajuće stranice, unosi ime člana "ivan ivanic" u polje za pretragu i klikne na gumb za brisanje. Na kraju se provjerava da gumb više nije prisutan na stranici, što potvrđuje uspješno brisanje. Važno je napomenuti da se član mora prethodno nalaziti u bazi kako bi test bio valjan.

2. Ispitivanje - Dodavanje člana

1. Ulazni podaci:

- username: marko markec
- email: markomarkec@test.com
- password: 123

---

2. Koraci testiranja:

- Kliknuti na izbornik (tri crtice u gornjem lijevom kutu)
- Kliknuti "Dodaj člana"
- Unijeti korisničko ime člana
- Unijeti email člana
- Unijeti lozinku člana<
- Unijeti ulogu člana
- Kliknuti "Dodaj člana"

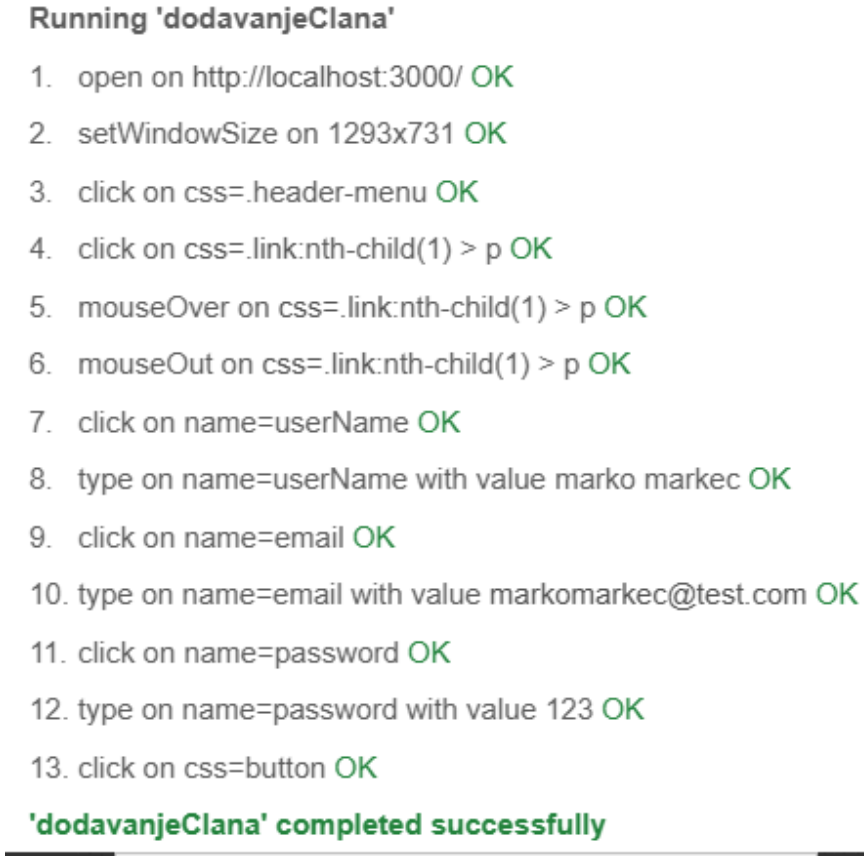
---

3. Očekivani izlaz

- Usmjerenje na početnu stranicu aplikacije

---

4. Dobiveni izlaz



Slika 6.3 Dobiveni izlaz za Testiranje dodavanje člana

5. Izvorni kod ispitnog slučaja

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Threading;
using OpenQA.Selenium;
using OpenQA.Selenium.Chrome;
using OpenQA.Selenium.Firefox;
using OpenQA.Selenium.Remote;
using OpenQA.Selenium.Support.UI;
using OpenQA.Selenium.Interactions;
using NUnit.Framework;
[TestFixture]
0 references
public class DodavanjeClanaTest {
  18 references
  private IWebDriver driver;
  1 reference
  public IDictionary<string, object> vars {get; private set;}
  1 reference
  private IJavaScriptExecutor js;
  [SetUp]
  1 reference
  public void SetUp() {
    driver = new ChromeDriver();
    js = (IJavaScriptExecutor)driver;
    vars = new Dictionary<string, object>();
  }
  [TearDown]
  1 reference
  protected void TearDown() {
    driver.Quit();
  }
  [Test]
  0 references
  public void dodavanjeClana() {
    driver.Navigate().GoToUrl("https://apartmeet.onrender.com/");
    driver.Manage().Window.Size = new System.Drawing.Size(1293, 731);
    driver.FindElement(By.CssSelector(".header-menu")).Click();
    driver.FindElement(By.CssSelector(".link:nth-child(1) > p")).Click();
    {
      var element = driver.FindElement(By.CssSelector(".link:nth-child(1) > p"));
      Actions builder = new Actions(driver);
      builder.MoveToElement(element).Perform();
    }
    {
      var element = driver.FindElement(By.tagName("body"));
      Actions builder = new Actions(driver);
      builder.MoveToElement(element, 0, 0).Perform();
    }
    driver.FindElement(By.Name("userName")).Click();
    driver.FindElement(By.Name("userName")).SendKeys("marko markec");
    driver.FindElement(By.Name("email")).Click();
    driver.FindElement(By.Name("email")).SendKeys("markomarkec@test.com");
    driver.FindElement(By.Name("password")).Click();
    driver.FindElement(By.Name("password")).SendKeys("123");
    driver.FindElement(By.CssSelector("button")).Click();
  }
}
```

Slika 6.4 Kod ispitnog slučaja Testiranje dodavanje člana

Test dodavanja člana uspješno je izvršen i pokazuje da se novi korisnik može dodati u sustav. Test otvara aplikaciju, navigira do stranice za dodavanje korisnika, unosi ime, e-mail i lozinku, te potvrđuje unos. Očekivani rezultat je da se korisnik pravilno pohrani u sustav i bude dostupan za daljnje operacije.

3. Ispitivanje - Dodavanje člana bez username-a

1. Ulazni podaci:

- email: markomarkec@test.com
- password: 123

2. Koraci testiranja:

- Kliknuti na izbornik (tri crtice u gornjem lijevom kutu)
- Kliknuti "Dodaj člana"
- Unijeti email člana
- Unijeti lozinku člana
- Namjerno izostaviti unos korisničkog imena
- Pokušati stisnuti "Dodaj člana"

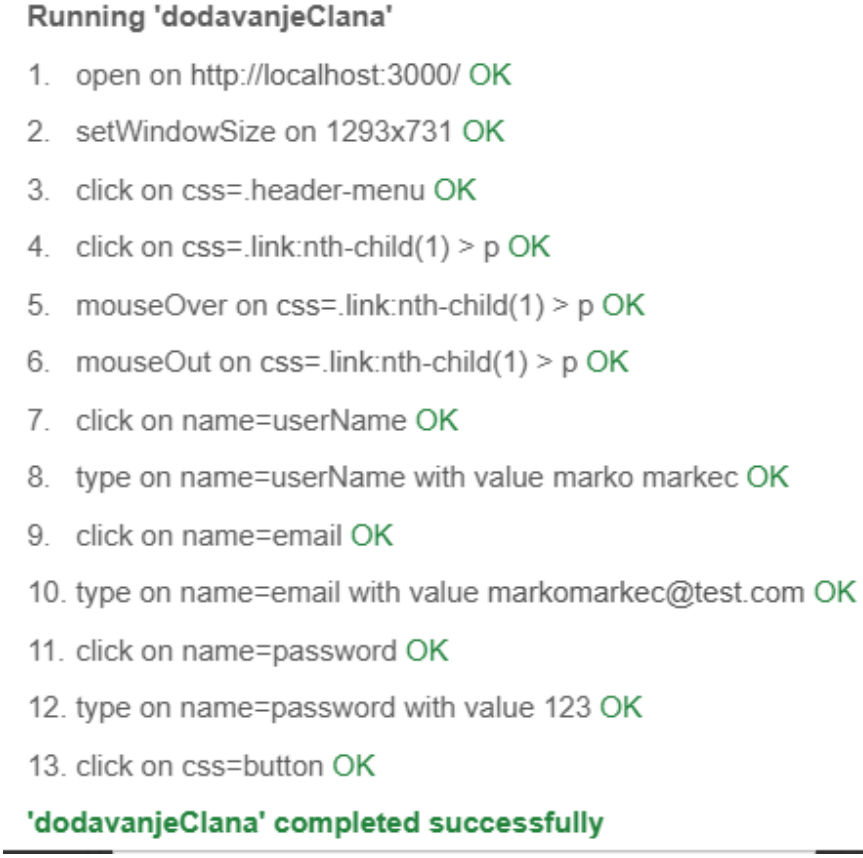
---

3. Očekivani izlaz

- Zaslon ostaje nepromjenjen (gumb "Dodaj člana" nije dostupan)

---

4. Dobiveni izlaz



Slika 6.5 Dobiveni izlaz za Testiranje dodavanja člana

---

5. Izvorni kod ispitnog slučaja

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Threading;
using OpenQA.Selenium;
using OpenQA.Selenium.Chrome;
using OpenQA.Selenium.Firefox;
using OpenQA.Selenium.Remote;
using OpenQA.Selenium.Support.UI;
using OpenQA.Selenium.Interactions;
using NUnit.Framework;
[TestFixture]
0 references
public class DodavanjeClanaTest {
  18 references
  private IWebDriver driver;
  1 reference
  public IDictionary<string, object> vars {get; private set;}
  1 reference
  private IJavaScriptExecutor js;
  [SetUp]
  1 reference
  public void SetUp() {
    driver = new ChromeDriver();
    js = (IJavaScriptExecutor)driver;
    vars = new Dictionary<string, object>();
  }
  [TearDown]
  1 reference
  protected void TearDown() {
    driver.Quit();
  }
  [Test]
  0 references
  public void dodavanjeClana() {
    driver.Navigate().GoToUrl("https://apartmeet.onrender.com/");
    driver.Manage().Window.Size = new System.Drawing.Size(1293, 731);
    driver.FindElement(By.CssSelector(".header-menu")).Click();
    driver.FindElement(By.CssSelector(".link:nth-child(1) > p")).Click();
    {
      var element = driver.FindElement(By.CssSelector(".link:nth-child(1) > p"));
      Actions builder = new Actions(driver);
      builder.MoveToElement(element).Perform();
    }
    {
      var element = driver.FindElement(By.tagName("body"));
      Actions builder = new Actions(driver);
      builder.MoveToElement(element, 0, 0).Perform();
    }
    driver.FindElement(By.Name("userName")).Click();
    driver.FindElement(By.Name("userName")).SendKeys("marko markec");
    driver.FindElement(By.Name("email")).Click();
    driver.FindElement(By.Name("email")).SendKeys("markomarkec@test.com");
    driver.FindElement(By.Name("password")).Click();
    driver.FindElement(By.Name("password")).SendKeys("123");
    driver.FindElement(By.CssSelector("button")).Click();
  }
}

```

Slika 6.6 Kod ispitnog slučaja Testiranje dodavanja člana

Test dodavanja člana bez korisničkog imena provjerava kako sustav reagira kada se pokušava unijeti korisnik kojem nedostaju određeni podatci. Skripta otvara aplikaciju, navigira do stranice za dodavanje korisnika, unosi samo e-mail i lozinku, te pokušava potvrditi unos. Očekivani rezultat je da se član ne može dodati , što test uspješno potvrđuje.

4. Ispitivanje - Dodavanje člana bez lozinke

1. Ulazni podaci:

- email: ivan@test.com
- username: ivan

---

2. Koraci testiranja:

- Kliknuti na izbornik (tri crtice u gornjem lijevom kutu)
- Kliknuti "Dodaj člana"
- Unijeti email člana
- Unijeti username člana
- Namjerno izostaviti unos lozinke
- Pokušati stisnuti "Dodaj člana"

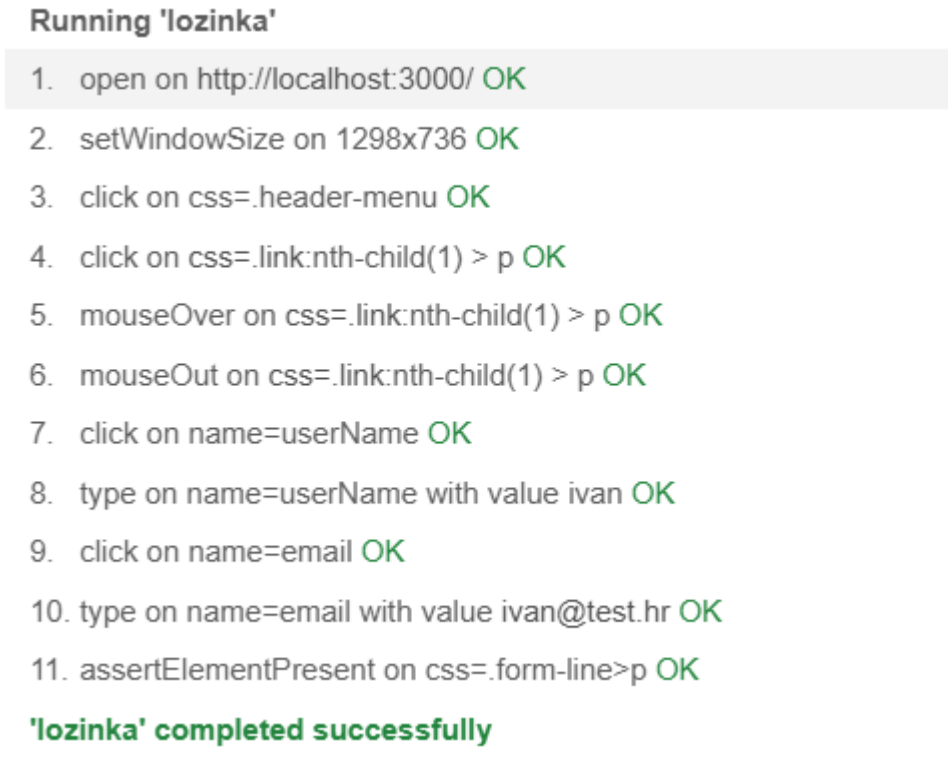
---

3. Očekivani izlaz

- Zaslon ostaje nepromjenjen (gumb "Dodaj člana" nije dostupan)

---

4. Dobiveni izlaz



Slika 6.7 Dobiveni izlaz za Testiranje dodavanja člana

---

5. Izvorni kod ispitnog slučaja

```
using System.Collections.Generic;
using System.Linq;
using System.Threading;
using OpenQA.Selenium;
using OpenQA.Selenium.Chrome;
using OpenQA.Selenium.Firefox;
using OpenQA.Selenium.Remote;
using OpenQA.Selenium.Support.UI;
using OpenQA.Selenium.Interactions;
using NUnit.Framework;
[TestFixture]
0 references
public class LozinkaTest {
  16 references
  private IWebDriver driver;
  1 reference
  public IDictionary<string, object> vars {get; private set;}
  1 reference
  private IJavaScriptExecutor js;
  [SetUp]
  1 reference
  public void SetUp() {
    driver = new ChromeDriver();
    js = (IJavaScriptExecutor)driver;
    vars = new Dictionary<string, object>();
  }
  [TearDown]
  1 reference
  protected void TearDown() {
    driver.Quit();
  }
  [Test]
  0 references
  public void lozinka() {
    driver.Navigate().GoToUrl("https://apartmeet.onrender.com/");
    driver.Manage().Window.Size = new System.Drawing.Size(1298, 736);
    driver.FindElement(By.CssSelector(".header-menu")).Click();
    driver.FindElement(By.CssSelector(".link:nth-child(1) > p")).Click();
    {
      var element = driver.FindElement(By.CssSelector(".link:nth-child(1) > p"));
      Actions builder = new Actions(driver);
      builder.MoveToElement(element).Perform();
    }
    {
      var element = driver.FindElement(By.tagName("body"));
      Actions builder = new Actions(driver);
      builder.MoveToElement(element, 0, 0).Perform();
    }
    driver.FindElement(By.Name("userName")).Click();
    driver.FindElement(By.Name("userName")).SendKeys("ivan");
    driver.FindElement(By.Name("email")).Click();
    driver.FindElement(By.Name("email")).SendKeys("ivan@test.hr");
    var elements = driver.FindElements(By.CssSelector(".form-line>p"));
    Assert.True(elements.Count > 0);
  }
```

Slika 6.8 Kod ispitnog slučaja Testiranje dodavanja člana

Test dodavanja člana bez lozinke provjerava kako sustav reagira na pokušaj unosa korisnika bez postavljene lozinke. Skripta otvara aplikaciju, navigira do stranice za dodavanje korisnika, unosi korisničko ime i e-mail, ali ne i lozinku, te pokušava potvrditi unos. Očekivani rezultat je nedostupnost gumba "Dodaj člana". Time je potvrđeno da sustav sprječava registraciju korisnika bez lozinke.

5. Ispitivanje - Dodavanje člana bez emaila

1. Ulazni podaci:

- password: 123
- username: ivan

---

2. Koraci testiranja:

- Kliknuti na izbornik (tri crtice u gornjem lijevom kutu)
- Kliknuti "Dodaj člana"
- Unijeti username člana
- Unijeti lozinku člana
- Namjerno izostaviti unos emaila
- Pokušati stisnuti "Dodaj člana"

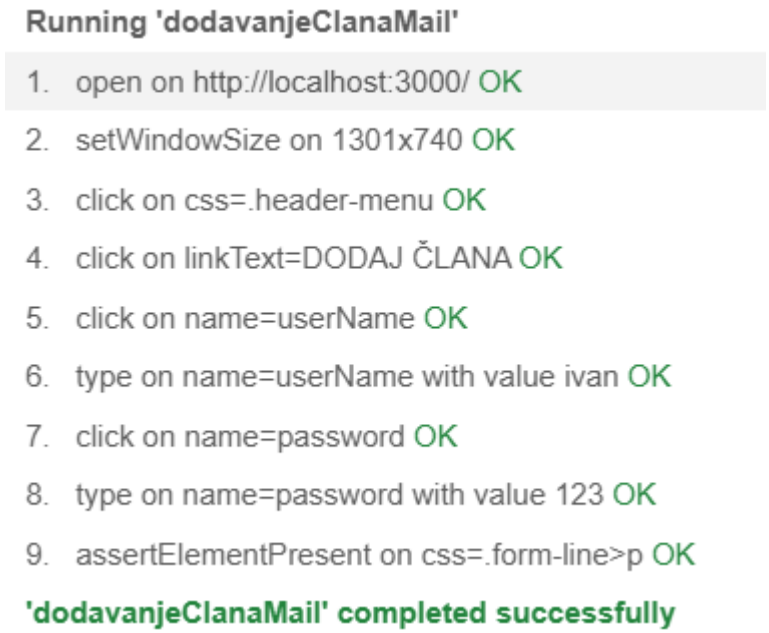
---

3. Očekivani izlaz

- Zaslon ostaje nepromjenjen (gumb "Dodaj člana" nije dostupan)

---

4. Dobiveni izlaz



Slika 6.9 Dobiveni izlaz za Testiranje dodavanja člana

---

5. Izvorni kod ispitnog slučaja

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Threading;
using OpenQA.Selenium;
using OpenQA.Selenium.Chrome;
using OpenQA.Selenium.Firefox;
using OpenQA.Selenium.Remote;
using OpenQA.Selenium.Support.UI;
using OpenQA.Selenium.Interactions;
using NUnit.Framework;
[TestFixture]
0 references
public class DodavanjeClanaMailTest {
  12 references
  private IWebDriver driver;
  1 reference
  public IDictionary<string, object> vars {get; private set;}
  1 reference
  private IJavaScriptExecutor js;
  [SetUp]
  1 reference
  public void SetUp() {
    driver = new ChromeDriver();
    js = (IJavaScriptExecutor)driver;
    vars = new Dictionary<string, object>();
  }
  [TearDown]
  1 reference
  protected void TearDown() {
    driver.Quit();
  }
  [Test]
  0 references
  public void dodavanjeClanaMail() {
    driver.Navigate().GoToUrl("https://apartmeet.onrender.com/");
    driver.Manage().Window.Size = new System.Drawing.Size(1301, 740);
    driver.FindElement(By.CssSelector(".header-menu")).Click();
    driver.FindElement(By.LinkText("DODAJ ČLANA")).Click();
    driver.FindElement(By.Name("userName")).Click();
    driver.FindElement(By.Name("userName")).SendKeys("ivan");
    driver.FindElement(By.Name("password")).Click();
    driver.FindElement(By.Name("password")).SendKeys("123");
    var elements = driver.FindElements(By.CssSelector(".form-line>p"));
    Assert.True(elements.Count > 0);
  }
}
```

Slika 6.10 Kod ispitnog slučaja Testiranje dodavanja člana

Test dodavanja člana bez emaila provjerava kako sustav reagira na pokušaj unosa korisnika bez emaila. Skripta otvara aplikaciju, navigira do stranice za dodavanje korisnika, unosi korisničko ime i lozinku, ali ne i email, te pokušava potvrditi unos. Očekivani rezultat je nedostupnost gumba "Dodaj člana". Time je potvrđeno da sustav sprječava registraciju korisnika bez emaila.

6. Ispitivanje - Brisanje člana koji ne postoje

1. Ulazni podaci:

- username: teststanar

---

2. Koraci testiranja:

- Kliknuti na izbornik (tri crtice u gornjem lijevom kutu)

- Kliknuti "Brisanje člana"
- Unijeti pogrešni username člana
- Kliknuti "Obriši korisnika"

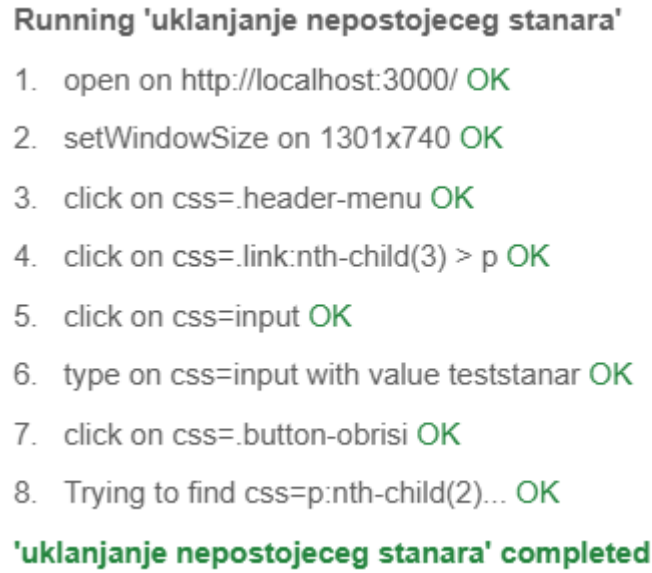
---

3. Očekivani izlaz

- "Greška pri brisanju korisnika"

---

4. Dobiveni izlaz



Slika 6.11 Dobiveni izlaz za Testiranje brisanja nepostojećeg člana

---

5. Izvorni kod ispitnog slučaja

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Threading;
using OpenQA.Selenium;
using OpenQA.Selenium.Chrome;
using OpenQA.Selenium.Firefox;
using OpenQA.Selenium.Remote;
using OpenQA.Selenium.Support.UI;
using OpenQA.Selenium.Interactions;
using NUnit.Framework;
[TestFixture]
0 references
public class UklanjanjenepostojecegstanaraTest {
  11 references
  private IWebDriver driver;
  1 reference
  public IDictionary<string, object> vars {get; private set;}
  1 reference
  private IJavaScriptExecutor js;
  [SetUp]
  1 reference
  public void SetUp() {
    driver = new ChromeDriver();
    js = (IJavaScriptExecutor)driver;
    vars = new Dictionary<string, object>();
  }
  [TearDown]
  1 reference
  protected void TearDown() {
    driver.Quit();
  }
  [Test]
  0 references
  public void uklanjanjenepostojecegstanara() {
    driver.Navigate().GoToUrl("https://apartmeet.onrender.com/");
    driver.Manage().Window.Size = new System.Drawing.Size(1301, 740);
    driver.FindElement(By.CssSelector(".header-menu")).Click();
    driver.FindElement(By.CssSelector(".link:nth-child(3) > p")).Click();
    driver.FindElement(By.CssSelector("input")).Click();
    driver.FindElement(By.CssSelector("input")).SendKeys("teststanar");
    driver.FindElement(By.CssSelector(".button-obrisi")).Click();
    Assert.That(driver.FindElement(By.CssSelector("p:nth-child(2)")).Text, Is.EqualTo("Greška pri brisanju korisnika"));
  }
}
```

Slika 6.12 Kod ispitnog slučaja Testiranje brisanja člana

Test uklanjanja nepostojećeg stanara provjerava kako sustav reagira na pokušaj brisanja korisnika koji nije u bazi podataka. Skripta otvara aplikaciju, navigira do stranice za brisanje korisnika, unosi ime "teststanar" i pokušava ga obrisati.

Očekivani rezultat je prikaz poruke "Greška pri brisanju korisnika", što test uspješno verificira. Time je potvrđeno da sustav ne dopušta brisanje nepostojećih korisnika.

Ispitivanje sustava

1. Ispitivanje - Prijava korisnika

1. Ulazni podaci:

- Korisničko ime: ivan
- Lozinka: test1234

---

2. Koraci testiranja:

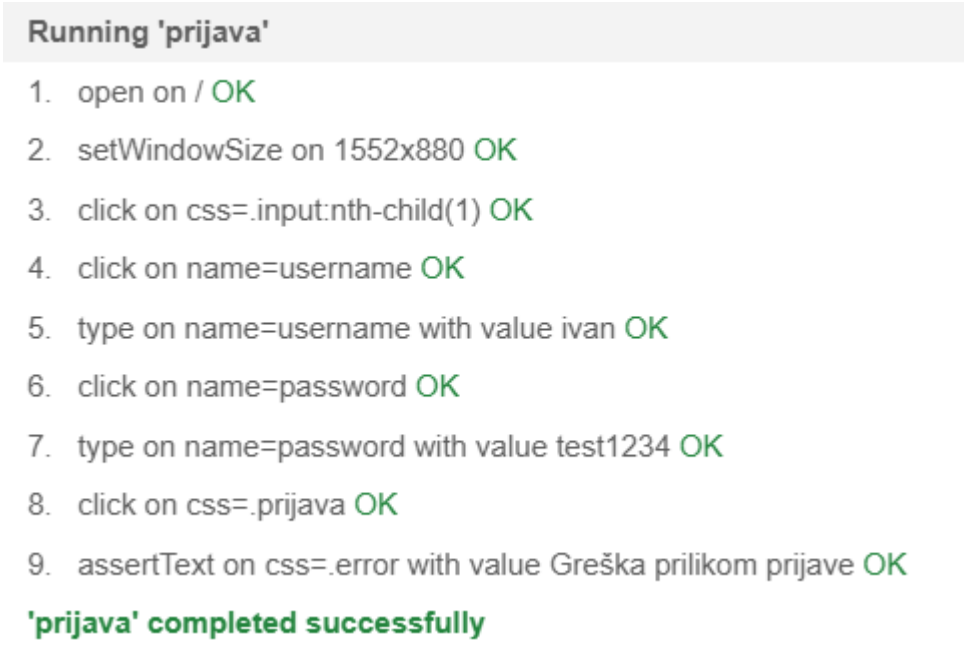
1. Otvoriti aplikaciju
2. Unijeti korisničko ime i lozinku u odgovarajuća polja
3. Kliknuti na gumb za prijavu
4. Provjeriti je li prijava bila neuspješna

---

3. Očekivani izlaz Greška prilikom prijave

---

4. Dobiveni izlaz



Slika 6.1 Dobiveni izlaz za Testiranje neuspješne prijave

Testirali smo neuspješnu prijavu kako bismo provjerili kako sustav reagira na neispravne korisničke podatke. Očekivani rezultat bio je prikaz poruke o grešci, što se i dogodilo. Time je test uspješno izvršen jer sustav ne dopušta neovlašteni pristup aplikaciji.

2. Ispitivanje - Odjava korisnika

1. Ulazni podaci:

- Kliknuti na izbornik (simbol tri crtice u gornjem lijevom kutu)
- Kliknuti na "Odjavi se"

---

2. Koraci testiranja:

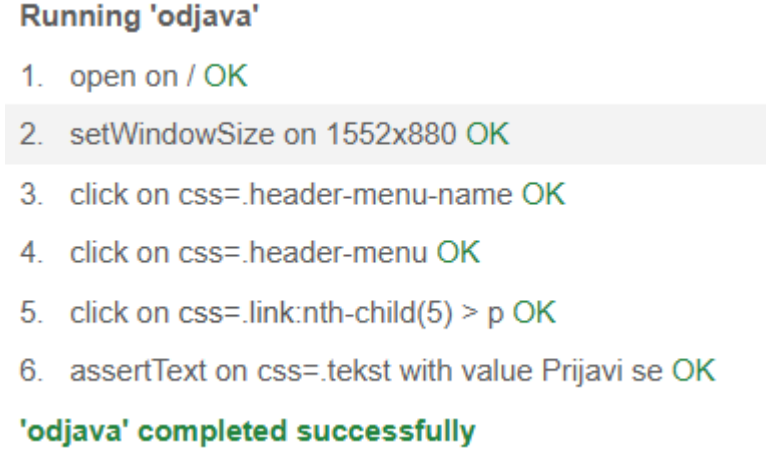
1. Otvariti izbornik (kliknuti na simbol u gornjem lijevom kutu)
2. Kliknuti na "Odjavi se"
3. Provjeriti jesmo li "izašli" iz aplikacije

---

3. Očekivani izlaz Prikaz ekrana za prijavu u aplikaciju

---

4. Dobiveni izlaz



Slika 6.2 Dobiveni izlaz za Testiranje odjave korisnika
Testirali smo funkcionalnost odjave korisnika iz sustava. Očekivani rezultat bio je prikaz opcije "Prijavi se", to jest izlazak iz apliakcije, što se i dogodilo, potvrđujući da je korisnik uspješno odjavljen. Time je test verificirao da sustav pravilno izvršava proces odjave i vraća korisnika na početnu stranicu prijave.

3. Ispitivanje - Pozivanje nepostojeće funkcije

1. Ulazni podaci:

- Unijeti https://apartmeet.onrender.com/izmjeniClana kao URL adresu

---

2. Koraci testiranja:

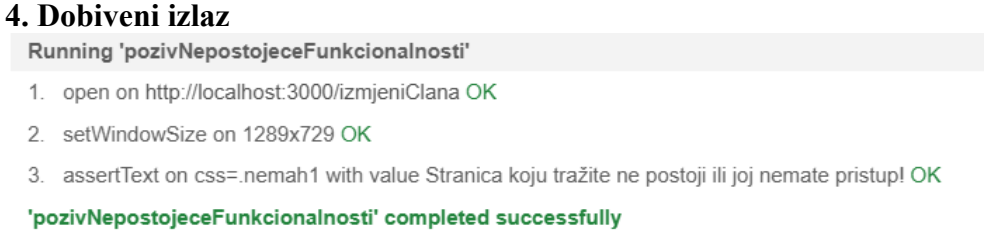
1. Biti ulogirani u aplikaciju
2. Dodati /izmjeniClana na kraju URL-a (nije bitno nalazi li se Korisnik na početnoj stranici ili nekoj drugoj)
3. Provjeriti što prikazuje dobivena stranica

---

3. Očekivani izlaz Stranica koju tražite ne postoji ili joj nemate pristup!

---

4. Dobiveni izlaz



Slika 6.3 Dobiveni izlaz za Testiranje poziva funkcionalnosti koja ne postoji
Testirali smo pristup nepostojećoj funkcionalnosti unutar sustava. Očekivani rezultat bio je prikaz poruke "Stranica koju tražite ne postoji ili joj nemate pristup!", što se i

dogodilo, potvrđujući da sustav ispravno reagira na pozive funkcionalnosti koje ne postoje. Time je test uspješno odrađen.

4. Ispitivanje - Pozivanje nedostupne funkcije

1. Ulazni podaci:

- Unijeti https://apartmeet.onrender.com/kreirajSastanak kao URL adresu

---

2. Koraci testiranja:

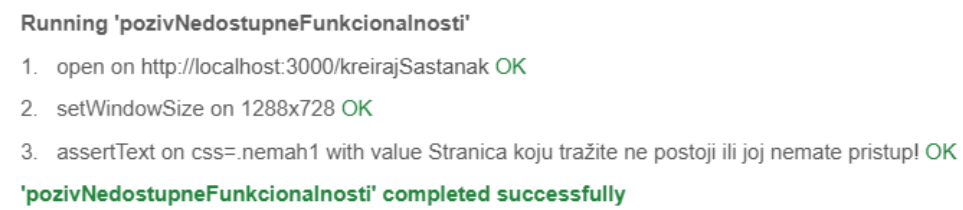
1. Biti ulogirani u aplikaciju
2. Dodati /kreirajSastanak na kraju URL-a (nije bitno nalazi li se Admin/Stanar na početnoj stranici ili nekoj drugoj)
3. Provjeriti što prikazuje dobivena stranica

---

3. Očekivani izlaz Stranica koju tražite ne postoji ili joj nemate pristup!

---

4. Dobiveni izlaz



Slika 6.4 Dobiveni izlaz za Testiranje poziva funkcije kojoj korisnik nema pristup

Testirali smo pristup nedostupnoj funkcionalnosti unutar sustava. Očekivano je bilo da se prikaže poruka "Stranica koju tražite ne postoji ili joj nemate pristup!", što se i dogodilo, potvrđujući ispravnu reakciju sustava. Test je uspješno pokazao da samo korisnici s određenim ovlastima (Predstavnici) mogu koristiti opciju Kreiranja Sastanka.

7. Tehnologije za implementaciju aplikacije

Korištene tehnologije i alati

- 1. Programski jezici: C# 12.0, JavaScript ES2023
- 1. Radni okviri i biblioteke: Node 20.14.0, React 18
- 1. Baza podataka: SQlite 3.42
- 1. Razvojni alati: VS Code 1.85
- 1. Alati za ispitivanje: Selenium 4.0
- 1. Alati za razmeštaj: Docker 24.0.6
- 1. Cloud platforma: Render

Opis korištenih tehnologija

Za backend koristimo C# i Node.js. C# je dobar izbor jer se dobro uklapa s .NET-om.

Frontend smo odlučili napraviti u Reactu (verzija 18) jer je jednostavan za korištenje. To nam olakšava održavanje koda i ponovnu upotrebu elemenata sučelja.

Kao bazu podataka koristimo SQLite (verzija 3.42), jer je lagan i jednostavan za korištenje. SQLite nam je odgovarao jer ne zahtijeva zaseban server, što je korisno za manje aplikacije i lokalni razvoj.

Za testiranje smo odabrali Selenium (verzija 4.10) jer omogućuje automatizirano testiranje web aplikacija.

Za deployment koristimo Docker (verzija 24.0.6) jer omogućuje jednostavno pokretanje aplikacije u različitim okruženjima.

Aplikacija je hostana na Renderu, jer je jednostavan za korištenje i omogućuje besplatan hosting za manje projekte, što nam je bilo korisno tijekom razvoja.

8. Upute za puštanje u pogon

Lokalna instalacija

Backend

Za korištenje backenda potrebni su Docker 27.3.1 i dotnet 8.0.403 te paketi:

- Microsoft.AspNetCore.Authentication.Google 8.0.10
- Microsoft.AspNetCore.Authentication.JwtBearer 8.0.10
- Microsoft.EntityFrameworkCore 8.0.10
- Microsoft.EntityFrameworkCore.Design 8.0.10
- Microsoft.EntityFrameworkCore.Sqlite 8.0.10.

Instalacija se provodi naredbom:

```
git clone https://github.com/fer-ncakija/progiG11.3.git
```

Pokretanje se provodi naredbama:

```
cd .\IzvorniKod\Apartmeet\Apartmeet.Api

dotnet run
```

Server se pokreće na adresi http://localhost:5066.

Frontend

Za korištenje backenda potreban je Node 20.14.0.

Instalacija se provodi naredbama:

```
git clone https://github.com/fer-ncakija/progiG11.3.git

cd .\IzvorniKod\"Apartmeet - Front"

npm install
```

Pokretanje se provodi naredbom:

```
npm start
```

Server se pokreće na adresi http://localhost:3000.

Konfiguracijske datoteke

U datoteku `appsettings.json` potrebno je zamijeniti `"ClientId": "Environment:ClientId", "ClientSecret": "Environment:ClientSecret"` s `"ClientId": "418123801091-j7m2506kqlf26kfvh1teq9doe7pu5us1.apps.googleusercontent.com", "ClientSecret": "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"` te `"MailClient" : { "Sender":"", "Host":"", "Port":0, "UserName":"", "Password":"", "Recepient":"" }` s `"MailClient" : { "Sender":"apartmeetprogi@gmail.com", "Host":"smtp-relay.brevo.com", "Port":587, "UserName":"83898d001@smtp-brevo.com", "Password":"YWPky16GpwD8jE3z", "Recepient":"apartmeetprogi@gmail.com" }`.

Instalacija na javnom poslužitelju

Na render.com napraviti Static Site i Web Service te oboje povezati s GitHub repozitorijem i uključiti Auto Deploy.

Konfiguracijske datoteke

U `appsettings.json` napraviti suprotne promjene od onih za tu datoteku kod lokalne instalacije. Na Renderu u Environment Variables postaviti `"ClientId": "418123801091-j7m2506kqlf26kfvh1teq9doe7pu5us1.apps.googleusercontent.com"` i `"ClientSecret": "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"` te dodati Secret File `appsettings.User.json` koji sadrži `"MailClient" : { "Sender":"apartmeetprogi@gmail.com", "Host":"smtp-relay.brevo.com", "Port":587, "UserName":"83898d001@smtp-brevo.com", "Password":"YWPky16GpwD8jE3z", "Recepient":"apartmeetprogi@gmail.com" }`.

Pristup aplikaciji na javnom poslužitelju

Korisnici pristupaju aplikaciji na URL-u: https://apartmeet.onrender.com/. Početno postoji samo korisnik administrator s podacima:

username: admin

password: admin.

Administrator mora kreirati druge korisnike kako bi im omogućio običnu prijavu ili prijavu pomoću Google-a.

Baza podataka

U svim slučajevima baza podataka se automatski pokreće pokretanjem backend servera i sadrži korisnika administratora s podacima:

username: admin

password: admin

role: admin.

A. Dnevnik promjena dokumentacije

Rev.	Opis promjene/dodatka	Autori	Datum
0.1	Kreirana stranica Home	Borna Elez	29.10
0.1.1	Home stranica ispunjena osnovnim podacima	Borna Elez	29.10
0.2	Kreirana stranica 1. Opis projektnog zadatka	Borna Elez	29.10
0.2.1	Stranica 1. Opis projektnog zadatka ispunjena sa osnovnim podacima o projektu	Borna Elez	29.10
0.3	Kreirane stranice 2. Analiza zahtjeva i 4. Specifikacija zahtjeva sustava	Borna Elez	3.11
0.3.1	Dodani detaljniji opisi funkcijski i nefunkcijskih zahtjeva sa vlastitim ID-om te Dijagram obrazaca uporabe za kritične sustave i integracija	Borna Elez	3.11
0.4	Dodani dijagrami obrazaca	Borna Elez	10.11
0.41	Dodani sekvencijski dijagrami	Borna Elez	10.11
0.42	Dodana arhitektura i dizajn sustava	Nina Čakija, Petar Rihtarec, Borna Kučić	10.11
0.43	Dodan detaljan opis Baze podataka	Borna Elez	10.11
0.44	Uređivanje izgleda Wiki strance	Borna Elez	10.11
0.5	Ispravak sekvencijskih dijagrama i dijagrama razreda	Borna Elez	15.11

B. Prikaz aktivnosti grupe

Dnevnik sastajanja

1. sastanak

- *Datum*: 18.10.2024.
- *Prisustvovali*: N. Čakija, B. Kučić, P. Rihtarec, E. Bradić, D. Požega, B. Elez, N. Špehar
- *Teme sastanka*: Kreiran Discord server, dogovor oko tehnologija koje će se korisiti na frontendu i backendu, dogovor oko podjela uloga u timu

2. sastanak

- *Datum*: 21.10.2024.
- *Prisustvovali*: N. Čakija, B. Kučić, P. Rihtarec, E. Bradić, D. Požega, B. Elez, N. Špehar
- *Teme sastanka*: Razrada ideje aplikacije: funkcionalni i nefunkcionalni zahtjevi, izgled aplikacije, organizacija unutar aplikacije

3. sastanak

- *Datum*: 30.10.2024.
- *Prisustvovali*: E. Bradić, D. Požega, N. Špehar
- *Teme sastanka*: Napravljena Whatsapp grupa za lakšu komunikaciju frontenda, razmjena materijala namjenjenog za učenje odabrane tehnologije React, raspoređen posao unutar podtima

4. sastanak

- *Datum*: 31.10.2024.
- *Prisustvovali*: N. Čakija, B. Kučić, P. Rihtarec
- *Teme sastanka*: Razmjena materijala namjenjenog za učenje odabrane tehnologije ASP.Net Core, raspoređen posao unutar podtima

5. sastanak

- *Datum*: 6.11.2024.
- *Prisustvovali*: N. Čakija, B. Kučić, P. Rihtarec
- *Teme sastanka*: Dogovor oko Login funkcionalnosti i zajedničko rješavanje određenih nejasnoća vezanih uz Login i aplikacije općenito

6. sastanak

- *Datum*: 11.11.2024.
- *Prisustvovali*: N. Čakija, B. Kučić, P. Rihtarec, E. Bradić, D. Požega, B. Elez, N. Špehar
- *Teme sastanka*: Spajanje frontenda s backendom i usklađivanje funkcionalnosti login-a

7. sastanak

- *Datum*: 12.11.2024.
- *Prisustvovali*: N. Čakija, B. Kučić, P. Rihtarec, E. Bradić, D. Požega, B. Elez, N. Špehar
- *Teme sastanka*: Usavršavanje dodatne funkcionalnosti dodavanja članova u bazu podataka pomoću forme koja traži ime, mail, lozinku te ulogu

8.sastanak

- *Datum*: 12.12.2024.
- *Prisustvovali*: N. Čakija, B. Kučić, P. Rihtarec, E. Bradić, D. Požega, B. Elez, N. Špehar
- *Teme sastanaka*: Definirati daljnji tok projekta, podjela zadataka, razrada ideje dizajna sustava

9.sastanak

- *Datum*: 18.12.2024.
- *Prisustvovali*: N. Čakija, B. Kučić, P. Rihtarec, E. Bradić, D. Požega, B. Elez, N. Špehar
- *Teme sastanaka*: Usklađivanje slanja mail-a s novim uređenjem sastanka

10.sastanak

- *Datum*: 11.1.2025.
- *Prisustvovali*: B. Kučić, P. Rihtarec
- *Teme sastanaka*: Implementacija funkcionalnosti na temelju vanjske aplikacije

11.sastanak

- *Datum*: 21.1.2025.
- *Prisustvovali*: N. Čakija, B. Kučić, P. Rihtarec, E. Bradić, D. Požega, B. Elez, N. Špehar
- *Teme sastanaka*: Finaliziranje aplikacije

Plan rada

- kreiranje baze podataka i modela korištenih u aplikaciji
- definiranje svih putanja WebApi-a
- konfiguracija OAuth 2.0 servisa za autentifikaciju korisnika
- konfiguracija servisa Google za OAuth prijavu
- dizajn početne stranice i ostvarenje potrebnih funkcionalnosti iste
- dizajn stranice za dodavanje članova
- ostvarenje funkcionalnosti dodavanja članova samo za admina

Tablica Aktivnosti

Aktivnost	Borna Kučić	Petar Rihtarec	Ema Bradić	Dorotea Požega	Borna Elez	Nikola Špehar	Nina Čakija
Priprema i istraživanje za rad	16	14	15	12	3	6	10
Upravljanje projektom							18
Opis projektnog zadatka					4		
Funkcionalni zahtjevi			1	1	2	1	
Opis pojedinih obrazaca					3	2	
Dijagram obrazaca					3		
Sekvencijski dijagrami					3	1.5	
Opis ostalih zahtjeva					3		
Arhitektura i dizajn sustava	2	2			1		3
Baza podataka	2						
Dijagram razreda					2	1.5	
Dijagram stanja					2		
Dijagram aktivnosti					2		
Dijagram komponenti					2		
Korištene tehnologije i alati					1		
Ispitivanje programskog rješenja	4		1	1		1	3
Dijagram razmještaja					3		
Upute za puštanje u pogon	2						
Puštanje u pogon	6						
Dnevnik sastajanja				1	2		2
Zaključak i budući rad							
Popis literature					1		
Dizajn aplikacije			4	2		1	
Izrada aplikacije	22	21	10	10		8	22
Izrada login stranice				5			
Izrada pocetne stranice			3			1	
Izrada forme za dodavanje clanova			3			2	
Izrada baze podataka	4	2					
Spajanje s bazom podataka	2	3					1
Izrada prezentacije	2				3		1

C. Popis literature

- Nikolina Frid, Alan Jović Modeliranje programske potpore UML-dijagramima (E-skripta (radna inačica))
- Programsko inženjerstvo, FER ZEMRIS, http://www.fer.hr/predmet/proinz
- Astah Community, http://astah.net/editions/uml-new

- Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva, FER ZEMRIS, PROCESI PROGRAMSKOG INŽENJERSTVA (e-skripta)
- The Unified Modeling Language, https://www.uml-diagrams.org/
- ASP.NET Core Full Course For Beginners, https://www.youtube.com/watch?v=AhAxLiGC7Pc