Programsko inženjerstvo

Ak. god. 2023./2024.

Iznajmi Romobil

Dokumentacija, Rev. *1*

Grupa: *Codeblaze*

Voditelj: *Marin Kvesić*

Datum predaje: *17.11.2023.*

Nastavnik: *Igor Stančin*

Sadržaj

1 Dnevnik promjena dokumentacije 3

2 Opis projektnog zadatka 5

3 Specifikacija programske potpore 10

3.1 Funkcionalni zahtjevi 10

3.1.1 Obrasci uporabe 12

3.1.2 Sekvencijski dijagrami 29

3.2 Ostali zahtjevi 35

4 Arhitektura i dizajn sustava 36

4.0.1 Opis arhitekture 36

4.0.2 MVC arhitektura 37

4.1 Baza podataka 38

4.1.1 User 39

4.1.2 Preferences 40

4.1.3 Social Media 40

4.1.4 PrivacySettings 41

4.1.5 Document 41

4.1.6 Scooter 42

4.1.7 Listing 42

4.1.8 Review 43

4.1.9 Transaction 44

4.1.10 Invoice 44

4.1.11 Notification 44

4.1.12 ChatSession 45

4.1.13 Message 45

4.1.14 ImageChangeRequest 46

4.1.15 User 46

4.1.16 Preferences 47

4.1.17 Social Media 48

4.1.18 PrivacySettings 48
4.1.19 Document 49
4.1.20 Scooter 49
4.1.21 Listing 50
4.1.22 Review 51
4.1.23 Transaction 52
4.1.24 Invoice 52
4.1.25 Notification 53
4.1.26 ChatSession 53
4.1.27 Message 54
4.1.28 ImageChangeRequest 54
4.1.29 Dijagram baze podataka 56
4.1.30 Dijagram razreda 56
4.2 Dijagram stanja 61
4.3 Dijagram aktivnosti 63
4.4 Dijagram komponenti 65

5 Implementacija i korisničko sučelje 66
5.1 Korištene tehnologije i alati 66
5.2 Ispitivanje programskog rješenja 67
5.2.1 Ispitivanje komponenti 67
5.2.2 Ispitivanje sustava 73
5.3 Dijagram razmještaja 76
5.4 Upute za puštanje u pogon 78

6 Zaključak i budući rad 82

Popis literature 84

Indeks slika i dijagrama 86

Dodatak: Prikaz aktivnosti grupe 87

1. Dnevnik promjena dokumentacije

Kontinuirano osvježavanje

Rev.	Opis promjene/dodatka	Autori	Datum
0.1	Napravljen predložak.	Matija Jakovac	30.10.2023.
0.2	Dodan opisa projekta. Dodani obrasci uporabe i njihovi opisi.	Karla Šoštar, Katarina Đuroković, Matea Bušić	07.11.2023.
0.2.1	Dodan ostatak obrazaca uporabe i njihovi opisi.	Karla Šoštar, Katarina Đuroković, Matea Bušić	08.11.2023.
0.3	Arhitektura i dizajn sustava	Mirna Knez	08.11.2023.
0.4	Ispravak obrazaca uporabe	Matea Bušić	10.11.2023.
0.5	Dodani zapisnici sastanaka	Matea Bušić	11.11.2023.
0.6	Popravak baze podataka	Mirna Knez	12.11.2023.

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Rev.	**Opis promjene/dodatka**	**Autori**	**Datum**
0.9	Dodani funkcionalni zahtjevi	Matea Bušić	13.11.2023.
0.10	Dodani dijagrami razreda	Mirna Knez	14.11.2023.
0.10.1	Ispravak obrazaca uporabe	Karla Šoštar, Katarina Đuroković, Matea Bušić	15.11.2023.
0.11	Dodani sekvencijski dijagrami i dijagrami obrazaca uporabe	Karla Šoštar, Katarina Đuroković	17.11.2023.
1.0	Završna verzija za prvu reviziju	Matea Bušić	17.11.2023.

2. Opis projektnog zadatka

Cilj ovog projekta je razviti programsku podršku za stvaranje web aplikacije „Iznajmi romobil“ koja će korisnicima omogućiti da iznajme svoj električni romobil u periodima dana kada ga ne koriste. Pomoću aplikacije korisnici će moći brzo i jednostavno pristupiti električnim romobilima dostupnima za najam kao i postaviti ponudu za iznajmljivanje svog romobila. Električni romobili danas su odlična alternativa za automobile te zbog svoje ekološke prihvatljivosti i praktičnosti postaju sve popularniji u većim urbanim sredinama. Korisnik koji želi iznajmiti svoj romobil, prilikom registracije romobila, unosi osnovne informacije o romobilu. Kada je romobil registriran, za njega se može objaviti oglas za iznajmljivanje pri čemu vlasnik upisuje gdje se romobil trenutno nalazi te gdje i kada mora biti vraćen. Korisnik koji želi unajmiti romobil, na temelju ponuđenih romobila i informacija o njima odabire onaj koji je u tom trenutku dostupan i najviše odgovara njegovim potrebama.

Prilikom pokretanja aplikacije, korisnicima se neovisno o tome jesu li prijavljeni ili ne prikazuje popis svih aktivnih oglasa romobila. Uz svaki su romobil navedene njegove karakteristike, cijena, lokacija na kojoj se romobil nalazi te vrijeme i lokacija gdje romobil mora biti vraćen. **Neregistrirani korisnici** mogu pregledavati trenutno dostupne romobile i njihova cijene, ali ih ne mogu iznajmiti. Nakon što kreiraju novi korisnički račun, ponuđeni romobili im postaju dostupni za najam. Prilikom kreiranja novog računa korisnici moraju unijeti sljedeće podatke:

- *ime i prezime*
- *email adresa*
- *nadimak*
- *broj kartice*

Osim navedenog, korisnici prilikom registracije moraju dostaviti kopiju osobne iskaznice i potvrdu o nekažnjavanju. Nakon što su uneseni svi podaci i dostavljeni potrebni dokumenti, korisnik čeka pregled dokumenata odnosno odobravanje ili odbijanje registracije. Dok registracija nije odobrena, korisnik se ne može prijaviti

u sustav. Ako je zahtjev za registraciju odbijen zbog neispravnosti dostavljenih dokumenata, korisnik može ponovno predati dokumente. Korisnici koji su unaprijed registrirani, mogu se prijaviti u sustav sa svojim postojećim korisničkim računom tako da unesu email adresu i lozinku. Ako korisnik prilikom unosa podataka za prijavu unese podatke koji ne odgovaraju nijednom registriranom korisniku u bazi, šalje mu se obavijest o neispravnosti podataka. **Klijent** je korisnik aplikacije koji može pregledavati i unajmljivati romobile, ali nema registriranih vlastitih romobila. Kada odabere romobil koji želi unajmiti, klijent se javlja na oglas s automatski generiranom porukom koja sadrži zahtjev za unajmljivanje. Nakon što je ponuda prihvaćena, klijentu se šalje obavijest da je iznajmljivanje odobreno i oglas se briše. Klijent prije pokretanja romobila provjerava odgovara li fotografija romobila njegovom stvarnom stanju. Ako ne odgovara, on ima mogućnost odabirom opcije "Zamijeni sliku" zamijeniti sliku romobila novom slikom i kratkim opisom o razlikama između nove i stare slike. Na kraju iznajmljivanja, klijent vraća romobil i u aplikaciji potvrđuje da ga je vratio. Nakon toga slijedi provjera je li romobil vraćen u pravo vrijeme te se izračunava cijena koju klijent mora platiti. Svaki klijent u aplikaciji može pregledati svoj profil na kojem se nalaze njegovi osobni podaci. Za sve podatke on sam određuje hoće li oni biti javno dostupni ili ne, a iste može urediti odabirom opcije "Uredi profil". Pri uređivanju profila provjerava se jesu li novi podatci uneseni u ispravnom formatu, ako nisu korisnik dobiva obavijest o neispravnosti. Nakon unosa promjena, klijent mora odabrati opciju "Spremi promjene" kako bi potvrdio pohranjivanje promjena u bazu podataka. Na profilu svakog klijenta moraju biti vidljive ocjene i komentari iznajmljivača. Klijent, u trenutku kada registrira svoj romobil, postaje **iznajmljivač**. Prilikom registracije romobila potrebno je unijeti sljedeće podatke o romobilu:

- *naziv proizvođača*
- *naziv modela*
- *kapacitet baterije*
- *maksimalna brzina*
- *URL fotografije romobila*
- *maksimalni domet*
- *godina proizvodnje*
- *dostupnost*
- *dodatne informacije (po želji)*

Prilikom postavljanja ponude za iznajmljivanje, iznajmljivač unosi trenutnu lo-

kaciju romobila, lokaciju na koju želi da se romobil vrati, vrijeme do kada romobil mora biti vraćen, cijenu iznajmljivanja po prijeđenom kilometru te iznos novčane kazne u slučaju da romobil ne bude vraćen na vrijeme. Ako je neki romobil dostupan za iznajmljivanje, iznajmljivač oglas može podijeliti i na nekoj društvenoj mreži odabirom opcije "Objavi na društvenu mrežu". Svaki iznajmljivač može pregledavati svoje registrirane romobile, brisati postojeće i dodavati nove. Ako iznajmljivač izbriše sve svoje registrirane romobile, ono ponovno postaje klijent. Unutar aplikacije dostupna je mogućnost izmjenjivanja poruka kako bi se klijent i iznajmljivač mogli dogovoriti oko najma. Iznajmljivač pregledava zahtjeve za iznajmljivanje i profile klijenata te tada može prihvatiti ili odbiti ponudu. Ako klijent zamijeni sliku romobila jer smatra da ne prikazuje stvarno stanje romobila, iznajmljivaču se šalje obavijest o zamjeni. On tada, ako smatra da klijentova slika nije ispravna, može poslati žalbu na zamjenu slika. Nakon što administrator donese odluku o zamjeni slika, klijentu i iznajmljivaču se šalje obavijest o donesenoj odluci. Na kraju iznajmljivanja, iznajmljivač kao i klijent dobiva obavijest da je iznajmljivanje završeno i cijenu iznajmljivanja koju mu klijent treba platiti. Iznajmljivač nakon završetka iznajmljivanja može ocijeniti klijenta i napisati komentar. **Administrator** pregledava dokumente dostavljene prilikom registracije, odbija ili prihvaća zahtjeve za registraciju, dodjeljuje nove uloge administratora te iste briše. Osim toga, on zaprima žalbe na zamjenu slika romobila. Nakon zaprimanja žalbe administrator pregledava slike i odabire onu koja će se pohraniti u bazu. Administrator ima pravo blokirati korisnika odnosno zabraniti mu pristup aplikaciji odabirom opcije "Blokiraj korisnika". Za takvog se korisnika u bazu podataka upisuje da je blokiran te će mu pri sljedećoj prijavi biti onemogućen pristup sustavu.

Ovaj način izvedbe aplikacije bi definitivno bio koristan za današnje društvo. Korištenje aplikacije pridonijelo bi smanjenju emisija štetnih plinova i gužve te povećanju mobilnosti. Električni romobili, kao što im naziv govori pokreću se električnom energijom te kao takvi doprinose smanjenju emisija štetnih plinova i zagađenja zraka. Kada bi se povećao broj korisnika električnih romobila odnosno kada bi ljudi umjesto automobila i taksija koristili električne romobile, smanjili bi se prometni zastoji te bi se poboljšala fluidnost prometa. Široka rasprostranjenost romobila po gradu i jednostavno pronalaženje oglasa povećali bi dostupnost prijevoza. Osim navedenih prednosti, aplikacija bi iznajmljivačima omogućila do-

datni prihod od iznajmljivanja romobila u periodima kada ga ne koriste, čime se maksimizira iskorištavanje resursa i potiče održiv način dijeljenja prijevoznih sredstava. Aplikacija je namijenjena stanovnicima urbanih sredina i turistima, pružajući praktično rješenje za brz i ekološki prihvatljiv prijevoz. Stanovnici gradova mogu iskoristiti romobile kao učinkovito sredstvo prijevoza između odredišta te mogu iznajmljivati svoje romobile, dok turisti mogu istraživati grad na jednostavan način, dodatno obogaćujući svoje iskustvo boravka. Aplikacija bi se mogla unaprijediti tako da web aplikaciju pretvorimo u mobilnu kako bi se olakšala njena upotreba i dodale dodatne funkcionalnosti koje bi ju učinile još popularnijom.

Danas, kako električni romobili postaju sve popularniji, susrećemo se s mnogo sličnih aplikacija koje su aktivne u svim većim gradovima u svijetu. Nažalost, u Hrvatskoj takve aplikacije i mogućnost najma i iznajmljivanja i dalje nisu dostupne u većini gradova. Ključna razlika između aplikacije Iznajmi romobil i postojećih rješenja jest ta što su sve izvedene u obliku mobilnih aplikacija te uglavnom vlasnik aplikacije iznajmljuje veći broj svojih romobila što znači da korisnici mogu unajmiti romobile na aplikaciji, ali ne i iznajmiti svoje. Ta je razlika upravo prednost aplikacije Iznajmi romobil jer svi korisnici mogu objaviti oglas za svoj romobil, neovisno o tome imaju li jedan romobil ili više njih. Primjer sličnog rješenja u obliku mobilne aplikacije jest Bird. Bird je mobilna aplikacija koja korisnicima omogućuje najam električnih romobila i bicikala. Romobili i bicikli se nalaze na više lokacija u gradovima. Korisnici se prilikom pokretanja aplikacije registriraju odnosno prijave u sustav, a nakon toga na mapi pronađu gdje se nalazi najbliži romobil/bicikl dostupan za najam. Kada dođu do te točke, odaberu romobil/bicikl, skeniraju QR kod na njemu i započinju svoju vožnju. Cijena najma ovisi o lokaciji, a računa se prema pređenim kilometrima. Po završetku vožnje, u aplikaciji se označava da je vožnja završena, romobil/bicikl se ostavlja na za to predviđenom mjestu, a cijena najma naplaćuje se s bankovnog računa povezanog na korisnikov profil. Neke od dodatnih mogućnosti koje aplikacija nudi su zakazivanja grupnih vožnji što je iznimno pogodno za turističke skupine, rezervacija vozila do 30 minuta unaprijed te brojne cjenovne pogodnosti za stalne korisnike.

Slika 2.1: Prikaz pronalaska romobila i početka vožnje

Slika 2.2: Prikaz dodatnih mogućnosti

3. Specifikacija programske potpore

3.1 Funkcionalni zahtjevi

Dionici:

1. Klijenti
2. Iznajmljivači
3. Administratori sustava
4. Razvojni tim

Aktori i njihovi funkcionalni zahtjevi:

1. Neregistrirani korisnik/posjetilac (inicijator) može:
   (a) pregledavati dostupne romobile
   (b) odabrati romobil da dobije detaljnije podatke o njemu
   (c) registrirati se u sustav, stvoriti novi korisnički račun
2. Klijent (inicijator) može:
   (a) pregledavati dostupne romobile
   (b) pregledati osobne podatke na svome profilu i urediti ih
   (c) izbrisati svoj profil
   (d) registrirati romobil
   (e) iznajmiti romobil
   (f) zamijeniti sliku romobila
   (g) vratiti romobil
   (h) pregledati svoje prethodne transakcije
   (i) pogledati profil drugog korisnika
   (j) pogledati razgovor s nekim drugim korisnikom
   (k) poslati poruku drugom korisniku
   (l) izbrisati razgovor s drugim korisnikom
3. Iznajmljivač (inicijator) može:

(a) pogledati detaljne informacije o svom registriranom romobilu
(b) promijeniti informacije o registriranom romobilu
(c) izbrisati registrirani romobil
(d) postaviti oglas za registrirani romobil
(e) objaviti oglas za iznajmljivanje romobila na društvene mreže
(f) pregledati i uređivati svoj oglas
(g) izbrisati oglas za iznajmljivanje romobila
(h) prihvatiti ili odbiti zahtjev za iznajmljivanje
(i) poslati žalbu na zamjenu slika romobila
(j) ocijeniti klijenta
(k) izbrisati svoju recenziju o drugom korisniku

4. Administrator (inicijator) može:

(a) pregledati dostupne romobile i detalje o njima
(b) pregledati dokumentaciju za registraciju i prihvatiti ili odbiti zahtjev za registraciju
(c) pregledati svoj profil
(d) urediti svoj profil
(e) izbrisati svoj profil
(f) Izbrisati oglas
(g) odlučiti poništava li se zamjena slika ili ne
(h) pogledati profil drugog korisnika
(i) izbrisati recenziju o nekom korisniku
(j) blokirati korisnika
(k) odblokirati korisnika
(l) dodati drugom korisniku ulogu administratora
(m) oduzeti drugom administratoru ulogu administratora

5. Baza podataka(sudionik):

(a) pohranjuje sve podatke o korisnicima i njihovim ovlastima
(b) pohranjuje sve podatke o romobilima, njihovim iznajmljivanjima i transakcijama

6. Banka (inicijator) može:

(a) umanjiti iznos novca na računu klijenta za iznos cijene iznajmljivanja
(b) uplati cijenu iznajmljivanja iznajmljivaču na račun

3.1.1 Obrasci uporabe

Opis obrazaca uporabe

UC1 - Pregledaj romobil

- **Glavni sudionik:** Neregistrirani korisnik, klijent, administrator
- **Cilj:** Pregledati dostupne romobile i detalje o njima i ponudi
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
  1. Prilikom učitavanja aplikacije prikazuju se romobili dostupni za iznajmljivanje
  2. Korisnik pronalazi i odabire željeni romobil
  3. Korisniku se prikazuju detaljnije informacije o romobilu i uvjetima iznajmljivanja

UC2 - Registriraj se

- **Glavni sudionik:** Neregistrirani korisnik
- **Cilj:** Napraviti korisnički račun kojim se pristupa sustavu
- **Sudionici:** Administrator, baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
  1. Neregistrirani korisnik odabire opciju „Registriraj se“
  2. Neregistriranom korisniku prikazuje se stranica za registraciju
  3. Neregistrirani korisnik unosi podatke za registraciju
  4. Unesena kopija osobne iskaznice i potvrda o nekažnjavanju šalju se administratoru na pregled
  5. Stvara se novi korisnički račun čiji se podatci pohranjuju u bazu podataka
  6. Neregistriranom korisniku se prikazuje stranica za čekanje odobrenja registracije
- **Opis mogućih odstupanja:**

  3.a Unos podataka u nedozvoljenom formatu ili unos već zauzetog nadimka ili e-mail adrese
    1. Neregistrirani korisnik dobiva odgovarajuću obavijest o neispravnosti podataka

2. Neregistrirani korisnik mijenja potrebne podatke i završava s unosom ili odustaje od registracije

UC3 - Odobri ili odbij registraciju

- **Glavni sudionik:** Administrator
- **Cilj:** Odlučiti o valjanosti kopije osobne iskaznice i potvrde o nekažnjavanju
- **Sudionici:** Klijent, baza podataka
- **Preduvjet:** Postoji barem jedan zahtjev za registraciju
- **Opis osnovnog tijeka:**
  1. Administrator pregledava dokumentaciju poslanu pri registraciji korisnika ili pri ponovnom slanju dokumentacije
  2. Administrator odlučuje o potvrđivanju dokumenata i njegova odluka se zapisuje u bazu podataka
  3. Klijenta čija se registracija provjerava se preusmjerava na stranicu za prijavu

UC4 - Ponovno predaj dokumente

- **Glavni sudionik:** Neregistrirani korisnik
- **Cilj:** Ponovno predati kopiju osobne iskaznice i potvrdu o nekažnjavanju na pregled
- **Sudionici:** Administrator, baza podataka
- **Preduvjet:** Prethodno odbijen zahtjev za registraciju
- **Opis osnovnog tijeka:**
  1. Neregistrirani korisnik unosi novu dokumentaciju za registraciju u sustav
  2. Dokumentacija se šalje administratoru na pregled
  3. Neregistriranom korisniku se prikazuje stranica za čekanje odobrenja registracije
- **Opis mogućih odstupanja:**

  1.a Unos dokumentacije u nedozvoljenom formatu
    1. Neregistrirani korisnik dobiva odgovarajuću obavijest o neispravnosti podataka
    2. Neregistrirani korisnik ispravno unosi dokumentaciju i završava s unosom ili odustaje od ponovne predaje dokumentacije

UC5 - Prijavi se

- **Glavni sudionik:** Klijent, administrator
- **Cilj:** Dobiti pristup određenim korisničkim funkcijama
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik se registrirao i odobrena mu je registracija
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju „Prijavi se“
  2. Korisniku se prikazuje stranica za prijavu
  3. Korisnik unosi podatke za prijavu
  4. Prijava je odobrena i korisnik dobiva pristup svojim korisničkim funkcijama
- **Opis mogućih odstupanja:**

  3.a Unos podataka koji ne odgovaraju nijednom registriranom korisniku u bazi podataka
    1. Korisnik dobiva odgovarajuću obavijest o neispravnosti podataka
    2. Korisnik mijenja potrebne podatke i završava s unosom ili odustaje od prijave

  4.a Prijava nije odobrena jer korisnik čeka na odobrenje registracije
    1. Korisniku se prikazuje stranica za čekanje odluke o registraciji

  4.b Prijava nije odobrena jer je korisniku odbijena dokumentacija za registraciju
    1. Korisnika se preusmjerava na stranicu za ponovnu predaju dokumentacije za registraciju

  4.c Prijava nije odobrena jer je korisnik blokiran i nema pristup sustavu
    1. Korisnik dobiva obavijest da je blokiran i nema više pristup sustavu

<u>UC6 - Odjavi se</u>

- **Glavni sudionik:** Klijent, administrator
- **Cilj:** Odjaviti se iz sustava
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju „Odjavi se“
  2. Korisnik napušta sustav

<u>UC7 - Pregledaj svoj profil</u>

- **Glavni sudionik:** Klijent, administrator

- **Cilj:** Pregledati korisničke podatke svog profila
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju „Moj profil“
  2. Korisniku se prikazuje stranica vlastitog profila i svi njegovi korisnički podatci
  3. Korisnik pregledava informacije o svom profilu

UC8 - Uredi svoj profil

- **Glavni sudionik:** Klijent, administrator
- **Cilj:**Promijeniti korisničke podatke i odlučiti koji će od njih biti javni, a koji privatni
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik na stranici svog profila odabire opciju „Uredi profil“
  2. Korisnik mijenja svoje korisničke podatke i njihovu dostupnost
  3. Korisnik potvrđuje promjene odabirom opcije „Spremi promjene“
  4. Baza podataka se ažurira
- **Opis mogućih odstupanja:**

  2.a Unos podataka u nedozvoljenom formatu ili unos već zauzetog nadimka ili e-mail adrese
    1. Korisnik dobiva odgovarajuću obavijest o neispravnosti podataka
    2. Korisnik mijenja potrebne podatke i završava s unosom ili odustaje od promjene

  3.a Korisnik ne potvrdi promjenu odabirom opcije „Spremi promjene“
    1. Korisnik dobiva obavijest da nije spremio podatke prije izlaska iz prozora za promjenu podataka
    2. Korisnik sprema promjene

UC9 - Izbriši svoj profil

- **Glavni sudionik:** Klijent, administrator
- **Cilj:** Izbrisati svoj korisnički račun iz baze podataka
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen

- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju „Izbriši profil“
  2. Korisnik potvrđuje svoj odabir
  3. Korisnikov profil se briše iz baze podataka
  4. Otvara se stranica za registraciju

UC10 - Registriraj romobil

- **Glavni sudionik:** Klijent
- **Cilj:** Dodavanje romobila na popis svojih romobila
- **Sudionici:** Baza podataka
- **Preduvjet:** Klijent je prijavljen
- **Opis osnovnog tijeka:**
  1. Klijent odabire opciju „Registriraj romobil“
  2. Klijent unosi podatke o romobilu i sliku trenutnog stanja romobila
  3. Registracija romobila se zabilježi u bazi podataka
- **Opis mogućih odstupanja:**

  2.a Unos podataka o romobilu u nedozvoljenom formatu
    1. Korisnik dobiva odgovarajuću obavijest o neispravnosti podataka
    2. Korisnik mijenja potrebne podatke i završava s unosom ili odustaje od registracije romobila

UC11 - Pregledaj svoj romobil

- **Glavni sudionik:** Iznajmljivač
- **Cilj:** Pregledati detaljne informacije o svom registriranom romobilu
- **Sudionici:** Baza podataka
- **Preduvjet:** Iznajmljivač je prijavljen, ima barem jedan registrirani romobil
- **Opis osnovnog tijeka:**
  1. Prikazuje se stranica s iznajmljivačevim registriranim romobilima
  2. Iznajmljivač pronalazi i odabire željeni romobil
  3. Iznajmljivaču se prikazuju detaljnije informacije o romobilu

UC12 - Uredi svoj romobil

- **Glavni sudionik:** Iznajmljivač
- **Cilj:** Promijeniti informacije o svom registriranom romobilu
- **Sudionici:** Baza podataka
- **Preduvjet:** Iznajmljivač je prijavljen, ima barem jedan registrirani romobil

- **Opis osnovnog tijeka:**
  1. Prikazuje se stranica s iznajmljivačevim registriranim romobilima
  2. Iznajmljivač pronalazi i odabire željeni romobil i opciju „Uredi“
  3. Iznajmljivač mijenja informacije o romobilu
  4. Iznajmljivač potvrđuje promjene odabirom opcije „Spremi promjene“
  5. Baza podataka se ažurira
- **Opis mogućih odstupanja:**

  3.a Unos podataka o romobilu u nedozvoljenom formatu
    1. Iznajmljivač dobiva odgovarajuću obavijest o neispravnosti podataka
    2. Iznajmljivač mijenja potrebne podatke i završava s unosom ili odustaje od registracije romobila

  4.a Korisnik ne potvrdi promjenu odabirom opcije „Spremi promjene“
    1. Korisnik dobiva obavijest da nije spremio podatke prije izlaska iz prozora za zamjenu podataka
    2. Korisnik sprema podatke

UC13 - Izbriši svoj romobil

- **Glavni sudionik:** Iznajmljivač
- **Cilj:** Izbrisati romobil iz registriranih romobila
- **Sudionici:** Baza podataka
- **Preduvjet:** Iznajmljivač je prijavljen, ima barem jedan registrirani romobil
- **Opis osnovnog tijeka:**
  1. Prikazuje se stranica s iznajmljivačevim registriranim romobilima
  2. Iznajmljivač pronalazi i odabire željeni romobil i opciju „Izbriši romobil“
  3. Romobil se briše iz registriranih romobila
  4. Ako nakon brisanja romobila iznajmljivač nema više registriranih romobila on postaje klijent
  5. Baza podataka se ažurira

UC14 - Oglasi romobil

- **Glavni sudionik:** Iznajmljivač
- **Cilj:** Objaviti oglas da je romobil dostupan za iznajmljivanje
- **Sudionici:** Baza podataka
- **Preduvjet:** Iznajmljivač je prijavljen, ima barem jedan registrirani romobil

- **Opis osnovnog tijeka:**
  1. Prikazuje se stranica s iznajmljivačevim registriranim romobilima
  2. Iznajmljivač pronalazi i odabire željeni romobil i opciju „Oglasi romobil“
  3. Iznajmljivač unosi uvjete iznajmljivanja i objavljuje oglas
  4. Oglas se prikazuje među dostupnim romobilima
- **Opis mogućih odstupanja:**

  3.a Unos neispravnih podataka o iznajmljivanju
    1. Iznajmljivač dobiva odgovarajuću obavijest o neispravnosti podataka
    2. Iznajmljivač mijenja potrebne podatke i završava unos ili odustaje od postavljanja

UC15 - Objavi oglas na društvene mreže

- **Glavni sudionik:** Iznajmljivač
- **Cilj:** Objaviti oglas za iznajmljivanje romobila na odabranu društvenu mrežu
- **Sudionici:** Baza podataka
- **Preduvjet:** Iznajmljivač je prijavljen, ima barem jedan aktivan oglas
- **Opis osnovnog tijeka:**
  1. Iznajmljivač odabire opciju „Objavi na društvenu mrežu“
  2. Iznajmljivač odabire društvenu mrežu na koju bi htio objaviti svoj oglas
  3. Generira se objava za iznajmljivanje romobila
  4. Iznajmljivač potvrđuje objavu
  5. Oglas se objavljuje na društvenoj mreži

UC16 - Pregledaj svoj oglas

- **Glavni sudionik:** Iznajmljivač
- **Cilj:** Pregledati detaljne informacije o svom oglasu
- **Sudionici:** Baza podataka
- **Preduvjet:** Iznajmljivač je prijavljen, ima barem jedan aktivan oglas
- **Opis osnovnog tijeka:**
  1. Prikazuje se stranica s iznajmljivačevim oglašenim romobilima
  2. Iznajmljivač pronalazi i odabire željeni oglas
  3. Iznajmljivaču se prikazuju detaljnije informacije o oglasu

UC17 - Uredi oglas

- **Glavni sudionik:** Iznajmljivač
- **Cilj:** Promijeniti informacije o svom oglasu
- **Sudionici:** Baza podataka
- **Preduvjet:** Iznajmljivač je prijavljen, ima barem jedan aktivan oglas
- **Opis osnovnog tijeka:**
  1. Prikazuje se stranica s iznajmljivačevim oglašenim romobilima
  2. Iznajmljivač pronalazi i odabire željeni oglas i opciju „Uredi“
  3. Iznajmljivač mijenja informacije o uvjetima iznajmljivanja
  4. Iznajmljivač potvrđuje promjene odabirom opcije „Spremi promjene“
  5. Baza podataka se ažurira
- **Opis mogućih odstupanja:**

  3.a Unos podataka o oglasu u nedozvoljenom formatu
    1. Iznajmljivač dobiva odgovarajuću obavijest o neispravnosti podataka
    2. Iznajmljivač mijenja potrebne podatke i završava s unosom ili odustaje od oglašavanja romobila

  4.a Korisnik ne potvrdi promjenu odabirom opcije „Spremi promjene“
    1. Korisnik dobiva obavijest da nije spremio podatke prije izlaska iz prozora za zamjenu podataka
    2. Korisnik sprema podatke

<u>UC18 - Izbriši oglas</u>

- **Glavni sudionik:** Iznajmljivač, administrator
- **Cilj:** Izbrisati oglas o romobilu s popisa oglašenih romobila
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen, ako je korisnik iznajmljivač mora imati barem jedan svoj aktivan oglas, a ako je korisnik administrator mora postojati barem jedan aktivan oglas bilo kojeg iznajmljivača
- **Opis osnovnog tijeka:**
  1. Korisnik pronalazi oglas kojeg želi izbrisati
  2. Korisnik odabire opciju „Izbriši oglas“
  3. Oglas o romobilu se miče iz pregleda dostupnih oglasa
  4. Ažurira se baza podataka

<u>UC19 - Iznajmi romobil</u>

- **Glavni sudionik:** Klijent

- **Cilj:** Iznajmiti romobil od iznajmljivača
- **Sudionici:** Iznajmljivač, baza podataka
- **Preduvjet:** Klijent je prijavljen, postoji barem jedan aktivan oglas
- **Opis osnovnog tijeka:**
  1. Klijent na oglasu o romobilu odabire opciju „Iznajmi romobil“
  2. Iznajmljivaču se šalje poruka sa zahtjevom za iznajmljivanje

UC20 - Reagiraj na zahtjev za iznajmljivanje

- **Glavni sudionik:** Iznajmljivač
- **Cilj:** Prihvatiti ili odbiti zahtjev klijenta za iznajmljivanje romobila
- **Sudionici:** Klijent, baza podataka
- **Preduvjet:** Postoji zahtjev za iznajmljivanje na kojeg iznajmljivač još nije reagirao
- **Opis osnovnog tijeka:**
  1. Iznajmljivač pregledava zahtjev za iznajmljivanje
  2. Iznajmljivač prihvaća ili odbija ponudu
  3. Klijentu se šalje obavijest o odluci
  4. Ako je ponuda prihvaćena, oglas o romobilu se miče iz dostupnih romobila

UC21 - Zamijeni sliku romobila

- **Glavni sudionik:** Klijent
- **Cilj:** Zamijeniti sliku romobila kako bi prikazivala trenutno stanje romobila
- **Sudionici:** Iznajmljivač, baza podataka
- **Preduvjet:** Klijent je prijavljen, trenutno iznajmljuje romobil
- **Opis osnovnog tijeka:**
  1. Klijent pregledava uvjete trenutno aktivnog najma
  2. Klijent odabire opciju „Zamijeni sliku“
  3. Klijent postavlja novu sliku romobila i opis što je drugačije na slici
  4. Šalje se obavijest iznajmljivaču da je došlo do zamjene slike
  5. Ažurira se baza podataka
- **Opis mogućih odstupanja:**

  3.a Unos slike u nedozvoljenom formatu
    1. Korisnik dobiva odgovarajuću obavijest o neispravnosti formata slike
    2. Korisnik ispravno unosi sliku i završava s unosom ili odustaje od zamjene slike

UC22 - Prijavi zamjenu slike

- **Glavni sudionik:** Iznajmljivač
- **Cilj:** Poslati žalbu na zamjenu slike romobila koju je obavio klijent
- **Sudionici:** Administrator, baza podataka
- **Preduvjet:** Iznajmljivač je prijavljen, klijent napravio zamjenu slike romobila
- **Opis osnovnog tijeka:**
  1. Iznajmljivač pregledava zamjenu slike romobila
  2. Iznajmljivač šalje žalbu administratoru

UC23 - Odluči o zamjeni slike

- **Glavni sudionik:** Administrator
- **Cilj:** Odlučiti poništava li se zamjena slike ili ne te obavijestiti korisnike o toj odluci
- **Sudionici:** Klijent, iznajmljivač, baza podataka
- **Preduvjet:** Iznajmljivač se žalio na zamjenu slike
- **Opis osnovnog tijeka:**
  1. Administrator pregledava žalbu
  2. Administrator odabire sliku koju želi pohraniti u bazu podataka
  3. Baza podataka se ažurira
  4. Šalju se odgovarajuće obavijesti o odluci klijentu i iznajmljivaču

UC24 - Vrati romobil

- **Glavni sudionik:** Klijent
- **Cilj:** Vratiti romobil iznajmljivaču kako bi iznajmljivanje završilo i zabilježilo se
- **Sudionici:** Iznajmljivač, baza podataka
- **Preduvjet:** Klijent je prijavljen, trenutno iznajmljuje romobil
- **Opis osnovnog tijeka:**
  1. Klijent u aplikaciji potvrđuje da je vratio romobil čime iznajmljivanje završava
  2. Provjerava se da je romobil vraćen u pravo vrijeme
  3. Izračunava se cijena koju klijent mora platiti iznajmljivaču
  4. Klijent i iznajmljivač dobivaju obavijest da je iznajmljivanje završeno i cijenu iznajmljivanja
  5. Podatci o transakciji se spremaju u bazu podataka

- **Opis mogućih odstupanja:**
  2.a Romobil je vraćen prekasno
    1. U ukupnu cijenu iznajmljivanja dodaje se iznos novčane kazne za prekasno vraćanje romobila

UC25 - Ocijeni klijenta

- **Glavni sudionik:** Iznajmljivač
- **Cilj:** Ocijeniti klijenta po završetku iznajmljivanja romobila
- **Sudionici:** Klijent, baza podataka
- **Preduvjet:** Klijent je vratio romobil, iznajmljivanje završeno
- **Opis osnovnog tijeka:**
  1. Iznajmljivač ocjenjuje klijenta i ostavlja komentar o klijentu
  2. Recenzija se prikazuje na profilu klijenta
  3. Ažurira se baza podataka

UC26 - Isplati s računa

- **Glavni sudionik:** Banka
- **Cilj:** Smanjiti iznos novaca na računu klijenta za iznos cijene iznajmljivanja
- **Sudionici:** Baza podataka
- **Preduvjet:** Klijent je vratio romobil, iznajmljivanje je završeno
- **Opis osnovnog tijeka:**
  1. Po završetku iznajmljivanja dogovoreni iznos se skida s računa klijenta u banci
  2. Banka prima iznos cijene iznajmljivanja

UC27 - Uplati na račun

- **Glavni sudionik:** Banka
- **Cilj:** Povećati iznos novaca na računu iznajmljivača za iznos cijene iznajmljivanja
- **Sudionici:** Baza podataka
- **Preduvjet:** Klijent je vratio romobil, iznajmljivanje je završeno
- **Opis osnovnog tijeka:**
  1. Po završetku iznajmljivanja dogovoreni iznos se stavlja na račun iznajmljivača u banci

UC28 - Pregledaj transakcije

- **Glavni sudionik:** Klijent
- **Cilj:** Dobiti uvid u sve prethodne transakcije gdje je klijent iznajmljivao romobil
- **Sudionici:** Baza podataka
- **Preduvjet:** Klijent je prijavljen
- **Opis osnovnog tijeka:**
  1. Klijent odabire opciju „Moje transakcije“
  2. Klijent dobiva pregled svih njegovih prethodnih transakcija

UC29 - Pregledaj profil korisnika

- **Glavni sudionik:** Klijent, administrator
- **Cilj:** Pregledati informacije o drugom korisniku, njegovu ocjenu i recenzije o njemu
- **Sudionici:** Baza podataka
- **Preduvjet:** Klijent/administrator je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik pronalazi korisnika čiji profil želi pregledati
  2. Korisnik odlazi na profil tog korisnika
  3. Korisnik pregledava profil drugog korisnika, njegove javno dostupne podatke, ocjene i recenzije

UC30 - Izbriši recenziju

- **Glavni sudionik:** Iznajmljivač, administrator
- **Cilj:**Izbrisati recenziju o drugom korisniku
- **Sudionici:**Baza podataka
- **Preduvjet:**Ako je korisnik iznajmljivač, mora postojati recenzija na profilu klijenta koju je iznajmljivač napisao i koju želi izbrisati. Administrator može obrisati bilo koju recenziju.
- **Opis osnovnog tijeka:**
  1. Korisnik pronalazi profil drugog korisnika
  2. Korisnik odabire opciju „Izbriši recenziju“
  3. Recenzija se briše s profila korisnika na kojem je bila objavljena

UC31 - Pregledaj razgovor

- **Glavni sudionik:** Klijent
- **Cilj:**Pregledati razgovor s nekim drugim korisnikom

- **Sudionici:**Baza podataka
- **Preduvjet:**Klijent je prijavljen
- **Opis osnovnog tijeka:**
  1. Klijent pronalazi razgovor kojeg želi pregledati
  2. Klijent ulazi u razgovor i pregledava ga

UC32 - Pošalji poruku

- **Glavni sudionik:** Klijent
- **Cilj:** Poslati poruku drugom korisniku
- **Sudionici:** Baza podataka
- **Preduvjet:** Klijent je prijavljen
- **Opis osnovnog tijeka:**
  1. Klijent pronalazi razgovor s korisnikom kojem želi poslati poruku
  2. Klijent upisuje poruku i šalje ju drugom korisniku
- **Opis mogućih odstupanja:**

  3.a Korisnik unio poruku, ali ju nije poslao prije izlaska iz razgovora
    1. Poruka se briše i ne šalje

UC33 - Izbriši razgovor

- **Glavni sudionik:** Klijent
- **Cilj:** Izbrisati razgovor s drugim korisnikom
- **Sudionici:** Baza podataka
- **Preduvjet:** Klijent je prijavljen
- **Opis osnovnog tijeka:**
  1. Klijent pronalazi razgovor s korisnikom koji želi obrisati
  2. Klijent odabire opciju „Izbriši razgovor”
  3. Razgovor se briše iz pregleda razgovora kod oba korisnika
  4. Ažurira se baza podataka

UC34 - Blokiraj korisnika

- **Glavni sudionik:** Administrator
- **Cilj:** Korisniku zabraniti pristup aplikaciji
- **Sudionici:** Baza podataka
- **Preduvjet:** Administrator je prijavljen
- **Opis osnovnog tijeka:**
  1. Administrator pregledava profile prijavljenih korisnika

2. Administrator odabire opciju „Blokiraj korisnika” kod odabranog korisnika
3. U bazu podataka se zapisuje da je korisnik blokiran i pri sljedećoj prijavi bit će mu onemogućen pristup sustavu

UC35 - Odblokiraj korisnika

- **Glavni sudionik:** Administrator
- **Cilj:** Ponovno omogućiti korisniku pristup sustavu
- **Sudionici:** Baza podataka
- **Preduvjet:** Administrator je prijavljen, postoji barem jedan blokirani korisnik
- **Opis osnovnog tijeka:**
  1. Administrator pregledava profile blokiranih korisnika
  2. Administrator odabire opciju „Odblokiraj korisnika” kod odabranog korisnika
  3. U bazu podataka se zapisuje da je korisnik odblokiran i pri sljedećoj prijavi bit će mu onemogućen pristup sustavu

UC36 - Dodaj administratora

- **Glavni sudionik:** Administrator
- **Cilj:** Dodijeliti drugom korisniku ulogu administratora
- **Sudionici:** Baza podataka
- **Preduvjet:** Administrator je prijavljen
- **Opis osnovnog tijeka:**
  1. Administrator pregledava profile prijavljenih korisnika
  2. Administrator odabire opciju „Dodaj administratora” kod odabranog korisnika
  3. Odabranom korisniku dodjeljuje se uloga administratora
  4. Ažurira se baza podataka

UC37 - Izbriši administratora

- **Glavni sudionik:** Administrator
- **Cilj:** Oduzeti drugom administratoru ulogu administratora
- **Sudionici:** Baza podataka
- **Preduvjet:** Administrator je prijavljen, postoji barem još jedan administrator osim njega

- **Opis osnovnog tijeka:**
  1. Administrator pregledava postojeće administratore
  2. Administrator odabire opciju „Izbriši administratora” kod odabranog administratora
  3. Odabranom korisniku oduzima se uloga administratora
  4. Ažurira se baza podataka

Dijagrami obrazaca uporabe

Slika 3.1: Dijagram obrasca uporabe - funkcionalnost iznajmljivača

Slika 3.2: Dijagram obrasca uporabe - funkcionalnost klijenta i banke

Slika 3.3: Dijagram obrasca uporabe - funkcionalnost neregistriranog korisnika i administratora

3.1.2 Sekvencijski dijagrami

Obrazac uporabe UC10 - Registriraj romobil

Klijent šalje zahtjev za prikaz stranice s njegovim registriranim romobilima. Poslužitelj iz baze dohvaća registrirane romobile tog klijenta. Ako klijent ima već registrirane romobile, poslužitelj ih prikazuje, inače poslužitelj prikazuje klijentu poruku da nema registriranih romobila. Klijent započinje registraciju romobila tako što zatraži od poslužitelja da mu prikaže obrazac za registraciju. Poslužitelj mu prikazuje prazan obrazac i klijent unosi podatke za registraciju te ih vraća poslužitelju. Poslužitelj provjerava jesu li podatci ispravno uneseni. Sve dok klijent ne unese ispravne podatke ispisuje mu se poruka da su podatci neispravni, unosi nove podatke koje zatim provjerava poslužitelj i vraća informaciju jesu li podatci ispravni ili ne. Kada klijent unese ispravne podatke za registraciju, potvrđuje svoj unos i poslužitelj te podatke prosljeđuje bazi koja pohranjuje registraciju. Romobil se nakon registracije prikazuje među klijentovim registriranim romobilima. Ako klijent do sada nije imao ulogu iznajmljivača, sada mu se dodjeljuje.

Slika 3.4: Sekvencijski dijagram obrasca UC10 - Registriraj romobil

Obrazac uporabe UC14 - Oglasi romobil

Iznajmljivač šalje zahtjev za prikaz stranice s njegovim registriranim romobilima. Poslužitelj iz baze dohvaća registrirane romobile tog iznajmljivača i prikazuje mu ih. Iznajmljivač započinje stvaranje oglasa tako što zatraži da mu poslužitelj pošalje obrazac koji mora ispuniti. Poslužitelj prikazuje iznajmljivaču obrazac i iznajmljivač unosi podatke o oglasu te ih vraća poslužitelju. Poslužitelj provjerava jesu li podatci ispravno uneseni. Sve dok iznajmljivač ne unese ispravne podatke ispisuje mu se poruka da su podatci neispravni, unosi nove podatke koje zatim provjerava poslužitelj i vraća informaciju jesu li podatci ispravni ili ne. Kada iznajmljivač unese ispravne podatke za registraciju, potvrđuje svoj unos i poslužitelj te podatke prosljeđuje bazi koja pohranjuje oglas. Iznajmljivač dobiva poruku da je oglas uspješno postavljen. Oglas se prikazuje među oglasima dostupnim za iznajmljivanje.

Slika 3.5: Sekvencijski dijagram obrasca UC14 - Oglasi romobil

Obrazac uporabe UC20 - Reagiraj na zahtjev za iznajmljivanje

Iznajmljivač inicira zahtjev za pregled trenutačnih zahtjeva za najam kako bi odgovorio na klijentovu želju za najmom romobila. Poslužitelj preuzima trenutne zahtjeve koje je iznajmljivač poslao i prikazuje ih. Iznajmljivač zatim donosi odluku o prihvaćanju ili odbijanju zahtjeva, a svoj odabir označava u aplikaciji. Poslužitelj prosljeđuje tu informaciju bazi podataka kako bi se ažurirao status zahtjeva. U slučaju prihvaćanja zahtjeva za najam, oglas za najam romobila se uklanja iz baze podataka, a informacija o nedostupnosti romobila se pohranjuje. Nakon što se promjene spreme u bazu podataka, sustav obavještava klijenta o odluci iznajmljivača putem aplikacijske obavijesti.

Slika 3.6: Sekvencijski dijagram obrasca UC20 - Reagiraj na zahtjev za iznajmljivanje

Obrazac uporabe UC24 - Vrati romobil

Pri povratku romobila na kraju iznajmljivanja klijent zatraži od poslužitelja sve svoje aktivne najmove. Baza podataka pronalazi aktivne najmove tog klijenta i vraća ih poslužitelju. Ako klijent nema nijedan aktivan najam poslužitelj mu vraća poruku koja ga o tome obavještava. U slučaju da aktivni najmovi postoje, poslužitelj ih vraća klijentu. Klijent odabire za koji romobil želi potvrditi da je vraćen i šalje poslužitelju zahtjev da završi iznajmljivanje. Poslužitelj zabilježava kraj u bazi podataka i izračunava broj prijeđenih kilometara i cijenu najma. Informacije o transakciji za taj najam spremaju se u bazu podataka. Poslužitelj prikazujje obavijest o cijeni najma iznajmljivaču i klijentu koji su u najmu sudjelovali.

Slika 3.7: Sekvencijski dijagram obrasca UC24 - Vrati romobil

3.2 Ostali zahtjevi

- Aplikacija mora se moći pokrenuti na svakome web-pregledniku
- Aplikacija mora biti javno dostupna
- Sustav mora omogućiti rad više korisnika u stvarnom vremenu
- Korisničko sučelje i sustav moraju podržavati hrvatsku abecedu (dijakritičke znakove)
- Neispravno korištenje korisničkog sučelja ne smije narušiti funkcionalnost i rad sustava
- Službena valuta sustavu je EURO (€)
- Sustav treba biti jednostavan za korištenje
- Sustav mora biti ostvaren koristeći objektno orijentirane jezike

4. Arhitektura i dizajn sustava

4.0.1 Opis arhitekture

Detaljnom razradom cilja projektnog zadatka, u kojem je fokus izrada aplikacije za iznajmljivanje električnih romobila, definirali smo razinu klijenta, razinu web-aplikacije te sloj pristupa podatcima kao osnovne razine naše aplikacije.

Razina klijenta

Razina klijenta predstavlja korisničko sučelje web-aplikacije koje korisnici vide i s kojim interagiraju. U razvoju projekta korišten je React, odnosno JavaScript knjižica za izradu korisničkog sučelja. Organizirano je u komponente koje predstavljaju određene dijelove korisničkog sučelja. Korišten je virtualni DOM (Document Object Model), kojim se ubrzava proces ažuriranja promjena korisničkog sučelja u svrhu poboljšavanja performansi web-aplikacije.

Razina web-aplikacije

Sloj web-aplikacije je odgovoran za obradu zahtjeva korisnika, poslovnu logiku i komunikaciju s bazom podataka. U razvoju projekta korišten je okvir za razvoj web aplikacija Spring Boot u programskom jeziku Java. U Spring Bootu, kontroleri su odgovorni za obradu HTTP zahtjeva i usmjeravanje na odgovarajuće servise za obradu zahtjeva. Obradu podataka, validaciju te logiku obavljaju servisi, dok modeli predstavljaju strukturu podataka koja se koristi za komunkaciju s bazom podataka te prenošenje podataka između kontrolera i servisa.

Sloj pristupa podatcima

Sloj pristupa podacima je odgovoran za komunikaciju s bazom podataka i pristupanje podacima. Građen je od entiteta s vlastitim atributima koji predstavljaju modele podataka koji odgovaraju tablicama u bazi podataka.

Sinteza ovih slojeva - korisničkog sučelja na razini Reacta, web-aplikacijskog sloja u Spring Bootu i sloja pristupa podacima u Spring Bootu - stvara temelj za razvoj visoko skalabilnih i funkcionalnih web-aplikacija. Korisnici ostvaruju interakciju s aplikacijom preko intuitivnog React korisničkog sučelja, dok Spring Boot preuzima odgovornost za obradu njihovih zahtjeva i poslovne logike. Istovremeno, sloj pristupa podacima omogućuje efikasnu komunikaciju s bazom podataka, omogućujući pohranu i dohvat podataka s pouzdanošću i učinkovitošću.

4.0.2 MVC arhitektura

Model-View-Controller (MVC) je arhitekturni obrazac koji se koristi za organizaciju komponenti u softverskim aplikacijama, posebno u razvoju web-aplikacija. Osnovna svrha MVC-a je odvajanje različitih aspekata aplikacije kako bi se omogućila bolja organizacija, održavanje i skalabilnost. Sastoji se od tri glavne komponente:

- **Model** - predstavlja sloj koji je odgovoran za obradu podataka i poslovnu logiku aplikacije te sadrži podatke i pravila za njihovu obradu.
- **View** - predstavlja sloj koji se odnosi na korisničko sučelje aplikacije i odgovoran je za prikazivanje podataka korisnicima. Ne obavlja nikakvu poslovnu logiku, samo prikazuje podatke koji mu se dostave iz modela.
- **Kontroler** - posrednik između Model i View komponenti. Prima korisničke zahtjeve, obrađuje ih te komunicira s Modelom radi dohvaćanja ili ažuriranja podataka. Također, odlučuje koji View treba biti prikazan korisniku na temelju podataka iz Modela te korisničkih zahtjeva, upravlja tokom aplikacije te sadrži logiku za validaciju, autorizaciju i upravljanje sesijama.

MVC arhitektura omogućuje precizno razgraničenje odgovornosti unutar aplikacije. Ovo strukturalno odvajanje olakšava razvoj aplikacije, čini ju lakšom za održavanje i omogućava efikasnije testiranje. Svaka od tri glavne komponente - Model, View i Controller - može se ponovno koristiti na različitim dijelovima aplikacije. To potiče efikasnost razvoja jer se već razvijeni dijelovi aplikacije mogu lako iskoristiti u drugim kontekstima. MVC omogućuje skalabilnost aplikacije jer se jasno razdvajaju različiti aspekti. Novi dijelovi funkcionalnosti mogu se dodavati bez narušavanja postojeće arhitekture, što omogućava aplikaciji da raste i prilagodi se promjenama.

Slika 4.1: Prikaz MVC obrasca

4.1 Baza podataka

U kontekstu našeg sustava, baza podataka igra ključnu ulogu, pružajući strukturiranu platformu za modeliranje stvarnog svijeta. Temeljni građevni blok ove baze je relacija, odnosno tablica koja je jasno definirana svojim imenom i skupom atributa.

Glavna svrha baze podataka je olakšati brzu i jednostavnu pohranu, promjenu te izvlačenje podataka kako bi se omogućila daljnja analiza i obrada. Unutar baze podataka za našu aplikaciju, identificiramo nekoliko ključnih entiteta:

- *User*
- *Preferences*
- *SocialMedia*
- *PrivacySettings*
- *Document*
- *Scooter*
- *Listing*
- *Review*
- *Transaction*
- *Invoice*
- *Notification*
- *ChatSession*
- *Message*
- *ImageChangeRequest*

4.1.1 User

Ovaj entitet sadrzava sve važne informacije o korisniku aplikacije. Sadrži atribute: userId, nickname, firstName, lastName, cardNumber, email, phoneNumber, password, role te status. Ovaj entitet ima One-to-one vezu s entitetom Preferences preko atributa userId, One-to-one vezu s entitetom PrivacySettings preko atributa userId, One-to-one vezu s entitetom SocialMedia preko atributa userId, One-to-many vezu s entitetom Documents preko atributa userId, One-to-many vezu s entitetom Scooters preko atributa userId, One-to-many vezu s entitetom Listings preko atributa renterUsername, One-to-many vezu s entitetom Reviews preko atributa reviewerUsername, One-to-many vezu s entitetom ChatSessions preko atributa user1 ili atributa user2, One-to-many vezu

s entitetom Messages preko atributa senderUsername, One-to-many vezu s entitetom ImageChangeRequest preko atributa requesterId te One-to-many vezu s entitetom Notifications preko atributa userId, requestingUser te decisionAdmin.

User		
userId	INT	jedinstveni identifikator korisnika
nickname	VARCHAR	jedinstveni nadimak korisnika
firstName	VARCHAR	ime korisnika
lastName	VARCHAR	prezime korisnika
cardNumber	INT	broj kartice korisnika
email	VARCHAR	jedinstvena email adresa korisnika
phoneNumber	INT	jedinstveni kontakt broj korisnika
password	VARCHAR	zaporka za prijavu korisnika
role	UserRole	uloga korisnika ('unregistered', 'registered', 'renter', 'admin')
status	UserStatus	status korisnika ('pending', 'rejected', 'accepted', 'blocked')

4.1.2 Preferences

Ovaj entitet sadrzava sve važne informacije o preferencama korisnika aplikacije. Sadrži atribute: userId, language i darkMode. Ovaj entitet ima One-to-one vezu s entitetom User preko atributa userId,

Preferences		
userId	INT	jedinstveni identifikator korisnika
language	UserLanguage	jezik korisnika
darkMode	BOOLEAN	omogućen dark mode

4.1.3 Social Media

Ovaj entitet sadrzava sve važne informacije o socijalnim mrežama korisnika aplikacije. Sadrži atribute: userId, instagram, facebook, google i tikTok. Ovaj entitet ima One-to-

one vezu s entitetom User preko atributa userId.

Social Media		
userId	INT	jedinstveni identifikator korisnika
instagram	VARCHAR	instagram account korisnika
facebook	VARCHAR	facebook account korisnika
google	VARCHAR	google account korisnika
tikTok	VARCHAR	tikTok account korisnika

4.1.4 PrivacySettings

Ovaj entitet sadrzava sve važne informacije o postavkama privatnosti korisnika aplikacije. Sadrži atribute: userId, isFirstNameVisible, isLastNameVisible, isNicknameVisible i isEmailVisible. Ovaj entitet ima One-to-one vezu s entitetom User preko atributa userId.

Privacy Settings		
userId	INT	jedinstveni identifikator korisnika
isFirstNameVisible	BOOLEAN	omogućena vidljivost imena korisnika
isLastNameVisible	BOOLEAN	omogućena vidljivost prezimena korisnika
isNicknameVisible	BOOLEAN	omogućena vidljivost nadimka korisnika
isEmailVisible	BOOLEAN	omogućena vidljivost emaila korisnika

4.1.5 Document

Ovaj entitet sadrzava sve važne informacije o dokumentima korisnika aplikacije. Sadrži atribute: userId, documentType, url i status. Ovaj entitet ima One-to-one vezu s entitetom User preko atributa userId.

Document		
userId	INT	jedinstveni identifikator korisnika

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Document		
documentType	DocumentType	tip dokumenta ('criminalRecord', 'identification')
url	VARCHAR	url dokumenta
status	DocumentStatus	status dokumenta ('pending', 'approved', 'rejected')

4.1.6 Scooter

Ovaj entitet sadrzava sve važne informacije o pojedinom romobilu. Sadrži atribute:scooterId, manufacturer, model, batteryCapacity, maxSpeed, imageUrl, maxRange, yearOfManufacture, additionalInformation, userId i availability. Ovaj entitet ima One-to-one vezu s entitetom User preko atributa userId te One-to-many vezu s entitetom Listings preko atributa scooterId.

Scooter		
scooterId	INT	jedinstveni identifikator romobila
manufacturer	VARCHAR	proizvođač romobila
model	VARCHAR	model romobila
batteryCapacity	INT	kapacitet baterije
maxSpeed	INT	maksimalna brzina
imageUrl	TEXT	url slike
maxRange	FLOAT	maksimalni domet
yearOfManufacture	INT	godina proizvodnje
additionalInformation	TEXT	dodatne informacije
userId	INT	jedinstveni identifikator korisnika
availability	BOOLEAN	dostupnost

4.1.7 Listing

Ovaj entitet sadrzava sve važne informacije o pojedinom oglasu. Sadrži atribute:listingId, currentAddress, returnAddress, returnByTime, pricePerKilometer, penaltyFee, scoote-

rId, listingTime, notes i status. Ovaj entitet ima One-to-one vezu s entitetom Scooter preko atributa scooterId, One-to-many vezu s entitetom Reviews preko atributa transactionId te One-to-many vezu s entitetom Transactions preko atributa listingId.

Listing		
listingId	INT	jedinstveni identifikator oglasa
currentAddress	VARCHAR	trenutna adresa
returnAddress	VARCHAR	adresa povratka
returnByTime	TIMESTAMP	vrijeme vraćanja
pricePerKilometer	FLOAT	cijena po kilometru
penaltyFee	FLOAT	kaznena naknada
scooterId	INT	jedinstveni identifikator romobila
listingTime	TIMESTAMP	vrijeme objave oglasa
notes	TEXT	bilješke
status	ListingStatus	status oglasa ('active', 'finished', 'cancelled')

4.1.8 Review

Ovaj entitet sadrzava sve važne informacije o pojedinom osvrtu. Sadrži atribute: transactionId, reviewerUsername, renterUsername, stars, comment te reviewTime. Ovaj entitet ima Many-to-one vezu s entitetom User preko atributa reviewerUsername, Many-to-one vezu s entitetom User preko atributa renterUsername te One-to-one vezu s entitetom Transaction preko atributa transactionId.

Review		
transactionId	INT	jedinstveni identifikator transakcije
reviewerUsernamе	VARCHAR	korisničko ime recenzenta
renterUsername	VARCHAR	korisničko ime iznajmljivača
stars	INT	broj zvjezdica/ocjena
comment	TEXT	komentar
reviewTime	TIMESTAMP	vrijeme osvrta

4.1.9 Transaction

Ovaj entitet sadrzava sve važne informacije o pojedinoj transakciji. Sadrži atribute: transactionId, kilometersTraveled, totalPrice, listingId, paymentTime te previousTransactionStatus. Ovaj entitet ima One-to-one vezu s entitetom Listing preko atributa listingId te One-to-one vezu s entitetom Invoice preko atributa transactionId.

Transaction		
transactionId	INT	jedinstveni identifikator transakcije
kilometersTraveled	FLOAT	broj prijeđenih kilometara
totalPrice	FLOAT	ukupna cijena
listingId	INT	jedinstveni identifikator oglasa
paymentTime	TIMESTAMP	vrijeme plaćanja
transactionStatus	TransactionStatus	status transakcije

4.1.10 Invoice

Ovaj entitet sadrzava sve važne informacije o pojedinoj dostavnici. Sadrži atribute:transactionId, invoiceNumber te paymentMethod. Ovaj entitet ima One-to-one vezu s entitetom Transaction preko atributa transactionId.

Invoice		
transactionId	INT	jedinstveni identifikator transakcije
invoiceNumber	INT	broj fakture
paymentMethod	PaymentMethod	način plaćanja ('PayPal', 'kekspay', 'Revolut')

4.1.11 Notification

Ovaj entitet sadrzava sve važne informacije o pojedinoj obavijesti. Sadrži atribute: notificationId, userId, requestingUser, decisionAdmin, content, isRead te sentTime. Ovaj entitet ima Many-to-one vezu s entitetom User preko atributa userId, Many-to-one vezu s entitetom User preko atributa requestingUser te Many-to-one vezu s entitetom User preko atributa decisionAdmin.

Notification		
notificationId	INT	jedinstveni identifikator obavijesti
userId	INT	jedinstveni identifikator korisnika
requestingUser	INT	korisnik koji zahtjeva
decisionAdmin	INT	admin za odluku
content	TEXT	sadržaj
isRead	BOOLEAN	pročitanost
sentTime	TIMESTAMP	vrijeme slanja

4.1.12 ChatSession

Ovaj entitet sadrzava sve važne informacije o pojedinom razgovoru. Sadrži atribute: chatId, user1, user2, startCommunicationTime te lastMessageTime. Ovaj entitet ima Many-to-one vezu s entitetom User preko atributa user1, Many-to-one vezu s entitetom User preko atributa user2 te One-to-many vezu s entitetom Messages preko atributa sessionId.

ChatSession		
chatId	INT	jedinstveni identifikator razgovora
user1	INT	korisnik 1
user2	INT	korisnik 2
startCommunicationTime	TIMESTAMP	vrijeme početka komunikacije
lastMessageTime	TIMESTAMP	vrijeme zadnje poslane poruke

4.1.13 Message

Ovaj entitet sadrzava sve važne informacije o pojedinoj poruci. Sadrži atribute:messageId, senderUsername, sessionId, text, sentTime te status. Ovaj entitet ima One-to-one vezu s entitetom User preko atributa senderUsername te Many-to-one vezu s entitetom ChatSession preko atributa sessionId.

Message		
messageId	INT	jedinstveni identifikator poruke
senderUsername	VARCHAR	nadimak pošiljatelja
sessionId	INT	jedinstveni identifikator razgovora
text	TEXT	tekst
sentTime	TIMESTAMP	vrijeme slanja
status	MessageStatus	status poruke ('read', 'unread')

4.1.14 ImageChangeRequest

Ovaj entitet sadrzava sve važne informacije o zahtjevu za promjenom slike. Sadrži atribute: imageId, requesterId, listingId, newImageUrl, complaintTime, additionalComments, status, approvalTime te rejectionReason. Ovaj entitet ima One-to-one vezu s entitetom User preko atributa requesterId te One-to-one vezu s entitetom Listing preko atributa listingId.

ImageChangeRequest		
imageId	INT	jedinstveni identifikator slike
requesterId	INT	jedinstveni identifikator pošiljatelja
listingId	INT	jedinstveni identifikator oglasa
newImageUrl	VARCHAR	url nove slike
complaintTime	TIMESTAMP	vrijeme žalbe
additionalComments	TEXT	dodatni komentari
status	ImageChangeRequestStatus	status zahtjeva ('approved', 'rejected', 'pending')
approvalTime	TIMESTAMP	vrijeme odobrenja
rejectionReason	TEXT	razlog odbitka

4.1.15 User

Ovaj entitet sadrzava sve važne informacije o korisniku aplikacije. Sadrži atribute: userId, nickname, firstName, lastName, cardNumber, email, phoneNumber, pa-

ssword, role te status. Ovaj entitet ima *One-to-one* vezu s entitetom Preferences preko atributa userId, *One-to-one* vezu s entitetom PrivacySettings preko atributa userId, *One-to-one* vezu s entitetom SocialMedia preko atributa userId, *One-to-one* vezu s entitetom Document preko atributa userId, *One-to-many* vezu s entitetom Scooter preko atributa userId, *One-to-many* vezu s entitetom Listing preko atributa renterUsername, *One-to-one* vezu s entitetom Review preko atributa reviewerUsername, *One-to-one* vezu s entitetom Review preko atributa renterUsername, *One-to-many* vezu s entitetom ChatSessions preko atributa user1 ili atributa user2, *Many-to-one* vezu s entitetom Messages preko atributa senderUsername, *One-to-many* vezu s entitetom ImageChangeRequest preko atributa requesterId te *One-to-many* vezu s entitetom Notification preko atributa userId, requestingUser te decisionAdmin.

User		
userId	INT	jedinstveni identifikator korisnika
nickname	VARCHAR	jedinstveni nadimak korisnika
firstName	VARCHAR	ime korisnika
lastName	VARCHAR	prezime korisnika
cardNumber	INT	broj kartice korisnika
email	VARCHAR	jedinstvena email adresa korisnika
phoneNumber	INT	jedinstveni kontakt broj korisnika
password	VARCHAR	zaporka za prijavu korisnika
role	UserRole	uloga korisnika (unregistered, registered, renter, admin)
status	UserStatus	status korisnika (pending, rejected, accepted, blocked)

4.1.16 Preferences

Ovaj entitet sadrzava sve važne informacije o preferencama korisnika aplikacije. Sadrži atribute: userId, language i darkMode. Ovaj entitet ima *One-to-one* vezu s entitetom User preko atributa userId.

Preferences		
userId	INT	jedinstveni identifikator korisnika (Ref: User.userId)
language	UserLanguage	jezik korisnika
darkMode	BOOLEAN	omogućen dark mode

4.1.17 Social Media

Ovaj entitet sadrzava sve važne informacije o socijalnim mrežama korisnika aplikacije. Sadrži atribute: userId, instagram, facebook, google i tikTok. Ovaj entitet ima *One-to-one* vezu s entitetom User preko atributa userId.

Social Media		
userId	INT	jedinstveni identifikator korisnika (Ref: User.userId)
instagram	VARCHAR	instagram account korisnika
facebook	VARCHAR	facebook account korisnika
google	VARCHAR	google account korisnika
tikTok	VARCHAR	tikTok account korisnika

4.1.18 PrivacySettings

Ovaj entitet sadrzava sve važne informacije o postavkama privatnosti korisnika aplikacije. Sadrži atribute: userId, isFirstNameVisible, isLastNameVisible, isNicknameVisible i isEmailVisible. Ovaj entitet ima *One-to-one* vezu s entitetom User preko atributa userId.

Privacy Settings		
userId	INT	jedinstveni identifikator korisnika (Ref: User.userId)
isFirstNameVisible	BOOLEAN	omogućena vidljivost imena korisnika

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Privacy Settings		
isLastNameVisible	BOOLEAN	omogućena vidljivost prezimena korisnika
isNicknameVisible	BOOLEAN	omogućena vidljivost nadimka korisnika
isEmailVisible	BOOLEAN	omogućena vidljivost emaila korisnika

4.1.19 Document

Ovaj entitet sadrzava sve važne informacije o dokumentima korisnika aplikacije. Sadrži atribute: userId, criminalRecordURL, identificationDocumentURL i status. Ovaj entitet ima *One-to-one* vezu s entitetom User preko atributa userId.

Document		
userId	INT	jedinstveni identifikator korisnika (Ref: User.userId)
criminalRecordURL	VARCHAR	url kaznene evidencije
identification-DocumentURL	VARCHAR	url identifikacijskog dokumenta
status	DocumentStatus	status dokumenta (pending, approved, rejected)

4.1.20 Scooter

Ovaj entitet sadrzava sve važne informacije o pojedinom romobilu. Sadrži atribute: scooterId, manufacturer, model, batteryCapacity, maxSpeed, imageUrl, maxRange, yearOfManufacture, additionalInformation, userId i availability. Ovaj entitet ima *Many-to-one* vezu s entitetom User preko atributa userId te *One-to-many* vezu s entitetom Listings preko atributa scooterId.

Scooter		
scooterId	INT	jedinstveni identifikator romobila
manufacturer	VARCHAR	proizvođač romobila
model	VARCHAR	model romobila
batteryCapacity	INT	kapacitet baterije
maxSpeed	INT	maksimalna brzina
imageUrl	TEXT	url slike
maxRange	FLOAT	maksimalni domet
yearOfManufacture	INT	godina proizvodnje
additionalInformation	TEXT	dodatne informacije
userId	INT	jedinstveni identifikator korisnika (Ref: User.userId)
availability	BOOLEAN	dostupnost

4.1.21 Listing

Ovaj entitet sadrzava sve važne informacije o pojedinom oglasu. Sadrži atribute: listingId, currentAddress, returnAddress, returnByTime, pricePerKilometer, penaltyFee, scooterId, listingTime, notes i status. Ovaj entitet ima *Many-to-one* vezu s entitetom Scooter preko atributa scooterId, *Many-to-one* vezu s entitetom User preko atributa renterUsername te *One-to-many* vezu s entitetom Transactions preko atributa listingId.

Listing		
listingId	INT	jedinstveni identifikator oglasa
renterUsername	VARCHAR	naziv iznajmljivača
currentAddress	VARCHAR	trenutna adresa
returnAddress	VARCHAR	adresa povratka
returnByTime	TIMESTAMP	vrijeme vraćanja
pricePerKilometer	FLOAT	cijena po kilometru

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Listing		
penaltyFee	FLOAT	kaznena naknada
scooterId	INT	jedinstveni identifikator romobila (Ref: Scooter.scooterId)
listingTime	TIMESTAMP	vrijeme objave oglasa
notes	TEXT	bilješke
status	ListingStatus	status oglasa (active, finished, cancelled)

4.1.22 Review

Ovaj entitet sadrzava sve važne informacije o pojedinom osvrtu. Sadrži atribute: transactionId, reviewerUsername, renterUsername, stars, comment te reviewTime. Ovaj entitet ima *One-to-one* vezu s entitetom User preko atributa reviewerUsername, *One-to-one* vezu s entitetom User preko atributa renterUsername te *One-to-one* vezu s entitetom Transaction preko atributa transactionId.

Review		
transactionId	INT	jedinstveni identifikator transakcije (Ref: Transaction.transactionId)
reviewerUsername	VARCHAR	korisničko ime recenzenta (Ref: User.nickname)
renterUsername	VARCHAR	korisničko ime iznajmljivača (Ref: User.nickname)
stars	INT	broj zvjezdica/ocjena
comment	TEXT	komentar
reviewTime	TIMESTAMP	vrijeme osvrta

4.1.23 Transaction

Ovaj entitet sadrzava sve važne informacije o pojedinoj transakciji. Sadrži atribute: transactionId, kilometersTraveled, totalPrice, listingId, paymentTime te previousTransactionStatus. Ovaj entitet ima *Many-to-one* vezu s entitetom Listing preko atributa listingId te *One-to-one* vezu s entitetom Invoice preko atributa transactionId.

Transaction		
transactionId	INT	jedinstveni identifikator transakcije
kilometersTraveled	FLOAT	broj prijeđenih kilometara
totalPrice	FLOAT	ukupna cijena
listingId	INT	jedinstveni identifikator oglasa (Ref: Listing.listingId)
paymentTime	TIMESTAMP	vrijeme plaćanja
transactionStatus	Transaction-Status	status transakcije

4.1.24 Invoice

Ovaj entitet sadrzava sve važne informacije o pojedinoj dostavnici. Sadrži atribute: transactionId, invoiceNumber te paymentMethod. Ovaj entitet ima *One-to-one* vezu s entitetom Transaction preko atributa transactionId.

Invoice		
transactionId	INT	jedinstveni identifikator transakcije (Ref: Transaction.transactionId)
invoiceNumber	INT	broj fakture
paymentMethod	PaymentMethod	način plaćanja (PayPal, kekspay, Revolut)

4.1.25 Notification

Ovaj entitet sadrzava sve važne informacije o pojedinoj obavijesti. Sadrži atribute: notificationId, userId, requestingUser, decisionAdmin, content, isRead te sentTime. Ovaj entitet ima *Many-to-one* vezu s entitetom User preko atributa userId, *Many-to-one* vezu s entitetom User preko atributa requestingUser te *Many-to-one* vezu s entitetom User preko atributa decisionAdmin.

Notification		
notificationId	INT	jedinstveni identifikator obavijesti
userId	INT	jedinstveni identifikator korisnika (Ref: User.userId)
requestingUser	INT	korisnik koji zahtjeva (Ref: User.userId)
decisionAdmin	INT	admin za odluku (Ref: User.userId)
content	TEXT	sadržaj
isRead	BOOLEAN	pročitanost
sentTime	TIMESTAMP	vrijeme slanja

4.1.26 ChatSession

Ovaj entitet sadrzava sve važne informacije o pojedinom razgovoru. Sadrži atribute: chatId, user1, user2, startCommunicationTime te lastMessageTime. Ovaj entitet ima *Many-to-one* vezu s entitetom User preko atributa user1, *Many-to-one* vezu s entitetom User preko atributa user2 te *One-to-many* vezu s entitetom Messages preko atributa sessionId.

ChatSession		
chatId	INT	jedinstveni identifikator razgovora
user1	INT	korisnik 1 (Ref: User.userId)
user2	INT	korisnik 2 (Ref: User.userId)

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

ChatSession		
startCommunication-Time	TIMESTAMP	vrijeme početka komunikacije
lastMessageTime	TIMESTAMP	vrijeme zadnje poslane poruke

4.1.27 Message

Ovaj entitet sadrzava sve važne informacije o pojedinoj poruci. Sadrži atribute: messageId, senderUsername, sessionId, text, sentTime te status. Ovaj entitet ima *One-to-many* vezu s entitetom User preko atributa senderUsername te *Many-to-one* vezu s entitetom ChatSession preko atributa sessionId.

Message		
messageId	INT	jedinstveni identifikator poruke
senderUsername	VARCHAR	nadimak pošiljatelja (Ref: User.nickname)
sessionId	INT	jedinstveni identifikator razgovora (Ref: ChatSession.chatId)
text	TEXT	tekst
sentTime	TIMESTAMP	vrijeme slanja
status	MessageStatus	status poruke (read, unread)

4.1.28 ImageChangeRequest

Ovaj entitet sadrzava sve važne informacije o zahtjevu za promjenom slike. Sadrži atribute: imageId, requesterId, listingId, newImageUrl, complaintTime, additionalComments, status, approvalTime te rejectionReason. Ovaj entitet ima *Many-to-one* vezu s entitetom User preko atributa requesterId te *Many-to-one* vezu s entitetom Listing preko atributa listingId.

ImageChangeRequest		
imageId	INT	jedinstveni identifikator slike
requesterId	INT	jedinstveni identifikator pošiljatelja (Ref: User.userId)
listingId	INT	jedinstveni identifikator oglasa (Ref: Listing.listingId)
newImageUrl	VARCHAR	url nove slike
complaintTime	TIMESTAMP	vrijeme žalbe
additionalComments	TEXT	dodatni komentari
status	ImageChange-RequestStatus	status zahtjeva (approved, rejected, pending)
approvalTime	TIMESTAMP	vrijeme odobrenja
rejectionReason	TEXT	razlog odbitka

4.1.29 Dijagram baze podataka

Slika 4.2: Prikaz dijagrama baze podataka

4.1.30 Dijagram razreda

Prvi isječak dijagrama razreda sadrži klasu User, enumeracije UserStatus i UserRole koje se nadovezuju na nju, klase PrivacySettings, Notification, SocialMedia, klasu Preference i odgovarajuću enumeraciju PreferenceUserLanguage te klasu Document povezanu s enumeracijom DocumentStatus.

Slika 4.3: Prikaz prvog isječka dijagrama razreda

Drugi isječak dijagrama razreda sadrži klase User, Scooter, klasu Listing s vlastitom enumeracijom ListingStatus, klasu ImageChangeRequest s vlastitom enumeracijom ImageChangeRequestStatus, klasu Review i klasu Transaction s vlastitom enumeracijom TransactionStatus.

Slika 4.4: Prikaz drugog isječka dijagrama razreda

Treći isječak dijagrama razreda sadrži klasu User, ChatSession povezanu s klasom Message koja ima dvije vlastite enumeracije, MessageType i MessageStatus.

Slika 4.5: Prikaz trećeg isječka dijagrama razreda

Slika 4.6: Prikaz Controller dijagrama

Slika 4.7: Prikaz Service dijagrama

4.2 Dijagram stanja

Dijagram stanja prikazuju objekte, njihova stanja i događaje kojima objekti prelaze iz jednog stanja u drugo. Slika prikazuje dijagram stanja za prijavljenog korisnika. Korisniku se inicijalno prikazuje početna stranica s trenutno objavljenim oglasima, ali ima i opciju pregleda svojih romobila, poruka i transakcija klikom na određenu opciju u navigacijskoj traci. Korisnik klikom na oglas na početnoj stranici može pregledati detalje o njemu i unajmiti ga. Ako korisnik već unajmljuje neki romobil, onda mu se ti romobili prikazuju s opcijom za vraćanje romobila. Na stranici za prikaz korisnikovih romobila postoje opcije za registraciju novih romobila, nji-

hovo oglašavanje i pregled oglasa koje je korisnik već objavio. Klikom na ikonu korisničkog profila na navigacijskoj traci korisnik dobiva opciju pregleda, uređivanja i brisanja vlastitog profila.

Slika 4.8: Dijagram stanja - klijent

4.3 Dijagram aktivnosti

Dijagramom aktivnosti prikazan je tijek prijave registriranog korisnika u sustav i njegova objava novog oglasa. Pravilnom prijavom u sustav korisnik dobiva mogućnost pregleda svojih registriranih romobila. Ako korisnik nema registriranih romobila, mora ih prvo registrirati popunjavanjem forme za registraciju. Nakon unosa podataka o romobilu podatci se spremaju u bazu podataka. Da bi objavio oglas korisnik prvo mora unijeti podatke o oglasu i oni se spremaju u bazu podataka. Uspješnim oglašavanjem korisnika se preusmjerava na početnu stranicu.

Slika 4.9: Dijagram aktivnosti - oglašavanje romobila

4.4 Dijagram komponenti

Dijagram komponenti prikazuje arhitekturu sustava i način na koji se komponente međusobno povezuju. Na slici 4.10 prikazan je UML dijagram komponenti sustava. Iz vanjske komponente Web preglednika se putem HTTP Request sučelja šalju klijentski zahtjevi prema aplikaciji. Codeblaze aplikacija sadrži dvije unutarnje komponente Frontend i Backend. Komunikacija između njih ostvarena je preko sučelja REST API. Frontend dio aplikacije izgrađen je pomoću React biblioteke, a sastoji se od React view koji služi za prikaz sučelja, React routera koji služi za upravljanje navigacijom i App.js koji služi kao središnja točka spajanja svih frontend elemenata. Na backendu, Rest Controller prima i obrađuje zahtjeve te ih prosljeđuje odgovarajućoj servisnoj komponenti. DTO komponenta služi za prijenos podataka između slojeva backenda. Komponenta repozitorija, koji koristi JPA za interakciju s bazom podataka, i domenske komponente, čine osnovu za upravljanje podacima. Preko sučelja PostgeSQL aplikacija se spaja na bazu podataka.

Slika 4.10: Dijagram komponenti

5. Implementacija i korisničko sučelje

5.1 Korištene tehnologije i alati

Timsku komunikaciju smo uspješno ostvarili kroz aplikaciju Discord, pružajući jednostavnu i iznimno organiziranu interakciju putem "channel" i "thread" opcija. Za izradu UML dijagrama smo koristili alat Astah UML, dok smo za upravljanje izvornim kodom primijenili GitHub.

Kao razvojno okruženje odabrali smo IntelliJ IDEA, integrirano razvojno okruženje (IDE) tvrtke JetBrains. Ovo okruženje pruža napredne mogućnosti kao što su dovršavanje koda analizom konteksta, navigacija koda s mogućnošću izravnog skakanja na klasu ili deklaraciju u kodu, refaktoriranje koda, rješavanje pogrešaka te ugrađene naredbe za Git, uz mnogobrojna proširenja za različite programske jezike i alate.

Na strani klijentske aplikacije (frontend), koristili smo React.js, popularnu JavaScript biblioteku za izradu korisničkih sučelja u web aplikacijama. Virtualna DOM tehnologija ovog okvira doprinosi efikasnijem ažuriranju sučelja, unaprjeđujući performanse i korisničko iskustvo.

Za poslužiteljsku stranu aplikacije (backend) odabrali smo Spring Framework, sveobuhvatan okvir za Java programski jezik. Poznat po konceptima inverzije kontrole i ubrizgavanja ovisnosti, Spring olakšava razvoj aplikacija pružajući modularnost, testabilnost te olakšava održavanje koda. Za upravljanje bazama podataka koristili smo PostgreSQL, moćan objektno-relacijski sustav otvorenog koda. Njegova pouzdanost, skalabilnost i podrška za kompleksne upite čine ga optimalnim izborom za naše potrebe.

5.2 Ispitivanje programskog rješenja

5.2.1 Ispitivanje komponenti

Provedbu ispitivanja implementiranih funkcionalnosti proveli smo na razini komponenti kroz detaljne unit testove. Fokusirali smo se na provjeru ispravnosti pomoćnih funkcija koje su ključne u različitim dijelovima koda. Ukupno smo izvršili sedam testova, a u nastavku donosimo sažete opise svakog testa zajedno s pripadajućim kodovima. Svaki opis testa popraćen je slikom koja jasno prikazuje rezultate izvođenja. Ovaj pristup omogućio nam je temeljitu analizu osnovne funkcionalnosti i identifikaciju rubnih uvjeta, pridonoseći sigurnosti i stabilnosti cijelog sustava. Nastojali smo osigurati sveobuhvatan pregled implementiranih funkcionalnosti kroz precizno definirane ispitne slučajeve.

Ispitni slučaj 1.: Test provjere svojstva dokumenta

Ulaz:

- Stvaranje primjerka korisnika (User) i postavljanje njegovih svojstava (ID i nadimak)
- Stvaranje primjerka dokumenta (Document) i postavljanje njegovih svojstava (ID korisnika, korisnik, status dokumenta).

Očekivani rezultati:

- Očekujemo da će dokument imati ispravno postavljene vrijednosti svojstava, u skladu s postavljenim ulaznim podacima.

Rezultat:

- Izvršavanje ovog dijela testa provjerava ispravnost postavljanja svojstava dokumenta i uspoređuje ih s očekivanim rezultatima.

```java
@Test
void testDocumentProperties() {
    // Create a sample user
    User user = new User();
    user.setUserId(1L);
    user.setNickname("testUser");

    Document document = new Document();
    document.setUserId(1L);
    document.setUser(user);
    document.setStatus(DocumentStatus.PENDING);

    assertEquals( expected: 1L, document.getUserId());
    assertEquals(user, document.getUser());
    assertEquals(DocumentStatus.PENDING, document.getStatus());
}
```

Slika 5.1: Test provjere svojstva dokumenta

Ispitni slučaj 2.: Test provjere funkcionalnosti ImageChangeRequestStatus

Ulaz:

- Nema posebnog ulaza jer testira vrijednosti unutar samog enuma

Očekivani rezultati:

- Očekujemo da će vrijednosti enuma biti ispravno postavljene.
- Očekujemo da će konverzija enuma u String biti ispravna.

Rezultat:

- Izvršavanje testa provjerava da li su vrijednosti enuma ImageChangeRequestStatus ispravno postavljene
- Izvršavanje testa provjerava da li su vrijednosti enuma ImageChangeRequestStatus ispravno postavljene

```java
@Test
void testEnumValues() {

    assertEquals( expected: "APPROVED", ImageChangeRequestStatus.APPROVED.getCode());
    assertEquals( expected: "REJECTED", ImageChangeRequestStatus.REJECTED.getCode());
    assertEquals( expected: "PENDING", ImageChangeRequestStatus.PENDING.getCode());

    assertEquals( expected: "Approved", ImageChangeRequestStatus.APPROVED.getDisplayValue());
    assertEquals( expected: "Rejected", ImageChangeRequestStatus.REJECTED.getDisplayValue());
    assertEquals( expected: "Pending", ImageChangeRequestStatus.PENDING.getDisplayValue());
}
```

Slika 5.2: Test provjere funkcionalnosti ImageChangeRequestStatus

```
@Test
void testEnumToString() {
    assertEquals( expected: "APPROVED", ImageChangeRequestStatus.APPROVED.toString());
    assertEquals( expected: "REJECTED", ImageChangeRequestStatus.REJECTED.toString());
    assertEquals( expected: "PENDING", ImageChangeRequestStatus.PENDING.toString());
}
```

Slika 5.3: Test provjere funkcionalnosti ImageChangeRequestStatus

Ispitni slučaj 3.: Provjera vraćanja vremena u razredu Listing

Ulaz:

- Stvaranje instance razreda Listing
- Postavljanje vremena povratka (returnByTime) na trenutno vrijeme

Očekivani rezultati:

- Očekujemo da će vrijeme povratka biti jednako trenutnom vremenu u obliku Timestamp objekta
- Očekujemo da će vrijednost vremena povratka biti null, budući da nije postavljena

Rezultat:

- Izvršavanje testa provjerava postavljanje i dohvaćanje vremena povratka, uspoređujući ga s očekivanim rezultatom
- Izvršavanje testa provjerava da li je vrijednost vremena povratka null, što ukazuje na ispravno ponašanje u slučaju ne postavljanja vremena povratka

```
@Test
void returnByTimeTestNull() {
    Listing listing = new Listing();
    Assertions.assertNull(listing.getReturnByTime());
}
```

Slika 5.4: Provjera vraćanja vremena u razredu Listing

```
@Test
void returnByTimeTest() {
    Listing listing = new Listing();
    LocalDateTime time = LocalDateTime.now();
    listing.setReturnByTime(time);

    Assertions.assertEquals(Timestamp.valueOf(time), listing.getReturnByTime());
}
```

Slika 5.5: Provjera vraćanja vremena u razredu Listing

Ispitni slučaj 4.: Test provjere svojstava skutera

Ulaz:

- Stvaranje primjerka korisnika (User) i postavljanje njegovih svojstava (ID i nadimak)
- Stvaranje primjerka skutera (Scooter) i postavljanje njegovih svojstava (ID, proizvođač, model, kapacitet baterije, maksimalna brzina, putanja do slike, maksimalni doseg, godina proizvodnje, dodatne informacije, korisnik, dostupnost)

Očekivani rezultati:

- Očekujemo da će svojstva skutera biti ispravno postavljena prema unesenim vrijednostima.

Rezultat:

- Izvršavanje testa provjerava ispravnost postavljanja svojstava skutera i uspoređuje ih s očekivanim rezultatima. postavljanja vremena povratka

```
@Test
void testScooterProperties() {

    User user = new User();
    user.setUserId(1L);
    user.setNickname("testUser");

    Scooter scooter = new Scooter();
    scooter.setScooterId(1L);
    scooter.setManufacturer("ScooterCo");
    scooter.setModel("FastScoot");
    scooter.setBatteryCapacity(500);
    scooter.setMaxSpeed(30);
    scooter.setImagePath("/path/to/scooter/image");
    scooter.setMaxRange(100.0);
    scooter.setYearOfManufacture(2022);
    scooter.setAdditionalInformation("Some information about the scooter");
    scooter.setUser(user);
    scooter.setAvailability(true);

    assertEquals( expected: 1L, scooter.getScooterId());
    assertEquals( expected: "ScooterCo", scooter.getManufacturer());
    assertEquals( expected: "FastScoot", scooter.getModel());
    assertEquals( expected: 500, scooter.getBatteryCapacity());
    assertEquals( expected: 30, scooter.getMaxSpeed());
    assertEquals( expected: "/path/to/scooter/image", scooter.getImagePath());
    assertEquals( expected: 100.0, scooter.getMaxRange());
    assertEquals( expected: 2022, scooter.getYearOfManufacture());
    assertEquals( expected: "Some information about the scooter", scooter.getAdditionalInformation());
    assertEquals(user, scooter.getUser());
    assertEquals( expected: true, scooter.getAvailability());
}
```

Slika 5.6: Test provjere svojstava skutera

Ispitni slučaj 5.: Test provjere vremena plaćanja u razredu transakcije

Ulaz:

- Stvaranje instance razreda Transaction
- Postavljanje vremena plaćanja (paymentTime) na trenutno vrijeme

Očekivani rezultati:

- Očekujemo da će vrijeme plaćanja biti jednako trenutnom vremenu u obliku Timestamp objekta
- Očekujemo da će vrijednost vremena plaćanja biti null, budući da nije postavljena

Rezultat:

- Izvršavanje testa provjerava postavljanje i dohvaćanje vremena plaćanja, uspoređujući ga s očekivanim rezultatom
- Izvršavanje testa provjerava da li je vrijednost vremena plaćanja null, što ukazuje na ispravno ponašanje u slučaju ne postavljanja vremena plaćanja.

```
@Test
void getPaymentTimestampTest() {
    Transaction transaction = new Transaction();
    LocalDateTime time = LocalDateTime.now();
    transaction.setPaymentTime(time);
    Assertions.assertEquals(Timestamp.valueOf(time), transaction.getPaymentTimestamp());
}
```

Slika 5.7: Test provjere vremena plaćanja u razredu transakcije

```
@Test
void returnByTimeTestNull() {
    Transaction transaction = new Transaction();
    Assertions.assertNull(transaction.getPaymentTimestamp());

}
```

Slika 5.8: Test provjere vremena plaćanja u razredu transakcije

Ispitni slučaj 6.: Test provjere svojstva korisnika

Ulaz:

- Stvaranje primjerka korisnika (User) i postavljanje njegovih svojstava (ID, nadimak, ime, prezime, broj kartice, e-mail, broj telefona, lozinka, uloga, status)

Očekivani rezultati:

- Očekujemo da će svojstva korisnika biti ispravno postavljena prema unesenim vrijednostima

Rezultat:

- Izvršavanje testa provjerava ispravnost postavljanja svojstava korisnika i uspoređuje ih s očekivanim rezultatima

```
@Test
void testUserProperties() {

    User user = new User();
    user.setUserId(1L);
    user.setNickname("testUser");
    user.setFirstName("John");
    user.setLastName("Doe");
    user.setCardNumber("1234567890123456");
    user.setEmail("john.doe@example.com");
    user.setPhoneNumber("+1234567890");
    user.setPassword("securePassword");
    user.setRole(UserRole.USER);
    user.setStatus(UserStatus.ACCEPTED);

    assertEquals( expected: 1L, user.getUserId());
    assertEquals( expected: "testUser", user.getNickname());
    assertEquals( expected: "John", user.getFirstName());
    assertEquals( expected: "Doe", user.getLastName());
    assertEquals( expected: "1234567890123456", user.getCardNumber());
    assertEquals( expected: "john.doe@example.com", user.getEmail());
    assertEquals( expected: "+1234567890", user.getPhoneNumber());
    assertEquals( expected: "securePassword", user.getPassword());
    assertEquals(UserRole.USER, user.getRole());
    assertEquals(UserStatus.ACCEPTED, user.getStatus());
}
```

Slika 5.9: Test provjere svojstva korisnika

Ispitni slučaj 7.: Test provjere funkcionalnosti prijave korisnika

Ulaz:

- Koristi se Mock objekt UserRepository
- Koristi se UserService objekt, gdje je userService injektiran u UserService pomoću @InjectMocks
- Postavlja se Mock korisnik (mockUser) s podacima za prijavu

Očekivani rezultati:

- Očekuje se da će korisničko ime i lozinka odgovarati podacima Mock korisnika
- Očekuje se da će userService uspješno vratiti Mock korisnika

Rezultat:

- Izvršava se prijava korisnika pomoću userService.login(”test@example.com”, ”password”)
- Provjerava se je li rezultat jednak Mock korisniku (mockUser)

```java
@SpringBootTest
public class LoginTest {

    @Mock
    private UserRepository userRepository;
    @InjectMocks
    private UserService userService;
    @Mock
    private User mockUser;
    @BeforeEach
    public void setUp() { MockitoAnnotations.openMocks( testClass: this); }

    @Test
    void testValidUserLogin() {
        String admin = "admin";
        Mockito.when(mockUser.getNickname()).thenReturn(admin);
        Mockito.when(mockUser.getPassword()).thenReturn(admin);
        Mockito.when(userService.login( email: "test@example.com",  password: "password")).thenReturn(mockUser);
        User result = userService.login( email: "test@example.com",  password: "password");
        assertEquals(mockUser, result);
    }

    @Test
    void testInvalidUserLogin() {
        UserService userService = new UserService(userRepository);
        Mockito.when(userRepository.findByEmail("nonexistent@example.com")).thenReturn( value: null);
        User result = userService.login( email: "nonexistent@example.com",  password: "wrongPassword");
        assertNull(result);
    }
}
```

Slika 5.10: Test provjere funkcionalnosti prijave korisnika

5.2.2 Ispitivanje sustava

Ispitni slučaj 1: Provjera ispravnosti registracije

Cilj:

- Automatsko testiranje procesa registracije novog korisnika koji ne unese broj mobitela na web stranici pomoću Selenium WebDriver s Chrome preglednikom.

Koraci testiranja:

- Navigacija na stranicu za registraciju

- Ispunjavanje svih polja registracijskog obrazca osim polja za broj mobitela
- Potvrda Registracije
- Provjera je li se korisnik registrirao

Očekivani rezultati:

- Korisnik se nije uspio registrirati
- Ispisana odgovarajuća poruka

```python
from selenium import webdriver
import os
import time
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.by import By

chrome_driver_path = 'C:/Users/karla/Desktop/chromedriver-win64/chromedriver'
chrome_options = webdriver.ChromeOptions()
chrome_options.add_argument(f"webdriver.chrome.driver={chrome_driver_path}")
driver = webdriver.Chrome(options=chrome_options)

try:

    driver.get('http://localhost:3000/register')

    ime_input = driver.find_element(By.XPATH, f'//label[text()="{"Ime:"}"]/following-sibling::input')
    prezime_input = driver.find_element(By.XPATH, f'//label[text()="{"Prezime:"}"]/following-sibling::input')
    nadimak_input = driver.find_element(By.XPATH, f'//label[text()="{"Nadimak:"}"]/following-sibling::input')
    email_input = driver.find_element(By.XPATH, f'//label[text()="{"Email:"}"]/following-sibling::input')
    kazna_input = WebDriverWait(driver, 10).until(EC.element_to_be_clickable((By.XPATH, '//input[@type="file"]')))
    dokument_input = driver.find_element(By.XPATH, f'//label[text()="{"Dokument Identifikacije:"}"]/following-sibling::input')
    phone_input = driver.find_element(By.CLASS_NAME, "PhoneInput")
    card_input = driver.find_element(By.XPATH, f'//label[text()="{"Broj kartice:"}"]/following-sibling::input')
    password_input = driver.find_element(By.XPATH, '//input[@type="password" and @placeholder="Upišite lozinku"]')
    confirm_password_input = driver.find_element(By.XPATH, '//input[@type="password" and @placeholder="Upišite ponovno istu lozinku"]')
    register_button = driver.find_element(By.XPATH, '//button[@type="submit" and @class="register-form-button"]')

    ime_input.send_keys('New')
    prezime_input.send_keys('User')
    nadimak_input.send_keys("usernew")
    email_input.send_keys("new_user@example.com")
    file_path = 'C:/Users/karla/Desktop/Progi_projekt/Codeblaze/IzvorniKod/frontend/slika.jpg'
    kazna_input.send_keys(file_path)
    dokument_input.send_keys(file_path)
    card_input.send_keys("123412341234")
    password_input.send_keys('password123')
    confirm_password_input.send_keys('password123')

    print("Filled out the form.")

    register_button.click()

    print("Pressed Register.")

    time.sleep(2)

    current = driver.current_url
    if (current != 'http://localhost:3000/home'):
        print("You must type phone number")


finally:

    driver.quit()
```

Slika 5.11: Provjera ispravnosti registracije

```
PS C:\Users\karla\Desktop\Progi_projekt> python registration.py

DevTools listening on ws://127.0.0.1:56131/devtools/browser/0cdf4c1b-4683-4523-a5bd-61181940e7c3
Filled out the form.
Pressed Register.
You must type phone number
```

Slika 5.12: Provjera ispravnosti registracije

Ispitni slučaj 2: Provjera ispravnosti prijave

Ovaj Selenium test simulira korisničku prijavu na web stranici, unoseći korisničko ime i lozinku te provjerava očekivani rezultat, tj. naslov stranice nakon prijave.

Cilj:

- Automatsko testiranje procesa prijavljivanja na web stranicu pomoću Selenium WebDriver s Chrome preglednikom.

Koraci testiranja:

- Navigacija na stranicu za prijavu
- Unos podataka za prijavu
- Provjera stranice nakon prijave

Očekivani rezultati:

- Poruka "Assertion successful: 'Codeblaze' is in the title." ispisuje se kao potvrda uspješne prijave.

```python
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.by import By

chrome_driver_path = 'C:/Users/karla/Desktop/chromedriver-win64/chromedriver'

chrome_options = webdriver.ChromeOptions()
chrome_options.add_argument(f"webdriver.chrome.driver={chrome_driver_path}")
driver = webdriver.Chrome(options=chrome_options)

try:
    print("Navigating to the login page...")
    driver.get('http://localhost:3000/login')

    username_input = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.ID, 'email'))
    )

    print("Performing login...")
    username_input = driver.find_element(By.ID, 'email')
    password_input = driver.find_element(By.ID, 'password')
    login_button = driver.find_element(By.CLASS_NAME, 'login-form-button')

    username_input.send_keys('admin@gmail.com')
    password_input.send_keys('admin')
    login_button.click()


    WebDriverWait(driver, 10).until(EC.visibility_of_element_located((By.CLASS_NAME, 'navbar-account')))


    assert 'Codeblaze' in driver.title
    print("Assertion successful: 'Codeblaze' is in the title.")

finally:
    print("Quitting the WebDriver...")
    driver.quit()
```

Slika 5.13: Provjera ispravnosti prijava

```
PS C:\Users\karla\Desktop\Progi_projekt> python login.py

DevTools listening on ws://127.0.0.1:51264/devtools/browser/5bfe62af-3b67-4e78-8f2a-da180e99459d
Navigating to the login page...
Performing login...
Assertion successful: 'Codeblaze' is in the title.
Quitting the WebDriver...
PS C:\Users\karla\Desktop\Progi_projekt>
```

Slika 5.14: Provjera ispravnosti prijava

5.3 Dijagram razmještaja

Dijagram rasporeda ilustrira kako su fizički i softverski resursi distribuirani unutar operativnog okvira sustava. Na serveru se smještaju dva ključna servisa: servis za web i servis za upravljanje bazom podataka. Kroz web preglednike, korisnici stječu pristup funkcionalnostima web aplikacije. Ovaj sustav funkcionira po modelu klijent-server arhitekture gdje je komunikacijski protokol između korisničkih uređaja i servera omogućen putem HTTP veze.

Slika 5.15: Prikaz dijagrama razmještanja

5.4 Upute za puštanje u pogon

Nakon pažljivog razvoja naše aplikacije, usmjereni smo na implementaciju u produkcijsko okruženje koristeći Render, platformu koja pruža PaaS rješenje. Ovaj proces je zahtijevao integraciju svih aplikacijskih komponenti na GitHubu, omogućujući time stvaranje Docker kontejnera, ključnih za funkcionalnost aplikacije u radnom okruženju.

Slika 5.16: Prikaz Render platforme.

Pregledom izvornog koda na GitHub repozitoriju osigurali smo praćenje i upravljanje verzijama koda. Izvorni kod naše aplikacije dostupan je na https://github.com/KvesaFer/Codeblaze. Prilikom pripreme za deploy, pažljivo smo postavili environment varijable u konfiguraciju našeg IDE-a kako bismo osigurali nesmetano funkcioniranje lokalnog razvojnog okruženja. Posebna pažnja posvećena je Dockerfile-u, gdje smo se pobrinuli da putanje unutar COPY naredbi budu precizno definirane.

Konfiguracijska datoteka application.properties sadrži ključne informacije za povezivanje s bazom podataka. U razvojnom okruženju, koristili smo H2 in-memory bazu, što je vidljivo iz konfiguracije spring.datasource parametara koji ukazuju na H2 bazu. Za produkcijsko okruženje, komentirane linije sadrže konfiguraciju za povezivanje s PostgreSQL bazom podataka hostiranom na Renderu.

Dodatno, omogućili smo pristup H2 konzoli postavljanjem spring.h2.console.enabled na true, što olakšava debugiranje i testiranje aplikacije u razvoju. Secret key, definiran u application.properties, koristi se za JWT autentifikaciju, dok logiranje različitih komponenti aplikacije postavljamo na odgovarajuće razine za detaljno praćenje tijekom razvoja i debugiranja.

```
spring.datasource.url=jdbc:h2:mem:testdb
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=admin
spring.datasource.password=admin
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect

spring.h2.console.enabled=true
```

Slika 5.17: Lokalni razvoj (koristeći H2 bazu).

Dockerfile.txt sadrži naredbe za kreiranje Docker slike aplikacije. Korištenjem dvostupanjskog procesa izgradnje, prvo se stvara build container koji izvršava Maven build, a zatim se finalna izvršna JAR datoteka kopira u runtime container.

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

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

# Pokretanje builda
RUN ./mvnw clean package

# Stvaranje containera u kojem ce se vrtiti aplikacija
FROM openjdk:17-alpine

## Ovdje je moguce instalirati alate potrebne za rad aplikacije. Vjerojatno vam nece trebati, no dobro je znati.
## Linux distro koji se koristi je Alpine, stoga se kao package manager koristi apk
#RUN apk install <nesto>

# Kopiranje izvrsnog JAR-a iz build containera u izvrsni container
COPY --from=builder target/*.jar /app.jar

# Izlaganje porta
EXPOSE 8080

# Naredba kojom se pokrece aplikacija
ENTRYPOINT ["java","-jar","/app.jar"]
```

Slika 5.18: Izgradnja i konfiguracija Docker kontejnera

setupproxy.js definira proxy middleware koji preusmjerava zahtjeve na /api endpointu na backend server, što omogućuje jednostavniju integraciju frontend i backend dijelova aplikacije tijekom razvoja.

```
const { createProxyMiddleware } = require("http-proxy-middleware");

module.exports = function (app) : void  {
    app.use(
        "/api",
        createProxyMiddleware( context: {
            target: "http://localhost:8080/",
            changeOrigin: true,
        })
    );
};
```

Slika 5.19: Konfiguracija proxy middlewarea za preusmjeravanje API zahtjeva

package.json definira metapodatke projekta, dependency-e potrebne za frontend dio aplikacije, kao i skripte koje se koriste za pokretanje i izgradnju projekta. Vidimo i konfiguraciju Node verzije koja se koristi, što osigurava kompatibilnost između različitih razvojnih okruženja.

```
"scripts": {
  "start": "react-scripts start",
  "build": "yarn install && react-scripts build",
  "start-prod": "node app.js",
  "test": "react-scripts test",
  "eject": "react-scripts eject"
},
"engines": {
  "node": ">=18.18.0 < 19.0.0"
},
```

Slika 5.20: Definicija skripti za pokretanje i izgradnju projekta

app.js postavlja express server koji služi statičke datoteke iz build direktorija i postavlja proxy za API zahtjeve. Konfiguracija se čita iz environment varijabli, što omogućuje fleksibilnost i jednostavnost konfiguracije u različitim okruženjima.

```
const express : e | () => core.Express = require("express");
const { createProxyMiddleware } = require("http-proxy-middleware");
require("dotenv").config();
const path : PlatformPath | path = require("path")
const app : any | Express = express();
// Configuration
const { PORT } = process.env;
const { HOST } = process.env;
const { API_BASE_URL } = process.env;
// Proxy
app.use(
    "/api",
    createProxyMiddleware( context: {
        target: API_BASE_URL,
        changeOrigin: true,
    })
);
app.use(express.static(path.join(__dirname, 'build')))
mkvesic +1
app.listen(PORT, HOST, backlog: () => {
    //console.log(`Starting Proxy at ${HOST}:${PORT}`);
});
mkvesic
app.get("*", async (req : Request<P, ResBody, ReqBody, ReqQuery, LocalsObj> , res : Response<ResBody, LocalsObj> ) : Promise<void> => {
        res.sendFile(path.join(__dirname, 'build', 'index.html'))
    }
);
```

Slika 5.21: Postavljanje Express servera s proxyjem i statickim serviranjem

Kroz ovaj proces, stvorili smo stabilnu, sigurnu i skalabilnu aplikaciju spremnu za produkcijsko okruženje. Deploy aplikacije na Render platformi bio je ključan korak u dostizanju naših operativnih ciljeva, omogućujući nam da našu aplikaciju učinimo dostupnom široj publici. Konačna aplikacija dostupna je na poveznici: https://codeblazefe.onrender.com/home.

6. Zaključak i budući rad

Projektni zadatak naše grupe bio je izrada web aplikacije koja će korisnicima omogućiti iznajmljivanje vlastitih romobila kao i unajmljivanje romobila drugih korisnika.
Za izradu aplikacije imali smo otprilike 19 tjedana, a ona je podijeljena u dvije faze. U prvoj je fazi naglasak bio na izradi dokumentacije i implementaciji generičkih funkcionalnosti dok je u drugoj fazi naglasak bio na implementaciji projekta, ispitivanju implementacije izrađene aplikacije te puštanju aplikacije u pogon. Nakon uvodne vježbe s asistentom i međusobnog upoznavanja, započeli smo s diskusijom tehnologija koje ćemo koristiti i planiranjem faza izrade aplikacije. Nakon toga započeli smo s intenzivnim radom na izradi dokumentacije; obrazaca uporabe, dijagrama obrazaca uporabe, sekvencijskih dijagrama i arhitekture sustava. Kako smo se svi prvi put susreli s ovim oblikom izrade projekta, imali smo dosta nedoumica oko toga kako na najbolji i najefikasniji način implementirati zadanu apolikaciju, ali i oko izrade dijagrama koje smo tek tada usvajali na nastavi. Kada smo ispravno definirali obrasce uporabe i iste uspješno prikazali dijagramima, razjasnili smo brojne nedoumice oko funkcionalnosti aplikacije koje smo prethodno imali. Nakon toga, dio tima radio je na izradi ostatka dokumentacije, a dio je počeo raditi na implementaciji generičkih funkcionalnosti. Do prve predaje uspješno smo napisali svu potrebnu dokumentaciju i implementirali sve do tada potrebne funkcionalnosti.
Druga faza našeg rada na aplikaciji započela je nakon dodjele bodova kada smo detaljnije razradili plan za daljnji razvoj. Kako je u toj fazi bilo više posla oko same implementacije, a manje oko dokumentacije, odlučili smo se podijeliti u podtimove. Svaki podtim imao je članove koji su radili na frontendu i one koji su radili na backendu aplikacije. Kako bi bili što efikasniji, podijelili smo zadatke tako da je svaki podtim morao napraviti određenu funkcionalnost aplikacije. Naravno, kada bi netko naišao na neki problem prilikom pisanja koda, međusobno smo si pomagali. Tijekom cijelog rada na projektu često smo se nalazili na sastancima kako bismo prodiskutirali nedoumice i izvijestili ostatak tima o napretku pojedinog podtima. Komunikacija među članovima tima odvijala se putem Discorda

gdje smo se redoviti čuli i obavještavali o napredcima i problemima s kojima smo se suočavali.
Kao i većina timova, posebice onih s manjkom iskustva poput nas, i mi smo se susreli s brojnim izazovima i poteškoćama rada u timu. Naša raspodjela u podtimove je bila dobra, no zadatke nismo baš najbolje podijelili. Događalo da jedan podtim ovisi o drugom odnosno da ne može raditi na svom zadatku dok drugi podtim ne dovrši svoj zadatak. Time smo si malo otežali posao jer bi bilo trenutaka kada su neki imali jako puno posla dok drugi nisu imali šta za raditi. Srećom, shvatili smo to na vrijeme te nakon toga pazili da ne ponavljamo takve greške prilikom daljnje organizacije rada.
Unatoč svemu, stekli smo neke nove i nadogradili postojeće vještine poput rada s GitHubom, primjene UML dijagrama, izrade i analize modela dizajna, izrade dokumentacije u Latexu te izrade funkcionalnog programskog proizvoda koristeći React.js i Spring Boot. Osim toga, stekli smo i iskustvo rada u timu koje sa sobom nosi određenu odgovornost i discipliniranost. Kroz sudjelovanje na projektu, većina nas stekla je prvo praktično iskustvo programskog inženjerstva koje će nam sigurno uvelike biti od koristi u daljnjem obrazovanju i kasnije na našem karijernom putu. Kao tim, iznimno smo zadovoljni ostvarenim ciljem te svjesni da postoji prostor za nadogradnju aplikacije. Moguće proširenje postojeće aplikacije je izrada istoimene mobilne aplikacije čime bi se povećala njena popularnost i olakšalo korištenje.

Popis literature

1. Programsko inženjerstvo, FER ZEMRIS
   http://www.fer.hr/predmet/proinz
2. Dokumentacija Spring Framework
   https://docs.spring.io/spring-framework/reference/index.html
3. Dokumentacija Spring Boot
   https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/
4. Dokumentacija React
   https://react.dev/blog/2023/03/16/introducing-react-dev
5. Dokumentacija Jakarta
   https://jakarta.ee/resources/
6. Spring Boot Archives
   https://studygyaan.com/category/spring-boot
7. Dokumentacija Lombok
   https://projectlombok.org/features/
8. Dodatno predavanje za backend
   https://gitlab.com/hrvojesimic/progi-project-teams-backend
9. Dodatno predavanje za frontend
   https://gitlab.com/jtomic/opp-project-teams-frontend
10. Dodatno predavanje deploy
    https://github.com/progi-devops
11. Prezentacije s dodatnih predavanja
    https://moodle.fer.hr/mod/folder/view.php

Indeks slika i dijagrama

2.1 Prikaz pronalaska romobila i početka vožnje 9
2.2 Prikaz dodatnih mogućnosti 9

3.1 Dijagram obrasca uporabe - funkcionalnost iznajmljivača 26
3.2 Dijagram obrasca uporabe - funkcionalnost klijenta i banke 27
3.3 Dijagram obrasca uporabe - funkcionalnost neregistriranog korisnika i administratora 28
3.4 Sekvencijski dijagram obrasca UC10 - Registriraj romobil 30
3.5 Sekvencijski dijagram obrasca UC14 - Oglasi romobil 32
3.6 Sekvencijski dijagram obrasca UC20 - Reagiraj na zahtjev za iznajmljivanje 33
3.7 Sekvencijski dijagram obrasca UC24 - Vrati romobil 34

4.1 Prikaz MVC obrasca 38
4.2 Prikaz dijagrama baze podataka 56
4.3 Prikaz prvog isječka dijagrama razreda 57
4.4 Prikaz drugog isječka dijagrama razreda 58
4.5 Prikaz trećeg isječka dijagrama razreda 59
4.6 Prikaz Controller dijagrama 60
4.7 Prikaz Service dijagrama 61
4.8 Dijagram stanja - klijent 62
4.9 Dijagram aktivnosti - oglašavanje romobila 64
4.10 Dijagram komponenti 65

5.1 Test provjere svojstva dokumenta 68
5.2 Test provjere funkcionalnosti ImageChangeRequestStatus 68
5.3 Test provjere funkcionalnosti ImageChangeRequestStatus 69
5.4 Provjera vraćanja vremena u razredu Listing 69
5.5 Provjera vraćanja vremena u razredu Listing 69
5.6 Test provjere svojstava skutera 70
5.7 Test provjere vremena plaćanja u razredu transakcije 71

5.8 Test provjere vremena plaćanja u razredu transakcije 71
5.9 Test provjere svojstva korisnika 72
5.10 Test provjere funkcionalnosti prijave korisnika 73
5.11 Provjera ispravnosti registracije 74
5.12 Provjera ispravnosti registracije 74
5.13 Provjera ispravnosti prijava 75
5.14 Provjera ispravnosti prijava 75
5.15 Prikaz dijagrama razmještanja 77
5.16 Prikaz Render platforme. 78
5.17 Lokalni razvoj (koristeći H2 bazu). 79
5.18 Izgradnja i konfiguracija Docker kontejnera 79
5.19 Konfiguracija proxy middlewarea za preusmjeravanje API zahtjeva . 80
5.20 Definicija skripti za pokretanje i izgradnju projekta 80
5.21 Postavljanje Express servera s proxyjem i statickim serviranjem . . . 81

Dodatak: Prikaz aktivnosti grupe

Dnevnik sastajanja

1. sastanak
   - Datum: 20. listopada 2023.
   - Prisustvovali: M. Kvesić, M. Jakovac, M. Knez, K. Šoštar, K. Đuroković, J. Gunjača, M. Bušić
   - Teme sastanka:
     - upoznavanje članova tima
     - proučavanje zadatka
     - podjela prvih zadataka
     - postavljanje GitHuba
     - napravljen početni plan projekta
2. sastanak
   - Datum: 25. listopada 2023.
   - Prisustvovali: M. Kvesić, M. Jakovac, M. Knez, K. Šoštar, K. Đuroković, J. Gunjača, M. Bušić
   - Teme sastanka:
     - komentiranje obavljenih zadataka
     - rasprava o sljedećim koracima
     - napravljena skica projekta
     - podjela novih zadataka
3. sastanak
   - Datum: 04. studenoga 2023.
   - Prisustvovali: M. Kvesić, M. Jakovac, M. Knez, K. Šoštar, K. Đuroković, J. Gunjača, M. Bušić
   - Teme sastanka:
     - komentiranje obavljenih zadataka
     - detaljna analiza i popravak postojećih obrazaca uporabe
     - analiza baze podataka
     - podjela novih zadataka

4. sastanak
    - Datum: 11. studenoga 2023.
    - Prisustvovali: M. Kvesić, M. Jakovac, M. Knez, K. Šoštar, K. Đuroković, J. Gunjača, M. Bušić
    - Teme sastanka:
        - komentiranje obavljenih zadataka
        - dogovoren plan za prvu reviziju
        - rasprava o nedoumicama oko aplikacije
        - dogovoren popravak dokumentacije
        - testiranje napravljene aplikacije
        - podjela novih zadataka
5. sastanak
    - Datum: 17. studenoga 2023.
    - Prisustvovali: M. Kvesić, M. Jakovac, M. Knez, K. Šoštar, K. Đuroković, J. Gunjača, M. Bušić
    - Teme sastanka:
        - provjera dokumentacije za prvu predaju
6. sastanak
    - Datum: 1. prosinca 2023.
    - Prisustvovali: M. Kvesić, M. Jakovac, M. Knez, K. Šoštar, K. Đuroković, J. Gunjača, M. Bušić
    - Teme sastanka:
        - razrada plana za daljnji razvoj aplikacije
        - podjela u podtimove
        - podjela zadataka
7. sastanak
    - Datum: 15. prosinca 2023.
    - Prisustvovali: M. Kvesić, M. Jakovac, M. Knez, K. Šoštar, K. Đuroković, J. Gunjača, M. Bušić
    - Teme sastanka:
        - diskutiranje nedoumica
        - testiranje postojećih funkcionalnosti
8. sastanak
    - Datum: 22. prosinca 2023.
    - Prisustvovali: M. Kvesić, M. Jakovac, M. Knez, K. Šoštar, K. Đuroković,

J. Gunjača, M. Bušić
- Teme sastanka:
  - podjela preostalih zadataka

9. sastanak
   - Datum: 12. siječnja 2024.
   - Prisustvovali: M. Kvesić, M. Jakovac, M. Knez, K. Šoštar, K. Đuroković, J. Gunjača, M. Bušić
   - Teme sastanka:
     - pregled svega napravljenog
     - dogovor oko izrade preostalog dijela dokumentacije
10. sastanak
    - Datum: 15. siječnja 2024.
    - Prisustvovali: M. Kvesić, M. Jakovac, M. Knez, K. Šoštar, K. Đuroković, J. Gunjača, M. Bušić
    - Teme sastanka:
      - završetak implementacije
11. sastanak
    - Datum: 18. siječnja 2024.
    - Prisustvovali: M. Kvesić, M. Jakovac, M. Knez, K. Šoštar, K. Đuroković, J. Gunjača, M. Bušić
    - Teme sastanka:
      - dovršavanje dokumentacije
      - pregled aplikacije

Tablica aktivnosti

	Marin Kvesić	Matija Jakovac	Mirna Knez	Jerko Gunjača	Karla Šoštar	Katarina Đuroković	Matea Bušić
Upravljanje projektom	24	21	12	14	11	11	15
Opis projektnog zadatka	0	0	0	0	0	0	13
Funkcionalni zahtjevi	0	0	0	0	6	6	5
Opis pojedinih obrazaca	0	0	0	0	9	15	7
Dijagram obrazaca	0	0	0	0	7	8	0
Sekvencijski dijagrami	0	0	0	0	5	7	0
Opis ostalih zahtjeva	0	0	0	0	0	0	1
Arhitektura i dizajn sustava	0	0	13	0	0	0	2
Baza podataka	0	0	9	0	0	0	0
Dijagram razreda	0	0	6	0	0	2	0
Dijagram stanja	0	0	0	0	0	2	0
Dijagram aktivnosti	0	0	0	0	0	1	0
Dijagram komponenti	0	0	0	0	0	0	5
Korištene tehnologije i alati	0	0	0	0	1	0	0
Ispitivanje programskog rješenja	0	0	0	0	18	0	0
Dijagram razmještaja	0	0	3	0	0	0	0
Upute za puštanje u pogon	0	0	5	0	0	0	0
Dnevnik sastajanja	0	0	0	0	0	0	2
Zaključak i budući rad	0	0	0	0	0	0	2

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

	Marin Kvesić	**Matija Jakovac**	**Mirna Knez**	**Jerko Gunjača**	**Karla Šoštar**	**Katarina Đuroković**	**Matea Bušić**
Popis literature	0	0	0	0	0	0	1
Izrada početne stranice	5	3	8	14	0	0	5
Izrada baze podataka	6	14	3	0	0	0	0
Spajanje s bazom podataka	6	5	0	0	0	0	0
Back end	63	65	2	0	35	15	35
Front end	50	45	70	60	20	45	30
Postavljanje web aplikacije na poslužitelj	5	5	0	0	0	0	0

Dijagrami pregleda promjena

dio 2. revizije

Prenijeti dijagram pregleda promjena nad datotekama projekta. Potrebno je na kraju projekta generirane grafove s gitlaba prenijeti u ovo poglavlje dokumentacije. Dijagrami za vlastiti projekt se mogu preuzeti s gitlab.com stranice, u izborniku Repository, pritiskom na stavku Contributors.