

Programsko inženjerstvo

Ak. god. 2023./2024.

Digitalizacija

Dokumentacija, Rev. 1

Grupa: *Inženjerski Pristup*

Voditelj: *Vilim Branica*

Datum predaje: 17. 11. 2023.

Nastavnik: *Vlado Sruk*

Sadržaj

1 Dnevnik promjena dokumentacije	3
2 Opis projektnog zadatka	5
2.1 Uvod	5
2.2 Osnovne funkcionalnosti	5
2.3 Tehnički aspekti i razvojni ciklus	6
2.4 Prava određenih korisnika	7
2.5 Slična rješenja	8
2.6 Partnerstva i integracije	8
2.7 Zaključak	9
3 Specifikacija programske potpore	10
3.1 Funkcionalni zahtjevi	10
3.1.1 Obrasci uporabe	12
3.1.2 Sekvencijski dijagrami	21
3.2 Ostali zahtjevi	23
4 Arhitektura i dizajn sustava	25
4.1 Baza podataka	26
4.1.1 Opis tablica	27
4.1.2 Dijagram baze podataka	38
4.2 Dijagram razreda	40
4.3 Dijagram stanja	43
4.4 Dijagram aktivnosti	44
4.5 Dijagram komponenti	45
5 Implementacija i korisničko sučelje	46
5.1 Korištene tehnologije i alati	46
5.2 Ispitivanje programskog rješenja	47
5.2.1 Ispitivanje komponenti	47
5.2.2 Ispitivanje sustava	56

5.3	Dijagram razmještaja	58
5.4	Upute za puštanje u pogon	59
6	Zaključak i budući rad	62
	Popis literature	64
	Indeks slika i dijagrama	65
	Dodatak: Prikaz aktivnosti grupe	66

1. Dnevnik promjena dokumentacije

Kontinuirano osvježavanje

Rev.	Opis promjene/dodatka	Autori	Datum
0.1	Napravljen predložak dokumentacije.	Marko Še-lendić	24.10.2023.
0.2	Dodani funkcionalni zahtjevi i obrasci upo-rabe.	Marko Še-lendić	25.10.2023.
0.3	Dodan opis projekta.	Tomislav Čupić	30.10.2023.
0.4	Dodani sekvencijski dijagrami	Nika Mili-čević	4.11.2023.
0.5	Dodani <i>Use Case</i> dijagrami	Nika Mili-čević	4.11.2023.
0.6	Dodani prvi sastanci	Marko Še-lendić	8.11.2023.
0.6.1	Dodani ostali sastanci	Tomislav Čupić	9.11.2023.
0.7	Dodani opis baze	Filip Kril-čić	13.11.2023.
0.8	Dodani opis arhitekture	Zvonimir Pipić	14.11.2023.
1.0	Osnovni model aplikacije pušten u pogon	Svi	17.11.2023.
1.1	Dodane korištene tehnologije	Filip Kril-čić	4.1.2024.

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Rev.	Opis promjene/dodatka	Autori	Datum
1.2	Dodani dijagrami aktivnosti i dijagrami komponenti	Nika Miličević	9.1.2024.
1.3	Dodano ispitivanje komponenti	Filip Krilić	10.1.2024.
1.3.1	Popravljeni dijagrami	Nika Miličević	13.1.2024.
1.3.2	Dodan sastanak	Tomislav Čupić	13.1.2024.
1.4	Dodane upute za puštanje u pogon	Nika Miličević	18.1.2024.

Moraju postojati glavne revizije dokumenata 1.0 i 2.0 na kraju prvog i drugog ciklusa. Između tih revizija mogu postojati manje revizije već prema tome kako se dokument bude nadopunjavao. Očekuje se da nakon svake značajnije promjene (dodatka, izmjene, uklanjanja dijelova teksta i popratnih grafičkih sadržaja) dokumenta se to zabilježi kao revizija. Npr., revizije unutar prvog ciklusa će imati oznake 0.1, 0.2, ..., 0.9, 0.10, 0.11.. sve do konačne revizije prvog ciklusa 1.0. U drugom ciklusu se nastavlja s revizijama 1.1, 1.2, itd.

2. Opis projektnog zadatka

2.1 Uvod

Cilj ovog projekta je razviti web aplikaciju koja će značajno ubrzati proces digitalizacije u računovodstvenim tvrtkama. Tradicionalni način obrade dokumenata je gotovo uvijek spor i nepouzdan jer je podložan ljudskim pogreškama. Naša nova aplikacija rješava te probleme te pruža brži, precizniji i pouzdaniji način upravljanja dokumentima. Ključna prednost nad tradicionalnim načinom je povećanje produktivnosti. Korisnici aplikacije će moći obrađivati mnogo veći broj dokumenata u puno kraćem vremenskom periodu, što rezultira značajnim uštedama vremena i resursa uložениh u rad.

2.2 Osnovne funkcionalnosti

Glavna funkcionalnost ove aplikacije je detekcija dokumenata na slikama čime se eliminira potreba za ručnim označavanjem i kategorizacijom dokumenata. Nakon toga dolazi izvođenje optičkog prepoznavanja teksta (OCR) iz tih dokumenata. Ovaj ključni proces omogućava tvrtki da izvuče vrijedne informacije iz papirnatih dokumenata i prevede ih u digitalni oblik. Sofisticiranim OCR sustavom se može postići vrlo visoka točnost u prepoznavanju teksta. Aplikacija je fleksibilna i skalabilna, omogućit će se korisnicima učitavanje do 50 slika istovremeno.

Značaj ovog projekta je unaprijediti i ubrzati proces digitalizacije dokumenata, osiguravajući točnost i pouzdanost u svakom koraku, od samog skeniranja do arhiviranja. S ovom aplikacijom će se pomoći smanjiti vjerojatnost ljudskih pogrešaka što je od velike važnosti za očuvanje točnosti financijskih podataka, te se time povećava učinkovitost poslovanja.

Prilikom pokretanja aplikacije prikazuje se mogućnost prijave u sustav s postojećim računom za što su potrebni:

- Korisničko ime
- Zaporka



Sign in

SIGN IN

Copyright © Inženjerski pristup, Digitalizacija 2024.

Slika 2.1: Slika prijave korisnika u sustav

2.3 Tehnički aspekti i razvojni ciklus

Aplikacija će biti razvijena korištenjem modernih tehnologija kako bi osigurala visoku razinu sigurnosti i performansi. Backend će biti implementiran pomoću popularnog okvira kao što je Django, osiguravajući robusnost i skalabilnost sustava. Kako bi se postigla brza i responzivna korisnička sučelja, frontend će se temeljiti na Reactu. Važan aspekt svake aplikacije je korisničko sučelje koje smo dizajnirali tako da je vrlo intuitivno i jednostavno za korištenje. Dizajnerske odluke temelje se na najboljim praksama kako bi se različitim korisnicima osiguralo ugodno is-

kustvo. Kroz pažljivo planiranje korisničkih tokova i jasnu navigaciju, aplikacija će omogućiti korisnicima brz i učinkovit pristup funkcionalnostima. Razvoj će se odvijati u iteracijama koristeći agilne metodologije, što će omogućiti fleksibilnost u prilagodbi promjenama zahtjeva. Redovito testiranje, uključujući testiranje jedinica, integracije i sustava, bit će ključno za osiguravanje stabilnosti i performansi aplikacije.

2.4 Prava određenih korisnika

Prijavom u sustav korisniku se dodjeljuju njegova prava. Svaki korisnik ima osnovne mogućnosti:

- mijenjanje lozinke korisničkog računa
- skeniranje dokumenata
- pristup povijesti vlastitih skeniranih dokumenata
- Nakon skeniranja dokumenta ima pravo potvrditi njegovu točnost, odnosno odbiti ga.

S obzirom na osjetljivost financijskih podataka, sigurnost je ključna komponenta. Pristup određenim funkcionalnostima bit će strogo kontroliran putem sustava ovlasti, osiguravajući da svaki korisnik ima pristup samo onim informacijama i radnjama koje su relevantne za njegovu ulogu.

Dodatna prava koja ovise o njegovu položaju u tvrtki su:

- **Zaposlenici:**
 - Imaju pravo slati svoje skenirane dokumente revizoru.
- **Revizori:**
 - Imaju odgovornost za pregled svih pristiglih dokumenata i daljnje usmjeravanje prema računovođama
 - odbijanje dokumenta ako ne zadovoljava potrebne uvjete
- **Računovođe:**
 - Imaju ključnu ulogu u arhiviranju primljenih dokumenata i dodjeljivanju jedinstvenih brojeva arhivama
 - slanje dokumenata direktoru na potpis
- **Direktor:**

- Ima najveće ovlasti, potpunu preglednost nad svim dokumentima i statistikama zaposlenika
- Pregled povijesti dokumenata i zaposlenika
- mogućnost dijeljenja odabranih dokumenata na društvene mreže
- Jedini ima mogućnost registriranja novih korisnika, pri čemu on i dodjeljuje ulogu svakom novom korisniku.
- Može i obrisati korisnika koji se nalazi u sustavu iz sustava.

2.5 Slična rješenja

Ova aplikacija nije jedina takva na tržištu, slična rješenja uključuju Adobe Acrobat za OCR i upravljanje dokumentima, te ABBYY FineReader za napredno optičko prepoznavanje znakova. No ova aplikacija se razlikuje po svojoj kombinaciji funkcionalnosti. Ona će kombinirati najbolje aspekte ovih rješenja i dodati posebne značajke prilagođene potrebama računovodstva. To uključuje automatsku detekciju i organizaciju dokumenata te više razine pristupa.

2.6 Partnerstva i integracije

Staviti ćemo naglasak na uspostavu suradnje s drugim tehnološkim tvrtkama i pružateljima usluga kako bi se proširile funkcionalnosti aplikacije. Partnerstva mogu omogućiti pristup dodatnim resursima, stručnost u specifičnim područjima te ubrzanje inovacija. Prvenstveno ćemo se fokusirati na partnere koji se bave područjima naprednog OCR-a, sigurnosnih rješenja ili prilagođenih integracija. Kroz partnerstva i integracije, aplikacija će pružiti dodatnu vrijednost korisnicima. Na primjer, mogućnost sinkronizacije podataka s računovodstvenim softverom ili automatsko prepoznavanje i integraciju informacija iz drugih aplikacija može značajno poboljšati korisničko iskustvo. Razvojni tim će redovito pratiti tehničke trendove i inovacije kako bi osigurao da partnerstva i integracije prate najnovije tehnološke standarde. Ova proaktivna pristup osigurat će dugoročnu relevantnost aplikacije u dinamičnom okruženju tehnoloških promjena.

2.7 Zaključak

U konačnici, ovim projektom stvaraju se temelji za razvoj napredne i inovativne web aplikacije koja će uvelike ubrzati i poboljšati proces digitalizacije u računovodstvenim tvrtkama i donijeti korisnicima novo i učinkovito rješenje s minimaliziranom količinom pogrešaka za obradu dokumenata. Uz kombinaciju tehnoloških rješenja, sigurnosnih mjera i korisničkih sučelja, očekuje se da će ova aplikacija postati ključni alat u optimizaciji poslovnih operacija. Njezina prilagodljivost i skalabilnost čine je perspektivnim rješenjem koje će unaprijediti digitalizaciju dokumenata na novu razinu. Kroz ove aktivnosti, projekt će izgraditi snažnu mrežu podrške i suradnje, čime će osigurati da aplikacija ostane fleksibilna i prilagodljiva promjenama u tehnologiji i zahtjevima korisnika.

3. Specifikacija programske potpore

3.1 Funkcionalni zahtjevi

Dionici:

1. Zaposlenik
2. Revizor
3. Računovođa
4. Direktor
5. Razvojni tim

Aktori i njihovi funkcionalni zahtjevi:

1. Svaki neregistrirani/neprijavljeni korisnik (inicijator) može:
 - (a) prijaviti se u sustav
2. Svaki registrirani/prijavljeni korisnik (inicijator) može:
 - (a) promijeniti lozinku korisničkog računa
 - (b) vidjeti povijest svojih skeniranja dokumenata
 - (c) skenirati novi dokument i:
 - i. potvrditi točnost skeniranog dokumenta
 - ii. odbiti skenirani dokument
 - (d) odjaviti se iz sustava
3. Zaposlenik (inicijator) može:
 - (a) poslati skenirani dokument revizoru na reviziju
4. Revizor (inicijator) može:
 - (a) provjeriti pristigle dokumente, uključujući svoje, i:
 - i. potvrditi dokument i proslijediti ga nadležnom računovođi
 - ii. odbiti dokument
5. Računovođa (inicijator) može:

- (a) arhivirati pristigle dokumente
- (b) proslijediti pristigli dokument direktoru na potpis

6. Direktor (inicijator) može:

- (a) potpisati dokumente pristigle iz računovodstva
- (b) vidjeti povijest svih skeniranja dokumenata
- (c) vidjeti povijest skeniranja i statistike svih zaposlenika
- (d) objaviti određeni dokument na društvenim mrežama
- (e) registrirati novog korisnika i dodijeliti mu ulogu
- (f) obrisati postojećeg korisnika

7. Baza podataka (sudionik):

- (a) pohranjuje sve podatke o korisnicima i njihovim ovlastima (ulogama)
- (b) pohranjuje sve podatke o skeniranim dokumentima

3.1.1 Obrasci uporabe

UC1 — Prijava u sustav

- **Glavni sudionik:** Bilo koji korisnik
- **Cilj:** Dobiti pristup korisničkom sučelju
- **Sudionici:** Baza podataka
- **Preduvjet:** Posjedovanje vlastitog korisničkog računa
- **Opis osnovnog tijeka:**
 1. Korisnik unosi korisničko ime i lozinku
 2. Baza podataka provjerava ispravnost unesenih podataka
 3. Korisnik dobiva pristup korisničkom sučelju
- **Opis mogućih odstupanja:**
 - 2.a Korisnik je unio neispravno korisničko ime ili lozinku
 1. Sustav obavještava korisnika o neuspjeloj prijavi i vraća ga na stranicu za prijavu

UC2 — Promjena lozinke korisničkog računa

- **Glavni sudionik:** Bilo koji korisnik
- **Cilj:** Promijeniti lozinku korisničkog računa
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire opciju promjene lozinke računa
 2. Korisnik unosi trenutnu lozinku računa
 3. Baza podataka provjerava ispravnost unesene lozinke
 4. Korisnik unosi novu lozinku računa
 5. Baza podataka pohranjuje promjenu lozinke
- **Opis mogućih odstupanja:**
 - 3.a Korisnik je unio neispravnu trenutnu lozinku
 1. Sustav obavještava korisnika o pogrešnom unosu trenutne lozinke i vraća ga na stranicu za unos lozinke

UC3 — Pregled povijesti skeniranih dokumenata

- **Glavni sudionik:** Zaposlenik

- **Cilj:** Vidjeti povijest svojih skeniranja dokumenata
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Aplikacija korisniku na početnom zaslonu prikazuje njegovu povijest skeniranih dokumenata

UC4 — Skeniranje novog dokumenta

- **Glavni sudionik:** Bilo koji korisnik
- **Cilj:** Skenirati novi dokument
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik na početnom zaslonu odabire opciju za skeniranje novog dokumenta
 2. Korisnik učitava jedan ili više dokumenata u slikovnom formatu u aplikaciju
 3. Aplikacija provjerava ispravnost učitanih dokumenata
 4. Aplikacija korisniku prikazuje učitani dokument i razvrstava ga u jednu od kategorija:
 - (a) račun
 - (b) ponuda
 - (c) interni dokument
 5. Korisnik potvrđuje točnost učitanoog dokumenta ili ga odbija
- **Opis mogućih odstupanja:**
 - 3.a Dokument nije ispravno skeniran
 1. Aplikacija obavještava korisnika o neuspjelom skeniranju te uzroku pogreške i vraća ga na zaslon za učitavanje dokumenta

UC5 — Slanje skeniranog dokumenta na reviziju

- **Glavni sudionik:** Zaposlenik
- **Cilj:** Poslati skenirani dokument revizoru na reviziju
- **Sudionici:** Baza podataka, revizor
- **Preduvjet:** Zaposlenik je prijavljen u sustav i skenirao je dokument
- **Opis osnovnog tijeka:**
 1. Zaposlenik odabire opciju za slanje skeniranog dokumenta na reviziju

2. Aplikacija od baze podataka traži popis svih revizora
3. Baza podataka vraća aplikaciji popis svih revizora
4. Aplikacija zaposleniku prikazuje popis svih revizora
5. Zaposlenik odabire revizora kojemu želi poslati dokument
6. Aplikacija šalje dokument odabranom revizoru te o tome obavještava bazu podataka

UC6 — Provjera pristiglih dokumenata

- **Glavni sudionik:** Revizor
- **Cilj:** Provjeriti pristigle dokumente, uključujući svoje
- **Sudionici:** Baza podataka, zaposlenik, računovođa
- **Preduvjet:** Revizor je prijavljen u sustav i ima pristiglih dokumenata
- **Opis osnovnog tijeka:**
 1. Revizor odabire opciju za provjeru pristiglog dokumenta
 2. Aplikacija revizoru prikazuje dokument
 3. Revizor potvrđuje točnost dokumenta ili ga odbija:
 - (a) točan dokument aplikacija prosljeđuje nadležnom računovođi te o tome obavještava bazu podataka
 - (b) netočan dokument aplikacija odbacuje i o tome dojavljuje zaposlenika

UC7 — Arhiviranje pristiglih dokumenata

- **Glavni sudionik:** Računovođa
- **Cilj:** Arhivirati pristigle dokumente
- **Sudionici:** Baza podataka
- **Preduvjet:** Računovođa je prijavljen u sustav i ima pristiglih dokumenata
- **Opis osnovnog tijeka:**
 1. Računovođa odabire opciju za arhiviranje pristiglog dokumenta
 2. Aplikacija dojavljuje bazi podataka da arhivira dokument
 3. Baza podataka arhivira dokument i dodjeljuje mu jedinstveni broj arhiva

UC8 — Slanje dokumenata na potpis direktoru

- **Glavni sudionik:** Računovođa
- **Cilj:** Poslati dokument direktoru na potpis
- **Sudionici:** Baza podataka, direktor
- **Preduvjet:** Računovođa je prijavljen u sustav i ima pristiglih dokumenata

- **Opis osnovnog tijeka:**

1. Računovođa odabire opciju za slanje dokumenta direktoru na potpis
2. Aplikacija šalje dokument direktoru na potpis te o tome obavještava bazu podataka
3. Direktor dobiva obavijest o pristiglom dokumentu

UC9 — Potpisivanje dokumenata

- **Glavni sudionik:** Direktor
- **Cilj:** Potpisati dokumente pristigle iz računovodstva
- **Sudionici:** Baza podataka, računovođa
- **Preduvjet:** Direktor je prijavljen u sustav i ima pristiglih dokumenata
- **Opis osnovnog tijeka:**
 1. Direktor odabire opciju za potpisivanje dokumenta
 2. Aplikacija dojavljuje bazi podataka da je dokument potpisan
 3. Baza podataka označava dokument kao potpisan
 4. Aplikacija šalje potpisani dokument računovođi

UC10 — Pregled povijesti dokumenata

- **Glavni sudionik:** Direktor
- **Cilj:** Vidjeti povijest svih dokumenata
- **Sudionici:** Baza podataka
- **Preduvjet:** Direktor je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Aplikacija direktoru na početnom zaslonu prikazuje povijest svih skeniranja dokumenata

UC11 — Pregled povijesti i statistika zaposlenika

- **Glavni sudionik:** Direktor
- **Cilj:** Vidjeti povijest i statistike svih zaposlenika
- **Sudionici:** Baza podataka
- **Preduvjet:** Direktor je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Aplikacija direktoru na početnom zaslonu prikazuje povijest i statistike svih zaposlenika

UC12 — Registracija novog korisnika

- **Glavni sudionik:** Direktor
- **Cilj:** Registrirati novog korisnika i dodijeliti mu ulogu
- **Sudionici:** Baza podataka
- **Preduvjet:** Direktor je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Direktor odabire opciju za registraciju novog korisnika
 2. Aplikacija prikazuje formu za registraciju novog korisnika
 3. Direktor unosi podatke o novom korisniku, uključujući njegovu ulogu
 4. Aplikacija provjerava ispravnost unesenih podataka
 5. Aplikacija dojavljuje bazi podataka da registira novog korisnika
 6. Baza podataka pohranjuje podatke o novom korisniku
- **Opis mogućih odstupanja:**
 - 4.a Direktor je unio neispravne podatke o novom korisniku
 1. Aplikacija obavještava direktora o neuspjeloj registraciji i vraća ga na formu za registraciju novog korisnika

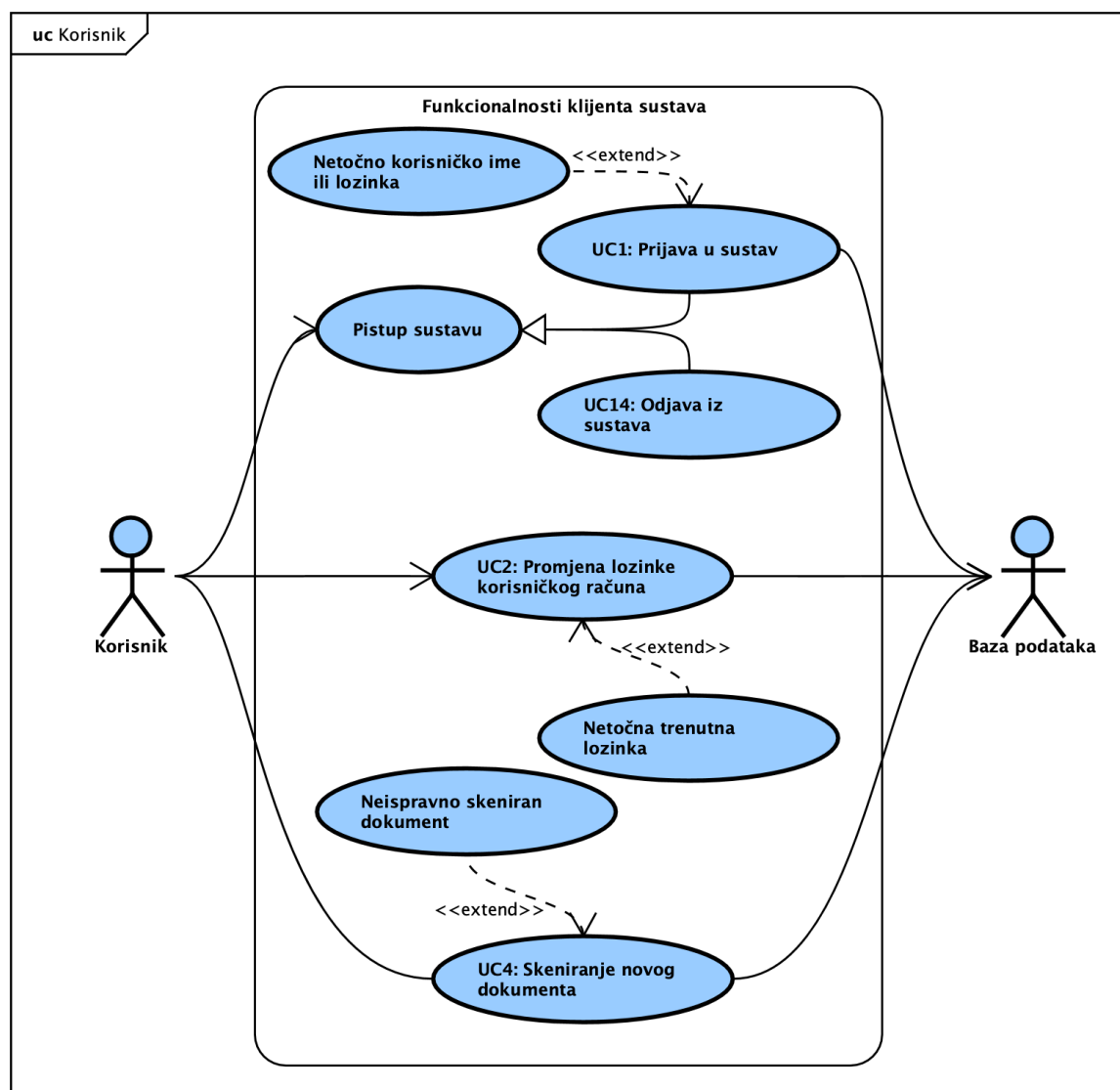
UC13 — Objavljivanje dokumenta na društvene mreže

- **Glavni sudionik:** Direktor
- **Cilj:** Objaviti određeni dokument na društvene mreže
- **Sudionici:** Baza podataka, društvene mreže
- **Preduvjet:** Direktor je prijavljen u sustav i dokument je potpisan
- **Opis osnovnog tijeka:**
 1. Direktor odabire opciju za objavljivanje dokumenta na društvene mreže
 2. Aplikacija direktoru prikazuje izbor društvenih mreža
 3. Direktor odabire društvenu mrežu na koju želi objaviti dokument
 4. Aplikacija direktora preusmjerava na odabranu društvenu mrežu

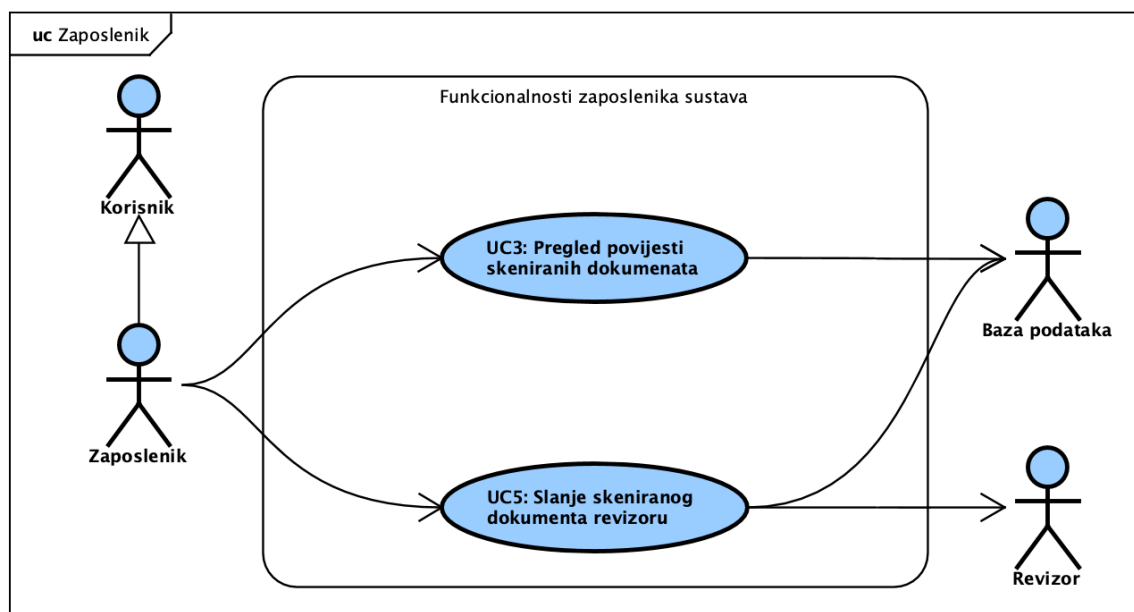
UC14 — Odjava iz sustava

- **Glavni sudionik:** Korisnik
- **Cilj:** Odjaviti se iz sustava
- **Sudionici:** —
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire opciju za odjavu iz sustava
 2. Aplikacija korisnika odjavljuje iz sustava i preusmjerava na stranicu za prijavu

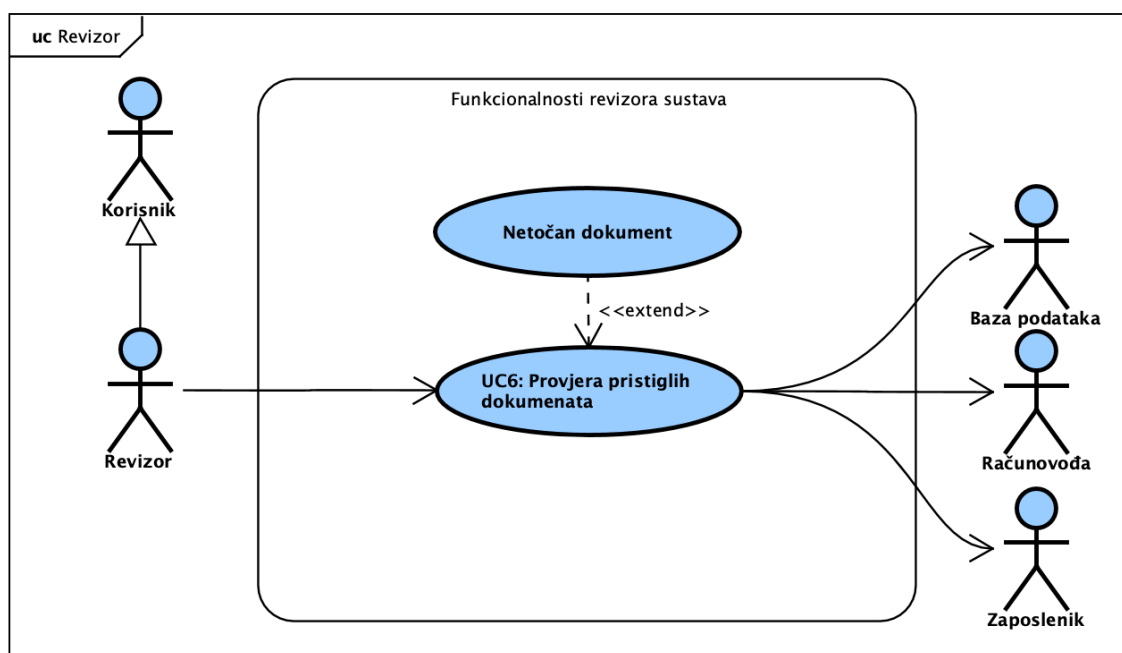
Dijagrami obrazaca uporabe



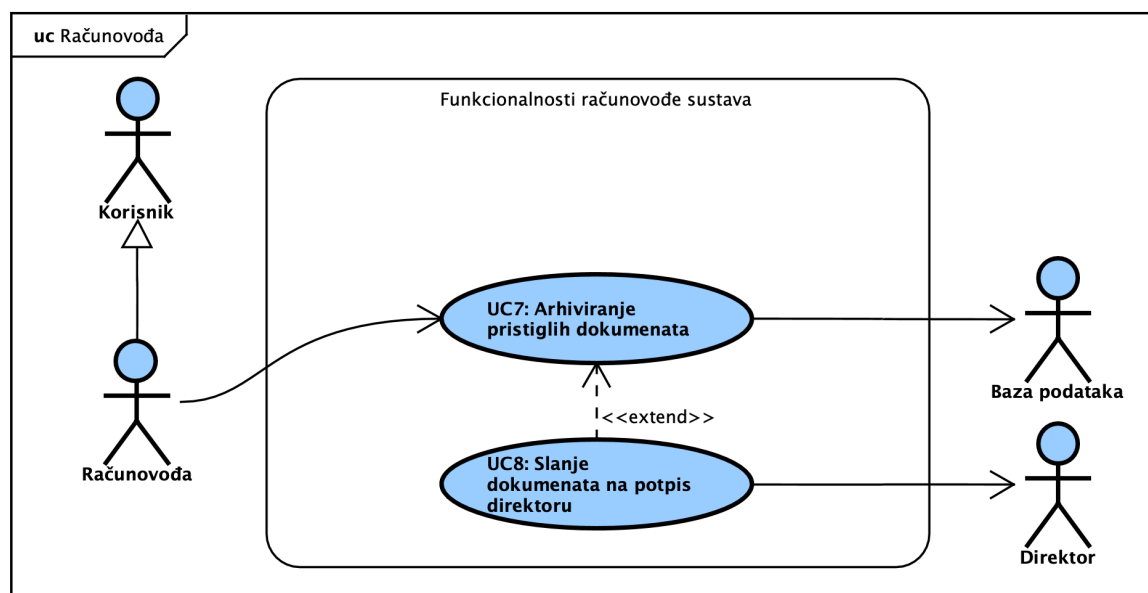
Slika 3.1: Dijagram obrasca uporabe, funkcionalnost korisnika



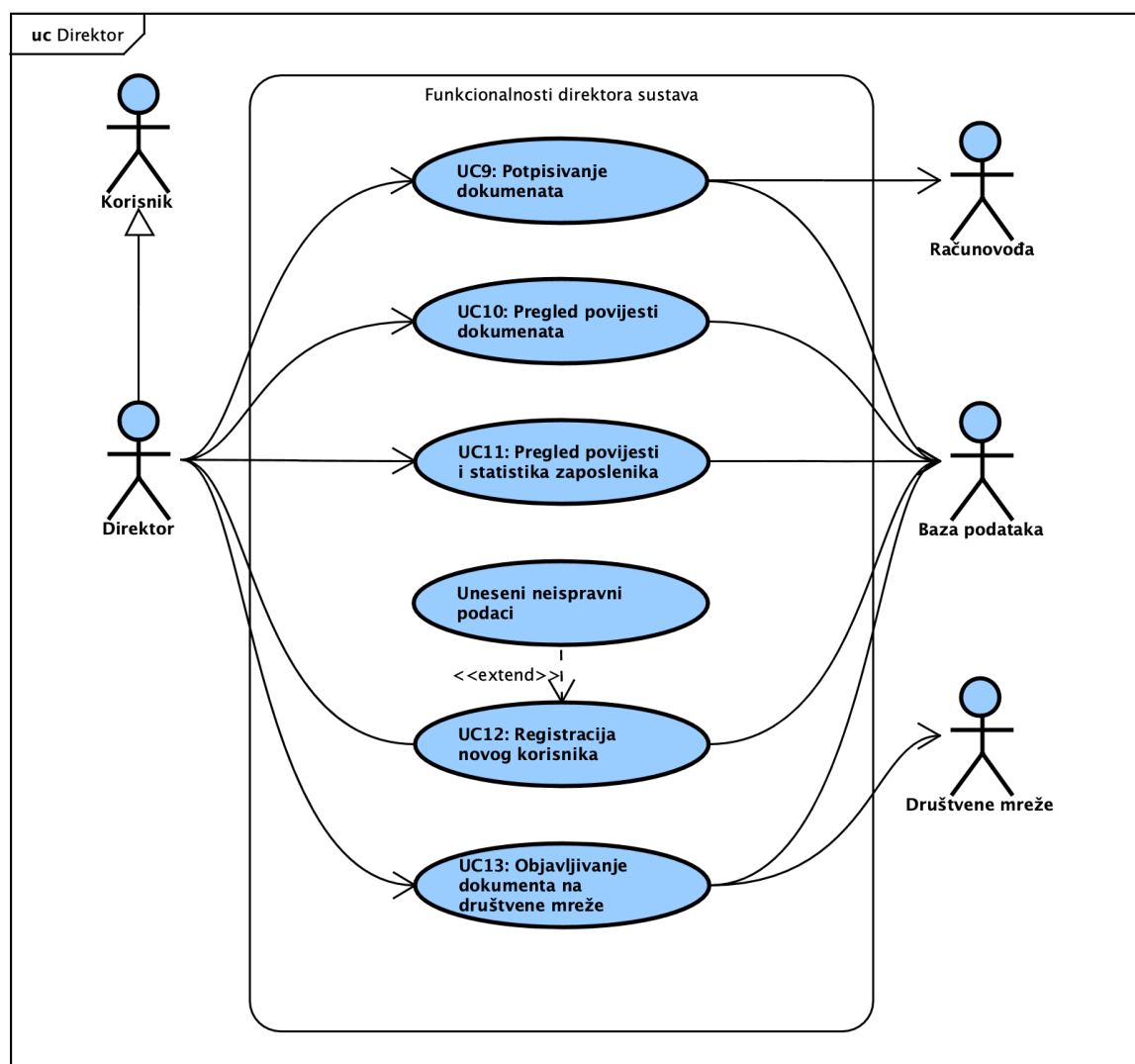
Slika 3.2: Dijagram obrasca uporabe, funkcionalnost zaposlenika



Slika 3.3: Dijagram obrasca uporabe, funkcionalnost revizora



Slika 3.4: Dijagram obrasca uporabe, funkcionalnost računovođe

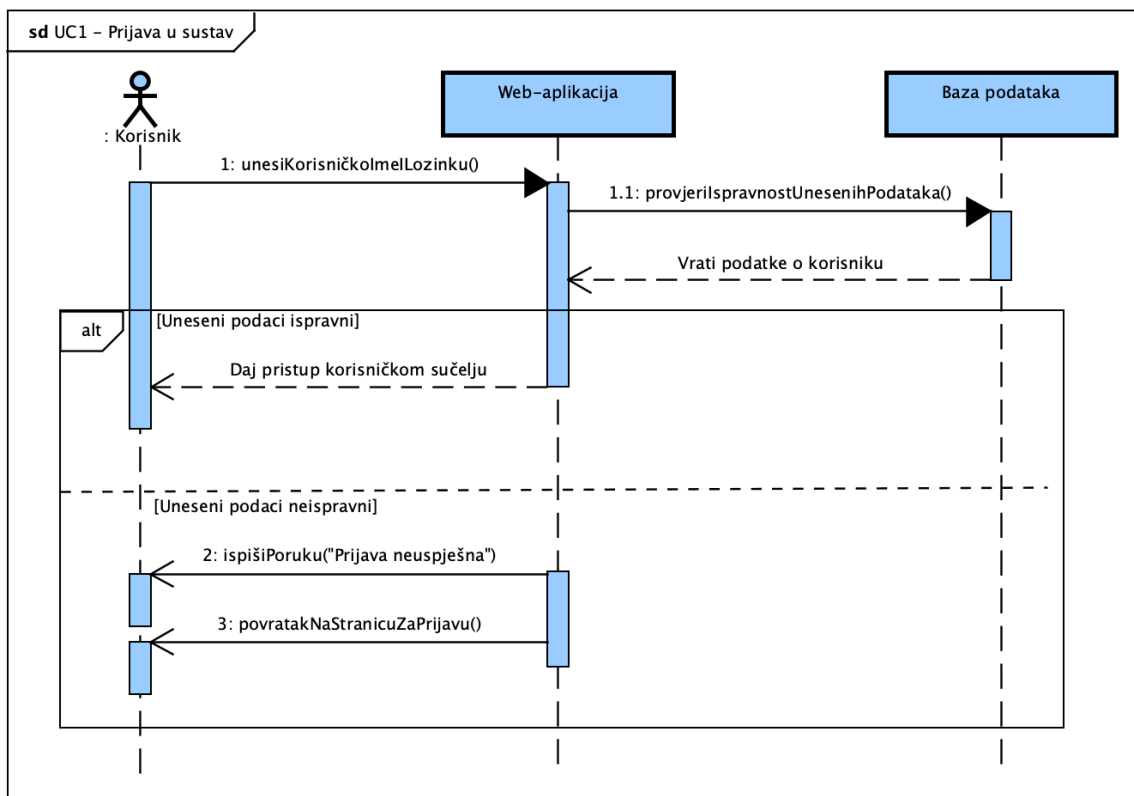


Slika 3.5: Dijagram obrasca uporabe, funkcionalnost direktora

3.1.2 Sekvencijski dijagrami

Obrazac uporabe UC1 - Prijava u sustav

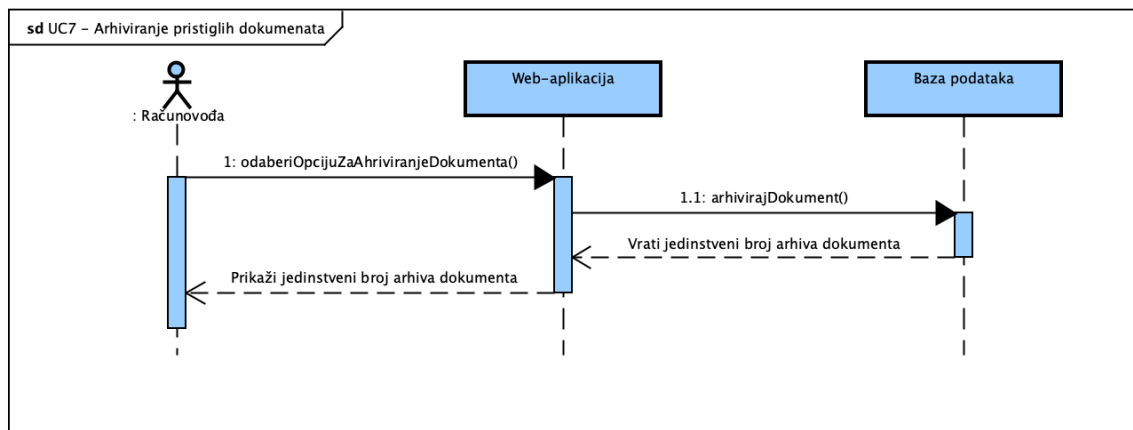
Korisnik na stranici za prijavu u sustav unosi korisničko ime i lozinku. Aplikacija bazi podataka šalje unesene podatke, baza provjerava njihovu ispravnost te vraća podatke o korisniku. Ukoliko je korisnik unio neispravno korisničko ime ili lozinku, baza to javlja aplikaciji koja obavještava korisnika o neuspjeloj prijavi i vraća ga na stranicu za prijavu. Ako su uneseni podaci ispravni sustav korisniku daje pristup korisničkom sučelju.



Slika 3.6: Sekvencijski dijagram za UC01

Obrazac uporabe UC7 - Arhiviranje pristiglih dokumenata

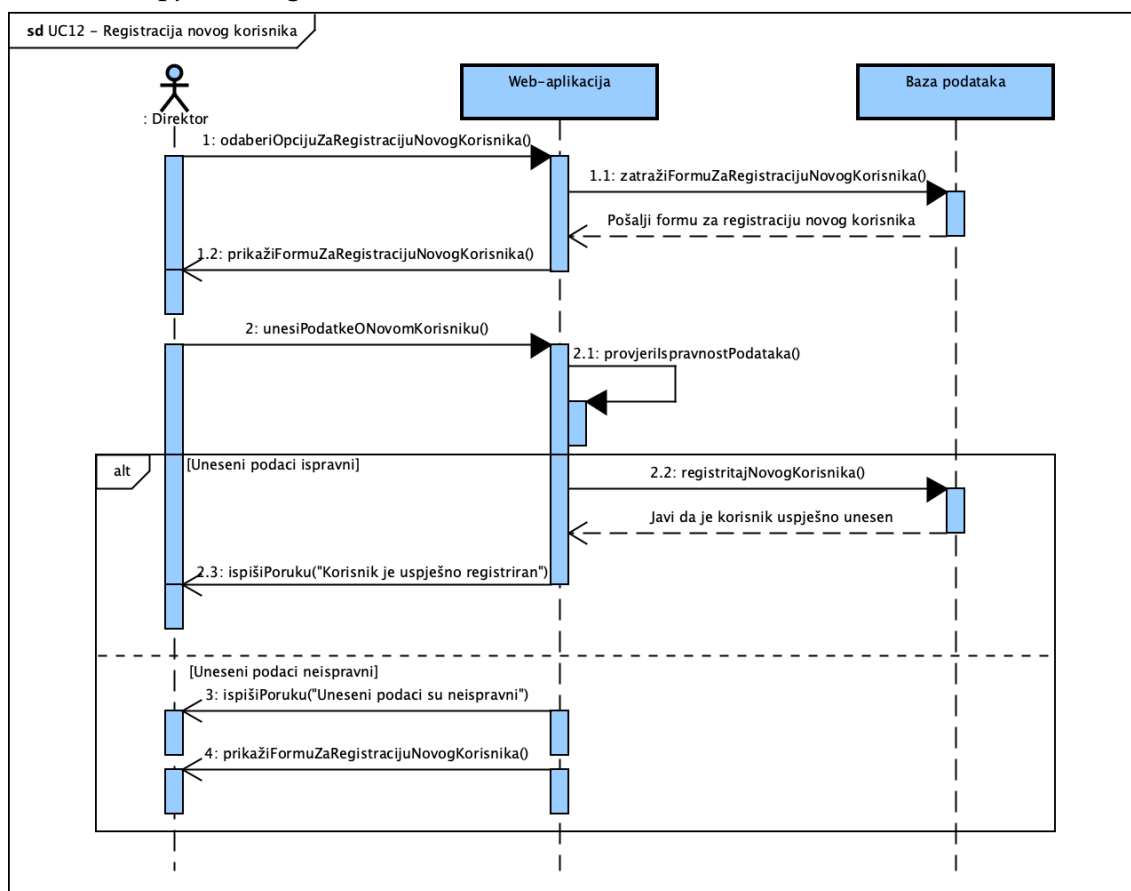
Računovođa aplikaciji šalje zahtjev za prikaz opcija te dabi opciju za arhiviranje pristiglog dokumenta. Aplikacija dojavljuje bazi podataka da arhivira dokument. Baza podataka arhivira dokument i dodjeljuje mu jedinstveni broj arhiva te ga vraća aplikaciji, koja ga prikazuje računovođi.



Slika 3.7: Sekvencijski dijagram za UC07

Obrazac uporabe UC12 - Registracija novog korisnika

Direktor aplikaciji šalje zahtjev za prikaz opcija te odabire opciju za registraciju novog korisnika. Aplikacija bazi podataka šalje zahtjev za formu za registraciju novog korisnika te ju dobiva od baze. Aplikacija prikazuje formu za registraciju novog korisnika. Direktor unosi podatke o novom korisniku, uključujući njegovu ulogu, a aplikacija provjerava ispravnost unesenih podataka. Ukoliko je direktor unio neispravne podatke o novom korisniku, aplikacija obavještava direktora o neuspjeloj registraciji i vraća ga na formu za registraciju novog korisnika. Ako su uneseni podaci ispravni, aplikacija dojavljuje bazi podataka da registrira novog korisnika. Baza podataka pohranjuje podatke o novom korisniku te javlja aplikaciji da je korisnik uspješno unesen. Zatim aplikacija direktoru prosljeđuje da je korisnik uspješno registriran.



Slika 3.8: Sekvencijski dijagram za UC12

3.2 Ostali zahtjevi

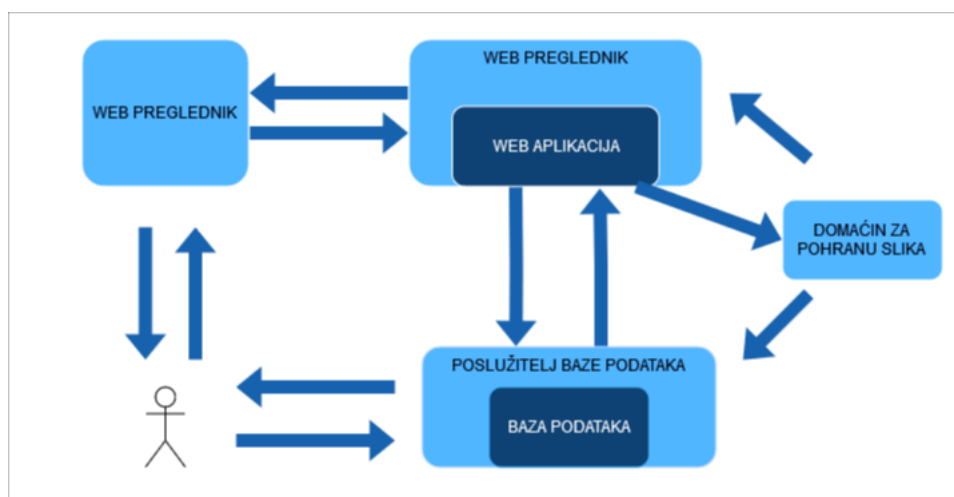
- Sustav treba omogućiti rad više korisnika u stvarnom vremenu
- Vrijeme odziva sustava ne smije prelaziti nekoliko sekundi za bilo koju akciju korisnika

- Sustav treba podržavati istovremeno učitavanje 50 slika s minimalnim gubicima brzine odziva
- Korisničko sučelje i sustav moraju podržavati hrvatsku i englesku abecedu (dijakritičke znakove) pri unosu i prikazu tekstualnog sadržaja
- Veza s bazom podataka mora biti vrlo dobro zaštićena, brza i otporna na vanjske greške
- Korisničko sučelje mora biti jednostavno, intuitivno i prilagođeno korisnicima bez tehničkog predznanja
- Pogrešno korištenje sučelja ne smije dovesti do pada sustava
- Korisničko sučelje mora biti dostupno na najmanje dva jezika: hrvatskom i engleskom
- Vizualni dizajn treba biti responzivan, odnosno prilagodljiv različitim uređajima: računala, tableti, mobilni telefoni
- Sustav mora imati automatsko vraćanje u prethodno stanje u slučaju neočekivanog prekida rada
- Pristup sustavu mora biti omogućen iz javne mreže pomoću HTTPS
- Sustav kao valutu koristi EUR.

4. Arhitektura i dizajn sustava

Arhitektura se može podijeliti na četiri podsustava:

- Web poslužitelj
- Web aplikacija
- Baza podataka
- Domaćin za pohranu slika



Slika 4.1: Arhitektura sustava

Web preglednik je alat koji korisniku omogućava pregled web stranica i svih sadržaja povezanih s isitima. Svaki web preglednik može prevesti kod web aplikacije. Korisnik putem web preglednika šalje zahtjeve na poslužitelj.

Web poslužitelj ima zadaću vršiti komunikaciju između klijenta i aplikacije. Komunikacija se odvija pomoću HTTP protokola (*Hyper Text Transfer Protocol*). Poslužitelj je ili pokretač web aplikacije, ili server na kojem je web aplikacija postavljena.

Web aplikacija služi za obradu zahtjeva. Ovisno o zahtjevu, web aplikacija pristupa bazi podataka ili domaćinu za pohranu slika te korisniku vraća generirani HTML dokument koji je vidljiv u web pregledniku.

Programski jezik koji je korišten za izradu web aplikacije je Python za *backend* i Javascript za *frontend*. Preciznije za *backend* dio koristi se Django, a za *frontend* dio koristi se React. Time smo odvojili *backend* za logiku i obradu podataka, a *frontend* za komunikaciju s korisnikom. Django služi za upravljanje podacima te omogućava definiranje modela podataka i logike aplikacije. React služi za kreiranje korisničkog sučelja. Omogućava brzo i efikasno kreiranje dinamičkih i interaktivnih korisničkih sučelja. Razvojno okruženje u kojem radimo je Microsoft Visual Studio Code.

Domaćin za pohranu slika je vanjski servis na koji se spremaju slike iz korisničkih zahtjeva poslanih iz web preglednika. Za to koristimo `https://imgur.com/`.

4.1 Baza podataka

Za našu aplikaciju koristit ćemo relacijsku bazu podataka koja će nam olakšati modeliranje samog zadatka. Relacijska baza podataka je baza podataka koja je organizirana u skup tablica koje su međusobno povezane. Svaka tablica predstavlja jednu entitetnu klasu, a svaki redak tablice predstavlja jedan objekt dok stupci predstavljaju attribute. Svaki objekt ima svoj jedinstveni identifikator koji se zove primarni ključ, a može imati i strane ključeve koji su referenca na primarni ključ nekog drugog objekta. Baza podataka se koristi za brzu i jednostavnu pohranu podataka koje naknadno možemo lako dohvatiti ili promijeniti. Baza podataka naše aplikacije sadrži slijedeće entitete:

- Korisnik
- Pripada
- Grupa
- SpecijalizacijaRačunovođe
- Dokument
- Račun
- Ponuda
- InterniDokument
- NedefiniraniDokument
- Arhiva
- RačunArhiviran
- PonudaArhivirana
- InterniDokumentArhiviran

- NedefiniraniDokumentArhiviran
- Artikl
- NaRačunu
- UPonudi
- NaArhiviranomRačunu
- UArhiviranojPonudi

4.1.1 Opis tablica

Korisnik: ovaj entitet sadrži sve informacije o korisniku aplikacije. Sadrži attribute: KorisnikId, Ime, Prezime, E-mail, Zaporka, KorisnickoIme, JeAdmin, ZadnjiLogin, JeAktivan, DatumZaposlenja. KorisnikId je primarni ključ. Veze s entitetom Korisnik su: One-to-many veza s entitetom Dokument preko atributa KorisnikId, One-to-many veza s entitetom Dokument preko atributa KorisnikId (ZaPotvrditiOdDirektoraKorisnikId), One-to-many veza s entitetom Dokument preko atributa KorisnikId (ZaPotvrditiOdRevizoraKorisnikId), One-to-many veza s entitetom Dokument preko atributa KorisnikId (ZaPotvrditiOdRačunovođeKorisnikId), One-to-many veza s entitetom Arhiva preko atributa KorisnikId, One-to-many veza s entitetom Arhiva preko atributa KorisnikId (PotvrdioRevizorKorisnikId), One-to-many veza s entitetom Arhiva preko atributa KorisnikId (PotvrdioRačunovođaKorisnikId), One-to-many veza s entitetom Arhiva preko atributa KorisnikId (PotvrdioDirektorKorisnikId), One-to-many veza s entitetom Pripada preko atributa KorisnikId, One-to-many veza s entitetom SpecijalizacijaRačunovođe preko atributa KorisnikId.

Korisnik		
KorisnikId	INT	jedinstveni identifikator korisnika
Ime	VARCHAR	ime korisnika
Prezime	VARCHAR	prezime korisnika
E-mail	VARCHAR	E-mail korisnika
Zaporka	VARCHAR	hash zaporce
KorisnickoIme	VARCHAR	korisničko ime korisnika

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Korisnik		
JeAdmin	BOOLEAN	oznaka je li korisnik admin aplikacije
ZadnjiLogin	DATETIME	datum i vrijeme zadnjeg login-a korisnika
JeAktivan	BOOLEAN	oznaka je li korisnik i dalje zaposlen
DatumZaposlenja	DATETIME	datum i vrijeme kada je korisnik počeo raditi

Pripada: ovaj entitet sadrži sve informacije o tome kojem korisniku pripadaju koje grupe. Sadrži attribute: KorisnikId, GrupaId. KorisnikId i GrupaId su strani ključevi te ujedno i primarni ključ. Veze s entitetom Pripada su: Many-to-one veza s entitetom Korisnik preko atributa KorisnikId, Many-to-one veza s entitetom Grupa preko atributa GrupaId.

Pripada		
KorisnikId	INT	jedinstveni identifikator korisnika (Korisnik.KorisnikId)
GrupaId	INT	jedinstveni identifikator grupe (Grupa.GrupaId)

Grupa: ovaj entitet sadrži sve informacije o grupama. Sadrži attribute: GrupaId, ImeGrupe. GrupaId je primarni ključ. Veza s entitetom Grupa je: One-to-many veza s entitetom Pripada preko atributa GrupaId.

Grupa		
GrupaId	INT	jedinstveni identifikator grupe
imeGrupe	VARCHAR	naziv grupe

SpecijalizacijaRačunovođe: ovaj entitet sadrži sve informacije o specijalizacijama računovođa. Sadrži attribute: SpecijalizacijaId, SpecijalizacijaIme, KorisnikId. SpecijalizacijaId je primarni ključ. KorisnikId je strani ključ. Veza s entitetom SpecijalizacijaRačunovođe su: Many-to-one veza s entitetom Korisnik preko atributa

KorisnikId.

SpecijalizacijaRačunovođe		
SpecijalizacijaId	INT	jedinstveni identifikator specijalizacije
SpecijalizacijaIme	VARCHAR	naziv specijalizacije
KorisnikId	INT	jedinstveni identifikator računovođe koji je specijaliziran za određenu vrstu dokumenata (Korisnik.KorisnikId)

Dokument: ovaj entitet sadrži sve informacije o dokumentima. Sadrži attribute: DokumentId, TekstDokumenta, LinkSlike, VrijemeSkeniranja, PotpisaoDirektor, PotvrdioRevizor, PregledaoRačunovođa, KorisnikId, ZaPotvrditiOdDirektoraKorisnikId, ZaPotvrditiOdRevizoraKorisnikId, ZaPotvrditiOdRačunovođeKorisnikId. DokumentId je primarni ključ. KorisnikId, ZaPotvrditiOdDirektoraKorisnikId, ZaPotvrditiOdRevizoraKorisnikId, ZaPotvrditiOdRačunovođeKorisnikId su strani ključevi. Veze s entitetom Dokument su: Many-to-one veza s entitetom Korisnik preko atributa KorisnikId, Many-to-one veza s entitetom Korisnik preko atributa KorisnikId (ZaPotvrditiOdDirektoraKorisnikId), Many-to-one veza s entitetom Korisnik preko atributa KorisnikId (ZaPotvrditiOdRevizoraKorisnikId), Many-to-one veza s entitetom Korisnik preko atributa KorisnikId (ZaPotvrditiOdRačunovođeKorisnikId), One-to-one veza s entitetom Račun preko atributa DokumentId, One-to-one veza s entitetom Ponuda preko atributa DokumentId, One-to-one veza s entitetom InterniDokument preko atributa DokumentId, One-to-one veza s entitetom NedefiniraniDokument preko atributa DokumentId.

Dokument		
DokumentId	INT	jedinstveni identifikator dokumenta
TekstDokumenta	VARCHAR	tekst skeniranog dokumenta

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Dokument		
LinkSlike	VARCHAR	poveznica s poslužitelja na kojem se nalazi spremljena slika dokumenta
VrijemeSkeniranja	DATETIME	datum i vrijeme kada je napravljeno skeniranje slike dokumenta
PotpisaoDirektor	BOOLEAN	oznaka je li direktor potpisao dokument
PotvrdioRevizor	BOOLEAN	oznaka je li revizor potvrdio dokument
PregledaoRačunovođa	BOOLEAN	oznaka je li računovođa pregledao dokument
KorisnikId	INT	jedinstveni identifikator korisnika koji je skenirao dokument(Korisnik.KorisnikId)
ZaPotvrditiOdDirektoraKorisnikId	INT	jedinstveni identifikator direktora koji mora potvrditi dokument (Korisnik.KorisnikId)
ZaPotvrditiOdRevizoraKorisnikId	INT	jedinstveni identifikator revizora koji mora potvrditi dokument (Korisnik.KorisnikId)
ZaPotvrditiOdRačunovođeKorisnikId	INT	jedinstveni identifikator računovođe koji mora potvrditi dokument (Korisnik.KorisnikId)

Račun: ovaj entitet sadrži sve informacije o računima. Sadrži attribute: Doku-

mentId, UkupnaCijena, ImeKlijenta. DokumentId je strani ključ. Veze s entitetom Račun su: One-to-one veza s entitetom Dokument preko atributa DokumentId, One-to-many veza s entitetom NaRačunu preko atributa DokumentId.

Račun		
DokumentId	INT	jedinstveni identifikator dokumenta (Dokument.DokumentId)
UkupnaCijena	DECIMAL	ukupna cijena svih artikala na računu
ImeKlijenta	VARCHAR	ime klijenta

Ponuda: ovaj entitet sadrži sve informacije o ponudama. Sadrži attribute: DokumentId, UkupnaCijena. DokumentId je strani ključ. Veze s entitetom Ponuda su: One-to-one veza s entitetom Dokument preko atributa DokumentId, One-to-many veza s entitetom UPonudi preko atributa DokumentId.

Ponuda		
DokumentId	INT	jedinstveni identifikator dokumenta (Dokument.DokumentId)
UkupnaCijena	DECIMAL	ukupna cijena svih artikala u ponudi

InterniDokument: ovaj entitet sadrži sve informacije o internim dokumentima. Sadrži atribut: DokumentId. DokumentId je strani ključ. Veze s entitetom InterniDokument su: One-to-one veza s entitetom Dokument preko atributa DokumentId.

InterniDokument		
DokumentId	INT	jedinstveni identifikator dokumenta (Dokument.DokumentId)

NedefiniraniDokument: ovaj entitet sadrži sve informacije o nedefiniranim dokumentima. Sadrži atribut: DokumentId. DokumentId je strani ključ. Veze s entitetom NedefiniraniDokument su: One-to-one veza s entitetom Dokument preko atributa DokumentId.

NedefiniraniDokument		
DokumentId	INT	jedinstveni identifikator dokumenta (Dokument.DokumentId)

Arhiva: ovaj entitet sadrži sve informacije o arhiviranim dokumentima. Sadrži attribute: ArhivaId, TekstDokumenta, LinkSlike, VrijemeSkeniranja, DokumentId, VrijemeArhiviranja, PotpisaoDirektor, PotvrdioRevizor, PregledaoRačunovođa, KorisnikId, ZaPotvrditiOdDirektoraKorisnikId, ZaPotvrditiOdRevizoraKorisnikId, ZaPotvrditiOdRačunovođeKorisnikId. ArhivaId je primarni ključ. KorisnikId, ZaPotvrditiOdDirektoraKorisnikId, ZaPotvrditiOdRevizoraKorisnikId, ZaPotvrditiOdRačunovođeKorisnikId su strani ključevi. Veze s entitetom Arhiva su: Many-to-one veza s entitetom Korisnik preko atributa KorisnikId, Many-to-one veza s entitetom Korisnik preko atributa KorisnikId (ZaPotvrditiOdDirektoraKorisnikId), Many-to-one veza s entitetom Korisnik preko atributa KorisnikId (ZaPotvrditiOdRevizoraKorisnikId), Many-to-one veza s entitetom Korisnik preko atributa KorisnikId (ZaPotvrditiOdRačunovođeKorisnikId), One-to-one veza s entitetom RačunArhiviran preko atributa ArhivaId, One-to-one veza s entitetom PonudaArhivirana preko atributa ArhivaId, One-to-one veza s entitetom InterniDokumentArhiviran preko atributa ArhivaId, One-to-one veza s entitetom NedefiniraniDokumentArhiviran preko atributa ArhivaId.

Arhiva		
ArhivaId	INT	jedinstveni identifikator arhive
TekstDokumenta	VARCHAR	tekst arhiviranog dokumenta
LinkSlike	VARCHAR	poveznica s poslužitelja na kojem se nalazi spremljena slika arhiviranog dokumenta

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Arhiva		
VrijemeSkeniranja	DATETIME	datum i vrijeme kada je napravljeno skeniranje slike arhiviranog dokumenta
DokumentId	INT	jedinstveni identifikator dokumenta
VrijemeArhiviranja	DATETIME	datum i vrijeme kada je dokument arhiviran
PotpisaoDirektor	BOOLEAN	oznaka je li direktor potpisao arhivirani dokument
PotvrdioRevizor	BOOLEAN	oznaka je li revizor potvrdio arhivirani dokument
PregledaoRačunovođa	BOOLEAN	oznaka je li računovođa pregledao arhivirani dokument
KorisnikId	INT	jedinstveni identifikator korisnika koji je skenirao arhivirani dokument(Korisnik.KorisnikId)
ZaPotvrditiOdDirektoraKorisnikId	INT	jedinstveni identifikator direktora koji je morao potvrditi dokument prije arhiviranja(Korisnik.KorisnikId)

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Arhiva		
ZaPotvrditiOdRevizoraKorisnikId	INT	jedinstveni identifikator revizora koji je morao potvrditi dokument prije arhiviranja (Korisnik.KorisnikId)
ZaPotvrditiOdRačunovođeKorisnikId	INT	jedinstveni identifikator računovođe koji je morao potvrditi dokument prije arhiviranja (Korisnik.KorisnikId)

RačunArhiviran: ovaj entitet sadrži sve informacije o računima koji su arhivirani. Sadrži attribute: ArhivaId, UkupnaCijena, ImeKlijenta. ArhivaId je strani ključ. Veze s entitetom Račun su: One-to-one veza s entitetom Arhiva preko atributa ArhivaId, One-to-many veza s entitetom NaArhiviranomRačunu preko atributa ArhivaId.

RačunArhiviran		
ArhivaId	INT	jedinstveni identifikator arhive (Arhiva.ArhivaId)
UkupnaCijena	DECIMAL	ukupna cijena svih artikala na arhiviranom računu
ImeKlijenta	VARCHAR	ime klijenta na arhiviranom računu

PonudaArhivirana: ovaj entitet sadrži sve informacije o ponudama koje su arhivirane. Sadrži attribute: ArhivaId, UkupnaCijena. ArhivaId je strani ključ. Veze s entitetom Ponuda su: One-to-one veza s entitetom Arhiva preko atributa ArhivaId, One-to-many veza s entitetom UArhiviranojPonudi preko atributa ArhivaId.

PonudaArhivirana		
ArhivaId	INT	jedinstveni identifikator arhive (Arhiva.ArhivaId)
UkupnaCijena	DECIMAL	ukupna cijena svih artikala u arhiviranoj ponudi

InterniDokumentArhiviran: ovaj entitet sadrži sve informacije o internim dokumentima koji su arhivirani. Sadrži atribut: ArhivaId. ArhivaId je strani ključ. Veze s entitetom InterniDokument su: One-to-one veza s entitetom Arhiva preko atributa ArhivaId.

InterniDokumentArhiviran		
ArhivaId	INT	jedinstveni identifikator arhive (Arhiva.ArhivaId)

NedefiniraniDokumentArhiviran: ovaj entitet sadrži sve informacije o nedefiniranim dokumentima koji su arhivirani. Sadrži atribut: ArhivaId. ArhivaId je strani ključ. Veze s entitetom NedefiniraniDokument su: One-to-one veza s entitetom Arhiva preko atributa ArhivaId.

NedefiniraniDokumentArhiviran		
ArhivaId	INT	jedinstveni identifikator arhive (Arhiva.ArhivaId)

Artikl: ovaj entitet sadrži sve informacije o artiklima. Sadrži attribute: ImeArtikla, Cijena. ImeArtikla je primarni ključ. Veze s entitetom Artikl su: One-to-many veza s entitetom NaRačunu preko atributa ImeArtikla, One-to-many veza s entitetom UPonudi preko atributa ImeArtikla, One-to-many veza s entitetom NaArhiviranomRačunu preko atributa ImeArtikla, One-to-many veza s entitetom UArhiviranojPonudi preko atributa ImeArtikla.

Artikl		
ImeArtikla	VARCHAR	jedinstveno ime artikla

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Artikl		
Cijena	DECIMAL	cijena artikla

NaRačunu: ovaj entitet sadrži sve informacije o tome koji artikli se nalaze na računu. Sadrži attribute: DokumentId, ImeArtikla. DokumentId i ImeArtikla su strani ključevi. Veze s entitetom NaRačunu su: Many-to-one veza s entitetom Račun preko atributa DokumentId, Many-to-one veza s entitetom Artikl preko atributa ImeArtikla.

NaRačunu		
DokumentId	INT	jedinstveni identifikator dokumenta (Dokument.DokumentId)
ImeArtikla	VARCHAR	jedinstveno ime artikla (Artikl.ImeArtikla)

UPonudi: ovaj entitet sadrži sve informacije o tome koji artikli se nalaze u ponudi. Sadrži attribute: DokumentId, ImeArtikla. DokumentId i ImeArtikla su strani ključevi. Veze s entitetom UPonudi su: Many-to-one veza s entitetom Ponuda preko atributa DokumentId, Many-to-one veza s entitetom Artikl preko atributa ImeArtikla.

UPonudi		
DokumentId	INT	jedinstveni identifikator dokumenta (Dokument.DokumentId)
ImeArtikla	VARCHAR	jedinstveno ime artikla (Artikl.ImeArtikla)

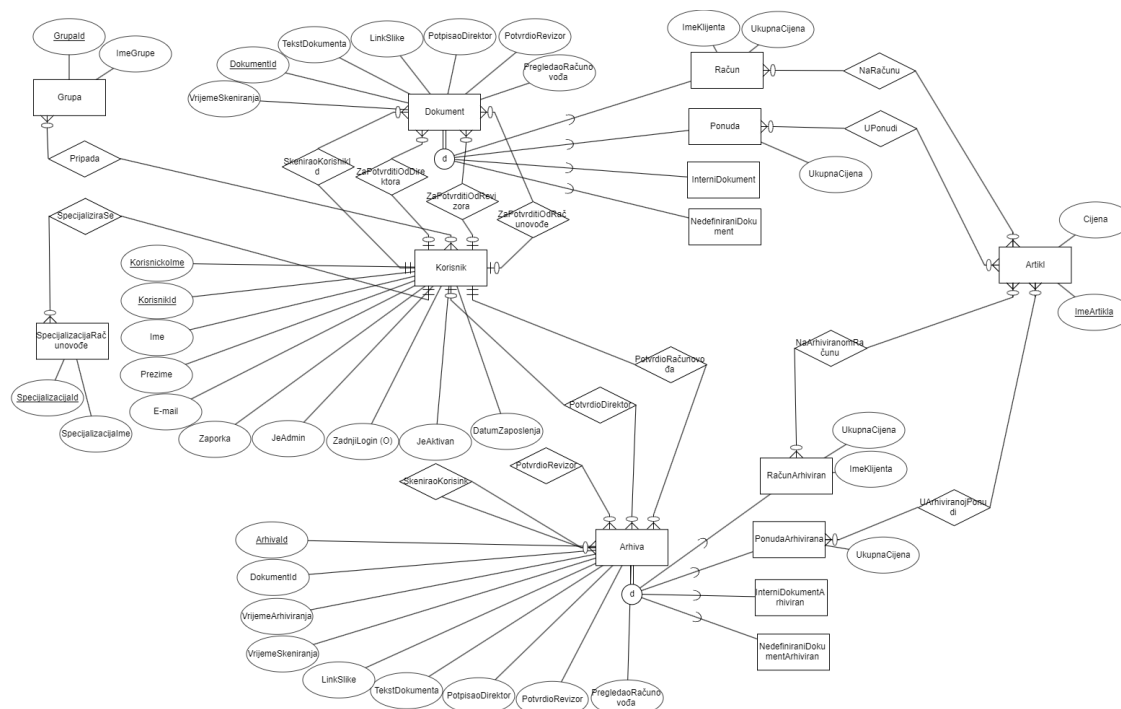
NaArhiviranomRačunu: ovaj entitet sadrži sve informacije o tome koji artikli se nalaze na arhiviranom računu. Sadrži attribute: ArhivaId, ImeArtikla. ArhivaId i ImeArtikla su strani ključevi. Veze s entitetom NaArhiviranomRačunu su: Many-to-one veza s entitetom RačunArhiviran preko atributa ArhivaId, Many-to-one veza s entitetom Artikl preko atributa ImeArtikla.

NaArhiviranomRačunu		
ArhivaId	INT	jedinstveni identifikator arhive (Arhiva.ArhivaId)
ImeArtikla	VARCHAR	jedinstveno ime artikla (Artikl.ImeArtikla)

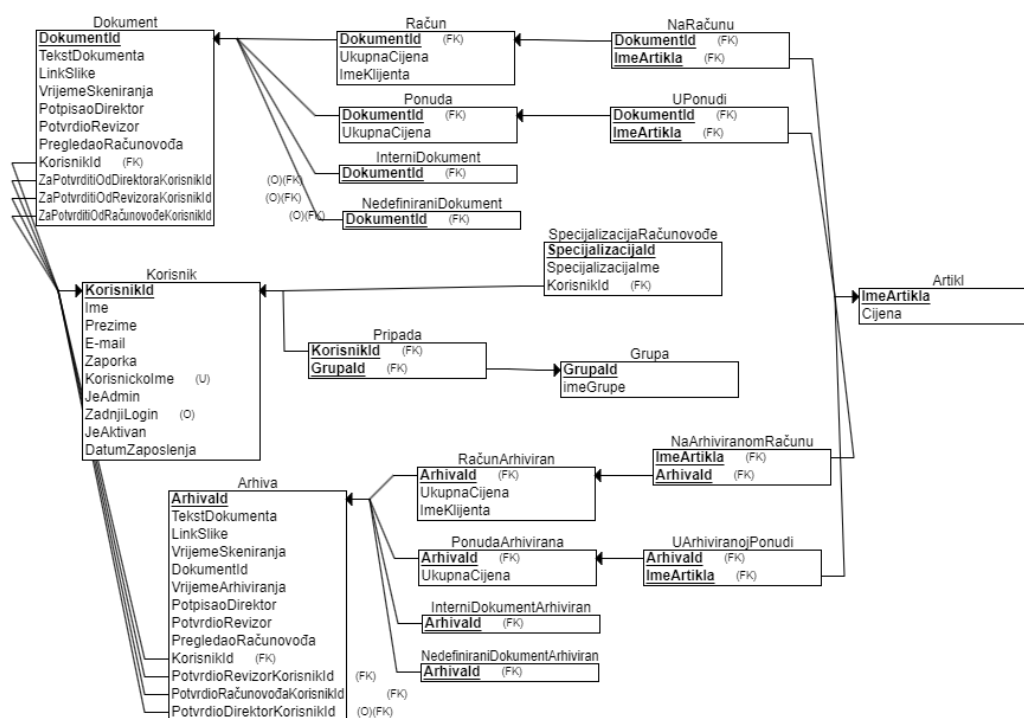
UArhiviranojPonudi: ovaj entitet sadrži sve informacije o tome koji artikli se nalaze u arhiviranoj ponudi. Sadrži attribute: ArhivaId, ImeArtikla. ArhivaId i ImeArtikla su strani ključevi. Veze s entitetom UArhiviranojPonudi su: Many-to-one veza s entitetom PonudaArhivirana preko atributa ArhivaId, Many-to-one veza s entitetom Artikl preko atributa ImeArtikla.

UArhiviranojPonudi		
ArhivaId	INT	jedinstveni identifikator arhive (Arhiva.ArhivaId)
ImeArtikla	VARCHAR	jedinstveno ime artikla (Artikl.ImeArtikla)

4.1.2 Dijagram baze podataka



Slika 4.2: ER dijagram baze podataka



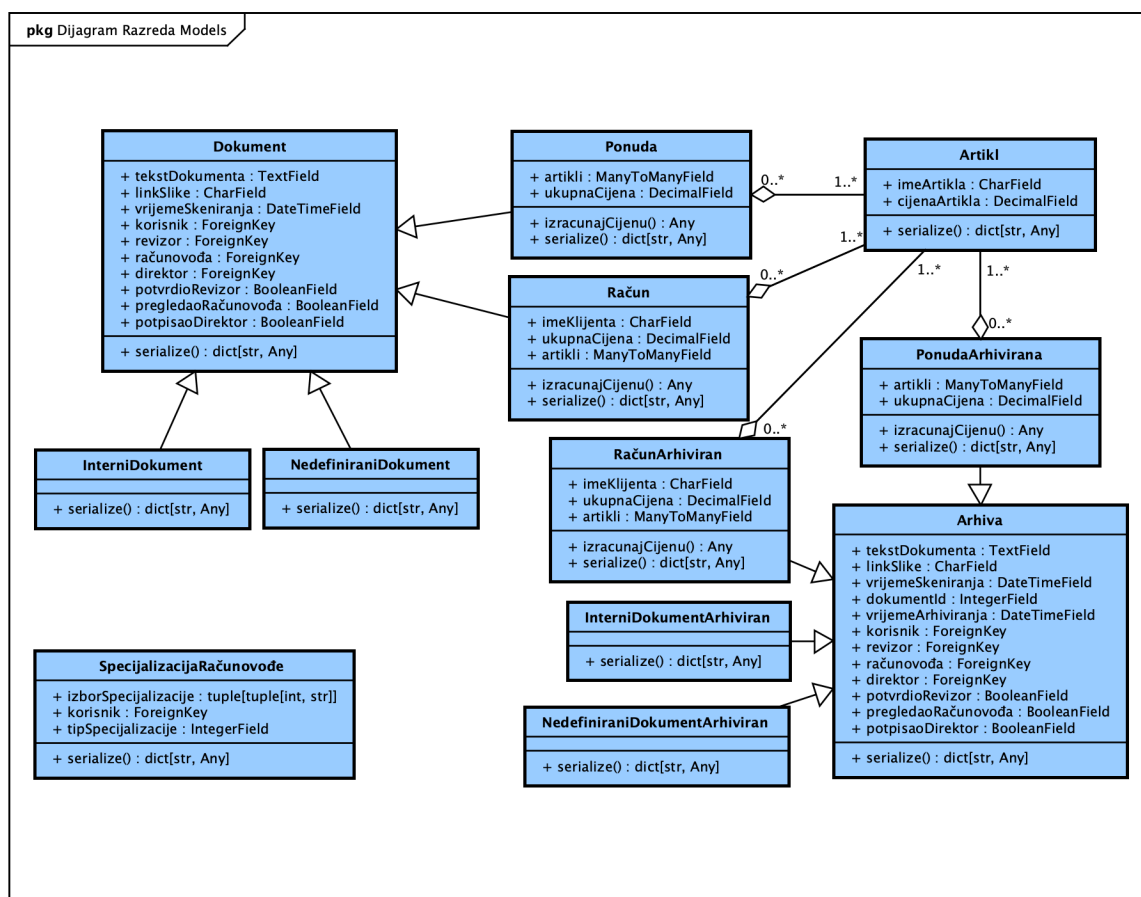
Slika 4.3: Relacijska shema baze podataka

4.2 Dijagram razreda

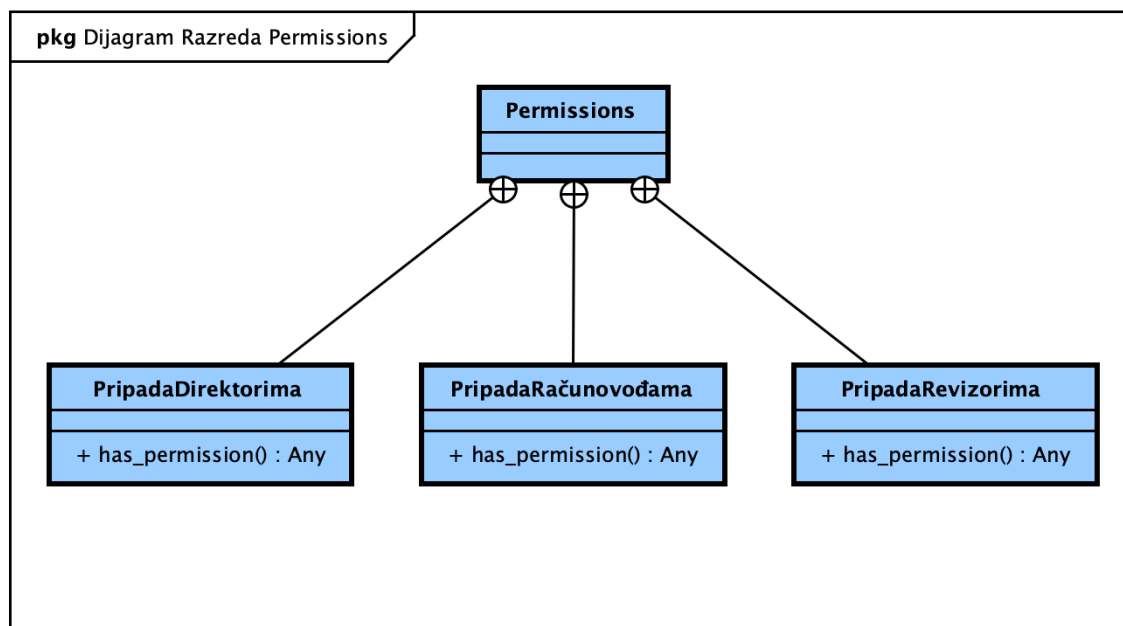
Na dijagramu razreda Models (slika 4.4) prikazani su razredi unutar datoteke models.py te odnosi među njima. U datoteci postoji ukupno 12 razreda. Dokument je nadrazred 4 od tih razreda, a Arhiva je također nadrazred 4 od tih razreda.

Dijagram razreda Permissions (slika 4.5) prikazuje razred Permissions, koji je nadrazred razredima PripadaDirektorima, PripadaRačunovođama i PripadaRevizorima. Ti se razredi nalaze unutar datoteke permissions.py te definiraju kojoj grupi korisnik mora pripadati da bi mogao koristiti određeni prikaz web-aplikacije.

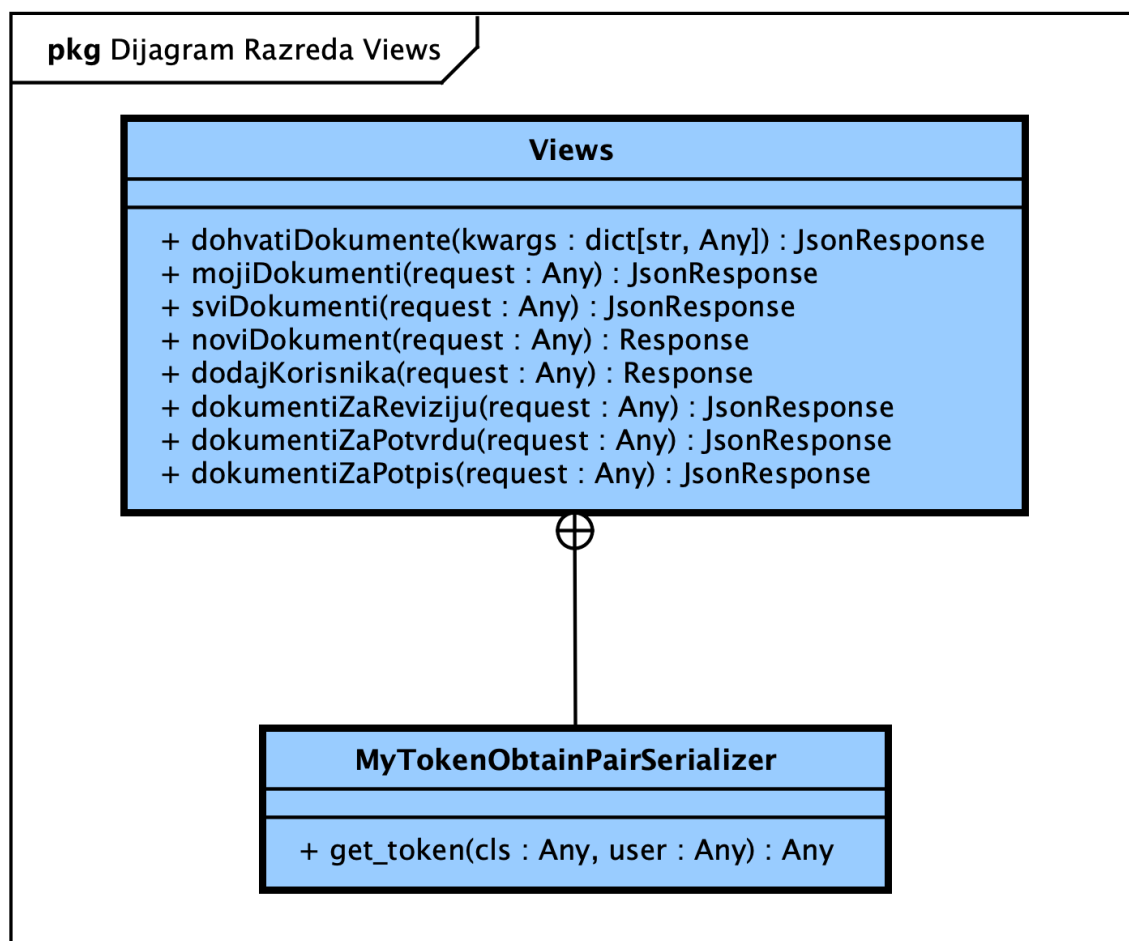
Razred Views nalazi se unutar datoteke views.py, u kojoj se također nalazi i razred MyTokenObtainPairSerializer kojemu je Views nadrazred. Views koristi taj razred kako bi serveru bilo preneseno koje dozvole su prisutne tijekom pokušaja pristupa aplikaciji. Na dijagramu razreda Views (slika 4.6) prikazan je odnos između ta dva razreda.



Slika 4.4: Dijagram razreda - dio Models



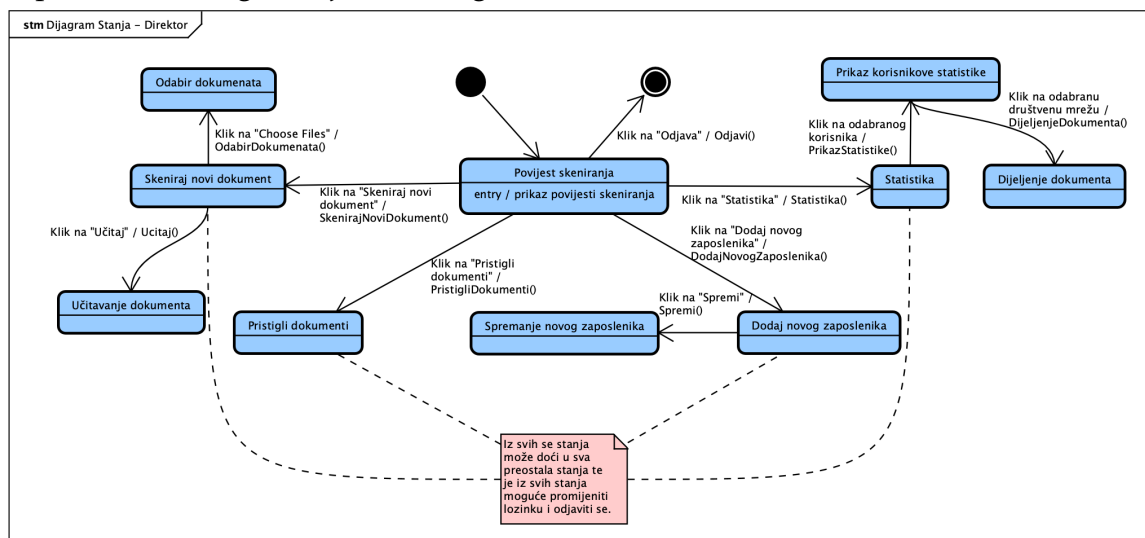
Slika 4.5: Dijagram razreda - dio Permissions



Slika 4.6: Dijagram razreda - dio Views

4.3 Dijagram stanja

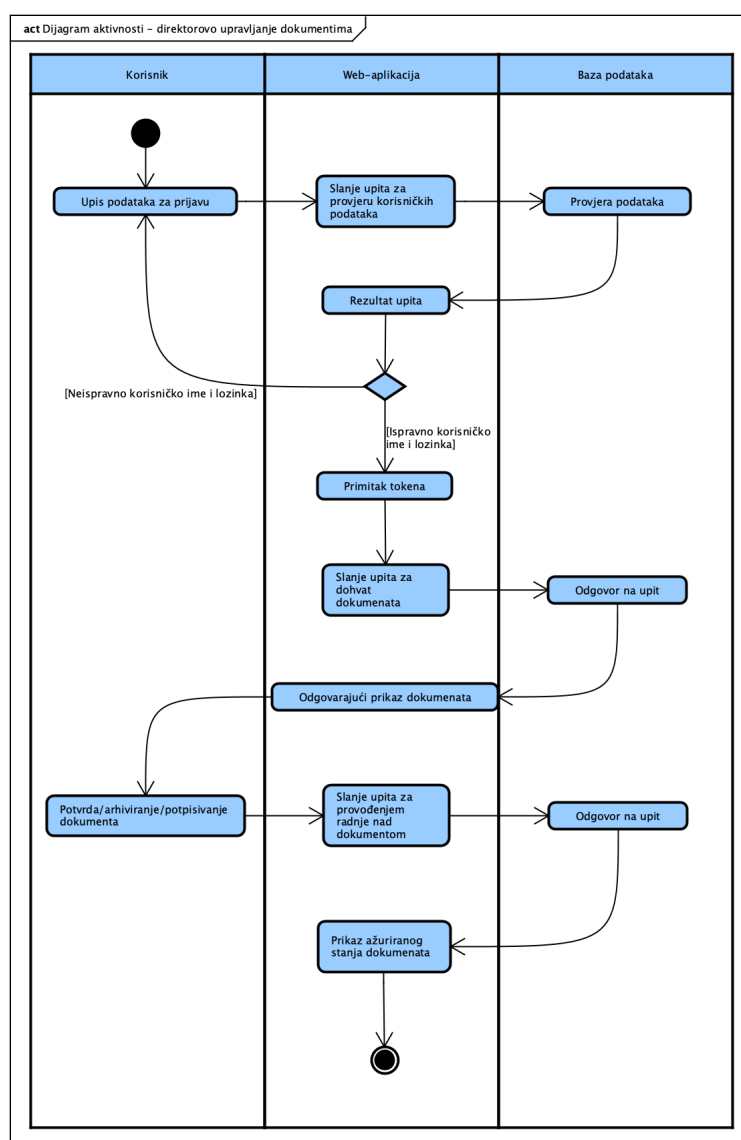
Na slici 4.7 prikazan je dijagram stanja za direktora nakon obavljene prijave. Prvo što se direktoru prikazuje je početna stranica, koja se također prikaže i kada se klikne na "Povijest skeniranja". Klikom na "Skeniraj novi dokument" otvara se prikaz za odabira novog dokumenta s računala te se za odabrani dokument nudi opcija njegova učitavanja. Osim toga, s početne se stranice može ući u pregled pristiglih dokumenata te pregled statistike dokumenata svakog postojećeg korisnika, koja uključuje dijeljenje dokumenata na jednu od ponuđenih društvenih mreža. Odabirom "Dodaj novog zaposlenika" također je moguće dodati i spremiti novog zaposlenika s odgovarajućom ulogom.



Slika 4.7: Dijagram stanja

4.4 Dijagram aktivnosti

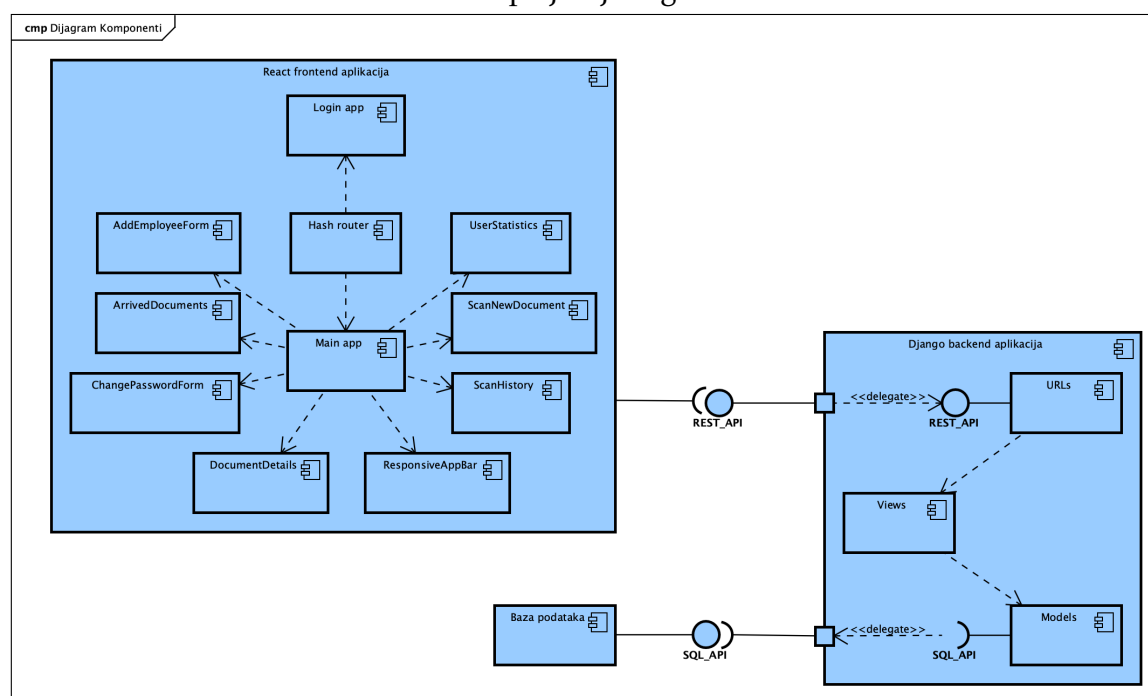
Dijagramom aktivnosti na slici 4.8 jednostavno je prikazan tok za direktorovu prijavu u sustav te upravljanje dokumentima. Aktivnosti na dijagramu odvijaju se slijedno, bez preklapanja. Nakon što se direktor uspješno prijavi u sustav, on dobiva svoj token pomoću kojega web-aplikacija može odrediti ulogu prijavljenog korisnika te mu omogućiti odgovarajući prikaz dokumenata. Direktor zatim može napraviti odabranu radnju nad dokumentima između potvrde, arhiviranja i potpisivanja. Nakon izvršavanja neke od tih radnji, direktoru se prikazuje ažurirano stanje dokumenata.



Slika 4.8: Dijagram aktivnosti

4.5 Dijagram komponenti

Na slici 4.9 prikazan je dijagram komponenti. Korisnik pristupa Django backend aplikaciji koristeći React frontend aplikaciju i sučelje REST_API. Django backend aplikacija organizirana je modularno; modul URLs komunicira s modulom Views. On komunicira s modulom Models koji je zadužen za dohvat podataka iz baze preko sučelja SQL_API. Komponenta Hash router unutar React frontend aplikacije na temelju URL-a određuje hoće li se na sučelje poslužiti Login ili Main aplikacija. Main aplikacija sastoji se od nekoliko JavaScript datoteka koje određuju prikaze određenih elemenata na stranici za prijavljenog korisnika.



Slika 4.9: Dijagram komponenti

5. Implementacija i korisničko sučelje

5.1 Korištene tehnologije i alati

Svi članovi tima su sudjelovali u odabiru tehnologija i alata koji će se koristiti za izradu aplikacije. Kao sredstvo komunikacije, odabrana je aplikacija WhatsApp¹. Pomoću WhatsApp-a, članovi tima su mogli komunicirati u stvarnom vremenu, razmjenjivati datoteke, te se dogovarati o terminima sastanaka. Za izradu UML dijagrama korišten je alat Astah Professional² dok se Git³ koristio kao sustav za upravljanje izvornim kodom. Na web platformi GitHub⁴ je dostupan udaljeni repozitorij projekta. Za izradu dokumentacije korišten je LaTeX⁵. Za razvojno okruženje korišten je Visual Studio Code⁶ budući da je vrlo popularan za razvoj web i mobilnih aplikacija kao i drugih programa te je vrlo pregledan i jednostavan za korištenje. Za izradu naše web aplikacije korišteni su biblioteka React⁷ i jezik Javascript⁸ za izradu frontenda. React je biblioteka za izradu korisničkih sučelja koja je održavana od strane Facebooka. React se koristi za izradu jednostraničnih aplikacija (engl. single-page application) ili mobilnih aplikacija. React se fokusira na izradu korisničkog sučelja, dok se za ostale funkcionalnosti aplikacije koristi JavaScript - dinamički tipizirani programski jezik koji je jedan od najpopularnijih na svijetu za izradu web aplikacija. Za izradu backenda korišten je jezik Python⁹ i biblioteka Django¹⁰. Django, web framework za Python, poslužio je za razvoj backend dijela aplikacije, dok se za ostale funkcionalnosti koristi Python - još jedan dinamički tipizirani programski jezik često korišten za izradu web aplikacija.

¹<https://www.whatsapp.com/>

²<http://astah.net/editions/professional>

³<https://git-scm.com/>

⁴<https://github.com/>

⁵<https://www.latex-project.org/>

⁶<https://code.visualstudio.com/>

⁷<https://reactjs.org/>

⁸<https://www.javascript.com/>

⁹<https://www.python.org/>

¹⁰<https://www.djangoproject.com/>

5.2 Ispitivanje programskog rješenja

5.2.1 Ispitivanje komponenti

```
class ViewTest(TestCase):
    def setUp(self):
        Group.objects.create(name='Zaposlenici')
        Group.objects.create(name='Direktori')

        self.base_url = "/api/"
        self.zaposlenik = User.objects.create_user(
            username='test', password='test')
        Group.objects.get(name='Zaposlenici').
            user_set.add(self.zaposlenik)
        self.direktor = User.objects.create_user(
            username='test2', password='test2')
        Group.objects.get(name='Direktori').user_set
            .add(self.direktor)

        self.client = Client()
        self.zaposlenik_token = self.client.post(
            self.base_url + 'token/', {'username': '
            test', 'password': 'test'}).data.get('
            access')
        self.direktor_token = self.client.post(self.
            base_url + 'token/', {'username': 'test2
            ', 'password': 'test2'}).data.get('access
            ')

        self.interni_dokument1 = InterniDokument.
            objects.create(tekstDokumenta='tekst 1',
            linkSlike='link 1', vrijemeSkeniranja=
            timezone.datetime(2021, 5, 5, 10, 10, 10,
            tzinfo=pytz.UTC), korisnik=self.direktor
            )
```



```
self.interni_dokument2 = InterniDokument.  
    objects.create(tekstDokumenta='tekst 2',  
        linkSlike='link 2', vrijemeSkeniranja=  
            timezone.datetime(2021, 5, 5, 10, 10, 10,  
                tzinfo=pytz.UTC), korisnik=self.  
                zaposlenik)  
  
# Testovi funkcionalnosti korisnika  
  
def test_promijeni_lozinku(self):  
    self.assertTrue(self.zaposlenik.  
        check_password('test'))  
  
    response = self.client.put(  
        self.base_url + 'promijeniLozinku/',  
        {"trenutnaLozinka": "test", "  
            novaLozinka": "test"},  
        HTTP_AUTHORIZATION='Bearer ' + self.  
            zaposlenik_token ,  
        content_type='application/json'  
    )  
    self.assertEqual(response.status_code , 418)  
  
    response = self.client.put(  
        self.base_url + 'promijeniLozinku/',  
        {"trenutnaLozinka": "test", "  
            novaLozinka": "new_password"},  
        HTTP_AUTHORIZATION='Bearer ' + self.  
            zaposlenik_token ,  
        content_type='application/json'  
    )  
    self.assertEqual(response.status_code , 200)  
    self.zaposlenik.refresh_from_db()
```

```
self.assertFalse(self.zaposlenik.
    check_password('test'))
self.assertTrue(self.zaposlenik.
    check_password('new_password'))

def test_dodaj_korisnika(self):
    response = self.client.post(
        self.base_url + 'dodajKorisnika/',
        {"username": "test3", "password": "
            test3"},
        HTTP_AUTHORIZATION='Bearer ' + self.
            zaposlenik_token,
        content_type='application/json'
    )
    self.assertEqual(response.status_code, 403)

    response = self.client.post(
        self.base_url + 'dodajKorisnika/',
        {"username": "test3", "password": "
            test3", "email": "abc@example.com
            ", "ime": "test", "prezime": "
            test", "group": "Zaposlenici"},
        HTTP_AUTHORIZATION='Bearer ' + self.
            direktor_token,
        content_type='application/json'
    )
    self.assertEqual(response.status_code, 201)
    self.assertTrue(User.objects.filter(username
        ="test3").exists())
    user = User.objects.get(username="test3")
    self.assertTrue(Group.objects.get(name='
        Zaposlenici').user_set.filter(username="
        test3").exists())
    self.assertTrue(user.check_password('test3'))
    )
```

```
self.assertEqual(user.email, "abc@example.com")
self.assertEqual(user.first_name, "test")
self.assertEqual(user.last_name, "test")

def test_dohvati_korisnike_grupe(self):
    response = self.client.get(
        self.base_url + '
            dohvatiKorisnikeGrupe/Zaposlenici
        ',
        HTTP_AUTHORIZATION='Bearer ' + self.
            zaposlenik_token,
        content_type='application/json'
    )

    self.assertEqual(response.status_code, 200)
    data = response.json()
    self.assertEqual(data['korisnici'], [{ 'id':
        self.zaposlenik.id, 'username': 'test' }])

    response = self.client.get(
        self.base_url + '
            dohvatiKorisnikeGrupe/Direktori ',
        HTTP_AUTHORIZATION='Bearer ' + self.
            direktor_token,
        content_type='application/json'
    )

    self.assertEqual(response.status_code, 200)
    data = response.json()
    self.assertEqual(data['korisnici'], [{ 'id':
        self.direktor.id, 'username': 'test2' }])

def test_dohvati_specijalizirane_racunovodje(self):
    Group.objects.create(name='Računovođe')
```

```
računovođa = User.objects.create_user(  
    username='test3 ', password='test3 ')  
Group.objects.get(name='Računovođe').  
    user_set.add(ráčunovođa)  
  
SpecijalizacijaRačunovođe.objects.create(  
    korisnik=računovođa, tipSpecijalizacije  
    =0)  
  
response = self.client.get(  
    self.base_url + '  
        dohvatiSpecijaliziraneRačunovođe/  
        abc',  
    HTTP_AUTHORIZATION='Bearer ' + self.  
        zaposlenik_token ,  
    content_type='application/json '  
)  
self.assertEqual(response.status_code , 400)  
  
response = self.client.get(  
    self.base_url + '  
        dohvatiSpecijaliziraneRačunovođe/  
        Računi',  
    HTTP_AUTHORIZATION='Bearer ' + self.  
        zaposlenik_token ,  
    content_type='application/json '  
)  
  
self.assertEqual(response.status_code , 200)  
data = response.json()  
self.assertEqual(data['korisnici'], [{ 'id':  
    računovođa.id, 'username': 'test3 ' }])  
  
response = self.client.get(  

```

```
        self.base_url + '
            dohvatiSpecijaliziraneRačunovođe/
            Ponude',
        HTTP_AUTHORIZATION='Bearer ' + self.
            zaposlenik_token ,
        content_type='application/json'
    )

    self.assertEqual(response.status_code , 200)
    data = response.json()
    self.assertEqual(data['korisnici'], [])

# Testovi funkcionalnosti rada s dokumentima

def test_moji_dokumenti(self):
    response = self.client.get(
        self.base_url + 'mojiDokumenti/',
        HTTP_AUTHORIZATION='Bearer ' + self.
            direktor_token ,
        content_type='application/json'
    )
    self.assertEqual(response.status_code , 200)
    data = response.json()
    self.assertEqual(len(data['dokumenti']), 1)
    self.assertEqual(data['dokumenti'][0]['id'],
        self.interni_dokument1.id)

def test_svi_dokumenti(self):
    response = self.client.get(
        self.base_url + 'sviDokumenti/',
        HTTP_AUTHORIZATION='Bearer ' + self.
            direktor_token ,
        content_type='application/json'
    )
```

```
self.assertEqual(response.status_code, 200)
data = response.json()
self.assertEqual(len(data['dokumenti']), 2)
self.assertEqual(data['dokumenti'][0]['id'],
                  self.interni_dokument1.id)
self.assertEqual(data['dokumenti'][1]['id'],
                  self.interni_dokument2.id)

def test_označi_točnost_skeniranja(self):
    self.assertFalse(self.interni_dokument2.toč
                     noSkeniran)
    response = self.client.put(
        self.base_url + 'označiToč
        nostSkeniranja/' + str(self.
        interni_dokument2.id),
        {"tocnost": True},
        HTTP_AUTHORIZATION='Bearer ' + self.
        zaposlenik_token,
        content_type='application/json'
    )
    self.assertEqual(response.status_code, 200)
    self.interni_dokument2.refresh_from_db()
    self.assertTrue(self.interni_dokument2.toč
                    noSkeniran)

def test_potpiši(self):
    self.assertFalse(self.interni_dokument1.
                     potpisaoDirektor)
    response = self.client.put(
        self.base_url + 'potpiši/' + str(
        self.interni_dokument1.id),
        HTTP_AUTHORIZATION='Bearer ' + self.
        direktor_token,
        content_type='application/json'
    )
```

```
self.assertEqual(response.status_code, 200)
self.interni_dokument1.refresh_from_db()
self.assertTrue(self.interni_dokument1.
    potpisaoDirektor)

def test_dodijeli_revizora(self):
    revizori = Group.objects.create(name='
        Revizori ')
    revizor = User.objects.create_user(username
        ='test3 ', password='test3 ')
    revizori.user_set.add(revizor)

    self.assertIsNone(self.interni_dokument1.
        revizor)
    response = self.client.put(
        self.base_url + 'dodijeliRevizora/'
        + str(self.interni_dokument1.id),
        {"korisnik_id": revizor.id},
        HTTP_AUTHORIZATION='Bearer ' + self.
            zaposlenik_token,
        content_type='application/json'
    )

    self.assertEqual(response.status_code, 200)
    self.interni_dokument1.refresh_from_db()
    self.assertEqual(self.interni_dokument1.
        revizor, revizor)

    response = self.client.put(
        self.base_url + 'dodijeliRevizora/'
        + str(self.interni_dokument1.id),
        {"korisnik_id": self.direktor.id},
        HTTP_AUTHORIZATION='Bearer ' + self.
            zaposlenik_token,
```

```
        content_type='application/json'
    )
    self.assertEqual(response.status_code, 400)

    response = self.client.put(
        self.base_url + 'dodijeliRevizora/'
        + '0',
        {"korisnik_id": revizor.id},
        HTTP_AUTHORIZATION='Bearer ' + self.
            zaposlenik_token,
        content_type='application/json'
    )
    self.assertEqual(response.status_code, 404)

def test_arhiviraj(self):
    Group.objects.create(name='Računovođe')
    računovođa = User.objects.create_user(
        username='test3', password='test3')
    Group.objects.get(name='Računovođe').
        user_set.add(računovođa)

    računovođa_token = self.client.post(self.
        base_url + 'token/', {'username': 'test3'
        , 'password': 'test3'}).data.get('access
        ')

    pk = self.interni_dokument1.pk
    tekst = self.interni_dokument1.
        tekstDokumenta
    self.assertTrue(InterniDokument.objects.
        filter(pk=pk).exists())
    self.assertFalse(InterniDokumentArhiviran.
        objects.filter(dokumentId=pk).exists())
    response = self.client.put(
```



```
        self.base_url + 'arhiviraj/' + str(
            self.interni_dokument1.id),
        HTTP_AUTHORIZATION='Bearer ' + rač
            unovoda_token,
        content_type='application/json'
    )

    self.assertEqual(response.status_code, 200)
    self.assertFalse(InterniDokument.objects.
        filter(pk=pk).exists())

    documents = InterniDokumentArhiviran.objects
        .filter(dokumentId=pk)
    self.assertTrue(documents.exists())
    self.assertTrue(len(documents) == 1)
    self.assertTrue(documents[0].tekstDokumenta
        == tekst)
```

Rezultat izvođenja u vscodeu:

Found 10 test(s).

Creating test database for alias 'default'...

System check identified no issues (0 silenced).

.....

Ran 10 tests in 11.969s

OK

Destroying test database for alias 'default'...

5.2.2 Ispitivanje sustava

Test 1 (test_login_fail):

Otvorimo URI web stranice. Pokušamo se ulogirati s neispravnom kombinacijom korisničkog imena i zaporce. Osiguramo da se pojavi alert s tekстом "Pogrešno

korisničko ime ili lozinka".

Test 2 (test_login_and_logout):

Otvorimo URI web stranice. Ulogiramo se s ispravnim korisničkim imenom i lozinkom. Izlogiramo se. Osiguramo da smo opet završili na login stranici.

Test 3 (test_change_password):

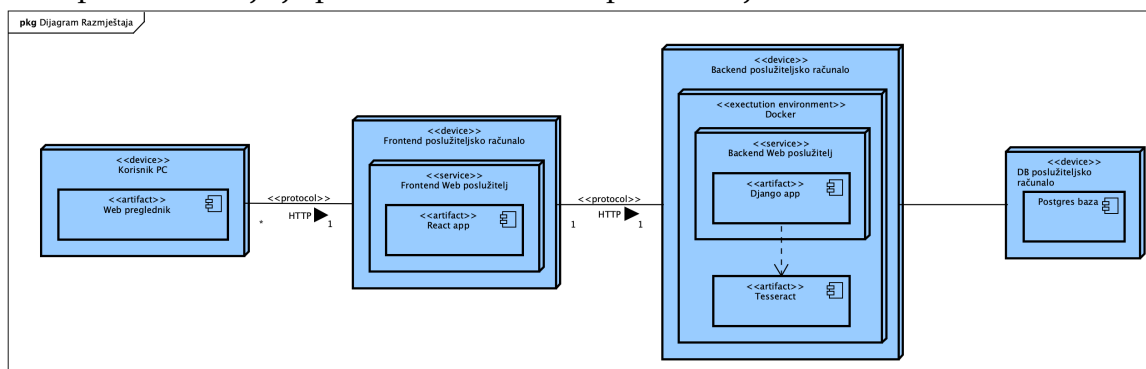
Otvorimo URI web stranice. Ulogiramo se s ispravnim korisničkim imenom i lozinkom. Pokušamo promijeniti lozinku korisnika pri čemu unesemo pogrešnu trenutnu lozinku. Osiguramo da se pojavi alert teksta "Unesite ispravnu trenutnu lozinku" i da se lozinka korisnika nije promijenila. Ponovimo isto ali s ispravnom trenutnom lozinkom i novom lozinkom istom kao i starom. Osiguramo da se pojavi alert teksta "Nova lozinka mora biti različita od stare" i da se lozinka korisnika nije promijenila. Ponovimo isto s ispravnom trenutnom lozinkom i novom lozinkom različitom od stare. Osiguramo da se pojavio alert teksta "Lozinka uspješno promijenjena" i da lozinka korisnika promijenila. Vratimo lozinku na staru.

Test 4 (test_add_new_employee):

Otvorimo URI web stranice. Ulogiramo se s ispravnim direktorovim korisničkim imenom i lozinkom. Otvorimo stranicu za dodavanje novog zaposlenika. Pokušamo dodati novog zaposlenika s istim korisničkim imenom kao neki već postojeći. Osiguramo da se pojavi alert teksta "Greška prilikom dodavanja zaposlenika". Ponovimo isto ali s ispravnim korisničkim imenom. Osiguramo da se pojavi alert teksta "Zaposlenik uspješno dodan". Osiguramo da se novi korisnik pojavio u bazi sa svim potrebnim atributima. Izbrišemo novonastalog korisnika.

5.3 Dijagram razmještaja

Dijagram razmještaja na slici 5.1 prikazuje topologiju sklopovlja i programsku potporu web-aplikacije. Sustav je baziran na arhitekturi "klijent-poslužitelj". Komunikacija između računala korisnika i frontend poslužiteljskog računala, kao i između frontend i backend poslužiteljskog računala, odvija se preko HTTP veze. Korisnici pristupaju web-aplikaciji koristeći web preglednik te im frontend poslužiteljsko računalo, na kojem se nalazi frontend web poslužitelj, daje odgovarajući prikaz. Na backend poslužiteljskom računalu nalazi se Docker u kojemu se nalaze backend web poslužitelj i Tesseract, a Postgres baza nalazi se na poslužiteljskom računalu baze podataka koje je povezano s backend poslužiteljskim računalom.



Slika 5.1: Dijagram razmještaja

5.4 Upute za puštanje u pogon

Baza Podataka

Za puštanje web aplikacije u pogon potrebno je prvo pronaći online uslugu koji pruža uslugu posluživanja baze podataka.

Na tom servisu potrebno je stvoriti novu PostgreSQL bazu te korisnika koji ima pristup bazi. Nije potrebno inicijalizirati tablice u bazi. Potreban je samo URL.

New PostgreSQL [Read the docs](#)

Name
A unique name for your PostgreSQL instance.
DigitalizacijaDB

Database Optional
The PostgreSQL 'dbname'
randomly generated unless specified

User Optional
randomly generated unless specified

Region
The **region** where your PostgreSQL instance runs. Services must be in the same region to communicate privately and you currently have services running in **Frankfurt**.
Oregon (US West)

PostgreSQL Version
15

Datadog API Key Optional
The API key to use for sending metrics to Datadog. Setting this will enable Datadog monitoring.

Slika 5.2: Primjer stvaranja baze podataka na render.com

Backend

Za puštanje backenda u pogon potrebno je na Git-u podesiti datoteku „IzvorniKod/Digitalizacija/Digitalizacija/.env“ tako da u njoj postavite varijablu okoline DATABASE_URL na URL baze podataka stvorene u prijašnjem koraku.

```
1 DATABASE_URL=postgres://baza2_user:a01U0Bw1qVF5bP95H4NF5QpXxyMnvYhI@dpg-cma3v5mn7f5s73ddhvg-a.oregon-postgres.render.com/baza2
2 SECRET_KEY=django-insecure-hbsqj-uuu)vq_ee$o8y+9-)^c9+^%nk_4r%9b1#osaa6l9j(1
```

Slika 5.3: Primjer .env datoteke

Nakon toga potrebno je pronaći uslugu koji omogućava puštanje web servisa u pogon na temelju Dockerfile-a na Git-u. Bitno je da korijenski direktorij iz kojeg pogonimo ovaj servis bude postavljen na „IzvorniKod/Digitalizacija/“.

You are deploying a web service for [vilim-cro/InzenjerskiPristup](#).

The screenshot shows the Render deployment configuration interface. It includes five sections, each with a label, a description, and a form field:

- Name:** A unique name for your web service. The field contains "Inzinjerskipristup".
- Region:** The region where your web service runs. Services must be in the same region to communicate privately and you currently have services running in Frankfurt. The dropdown menu shows "Frankfurt (EU Central)".
- Branch:** The repository branch used for your web service. The dropdown menu shows "main".
- Root Directory:** Defaults to repository root. When you specify a root directory that is different from your repository root, Render runs all your commands in the specified directory and ignores changes outside the directory. The field contains "IzvorniKod/Digitalizacija/".
- Runtime:** The runtime for your web service. The dropdown menu shows "Docker".

Slika 5.4: Primjer puštanja backenda u pogon na render.com

Jednom kad smo postavili sve postavke, web usluga će na temelju Dockerfile-a upogoniti backend.

Ako želimo backend uslugu pokrenuti lokalno, potrebno je instalirati docker. On se može instalirati prema uputama koje se mogu naći ovdje <https://docs.docker.com/get-docker/>. Nakon toga, potrebno je klonirati sadržaj prije navedenog direktorija te u njemu pokrenuti naredbu „docker build -t digitalizacija.“ te pričekati da se izgradi slika. Nakon toga backend možemo pokrenuti naredbom „docker run -p 8000:8000 digitalizacija“.

```
[lovro@avalon Digitalizacija]$ docker run -p8000:8000 digitalizacija
[2024-01-17 21:28:10 +0000] [1] [INFO] Starting gunicorn 21.2.0
[2024-01-17 21:28:10 +0000] [1] [INFO] Listening at: http://0.0.0.0:8000 (1)
[2024-01-17 21:28:10 +0000] [1] [INFO] Using worker: sync
[2024-01-17 21:28:10 +0000] [7] [INFO] Booting worker with pid: 7
```


Slika 5.5: Primjer lokalnog pokretanja backenda

Frontend

Za puštanje frontenda u pogon potrebno je pronaći web uslugu koja može posluživati našu React web aplikaciju.

Potrebno je kao korijenski direktorij postaviti „IzvorniKod/reactapp/“ te osigurati da se pri pokretanju koriste naredbe „npm install“ te „npm run build“. Izvršnu izgradnju možemo jednostavno staviti u direktorij „build“.

You are deploying a static site for [vilim-cro/InzenjerskiPristup](#).

Name A unique name for your static site.	<input type="text" value="Digitalizacija"/>	
Branch The repository branch used for your static site.	<input type="text" value="main"/>	
Root Directory <small>Optional</small> Defaults to repository root. When you specify a root directory that is different from your repository root, Render runs all your commands in the specified directory and ignores changes outside the directory.	<input type="text" value="IzvorniKod/reactapp/"/>	
Build Command This command runs in the root directory of your repository when a new version of your code is pushed, or when you deploy manually. It is typically a script that installs libraries, runs migrations, or compiles resources needed by your app.	<input type="text" value="IzvorniKod/reactapp/"/>	<input type="text" value="\$ npm install;npm run build"/>
Publish directory The relative path of the directory containing built assets to publish. Examples: <code>./</code> , <code>./build</code> , <code>dist</code> and <code>frontend/build</code> .	<input type="text" value="IzvorniKod/reactapp/"/>	<input type="text" value="build"/>
Environment Variables <small>Optional</small> Set environment-specific config and secrets (such as API keys), then read those values from your code. Learn more .	<input type="text" value="REACT_APP_BACKEND_URL"/>	<div><input type="text" value="https://inzenjerskipristup.onrender.com"/></div> <div>+ Add Environment Variable</div>

Slika 5.6: Primjer puštanja frontenda u pogon na render.com

Konačno, prije puštanja frontenda u pogon, moramo mu putem varijable okoline `REACT_APP_BACKEND_URL` prenijeti URL backenda koji smo stvorili u prošlom koraku.

Ako želimo frontend pokrenuti lokalno potrebno je prvo instalirati `node.js` i `npm`. Njih možemo instalirati prema ovim uputama <https://docs.npmjs.com/downloading-and-installing-node-js-and-npm>. Nakon toga potrebno je klonirati sadržaj direktorija „IzvorniKod/reactapp/“ na Git-u. Također je potrebno u ljusci postaviti varijablu okoline te pokrenuti naredbe „`npm install`“ i „`npm run build`“. Frontend sada možemo pokrenuti naredbom „`serve -s build`“.

Nakon toga na URL-u na kojem smo pogonili frontend možemo pristupiti aplikaciji.

6. Zaključak i budući rad

Zadatak naše grupe bio je izrada web aplikacije za ubrzavanje procesa digitalizacije računovodstvenih tvrtki. Naša aplikacija ima jednostavno korisničko sučelje koje omogućuje korisniku da brzo i jednostavno skenira i upravlja dokumentima. Izrada aplikacije trajala je otprilike 14 tjedana i u tom vremenu smo uspjeli ostvariti sve funkcionalnosti koje smo planirali. Prije same izrade aplikacije, cijeli tim se sastao kako bi se dogovorili koje alate i tehnologije ćemo koristiti pri izradi aplikacije te kako bi se rasporedili u podtimove i odredili uloge svakog člana tima. Osim toga napravili smo detaljnu analizu zahtjeva kako bi što bolje razumjeli što je potrebno napraviti i kako bi mogli projekt podijeliti na manje dijelove koji su se mogli razvijati paralelno i kako bi se mogli rasporediti zadaci svakom članu tima. Pri izradi aplikacije susreli smo se s nekoliko tehničkih izazova, ali smo ih uspješno riješili. Budući da je većini tima ovo prvi projekt u ovom obliku, nitko nije bio bolje upoznat s tehnologijama koje smo koristili tako da je prvi izazov bio upoznavanje s tehnologijama. Taj izazov smo riješili tako da smo našli nekoliko pouzdanih resursa koji su nam pomogli da se upoznamo s tehnologijama te smo jednostavno krenuli raditi i učili smo što nam je bilo potrebno. S obzirom da smo prije početka izrade aplikacije napravili detaljnu analizu zahtjeva, sama izrada aplikacije je bila uvelike olakšana jer smo imali jasnu sliku što je potrebno napraviti. Zbog toga smo od početka izrade aplikacije bili vrlo dobro organizirani i nismo imali problema s koordinacijom između članova tima te nam je to uvelike olakšalo izvođenje traženih zahtjeva. Drugi izazov je bio povezivanje frontenda i backenda, odnosno slanje podataka između njih. Taj izazov smo riješili tako da smo se raspitali i pronašli smo nekoliko pouzdanih resursa koji su nam pomogli da se upoznamo s načinom na koji se to radi. Osim toga nije bilo većih izazova pri izradi aplikacije, sve ostalo su bile manje poteškoće na koje smo naišli tokom ostvarivanja pojedinih zahtjeva, a sve takve poteškoće je rješavao član tima koji je bio zadužen za taj dio aplikacije dodatnim istraživanjem i učenjem te su se po potrebi konzultirali s ostalim članovima tima. S obzirom da je ovo prvi projekt u ovom obliku za većinu članova tima, svi smo stekli nova znanja o tehnologijama koje smo koristili te iskustvo rada u timu s kojim smo razvili vještine komunikacije i organizacije koje će nam sigurno

koristiti u budućnosti.

Popis literature

1. Programsko inženjerstvo, FER ZEMRIS, <http://www.fer.hr/predmet/proinz>
2. I. Sommerville, "Software engineering", 8th ed, Addison Wesley, 2007.
3. T.C.Lethbridge, R.Langaniere, "Object-Oriented Software Engineering", 2nd ed. McGraw-Hill, 2005.
4. I. Marsic, Software engineering book“, Department of Electrical and Computer Engineering, Rutgers University, <http://www.ece.rutgers.edu/~marsic/books/SE>
5. The Unified Modeling Language, <https://www.uml-diagrams.org/>
6. Astah Community, <http://astah.net/editions/uml-new>
7. CS50's Web Programming with Python and JavaScript, <https://cs50.harvard.edu/web/2020/weeks/3/>
8. Django, <https://www.djangoproject.com/>
9. React Course - Beginner's Tutorial for React JavaScript Library [2022], <https://www.youtube.com/watch?v=bMknfKXIFA8>
10. Stack overflow, <https://stackoverflow.com/>

Indeks slika i dijagrama

2.1	Slika prijave korisnika u sustav	6
3.1	Dijagram obrasca uporabe, funkcionalnost korisnika	17
3.2	Dijagram obrasca uporabe, funkcionalnost zaposlenika	18
3.3	Dijagram obrasca uporabe, funkcionalnost revizora	18
3.4	Dijagram obrasca uporabe, funkcionalnost računovođe	19
3.5	Dijagram obrasca uporabe, funkcionalnost direktora	20
3.6	Sekvencijski dijagram za UC01	21
3.7	Sekvencijski dijagram za UC07	22
3.8	Sekvencijski dijagram za UC12	23
4.1	Arhitektura sustava	25
4.2	ER dijagram baze podataka	38
4.3	Relacijska shema baze podataka	39
4.4	Dijagram razreda - dio Models	40
4.5	Dijagram razreda - dio Permissions	41
4.6	Dijagram razreda - dio Views	42
4.7	Dijagram stanja	43
4.8	Dijagram aktivnosti	44
4.9	Dijagram komponenti	45
5.1	Dijagram razmještaja	58
5.2	Primjer stvaranja baze podataka na render.com	59
5.3	Primjer .env datoteke	59
5.4	Primjer puštanja backenda u pogon na render.com	60
5.5	Primjer lokalnog pokretanja backenda	60
5.6	Primjer puštanja frontenda u pogon na render.com	61
6.1	Prikaz aktivnosti na repozitoriju	70

Dodatak: Prikaz aktivnosti grupe

Dnevnik sastajanja

1. sastanak

- Datum: 20. listopada 2023.
- Prisustvovali: svi
- Teme sastanka:
 - odabir razvojnih alata i tehnologija
 - generalna podjela poslova

2. sastanak

- Datum: 25. listopada 2023.
- Prisustvovali: svi
- Teme sastanka:
 - detaljnija podjela poslova
 - rasprava oko funkcionalnih i nefunkcionalnih zahtjeva

3. sastanak

- Datum: 26. listopada 2023.
- Prisustvovali: Filip Krilčić, Vilim Branica, Lovro Mužar
- Teme sastanka:
 - organizacija baze podataka
 - dijagrami i implementacija baze podataka

4. sastanak

- Datum: 27. listopada 2023.
- Prisustvovali: Filip Krilčić, Vilim Branica, Zvonimir Pipić, Nika Miličević, Marko Šelendić, Lovro Mužar
- Teme sastanka:
 - završavanje prošlo podijeljenih poslova te provjera prethodno završenog posla
 - podijela novih poslova - OCR demo, dokumentacija (Sekvencijski dijagrami te opis projekta), opis baze i početna stranica, te određi-

vanje roka za završetak istih

5. sastanak

- Datum: 28. listopada 2023.
- Prisustvovali: Filip Krilčić, Vilim Branica i Zvonimir Pipić
- Teme sastanka:
 - podjela poslova na backendu

6. sastanak

- Datum: 4. studenoga 2023.
- Prisustvovali: Filip Krilčić, Vilim Branica, Zvonimir Pipić, Nika Miličević, Marko Šelendić, Tomislav Čupić
- Teme sastanka:
 - analiza commitanih fileova
 - prezentacija izrađenih osnovnih funkcionalnosti backenda

7. sastanak

- Datum: 12. prosinca 2023.
- Prisustvovali: svi
- Teme sastanka:
 - podjela bodova 1. revizije
 - analiza riješenih zadataka
 - određivanje preostalih zadataka

8. sastanak

- Datum: 16. prosinca 2023.
- Prisustvovali: Marko Šelendić, Tomislav Čupić, Zvonimir Pipić
- Teme sastanka:
 - podjela poslova na frontendu

9. sastanak

- Datum: 13. siječnja 2024.
- Prisustvovali: svi
- Teme sastanka:
 - pregled svih napravljenih funkcionalnosti, provjera rada aplikacije i ispravljanje sitnih grešaka

Tablica aktivnosti

Kontinuirano osvježavanje

Napomena: Doprinosi u aktivnostima treba navesti u satima po članovima grupe po aktivnosti.

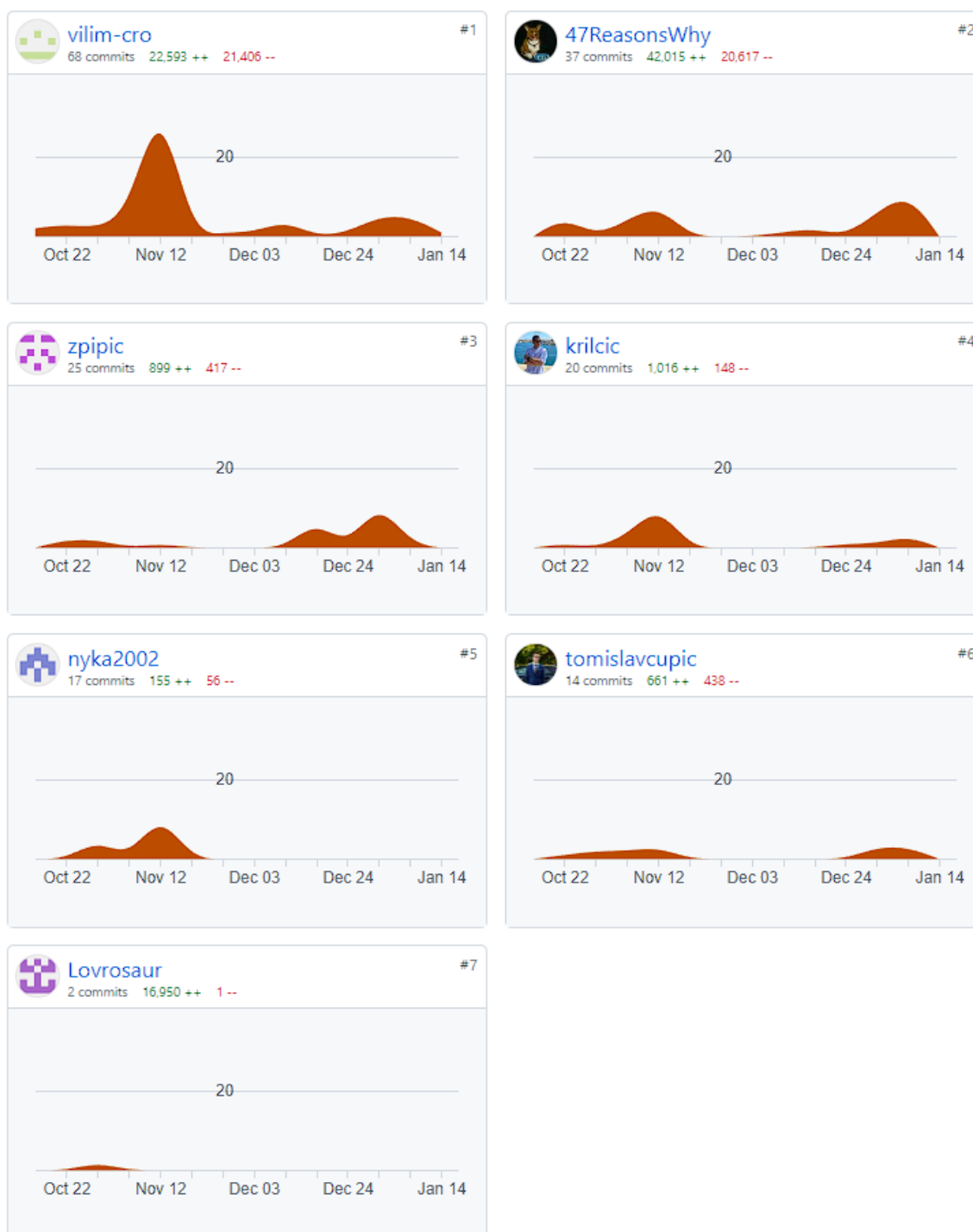
	Vilim Branica	Tomislav Čupić	Filip Krilčić	Nika Miličević	Lovro Mužar	Zvonimir Pipić	Marko Šelendić
Upravljanje projektom	21						
Opis projektnog zadatka		2					
Funkcionalni zahtjevi	1	3			1	1	3
Opis pojedinih obrazaca							3
Dijagram obrazaca				6			
Sekvencijski dijagrami				7			
Opis ostalih zahtjeva		1					
Arhitektura i dizajn sustava		1				4	
Baza podataka			5				
Dijagram razreda				12			
Dijagram stanja							
Dijagram aktivnosti							
Dijagram komponenti							
Korištene tehnologije i alati			1				
Ispitivanje programskog rješenja							
Dijagram razmještaja							
Upute za puštanje u pogon					2		

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

	Vilim Branica	Tomislav Čupić	Filip Krilčić	Nika Miličević	Lovro Mužar	Zvonimir Pipić	Marko Šelendić
Dnevnik sastajanja		2	1				1
Zaključak i budući rad			1				
Popis literature			0.5				
<i>Frontend</i>		25					30
<i>Backend (views i slično)</i>	25		25				
<i>Povezivanje frontenda i backenda</i>	20	5					20
<i>Izrada opisa baze</i>	10		10		4	2	
<i>Implementacije baze</i>	8		7				
<i>Povezivanje na vanjski API za upload slike</i>			5			10	
<i>Implementacija OCR-a</i>					55		
<i>Deployment</i>	10				45		10

Dijagrami pregleda promjena



Slika 6.1: Prikaz aktivnosti na repozitoriju