Programsko inženjerstvo

Ak. god. 2023./2024.

Digitalizacija

Dokumentacija, Rev. *2*

Grupa: *Jura Hostić i Film*

Voditelj: *Tvrtko Puškarić*

Datum predaje: *19. 1. 2024.*

Nastavnik: *Igor Stančin*

Sadržaj

1. Dnevnik promjena dokumentacije

Rev.	Opis promjene/dodatka	Autori	Datum
0.1	Stvoren prazan predložak.	*	05.11.2023.
0.2	Podjela posla, započeta poglavlja 2, 3 i 4.	*	08.11.2023.
0.3	Dodano poglavlje 3.2, sekvencijski dijagrami	Katarina Pešić	16.11.2023.
0.4	Dodane osnovne informacije, zapisnik sastanka	Luka Bradarić Lisić, Tvrtko Puškarić	17.11.2023.
0.5	Dodano poglavlje 2	Tvrtko Puškarić	17.11.2023.
0.6	Dodano poglavlje 4	Andrea Milanović, Jura Hostić, Karlo Grgičin	17.11.2023.

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Rev.	Opis promjene/dodatka	Autori	Datum
0.7	Dodano poglavlje 3.1	Luka Bradarić Lisić, Martin Subotić, Katarina Pešić	17.11.2023.
1.0	Prva revizija	*	17.11.2023.
1.1	Dodan dijagram aktivnosti	Martin Subotić, Andrea Milanović	16.1.2024.
1.2	Prilagodit opis baze podataka stvarnom stanju	Jura Hostić	18.1.2024.
1.3	Dodane korištene tehnologije i alati	Jura Hostić	18.1.2024.
1.4	Prilagođen dijagram razreda stvarnom stanju	Karlo Grgičin	18.1.2024.
1.5	Dodano ispitivanje komponenti	Andrea Milanović	18.1.2024.
1.6	Dodan dijagram komponenti	Karlo Grgičin, Luka Bradarić Lisić, Katarina Pešić	19.1.2024.

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Rev.	Opis promjene/dodatka	Autori	Datum
1.7	Dodan dijagram ramještaja	Jura Hostić	19.1.2024.
1.8	Dodan dijagram stanja	Luka Bradarić Lisić	19.1.2024.
1.9	Popravljanje dijagrama obrazaca uporabe	Martin Subotić	19.1.2024.
1.10	Dodane upute za puštanje u pogon	Jura Hostić, Tvrtko Puškarić	19.1.2024.
1.11	Dodan zaključak	Tvrtko Puškarić	19.1.2024.
1.12	Dodano ispitivanje sustava	Tvrtko Puškarić	19.1.2024.
1.13	Administrativne promjene	Jura Hostić, Tvrtko Puškarić	19.1.2024.
2.0	Druga revizija	*	19.1.2024.

2. Opis projektnog zadatka

Cilj ovog projekta je pružiti specifičnom klijentu, u ovom slučaju računovodstvenom uredu, prilagođeno i primjenjivo rješenje za njihove potrebe. Rješenje se ostvaruje u obliku mobilne aplikacije koja mora biti prilagođena i konfigurirana prema njihovim specifičnim potrebama i strukturi posla što osigurava da klijent dobije optimalno rješenje za digitalizaciju i održavanje postojeće arhive dokumenata, usklađeno s njihovom poslovnom strukturom. To podrazumijeva sagledavanje potreba svih zaposlenika, proučavanje obrazaca upotrebe i pronalaska najboljeg načina za tehnološko ostvarenje. Potrebno je dostaviti gotovo, primjenjivo rješenje koje istovremeno zadovoljava zadane uvjete i obrasce upotrebe te primijeniti valjana inženjerska načela kako bi se osigurala kvaliteta proizvoda, kao i mogućnost lake održivosti te dugovremene upotrebe.

2.1 Skup ciljanih korisnika i upotrebe

Skup ciljanih korisnika čine svi zaposlenici firme, uz dodatnog vanjskog administratora koji početno postavlja sustav. Rješenje mora biti prilagođeno zahtjevima svih vrsta korisnika, odnosno prilagođeno potrebama za rad svih zaposlenika unutar ureda. U daljnjem tekstu će zaposlenici biti vezani s istoimenim ulogama. Važno je napomenuti kako je za sve namijene potrebno ostvariti funkcionalno i jednostavno sučelje.

Kako je računovodstveni ured zatvoreni sustav, nepovezan s vanjskim čimbenicima, generalno nije potrebno imati nadstrukturu koja će upravljati sustavom već ured može upravljati samim sobom, za što je zadužen direktor. Vanjski administrator postoji jedino u vidu početnog postavljanja sustava gdje on stvara direktora i po potrebi druge korisnike. Vanjski administrator također može uređivati korisnike po potrebi ako dođe do nepredviđenih grešaka u radu, no zamišljen je samo kao pomoćni alat koji odstranjuje potrebu za direktnim pristupom bazi podataka. Prvenstveno bi administrator trebao biti vanjski akter, na primjer član razvojnog tima, no ulogu je umjesto toga također moguće dati zaposleniku ureda ovisno o klijentovim željama.

Sukladno zahtjevima zaposlenici ureda se dijele na općenite zaposlenike, revizore, računovođe i direktore. Svi zaposlenici trebaju imati mogućnost digitalizacije dokumenata, kao i pregleda vlastitih, prethodno predanih. Također, valjalo bi omogućiti pregled statusa pojedinog predanog dokumenta u tijeku procesa potvrđivanja. Nadalje, općeniti zaposlenik je onaj koji nema dodatnih ovlaštenja osim digitalizacije pojedinog dokumenta.

Nakon predaje dokumenta potrebno ga je revidirati, čime se bave revizori. Revizorima se dokumenti trebaju dodijeliti ovisno o trenutnom statusu i zaposlenosti, nakon čega je odabranog revizora potrebno obavijestiti. Revizor pojedini dokument može odbiti ili potvrditi, a potvrđene prosljeđuje dalje računovođi. Sustav bi trebao omogućiti lako prosljeđivanje kako bi se proces ubrzao, stoga se mogu ponuditi opcije za automatsko ili ručno prosljeđivanje. U slučaju automatskog prosljeđivanja bira se valjani računovođa s trenutno najmanje posla. Dodatno, valjalo bi revizoru omogućiti pregled prethodno revidiranih dokumenata i njihovog daljnjeg statusa.

Računovođe se dijele ovisno o vrstama dokumenata koje mogu arhivirati. Prije arhiviranja određenog dokumenta računovođa također ima mogućnost prosljeđivanja

dokumenta direktoru na potpis, na koji se treba čekati u tom slučaju. Računovođu valja obavijestiti o dokumentima spremnim za arhiviranje, bili oni novi ili novopotpisani. Kao i prethodnim zaposlenicima valjalo bi računovođi omogućiti pregled dokumenata na kojima je radio, konkretnije prethodno arhiviranih.

Direktor je glavni upravitelj ureda i kao takav mora imati pristup stvaranju i brisanju korisnika, kao i uređivanju podataka i dopuštenja istih. Stoga mu je potrebno omogućiti pregled svih korisnika, njihovih statistika i dokumenata na kojima su radili. Sam direktor također sudjeluje u tijeku digitalizacije dokumenata ako se od njega traži potpis, o čemu ga se mora obavijestiti. Konačno, direktoru se mora omogućiti objavljivanje dokumenata na društvenim mrežama.

Rješenje osim što mora zadovoljavati početne zahtjeve, također mora biti lako prilagodljivo novim zahtjevima, zaposlenicima ili ulogama, zbog čega ga je važno pravilno koncipirati.

2.2 Osnovni zahtjevi

Osnovni zahtjevi koje planirana aplikacija mora ostvariti odnose se na procese digitalizacije i upravljanja prenesenim dokumentima te upravljanja zaposlenicima u sklopu aplikacije. Svaki od navedenih procesa ima svoje specifične zahtjeve, a oni koji nisu detaljnije obrađeni pri pregledu skupa korisnika opisani su u nastavku.

Proces digitalizacije sastoji se od unosa dokumenata u aplikaciju. Mora biti omogućen unos do 50 dokumenata odjednom. Za svaki dokument mora se provesti OCR (optical character recognition), što čini osnovnu funkcionalnost samog procesa. Prije same digitalizacije za svaki predani dokument valja provjeriti zadovoljava li sve predodređene uvjete, što se odnosi na kut slikanja i dobiveni oblik sadržaja. Uvjete valja provjeriti kako bi se ispravno mogao provesti OCR proces. Ako je digitalizacija pojedinog dokumenta uspješna korisniku se prikazuje sažetak dokumenta, nakon čega mora označiti dokument pravilno ili nepravilno skeniranim, ovisno o čemu se prosljeđuje dalje u sklopu tijeka upravljanja prenesenim dokumentima.

Važno je napomenuti kako se skup dokumenata koje je moguće skenirati sastoji od tri tipa dokumenata, odnosno računa, ponuda i internih dokumenta. Svaki tip zadovoljava određeni format po čemu se mogu razvrstati u procesu upravljanja. Računi će u svom tekstu nakon OCR-a imati oznaku računa koja je veliko slovo R te šest znamenaka, oznaka ponude imat će veliko slovo P i devet znamenaka, a oznaka internog dokumenta „INT“ i četiri znamenke. Računi osim oznake sadrže ime klijenta, artikle s cijenama i ukupnu cijenu. Ponude su kao računi, ali ne sadrže ime klijenta. Interni dokumenti sadrže samo nestrukturirani tekst. Očekuje se kako će u slučaju ispravnog predanog dokumenta, valjanog formata, aplikacija biti u stanju prepoznati tip i pravilno ga razvrstati.

Očekuje se kako će pravilno skenirani dokumenti biti spremljeni u bazi podataka te biti dostupni za pregled svim ovlaštenim zaposlenicima, ovisno o stadiju u kojem se nalaze.

2.3 Korisnost rješenja

Planirano rješenje korisno je u okviru procesa digitalizacije i održavanja postojećih arhiva dokumenata zbog mogućnosti za optimizaciju rada, što se odnosi na raspodjelu posla i točno definiranim koracima koji se provode. Rješenje će omogućiti brzu i efikasnu digitalizaciju postojećih papirnatih dokumenata s pomoću algoritama za prepoznavanje teksta i automatsku klasifikaciju dokumenata, što znatno ubrzava proces pretvaranja fizičkih dokumenata u digitalni format. To čini arhiviranje i pretraživanje dokumenata mnogo jednostavnijim, smanjuje potrebu za skladištenjem fizičkih dokumenata i minimizira rizik od gubitka ili oštećenja dokumenata. Aplikacija također omogućuje digitalno-analogno upravljanje dokumentima. Zaposlenici mogu lako pristupiti digitalnim kopijama dokumenata putem sučelja, pregledavati ih, uređivati i označavati kako bi zadovoljili svoje specifične potrebe. Ovo omogućuje rad s dokumentima na način koji je prilagođen njihovim zahtjevima, bez potrebe za povratkom fizičkih dokumenata iz arhive. Najbitnije je što se olakšava raspodjela posla za održavanje arhiva. Sustav omogućuje postavljanje različitih pristupnih razina i ovlaštenja za korisnike, omogućavajući preciznu kontrolu nad tim tko može pristupiti, pregledavati ili uređivati određene dokumente. To olakšava timski rad, omogućava bolju suradnju i smanjuje rizik od neovlaštenog pristupa. Na kraju rješenje pomaže klijentu da iskoristi prednosti digitalizacije, učinkovitije upravlja svojim arhivom dokumenata te ostvaruju uštede u vremenu, prostoru i resursima.

2.4 Slična postojeća rješenja

Uz zahtjeve i želje klijenta nužno je proučiti i druga dostupna rješenja koja se bave sličnom problematikom kako bi se upotpunila lista zahtjeva i poboljšao plan rješenja. Načini na koje postojeća rješenja pristupaju određenim odrednicama problema mogu pomoći pri razradi vlastitog rješenja, a pogotovo ako se klijent već susretao s nekim od alata ili već koristi jedan od istih. U nastavku su navedena najpoznatija i najraširenija postojeća rješenja koja se mogu pronaći u upotrebi, kao i kratki opis svakog.

Adobe Acrobat je poznata platforma za uređivanje i upravljanje dokumentima. S obzirom na proširenja, omogućava napredne mogućnosti za obradu dokumenata, uključujući OCR za pretvaranje skeniranih dokumenata u pretražive tekstualne datoteke. Dodatno, proširenja kao što su Adobe Sign omogućuju digitalno potpisivanje dokumenata, pojednostavljujući poslovne procese.

ABBYY FineReader i FlexiCapture su rješenja specijalizirana za OCR i automatizaciju procesa. FineReader se koristi za OCR, konverziju i prepoznavanje teksta na visokoj razini preciznosti, dok FlexiCapture omogućava automatizaciju prikupljanja podataka iz različitih izvora i vrsta dokumenata, uključujući obrasce i fakturiranje.

Amazon Textract je Amazonova usluga za automatsko prepoznavanje teksta i strukturu dokumenata, a može se integrirati s drugim AWS uslugama kao što su S3, Lambda i Rekognition. Textract kombinira OCR s naprednim algoritmima strojnog učenja kako bi izvlačio podatke iz dokumenata, olakšavajući njihovu analizu i upotrebu.

Google Workspace (ranije poznat kao G Suite) omogućava korisnicima pohranu, zajedničko uređivanje i dijeljenje dokumenata u oblaku. Dodatno, Google Document AI koristi strojno učenje za analizu i ekstrakciju informacija iz dokumenata, čime se olakšava pretraživanje i organizacija dokumenata unutar platforme.

Microsoft SharePoint je platforma za upravljanje sadržajem i suradnju koja omogućava organizacijama da pohrane, dijele i upravljaju svojim dokumentima. Ima ugrađene mogućnosti za OCR, omogućujući pretraživanje i indeksiranje sadržaja dokumenata, čime olakšava njihovo upravljanje i održavanje unutar SharePoint okoline.

Glavno što je zajedničko navedenim rješenjima jest mogućnost baratanja dokumentima, tijek digitalizacije dokumenata kao i mogućnosti za OCR. Prednost vlas-

titog rješenja nalazi se u većoj prilagođenosti specifičnim zahtjevima, što dodatno povlači lakšu integraciju u postojeći tijek rada čime nestaje potreba za proučavanjem mnoštva dostupnih alata i prilagodbe istih klijentovim potrebama.

2.5 Mogućnosti dodatnih proširenja

Mogućnost slikanja dokumenata direktno putem mobilne aplikacije može biti vrlo korisna značajka koja se može ostvariti zbog izabrane platforme. Predstavljala bi velika prednost nad klasičnim, računalnim programima pružajući korisnicima brz i praktičan način za unos i obradu dokumenata. Uporaba kamere mobilnog uređaja omogućuje korisnicima lak unos slike dokumenta i obrade putem OCR-a. Ovo bi bilo posebno korisno zaposlenicima koji su često u pokretu i žele brzo dokumentirati račune, ponude ili interne dokumente.

Povezano s time, razvoj paralelne web inačice aplikacije proširio bi korisničko iskustvo omogućujući pregled dokumenata na računalima. Web inačica omogućila bi korisnicima, posebice računovođama i direktorima, pristup dokumentima putem web preglednika. Ovo je korisno za dublje analize, pregledavanje većih količina dokumenata ili jednostavno pristupanje informacijama na radnom računalu.

Teoretska web inačica zadržala bi sve funkcionalnosti mobilne aplikacije, uključujući pregled sažetaka dokumenata, praćenje povijesti, označavanje ispravno ili krivo skeniranih dokumenata te objavljivanje određenih dokumenata na društvenim mrežama. Ovo proširenje platforme omogućilo bi korisnicima fleksibilnost u radu zbog olakšanog pristupa informacijama iz različitih okruženja, bilo putem mobilnog uređaja ili računala.

2.6 Daljnja prilagodba, razvoj i održavanje

Velika prednost upotrebe cross-platform tehnologije za razvoj mobilne aplikacije je što olakšava buduću prilagodbu i održavanje. Razvojni okvir Flutter omogućuje ponovnu upotrebu koda između operativnih sustava za mobilne uređaje što pojednostavljuje popravak grešaka, poboljšanje postojećih funkcionalnosti, kao i implementaciju novih, čime se smanjuje vrijeme i trošak održavanja sustava. Razvoj inačice za IOS uređaje u budućnosti bi mogao imati veliki prioritet kako bi svi zaposlenici mogli koristiti aplikaciju na vlastitim uređajima.

3. Specifikacija programske potpore

3.1 Funkcionalni zahtjevi

Dionici:

1. Vlasnik (naručitelj)
2. Članovi organizacije
   (a) Zaposlenik
   (b) Revizor
   (c) Računovođa
   (d) Direktor
3. Administrator
4. Razvojni tim

Aktori i njihovi funkcionalni zahtjevi:

1. Neregistrirani/neprijavljeni korisnik (inicijator) može:
   (a) se prijaviti u sustav pomoću kredencijala (e-mail adresa i lozinka) koje je dobio od superriornije osobe
2. Zaposlenik (inicijator) može:
   (a) prijaviti se u sustav
   (b) pregledati svoje osobne podatke (ime, prezime, nadređeni revizor)
   (c) skenirati dokumente koje se spremaju u njegovu internu bazu
   (d) potvrditi dokumente koje je prethodno skenirao
   (e) primiti obavijesti od revizora i nadređenih zaposlenika tvrtke
   (f) poslati revizirane dokumente od njegove strane na dodatan pregled
   (g) vidjeti svoju internu bazu prethodno skeniranih dokumenata
3. Revizor (inicijator) može:
   (a) prijaviti se u sustav
   (b) pregledati svoje osobne podatke (ime, prezime, popis podređenih zaposlenika)

(c) dobiti obavijesti o skeniranim dokumentima od strane podređenog zaposlenika
(d) skenirati dokumente koje se spremaju u njegovu internu bazu
(e) revizirati primljene dokumente
(f) proslijediti određenom računovođi dokument skeniran i potvrđen od strane podređenog zaposlenika

4. Računovođa (inicijator) može:
   (a) prijaviti se u sustav
   (b) pregledati svoje osobne podatke (ime, prezime, vrstu dokumenta za koju je zaslužan)
   (c) pregledati dokumente spremne za arhiviranje
   (d) arhivirati dokumente
   (e) pregledati povijest dokumenata koje je arhivirao u svojoj internoj bazi
   (f) prosljediti dokumente direktoru na potpis prije arhiviranja
5. Direktor (inicijator) može:
   (a) prijaviti se u sustav
   (b) pregledati svoje osobne podatke (ime, prezime, ime tvrtke za koju je zaslužan)
   (c) primati obavijesti o zatraženim potpisima o računovođi
   (d) pregledati popis svih korisnika aplikacije
   (e) pregledati statistike zaposlenika (broj skeniranih dokumenata, prosjek skeniranih dokumenata svih zaposlenika te ostale statistike)
   (f) promjeniti uloge i ostale podatke o svim ostalim korisnicima unutar tvrtke
   (g) brisati račune otpuštenih zaposlenika tvrtke te ih također stvarati
   (h) pregledati sve dokumente i njihovu povijest(tko ih je skenirao, revizirao i koji računovođa je bio zaslužan za njihovo arhiviranje)
   (i) objaviti određeni dokument na društvenim mrežama
6. Administrator (inicijator) može:
   (a) prijaviti se u sustav
   (b) dodavati različite tvrtke i njihove direktore
   (c) pregledati osobne podatke svih korisnika aplikacije

7. Baza podataka (sudionik) može:
   (a) pohraniti podatke o svim korisnicima aplikacije
   (b) pohraniti sve podatke o tvrtkama te arhivitra sve dokumente određene tvrtke

3.1.1 Obrasci uporabe

Opis obrazaca uporabe

UC1 -Registracija u sustav

- **Glavni sudionik:** Direktor, Administrator
- **Cilj:** Stvoriti korisnički račun za pristup sustavu
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
  1. Aktor odabire opciju za registraciju
  2. Aktor unosi osobne podatke za novog korisnika
  3. Šalje se mail novododanom korisniku s njegovim podacima za prijavu
- **Opis mogućih odstupanja:**

  2.a Korisnik je već registriran u sustav
  1. Sustav daje audio-vizualnu obavjest aktoru da je korisnik već registriran

  2.b Neispravna e-mail adresa
  1. Sustav daje auditornu i vizualnu obavjest aktoru da je e-mail adresa neispravna
  2. Aktor upisuje ispravnu e-mail adresu

UC2 -Prijava u sustav

- **Glavni sudionik:** članovi organizacije, Administrator
- **Cilj:** Prijava za pristup funkcionalnost aplikacije
- **Sudionici:** Baza podataka
- **Preduvjet:** Aktor je registriran
- **Opis osnovnog tijeka:**
  1. Aktor unosi svoje osobne podatke
  2. Sustav provjerava jesu li podnešeni podaci ispravni
  3. Aktor biva prijavljen u aplikaciju
- **Opis mogućih odstupanja:**

  2.a Unešeni osobni podaci ne odgovaraju niti jednom registriranom korisniku
  1. Sustav daje audio-vizualnu obavjest aktoru da su osobni podaci neispravni

2. Sustav daje priliku aktoru za ponovnu prijavu

UC3 -Brisanje korisnika iz sustava

- **Glavni sudionik:** Direktor, Administrator
- **Cilj:** Brisanje korisničkog računa
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik računa je registriran u sustav
- **Opis osnovnog tijeka:**
  1. Aktor otvara administrativnu ploču
  2. Aktor odabire popis svih registriranih klijenata
  3. Aktor odabire korisnika kojeg želi izbrisati
  4. Aktor briše korisnika
  5. Baza podataka briše račun korisnika
  6. Aktora se vraća na popis svih registriranih klijenata

UC4 -Pregled korisničkih podataka

- **Glavni sudionik:** Direktor, Administrator
- **Cilj:** Dobiti uvid u osobne podatke klijenta
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen
- **Opis osnovnog tijeka:**
  1. Aktor otvara listu svih korisnika
  2. Baza podataka dohvaća listu svih registriranih korisnika
  3. Aktor odabire određenog korisnika na danom popisu
  4. Baza podataka dohvaća osobne podatke odabranog korisnika

UC5 -Promjena podataka zaposlenika

- **Glavni sudionik:** Direktor
- **Cilj:** Napraviti promjena nad osobnim podatcima zaposlenika tvrtke
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen u sustav
- **Opis osnovnog tijeka:**
  1. Direktor je odabrao opciju za prikaz svih zaposlenika tvrtke
  2. Baza dohvaća popis svih zaposlenika tvrtke
  3. Direktor odabire određenog zaposlenika
  4. Baza podataka dohvaća odabranog zaposlenika

5. Direktoru se prikazuju podatci zaposlenika
6. Direktor mijenja njegove podatke
7. Direktor sprema podatke pritiskom na tipku
8. Baza podataka sprema nove promijene nad zaposlenikom

- **Opis mogućih odstupanja:**

7.a Direktor nije odabrao opciju spremanja podataka prije izlaska iz sučelja
1. Sustav daje audio-vizualnu obavjest direktoru da se promijenjeni podatci neće spremiti
2. Direktor odabire želi li odbaciti/spremiti novo nastale promijene

7.b Direktor je unio nove neispravne podatke
1. Sustav daje auditornu i vizualnu obavjest direktoru da su određeni podatci neispravni
2. Direktor odabire želi li promijeniti ili odbaciiti neispravne podatke

UC6 - Primanje obavijesti

- **Glavni sudionik:** Zaposlenik, Revizor, Računovođa, Direktor
- **Cilj:** Primiti obavijesti od aplikacije
- **Sudionici:** Aplikacija
- **Preduvjet:** Prijavljen u sustav
- **Opis osnovnog tijeka:**
  1. Aplikacija šalje obavijesti korisniku o relevantnim događajima ili zadacima
  2. Korisnik prima obavijesti putem aplikacije ili e-pošte

UC7 - Pregledavanje statistike zaposlenika tvrtke

- **Glavni sudionik:** Direktor
- **Cilj:** Pregledati statističke podatke o zaposlenicima tvrtke
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen u sustav
- **Opis osnovnog tijeka:**
  1. Direktor otvara statističku sekciju aplikacije
  2. Sustav dohvaća statističke podatke o zaposlenicima
  3. Direktor pregledava informacije o radu i aktivnostima zaposlenika

UC8 - Pregled svih dokumenata

- **Glavni sudionik:** Direktor

- **Cilj:** Pregledati sve dokumente
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen u sustav
- **Opis osnovnog tijeka:**
    1. Direktor odabire pregled popisa svih dokumenata
    2. Sustav prikazuje popis svih dokumenata
    3. Direktor pregledava dokumente i njihove detalje

UC9 - Pregled skeniranih dokumenata

- **Glavni sudionik:** Zaposlenik, Revizor, Računovođa, Direktor
- **Cilj:** Pregledati skenirane dokumente
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen u sustav
- **Opis osnovnog tijeka:**
    1. Aktor odabire pregled popisa svih dokumenata koje je skenirao
    2. Sustav prikazuje popis svih dokumenata skeniranih od strane aktora
    3. Aktor pregledava dokumente i njihove detalje

UC10 - Pregled dokumenata spremnih za reviziranje

- **Glavni sudionik:** Revizor
- **Cilj:** Pregledati dokumente spremne za reviziranje
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen u sustav
- **Opis osnovnog tijeka:**
    1. Revizor odabire pregled popisa svih dokumenata koje je skenirao
    2. Sustav prikazuje popis svih dokumenata spremnih za reviziranje
    3. Revizor pregledava dokumente spremne za reviziranje i njihove detalje

UC11 - Pregled dokumenata spremnih za arhiviranje

- **Glavni sudionik:** Računovođa
- **Cilj:** Pregledati dokumente spremne za arhiviranje
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen u sustav
- **Opis osnovnog tijeka:**
    1. Računovođa odabire pregled popisa svih dokumenata spremnih za arhiviranje

2. Sustav prikazuje popis svih dokumenata spremnih za arhiviranje
3. Računovođa pregledava dokumente spremne za arhiviranje i njihove detalje

UC12 - Pregled dokumenata koji čekaju potpis

- **Glavni sudionik:** Računovođa, Direktor
- **Cilj:** Pregledati dokumente za koje je potreban potpis
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen u sustav
- **Opis osnovnog tijeka:**
  1. Računovođa odabire pregled popisa svih dokumenata za koje je potreban potpis
  2. Sustav prikazuje popis svih dokumenata za koje je potreban potpis
  3. Računovođa pregledava dokumente spremne za koje je potreban potpis i njihove detalje

UC13 - Pregled svih arhiviranih dokumenata određenog tipa

- **Glavni sudionik:** Računovođa
- **Cilj:** Pregledati sve arhivirane dokumente za čiji je tip zaslužan
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen u sustav
- **Opis osnovnog tijeka:**
  1. Računovođa odabire pregled popisa svih arhiviranih dokumenata za čiji je tip zaslužan
  2. Sustav prikazuje popis svih takvih dokumenata
  3. Računovođa pregledava arhivirane dokumente

UC14 - Pregled svih reziviranih dokumenata određenog tipa

- **Glavni sudionik:** Računovođa
- **Cilj:** Pregledati sve revizirane dokumente za čiji je tip zaslužan
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen u sustav
- **Opis osnovnog tijeka:**
  1. Računovođa odabire pregled popisa svih reviziranih dokumenata za čiji je tip zaslužan
  2. Sustav prikazuje popis svih takvih dokumenata

3. Računovođa pregledava revizirane dokumente

UC15 - Objava dokumenta na društvene mreže

- **Glavni sudionik:** Direktor
- **Cilj:** Objaviti odabrani dokument na društvenim mrežama
- **Sudionici:** Društvene mreže, Baza podataka
- **Preduvjet:** Prijavljen u sustav
- **Opis osnovnog tijeka:**
  1. Direktor odabire opciju za prikaz popisa svih dokumenata tvrtke
  2. Direktor odabire dokument koji želi podijeliti na društvenim mrežama
  3. Sustav omogućuje izbor društvenih mreža za objavu
  4. Direktor potvrđuje i dijeli dokument na odabranim društvenim mrežama

UC16 - Pregledavanje dokumenata s liste

- **Glavni sudionik:** Zaposlenik, Revizor, Računovođa, Direktor
- **Cilj:** Pregledati dostupne dokumente
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen u sustav
- **Opis osnovnog tijeka:**
  1. Aktor odabire neku od opcija za prikaz liste dokumenata
  2. Sustav prikazuje popis dostupnih dokumenata
  3. Aktor odabire dokument za pregled
  4. Sustav prikazuje detalje odabranog dokumenta

UC17 - Arhiviranje dokumenata

- **Glavni sudionik:** Računovođa
- **Cilj:** Arhivirati odabrane dokumente
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen u sustav
- **Opis osnovnog tijeka:**
  1. Računovođa otvara listu potpisanih dokumenata ili listu dokumenata spremnih za arhiviranje
  2. Računovođa odabire dokumente koje želi arhivirati
  3. Sustav omogućuje odabir mjesta za arhiviranje
  4. Računovođa potvrđuje i arhivira dokumente

UC18 - Potpisivanje dokumenata

- **Glavni sudionik:** Direktor
- **Cilj:** Potpisati odabrane dokumente prije njihovog arhiviranja
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen u sustav
- **Opis osnovnog tijeka:**
    1. Direktor odabire opciju za prikaz dokumenata koji čekaju potpis
    2. Direktor odabire dokumente koje želi potpisati
    3. Sustav omogućuje pregled odabranih dokumenata
    4. Nadležni direktor potpisuje dokumente

UC19 - Slanje dokumenta na potpis

- **Glavni sudionik:** Računovođa
- **Cilj:** Poslati odabrani dokument na potpis nadležnom direktoru
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen u sustav
- **Opis osnovnog tijeka:**
    1. Računovođa odabire opciju za prikaz dokumenata koji čekaju arhiviranje
    2. Računovođa odabire dokument koji želi poslati na potpis
    3. Dokument se šalje na potpis nadležnom direktoru
    4. Nadležni direktor prima obavijest o dokumentu za potpis

UC20 - Učitavanje slika dokumenata

- **Glavni sudionik:** Zaposlenik, Revizor, Računovođa, Direktor
- **Cilj:** Učitavanje slika dokumenata za digitalizaciju
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen u sustav
- **Opis osnovnog tijeka:**
    1. Aktor odabire opciju za učitavanje slika dokumenata
    2. Aktor odabire do 50 slika za učitavanje
    3. Sustav šalje slike na OCR
- **Opis mogućih odstupanja:**

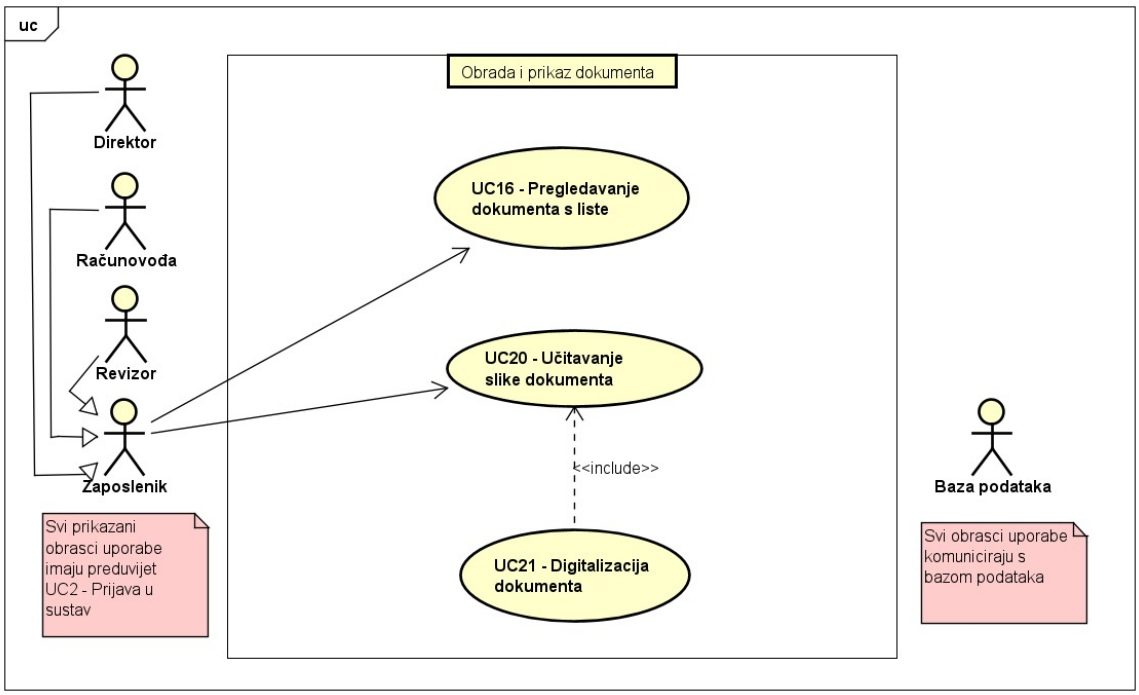
    2.a Aktor odabire više od 50 slika za učitavanje
        1. Sustav daje audio-vizualnu obavijest korisniku da je premašen maksimalni broj slika
        2. Aktor odabire do 50 slika za učitavanje

UC21 - Digitalizacija dokumenta

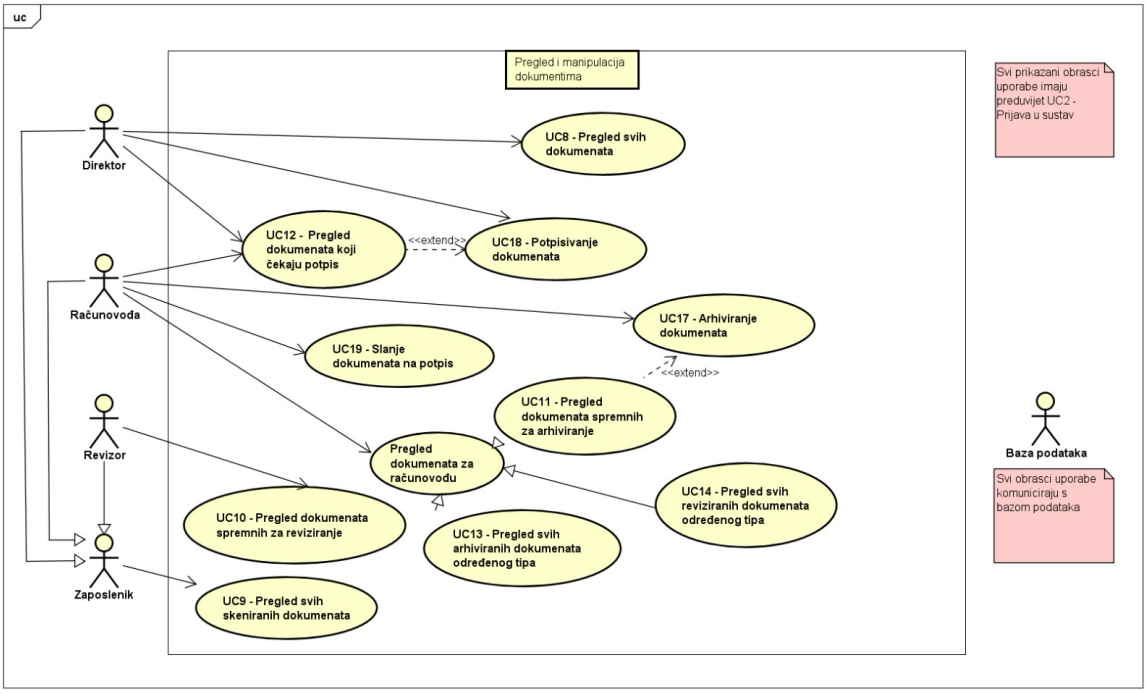
- **Glavni sudionik:** Sustav
- **Cilj:** Detektirati dokument sa učitane slike te ga pretvoriti u digitalni oblik
- **Sudionici:** Baza podataka
- **Preduvjet:** Dokument je uslikan
- **Opis osnovnog tijeka:**
  1. Sustav učitava sliku primljenu na danju obradu od strane aktora
  2. Sustav pregledava danu sliku te utvrđuje nalazi li se dokument na slici
  3. Sustav primjenjuje OCR na detektirani dokument na slici
  4. Sustav sprema novonastali dokument u bazu podataka
- **Opis mogućih odstupanja:**

  2.a Sustav nije pronašao dokument na slici
  1. Sustav daje audio-vizualnu obavijest aktoru da učitana slika ne sadrži dokument
  2. Sustav daje opciju aktoru da priloži drugu sliku ili da odustane od daljnje obrade

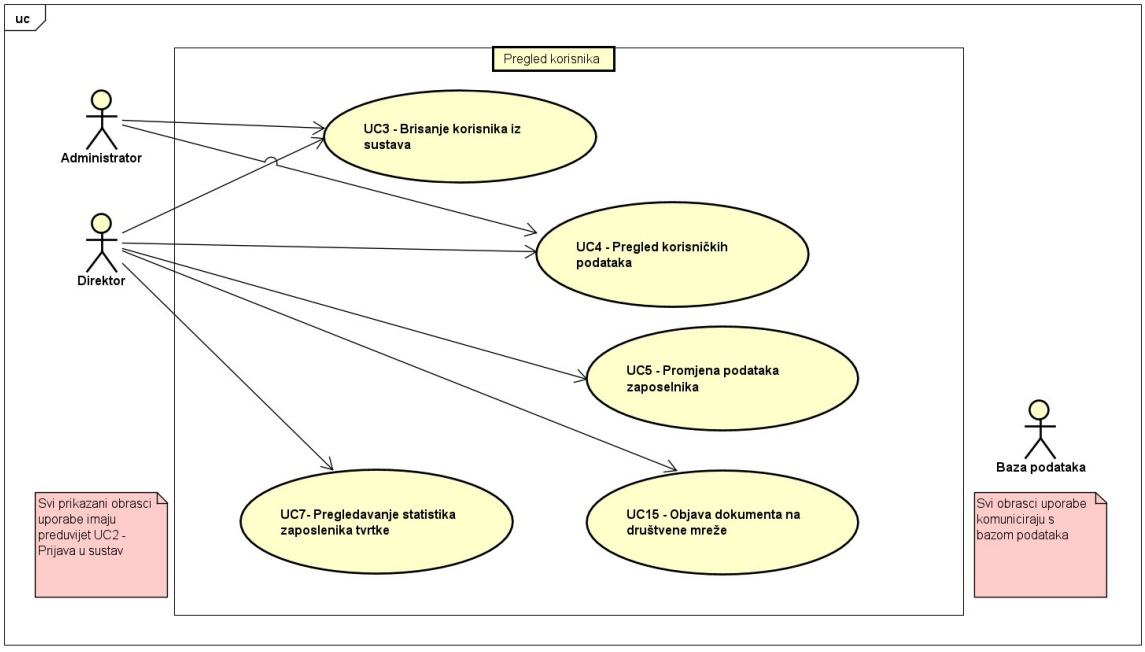
Dijagrami obrazaca uporabe



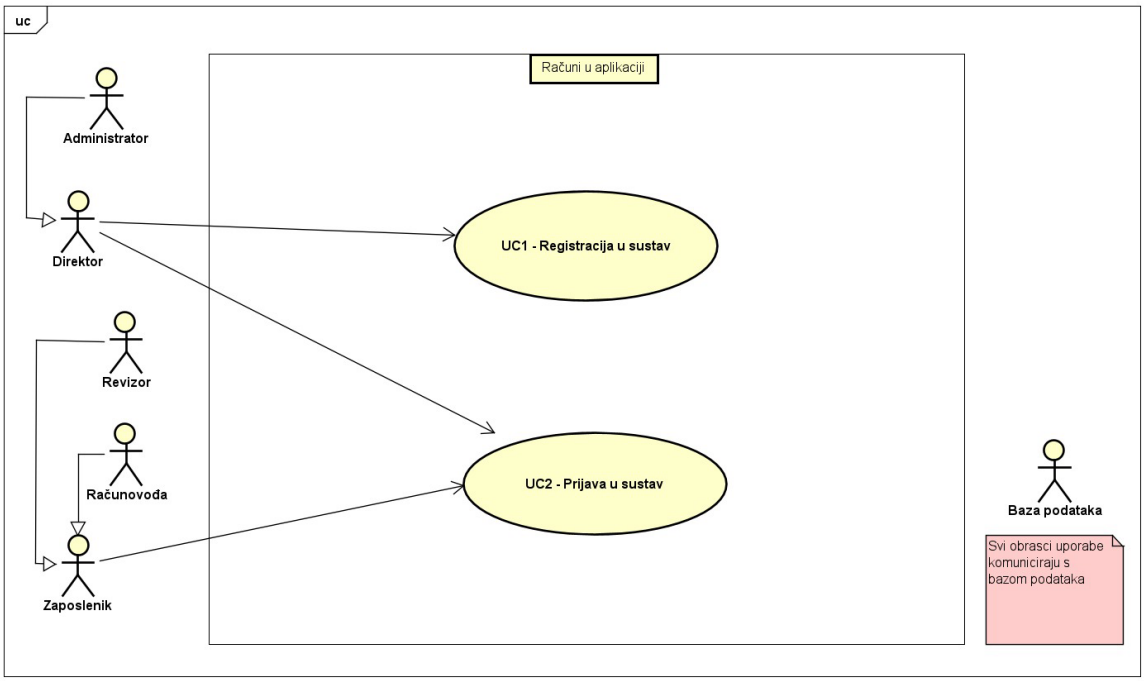
Slika 3.1: Dijagram obrasca uporabe, funkcionalnost svih aktora



Slika 3.2: Dijagram obrasca uporabe, funkcionalnost svih aktora



Slika 3.3: Dijagram obrasca uporabe, funkcionalnost Direktora i Administratora

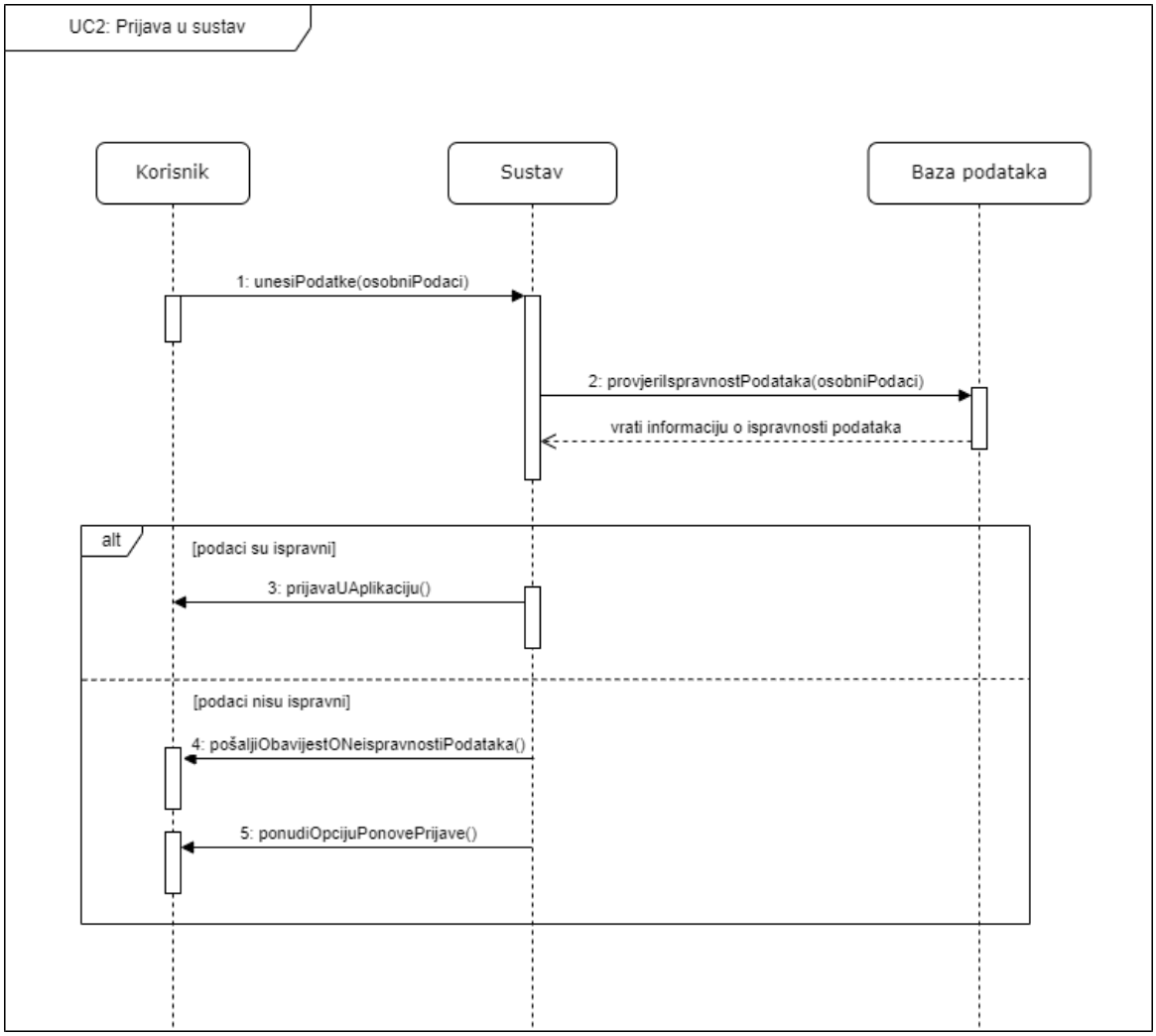


Slika 3.4: Dijagram obrasca uporabe, funkcionalnost svih aktora

3.1.2 Sekvencijski dijagrami

Obrazac uporabe UC2 - Prijava u sustav

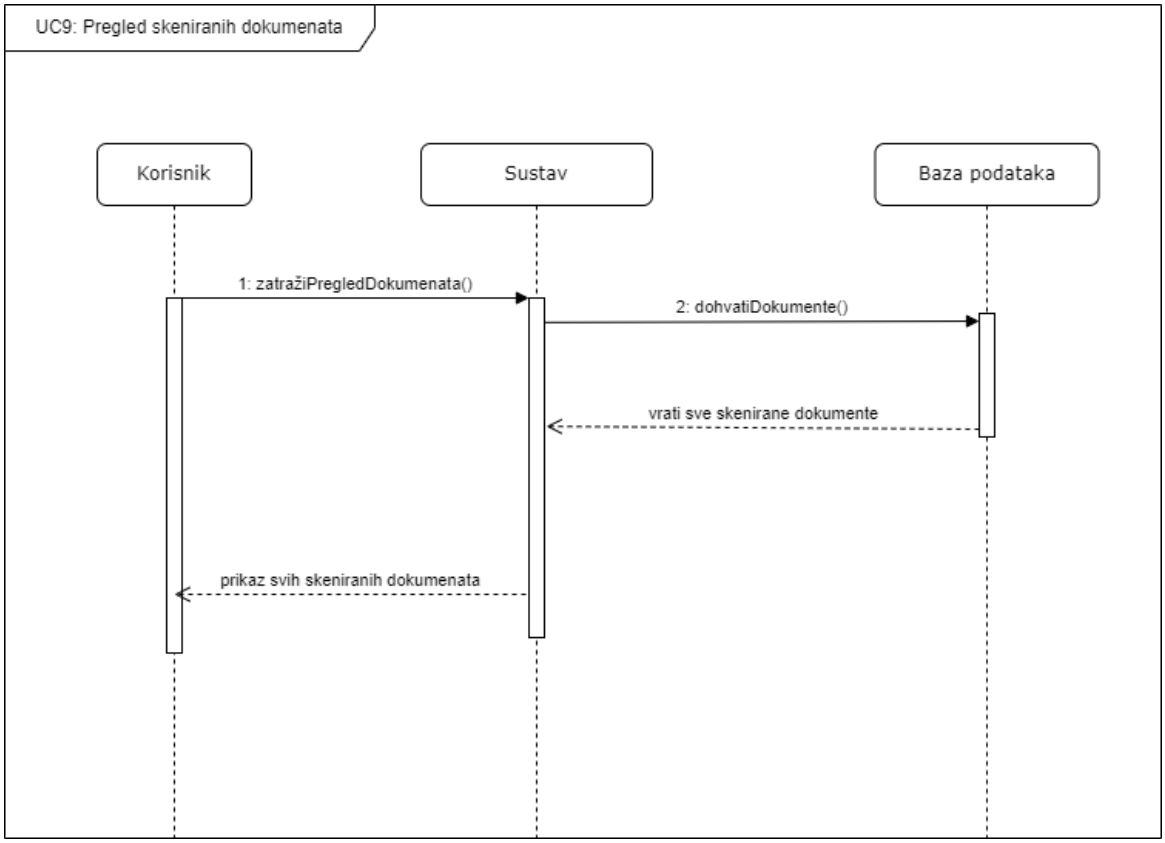
Korisnik unosi svoje osobne podatke. Sustav provjerava u bazi podataka jesu li uneseni podaci ispravni. Ako jesu, korisnik se prijavi u sustav. No ako uneseni podaci ne odgovaraju niti jednom registriranom korisniku u bazi, sustav korisniku šalje obavijest da su osobni podaci neispravni i daje mu priliku za ponovnu prijavu.



Slika 3.5: Slika broj: Sekvencijski dijagram za UC2

Obrazac uporabe UC9 - Pregled skeniranih dokumenata

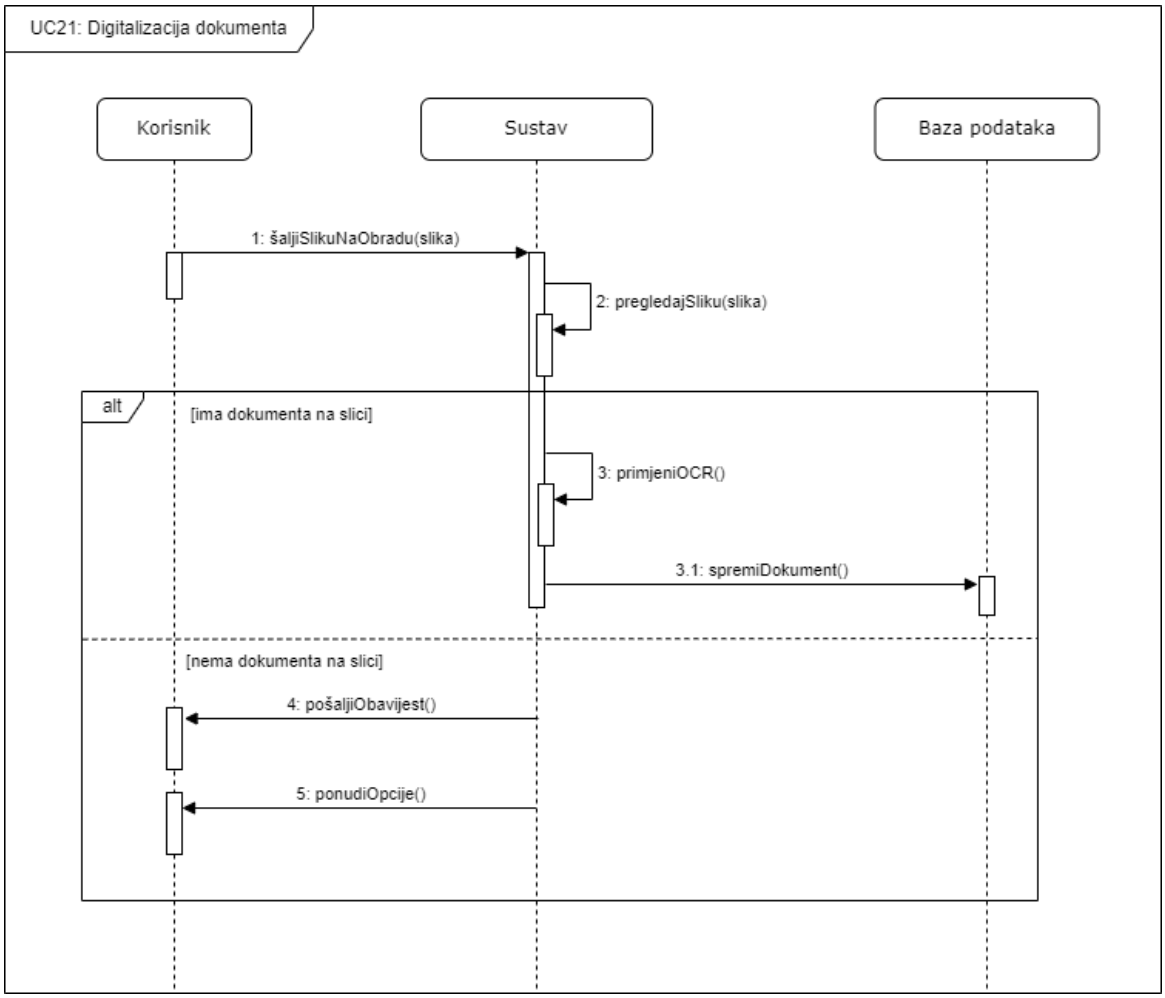
Korisnik odabire pregled popisa svih dokumenata koje je skenirao. Sustav pretraži u bazi njegove skenirane dokumente i vraća ih korisniku na uvid.



Slika 3.6: Slika broj: Sekvencijski dijagram za UC9

Obrazac uporabe UC21 - Digitalizacija dokumenta

Korisnik šalje sliku u sustav koji zatim tu sliiku učitava i pregledava je li na slici dokument. Ako je, onda na njega primjenjuje OCR te novonastali dokument sprema u bazu podataka. Ako na slici nije očitan dokument, sustav korisniku šalje obavijst da učitana slika ne sadrži dokument te mu daje opciju da priloži novu sliku.



Slika 3.7: Slika broj: Sekvencijski dijagram za UC21

3.2 Ostali zahtjevi

- Sustav treba omogućiti rad više korisnika u istom razdoblju
- Korisničko sučelje ne smije imati greške(bug-ove) kojima je moguće da klijent ima pristup ovlastima koje mu ne bi trebale biti dozvoljene
- Sustav treba biti implementiran kao mobilna aplikacija koja koristeći objektno-orijentirane jezike
- Korisničko sučelje treba biti u stanju podržavati različite dijakritičke znakove u tekstualnom sučelju
- Dohvaćanje podataka iz baze podataka i njihov prikaz ne smije trajati duže od nekoliko sekundi
- Veza s bazom mora biti sigurna i zaštićena kako bi bila otporna na krađu klasificiranih podataka tvrtke
- Sustav treba imati intuitivno i jednostavno korisničko sučelje za koje nisu potrebne upute korištenja
- Eventualna nadogradnja sustava ne smije brisati značajke aplikacije protiv volje korisnika aplikacije

4. Arhitektura i dizajn sustava

Arhitektura rješenja se sastoji od 3 sustava: Mobilne aplikacije(Frontend), Web aplikacije(Backend) i Baze podataka.

Mobilna aplikacija je softver namjenjen pokretanju na mobilnim uređajima poput mobitela i tableta. Ona pomoću API-a **(Application programming interface)** komunicira s web aplikacijom koja se brine za logiku. API je je skup pravila, protokola i definicija koje omogućuje komunikaciju između različitih programa ili aplikacija. Sama komunikacija se odvija pomoću HTTPS-a **(Hypertext Transfer Protocol Secure)**, što je sigurnija inačica HTTP-a **(Hypertext Transfer Protocol)** koji omogućuje kkomunikaciju na internetu. Kada web aplikacija primi zahtjev od mobilne aplikacije, ona ovisno o zahtjevu, pristupa bazi podataka kako bi dohvatila, promjenila ili dodala informacije. Ovisno o zahtjevu, web aplikacija vraća odgovor na orginalni zahtjev mobilne aplikacije i mobilna aplikacija prikazuje dobivene informacije kroz svoje sučelje.

Mobilna aplikacija je razvijena pomoću programskog jezika *Dart* te radnog okvira *Flutter* unutar razvojnog okruženja *Android Studio*. Web aplikacija je razvijena pomoću programskog jezika *Python* te radnog okvira *FastAPI* unutar razvojnog okruženja *PyCharm* te je postavljena na *render.com*. Sve podatke pohranjujemo unutar *PostgreSQL* baze podataka koja je na *Azure*-u.

Aplikacija se temelji na MVC konceptu u modernijoj izvedbi koja je na web aplikaciji bliža tipičnoj arhitekturi fastapi aplikacija:

- **Model** - Model predstavlja strukture podatakae. On pristupa bazi podataka, te upravlja podacima. Unutar našeg rješenja, postoji poseban sloj u web aplikaciji koji komunicira s bazom podataka (CRUD klase).

- **View** - Pogled predstavlja korisničko sučelje aplikacije te prikazuje sve informacije korisniku. Pomoću njega, korisnik može imati interakciju s ostatkom aplikacije. U našoj arhitekturi, to bi bila mobilna aplikacija koja prikazuje sve informacije i sama po sebi ima minimalnu logiku te primarna svrha je prikaz informacija korisniku.

- **Controller** - Kontroler postoji između Viewa i Modela te služi da bi primao

zahtjeve od Viewa, slao odgovore natrag Viewu, vrši poslovnu logiku te prosljeduje zahtjeve pripadnom djelu modela. Unutar naše izvedbe, Controller je dio Web aplikacije koji prima zahtjeve od mobilne aplikacije, vrši poslovnu logiku nad njima te prosljeđuje potrebne podatke iz zahtjeva modelu te od modela dobiva natrag podatke koje u prikladnom obliku prosljeđuje mobilnoj aplikaciji. U našoj izvedbi je izveden pomoću route funckija i Service klasa.

4.1 Baza podataka

Koristimo relacijsku bazu podataka s implementacijom pomoću PostgreSQL-a. Za opis naše baze podataka koristimo DBML. DBML (Database Markup Language) je jezik označavanja koji omogućuje deklarativno opisivanje relacijskih baza podataka. Time možemo jednostavno definirati strukture baze podataka pomoću tekstualnog zapisa. Zadaća baze podataka je da olakša i ubrza pristup podacima za daljnju obradu ili pronalaženje arhiviranih podataka. Glavne komponente naše baze podataka su:

- **documents**
- **users**
- **signatures**
- **audits**
- **archive**
- **image**
- **user_role**
- **roles**

4.1.1 Opis tablica

Documents Sadrži informacije o dokumentima, uključujući vrstu dokumenta, sliku dokumenta, sažetak, status, vlasnika i vremensku oznaku skeniranja. Ovaj entitet u vezi je *One-to-Many* s entitetom **users** preko atributa *owner_id*, u vezi je *One-to-Many* s entitetom **signatures** preko atributa *id*, u vezi je *One-to-Many* s entitetom **audits** preko atributa *id*, u vezi je *One-to-Many* s entitetom **archive** preko atributa *id*, u vezi je *One-to-One* s entitetom **image** preko atributa *image_id*, u vezi je *Many-to-One* s entitetom **users** preko atributa *owner_id*.

documents		
id	INT	identifikacijski broj dokumenta
document_type	ENUM	tip skeniranog dokumenta: račun, ponuda ili interni
image_id	INT	identifikacijski broj slike dokumenta
summary	TEXT	tekst koji sadrži skenirani dokument
document_status	ENUM	trenutni status dokumenta: skeniran, odobren, odbijen, potpisan, reviziran, potpisan i arhiviran ili arhiviran
owner_id	INT	identifikacijski broj korisnika koji je skenirao dokument
scan_timestamp	DATETIME	oznaka datuma i vremena skeniranja dokumenta

Users Sprema informacije o korisnicima, uključujući korisničko ime i lozinku, ime i prezime korisnika te email. Ovaj entitet u vezi je *One-to-Many* s entitetom **documents** preko atributa *id*, u vezi je *One-to-Many* s entitetom **signatures** preko atributa *id*, u vezi je *One-to-Many* s entitetom **audits** preko atributa *id*, u vezi je *One-to-Many* s entitetom **archive** preko atributa *id*, u vezi je *One-to-Many* s entitetom **user_role** preko atributa *id*.

users		
id	INT	identifikacijski broj korisnika
email	VARCHAR	email korisnika
first_name	VARCHAR	ime korisnika
last_name	VARCHAR	prezime korisnika
username	VARCHAR	korisničko ime za prijavu korisnika
password	VARCHAR	šifa za prijavu korisnika

Signatures Služi za praćenje potpisa na dokumentima, s informacijama o korisniku koji je potpisao, statusu potpisa i vremenskoj oznaci potpisa. Ovaj entitet u vezi je *Many-to-One* s entitetom **users** preko atributa *sign_by*, u vezi je *One-to-One*

s entitetom **documents** preko atributa *document_id*.

signatures		
signature_id	INT	identifikacijski broj potpisanog dokumenta
document_id	INT	identifikacijski broj dokumenta
sign_by	INT	identifikacijski broj korisnika koji je potpisao ili treba potpisati dokument
status	ENUM	status dokumenta za potpisivanje: potpisan ili čeka na potpisivanje
signed_at	DATETIME	oznaka datuma i vremena potpisivanja dokumenta

Audits Sadrži podatke o revizijama dokumenata, uključujući korisnika koji je obavio reviziju, status revizije i vremensku oznaku. Ovaj entitet u vezi je *Many-to-One* s entitetom **users** preko atributa *audited_by*, u vezi je *One-to-One* s entitetom **documents** preko atributa *document_id*.

audits		
audit_id	INT	identifikacijski broj reviziranog dokumenta
document_id	INT	identifikacijski broj dokumenta
audited_by	INT	identifikacijski broj korisnika koji je revizirao ili treba revizirati dokument
status	ENUM	status dokumenta za reviziju: reviziran ili čeka na reviziranje
audited_at	DATETIME	oznaka datuma i vremena reviziranja dokumenta

Archive Koristi se za praćenje arhiviranja dokumenata, s informacijama o korisniku koji je arhivirao dokument, statusu arhiviranja, vremenskoj oznaci i broju arhive. Ovaj entitet u vezi je *One-to-One* s entitetom **documents** preko atributa *document_id*, u vezi je *Many-to-One* s entitetom **users** preko atributa *archive_by*.

archive		
archive_number	INT	identifikacijski broj arhiviranog dokumenta
document_id	INT	identifikacijski broj dokumenta
archive_by	INT	identifikacijski broj korisnika koji je arhivirao ili treba arhivirati dokument
status	ENUM	status dokumenta za arhiviranje: arhiviran ili čeka na arhiviranje
archived_at	DATETIME	oznaka datuma i vremena arhiviranja dokumenta

Image Sprema slike dokumenata. Ovaj entitet u vezi je *One-to-One* s entitetom **documents** preko atributa *id*.

image		
id	INT	identifikacijski broj slike dokumenta
image	VARCHAR	putanja do spremljene slike

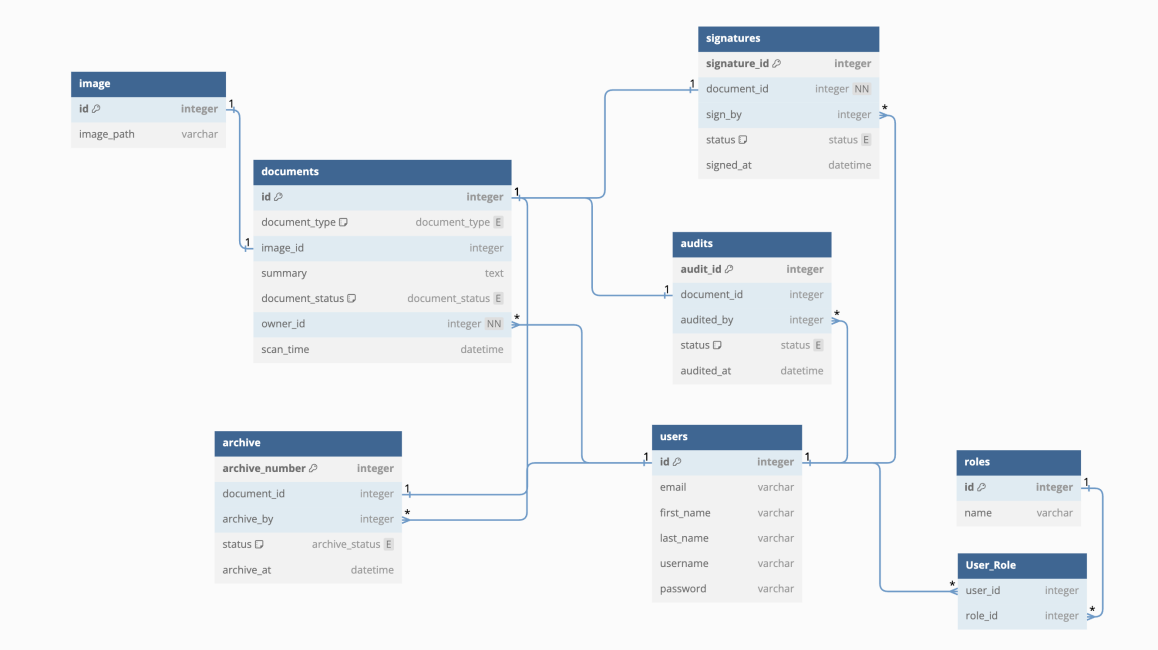
Roles Definira različite uloge korisnika u sistemu. Ovaj entitet u vezi je *One-to-Many* s entitetom **user_role** preko atributa *role_id*.

roles		
role_id	INT	identifikacijski broj kategorije korisnika
role_name	VARCHAR	ime uloge korisnika

User_role Tablica koja povezuje korisnike s njihovim ulogama. Ovaj entitet u vezi je *Many-to-One* s entitetom **roles** preko atributa *role_id*, u vezi je *Many-to-One* s entitetom **users** preko atributa *user_id*.

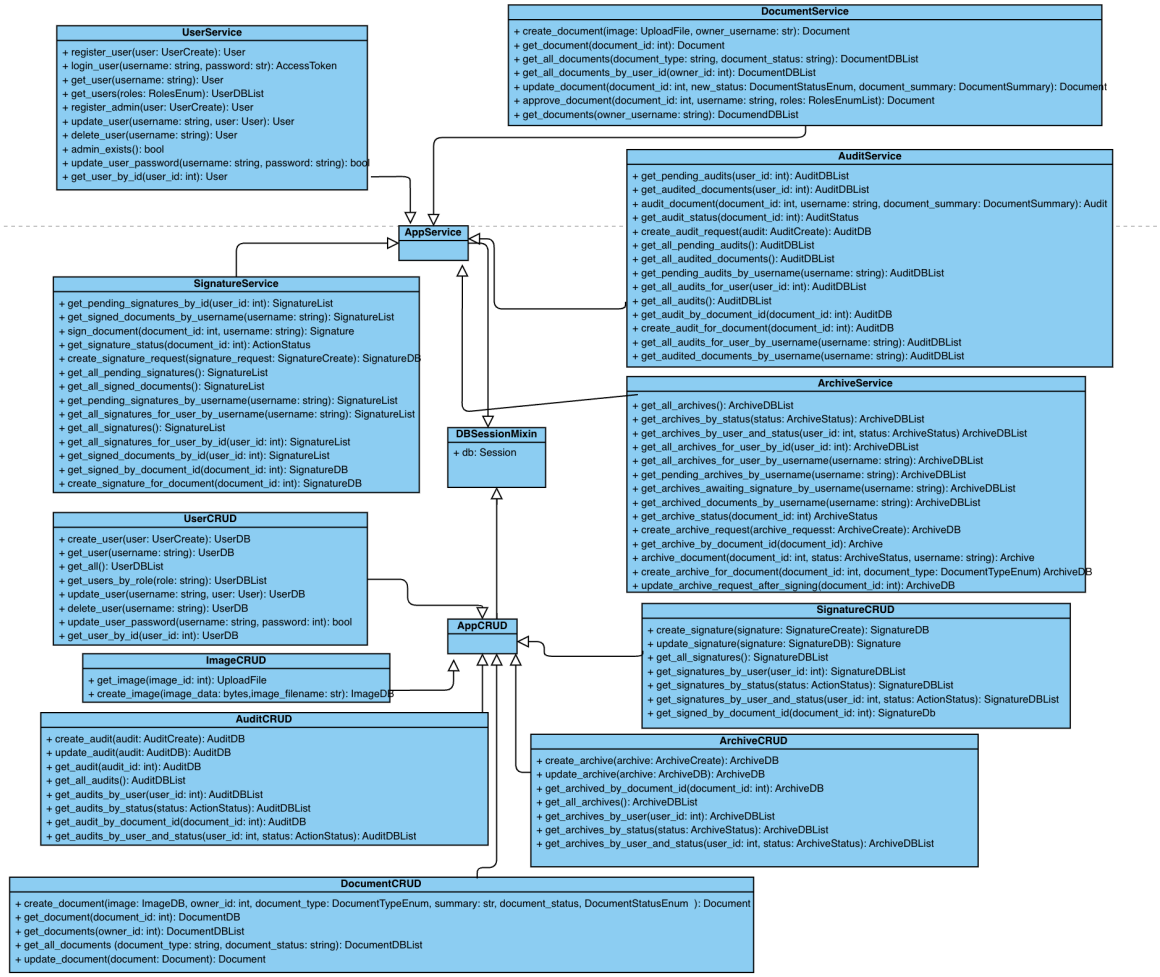
user_role		
user_id	INT	identifikacijski broj korisnika
role_id	INT	identifikacijski broj kategorije korisnika

4.1.2 Dijagram baze podataka

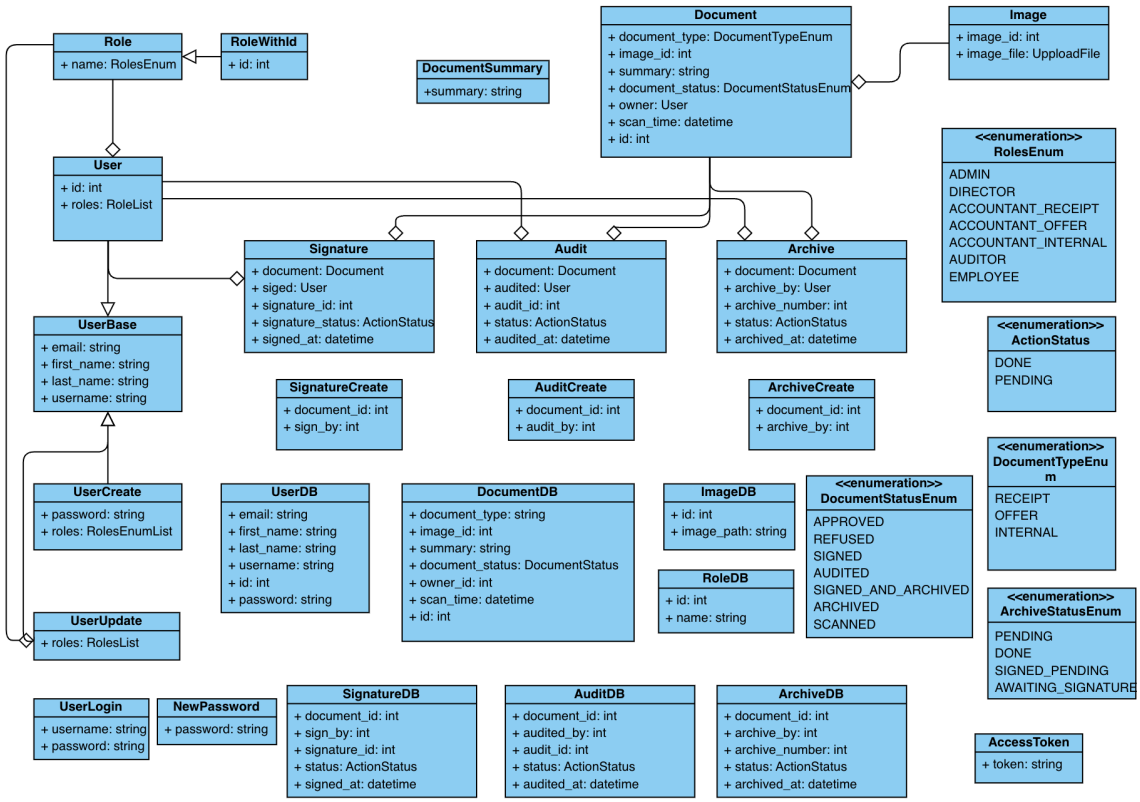


Slika 4.1: Dijagram baze podataka

4.2 Dijagram razreda



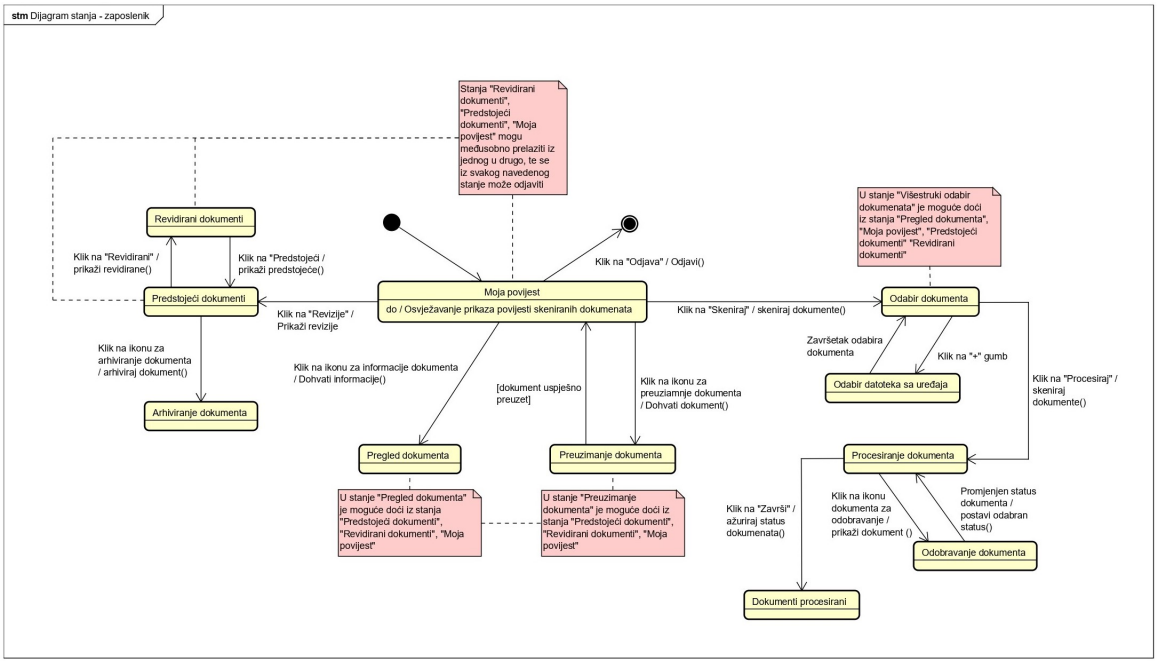
Slika 4.2: Dijagram razreda - servisi i CRUD



Slika 4.3: Dijagram razreda - shema i model

4.3 Dijagram stanja

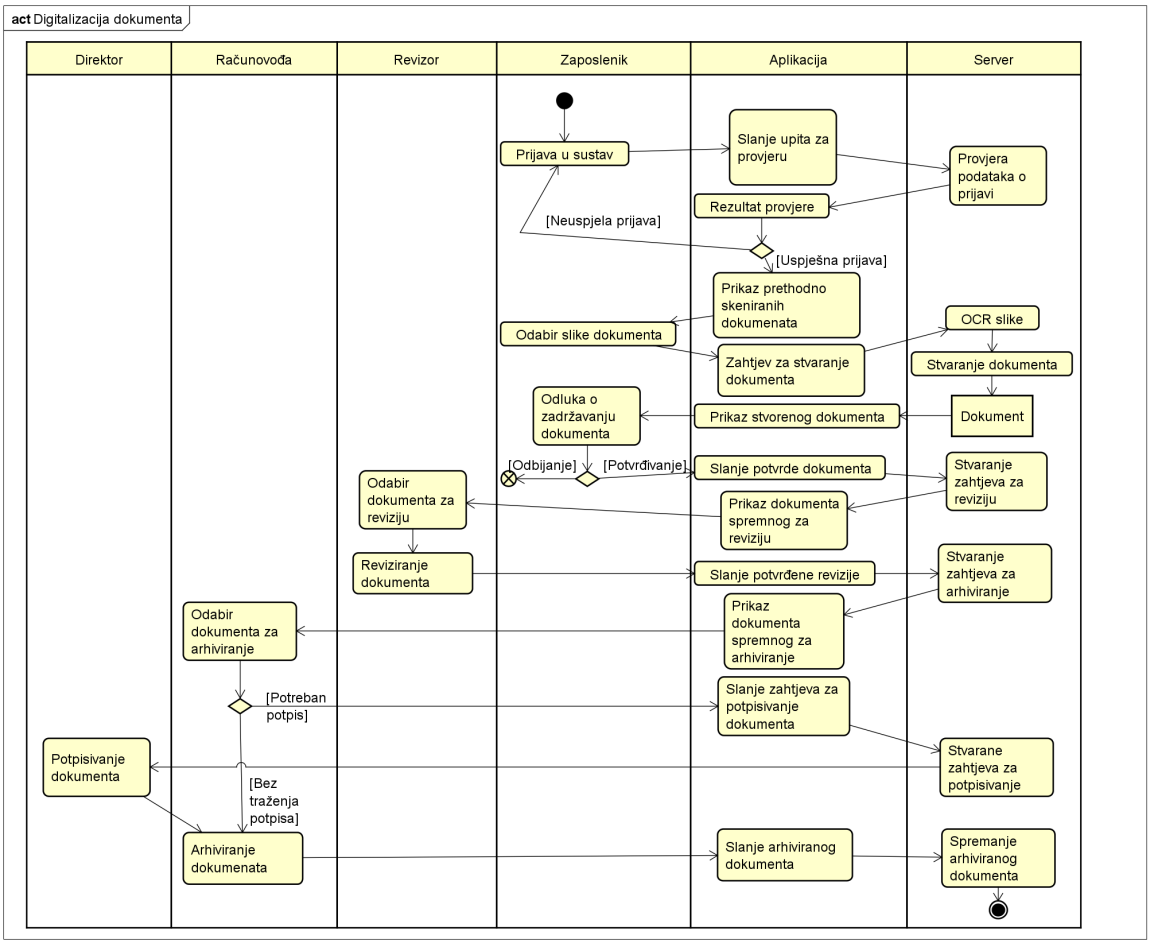
Na dijagramu stanja su prikazani prijelazi između stanja pomoću određenih događanja. Prikazan je dijagram stanja za zaposlenika prijavljenog u sustav. Nakon što je korisnik uspješno prijavljen u sustav na zaslonu mu se prikazuje zaslon sa svim njegovim prijašnjim dokumentima učitanim u sustav do kojeg je također moguće doći i pritiskom na ”Moja povijest”. Za svaki učitani dokument postoji opcija pregleda njegovog sadržaja te preuzimanje samog dokumenta lokalno na uređaj. Pritiskom na dodatni izbornik korisnik može također odabrati opciju da vidi predstojeće dokumente i revidirane dokumente pritiskom na ”Revizije”. Također zaposlenik ima i mogućnost skenirati i dodati novi dokument u sustav klikom na ”Skeniraj” koji je vidljiv na svima prije naveenim zaslonima.



Slika 4.4: Dijagram stanja

4.4 Dijagram aktivnosti

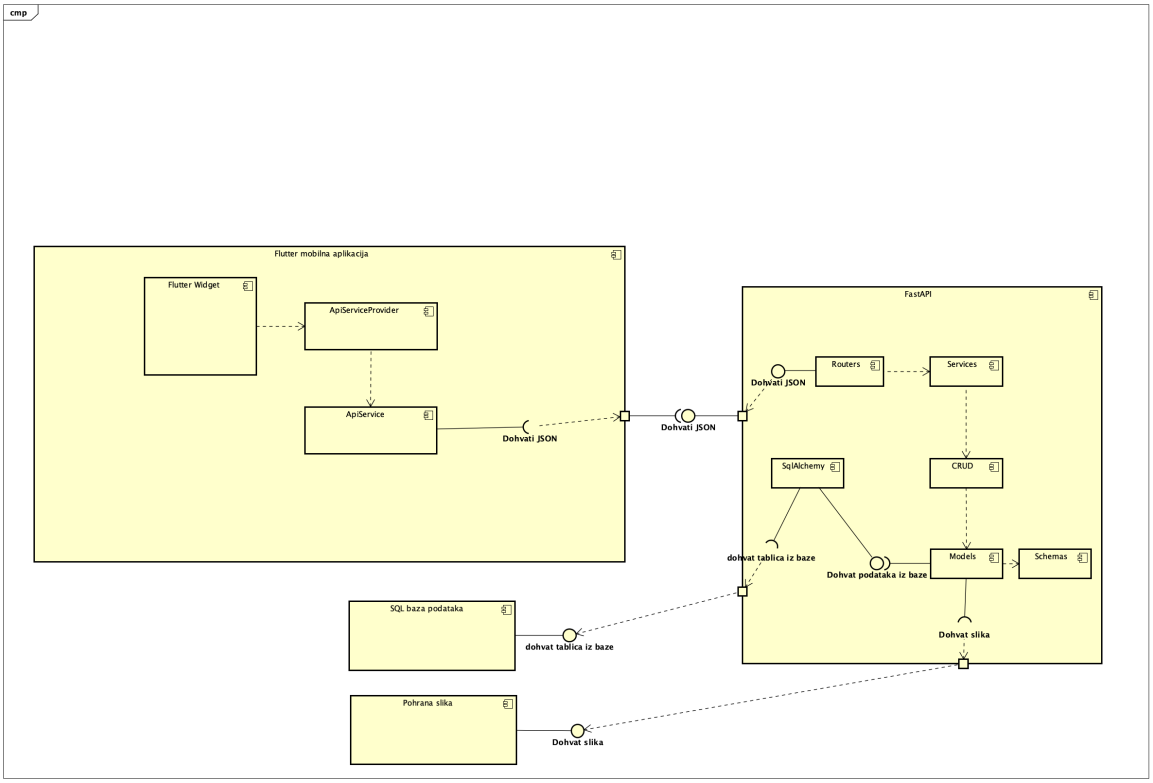
Na dijagramu aktivnosti prikazan je proces digitalizacije dokumenta. Nakon prijave zaposlenika slijedi odabir slike dokumenta koji se šalje na obradu. Stvara se objekt dokumenta koji se zatim redom šalje na reviziju, arhiviranje i po potrebi na potpis.



Slika 4.5: Dijagram aktivnosti - digitalizacija dokumenta

4.5 Dijagram komponenti

Dijagram komponenti prikazan na slici 4.5 opisuje organizaciju komponenti, internih struktura i komunikaciju različitih dijelova aplikacije. Sustav možemo podijeliti na 2 cjeline: **Flutter mobilnu aplikaciju** i **FastAPI backend**. Korisničko sučelje se nalazi unutar **Flutter mobilne aplikacije**, te sve korisničke interakcije su unutar te mobilne aplikacije. Aplikacija se sastoji od hijerarhije *Flutter widgeta*, koji su zapravo manje cjeline koje grade korisničko sučelje. Interakcijom s korisničkim sučeljem, widget s pomoću *ApiServiceProvidera* koristi *ApiService* te taj servis dohvaća JSON podatke s backend dijela aplikacije. FastAPI backend je REST API i on s pomoću *routera* preusmjerava requestove pripadajućem servisu koji ih dalje preusmjerava pripadajućim *CRUD* klasama. *CRUD* klase zatim preko *Modela* s pomoću *SQLAlchemy*-a dohvaćaju podatke iz baze podataka ili slike s pohrane na oblaku. Ti podaci su pri komunikaciji u obliku *model* objekata ili *schema* pri obrađivanju zahtjeva i odgovoru na njih. Ovisno o JSON podacima koji su se dobili s backenda, frontend prilagođava svoje sučelje.



Slika 4.6: Dijagram komponenti

5. Implementacija i korisničko sučelje

5.1 Korištene tehnologije i alati

Za komunikaciju unutar tima koristili smo **Whatsapp**[1] i **Discord**[2]. Za izradu UML dijagrama koristili smo: **dbdiagram**[3] za dijagram baze podataka, **Visual Paradigm Online**[4] za dijagram razreda te **Astah**[5] za ostale UML dijagrame. Kao sustav upravljanja verzijama je korišten **Git**[6] te su naši repozitoriji bili dostupni na **Github**u[7] udaljenom repozitoriju.

Kao razvojno okruženje je korišten **PyCharm**[8] za razvoj API-ja te **Android Studio**[9] za razvoj mobilne aplikacije. Pycharm je razvojno okruženje za python, napravljeno od JetBrains-a te Android Studio je modificirana inačica drugog JetBrains razvojnog okruženja (IntelliJ, razvojno okruženje za Javu), prilagođena razvoju android mobilnih aplikacija.

Mobilna aplikacija je napravljena s pomoću **Flutter**[10] razvojnog okvira koji koristi **Dart**[11] programski jezik. Flutter i Dart su projekti razvijeni od strane Googlea. Flutter je razvojni okvir za izradu aplikacija na različitim platformama, vrlo popularan zbog mogućnosti kreiranja aplikacije za više platforma paralelno. Dart je moderni programski jezik koji se primarno koristi za razvoj Flutter aplikacija. API je kreiran s pomoću **Pythona**[12] te **FastAPI**[13] razvojnog okvira. FastAPI je novije razvojno okruženje bazirano na Flasku, vrlo popularnom python razvojnom okruženju, s mogućnosti paralelnog izvođenja zahtjeva te automatskom dokumentacijom po OpenAPI standardu.

[1] https://www.whatsapp.com/
[2] https://discord.com/
[3] https://dbdiagram.io/
[4] https://online.visual-paradigm.com/
[5] https://astah.net/
[6] https://git-scm.com/
[7] https://github.com/
[8] https://www.jetbrains.com/pycharm/
[9] https://developer.android.com/studio
[10] https://flutter.dev/
[11] https://dart.dev/
[12] https://www.python.org/
[13] https://fastapi.tiangolo.com/

Aplikacija je puštena u pogon na **Renderu**[14], te je korišten **Microsoft Azure**[15]. za spremanje slika, za prepoznavanje teksta na slikama te se tamo nalazi naša baza podataka.

[14] https://render.com/
[15] https://azure.microsoft.com/

5.2 Ispitivanje programskog rješenja

U naredna dva potpoglavlja opisuje se testiranje komponenti i testiranje sustava. Testiranje komponenti vrši se na poslužiteljskoj strani, a testiranje sustava iz okvira mobilne aplikacije.

5.2.1 Ispitivanje komponenti

Ispitivanje funkcionalnosti sustava provedeno je uz pomoć unit testova. Testirani su service razredi te pripadajući routeri. U nastavku je priloženo nekoliko primjera tih testova te prikaz rezultata testiranja u razvojnom okruženju.

Test za razred UserService kojim se testira ispravno kreiranje novog korisnika. Servisu se šalje objekt tipa UserCreate te u slučaju da direktor još ne postoji očekujemo da će povratna vrijednost biti objekt User s podacima o novom korisniku.

```
1  user_create = UserCreate(email="mail@mail.com", first_name="Test",
     last_name="Test", username="test", password="test", roles=[RolesEnum.
     ADMIN],)
2  user = User(id=1, email="test@email.com", first_name="Test", last_name
     ="Test", username="test", roles=[],)
3
4  def test_register_user():
5    mock_user_crud = Mock()
6    mock_user_crud.create_user.return_value = user.model_dump()
7    mock_user_crud.get_users_by_role.return_value = None
8    db = Mock()
9
10   user_service = UserService(db)
11
12   with patch("app.services.users.UserCRUD",
13         return_value=mock_user_crud):
14     result = user_service.register_user(user_create)
15
16   mock_user_crud.create_user.assert_called_once_with(user_create)
17   mock_user_crud.get_users_by_role.assert_not_called()
18   assert result == user.model_dump()
```

Sljedećim testom za documents router testira se reakcija pokušaja dohvaćanja povijesti skeniranih dokumenata za korisnika koji nije ulogiran te se očekuje 'NotAuthenticated' iznimka od servera.

```
1  def test_get_me_not_authenticated():
2      response = client.get("/documents/me")
3      assert response.status_code == 401
4      assert response.json() == {'app_exception': 'NotAuthenticated', '
       context': {}}
```

Test za razred DocumentService koji testira kreiranje dokumenta. Servisu šaljemo sliku i korisničko ime te ako je dokument uspješno kreiran očekujemo da će nam servis vratiti kreirani novi objekt tipa Document sa svim podacima o dokumentu.

```
1  @patch("app.services.documents.detect_document", return_value="R123456
       text")
2  def test_create_document(detect_document):
3      mock_document_crud = Mock()
4      mock_document_crud.create_document.return_value = document1
5
6      mock_user_service = Mock()
7      mock_user_service.get_user.return_value = admin
8
9      mock_image_service = Mock()
10     mock_image_service.create_image.return_value = imageDB
11     db = Mock()
12
13     document_service = DocumentService(db)
14
15     with patch("app.services.documents.DocumentCRUD", return_value=
       mock_document_crud), patch("app.services.documents.ImageService",
       return_value=mock_image_service), patch("app.services.documents.
       UserService", return_value=mock_user_service):
16         result = document_service.create_document(uploaded_image, "
       username")
17
18     detect_document.assert_called_once()
19     mock_user_service.get_user.assert_called_once()
20     mock_image_service.create_image.assert_called_once()
21
22     assert result == document1
```

Sljedeća dva testa testiraju ažuriranje statusa dokumenta u razredu DocumentService. Prvi provjerava dozvoljeni slučaj iz APPROVED želimo status postaviti na AUDITED, stoga nakon što servisu pošaljemo id dokumenta novi status te eventualne promjene u samom tekstualnom sadržaju očekujemo da će nam vratiti taj

dokument s novim statusom. Drugi provjerava slučaj u kojem smo nakon statusa APPROVED odmah skočili na status ARCHIVED. Budući da to nije dozvoljeni slijed statusa očekujemo da će nam servis baciti iznimku "DocumentException.DocumentStatusNotCompatible".

```
1  document2 = Document(id=2, image_id=1, owner=director, document_type=
     DocumentTypeEnum.INTERNAL, summary="test", document_status=
     DocumentStatusEnum.APPROVED, scan_time="2021-01-01T00:00:00",)
2  document3 = Document(id=2, image_id=1, owner=director, document_type=
     DocumentTypeEnum.INTERNAL, summary="test", document_status=
     DocumentStatusEnum.AUDITED, scan_time="2021-01-01T00:00:00",)
3
4  def test_update_document():
5      mock_document_crud = Mock(spec=DocumentCRUD)
6      mock_document = Mock(spec=document2, document_status=
     DocumentStatusEnum.APPROVED)
7      mock_document_crud.get_document.return_value = mock_document
8      mock_document_crud.update_document.return_value = document3
9      db = Mock()
10
11     document_service = DocumentService(db)
12
13     with patch("app.services.documents.DocumentCRUD", return_value=
     mock_document_crud):
14     result = document_service.update_document(2, DocumentStatusEnum.
     AUDITED, None)
15
16     mock_document_crud.update_document.assert_called_once()
17     assert result == document3
18
19 def test_update_document_with_invalid_status():
20     mock_document_crud = Mock(spec=DocumentCRUD)
21     mock_document = Mock(spec=document2, document_status=
     DocumentStatusEnum.APPROVED)
22     mock_document_crud.get_document.return_value = mock_document
23     mock_document_crud.update_document.return_value = document3
24     db = Mock()
25
26     document_service = DocumentService(db)
27
28     with patch("app.services.documents.DocumentCRUD", return_value=
     mock_document_crud):
29     with raises(DocumentException.DocumentStatusNotCompatible):
```

```
30      document_service.update_document(2, DocumentStatusEnum.ARCHIVED,
        None)
```

Posljednji testovi testiraju arhiviranje dokumenta u sklopu ArchiveService-a. Dokument se smije arhivirati ako je prethodni status bio PENDING ili SIGNED_PENDING. U prvom testu želimo arhivirati dokument čiji status arhiviranja je PENDING te ga želimo postaviti na DONE što je dozvoljeni slijed statusa. Očekuje se uspješni završetak metode archive_document koja onda vraća arhivirani dokument s novim statusom DONE. U drugom testu želimo isprobati slučaj u kojem pokušavamo dokument iz AWAITING_SIGNATURE statusa pretvoriti u DONE što nije dopušteni slijed stanja pa je i očekivano bacanje iznimke "ArchiveException.IllegalArchiveStatus".

```
1   def test_archive_document():
2       mock_archive_crud = Mock()
3       mock_archive_crud.get_archived_by_document_id.return_value =
        archive1
4       mock_archive_crud.update_archive.return_value = archive1
5
6       mock_user_service = Mock(spec=UserService)
7       mock_user_service.get_user.return_value = admin
8
9       mock_document_service = Mock(spec=DocumentService)
10      mock_document_service.update_document.return_value = None
11
12      db = Mock()
13
14      archive_service = ArchiveService(db)
15
16      with patch("app.services.archives.ArchiveCRUD", return_value=
        mock_archive_crud):
17          with patch("app.services.archives.UserService", return_value=
        mock_user_service):
18              with patch("app.services.archives.DocumentService", return_value
        =mock_document_service):
19                  result = archive_service.archive_document(1, ArchiveStatus.
        DONE, "test")
20
21      mock_archive_crud.get_archived_by_document_id.
        assert_called_once_with(1)
22      mock_archive_crud.update_archive.assert_called_once()
23      assert result == archive1
24
```

```
25  def test_archive_document_forbidden():
26      mock_archive_crud = Mock()
27      mock_archive_crud.get_archived_by_document_id.return_value =
        archive3
28      mock_archive_crud.update_archive.return_value = archive1
29
30      mock_user_service = Mock(spec=UserService)
31      mock_user_service.get_user.return_value = admin
32
33      mock_document_service = Mock(spec=DocumentService)
34      mock_document_service.update_document.return_value = None
35
36      db = Mock()
37
38      archive_service = ArchiveService(db)
39
40      with patch("app.services.archives.ArchiveCRUD", return_value=
        mock_archive_crud):
41          with patch("app.services.archives.UserService", return_value=
        mock_user_service):
42              with patch("app.services.archives.DocumentService", return_value
        =mock_document_service):
43                  with raises(ArchiveException.IllegalArchiveStatus):
44                      archive_service.archive_document(1, ArchiveStatus.DONE, "
        test")
```

Svi navedeni ispitni slučajevi prošli su testiranje.

```
tests/archives/service_test.py::test_archive_document PASSED
tests/archives/service_test.py::test_archive_document_forbidden PASSED
tests/documents/router_test.py::test_get_me_not_authenticated PASSED
tests/documents/service_test.py::test_create_document PASSED
tests/documents/service_test.py::test_update_document PASSED
tests/documents/service_test.py::test_update_document_with_invalid_status PASSED
tests/users/service_test.py::test_register_user PASSED
```

Slika 5.1: Prikaz rezultata ispitnih slučajeva u razvojnom okruženju

5.2.2 Ispitivanje sustava

Razvojni okvir Flutter pruža niz službenih, integriranih alata za izvršavanje više vrsta testova. Ovisno o razini ili dijelu sustava koji se testira dostupni testovi

unutar Flutter paketa su Unit, Widget i Integration testovi. Unutar Flutter okvira Integration testovi služe za testiranje čitavog ili pojedinih dijelova sustav, odnosno preuzimaju ulogu integracijskih i sustavnih testova ovisno o implementaciji, a pišu se unutar samog projekta s pomoću jezika Dart, što olakšava održavanje. Integrirani paketi također pružaju mogućnost automatizacije testova, simulacije i praćenja korisničkih unosa, praćenje tijeka izvođenja i mrežnog prometa na simuliranim i stvarnim uređajima te lakog pregleda uspješnosti i ispisa provedenih testova. Iz navedenih razloga za testiranje aplikacije korišteni su integrirani alati dostupni unutar paketa flutter_test i integration_test. Druga mogućnost bila bi uporaba Appium alata koji se temelji na Selenium strukturi. Dostupni su preko javno održavanih paketa za prilagodbu, no integrirani alati pokazuju se boljim rješenjem za dani slučaj.

U nastavku su opisani izvedeni testovi sustava. Tijelo svakog testa se sastoji od grupa, a tijelo svake grupe od podtestova. Sve grupe i podtestovi se vrše slijedno, sukladno hijerarhiji definiranoj u implementaciji te predstavljaju korake i potkorake u testiranju. Cilj sljedećih testova je utvrditi ispunjava li potpuno integrirani sustav zadaće u konkretnim slučajevima i prilagođava li se mogućim rubnim slučajevima. Također se želi sagledati mogu li se svi testovi izvršiti slijedno čime se utvrđuje stabilnost sustava. Za svaki testni slučaj opisani su ulazni podaci i početno stanje, koraci testa, očekivani i stvarni rezultat te ocjena rezultata i tijeka izvođenja. Testovi se izvode u trenutku kad postoji konkretna količina podataka za testiranje kao i sve vrste potrebnih korisnika.

Test uspješne i neuspješne prijave te odjave

- **Ulazni podaci:**
  - username1: admin
  - password1: admin
  - username2: admin
  - password2: neadmin
- **Početno stanje:** Aplikacija u početnom stanju bez prijavljenog korisnika.
- **Koraci testa:** Koraci koji se provode u testu jesu; upisivanje prvih ulaznih podataka u formular za prijavu i pritisak na gumb za prijavu, navigacija na home screen, otvaranje sučelja za navigaciju i pritisak na gumb za odjavu,

upisivanje drugih ulaznih podatak u formular za prijavu i pritisak na gumb za prijavu.

- **Očekivani rezultat:** Očekuje se navigacija na home screen nakon upisa valjanih podataka za prijavu, pravilno otvaranje sučelja za navigaciju, povratak na login screen te ostanak na istom nakon upisa nevaljanih podataka.
- **Stvarni rezultat:** Provedeni koraci se poklapaju s očekivanim, test prolazi.
- **Ocjena:** Testirane funkcionalnosti su zadovoljavajuće, rade ispravno, no može se primijetiti nedostatak povratnih informacija pri neuspješnom prijavljivanju zbog neimplementiranih funkcionalnosti.

Test pregleda korisnika

- **Ulazni podaci:**
  - username: owner
  - password: owner
- **Početno stanje:** Nastavak nakon prethodnog testa, bez prijavljenog korisnika (reset sa zastavicom.
- **Koraci testa:** Koraci koji se provode u testu jesu; upisivanje ulaznih podataka u formular za prijavu i pritisak na gumb za prijavu, navigacija na home screen, otvaranje sučelja za navigaciju i navigacija na users screen (Korisnici), ako se pronađe korisnik pritisak na gumb za statistiku, nakon otvaranja pritisak na gumb za zatvaranje dijaloga.
- **Očekivani rezultat:** Očekuje se navigacija na home screen nakon upisa valjanih podataka za prijavu, pravilno otvaranje sučelja za navigaciju, navigacija na users screen, ako se pronađe korisnik uspješno otvaranje dijaloga za pregled statistike te uspješno zatvaranje dijaloga i ostanaka na users screen.
- **Stvarni rezultat:** Provedeni koraci se poklapaju s očekivanim, test prolazi.
- **Ocjena:** Testirane funkcionalnosti su zadovoljavajuće, rade ispravno.

Test pregleda dokumenata

- **Ulazni podaci:**
  - username: owner
  - password: owner
- **Početno stanje:** Nastavak nakon prethodnog testa, bez prijavljenog korisnika (reset sa zastavicom).
- **Koraci testa:** Koraci koji se provode u testu jesu; upisivanje ulaznih podataka u formular za prijavu i pritisak na gumb za prijavu, navigacija na home screen, otvaranje sučelja za navigaciju i navigacija na documents screen (Povijest), ako se pronađe dokument pritisak na gumb za pregled PDF dokumenta i navigacija na sučelje za pregled, simulacija izlaska iz sučelja.
- **Očekivani rezultat:** Očekuje se navigacija na home screen nakon upisa valjanih podataka za prijavu, pravilno otvaranje sučelja za navigaciju, navigacija na documents screen (Povijest), ako se pronađe dokument uspješno otvaranje sučelja za pregled PDF dokumenta te uspješan povratak na pregled, u suprotnom ostanak na documents screen.
- **Stvarni rezultat:** Provedeni koraci se poklapaju s očekivanim, test prolazi.
- **Ocjena:** Testirane funkcionalnosti su zadovoljavajuće, rade ispravno.

Test revizije dokumenata

- **Ulazni podaci:**
  - username: revizor1
  - password: revizor1
- **Početno stanje:** Nastavak nakon prethodnog testa, bez prijavljenog korisnika (reset sa zastavicom).
- **Koraci testa:** Koraci koji se provode u testu jesu; upisivanje ulaznih podataka u formular za prijavu i pritisak na gumb za prijavu, navigacija na home screen, otvaranje sučelja za navigaciju i navigacija na revision screen (Revizije), ako se pronađe dokument pritisak na gumb za revidiranje i navigacija na sučelje za revidiranje, pritisak na gumb za revidiranje, otvaranje dijaloga

za uspješnu reviziju, pritisak na gumb za zatvaranje dijaloga i povratak na revision screen, događa se reset, ponavljaju se koraci za navigaciju na revision screen, pritisak na tab za revidirane dokumente (Revidirani).

- **Očekivani rezultat:** Očekuje se navigacija na home screen nakon upisa valjanih podataka za prijavu, pravilno otvaranje sučelja za navigaciju, navigacija na revision screen (Revizije), ako se pronađe dokument uspješno otvaranje sučelja za revidiranje, uspješno revidiranje i otvaranje povezanog dijaloga, zatvaranje dijaloga i izlazak na revision screen, nakon reseta uspješan povratak na revision screen i uspješno otvaranje taba za pregled revidiranih dokumenata.
- **Stvarni rezultat:** Provedeni koraci se poklapaju s očekivanim, test prolazi.
- **Ocjena:** Testirane funkcionalnosti su zadovoljavajuće, rade ispravno.

Test arhiviranja dokumenata

- **Ulazni podaci:**
  - username: racunovoda1
  - password: racunovoda1
- **Početno stanje:** Nastavak nakon prethodnog testa, bez prijavljenog korisnika (reset sa zastavicom).
- **Koraci testa:** Koraci koji se provode u testu jesu; upisivanje ulaznih podataka u formular za prijavu i pritisak na gumb za prijavu, navigacija na home screen, otvaranje sučelja za navigaciju i navigacija na archive screen (Arhiva), ako se pronađe dokument pritisak na gumb za arhiviranje i navigacija na sučelje za arhiviranje, pritisak na gumb za arhiviranje, otvaranje dijaloga za uspješno arhiviranje, pritisak na gumb za zatvaranje dijaloga i povratak na archive screen, događa se reset, ponavljaju se koraci za navigaciju na archive screen, pritisak na tab za arhivirane dokumente (Arhivirani).
- **Očekivani rezultat:** Očekuje se navigacija na home screen nakon upisa valjanih podataka za prijavu, pravilno otvaranje sučelja za navigaciju, navigacija na archive screen (Arhiva), ako se pronađe dokument uspješno otvaranje sučelja za arhiviranje, uspješno arhiviranje i otvaranje povezanog dijaloga,

zatvaranje dijaloga i izlazak na archive screen, nakon reseta uspješan povratak na archive screen i uspješno otvaranje taba za pregled arhiviranih dokumenata.

- **Stvarni rezultat:** Provedeni koraci se poklapaju s očekivanim, test prolazi.
- **Ocjena:** Testirane funkcionalnosti su zadovoljavajuće, rade ispravno.

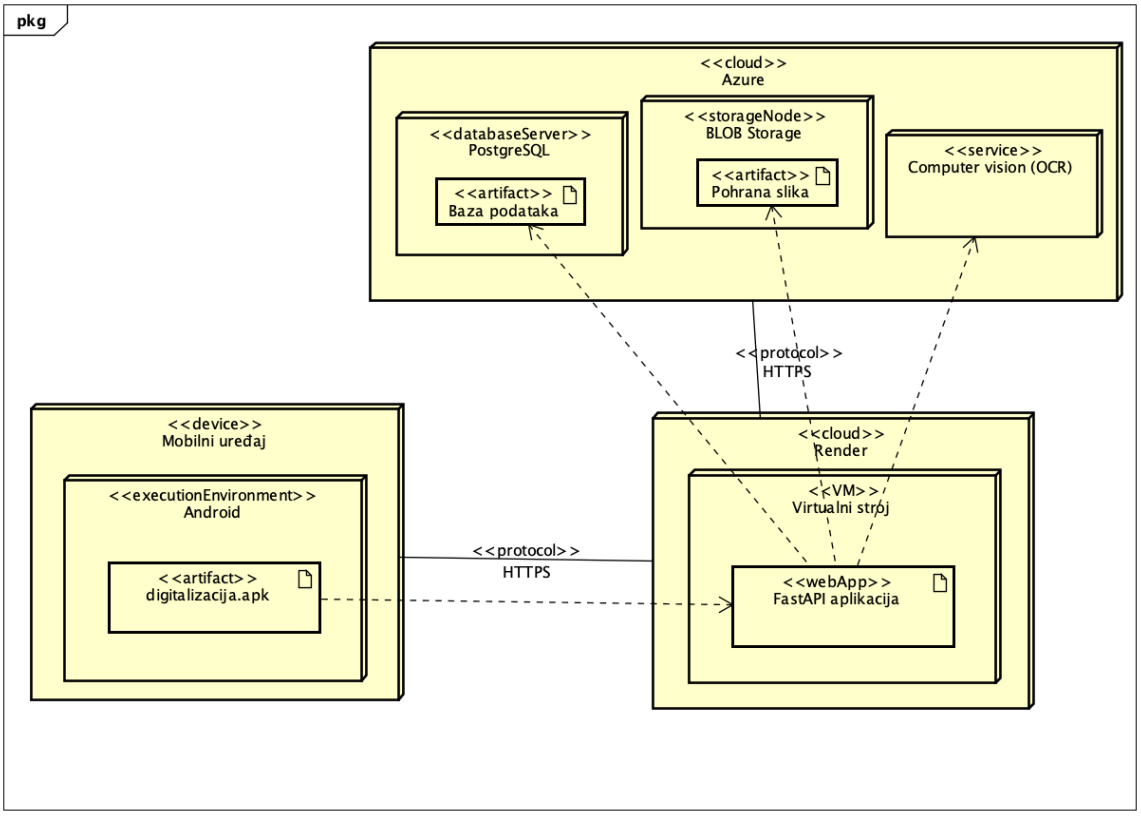
Zaključak i osvrt na testiranje sustava

Svi testovi uspješno su slijedno provedeni čime se može zaključiti kako su proučene funkcionalnosti skladne i ispravne. Nisu pronađene greške u sučelje koje onemogućuju izvođenje očekivanih nizova akcija. Nisu pronađene greške u pozadinskom baratanju stanja koje bi onemogućilo pravilno izmjenjivanja sesija, što se može zaključiti iz ispravne promjene aktivnih korisnika između testova. Dohvaćanje podataka i komunikacija s poslužiteljem u danim slučajevima također se mogu procijeniti kao ispravni zbog izostanka grešaka.

Zamjerke se odnose na moguća poboljšanja sučelja za prijavu gdje ne postoje nikakve povratne informacije u slučaju neispravnih podataka za prijavu. Također se može napomenuti kako trenutna količina testova nije dovoljna za detaljno testiranje sustava te se ne provede testovi za neke ključne funkcionalnosti poput stvaranja korisnika, skeniranja i digitalizacije dokumenata te prosljeđivanja. Glavni razlog ne provođenja veće količine testova je vrijeme, a može se spomenuti i problematika testiranja funkcionalnosti koje se oslanjaju na upotrebljene pakete za čiju provjera bi valjalo detaljnije proučiti implementaciju samih paketa.

5.3 Dijagram razmještaja

Dijagrami razmještaja opisuju topologiju sklopovlja i programsku potporu sustava te njegovo radno okruženje. Klijenti koriste android mobilni uređaj kako bi pristupili aplikaciji. Aplikacija zatim s pomoću https protokola komunicira s FastAPI aplikacijom koja se nalazi na virtualnom stroju na *Render* cloudu. Ovisno o potrebi, FastAPI aplikacija šalje https zahtjeve na *Azure* cloud gdje se nalaze baza podataka, pohrana slika te servis za prepoznavanje teksta.



Slika 5.2: Dijagram razmještaja

5.4 Upute za puštanje u pogon

U narednim potpoglavljima objašnjeno je kako pokrenuti sve dijelove aplikacije lokalno ili postavljanjem na besplatne servise.

5.4.1 Puštanje poslužitelja u pogon

U narednim potpoglavljima objašnjene se opcije i postupak postavljanja poslužitelja.

Dodavanje varijabli okruženja (opcionalno)

Varijable okruženja se prilažu unutar .env datoteke. Potrebno je stvoriti datoteku s imenom '.env' unutar glavnog direktorija projekta i unutar njega se stavljaju vrijednosti koje se žele postavljati s pomoću varijabli okruženja, ovo su varijable koje je moguće postavljati.

- database_url
- secret_key
- access_token_expire_hours
- DefaultEndpointsProtocol
- AccountName
- AccountKey
- EndpointSuffix
- image_path
- vision_key
- vision_endpoint

Konfiguracija vanjske baze podataka (opcionalno)

Za potrebe backend dijela aplikacije, potrebno je imati bazu podataka. Osim ako se kroz varijable okruženja ne postavi drugačije, aplikacija sama stvara svoju SQLite bazu podataka u svom direktoriju. Ako se želi koristiti vanjska baza podataka, potrebno je konfigurirati baza podataka.

PostgreSQL bazu podataka je moguće konfigurirati na više lokacija, jedna od kojih je Azure cloud[16]. Nakon postavljanja, potrebno je napraviti link za konekciju ovog oblika: 'postgresql://username:password@host:port/postgres'. Primjer tog linka bi bio: 'postgresql://username:L216Yq6ber6Rma9@digitalizacija-api-database.postgres.datab
Nakon kreiranja linka potrebno ga je staviti u varijablu okruženja s ključem 'database_url'.

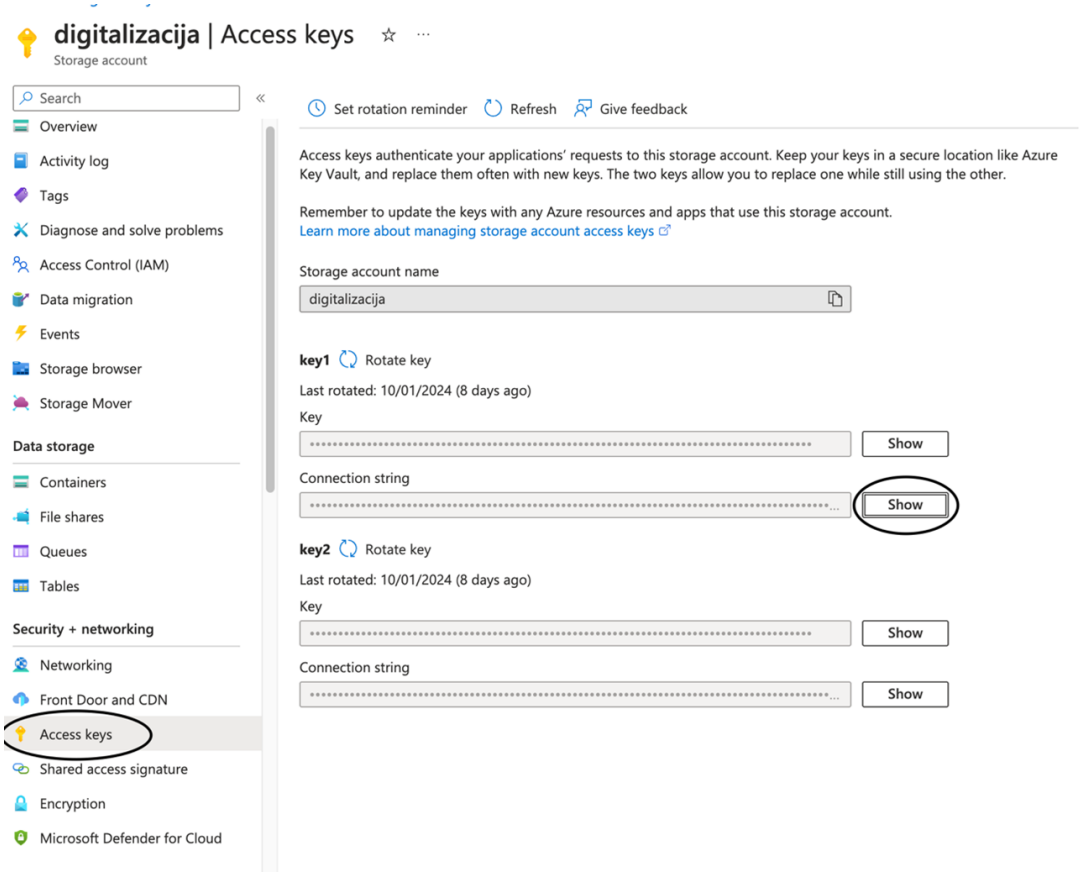
Konfiguracija vanjskog spremnika slika (opcionalno)

Bez zasebne konfiguracije, slike se spremaju u direktorij aplikacije unutar images pod direktorija koji je automatski kreiran. Taj direktorij se može promijeniti dodavanje 'image_path' varijablom okruženja čime se određuje relativna putanja gdje će se spremati slike. Također je moguća konfiguracija Azure blob storage[17].

Nakon konfiguracije potrebno je pronaći Connection string koji se nalazi unutar 'Access keys', kao što se vidi na slici. Potrebno je stisnuti show te kopirati taj string.

[16] https://learn.microsoft.com/en-us/azure/postgresql/single-server/quickstart-create-server-database-portal

[17] https://learn.microsoft.com/en-us/azure/storage/blobs/storage-quickstart-blobs-portal



Slika 5.3: Primjer pronalaska ključa

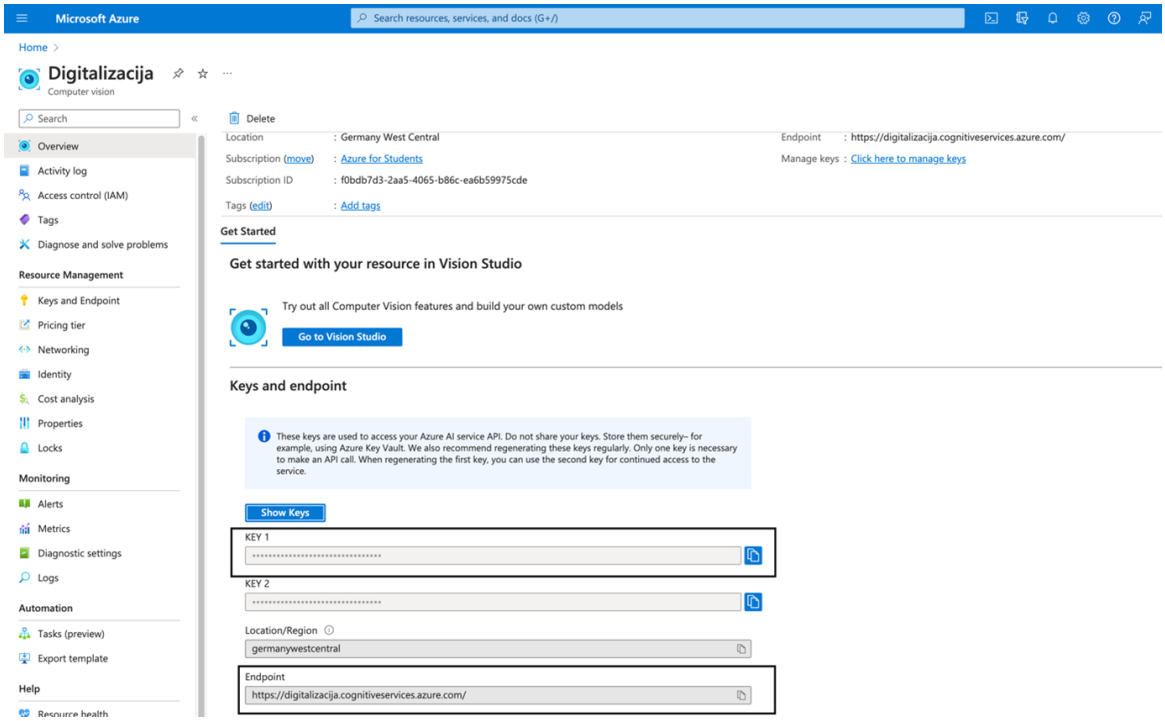
Taj string je potrebno postaviti unutar varijabli okruženje rastavljen na ove vrijednosti: 'DefaultEndpointsProtocol', 'AccountName', 'AccountKey' te 'EndpointSuffix'. Nakon toga, ime specifičnog kontejnera gdje će se spremati slike se postavi s 'image_path' varijablom okruženja ili će biti images ako se ne postavi.

Konfiguracija vanjskog prepoznavanja slova (opcionalno)

Bez dodatne konfiguracije, aplikacija ima ugrađeno prepoznavanje teksta na slici ali nije preporučeno korištenje ugrađenog prepoznavanja na svim uređajima. Prepoznavanje na uređajima je spor i vrlo resursno intenzivan proces. Za najbolje rezultate prepoznavanja, potrebno je imati nVIDIA grafičku karticu 2GB video memorije ili barem 3GB slobodnog RAM-a u slučaju da ne postoji nVIDIA grafička kartica. Nije preporučeno pokretanje lokalnog prepoznavanja slova ako nije dostupna grafička kartica jer će proces biti iznimno dug. Zato je preporučeno postaviti computer vision servis na Azure cloudu[18].

[18] `https://portal.azure.com/#create/Microsoft.CognitiveServicesComputerVision`

Nakon kreiranja resursa, potrebno je pronaći endpoint i ključ za vision api. Oni se nalaze na overview stranici Computer Vision dashboarda na Azure cloudu. Na slici se vidi gdje se oni nalaze.



Slika 5.4: Primjer pronalaska ključa za vision api

Potrebno je kopirat te dvije vrijednosti i staviti ih u varijable okruženja pod ključevima 'vision_key' i 'vision_endpoint'. Kada se to postavi, konfigurirano je prepoznavanje teksta na oblaku.

Pokretanje backend aplikacije

Kako bi se pokrenula aplikacija potrebno je pratiti ove korake:

1. Potrebno je instalirati python 3.11[19]
2. Potrebno je klonirati repozitorij lokalno: 'git clone https://github.com/Jura-Hostic-i-Film/Jura-Hostic-i-Film-api.git'
3. Potrebno je instalirati potrebne pakete unutar direktorija aplikacije: 'pip install -r requirements.txt'

[19] https://www.python.org/downloads/

4. (Opcionalno) dodavanje varijabli okruženja
5. Pokretanje servera naredbom: 'uvicorn app.main:app –reload '
6. Unutar terminala je prikazan link kojim se pristupa API-u, dodavanje '/docs' na kraj linka se pristupa dokumentaciji

5.4.2 Puštanje mobilne aplikacije u pogon

U naredna dva potpoglavlja dane su opcije pokretanja i izgradnje Flutter mobilne aplikacije te mogućnost objave.

Instalacija potrebnih alata za pokretanje i lokalna izgradnja

U nastavku je dan niz naredbi za lokalno preuzimanje i pokretanje aplikacije.

1. Namjestiti Flutter razvojni okvir i razvojno okruženje po želji koristeći službene upute[20]
2. Klonirati repozitorij s izvornim kodom lokalno s pomoću Git naredbi
3. Unutar src direktorija pokrenuti sljedeće naredbe

```
flutter pub get
flutter build apk --dart-define=APP_NAME=Digitalizacija
--dart-define=BASE_URL=jura-hostic-i-film-api.onrender.com --release
```

Dobiveni apk paket ili onaj preuzet s repozitorija može se instalirati na emulatoru Android uređaja ili osobnom mobilnom uređaju s Android operacijskim sustavom.

Objava aplikacije na F-Droid platformi

F-Droid je platforma za objavljivanje besplatnih mobilnih aplikacija s javno dostupnim izvornim kodom. Omogućava postavljanje vlastitih aplikacija predajom podataka za izgradnju iz javnog Git repozitorija i dijeljenja s pomoću metapodataka.

U nastavku je dan niz uputa koje valja slijediti ako se aplikacija želi objaviti putem F-Droid platforme.

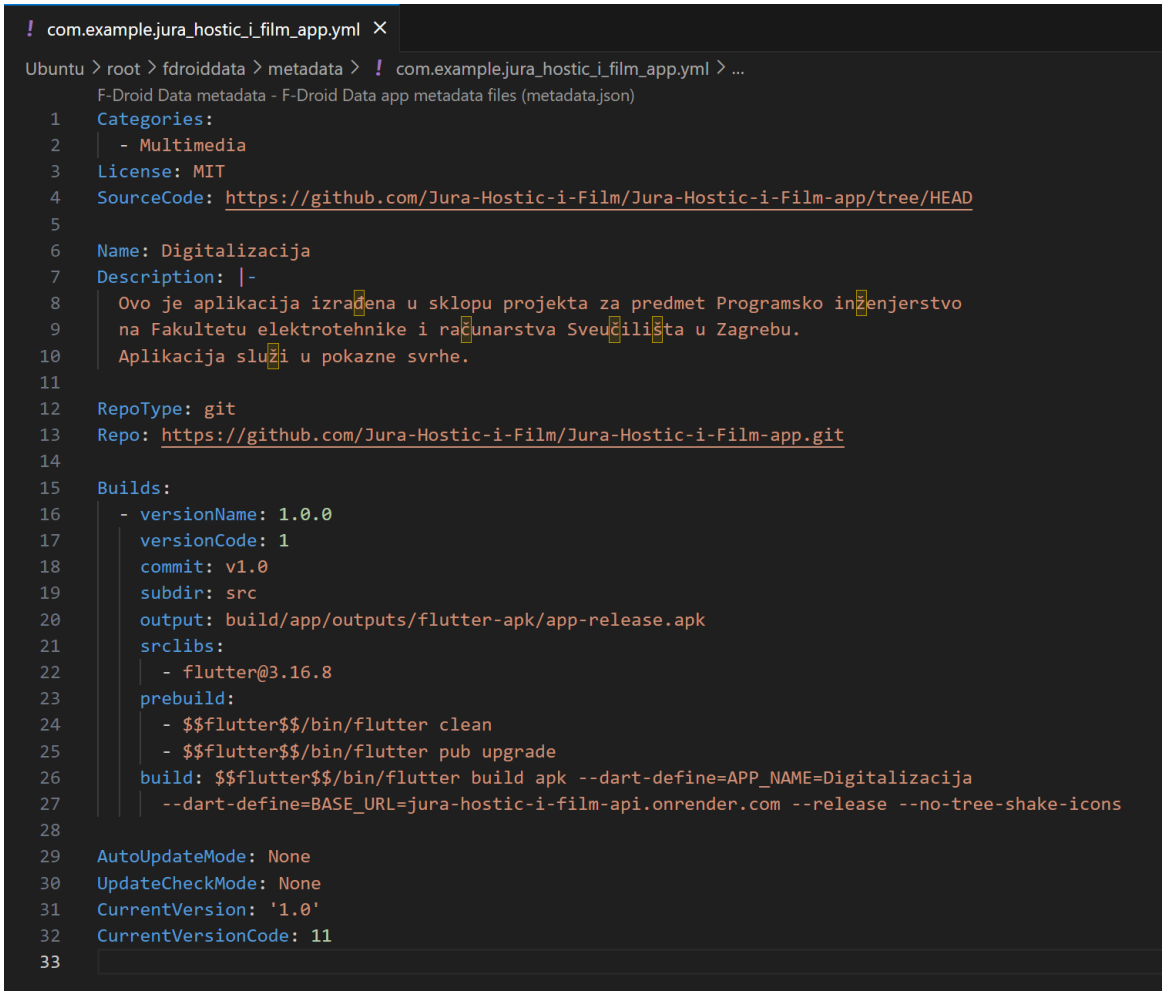
[20] https://docs.flutter.dev/get-started/editor

1. Izraditi GitLab[21] račun
2. Izvršiti git fork nad fdroiddata[22] repozitorijem
3. Daljnje korake izvršiti u sklopu Linux okvira, za Windows operacijski sustav preporuča se upotreba WSL 2[23] alata
4. Preuzeti stvoreni repozitorij

```
git clone --depth=1 https://gitlab.com/IME_RACUNA/fdroiddata ~/fdroiddata
cd ~/fdroiddata
git checkout -b com.example
cp templates/build-flutter.yml metadata/com.example.jura_hostic_i_film_app.yml
```

5. Urediti com.example.jura_hostic_i_film_app.yml metadata datoteku po želji, preporuča se sljedeća konfiguracija

[21] https://about.gitlab.com/
[22] https://gitlab.com/fdroid/fdroiddata
[23] https://learn.microsoft.com/en-us/windows/wsl/install



Slika 5.5: Prikaz postavki u metadata datoteci

6. Preuzeti zadnju inačicu kontejnera sa serverskim alatima, na Windows operacijskom sustavu dodatno treba instalirati Docker[24] alat i omogućiti uporabu unutar WSL 2 okruženja

```
git clone --depth=1 https://gitlab.com/fdroid/fdroidserver \
~/fdroidserver
sudo sh -c 'apt-get update &&apt-get install -y docker.io'
sudo docker run --rm -itu root --entrypoint /bin/bash \
-v ~/fdroiddata:/build:z \
-v ~/fdroidserver:/home/vagrant/fdroidserver:Z \
registry.gitlab.com/fdroid/fdroidserver:buildserver
```

[24] https://docs.docker.com/desktop/install/windows-install/

7. Unutar kontejnera pokrenuti sljedeći niz naredbi

```
. /etc/profile
export PATH="$fdroidserver:$PATH" PYTHONPATH="$fdroidserver"
export JAVA_HOME=$(java -XshowSettings:properties -version 2>&1 > \
/dev/null | grep 'java.home' | awk -F'=' '{print $2}' | tr -d ' ')
cd /build
fdroid readmeta
fdroid rewritemeta com.example.jura_hostic_i_film_app
fdroid checkupdates --allow-dirty com.example.jura_hostic_i_film_app
fdroid lint com.example.jura_hostic_i_film_app
fdroid build com.example.jura_hostic_i_film_app
```

8. Nakon uspješne izgradnje paketa izvršiti sljedeći niz naredbi

```
exit
cd ~/fdroiddata
git add metadata/com.example.jura_hostic_i_film_app.yml
git commit -m "New App: com.example.jura_hostic_i_film_app"
git push origin com.example
```

9. Na kraju otvoriti zahtjev za spajanje vlastite grane na fdroiddata repozitorij

10. F-Droid tim može imati dodatnih pitanja u slučaju krivo provedenog postupa, stoga je bitno ostati na raspolaganju

Prilikom izrade zahtjeva za objavu važno je pratiti konvencije kojih se pridržava F-Droid platforma, koje se zajedno sa svim dodatnim informacijama mogu naći na stranicama platforme. [25]

U slučaju ispravne predaje zahtjev za objavu se obrađuje, F-Droid tim vrši verifikaciju te se aplikacija objavljuje u najkraćem mogućem vremenu.

[25] https://f-droid.org/docs/Submitting_to_F-Droid_Quick_Start_Guide/

6. Zaključak i budući rad

Projektni zadatak bio je razviti prilagođeno rješenje za digitalizaciju i barataNje digitaliziranim dokumentima. Glavne ciljne značajke bile su mogućnost digitalizacije određenih formata dokumenata, razvrstavanje u definirane kategorije, mogućnost rada sa spremljenim dokumentima na razini različitih zaposlenika, u što ulazi revidiranje, arhiviranje i potpisivanje te slanje dokumenata putem društvenih mreža. Dodatno se podrazumijeva sustav upravljanja korisnicima . Sve glavne značajke uspješno su implementirane te je provedeno testiranje pojedinih komponenti i testiranje sustava.

Na projektnom zadatku radio je studentski tim u efektivnom razdoblju od deset radnih tjedana. Razvojno razdoblje se može sagledati u dva jednaka dijela. Prvo dio odnosio se prvenstveno na proučavanje i dokumentiranje zahtjeva te osmišljanje, razradu i modeliranje rješenja. Cilj je bio izraditi stabilnu bazu na kojoj se dalje moglo raditi u sklopu projekta te sagledati moguće buduće probleme. U sklopu prvog dijela projekta fokus većine tima bio je na proučavanju i izradi valjane dokumentacije te savladavanju osnova odabranih tehnologija. Osnovne funkcionalnosti idejnog rješenja također su implementirane u sklopu prvog dijela. Glavni cilj drugog dijela razvojnog razdoblja bila je puna implementacija idejnog rješenja, dorada izvedenih značajki, testiranje te finalizacija dokumentacije. U drugom dijelu članovi su paralelno nastavili učiti odabrane tehnologije i koristili naučeno pri izradi rješenja. Tijekom rada nisu zabilježeni veći problemi u izradi i implementaciji, niti se razvojni tim susreo s problemima koje nisu mogli riješiti. Kao veći problem može se spomenuti nenadan nedostatak resursa odabranog okruženja na koje je postavljen poslužiteljski dio aplikacije zbog čega se nije moglo koristiti ostvareno OCR rješenje, no tim je efektivno to riješio povezivanje s vanjskim servisom za vršenje OCR procesa nad kojim se zatim vrše dodatne operacije za prilagodbu.

S obzirom na to da je razvojni tim uložio mnogo vremena i truda u razradu svih dijelova projekta smatramo kako ne bi bilo problema u slučaju nastavka rada na projektu te bi produktivnost bila na izrazito visokoj razini zbog svega naučenog. Također smatramo kako je projekt u cjelini vrlo dobro postavljen s dovoljno opširnom dokumentacijom što bi omogućilo laku i efektivnu primopredaju drugom razvoj-

nom timu radi nastavka razvoja i dorade rješenja.

Razvojni tim smatra kako je vrlo dobro iskoristio svoje vrijeme pri radu na projektu te kako su svi članovi tima imali priliku naučiti nešto novo i razviti se kao budući inženjeri računarstva. Rad na projektu pridonio je poboljšanju komunikacijskih i organizacijskih vještina kod članova, stavio ih u nove situacije gdje su morali naučiti koristiti nove tehnologije, kako za sam razvoj aplikacije već i za izradu dokumentacije te organizaciju razvojnog tima. Razvojni tim smatra kako je projekt završio uspješno te je zadovoljan postignutim rezultatom.

Popis literature

Kontinuirano osvježavanje
Popisati sve reference i literaturu koja je pomogla pri ostvarivanju projekta.

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. Fastapi Tutorial, https://fastapi.tiangolo.com/tutorial/
8. SQLAlchemy Documentation, https://docs.sqlalchemy.org/
9. Microsoft Azure Documentation, https://learn.microsoft.com/en-us/azure/
10. Flutter Documentation, https://docs.flutter.dev/

Indeks slika i dijagrama

Dodatak: Prikaz aktivnosti grupe

Dnevnik sastajanja

1. sastanak
    - Datum: 13. listopad 2023.
    - Prisustvovali: Tvrtko Puškarić, Luka Bradarić Lisić, Karlo Grgičin, Jura Hostić, Andrea Milanović, Katarina Pešić, Martin Subotić
    - Teme sastanka:
        - Početni dogovori vezano za odabir teme projekta
        - Upoznavanje s kolegama
        - Pregled predložene teme
        - Razrada novo-predložene teme
2. sastanak
    - Datum: 28. listopad 2023.
    - Prisustvovali: Tvrtko Puškarić, Luka Bradarić Lisić, Karlo Grgičin, Jura Hostić, Andrea Milanović, Katarina Pešić, Martin Subotić
    - Teme sastanka:
        - Odabir konačne teme
        - Razrada odabranog projekta
        - Odabir platforme za izradu aplikacije
        - Raspodjela zadataka
        - Upoznavanje s LaTex dokumentacijom
3. sastanak
    - Datum: 5.studenoga 2023.
    - Prisustvovali: Tvrtko Puškarić, Luka Bradarić Lisić, Karlo Grgičin, Jura Hostić, Andrea Milanović, Katarina Pešić, Martin Subotić
    - Teme sastanka:
        - Stvoren početni predložak dokumentacije
        - Upoznavanje kolega zaduženih za mobilnu aplikacijom s Flutter razvojnim okvirom

  - Upoznavanje kolega zaduženih za pozadinski server s FastAPI razvojnim okvirom
  - Održana interna radionica upotrebe Git alata
  - Raspodjela zadataka za izradu dokumentacije

4. sastanak
   - Datum: 8.studenoga 2023.
   - Prisustvovali: Tvrtko Puškarić, Luka Bradarić Lisić, Karlo Grgičin, Jura Hostić, Andrea Milanović, Katarina Pešić, Martin Subotić
   - Teme sastanka:
     - Početak izrade dokumentacije
     - Početak izrade mobline aplikacije
     - Početak izrade pozadinskog servera
     - Pregled modela baze podataka i njezino implementiranje
5. sastanak
   - Datum: 13.studenoga 2023.
   - Prisustvovali: Tvrtko Puškarić, Luka Bradarić Lisić, Karlo Grgičin, Jura Hostić, Andrea Milanović, Katarina Pešić, Martin Subotić
   - Teme sastanka:
     - Interne diskusije oko statusa dokumentacije
     - Diskusije oko statusa usvajanja znanja odabranih tehnologija
6. sastanak
   - Datum: 6.prosinca 2023.
   - Prisustvovali: Tvrtko Puškarić, Karlo Grgičin, Jura Hostić, Andrea Milanović, Katarina Pešić
   - Teme sastanka:
     - Raspodjela zadataka frontenta i diskusija
7. sastanak
   - Datum: 11.prosinca 2023.
   - Prisustvovali: Karlo Grgičin, Jura Hostić, Andrea Milanović, Katarina Pešić
   - Teme sastanka:
     - Učenje FastAPI-a
     - Početna podjela backend zadataka
     - Diskusija backenda
8. sastanak

- Datum: 23.prosinca 2023.
- Prisustvovali: Tvrtko Puškarić, Luka Bradarić Lisić, Martin Subotić
- Teme sastanka:
  - Raspodjela zadataka frontenta
  - Diskusija raspodjele frontenda

9. sastanak
   - Datum: 8.siječnja 2024.
   - Prisustvovali: Karlo Grgičin, Jura Hostić, Andrea Milanović, Katarina Pešić
   - Teme sastanka:
     - Rješavanje bugova
     - Podjela preostalih zadataka
10. sastanak
    - Datum: 13.siječnja 2024.
    - Prisustvovali: Tvrtko Puškarić, Luka Bradarić Lisić, Jura Hostić, Andrea Milanović, Katarina Pešić, Martin Subotić
    - Teme sastanka:
      - Status zadnjih zadataka
      - Lista preostalih zadataka
      - Podjela dokumentacije
11. sastanak
    - Datum: 14.siječnja 2024.
    - Prisustvovali: Tvrtko Puškarić, Luka Bradarić Lisić, Martin Subotić
    - Teme sastanka:
      - Diskusija dovršavanja frontenda

Tablica aktivnosti

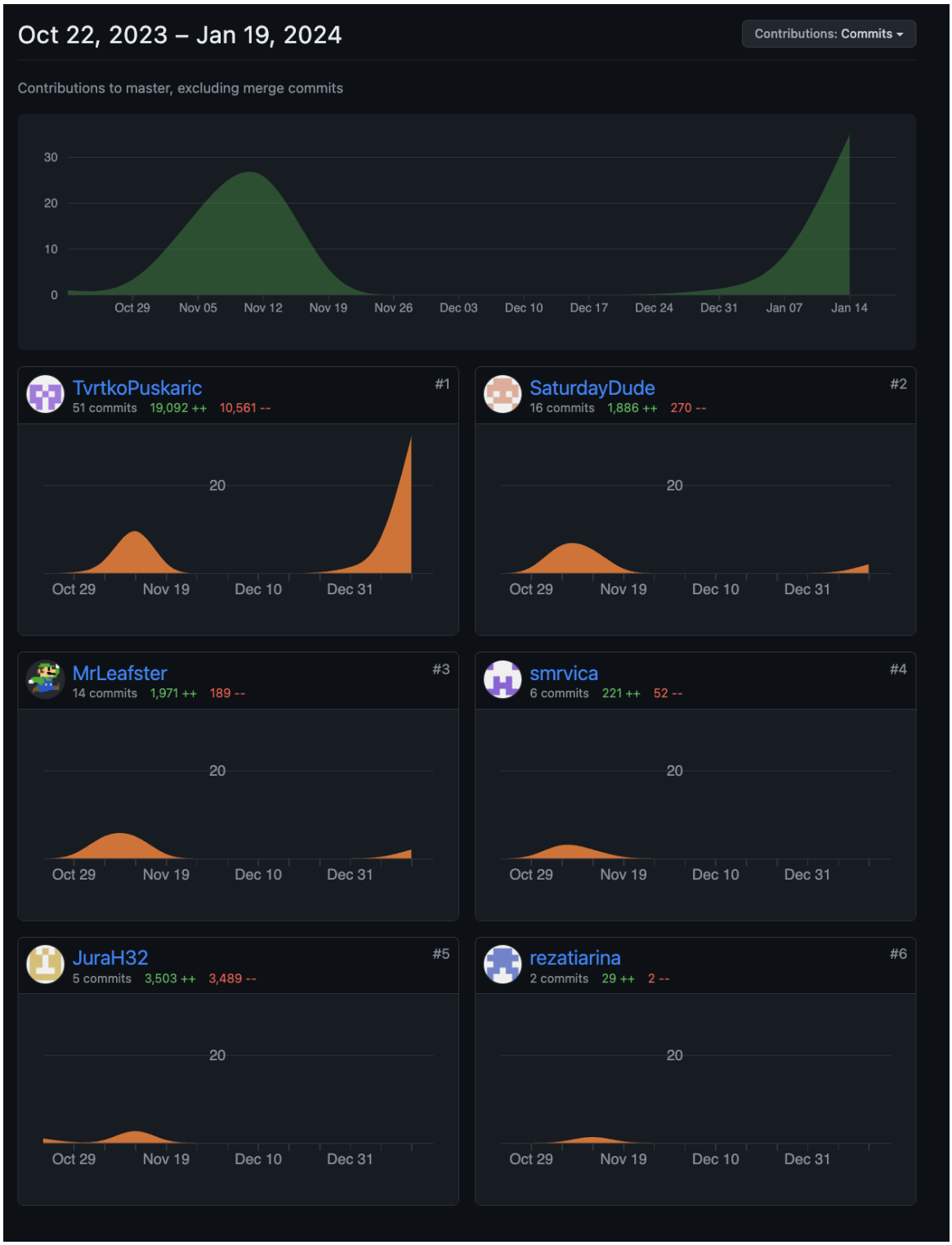
	Tvrtko Puškarić	Luka Bradarić Lisić	Karlo Grgičin	Jura Hostić	Andrea Milanović	Katarina Pešić	Martin Subotić
Upravljanje projektom	9	2		8			
Postavljanje projekta na udaljeni repozitorij	3	4		3			
Opis projektnog zadatka	5						
Funkcionalni zahtjevi		3					2
Opis pojedinih obrazaca		4					5
Dijagram obrazaca							3
Sekvencijski dijagrami						6	
Opis ostalih zahtjeva							1
Arhitektura i dizajn sustava		2		3	6		
Baza podataka			4	5	3		
Dijagram razreda			6	6			
Rad na mobilnoj aplikaciji	64	26					28
Rad na API rješenju			26	55	29	28	
Postavljanje rješenja na server				4			
Uvod u Flutter	1	1	1	1	1	1	1
Uvod u razvoj API servisa	1	1	1	1	1	1	1
Git radionica			3	3	3	3	3
Učenje razvojnog okvira Flutter		34					35

Nastavljeno na idućoj stranici

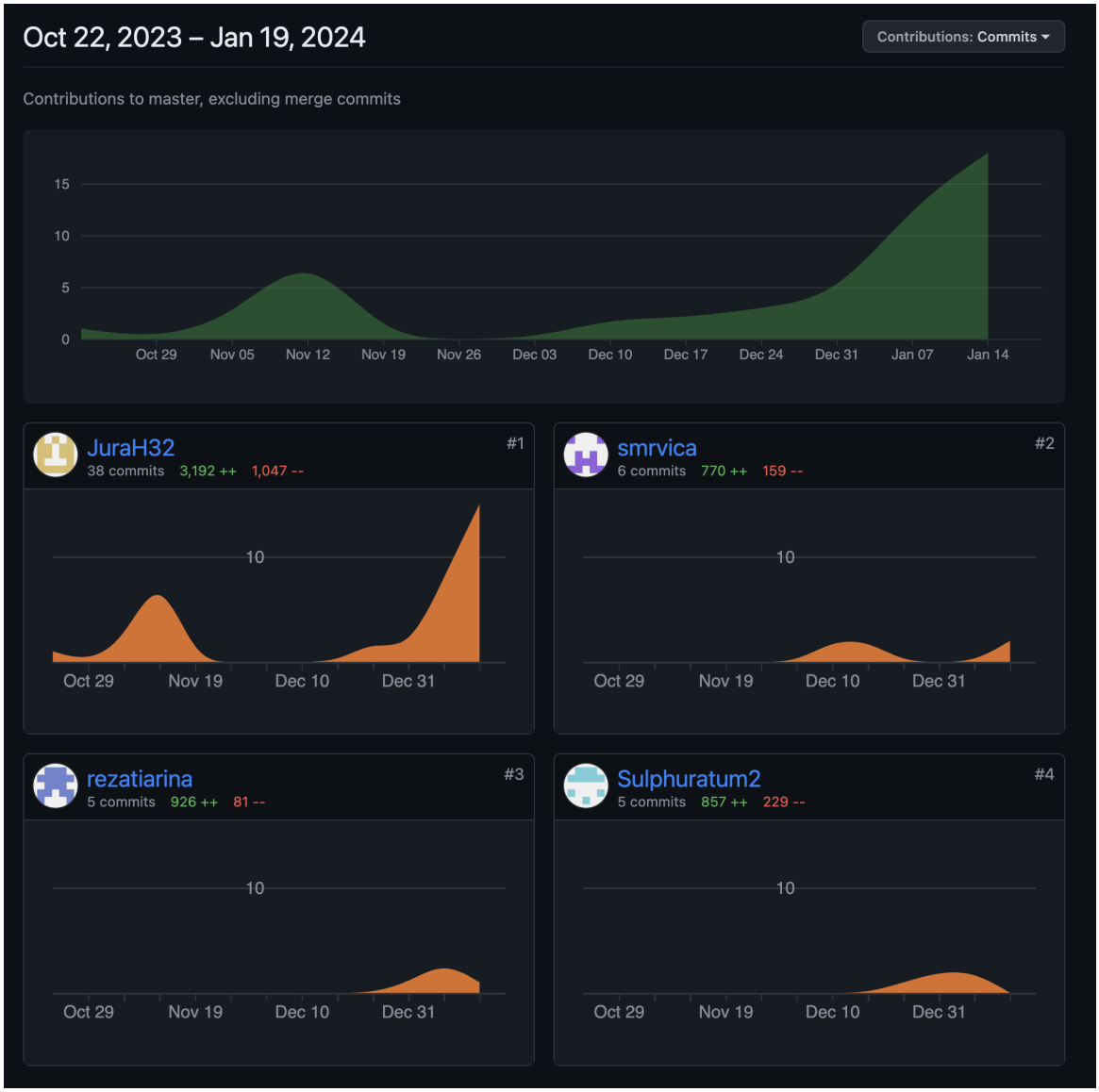
Nastavljeno od prethodne stranice

	Tvrtko Puškarić	**Luka Bradarić Lisić**	**Karlo Grgičin**	**Jura Hostić**	**Andrea Milanović**	**Katarina Pešić**	**Martin Subotić**
Učenje razvojnog okvira FastAPI			30	6	27	33	
Grupne diskusije i sastanci	9	9	9	9	9	9	9
Dijagram stanja		8					
Dijagram aktivnosti					5		5
Dijagram komponenti		3	1	1		3	
Korištene tehnologije i alati				2			
Ispitivanje programskog rješenja	5			7			
Dijagram razmještaja				1			
Upute za puštanje u pogon	1		1				
Dnevnik sastajanja	1			1			
Zaključak i budući rad	3						
Popis literature	1			1			
Postavljanje aplikacije na F-Droid	6						
Postavljanje azure servisa				4			

Dijagrami pregleda promjena



Slika 6.1: Dijagram promjena aplikacije i dokumentacije



Slika 6.2: Dijagram promjena backenda