

ProgInz TG08.3 Home .....	.2
Službena Dokumentacija .....	3
Home .....	4
1. Opis projektnog zadatka .....	5
2. Analiza zahtjeva .....	8
3. Specifikacija zahtjeva sustava .....	14
4. Arhitektura i dizajn sustava .....	32
5. Arhitektura komponenata i razmještaja .....	47
6. Ispitivanje programskog rješenja .....	49
7. Tehnologije za implementaciju aplikacije .....	61
8. Upute za puštanje u pogon .....	62
9. Zaključak i budući rad .....	65
A. Popis literature .....	66
A. Dnevnik promjena dokumentacije .....	67
B. Prikaz aktivnosti grupe .....	74
Meetings .....	78
Meeting 29.10.2024 .....	79
Meeting 28.10.2024 .....	80
Meeting 25.10.2024 .....	85
Meeting 23.10.2024 .....	86
Meeting 22.10.2024 .....	88
Meeting 18.10.2024 .....	90
Meeting 13.10.2024 .....	91
Meeting 10.10.2024 .....	92
Meeting 28.11.2024 .....	93
BE - Meeting 30.11.2024 .....	95
BE - Meeting 9.12.2024 .....	96
Meeting 23.12.2024 .....	97

## ProgInz TG08.3 Home

Site location: [Home Page - Duck!](#)

- Filip Belina - Project Lead
- Jakov Lovaković - Backend Team Lead
- Jan Lalić - Backend Engineer
- Leo Marušić - DevOps/Database Engineer
- Mislav Marinović - Lead design
- Jan Badel - Frontend Engineer
- Martin Šainčević - Frontend Team Lead
  
- Frontend - Razor, Bootstrap
- Backend - ASP.NET
- Database - SQLite
- Project Management - JIRA, Confluence



# Službena Dokumentacija

Home

## **Programsko inženjerstvo ak.god 2024./2025**

**Sveučilište u Zagrebu**

**Fakultet elektrotehnike i računarstva**

### **DuckI**

Tim: <TG08.3>

Gumene Patkice

Nastavnik: Vlado Sruk

# 1. Opis projektnog zadatka

## Funkcionalnost

Kao projektni zadatak izvodit će se aplikacija za pomoć pri učenju u obliku AI asistenta. Koristeći materijale tekstualnih oblika u PDF formatu, AI asistent bi kreirao brojan sadržaj s kojim bi korisnik mogao bolje shvatiti određenu temu uz osjećaj personaliziranog podučavanja. Uz studente aplikaciju bi koristili i edukatori koji bi imali mogućnost objavljivanja svog sadržaja, resursa i materijala, također u PDF formatu, koje može koristiti student ili ih AI asistent može koristiti kao izvor podataka. Nad edukatorima nadgledaju i recenzenti (engl. Reviewers) čija je uloga provjeravati točnost sadržaja koji su dodali edukatori i po potrebi ih izbrisati. Administratori imaju mogućnosti dodavanje oznaka u sustav koje se koriste za označavanje PDF dokumenata (engl. Tags) te dodjeljivanje uloga korisniku i brisanje korisnika iz sustava.

Studentu aplikacija služi za spremište i upravljanje svojim materijalima. Kada će student htjeti interakciju s AI asistentom, generirali bi se resursi u obliku raznih flash kartica što bi studentu pomoglo pri savladavanju određenih tema. Kao izvore podataka bi se koristili upravo materijali koji su spremljeni na aplikaciju, tako da se svakome studentu može kreirati točno taj sadržaj koji njemu treba.

AI asistent je izведен pomoću Google Gemini 1.5 Flash modela. Model će primati dokumente (PDF format), te na temelju njih stvoriti sadržaj, zadatke i ostale resurse koji bi pomogli studentu. Student sam bira dokumente iz kojih želi generirati sadržaj.

Uz studenta, aplikacijom se mogu koristiti i edukatori koji bi imali mogućnost kreiranja i dodavanja svojih materijala za koje oni smatraju da bi mogli pomoći studentu, te student može spremiti dodatne materijale što bi omogućilo AI asistentu da se služi s tim materijalima pri generiranju sadržaja.

Materijale edukatora kontroliraju recenzenti. Recenzenti mogu izbrisati javni materijal koji smatraju da ne zadovoljava zahtjeve kvalitete. Pri brisanju Recenzenti ostavljaju povratnu poruku kako bi edukator koji je vlasnik materijala bio upućen u razloge brisanja.

Administratori nadgledaju rad korisnika u sustavu i dodavaju oznake za PDF dokumente. Administrator je zadužen za dodavanje uloge korisniku te za brisanje korisnika ako smatra da njihovo ponašanje ugrožava sustav.

Svi korisnici aplikacije se moraju prijaviti i potvrditi svoj identitet, te je u tu svrhu korištena OAuth2 autentifikacija. Koristit će se principi responsivnog dizajna za dinamičku prilagodbu korisničkog sučelja različitim veličinama i razlučivostima zaslona.

Aplikacija također sadrži kalendar preko kojega korisnici mogu upravljati svojim rasporedom sati. Mogu staviti svoj predefinirani raspored iz .csv datoteke ili izraditi svoj klikanjem na željene datume.

Svim korisnicima koji imaju neku od uloga sustava (Administrator, Edukator, Recenzent, Student) dozvoljeno je upravljati personaliziranim kalendarom.

Studenti mogu ostaviti recenzije o materijalima. Te recenzije studenti i ostali korisnici mogu koristiti u svrhu procijenjivanja vjerodostojnosti materijala.

## 1. Svrha i ciljevi:

Ovaj projekt radi na razvoju aplikacije s AI asistentom pri učenju. Ovo bi bio asistent za učenje vođen umjetnom inteligencijom koji personalizira proces učenja za učenike putem prilagođenih obrazovnih resursa u obliku flash kartica temeljenih na materijalu koji će studenti učitati s dodatnim resursima edukatora ili svojim vlastitim.

Aplikacija će nastavnicima dati platformu za objavljivanje dodatnog materijala za učenje koji učenici mogu koristiti za dublje proučavanje tema. Kako bi osigurali kvalitetu i integritet obrazovanja, recenzenti i administratori će moderirati aktivnosti edukatora i ostalih korisnika. Aplikacija koristi OAuth2 autentifikaciju s responsivnim dizajnom kako bi

pružila iskustva na uređajima na siguran, pristupačan i prilagodljiv način.

## 2. Moguće koristi:

Koristi su najbolje vidljive za dvije skupine ljudi:

Studenti imaju koristi od prilagođenih pomagala za učenje koja zadovoljavaju individualne potrebe u učenju, što poboljšava razumijevanje i prisjećanje naučenoga. Sposobnost kreiranja personaliziranog sadržaja u aplikaciji pomaže premostiti te nedostatke u razumijevanju i omogućuje učeniku da se koncentriра na ona područja koja trebaju više pozornosti.

Edukatori će moći dijeliti materijale koji mogu pomoći učenicima da bolje razumiju određene koncepte. Oni mogu učinkovitije olakšati učenje prikladnim primjerima, vježbama i vodičima za učenje.

## 3. Povezani radovi i razlike:

Iako već postoje slična rješenja za obrazovne svrhe, nema puno proizvoda koji kombiniraju personaliziranog asistenta uz pomoć umjetne inteligencije sa stvaranjem sadržaja u stvarnom vremenu s opcijom postavljanja globalnih materijala za edukatore. Slični koncepti na tržištu: Khan Academy: Tečajevi s modulima, ali ne bilo kakvim; dinamičkim, AI generiranim resursima za učenje koji se temelje na materijalima koje su učitali određeni studenti. Quizlet: Flash kartice i kvizovi, ali oni se moraju unijeti ručno; ovaj projekt automatski generira pomoćna sredstva za učenje. Coursera/EdX: Vrijednost ovih platformi leži u strukturiranim tečajevima koje edukatori stvaraju. Ne postoji interakcija uživo s kojom bi AI prilagodio sadržaj na temelju datoteka koje su poslali korisnici. MindGrasp.ai: Najsličniji proizvod, na bazi AI materijale pretvara u notes, flashcards, i tasks, no ne postoji nikakva interakcija ili sistem s edukatorom, što bi činilo ovo teškim za organizirati u razrednom okruženju.

## 4. Ciljane korisničke skupine:

Ciljane skupine korisnika koje ova aplikacija namjerava obuhvatiti su:

Studenti: Primarni korisnici, traže personaliziranu poduku i pomoć pri učenju na zahtjev.

Edukatori: Sekundarni korisnici koji mogu proširiti iskustvo učenja dijeljenjem kvalitetnih obrazovnih resursa.

## 5. Prilagodba i prilagodljivost:

Ova je aplikacija namijenjena jednostavnoj prilagodljivosti različitim obrazovnim okruženjima. Sadržaj vođen umjetnom inteligencijom, personaliziran za svakog učenika, može obuhvaćati širok raspon predmeta i metodologija podučavanja/učenja.

## 6. Opseg projekta:

Funkcionalnost aplikacije bit će višestruka:

Generiranje sadržaja s pomoću umjetne inteligencije: To omogućuje stvaranje resursa za učenje iz PDF-ova korištenjem modela Google Gemini 1.5 Flash.

Autentifikacija korisnika: Omogućuje funkcionalnost provjere autentičnosti korisnika na siguran način korištenjem OAuth2 za održavanje privatnosti osobnih podataka.

Kontrola kvalitete/Pregled i odobrenje sadržaja: Pomaže recenzentima i administratorima u praćenju kvalitete materijala i aktivnosti korisnika.

Mehanizam povratnih informacija korisnika: Mehanizam za bilježenje i integraciju studentskih recenzija kako bi se osigurala kvaliteta materijala.

Interaktivni Kalendar: služi za laganu kontrolu upravljanja rasporedom.

## 7. Budućnost projekta

Ušli smo u ovaj projekt s nadom izrade aplikacije koju bi i sami koristili na dnevnoj bazi. Prva stvar koju bi voljeli je koristiti aplikaciju i sami kako bi lakše polagali kolegije na FER-u, te nam je želja ovo nakon predaje pretvoriti u open source repozitorij koji bilo tko besplatno može upogoniti i sam koristiti aplikaciju lokalno (naravno bez dodataka dodanih za ispunjavanje kriterija predmeta)

## 2. Analiza zahtjeva

### 2.1 Dionici

U okružju aplikacije, dionici će se sastojati od:

- Vlasnik sustava (naručitelj)
- Studenti
- Edukatori (predavači, instruktori, profesori, učitelji)
- Recenzenti (engl. Reviewers)
- Administratori
- Razvojni tim

Aktori, to jest dionici će imati zasebne uloge i mogućnosti, te će svaki moći raditi i obavljati iduće funkcionalnosti:

#### 1. A-1 Student (inicijator)

Student koristi aplikaciju za organizaciju učenja, pregled materijala i dodavanje vlastitih sadržaja.

Student može:

- Uploadati vlastiti sadržaj u obliku PDF-a
  - F-002: Prijenos PDF datoteka s vlastitim sadržajem
- Generirati "flashcards"-e iz sadržaja u aplikaciji
  - F-003: flashcards-a pomoću AI asistenta
- Uploadati kalendar u obliku CSV datoteke
  - F-008: Prijenos kalendara u CSV formatu
- Dodavati datume u kalendar unutar aplikacije
  - F-006: Upravljanje osobnim kalendarom u sustavu
- Davati recenzije javno dostupnim materijalima
  - F-007: Ostavljanje recenzija na javne materijale
- Brisati vlastiti sadržaj sa sustava
  - F-009: Uklanjanje vlastitih datoteka i unosa
- Pregledavanje PDF datoteka
  - F-014: Student može pregledati javne i svoje osobne materijale

#### 2. A-2 Edukator (inicijator)

Edukator koristi aplikaciju za dijeljenje obrazovnih materijala i upravljanje vlastitim sadržajem.

Edukator može:

- Uploadati javno dostupne obrazovne materijale
  - F-004: Prijenos materijala koji su javno dostupni
- Brisati vlastiti sadržaj sa sustava
  - F-009: Uklanjanje vlastitih materijala iz sustava
- Uploadati kalendar u obliku CSV datoteke
  - F-008: Prijenos kalendara u CSV formatu
- Dodavati datume u kalendar unutar aplikacije
  - F-006: Upravljanje osobnim kalendarom u sustavu

- Pregledavanje PDF datoteka
  - F-015: Edukator može pregledavati javno dostupne materijale.

### **3. A-3 Recenzent / Reviewer (inicijator)**

Reviewer pomaže u održavanju kvalitete sadržaja putem evaluacije javnih materijala.

#### Reviewer može:

- Pregledavati javno dostupan sadržaj
  - F-005: Pregled javno dosupnog sadržaja
- Brisati javno dostupan sadržaj sa sustava
  - F-005: Uklanjanje materijala koji ne zadovoljavaju standarde, te ostaviti povratnu poruku
- Uploadati kalendar u obliku CSV datoteke
  - F-008: Prijenos kalendara u CSV formatu
- Dodavati datume u kalendar unutar aplikacije
  - F-006: Upravljanje osobnim kalendarom u sustavu
- Pregledavanje PDF datoteka
  - F-016: Recenzent može pregledavati javno dostupne materijale.

### **4. A-4 Administrator (inicijator)**

Administrator upravlja edukatorima i korisnicima te osigurava pravilno funkcioniranje aplikacije.

#### Administrator može:

- Odobravati edukatore i reviewere
  - F-010: Provjera i odobravanje novih edukatora i reviewera
- Brisati korisnike sa sustava
  - F-011: Administrator može izbrisati korisnike sa sustava
- Uploadati kalendar u obliku CSV datoteke
  - F-008: Prijenos kalendara u CSV formatu
- Dodavati datume u kalendar unutar aplikacije
  - F-006: Upravljanje osobnim kalendarom u sustavu
- Pregledavanje i kreacija oznaka
  - F-012: Pregledavanje i dodavanje oznaka od strane administratora.
- Pregledavanje PDF datoteka
  - F-017: Administrator može pregledavati javno dostupne materijale

### **5. A-5 Default user (inicijator)**

- Prijava na ulogu
  - F-013: Default user se može prijaviti na role
- Izbrisati vlastiti account

### **6. A-6 Baza podataka (sudionik)**

Baza podataka pohranjuje sve podatke o korisnicima, materijale, generirane flashcards-e, recenzije.

### **7. A-7 Gemini 1.5 Flash (sudionik)**

Služi kao AI pomoću kojeg će se generirati flashcards-i pomoću definiranih promptova.

## 2.2 Funkcionalni zahtjevi

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
F-001	Sustav omogućuje korisnicima registraciju putem e-mail adrese ili OAuth2 autentifikacije.	Visok	Zahtjev dionika	Korisnik može kreirati račun putem e-maila, primiti potvrdu i uspješno se prijaviti.
F-002	Sustav omogućuje studentima dodavanje vlastitih materijala (PDF) u osobni profil.	Visok	Zahtjev dionika	Student može uspješno prenijeti materijale i pregledati ih unutar aplikacije.
F-003	Sustav omogućuje Studentima generiranje personaliziranih "flashcards"-a iz odabralih materijala putem AI asistenta.	Visok	Specifikacija projekta	Na temelju priloženih materijala, AI kreira prilagođeni sadržaj za korisnika.
F-004	Edukatori mogu objavljivati dodatne obrazovne resurse dostupne studentima.	Srednji	Povratne informacije korisnika	Edukator može uspješno dodati resurse, a studenti ih mogu pregledati i koristiti.
F-005	Revieweri mogu pregledati i brisati sadržaj objavljen od strane edukatora.	Visok	Zahtjev dionika	Reviewer može uspješno pregledati i potvrditi/odbiti dodane resurse.
F-006	Aplikacija omogućuje svim korisnicima osim default usera pregled rasporeda kroz kalendar.	Srednji	Specifikacija projekta	Svi osim default usera može dodati termine ispita u kalendar i pregledati ih kasnije.

F-007	Sustav omogućuje studentima ocjenjivanje materijala.	Srednji	Specifikacija projekta	Student može ostaviti recenziju na materijal.
F-008	Svi osim default usera mogu uploadati kalendar u obliku CSV datoteke.	Srednji	Zahtjev dionika	Svi osim default usera mogu uspješno prenijeti CSV datoteku, a sustav je prikazuje u osobnom kalendaru unutar aplikacije.
F-009	Korisnici mogu brisati vlastiti sadržaj s aplikacije	Visok	Zahtjev dionika	Svi korisnici koji mogu staviti sadržaj na aplikaciju mogu ga i izbrisati.
F-010	Administrator može odobravati nove edukatore i reviewere na sustavu.	Visok	Zahtjev dionika	Administrator može pregledati i odobriti/odbiti zahtjeve edukatora za pridruživanje sustavu.
F-011	Administrator može izbrisati korisnike sa sustava	Srednji	Zahtjev dionika	Administrator briše korisnika i sve što je korisnik stavio na sustav.
F-012	Pregledavanje i dodavanje oznaka od strane administratora.	Srednji	Zahtjev dionika	Administrator treba moći kreirati i brisati oznake.
F-013	Default user se može prijaviti na role	Visoki	Zahtjev dionika	Default user može se prijaviti za Studenta ili poslati prijavu za role Edukatora i Reviewera.
F-014	Student može pregledati javne i svoje osobne materijale	Visoki	Zahtjev dionika	Student može pregledati javne i svoje osobne materijale tako da ih direktno otvoriti u

				browseru ili preuzme na vlastiti uređaj, ovisno o uređaju.
F-015	Edukator može pregledavati javno dostupne materijale.	Visoki	Zahtjev dionika	Edukator može pregledavati javno dostupne materijale tako da ih direktno otvorи u browseru ili preuzme na vlastiti uređaj, ovisno o uređaju.
F-016	Recenzent može pregledavati javno dostupne materijale.	Visoki	Zahtjev dionika	Recenzent može pregledavati javno dostupne materijale tako da ih direktno otvorи u browseru ili preuzme na vlastiti uređaj, ovisno o uređaju.
F-017	Administrator može pregledavati javno dostupne materijale.	Visoki	Zahtjev dionika	Administrator može pregledavati javno dostupne materijale tako da ih direktno otvorи u browseru ili preuzme na vlastiti uređaj, ovisno o uređaju.

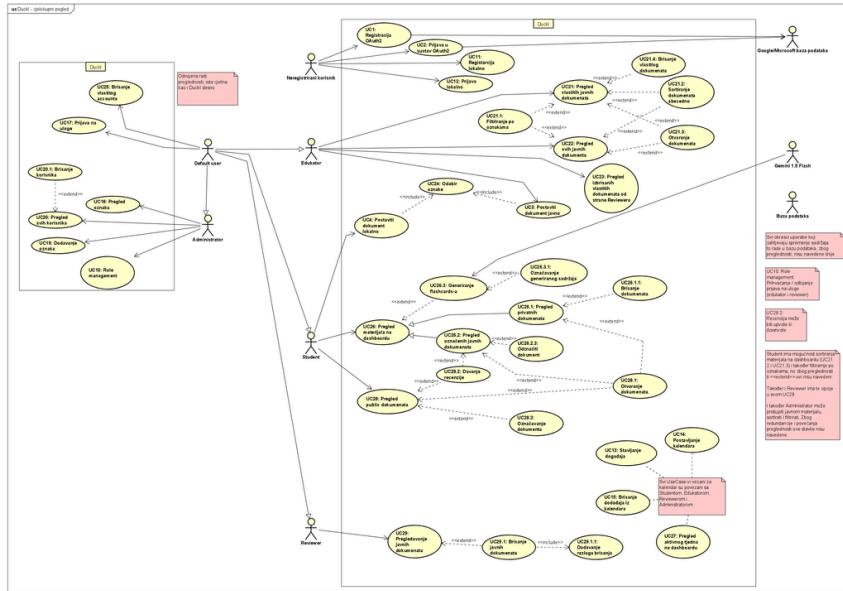
## 2.3 Ostali zahtjevi

ID zahtjeva	Opis	Prioritet
NF-1	Sustav treba omogućiti rad više korisnika u stvarnom vremenu	Visok
NF-2	Sustav treba podržavati primarno hrvatsku abecedu, te imati korisničko sučelje na engleskome jeziku.	Srednji

NF-3	Izvršavanje dijela programa koji pristupa bazi podataka ne smije trajati dulje od 10 sekundi.	Visok
NF-4	Sustav treba biti implementiran kao web aplikacija, te je pristup pomoću HTTPS protokola.	Visok
NF-5	Sustav mora biti intuitivan, trebalo bi ga se moći koristiti bez uputa i vodiča.	Srednji
NF-6	Sustav treba podržavati nadogradnje bez narušavanja postojećih funkcionalnosti.	Visok
NF-7	Veza s bazom podataka mora biti zaštićena, brza i otporna na vanjske pogreške.	Visok
NF-8	Prikaz i izvođenje sadržaja u aplikaciji mora biti prilagođeno za rad na manjim ekranima mobilnih uređaja.	Nizak
NF-9	Neispravno korištenje korisničkog sučelja ne smije narušiti funkcionalnost i rad sustava.	Visok

### 3. Specifikacija zahtjeva sustava

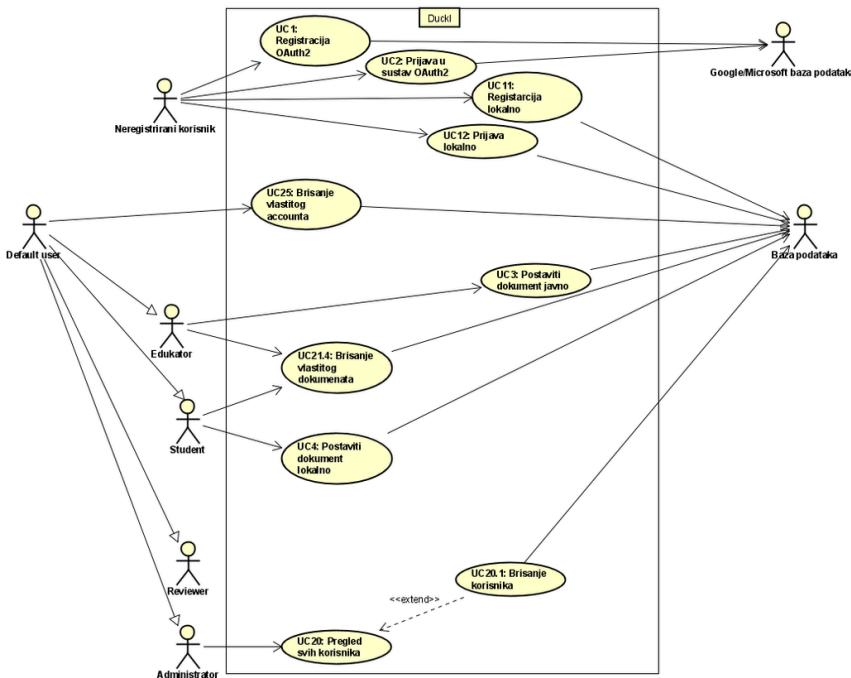
#### 1. Visokorazinski dijagram obrazaca uporabe cijelog sustava



Slika 4.1: Dijagram obrasca uporabe, cjelokupni pregled

#### 2. dijagram obrazaca uporabe za ključne značajke

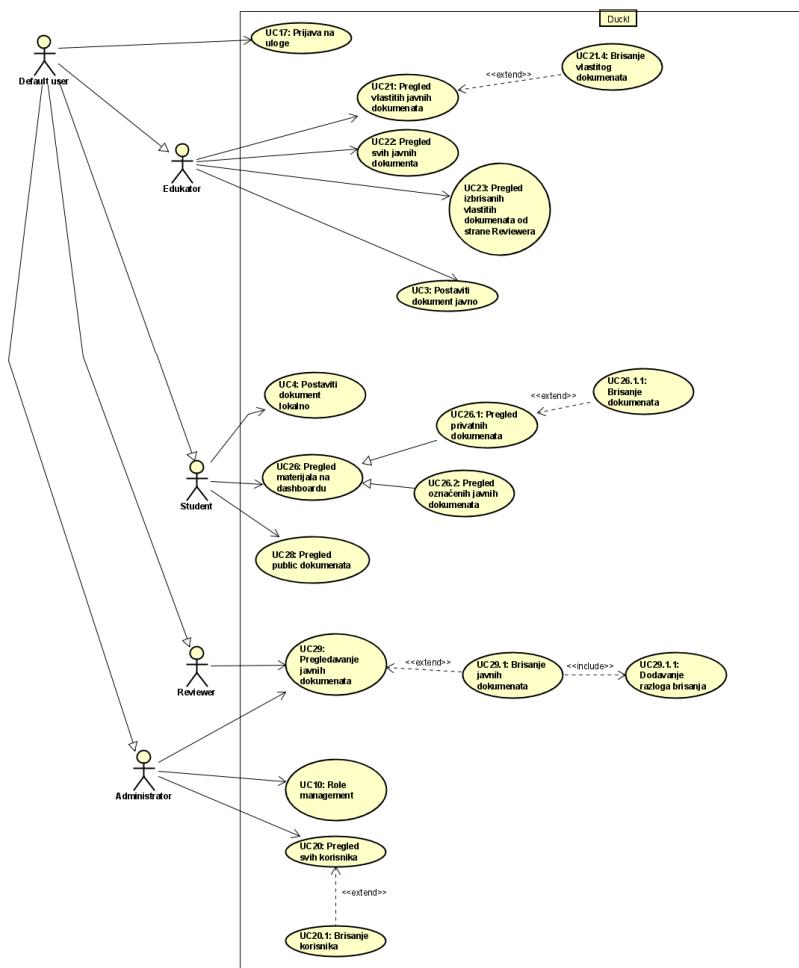
Prikazani svi obrasci uporabe koji imaju veze sa prijavom i pohranom podataka korisnika, te kako oni mogu utjecati na te podatke.



Slika 4.2: Dijagram obrasca uporabe, ključne značajke

#### 3. dijagram obrazaca uporabe za korisničke uloge

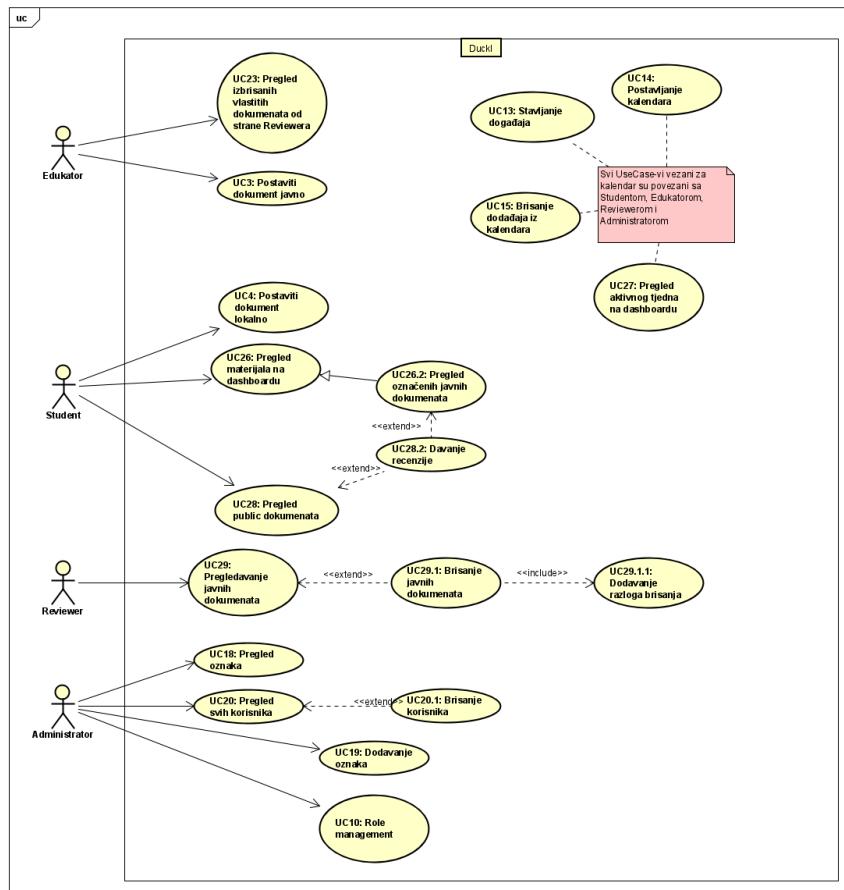
Prikazuju se najbitnije značajke svakog korisnika i kako oni međusobno utječu na sebe i svoju okolinu.



Slika 4.2: Dijagram obrazca uporabe, korisničke uloge

#### 4. dijagram obrazaca uporabe za osnovne poslovne procese

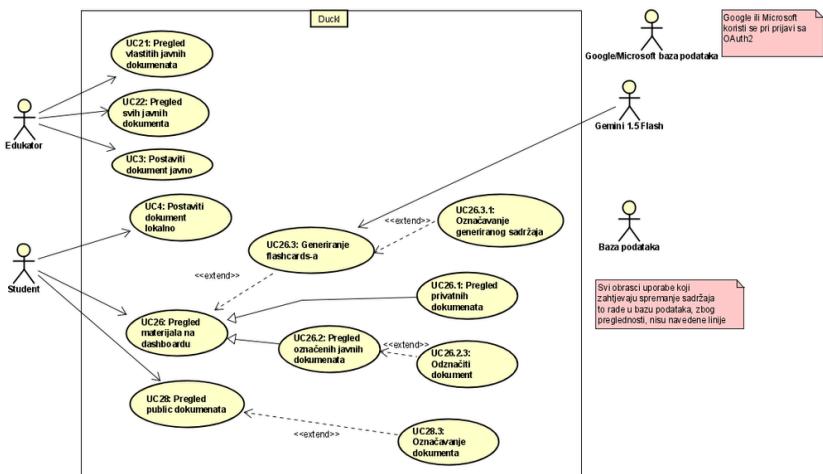
Prikazane su najbitnije "poslovne" prakse u sustavu, te kako svaki korisnik ima mogućnost upravljanjem materijala i vremena, te kako drugi utječu na to.



Slika 4.3: Dijagram obrasca uporabe, poslovni procesi

## 5. dijagram obrazaca uporabe za kritične sustave i integracije

Prikaze interakcije sa vanjskim entitetima.



Slika 4.4: Dijagram obrasca uporabe, kritični sustavi i integracije

## Opis obrazaca uporabe

Zbog preglednosti opisi su navedeni redoslijedom kako su navedeni u cijelokupnom pregledu, jer je kontekst opisa jasniji tako.

### UC1 - Registracija pomoću OAuth2, neregistrirani korisnik

Glavni sudionik: Neregistrirani korisnik

Cilj: Registrirati novog korisnika putem OAuth2 autentifikacije (Google, Microsoft)

Sudionici: OAuth2 pružatelj usluge (npr. Google)

Preduvjet: Korisnik nema postojeći račun na sustavu

Opis osnovnog tijeka:

1. Korisnik otvara stranicu za registraciju i bira opciju "Registracija putem OAuth2" (referenca na funkcionalni zahtjev F-001)
2. Sustav preusmjerava korisnika na stranicu pružatelja usluge (OAuth2).
3. Korisnik unosi podatke za autentifikaciju kod pružatelja usluge.
4. Pružatelj usluge vraća potvrdu sustavu i sustav registrira korisnika.
5. Sustav prikazuje potvrdu uspješne registracije i preusmjerava korisnika na početnu stranicu.

Opis mogućih odstupanja:

1. Ako korisnik prekine postupak autentifikacije kod pružatelja usluge, sustav prikazuje poruku da registracija nije dovršena.
2. Ako pružatelj usluge odbije autentifikaciju, sustav korisniku prikazuje poruku o neuspješnoj registraciji.

#### **UC2 - Prijava u sustav OAuth2**

Glavni sudionik: Neregistrirani korisnik

Cilj: Prijaviti se u sustav pomoću OAuth2 tehnologije.

Sudionici: OAuth2 pružatelj usluge (npr. Google)

Preduvjet: Korisnik ima račun kod pružatelja usluge i registriran je na sustav.

Opis osnovnog tijeka:

1. Korisnik odabire opciju "Prijava putem OAuth2" (referenca na funkcionalni zahtjev F-001).
2. Sustav preusmjerava korisnika na stranicu pružatelja usluge (OAuth2).
3. Korisnik unosi podatke za autentifikaciju kod pružatelja usluge.
4. Pružatelj usluge vraća potvrdu sustavu i sustav prijavljuje korisnika.
5. Sustav preusmjerava korisnika na početnu stranicu.

Opis mogućih odstupanja:

1. Ako korisnik prekine autentifikaciju, sustav prikazuje poruku o neuspješnoj prijavi.
2. Ako pružatelj usluge odbije autentifikaciju, sustav korisniku prikazuje poruku o grešci.

#### **UC11 - Registracija lokalno**

Glavni sudionik: Neregistrirani korisnik

Cilj: Registrirati se pomoću vlastite e-mail adrese i šifre.

Preduvjet: Korisnik nije registriran na sustav.

Opis osnovnog tijeka:

1. Korisnik otvara stranicu za registraciju.
2. Korisnik unosi e-mail adresu i šifru te potvrđuje registraciju (referenca na funkcionalni zahtjev F-001).
3. Sustav provjerava unesene podatke.
4. Ako su podaci valjni, sustav registrira korisnika i prikazuje poruku o uspješnoj registraciji.

Opis mogućih odstupanja:

1. Ako korisnik uneše e-mail koji je već registriran, sustav prikazuje poruku o grešci.
2. Ako su uneseni podaci nevaljni, sustav traži ispravak.

#### **UC12 - Prijava lokalno**

Glavni sudionik: Neregistrirani korisnik

Cilj: Prijaviti se pomoću lokalno kreiranog računa.

Preduvjet: Korisnik ima registriran račun na sustavu.

Opis osnovnog tijeka:

1. Korisnik otvara stranicu za prijavu.
2. Korisnik unosi e-mail i šifru te potvrđuje prijavu (referenca na funkcionalni zahtjev F-001).
3. Sustav provjerava unesene podatke i prijavljuje korisnika.
4. Sustav preusmjerava korisnika na početnu stranicu.

Opis mogućih odstupanja:

1. Ako korisnik uneše neispravan e-mail ili šifru, sustav prikazuje poruku o grešci.

#### **UC21 - Pregled vlastitih javnih dokumenata**

Glavni sudionik: Edukator

Cilj: Pregledati koje javne materijale je edukator postavio na sustav.

Preduvjet: Edukator ima javne dokumente na sustavu.

Opis osnovnog tijeka:

1. Edukator otvara stranicu s vlastitim dokumentima.

2. Sustav prikazuje popis svih javnih dokumenata koje je edukator postavio (referenca na funkcijski zahtjev F-015).

#### **UC21.1 - Filtriranje po oznakama**

Mogući glavni sudionik: Student ili Edukator ili Administrator ili Reviewer

Cilj: Filtrirati dokumente po oznakama.

Opis osnovnog tijeka:

1. Korisnik otvara stranicu s dokumentima.
2. Korisnik odabire filter za oznake.
3. Sustav prikazuje samo dokumente koji odgovaraju odabranoj oznaci.

#### **UC21.2 - Sortiranje dokumenata abecedno**

Mogući glavni sudionik: Edukator ili Student ili Administrator ili Reviewer

Cilj: Sortirati listu dokumenata abecedno po imenu dokumenta.

Opis osnovnog tijeka:

1. Korisnik otvara stranicu s dokumentima.
2. Korisnik odabire opciju za abecedno sortiranje.
3. Sustav prikazuje dokumente sortirane abecednim redom.

#### **UC21.3 i UC28.1 - Otvaranje dokumenta**

Mogući glavni sudionik: Edukator ili Student ili Administrator ili Reviewer

Cilj: Otvoriti dokument za pregled.

Opis osnovnog tijeka:

1. Korisnik odabire dokument s liste.
2. Sustav otvara dokument za pregled u novom prozoru ili sekciji.

#### **UC21.4 - Brisanje dokumenta**

Glavni sudionik: Edukator

Cilj: Obriši vlastiti javno dostupni dokument klikom na "Delete".

Preduvjet: Dokument mora biti u vlasništvu edukatora.

Opis osnovnog tijeka:

1. Edukator otvara stranicu s vlastitim dokumentima.
2. Edukator odabire opciju "Delete" pored želenog dokumenta.
3. Sustav briše dokument i ažurira prikaz.  
*(referenca na funkcijski zahtjev F-009)*

#### **UC22 i UC28 i UC29 - Pregled svih javnih dokumenata**

Mogući glavni sudionik: Edukator ili Student ili Reviewer ili Administrator

Cilj: Pregledati sve javno dostupne dokumente.

Opis osnovnog tijeka:

1. Korisnik otvara stranicu s javnim dokumentima.
2. Sustav prikazuje listu svih dostupnih javnih dokumenata.  
*(referenca na funkcijski zahtjev F-014, F-015, F-016, F-017)*

#### **UC23 - Pregled izbrisanih vlastitih dokumenata od strane Reviewera**

Glavni sudionik: Edukator

Cilj: Pregled liste vlastitih izbrisanih dokumenata sa razlogom brisanja.

Sudionik: Reviewer

Opis osnovnog tijeka:

1. Edukator otvara stranicu s izbrisanim poviješću dokumenata.
2. Sustav prikazuje popis izbrisanih dokumenata i razlog brisanja.

#### **UC4 - Postaviti dokument lokalno**

Glavni sudionik: Student

Cilj: Postaviti dokument na sustav.

Preduvjet: Korisnik ima dokument u podržanom formatu (PDF).

Opis osnovnog tijeka:

1. Student otvara stranicu za postavljanje dokumenata.
2. Student odabire opciju "Browse".
3. Sustav omogućuje odabir lokalne datoteke.
4. Sustav učitava dokument i prikazuje poruku o uspješnom postavljanju.  
*(referenca na funkcijski zahtjev F-002)*

**UC24 - Odabir oznake za dokument koji se planira postaviti**

Mogući glavni sudionik: Edukator ili Student

Cilj: Odabratи oznaku koja daje kontekst o tipu materijala koji se postavlja na sustav.

Opis osnovnog tijeka:

1. Korisnik odabire oznaku s popisa dostupnih oznaka.
2. Sustav povezuje oznaku s dokumentom.

**UC3 - Postaviti dokument javno**

Glavni sudionik: Edukator

Cilj: Postaviti javno dostupan dokument na sustav.

Preduvjet: Dokument je označen odgovarajućom oznakom.

Opis osnovnog tijeka:

1. Edukator odabire opciju "Upload Public PDF".
2. Sustav provjerava valjanost dokumenta.
3. Sustav označava dokument kao javno dostupan i obavještava edukatora o uspješnom postavljanju.  
*(referenca na funkcijски заhtјев F-004)*

**UC26 - Pregled materijala na dashboardu**

Glavni sudionik: Student

Cilj: Pregledati vlastite i označene javne dokumente na osobnom dashboardu.

Opis osnovnog tijeka:

1. Student otvara dashboard.
2. Sustav prikazuje listu vlastitih dokumenata i javnih dokumenata označenih od strane studenta.  
*(referenca na funkcijски заhtјев F-014)*

**UC26.3 - Generiranje flashcards-a**

Glavni sudionik: Student

Sudionici: Gemini 1.5 Flash

Cilj: Generirati flashcards-e za ponavljanje sa izvorom informacija iz selektiranog dokumenta.

Preduvjet: Dokument mora biti u kompatibilnom formatu.

Opis osnovnog tijeka:

1. Student odabire dokument na dashboardu.
2. Student odabire opciju "Generiraj flashcards".
3. Sustav koristi Gemini 1.5 Flash za generiranje sadržaja flashcards-a.
4. Sustav prikazuje generirane flashcards-e.  
*(referenca na funkcijски заhtјев F-003)*

**UC26.3.1 - Označavanje generiranog sadržaja**

Glavni sudionik: Student

Cilj: Označiti i spremiti flashcards za buduću uporabu.

Opis osnovnog tijeka:

1. Student pregledava generirane flashcards-e.
2. Student odabire koje flashcards želi spremiti.
3. Sustav pohranjuje odabrane flashcards-e.

**UC26.1 - Pregled privatnih dokumenata**

Glavni sudionik: Student

Cilj: Pregledati privatne vlastite dokumente postavljene na sustav.

Preduvjet: Korisnik ima privatne dokumente na sustavu.

Opis osnovnog tijeka:

1. Student otvara dashboard.
2. Sustav prikazuje listu privatnih dokumenata korisnika.

**UC26.1.1 - Brisanje dokumenata**

Glavni sudionik: Student

Cilj: Izbrisati privatne vlastite dokumente sa sustava.

Preduvjet: Dokument postoji u korisnikovom privatnom profilu.

Opis osnovnog tijeka:

1. Student odabire opciju "Delete" pored dokumenta.
2. Sustav traži potvrdu o brisanju.

3. Sustav briše dokument.

(referenca na funkcijski zahtjev F-009)

#### **UC26.2 - Pregled označenih javnih dokumenata**

Glavni sudionik: Student

Cilj: Pregledati javne dokumente postavljene na sustav i prethodno označene od strane studenta.

Opis osnovnog tijeka:

1. Student otvara dashboard.

2. Sustav prikazuje listu javnih dokumenata označenih od strane studenta.

#### **UC26.2.3 - Odznačiti dokument**

Glavni sudionik: Student

Cilj: Odznačiti dokument i maknuti ga iz dashboarda, više neće biti vidljiv u studentovim materijalima.

Preduvjet: Dokument je prethodno označen.

Opis osnovnog tijeka:

1. Student odabire označeni dokument.

2. Student bira opciju "Odznači".

3. Sustav uklanja dokument s liste označenih i ažurira prikaz.

#### **UC28.2 - Davanje recenzije**

Glavni sudionik: Student

Cilj: Dati pozitivnu ili negativnu povratnu informaciju na selektirani javni dokument (upvote ili downvote).

Opis osnovnog tijeka:

1. Student otvara stranicu s dokumentima.

2. Student bira opciju za davanje recenzije.

3. Sustav bilježi recenziju i prikazuje ažurirane ocjene dokumenta.

(referenca na funkcijski zahtjev F-007)

#### **UC28.3 - Označavanje dokumenta**

Glavni sudionik: Student

Cilj: Postaviti oznaku na dokument koji postaje vidljiv na studentovom dashboardu za daljnju uporabu.

Opis osnovnog tijeka:

1. Student pregledava listu dokumenata.

2. Student odabire opciju "Označi".

3. Sustav dodaje dokument na dashboard korisnika.

#### **UC29.1 - Brisanje javnih dokumenata**

Glavni sudionik: Reviewer

Cilj: Brisati javno postavljeni dokument od strane Edukatora.

Opis osnovnog tijeka:

1. Reviewer otvara stranicu s javnim dokumentima.

2. Reviewer odabire opciju "Delete" pored dokumenta.

3. Sustav briše dokument i obavještava edukatora o brisanju.

(referenca na funkcijski zahtjev F-005)

#### **UC29.1.1 - Dodavanje razloga brisanja**

Glavni sudionik: Reviewer

Cilj: Dodati kontekst za razlog brisanja dokumenta sa sustava.

Opis osnovnog tijeka:

1. Reviewer unosi tekstualni opis razloga brisanja.

2. Sustav bilježi razlog i obavještava edukatora.

#### **UC27 - Pregled aktivnog tjedna na dashboardu**

Mogući glavni sudionici: Student ili Edukator ili Reviewer ili Administrator

Cilj: Pregledati trenutni tjedan na dashboardu sa prikazanim događajima koji su u tom danu.

Opis osnovnog tijeka:

1. Korisnik otvara dashboard.

2. Sustav prikazuje tjedni prikaz s popisom događaja za svaki dan.

(referenca na funkcijski zahtjev F-006)

#### **UC13 - Stavljanje događaja**

Mogući glavni sudionici: Student ili Edukator ili Reviewer ili Administrator

Cilj: Postaviti novi događaj klikom na jedan dan u mjesecu.

Opis osnovnog tijeka:

1. Korisnik otvara kalendar na dashboardu.
2. Korisnik odabire određeni datum.
3. Sustav omogućuje unos podataka za novi događaj.
4. Korisnik potvrđuje unos, a sustav dodaje događaj u kalendar.

#### **UC14 - Postavljanje kalendarja**

Mogući glavni sudionici: Student ili Edukator ili Reviewer ili Administrator

Preduvjet: Kalendar mora biti u CSV obliku.

Cilj: Uploadati cijeli kalendar događaja u sustav.

Opis osnovnog tijeka:

1. Korisnik otvara opciju za postavljanje kalendara.
2. Korisnik odabire CSV datoteku s kalendarom.
3. Sustav validira i učitava podatke iz CSV datoteke.
4. Sustav dodaje događaje u korisnikov kalendar i prikazuje poruku o uspješnom uploadu.  
*(referenca na funkcijски заhtјев F-008)*

#### **UC15 - Brisanje događaja iz kalendarja**

Mogući glavni sudionici: Student ili Edukator ili Reviewer ili Administrator

Preduvjet: Događaj postoji u kalendaru.

Cilj: Izbrisati postojeći događaj iz kalendarja.

Opis osnovnog tijeka:

1. Korisnik otvara kalendar i odabire događaj koji želi izbrisati.
2. Korisnik odabire opciju "X".
3. Sustav uklanja događaj iz kalendarja i prikazuje poruku o uspješnom brisanju.

#### **UC25 - Brisanje vlastitog računa**

Glavni sudionik: Default user

Preduvjet: Korisnik ima postojeći račun.

Cilj: Izbrisati trenutni korisnički račun iz sustava.

Opis osnovnog tijeka:

1. Korisnik otvara stranicu za postavke računa.
2. Korisnik odabire opciju "Personal data" i "Delete".
3. Sustav traži potvrdu o brisanju računa.
4. Sustav briše račun i prikazuje poruku o uspješnom brisanju.  
*(referenca na funkcijски заhtјев F-009)*

#### **UC17 - Prijava na uloge**

Glavni sudionik: Default user

Preduvjet: Korisnik nema izabranu ulogu.

Cilj: Dodijeliti novu ulogu korisniku. Ukoliko izabere Edukator ili Reviewer, šalje se upit administratoru za odobravanje.

Opis osnovnog tijeka:

1. Korisnik otvara stranicu za uloge.
2. Korisnik odabire željenu ulogu.
3. Ako je uloga Edukator ili Reviewer, sustav šalje zahtjev administratoru na odobrenje.
4. Sustav potvrđuje dodjelu uloge ili obavještava korisnika o čekanju odobrenja.  
*(referenca na funkcijски заhtјев F-013)*

#### **UC18 - Pregled oznaka**

Glavni sudionik: Administrator

Cilj: Pregledati postojeće oznake u sustavu.

Opis osnovnog tijeka:

1. Administrator otvara stranicu za upravljanje oznakama.
2. Administrator klikne gumb za prikaz postojećih oznaka.
3. Sustav prikazuje popis svih dostupnih oznaka.  
*(referenca na funkcijски заhtјев F-012)*

#### **UC19 - Dodavanje oznaka**

Glavni sudionik: Administrator

Cilj: Dodati nove oznake koje drugi edukatori mogu koristiti za obilježavanje materijala.

Opis osnovnog tijeka:

1. Administrator otvara stranicu za upravljanje oznakama.
2. Administrator unosi naziv nove oznake.
3. Administrator odabire opciju "Add Tag".
4. Sustav dodaje oznaku u bazu podataka i obavještava o uspješnom dodavanju.  
*(referenca na funkcijски заhtјев F-012)*

#### **UC20 - Pregled svih korisnika**

Glavni sudionik: Administrator

Cilj: Pregledati sve postojeće korisnike u sustavu.

Opis osnovnog tijeka:

1. Administrator otvara stranicu za upravljanje korisnicima.
2. Sustav prikazuje popis korisnika s osnovnim informacijama (ime, e-mail, uloga).

#### **UC20.1 - Brisanje korisnika**

Glavni sudionik: Administrator

Cilj: Izbrisati postojeće korisnike iz sustava.

Opis osnovnog tijeka:

1. Administrator odabire korisnika na popisu.
2. Administrator odabire opciju "Delete".
3. Sustav briše korisnika i obavještava o uspješnom brisanju.  
*(referenca na funkcijски заhtјев F-011)*

#### **UC10 - Upravljanje ulogama**

Glavni sudionik: Administrator

Cilj: Pregledati postojeće upite za uloge te odobravati ili odbijati korisnike za dobivanje uloge.

Opis osnovnog tijeka:

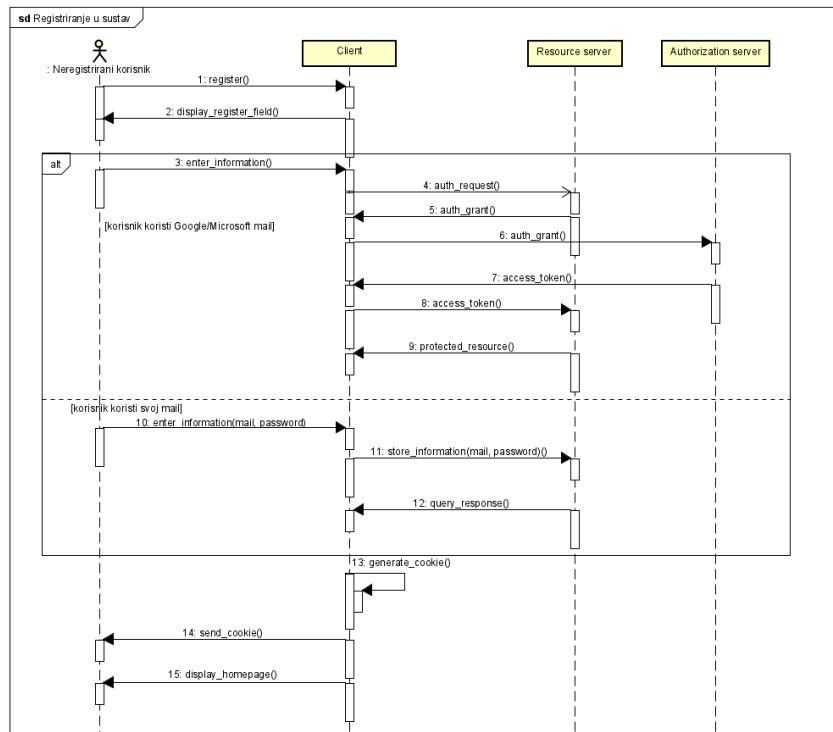
1. Administrator otvara stranicu za upravljanje ulogama.
2. Sustav prikazuje popis zahtjeva za uloge (Edukator, Reviewer).
3. Administrator odabire zahtjev i donosi odluku o odobravanju ili odbijanju.
4. Sustav obavještava korisnika o odluci.  
*(referenca na funkcijски заhtјев F-010)*

### **Sekvencijski dijagrami**

#### **Pregledavanje materijala**

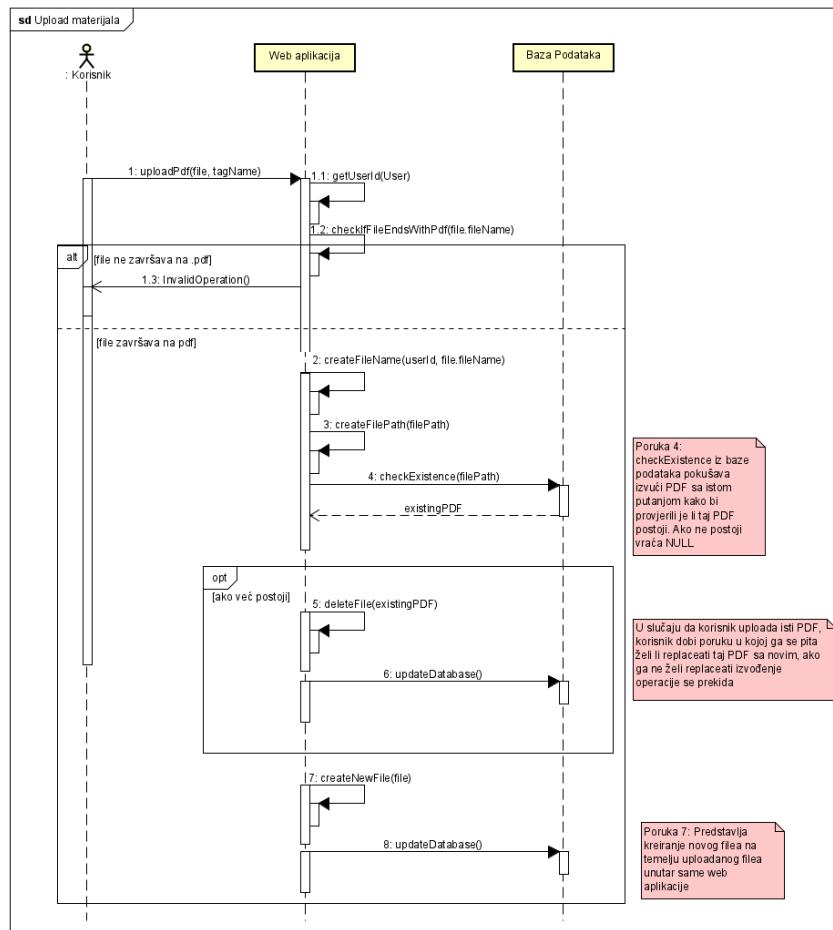
Slika 4.5: Pregledavanje materijala

## Registriranje u sustav



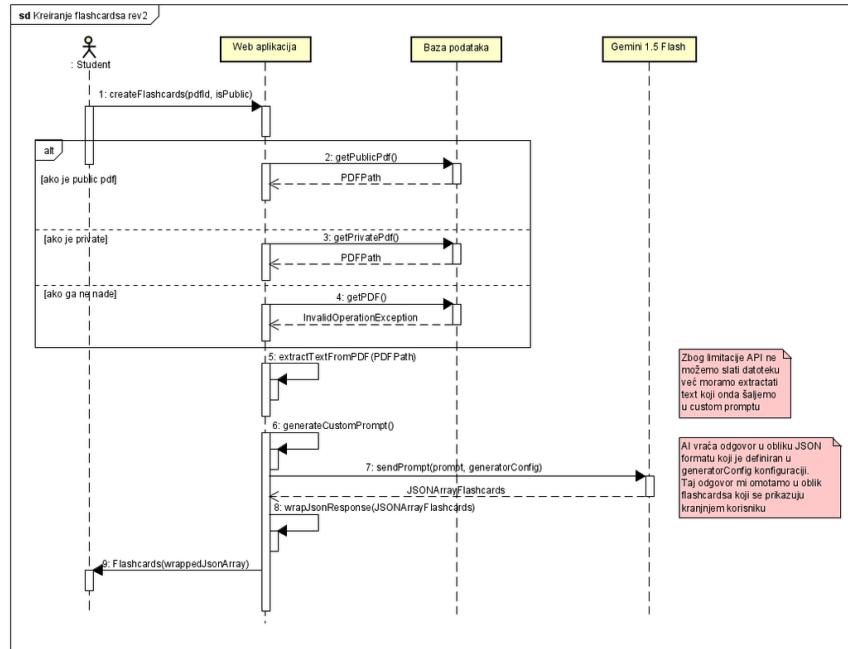
Slika 4.6: Sekvencijski dijagram, registracija u sustav

## Postavljanje materijala na sustav



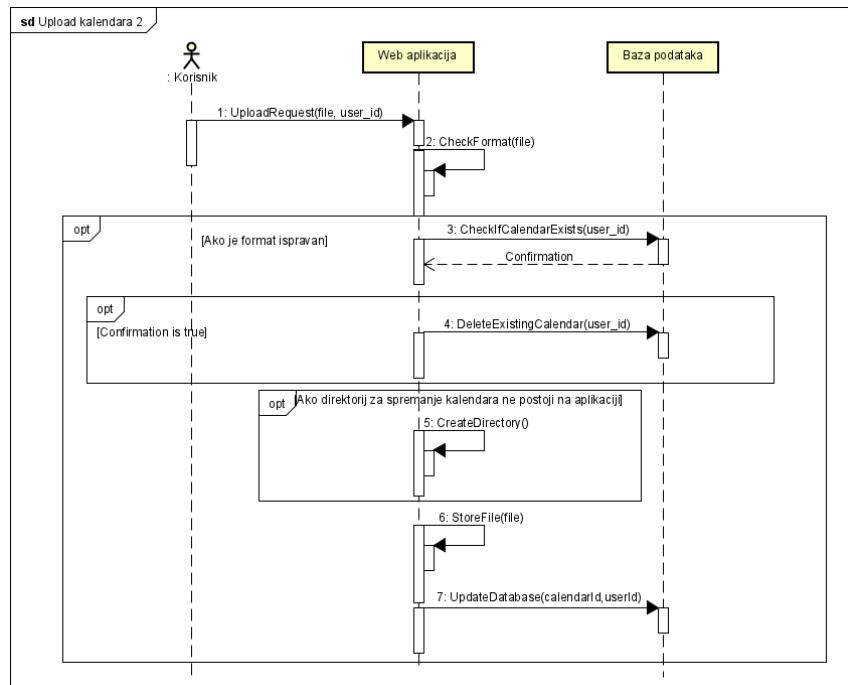
Slika 4.7: Sekvencijski dijagram, postavljanje materijala u sustav

## Generiranje flashcardsa



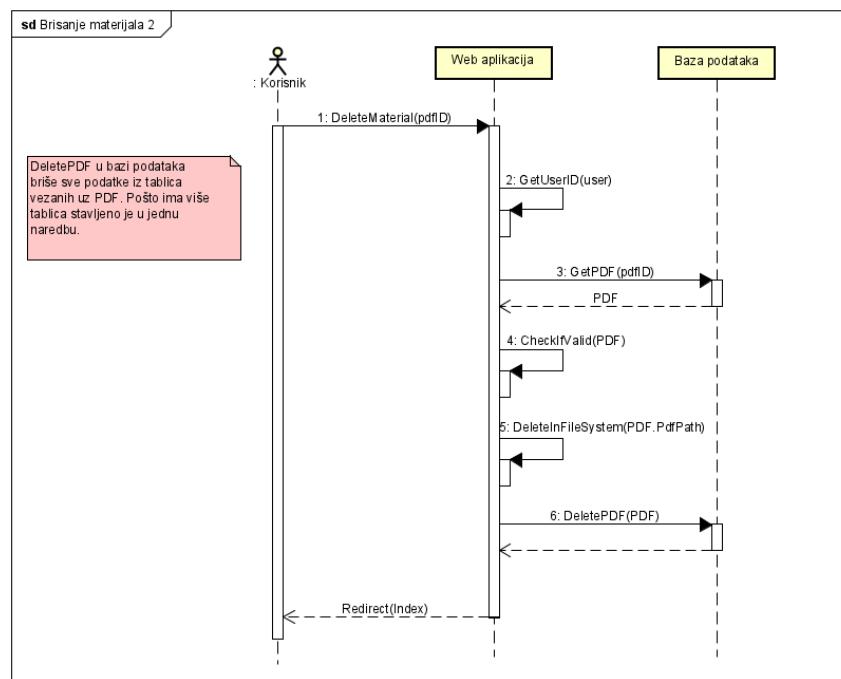
Slika 4.8: Sekvensijski dijagram, generiranje flashcardsa

## Postavljanje kalendara u sustav



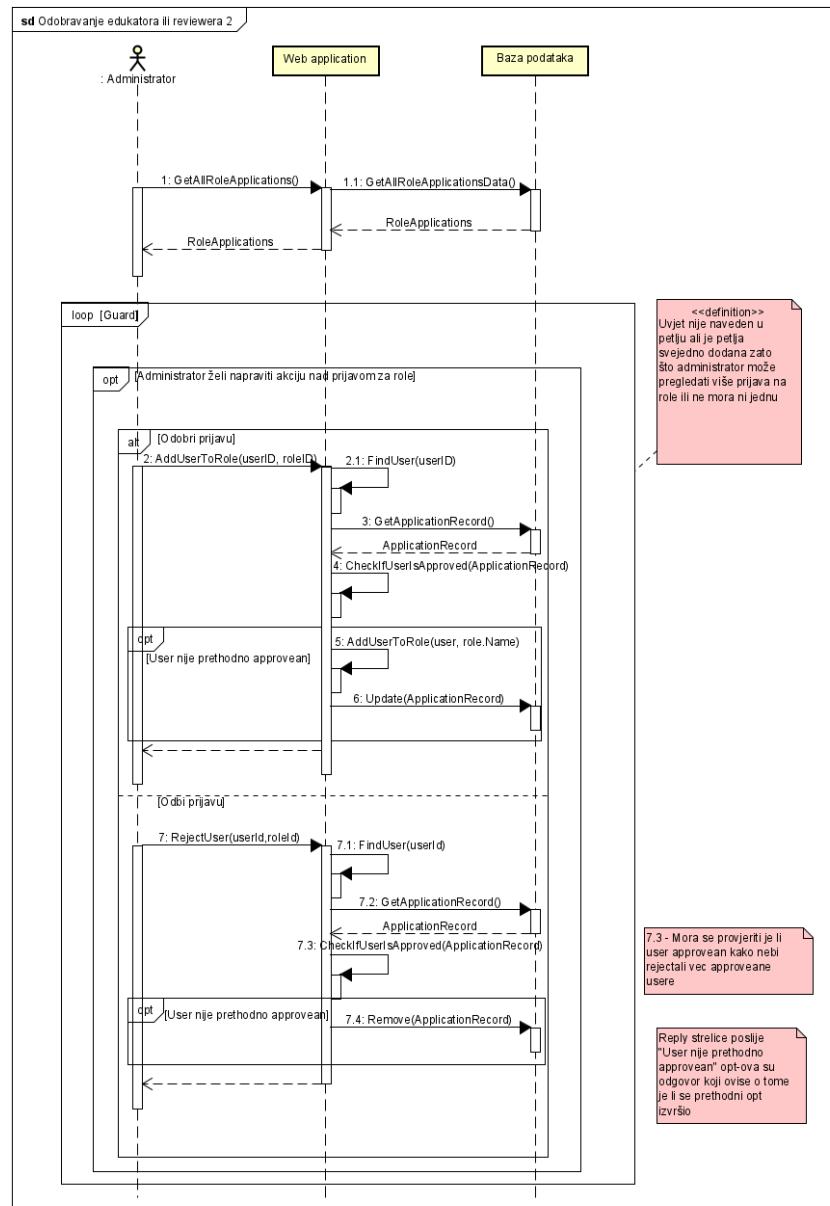
Slika 4.9: Sekvensijski dijagram, postavljanje kalendara u sustav

## Brisanje materijala



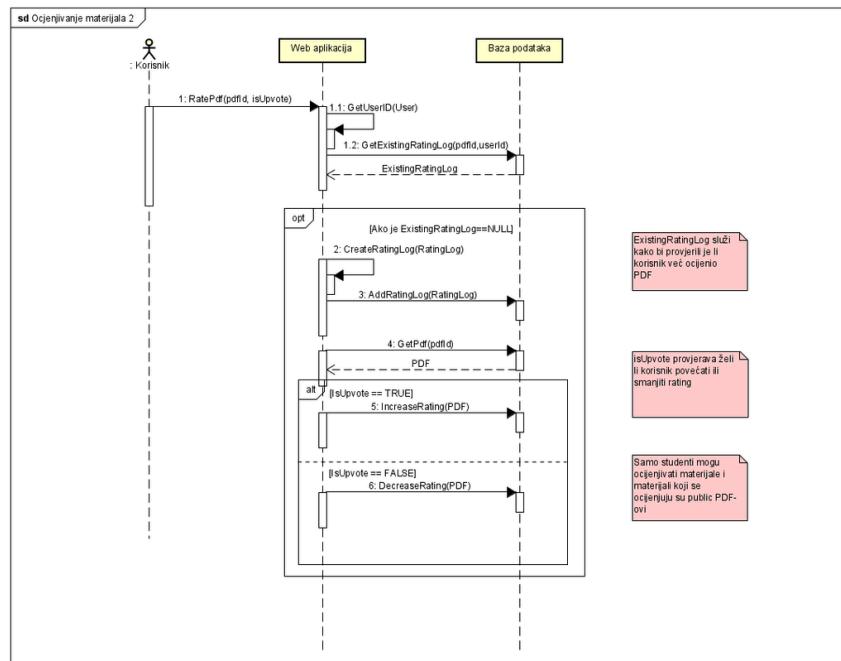
Slika 4.11: Sekvenčni dijagram, brisanje materijala

## Odobravanje edukatora ili reviewera



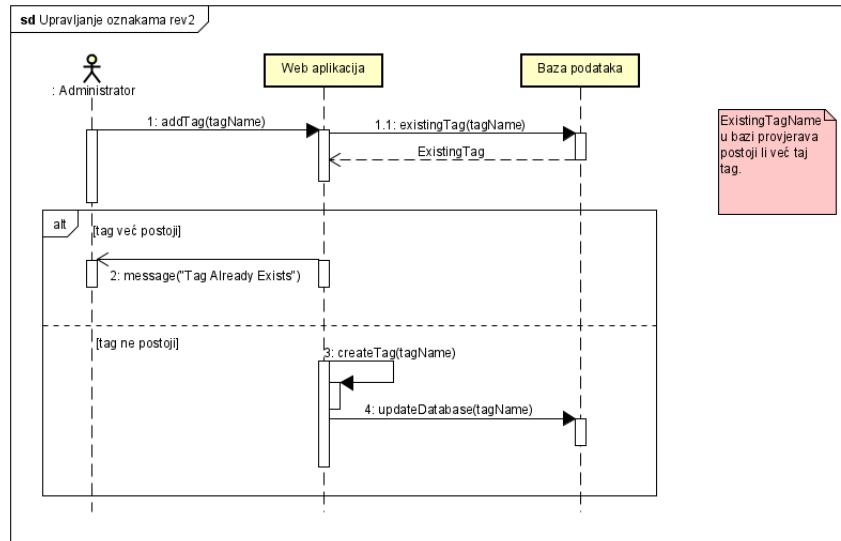
Slika 4.12: Sekvencijski dijagram, approval edukatora ili reviewera

## Ocjenvivanje materijala



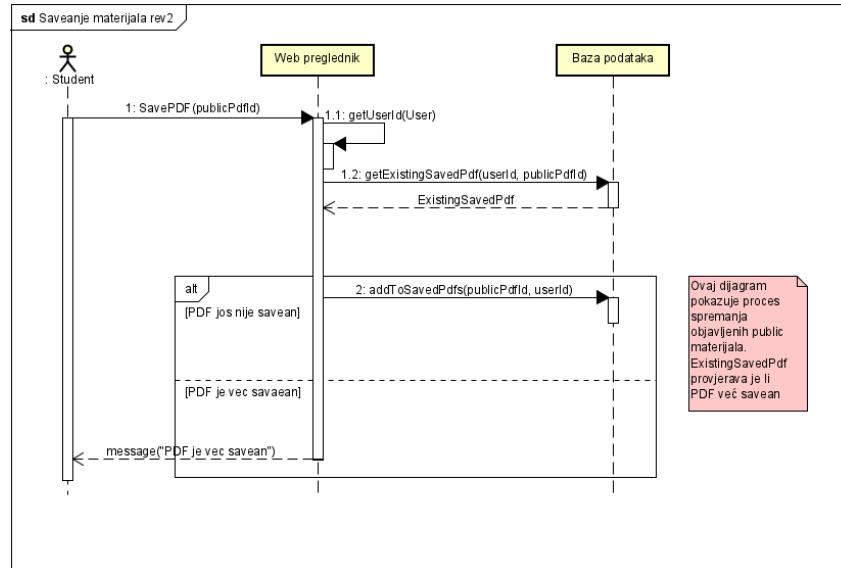
Slika 4.13: Sekvencijski dijagram, ocjenjivanje materijala

## Upravljanje oznakama (eng. Tags)



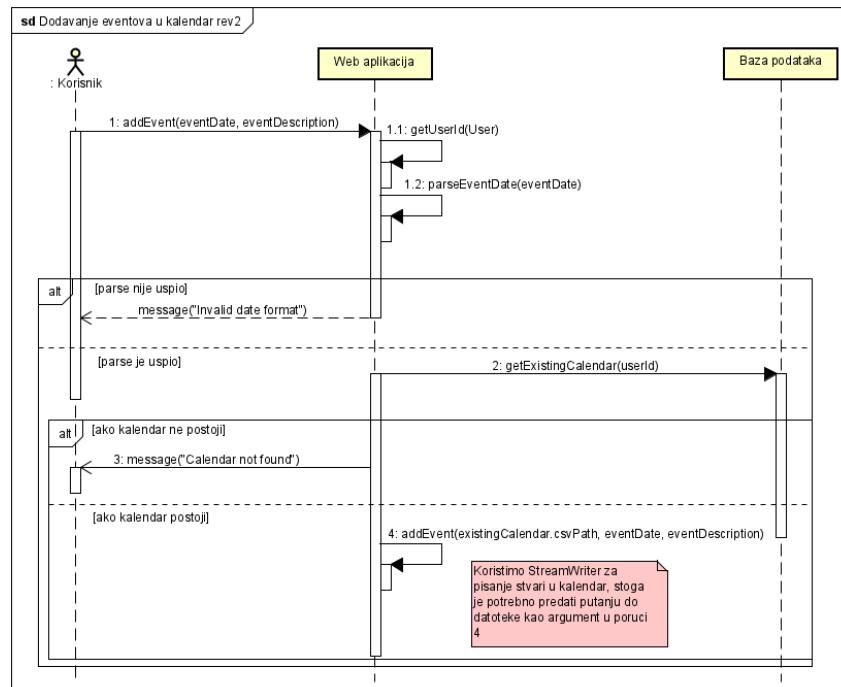
Slika 4.14: Sekvencijski dijagram, upravljanje oznakama

## Save-anje materijala



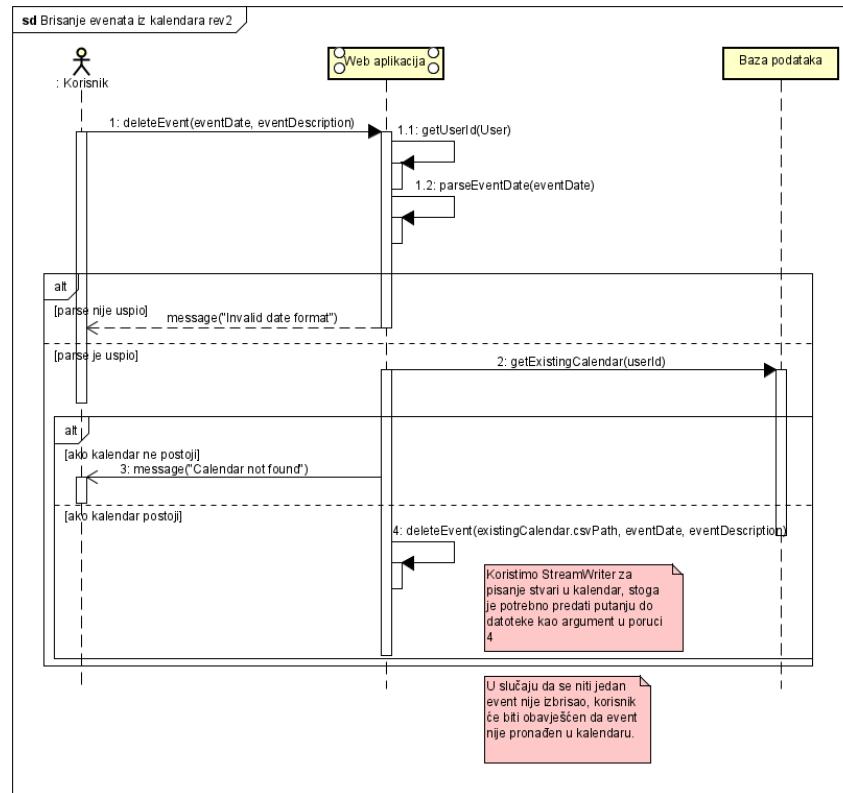
Slika 4.15: Sekvencijski dijagram, saveanje materijala u sustav

## Dodavanje događaja u kalendar



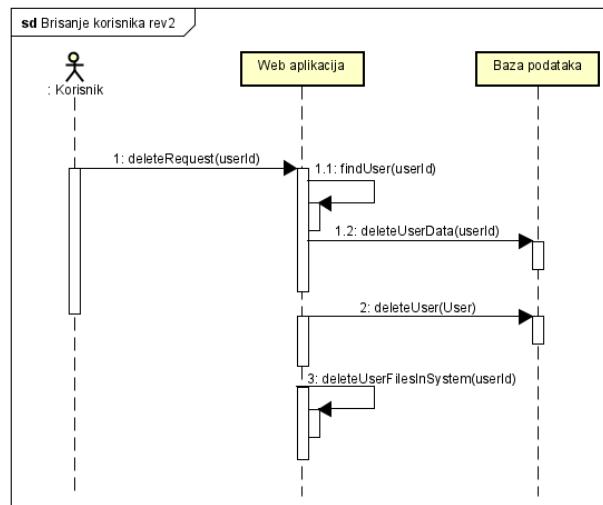
Slika 4.16: Sekvencijski dijagram, dodavanje događaja u kalendar

### Brisanje događaja iz kalendara



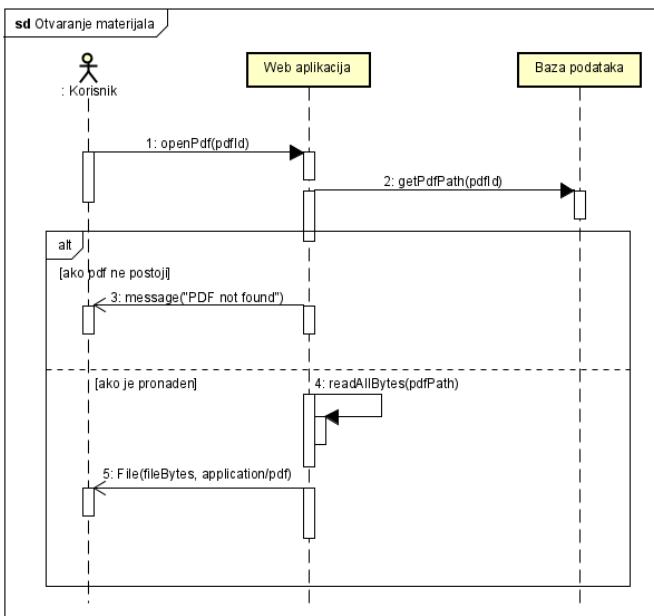
Slika 4.17: Sekvencijski dijagram, brisanje događaja iz kalendara

### Brisanje korisnika sa sustava



Slika 4.18: Sekvencijski dijagram, brisanje korisnika iz sustava

## Otvaranje materijala



Slika 4.19: Sekvencijski dijagram, otvaranje materijala

## Provjera uključenosti ključnih funkcionalnosti u obrasce uporabe

ID obrasca uporabe	Naziv	Obuhvaćeni funkcionalni zahtjevi
UC1	Registracija pomoću OAuth2	F-001
UC2	Prijava u sustav OAuth2	F-001
UC11	Registracija lokalno	F-001
UC12	Prijava lokalno	F-001
UC4	Postaviti dokument lokalno	F-002
UC26.3	Generiranje flashcards-a	F-003
UC3	Postaviti dokument javno	F-004
UC29.1	Brisanje javnih dokumenata	F-005
UC27	Pregled aktivnog tjedna na dashboardu	F-006
UC28.2	Davanje recenzije	F-007
UC14	Postavljanje kalendarja	F-008
UC15	Brisanje događaja iz kalendarja	F-008
UC26.1.1	Brisanje dokumenata	F-009
UC29.1.1	Dodavanje razloga brisanja	F-005
UC10	Upravljanje ulogama	F-010
UC20	Pregled svih korisnika	F-011
UC20.1	Brisanje korisnika	F-011
UC18	Pregled oznaka	F-012

UC19	Dodavanje oznaka	F-012
UC17	Prijava na uloge	F-013
UC22, UC28, UC29	Pregled svih javnih dokumenata	F-014, F-015, F-016, F-017
UC21	Pregled vlastitih javnih dokumenata	F-015
UC26	Pregled materijala na dashboardu	F-014

## 4. Arhitektura i dizajn sustava

### Arhitektura sustava

Cilj ovog poglavlja je pružiti jasan, sažet i strukturiran pregled arhitekture DuckI programskog sustava u razvoju, uz razrađivanje ključnih komponenata i njihovih međusobnih odnosa. Ova dokumentacija treba omogućiti svim sudionicima projekta potpuno razumijevanje arhitekture sustava, načina implementacije, podjele odgovornosti te kako sustav funkcioniра kao cjelina. Uključivanje odgovarajućih dijagrama pomaže u kvalitetnom dokumentiranju i vizualizaciji strukture sustava i njegovih ključnih komponenti.

### Opis arhitekture

Ovdje je opisan pregled arhitekture na visokoj razini, uključujući stil arhitekture sustava.

- Stil arhitekture: DuckI programski sustav koristi monolitnu arhitekturu sustava, te koristi principe „povećaj koheziju“ (nadogradnja na podjeli pa vladaj) pošto su međusobno povezani elementi grupirani (npr. kontroleri), „smanji međuvisnost“ pošto su komponente uglavnom „single-task“, tj. fokusiraju se na jedan zadatak te se mogu samostalno testirati, „povećaj uporabu postojećeg“ pošto su dijelovi programskog koda ponovno uporabljivi, „oblikuj za ispitivanje“ pošto je kod strukturiran na način pogodan za testiranje što je ključno za programsku potporu koja koristi monolitnu arhitekturu, „oblikuj konzervativno“ pošto se rubni slučajevi prate, te se u njihovim slučajevima izbacuju iznimke. Pošto su nam resursi bili ograničeni, umjesto klijent - server arhitekture odlučili smo se za monolitnu arhitekturu kako bi olakšali implementiranje aplikaciju na neku platformu.
- Podsustavi: Podsustavi koji se koriste u backend dijelu su servisi, kontroleri, modeli, podaci. Ovisno o vrsti kontrolera, kontroleri mogu vratiti stranicu ili neke druge podatke. Servisi se koriste za implementaciju servisa koje koriste kontrolери. Podsustavi modeli i podaci su međusobno povezani pošto oni rješavaju bazu podataka. Unutar modela su opisane tablice koje se koriste, dok unutar podatkovnog podsustava možemo pronaći datoteke koje ne možemo spremiti u bazu organizirane po direktorijima, te ApplicationDbContext klasu koja upravlja bazom podataka.
- Preslikavanje na radnu platformu: Trenutačan deployment održan lokalno na računalu preko port-forwardinga za svrhe prezentacije početne funkcionalnosti.
- Spremišta podataka: Relacijska SQLite baza podataka u kojoj se spremaju sve informacije za aplikaciju (user login info, user data, public data, other app data). Datoteke poput PDF-ova, CSV-a i JSON-a koji će se koristiti za prikaz korisničkih informacija, spremaju se direktno na datotečni sustav, te u bazi podataka će samo biti putanje koje pokazuju na te datoteke.
- Mrežni protokoli: HTTPS
- Globalni upravljački tok: Nakon što korisnik stisne neki gumb ili preko neke druge komponente zatraži nekakve podatke, on poziva određenu rutu preko kontrolera zaduženog za tu rutu. Prvo se provjerava ima li korisnik pristup određenoj ruti. Ako se utvrdi da ima, njegov zahtjev se proslijeđuje bazi podataka te ovisno o vrsti zahtjeva (GET, POST, ili nešto drugo), u bazi se dohvataju podaci/uređuju podaci. Nakon rada s podacima korisniku se preko kontrolera uredi stranica s izmjenama koje je zatražio.
- Sklopovskoprogramske zahtjevi: Opišite potrebne tehničke zahtjeve vezane uz sklopljive i program, uključujući podršku za specifične operativne sustave, procesore, memoriju i druge komponente.

### Obrazloženje odabira arhitekture

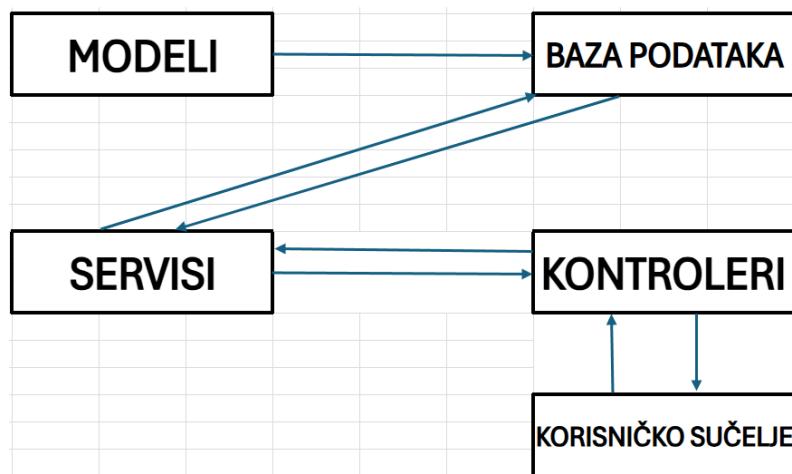
Kako je navedeno u prethodnim točkama, DuckI programski sustav koristi monolitnu arhitekturu sustava, te koristi principe „povećaj koheziju“ (nadogradnja na podjeli pa vladaj) pošto su međusobno povezani elementi grupirani (npr. kontroleri), „smanji međuvisnost“ pošto su komponente uglavnom „single-task“, tj. fokusiraju se na jedan zadatak te se mogu samostalno testirati, „povećaj uporabu postojećeg“ pošto su dijelovi programskog koda ponovno uporabljivi, „oblikuj za ispitivanje“ pošto je kod strukturiran na način pogodan za testiranje što je ključno za programsku potporu koja koristi monolitnu arhitekturu, „oblikuj konzervativno“ pošto se rubni slučajevi prate, te se u njihovim slučajevima izbacuju

iznimke. Za monolitnu arhitekturu smo se odlučili radi njene jednostavnosti pošto sadrži sve što je nama potrebno za aplikaciju (autorizaciju, prezentaciju, „biznis“ logiku, podatkovni sloj, jednostavne notifikacije). Također za monolitnu arhitekturu smo se odlučili jer programska potpora rađena u ovoj arhitekturi obično ima komponente povezane u jednu veliku aplikaciju. Time se iznimno olakšava preslikavanje na radnu platformu, što nam je predstavljalo najveće ograničenje u radu na aplikaciji. Prethodno smo razmatrali i klijent - server arhitekturu, ali zbog otežanog preslikavanje na radnu platformu odlučili smo se ipak za monolitnu arhitekturu.

Jednostavna skica monolitne arhitekture (vlastita slika):



Inicijalni dijagram visoke razine (vlastita slika):



## Organizacija sustava na visokoj razini

Organizacija sustava na najvišoj razini apstrakcije:

- Baza podataka: Koristi se SQLite baza podataka. U njoj se pohranjuju kalendari, pdf materijali i ostali važni podaci za korisnike, te se također koristi za dodjeljivanje uloga korisnika i autorizaciju.
- Datotečni sustav: Unutar Data > Files direktorija pohranjujemo podatke koje se ne mogu pohraniti u bazu podataka. U bazu podataka se pohranjuju rute na datoteku spremljenu u Data > Files, umjesto same datoteke.
- Grafičko sučelje: Aplikacija je web aplikacija koja koristi različite rute za prikazivanje stranica te dohvaćanje i manipulaciju podataka u bazi.
- Vanjski sustavi: Gemini 1.5 Flash model, Google i Microsoft OAuth2.

## Organizacija aplikacije

Organizacija aplikacije na složenijoj razini, uključujući strukturu slojeva i komponenata aplikacije:

- Frontend i Backend slojevi: Frontend sloj koristi viewove za prikaz podataka. Backend koristi modele, servise i kontrolere za rad s rutama i bazom.

- MVC arhitektura: viewovi su zaduženi za prikaz korisničkog sučelja, servisi su zaduženi za logiku, dok su kontroleri zaduženi za pozivanje servisa

## Baza podataka

Za bazu podataka izabran je SQLite zbog njegove jednostavnosti. SQLite je library jezika C koji implementira mali, brzi, samostalni, visokopouzdani i potpuni SQL database engine. Cijela baza nalazi se unutar jedne *app.db* datoteke u *root* direktoriju, te se njoj može pristupati bez potrebe za serverom, što pojednostavljuje upravljanje bazom podataka i čini je visoko prenosivom na različitim platformama. SQLite se široko koristi u ugrađenim sustavima, mobilnim aplikacijama i desktop aplikacijama.

### Opis tablica

Napomena: Primarni ključevi i kompozitni primarni ključevi su označeni oznakom (**PK**) uz ime propertyja, dok su strani ključevi označeni s (**FK**).

#### `sqlite_master`

Property	Type	Description
type	text	Vrsta objekta baze podataka kao što je tablica, indeks, okidač ili prikaz.
name	text	Naziv objekta baze podataka.
tbl_name	text	Naziv tablice s kojom je povezan objekt baze podataka.
rootpage	int	Root stranica.
sql	text	SQL korišten da se napravi objekt baze podataka.

#### `_EFMigrationsHistory`

Property	Type	Description
ProductVersion	text	Verzija EF Core koja je korištena za primjenu migracija.
MigrationId ( <b>PK</b> )	text	Jedinstveni identifikator za migraciju.

#### `sqlite_sequence`

Property	Type	Description
name	unknown	Tablica asocirana s AUTOINCREMENT stupcem.
seq	unknown	Zadnji redni broj korišten u stupcu AUTOINCREMENT.

## AspNetUsers

Property	Type	Description
AccessFailedCount	integer	Broj neuspjelih pokušaja prijave. Ovo se koristi za funkciju zaključavanja.
ConcurrencyStamp	text	Jedinstvena vrijednost koja se mijenja kad god se korisnički profil ažurira, koristi se za optimističnu kontrolu istovremenosti, osiguravajući integritet podataka. U scenarijima u kojima postoji više pokušaja izmjene istog korisničkog zapisa istovremeno, ConcurrencyStamp osigurava da promjene nisu u sukobu.
Email	text	Adresa e-pošte korisnika.
EmailConfirmed	integer	Booleova vrijednost označava je li korisnikova adresa e-pošte potvrđena.
LockoutEnabled	integer	Booleova vrijednost označava može li se račun zaključati za korisnika.
LockoutEnd	text	Datum i vrijeme završetka zaključavanja (ako je korisnik trenutačno zaključan).
NormalizedEmail	text	Normalizirana verzija e-pošte za dosljedno postavljanje upita, obično velikim slovima. Ovo se koristi u usporedbama bez obzira na velika i mala slova.
NormalizedUserName	text	Normalizirana verzija korisničkog imena za dosljedno postavljanje upita.
PasswordHash	text	Hash verzija korisničke zaporce. AspNetCore Identity koristi siguran mehanizam za hashanje lozinki.
PhoneNumber	text	Telefonski broj korisnika.

PhoneNumberConfirmed	integer	Booleova vrijednost označava je li telefonski broj korisnika potvrđen.
SecurityStamp	text	Slučajna vrijednost označava jesu li se promijenile informacije o korisniku povezane sa sigurnošću. Na primjer, mijenja se kada korisnik promijeni lozinku, poništi lozinku ili doda vanjsku prijavu. Njegova primarna svrha je ponisti sve postojeće sesije ili kolačiće kada se promijene informacije povezane sa sigurnošću. Ovo je sigurnosna mjera kojom se osigurava da stare sesije više nisu valjane ako su vjerodajnice ugrožene i zatim promijenjene.
TwoFactorEnabled	integer	Booleova vrijednost označava je li dvofaktorska provjera autentičnosti omogućena za korisnika.
UserName	text	Korisničko ime korisnika. Jedinstven je i koristi se za identifikaciju.
<b>Id (PK)</b>	text	Primarni ključ za korisnika. Jedinstveni identifikator za svakog korisnika, obično GUID niz.

### AspNetUserTokens

Property	Type	Description
Value	text	Vrijednost tokena.
<b>UserId (PK)</b>	text	ID korisnika iz tablice AspNetUsers. Dio kompozitnog primarnog ključa.
<b>LoginProvider (PK)</b>	text	Ovaj stupac navodi naziv pružatelja koji je generirao token. Na primjer, to može biti interni pružatelj (poput sustava za poništavanje zaporce ili sustava za potvrdu e-pošte) ili vanjski pružatelj

		autentifikacije (kao što su Google, Facebook itd.) ako je token povezan s vanjskom autentifikacijom. Dio je složenog primarnog ključa.
Name ( <b>PK</b> )	text	Ovaj stupac pohranjuje naziv tokena, kao što je token za potvrdu e-pošte, token za poništavanje lozinke ili token za dvofaktornu provjeru autentičnosti. To može biti nešto poput PasswordReset, EmailConfirmation, AccessToken, itd. Također je dio složenog primarnog ključa.

### AspNetRoles

Property	Type	Description
ConcurrencyStamp	text	Jedinstveni pečat koji upravlja istodobnim uređivanjem istog zapisa uloge. Pomaže u održavanju integriteta podataka osiguravajući da istodobne operacije ne prebrišu promjene.
Name	text	Naziv uloge. Ovo je čovjeku čitljiv naziv koji se koristi u našem aplikacijskom kodu kada dodjeljujemo uloge korisnicima ili autoriziramo korisnike na temelju njihovih uloga.
NormalizedNome	text	Normalizirana verzija polja Name, obično uppercase verzija imena uloge i koristi se u usporedbama bez obzira na velika i mala slova.
<b>Id (<b>PK</b>)</b>	text	Primarni ključ za ulogu. Jedinstveni identifikator za svaku ulogu, obično GUID niz.

### AspNetUserRoles

Property	Type	Description
<b>UserId (<b>PK</b>) (<b>FK</b>)</b>	text	ID korisnika iz tablice AspNetUsers. Dio

		kompozitnog primarnog ključa. Odgovara stupcu Id u tablici AspNetUsers. Djeluje kao strani ključ koji se povezuje s tablicom AspNetUsers.
RoleId <b>(PK) (FK)</b>	text	ID uloge iz tablice AspNetRoles. Dio kompozitnog primarnog ključa. Odgovara stupcu Id u tablici AspNetRoles. Djeluje kao strani ključ koji se povezuje s tablicom AspNetRoles.

#### AspNetRoleClaims

Property	Type	Description
ClaimType	text	Ovaj stupac pohranjuje vrstu zahtjeva, kao što je dozvola, razina pristupa ili bilo koja druga vrsta koja ima smisla u kontekstu vaše aplikacije.
ClaimValue	text	Stvarna vrijednost tražbine. Na primjer, ako je vrsta zahtjeva Dozvola, vrijednost zahtjeva može biti Edit_User ili View_Reports, itd.
RoleId <b>(FK)</b>	text	ID uloge povezan s ovim zahtjevom. Strani ključ koji se povezuje sa stupcem Id u tablici AspNetRoles.
<b>Id (PK)</b>	integer	Ovo je primarni ključ tablice.

#### AspNetUserClaims

Property	Type	Description
ClaimType	text	Vrsta zahtjeva (npr. "datum rođenja").
ClaimValue	text	Vrijednost zahtjeva (npr. "1980-01-01").
UserId <b>(FK)</b>	text	ID korisnika povezanog s ovim zahtjevom. Djeluje kao strani ključ koji povezuje stupac Id u tablici AspNetUsers.

<b>Id (PK)</b>	integer	Primarni ključ za zahtjev korisnika.
----------------	---------	--------------------------------------

### AspNetUserLogins

Property	Type	Description
ProviderDisplayName	text	Ovaj izborni stupac može pohraniti ime za prikaz davatelja usluge prijave (npr. "Google" umjesto „ <a href="https://google.com">https://google.com</a> “). Uglavnom se koristi za potrebe prikaza u korisničkom sučelju.
UserId (FK)	text	Lokalni korisnički ID koji je povezan s ovom prijavom. Djeluje kao strani ključ koji se povezuje sa stupcem Id u tablici AspNetUsers. Identificira korisnika povezanog s određenom prijavom.
LoginProvider (PK)	text	Ovaj stupac pohranjuje naziv vanjskog pružatelja autentifikacije (npr. Google, Facebook, Microsoft itd.). To je dio složenog primarnog ključa za tablicu.
ProviderKey (PK)	text	Ovaj stupac pohranjuje jedinstveni identifikator od davatelja usluge prijave za korisnika. Na primjer, kada se korisnik prijavi pomoću Googlea, ovo polje će pohraniti jedinstveni ID koji je korisniku dodijelio Google. Također je dio složenog primarnog ključa.

### Calendars

Property	Type	Description
CsvPath	text	Putanja na kojoj je spremljen .csv file.
CalendarId (PK)	integer	Id kalendara.

## UserCalendars

Property	Type	Description
UserId ( <b>PK</b> ) ( <b>FK</b> )	text	Id korisnika koji je foreign key na tablicu AspNetUsers.
CalendarId ( <b>PK</b> ) ( <b>FK</b> )	integer	Id kalendarja koji je foreign key na tablicu Calendars.

## PublicPdfs

Property	Type	Description
PdfPath	text	Putanja do PDF dokumenta unutar aplikacije.
Rating	integer	Ocjena PDF-a.
PdfName	text	Ime PDF dokumenta.
PublicPdfId ( <b>PK</b> )	integer	ID PDF dokumenta.

## PrivatePdfs

Property	Type	Description
PrivatePdfId ( <b>PK</b> )	integer	ID PDF dokumenta.
PdfPath	text	Putanja do PDF dokumenta unutar aplikacije
PdfName	text	Ime PDF dokumenta.

## Tags

Property	Type	Description
TagId ( <b>PK</b> )	integer	ID taga.
TagName	text	Ime, tj. vrijednost, taga.

## PublicPdfTags

Property	Type	Description
TagId ( <b>FK</b> )	integer	Id taga koji pripada određenom public PDF-u (veže tag s PDF dokumentom).
PublicPdfId ( <b>PK</b> ) ( <b>FK</b> )	integer	Id public PDF-a kojem pripada određen tag (veže PDF s tagom).

### **PrivatePdfTags**

<b>Property</b>	<b>Type</b>	<b>Description</b>
TagId <b>(FK)</b>	integer	Id taga koji pripada određenom private PDF-u (veže tag s PDF dokumentom).
PrivatePdflId <b>(PK) (FK)</b>	integer	Id private PDF-a kojem pripada određen tag (veže PDF s tagom).

### **StudentPdfs**

<b>Property</b>	<b>Type</b>	<b>Description</b>
UserId <b>(FK)</b>	text	Id korisnika kojem pripada privatni PDF.
PrivatePdflId <b>(PK) (FK)</b>	integer	Id privatnog PDF dokumenta.

### **EducatorPdfs**

<b>Property</b>	<b>Type</b>	<b>Description</b>
UserId <b>(FK)</b>	text	Id korisnika kojem pripada public PDF.
PublicPdflId <b>(PK) (FK)</b>	integer	Id public PDF dokumenta.

### **RemovedLogs**

<b>Property</b>	<b>Type</b>	<b>Description</b>
RemoveLogId <b>(PK)</b>	integer	Id jednog remove log recorda.
ReviewerId <b>(FK)</b>	text	Id Reviewera koji je izbrisao PDF.
EducatorId <b>(FK)</b>	text	Id Educatora čiji je PDF izbrisana.
Description	text	Opis razloga brisanja PDF-a.
FileName	text	Ime izbrisano PDF dokumenta.

### **FlaggedPdfs**

<b>Property</b>	<b>Type</b>	<b>Description</b>
UserId <b>(PK) (FK)</b>	text	Id studenta koji je spremio public PDF.

PublicPdfId <b>(PK) (FK)</b>	integer	Id public PDF-a.
------------------------------	---------	------------------

### RatingLogs

Property	Type	Description
UserId <b>(PK) (FK)</b>	text	Id korisnika koji je ostavio ocjenu na jednom PDF-u.
PublicPdfId <b>(PK) (FK)</b>	integer	Id public PDF-a na kojem je korisnik ostavio ocjenu.

### Flashcards

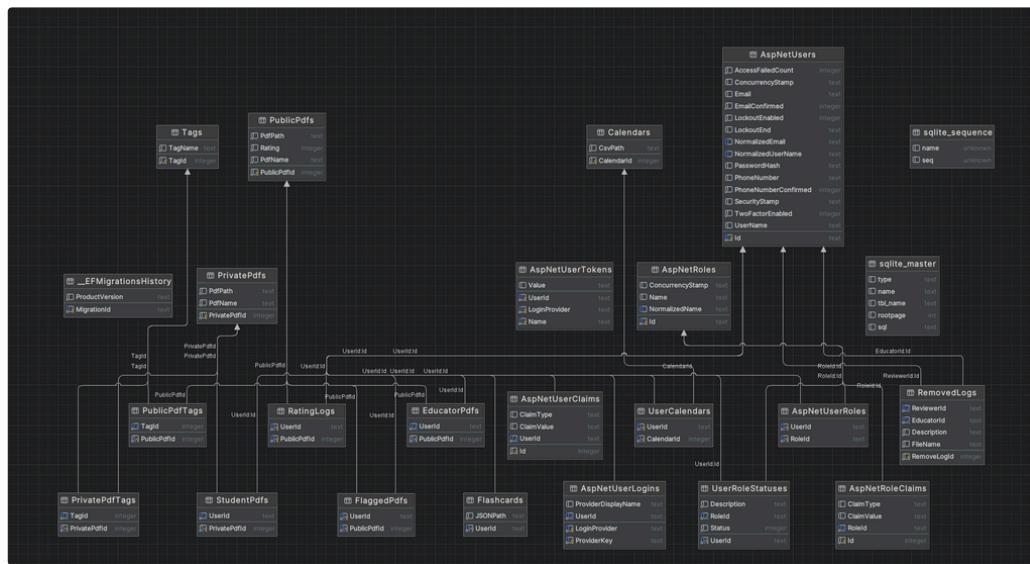
Property	Type	Description
UserId <b>(PK) (FK)</b>	text	Id korisnika kojem pripada flashcard.
JSONPath	text	Putanja do JSON datoteke flashcarda.

### UserRoleStatuses

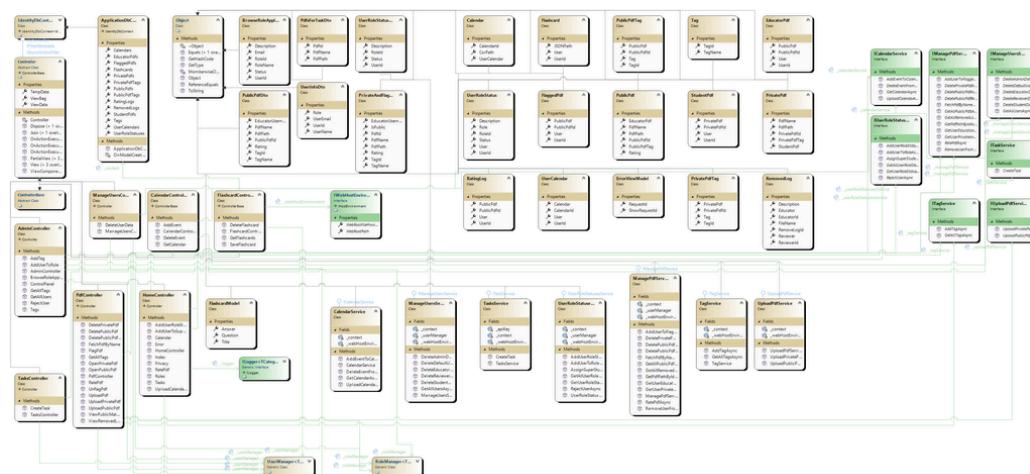
Property	Type	Description
UserId <b>(PK) (FK)</b>	text	Id korisnika koji se prijavio za neku ulogu.
RoleId <b>(FK)</b>	text	Id uloge za koju se korisnik prijavio.
Description	text	Tekstualni opis prijave.
Status	integer	Status prijave.

### Dijagram baze podataka

Ovdje se prikazuje dijagram baze podataka koja je realizirana za drugu i finalnu kontrolnu točku (vlastita slika).



## Dijagram razreda

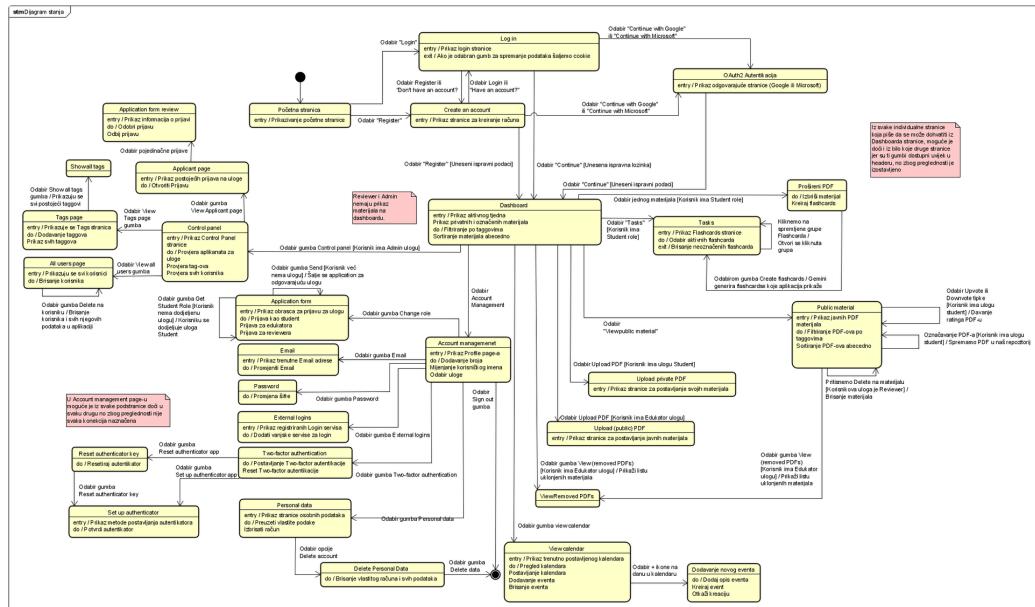


Dijagram razreda prikazuje razrede, njihova svojstva (eng. properties), metode (eng. methods) i odnosi među njima. U gornjem lijevom kutu dijagrama imamo najprije ApplicationDbContext, potom DTO-s (Data Transfer Object); UploadPdfDto, CreateTaskDto, PdfDto, TagDto, RatePdfDto, FlagPdfDto, UnflagPdfDto, DeletePdfDto, FetchPdfDto koje koristimo za enkapsulaciju podataka pri prijenosu istih između različitih slojeva aplikacije. U sredini gornjeg dijela dijagram su predstavljeni podatkovni entiteti kao što su PublicPdf, PrivatePdf, UserRoleStatus, Calendar, Flashcard... Oni definiraju podatkovne modele s pripadajućim svojstvima. Skroz gore desno su prikazana sučelja kao što su: ICalendarService, IManagePdfService, IUploadPdfService, koja definiraju metode definirane u odgovarajućim klasama. Donji lijevi kut prikazuje implementirane kontrolere; TasksController, PdfController, HomeController, CalendarController, AdminController koji obrađuju pripadajuće korisničke zahteve. Te u sredini donjeg dijela grafa imamo servise koji nam služe za implementiraju poslovnu logiku. Implementirani servsi su: CalendarService, ManagePdfService, ManageUserService, TagService, TasksService, UploadPdfService, UserRoleStatusesService.

# Dinamičko ponašanje aplikacije

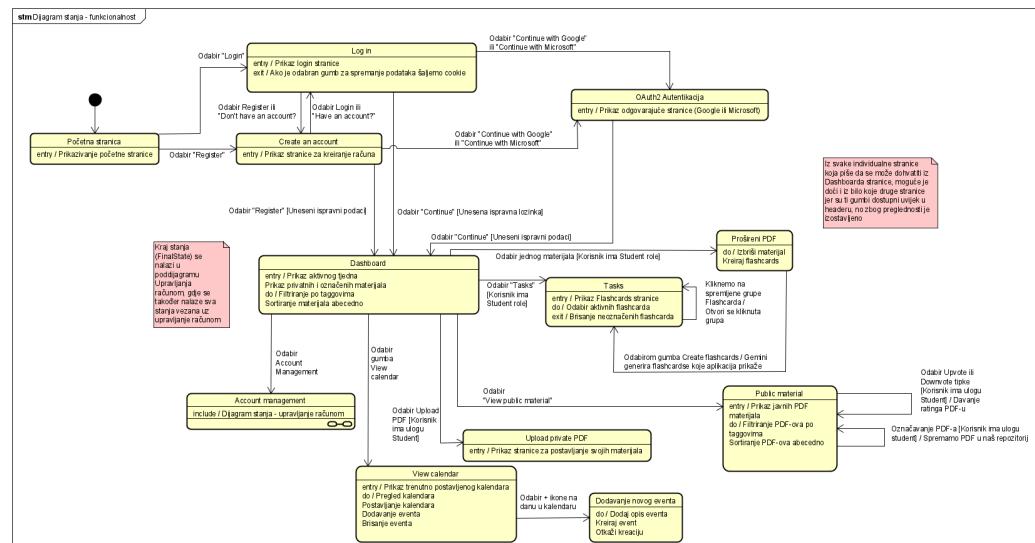
Ispod ovog poglavlja su postavljeni neki od važnijih dijagrama koji prikazuju dinamičko ponašanje aplikacije.

## Dijagram stanja



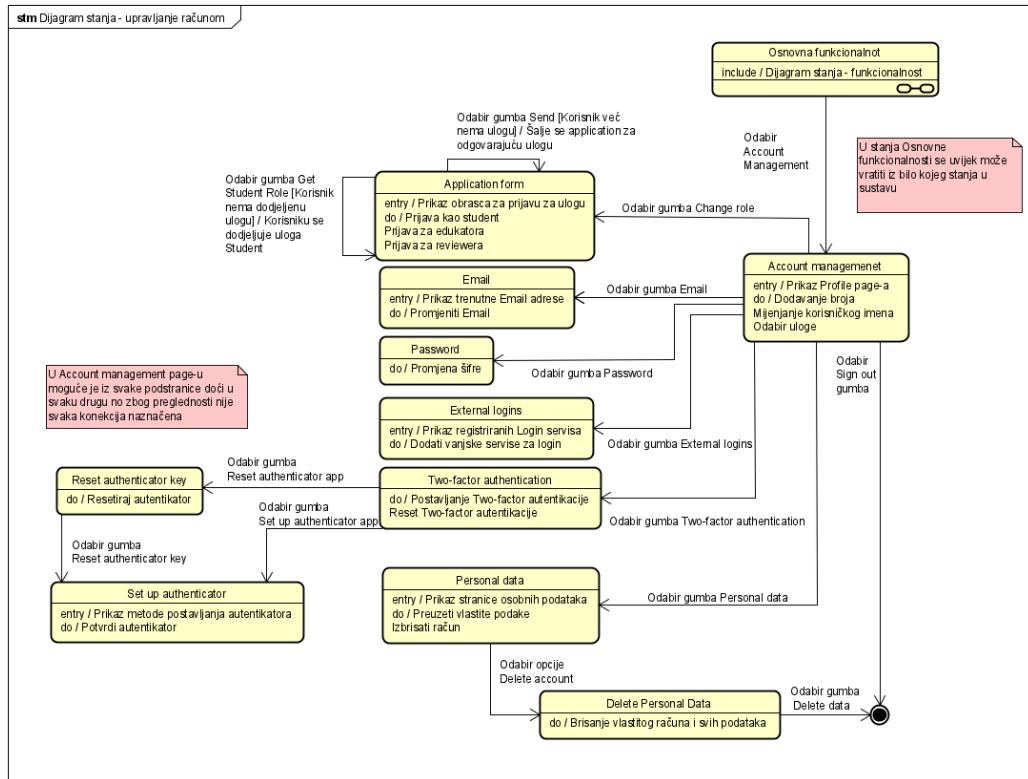
Dijagram stanja - cjelokupni pregled

## Dijagram stanja - osnovne funkcionalnosti



Dijagram stanja - osnovne funkcionalnosti

## Dijagram stanja - upravljanje računom

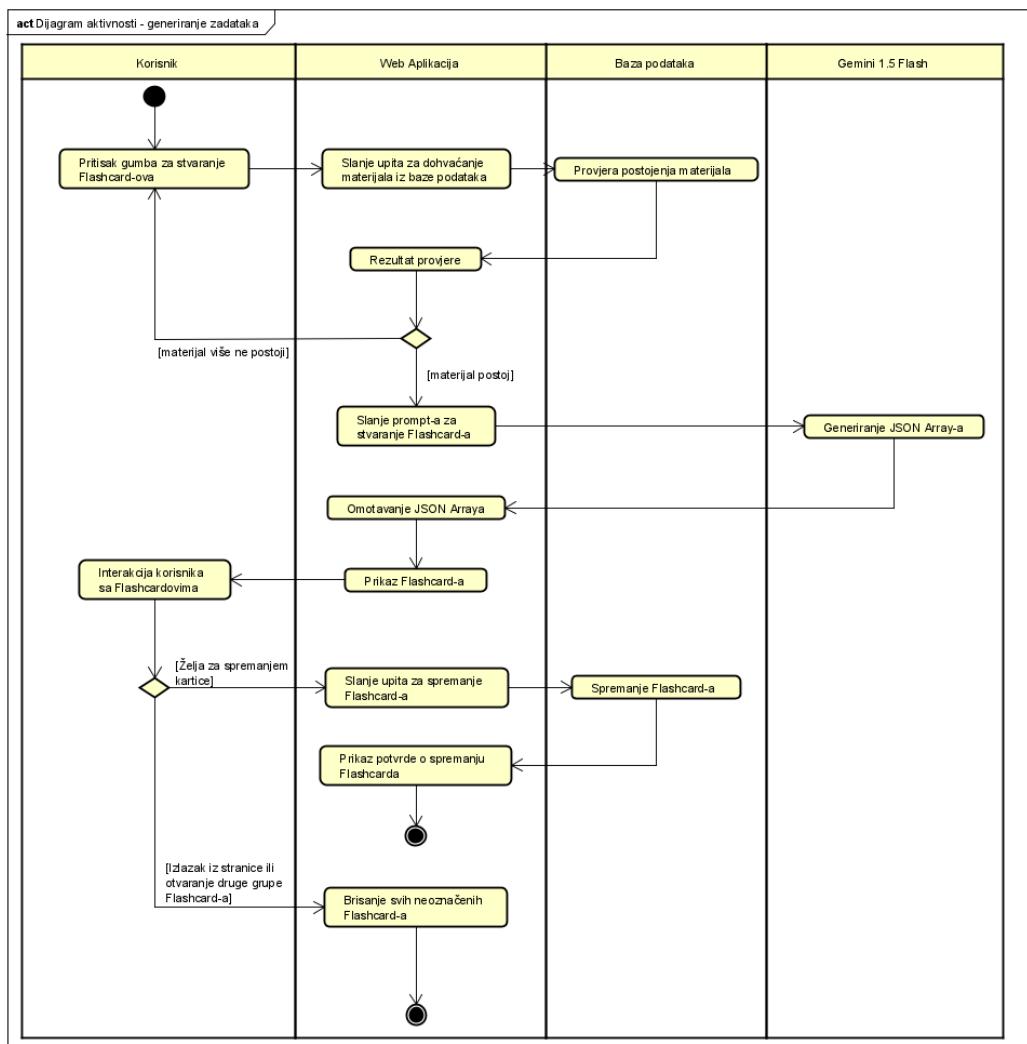


Dijagram stanja - upravljanje računom

## Dijagram aktivnosti - generiranje flashcarda

Za dijagram aktivnosti je izabrana najvažnija aktivnost oko koje se gradi cijela aplikacija, a to je stvaranje Flashcard-a za pomoć pri učenju. Prikazan je mogući tok i kako naš sustav reagira i ponaša se kada Student iskaže želju za stavarne Flashcard-a. Prikazana je interakcija sustava sa bazom podataka, te interakcija sustava sa Gemini-jom. Bitno je naglasiti da Gemini pri generiranju JSON Array-a u kojem su informacije u Flashcard-ovima, koristi informacije samo koje se nalaze u materijalu koji smo mu ponudili. U tu svrhu koristimo Gemini kao alat i ne želimo da ima mogućnost povući nekakve informacije koje nisu obuhvaćene u dokumentu.

Navedeno je i kako Student ima izbora odabrati one Flashcard-ove koje se njemu najviše sviđaju, te spremanje ih za ponovu reviziju kasnije.



Dijagram aktivnosti - generiranje Flashcard-a

Reference koje su se koristile u datumu pisanja ove stranice dokumentacije (13. 11. 2024):

- [Getting Started — OAuth](#)
- [Using OAuth and Cookies in Browser Based Apps | Best Practices | Curity](#)
- [Introduction to Identity on ASP.NET Core](#)
- [ASP.NET MVC Pattern | .NET](#)
- [What is monolithic architecture in software? | Definition from Tech...](#)
- [SQLite: System Tables](#)
- [ASP.NET Core Identity Tables](#)
- [What is EFMigrationsHistory table in EF core?](#)

Materijali s [Fakultet elektrotehnike i računarstva](#), kolegij „Programsko inženjerstvo“.

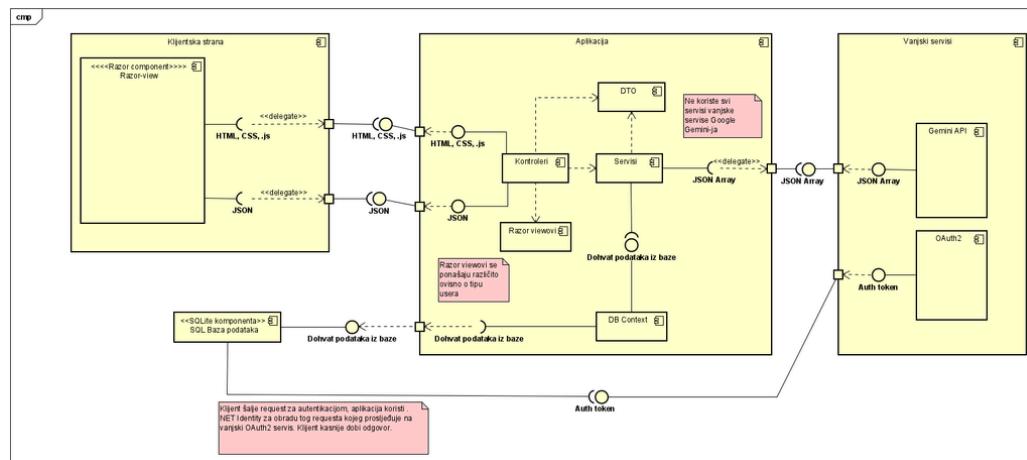
## 5. Arhitektura komponenata i razmještaja

Kako se aplikacija služi vanjskim komponentama, u svrhu predstavljanja i razjašnjavanja arhitekture priložena su dva dijagrama koja predstavljaju međuodnos komponenata aplikacije, korisnika i vanjskih sudionika koji pridonose ispravnom radu aplikacije.

U dijagramu komponenti je objašnjeno koja komponenta obavlja koju funkciju, te kako druge komponente se služe i pridonose cijelokupnom sistemu. Bitno je naglasiti da se razlikuju tri glavna djela sustava - onaj korisnički u kojem objašnjavamo rad prikaza aplikacije na korisnikovom sučelju, sama aplikacija u kojoj prikazujemo internu strukturu komponenata i njihov međuodnos, te vanjski servisi kojima integriramo povezanost našoj aplikaciji i ostvarujemo međusobnu raspodjelu podataka koji oni koriste i šalju nama nazad relevantne informacije, koje obrađuju u svojim sistemima, nama nepoznatim i nerelevantni internalno.

### Dijagram komponenata

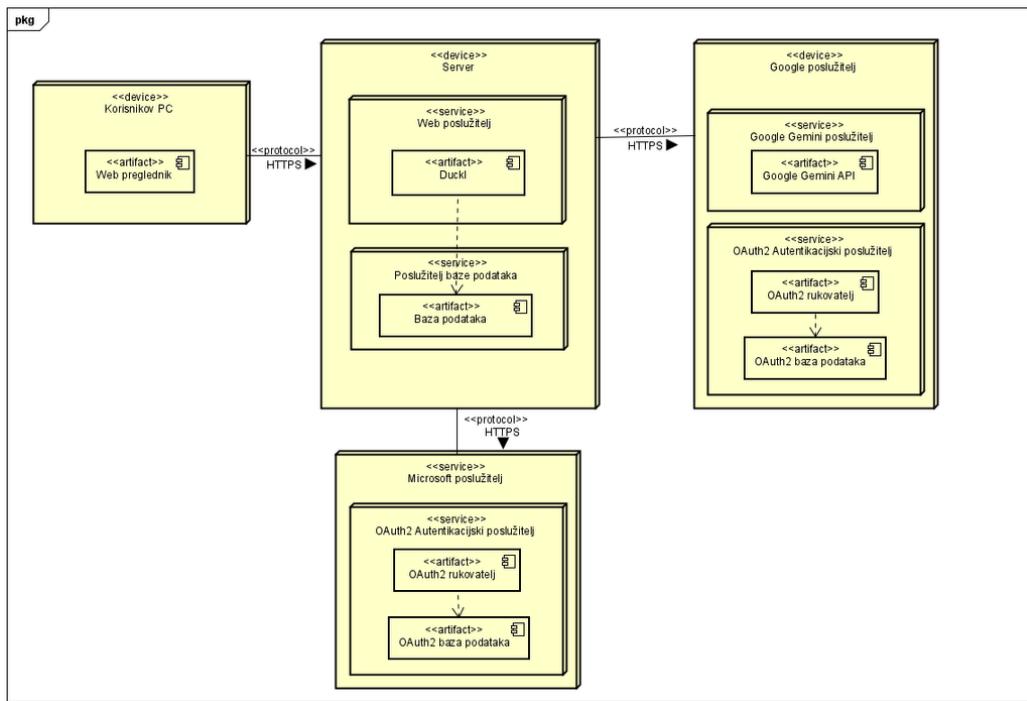
U dijagramu komponenti prikazujemo najvažnije nazne komponente, te se držimo jednostavnosti. Nije cilj prikazati sve, već one najbitnije cjeline - komponente koje nam ostvaruju rad aplikacije.



Slika 5.1: Dijagram komponenti

### Dijagram razmještaja

U dijagramu razmještaja definiramo 4 bitne cjeline. Posebno treba naglasiti da je rad realiziran preko HTTPS protokola, te je ta povezanost sa korisnikom i aplikacijom ostvarena. No uz to, preko HTTPS protokola se prenose i bitne OAuth2 informacije pomoću kojih ostvarujemo spajanje naših Google ili Microsoft računa sa aplikacijom. I uz sve to koristimo Gemini API koji se također nalazi u vanjskom uređaju, u Google-ovim serverima. Pri svakoj generaciji Flashcards-a preko veze se dijele informacije i dobivamo od samog Gemini-ja ono što ćemo prikazivati krajnjem korisniku, uz malu doradu.



Slika 5.2: Dijagram razmještaja

## 6. Ispitivanje programskog rješenja

### Ispitivanje programskog rješenja

Testiranje aplikacije izvodi se iz jednog file-a pod imenom TestSuite.cs, sastoji se od 11 automatiziranih testova pisanih uz Selenium, 6 testova komponenti te 5 testa sustava. Izvorni kod:

```
1  using NUnit.Framework;
2  using OpenQA.Selenium;
3  using OpenQA.Selenium.Chrome;
4  using OpenQA.Selenium.Support.UI;
5  using WebDriverManager;
6  using WebDriverManager.DriverConfigs.Impl;
7
8  namespace DuckI.Tests
9  {
10     [TestFixture]
11     class TestSuite
12     {
13         // Radimo usera, nema delete metode dobre pa se generatea random username, be warned runnanje testova n puta
14         // kreira n usera
15         private IWebDriver _driver;
16         private string _testEmail = $"testuser_{Guid.NewGuid()}@exampletest.com";
17         private string _testPassword = "Test@1234";
18
19         [SetUp]
20         public void SetUp()
21         {
22             new DriverManager().SetUpDriver(new ChromeConfig());
23             _driver = new ChromeDriver();
24         }
25
26         [TearDown]
27         public void TearDown()
28         {
29             _driver.Quit();
30         }
31
32         // lista testova, samo na dodavati testove koje zelite, order odreduje kojim se redoslijedom izvrsavaju
33         // TestOpeningPageDisplaysWelcomeMessage sluzi da provjeri ako se pocetni site uspjesno otvara
34         [Test, Order(1)]
35         public void TestOpeningPageDisplaysWelcomeMessage()
36         {
37             _driver.Navigate().GoToUrl("https://localhost:44385/");
38
39             var welcomeElement = _driver.FindElement(By.ClassName("display-4"));
40             Assert.That(welcomeElement.Text, Is.EqualTo("Welcome To DuckI!"));
41         }
42
43         // Updateana verzija TestOpeningPageDisplaysWelcomeMessage sluzi da provjeri ako se pocetni site uspjesno otvara
44
45         [Test, Order(1)]
46         public void TestOpeningPageDisplaysWelcomeMessageNew()
47         {
48             _driver.Navigate().GoToUrl("https://localhost:44385/");
49
50             var welcomeElement = _driver.FindElement(By.ClassName("text-primary"));
51             Assert.That(welcomeElement.Text, Is.EqualTo("Welcome to DuckI!"));
52         }
53
54         // TestRegisterUser sluzi za provjeru registracije
55         [Test, Order(2)]
56         public void TestRegisterUser()
57         {
58             _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Register");
59
60             var usernameField = _driver.FindElement(By.Id("Input_Email"));
61             var passwordField = _driver.FindElement(By.Id("Input_Password"));
62             var confirmPasswordField = _driver.FindElement(By.Id("Input_ConfirmPassword"));
63             var registerButton = _driver.FindElement(By.CssSelector("button[type='submit']"));
64
65             usernameField.SendKeys(_testEmail);
66             passwordField.SendKeys(_testPassword);
67             confirmPasswordField.SendKeys(_testPassword);
68             registerButton.Click();
69
70             Assert.That(_driver.Url, Is.EqualTo("https://localhost:44385/"));
71         }
}
```

```

72 // TestLogin provjerava login korisnika stvorenog sa TestRegisterUser
73 [Test, Order(3)]
74 public void TestLogin()
75 {
76     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Login");
77
78     var usernameField = _driver.FindElement(By.Id("Input_Email"));
79     var passwordField = _driver.FindElement(By.Id("Input_Password"));
80     var loginButton = _driver.FindElement(By.Id("login-submit"));
81
82     usernameField.SendKeys(_testEmail);
83     passwordField.SendKeys(_testPassword);
84     loginButton.Click();
85
86     TestContext.WriteLine($"Logging in with Email: {_testEmail}, Password: {_testPassword}");
87
88     Assert.That(_driver.Url, Is.EqualTo("https://localhost:44385/"));
89 }
90
91
92 // Test registracije za neispravan format emaila
93
94 [Test, Order(4)]
95 public void TestRegisterUserIncorrectEmail()
96 {
97     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Register");
98
99     var usernameField = _driver.FindElement(By.Id("Input_Email"));
100    var passwordField = _driver.FindElement(By.Id("Input_Password"));
101    var confirmPasswordField = _driver.FindElement(By.Id("Input_ConfirmPassword"));
102    var registerButton = _driver.FindElement(By.CssSelector("button[type='submit']"));
103
104    usernameField.SendKeys("invalidemail");
105    passwordField.SendKeys(_testPassword);
106    confirmPasswordField.SendKeys(_testPassword);
107    registerButton.Click();
108
109    var errorElement = _driver.FindElement(By.Id("Input_Email-error"));
110
111    Assert.That(errorElement.Text, Is.EqualTo("The Email field is not a valid e-mail address."));
112 }
113
114 // Test za ispitivanje public materials navigacijskog gumba
115 [Test, Order(5)]
116 public void TestPublicMaterialsButton()
117 {
118     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Login");
119
120     var usernameField = _driver.FindElement(By.Id("Input_Email"));
121     var passwordField = _driver.FindElement(By.Id("Input_Password"));
122     var loginButton = _driver.FindElement(By.Id("login-submit"));
123
124     usernameField.SendKeys(_testEmail);
125     passwordField.SendKeys(_testPassword);
126     loginButton.Click();
127
128     var publicMaterialsButton = _driver.FindElement(By.XPath("//a[@href='/Pdf/ViewPublicMaterial']"));
129
130     publicMaterialsButton.Click();
131
132     Assert.That(_driver.Url, Is.EqualTo("https://localhost:44385/Pdf/ViewPublicMaterial"));
133 }
134
135 // Test za potpun proces od logina do uploadanja pdf-a
136 [Test, Order(6)]
137 public void TestLoginAndUploadPdf()
138 {
139     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Login");
140
141     var usernameField = _driver.FindElement(By.Id("Input_Email"));
142     var passwordField = _driver.FindElement(By.Id("Input_Password"));
143     var loginButton = _driver.FindElement(By.Id("login-submit"));
144
145     usernameField.SendKeys(_testEmail);
146     passwordField.SendKeys(_testPassword);
147     loginButton.Click();
148
149     var userButton = _driver.FindElement(By.XPath("//a[@href='/Identity/Account/Manage']"));
150     userButton.Click();
151
152     var roleButton = _driver.FindElement(By.XPath("//a[@href='/Home/Roles']"));
153     roleButton.Click();
154 }
```

```

155
156     var studentRoleButton = _driver.FindElement(By.XPath("//button[text()='Get Student Role']"));
157     studentRoleButton.Click();
158
159     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Manage");
160     var logoutButton = _driver.FindElement(By.XPath("//button[text()='Sign out']"));
161     logoutButton.Click();
162
163     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Login");
164
165     var usernameFieldAggain = _driver.FindElement(By.Id("Input_Email"));
166     var passwordFieldAggain = _driver.FindElement(By.Id("Input_Password"));
167     var loginButtonAggain = _driver.FindElement(By.Id("login-submit"));
168
169     usernameFieldAggain.SendKeys(_testEmail);
170     passwordFieldAggain.SendKeys(_testPassword);
171     loginButtonAggain.Click();
172
173     var uploadPdfButton = _driver.FindElement(By.XPath("//a[@href='/Pdf/UploadPdf']"));
174     uploadPdfButton.Click();
175
176     var fileInput = _driver.FindElement(By.Id("file"));
177     var basePath = AppDomain.CurrentDomain.BaseDirectory;
178     var projectPath = Directory.GetParent(basePath).Parent.Parent.FullName;
179     var filePath = Path.Combine(projectPath, "Tests", "TestPDF.pdf");
180
181     fileInput.SendKeys(filePath);
182
183     var uploadButton = _driver.FindElement(By.XPath("//button[text()='Upload Private PDF']"));
184     ((IJavaScriptExecutor)_driver).ExecuteScript("arguments[0].click()", uploadButton);
185     Thread.Sleep(1000);
186     var confirmation = _driver.FindElement(By.XPath("//*[text()='File uploaded successfully.']"));
187     var confirmationText = ((IJavaScriptExecutor)_driver)
188         .ExecuteScript("return arguments[0].innerText;", confirmation).ToString();
189
190     Assert.That(confirmationText, Is.EqualTo("File uploaded successfully."));
191 }
192
193 // Test za upload kalendra
194 [Test, Order(7)]
195 public void TestFileUpload()
196 {
197     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Login");
198
199     var usernameField = _driver.FindElement(By.Id("Input_Email"));
200     var passwordField = _driver.FindElement(By.Id("Input_Password"));
201     var loginButton = _driver.FindElement(By.Id("login-submit"));
202
203     usernameField.SendKeys(_testEmail);
204     passwordField.SendKeys(_testPassword);
205     loginButton.Click();
206
207     var calendarButton = _driver.FindElement(By.XPath("//a[@href='/Home/Calendar']"));
208     calendarButton.Click();
209
210     var uploadCalendarButton = _driver.FindElement(By.XPath("//button[text()='Upload Calendar']"));
211     uploadCalendarButton.Click();
212
213     var fileInput = _driver.FindElement(By.Id("file"));
214     var basePath = AppDomain.CurrentDomain.BaseDirectory;
215     var projectPath = Directory.GetParent(basePath).Parent.Parent.FullName;
216     var filePath = Path.Combine(projectPath, "Tests", "TestingCalendar.csv");
217     var confirmCalendarButton = _driver.FindElement(By.XPath("//button[text()='Upload']"));
218
219     fileInput.SendKeys(filePath);
220     confirmCalendarButton.Click();
221
222     Assert.That(_driver.Url, Is.EqualTo("https://localhost:44385/"));
223 }
224
225 // Testing uploading something that isn't a pdf, calendar used as exsample
226 [Test, Order(8)]
227 public void TestLoginAndUploadNonPdf()
228 {
229     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Login");
230
231     var usernameField = _driver.FindElement(By.Id("Input_Email"));
232     var passwordField = _driver.FindElement(By.Id("Input_Password"));
233     var loginButton = _driver.FindElement(By.Id("login-submit"));
234
235     usernameField.SendKeys(_testEmail);
236     passwordField.SendKeys(_testPassword);
237     loginButton.Click();

```

```

238
239     var uploadPdfButton = _driver.FindElement(By.XPath("//a[@href='/Pdf/UploadPdf']"));
240     uploadPdfButton.Click();
241
242     var fileInput = _driver.FindElement(By.Id("file"));
243     var basePath = AppDomain.CurrentDomain.BaseDirectory;
244     var projectPath = Directory.GetParent(basePath).Parent.Parent.FullName;
245     var filePath = Path.Combine(projectPath, "Tests", "TestingCalendar.csv");
246
247     fileInput.SendKeys(filePath);
248     var uploadButton = _driver.FindElement(By.XPath("//button[text()='Upload Private PDF']"));
249     ((IJavaScriptExecutor)_driver).ExecuteScript("arguments[0].click();", uploadButton);
250     Thread.Sleep(1000);
251     var confirmation =
252         _driver.FindElement(
253             By.XPath("//*[text()='Error: Internal server error: Only PDF files are allowed.']"));
254     var confirmationText = ((IJavaScriptExecutor)_driver)
255         .ExecuteScript("return arguments[0].innerText;", confirmation).ToString();
256
257     Assert.That(confirmationText, Is.EqualTo("Error: Internal server error: Only PDF files are allowed."));
258 }
259
260 // Dodavanje eventa u kalendar putem pritiska na relevantni gumb
261 [Test, Order(9)]
262 public void TestAddEvent()
263 {
264     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Login");
265
266     var usernameField = _driver.FindElement(By.Id("Input_Email"));
267     var passwordField = _driver.FindElement(By.Id("Input_Password"));
268     var loginButton = _driver.FindElement(By.Id("login-submit"));
269
270     usernameField.SendKeys(_testEmail);
271     passwordField.SendKeys(_testPassword);
272     loginButton.Click();
273
274     var calendarButton = _driver.FindElement(By.XPath("//a[@href='/Home/Calendar']"));
275     calendarButton.Click();
276
277     var uploadCalendarButton = _driver.FindElement(By.XPath("//button[text()='Upload Calendar']"));
278     uploadCalendarButton.Click();
279
280     var fileInput = _driver.FindElement(By.Id("file"));
281     var basePath = AppDomain.CurrentDomain.BaseDirectory;
282     var projectPath = Directory.GetParent(basePath).Parent.Parent.FullName;
283     var filePath = Path.Combine(projectPath, "Tests", "TestingCalendar.csv");
284     var confirmCalendarButton = _driver.FindElement(By.XPath("//button[text()='Upload']"));
285
286     fileInput.SendKeys(filePath);
287     confirmCalendarButton.Click();
288
289     var calendarButtonTwo = _driver.FindElement(By.XPath("//a[@href='/Home/Calendar']"));
290     calendarButtonTwo.Click();
291
292     var element = _driver.FindElement(By.XPath("//div[@data-day='1']"));
293     Console.WriteLine(element);
294     element.Click();
295
296
297     var addInputDescription = _driver.FindElement(By.Id("eventDescription"));
298     var confirmButton = _driver.FindElement(By.CssSelector("button.btn.btn-success.text-light"));
299
300     addInputDescription.SendKeys("Test event");
301     confirmButton.Click();
302     Thread.Sleep(1000);
303     var firstEventTextDiv = _driver.FindElement(By.CssSelector("div.event-text"));
304     Assert.That(firstEventTextDiv.Text, Is.EqualTo("Test event"));
305 }
306
307 // Test deletanja novostvorenog usera
308 [Test, Order(10)]
309 public void TestDeleteUser()
310 {
311     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Login");
312
313     var usernameField = _driver.FindElement(By.Id("Input_Email"));
314     var passwordField = _driver.FindElement(By.Id("Input_Password"));
315     var loginButton = _driver.FindElement(By.Id("login-submit"));
316
317     usernameField.SendKeys("Adminko@Adminovic1.admin");
318     passwordField.SendKeys("Adminko@Adminovic1.admin");
319     loginButton.Click();
320

```

```

321     var controlPannel = _driver.FindElement(By.XPath("//a[@href='/Admin/ControlPanel']"));
322     controlPannel.Click();
323 
324     var listAllUsersButton = _driver.FindElement(By.XPath("//a[@href='/Admin/GetAllUsers']"));
325     listAllUsersButton.Click();
326 
327     var lastButton = _driver.FindElement(By.XPath("//button[contains(@class, 'btn btn-danger')][last()]"));
328     lastButton.Click();
329 
330     bool isEmailPresent = _driver.PageSource.Contains(_testEmail);
331 
332     Assert.That(isEmailPresent, Is.False, $"The email {_testEmail} was found on the screen after deletion.");
333 }
334 }
335 }
```

## Ispitivanje komponenti

Ispitivanje komponenti sastoji se od 6 test caseova, oni su redom:

### 1. TestOpeningPageDisplaysWelcomeMessage

ovaj test služio je kako bi se testiralo otvaranje originalne stranice, u međuvremenu početna stranica se promjenila pa je ovo jedini test koji returna fail.

```

1 public void TestOpeningPageDisplaysWelcomeMessage()
2 {
3     _driver.Navigate().GoToUrl("https://localhost:44385/");
4 
5     var welcomeElement = _driver.FindElement(By.ClassName("display-4"));
6     Assert.That(welcomeElement.Text, Is.EqualTo("Welcome To DuckI"));
7 }
```

### 2. TestOpeningPageDisplaysWelcomeMessageNew

ovaj test je zamijenio ulogu prethodnoga testa te testira otvaranje početne stranice sa trenutnom implementacijom koda.

```

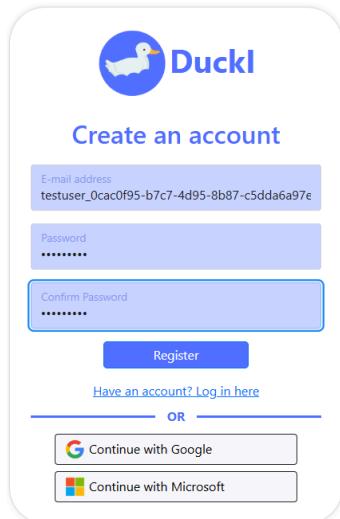
1 public void TestOpeningPageDisplaysWelcomeMessageNew()
2 {
3     _driver.Navigate().GoToUrl("https://localhost:44385/");
4 
5     var welcomeElement = _driver.FindElement(By.ClassName("text-primary"));
6     Assert.That(welcomeElement.Text, Is.EqualTo("Welcome to DuckI"));
7 }
```

### 3. TestRegisterUser

ovaj test je prvi test potreban za testiranje svega ostalog u aplikaciji, za pristup bilo čemu potrebno je registrirati korisnika u aplikaciji, ovaj test testira registriranje korisnika, te se nakon taj isti korisnik koristi u svakom testu.

```

1 public void TestRegisterUser()
2 {
3     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Register");
4 
5     var usernameField = _driver.FindElement(By.Id("Input_Email"));
6     var passwordField = _driver.FindElement(By.Id("Input_Password"));
7     var confirmPasswordField = _driver.FindElement(By.Id("Input_ConfirmPassword"));
8     var registerButton = _driver.FindElement(By.CssSelector("button[type='submit']"));
9 
10    usernameField.SendKeys(_testEmail);
11    passwordField.SendKeys(_testPassword);
12    confirmPasswordField.SendKeys(_testPassword);
13    registerButton.Click();
14 
15    Assert.That(_driver.Url, Is.EqualTo("https://localhost:44385/"));
16 }
```

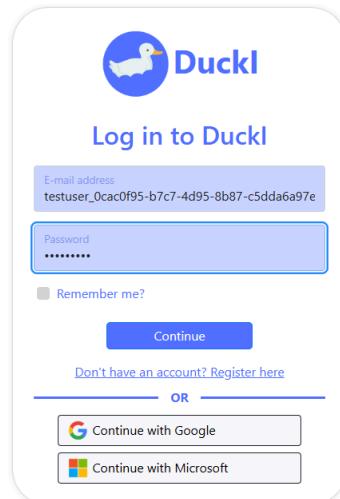


#### 4. TestLogin

ovaj test služi za testiranje prijave korisnika koji se u prethodnome testu registrirao, uzmu se isti podatci korišteni za registraciju te se unesu u komponente za prijavu.

```

1 public void TestLogin()
2 {
3     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Login");
4
5     var usernameField = _driver.FindElement(By.Id("Input_Email"));
6     var passwordField = _driver.FindElement(By.Id("Input_Password"));
7     var loginButton = _driver.FindElement(By.Id("login-submit"));
8
9     usernameField.SendKeys(_testEmail);
10    passwordField.SendKeys(_testPassword);
11    loginButton.Click();
12
13    TestContext.WriteLine($"Logging in with Email: {_testEmail}, Password: {_testPassword}");
14
15    Assert.That(_driver.Url, Is.EqualTo("https://localhost:44385/"));
16 }
```



#### 5. TestRegisterUserIncorectEmail

ovaj test je zamislen za testiranje ponašanja kada korisnik unese neki podatak koji nije ispravna e-pošta.

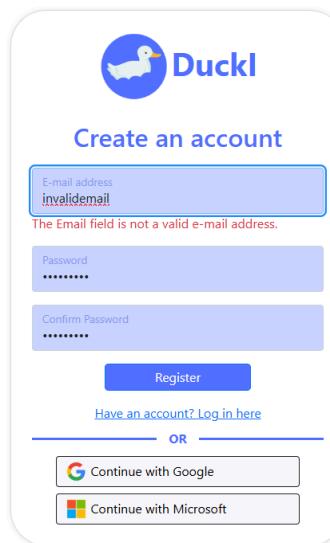
```

1 public void TestRegisterUserIncorectEmail()
2 {
3     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Register");
4
5     var usernameField = _driver.FindElement(By.Id("Input_Email"));
6     var passwordField = _driver.FindElement(By.Id("Input_Password"));
7     var confirmPasswordField = _driver.FindElement(By.Id("Input_ConfirmPassword"));
8     var registerButton = _driver.FindElement(By.CssSelector("button[type='submit']"));
```

```

9     usernameField.SendKeys("invalidemail");
10    passwordField.SendKeys(_testPassword);
11    confirmPasswordField.SendKeys(_testPassword);
12    registerButton.Click();
13
14    var errorElement = _driver.FindElement(By.Id("Input_Email-error"));
15
16    Assert.That(errorElement.Text, Is.EqualTo("The Email field is not a valid e-mail address."));
17
18 }

```



6. TestPublicMaterialsButton  
ovaj test testira jednu od komponenata na navigacijskoj traci, specifično "view public materials" komponentu.

```

1 public void TestPublicMaterialsButton()
2 {
3     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Login");
4
5     var usernameField = _driver.FindElement(By.Id("Input_Email"));
6     var passwordField = _driver.FindElement(By.Id("Input_Password"));
7     var loginButton = _driver.FindElement(By.Id("login-submit"));
8
9     usernameField.SendKeys(_testEmail);
10    passwordField.SendKeys(_testPassword);
11    loginButton.Click();
12
13    var publicMaterialsButton = _driver.FindElement(By.XPath("//a[@href='/Pdf/ViewPublicMaterial']"));
14
15    publicMaterialsButton.Click();
16
17    Assert.That(_driver.Url, Is.EqualTo("https://localhost:44385/Pdf/ViewPublicMaterial"));
18 }

```

[Duckl](#) [Tasks](#) [View public material](#) [View calendar](#) [Account management](#)

Welcome To Duckl

## Ispitivanje sustava

Ispitivanje sustava se sastoji od 5 testova.

### 1. TestLoginAndUploadPdf

Prvi od testova sustava ispitiva proces potreban da korisnik objavi pdf. Redoslijed je sljedeći:

Korisnik dolazi na aplikaciju, prijavljuje se putem login, unese svoje podatke te se prijavi, zatim prijavljeni korisnik odabire ulogu studenta kako bi imao pristup dijelu aplikacije za upload pdf-ova, zbog .NETa korisnik mora napraviti ponovni login kako bi se povukla njegova prava bazirana na ulozi. Kada se ovo dogodi odlazi se na upload pdf stranicu te se odabire pdf. Na kraju se očekuje poruka "File uploaded successfully. ".

Ulazi:

a. login podatci:

- i. `_testEmail`
- ii. `_testPassword`

b. pdf: `TestPDF.pdf`

Izlaz: Rezultat ispitivanja ( Assert.That(confirmationText, Is.EqualTo("File uploaded successfully.")); )

```
1  public void TestLoginAndUploadPdf()
2  {
3      _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Login");
4
5      var usernameField = _driver.FindElement(By.Id("Input_Email"));
6      var passwordField = _driver.FindElement(By.Id("Input_Password"));
7      var loginButton = _driver.FindElement(By.Id("login-submit"));
8
9      usernameField.SendKeys(_testEmail);
10     passwordField.SendKeys(_testPassword);
11     loginButton.Click();
12
13     var userButton = _driver.FindElement(By.XPath("//a[@href='/Identity/Account/Manage']"));
14     userButton.Click();
15
16
17     var roleButton = _driver.FindElement(By.XPath("//a[@href='/Home/Roles']"));
18     roleButton.Click();
19
20     var studentRoleButton = _driver.FindElement(By.XPath("//button[text()='Get Student Role']"));
21     studentRoleButton.Click();
22
23     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Manage");
24     var logoutButton = _driver.FindElement(By.XPath("//button[text()='Sign out']"));
25     logoutButton.Click();
26
27     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Login");
28
29     var usernameFieldAggain = _driver.FindElement(By.Id("Input_Email"));
30     var passwordFieldAggain = _driver.FindElement(By.Id("Input_Password"));
31     var loginButtonAggain = _driver.FindElement(By.Id("login-submit"));
32
33     usernameFieldAggain.SendKeys(_testEmail);
34     passwordFieldAggain.SendKeys(_testPassword);
35     loginButtonAggain.Click();
36
37     var uploadPdfButton = _driver.FindElement(By.XPath("//a[@href='/Pdf/UploadPdf']"));
38     uploadPdfButton.Click();
39
40     var fileInput = _driver.FindElement(By.Id("file"));
41     var basePath = AppDomain.CurrentDomain.BaseDirectory;
42     var projectPath = Directory.GetParent(basePath).Parent.Parent.FullName;
43     var filePath = Path.Combine(projectPath, "Tests", "TestPDF.pdf");
44
45     fileInput.SendKeys(filePath);
46
47     var uploadButton = _driver.FindElement(By.XPath("//button[text()='Upload Private PDF']"));
48     ((IJavaScriptExecutor)_driver).ExecuteScript("arguments[0].click();", uploadButton);
49     Thread.Sleep(1000);
50     var confirmation = _driver.FindElement(By.XPath("//*[text()='File uploaded successfully.']"));
51     var confirmationText = ((IJavaScriptExecutor)_driver)
52         .ExecuteScript("return arguments[0].innerText;", confirmation).ToString();
53
54     Assert.That(confirmationText, Is.EqualTo("File uploaded successfully."));
55 }
```

## Upload private PDF



For uploading PDFs and adding them to the private repository.

Tags make it easier for students to sort out the materials they need. Add relevant tags to your PDF using the drop table.

File uploaded successfully.

### 2. TestFileUpload

Ovaj test služi za provjeru procesa potrebnog za objavu csv kalendara. Koristi korisnika iz prijašnjeg testa sa ulogom studenta kako bi mogao objaviti svoj kalendar u sustav.

Uzaci:

- a. login podatci:
  - i. `_testEmail`

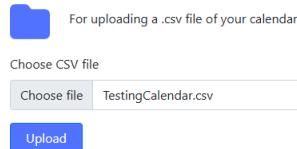
ii. `_testPassword`  
 b. pdf: `TestingCalendar.csv`  
 Izlaz: rezultat ispitivanja (`Assert.That(_driver.Url, Is.EqualTo("https://localhost:44385/"));`) (očekivano ponašanje funkcionalnosti je povratak na glavnu stranicu što se provjerava preko url-a)

```

1 public void TestFileUpload()
2 {
3     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Login");
4
5     var usernameField = _driver.FindElement(By.Id("Input_Email"));
6     var passwordField = _driver.FindElement(By.Id("Input_Password"));
7     var loginButton = _driver.FindElement(By.Id("login-submit"));
8
9     usernameField.SendKeys(_testEmail);
10    passwordField.SendKeys(_testPassword);
11    loginButton.Click();
12
13    var calendarButton = _driver.FindElement(By.XPath("//a[@href='/Home/Calendar']"));
14    calendarButton.Click();
15
16    var uploadCalendarButton = _driver.FindElement(By.XPath("//button[text()='Upload Calendar']"));
17    uploadCalendarButton.Click();
18
19    var fileInput = _driver.FindElement(By.Id("file"));
20    var basePath = AppDomain.CurrentDomain.BaseDirectory;
21    var projectPath = Directory.GetParent(basePath).Parent.Parent.FullName;
22    var filePath = Path.Combine(projectPath, "Tests", "TestingCalendar.csv");
23    var confirmCalendarButton = _driver.FindElement(By.XPath("//button[text()='Upload']"));
24
25    fileInput.SendKeys(filePath);
26    confirmCalendarButton.Click();
27
28    Assert.That(_driver.Url, Is.EqualTo("https://localhost:44385/"));
29 }

```

## Upload Calendar



### 3. `TestLoginAndUploadNonPdf`

Ovaj ispitni slučaj, sličan je prvoj (`TestLoginAndUploadPdf`) ali testira situaciju kada se u materijal pokuša staviti neki format koji aplikacija ne podržava.

Ulazi:

a. login podaci:

- i. `_testEmail`
- ii. `_testPassword`

b. pdf: `TestingCalendar.csv`

Izlaz: Rezultat ispitivanja (`Assert.That(confirmationText, Is.EqualTo("Error: Internal server error: Only PDF files are allowed."));`) aplikacija prepoznaje da se radi o ne podržanome formatu te vraća korisniku pogrešku.

```

1 public void TestLoginAndUploadNonPdf()
2 {
3     _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Login");
4
5     var usernameField = _driver.FindElement(By.Id("Input_Email"));
6     var passwordField = _driver.FindElement(By.Id("Input_Password"));
7     var loginButton = _driver.FindElement(By.Id("login-submit"));
8
9     usernameField.SendKeys(_testEmail);
10    passwordField.SendKeys(_testPassword);
11    loginButton.Click();
12
13    var uploadPdfButton = _driver.FindElement(By.XPath("//a[@href='/Pdf/UploadPdf']"));
14    uploadPdfButton.Click();
15
16    var fileInput = _driver.FindElement(By.Id("file"));
17    var basePath = AppDomain.CurrentDomain.BaseDirectory;
18    var projectPath = Directory.GetParent(basePath).Parent.Parent.FullName;
19    var filePath = Path.Combine(projectPath, "Tests", "TestingCalendar.csv");

```

```

20     fileInput.SendKeys(filePath);
21     var uploadButton = _driver.FindElement(By.XPath("//button[text()='Upload Private PDF']"));
22     ((IJavaScriptExecutor)_driver).ExecuteScript("arguments[0].click();", uploadButton);
23     Thread.Sleep(1000);
24     var confirmation =
25         _driver.FindElement(
26             By.XPath("//*[text()='Error: Internal server error: Only PDF files are allowed.']"));
27     var confirmationText = ((IJavaScriptExecutor)_driver)
28         .ExecuteScript("return arguments[0].innerText;", confirmation).ToString();
29
30     Assert.That(confirmationText, Is.EqualTo("Error: Internal server error: Only PDF files are allowed."));
31 }
32

```

## Upload private PDF



For uploading PDFs and adding them to the private repository.

TestingCalendar.csv

Tags make it easier for students to sort out the materials they need. Add relevant tags to your PDF using the drop table.

Error: Internal server error: Only PDF files are allowed.

#### 4. TestAddEvent

Ovaj test ispituje proces potreban za dodavanje događaja u kalendar ručno (tj. dodavanje događaja koji nije u csv datoteci.) test ispituje stariju verziju koja je zahtjevala upload nekog kalendaru kako bi se moglo pristupiti toj funkcionalnosti.

Ulazi:

- a. login podatci:
    - i. `_testEmail`
    - ii. `_testPassword`
  - b. pdf: `TestingCalendar.csv`
  - c. string sa eventom: "Test event"
- Izlaz: Rezultat ispitivanja ( `Assert.That(firstEventTextDiv.Text, Is.EqualTo("Test event"));` )

```

1  public void TestAddEvent()
2  {
3      _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Login");
4
5      var usernameField = _driver.FindElement(By.Id("Input_Email"));
6      var passwordField = _driver.FindElement(By.Id("Input_Password"));
7      var loginButton = _driver.FindElement(By.Id("login-submit"));
8
9      usernameField.SendKeys(_testEmail);
10     passwordField.SendKeys(_testPassword);
11     loginButton.Click();
12
13     var calendarButton = _driver.FindElement(By.XPath("//a[@href='/Home/Calendar']"));
14     calendarButton.Click();
15
16     var uploadCalendarButton = _driver.FindElement(By.XPath("//button[text()='Upload Calendar']"));
17     uploadCalendarButton.Click();
18
19     var fileInput = _driver.FindElement(By.Id("file"));
20     var basePath = AppDomain.CurrentDomain.BaseDirectory;
21     var projectPath = Directory.GetParent(basePath).Parent.Parent.FullName;
22     var filePath = Path.Combine(projectPath, "Tests", "TestingCalendar.csv");
23     var confirmCalendarButton = _driver.FindElement(By.XPath("//button[text()='Upload']"));
24
25     fileInput.SendKeys(filePath);
26     confirmCalendarButton.Click();
27
28     var calendarButtonTwo = _driver.FindElement(By.XPath("//a[@href='/Home/Calendar']"));
29     calendarButtonTwo.Click();
30
31     var element = _driver.FindElement(By.XPath("//div[@data-day='1']"));
32     Console.WriteLine(element);
33     element.Click();
34
35

```

```

36     var addInputDescription = _driver.FindElement(By.Id("eventDescription"));
37     var confirmButton = _driver.FindElement(By.CssSelector("button.btn.btn-success.text-light"));
38
39     addInputDescription.SendKeys("Test event");
40     confirmButton.Click();
41     Thread.Sleep(1000);
42     var firstEventTextDiv = _driver.FindElement(By.CssSelector("div.event-text"));
43     Assert.That(firstEventTextDiv.Text, Is.EqualTo("Test event"));
44 }

```

##### 5. TestDeleteUser

Posljednji test služi za ispitivanje funkcionalnosti brisanja korisnika, ovaj test koristi kredencijale admin usera kako bi se obrisali podaci korisnika stvorenog u ispitnoj aplikaciji.

Ulazi:

a. login podatci:

- i. \_testEmail : "Adminko@Adminovic1.admin"
- ii. \_testPassword : "Adminko@Adminovic1.admin"

Izlaz: Rezultat ispitivanja ( Assert.That(isEmailPresent, Is.False, \$"The email {\_testEmail} was found on the screen after deletion."); ) (provjerava nalazi li se obrisani korisnik na listi svih korisnika aplikacije)

```

1  public void TestDeleteUser()
2  {
3      _driver.Navigate().GoToUrl("https://localhost:44385/Identity/Account/Login");
4
5      var usernameField = _driver.FindElement(By.Id("Input_Email"));
6      var passwordField = _driver.FindElement(By.Id("Input_Password"));
7      var loginButton = _driver.FindElement(By.Id("login-submit"));
8
9      usernameField.SendKeys("Adminko@Adminovic1.admin");
10     passwordField.SendKeys("Adminko@Adminovic1.admin");
11     loginButton.Click();
12
13     var controlPannel = _driver.FindElement(By.XPath("//a[@href='/Admin/ControlPanel']"));
14     controlPannel.Click();
15
16     var listAllUsersButton = _driver.FindElement(By.XPath("//a[@href='/Admin/GetAllUsers']"));
17     listAllUsersButton.Click();
18
19     var lastButton = _driver.FindElement(By.XPath("//button[contains(@class, 'btn btn-danger')][last()]"));
20     lastButton.Click();
21
22     bool isEmailPresent = _driver.PageSource.Contains(_testEmail);
23
24     Assert.That(isEmailPresent, Is.False, $"The email {_testEmail} was found on the screen after deletion.");
25 }

```

## List of Users

User Name	User Email	Roles	Actions
testuser_42de25d0-e936-44b6-9e38-71d12f6f1141@exampletest.com	testuser_42de25d0-e936-44b6-9e38-71d12f6f1141@exampletest.com	Default User	<button>Delete</button>
testuser_1dc4fe54-5147-41ed-be4b-958dfec069fc@exampletest.com	testuser_1dc4fe54-5147-41ed-be4b-958dfec069fc@exampletest.com	Default User	<button>Delete</button>
testuser_3ef07fb2-24da-4528-a0ae-a0e8085cf185@exampletest.com	testuser_3ef07fb2-24da-4528-a0ae-a0e8085cf185@exampletest.com	Default User	<button>Delete</button>
testuser_gabe7dd6-e380-46f5-822f-32893dfdb8f7@exampletest.com	testuser_gabe7dd6-e380-46f5-822f-32893dfdb8f7@exampletest.com	Default User	<button>Delete</button>
testuser_cad61828-2241-4eb8-9c7b-f24d00105680@exampletest.com	testuser_cad61828-2241-4eb8-9c7b-f24d00105680@exampletest.com	Default User	<button>Delete</button>
Edukatorko@edukatoric69.eu	Edukatorko@edukatoric69.eu	Educator	<button>Delete</button>
EdukatorkoEdukatovic3@skole.hr	EdukatorkoEdukatovic3@skole.hr	Educator	<button>Delete</button>

### Rezultati ispitivanja:

Rezultati su kao što su i očekivani, svaki od navedenih testova prolazi osim `TestOpeningPageDisplaysWelcomeMessage` za koji se očekuje pad pošto testira prikaz elementa koji više ne postoji u aplikaciji. Isti taj test i vraća poruku da ne postoji element koji pokušavamo dohvatiti.

```

D Duck (11 tests) Failed: 1 test failed
  D Duck Tests (11 tests) Failed: 1 test failed
    TestSuite (11 tests) Failed: One or more child tests had errors: 1 test failed
      TestAddEvent Success
      TestDeleteUser Success
      TestFileUpload Success
      TestHome Success
      TestLoginAndLogoutPDF Success
      TestLogoutAndLogoutPDF Success
      TestOpeningPageDisplaysWelcomeMessage Failed: OpenQA.Selenium.NoSuchElementException: no such element: Unable to locate element: {"method":"css selector","selector":"display:4"}
      TestOpeningPageDisplaysWelcomeMessageNew Success
      TestPublicMaterialsButton Success
      TestRegisterUser Success
      TestRegisterUserIncorrectEmail Success

```

## 7. Tehnologije za implementaciju aplikacije

Deployment:

- Provider: Microsoft Azure
- Server: Ubuntu server
- Reverse proxy: Nginx
- Application host: Systemd

Frontend, backend i project management:

- IDE: Rider: The Cross-Platform .NET IDE from JetBrains
  - Rider je IDE koji koriste programeri koji se bave radom na cijelom .NET technology stack-u, kao i oni koji se bave razvojem videoigara.
  - [Rider: The Cross-Platform .NET IDE from JetBrains](#)
- Backend i frontend: [ASP.NET Core | Open-source web framework for .NET](#), uz Razor stranice, JavaScript i Bootstrap te C#.
  - [ASP.NET Core | Open-source web framework for .NET](#) je server-side framework za izradu dinamičkih web stranica. Ducki koristi [ASP.NET Core | Open-source web framework for .NET](#) MVC koja omogućuje izradu web stranica pomoću Model-View-Controller design patterna.
  - Razor je [ASP.NET Core | Open-source web framework for .NET](#) template markup sintaksa koja se koristi za izradu dinamičkih web stranica pomoću programskog jezika C#.
  - JavaScript je objektno orijentirani skriptni jezik koji se koristi za interaktivnost web stranica. Npr. skočni prozori, gumbi na koje se može kliknuti, animacije itd.
  - Bootstrap je HTML, CSS te JavaScript library kojemu je cilj pojednostavljenje razvijanja responzivnog korisničkog sučelja web stranica. Sadrži design templates za button, navbar i slične komponente sučelja.
  - C# je programski jezik je objektno orijentirani jezik koji ima široku upotrebu u različitim područjima. Ovaj projekt koristi verziju 6.0.
  - [Introduction to Razor Pages in ASP.NET Core](#)
  - [What is ASP.NET? | .NET](#)
  - [Introduction - JavaScript | MDN](#)
  - [Bootstrap](#)
- Project management: Jira i Confluence
  - Jira omogućuje praćenje bugova, issue-a i upravljanje projekata na agilan način. Odličan je alat za project management.
  - Confluence je corporate wiki. Naširoko se koristi za kreiranje, dijeljenje i upravljanje sadržajem unutar razvojnog tima. Confluence se također naširoko koristi i za projektnu dokumentaciju.
  - [Jira | Issue & Project Tracking Software | Atlassian](#)
  - [Confluence | Your Remote-Friendly Team Workspace | Atlassian](#)
- Testovi: NUnit 4.3.2 i Selenium 4.28.1.

## 8. Upute za puštanje u pogon

Za deployment aplikacije, koristi se provider Microsoft Azure, na kojem je pokrenuta besplatna B1 virtualna mašina sa Ubuntu serverom. Na sustav je najprije potrebno instalirati dotnet i dotnet-ef kako bi se aplikacija mogla izgraditi (sam proces instaliravanja *proprietary* Microsoft alata na linux sistem se može zakomplicirati, te njihova instalacija neće biti pokrivena).

Izvorni kod aplikacije može se preuzeti preko Git-a, pokretanjem komande poput:

```
1 git clone -b test/DevOps-Testing https://github.com/Gumene-Patkice/DuckI.git
```

Nakon što se preko naredbenog retka pomakne u direktorij izvornog koda aplikacije, u datotekama koje koriste API ključeve, lokalne varijable se trebaju zamijeniti za pravim API ključevima:

- Program.cs
  - builder.Configuration["Authentication:Google:Client"]
  - builder.Configuration["Authentication:Google:ClientSecret"]
  - builder.Configuration["Authentication:Microsoft:Client"]
  - builder.Configuration["Authentication:Microsoft:ClientSecret"]
- TasksService.cs
  - configuration["Key:Gemini"]

Zatim se mora izgraditi aplikacija i primijeniti migracije nad bazom podataka:

```
1 dotnet build
2 dotnet ef database update
```

Kako bi dalje nastavili, moramo imati ispravno postavljeni Nginx reverse proxy kako bi korisnici mogli pristupiti našoj aplikaciji iz javne domene. Detaljne upute mogu se pronaći na: [How To Install Nginx on Ubuntu 22.04 | DigitalOcean](#)

Isto tako, ako se želi postaviti HTTPS na poslužitelju, mora se kupiti domena, izdati certifikat, te dodatno konfigurirati Ngnix, detaljne upute mogu se pronaći na:

[How To Secure Nginx with Let's Encrypt on Ubuntu | DigitalOcean](#)

Nakon što je sve ispravno postavljeno, aplikacija se može izvesti u Nginx registrirani direktorij:

```
1 sudo dotnet publish -c Release -o /var/www/ducki.space/DuckI --runtime linux-x64
```

Aplikaciju sada možemo pokrenuti kao *systemd* servis:

```
1 nano /etc/systemd/system/app.service
```

Ta datoteka izgleda kao:

```
1 [Unit]
2 Description=DuckI application for Program Engineering class.
3
4 [Service]
5 WorkingDirectory=/var/www/ducki.space/DuckI/
6 ExecStart=/usr/bin/dotnet /var/www/ducki.space/DuckI/DuckI.dll
7 Restart=always
8 # Restart service after 10 seconds if the dotnet service crashes:
9 RestartSec=10
```

```
10 KillSignal=SIGINT
11 SyslogIdentifier=DuckI-App
12 User=root
13 Environment=ASPNETCORE_ENVIRONMENT=Production
14
15 [Install]
16 WantedBy=multi-user.target
```

Kreiranjem novog servisa, trebamo ponovno pokrenuti *systemctl daemon*, te pokrenuti novo kreirani servis:

```
1 systemctl daemon-reload
2 systemctl start app
```

Konfiguracija za Nginx može se pronaći sa:

```
1 sudo nano /etc/nginx/conf.d/app.conf
```

Ta datoteka izgleda kao:

```
1 server {
2     root /var/www/ducki.space/DuckI;
3     server_name ducki.space www.ducki.space;
4
5     location / {
6         proxy_pass http://localhost:5000/;
7         proxy_http_version 1.1;
8         proxy_set_header Upgrade $http_upgrade;
9         proxy_set_header Connection keep-alive;
10        proxy_set_header Host $host;
11        proxy_cache_bypass $http_upgrade;
12        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
13        proxy_set_header X-Forwarded-Proto $scheme;
14    }
15
16    listen 443 ssl; # managed by Certbot
17    ssl_certificate /etc/letsencrypt/live/ducki.space/fullchain.pem; # managed by Certbot
18    ssl_certificate_key /etc/letsencrypt/live/ducki.space/privkey.pem; # managed by Certbot
19    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
20    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot
21
22
23 }
24 server {
25     if ($host = www.ducki.space) {
26         return 301 https://$host$request_uri;
27     } # managed by Certbot
28
29
30     if ($host = ducki.space) {
31         return 301 https://$host$request_uri;
32     } # managed by Certbot
33
34
35     listen 80;
36     server_name ducki.space www.ducki.space;
37     return 404; # managed by Certbot
38
39 }
```

Ponovno pokrećemo Nginx servis preko komande:

```
1 sudo systemctl restart nginx
```

Tijekom svakog novog deploy-a, ne treba se napraviti novi *systemd* servis, već se samo može resetirati preko:

```
1 sudo systemctl restart app
```

Ako je potrebno pregledati aplikacijske logove, može se to napraviti preko komande:

```
1 sudo journalctl -u app -r
```

## 9. Zaključak i budući rad

Projekt se radio kroz nekoliko mjeseci, počevši od 10. mjeseca 2024. pa do 1. mjeseca 2025. Zahtjevni tehnički izazovi su bila integracija Gemini 1.5 Flash modela i upotreba Geminija za generiranje flashcarda. Taj se izazov riješio čitanjem stringa iz PDF dokumenta te slanjem tog stringa na obradu Gemini Flash modelu. Još jedan od tehničkih izazova je bilo i provedba puštanje aplikacije u pogon, kao i implementacija OAuth2. Taj smo problem riješili redizajniranjem arhitekture sustava i uporabom [ASP.NET](#) MVC arhitekture. Još jedan od izazova je bio način spremanje PDF, CSV i JSON datoteka na sustav. Taj smo izazov riješili spremanjem tih dokumenata u datotečni sustav aplikacije te spremanjem putanja na te datoteke unutar same baze podataka.

Sudionici projekta su stekli znanja o programskom jeziku C#, [ASP.NET](#) framework-u i upravljanje projektima, a znanje [ASP.NET](#) framework-a bi bilo korisno imati unaprijed kako bi se brže i kvalitetnije ostvario projekt.

## A. 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, [Unified Modeling Language \(UML\) description, UML diagram examples, tutorials and reference for all types of UML diagrams - use case diagrams, class, package, component, composite structure diagrams, deployments, activities, interactions, profiles, etc.](#)
6. Astah Community, [Powerful and Fast UML Diagramming Software - Astah](#)
7. [Getting Started — OAuth](#)
8. [Using OAuth and Cookies in Browser Based Apps | Best Practices | Curity](#)
9. [Introduction to Identity on ASP.NET Core](#)
10. [ASP.NET MVC Pattern | .NET](#)
11. [What is monolithic architecture in software? | Definition from Tech...](#)
12. [SQLite: System Tables](#)
13. [ASP.NET Core Identity Tables](#)
14. [What is EFMigrationsHistory table in EF core?](#)
15. [Introduction to Razor Pages in ASP.NET Core](#)
16. [What is ASP.NET? | .NET](#)
17. [Jira | Issue & Project Tracking Software | Atlassian](#)
18. [Confluence | Your Remote-Friendly Team Workspace | Atlassian](#)
19. Materijali s [Fakultet elektrotehnike i računarstva](#), kolegij „Programsko inženjerstvo“.

## A. Dnevnik promjena dokumentacije

<b>Rev.</b>	<b>Opis promjene/dodataka</b>	<b>Autori</b>	<b>Datum</b>
0.1	Napravljen predložak dokumentacije i inicijalizacija repozitorija	Leo Marušić	10.10.2024
0.2	Postavljanje timskih uloga i općih ciljeva; inicijalna struktura aplikacije	Filip Belina	10.10.2024
0.3	Definiranje projekta Ducki - AI asistent za učenje; razmatranje tehnologija: ReactJS za frontend, <a href="#">ASP.NET Core   Open-source web framework for .NET</a> za backend, PostgreSQL za bazu podataka	Cjeli tim	10.10.2024
0.4	Demo aplikacija: kreiran inicijalni UI i povezivanje s PostgreSQL bazom podataka	Jakov Lovaković	25.10.2024
0.5	Standardizacija procesa razvoja: uspostava pravila za ticketing u JIRA-i, kreiranje grana, pull request standardi	Filip Belina, Leo Marušić	25.10.2024
0.6	Izrada Use-Case i sekvensijskih dijagrama za funkcionalnosti poput upload materijala, registracija, dodavanje kalendarja, kopirani confluence pagevi na wiki	Filip Belina, Mislav Marinović, Leo Marušić, Jakov Lovaković	28.10.2024
0.7	Project redesign, implementiran Oauth2	Leo Marušić	5.11.2024
0.8	Napravljena prva verzija kalendar-a	Jakov Lovaković	7.11.2024
0.9	Napravljen globalni error handler	Jan Lalić	11.11.2024

0.10	Napravljen izgled kalendara za predaju	Martin Šainčević	12.11.2024
0.11	Mali bug fix za register vezan uz registraciju novog korisnika ako je korisnik trenutno ulogiran	Jakov Lovaković	13.11.2024
0.12	Dodan PDF dokumentacije	Filip Belina	13.11.2024
0.13	Deployana aplikacija	Leo Marušić	14.11.2024
0.14	<ul style="list-style-type: none"> <li>• Added Google Gemini text prompting</li> <li>• Fixed calendar working on Windows/Linux systems</li> <li>• Calendar directory is now being created if it does not exist</li> <li>• Removed the redundant email confirmation from the user registration process</li> <li>• User will now automatically login after registration</li> <li>• User can now set an username (had to make a custom sign-in manager for that... wonderful)</li> <li>• Also updated .gitignore to include user data files</li> </ul>	Leo Marušić	30.11.2024
0.15	Napravljena stranica za kalendar i poboljšan njegov izgled	Martin Šainčević	10.12.2024
0.16	<ul style="list-style-type: none"> <li>• implementiran dizajn za sve</li> </ul>	Jan Badel	12.12.2024

	<p>stranice pod rutom /Identity/Account/</p> <ul style="list-style-type: none"> <li>• implementiran dizajn za sve stranice pod rutom /Identity/Account/ Manage/</li> <li>• implementiran dizajn za header i footer</li> <li>• Home i Privacy buttons su maknuti, Home je accessible preko logtipa, a privacy preko footera</li> <li>• resend email confirmation se sada nalazi na register screenu</li> <li>• CSS se nalazi u zasebnim fileovima</li> <li>• custom.css je nadodani file u kojem se overrideaju neke Bootstrap klase (vezane uz boje)</li> <li>• buttoni veiw material (za sada beskoristan)/view calendar/chat se sada nalaze u _LoginPartial pageu</li> </ul>		
0.17	<p>Dodata nova tablica UserRoleStatuses. Na ruti /Home/Roles moguće je pristupiti testnom sučelju za dodavanje roleova (korišteno pri provjeri ispravnosti implementacije). Testno sučelje može koristiti frontend tim da vide kako se implementira backend dio ove funkcionalnosti u</p>	Jan Lalić, Jakov Lovaković	2. 12. 2024.

	frontend, ili ga može redizajnirati i ostaviti kakav je. Popravljene manjih stvari (dodata autorizacija na određene rute, promijenjeno ime CalendarController klase kako bi kod bio konzistentniji).		
0.18	Upravljanje ulogama kao admin.	Jan Lalić, Jakov Lovaković	8. 12. 2024.
0.19	Izvadio sam javascript i css dijelove koda iz Index.cshtml te stavio to u dvije odvojene datoteke. Stvorio sam novi view Calendar.cshtml te uz pomoć Jakova napravio da se može otići na tu stranicu kroz nav bar. Također sam napravio odvojeni index.js koji je skoro identičan calendar.js a koji će u budućnosti mijenjati za ostvarivanje malo drukčijeg kalendara na home stranici. Home i Calendar stranice koriste isti prijašnje navedeni css dokument. Home stranicu će dalje mijenjati kad se ostvare funkcionalnosti koje ćemo prikazivati na dashboardu na toj stranici. Nakon uploada novog csv dokumenta za kalendar korisnik će automatski preusmjeren na Home stranicu što smatram da nije	Martin Šainčević	10. 12. 2024.

	problem no htio sam samo napomenuti.		
0.20	Dodan SuperStudent role i način za davanje tog rolea korisniku.	Jan Lalić, Jakov Lovaković	12. 12. 2024.
0.21	Dodano uploadanje PDF-ova za edukatora i studenta. Ruta za upload je pdf/uploadpdf. Dodana lokalizacija. Dodano postavljanje tagova i njihovo pregledavanje.	Jan Lalić, Jakov Lovaković	12. 12. 2024.
0.22	Dodano browseanje flagganih i private PDF-ova (na index page, i za superstudenta i educatora, samo sto educator ima samo svoje, on nema opcije unflagganja s te stranice pošto ni nemože flaggat), browseanje i flagganje public pdfova, unflagganje public materiala s index pagea, otvaranje pdfova	Jan Lalić, Jakov Lovaković	14. 12. 2024.
0.23	Dodana je mogućnost upvotanja i downvotanja public pdfova. Dodana mogućnost removanja pdfova; student može svoje privatne pdfove maknuti, educator svoje i reviewer može educatorove pdfove maknuti uz opis zašto je maknut. Edukator ima reportlog u kojemu može vidjeti	Jan Lalić, Jakov Lovaković	18. 12. 2024.

	koji su mu kanuti pdfovi.		
0.24	<ul style="list-style-type: none"> <li>• implementiran dizajn za Roles page</li> <li>• implementiran dizajn za BrowseRoleApplication page</li> <li>• veći font za izbornik stranica Account Management</li> <li>• vidljive Role u Profile pageu</li> </ul>	Jan Badel	19. 12. 2024.
0.25	Implementirane finalne funkcionalnosti za kalendar vezane uz backend	Jan Lalić, Jakov Lovaković	2.1.2025
0.26	Dodana notifikacija za upozorenje o uploadu istoimenog pdf-a	Jakov Lovaković	2.1.2025
0.27	Implementiran dizajn za stvari povezane sa pdf-ovima	Jan Badel	3.1.2025
0.28	Implementirano brisanje korisnika i njihovih podataka	Jan Lalić, Leo Marušić, Jakov Lovaković	4.1.2025
0.29	<p>Dodana podrška za testiranje uz pomoć Selenium-a.</p> <p>Dodano par testova za primjer</p>	Filip Belina	8.1.2025
0.30	Implementirana izrada flashcardova u backendu pomoću Gemini api-a	Leo Marušić	12.1.2025
0.31	Dovršen ostatak funkcionalnosti na frontendu vezanih uz kalendar	Jan Badel	14.1.2025
0.32	Implementiran novi dizajn za poruke uspjeha i neuspjeha u aplikaciji te uklonjeni nepotrebni alertovi	Jan Badel	14.1.2025

0.33	Implementiran frontend dio vezan uz generaciju flashcardova	Martin Šainčević	14.1.2025
0.34	Dodan default kalendar za svakog korisnika kako bi mogao pristupiti dodavanju događaja u kalendar	Jakov Lovaković	19.1.2025
0.35	Dodana manja poboljšanja korisničkog sučelja na frontendu	Jan Badel	21.1.2025
0.36	Rješen bug koji nije pravilno dao rad sa duplikatnim karticama	Martin Šainčević	23.1.2025
0.37	Riješen bug u kojem Edukator nije mogao pristupiti svom PDF-u	Jakov Lovaković	23.1.2025
0.38	Dodani svi automatizirani testovi	Filip Belina	23.1.2025
0.39	Dodane završne opaske u kod kako bi bilo spremno za predaju	Jakov Lovaković, Jan Badel	23.1.2025
0.40	Zadnji popravak frontend bug-a prije deploya	Jan Badel	24.1.2025
1.0	Deployana finalna verzija aplikacije	Leo Marušić	24.1.2025

## B. Prikaz aktivnosti grupe

### Dnevnik Sastajanja

Svi sastanci dodani su kao svoji page-evi radi organizacije u ovaj page na Confluencu. (Ako se ovo gleda na wikiju sastanci su na kraju izlistani kao dodani page-evi)

### Plan rada

Za ostvarivanje plana rada korišten je Jira ticketing sustav. Sve za prvu predaju bit će realizirano unutar jednog sprinta, dok će sve za drugu predaju biti unutar drugoga. U praksi trajanje sprinta je bliže 3 tjedna nego 6 no u svrhu organizacije i bez nepotrebnih komplikacija smatrali smo da je najbolje podijeliti u 2 sprinta sve.

### Tablica aktivnosti

Aktivnosti su vidljive u tablici dolje prvi stupac predstavlja imena aktivnosti, dok svaki drugi broj sati koja je određena osoba napravila na toj aktivnosti.

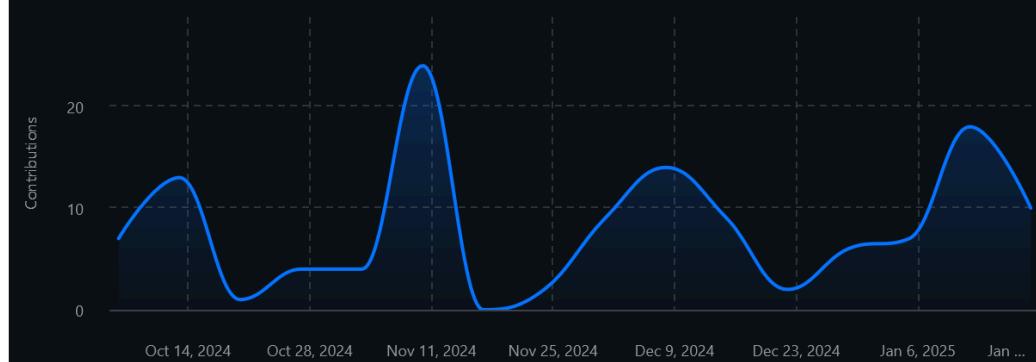
	<b>Belina Filip</b>	Marušić Leo	Marinović Mislav	Lovaković Jakov	Lalić Jan	Badel Jan	Šainčević Martin
Upravljanje projektom	20	5		2			
Opis projektnog zadatka	2						
Funkcionalni zahtjevi	2		3				
Opis pojedinih obrazaca			4				
Dijagram obrazaca			7				
Sekvencijski dijagrami	4	4	10	4			
Opis ostalih zahtjeva			1				
Rad na dokumentaciji	4	1	12	9.5	1		
Arhitektura i dizajn sustava	8			2			
Baza podataka		4		24			
Dijagram razreda				2	4		
Dijagram stanja			6				
Dijagram aktivnosti			4				
Dijagram komponenti			5		4		
Korištene tehnologije i alati	2						
Ispitivanje programskog rješenja	20			4	4		

Dijagram razmještaja			3				
Upute za puštanje u pogon		1					
Dnevnik sastajanja	2						
Zaključak i budući rad	0.5		0.25				
Popis literature			0.25				
Implementacija Oauth2		20		12			
Pisanje dev guidelineova	2						
Učenje tehnologija	8	10	8	16	16	16	16
Sastanci	24	22	18	24	19	15	15
Pregledavanje pull requestova	4	1		1			
Bugfixing	2	2		3			
Kalendar				8	4		12
Depolyment		24					
Home page						16	9
Modeliranje izgleda aplikacije			40				
Exception handeler					8		
Pisanje backend koda		28		82.4	81.6		
Pisanje frontend koda	2					60.4	48
Automatsko testiranje	32						
Ukupno	138.5	122	121.5	193.9	141.6	107.4	100

## Dijagram pregleda promjena

## Commits over time

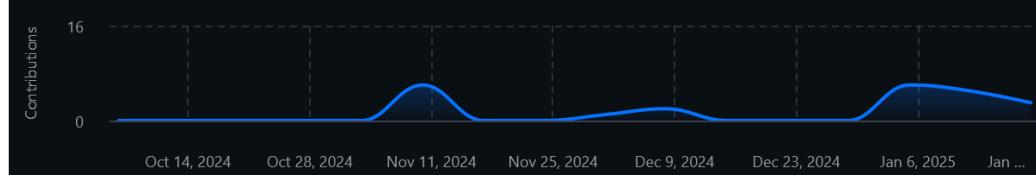
Weekly from Oct 6, 2024 to Jan 19, 2025



## mimarinovic's Commits



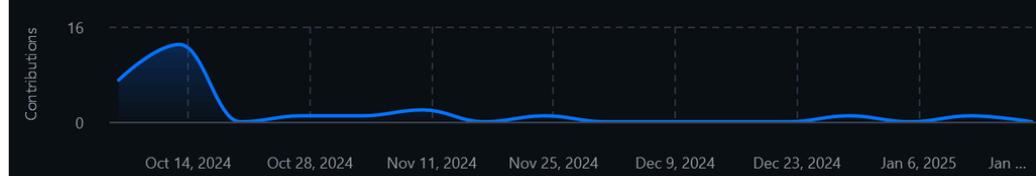
## martinsaincevicfer's Commits

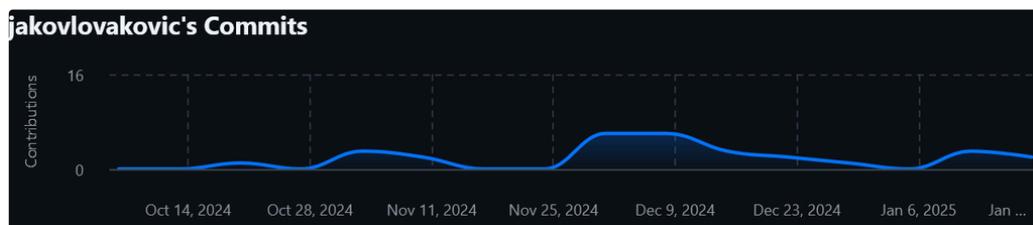
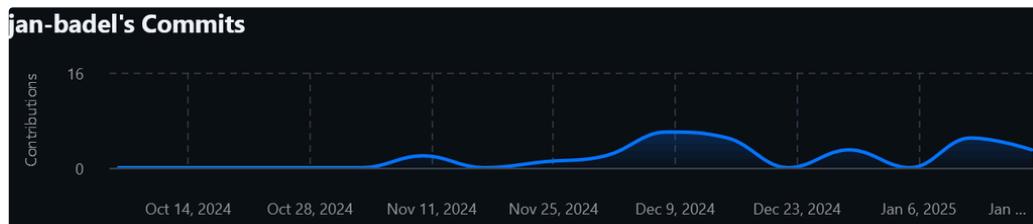


## filipbelina's Commits



## leomarusic's Commits





Posljednji mjesec rada na prvoj predaji  
Redom prikazani: Lovaković, Belina, Šainčević, Marušić, Lalić, Badel, Marinović



## Posljednjih mjesec dana na drugoj predaji

## Kjučni izazovi i rješenja

- Najveći izazovi bio je .NET povezati sa Oauth2, te deployanje .NET aplikacije
  - Opis izazova: Naše tehnologije činile su povezivanje Oauth2 iznimno izazovnime, te .NET nije bio lagan za deployati na Linux serveru
  - Rješenja: Projekt je dobio veliki redesign koji je predvodio Leo Marušić koji nam je olakšao implementaciju.

## Meetings

Lista svih sastanaka do sada:

- Meeting 29.10.2024.
- Meeting 28.10.2024.
- Meeting 25.10.2024.
- Meeting 23.10.2024.
- Meeting 22.10.2024.
- Meeting 18.10.2024.
- Meeting 13.10.2024.
- Meeting 10.10.2024.
- Meeting 28.11.2024
- BE - Meeting 30.11.2024.
- BE - Meeting 9.12.2024.
- Meeting 23.12.2024

## Meeting 29.10.2024.

**Datum:** 29.10.2024.

**Vrijeme:** 18:00 - 19:00

**Sudionici:** Mislav Marinković, Filip Belina, Jakov Lovaković, Leo Marušić, Jan Badel, Martin Šainčević, Jan Lalić

**Dnevni red:**

1. Generalna rasprava

# Meeting 28.10.2024.

**Datum:** 28.10.2024.

**Vrijeme:** 17:00 - 22:00

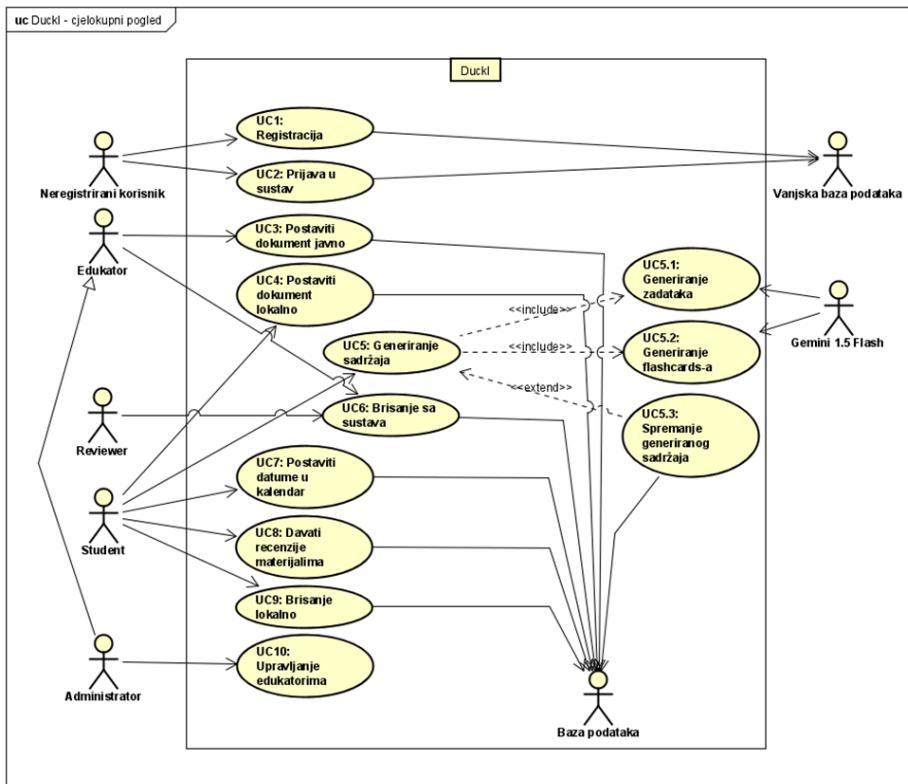
**Sudionici:** Mislav Marinković, Filip Belina, Jakov Lovaković, Leo Marušić

## Dnevni red:

1. Izrada Use-Case dijagrama
2. Izrada sekvencijskih dijagrama

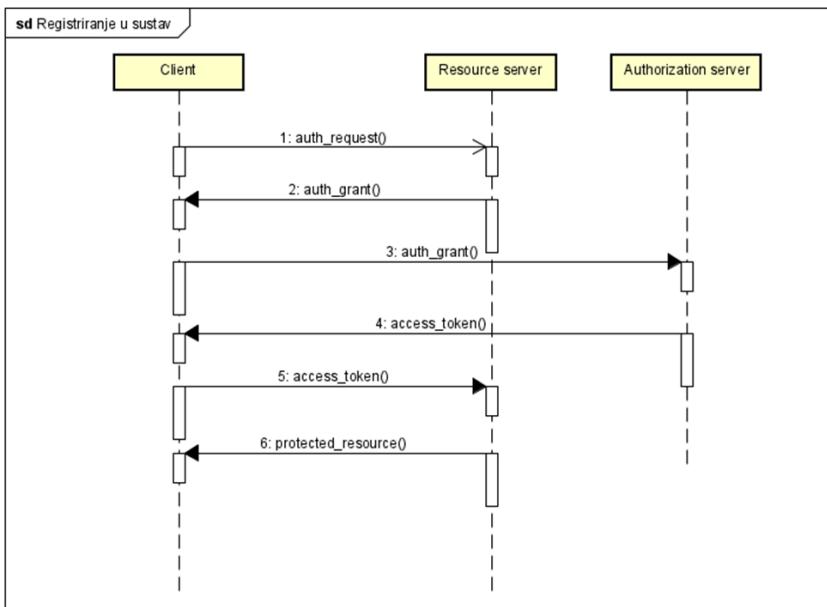
## Dijagram obrazaca uporabe

(potencijalno dodati obrasce uporabe u obliku tablica, poput one na primjeru na Moodlu)

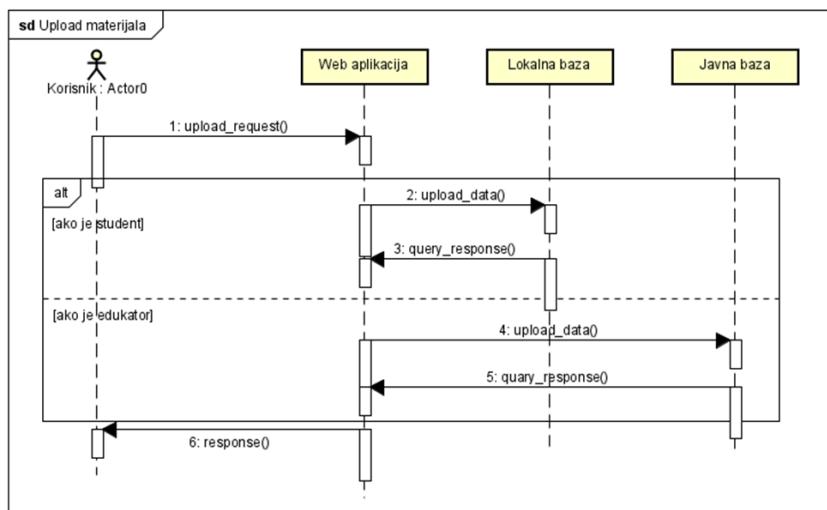


## Sekvencijski dijagrami

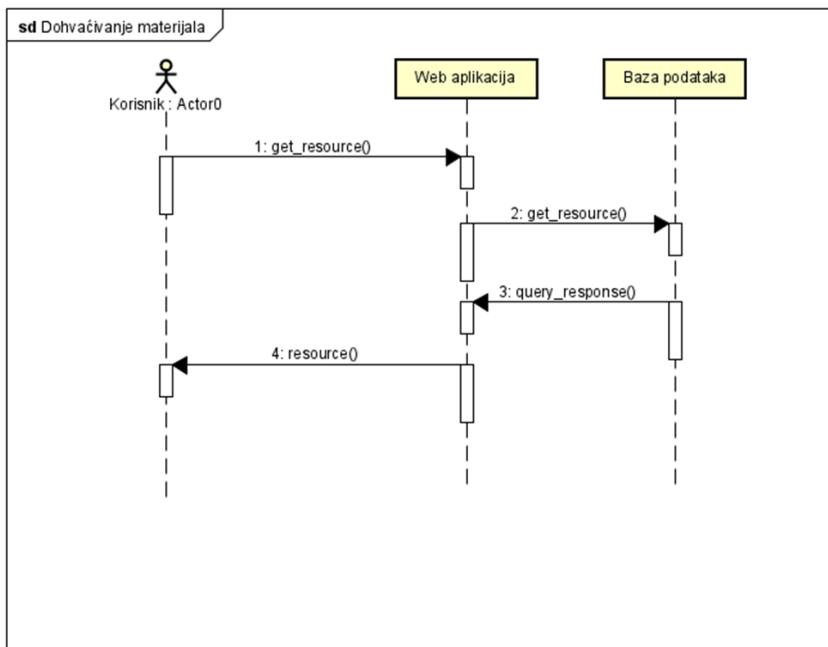
*Registriranje u sustav*



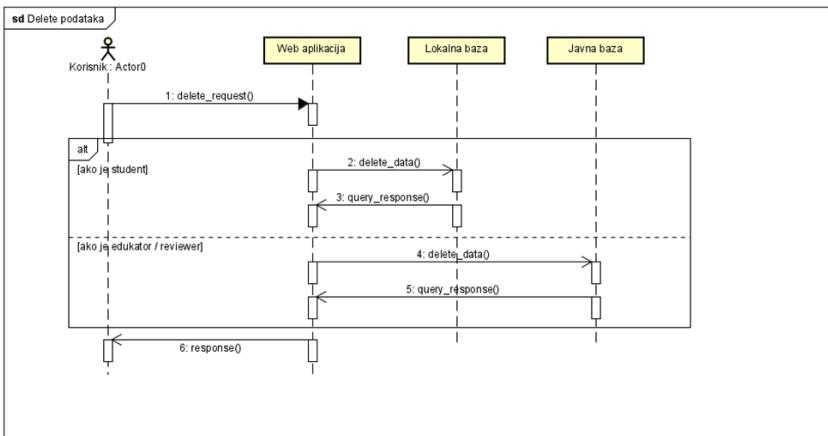
*Upload materijala*



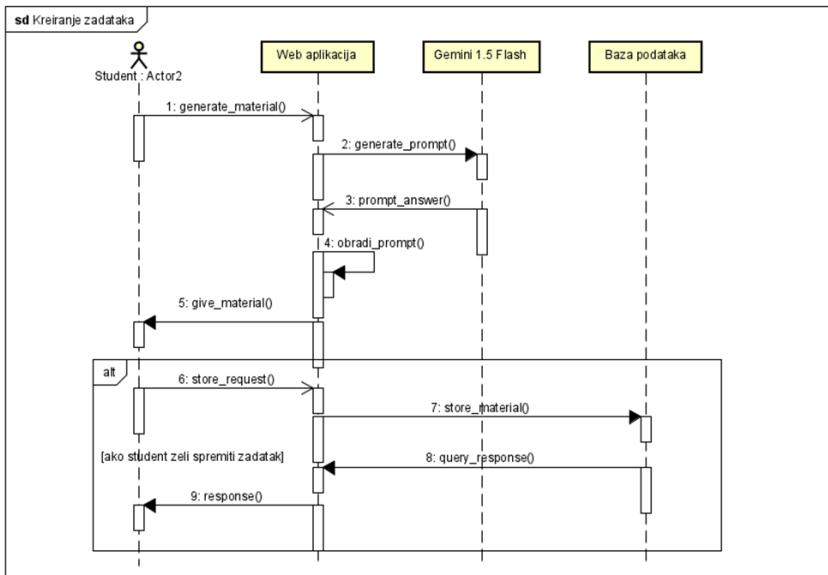
*Dohvaćivanje materijala*



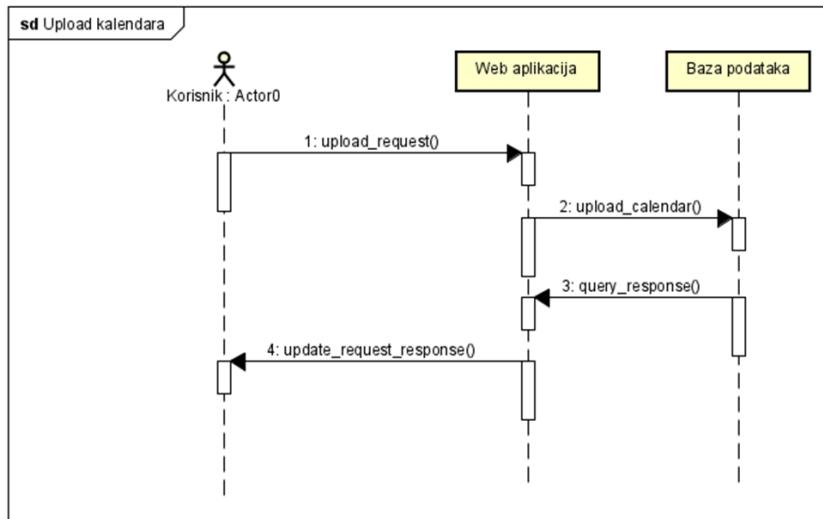
Brisanje materijala



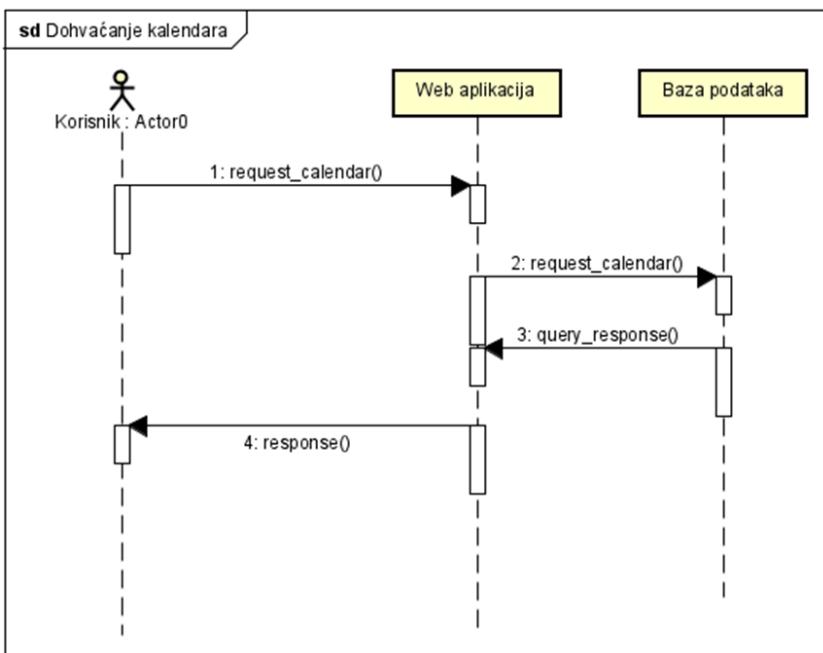
Kreiranje zadataka



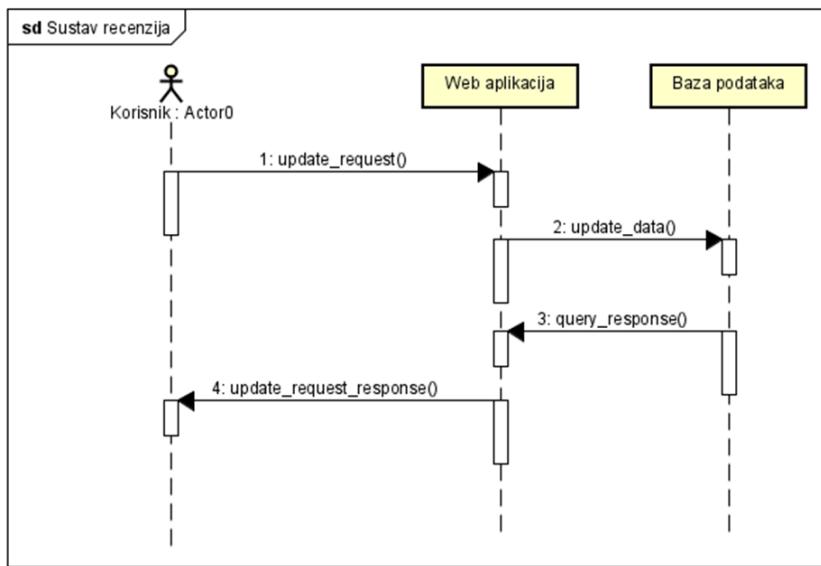
### Upload kalendarra



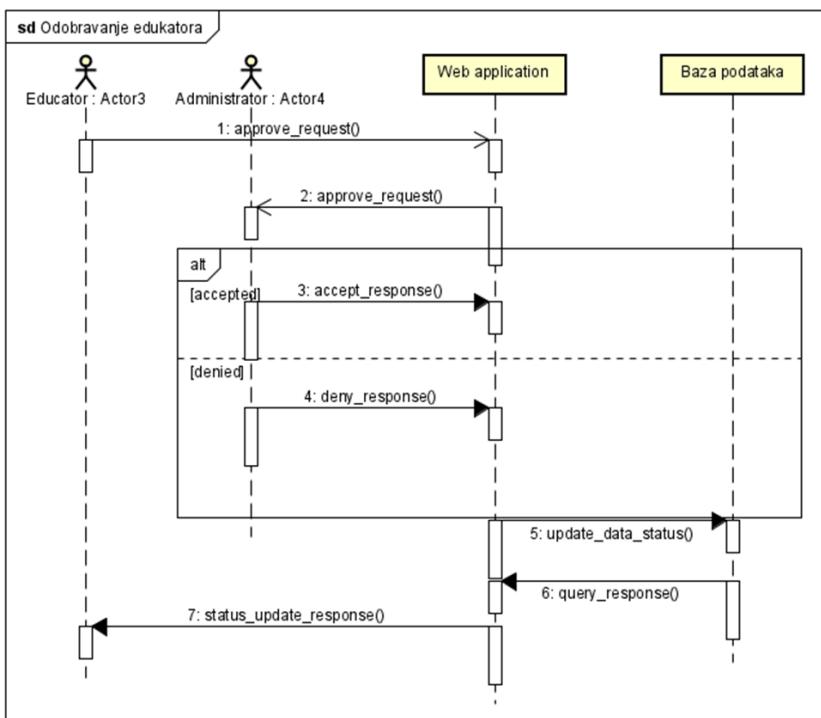
### Dohvaćanje kalendarra



### Sustav recenzija



Odobravanje edukatora



# Meeting 25.10.2024.

**Datum:** 25.10.2024.

**Vrijeme:** 19:30 - 20:30

**Sudionici:** Filip Belina, Jakov Lovaković, Jan Badel, Leo Marušić, Martin Šainčević, Jan Lalić

## **Dnevni red:**

1. Struktura web aplikacije
2. Implementacija baze podataka
3. Standardizacija procesa razvoja

### **1. Struktura web aplikacije**

Definirana je struktura web aplikacije, uključujući organizaciju datoteka i koda. Backend tim je izradio prvu fazu implementacije baze podataka za pohranu korisničkih podataka.

### **2. Implementacija baze podataka**

Voditelj backend tima je prezentirao strukturu datoteka i koda te objasnio kako raditi s njima.

### **3. Standardizacija procesa razvoja**

Kako bi se osigurao efikasan i ujednačen proces razvoja, definirani su standardi za:

- **Kreiranje zadataka (ticketa):** Utvrđen je način kreiranja zadataka u JIRA-i, uključujući potrebne informacije i format opisa.
- **Kreiranje grana (brancheva):** Definisana je konvencija imenovanja grana u Git repozitoriju.
- **Pull zahtjevi (pull requests):** Utvrđene su smjernice za kreiranje pull zahtjeva, uključujući opis promjena i proces revizije koda.
- **Confluence dokumentacija:** Definiran je način dokumentiranja u Confluence-u, uključujući strukturu stranica i potrebne informacije.

Sva će dokumentacija biti prvo kreirana u Confluence-u, a zatim će biti migrirana u GitHub wiki. Ova će dokumentacija poslužiti kao osnova za izradu službene dokumentacije projekta.

## **Zaključak:**

Sastanak je bio uspješan, a tim je napravio značajan korak u definiranju strukture web aplikacije i standardizaciji procesa razvoja.

# Meeting 23.10.2024.

**Datum:** 23.10.2024.

**Vrijeme:** 19:00 - 20:00

**Sudionici:** Jakov Lovaković, Leo Marušić, Filip Belina, Martin Šainčević

## Dnevni red:

1. Ažuriranje zahtjeva projekta
2. Redizajn sheme baze podataka
3. Demo aplikacija i testiranje

### 1. Ažuriranje zahtjeva projekta

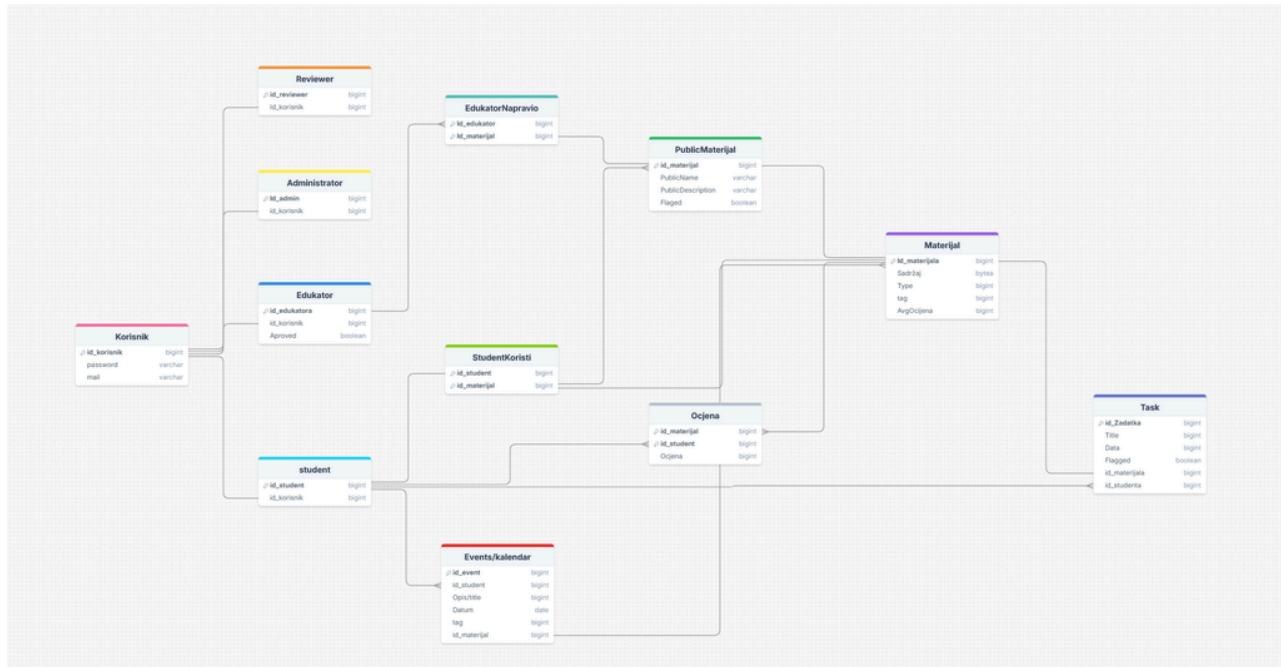
Nakon primanja dodatnih zahtjeva za projekt, neke od optionalnih funkcionalnosti sada su postale obavezne. To je utjecalo na dizajn baze podataka i zahtjevalo je redizajn.

### 2. Redizajn sheme baze podataka

Shema baze podataka je redizajnirana kako bi se prilagodila novim funkcionalnostima. Ipak, neki aspekti sheme još nisu finalizirani, uključujući:

- **Tipove podataka:** Potrebno je donijeti konačnu odluku o tipovima podataka za određene atribute.
- **Polja tablica:** Neka polja u tablicama zahtijevaju dodatnu diskusiju i definiranje.
- **Relacije:** Odredene relacije između entiteta trebaju biti detaljnije razrađene.

Ove će se točke razjasniti na sljedećem sastanku.



### 3. Demo aplikacija i testiranje

Demo aplikacija je modificirana kako bi koristila PostgreSQL umjesto Microsoft SQL baze podataka. Testiranje aplikacije je započelo na više računala prije samog deploymenta.

## Zaključak:

Unatoč promjenama u zahtjevima, tim je uspješno redizajnirao shemu baze podataka i napravio napredak u razvoju demo aplikacije.

# Meeting 22.10.2024.

## Zapisnik sa sastanka

Datum: 22.10.2024.

Vrijeme: 17:00 - 21:00

Sudionici: Filip Belina, Jakov Lovaković, Leo Marušić, Jan Lalić

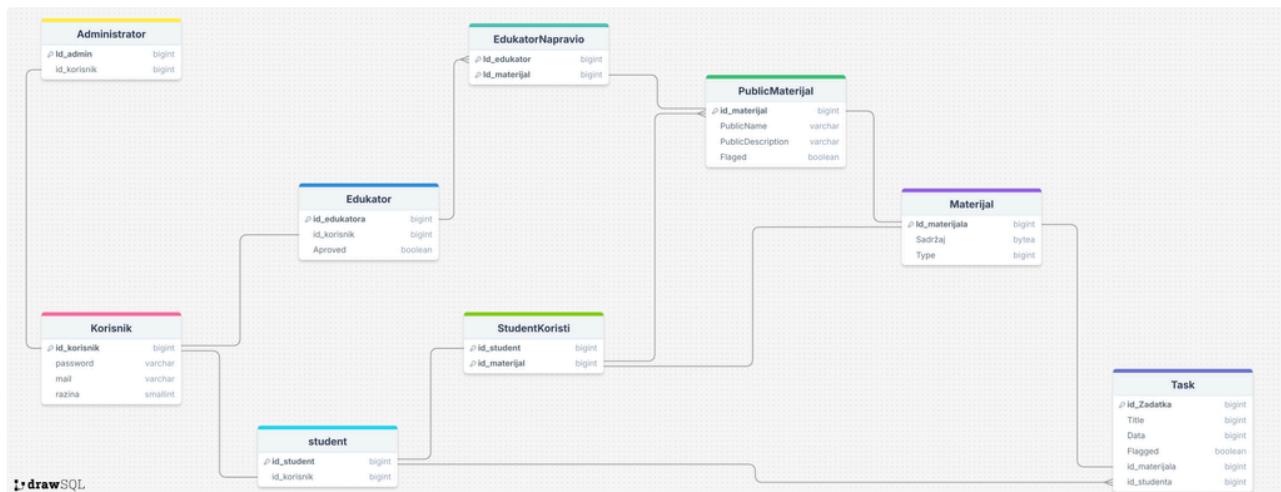
### Dnevni red:

1. Dizajn sheme baze podataka
2. Proces učitavanja podataka
3. Demo aplikacija

#### 1. Dizajn sheme baze podataka

Tim je detaljno razradio shemu baze podataka, definirajući:

- **Entitete:** Identificirani su ključni entiteti potrebni za pohranu podataka aplikacije (npr. korisnik, dokument, pitanje, flash kartica, kalendar).
- **Relacije:** Definisane su relacije između entiteta, uključujući tipove relacija (jedan-na-jedan, jedan-na-više, više-na-više).
- **Atribute:** Za svaki entitet su specificirani atributi i njihovi tipovi podataka. [TBD]



#### 2. Proces učitavanja podataka

Razmatran je proces učitavanja podataka u bazu podataka od strane korisnika. Definisani su:

- **Formati podataka:** Utvrđeni su formati datoteka koje će korisnici moći učitavati (npr. PDF za dokumente).
- **Metode učitavanja:** Razmotrene su različite metode učitavanja podataka (npr. direktno učitavanje, učitavanje putem API-ja).
- **Validacija podataka:** Definisane su procedure za validaciju učitanih podataka kako bi se osigurala konzistentnost i integritet baze podataka.

#### 3. Demo aplikacija

Tim je započeo s izradom demo aplikacije s ciljem:

- **Postavljanja baze podataka:** Kreiranje baze podataka na temelju definirane sheme.

- **Konfiguriranja okruženja za razvoj:** Postavljanje razvojnog okruženja i potrebnih alata.
- **Testiranja procesa učitavanja podataka:** Verifikacija funkcionalnosti učitavanja podataka u bazu.

**Zaključak:**

Sastanak je bio produktivan, a tim je ostvario značajan napredak u dizajnu baze podataka i definiranju procesa učitavanja podataka. Izrada demo aplikacije će omogućiti daljnje testiranje i validaciju arhitekture baze podataka.

# Meeting 18.10.2024.

**Datum:** 18.10.2024.

**Vrijeme:** 20:00 - 21:00

**Sudionici:** Svi

**Dnevni red:**

1. Uvod u JIRA i Confluence

## 1. Uvod u JIRA i Confluence

Održana je prezentacija o korištenju JIRA i Confluence alata za upravljanje projektima i timsku suradnju.

Sudionici su upoznati s osnovnim funkcionalnostima JIRA-e, uključujući:

- **Kreiranje zadataka (issues):** Objasnjen je proces kreiranja novih zadataka, definiranje prioriteta, dodjeljivanje tipova zadataka i postavljanje rokova.
- **Dodjeljivanje i upravljanje zadacima:** Prikazano je kako dodijeliti zadatke članovima tima, pratiti njihov napredak i ažurirati status.
- **Povezivanje zadataka:** Demonstrirano je kako povezati međusobno povezane zadatke radi bolje organizacije i praćenja.

Također, prezentirane su i ključne funkcionalnosti Confluence-a:

- **Kreiranje stranica:** Objasnjeno je kako kreirati i uređivati stranice, dodavati sadržaj (tekst, slike, tablice) te ih organizirati u hijerarhijske strukture.
- **Povezivanje JIRA zadataka i Confluence stranica:** Prikazano je kako povezati JIRA zadatke s relevantnim Confluence stranicama radi bolje dokumentacije i centraliziranog pristupa informacijama.
- **Timská suradnja:** Istaknute su mogućnosti Confluence-a za timsku suradnju, kao što su dijeljenje dokumenata, komentiranje i praćenje promjena.

**Zaključak:**

Sudionici su stekli osnovno znanje o korištenju JIRA i Confluence alata. Ova će znanja omogućiti efikasnije upravljanje projektom, bolju organizaciju zadataka i transparentniju komunikaciju unutar tima.

# Meeting 13.10.2024.

**Datum:** 13.10.2024.

**Vrijeme:** 19:00 - 20:00

**Sudionici:** svi

## **Dnevni red:**

1. Definiranje teme projekta
2. Osnovna ideja aplikacije
3. Odabir tehnologija

### **1. Definiranje teme projekta**

Na sastanku je detaljnije razrađena tema projekta i finaliziran njen opis.

### **2. Osnovna ideja aplikacije**

Definirana je osnovna ideja aplikacije koja će koristiti AI, putem Gemini API-ja, za obradu PDF dokumenata i kreiranje personaliziranih materijala za učenje (flash kartice, pitanja i sl.). Aplikacija će uključivati kalendar za praćenje ispita i planiranje učenja, te sustav dnevnih podsjetnika ili pitanja za kontinuirano učenje.

### **3. Odabir tehnologija**

Započeta je diskusija o odabiru tehnologija za razvoj aplikacije. Sljedeće tehnologije su predložene:

- **Frontend:** ReactJS
- **Backend:**  [ASP.NET Core | Open-source web framework for .NET](#)
- **Baza podataka:** PostgreSQL
- **Upravljanje projektom:** JIRA + Confluence

Konačna odluka o korištenju ovih tehnologija bit će donesena nakon dodatnog istraživanja i razmatranja alternativnih opcija.

## **Zaključak:**

Na sastanku je uspješno definirana tema projekta i osnovna ideja aplikacije. Započet je proces odabira tehnologija, a konačne odluke će biti donesene na sljedećem sastanku.

# Meeting 10.10.2024.

**Datum:** 10.10.2024.

**Vrijeme:** 18:00 - 19:00

**Sudionici:** Svi

## **Dnevni red:**

1. Upoznavanje i podjela uloga
2. Odabir teme projekta
3. Postavljanje Github repozitorija
4. Odabir imena tima

### **1. Upoznavanje i podjela uloga**

Članovi tima su se međusobno upoznali i predstavili. Definisane su sljedeće uloge unutar tima:

- Filip Belina - Project Lead
- Jakov Lovaković - Backend Team Lead
- Jan Lalić - Backend Engineer
- Leo Marušić - DevOps/Database Engineer
- Mislav Marinović - Lead design
- Jan Badel - Frontend Engineer
- Martin Šainčević - Frontend engineer

### **2. Odabir teme projekta**

Tim je donio odluku o razvoju projekta na vlastitu temu. Ukratko su predstavljene ideje za potencijalne teme. Konačna odluka o temi će biti donesena na sljedećem sastanku.

### **3. Postavljanje Github repozitorija**

Dogovoreno je postavljanje Github repozitorija za upravljanje kodom projekta. [Ime] će biti zadužen/a za kreiranje repozitorija i dodjeljivanje pristupa ostalim članovima tima.

### **4. Odabir imena tima**

Predložena su sljedeća imena za tim: [Navedite predložena imena]. Konačan odabir imena će se provesti putem glasanja do [datum].

## **Zaključak:**

Sastanak je uspješno održan, te su definirane osnovne smjernice za daljnji rad na projektu.

# Meeting 28.11.2024

Pogledati PR-ove:

- layout html bug request srediti ce se kroz iteracije frontenda, low priority, closeat cemo
- frontend uredivanje izgleda, jan badel radi, treba dizajn frontpagea se napraviti, treba izdvojiti css van koda (bootstrap moze zadrzati), martin nadodaje da ce se kroz par dana dovrsti css za kalendar
  - jakov kaze da miljenko ne gleda kod, i dalje filip smatra da se treba razdvojiti, jakov kaze da na neko standardno mjesto se stave css fileovi
  - request ostaje otvoren, frontend ce raditi na tome i raditi kroz jedan pull request
  - ostat cemo na bootstrapu i razor zbog jednostavnosti
  - index page nismo siguran je li postoji, jakov kaze da drzimo kontrolera.

```
1 leo addat ce google gemini prompting support, dodao je input (pitati Sv. Leopolda, napisao je komentar sa API keys, promptovi rade sad sa outputom)
2     - jakov i leo ce se sastati da imaju intimni sastanak u vezi ovoga : )
3 filip je napravio sprint 2, bilo bi dobro da je kalendar zasebna stvar da nije na front pageu
4     - ticket ce ostati sam po sebi, oba ticketa se mogu napraviti istim pull requestom, neka se frontend dogovori.
```

Segue jakovov: "sto mislite o ideji miljenkovoj da budemo ranije gotovi sa projektom." (do bozica / nove godine)

- filip nakazuje da ce biti nepredvidljivih problema, jakov kaze da AI nebi trebao davat greske, no concerned je za uloge korisnika i handling.
  - do 30.12 - 5.1 cemo PROBAT sve završiti, do 15.1. sigurno mora biti gotovo
  - druga stvar: ima dosta pull requesta, jakov pita kako cemo to rjesiti.
- filip: prvi koji prode ce biti bez problema, a drugi cemo mergeat conflicte (ako diraju isti dio koda cemo rewriteat i sl.), ugl. nije veliki problem.

Feature planning:

jakov: u iducih tj dana na backendu se treba rjesiti approveanje roleova, upload pdf materijala (jakov ce raditi

roleova, jan l. ce raditi materijale)

- filip kaze da se treba 2 dodatna tipa korisnika napraviti, te njihove prikaze, trebaju se uskladiti back i front teamovi
  - svaki korisnik bi imao neki gumb, samo se za jedan role stvatzmo prijaviti, kada se stisne se pozove ruta te ce u tablici uz id korisnika biti zabiljezen za koji se role prijavio, te ce onda admin mog vidjeti podatke iz tablice applyeva, uz ime se displaya gumb approve i disapprove, i stupac already reviewed
    - treba dat dokaze da si worthy uloge reviewer i edukator, ovisno kak je zamisljena aplikacija

```
1 webpage: - korisnik se logina kao student uvijek, featurei ce se pojavit ovisno o roleovima (inject C#, ako je role == student -> display kalendar... ista logika za ostatak roleova, cak ak netko zna rutu nece mog vidjet ili ista napraviti zlo)
2     - umjesto pocetnog ekerana ce biti prikazano, "koji role hoces?" -> ovisno sto kliknes odvede te na student home page ili formu za approveanje ostalih roleova i kojemu ce biti tekst (samo string) za zivotopis il slicno
3         (jakov kaze da ce leopold morati rewriteati kod, treba vidjeti s njim)
4         {- default role koji nije jos nista bi mogao biti dodan (role korisnik) - ovo dodajemo zbog prikaza elemenata i funkcionalnosti} - potencijalno puno posla pa moze student ostati default korisnik koji su svi kada udu u aplikaciju
5         - nema promjene rolea nakon sto se odabere role, dvije rute ce biti napravljene - dodavanje studenata i sve ostalo (ili jedna ruta, jakov kaze da nije bitno), podaci o studentu se vracaju u .json pa bi se ID korisnika mogao izvuc
6         - backend i frontend ce dobiti jedan ticket za cijeli koncept, svatko radi svoje podtickete; kada radimo reviewe moramo raditi detaljno i sa razumjevanjem da nebi bili unintended bugovi i nalazimo greske.
```

Sto se tice dokumentacije: napravljeno okayy ali od danas nadalje svaki pull request i update dodati u tablicu sto, tko, datum za olaksat namivot

- u daily report na dane kad ste radili da se doda sto se radilo zbog pracenja.

U subotu 30.11. backend call gdje ce se raspravljati tko sto i kako ce se raditi.

Do petka 29.11 navecer mislav ce probati dizajnove napraviti da imamo na osnovu cega raditi frontend.

Pogledati Leopoldov ticket, ne sramiti se stavljati komentare i tak.

Stavljeni komentari u kodu zbog profesora koji ce to gledati (trazeno je od nas, te se ocijenjuju).

Zaključak: danas dobivamo tickete, dodavati stvari u dnevnik zapisa i pisanje dnevne update, na backendu rjesiti handling roleova i AI.

Trajanje 2h.

## BE - Meeting 30.11.2024.

Jakov Lovaković i Jan Lalić

45 min

- diskutiranje o dodavanju novih tablica, interakciji admina s bazom, provjera postojanosti prijave na role, općenito diskutiranje implementacije roleova i podjela posla

## BE - Meeting 9.12.2024.

Sudionici: Jankov Lovaković, Filip Belina i Jan Lalić

Trajanje: 1 h

Diskutiranje implementacije i planu rada na projektu vezano uz dodavanje rolea SuperStudent, uploada, reviewa i searchanja/selekcija pdfova

## Meeting 23.12.2024

backend tim - jakov pokazao tagove za materijale, uploadnje i "flaganje" materijala da postanu privatni, reviewer delete public Materials page, removed pdf logovi neki zadatci:

- promijenit flaganje izraz - dodavanje pdf u private workspace
  - promijenit da se rating ne pokazuje, upvote downvote gumbove
    - promijenit alertove u nesto ne toxic
    - delete account gumb raspraviti
    - spremanje taskova je upitno
    - promijenit uploadpdf view
    - maknuti removedlogid stupac u removed pdf logs
    - promijeniti admin control panel ako nismo
    - jos kalendar i generiranje
    - fixati addeventlistener error - staviti script u text field i onda to sve u if
      - izvaditi js kod iz indexa

frontend - dovrstili roleove i applyanje roleova planovi: -test caseove

Trajanje 2h.