

Programsko inženjerstvo ak.god 2024./2025

Sveučilište u Zagrebu

Fakultet elektrotehnike i računarstva

Noodle

Tim: <TG09.2>

Ime tima: Noodle

Nastavnik: Vlado Sruk

Aplikacija Noodle je digitalno rješenje koje povezuje učenike i školsko osoblje s ciljem unaprjeđenja organizacije školskih aktivnosti i komunikacije unutar srednje škole.

Potencijalna korist ovog projekta

U današnje vrijeme, većina sustava pokušava digitalizirati i pojednostaviti korištenje korisnicima. Platforma Noodle na sličan način donosi brojne koristi za obrazovni sustav i njegove sudionike. Aplikacija je usmjerena na poboljšanje organizacije i komunikacije u srednjoj školi te na centraliziranje informacija i jednostavan pristup za sve korisnike. Cilj aplikacije je olakšati praćenje rasporeda, obaveza, aktivnosti i dostupnih resursa za učenike i nastavnike te podržati međusobnu komunikaciju i suradnju. Učenici, koji se prilikom upisa opredjeljuju za određeni smjer, dobivaju predmete prilagođene svom smjeru u skladu s kurikulumom, čime se osigurava usklađenost s obrazovnim standardima. Aplikacija automatski generira raspored sati, uključujući organizaciju praktične nastave u specijaliziranim prostorima te odlazak na prakse, uz automatsko obavlještavanje svih uključenih osoba o promjenama putem elektroničke pošte. Omogućene su i dodatne funkcionalnosti, poput slanja poruka između korisnika, objava nastavnih materijala isključivo učenicima kojima su namijenjeni te praćenje statistike pristupa materijalima. Učenici također imaju mogućnosti izvaditi potvrdu o upisu za potrebe izrade pokaza za javni prijevoz. Aplikacija omogućuje i korištenje kartografskih usluga u sklopu podrške za terensku nastavu i/ili prakse te prikaz vremenske prognoze za lokaciju na kojoj se škola nalazi. Kako bi platforma bila široko dostupna, nudi responzivni dizajn prilagođen različitim uređajima i podršku za autentifikaciju putem vanjskih servisa poput OAuth2, čime se osigurava zaštita podataka za korisnike. Platforma doprinosi učinkovitosti i organizaciji nastave u školama te omogućava jednostavniju komunikaciju korisnika i dijeljenje materijala na svima razumljiv i dostupan način.

Postojeća slična rješenja

Na internetu već postoje rješenja koja imaju slične funkcionalnosti kao Noodle te koja se koriste u obrazovnim sustavima. Jedan od takvih primjera je Google Classroom, koji omogućuje nastavnicima kreiranje digitalnih razreda, dijeljenje zadataka i materijala te praćenje napretka učenja. Koristeći pomoćne Google alate, poput Google Meeta ili Google Calendara, vrlo je lako ostvarena komunikacija između sudionika i efikasna organizacija vremena. Postoji i aplikacija Moodle, koja služi za dijeljenje materijala i e-učenje putem interaktivnih kvizova. Sličnosti vidimo i na primjeru FER-ovog Intraneta koji omogućuje pristup materijalima za sve predmete koje student sluša, kao i efikasno praćenje rasporeda predavanja i vježbi unutar akademске godine. Razlika u odnosu na navedene primjere je da platforma Noodle nudi integriranu podršku za cijelokupno upravljanje školskim resursima, od optimalnog slaganja rasporeda, gdje se u obzir uzimaju predmeti, dostupnost nastavnika i učionica te mogućnost korištenja opreme, do objave materijala i generiranja potvrda za učenike. Aplikacija veći fokus stavlja na općenitu organizaciju obrazovnog sustava i efikasniju komunikaciju između korisnika, nego na aspekte e-učenja, koje navedeni primjeri implementiraju. Također, aplikacija nudi mogućnost korištenja Google Mapsa kako bi učenicima olakšali odlazak na praksu. Osim toga, omogućena je autentifikacija putem OAuth2 servisa, što nudi veću fleksibilnost i sigurnost pri prijavi korisnika, što može biti poželjno kad su u pitanju učenici srednje škole. Poput i većine aplikacija danas, omogućen je responzivni dizajn kako bi korisnici mogli na bilo kojem uređaju pristupiti platformi, čime se osigurava češće i jednostavnije korištenje.

FER-Moodle Naslovnica Moja naslovnica Moji e-kolegiji

FER - Moodle

Site news

Pretplatite se na ovaj forum

Arhiva Moodle za prethodne godine
napisao/la Admin User - petak, 17. veljače 2012., 18:06

Arhiva za prethodnu akademsku godinu dostupna je na zahtjev.

Trajna poveznica
Raspravljajte o ovoj temi (0 odgovora za sada)

Kalendar
studenoga 2024.
Ned Pon Uto Sri Čet Pet Sub
1 2
3 4 5 6 7 8 9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30

Cijeli kalendar • Uvezite ili izvezite kalendare

Popis e-kolegija

Ostali kolegiji (12)
FER-ovi nastavni kolegiji (401)

Pretraži e-kolegije

?

1.1 Prikaz platforme Moodle

☰ Classroom

- Home
- Calendar
- Archived classes
- Settings

Don't see your classes?
Try another account

Add a class to get started

Create class Join class

②

1.2 Prikaz platforme Google Classroom

Skup korisnika koji bi mogao biti zainteresiran za ostvareno rješenje

Rješenje poput Noodlea bilo bi zanimljivo širokom spektru korisnika u obrazovnim institucijama. Krenuvši od samih učenika, kao krajnji korisnici najviše bi profitirali od lakšeg pristupa rasporedu, materijalima i obavijestima. Sama komunikacija bila bi olakšana i pristup osoblju bio bi jednostavniji. Nastavnici bi koristili aplikaciju za organizaciju nastave, upravljanje materijalima te komunikaciju s učenicima. Omogućeno je planiranje rasporeda i uvažavanje njihovih slobodnih termina. Obrazovne institucije, u ovom slučaju srednje škole, zainteresirane su za digitalizaciju i poboljšanje učinkovitosti svojih obrazovnih procesa. Administrativno osoblje i ravnatelj imali bi koristi od centraliziranog upravljanja resursima i školskom logistikom. Aplikacija nudi detaljno praćenje zauzetosti prostorija i specijalizirane opreme, automatsko generiranje potvrda o upisu te izvještaje o izdanim dokumentima, što doprinosi smanjenju administrativnog

opterećenja i boljoj organizaciji. Satničari i administrator bi imali bolji uvid i kontrolu nad složenim rasporedom. Satničaru se smanjuje opseg posla jer samo prati promjene u rasporedu, a uloga administratora dodjeljuje se ravnatelju škole kako bi mogao imati pristup i kontrolu nad svim potrebnim podacima. Aplikacija predstavlja univerzalno rješenje koje je lako primjenjivo za različite potrebe unutar same srednje škole, ali i unutar cijelog sustava srednjih škola koje žele digitalizirati svoj sustav.

Mogućnost prilagodbe rješenja

Moguće prilagodbe za aplikaciju:

- fleksibilno dodavanje korisničkih uloga (npr. mentori, studentska služba, roditelji)
- prilagodba funkcionalnosti prema specifičnim potrebama korisnika (sustav funkcioniра za različite škole)
- mogućnost proširenja baze podataka (npr. uvodenje upisivanja ocjena, praćenje izostanaka)
- skalabilnost za veći broj korisnika i podataka (sustav je skalabilan, podržava rast škola i implementaciju za povećan broj osoblja i predmeta)
- sustav je prilagođen za izvođenje na različitim uređajima (kako bi se omogućili korisnicima pristup gdje god se nalazili)
- mogućnost integracije s drugim obrazovnim alatima po potrebi (npr. Zoom, Notebook LM)
- mogućnost promjene dizajna/korisničkog sučelja da odgovara zahtjevima/izgledu škole

Opseg projektnog zadatka

Projekt uključuje izradu aplikacije koja omogućava slaganje rasporeda, praćenje nastavnih aktivnosti i školskih resursa, kao što su specijalizirane prostorije i oprema te olakšava školama organizaciju nastave u skladu s potrebama različitih smjerova. Prvi korak za nove korisnike je registracija. Neregistrirani korisnik može se putem vanjskog servisa za autentifikaciju (OAuth2) ulogirati u sustav te mu se odmah na početku dodjeljuje uloga. Moguće uloge su: učenik, nastavnik, ravnatelj (ulozi administratora) i satničar. Uvezši u obzir da se radi o školskom sustavu, nije moguće dozvoliti pristup bilo kojem korisniku, već postoji lista čekanja (popis imena i prezimena dozvoljenog osoblja) uz pomoć koje ravnatelj dodjeljuje ulogu korisniku. Ravnatelju se dodjeljuje uloga administratora prije pokretanja stranice kako bi mogao voditi evidenciju za školu. Osobi koja sudjeluje u slaganju rasporeda se također dodjeljuje uloga (satničar) i daje mu se ovlast za pristup podacima koji su potrebni za slaganje rasporeda kako bi mogao raditi izmjene i slati obavijesti. Ovisno o dodijeljenoj ulozi, slijede koraci nakon registracije za postavljanje profila na stranici. U slučaju uloge učenika, bira se smjer (tehničar za računalstvo ili tehničar za elektroniku) kako bi se mogli odrediti obavezni i izborni predmeti u skladu s kurikulumom. U slučaju uloge nastavnika, odabiru se predmeti koje nastavnik smije predavati i dostupnost nastavnika tijekom tjedna. Nakon toga slaže se raspored za učenike i nastavnike te se učenici raspodijele u razrede, ovisno o upisanom smjeru. Svakom razredu dodjeljuje se razrednik. U oba slučaja se nakon generiranja rasporeda na elektroničku poštu šalje obavijest korisnicima o rasporedu. Na isti način nastavnici i učenici primaju obavijesti ako dođe do promjena u rasporedu. Nastavnik ima mogućnosti objave materijala u repozitorij za predmete koje predaje, a njihova dostupnost ograničena je na učenike koji slušaju predmet. Također, prati se statistika pristupa i korištenja materijala kako bi nastavnici imali povratnu informaciju. Učenici još putem elektroničke pošte mogu primiti potvrdu o upisu ako to zatraže. Sva ostala komunikacija odvija se unutar grupa za slanje poruka na samoj stranici, koje su automatski generirane po razredima. Unutar grupe razrednici mogu slati obavijesti o predmetima ili materijalima. Nastavnici također mogu od satničara zatražiti da obavi izmjenu u rasporedu po potrebi. Osim navedenih osnovnih funkcionalnosti, postoje još dvije dodane funkcionalnosti stranice. Prvo je

obavještavanje učenika o praksama i/ili terenskoj nastavi korištenjem usluga Google Mapsa. Prakse su dio rasporeda te se tjedan dana prije na elektroničku poštu učenicima šalje obavijest o lokaciji i kako doći do nje od škole. Zadnja funkcionalnost platforme je prikaz vremenske prognoze za mjesto u kojem se škola nalazi na naslovnoj stranici. Sama stranica implementira responzivni dizajn kako bi se omogućio pristup putem različitih uređaja i time olakšalo korištenje korisnicima.

Moguće nadogradnje projektnog zadatka

Moguće nadogradnje za platformu Noodle:

- proširenje pristupa aplikacije roditeljima kako bi mogli dobivati obavijesti o djetetu i direktno komunicirati s nastavnicima
- mogućnost offline rada kako bi učenici mogli koristiti aplikaciju bez pristupa internetu
- uvođenje mogućnosti videopoziva u svrhu hibridne nastave ili online konzultacija
- integracija s dodatnim obrazovnima alatima poput Kahoota ili Quizleta
- implementacija prikaza ocjena i povratnih komentara nastavnika kako bi učenici imali uvid u svoj napredak
- uvođenje alata za pristupačnost za učenike s poteškoćama u razvoju, npr. čitači ekrana, prilagodljivi font

Funkcionalni zahtjevi

ID zahtjeva	Opis	Prioritet	Izvor	Kriterij prihvaćanja
F-001	Sustav omogućuje registraciju neregistriranih korisnika putem sustava za autentifikaciju OAuth2.0	Visok	Zahtjevdionika	Korisnik može napraviti račun putem OAuth2.0 servisa (Google račun) te primiti potvrdu o registraciji na e-mail
F-002	Sustav omogućuje prijavu korisnika u sustav	Visok	Zahtjevdionika	Korisnik unosi svoje podatke i pristupa svom profilu na stranici
F-003	Sustav omogućuje stvaranje razrednih grupa	Visok	Zahtjevdionika	Učenici se raspodjeljuju u razredne grupe nasumično ovisno o upisanom smjeru te se svakoj grupi nasumično dodjeljuje razredni nastavnik
F-004	Sustav generira raspored sati prema smjerovima i razredima	Visok	Dokument zahtjeva	Raspored sati se generira s obzirom na kapacitet prostorija, dostupnost opreme i nastavnika te u skladu s kurikulumom
F-005	Sustav šalje obavijesti o izradi rasporeda	Visok	Povratne informacije	Obavijesti se šalju svim korisnicima unutar 5 minuta nakon izrade rasporeda
F-006	Sustav omogućuje pravovremeno obavještavanje korisnika o aktivnostima i promjenama u sustavu	Visok	Zahtjevdionika	Sustav šalje obavijesti putem elektroničke pošte svim korisnicima koji bi imali korist od primanja obavijesti
F-007	Sustav omogućuje vađenje potvrde o upisu	Srednji	Zahtjev učenika	Učenici na zahtjev primaju potvrdu o upisu elektroničkom poštom
F-008	Sustav omogućuje upućivanje učenika na stručnu praksu	Srednji	Dokument zahtjeva	Učenici imaju pristup karti s podacima o lokaciji i načinu dolaska na stručnu praksu
F-009	Sustav omogućuje direktnu komunikaciju između korisnika	Visok	Postojeći sustav	Razrednik i učenici mogu slati i primati poruke u chat unutar razrednih grupa
F-010	Sustav omogućuje nastavnicima objavu materijala	Visok	Dokument zahtjeva	Nastavnici mogu objaviti materijale (prezentacije, PDF dokumenti, video itd.) na repozitorij sustava

ID zahtjeva	Opis	Prioritet	Izvor	Kriterij prihvatanja
F-011	Sustav omogućuje učenicima pristup materijalima	Visok	Zahtjev dionika	Učenici mogu pristupiti samo materijalima iz predmeta koji su dio njihovog rasporeda, mogu pregledavati materijale ili ih skinuti na vlastito računalo
F-012	Sustav prati statistiku pristupa materijalima	Srednji	Povratne informacije	Nastavnici mogu vidjeti broj pregleda i preuzimanja za pojedine materijale koje objavljuju
F-013	Sustav omogućuje praćenje izostanaka učenika	Srednji	Zahtjev dionika	Nastavnici mogu unositi izostanke učenika za pojedine dane u tjednu koje su propustili
F-014	Sustav omogućava promjene rasporeda	Visok	Zahtjev dionika	Nastavnici mogu zatražiti od satničara da provede promjenu u rasporedu, (obavijest o promjeni rasporeda šalje se na e-mail)

Ostali zahtjevi

Nefunkcionalni zahtjevi

Zahtjevi performansi

ID zahtjeva	Opis	Prioritet
NF-1.1	Sustav treba podržavati rad više korisnika u stvarnom vremenu	Visok
NF-1.2	Sustav treba biti dizajniran tako da može lako proširiti funkcionalnosti i podržati veći broj korisnika bez smanjena performansi	Srednji
NF-1.3	Sustav treba dopustiti korisnicima upravljanje aplikacijom s odgovorima na akcije unutar 3 sekunde	Visok

Zahtjevi pouzdanosti

ID zahtjeva	Opis	Prioritet
NF-2.1	Sustav treba biti dostupan 99 % vremena	Visok
NF-2.1.1	Sustav mora imati omogućen pristup iz javne mreže	Srednji

ID zahtjeva	Opis	Prioritet
NF-2.2	Sustav treba biti otporan na neispravno korištenje korisničkog sučelja i ne smije dovesti do pada sustava	Srednji

Sigurnosni zahtjevi

ID zahtjeva	Opis	Prioritet
NF-3.1	Sustav treba implementirati autentifikaciju putem OAuth2.0, omogućujući autentifikaciju putem Google računa	Visok
NF-3.2	Sustav mora ograničiti korisnika na pristup jedino onim resursima kojima ima pristup koristeći role-based access control(RBAC), odnosno u ovisnosti s ulogama korisnika	Visok
NF-3.3	Sustav mora koristiti HTTPS protokol za sigurnu komunikaciju između klijenta i poslužitelja	Visok

Zahtjevi za korisničko iskustvo

ID zahtjeva	Opis	Prioritet
NF-4.1	Korisničko sučelje treba biti jednostavno i intuitivno za korištenje, s jasnim navigacijama i vizualnim elementima	Visok
NF-4.1.1	Sustav treba omogućiti podršku za hrvatski jezik i jasno prikazivanje informacija za sve korisnike	Srednji
NF-4.1.2	Sustav treba implementirati sučelje koje je responzivno i optimizirano za različite uređaje (pametni telefoni, tableti, desktop uređaji)	Visok
NF-4.2	Sustav treba koristiti OpenRouteService prilikom upućivanja učenika na praksu	Visok
NF-4.3	Sustav treba koristiti OpenWeatherMap za prikaz prognoze na naslovnoj stranici koja se ažurira svakih 30 minuta	Srednji

Zahtjevi za održavanje

ID zahtjeva	Opis	Prioritet
NF-5.1	Sustav treba biti dizajniran tako da se može lako održavati i nadograditi, s jasno definiranim modulima i mogućnošću jednostavne implementacije novih funkcionalnosti	Visok
NF-5.1.1	Sustav treba imati dovoljnu dokumentaciju	Visok

ID zahtjeva	Opis	Prioritet
NF-5.1.1	Sustav treba biti opisan putem dokumenta oblikovanja /SRS/	Visok
NF-5.1.2	Sustav treba implementirati koristeći objektno-orientirane jezike	Visok

Zahtjevi domene

ID zahtjeva	Opis	Prioritet
D-001	Sustav mora osigurati da je raspored za svaki smjer u skladu s informacijama objavljenim na službenim stranicama Ministarstva znanosti, obrazovanja i sporta	Visok
D-002	Sustav mora omogućiti prikaz rasporeda po tjednima u tabličnom obliku za akademsku godinu, uvezši u obzir blagdane, neradne dane i vikende	Visok
D-003	Sustav treba biti uskladen s pravilima zaštite podataka, uključujući GDPR ili slične regulative	Visok
D-004	Sustav treba omogućiti pohranu podataka o učenicima i nastavnicima u skladu s obrazovnim pravilnicima i zakonima o zaštiti podataka	Visok
D-005	Sustav mora omogućiti upravljanje materijalima tako da svi materijali za nastavu budu organizirani prema predmetima i godinama	Srednji
D-006	Sustav mora omogućiti ravnatelju škole da prati podatke o učenicima i nastavnicima	Visok

Dionici

Dionici:

1. Razvojni tim
2. Administrator sustava
3. Korisnici

Aktori:

A-1 Neregistrirani korisnik može:

1. Napraviti registraciju (F-001)

A-2 Registrirani korisnik (ravnatelj, satničar, nastavnik ili učenik) može:

1. Prijaviti se u sustav (F-002)
2. Pregledavati raspored sati (F-004)
3. Primati obavijesti vezano uz školske aktivnosti, raspored i sl. (F-006)

A-3 Učenik može:

1. Pronaći rutu do prakse (F-008)
2. Komunicirati u razrednoj grupi (F-003, F-009)
3. Pristupati materijalima (F-011)
4. Zatražiti potvrdu o upisu (F-007)

A-4 Nastavnik može:

1. Komunicirati unutar grupe (F-003, F-009, samo ako je nastavnik ujedno i razrednik)
2. Objavljivati materijale (F-010)
3. Pratiti statistiku korištenja materijala (F-012)
4. Zapisivati i pregledavati izostanke učenika (F-013)
5. Tražiti promjene u rasporedu (F-014)

A-5 Satničar može:

1. Mijenjati raspored sati (F-014)
2. Komunicirati s ravnateljom ili nastavnicima o promjenama u rasporedu

A-6 Ravnatelj (u ulozi administratora) može:

1. Odobriti ili odbiti zahtjev za registraciju (F-001)
2. Dodijeliti ulogu registriranom korisniku (F-001)
3. Nadgledati sigurnost sustava
4. Pregledavati statistiku korištenja materijala i izostanaka učenika (F-012, F-013)

Obrasci uporabe

1. Visokorazinski dijagram obrazaca uporabe cijelog sustava

UC1 - Registracija novih korisnika

UC2 - Prijava korisnika u sustav

UC6 - Generiranje rasporeda sati

UC7 - Objava materijala

UC8 - Pristupanje materijalima

UC10 - Upute za odlazak na praksu

2. Dijagram obrazaca uporabe za ključne funkcionalnosti

UC5 - Generiranje razrednih grupa

UC9 - Izdavanje potvrda o upisu

UC11 - Promjene u rasporedu

UC12 - Unos izostanaka

3. Dijagram obrazaca uporabe za korisničke role

UC3 - Biranje smjera

UC4 - Biranje predmeta

UC7 - Objava materijala

UC8 - Pristupanje materijalima

UC11 - Promjene u rasporedu

UC12 - Unos izostanaka

4. Dijagram obrazaca uporabe za osnovne poslovne procese

UC5 - Generiranje razrednih grupa

UC6 - Generiranje rasporeda sati

UC9 - Izdavanje potvrda o upisu

5. Dijagram obrazaca uporabe za kritične sustave i integracije

UC10 - Upute za odlazak na praksu

Opis obrazaca uporabe

UC1 - Registracija novih korisnika

- Glavni sudionik: Neregistrirani korisnik
- Cilj: Omogućiti novom korisniku da se registrira u sustav
- Sudionici: Baza podataka, administrator, OAuth2.0
- Preduvjet: Korisnik mora biti učenik ili djelatnik škole
- Opis osnovnog tijeka:

1. Korisnik pristupa stranici za registraciju i odabire jedna od računa (Google ili Microsoft) za registraciju u sustav. (F-001)
2. Korisnik šalje zahtjev za registraciju.
3. Administrator prima zahtjev i pregledava podatke.
4. Administrator potvrđuje registraciju.
5. Korisnik dobiva potvrdu putem e-maila da je uspješno registriran.

- Opis mogućih odstupanja:
 - 3.1 Korisnik ne zadovoljava uvjete za registraciju, ne nalazi se na listi ovlaštenih.

Rješenje: Administrator odbija zahtjev za registraciju.

3.2 Korisnik je već registriran u sustavu.

Rješenje: Prikazati korisniku poruku da je već obavio registraciju i sugerirati prijavu u sustav.

5. Potvrda nije poslana.

Rješenje: Korisnik može zatražiti ponovno slanje potvrde.

UC2 - Prijava korisnika u sustav

- Glavni sudionik: Korisnik
- Cilj: Uspješno prijaviti korisnika u sustav
- Sudionici: Baza podataka
- Preduvjet: Korisnik je uspješno obavio registraciju
- Opis osnovnog tijeka:

1. Korisnik pristupa stranici za prijavu u sustav.
2. Korisnik se prijavljuje putem Google ili Microsoft računa.
3. Sustav provodi validaciju unesenih podataka s podacima u bazi podataka. (F-002)
4. Korisnik se prijavljuje i pristupa naslovnoj stranici.

- Opis mogućih odstupanja:

3. Autentifikacija putem OAuth2.0 sustava nije uspjela

Rješenje: Prikazuje se obavijest o pogrešnim podacima i korisniku se omogućava ponovna prijava.

UC3 - Biranje smjera

- Glavni sudionik: Administrator
- Cilj: Omogućiti adminu odabir smjera za učenika

- Sudionici: Baza podataka
- Preduvjet: Korisniku je dodijeljena uloga učenika u sustavu
- Opis osnovnog tijeka:
 1. Učenik pristupa stranici za registraciju i unosi podatke.
 2. Zahtjev se šalje na odobravanje.
 3. Administrator prima zahtjeva i odabire smjer za učenika (matematički ili informatički). *ovdje uzimamo u obzir da administrator (ravnatelj) posjeduje listu svih upisanih učenika i njihovih smjerova
 4. Administrator odobrava zahtjev.
 5. Sustav pohranjuje odabrani smjer u bazu podatka.
- Opis mogućih odstupanja:
 3. Učeniku nije zapisan smjer koji je upisao

Rješenje: Administrator šalje mail osobno učeniku i provjerava podatke.

UC4 - Biranje predmeta

- Glavni sudionik: Administrator
- Cilj: Omogućiti administratoru odabir predmeta za nastavnike
- Sudionici: Baza podataka
- Preduvjet: Korisniku je dodijeljena uloga nastavnika u sustavu
- Opis osnovnog tijeka:
 1. Nastavnik obavlja registraciju i unosi podatke.
 2. Zahtjev se šalje na odobravanje.
 3. Administrator prima zahtjev i odabire predmete za nastavnika. *ovdje uzimamo u obzir da administrator (ravnatelj) posjeduje listu svih nastavnika koji predaju i njihovih smjerova
 4. Administrator odobrava zahtjev.
 5. Sustav pohranjuje odabrani smjer u bazu podatka.
- Opis mogućih odstupanja:
 3. Nisu zapisani predmeti koje nastavnik predaje.

Rješenje: Administrator šalje mail osobno nastavniku i provjerava podatke.

UC5 - Generiranje razrednih grupa

- Glavni sudionik: Sustav
- Cilj: Automatski generirati razredne grupe za učenike prema odabranom smjeru i svakoj dodijeliti razrednika
- Sudionici: Baza podataka
- Preduvjet: Sustav ima podatke o svim učenicima i njihovim odabranim smjerovima.
- Opis osnovnog tijeka:
 1. Sustav pristupa podacima o učenicima koji su smjer odabrali (tehničar za računalstvo ili tehničar za elektroniku).
 2. Sustav nasumično dijeli učenike s obzirom na odabrani smjer, tako da u jednom razredu bude maksimalno 25 učenika.
 3. Sustav zatim svakoj grupi nasumično dodjeljuje jednog nastavnika koji postaje razrednik grupe.
 4. Grupa i razrednik se pohranjuju u bazu podataka. (F-003)

5. Učenici i nastavnici na e-mail dobivaju obavijest o dodjeli grupe i razrednika. (F-006)
6. Nakon formiranja grupe, sustav automatski stvara privatni chat za svaku grupu, u koji su dodani svi članovi grupe i unutar kojeg je omogućena komunikacija između članova. (F-009)
7. Gotove grupe se prikazuju na korisničkim stranicama za učenike i nastavnike, uključujući chat prostor.

- Opis mogućih odstupanja:
 7. Korisnicima se dodijeljena grupa ne prikazuje nigdje na stranici.

Rješenje: Administrator ručno dodaje korisnika u slobodnu grupu.

UC6 - Generiranje rasporeda sati

- Glavni sudionik: Sustav
- Cilj: Automatski generirati raspored sati po smjerovima
- Sudionici: Baza podataka
- Preduvjet: Svi relevantni podaci(dostupnost nastavnika i učionica, smjerovi svih učenika) su pohranjeni i dostupni u sustavu, stvorene su razredne grupe
- Opis osnovnog tijeka:
 1. Sustav koristi algoritam za generiranje rasporeda na temelju podataka u bazi.
 2. Algoritam u obzir uzima: dostupnost nastavnika, dostupnost i broj učionica, razredne grupe, odabrane smjerove i kurikulum.
 3. Sustav automatski generira raspored za učenike i nastavnike, po razrednim grupama. (F-004)
 4. Raspored se pohranjuje u bazu podataka.
 5. Sustav generira obavijest i šalje učenicima, nastavnicima i satničaru na e-mail adresu. (F-005)
 6. Gotov raspored se prikazuje na stranici.
- Opis mogućih odstupanja:
 3. Algoritam ne uspijeva generirati raspored.

Rješenje: Sustav prikazuje obavijest administratoru o grešci u generiranju rasporeda.

UC7 - Objava materijala

- Glavni sudionik: Nastavnik
- Cilj: Omogućiti nastavniku da objavi materijale za predmete na stranicu
- Sudionici: Baza podataka
- Preduvjet: Korisnik ima ulogu nastavnika
- Opis osnovnog tijeka:
 1. Nastavnik pristupa stranici repozitorija u sustavu.
 2. Nastavnik odabire predmet za koji želi objaviti datoteku.
 3. Nastavnik odabire vrstu materijala koju želi dodati (PDF, video, slika, itd.)
 4. Nastavnik učitava materijal na repozitorij. (F-010)
 5. Nastavnik može dodati opis materijala po potrebi.
 6. Sustav pohranjuje samo metapodatke materijala u bazu podataka.
 7. Nakon objave, materijal postaje vidljiv samo učenicima upisanima na predmet.
- Opis mogućih odstupanja:
 4. Nastavnik nije uspio učitati materijal (npr. loša internetska veza, neispravan format datoteke).

Rješenje: Sustav prikazuje poruku o grešci i omogućuje ponovno učitavanje materijala.

UC8 - Pristupanje materijalima

- Glavni sudionik: Učenik
- Cilj: Omogućiti učeniku pristup potrebnim materijalima
- Sudionici: Baza podataka
- Preduvjet: Učenik je prijavljen u sustav i upisan je na predmet čijim materijalima želi pristupiti
- Opis osnovnog tijeka:

1. Učenik pristupa stranici repozitorija u sustavu.
2. Učenik odabire predmet za koji želi pregledati materijale.
3. Sustav prikazuje popis materijala za odabrani predmet.
4. Učenik odabire materijal koji želi pregledati ili preuzeti. (F-011)
5. Učenik pregledava materijal.

- Opis mogućih odstupanja:
 2. Učeniku je zabranjen pristup za pregled materijala na odabranom predmetu.
- Rješenje:** Učenik može zatražiti pristup materijalima tako da pošalje e-mail administratoru, nakon čega mu administrator ručno dodjeljuje prava ako zaključi da je riječ o pogrešci.

UC9 - Izdavanje potvrda o upisu

- Glavni sudionik: Učenik
- Cilj: Omogućiti učeniku preuzimanje potvrde o upisu
- Sudionici: Baza podataka
- Preduvjet: Učenik je prijavljen u sustavu i upisan u bazu podataka
- Opis osnovnog tijeka:

1. Učenik pristupa stranici za potvrde unutar sustava.
2. Učenik odabire opciju "Izdavanje potvrde o upisu".
3. Sustav generira potvrdu o upisu koristeći podatke o učeniku iz baze.
4. Potvrda o upisu se šalje ne e-mail učenika u roku 15 minuta. (F-007)

- Opis mogućih odstupanja:
 4. Potvrda nije stigla na email
- Rješenje:** Učenik može ponovno zatražiti generiranje potvrde, ali tek nakon 60 minuta od prvog pokušaja.

UC10 - Upute za odlazak na praksu

- Glavni sudionik: Učenik
- Cilj: Omogućiti učeniku prikaz uputa za dolazak na lokaciju prakse od škole
- Sudionici: Baza podataka, OpenRouteService API
- Preduvjet: Učenik je prijavljen u sustav
- Opis osnovnog tijeka:

1. Učenik pristupa stranici karte na aplikaciji.
2. Na stranci se prikazuje karta koristeći OpenRouteService API-ja.
3. Učenik upisuje lokaciju od koje želi krenuti i lokaciju na koju treba doći.
4. Na karti se ispisuje prikaz rute za željene lokacije. (F-008)

- Opis mogućih odstupanja:

2./4. Karta ili upute nisu ispravno prikazane na stranici.

Rješenje: Učenik može pokušati ponovno učitati stranicu ili obavijestiti administratora o problemu.

UC11 - Promjene u rasporedu

- Glavni sudionik: Nastavnik

- Cilj: Omogućiti nastavniku da zatraži zamjenu u rasporedu putem obrasca u aplikaciji

- Sudionici: Baza podataka, satničar

- Preduvjet: Nastavnik je prijavljen u sustav

- Opis osnovnog tijeka:

1. Nastavnik pristupa stranici za promjene u rasporedu.
2. Nastavnik ispunjava obrazac za promjenu koju želi (unosi predmet, razlog zamjene i datum)
3. Nakon što nastavnik ispunji obrazac, sustav ga automatski proslijeđuje satničaru. (F-014)
4. Satničar prima zahtjev.
5. Satničar prihvata zahtjev i ručno obavlja zadani promjene unutar rasporeda.
6. Promjena se ažurira u bazi podataka.
7. Nastavnik dobiva obavijest o obradi zahtjeva unutar stranice za zamjene.
8. Korisnici dobivaju email o provedenoj promjeni.

- Opis mogućih odstupanja:

2. Nastavnik nije ispunio sve potrebne podatke u obrascu.

Rješenje: Sustav ne dopušta slanje obrasca dok svi podaci nisu ispunjeni.

5. Satničar zbog konflikata ne uspijeva prihvati zahtjev nastavnika.

Rješenje: Satničar odbija zahtjev i nastavniku se šalje obavijest o odbijenom zahtjevu za zamjenu.

Postupak se ovdje zaustavlja dok se ne pošalje novi zahtjev.

UC12 - Unos izostanaka

- Glavni sudionik: Nastavnik

- Cilj: Omogućiti nastavniku unos izostanaka za učenike.

- Sudionici: Baza podataka, Google Maps API

- Preduvjet: Nastavnik je prijavljen u sustav i ima ulogu razrednika

- Opis osnovnog tijeka:

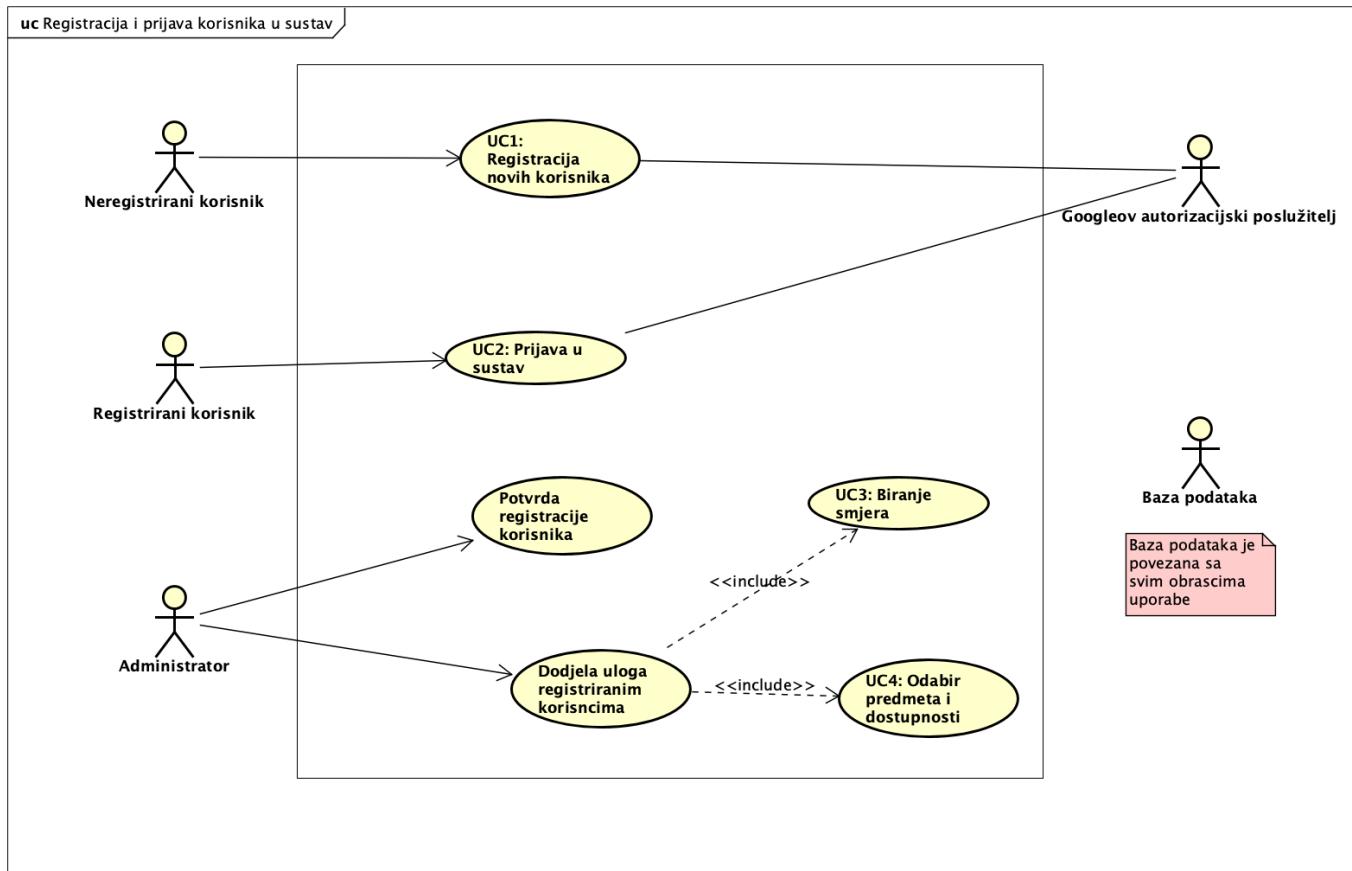
1. Nastavnik pristupa stranici za unos izostanaka unutar sustava.
2. Sustav prikazuje popis učenika iz razredne grupe nastavnika.
3. Nastavnik označava učenika/e koji je/su izostao/li.
4. Nastavnik unosi opis i datum izostanka za učenika/e.
5. Nastavnik potvrđuje unos izostanka.
6. Sustav pohranjuje podatke o izostanku u bazu podataka.

- Opis mogućih odstupanja:

4. Nastavnik ne uneše sve potrebne podatke (opis, status).

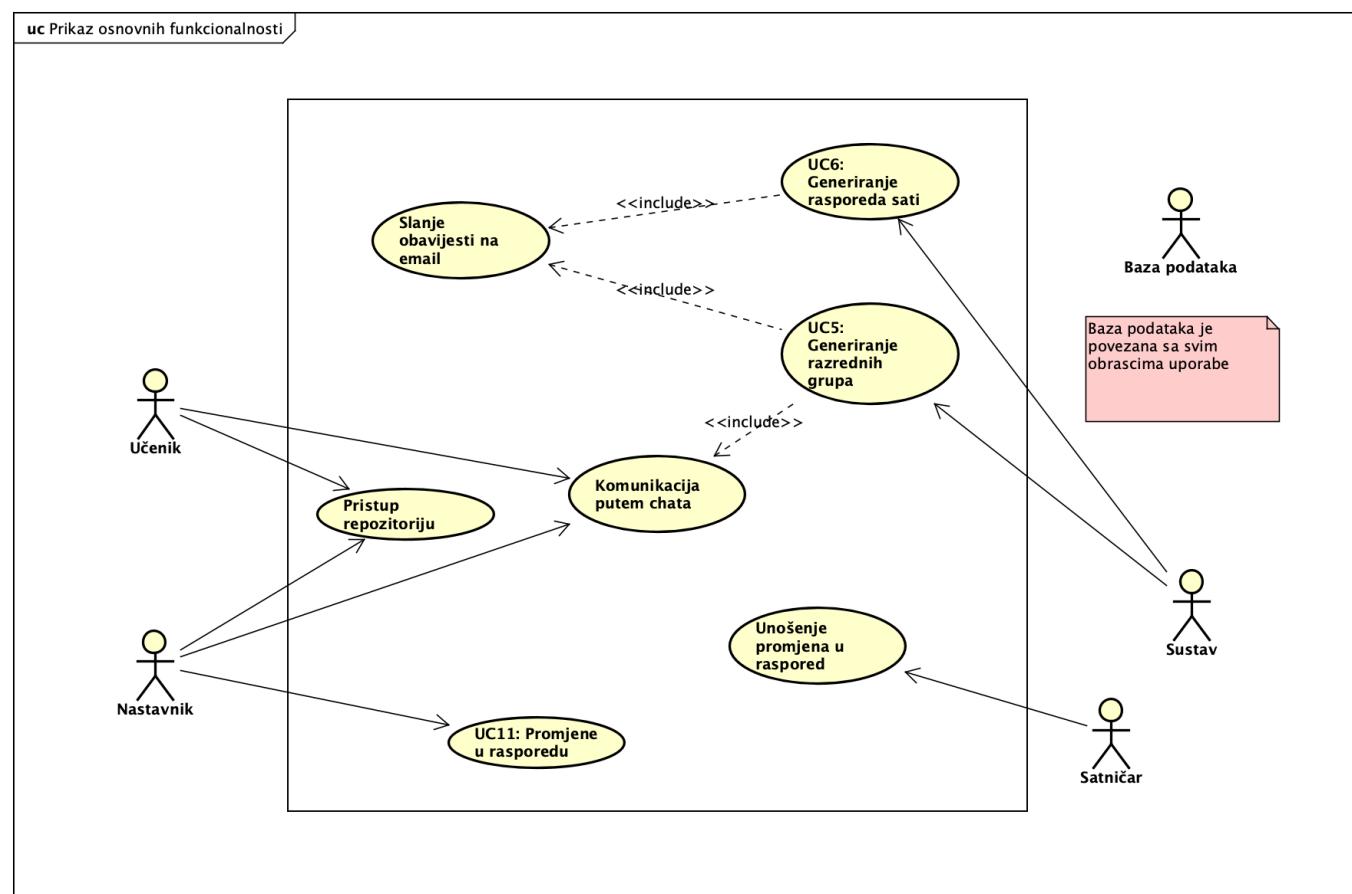
Rješenje: Sustav ne dopušta unos izostanaka dok svi podaci nisu ispunjeni.

Dijagrami obrazaca uporabe



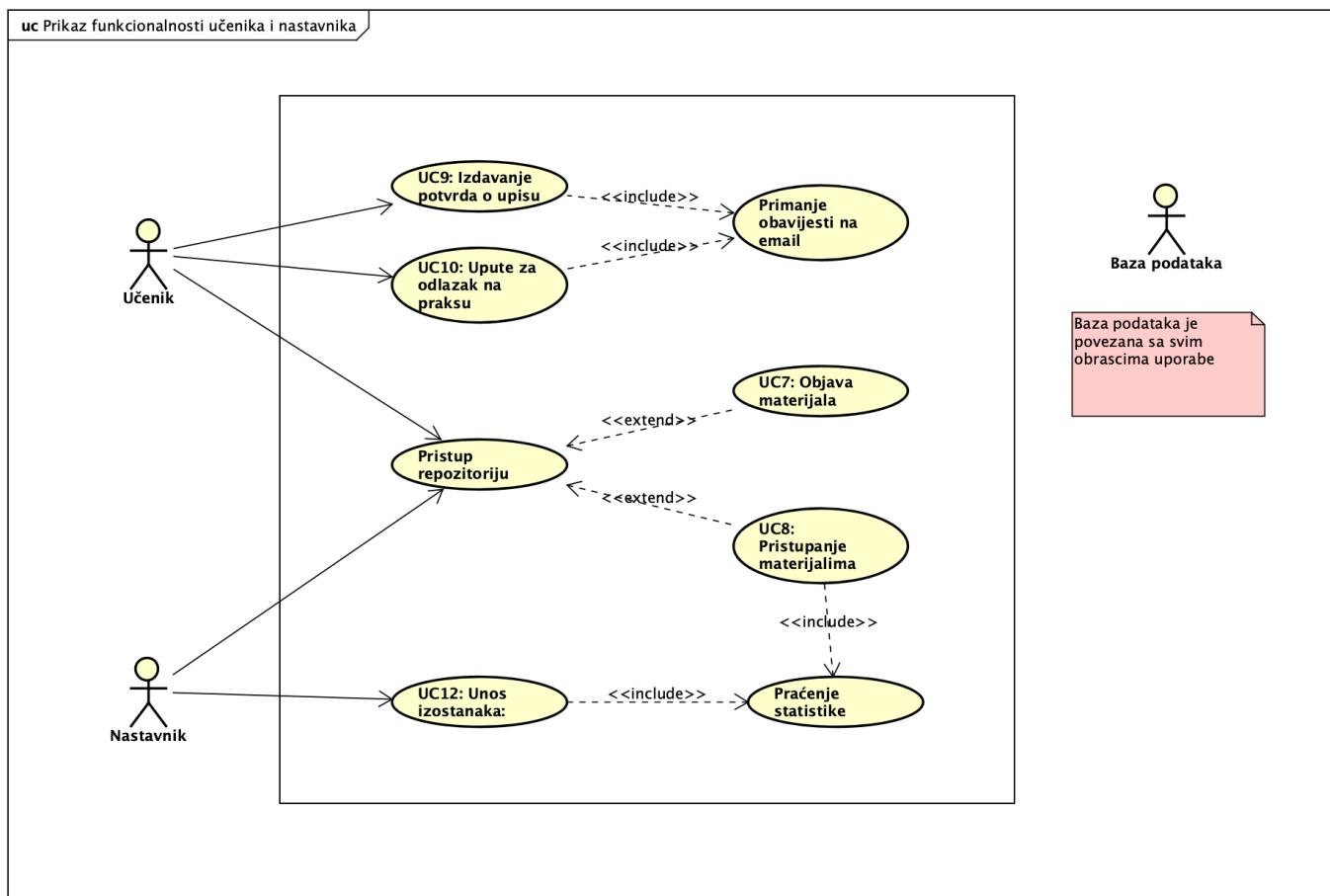
1.1 Prikaz registracije i prijave korisnika u sustav

Neregistrirani korisnik obavlja registraciju koristeći Googleov autorizacijski poslužitelj, a registrirani korisnik ga koristi za prijavu. Nakon prijave u sustav, korisnik ovisno o ulozi bira smjer ako je učenik ili predmete ako je nastavnik. Administrator (ravnatelj) je odgovoran za potvrdu registracije korisnika i dodjeljivanje uloga.



1.2 Prikaz osnovnih funkcionalnosti sustava

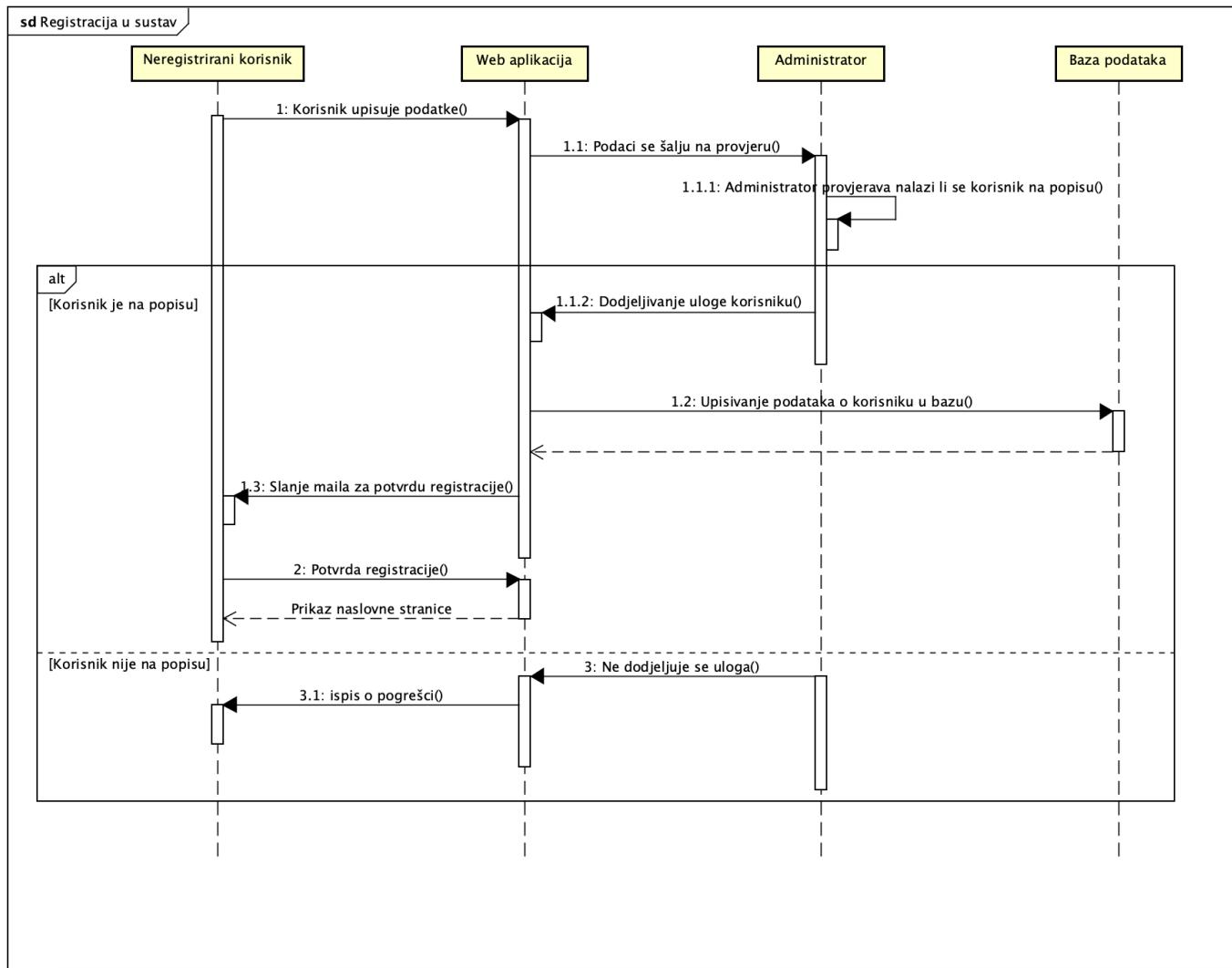
*obrasci uporabe UC5 i UC6 odnose se na izvođenje algoritama za generiranje grupa i rasporeda koje razvija razvojni tim, ali smo stavili sustav kao aktora budući da aplikacija, tj. sustav izvodi navedene obrasce



1.3 Prikaz funkcionalnosti učenika i nastavnika

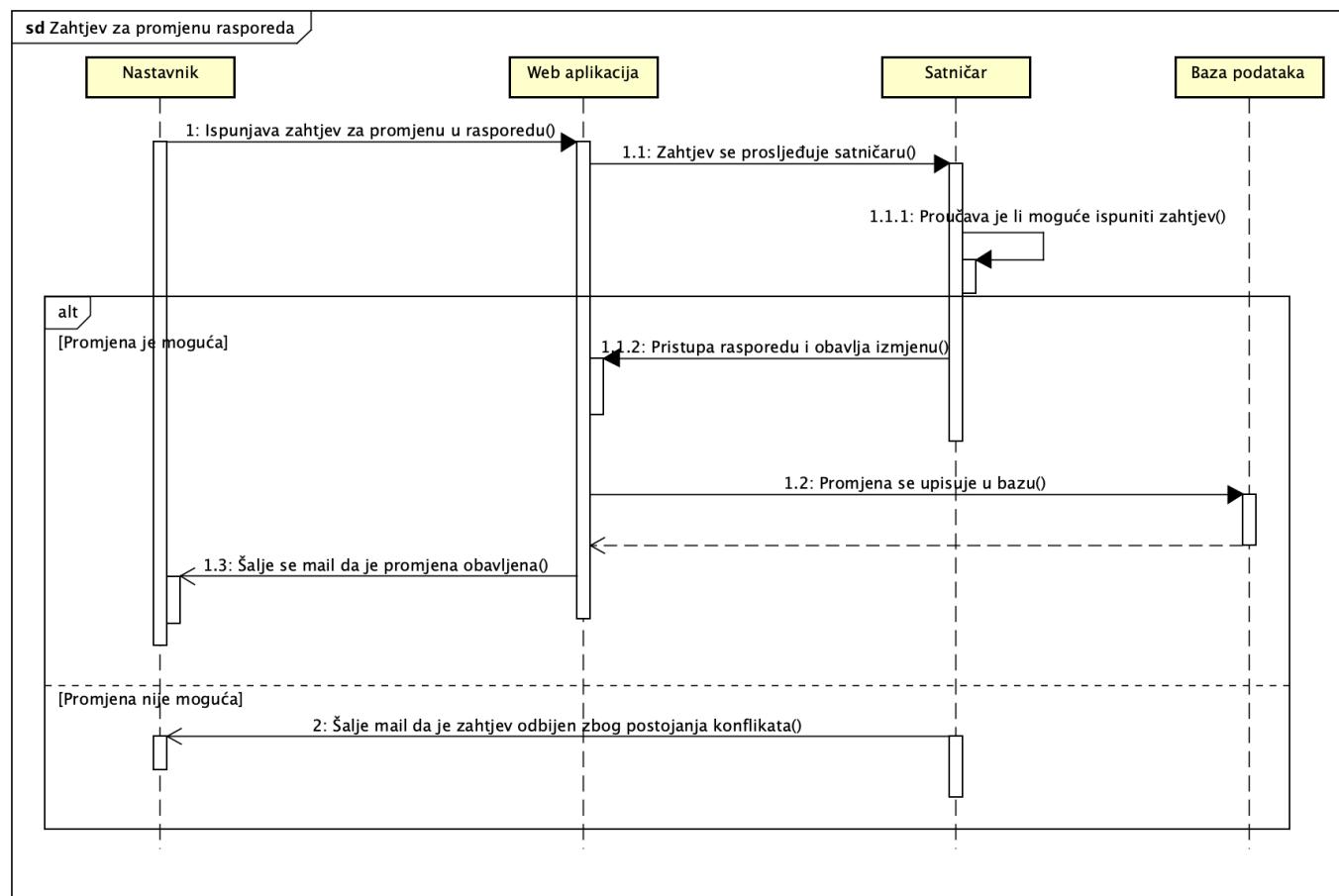
I nastavnici i učenici imaju pristup repozitoriju, učenici za pregledavanje materijala, a nastavnici za objavu materijala. Uz to učenici mogu i zatražiti potvrdu o upisu te primaju obavijesti o odlasku na praksu, a nastavnici imaju mogućnost unosa izostanaka. Za materijale i izostanke prati se statistika.

Sekvencijski dijagrami



2.1 Sekvencijski dijagram prikazuje kako izgleda registracija novih korisnika u sustav

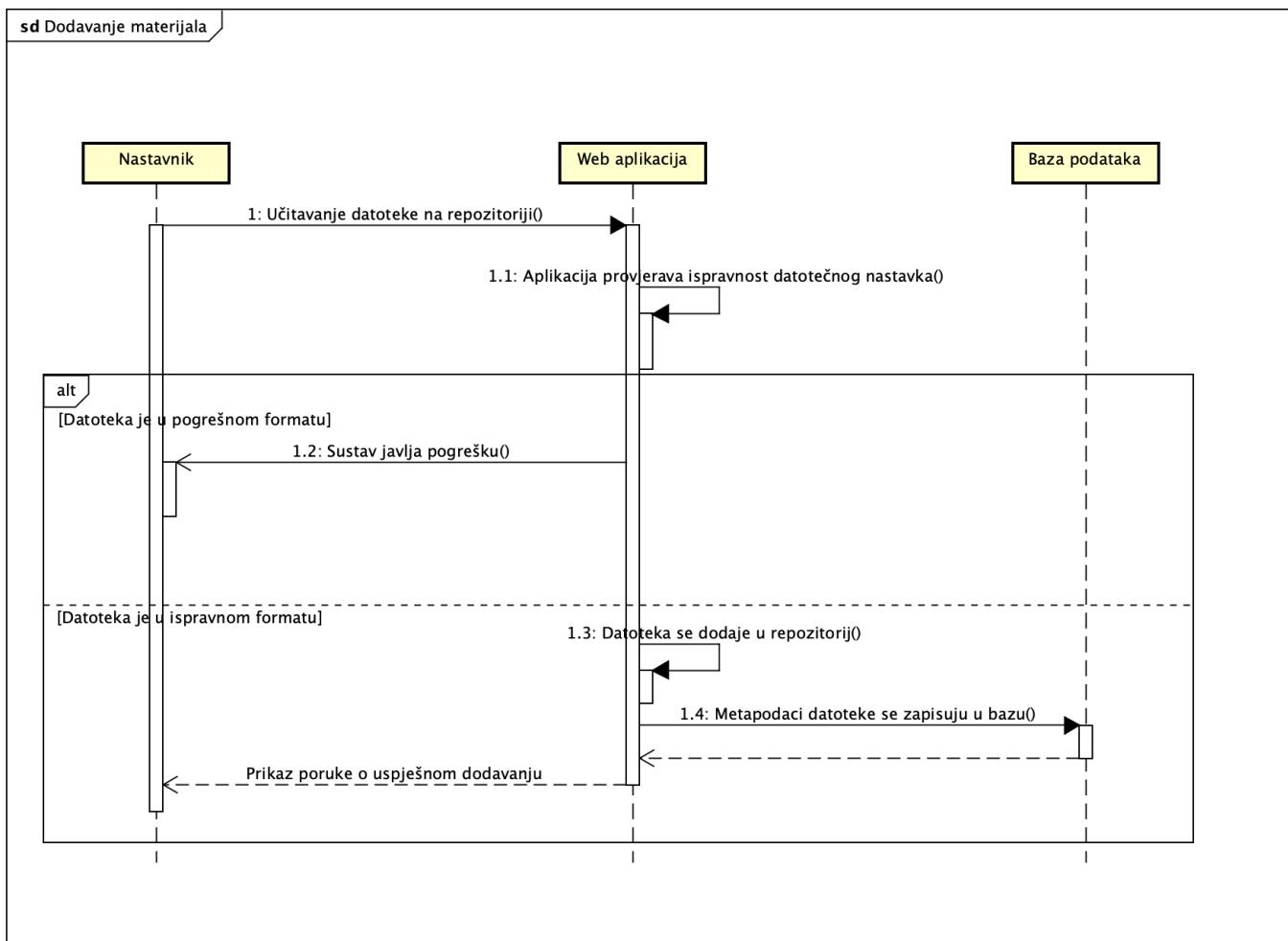
Prilikom registracije korisnik upisuje podatke, odnosno koristi svoj Google ili Microsoft račun. Te podatke zatim administrator (ravnatelj) osobno provjerava, odnosno ima listu dozvoljenog osoblja i učenika škole te provjerava nalazi li se neregistrirani korisnik na tom popisu. Ako se korisnik nalazi na popisu ravnatelj mu dodjeljuje odgovarajuću ulogu te potvrđuje njegovu registraciju. Podaci o novom korisniku upisuju se u bazu te aplikacija šalje obavijest korisniku na mail da je registracija uspjela uz zahtjev za potvrdu registracije. Jednom kad korisnik potvrdi registraciju, prikazuje mu se naslovna stranica aplikacije. Ako ravnatelj ne pronalazi neregistriranog korisnika na popisu, odbija njegovu registraciju i aplikacija šalje obavijest da registracija nije uspjela.



2.2 Sekvensijski dijagram prikazuje kako nastavnik može zatražiti od satničara da obavi promjenu u rasporedu

Jednom kad nastavnik odluči zatražiti promjenu za rasporedu ispunjava zahtjev na stranici o navedenoj promjeni. Satničar prima ispunjeni obrazac i ručno provjerava je li moguće obaviti promjenu koju je nastavnik zatražio. Ako je promjena moguća satničar pristupa rasporedu i unosi promjenu koja se onda upisuje u bazu podataka. Nakon toga aplikacija šalje obavijest o promjeni na e-mail nastavniku (i učenicima). Ako navedeni zahtjev nije bilo moguće provesti, satničar osobno putem e-maila javlja nastavniku da nije uspio obaviti

promjenu jer postoje konflikti u rasporedu.



2.3 Sekvencijski dijagram prikazuje kako izgleda kad nastavnik dodaje materijale u repozitorij

Kad nastavnik želi objaviti novi nastavni materijal pristupa stranici repozitorija i odabire predmet u sklopu kojeg želi objaviti datoteku. Zatim odabire format datoteke koji želi objaviti (.pdf, .txt, .mp4, itd.) te učitava datoteku. Aplikacija zatim provjerava odgovara li nastavak datoteka koju treba učitati odabranom nastavku. Ako se formati ne poklapaju, sustav ispisuje poruku o grešci i nastavnik ima mogućnost ponovno učitati datoteku. Inače, datoteka se dodaje u repozitorij predmeta, a njeni metapodaci spremaju se u bazu.

Aplikaciju prikazuje poruku o uspješnom dodavanju materijala.

*Metapodaci u ovom slučaju označavaju URL poveznice na materijale, ime autora, veličinu datoteke i sl., odnosno naglašavamo da se same datoteke ne spremaju direktno u bazu

Provjera uključenosti ključnih funkcionalnosti u obrasce uporabe

Obrazac uporabe	Funkcionalni zahtjev
UC1 - Registracija novih korisnika	F-001
UC2 - Prijava korisnika u sustav	F-002
UC3 - Biranje smjera	F-004, F-007
UC4 - Biranje predmeta	F-004

Obrazac uporabe	Funkcionalni zahtjev
UC5 - Generiranje razrednih grupa	F-003, F-006, F-009
UC6 - Generiranje rasporeda sati	F-004, F-005
UC7 - Objava materijala	F-010
UC8 - Pristupanje materijalima	F-011
UC9 - Izdavanje potvrda o upisu	F-007
UC10 - Upute za odlazak na praksu	F-006, F-008
UC11 - Promjene u rasporedu	F-014
UC12 - Unos izostanaka	F-013

Arhitektura sustava

Opis arhitekture

Aplikacija koristi MVC (Model-View-Controller) arhitekturu u sklopu klijent-poslužitelj stila arhitekture koja omogućava jasnu podjelu odgovornosti, jednostavno proširenje sustava, lako održavanje i visoku skalabilnost.

Stil arhitekture

U aplikaciji, klijenti (učenici, nastavnici, administrator) su odvojeni od poslužitelja (backend sustav koji obrađuje zahtjeve i pohranjuje podatke) te zato možemo reći da se radi o klijent-poslužitelj arhitekturi. Klijent-poslužitelj arhitektura omogućava skalabilnost u slučaju širenja aplikacije i jednostavnu interakciju između frontenda i backenda. Na poslužitelju se koristi MVC arhitektura kako bi se organizirala poslovna logika i korisničko sučelje. Aplikacija se dijeli na tri glavna dijela:

1. Model: Model predstavlja podatke i poslovnu logiku (npr. podaci o učenicima, predmetima itd.). Modeli obavljaju operacije nad podacima (spremanje, ažuriranje i dohvaćanje podataka iz baze)
2. View: View je odgovoran za prezentaciju podataka korisnicima (pričekivanje korisničkog sučelja) te za upravljanje korisničkim interakcijama.
3. Controller: Controller upravlja korisničkim zahtjevima, povezuje Model i View te implementira poslovnu logiku za upravljanje aplikacijom.

Podsistavi:

Podsustav za upravljanje korisnicima

- Opis: Ovaj podsustav obuhvaća registraciju, autentifikaciju (OAuth2), upravljanje korisničkim profilima i autentifikaciju putem vanjskih servisa (Google, Microsoft). Također, podržava diferencijaciju korisničkih uloga (učenici, nastavnici, administrator, ravnatelji, satničari).
- Funkcije: Prijava i registracija korisnika, autentifikacija, upravljanje korisničkim podacima i pravima pristupa.

Podsustav za upravljanje rasporedom

- Opis: Automatizirano generiranje rasporeda sati za različite smjerove i razrede, uključujući specifične predmete i praktične aktivnosti.
- Funkcije: Generiranje i prilagodba rasporeda, obavještavanje korisnika o promjenama putem e-pošte, upravljanje posebnim prostorima za praktičnu nastavu.

Podsustav za obavijesti i komunikaciju

- Opis: Zadužen za slanje obavijesti i omogućavanje komunikacije među korisnicima putem poruka.
- Funkcije: Slanje e-mail obavijesti, sustav za razmjenu poruka među korisnicima, mogućnost grupnih obavijesti za određene razrede ili smjerove.

Podsustav za upravljanje nastavnim materijalima

- Opis: Omogućuje objavljivanje i organizaciju nastavnih materijala isključivo za učenike kojima su namijenjeni.
- Funkcije: Postavljanje i distribucija materijala prema smjeru ili razredu, upravljanje pristupom za učenike i nastavnike.

Podsustav za izdavanje potvrda i administraciju dokumenata

- Opis: Automatizira izdavanje potvrda (npr. potvrde o upisu) za učenike radi olakšavanja administrativnih zadataka, poput pribavljanja pokaznih kartica za prijevoz.
- Funkcije: Izdavanje potvrda i drugih potrebnih dokumenata, integracija s dokumentacijskim standardima škole.

Podsustav za praćenje aktivnosti i statistike

- Opis: Praćenje pristupa aplikaciji i statistička obrada korištenja funkcionalnosti kako bi se razumjelo ponašanje korisnika i olakšalo donošenje administrativnih odluka.
- Funkcije: Prikupljanje i analiza podataka o pristupima, kreiranje izvještaja za administraciju i omogućavanje uvida u popularnost različitih dijelova aplikacije.

Podsustav za integraciju kartografskih i vremenskih podataka

- Opis: Podrška za terensku nastavu i prakse, uključujući prikaz vremenske prognoze i kartografskih podataka za lokaciju škole i drugih relevantnih mesta.
- Funkcije: Prikaz vremenske prognoze, integracija s kartografskim API-jem, prilagodba prema lokaciji korisnika ili aktivnosti.

Preslikavanje na radnu platformu

Za preslikavanje web-aplikacije koristit ćemo platformu Render. Render omogućuje sigurno dodavanje i upravljanje varijablama okruženja, koje su ključne za rad aplikacije u produkciji. Potrebno je definirati sljedeće varijable okruženja:

- OAUTH_CLIENT_ID i OAUTH_CLIENT_SECRET - ključevi za OAuth2 autentifikaciju putem treće strane (npr. Google, Microsoft...)
- DATABASE_URL - URL baze podataka koju koristi aplikacija. Render nudi PostgreSQL baze podataka koje se mogu koristiti, ili se može povezati s vanjskom bazom
- NODE_ENV - postavljena na "production" kako bi aplikacija radila u produkcijskom modu
- DB_HOST \
- DB_NAME \
- DB_PASSWORD - podatci za spajanje na PostgreSQL bazu
- DB_PORT /
- DB_USER /
- EMAIL_PASS - lozinka mail-a s kojeg se šalju obavijesti učenicima
- EMAIL_USER - korisničko ime mail-a s kojeg se šalju obavijesti učenicima
- GOOGLE_CLIENT_ID - potrebno za konfiguraciju Google Drive-a koji se koristi kao repozitorij
- GOOGLE_CLIENT_SECRET - potrebno za konfiguraciju Google Drive-a koji se koristi kao repozitorij
- GOOGLE_DRIVE_CREDENTIALS - potrebno za konfiguraciju Google Drive-a koji se koristi kao repozitorij
- MONGO_URI - URI za MongoDB bazu koja sprema korisničke sesije

- SESSION_SECRET - ključ za spremanje User sesija
- WEATHER_API_KEY - ključ za pristup API-u za dohvaćanje vremenske prognoze

Render također automatski pruža HTTPS, pa je potrebno osigurati redirekciju s HTTP-a na HTTPS. Ovakva konfiguracija aplikacije na platformi Render osigurava optimalne sigurnosne i konfiguracijske postavke za stabilan rad web-aplikacije.

Spremišta podataka

Za spremište podatka koristimo relacijsku bazu podataka (PostgreSQL), što znači da su podaci strukturirani u tablicama koje su međusobno povezane pomoću ključeva, što olakšava organizaciju i pretraživanje. Za modeliranje baze koristimo ERDplus kako bismo jasno prikazali odnose među entitetima, kao što su UČENIK, DJELATNIK, PREDMET itd.

Mrežni protokoli

Web-aplikacija koristi nekoliko protokola kako bi garantirala sigurnu i efikasnu komunikaciju između korisnika i servera:

1. TCP/IP (Transmission Control Protocol/Internet Protocol): TCP/IP je bazični protokol za sigurni prijenos podataka preko mreža. IP usmjerava pakete, čime osigurava da su došli do točnog odredišta, dok TCP osigurava integritet poslanih paketa, što osigurava da su došli na odredište u dobrom obliku. Svi protokoli koji rade na stranici djeluju preko TCP/IP-a, što osigurava točnost i pouzdanost prijenosa.
2. TLS/SSL (Transport Layer Security/Secure Sockets Layer): TLS/SSL su kriptografski protokoli koji služe za osiguravanje prijenosa podataka između klijenta i servera. Ovi protokoli omogućuju enkripciju, integritet i autentifikaciju podataka čime garantiraju da podatci ostanu privatni i nepromijenjeni.
3. HTTPS (HTTP Secure): HTTPS se nalazi unutar svakog web-prometa i osigurava sigurni prijenos podataka između klijentata i servera koristeći TLS/SSL enkripcije. Ovaj protokol je osnova osiguravanja privatnosti i integriteta stranice osobito pri prijavi korisnika u sustav ili poruka između korisnika.
4. OAuth 2.0: OAuth se koristi za autentifikaciju korisnika preko "third-party" poslužiteljima kao što su Google i Microsoft. Ovaj protokol omogućava sigurni pristup tako što dopušta korisnicima da se prijave koristeći postojeće račune bez upisivanja lozinke.
5. SMTP (Simple Mail Transfer Protocol): SMTP služi slanju email-ova od aplikacije do korisnika. Ovaj protokol pomaže pri slanju obavijesti, potvrda i sličnih poruka na siguran način. Kada je ovaj protokol u kombinaciji s TSL/SSL protokolima, osigurava enkripciju prometa, obavijesti i poruka.
6. WebSocket: WebSocket služi za razmjenjivanje poruka između korisnika. Pomoću ovog protokola poruke mogu biti razmijenjene bez potrebe da korisnik osvježava stranicu ili da se podatci o poruci traže sa servera. Također WebSocket protokol također koristi TLS/SSL protokole za enkripciju poruka.

Globalni upravljački tok

Web-aplikacija sadrži Node.js i Express.js kao glavne tehnologije za backend, te React za frontend. Za prijavu korisnika koristi se OAuth2.0 autentifikacija, nakon čega se korisnikov access token spremi u cookie s istekom od sat vremena. Refresh token se spremi u bazu podataka i koristi se za dobivanje novog access

tokena kako se korisnik ne bi morao često ponovno prijavljivati u stranicu. Svaki korisnik ima određena dopuštenja, koja se kontroliraju JSON Web Tokenom (provjera autentičnosti). Ako korisnik nema dopuštenje za neku radnju na stranici bit će mu odbijen pristup u slučaju pokušaja odradivanja te radnje. Korisnički podatci se spremaju u bazu podataka, kao i dokumenti koji će biti učitani na stranicu. Općeniti tok kroz aplikaciju izgleda otprilike ovako:

1. Korisnik interakcijom s frontendom šalje HTTPS zahtjev prema backendu.
2. Provjera autentifikacije: Radi se provjera ima li korisnik valjan JWT za radnju koju pokušava odraditi.
3. Ako je zahtjev ovlašten, aplikacija obrađuje podatke i izvršava potrebnu logiku (npr. vraća korisničke podatke, obrađuje prijavu, spremi dokumente, itd.)
4. Aplikacija šalje odgovor u JSON formatu frontendu koji korisniku prezentira rezultat radnje (npr. uspjela radnja, greška u izradi, odbijen pristup, itd.)

Sklopovaljivopravni zahtjevi

Neki web-browseri blokiraju kolačice aplikacije (npr. Firefox, Brave...). Za očekivani rad aplikacije u takvim web-browserima potrebno je isključiti napredne sigurnosti kolačića. Za uspješno pokretanje aplikacije potrebna je stabilna internetska veza za podršku HTTPS zahtjeva (preporučena brzina je najmanje 1 Mbps). Aplikacija podržava većinu operacijskih sustava (Windows, Linux i sl.). Za razvoj potreban je minimalno dvojezreni procesor te minimalno 1 GB RAM-a (za produkciju preporučeno je 2 GB RAM-a ili više). Potrebna pohrana je oko 500 MB, ali u rijetkim slučajevima može biti potrebno i više ako se radi s nekim većim datotekama.

Softverski zahtjevi za razvoj uključuju Node.js (14.x ili noviji), Express.js, Passport.js, JWT, bazu podataka (SQL, MongoDB) te razne knjižnice za razvoj poput cookie-parsera, body-parsera, dotenv i sl. Potrebne su i varijable okruženja koje će biti uključene putem Render platforme.

Obrazloženje odabira arhitekture

Ključni čimbenici koji su utjecali na izbor arhitekture:

1. Podrška za različite korisničke uloge i funkcionalnosti - MVC arhitektura omogućava jasno razdvajanje između logike, prikaza i interakcija, što nam omogućava upravljanje različitim ulogama i potrebama
2. Jednostavna integracija novih funkcionalnosti - modularnost MVC arhitekture omogućava razvoj novih funkcionalnosti neovisno o postojećem sustavu što čini aplikaciju fleksibilnom i lako proširivom
3. Skalabilnost i performanse - klijent-poslužitelj arhitektura omogućava da svi podaci i poslovna logika budu centralizirani na poslužitelju čime se osigurava visoka učinkovitost pri rukovanju velikim brojem korisničkih zahtjeva
4. Sigurnost - centralizirano pohranjivanje podataka na poslužitelju omogućava implementaciju snažnih sigurnosnih mjera, poput enkripcije podataka, autentifikacije korisnika i zaštite pristupa; MVC obrazac omogućava kontrolu nad interakcijama korisnika i podacima, čime se smanjuje rizik od sigurnosnih propusta

Izbor arhitekture temeljen na principima oblikovanja:

U procesu odabira arhitekture, korišteni su ključni principi oblikovanja arhitekture koji osiguravaju da aplikacija Noodle zadovoljava ciljeve skalabilnosti, održivosti i modularnosti, kao i druge zahtjeve specifične za projekt.

1. Visoka kohezija i niska povezanost - visoka kohezija znači svaka komponenta (Model, View, Controller) ima jasno definiranu odgovornost i unutar te odgovornosti obavlja slične ili povezane zadatke, a niska povezanost znači da komponente sustava nisu čvrsto povezane, što omogućava da se promjene u jednoj komponenti (npr. promjena načina prikaza podataka) obavljaju bez utjecaja na druge komponente
2. Fleksibilnost - fleksibilnost MVC arhitekture omogućava razvoj novih značajki ili proširenje postojećih s minimalnim rizikom od negativnih posljedica na ostatak sustava
3. Održivost - održiva arhitektura znači da sustav mora biti dugoročno održiv i jednostavan za prilagodbu novim zahtjevima; korištenjem MVC arhitekture, svaka komponenta sustava može se razvijati i održavati neovisno o drugima, čime se smanjuje složenost sustava i omogućuje lakše prilagođavanje novim potrebama
4. Sigurnost - korištenje klijent-poslužitelj arhitekture omogućava centralizirano upravljanje autentifikacijom, autorizacijom i enkripcijom podataka. MVC obrazac omogućava precizno upravljanje pristupom podacima i korisničkim interakcijama, čime se smanjuje mogućnost sigurnosnih prijetnji

Razmatrane alternative:

Mikroservisi

- Opis: Mikroservisna arhitektura razdvaja aplikaciju na niz manjih, samostalnih servisa koji komuniciraju putem API-ja.
- Prednosti: Olakšan razvoj, implementacija i testiranje jer je svaki servis neovisan. Skaliranje samo onih dijelova sustava koji zahtijevaju dodantne resurse. Različiti mikroservisi mogu koristiti različite tehnologije prema potrebama.
- Nedostaci: Velika složenost, osobito u velikim sustavima. Spora komunikacija između mikroservisa zbog mrežnih zahtjeva. Skupo za održavanje.
- Primjena: Koristi se u velikim, distribuiranim sustavima ili u aplikacijama koje zahtijevaju veliku skalabilnost.

Monolitna arhitektura

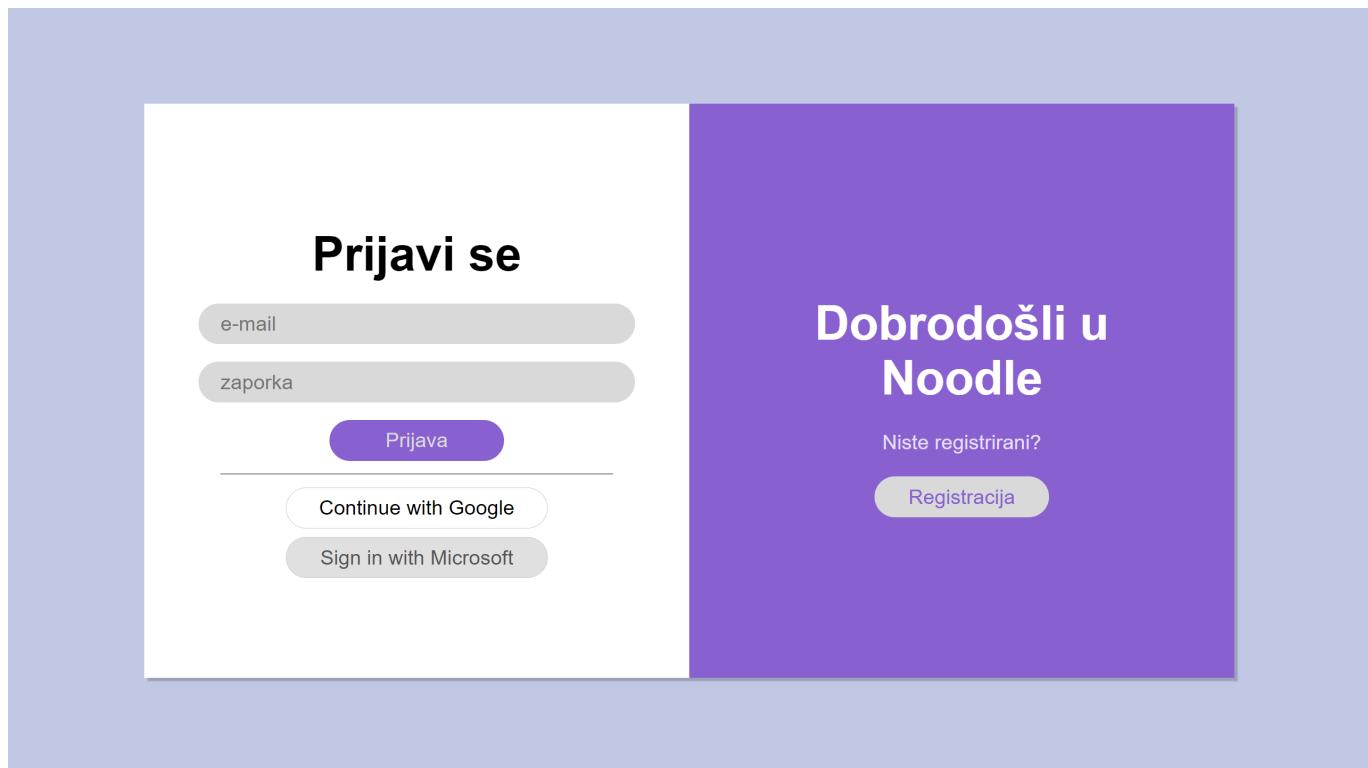
- Opis: Svi dijelovi aplikacije (frontend, backend, baza podataka) su povezani i implementirani kao jednostavni entitet.
- Prednosti: Jednostavnije za implementaciju s manjim sustavima. Brža komunikacija između dijelova jer se svi nalaze u jednom procesu.
- Nedostaci: Teško skaliranje jer je potrebno skalirati cijelu aplikaciju. Teško održavanje jer je cijeli sustav međusobno povezan.
- Primjena: U manjim sustavima, internim alatima ili aplikacijama koje ne zahtijevaju veliku skalabilnost.

Organizacija sustava na visokoj razini

- **Klijent-poslužitelj:** Aplikacija slijedi klijent-poslužitelj arhitekturu što znači da se sastoji od dva glavna dijela: **klijent** (frontend) i **poslužitelj** (backend). Klijent je odgovoran za prikaz korisničkog sučelja, slanje zahtjeva i prijem i prikaz odgovora, dok poslužitelj zaprima zahtjeve, obrađuje podatke i izvršava poslovnu logiku. Poslužitelj pristupa bazi podataka kada je potrebno, generira odgovor i šalje ga nazad klijentu u JSON formatu koji prezentira podatke na odgovarajući način. Komunikacija između

klijenta i poslužitelja odvija se preko HTTPS protokola, a za siguran pristup koristi se JWT za autorizaciju korisnika.

- **Baza podataka:** Baza podataka je sastavljena od osobnih podataka o korisnicima aplikacije, repozitorija sa svim linkovima koji se koriste, predmetima u školskom kurikulumu, prostorijama škole te rasporeda sati. Služi nam kako bismo mogli spremati sve to na lako dostupno mjesto za potrebe aplikacije kao što su dohvati tih podataka, razlikovanju korisnika po pristupu, slaganja rasporeda itd.
- **Datotečni sustav:** Baza podataka koristi spremljene linkove koji su povezani na mape na cloudu kako bi dohvaćali potrebne datoteke.
- **Grafičko sučelje:** Web aplikacija s responzivnim dizajnom prilagođenim korištenju na različitim uređajima (računala, tableti, mobilni uređaji) koja korisnicima (učenicima, nastavnicima, administratorima) omogućuje jednostavan pristup svim funkcionalnostima sustava putem intuitivnog sučelja. Frontend je implementiran korištenjem Reacta.



Organizacija aplikacije

Frontend i backend slojevi

- **Slojevi backenda omogućuju sigurnost i održivost aplikacije:**

1. Sloj za rute - sloj za rute upravlja HTTP zahtjevima i usmjerava ih na odgovarajuće kontrolere koji obrađuju logiku.
2. Kontrolni sloj - obrađuje logiku i komunicira s bazom podataka ili drugim servisima.
3. Sloj za autentifikaciju i autorizaciju - sloj odgovoran za prijavljivanje korisnika i osigurava pristup zaštićenim dijelovima aplikacije samo ovlaštenim korisnicima.
4. Sloj za Middleware - detektira zahtjeve prije nego što dođu do kontrolera i odgovore nakon što su kreirani. Omogućava dodatne funkcionalnosti kao rukovanje kolačićima, parsiranje tijela zahtjeva i sl.
5. Sloj za upravljanje okruženjem - koristi varijable okruženja kako bi osigurao sigurnost osjetljivih podataka i konfiguracija kao tokeni, port
6. Sloj za povezivanje s bazom podataka - omogućava pohranu i dohvaćanje podataka iz baze

7. Sloj za sigurnost - sloj koji koristi HTTPS i SSL (HTTP sigurnosni sloj) kako bi se osigurala sigurnost podataka i spriječili napadi

- **Slojevi frontenda omogućuju interaktivnost i optimalno korisničko iskustvo aplikacije:**

1. Sloj za prikaz - sloj za prikaz odgovoran je za vizualno predstavljanje aplikacije, koristeći alate i tehnologije za stiliziranje elemenata (npr. CSS).
2. Sloj za upravljanje komponentama - upravlja stvaranjem, organizacijom i ponovnom upotrebom komponenti kako bi bile modularne i jednostavne za održavanje, što olakšava razvoj aplikacije.
3. Sloj za komunikaciju s API-jem - sloj koji upravlja slanjem zahtjeva prema backendu za dohvaćanje ili slanje podataka te obrađuje odgovore, uključujući rukovanje pogreškama i učitavanjem podataka.
4. Sloj za optimizaciju performansi - sloj koji poboljšava korisničko iskustvo smanjujući vrijeme učitavanja i ubrzavajući aplikaciju.
5. Sloj za validaciju unosa - provjerava ispravnost podataka koje korisnici unose prije slanja na backend, kao što su ispravnost e-maila, formata broja ili obavezna polja.

Baza podataka

Za svaku školu postoji zasebna baza podataka koja uključuje osobne podatke korisnika, podatke korisnika ovisno o njihovoj ulozi, predmete iz školskog kurikuluma, repozitorij školskih materijala i obavijesti te podatke o prostorijama škole.

Opis tablica

ŠKOLA

Atribut	Tip podatka	Opis varijable
školaID	INT	Jedinstveni identifikator škole
imeŠkole	VARCHAR	Naziv škole

- primarni ključ: školaID
- **imaRaspored** - relacija koja označava da škola ima raspored
- **ima (PROSTORIJA)** - relacija koja označava koje prostorije škola ima
- **imaOnline** - relacija koja označava da škola ima svoj repozitorij odnosno obavijesti i materijale spremljene u foldere na cloud

KORISNIK

Atribut	Tip podatka	Opis varijable
OIB	VARCHAR	Jedinstveni identifikator korisnika
ime	VARCHAR	Ime korisnika
prezime	VARCHAR	Prezime korisnika
datumrod	DATE	Datum rođenja korisnika
spol	VARCHAR	Spol korisnika

Atribut	Tip podatka	Opis varijable
adresa	VARCHAR	Adresa korisnika
email	VARCHAR	Elektronička pošta korisnika
zaporka	VARCHAR	Zaporka korisnika
školaID	INT	Jedinstveni identifikator škole

- primarni ključ: OIB
- strani ključ: školaID
- predstavlja bilo kojeg korisnika koji se registrira/prijavljuje na stranici škole
- **sudionik** - relacija koja označava sve korisnike koji se nalaze u školi (gosti, učenici i djelatnici)

GOST

Atribut	Tip podatka	Opis varijable
gostID	VARCHAR	Jedinstveni identifikator gosta
datumPristupa	TIMESTAMP	Označava kad je gost pristupio stranici
OIB	VARCHAR	Jedinstveni identifikator korisnika
		<ul style="list-style-type: none"> • primarni ključ: gostID • strani ključ: OIB • predstavlja podtip registrirane osobe koja čeka dozvolu za ulazak na stranicu

DJELATNIK

Atribut	Tip podatka	Opis varijable
djelatnikID	VARCHAR	Jedinstveni identifikator
mobBroj	VARCHAR	Broj mobitela djelatnika
razred	VARCHAR	Popis razreda kojima djelatnik predaje
status	VARCHAR	Određuje uloga djelatnika
razrednik	VARCHAR	Određuje je li djelatnik razrednik nekom razredu
OIB	VARCHAR	Jedinstveni identifikator korisnika

- primarni ključ: djelatnikID
- strani ključ: OIB
- predstavlja podtip registriranog korisnika koji je djelatnik u školi
- djelatnik može biti: ravnatelj, profesor i satničar (određeno atributom status)
- podtipovi djelatnika imaju različite dozvole pristupa, ovisno što im je potrebno
- jedna osoba može biti svaki podtip odjednom po potrebi
- ravnatelj može biti samo jedan, a odobrava ga administrator te ima pristup odobravanju dalnjih registriranih osoba i dodjeljivanju uloga te sve ostale mogućnosti administratora unutar škole
- satničar ima pristup promjeni rasporeda i objavljivanju obavijesti

- profesor ima pristup objavljivanju obavijesti i materijala te podatcima svakog učenika kojem je razrednik
- razrednik** - relacija koja označava kojem učeniku je profesor razrednik
- predaje** - relacija koja označava predmete koje profesor predaje

UČENIK

Atribut	Tip podatka	Opis varijable
učenikID	VARCHAR	Jedinstveni identifikator učenika
razred	VARCHAR	Označava kojem razredu učenik pripada
škGod	VARCHAR	Označava godinu početka školovanja učenika
OIB	VARCHAR	Jedinstveni identifikator korisnika
smjer	VARCHAR	Koji smjer pohađa učenik

- primarni ključ: učenikID
- strani ključ: OIB
- predstavlja podtip registriranog korisnika koja pohađa školu
- jelmao** (IZOSTANAK) - relacija koja označava je li i koliko je učenik imao izostanaka

IZOSTANAK

Atribut	Tip podatka	Opis varijable
izostanakID	VARCHAR	Jedinstveni identifikator izostanka
izostanakSat	VARCHAR	Označava s kojeg je sata učenik izostao
izostanakOpis	VARCHAR	Opisuje uzrok izostanka
izostanakDatum	DATE	Označava datum izostanka
izostanakStatus	VARCHAR	Označava je li izostanak opravdan ili ne
učenikID	VARCHAR	Jedinstveni identifikator učenika

- primarni ključ: izostanakDatum
- strani ključ: učenikID
- veže se za učenika i označava svaki izostanak zasebno

PREDMET

Atribut	Tip podatka	Opis varijable
predmetID	INT	Jedinstveni identifikator predmeta
imePredmet	VARCHAR	Ime predmeta
brojSatova	VARCHAR	Broj sati koje predmet ima u tjednu
brojLab	VARCHAR	Broj laboratorijskih vježbi koje predmet ima u tjednu

Atribut	Tip podatka	Opis varijable
školaID	INT	Jedinstveni identifikator škole
smjer	VARCHAR	Na kojem smjeru se drži predmet
godine	VARCHAR	Na kojoj je godini predmet

- primarni ključ: predmetID
- strani ključ: školaID
- predstavlja predmet koji se drži u školi
- **uKurikulumu** - relacija koja označava koje predmete škola drži
- **predajeSe** - relacija koja označava da se predmeti predaju u rasporedu

RASPORED

Atribut	Tip podatka	Opis varijable
terminID	INT	Jedinstveni identifikator rasporeda
vrijeme	TIME	Vrijeme kada se održava sat
dan	INT	Dan kada se održava sat
razred	VARCHAR	Razred kojem je dodijeljen raspored
oznaka	VARCHAR	Jedinstveni identifikator prostorije
imePredmet	VARCHAR	Ime predmeta
školaID	INT	Jedinstveni identifikator škole

- predstavlja raspored koji se dodjeljuje učenicima i profesorima
- primarni ključ: terminID
- strani ključ: školaID, oznaka
- **uSobi** - relacija koja označava u kojoj se učionici odvija raspored

PROSTORIJA

Atribut	Tip podatka	Opis varijable
oznaka	VARCHAR	Jedinstveni identifikator
kapacitet	VARCHAR	Označava kapacitet prostorije
tipProstorije	VARCHAR	Označava je li prostorija učionica ili dvorana
školaID	INT	Jedinstveni identifikator škole

- predstavlja prostorije koje se nalaze u školi
- primarni ključ: oznaka
- strani ključ: školaID

REPOZITORIJ

Atribut	Tip podatka	Opis varijable
repID	INT	Jedinstveni identifikator repozitorija
imeRep	VARCHAR	Naziv repozitorija
metaData	VARCHAR	Dodatne informacije poput opisa ili tagova mapa
unutar_repID	INT	Jedinstveni identifikator podrepositorija
školaID	INT	Jedinstveni identifikator škole

- primarni ključ: repID
- strani ključ: školaID, unutar_repID
- predstavlja repozitorij škole
- **unutar** - relacija koja označava da je svaki repozitorij zapravo folder na cloudu i može imati svoje subfoldere

LINK

Atribut	Tip podatka	Opis varijable
autor	VARCHAR	Predstavlja korisnika koji je objavio materijale
razred	VARCHAR	Predstavlja razred kojem su obajavljeni materijali
brojPregleda	VARCHAR	Označava koliko se puta pristupilo dokumentu
datumObjave	TIMESTAMP	Datum i vrijeme objave dokumenta
linkTekst	VARCHAR	Jedinstveni identifikator linka
repID	VARCHAR	Jedinstveni identifikator repozitorija

- primarni ključ: linkTekst
- strani ključ: repID
- predstavlja sve što se može nalaziti u repozitoriju odnosno DATOTEKA ili OBAVIJEST
- **sadrži** - relacija koja označava u kojem se folderu nalazi LINK

OBAVIJEST

Atribut	Tip podatka	Opis varijable
naslov	VARCHAR	Naslov obavijesti
tekst	VARCHAR	Tekst obavijesti
linkTekst	VARCHAR	Jedinstveni identifikator linka

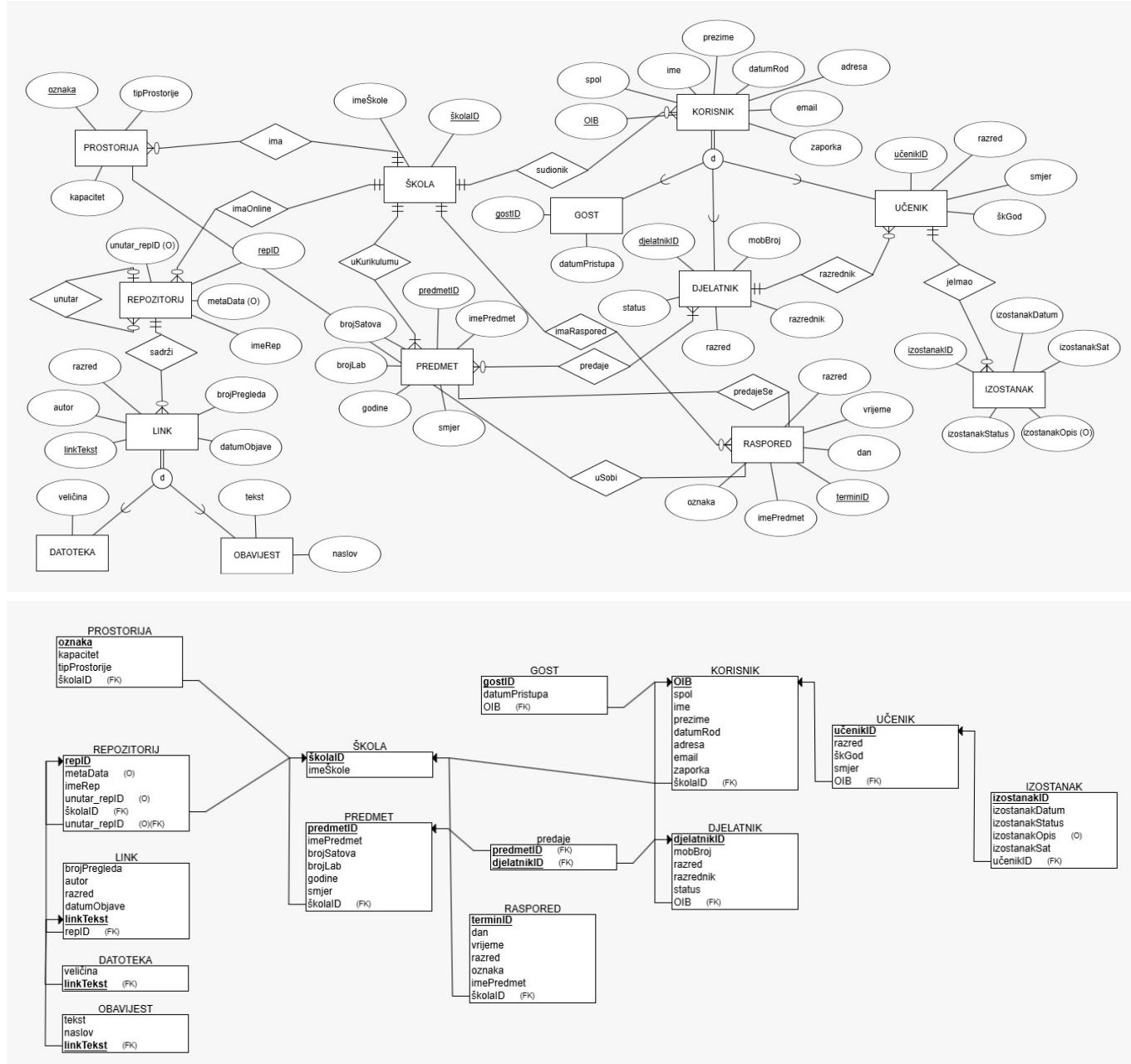
- primarni ključ: linkTekst
- strani ključ: linkTekst
- predstavlja podtip linka koji je obavijest

DATOTEKA

Atribut	Tip podatka	Opis varijable
veličina	VARCHAR	Veličina datoteke u MB
linkTekst	VARCHAR	Jedinstveni identifikator linka

- primarni ključ: linkTekst
- strani ključ: linkTekst
- predstavlja podtip linka koji je datoteka

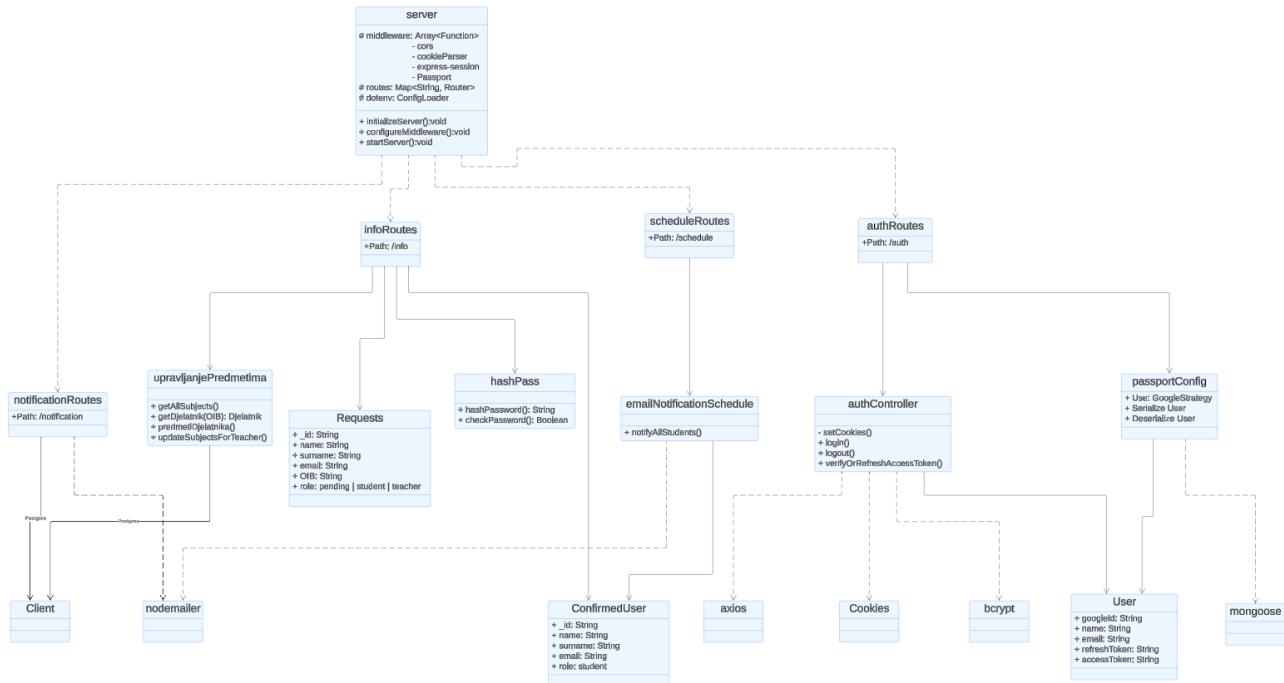
Dijagram baze podataka



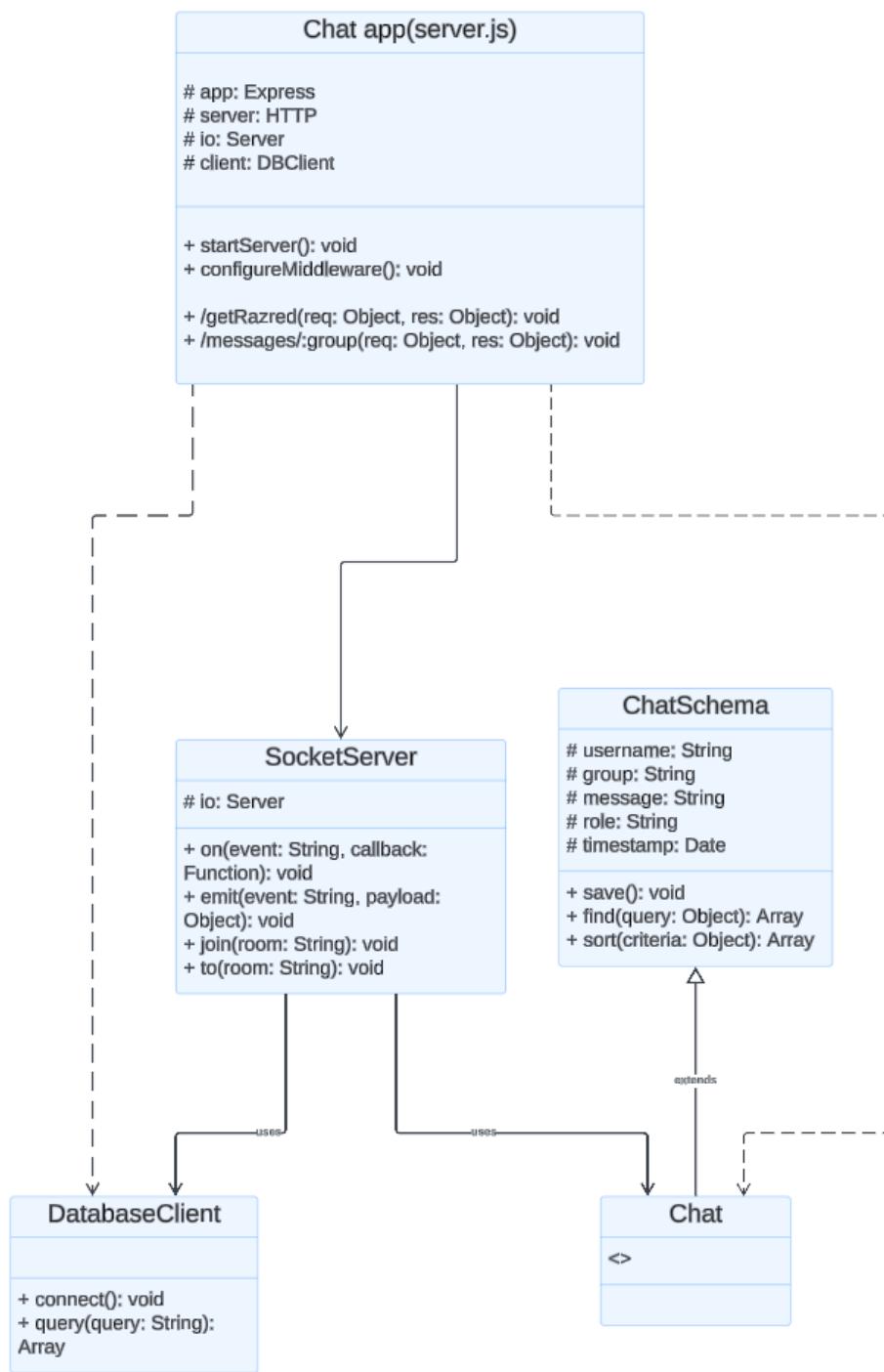
Dijagram razreda

Backend dijagrami razreda

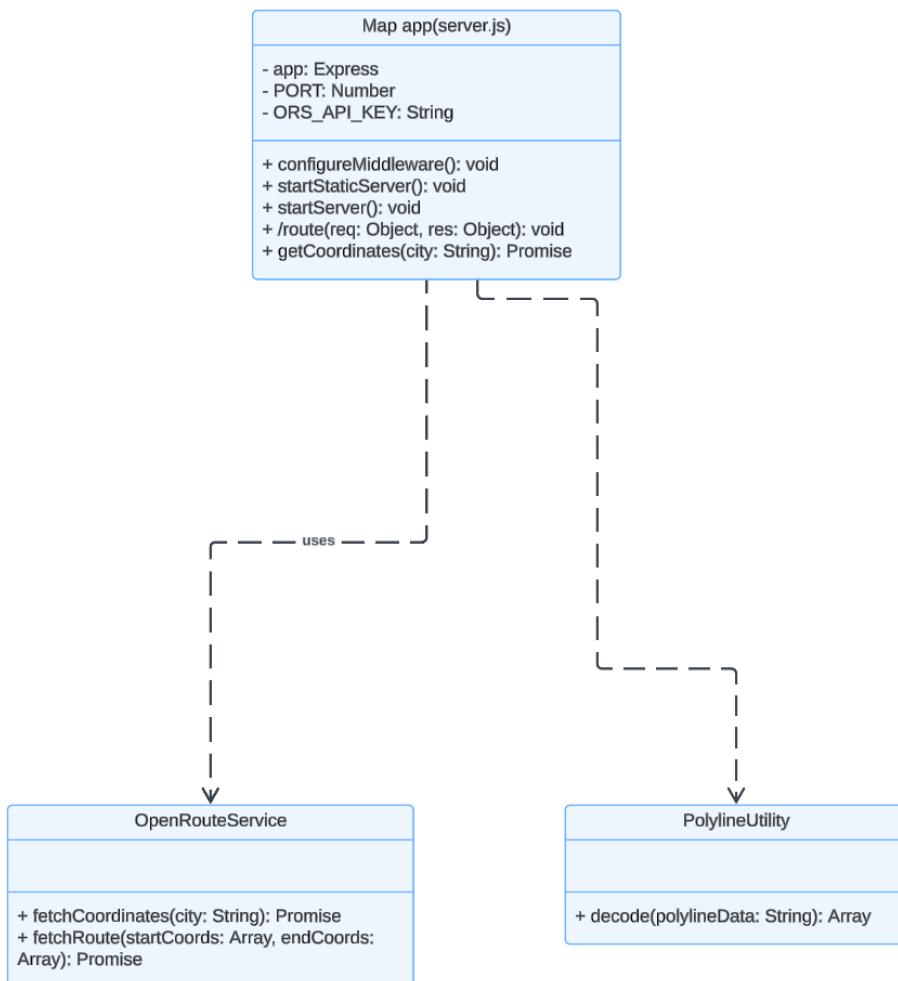
Auth dijagram razreda:



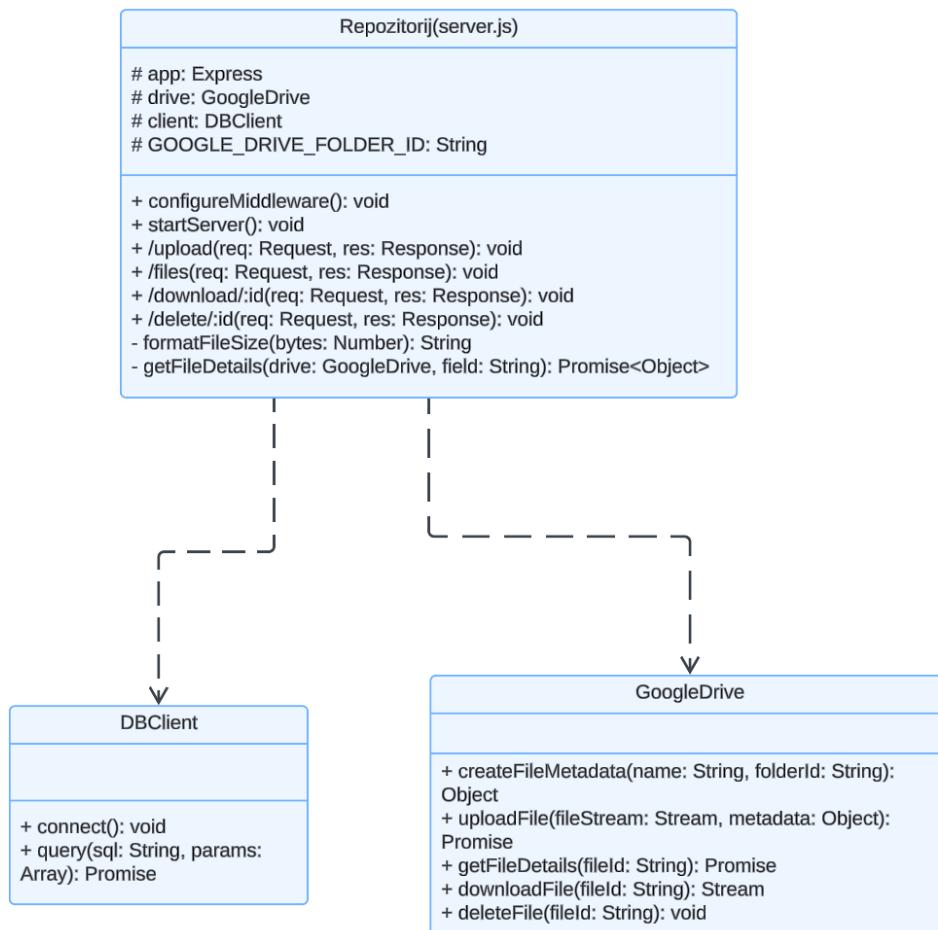
Chat dijagram razreda:



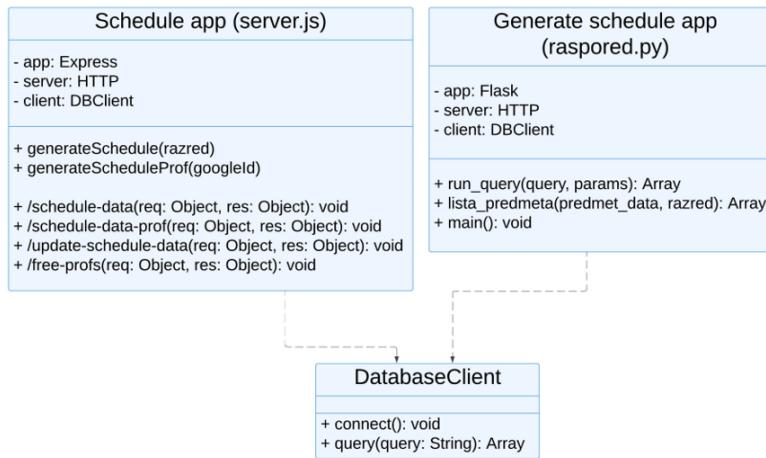
Map dijagram razreda:



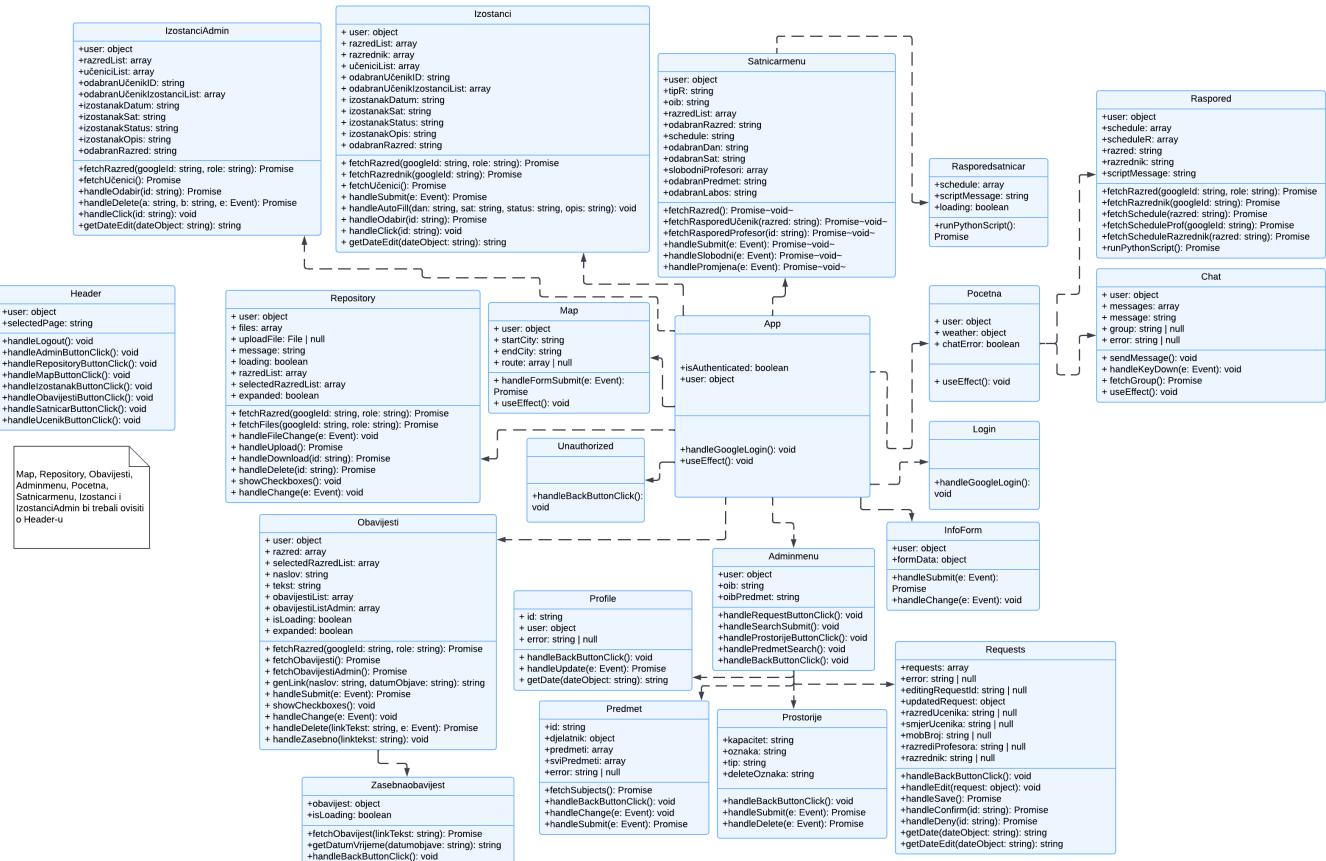
Repozitorij dijagram razreda:



Raspored dijagram razreda:



Frontend dijagram razreda:

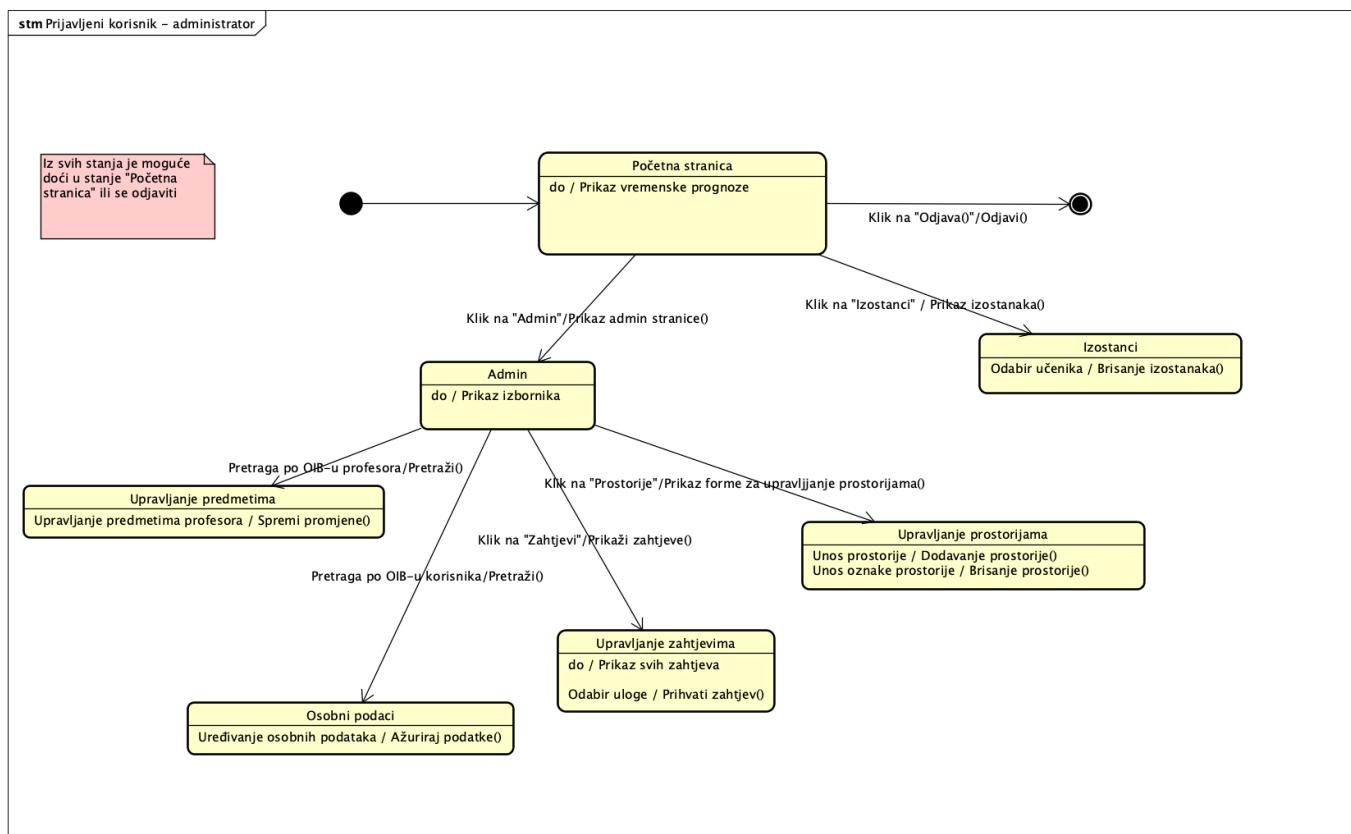


Dinamičko ponašanje aplikacije

UML dijagrami stanja

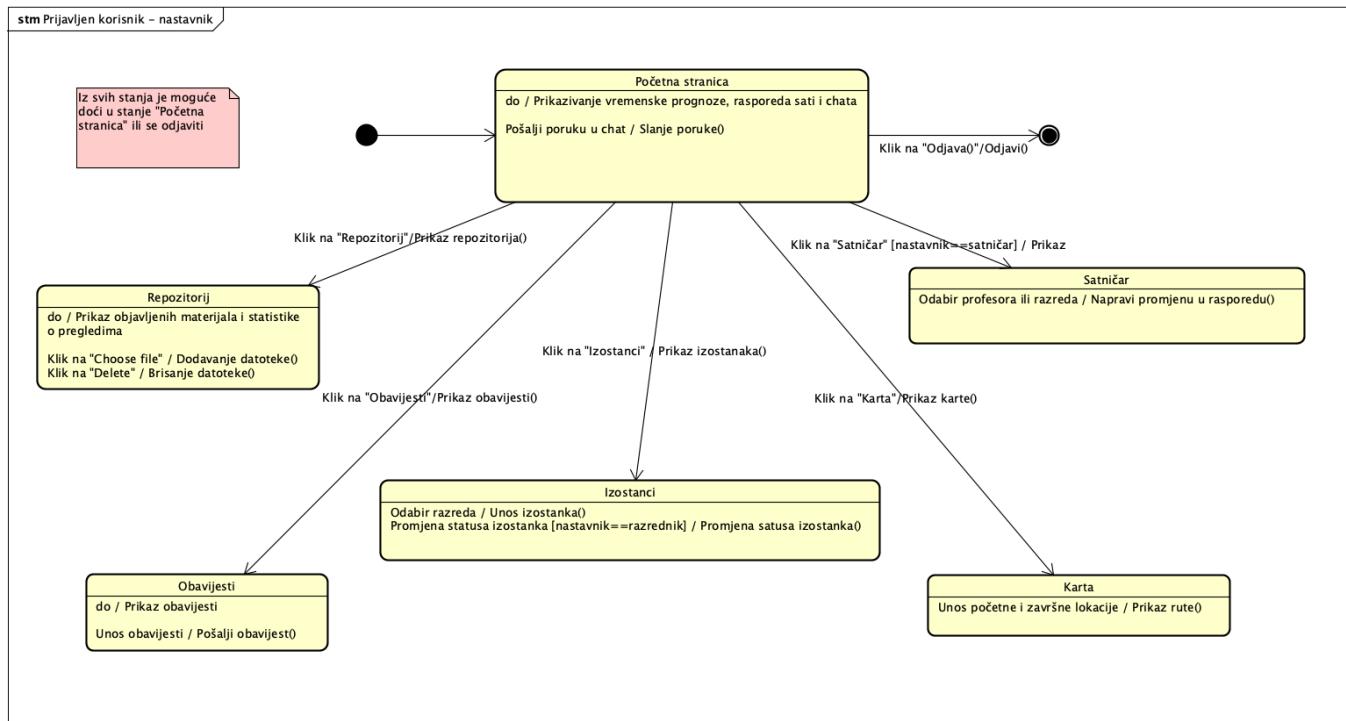
Za aplikaciju možemo promatrati tri dijagrama stanja: iz perspektive administratora, korisnika ili učenika

UML dijagram stanja 1 - Prijavljeni korisnik (administrator)



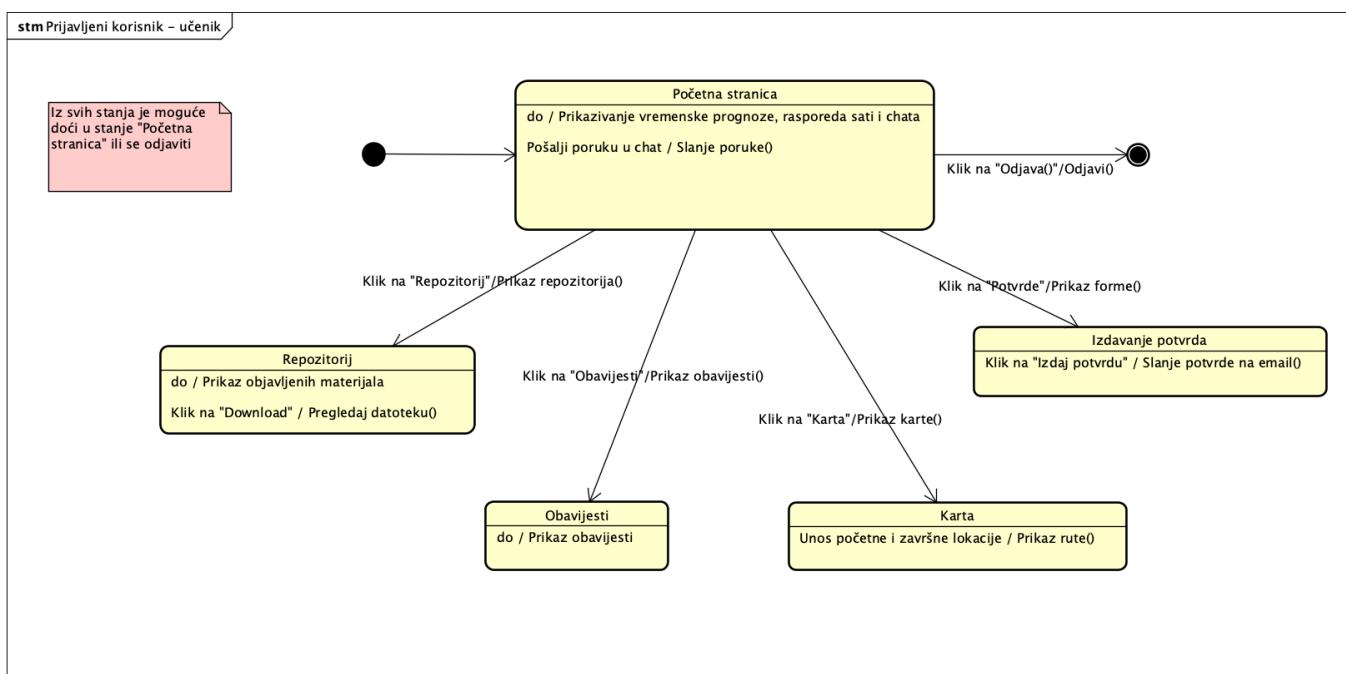
Ako promatramo sustav iz perspektive prijavljenog korisnika koji ima ulogu administratora, prikazuje se početna stranica s prikazom vremenske prognoze. Klikom na gumb "Admin" prikazuje se izbornik pomoću kojeg administrator upravlja zahtjevima, prostorijama, predmetima profesora i uređivanjem profila ostalih korisnika. Upravljanje zahtjevima omogućuje dodavanje uloga korisnicima koji su se registrirali. Ako se radi o učenicima, moguće je odabrati smjer i razred. Ako se radi o nastavnicima, moguće je odabrati kojim razredima predaje i ima li ulogu razrednika. Za prostorije je moguće unositi nove prostorije (odabir kapaciteta, označke i tipa) ili obrisati postojeće. Također je moguće pretražiti nastavnika po OIB-u i odabrati predmete koje nastavnik predaje te je moguće pretraživati korisnike po OIB-u i uređivati njihove osobne podatke.

UML dijagram stanja 2 - Prijavljeni korisnik (nastavnik)



Ako promatramo sustav iz perspektive prijavljenog korisnika koji ima ulogu nastavnika, na početnoj se stranici uz prognozu prikazuje i raspored nastavnika te ukoliko je razrednik prikazuje se i raspored njegovog razreda te chat. Klikom na gumb "Repozitorij" nastavnici mogu objavljivati materijale u odabrane razrede i pratiti statisku pregleda. Klikom na gumb "Obavijesti" prikazuju se sve obavijesti i moguće je unijeti i objaviti novu obavijest. Klikom na gumb "Izostanci" nastavnik može zapisati izostanke učenika iz odabranog razreda, ali samo razrednik može promijeniti status izostanka u opravдан/neopravdan. Nastavnik ima mogućnost korištenja karte, odnosno prikaz odabrane rute. Ukoliko je nastavnik satničar, klikom na gumb "Satničar" omogućeno mu je unošenje promjena u odabrani raspored.

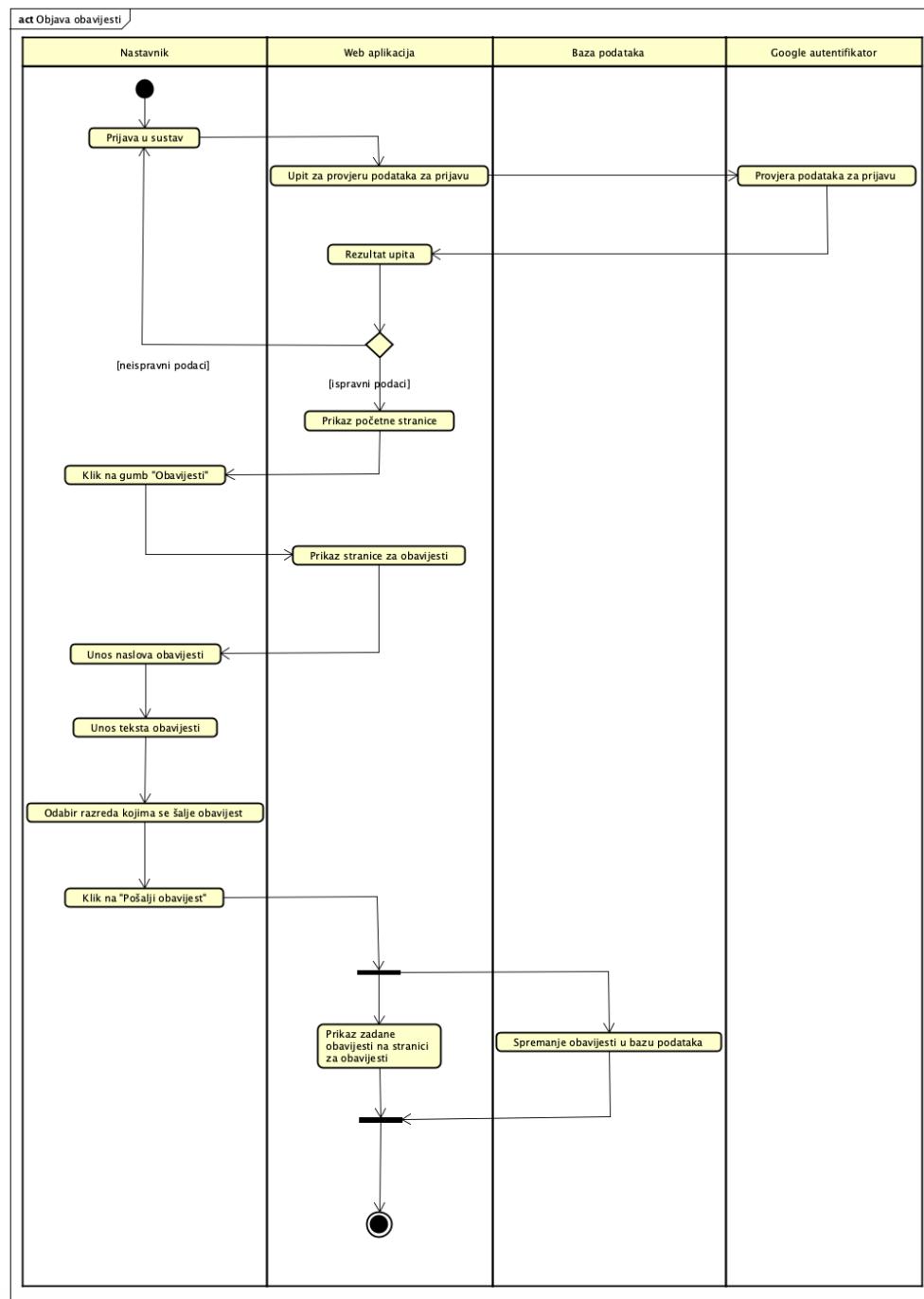
UML dijagram stanja 3 - Prijavljeni korisnik (učenik)



Zadnji dijagram opisuje sustav iz perspektive učenika. Početna stranica je ista kao i kod nastavnika, samo što se prikazuje raspored samo za razred unutar kojeg je učenik. Učenik također ima mogućnost slanja

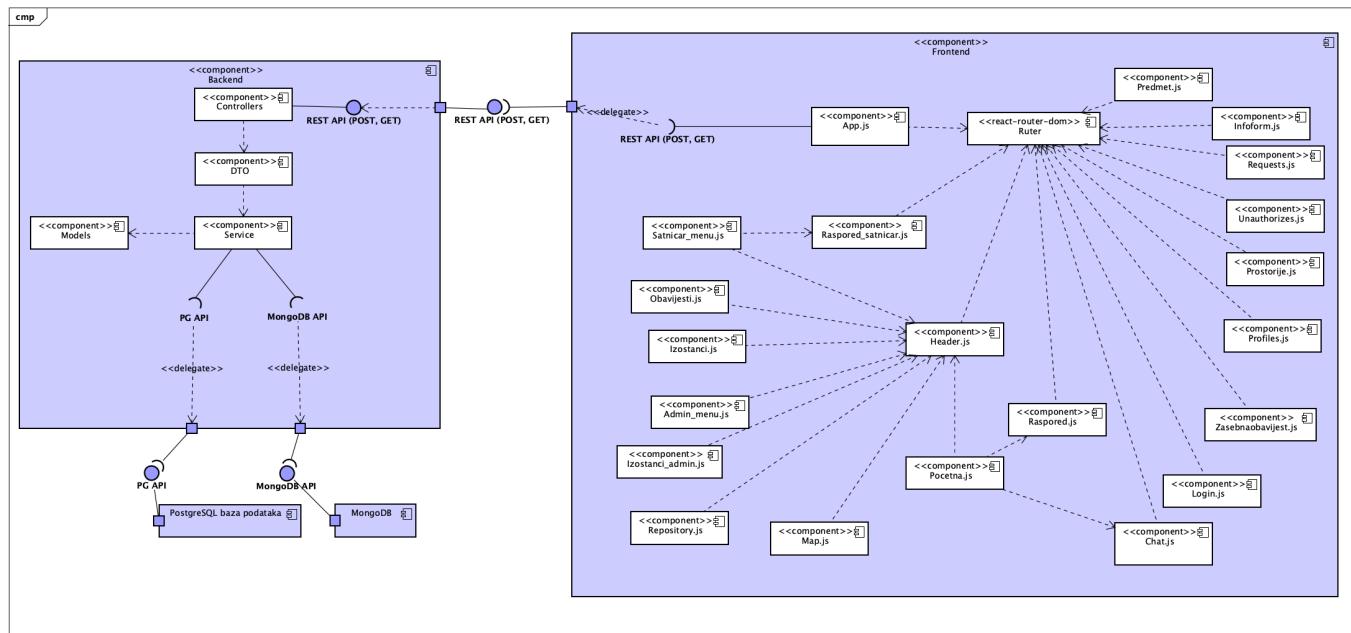
poruka unutar chata i pregleda repozitorija, ali nema mogućnost objave materijala, već ih može samo pregledavati. Isto tako može pregledavati obavijesti, ali ih ne može objavljivati te klikom na gumb "Potvrde" može odabrati izdavanje potvrde o upisu koja se onda šalje na email.

UML dijagrami aktivnosti



Zadani dijagram aktivnosti opisuje aktivnost nastavnika kad stavlja novu obavijest na stranicu. Prvo se nastavnik prijavljuje u sustav pomoću Google autentifikatora koji provjerava ispravnost podataka za prijavu. Ako podaci nisu ispravni nastavnik se opet mora prijaviti. Nakon uspješne prijave, prikazuje se početna stranica te se nastavniku klikom na gumb "Obavijesti" prikazuje stranica za obavijesti na kojoj se nalazi forma koju nastavnik popunjava. Potrebno je unijeti naslov i tekst obavijesti te odabrati razred/e koji primaju obavijest. Nakon unosa podataka nastavnik klikom na gumb "Pošalji obavijest" šalje obavijest koja se zatim prikazuje na stranici i paralelno se spremi u bazu podataka i to označava kraj aktivnosti.

Dijagram komponenata

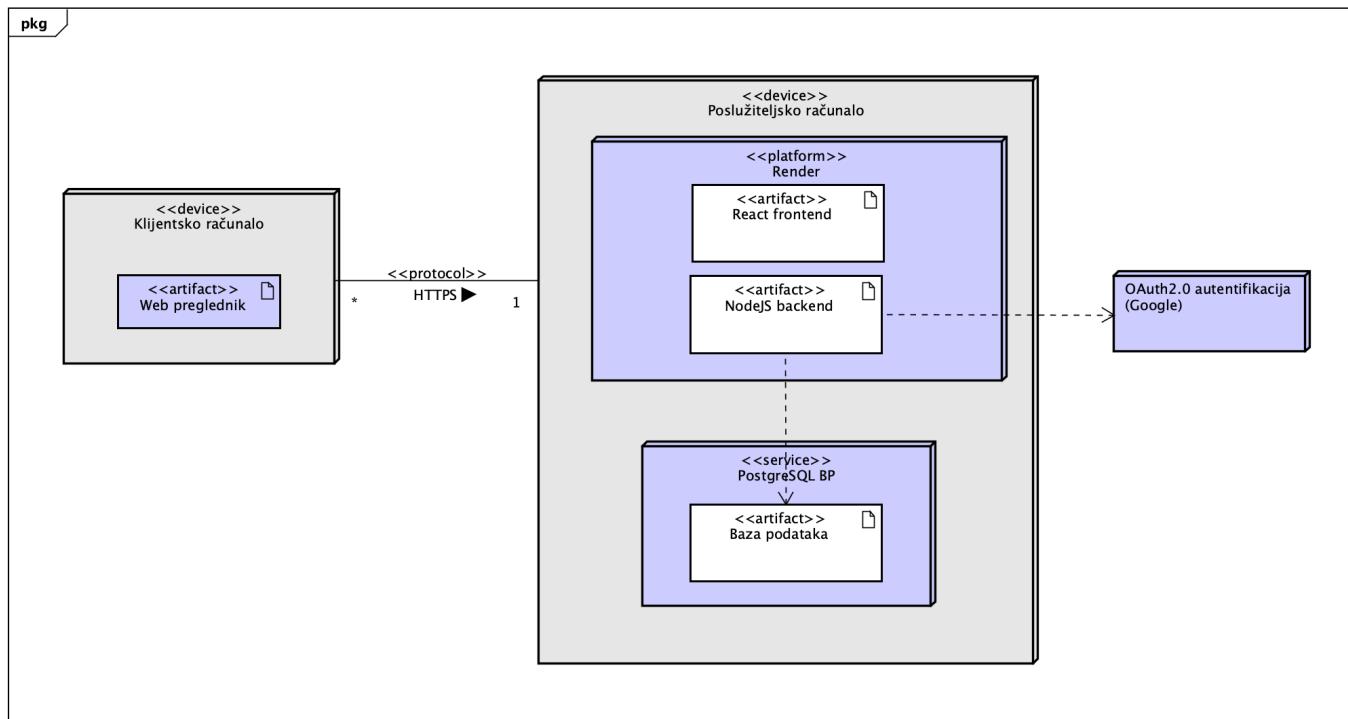


Dijagram komponenata prikazuje internu strukturu i međuvisnost komponenti web-aplikacije. Na dijagramu su prikazane tri glavne cjeline aplikacije: frontend, backend i baze podataka. Korisnik putem web-preglednika započinje akcije koje fronted obrađuje.

Frontend aplikacije je izrađen kao skup React komponenti koje predstavljaju razne funkcionalnosti aplikacije, od kojih je najbitnija React-router-dom. React-router-dom ključna je komponenta koja omogućuje navigaciju između različitih stranica i ruta aplikacije. Ostale komponente poput Raspored.js, Pocetna.js i slično direktno komuniciraju s backendom putem REST API-ja HTTP zahtjevima. Header.js je centralna komponenta koja služi kao navigacijski element unutar aplikacije.

Backend aplikacije strukturiran je kao skup komponenti koje upravljaju logikom aplikacije i interakcijom s bazom podataka. Controlleri obrađuju zahtjeve koje šalje fronted (putem REST API-ja) te prosleđuju podatke prema DTO sloju za validaciju i transformaciju podataka. Service sloj implementira logiku aplikacije i delegira upite prema odgovarajućim API-jevima. Bazi PostgreSQL se pristupa putem PG API-ja, baza se koristi za strukturirane podatke unutar sustava (poput korisnika, učionica, informacija o školi itd.). MongoDB bazi pristupa se putem MongoDB API-ja, ta baza koristi se za nestrukturirane podatke, poput spremanja korisničkih sesija ili chat poruka.

Dijagram razmještaja



Dijagram razmještaja prikazuje arhitekturu razmještaja sustava. Prikazana je arhitektura klijent-poslužitelj na kojoj se zasniva sustav. Klijenti koriste web preglednik za pristupanje aplikaciji. Komunikacija klijenta i poslužitelja se odvija putem HTTPS protokola. Jedan poslužitelj može posluživati više klijenata, dok se klijent može spojiti na samo jedan poslužitelj. Za posluživanje frontenda, backenda i baze podataka koristi se platforma Render. Backend koristi vanjski sustav za autentifikaciju Oauth2.0 za registraciju preko Google računa.

Ispitivanje komponenti

1. Komponenta: Google oAuth login

- Ispitivana funkcionalnost: korisnici se pokušavaju prijaviti putem Google oAuth-a

Ispitni slučaj 1: Login korisnika bez zapamćene lozinke i mail-a

- Ulazni podaci: Gmail i lozinka računa korisnika
- Očekivani rezultat: Korisnik je uspješno prijavljen i prebačen na formu za registraciju u školu
- Dobiveni rezultat: Prolaz
- Postupak ispitivanja:

- Korisnik otvara login stranicu.
- Korisnik klikne na gumb "Continue with Google".
- Korisnik klikne na "Upotrijebite neki drugi račun".
- Korisnik upisuje svoj mail.
- Korisnik upisuje šifru za svoj mail.

The screenshot displays the Noodle Testing interface with a recorded Selenium script. The script consists of 12 commands, each with a target element and a value. The commands include opening a browser, clicking on a Google login button, and entering a password. The recorded log shows the execution of these commands, and the reference log shows the expected execution. The interface includes a toolbar at the top, a command list on the left, and a detailed log table on the right.

Command	Target	Value
1. open	login	
2. set window size	1280x1372	
3. click	css=google-login-button	
4. click	css=AyTfBd	
5. type	id=password	jurajcic0@gmail.com
6. click	css=VFPykd-kz2d-mvPkd3d-OVXEx-TvB5d-L	
7. mouse over	css=VFPykd-Lgb5e-OVXEx-kBQjd > VFPykd-vQzfId	
8. mouse out	css=VFPykd-Lgb5e-OVXEx-kBQjd > VFPykd-vQzfId	
9. click	css=VFPykd-Lgb5e-OVXEx-kBQjd > VFPykd-vQzfId	
10. type	name=Passwd	
11. click	css=VFPykd-Lgb5e-OVXEx-kBQjd > VFPykd-vQzfId	
12. click	css=VFPykd-kz2d-mvPkd3d-OVXEx-TvB5d-LUmc > VFPykd-RLmub	

Run: 1 | Failure: 0

Log | Reference

- open on login OK
- setWindowSize on 1280x1372 OK
- click on google-login-button OK
- click on css=AyTfBd OK
- type on id=password with value jurajcic0@gmail.com OK
- click on css=VFPykd-kz2d-mvPkd3d-OVXEx-TvB5d-L OK
- mouseOver on css=VFPykd-Lgb5e-OVXEx-kBQjd > VFPykd-vQzfId OK
- mouseOut on css=VFPykd-Lgb5e-OVXEx-kBQjd > VFPykd-vQzfId OK
- click on name=Passwd OK
- type on name=Passwd with value JurajCic0 OK
- click on css=VFPykd-Lgb5e-OVXEx-kBQjd > VFPykd-vQzfId OK
- Try to find css=VFPykd-kz2d-mvPkd3d-OVXEx-TvB5d-LUmc - OK
- click on id=tost OK
- click on id=tost OK

"Login-košar-nove-mail" completed successfully

Ispitni slučaj 2: Login korisnika sa zapamćenom lozinkom

- Ulazni podaci: Gmail i lozinka računa korisnika
- Očekivani rezultat: Korisnik je uspješno prijavljen i prebačen na formu za registraciju u školu
- Dobiveni rezultat: Prolaz
- Postupak ispitivanja:

- Korisnik otvara login stranicu.
- Korisnik klikne na gumb "Continue with Google".
- Korisnik odabire svoj mail.

4. Korisnik je automatski preusmjeren na formu za registraciju.

The screenshot shows the Selenium IDE interface with the following details:

- Project:** Noodle Testing*
- Test Case:** login-postojeći-email*
- Test Steps:**

	Command	Target	Value
1	✓ open	/login	
2	✓ set window size	1265x1372	
3	✓ click	css=.google-login-button	
4	✓ click	css=.aZvCDf:nth-child(1) .pGzURd	
5	✓ click	css=.VfPpkd-ksKsZd-mWPk3d-OWXExe-Tv8l5d-lJfZMc > .VfPpkd-RLmnJb	
- Run Status:** Runs: 1 Failures: 0
- Log Panel:**

Log	Reference	Date
1. open on /login	OK	19:43:07
2. setWindowSize on 1265x1372	OK	19:43:08
3. click on css=.google-login-button	OK	19:43:10
4. click on css=.aZvCDf:nth-child(1) .pGzURd	OK	19:43:12
5. Trying to find css=.VfPpkd-ksKsZd-mWPk3d-OWXExe-Tv8l5d-lJfZMc > .VfPpkd-RLmnJb...	OK	19:43:15
6. click on css=.logout-gumb	OK	19:43:31
7. close	OK	19:43:34
'login-postojeći-email' completed successfully		19:43:34

Ispitni slučaj 3: Korisnik se pokuša prijaviti s krivom šifrom

- Ulazni podaci: Gmail i (kriva) lozinka računa korisnika
- Očekivani rezultat: Korisnik dobiva obavijest da je lozinka pogrešna i neka pokuša ponovno
- Dobiveni rezultat: Prolaz
- Postupak ispitivanja:

1. Korisnik otvara login stranicu.
2. Korisnik klikne na gumb "Continue with Google".

3. Korisnik odabire svoj mail i upisuje krivu lozničku.

Selenium IDE - Noodle Testing*

Project: Noodle Testing*

Executing: login-kriva-lozinka*

https://noodle-frontend.onrender.com

Command	Target	Value
1 ✓ open	/login	
2 ✓ set window size	1265x1372	
3 ✓ click	css=.google-login-button	
4 ✓ click	css=.aZvCDf:nth-child(1).pGzURd	
5 ✓ type	name=Passwd	cizicj
6 ✓ click	css=S7xv8	
7 ✓ click	css=.VfPpkd-LgbsSe-OWXEXe-k8QpJ > .VfPpkd-vQzf8d	

Command Target Value

Target

Value

Description

Runs: 1 Failures: 0

Log Reference

Running 'login-kriva-lozinka'	19:53:39	
1. open on /login	OK	19:53:39
2. setWindowSize on 1265x1372	OK	19:53:41
3. click on css=.google-login-button	OK	19:53:43
4. click on css=.aZvCDf:nth-child(1).pGzURd	OK	19:53:45
5. type on name=Passwd with value cizicj	OK	19:53:48
6. click on css=S7xv8	OK	19:53:50
7. click on css=.VfPpkd-LgbsSe-OWXEXe-k8QpJ > .VfPpkd-vQzf8d	OK	19:53:52
'login-kriva-lozinka' completed successfully		19:53:52

Prijava putem Googlea

Juraj Čižić

juricacizic@gmail.com

Unesite zaporku

! Zaporka je pogrešna. Pokušajte ponovo ili kliknite "Zaboravili ste zaporku?" da biste je poništili.

Prikaži zaporku

[Pokušajte na neki drugi način](#)

[Dalje](#)

Ispitni slučaj 4: Korisnik se pokuša prijaviti s krivim mail-om

- Ulazni podaci: (krivi) Gmail računa korisnika
- Očekivani rezultat: Korisnik dobiva obavijest da je mail pogrešan i neka pokuša ponovno
- Dobiveni rezultat: Prolaz
- Postupak ispitivanja:
 1. Korisnik otvara login stranicu.
 2. Korisnik klikne na gumb "Continue with Google".
 3. Korisnik odabire "Upotrijebite neki drugi račun".
 4. Korisnik upisuje krivi mail.

5. Korisnik dobiva obavijest da je mail pogrešan i da pokuša ponovno.

The screenshot shows the Selenium IDE interface with the following details:

- Project:** Noodle Testing*
- Executing:** ✓ login-krivi-mail*
- URL:** https://noodle-frontend.onrender.com
- Test Case Script:** (Listed in the main pane)
 - ✓ open /login
 - ✓ set window size 1265x1372
 - ✓ click css=.google-login-button
 - ✓ click css=.B682ne > .VV3oRb
 - ✓ type id=identifierId askned12nk3ććasd
 - ✓ click css=.VfPpkd-LgbsSe-OWXExe-k8QpJ > .VfPpkd-vQzf8d
- Run Statistics:** Runs: 1 Failures: 0
- Log:** (Listed in the bottom pane)

Log Message	Timestamp
Running 'login-krivi-mail'	20:03:26
1. open on /login OK	20:03:26
2. setWindowSize on 1265x1372 OK	20:03:28
3. click on css=.google-login-button OK	20:03:30
4. click on css=.B682ne > .VV3oRb OK	20:03:32
5. type on id=identifierId with value askned12nk3ććasd OK	20:03:34
6. click on css=.VfPpkd-LgbsSe-OWXExe-k8QpJ > .VfPpkd-vQzf8d OK	20:03:37
'login-krivi-mail' completed successfully	20:03:37



Sign in

to continue to [Noodle](#)

Email or phone

askned12nk3ććasd

Enter a valid email or phone number

[Forgot email?](#)

[Create account](#)

[Next](#)

2. Komponenta: Forma za registraciju

- Ispitivana funkcionalnost: Korisnik se prijavljuje putem Google oAuth-a te ispunjava i pokušava poslati formu za registraciju

Ispitni slučaj 1: Korisnik se uspješno prijavljuje te ispunjava i šalje formu za registraciju, preusmjeren je na stranicu "Niste još prihvaćeni"

- Ulazni podaci: Google mail i lozinka korisnika, nova šifra za Noodle, korisnikov OIB, spol, adresa, datum rođenja, te ID škole
- Očekivani rezultat: Korisnik se uspješno prijavljuje, prikazuje mu se forma, korisnik ispunjava formu i uspješno ju šalje (dobiva obavijest o uspješnom slanju zahtjeva), preusmjeren je na stranicu "Niste još prihvaćeni"
- Dobiveni rezultat: Prolaz
- Postupak ispitivanja:

- Korisnik otvara login stranicu.
- Korisnik klikne na gumb "Continue with Google".
- Korisnik bira svoj mail i prijavljuje se.
- Korisnik upisuje šifru, OIB, spol, adresu, datum rođenja i ID škole.

5. Korisnik klikne na "Predaj zahtjev".

Selenium (Selenium IDE) - Selenium IDE - Noodle Testing — Mozilla Firefox

Project: Noodle Testing

Executing: ✓ registracija

https://noodle-frontend.onrender.com

Command	Target	Value
1 ✓ open	/login	
2 ✓ set window size	822x629	
3 ✓ click	css=google-login-button	
4 ✓ click	css=a2xC0f:nth-child(2).pGzURd	
5 ✓ click	css=Vfpkd-ksKsZd-mWPk3d-OWXExe-Tv0i5d-UHZMc > Vfpkd-RLmnJb	
6 ✓ click	name=password	
7 ✓ type	name=password	Lozinka123!
8 ✓ click	name=OIB	
9 ✓ type	name=OIB	86459129312
10 ✓ click	name=spol	
11 ✓ select	name=spol	label=Muško
12 ✓ click	css=option:nth-child(2)	
13 ✓ click	name=address	
14 ✓ type	name=address	Adresa 23G
15 ✓ click	name=dateOfBirth	
16 ✓ click	name=dateOfBirth	
17 ✓ type	name=dateOfBirth	2025-01-01
18 ✓ click	name=primarySchool	
19 ✓ type	name=primarySchool	1
20 ✓ click	css=inform	
21 ✓ click	css=Predaj_zahujev	

Command Target Value

Runs: 1 Failures: 0

Description

Log Reference

```

14. type on name=address with value Adresa 23G OK
15. click on name=dateOfBirth OK
16. click on name=dateOfBirth OK
17. type on name=dateOfBirth with value 2025-01-01 OK
18. click on name=primarySchool OK
19. type on name=primarySchool with value 1 OK
20. click on css=inform OK
21. click on css=Predaj_zahujev OK
    
```

20:36:49
20:36:50
20:36:51
20:36:52
20:36:53
20:36:54
20:36:55
20:36:56

noodle-frontend.onrender.com says

Zahtjev uspješno poslan!

OK

Unesite podatke:

Juraj

Ćižić

juricacizic@gmail.com

.....

86591823741

Muško



Adresa 23G

01/06/2025



1

Predaj zahtjev



Bok, Juraj Čižić!

Niste još prihvaćeni.

Ispitni slučaj 2: Korisnik se uspješno prijavljuje te ispunjava formu, ali upisuje krivi ID škole, zatim pokušava poslati zahtjev

- Ulazni podaci: Google mail i lozinka korisnika, nova šifra za Noodle, korisnikov OIB, spol, adresa, datum rođenja, te (krivi) ID škole
- Očekivani rezultat: Korisnik se uspješno prijavljuje, prikazuje mu se forma, korisnik pogrešno ispunjava formu i pokušava ju poslati (dobiva obavijest da zahtjev nije poslan)
- Dobiveni rezultat: Prolaz
- Postupak ispitivanja:
 1. Korisnik otvara login stranicu.
 2. Korisnik klikne na gumb "Continue with Google".
 3. Korisnik bira svoj mail i prijavljuje se.
 4. Korisnik upisuje šifru, OIB, spol, adresu, datum rođenja i (krivi) ID škole.

5. Korisnik klikne na "Predaj zahtjev" i dobiva obavijest da zahtjev nije poslan.

Project: Noodle Testing*

Tests + ✓ open

Search tests... ✓ set window size

https://noodle-frontend.onrender.com

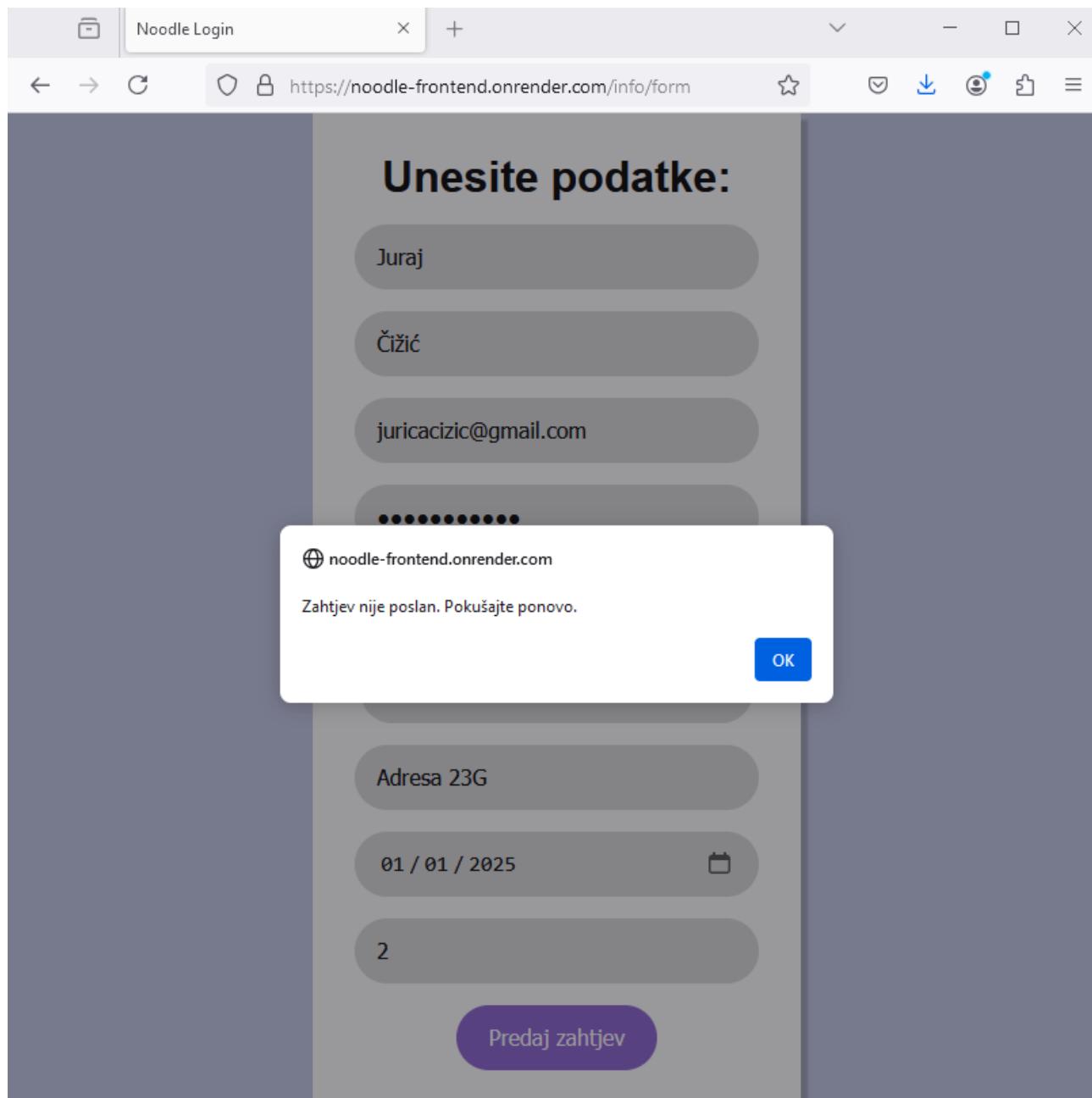
Command	Target	Value
1 ✓ open	#login	
2 ✓ set window size	822x829	
3 ✓ click	css=.google-login-button	
4 ✓ click	css=aZvCD:nth-child(2).pGzURd	
5 ✓ click	css=VfPpkd-ksKsZd-mWPk3d-OWXExe-Tv8i5d-LfZMc > .VfPpkd-RLmnJb	
6 ✓ click	name=password	Lozinka123!
7 ✓ type	name=pass	86459129312
8 ✓ click	name=OB	
9 ✓ type	name=OB	label=Muško
10 ✓ click	name=spol	
11 ✓ select	css=option:nth-child(2)	
12 ✓ click	name=address	Adresa 23G
13 ✓ click	name=dateOfBirth	2025-01-01
14 ✓ type	name=dateOfBirth	
15 ✓ click	name=primarySchool	2
16 ✓ click	css=infiform	
17 ✓ type	css=Predaj_zahtjev	
18 ✓ click		
19 ✓ click		
20 ✓ click		
21 ✓ click		

Command Target Value Description

Log Reference 🕒 20:44:33

- 15. click on name=dateOfBirth OK 20:44:34
- 16. click on name=dateOfBirth OK 20:44:35
- 17. type on name=dateOfBirth with value 2025-01-01 OK 20:44:36
- 18. click on name=primarySchool OK 20:44:37
- 19. type on name=primarySchool with value 2 OK 20:44:39
- 20. click on css=infiform OK 20:44:40
- 21. click on css=Predaj_zahtjev OK 20:44:40

'registracijaNeuspjesna' completed successfully



3. Komponenta: Upravljanje zahtjevima (uređivanje te prihvatanje ili odbijanje zahtjeva za registraciju)

*Ispitivana funkcionalnost: prihvatanje učenika ili profesora u sustav škole

Ispitni slučaj 1: Admin uređuje korisnikov zahtjev, postavlja ga kao učenika

- Ulazni podaci: Status korisnika (učenik), razred koji će učenik pohađati, smjer učenika (informatički ili matematički)
- Očekivani rezultat: Admin uspješno uređuje korisnikov zahtjev, sprema uređene promjene te uspješno prihvata zahtjev (dobiva obavijest o uspješnom prihvatanju)
- Dobiveni rezultat: Prolaz
- Postupak ispitivanja:

1. Admin otvara Noodle stranicu.

2. Admin klikne na Admin gumb, zatim klikne na gumb "Zahtjevi".
3. Admin bira zahtjev koji želi urediti te klikne na gumb "Uredi".
4. Admin uređuje zahtjev, postavlja status učenika, upisuje razred i odabire smjer učenika.
5. Admin klikne na gumb spremi te dobiva obavijest o uspješno spremljenim promjenama.
6. Admin klikne na prihvati te dobiva obavijest o uspješno prihvaćenom zahtjevu.

The screenshot shows the Selenium IDE interface with a recorded test case titled 'prihvacanjeUcenika'. The test consists of 17 commands, mostly 'click' actions, which interact with various UI elements like menus, dropdowns, and buttons. The 'Log' section at the bottom shows the execution details for each command, indicating they were all successful ('OK').

```

Log Reference
11: click on name=smjер OK
12: select on name=smjер with value label=Informatički OK
13: click on css=.redak:nth-child(11) option:nth-child(3) OK
14: click on css=.req-button:nth-child(1) OK
15: click on css=.req-button:nth-child(20) OK
16: click on css=.back-button OK
17: click on css=.back-button OK
'prihvacanjeUcenika' completed successfully
  
```

noodle-frontend.onrender.com

Uspjesno prihvacen zahtjev.

Don't allow noodle-frontend.onrender.com to prompt you again

OK

Ispitni slučaj 2: Admin uređuje korisnikov zahtjev, postavlja ga kao profesora

- Ulazni podaci: Status korisnika (profesor), broj mobitela, razrede kojima će profesor predavati te razred kojem će profesor biti razrednik (opcionalno)
- Očekivani rezultat: Admin uspješno uređuje korisnikov zahtjev, sprema uređene promjene te uspješno prihvaca zahtjev (dobiva obavijest o uspješnom prihvaćanju)
- Dobiveni rezultat: Prolaz
- Postupak ispitivanja:

1. Admin otvara Noodle stranicu.
2. Admin klikne na Admin gumb, zatim klikne na gumb "Zahtjevi".
3. Admin bira zahtjev koji želi urediti te klikne na gumb "Uredi".

4. Admin uređuje zahtjev, postavlja status profesora, upisuje broj mobitela, razrede i odabire hoće li profesor biti razrednik i ako da bira kojem razredu.
5. Admin klikne na gumb spremi te dobiva obavijest o uspješno spremljenim promjenama.
6. Admin klikne na prihvati te dobiva obavijest o uspješno prihvaćenom zahtjevu.

The screenshot shows the Selenium IDE interface with a recorded test case. The test case details the steps required to accept a professor application, including selecting the professor role, entering a mobile number, and choosing classes. The log at the bottom of the interface shows the successful execution of each step.

Command	Target	Value
✓ open	/auth/pocetna	
✓ set window size	1706x1214	
✓ click	css=.admin-gumb	
✓ click	css=.menu-container > .req-button:nth-child(19)	
✓ click	css=.req-button:nth-child(19)	
✓ click	name=role	
✓ select	label=Profesor	
✓ click	name=role	
✓ click	css=option:nth-child(4)	
✓ click	name=moBrodj	
✓ type	0987561234	
✓ click	name=razredProfesora	
✓ type	2A,3C	
✓ click	name=razredProfesora	
✓ type	3C	
✓ click	css=req-button:nth-child(1)	
✓ click	name=razrednik	
✓ type		
✓ click	css=req-button:nth-child(20)	
✓ click	css=req-button:nth-child(20)	

Log Reference

```

11. click on name=razrednik OK
12. type on name=razredProfesora with value 2A,3C OK
13. click on css=input:nth-child(1) OK
14. click on name=razrednik OK
15. type on name=razrednik with value 3C OK
16. click on css=req-button:nth-child(1) OK
17. click on css=req-button:nth-child(20) OK
'prihvacanjeProfesora' completed successfully

```

 noodle-frontend.onrender.com

Uspjesno prihvacen zahtjev.

Don't allow noodle-frontend.onrender.com to prompt you again

OK

4. Komponenta: Repozitorij

*Ispitivana funkcionalnost: Upload i download datoteka sa repozitorija

Ispitni slučaj 1: Učenik preuzima materijale sa repozitorija

- Ulazni podaci: Nema
- Očekivani rezultat: Učenik uspješno preuzima datoteku na svoje računalo
- Dobiveni rezultat: Prolaz
- Postupak ispitivanja:

1. Učenik otvara Noodle stranicu.
2. Učenik klikne na "Repozitorij".

3. Učenik odabire materijal s repozitorija i klikne "Download".

The screenshot shows the Selenium IDE interface with a recorded script for a download operation. The script details are as follows:

Command	Target	Value
1. open	/auth/pocetna	
2. set window size	1537x1023	
3. click	css=repository-gumb	
4. click	css=.download-btn	

The 'Log' section shows the execution results:

```

FINE: uploding is only supported in Chrome at this time
'objavaMaterijala' ended with 1 error(s)
Running 'preuzimanjeMaterijalaUcenik'
1. open on /auth/pocetna OK
2. setWindowSize on 1537x1023 OK
3. click on css=repository-gumb OK
4. click on css=.download-btn OK
'preuzimanjeMaterijalaUcenik' completed successfully

```

The timestamp for each log entry is listed on the right.

Below the IDE, a screenshot of a web page titled "Repozitorij" shows a file named "message.txt" with a "Download" button.

Ispitni slučaj 2: Profesor učitava materijale za razrede koje izabere na izborniku

- Ulazni podaci: Datoteka koju profesor želi prebaciti na repozitorij
 - Očekivani rezultat: Profesor uspješno učitava datoteku
 - Dobiveni rezultat: Pad (Radi ručno, ali Selenium dobiva grešku kada pokušava učitati datoteku)
 - Postupak ispitivanja:
1. Profesor otvara Noodle stranicu.
 2. Profesor klikne na "Repozitorij".
 3. Profesor sa drop-down izbornika bira razrede za koje želi učitati datoteke.
 4. Profesor klikne na "Browse" te odabire datoteku koju želi učitati.

5. Profesor klikne na "Upload".

Selenium IDE - Noodle Testing*

Project: Noodle Testing*

Executing X objavaMaterijala*

https://noodle-frontend.onrender.com

	Command	Target	Value
1	✓ open	/auth/pocetna	
2	✓ set window size	1706x1214	
3	✓ click	css=.repository-gumb	
4	✓ click	css=.upload-section > input	
5	X type	css=.upload-section > input	C:\Desktop\message.txt
6	click	css=.overSelect	
7	click	css=label:nth-child(1)	
8	click	css=.upload-button	

Command Target Value

Target Description

Runs: 1 Failures: 1

Log Reference

```

Running 'objavaMaterijala'                                         21:36:21
1. open on /auth/pocetna OK                                         21:36:22
2. setWindowSize on 1706x1214 OK                                     21:36:24
3. click on css=.repository-gumb OK                                 21:36:27
4. click on css=.upload-section > input OK                         21:36:29
5. type on css=.upload-section > input with value C:\Desktop\message.txt Failed:
  {"code":-32000,"message":"Not allowed"}                           21:36:32
'objavaMaterijala' ended with 1 error(s)                           21:36:35

```

The screenshot shows a file upload interface titled "Repozitorij". At the top left is a "Browse..." button followed by the file name "message.txt". To the right is a purple "Upload" button. Below these is a dropdown menu labeled "Odaberite razred:" with two options: "2A" (checked) and "3C". Underneath the dropdown is a section labeled "Datoteke" which lists "message.txt" with "Download" and "Delete" buttons.

5. Komponenta: Karta za upute kako doći do lokacije

*Ispitivana funkcionalnost: Učenik upisuje početnu i krajnju adresu i dobiva najbržu rutu do lokacije

Ispitni slučaj 1: Učenik upisuje adrese i dobiva upute kako doći do određene lokacije

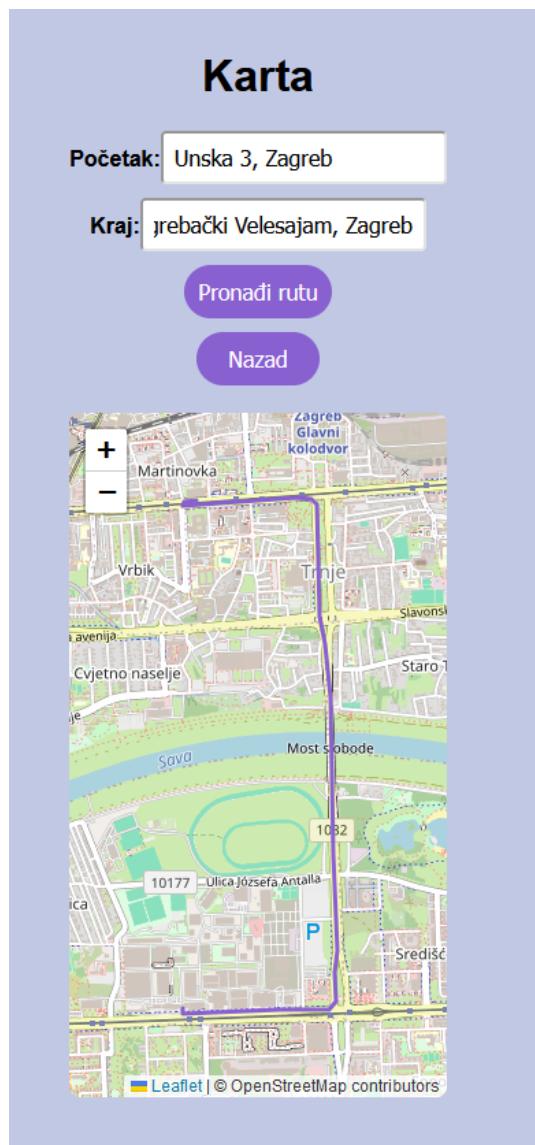
- Ulazni podaci: Početna adresa, krajnja adresa
- Očekivani rezultat: Učeniku se na karti prikazuje željena ruta
- Dobiveni rezultat: Prolaz
- Postupak ispitivanja:

1. Učenik otvara Noodle stranicu.
2. Učenik klikne na gumb "Karta".

3. Učenik upisuje početnu i krajnju adresu u odgovarajuća polja.

The screenshot shows the Selenium IDE interface with the following details:

- Project:** Noodle Testing*
- Test:** koristenjeMape-učenik*
- URL:** https://noodle-frontend.onrender.com
- Log:**
 - 3. click on css=map-gumb OK
 - 4. click on css=label:nth-child(1) > input OK
 - 5. type on css=label:nth-child(1) > input with value Unska 3, Zagreb OK
 - 6. click on css=label:nth-child(2) > input OK
 - 7. type on css=label:nth-child(2) > input with value Zagrebački Velesajam, Zagreb OK
 - 8. click on css=route-form OK
 - 9. click on css=button:nth-child(3) OK
- Summary:** "koristenjeMape-učenik" completed successfully



6. Komponenta: Izostanci

*Ispitivana funkcionalnost: Upisivanje izostanaka i mijenjanje statusa izostanka (opravdan, neopravdan, na čekanju)

Ispitni slučaj 1: Profesor upisuje izostanak učeniku jer nije prisutan, a drugi izostanak mijenja u "opravdan" jer je učenik donio Doktorsku ispričnicu

- Ulazni podaci: Datum, sat, status i opis izostanka
- Očekivani rezultat: Profesor uspješno upisuje izostanak učeniku i vidi upisani izostanak
- Dobiveni rezultat: Prolaz
- Postupak ispitivanja:
 1. Profesor otvara Noodle stranicu.
 2. Profesor klikne na "Izostanci".
 3. Profesor odabire razred.
 4. Profesor bira učenika sa liste.

5. Profesor upisuje datum, sat, status i opis izostanka te upisuje izostanak.

Extension: (Selenium IDE) - Selenium IDE - Noodle Testing* — Mozilla Firefox

Project: Noodle Testing*

Executing ✓ professorizostanakObjavljen https://noodle-frontend.onrender.com

Command	Target	Value
18 ✓ click	name=izostanakOpis	
19 ✓ type	name=izostanakOpis	Učenik nije prisutan.
20 ✓ click	css=.req-button	
21 ✓ click	name=izostanakSat	
22 ✓ type	name=izostanakSat	3.
23 ✓ click	name=izostanakDatum	
24 ✓ type	name=izostanakDatum	2025-01-01
25 ✓ click	name=izostanakOpis	
26 ✓ click	name=izostanakStatus	
27 ✓ select	name=izostanakStatus	label=Opravdan.
28 ✓ click	css=select:nth-child(3) > option:nth-child(2)	
29 ✓ click	name=izostanakOpis	
30 ✓ type	name=izostanakOpis	Doktorska ispričnica
31 ✓ click	css=.req-button	

Command Target Value

Target

Value

Description

Runs: 1 Failures: 0

Log Reference

```

25. click on name=izostanakOpis OK
26. click on name=izostanakStatus OK
27. select on name=izostanakStatus with value label=Opravdan. OK
28. click on css=select:nth-child(3) > option:nth-child(2) OK
29. click on name=izostanakOpis OK
30. type on name=izostanakOpis with value Doktorska ispričnica OK
31. click on css=.req-button OK
'profesorizostanakObjavljenLogout' completed successfully

```

21:49:20
21:49:21
21:49:22
21:49:23
21:49:24
21:49:25
21:49:26
21:49:27

2A

siniša Oib:
slinz 089612552 Odaberi

01 / 01 / 2025 Calendar icon

3.

Opravdan. dropdown arrow

Doktorska ispričnica

Unesi izostanak

Izostanci:

2025-01-01, 1. sat: Na čekanju
Opis: Učenik nije prisutan.

2025-01-01, 3. sat: Opravdan
Opis: Doktorska ispričnica

Ispitivanje sustava

Ispitni slučaj 1: Učenik se prijavljuje na stranicu, provjerava raspored i prognozu, upisuje i provjerava rutu do prakse nakon čega se odjavljuje

- Ulazni podaci: Gmail i lozinka učenika, početna adresa i krajnja adresa (adresa prakse)
- Očekivani rezultat: Učenik je uspješno prijavljen, raspored i prognoza se prikazuju, učenik uspješno upisuje adrese i prikazuje mu se ruta do prakse, učenik se uspješno odjavljuje
- Dobiveni rezultat: Prolaz
- Postupak ispitivanja:

1. Učenik otvara login stranicu.
2. Učenik odabire "Continue with Google" bira svoj mail i prijavljuje se.
3. Raspored i prognoza se prikazuju na početnoj stranici.
4. Učenik klikne na "Karta", otvara se stranica karte.
5. Učenik upisuje adrese i prikazuje mu se ruta do prakse.
6. Učenik se odjavljuje.

The screenshot shows the Selenium IDE interface with a test case titled "koristenjeMape-učenik". The test steps are as follows:

- 1. open (Target: /login)
- 2. set window size (Value: 1537x1023)
- 3. click (Target: css=.google-login-button)
- 4. click (Target: css=a2vCDf:nth-child(3).pGzURd)
- 5. click (Target: css=VfPpkd-ksKsZd-mWPI3d-OWXExe-Tv8l5d-IJZMc > .VfPpkd-vQzf8d)
- 6. click (Target: css=map-gumb)
- 7. click (Target: css=label:nth-child(1) > input)
- 8. type (Value: Unska ulica 3, Zagreb)
- 9. click (Target: css=label:nth-child(2) > input)
- 10. type (Value: Zagrebački Velesajam, Zagreb)
- 11. click (Target: css=button:nth-child(3))
- 12. click (Target: css=back-button)
- 13. click (Target: css=logout-gumb)

The log section shows the execution results for each step, indicating they were successful (OK). The final message is "'učenikLoginRutaLogout' completed successfully".

Ispitni slučaj 2: Profesor se prijavljuje, opravdava izostanke za koje je dobio ispričnice, šalje obavijest učenicima da provjere jesu li im svima izostanci opravdani

- Ulazni podaci: Gmail i lozinka profesora, razred, učenici kojima se uređuju izostanci, odgovarajući izostanci, status i opis izostanka
- Očekivani rezultat: Profesor se uspješno prijavljuje, odlazi na stranicu za izostanke, uspješno uređuje izostanke, odlazi na obavijesti i uspješno šalje obavijest odgovarajućem razredu
- Dobiveni rezultat: Prolaz
- Postupak ispitanja:

1. Profesor otvara login stranicu.
2. Profesor odabire "Continue with Google" bira svoj mail i prijavljuje se.
3. Profesor klikne na "Izostanci", odabire razred sa drop-down izbornika, mijenja status i opis izostanka koje želi promjeniti.
4. Profesor se vraća na početnu stranicu, klikne na "Obavijesti", upisuje naslov i tekst obavijesti te bira za koji razred ili razrede je obavijest namijenjena.
5. Profesor uspješno šalje obavijest.

6. Profesor se odjavljuje.

The screenshot shows the Selenium IDE interface with the following details:

- Project:** Noodle Testing
- Executing:** https://moodle-frontend.onrender.com
- Script:** sustavProfesorOpravdava
- Log:**
 - 29 type on name=nastlov with value Izostanci opravdani OK
 - 30 click on css=overSelect OK
 - 31 click on css=label:nth-child(1) OK
 - 32 click on name=tekst OK
 - 33 type on name=tekst with value Svi izostanci su opravdani, provjerite svoje izostanke. OK
 - 34 click on css=req-button OK
 - 35 click on css=logout-gumb OK
- Output:** 'sustavProfesorOpravdavanjeObavijest' completed successfully

Ispitni slučaj 3: Admin se prijavljuje, provjerava zahtjeve na čekanju, prihvata jednog učenika i smješta ga u odgovarajući razred, zatim prihvata profesora i dodjeljuje mu odgovarajuće razrede, označava da će biti razrednik i bira kojem razredu, nakon svega se odjavljuje

- Ulazni podaci: Gmail i lozinka admina, razred učenika, status učenika, status profesora, mobilni broj profesora, razredi profesora, razred kojem će profesor biti razrednik
- Očekivani rezultat: Admin se uspješno prijavljuje, provjerava zahtjeve, uspješno uređuje zahtjeve i prihvata korisnike (profesora i učenika), korisnici se uspješno spremaju u bazu, admin se uspješno odjavljuje
- Dobiveni rezultat: Prolaz
- Postupak ispitivanja:

1. Admin otvara login stranicu.
2. Admin odabire "Continue with Google" bira svoj mail i prijavljuje se.
3. Admin klikne na "Admin gumb", zatim klikne na "Zahtjevi".
4. Admin bira prvi zahtjev i klikne na "Uredi", postavlja status učenika i upisuje razred učenika.
5. Admin prihvata učenikov zahtjev.
6. Admin bira sljedeći zahtjev i klikne na "Uredi", postavlja status profesora, upisuje mobilni broj profesora, razrede kojima profesor predaje, označava da će profesor biti razrednik i upisuje kojem razredu.
7. Admin prihvata profesorov zahtjev.

8. Admin se odjavljuje.

The screenshot shows the Selenium IDE interface with a test log titled "Noodle Testing". The log details the steps taken by the admin to log out, including clicking on various buttons and input fields. The test was executed successfully with one run and zero failures. The log also includes a timestamped list of events.

Command	Target	Value
✓ select	name-role	label=Ucenik
✓ click	name-class	2A
✓ type	name-class	
✓ click	name-singer	
✓ select	name-singer	label=Matematički
✓ click	css=>req-button:nth-child(1)	
✓ click	css=>req-button:nth-child(1) > req-button nth-child(20)	
✓ click	css=>req-button:nth-child(19)	
✓ click	name-role	
✓ select	name-role	label=Profesor
✓ click	name-mot@rg	091838723
✓ type	name-mot@rg	
✓ click	name-razred@profesora	2B,2A,3C
✓ type	name-razred@profesora	
✓ click	css=input:nth-child(1)	
✓ click	name-razrednik	
✓ type	name-razrednik	
✓ click	css=>req-button:nth-child(1)	
✓ click	css=>req-button:nth-child(20)	
✓ click	css=>back-button	
✓ click	css=>back-button	
✓ click	css=>logout-gumb	
✓ click	css=>logout-gumb	

Runs: 1 Failures: 0

Log Reference

```

25: click on name-razrednik OK
26: type on name-razrednik with value 2A OK
27: click on css=>req-button nth-child(1) OK
28: click on css=>req-button nth-child(20) OK
29: click on css=>back-button OK
30: click on css=>back-button OK
31: click on css=>logout-gumb OK
'admin.logoutZahajivLogout' completed successfully

```

23:46:42
23:46:43
23:46:44
23:46:45
23:46:47
23:46:48
23:46:50
23:46:51

Ispitni slučaj 4: Učenik se prijavljuje, provjerava adresu prakse u obavijestima, odlazi na kartu, provjerava rutu do prakse, odlazi na početnu stranicu i u razredni chat potvrđuje svoj dolazak na praksu, odjavljuje se

- Ulazni podaci: Gmail i lozinka učenika, adresa od kud učenik kreće, adresa prakse, poruka za chat
- Očekivani rezultat: Učenik se uspješno prijavljuje, odlazi na obavijesti i čita adresu prakse, odlazi na obavijesti, upisuje adrese i dobiva rutu do prakse, vraća se na početnu stranicu i uspješno šalje poruku u razredni chat
- Dobiveni rezultat: Prolaz
- Postupak ispitivanja:

1. Učenik otvara login stranicu.
2. Učenik odabire "Continue with Google" bira svoj mail i prijavljuje se.
3. Učenik klikne na "Obavijesti" i pregledava obavijesti vezane za praksu (adresu prakse).
4. Učenik se vraća na početnu stranicu i klikne na "Karte".
5. Učenik upisuje svoju adresu i adresu prakse te dobiva rutu.
6. Učenik se vraća na početnu stranicu.
7. Učenik potvrđuje svoj dolazak na praksu u razredni chat.

8. Učenik se odjavljuje.

Selenium IDE - Noodle Testing*

Project: Noodle Testing*

Executing ▾ https://noodle-frontend.onrender.com ✓ učenikRasporedObavijest

Command	Target	Value
1. ✓ open	/login	
2. ✓ set window size	1265x1372	
3. ✓ click	css=.google-login-button	
4. ✓ click	css=aZvCD!nth-child(3) yAIK0b	
5. ✓ click	css=VIFpkd-ksKsZd-mWPk3d-OWXExe-Tv8l5d-UjZMc > .VIFpkd-vQzfbd	
6. ✓ click	css=.obavijest-gumb	
7. ✓ click	css=map-gumb	
8. ✓ click	css=label:nth-child(1) > input	
9. ✓ type	css=label:nth-child(1) > input	Talani 33A, Zagreb
10. ✓ click	css=label:nth-child(2) > input	
11. ✓ type	css=label:nth-child(2) > input	Zagrebački Vjesajam, Zagreb
12. ✓ click	css=button:nth-child(3)	
13. ✓ click	css=back-button	
14. ✓ click	css=input	
15. ✓ type	css=input	Potvrđujem dolazak na praksu sutra.
16. ✓ send keys	css=input	\$[KEY_ENTER]
17. ✓ click	css=logout-gumb	

Command Target Value

Target

Value

Description

Runs: 1 Failures: 0

Log Reference

```
11. type on css=label:nth-child(2) > input with value Zagrebački Vjesajam, Zagreb OK
12. click on css=button:nth-child(3) OK
13. click on css=back-button OK
14. click on css=input OK
15. type on css=input with value Potvrđujem dolazak na praksu sutra. OK
16. sendKeys on css=input with value $[KEY_ENTER] OK
17. click on css=logout-gumb OK
'učenikRasporedObavijestRuta/Chat' completed successfully
```

00:16:06
00:16:08
00:16:09
00:16:10
00:16:11
00:16:12
00:16:14
00:16:14

Bok, Juraj Čižić!

MAP REPOZITORIJ OBAVIJESTI LOGOUT

Raspored za 2A razred

Vrijeme	Pon	Uto	Sri	Čet	Pet
8:00	Sat razrednika [a4]	Kemija [a4]	Hrvatski jezik [b8]	Fizika [a4]	Kemija [a4]
8:50	Matematika I1 [a4]	Biologija [a4]	Informatika I [b8]	Hrvatski jezik [a4]	Geografija [a4]
9:40	Hrvatski jezik [a4]	Geografija [a4]	Engleski jezik [a4]	Povijest [a4]	Informatika I [b8]
10:40	Informatika I [a4]	Povijest [a4]	Glazbena umjetnost [a4]	Latinski jezik [b5]	Engleski jezik [b8]
11:30	Engleski jezik [a4]	TZK [DVORANA]	Likovna Kultura [a4]	Biologija [b8]	Matematika I1 [a4]
12:20	Fizika [a4]	Latinski jezik [b8]	Vjerouač [a4]	TZK [DVORANA]	Hrvatski jezik [a4]
13:10	Fizika [a4]	Matematika I1 [b5]	Matematika I1 [a4]	Matematika I1 [a4]	PRAZAN_SAT [/]

Run Python Script

Zagreb HR 6°C kiša jakog intenziteta

Juraj Čižić: Potvrđujem dolazak na praksu sutra.

Upišite poruku Pošalji

Korišteni alati i tehnologije

Programski jezici

Za izradu aplikacije korišten je JavaScript, glavni jezik za razvoj dinamičkih web aplikacija. Zahvaljujući svojoj prilagodljivosti, JavaScript je korišten za izradu interaktivnog korisničkog sučelja na klijentskoj strani, dok je na poslužiteljskoj strani omogućio učinkovitu implementaciju backend funkcionalnosti. Pored toga, korišten je Python za izradu posebnih funkcionalnosti poput generiranja rasporeda i komunikacije s bazom podataka putem Flask frameworka. SQL je primijenjen za upravljanje relacijskom bazom podataka PostgreSQL. Ova kombinacija programskih jezika omogućila je implementaciju interaktivne, sigurne i fleksibilne aplikacije.

Radni okviri i biblioteke

Frontend aplikacije temelji se na React.js (verzija 18.3.1), koji omogućuje modularnu strukturu i komponentni pristup razvoju. Axios je korišten za asinkronu komunikaciju s backendom, dok su prilagođeni stilovi definirani unutar CSS datoteka. Interaktivnost aplikacije dodatno je poboljšana real-time komunikacijom preko Socket.IO. React-Leaflet biblioteka korištena je za kartu temeljenu na OpenRouteService podatcima. Autentifikacija je ostvarena putem Google prijava koristeći OAuth 2.0. Backend aplikacije izgrađen je na Node.js (verzija 23.6.0) uz pomoć Express.js frameworka, dok se Python koristi kroz Flask za napredne funkcionalnosti poput generiranja rasporeda (raspored.py).

Baza podataka

Za potrebe aplikacije korištene su dvije baze podataka kako bi se osigurala optimalna funkcionalnost i prilagodljivost sustava. PostgreSQL (verzija 16.0) služi za: upis škole i njenih korisnika, pohranu podataka o školi (poput predmeta, prostorija i rasporeda), te spremanje podataka o školskom repozitoriju i obavijestima. Njegova relacijska struktura omogućuje stabilnu i brzu pohranu korisničkih podataka, rasporeda, i ostalih informacija. S druge strane, MongoDB se koristi za upravljanje dinamičnim podacima. Ova baza služi za spremanje sesija korisnika, osiguravajući kontinuiranu autentifikaciju i praćenje aktivnosti, te za implementaciju funkcionalnosti chata, gdje MongoDB omogućuje brzu i fleksibilnu pohranu poruka u realnom vremenu. Kombinacija PostgreSQL-a i MongoDB-a omogućila je iskorištanje prednosti obje baze, čime se osigurava pouzdanost, skalabilnost i fleksibilnost u radu aplikacije.

Razvojni alati

Visual Studio Code (verzija 1.95.3) bio je glavno razvojno okruženje zbog svoje prilagodljivosti i podrške za različite tehnologije poput JavaScripta, Reacta i Pythona. Proširenja dostupna u VS Codeu dodatno su ubrzala razvojni proces, omogućujući efikasno debugiranje i testiranje. Za verzioniranje koda korišten je Git (git version 2.47.0), koji je osigurao praćenje promjena, povijest razvoja i neometanu suradnju među članovima tima. Centralizirano spremanje koda putem Git repozitorija omogućilo je jednostavnu integraciju promjena i rješavanje sukoba tijekom timskog rada. Za pisanje dokumentacije koristili smo Wiki unutar Github repozitorija te smo ju oblikovali sukladno predlošku profesora. Figma je korištena za dizajniranje korisničkog sučelja, pružajući vizualnu prezentaciju aplikacije prije implementacije. Ovaj alat omogućio je kolaborativan rad na prototipovima i brz pregled dizajnerskih rješenja. Kombinacija ovih alata značajno je poboljšala učinkovitost i kvalitetu razvojnog procesa. Za upravljanje bazom podataka koristili smo PGAdmin4, alat namijenjen administraciji PostgreSQL-a. Svojim korisnički prilagođenim sučeljem omogućio

je lako upravljanje bazom, uključujući pregled tablica, izvršavanje SQL upita i testiranje funkcionalnosti baze tijekom razvoja aplikacije.

Alati za ispitivanje

Korišten je Selenium IDE 3.17.2. Razlog korištenja je jednostavan UI i jednostavno provođenje automatiziranih testova za sve slučajeve u sustavu. Problem se javio kod radnje upload-a datoteka na repozitorij u kojem slučaju Selenium IDE nije mogao izvesti tu radnju pa sejavljala greška, iako je ručno sustav radio očekivano. Prednosti su još brzo izvođenje testova i jednostavno kreiranje novih testova.

Cloud platforma

Aplikacija je deployana na platformi Render. Prednosti Render-a su jednostavnost uporabe, automatizacija većine koraka/procesa potrebnih za deploy stranice, automatsko skaliranje (Render automatski prilagođava resurse prema potrebama stranice). Platforma Render omogućava deploy više raznih web-servisa besplatno. Render također ima ugrađeni HTTPS certifikat za sve aplikacije, kojeg sam primjenjuje. Omogućeno je jednostavno spajanje sa Github repozitorijem na kojem se nalaze mape projekta.

Alati za komunikaciju

Komunikacija unutar tima se osim uživo odvijala preko Whatsapp grupe i Discorda. U Whatsapp grupi smo komunicirali vezano uz obavijesti, sastanke i bitne točke vezano uz projekt, a Discord smo koristili za zajedničko rješavanje problema i pomoći pri implementaciji aplikacije. Osim toga, koristili smo i Microsoft Teams za primanje obavijesti od asistenta i komunikaciju s demonstratoricom.

1. Instalacija

Preduvjeti: Node.js 22.11.0, React 19.0.0

Preuzimanje:

```
git clone https://github.com/jana-gazdek/Noodle.git
```

```
cd Noodle
```

Instalacija ovisnosti:

potrebno je nalaziti se u Noodle mapi

```
npm install
```

2. Postavke

Detaljne upute za konfiguraciju aplikacije:

Konfiguracijske datoteke:

ENV datoteke dostupne su održavateljima sustava

ENV datoteka mora sadržavati:

DB_HOST
DB_NAME
DB_PASSWORD
DB_PORT
DB_USER
EMAIL_PASS
EMAIL_USER
GOOGLE_CLIENT_ID
GOOGLE_CLIENT_SECRET
GOOGLE_DRIVE_CREDENTIALS
MONGO_URI
NODE_VERSION (samo za deployment na Render-u)
SESSION_SECRET
WEATHER_API_KEY

Postavke baze podataka:

Razvojno okruženje:

Napraviti bazu u pgAdminu pomoću podataka iz knexfile.js pod "connection:", pokrenuti 'node dbSetup.js' u cmd unutar mape /database, što pokreće i migracije i postavljanje seeda (inicijalnih podataka).

Produkcijsko okruženje: Na Renderu je baza podataka već postavljena.

3. Pokretanje aplikacije

Upute za pokretanje aplikacije u različitim okruženjima:

Razvojno okruženje: npm run dev

Produkcijsko okruženje: Render služi kao produkcijsko okruženje

Prevođenje aplikacije: npm run build

Pokretanje poslužitelja: npm start

Pokretanje servera: node server.js (u svakoj odgovarajućoj mapi za određeni mikroservis)

Provjera rada:

<http://localhost:3001/>

4. Upute za administratore

Smjernice za administratore aplikacije nakon puštanja u pogon:

Administrator može uređivati te prihvati ili odbijati zahtjeve. Također je omogućen unos prostorija i mijenjanje informacija o korisnicima registriranim u sustav.

Pristup administratorskom sučelju:

URL za admin sučelje: <https://noodle-frontend.onrender.com/info/admin-menu> Alternativno admini na stranici imaju specijalni Admin gumb

Redovito održavanje:

Arhiviranje baze podataka.

Pregled logova.

Ažuriranje aplikacije (primjer: povlačenje novih verzija iz Git repozitorija i ponovno pokretanje aplikacije).

git pull origin main

npm install

npm run build

npm start

Rješavanje problema:

Error logovima pristupa se na Render platformi. Prijavite se na Render platformu, odaberite odgovarajući mikroservis kod kojeg se javlja problem, lijevo pritisnite na "Logs".

5. Primjer za Render platformu (Cloud Deploy)

Render je popularna cloud platforma za jednostavno smještanje aplikacija.

PRIPREMA REPOZITORIJA:

Repozitorij mora u root-u sadržavati:

client folder sa svim potrebnim datotekama za static frontend
server folder sa svim potrebnim datotekama za web mikroservise
package.json sa svim dependency-ima package-lock.json

RENDER:

render.yaml:

services:

- type: web

name: Noodle

env: node

buildCommand: npm ci

startCommand: cd server/mikroservis && npm start (pogledajte nastavak za objašnjenje)

- type: static

name: client-static-site

env: static

buildCommand: npm run build

staticPublishPath: client/build

plan: free

Postavljanje na [Render](#):

Prijavite se na Render.

Kreirajte novi Web Service i povežite ga s vašim GitHub repozitorijem.

Konfigurirajte postavke:

Build Command: npm ci

Start Command: cd server/auth && npm start

*Napomena: Start Command se mijenja ovisno koji mikroservis pokrećete: cd server/mikroservis && npm start

Na mjesto mikroservis upišite mikroservis koji pokrećete. Potrebno je pokrenuti više Web Service-a za sve mikroservise.

Lista svih mikroservisa (upisujete umjesto mikroservis):

auth
objavaMaterijala
chat
map
raspored

Dodajte environment varijable (npr. DATABASE_URL, API_KEY).

Pokretanje aplikacije: Render će automatski preuzeti repozitorij, instalirati ovisnosti i pokrenuti aplikaciju. Nakon deploja, aplikaciji možete pristupiti putem generiranog URL-a (npr. <https://my-web-app.onrender.com/>).

Opis pristupa aplikaciji na javnom poslužitelju

Za pristup aplikaciji u web-pregledniku upišite URL: <https://noodle-frontend.onrender.com/>

Ako nakon prijave aplikacija ne radi očekivano (niste prebačeni na početnu stranicu), ugasite blokiranje kolačića u trenutnom pregledniku ili pokušajte pristupiti stranici putem drugog preglednika. Preporučeni preglednik je Chrome budući da na njemu nema potrebe mijenjati konfiguracije.

Za pristup admin sučelju potrebno je prijaviti se računom koji ima admin ovlasti. Zatim se na početnoj stranici prikazuje "Admin gumb". Pritisom na gumb odlazite na admin sučelje. Alternativno upišite URL: <https://noodle-frontend.onrender.com/info/admin-menu> (pristup će biti odbijen ukoliko niste prijavljeni valjanim admin računom).

Izrada aplikacije Noodle bio je kompleksan, ali izuzetno poučan proces koji je zahtijevao temeljito planiranje, tehničke vještine i timsku suradnju. Tijekom 15 tjedana, koliko je trajala izrada naše aplikacije, susreli smo se s brojnim izazovima i novim tehnologijama, ali smo isto tako stekli nova znanja i vještine.

Vrijeme izrade projekta

Ukupno vrijeme izrade projekta trajalo je 15 tjedana i uključivalo je različite faze razvoja. Prva faza bila je upoznavanje unutar tima, podjela poslova te razumijevanje zadatka. Druga faza uključivala je planiranje i oblikovanje funkcionalnosti za aplikaciju te dokumentiranje svih zahtjeva, obrazaca uporabe i arhitekture. Veliki naglasak prije prve revizije projekta bio je na izradi dobre dokumentacije i detaljnog opisivanja sustava. Sukladno tome, često smo se sastajali, ili uživo ili preko Discord poziva, i zajednički komentirali moguća rješenja. Uz privatne sastanke, sastajali smo se i s asistentom svaki tjedan i diskutirali ideje kako bismo mogli potvrditi idemo li u dobrom smjeru. Također smo se bavili implementacijom registracije korisnika i prikazom početne stranice. Nakon druge faze uslijedila je prva revizija projekta te smo se nakon toga bacili u treću fazu koja je uključivala implementaciju svih zahtjeva i izradu aplikacije. Ovdje se većina posla odvijala zasebno, odnosno svatko je radio svoj dio, a međusobno smo mogli pratiti napredak preko Githuba. Naravno, komunikacija je još uvijek bila ključna kako bi svaki član imao uvid u napredak aplikacije i uglavnom se odvijala preko Discord poziva. Do alfa verzije aplikacije, imali smo implementiranu većinu funkcionalnosti te je bilo potrebno provesti ispitivanja i popraviti pogreške, što je bio dio četvrte faze razvoja našeg projekta. Uz to, bilo je bitno i dokumentirati proces ispitivanja te dodati potrebne UML dijagrame. Prije druge revizije, bili smo u maloj stisci s vremenom, ali smo uspjeli implementirati sve funkcionalnosti i dokumentirati potrebne dijelove. Druga i treća faza bile su najzahtjevnije te su nam oduzele najviše vremena.

Tehnički izazovi

Susreli smo se s brojnim tehničkim izazovima u izradi aplikacije. Najveći izazov bio je osmisлитi algoritam za generiranje rasporeda sati jer je u obzir trebalo uzeti brojne faktore (smjerovi, usklađenost s kurikulumom, dostupnost nastavnika i prostorija, itd.). Također, prvi smo se puta susreli s tehnologijama i radnim okruženjima (npr. rad s Reactom, korištenje vanjskih sustava za autentifikaciju, implementacija određenih funkcionalnosti pomoću API-ja) koja prije nismo koristili pa je veliki dio posla uključivao i snalaženje u novim alatima. Korištenje platforme Github bio je veliki dio u procesu izrade aplikacije i većina se nije do sad susrela s radom na Githubu pa smo morali dobro naučiti kako funkcionira, što je s vremenom postajalo sve lakše. Izazovi su uključivali pravilno commitanje i fetchanje repozitorija, kako ne bismo stvarali konflikte te korištenje Wikija za pisanje dokumentacije. Pisanje dokumentacije bio je izazov sam po sebi koji smo savladali praćenjem predloška te kontinuiranim popravljanjem i istraživanjem pisanja pravilne dokumentacije. Osim same implementacije aplikacije, još jedan od izazova bilo je stavljanje aplikacije na javni poslužitelj (deployment). Na kraju smo još morali provesti ispitivanje aplikacije što nam nije oduzelo previše vremena.

Stečena znanja

Rad na projektu omogućio nam je stjecanje brojnih novih znanja. Svi tehnički izazovi, iako su nam otežavali rad, su nam također pomogli da se naučimo koristiti novim tehnologijama što je veoma bitno za napredak u akademskom, ali kasnije i poslovnom svijetu. Naučili smo koristiti se Reactom za izradu responzivnog korisničkog sučelja, pravilno spajanje frontenda i backenda aplikacije te kako se izrađuje baza podataka za aplikaciju. Također smo naučili kako postaviti izrađenu aplikaciju na javni poslužitelj (u našem slučaju

Render) i kako se ispituju komponente i sustav aplikacije. Imali smo priliku naučiti kako pravilno i ispravno pisati dokumentaciju za projekt te koliko je dobra dokumentacija bitna za razumijevanja aplikacije i posla koji radimo, što za tim koji sudjeluje u razvoju, što za vanjske suradnike i korisnike. Dokumentacija nam je pomogla u učenju izrade UML dijagrama koji su bitan dio procesa izrade aplikacije. I naravno posljednje, ali ne manje bitno, projekt nam je pokazao kako funkcioniра rad u timu i koliko je bitna dobra organizacija i podjela rada. Imali smo priliku prvi se puta susresti s radom u većoj grupi ljudi i naučiti važnost dobre komunikacije i kompromisa.

Potrebna znanja za napredak

Iako je većina izrade aplikacije išla po planu, uvijek ima mjesta za napredak. Pri završetku ovog procesa bilo je lako osvrnuti se na pogreške koje smo radili i načine na koje bismo mogli unaprijediti efektivnost tima za buduće projekte. Jedna od stvari koje se može popraviti je bolja organizacija vremena. Pri kraju projekta smo morali ubrzati tempo i radili smo gotovo cijele dane, što se moglo izbjegići da smo bolje organizirali vrijeme u prošlosti. Unatoč tome, uspjeli smo implementirati aplikaciju do kraja te smo uložili jako puno vremena u ovaj projekt. Podjelu poslova i zadatka smo dosta dobro podijelili i nismo previše odsakali od zadanih uloga. Svi smo međusobno puno surađivali, pogotovo u drugoj polovici izrade projekta kad je bilo potrebno dobro komunicirati sve što radimo kako ne bi došlo do konflikata u implementaciji aplikacije. Što se tiče grupne dinamike, nismo imali većih problema s organizacijom, a konflikata gotovo i nije bilo. Svatko je odradio dio posla koji mu je bio zadan, a sve probleme ili nedoumice rješavali smo zajednički i međusobno smo pomagali jedni drugima kad je to bilo potrebno. Uz rješavanja pogrešaka u kodu (tzv. debuggiranje), drugi najveći problem bio je dobro razumijevanje aplikacije i koje funkcionalnosti želimo implementirati te na koji način. U početku nam je trebalo neko vrijeme da uskladimo ideje, a dogovaranje detalja i specifičnosti trajalo je tijekom cijelog projekta. Ponekad smo zbog nedostatka komunikacije imali različite poglede na implementaciju aplikacije (npr. usklađenost dijagrama i stvarne implementacije), ali smo to popravili s češćim zajedničkim radom i dijeljenjem svega što smo napravili. U svakom slučaju, ako bismo nastavili nadograđivati aplikaciju Noodle ili započeli novi projekt, najbitnija je dobra organizacija poslova i vremena te ostvarenje uspješne komunikacije među članovima tima kako bi proces bio što efikasniji i lakši za praćenje jer se na taj način i svi tehnički problemi lakše rješavaju.

Perspektive za nastavak rada

Što se tiče perspektive za nastavak rada i mogućnosti rasta aplikacije postoji puno načina na koje bismo mogli unaprijediti trenutačne funkcionalnosti ili dodati nove. Aplikacija se može nadograditi uvođenjem funkcionalnosti poput integracije sustava za online nastavu poput Zooma, koji bi omogućio nastavnicima držanje nastave na daljinu. Algoritam za raspored mogao bi se nadograđivati da bude još precizniji i uzima u obzir još više komponenata. Isto tako bilo bi korisno uvesti mogućnost dodavanja zadataka, zadaća i ispita kako bi učenici mogli testirati svoje znanje. Također, moguće je prilagoditi aplikaciju da bude dostupna na više različitim jezika kako bismo omogućili internacionalno korištenje. Uz moguće nadogradnje, bitno je nastaviti održavanje postojećih funkcionalnosti kako bi aplikacija omogućila korisnicima što lakše i intuitivnije korištenje. Osim tehničkih aspekata, ako bi Noodle postao popularan među srednjim školama, mogli bismo razmišljati kako proširiti sustav da funkcioniра i za osnovne škole ili kako bismo mogli plasirati aplikaciju na tržište i napraviti ju dostupnom u više država. To bi zahtijevalo velike troškove i uvođenje novih članova u tim, poput dodatnih programera, dizajnera i marketinških stručnjaka.

Rev.	Opis promjene/dodataka	Autori	Datum
0.1	Napravljen predložak.	Jana Gazdek	27.10.2024
0.2	Dodana analiza zahtjeva i obrasci uporabe. Dodane reference.	Jana Gazdek	28.10.2024
0.3	Dodani sekvencijski dijagrami i dijagrami obrazaca uporabe.	Jana Gazdek	29.10.2024
0.4	Napravljena baza podataka, dodan opis tablica i dijagram baze podataka.	Damjan Popović, Roko Markov	5.11.2024
0.5	Dodani novi sevencijski dijagrami i dijagrami obrazaca uporabe.	Jana Gazdek	5.11.2024
0.6	Napravljen predložak za arhitekturu i dizajn sustava.	Jana Gazdek	11.11.2024
0.7	Dodani opis i obrazloženje arhitektura sustava. Dodana organizacija sustava i aplikacije, dodani dijagrami razreda.	Juraj Čižić, Jana Jovanović, Roko Markov, Damjan Popović, Filip Vuletić-Antić	12.11.2024
0.8	Popravljeni funkcionalni i ostali zahtjevi. Popravljeni sekvencijski dijagrami i dijagrami obrazaca uporabe.	Jana Gazdek	13.11.2024
0.9	Revizija cijele dokumentacije.	Jana Gazdek	14.11.2024
0.10	Dodani ostali dijagrami razreda.	Siniša Horvatić	15.11.2024
1.0	Dovršena verzija dokumentacije s dijelovima bitnim za 1. reviziju.	Jana Gazdek	15.11.2024
1.1	Popravljene pogreške iz 1. revizije.	Jana Gazdek	10.12.2024
1.2	Dodana nova poglavla i prepravljen ostatak dokumentacije prema novoj verziji predloška.	Jana Gazdek	9.1.2025
1.3	Raspisan predložak za ispitne slučajeve.	Jana Gazdek	14.1.2025
1.4	Dodan dijagram razmještaja	Jana Gazdek	17.1.2025
1.5	Dodan dijagram stanja i dijagram aktivnosti, započeto pisanje zaključka i korištenih tehnologija	Jana Gazdek	20.1.2025
1.6	Mala promjena baze podataka da odgovara implementaciji. Dodan dio zaključka	Jana Gazdek	22.1.2025
1.7	Popravljeni dijagrami stanja i dijagram aktivnosti. Dodani opisi dijagrama stanja, dijagrama aktivnosti i dijagrama razmještaja	Jana Gazdek	23.1.2025
1.8	Dodano ispitivanje i upute za puštanje u pogon	Juraj Čižić	24.1.2025
1.9	Dodan dijagram komponenti. Dodan zaključak	Jana Gazdek	24.1.2025

Rev.	Opis promjene/dodataka	Autori	Datum
1.10	Dodano korištene tehnologije i alati	Jana Jovanović, Jana Gazdek, Juraj Čižić	24.1.2025
1.11	Dodan opis dijagrama komponenti	Juraj Čižić	24.1.2025
1.12	Dodani ažurirani dijagrami razreda	Juraj Čižić, Siniša Horvatić	24.1.2025
1.13	Ažurirana tablica aktivnosti i dodani grafovi aktivnosti. Prepravljanje gramatičkih i pravopisnih pogrešaka. Dovršena verzija dokumentacije za 2. reviziju	Jana Gazdek	24.1.2025

Kontinuirano osvježavanje

Popisati sve reference i literaturu koja je pomogla pri pisanju projekta.

1. Programsко инженерство, FER ZEMRIS, <http://www.fer.hr/predmet/proinz>
2. The Unified Modeling Language, <https://www.uml-diagrams.org/>
3. Astah Community, <http://astah.net/editions/uml-new>
4. Tehnička škola Ruđer Bošković, <https://www.tsrb.hr/>
5. Ministarstvo znanosti, obrazovanja i mladih, <https://mzom.gov.hr/>

Dnevnik sastajanja

1. sastanak

- Datum: 14. listopada 2024.
- Prisustvovali: J.Gazdek, S.Horvatić, J.Jovanović, D.Popović, F.Vuletić-Antić
- Teme sastanka:
 - i. Upoznavanje s temom - diskutiranje ideja, odabir imena
 - ii. Raspored poslova - odluka tko radi na kojem dijelu aplikacije
 - iii. Dogovorena komunikacija - izrada Whatsapp grupe i Discord kanala

2. sastanak

- Datum: 20. listopada 2024.
- Prisustvovali: J.Gazdek, J.Jovanović
- Teme sastanka:
 - i. Izrada GitHub repozitorija - dogovor za pisanje dokumentacije i postavljanje GitHub repozitorija

3. sastanak

- Datum: 28. listopada 2024.
- Prisustvovali: J.Čižić, J.Gazdek, S.Horvatić, J.Jovanović, R.Markov, D.Popović, F.Vuletić-Antić
- Teme sastanka:
 - i. Razrada analize zahtjeva aplikacije - diskusija o funkcionalnim i ostalim zahtjevima, dogovor o načinu implementacije i alatima za izradu
 - ii. Dogovor za izradu Figma sučelja za prikaz izgleda aplikacije

4. sastanak

- Datum: 29. listopada 2024.
- Prisustvovali: J.Čižić, J.Gazdek, S.Horvatić, J.Jovanović, R.Markov, D.Popović, F.Vuletić-Antić
- Teme sastanka:
 - i. Detaljnija podjela poslova - dogovor za izradu baze podataka, unapredivanje UML dijagrama
 - ii. Dogovoren plan rada - razrada i rokovi za dodijeljene zadatke

5. sastanak:

- Datum: 6. studenog 2024.
- Prisustvovali: J.Čižić, J.Gazdek, S.Horvatić, J.Jovanović, R.Markov, D.Popović, F.Vuletić-Antić
- Teme sastanka:
 - i. Dogovor za frontend i backend - početak implementacije login stranice i naslovne stranice, implementacija logina

6. sastanak:

- Datum: 11. studenog 2024.
- Prisustvovali: J.Čižić, J.Gazdek, S.Horvatić, J.Jovanović, R.Markov, D.Popović, F.Vuletić-Antić
- Teme sastanka:
 - i. Dogovor za arhitekturu sustava - raspravljanje o mogućim rješenjima i podjela posla za pisanje dokumentacije
 - ii. Finalni dogovori - rasprava o detaljima koje treba popraviti prije 1. revizije

7. sastanak:

- Datum: 10. prosinca 2024.
- Prisustvovali: J.Čižić, J.Gazdek, S.Horvatić, J.Jovanović, D.Popović, F.Vuletić-Antić
- Teme sastanka:
 - i. Revizija posla koji smo do sad napravili - na što se trebamo više fokusirati, što smo mogli bolje
 - ii. Dogovor i plan rada za 2. ciklus - podjela poslova, istraživanje o algoritmu za generiranje rasporeda

8. sastanak:

- Datum: 9. siječnja 2025.
- Prisustvovali: J.Gazdek, S.Horvatić, J.Jovanović, R.Markov, D.Popović, F.Vuletić-Antić
- Teme sastanka:
 - i. Pregled svega što smo uspjeli implementirati - pregled funkcionalnosti aplikacije i odradenih zahtjeva
 - ii. Dogovor za usavršavanje alfa verzije aplikacije - ostale funkcionalnosti koje treba dodati u aplikaciju i popravak pogrešaka
 - iii. Dogovor za ispitivanje aplikacije - kako provesti i dokumentirati ispitivanje

9. sastanak:

- Datum: 23. siječnja 2025.
- Prisustvovali: J.Čižić, J.Gazdek, S.Horvatić, J.Jovanović, R.Markov, D.Popović, F.Vuletić-Antić
- Teme sastanka:
 - i. Zajedničko popravljanje pogrešaka i pisanje dokumentacije preko Discord poziva

10. sastanak:

- Datum: 24. siječnja 2025.
- Prisustvovali: J.Čižić, J.Gazdek, S.Horvatić, J.Jovanović, R.Markov, D.Popović, F.Vuletić-Antić
- Teme sastanka:
 - i. Završno uređivanje aplikacije preko Discord poziva

Plan rada

Zadatak	14.10-21.10	21.10-28.10	28.10-3.11	3.11-11.11	11.11-15.11	10.12-18.12	18.12-5.1	5.1-15.1	15.1
Planiranje	Razumjeti ideju projekta i postaviti osnovne ciljeve, podjela poslova	Razraditi zahtjeve i implementaciju sustava, pisati potrebnu dokumentaciju	Nadopunjavanje dokumentacije, detaljnija razrada zahtjeva, izrada baze podataka	Izrada Figma sućelja i razrada baze podataka	Finalni dogовори за izradu frontenda i backenda te deployment aplikacije	Plan rada za 2. ciklus	Planiranje detalja za funkcionalnost aplikacije		Zav. plan dovr. aplik.
Dokumentiranje	Napraviti predložak, napisati opis projektnog zadatka	Napisati analizu zahtjeva i specifikacije zahtjeva sustava	Usavršiti zahtjeve i UML dijagrame	Dodati arhitekturu i dizajn sustava	Razrada arhitekture i dovršavanje dokumentacije za 1. reviziju	Popravljene pogreške iz prve predaje	Preuređiti dokumentaciju da odgovara novom predlošku	Dokumentirajući sve potrebe 2. re	
Izrada baze podataka		Započeti izradu baze podataka	Nadopuna baze podataka po potrebi		Razrada baze podataka	Razrada i testiranje baze podataka	Razrada i testiranje baze podataka		
Frontend		Napraviti login i homepage	Deployment frontenda	Nastavak rada na frontendu	Dodati fronted za ostatak aplikacije	Ispitivanje i popravak frontenda	Ispitivanje i popravak frontenda		
Backend		Implementirati login putem sustava za autentifikaciju	Deployment backenda	Nastavak rada na backendu	Dodati backend za ostatak aplikacije	Ispitivanje i popravak backenda	Ispitivanje i popravak backenda		
Deployment		Omogućiti deployment aplikacije						Omo deplo aplik	

Tablica aktivnosti

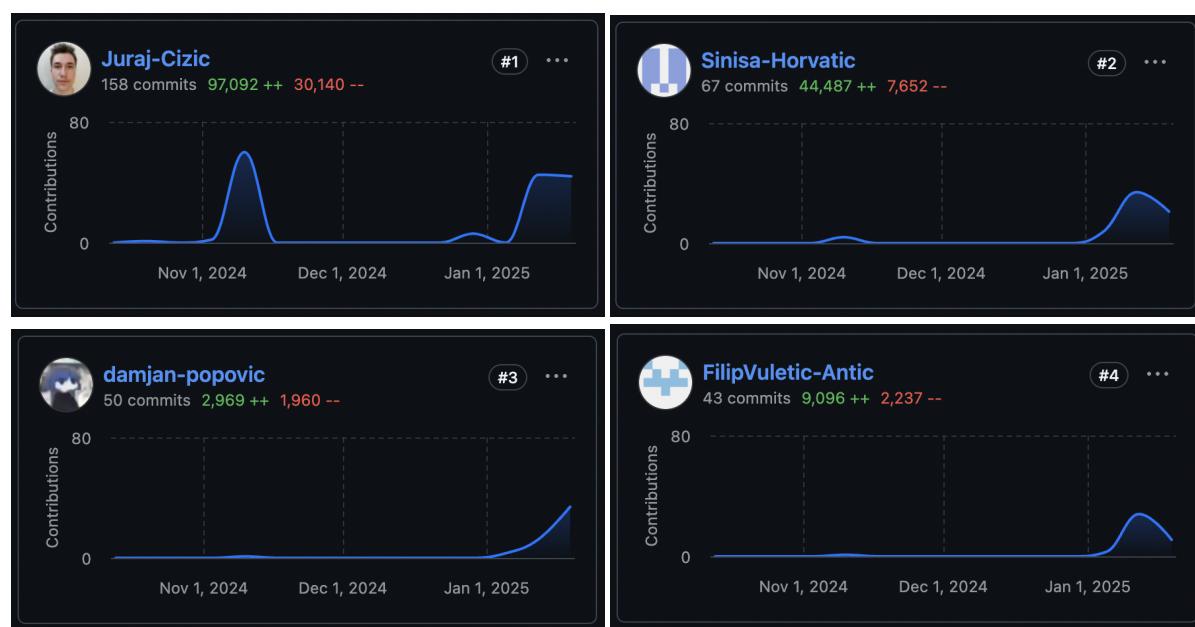
Kontinuirano osvježavanje

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

	Juraj Čižić	Jana Gazdek	Siniša Horvatić	Jana Jovanović	Roko Markov	Damjan Popović	Filip Vuletić-Antić
Upravljanje projektom	15		0.5				
Opis projektnog zadatka	4						
Funkcionalni zahtjevi	5						
Opis pojedinih obrazaca	8						
Dijagram obrazaca	6						
Sekvencijski dijagrami	5						
Opis ostalih zahtjeva	4						
Arhitektura i dizajn sustava	4	2	0.5	2	2.5	2.5	2
Baze podataka		1			5	5	
Dijagram razreda	4		1				
Dijagram aktivnosti		4					
Dijagram stanja		7					

	Juraj Čižić	Jana Gazdek	Siniša Horvatić	Jana Jovanović	Roko Markov	Damjan Popović	Filip Vuletić- Antić
Dijagram komponenti		8					
Korištene tehnologije i alati	1	2		3			
Ispitivanje programskog rješenja	3						
Dijagram razmještaja		3					
Upute za puštanje u pogon	1					1	
Dnevnik sastajanja		1.5					
Zaključak i budući rad		5					
Popis literature		0.5					
<i>Dodatne stavke kako smo podijelili izradu aplikacije</i>							
Izrada baze podataka				6	6		
Spajanje s bazom podataka		3		16	30	11	
Backend	78	2		15	25	51	
Algoritam za raspored				32	11		
Frontend		83	24			12	
Figma		1	8				
Izrada prezentacija		5					

Dijagram pregleda promjena





Ključni izazovi i rješenja

Zaključno

Aplikacija Noodle nije samo tehnički projekt, već i rezultat timskog rada, prilagodbe izazovima i korištenja naučenih vještina. Tijekom izrade smo stekli vrijedna iskustva koja će nam služiti u budućim projektima i koja su nam pomogla da proširimo naše znanje. Sama aplikacija prikazuje korak naprijed u digitalizaciji školskog sustava i stvaranju efikasnih rješenja koja olakšavaju obrazovanje za učenike i nastavnike.

Opis izazova

- Kašnjenje u razvoju:** Tijekom razvoja aplikacije Noodle, vremenska ograničenja predstavljala su izazov, posebno zbog paralelnih obaveza na fakultetu i rada timu. Neki moduli, poput osmišljivanja algoritma za raspored ili integracije s vanjskim autentifikatorom OAuth2.0, zahtijevali su više vremena nego što je planirano.
- Tehnički problemi:** Poteškoće s integracijom frontenda (React) i poslužitelja zbog problema u komunikaciji API-ja. Postavljanje baze podataka za praćenje statistike pregleda materijala, pri čemu su optimizacija i sigurnost podataka zahtijevale dodatnu pažnju.
- Razjašnjavanje zahtjeva:** Povremeno je dolazilo do nejasnoća u definiranju funkcionalnosti i uskladenosti s implementacijom.

Rješenja

- Efikasnija raspodjela zadataka:** Pred kraj projekta bilo je potrebno raditi malo više, a spavati malo manje, ali smo uspjeli implementirati sve funkcionalnosti i uspješno izradili aplikaciju.
- Istraživanje:** Sve tehničke probleme riješili smo s pomoći pune debuggiranja, istraživanja i medusobne komunikacije.
- Povećana komunikacija:** Uveli smo češće sastanke i više komunicirali o tome kako bismo htjeli da aplikacija implementira odredene aktivnosti kako bismo došli do rješenja s kojim se svi slažu.