Home

miazze edited this page on Oct 29, 2024 · 7 revisions

Programsko inženjerstvo ak.god 2024./2025.

Sveučilište u Zagrebu

Fakultet elektrotehnike i računarstva

LinguaLink

Tim: <TG17.2>

LinguaLink

Dario Jurgec / djurgec

Mihovil Mađerić / mihovil13

Zita Martinović / zita-m

Adi Stajku / asttajku

Borna Svjetličić / borna21

Adam Šinjori / aSinjori

Mia Zekić / miazze

Nastavnik: Vlado Sruk

Pages 13

Find a page…

Home
- Programsko inženjerstvo ak.god 2024./2025.
  - Sveučilište u Zagrebu
  - Fakultet elektrotehnike i računarstva
- LinguaLink
  - Tim: <TG17.2>
  - LinguaLink

1. Opis projektnog zadatka
2. Analiza zahtjeva
3. Specifikacija zahtjeva sustava
4. Arhitektura i dizajn sustava
5. Arhitektura komponenata i razmj…
6. Ispitivanje programskog rješenja
7. Tehnologije za implementaciju a…
8. Upute za puštanje u pogon
9. Zaključak i budući rad
A. Dnevnik promjena dokumentacije
B. Popis literature
C. Prikaz aktivnosti grupe

Clone this wiki locally

https://github.com/mihovil13/Ling

© 2025 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

1. Opis projektnog zadatka

Zita Martinović edited this page on Nov 14, 2024 · 3 revisions

Opis projektnog zadatka

U današnjem sve globaliziranijem svijetu, sposobnost govorenja više jezika vrijedna je vještina za osobni i profesionalni razvoj. Ljudi različitih pozadina žele naučiti nove jezike kako bi poboljšali svoje izglede za karijeru, iskustva s putovanja ili kulturno razumijevanje. Tradicionalne škole jezika mogu biti skupe, stroge u smislu rasporeda ili ograničene u ponudi jezika dok pronalaženje pravog učitelja jezika može biti značajan izazov. Cilj je stvoriti funkcionalnu platformu za spajanje učitelja stranih jezika s pojedincima koji žele naučiti novi jezik. Web aplikacija na jednom mjestu pruža učiteljima i učenicima alat za pretraživanje, komunikaciju te raspored lekcija.

Korist ovog projekta:

- Olakšano povezivanje i komunikacija između učitelja i učenika
- Konkretno za učenike:
  - biranje učitelja koji odgovaraju vlastitom stilu učenja, razini znanja i potrebama
  - učenje na bilo kojoj lokaciji, bitna je samo veza s internetom
- Konkretno za učitelje:
  - mogu prilagoditi raspored prema vlastitim obavezama, birajući sate koji im najbolje odgovaraju
  - povezivanje s učenicima na širem području, a ne samo lokalno
  - također kao i učenici mogu raditi na bilo kojoj lokaciji
  - učenici mogu ostaviti ocjenu učitelju, pa učitelj može na temelju toga poboljšavati pristup ili steći određenu popularnost

Slična rješenja:

- https://www.italki.com/
  - stranica koja omogućuje spajanje učenika i učitelja, vrlo slično našem zadatku

Skup korisnika koji bi mogao biti zainteresiran za ostvareno rješenje

- to su svi ljudi koji žele naučiti novi jezik na fleksibilan način i ljudi koji žele svoje znanje prenijeti drugima te uz to i zaraditi

Mogućnost prilagodbe rješenja i moguće nadogradnje

- nakon originalne verzije rješenja, aplikacija se može dodatno prilagoditi prema savjetima korisnika

Opseg projektnog zadatka

- autorizacija unutar aplikacije preko OAuth 2.0 protokola koristeći tokene
- kreiranje i uređivanje profila
- implementacija funkcionalnosti kalendara s terminima dostupnosti
- pretraga koristeći filtriranje po kategorijama
- slanje i (ne)odobravanje zahtjeva
- funkcionalnost čavrljanja
- mogućnost ocjenjivanja, ostavljanja komentara i prikaza broja lekcija na profilu učitelja

Pages 13
Find a page…
Home
1. Opis projektnog zadatka
Opis projektnog zadatka
Korist ovog projekta:
Slična rješenja:
Skup korisnika koji bi mogao biti zainteresiran za ostvareno rješenje
Mogućnost prilagodbe rješenja i moguće nadogradnje
Opseg projektnog zadatka
2. Analiza zahtjeva
3. Specifikacija zahtjeva sustava
4. Arhitektura i dizajn sustava
5. Arhitektura komponenata i razmj…
6. Ispitivanje programskog rješenja
7. Tehnologije za implementaciju a…
8. Upute za puštanje u pogon
9. Zaključak i budući rad
A. Dnevnik promjena dokumentacije
B. Popis literature
C. Prikaz aktivnosti grupe

Clone this wiki locally

https://github.com/mihovil13/Ling

© 2025 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

2. Analiza zahtjeva

Mia Zekić edited this page 4 hours ago · 17 revisions

Funkcionalni zahtjevi

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
F-001	Sustav omogućuje korisnicima kreiranje računa pomoću e-mail adrese.	Visok	Zahtjev dionika	Korisnik se može registrirati e-mailom.
F-002	Sustav omogućuje prijavu i odjavu korisnika u tj. iz aplikacije.	Visok	Zahtjev dionika	Korisnik se nakon registracije može prijaviti i odjaviti iz sustava neograničen broj puta.
F-003	Sustav omogućuje korisniku brisanje računa.	Nizak	Zahtjev dionika	Korisnik može zatražiti brisanje računa nakon čega se oni brišu iz baze podataka. Korisnik nakon toga može kreirati novi račun s korištenim email-om.
F-004	Sustav omogućuje korisniku odabir uloge.	Visok	Zahtjev dionika	Korisnik prilikom registracije odabire ulogu učenika ili učitelja.
F-005	Sustav omogućuje korisniku posjetitelju uvid u profile učitelja.	Visok	Zahtjev dionika	Neregistrirani korisnik može vidjeti rasporede, načine predavanja te podatke koje učitelji imaju na svojem profilu.
F-006	Sustav omogućuje korisnicima uređivanje svojih podataka.	Visok	Zahtjev dionika	Korisnik nakon registracije može uređivati svoje podatke.
F-007	Sustav omogućuje pretraživanje učitelja (s ili bez filtera).	Srednji	Zahtjev dionika	Korisnik može pretraživati učitelje po imenu ili po odabranim kriterijima.
F-008	Sustav omogućuje učenicima slanje zahtjeva za lekciju.	Visok	Zahtjev korisnika	Učenik može, ako mu se određeni učitelj sviđa, poslati zahtjev tom učitelju za lekciju iz nekog predmeta s odabranim terminom.
F-009	Sustav omogućuje učiteljima prihvaćanje zahtjeva za lekciju.	Visok	Zahtjev korisnika	Učitelj može, ako nema preklapanja s rasporedom, prihvatiti zahtjev za lekciju.
F-010	Sustav omogućuje uspostavu komunikacije, kada je lekcija prihvaćena, između učenika i učitelja.	Visok	Zahtjev korisnika	Učeniku i učitelju, koji su dogovorili lekciju, otvara se mogućnost razgovora putem chat-a.
F-011	Sustav omogućuje primanje i slanje poruka između učenika i učitelja.	Srednji	Zahtjev korisnika	Učenik i učitelj međusobno mogu slati i primati poruke putem chat-a te se dogovoriti za detalje.
F-012	Sustav omogućuje učitelju uređivanje kalendara i označavanje slobodnih termina.	Srednji	Zahtjev korisnika	Učitelj može označiti u kalendaru u kojim terminima je slobodan.
F-013	Sustav omogućuje učitelju odbijanje zahtjeva.	Srednji	Zahtjev korisnika	Učitelj može, ako mu ne odgovara, odbiti učenikov zahtjev za lekciju.
F-014	Sustav omogućuje učeniku ostavljanje recenzije.	Srednji	Zahtjev korisnika	Učenik može nakon odrađenog predavanja ostaviti recenziju, tj. ocjenu i komentar o učitelju.
F-015	Sustav omogućuje postavljanje video predavanja u 'arhivirane lekcije'.	Nizak	Zahtjev korisnika	Učitelj može postaviti video lekcije, kojem će i taj učenik moći pristupiti, na svoj profil.

Ostali zahtjevi

Zahtjevi za održavanje

ID zahtjeva	Opis	Prioritet
NF-1.1.	Sustav treba biti oblikovan tako da omogućuje jednostavno održavanje.	Visok
NF-1.1.1.	Sustav treba imati dovoljno dokumentacije.	Visok
NF-1.2.	Sustav ispunjava sve funkcionalne zahtjeve.	Visok
NF-1.3.	Sustav treba zaštititi podatke.	Visok
NF-1.3.1.	Osobni podaci trebaju biti dostupni samo korisnicima.	Srednji
NF-1.3.2.	Podaci moraju biti očuvani u slučaju pada sustava	Visok
NF-1.3.3.	Baza podataka mora ispunjavati 3NF (Treću normalnu formu - tablice u 2NF i ne postoji tranzitivna ovisnost neključnog atributa od ključa)	Srednji
NF-1.4.	Sučelje aplikacije mora biti jednostavno za snalaženje	Visok
NF-1.4.1.	Sučelje je vizualno ugodno.	Srednje
NF-1.4.2.	Sučelje mora biti prilagođeno za mobilne uređaje	Srednji

---

Dionici:

1. Učitelji
2. Učenici
3. Administratori
4. Razvojni tim
5. Vanjski suradnici

---

1. Korisnik posjetitelj može:

> - Posjetiti profile svih učitelja
> - Vidjeti njihove rasporede
> - Za sve ostale funkcionalnosti, korisnik se treba registrirati/prijaviti u sustav

2. Neregistrirani korisnik može:

> - Napraviti korisnički račun
> - Registracija: neregistrirani korisnik, unese osnovne podatke i može birati između uloge učitelja ili učenika
> - Adresa mora biti važeća (e-mail) -> autorizacija: OAuth 2.0 protokol koristeći tokene

3. Registrirani korisnik može:

> - Prilikom kreacije korisničkog računa, korisnik bira hoće li biti učenik ili učitelj
> - Ostale funkcionalnosti ovise o tom odabiru

4. Učitelji mogu:

> - Urediti profil tako da opišu: koje jezike podučavaju, iskustvo, kvalifikacije, stil podučavanja, sliku profila i satnicu
> - Postavljanje termina u kojima su slobodni za podučavanje -> za integraciju kalendara koristimo postojeća rješenja
> - Prihvaćanje ili odbijanje zahtjeva učenika - učitelj je obaviješten o svakom zahtjevu za lekcijom
> - Nakon uspješnog dogovora, sastanak je evidentiran u kalendaru i otvara se mogućnost chat-a između učitelja i učenika
> - U chatu se mogu dogovoriti za detalje o platformi u kojoj će održati lekciju, razmjeni brojeva te načinu plaćanja
> - Nakon održane lekcije, učitelj mora prenijeti lekciju kako bi joj se moglo pristupiti s obje strane
> - Pristupiti arhiviranim lekcijama

5. Učenici mogu:

> - Urediti profil tako da opišu: koje jezike žele učiti, svoju trenutnu razinu znanja [početna, srednja, napredna], preferirani stil podučavanja, mogu postaviti ciljeve učenja (npr. putovanje, poslovna komunikacija, priprema za ispit itd.)
> - Pretražiti učitelje po različitim preferencijama i filterima
> - Ocijeniti i recenzirati učitelje kod kojih su odslušali barem jednu lekciju
> - Pristupiti arhiviranim lekcijama koje su odslušali

6. Baza podataka:

> - Pohranjuje informacije o korisničkim profilima
> - Pohranjuje arhivirane lekcije
> - Pohranjuje recenzije

7. Poslužitelj servisa za čavrljanje:

> - Nakon uspješnog dogovora za termin, omogućuje komunikaciju između učitelja i učenika

8. Poslužitelj kalendara:

> - Omogućuje funkcionalnosti kalendara poput pregleda i izmjene termina

9. Administrator:

> - Može pregledavati, uređivati i brisati sve korisničke račune

Pages 13
Find a page...
Home
1. Opis projektnog zadatka
2. Analiza zahtjeva
Funkcionalni zahtjevi
Ostali zahtjevi
Zahtjevi za održavanje
Dionici:
3. Specifikacija zahtjeva sustava
4. Arhitektura i dizajn sustava
5. Arhitektura komponenata i razmj...
6. Ispitivanje programskog rješenja
7. Tehnologije za implementaciju a...
8. Upute za puštanje u pogon
9. Zaključak i budući rad
A. Dnevnik promjena dokumentacije
B. Popis literature
C. Prikaz aktivnosti grupe

Clone this wiki locally
https://github.com/mihovil13/Ling

3. Specifikacija zahtjeva sustava

Ivo Marković, Matija Cupić, Jan Šnajder, Josip Mihaljević

Obrasci uporabe

UC1 – Prijava profila učenja

- Glavni sudionik: Korisnik (polaznik)
- Cilj: Pokrenuti profilno orijentiranu kontrolu pristupa
- Sudionici: Baza podataka
- Preduvjet: Za polaznika postoji aktivna verzija profilnih podataka
- Glavni scenarij tijeka:
  1. Korisnik pokreće prijavu sheme
  2. Vraća se u bazu sveti pokazatelji sheme
  3. Pokreće se profilni upitnik kroz koji polaznik unosi traženi kontekst
- Glavni ishodi izvršenja:
  - Aktivna sesija s profilnim kontekstom

UC2 – Registracija

- Glavni sudionik: Neregistrirani korisnik
- Cilj: Registrirati korisnika kako bi mogao koristiti sustav
- Sudionici: Baza podataka
- Preduvjet: Korisnik nije prijavljen u sustav i želi prijavu
- Glavni scenarij tijeka:
  1. Korisnik pokreće zahtjev za registracijom
  2. Korisnik upisuje e-mail adresu te korisničko ime i zadanu lozinku
  3. Nakon uspješnog slanja, sustav ga preusmjerava na početnu stranicu
- Glavni ishodi izvršenja:
  - Novi korisnik dobiva ulogu „Korisnik" i može se prijaviti u sustav

UC3 – Prijava

- Glavni sudionik: Registrirani korisnik
- Cilj: Korisniku korisničkom aplikacija koja mu omogućuje samo mogućnosti korisnika
- Sudionici: Baza podataka
- Preduvjet: Nepostojeći korisnik
- Glavni scenarij tijeka:
  1. Korisnik ulazi u prijavu
  2. U prijavu ulazi ime (ili email) i lozinka
  3. Ako su podaci ispravni, korisnik se prijavljuje u sustav
  4. Korisnik se preusmjerava na početnu stranicu
- Glavni ishodi izvršenja:
  - Korisnik prijavljen u sustav

UC4 – Odjava

- Glavni sudionik: Registrirani korisnik
- Cilj: Odjaviti se iz sustava
- Sudionici: Baza podataka
- Preduvjet: Korisnik je prijavljen
- Glavni scenarij tijeka:
  1. Korisnik pokreće odjavu
  2. Korisnik se preusmjerava na početnu stranicu
- Glavni ishodi izvršenja:
  - Korisnik je odjavljen

UC5 – Uređivanje informacija profila

- Glavni sudionik: Uloga
- Cilj: Ažuriranje/dopuna osnovnih podataka polaznika
- Sudionici: Baza podataka
- Preduvjet: Prijavljen u sustav
- Glavni scenarij tijeka:
  1. Korisnik ulazi u „moj profil"
  2. Odabire opciju uređivanja profila
  3. Nakon što unesu potrebne, sprema ih
- Glavni ishodi izvršenja:
  - Ažurirani podaci u bazi

UC6 – Pretraživanje ustanova

- Glavni sudionik: Učenik
- Cilj: Pronaći odgovarajuće ustanove u okruženju
- Sudionici: Baza podataka
- Preduvjet: Korisnik je prijavljen
- Glavni scenarij tijeka:
  1. Korisnik ulazi u „moje ustanove"
  2. Unosi kriterije pretraživanja (npr. udaljenost, vrsta programa, ocjene)
  3. Sustav mu prikazuje popis ustanova koje zadovoljavaju kriterije
- Glavni ishodi izvršenja:
  - Prikaz liste ustanova s osnovnim informacijama

UC7 – Pregled kalendara

- Glavni sudionik: Korisnik (polaznik)
- Cilj: Pregledati u kalendaru dostupne termine
- Sudionici: Baza podataka
- Preduvjet: Baza podataka
- Glavni scenarij tijeka:
  1. Korisnik ulazi u profilnu prikaz „kalendar"
  2. Otvara se pozivi koji pokazuje kalendar zahtjeva
- Glavni ishodi izvršenja:
  - Prikaz kalendara

UC8 – Slanje zahtjeva za lekciju

- Glavni sudionik: Učenik
- Cilj: Poslati zahtjev za lekcijom nekoj ustanovi
- Sudionici: Baza podataka
- Preduvjet: Korisnik je prijavljen kao učenik, a učenik je ministarski u ustanovama termina
- Glavni scenarij tijeka:
  1. Učenik otvara termin lekcije
  2. Učenik pošalja zahtjev „Pošalji zahtjev"
- Glavni ishodi izvršenja:
  - Novi zahtjev dodan u ustanovarski termin
  - Sustav obavještava učenika o nedostupnosti profilnosti

UC9 – Prihvaćanje zahtjeva

- Glavni sudionik: Učitelj
- Cilj: Prihvatiti zahtjev za lekcijom koji je poslao učenik
- Sudionici: Baza podataka
- Preduvjet: Korisnik je prijavljen kao učitelj te ima pristup profilnoj zahtjevu
- Glavni scenarij tijeka:
  1. Učitelj ulazi na profilnu prikazuje zahtjev za lekcijom
  2. Učitelj profilnu gumb „Prihvati zahtjev"
  3. Sustav u bazi ustanovarski termin
- Glavni ishodi izvršenja:
  - Novi termin dodan u ustanovarski termin

UC10 – Upravljanje komunikacijom

- Glavni sudionik: Pravna/fizička osoba
- Cilj: Omogućiti korisnicima izravan kontakt s ustanovama putem sustava
- Sudionici: Baza podataka
- Preduvjet: Korisnik je prijavljen u sustav
- Glavni scenarij tijeka:
  1. Korisnik ulazi u „moje poruke" i bira ustanovu
  2. Upisuje poruku i pritiska „Pošalji"
  3. Poruka se pohranjuje u bazu podataka
- Glavni ishodi izvršenja:
  - Nova poruka spremljena u bazu

UC11 – Slanje i primanje poruka

- Glavni sudionik: Registrirani korisnik
- Cilj: Izgrađivanje dvo-smjerne poruka kroz sustav
- Sudionici: Pravna/fizička osoba
- Preduvjet: Korisnik je kao korisnik prijavljen u sustav
- Glavni scenarij tijeka:
  1. Korisnik ulazi u „moje poruke"
  2. Pritiska „Pošalji" i upisuje poruku (npr. „Pošalji")
  3. Pritiska „Pošalji" i upisuje poruku (npr. „Pošalji")
- Glavni ishodi izvršenja:
  - Nova poruka dodana u bazu

UC12 – Uređivanje vlastitog kalendara

- Glavni sudionik: Učitelj
- Cilj: Označavanje dostupnih termina u svojem kalendaru
- Sudionici: Baza podataka
- Preduvjet: Korisnik je prijavljen kao učitelj
- Glavni scenarij tijeka:
  1. Učitelj ulazi u profilnu „kalendar"
  2. Učitelj odabira termin
  3. Učitelj klikne gumb „Dodaj termin"
- Glavni ishodi izvršenja:
  - Novi termin dodan u kalendar

UC13 – Odbijanje zahtjeva

- Glavni sudionik: Učitelj
- Cilj: Odbiti zahtjev za lekciju koji je poslao učenik
- Sudionici: Baza podataka
- Preduvjet: Korisnik je prijavljen kao učitelj te ima pristup profilnom zahtjevu
- Glavni scenarij tijeka:
  1. Učitelj ulazi na profilnu prikazuje zahtjev za lekcijom
  2. Učitelj pritiska gumb „Odbij zahtjev"
  3. Učitelj unosi razlog (npr. „Nemam vremena")
- Glavni ishodi izvršenja:
  - Zahtjev je označen kao odbijen

UC14 – Dodavanje recenzije

- Glavni sudionik: Učenik
- Cilj: Ostaviti komentar ili ocjenu na profilu učitelja
- Sudionici: Baza podataka
- Preduvjet: Korisnik je prijavljen kao učenik, učitelj je bio/bio je barem jednu lekciju kod učitelja
- Glavni scenarij tijeka:
  1. Učenik ulazi na profilnu profil
  2. Učenik bira ocjenu (npr. „Izvrsni učitelj")
  3. Učenik bira ocjenu (npr. „Izvrsni učitelj")
- Glavni ishodi izvršenja:
  - Nova ocjena dodana u bazu

UC15 – Objava ostvarene lekcije

- Glavni sudionik: Učitelj
- Cilj: Objaviti lekciju nakon što je lekcija stvarno i bila izvršena
- Sudionici: Baza podataka
- Preduvjet: Korisnik je prijavljen kao učitelj i učitelj je lekciju
- Glavni scenarij tijeka:
  1. Nakon uspješno lekcije, učitelj ulazi u dnevnik te ima objavi lekciju
  2. Učitelj bira opciju „Objavi lekciju"
- Glavni ishodi izvršenja:
  - Lekcija je označena kao izvršena

UC16 – Uvid u arhiviranu lekciju

- Glavni sudionik: Registrirani korisnik
- Cilj: Pregled i izvoza prethodno arhiviranih lekcija
- Sudionici: Baza podataka
- Preduvjet: Korisnik je prijavljen, sudjelovao je u prošlosti barem kao učenik ili učitelj te je učitelj objavio lekciju
- Glavni scenarij tijeka:
  1. Nakon uspješno lekcije, korisnik ulazi u „moje lekcije" i bira opciju „Arhivirane lekcije"
  2. Odabiranje na stavki prvi put otvaraju „arhivirane lekcije", nuda se prikazuje sum prošlijeđe lekcijama
- Glavni ishodi izvršenja:
  - Prikaz arhiviranih lekcija

UC17 – Uređivanje profila korisnika

- Glavni sudionik: Administrator
- Cilj: Ažuriranje podataka o korisniku nekih postoje profil
- Sudionici: Baza podataka
- Preduvjet: Administrator je prijavljen
- Glavni scenarij tijeka:
  1. Administrator ulazi u profilnu korisnika
- Glavni ishodi izvršenja:
  - Ažurirani podaci

UC18 – Brisanje profila

- Glavni sudionik: Administrator
- Cilj: Trajno uklanjanje profila korisnika s baze podataka
- Sudionici: Baza podataka
- Preduvjet: Administrator je prijavljen
- Glavni scenarij tijeka:
  1. Administrator pristupa popisu svih korisničkih racuna
  2. Administrator bira korisnika za brisanje
  3. Sustav zatraži potvrdu na kraju u bazu podataka
- Glavni ishodi izvršenja:
  - Korisnik je uklonjen iz baze podataka, ima se mogućnost obnavljanja, ima se mogućnost obnavljanja

Dijagrami obrazaca uporabe

UC1 dijagram za opis ruteta

UC dijagram funkcionalnosti učitelja

UC dijagram funkcionalnosti učenika

Sekvencijalni dijagrami

Obrazac uporabe UC6 – Pretraživanje učitelja

Korisnik otvara „moje ustanove" i ulazi u sve pretraživanje nakon Ritmo ta termina koji pokreće upit prema bazi. Nakon prikaza ustanova, korisnik bira jednu ustanovu i ulazi u detaljnu stranicu ustanove. Nakon toga, korisnik ulazi u „moje lekcije" i bira opciju „Arhivirane lekcije". Nakon prikaza arhiviranih lekcija, bira jednu lekciju i ulazi u detaljnu stranicu lekcije. Nakon toga, korisnik ulazi u „moje ocjene" i bira opciju „Dodaj ocjenu". Korisnik zatim mora pohraniti na ime učitelja prethodi tekstom ocjene profilu. Ako u bazi podataka ne postoje učitelji koji odgovaraju kriterijima pretraživanja, baza podataka to vraća aplikaciji i prikazuje se poruka o tome da nema rezultata. Ako postoje odgovarajuće ustanove, baza podataka ih vraća aplikaciji i prikazuje se popis.

Obrazac uporabe UC8 – Slanje zahtjeva za lekciju

Korisnik otvara „moje ustanove" i ulazi u detaljnu stranicu ustanove. Nakon toga, korisnik ulazi u „moje lekcije" i bira opciju „Arhivirane lekcije". Nakon prikaza arhiviranih lekcija, bira jednu lekciju i ulazi u detaljnu stranicu lekcije. Nakon toga, korisnik ulazi u „moje ocjene" i bira opciju „Dodaj ocjenu". Korisnik zatim mora pohraniti na ime učitelja prethodi tekstom ocjene profilu. Ako u bazi podataka ne postoje učitelji koji odgovaraju kriterijima pretraživanja, baza podataka to vraća aplikaciji i prikazuje se poruka o tome da nema rezultata. Ako postoje odgovarajuće ustanove, baza podataka ih vraća aplikaciji i prikazuje se popis.

Obrazac uporabe UC9 – Prihvaćanje zahtjeva

Korisnik otvara „moje lekcije" i ulazi u detaljnu stranicu zahtjeva i prihvaćam zahtjeva na lekciju od nekog učenika. Nakon prihvaćanja ga pošalje "Dalje", nakon aplikacija u bazi podataka ažurira status odgovarajućeg zahtjeva na "Prihvaćen". Nakon toga, nakon aplikacija obavještava učenika o prihvaćenoj zahtjevu na lekciju.

Obrazac uporabe UC13 – Odbijanje zahtjeva

Korisnik otvara „moje lekcije" i ulazi u detaljnu stranicu zahtjeva i prihvaćam zahtjeva na lekciju od nekog učenika. Nakon toga, korisnik "Odbij" zahtjeva i ispunjava razlog zbog kojeg odbija zahtjev. Nakon toga, nakon aplikacija u bazi podataka ažurira status odgovarajućeg zahtjeva na "Odbijen". Nakon toga, nakon aplikacija obavještava učenika o odbijanju zahtjeva na lekciju.

Obrazac uporabe UC16 – Uvid u arhiviranu lekciju

Korisnik otvara „moje lekcije" i ulazi u detaljnu stranicu zahtjeva i prihvaćam zahtjeva na lekciju od nekog učenika. Nakon toga, korisnik "Odbij" zahtjeva i ispunjava razlog zbog kojeg odbija zahtjev. Nakon toga, nakon aplikacija u bazi podataka ažurira status odgovarajućeg zahtjeva na "Odbijen". Nakon toga, nakon aplikacija obavještava učenika o odbijanju zahtjeva na lekciju.

Provjera usklađenosti ključnih funkcionalnosti u obrazce uporabe

Obrazac uporabe	Ključna funkcionalnost
UC1 – Prikaz profilu učitelja	"Uvid u bazu koli dostupnih učitelja i njihovih rasporeda omogućuje pr ... " F-006
UC2 – Registracija	"Omogućuje za korištenje bez toga drugi funkcionalnosti aplikacije je registracija" F-001
UC3 – Prijava	"omogućuje za korištenje bez toga drugi funkcionalnosti aplikacije je prijava korisnika" F-002
UC4 – Odjava	F-002
UC5 – Uređivanje informacija profila	"Uređivanje/dopuna osnovnih podataka polaznika" F-004
UC6 – Pretraživanje učitelja	"Uvid u bazu koli dostupnih učitelja i njihovih rasporeda omogućuje pr ... " F-006
UC7 – Pregled kalendara	"Uvid u bazu koli dostupnih učitelja i njihovih rasporeda omogućuje pr ... " F-006
UC8 – Slanje zahtjeva za lekciju	"Uvid u bazu koli dostupnih učitelja i njihovih rasporeda omogućuje pr ... " F-006
UC9 – Prihvaćanje zahtjeva	"Uvid u ze svaki zahtjev koji mogu/ima prihvaćaju ili odbijaju" F-009
UC10 – Upravljanje komunikacijom	"Uvid u ze svaki zahtjev koji mogu/ima prihvaćaju ili odbijaju" F-009
UC11 – Slanje i primanje poruka	"Uvid u ze svaki zahtjev koji mogu/ima prihvaćaju ili odbijaju" F-009
UC12 – Uređivanje vlastitog kalendara	"Uvid u ze svaki zahtjev koji mogu/ima prihvaćaju ili odbijaju" F-009
UC13 – Odbijanje zahtjeva	"Uvid u ze svaki zahtjev koji mogu/ima prihvaćaju ili odbijaju" F-009
UC14 – Dodavanje recenzije	"Uvid u ze svaki zahtjev koji mogu/ima prihvaćaju ili odbijaju" F-009
UC15 – Objava ostvarene lekcije	"Uvid u ze svaki zahtjev koji mogu/ima prihvaćaju ili odbijaju" F-009
UC16 – Uvid u arhiviranu lekciju	"Uvid u ze svaki zahtjev koji mogu/ima prihvaćaju ili odbijaju" F-009
UC17 – Uređivanje profila korisnika	"Uvid u ze svaki zahtjev koji mogu/ima prihvaćaju ili odbijaju" F-009
UC18 – Brisanje profila	"Uvid u ze svaki zahtjev koji mogu/ima prihvaćaju ili odbijaju" F-009

4. Architektura i dizajn budowy

Last modified: 2 Kwietnia 2026, 11:11 AM

Architektura tworzenia

Opis architektury

Na poziomie C4 model, system Kolejka składa się z czterech kontenerów aplikacji. Klienci to aplikacja mobilna konsumenta (Flutter), panel administracyjny dla personelu (Flutter Web) oraz wyświetlacz publiczny (Flutter Web). Wszystkie komunikują się z jednym serwerem backendowym (Spring Boot) przez REST API. Backend zapisuje dane do bazy PostgreSQL i publikuje zdarzenia domenowe do kolejki RabbitMQ. Kolejka obsługuje 3 kolejki (queues) i 1 wymianę (exchange) typu topic, z trzema konsumentami:

- Powiadomienia — wysyła SMS przez Twilio
- Statystyki — zapisuje dane do tabeli agregacji w PostgreSQL
- Wyświetlacz — aktualizuje stan kolejki na wyświetlaczu publicznym

Kontenery:

- Backend (Spring Boot) — jedyny serwer REST API. Obsługuje logikę biznesową, zapisuje do bazy PostgreSQL, publikuje zdarzenia do RabbitMQ. Zawiera 3 moduły: kolejka, powiadomienia, statystyki.
- Aplikacja mobilna (Flutter) — interfejs konsumenta: dołączanie do kolejki, śledzenie pozycji, anulowanie, powiadomienia push.
- Panel administracyjny (Flutter Web) — interfejs dla personelu: wywoływanie następnej osoby, zarządzanie kolejką, statystyki.
- Wyświetlacz (Flutter Web) — publiczny ekran z numerami aktualnie obsługiwanych klientów.

Kontenery zewnętrzne:

- PostgreSQL — baza danych
- RabbitMQ — broker wiadomości. Kolejka: queue.kolejka. Wymiana: kolejka.exchange (typ topic). Klucze routingu: kolejka.dodano, kolejka.wezwano, kolejka.anulowano, kolejka.zmieniono.
- Twilio — bramka SMS. Powiadomienia SMS o wezwaniu.

Komponenty wewnątrz backendu:

- Kontrolery — REST API, walidacja DTO, mapowanie na encje
- Serwisy — logika biznesowa, zarządzanie transakcjami
- Repozytoria — dostęp do bazy PostgreSQL
- Publikacja zdarzeń — publikacja zdarzeń domenowych do RabbitMQ
- Konsumenci — nasłuchiwanie kolejek i przetwarzanie zdarzeń
- Harmonogramy — zadania cykliczne (np. czyszczenie starych danych)

Decyzje:

- Monolit zamiast mikroserwisów — prostota wdrożenia i utrzymania dla małego zespołu
- Jedna wymiana typu topic — elastyczne routing, łatwe dodawanie nowych konsumentów bez zmian w producencie
- Trzy kolejki — izolacja awarii, niezależne ponowienia dla każdego konsumenta
- Flutter dla wszystkich aplikacji — jeden framework, jeden zespół, współdzielenie kodu między platformami

Diagramy zależności między komponentami

Decyzje techniczne

- Monolit zamiast mikroserwisów — prostota wdrożenia i utrzymania dla małego zespołu
- Jedna wymiana typu topic — elastyczne routing, łatwe dodawanie nowych konsumentów bez zmian w producencie
- Trzy kolejki — izolacja awarii, niezależne ponowienia dla każdego konsumenta
- Flutter dla wszystkich aplikacji — jeden framework, jeden zespół, współdzielenie kodu między platformami

Obszarowy układ architektoniczny

- Kontrolery — REST API, walidacja DTO, mapowanie na encje
- Serwisy — logika biznesowa, zarządzanie transakcjami
- Repozytoria — dostęp do bazy PostgreSQL
- Publikacja zdarzeń — publikacja zdarzeń domenowych do RabbitMQ
- Konsumenci — nasłuchiwanie kolejek i przetwarzanie zdarzeń
- Harmonogramy — zadania cykliczne (np. czyszczenie starych danych)

Organizacja kontenerów na wdrożenia

- Backend — jedyny serwer REST API. Obsługuje logikę biznesową, zapisuje do bazy PostgreSQL, publikuje zdarzenia do RabbitMQ. Zawiera 3 moduły: kolejka, powiadomienia, statystyki.
- Aplikacja mobilna (Flutter) — interfejs konsumenta: dołączanie do kolejki, śledzenie pozycji, anulowanie, powiadomienia push.
- Panel administracyjny (Flutter Web) — interfejs dla personelu: wywoływanie następnej osoby, zarządzanie kolejką, statystyki.
- Wyświetlacz (Flutter Web) — publiczny ekran z numerami aktualnie obsługiwanych klientów.

Organizacja aplikacji

- Kontrolery — REST API, walidacja DTO, mapowanie na encje
- Serwisy — logika biznesowa, zarządzanie transakcjami
- Repozytoria — dostęp do bazy PostgreSQL
- Publikacja zdarzeń — publikacja zdarzeń domenowych do RabbitMQ
- Konsumenci — nasłuchiwanie kolejek i przetwarzanie zdarzeń
- Harmonogramy — zadania cykliczne (np. czyszczenie starych danych)

Decyzje techniczne

- Monolit zamiast mikroserwisów — prostota wdrożenia i utrzymania dla małego zespołu
- Jedna wymiana typu topic — elastyczne routing, łatwe dodawanie nowych konsumentów bez zmian w producencie
- Trzy kolejki — izolacja awarii, niezależne ponowienia dla każdego konsumenta
- Flutter dla wszystkich aplikacji — jeden framework, jeden zespół, współdzielenie kodu między platformami

Baza Podstawa

Opis tabelka

Kolejki

Tabela przechowuje kolejki, które obsługują konkretne usługi/zasoby. Każda kolejka ma własną konfigurację (czas wizyty, limit klientów, interwał ostrzegania klienta), jeden opiekuna (kierownika), kolejkę do powiadomień na Kafkie i status.

Kolumna	Typ w postgresie	Opis/ograniczenia
id_kolejki	BIGSERIAL	primary key
nazwa	VARCHAR(255) NOT NULL	nazwa kolejki
opis	VARCHAR	opis kolejki
czas_wizyty	INTEGER NOT NULL	czas wizyty w minutach
limit_klientow	INTEGER	maksymalna liczba klientów w kolejce
interwal_ostrzegania	INTEGER NOT NULL	interwał ostrzegania klienta w minutach
id_opiekuna	BIGINT NOT NULL	id opiekuna (kierownika)
kafka_topic	VARCHAR	nazwa tematu Kafka dla powiadomień
status	VARCHAR NOT NULL	status kolejki (np. OTWARTA, ZAMKNIĘTA, WSTRZYMANA)

Klienci

Tabela przechowuje dane klientów. Jeden klient może być w wielu kolejkach (relacja przez tabelę kolejki_klienci). Dane osobowe (imię, nazwisko, email, telefon) są opcjonalne — klient może być anonimowy.

Kolumna	Typ w postgresie	Opis/ograniczenia
id_klienta	BIGSERIAL NOT NULL	primary key
imie	VARCHAR	imię klienta
nazwisko	VARCHAR	nazwisko klienta
email	VARCHAR NOT NULL	adres email (unikalny)
telefon	VARCHAR	numer telefonu
data_rejestracji	TIMESTAMP NOT NULL	data i czas rejestracji

Kolejki_klienci

Tabela łączy klientów z kolejkami (relacja wielu-do-wielu). Każdy wpis reprezentuje jeden udział klienta w kolejce. Zawiera pozycję w kolejce, czas dołączenia, czas wezwania i czas zakończenia.

Kolumna	Typ w postgresie	Opis/ograniczenia
id_kolejki_klienta	BIGSERIAL NOT NULL	primary key
id_kolejki	BIGINT NOT NULL	id kolejki
id_klienta	BIGINT NOT NULL	id klienta
pozycja	INTEGER NOT NULL	pozycja w kolejce
czas_dolaczenia	TIMESTAMP NOT NULL	czas dołączenia do kolejki
czas_wezwania	TIMESTAMP	czas wezwania klienta
czas_zakonczenia	TIMESTAMP	czas zakończenia (obsługi lub anulowania)

Pracownicy

Tabela przechowuje dane pracowników (kierowników). Jeden pracownik może być opiekunem wielu kolejek.

Kolumna	Typ w postgresie	Opis/ograniczenia
id_pracownika	BIGSERIAL NOT NULL	primary key
imie	VARCHAR NOT NULL	imię pracownika
nazwisko	VARCHAR NOT NULL	nazwisko pracownika
email	VARCHAR NOT NULL	adres email (unikalny)
telefon	VARCHAR	numer telefonu
data_zatrudnienia	DATE NOT NULL	data zatrudnienia

Powiadomienia

Tabela przechowuje wysłane powiadomienia (SMS, email, push). Jeden klient może mieć wiele powiadomień. Zawiera typ, treść, status (WYSŁANE, NIEPOWODZENIE, OCZEKUJĄCE) i ewentualny komunikat błędu.

Kolumna	Typ w postgresie	Opis/ograniczenia
id_powiadomienia	BIGSERIAL NOT NULL	primary key
id_klienta	BIGINT NOT NULL	id klienta
typ	VARCHAR NOT NULL	typ powiadomienia (SMS, EMAIL, PUSH)
tresc	TEXT NOT NULL	treść powiadomienia
status	VARCHAR NOT NULL	status (WYSŁANE, NIEPOWODZENIE, OCZEKUJĄCE)
czas_wysylki	TIMESTAMP	czas wysłania
komunikat_bledu	TEXT	komunikat błędu (jeśli nie powiodło się)

Statystyki_dzienne

Tabela przechowuje dzienne agregaty statystyk dla każdej kolejki (jeden wiersz na kolejkę i datę). Zawiera liczbę obsłużonych, anulowanych, pominiętych klientów oraz średnie czasy obsługi i oczekiwania.

Kolumna	Typ w postgresie	Opis/ograniczenia
id_statystyki	BIGSERIAL NOT NULL	primary key
id_kolejki	BIGINT NOT NULL	id kolejki
data	DATE NOT NULL	data agregacji
liczba_obsluzonych	INTEGER NOT NULL	liczba obsłużonych klientów
liczba_anulowanych	INTEGER NOT NULL	liczba anulowanych klientów
liczba_pominietych	INTEGER NOT NULL	liczba pominiętych klientów
srednia_czas_obslugi	INTEGER	średni czas obsługi (minuty)
srednia_czas_oczekiwania	INTEGER	średni czas oczekiwania (minuty)

Diagram tabel podstawa

Diagramy razrzeda

Diagramy razrzeda przedstawiają, które encje/tabele są ze sobą powiązane, jak wygląda ich struktura i jakie są rodzaje relacji (one-to-many, many-to-many). Pomagają zrozumieć model danych przed implementacją — widać od razu, gdzie są klucze obcy, gdzie są tabele pośredniczące (junction tables), a gdzie są encje niezależne.

Diagram razrzeda - aplikacja/backend .png

Diagram razrzeda - wyświetlacz.jpg

Diagram razrzeda - panel.jpg

Diagram razrzeda - kafka.jpg

Diagram klas - aplikacja/backend

Diagram klas pokazuje strukturę kodu — jakie klasy istnieją, jakie mają pola i metody, i jak się ze sobą komunikują. Pokazuje warstwy aplikacji (kontrolery, serwisy, repozytoria), encje, DTO, enumy i ich relacje. Przydatny przy implementacji — wiadomo, jakie klasy trzeba napisać i jak się ze sobą połączą.

Diagram klas - kolejka

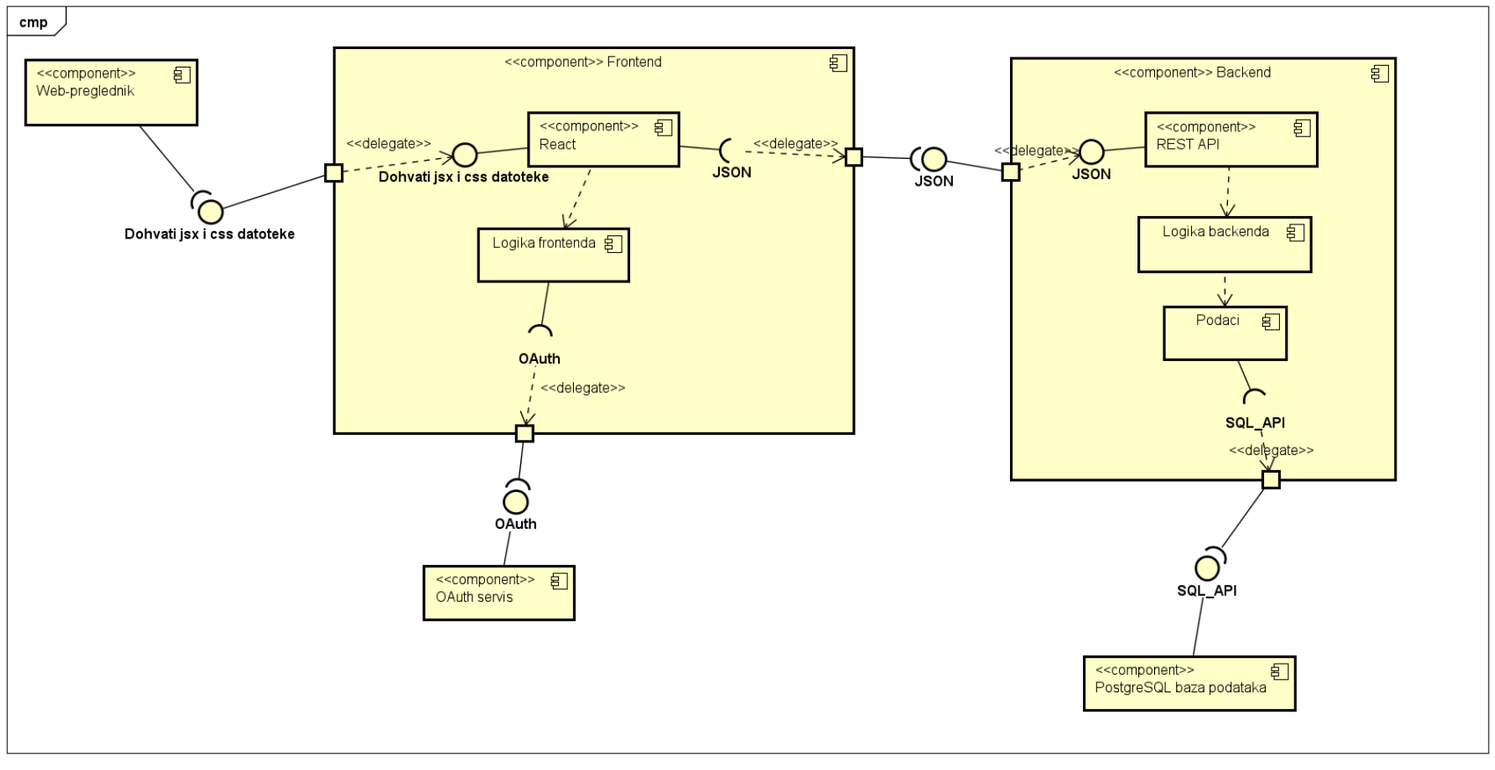
KolejkaController
+ getAll()
+ getById(id)
+ create(request)
+ update(id, request)
+ delete(id)
+ wezwanie(id)
+ zmienLimit(id, limit)
+ zmienInterwal(id, interwal)
+ zmienOpiekuna(id, opiekunId)
+ zmienStatus(id, status)
+ getPozycja(id, klientId)
KolejkaService
+ getAll()
+ getById(id)
+ create(request)
+ update(id, request)
+ delete(id)
+ wezwanie(id)
+ zmienLimit(id, limit)
+ zmienInterwal(id, interwal)
+ zmienOpiekuna(id, opiekunId)
+ zmienStatus(id, status)
+ getPozycja(id, klientId)
+ publishEvent(event)
KolejkaRepository
+ findAll()
+ findById(id)
+ save(kolejka)
+ delete(id)
+ existsBy(id)
+ countBy(status)
+ findByOpiekunId(opiekunId)
KolejkaMapper
+ toDto(kolejka)
+ toEntity(request)
+ updateEntity(request, kolejka)
KolejkaRequest
+ nazwa
+ opis
+ czasWizyty
+ limitKlientow
+ interwalOstrzegania
+ idOpiekuna
+ kafkaTopic
+ status
KolejkaResponse
+ id
+ nazwa
+ opis
+ czasWizyty
+ limitKlientow
+ interwalOstrzegania
+ idOpiekuna
+ kafkaTopic
+ status
+ liczbaOczekujacych
+ liczbaObsluzonych
+ liczbaAnulowanych
+ liczbaPominietych
+ sredniaCzasObslugi
+ sredniaCzasOczekiwania
+ czasOtwarcia
+ czasZamknicia
+ aktualnie
+ pozycja
+ czasDolaczenia
+ czasWezwania
+ czasZakonczenia
+ klienci
+ pracownicy
+ powiadomienia
+ statystyki
+ kolejki

5. Arhitektura komponenata i razmještaja

Mia Zekić edited this page yesterday · 7 revisions

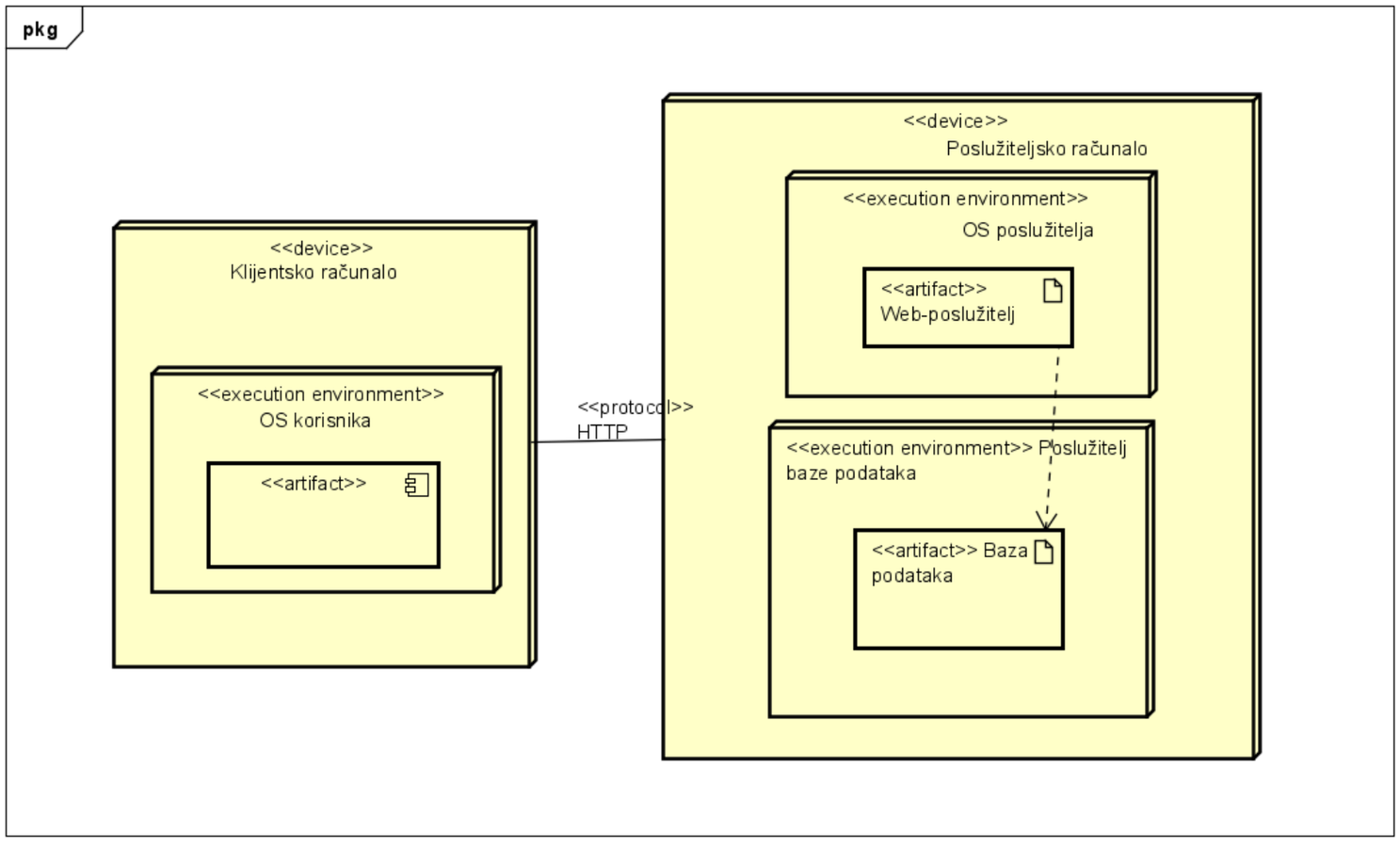
Dijagram komponenata

Sljedeći dijagram komponenata predstavlja bitne dijelove aplikacije koje obavljaju različite funkcije. Vizualizira organizaciju i međuovisnost između komponenti. Komponenta Web-preglednik spojnicom je povezana a komponentom Frontenda od koje zahtjeva jsx i css datoteke potrebne za prikaz korisničkog sučelje na Web-pregledniku. Unutar Frontend komponente nalaze se komponenta React-a te cijela logika frontenda (App: ProfilePage, HomePage, Lections, LoginPage, RegisterPage...). Dio logike Frontenda je login/register koji od OAuth servira zahtijevaju prijavu. Frontend i Backend povezani su tako što Frontend zahtjeva od Backenda JSON podatke. Backend komponenta sastoji se od Rest Apija, logike Backenda te podataka. Podatke zatražuje od PostgreSQL baze podataka.



Dijagram razmještaja

Dijagram razmještaja vrsta je UML-dijagrama koji prikazuju fizičku arhitekturu i konfiguraciju razmještaja programskog sustava. Klijentsko računalo pristupa aplikaciji putem web preglednika i protokola HTTPS. Web-poslužitelj i poslužitelj baze podataka nalaze se na istom računalu.



Pages 13
Find a page…
Home
1. Opis projektnog zadatka
2. Analiza zahtjeva
3. Specifikacija zahtjeva sustava
4. Arhitektura i dizajn sustava
5. Arhitektura komponenata i razmj…
Dijagram komponenata
Dijagram razmještaja
6. Ispitivanje programskog rješenja
7. Tehnologije za implementaciju a…
8. Upute za puštanje u pogon
9. Zaključak i budući rad
A. Dnevnik promjena dokumentacije
B. Popis literature
C. Prikaz aktivnosti grupe

Clone this wiki locally

https://github.com/mihovil13/Ling

6. Ispitivanje programskog rješenja

Mia Zorić edited this page 8 months ago · 23 revisions

Ispitivanje komponenti

```
✓ Test                                  1 sec 870 ms
  ✓ testValidPredavanje()               870 ms
  ✓ testServiceSave()                   0 ms
  ✓ testRegisterFail()                  0 ms
  ✓ testCreateUser_NoImportance()       0 ms
  ✓ testOneByEmail()                    0 ms
  ✓ testSendEmail()                     0 ms
  ✓ testAddBackup()                     0 ms
```

Prvi ispitni slučaj - tzazivanje pogreške (exception throwing)

1. Registracija vaš registriranim emailom
2. Ispitni slučaj
   - Ulazni podaci: email "kera.mari@gmail.com", korisnik "123456", uloga "UStud"
   - Očekivani rezultat: registracija neuspješna, izazivanje pogreške
   - Dobiveni rezultat: prikaz ispitivanja
3. U bazi podataka postoji "kera.mari@gmail.com" te se pokuša stvarajuć da je osoba već registrirana. Test je uspješan.
4. KSE

```
Test  ok
public void testRegisterFail() {
    Korisnik korisnik = new Korisnik();
    korisnik.setEmail("kera.mari@gmail.com");
    korisnik.setPass("123456");
    korisnik.setPrezime("Mario");
    korisnik.setLozinka("123456");
    korisnik.setUlog("UStud");

    assertThrows(IllegalArgumentException.class, () -> {
        korisnikService.registerUser(korisnik);
    });
}
```

Drugi ispitni slučaj - Nepostojeće funkcionalnosti

1. Brisanje korisnika
2. Navedite ispitni slučaj
   - Ulazni podaci: stvoreni Korisnik email "korisnik.za.brisanje@gmail.com"
   - Očekivani rezultat: korisnik uspješno spremljen u bazu
   - Dobiveni rezultat: prikaz ispitivanja
3. Pokaz za novog Korisnika sa email = "korisnik@gmail.com", ime="Ana", prezime="Horvat", korisnik="pasivaxist" te ulogo="UAnk". Nakon definiranja objekta Korisnika, taj objekt pokušamo spremiti u bazu pomoću funkcije save().
4. KSE

```
Test  ok
public void testSaveUser() {
    Korisnik korisnik = new Korisnik();
    korisnik.setEmail("ana@gmail.com");
    korisnik.setPass("ana");
    korisnik.setPrezime("Horvat");
    korisnik.setLozinka("123456");
    korisnik.setUlog("UAnk");

    Korisnik savedKorisnik = korisnikRepository.save(korisnik);

    Assertions.assertNotNull(savedKorisnik);
    Assertions.assertEquals("ana@gmail.com", savedKorisnik.getEmail());
}
```

Četvrti ispitni slučaj - redovan slučaj

1. Brisanje korisnika pomoć korisničkog jedinstvenog ID-ja.
2. Navedite ispitni slučaj
   - Ulazni podaci: pohrana novog Korisnik s emailom "ana_hr@gmail.com"
   - Očekivani rezultat: Korisnik se čeli uklonjen iz baze funkcijom deleteById()
   - Dobiveni rezultat: prikaz ispitivanja
3. Pruz je parametre spremni novog Korisnik s bazu. Podaci za to su: email = "ana_hr@gmail.com", ime="Ana", prezime="Horvat", korisnik="pasivaxist" te ulogo="UAnk". Nakon toga s KorisnikRepository-jem briše Korisnika s tim ID-om.
4. KSE

```
Test  ok
public void testDeleteById() {
    Korisnik korisnik = new Korisnik();
    korisnik.setEmail("ana_hr@gmail.com");
    korisnik.setPass("ana");
    korisnik.setPrezime("Horvat");
    korisnik.setLozinka("123456");
    korisnik.setUlog("UAnk");

    Korisnik savedKorisnik = korisnikRepository.save(korisnik);
    Integer id = savedKorisnik.getUser_id();

    korisnikRepository.deleteById(id.valueOf(id));
    Assertions.assertFalse(korisnikRepository.findById(id.valueOf(id)).isPresent());
}
```

Peti ispitni slučaj - redovan slučaj

1. Dohvat objekta Korisnik pomoću emaila
2. Navedite ispitni slučaj
   - Ulazni podaci: Pohranjen novi Korisnik s emailom "ana.horvat@gmail.com", pokušaj dohvata podataka Korisnik pomoću emaila
   - Očekivani rezultat: dohvat Korisnika se čeli uspješan, Korisnik neće biti null
   - Dobiveni rezultat: prikaz ispitivanja
3. Kada želimo pročitati podatkove Korisnika dohvaćamo ga pomoću jedinstvenog emaila. Ovaj test pokazuje dohvat pomoću emaila te provjeru je li Korisnik pronađen. Pruz je potrebno spremiti novog Korisnika s bazu. Podaci za to su: email = "ana.horvat@gmail.com", ime="Ana", prezime="Horvat", korisnik="pasivaxist" te ulogo="UAnk".
4. KSE

```
Test  ok
public void testGetByEmail() {
    String email = "ana.horvat@gmail.com";
    Korisnik korisnik = new Korisnik();
    korisnik.setEmail(email);
    korisnik.setPass("ana");
    korisnik.setPrezime("Horvat");
    korisnik.setLozinka("123456");
    korisnik.setUlog("UAnk");

    korisnikRepository.save(korisnik);

    Optional<Korisnik> savedKorisnik = korisnikRepository.findByEmail(email);
    Assertions.assertTrue(savedKorisnik.isPresent());
    Assertions.assertEquals(email, savedKorisnik.get().getEmail());
}
```

Šesti ispitni slučaj - redovan slučaj

1. Pohrana novog predavanja u bazu
2. Navedite ispitni slučaj
   - Ulazni podaci: stvaranje novog objekta Predavanje s novom učionikom i učiteljem te izmuzetnim vremenom
   - Očekivani rezultat: Predavanje je spremljeno u bazu
   - Dobiveni rezultat: prikaz ispitivanja
3. Pruz je pohrane novi učionik i učitelj te se ovi postave u predavanje, s izmuzetnim vremenom. Predavanje je pohrani te se projekt je k pohranjeno.
4. KSE

```
Test  ok
public void testPredavanje() {
    Predavanje predavanje = new Predavanje();
    predavanje.setUsersUsersId(1L);
    predavanje.setUcionicaId(1L);
    predavanje.setDatumVremenoOd("2025-05-05 10:00:00");
    predavanje.setDatumVremenoDo("2025-05-05 12:00:00");

    Predavanje saved = predavanjeRepository.save(predavanje);
    assertNotNull(saved.getId());
    assertEquals(predavanje.getUsers_users_id(), saved.getUsers_users_id());
}
```

Sedmi ispitni slučaj - redovan slučaj

1. Pohrana nove recenzije u bazu
2. Navedite ispitni slučaj
   - Ulazni podaci: stvaranje novog objekta Predavanje s novom učionikom i učiteljem te izmuzetnim "5" i komentarom "super"
   - Očekivani rezultat: Predavanje je spremljeno u bazu
   - Dobiveni rezultat: prikaz ispitivanja
3. Pruz je pohrane novi učionik i učitelj te se ovi postave u recenziju, s ocjenom "5" i komentarom "super". Recenzija se pohrani te se projekt je k pohranjeno.
4. KSE

```
Test  ok
public void testRecenzija() {
    Recenzija recenzija = new Recenzija();
    recenzija.setUsersUsersId(1L);
    recenzija.setPredavanjeId(1L);
    recenzija.setOcjena("5");
    recenzija.setKomentar("super");

    Recenzija saved = recenzijaRepository.save(recenzija);
    assertNotNull(saved.getId());
    assertEquals(recenzija.getOcjena(), saved.getOcjena());
}
```

Ispitivanje sustava

Prvi ispitni slučaj - Prijava te promjena podataka

1. Ulazi:
   - podaci koji se unose u prijavu: email "kera.kari@gmail.com", korisnik "kolegica123"
   - prikaz podataka na prijavu, mijenjanje podataka te njihov spremanje
2. Koraci ispitivanja
   a. Otvoriti aplikaciju u preglednika.
   b. Unijeti podatke u formu.
   c. Pritisak gumba za prijavu.
   d. Verificirati očekivane rezultate.
   e. Pritisak gumba za uredivanje profila.
   f. Unijeti podatke za promjenu.
   g. Pritisak gumba za pohranu podataka.
3. Očekivani ishod:
   - Očekivani rezultat ispitivanja: korisnik nakon prijave preusmjeren je na stranicu profila te su pohranjeni podaci vidljivi na profilu.
4. Dobiveni ishod:

```
Running "promjena_podataka_uvrste"
1. open on / OK
2. click on / on/ profile-link OK
3. click on / on/ edit-profile-button OK
4. click on / on/ pass-login-input-1 / button OK
5. mouseClick on / on/ pass-login-input-1 / button OK
6. mouseClick on / on/ pass-login-input-1 / button OK
7. click on / on/ pass-login-input-1/ select-info-chkbx0 OK
8. select on / on/ pass-login-input-1/ select-info-chkbx0/ Ulogovan OK
9. click on / on/ select-info-chkbx0 OK
10. select on / on/ select-info-chkbx0/ select-info-Sveobja OK
11. click on / on/ pass-login-input-1 / button OK
12. mouseClick on / on/ pass-login-input-1 / button OK
13. mouseClick on / on/ button-info-chkbx0 OK
14. click on / on/ div-info-chkbx0 / select-info-chkbx0 OK
15. select on / on/ div-info-chkbx0 / select-info-chkbx0/ select-info-Ripmacho OK
16. click on / on/ div-info-chkbx0 / select-info-chkbx0 OK
17. click on / on/ spinnerLoaded / button-info-chkbx0 OK
18. assertElementPresent on / on/ div-info-chkbx0 / OK
'promjena_podataka_uvrste' completed successfully
```

Drugi ispitni slučaj - Odabir učitelja, termina te slanje zahtjeva za termin (Učenik)

1. Ulazi:
   - podaci koji se unose u prijavu: email "ana.kari@gmail.com", korisnik "kolegica123"
   - prikaz podataka na prijavu, odlazak na listu učitelja, pritisak na učitelja, prikaz kalendara učitelja te slanje zahtjeva
2. Koraci ispitivanja
   a. Prijavit se
   b. Otvoriti popis učitelja te odabrati učitelja.
   c. Pristupiti kalendaru na profilu učitelja.
   d. Pritisnuti želijeni dan te odabrati vremne.
   e. Pritisnuti potvrdu slanja zahtjeva.
   f. Verificirati očekivane rezultate, zahtjev se nalazi u "Moji zahtjevi"
3. Očekivani ishod:
   - Očekivani rezultat ispitivanja: zahtjev če se poslati te nalazi u "Moji zahtjevi"
4. Dobiveni ishod:

```
Running "kontrol2"
1. open on http://localhost:3000/ OK
2. click on / on/ profile-link OK
3. click on / on/ teachers-button OK
4. click on / on/ teachers-container/ mb-chkbx15 OK
5. click on / on/ calendar-button OK
6. click on / on/ mb-chkbx15 / 15-day-slot 15-daysl-day-events OK
7. click on / on/ time-button mb-chkbx15 OK
8. click on / on/ confirm-button OK
9. waitUntilElementPresent on /on/ modal-content with value 30000 OK
10. assertElementPresent on /on/ modal-content OK
'kontrol2' completed successfully
```

Treći ispitni slučaj - Odobrenje zahtjeva za termin (Učitelj)

1. Ulazi:
   - Korisnicki podaci koji se unose u sustav email "kera.mari@gmail.com", korisnik "ziaxs123"
   - Učitelj se prijavi, pritisne zahtjev, pritisnim odobri zahtjev na Sakaru
2. Koraci ispitivanja
   a. Prijavit se.
   b. Pritisak na gumb za zahtjeve.
   c. Kliknut na gumb za potvrdu.
   d. Verificirati očekivane rezultate, u "Moji zahtjevi" zahtjev označen kao "Prihvaćen".
3. Očekivani ishod:
   - Očekivani rezultat ispitivanja: Zahtjev če se odobriti, te prikazati kao prihvaćen u "Moji zahtjevi"
4. Dobiveni ishod:

```
Running "kontrol7"
1. open on http://localhost:3000/ OK
2. click on / on/ profile-link OK
3. click on / on/ zahtjevi-button OK
4. click on / on/ request-container mb-chkbx15/ accept-button OK
5. click on / on/ close-button OK
6. click on / on/ status-accepted OK
'kontrol7' completed successfully
```

Četvrti ispitni slučaj - Filtriranje učitelja po preferencijama

1. Ulazi:
   - Prijava:
   - Pristupit pregledu učitelja te filtriraj po učiteljima koji predaju odabrane predmete (engleski).
2. Koraci ispitivanja
   a. Prijava u aplikaciju.
   b. Pritisak na gumb pregleda učitelja.
   c. U filteru odabrati engleski jezik.
   d. Verificirati očekivane rezultate, u pregledu samo učitelji koji predaju engleski.
3. Očekivani ishod:
   - Očekivani rezultat ispitivanja: prikaz dva učitelja koja predaju engleski.
4. Dobiveni ishod:

```
Running "kontrol4"
1. open on http://localhost:3000/ OK
2. click on / on/ profile-link OK
3. click on / on/ teachers-button OK
4. click on / on/ filter-group mb-chkbx15 / mb-chkbx15 / input OK
5. click on / on/ filter-button / button OK
6. click on / on/ container OK
7. mouseOverOn on / on/ teacher-container mb-chkbx15 with value 37.399992095404344,12.0 OK
8. mouseMoveOn on / on/ teacher-container mb-chkbx15 with value 37.399992095404344,12.0 OK
9. click on / on/ teacher-container mb-chkbx15 OK
'kontrol4' completed successfully
```

Peti ispitni slučaj - (Nepostojeća funkcionalnost) Pristup stranici galerije slika

1. Ulazi:
   - Prijava, nepripadjan korisnik
   - Otvoriti prazan pristupit stranici galerije slika te kliknuti na odredenu sliku
2. Koraci ispitivanja
   a. Pritisak na gumb ulaz na početnoj stranici.
   b. Pristup stranici galerije.
   c. Klik na sliku.
3. Očekivani ishod:
   - Očekivani rezultat ispitivanja: ne može se otvoriti stranica galerije.
4. Dobiveni ishod:

```
'kontrol' ended with 1 error(s)
Running "kontrol7"
1. open on http://localhost:3000/ OK
2. setWindowSize on 1920x1040 OK
3. click on / on/ hero-sekcija-text-wrap OK
4. assertLocation on http://localhost:3000/galery Failed
Unknown command: assertLocation
'kontrol7' ended with 1 error(s)
```

Pages 19
Find a page...
Home
1. Opis projektnog zadatka
2. Analiza zahtjeva
3. Specifikacija zahtjeva sustava
4. Arhitektura i dizajn sustava
5. Arhitektura komponenata i razreda
6. Ispitivanje programskog rješenja
Ispitivanje komponenti
Prvi ispitni slučaj - tzazivanje pogreške (exception throwing)
Drugi ispitni slučaj - Nepostojeće funkcionalnosti
Treći ispitni slučaj - redovan slučaj
Četvrti ispitni slučaj - redovan slučaj
Peti ispitni slučaj - redovan slučaj
Šesti ispitni slučaj - redovan slučaj
Sedmi ispitni slučaj - redovan slučaj
Ispitivanje sustava
Prvi ispitni slučaj - Prijava te promjena podataka
Drugi ispitni slučaj - Odabir učitelja, termina te slanje zahtjeva za termin (Učenik)
Treći ispitni slučaj - Odobrenje zahtjeva za termin (Učitelj)
Četvrti ispitni slučaj - Filtriranje učitelja po preferencijama
Peti ispitni slučaj - (Nepostojeća funkcionalnost) Pristup stranici galerije slika
7. Tehnologije za implementaciju a...
8. Izazov za pažljivije u pisanju
9. Zaključak i budući rad
A. Dnevnik promjena dokumentacije
B. Popis literature
C. Prikaz aktivnosti grupe

Clone this wiki locally
https://github.com/mihovil13/LongueLink.wiki

7. Tehnologije za implementaciju aplikacije

Mia Zekić edited this page yesterday · 7 revisions

Korištene tehnologije i alati

1. Programski jezici: Za implementaciju aplikacije korišteni su JavaScript i Java 19. JavaScript je odabran kao glavni jezik za razvoj prednjeg dijela aplikacije zbog svoje svestranosti i široke primjene u web razvoju. Java 19 korištena je za pozadinski dio aplikacije, jer nudi visoke performanse i moderni ekosustav koji uključuje bogate biblioteke i alate za razvoj poslovnih aplikacija.

2. Radni okviri i biblioteke: Za razvoj prednjeg dijela aplikacije korišten je React 18, najpopularnija JavaScript biblioteka za izgradnju korisničkih sučelja, koju održava Facebook. React omogućava brzu i efikasnu izradu dinamičkih web aplikacija, dok se JavaScript XML (JSX) koristi kao sintaktička ekstenzija koja omogućava integraciju HTML-a unutar JavaScript koda, što olakšava rad s komponentama. Za razvoj pozadinskog dijela aplikacije korišten je Spring Boot 3.3.5, što je najnovija verzija popularnog radnog okvira za Javu. Spring Boot omogućava brzo postavljanje web aplikacija s minimalnim konfiguracijama, dok Node.js 20 omogućava izgradnju brze i skalabilne serverske strane aplikacije.

3. Baza podataka: Za pohranu podataka korišten je PostgreSQL 16, snažan sustav za upravljanje bazama podataka poznat po svojoj pouzdanosti, skalabilnosti i bogatstvu značajki. Za administraciju baze podataka korišten je pgAdmin 4, koji omogućava jednostavan vizualni pristup i upravljanje PostgreSQL bazama. Za modeliranje baze podataka korišten je ERDPlus alat, koji omogućava jednostavno crtanje ER dijagrama.

4. Razvojni alati: Za razvoj aplikacije korišteni su popularni razvojni alati poput VS Code i IntelliJ IDEA. VS Code je korišten za razvoj front-end dijela aplikacije zbog svoje brzine i podrške za mnoge ekstenzije koje pojednostavljuju rad s Reactom i JavaScript-om. IntelliJ IDEA korišten je za razvoj back-end dijela aplikacije, s obzirom na njegovu izvrsnu podršku za Java i Spring Boot. Za kontrolu verzija korišten je Git 2.45 u kombinaciji s GitHub Desktop za jednostavno praćenje i upravljanje promjenama u kodu. Repozitorij projekta dostupan je na platformi GitHub

5. Alati za ispitivanje: Aplikacija je testirana pomoću Selenium IDE 4.27 i Postman 11. Selenium IDE korišten je za automatizaciju testova korisničkog sučelja, dok je Postman korišten tijekom izrade za testiranje upita na pozadinski dio aplikacije.

6. Alati za razmještaj: Aplikacija i baza podataka postavljeni su na Render, cloud platformu koja nudi besplatnu opciju za razmještaj aplikacija i baza podataka s ograničenim resursima.

7. Komunikacija: Tijekom izrade aplikacije komunikacija između članova tima se vodila putem aplikacija WhatsApp i Discord.

Pages 13
Find a page…
Home
1. Opis projektnog zadatka
2. Analiza zahtjeva
3. Specifikacija zahtjeva sustava
4. Arhitektura i dizajn sustava
5. Arhitektura komponenata i razmj…
6. Ispitivanje programskog rješenja
7. Tehnologije za implementaciju a…
Korištene tehnologije i alati
8. Upute za puštanje u pogon
9. Zaključak i budući rad
A. Dnevnik promjena dokumentacije
B. Popis literature
C. Prikaz aktivnosti grupe

Clone this wiki locally

https://github.com/mihovil13/Ling

8. Upute za puštanje u pogon

Mia Zekić edited this page 6 minutes ago · 15 revisions

1. Instalacija

- **preduvjeti**: instaliran JDK 21 ili novija verzija, PostgreSQL baza podataka, Node.js verzija 18 ili novija verzija, instaliran npm, Docker verzija 26 ili novija
- **preuzimanje**: kloniranje repozitorija s Git-a:

  ```
  git clone https://github.com/mihovil13/LinguaLink
  ```

  ```
  cd repo
  ```

  u direktoriju "Frontend" pokrenuti

  ```
  npm install
  ```

2. Postavke

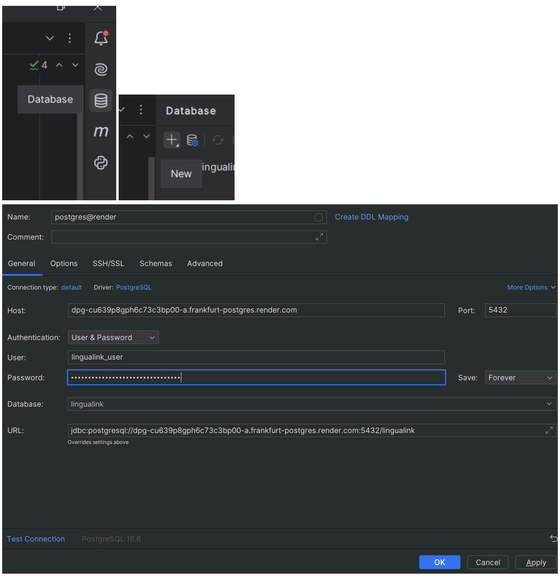
- **Konfiguracijske datoteke:**: u datoteci application.properties definirana je baza:

  ```
  spring.datasource.url=jdbc:postgresql://dpg-cu639p8gph6c73c3bp00-a:5432/lingualink
  ```

  ```
  spring.datasource.username=lingualink_user
  ```

  ```
  spring.datasource.password=P9cmFBGv5MiAEgAU89DNJnDfpbUxAkDB
  ```

PostgreSQL baza se nalazi na servisu Render te se na nju može povezati sa sljedećim postavkama: primjer za Intellij IDEA:



- **Postavke baze podataka**: PostgreSQL baza se nalazi na servisu Render. Za bazu podataka korišteno je Hibernate automatsko generiranje. Tablice se kreiraju automatski na temelju entiteta (@Entity u kôdu). primjer kôda:

  ```
  @Entity
  ```

  ```
  @Table(name = "recenzije")
  ```

  ```
  public class Recenzija {
  ```

  ...

  U bazi podataka stvara se tablica "recenzije"

3. Pokretanje aplikacije

Za backend:

- **Razvojno okruženje:** `mvn spring-boot:run`
- **Produkcijsko okruženje:** `java -jar target/app.jar`

Za Frontend: u direktoriju Frontenda

- **Razvojno okruženje:** `npm run dev`
- **Produkcijsko okruženje:**
  - **Prevođenje aplikacije:** `npm run build`
- **Pokretanje poslužitelja:** `npm start`
- **Provjera rada:** otvoriti preglednik te otići na http://localhost:3000/

Pošto je aplikacija deployana frontendu je moguće pristupiti linkom https://lingualink-frontend-v1.onrender.com/ , no kada se kod mijenja i želi se prikazati frontend pristup je pomoću localhosta. Backend je već pokrenut za oba slučaja.

4. Upute za administratore

- **Pristup administratorskom sučelju:**
  - URL za admin panel: /users
  - Početni podaci za prijavu: email: "ivan.horvat@gmail.com", lozinka: "ovojeadminsifra"
- **Redovito održavanje:**
  - Arhiviranje baze podataka.
  - Pregled logova.
  - Ažuriranje aplikacije - povlačenje novih verzija iz Git repozitorija i ponovno pokretanje aplikacije.

    ```
    git pull origin main
    ```

    ```
    npm install
    ```

    ```
    npm run build
    ```

    ```
    npm start
    ```
- **Rješavanje problema:** pristup log-ovima te dijagnosticiranje grešaka može se naći u Render konzoli

5. Deploy platformom Render

- Potrebne Docker datoteke:
  - Za backend:

    ```
    RUN chmod +x mvnw
    ```

    ```
    RUN ./mvnw clean package
    ```

    ```
    FROM openjdk:21-jdk-slim
    ```

    ```
    COPY --from=builder target/*.jar /app.jar
    ```
  - Za frontend:

    ```
    WORKDIR /app
    ```

    ```
    COPY ./IzvorniKod/frontend/package*.json ./
    ```

    ```
    RUN npm install
    ```
- Postavljanje na Render:
  - Kreirajte novi Web Service i povežite ga s vašim GitHub repozitorijem.
  - Konfigurirajte postavke: `Backend: java -jar /app.jar` , `Frontend: npm start`
  - Dodajte environment varijable (npr. DATABASE_URL, API_KEY).
- Pokretanje aplikacije: https://lingualink-frontend-v1.onrender.com/

Opis pristupa aplikaciji na javnom poslužitelju

- **Ograničenja:** Render ima ograničenja, na Frontendu je dostupno 2 GB RAM-a i 1 CPU, za backend dostupno je 512 MB RAM-a i 0.5 CPU te za bazu dostupno je 256 MB RAM-a i 1 GB pohrane. Postavljena slika profila mora biti u PNG formatu i maksimalne veličine do 3 MB.
- **Pristup aplikaciji putem internetskog preglednika:** aplikaciji se može pristupiti pomoću URL: https://lingualink-frontend-v1.onrender.com/
- **Koraci za pristup administratorskom sučelju:**
  - Početni podaci za prijavu: email: "ivan.horvat@gmail.com", lozinka: "ovojeadminsifra"

Pages 13
Find a page...
Home
1. Opis projektnog zadatka
2. Analiza zahtjeva
3. Specifikacija zahtjeva sustava
4. Arhitektura i dizajn sustava
5. Arhitektura komponenata i razmj...
6. Ispitivanje programskog rješenja
7. Tehnologije za implementaciju a...
8. Upute za puštanje u pogon
1. Instalacija
2. Postavke
3. Pokretanje aplikacije
4. Upute za administratore
5. Deploy platformom Render
Opis pristupa aplikaciji na javnom poslužitelju
9. Zaključak i budući rad
A. Dnevnik promjena dokumentacije
B. Popis literature
C. Prikaz aktivnosti grupe

Clone this wiki locally

https://github.com/mihovil13/Ling

9. Zaključak i budući rad

Mia Zekić edited this page 6 minutes ago · 8 revisions

Cilj projekta bio je razviti web-aplikaciju koja će povezivati učenike i učitelje različitih jezika, gdje učenici šalju zahtjeve učiteljima za online lekcije odabranog predmeta. Tim se sastojao od sedam članova koji su radili na projektu tijekom zimskog semestra.

Tehnički izazovi:

Kako nitko iz tima nije prije samostalno radio aplikaciju, svi smo se našli s mnogim tehničkim izazovima. Većina je vremena potrošena na istraživanje o alatima u kojima radimo te na uklanjanje grešaka. Jedan od izazova bio je povezati aplikaciju s PostgreSQL bazom podataka što je tek uspješno obavljeno u drugom ciklusu semestra. Sigurnost aplikacije također je bila jedna od izazova, implementacija HTTPS-a, JWT tokeni te prijava pomoću OAuth-a. Deploy aplikacije bio je zadnji tehnički izazov do kojega smo došli.

Stečena znanja:

Tijekom razvijanja web-aplikacije, cijeli je tim stekao znanja osmišljavanja i dizajniranja aplikacije, razvoj poslužiteljske stane, osmišljavanje te implementiranje baze podataka, rad i komunikacija unutar grupe, podjele posla te pisanje dokumentacije.

Nastavak rada:

Za nastavak razvijanja aplikacije bilo bi potrebno osmisliti dodatne funkcionalnosti poput postavljanja PDF dokumenata nakon predavanja, verifikacija profesora i njihovog iskustva, ocjene učeniku za položene predmete, izraditi mobilnu aplikaciju...

Funkcionalnosti koje nisu napravljene:

Chat komunikacija između učenika i učitelja.

Pages 13
Find a page…
Home
1. Opis projektnog zadatka
2. Analiza zahtjeva
3. Specifikacija zahtjeva sustava
4. Arhitektura i dizajn sustava
5. Arhitektura komponenata i razmj…
6. Ispitivanje programskog rješenja
7. Tehnologije za implementaciju a…
8. Upute za puštanje u pogon
9. Zaključak i budući rad
Tehnički izazovi:
Stečena znanja:
Nastavak rada:
Funkcionalnosti koje nisu napravljene:
A. Dnevnik promjena dokumentacije
B. Popis literature
C. Prikaz aktivnosti grupe

Clone this wiki locally

https://github.com/mihovil13/Ling

© 2025 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

A. Dnevnik promjena dokumentacije

Mia Zekić edited this page 25 minutes ago · 23 revisions

Rev.	Opis promjene/dodatka	Autori	Datum
0.0	Napravljen predložak	Mađerić	23.10.2024
0.1.	Dodan osnovni opis projektnog zadatka	Mađerić	23.10.2024.
0.2	Dodani dionici i ostali zahtjevi	Jurgec, Stajku, Svjetličić, Zekić	24.10.2024.
0.3	Specifikacija programske potpore	Jurgec, Mađerić, Martinović	29.10.2024.
0.3.1	Opisivanje UC 1-6	Martinović	29.10.2024.
0.3.2.	Opisivanje UC 7-11	Jurgec	29.10.2024.
0.3.3.	Opisivanje UC 12-17	Mađerić	29.10.2024.
0.4	Arhitektura i dizajn sustava (baza podataka)	Šinjori, Zekić	29.10.2024.
0.4.1	Opis tablice baze podataka	Šinjori, Zekić	29.10.2024.
0.4.2.	Napravljen dijagram baze podataka	Šinjori, Zekić	30.10.2024
0.5.	Sekvencijski dijagrami, Provjera uključenosti ključnih funkcionalnosti u obrasce uporabe, Dijagrami obrasca uporabe	Jurgec, Mađerić, Martinović	30.10.2024.
0.5.1.	Sekvencijski dijagrami, Provjera uključenosti ključnih funkcionalnosti u obrasce uporabe	Jurgec	30.10.2024.
0.5.2.	Dijagrami obrasca uporabe	Martinović	30.10.2024.
0.6.	Opis arhitekture, obrazloženje odabira, organizacija arhitekture, dijagram baze	Zekić	13.11.2024.
0.7.	Funkcionalni zahtjevi uređeni, poredani po ID-jevima	Zekić	13.11.2024.
0.7.1.	Obrasci uporabe povezani sa funkcionalnim zahtjevima pomoću ID-jeva zahtjeva	Zekić	13.11.2024
0.7.2.	Uređeni nefunkcionalni zahtjevi	Zekić	14.11.2024.
0.8.	Dijagram podsustava, ažurirana organizacija aplikacije	Zekić	14.11.2024.
0.9.	Dovršavanje arhitekture i dizajna sustava	Zekić	14.11.2024.
0.10.	Dijagrami razreda	Jurgec, Zekić	15.11.2024.
0.11.	Pregled cijele dokumentacije i finalne izmjene	Martinović, Jurgec, Zekić	15.11.2024.
0.12.	Tehnologije za implementaciju aplikacije	Jurgec	15.1.2025.
0.13.	Ispitivanje programskog rješenja	Zekić	16.1.2025.
0.14.	Upute za puštanje u pogon	Zekić	18.1.2025.
0.15.	Dijagram stanja	Zekić	19.1.2025.
0.15.1.	Opis dijagrama stanja	Zekić	22.1.2025.
0.16.	Dijagram aktivnosti	Zekić	19.1.2025.
0.16.1.	Opis dijagrama aktivnosti	Zekić	22.1.2025.
0.17.	Budući rad i zaključak	Zekić	20.1.2025.
0.18.	Ažuriranje dijagrama razreda	Šinjori, Jurgec	24.1.2025.
0.19.	Pregled cijele dokumentacije i finalne izmjene	Zekić	24.1.2025.

Pages 13
Find a page…
Home
1. Opis projektnog zadatka
2. Analiza zahtjeva
3. Specifikacija zahtjeva sustava
4. Arhitektura i dizajn sustava
5. Arhitektura komponenata i razmj…
6. Ispitivanje programskog rješenja
7. Tehnologije za implementaciju a…
8. Upute za puštanje u pogon
9. Zaključak i budući rad
A. Dnevnik promjena dokumentacije
B. Popis literature
C. Prikaz aktivnosti grupe

Clone this wiki locally

https://github.com/mihovil13/Ling

© 2025 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

B. Popis literature

Mia Zekić edited this page 21 minutes ago · 1 revision

1. Programsko inženjerstvo, FER ZEMRIS, http://www.fer.hr/predmet/proinz
2. Nikolina Frid, Alan Jović, Modeliranje programske potpore UML-dijagramima E-skripta (radna inačica)
3. Nikolina Frid, Alan Jović, Procesi programskog inženjerstva
4. Vlado Sruk, predložak dokumentacije https://github.com/VladoSruk/Programsko-inzenjerstvo
5. Dokumentacija programskog alata Astah, https://astah.net/support/astah-pro/user-guide/ (siječanj 2025.)
6. Dokumentacija Spring Boot-a, https://docs.spring.io/spring-boot/index.html (siječanj 2025.)
7. Postman, https://community.postman.com/t/enter-authorization-token-to-sign-into-postman/52539 (siječanj 2025.)
8. Baze podataka, FER ZPR, https://www.fer.unizg.hr/predmet/bazpod
9. React dokumentacija, https://react.dev/ (siječanj 2025.)
10. PostgreSQL dokumentacija, https://www.postgresql.org/ (siječanj 2025.)
11. Git dokumetacija, https://github.com/git-guides (siječanj 2025.)
12. ChatGPT, https://chatgpt.com/

Pages 13

Find a page…

- Home
- 1. Opis projektnog zadatka
- 2. Analiza zahtjeva
- 3. Specifikacija zahtjeva sustava
- 4. Arhitektura i dizajn sustava
- 5. Arhitektura komponenata i razmj…
- 6. Ispitivanje programskog rješenja
- 7. Tehnologije za implementaciju a…
- 8. Upute za puštanje u pogon
- 9. Zaključak i budući rad
- A. Dnevnik promjena dokumentacije
- B. Popis literature
- C. Prikaz aktivnosti grupe

Clone this wiki locally

https://github.com/mihovil13/Ling

© 2025 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

C. Prikaz aktivnosti grupe

Mia Zekić edited this page 14 minutes ago · 2 revisions

Dnevnik sastajanja

1. Sastanak

- Datum: 15. listopada 2024.
- Prisustvovali: A. Stajku, A. Šinjori, B. Svjetličić, D. Jurgec, M. Mađerić, M. Zekić, Z. Martinović
- Teme sastanka:

> - Inicijalna podjela uloga
> - Odabir tehnologija
> - Dogovor načina komunikacije

2. Sastanak

- Datum: 29. listopada 2024.
- Prisustvovali: A. Stajku, A. Šinjori, B. Svjetličić, D. Jurgec, M. Mađerić, M. Zekić, Z. Martinović
- Teme sastanka:

> - Podjela poslova vezanih uz dokumentaciju, dijagrame i obrasce uporabe
> - Modeliranje strukture baze
> - Diskutiranje i osmišljanje pitanja za iduću laboratorijsku vježbu

3. Sastanak

- Datum: 12. studenog 2024.
- Prisustvovali: A. Stajku, A. Šinjori, B. Svjetličić, D. Jurgec, M. Mađerić, M. Zekić, Z. Martinović
- Teme sastanka:

> - Podjela oko završnih detalja prije prve predaje projekta
> - Diskutiranje o postojećim nejasnoćama i pitanjima
> - Rekapitulacija dosad izrađenog dijela projekta i predlaganje izmjena

4. Sastanak

- Datum: 12. prosinca 2024.
- Prisustvovali: A. Stajku, A. Šinjori, B. Svjetličić, D. Jurgec, M. Mađerić, M. Zekić, Z. Martinović
- Teme sastanka:

> - Podjela zadataka drugog ciklusa
> - Rasprava o daljnjim zadacima

5. Sastanak

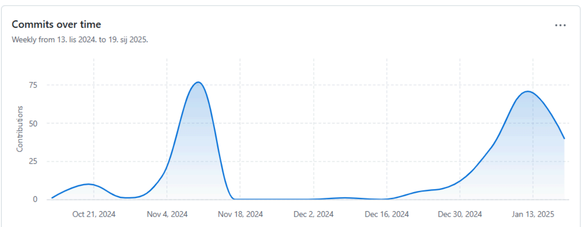
- Datum: 7. siječnja 2025.
- Prisustvovali: A. Stajku, A. Šinjori, B. Svjetličić, D. Jurgec, M. Mađerić, M. Zekić
- Teme sastanka:

> - Podjela zadataka
> - Rasprava o daljnjim zadacima

Tablica aktivnosti

	Dario Jurgec	Mihovil Mađerić	Zita Martinović	Adi Stajku	Borna Svjetličić	Adam Šinjori	Mia Zekić
Upravljanje projektom	1	1	1	1	1	1	1
Opis projektnog zadatka	1		1				2
Funkcionalni zahtjevi	0.5						4
Opis pojedinih obrazaca	2	1	2				
Dijagram obrazaca			3				
Sekvencijski dijagrami	3						
Opis ostalih zahtjeva	0.5						2
Arhitektura i dizajn sustava							5
Baza podataka						2	2
Dijagram razreda	2						0.5
Dijagram stanja							7
Dijagram aktivnosti							7
Dijagram komponenti							5
Korištene tehnologije i alati							0.1
Ispitivanje programskog rješenja							12
Dijagram razmještaja							7
Upute za puštanje u pogon							3
Dnevnik sastajanja		1					1
Zaključak i budući rad							0.5
Popis literature							1
Učenje novih tehnologija	20	20	20	20	20	20	20
Izrada korisničkog sučelja	5		35	20	25		
Povezivanje s bazom		10				20	10
Izrada backend	7	35				35	7
Izrada frontend	40			20	10		
Postavljanje na server		15			2		

Dijagram pregleda promjena



Ključni izazovi i rješenja

Ključni izazovi tijekom rada u grupi bili su kašnjenje u razvoju i tehnički problemi. Tijekom prvog ciklusa cijela je grupa odlučila kako ranije moramo krenuti u izradu određenih komponenti. Tijekom drugog ciklusa nije bilo većih kašnjenja u razvoju. Najveći tehnički problemi bili su riješeni uz zajedničku pomoć gdje je posebno bila važna pravovremena komunikacija između osoba na Frontend i Backend dijelu.

Pages 13
Find a page...
Home
1. Opis projektnog zadatka
2. Analiza zahtjeva
3. Specifikacija zahtjeva sustava
4. Arhitektura i dizajn sustava
5. Arhitektura komponenata i razmj...
6. Ispitivanje programskog rješenja
7. Tehnologije za implementaciju a...
8. Upute za puštanje u pogon
9. Zaključak i budući rad
A. Dnevnik promjena dokumentacije
B. Popis literature
C. Prikaz aktivnosti grupe
Dnevnik sastajanja
1. Sastanak
2. Sastanak
3. Sastanak
4. Sastanak
5. Sastanak
Tablica aktivnosti
Dijagram pregleda promjena
Ključni izazovi i rješenja

Clone this wiki locally
https://github.com/mihovil13/Ling