pitajStrucnjaka

Web aplikacija za konsultacije sa stručnjacima

Arhitekturni projekat

Verzija 1.3

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 09.04.2024. | 1.0 | Inicijalna verzija | Anđela Mia |
| 10.04.2024. | 1.1 | Sve osim poglavlja 8 i 10 | Aleksa |
| 11.04.2024. | 1.2 | Završetak 10, konsultacija sa profesorom | Anđela Mia, Filip, Mateja |
| 24.04.2024. | 1.3 | Završeno sve osim stavke 10.1 i 10.2 | Anđela Mia, Filip |
| 27.04.2024. | 1.4 | Finalna verzija, odradjen model domena i sema baze podataka | Aleksa, Ilija, Filip |

Sadržaj

1. Cilj dokumenta 5

2. Opseg dokumenta 5

3. Reference 5

4. Predstavljanje arhitekture 5

5. Ciljevi i ograničenja arhitekture 5

6. Pogled na slučajeve korišćenja 6

6.1 Dijagrami slučajeva korišćenja 7

6.2 Kratak opis slučajeva korišćenja 8

6.2.1 Registrovanje 8

6.2.2 Prijavljivanje 8

6.2.3 Pregled osnovnih informacija 8

6.2.4 Pregled oglasa stručnjaka 9

6.2.5 Pregled oglasa korisnika 9

6.2.6 Naslovna stranica (čemu je namenjena web aplikacija) 9

6.2.7 Pregled nekih korisničkih recenzija 9

6.2.8 Kreiranje klijentskih oglasa 9

6.2.9 Kreiranje formi za praćenje konsultacija 9

6.2.10 Pretraga oglasa stručnjaka preko filtera 9

6.2.11 Kreiranje oglasa stručnjaka 9

6.2.12 Pretraživanje klijentskih oglasa (putem tagova) 9

6.2.13 Ocenjivanje 9

6.2.14 Zakazivanje konsultacija 10

6.2.15 Komunikacija preko video poziva i slanjem poruka 10

6.2.16 Pružanje korisničke pomoći 10

6.2.17 FAQ (često postavljanja pitanja) 10

6.2.18 Sistem za podršku putem tiketa 10

6.2.19 Opcija za direktnu komunikaciju sa administratorom 10

6.2.20 Kredit na aplikaciji za plaćanje 10

6.2.21 Pregled trenutnog stanja na računu 10

6.2.22 Transparentni prikaz troškova i transakcija 10

6.2.23 Sigurnost pri transakcijima povraćajem novca ukoliko dođe do problema 11

6.2.24 Upravljanje podacima 11

6.2.25 Održavanje 11

7. Pogled na logičku arhitekturu sistema 11

7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 12

7.1.1 Korisnički interfejs 12

7.1.2 Aplikaciona logika 12

7.1.3 Pristup podacima 12

7.1.4 HTML 12

7.1.5 CSS/Bootstrap 5 12

7.1.6 JavaScript i TypeScript 13

7.1.7 Angular 13

7.1.8 NestJS 13

7.1.9 MySQL MySQL je popularni relacioni sistem za upravljanje bazama podataka koji se koristi za čuvanje i manipulaciju podacima u struktuiranom formatu. 13

7.1.10 WebSockets 13

8. Pogled na procese 13

8.1 Procesi 14

8.1.1 Web čitač 14

8.1.2 Web server (NGINX Reverse Proxy) 14

8.1.3 NodeJS (ExpressJS) 14

8.1.4 MySQL Server 15

9. Pogled na raspoređivanje sistema 15

*9.1* *Klijent* 15

*9.2* *Web server (NGINX Reverse Proxy)* 15

*9.3* *Aplikacioni server (ExpressJS)* 15

*9.4* *RDBMS server* 15

10. Pogled na implementaciju sistema 16

*10.1* *Model domena* 16

*10.2* *Šema baze podataka* 17

*10.3* *Komponente sistema* 17

10.3.1 Komponente korisničkog interfejsa 17

10.3.2 Komponente aplikacione logike 19

10.3.3 Komponente za pristup podacima 19

11. Performanse 20

12. Kvalitet 20

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture web aplikacije pitajStrucnjaka.

# Opseg dokumenta

Dokument se odnosi na web aplikaciju pitajStrucnjaka koja će biti razvijena od strane tima Stručnjaci. Namena web aplikacije pitajStrucnjaka je omogućavanje brze komunikacije između klijenta i određenog stručnjaka radi pružanja usluga i rešavanja problema, čime se olakšava korisnicima da dobiju potrebnu pomoć i savete direktno putem online interakcije.

# Reference

Spisak korišćene literature:

1. pitajStručnjaka – D01\_Predlog\_Projekta.docx, Stručnjaci-pitajStrucnjaka-01, V1.1, 2024, Stručnjaci.
2. pitajStručnjaka – D02\_Vizija\_Sistema.docx, Stručnjaci-pitajStrucnjaka-02, V1.2, 2024, Stručnjaci.
3. pitajStručnjaka – D03\_Plan\_realizacije\_projekta.docx, Stručnjaci-pitajStrucnjaka-03, V1.1, 2024, Stručnjaci.
4. pitajStručnjaka – D03\_Raspored\_Aktivnosti.pod, Stručnjaci-pitajStrucnjaka-03, V1.0, 2024, Stručnjaci.
5. pitajStručnjaka – D04\_Spec\_Zahteva.docx, Stručnjaci-pitajStrucnjaka-04, V1.2, 2024, Stručnjaci.

# Predstavljanje arhitekture

Arhitektura sistema u dokumentu je prikazana kao serija pogleda na sistem: pogled na slučajeve korišćenja, pogled na logičku arhitekturu sistema, pogled na procese, pogled na razmeštaj komponenti sistema i pogled na implementaciju. Ovi pogledi su predstavljeni odgovarajućim UML dijagramima.

# Ciljevi i ograničenja arhitekture

Ključni zahtevi i sistemska ograničenja koja imaju značajan uticaj na izbor arhitekture i projektovanje sistema su:

1. pitajStručnjaka Web aplikacija će biti implementirana kao Web aplikacija zasnovana na Angular framework-u, Bootstrap i PrimeNG tehnologijama na klijentskoj strani, NestJS, TypeScript, WebSockets (Socket.io), PassportJS, MySQL baza podataka i korišćenjem TypeORM alata na serverskoj strani [4].
2. Klijentski deo web aplikacije pitajStrucnjaka će biti optimizovan za sve najnovije web browser-e, uključujući Google Chrome, Mozilla Firefox, Apple Safari, Microsoft Edge, Brave i Opera, pružajući korisnicima široku dostupnost i pristupačnost [4].
3. Svi zahtevi u pogledu performansi dati u [5] moraju biti uzeti u obzir pri izboru arhitekture i razvoju sistema.

# Pogled na slučajeve korišćenja

U ovom odeljku je dat pogled na slučajeve korišćenja definisane u specifikaciji zahteva [5].

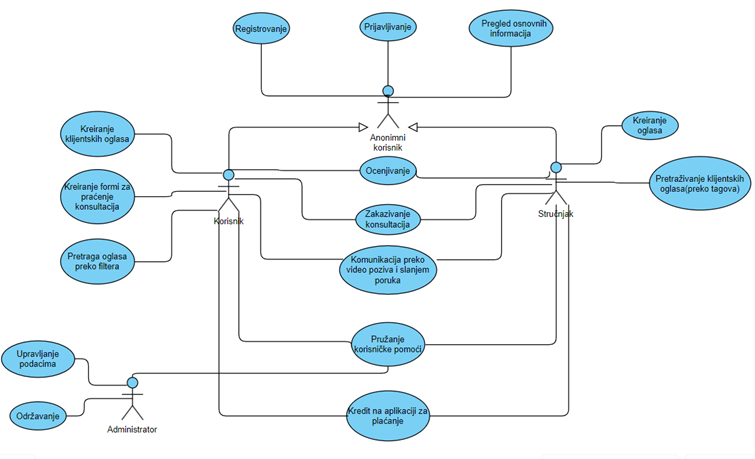
Slučajevi korišćenja pitajStrucnjaka web aplikacije su:

* Registrovanje
* Prijavljivanje
* Pregled osnovnih informacija
  + Pregled oglasa stručnjaka
  + Pregled oglasa korisnika
  + Naslovna stranica (čemu je namenjena web aplikacija)
  + Pregled nekih korisničkih recenzija
* Kreiranje klijentskih oglasa
* Kreiranje formi za praćenje konsultacija
* Pretraga oglasa stručnjaka preko filtera
* Kreiranje oglasa stručnjaka
* Pretraživanje klijentskih oglasa (putem tagova)
* Ocenjivanje
* Zakazivanje konsultacija
* Komunikacija preko video poziva i slanjem poruka
* Pružanje korisničke pomoći
  + FAQ (često postavljanja pitanja)
  + Sistem za podršku putem tiketa
  + Opcija za direktnu komunikaciju sa administratorom
* Kredit na aplikaciji za plaćanje
  + Pregled trenutnog stanja na računu
  + Transparentni prikaz troškova i transakcija
  + Sigurnost pri transakcijama povraćajem novca ukoliko dođe do problema
* Upravljanje podacima
* Održavanje

Ove slučajevi korišćenja mogu da iniciraju administrator, anonimni korisnik, korisnik, stručnjak.

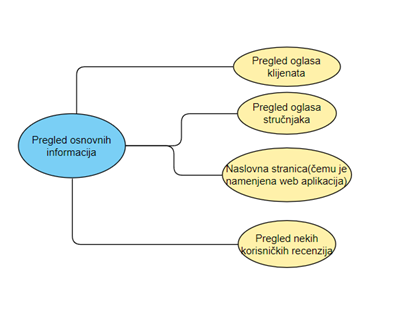
## Dijagrami slučajeva korišćenja

Osnovni UML dijagram koji prikazuje korisnike i slučajeve korišćenja web aplikacije pitajStrucnjaka prikazan je na sledećoj slici:

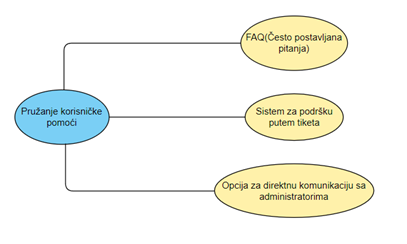


Slučajevi korišćenja *pregled informacija* i *kreiranje, brisanje i arhiviranje članova* i *ažuriranje podataka o publikacijama* obuhvataju složenije radnje koje se mogu razložiti dalje razložiti na pojedinačne slučajeve korišćenja.

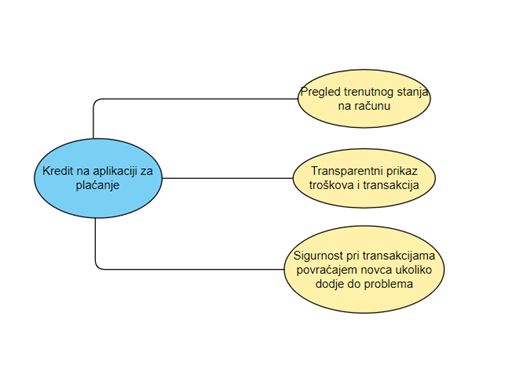
Detaljni UML dijagram za slučaj korišćenja *pregled osnovnih informacija* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *pružanje korisničke pomoći* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj *kredit na aplikaciji za plaćanje* je prikazan na sledećoj slici:



## Kratak opis slučajeva korišćenja

### Registrovanje

Kratak opis: Korisnik može prilikom registrovanja da izabere između 2 tipa naloga, korisnik i stručnjak. Ukoliko se izabere opcija stručnjak, prilikom prijavljivanja će biti zahtevana dodatna verifikacija.

Akteri koji iniciraju slučaj korišćenja: Anonimni korisnik.

### Prijavljivanje

Kratak opis: Unos svih zahtevanih podataka za prijavljivanje na nalog.

Akteri koji iniciraju slučaj korišćenja: Anonimni korisnik.

### Pregled osnovnih informacija

Kratak opis: Prikaz profila korisnika sa njihovim informacijama i podacima.

Akteri koji iniciraju slučaj korišćenja: Administrator, Korisnik, Stručnjak, Anonimni korisnik.

### Pregled oglasa stručnjaka

Kratak opis: Korisnici mogu da pretražuju i gledaju sve postavljenje oglase stručnjaka na web aplikaciji.

Akteri koji iniciraju slučaj korišćenja: Administrator, Anonimni korisnik, Stručnjak, Korisnik.

### Pregled oglasa korisnika

Kratak opis: Stručnjaci mogu da pretražuju i gledaju sve postavljenje oglase korisnika na web aplikaciji.

Akteri koji iniciraju slučaj korišćenja: Administrator, Anonimni korisnik, Stručnjak, Korisnik.

### Naslovna stranica (čemu je namenjena web aplikacija)

Kratak opis: Služi za prikaz osnovne namene web aplikacije korisnicima.

Akteri koji iniciraju slučaj korišćenja: Anonimni korisnik, korisnik, stručnjak, administrator.

### Pregled nekih korisničkih recenzija

Kratak opis: Stranica gde korisnici mogu videti ocene vlasnika oglasa.

Akteri koji iniciraju slučaj korišćenja: Anonimni korisnik, Korisnik, Stručnjak, Administrator.

### Kreiranje klijentskih oglasa

Kratak opis: Klijent kreira oglas na osnovu potrebe koju potražuje.

Akteri koji iniciraju slučaj korišćenja: Klijent, Administrator.

### Kreiranje formi za praćenje konsultacija

Kratak opis: Klijent pre početka konsultacije kreira svoju formu koja predstavlja zahteve koje odabrani stručnjak mora da ispuni tokom konsultacija.

Akteri koji iniciraju slučaj korišćenja: Korisnik, Stručnjak.

### Pretraga oglasa stručnjaka preko filtera

Kratak opis: Omogućava klijentu da brzo i lako pretražuje oglase stručnjaka preko filtriranja.

Akteri koji iniciraju slučaj korišćenja: Korisnici, Administratori.

### Kreiranje oglasa stručnjaka

Kratak opis: Stručnjak kreira oglas u kome navodi sve usluge koje pruža.

Akteri koji iniciraju slučaj korišćenja: Stručnjak, Administrator.

### Pretraživanje klijentskih oglasa (putem tagova)

Kratak opis: Stručnjaci odabirom tagova mogu da olakšaju sebi pretragu oglasa klijenata tako što će im se prikazivati samo oglasi za koje su stručni. Dodatno, stručnjaci će dobijati notifikacije, ukoliko omoguće tu opciju, za oglase koji odgovaraju veštinama koje stručnjak poseduje.

Akteri koji iniciraju slučaj korišćenja: Administrator, Stručnjak.

### Ocenjivanje

Kratak opis: Nakon završene komunikacije moguće je međusobno ocenjivanje stručnjaka i korisnika, koji su imali interakciju.

Akteri koji iniciraju slučaj korišćenja: Administrator, Korisnik, Stručnjak.

### Zakazivanje konsultacija

Kratak opis: Omogućiće brzo i jednostavno zakazivanje konsultacija između korisnika i stručnjaka, preko formi, video četa i četa.

Akteri koji iniciraju slučaj korišćenja: Stručnjak, Klijent.

### Komunikacija preko video poziva i slanjem poruka

Kratak opis: Omogućena komunikacija preko četa u svrhu dogovora o konsultacijama, same konsultacije se održavaju preko video poziva.

Akteri koji iniciraju slučaj korišćenja: Stručnjak, Korisnik.

### Pružanje korisničke pomoći

Kratak opis: U svakom trenutku ako klijent naiđe na problem, aplikacija podržava pružanje pomoći na različite načine.

Akteri koji iniciraju slučaj korišćenja: Administrator, Korisnik, Stručnjak, Anonimni Korisnik.

### FAQ (često postavljanja pitanja)

Kratak opis: Stranica gde će korisnici moći da vide odgovore na cesto postavljena pitanja.

Akteri koji iniciraju slučaj korišćenja: Korisnik, Stručnjak, Anonimni Korisnik.

### Sistem za podršku putem tiketa

Kratak opis: Stranica gde korisnik može da otvori novi tiket gde će biti poslato administratoru u cilju pružanja korisničke podrške.

Akteri koji iniciraju slučaj korišćenja: Stručnjak, Korisnik, Administrator.

### Opcija za direktnu komunikaciju sa administratorom

Kratak opis: Svaki prijavljeni korisnik, nebitno da li je klijent ili stručnjak može imati direktnu komunikaciju sa administratorom u slučaju problema sa oglasom ili sajtom.

Akteri koji iniciraju slučaj korišćenja: Stručnjak, Korisnik, Administrator.

### Kredit na aplikaciji za plaćanje

Kratak opis: Na web aplikaciji postoji kredit koji se kupuje i sa kojim se plaćaju sve transakcije na aplikaciji.

Akteri koji iniciraju slučaj korišćenja: Administrator, Korisnik, Stručnjak.

### Pregled trenutnog stanja na računu

Kratak opis: Proveravanje količine kredita koju trenutno imaju korisnici i stručnjaci.

Akteri koji iniciraju slučaj korišćenja: Korisnik, Stručnjak.

### Transparentni prikaz troškova i transakcija

Kratak opis: Jasan prikaz istorije troškova i transakcija koje su obavljene na web aplikaciji.

Akteri koji iniciraju slučaj korišćenja: Korisnik, Stručnjak.

### Sigurnost pri transakcijima povraćajem novca ukoliko dođe do problema

Kratak opis: Mogućnost povratka novca u slučaju greške pri transakciji.

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Upravljanje podacima

Kratak opis: Mogućnost upravljanja svim bitnim informacijama od strane administratora.

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Održavanje

Kratak opis: Prostor za unapređivanje web aplikacije komunikacijom korisnika i administratora.

Akteri koji iniciraju slučaj korišćenja: Administrator, Korisnik, Stručnjak.

# Pogled na logičku arhitekturu sistema

U ovom odeljku je dat pregled logičke arhitekture sistema. Ovaj pogled sadrži opis najznačajnijih klasa, njihove organizacije u pakete i podsisteme, i organizacija podsistema u slojeve. U cilju opisivanja dinamičkih aspekata arhitekture, ovaj odeljak može da uključi opise realizacije najznačajnijih slučajeva korišćenja. Da bi se ilustrovala veza između arhitekturno značajnih klasa, podsistema, paketa ili slojeva moguće je uključiti i odgovarajuće dijagrame klasa.

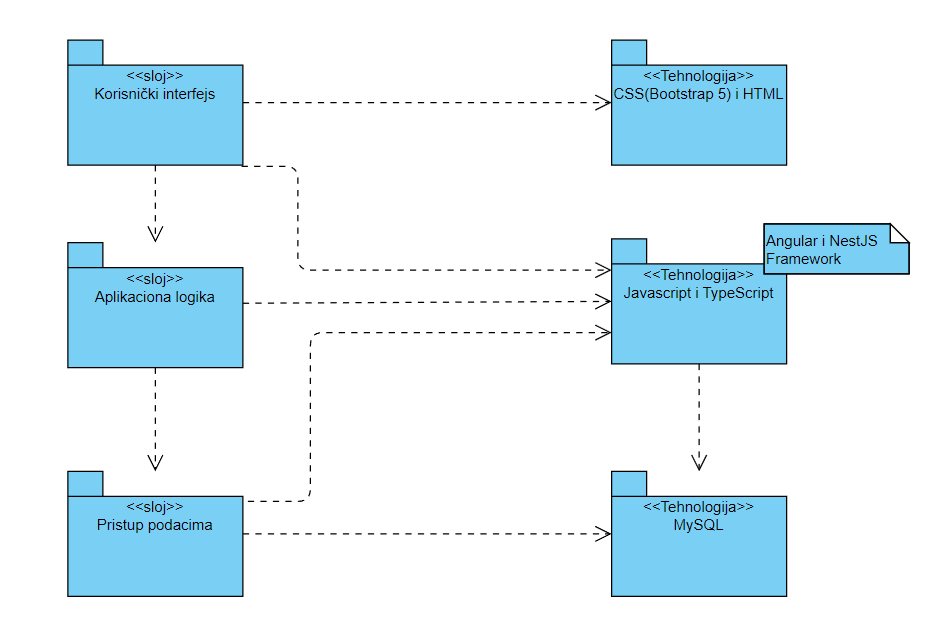
Logički pogled na PitajStrucnjaka aplikacije obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket Korisnički interfejs sadrži Web stranice čiji je sadržaj prikazan pomoću HTML-a, za stilizaciju CSS-a / Bootstrap 5, i multimedijalni sadržaj koji ostvaruju grafički dizajn i forme preko kojih korisnici sistema komuniciraju sa sistemom.

Paket Aplikaciona logika predstavlja srednji sloj sistema koji sadrži JavaScript i TypeScript kod zadužen za ostvarivanje funkcionalnosti specifičnih za domen sistema koji se razvija, koristeći Angular i NestJS framework.

Paket Pristup podacima sadrži skripte koje predstavljaju interfejs za pristup, dodavanje i ažuriranje podataka koji se čuvaju u bazi podataka MySQL, uz podršku TypeORM-a.

## Pregled arhitekture – organizacija paketa i podsistema u slojeve



### Korisnički interfejs

Ovaj sloj implementira korisnički interfejs portala. U njemu su sadržani svi HTML elementi, multimedijalni sadržaji, kao i CSS/Bootstrap 5 stilizacija koja oblikuje HTML stranice i omogućava interaktivnost korisnika sa sistemom.

Sloj korisničkog interfejsa zavisi od sloja aplikacione logike, kao i tehnologija HTML, CSS/Bootstrap 5 i JavaScript/TypeScript.

### Aplikaciona logika

Sloj aplikacione logike predstavlja središnji sloj u arhitekturi pitajStrucnjaka portala. Ovde se nalazi JavaScript/TypeScript kod koji implementira funkcionalnosti specifične za domen primene portala, uspostavljajući vezu između korisničkog interfejsa i sloja za pristup podacima.

Ovaj sloj zavisi od sloja za pristup podacima, kao i od tehnologija koje koristite za razvoj frontenda, poput Angular framework-a i drugih povezanih tehnologija (npr. NestJS za backend).

### Pristup podacima

Sloj za pristup podacima se nalazi na dnu troslojne arhitekture i sadrži skripte zadužene za pribavljanje, dodavanje i ažuriranje podataka koji se čuvaju u MySQL bazi podataka.

Ovaj sloj nije direktno zavisan od drugih slojeva, ali je zavisan od tehnologija koje koristite, uključujući MySQL bazu podataka i TypeORM maper modela na same tabele MySQL baze.

### HTML

Tehnologija HTML-a definiše gradivne elemente stranica koje se prikazuju u Web čitaču i koje omogućavaju prikaz formatiranih informacija i realizaciju formi za unos i ažuriranje podataka.

### CSS/Bootstrap 5

CSS (Cascading Style Sheets) / Bootstrap 5 pruža stilizaciju i formatiranje elemenata HTML-a, omogućavajući kreiranje privlačnog i prilagodljivog dizajna stranica.

### JavaScript i TypeScript

JavaScript i TypeScript pružaju funkcionalnost za interaktivnost na stranicama, omogućavajući dinamičko ažuriranje sadržaja i manipulaciju elementima na stranici.

### Angular

Angular je frontend framework koji olakšava razvoj kompleksnih web aplikacija kroz upravljanje komponentama, rutama i upravljanjem stanja aplikacije.

### NestJS

NestJS pruža backend framework zasnovan na Node.js-u za izgradnju skalabilnih i modularnih server aplikacija.

### MySQL MySQL je popularni relacioni sistem za upravljanje bazama podataka koji se koristi za čuvanje i manipulaciju podacima u struktuiranom formatu.

### WebSockets

WebSockets (Socket.io) omogućava stvaranje dvosmernih komunikacijskih kanala između klijenta i servera putem internetskog protokola, što omogućava brzu i efikasnu razmenu podataka u realnom vremenu. Spada u deo aplikacione logike.

# Pogled na procese

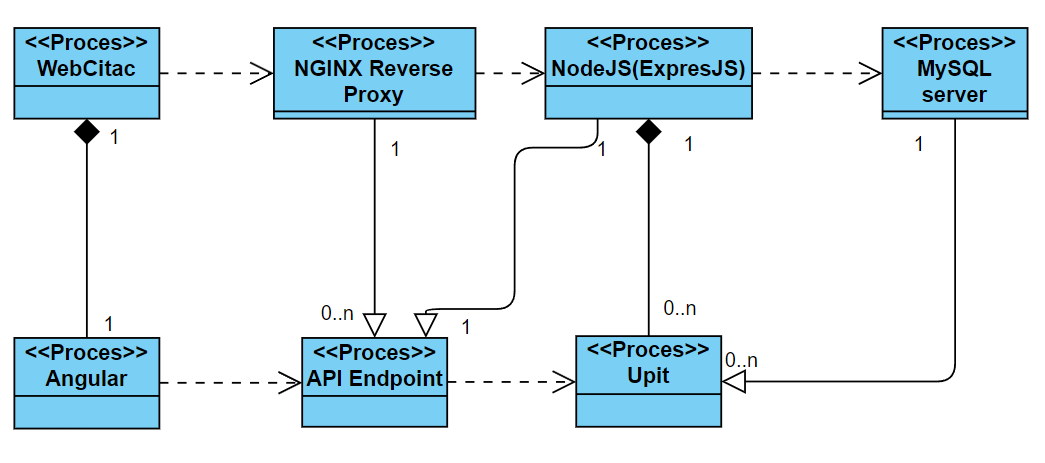
U ovom odeljku je sadržan pogled na procesnu arhitekturu sistema. Ovaj opis treba da sadrži specifikaciju različitih zadataka (procesa i niti) uključenih u rad sistema. Takođe je potrebno dati dijagrame koji pokazuju njihovu interakciju i konfiguraciju. Dodela objekata i klasa na određene zadatke takođe spada u opis procesne arhitekture.

Web aplikacije zasnovane na NodeJS (NestJS framework) imaju fleksibilan procesni model koji omogućava efikasno upravljanje resursima. Za razliku od drugih tehnologija, kod NodeJS aplikacija dizajneri moraju uzeti u obzir aspekte kao što su asinhrono izvršavanje i upravljanje događajima. Međutim, ovo pruža veću kontrolu nad performansama i skalabilnošću aplikacije Sa stanovišta projektanta.

Node.js Web aplikacije zahtevaju pažljivo vođenje računa o načinu rada Web servera i načinu izvršavanja skripti. Za razliku od drugih tehnologija, kod NodeJS aplikacija dizajneri su odgovorni za optimizaciju rada Web servera i efikasno upravljanje izvršavanjem skripti radi postizanja boljih performansi.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju pitajStrucnjaka web aplikacije. Dijagram je opšteg tipa i može se primeniti na bilo koju Web aplikaciju zasnovanu na NodeJS-u (NestJS framework koji koristi ExpressJS) i MySQL bazi podataka.



### Web čitač

Web čitač je proces koji izvršava funkcionalnost aplikacije za prikaz HTML stranica koje generiše Angular. Umesto da direktno prikazuje statički HTML dobijen od Web servera, Angular aplikacija generiše i dinamički prikazuje HTML, CSS i JavaScript na osnovu definisanih komponenata, direktiva i logike aplikacije.

U najopštijem slučaju Web čitač u jednom trenutku može da prikazuje samo jednu HTML stranicu.

### Web server (NGINX Reverse Proxy)

Web server je proces koji izvršava funkcionalnost opsluživanja zahteva prispelih sa više Web čitača. Web server koji će se koristiti u projektu je NGINX, tj. NGINX Reverse Proxy. Kada NGINX prosleđuje zahtev, on šalje zahtev određenom serveru na koji se prosleđuje, dobija odgovor i šalje ga nazad klijentu. Moguće je prosleđivati zahteve ka HTTP serveru (drugi NGINX server ili bilo kom drugom serveru) ili ka serveru koji nije HTTP (koji može izvršavati aplikaciju razvijenu u određenom okviru).

U našem slučaju NGINX će prosleđivati zahtev ka NodeJS (ExpressJS) serveru, gde će taj server dalje obrađivati pristigle zahteve.

### NodeJS (ExpressJS)

ExpressJS je aplikacioni server koji može da obrađuje zahteve samostalno ili zajedno sa web serverom kao što su NGINX, Apache HTTP server ili Microsoft IIS. U našoj web aplikaciji, ExpressJS će raditi zajedno sa NGINX web serverom. Kada korisnici šalju zahteve, NGINX web server prima te zahteve i prosleđuje ih ExpressJS serveru. Zatim ExpressJS obrađuje te zahteve, generiše odgovor i šalje nazad NGINX serveru, koji potom šalje odgovor korisniku. Ova kombinacija omogućava brzu i efikasnu obradu zahteva korisnika i omogućava skalabilnost i bolju kontrolu nad web aplikacijom.

MySQL bazi podataka će biti pristupano pomoću TypeORM alata za efikasno upravljanje podacima u našoj web aplikaciji.

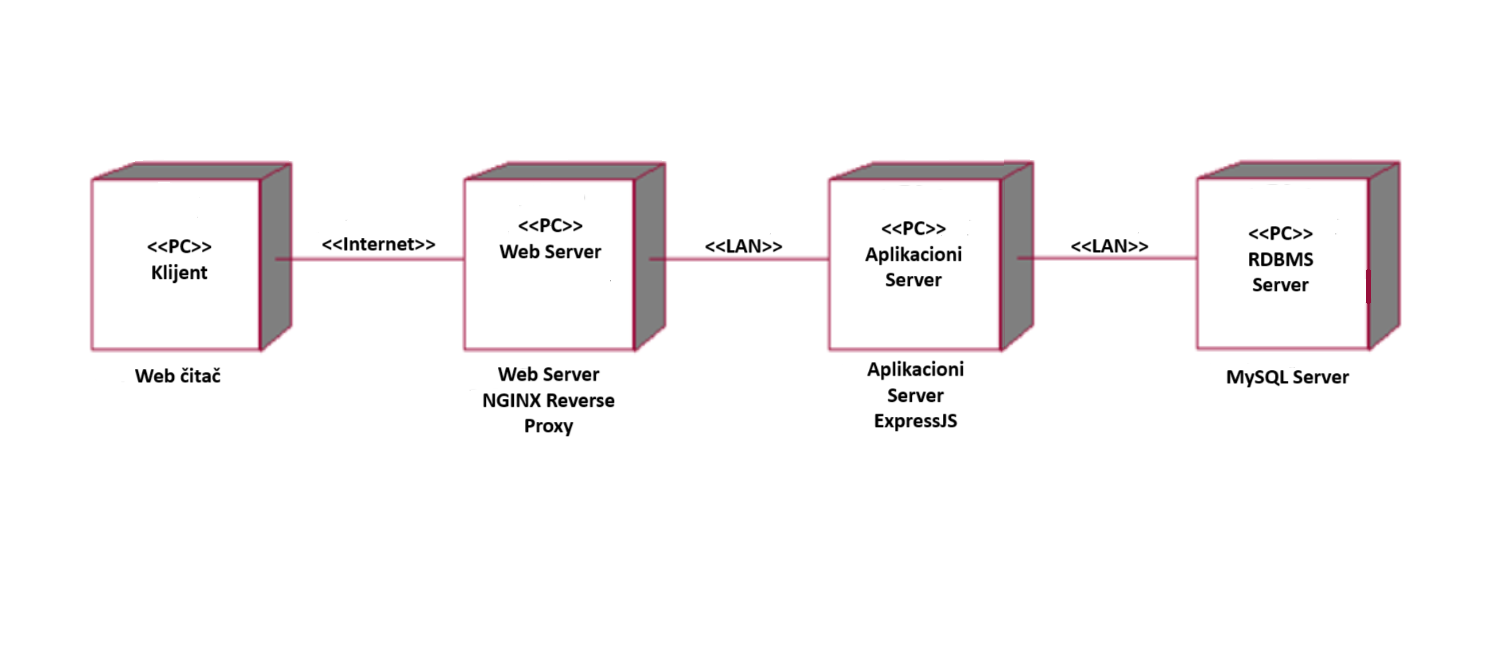
### MySQL Server

MySQL Server predstavlja proces koji obavlja osnovne zadatke MySQL sistema za upravljanje bazama podataka. Ovaj proces je u mogućnosti da istovremeno obrađuje više upita, izvršava ih nad bazom podataka i vraća rezultate korisnicima koji su te upite poslali.

# Pogled na raspoređivanje sistema

Pogled na raspoređivanje sistema prikazuje različite fizičke čvorove za najopštiju konfiguraciju sistema. Fizičkim čvorovima koji predstavljaju procesore vrši se dodeljivanje identifikovanih procesa.

Na sledećoj slici dat je UML dijagram raspoređivanja pitajStrucnjaka web aplikacije.



## *Klijent*

Korisnici pristupaju web aplikaciji pitajStrucnjaka putem svojih računara koji su opremljeni web browser-ima. Korišćenjem internetske infrastrukture, klijenti i stručnjaci se povezuju sa web serverom bez obzira na njihovu geografsku lokaciju, što omogućava slobodan pristup bez ikakvih ograničenja.

## *Web server (NGINX Reverse Proxy)*

Na udaljenoj mašini, koja služi kao Web server, višestruki klijenti pristupaju preko Interneta. Web server prosleđuje zahteve aplikacionom serveru, koji obrađuje te zahteve i vraća web serveru koji dalje prosleđuje klijentu.

## *Aplikacioni server (ExpressJS)*

Aplikacioni server prima zahteve od web servera, izvršava biznis logiku tj. obrađuje zahtev, validaciju podataka, generisanje HTML ili JSON-a, pristup bazi podataka, itd...Nakon toga obrađeni zahtev se šalje nazad web serveru.

## *RDBMS server*

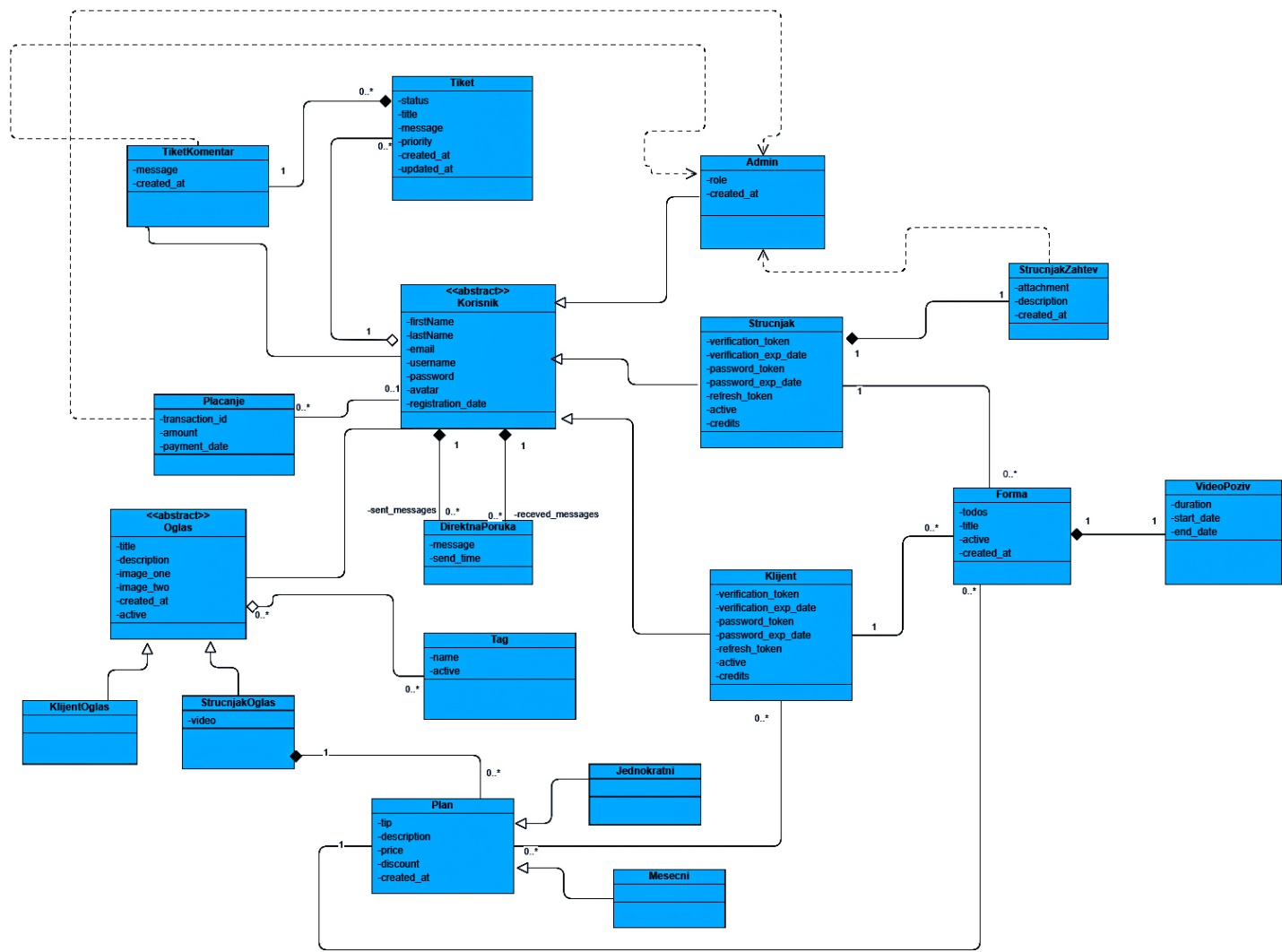
Na računaru koji služi kao RDBMS server, MySQL Server proces je odgovoran za izvršavanje funkcionalnosti sistema za upravljanje bazama podataka. Pristup bazi podataka je ograničen samo na računare unutar lokalne mreže (LAN), što je učinjeno radi osiguranja sigurnosti podataka pohranjenih na tom računaru.

# Pogled na implementaciju sistema

Pogled na implementaciju prikazuje različite aspekte bitne za implementaciju sistema. U slučaju pitajStrucnjaka web aplikacije ovaj odeljak sadrži model domena, šemu baze podataka i prikaz komponenti sistema razvrstanih u ranije identifikovane pakete.

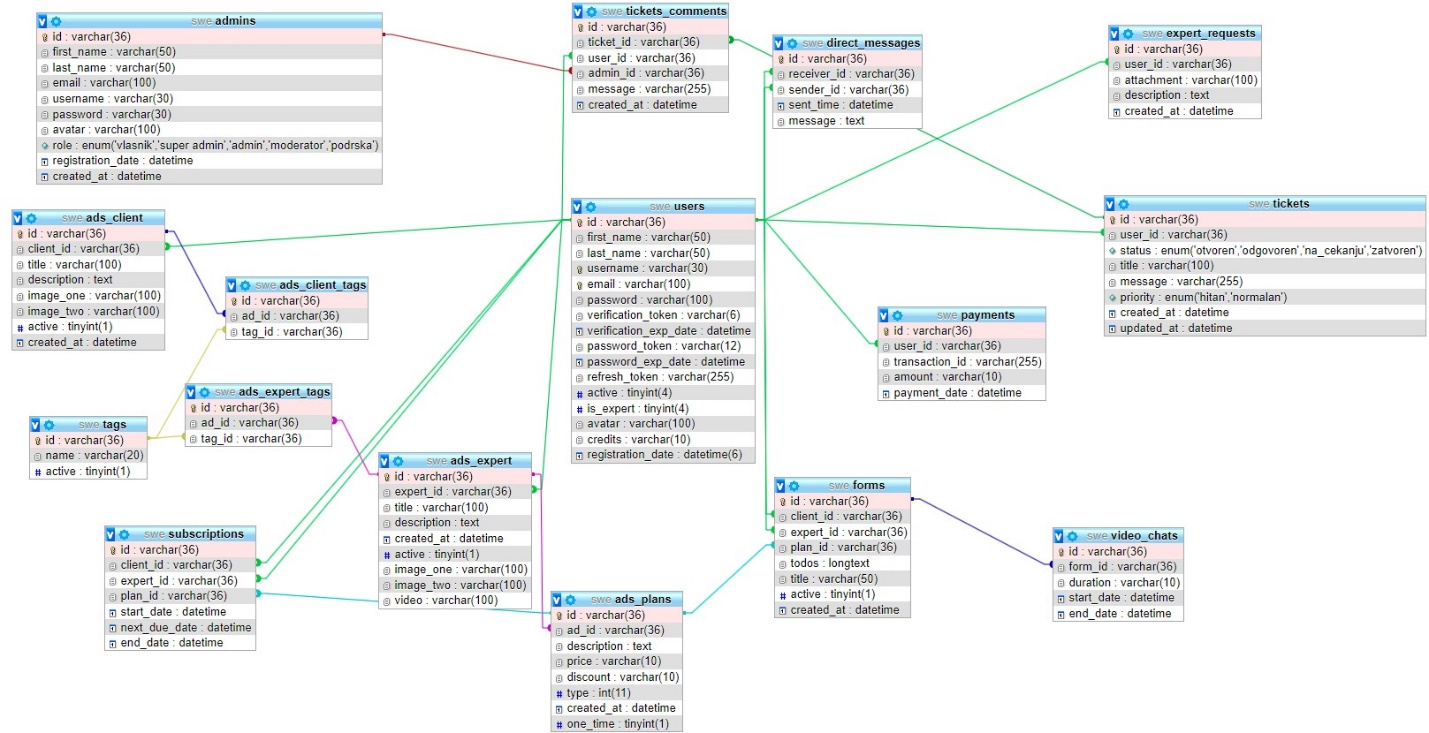
## *Model domena*

Odeljak o implementaciji pruža pregled različitih ključnih elemenata potrebnih za realizaciju sistema. U slučaju web aplikacije "Pitaj Stručnjaka", ovaj segment obuhvata domenski model, strukturu baze podataka i pregled komponenti sistema grupisanih prema prethodno identifikovanim paketima.



## *Šema baze podataka*

Detaljna šema baze podataka je prikazana na sledećem dijagramu. Baza podataka i dijagram su kreirani korišćenjem *phpMyAdmin-a*.



## *Komponente sistema*

Komponente sistema web aplikacije pitajStrucnjaka su predstavljene kao moduli i kontroleri u NestJS aplikaciji, organizovani prema arhitekturi servisa i repozitorijuma. Za ilustraciju će biti korišćeni UML dijagrami komponenti, ali i dijagrami klasa. Atributi ovih klasa predstavljaju ulazne podatke koji se dobijaju putem HTTP zahteva, dok metodi predstavljaju funkcije definisane u okviru odgovarajućih servisa i kontrolera.

### Komponente korisničkog interfejsa

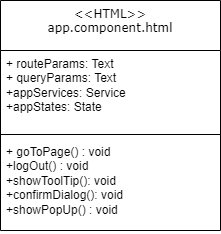
Dizajn korisničkog interfejsa je obuhvaćen dvema komponentama:



Komponenta **app.component.html** implementira stranicu web aplikacije pitajStrucnjaka čiji sadržaj može da varira u zavisnosti od putanje i parametra u web browser-u.

Komponenta **styles.css** predstavlja opis stilova za pojedine HTML elemente koji se javljaju na različitim stranicamaSS.

Parametri koji utiču na izbor i jezik za prikaz stranice ilustrovani su sledećim dijagramom klasa:

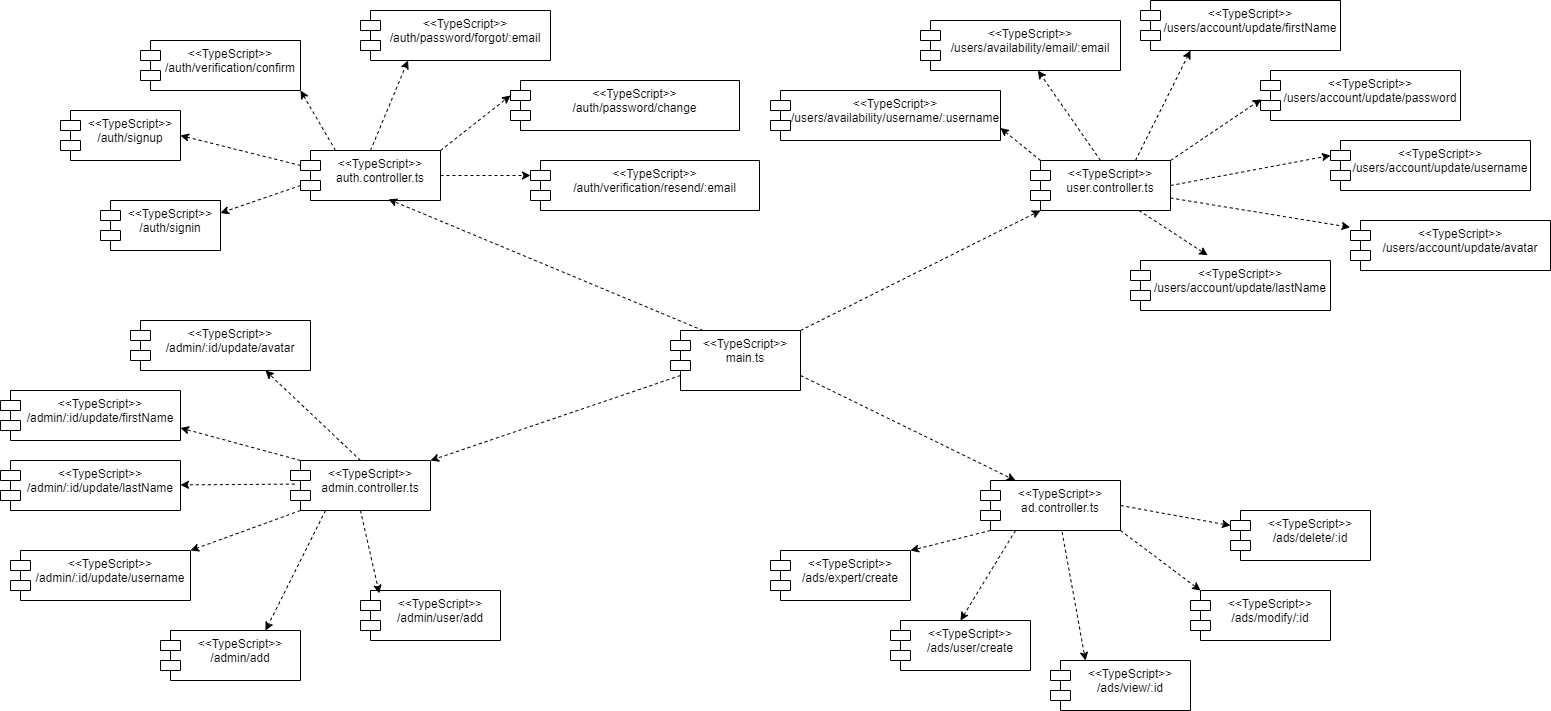


Značenje atributa je sledeće:

* Route Parameters (routeParams): Parametri putanje se koriste za prosleđivanje informacija između različitih delova aplikacije kroz URL. Na primer, u putanji /user/:id, id je parametar putanje koji može uticati na sadržaj stranice, na primer prikazivanjem detalja određenog korisnika.
* Query Parameters (queryParams): Query string parametri se koriste za prosleđivanje opcionalnih informacija u URL-u. Na primer, u URL-u /search?q=query, q je query parameter koji može sadržati ključnu reč za pretragu. Na osnovu ovog parametra, sadržaj stranice može se prilagoditi rezultatima pretrage.
* Angular Service parametri (appServices): Možemo koristiti Angular servise za deljenje podataka između komponenata. Parametri ili podaci koji se čuvaju u Angular servisima mogu uticati na sadržaj stranice, na primer prikazivanjem podataka koje je korisnik unoseo u formi ili izabranog elementa na stranici.
* Stanje aplikacije (appStates): Parametri koji se čuvaju u stanju aplikacije, na primer u ngrx store-u ili u lokalnom skladištu (local storage), takođe mogu uticati na sadržaj stranice. Na primer, ako korisnik izabere određenu opciju ili promeni postavke, to može uticati na prikaz stranice.

### Komponente aplikacione logike

Komponente koje realizuju domen problema se uključuju isključivo preko **main.ts** komponente korisničkog interfejsa. Na taj način zadržavaju sva podešavanja stila definisana u ovom skriptu. Na sledećem dijagramu su prikazane komponente ovog sloja i njihove međusobne zavisnosti:

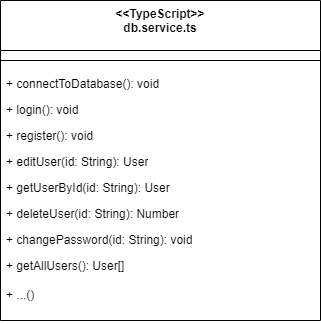


### Komponente za pristup podacima

Pristup bazi podataka je u potpunosti zatvoren unutar funkcija definisanih u okviru TypeORM servisa u NestJS aplikaciji. Pomenuti servis se uvozi na početku glavnog modula aplikacije, tako da su sve funkcije za pristup podacima dostupne svim komponentama.



Na sledećem UML dijagramu klasa pobrojane su funkcije za pristup podacima iz baze:



Navedene funkcije obavljaju sledeće zadatke:

• connectToDatabase - funkcija za povezivanje na bazu podataka, poziva se u glavnom servisu ili modulu

• login - funkcija za prijavu korisnika

• register - funkcija za registraciju novog korisnika

• editUser - funkcija za izmenu podataka postojećeg korisnika

• getUserById - funkcija za dobijanje korisnika na osnovu ID-ja

• deleteUser - funkcija za brisanje korisnika

• changePassword - funkcija za promenu lozinke korisnika

• getAllUsers - funkcija za dobijanje svih korisnika

• (ostale funkcije)

# Performanse

Odabrana softverska arhitektura omogućava ispunjenje zahteva u vezi sa brojem korisnika koji mogu istovremeno pristupati sistemu i vremenom odziva za pristup bazi podataka, kako je specificirano u performansama [5]:

1. Sistem će da podrži do 10000 simultanih pristupa korisnika portalu.
2. Vreme potrebno za pristupanje bazi podataka u cilju izvršenje nekog upita ne sme da bude veće od 5 sekundi.

Odabrane tehnologije za razvoj sistema i definisane hardverske platforme osiguravaju zadovoljenje traženih performansi [5].

# Kvalitet

Izabrana arhitektura softvera podržava zahteve u pogledu dostupnosti i srednjeg vremena između otkaza specificirane u zahtevima u pogledu pouzdanosti [5]:

1. pitajStrucnjaka portal će biti dostupan 24 časa dnevno, 7 dana u nedelji.
2. Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 24 sati.