# Mensarium

# Arhitekturni projekat

# Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 23.04.2017. | 1.0 | Inicijalna verzija | Marija Stojković, 15422  Nikola Savić, 15355,  Stefan Stanković, 15382  Dalibor Aleksić, 15024 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Sadržaj

[1. Cilj dokumenta 4](#_Toc484217648)

[2. Opseg dokumenta 4](#_Toc484217649)

[3. Reference 4](#_Toc484217650)

[4. Predstavljanje arhitekture 4](#_Toc484217651)

[5. Ciljevi i ograničenja arhitekture 4](#_Toc484217652)

[6. Pogled na slučajeve korišćenja 4](#_Toc484217653)

[6.1 Dijagrami slučajeva korišćenja 5](#_Toc484217654)

[6.2 Kratak opis slučajeva korišćenja 9](#_Toc484217655)

[6.2.1 Registracija na sistem 9](#_Toc484217656)

[6.2.2 Uplata putem SMS-a 9](#_Toc484217657)

[6.2.3 Ažuriranje informacija o menzama 9](#_Toc484217658)

[6.2.4 Obaveštavanje o trenutnim obrocima 9](#_Toc484217659)

[6.2.5 Pozivanje na obrok 9](#_Toc484217660)

[6.2.6 Praćenje objava korisnika 9](#_Toc484217661)

[6.2.7 Pregled ličnih informacija 9](#_Toc484217662)

[6.2.8 Uplata obroka 9](#_Toc484217663)

[6.2.9 Umanjenje broja obroka 9](#_Toc484217664)

[6.2.10 Provera stanja naloga 9](#_Toc484217665)

[6.2.11 Prijavljivanje na sistem 9](#_Toc484217666)

[6.2.12 Pregled gužve u menzama 9](#_Toc484217667)

[6.2.13 Pregled informacija o radnom vremenu 9](#_Toc484217668)

[6.2.14 Uređivanje naloga administratora 10](#_Toc484217669)

[6.2.15 Uređivanje naloga studenata 10](#_Toc484217670)

[6.2.16 Uređivanje naloga šalterskih radnika 10](#_Toc484217671)

[6.2.17 Uređivanje naloga upošljenika za kreiranje studnetskih naloga 10](#_Toc484217672)

[6.2.18 Kreiranje studenta 10](#_Toc484217673)

[6.2.19 Modifikacija studenta 10](#_Toc484217674)

[6.2.20 Brisanje 10](#_Toc484217675)

[6.2.21 Pregled informacija o studentima 10](#_Toc484217676)

[7. Pogled na logičku arhitekturu sistema 10](#_Toc484217677)

[7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 11](#_Toc484217678)

[7.1.1 Korisnički interfejs 11](#_Toc484217679)

[7.1.2 Aplikaciona logika 11](#_Toc484217680)

[7.1.3 Pristup podacima 11](#_Toc484217681)

[7.1.4 .NET 11](#_Toc484217682)

[7.1.5 Android 12](#_Toc484217683)

[7.1.6 ASP.NET 12](#_Toc484217684)

[7.1.7 MySQL 12](#_Toc484217685)

[8. Pogled na procese 12](#_Toc484217686)

[8.1 Procesi 12](#_Toc484217687)

[8.1.1 Android aplikacija 12](#_Toc484217688)

[8.1.3 Web server 13](#_Toc484217689)

[8.1.4 MySQL Server 13](#_Toc484217690)

[9. Pogled na raspoređivanje sistema 13](#_Toc484217691)

[9.1 Klijent 13](#_Toc484217692)

[9.2 Klijent 13](#_Toc484217693)

[9.3 Web server 13](#_Toc484217694)

[9.4 DBMS server 13](#_Toc484217695)

[10. Pogled na implementaciju sistema 14](#_Toc484217696)

[10.1 Model domena 14](#_Toc484217697)

[10.2 Šema baze podataka 15](#_Toc484217698)

[10.3 Komponente Sistema 16](#_Toc484217699)

[10.3.1 Komponenta za pristup podacima 16](#_Toc484217700)

[10.3.2 Komponenta aplikacione logike 20](#_Toc484217701)

[10.3.3 Komponenta korisničkog interfejsa 25](#_Toc484217702)

[11. Performanse 28](#_Toc484217703)

[12. Kvalitet 29](#_Toc484217704)

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture Mensarium sistema.

# Opseg dokumenta

Dokument se odnosi na Mensarium sistem koji će biti razvijen od strane InnoStorm tima. Mensarium predstavlja aplikaciju namenjenu korisnicima koji imaju pravo na ishranu u studentskim restoranima(menzama) i zapošljenima u Studetskim centrima. Namena sistema je efiksno korišćenje prava na ishranu i upravljanje informacijama o korisnicima ishrane

# Reference

Spisak korišćene literature:

1. Mensarium, predlog prijekta, SWE-Mensarium-01, V1.0, 2017, InnoStorm.
2. Mensarium – Planirani raspored aktivnosti na projektu, V1.0, 2017, InnoStorm..
3. Mensarium – Plan realizacije projekta, V1.0, 2017, InnoStorm.
4. Mensarium – Vizija sistema, V1.1, 2017, InnoStorm.
5. Mensarium – Specifikacija zahteva, V1.1, 2017, InnoStorm.

# 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. Mensarium sistem će biti implementiran kao android i desktop aplikacija zasnovana na ASP.NET *framework-u*  MySQL bazi podataka [4].
2. Klijentski deo Mensarium-a će biti optimizovan za Windows7+ i Android4.4+ [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].

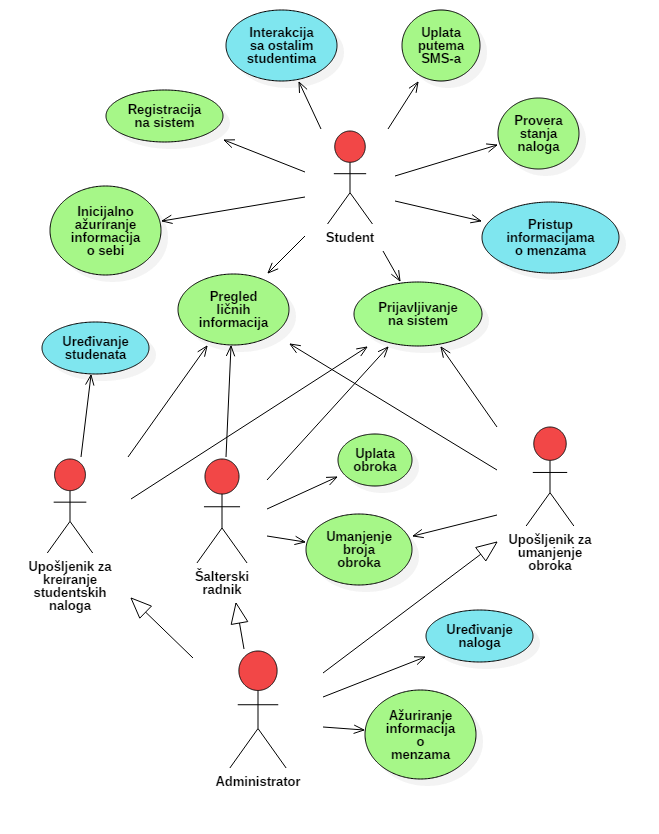
Slučajevi korišćenja Mensarium sistema su:

* *Interakcija sa ostalim studentima*
  + Obaveštavanje o trenutnim obrocima
  + Pozivanje na obrok
  + Praćenje objava korisnika
* Uplata putem SMS-a
* Provera stanja naloga
* *Pristup informacijama o menzama*
  + Pregled gužve u menzama
  + Pregled informacija o radnom vremenu
* Prijavljivanje na sistem
* Pregled ličnih informacija
* Inicijalno ažuriranje informacija o sebi
* Registracija na sistem
* *Uređivanje studenata*
  + Kreiranje studenata
  + Modifikacija studenata
  + Brisanje studenata
  + Pregled informacija o studentima
* Uplata obroka
* Umanjenje broja obroka
* *Uređivanje naloga*
  + Upošljenik za kreiranje studentskih naloga CRUD
  + Šalterski radnik CRUD
  + Upošljenik za umanjenje obroka CRUD
  + Administrator CRUD
* Ažuriranje informacija o menzama

Ove slučajevi korišćenja mogu da iniciraju student, upošljenik za kreiranje studentskih naloga, šalterski radnik, upošljenik za umanjenje obroka i administrator.

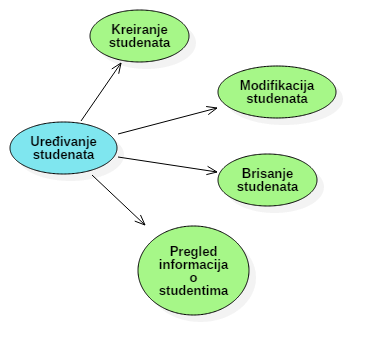
## Dijagrami slučajeva korišćenja

Osnovni UML dijagram koji prikazuje korisnike i slučajeve korišćenja Mensarium sistema prikazan je na sledećoj slici:

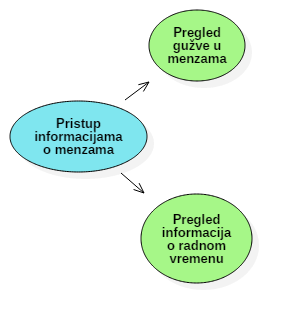


Slučajevi korišćenja *uređivanje studenata*, *pristup informacijama o menzama*, *interakcija sa ostalim studentima* i *uređivanje naloga* obuhvataju složenije radnje koje se mogu razložiti dalje razložiti na pojedinačne slučajeve korišćenja.

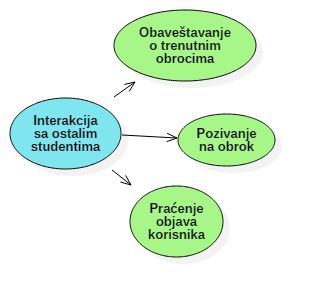
Detaljni UML dijagram za slučaj korišćenja *uređivanje studenata* je prikazan na sledećoj slici:



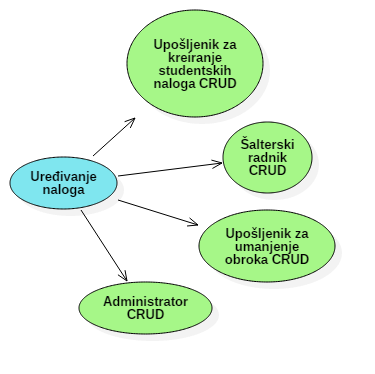
Detaljni UML dijagram za slučaj korišćenja *pristup informacijama o menzama* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *interakcija sa ostalim studentima* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *uređivanje naloga* je prikazan na sledećoj slici:



## Kratak opis slučajeva korišćenja

### Registracija na sistem

Kratak opis: Registracija na sistem pomoću dobijenog ID-a i inicijalne šifre dobijene od menadžera i otključavanje svih funkcionalnosti sistema.

Akter koji inicira slučaj korišćenja: Student.

### Uplata putem SMS-a

Kratak opis: Uplata obroka na nalog slanjem SMS-a.

Akter koji inicira slučaj korišćenja: Student.

### Ažuriranje informacija o menzama

Kratak opis: Ažuriranje informacija vezanih za studentske restorane kao što su lokacija ili radno vreme.

Akter koji inicira slučaj korišćenja: Administrator.

### Obaveštavanje o trenutnim obrocima

Kratak opis: Slanje obaveštenja drugim korisnicima o trenutnom meniju u menzi.

Akter koji inicira slučaj korišćenja: Student.

### Pozivanje na obrok

Kratak opis: Studentima koji interaguju sa datim korisnikom se šalje poziv za zajednički odlazak u menzu.

Akter koji inicira slučaj korišćenja: Student.

### Praćenje objava korisnika

Kratak opis: Prikaz objava o ponudi u menzama koje su poslali studenti sa kojima dati korisnik interaguje.

Akter koji inicira slučaj korišćenja: Student.

### Pregled ličnih informacija

Kratak opis: Pregled ličnih informacija o korisniku.

Akteri koji iniciraju slučaj korišćenja: Student, Upošljenik za kreiranjestudentskih naloga, Šalterski radnik, Upošljenik za umanjenje obroka, Administrator.

### Uplata obroka

Kratak opis: Uplata jednog ili većeg broja obroka na nalog studenta.

Akteri koji iniciraju slučaj korišćenja: Šalterski radnik, Administrator.

### Umanjenje broja obroka

Kratak opis: Umanjenje broja obroka studentu u trenutku iskorišćenja ili zbog greške radnika.

Akteri koji iniciraju slučaj korišćenja: Šalterski radnik, Upošljenik za umanjenje obroka, Administrator.

### Provera stanja naloga

Kratak opis: Provera broja obroka na nalogu.

Akter koji inicira slučaj korišćenja: Student.

### Prijavljivanje na sistem

Kratak opis: Akter se prijavljuje na sistem sa svojim korisničkim imenom i lozinkom.

Akteri koji iniciraju slučaj korišćenja: Student, Upošljenik za kreiranje studentskih naloga, Šalterski radnik, Upošljenik za umanjenje obroka, Administrator.

### Pregled gužve u menzama

Kratak opis: Biranje menze i prikaz trenutnog broja studenata u izabranoj menzi.

Akter koji inicira slučaj korišćenja: Student.

### Pregled informacija o radnom vremenu

Kratak opis: Biranje menze i prikaz radnog vremena.

Akter koji inicira slučaj korišćenja: Student.

### Uređivanje naloga administratora

Kratak opis: Kreiranje, modifikacija i brisanje naloga administratora.

Akter koji inicira slučaj korišćenja: Administrator.

### Uređivanje naloga studenata

Kratak opis: Kreiranje, modifikacija i brisanje naloga studenta.

Akter koji inicira slučaj korišćenja: Administrator.

### Uređivanje naloga šalterskih radnika

Kratak opis: Kreiranje, modifikacija i brisanje naloga šalterskih radnika.

Akter koji inicira slučaj korišćenja: Administrator.

### Uređivanje naloga upošljenika za kreiranje studnetskih naloga

Kratak opis: Kreiranje, modifikacija i brisanje naloga upošljenika za kreiranje studentskih naloga,.

Akter koji inicira slučaj korišćenja: Administrator.

### Kreiranje studenta

Kratak opis: Kreiranje novog korisničkog naloga tipa student na sistemu.

Akteri koji iniciraju slučaj korišćenja: Student,, Administrator.

### Modifikacija studenta

Kratak opis: Promena informacija o studentu na već kreiranom nalogu.

Akteri koji iniciraju slučaj korišćenja: Upošljenik za kreiranje studentskih naloga, Administrator.

### Brisanje

Kratak opis: Uklanjanje korisnika iz sistema.

Akteri koji iniciraju slučaj korišćenja: Upošljenik za kreiranje studentskih naloga, Administrator.

### Pregled informacija o studentima

Kratak opis: Pregled svih informacija o studentskim nalozima.

Akteri koji iniciraju slučaj korišćenja: Upošljenik za kreiranje studentskih naloga, Administrator.

# 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.

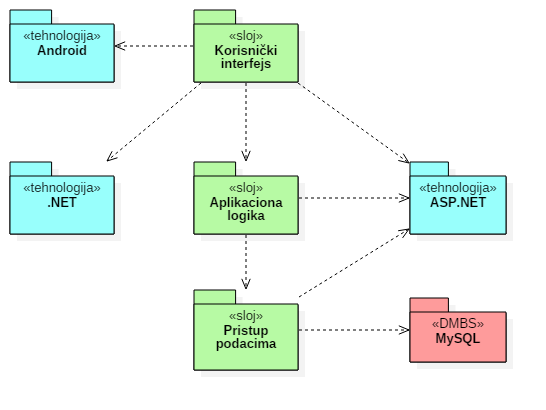
Logički pogled na Mensarium-a obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket *Korisnički interfejs* sadrži Android aplikaciju, desktop aplikaciju i ASP.NET *source* fajlove koji realizuju grafički dizajn i forme preko kojih korisnici sistema komuniciraju sa sistemom.

Paket *Aplikaciona logika* predstavlja srednji sloj sistema koji sadrži ASP.NET *source* fajlove zadužene za realizaciju funkcionalnosti specifičnih za domen sistema koji se razvija.

Paket *Pristup podacima* sadrži ASP.NET *source* fajlove koji predstavljaju interfejs za pristup, dodavanje i ažuriranje podataka koji se čuvaju u bazi podataka.

## Pregled arhitekture – organizacija paketa i podsistema u slojeve



### Korisnički interfejs

sloj

Ovaj sloj realizuje korisnički interfejs sistema. U njemu su sadržani android aplikacija, desktop aplikacija i web servis izrađen u ASP.NET-u koji obrađuje zahteve korisnika. Korisnik preko ovog sloja komunicira sa sistemom. Sloj korisničkog interfejsa zavisi od sloja aplikacione logike kao i od paketa Android, .NET i ASP.NET.

### Aplikaciona logika

sloj

Sloj aplikacione logike je srednji sloj u troslojnoj arhitekturi Mensarium-a. Sadrži ASP.NET *source* fajlove koji realizuju funkcionalnost karakterističnu za domen primene sistemam i uspostavljaju vezu između korisničkog interfejsa i sloja za pristup podacima.

Ovaj sloj zavisi od sloja za pristup podacima i ASP.NET paketa.

### Pristup podacima

sloj

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

Ovaj sloj ne zavisi od drugih slojeva, ali je zavisan od paketa ASP.NET i MySQL baza podataka.

### .NET

tehnologija

Tehnologija .NET definiše gradivne elemente desktop aplikacije koji se prikazuju na računaru i omogućavaju prikaz formatiranih informacija i realizaciju formi za unos i ažuriranje podataka.

### Android

tehnologija

Tehnologija Android definiše gradivne elemente mobilne aplikacije koji se prikazuju na mobilnom telefonu i omogućavaju prikaz formatiranih informacija i realizaciju formi za unos i ažuriranje podataka.

### ASP.NET

tehnologija

Tehnologija ASP.NET obezbeđuje mehanizam za pisanje i izvršavanje koda na strani servera. Ovaj kod obrađuje i generiše podatke koje su potrebni korisničkom interfejsu i pristupaja bazi podataka u cilju pribavljanja, unosa i ažuriranja podataka.

### MySQL

DBMS

MySQL predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju Mensarium sistema.

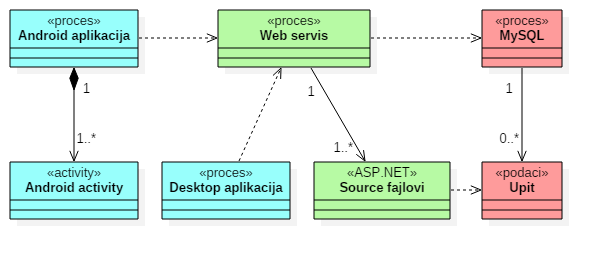
# Pogled na procese

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.

Ilustracije radi u nastavku je dat opis procesa uključenih u izvršenje Mensarium sistema kao android i desktop aplikacije.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju Mensarium sistema.



### Android aplikacija

Android aplikacija je proces koji se izvršava na mobilnom telefonu o pikazuje određene *activity-*je. U najopštijem slučaju prikazuje samo jedan *activity.*

Androdi aplikacija zavisi od web servera koji obrađuje i šalje podatke aplikaciji.

* + 1. *Desktop aplikacija*

Desktop aplikacija je proces koji se izvršava na desktop računaru klijenta. U najoopštijem slučaju se ovaj proces sastoji od jedne niti.

Desktop aplikacija zavisi od web servera koji obrađuje i šalje podatke aplikaciji.

### Web server

Web server je proces koji izvršava funkcionalnost opsluživanja zahteva prispelih sa više android i desktop aplikacija. Web servirs može da se sastoji iz više procesa i niti u cilju opsluživanja više klijenata.

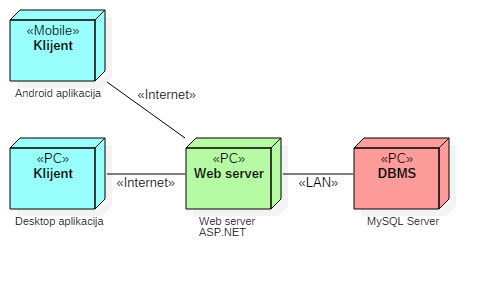
### MySQL Server

MySQL Server je proces koji izvršava funkcionalnost MySQL sistema za upravljanje bazama podataka. Ovaj proces može konkurentno da prihvati određen broj upita, izvrši ih nad bazom podataka i vrati rezultate procesu koji je upite postavio.

# 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 Mensarium sistema.



## Klijent

Pristup Mensarium sistemu se obavlja preko klijentskih računara na kojima se izvršava desktop aplikacija. Za povezivanje između klijenta i Web servera koristi se Internet infrastruktura tako da nema ograničenja u pogledu lokacije klijenta.

## Klijent

Pristup Mensarium sistemu se obavlja preko klijentskih mobilnih telefona na kojima se izvršava Android aplikacija. Za povezivanje između klijenta i Web servera koristi se Internet infrastruktura tako da nema ograničenja u pogledu lokacije klijenta.

## Web server

Računar na kome se izvršava Web server opslužuje više klijenata koji pristupaju preko Interneta. U najopštioj konfiguraciji DBMS se izvršava na posebnoj mašini koja je sa Web serverom u lokalnoj mreži (LAN).

## DBMS server

DBMS server je računar na kome se izvršava MySQL Server proces koji realizuje funkcionalnost sistema za upravljanje bazama podataka. Zbog sigurnosti podataka koji se na ovom računaru čuvaju pristup bazi je ograničen samo na računare iz lokalne mreže (LAN).

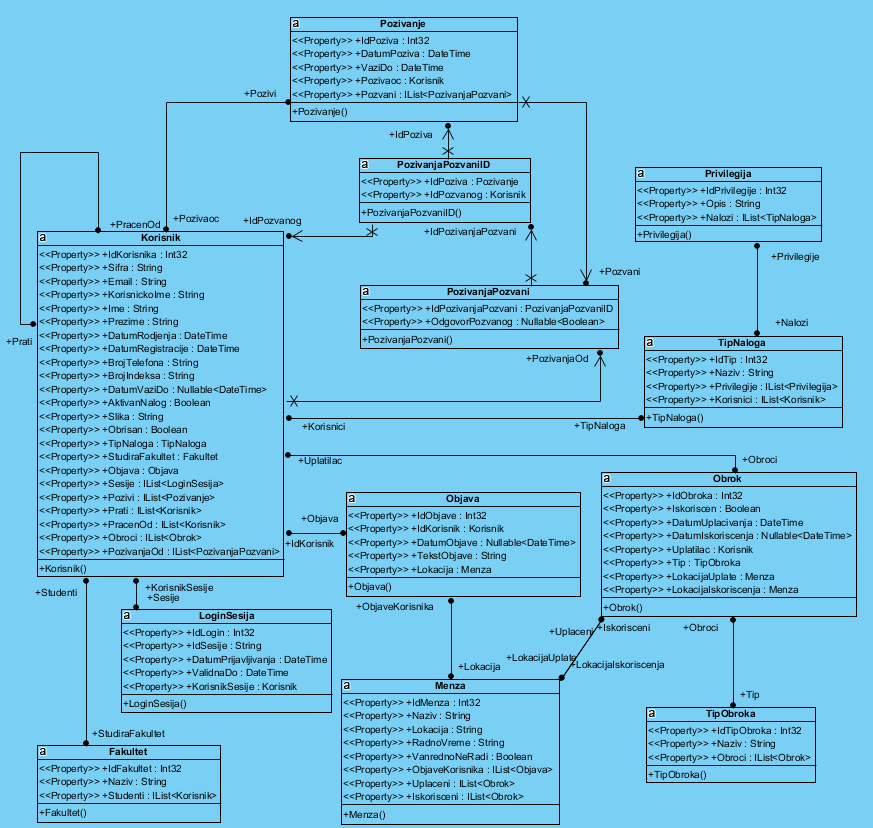
# Pogled na implementaciju sistema

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

## Model domena

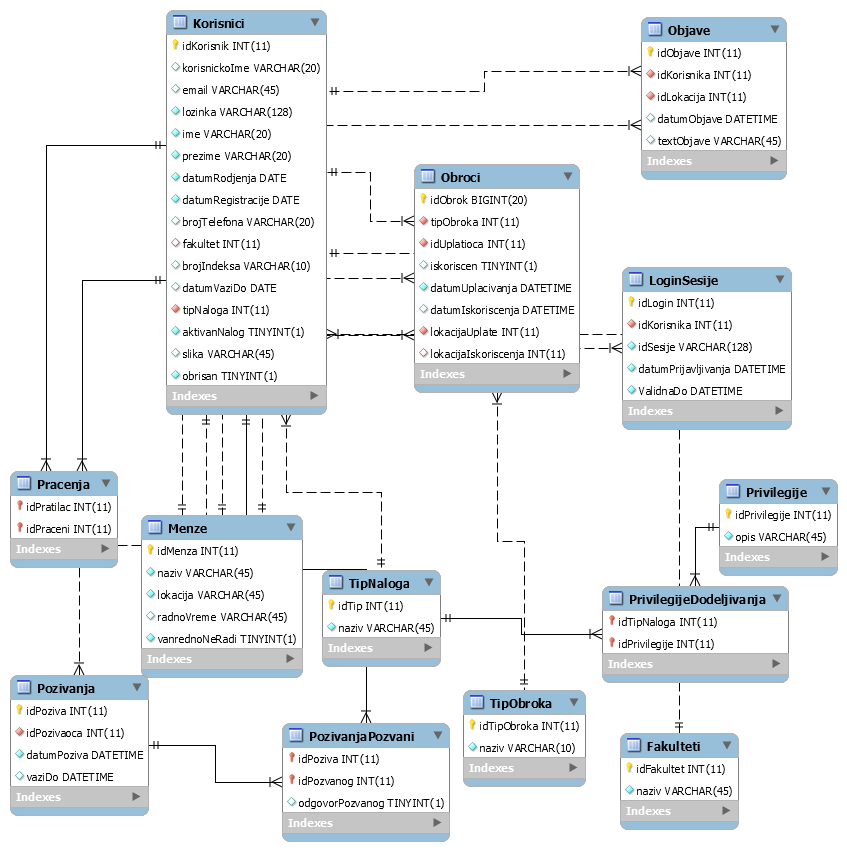
Model domena za koji se Mensarium sistem projektuje je ilustrovan UML dijagramom klasa. U njemu su prikazane domenske klase, neki od njihovih atributa, kao i veze koje se mogu identifikovati između njih.

Model domena predstavlja osnovu za projektovanje baze podataka, ali i identifikaciju nekih od komponenti (ASP.NET source fajlova) koje će biti implementirane.

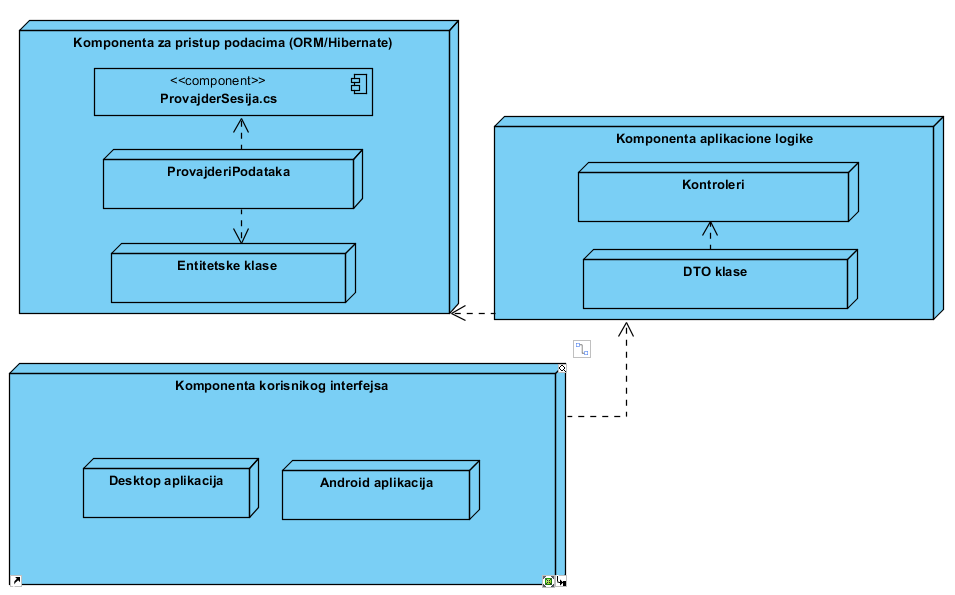


## Šema baze podataka

Detaljna šema baze podataka je prikazana na sledećem dijagramu. Baza podataka i dijagram su kreirani korišćenjem *MySQL Workbench-a*. Migracija na MySQL je obavljena pomoću tog alata.



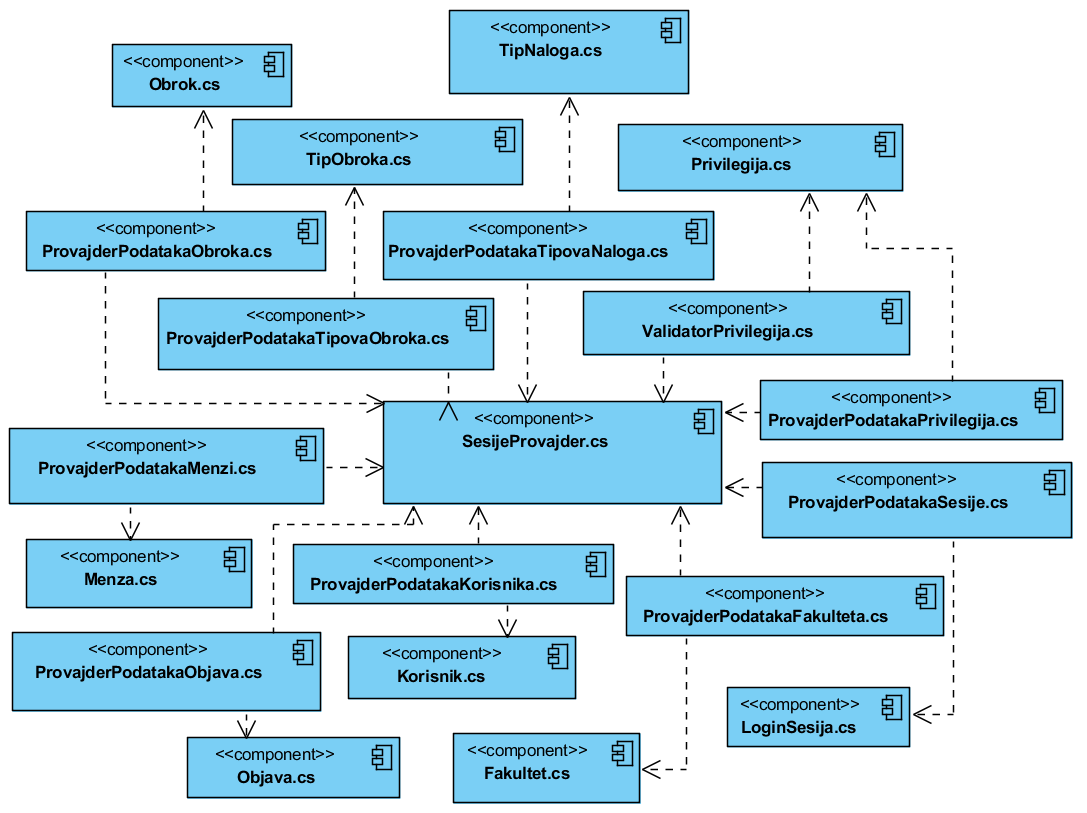
## Komponente Sistema

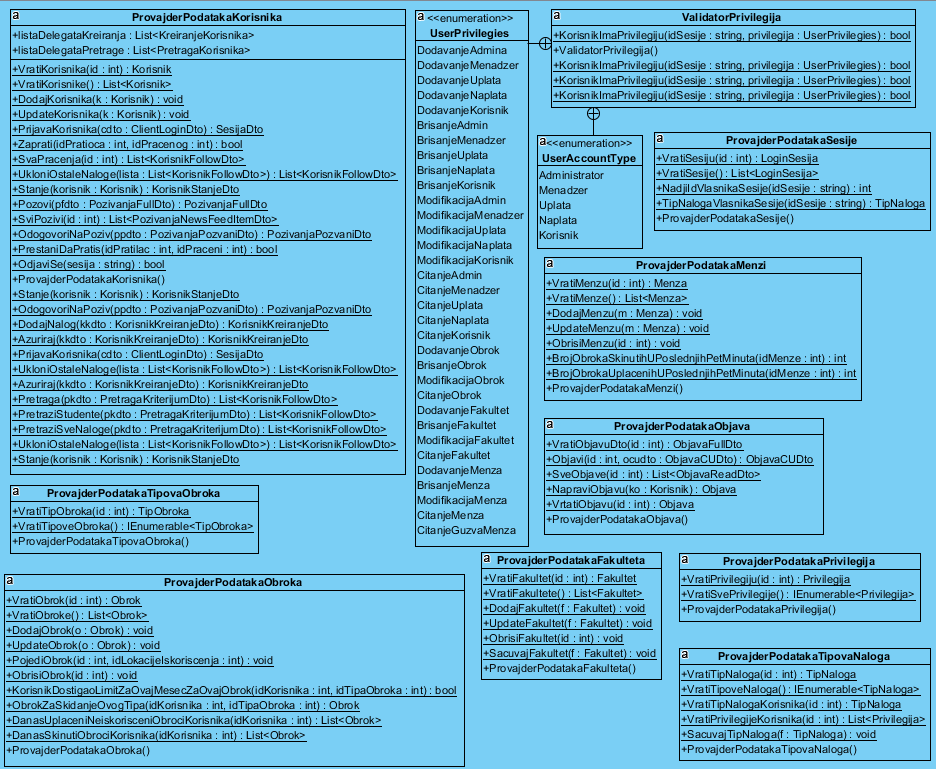


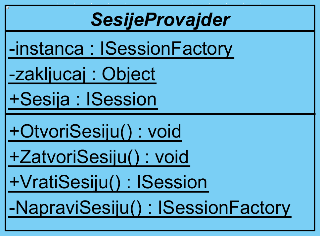
### Komponenta za pristup podacima

Pristup bazi podataka zatvoren je u funkcije koje su definisane u okviru skupa fajlova pod imenom Provajderi podataka, a koji pristupaju preko Provajdera sesije. Podaci koji se ovako dobijaju su entitentskog tipa.

Na sledećim slikama predstavljen je dijagram razmeštaja na kome je prikazano kako kompuniciraju Provajderi podataka sa bazom I entitetskim klasama. Dati su I detaljni klasni dijagrami Provajdera podataka I Provajdera sesije, gde su pobrojane funkcije koje obavljaju, dok su klasni dijagrami entiteta je već prikazani u okvitu sekcije Model domena (10.1).







Zadaci koje obavljaju funkcije Provajdera podataka korisnika:

* VratiKorisnika – vraća korisnika sa zadatim ID-em
* VratiKorisnike – vraća listu svih korisnika
* DodajKorisnika – dodaje korisnika
* UpdateKorisnika – ažurira korisnika
* PrijavaKorisnika – loguje korisnika
* Zaprati – student zapraćuje drugog studenta
* SvaPracenja – vraća listu studenata koje korisnik pratie
* Stanje – vraća korisnikovo stanje obroka
* Pozovi – korisnik poziva drugog na zajednički obrok
* SviPozivi - vraća listu svih poziva korisnika
* OdgovoriNaPoziv – odgovara na poziv za zajednički obrok
* PrestaniDaPratis – korisnik prestaje da prati korisnika koga je pratio
* OdjaviSe – odjaljuje korisnika sa sistema
* OdgovoriNaPoziv – odgovra potvrdo ili odrično na poziv za zajednički obrok
* DodajNalog – dodaje korisnički nalog
* Azuriraj – ažurira korisnički nalog
* PretraziStudente – vrši pretragu studenata po zadatom kriterijumu
* PretraziSveNaloge – vrši pretragu svih naloga po zadatom kriterijumu

Zadaci koje obavljaju funkcije Provajdera podataka objava:

* VratiObjavu – vraća objavu sa zadatim ID-em
* Objavi – ažurira objavu korisnika koju vide drugi
* SveObjave – vraća listu svih objava
* NapraviObjavu – kreira novu objavu

Zadaci koje obavljaju funkcije Provajdera podataka fakulteta:

* VratiFakultet – vraća fakultet sa zadatim ID-em
* VratiFakultete – vraća listu svih fakulteta
* DodajFakultet – dodaje fakultet
* UpdateFakultet – ažurira fakultet
* ObrisiFakultet – briše fakultet sa zadatim ID-em
* SacuvajFakultet – čuva fakultet

Zadaci koje obavljaju funkcije Provajdera podataka menzi:

* VratiMenzu – vraća menzu sa zadatim ID-em
* VratiMenze – vraća listu svih menzi
* DodajMenzu – dodaje menzu
* UpdateMenzu – ažurira menzu
* BrojObrokaSkinutihUPoslednjihPetMinuta – vraća broj obroka potrošenih u poslednjih pet minuta u izabranoj menzi
* BrojObrokaUplacenihUPoslenjihPetMinuta – vraća broj obroka uplaćenih u poslednjih pet minuta u izabranoj menzi

Zadaci koje obavljaju funkcije Provajdera podataka obroka:

* VratiObrok – vraća obrok sa zadatim ID-em
* VratiObroke – vraća listu svih obroka
* DodajObrok – dodaje obrok
* UpdateObrok – ažurira obrok
* PojediObrok – označava obrok kao iskorišćen
* ObrisiObrok – briše obrok sa zadatim ID-em
* KorisnikDostigaoLimitZaOvajMesecZaOvajObrok – vraća da li je korisnik dostigao limit od 30 obroka mesrčno za određeni tip obroka
* ObrokZaSkidanjeOvogTipa – vraća korisnikov prvi obrok ovog tipa koji je neiskorišćen, a uplaćen ovog meseca
* DanasUplaceniNeiskorisceniObrociKorisnika – vraća listu obroka koje je korisnik danas uplatio, a nije iskoristio
* DanasSkinutiObrociKorisnika – vraća istu obroka koji su danas naplaćeni korisniku

Zadaci koje obavljaju funkcije Provajdera podataka sesije:

* VratiSesiju – vraća sesiju sa zadatim ID-em
* VratiSesije – vraća listu svih sesija
* NadjiIdVlasnikaSesije – vraća ID korisnika zadate sesije
* TipNalogaVlasnikaSesije – vraća tip naloga korisnika zadate sesije

Zadaci koje obavljaju funkcije Provajdera podataka tipova obroka:

* VratiTipObroka – vraća tip obroka sa zadatim ID-em
* VratiSveTipoveObroka – vraća listu svih tipova obroka

Zadaci koje obavljaju funkcije Provajdera podataka tipova naloga:

* VratiTipNaloga – vraća tip naloga sa zadatim ID-em
* VratiTipoveNaloga – vraća listu svih tipova naloga
* VratiTipNalogaKorisnika – vraća tip naloga zadatog korisnika
* VratiPrivilegijeKorisnika – vraća listu privilegija korisnika
* SacuvajTipNaloga – čuva tip naloga

Zadaci koje obavljaju funkcije Provajdera podataka privilegija

* VratiPrivilegiju – vraća privilegiju sa zadatim ID-em
* VratiSvePrivilegije – vraća listu svih privilegija

Zadaci koje obavljaju funkcije Validatora privilegija:

* KorisnikImaPrivilegiju – vraća da li korisnik ima određenu privilegiju

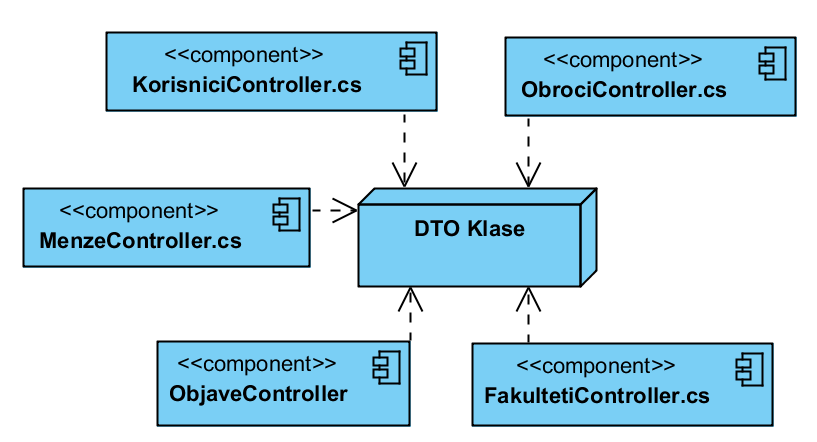
Zadaci koje obavljaju funkcije Provajdera Sesije:

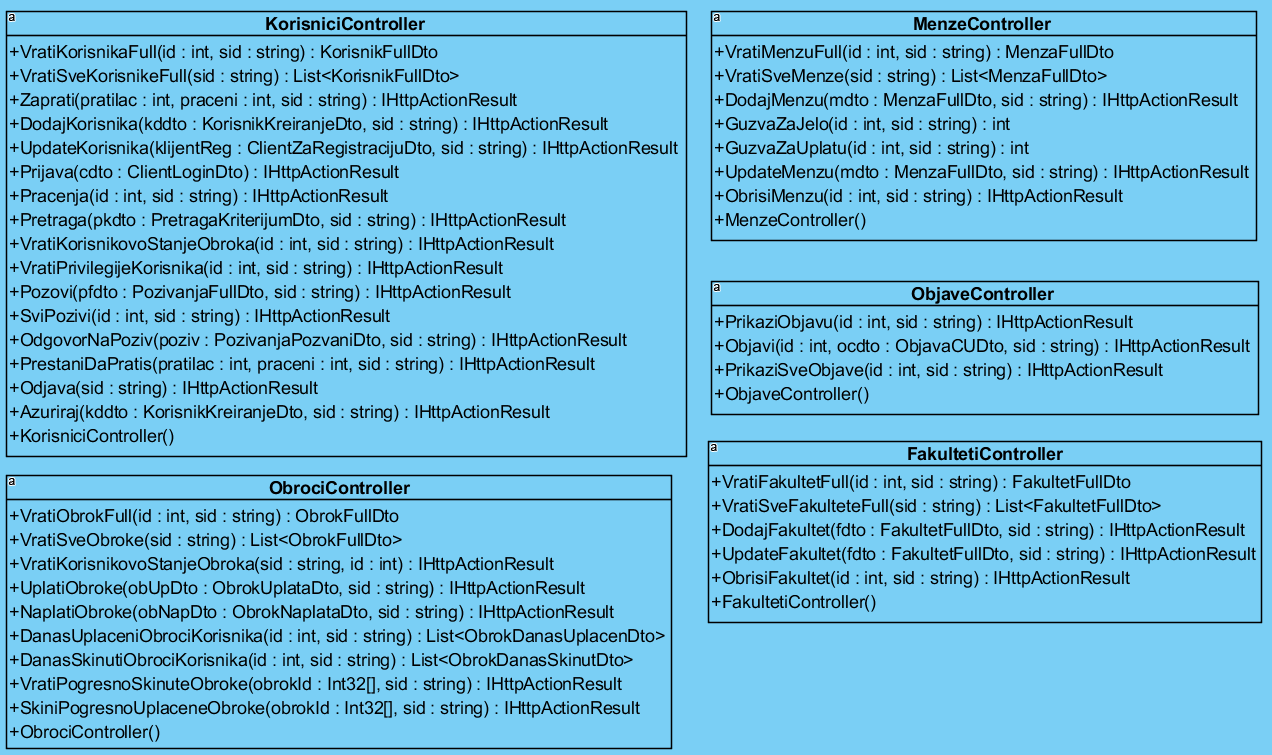
* OtvoriSesiju –otvara sesiju za rad sa bazom
* ZatvoriSesiju – zatvara sesiju
* NapraviSesiju – konfiguriše i kreira instance sesije

### Komponenta aplikacione logike

Aplikaciona logika podrazumeva kontrolere koji zahvaljujući komponenti za pristup podacima dobijaju podatke koje zatim enkapsuliraju u odgovarajuće DTO klase I šalju ih komponenti korisničkog interfejsa. Obrnuta komunikacija je kada od komponente korisničkog interfejsa dobiju podatke koje treba proslediti komponenti za rad sa bazom(komponenta za pristup podacima).

Na sledećim slikama dat je dijagram razmeštaja ove komponente, a zatim i detaljni klasni dijagrami kontrolera I DTO klasa.





Zadaci koje obavljaju funkcije kontrolera Korisnici:

* VratiKorisnikaFull – vraća korisnika sa zadatim ID-em
* VratiSveKorisnikeFull – vraća se lista svih korisnika
* Zaprati – jedan korisnik zapraćuje drugog
* DodajKorisnika – dodaje korisnika u bazu podataka
* UpdateKorisnika – ažurira korsnika
* Prijava – prijavljuje korisnika na sistem
* Pracenja – prikaz svih korisnika koje korisnik prati
* Pretraga – pretražuje korisnike na osnovu zadatog kriterijuma
* VratiKorisnikovoStaneObroka – vraća broj doručka, ručkova I večera korisnika sa zadatim ID-em
* VratiPrivilegijeKorisnika – vraća listu privilegija koje ima ovaj korisnik
* Pozovi – pozivaju se na zajednički obrok željeni studenti iz liste studenata sa kojima se interaguje
* SviPozivi – vraća sve pozive upućene jednom korisniku
* OdgovorNaPoziv – odgovara potvrdno ili odrično na poziv za zajednički obrok
* PrestaniDaPratis – student prestaje da prati izabranog studenta
* Odjava – korisnik se odjavljuje sa Sistema
* Azuriraj – dodaju se potrebni podaci pri prvoj prijavi na aplikaciju

Zadaci koje obavjaju funkcije kontrolera Obroci:

* VratiObrokFull – vraća obrok sa zadatim ID-em
* VratiSveObroke – vraća listu svih obroka
* UplatiObroke – uplaćuje željeni broj obroka određenog tipa korisniku
* NaplatiObroke – naplaćuje željeni broj obroka određenog tipa korisniku
* DanasUpaceniObrociKorisnika – vraća listu obroka koji su danas uplaćeni datom korisniku
* DanasSikuntiObrociKorisnika – vraća listu obroka koji su danas naplaćeni datom korisniku
* VratiPogresnoSkinuteObroke – označava kao neiskorišćene obroke sa zadatim ID-evima
* SkiniPogresnoUplaceneObroke – briše obroke sa zadatim ID-evima

Zadaci koje obavljaju funkcije kontrolera Menze:

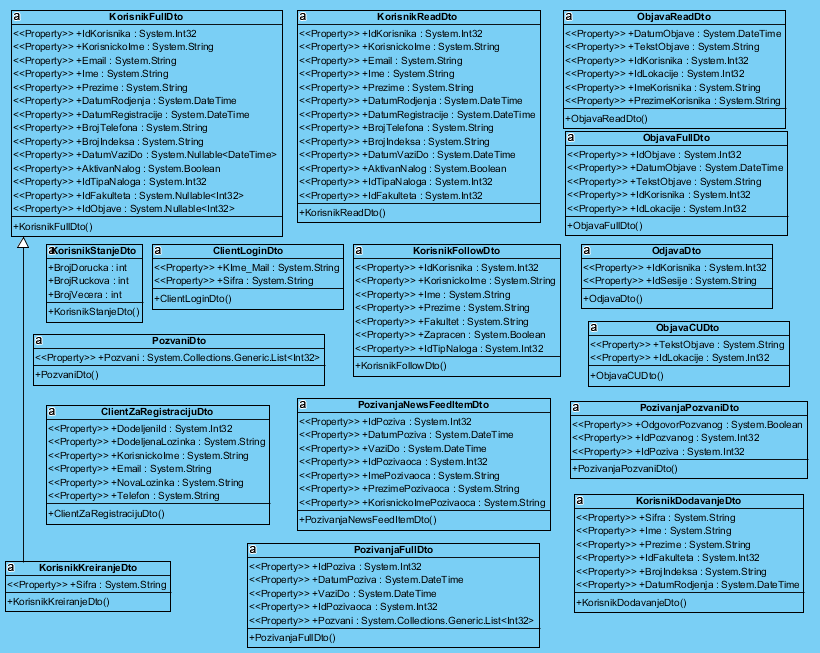
* VratiMenzuFull – vraća menzu sa zadatim ID-em
* VratiSveMeze – vraća listu svih menzi
* DodajMenzu – dodaje menzu u bazu podataka
* GuzvaZaJelo – vraća procenat gužve u redu za obroke u izabranoj menzi
* GuzvaZaUplatu – vraća procenat gužve u redu za uplatu u izabranoj menzi
* UpdateMenzu – ažurira menzu
* ObrisiMenzu – briše menzu sa zadatim ID-wm

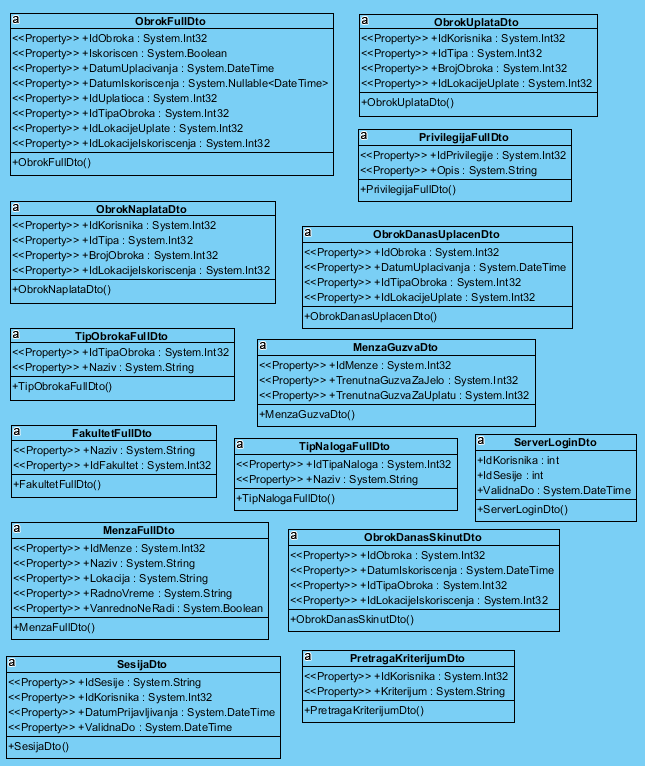
Zadaci koje obavljaju funkcije kontrolera Fakulteti

* VratiFakultetFull – vraća fakultet sa zadatim ID-em
* VratiSveFakultete – vraća listu svih fakulteta
* UpdateFakultet – ažurira fakultet
* ObrisiFakultet – briše fakultet sa zadatim ID-em

Zadaci koje obavljaju funkcije kontrolera Objave:

* PrikaziObjavu – vraća objavu sa zadatim ID-em
* Objavi – ažurira objavu korisnika
* PrikaziSveObjave – prikazuje objave korisnika koje korisnik prati

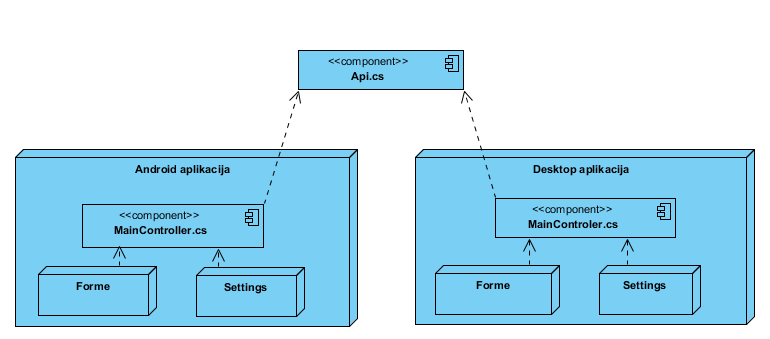




### Komponenta korisničkog interfejsa

Ovu komponentu čine desktop i android aplikacija. One komuniciraju sa komponentom aplikacione logike.

Dijagrami razmeštaja desktop I android aplikacije:



Na sledećim slikama dati su UML dijagrami fajlova Api.cs i MainController.cs vezani za rad desktop aplikacije, dijagram vezan za komponentu Settings, a zatim I objašnjenja funkcionalnosti.

UML dijagrami kontrolera:



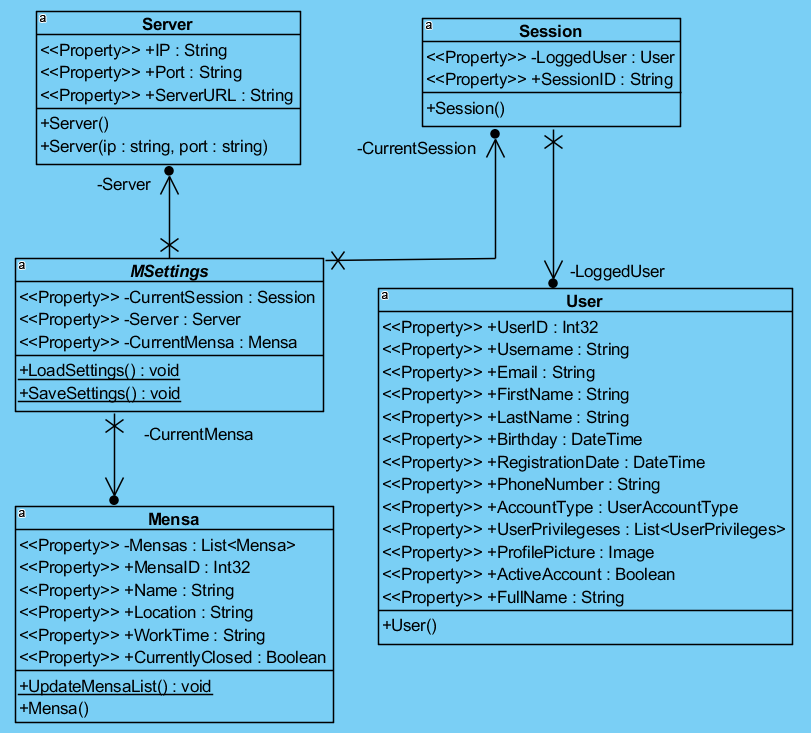
Zadaci koje obavljaju funkcije kontrolera Api:

* TestConnection – testira konekciju ka serveru
* GetUserFull – vraća korisnika sa zadatim ID-em
* GetUsersFull – vraća listu svih korisnika
* FollowUser – student zapraćuje drugog studenta
* AddNewUser – dodaje novog korisnika
* AndroidUserRegistration – vrši registraciju korisnika na sistem prilikom prvog pristupa aplikaciji
* LoginUser – prijavljuje korisnika na sistem
* UsersThatFollows – vraća listu studenata koje student prati
* SearchUsers – pretražuje korisnike po zadatom kriterijumu
* UserMealsCount – vraća broj obroka studenta
* UserPriviledges – vraća privilegije korisnika
* InviteUser – student poziva studenta na obrok
* UserCalledBy – vraća listu studenata koji su pozvali studenta na zajednički obrok
* Respond2Invite – odgovara na poziv za zajednički obrok
* Unfolow – student otpraćuje studenta
* LogoutUser – odjavljuje korisnika sa sistema
* UpdateUser – ažurira korisnika
* SetUserImage – postavlja sliku korisnika
* GetCurrentUserImage – vraća sliku trenutno aktivnog korisnika
* GetUserImage – vraća sliku traženog korisnika
* AddNewUser – dodaje novog korisnika
* UpdateFaculty – ažurira podatke o fakultetu
* DeleteFaculty – briše fakultet sa zadatim ID-em
* GetAllFaculties – vraća listu svih fakulteta
* GetFacultyInfo – vraća podatke od fakultetu
* GetAllMensas – vraća listu svih menzi
* GetMensaInfo – braća podatke o menzi
* AddNewMensa – dodaje novu menzu
* DeleteMensa – briše menzu sa zadatim ID-em
* CrowdInMensa – vraća procenat gužve za jelo u izabranoj menzi
* CrowdInPaymentCounter – vraća procenat gužve za uplatu obroku u izabranoj menzi
* UpdateMenza – ažurira podatke o menzi
* GetMealInfo – vraća informacije o izabranom obroku
* GetAllMealsInfo – vraća informacije o svim obrocima
* AddMeal – dodaje novi obrok
* UseMeal – naplaćuje obrok
* TodayAddedMeals – vraća listu danas uplaćenih obroka
* UndoUseMals – vraća naplaćene obroke na listu neiskorišćenih
* UndoAddMeals – briše uplaćene obroke
* ShowUserPost – prikazuje objavu studenta
* UserNewPost – vraća trenutnu objavu studenta
* GetFollowedUserPosts – vraća listu objava studenata koje student prati

Zadaci koje obavljaju funkcije kontrolera MainController:

* InitAplication – vrši inicijalizaciju parametara potrebnih za funkcionisanje aplikacije
* PostLoginInit – vrši inicijalizaciju parametara potrebnih parametara nakon logovanja
* Shutdown – zatvra aplikaciju
* TestConnection – testira konekciju ka serveru
* LoginUser – prijavljuje korisnika na sistem
* LogoutUser – odjavljuje korisnika sa sistema
* ChangeServerIP – menja IP adresu servera
* ChangeServerPort – menje port servera
* ChangeServer – menja server
* ChangeCurrentMensa – menja trenutnu menzu
* LoadUserCard – očitava karticu studenata
* AddUserMeal – uplaćuje obrok korisniku
* AddUserMeals – uplaćuje više obroka korisniku

UML dijagrami vezani za podešavanja:



Zadaci koje obavljaju prikazane funkcije:

* Server – vrši podešavanje server
* LoadSettings – učitava podešavanja
* SaveSettings – čuva podešavanja
* UpdateMensaList – ažurura listu menzi

# Performanse

Izabrana arhitektura softvera podržava zahteve u pogledu broja korisnika koji mogu simultano pristupati sistemu i vremena odziva za pristup bazi podataka specificirane u zahtevima u pogledu performansi [5]:

1. Sistem će da podrži do 2000 simultanih pristupa korisnika sistemu..
2. Vreme potrebno za pristupanje bazi podataka u cilju izvršenje nekog upita ne sme da bude veće od 5 sekundi.

Zahtevane performanse su zadovoljene izborom tehnologija na kojima će sistem biti razvijen i definisane hardverske platforme [5].

# Kvalitet

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

1. Mensarium sistem će biti dostupan 24 časa dnevno, 7 dana u nedelji. Vreme kada portal nije dostupan ne sme da pređe 10%.
2. Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 120 sati.