Food Assistant

Android aplikacija za planiranje ishrane

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 19.04.2018. | 1.0 | Inicijalna verzija | Aleksandar |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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 5

6.1 Dijagrami slučajeva korišćenja 6

6.2 Kratak opis slučajeva korišćenja 8

6.2.1 Izbor jezika 8

6.2.2 Pregled osnovnih podataka o laboratoriji 8

6.2.3 Pregled spiska članova 8

6.2.4 Pregled podataka o određenom članu laboratorije 8

6.2.5 Pregled publikacija po autoru 8

6.2.6 Pregled publikacija po tipu 8

6.2.7 Pregled publikacija po godini 8

6.2.8 Pregled spiska projekata 8

6.2.9 Pregled podataka o određenom projektu 8

6.2.10 Prijavljivanje 9

6.2.11 Ažuriranje podataka o članu 9

6.2.12 Dodavanje nove publikacije 9

6.2.13 Brisanje postojeće publikacije 9

6.2.14 Ažuriranje osnovnih podataka o laboratoriji 9

6.2.15 Kreiranje novog člana 9

6.2.16 Brisanje postojećeg člana 9

6.2.17 Arhiviranje postojećeg člana 9

6.2.18 Kreiranje projekta i postavljanje vođe 9

6.2.19 Ažuriranje podataka o projektu 9

7. Pogled na logičku arhitekturu sistema 9

7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 10

7.1.1 Korisnički interfejs 10

7.1.2 Aplikaciona logika 10

7.1.3 Pristup podacima 10

7.1.4 HTML 10

7.1.5 PHP 11

7.1.6 MySQL 11

8. Pogled na procese 11

8.1 Procesi 11

8.1.1 Web čitač 11

8.1.2 Web server 11

8.1.3 PHP-CGI 11

8.1.4 MySQL Server 12

9. Pogled na raspoređivanje sistema 12

9.1 Klijent 12

9.2 Web server 12

9.3 DBMS server 12

10. Pogled na implementaciju sistema 12

10.1 Model domena 12

10.2 Šema baze podataka 13

10.3 Komponente sistema 14

10.3.1 Komponente korisničkog interfejsa 14

10.3.2 Komponente aplikacione logike 14

10.3.3 Komponente za pristup podacima 15

11. Performanse 16

12. Kvalitet 16

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture Food Assistant aplikacije.

# Opseg dokumenta

Dokument se odnosi na Food Assistant aplikaciju koja će biti razvijena od strane tima Skywalkers. Namena aplikacije je organizovanje ishrane i omogućavanje korisniku na osnovu namirnica koje poseduje ostvari najoptimalniju i najzdraviju ishranu.

# Reference

Spisak korišćene literature:

1. Food Assistant– Predlog projekta V1.1, 2018, Skywalkers
2. Food Assistant – Planirani raspored aktivnosti na projektu, V1.0, 2018, Skywalkers.
3. Food Assistant – Plan realizacije projekta, V1.0, 2018, Skywalkers.
4. Food Assistant – Vizija sistema, V1.0, 2018, Skywalkers.
5. Food Assistant – Specifikacija zahteva, V1.0, 2018, Skywalkers.

# 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. Aplikacija Food Assistant će biti implementirana kao Android aplikacija zasnovana na jeziku Javi, Android API-ju i MySQL bazi podataka.
2. 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 Food Assistant portala su:

1.Prijava

2.Pristupanje grupi

3.Ažuriranje frižidera

4.Sastavljanje predloga jela

5.Kreiranje liste omiljenih namirnica

6.Kreiranje liste nepoželjnih namirnica

7.Prikaz informacija

7.1.Mehanizam koji nudi naredni obrok

7.2.Kreiranje liste omiljenih namirnica

7.3.Kreiranje liste nepoželjnih namirnica

7.4.Statistika o ishrani

7.5.Pregled namirnica i jela

7.6.Kraći kviz

8.Ažuriranje frižidera

8.1.Dodavanje namirnica

8.2.Ažuriranje namirnica

9.Izbor režima ishrane

9.1.Promena režima ishrane

9.2 Izbor režima ishrane od četiri ponuđena režima ishrane (ishrana sa malo secera ,vegetarijanska, proteinska i izbalansirana ishrana)

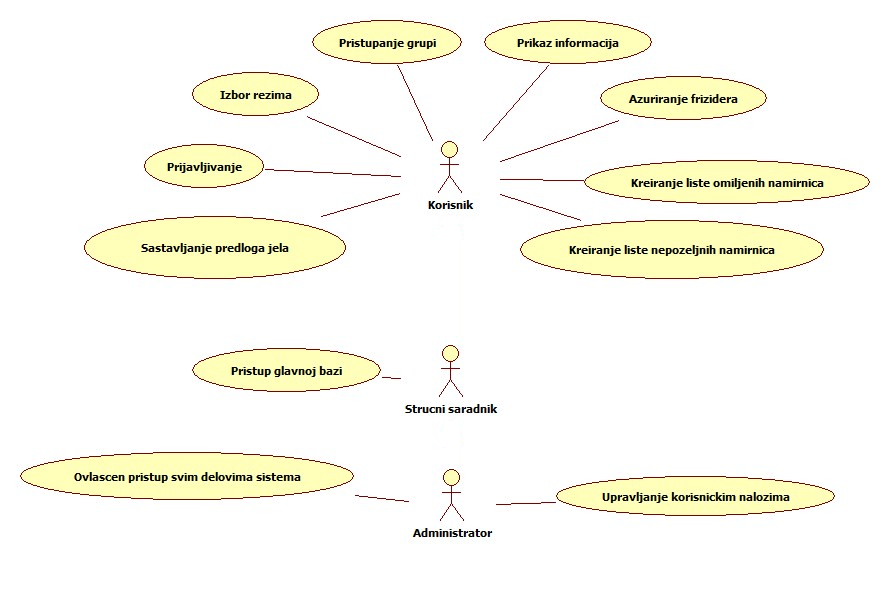
10.Pristup glavnoj bazi

10.1.Dodavanje namirnica / jela

10.2.Ažuriranje namirnica / jela

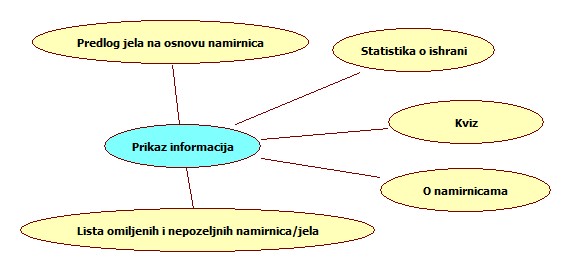
## Dijagrami slučajeva korišćenja

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

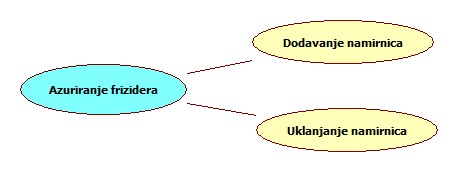


Slučajevi korišćenja: *prikaz informacija*, *ažuriranje frižidera*, *izbor režima, pristup glavnoj bazi* obuhvataju složenije radnje koje se mogu razložiti dalje razložiti na pojedinačne slučajeve korišćenja.

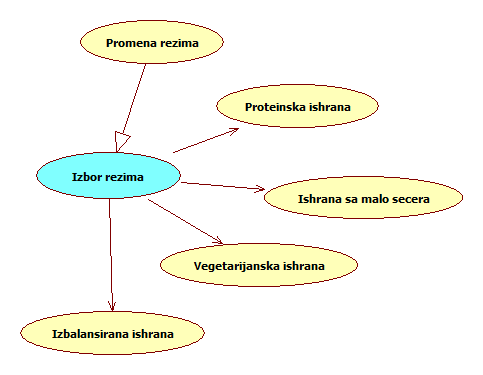
Detaljni UML dijagram za slučaj korišćenja *prikaz informacija* je prikazan na sledećoj slici:



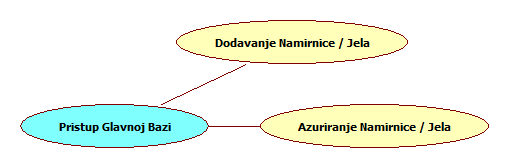
Detaljni UML dijagram za slučaj korišćenja *ažuriranje frižidera* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *izbor režima* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *pristup glavnoj bazi* je prikazan na sledećoj slici:



## Kratak opis slučajeva korišćenja

### Prijava

Kratak opis:Prijavljivanje korisnika u cilju pristupa aplikaciji.

Akteri :Korisnici aplikacije.

### Pristupanje grupi

Kratak opis:Korisniku se omogućava pristupanje grupi, odnosno pristupanje frižideru koji će koristiti više ljudi.

Akteri :Korisnici aplikacije.

### Ažuriranje frižidera

Kratak opis:Korisniku se daje mogućnost da dodaje namirnice (obavlja kupovinu). Takođe, korisnik može da ukloni namirnice iz frižidera, ukoliko je, primera radi, datoj namirnici istekao rok trajana. Korisnik se, dakle, ne prisiljava da mora da pojede sve što je uneseno u frižider.

Akteri:Korisnik.

### Sastavljanje predloga jela

Kratak opis:Korisniku se omogućava kreiranje sopstvenih jela na osnovnu namirnica iz glavne baze, a koja ne postoje u istoj.

Akteri:Korisnik.

### Kreiranje liste omiljenih namirnica

Kratak opis:Korisniku se omogućava da kreira listu omiljenih namirnica. Na taj način olakšaće asistentu predlaganje obroka.

Akteri:Korisnik.

### Kreiranje liste nepoželjnih namirnica

Kratak opis:Korisniku se omogućava da kreira listu nepoželjnih namirnica. Ovo se uvodi da bi asistent znao šta da ne uzima u obzir prilikom predlaganja jela, a pritom se nalazi u zajedničkom frižideru (to jelo je, primera radi, dodao drugi korisnik).

Akteri:Korisnik.

### Prikaz informacije

#### *Mehanizam koji nudi naredni obrok*

Kratak opis:Aplikacija nudi predloge na osnovu jela koja su sastavljena od strane korisnika, ali je osnovni preduslov da za ta jela trenutno u frižideru postoje sastojci

Akteri:Korisnik aplikacije.

#### *Kreiranje liste omiljenih namirnica*

Kratak opis:Korisniku se omogućava da kreira listu omiljenih namirnica. Na taj način olakšaće asistentu predlaganje obroka.

Akteri:Korisnik.

#### *Kreiranje liste nepoželjnih namirnica*

#### Kratak opis:Korisniku se omogućava da kreira listu nepoželjnih namirnica. Ovo se uvodi da bi asistent znao šta da ne uzima u obzir prilikom predlaganja jela, a pritom se nalazi u zajedničkom frižideru (to jelo je, primera radi, dodao drugi korisnik).

Akteri:Korisnik.

#### *Statistika o ishrani*

Kratak opis:Prikaz stranice sa grafičkim prikazom statistike o ishrani.

Akteri:Korisnik aplikacije.

#### *Pregled namirnica i jela*

Kratak opis:Pregled svih jela u bazi podataka i podataka o njima

Akteri:Korisnik aplikacije

#### *Kraći kviz*

Kratak opis:

U toku dana, korisnik će dobijati jedno ili više pitanja koja će sadržati korisne informacije u vezi zdrave ishrane. Cilj ovih pitanja je da korisnik svakog dana nauči nešto novo i da bolje sastavlja predloge jela.

Akteri:Korisnik aplikacije

### Ažuriranje frižidera

Kratak opis:Korisniku se daje mogućnost da dodaje namirnice (obavlja kupovinu). Takođe, korisnik može da ukloni namirnice iz frižidera, ukoliko je, primera radi, datoj namirnici istekao rok trajana. Korisnik se, dakle, ne prisiljava da mora da pojede sve što je uneseno u frižider.

Akteri:Korisnik.

### Izbor režima ishrane

#### *Promena režima ishrane*

Kratak opis:Korisnik može da promeni režim ishrane po želji.

Akteri:Korisnik.

#### *Izbor režima ishrane od 4 ponuđena režima*

Kratak opis:Izbor režima ishrane prilikom prvog prijavljivanja.

Akteri:Korisnik.

### Pristup bazi podataka

#### *Dodavanje namirnice / jela*

Kratak opis:

Unos novih namirnica i jela u glavnu bazu podataka.

Akteri:Stručni saradnik, Administrator.

#### Ažuriranje namirnice / jela

Kratak opis:Izmena podataka o namirnicama i jelima iz glavne baze podataka.

Akteri:Stručni saradnik, 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.

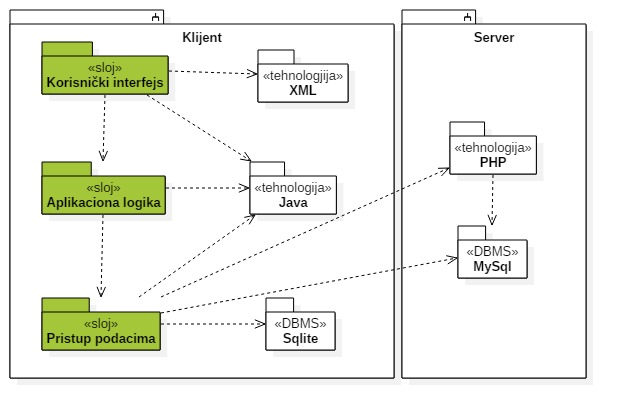
Logički pogled na Food Assistant aplikaciju obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket *Korisnički interfejs* sadrži XML datoteke za kreiranje korisničkog interfejsa na Android platformi kao i dizajn i forme preko kojih korisnici sistema komuniciraju sa sistemom.

Paket *Aplikaciona logika* predstavlja celokupnu logiku sistema realizovanu u programskom jeziku Java.

Paket *Pristup podacima* sadrži Java klase koje predstavljaju interfejs za pristup, dodavanje i ažuriranje podataka koji se čuvaju u bazi podataka. Ovaj paket se sastoji od dva podpaketa, jedan sadrži java klase za pristup serveru , ove klase se koriste za slanje zahteva i prihvatanje podataka od servera, dok drugi sadrži Java klase za pristup serveskoj bazi podataka.

## Pregled arhitekture – organizacija paketa i podsistema u slojeve



### Korisnički interfejs

Sloj

Ovaj sloj realizuje korisnički interfejs aplikacije. U njemu su sadržani svi XML fajlovi koji generišu korisnički interfejs i ostali elemeniti koji učestvuju u kreiranju korisničkog interfejsa,kao i forme preko kojih korisnici komuniciraju sa sistemom.

Sloj korisničkog interfejsa zavisi od sloja aplikacione logike, kao i podržanog XML-a.

### Aplikaciona logika

Sloj

Sloj aplikacione logike sadrži Java klase koje realizuju funkcionalnost celog sistema i uspostavljaju vezu između korisničkog interfejsa i sloja za pristup podacima.

Ovaj sloj zavisi od biblioteka klasa iz Jave koje pruža Android razvojno okruženje (u ovom slučaju Android Studio).

### Pristup podacima

Sloj

Sloj za pristup podacima se nalazi na dnu troslojne arhitekture i sadrži Java klase zadužene za pribavljanje, dodavanje i ažuriranje podataka koji se čuvaju u bazi podataka (na serveru i u lokalnoj bazi) .

Ovaj sloj ne zavisi od drugih slojeva, ali je zavisan od paketa SQLLite baza podataka i od biblioteka klasa koje pruža Android Studio, kao i od biblioteka koje se koriste za komunikaciju sa serverom.

### XML

Tehnologija

XML tehnologija je zadužena za definisanje statičkog dela korisničkog interfejsa kod Android programiranja.

### Java

Tehnologija

Osnovu programiranja za Android platformu predstavlja programski jezik Java. Java tehnologija obezbeđuje funkcionalnost celog sistema. Klase biblioteka koje pruža Java mogu da dinamički generišu delove korisničkog interfejsa. Ovaj programski jezik je takođe potpuno nezavistan od konkretnih sistema za upravljanje bazama podataka čime se uz obezbeđivanje dogovarajućih upravljačkih programa omogućava povezivanje aplikacije sa konkretnim DBMS sistemima i preslikavanje opštih Java naredbi u naredbe konkretnog DBMS-a.

### MySQL

DBMS

MySQL predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju serverske baze podataka gde će se čuvati svi podaci vezani za aplikaciju.

### SQLite

DBMS

SQLite predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju lokalne baze podataka na korisnikovom uređaju gde će se čuvati svi podaci vezani lično za korisnika. Tim informacijama će korisnik moći da pristupi kada nema pristup Internetu.

### PHP

Tehnologija

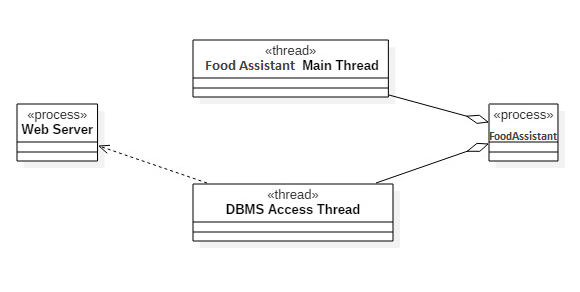
Tehnologija PHP-a obezbeđuje mehanizam za pisanje i izvršavanje skripti na strani servera. Ove skripte pristupaju bazi podataka na serveru u cilju pribavljanja, unosa i ažuriranja podataka.

# 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.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju izvršenju Food Assistant aplikacije. Dijagram je opšteg tipa i može se primeniti na bilo koju Android aplikaciju zasnovanu na klijent-server modelu koja koristi PHP-u i MySQL bazu podataka, ali koristi i Sqlite DBMS za upravljanje lokalnom bazom podataka na korisnikovom uređaju.



### FoodAssistant

Proces Food Assistant izvršava glavnu funkcionalnost aplikacije. Pri startovanju aplikacije , ovaj proces kreira glavnu nit.

### FoodAssistant Main Thread

Pri startovanju aplikacije sistem kreira jednu nit u kojoj sve komponente aplikacije rade. Uloga ove niti je rukovanje korisničkim interfejsom u pogledu događaja i interakcije sa pogledima (views) u okviru korisničkog interfejsa.

### DBMS Access Thread

Uloga ove niti je komunikacija sa serverom, odnosno rukovanje prilikom slanja zahteva serveru i pribavljanja podataka.

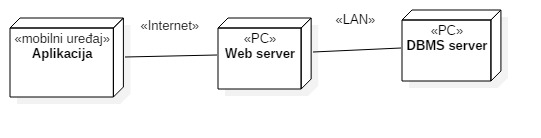
### Web Server

Proces *Web Server* komunicira sa Food Assistant aplikacijom u pogledu prihvatanja zahteva i razmene podataka. Takođe, obavlja sve funkcionalnosti koje zahtevaju pristup bazi podataka.

# 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 Food Assistant aplikacije.



## Klijent aplikacija

Pristup Food Assistant aplikaciji se obavlja preko mobilnih uređaja koji rade na Android platformi. Za povezivanje između klijentske aplikacija i Web servera koristi se Internet infrastruktura tako da nema ograničenja u pogledu lokacije klijenta.

## Web server

Računar na kome se izvršava Web server može da opslužuje više klijenata koji pristupaju preko Interneta. Na ovom računaru se izvršava osnovni proces koji realizuje funkcionalnost Web servera obradom datih PHP skripti. 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

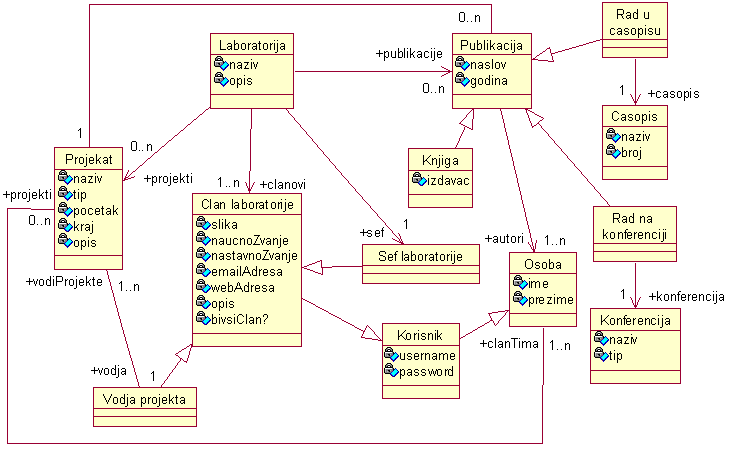
# Pogled na implementaciju sistema

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

## Model domena

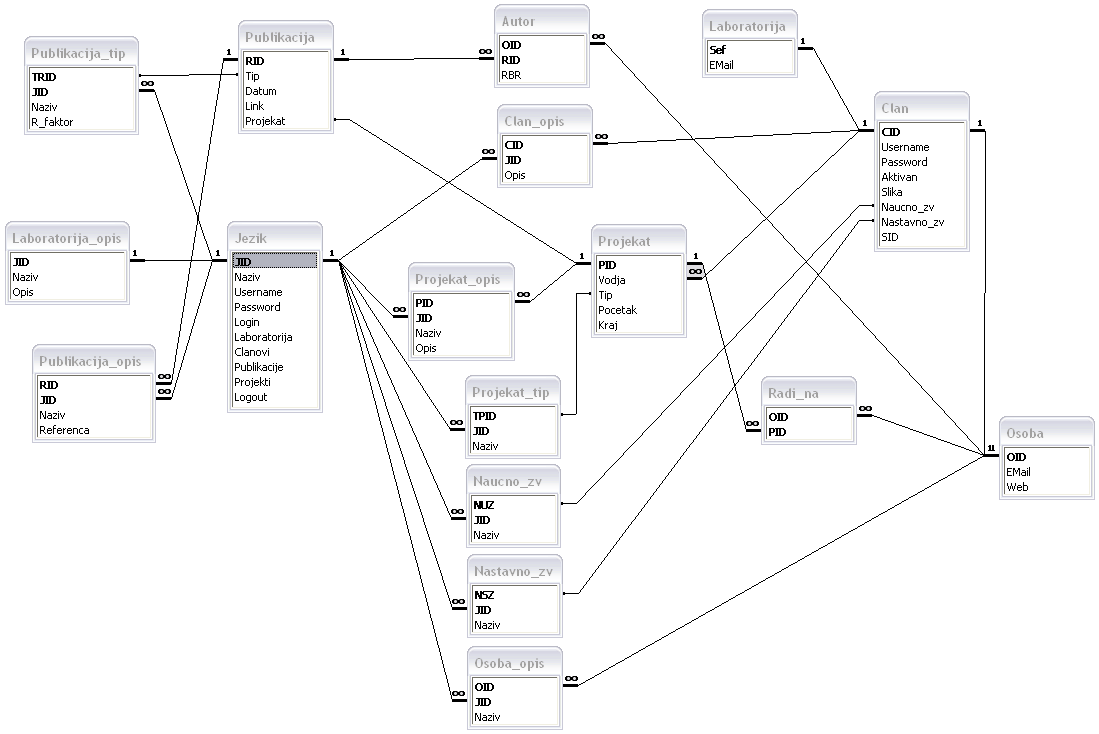
Model domena za koji se PeNcIL portal projektuje je ilustrovan UML dijagramom klasa. U njemu su prikazane domenske klase, neki od njihovih atributa, kao i veze koje se mogu identifikovati između njih.

Model domena predstavlja osnovu za projektovanje baze podataka, ali i identifikaciju nekih od komponenti (PHP skripti) koje će biti implementirane.



## Šema baze podataka

Detaljna šema baze podataka je prikazana na sledećem dijagramu. Baza podataka i dijagram su kreirani korišćenjem *MS Access*-a, dok je migracija na MySQL obavljena pomoću *MySQL Migration Toolkit*-a.



## Komponente sistema

*NAPOMENA: PeNcIL projekat nije kompletno implementiran, tj. implementiran je na nivou arhitekturnog prototipa. Zbog ovoga sledeću sekciju trebate shvatiti kao ilustraciju kako bi dokument trebao da izgleda. Opis komponenti sistema za vaše projekte mora biti kompletan! Procena složenosti vašeg rešenja će zavisiti od opisa sistema koji ovde date.*

Komponente sistema PeNcIL portala su PHP skripti čiji će pregled biti dat po arhitekturnim slojevima. Za ilustraciju će biti korišćeni UML dijagrami komponenti, ali i dijagrami klasa. U slučajevima gde je PHP skript prikazan kao klasa atributi predstavljaju ulazne podatke koji se uzimaju iz GET ili POST dela HTTP poruke, dok metodi predstavljaju funkcije definisane u okviru skripta.

### Komponente korisničkog interfejsa

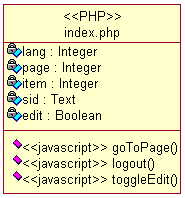
Dizajn korisničkog interfejsa je obuhvaćen dvema komponentama:



Komponenta **index.php** je implementira stranicu portala čiji sadržaj može da varira od parametra koji joj se proslede u zahtevu.

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

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

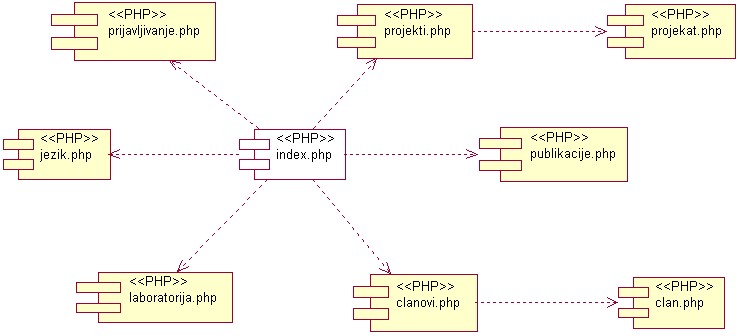


Značenje atributa je sledeće:

* lang – identifikator jezika (JID iz tabele Jezik)
* page – interni identifikator stranice (o laboratoriji, članovi, publikacije, projekti)
* item – identifikator stavke koja se detaljno prikazuje na stranici (određeni član ili projekat)
* sid – identifikator sesije kada je korisnik ulogovan
* edit – definiše da li je uključeno uređivanje

### Komponente aplikacione logike

Komponente koje realizuju domen problema se uključuju isključivo preko **index.php** 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:



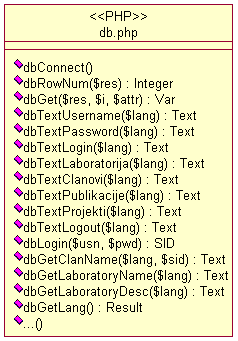
... opis svih komponenti sa dijagrama sa detaljnim dijagramima po potrebi.

### Komponente za pristup podacima

Pristup bazi podataka je u potpunosti zatvoren u funkcije koje su definisane u okviru PHP skripta **db.php**. Pomenuti skript se uključuje na početku index.php-a, 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:

* dbConnect – povezivanje na bazu podataka, poziva se u samom skriptu
* dbRowNum – vraća broj slogova sadržanih u rezultatu upita
* dbGet – vraća vrednost polja iz rezultata upita
* dbText... – vraća vrednost pojedinih konstanti koje se definišu uz jezik
* dbLogin – loguje korisnika i vraća SID
* dbGetClanName – vraća ime člana po jeziku i SID-u
* dbGetLaboratoryName – vraća naziv laboratorije za zadati jezik
* dbGetLaboratoryDesc – vraća opis laboratorije za zadati jezik
* dbGetLang – vraća rezultat upita nad jezicima definisanim u bazi podataka
* ... – ostale funkcije

# 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 1000 simultanih pristupa korisnika portalu.
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. PeNcIL portal 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.