Adiutor

Portal za edukaciju

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 27.4.2016 | 1.0 | Konačna verzija | Nikola Ranković, Marko Radosavljević, Miloš Mladenović, Predrag Nikolić |
|  |  |  |  |
|  |  |  |  |

[1. Cilj dokumenta 6](#_Toc455436050)

[2. Opseg dokumenta 6](#_Toc455436051)

[3. Reference 6](#_Toc455436052)

[4. Predstavljanje arhitekture 6](#_Toc455436053)

[5. Ciljevi i ograničenja arhitekture 6](#_Toc455436054)

[6. Pogled na slučajeve korišćenja 7](#_Toc455436055)

[6.1 Dijagrami slučajeva korišćenja 7](#_Toc455436056)

[6.2 Kratak opis slučajeva korišćenja 10](#_Toc455436057)

[7. Pogled na logičku arhitekturu sistema 13](#_Toc455436058)

[7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 14](#_Toc455436059)

[7.1.1 Korisnički interfejs i obrada 14](#_Toc455436060)

[7.1.2 Pristup podacima 14](#_Toc455436061)

[7.1.3 HTML 15](#_Toc455436062)

[7.1.4 JavaScript 15](#_Toc455436063)

[7.1.5 .NET 15](#_Toc455436064)

[7.1.6 Oracle DB 15](#_Toc455436065)

[8. Pogled na procese 15](#_Toc455436066)

[16](#_Toc455436067)

[8.1 Procesi 16](#_Toc455436068)

[8.1.1 Web čitač 16](#_Toc455436069)

[8.1.2 Web server 16](#_Toc455436070)

[8.1.3 Oracle Server 16](#_Toc455436071)

[9. Pogled na raspoređivanje sistema 16](#_Toc455436072)

[9.1 Klijent 17](#_Toc455436073)

[9.2 Web server 17](#_Toc455436074)

[9.3 DBMS server 17](#_Toc455436075)

[10. Pogled na implementaciju sistema 17](#_Toc455436076)

[10.1 Model domena 17](#_Toc455436077)

[10.2 Šema baze podataka 19](#_Toc455436078)

[10.3 Komponente sistema 19](#_Toc455436079)

[10.3.1 Komponente korisničkog interfejsa i obrade 20](#_Toc455436080)

[10.3.2 Komponente za pristup podacima 23](#_Toc455436081)

[11. Performanse 26](#_Toc455436082)

[12. Kvalitet 26](#_Toc455436083)

Sadržaj

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture Adiutor portala.

# Opseg dokumenta

Dokument se odnosi na Web portal za pomoć pri edukaciji „Adiutor“, koji će biti razvijen od strane A4 Team-a. „Adiutor“ na latinskom znači pomoćnik/asistent. Namena sistema je, pre svega:

1. Efikasno kreiranje, modifikovanje i organizovanje predmeta i oblasti iz svakog predmeta, a u skladus a tim i pitanja i zadataka sa kojima se studenti susreću.
2. Pružanje funkcionalnosti ocenjivanje svakog komentara i odgovora na određeni zadati problem, a samim tim i podizanje svesti o relevantnosti svakog odgovora, ali i korisnika koji pružaju odgovore.
3. Pružanje podrške za redovno i blagovremeno ažuriranje bitnih informacija i dodavanje novih predmeta, kao i brisanje i samo-uklanjanje odgovora koji su pogrešni.
4. Pružanje funkcionalnosti za pretragu opštih tema nevezanih za konkretan kurs na osnovu ponuđenih tagova. Dodavanje novih tagova u bazen tagova, koje bi administrator trebalo da odobri.

# Reference

Spisak korišćene literature:

1. Adiutor – Predlog projekta - V1.0 - 2016 – A4 Team
2. Adiutor – Planirani raspored aktivnosti na projektu - V1.0 - 2016 – A4 Team
3. Adiutor – Plan realizacije projekta - V1.0 - 2016 – A4 Team
4. Adiutor – Vizija sistema - V1.0 - 2016 – A4 Team
5. Adiutor – Specifikacija zahteva - V1.0 - 2016 – A4 Team

# 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
* 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. Portal Adiutor će biti implementiran kao Web aplikacija zasnovana na .NET tehnologiji i Oracle bazi podataka [4].
2. Klijentski deo Adiutor portala će biti optimizovan za sledeće Web čitače[4]:
   * Google Chrome
   * FireFox (Mozilla)
   * Microsoft Edge
   * Opera
3. Svi zahtevi u pogledu performansi dati u [5] moraju biti uzeti u obzir pri izboru arhitekture i razvoju sistema.

# Pogled na slučajeve korišćenja

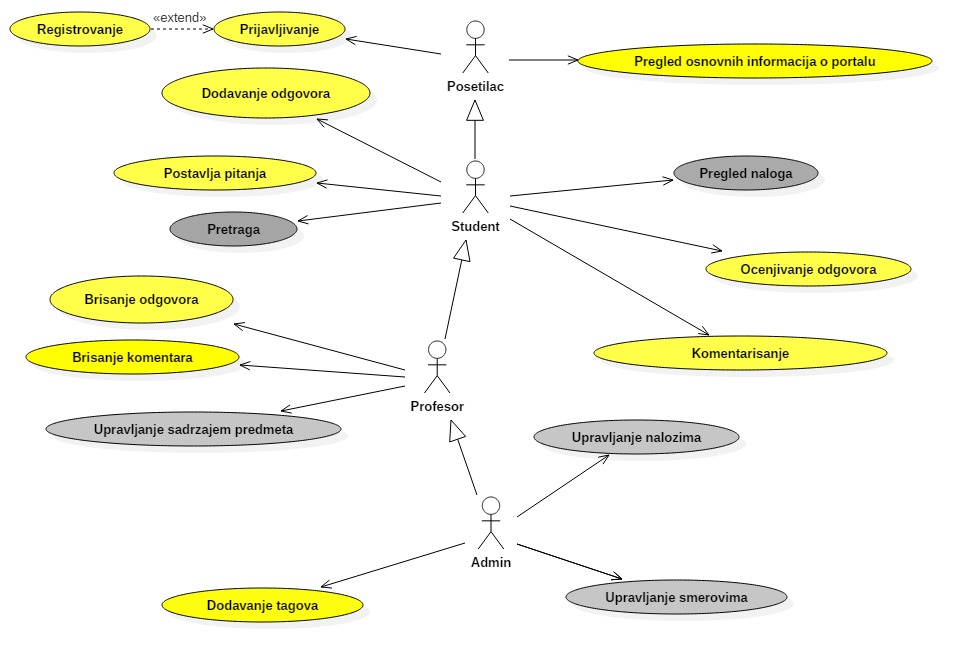
U ovom odeljku je dat pogled na slučajeve korišćenja definisane u specifikaciji zahteva [5].

Slučajevi korišćenja portala Adiutor su

* + - Prijavljivanje
    - Registrovanje
    - Pregled osnovnih informacija o portalu
    - Postavlja pitanje
    - Postavlja odgovor
    - Ocenjuje odgovor
    - Osnovni podaci o korisniku
    - Lista urađenih zadataka
    - Lista postavljenih pitanja
    - Prosečni rejting
    - Komentarisanje
    - Pretraga po tagovima
    - Pretraga naloga
    - Brisanje odgovora
    - Brisanje komentara
    - Dodavanje literature
    - Dodavanje oblasti
    - Ažuriranje oblasti
    - Brisanje oblasti
    - Dodeljivanje profesora
    - Dodavanje predmeta
    - Brisanje predmeta
    - Dodavanje smera
    - Brisanje smera
    - Pregled liste naloga
    - Promena tipa naloga
    - Prihvatanje zahteva za registraciju
    - Dodavanje tagova

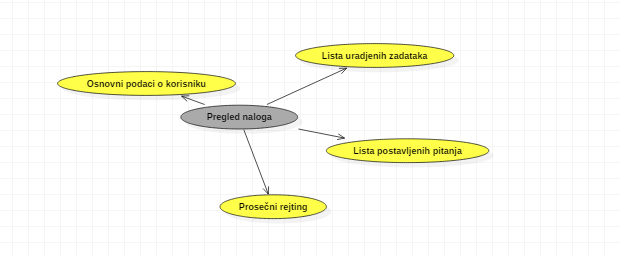
## Dijagrami slučajeva korišćenja

U nastavku su dati UML dijagrami korišćenja za projekat „Adiutor“. Bitno je naglasiti da su sivom bojom označeni složeniji slučajevi korišćenja, a žutom bojom jednostavniji (atomični) slučajevi korišćenja koji su detaljno objašnjeni. Osnovni UML dijagram koji prikazuje korisnike i slučajeve korišćenja Adiutor portala prikazan je na sledećoj slici:

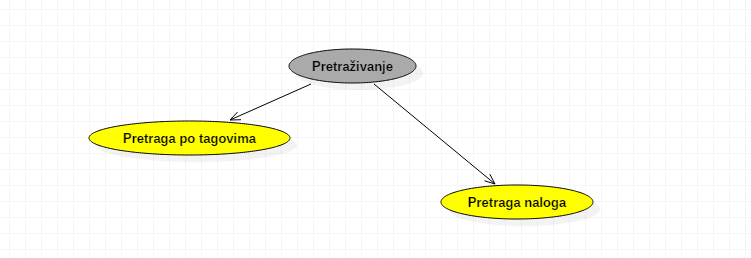


Slučajevi korišćenja *Pregled naloga, Pretraživanje, Upravljanje nalozima, Upravljanje smerovima,* *Upravljanje sadržajem predmeta* obuhvataju složenije radnje koje se mogu razložiti dalje razložiti na pojedinačne slučajeve korišćenja.

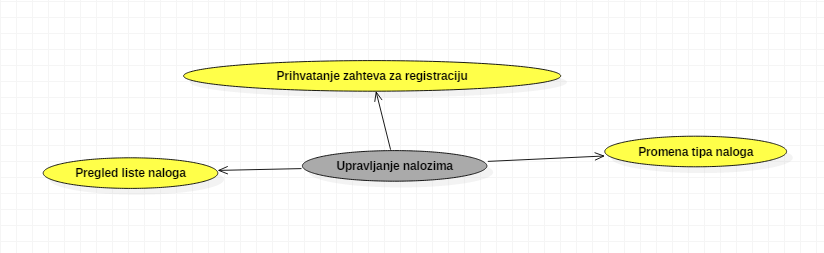
Detaljni UML dijagram za slučaj korišćenja *Pregled naloga* je prikazan na sledećoj slici:



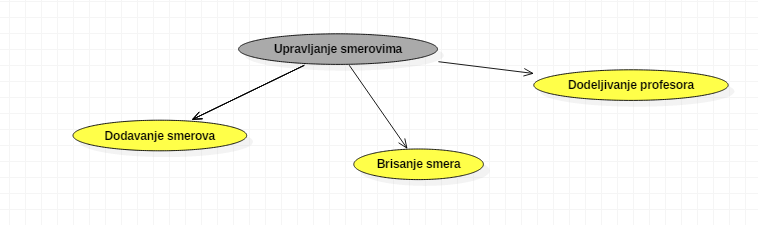
Detaljni UML dijagram za slučaj korišćenja *Pretraživanje* je prikazan na sledećoj slici:



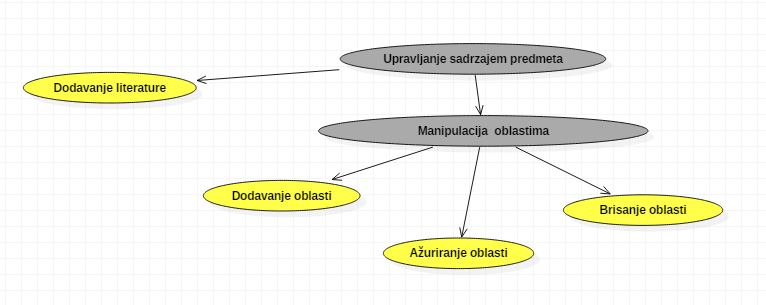
Detaljni UML dijagram za slučaj korišćenja *Upravljanje nalozima* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *Upravljanje smerovima* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *Upravljanje sadržajem predmeta* je prikazan na sledećoj slici:



## Kratak opis slučajeva korišćenja

1. Prijavljivanje

Kratak opis:

Korisnici se prijavljuju na portal kako bi mogli da pristupe zaštićenom materijalu.

Akteri:

Posetilac, Student, Profesor, Admin

1. Registrovanje

Kratak opis:

Posetioci koji nemaju nalog pre prijavljivanja moraju da se registruju.

Akteri:

Posetilac

1. Pregled osnovnih informacija o portalu

Kratak opis:

Svi korisnici imaju pravo pregleda osnovnih informacija o portalu.

Akteri:

Posetilac, Student, Profesor, Admin

1. Postavlja pitanje

Kratak opis:

Postavljanje pitanja na koje nema odgovora

Akteri:

Student, Profesor, Admin

1. Postavlja odgovor

Kratak opis:

Postavljanje odgovora na konkretno pitanje

Akteri:

Student, Profesor, Admin

1. Ocenjuje odgovor

Kratak opis:

Dodavanje ocene odgovoru na određeno pitanje.

Akteri:

Student, Profesor, Admin

1. Osnovni podaci o korisniku

Kratak opis:

Osnovne informacije koje karakterišu svakog korisnika (ime, prezime, godina rođenja i sl.).

Akteri:

Student, Profesor, Admin

1. Lista urađenih zadataka

Kratak opis:

Lista zadataka koje je uradio određeni korisnik.

Akteri:

Student, Profesor, Admin

1. Lista postavljenih pitanja

Kratak opis:

Lista pitanja koja je postavio određeni korisnik.

Akteri:

Student, Profesor, Admin

1. Prosečni rejting

Kratak opis:

Prosečna ocena na komentare koje je postavio određeni korisnik.

Akteri:

Student, Profesor, Admin

1. Komentarisanje

Kratak opis:

Dodavanje kometara (odgovora) na pitanje koje je postavio drugi korisnik.

Akteri:

Student, Profesor, Admin

1. Pretraga po tagovima

Kratak opis:

Vrši se pretraga po dostipnim tagovima.

Akteri:

Student, Profesor,Admin

1. Pretraga naloga

Kratak opis:

Vrši se pretraga po odgovarajućem korisničkom imenu.

Akteri:

Student, Profesor, Admin

1. Brisanje odgovora

Kratak opis:

Uklanjanje odgovora na pitanje koji je postavio drugi korisnik.

Akteri:

Profesor, Admin

1. Brisanje komentara

Kratak opis:

Uklanjanje kometara na pitanje koji je postavio drugi korisnik.

Akteri:

Profesor, Admin

1. Dodavanje literature

Kratak opis:

Dodaje literaturu vezanu za neku oblast.

Akteri:

Profesor, Admin

1. Dodavanje oblasti

Kratak opis:

Dodaje oblast vezanu za neki predmet.

Akteri:

Profesor, Admin

1. Ažuriranje oblasti

Kratak opis:

Ažurira oblast vezanu za neki predmet.

Akteri:

Profesor, Admin

1. Brisanje oblasti

Kratak opis:

Briše oblast vezanu za neki predmet.

Akteri:

Profesor, Admin

1. Dodeljivanje profesora

Kratak opis:

Nekom kursu se dodeljuje profesor koji je zadužen za taj kurs.

Akteri:

Admin

1. Dodavanje predmeta

Kratak opis:

Dodavanje novog predmeta

Akteri:

Admin

1. Brisanje predmeta

Kratak opis:

Brisanje postojećeg predmeta

Akteri:

Admin

1. Dodavanje smera

Kratak opis:

Dodavanje novog smera.

Akteri:

Admin

1. Brisanje smera

Kratak opis:

Brisanje postojećeg smera.

Akteri:

Admin

1. Pregled liste naloga

Kratak opis:

Prikazivanje liste korisničkih naloga odgovarajuceg tipa (studentski, profesorski, i admin nalozi).

Akteri:

Admin

1. Promena tipa naloga

Kratak opis:

Unapređivanje naloga iz npr. studentskog u profesorski ili iz profesorski u admin i slično.

Akteri:

Admin

1. Prihvatanje zahtava za registraciju

Kratak opis:

Korisnik obrađuje zahteve za registraciju novih korisnika

Akteri:

Admin

1. Dodavanje tagova

Kratak opis:

Dodavanje novog taga.

Akteri:

Admin

# Pogled na logičku arhitekturu sistema

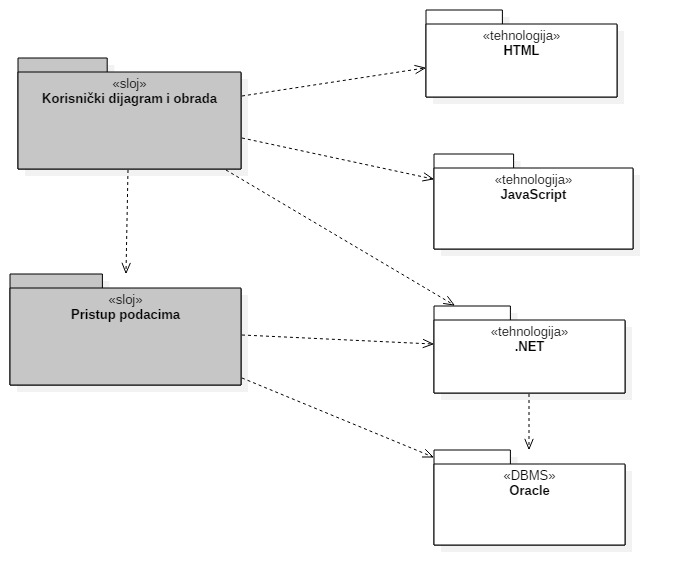
U ovom odeljku je dat pregled logičke arhitekture sistema. Ovaj pogled sadrži opis najznačajnijih klasa, njihove organizacije u pakete i podsisteme, i organizacija podsistema u slojeve. U cilju opisivanja dinamičkih aspekata arhitekture, ovaj odeljak može da uključi opise realizacije najznačajnijih slučajeva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

Logički pogled na Adiutor portal obuhvata 2 glavna paketa: *Korisnički interfejs i obrada* i *Pristup podacima.*

Paket *Korisnički interfejs i obrada*  sadrži Web stranice, .NET kontrolere za obradu i JavaScript skripte koje realizuju grafički dizajn i forme preko kojih korisnici sistema komuniciraju sa sistemom. Ovaj paket takodje sadrzi .NET kontrolere i JavaScript skripte koje vrse obradu podataka i realizuju funkcionalnosti specificne za domen sistema koji se razvija.

Paket *Pristup podacima* sadrži .NET kontrolere koji predstavljaju osnovne funkcije za rad sa bazom podataka.

## Pregled arhitekture – organizacija paketa i podsistema u slojeve



### Korisnički interfejs i obrada

sloj

Ovaj sloj realizuje korisnički interfejs portala tj. njegov glavni deo koji je stalno prisutan.

Sadrži .NET kontrolere, JavaScript fajlove i HTML stranice koje realizuju najveći deo interfejsa portala.

Sadrži i .NET kontrolere koje realizuju funkcionalnost karakterističnu za domen primene portala i uspostavljaju vezu između korisničkog interfejsa i sloja za pristup podacima. Sadrži i JavaScript skripte koje vrše obradu podataka.

Ovaj sloj zavisi od sloja za pristup podacima, kao i od paketa HTML, JavaScript i .NET.

### Pristup podacima

sloj

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

Ovaj sloj ne zavisi od drugih slojeva, ali je zavisan od kontrolera .NET i Oracle baza podataka.

### HTML

tehnologija

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.

### JavaScript

tehnologija

Tehnologija JavaScript obezbeđuje mehanizam za dodavanje dinamičkih elemenata Web stranici. Takođe, preko nje je moguće slati i primati podatke u JSON obliku od servera bez osvežavanja stranice.

### .NET

tehnologija

.Net obezbeđuje mehanizam za programiranje na strani servera. Bazne klase pružaju širok spektar mogućnosti, uključujući korisnički interfejs, pristup podacima, bazama, kriptografija, razvoj veb-aplikacija, numerički algoritmi i mrežne komunikacije. Biblioteke klasa se koriste od strane programera, koji ga kombinuju sa svojim kodom za izradu aplikacija.

### Oracle DB

DBMS

Oracle DB predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju portala Adiutor.

# 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 .NET-u imaju relativno jednostavan procesni model koji je u potpunosti pod kontrolom Web servera. Sa stanovišta projektanta .NET Web aplikacije nije potrebno voditi računa o načinu rada Web servera.

Ilustracije radi u nastavku je dat opis procesa uključenih u izvršenje portala Adiutor kao Web aplikacije.

## 

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju Adiutor portala.

### Web čitač

Web čitač je proces koji izvršava funkcionalnost aplikacije za prikaz HTML stranica i izvrsavanje JavaScript scripti dobijenih od nekog Web servera

Web čitač zavisi od Web servera koji generiše i vraća odgovarajuću HTML stranicu sa JavaScript skriptima na zahtev.

### Web server

Web server je proces koji izvršava funkcionalnost opsluživanja zahteva prispelih sa više Web čitača. Ukoliko je zahtevana stranica .NET kontroler, Web server inicira izvršenje odgovarajućeg skripta i generiše sadržaj koji se vraća čitaču. Web server može paralelno da inicira izvršenje većeg broja .NET kontrolera.

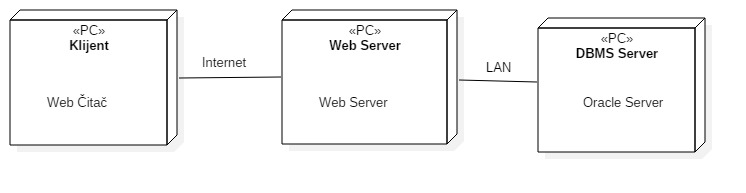
### Oracle Server

Oracle Server je proces koji izvršava funkcionalnost Oracle sistema za upravljanje bazama podataka. Ovaj proces može konkurentno da prihvati određen broj upita, izvrši ih nad bazom podataka i vrati rezultate Web serveru (tj. .NET kontroleru) kome su bili potrebni.

# 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 portala Adiutor.



## Klijent

Pristup Adiutor portalu se obavlja preko bilo kog uređaja na kome može da se izvršava Web čitač. 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

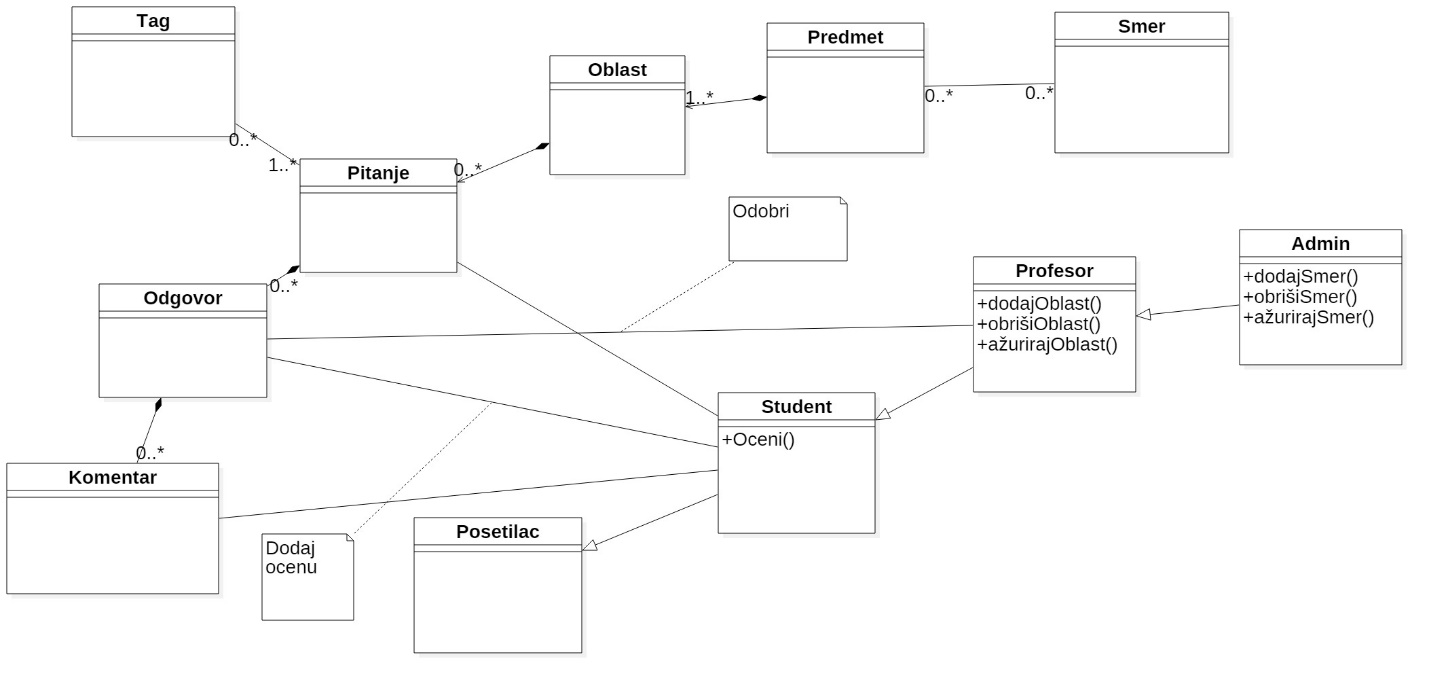
DBMS server je računar na kome se izvršava Oracle Server proces koji realizuje funkcionalnost sistema za upravljanje bazama podataka. Zbog efikasnosti ali i sigurnosti podataka koji se na ovom računaru čuvaju pristup bazi je ograničen samo na računare iz lokalne mreže (LAN).

# Pogled na implementaciju sistema

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

## Model domena

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



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

## Šema baze podataka

Detaljna šema baze podataka je prikazana na sledećem dijagramu. Baza podataka i dijagram su kreirani korišćenjem alata *Sql Manager 2012,* baza je kreirana na lokalnom serveru za potrebe testiranja.

## Komponente sistema

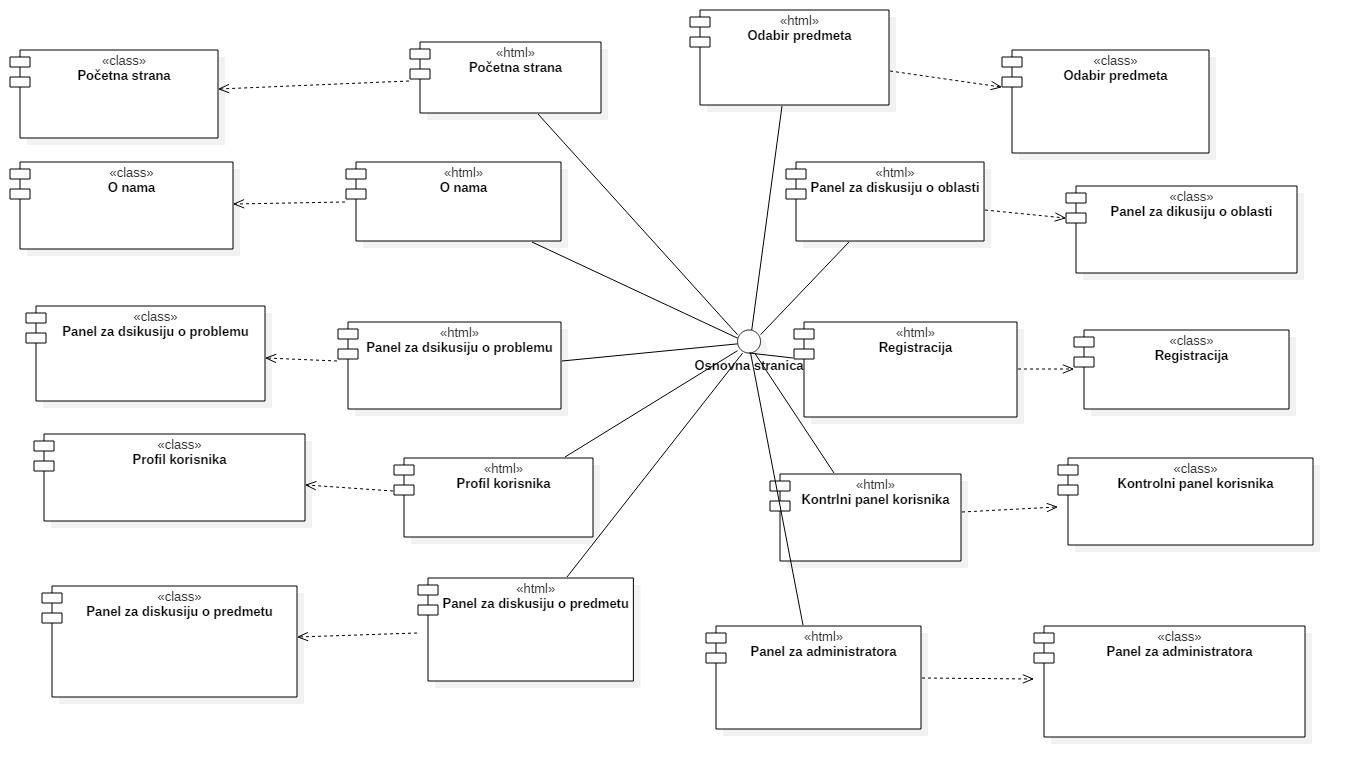
Komponente sistema Adiutor portala su .NET klase (kontroleri) čiji će pregled biti dat po arhitekturnim slojevima. Za ilustraciju će biti korišćeni UML dijagrami komponenti, ali i dijagrami klasa. U delu gde se detaljnije opisuju komponente, trebalo bi da stoje UML dijagrami klasa, ali zbog velikog broja metoda oni su izostavljeni. Atributi predstavljaju ulazne podatke koji se uzimaju iz GET ili POST dela HTTP poruke, dok metodi predstavljaju funkcije definisane u okviru klase.

Podaci o korisniku portala se pamte u promenljivama sesije. I to:

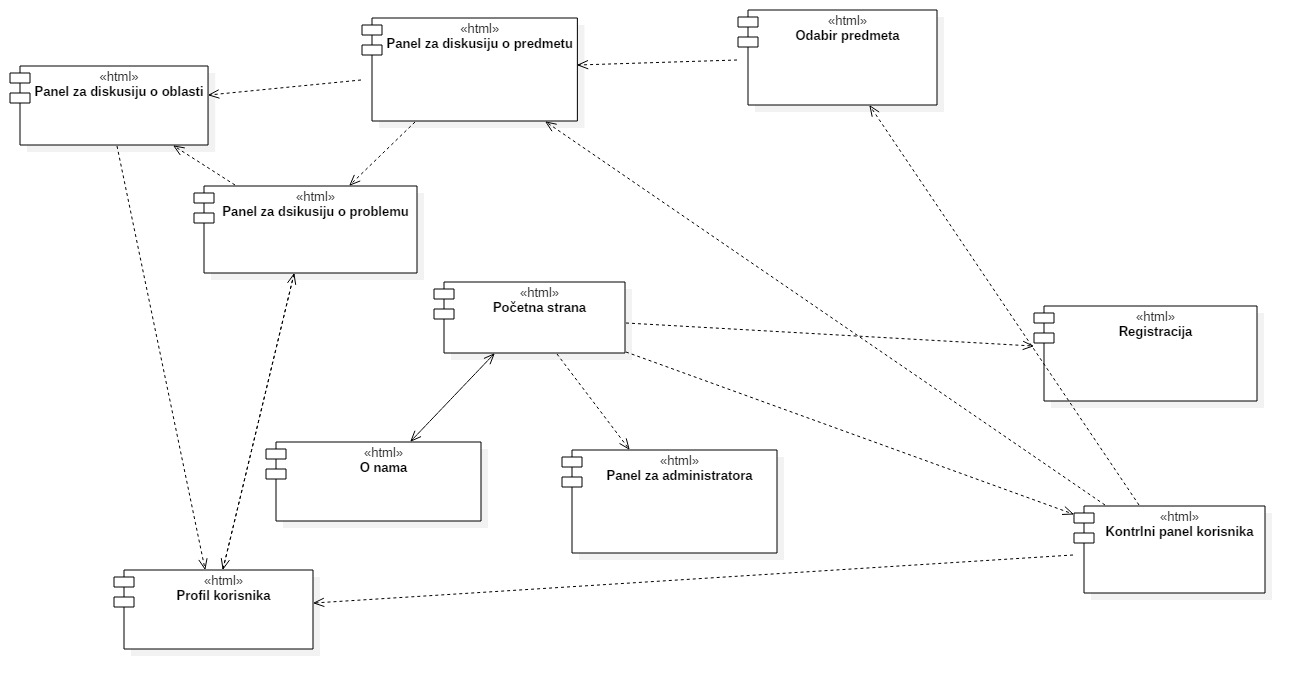
* Session[”Id”]-broj koji oznacava id korisnika u bazi podataka
* Session[“Korisnik”]-niz koji sadrži osnovne podatke o korisniku (rola, status, ime, przime, username, password, godina studija).

### Komponente korisničkog interfejsa i obrade

Ovaj sloj obuhvata sledeće komponente:



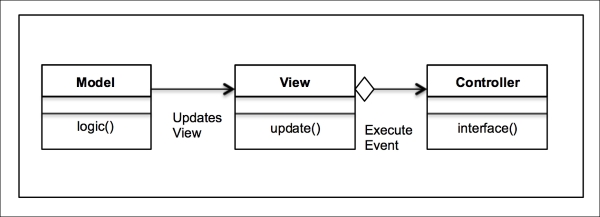
Relacija zavisnosti označava da komponenta direktno koristi drugu (sa stranica se zovu kontroleri).



Relacija zavisnosti bez stereotipa označava da su komponente međusobno povezane, tj. da se iz prve može preci na drugu komponentu (stranicu portala).

Interfejs **Osnovna stranica** predstavlja apstrakciju bilo koje stranice portala. Ona sadrži **header** i **footer** koji će stalno biti prisutni u prikazu. Stilovi se preuzimaju iz bootstrap-a i uz dodatna pravila iz fajla **mojstil.css.** Za responsive dizajn se koristi bootstrap biblioteke.

U nastavku je dat opis najvažnijih funkcija ostalih komponenti. Ostale komponete su realizovane kao MVC komponente. Statoje se iz tri dela: view (HTML stranica), controller (klasa koja se sastoji iz funkcija zaduženih za obradu akcija na view) i model (klasa koja opisuje model koji služi za prenos podataka između kontrolera i pogleda).



U nastvaku je dat pregled logike ostalih komponenti (kontroleri i funkcionalnosti koje se nalaze u njima).

**Početna strana:**

.Net kontroler zadužen za prikaz početne strane i forme za logovanje.

Funkcije:

Index() – funkcija koja vraća HTML početnu stranicu.

Login(LoginModel log) – funkcija koja proverava validnost unetih podataka kod logovanja i vraća *kontrolni panel korisnika.*

About() – funkcija koja vraća HTML stranicu sa podacima o autorima aplikacije – O nama

Contact()- funkcija koja vraća HTML stranicu sa podacima o kontaktu

LogOut() – funkcija koja služi da izloguje korisnika

**Panel za diskusiju o problemu:**

.Net kontroler za prikaz konkretnog problema, odgovora i komentara.

Funkcije:

Discussion(int id) – vraća HTML stranicu i određenu diskusiju sa svim odgovorima i komentarima, preko vrednosti id diskusije

AddReply(OdgovorModel odg) – dodaje odgovor navedenoj diskusiji

AddComment(KomentarModel kom) – dodaje komentar navedenom odgovoru

Plus(OdgovorModel odg) – dodaje pozitivnu ocenu odgovoru

Minus(OdgovorModel odg) – dodaje negativnu ocenu odgovoru

Approve(OdgovorModel odg) – omogućava profesoru odobravanje odgovora

UserProfile(int id) – vraća stranicu sa podacima o korisniku

SubjectField(int id) – vraća stranicu sa navedenom oblašću

RemoveReply(OdgovorModel odg) – omogućava brisanje određenog odgovora.

RemoveComment (OdgovorModel odg) – omogućava brisanje određenog komentara.

**Panel za diskusiju o predmetu:**.Net kontroler koji služi za prikaz oblasti na koje je podeljen svaki predmet, osnovnih informacija i literature vezane za isti.

Funkcije:  
SubjectDiscussion(int id) – vraća HTML stranicu predmeta sa osnovnim podacima, oblastima i literaturom

FieldDiscussion (int id) – poziva konroler zadužen za prikaz diskusije o konkretnoj oblasti.

UpdateInfo (PredmetModel prm) – ažurira podatke o predmetu.

AddSubjectField (OblastModel om) - omogućava dodavanje nove oblasti administratoru i profesoru.

DeleteSubjectField (OblastModel om) - omogućava brisanje nove oblasti administratoru i profesoru.

**Panel za diskusiju o oblastima:**.Net kontroler zadužen za prikaz oblasti određenog predmeta i sve diskusije koje su u njima obavljane.

Funkcije:  
FieldDiscussion (int id) – vraća HTML stranicu oblasti i sve diskusije obavljene u njoj.

OpenDiscussion (int id) – poziva kontroler zadužen za diskusiju o konkretnom problemu.

AddDiscussion (DiskusijaModel dsm) – omogućava dodavanje nove diskusije u datoj oblasti.

RemoveDiscussion (DiskusijaModel dsm) – omogućava administratoru uklanjanje određene diskusije.

SearchDiscussion (string tag) – omogućava pretragu diskusija po tagovima.

SuggestTag (string tag) – omogućava predlaganje novog taga.

AddLiterature (LiteraturaModel prm) – dodaje raspoloživu literaturu za predmet.

RemoveLiterature (LiteraturaModel prm) – briše raspoloživu literaturu za predmet.

**Odabir Predmeta**.NET kontroler za prikaz smerova i predmeta po smerovima i selekciju predmeta.

Funkcije:  
ChooseSubject () – vraća HTML stranicu za prikaz svih smerova i odabir određenog predmeta u skladu sa smerom.

SelectSubject (int id) – poziva kontroler koji vraća stranicu za prikaz diskusije o predmetu.

**Korisnički panel korisnika**

.NET kontroler za prikaz osnovnih informacija o korisniku nakon logovanja i omogućavanje navigacije kroz sajt na odabrani način.

Funkcije:  
ControlPanel (int id) – vraća HTML stranicu sa podacima o korisniku koji se prijavio.

ChooseSubject () – poziva kontroler koji prikazuje stranicu za odabiranje predmeta.

SearchDiscussion (string tag) – pretražuje sve disksusije sa portala po tagovima.

SuggestTag (string tag) – omogućava predlaganje novog taga.

UpdateUserInfo (KorisnikModel km) – omogućava promenu podataka o ulogovanom korisniku.

PredgledPitanja() – omogućava pregled pitanja koje je postavio korisnik.

PregledProfila() – omogućava pregled profila drugog korisnika kome se pristupa preko linka u postavljenom pitanju, odgovoru ili komentaru

**Panel za registraciju**

**.**NET kontroler koji služi za prikaz forme za registrovanje novih korisnika.

Funkcije:  
Registration() – kontroler koji prikazuje formu za registraciju.

SubmitRegistration (RegistracijaModel regm) – prihvata podatke o novo-registrovanom korisniku i šalje ih administratoru na odobrenje.

**Panel za administratora**.NET kontroler koji služi za administraciju portala.

Funkcije:  
Administration () – prikaz HTML stranice administratora.

AddSubject (PredmetModel prm) – služi za dodavanje novog predmeta.

RemoveSubject (PredmetModel prm) – služi za brisanje postojećeg predmeta.

AddFieldOfStudy (SmerModel sm ) – služi za dodavanje novog smera.

RemoveFieldOfStudy (SmerModel sm) – služi za brisanje određenog smera.

EditFieldOfStudy (SmerModel sm) – služi za izmenu određenog smera.

ChangeUserStatus (KorisnikModel km) – služi za menjanje statusa korisniku između aktivnog, deaktiviranog i blokiranog.

ChangeUserRole (KorisnikModel km) – služi za menjanje role korisniku između korisnika, profesora i administratora

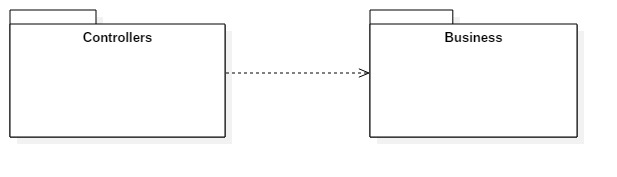
ApproveTag (TagModel tm) – omogućava odobravanje predloženog taga od strane korisnika.

DeleteTag (TagModel tm) – omogućava brisanje postojećeg taga iz bazena tagova.

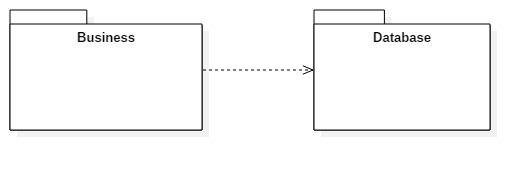
AddTag (TagModel tm) – omogućava dodavanje novog taga.

### Komponente za pristup podacima

Pristup bazi podataka realizovan je preko FluentNHIbernate ORM-a. Za generisanje klasa i atributa potrebnih za pristup bazi korišćen je LINQtoSQL. Funkcije koje služe za pristup bazi se nalaze u okviru klasa koje su sadržane u business sloju aplikacije. Ove funkcije se pozivaju od strane funkcija iz kontrolera radi upisivanja, ažuriranja, brisanja i čitanja podataka iz baze.



Database sadrži sve neophodne parametre za pristup bazi u klasi DataLayer. Takođe ovaj sloj sadrži sve klase koje su dobijene kao rezultat prevođenja baze u objektni model. Te klase se nalaze u okviru paketa kasa Entiteti. Ove klase služe za pristup podacima. Ovaj sloj sadrži klase koje opisuju način mapiranja relacione baze u objekni model. Te klase se nalaze u okviru paketa klasa Mapiranja.

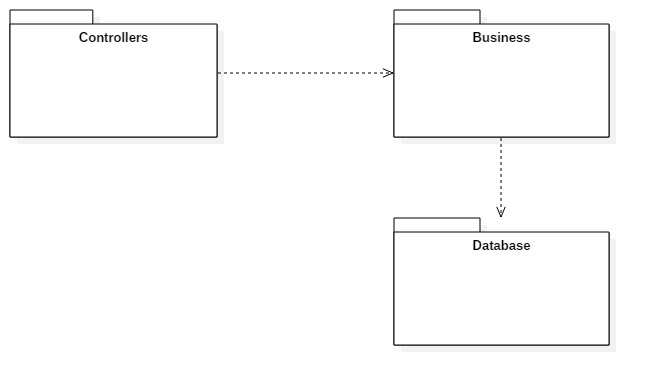


**Database**

Database je sloj aplikacije koji služi za povezivanje i razmenu podataka sa bazom koja se na lazi na Oracle serveru. Za povezivanje na bazu podataka koristi se klasa DataLayer. Potrebno je instancirati objekat tipa Session da bi se omogućio pristup. Bitne funkcije koje su implementirane u okviru Session klase su Load, Save, Update, Delete, Flush i Close.

* Load je templejtska f-ja i služi za učitavanje jednog reda određene tabele ili svih redova iz određene tabele. Kao parametar za template se stavlja ime klase koja predstvlja konkretnu tabelu u bazi.
* Save je funkcija koja služi za upis podataka u bazu. Kao argument joj se zadaje objekat klase entiteta.
* Update je funkcija koja služi za ažuriranje podataka u bazi. Kao argument joj se zadaje objekat klase entiteta.
* Delete je funkcija koja služi za ažuriranje podataka u bazi. Kao argument joj se zadaje objekat klase entiteta.
* Flush je funkcija koja služi za iniciranje promena u bazi
* Close je funkcija koja služi za zatvaranje sesije

Sve funkcije koje služe za pristup bazi, koje se nalaze u business sloju, koriste sledeću naredbu za rad sa ISession s = DataLayer.GetSession(); Ova naredba služi za povezivanje za inicijalizaciju sesije koja služi za pristup bazi. Upiti nad bazom se pisu posebnom sintaksom LINQ C# jezika, a izvršavaju se kroz kod C# jezika. Za upis, brisanje, ažuriranje i dobijanje podataka koriste se gore opisane funkcije sesije.



U nastavku su date klase koje se nalaze u business sloju i spisak f-ja okoje one sadrže.

**Korisnici.cpp:**

* + Create(Korisnik user) - Pruža interfejs za kreiranje novog korisnika i ažurira bazu podataka
  + Read(int id) – Čita podatke korinika sa navedenim id-jem
  + Update(Korisnik user) - Pruža interfejs za promenu podataka novedenog korisnika i ažurira bazu podataka
  + Delete(int id) – Briše podatke o korisniku sa navedenim id-jem
  + FindUser(string username) – Traži korisnika po navedenom imenu u bazi podataka
  + CheckPass(int id, string password) – Proverava ispravnost sifre kod navedenog korisnika
  + FindUser(int brojIndeksa) - Traži korisnika po navedenom broju indeksa u bazi podataka
  + ChangeStatus(Status s) – Menja status korisnika u bazi podataka
  + ChangeRole(Role r) – Menja ulogu korisnika u bazi podataka

**Predmeti.cpp:**

* + Create(Predmet s) - Pruža interfejs za kreiranje novog predmeta i ažurira bazu podataka
  + Read(int id) - Čita podatke premeta sa navedenim id-jem
  + Update(Predmet s) - Pruža interfejs za promenu podataka navedenog predmeta i ažurira bazu podataka
  + Delete(int id) - Briše podatke o predmetu sa navedenim id-jem
  + ReturnAllPredmet() – Pribavlja sve predmete iz baze podataka
  + AddProfesor(int id, Professor p) – Dodaje profesora navedenom predmetu
  + RemoveProfesor(int id, Professor p) – Briše profesora sa navedenog predmeta
  + AddResponsiblePerson(int id) – Dodaje profesora koji je nadležan za uredjivanje materiala na kursu
  + RemoveResponsiblePerson(int id) - Briše profesora koji je nadležan za uredjivanje materiala na kursu
  + ReturnAllSmer() - Pribavlja sve smerove iz baze podataka
  + AddSmer(Smer s) – Pruža interfejs za kreiranje novog smera i ažurira bazu podataka
  + RemoveSmer(Smer s) – Briše prosleđeni smer iz baze podataka
  + AddPredmetToSmer(int pid, int sid) - Pridruzuje predmet prosledjenom smeru
  + RemovePredmetFromSmer(int pid, int sid) - Predmet se ukida sa datog smera

**Oblasti.cpp**

* Create(Oblast s) - Pruža interfejs za kreiranje nove oblasti i ažurira bazu podataka
* Read(int id) – Čita iz baze oblast sa navedenim id-jem
* Update(Oblast s) - Pruža interfejs za izmenu navedene oblasti i ažurira bazu podataka
* Delete(int id) - Briše oblast sa navedenim id-jem
  + ReturnAllOblast() - Pribavlja sve oblasti iz baze podataka
* AddLiterature(int id, Literatura l) – Dodaje prosleđenu literaturu oblasti sa navedenim id-jem
* RemoveLiterature(int id, Literatura l) - Briše prosleđenu literaturu iz oblasti sa navedenim id-jem

**Pitanja.cpp:**

* Create(Pitanje q) - Pruža interfejs za kreiranje novog pitanja i ažurira bazu podataka
* Read(int id) - Čita iz baze oblast sa navedenim id-jem
* Update(Pitanje q) - Pruža interfejs za izmenu navedenog pitanja i ažurira bazu podataka
* Delete(int id) - Briše pitanje sa navedenim id-jem
* ReturnAll(int oblastId) – Vraća sva pitanja iz navedene oblasti
* ReturnAll(int korisnikId) – Vraća sva pitanja jednog korisnika
* ReturnAll(int tagId) – Vraća sva pitanja sa navedenim tagom

**Odgovori.cpp:**

* Create(Odgovor a) - Pruža interfejs za dodavanje novog odgovora i ažurira bazu podataka
* Read(int id) - Čita iz baze odgovor sa navedenim id-jem
* Update(Odgovor a) - Pruža interfejs za izmenu navedenog odgovora i ažurira bazu podataka
* Delete(int id) - Briše odgovor sa navedenim id-jem
* ReturnAll(int pitanjeID) – vraća sve odgovore prosleđenog komentara
* AddPlus(int id) – Dodaje plus odgovoru sa navedenim id-jem
* AddMinus(int id) - Dodaje minus odgovoru sa navedenim id-jem
* Aprove(int id) – Menja status odgovora u odobren

**Komentari.cpp:**

* Create(Komentar c) - Pruža interfejs za dodavanje novog komentara i ažurira bazu podataka
* Read(int id) - Čita iz baze komentar sa navedenim id-jem
* Update(Komentar c) - Pruža interfejs za izmenu navedenog komentara i ažurira bazu podataka
* Delete(int id) - Briše komentar sa navedenim id-jem
* ReturnAll(int odgovorId) – Vraća sve komentare prosleđenog odgovora

**Tagovi.cs:**

* Create(Tag t) - Pruža interfejs za dodavanje novog taga i ažurira bazu podataka
* Read(int id) - Čita iz baze tag sa navedenim id-jem
* Update(Tag t) - Pruža interfejs za izmenu navedenog taga i ažurira bazu podataka
* Delete(int id) - Briše tag sa navedenim id-jem
* Find(string tagName) – Pronalazi tag sa ditim imenom
* ReturnAll() - Pribavlja sve tagove iz baze podataka
* ReturnAll(string ime) - Pribavlja sve tagove iz baze podataka koji pocinju kao string ime
* AproveTag(int predlozenitagId) – Odobrava predloženi tag sa navedenim id-jem

# 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 1400 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. Adiutor portal će biti dostupan 24 časa dnevno, 7 dana u nedelji. Vreme kada portal nije dostupan ne sme da pređe 2%.
2. Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 500 sati.