e-Čitaonica

Web aplikacija za studentsku saradnju, informisanje i pripremu ispita

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 22.04.2022. | 1.0 | Inicijalna verzija | Andrija Tošić, Milica Rangelov, Stefan Stojadinović, Božidar Tošić |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Sadržaj

1. Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture E-index web aplikacije.

2. Opseg dokumenta

Dokument se odnosi na E-index web aplikaciju koja će biti razvijena od strane GPT. E-index predstavlja skraćenicu elektronskog indeksa koji ce biti namenjen profesorima i studentima za koriscenje. Namena aplikacije jeste sto efikasniji i pregledniji uvid u , kreiranje i održavanje sadržaja vezanih za rad jedne naučno-istraživačke laboratorije.

3. Reference

Spisak korišćene literature:

1. D01\_Predlog\_Projekta .
2. D02\_Vizija\_sistema
3. D03\_Plan\_realizacije
4. D03\_Raspored\_Aktivnosti.mpp
5. D04\_Spec\_Zahteva

4. 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.

5. Ciljevi i ograničenja arhitekture

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

1. Web aplikacija E-index će biti implementirana kao Web aplikacija zasnovana na korišćenju React fronted framework-a , nest.js i pgadmin4 baze.
2. Korisnički interfejs Web aplikacije mora da bude optimizovan za sledeće verzija Web pregledača i operativnih sistema: Chrome – 63 +, Firefox – 58 +, Opera – 50 +, Edge – 13 +, Internet Explorer – 11 +, Safari – 9 +, iOS – 9 +, Android- 4.4 +.
3. Svi zahtevi u pogledu performansi dati u [5] moraju biti uzeti u obzir pri izboru arhitekture i razvoju sistema.

6. Pogled na slučajeve korišćenja 5

6.1 Dijagrami slučajeva korišćenja 6

6.2 Kratak opis slučajeva korišćenja 8

6.2.1 Registracija e-mailom i lozinkom 8

6.2.2 Registracija putem single sign-on-a 8

6.2.3 Prijavljivanje e-mailom i lozinkom 9

6.2.4 Prijavljivanje putem single sign-on-a 9

6.2.5 Pregled početne stranice aplikacije 9

6.2.6 Pregled stranica korisnika 9

6.2.7 Pregled stranica predmeta 9

6.2.8 Pregled blanketa sa stranice predmeta 9

6.2.9 Pregled blanketa sa stranice predmeta po ispitnom roku 9

6.2.10 Pregled blanketa sa stranice predmeta po godini 9

6.2.11 Pregled blanketa sa stranice predmeta po tipu blanketa 9

6.2.12 Pregled zadataka i pitanja iz oblasti 9

6.2.13 Pregled diskusija iz oblasti 9

6.2.14 Pretraga predmeta 9

6.2.15 Pretraga studenata i profesora 9

6.2.16 Prijava neprikladnog sadržaja 10

6.2.17 Predlaganje rešenja zadatka ili odgovora na pitanje 10

6.2.18 Izmena predloga rešenja 10

6.2.19 Brisanje predloga rešenja 10

6.2.20 Zahvaljivanje na odgovoru 10

6.2.21 Sastavljanje blanketa za vežbu 10

6.2.22 Postavljanje pitanja 10

6.2.23 Izmena pitanja 10

6.2.24 Brisanje pitanja 10

6.2.25 Prihvatanje odgovora na pitanje 10

6.2.26 Arhiviranje odgovora 10

6.2.27 Brisanje odgovora 10

6.2.28 Arhiviranje pitanja 11

6.2.29 Postavljanje blanketa 11

6.2.30 Ažuriranje informacija o predmetima 11

6.2.31 Podela predmeta po oblastima 11

6.2.32 Potvrđivanje tačnosti odgovora 11

6.2.33 Provera znanja izradom blanketa 11

6.2.34 Dodavanje predmeta 11

6.2.35 Brisanje predmeta 11

6.2.36 Raspoređivanje profesora po predmetima 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 Angular 13

7.1.5 ASP.NET Web API 13

7.1.6 SQL Server 13

8. Pogled na procese 13

8.1 Procesi 13

8.1.1 Web čitač 13

8.1.2 Web server 14

8.1.3 WEB API 14

8.1.4 SQL Server 14

9. Pogled na raspoređivanje sistema 14

9.1 Klijent 14

9.2 Web server 14

9.3 Web API server 14

9.4 DBMS server 14

10. Pogled na implementaciju sistema 14

10.1 Model domena 15

10.2 Šema baze podataka 15

10.3 Komponente sistema 16

10.3.1 Komponente korisničkog interfejsa 16

10.3.2 Komponente aplikacione logike 17

10.3.3 Komponente za pristup podacima 18

11. 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 500 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].

12. Kvalitet

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

1. E-index web aplikacija će biti dostupna 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.

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture aplikacije e-Čitaonica.

# Opseg dokumenta

Dokument se odnosi na e-Čitaonica aplikaciju koja će biti razvijena od strane RST-a. E-Čitaonica predstavlja skraćenicu za web aplikaciju za studentsku saradnju, informisanje i pripremu ispita. Namena sistema je efikasno podsticanje diskusije između studenata, nadgledanje njihovog napretka i unapređivanje načina studiranja, učenja i testiranja znanja.

# Reference

Spisak korišćene literature:

1. E-Čitaonica– Predlog projekta, RST-e-Čitaonica, V1.1, 2022, RST.
2. E-Čitaonica – Planirani raspored aktivnosti na projektu, V1.0, 2022, RST.
3. E-Čitaonica – Plan realizacije projekta, V1.0, 2022, RST.
4. E-Čitaonica – Vizija sistema, V1.0, 2022, RST.
5. E-Čitaonica – Specifikacija zahteva,V1.0, 2022,RST

# 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 e-Čitaonica će biti implementirana kao Web aplikacija, zasnovana na korišćenju Angular frontend framework-a, ASP.NET web framework-a i SQL Server DBMS-a [4].
2. Korisnički interfejs Web aplikacije mora da bude optimizovan za sledeće verzija Web pregledača i operativnih sistema: Chrome – 63 +, Firefox – 58 +, Opera – 50 +, Edge – 13 +, Internet Explorer – 11 +, Safari – 9 +, iOS – 9 +, Android- 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].

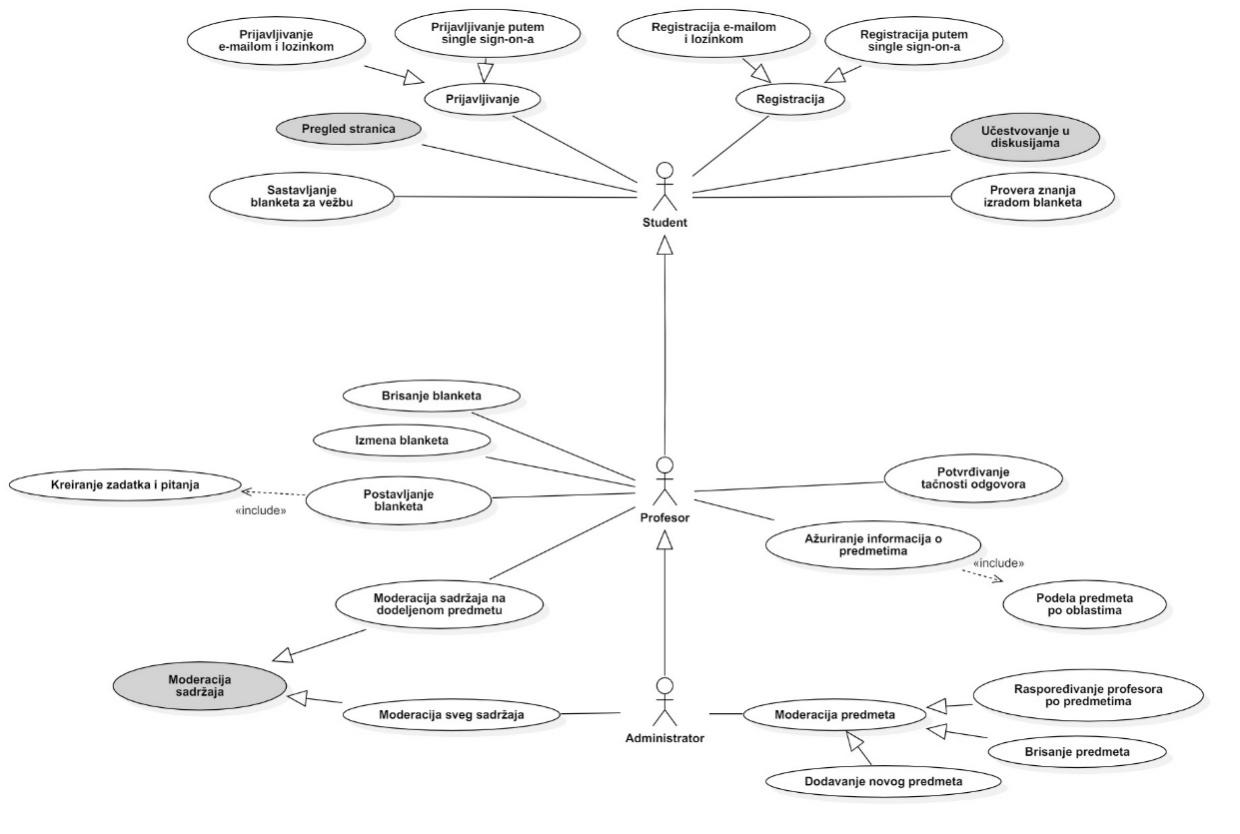
Slučajevi korišćenja aplikacije e-Index su:

* Prijava emailom i lozinkom
* Pregled početne stranice aplikacije
* Pregled stranica korisnika
* Uvid u predmete studenta
* Pregled osnovnih informacija o studentu
* Student prijavljuje ispit
* Podešavanje profila
* Odabir izbornih predmeta studenta
* Anketiranje profesora
* Uvid studenta u dosadašnje postignute rezultate
* Uvid u predmete profesora
* Pregled osnovnih informacija o profesoru
* Pregled ocena i komentara profesora
* Dodavanje studenta
* Brisanje studenta
* Ažuriranje studenta
* Dodavanje profesora
* Brisanje profesora
* Ažuriranje profesora

Ove slučajeve korišćenja mogu da iniciraju student, profesor ili administrator.

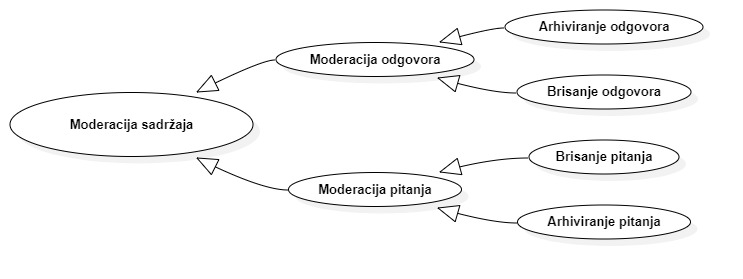
## Dijagrami slučajeva korišćenja

Osnovni UML dijagram koji prikazuje korisnike i slučajeve korišćenja aplikacije e-Čitaonica prikazan je na sledećoj slici:

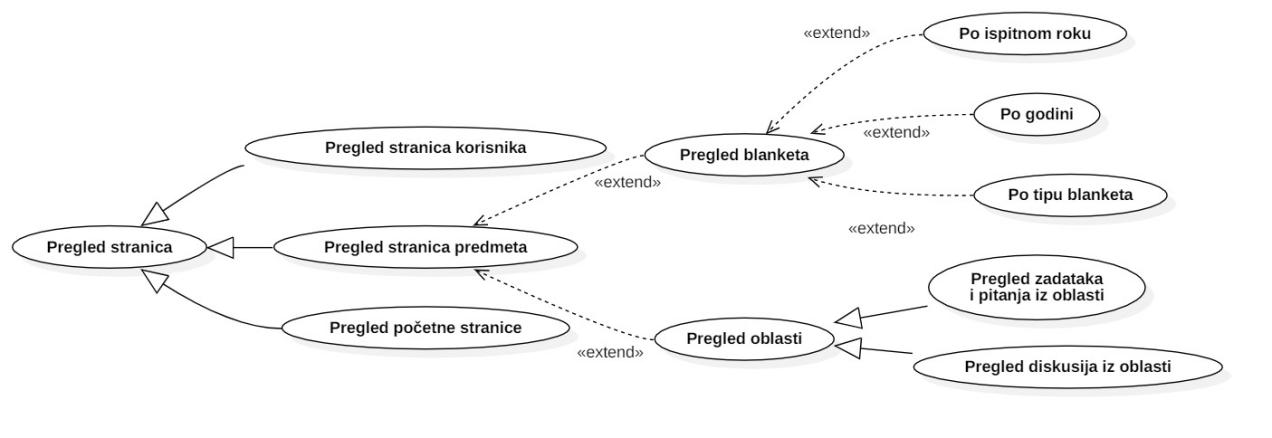


Slučajevi korišćenja *moderacija sadržaja*, *pregled stranica, učestvovanje u diskusijama* i *pretraga* obuhvataju složenije radnje koje se mogu razložiti dalje na pojedinačne slučajeve korišćenja.

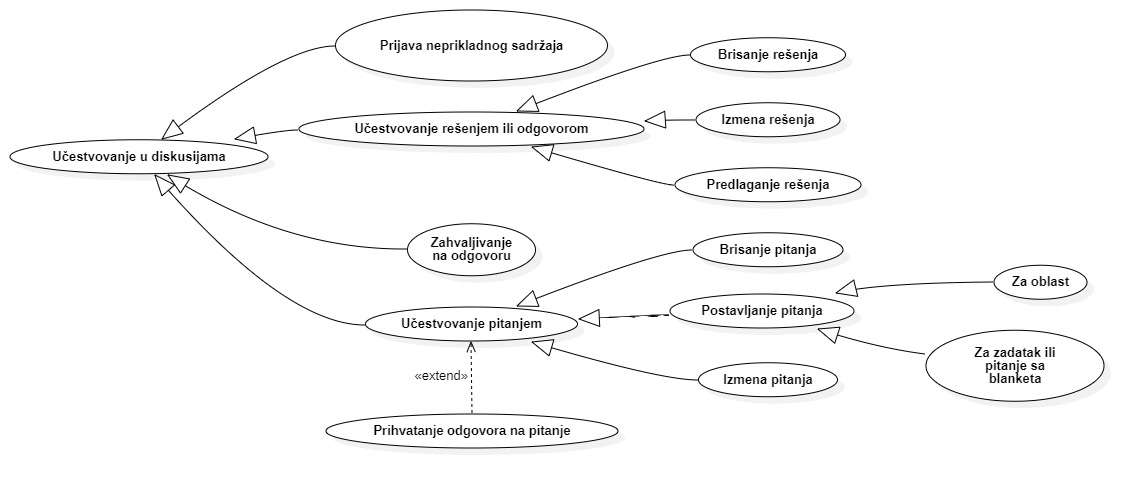
Detaljni UML dijagram za slučaj korišćenja *moderacija sadržaja* je prikazan na sledećoj slici:



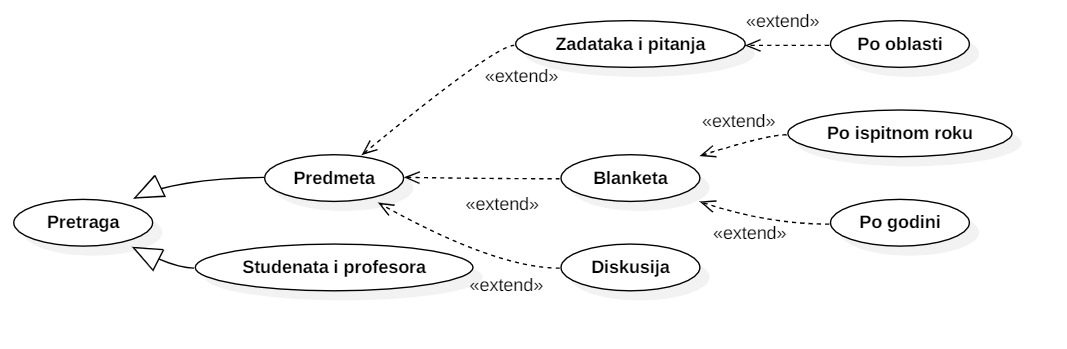
Detaljni UML dijagram za slučaj korišćenja *pregled stranica* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *učestvovanje u diskusijama* je prikazan na sledećoj slici:



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



## Kratak opis slučajeva korišćenja

### Prijava e-mailom i lozinkom

Kratak opis: Prijava na aplikaciju.

Akteri koji iniciraju slučaj korišćenja: Student, Profesor, Administrator.

### Pregled početne stranice aplikacije

Kratak opis: Prikaz naslovne stranice aplikacije sa bazičnim informacijama koje su od interesa korisniku.

Akteri koji iniciraju slučaj korišćenja: Student, Profesor, Administrator.

### Pregled stranica korisnika

Kratak opis: Prikaz stranice sa informacija o određenom korisniku.

Akteri koji iniciraju slučaj korišćenja: Student, Profesor,Administartor.

### Uvid u predmete studenta

Kratak opis: Prikaz stranice sa spiskom svih predmeta koje student pohađa.

Akteri koji iniciraju slučaj korišćenja: Student.

### Pregled osnovnih informacija o studentu

Kratak opis: Prikaz stranice sa osnovnim informacijama o studentu.

Akteri koji iniciraju slučaj korišćenja: Student.

### Student prijavljuje ispit

Kratak opis: Prikaz stranice sa podacima o trenutnom roku i lista svih ispita koje student može da prijavi za predstojeći rok.

Akteri koji iniciraju slučaj korišćenja: Student.

### Podešavanje profila

Kratak opis: Prikaz stranice sa podešavanjima osnovnih podataka/informacije.

Akteri koji iniciraju slučaj korišćenja: Student, Profesor.

### Odabir izbornih predmeta studenta

Kratak opis: Prikaz stranice za odabir izbornih predmeta studenta.

Akteri koji iniciraju slučaj korišćenja: Student.

### Anketiranje profesora

Kratak opis: Student može da na osnovu ličnog iskustva da ocenu profesora za predmet koji je položio i da takođe ostavi neko lično mišljenje, odnosno komentar, na celokupni rad profesora.

Akteri koji iniciraju slučaj korišćenja: Student.

### Uvid studenta u dosadašnje postignute rezultate

Kratak opis: Student može da dobije listu svih predmeta koje je položio i to sa ocenom sa kojom je položio. Takođe može da filtrira te podatke na osnovu godine studija, semestra ili ispitnog roka. Može da dobije podatke o osvojenim ESP bodovima po istim kriterijumima filtracije.

Akteri koji iniciraju slučaj korišćenja: Student.

### Uvid u predmete profesora

Kratak opis: Prikaz stranice sa spiskom svih predmeta na kojima profesor predaje.

Akteri koji iniciraju slučaj korišćenja: Profesor.

### Pregled osnovih informacija o profesoru

Kratak opis: Prikaz stranice sa osnovim informacijama o profesoru.

Akteri koji iniciraju slučaj korišćenja: Profesor.

### Pregled ocena i komentara profesora

Kratak opis: Prikaz stranice sa ocenama i komentarima o profesoru.

Akteri koji iniciraju slučaj korišćenja: Student, Profesor.

### Dodavanje studenta

Kratak opis: Adminstrator u bazu podataka dodaje novog studenta.

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Brisanje studenta

Kratak opis: Adminstrator iz baze podataka briše studenta.

Akteri koji iniciraju slučaj korišćenja: Administartor.

### Ažuriranje studenta

Kratak opis: Adminstrator u bazi podataka ažurira studenta.

Akteri koji iniciraju slučaj korišćenja: Administartor.

### Dodavanje profesora

Kratak opis:Adminstrator u bazu podataka dodaje novog studenta.

Akteri koji iniciraju slučaj korišćenja: Administartor.

### Brisanje profesora

Kratak opis: Adminstrator iz baze podataka briše profesora.

Akteri koji iniciraju slučaj korišćenja: Administartor.

### Ažuriranje profesora

Kratak opis: Adminstrator u bazi podataka ažurira profesora.

Akteri koji iniciraju slučaj korišćenja: Administartor.

# 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.

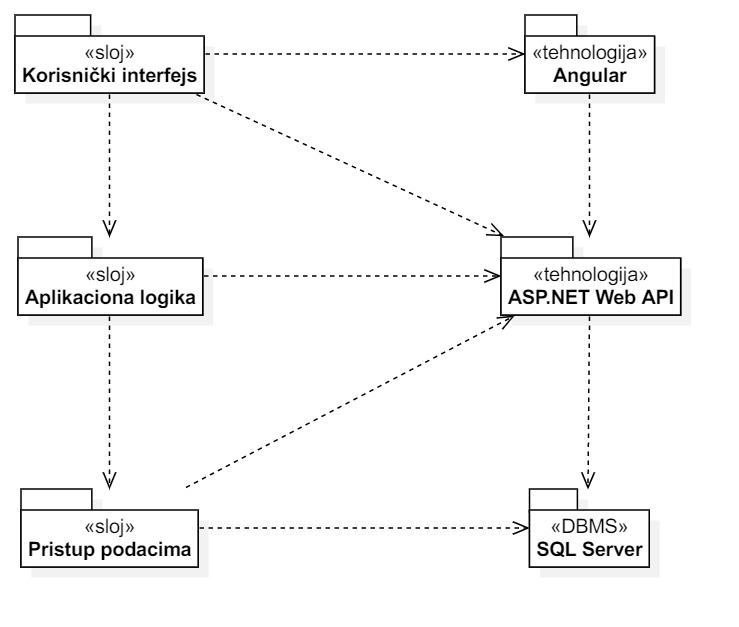
Logički pogled na aplikaciju e-Čitaonica obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket *Korisnički interfejs* sadrži Web stranice, Angular skripte i multimedijalni sadržaj koji realizuju grafički dizajn i forme preko kojih korisnici sistema komuniciraju sa sistemom.

Paket *Aplikaciona logika* predstavlja srednji sloj sistema koji sadrži skripte zadužene za realizaciju funkcionalnosti specifičnih za domen sistema koji se razvija.

Paket *Pristup podacima* sadrži Angular skripte koje 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 za Web Aplikaciju e-Čitaonica. Sadrži sve HTML,CSS stilove i Angular skripte koje se koriste radi generisanja adekvatnih HTML stranica putem kojih korisnik komunicira sa sistemom.

Ovaj sloj zavisi od sloja aplikacione logike i Angular-a.

### Aplikaciona logika

sloj

Ovaj sloj sadrži Angular skripte putem kojih se uspostavlja veza između korisničkog interfejsa i sloja za pristup podacima.

Sloj aplikacione logike zavisi od sloja Angular i sloja za pristup podacima.

### Pristup podacima

sloj

Sloj za pristup podacima se nalazi na dnu arhitekture i sadrži adekvatne skripte za pribavljanje, dodavanje, brisanje i izmenu podataka koji se nalaze u bazi.

Zadati sloj zavisi od Angular-a i SQL Servera.

### Angular

tehnologija

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

### ASP.NET Web API

tehnologija

Tehnologija ASP.NET Web API obezbeđuje mehanizam za pisanje i izvršavanje operacija na strani servera. Ove operacije mogu da generišu rezultate koji se prikazuju na strani korisnika i pristupaju bazi podataka u cilju pribavljanja, unosa i ažuriranja podataka.

### SQL Server

DBMS

SQL Server predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju aplikacije e-Čitaonica.

# Pogled na procese

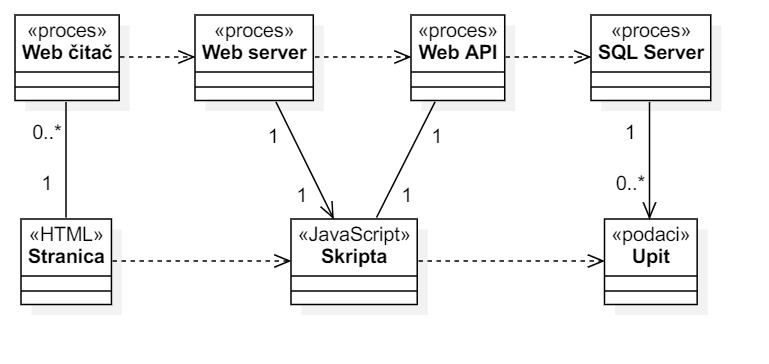
U ovom odeljku je sadržan pogled na procesnu arhitekturu sistema. Ovaj opis treba da sadrži specifikaciju različitih zadataka (procesa i niti) uključenih u rad sistema. Takođe je potrebno dati dijagrame koji pokazuju njihovu interakciju i konfiguraciju. Dodela objekata i klasa na određene zadatke takođe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

Web aplikacije zasnovane na SQLServeru, ASP .NET-u i Angular-u imaju relativno jednostavan procesni model koji je u potpunosti pod kontrolom Web servera. Sa stanovišta projektanta Web aplikacije nije potrebno voditi računa o načinu rada Web servera i načinu izvršavanja skripti.

Ilustracije radi u nastavku je dat opis procesa uključenih u izvršenje e-Čitaonice kao Web aplikacije.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju aplikacije e-Čitaonica. Dijagram je opšteg tipa i može se primeniti na bilo koju Web aplikaciju zasnovanu na Angular-u, ASP.NET Web API-ju i SQL Server bazi podataka.



### Web čitač

Web čitač je proces koji izvršava funkcionalnost aplikacije za prikaz HTML stranica dobijenih od nekog Web servera. U najopštijem slučaju Web čitač u jednom trenutku može da prikazuje samo jednu HTML stranicu.

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

### Web server

Web server je proces koji izvršava funkcionalnost opsluživanja zahteva prispelih sa više Web čitača. Ukoliko je zahtevana stranica, Web server inicira izvršenje WEB API procesa koji obrađuje odgovarajući zahtev i generiše sadržaj koji se vraća čitaču. Web server može paralelno da inicira veći broj procesa.

### WEB API

WEB API proces obavlja posao obrade i generiše odgovarajući tekstualni sadržaj koji Web server šalje Web čitaču. Za izvršenje obrade ovaj proces može da zahteva usluge SQL servera-a. Komunikacija između WEB API procesa i SQL servera se obavlja preko prosleđivanja upita i vraćanja rezultat.

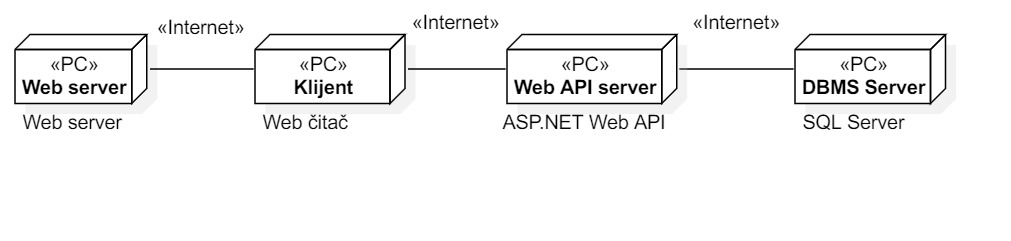
### SQL Server

SQL Server je proces koji izvršava funkcionalnost SQL 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 aplikacije e-Čitaonica.



## Klijent

Pristup aplikaciji e-Čitaonica se obavlja preko klijentskih računara na kojima se izvršava Web čitač. Za povezivanje između klijenta i Web API servera koristi se Internet infrastruktura tako da nema ograničenja u pogledu lokacije klijenta.

## Web server

Predstavlja računar koji hostuje i omogućava klijentima preuzimanje klijentskog dela aplikacije. To uključuje resurse kao što su .html, .css, .js datoteke, ali i slike i datoteke drugog tipa.

## Web API server

Računar na kome se ispunjavaju zahtevi klijenata i vrši se pristupanje bazi radi preuzimanja, izmene, dodavanja i brisanja podataka. Sa samim DBMS Serverom povezan je putem Interneta.

## DBMS server

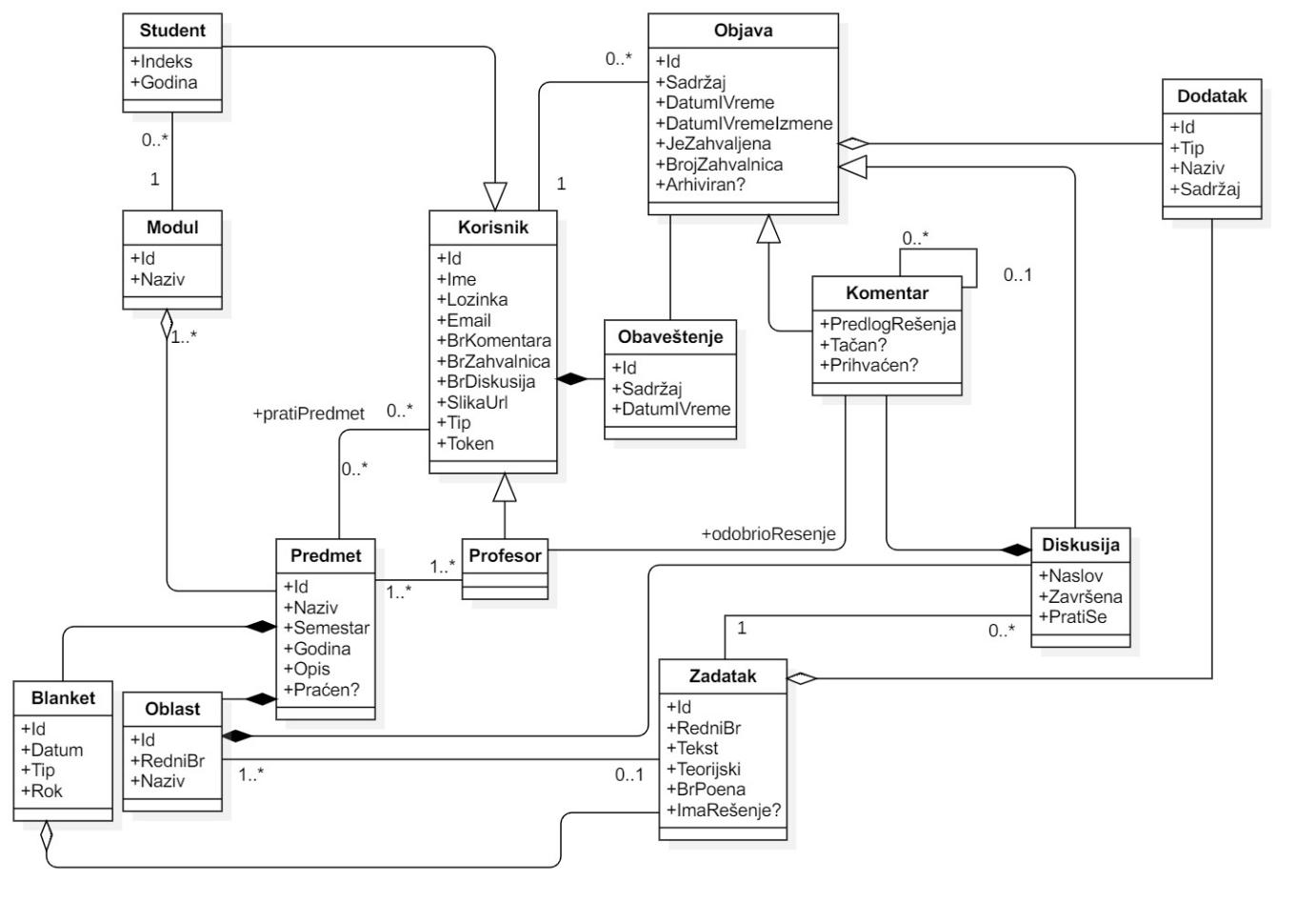
DBMS server je računar na kome se izvršava SQL Server proces koji realizuje funkcionalnost sistema za upravljanje bazama podataka. Uslugama DBMS-a se pristupa putem Internet mreže.

# Pogled na implementaciju sistema

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

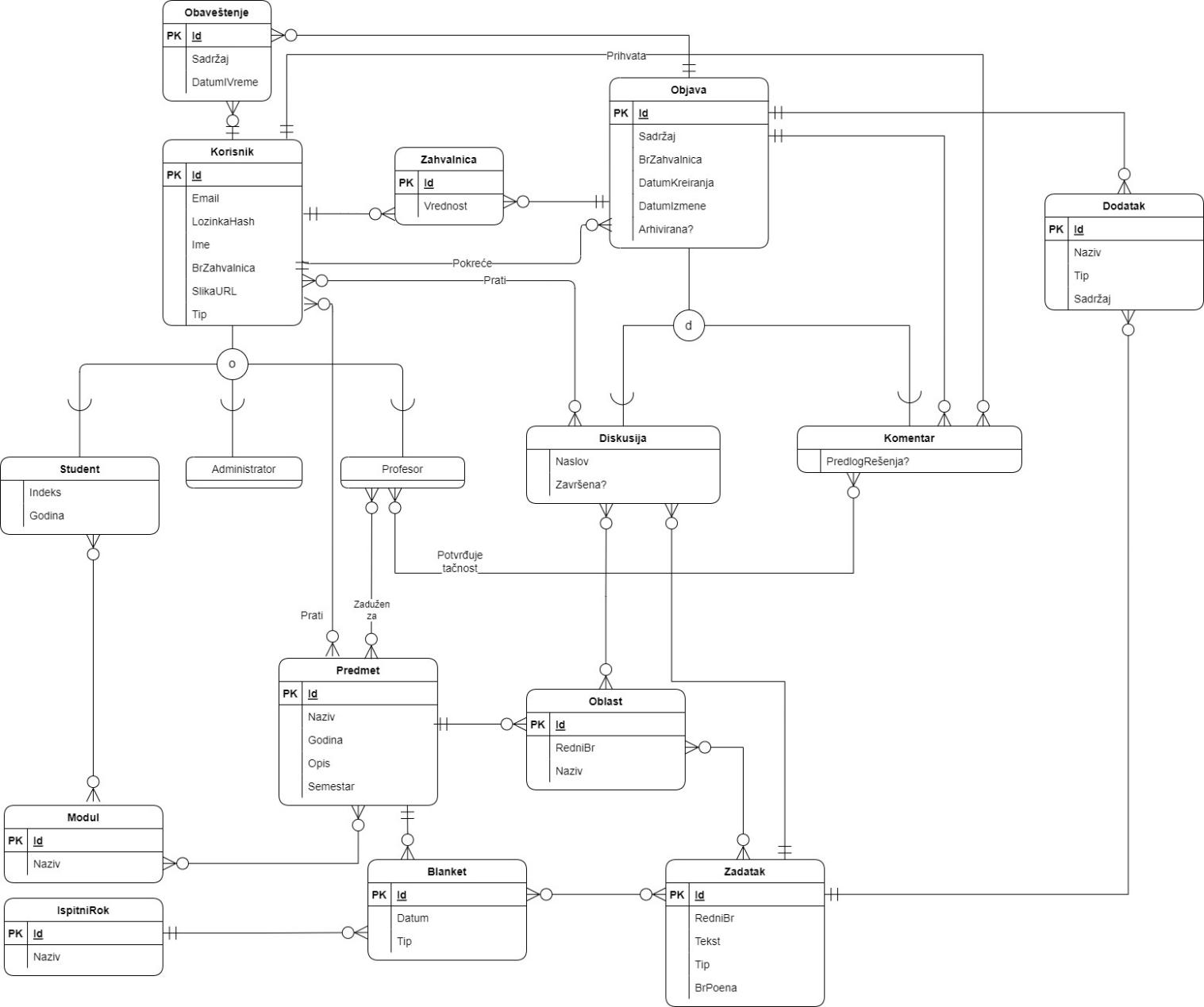
## Model domena

Model domena za koji se projektuje Web aplikacija e-Čitaonica 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 koje će biti implementirane.



## Šema baze podataka

Detaljna šema baze podataka prikazana je na sledećem EER dijagramu.

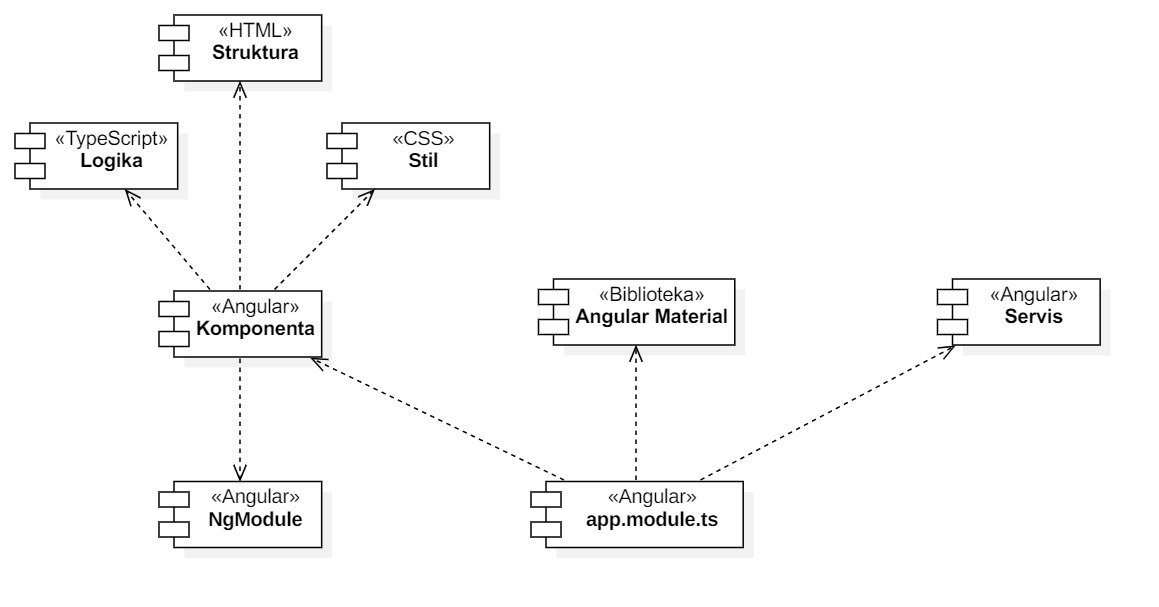


## Komponente sistema

Komponente sistema Web aplikacije e-Čitaonica su Angular komponente, CSS, HTML, API kontroleri, servisi čiji će pregled biti dat po arhitekturnim slojevima. Za ilustraciju će biti korišćeni UML dijagrami komponenti, ali i dijagrami klasa. U slučajevima gde je Angular 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 skripti.

### Komponente korisničkog interfejsa

Dizajn korisničkog interfejsa je obuhvaćen sledećim komponentama:



Komponenta **Struktura** se koristi za definisanje struktura stranica koje se prikazuju korisniku.

Komponenta **Stil** predstavlja opis stilova za pojedine HTML elemente i komponente koji se javljaju na različitim stranicama.

Komponenta **Logika** predstavlja deo koji poseduje logiku obrade podataka na klijentskoj strani, ulogu dinamičke modifikacije prikaza, asinhrono slanje http zahteva web API-ju.

**Komponenta** predstavlja osnovni element korisničkog interfejsa u Angular projektu. Sama komponenta predstavlja celinu koja enkapsulira odgovarajuće stilove, strukturu i logiku.

**Angular Material** je biblioteka komponenti korisničkog interfejsa koja omogućava brzo i lako uvođenje i korišćenje gotovih komponenti.

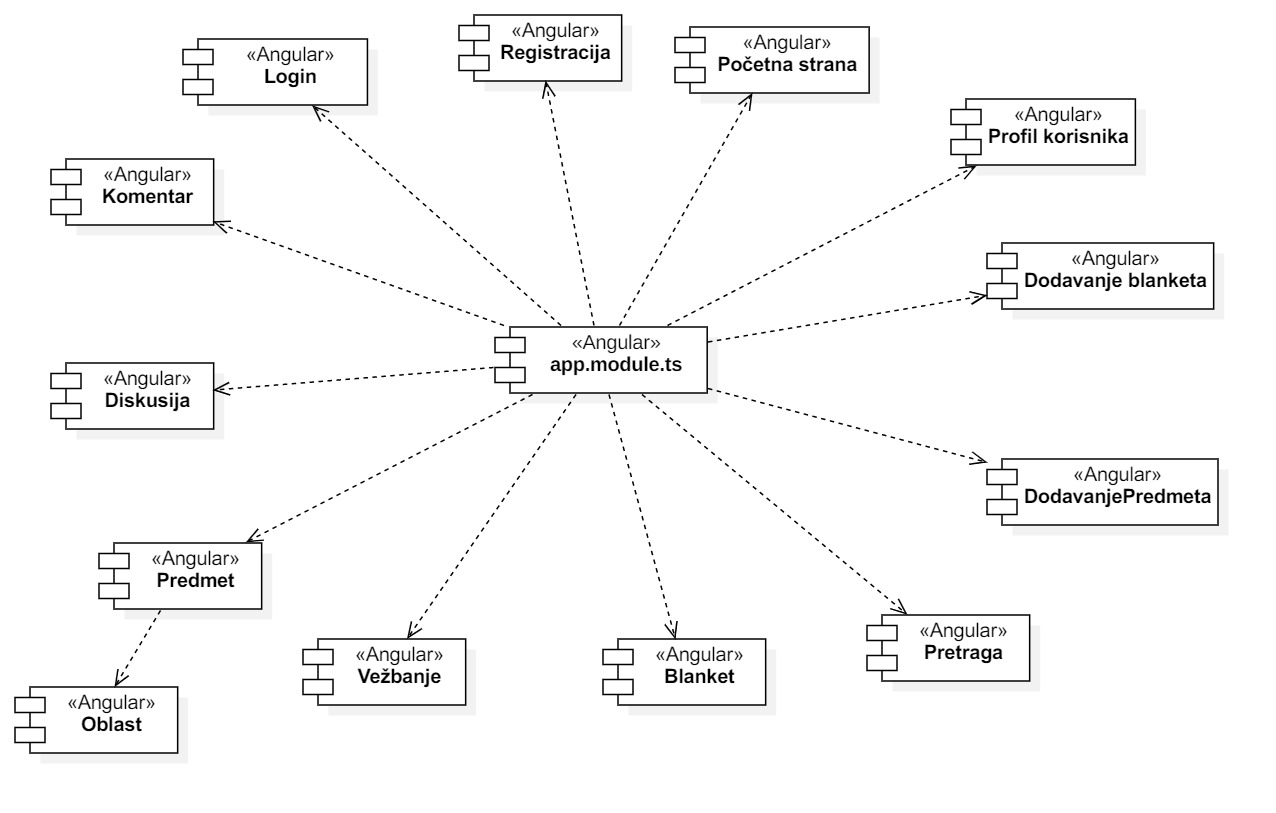
Komponenta **Servis** predstavlja deo aplikacije, odnosno klasu sa specifičnom svrhom, uglavnom sa ciljem komunikacije sa web API-jem.

Komponenta **app.module.ts** predstavlja skriptu koja organizuje zavisnosti angular aplikacije, uvođenje novih komponenata, eksternih modula, servisa.

Komponenta **NgModule** predstavlja komponentu koja omogućava organizovanje povezanih komponenata.

### Komponente aplikacione logike

Na sledećem dijagramu su prikazane komponente ovog sloja i njihove međusobne zavisnosti:

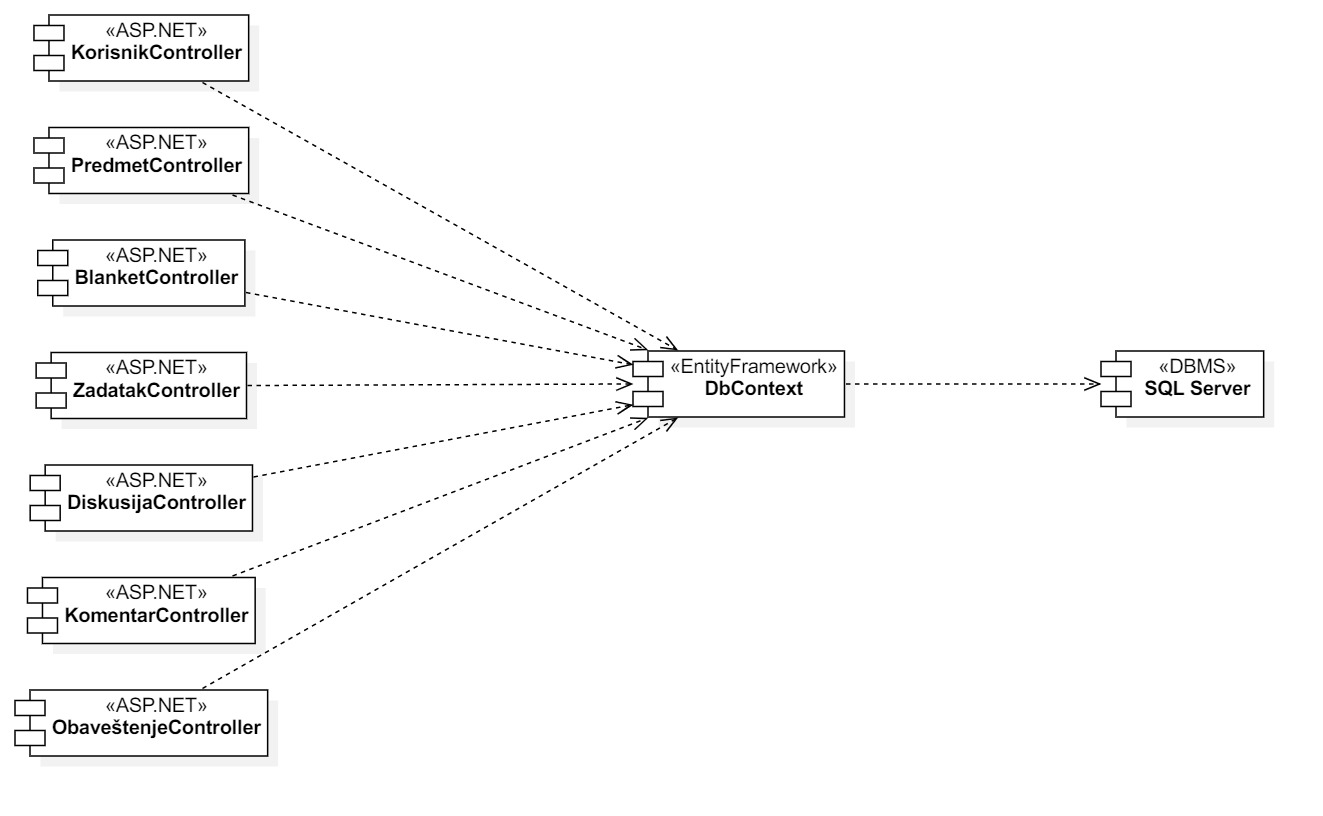


Pregled funkcija svake od ponuđenih komponenata:

* Početna strana: Prikaz početne stranice čiji sadržaj će zavisiti od tipa korisnika i od toga koji korisnik joj pristupa.
* Registracija: Stranica koja omogućava registrovanje novih korisnika.
* Login: Stranica koja omogućava korisniku prijavljivanje na nalog.
* Komentar: Stranica koja prikazuje odabrani komentar sa referencom na odgovarajuću diskusiju, roditeljski komentar.
* Diskusija: Stranica koja prikazuje pokrenutu diskusiju u okviru predmeta, zajedno sa relevantnim komentarima.
* Predmet: Stranica odabranog predmeta, predstavlja celinu osnovnih informacija o predmetu, pokrenutih diskusija, blanketa.
* Oblast: Stranica koja prikazuje sadržaj vazan za određenu oblast na odgovarajućem predmetu.
* Vežbanje: Stranica koja omogućava korisniku kreiranje sopstvenih blanketa radi vežbe.
* Blanket: Stranica koja prikazuje zadatke sa odabranog blanketa.
* Pretraga: Stranica koja pruža mogućnost pretraživanja sadržaja.
* DodavanjePredmeta: Stranica koja omogućava administratoru dodavanje novog predmeta.
* DodavanjeBlanketa: Stranica koja određenom tipu korisnika omogućava kreiranje novih blanketa.
* ProfilKorisnika: Stranica koja prikazuje osnovne podatke o korisniku i njegove pokrenute diskusije i komentare.

### Komponente za pristup podacima

Pristup bazi podataka omogućen je WebAplikaciji preko Aplikativnog sloja. WebAplikacija pristupa Kontrolerima i njihovim funkcijama.



Kontroleri kojima WebAplikacija pristupa radi pribavljanja podataka:

* KorisnikController: Sadrži sve potrebne funkcije za pristup podacima korisnika, kreiranje novih korisnika, njegovo uspešno prijavljivanje.
* PredmetController: Sadrži sve potrebne funkcije potrebne za pribavljanje podataka o određenom predmetu, dodavanje novog predmeta, brisanje postojećeg predmeta, dodavanje oblasti u okviru predmeta, ažuriranje podataka o predmetu, raspoređivanje profesora na predmetu.
* BlanketController: Sadrži sve potrebne funkcije za pribavljane blanketa, njihovu izmenu, brisanje i kreiranje novih .
* ZadatakController: Sadrži sve funkcije potrebne za pribavljanje podataka o zadatku, izmenu i brisanje postojećeg ili dodavanje novog, ili pak pridruživanje adekvatnog rešenja.
* DiskusijaControler: Sadrži sve potrebne funkcije za pribavljanje svih podataka povezanih sa diskusijom, kreiranje novih diskusija, brisanje postojećih, za dodavanje novog sadržaja.
* KomentarController: Sadrži sve potrebne funkcije za dodavanje komentara, brisanje ili izmenu, potvrđivanje tačnosti odgovaraućeg komentara,njihovo arhiviranje.
* ObaveštenjeController: Sadrži potrebne funkcije za dobijanje podataka određenog obaveštenja, kreiranje novih obaveštenja ili uklanjanju postojećih.

# 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. Web aplikacija e-Čitaonica će biti dostupana 24 časa dnevno, 7 dana u nedelji. Vreme kada Web aplikacija nije dostupana ne sme da pređe 10%.
2. Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 120 sati.