LMS

Library Management System

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 28.04.2022. | 1.0 | Inicijalna verzija | Todor, Vukadin, Lazar,  Nikola |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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 Prijavljivanje 8

6.2.2 Registracija 8

6.2.3 Pregled kataloga knjiga po književnoj vrsti 8

6.2.4 Pregled kataloga knjiga po književnom rodu 8

6.2.5 Pregled kataloga knjiga po žanru 8

6.2.6 Pregled kataloga knjiga po autoru 8

6.2.7 Pregled kataloga knjiga po jeziku 8

6.2.8 Upis u red čekanja za knjigu 8

6.2.9 Generisanje predloga za čitanje 8

6.2.10 Diskusija o knjigama 8

6.2.11 Dobijanje e-mail obaveštenja 8

6.2.12 Naplata članarine 8

6.2.13 Koordinacija radnika 8

6.2.14 Prikaz kompletnog izveštaja 9

6.2.15 Prikaz podataka za ogranak 9

6.2.16 Prikaz podataka o radniku 9

7. Pogled na logičku arhitekturu sistema 9

7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 9

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 ReactJS 10

7.1.6 .NET Core 10

7.1.7 Microsoft SQL Server 10

8. Pogled na procese 10

8.1 Procesi 10

8.1.1 Web čitač 11

8.1.2 Web server 11

8.1.3 .NET Core 11

8.1.4 Microsoft SQL Server 11

9. Pogled na raspoređivanje sistema 11

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 13

10.3.1 Komponente korisničkog interfejsa 13

10.3.2 Komponente aplikacione logike 14

10.3.3 Komponente za pristup podacima 14

11. Performanse 15

12. Kvalitet 15

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture LMS Web aplikacije.

# Opseg dokumenta

Dokument se odnosi na LMS Web aplikaciju koji će biti razvijen od strane LMSTeam-a. LMS predstavlja skraćenicu za Library Management System. Namena sistema je pružanje usluga biblioteke online i prikaz raznih analitičkih podataka rukovodstvenom delu.

# Reference

Spisak korišćene literature:

1. D01\_Predlog\_Projekta, V1.0, 2022, LMSTeam.
2. D02\_Vizija\_Sistema, V1.0, 2022, LMSTeam.
3. D03\_Plan\_Realizacije, V1.0, 2022, LMSTeam.
4. D03\_Raspored\_Aktivnosti, V1.0, 2022, LMSTeam.
5. D04\_Spec\_Zahteva, V1.0, 2022, LMSTeam.

# 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. LMS će biti implementiran kao Web aplikacija zasnovana na ReactJS skripting jeziku, .NET Core i Microsoft SQL bazi podataka[2].
2. Klijentski deo LMS Web aplikacije će biti optimizovan za sledeće Web čitače: Google Chrome, Opera Firefox (Mozilla)[2].
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].

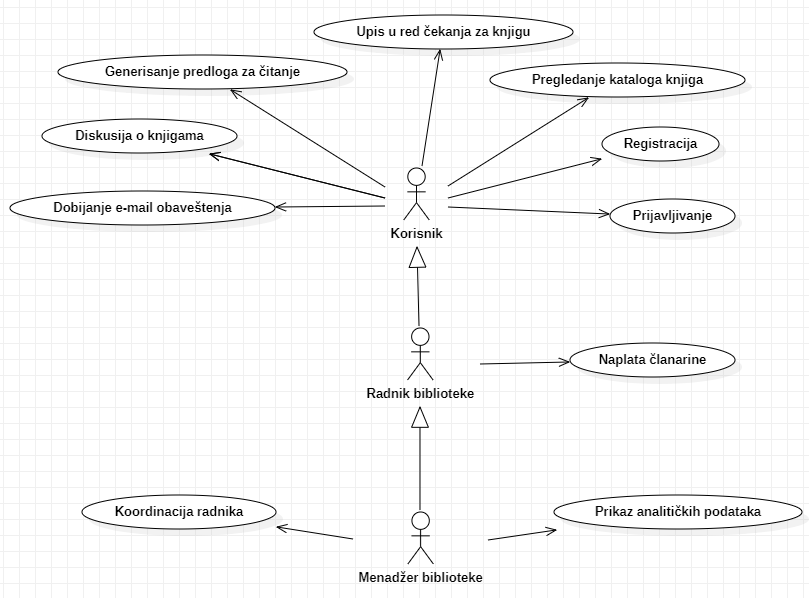
Slučajevi korišćenja LMS Web aplikacije su:

* Prijavljivanje
* Registracija
* *Pregled kataloga knjiga*
  + Pregled kataloga knjiga po književnoj vrsti
  + Pregled kataloga knjiga po književnom rodu
  + Pregled kataloga knjiga po žanru
  + Pregled kataloga knjiga po autoru
  + Pregled kataloga knjiga po jeziku
* Upis u red čekanja za knjigu
* Generisanje predloga za čitanje
* Diskusija o knjigama
* Dobijanje e-mail obaveštenja
* Naplata članarine
* Koordinacija radnika
* *Prikaz analitičkih podataka*
  + Prikaz kompletnog izveštaja
  + Prikaz podataka za ogranak
  + Prikaz podataka o radniku

Ove slučajevi korišćenja mogu da iniciraju korisnik usluga biblioteke, radnik biblioteke i menadžer biblioteke.

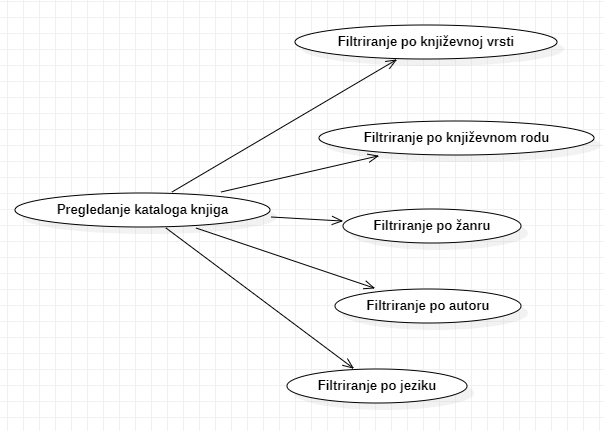
## Dijagrami slučajeva korišćenja

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

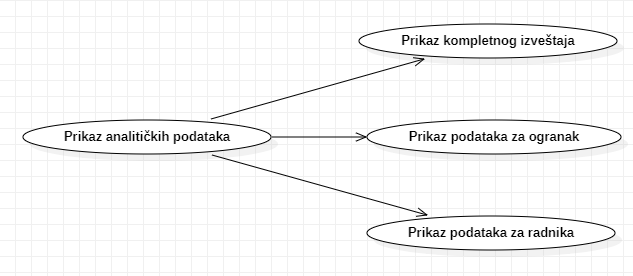


Slučajevi korišćenja *pregledanje kataloga knjiga* i *prikaz analitičkih podataka* obuhvataju složenije radnje koje se mogu razložiti dalje razložiti na pojedinačne slučajeve korišćenja.

Detaljni UML dijagram za slučaj korišćenja *pregledanje kataloga knjiga* je prikazan na sledećoj slici:



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



## Kratak opis slučajeva korišćenja

### Prijavljivanje

Kratak opis: Prijavljivanje korisnika u aplikaciji u cilju pristupa specifičnim funkcijama koje zahtevaju autorizaciju.

Akteri koji iniciraju slučaj korišćenja: Korisnik biblioteke, Radnik biblioteke, Menadžer biblioteke.

### Registracija

Kratak opis: Registracija korisnika u aplikaciji u cilju pristupa specifičnim funkcijama koje zahtevaju autorizaciju.

Akteri koji iniciraju slučaj korišćenja: Korisnik biblioteke.

### Pregled kataloga knjiga po književnoj vrsti

Kratak opis: Prikaz kataloga knjiga filtriranih po književnoj vrsti.

Akteri koji iniciraju slučaj korišćenja: Korisnik biblioteke, Radnik biblioteke, Menadžer biblioteke.

### Pregled kataloga knjiga po književnom rodu

Kratak opis: Prikaz kataloga knjiga filtriranih po književnoj rodu.

Akteri koji iniciraju slučaj korišćenja: Korisnik biblioteke, Radnik biblioteke, Menadžer biblioteke.

### Pregled kataloga knjiga po žanru

Kratak opis: Prikaz kataloga knjiga filtriranih po žanru.

Akteri koji iniciraju slučaj korišćenja: Korisnik biblioteke, Radnik biblioteke, Menadžer biblioteke.

### Pregled kataloga knjiga po autoru

Kratak opis: Prikaz kataloga knjiga filtriranih po autoru.

Akteri koji iniciraju slučaj korišćenja: Korisnik biblioteke, Radnik biblioteke, Menadžer biblioteke.

### Pregled kataloga knjiga po jeziku

Kratak opis: Prikaz kataloga knjiga filtriranih po jeziku.

Akteri koji iniciraju slučaj korišćenja: Korisnik biblioteke, Radnik biblioteke, Menadžer biblioteke.

### Upis u red čekanja za knjigu

Kratak opis: Upis u red čekanja za odabranu knjigu ukoliko trenutno nije dostupna.

Akteri koji iniciraju slučaj korišćenja: Korisnik biblioteke, Radnik biblioteke, Menadžer biblioteke.

### Generisanje predloga za čitanje

Kratak opis: Prikaz stranice sa predlogom knjiga za čitanje na osnovu ukusa korisnika.

Akteri koji iniciraju slučaj korišćenja: Korisnik biblioteke.

### Diskusija o knjigama

Kratak opis: Diskusija o pročitanim knjigama u obliku komentara.

Akteri koji iniciraju slučaj korišćenja: Korisnik biblioteke.

### Dobijanje e-mail obaveštenja

Kratak opis: Omogućavanje dobijanja obaveštenja vezanih za korisnike na e-mail.

Akteri koji iniciraju slučaj korišćenja: Korisnik biblioteke.

### Naplata članarine

Kratak opis: Plaćanje mesečne članarine kod radnika biblioteke.

Akteri koji iniciraju slučaj korišćenja: Korisnik biblioteke, Radnik biblioteke.

### Koordinacija radnika

Kratak opis: Premeštanje radnika iz jednog ogranka biblioteke u drugi.

Akteri koji iniciraju slučaj korišćenja: Menadžer biblioteke.

### Prikaz kompletnog izveštaja

Kratak opis: Prikaz rada biblioteke u vidu izveštaja sa svim relevantnim podacima.

Akteri koji iniciraju slučaj korišćenja: Menadžer biblioteke.

### Prikaz podataka za ogranak

Kratak opis: Prikaz analitičkih podataka za izabrani ogranak.

Akteri koji iniciraju slučaj korišćenja: Menadžer biblioteke.

### Prikaz podataka o radniku

Kratak opis: Prikaz analitičkih podataka o učinku radnika.

Akteri koji iniciraju slučaj korišćenja: Menadžer biblioteke.

# 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.

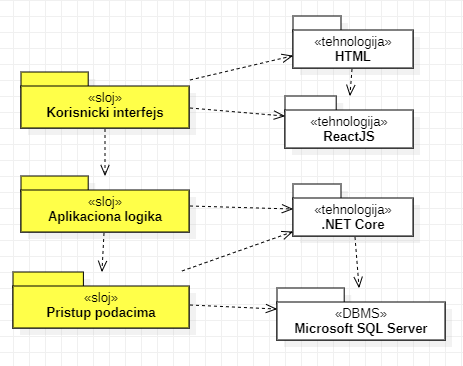
Logički pogled na LMS Web aplikaciju obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket *Korisnički interfejs* sadrži Web stranice, ReactJS 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 .NET Core datoteke zadužene za realizaciju funkcionalnosti specifičnih za domen sistema koji se razvija.

Paket *Pristup podacima* sadrži .NET Core datoteke 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 portala. U njemu su sadržane sve HTML, multimedijalni sadržaji i ReactJS skripte koje generišu HTML stranice preko kojih korisnici komuniciraju sa sistemom.

Sloj korisničkog interfejsa zavisi od sloja aplikacione logike, kao i paketa ReactJS.

### Aplikaciona logika

sloj

Sloj aplikacione logike je srednji sloj u troslojnoj arhitekturi LMS Web aplikacije. Sadrži .NET Core datoteke koje realizuju funkcionalnost karakterističnu za domen primene aplikacije i uspostavljaju vezu između korisničkog interfejsa i sloja za pristup podacima.

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

### Pristup podacima

sloj

Sloj za pristup podacima se nalazi na dnu troslojne arhitekture i sadrži .NET Core datoteke zadužene za pribavljanje, dodavanje i ažuriranje podataka koji se čuvaju u Microsoft SQL Server bazi podataka.

Ovaj sloj ne zavisi od drugih slojeva, ali je zavisan od paketa .NET Core i Microsoft SQL Server 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

### ReactJS

tehnologija

Tehnologija ReactJS omogućava kreiranje dinamičkih Web stranica za prikaz u Web čitaču i prikaz formatiranih informacija i realizaciju formi za unos i ažuriranje podataka.

### .NET Core

tehnologija

Tehnologija .NET Core omogućava izgradnju serverskog dela Web aplikacije uz pomoć raznih alata i podržanih tehnologija i daje interfejs prema bazi podataka u vidu Entity Frameworka.

### Microsoft SQL Server

DBMS

Microsoft SQL Server predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju LMS Web aplikacije.

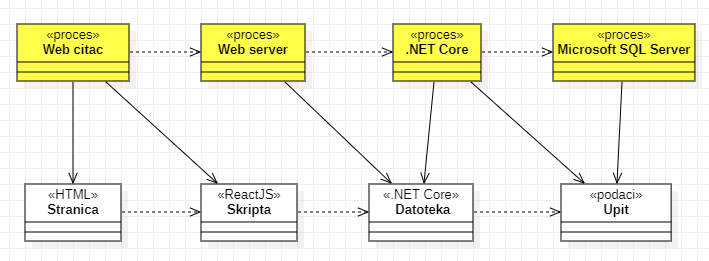
# Pogled na procese

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 LMS kao Web aplikacije.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju LMS Web aplikacije. Dijagram je opšteg tipa i može se primeniti na bilo koju Web aplikaciju zasnovanu na .NET Core i Microsoft 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 ReactJS skript, Web server inicira izvršenje .NET Core procesa koji obrađuje odgovarajući skript i generiše sadržaj koji se vraća čitaču. Web server može paralelno da inicira veći broj .NET Core procesa.

### .NET Core

.NET Core proces obavlja posao obrade zadatog ReactJS skripta i generiše odgovarajući tekstualni sadržaj koji Web server šalje Web čitaču. Za izvršenje ReactJS skripta ovaj proces može da zahteva usluge Microsoft SQL Servera. Komunikacija između .NET Core procesa i Microsoft SQL Servera se obavlja preko prosleđivanja upita i vraćanja rezultata.

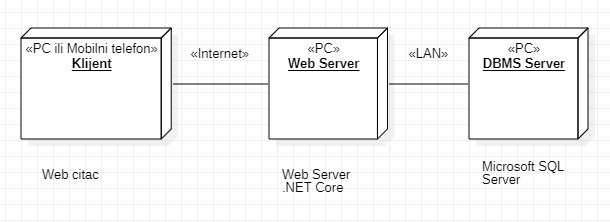
### Microsoft SQL Server

Microsoft SQL Server je proces koji izvršava funkcionalnost Microsoft 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 LMS Web aplikacije.



## Klijent

Pristup LMS Web aplikaciji se obavlja preko klijentskih računara ili mobilnih telefona na kojima 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. Pored osnovnog procesa koji realizuje funkcionalnost Web servera, na ovom računaru mogu da se izvršavaju i procesi .NET Core koji vrše obradu zadatih ReactJS 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 Microsoft SQL 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).

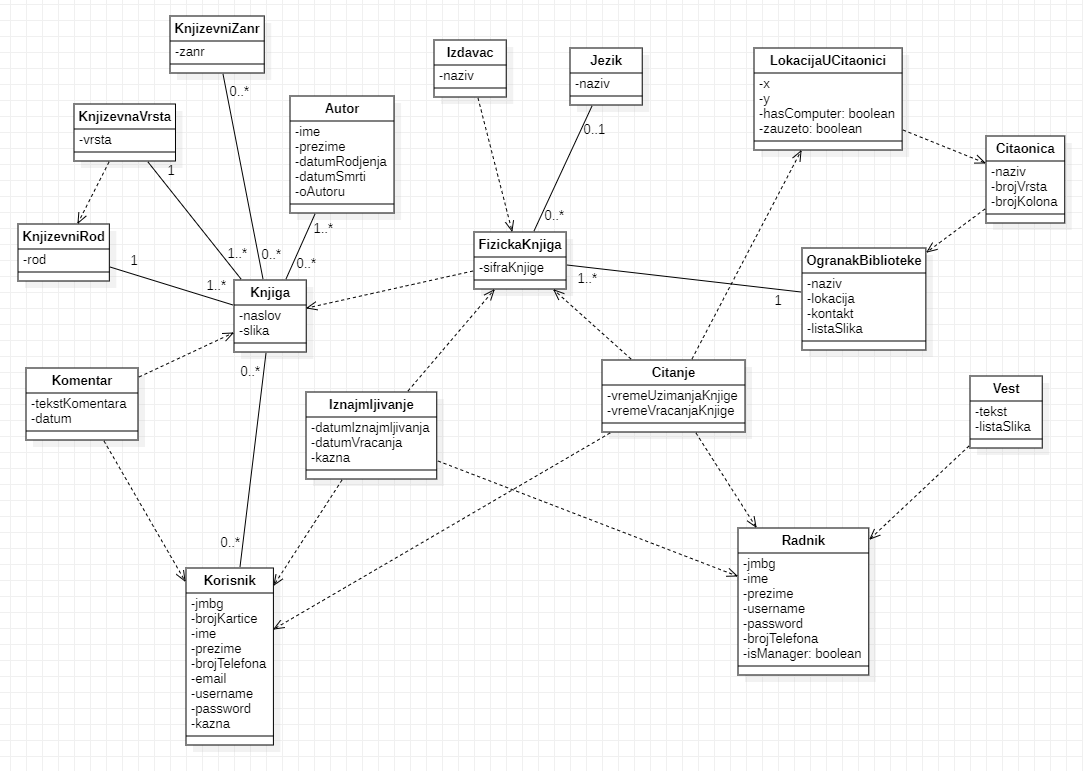
# Pogled na implementaciju sistema

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

## Model domena

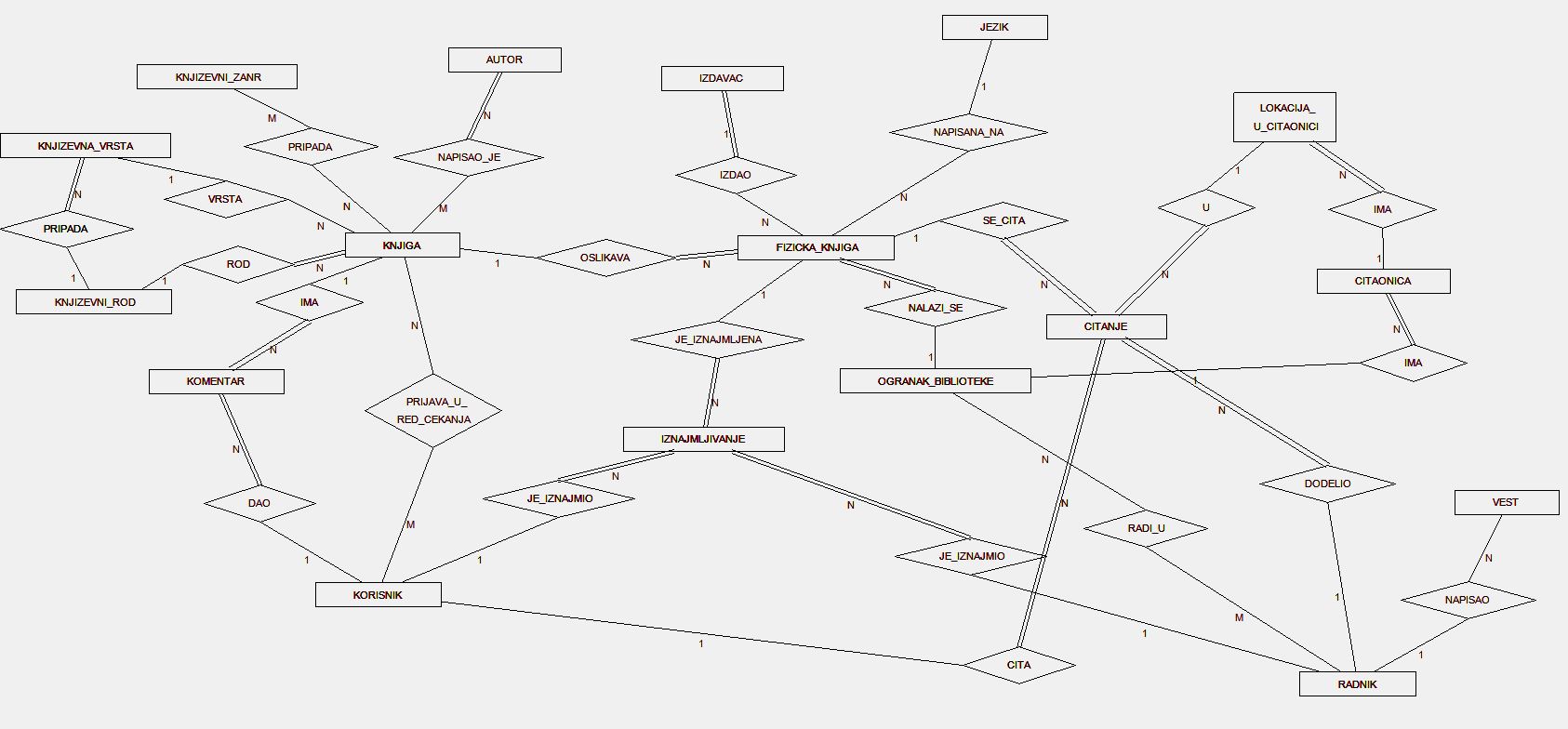
Model domena za koji se LMS Web aplikacija projektuje ilustrovan je UML dijagramom klasa. U njemu su prikazane domenske klase, neki od njihovih atributa, kao i veze koje se mogu identifikovati između njih.

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 je prikazana na sledećem dijagramu. Baza podataka je kreirana u Azure Data Studio. Dijagram je kreiran korišćenjem *EER* programa, dok je migracija na Microsoft SQL Server obavljena pomoću Entity Frameworka.

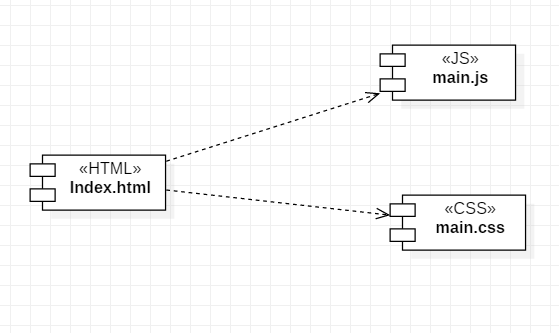


## Komponente sistema

Komponente sistema LMS Web aplikacije su HTML dokumenti, ReactJS skripte i .NET datoteke čiji će pregled biti dat po arhitekturnim slojevima. Za ilustraciju će biti korišćeni UML dijagrami komponenti, ali i dijagrami klasa. U slučajevima gde je HTML dokument prikazan kao klasa atributi predstavljaju ulazne podatke koji se uzimaju iz GET ili POST dela HTTP poruke, dok metodi predstavljaju funkcije definisane u okviru ReactJS skripta.

### Komponente korisničkog interfejsa

Dizajn korisničkog interfejsa je obuhvaćen sa tri komponente:



Za korisničku aplikaciju **index.html** je komponenta koja u sadejstvu sa **main.js** komponentom

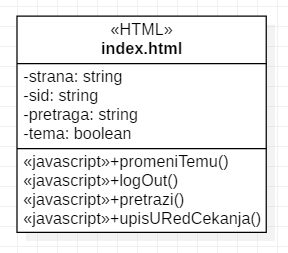
prikazuje sadržaj koji biva izgrađen dinamički. Sadržaj će zavisiti od toga koji je korisnik

ulogovan i koji sadržaj korisnik pretražuje(parametri koji utiču ime strane, podaci pretrage,

id sesije...). Komponenta main.js je napravljena da iscrtava ReactJS komponente(promene na virtual dom-u se prenose na realan dom). Komponenta **main.css** posle bundle-ovanja će sadržati sve stilove, tj.

stilove svih ReactJS komponenti pa samim tim će definisati celokupan izgled korisničkog interfejsa.

Napomena ove tri komponente se generisu posle bundle-ovanja svih zavisnosti.



Značenje atributa je sledeće:

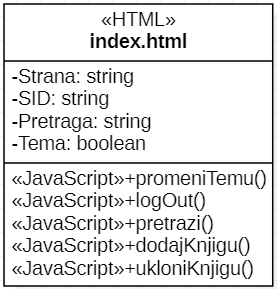
-strana identifikator stranice (početna, knjige, nalog...)

-pretraga dodatnih informacija kojom se dalje određuje izgled strane

-sid identifikator sesije kojom je korisnik ulogovan

-tema - ova promenljiva određuje koja tema je trenutno primenjena

Aplikacija namenjena menadžerima sadrži 3 fajla nakon build-ovanja, kao i aplikacija namenjena korisnicima. Razlika u fajlovima je takva da je kod aplikacije za menadžere interfejs prilagođen potrebama menadžera.



Značenje atributa je sledeće:

-strana identifikator stranice (knjige, forum...)

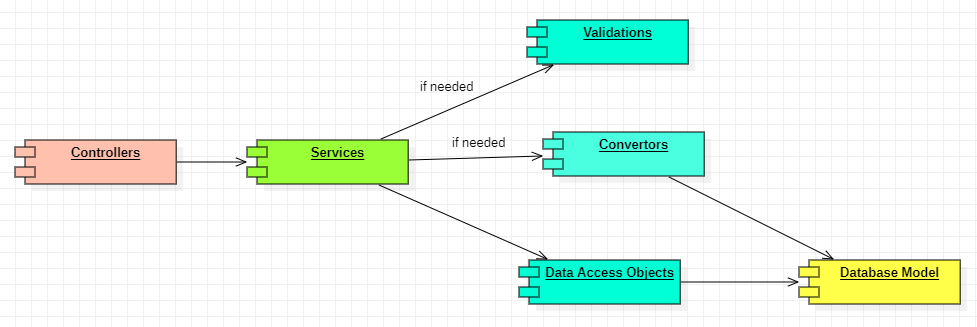
-pretraga dodatnih informacija kojom se dalje određuje izgled strane

-sid identifikator sesije kojom je menadžer ulogovan

-tema ova promenljiva određuje koja tema je trenutno primenjena

### Komponente aplikacione logike

Aplikacionu logiku čine komponente : Controllers, Validations, Convertors, Data Access Objects i Database Model.

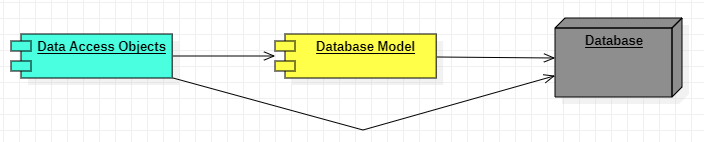


***Controllers*** predstavlja skup svih kontrolera koji prihvataju zahtev klijentske strane aplikacije. Oni šalju poruku za obradu zahteva odgovarajućem servisu iz ***Services***. U slučaju zahteva za unos ili izmenu postojećih podataka mora se uraditi neke od validacija iz ***Validations*** kako ne bi došlo do narušavanja homogenosti sistema. Takođe u tim slučajevima, kada se dodaje ili menja se može uraditi neki od konvertovanja iz ***Convertors*** ulaznih parametara u parametre sa kojima može da se pristupa bazi. Na kraju servis kontaktira jedan ili nekoliko Data Access objekta iz ***Data Access Objects*** koji ima funkciju pristupanja bazi podataka radi izvršenja CRUD operacija.

### Komponente za pristup podacima

Pristup bazi podataka je u potpunosti zatvoren u funkcije koje su definisane u okviru Data Access Object-a.

Samo ti objekti imaju dozvolu pristupa bazi podataka u celom sistemu.



Neke od funkcija obavljaju sledeće zadatke:

* getKorisnikByBrojKartice – preuzima korisnika sa odgovarajućim brojem kartice iz baze podataka
* getFizickeKnjigeByNaslov – vraća sve prisutne knjige u sistemu odgovarajućeg naslova
* createKorisnik – kreira novog korisnika
* platiClanarinu – plaća članarinu određenog korisnika
* ulogujSe – loguje korisnika i vraća SID
* getKnjigeByVrsta – vraća sve knjige odgovarajuće književne vrste
* getKnjigeByRod – vraća sve knjige odgovarajućeg književnog roda
* getKnjigeByZanr – vraća sve knjige odgovarajućeg žanra
* getKnjigeByAutor – vraća sve knjige odgovarajućeg autora
* getKnjigeByJezik – vraća sve knjige na odgovarajućem jeziku
* iznajmiKnjigu – iznajmljuje se korisniku odgovarajuća knjiga
* vratiKnjigu – vraća se odgovarajuća knjiga u biblioteku
* premestiRadnika – premešta odgovarajućeg radnika iz jednog ogranka u drugi
* prikaziPodatkeORadniku – prikazuje izveštaj o odgovarajućem radniku
* ... – 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. LMS Web aplikacija 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.