Cogit

Softver za upravljanje projektima

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 17.04.2021. | 1.0 | Inicijalna verzija | Aleksandar, Ana, Nikola, Mateja |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Sadržaj

1. Cilj dokumenta 5

2. Opseg dokumenta 5

3. Predstavljanje arhitekture 5

4. Ciljevi i ograničenja arhitekture 5

5. Pogled na slučajeve korišćenja 5

5.1 Dijagrami slučajeva korišćenja 6

5.2 Kratak opis slučajeva korišćenja 11

5.2.1 Kreiranje naloga 11

5.2.2 Prijavljivanje na sistem 11

5.2.3 Ažuriranje podataka o sebi 12

5.2.4 Podešavanje izgleda radne površine 12

**5.2.5** Kreiranje tima 12

5.2.6 Dodavanje događaja u kalendar 12

5.2.7 Chat 12

5.2.8 Dodavanje reakcija 12

5.2.9 Dodavanje komentara 12

5.2.10 Prikaz projekata 12

5.2.11 Kreiranje podzadatka u okviru svog zadatka 12

Premeštanje zadataka iz jedne liste u drugu 12

5.2.12 Označavanje zadatka prioritetnim 12

5.2.13 Arhiviranje/brisanje zadatka 13

5.3 Napuštanje tima 13

5.3.1 Kreiranje i dodeljivanje zadatka članovima 13

5.4.1 Kreiranje Projekta 13

6. Pogled na logičku arhitekturu sistema 14

6.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 14

6.1.1 Korisnički interfejs 14

6.1.2 Aplikaciona logika 15

6.1.3 Pristup podacima 15

6.1.4 HTML 15

6.1.5 Nodejs 15

6.1.6 NoSql 15

7. Pogled na procese 15

7.1 Procesi 15

7.1.1 Web čitač 16

7.1.2 Web server 16

7.1.3 CGI-Node 16

7.1.4 NoSQL Server 16

8. Pogled na raspoređivanje sistema 16

8.1 Klijent 16

8.2 Web server 17

8.3 DBMS server 17

9. Pogled na implementaciju sistema 17

9.1 Model domena 17

9.2 Šema baze podataka 18

9.3 Komponente sistema 19

9.3.1 Komponente korisničkog interfejsa 19

9.3.2 Komponente aplikacione logike 21

9.3.3 Komponente za pristup podacima 21

10. Performanse 22

11. Kvalitet 22

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture Cogit aplikacije.

# Opseg dokumenta

Dokument se odnosi na Cogit aplikaciju koji će biti razvijen od strane AmanSoft tima. Namena sistema je rešavanje problema lošeg upravljanja ljudskim resursima i vremenom prilikom realizacije projekata .

# Reference

Spisak korišćene literature:

* 1. Cogit – Predlog projekta, SWE-Cogit-01, V1.0, 2021, AmanSoft.
  2. Cogit – Planirani raspored aktivnosti na projektu, V1.0, 2021, AmanSoft.
  3. Cogit– Plan realizacije projekta, V1.0, 2021, AmanSoft.
  4. Cogit – Vizija sistema, V1.0, 2021, AmanSoft.
  5. Cogit – Specifijkacija zahteva, V1.0, 2021, AmanSoft

# 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. Cogit aplikacija će biti implementiran kao Web aplikacija zasnovana na javascript skripting jeziku i
2. NoSql bazi podataka [4].
3. Klijentski deo Cogit aplikacije će biti optimizovan za sledeće Web čitače: Internet Explorer 6.0 i noviji, Opera 8.0 i noviji, Firefox (Mozilla) kao i Google Chrome [4].
4. Svi zahtevi u pogledu performansi dati u [5] moraju biti uzeti u obzir pri izboru arhitekture i razvoju sistema.
5. Klasifikaciju publikacija treba standardizovati sa postojećom tipologijom propisanom od strane Ministarstva za nauku.

# Pogled na slučajeve korišćenja

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

Slučajevi korišćenja Cogit aplikacije su:

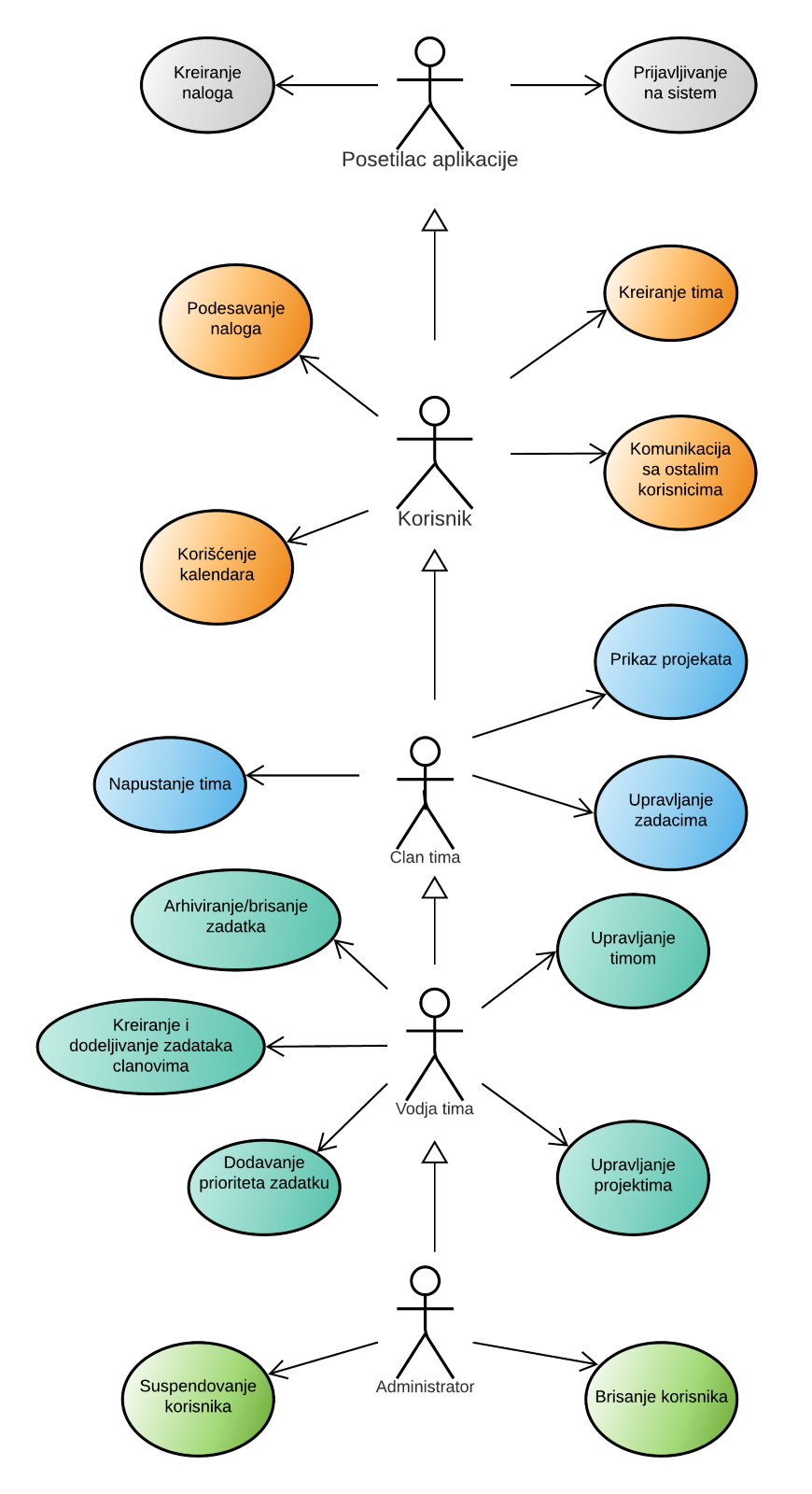
* Kreiranje naloga
* Prijavljivanje na sistem
* Podesavanje naloga
  + Azuriranje podataka o sebi
    - Promena lozinke
    - Promena korisnickog imena
  + Podesavanje izgleda radne povrsine
* Kreiranje tima
* Komunikacija sa ostalim korisnicima
  + Chat
  + Dodavanje reakcija
  + Dodavanje komentara
* Koriscenje kalendara
  + Brisanje dogadjaja
  + Dodavanje dogadjaja
  + Pregled dogadjaja
* Prikaz projekata
* Upravljanje zadacima
  + Kreiranje podzadatka unutar svog zadatka
  + Premestanje zadatka iz jedne liste u drugu
  + Oznacavanje zadatka prioritetnim
* Napustanje tima
* Upravljanje timom
  + Brisanje tima
  + Dodavanje/Izbacivanje clanova
* Upravljanje projektima
  + Kreiranje/brisanje projekta
  + Koriscenje napravljenih sablona
  + Kreiranje/brisanje liste zadataka
  + Dodela tagova projektima
* Dodavanje prioriteta zadatku
* Kreiranje i dodeljivanje zadataka clanovima
* Arhiviranje/brisanje zadataka
* Brisanje korisnika
* Suspendovanje korisnika

Ove slučajevi korišćenja mogu da iniciraju posetilac aplikacije, korisnik, clan tima, vodja tima, administrator.

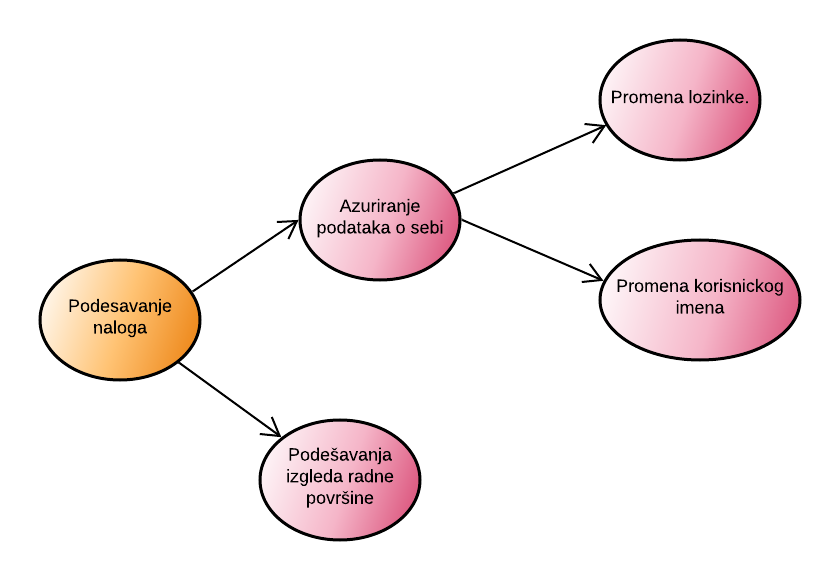
Ove slučajevi korišćenja mogu da iniciraju posetilac portala, član laboratorije, šef laboratorije, vođa projekta ili administrator.

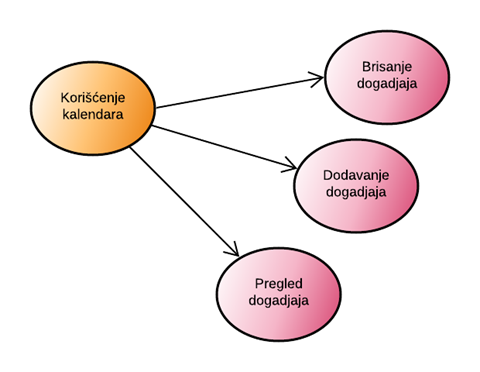
## Dijagrami slučajeva korišćenja

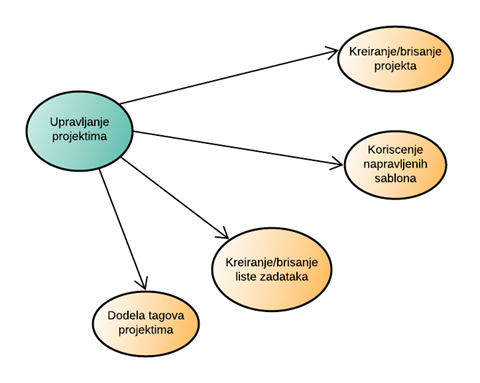
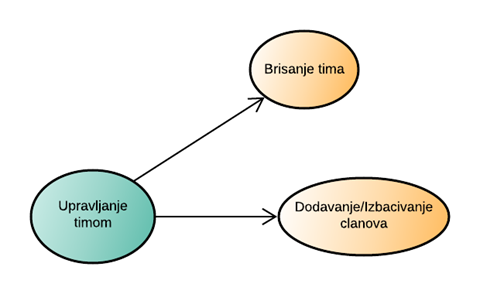
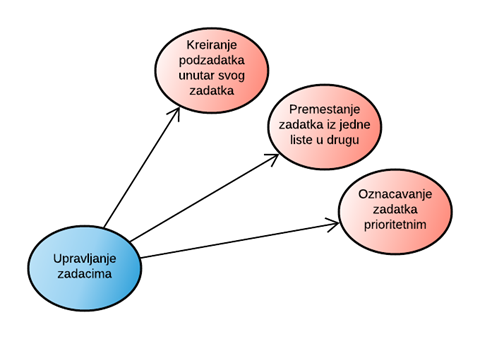
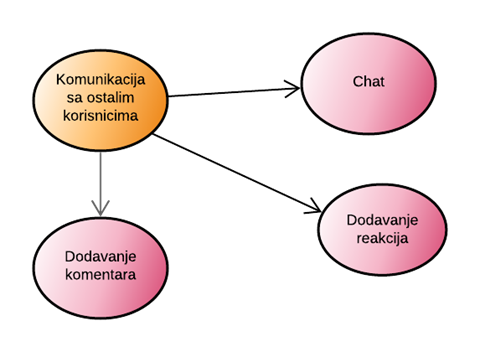
Osnovni UML dijagram koji prikazuje korisnike i slučajeve korišćenja Cogit aplikacije prikazan je na sledećoj slici:



Detaljni UML dijagrami za slučajeve korišćenja: upravljanje timom, upravljanje projektima, komunikacija sa ostalim korisnicima, korišćenje kalendara, podešavanje naloga i upravljanje zadacima su prikazani u nastavku:







## Kratak opis slučajeva korišćenja

### Kreiranje naloga

Kratak opis: Prikaz stranice aplikacije sa formom za kreiranje naloga..

Akteri koji iniciraju slučaj korišćenja: Posetilac aplikacije.

### Prijavljivanje na sistem

Kratak opis: Prikaz stranice aplikacije sa formom za prijavljivanje.

Akteri koji iniciraju slučaj korišćenja: Posetilac aplikacije.

### Ažuriranje podataka o sebi

Kratak opis: Prikaz stranice sa informacijama o tom korisniku.

Akteri koji iniciraju slučaj korišćenja: Član tima, Vođa tima, Administrator.

### Podešavanje izgleda radne površine

Kratak opis: Korisnik može da kreira promeni izgled svoje radne površine.

Akteri koji iniciraju slučaj korišćenja: Član tima, Vođa tima, Administrator.

### Kreiranje tima

Kratak opis: Korisnik može da kreira tim.

Akteri koji iniciraju slučaj korišćenja: Član tima, Vođa tima, Administrator.

### Dodavanje događaja u kalendar

Kratak opis: Dodavanje događaja u kalendaru tog korisnika .

Akteri koji iniciraju slučaj korišćenja: Član tima, Vođa tima, Administrator.

### Chat

Kratak opis: Grupno ili direktno razmenjivanje poruka..

Akteri koji iniciraju slučaj korišćenja: Član tima, Vođa tima, Administrator.

### Dodavanje reakcija

Kratak opis: Mogućnost dodavanja reakcija koje su vidljive ostalim članovima tima.

Akteri koji iniciraju slučaj korišćenja: Član tima, Vođa tima, Administrator.

### Dodavanje komentara

Kratak opis: Mogućnost dodavanja komentara koji su vidljivi ostalim članovima tima.

Akteri koji iniciraju slučaj korišćenja: Član tima, Vođa tima, Administrator.

### Prikaz projekata

Kratak opis: Izbor načina prikaza projekata.

Akteri koji iniciraju slučaj korišćenja: Član tima, Vođa tima, Administrator.

### Kreiranje podzadatka u okviru svog zadatka

Kratak opis: Korisnik kreira novi zadatak unutar već postojećeg zadatka.

Akteri koji iniciraju slučaj korišćenja: Član tima, Vođa tima, Administrator.

### Premeštanje zadataka iz jedne liste u drugu

Kratak opis: Korisnik ima mogućnost prebacivanja svojih zadatka iz jedne liste u drugu.

Akteri koji iniciraju slučaj korišćenja: Član tima, Vođa tima, Administrator.

### Označavanje zadatka prioritetnim

Kratak opis: Korisnik može da oznaci zadatak prioritetnim.

Akteri koji iniciraju slučaj korišćenja: Član tima, vođa tima, Administrator

### Arhiviranje/brisanje zadatka

Kratak opis: Vođa tima može da obriše ili arhivira zadatak.

Akteri koji iniciraju slučaj korišćenja: Vođa tima, Administrator

### Napuštanje tima

Kratak opis : Korisnik može da napusti tim.

Akteri koji iniciraju slučuja korišćenja : Član tima, Vođa tima, Administrator.

### Kreiranje i dodeljivanje zadatka članovima

Kratak opis: Vođa tima može da kreira zadatak i da ga dodeli bilo kom članu tima..

Akteri koji iniciraju slučaj korišćenja: Vođa tima, Administrator.

### Dodeljivanje prioriteta zadatku

Kratak opis **:** Vođa tima može da dodeli prioritet zadatku u okviru projekta.

Akteri koji iniciraju slučaj korišćenja : Vođa tima, Administrator.

### Dodavanje izbacivanje članova iz tima

Kratak opis: Vođa tima može da dodaje ili izbacuje članove.

Akteri koji iniciraju slučaj korišćenja: Vođa tima, Administrator.

### Kreiranje Projekta

Kratak opis: Izmena podataka o projektu od strane vođe projekta.

Akteri koji iniciraju slučaj korišćenja: Vođa projekta, Administrator.

### Brisanje projekta

Kratak opis: Vođa tima može da izbriše postojeći projekat..

Akteri koji iniciraju slučaj korišćenja: Vođa tima, Administrator.

### Korišćenje napravljenih šablona

Kratak opis: Vođa tima može da koristi šablone prilikom kreiranja projekta..

Akteri koji iniciraju slučaj korišćenja: Vođa tima, Administrator.

### Kreiranje i brisanje liste zadataka

Kratak opis: Administrator može da obrise naloge korisnika aplikacije.

Akteri koji iniciraju slučaj korišćenja: Vođa tima, Administrator.

### Dodela tagova projektima

Kratak opis: Vođa tima može da dodeli projektima određene tagove..

Akteri koji iniciraju slučaj korišćenja: Vođa tima, Administrator.

### Brisanje Korisnika

Kratak opis: Administrator može da obrise naloge korisnika aplikacije.

Akteri koji iniciraju slučaj korišćenja: Vođa tima, Administrator.

### Suspendovanje Korisnika

Kratak opis: Administrator može da suspenduje naloge korisnika aplikacije.

Akteri koji iniciraju slučaj korišćenja: Vođa tima, 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.

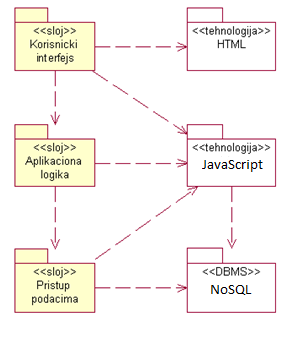
Logički pogled na Cogit aplikacije obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket *Korisnički interfejs* sadrži Web stranicu, javascript 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 javascript skripte zadužene za realizaciju funkcionalnosti specifičnih za domen sistema koji se razvija.

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

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

### Aplikaciona logika

sloj

Sloj aplikacione logike je srednji sloj u troslojnoj arhitekturi Cogit aplikacije. Sadrži JavaScript skripte koje realizuju funkcionalnost karakterističnu za domen primene portala i uspostavljaju vezu između korisničkog interfejsa i sloja za pristup podacima.

Ovaj sloj zavisi od sloja za pristup podacima i JavaScript paketa.

### Pristup podacima

sloj

Sloj za pristup podacima se nalazi na dnu troslojne arhitekture i sadrži JavaScript skripte zadužene za pribavljanje, dodavanje i ažuriranje podataka koji se čuvaju u NoSql bazi podataka.

Ovaj sloj ne zavisi od drugih slojeva, ali je zavisan od paketa JavaScript-a i NoSql 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.

### Nodejs

tehnologija

Tehnologija Nodejs-a obezbeđuje mehanizam za izvršavanje JavaScript skripti na strani servera.

### NoSql

DBMS

NoSQL predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju Cogit aplikacije.

# Pogled na procese

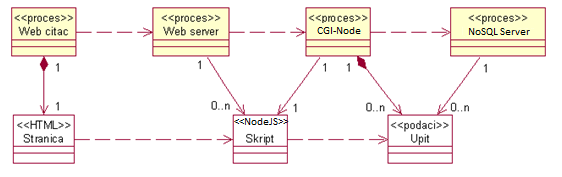
U ovom odeljku je sadržan pogled na procesnu arhitekturu sistema. Ovaj opis treba da sadrži specifikaciju različitih zadataka (procesa i niti) uključenih u rad sistema. Takođe je potrebno dati dijagrame koji pokazuju njihovu interakciju i konfiguraciju. Dodela objekata i klasa na određene zadatke takođe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

Web aplikacije zasnovane na JavaScriptu-u imaju relativno jednostavan procesni model koji je u potpunosti pod kontrolom Web servera.

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

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju Cogit aplikacije. Dijagram je opšteg tipa i može se primeniti na bilo koju Web aplikaciju zasnovanu na JavaScript-u i NoSQL 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.

### CGI-Node

CGI-Node proces obavlja posao obrade zadate JavaScript skripte i generiše odgovarajući tekstualni sadržaj koji Web server šalje Web čitaču. Za izvršenje JavaScript skripte ovaj proces može da zahteva usluge NoSQL servera-a. Komunikacija između CGI-Node procesa i NoSQL servera se obavlja preko prosleđivanja upita i vraćanja rezultat.

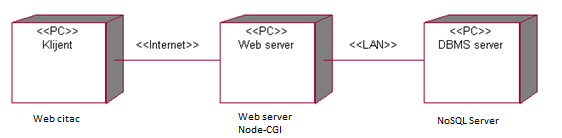
### NoSQL Server

NoSQL Server je proces koji izvršava funkcionalnost NoSQL 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 Cogit aplikacije.



## Klijent

Pristup Cogit aplikaciji se obavlja preko klijentskih računara 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 CGI-Node koji vrše obradu zadatih JavaScript 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 NoSQL 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).

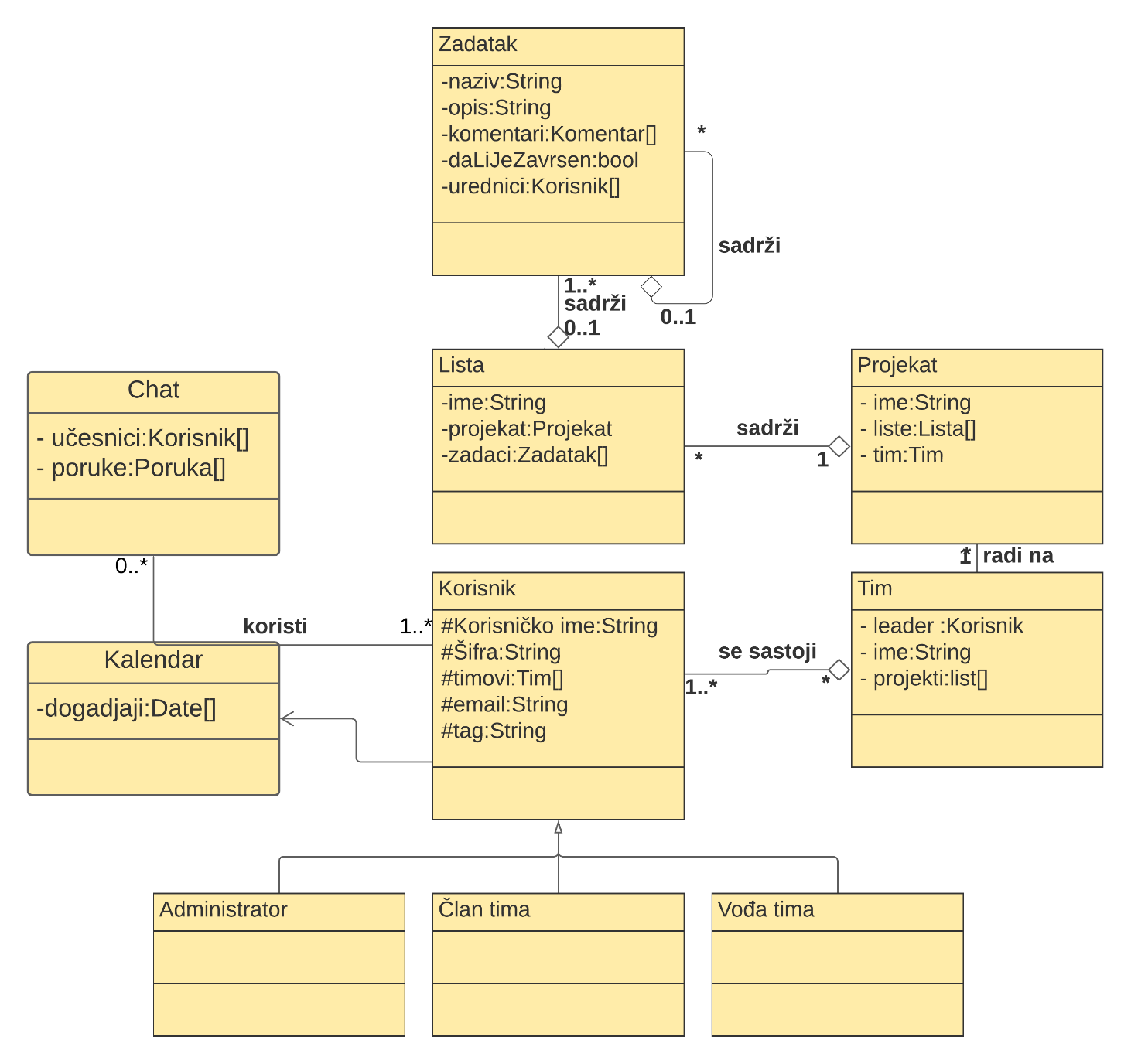
# Pogled na implementaciju sistema

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

## Model domena

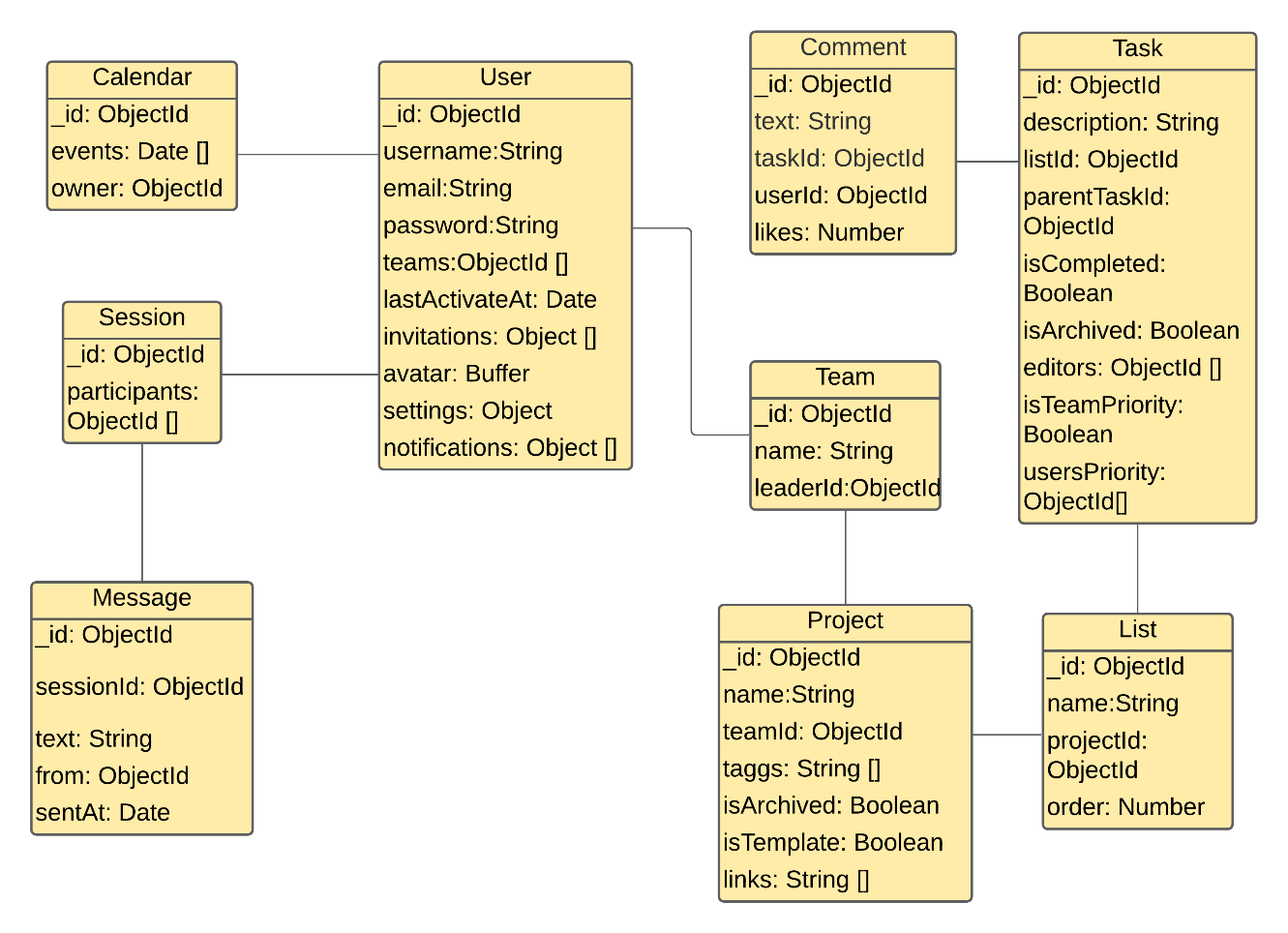
Model domena za koji se Cogit aplikacija projektuje je ilustrovan UML dijagramom klasa. U njemu su prikazane domenske klase, neki od njihovih atributa, kao i veze koje se mogu identifikovati između njih.

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



## Šema baze podataka

Detaljna šema baze podataka je prikazana na sledećem dijagramu. Baza podataka i dijagram su kreirani korišćenjem *LucidChart-a*.

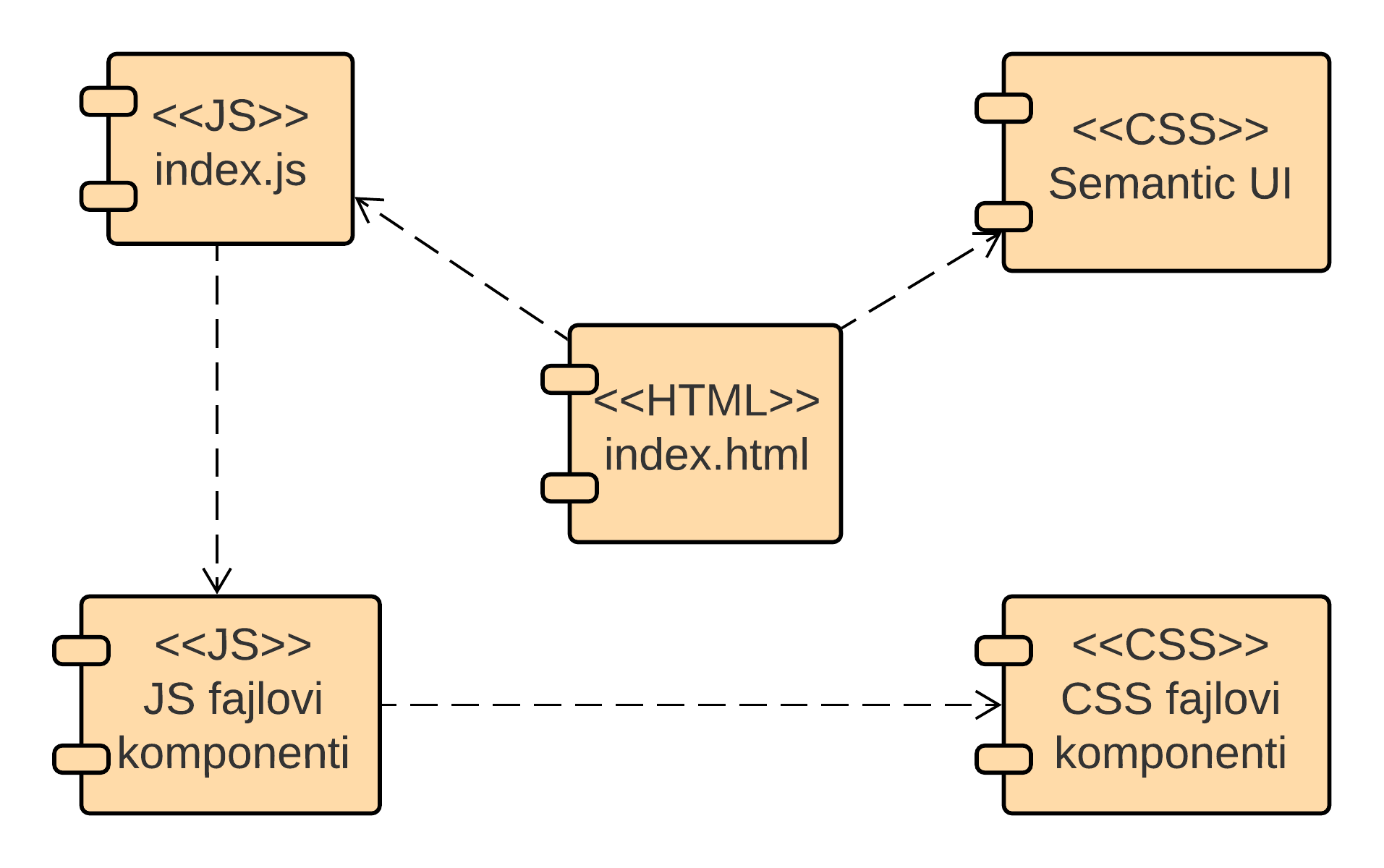


## Komponente sistema

Komponente sistema Cogit aplikacije su JavaScript skripte čiji će pregled biti dat po arhitekturnim slojevima. Za ilustraciju će biti korišćeni UML dijagrami komponenti, ali i dijagrami klasa. U slučajevima gde je JavaScript skripta prikazana 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 sledećim komponentama:



Komponenta **index.html** implementira stranicu aplikacije čiji sadržaj može da varira od parametra koji joj se proslede u zahtevu.

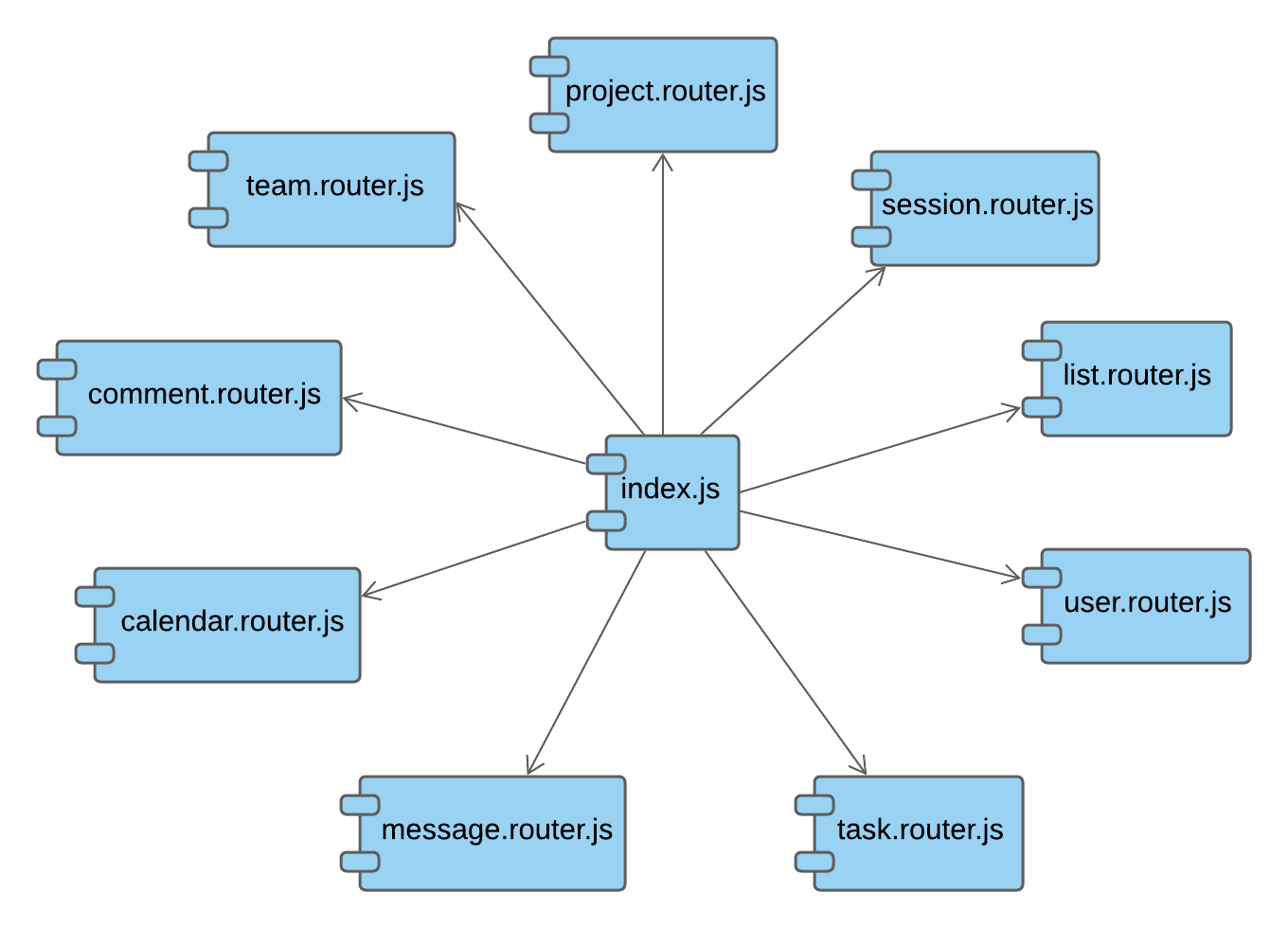
**JS fajl** implementira konkretnu React komponentu.

**CSS fajl** definiše stil za odgovarajuću React komponentu.

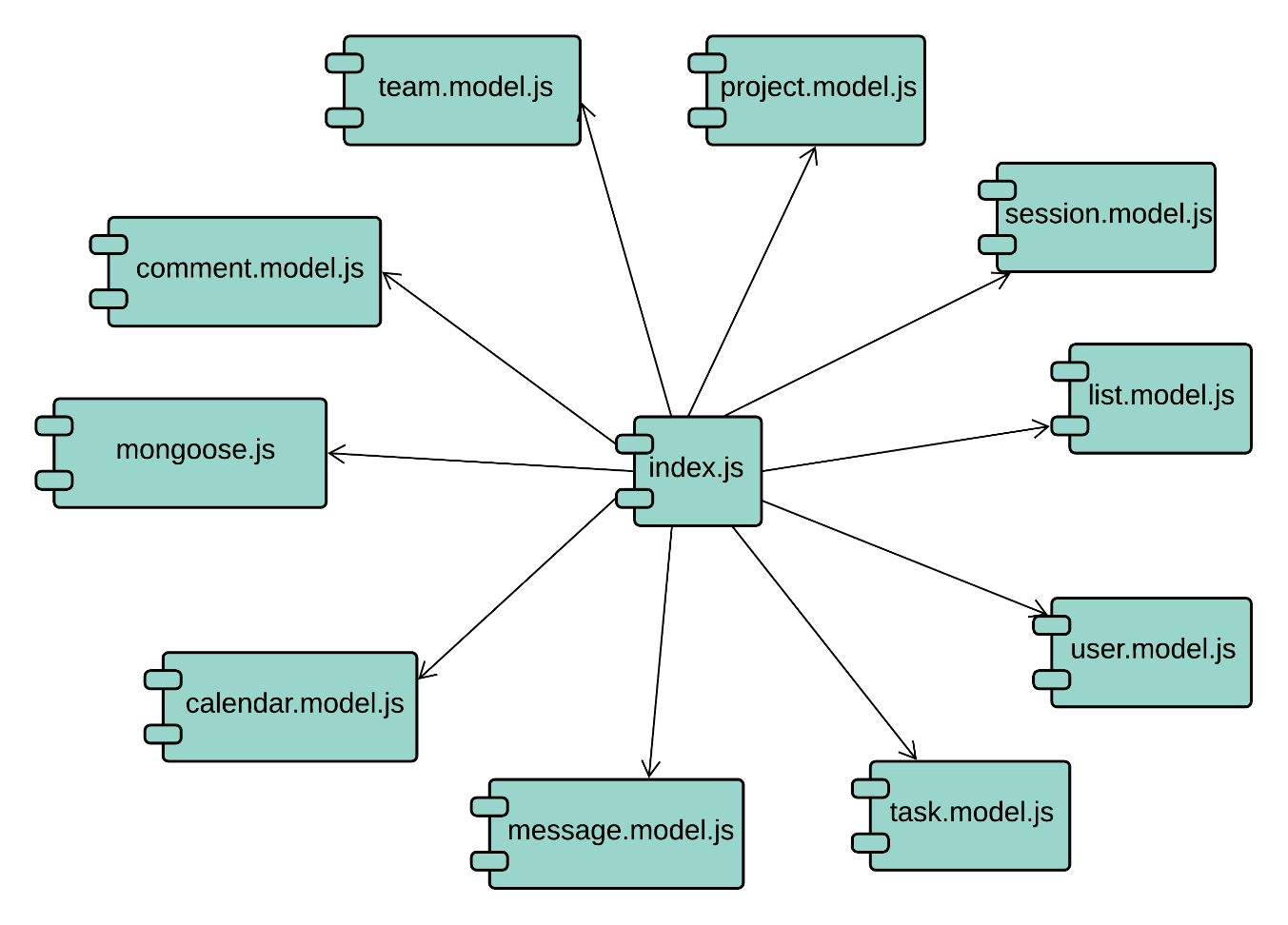
**Semantic UI** predstavlja okruženje koje nam daje bogatu bibilioteku već definisanih stilova koje možemo da integrišemo u naš projekat.

Komponenta **index.js** predstavlja osnovni js fajl Cogit aplikacije.

### Komponente aplikacione logike

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

### Komponente za pristup podacima

Na sledećem dijagramu su prikazane komponente za pristup podacima:

Navedene funkcije obavljaju sledeće zadatke:

* LoginUserHandler - loguje korisnika i vraća usera i token
* DeleteUserHandler - briše i vraća usera
* AssignUserHandler – dodeljuje task useru, preko prosleđenih id-a
* SendTeamInviteHandler – šalje pozivnicu za pristup timu korisniku, preko prosleđenih id-a
* GetMyProjectsHandler – vraća sve projekte usera, projekte koji su deo timova čiji je član korisnik dobijen autentifikacijom (korišćenjem tokena).
* GetProfileHandler – vraća usera dobijenim autentifikacijom (korišćenjem tokena)
* CreateCommentHandler – upisuje prosleđeni komentar u bazu ukoliko je autentifikacija uspesna.
* ... – 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. Cogit aplikacija će biti dostupna 24 časa dnevno, 7 dana u nedelji. Vreme kada aplikacija nije dostupna ne sme da pređe 10%.
2. Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 120 sati.