**Uvod**

Informacione tehnologije predstavljaju ključni faktor razvoja I napretka savremenog društva. Zahvaljujući kontinuiranom unapređivanju informacionih tehnologija, funkcionisanje velikih poslovnih sistema je znatno olakšano.

Konsultujući se sa mentorom diplomskog rada, došli smo na ideju kreiranja veb aplikacije za zakazivanje konsultacija. Oblast informacionih tehnologija podrazumeva kontinuirano učenje, napredovanje i razvijanje, a za to je pored upornosti i truda potrebna i pomoć iskusnijih. Studenti imaju tu prednost da oko svih nejasnoća mogu zakazati konsultacije i pitati predmenog profesora oko bilo kakvog problema. Međutim, kada ogroman broj studenata treba pomoć, potrebno je sve to uskladiti i postići, a ništa se ne može učiniti kako valja bez dobre organizacije. Upravo zbog toga se javlja potreba da se napravi aplikacija koja će nastavnicima pružiti mogućnost da vode evidenciju o svim svojim dogadjajima, bile to konsultacije, nastava ili nešto treće, a s druge strane studentima omogućiti jednostavan i brz način zakazivanja konsultacija, uz mogućnost postavljanja propratnih materijala i drugih pogodnosti u odnosu na tradicionalan način zakazivanja putem mejla.

Sistem je razvijen kao jednostranična veb aplikacija u kojoj nastavnici i studenti mogu komunicirati oko termina konsultaija. Omogućena je registracija studenta, logovanje studenta. Kada se student uloguje, on može vršiti pretragu nastavnika a zatim izborom na pojedinog nastavnika može pregledati njegov kalendar dogadjaja. Kada student utvrdi koji su termini konsultacija pomoću lepo dizajniranog kalendara dogadjaja za tog nastavnika, student ima mogućnost da klikom na dogadjaj iz kalendara zakaže konsultacije kod tog nastavnika. Student popunjava neophodna polja i postavlja prilog ukoliko želi a zatim pritiskom na dugme Zakazi konsultacije zakazuje konsultacije kod tog nastavnika. Student takodje ima mogućnost pregleda svih do sada zakazanih njegovih konsultacija. Ukoliko je prilikom zakazivanja student pogrešio prilikom unosa nekog od polja, student može izmeniti zakazane konsultacije klikom na određene konsultacije iz tabele.

Da bi se student registrovao, neophodno je da pre toga postoji u bazi. To znači da u sistem mogu da se registruju samo studenti koji su studenti Fakulteta organizacionih nauka. Prilikom registracije, sistem proverava da li na fakultetu postoji student sa unetim brojem indeksa, a zatim ukoliko postoji, da li je student sa tim brojem indeksa vec registrovan.

Nastavnici se takođe nalaze u bazi i nastavnici već imaju svoje naloge u bazi, tako da su oni već registrovani i nije omogućena registracija za njih. Kada se nastavnik uloguje u sistem, on najpre bira kalendar i školsku godinu čijim podacima želi da pristupi i da menja. Nako izbora kalendara, nastavniku se prikazuje kalendar sa svim dogadjajima koje je do tada zakazao. Nastavnik ima mogućnost generisanja novih dogadjaja, kao i pregled svih zakazanih konsultacija kod njega. Takođe, nastavnik može, klikom na pojedinačni dogadjaj iz kalendara da pregleda zakazane konsultacije za taj termin konsultacija.

**2. Java**

Јава (енгл. Java, изговор: јава, ∗џава) је објектно-оријентисани програмски језик, који је развила компанија Sun Microsystems почетком 1990-их година.

Многи концепти Јаве су засновани на језику Оберон, Никлауса Вирта, творца Паскала, Модуле и других језика, и Ханспетера Месенбека. Избацили су концепт модула и увели пакете какве данас знамо, који се ослањају на фајл систем и увели формално концепт класа из објектно-оријентисане парадигме. Осим тога, језик има синтаксу сличну језицима C и C++, али је много строжи при превођењу, дизајниран тако да буде независан од платформе, и са поједностављеним управљањем меморијом. Претпоставља се да је ово урађено због популарности језика C, али и због једноставности неких структура. Прва верзија је званично објављена 1995. године.

Данас је Јава подразумевани језик у изради мобилних апликација за Андроид уређаје.[1]

2.1. Java platforma

Java platforma se razlikuje od većine drugih platformi u smislu da je to softverski zasnovana platforma koja radi na drugim hardverski zasnovanim platformama. Čine je dve komponente: Runtime okruženje i API (Application Programming Interface). Java kod se može pokrenuti na različitim platformama poput Windows, Linux, Sun Solaris, Mac/OS itd. Kod napisan u java jeziku se prvo kompajlira, a onda konvertuje u bajtkod (eng. Bytecode). Bajtkod predstavlja platformno nezavisan kod i može se pokrenuti na različitim platformama. Ovaj princip je nazvan “Napiši jednom, pokreni svuda” (eng. Write Once Run Anywhere (WORA). [1]

**OracleJET**

Oracle JavaScript Extension Toolkit (JET) омогућује развојним програмерима пружање модуларног "open source" алата темељеног на сувременим начелима дизајна и развоја JavaScript, CSS3 и HTML5. Oracle ЈЕТ усмјерен је на средње до напредне JavaScript програмере који раде на апликацијама на страни клијента. То је збирка ЈаваСцрипт библиотека отвореног кода, заједно с низом Oраклових ЈаваСцрипт библиотека које га чине једноставним и учинковитијим за изградњу апликација које конзумирају и комуницирају с Oracleovim производима и услугама, посебице Орацле Cloud услугама.

**O Oracle JET-u**

Oracle JET je skup Oracleovih i open source JavaScript biblioteka koji omoogućava jednostavniju i efikasniju izgradnju klijentskih aplikacija i hibridnih mobilnih aplikacija koje se zasnivaju na JavaScript-u, HTML5-u i CSS-u.

Oracle JET je dizajniran da zadovolji sljedeće potrebe aplikacije:

• Dodavanje interaktivnosti na postojeću stranicu

• Kreiranje end-to-end klijentskih veb aplikacija koristeći JavaScript, HTML5 i CSS

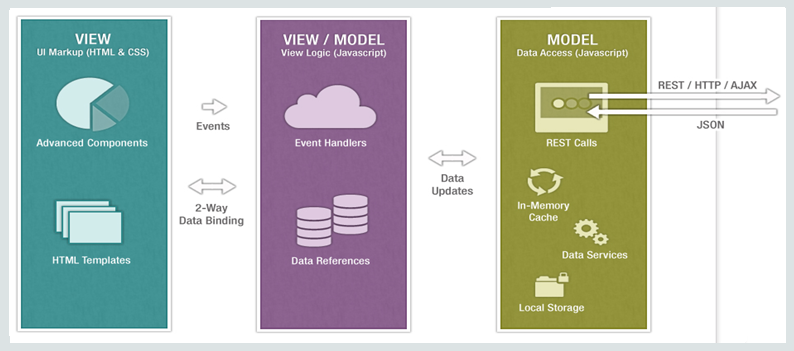
• Responzivan dizajn

• Kreiranje hibridnih mobilnih aplikacija koje u potpunosti izgledaju kao iOS, Android ili Windows aplikacije.

**Oracle JET Framework**

Oracle JET podržava Model-View-ViewModel (MVVM) patern

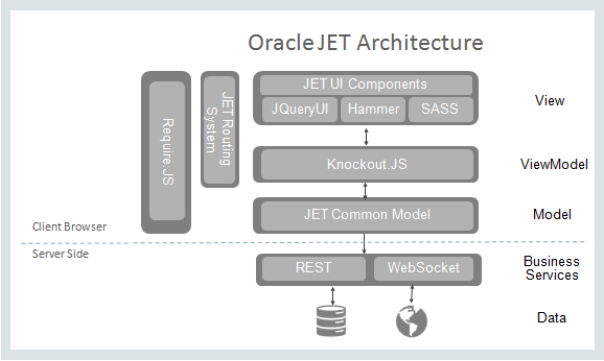
U MVVM-u, Model predstavlja podatke aplikacije, a View predstavlja prezentaciju podataka. ViewModel izlaže podatke iz Modela na View i održava stanje aplikacije.



*Podržava* ***Model-View-ViewModel (MVVM****) arhitekturu, gde Model predstavlja podatke i poslovnu logiku aplikacije, View korisnički interfejs, dok je ViewModel odgovoran za konverziju objekata iz modela. U suštini, ViewModel je više Model nego View i upravlja velikim delom, ako ne i celom prezentacionom logikom.*

Za podršku MVVM dizajna, Oracle JET je izgrađen kao modularni frejmvork koji

uključuje skup *third-party* biblioteka kao i skup falova, skripti,i biblioteka koje pruža Oracle.



**2.Šta je uključeno u Oracle JET**

Oracle JET distribucija uključuje Oracle JET frejmvork i sve third-partzy biblioteke koje frejmvork koristi.

Naime, Oracle JET uključuje sledeće datoteke i biblioteke:

• CSS i SCSS datoteke za Alta temu

• Minimalizirane i debug verzije Oracle JET biblioteka

• Alate za vizualizaciju podataka (Data Visualiyation Tools), CSS i JavaScript

• Knockout i Knockout Mapping biblioteke

• jQuery biblioteke

• jQuery UI biblioteke

• RequireJS

• Crossroads.js

• Hammer.js

**3.Oracle JET i responzivan veb dizajn**

Odgovarajući web dizajn opisuje koncept dizajna koji koristi tekuće rešetke, skalabilne

slika i medijskih upita kako bi prikazali alternativne izglede uvjetno na temelju

vrsta medija. S odzivnim web dizajnom možete konfigurirati aplikaciju

vizualno privlačan na širokom rasponu uređaja, od malih telefona do

widescreen stolnih računala.

Oracle JET nudi sustav rešetke od 12 stupaca i oblike rasporeda obrazaca koji uključuju

stilova za male, srednje, velike i ekstra velike ekrane ili uređaje. Sustavi rešetke

i klase oblika koriste medijske upite za postavljanje stila na temelju veličine zaslona ili

uređaja i možete ih koristiti za prilagodbu izgleda stranice na temelju korisničkih postavki

potrebe.

Pored toga, medijski upiti čine osnovu za responzivne klase pomagača koji prikazuju ili

sakriti sadržaj, poravnati tekst ili plutaju sadržaj na temelju veličine zaslona. Oni su također osnova

za responzivni JavaScript koji učitava sadržaj uvjetno ili postavlja opciju komponente

na temelju veličine zaslona.

**4. Oracle JET i RequireJS**

RequireJS je JavaScript datoteka i učitavač modula koji olakšava upravljanje bibliotekama

i osmišljen je za poboljšanje brzine i kvaliteta koda. RequireJS implementira Asynchronous Module Definition (AMD) API koji

pruža mehanizam za asinhrono učitavanje modula i njegovih dipendencija. Modularna organizacija Oracle JET-a omogućuje razvojnim programerima da učitaju podskup

potrebne feature-a bez potrebe za izvršavanjem require() poziva za svaki referencirani

objekat. Svaki Oracle JET modul predstavlja jedno funkcionalno područje(AREA) frejmvorka, i to

obično definiše više od jednog JavaScript objekta. Modul ojs/ojcore definiše

oj namespace objekat, a svi kasnije učitani moduli učitavaju svoje

objekte unutar oj namespace-a.

**Oracle JET Module Organization**

..možda treba ubaciti

**5. Oracle JET i jednostranična aplikacija**

Jednostranična aplikacija (Single-page application) ili jednostranični interfejs je veb aplikacija smeštena u okviru jedne veb stranice sa ciljem pružanja lakše interakcije. U jednostraničnim aplikacijama sav potreban kod - HTML, JavaScript i CSS - se preuzima u okviru učitavanja osnovne stranice aplikacije, ili se odgovarajući resursi učitavaju dinamički i po potrebi dodaju stranici, obično kao posledica neke akcije od strane korisnika. Karakteriše je to da se stranica ne učitava ponovno ni u jednoj tački procesa, niti se prebacuje na neku drugu stranicu. Stranica ne mora značiti samo jedan pregled, novije veb tehnologije pružaju mogućnost percepcije i navigacije kroz više logičkih stranica u okviru iste aplikacije. Interakcija sa jednostranom aplikacijom često uključuje dinamičku komunikaciju sa veb-serverom u pozadini. Termin jednostranična aplikacija skovao je Stiv Jen (eng. Steve Yen) 2005. godine

Oracle Jet ima podršku za single-page aplikacije koristeći oj.Router klasu za virtuelnu navigaciju.

**Single Page aplikacija** je veb aplikacija koja učitava jednu HTML stranicu a onda dinamički ažurira tu stranicu kada korisnik interaktuje sa aplikacijom. Koristi AJAX i HTML5 da kreira repsponzivne veb aplikacije bez ponovnog učitavanja stranica.

Umesto da se celokupan sadržaj aplikacije nalazi u index.html fajlu, Oracle Jet koristi Knockout data-bind sintaksu I ojModul-e.

Oracle JET frejmvork uključuje Knockout.js biblioteku, ojModule binding i

Oj.Router frejmvork klasu koje možete koristiti za stvaranje jednostraničnih aplikacija kako biste

simulirli izgled desktop aplikacija. OjModule binding za

Knockout.js upravlja zamenom sadržaja unutar određene regije stranice.

Oj modul binding može raditi zajedno s oj.Router klasom, koji pruža

podršku za rutiranje u Oracle JET aplikacijama

**5.1 Dizajniranje jednostranične aplikacije koristeći Oracle JET**

Oracle JET frejmvork uključuje Knockout za razdvajanje sloja modela od

prezentacionog sloja i upravljanje interakcijom između njih. Korištenje Knockout-a, Oracle

JET ojModule binding-e, i Oracle JET oj.Router klasu, možete

izradite jednostranične aplikacije koje izgledaju(imaju look and feel) kao desktop aplikacije.

Jednostranične aplikacije (SinglePage Applications) obično se koriste za simulaciju izgleda i dojma

samostalnih desktop aplikacija. Umesto korištenja više web stranica s vezama

između njih za navigaciju, aplikacija koristi jednu web stranicu koja je učitana

samo jednom. Ako se stranica promeni zbog interakcije korisnika, samo se dinamički menja deo stranice.

Oracle JET okvir uključuje podršku za jednostranične aplikacije pomoću

oj.Router klase za virtuelnu navigaciju na stranici, ojModule bindings za

upravljanje View-templejtima (HTML fragmentima) i viewModel skriptama (js skripte),kao i Knockout za razdvajanje sloja modela od prezentacionog sloja i upravljanje interakcijom između njih.

U Oracle JET SPA-ojModule-ojRouter aplikaciji prikazanoj na donjoj slici,

index.html stranica prikazuje jednostavnu stranicu s toolbar-om za odabir Home,

Book, i Tables sadržaja.

**5.2 Oracle JET i ojModule**

Oracle JET - ojModule nam omogućava promenu sadržaja unutar stranice bez ponovnog učitavanja stranice. To nam omogućava da upravljamo navigacijom u jednostraničnim aplikacijama.

OjModule nam takođe omogućava da sadržaj koji se nalazi u okviru index.html-a čuvamo u posebnim HTML templejtima u folderu js/views, kao i da JavaScript fajlove za odgovarajuće templejte u čuvamo u js/viewmodels. HTML templejti će se dinamički prikazivati u zavisnosti od korisnikove interakcije sa aplikacijom kroz navigaciju.

OjModule i oj.Router klasa omogućavaju nam da postignemo virtuelnu navigaciju kroz jednostraničnu aplikaciju. Kada korisnik promeni trenutni templejt kroz navigaciju, promeniće se I stanje rutera a ojModule će automatski učitati i promeniti sadržaj modula koji je specificiran u RouterState objektu.

**5.2 Knockout**

Knockout je JavaScript biblioteka koja vam pomaže da stvorite bogate, responzivne korisničke interfejse s jednostavnim odgovoarajućim modelom podataka. Kad god imate delove korisničkog interfejsa koji se dinamički ažuriraju (npr. Menja se zavisno od korisničkih akcija), Knockout vam može olakšati da to jednostavnije implementirate.

Oracle JET framework koristi Knockout da bi razdvojio model od korisničkog interfejsa i upravlja interakcijom izmedju njih.

**5.2Using Knockout.js Templates and the ojModule Binding**

Oracle JET Cookbook i mnogi primjeri primene koriste prirodni mehanizam templejtova koji je uključen sa Knockout.js i Oracle JET ojModule za upravljanje povezivanjem(binding). Knockout templejti omogućuju vam da ponovno upotrijebite kod i pružite jednostavan mehanizam za izgradnju struktura korisničkog interfejsa kao funkciju vašeg viewModel-a.

Na primjer, SPA-ojModule-ojRouter sample jednostranična aplikacija upotrebljava pet templejta za definisanje HTML markup-a aplikacije. Sledeća slika prikazuje glavnu stranicu jednostranične Oracle JET aplikacije, a strelice upućuju na nazive templejta koji određuju koji će sadržaj biti prikazan u aplikaciji.

<https://docs.oracle.com/middleware/jet112/jet/developer/img/GUID-B191E088-B8A9-42B2-9DFE-D132EA52C553-default.png>

# Biblioteke

**jQuery**: JavaScript biblioteka sa mnoštvo korisnih mogućnosti za REST, AJAX, animacije i upravljanje JSON-om.

**Knockout JS**: Omogućava two-way data binding.

**RequireJS**: JavaScript fajl i učitavač modula koji olakšava upravljanje bibliotekama. Dizajniran je da poboljša brzinu i kvalitet koda. OracleJET koristi RequireJS po default-u kada napravimo novi projekat i neophodan je ukoliko planiramo da koristimo Oracle Jet internacionalizaciju, komponente za data vizuelizaciju i slično.

**Syntactically Awesome Style Sheets (SASS)**: Nasleđuje CSS3

**Apache Cordova**: Za razvoj mobilnih aplikacija.

**Oracle Alta UI**: Korisnički dizajn za moderan izgled aplikacije.

**6. Studijski primer**

Životni ciklus razvoja softvera može sadržati različite faze, u zavisnosti od izabrane metode. U studijskom primeru koji sledi primenjena je pojednostavljena Larmanova metoda koja obuhvata pet faza [15]:

1.Prikupljanje zahteva od korisnika – u ovoj fazi definišu se zahtevi i osobine koje softverski sistem treba da ispuni. Zahtevi se mogu podeliti u funkcionalne, koji definišu osnovne funkcije koje sistem treba da pruži, i nefunkcionalne, koji predstavljaju dodatne zahteve kao što su performanse, pouzdanost itd. Rezultat ove faze predstavlja model slučajeva korišćenja koji sadrži elemente strukture i ponašanja sistema.

2. Analiza – u ovoj fazi opisuju se logička struktura i ponašanje sistema, tj. poslovna logika sistema. Za opis strukture sistema koriste se konceptualni i relacioni model, a za opis ponašanja dijagrami sekvenci i sistemske operacije.

3. Projektovanje – faza u kojoj se opisuje arhitektura softverskog sistema. Arhitektura se sastoji iz tri nivoa: korisničkog interefjsa, aplikacione logike i skladišta podataka. Aplikacionu logiku čine kontroler aplikacione logike, poslovna logika i broker baze podataka.

4. Implementacija – u ovoj fazi kreiraju se implementacione komponente koje realizuju arhitekturu opisanu u fazi projektovanja. Komponente se mogu realizovati korišćenjem različitih tehnologija, kao što su Java, .NET itd.

5. Testiranje – poslednja faza u kojoj se svaka implementaciona komponenta testira korišćenjem test slučajeva, test procedura i test komponenti. Svaka od ovih faza biće opisana u nastavku rada na primeru aplikacije “Pitaj-odgovori”.

7.1. Korisnički zahtevi

7.1.1. Verbalni opis

Potrebno je napraviti veb aplikaciju za zakazivanje konsultacija u kojoj će nastavnici i studenti moći komunicirati oko zakazivanje termina konsultaija. Aplikaciju mogu koristiti samo studenti i nastavnici Fakulteta organizacionih nauka. Potrebno je da postoji registracija studenata. Da bi se student registrovao, neophodno je da unese svoj broj indeksa. Sistem nakon toga proverava da li na fakultetu postoji student sa unetim brojem indeksa. Ukoliko postoji takav student i ukoliko student sa tim brojem indeksa nije već registrovan, registracija je uspešna. Studentu je je nakon toga omogućeno logovanje u sistem. Potrebno je obezbediti različite funkcionalnosti u zavisnosti od toga da li je ulogvan student ili nastavnik.

Kada se u sistem uloguje student, on može videti sve nastavnike koji se nalate u sistemu kao i vršiti pretragu nastavnika. Izborom na pojedinog nastavnika, student može pregledati njegov kalendar dogadjaja, odnosno termine kada se održavaju konsultacije kod tog nastavnika. Kada student utvrdi koji su termini konsultacija pomoću lepo dizajniranog kalendara dogadjaja za tog nastavnika, student ima mogućnost da klikom na dogadjaj iz kalendara zakaže konsultacije kod tog nastavnika. Student popunjava neophodna polja i postavlja prilog ukoliko to želi a zatim pritiskom na dugme Zakaži konsultacije zakazuje konsultacije kod tog nastavnika. Student takodje ima mogućnost pregleda svih konsultacija koje je do tada zakazao. Ukoliko je prilikom zakazivanja student pogrešio prilikom unosa nekog od polja, student može izmeniti zakazane konsultacije klikom na određene konsultacije iz tabele.

Nastavnici već imaju svoje naloge i u aplikaciji ne postoji registracija za njih. Kada se nastavnik uloguje u sistem, on najpre bira kalendar i školsku godinu a zatim mu je omogućen pristup podacima za izabranu škoslku godinu. Nastavniku se prikazuje kalendar sa svim dogadjajima koje je do tada zakazao. Nastavnik ima mogućnost generisanja novih konsultacija, kao i pregled svih do tada zakazanih konsultacija kod njega. Takođe, nastavnik može, klikom na pojedinačni dogadjaj iz kalendara da pregleda zakazane konsultacije za taj termin konsultacija.

7.1.2. Slučajevi korišćenja

U sistemu su identifikovani sledeći slučajevi korišćenja u zavisnosti od uloge:

Nastavnik je aktor sledećih slučajeva korišćenja:

1. Registracija novog korisnika

2. Pristup korisnika (provera tokena)

3. Izmena podataka korisničkog naloga

4. Postavljanje pitanja

5. Selekcija pitanja po kategoriji

6. Pretraga pitanja

7. Pregled pitanja korisnika

8. Unos novog odgovora

9. Davanje glasa odgovoru - “Izabran”

10. Davanje glasa odgovoru - “Prijavljen”

11. Davanje glasa odgovoru - “Pozitivan”

12. Davanje glasa odgovoru - “Negativan”

Student je korisnik koji je aktor sledećih slučajeva korišćenja:

13. Dodeljivanje privilegija korisniku

14. Brisanje pitanja

15. Brisanje prijavljenih odgovora

16. Dodavanje nove kategorije

Radi izbegavanja opširne dokumentacije, u okviru modela slučajeva korišćenja biće prikazani samo sledeći slučajevi korišćenja aplikacije:

1. Registracija studenta

2.

## СК1: Случај коришћења – Регистрација студента

**Назив СК**

Регистрација студента

### Актори СК

Студент

### Учесници СК

Студент и систем (програм)

**Предуслов**: Систем је укључен. Систем приказује форму за регистрацију студента.

### Основни сценарио СК

1. Студент **уноси** податке за кориснички налог. (АПУСО)
2. Студент **контролише** да ли је коректно унео податке. (АНСО)
3. Студент **позива** систем да запамти податке и региструје га. (АПСО)
4. Систем **памти** податке о корисничком налогу. (СО)
5. Систем **приказује** Студенту поруку: “Регистрација је успешна“. (ИА)

Алтернативна сценарија

5.1 Уколико систем не може да региструје студента он приказује поруку: “Дошло је до грешке приликом регистрације”. (ИА)

## СК2: Случај коришћења – Логовање корисника

**Назив СК**

**Логовање корисника**

### Актори СК

Корисник

### Учесници СК

Корисник и систем (програм)

**Предуслов**: Систем је укључен. Систем приказује форму за логовање корисника.

### Основни сценарио СК

1. Корисник **уноси** своје податке неопходне за логовање на систем. (АПУСО)
2. Корисник **контролише** да ли је коректно унео податке. (АНСО)
3. Корисник **позива** систем да га улогује. (АПСО)
4. Систем **претражује кориснике** користећи унете податке о кориснику. (СО)
5. Систем **приказује** Кориснику почетну страницу са обавештењем о успешности логовања (ИА)

Алтернативна сценарија

5.1 Уколико систем не може да улогује студента он Студенту приказује поруку са грешком до које је дошло”. (ИА)

## СК3: Случај коришћења – Претрага наставника

**Назив СК**

Претрага наставника

### Актори СК

Студент

### Учесници СК

Студент и систем (програм)

**Предуслов**: Систем је укључен. Систем приказује форму за претраживање наставника.

### Основни сценарио СК

1. Студент **уноси** вредност по којој претражује наставнике. (АПУСО)
2. Студент **позива** систем да нађе наставнике по задатој вредности. (АПСО)
3. Систем **тражи** наставнике по задатој вредности. (СО)
4. Систем приказује Студенту наставнике по задатој вредности унутар листе за приказ наставника. (ИА)
5. Студент **бира** наставника за којег жели да види податке. (АПУСО)
6. Студент **позива** систем да учита податке о изабраном наставнику. (АПСО)
7. Систем **учитава** податке о изабраном наставнику. (СО)
8. Систем приказује Студенту календар консултација за изабраног наставника. (ИА)

Алтернативна сценарија

4.1 Уколико систем не може да нађе наставнике он приказује Студенту празну листу за приказ наставника. (ИА)

8.1 Уколико систем не може да учита календар консултација за изабраног наставника, он приказује Студенту поруку: „Систем не може да учита календар консултација за изабраног наставника.“

## СК4: Случај коришћења – Заказивање консултација

### Назив СК

Заказивање консултација

### Актори СК

Студент

### Учесници СК

Студент и систем (програм)

**Предуслов**: Систем је укључен и Студент je улогован под својом шифром. Систем приказује форму за заказивање консултација. На страни су учитане све консултације за изабраног наставника.

### Основни сценарио СК

1. Студент **уноси** податке за нове консултације. (АПУСО)
2. Студент **контролише** да ли је коректно унео податке за нове консултације. (АНСО)
3. Студент **позива** систем да запамти податке о новим консултацијама. (АПСО)
4. Систем **памти** податке о заказаним консултацијама. (СО)
5. Систем **приказује** Студенту заказане консултације и поруку: “Заказане консултације “. (ИА)

Алтернативна сценариа

5.1 Уколико систем не може да закаже консултације он приказује Студенту поруку “Систем не може да закаже консултације”. (ИА)

## СК5: Случај коришћења – Брисање заказаних консултација

### Назив СК

Брисање заказаних консултација

### Актори СК

Студент

### Учесници СК

Студент и систем (програм)

**Предуслов**: Систем је укључен и Студент je улогован под својом шифром. Систем приказује форму за рад са заказаним консултацијама. На страни су учитане све заказане консултације за улогованог студента.

### Основни сценарио СК

1. Студенту **бира** заказане консултације које жели да обрише. (АПУСО)
2. Студенту **позива** систем да обрише заказане консултације. (АПСО)
3. Систем **брише** заказане консултације. (СО)
4. Систем **приказује** Студенту поруку: “Систем је отказао заказане консултације.” (ИА)

Алтернативна сценарија

4.1 Уколико систем не може да обрише заказане консултације он приказује Студенту поруку “Систем не може да обрише заказане консултације”. (ИА)

## СК6: Случај коришћења – Генерисање термина консултација

### Назив СК

Генерисање термина консултација

### Актори СК

Наставник

### Учесници СК

Наставник и систем (програм)

**Предуслов**: Систем је укључен и Наставник je улогован под својом шифром. Систем приказује форму за генерисање термина консултација.

### Основни сценарио СК

1. Наставник **уноси** податке за нови термин консултација. (АПУСО)
2. Наставник **контролише** да ли је коректно унео податке за нови термин консултација. (АНСО)
3. Наставник **позива** систем да запамти податке о новом термину консултација. (АПСО)
4. Систем **памти** податке о новом термину консултација. (СО)
5. Систем **приказује** Наставнику запамћени термин консултација и поруку: “Систем је запамтио консултације“. (ИА)

Алтернативна сценариа

5.1 Уколико систем не може да запамти нови термин консултација он приказује Наставнику поруку “Систем не може да запамти нови термин консултација”. (ИА)

## СК7: Случај коришћења – Отказивање термина консултација

### Назив СК

Отказивање термина консултација

### Актори СК

Наставник

### Учесници СК

Наставник и систем (програм)

**Предуслов**: Систем је укључен и Наставник je улогован под својом шифром. Систем приказује форму за рад са консултацијама. На страни су учитани сви термини консултација за улогованог наставника.

### Основни сценарио СК

1. Наставник **бира** термин консултација које жели да откаже. (АПУСО)
2. Наставник **позива** систем да откаже термин консултација и пошаље мејл студентима који су заказали консултације за тај термин са обавештењем о отказу. (АПСО)
3. Систем **брише** термин консултација. (СО)
4. Систем **приказује** Наставнику поруку: “Систем је отказао термин консултација.” (ИА)

Алтернативна сценарија

4.1 Уколико систем не може да откаже термин консултација систем приказује Наставнику поруку “Систем не може да обрише термин консултација”. (ИА)

# Анализа

Фаза анализе описује пословну логику софтверског система, односно његову

структуру и понашање. Главни дијаграми који се креирају у овој фази су:

1. Концептуални модел – служи за опис структуре система;

2. Секвенцијални дијаграми система – служе за опис понашања система;

3. Уговори – служе за опис понашања система.

### 7.2.1. Понашање софтверског система - Системски дијаграми секвенци

Понашање софтверског система описује се помоћу системских дијаграма секвенци и

системских операција.

Системски дијаграм секвенци служи да за конкретан издвојени случај коришћења прикаже

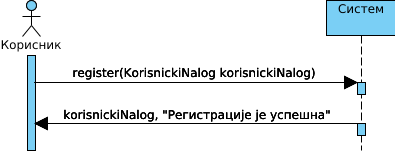
редослед догађаја који представљају интеракцију између корисника и софтверског

система

## ДС1: Дијаграм секвенци случаја коришћења – Регистрација студента

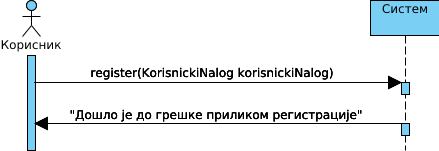
### Основни сценарио СК

1. Студент **позива** систем да запамти податке и региструје га. (АПСО)
2. Систем **приказује** Студенту поруку: “Регистрација је успешна“. (ИА)



Алтернативна сценарија

5.1 Уколико систем не може да региструје студента он приказује поруку: “Дошло је до грешке приликом регистрације”. (ИА)



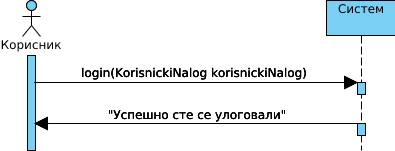
Са наведених секвенцних дијаграма уочавају се следеће системске операције:

1. сигнал register(KorisnickiNalog korisnickiNalog);

## СК2: Случај коришћења – Логовање корисника

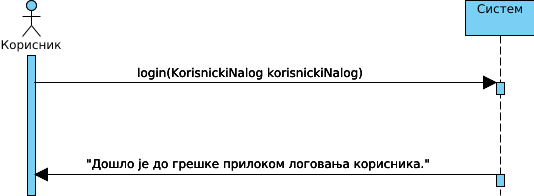
### Основни сценарио СК

1. Корисник **позива** систем да га улогује. (АПСО)
2. Систем **приказује** Кориснику почетну страницу са обавештењем о успешности логовања (ИА)



Алтернативна сценарија

2.1 Уколико систем не може да улогује студента он Студенту приказује поруку "Дошло је до грешке приликом логовања корисника”. (ИА)



## СК3: Случај коришћења – Претрага наставника

### Основни сценарио СК

1. Студент **позива** систем да нађе наставнике по задатој вредности. (АПСО)
2. Систем приказује Студенту наставнике по задатој вредности унутар листе за приказ наставника. (ИА)
3. Студент **позива** систем да учита податке о изабраном наставнику. (АПСО)
4. Систем приказује Студенту календар консултација за изабраног наставника. (ИА)

Алтернативна сценарија

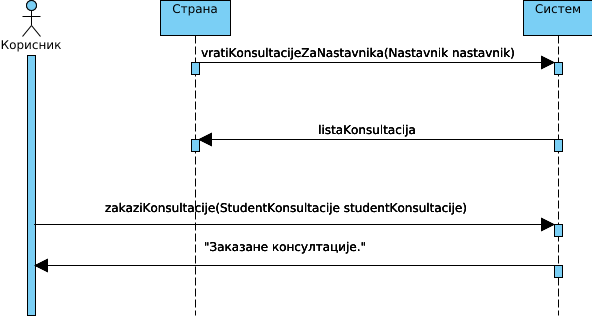
2.1 Уколико систем не може да нађе наставнике он приказује Студенту празну листу за приказ наставника. (ИА)

4.1 Уколико систем не може да учита календар консултација за изабраног наставника, он приказује Студенту поруку: „Систем не може да учита календар консултација за изабраног наставника.“

## СК4: Случај коришћења – Заказивање консултација

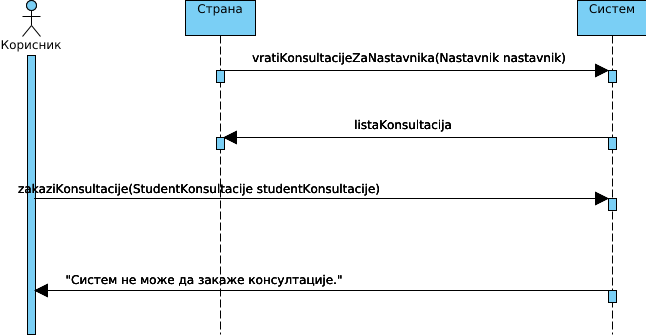
### Основни сценарио СК

1. Студент **позива** систем да запамти податке о новим консултацијама. (АПСО)
2. Систем **приказује** Студенту заказане консултације и поруку: “Заказане консултације “. (ИА)



Алтернативна сценариа

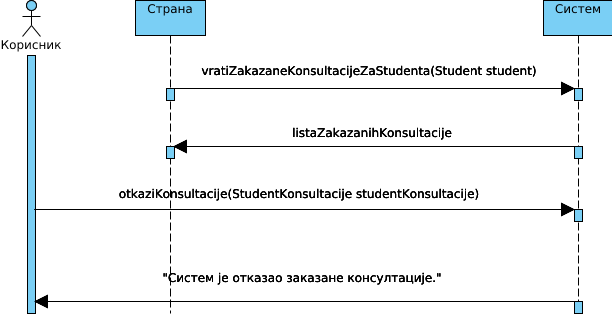
2.1 Уколико систем не може да закаже консултације он приказује Студенту поруку “Систем не може да закаже консултације”. (ИА)



## СК5: Случај коришћења – Брисање заказаних консултација

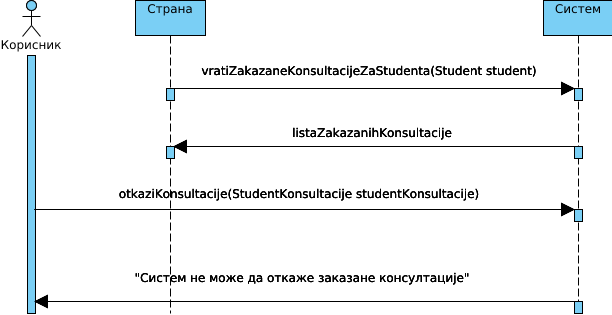
### Основни сценарио СК

1. Студент **позива** систем да обрише заказане консултације. (АПСО)
2. Систем **приказује** Студенту поруку: “Систем је отказао заказане консултације.” (ИА)



Алтернативна сценарија

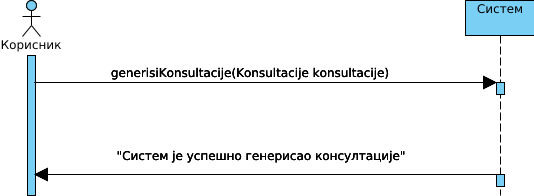
2.1 Уколико систем не може да обрише заказане консултације он приказује Студенту поруку “Систем не може да обрише заказане консултације”. (ИА)



## СК6: Случај коришћења – Генерисање термина консултација

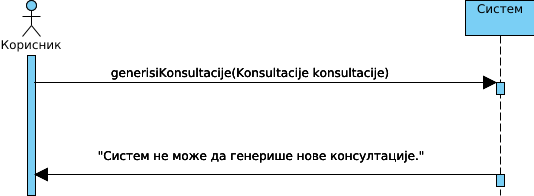
### Основни сценарио СК

1. Наставник **позива** систем да запамти податке о новом термину консултација. (АПСО)
2. Систем **приказује** Наставнику запамћени термин консултација и поруку: “Систем је запамтио консултације“. (ИА)



Алтернативна сценариа

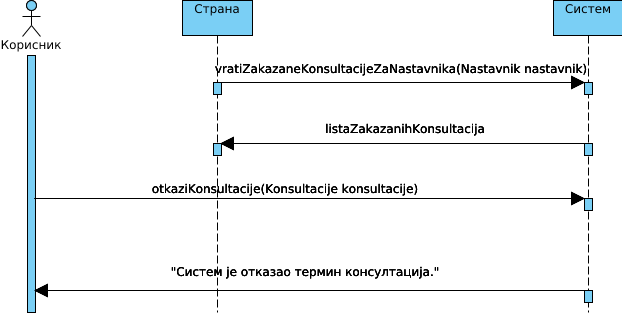
2.1 Уколико систем не може да запамти нови термин консултација он приказује Наставнику поруку "Систем не може да генерише нове консултације." (ИА)



## СК7: Случај коришћења – Отказивање целог термина консултација

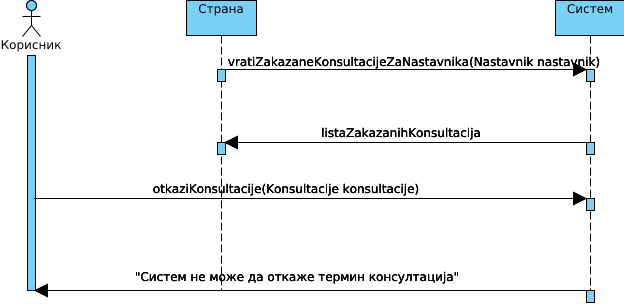
### Основни сценарио СК

1. Наставник **позива** систем да откаже термин консултација(АПСО)
2. Систем **приказује** Наставнику поруку: “Систем је отказао термин консултација (ИА)



Алтернативна сценарија

2.1 Уколико систем не може да откаже термин консултација систем приказује Наставнику поруку “Систем не може да обрише термин консултација”. (ИА)



Kao rezultat analize scenarija dobijeno je ukupno 16 sistemskih operacija koje treba projektovati:

### 7.2.2. Понашање софтверског система - Дефинисање уговора о системским операцијама

У овом студијском примеру слој који има приступ подацима базе података је креиран преко ДАО слоја. Овај слој користи Хибернате АПИ како би вршио интеракцију са базом. ДАО слој се позива од стране сервисног слоја који обезбеђује апликацији трансакционалне операције саме пословне логике. Идеја која се крије иза сервиса је извршавање сервисних метода које представљају мале, атомске операције које апликација извршава над базом, тако да сервисна метода или успе да изврши поменуте тако да резултира конзистентним статусом базе или (уколико је дошло до грешке приликом извршавања) враћа стање базе на њено. Захтеви који стижу од клијента се прихватају од стране контролера који даље позива одговарајуће методе сервиса које извршавају жељену операцију.

1. Уговор УГ1: registrujSe

**Операција**: registrujSe (KorisnickiNalog korisnickiNalog);

**Веза са СК**: СК1

**Предуслови**: Просто вредносно ограничење над објектом Кориснички налог мора бити

задовољено. Структурно ограничење над објектом Кориснички налог мора бити задовољено.

**Постуслови**: Креиран је нови кориснички налог.

2. Уговор УГ1: ulogujSe

**Операција**: ulogujSe(KorisnickiNalog korisnickiNalog);

**Веза са СК**: СК2

**Предуслови**: Просто вредносно ограничење над објектом Кориснички налог мора бити

задовољено. Структурно ограничење над објектом Кориснички налог мора бити задовољено.

**Постуслови**: Креиран је нови кориснички налог.

3. Уговор УГ1: vratiKonsultacijeZaNastavnika

**Операција**: vratiKonsultacijeZaNastavnika (String jmbgNastavnika);

**Веза са СК**: СК4

**Предуслови**: /

**Постуслови**: /

4. Уговор УГ1: zakaziKonsultacije

**Операција**: zakaziKonsultacije (StudentKonsultacije studentKonsultacije);

**Веза са СК**: СК4

**Предуслови**: Просто вредносно ограничење над објектом *StudentKonsultacije* мора бити

задовољено. Структурно ограничење над објектом *StudentKonsultacije* мора бити задовољено.

**Постуслови**: Креиран је нови кориснички налог.

5. Уговор УГ1: vratiZakazaneKonsultacijeZaStudenta

**Операција**: vratiZakazaneKonsultacijeZaStudenta (String brojIndeksaStudenta);

**Веза са СК**: СК5

**Предуслови**: /

**Постуслови**: /

6. Уговор УГ1: otkaziZakazaneKonsultacije

**Операција**: otkaziZakazaneKonsultacije(StudentKonsultacije studentKonsultacije);

**Веза са СК**: СК5

**Предуслови**: Просто вредносно ограничење над објектом *StudentKonsultacije* мора бити

задовољено. Структурно ограничење над објектом *StudentKonsultacije* мора бити задовољено.

**Постуслови**: Zakazane konsultacije (*StudentKonsultacije*) su obrisane.

7. Уговор УГ1: generisiNoveKonsultacije

**Операција**: generisiNoveKonsultacije (Konsultacije konsultacije);

**Веза са СК**: СК6

**Предуслови**: Просто вредносно ограничење над објектом Konsultacije мора бити

задовољено. Структурно ограничење над објектом Konsultacije мора бити задовољено.

**Постуслови**: Креиранe су нове консултације(Konsultacije).

8. Уговор УГ1: vratiZakazaneKonsultacijeZaNastavnika

**Операција**: vratiZakazaneKonsultacijeZaNastavnika (StudentKonsultacije studentKonsultacije);

**Веза са СК**: СК7

**Предуслови**: /

**Постуслови**: /

9. Уговор УГ1: otkaziKonsultacije

**Операција**: otkaziKonsultacije (Konsultacije konsultacije);

**Веза са СК**: СК7

**Предуслови**: Просто вредносно ограничење над објектом Konsultacije мора бити

задовољено. Структурно ограничење над објектом Konsultacije мора бити задовољено.

**Постуслови**: Отказане су консултације.

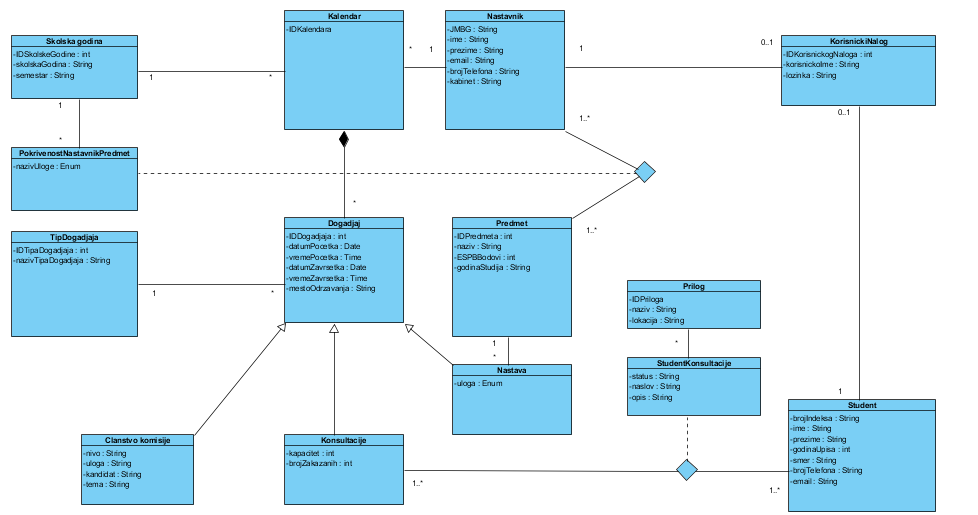
### 7.2.3. Структура софтверског система - Концептуални (доменски) модел

Концептуални модел описује концептуалне класе домена проблема. Он садржи

концептуалне класе, које се називају доменски објекти, и асоцијације између

концептуалних класа. Концептуалне класе се идентификују на основу атрибута, који

описују особине класа.



Као резултат анализе сценарија СК и прављења концептуалног модела добија се логичка структура и понашање софтверског система који су следећем дијаграму:

ovde fali

### 7.2.4. Структура софтверског система - Релациони модел

На основу концептуалног модела прави се релациони модел. Релациони модел служи као

основа за креирање релационе базе података. На основу концептуалног модела,

добијамо следећи релациони модел:

**DOGADJAJ** (ID\_KALENDARA, ID\_DOGADJAJA, DATUM\_I\_VREME\_POCETKA, DATUM\_I\_VREME\_ZAVRSETKA, MESTO\_ODRZAVANJA, NIVO, ULOGA, KANDIDAT, TEMA ,KAPACITET, BROJ\_ZAKAZANIH, ULOGA\_NASTAVNIKA, *ID\_PREDMETA, ID\_TIPA\_DOGADJAJA*)

**KALENDAR**(ID\_KALENDARA, *ID\_SKOLSKE\_GODINE, JMBG\_NASTAVNIKA*)

**KORISNICKI\_NALOG**(ID\_KOR\_NAL, KORISNICKO\_IME, LOZINKA, *JMBG\_NASTAVNIKA, BROJ\_INDEKSA\_STUDENTA*)

**NASTAVNIK**(JMBG, IME, PREZIME, EMAIL, BROJ\_TELEFOA, KABINET, SLIKA)

**POKRIVENOST\_NASTAVE**(*JMBG\_NASTAVNIKA, ID\_PREDMETA*, ULOGA, *ID\_SKOLSKE\_GODINE*)

**PREDMET**(ID\_PREDMETA, NAZIV, ESPB\_BODOVI, GODINA\_STUDIJA)

**PRILOG**(ID\_PRILOGA, NAZIV, LOKACIJA*, ID\_DOGADJAJA, ID\_KALENDARA, BROJ\_INDEKSA\_STUDENTA*)

**SKOLSKA\_GODINA**(ID\_SKOLSKE\_GODINE, SKOLSKA\_GODINA, SEMESTAR)

**STUDENT**(BROJ\_INDEKSA, IME, PREZIME, GODINA\_UPISA, SMER, BROJ\_TELEFONA, EMAIL)

**STUDENT**\_**KONSULTACIJE**(*ID\_KALENDARA, ID\_DOGADJAJA, BROJ\_INDEKSA\_STUDENTA*, STATUS, NASLOV, OPIS)

**TIP\_DOGADJAJA**(ID\_TIPA\_DOGADJAJA, NAZIV\_TIPA\_DOGADJAJA)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Табела | | DOGADJAJ | Просто вредносно ограничење | | Сложено вредносно ограничење | | Структурно ограничење |
| Атрибути | Ime | | Тип атрибута | Вредност атрибута | Међузависност атрибута једне табеле | Међузависно ст атрибута више табела | **Insert** /  **Update** Cascades Pacijent |
|  | |  |  |  |  |
|  | ID\_KALENDARA | | Long | not null | **Delete** Restricted Pacijent |
| ID\_DOGADJA | | Long | not null |  |  |
| DATUM\_I\_VREME\_POCETKA | |  |  |  |  |
| DATUM\_I\_VREME\_ZAVRSETKA | | Date |  |  |  |
| MESTO\_ODRZAVANJA | | String |  |  |  |
| NIVO | | String |  |  |  |
| ULOGA | | String |  |  |  |
| KANDIDAT | | String |  |  |  |
| TEMA | | String |  |  |  |
| KAPACITET | | Integer |  |  |  |
| BROJ  ZAKAZANIH | | Integer |  |  |  |
| ULOGA\_NASTAVNIKA | | String |  |  |  |
| ID\_PREDMETA | | Long |  |  |  |
| ID\_TIPA\_DOGADJAJA | | Long |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Табела | | KALENDAR | Просто вредносно ограничење | | Сложено вредносно ограничење | | Структурно ограничење |
| Атрибути | Ime | | Тип атрибута | Вредност атрибута | Међузависност атрибута једне табеле | Међузависно ст атрибута више табела | **Insert** /  **Update** Cascades Pacijent |
|  | |  |  |  |  |
|  | ID\_KALENDARA | | Long | not null | **Delete** Restricted Pacijent |
| ID\_SKOLSKE\_GODINE | | Long | not null |  |  |
| JMBG\_NASTAVNIKA | |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Табела | | KORISNICKI\_NALOG | Просто вредносно ограничење | | Сложено вредносно ограничење | | Структурно ограничење |
| Атрибути | Ime | | Тип атрибута | Вредност атрибута | Међузависност атрибута једне табеле | Међузависно ст атрибута више табела | **Insert** /  **Update** Cascades Pacijent |
|  | |  |  |  |  |
|  | ID\_KOR\_NAL | | Long | not null | **Delete** Restricted Pacijent |
| KORISNICKO\_IME | | Long | not null |  |  |
| *JMBG\_NASTAVNIKA* | |  |  |  |  |
| *BROJ\_INDEKSA\_STUDENTA* | |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Табела | | NASTAVNIK | Просто вредносно ограничење | | Сложено вредносно ограничење | | Структурно ограничење |
| Атрибути | JMBG | | Тип атрибута | Вредност атрибута | Међузависност атрибута једне табеле | Међузависно ст атрибута више табела | **Insert** /  **Update** Cascades Pacijent |
|  | |  |  |  |  |
|  | IME | | Long | not null | **Delete** Restricted Pacijent |
| PREZIME | | Long | not null |  |  |
| *EMAIL* | |  |  |  |  |
| *BROJ\_TELEFONA* | |  |  |  |  |
| KABINET | |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Табела | | POKRIVENOST\_NASTAVE | Просто вредносно ограничење | | Сложено вредносно ограничење | | Структурно ограничење |
| Атрибути | Ime | | Тип атрибута | Вредност атрибута | Међузависност атрибута једне табеле | Међузависно ст атрибута више табела | **Insert** /  **Update** Cascades Pacijent |
|  | |  |  |  |  |
|  | *JMBG\_NASTAVNIKA* | | Long | not null | **Delete** Restricted Pacijent |
| ID\_PREDMETA | | Long | not null |  |  |
| ULOGA | |  |  |  |  |
| *ID\_SKOLSKE\_GODINE* | |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Табела | | PREDMET | Просто вредносно ограничење | | Сложено вредносно ограничење | | Структурно ограничење |
| Атрибути | Ime | | Тип атрибута | Вредност атрибута | Међузависност атрибута једне табеле | Међузависно ст атрибута више табела | **Insert** /  **Update** Cascades Pacijent |
|  | |  |  |  |  |
|  | ID\_PREDMETA | | Long | not null | **Delete** Restricted Pacijent |
| NAZIV | | Long | not null |  |  |
| ESPB\_BODOVI | |  |  |  |  |
| GODINA\_STUDIJA | |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Табела | | PRILOG | Просто вредносно ограничење | | Сложено вредносно ограничење | | Структурно ограничење |
| Атрибути | Ime | | Тип атрибута | Вредност атрибута | Међузависност атрибута једне табеле | Међузависно ст атрибута више табела | **Insert** /  **Update** Cascades Pacijent |
|  | |  |  |  |  |
|  | ID\_PRILOGA | | Long | not null | **Delete** Restricted Pacijent |
| NAZIV | | Long | not null |  |  |
| LOKACIJA | |  |  |  |  |
| *ID\_DOGADJAJA* | |  |  |  |  |
| *ID\_KALENDARA* | |  |  |  |  |
| *BROJ\_INDEKSA\_STUDENTA* | |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Табела | | SKOSKA\_GODINA | Просто вредносно ограничење | | Сложено вредносно ограничење | | Структурно ограничење |
| Атрибути | Ime | | Тип атрибута | Вредност атрибута | Међузависност атрибута једне табеле | Међузависно ст атрибута више табела | **Insert** /  **Update** Cascades Pacijent |
| ID\_SKOLSKE\_GODINE | |  |  |  |  |
|  |  | | Long | not null | **Delete** Restricted Pacijent |
| SKOLSKA\_GODINA | | Long | not null |  |  |
| SEMESTAR | |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Табела | | STUDENT | Просто вредносно ограничење | | Сложено вредносно ограничење | | Структурно ограничење |
| Атрибути | Ime | | Тип атрибута | Вредност атрибута | Међузависност атрибута једне табеле | Међузависно ст атрибута више табела | **Insert** /  **Update** Cascades Pacijent |
|  | |  |  |  |  |
|  | BROJ\_INDEKSA | | Long | not null | **Delete** Restricted Pacijent |
| IME | | Long | not null |  |  |
| PREZIME | |  |  |  |  |
| GODINA\_UPISA | |  |  |  |  |
| SMER | |  |  |  |  |
| BROJ\_TELEFONA | |  |  |  |  |
| EMAIL | |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Табела | | STUDENT\_KONSULTACIJE | Просто вредносно ограничење | | Сложено вредносно ограничење | | Структурно ограничење |
| Атрибути | Ime | | Тип атрибута | Вредност атрибута | Међузависност атрибута једне табеле | Међузависно ст атрибута више табела | **Insert** /  **Update** Cascades Pacijent |
|  | |  |  |  |  |
|  | ID\_KALENDARA | | Long | not null | **Delete** Restricted Pacijent |
| ID\_DOGADJAJA | | Long | not null |  |  |
| *BROJ\_INDEKSA\_STUDENTA* | |  |  |  |  |
| *STATUS* | |  |  |  |  |
| *NASLOV* | |  |  |  |  |
| OPIS | |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Табела | | TIP\_DOGADJAJA | Просто вредносно ограничење | | Сложено вредносно ограничење | | Структурно ограничење |
| Атрибути | Ime | | Тип атрибута | Вредност атрибута | Међузависност атрибута једне табеле | Међузависно ст атрибута више табела | **Insert** /  **Update** Cascades Pacijent |
|  | |  |  |  |  |
|  | ID\_TIPA\_DOGADJAJA | | Long | not null | **Delete** Restricted Pacijent |
| NAZIV\_TIPA\_DOGADJAJA | | Long | not null |  |  |

7.3. Пројектовање

Фаза пројектовања описује физичку структуру и понашање софтверског система

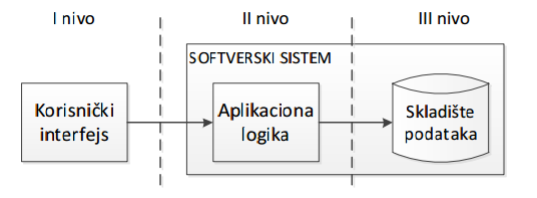
(архитектуру софтверског система).

Архитектура софтверског система је тронивојска и састоји се од следећих нивоа:

* Кориснички интерфејс
* Апликациона логика
* Складиште података

Ниво корисничког интерфејса је на страни клијента, а апликациона логика и складиште

података на страни сервера.

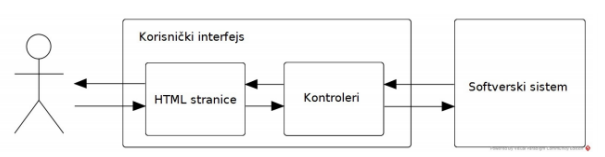


Slika 1 Tronivojska arhitektura aplikacije

7.3.1. Пројектовање корисничког интерфејса

Кориснички интерфејс представља реализацију улаза/излаза софтверског система и састоји се

од екранских форми.



Slika 2 Odnos korisničkog interfejsa i softverskog sistema

Пројектовање страница и екренских форми

Кориснички интерфејс дефенисан је преко скупа екранских форми. Сценарија коришћења

екранских форми су директно повезана са сценаријима случајева коришћења.

Екранске форме имплементиране су помоћу ХТМЛ-а и ЦСС-а. ХТМЛ (Хyпертеxт

Маркуп Лангуаге) је језик за означавање који се користи за презентовање садржаја

на вебу. Помоћу овог језика дефинишу се компонете и структура веб странице, као и

контроле са којима корисник врши итеракцију. Док ХТМЛ служи са дефинисање структуре и

садржаја странице, ЦСС (Цасцадинг Стyле Схеетс) је језик који се користи за дефинисање

њеног изгледа (позиција компоненти, распоред, величина, боја).

Изглед почетне странице:

………itd itd

7.3.2. Пројектовање апликационе логике

Апликациони сервери су одговорни да обезбеде сервисе који ће да омогуће реализацију

апликационе логике софтверског система. Пројектовани апликациони сервер садржи:

* део за комуникацију са клијентима,
* контролер апликационе логике,
* део за комуникацију са складиштем података (Брокер базе података),
* део који садржи пословну логику.

Слика 48. Слојеви софтверског система ovo mi fali

Комуникација са клијентима

Клијент шаље захтев за извршење неке од операција, сервер прихвата и прослеђује га до

контролера апликационе логике. Након извршења операције, контролер враћа резултат

клијенту у виду ЈСП стране и одредјеним атрибутима у зависности од захтева.

Комуникација између клијента и сервера се обавља разменом порука преко објекта

класе МоделАндВиеw.

Комуникација корисничког интерфејса и софтверског система имплементирана је у

контролерима корисничког интерфејса (помоћу *OracleJET JavaScript* фрејмворка) који

прихвата податке и догађаје са ХТМЛ страница и прослеђује их софтерског систему преко

РЕСТ АПИ-ја. Такође, контролери КИ прихватају податке и резултате обраде софтверског

система и прослеђују их ХТМЛ страницама у облику погодном за презентовање

кориснику. У наставку приказан је део контролера корисничког интерфејса задуженог за

100

постављање питања.

Контролер апликационе логике

Контролери апликационе логике имплементирају веб интерфејс за комуникацију софтверског

система са клијентима. Након пријема захтева од стране клијента, контролери апликационе

логике позивају одговарајуће сервисе пословне логике који имплементирају пословна

правила.

Контролер апликационе логике прихвата захтеве за извршење операција и исте прослеђује

до модела те класе. Након извршења операције, контролер прихвата одговор и враћа

назад позиваоцу. Како је у фазама прикупљања захтева и анализе дата спецификација

структуре и понашања софтверског система, односно спецификација пословне логике

софтверског система, следећа слика даје опис система након фазе пројектовања

комуникације са клијентима и контролера апликационе логике. Контролери су означени

101

анотацијама @РестЦонтроллер, као и @РеqуестМаппинг преко чега се одређени

контролор мапира и затим позивају одређене операције за извршавање. На основу

захтева контролер одређује која ће операција да се изврши, након чега као резултат

враћа ЈСОН.

neki kod fali

### 7.3.3. Пројектовање системских операција

За архитектуру апликације коришћен је ДАО патерн у повезаности са сервисима. Дата аццесс објецт патерн је корисћен само да обезбеди везу са базом. Направљен је генерички, тако да се мозе користити за разлиците објекте: Системске операције дефинишу понашање софтверског система. Пројектоване су помоћу метода сервисних интерфејса. Свака метода представља једну атомску операцију пословне логике која комуницира са слојем за приступ подацима. Све операције треба да буду међусобно независне и да очувају интегритет и конзистентност базе података по свом завршетку, без обзира да ли их је систем успешно или неуспешно извршио. У овом студијском примеру дефинисани су следећи сервисни интерфејси: 102 Пример Корисничког сервисног интерфејса из студијског примера:

KORISNICKI NALOG SERVICE

За сваки од уговора системских операција дефинисаних у фази анализе пројектује се концептуално решење.

1. Уговор УГ1: registrujSe

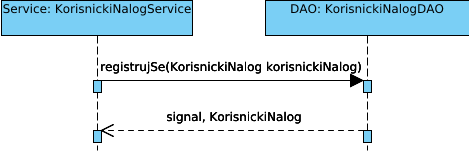
**Операција**: registrujSe (KorisnickiNalog korisnickiNalog): signal;

**Веза са СК**: СК1

**Предуслови**: Просто вредносно ограничење над објектом Кориснички налог мора бити

задовољено. Структурно ограничење над објектом Кориснички налог мора бити задовољено.

**Постуслови**: Креиран је нови кориснички налог.



2. Уговор УГ1: ulogujSe

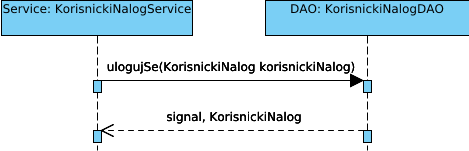
**Операција**: ulogujSe(KorisnickiNalog korisnickiNalog);

**Веза са СК**: СК2

**Предуслови**: Просто вредносно ограничење над објектом Кориснички налог мора бити

задовољено. Структурно ограничење над објектом Кориснички налог мора бити задовољено.

**Постуслови**: Креиран је нови кориснички налог.



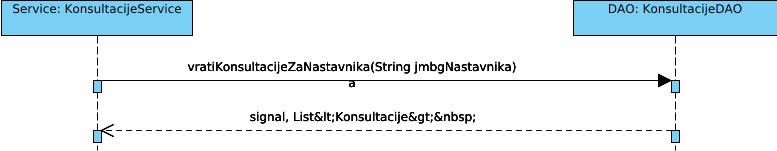
3. Уговор УГ3: vratiKonsultacijeZaNastavnika

**Операција**: vratiKonsultacijeZaNastavnika (String jmbgNastavnika);

**Веза са СК**: СК4

**Предуслови**: /

**Постуслови**: /



4. Уговор УГ4: zakaziKonsultacije

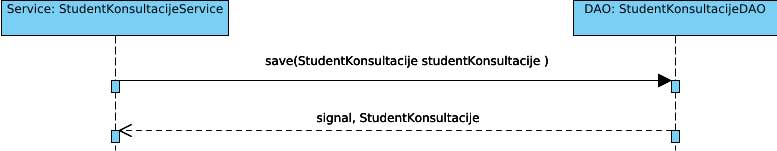
**Операција**: zakaziKonsultacije (StudentKonsultacije studentKonsultacije);

**Веза са СК**: СК4

**Предуслови**: Просто вредносно ограничење над објектом *StudentKonsultacije* мора бити

задовољено. Структурно ограничење над објектом *StudentKonsultacije* мора бити задовољено.

**Постуслови**: Креиран је нови кориснички налог.



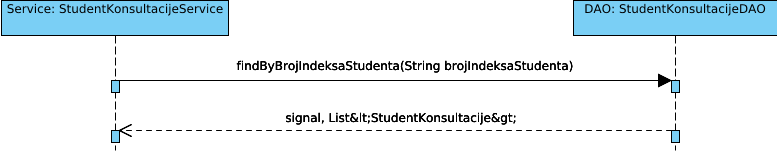
5. Уговор УГ5: vratiZakazaneKonsultacijeZaStudenta

**Операција**: vratiZakazaneKonsultacijeZaStudenta (String brojIndeksaStudenta);

**Веза са СК**: СК5

**Предуслови**: /

**Постуслови**: /



6. Уговор УГ6: otkaziZakazaneKonsultacije

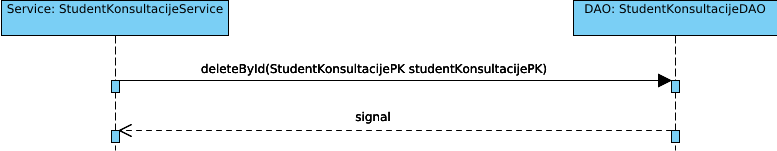
**Операција**: otkaziZakazaneKonsultacije(StudentKonsultacije studentKonsultacije);

**Веза са СК**: СК5

**Предуслови**: Просто вредносно ограничење над објектом *StudentKonsultacije* мора бити

задовољено. Структурно ограничење над објектом *StudentKonsultacije* мора бити задовољено.

**Постуслови**: Zakazane konsultacije (*StudentKonsultacije*) su obrisane.



7. Уговор УГ7: generisiNoveKonsultacije

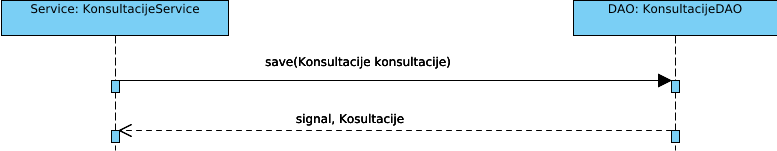
**Операција**: generisiNoveKonsultacije (Konsultacije konsultacije);

**Веза са СК**: СК6

**Предуслови**: Просто вредносно ограничење над објектом Konsultacije мора бити

задовољено. Структурно ограничење над објектом Konsultacije мора бити задовољено.

**Постуслови**: Креиранe су нове консултације(Konsultacije).



8. Уговор УГ8: otkaziKonsultacije

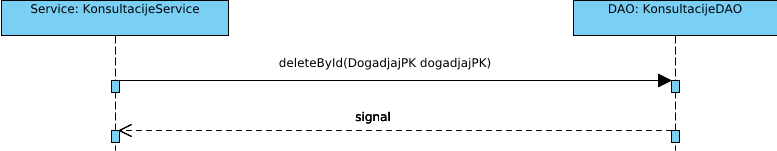
**Операција**: otkaziKonsultacije (Konsultacije konsultacije);

**Веза са СК**: СК7

**Предуслови**: Просто вредносно ограничење над објектом Konsultacije мора бити

задовољено. Структурно ограничење над објектом Konsultacije мора бити задовољено.

**Постуслови**: Отказане су консултације.



Slika 49. Klase odgovorne za SO implementaciju servisnih interfejsa ово ми фали