Универзитет у Београду

Факултет организационих наука

Лабораторија за софтверско инжењерство

Предмет : Пројектовање софтвера

Дипломски рад

Тема : Развој софтверског система за праћење рада физио центра употребом .NET технологија

Ментор : Студент :

Татјана Стојановић Ђорђе Петровић 0271/2018

Београд, 2024.

# Садржај

[Садржај 2](#_Toc165233358)

[1 Увод 5](#_Toc165233359)

[2 Ларманова метода развоја софтверског система 7](#_Toc165233360)

[2.1 Прикупљање захтева од корисника 8](#_Toc165233361)

[2.2 Анализа 10](#_Toc165233362)

[2.3 Пројектовање 11](#_Toc165233363)

[2.4 Имплементација 12](#_Toc165233364)

[2.5 Тестирање 12](#_Toc165233365)

[3 C# Програмски језик 13](#_Toc165233366)

[4 .NET технологије 15](#_Toc165233367)

[4.1 .NET платформа 15](#_Toc165233368)

[4.1.1. Windows форме 16](#_Toc165233369)

[4 2. Развој софтверског система 17](#_Toc165233370)

[1. 2.1 Прикупљање корисничких захтева 17](#_Toc165233371)

[2. Вербални опис 17](#_Toc165233372)

[3. Случајеви коришћења 18](#_Toc165233373)

[СК1: Случај коришћења – Пријава администратора на систем 19](#_Toc165233374)

[СК2 : Случај коришћења – Унос новог клијента 20](#_Toc165233375)

[СК3 : Случај коришћења – Брисање клијента 22](#_Toc165233376)

[СК4 : Случај коришћења – Измена података о клијенту 23](#_Toc165233377)

[СК5 : Случај коришћења – Унос нове услуге 25](#_Toc165233378)

[СК6 : Случај коришћења – Брисање услуге 26](#_Toc165233379)

[СК7 : Случај коришћења – Унос нових термина 28](#_Toc165233380)

[СК8 : Случај коришћења – Брисање заказаног термина 30](#_Toc165233381)

[СК9 : Случај коришћења – Претрага заказаних термина 32](#_Toc165233382)

[5 Анализа софтверског система 33](#_Toc165233383)

[4. Понашање софтверског система – Системски дијаграм секвенци 33](#_Toc165233384)

[ДС1: Дијаграм секвенци случаја коришћења – Пријава администратора на систем 33](#_Toc165233385)

[ДС2: Дијаграм секвенци случаја коришћења – Унос новог клијента 36](#_Toc165233386)

[ДС3: Дијаграм секвенци случаја коришћења – Брисање клијента 38](#_Toc165233387)

[ДС4: Дијаграм секвенци случаја коришћења – Измена података о клијенту 44](#_Toc165233388)

[ДС5: Дијаграм секвенци случаја коришћења – Унос нове услуге 51](#_Toc165233389)

[ДС6: Дијаграм секвенци случаја коришћења – Брисање услуге 53](#_Toc165233390)

[ДС7: Дијаграм секвенци случаја коришћења – Унос нових термина 59](#_Toc165233391)

[ДС8: Дијаграм секвенци случаја коришћења – Брисање заказаног термина 61](#_Toc165233392)

[ДС9: Дијаграм секвенци случаја коришћења – Претрага заказаних термина 66](#_Toc165233393)

[5. Понашање софтверског система – Дефинисање уговора 70](#_Toc165233394)

[Уговор УГ1: Пријави администратора 70](#_Toc165233395)

[Уговор УГ2: Врати листу типова клијената 70](#_Toc165233396)

[Уговор УГ3: Врати листу клијената 70](#_Toc165233397)

[Уговор УГ4: Учитај клијента 70](#_Toc165233398)

[Уговор УГ5: Претражи клијенте 71](#_Toc165233399)

[Уговор УГ6: Запамти клијента 71](#_Toc165233400)

[Уговор УГ7: Измени клијента 71](#_Toc165233401)

[Уговор УГ8: Обриши клијента 71](#_Toc165233402)

[Уговор УГ9: Врати листу врста услуга 72](#_Toc165233403)

[Уговор УГ10: Врати листу услуга 72](#_Toc165233404)

[Уговор УГ11: Врати листу типова напомена за услугу 72](#_Toc165233405)

[Уговор УГ12: Учитај услугу 72](#_Toc165233406)

[Уговор УГ13: Претражи услугу 72](#_Toc165233407)

[Уговор УГ14: Запамти услугу 73](#_Toc165233408)

[Уговор УГ15: Обриши услугу 73](#_Toc165233409)

[Уговор УГ16: Врати листу термина 73](#_Toc165233410)

[Уговор УГ17: Претражи термине 73](#_Toc165233411)

[Уговор УГ18: Учитај термин 74](#_Toc165233412)

[Уговор УГ19: Запамти термине 74](#_Toc165233413)

[Уговор УГ20: Обриши термин 74](#_Toc165233414)

[6. Структура софтверског система – концептуални модел 75](#_Toc165233415)

[6 Пројектовање софтверског система 77](#_Toc165233416)

[8. Структура софтверског система – релациони модел 78](#_Toc165233417)

[9. Пројектовање корисничког интерфејса 82](#_Toc165233418)

[СК1: Случај коришћења – Пријава администратора на систем 84](#_Toc165233419)

[СК2: Случај коришћења – Унос новог клијента 87](#_Toc165233420)

[СК3: Случај коришћења – Брисање клијента 89](#_Toc165233421)

[СК4: Случај коришћења – Измена података о клијенту 92](#_Toc165233422)

[СК5: Случај коришћења – Унос нове услуге 94](#_Toc165233423)

[СК6: Случај коришћења – Брисање услуге 97](#_Toc165233424)

[СК7: Случај коришћења – Унос заказаних термина 99](#_Toc165233425)

[СК8: Случај коришћења – Брисање заказаног термина 102](#_Toc165233426)

[СК9: Случај коришћења – Претрага заказаних термина 104](#_Toc165233427)

[10. Пројектовање апликационе логике 107](#_Toc165233428)

[Комуникација са клијентима 107](#_Toc165233429)

[Контролер апликационе логике 108](#_Toc165233430)

[Пословна логика 112](#_Toc165233431)

[Пројектовање понашања софтверског система (системске операције) 112](#_Toc165233432)

[Уговор УГ1: Пријави администратора 113](#_Toc165233433)

[Уговор УГ2: Врати листу типова клијената 114](#_Toc165233434)

[Уговор УГ3: Врати листу клијената 114](#_Toc165233435)

[Уговор УГ4: Учитај клијента 115](#_Toc165233436)

[Уговор УГ5: Претражи клијенте 116](#_Toc165233437)

[Уговор УГ6: Запамти клијента 116](#_Toc165233438)

[Уговор УГ7: Измени клијента 117](#_Toc165233439)

[Уговор УГ8: Обриши клијента 117](#_Toc165233440)

[Уговор УГ9: Врати листу врста услуга 118](#_Toc165233441)

[Уговор УГ10: Врати листу услуга 119](#_Toc165233442)

[Уговор УГ11: Врати листу типова напомена за услугу 119](#_Toc165233443)

[Уговор УГ12: Учитај услугу 120](#_Toc165233444)

[Уговор УГ13: Претражи услуге 120](#_Toc165233445)

[Уговор УГ14: Запамти услугу 121](#_Toc165233446)

[Уговор УГ15: Обриши услугу 122](#_Toc165233447)

[Уговор УГ16: Врати листу термина 122](#_Toc165233448)

[Уговор УГ17: Претражи термине 123](#_Toc165233449)

[Уговор УГ18: Учитај термин 123](#_Toc165233450)

[Уговор УГ19: Запамти термине 124](#_Toc165233451)

[Уговор УГ20: Обриши термин 124](#_Toc165233452)

[Пројектовање структуре софтверског система 126](#_Toc165233453)

[Комуникација са складиштем података 128](#_Toc165233454)

[11. Пројектовање складишта података 134](#_Toc165233455)

[7 Имплементација софтверског система 137](#_Toc165233456)

[6 Тестирање софтверског система 138](#_Toc165233457)

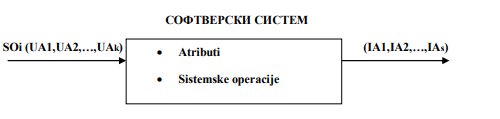
[7 Литература 139](#_Toc165233458)

# Увод

Систем, у суштини, представља комплексни скуп међусобно повезаних компоненти које користе различите ресурсе како би заједничким деловањем постигле одређене циљеве. Они су кључни за разумевање и решавање различитих проблема, као и за анализу, пројектовање и управљање сложеним ентитетима. Сваки систем има своје улазе - ресурсе које користи, и излазе - резултате свог рада.

Примењујући системски приступ, софтверски систем се може дефинисати као скуп атрибута и системских операција. Атрибути служе за представљање структуре софтверског система, која се може описати концептуалним и релационим моделима, док системске операције описују понашање система кроз дијаграме секвенци. Атрибути су у основи концепти реалног система који се користе за описивање статичких карактеристика система, док су системске операције основне функције система које се могу позивати из окружења система.

Улаз у софтверски систем је представљен системском операцијом и скупом улазних аргумената, док је излаз из система резултат извршења одређених системских операција над атрибутима.



*Слика 1. Софтверски систем (преузето са 5)*

Квалитетан софтверски систем се карактерише поузданошću, ефикасношću, лаком одрживошću и прилагодљивошću потребама корисника. За постизање ових стандарда, развој софтверског система се заснива на различитим методологијама, техникама и алатима, укључујући објектно-оријентисано програмирање, агилне приступе, test-driven development и друге.

Развој софтверског система је објашњен помоћу поједностављене Ларманове методе развоја софтвера, која омогућава итеративни животни циклус софтвера са циљем смањења ризика од неуспеха. У наставку, у другом поглављу, биће дат опширнији приказ ове методологије.

У трећем поглављу је представљен и описан C# као програмски језик, укључујући основне концепте који га карактеришу. C# је програмски језик који се базира на објектно-оријентисаној парадигми, што значи да се користе класе и објекти за организацију и рад са подацима. Једна од карактеристика C# језика је и могућност вишенитног програмирања, што омогућује извршавање више задатака истовремено у програму. Концепт сокета омогућава двосмерну комуникацију између два програма на мрежи, преко повезивања серверског и клијентског програма. За графички приказ апликације, користи се графички кориснички интерфејс.

У четвртом поглављу пружен је кратак преглед и опис .NET и .NET frameworka.

Пето поглавље је посвећено конкретном примеру - развоју софтверског система за праћење рада физио центра. У овом примеру користе се .NET технологије, примењујући претходно објашњену упрошћену Ларманову методу развоја софтвера и C# као програмски језик.

У шестом поглављу налази се закључак.

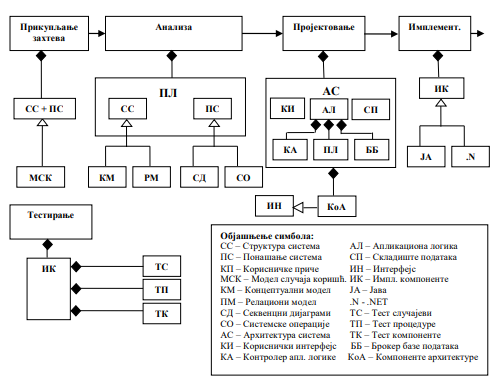
Седмо поглавље садржи саму литературу коришћену за израду овог рада.

# Ларманова метода развоја софтверског система

Развој, односно животни циклус, софтверског система јесте процес креирања софтверског производа зарад решавања разноврсних корисничких потреба и проблема. Развој се састоји од следећих фаза:

* прикупљање корисничких захтева,
* анализа,
* пројектовање,
* имплементација,
* тестирање.

У фази прикупљања захтева, детаљно се анализирају и дефинишу сва потребна својства и услови које софтверски систем треба да испуни. У фази анализе, изучава се логичка структура и понашање система, са фокусом на пословној логици коју систем треба да обезбеди. Затим, у фази пројектовања, утврђује се архитектура система, укључујући кориснички интерфејс, апликациону логику и складиште података, с обзиром на захтеве и анализу. У фази имплементације, креирају се и развијају компоненте у оквиру постојећих технологија као што су .NET или Java. Коначно, у фази тестирања, свака компонента се детаљно испитује користећи различите методе тестирања, укључујући тест случајеве, процедуре и компоненте, с циљем осигуравања исправног рада целокупног система. **5**



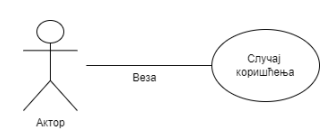
*Слика 2. Развој софтверског система (преузето са 5)*

## 2.1 прикупљање захтева од корисника

Захтеви представљају својства и услове које систем или пројекат мора да испуни. Често се помиње подела на функционалне и нефункционалне захтеве. Функционални захтеви одређују функције које систем треба да обавља, док нефункционални захтеви укључују све остале захтеве, као што су употребљивост, поузданост, перформансе и подрживост система. У смислу квалитета софтверског система, нефункционални захтеви представљају атрибуте квалитета софтверског система. **5**Према Ларману, уз помоћ UML Модела случаја коришћења (Use-Case Model) описују је сви захтеви. **5**

Модел случаја коришћења чине:

* + Скуп случајева коришћења (СК) – описи свих интеракција између корисника и система
  + Актори случајева коришћења (АК) – спољни корисник, односно учецник у систему
  + Веза између актора и случајева коришћења



*Слика 3. Модел случаја коришћења*

Сваки случај коришћења може бити повезан са више актора система, док један актор система може бити повезан са више случајева коришћења. Однос између актора и случаја коришћења представља везу, при чему свака веза повезује пар актора и случаја коришћења. Случај коришћења садржи скуп сценарија појављивања, где се сваки сценарио односи на једно очекивано коришћење система од стране актора. Ови сценарији могу укључивати основни сценарио као и један или више алтернативних сценарија. Сценарији описују секвенце акција и интеракције између актора и система, при чему актор позива једну или више операција система. **5**

У зависности од жељених функција система, основне операције СК се извршавају следећим редом: **5**

* актор припрема улазне параметре за системску операцију (АПУСО),
* актор позива систем да изврши системску операцију (АПСО),
* актор извршава несистемску операцију (АНСО),
* систем извршава системску операцију (СО),
* резултат извршавања системске операције се прослеђује до актора (ИА).

У самом почетку развоја софтверског система случајеви коришћења (СК) се представљају текстуално, да би се касније приказали коришћењем секвенцијалних дијаграма.

Текстуални опис СК се састоји из: **5**

* назива СК,
* актора СК,
* учесника СК,
* предуслова нужних за почетак извршавања СК,
* основног сценарија,
* постуслова нужних за потврду успешнох извршења СК,
* алтернативних сценарија,
* специјалних захтева,
* технолошких захтева,
* отворених питања.

## 2.2 Анализа

Фаза анализе подразумева детаљан опис логичке структуре и функционисања софтверског система, односно пословне логике која га покреће. За опис функционисања софтверског система користе се дијаграми секвенци и системски оперативни токови, док се за опис структуре користе концептуални и релациони модел.

УМЛ дијаграми секвенци приказују, за одређени сценарио коришћења, догађаје који успостављају интеракцију између актора и система, али у одређеном редоследу. На њима се приказују само акције АПСО и ИА. **5**

Diagram, box and whisker chart

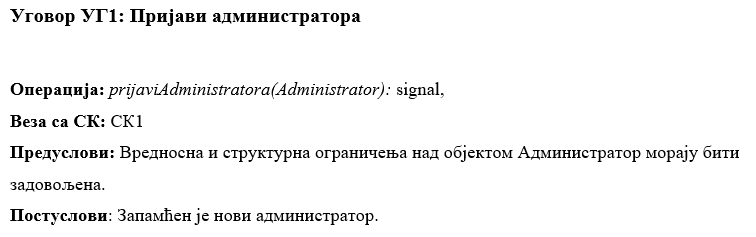
Description automatically generated

*Слика 4. Дијаграм секвенци*

На основу дијаграма секвенци за сваку СО креирају се одговарајући уговори који дају објашњење шта СО треба да уради искључујучи начин на који то чини.

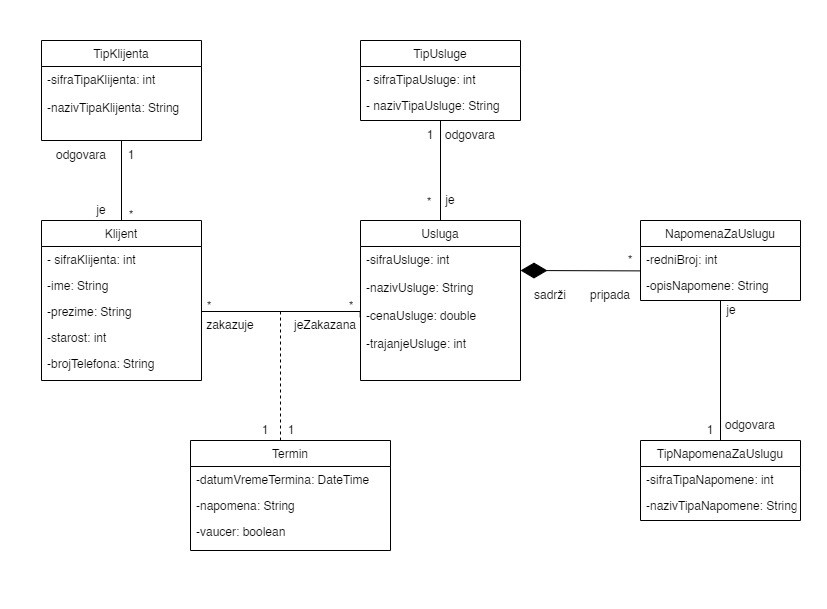
Структуру уговора чине следеће секције: **5**

* Операција – име, улазни аргумент и излазни аргумент
* Веза са СК – имена СК у којима позива системска операција
* Предуслов – одређени (пред)услови који је неопходно да буду испуњени пре извршења системске операције
* Постуслови – одређени (пост)услови који је неопходно да буду испуњени после извршења системске операције



*Слика 5. Уговор*

У концептуалном моделу се налазе концептуалне класе и везе између њих. Концептуалне класе су скупови података који описују структуру система, односно карактеристике сваке класе. Ове класе садрже атрибуте који представљају особине класе, односно информације које описују њихову структуру или понашање.



*Слика 6. Концептуални модел*

## 2.3 Пројектовање

На основу концептуалног модела, изводи се затим релациони модел чија је сврха да представља основу за креирање релационе базе података.

ТipKlijenta(sifraTipKlijenta, nazivTipaKlijenta)

Klijent(sifraKlijenta, ime, prezime, starost, brojTelefona, *sifraTipaKlijenta*)

TipUsluge(sifraTipaUsluge, nazivTipaUsluge)

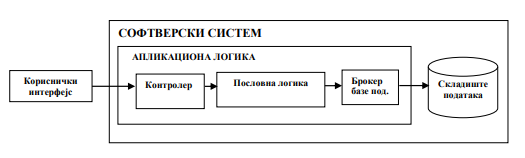
Usluga(sifraUsluge, nazivUsluge, cenaUsluge, trajanjeUsluge, *sifraTipaUsluge*)

TipNapomene(sifraTipaNapomene, nazivNapomene)

Napomena(*SifraUsluge*, redniBroj, opis, *SifraTipaNapomene*)

Termin(*sifraKlijenta*, *sifraUsluge*, datumTermina, vaucer, napomena)

У фази пројектовања описује се како ће изгледати и како ће се понашати софтверски систем. Архитектура софтверског система има три основна дела: кориснички интерфејс, апликациона логика и база података. Пројектовање корисничког интерфејса обухвата дизајнирање екраних форми и контролера. Пројектовање апликационе логике се односи на развој контролера за обраду података и логику која стоји иза њих. Када је у питању пословна логика, пројектује се логичка структура и начин на који ће систем поступати. **5**



*Слика 7. Тронивојска архитектура (преузето са 5)*

## Имплементација

Фаза имплементације представља конкретни процес развоја софтверског система, у којем се кодирају различите функционалности користећи одређени програмски језик. У овој фази, циљ је да се креира функционалан софтверски систем који испуњава све захтеве и очекивања корисника. Имплементација је кључна за превођење дизајна у реалан код, који ће омогућити функционисање система. Ова фаза је утемељена на претходној фази пројектовања, јер се током имплементације реализују дизајниране функционалности. Након имплементације, следи фаза тестирања, која ће проверити исправност и рад функционалности које су имплементиране.

## ТЕСТИРАЊе

У процесу развоја софтвера последња фаза јесте тестирање. Оно подразумева тестирање сваке од софтверских компоненти добијениџ у фази имплементације.

# 3 C# Програмски језик

C# је програмски језик који је дизајниран и први пут је представљен јавности 2000. године као део .NET платформе. У својој синтакси, он је сличан језицима Java и C++, али не постоји директан начин за миграцију програма написаних у тим језицима у C#.

C# је модеран, објектно-оријентисани програмски језик који је развијен од стране Microsoft-a као део стратегије за развој апликација на Windows платформи и у веб окружењу. Он је део .NET екосистема и представља један од кључних језика за развој апликација на .NET платформи.

Језик C# се дистрибуира заједно са специјалним окружењем на коме се извршава, које се зове Common Language Runtime (CLR). Захваљујући оквиру CLR, програми су преносиви и, једном када су написани, могу да функционишу са малим или нултим променама на различитим хардверским платформама и оперативним системима. Програми на C# најчешће се извршавају на MS Windows али свакако могу бити извршавани под Linux-ом, FreeBSD-ом, iOS-ом, Android-ом и другим оперативним системима преко бесплатне имплементације .NET Framework-а, Mono-а, која, међутим, није званично подржана од стране Microsoft-a. **2**

Неке од најзначајнијих карактеристика и аспеката C# програмског језика укључују: **3**

* **Објектно-оријентисана парадигма**: Код је организован у облику објеката који поседују својства и методе. Подржава концепте наслеђивања, полиморфизма, енкапсулације и апстракције.

• **Вишејезичност**: Поред објектно-оријентисане парадигме, подржава и процедуралну и функционалну парадигму.

* **Вишенитност**: C# омогућава развој интерактивних мрежних програма преко вишенитног програмирања, што омогућава извршавање више задатака истовремено.

• **Безбедност**: Има уграђене механизме безбедности, укључујући тип безбедности и контролу приступа коду.

• **Једноставна синтакса**: Има јасну и читљиву синтаксу, што је погодно за програмере на свим нивоима искуства.

• **Интероперабилност**: Омогућава сарадњу са другим програмским језицима као што је C++.

• **Строга типизација**: Типови променљивих морају бити декларисани и проверени у току компајлирања, што помаже у откривању грешака у раној фази развоја.

* **Интеграција са .NET екосистемом**: Интегрише се са .NET библиотеком класа (FCL), што омогућава коришћење готових компоненти за развој апликација.

• **Портабилност**: C# апликације се могу изводити на различитим платформама због подршке за cross-platform развој.

* **Богат екосистем алата**: Програмерима је на располагању широк спектар развојних алата, укључујући Visual Studio и Visual Studio Code.
* **Мобилни и веб развој**: Користи се за развој мобилних апликација преко Xamarin платформе и за веб развој преко ASP.NET технологија.

# 4 .NET технологије

.NET је бесплатан развојни оквир који подржава различите платформе и отвореног је кода и намењен изградњи различитих апликација. Омогућава извршавање програма на више језика, с тим да је C# најчешће коришћен. **1**

## 4.1 .NET платформа

.NET платформа је дизајнирана са циљем да омогући високу продуктивност, перформансе, сигурност и поузданост. Меморијом управља аутоматски помоћу „Garbage collector-a“, што је чини сигурном по питању типова и меморије захваљујући строгој контроли језика. Платформа укључује велики број библиотека које нуде широк спектар функционалности оптимизоване за високе перформансе на различитим оперативним системима и архитектурама чипова. **1**

.NET окружење се састоји од следеће компоненте: **1**

* .NET језици: C#, Visual Basic, F#, JScript .NET (серверски део JScript-а ), J# (Java клон) и C++;
* CLR (Common Language Runtime) – извршно оркужење заједничко за све језике. CLR функционише као виртуелна машина која омогућава извршаванје програма написаних у различитим програмским језицима, а који се компајлирају у CIL (Common Intermediate Language), такође познат као MSIL (Microsoft Intermediate Language). Поред тога, CLR пружа значајне услуге као што су безбедност, управљање меморијом и руковање изузецима
* .NET Framework библиотека класа – обимна колекција класа, типова и функција која је саставни део .NET-а. Она пружа готове компоненте и функционалности које програмери користе за бржи и једноставнији развој апликација. Функционалности су организоване у различите технолошке скупове, као што су ADO.NET (за рад са базама података) и Windows Forms (за креирање корисничког интерфејса).
* ASP.NET - делови .NET платформе који су специјализовани за развој веб апликација и сервиса. Ово укључује ASP.NET Web platforms за традиционалне веб апликације, ASP.NET MVC (Model-View-Controller) за веб апликације засноване на MVC архитектури и ASP.NET Core за cross-platform веб развој.
* Visual Studio - интегрисано развојно окружење које се карактерише интеграцијом са .NET платформом

### 4.1.1. Windows форме

Windows Forms је колекција управљаних библиотека у .NET Framework-у, намењених за развој комплексних клијентских апликација. Омогућава приказ података и управљање корисничким интеракцијама са поједностављеним инсталационим процесом и бољом безбедношћу у клијентским апликацијама.

Windows Forms пружа обимну клијентску библиотеку која омогућава приступ уграђеним елементима графичког корисничког интерфејса Windows и графицима из управљаног кода. Изграђен је са архитектуром заснованом на догађајима, што омогућава апликацијама да реагују на унос корисника за њихову извршну функцију. **4**

# 2. Развој софтверског система

## 2.1 Прикупљање корисничких захтева

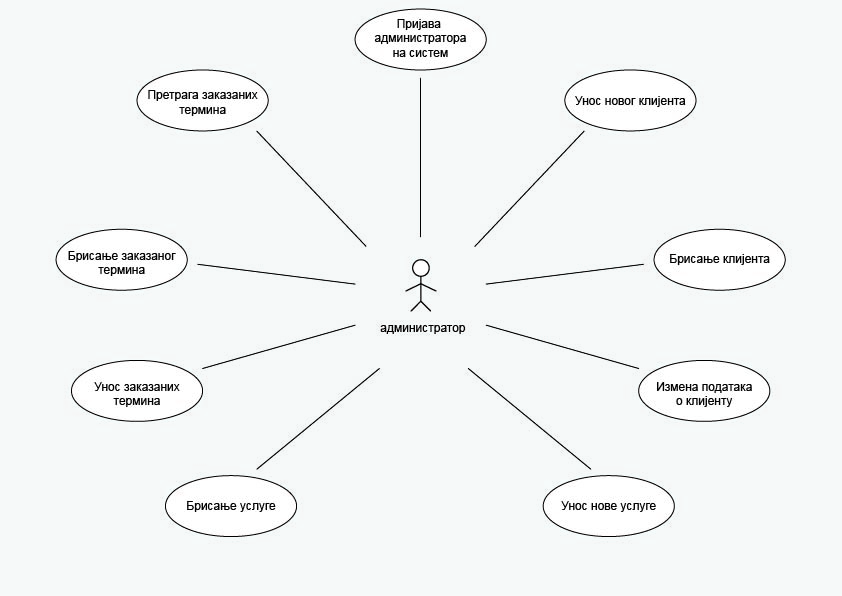
## Вербални опис

Неопходно је направити софтверски систем за праћење рада центра за физикалну рехабилитацију. Систем треба да води евиденцију о клијентима (пацијентима) самог центра и њиховим услугама, услугама које центар нуди и заказаним терминима. Свака услуга је јединствена јер се односи на једног клијента, и састоји је од једне или више услуга које сам администратор преписује. Поред тога, услуге је могуће изменити, тј. услуге је могуће обрисати након завршетка или напретка у лечењу клијента. Заказивање термина треба да садржи податке о клијенту, о услугама које се заказују и времену и датуму термина. Претрагу заказаних термина је могуће вршити према различитим критеријумима. Самим тим је могућа и измена термина, односно брисање постојећих термина.

## Случајеви коришћења

У овој апликацији идентификовано је 8 случајева коришћења :

1. Пријава администратора на систем
2. Унос новог клијента
3. Брисање клијента
4. Измена података о клијенту
5. Унос нове услуге (*Сложен случај коришћења*)
6. Брисање услуге
7. Унос нових термина (*Сложен случај коришћења*)
8. Брисање заказаних термина
9. Претрага заказаних терминa



*Слика 2. Dijagram sluчајева коришћења*

### СК1: Случај коришћења – Пријава администратора на систем

**Назив СК**

Пријава администратора на систем

**Актори СК**

Администратор

**Учесници СК**

Администратор и систем (програм)

**Предуслов :** Систем је укључен и приказана је форма за пријаву администратора на систем.

**Основини сценарио СК**

1. Администратор **уноси** податке за пријављивање. (АПУСО)
2. Администратор **проверава** да ли је исправно унео своје податке. (АНСО)
3. Администратор **позива** систем да пријави администратора. (АПСО)
4. Систем **пријављује** администратора. (СО)
5. Систем **приказује** администратору поруку : „Успешна пријава“. (ИА)

**Алтернативна сценарија**

* 1. Уколико систем буде неуспешан у пријави администратора, приказује поруку: „Унели сте погрешне креденцијале“. (ИА)

### СК3 : Случај коришћења – Брисање клијента

**Назив СК**

Брисање клијента

**Актори СК**

Администратор

**Учесници СК**

Администратор и систем (програм)

**Предуслов :** Систем је укључен и администратор је пријављен на систем под својом шифром. Систем приказује форму за рад са клијентима. Листа клијената је учитана.

**Основни сценарио СК**

1. Администратор **уноси** критеријум претраге клијената. (АПУСО)
2. Администратор **контролише** исправност унетог критеријума. (АНСО)
3. Администратор **позива** систем да нађе клијента користећи задати критеријум. (АПСО)
4. Систем **тражи** клијента по задатом критеријуму. (СО)
5. Систем **приказује** администратору пронађене клијенте. (ИА)
6. Администратор **бира** клијента чије податке намерава да обрише. (АПУСО)
7. Администратор **позива** систем да учита податке изабраног клијента. (АПСО)
8. Систем **учитава** податке о изабраном клијенту. (СО)
9. Систем **приказује** администратору одабраног клијента. (ИА)
10. Администратор **позива** систем да обрише клијента. (АПСО)
11. Систем **брише** клијента. (СО)
12. Систем **приказује** поруку: „Клијент је успешно обрисан“. (ИА)

**Алтернативна сценарија**

* 1. Уколико систем не нађе клијента који испуњава задати критеријум он приказује поруку администратору: „Није пронађен ниједан клијент по задатом критеријуму.“ Прекида се извршење сценарија. (ИА)

9.1 Уколико систем не може да учита клијента он приказује администратору поруку: “Систем не може да учита клијента ”. Прекида се извршење сценарија. (ИА)

* 1. Уколико систем не може да обрише клијента он приказује администратору поруку: „Клијент не може бити обрисан. Покушајте поново“. (ИА)

### СК4 : Случај коришћења – Измена података о клијенту

**Назив СК**

Измена података о клијенту

**Актори СК**

Администратор

**Учесници СК**

Администратор и систем (програм)

**Предуслов :** Систем је укључен и администратору је улогован. Систем приказује форму за рад са клијентима. Листа клијената и листа типова клијената су учитане.

**Основини сценарио СК**

1. Администратор **уноси** критеријум претраге клијената. (АПУСО)
2. Администратор **позива** систем да нађе клијента користећи задати критеријум. (АПСО)
3. Систем **тражи** клијента по задатом критеријуму. (СО)
4. Систем **приказује** администратору пронађене клијенте. (ИА)
5. Администратор **бира** клијента чије податке намерава да обрише. (АПУСО)
6. Администратор **позива** систем да учита податке изабраног клијента. (АПСО)
7. Систем **учитава** податке о изабраном клијенту. (СО)
8. Систем **приказује** администратору изабраног клијента. (ИА)
9. Администратор **мења** податке о клијенту. (АПУСО)
10. Администратор **проверава** да ли су унети подаци исправни. (АНСО)
11. Администратор **позива** систем да измени податке. (АПСО)
12. Систем **мења** податке о клијенту. (СО)
13. Систем **приказује** поруку: „Измена података је успешна“. (ИА)

**Алтернативна сценарија**

4.1 Уколико систем не нађе клијента који испуњава задати критеријум он приказује поруку администратору: „Није пронађен ниједан клијент по задатом критеријуму.“ Прекида се извршење сценарија. (ИА)

8.1 Уколико систем не може да учита клијента он приказује администратору поруку “Систем не може да учита одабраног клијента ”. Прекида се извршење сценарија. (ИА)

13.1 Уколико систем не може да промени податке о клијенту приказује поруку администратору: „Систем не може да измени клијента. Покушајте поново“. (ИА)

### СК5 : Случај коришћења – Унос нове услуге

**Назив СК**

Унос нове услуге

**Актори СК**

Администратор

**Учесници СК**

Администратор и систем (програм)

**Предуслов**: Систем је укључен и администратор je улогован под својом шифром. Систем приказује форму за унос нове услуге. Учитана је листа услуга и листу напомена.

**Основни сценарио СК**

1. Администратор **проверава** стање клијента. (АНСО)
2. Администратор **одлучује** о услузи клијента на основу прегледа. (АНСО)
3. Администратор **уноси** податке о услузи. (АПУСО)
4. Администратор **проверава** да ли су подаци о услузи исправно унети у форму. (АНСО)
5. Администратор **позива** систем да запамти податке о новој услузи. (АПСО)
6. Систем **памти** податке о новој услузи. (СО)
7. Систем **приказује** администратору поруку: „Услуга је успешно запамћена“. (ИА)

**Алтернативна сценарија**

7.1 Уколико систем не може да запамти податке о новонасталој услузи он приказује поруку администратору: „Систем не може да запамти услугу. Покушајте поново“. (ИА)

### СК6 : Случај коришћења – Брисање услуге

**Назив СК**

Брисање услуге

**Актори СК**

Администратор

**Учесници СК**

Администратор и систем (програм)

**Предуслов :** Систем је укључен и администратор је улогован на систем под својом шифром. Систем приказује форму за рад са услугама. Учитана је листа услуга.

**Основни сценарио СК**

1. Администратор **уноси** критеријум по ком претражује услуге. (АПУСО)
2. Администратор **проверава** да ли је критеријум исправно унет. (АНСО)
3. Администратор **позива** систем да нађе услугу по задатом критеријуму. (АПСО)
4. Систем **тражи** услуге по задатом критеријуму. (СО)
5. Систем **приказује** администратору услуге. (ИА)
6. Администратор **бира** услугу коју жели да обрише. (АПУСО)
7. Администратор **позива** систем да учита податке о одабраној услузи. (АПСО)
8. Систем **учитава** податке о одабраној услузи. (СО)
9. Систем **приказује** администратору одабрану услугу. (ИА)
10. Администратор **позива** систем да обрише услугу. (АПСО)
11. Систем **брише** услугу. (СО)
12. Систем **приказује** администратору поруку: „Услуга је успешно обирсана“. (ИА)

**Алтернативна сценарија**

5.1. Уколико систем не нађе услугу он приказује поруку администратору: „Услуга са задатим критеријумом није пронађена“. Прекида се извршење сценарија. (ИА)

9.1 Уколико систем не може да учита услугу он приказује администратору поруку “Систем не може да учита услугу ”. Прекида се извршење сценарија. (ИА)

12.1 Уколико систем не може да обрише услугу он приказује поруку администратору: „Услуга није обирсана. Покушаје поново“. (ИА)

### СК7 : Случај коришћења – Унос нових термина

**Назив СК**

Унос нових термина

**Актори СК**

Администратор

**Учесници СК**

Администратор и систем (програм)

**Предуслов**: Систем је укључен и администратор je улогован под својом шифром. Систем приказује форму за унос заказаног термина. Листа услуга и листа клијената су учитане.

**Основни сценарио СК**

1. Администратор **уноси** податке о заказаним терминима. (АПУСО)
2. Администратор **проверава** да ли исправно унео податке о терминима. (АНСО)
3. Администратор **позива** систем да запамти податке о заказаним терминима. (АПСО)
4. Систем **памти** податке о заказаним терминима. (СО)
5. Систем **приказује** администратору запамћене термине уз поруку: „Термини су успешно запамћени“. (ИА)

**Алтернативна сценарија**

5.1 Уколико систем не успе да запамти податке о новим заказаним терминима он приказује поруку администратору: „Термини нису запамћени. Покушајте поново“. (ИА)

### СК8 : Случај коришћења – Брисање заказаног термина

**Назив СК**

Брисање заказаног термина

**Актори СК**

Администратор

**Учесници СК**

Администратор и систем (програм)

**Предуслов**: : Систем је укључен и администратор je улогован под својом шифром. Систем приказује форму за рад са постојећим терминима. Учитана је листа заказаних термина.

**Основни сценарио СК**

1. Администратор **уноси** критеријум по ком врши претрагу заказаних термина. (АПУСО)
2. Администратор **проверава** исправност унетих критеријума. (АНСО)
3. Администратор **позива** систем да нађе термине по задатом критеријуму. (АСПО)
4. Систем **тражи** заказане термине по задатом критерујуму. (СО)
5. Систем **приказује** администратору заказане термине. (ИА)
6. Администратор **бира** заказани термин који жели да обрише (АПУСО)
7. Администратор **позива** систем да учита податке о изабраном термину. (АПСО)
8. Систем **учитава** податке о изабраном термину. (СО)
9. Систем **приказује** администратору о изабраном термину. (ИА)
10. Администратор **позива** систем да обрише изабрани термин. (АПСО)
11. Систем **брише** одабрани термин. (СО)
12. Систем **приказује** поруку администратору: „Термин је успешно обрисан“. (ИА)

**Алтернативна сценарија**

* 1. Уколико систем не нађе заказане термине он приказује администратору поруку: „Систем не може да нађе термине по задатом критеријуму“. Прекида се извршење сценарија. (ИА)

9.1 Уколико систем не може да учита податке о одабраном термину он приказује администратору поруку “Систем не може да учита одабрани термин ”. Прекида се извршење сценарија. (ИА)

* 1. Уколико систем не може да обрише изабрани термин он приказује поруку администратору: „Термин није обрисан. Покушајте поново“. (ИА)

### СК9 : Случај коришћења – Претрага заказаних термина

**Назив СК**

Претрага заказаних термина

**Актори СК**

Администратор

**Учесници СК**

Администратор и систем (програм)

**Предуслов**: : Систем је укључен и администратор je улогован под својом шифром. Систем приказује форму за рад са постојећим терминима. Учитана је листа заказаних термина.

**Основни сценарио СК**

1. Администратор **уноси** критеријум за тражење заказаних термина. (АПУСО)
2. Администратор **проверава** да ли је унос критеријума исправан. (АНСО)
3. Администратор **позива** систем да нађе заказане термине по задатом критеријуму. (АПСО)
4. Систем **тражи** заказане термине по задатом критеријуму. (СО)
5. Систем **приказује** администратору заказане термине. (ИА)
6. Администратор **бира** заказани термин који жели. (АПУСО)
7. Администратор **позива** систем да учита податке о изабраном термину. (АПСО)
8. Систем **учитава** податке изабраног термина. (СО)
9. Систем **приказује** администратору податке о изабраном термину. (ИА)

**Алтернативна сценарија**

* 1. Уколико систем не нађе заказане темине он приказује поруку администратору:„Систем не може да нађе термине по задатом критеријуму“. Прекида се извршење сценарија. (ИА)

9.1 Уколико систем не може да учита податке о одабраном термину он приказује администратору поруку “Систем не може да учита одабрани термин. Покушајте поново ”. (ИА)

# Анализа софтверског система

## Понашање софтверског система – Системски дијаграм секвенци

### ДС1: Дијаграм секвенци случаја коришћења – Пријава администратора на систем

Diagram, box and whisker chart

Description automatically generated**Основни сценарио СК**

1. Администратор **позива** систем да пријави администратора. (АПСО)
2. Систем **приказује** администратору поруку : „Успешна пријава“. (ИА)

**Aлтернативна сценарија**

* 1. Уколико систем није успео да пријави администратора, он приказује поруку: „Унели сте погрешне креденцијале“. (ИА)

Diagram, box and whisker chart

Description automatically generated

Системске операције које треба пројектовати на основу дијаграма секвенци:

1. *Signal* **prijaviAdministratora(Administrator)**

### ДС2: Дијаграм секвенци случаја коришћења – Унос новог клијента

**Основни сценарио СК**

1. Форма **позива** систем да прикаже листу типова клијената. (АПСО)
2. Систем **приказује** листу типова клијената. (ИА)
3. Администратор **позива** систем да запамти податке о новом клијенту. (АПСО)
4. Систем **приказује** администратору поруку: “Клијент је успешно запамћен“. (ИА)

Diagram

Description automatically generated

**Алтернативна сценарија**

4.1 Уколико систем не може да унесе податке о клијенту он приказује поруку: “Није успело памћење клијента. Покушајте поново”. (ИА)

Diagram

Description automatically generated

Системске операције које треба пројектовати на основу дијаграма секвенци:

1. *Signal* **vratiListuTipovaKlijenata(List<TipKlijenta>)**
2. *Signal* **zapamtiKlijenta(Klijent)**

### ДС3: Дијаграм секвенци случаја коришћења – Брисање клијента

**Основни сценарио СК**

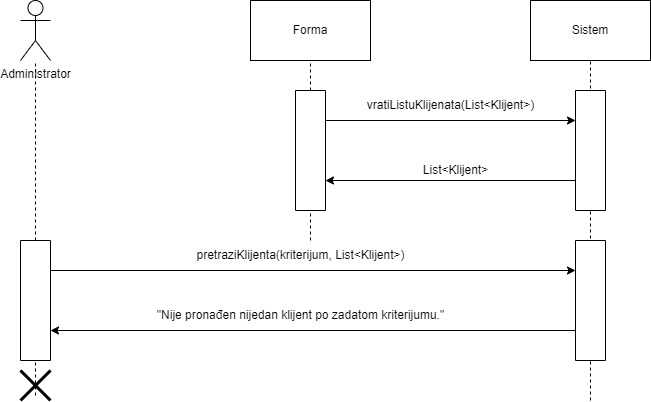
1. Форма **позива** систем да прикаже листу клијената. (АПСО)
2. Систем **приказује** листу клијената. (ИА)
3. Администратор **позива** систем да нађе клијенте по задатом критеријуму. (АПСО)
4. Систем **приказује** администратору клијенте. (ИА)
5. Администратор **позива** систем да учита податке о одабраном клијенту. (АПСО)
6. Систем **приказује** администратору одабраног клијента. (ИА)
7. Администратор **позива** систем да обрише клијента. (АПСО)
8. Систем **приказује** администратору поруку: “Клијент је успешно обрисан.” (ИА)

Diagram

Description automatically generated

**Алтернативна сценарија**

* 1. Уколико систем не може да нађе клијенте он приказује администратору поруку: “Није пронађен ниједан клијент по задатом критеријуму”. Прекида се извршење сценарија. (ИА)



6.1 Уколико систем не може да учита клијента он приказује администратору поруку “Систем не може да учита одабраног клијента ”. Прекида се извршење сценарија. (ИА)

Diagram, engineering drawing

Description automatically generated

* 1. Уколико систем не може да обрише клијента он приказује администратору поруку “Клијент не може бити обрисан”. (ИА)

Diagram

Description automatically generated

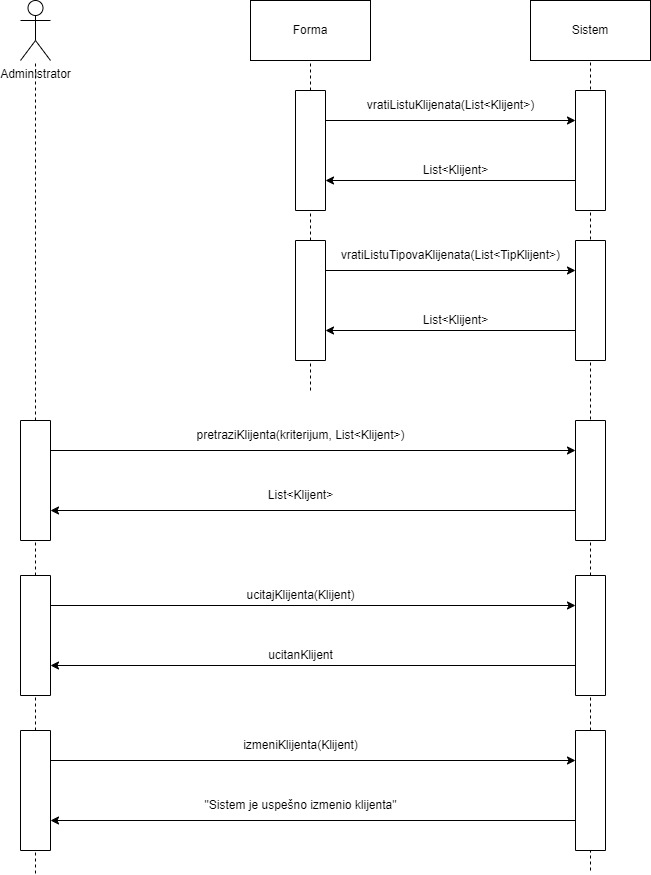
Системске операције које треба пројектовати на основу дијаграма секвенци:

1. *Signal* **vratiListuKlijenata(List<Klijent>)**
2. *Signal* **pretraziKlijente(kriterijum, List<Klijent>)**
3. *Signal* **ucitajKlijenta(Klijent)**
4. *Signal* **obrisiKlijenta(Klijent)**

### ДС4: Дијаграм секвенци случаја коришћења – Измена података о клијенту

**Основни сценарио СК**

1. Форма **позива** систем да прикаже листу клијената. (АПСО)
2. Систем **приказује** листу клијената. (ИА)
3. Форма **позива** систем да прикаже листу типова клијената. (АПСО)
4. Систем **приказује** листу клијената. (ИА)
5. Администратор **позива** систем да нађе клијенте по задатом критеријуму. (АПСО)
6. Систем **приказује** администратору клијентe. (ИА)
7. Администратор **позива** систем да учита податке о одабраном клијенту. (АПСО)
8. Систем **приказује** администратору одабраног клијента. (ИА)
9. Администратор **позива** систем да измени податке о клијенту. (АПСО)
10. Систем **приказује** администратору поруку: “Систем је успешно изменио клијента.” (ИА)



**Алтернативна сценарија**

6.1 Уколико систем не може да нађе клијенте он приказује администратору поруку: “Није пронађен ниједан клијент по задатом критеријуму”. Прекида се извршење сценарија. (ИА)

Diagram

Description automatically generated

8.1 Уколико систем не може да учита клијента он приказује администратору поруку “Систем не може да учита одабраног клијента ”. Прекида се извршење сценарија. (ИА)

Diagram

Description automatically generated

* 1. Уколико систем не може да измени податке о клијенту он приказује администратору поруку “Систем не може да измени клијента”. (ИА)

Diagram

Description automatically generated

Системске операције које треба пројектовати на основу дијаграма секвенци:

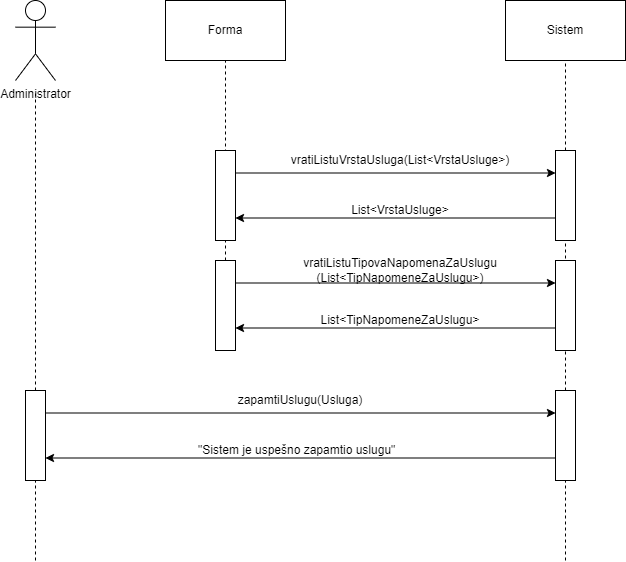
1. *Signal* **vratiListuKlijenata(List<Klijent>)**
2. *Signal* **vratiListuTipovaKlijenata(List<TipovaKlijenata>)**
3. *Signal* **pretraziKlijente(kriterijum, List<Klijent>)**
4. *Signal* **ucitajKlijenta(Klijent)**
5. *Signal* **izmeniKlijenta(Klijent)**

### ДС5: Дијаграм секвенци случаја коришћења – Унос нове услуге

**Основни сценарио СК**

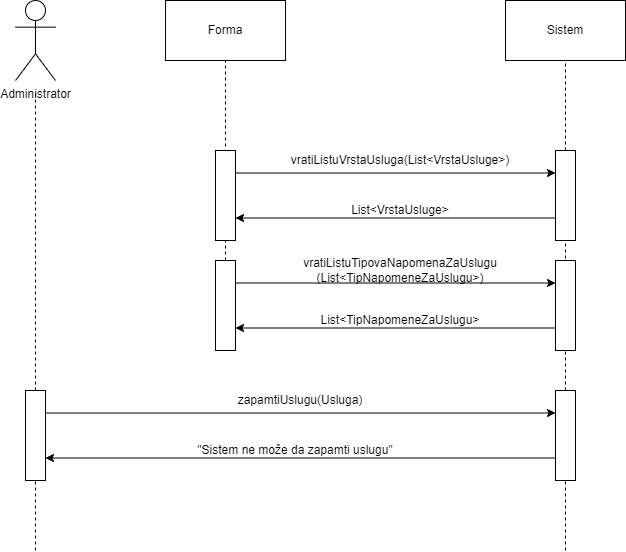
1. Форма **позива** систем да прикаже листу услуга. (АПСО)
2. Систем **приказује** листу услуга. (ИА)
3. Форма **позива** систем да прикаже листу напомена за услугу. (АПСО)
4. Систем **приказује** листу напомена за услугу. (ИА)
5. Администратор **позива** систем да запамти податке о новој услузи. (АПСО)
6. Систем **приказује** администратору поруку: “Услуга је успешно запамћена“.

(ИА)



**Алтернативна сценарија**

* 1. Уколико систем не може да запамти податке о новој услузи он приказује администратору поруку “Систем не може да запамти услугу”. (ИА)



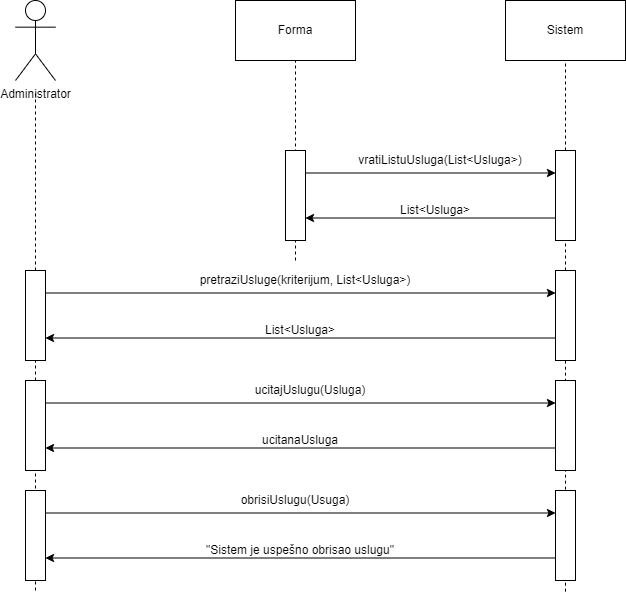
Са наведених дијаграма секвенци уочавају се системске операције које треба пројектовати:

1. *Signal* **vratiListuTipovaUsluga(List<VrstaUsluge>)**
2. *Signal* **zapamtiUslugu(Usluga)**
3. *Signal* **vratiListuTipovaNapomena(List<TipNapomene>)**

### ДС6: Дијаграм секвенци случаја коришћења – Брисање услуге

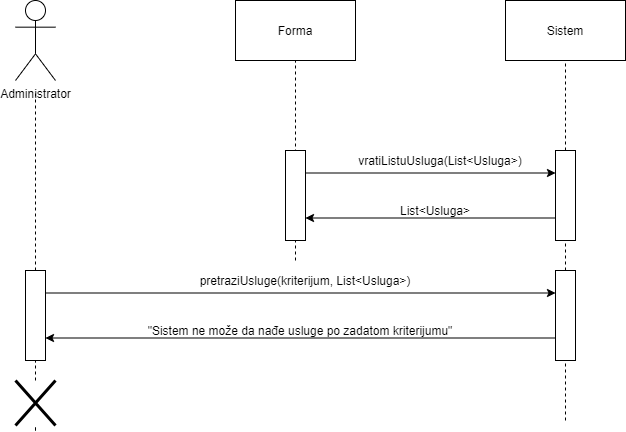
**Основни сценарио СК**

1. Форма **позива** систем да прикаже листу услуга. (АПСО)
2. Систем **приказује** листу услуга. (ИА)
3. Администратор **позива** систем да нађе услуге по задатом критеријуму. (АПСО)
4. Систем **приказује** администратору услуге. (ИА)
5. Администратор **позива** систем да учита податке о одабраној услузи. (АПСО)
6. Систем **приказује** администратору одабрану услугу. (ИА)
7. Администратор **позива** систем да обрише услугу. (АПСО)
8. Систем **приказује** администратору поруку: “Услуга је успешно обрисана.” (ИА)

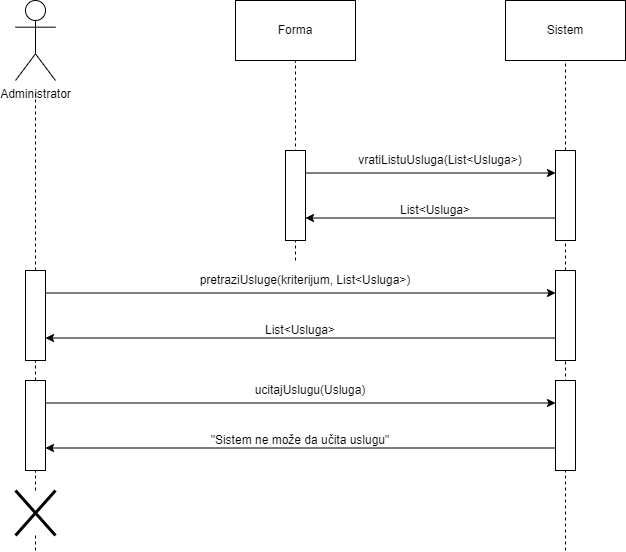


**Алтернативна сценарија**

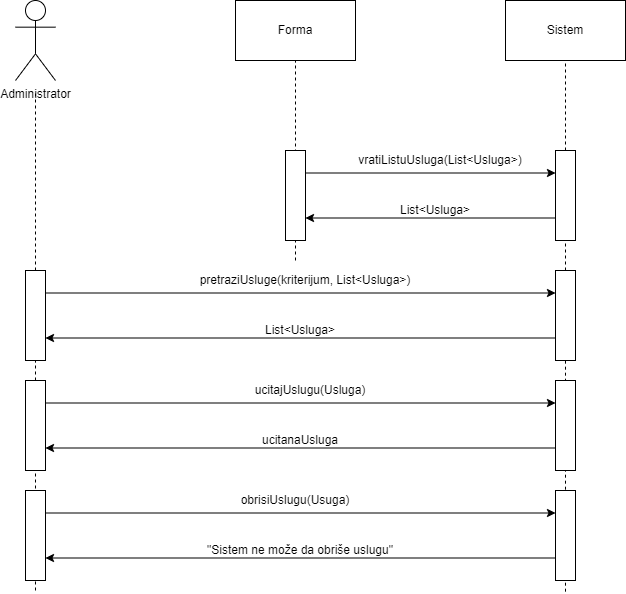
* 1. Уколико систем не може да нађе услуге он приказује администратору поруку: “Није пронађена ниједна услуга са задатим критеријумом”. (ИА)



6.1 Уколико систем не може да учита услугу он приказује администратору поруку “Систем не може да учита услугу ”. Прекида се извршење сценарија. (ИА)



* 1. Уколико систем не може да обрише услугу он приказује администратору поруку “Услуга није обрисана”. (ИА)



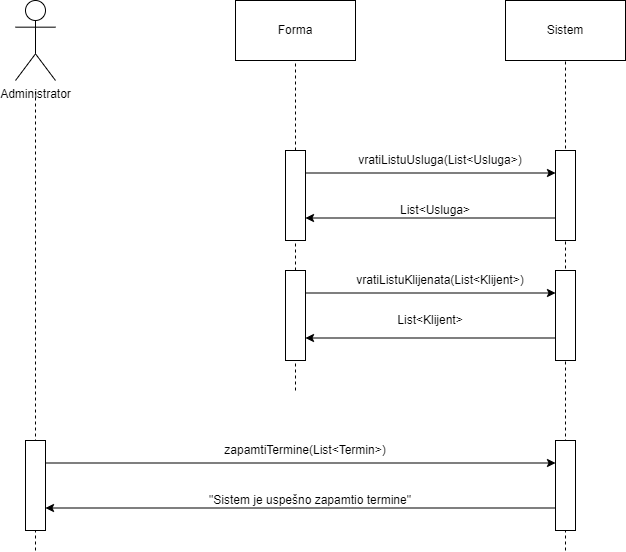
Са наведених дијаграма секвенци уочавају се системске операције које треба пројектовати:

1. *Signal* **vratiListuUsluga(List<Usluga>)**
2. *Signal* **pretraziUslugee(kriterijum, List<Usluga>)**
3. *Signal* **ucitajUslugu(Usluga)**
4. *Signal* **obrisiUslugu(Usluga)**

### ДС7: Дијаграм секвенци случаја коришћења – Унос нових термина

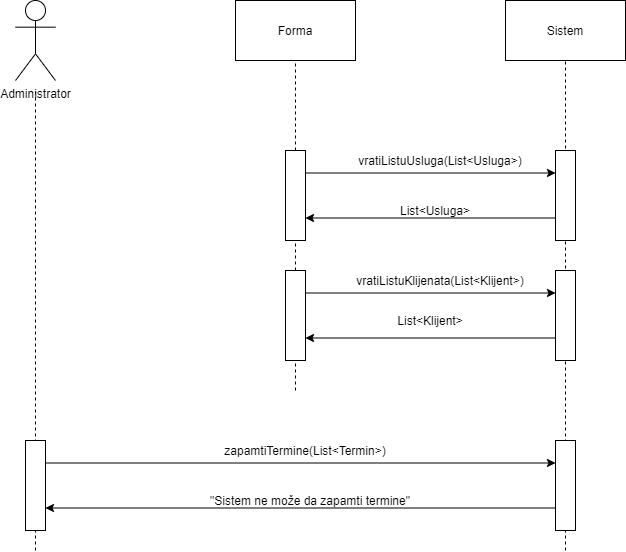
**Основни сценарио СК**

1. Форма **позива** систем да прикаже листу услуга. (АПСО)
2. Систем **приказује** листу услуга. (ИА)
3. Форма **позива** систем да прикаже листу клијената. (АПСО)
4. Систем **приказује** листу клијената. (ИА)
5. Администратор **позива** систем да запамти податке о заказаним терминима. (АПСО)
6. Систем **приказује** администратору поруку: “Термини су успешно запамћени“. (ИА)



**Алтернативна сценарија**

* 1. Уколико систем не може да запамти податке о заказаним терминима он приказује администратору поруку: “Термини нису запамћени”. (ИА)



Са наведених дијаграма секвенци уочавају се системске операције које треба пројектовати:

1. *Signal* **vratiListuUsluga(List<Usluga>)**
2. *Signal* **vratiListuKlijenata(List<Klijent>)**
3. *Signal* **zapamtiTerminе(****List< Termin>)**

### ДС8: Дијаграм секвенци случаја коришћења – Брисање заказаног термина

**Основни сценарио СК**

1. Форма **позива** систем да прикаже листу заказаних термина. (АПСО)
2. Систем **приказује** листу заказаних термина. (ИА)
3. Администратор **позива** систем да нађе заказане термине по задатом критеријуму. (АПСО)
4. Систем **приказује** администратору заказане термине. (ИА)
5. Администратор **позива** систем да учита податке о одабраном термину. (АПСО)
6. Систем **приказује** администратору податке о одабраном термину. (ИА)
7. Администратор **позива** систем да обрише одабрани термин. (АПСО)
8. Систем **приказује** администратору поруку: “Термин је успешно обрисан.” (ИА)

Diagram

Description automatically generated

**Алтернативна сценарија**

* 1. Уколико систем не може да нађе заказане термине он приказује администратору поруку: “Систем не може да нађе термине по задатом критеријуму”. Прекида се извршење сценарија. (ИА)

Diagram

Description automatically generated

6.1 Уколико систем не може да учита податке о одабраном термину он приказује администратору поруку: “Систем не може да учита одабрани термин ”. Прекида се извршење сценарија. (ИА)

Diagram

Description automatically generated

8.1 Уколико систем не може да обрише одабрани термин он приказује администратору поруку: “Термин није обрисан”. (ИА)

Diagram

Description automatically generated

Са наведених дијаграма секвенци уочавају се системске операције које треба пројектовати:

1. *Signal* **vratiListuTermina(List<Termin>)**
2. *Signal* **pretraziTermine(kriterijum, List<Termin>)**
3. *Signal* **ucitajTermin(Termin)**
4. *Signal* **obrisiTermin(Termin)**

### ДС9: Дијаграм секвенци случаја коришћења – Претрага заказаних термина

**Основни сценарио СК**

1. Форма **позива** систем да прикаже листу заказаних термина. (АПСО)
2. Систем **приказује** листу заказаних термина. (ИА)
3. Администратор **позива** систем да нађе заказане термине по задатом критеријуму. (АПСО)
4. Систем **приказује** администратору заказане термине. (ИА)
5. Администратор **позива** систем да учита податке о одабраном термину. (АПСО)
6. Систем **приказује** администратору податке о одабраном термину. (ИА)

Diagram

Description automatically generated

**Алтернативна сценарија**

* 1. Уколико систем не може да нађе заказане термине он приказује администратору поруку: “Систем не може да нађе термине по задатом критеријуму”. Прекида се извршење сценарија. (ИА)

Diagram

Description automatically generated

6.1 Уколико систем не може да учита податке о одабраном термину он приказује администратору поруку “Систем не може да учита одабрани термин ”. (ИА)

Diagram

Description automatically generated

Са наведених дијаграма секвенци уочавају се системске операције које треба пројектовати:

1. *Signal* **vratiListuTermina(List<Termin>)**
2. *Signal* **pretraziTermine(kriterijum, List<Termin>)**
3. *Signal* **ucitajTermin(Termin)**

На основу свиx сценарија добијене су следеће системске операције:

1. *Signal* prijaviAdministratora(Administrator)
2. *Signal* vratiListuTipovaKlijenata(List<TipKlijenta>)
3. *Signal* vratiListuKlijenata(List<Klijent>)
4. *Signal* ucitajKlijenta(Klijent)
5. *Signal* pretraziKlijente(kriterijum, List<Klijent>)
6. *Signal* zapamtiKlijenta(Klijent)
7. *Signal* izmeniKlijenta(Klijent)
8. *Signal* obrisiKlijenta(Klijent)
9. *Signal* vratiListuTipovaUsluge(List<TipUsluge>)
10. *Signal* vratiListuTerapija(List<Terapije>)
11. *Signal* vratiListuTipovaNapomena(List<TipNapomeneu>)
12. *Signal* ucitajTerapiju(Terapija)
13. *Signal* zapamtiTerapiju(Terapija)
14. *Signal* obrisiTerapiju(Terapija)
15. *Signal* vratiListuTermina(List<Termin>)
16. *Signal* pretraziTermine(kriterijum, List<Termin>)
17. *Signal* ucitajTermin(Termin).
18. *Signal* zapamtiTerminе(List< Termin>)
19. *Signal* obrisiTermin(Termin)

## Понашање софтверског система – Дефинисање уговора

### Уговор УГ1: Пријави администратора

**Операција:** *prijaviAdministratora(Administrator):* signal,

**Веза са СК:** СК1

**Предуслови:** /

**Постуслови:** /

### Уговор УГ2: Врати листу типова клијената

**Операција:** *vratiListuTipovaKlijenata(List<TipKlijenta>):* signal,

**Веза са СК:** СК2, СК4

**Предуслови:** /

**Постуслови:** /

### Уговор УГ3: Врати листу клијената

**Операција:** *vratiListuKlijenata(List<Klijent>):* signal,

**Веза са СК:** СК3, СК4, СК7

**Предуслови:** /

**Постуслови:** /

### Уговор УГ4: Учитај клијента

**Операција:** *ucitajKlijenta(Klijent):* signal,

**Веза са СК:** СК3, СК4

**Предуслови:** /

**Постуслови:** /

### Уговор УГ5: Претражи клијенте

**Операција:** *pretraziKlijente(kriterijum, List<Klijent>):* signal,

**Веза са СК:** СК3, СК4

**Предуслови:** /

**Постуслови:** /

### Уговор УГ6: Запамти клијента

**Операција:** *zapamtiKlijenta(Klijent):* signal,

**Веза са СК:** СК2, СК4

**Предуслови:** Вредносна и структурна ограничења над објектом Клијент морају бити задовољена.

**Постуслови:** Запамћен је нови клијент.

### Уговор УГ7: Измени клијента

**Операција:** *izmeniKlijenta(Klijent):* signal,

**Веза са СК:** СК4

**Предуслови:** Вредносна и структурна ограничења над објектом Клијент морају бити задовољена.

**Постуслови:** Клијент је измењен.

### Уговор УГ8: Обриши клијента

**Операција:** *obrisiKlijenta(Klijent):* signal,

**Веза са СК:** СК4

**Предуслови:** Структурна ограничења над објектом Клијент морају бити задовољена.

**Постуслови:** Клијент је обрисан.

### Уговор УГ9: Врати листу врста услуга

**Операција:** *vratiListuVrstaUsluga(List<VrstaTerapije>):* signal,

**Веза са СК:** СК5

**Предуслови:** /

**Постуслови:** /

### Уговор УГ10: Врати листу услуга

**Операција:** *vratiListuUsluga(List<Usluga>):* signal,

**Веза са СК:** СК5

**Предуслови:** /

**Постуслови:** /

### Уговор УГ11: Врати листу типова напомена за услугу

**Операција:** *vratiListuTipovaNapomena(List<TipNapomena>):* signal,

**Веза са СК:** СК5

**Предуслови:** /

**Постуслови:** /

### Уговор УГ12: Учитај услугу

**Операција:** *ucitajTerapiju(Terapija):* signal,

**Веза са СК:** СК7

**Предуслови:** /

**Постуслови:** /

### Уговор УГ13: Претражи услугу

**Операција:** *pretraziTerapije(**kriterijum, List<Terapija>):* signal,

**Веза са СК:** СК6

**Предуслови:** /

**Постуслови:** /

### Уговор УГ14: Запамти услугу

**Операција:** *zapamtiTerapiju(Terapija):* signal,

**Веза са СК:** СК5

**Предуслови:** Вредносна и структурна ограничења над објектом Услуга морају бити задовољена.

**Постуслови:** Запамћена је нова услуга.

### Уговор УГ15: Обриши услугу

**Операција:** *obrisiTerapiju(Terapija):* signal,

**Веза са СК:** СК5

**Предуслови:** Структурна ограничења над објектом Tерапија морају бити задовољена.

**Постуслови:** Услуга је обрисан.

### Уговор УГ16: Врати листу термина

**Операција:** *vratiListuTermina(List<Termin>):* signal,

**Веза са СК:** СК8, СК9

**Предуслови:** /

**Постуслови:** /

### Уговор УГ17: Претражи термине

**Операција:** *pretraziTermine(kriterijum, List<Termin>):* signal,

**Веза са СК:** СК8, СК9

**Предуслови:** /

**Постуслови:**

### Уговор УГ18: Учитај термин

**Операција:** *ucitajTermin(Termin):* signal,

**Веза са СК:** СК8, СК9

**Предуслови:** /

**Постуслови:** /

### Уговор УГ19: Запамти термине

**Операција:** *zapamtiTerminе(List< Termin>):* signal,

**Веза са СК:** СК7

**Предуслови:** Вредносна и структурна ограничења над објектима Термин морају бити задовољена.

**Постуслови:** Запамћени су нови термини.

### Уговор УГ20: Обриши термин

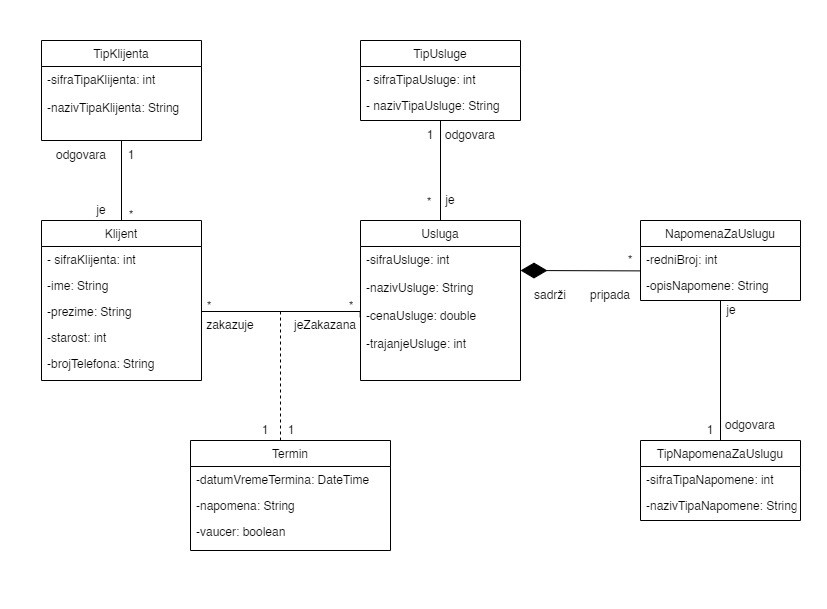
**Операција:** *obrisiTermin(Termin):* signal,

**Веза са СК:** СК8

**Предуслови:** Структурна ограничења над објектом Термин морају бити задовољена.

**Постуслови:** Термин је обрисан.

## Структура софтверског система – концептуални модел



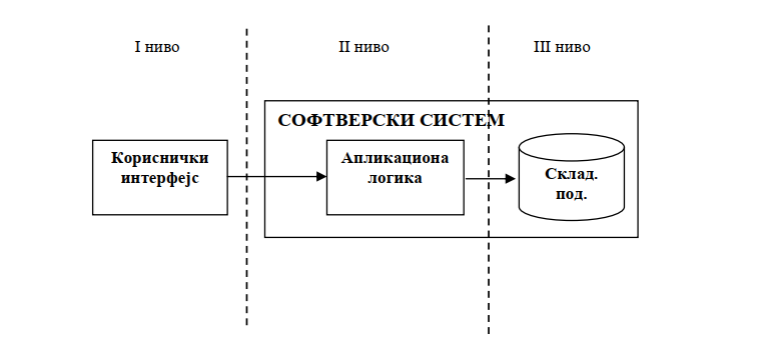
# Пројектовање софтверског система

Ова етапа анализира физичку организацију и функционалност софтверског система, познату као архитектура софтверског система. Архитектура софтверског система је тронивојска и обухвата следеће слојеве:

• Кориснички интерфејс - представља улазно-излазну репрезентацију софтверског система.   
• Апликациона логика - описује структуру и функционалности софтверског система.

• Складиште података - чува стање атрибута софтверског система.

Ниво корисничког интерфејса се налази на страни клијента, док су апликациона логика и складиште података на страни сервера.



*Тронивојска архитектура софтверског система*

Пројектовање архитектуре софтверског система обухвата:

* Пројектовање корисничког интерфејса
* Пројектовање апликационе логике
* Пројектовање складишта података

## Структура софтверског система – релациони модел

Релациони модел правимо на основу концептуалног модела који представља основу за пројектовање релационе базе података.

ТipKlijenta(sifraTipKlijenta, nazivTipaKlijenta)

Klijent(sifraKlijenta, ime, prezime, starost, brojTelefona, *sifraTipaKlijenta*)

TipUsluge(sifraTipaUsluge, nazivTipaUsluge)

Usluga(sifraUsluge, nazivUsluge, cenaUsluge, trajanjeUsluge, *sifraTipaUsluge*)

TipNapomene(sifraTipaNapomene, nazivNapomene)

Napomena(*SifraUsluge*, redniBroj, opis, *SifraTipaNapomene*)

Termin(*sifraKlijenta*, *sifraUsluge*, datumTermina, vaucer, napomena)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела** ТipKlijenta | | **Просто бредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT /  UPDATE CASCADES Klijent  DELETE RESTRICTED Klijent |
| sifraTipaKlijenta | int | not null and > 0 |  |  |
| nazivTipaKlijenta | String | not null |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела** *TipUsluge* | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT /  UPDATE CASCADES Usluga  DELETE RESTRICTED Usluga |
| sifraTipaUsluge | int | not null and > 0 |  |  |
| nazivTipaUsluge | String | not null |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела** *Klijent* | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT RESTRICTED TipKlijenta  UPDATE RESTRICTED TipKlijenta, CASCADES Termin  DELETE CASCADES Termin |
| sifraKlijenta | int | not null and > 0 |  |  |
| ime | String | not null |  |  |
| prezime | String | not null |  |  |
| starost | int | not null and > 14 |  |  |
| brojTelefona | String | not null |  |  |
| *sifraTipaKlijenta* | int | not null and > 0 |  |  |

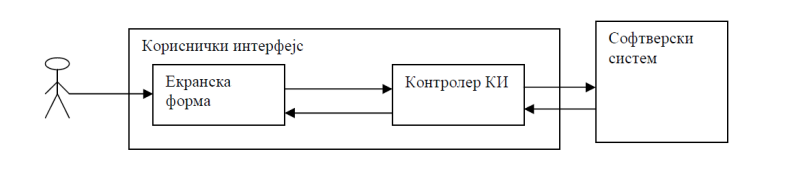
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Табела** *Usluga* | | | **Просто вредносно ограничење** | | | | | **Сложено вредносно ограничење** | | | | | **Структурно ограничење** | |
| **Атрибути** | | **име** | **Тип атрибута** | | **Вредност атрибута** | | **Међузав. атрибута једне табеле** | | | **Међузав. атрибута више табела** | | INSERT RESTRICTED TipUsluge  UPDATE RESTRICTED TipUsluge CASCADES Termin, Napomena  DELETE RESTRICTED Termin CASCADES Napomena | | |
| sifraUsluge | int | | not null and > 0 | |  | | |  | |
| nazivUsluge | String | | not null | |  | | |  | |
| cenaUsluge | double | | not null and > 0 | |  | | |  | |
| trajanjeUsluge | int | | not null and > 0 | |  | | |  | |
| *sifraTipaUsluge* | int | | not null and > 0 | |  | | |  | |
| **Табела** *TipNapomene* | | | | **Просто вредносно ограничење** | | | | | **Сложено вредносно ограничење** | | | | | **Структурно ограничење** | |
| **Атрибути** | **име** | | | **Тип атрибута** | | **Вредност атрибута** | | | **Међузав. атрибута једне табеле** | | **Међузав. атрибута више табела** | | | INSERT /  UPDATE CASCADES Napomena  DELETE RESTRICTED Napomena | |
| sifraTipaNapomene | | | int | | not null and > 0 | | |  | |  | | |
| nazivTipaNapomene | | | String | | not null | | |  | |  | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела** *Napomena* | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT RESTRICTED Usluga, TipNapomene  UPDATE RESTRICTED Usluga, TipNapomene  DELETE / |
| *sifraUsluge* | int | not null and > 0 |  |  |
| redniBroj | int | not null and > 0 |  |  |
| opisNapomene | String | not null |  |  |
| *sifraTipaNapomene* | int | not null and > 0 |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела** *Termin* | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT RESTRICTED Klijent, Usluga  UPDATE RESTRICTED Klijent, Usluga  DELETE / |
| *sifraKlijenta* | int | not null and > 0 |  |  |
| *sifraUsluge* | int | not null and > 0 |  |  |
| datumTermina | DateTime | not null |  |  |
| napomena | String |  |  |  |
|  | Vaucer | boolean |  |  |  |  |

## Пројектовање корисничког интерфејса

Кориснички интефејс се састоји од екранских форми које су одговорне за прихватање и података и догађаја које уносе актори и прослеђивање тих података ка контролeру графичког интерфејса и приказивање одговора система кориснику. Контролер потом прихвата податке које добија од форми, шаље до система, али и прихвата излазе из система и омогућава њихов приказ на графичким елементима.



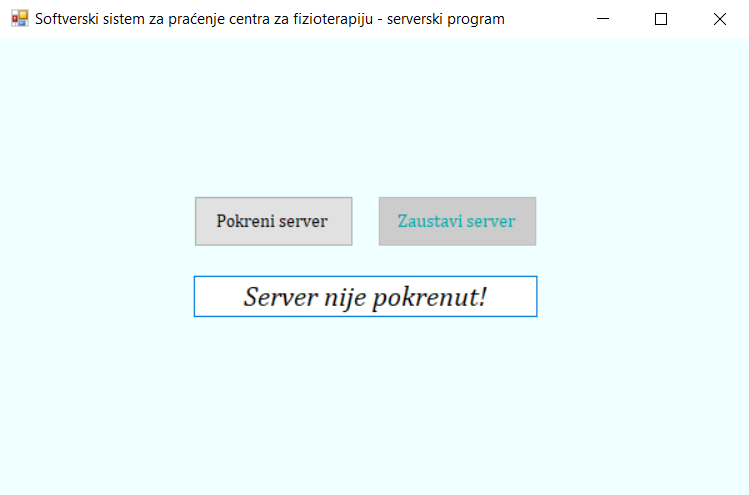
*Структура корисничког интерфејса софтверског система*

Сценарији коришћења екранских форми директно су повезани са сценаријима случајева коришћења.

Сервер апликација има једну екранску форму. Покретање сервера се врши кликом на дугме ***Pokreni server***, док се заустављање сервера врши на дугме ***Zaustavi server***.



*Екранска форма за покренут сервер*



*Екранска форма за непокренут сервер*

На клијентској страни, приликом покретања програма, администратору се приказује форма за пријављивање на систем, а након успешног пријављивања, администратору се приказује главна екранска форма.

### СК1: Случај коришћења – Пријава администратора на систем

**Назив СК**

Пријава администратора на систем

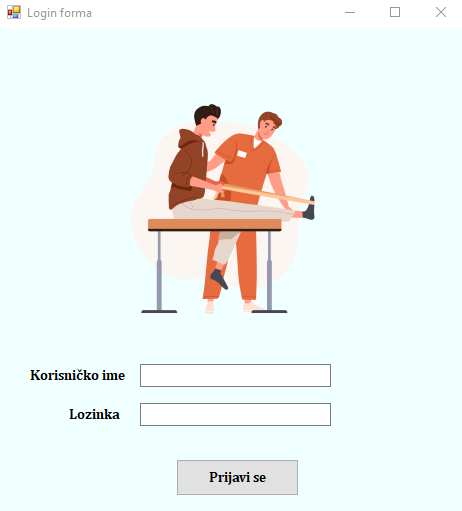
**Актори СК**

Администратор

**Учесници СК**

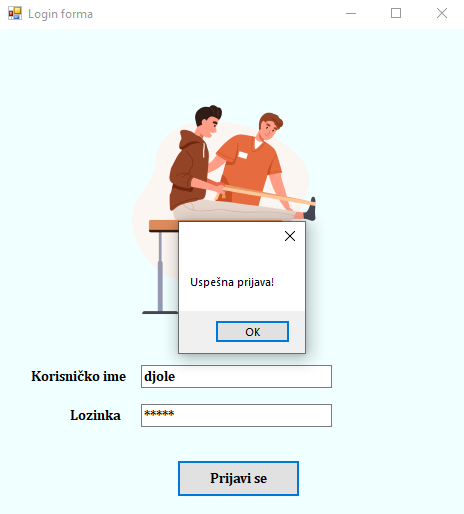
Администратор и систем (програм)

**Предуслов**: Систем је укључен и приказана је форма за пријаву администратора на систем.



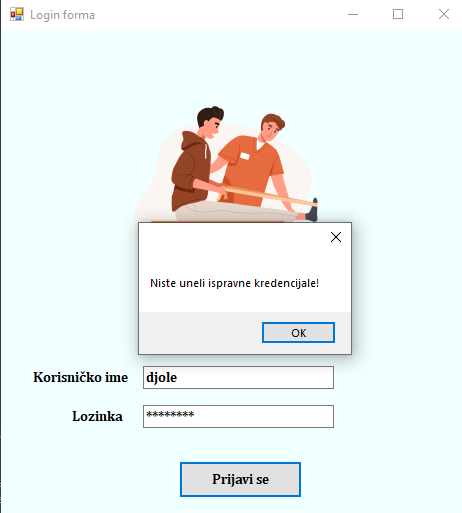
**Основни сценарио СК**

1. Администратор **уноси** своје податке за пријављивање. (АПУСО)
2. Администратор **проверава** да ли је коректно унео податке. (АНСО)
3. Администратор **позива** систем да пријави администратора. (АПСО)
4. Систем **пријављује** администратора. (СО)
5. Систем **приказује** администратору поруку: „Успешна пријава”. (ИА)



Алтернативна сценарија

5.1 Уколико систем није успео да пријави администратора, он приказује поруку: „Нисте унели исправне креденцијале“. (ИА)



### СК2: Случај коришћења – Унос новог клијента

**Назив СК**

Унос новог клијента

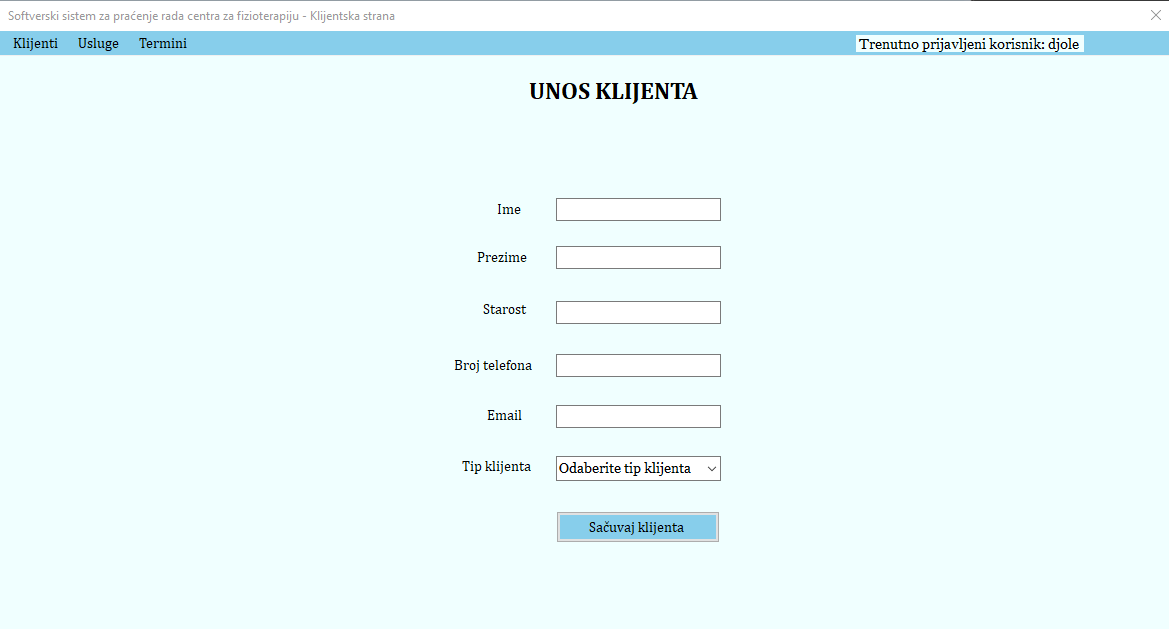
**Актори СК**

Администратор

**Учесници СК**

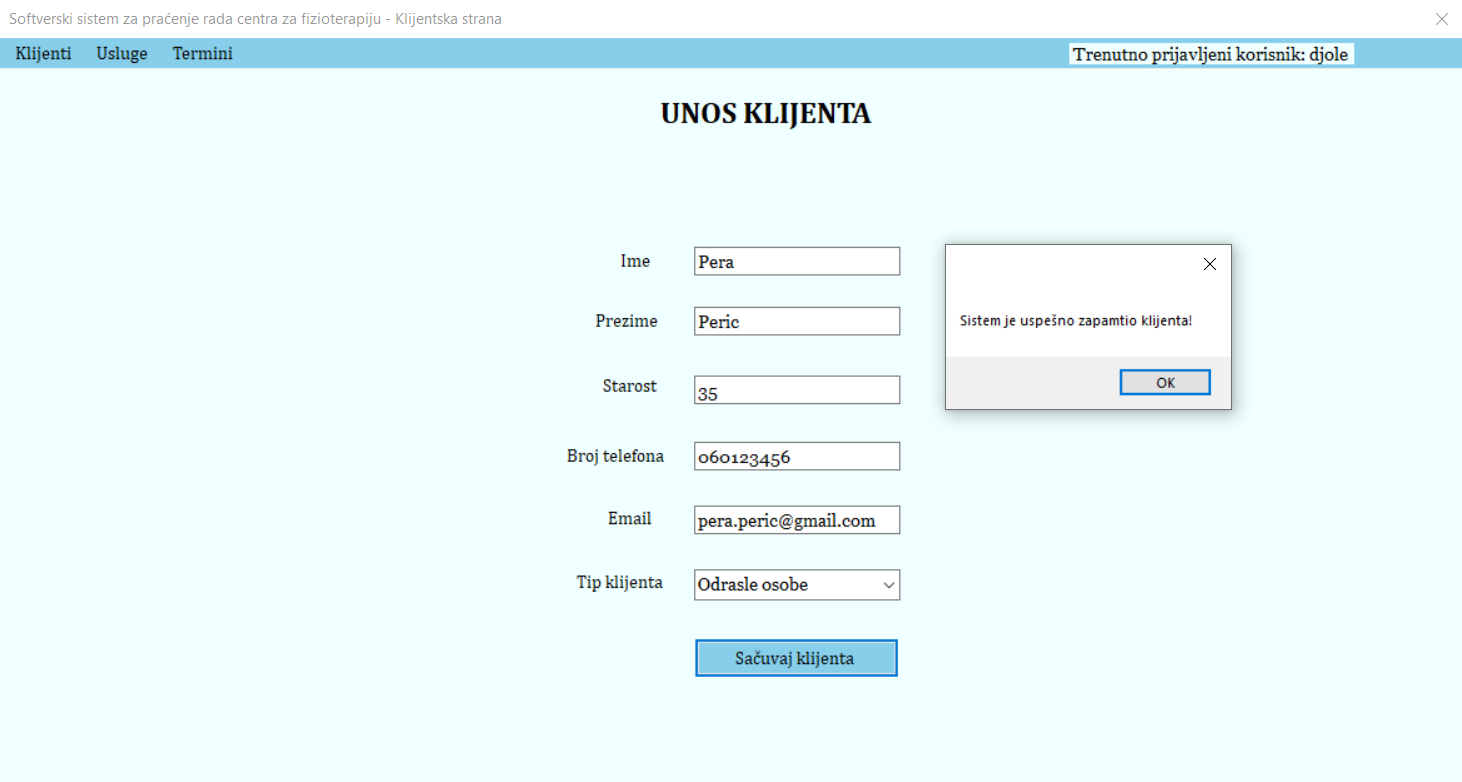
Администратор и систем (програм)

**Предуслов**: Систем је укључен и администратор je улогован под својом шифром. Систем приказује форму за унос новог клијента. Учитана је листа типова клијената.



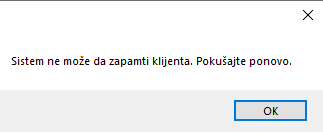
**Основни сценарио СК**

1. Администратор **уноси** податке о новом клијенту у форму. (АПУСО)
2. Администратор **проверава** да ли је коректно унео податке о новом клијенту у форму. (АНСО)
3. Администратор **позива** систем да запамти податке о новом клијенту. (АПСО)
4. Систем **памти** податке о новом клијенту. (СО)
5. Систем **приказује** администратору поруку: “Систем је успешно запамтио клијента“. (ИА)



Алтернативна сценарија

5.1 Уколико систем не може да унесе податке о клијенту он приказује поруку: “Систем не може да запамти клијента. Покушајте поново.” (ИА)



### СК3: Случај коришћења – Брисање клијента

**Назив СК**

Брисање клијента

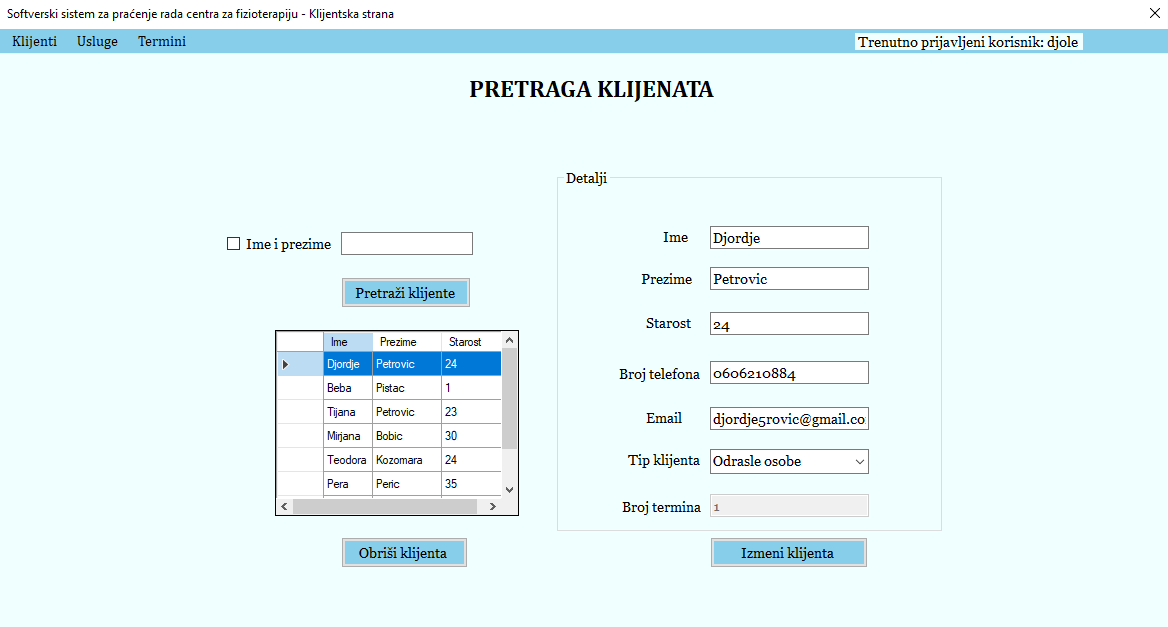
**Актори СК**

Администратор

**Учесници СК**

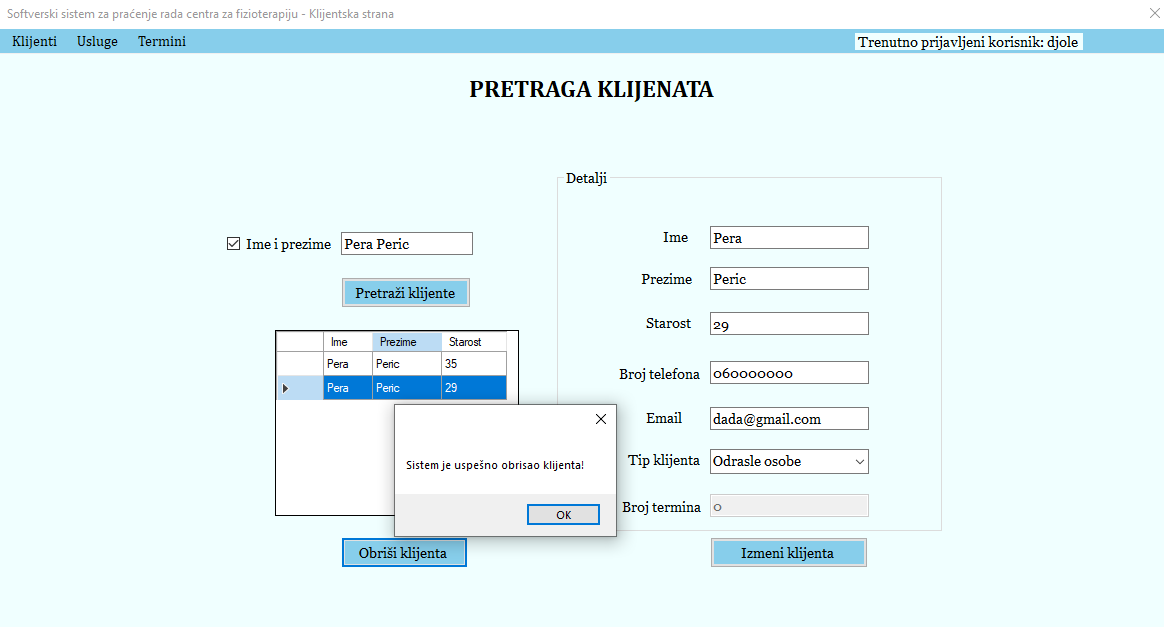
Администратор и систем (програм)

**Предуслов**: Систем је укључен и администратор je улогован под својом шифром. Систем приказује форму за рад са клијентима. Учитана је листа клијената.



**Основни сценарио СК**

1. Администратор **уноси** критеријумпо ком претражује клијенте. (АПУСО)
2. Администратор **проверава** да ли је коректно унео критеријум. (АНСО)
3. Администратор **позива** систем да нађе клијенте по задатом критеријуму. (АПСО)
4. Систем **тражи** клијенте по задатом критеријуму. (СО)
5. Систем **приказује** администратору клијенте. (ИА)
6. Администратор **бира** клијента чије податке жели да обрише. (АПУСО)
7. Администратор **позива** систем да учита податке о одабраном клијенту. (АПСО)
8. Систем **учитава** податке о одабраном клијенту. (СО)
9. Систем **приказује** администратору одабраног клијента. (ИА)
10. Администратор **позива** систем да обрише клијента. (АПСО)
11. Систем **брише** клијента. (СО)
12. Систем **приказује** администратору поруку: “Систем је успешно обрисао клијента.” (ИА)

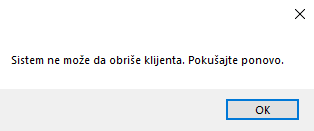


Алтернативна сценарија

5.1 Уколико систем не може да нађе клијенте он приказује администратору поруку: “Систем не може да нађе клијенте по задатом критеријуму”. Прекида се извршење сценарија. (ИА)

9.1 Уколико систем не може да учита клијента он приказује администратору поруку “Систем не може да учита клијента”. Прекида се извршење сценарија. (ИА)

12.1 Уколико систем не може да обрише клијента он приказује администратору поруку “Систем не може да обрише клијента. Покушајте поново”. (ИА)



### СК4: Случај коришћења – Измена података о клијенту

**Назив СК**

Измена података о клијенту

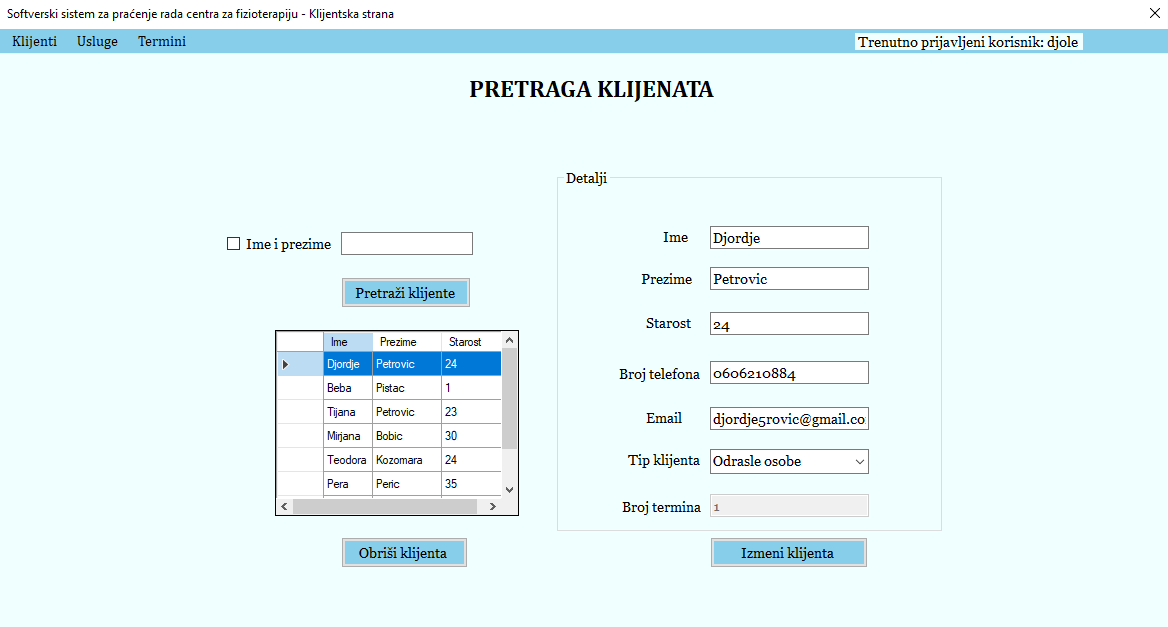
**Актори СК**

Администратор

**Учесници СК**

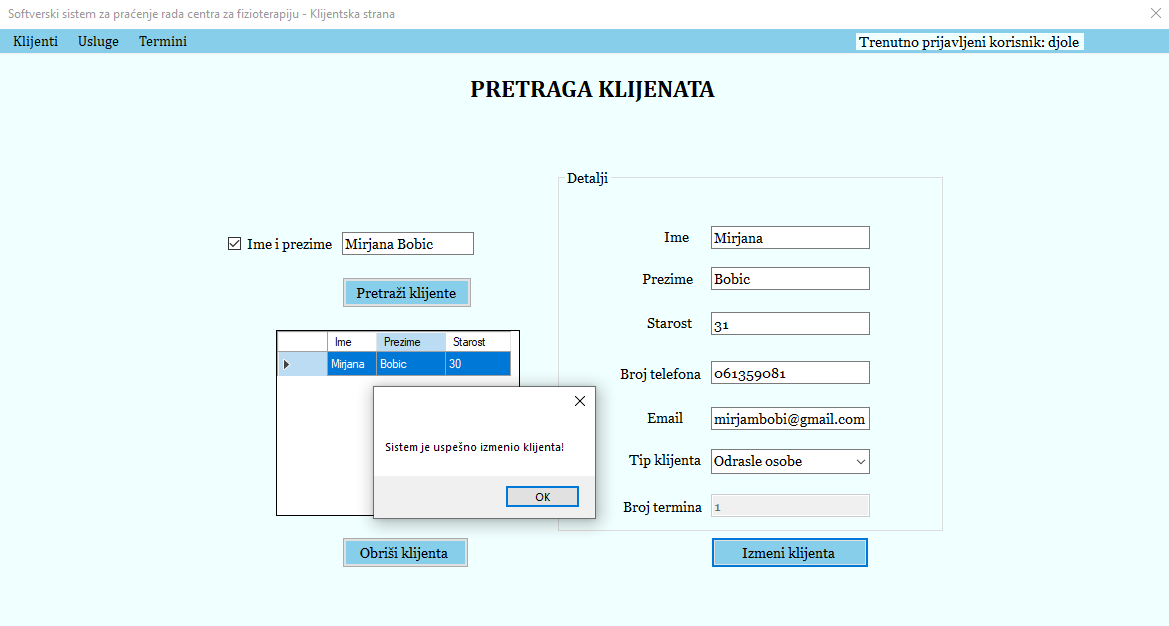
Администратор и систем (програм)

**Предуслов**: Систем је укључен и администратор je улогован под својом шифром. Систем приказује форму за рад са клијентом. Учитана је листа клијената и листа типова клијената.



**Основни сценарио СК**

1. Администратор **уноси** критеријумпо ком претражује клијенте. (АПУСО)
2. Администратор **позива** систем да нађе клијенте по задатом критеријуму. (АПСО)
3. Систем **тражи** клијенте по задатом критеријуму. (СО)
4. Систем **приказује** администратору клијенте. (ИА)
5. Администратор **бира** клијента чије податке треба да промени. (АПУСО)
6. Администратор **позива** систем да учита податке о одабраном клијенту. (АПСО)
7. Систем **учитава** податке о одабраном клијенту. (СО)
8. Систем **приказује** администратору одабраног клијента. (ИА)
9. Администратор **уноси** **(мења)** податке о клијенту. (АПУСО)
10. Администратор **проверава** да ли је коректно унео податке о клијенту. (АНСО)
11. Администратор **позива** систем да измени податке о клијенту. (АПСО)
12. Систем **мења** податке о клијенту. (СО)
13. Систем **приказује** администратору поруку: “Систем је успешно изменио клијента.” (ИА)

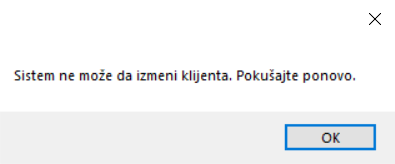


Алтернативна сценарија

4.1 Уколико систем не може да нађе клијенте он приказује администратору поруку: “Систем не може да нађе клијенте по задатом критеријуму”. Прекида се извршење сценарија. (ИА)

8.1 Уколико систем не може да учита клијента он приказује администратору поруку “Систем не може да учита одабраног клијента”. Прекида се извршење сценарија. (ИА)

13.1 Уколико систем не може да измени податке о клијенту он приказује администратору поруку “Систем не може да измени клијента. Покушајте поново”. (ИА)



### СК5: Случај коришћења – Унос нове услуге

**Назив СК**

Унос нове услуге

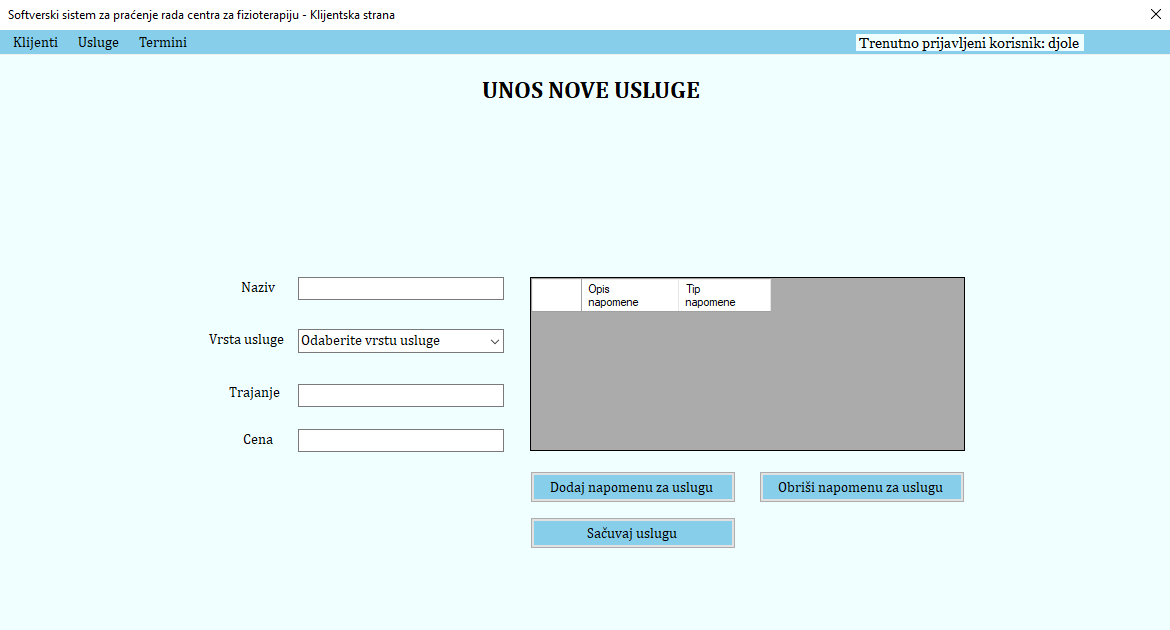
**Актори СК**

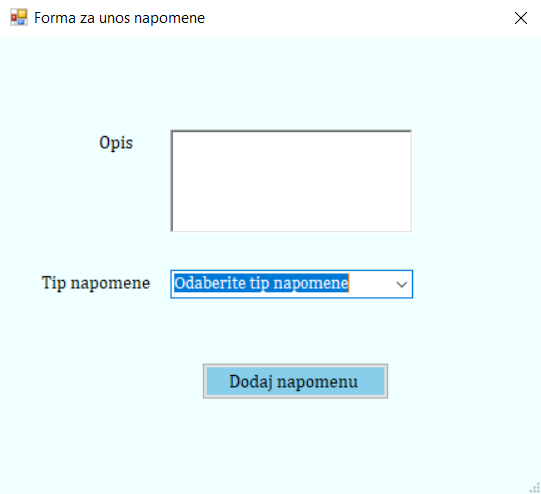
Администратор

**Учесници СК**

Администратор и систем (програм)

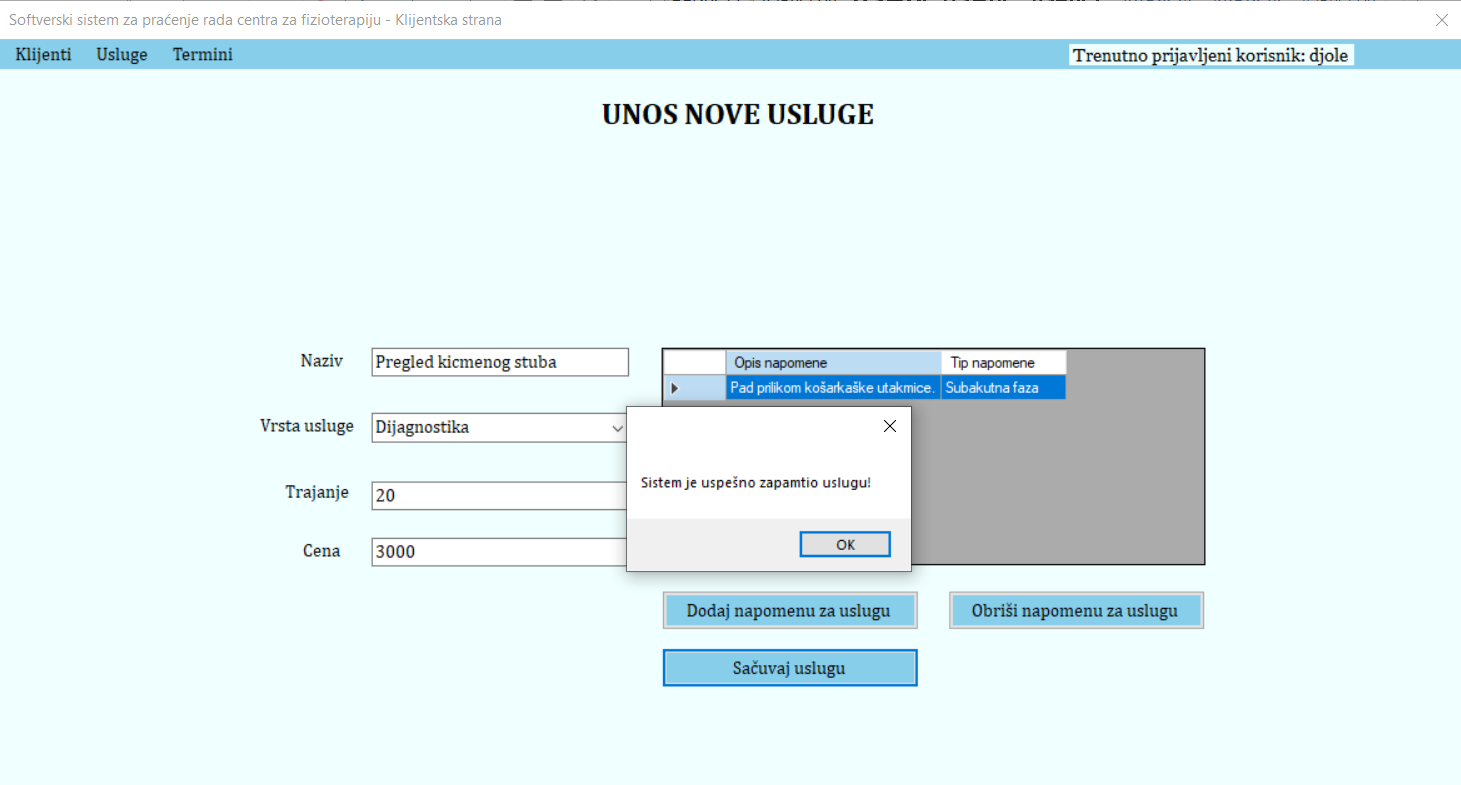
**Предуслов**: Систем је укључен и администратор je улогован под својом шифром. Систем приказује форму за унос нове услуге. Учитана је листа врста услуга и листа типова напомена за услугу.





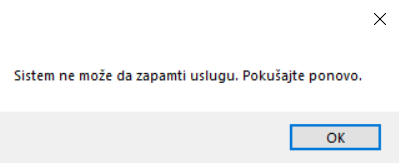
**Основни сценарио СК**

1. Администратор **уноси** податке о новој услузи у форму. (АПУСО)
2. Администратор **проверава** да ли је коректно унео податке о новој услузи у форму. (АНСО)
3. Администратор **позива** систем да запамти податке о новој услузи. (АПСО)
4. Систем **памти** податке о новој услузи. (СО)
5. Систем **приказује** администратору поруку: “Систем је успешно запамтио услугу“. (ИА)



Алтернативна сценарија

5.1 Уколико систем не може да запамти податке о новој услузи он приказује администратору поруку “Систем не може да запамти услугу. Покушајте поново”. (ИА)



### СК6: Случај коришћења – Брисање услуге

**Назив СК**

Брисање услуге

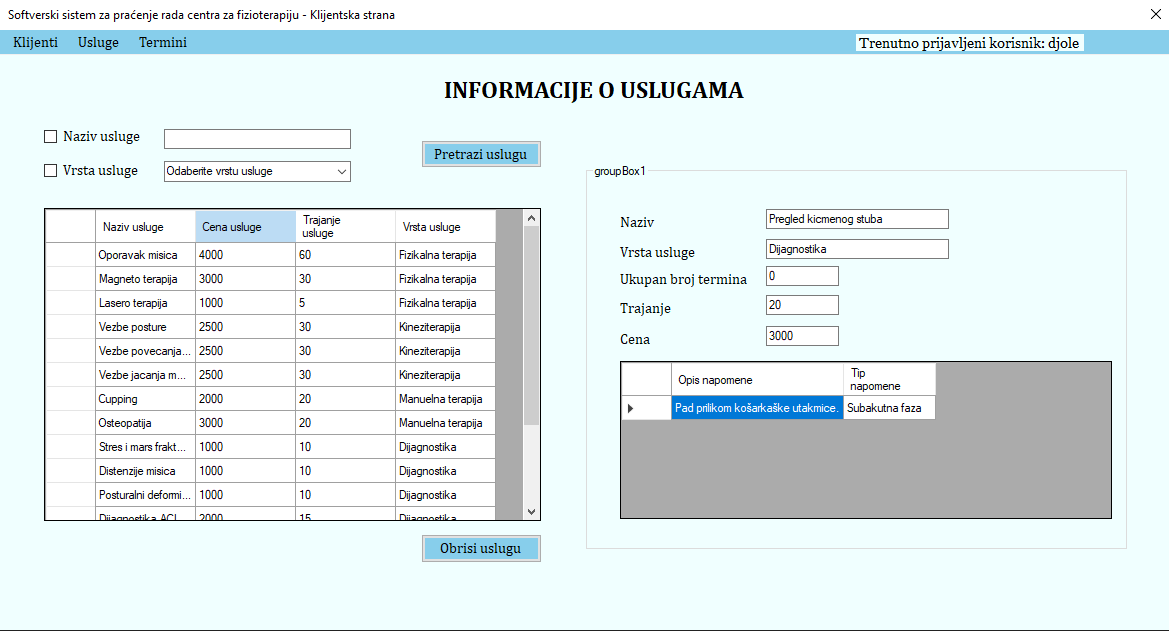
**Актори СК**

Администратор

**Учесници СК**

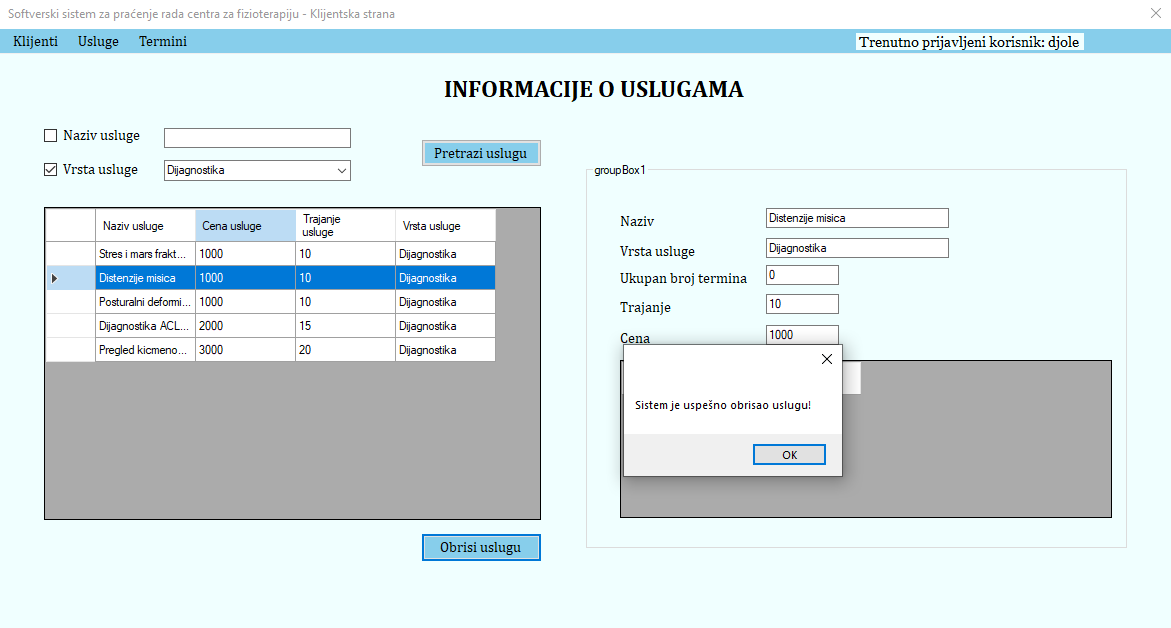
Администратор и систем (програм)

**Предуслов**: Систем је укључен и администратор je улогован под својом шифром. Систем приказује форму за рад са услугама. Учитана је листа услуга.



**Основни сценарио СК**

1. Администратор **уноси** критеријумпо ком претражује услуге. (АПУСО)
2. Администратор **проверава** да ли је коректно унео критеријум. (АНСО)
3. Администратор **позива** систем да нађе услуге по задатом критеријуму. (АПСО)
4. Систем **тражи** услуге по задатом критеријуму. (СО)
5. Систем **приказује** администратору услуге. (ИА)
6. Администратор **бира** услугу коју жели да обрише. (АПУСО)
7. Администратор **позива** систем да учита податке о одабраној услузи. (АПСО)
8. Систем **учитава** податке о одабраној услузи. (СО)
9. Систем **приказује** администратору одабрану услугу. (ИА)
10. Администратор **позива** систем да обрише услугу. (АПСО)
11. Систем **брише** услугу. (СО)
12. Систем **приказује** администратору поруку: “Систем је успешно обрисао услугу.” (ИА)

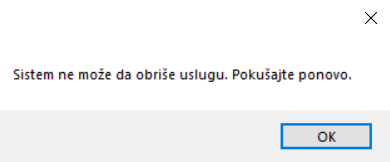


Алтернативна сценарија

5.1 Уколико систем не може да нађе услуге он приказује администратору поруку: “Систем не може да нађе услуге по задатом критеријуму”. Прекида се извршење сценарија. (ИА)

9.1 Уколико систем не може да учита услугу он приказује администратору поруку “Систем не може да учита услугу ”. Прекида се извршење сценарија. (ИА)

12.1 Уколико систем не може да обрише услугу он приказује администратору поруку “Систем не може да обрише услугу. Покушајте поново”. (ИА)



### СК7: Случај коришћења – Унос заказаних термина

**Назив СК**

Унос заказаних термина

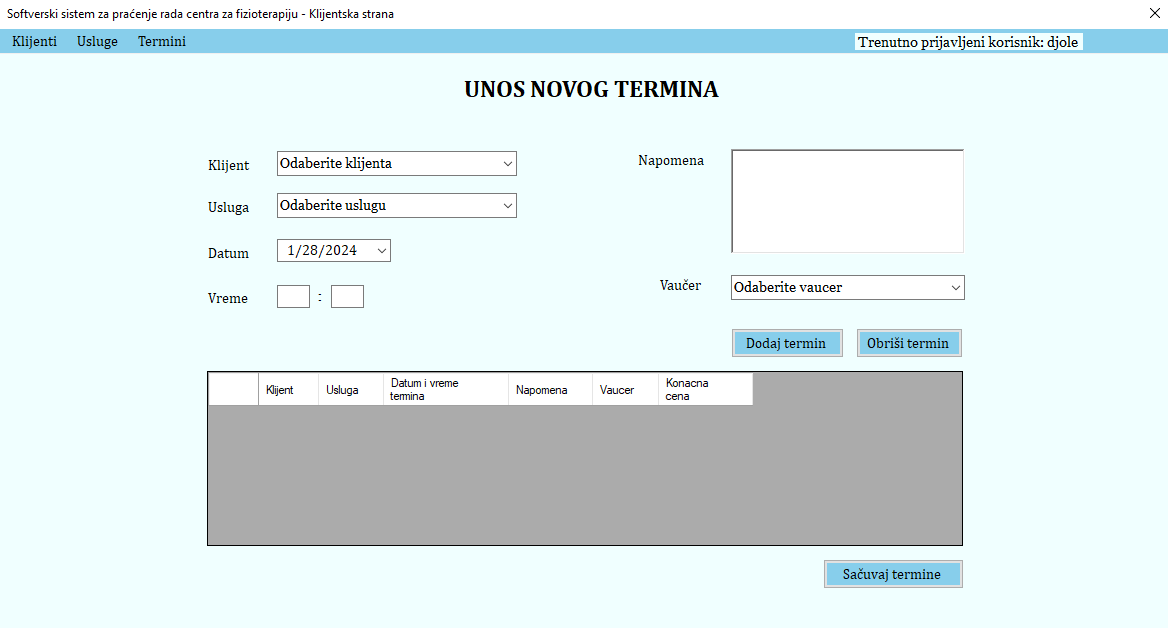
**Актори СК**

Администратор

**Учесници СК**

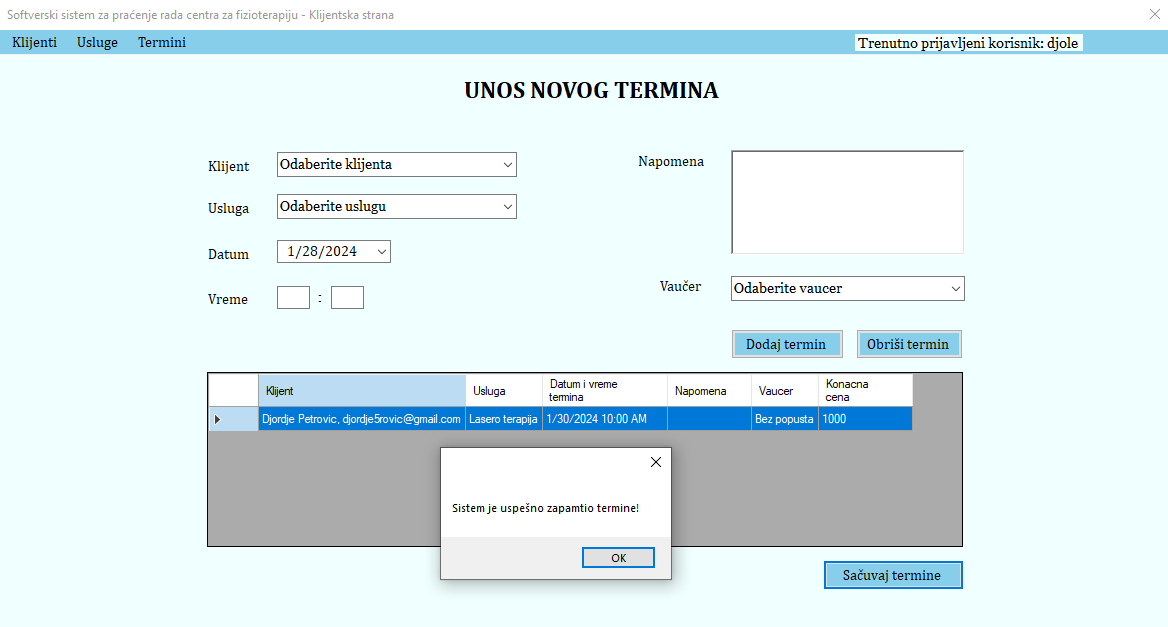
Администратор и систем (програм)

**Предуслов**: Систем је укључен и администратор je улогован под својом шифром. Систем приказује форму за унос заказаног термина. Учитана је листа услуга и листа клијената.



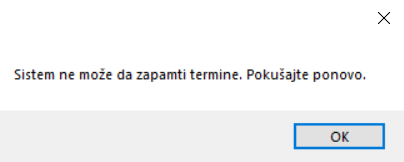
**Основни сценарио СК**

1. Администратор **уноси** податке о заказаним терминима. (АПУСО)
2. Администратор **проверава** да ли је коректно унео податке о заказаним терминима. (АНСО)
3. Администратор **позива** систем да запамти податке о заказаним терминима. (АПСО)
4. Систем **памти** податке заказаним терминима. (СО)
5. Систем **приказује** администратору запамћене термине и поруку: “Систем је успешно запамтио термине“. (ИА)



Алтернативна сценарија

5.1 Уколико систем не може да запамти податке о заказаним терминима он приказује администратору поруку “Систем не може да запамти термине. Покушајте поново”. (ИА)



### СК8: Случај коришћења – Брисање заказаног термина

**Назив СК**

Брисање заказаног термина

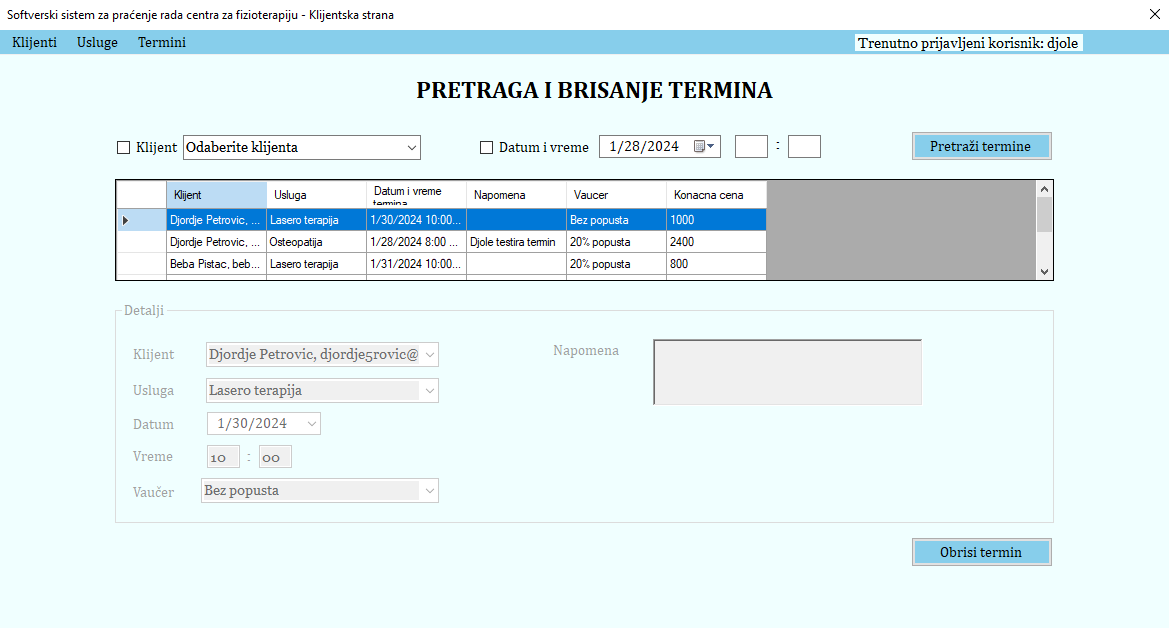
**Актори СК**

Администратор

**Учесници СК**

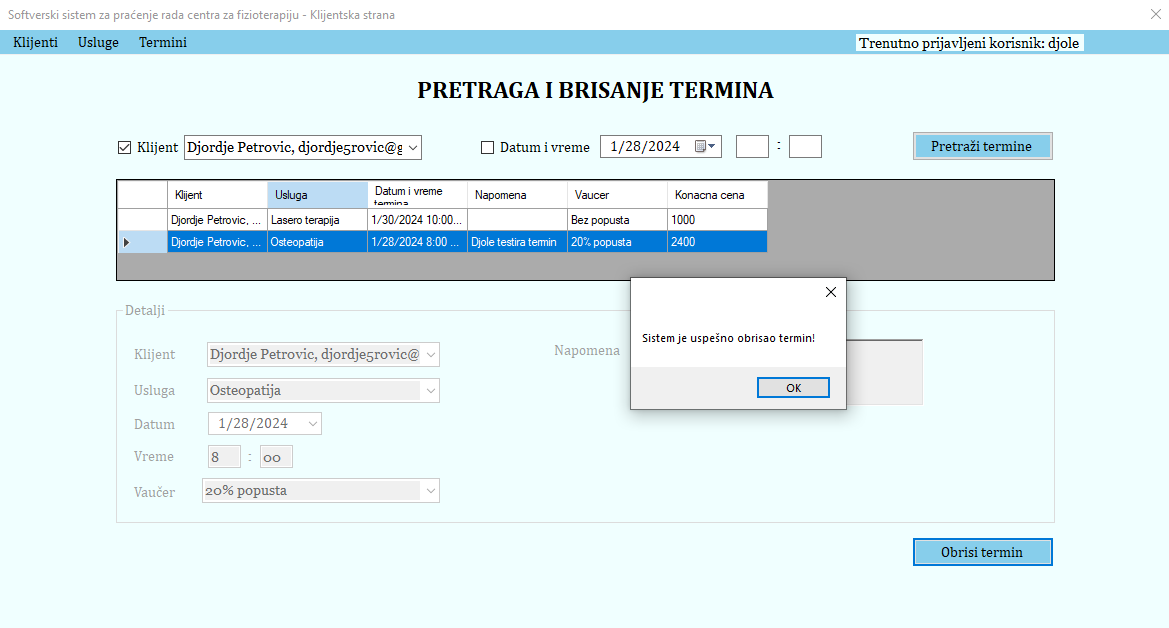
Администратор и систем (програм)

**Предуслов**: Систем је укључен и администратор je улогован под својом шифром. Систем приказује форму за рад са заказаним терминима. Учитана је листа заказаних термина.



**Основни сценарио СК**

1. Администратор **уноси** критеријумпо ком претражује заказане термине. (АПУСО)
2. Администратор **проверава** да ли је коректно унео критеријум. (АНСО)
3. Администратор **позива** систем да нађе заказане термине по задатом критеријуму. (АПСО)
4. Систем **тражи** заказане термине по задатом критеријуму. (СО)
5. Систем **приказује** администратору заказане термине. (ИА)
6. Администратор **бира** заказани термин који жели да обрише. (АПУСО)
7. Администратор **позива** систем да учита податке о одабраном термину. (АПСО)
8. Систем **учитава** податке о одабраном термину. (СО)
9. Систем **приказује** администратору податке о одабраном термину. (ИА)
10. Администратор **позива** систем да обрише одабрани термин. (АПСО)
11. Систем **брише** одабрани термин. (СО)
12. Систем **приказује** администратору поруку: “Систем је успешно обрисао термин.” (ИА)

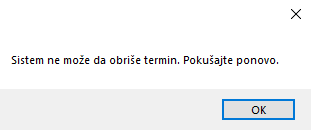


Алтернативна сценарија

5.1 Уколико систем не може да нађе заказане термине он приказује администратору поруку: “Систем не може да нађе заказане термине по задатом критеријуму”. Прекида се извршење сценарија. (ИА)

9.1 Уколико систем не може да учита податке о одабраном термину он приказује администратору поруку “Систем не може да учита одабрани термин ”. Прекида се извршење сценарија. (ИА)

12.1 Уколико систем не може да обрише одабрани термин он приказује администратору поруку “Систем не може да обрише термин. Покушајте поново”. (ИА)



### СК9: Случај коришћења – Претрага заказаних термина

**Назив СК**

Претрага заказаних термина

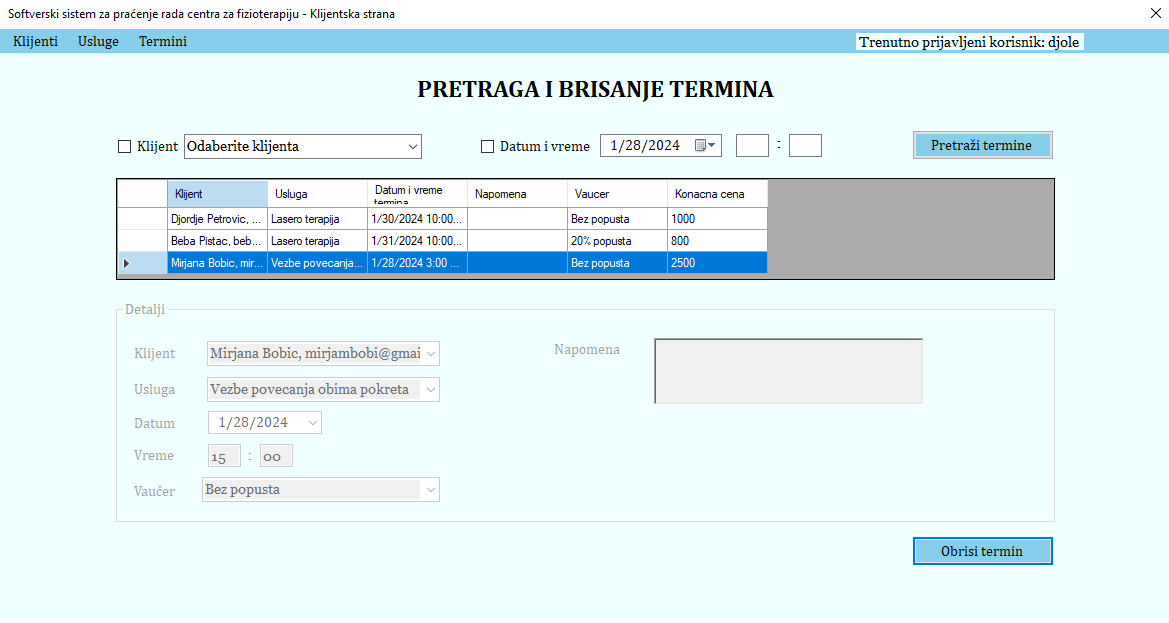
**Актори СК**

Администратор

**Учесници СК**

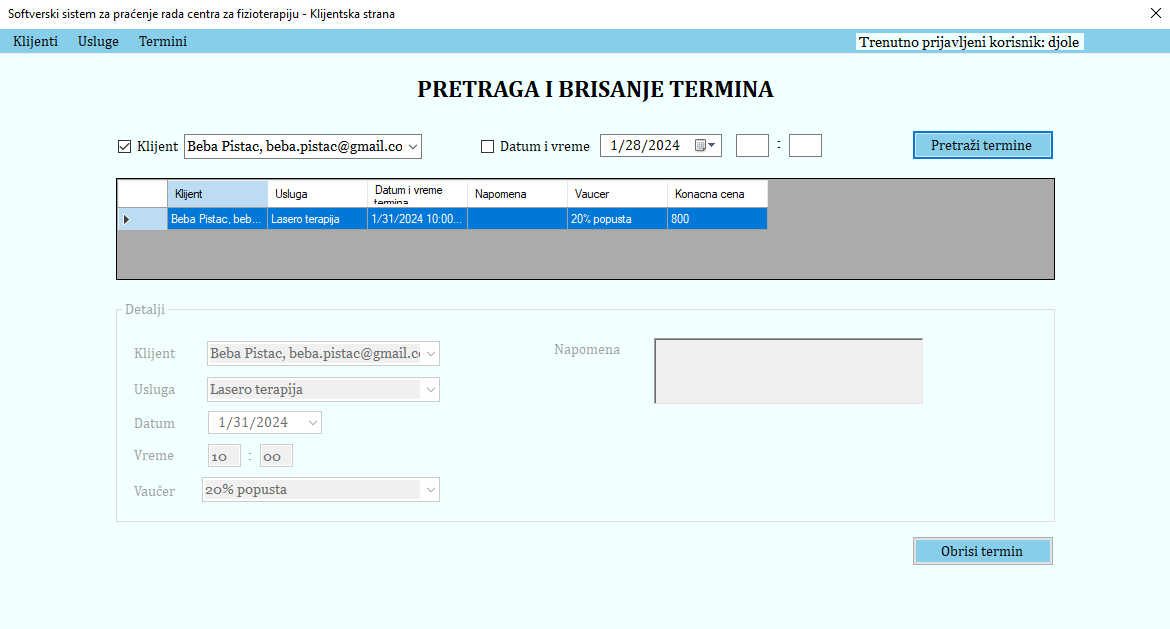
Администратор и систем (програм)

**Предуслов**: Систем је укључен и администратор je улогован под својом шифром. Систем приказује форму за рад са заказаним терминима. Учитана је листа заказаних термина.



**Основни сценарио СК**

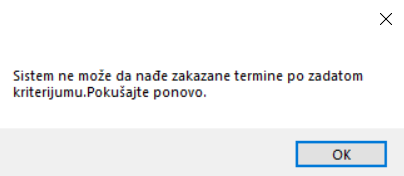
1. Администратор **уноси** критеријумпо ком претражује заказане термине. (АПУСО)
2. Администратор **проверава** да ли је коректно унео критеријум. (АНСО)
3. Администратор **позива** систем да нађе заказане термине по задатом критеријуму. (АПСО)
4. Систем **тражи** заказане термине по задатом критеријуму. (СО)
5. Систем **приказује** администратору заказане термине. (ИА)
6. Администратор **бира** жељени заказани термин. (АПУСО)
7. Администратор **позива** систем да учита податке о одабраном термину. (АПСО)
8. Систем **учитава** податке о одабраном термину. (СО)
9. Систем **приказује** администратору податке о одабраном термину. (ИА)



Алтернативна сценарија

5.1 Уколико систем не може да нађе заказане термине он приказује администратору поруку: “Систем не може да нађе заказане термине по задатом критеријуму”. Прекида се извршење сценарија. (ИА)

9.1 Уколико систем не може да учита податке о одабраном термину он приказује администратору поруку “Систем не може да учита одабрани термин. Покушајте поново ”. (ИА)



## Пројектовање апликационе логике

Апликациони сервери треба да обезбеде сервисе који ће омогућити реализацију апликационе логике софтверског система. Пројектовани апликациони сервер садржи:

* део за комуникацију са клијентима
* контролер апликационе логике
* део који садржи пословну логику
* део за комуникацију са складиштем података

### Комуникација са клијентима

Део за комуникацију, подиже серверски сокет који служи за ослушкивање мреже. Када клијент успостави конекцију, сервер генерише нит која ће бити одговорна за двосмерну везу са клијентом, ClientHandler. Клијент шаље захтев за извршење неке од системских операција, одговарајућа нит (додељена клијенту) прихвата захтев, позива класу Controller, тј. контролер апликационе логике. Након извршења системске операције, контролер враћа резултат извршења операције на исти начин до нити која је задужена за тог клијента. Резултат се затим прослеђује клијенту. Комуникација између клијента и сервера се обавља разменом објекта класе Request и класа Response :

public enum Operation

{

Login,

End,

IzmeniKlijenta,

ObrisiKlijenta,

ObrisiTermin,

ObrisiUslugu,

PretraziKlijente,

PretraziTermine,

PretraziUsluge,

UcitajKlijenta,

UcitajTermin,

UcitajUslugu,

VratiListuKlijenata,

VratiListuTermina,

VratiListuTipovaKlijenata,

VratiListuTipovaNapomenaZaUslugu,

VratiListuUsluga,

VratiListuVrstaUsluga,

ZapamtiKlijenta,

ZapamtiTermine,

ZapamtiUslugu,

Logout,

ProveriUlogovan,

}

[Serializable]

public class Request

{

public Operation Operation { get; set; }

public object RequestObject { get; set; } = null;

}

[Serializable]

public class Response

{

public string Message { get; set; }

public bool IsSuccessful { get; set; } = true;

public object Result { get; set; } = null;

}

### Контролер апликационе логике

Контролер апликационе логике прихвата захтеве за извршење системских операција и исте прослеђује до конкретне системске операције. Након извршења системске операције, контролер прихвата одговор и враћа назад до нити за обраду захтева. За сваку системску операцију праве се софтверске класе које треба да реализују системску операцију. 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. Пример специфичног контролера апликационе логике:

public class Controller

{

#region Singleton

private static Controller instance;

private Controller()

{

}

public static Controller Instance

{

get

{

if (instance == null)

{

instance = new Controller();

}

return instance;

}

}

#endregion

public void IzmeniKlijenta(Klijent klijent)

{

IzmeniKlijentaSO so = new IzmeniKlijentaSO();

so.ExecuteTemplate(klijent);

}

public void ObrisiKlijenta(Klijent klijent)

{

ObrisiKlijentaSO so = new ObrisiKlijentaSO();

so.ExecuteTemplate(klijent);

}

public void ObrisiTermin(Termin termin)

{

ObrisiTerminSO so = new ObrisiTerminSO();

so.ExecuteTemplate(termin);

}

public void ObrisiUslugu(Usluga usluga)

{

ObrisiUsluguSO so = new ObrisiUsluguSO();

so.ExecuteTemplate(usluga);

}

public List<Klijent> PretraziKlijente(Klijent kriterijum)

{

PretraziKlijenteSO so = new PretraziKlijenteSO();

so.ExecuteTemplate(kriterijum);

return so.Result;

}

public List<Termin> PretraziTermine(Termin kriterijum)

{

PretraziTermineSO so = new PretraziTermineSO();

so.ExecuteTemplate(kriterijum);

return so.Result;

}

public List<Usluga> PretraziUsluge(Usluga kriterijum)

{

PretraziUslugeSO so = new PretraziUslugeSO();

so.ExecuteTemplate(kriterijum);

return so.Result;

}

public Korisnik PrijaviAdministratora(Korisnik korisnik)

{

PrijaviAdministratoraSO so = new PrijaviAdministratoraSO();

so.ExecuteTemplate(korisnik);

return so.Result;

}

public Klijent UcitajKlijenta(Klijent klijent)

{

UcitajKlijentaSO so = new UcitajKlijentaSO();

so.ExecuteTemplate(klijent);

return so.Result;

}

public Termin UcitajTermin(Termin termin)

{

UcitajTerminSO so = new UcitajTerminSO();

so.ExecuteTemplate(termin);

return so.Result;

}

public Usluga UcitajUslugu(Usluga usluga)

{

UcitajUsluguSO so = new UcitajUsluguSO();

so.ExecuteTemplate(usluga);

return so.Result;

}

public List<Klijent> VratiListuKlijenata()

{

VratiListuKlijenataSO so = new VratiListuKlijenataSO();

so.ExecuteTemplate(new Klijent());

return so.Result;

}

public List<Termin> VratiListuTermina()

{

VratiListuTerminaSO so = new VratiListuTerminaSO();

so.ExecuteTemplate(new Termin());

return so.Result;

}

public List<TipKlijenta> VratiListuTipovaKlijenata()

{

VratiListuTipovaKlijenataSO so = new VratiListuTipovaKlijenataSO();

so.ExecuteTemplate(new TipKlijenta());

return so.Result;

}

public List<TipNapomeneZaUslugu> VratiListuTipovaNapomenaZaUslugu()

{

VratiListuTipovaNapomenaZaUsluguSO so = new VratiListuTipovaNapomenaZaUsluguSO();

so.ExecuteTemplate(new TipNapomeneZaUslugu());

return so.Result;

}

public List<Usluga> VratiListuUsluga()

{

VratiListuUslugaSO so = new VratiListuUslugaSO();

so.ExecuteTemplate(new Usluga());

return so.Result;

}

public List<VrstaUsluge> VratiListuVrstaUsluga()

{

VratiListuVrstaUslugaSO so = new VratiListuVrstaUslugaSO();

so.ExecuteTemplate(new VrstaUsluge());

return so.Result;

}

public Klijent ZapamtiKlijenta(Klijent klijent)

{

ZapamtiKlijentaSO so = new ZapamtiKlijentaSO();

so.ExecuteTemplate(klijent);

return so.Result;

}

public List<Termin> ZapamtiTermine(List<Termin> termini)

{

ZapamtiTerminеSO so = new ZapamtiTerminеSO

{

Termini = termini

};

so.ExecuteTemplate(new Termin());

return so.Result;

}

public Usluga ZapamtiUslugu(Usluga usluga)

{

ZapamtiUsluguSO so = new ZapamtiUsluguSO();

so.ExecuteTemplate(usluga);

return so.Result;

}

}

### Пословна логика

#### Пројектовање понашања софтверског система (системске операције)

У фази анализе, дефинисали смо уговоре о системским операцијама. Сваки од уговора описивао је понашање једне конкретне системске операције, али не и начин на који ћемо доћи до њега. Свака од системских операција, треба да наследи класу SystemOperationBase како би могла да се повеже са базом и како би се њено извршавање пратило као трансакција. Метода која обезбеђује наведене захтеве је метода ExecuteTemplate (IDomainObject domainObject), која даје кораке алгоритма извршавања сваке од системских операција.

public abstract class SystemOperationBase

{

protected IRepository repository = new GenericDbRepository();

public void ExecuteTemplate(IDomainObject domainObject)

{

try

{

repository.OpenConnection();

repository.BeginTransaction();

Execute(domainObject);

repository.Commit();

}

catch (Exception )

{

repository.Rollback();

throw;

}

finally

{

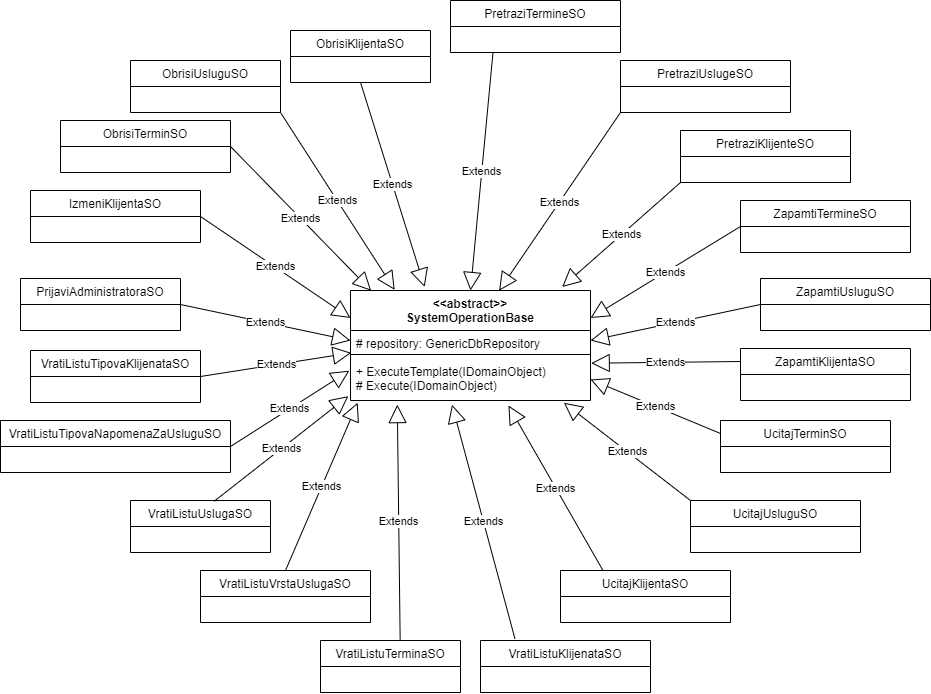
repository.CloseConnection();

}

}

protected abstract void Execute(IDomainObject domainObject);

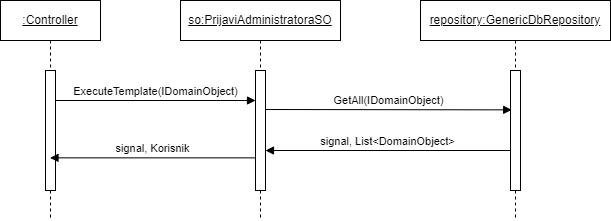
}



*Понашање* *система*

### Уговор УГ1: Пријави администратора

**Операција:** *prijaviAdministratora(Administrator):* signal,  
**Веза са СК:** СК1  
**Предуслови:** Вредносна и структурна ограничења над објектом Администратор морају бити задовољена.  
**Постуслови**: Запамћен је нови администратор.



*Дијаграм секвенци системске операције PrijaviAdministratoraSO*

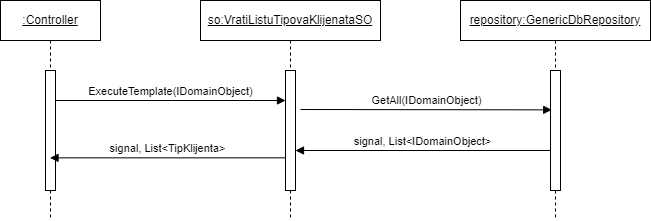
##### Уговор УГ2: Врати листу типова клијената

**Операција:** *vratiListuTipovaKlijenata(List<TipKlijenta>):* signal,

**Веза са СК:** СК2, СК4

**Предуслови:** /

**Постуслови:** /



*Дијаграм секвенци системске операције VratiListuTipovaKlijenataSO*

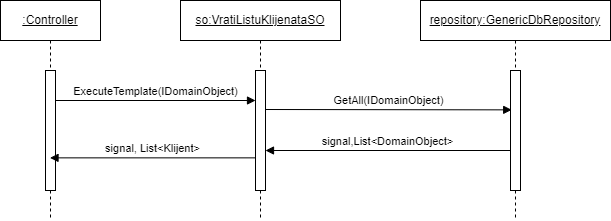
##### Уговор УГ3: Врати листу клијената

**Операција:** *vratiListuKlijenata(List<Klijent>):* signal,

**Веза са СК:** СК3, СК4, СК7

**Предуслови:** /

**Постуслови:** /



*Дијаграм секвенци системске операције VratiListuKlijenataSO*

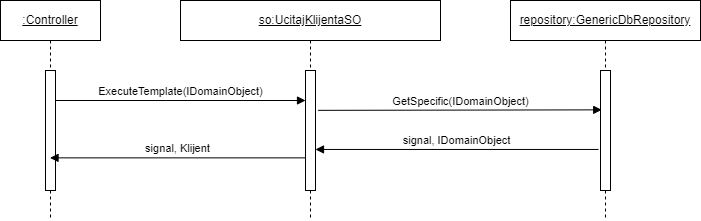
##### Уговор УГ4: Учитај клијента

**Операција:** *ucitajKlijenta(Klijent):* signal,

**Веза са СК:** СК3, СК4

**Предуслови:** /

**Постуслови:** /



*Дијаграм секвенци системске операције UcitajKlijentaSO*

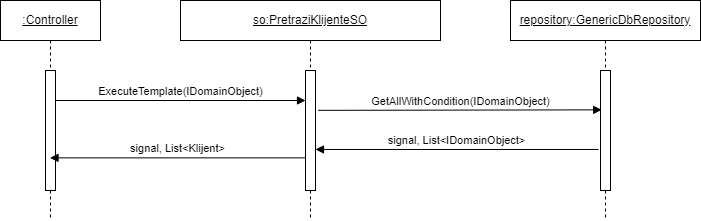
##### Уговор УГ5: Претражи клијенте

**Операција:** *pretraziKlijente(kriterijum, List<Klijent>):* signal,

**Веза са СК:** СК3, СК4

**Предуслови:** /

**Постуслови:** /



*Дијаграм секвенци системске операције PretraziKlijenteSO*

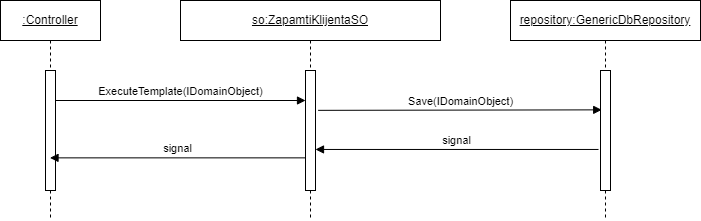
##### Уговор УГ6: Запамти клијента

**Операција:** *zapamtiKlijenta(Klijent):* signal,

**Веза са СК:** СК2, СК4

**Предуслови:** Вредносна и структурна ограничења над објектом Клијент морају бити задовољена.

**Постуслови:** Запамћен је нови клијент.



*Дијаграм секвенци системске операције ZapamtiKlijentaSO*

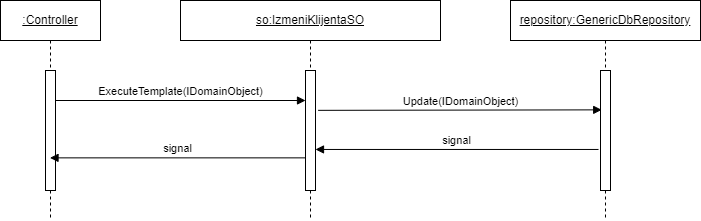
##### Уговор УГ7: Измени клијента

**Операција:** *izmeniKlijenta(Klijent):* signal,

**Веза са СК:** СК4

**Предуслови:** Вредносна и структурна ограничења над објектом Клијент морају бити задовољена.

**Постуслови:** Клијент је измењен.



*Дијаграм секвенци системске операције IzmeniKlijentaSO*

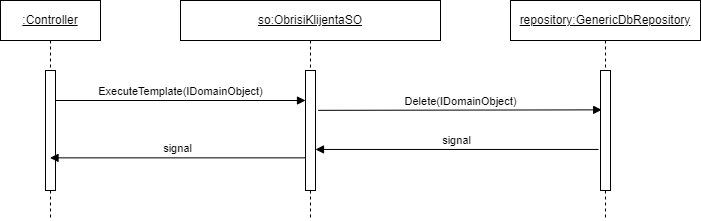
##### Уговор УГ8: Обриши клијента

**Операција:** *obrisiKlijenta(Klijent):* signal,

**Веза са СК:** СК4

**Предуслови:** Структурна ограничења над објектом Клијент морају бити задовољена.

**Постуслови:** Клијент је обрисан.



*Дијаграм секвенци системске операције ObrisiKlijentaSO*

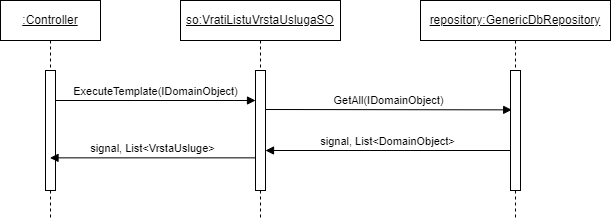
##### Уговор УГ9: Врати листу врста услуга

**Операција:** *vratiListuVrstaUsluga(List<VrstaUsluge>):* signal,

**Веза са СК:** СК5

**Предуслови:** /

**Постуслови:** /



*Дијаграм секвенци системске операције VratiListuVrstaUslugaSo*

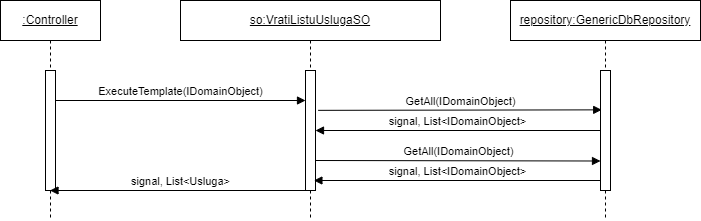
##### Уговор УГ10: Врати листу услуга

**Операција:** *vratiListuUsluga(List<Usluga>):* signal,

**Веза са СК:** СК6, СК7

**Предуслови:** /

**Постуслови:** /



*Дијаграм секвенци системске операције VratiListuUslugaSO*

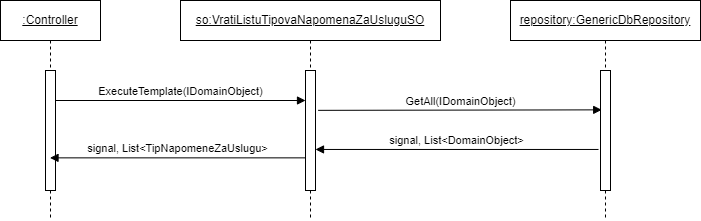
##### Уговор УГ11: Врати листу типова напомена за услугу

**Операција:** *vratiListuTipovaNapomenaZaUslugu(List<TipNapomeneZaUslugu>):* signal,

**Веза са СК:** СК5

**Предуслови:** /

**Постуслови:** /



*Дијаграм секвенци системске операције VratiListuTipovaNappomenaZaUsluguSO*

##### 

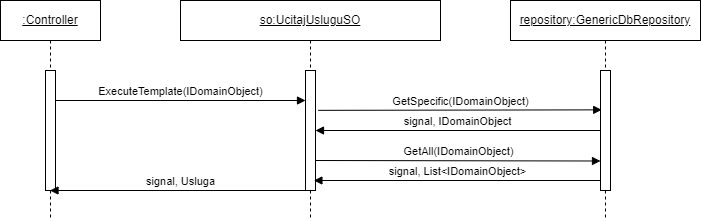
##### Уговор УГ12: Учитај услугу

**Операција:** *ucitajUslugu(Usluga):* signal,

**Веза са СК:** СК7

**Предуслови:** /

**Постуслови:** /



*Дијаграм секвенци системске операције UcitajUsluguSO*

##### 

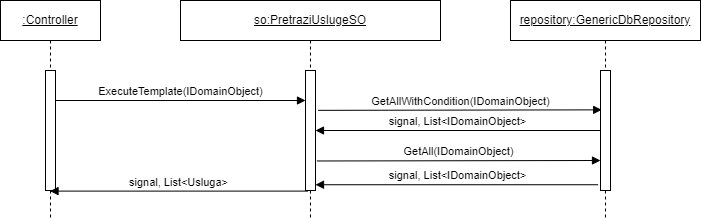
##### Уговор УГ13: Претражи услуге

**Операција:** *pretraziUsluge(kriterijum, List<Usluga>):* signal,

**Веза са СК:** СК6

**Предуслови:** /

**Постуслови:** /



*Дијаграм секвенци системске операције PretraziUslugeSO*

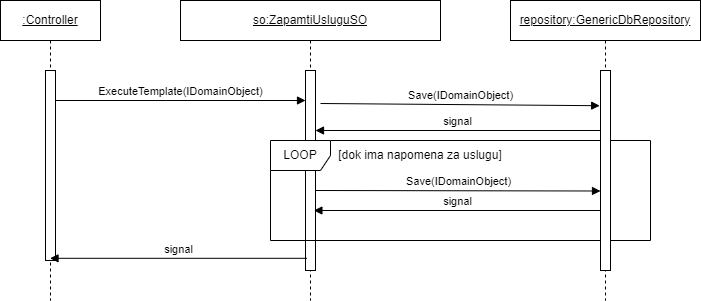
##### Уговор УГ14: Запамти услугу

**Операција:** *zapamtiUslugu(Usluga):* signal,

**Веза са СК:** СК5

**Предуслови:** Вредносна и структурна ограничења над објектом Услуга морају бити задовољена.

**Постуслови:** Запамћена је нова услуга.



*Дијаграм секвенци системске операције ZapamtiUsluguSO*

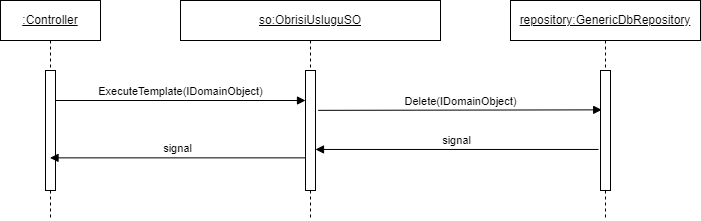
##### Уговор УГ15: Обриши услугу

**Операција:** *obrisiUslugu(Usluga):* signal,

**Веза са СК:** СК7

**Предуслови:** Структурна ограничења над објектом Услуга морају бити задовољена.

**Постуслови:** Услуга је обрисана.



*Дијаграм секвенци системске операције ObrisiUsluguSO*

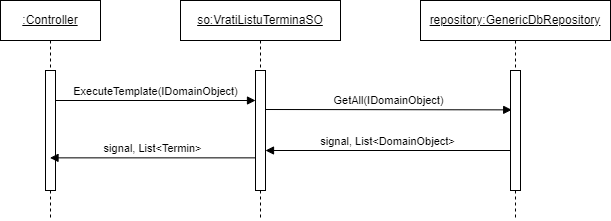
##### Уговор УГ16: Врати листу термина

**Операција:** *vratiListuTermina(List<Termin>):* signal,

**Веза са СК:** СК8, СК9

**Предуслови:** /

**Постуслови:** /



*Дијаграм секвенци системске операције VratiListuTerminaSO*

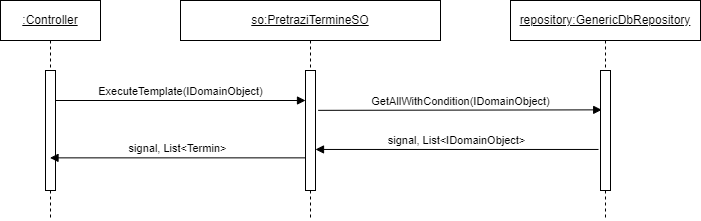
##### Уговор УГ17: Претражи термине

**Операција:** *pretraziTermine(kriterijum, List<Termin>):* signal,

**Веза са СК:** СК8, СК9

**Предуслови:** /

**Постуслови:** /



*Дијаграм секвенци системске операције PretraziTermineSO*

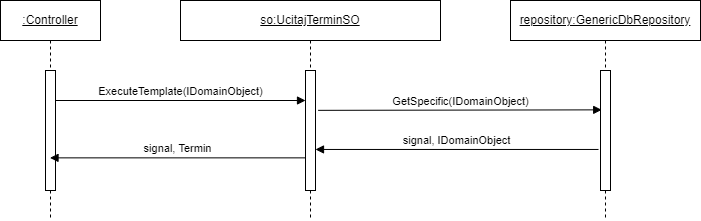
##### Уговор УГ18: Учитај термин

**Операција:** *ucitajTermin(Termin):* signal,

**Веза са СК:** СК8, СК9

**Предуслови:** /

**Постуслови:** /



*Дијаграм секвенци системске операцијe UcitajTerminSO*

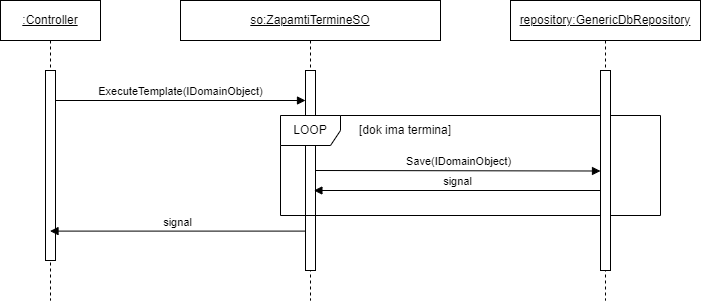
##### Уговор УГ19: Запамти термине

**Операција:** *zapamtiTerminе(List< Termin>):* signal,

**Веза са СК:** СК7

**Предуслови:** Вредносна и структурна ограничења над објектима Термин морају бити задовољена.

**Постуслови:** Запамћени су нови термини.



*Дијаграм секвенци системске операције ZapamtiTermineSО*

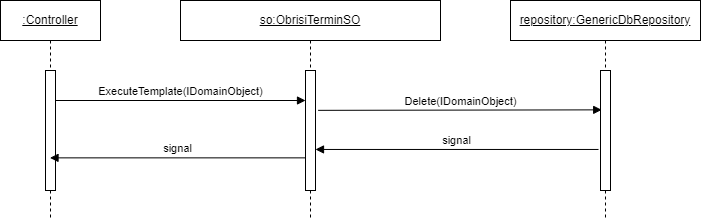
##### Уговор УГ20: Обриши термин

**Операција:** *obrisiTermin(Termin):* signal,

**Веза са СК:** СК8

**Предуслови:** Структурна ограничења над објектом Термин морају бити задовољена.

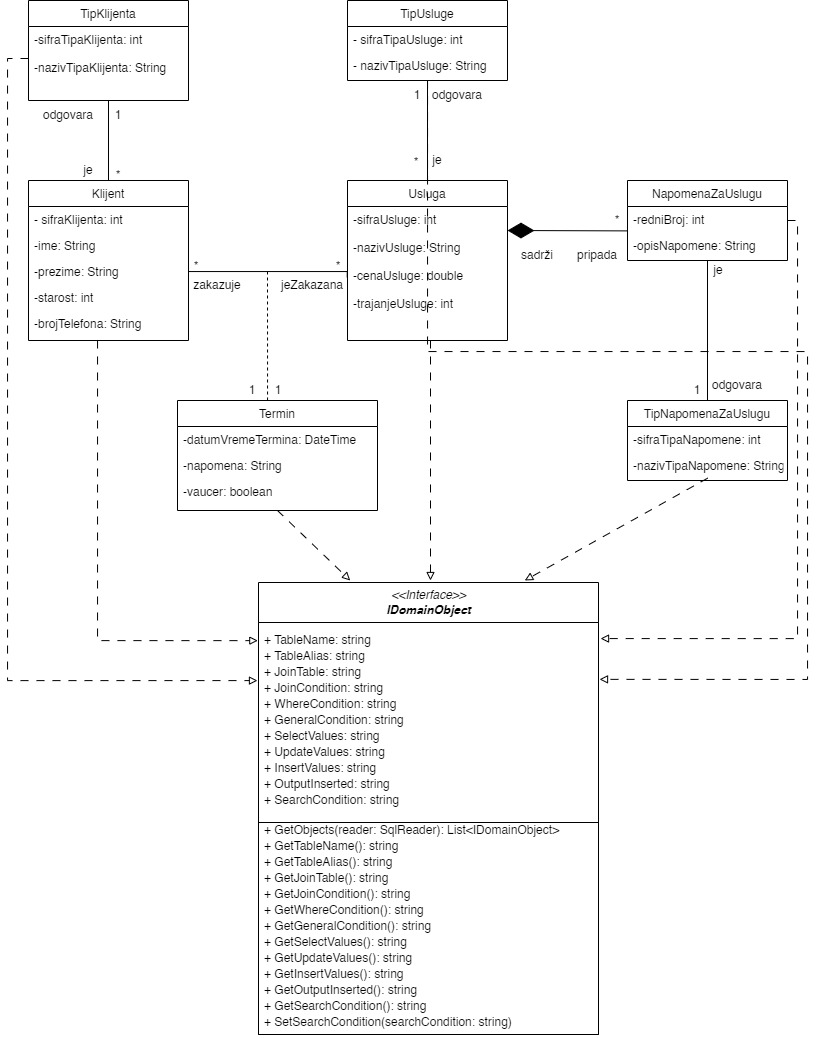
**Постуслови:** Термин је обрисан.



*Дијаграм секвенци системске операције ObrisiTerminSO*

#### Пројектовање структуре софтверског система

Пројектовање структуре описује пројектовање доменских класа са својим атрибутима. Свака доменска класа наслеђује интерфејс IDomainObject који у себи има низ атрибута и методу које свака од њих имплементира. Ти атрибути се користе за креирање генеричких упита над базом података.



*Структура софтверског система*

Пример имплементације интерфејса IDomainObject од стране доменске класе Usluga дат је испод:

[Serializable]

public class Usluga : IDomainObject

{

[Browsable(false)]

public int SifraUsluge { get; set; }

public string NazivUsluge { get; set; }

public double CenaUsluge { get; set; }

public int TrajanjeUsluge { get; set; }

public VrstaUsluge SifraVrsteUsluge { get; set; }

[Browsable(false)]

public List<NapomenaZaUslugu> NapomeneZaUslugu { get; set; }

[Browsable(false)]

public string TableName => "Usluga";

[Browsable(false)]

public string TableAlias => "u";

[Browsable(false)]

public string JoinTable => "join VrstaUsluge vu";

[Browsable(false)]

public string JoinCondition => "on (u.SifraVrsteUsluge=vu.SifraVrsteUsluge)";

[Browsable(false)]

public string WhereCondition => $"SifraUsluge={SifraUsluge}";

[Browsable(false)]

public string GeneralCondition => $"u.SifraUsluge={SifraUsluge}";

[Browsable(false)]

public object SelectValues => "\*";

[Browsable(false)]

public string UpdateValues => "";

[Browsable(false)]

public string InsertValues => $"'{NazivUsluge}', {CenaUsluge}, {TrajanjeUsluge}, {SifraVrsteUsluge.SifraVrsteUsluge}";

[Browsable(false)]

public string OutputInserted => "";

[Browsable(false)]

public string SearchCondition { get => searchCondition; set => searchCondition = value; }

private string searchCondition;

public override string ToString()

{

return NazivUsluge;

}

public List<IDomainObject> GetObjects(SqlDataReader reader)

{

List<IDomainObject> result = new List<IDomainObject>();

while (reader.Read())

{

result.Add(new Usluga

{

SifraUsluge = (int)reader["SifraUsluge"],

NazivUsluge = (string)reader["NazivUsluge"],

CenaUsluge = (double)reader["CenaUsluge"],

TrajanjeUsluge = (int)reader["TrajanjeUsluge"],

SifraVrsteUsluge = new VrstaUsluge

{

SifraVrsteUsluge = (int)reader["SifraVrsteUsluge"],

NazivVrsteUsluge = (string)reader["NazivVrsteUsluge"]

}

});

}

return result;

}

}

### Комуникација са складиштем података

Комуникација са складиштем података је остварена помоћу класа GenericDbRepository и Broker. Класа Broker је одговорна за чување везе са базом података, извршавање трансакција, креирање SQL команди и, као таква, одвојена је од пословне логике. GenericDbRepository представља слој између пословне логике (системских операција) и класе Brokera. Он прима позиве од системских операција и позива одговарајуће методе брокера које се односе на извршење трансакција и креирање SQL команди. На основу системске операције која га је позвала, GenericDbRepository позива одговарајућу сопствену генеричку методу и над SQL командом коју је добио од брокера извршава одговарајућу DML наредбу над базом података. Методе класе GenericDbRepository су пројектоване као генеричке, што значи да могу да прихвате различите доменске објекте преко својих параметара. На овај начин у GenericDbRepository немамо имплементацију појединачних метода за сваку доменску класу. За реализацију ових генеричких метода од кључног значаја је интерфејс IDomainObject, кога имплементирају све доменске класе. На овај начин, класе GenericDbRepository и Broker представљају перзистентни оквир који посредује у свим операцијама над базом података. Поменуте класе су представљене испод:

public class GenericDbRepository : IRepository

{

private readonly Broker broker = new Broker();

public void BeginTransaction()

{

broker.BeginTransaction();

}

public void OpenConnection()

{

broker.OpenConnection();

}

public void CloseConnection()

{

broker.CloseConnection();

}

public void Commit()

{

broker.Commit();

}

public void Rollback()

{

broker.Rollback();

}

public void Delete(IDomainObject domainObject)

{

SqlCommand command = broker.CreateCommand();

command.CommandText = $"delete from {domainObject.TableName} where {domainObject.WhereCondition}";

if (command.ExecuteNonQuery() != 1)

{

throw new Exception("Database error!");

}

}

public List<IDomainObject> GetAll(IDomainObject domainObject)

{

List<IDomainObject> result;

SqlCommand command = broker.CreateCommand();

command.CommandText = $"select {domainObject.SelectValues} from {domainObject.TableName} {domainObject.TableAlias} {domainObject.JoinTable} {domainObject.JoinCondition}";

SqlDataReader reader = command.ExecuteReader();

result = domainObject.GetObjects(reader);

reader.Close();

return result;

}

public List<IDomainObject> GetAllWithCondition(IDomainObject domainObject)

{

List<IDomainObject> result;

SqlCommand command = broker.CreateCommand();

command.CommandText = $"select {domainObject.SelectValues} from {domainObject.TableName} {domainObject.TableAlias} {domainObject.JoinTable} {domainObject.JoinCondition} where {domainObject.SearchCondition}";

SqlDataReader reader = command.ExecuteReader();

result = domainObject.GetObjects(reader);

reader.Close();

return result;

}

public List<IDomainObject> GetSpecific(IDomainObject domainObject)

{

List<IDomainObject> result;

SqlCommand command = broker.CreateCommand();

command.CommandText = $"select {domainObject.SelectValues} from {domainObject.TableName} {domainObject.TableAlias} {domainObject.JoinTable} {domainObject.JoinCondition} where {domainObject.GeneralCondition}";

SqlDataReader reader = command.ExecuteReader();

result = domainObject.GetObjects(reader);

reader.Close();

return result;

}

public void Save(IDomainObject domainObject)

{

SqlCommand command = broker.CreateCommand();

command.CommandText = $"insert into {domainObject.TableName} values ({domainObject.InsertValues})";

if (command.ExecuteNonQuery() != 1)

{

throw new Exception("Database error!");

}

}

public void Update(IDomainObject domainObject)

{

SqlCommand command = broker.CreateCommand();

command.CommandText = $"update {domainObject.TableName} set {domainObject.UpdateValues} where {domainObject.WhereCondition}";

if (command.ExecuteNonQuery() != 1)

{

throw new Exception("Database error!");

}

}

}

public class Broker

{

private SqlConnection connection;

private SqlTransaction transaction;

public Broker()

{

connection = new SqlConnection(@"Data Source=(localdb)\MSSQLLocalDB;Initial Catalog=FizioCentar;Integrated Security=True;Connect Timeout=30;Encrypt=False;TrustServerCertificate=False;ApplicationIntent=ReadWrite;MultiSubnetFailover=False");

}

public void OpenConnection()

{

connection.Open();

}

public void CloseConnection()

{

if (connection != null && connection.State != ConnectionState.Closed)

connection.Close();

}

public SqlCommand CreateCommand()

{

return new SqlCommand("", connection, transaction);

}

public void BeginTransaction()

{

transaction = connection.BeginTransaction();

}

public void Commit()

{

transaction?.Commit();

}

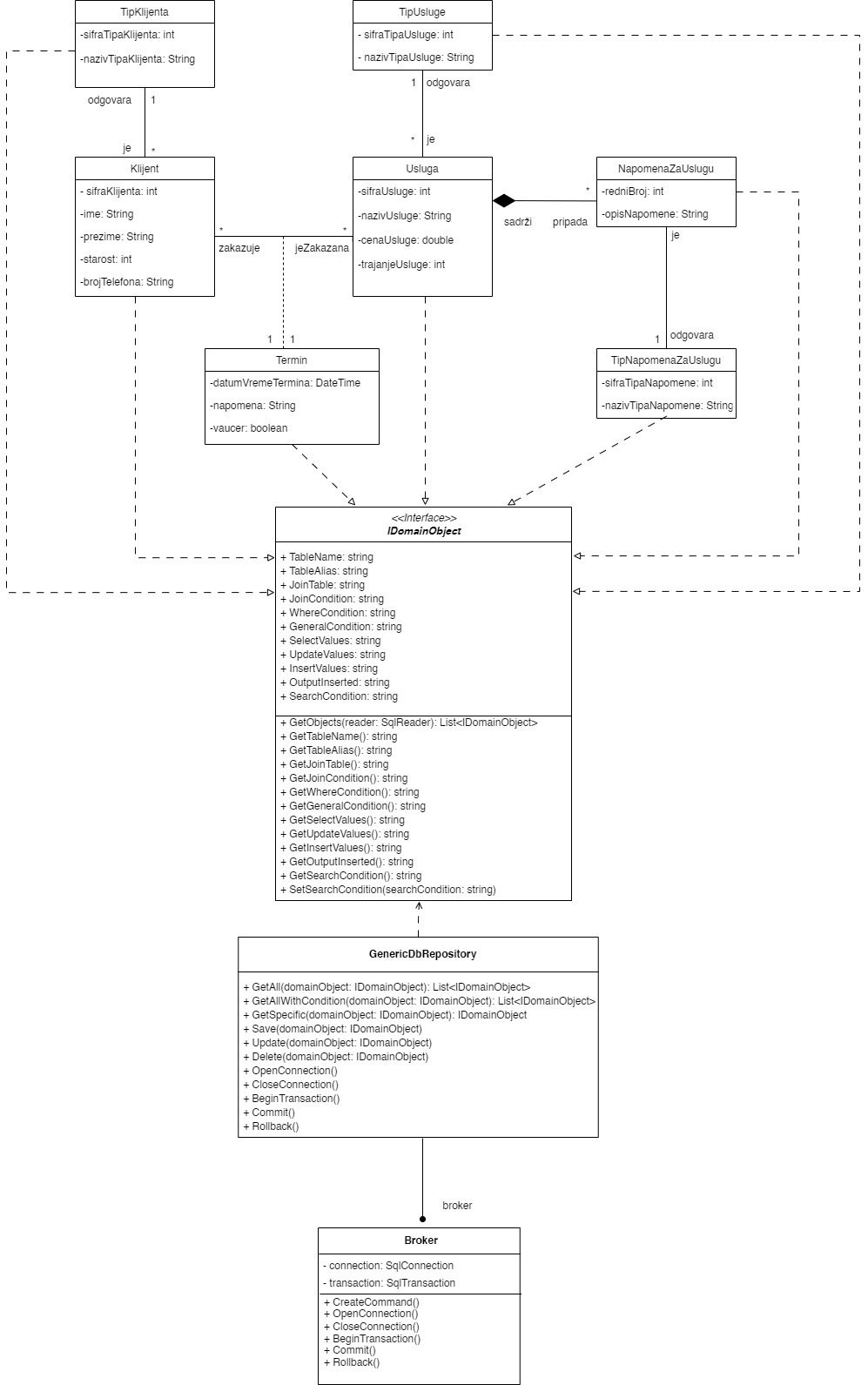
public void Rollback()

{

transaction?.Rollback();

}

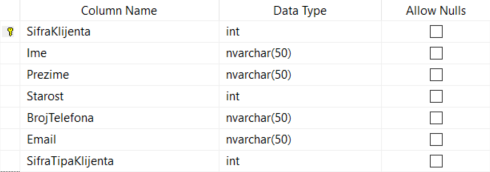
}



*Повезаност класа IDomainObject, GenericDbRepository и Broker*

## Пројектовање складишта података

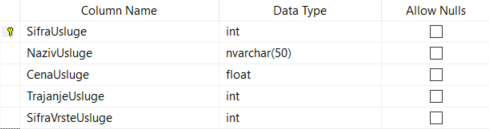
У наставку су приказане табеле у самој бази података. У питању је релациона SQL база података. Табеле су креиране и повезане на основу концептуалног и релационог модела из фазе анализе.



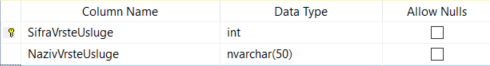
*Структура табеле Клијент*



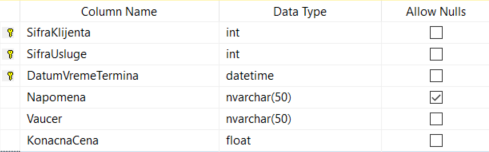
*Структура табеле Тип клијента*



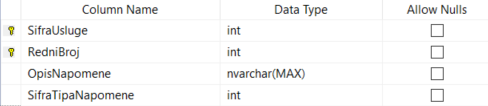
*Структура табеле Услуга*



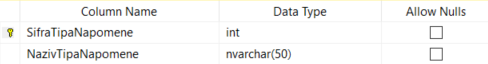
*Структура табеле Врста услуге*



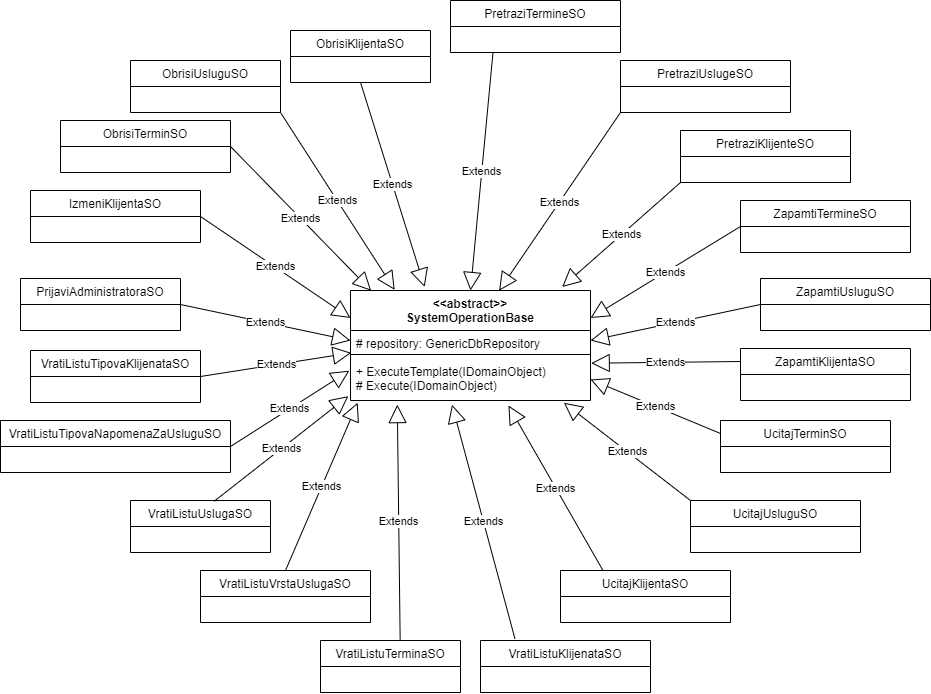
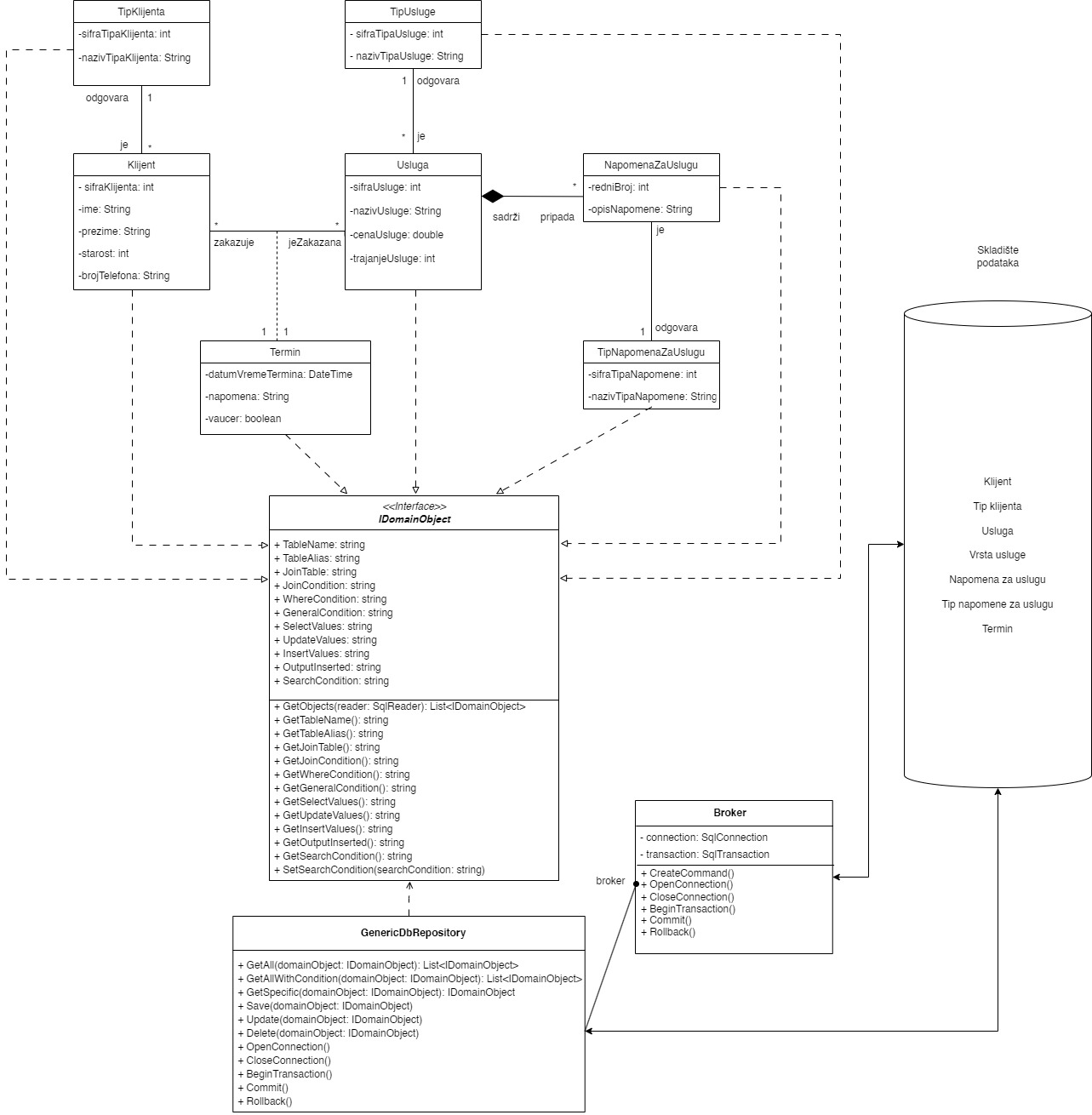
*Структура табеле Термин*

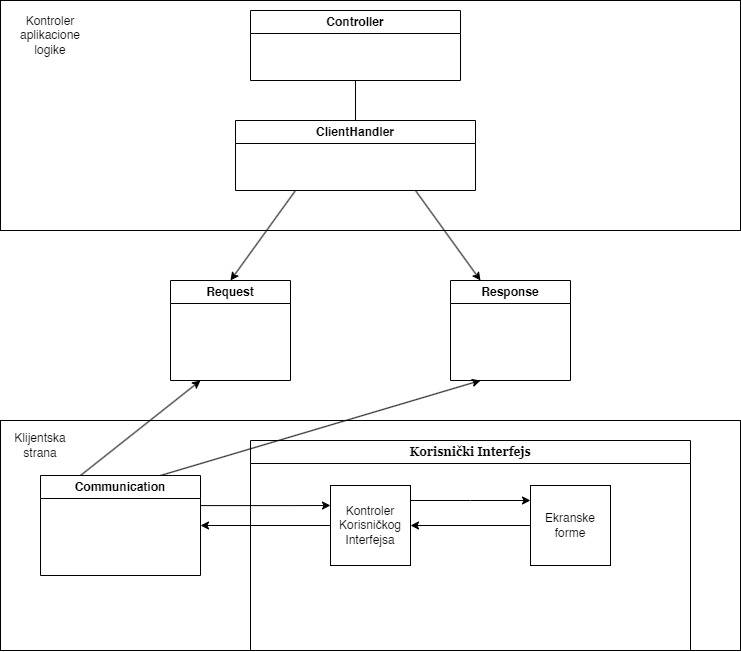


*Структура табеле Напомена за услугу*



*Структура табеле Тип напомене за услугу*

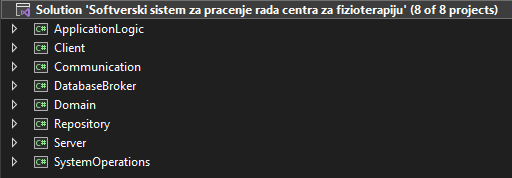




*Коначна архитектура софтверског система*

# Имплементација софтверског система

Развијени софтвер је изграђен у програмском језику C# и концептуализован као клијент-сервер апликација. Као окружење за развој коришћен је Visual Studio 2022, док је SQL Server служио као систем за управљање базом података. Целокупан систем је подељен у осам различитих пројеката: Server, Client, ApplicationLogic, SystemOperations, Communication, Domain, Repository и DatabaseBroker. Ова структура произилази из дизајна софтверског система и може се визуализовати на приложеној слици.



# 6 Тестирање софтверског система

Сваки од случајева коришћења је прошао кроз процес тестирања. Током тестирања, покушано је не само са исправним подацима, већ и са неисправним подацима како би се анализирао резултат извршења. На основу извршених тестова, идентификовани су и исправљени било какви уочени недостаци.

# 7 Литература

1. Microsoft. (2023). .NET documentation. <https://learn.microsoft.com/en-us/dotnet/core/introduction> (приступано: април 2024)
2. Nakov, Svetlin, and Veselin Kolev. 2013. Fundamentals of Computer Programming With C#: The Bulgarian C# Book.
3. Digiš. (2023). Programiranje u jeziku C#. [http://digis.edu.rs/course/view.php?id=300](http://digis.edu.rs/course/view.php?id=300%20)
4. (приступано: април 2024)
5. Rouse, Margaret. (2011). “Windows Forms.” Techopedia. December 20, 2011.<https://www.techopedia.com/definition/24300/windows-forms-net>. (приступано: април 2024)
6. Влајић, С. (2020). ПРОЈЕКТОВАЊЕ СОФТВЕРА (СКРИПТА - РАДНИ МАТЕРИЈАЛ - 2020)

Top of Form