

УНИВЕРЗИТЕТ У БЕОГРАДУ
ФАКУЛТЕТ ОРГАНИЗАЦИОНИХ НАУКА

ЗАВРШНИ РАД

Софтверски систем за праћење рада
туристичке агенције у Јава окружењу

Ментор:

Др Синиша Влајић,
Редовни професор

Студент:

Наталија Бранковић 2020/0065

Београд, 2024.

Садржај

1. Увод	5
2. Упрошћена Ларманова метода развоја софтвера	6
2.1. Прикупљање корисничких захтева.....	7
2.2. Фаза анализе	7
2.3. Фаза пројектовања.....	8
2.4. Фаза имплементације	8
2.5. Фаза тестирања	8
3. Јава технологије.....	9
3.1. Концепти објектно-оријентисаног програмирања и Јава програмског језика	10
3.1.1. Концепт класе објекта	11
3.1.2. Концепт наслеђивања	11
3.1.3. Апстрактне класе и интерфејси	12
3.2. Графички кориснички интерфејс у Јави	14
3.2.1. Јава Swing компоненте	14
3.2.2. Догађаји	15
3.3. Рад у мрежи	16
3.3.1. IP адреса.....	16
3.3.2. URL адреса	17
3.4. Сокети.....	17
3.5. Нити	18
3.6. Рад са базом података	19
4. Студијски пример	20
4.1. Прикупљање корисничких захтева.....	20
4.1.1. Вербални опис.....	20
4.1.2. Случајеви коришћења.....	21
СК1: Случај коришћења – Пријава запосленог.....	22
СК2: Случај коришћења – Креирање корисника.....	23
СК3: Случај коришћења – Претраживање корисника.....	24
СК4: Случај коришћења – Измена података о кориснику.....	25
СК5: Случај коришћења – Брисање корисника.....	26
СК6: Случај коришћења – Креирање аранжмана	27
СК7: Случај коришћења – Претраживање аранжмана.....	28

СК8: Случај коришћења – Брисање аранжмана	29
СК9: Случај коришћења – Измена података о аранжману.....	30
СК10: Случај коришћења – Креирање резервације (сложен СК).....	31
СК11: Случај коришћења – Измена резервације (сложен СК).....	32
4.2. Анализа	33
4.2.1. Понашање софтверског система - Системски дијаграми секвенци.....	33
ДС1: Дијаграм секвенце случаја коришћења – Пријава запосленог	33
ДС2: Дијаграм секвенце случаја коришћења – Креирање корисника.....	34
ДС3: Дијаграм секвенце случаја коришћења – Претраживање корисника	35
ДС4: Дијаграм секвенце случаја коришћења – Измена података о кориснику	37
ДС5: Дијаграм секвенце случаја коришћења – Брисање корисника	40
ДС6: Дијаграм секвенце случаја коришћења – Креирање аранжмана.....	43
ДС7: Дијаграм секвенце случаја коришћења – Претраживање аранжмана	44
ДС8: Дијаграм секвенце случаја коришћења – Брисање аранжмана.....	46
ДС9: Случај коришћења – Измена података о аранжману	49
ДС10: Дијаграм секвенце случаја коришћења – Креирање резервације	52
ДС11: Дијаграм секвенце случаја коришћења – Измена резервације	54
4.2.2. Понашање софтверског система – Дефинисање уговора о системским операцијама	58
4.2.3. Структура софтверског система – Концептуални (доменски) модел	61
4.2.4. Структура софтверског система – Релациони модел	62
4.3. Пројектовање	66
4.3.1. Архитектура софтверског система	66
4.3.2. Пројектовање корисничког интерфејса	67
4.3.2.1. Пројектовање екранских форми	67
СК1: Случај коришћења – Пријава запосленог.....	70
СК2: Случај коришћења – Креирање корисника.....	73
СК3: Случај коришћења – Претраживање корисника	76
СК4: Случај коришћења – Измена података о кориснику	81
СК5: Случај коришћења – Брисање корисника.....	87
СК6: Случај коришћења – Креирање аранжмана.....	92
СК7: Случај коришћења – Претраживање аранжмана.....	95
СК8: Случај коришћења – Брисање аранжмана.....	100
СК9: Случај коришћења – Измена података о аранжману	105

СК10: Случај коришћења – Креирање резервације (сложен СК)	111
СК11: Случај коришћења – Измена резервације (сложен СК).....	114
4.3.2.2. Пројектовање контролера корисничког интерфејса	120
4.3.3. Пројектовање апликационе логике.....	120
4.3.3.1. Контролер апликационе логике.....	121
4.3.3.2. Пословна логика	121
4.3.3.3 Пројектовање складишта података	137
4.4. Имплементација.....	139
4.5. Тестирање.....	142
5. Закључак.....	144
6. Литература.....	145

1. Увод

Туристичке агенције се све више ослањају на савремене информационе технологије како би ефикасно управљале својим услугама и задовољиле потребе корисника. Употреба модерних софтверских решења и апликација постао је неизоставан део свакодневног пословања, омогућавајући агенцијама да одрже конкурентност и пруже висок ниво квалитета услуга на тржишту које се стално мења.

У овом раду биће представљен развој софтверског система за праћење рада туристичке агенције. У питању је десктоп апликација за вођење евиденција и управљање корисницима, аранжманима и резервацијама на једноставан и ефикасан начин. Циљ овог софтверског система је да омогући запосленима у туристичкој агенцији лакшу и бржу обраду података, као и бољу организацију и прегледност информација.

Рад је подељен на теоријски и на практични део. Теоријски део започиње описом упрошћене Ларманове методологије развоја софтвера, која обухвата фазе прикупљања корисничких захтева, анализе, пројектовања, имплементације и тестирања [1].

Затим су детаљно описане различите Јава технологије које су коришћене у развоју овог софтверског система. Описани су основни концепти објектно-оријентисаног програмирања као што су класе, наслеђивање, апстрактне класе и интерфејси. Такође, разматране су и технологије за развој графичког корисничког интерфејса у Јави, укључујући Јава *Swing* компоненте. Додатно, обрађене су и теме везане за мрежно програмирање, укључујући *IP* и *URL* адресе, сокете и нити. На крају, посебна пажња је посвећена раду са базама података кроз *JDBC API*, који омогућава ефикасну комуникацију са базама података [3].

Практични део се састоји од студијског примера развоја софтверског система за праћење рада туристичке агенције развијеног у програмском језику Јава и пројектованог као клијент-сервер апликација. При пројектовању, примењена је тронивојска архитектура софтверског система. За управљање базом података коришћен је *MySQL*, док је развојно окружење *NetBeans IDE 14*. Пројекат је реализован пратећи пет фаза упрошћене Ларманове методологије развоја софтвера.

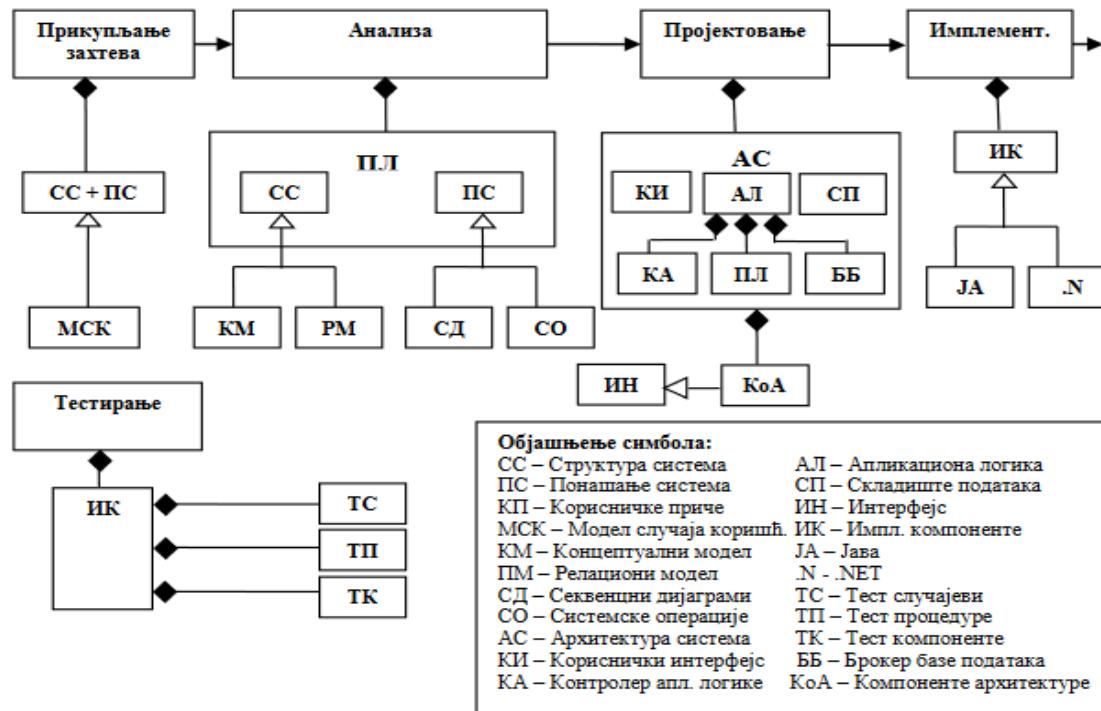
2. Упрошћена Ларманова метода развоја софтвера

Ларманова метода развоја софтверских система је једна од најпознатијих и често коришћених метода у софтверском инжењерству. Пружа јасан и структуриран приступ развоју софтвера, омогућавајући да се системски и ефикасно управља комплетним животним циклусом софтверског система.

Упрошћена Ларманова метода се састоји из следећих фаза [1]:

- Прикупљање захтева
- Анализа
- Пројектовање
- Имплементација
- Тестирање

Дијаграм Ларманове методе приказан је на слици 1.



Слика 1 - Развој софтверског система према Лармановој методи [1]

2.1. Прикупљање корисничких захтева

У фази прикупљања корисничких захтева идентификују се и дефинишу кориснички захтеви везани за пројекат, односно софтверски систем. Овде се прикупљају информације о томе како корисник жели да софтвер изгледа и функционише [1].

Прва фаза у развоју софтверског система се описује помоћу *UML* модела случаја коришћења који се састоји од [1]:

- **Случајева коришћења (СК):** Описују жељене функције система кроз скуп сценарија, односно скуп жељених коришћења система од стране актора.
- **Актори (АК):** Спомљни корисници система који интерагују са софтервом преко дефинисаних случајева коришћења.
- **Везе:** Представљају односе између случајева коришћења и актора.

Сценарио описује једно жељено коришћење система од стране актора и може имати један главни и више алтернативних сценарија. Случајеви коришћења иницијално се представљају текстуално, а касније се представљају преко дијаграма [1].

2.2. Фаза анализе

Фаза анализе описује логичку структуру и понашање софтверског система, који заједно чине пословну логику софтверског система [1].

Понашање софтверског система описује се преко [1]:

- **Системских дијаграма секвенци:** За сваки сценарио случаја коришћења (СК) се приказују догађаји у одређеном редоследу, који успостављају интеракцију између актора и софтверског система.
- **Системских операција:** Праве се уговори за сваку од уочених системских операција.

Логичка структура софтверског система описује се помоћу [1]:

- **Концептуалног модела:** Описује концептуалне класе домена проблема и њихове асоцијације.
- **Релационог модела:** Креира се из концептуалног модела и служи као основа за пројектовање релационе базе података.

2.3. Фаза пројектовања

У фази пројектовања се прави архитектура будућег софтверског система.

Архитектура софтверског система је тронивојска и састоји се од [1]:

- **Корисничког интерфејса:** Представља улазно–излазну репрезентацију софтверског система. Дефинисан је преко скупа екранских форми и контролера корисничког интерфејса.
- **Апликационе логике:** Описује структуру и понашање софтверског система. Састоји се од контролера апликационе логике, пословне логике и брокера базе података.
- **Складишта података:** Пројектује се на основу релационог модела.

2.4. Фаза имплементације

У фази имплементације врши се кодирање софтверског система у одређеном програмском језику, што представља имплементациону компоненту која ће се тестирати у наредној фази. Ова фаза је кључна јер представља прелазак од теорије до праксе, односно од анализе и пројектовања до конкретног софтверског система који се може користити. Имплементационе компоненте се праве и реализују у некој од технологија и оне представљају реализацију компоненти које су добијене у фази пројектовања [1].

2.5. Фаза тестирања

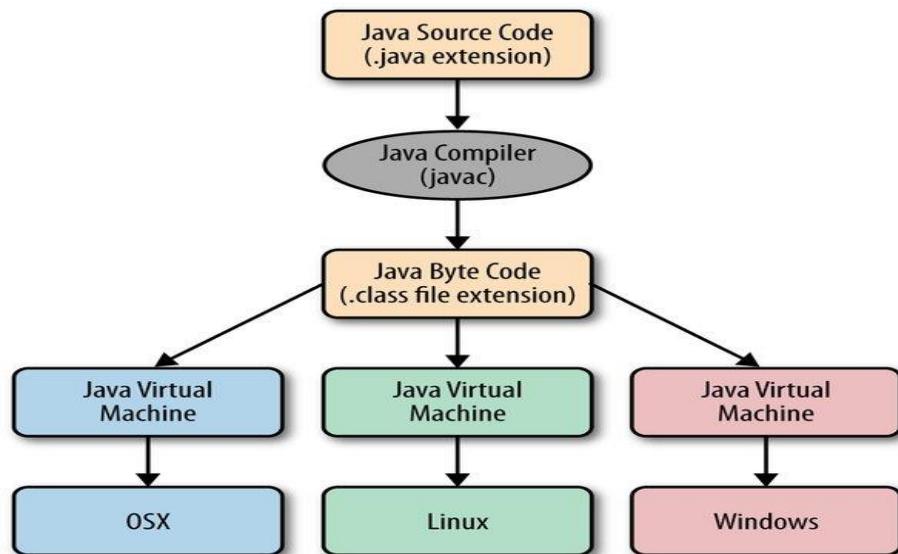
Фаза тестирања подразумева проверу сваке од компоненти које су имплементиране у претходној фази како би се утврдило да ли софтвер ради исправно и да ли задовољава функционалне и нефункционалне захтеве. Компоненте се тестирају тако што се за сваку од њих праве тест случајеви, тест процедуре и тест компоненте. Ова фаза је кључна јер обједињује процес развоја софтвера и осигуруја да све имплементационе компоненте функционишу како је предвиђено [1].

3. Јава технологије

Јава је један од најпопуларнијих програмских језика у свету софтверског инжењерства који је развијен од стране компаније *Sun Microsystems* 1995. године. Јава је пројектована као објектно-оријентисани програмски језик са посебним акцентом на преносивост, перформансе и безбедност. Њена главна предност лежи у филозофији "Пиши једном, ради свуда", што значи да се један исти програм може покренути на различитим оперативним системима без потребе за модификацијом кода [4].

Ова независност од платформе постиже се коришћењем Јава виртуелне машине (*JVM*), која је програмска платформа за извршавање Јава кода. Јава програм се најпре пише у изворном коду са екstenзијом *.java*, који се затим компајлира помоћу јава компајлера (*javac*) у бајт-код са екstenзијом *.class*. Бајт-код није специфичан ни за једну платформу и представља извршни код који је независан од хардвера и оперативног система [2].

Када се бајт-код извршава на *JVM*, он се интерпретира и прилагођава конкретном оперативном систему, што омогућава да исти бајт-код ради на *Windows*, *macOS*, *Linux* и другим оперативним системима без потребе за изменом кода. За сваки оперативни систем постоји посебна верзија *JVM-a*, што значи да је предуслов за извршавање Јава програма на неком систему постојање *JVM-a* за тај систем [3].



Слика 2 – Извршавање Јава кода

3.1. Концепти објектно-оријентисаног програмирања и Јава програмског језика

Објектно оријентисано програмирање (ООП) је парадигма програмирања која користи објекте као основне јединице за креирање софтверских решења.

Сваки објекат поседује [5]:

- **Стања:** Вредности које карактеришу објекат, познате као атрибути или поља.
- **Понашања:** Функције које дефинишу шта објекат може да ради, познате као методе.

Објекти представљају ентитете из стварног света, попут кућа, бицикала или књига. Користе се за извршавање различитих задатака у програму. На пример, бицикл има стања као што су брзина и тежина, као и понашања као што су кочење и мењање брзина [8].

ООП омогућава моделирање стварних ентитета и њихових интеракција на интуитиван и лако разумљив начин. Комбиновањем података (поља) и функција (метода) унутар објекта, постиже се већа модуларност и организованост кода. Ова парадигма омогућава програмерима да креирају апликације које су једноставне за одржавање и проширивање [5].

Кључни принципи ООП [8]:

- **Апстракција:** Пружа поједностављену слику стварног објекта, занемарујући неважне детаље и узимајући у обзир само оне карактеристике које су релевантне за програм.
- **Енкапсулација:** Штити податке у класи од директног приступа, дозвољавајући промену само кроз дефинисане методе.
- **Наслеђивање:** Механизам који омогућава да се из једне класе изведе нова, наслеђујући њене делове и додајући специфичне особине и методе. Ово омогућава креирање хијерархије класа и поновну употребу кода.
- **Полиморфизам:** Способност исте методе да делује различито у зависности од типа или броја прослеђених параметара.

3.1.1. Концепт класе објекта

У Јави, класе и објекти су основни концепти објектно-оријентисаног програмирања (ООП) који се користе за представљање стварних концепата и ентитета. Класа представља апстрактни концепт који дефинише скуп особина и понашања према којима ће се објекти груписати. Она дефинише атрибуте и методе објеката који припадају истој класи. Објекат представља конкретну инстанцу своје класе [8].

```
// Definicija klase Knjiga
public class Knjiga {
    // Atributi klase Knjiga
    String naslov;
    String autor;
    int godinaIzdavanja;

    // Metoda klase Knjiga
    void prikaziDetalje() {
        System.out.println("Naslov: " + naslov + ", Autor: "
            + autor + ", Godina izdavanja: " + godinaIzdavanja);
    }

    // Glavna metoda za kreiranje i korišćenje objekata
    public static void main(String[] args) {
        // Kreiranje objekta klase Knjiga
        Knjiga mojaKnjiga = new Knjiga();

        // Inicijalizacija atributa objekta
        mojaKnjiga.naslov = "Na Drini ćuprija";
        mojaKnjiga.autor = "Ivo Andrić";
        mojaKnjiga.godinaIzdavanja = 1945;

        // Poziv metode objekta
        mojaKnjiga.prikaziDetalje();
        // Izlaz: Naslov: Na Drini ćuprija, Autor: Ivo Andrić, Godina izdavanja: 1945
    }
}
```

Слика 3 – Пример класе и објекта у Јави

3.1.2. Концепт наслеђивања

Наслеђивање је један од кључних концепата у Јава програмском језику, који омогућава креирање нових класа на основу постојећих. Основну класу називамо надкласа (*superclass*), док изведену класу називамо подкласа (*subclass*) [2].

Класа која је изведена из основне класе наслеђује све атрибуте и методе основне класе. Поред заједничких карактеристика, класе могу имати и специфичне карактеристике за своје подгрупе (нпр. књиге могу бити романи, збирке песама, научни радови...). На пример, подкласа *Roman* наслеђује сва стања и методе дефинисане у класи *Knjiga*, али може имати и своје специфичне особине као што су *glavniLikovi* и *mestoRadnje*.

Свака класа може имати највише једну надкласу, док свака класа може имати више подкласа. Кључна реч за наслеђивање у Јави је *extends* [2].

```
public class Roman extends Knjiga{  
    // код који одређује само roman  
}
```

Слика 4 - Подкласа Роман

3.1.3. Апстрактне класе и интерфејси

Апстрактне класе и интерфејси су два кључна концепта у Јава програмском језику који омогућавају апстракцију, односно скривање детаља имплементације.

Апстрактне класе се дефинишу на исти начин као и обичне класе, али не могу имати своја појављивања. Користе се када класа садржи бар једну апстрактну методу, односно методу без тела чија имплементација се преноси на подкласе. Апстрактна класа сама по себи не може бити инстанцирана, већ служи као основна класа за друге класе које је проширују и имплементирају њене апстрактне методе. Испред апстрактне класе и апстрактне методе се ставља кључна реч *abstract* [2].

```
abstract class Radnik {  
    // Апстрактна метода  
    abstract void obavljaJPosao();  
    // Конкретна метода  
    void odmor() {  
        System.out.println("Радник је на одмору.");  
    }  
}
```

Слика 5 – Апстрактна класа Радник

```

// Подкласа Програмер која наслеђује апстрактну класу Радник
class Programer extends Radnik {
    // Реализација апстрактне методе
    @Override
    void obavljaJPosao() {
        System.out.println("Програмер пише код.");
    }
}

```

Слика 6 – Подкласа Програмер наслеђује апстрактну класу Радник

Интерфејси, с друге стране, развајају спецификацију метода од њихове имплементације. Све методе у интерфејсу су апстрактне. Имплементација метода из интерфејса се врши у класама које тај интерфејс имплементирају (наслеђују). Интерфејс се не може директно инстанцирати и не може да садржи конструкторе. Интерфејс се дефинише се преко кључне речи *interface* [2].

```

interface MuzickiInstrument {
    void sviraj();
    void stani();
}

```

Слика 7 – Интерфејс МузичкиИнструмент

```

// Класа Гитара која имплементира интерфејс МузичкиИнструмент
class Gitara implements MuzickiInstrument {
    // Реализација метода интерфејса
    @Override
    public void sviraj() {
        System.out.println("Гитара свира.");
    }

    @Override
    public void stani() {
        System.out.println("Гитара је престала да свира.");
    }
}

```

Слика 8 – Класа Гитара имплементира интерфејс МузичкиИнструмент

3.2. Графички кориснички интерфејс у Јави

Када су рачунари први пут уведени, били су доступни само стручњацима у контролисаним условима. Увођењем терминала и *time-sharing* система 1960-их, више људи је могло истовремено да користи рачунаре преко командно-линијског интерфејса [3].

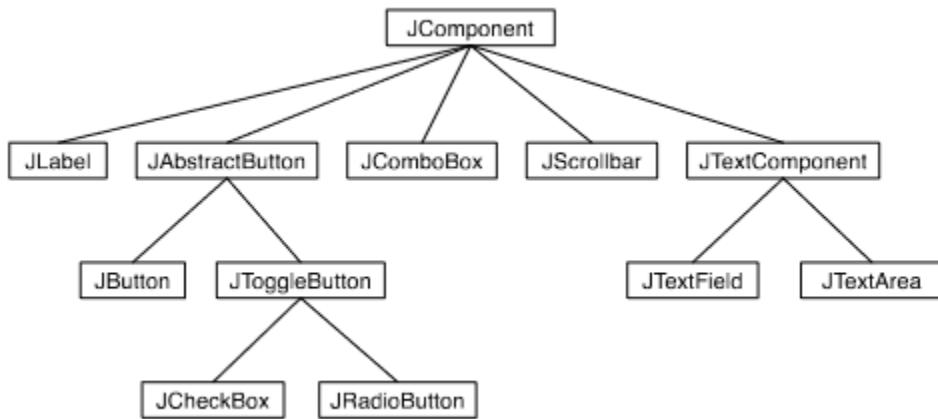
Овај начин интеракције је замењен графичким корисничким интерфејсом (*GUI*), који користи елементе као што су прозори, менији, дугмад, поља за унос текста... *GUI* омогућава корисницима интуитивну и лаку интеракцију са рачунаром, без потребе за познавањем сложених команди и програмских језика [4].

Јава програми користе стандардне *GUI* компоненте које су крос-платформске, што значи да изглед може мало да се разликује на различитим платформама, али функционалност остаје иста. Јава има два комплета *GUI* компоненти: *AWT (Abstract Windowing Toolkit)* и *Swing*. *Swing* је модернији и користи се у већини савремених Јава програма [3].

3.2.1. Јава Swing компоненте

Java Swing је библиотека за креирање графичких корисничких интерфејса (*GUI*). *Swing* је надоградња на *AWT (Abstract Window Toolkit)* и пружа већи број могућности и већу флексибилност у пројектовању корисничких интерфејса [3].

Једна од основних карактеристика *Swing* библиотеке је то што све компоненте наслеђују класу *JComponent*. Ова класа пружа заједничка својства и методе које деле све *Swing* компоненте, као што су могућност цртања, обраде догађаја и управљања распоредом. На пример, компоненте као што су *JLabel*, *JAbstractButton*, *JComboBox*, *JTextComponent* су подкласе *JComponent* класе, али свака од њих има своје специфичне функционалности и изглед [3].



Слика 9 – Хијерархија Јава Swing компоненти [3]

Swing компоненте укључују широк спектар елемената који омогућавају корисницима да интерагују са апликацијама на различите начине. *JButton* је једна од најчешће коришћених *Swing* компонента и представља дугме које корисници могу да кликну како би извршили неку акцију. *JLabel* се користи за приказ текста или слика, док *JTextField* омогућава корисницима да уносе текст. *JComboBox* представља падајући мени са опцијама које корисник може изабрати. Поред ових основних компоненти, *Swing* нуди и сложеније компоненте као што су *JTable* за приказ табела података и *JTree* за приказ хијерархијских структура података [4].

Swing такође подржава напредне функционалности као што су прилагођени изглед и осећај (*look and feel*), што омогућава програмерима да промене изглед својих апликација у складу са различитим платформама или личним преференцијама. Комбиновањем различитих компоненти и њиховим прилагођавањем, програмери могу креирати високо интерактивне и кориснички прилагођене апликације [3].

3.2.2. Догађаји

Када корисник интерактује са *GUI* компонентама, генерише се догађај (*event*). На пример, кликом на дугме или притиском на *Enter* током уноса у текстуалном пољу генерише се догађај. Када се догађај деси, креира се објекат који представља тај догађај, као што је објекат класе *ActionEvent* за догађај клика мишем. Овај објекат садржи информације о извору догађаја и његовом типу [10].

Објекат догађаја се затим прослеђује објекту слушаоца догађаја (*event handler*), који је одговоран за обраду догађаја и дефинисање реакције апликације. *Event handler* прима објекат догађаја и извршава одговарајућу методу као одговор на догађај. Да би се догађај успешно обрадио, неопходно је успоставити везу између извора догађаја и *event handler-a*, чиме се омогућава апликацији да реагује на корисничке акције [10].

3.3. Рад у мрежи

3.3.1. IP адреса

IP адреса (*Internet Protocol address*) је јединствена бројчана ознака која идентификује сваки уређај повезан на мрежу. Њена основна функција је да омогући уређајима да се идентификују и међусобно комуницирају унутар глобалне мреже, познате као интернет [9].

IP адреса је обично представљена као низ бројева одвојених тачкама. Пример такве адресе у верзији *IPv4* (*Internet Protocol Version 4*) је: „192.168.1.1“. Овај формат садржи четири броја у опсегу од 0 до 255. Свака од ових цифара је бинарни број од 8 бита, што укупно чини 32 бита. Адресе се обично раздвајају тачком ради лакшег читања и идентификације [9].

IPv4 адресе су најчешће коришћене, али због ограниченог броја доступних адреса, уведена је верзија *IPv6*, која користи 128-битне адресе и омогућава далеко већи број јединствених адреса [9].

Повезивање рачунара у мрежу и пренос података између њих се обично реализује путем *TCP/IP* (*Transmission Control Protocol/Internet Protocol*) протокола. IP протокол је одговоран за проналажење интернет адресе одредишног рачунара и усмеравању података до њега од извornog рачунара, док TCP протокол управља успостављањем и прекидањем веза између рачунара, као и контролним функцијама које обезбеђују поуздан пренос података [2].

3.3.2. URL адреса

URL (Uniform Resource Locator) адреса представља путању до одређеног садржаја на интернету. Чува се у облику стринга и састоји се од неколико компоненти: протокола, адресе рачунара, броја порта и путање до датотеке [2].

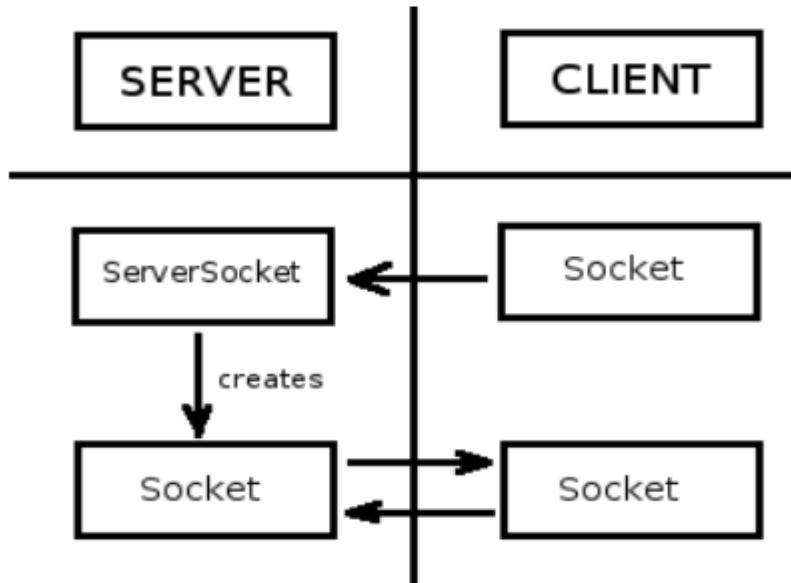
На пример, у *URL* адреси *http://example.com:8080/files/document.txt*, „*http*“ представља протокол (*HyperText Transfer Protocol*), „*example.com*“ је симболичка адреса рачунара, „8080“ је број порта, а „*/files/document.txt*“ је путања до датотеке.

3.4. Сокети

Клијент-сервер комуникација представља основни модел за размену података у мрежама, где сервер пружа услуге или ресурсе, а клијенти их захтевају. У овом моделу, сервер је увек спреман да одговори на захтеве клијената, омогућавајући стабилну и ефикасну размену података. Сокети играју кључну улогу у овом процесу, омогућавајући програмима на различитим рачунарима да комуницирају преко мреже користећи *TCP/IP* протоколе за повезивање и пренос података [6].

Адреса сокета се састоји од адресе рачунара и броја порта који је генерисан помоћу сокета. У Јави, сокети се креирају помоћу класа *Socket* и *ServerSocket*. *ServerSocket* класа служи за ослушкивање захтева клијената на одређеном порту и прихватање тих захтева, док *Socket* класа омогућава клијентима да успоставе конекцију са сервером [2].

Процес комуникације почиње креирањем серверског сокета који ослушкије захтеве клијената на одређеном порту. Када клијент жели да се повеже, он креира клијентски сокет и покушава да се повеже на адресу и порт сервера. Успостављањем конекције, серверска метода *accept()* враћа нови *Socket* објекат који представља везу са клијентом. Након успостављања конекције, обе стране отварају улазне и излазне токове за размену података. Комуникација се врши слањем и примањем података преко ових токова, а по завршетку комуникације, сокети се затварају како би се ослободили ресурси [6].



Слика 10 – Клијент - Сервер комуникација [6]

3.5. Нити

Конкурентност у програмирању подразумева истовремено извршавање више процеса који деле ресурсе система, што може довести до проблема као што су мртво закључавање (*deadlock*) и гладовање (*starvation*). Јава решава овај проблем кроз концепт нити (*threads*), који омогућава једном програму да извршава више задатака (нити) истовремено, делећи исти меморијски простор [2].

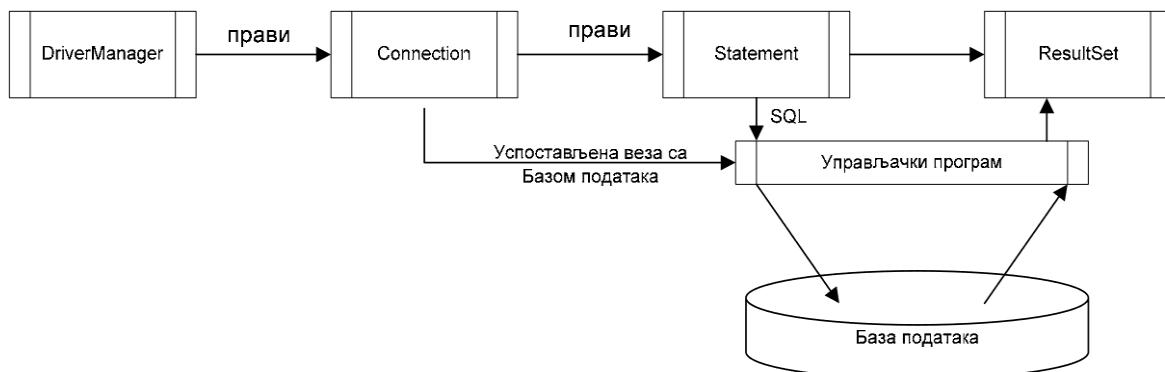
Програм у Јави, при покретању, аутоматски креира и извршава једну главну нит програма. Нове нити се креирају реализацијом интерфејса *Runnable* или проширењем класе *Thread*. *Runnable* интерфејс дефинише методу *run()* која садржи код који ће се извршавати у нити. Класа *Thread*, поред методе *run()*, обезбеђује и друге методе за контролу нити [2].

Нити се покрећу позивом методе *start()*, која иницира извршавање *run()* методе у новој нити. Јава обезбеђује различите начине за управљање нитима, укључујући синхронизацију како би се избегли проблеми са конкурентним приступом ресурсима, чиме се осигурува стабилно и ефикасно извршавање паралелних задатака [2].

3.6. Рад са базом података

Повезивање Јава програма са системима за управљање базама података (СУБП) остварује се путем *JDBC (Java Database Connectivity)* API-а. Овај API омогућава Јави да комуницира са различитим базама података. За сваку специфичну базу података потребан је одговарајући управљачки програм (драјвер) који омогућава приступ бази. *JDBC* је посебно користан за рад са релационим базама података, али може да ради са било којим табеларним приказом података [11].

Процес повезивања Јава програма са базом података се остварује помоћу *JDBC DriverManager* класе која креира објекат *Connection*, који представља успостављену везу са базом података. Након успостављања везе, креира се објекат *Statement* који служи за слање *SQL* упита бази података. Када се *SQL* упит изврши, резултати се чувају у објекту *ResultSet*, који омогућава итерацију кроз резултате и приступ подацима. Управљачки програм (драјвер) посредује у комуникацији између *Statement* објекта и базе података, преводећи *SQL* упите у формат који база разуме и враћајући резултате у формату који Јава може да обради [2].



Слика 11 – Поступак извршења операције над базом података [2]

4. Студијски пример

Студијски пример је развијен у програмском језику Јава, користећи описане Јава технологије и пратећи пет фаза упрошћене Ларманове методе.

4.1. Прикупљање корисничких захтева

4.1.1. Вербални опис

Апликација омогућава вођење евидентије о корисницима услуга туристичке организације, аранжманима које туристичка организација реализује, као и о резервацијама аранжмана од стране корисника. Како је овај систем намењен само запосленима у туристичкој агенцији, постоји могућност њихове пријаве на систем.

Запослени у туристичкој организацији може да креира и унесе новог корисника, претражује корисника, измени његове податке или га избрише из система. Запослени у туристичкој организацији може да креира аранжмане, врши њихову претрагу, измену или их обрише. Запослени у туристичкој организацији је задужен за креирање и измену резервација аранжмана.

4.1.2. Случајеви коришћења

У овој апликацији је идентификовано једанаест случајева коришћења:

1. Пријава запосленог
2. Креирање корисника
3. Претраживање корисника
4. Измена података о кориснику
5. Брисање корисника
6. Креирање аранжмана
7. Претраживање аранжмана
8. Измена података о аранжману
9. Брисање аранжмана
10. Креирање резервације (сложен СК)
11. Измена резервације (сложен СК)



Слика 12 - Модел случајева коришћења

СК1: Случај коришћења – Пријава запосленог

Назив СК

Пријава **запосленог**

Актори СК

Запослени

Учесници СК

Запослени и **систем** (програм)

Предуслов: **Систем** је укључен и отворена је форма за пријављивање.

Основни сценарио СК

1. **Запослени** уноси податке за пријаву на систем. (АПУСО)
2. **Запослени проверава** да ли је коректно унео податке. (АНСО)
3. **Запослени позива** **систем** да пронађе налог са задатим подацима. (АПСО)
4. **Систем тражи** налог. (СО)
5. **Систем приказује** **запосленом** поруку: „Успешна пријава“. (ИА)

Алтернативна сценарија

5.1 Уколико **систем** не може да нађе тражени налог, **систем** приказује **запосленом** поруку: „**Систем** не може да пронађе налог“. (ИА)

СК2: Случај коришћења – Креирање корисника

Назив СК

Креирање корисника

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са корисницима.

Основни сценарио СК

1. Запослени уноси податке о кориснику у систем. (АПУСО)
2. Запослени контролише да ли је коректно унео податке о кориснику у систем. (АНСО)
3. Запослени позива систем да креира новог корисника. (АПСО)
4. Систем креира новог корисника. (СО)
5. Систем приказује запосленом поруку: „Систем је успешно креирао новог корисника“. (ИА)

Алтернативна сценарија

5.1 Уколико систем не може да креира новог корисника, приказује запосленом поруку: „Систем не може да креира новог корисника“. (ИА)

СК3: Случај коришћења – Претраживање корисника

Назив СК

Претраживање корисника

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са корисницима.

Основни сценарио СК

1. Запослени уноси вредност по којој претражује кориснике. (АПУСО)
2. Запослени позива систем да нађе кориснике по задатој вредности. (АПСО)
3. Систем тражи кориснике по задатој вредности. (СО)
4. Систем приказује запосленом податке о корисницима и поруку: „Систем је нашао кориснике по задатој вредности“. (ИА)
5. Запослени бира жељеног корисника. (АПУСО)
6. Запослени позива систем да чита корисника. (АПСО)
7. Систем читава корисника. (СО)
8. Систем приказује запосленом податке о кориснику и поруку: „Систем је учитао корисника“. (ИА)

Алтернативна сценарија

4.1 Уколико систем не може да нађе кориснике по задатој вредности, он приказује запосленом поруку: „Систем не може да нађе кориснике по задатој вредности“. Прекида се извршење сценарија. (ИА)

8.1 Уколико систем не може да чита корисника, приказује запосленом поруку: „Систем не може да чита корисника“. (ИА)

СК4: Случај коришћења – Измена података о кориснику

Назив СК

Измена података о [кориснику](#)

Актори СК

[Запослени](#)

Учесници СК

[Запослени](#) и [систем](#) (програм)

Предуслов: [Систем](#) је укључен и [запослени](#) је улогован под својом шифром. [Систем](#) приказује форму за рад са [корисницима](#).

Основни сценарио СК

1. [Запослени](#) уноси вредност по којој претражује [кориснике](#). (АПУСО)
2. [Запослени](#) позива [систем](#) да нађе [кориснике](#) по задатој вредности. (АПСО)
3. [Систем](#) тражи [кориснике](#) по задатој вредности. (СО)
4. [Систем](#) приказује [запосленом](#) податке о [корисницима](#) и поруку: „[Систем](#) је нашао [кориснике](#) по задатој вредности“. (ИА)
5. [Запослени](#) бира жељеног [корисника](#). (АПУСО)
6. [Запослени](#) позива [систем](#) да учита [корисника](#). (АПСО)
7. [Систем](#) учитава [корисника](#). (СО)
8. [Систем](#) приказује [запосленом](#) податке о [кориснику](#) и поруку: „[Систем](#) је учитао [корисника](#)“. (ИА)
9. [Запослени](#) уноси ([мења](#)) податке о [кориснику](#). (АПУСО)
10. [Запослени](#) контролише да ли је коректно унео податке о [кориснику](#). (АХСО)
11. [Запослени](#) позива [систем](#) да измени податке о [кориснику](#). (АПСО)
12. [Систем](#) [мења](#) податке о [кориснику](#). (СО)
13. [Систем](#) приказује [запосленом](#) изменењеног [корисника](#) и поруку: „[Систем](#) је успешно изменио податке о [кориснику](#)“. (ИА)

Алтернативна сценарија

4.1 Уколико [систем](#) не може да нађе [кориснике](#) по задатој вредности, он приказује [запосленом](#) поруку: „[Систем](#) не може да нађе [кориснике](#) по задатој вредности“. Прекида се извршење сценарија. (ИА)

8.1 Уколико [систем](#) не може да учита податке о [кориснику](#), он приказује [запосленом](#) поруку „[Систем](#) не може да учита [корисника](#)“. Прекида се извршење сценарија. (ИА)

13.1 Уколико [систем](#) не може да измени податке о [кориснику](#), он приказује [запосленом](#) поруку: „[Систем](#) не може да измени податке о [кориснику](#)“. (ИА)

СК5: Случај коришћења – Брисање корисника

Назив СК

Брисање корисника

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са корисницима.

Основни сценарио СК

1. Запослени уноси вредност по којој претражује кориснике. (АПУСО)
2. Запослени позива систем да нађе кориснике по задатој вредности. (АПСО)
3. Систем тражи кориснике по задатој вредности. (СО)
4. Систем приказује запосленом податке о корисницима и поруку: „Систем је нашао кориснике по задатој вредности“. (ИА)
5. Запослени бира жељеног корисника. (АПУСО)
6. Запослени позива систем да учита корисника. (АПСО)
7. Систем чита корисника. (СО)
8. Систем приказује запосленом податке о кориснику и поруку: „Систем је учитао корисника“. (ИА)
9. Запослени позива систем да обрише корисника. (АПСО)
10. Систем брише корисника. (СО)
11. Систем приказује запосленом поруку: „Систем је успешно обрисао корисника“. (ИА)

Алтернативна сценарија

4.1 Уколико систем не може да нађе кориснике, он приказује запосленом поруку: „Систем не може да нађе кориснике по задатој вредности“. Прекида се извршење сценарија. (ИА)

8.1 Уколико систем не може да учита податке о кориснику, он приказује запосленом поруку „Систем не може да учита корисника“. Прекида се извршење сценарија. (ИА)

11.1 Уколико систем не може да обрише корисника, он приказује запосленом поруку: „Систем не може да обрише корисника“. (ИА)

СК6: Случај коришћења – Креирање аранжмана

Назив СК

Креирање аранжмана

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са аранжманима. Учитана је листа дестинација.

Основни сценарио СК

1. Запослени уноси податке о аранжману. (АПУСО)
2. Запослени контролише да ли је коректно унео податке о аранжману. (АНСО)
3. Запослени позива систем да креира нови аранжман. (АПСО)
4. Систем памти податке о аранжману. (СО)
5. Систем приказује запосленом креирани аранжман и поруку: „Систем је успешно креирао нови аранжман“. (ИА)

Алтернативна сценарија

5.1 Уколико систем не може да креира аранжман, он приказује запосленом поруку: „Систем не може да креира нови аранжман“. (ИА)

СК7: Случај коришћења – Претраживање аранжмана

Назив СК

Претраживање аранжмана

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са аранжманима.

Основни сценарио СК

1. Запослени уноси вредност по којој претражује аранжмане. (АПУСО)
2. Запослени позива систем да нађе аранжмане по задатој вредности. (АПСО)
3. Систем тражи аранжмане по задатој вредности. (СО)
4. Систем приказује запосленом податке о аранжманима и поруку: „Систем је нашао аранжмане по задатој вредности“. (ИА)
5. Запослени бира аранжман. (АПУСО)
6. Запослени позива систем да учита аранжман. (АПСО)
7. Систем учитава аранжман. (СО)
8. Систем приказује запосленом податке о аранжману и поруку: „Систем је учитао аранжман“. (ИА)

Алтернативна сценарија

4.1 Уколико систем не може да нађе аранжмане, он приказује запосленом поруку: „Систем не може да нађе аранжмане по задатој вредности“. Прекида се извршење сценарија. (ИА)

8.1 Уколико систем не може да учита аранжман, он приказује запосленом поруку: „Систем не може да учита аранжман“. (ИА)

СК8: Случај коришћења – Брисање аранжмана

Назив СК

Брисање аранжмана

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са аранжманима.

Основни сценарио СК

1. Запослени уноси вредност по којој претражује аранжмане. (АПУСО)
2. Запослени позива систем да нађе аранжмане по задатој вредности. (АПСО)
3. Систем тражи аранжмане по задатој вредности. (СО)
4. Систем приказује запосленом податке о аранжманима и поруку: „Систем је нашао аранжмане по задатој вредности“. (ИА)
5. Запослени бира аранжман. (АПУСО)
6. Запослени позива систем да учита аранжман. (АПСО)
7. Систем читава аранжман. (СО)
8. Систем приказује запосленом податке о аранжману поруку: „Систем је учитао аранжман“. (ИА)
9. Запослени позива систем да обрише аранжман. (АПСО)
10. Систем брише аранжман. (СО)
11. Систем приказује запосленом поруку: „Систем је успешно обрисао аранжман.“ (ИА)

Алтернативна сценарија

4.1 Уколико систем не може да нађе аранжмане, он приказује запосленом поруку: „Систем не може да нађе аранжмане по задатој вредности“. Прекида се извршење сценарија. (ИА)

8.1 Уколико систем не може да учита податке о аранжману, он приказује запосленом поруку „Систем не може да учита аранжман“. Прекида се извршење сценарија. (ИА)

11.1 Уколико систем не може да обрише аранжман, он приказује запосленом поруку „Систем не може да обрише аранжман“. (ИА)

СК9: Случај коришћења – Измена података о аранжману

Назив СК

Измена података о аранжману

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са аранжманима. Учитана је листа дестинација.

Основни сценарио СК

1. Запослени уноси вредност по којој претражује аранжмане. (АПУСО)
2. Запослени позива систем да нађе аранжмане по задатој вредности. (АПСО)
3. Систем тражи аранжмане по задатој вредности. (СО)
4. Систем приказује запосленом податке о аранжманима и поруку: „Систем је нашао аранжмане по задатој вредности”. (ИА)
5. Запослени бира аранжман. (АПУСО)
6. Запослени позива систем да учита аранжман. (АПСО)
7. Систем учитава аранжман. (СО)
8. Систем приказује запосленом податке о аранжману и поруку: „Систем је учитао аранжман”. (ИА)
9. Запослени уноси (мења) податке о аранжману. (АПУСО)
10. Запослени контролише да ли је коректно унео податке о аранжману. (АНСО)
11. Запослени позива систем да измени податке о аранжману. (АПСО)
12. Систем мења податке о аранжману. (СО)
13. Систем приказује запосленом изменjeni аранжман и поруку: „Систем је успешно изменио податке о аранжману.” (ИА)

Алтернативна сценарија

4.1 Уколико систем не може да нађе аранжмане, он приказује запосленом поруку: „Систем не може да нађе аранжмане по задатој вредности”. Прекида се извршење сценарија. (ИА)

8.1 Уколико систем не може да учита аранжман, он приказује запосленом поруку: „Систем не може да учита аранжман”. Прекида се извршење сценарија. (ИА)

13.1 Уколико систем не може да измени податке о аранжману, он приказује запосленом поруку: „Систем не може да измени податке о аранжману.” (ИА)

СК10: Случај коришћења – Креирање резервације (сложен СК)

Назив СК

Креирање резервације

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са резервацијама. Учитана је листа корисника и аранжмана.

Основни сценарио СК

1. Запослени уноси податке о резервацији. (АПУСО)
2. Запослени контролише да ли је коректно унео податке о резервацији. (АНСО)
3. Запослени позива систем да креира нову резервацију. (АПСО)
4. Систем памти податке о резервацији. (СО)
5. Систем приказује запосленом креiranу резервацију и поруку: „Систем је успешно креирао нову резервацију“. (ИА)

Алтернативна сценарија

5.1 Уколико систем не може да креира нову резервацију, он приказује запосленом поруку „Систем не може да креира нову резервацију“. (ИА)

СК11: Случај коришћења – Измена резервације (сложен СК)

Назив СК

Измена резервације

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са резервацијама. Учитана је листа корисника и аранжмана.

Основни сценарио СК

1. Запослени уноси вредност по којој претражује резервације. (АПУСО)
2. Запослени позива систем да нађе резервације по задатој вредности. (АПСО)
3. Систем тражи резервације по задатој вредности. (СО)
4. Систем приказује запосленом податке о резервацијама и поруку: „Систем је нашао резервације по задатој вредности”. (ИА)
5. Запослени бира резервацију. (АПУСО)
6. Запослени позива систем да учита резервацију. (АПСО)
7. Систем учитава резервацију. (СО)
8. Систем приказује запосленом податке о резервацији и поруку: „Систем је учитао резервацију”. (ИА)
9. Запослени уноси (мења) податке о резервацији. (АПУСО)
10. Запослени контролише да ли је коректно унео податке о резервацији. (АНСО)
11. Запослени позива систем да измени податке о резервацији. (АПСО)
12. Систем памти податке о резервацији. (СО)
13. Систем приказује запосленом изменјену резервацију и поруку: „Систем је успешно изменио податке о резервацији.” (ИА)

Алтернативна сценарија

4.1 Уколико систем не може да нађе резервације, он приказује запосленом поруку: „Систем не може да нађе резервације по задатој вредности”. Прекида се извршење сценарија. (ИА)

8.1 Уколико систем не може да учита податке о резервацији, он приказује запосленом поруку „Систем не може да учита резервацију”. Прекида се извршење сценарија. (ИА)

13.1 Уколико систем не може да измени податке о резервацији, он приказује запосленом поруку „Систем не може да измени податке о резервацији”. (ИА)

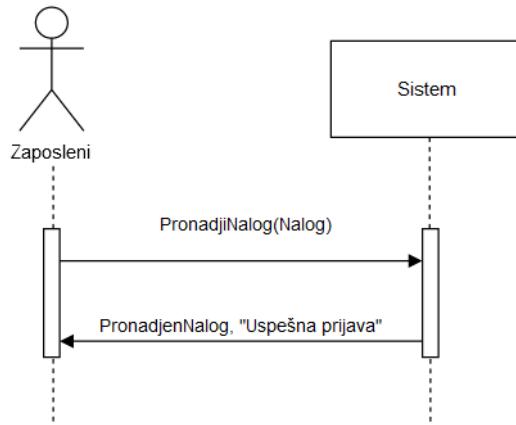
4.2. Анализа

4.2.1. Понашање софтверског система - Системски дијаграми секвенци

ДС1: Дијаграм секвенце случаја коришћења – Пријава запосленог

Основни сценарио СК

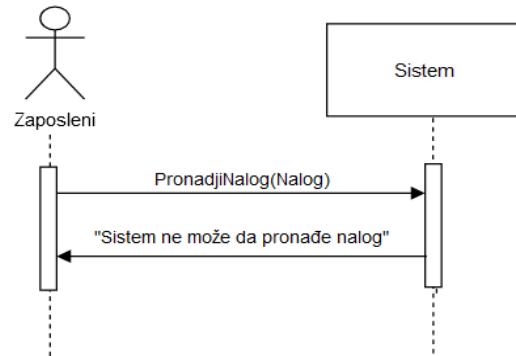
1. Запослени позива систем да пронађе налог са задатим подацима. (АПСО)
2. Систем приказује запосленом поруку: „Успешна пријава“. (ИА)



Слика 13 - Дијаграм секвенце случаја коришћења – Пријава запосленог (основни сценарио)

Алтернативна сценарија

- 2.1. Уколико систем не може да нађе тражени налог, систем приказује запосленом поруку: „Систем не може да пронађе налог“. (ИА)



Слика 14 - Дијаграм секвенце случаја коришћења – Пријава запосленог (алтернативни сценарио)

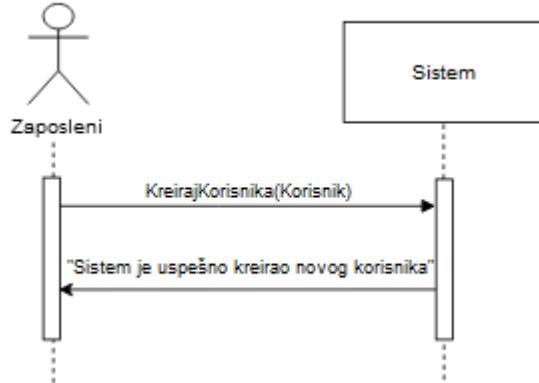
Са наведених дијаграма секвенци уочава се једна системска операција:

1. signal **PronadjiNalog(Nalog)**

ДС2: Дијаграм секвенце случаја коришћења – Креирање корисника

Основни сценарио СК

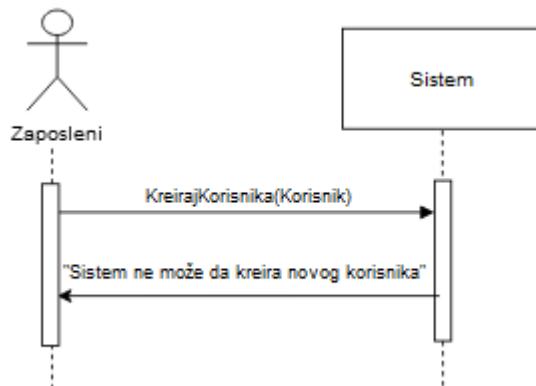
1. Запослени позива систем да креира новог корисника. (АПСО)
2. Систем приказује запосленом поруку: „Систем је успешно креирао новог корисника“. (ИА)



Слика 15 - Дијаграм секвенце случаја коришћења – Креирање корисника (основни сценарио)

Алтернативна сценарија

- 2.1 Уколико систем не може да креира новог корисника, приказује запосленом поруку: „Систем не може да креира новог корисника“. (ИА)



Слика 16 - Дијаграм секвенце случаја коришћења – Креирање корисника (алтернативни сценарио 1)

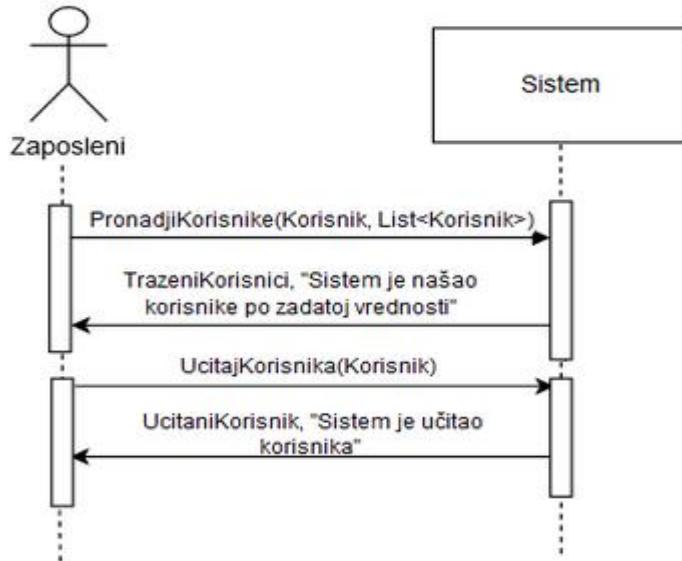
Са наведених дијаграма секвенци уочава се једна системска операција:

1. signal **KreirajKorisnika(Korisnik)**

ДСЗ: Дијаграм секвенце случаја коришћења – Претраживање корисника

Основни сценарио СК

1. Запослени позива систем да нађе кориснике по задатој вредности. (АПСО)
2. Систем приказује запосленом податке о корисницима и поруку: „Систем је нашао кориснике по задатој вредности“. (ИА)
3. Запослени позива систем да учита корисника. (АПСО)
4. Систем приказује запосленом податке о кориснику и поруку: „Систем је учитао корисника“. (ИА)



Слика 17 - Дијаграм секвенце случаја коришћења – Претраживање корисника (основни сценарио)

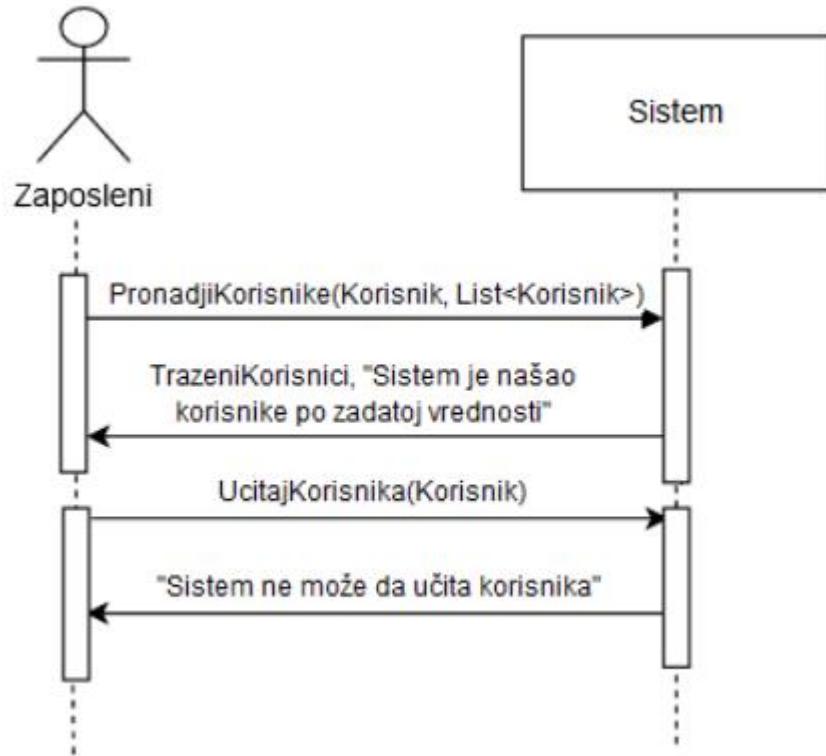
Алтернативна сценарија

- 2.1 Уколико систем не може да нађе кориснике, приказује запосленом поруку: „Систем не може да нађе кориснике по задатој вредности“. Прекида се извршење сценарија. (ИА)



Слика 18 – Дијаграм секвенце случаја коришћења – Претраживање корисника (алтернативни сценарио 1)

4.1 Уколико **систем** не може да учита **корисника**, приказује **запосленом** поруку: „**Систем не може да учита корисника**“. (ИА)



Слика 19 - Дијаграм секвенце случаја коришћења – Претраживање корисника (алтернативни сценарио 2)

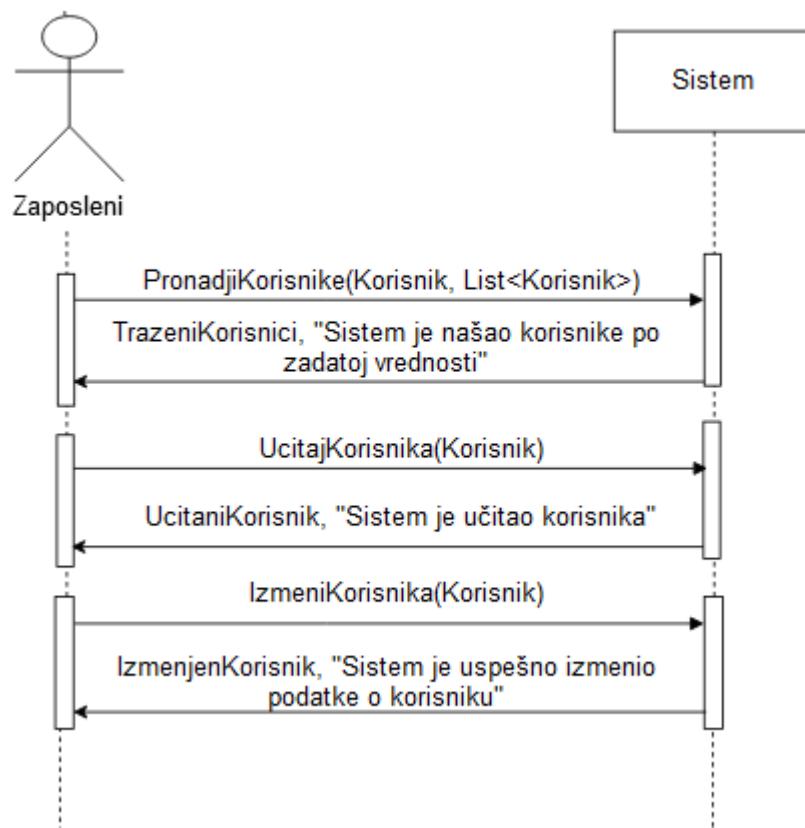
Са наведених дијаграма секвенци уочавају се две системске операције:

1. signal **PronadjiKorisnike(Korisnik, List<Korisnik>)**
2. signal **UcitajKorisnika(Korisnik)**

ДС4: Дијаграм секвенце случаја коришћења – Измена података о кориснику

Основни сценарио СК

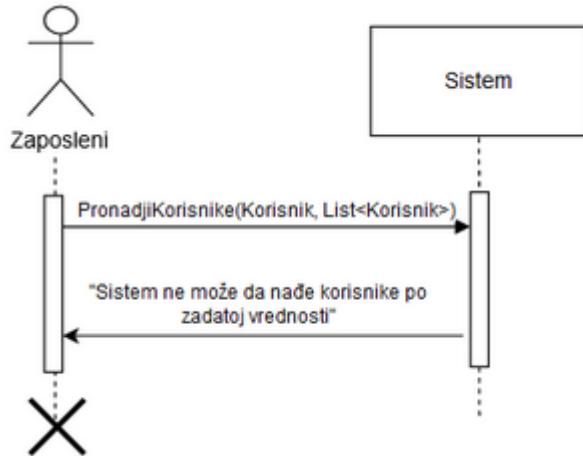
1. Запослени позива систем да нађе кориснике по задатој вредности. (АПСО)
2. Систем приказује запосленом податке о корисницима и поруку: „Систем је нашао кориснике по задатој вредности“. (ИА)
3. Запослени позива систем да учита корисника. (АПСО)
4. Систем приказује запосленом податке о кориснику и поруку: „Систем је учитао корисника“. (ИА)
5. Запослени позива систем да измени податке о кориснику. (АПСО)
6. Систем приказује запосленом изменjenog корисника и поруку: „Систем је успешно изменио податке о кориснику.“ (ИА)



Слика 20 - Дијаграм секвенце случаја коришћења – Измена података о кориснику (основни сценарио)

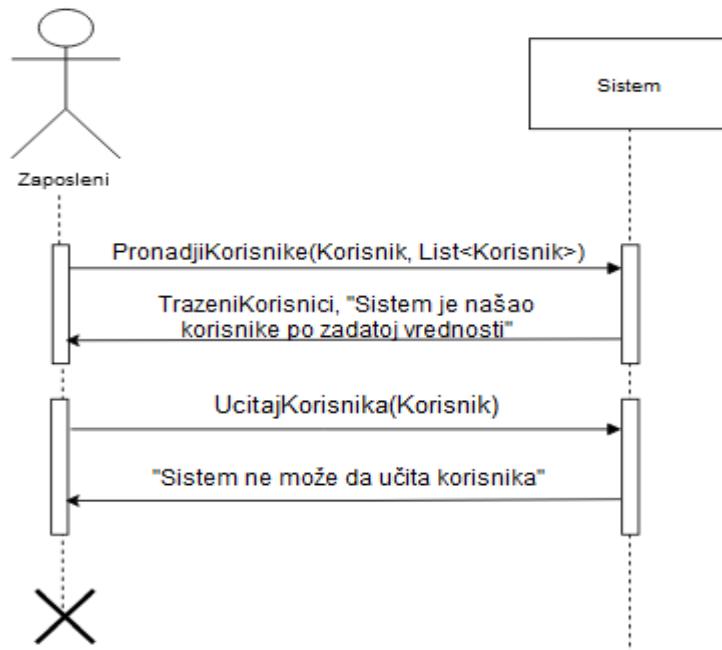
Алтернативна сценарија

2.1 Уколико систем не може да нађе кориснике, он приказује запосленом поруку: „Систем не може да нађе кориснике по задатој вредности“. Прекида се извршење сценарија. (ИА)



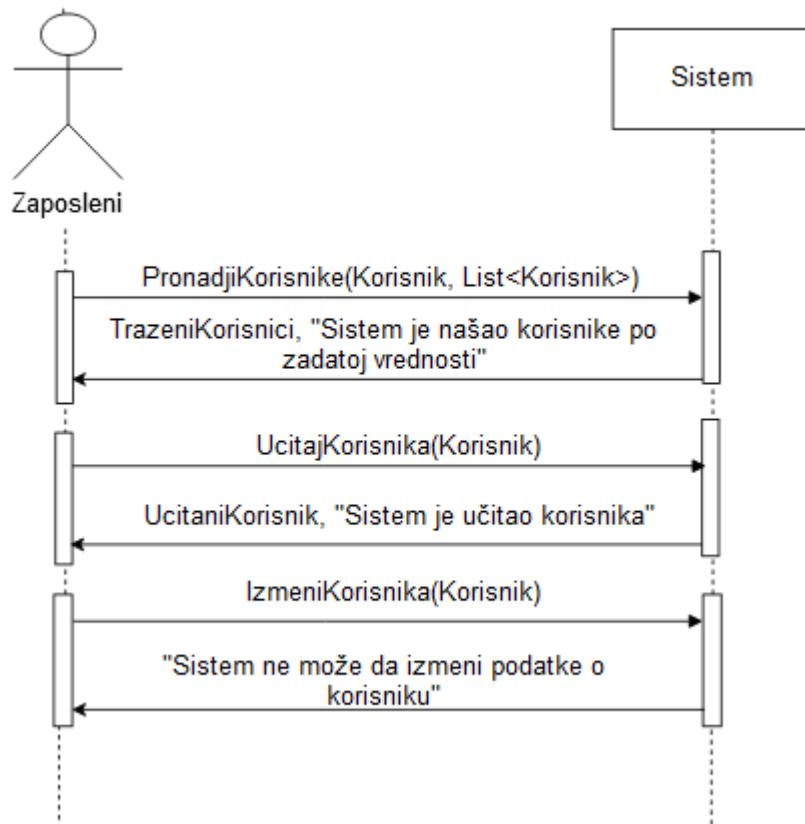
Слика 21 - Дијаграм секвенце случаја коришћења – Измена података о кориснику (алтернативни сценарио 1)

4.1 Уколико систем не може да учита податке о кориснику, он приказује запосленом поруку „Систем не може да учита корисника“. Прекида се извршење сценарија. (ИА)



Слика 22 - Дијаграм секвенце случаја коришћења – Измена података о кориснику (алтернативни сценарио 2)

6.1 Уколико **систем** не може да измени податке о **кориснику**, он приказује **запосленом** поруку: „**Систем** не може да измени податке о **кориснику**“. (ИА)



Слика 23 - Дијаграм секвенце случаја коришћења – Измена података о кориснику (алтернативни сценарио 3)

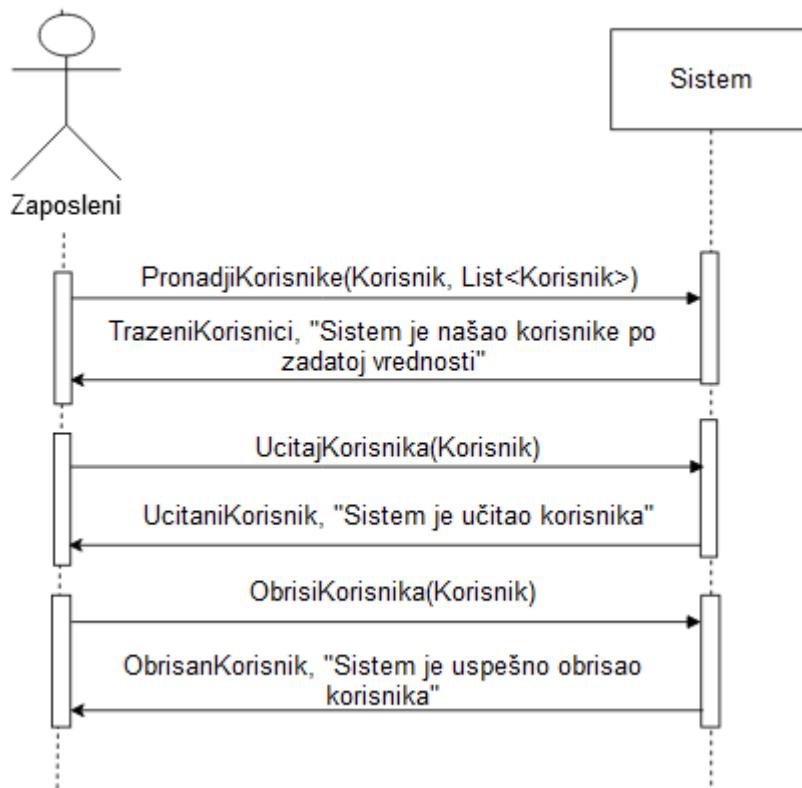
Са наведених дијаграма секвенци уочавају се три системске операције:

1. signal **PronadjiKorisnike(Korisnik, List<Korisnik>)**
2. signal **UcitajKorisnika(Korisnik)**
3. signal **IzmeniKorisnika(Korisnik)**

ДС5: Дијаграм секвенце случаја коришћења – Брисање корисника

Основни сценарио СК

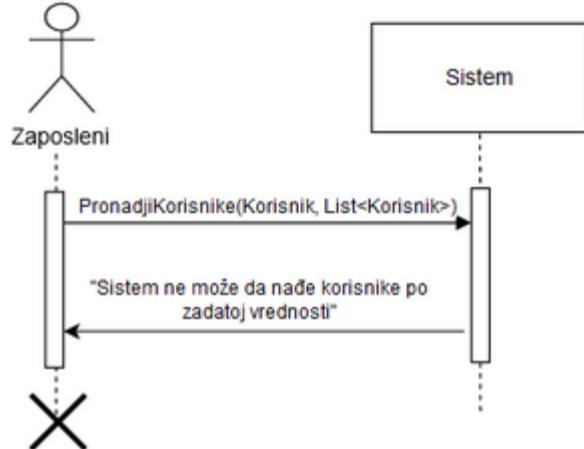
1. Запослени позива систем да нађе кориснике по задатој вредности. (АПСО)
2. Систем приказује запосленом податке о корисницима и поруку: „Систем је нашао кориснике по задатој вредности“. (ИА)
3. Запослени позива систем да учита корисника. (АПСО)
4. Систем приказује запосленом податке о кориснику и поруку: „Систем је учитао корисника“. (ИА)
5. Запослени позива систем да обрише корисника. (АПСО)
6. Систем приказује запосленом поруку: „Систем је успешно обрисао корисника.“ (ИА)



Слика 24 - Дијаграм секвенце случаја коришћења –Брисање корисника (основни сценарио)

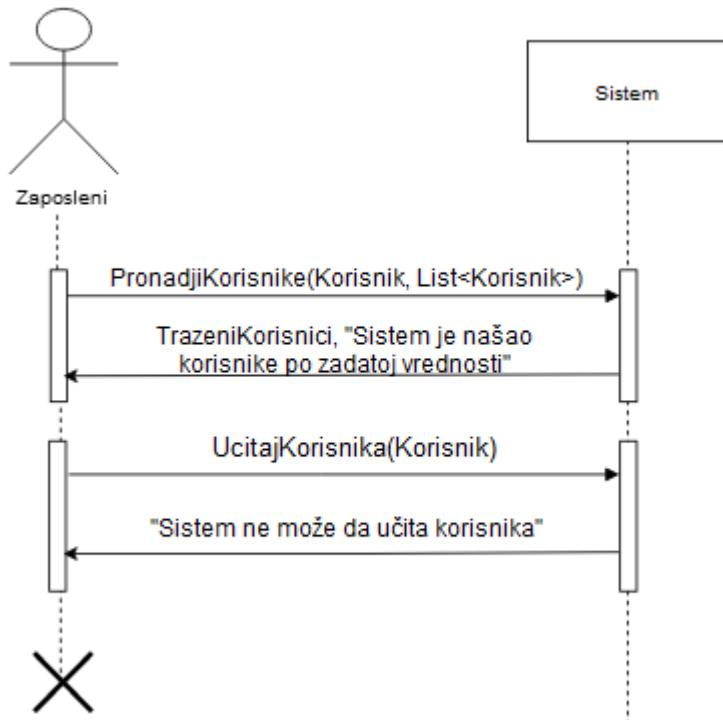
Алтернативна сценарија

2.1 Уколико **систем** не може да нађе **кориснике**, он приказује **запосленом** поруку: „**Систем** не може да нађе **кориснике** по задатој вредности”. Прекида се извршење сценарија. (ИА)



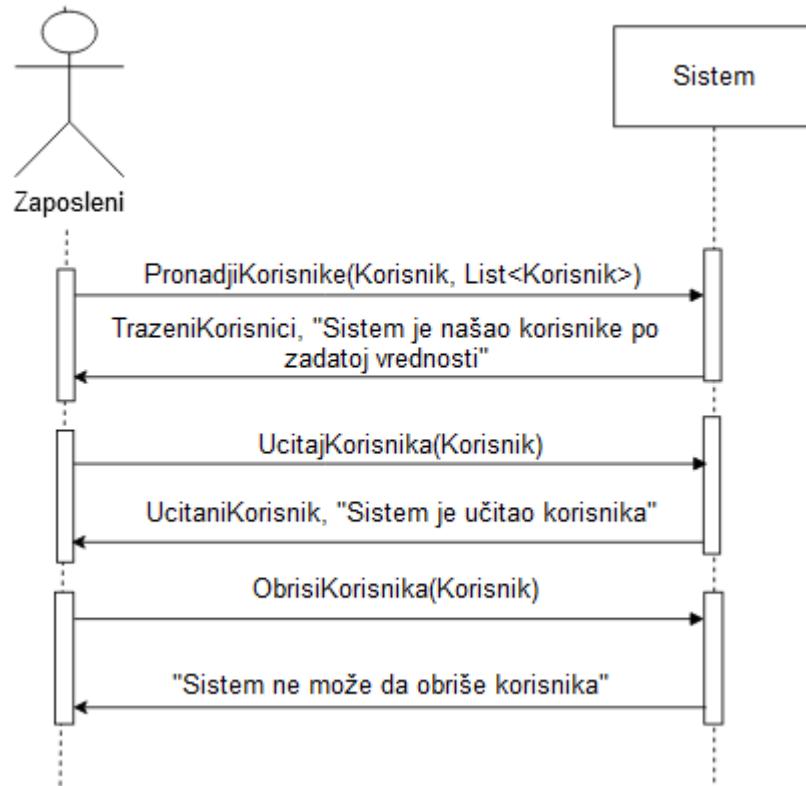
Слика 25 - Дијаграм секвенце случаја коришћења – Брисање корисника (алтернативни сценарио 1)

4.1 Уколико **систем** не може да учита податке о **кориснику**, он приказује **запосленом** поруку „**Систем** не може да учита **корисника**”. Прекида се извршење сценарија. (ИА)



Слика 26 - Дијаграм секвенце случаја коришћења – Брисање корисника (алтернативни сценарио 2)

6.1 Уколико **систем** не може да обрише **корисника**, он приказује **запосленом** поруку: „**Систем** не може да обрише **корисника**“. (ИА)



Слика 27 - Дијаграм секвенце случаја коришћења –Брисање корисника (алтернативни сценарио 3)

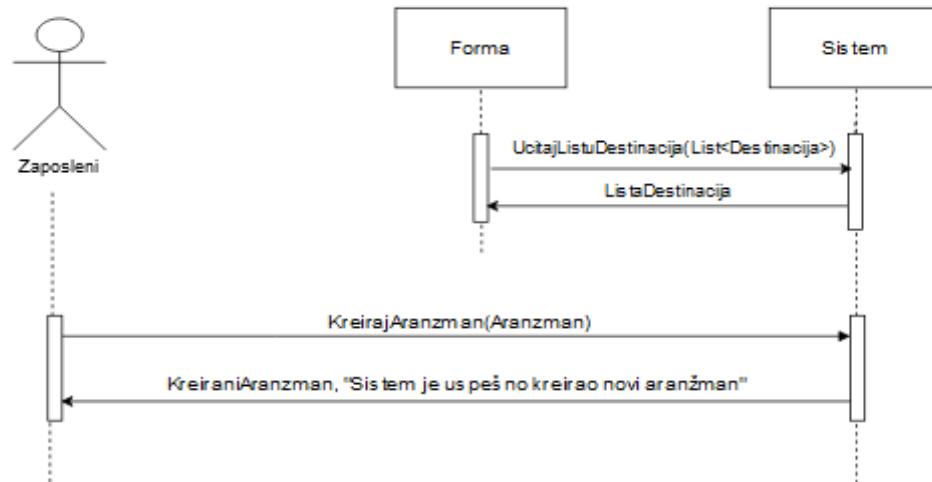
Са наведених дијаграма секвенци уочавају се три системске операције:

1. signal **PronadjiKorisnike(Korisnik, List<Korisnik>)**
2. signal **UcitajKorisnika(Korisnik)**
3. signal **Obrisikorisnika(Korisnik)**

ДС6: Дијаграм секвенце случаја коришћења – Креирање аранжмана

Основни сценарио СК

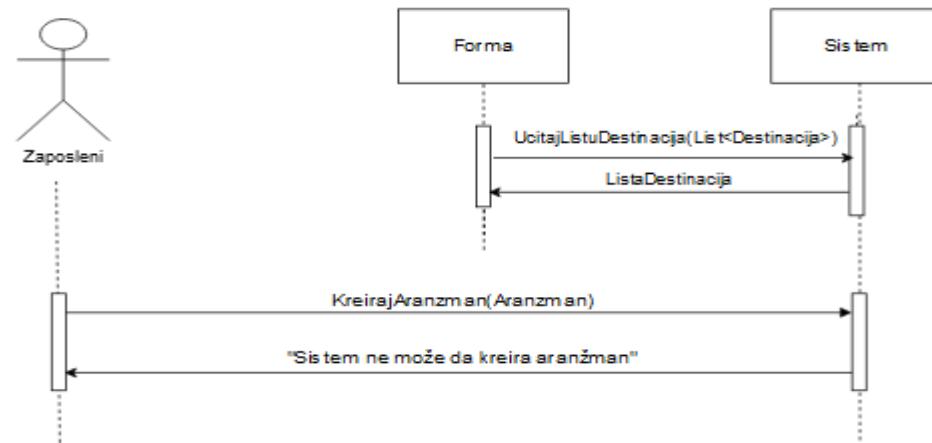
1. **Форма** позива **систем** да учита листу дестинација. (АПСО)
2. **Систем врaћa** **форми** листу дестинација. (ИА)
3. **Запослени** позива **систем** да креира нови аранжман. (АПСО)
4. **Систем приказујe** **запосленом** креирани аранжман и поруку: „**Систем је успешно креирао нови аранжман**“. (ИА)



Слика 28 - Дијаграм секвенце случаја коришћења – Креирање аранжмана (основни сценарио)

Алтернативна сценарија

- 4.1 Уколико **систем** не може да креира **аранжман**, он приказује **запосленом** поруку: „**Систем** не може да креира **аранжман**“. (ИА)



Слика 29 - Дијаграм секвенце случаја коришћења – Креирање аранжмана (алтернативни сценарио 1)

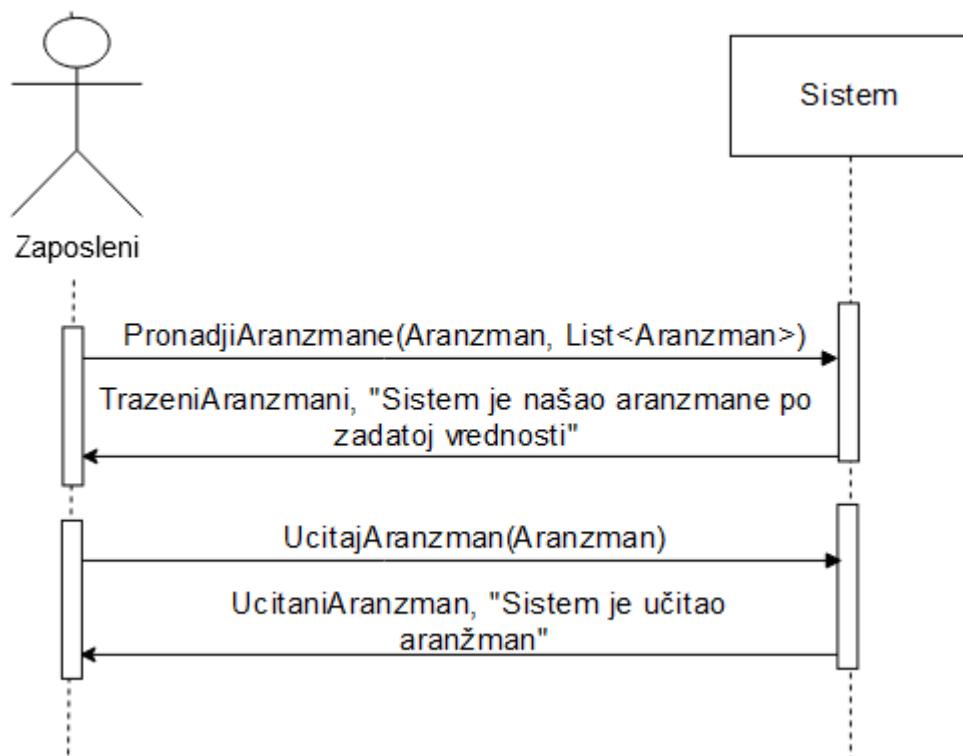
Са наведених дијаграма секвенци уочава се једна системска операција:

1. signal **KreirajAranzman(Aranzman)**

ДС7: Дијаграм секвенце случаја коришћења – Претраживање аранжмана

Основни сценарио СК

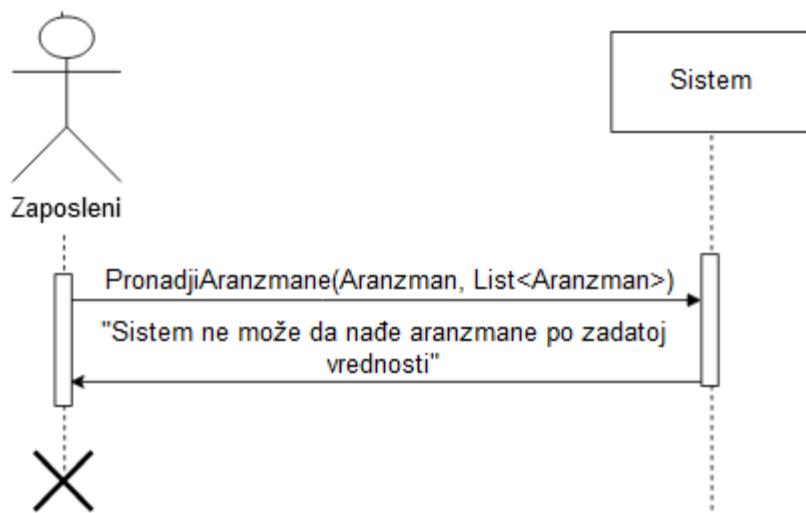
1. Запослени позива систем да нађе аранжмане по задатој вредности. (АПСО)
2. Систем приказује запосленом податке о аранжманима и поруку: „Систем је нашао аранжмане по задатој вредности”. (ИА)
3. Запослени позива систем да учита аранжман. (АПСО)
4. Систем приказује запосленом податке о аранжману и поруку: „Систем је учитао аранжман”. (ИА)



Слика 31 – Дијаграм секвенце случаја коришћења – Претраживање аранжмана (основни сценарио)

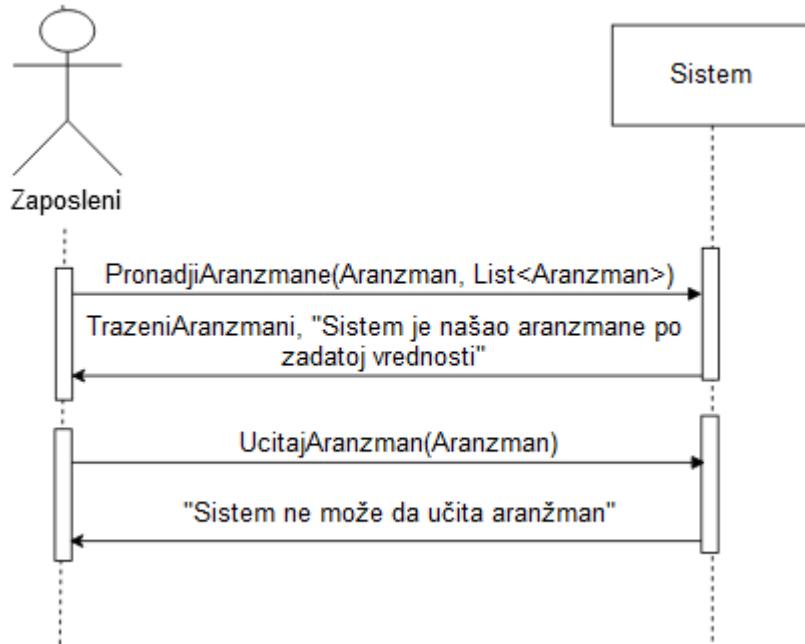
Алтернативна сценарија

- 2.1 Уколико систем не може да нађе аранжмане, он приказује запосленом поруку: „Систем не може да нађе аранжмане по задатој вредности”. Прекида се извршење сценарија. (ИА)



Слика 32 - Дијаграм секвенце случаја коришћења – Претраживање аранжмана (алтернативни сценарио 1)

4.1 Уколико **систем** не може да учита **аранжман**, он приказује **запосленом** поруку: „**Систем** не може да учита **аранжман**”. (ИА)



Слика 33 - Дијаграм секвенце случаја коришћења – Претраживање аранжмана (алтернативни сценарио 2)

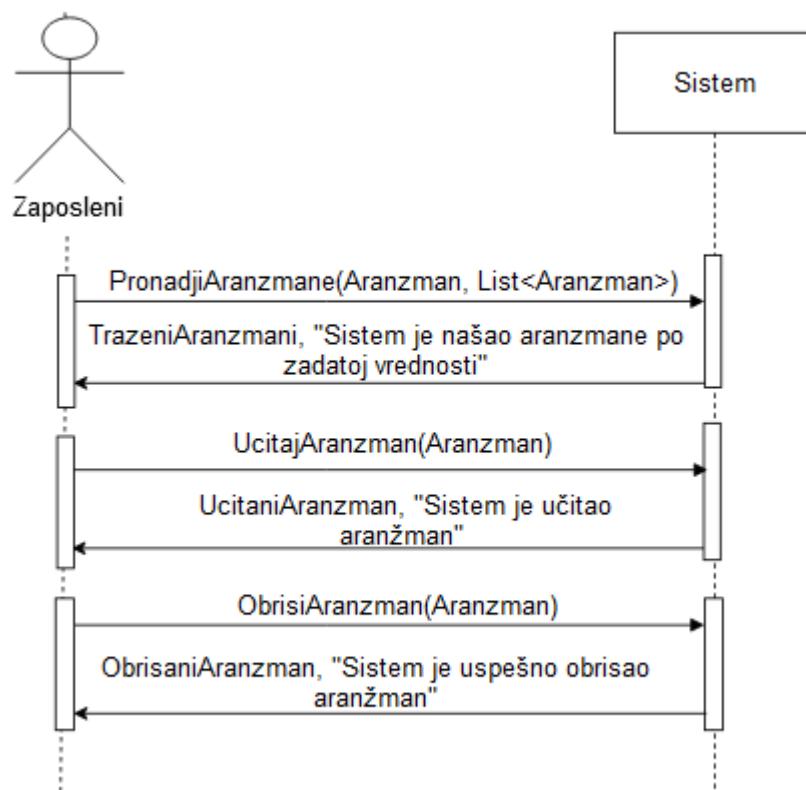
Са наведених дијаграма секвенци уочавају се две системске операције:

1. signal **PronadjiAranzmane(Aranzman, List<Aranzman>)**
2. signal **UcitajAranzman(Aranzman)**

ДС8: Дијаграм секвенце случаја коришћења – Брисање аранжмана

Основни сценарио СК

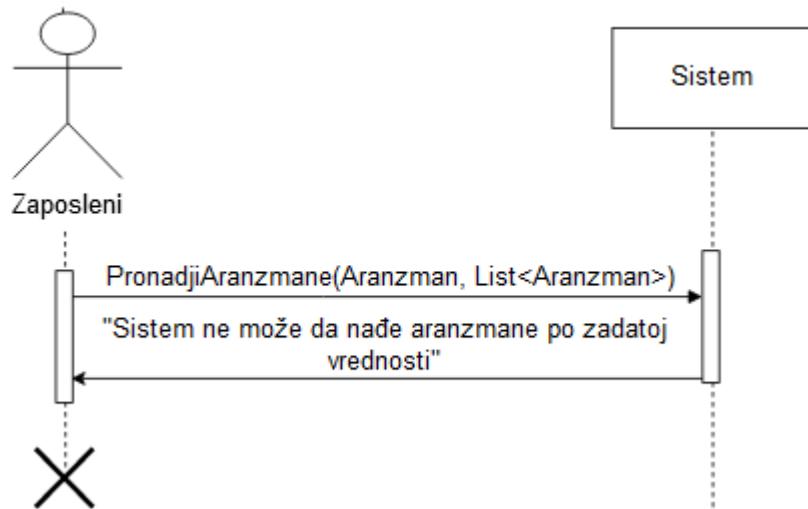
1. Запослени позива систем да нађе аранжмане по задатој вредности. (АПСО)
2. Систем приказује запосленом податке о аранжманима и поруку: „Систем је нашао аранжмане по задатој вредности“. (ИА)
3. Запослени позива систем да учита аранжман. (АПСО)
4. Систем приказује запосленом податке о аранжману поруку: „Систем је учитао аранжман“. (ИА)
5. Запослени позива систем да обрише аранжман. (АПСО)
6. Систем приказује запосленом поруку: „Систем је успешно обрисао аранжман.“ (ИА)



Слика 34 - Дијаграм секвенце случаја коришћења –Брисање аранжмана (основни сценарио)

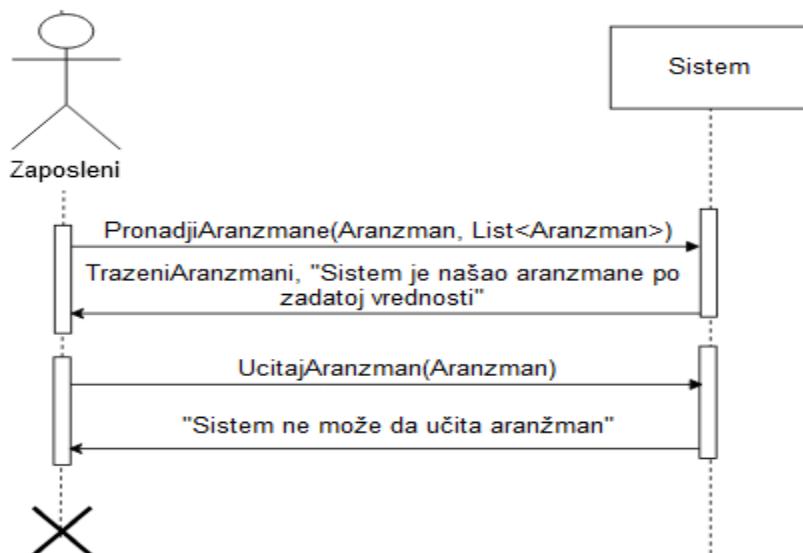
Алтернативна сценарија

2.1 Уколико **систем** не може да нађе аранжмане, он приказује **запосленом** поруку: „**Систем** не може да нађе аранжмане по задатој вредности”. Прекида се извршење сценарија. (ИА)



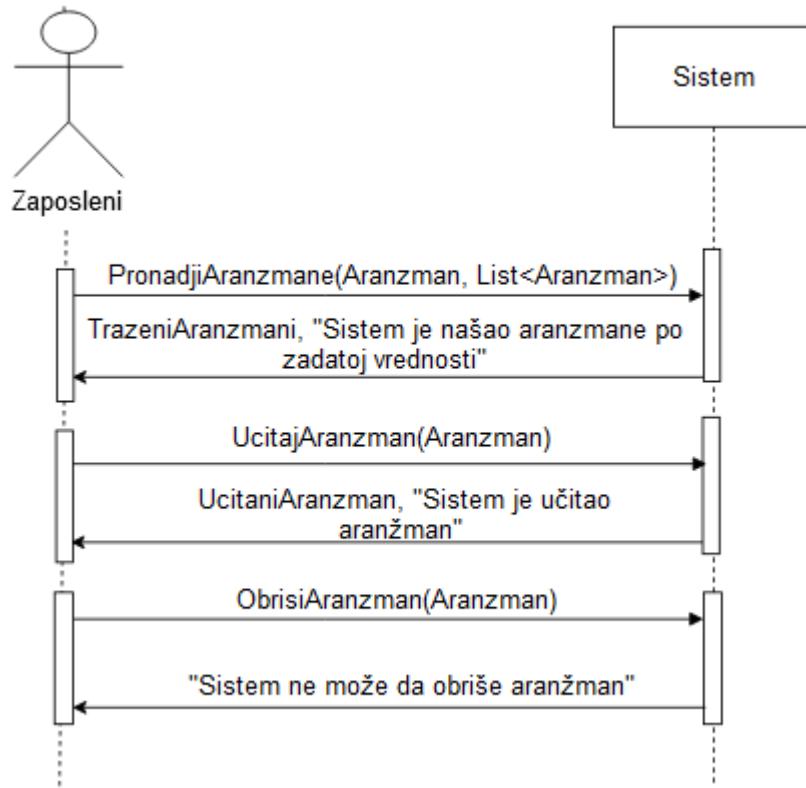
Слика 35 - Дијаграм секвенце случаја коришћења –Брисање аранжмана аранжмана (алтернативни сценарио 1)

4.1 Уколико **систем** не може да учита податке о **аранжману**, он приказује **запосленом** поруку „**Систем** не може да учита **аранжман**”. Прекида се извршење сценарија. (ИА)



Слика 36 - Дијаграм секвенце случаја коришћења –Брисање аранжмана аранжмана (алтернативни сценарио 2)

6.1 Уколико **систем** не може да обрише аранжман, он приказује **запосленом** поруку „**Систем** не може да обрише аранжман”. (ИА)



Слика 37 - Дијаграм секвенце случаја коришћења – Брисање аранжмана аранжмана (алтернативни сценарио 3)

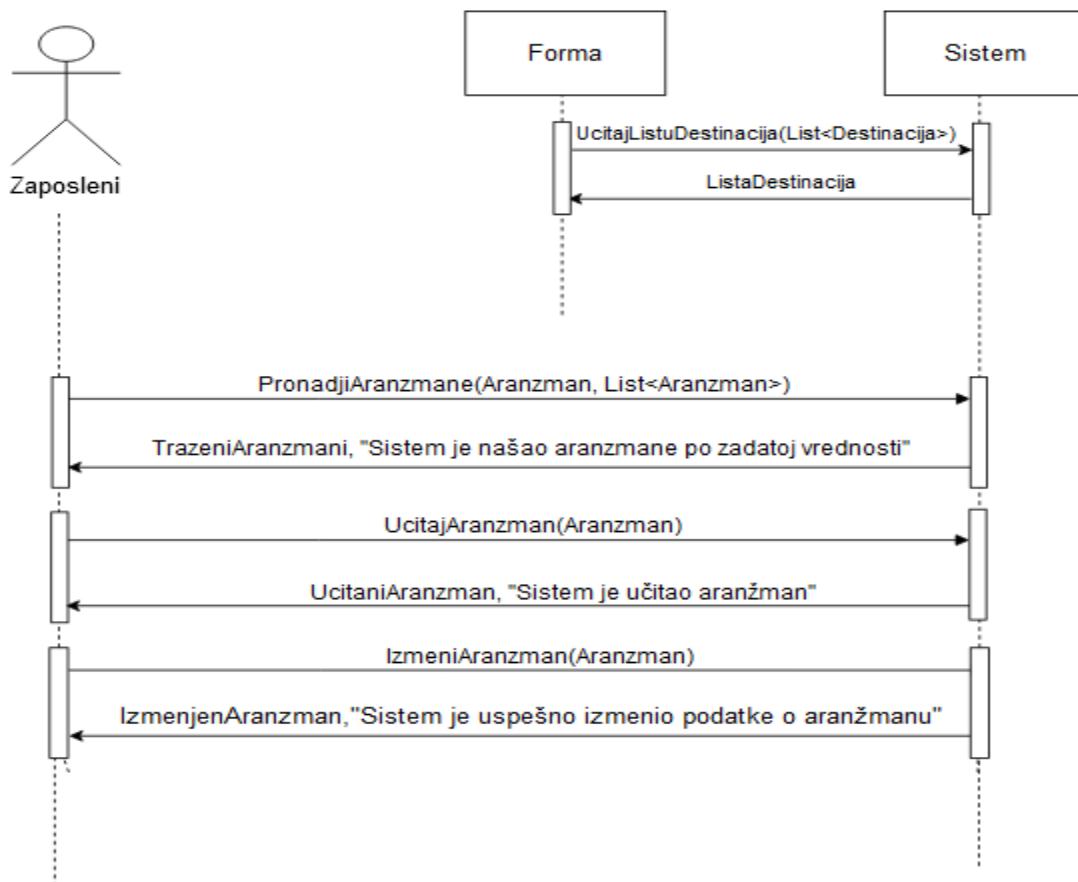
Са наведених дијаграма секвенци уочавају се три системске операције:

1. signal **PronadjiAranzmane(Aranzman, List<Aranzman>)**
2. signal **UcitajAranzman(Aranzman)**
3. signal **ObrisniAranzman(Aranzman)**

ДС9: Случај коришћења – Измена података о аранжману

Основни сценарио СК

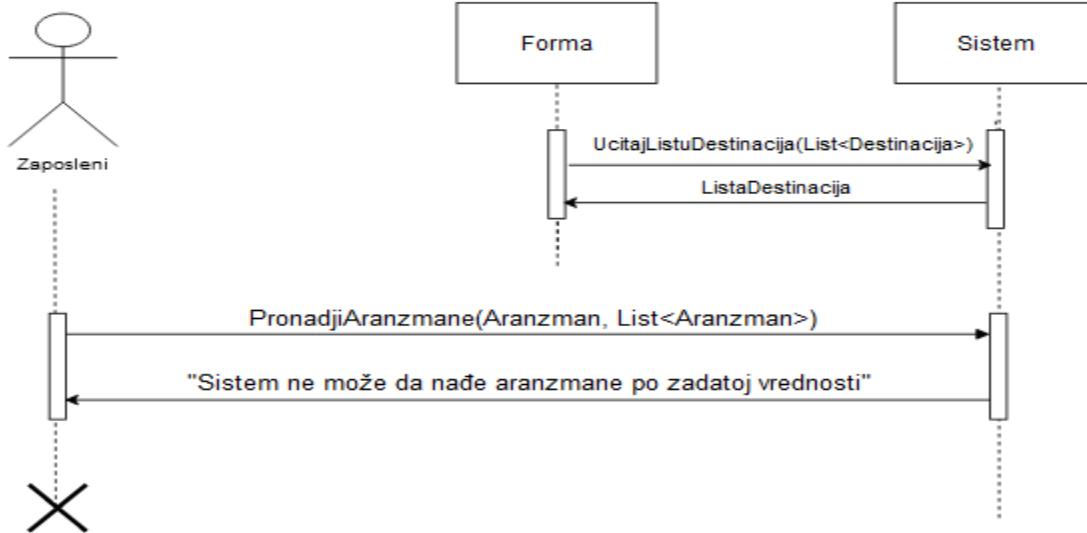
1. **Форма** позива **систем** да учита листу дестинација. (АПСО)
2. **Систем** враћа **форми** листу дестинација. (ИА)
3. **Запослени** позива **систем** да нађе аранжмане по задатој вредности. (АПСО)
4. **Систем** приказује запосленом податке о аранжманима и поруку: „**Систем** је нашао аранжмане по задатој вредности”. (ИА)
5. **Запослени** позива **систем** да учита аранжман. (АПСО)
6. **Систем** приказује запосленом податке о аранжману и поруку: „**Систем** је учитао аранжман”. (ИА)
7. **Запослени** позива **систем** да измени податке о аранжману. (АПСО)
8. **Систем** приказује запосленом изменењи аранжман и поруку: „**Систем** је успешно изменио податке о аранжману.” (ИА)



Слика 38 - Дијаграм секвенце случаја коришћења - Измена података о аранжману (основни сценарио)

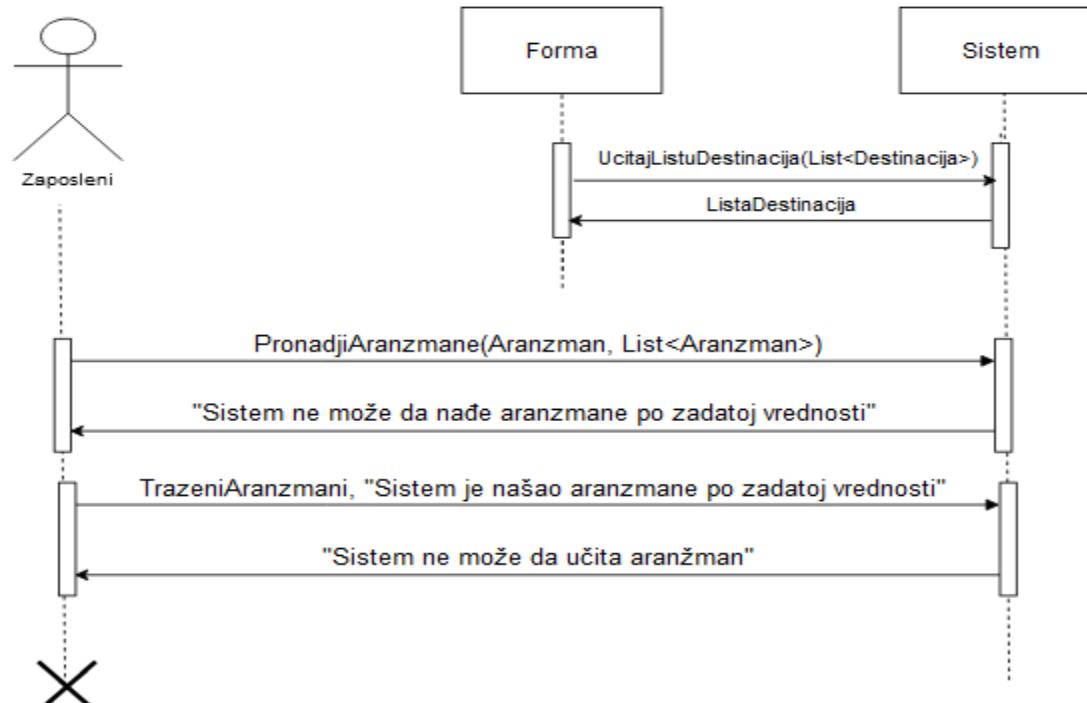
Алтернативна сценарија

4.1 Уколико **систем** не може да нађе **аранжмане**, он приказује **запосленом** поруку: „**Систем** не може да нађе **аранжмане** по задатој вредности”. Прекида се извршење сценарија. (ИА)



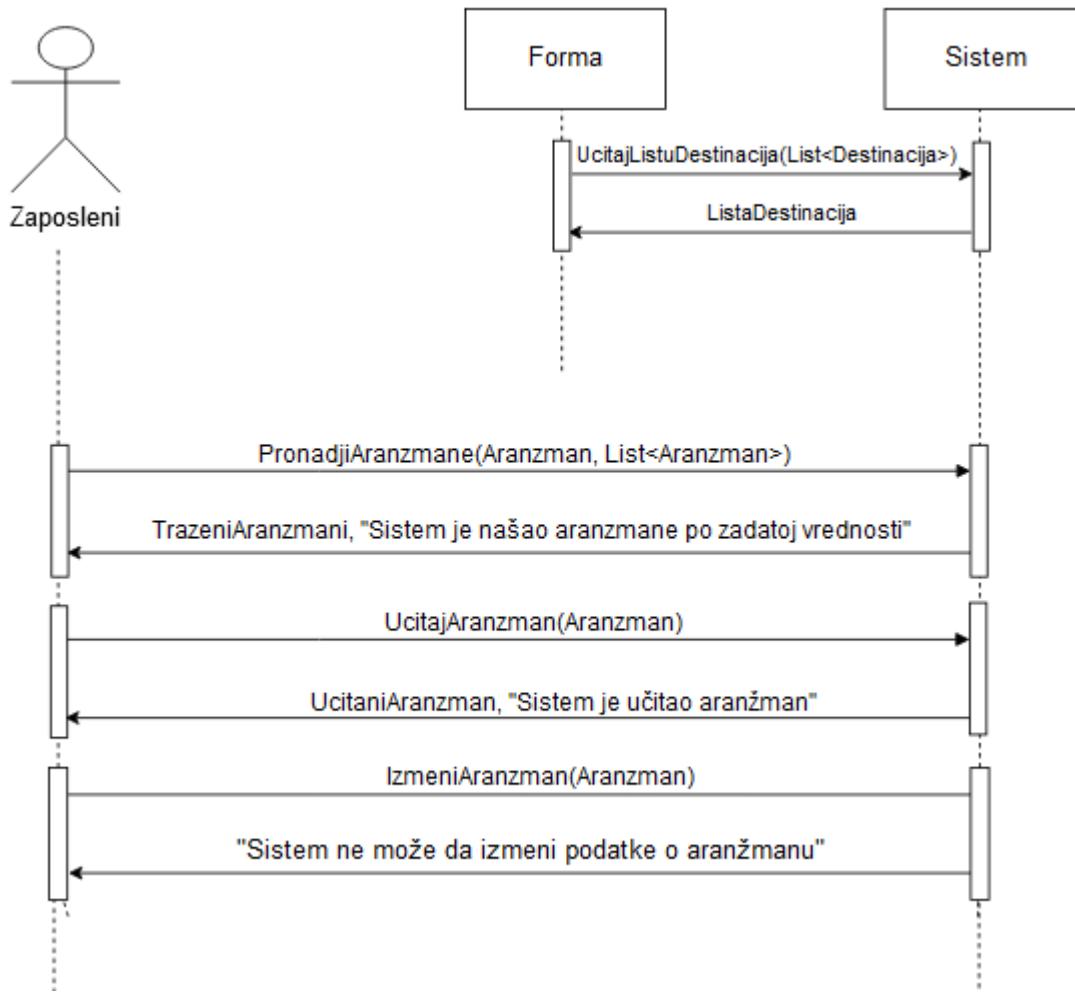
Слика 39 - Дијаграм секвенце случаја коришћења - Измена података о аранжману (алтернативни сценарио 1)

6.1 Уколико **систем** не може да учира **аранжман**, он приказује **запосленом** поруку: „**Систем** не може да учира **аранжман**”. Прекида се извршење сценарија. (ИА) (ИА)



Слика 40 - Дијаграм секвенце случаја коришћења - Измена података о аранжману (алтернативни сценарио 2)

8.1 Уколико **систем** не може да измени податке о **аранжману**, он приказује **запосленом** поруку: „**Систем** не може да измени податке о **аранжману**. (ИА)



Слика 41 - Дијаграм секвенце случаја коришћења - Измена података о аранжману (алтернативни скенарио 3)

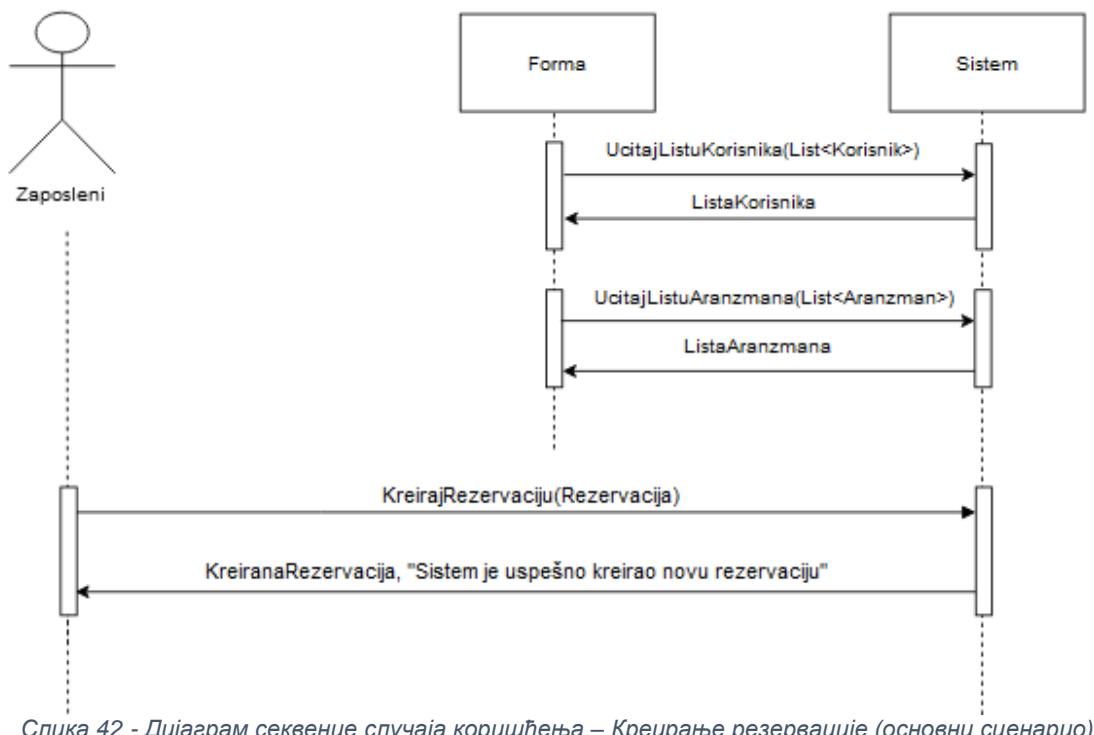
Са наведених дијаграма секвенци уочавају се четири системске операције:

1. signal **PronadjiAranzmane(Aranzman, List<Aranzman>)**
2. signal **UcitajAranzman(Aranzman)**
3. signal **IzmeniAranzman(Aranzman)**
4. signal **UcitajListuDestinacija (List<Aranzman>)**

ДС10: Дијаграм секвенце случаја коришћења – Креирање резервације

Основни сценарио СК

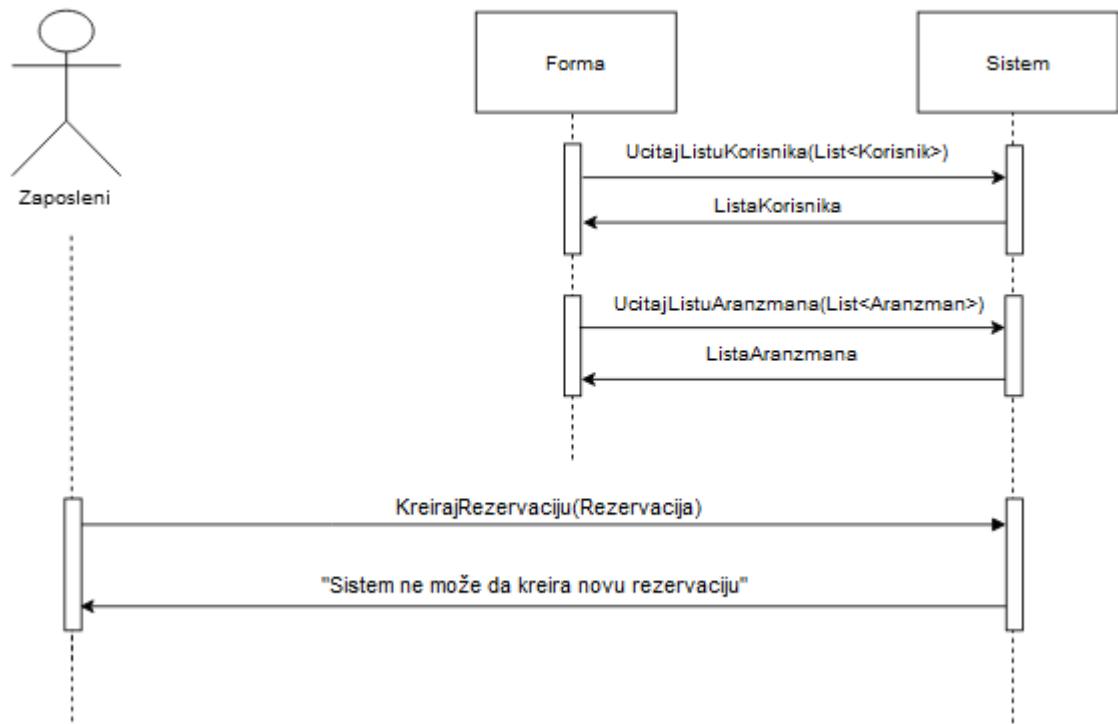
1. **Форма** позива **систем** да учита листу корисника. (АПСО)
2. **Систем** враћа **форми** листу корисника. (ИА)
3. **Форма** позива **систем** да учита листу аранжмана. (АПСО)
4. **Систем** враћа **форми** листу аранжмана. (ИА)
5. **Запослени** позива **систем** да креира нову резервацију. (АПСО)
6. **Систем** приказује запосленом креирану резервацију и поруку: „**Систем** је успешно креирао нову резервацију“. (ИА)



Слика 42 - Дијаграм секвенце случаја коришћења – Креирање резервације (основни сценарио)

Алтернативна сценарија

6.1. Уколико **систем** не може да креира нову **резервацију**, он приказује **запосленом** поруку „**Систем не може да креира нову резервацију**“. (ИА)



Слика 43 - Дијаграм секвенце случаја коришћења – Креирање резервације (алтернативни сценарио 1)

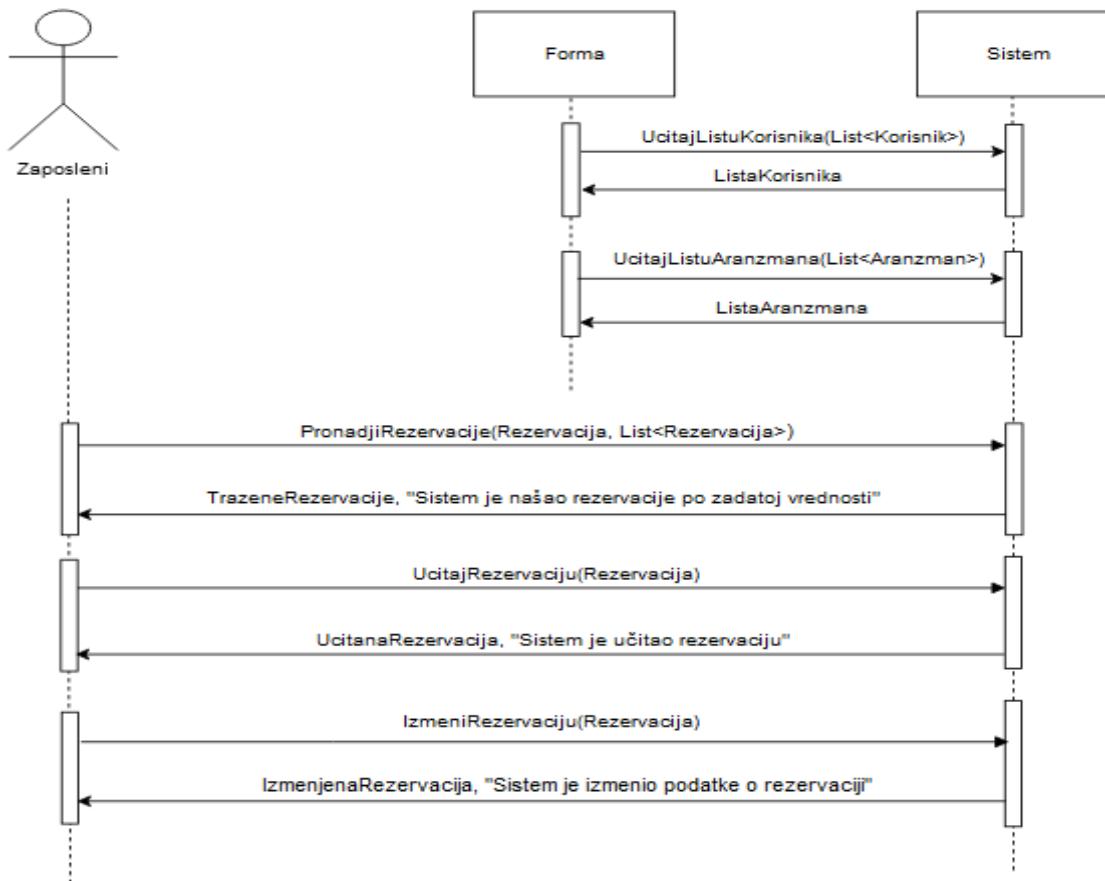
Са наведених дијаграма секвенци уочавају се четири системске операције:

1. signal **UcitajListuKorisnika(List<Korisnik>)**
2. signal **UcitajListuAranzmanu(List<Aranzman>)**
3. signal **KreirajRezervaciju(Rezervacija)**

ДС11: Дијаграм секвенце случаја коришћења – Измена резервације

Основни сценарио СК

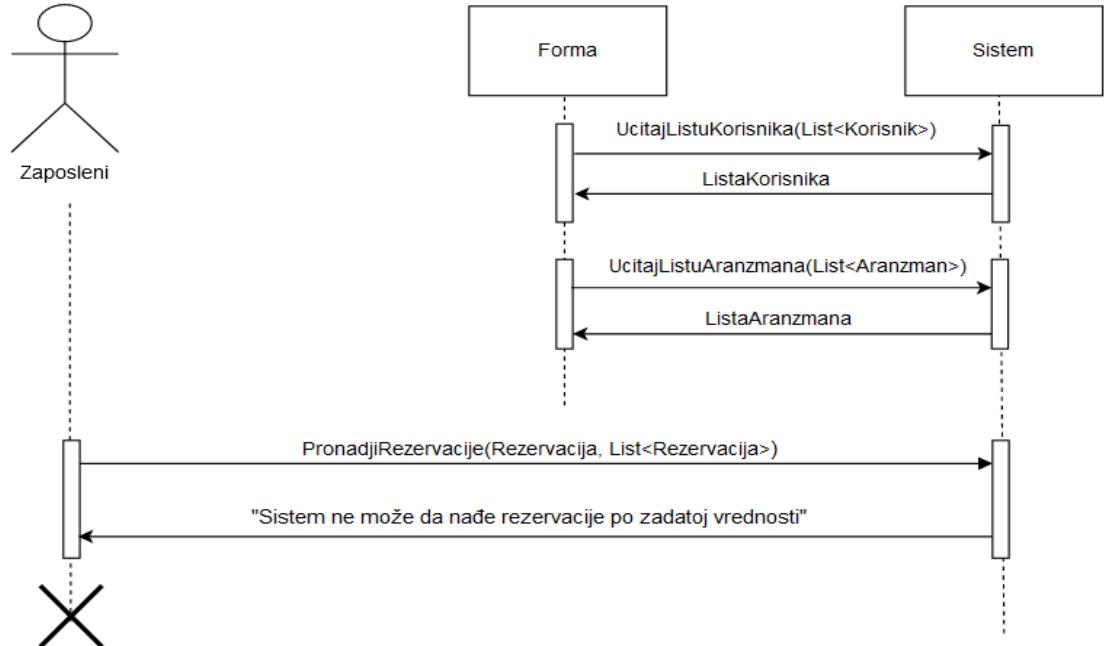
1. **Форма** позива **систем** да учита листу корисника. (АПСО)
2. **Систем** враћа **форми** листу корисника. (ИА)
3. **Форма** позива **систем** да учита листу аранжмана. (АПСО)
4. **Систем** враћа **форми** листу аранжмана. (ИА)
5. **Запослени** позива **систем** да нађе резервације по задатој вредности. (АПСО)
6. **Систем** приказује запосленом податке о резервацијама и поруку: „**Систем** је нашао резервације по задатој вредности”. (ИА)
7. **Запослени** позива **систем** да учита резервацију. (АПСО)
8. **Систем** приказује запосленом податке о резервацији и поруку: „**Систем** је учитао резервацију”. (ИА)
9. **Запослени** позива **систем** да измени податке о резервацији. (АПСО)
10. **Систем** приказује запосленом изменјену резервацију и поруку: „**Систем** је успешно изменио податке о резервацији.” (ИА)



Слика 44 - Дијаграм секвенце случаја коришћења – Измена резервације (основни сценарио)

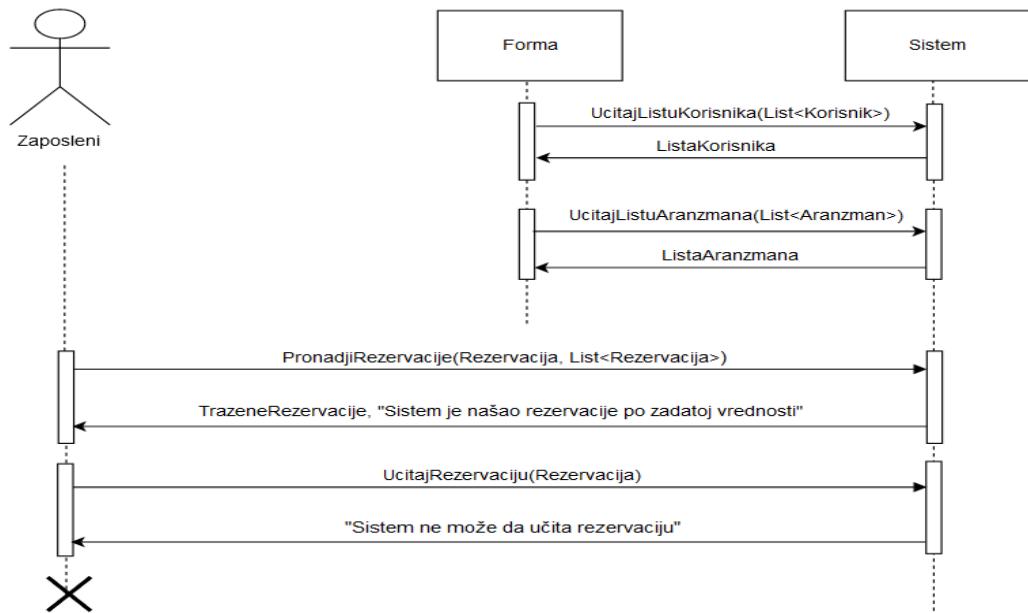
Алтернативна сценарија

6.1 Уколико **систем** не може да нађе **резервације**, он приказује **запосленом** поруку: „**Систем** не може да нађе **резервације** по задатој вредности”. Прекида се извршење сценарија. (ИА)



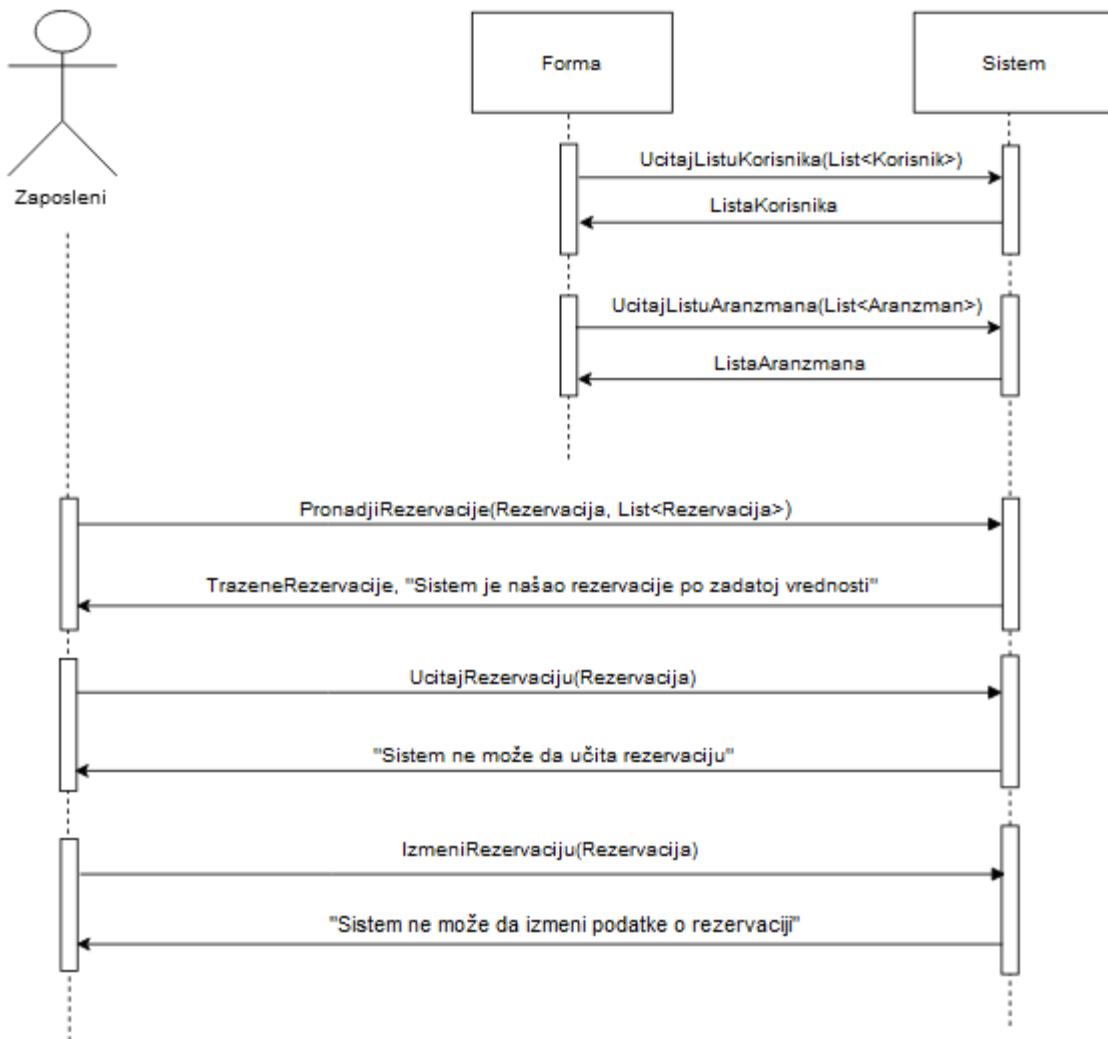
Слика 45 - Дијаграм секвенце случаја коришћења – Измена резервације (алтернативни сценарио 1)

8.1 Уколико **систем** не може да учита податке о **резервацији**, он приказује **запосленом** поруку „**Систем** не може да учита **резервацију**”. Прекида се извршење сценарија. (ИА)



Слика 46 - Дијаграм секвенце случаја коришћења – Измена резервације (алтернативни сценарио 2)

10.1 Уколико систем не може да измени податке о резервацији, он приказује **запосленом** поруку „Систем не може да измени податке о резервацији”. (ИА)



Слика 47 - Дијаграм секвенце случаја коришћења – Измена резервације (алтернативни сценарио 3)

Са наведених дијаграма секвенци уочава се пет системских операција:

1. signal **UcitajListuKorisnika(List<Korisnik>)**
2. signal **UcitajListuAranzmanu(List<Aranzman>)**
3. signal **PronadjiRezervacije(Rezervacija, List<Rezervacija>)**
4. signal **UcitajRezervaciju(Rezervacija)**
5. signal **IzmeniRezervaciju(Rezervacija)**

На основу анализе добијено је 18 системских операција:

1. signal **PronadjiNalog(Nalog)**
2. signal **KreirajKorisnika(Korisnik)**
3. signal **UcitajListuKorisnika(List<Korisnik>)**
4. signal **PronadjiKorisnike(Korisnik, List<Korisnik>)**
5. signal **UcitajKorisnika(Korisnik)**
6. signal **IzmeniKorisnika(Korisnik)**
7. signal **ObrisniKorisnika(Korisnik)**
8. signal **KreirajAranzman(Aranzman)**
9. signal **UcitajListuAranzmana(List<Aranzman>)**
10. signal **UcitajListuDestinacija (List<Destinacija>)**
11. signal **PronadjiAranzmane(Aranzman, List<Aranzman>)**
12. signal **UcitajAranzman(Aranzman)**
13. signal **ObrisniAranzman(Aranzman)**
14. signal **IzmeniAranzman(Aranzman)**
15. signal **KreirajRezervaciju(Rezervacija)**
16. signal **PronadjiRezervacije(Rezervacija, List<Rezervacija>)**
17. signal **UcitajRezervaciju(Rezervacija)**
18. signal **IzmeniRezervaciju(Rezervacija)**

4.2.2. Понашање софтверског система – Дефинисање уговора о системским операцијама

Уговор УГ1: signal **PronadjiNalog(Nalog)**

Веза са СК: CK1

Предуслови: /

Постуслови: Налог је пронађен, запослени је пријављен на систем.

Уговор УГ2: signal **KreirajKorisnika(Korisnik)**

Веза са СК: CK2

Предуслови: Вредносна и структурна ограничења над објектом Korisnik морају да буду задовољена.

Постуслови: Направљен је нови корисник.

Уговор УГ3: signal **UcitajListuKorisnika(List<Korisnik>)**

Веза са СК: CK10, CK11

Предуслови: /

Постуслови: /

Уговор УГ4: signal **PronadjiKorisnike(Korisnik, List<Korisnik>)**

Веза са СК: CK3, CK4, CK5

Предуслови: /

Постуслови: /

Уговор УГ5: signal **UcitajKorisnika(Korisnik)**

Веза са СК: CK3, CK4, CK5

Предуслови: /

Постуслови: /

Уговор УГ6: signal **IzmeniKorisnika(Korisnik)**

Веза са СК: CK4

Предуслови: Вредносна и структурна ограничења над објектом Korisnik морају да буду задовољена.

Постуслови: Подаци о кориснику су изменjeni.

Уговор УГ7: signal **ObrisniKorisnika(Korisnik)**

Веза са СК: CK5

Предуслови: Структурна ограничења над објектом Korisnik морају да буду задовољена.

Постуслови: Корисник је обрисан.

Уговор УГ8: signal **KreirajAranzman(Aranzman)**

Веза са СК: CK6

Предуслови: Вредносна и структурна ограничења над објектом Aranzman морају да буду задовољена.

Постуслови: Направљен је нови аранжман.

Уговор УГ9: signal **UcitajListuAranzmana(List<Aranzman>)**

Веза са СК: CK10, CK11

Предуслови: /

Постуслови: /

Уговор УГ10: signal **UcitajListuDestinacija(List<Destinacija>)**

Веза са СК: CK6, CK9

Предуслови: /

Постуслови: /

Уговор УГ11: signal **PronadjiAranzmane(Aranzman, List<Aranzman>)**

Веза са СК: CK7, CK8, CK9

Предуслови: /

Постуслови: /

Уговор УГ12: signal **UcitajAranzman(Aranzman)**

Веза са СК: CK7, CK8, CK9

Предуслови: /

Постуслови: /

Уговор УГ13: signal **ObrisitiAranzman(Aranzman)**

Веза са СК: CK8

Предуслови: Структурна ограничења над објектом Aranzman морају да буду задовољена.

Постуслови: Аранжман је обрисан.

Уговор УГ14: signal **IzmeniAranzman(Aranzman)**

Веза са СК: CK9

Предуслови: Вредносна и структурна ограничења над објектом Aranzman морају да буду задовољена.

Постуслови: Подаци о аранжману су изменjeni.

Уговор УГ15: signal **KreirajRezervaciju(Rezervacija)**

Веза са СК: CK10

Предуслови: Вредносна и структурна ограничења над објектом Rezervacija морају да буду задовољена.

Постуслови: Направљена је нова резервација.

Уговор УГ16: signal **PronadjiRezervacije(Rezervacija, List<Rezervacija>)**

Веза са СК: CK11

Предуслови: /

Постуслови: /

Уговор УГ17: signal **UcitajRezervaciju(Rezervacija)**

Веза са СК: CK11

Предуслови: /

Постуслови: /

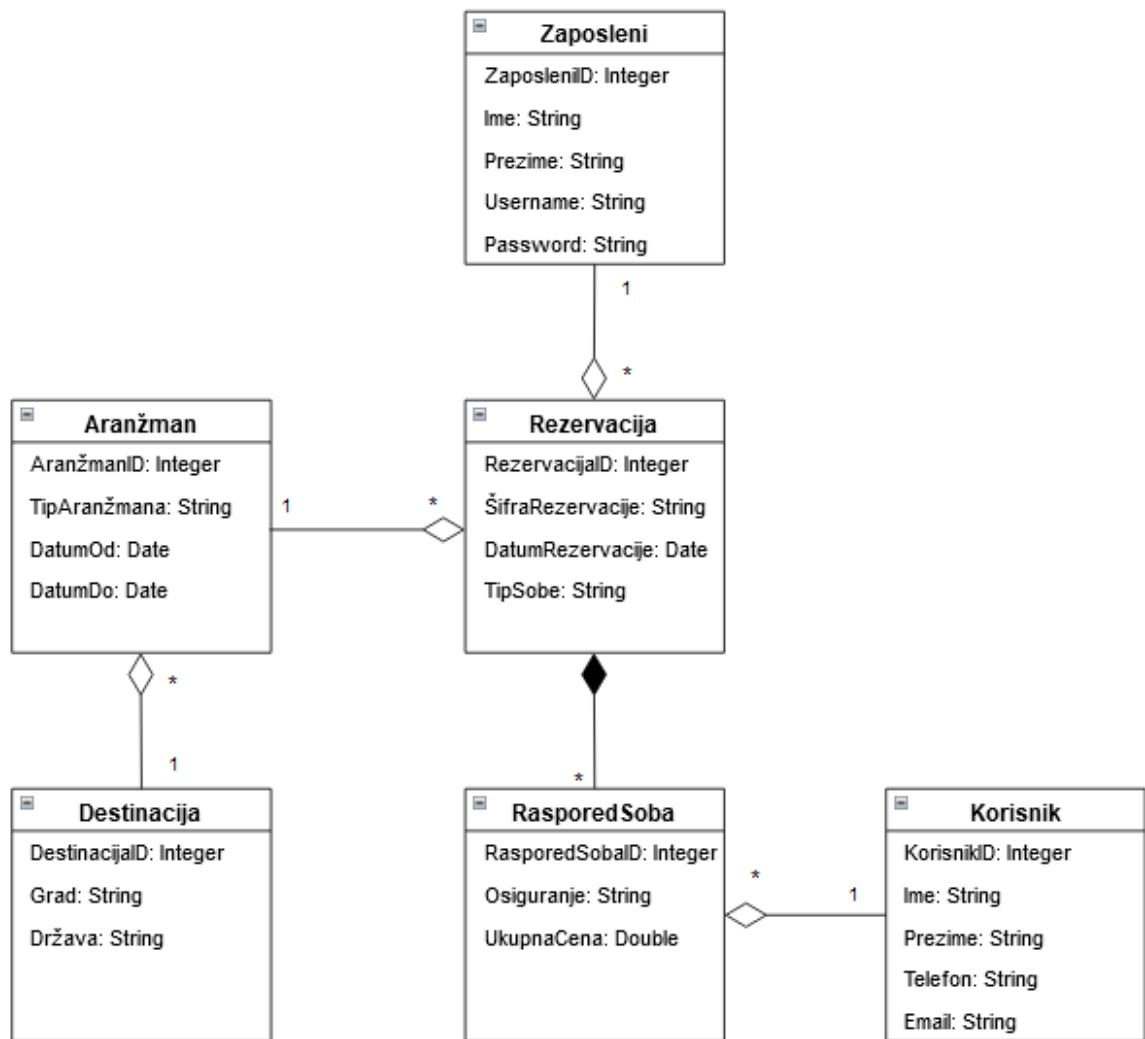
Уговор УГ18: signal **IzmeniRezervaciju(Rezervacija)**

Веза са СК: CK11

Предуслови: Вредносна и структурна ограничења над објектом Rezervacija морају да буду задовољена.

Постуслови: Подаци о резервацији су изменjeni.

4.2.3. Структура софтверског система – Концептуални (доменски) модел



Слика 38 - Концептуални модел

4.2.4. Структура софтверског система – Релациони модел

Korisnik(KorisnikID, Ime, Prezime, Telefon, Email)

Zaposleni(ZaposleniID, Ime, Prezime, Username, Password)

Destinacija(DestinacijaID, Grad, Država)

Aranžman(AranžmanID, TipAranžmana, DatumOd, DatumDo, *DestinacijaID*)

RasporedSoba(RasporedSobaID, RezervacijaID, Osiguranje, UkupnaCena, *KorisnikID*)

Rezervacija(RezervacijaID, ŠifraRezervacije, DatumRezervacije, TipSobe, *ZaposleniID*, *AranžmanID*)

Tabela Korisnik		Prosto vrednosno ograničenje		Složeno vrednosno ograničenje		Strukturno organičenje
Atributi	Ime	Tip atributa	Vrednost atributa	Međuzavisnost atributa jedne tabele	Međuzavisnost atributa više tabele	INSERT / UPDATE CASCADES RasporedSoba DELETE RESTRICTED RasporedSoba
	KorisnikID	Integer	not null and >0			
	Ime	String	not null			
	Prezime	String	not null			
	Telefon	String	not null			
	Email	String	not null			

Табела 1 – Korisnik

Tabela Zaposleni		Prosto vrednosno ograničenje		Složeno vrednosno ograničenje		Strukturno organičenje
Atributi	Ime	Tip atributa	Vrednost atributa	Međuzavisnost atributa jedne tabele	Međuzavisnost atributa više tabele	INSERT / UPDATE CASCADES Rezervacija DELETE RESTRICTED Rezervacija
	ZaposleniID	Integer	not null and >0			
	Ime	String	not null			
	Prezime	String	not null			
	Username	String	not null			
	Password	String	not null			

Табела 2 - Zaposleni

Tabela Destinacija		Prosto vrednosno ograničenje		Složeno vrednosno ograničenje		Strukturno organičenje
Atributi	Ime	Tip atributa	Vrednost atributa	Međuzavisnost atributa jedne tabele	Međuzavisnost atributa više tabele	INSERT / UPDATE CASCADES Aranžman DELETE RESTRICTED Aranžman
	DestinacijaID	Integer	not null and >0			
	Grad	String	not null			
	Država	String	not null			

Табела 3 - Destinacija

Tabela Aranžman		Prosto vrednosno ograničenje		Složeno vrednosno ograničenje		Strukturno organičenje
Atributi	Ime	Tip atributa	Vrednost atributa	Međuzavisnost atributa jedne tabele	Međuzavisnost atributa više tabele	INSERT RESTRICTED Destinacija UPDATE RESTRICTED Destinacija CASCADES Rezervacija DELETE RESTRICTED Rezervacija
	AranžmanID	Integer	not null and >0			
	TipAranžmana	String	not null			
	DatumOd	Date	not null			
	DatumDo	Date	not null			
	DestinacijaID	Integer	not null and >0			

Табела 4 – Aranžman

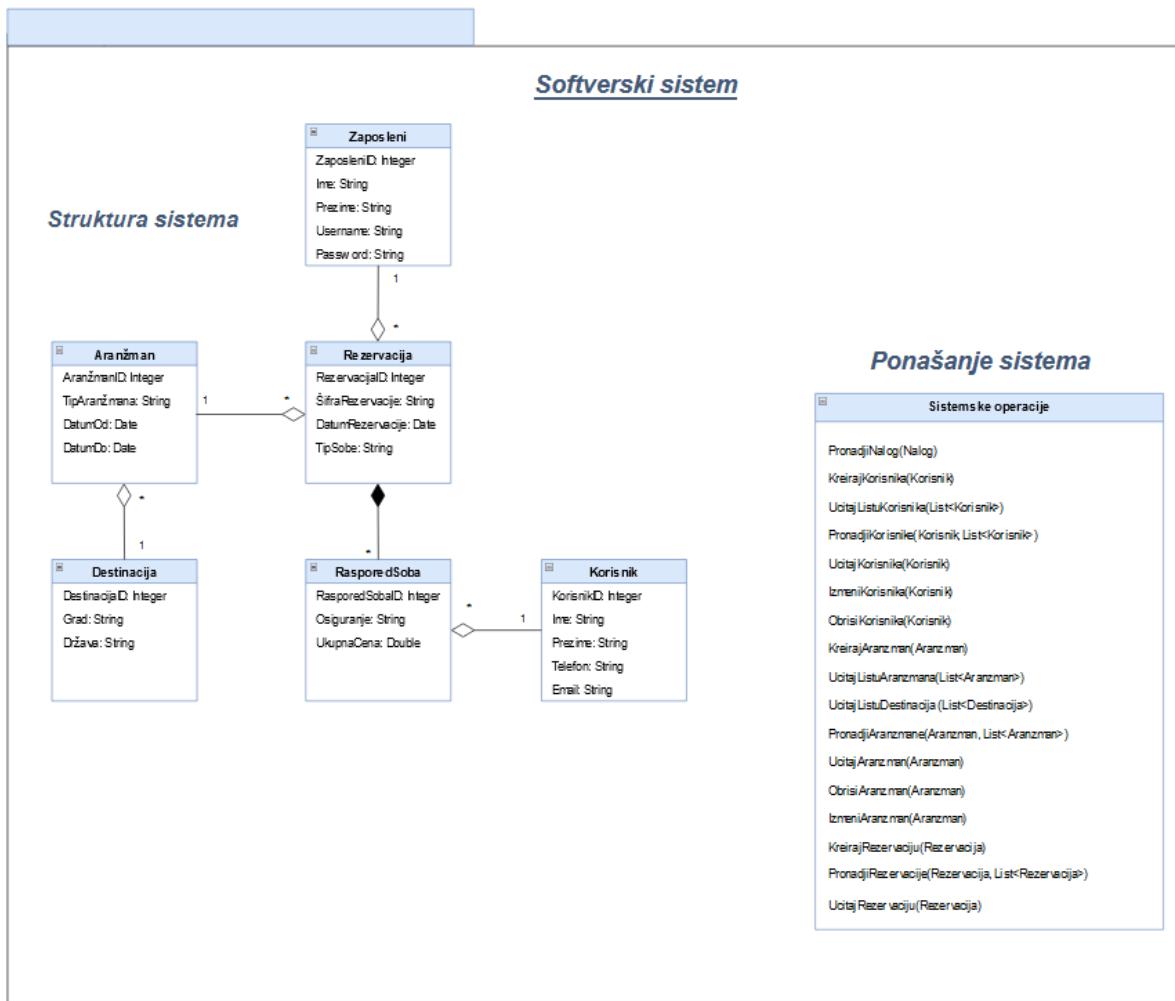
Tabela RasporedSoba		Prosto vrednosno ograničenje		Složeno vrednosno ograničenje		Strukturno organičenje
Atributi	Ime	Tip atributa	Vrednost atributa	Međuzavisnost atributa jedne tabele	Međuzavisnost atributa više tabele	INSERT RESTRICTED Korisnik, Rezervacija UPDATE RESTRICTED Korisnik, Rezervacija DELETE /
	RasporedSobalID	Integer	not null and >0			
	RezervacijalID	Integer	not null and >0			
	Osiguranje	String	not null			
	UkupnaCena	Double	not null			
	KorisnikID	Integer	not null and >0			

Табела 5 – RasporedSoba

Tabela Rezervacija		Prosto vrednosno ograničenje		Složeno vrednosno ograničenje		Strukturno organičenje
Atributi	Ime	Tip atributa	Vrednost atributa	Međuzavisnost atributa jedne tabele	Međuzavisnost atributa više tabele	INSERT RESTRICTED Zaposleni, Aranžman UPDATE RESTRICTED Zaposleni, Aranžman CASCADES TipSobe DELETE RESTRICTED TipSobe
	RezervacijalID	Integer	not null and >0			
	ŠifraRezervacije	String	not null			
	DatumRezervacije	Date	not null			
	TipSobe	String	not null			
	ZaposleniID	Integer	not null and >0			
	AranžmanID	Integer	not null and >0			

Табела 6 – Rezervacija

Како резултат анализе сценарија СК и прављења концептуалног модела, добија се логичка структура и понашање софтверског система:



Слика 39 – Софтверски систем

4.3. Пројектовање

Фаза пројектовања описује физичку структуру и понашање софтверског система (архитектура софтверског система).

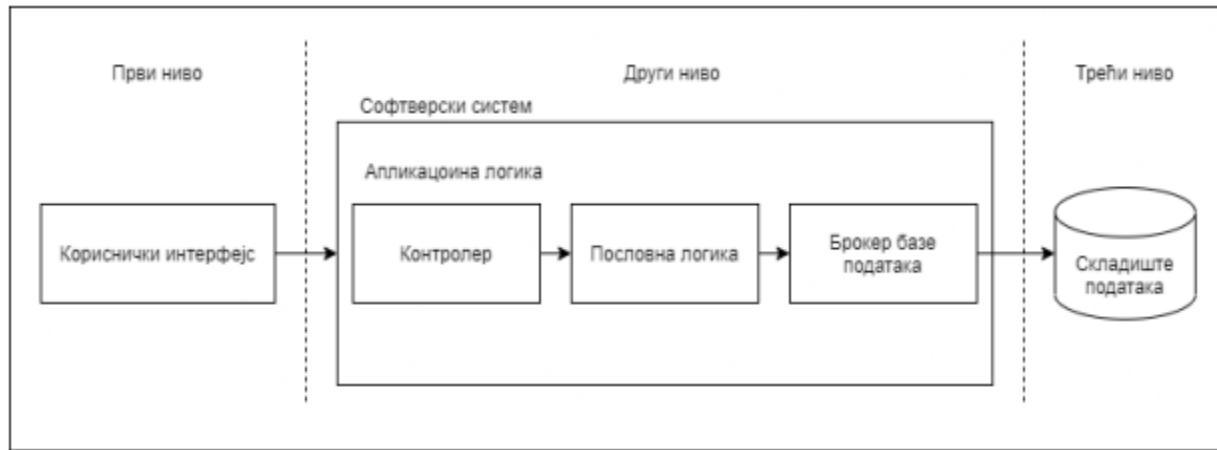
Пројектовање архитектуре софтверског система обухвата пројектовање корисничког интерфејса (пројектовање контролера корисничког интерфејса и екранских форми), апликационе логике (пројектовање контролера апликационе логике и пословне логике) и складишта података (брокер базе података) [1].

4.3.1. Архитектура софтверског система

Архитектура софтверског система је тронивојска и састоји се од следећих нивоа [1]:

- Кориснички интерфејс
- Апликациона логика
- Складиште података

Ниво корисничког интерфејса је на страни корисника, док су апликациона логика и складиште података на страни сервера.



Слика 40 - Тронивојска архитектура

4.3.2. Пројектовање корисничког интерфејса

Кориснички интерфејс представља начин на који софтверски систем комуницира са корисником.

Састоји се од [1]:

- Екранске форме
- Контролера корисничког интерфејса



Слика 41 - Структура корисничког интерфејса

Екранске форме прихватају податке и догађаје које корисник уноси, прослеђују те податке контролеру графичког интерфејса и приказују одговоре система кориснику. Контролер корисничког интерфејса прихвата податке од екранских форми, конвертује их у одговарајуће објекте доменских класа и шаље захтеве за извршење системских операција апликационом серверу. Након добијања резултата од софтверског система, контролер конвертује те податке у формат који је погодан за приказивање на графичким елементима екранске форме [1].

4.3.2.1. Пројектовање екранских форми

Кориснички интерфејс је дефинисан преко скупа екранских форми. Сценарија коришћења екранских форми су директно повезана са сценаријама случајева коришћења [1].

Постоје два аспекта пројектовања екранских форми [1]:

- Пројектовање сценарија СК који се изводе преко екранске форме
- Пројектовање метода екранске форме

На серверској страни програма налази се форма која регулише стање сервера (да ли је покренут сервер или не). Приликом покретања програма, она изгледа овако:



Слика 42 - Серверска форма

Кликом на дугме „**Покрени сервер**“, покреће се сервер и тада серверски сокет ослушају мрежу, преко које ће клијенти да се повежу са сервером. Тада ће форма сервера изгледати овако:



Слика 43 - Серверска форма: покренут сервер

Кликом на дугме „Заустави сервер“, сервер се поново зауставља и престаје да обрађује захтеве корисника. Тада ће форма сервера изгледати овако:



Слика 44 - Серверска форма: заустављен сервер

СК1: Случај коришћења – Пријава запосленог

Назив СК

Пријава **запосленог**

Актори СК

Запослени

Учесници СК

Запослени и **систем** (програм)

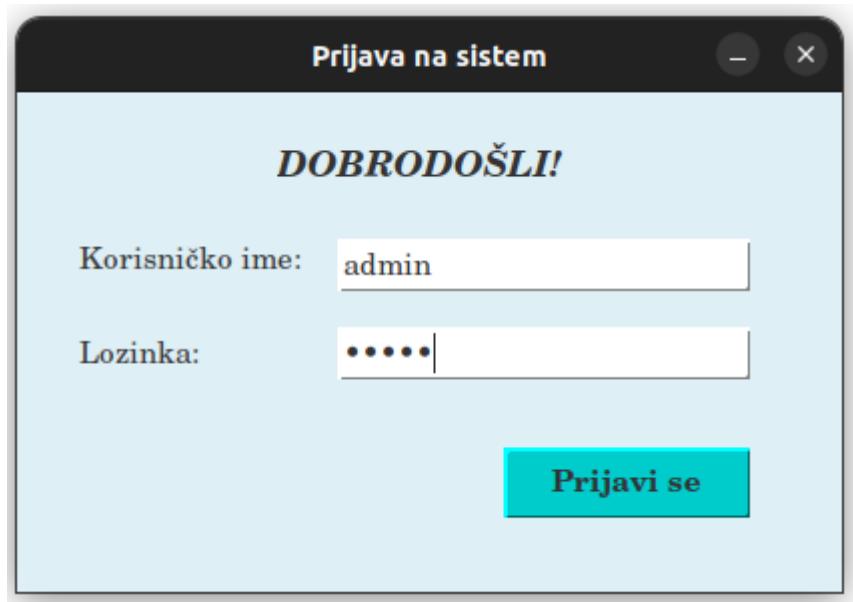
Предуслов: **Систем** је укључен и отворена је форма за пријављивање.

The screenshot shows a mobile application window titled "Prijava na sistem". At the top, there is a black header bar with the title and standard window controls (minimize and close). Below the header, the text "DOBRODOŠLI!" is displayed prominently in bold capital letters. The form consists of two input fields: "Korisničko ime:" followed by an input field, and "Lozinka:" followed by another input field. A large blue button at the bottom right contains the text "Prijavi se".

Слика 45 - Форма за пријављивање на систем

Основни сценарио СК

1. Запослени уноси податке за пријаву на систем. (АПУСО)
2. Запослени проверава да ли је коректно унео податке. (АНСО)

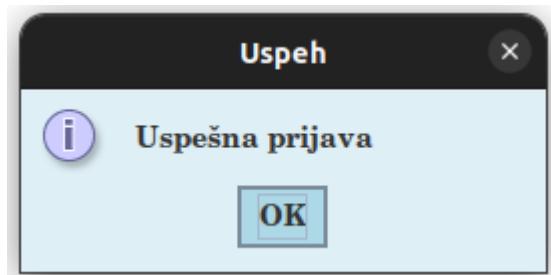


Слика 46 - Пријављивање запосленог на систем

3. Запослени позива систем да пронађе налог са задатим подацима. (АПСО)

Опис акције: Запослени притиском на дугме **Prijavi se** позива системску операцију *PronadijNalog(Nalog)*.

4. Систем тражи налог. (СО)
5. Систем приказује запосленом поруку: „Успешна пријава“. (ИА)



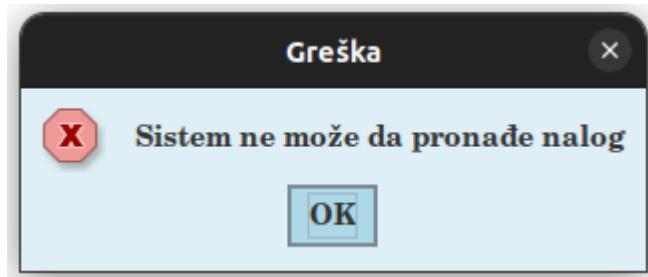
Слика 47 - Успешно пријављивање на систем



Слика 48 – Главна форма

Алтернативна сценарија

- 4.1 Уколико систем не може да нађе тражени налог, систем приказује запосленом поруку: „Систем не може да пронађе налог“. (ИА)



Слика 49 - Неуспешно пријављивање на систем

СК2: Случај коришћења – Креирање корисника

Назив СК

Креирање [корисника](#)

Актори СК

[Запослени](#)

Учесници СК

[Запослени](#) и [систем](#) (програм)

Предуслов: Систем је укључен и [запослени](#) је улогован под својом шифром. Систем приказује форму за рад са [корисницима](#).

The screenshot shows a user interface for creating a new user ('korisnik'). The title bar reads 'Glavna forma Korisnik'. The main section is titled 'FORMA KORISNIK' in bold capital letters. It contains four input fields: 'Ime:' (Name), 'Prezime:' (Last Name), 'Telefon:' (Phone), and 'Email:'. Below these fields is a teal-colored button labeled 'Dodaj korisnika' (Add user). The entire interface is set against a light blue background.

Слика 50 - Форма за креирање корисника

Основни сценарио СК

1. Запослени уноси податке о кориснику у систем. (АПУСО)

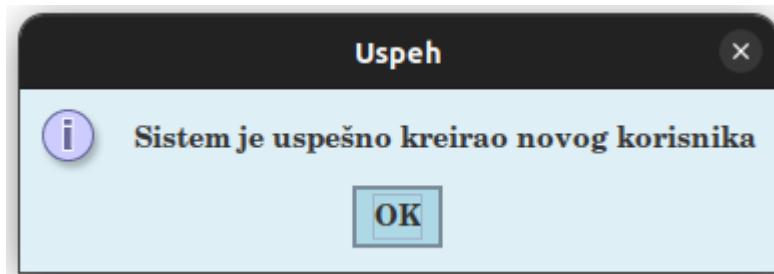
The screenshot shows a user creation form titled 'FORMA KORISNIK'. It contains four input fields: 'Ime' (Name) with value 'Milica', 'Prezime' (Last Name) with value 'Ivković', 'Telefon' (Phone) with value '+381636789543', and 'Email' with value 'milica@gmail.com'. Below the form is a teal button labeled 'Dodaj korisnika' (Add user).

Слика 51 - Попуњена форма за креирање корисника

2. Запослени контролише да ли је коректно унео податке о кориснику. (АНСО)
3. Запослени позива систем да креира новог корисника. (АПСО)

Опис акције: Запослени притиском на дугме **Dodaj korisnika** позива системску операцију **KreirajKorisnika(Korisnik)**.

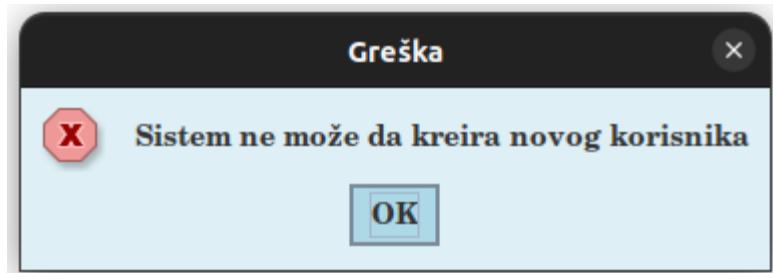
4. Систем креира новог корисника. (СО)
5. Систем приказује запосленом поруку: „Систем је успешно креирао новог корисника“. (ИА)



Слика 52 - Успешно креирање новог корисника

Алтернативна сценарија

5.1 Уколико систем не може да креира новог корисника, приказује запосленом поруку: „Систем не може да креира новог корисника”.



Слика 53 - Неуспешно креирање новог корисника

СК3: Случај коришћења – Претраживање корисника

Назив СК

Претраживање корисника

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са корисницима.

Pretraga korisnika

Pretraga korisnika po imenu:

Pretraga **Resetuj pretragu**

ime	prezime	telefon	email
Ana	Milošević	+381634679876	ana@gmail.com
Nikola	Ilić	+381667895436	nikola@gmail.com
Milica	Ivkovic	+381657890	milica@gmail.com
Nevena	Todorovic	+38164567890	nevena2@gmail.com
Ivan	Ognjanović	+381657890678	ivan@gmail.com
Vukašin	Branković	+38165907654	vukasin@gmail.com
Sofija	Jankovic	+38167890546	sofija@gmail.com
Milan	Mladenovic	+38145678543	milan@gmail.con
Sanja	Krstic	+38165789435	sofija@gmail.com
Milica	Zivkovic	+381654789	milica@gmail.com

Detalji

Слика 54 - Форма за приказ свих корисника

Основни сценарио СК

- Запослени уноси вредност по којој претражује кориснике. (АПУСО)

Pretraga korisnika

Pretraga korisnika po imenu:

Pretraga **Resetuj pretragu**

ime	prezime	telefon	email
Ana	Milošević	+381634679876	ana@gmail.com
Nikola	Ilić	+381667895436	nikola@gmail.com
Milica	Ivkovic	+381657890	milica@gmail.com
Nevena	Todorovic	+38164567890	nevena2@gmail.com
Ivan	Ognjanović	+381657890678	ivan@gmail.com
Vukašin	Branković	+38165907654	vukasin@gmail.com
Sofija	Jankovic	+38167890546	sofija@gmail.com
Milan	Mladenovic	+38145678543	milan@gmail.con
Sanja	Krstic	+38165789435	sofija@gmail.com
Milica	Zivkovic	+381654789	milica@gmail.com

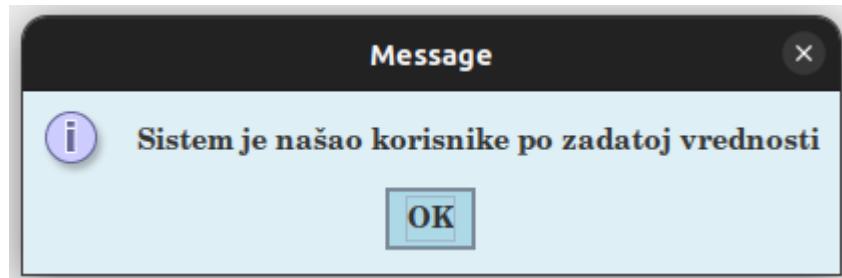
Detalji

Слика 55 - Унета вредност по којој се претражују корисници

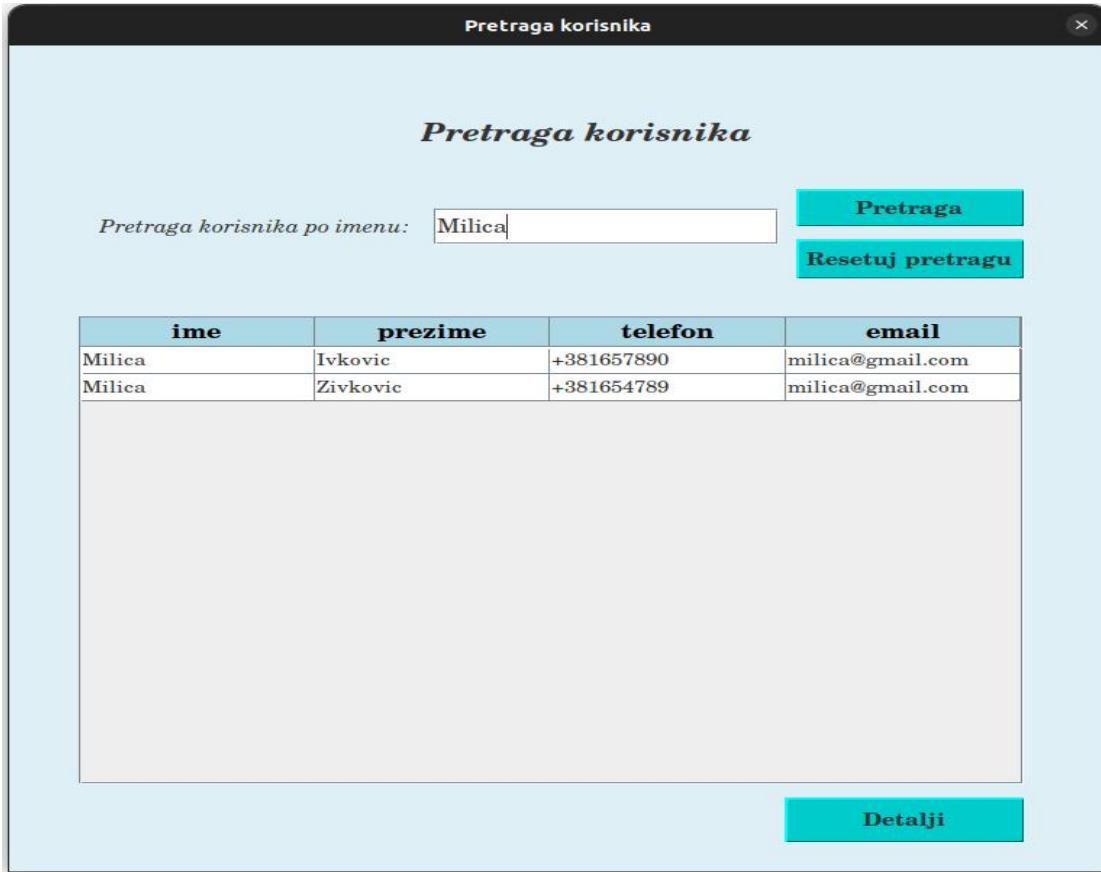
2. Запослени позива систем да нађе кориснике по задатој вредности. (АПСО)

Опис акције: Након што се форма за претрагу учита, запослени у поље за претрагу уноси име корисника и притиском на дугме **Pretraga** позива се системска операција **PronadjiKorisnike(Korisnik, List<Korisnik>)**.

3. Систем тражи кориснике по задатој вредности. (СО)
4. Систем приказује запосленом податке о корисницима и поруку: „Систем је нашао кориснике по задатој вредности“. (ИА)



Слика 56 - Успешно учитавање корисника по задатој вредности



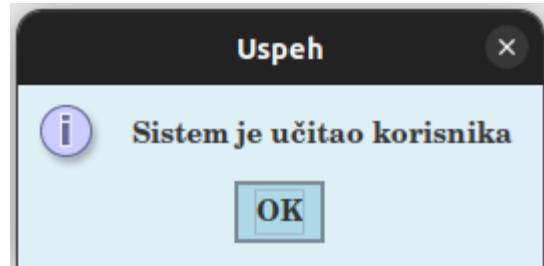
ime	prezime	telefon	email
Milica	Ivkovic	+381657890	milica@gmail.com
Milica	Zivkovic	+381654789	milica@gmail.com

Слика 57 - Понађени корисници по задатој вредности

5. Запослени бира корисника. (АПУСО)
6. Запослени позива систем да учита корисника. (АПСО)

Опис акције: Запослени прво селектује корисника из табеле којег жели да прикаже, кликом на ред у коме се тај корисник налази, а потом кликом на дугме *Detalji* позива системску операцију *UcitajKorisnika(Korisnik)*.

7. Систем учитава корисника. (СО)
8. Систем приказује запосленом податке о кориснику и поруку: „Систем је учитао корисника“. (ИА)



Слика 58 - Успешно учитан корисник

The image shows a mobile application's user interface titled "Glavna forma Korisnik" (Main User Form) at the top. Below the title, the form is titled "FORMA KORISNIK". The form contains five input fields with the following data:

- ID korisnika: 9
- Ime: Milica
- Prezime: Ivkovic
- Telefon: +381657890
- Email: milica@gmail.com

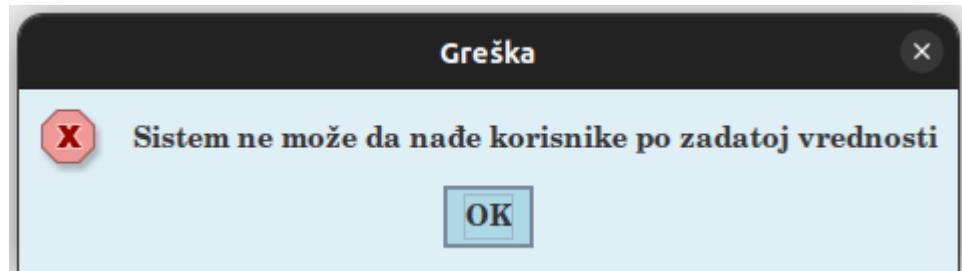
At the bottom of the form, there are two teal-colored buttons with white text:

- Izmeni podatke o korisniku (Change user data)
- Obriši korisnika (Delete user)

Слика 59 - Учитан корисник

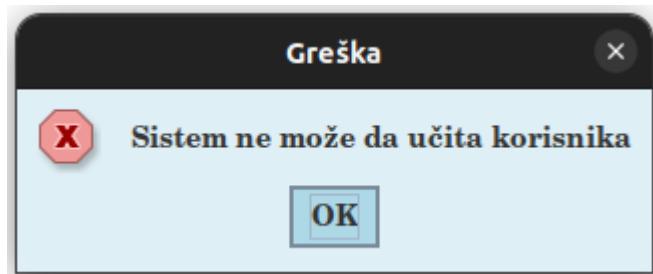
Алтернативна сценарија

4.1 Уколико систем не може да нађе кориснике, он приказује запосленом поруку: „Систем не може да нађе кориснике по задатој вредности”. Прекида се извршење сценарија. (ИА)



Слика 60 - Неуспешно учитавање корисника по задатој вредности

8.1 Уколико систем не може да учита корисника, он приказује запосленом поруку: „Систем не може да учита корисника”. (ИА)



Слика 61 - Неуспешно учитан корисник

СК4: Случај коришћења – Измена података о кориснику

Назив СК

Измена података о кориснику

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са корисницима.

Pretraga korisnika

Pretraga korisnika po imenu:

Pretraga **Resetuj pretragu**

ime	prezime	telefon	email
Ana	Milošević	+381634679876	ana@gmail.com
Nikola	Ilić	+381667895436	nikola@gmail.com
Milica	Ivkovic	+381657890	milica@gmail.com
Nevena	Todorovic	+38164567890	nevena2@gmail.com
Ivan	Ognjanović	+381657890678	ivan@gmail.com
Vukašin	Branković	+38165907654	vukasin@gmail.com
Sofija	Jankovic	+38167890546	sofija@gmail.com
Milan	Mladenovic	+38145678543	milan@gmail.con
Sanja	Krstic	+38165789435	sofija@gmail.com
Milica	Zivkovic	+381654789	milica@gmail.com

Detalji

Слика 62 - Форма за приказ свих корисника

Основни сценарио СК

- Запослени уноси вредност по којој претражује кориснике. (АПУСО)

Pretraga korisnika

Pretraga korisnika po imenu: Milica

Pretraga

Resetuj pretragu

ime	prezime	telefon	email
Ana	Milošević	+381634679876	ana@gmail.com
Nikola	Ilić	+381667895436	nikola@gmail.com
Milica	Ivkovic	+381657890	milica@gmail.com
Nevena	Todorovic	+38164567890	nevena2@gmail.com
Ivan	Ognjanović	+381657890678	ivan@gmail.com
Vukašin	Branković	+38165907654	vukasin@gmail.com
Sofija	Jankovic	+38167890546	sofija@gmail.com
Milan	Mladenovic	+38145678543	milan@gmail.con
Sanja	Krstic	+38165789435	sofija@gmail.com
Milica	Zivkovic	+381654789	milica@gmail.com

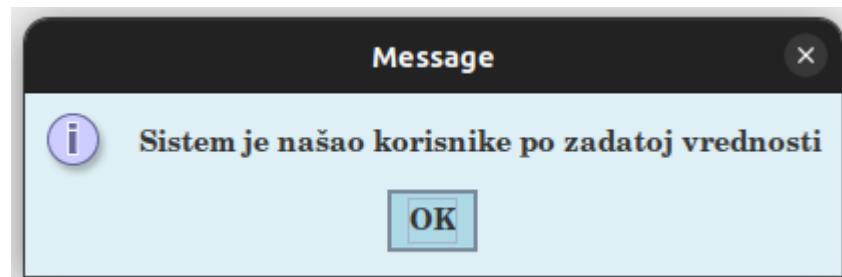
Detalji

Слика 63 - Унета вредност по којој се претражују корисници

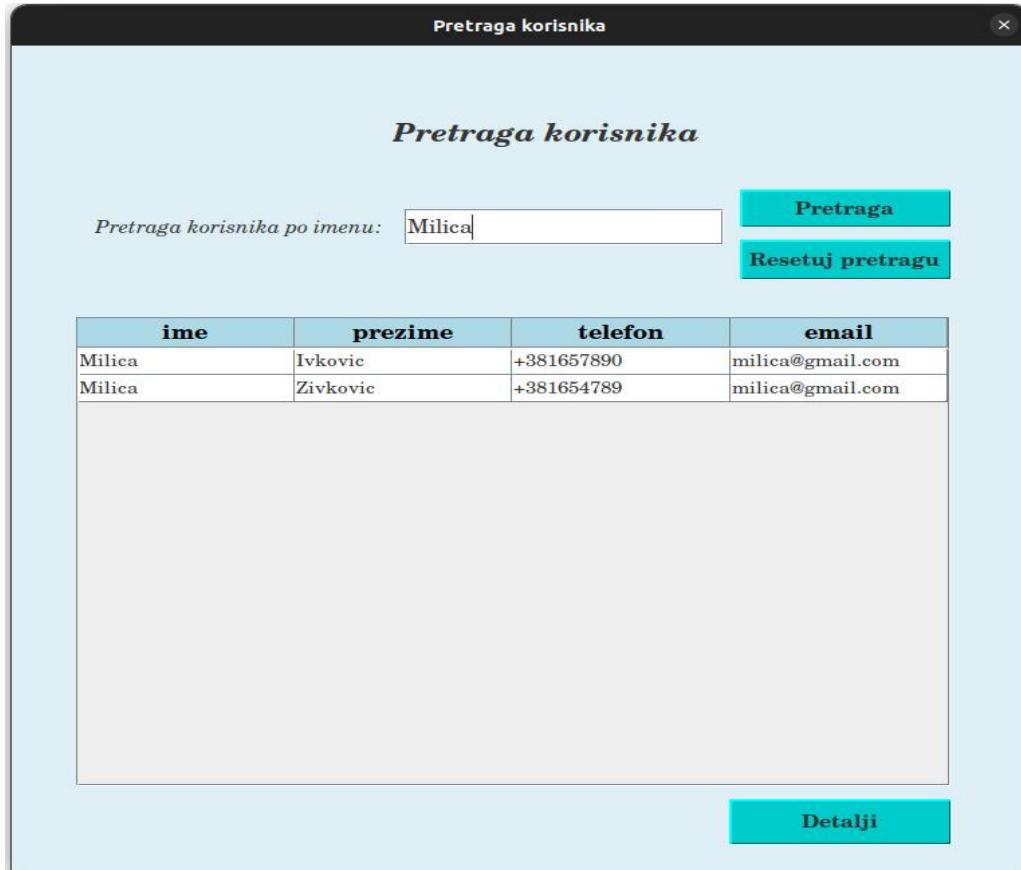
2. Запослени позива систем да нађе кориснике по задатој вредности. (АПСО)

Опис акције: Након што се форма за претрагу учита, запослени у поље за претрагу уноси име корисника и притиском на дугме **Pretraga** позива се системска операција **PronadjiKorisnike(Korisnik, List<Korisnik>)**.

3. Систем тражи кориснике по задатој вредности. (СО)
4. Систем приказује запосленом податке о корисницима и поруку: „Систем је нашао кориснике по задатој вредности“. (ИА)



Слика 64 - Успешно учитавање корисника по задатој вредности



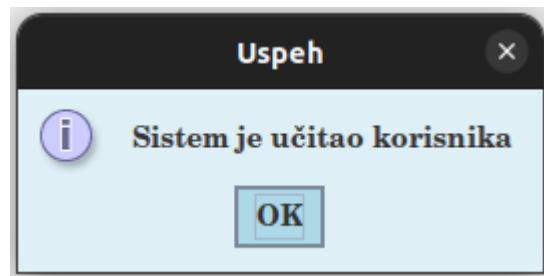
ime	prezime	telefon	email
Milica	Ivkovic	+381657890	milica@gmail.com
Milica	Zivkovic	+381654789	milica@gmail.com

Слика 65 - Понађени корисници по задатој вредности

5. Запослени бира жељеног корисника. (АПУСО)
6. Запослени позива систем да учита корисника. (АПСО)

Опис акције: Запослени прво селектује корисника из табеле којег жељи да измени, кликом на ред у коме се тај корисник налази, а потом кликом на дугме *Detalji* позива системску операцију *UcitajKorisnika(Korisnik)*.

7. Систем учитава корисника. (СО)
8. Систем приказује запосленом податке о кориснику и поруку: „Систем је учитао корисника“. (ИА)



Слика 66 - Успешно учитан корисник

Glavna forma Korisnik

FORMA KORISNIK

ID korisnika:	<input type="text" value="9"/>
Ime:	<input type="text" value="Milica"/>
Prezime:	<input type="text" value="Ivkovic"/>
Telefon:	<input type="text" value="+381657890"/>
Email:	<input type="text" value="milica@gmail.com"/>

Izmeni podatke o korisniku

Obriši korisnika

Слика 67 - Учитани корисник

9. Запослени уноси (мења) податке о кориснику. (АПУСО)
10. Запослени контролише да ли је коректно унео податке о кориснику. (АХСО)

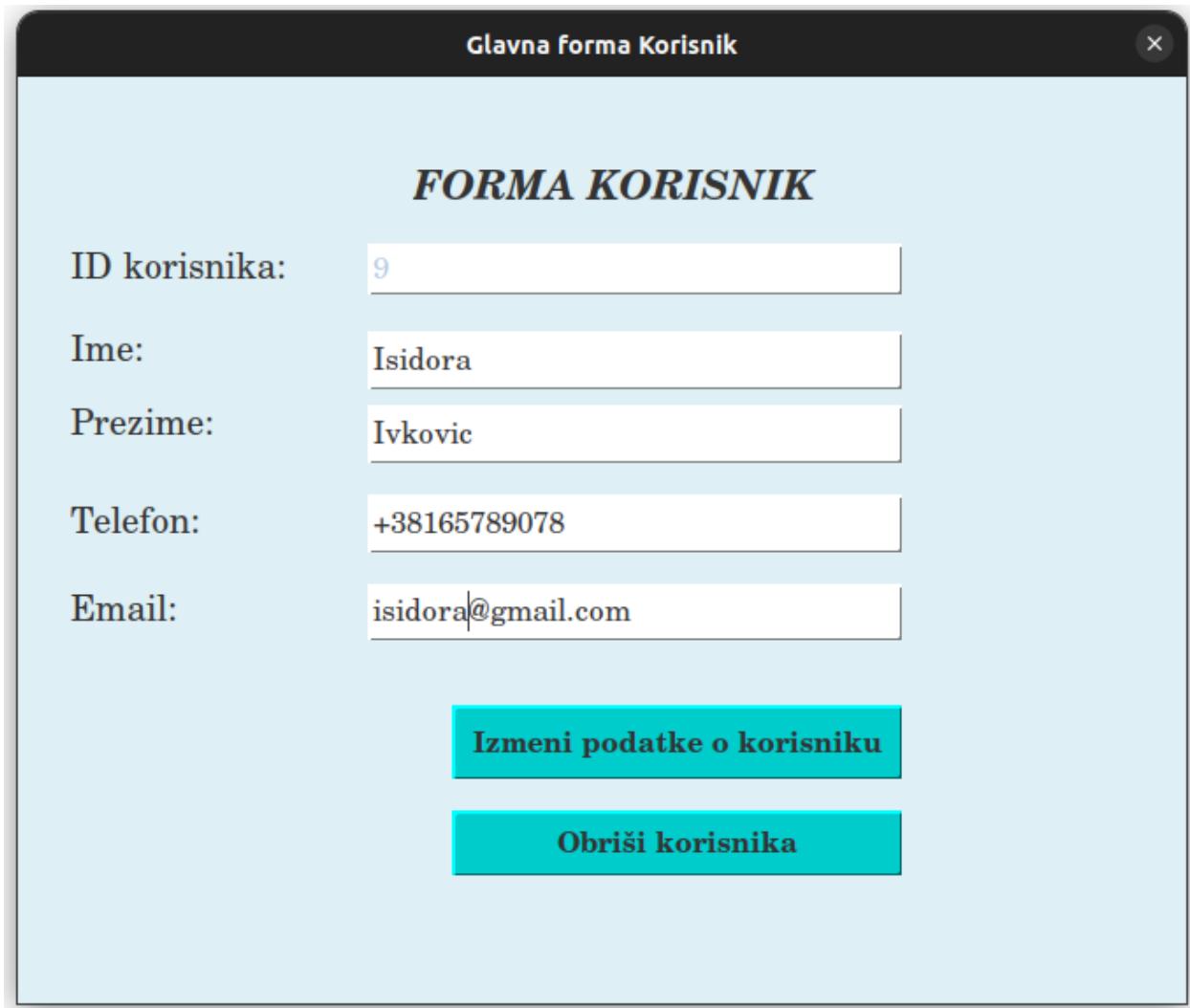
Glavna forma Korisnik

FORMA KORISNIK

ID korisnika:	9
Ime:	Isidora
Prezime:	Ivkovic
Telefon:	+38165789078
Email:	isidora@gmail.com

Izmeni podatke o korisniku

Obriši korisnika

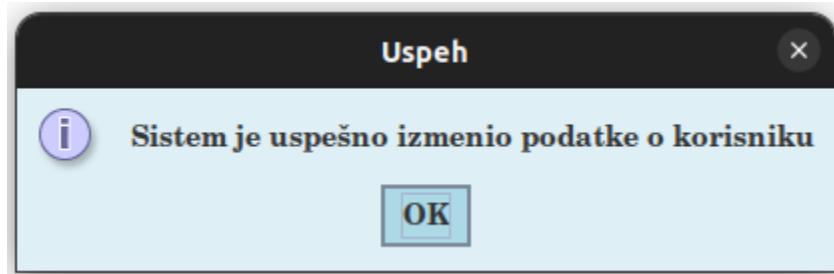


Слика 68 - Форма за измену корисника

11. Запослени позива систем да измени податке о кориснику. (АПСО)

Опис акције: Запослени кликом на дугме **Izmeni podatke o korisniku** позива системску операцију **IzmeniKorisnika(Korisnik)**.

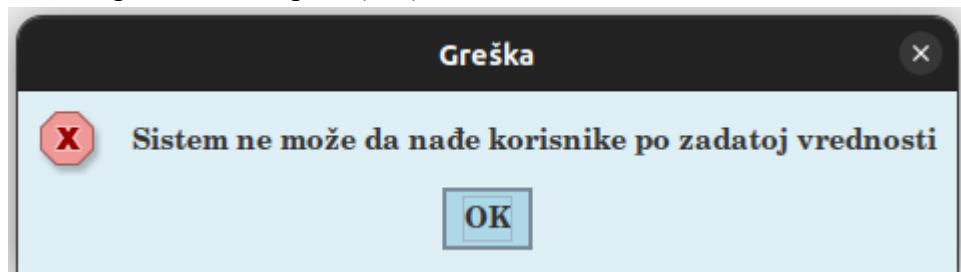
12. Систем мења податке о кориснику. (СО)
13. Систем приказује запосленом измененог корисника и поруку: „Систем је успешно изменио податке о кориснику.” (ИА)



Слика 69 - Успешно изменењи подаци о кориснику

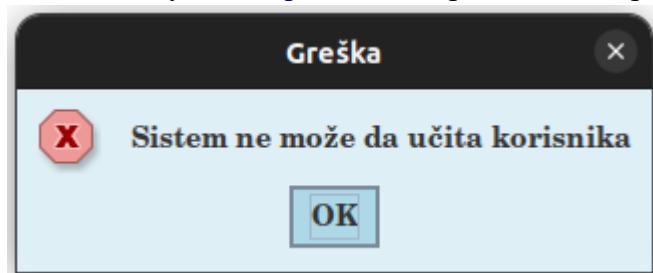
Алтернативна сценарија

4.1 Уколико систем не може да нађе кориснике по задатој вредности, он приказује запосленом поруку: „Систем не може да нађе кориснике по задатој вредности”. Прекида се извршење сценарија. (ИА)



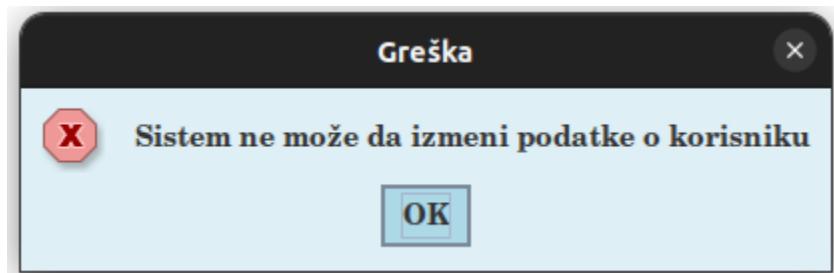
Слика 70 - Неуспешно учитавање корисника по задатој вредности

11.1 Уколико систем не може да учита податке о кориснику, он приказује запосленом поруку „Систем не може да учита корисника”. Прекида се извршење сценарија. (ИА)



Слика 71 - Неуспешно учитан корисник

13.1 Уколико систем не може да измени податке о кориснику, он приказује запосленом поруку: „Систем не може да измени податке о кориснику”. (ИА)



Слика 72 - Неуспешно изменењи подаци о кориснику

СК5: Случај коришћења – Брисање корисника

Назив СК

Брисање корисника

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са корисницима.

Pretraga korisnika

Pretraga korisnika po imenu:

Pretraga **Resetuj pretragu**

ime	prezime	telefon	email
Ana	Milošević	+381634679876	ana@gmail.com
Nikola	Ilić	+381667895436	nikola@gmail.com
Milica	Ivkovic	+381657890	milica@gmail.com
Nevena	Todorovic	+38164567890	nevena2@gmail.com
Ivan	Ognjanović	+381657890678	ivan@gmail.com
Vukašin	Branković	+38165907654	vukasin@gmail.com
Sofija	Jankovic	+38167890546	sofija@gmail.com
Milan	Mladenovic	+38145678543	milan@gmail.con
Sanja	Krstic	+38165789435	sofija@gmail.com
Milica	Zivkovic	+381654789	milica@gmail.com

Detalji

Слика 73 - Форма за приказ свих корисника

Основни сценарио СК

- Запослени уноси вредност по којој претражује кориснике. (АПУСО)

Pretraga korisnika

Pretraga korisnika po imenu: Milica

Pretraga

Resetuj pretragu

ime	prezime	telefon	email
Ana	Milošević	+381634679876	ana@gmail.com
Nikola	Ilić	+381667895436	nikola@gmail.com
Milica	Ivkovic	+381657890	milica@gmail.com
Nevena	Todorovic	+38164567890	nevena2@gmail.com
Ivan	Ognjanović	+381657890678	ivan@gmail.com
Vukašin	Branković	+38165907654	vukasin@gmail.com
Sofija	Jankovic	+38167890546	sofija@gmail.com
Milan	Mladenovic	+38145678543	milan@gmail.con
Sanja	Krstic	+38165789435	sofija@gmail.com
Milica	Zivkovic	+381654789	milica@gmail.com

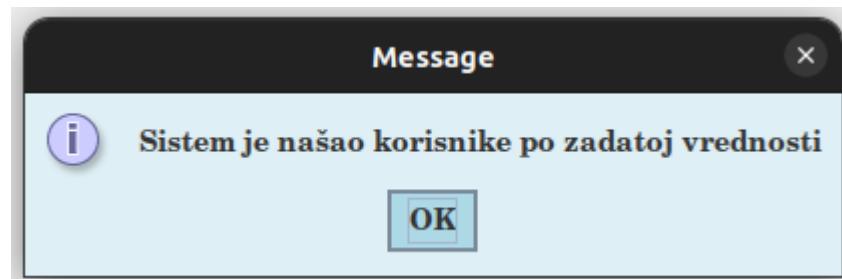
Detalji

Слика 74 - Унета вредност по којој се претражују корисници

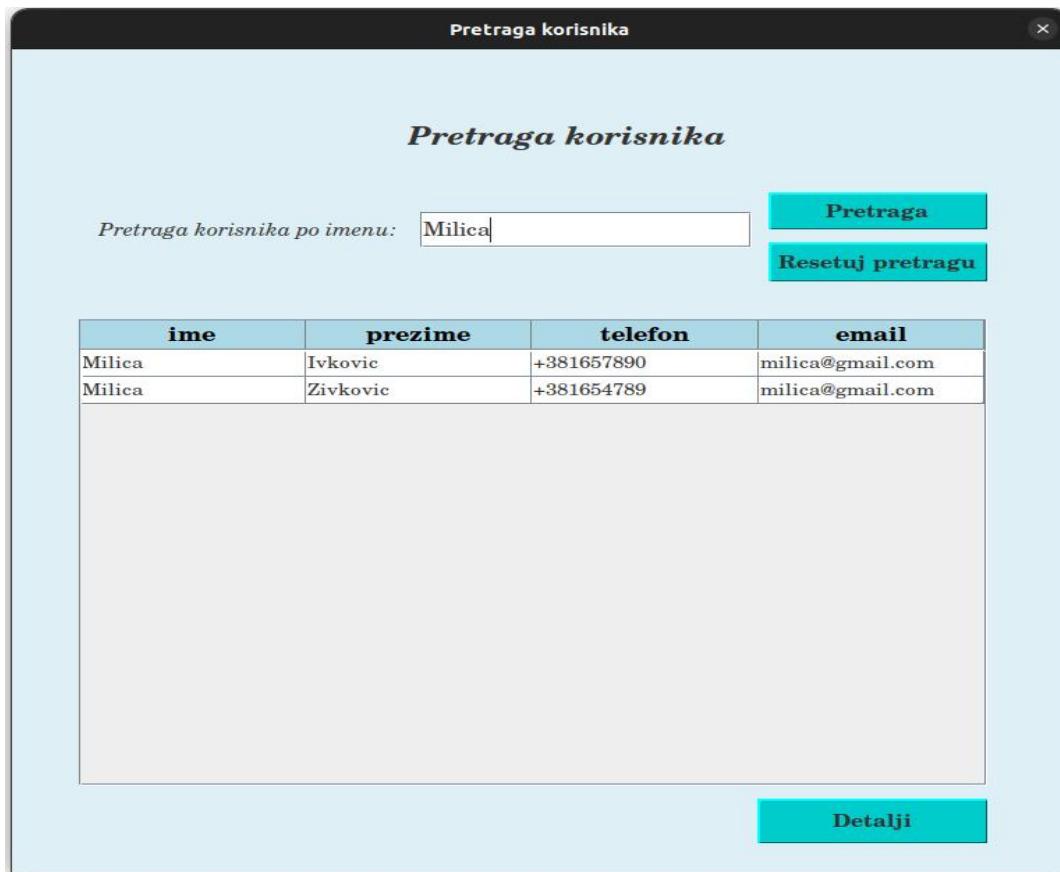
2. Запослени позива систем да нађе кориснике по задатој вредности. (АПСО)

Опис акције: Након што се форма за претрагу учита, запослени у поље за претрагу уноси име корисника и притиском на дугме **Pretraga** позива се системска операција **PronadjiKorisnike(Korisnik, List<Korisnik>)**.

3. Систем тражи кориснике по задатој вредности. (СО)
4. Систем приказује запосленом податке о корисницима и поруку: „Систем је нашао кориснике по задатој вредности“. (ИА)



Слика 75 - Успешно учитавање корисника по задатој вредности



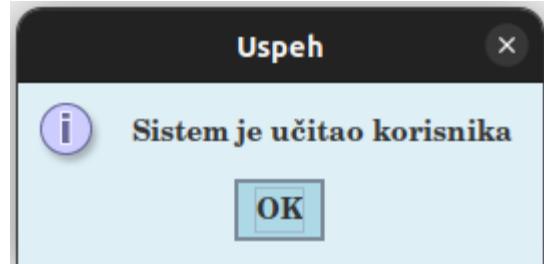
ime	prezime	telefon	email
Milica	Ivkovic	+381657890	milica@gmail.com
Milica	Zivkovic	+381654789	milica@gmail.com

Слика 76 - Понађени корисници по задатој вредности

5. Запослени бира жељеног корисника. (АПУСО)
6. Запослени позива систем да учита корисника. (АПСО)

Опис акције: Запослени прво селектује корисника из табеле којег жељи да измени, кликом на ред у коме се тај корисник налази, а потом кликом на дугме **Detalji** позива системску операцију **UcitajKorisnika(Korisnik)**.

7. Систем учитава корисника. (СО)
8. Систем приказује запосленом податке о кориснику и поруку: „Систем је учитао корисника“. (ИА)



Слика 77 - Успешно учитан корисник

ID korisnika:	9
Ime:	Milica
Prezime:	Ivkovic
Telefon:	+381657890
Email:	milica@gmail.com
Izmeni podatke o korisniku	
Obriši korisnika	

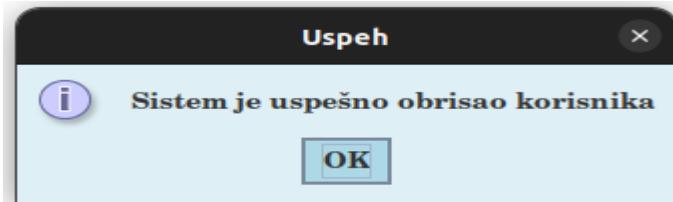
Слика 78 - Учитани корисник

9. Запослени позива систем да обрише корисника. (АПСО)

Опис акције: Запослени кликом на дугме **Obriši korisnika** позива системску операцију **ObrišiKorisnika(Korisnik)**.

10. Систем брише корисника. (СО)

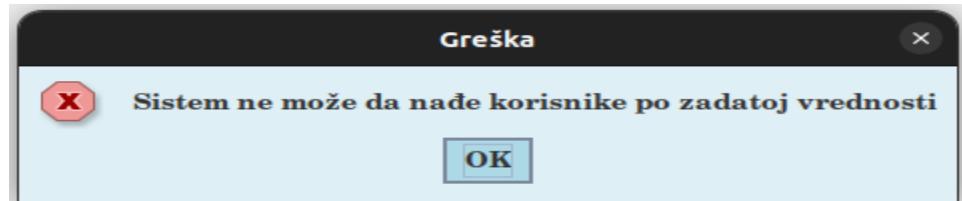
11. Систем приказује запосленом поруку: „Систем је успешно обрисао корисника.” (ИА)



Слика 79 - Успешно обрисан корисник

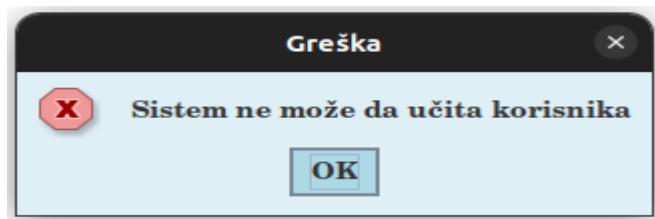
Алтернативна сценарија

4.1 Уколико систем не може да нађе кориснике, он приказује запосленом поруку: „Систем не може да нађе кориснике по задатој вредности”. Прекида се извршење сценарија. (ИА)



Слика 80 - Неуспешно учитавање корисника по задатој вредности

- 8.1 Уколико систем не може да учита податке о кориснику, он приказује запосленом поруку „Систем не може да учита корисника”. Прекида се извршење сценарија. (ИА)



Слика 81 - Неуспешно учитан корисник

- 11.1 Уколико систем не може да обрише корисника, он приказује запосленом поруку: „Систем не може да обрише корисника”. (ИА)



Слика 82 - Неуспешно обрисан корисник

СК6: Случај коришћења – Креирање аранжмана

Назив СК

Креирање аранжмана

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са аранжманима. Учитана је листа дестинација.

The screenshot shows a user interface titled 'Glavna forma Aranžman' (Main Arrangement Form). The main title 'FORMA ARANŽMAN' is centered at the top in a large, bold, serif font. Below it, there are four input fields and one button. The first field is labeled 'Tip aranžmana:' (Type of arrangement:) and contains the value 'LETOVANJE' in a dropdown menu. The second field is labeled 'Datum od (yyyy-MM-dd):' (From date (yyyy-MM-dd):) and is empty. The third field is labeled 'Datum do (yyyy-MM-dd):' (To date (yyyy-MM-dd):) and is also empty. The fourth field is labeled 'Destinacija:' (Destination:) and contains the value 'Budimpešta, Mađarska' in a dropdown menu. At the bottom right is a teal-colored button with the text 'Dodaj aranžman' (Add arrangement).

Слика 83 - Форма за креирање аранжмана

Основни сценарио СК

- Запослени уноси податке о аранжману. (АПУСО)

Glavna forma Aranžman

FORMA ARANŽMAN

Tip aranžmana: EVROPSKI_GRADOVI

Datum od (yyyy-MM-dd): 2024-09-10

Datum do (yyyy-MM-dd): 2024-15-10

Destinacija: Prag, Češka

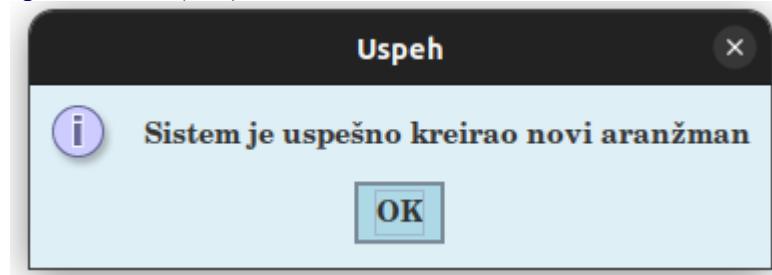
Dodaj aranžman

Слика 84 - Попуњена форма за креирање аранжмана

- Запослени контролише да ли је коректно унео податке о аранжману. (АХСО)
- Запослени позива систем да креира нови аранжман. (АПСО)

Опис акције: Запослени притиском на дугме **Dodaj aranžman** позива системску операцију **KreirajAranzman(Aranzman)**.

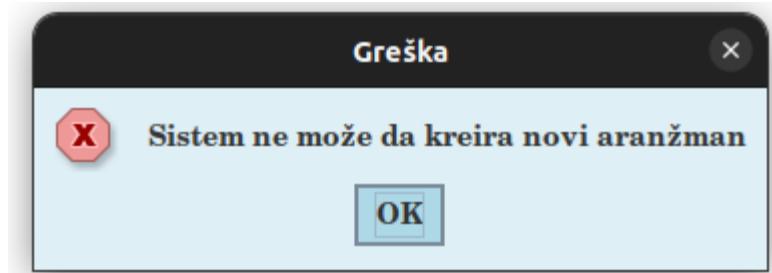
- Систем памти податке о аранжману. (СО)
- Систем приказује запосленом креирани аранжман и поруку: „Систем је успешно креирао нови аранжман“. (ИА)



Слика 85 - Успешно креирање аранжмана

Алтернативна сценарија

5.1 Уколико **систем** не може да креира **аранжман**, он приказује **запосленом** поруку:
„**Систем** не може да креира нови **аранжман**”.



Слика 86 - Неуспешно креирање аранжмана

СК7: Случај коришћења – Претраживање аранжмана

Назив СК

Претраживање аранжмана

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са аранжманима.

Pretraga Aranžmana

Pretraga aranžmana po tipu:

Pretraga **Resetuj pretragu**

tipAranžmana	DatumOd	DatumDo	Destinacija
ESKURZIJA	2024-03-20	2024-05-29	Rim, Italija
LETOVANJE	2024-09-09	2024-09-09	Lefkada, Grčka
PRVI_MAJ	2024-07-07	2024-07-07	Bansko, Bugarska
EVROPSKI_GRADOVI	2024-03-12	2024-03-21	Istanbul, Turska
ESKURZIJA	2024-05-26	2024-05-28	Rim, Italija
DALEKE_DESTINAC...	2024-08-08	2024-08-21	Tokio, Japan
IZLET	2024-05-05	2024-05-05	Temišvar, Румунија
EVROPSKI_GRADOVI	2024-09-10	2024-09-17	Pariz, Француска
NOVA_GODINA	2024-07-18	2024-07-24	Budimpešta, Мађарска
EVROPSKI_GRADOVI	2024-08-20	2024-08-27	Barcelona, Шпанија
EVROPSKI_GRADOVI	2024-09-10	2024-09-15	Prag, Чешка

Detalji

Слика 87 - Форма за приказ свих аранжмана

Основни сценарио СК

- Запослени уноси вредност по којој претражује аранжмане. (АПУСО)

Pretraga Aranžmana

Pretraga aranžmana po tipu: evropski

Pretraga **Resetuj pretragu**

tipAranžmana	DatumOd	DatumDo	Destinacija
ESKURZIJA	2024-03-20	2024-05-29	Rim, Italija
LETOVANJE	2024-09-09	2024-09-09	Lefkada, Grčka
PRVI_MAJ	2024-07-07	2024-07-07	Bansko, Bugarska
EVROPSKI_GRADOVI	2024-03-12	2024-03-21	Istanbul, Turska
ESKURZIJA	2024-05-26	2024-05-28	Rim, Italija
DALEKE_DESTINAC...	2024-08-08	2024-08-21	Tokio, Japan
IZLET	2024-05-05	2024-05-05	Temišvar, Rumunija
EVROPSKI_GRADOVI	2024-09-10	2024-09-17	Pariz, Francuska
NOVA_GODINA	2024-07-18	2024-07-24	Budimpešta, Mađarska
EVROPSKI_GRADOVI	2024-08-20	2024-08-27	Barcelona, Španija
EVROPSKI_GRADOVI	2024-09-10	2024-09-15	Prag, Češka

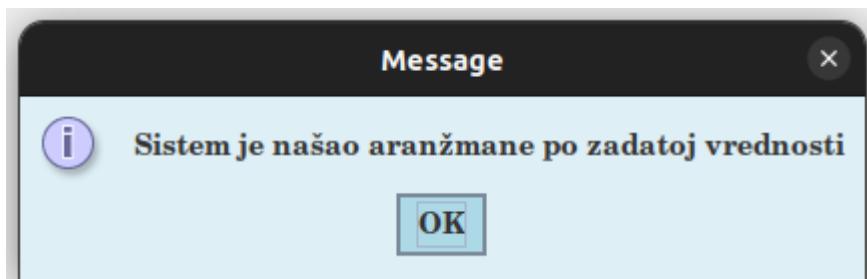
Detalji

Слика 88 - Унета вредност по којој се претражује аранжман

- Запослени позива систем да нађе аранжмане по задатој вредности. (АПСО)

Опис акције: Након што се форма за претрагу учита, запослени у поље за претрагу уноси тип аранжмана и притиском на дугме **Pretraga** позива се системска операција **PronadjiAranzmane(Aranzman, List<Aranzman>)**.

- Систем тражи аранжмане по задатој вредности. (СО)
- Систем приказује запосленом податке о аранжманима и поруку: „Систем је нашао аранжмане по задатој вредности“. (ИА)



Слика 89 - Успешно учитавање аранжмана по задатој вредности

Pretraga Aranžmana

Pretraga aranžmana po tipu: evropski

Pretraga **Resetuj pretragu**

tipAranžmana	DatumOd	DatumDo	Destinacija
EVROPSKI_GRADOVI	2024-03-12	2024-03-21	Istanbul, Turska
EVROPSKI_GRADOVI	2024-09-10	2024-09-17	Pariz, Francuska
EVROPSKI_GRADOVI	2024-08-20	2024-08-27	Barselona, Španija
EVROPSKI_GRADOVI	2024-09-10	2024-09-15	Prag, Češka

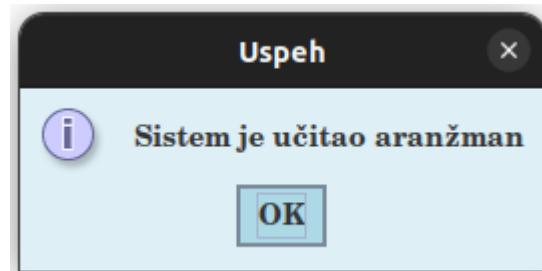
Detalji

Слика 90 - Пронађени аранжмани по задатој вредности

5. Запослени бира аранжман. (АПУСО)
6. Запослени позива систем да учита аранжман. (АПСО)

Опис акције: Запослени прво селектује аранжман из табеле који жељи да прикаже, кликом на ред у коме се тај аранжман налази, а потом кликом на дугме **Detalji** позива системску операцију **UcitajAranzman(Aranzman)**.

7. Систем учитава аранжман. (СО)
8. Систем приказује запосленом податке о аранжману и поруку: „Систем је учитао аранжман”. (ИА)



Слика 91 - Успешно учитан аранжман

Glavna forma Aranžman

FORMA ARANŽMAN

ID aranžmana:	9
Tip aranžmana:	EVROPSKI_GRADOVI
Datum od (yyyy-MM-dd):	2024-03-12
Datum do (yyyy-MM-dd):	2024-03-21
Destinacija:	Istanbul, Turska

Izmeni podatke o aranžmanu

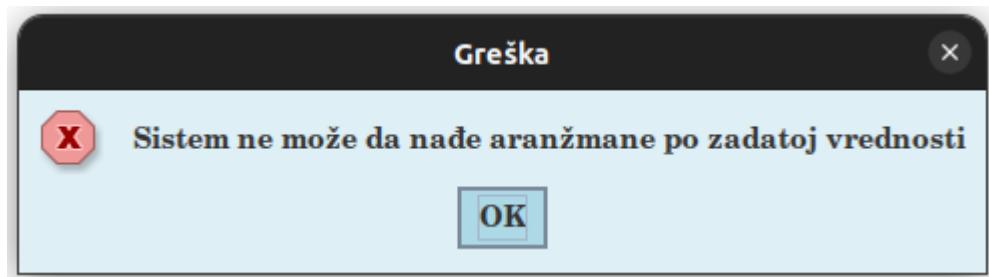
Obriši aranžman

A screenshot of a Windows application window titled 'Glavna forma Aranžman'. The main title is 'FORMA ARANŽMAN'. It contains five input fields: 'ID aranžmana' (9), 'Tip aranžmana' (EVROPSKI_GRADOVI), 'Datum od (yyyy-MM-dd)' (2024-03-12), 'Datum do (yyyy-MM-dd)' (2024-03-21), and 'Destinacija' (Istanbul, Turska). Below these fields are two teal-colored buttons: 'Izmeni podatke o aranžmanu' and 'Obriši aranžman'.

Слика 92 - Учитан аранжман

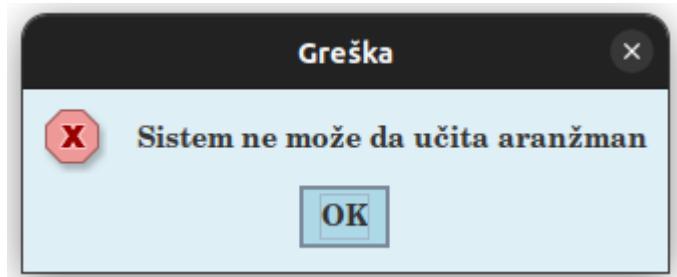
Алтернативна сценарија

4.1 Уколико **систем** не може да нађе **аранжмане**, он приказује **запосленом** поруку: „**Систем** не може да нађе **аранжмане** по задатој вредности”. Прекида се извршење сценарија. (ИА)



Слика 93 - Неуспешно учитавање аранжмана по задатој вредности

8.1 Уколико **систем** не може да учита **аранжман**, он приказује **запосленом** поруку: „**Систем** не може да учита **аранжман**”. (ИА)



Слика 94 - Неуспешно учитан аранжман

СК8: Случај коришћења – Брисање аранжмана

Назив СК

Брисање аранжмана

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са аранжманима.

Pretraga Aranžmana

Pretraga aranžmana po tipu:

Pretraga **Resetuj pretragu**

tipAranžmana	DatumOd	DatumDo	Destinacija
ESKURZIJA	2024-03-20	2024-05-29	Rim, Italija
LETOVANJE	2024-09-09	2024-09-09	Lefkada, Grčka
PRVI_MAJ	2024-07-07	2024-07-07	Bansko, Bugarska
EVROPSKI_GRADOVI	2024-03-12	2024-03-21	Istanbul, Turska
ESKURZIJA	2024-05-26	2024-05-28	Rim, Italija
DALEKE_DESTINAC...	2024-08-08	2024-08-21	Tokio, Japan
IZLET	2024-05-05	2024-05-05	Temišvar, Румунија
EVROPSKI_GRADOVI	2024-09-10	2024-09-17	Pariz, Француска
NOVA_GODINA	2024-07-18	2024-07-24	Budimpešta, Мађарска
EVROPSKI_GRADOVI	2024-08-20	2024-08-27	Barcelona, Шпанија
EVROPSKI_GRADOVI	2024-09-10	2024-09-15	Prag, Чешка

Detalji

Слика 95 - Форма за приказ свих аранжмана

Основни сценарио СК

- Запослени уноси вредност по којој претражује аранжмане. (АПУСО)

Pretraga Aranžmana

Pretraga aranžmana po tipu: evropski

Pretraga **Resetuj pretragu**

tipAranžmana	DatumOd	DatumDo	Destinacija
ESKURZIJA	2024-03-20	2024-05-29	Rim, Italija
LETOVANJE	2024-09-09	2024-09-09	Lefkada, Grčka
PRVI_MAJ	2024-07-07	2024-07-07	Bansko, Bugarska
EVROPSKI_GRADOVI	2024-03-12	2024-03-21	Istanbul, Turska
ESKURZIJA	2024-05-26	2024-05-28	Rim, Italija
DALEKE_DESTINAC...	2024-08-08	2024-08-21	Tokio, Japan
IZLET	2024-05-05	2024-05-05	Temišvar, Rumunija
EVROPSKI_GRADOVI	2024-09-10	2024-09-17	Pariz, Francuska
NOVA_GODINA	2024-07-18	2024-07-24	Budimpešta, Mađarska
EVROPSKI_GRADOVI	2024-08-20	2024-08-27	Barcelona, Španija
EVROPSKI_GRADOVI	2024-09-10	2024-09-15	Prag, Češka

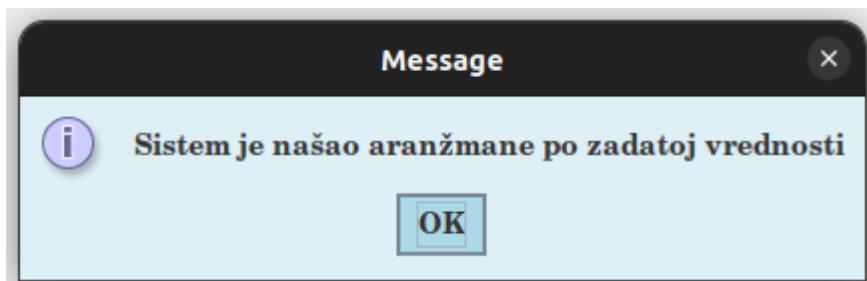
Detalji

Слика 96 - Унета вредност по којој се претражује аранжман

- Запослени позива систем да нађе аранжмане по задатој вредности. (АПСО)

Опис акције: Након што се форма за претрагу учита, запослени у поље за претрагу уноси тип аранжмана и притиском на дугме **Pretraga** позива се системска операција **PronadjiAranzmane(Aranzman, List<Aranzman>)**.

- Систем тражи аранжмане по задатој вредности. (СО)
- Систем приказује запосленом податке о аранжманима и поруку: „Систем је нашао аранжмане по задатој вредности“. (ИА)



Слика 97 - Успешно учитавање аранжмана по задатој вредности

Pretraga Aranžmana

Pretraga aranžmana po tipu:

evropski

Pretraga

Resetuj pretragu

tipAranžmana	DatumOd	DatumDo	Destinacija
EVROPSKI_GRADOVI	2024-03-12	2024-03-21	Istanbul, Turska
EVROPSKI_GRADOVI	2024-09-10	2024-09-17	Pariz, Francuska
EVROPSKI_GRADOVI	2024-08-20	2024-08-27	Barselona, Španija
EVROPSKI_GRADOVI	2024-09-10	2024-09-15	Prag, Češka

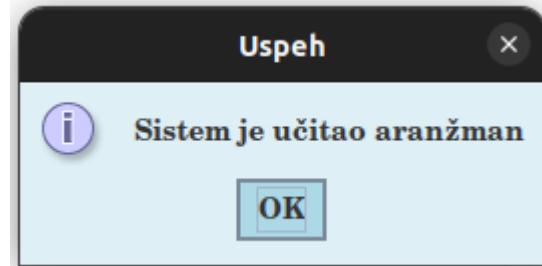
Detalji

Слика 98 - Пронађени аранжмани по задатој вредности

5. Запослени бира аранжман. (АПУСО)
6. Запослени позива систем да учита аранжман. (АПСО)

Опис акције: Запослени прво селектује аранжман из табеле који жељи да прикаже, кликом на ред у коме се тај аранжман налази, а потом кликом на дугме **Detalji** позива системску операцију **UcitajAranzman(Aranzman)**.

7. Систем учитава аранжман. (СО)
8. Систем приказује запосленом податке о аранжману и поруку: „Систем је учитао аранжман”. (ИА)



Слика 99 - Успешно учитан аранжман

A screenshot of a Windows application window titled 'Glavna forma Aranžman'. The main title is 'FORMA ARANŽMAN'. It contains several input fields:

- ID aranžmana:
- Tip aranžmana:
- Datum od (yyyy-MM-dd):
- Datum do (yyyy-MM-dd):
- Destinacija:

At the bottom are two teal-colored buttons:

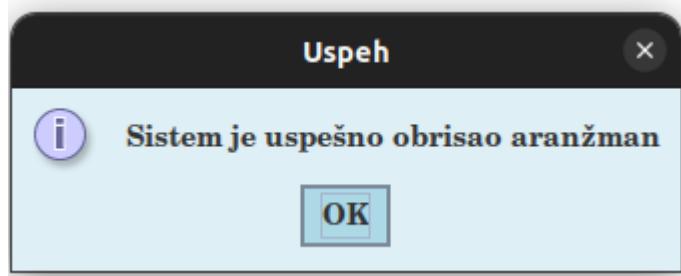
- Izmeni podatke o aranžmanu
- Obriši aranžman

Слика 100 - Учитан аранжман

9. Запослени позива систем да обрише аранжман. (АПСО)

Опис акције: Запослени кликом на дугме **Obriši aranžman** позива системску операцију **ObrisiAranzman(Aranzman)**.

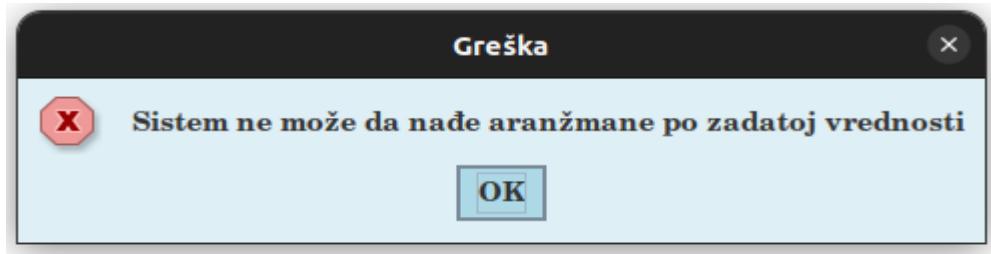
10. Систем брише аранжман. (СО)
11. Систем приказује запосленом поруку: „Систем је успешно обрисао аранжман.” (ИА)



Слика 101 – Успешно обрисан аранжман

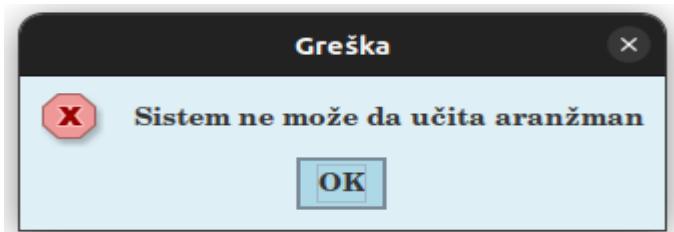
Алтернативна сценарија

4.1 Уколико систем не може да нађе аранжмане, он приказује запосленом поруку: „Систем не може да нађе аранжмане по задатој вредности”. Прекида се извршење сценарија. (ИА)



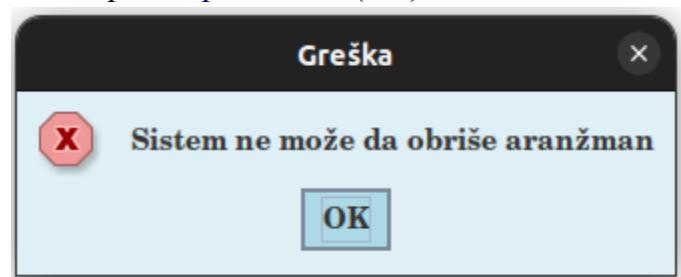
Слика 102 - Неуспешно учитавање аранжмана по задатој вредности

8.1 Уколико систем не може да учита аранжман, он приказује запосленом поруку: „Систем не може да учита аранжман”. (ИА)



Слика 103 - Неуспешно учитан аранжман

11.1 Уколико систем не може да обрише аранжман, он приказује запосленом поруку: „Систем не може да обрише аранжман”. (ИА)



Слика 104 - Неуспешно обрисан аранжман

СК9: Случај коришћења – Измена података о аранжману

Назив СК

Измена података о аранжману

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са аранжманима. Учитана је листа дестинација.

Pretraga Aranžmana

Pretraga aranžmana po tipu:

Pretraga **Resetuj pretragu**

tipAranžmana	DatumOd	DatumDo	Destinacija
ESKURZIJA	2024-03-20	2024-05-29	Rim, Italija
LETOVANJE	2024-09-09	2024-09-09	Lefkada, Grčka
PRVI_MAJ	2024-07-07	2024-07-07	Bansko, Bugarska
EVROPSKI_GRADOVI	2024-03-12	2024-03-21	Istanbul, Turska
ESKURZIJA	2024-05-26	2024-05-28	Rim, Italija
DALEKE_DESTINAC...	2024-08-08	2024-08-21	Tokio, Japan
IZLET	2024-05-05	2024-05-05	Temišvar, Румунија
EVROPSKI_GRADOVI	2024-09-10	2024-09-17	Pariz, Француска
NOVA_GODINA	2024-07-18	2024-07-24	Budimpešta, Мађарска
EVROPSKI_GRADOVI	2024-08-20	2024-08-27	Barcelona, Шпанија
EVROPSKI_GRADOVI	2024-09-10	2024-09-15	Prag, Чешка

Detalji

Слика 105 - Форма за приказ свих аранжмана

Основни сценарио СК

- Запослени уноси вредност по којој претражује аранжмане. (АПУСО)

Pretraga Aranžmana

Pretraga aranžmana po tipu: evropski

Pretraga **Resetuj pretragu**

tipAranžmana	DatumOd	DatumDo	Destinacija
ESKURZIJA	2024-03-20	2024-05-29	Rim, Italija
LETOVANJE	2024-09-09	2024-09-09	Lefkada, Grčka
PRVI_MAJ	2024-07-07	2024-07-07	Bansko, Bugarska
EVROPSKI_GRADOVI	2024-03-12	2024-03-21	Istanbul, Turska
ESKURZIJA	2024-05-26	2024-05-28	Rim, Italija
DALEKE_DESTINAC...	2024-08-08	2024-08-21	Tokio, Japan
IZLET	2024-05-05	2024-05-05	Temišvar, Rumunija
EVROPSKI_GRADOVI	2024-09-10	2024-09-17	Pariz, Francuska
NOVA_GODINA	2024-07-18	2024-07-24	Budimpešta, Mađarska
EVROPSKI_GRADOVI	2024-08-20	2024-08-27	Barcelona, Španija
EVROPSKI_GRADOVI	2024-09-10	2024-09-15	Prag, Češka

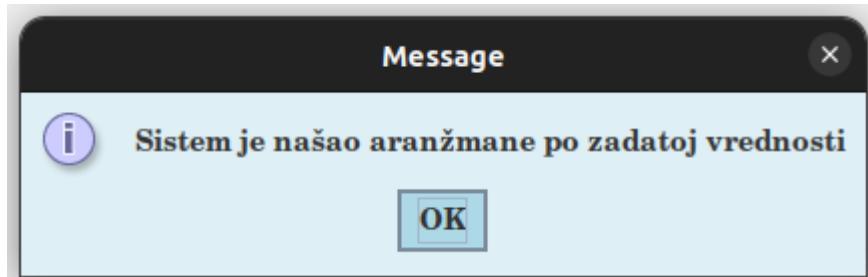
Detalji

Слика 106 - Унета вредност по којој се претражује аранжман

- Запослени позива систем да нађе аранжмане по задатој вредности. (АПСО)

Опис акције: Након што се форма за претрагу учита, запослени у поље за претрагу уноси тип аранжмана и притиском на дугме **Pretraga** позива се системска операција *PronadjiAranzmane(Aranzman, List<Aranzman>)*.

- Систем тражи аранжмане по задатој вредности. (СО)
- Систем приказује запосленом податке о аранжманима и поруку: „Систем је нашао аранжмане по задатој вредности“. (ИА)



Слика 107 - Успешно учитавање аранжмана по задатој вредности

The image shows a search interface titled 'Pretraga Aranžmana'. At the top is a search bar with the placeholder 'Pretraga aranžmana po tipu:' and a text input field containing 'evropski'. To the right of the input field are two buttons: a teal-colored 'Pretraga' (Search) button and a white 'Resetuj pretragu' (Reset search) button. Below the search bar is a table with four columns: 'tipAranžmana', 'DatumOd', 'DatumDo', and 'Destinacija'. The table contains four rows of data:

tipAranžmana	DatumOd	DatumDo	Destinacija
EVROPSKI_GRADOVI	2024-03-12	2024-03-21	Istanbul, Turska
EVROPSKI_GRADOVI	2024-09-10	2024-09-17	Pariz, Francuska
EVROPSKI_GRADOVI	2024-08-20	2024-08-27	Barselona, Španija
EVROPSKI_GRADOVI	2024-09-10	2024-09-15	Prag, Češka

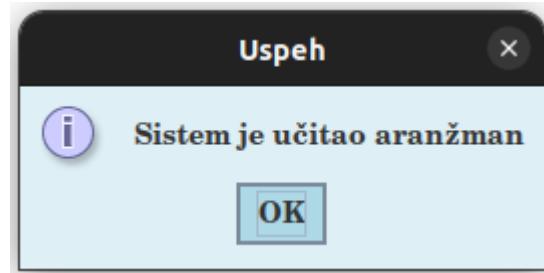
At the bottom right of the interface is a teal-colored button labeled 'Detalji' (Details).

Слика 108 - Пронађени аранжмани по задатој вредности

5. Запослени бира аранжман. (АПУСО)
6. Запослени позива систем да чита аранжман. (АПСО)

Опис акције: Запослени прво селектује аранжман из табеле који жељи да прикаже, кликом на ред у коме се тај аранжман налази, а потом кликом на дугме *Detalji* позива системску операцију *UcitajAranzman(Aranzman)*.

7. Систем учитава аранжман. (СО)
8. Систем приказује запосленом податке о аранжману и поруку: „Систем је учитао аранжман”. (ИА)



Слика 109 - Успешно учитан аранжман

A screenshot of a Windows application window titled 'Glavna forma Aranžman'. The main title is 'FORMA ARANŽMAN'. The form contains the following fields:

- ID aranžmana:
- Tip aranžmana:
- Datum od (yyyy-MM-dd):
- Datum do (yyyy-MM-dd):
- Destinacija:

At the bottom are two teal-colored buttons:

- Izmeni podatke o aranžmanu
- Obriši aranžman

Слика 110 - Учитан аранжман

9. Запослени уноси (мења) податке о аранжману. (АПУСО)
10. Запослени контролише да ли је коректно унео податке о аранжману. (АХСО)

Glavna forma Aranžman

FORMA ARANŽMAN

ID aranžmana:	9
Tip aranžmana:	ESKURZIJA
Datum od (yyyy-MM-dd):	2024-07-12
Datum do (yyyy-MM-dd):	2024-07-21
Destinacija:	Istanbul, Turska

[Izmeni podatke o aranžmanu](#)

[Obriši aranžman](#)

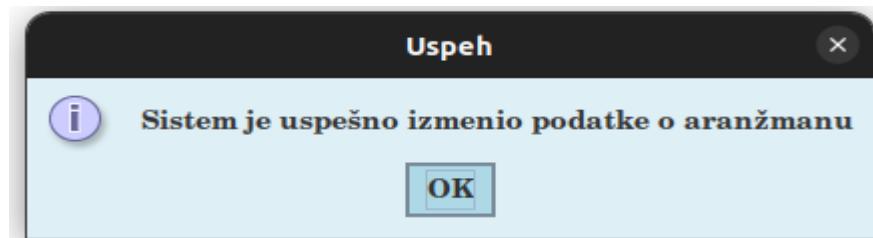
Слика 111 - Форма за измену аранжмана

11. Запослени позива систем да измени податке о аранжману. (АПСО)

Опис акције: Запослени кликом на дугме *Izmeni podatke o aranžmanu* позива системску операцију *IzmeniAranzman(Aranzman)*.

12. Систем мења податке о аранжману. (СО)

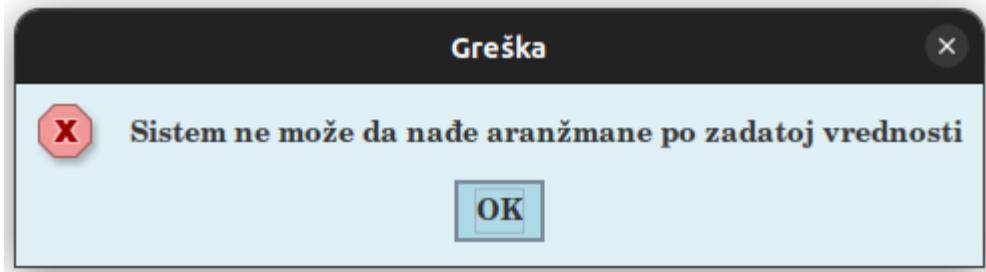
13. Систем приказује запосленом изменјени аранжман и поруку: „Систем је успешно изменио податке о аранжману.” (ИА)



Слика 112 - Успешно изменен аранжман

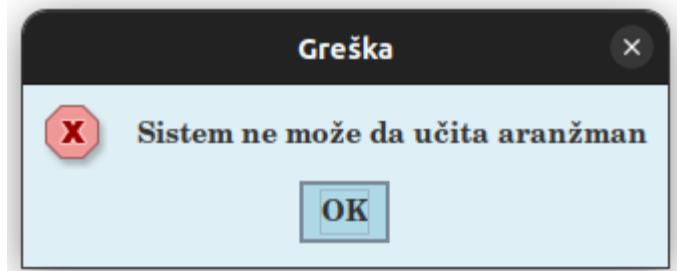
Алтернативна сценарија

4.1 Уколико **систем** не може да нађе **аранжмане**, он приказује **запосленом** поруку: „**Систем** не може да нађе **аранжмане** по задатој вредности”. Прекида се извршење сценарија. (ИА)



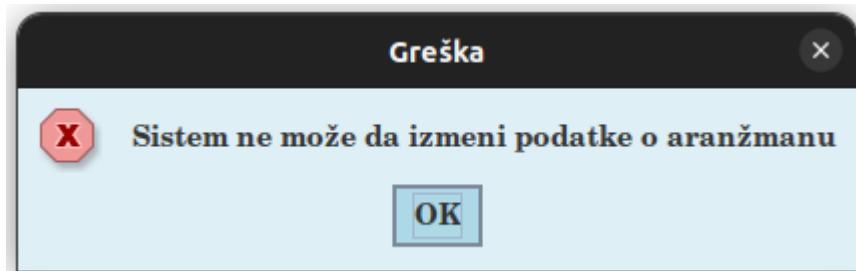
Слика 113 - Неуспешно учитавање аранжмана по задатој вредности

8.1 Уколико **систем** не може да учита **аранжман**, он приказује **запосленом** поруку: „**Систем** не може да учита **аранжман**”. (ИА)



Слика 114 - Неуспешно учитан аранжман

13.1 Уколико **систем** не може да измени податке о аранжману, он приказује **запосленом** поруку: „**Систем** не може да измени податке о **аранжману**. (ИА)



Слика 115 – Неуспешно изменењен аранжман

СК10: Случај коришћења – Креирање резервације (сложен СК)

Назив СК

Креирање резервације

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са резервацијама. Учитана је листа корисника и аранђмана.

Glavna forma Rezervacija

FORMA REZERVACIJA

Ulogovani zaposleni: Natalija Brankovic

Šifra rezervacije:		Datum rezervacije (yyyy-MM-dd):		Tip sobe:	1/1
--------------------	--	---------------------------------	--	-----------	-----

Aranđmani:

tipAranžmana	DatumOd	DatumDo	Destinacija
ESKURZIJA	2024-03-20	2024-05-29	Rim, Italija
LETOVANJE	2024-09-09	2024-09-09	Lefkada, Грека
PRVI_MAJ	2024-07-07	2024-07-07	Bansko, Бугарска
ESKURZIJA	2024-07-12	2024-07-21	Istanbul, Турска
DALEKE_DESTINACIJE	2024-08-08	2024-08-21	Tokio, Јапан
IZLET	2024-05-05	2024-05-05	Temišvar, Румунија
EVROPSKI_GRADOVI	2024-09-10	2024-09-17	Pariz, Француска
NOVA_GODINA	2024-07-18	2024-07-24	Budimpešta, Мађарска
EVROPSKI_GRADOVI	2024-08-20	2024-08-27	Barcelona, Шпанија
EVROPSKI_GRADOVI	2024-09-10	2024-09-15	

Raspored Soba

Korisnik:	Ana Milošević	Osiguranje:	DA	Ukupna cena (€):	
-----------	---------------	-------------	----	------------------	--

Korisnik	Osiguranje	Ukupna cena

Obriši raspored Dodaj raspored

Otkaži Sačuvaj

Слика 116 – Форма за креирање резервације

Основни сценарио СК

1. Запослени уноси податке о резервацији. (АПУСО)
2. Запослени контролише да ли је коректно унео податке о резервацији. (АНСО)

Glavna forma Rezervacija

FORMA REZERVACIJA

Ulogovani zaposleni: Natalija Brankovic

Šifra rezervacije:	bc89065	Datum rezervacije (yyyy-MM-dd):	2024-06-09	Tip sobe:	1/3
--------------------	---------	---------------------------------	------------	-----------	-----

Aranžmani:

tipAranžmana	DatumOd	DatumDo	Destinacija
LETOVANJE	2024-09-09	2024-09-09	Lefkada, Grčka
PRVI_MAJ	2024-07-07	2024-07-07	Bansko, Bugarska
ESKURZIJA	2024-07-12	2024-07-21	Istanbul, Turska
DALEKE_DESTINACIJE	2024-08-08	2024-08-21	Tokio, Japan
IZLET	2024-05-05	2024-05-05	Temišvar, Румунија
EVROPSKI_GRADOVI	2024-09-10	2024-09-17	Pariz, Француска
NOVA_GODINA	2024-07-18	2024-07-24	Budimpešta, Мађарска
EVROPSKI_GRADOVI	2024-08-20	2024-08-27	Barcelona, Шпанија
EVROPSKI_GRADOVI	2024-09-10	2024-09-15	Prag, Чешка

Raspored Soba

Korisnik:	Nevena Todorovic	Osiguranje:	NE	Ukupna cena (€): 210
-----------	------------------	-------------	----	----------------------

Korisnik	Osiguranje	Ukupna cena
Isidora Ivkovic	DA	230.0
Sofija Jankovic	DA	240.0

Obriši raspored Dodaj raspored

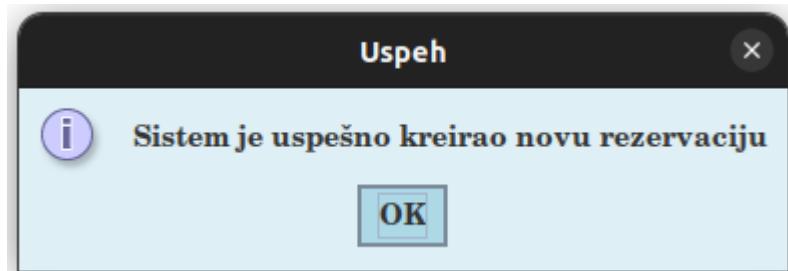
Otkaži Sačuvaj

Слика 117 – Попњена форма за креирање резервације

3. Запослени позива систем да креира нову резервацију. (АПСО)

Опис акције: Кликом на дугме **Сачувати**, запослени позива системску операцију **KreirajRezervaciju(Rezervacija)**.

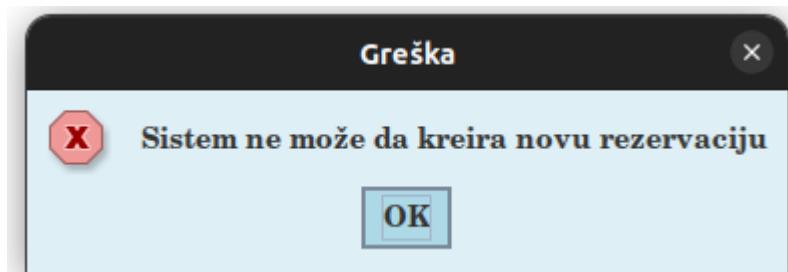
4. Систем памти податке о резервацији. (СО)
5. Систем приказује запосленом креирану резервацију и поруку: „Систем је успешно креирао нову резервацију“. (ИА)



Слика 118 – Успешно креирање резервације

Алтернативна сценарија

- 5.1 Уколико систем не може да креира нову резервацију, он приказује запосленом поруку „Систем не може да креира нову резервацију“. (ИА)



Слика 119 – Неуспешно креирање резервације

СК11: Случај коришћења – Измена резервације (сложен СК)

Назив СК

Измена резервације

Актори СК

Запослени

Учесници СК

Запослени и систем (програм)

Предуслов: Систем је укључен и запослени је улогован под својом шифром. Систем приказује форму за рад са резервацијама. Учитана је листа корисника и аранжмана.

Pretraga rezervacija

Pretraga rezervacija po šifri:

Pretraga **Resetuj pretragu**

Zaposleni	Šifra rezervacije	Datum rezervacije	Tip sobe	Aranžman
Natalija Brankovic	bv09878	2024-05-05	1/3	ESKURZIJA, Istanbul, T...
Natalija Brankovic	bcxx	2024-06-06	1/4	DALEKE_DESTINACIJ...
Natalija Brankovic	RI7897	2024-05-06	1/3	ESKURZIJA, Rim, Italija
Natalija Brankovic	bxafas	2024-09-09	1/1	PRVI_MAJ, Bansko, Bug...
Natalija Brankovic	bc89065	2024-06-09	1/3	EVROPSKI_GRADOVI, ...
Natalija Brankovic	cn8976	2024-06-01	1/2	NOVA_GODINA, Budim...
Nikola Jovanovic	nh768	2024-05-16	1/1	IZLET, Temišvar, Rumu...
Nikola Jovanovic	ji987	2024-03-07	1/2	EVROPSKI_GRADOVI, ...

Detalji

Слика 120 – Форма за приказ свих резервација

Основни сценарио СК

1. Запослени уноси вредност по којој претражује резервације. (АПУСО)

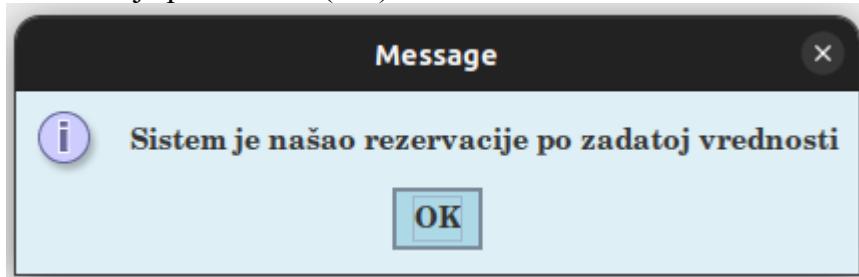
Zaposleni	Šifra rezervacije	Datum rezervacije	Tip sobe	Aranžman
Natalija Brankovic	bv09878	2024-05-05	1/3	ESKURZIJA, Istanbul, T...
Natalija Brankovic	bcxx	2024-06-06	1/4	DALEKE_DESTINACIJ...
Natalija Brankovic	RI7897	2024-05-06	1/3	ESKURZIJA, Rim, Italija
Natalija Brankovic	bxafas	2024-09-09	1/1	PRVI_MAJ, Bansko, Bug...
Natalija Brankovic	bc89065	2024-06-09	1/3	EVROPSKI_GRADOVI, ...
Natalija Brankovic	cn8976	2024-06-01	1/2	NOVA_GODINA, Budim...
Nikola Jovanovic	nh768	2024-05-16	1/1	IZLET, Temišvar, Rumu...
Nikola Jovanovic	ji987	2024-03-07	1/2	EVROPSKI_GRADOVI, ...

Слика 121 – Унета вредност за претрагу резервација

2. Запослени позива систем да нађе резервације по задатој вредности. (АПСО)

Опис акције: Након што се форма за претрагу учита, запослени у поље за претрагу уноси шифру резервације и кликом на дугме **Pretraga** позива се системска операција **PronadjiRezervacije(Rezervacija, List<Rezervacija >)**.

3. Систем тражи резервације по задатој вредности. (СО)
4. Систем приказује запосленом податке о резервацијама и поруку: „Систем је нашао резервације по задатој вредности“. (ИА)



Слика 122 – Успешно учитавање резервација по задатој вредности

Pretraga rezervacija

Pretraga rezervacija

Pretraga rezervacija po šifri:	<input type="text" value="bc"/>	Pretraga		
		Resetuj pretragu		
Zaposleni	Šifra rezervacije	Datum rezervacije	Tip sobe	Aranžman
Natalija Brankovic	bc89065	2024-06-09	1/3	EVROPSKI_GRADOVI, ...
Natalija Brankovic	bctxx	2024-06-06	1/4	DALEKE_DESTINACIJ...

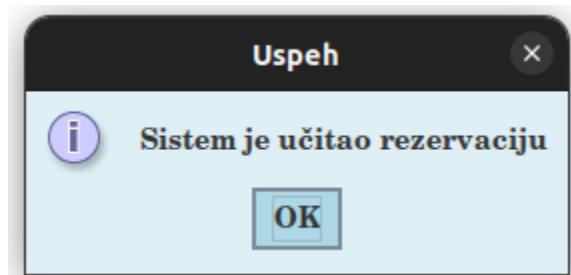
Detalji

Слика 123 – Приказ резервација по задатој вредности

5. Запослени бира резервацију. (АПУСО)
6. Запослени позива систем да учита резервацију. (АПСО)

Опис акције: Запослени прво селектује резервацију из табеле коју жели да измени, кликом на ред у коме се та резервација налази, а потом кликом на дугме *Detalji* позива системску операцију *UcitajRezervaciju(Rezervacija)*.

7. Систем учитава резервацију. (СО)
8. Систем приказује запосленом податке о резервацији и поруку: „Систем је учитао резервацију”. (ИА)



Слика 124 – Успешно учитана резервација

Glavna forma Rezervacija X

FORMA REZERVACIJA

B

Ulogovani zaposleni: Natalija Brankovic

Šifra rezervacije:	ji987	Datum rezervacije (yyyy-MM-dd):	2024-03-07	Tip sobe:	1/2
--------------------	-------	---------------------------------	------------	-----------	-----

Aranžmani:

tipAranžmana	DatumOd	DatumDo	Destinacija
ESKURZIJA	2024-03-20	2024-05-29	Rim, Italija
LETOVANJE	2024-09-09	2024-09-09	Lefkada, Grčka
PRVI_MAJ	2024-07-07	2024-07-07	Bansko, Bugarska
ESKURZIJA	2024-07-12	2024-07-21	Istanbul, Turska
DALEKE_DESTINACIJE	2024-08-08	2024-08-21	Tokio, Japan
IZLET	2024-05-05	2024-05-05	Temišvar, Rumunija
EVROPSKI_GRADOVI	2024-09-10	2024-09-17	Pariz, Francuska
NOVA_GODINA	2024-07-18	2024-07-24	Budimpešta, Mađarska
EVROPSKI_GRADOVI	2024-08-20	2024-08-27	Barcelona, Španija
EVROPSKI_GRADOVI	2024-08-10	2024-08-17	

Raspored Soba

Korisnik:	Ana Milošević	Osiguranje:	DA	Ukupna cena (€):	
-----------	---------------	-------------	----	------------------	--

Korisnik	Osiguranje	Ukupna cena
Sanja Krstic	DA	350.0
Ivan Ognjanović	DA	350.0

Obriši raspored Dodaj raspored Izmeni podatke

Слика 125 – Учитана резервација

9. Запослени уноси (мења) податке о резервацији. (АПУСО)
10. Запослени контролише да ли је коректно унео податке о резервацији. (АХСО)

Glavna forma Rezervacija

FORMA REZERVACIJA

1

Ulogovani zaposleni: Natalija Brankovic

Šifra rezervacije: Datum rezervacije (yyyy-MM-dd): Tip sobe:

Aranžmani:

tipAranžmana	DatumOd	DatumDo	Destinacija
ESKURZIJA	2024-03-20	2024-05-29	Rim, Italija
LETOVANJE	2024-09-09	2024-09-09	Lefkada, Grčka
PRVI_MAJ	2024-07-07	2024-07-07	Bansko, Bugarska
ESKURZIJA	2024-07-12	2024-07-21	Istanbul, Turska
DALEKE_DESTINACLJE	2024-08-08	2024-08-21	Tokio, Japan
IZLET	2024-05-05	2024-05-05	Temišvar, Rumunija
EVROPSKI_GRADOVI	2024-09-10	2024-09-17	Pariz, Francuska
NOVA_GODINA	2024-07-18	2024-07-24	Budimpešta, Madarska
EVROPSKI_GRADOVI	2024-08-20	2024-08-27	Barselona, Španija
EVROPSKI_GRADOVI	2024-08-10	2024-08-15	...

Raspored Soba

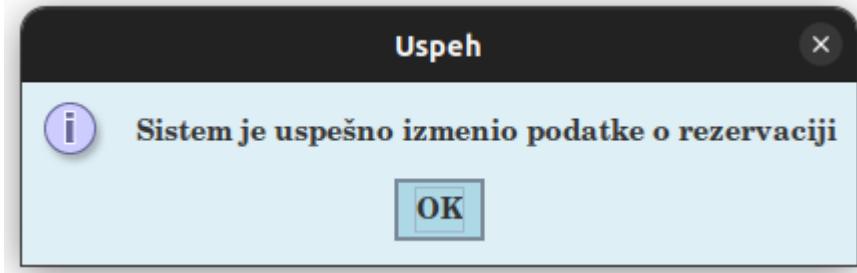
Korisnik: Osiguranje: Ukupna cena (€):

Korisnik	Osiguranje	Ukupna cena
Ana Milošević	DA	111.0
Nikola Ilić	DA	222.0
Isidora Ivkovic	DA	342.0
Sanja Krstic	NE	240.0

Obriši raspored **Dodaj raspored** **Izmeni podatke**

Слика 126 – Форма за измену резервације

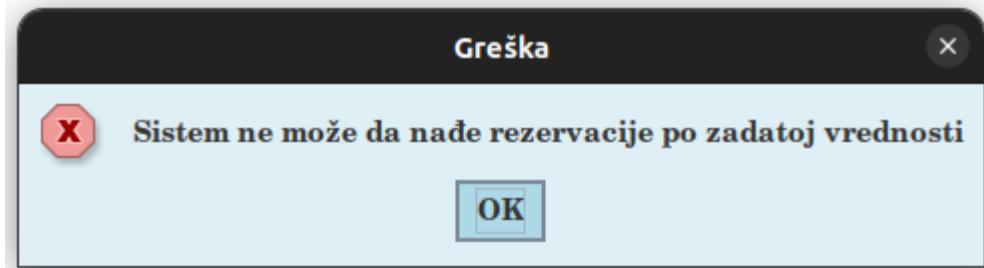
11. Запослени позива систем да запамти податке о резервацији. (АПСО)
12. Систем памти податке о резервацији. (СО)
13. Систем приказује запосленом запамћену резервацију и поруку: „Систем је успешно изменио податке о резервацији.” (ИА)



Слика 127 – Успешно изменењена резервација

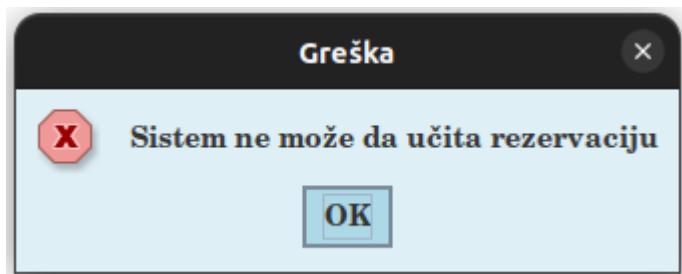
Алтернативна сценарија

4.1 Уколико **систем** не може да нађе резервације, он приказује **запосленом** поруку: „**Систем** не може да нађе **резервације** по задатој вредности”. Прекида се извршење сценарија. (ИА)



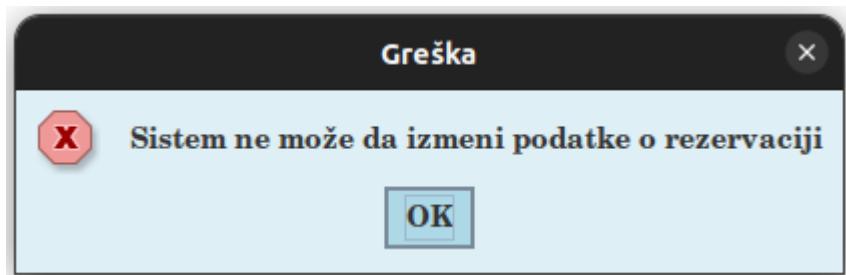
Слика 128 – Неуспешно учитавање резервација

8.1 Уколико **систем** не може да учита податке о **резервацији**, он приказује **запосленом** поруку „**Систем** не може да учита **резервацију**”. Прекида се извршење сценарија. (ИА)



Слика 129 – Неуспешно учитана резервација

13.1 Уколико **систем** не може да измени податке о **резервацији**, он приказује **запосленом** поруку „**Систем** не може да измене податке о **резервацији**”. (ИА)



Слика 130 – Неуспешно изменењена резервација

4.3.2.2. Пројектовање контролера корисничког интерфејса

Контролер корисничког интерфејса је одговоран за [1]:

- Прихваташање графичких објеката од екранске форме
- Конвертовање података који се налазе у графичким објектима у доменске објекте који ће бити прослеђени преко мреже до апликационог сервера
- Конвертовање доменских објеката у графичке објекте и њихово прослеђивање до екранске форме

4.3.3. Пројектовање апликационе логике

Апликациони сервери су одговорни да обезбеде сервисе који омогућавају реализацију апликационе логике софтверског система.

Пројектовани апликациони сервер садржи [1]:

- део за комуникацију са клијентом
- контролер апликационе логике
- део за комуникацију са складиштем података
- део који садржи пословну логику

Комуникација са клијентима

У делу за комуникацију, серверски сокет се подиже како би ослушкивао мрежу. Клијентски и серверски сокети успостављају комуникацију, након чега сервер генерише нит која успоставља двосмерну везу са клијентом.

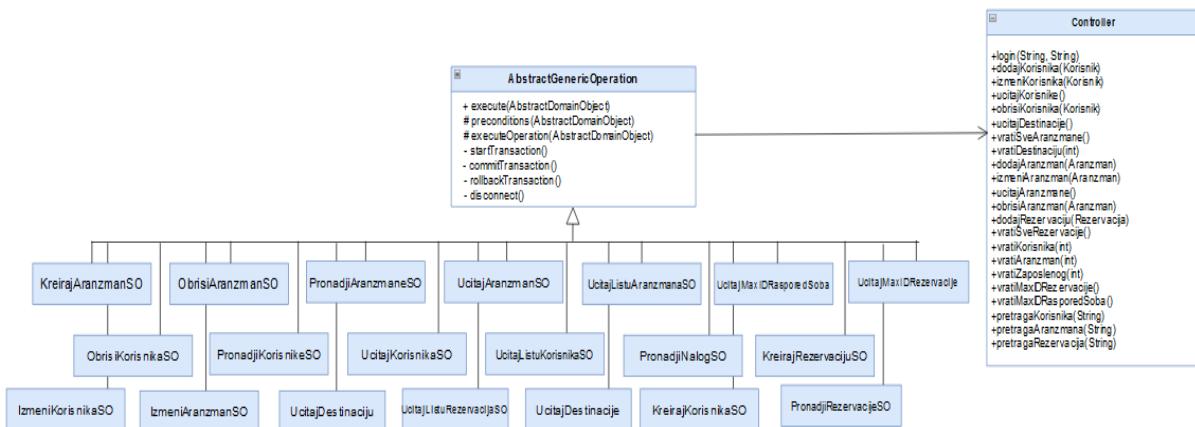
Слање и примање података од клијента се врши разменом објеката класа **Request** и **Response** преко сокета. Клијент шаље захтев за извршење неке системске операције до нити која је повезана са њим. Нит прихваташа захтев и прослеђује га контролеру апликационе логике. Након извршења операције, резултат се преко контролера враћа до нити клијента, која затим тај резултат шаље назад клијенту.

4.3.3.1. Контролер апликационе логике

Контролер апликационе логике приhvата захтев за извршење системске операције од нити клијента и даље га преусмерава до класа које су одговорне за извршење системских операција. Након извршења системске операције, контролер апликационе логике приhvата резултат и прослеђује га позиваоцу (нити клијента).

Свака системска операција имплементирана је као засебна класа. Класе које су одговорне за извршење системских операција, наслеђују апстрактну класу *AbstractGenericOperation* како би могле да се повежу са базом и како би се њихово извршење пратило као трансакција.

Следећа слика даје опис система након фазе пројектовања комуникације са клијентима и контролера апликационе логике.



Слика 131 – Пројектовање контролера апликационе логике

4.3.3.2. Пословна логика

Пословна логика је описана структуром (доменским класама) и понашањем (системским операцијама).

4.3.3.2.1 Пројектовање понашања софтверског система – системске операције

За сваку системску операцију треба направити концептуална решења која су директно повезана са логиком проблема.

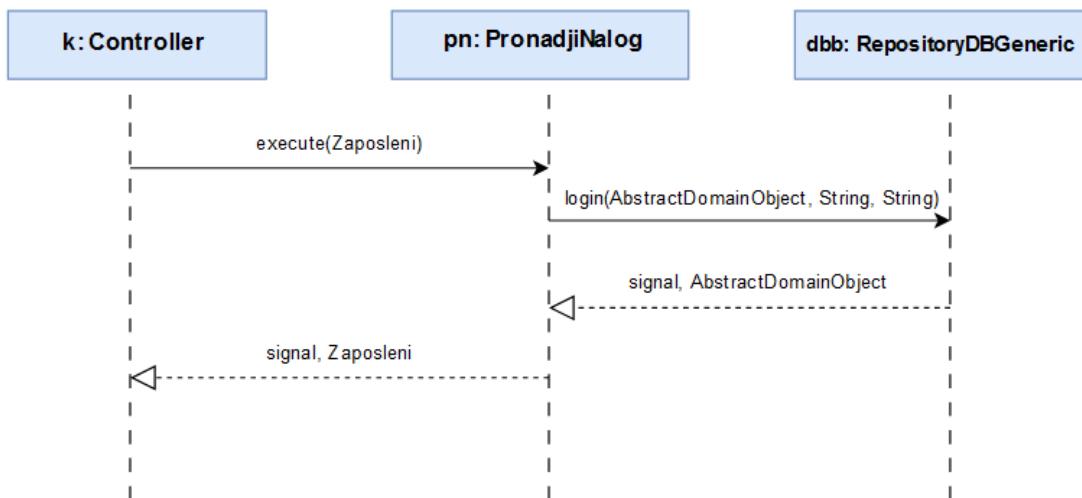
За сваки уговор пројектује се концептуално решење.

Уговор УГ1: signal **PronadjiNalog(Nalog)**

Веза са СК: CK1

Предуслови: /

Постуслови: Налог је пронађен, запослени је пријављен на систем.



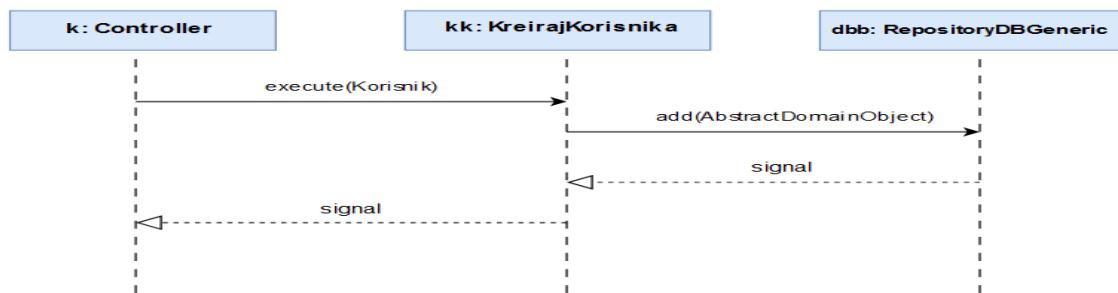
Слика 132 – Дијаграм секвенци за уговор *PronadjiNalog*

Уговор УГ2: signal KreirajKorisnika(Korisnik)

Веза са СК: CK2

Предуслови: Вредносна и структурна ограничења над објектом Korisnik морају да буду задовољена.

Постуслови: Направљен је нови корисник.



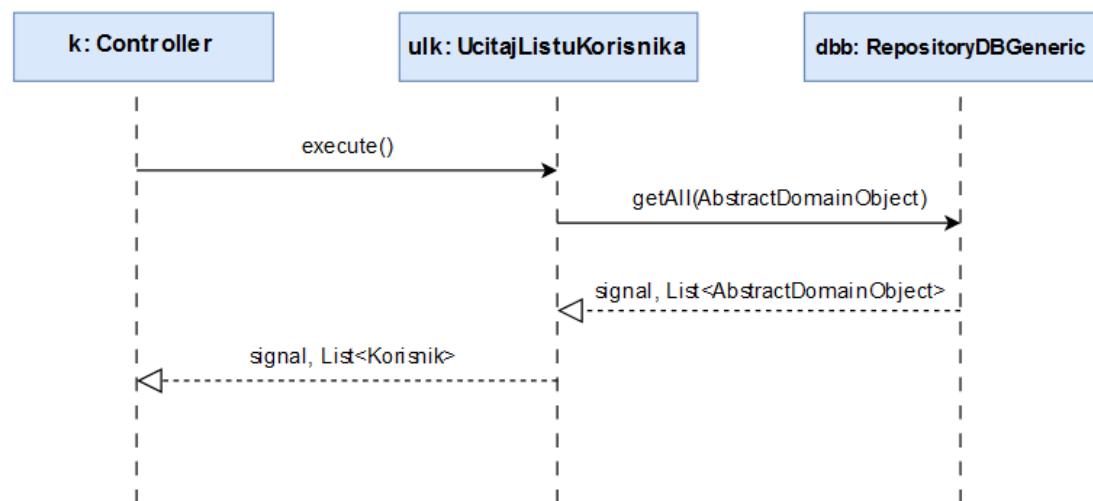
Слика 133 – Дијаграм секвенци за уговор KreirajKorisnika

Уговор УГ3: signal UcitajListuKorisnika(List<Korisnik>)

Веза са СК: CK10, CK11

Предуслови: /

Постуслови: /



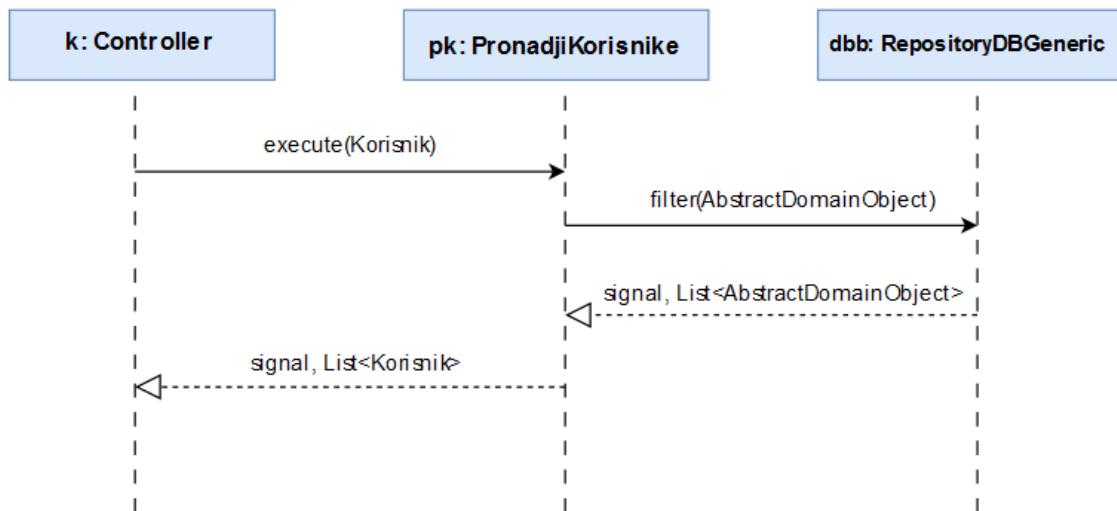
Слика 134 – Дијаграм секвенци за уговор UcitajListuKorisnika

Уговор УГ4: signal **PronadjiKorisnike(Korisnik, List<Korisnik>)**

Веза са СК: CK3, CK4, CK5

Предуслови: /

Постуслови: /



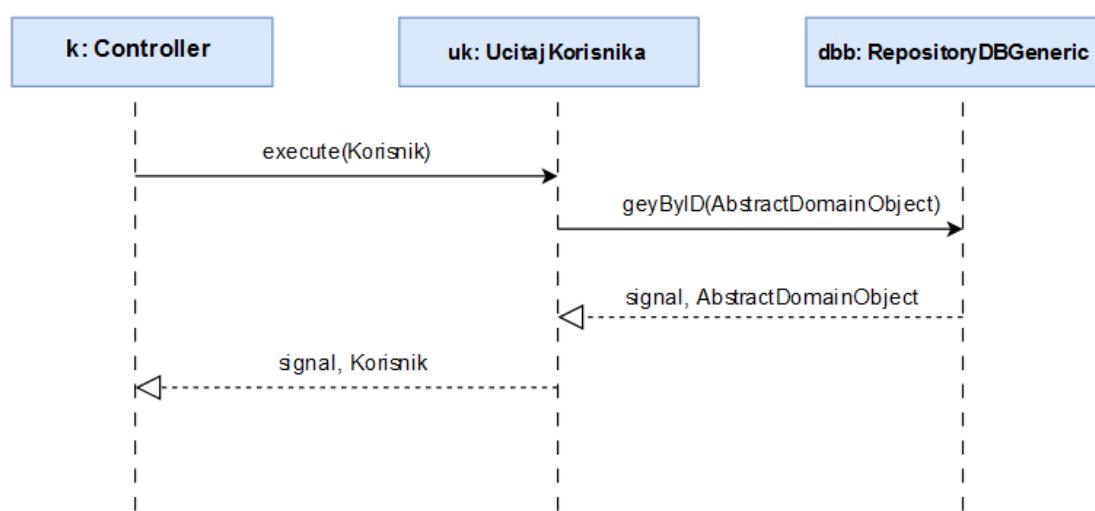
Слика 135 – Дијаграм секвенци за уговор *PronadjiKorisnike*

Уговор УГ5: signal **UcitajKorisnika(Korisnik)**

Веза са СК: CK3, CK4, CK5

Предуслови: /

Постуслови: /



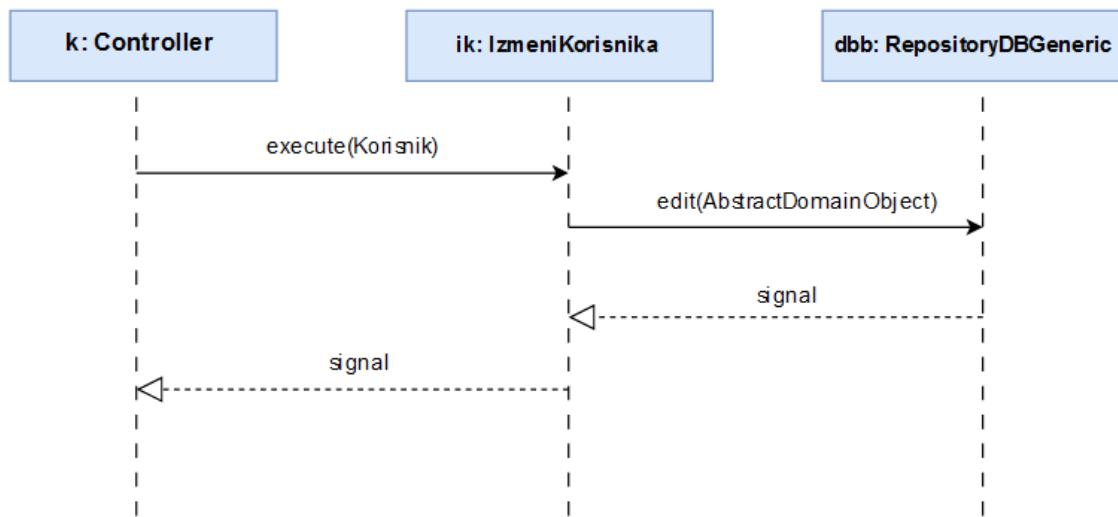
Слика 136 – Дијаграм секвенци за уговор *UcitajKorisnika*

Уговор УГ6: signal **IzmeniKorisnika(Korisnik)**

Веза са СК: СК4

Предуслови: Вредносна и структурна ограничења над објектом Korisnik морају да буду задовољена.

Постуслови: Подаци о кориснику су изменjeni.



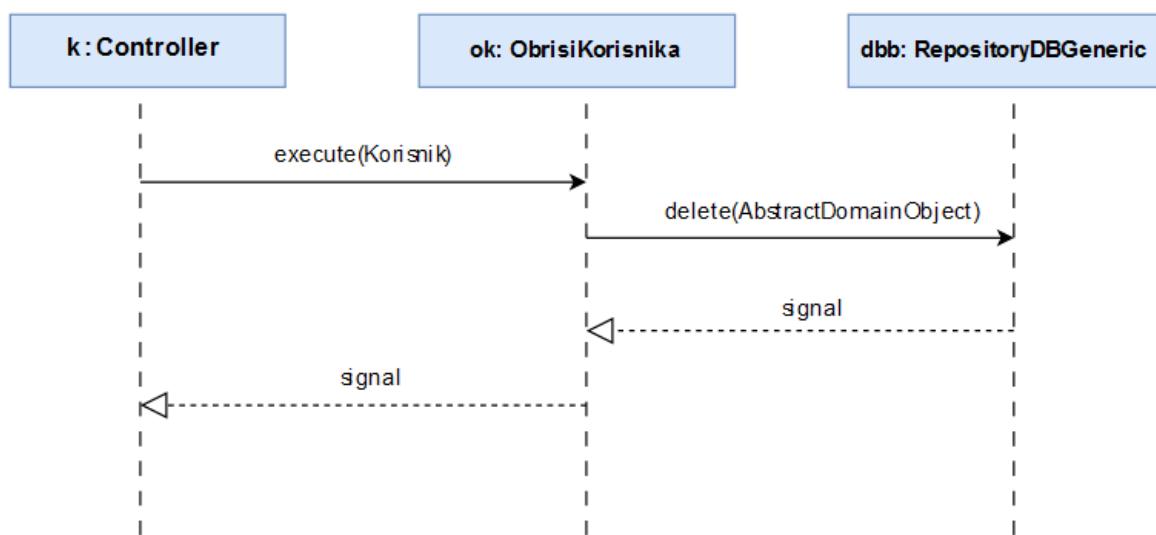
Слика 137 – Дијаграм секвенци за уговор *IzmeniKorisnika*

Уговор УГ7: signal **ObrisniKorisnika(Korisnik)**

Веза са СК: СК5

Предуслови: Структурна ограничења над објектом Korisnik морају да буду задовољена.

Постуслови: Корисник је обрисан.



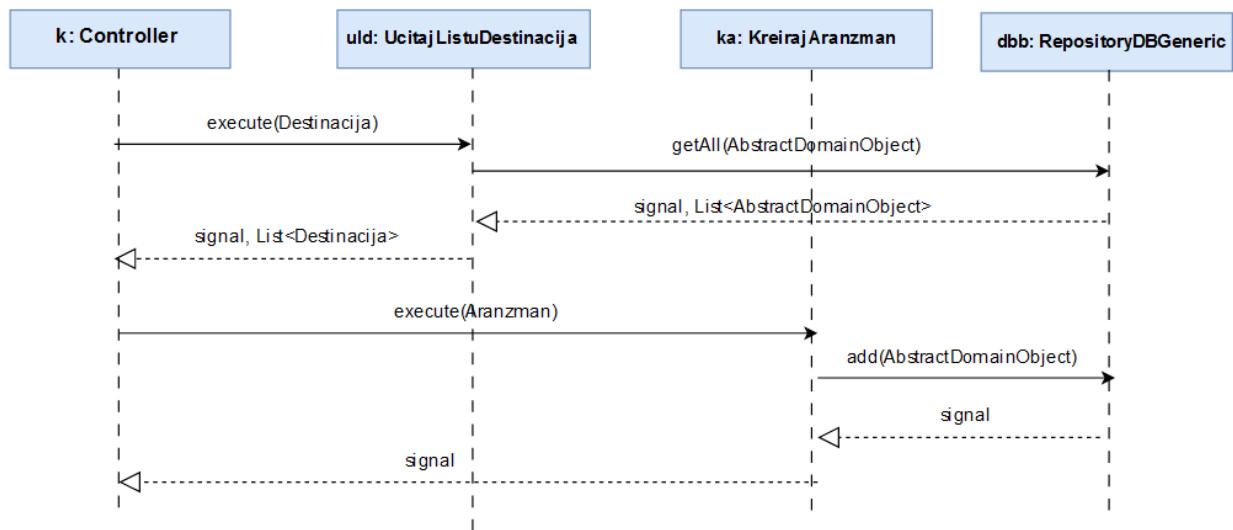
Слика 138 – Дијаграм секвенци за уговор *ObrisniKorisnika*

Уговор УГ8: signal KreirajAranzman(Aranzman)

Веза са СК: СК6

Предуслови: Вредносна и структурна ограничења над објектом Aranzman морају да буду задовољена.

Постуслови: Направљен је нови аранжман.



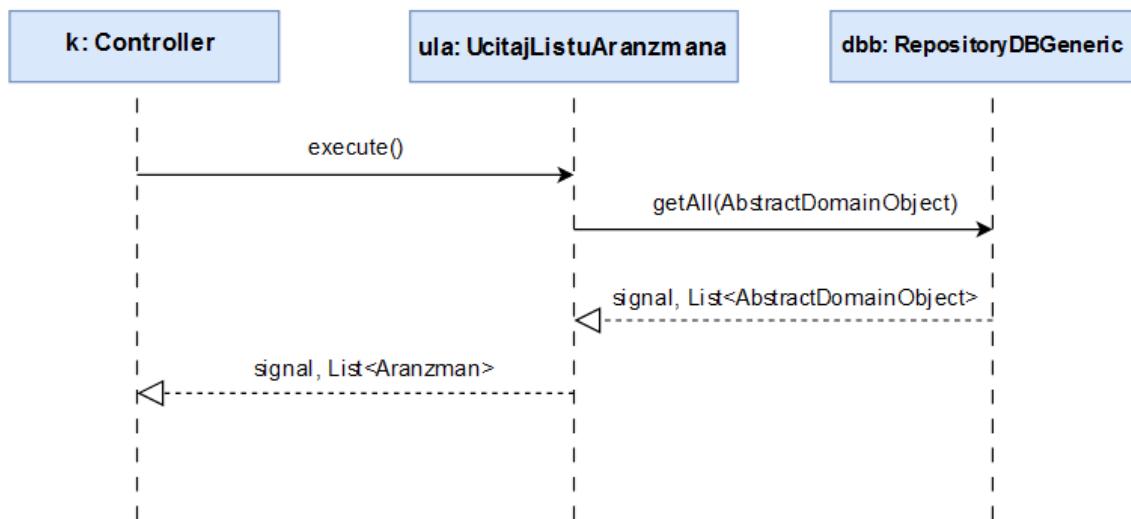
Слика 139 – Дијаграм секвенци за уговор KreirajAranzman

Уговор УГ9: signal UcitajListuAranzmana(List<Aranzman>)

Веза са СК: СК10, СК11

Предуслови: /

Постуслови: /



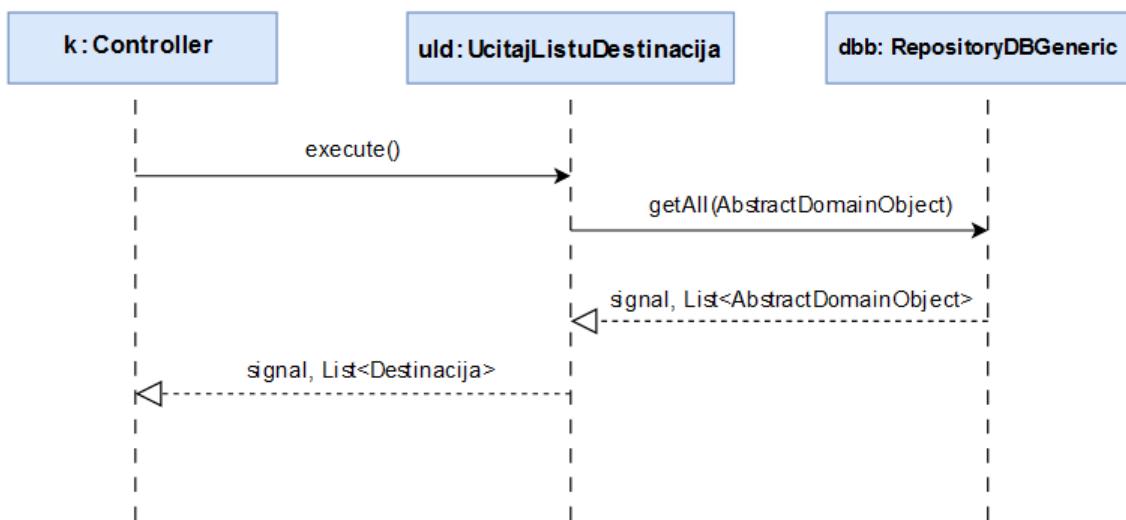
Слика 140 – Дијаграм секвенци за уговор UcitajListuAranzmana

Уговор УГ10: signal **UcitajListuDestinacija(List<Destinacija>)**

Веза са СК: СК6, СК9

Предуслови: /

Постуслови: /



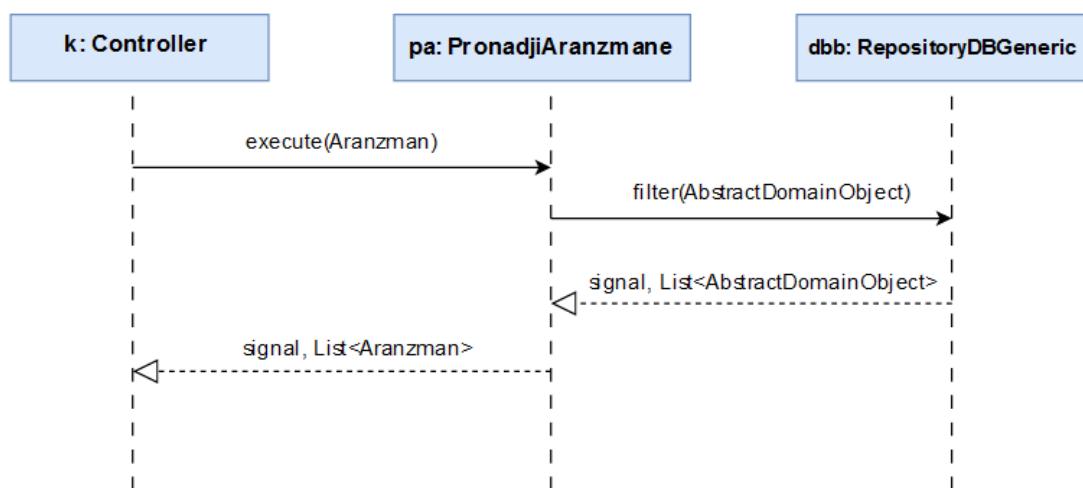
Слика 141 – Дијаграм секвенци за уговор *UcitajListuDestinacija*

Уговор УГ11: signal **PronadjiAranzmane(Aranzman, List<Aranzman>)**

Веза са СК: СК7, СК8, СК9

Предуслови: /

Постуслови: /



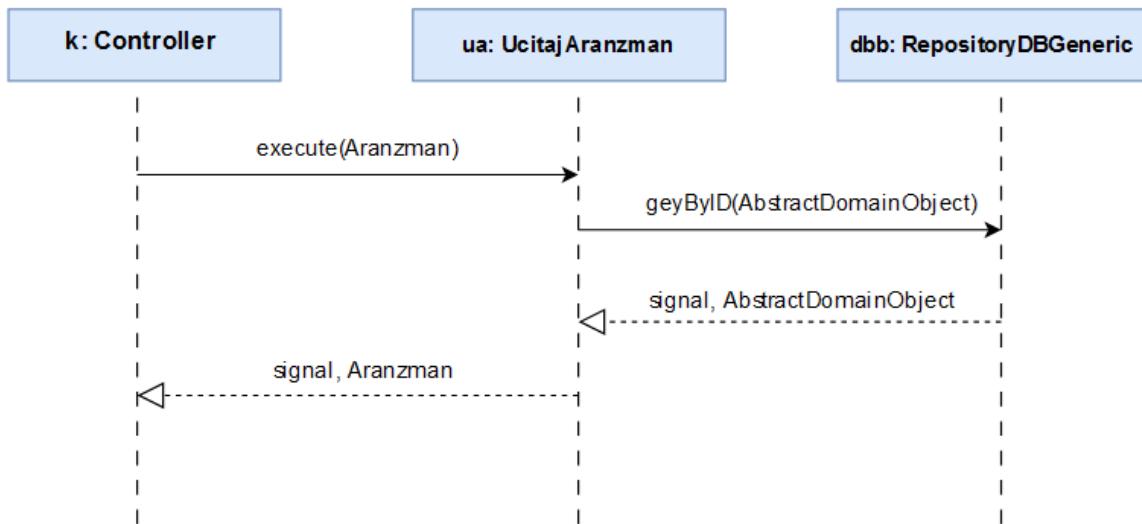
Слика 142 – Дијаграм секвенци за уговор *PronadjiAranzmane*

Уговор УГ12: signal UcitajAranzman(Aranzman)

Веза са СК: CK7, CK8, CK9

Предуслови: /

Постуслови: /



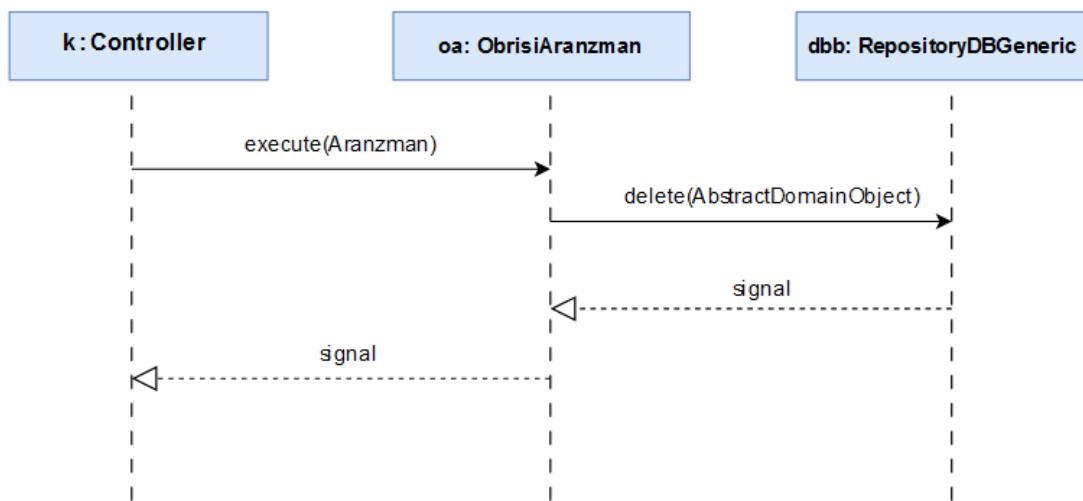
Слика 143 – Дијаграм секвенци за уговор *UcitajAranzman*

Уговор УГ13: signal ObrisiAranzman(Aranzman)

Веза са СК: CK8

Предуслови: Структурна ограничења над објектом Aranzman морају да буду задовољена.

Постуслови: Аранжман је обрисан.



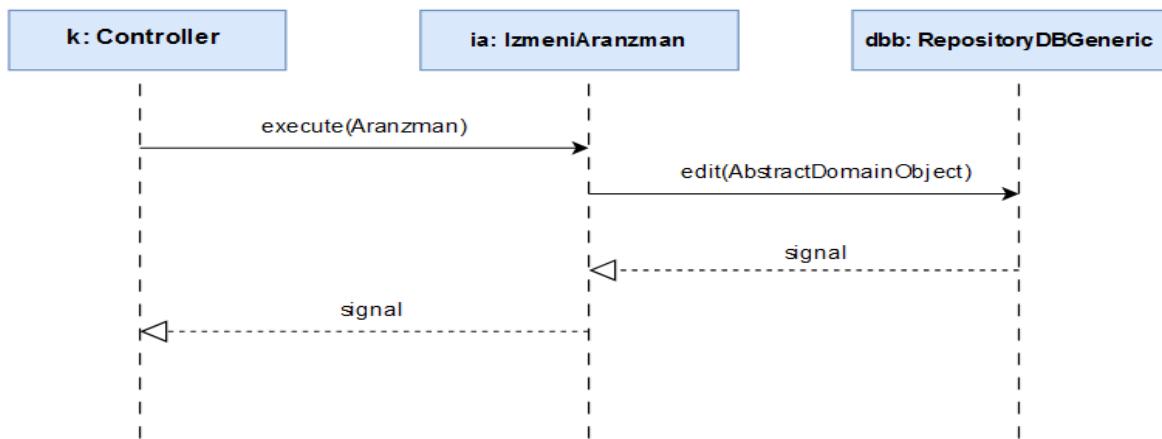
Слика 144 – Дијаграм секвенци за уговор *ObrisiAranzman*

Уговор УГ14: signal IzmeniAranzman(Aranzman)

Веза са СК: СК9

Предуслови: Вредносна и структурна ограничења над објектом Aranzman морају да буду задовољена.

Постуслови: Подаци о аранжману су изменењени.



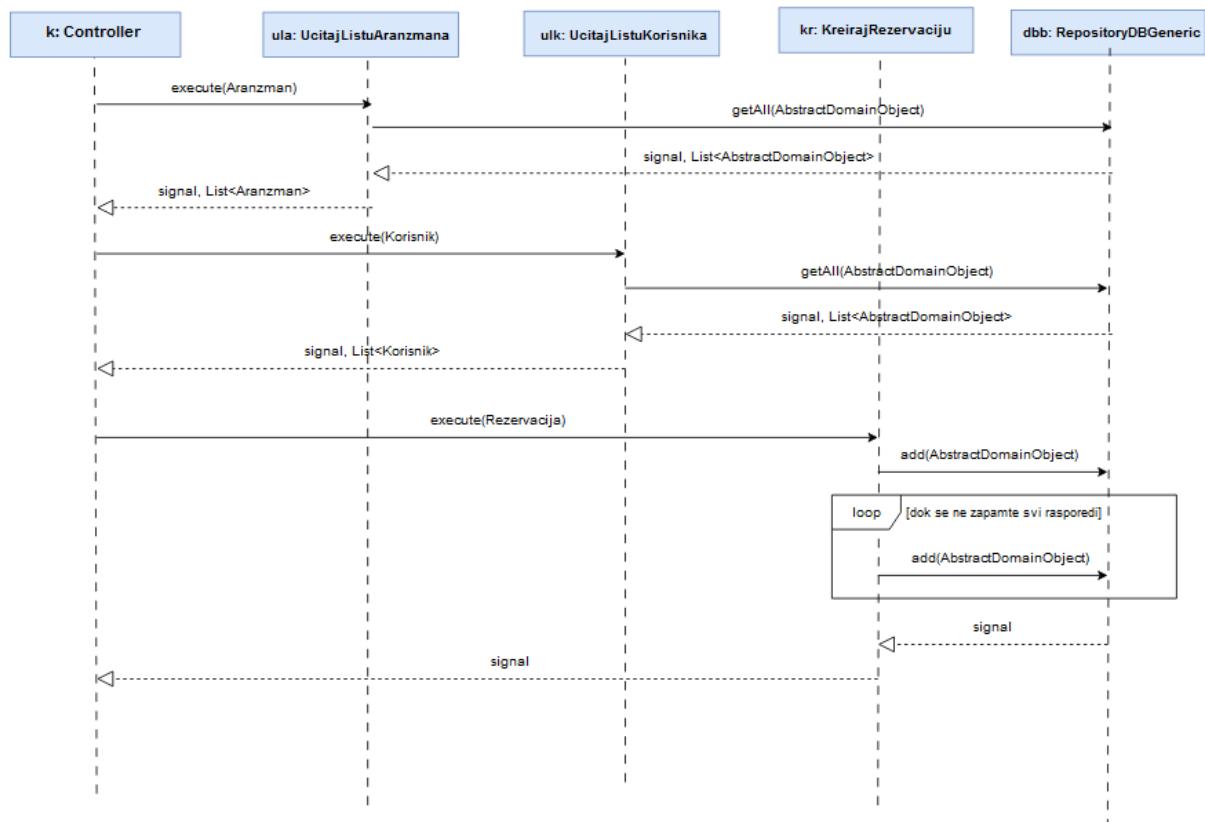
Слика 145 – Дијаграм секвенци за уговор *IzmeniAranzman*

Уговор УГ15: signal KreirajRezervaciju(Rezervacija)

Веза са СК: СК10

Предуслови: Вредносна и структурна ограничења над објектом Rezervacija морају да буду задовољена.

Постуслови: Направљена је нова резервација.



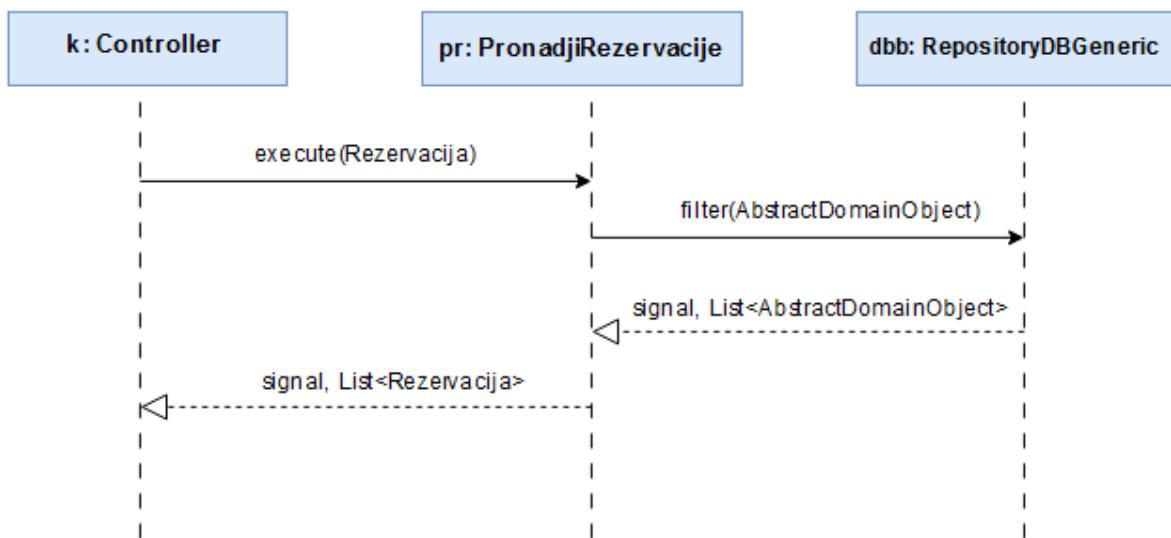
Слика 146 – Дијаграм секвенци за уговор KreirajRezervaciju

Уговор УГ16: signal PronadjiRezervacije(Rezervacija, List<Rezervacija>)

Веза са СК: CK11

Предуслови: /

Постуслови: /



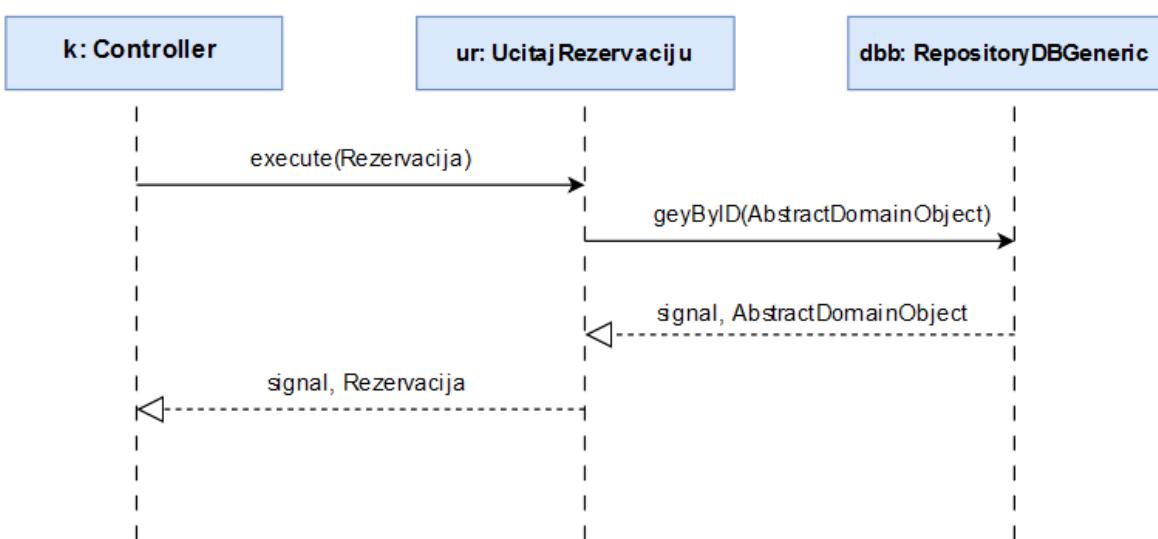
Слика 147 – Дијаграм секвенци за уговор *PronadjiRezervacije*

Уговор УГ17: signal UcitajRezervaciju(Rezervacija)

Веза са СК: CK11

Предуслови: /

Постуслови: /



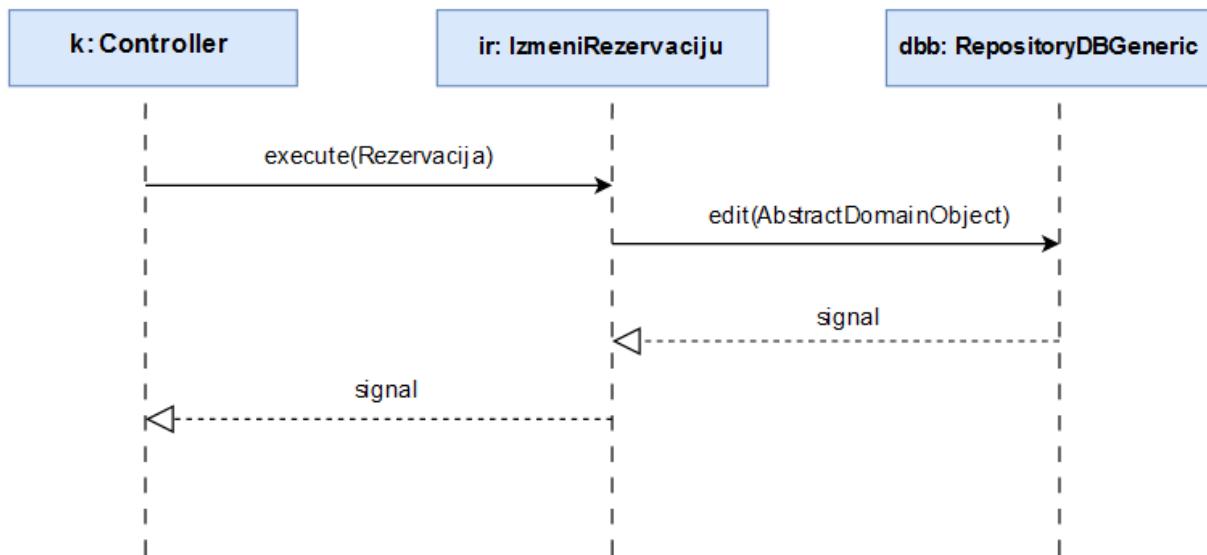
Слика 148 – Дијаграм секвенци за уговор *UcitajRezervaciju*

Уговор УГ18: signal IzmeniRezervaciju(Rezervacija)

Веза са СК: СК11

Предуслови: Вредносна и структурна ограничења над објектом Rezervacija морају да буду задовољена.

Постуслови: Подаци о резервацији су изменjeni.



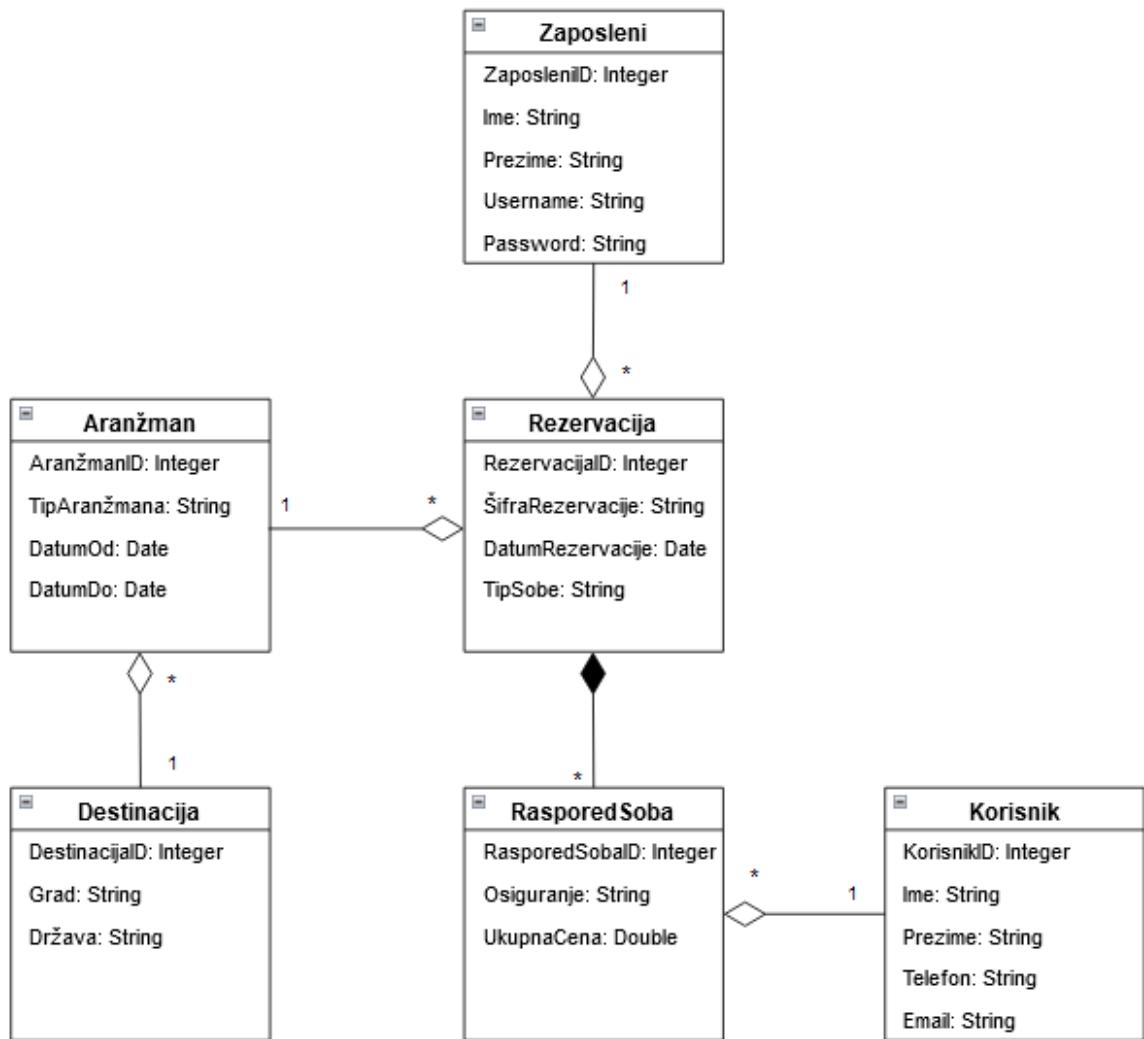
Слика 149 – Дијаграм секвенци за уговор *IzmeniRezervaciju*

Класе које су одговорне за извршење системских операција наслеђују класу *AbstractGenericOperation* како би могле да се повежу са базом. *AbstractGenericOperation* представља апстрактну класу чија главна метода (*execute()*) у себи садржи отварање конекције са базом, валидацију, проверу предуслова, извршење операције, потврду у бази уколико је извршење операције успешно, поништавање уколико извршење операције није било успешно и затварање конекције. Свака од системских операција даје своју имплементацију методе за проверу предуслова, уколико постоји, и методе за извршење конкретне системске операције.

4.3.3.2.2. Пројектовање структуре софтверског система – доменске класе

На основу концептуалних класа праве се софтверске класе структуре.

Концептуалне класе:



Слика 150 – Концептуални дијаграм класа

Софтверске класе структуре:

```
public class Zaposleni implements AbstractDomainObject{
    private int zaposleniID;
    private String ime;
    private String prezime;
    private String username;
    private String password;

    public Zaposleni() {
    }

    public Zaposleni(int zaposleniID, String ime, String prezime, String username, String password) {
        this.zaposleniID = zaposleniID;
        this.ime = ime;
        this.prezime = prezime;
        this.username = username;
        this.password = password;
    }
}
```

Слика 151 – Доменска класа Запослени

```
public class Korisnik implements AbstractDomainObject{
    private int korisnikID;
    private String ime;
    private String prezime;
    private String telefon;
    private String email;

    public Korisnik() {
    }

    public Korisnik(int korisnikID, String ime, String prezime, String telefon, String email) {
        this.korisnikID = korisnikID;
        this.ime = ime;
        this.prezime = prezime;
        this.telefon = telefon;
        this.email = email;
    }
}
```

Слика 152 – Доменска класа Корисник

```
public class Destinacija implements AbstractDomainObject{
    private int destinacijaID;
    private String grad;
    private String drzava;

    public Destinacija() {
    }

    public Destinacija(int destinacijaID, String grad, String drzava) {
        this.destinacijaID = destinacijaID;
        this.grad = grad;
        this.drzava = drzava;
    }
}
```

Слика 153 – Доменска класа Дестинација

```

public class Aranzman implements AbstractDomainObject{
    private int aranzmanID;
    private TipAranzman tipAranzmana;
    private Date datumOd;
    private Date datumDo;
    private Destinacija destinacija;

    private String filter;

    public Aranzman() {
    }

    public Aranzman(int aranzmanID, TipAranzman tipAranzmana, Date datumOd, Date datumDo, Destinacija destinacija) {
        this.aranzmanID = aranzmanID;
        this.tipAranzmana = tipAranzmana;
        this.datumOd = datumOd;
        this.datumDo = datumDo;
        this.destinacija = destinacija;
    }
}

```

Слика 154 – Доменска класа Аранжман

```

public class RasporedSoba implements AbstractDomainObject {
    private int rasporedSobaID;
    private Rezervacija rezervacija;
    private Korisnik korisnik;
    private String osiguranje;
    private double ukupnaCena;

    public RasporedSoba() {
    }

    public RasporedSoba(int rasporedSobaID, Rezervacija rezervacija, Korisnik korisnik, String osiguranje, double ukupnaCena) {
        this.rasporedSobaID = rasporedSobaID;
        this.rezervacija = rezervacija;
        this.korisnik = korisnik;
        this.osiguranje = osiguranje;
        this.ukupnaCena = ukupnaCena;
    }
}

```

Слика 155 – Доменска класа Распоред соба

```

public class Rezervacija implements AbstractDomainObject {
    private int rezervacijaID;
    private Zaposleni zaposleni;
    private String sifraRezervacije;
    private Date datumRezervacije;
    private String tipSobe;
    private Aranzman aranzman;

    private List<RasporedSoba> rasporediSoba;

    public Rezervacija() {
        rasporediSoba = new ArrayList<>();
    }

    public Rezervacija(int rezervacijaID, Zaposleni zaposleni, String sifraRezervacije, Date datumRezervacije, String tipSobe, Aranzman aranzman, List<RasporedSoba> rasporediSoba) {
        this.rezervacijaID = rezervacijaID;
        this.zaposleni = zaposleni;
        this.sifraRezervacije = sifraRezervacije;
        this.datumRezervacije = datumRezervacije;
        this.tipSobe = tipSobe;
        this.aranzman = aranzman;
        this.rasporediSoba = rasporediSoba;
    }
}

```

Слика 156 – Доменска класа Резервација

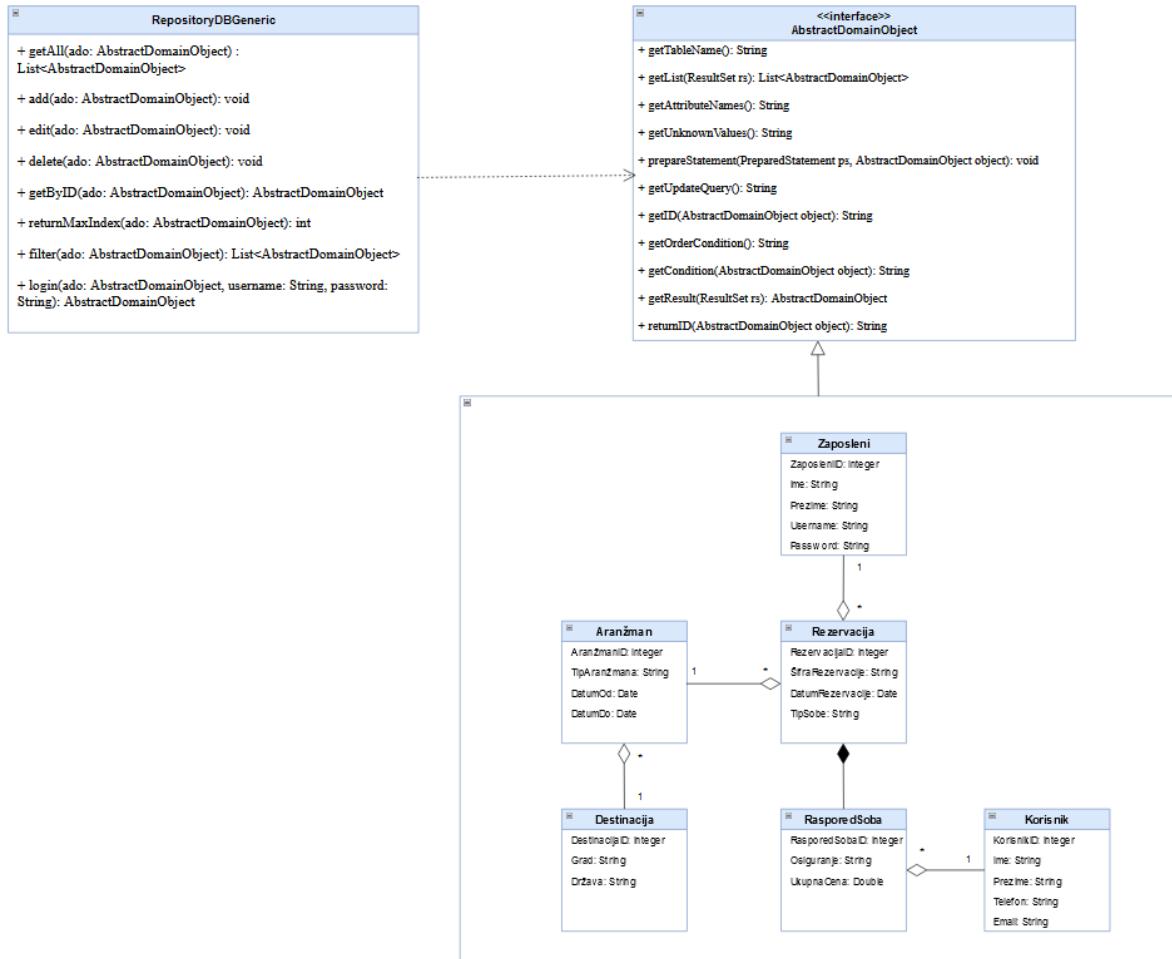
4.3.3.2.3. Пројектовање брокера базе података

Класа **RepositoryDBGeneric** представља перзистентан оквир који посредује у свим операцијама над базом података и реализације следеће методе:

- `List<T> getAll(AbstractDomainObject ado);`
- `void add(AbstractDomainObject ado);`
- `void edit(AbstractDomainObject ado);`
- `void delete(AbstractDomainObject ado);`
- `public AbstractDomainObject getByID(T param);`
- `public int returnMaxIndex(AbstractDomainObject ado);`
- `public List<AbstractDomainObject> filter(AbstractDomainObject ado)`
- `public AbstractDomainObject login(AbstractDomainObject ado, String username, String password);`
- `public List<AbstractDomainObject> getByRezervacijaID(AbstractDomainObject object, int rezervacijaID);`

Све методе класе **RepositoryDBGeneric** су пројектоване као генеричке, што значи да могу да прихватају различите доменске објекте преко параметара. Ово је остварено дефинисањем интерфејса **AbstractDomainObject** кога имплементирају све доменске класе:

- `public String getTableName();`
- `public List<AbstractDomainObject> getList(ResultSet rs);`
- `public String getAttributeNames();`
- `public String getUnknownValues();`
- `public void prepareStatement(PreparedStatement ps, AbstractDomainObject object);`
- `public String getUpdateQuery();`
- `public String getID(AbstractDomainObject object);`
- `public String getOrderCondition();`
- `public String getCondition(AbstractDomainObject object);`
- `public AbstractDomainObject getResult(ResultSet rs);`
- `public String returnID(AbstractDomainObject object) ;`



Слика 157 - Веза између брокера и општег доменског објекта `AbstractDomainObject`

4.3.3.3 Пројектовање складишта података

На основу доменских класа софтвера пројектоване су табеле (складишта података) релационог система за управљање базом података. Систем за управљање базом података који је коришћен у студијском примеру је *MySQL*.

Табела Запослени

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update
zaposleniID	bigint	20		✓	✓		✓		
ime	varchar	50							
prezime	varchar	50							
username	varchar	50							
password	varchar	50							

Слика 158 - Табела Запослени

Табела Корисник

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update
korisnikID	bigint	20		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ime	varchar	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
prezime	varchar	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
telefon	varchar	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
email	varchar	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Слика 159 - Табела Корисник

Табела Дестинација

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update
destinacijaID	bigint	20		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
grad	varchar	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
drzava	varchar	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Слика 160 - Табела Дестинација

Табела Аранжман

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update
aranzmanID	bigint	20		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
tipAranzmanu	varchar	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
datumOd	date			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
datumDo	date			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
destinacija	bigint	20		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Слика 161 - Табела Аранжман

Табела Распоред соба

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update
rasporedSobaID	bigint	20		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
rezervacija	bigint	20		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
korisnik	bigint	20		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
osiguranje	varchar	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ukupnaCena	double			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Слика 162 - Табела Распорд соба

Табела Резервација

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update
rezervacijaID	bigint	20		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
zaposleni	bigint	20		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
sifraRezervacije	varchar	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
datumRezervacije	date			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
tipSobe	varchar	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
aranzman	bigint	20		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Слика 163 - Табела Резервација

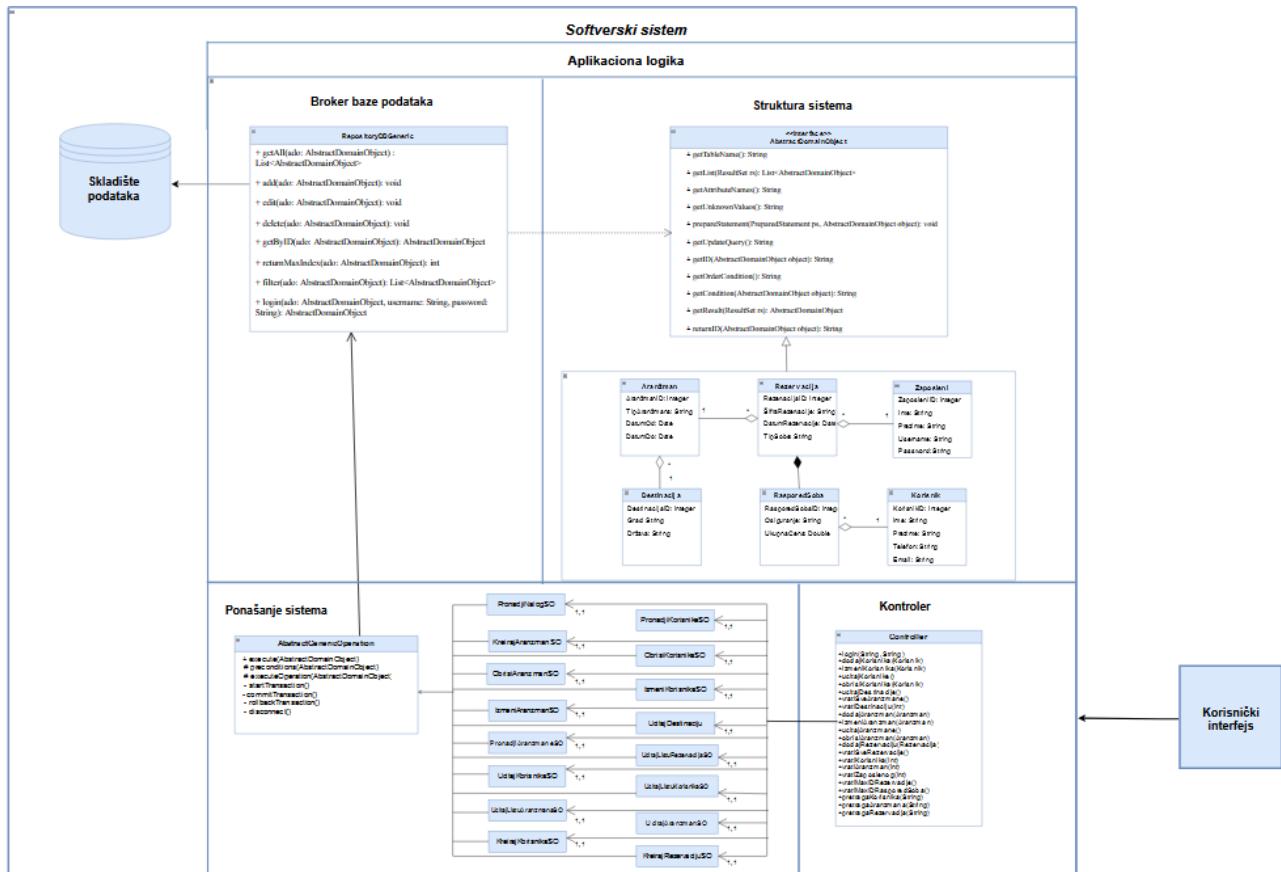
4.4. Имплементација

Софтверски систем је развијан у програмском језику Јава. Систем је пројектован као клијент-сервер апликација. Као систем за управљање базом података коришћен је MySQL, док је развојно окружење *NetBeans IDE 14*. На основу архитектуре софтверског система добијене су следеће софтверске класе:

- Travel_Agency_Client
 - communication/Communication
 - main/MainClient
 - view/controller/LoginFrmController
 - view/controller/MainFrmController
 - view/controller/aranzman/AranzmanFrmController
 - view/controller/aranzman/PretragaAranzmanaFrmController
 - view/controller/korisnik/KorisnikFrmController
 - view/controller/ korisnik/PretragaKorisnikaFrmController
 - view/controller/rezervacija/RezervacijaFrmController
 - view/controller/rezervacija/PretragaRezervacijaFrmController
 - view/coordinator/ClientFrmCoordinator
 - view/form/LoginFrm
 - view/form/MainFrm
 - view/form/aranzman/AranzmanFrm
 - view/form/aranzman/PretragaAranzmanaFrm
 - view/form/korisnik/KorisnikFrm
 - view/form/ korisnik/PretragaKorisnikaFrm
 - view/form/rezervacija/RezervacijaFrm
 - view/form/rezervacija/PretragaRezervacijaFrm
 - view/form/components/TableModel/AranzmanTableModel
 - view/form/components/TableModel/KorisnikTableModel
 - view/form/components/TableModel/RasporedSobaTableModel
 - view/form/components/TableModel/RezervacijaTableModel
 - view/from/util/FormMode
- Travel_Agency_Server
 - controller/Controller
 - main/MainServer
 - operation/AbstractGenericOperation
 - operation/aranzman/IzmeniAranzmanSO
 - operation/aranzman/KreirajAranzmanSO
 - operation/aranzman/ObrisniAranzmanSO
 - operation/aranzman/PronadjiAranzمانSO
 - operation/aranzman/UcitajAranzmanSO
 - operation/aranzman/UcitajListuAranzmanSO

- operation/aranzman/UcitajMaxIDRezervacije
 - operation/destinacija/UcitajDestinacije
 - operation/destinacija/UcitajDestinaciju
 - operation/korisnik/IzmeniKorisnikaSO
 - operation/korisnik/KreirajKorisnikaSO
 - operation/korisnik/ObrisniKorisnikaSO
 - operation/korisnik/PronadjiKorisnikeSO
 - operation/korisnik/UcitajKorisnikaSO
 - operation/korisnik/UcitajListuKorisnikaSO
 - operation/login/PronadjiNalogSO
 - operation/rasporedSoba/PronadjiRasporedSobaPoIDRezervacije
 - operation/rasporedSoba/UcitajMaxIDRasporedSoba
 - operation/rezervacija/IzmeniRezervacijuSO
 - operation/rezervacija/KreirajRezervacijuSO
 - operation/rezervacija/PronadjiRezervacijeSO
 - operation/rezervacija/UcitajListuRezervacijaSO
 - operation/zaposleni/UcitajZaposlenog
 - repository/Repository
 - repository/db/DbConnectionFactory
 - repository/db/DbRepository
 - repository/db/impl/RepositoryDBGeneric
 - server/Server
 - server/Settings
 - server/thread/ProcessClientsRequest
 - view/controller/KonfBazaController
 - view/controller/ServerFrmController
 - view/coordinator/ServerFormCoordinator
 - view/form/KonfBazaFrm
 - view/form/ServerFrm
- Travel_Agency_Common
 - communication/Receiver
 - communication/Request
 - communication/Response
 - communication/Sender
 - domain/AbstractDomainObject
 - domain/Aranzman
 - domain/Destinacija
 - domain/Korisnik
 - domain/RasporedSoba
 - domain/Rezervacija
 - domain/TipAranzmana
 - domain/Zaposleni
 - operations/Operation

На основу претходних целина, може се саставити цела архитектура софтверског система за праћење рада туристичке агенције.



Слика 164 – Комплетна архитектура софтверског система

4.5. Тестирање

Након имплементације, извршено је детаљно тестирање софтверског система. Сваки од имплементираних случајева коришћења је тестиран на два начина: прво ручно, а затим уз помоћ *JUnit4* алата за тестирање.

У првој фази, тестирање је вршено искључиво ручно. Приликом тестирања сваког случаја коришћења, уношени су како правилни, тако и неправилни подаци како би се утврдило понашање система и резултат извршења. Ручно тестирање је обухватило проверу функционалности, корисничког интерфејса и правилности обраде података. Све уочене грешке и проблеми током ручног тестирања су документовани и отклоњени.

У другој фази, тестирање је вршено коришћењем *JUnit4* алата за аутоматизовано тестирање. *JUnit4* тестови су креирани за све случајеве коришћења, са циљем да се обезбеди конзистентност и тачност функционалности софтвера. Ови тестови су омогућили аутоматизовано извршавање и проверу различитих аспеката система, укључујући:

- Предуслове (preconditions):** Тестови су осигурали да свака операција ради са исправним улазним параметрима и да се изузети бацају у случају неправилних података.
- Извршење операција (execution):** Тестови су проверили да се све операције извршавају коректно и да производе очекиване резултате.
- Додавање података у базу:** Проверено је да ли се подаци правилно додају у базу и да ли су сви страни кључеви и остала ограничења исправно поштована.

Приликом тестирања су коришћени и валидни и невалидни подаци како би се утврдило како систем реагује на различите улазне вредности.

Након фазе тестирања, софтвер је спреман за коришћење од стране крајњег корисника.

```

public class PronadjiNalogSO extends AbstractGenericOperation {
    Zaposleni zaposleni;

    @Override
    public void preconditions(Object param) throws Exception {
        if(param == null || !(param instanceof Zaposleni)){
            throw new Exception(message: "Nisu dobri parametri");
        }
    }

    @Override
    public void executeOperation(Object param) throws Exception {
        Zaposleni z = (Zaposleni) param;
        zaposleni = (Zaposleni) repository.login(object: z, username:z.getUsername(), password:z.getPassword());
    }

    public Zaposleni getZaposleni() {
        return zaposleni;
    }

    public void setZaposleni(Zaposleni zaposleni) {
        this.zaposleni = zaposleni;
    }
}

```

Слика 165 – Системска операција PronadjiNalogSO

```

public class PronadjiNalogSOTest {
    private PronadjiNalogSO pronadjiNalogSO;
    private Repository repository;
    private Zaposleni validZaposleni;
    private Zaposleni invalidZaposleni;

    @Before
    public void setUp() {
        pronadjiNalogSO = new PronadjiNalogSO();
        repository = new RepositoryDBGeneric();

        // Kreiramo validnog zaposlenog
        validZaposleni = new Zaposleni(zaposleniID: 1, ime: "User1", prezime: "pass1", username:"Marko", password:"Markovic");

        // Kreiramo nevalidnog zaposlenog
        invalidZaposleni = new Zaposleni(zaposleniID: 3, ime: null, prezime: null, username:null, password:null);

        pronadjiNalogSO.repository = repository;

        try {
            repository.add(param: validZaposleni);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

```

Слика 166a – Тестирање системске операције PronadjiNalogSO

```

@Test
public void testPreconditionsValid() throws Exception {
    pronadjiNalogSO.preconditions(param: validZaposleni);
}

@Test(expected = Exception.class)
public void testPreconditionsInvalidZaposleni() throws Exception {
    pronadjiNalogSO.preconditions(param: invalidZaposleni);
}

@Test
public void testExecuteOperationValidZaposleni() throws Exception {
    pronadjiNalogSO.executeOperation(param: validZaposleni);

    assertEquals(expected:validZaposleni.getUsername(), actual:pronadjiNalogSO.getZaposleni().getUsername());
    assertEquals(expected:validZaposleni.getPassword(), actual:pronadjiNalogSO.getZaposleni().getPassword());
}

@Test(expected = Exception.class)
public void testExecuteOperationInvalidZaposleni() throws Exception {
    pronadjiNalogSO.executeOperation(param: invalidZaposleni);
}

```

Слика 166б – Тестирање системске операције PronadjiNalogSO

5. Закључак

У овом раду представљен је развој софтверског система за праћење рада туристичке агенције, који омогућава ефикасно управљање информацијама и побољшавања укупног пословања у туристичком сектору. Овај систем пружа агенцијама алате за брзо и прецизно управљање подацима, што резултира бољом организацијом и повећаном продуктивношћу.

Детаљно пројектовање софтвера игра кључну улогу у обезбеђивању успешног развоја и имплементације софтверских решења. Добро испланиран и структуриран развој омогућава да се идентификују потенцијални проблеми и изазови у раној фази, што значајно смањује ризике и трошкове. Темељно пројектовање такође омогућава бољу координацију, јасније дефинисање захтева и ефикасније управљање ресурсима.

Методе развоја софтвера, као што је Ларманова упрошћена метода, значајно су допринеле систематичном приступу у развоју софтвера. Ова метода је помогла у структуирању процеса од прикупљања корисничких захтева до имплементације и тестирања, обезбеђујући тако да се сви аспекти пројекта пажљиво размотре и реализују.

Програмски језик Јава је показао своју значајну улогу у развоју оваквих система захваљујући својој стабилности, скалабилности и широкој подршци за различите платформе. Јава нуди погодну основу за развој сложених софтверских система, што је чини идеалним избором за апликације које захтевају високу поузданост и перформансе.

Софтверски систем развијен у овом раду представља функционалну и поуздану апликацију која може бити примењена у различитим туристичким агенцијама. Даљи развој система може укључити додавање нових функционалности, као што су интеграција са мобилним уређајима и развој веб апликације, што би омогућило корисницима лакши приступ и бољу интеракцију са системом.

6. Литература

- [1] Влајић, С. (2020). *Пројектовање софтвера (Скрипта – Радни материјал 2020)*. Факултет организационих наука, Београд.
- [2] Влајић, С., Савић, Д., Антовић, И., Станојевић, В., & Милић, М. (2008). *Пројектовање софтвера – Напредне Јава технологије*. Златни пресек, Београд.
- [3] Kurniawan, B. (2015). *Java: A Beginner's Tutorial, Updated for Java SE 8*. Brainy Software.
- [4] Eck, D. J. (2007). *Introduction to Programming Using Java*. Hobart and William Smith Colleges.
- [5] Вучковић, М., Турајлић, Н., Петровић, М. (2020). *Objektno-orientisano programiranje*. [<http://is.fon.bg.ac.rs/wp-content/uploads/2020/04/Objektno-orientisani-pristup.pdf>, датум приступа: 12.06.2024.]
- [6] Codethat. (2010). *Coffee Time with Java: Part 1 – Sockets*. [<https://codethat.wordpress.com/2010/01/14/coffee-time-with-java-part-1-sockets/>, датум приступа: 10.06.2024.]
- [7] Дракулић, Р. (2017). *Osnovni principi objektno orijentisanog programiranja*. [<https://radmiladrakulic.wordpress.com/2017/09/27/osnovni-principi-objektno-orijentisanog-programiranja/>, датум приступа: 15.06.2024.]
- [8] GeeksforGeeks. (n.d.). *Object-Oriented Programming (OOPs) Concept in Java*. [<https://www.geeksforgeeks.org/object-oriented-programming-oops-concept-in-java/>, датум приступа: 18.06.2024.]
- [9] Juniper Networks. (n.d.). *IPv4 and IPv6 Protocols*. [<https://www.juniper.net/documentation/us/en/software/junos/interfaces-security-devices/topics/topic-map/security-interface-ipv4-ipv6-protocol.html>, датум приступа: 21.06.2024.]

[10] TutorialsPoint. (n.d.). Swing Event Handling.

[https://www.tutorialspoint.com/swing/swing_event_handling.htm, датум приступа: 22.06.2024.]

[11] Javatpoint. (n.d.). Java JDBC. [<https://www.javatpoint.com/java-jdbc>, датум приступа: 23.06.2024.]