**УНИВЕРЗИТЕТ У БЕОГРАДУ**

**ФАКУЛТЕТ ОРГАНИЗАЦИОНИХ НАУКА**

**ЗАВРШНИ РАД**

**Тема : Софтверски систем за регистар наставника у Јава окружењу**

Ментор: Студент

Др Милош Милић Лука Кљајевић 129/14

Београд, 2018. године

Садржај

[1. Увод 3](#_Toc525494666)

[2. Опис технологија 4](#_Toc525494667)

[2.1 Основе објектно-оријентисаног програмирања 4](#_Toc525494668)

[2.2 Јава програмски језик 6](#_Toc525494669)

[2.2.1 Историја 6](#_Toc525494670)

[2.2.2 Извршавање програма 7](#_Toc525494671)

[2.3 Клијент-сервер архитектура 8](#_Toc525494672)

[2.3.1 Комуникација између клијента и сервера путем сокета 8](#_Toc525494673)

[2.4 Управљање подацима коришћењем JDBC 10](#_Toc525494674)

[2.4.1 Повезивање са системом за управљање базом података 11](#_Toc525494675)

[2.5 Извештавање коришћењем JasperReports библиотеке 12](#_Toc525494676)

[2.5.1 Креирање новог извештаја 12](#_Toc525494677)

[2.5.2 Поступак креирања извештаја путем програмског кода 15](#_Toc525494678)

[3. Студијски пример 16](#_Toc525494679)

[3.1 Прикупљање корисничких захтева 18](#_Toc525494680)

[3.1.1 Спецификација захтева путем модела случаја коришћења 18](#_Toc525494681)

[3.1.2 Вербални опис 18](#_Toc525494682)

[3.1.3 Случајеви коришћења 19](#_Toc525494683)

[3.2 Анализа 30](#_Toc525494684)

[3.2.1 Дијаграми секвенци 30](#_Toc525494685)

[3.2.2 Дефинисање уговора о системским операцијама 54](#_Toc525494686)

[3.2.3 Структура софтверског система – Концептуални (доменски) модел 57](#_Toc525494687)

[3.2.4 Структура софтверског система – Релациони модел 58](#_Toc525494688)

[3.3 Пројектовање 65](#_Toc525494689)

[3.3.1 Пројектовање корисничког интерфејса 65](#_Toc525494690)

[3.3.2 Пројектовање апликационе логике 99](#_Toc525494691)

[3.3.3 Пројектовање складишта података 115](#_Toc525494692)

[3.4 Имплементација 119](#_Toc525494693)

[4. Закључак 122](#_Toc525494694)

[5. Литература 123](#_Toc525494695)

# 1. Увод

Сви се слажу да је образовање веома важно. Представља дуготрајно усвајање новостечених знања и вештина у складу са предодређеним доктринама целокупног образовног система и индивидуалних афинитета појединца. Знање је веома широк појам у који спада вишегодишње акумулирање информација и вештина. Школа представља јединицу која као институција треба да негује просперитет појединца кроз различите научне дисциплине природних и друштвених наука. Циљ образовања је да нас од најранијег детињства припреми за изазове живота у свету одраслих.

Међутим, из перспективе садашњег система видимо да школе нису погодне и прилагођене већини ученика. Права сврха система образовања, које треба да нам помогне, треба да буде концентрисана у једне од најзначајнијих аспеката живота: одржавање добрих међуљудских односа и посао. Другим речима, школа, у идеалистичком смислу, представља место у којем учимо да живимо и упознајемо се са вештинама које помажу људима да живе боље.

Савремени образовни систем тренутно није у кораку са новонасталим компјутерским технологијама. Њихов потенцијал је великих размера и, уколико се адекватно и креативно употребе, могу комплементарно да утичу на функционалност и рад образовног система.

Образовне институције су у недостатку са адекватним софтверским алатима који би унапредили функционалност система података којима свакодневно рукују. Вођење евиденције захтева системску уређеност података која олакшава бележење значајних информација за рад образовне установе.

Циљ овог рада је да прикаже пример употребе информационих технологија које одговарају на захтеве корисника и позитивно утиче на целокупну функционалност њиховог радног процеса. Тема се бави развојем софтверског система за вођење евиденције о наставницима, њиховим референцама и предметима.

Након увода, у другом поглављу, дат је опис свих технологија које су коришћене за израду софтверског система.

У трећем поглављу је приказан конкретан пример развоја софтверског система коришћењем упрошћење Ларманове методе.

# 2. Опис технологија

У овом поглављу ће бити описане све технологије које су коришћене приликом израде софтверског система. У првом делу су приказани основни принципи објектно-оријентисаног програмирања, на ком је заснован Јава програмски језик. Након тога, дат је опис самог Јава програмског језика, кроз његову историју и начин на који се извршава на рачунару. Потом је описана клијент-сервер архитектура, на којој је заснован рад апликације. Затим, описана је JDBC технологија за управљање подацима и начин повезивања Јава апликације са системом за управљање базом података. На крају је дат кратак опис JasperReports бибилиотеке за креирање извештаја.

## 2.1 Основе објектно-оријентисаног програмирања

Објектно-оријентисано програмирање је принцип програмирања који користи концепте класа и објеката за пројектовање апликација. Класа се састоји од скупа атрибута и метода. Атрибути описују структуру класе, док методе описују њено понашање. Класа је општи представник неког скупа објеката који имају исте особине (атрибуте и методе). Објекат представља једно појављивање (инстанцу) класе.

Једна од главних предности принципа објектно-оријентисаног програмрања је поновно коришћење кода. Овакав приступ омогућава програмеру да једном написани код, користи на више различитих места у апликацији.

Неки од најпопуларнијих програмских језика који се заснивају на принципу објектно-оријентисаног програмирања, или га подржавају су: Јава, C++, C#, Python, Ruby и PHP.

Основни принципи на којима се заснива објектно-оријентисано програмирање су:

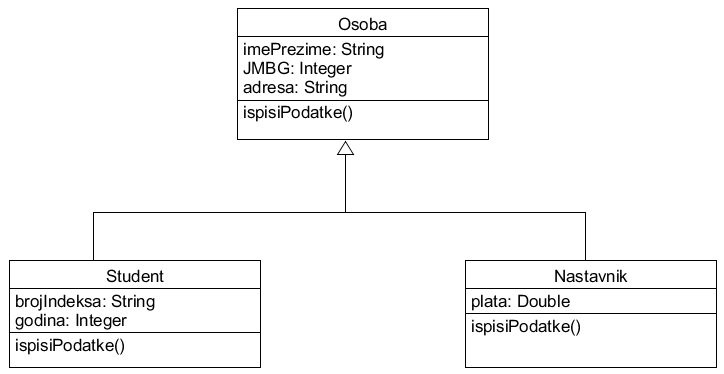
* Учаурења
* Наслеђивање
* Апстракција
* Полиморфизам

**Учаурење** је принцип објектно-оријентисаног програмирања којим се спречава приступ одређеним атрибутима и методама објеката неке класе ради повећања сигурности апликације. У Јава програмском језику, овај концепт је реализован коришћењем модификатора приступа којих има четири:

1. public – доступан приступ свима
2. protected – доступан приступ класама у оквиру истог пакета и подкласама
3. private – доступан приступ само у оквиру исте класе
4. default – доступан приступ само у оквиру истог пакета

**Наслеђивање** је концепт објектно-оријентисаног програмирања којим је омогућено да се заједничке особине и понашања дефинишу у оквиру једне класе – надкласе, на основу које могу бити изведене друге класе – подкласе. Подкласе наслеђују све атрибуте и методе надкласе и имају додатне сопствене атрибуте и/или методе. Главна предност концепта наслеђивања је могућност поновног коришћења једном написаног кода, као и успостављање хијерархије између класа апликације.

На следећој слици је дат графички приказ примера наслеђивања. Класа Osoba представља надкласу, коју наслељују класе Student и Nastavnik – подкласе. Класе Student и Nastavnik, поред својих атрибута и метода, имају и све наслеђене атрибуте и методе класе Osoba.



Слика . Наслеђивање

**Апстракција** представља концепт уклањања свих непотребних карактеристика, осим оних суштинских (есенцијалних). Са аспекта објектно-оријентисаног програмирања, то је принцип сакривања унутрашње имплементације објеката неке класе. Објектима различитих класа је потребно само да знају којим методама могу да приступе, а не и начин на који су оне имплементиране.

**Полиморфизам** је концепт објектно-оријентисаног програмирања који омогућава да се наслеђена метода у подкласи имплементира на другачији начин од исте методе надкласе. Уколико посматрамо класе Student и Osoba на слици 1, метода ispisiPodatke() класе Student може да испише додатне податке као што је број индекса, осим основних података као што су име и презиме и ЈМБГ.

## 2.2 Јава програмски језик

Јава је програмски језик опште намене, заснован на принципу објектно-оријентисаног програмирања, развијен 1995. године. Такође подржава концепт конкурентног програмирања, тј. рад са нитима. Главна предност Јава програмског језика је могућност његовог покретања на различитим оперативним системима. Поред тога, Јава се користи за стварање широког спектра апликација:

* Апликације за рачунаре са графичким корисничким интерфејсом
* Серверске апликације
* Апликације за мобилне телефоне

Управо због тога, Јава је тренутно најпопуларнији програмски језик у употреби, са преко 17% заступљености међу програмерима широм света.[4]

### 2.2.1 Историја

Џејмс Гослинг, Мајк Шеридан и Патрик Наутон, група инжењера компаније Sun Microsystems су први започели развијање Јава прогамског језика 1991. године под називом Храст (енг. Оак). Јава је првобинтно био намењен за програмирање кућних електронских уређаја, попут телевизора, али је био превише напредан за тадашњу телевизијску индустрију. Првобитни назив је касније промењен у Јава, по бренду Јава кафе, тако да лого Јава програмског језика приказује шољу вруће кафе.[1]

Објављен је велики број Јава верзија до данас. Последња објављена верзија Јаве је Java SE 10. Списак свих објављених верзија:

1. JDK Alpha and Beta (1995.)
2. JDK 1.0 (23. Јануар 1996.)
3. JDK 1.1 (19. Фебруар 1997.)
4. J2SE 1.2 (8. Децембар 1998.)
5. J2SE 1.3 (8. Maj 2000.)
6. J2SE 1.4 (6. Фебруар 2002.)
7. J2SE 5.0 (30. Септембар 2004.)
8. Java SE 6 (11. Децембар 2006.)
9. Java SE 7 (28. Јул 2011.)
10. Java SE 8 (18. Март 2014.)
11. Java SE 9 (21. Септембар 2017.)
12. Java SE 10 (20. Mart 2018.)

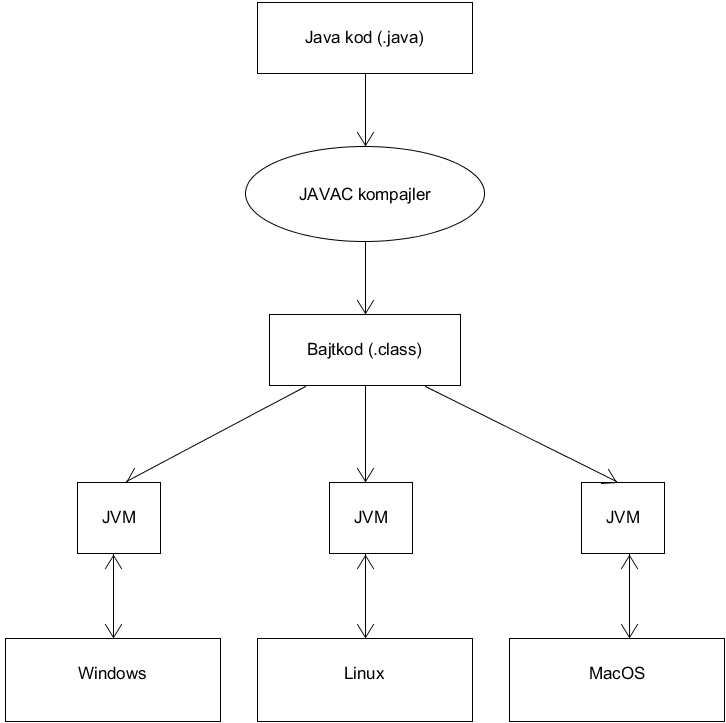
### 2.2.2 Извршавање програма

Један од главних циљева Јава програмског језика је могућност извршавања на различитим оперативним системима. Како би ово било могуће, пре извршавања програма, Јава програмски код се прво компајлира у међу-облик који се назива Јава бајткод (eng. bytecode).

Наредбе у Јава бајткоду се директно преводе у наредбе машинског кода. Како би програм написан у Јава програмском језику могао да се покрене, потребно је да постоји инсталирана Јава Виртуелна Машина (JVM) на рачунару која ће да преведе наредбе из бајткода у наредбе машинског кода.

Коришћење Јава бајткода је значајно поједноставило извршавање на различитим оперативним системима. Међутим, превођење бајткода у машински код је довело до тога да апликације раде спорије од оних које се директно преводе у машински код. Управо због овога програми написани у Јава програмском језику имају тенденцију да раде спорије од оних развијених у програмском језику C++.[1]

На следећој слици је дат приказ процеса извршавања Јава програма.

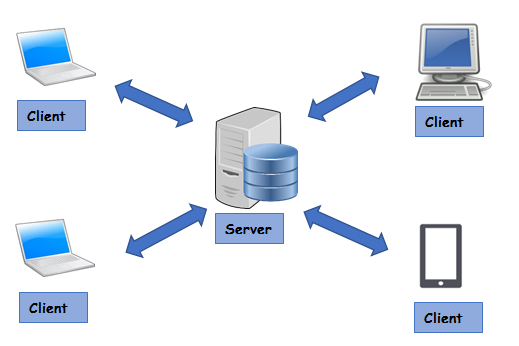


Слика . Извршавање Јава програма

## 2.3 Клијент-сервер архитектура

Клијент-сервер архитектура је структура дистрибуираних апликација која дели задатке између оних који обезбеђују ресурсе и/или услуге – сервера и оних који траже услуге – клијената. Сервер покреће серверски програм који дели своје ресурсе са клијентима. Клијент не дели своје ресурсе, већ захтева серверов садржај или услугу. Клијенти започињу комуникацију са серверима који чекају долазеће захтеве.

Сервер обезбеђује функцију или услугу једном или више клијената који захтевају ту услугу. Сервери се класификују по врсти услуге коју пружају. На пример, веб-сервер омогућава преглед веб-страница, док фајл-сервер обезбеђује компјутерске датотеке.



Слика . Клијент-сервер архитектура[[1]](#footnote-1)

### 2.3.1 Комуникација између клијента и сервера путем сокета

Сокет је механизам који омогућава комуникацију између програма који се извршавају у мрежи. Приликом повезивања два програма преко сокета, по један сокет се генерише за сваки програм. Сваки од сокета садржи референцу на други сокет.

Адреса сокета се састоји од два дела:

1. адресе рачунара на коме се налази програм
2. броја порта који је генерисан помоћу сокета

Прво се покреће програм на серверском рачунару. Пример адресе серверског рачунара је 10.10.10.10. Прави се серверски сокет који се повезује са произвољним портом, на пример 9090, наредбом:

ServerSocket ss = new ServerSocket(9090);

Након тога извршава се наредба којом серверски сокет прелази у стање чекања на захтеве клијената.

Socket serverSoket = ss.accept();

У клијентском програму се извршава наредба којом се врши повезивање клијента са сервером.

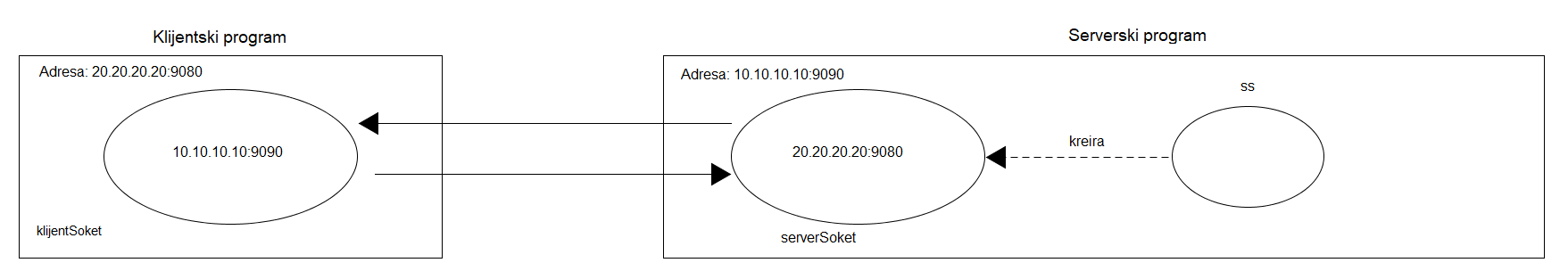
Socket klijentSoket = new Socket(“10.10.10.10”, 9090);

Сокет који је направљен на клијентској страни прослеђује до серверског сокета своју адресу.

Успостављање везе између серверског и клијентског сокета се одвија у следећим корацима:

* Клијентски сокет klijentSoket добија референцу на серверски сокет.
* Серверски сокет ss генерише нови сокет serverSoket. Адреса новог сокета ће бити иста као адреса серверског сокета.
* Нови сокет serverSoket добија референцу на клијентски.

На следећој слици дат је графички приказ претходно описаног поступка успостављања везе између серверског и клијентског сокета.



Слика . Комуникација између сокета [3]

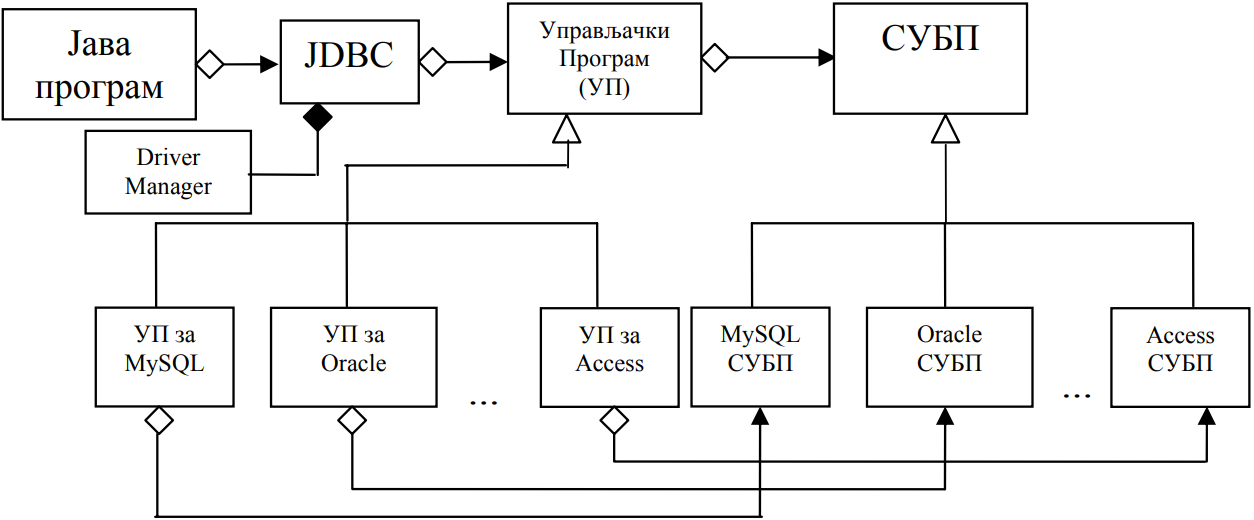
## 2.4 Управљање подацима коришћењем JDBC

Java Database Connectivity (JDBC) представља апликациони програмски интерфејс (API) који обезбеђује апликацијама написаним у Јава програмском језику могућност повезивања са различитим системима за управљање базом података (СУБП), као што су: MySQL, Oracle, SQL Server, MS Access.

Основни предуслов за повезивање Јава апликације преко JDBC-a, са неким СУБП јесте постојање одговарајућег управљачког програма (драјвера). За сваки СУБП се посебно прави управљачки програм.

Главна предност при коришћењу JDBC API-ja за повезивање са неким СУБП, је што се једном написан програм у Јава програмском језику може без промене користити за приступ и манипулацију подацима над различитим СУБП.

На следећој слици је дат графички приказ везе између Јава програма и различитих СУБП.



Слика . Веза Јава програма са различитим СУБП[3]

### 2.4.1 Повезивање са системом за управљање базом података

Повезивање Јава програма са изабраним СУБП одвија се у следећим корацима:

1. Укључивање JDBC API-a у Јава програм наредбом:

import java.sql.\*;

1. Учитавање управљачког програма у Јава програм наредбом:

Class.forName(“com.mysql.jdbc.Driver”);

1. Успостављање везе између Јава програма и изабраног СУБП, које се врши коришћењем JDBC DriverManager класе, наредбом:

Connection konekcija = DriverManager.getConnection(dbUrl, user, pass);

Као резултат успостављања везе са базом података преко JDBC DriverManager класе добија се објекат класе Connection који чува конекцију ка бази података.[3]

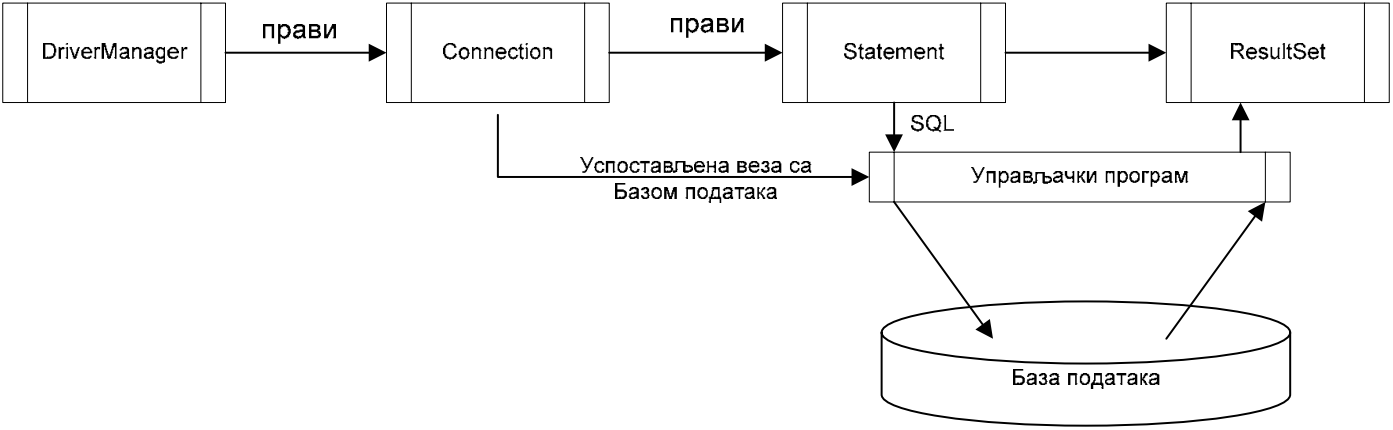
Када је успешно успостављена веза између Јава програма и изабраног СУБП, могуће је извршавање SQL наредби над базом података. За креирање наредбе, користи се JDBC Statement класа на следећи начин:

Statement naredba = konekcija.createStatement();

Помоћу објекта naredba, могуће је извршити произвољни SQL упит на следећи начин:

naredba.executeQuery(upit);

Као резултат претходне наредбе добија се објекат класе ResultSet, који садржи резултат извршеног упита. На следећој слици дат је графички приказ поступка извршења операција над базом података:



Слика . Поступак извршења операција над базом података[3]

## 2.5 Извештавање коришћењем JasperReports библиотеке

JasperReports је библиотека написана у Јава програмском језику, која пружа могућност лаког креирања динамичких извештаја, на основу неког извора података. Најчешће коришћени извор је база података, са којом се повезује коришћењем JDBC-a.

JasperReports омогућава кориснику рад са параметрима, изразима и променљивама. Такође, има и комплексније могућности као што су посебни извори података, скрипте и рад са подизвештајима који отварају могућност стварања веома сложених извештаја.

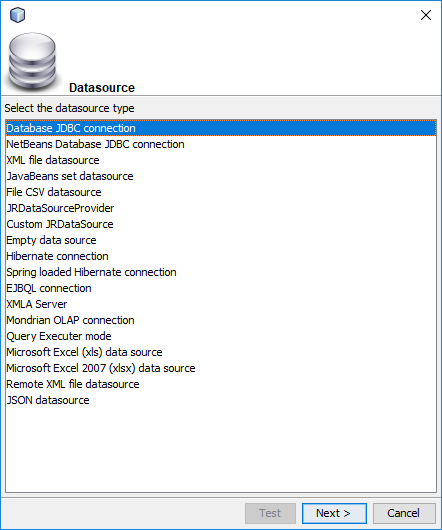
Једна од главних предности овог алата је могућност чувања извештаја у различитим форматима, као што су: PDF, HTML, Microsoft Excel, XML itd.

Сви JasperReports извештаји су засновни на XML формату и имају екстензију .jrxml. Коришћењем неког од алата за израду или апликације у Јава програмском језику, .jrxml фајл се компајлира у .jasper фајл који се попуњава подацима из одабраног извора.

Постоје разни алати за креирање извештаја, а најпознатији су: Jaspersoft Studio, iReport Designer i JasperAssistant.

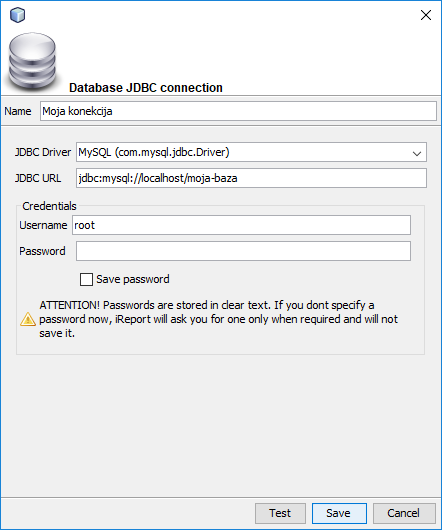
### 2.5.1 Креирање новог извештаја

Креирање новог извештаја се врши коришћењем iReport Designer алата. Након уноса података о називу извештаја и локацији на рачунару, потребно је додати нови извор података. Кликом на дугме New отвара се прозор за додавање новог извора података.



Слика . Подешавање новог извора података

Уколико се изабере Database JDBC connection, отвара се прозор за подешавање конекције са постојећом базом података, приказан на следећој слици. Након попуњавања података, креира се нови извештај.

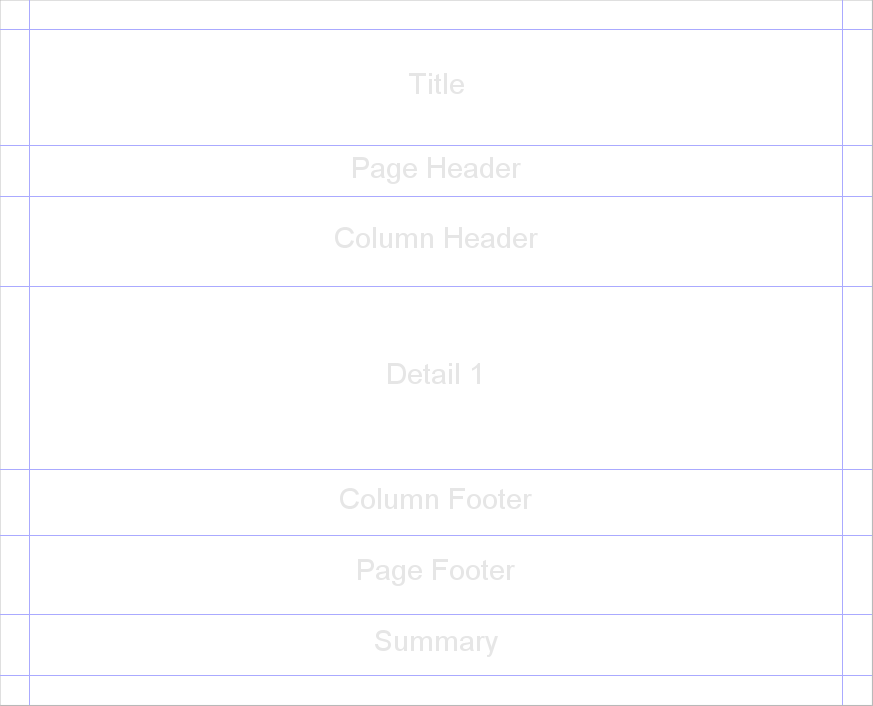


Слика . Креирање нове конекције са базом података

Сваки JasperReports извештај се састоји од следећих седам секција:

1. Наслов – Title: Садржи наслов извештаја и појављује се само једном, на почетку извештаја.
2. Заглавље странице – Page Header: Може да садржи податке као што су аутор извештаја и датум када је креиран. Појављује се на врху сваке странице.
3. Заглавље колона – Column Header: Садржи поља са називима колона које су приказане у извештају.
4. Секција за детаље – Detail: Садржи поља са конкретним подацима.
5. Доње заглавље колона – Column Footer: Може да садржи поља са називима колона на дну странице.
6. Доње заглавље – Footer: Може да садржи податке као што су редни број странице, укупан број страница и датум када је креиран извештај.
7. Збирни подаци – Summary: Садржи збирне податке који се односе за сваку од колона. Пример: Збир по колонама.

На следећој слици је дат графички приказ структуре JasperReports извештаја.



Слика . Структура JasperReports извештаја

Свака од секција се може мењати по величини. Такође, могуће је додавање нових и брисање одређених секција. Секције се попуњавају разним елементима из палете коју пружа iReports алат, као што су: статички текст (Static Text), текстуално поље за приказ вредности (Text Field), график (Chart), подизвештај (Subreport), редни број странице (Page number) итд.

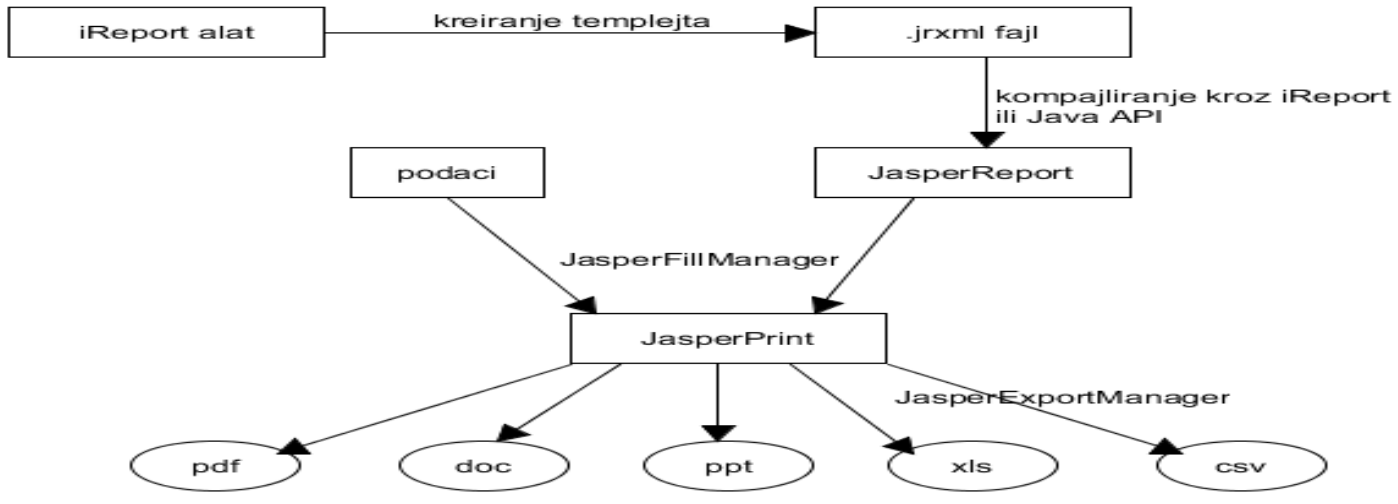
### 2.5.2 Поступак креирања извештаја путем програмског кода

Након дизајнирања извештаја, креирање путем програмског кода се врши следећим наредбама:

JasperReport jr = JasperCompileManager.compileReport("report.jrxml");  
JasperPrint jp = JasperFillManager.fillReport(jr, params, connection);  
JasperViewer.viewReport(jp);

Прва наредба врши компајлирање.jrxml фајла у .jasper фајл. Друга наредба попуњава компајлирани извештај одговарајућим параметрима и подацима из базе. Трећа и последња наредба приказује кориснику попуњени извештај.

На следећој слици дат је графички приказ целокупног процеса креирања JasperReports извештаја.



Слика . Процес креирања извештаја[[2]](#footnote-2)

# 3. Студијски пример

У овом студијском примеру, коришћена је упрошћена Ларманова метода за развој софтвера. Као резултат добијамо софтвер који је модуларан, скалабилан, употребљив, поуздан и лак за одржавање. Ларман препоручује инкрементално/итеративни “животни циклус” софтвера. Свака од итерација обрађује део дефинисаних случајева коришћења кроз све фазе животног циклуса и свака итерација додаје инкремент функционалности система.[5]

Развијање софтвера упрошћеном Лармановом методом се састоји од следећих пет фаза:

1. Прикупљање захтева корисника
2. Анализа
3. Пројектовање
4. Имплементација
5. Тестирање

**Фаза прикупљања захтева** обухвата дефинисање својстава и услова које софтверски систем треба да задовољи.

**Фаза анализе** обухвата опис структуре и понашања софтверског система, који заједно чине пословну логику. Структура система се описује преко концептуалног модела и релационог модела, док се понашање система описује путем секвенцних дијаграма и системских операција.

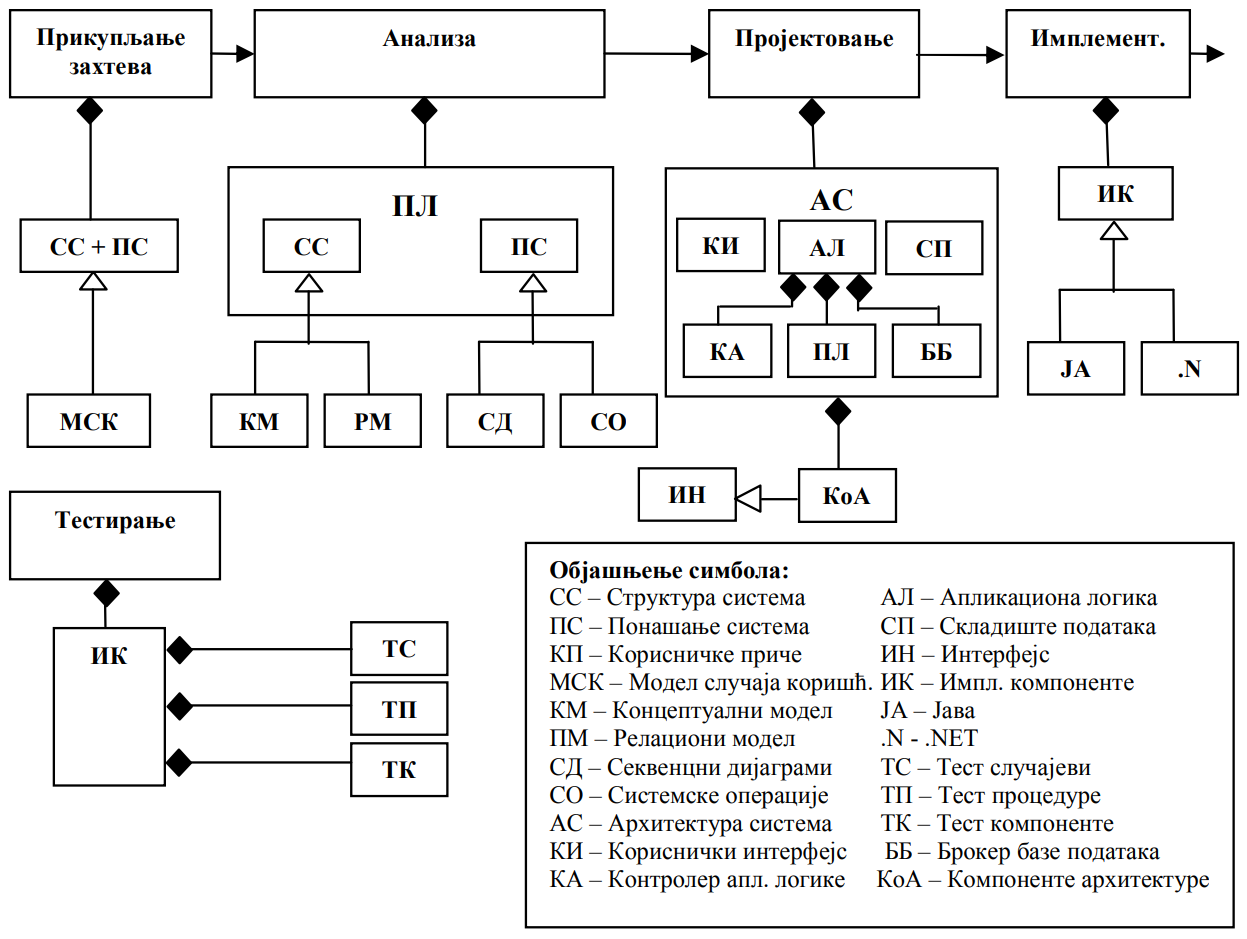
**Фаза пројектовања** обухвата процес пројектовања архитектуре софтверског система. Архитектура је тронивојска и састоји се од следећих елемената:

1. Кориснички интерфејс
2. Апликациона логика, коју чине:
   1. Контролер апликационе логике
   2. Пословна логика
   3. Брокер базе података
3. Складиште података

**Фаза имплементације** обухвата реализацију софтверског система у некој од изабраних технологија.

**Фаза тестирања** обухвата тестирање сваке компоненте, тако што се праве тест случајеви, тест процедуре и тест компоненте.

На следећој слици дат је графички приказ развоја софтверског система коришћењем упрошћене Ларманове методе.



Слика . Развој софтверског система коришћењем упрошћене Ларманове методе[6]

## 3.1 Прикупљање корисничких захтева

Кориснички захтеви представљају својства и услове које софтверски систем треба да задовољи. Могу се поделити на функционалне и нефункционалне. Функционални захтеви дефинишу захтеване функције система, док нефункционални захтеви дефинишу све остале захтеве.[6]

Опис корисничких захтева, код упрошћене Ларманове методе, врши се коришћењем UML Модела Случаја Коришћења (Use-Case Model).

### 3.1.1 Спецификација захтева путем модела случаја коришћења

Модел случаја коришћења се састоји из једног или више случаја коришћења, актора и веза између актора и случаја коришћења. Сваки случај коришћења се састоји од скупа сценарија.

Случај коришћења има један главни и више алтернативних сценарија. Сценарио описује једно жељено коришћење система од стране корисника. Сваки сценарио се састоји од низа акција које предузимају корисник и систем.

Постоје три врсте акција које може да изведе корисник:

1. Актор Припрема Улаз за Системску Операцију (АПУСО)
2. Актор Позива систем да изврши Системску Операцију (АПСО)
3. Актор извршава НеСистемску Операцију (АНСО)

Систем може да изврши две врсте акција:

1. Систем извршава Системску Операцију (СО)
2. Резултат системске операције (Излазни аргументи (IA)) се прослеђује до актора.[6]

### 3.1.2 Вербални опис

Потребно је напрвити апликацију која ће представљати регистар свих наставника на факултету. Апликација треба да омогући евидентирање наставника, вршење претраге свих наставника, измену података о наставнику и брисање неког од наставника.

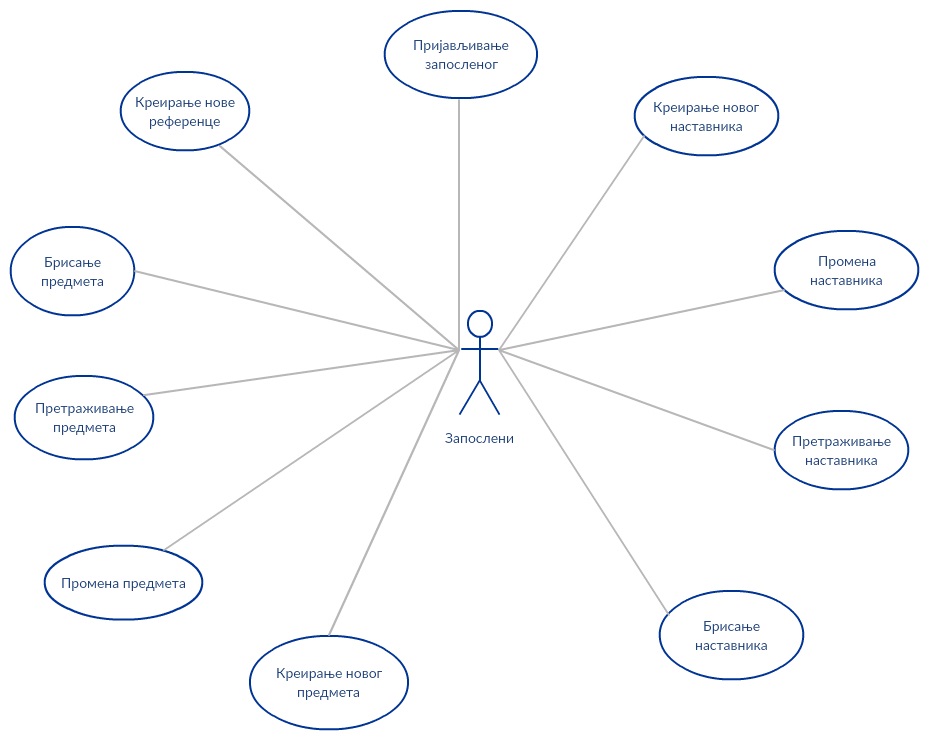
Сваки наставник може да предаје више различитих предмета. За сваки предмет који наставник предаје, потребно је водити евиденцију о студијском програму ком припада, степену студија, као и недељном фонду часова. Такође је потребно омогућити унос новог предмета, претрагу предмета, измену и брисање предмета.

Како би запослени могао да приступи апликацији, потребно је да буде улогован на систем.

### 3.1.3 Случајеви коришћења

Идентификовани су следећи случајеви коришћења приказани на следећој слици:

1. Пријављивање запосленог
2. Креирање новог наставника (сложен)
3. Промена наставника
4. Претраживање наставника
5. Брисање наставника
6. Креирање новог предмета
7. Промена предмета
8. Претраживање предмета
9. Брисање предмета
10. Креирање нове референце



Слика . Дијаграм случајева коришћења

#### 3.1.3.1 СК1: Случај коришћења - Пријављивање запосленог

**Назив СК**

Пријављивање запосленог

**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

**Предуслов**: Систем је укључен и приказује форму за пријављивање запосленог.

**Основни сценарио СК**

1. Запослени **уноси** податке за аутентификацију запосленог. (АПУСО)
2. Запослени **позива** систем да нађе запосленог по задатој вредности. (АПСО)
3. Систем **тражи** запосленог по задатој вредности. (СО)
4. Систем **приказује** запосленом поруку: “Успешно сте пријављени”. (ИА)

**Алтернативна сценарија**

4.1 Уколико систем не може да нађе запосленог он приказује запосленом поруку: “Систем не може да нађе запосленог по задатој вредности”. (ИА)

#### 3.1.3.2 СК2: Случај коришћења – Креирање новог наставника

**Назив СК**

Креирање новог наставника

**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

**Предуслов**: Систем је укључен, запослени je улогован под својом шифром и учитане су листе типова наставника, типова академских успеха, предмета и референци. Систем приказује форму за рад са наставником.

**Основни сценарио СК**

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

**Алтернативна сценарија**

5.1 Уколико систем не може да креира новог наставника он приказује запосленом поруку: “Систем не може да креира новог наставника”. (ИА)

#### 3.1.3.3 СК3: Случај коришћења – Промена наставника

**Назив СК**

Промена наставника

**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

**Предуслов**: Систем је укључен, запослени je улогован под својом шифром и учитане су листе типова наставника, типова академских успеха, предмета и референци. Систем приказује форму за рад са наставником.

**Основни сценарио СК**

1. Запослени **уноси** вредностпо којој претражује наставника. (АПУСО)
2. Запослени **позива** систем да нађе наставника по задатој вредности. (АПСО)
3. Систем **тражи** наставника по задатој вредности. (СО)
4. Систем **приказује** запосленом наставнике и поруку: “Систем је нашао наставнике по задатој вредности”. (ИА)
5. Запослени **бира** жељеног наставника из листе. (АПУСО)
6. Запослени **позива** систем да прикаже све податке о изабраном наставнику. (АПСО)
7. Систем **тражи** податке о изабраном наставнику. (СО)
8. Систем **приказује** податке о наставнику. (ИА)
9. Запослени **уноси** **(мења)** податке о наставнику. (АПУСО)
10. Запослени **контролише** да ли је коректно унео податке о наставнику. (АНСО)
11. Запослени **позива** систем да запамти податке о наставнику. (АПСО)
12. Систем **памти** податке о наставнику. (СО)
13. Систем **приказује** запосленом поруку: “Систем је запамтио наставника.” (ИА)

**Алтернативна сценарија**

4.1 Уколико систем не може да нађе наставника он приказује запосленом поруку: “Систем не може да нађе наставника по задатој вредности”. Прекида се извршење сценариа. (ИА)

8.1 Уколико систем не може да нађе податке о наставнику он приказује запосленом поруку: “Систем не може да нађе податке о наставнику”. Прекида се извршење сценариа. (ИА)

13.1 Уколико систем не може да запамти податке о наставнику он приказује запосленом поруку: “Систем не може да запамти наставника”. (ИА)

#### 3.1.3.4 СК4: Случај коришћења – Претраживање наставника

**Назив СК**

Претраживање наставника

**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

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

**Основни сценарио СК**

1. Запослени **уноси** вредностпо којој претражује наставника. (АПУСО)
2. Запослени **позива** систем да нађе наставника по задатој вредности. (АПСО)
3. Систем **тражи** наставника по задатој вредности. (СО)
4. Систем **приказује** запосленом податке о наставницима. (ИА)

**Алтернативна сценарија**

4.1 Уколико систем не може да нађе наставника он приказује запосленом поруку: “Систем не може да нађе наставника по задатој вредности”. (ИА)

#### 3.1.3.5 СК5: Случај коришћења – Брисање наставника

**Назив СК**

Брисање наставника

**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

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

**Основни сценарио СК**

1. Запослени **уноси** вредностпо којој претражује наставника. (АПУСО)
2. Запослени **позива** систем да нађе наставника по задатој вредности. (АПСО)
3. Систем **тражи** наставника по задатој вредности. (СО)
4. Систем **приказује** запосленом наставнике и поруку: “Систем је нашао наставнике по задатој вредности”. (ИА)
5. Запослени **бира** жељеног наставника из листе. (АПУСО)
6. Запослени **позива** систем да прикаже све податке о изабраном наставнику. (АПСО)
7. Систем **тражи** податке о изабраном наставнику. (СО)
8. Систем **приказује** податке о наставнику. (ИА)
9. Запослени **позива** систем да обрише наставника. (АПСО)
10. Систем **брише** наставника. (СО)
11. Систем **приказује** запосленом поруку: “Систем је обрисао наставника.” (ИА)

**Алтернативна сценарија**

4.1 Уколико систем не може да нађе наставника он приказује запосленом поруку: “Систем не може да нађе наставника по задатој вредности”. Прекида се извршење сценариа. (ИА)

8.1 Уколико систем не може да нађе податке о наставнику он приказује запосленом поруку: “Систем не може да нађе податке о наставнику”. Прекида се извршење сценариа. (ИА)

11.1 Уколико систем не може да обрише наставника он приказује запосленом поруку: “Систем не може да обрише наставника”. (ИА)

#### 3.1.3.6 СК6: Случај коришћења – Креирање новог предмета

**Назив СК**

Креирање новог предмета

**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

**Предуслов**: Систем је укључен, запослени je улогован под својом шифром и учитане су листе студијских програма и наставника. Систем приказује форму за рад са предметом.

**Основни сценарио СК**

1. Запослени **уноси** податке у нови предмет. (АПУСО)
2. Запослени **контролише** да ли је коректно унео податке у нови предмет. (АНСО)
3. Запослени **позива** систем да креира нови предмет. (АПСО)
4. Систем **креира** нови предмет. (СО)
5. Систем **приказује** запосленом нови предмет и поруку: “Систем је креирао нови предмет“. (ИА)

**Алтернативна сценарија**

5.1 Уколико систем не може да креира нови предмет он приказује запосленом поруку: “Систем не може да креира нови предмет ”. Прекида се извршење сценариа. (ИА)

#### 3.1.3.7 СК7: Случај коришћења – Промена предмета

**Назив СК**

Промена предмета

**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

**Предуслов**: Систем је укључен, запослени je улогован под својом шифром и учитане су листе студијских програма и наставника. Систем приказује форму за рад са предметом.

**Основни сценарио СК**

1. Запослени **уноси** вредностпо којој претражује предмет. (АПУСО)
2. Запослени **позива** систем да нађе предмет по задатој вредности. (АПСО)
3. Систем **тражи** предмет по задатој вредности. (СО)
4. Систем **приказује** запосленом предмете и поруку: “Систем је нашао предмете по задатој вредности”. (ИА)
5. Запослени **бира** жељени предмет из листе. (АПУСО)
6. Запослени **позива** систем да прикаже све податке о изабраном предмету. (АПСО)
7. Систем **тражи** податке о изабраном предмету. (СО)
8. Систем **приказује** податке о предмету. (ИА)
9. Запослени **уноси** **(мења)** податке о предмету. (АПУСО)
10. Запослени **контролише** да ли је коректно унео податке о предмету. (АНСО)
11. Запослени **позива** систем да запамти податке о предмету. (АПСО)
12. Систем **памти** податке о предмету. (СО)
13. Систем **приказује** запосленом поруку: “Систем је запамтио предмет.” (ИА)

**Алтернативна сценарија**

4.1 Уколико систем не може да нађе предмет он приказује запосленом поруку: “Систем не може да нађе предмет по задатој вредности”. Прекида се извршење сценариа. (ИА)

8.1 Уколико систем не може да нађе податке о предмету он приказује запосленом поруку: “Систем не може да нађе податке о предмету”. Прекида се извршење сценариа. (ИА)

13.1 Уколико систем не може да запамти податке о предмету он приказује запосленом поруку: “Систем не може да запамти предмет ”. (ИА)

#### 3.1.3.8 СК8: Случај коришћења – Претраживање предмета

**Назив СК**

Претраживање предмета

**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

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

**Основни сценарио СК**

1. Запослени **уноси** вредностпо којој претражује предмет. (АПУСО)
2. Запослени **позива** систем да нађе предмет по задатој вредности. (АПСО)
3. Систем **тражи** предмет по задатој вредности. (СО)
4. Систем **приказује** запосленом податке о предметима и поруку: “Систем је нашао предмете по задатој вредности”. (ИА)

**Алтернативна сценарија**

4.1 Уколико систем не може да нађе предмет он приказује запосленом поруку: “Систем не може да нађе предмет по задатој вредности”. (ИА)

#### 3.1.3.9 СК9: Случај коришћења – Брисање предмета

**Назив СК**

Брисање предмета

**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

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

**Основни сценарио СК**

1. Запослени **уноси** вредностпо којој претражује предмет. (АПУСО)
2. Запослени **позива** систем да нађе предмет по задатој вредности. (АПСО)
3. Систем **тражи** предмет по задатој вредности. (СО)
4. Систем **приказује** запосленом предмете и поруку: “Систем је нашао предмете по задатој вредности”. (ИА)
5. Запослени **бира** жељени предмет из листе. (АПУСО)
6. Запослени **позива** систем да прикаже све податке о изабраном предмету. (АПСО)
7. Систем **тражи** податке о изабраном предмету. (СО)
8. Систем **приказује** податке о предмету. (ИА)
9. Запослени **позива** систем да обрише предмет. (АПСО)
10. Систем **брише** предмет. (СО)
11. Систем **приказује** запосленом поруку: “Систем је обрисао предмет.” (ИА)

**Алтернативна сценарија**

4.1 Уколико систем не може да нађе предмет он приказује запосленом поруку: “Систем не може да нађе предмет по задатој вредности”. Прекида се извршење сценариа. (ИА)

8.1 Уколико систем не може да нађе податке о предмету он приказује запосленом поруку: “Систем не може да нађе податке о предмету”. Прекида се извршење сценариа. (ИА)

11.1 Уколико систем не може да обрише предмет он приказује запосленом поруку: “Систем не може да обрише предмет ”. (ИА)

#### 3.1.3.10 СК10: Случај коришћења – Креирање нове референце

**Назив СК**

Креирање нове референце

**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

**Предуслов**: Систем је укључен, запослени je улогован под својом шифром и учитане су листе наставника и спољних аутора. Систем приказује форму за рад са референцом.

**Основни сценарио СК**

1. Запослени **уноси** податке о новој референци. (АПУСО)
2. Запослени **контролише** да ли је коректно унео податке о новој референци. (АНСО)
3. Запослени **позива** систем да запамти податке о референци. (АПСО)
4. Запослени **позива** систем да креира нову референцу. (АПСО)
5. Систем **креира** нову референцу. (СО)
6. Систем **приказује** запосленом поруку: “Систем је креирао нову референцу“. (ИА)

**Алтернативна сценарија**

6.1 Уколико систем не може да креира нову референцу он приказује запосленом поруку: “Систем не може да креира нову референцу”. (ИА)

## 3.2 Анализа

Након прве фазе развоја софтвера, фазе прикупљања корисничких захтева, прелази се на следећу фазу – Фазу анализе. Главни циљ ове фазе је опис логичке структуре и понашања софтверског система.

Структура система се описује помоћу концептуалног модела и релационог модела. Понашање система се описује преко дијаграма секвенци и системских операција.

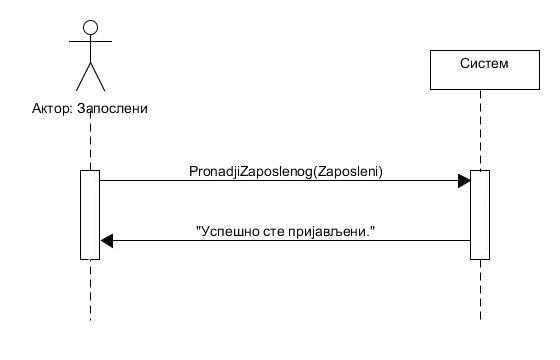
### 3.2.1 Дијаграми секвенци

Понашање софтверског система се описује коришћењем UML-ових дијаграм секвенци (дијаграма сарадње). Дијаграм секвенци приказује догађаје у одређеном редоследу, који представљају интеракцију између корисника и система.

За сваки случај коришћења, праве се системски дијаграми секвенци, који садрже само АПСО и ИА акције сценарија.

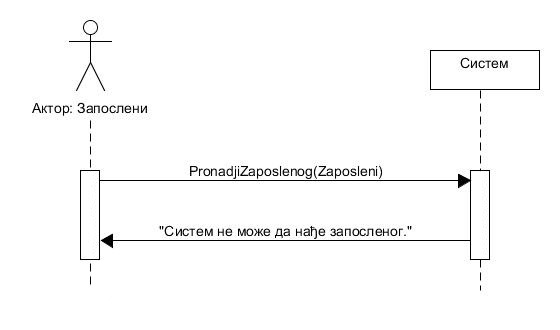
#### 3.2.1.1 ДС1: Дијаграм секвенци случаја коришћења – Пријављивање запосленог

1. Запослени **позива** систем да нађе запосленог по задатој вредности. (АПСО)
2. Систем **приказује** запосленом поруку: “Успешно сте пријављени”. (ИА)



**Алтернативна** **сценарија**

2.1 Уколико систем не може да нађе запосленог он приказује запосленом поруку: “Систем не може да нађе запосленог по задатој вредности”. (ИА)

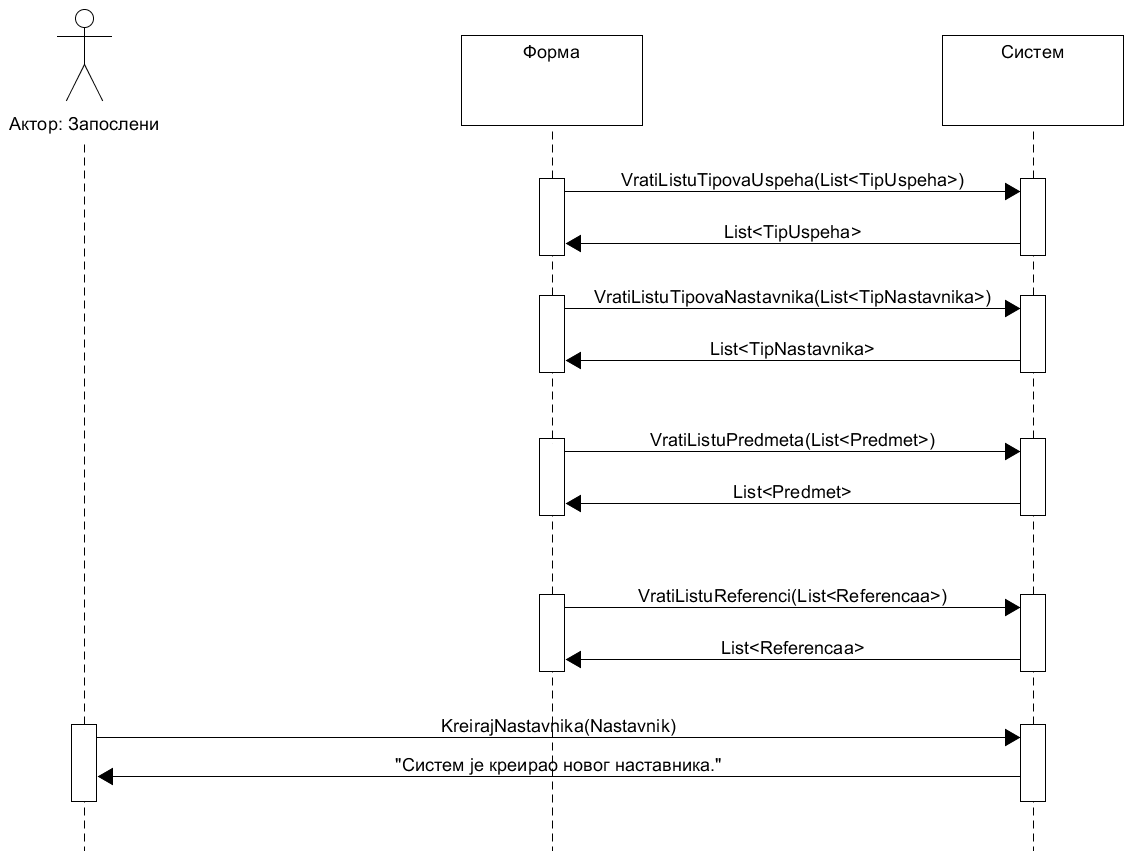


Са наведених секвенцних дијаграма уочава се 1 системска операција коју треба пројектовати:

1. *signal* **PronadjiZaposlenog**(Zaposleni);

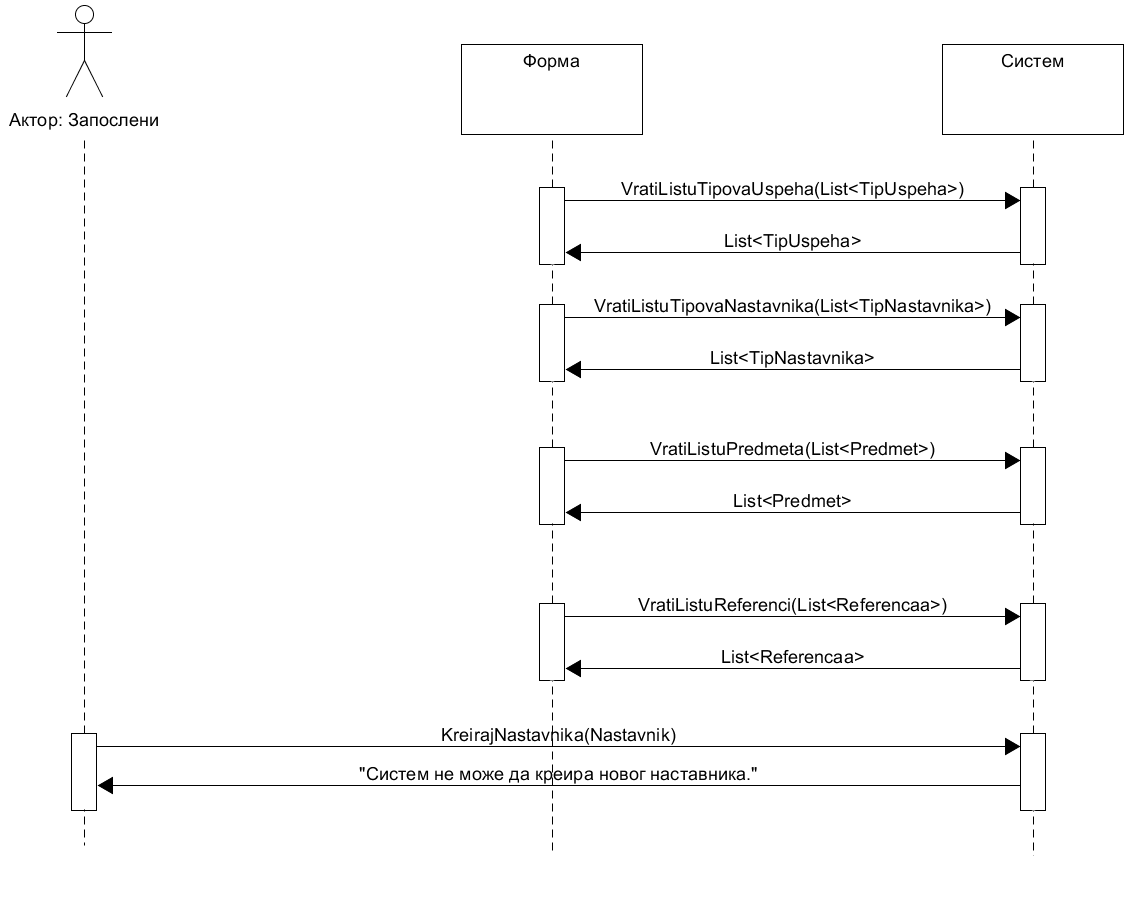
#### 3.2.1.2 ДС2: Дијаграм секвенци случаја коришћења – Креирање новог наставника

1. Форма **позива** систем да учита листу типова успеха. (АПСО)
2. Систем **учитава** листу типова успеха. (ИА)
3. Форма **позива** систем да учита листу типова наставника. (АПСО)
4. Систем **учитава** листу типова наставника. (ИА)
5. Форма **позива** систем да учита листу предмета. (АПСО)
6. Систем **учитава** листу предмета. (ИА)
7. Форма **позива** систем да учита листу референци. (ИА)
8. Систем **учитава** листу референци. (ИА)
9. Запослени **позива** систем да креира новог наставника. (АПСО)
10. Систем **приказује** запосленом поруку: “Систем је креирао новог наставника“. (ИА)



**Алтернативна сценарија**

10.1 Уколико систем не може да креира новог наставника он приказује запосленом поруку: “Систем не може да креира новог наставника”. (ИА)



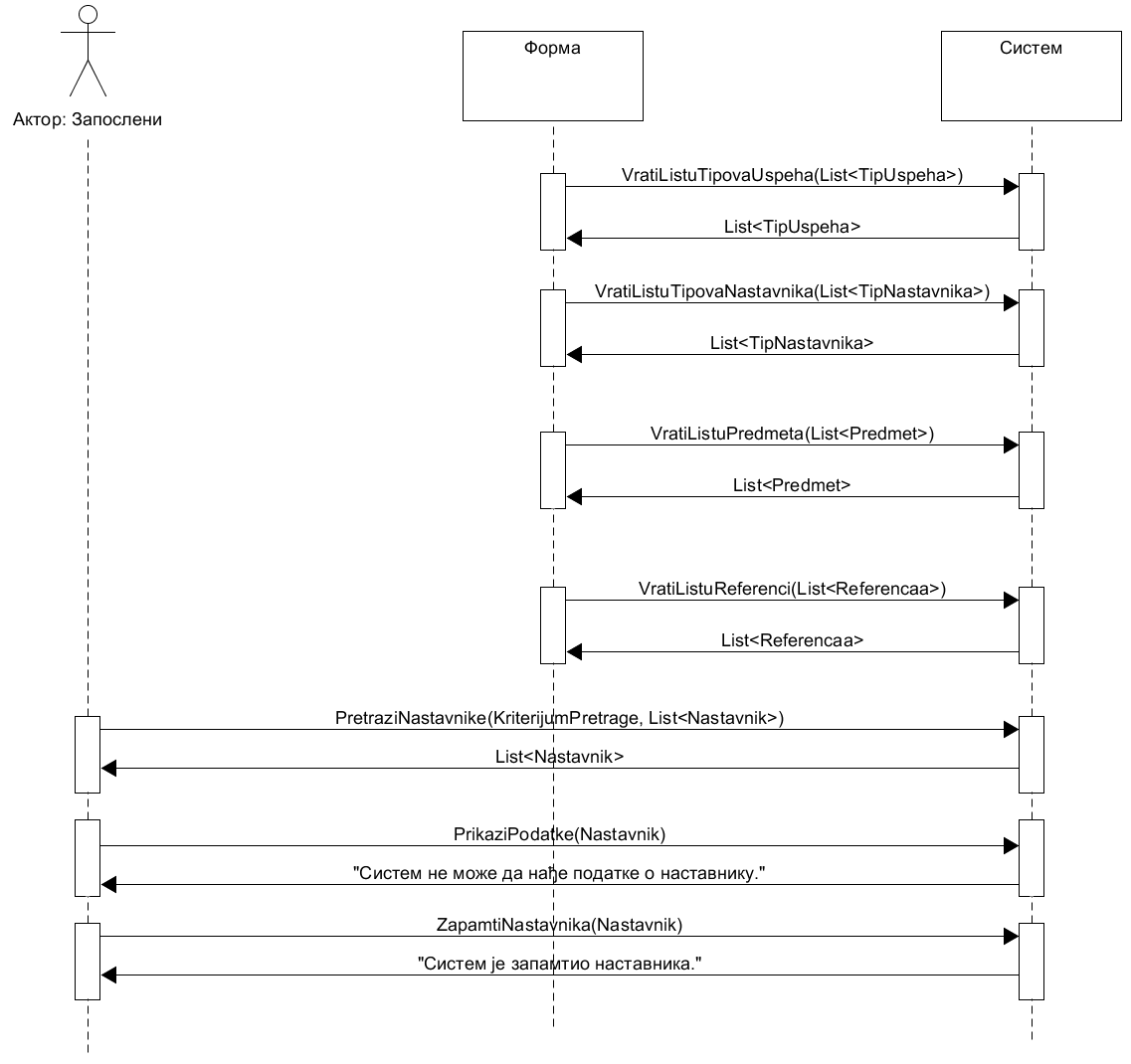
Са наведених секвенцних дијаграма уочавају се још 5 системских операција које треба пројектовати:

1. *signal* **VratiListuTipovaUspeha**(List<TipUspeha>);
2. *signal* **VratiListuTipovaNastavnika**(List<TipNastavnika>);
3. *signal* **VratiListuPredmeta**(List<Predmet>);
4. *signal* **VratiListuReferenci**(List<Referenca>);

5. *signal* **KreirajNastavnika**(Nastavnik);

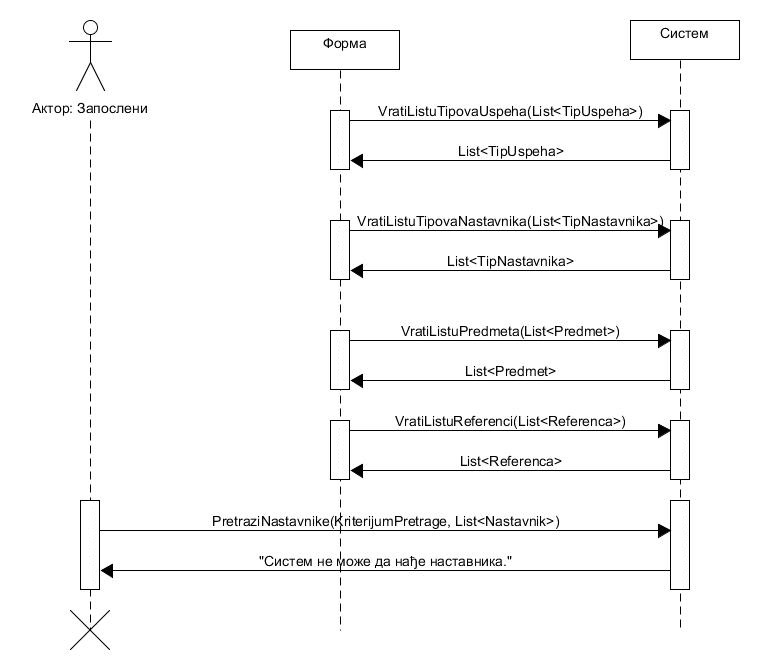
#### 3.2.1.3 ДС3: Дијаграм секвенци случаја коришћења – Промена наставника

1. Форма **позива** систем да учита листу типова успеха. (АПСО)
2. Систем **учитава** листу типова успеха. (ИА)
3. Форма **позива** систем да учита листу типова наставника. (АПСО)
4. Систем **учитава** листу типова наставника. (ИА)
5. Форма **позива** систем да учита листу предмета. (АПСО)
6. Систем **учитава** листу предмета. (ИА)
7. Форма **позива** систем да учита листу референци. (ИА)
8. Систем **учитава** листу референци. (ИА)
9. Запослени **позива** систем да нађе наставника по задатој вредности. (АПСО)
10. Систем **приказује** запосленом наставнике. (ИА)
11. Запослени **позива** систем да прикаже све податке о изабраном наставнику. (АПСО)
12. Систем **приказује** податке о наставнику. (ИА)
13. Запослени **позива** систем да запамти податке о наставнику. (АПСО)
14. Систем **приказује** запосленом поруку: “Систем је запамтио наставника.” (ИА)

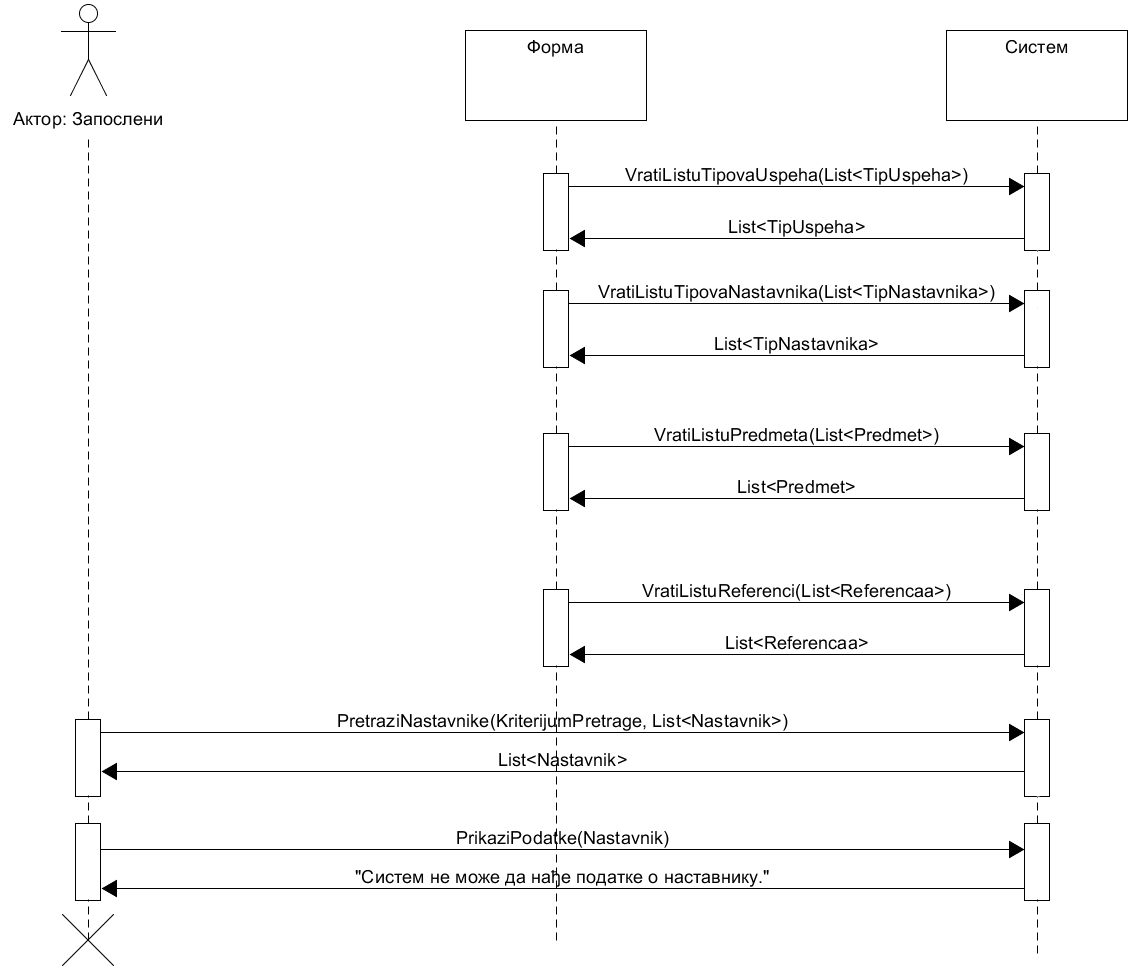


**Алтернативна сценарија**

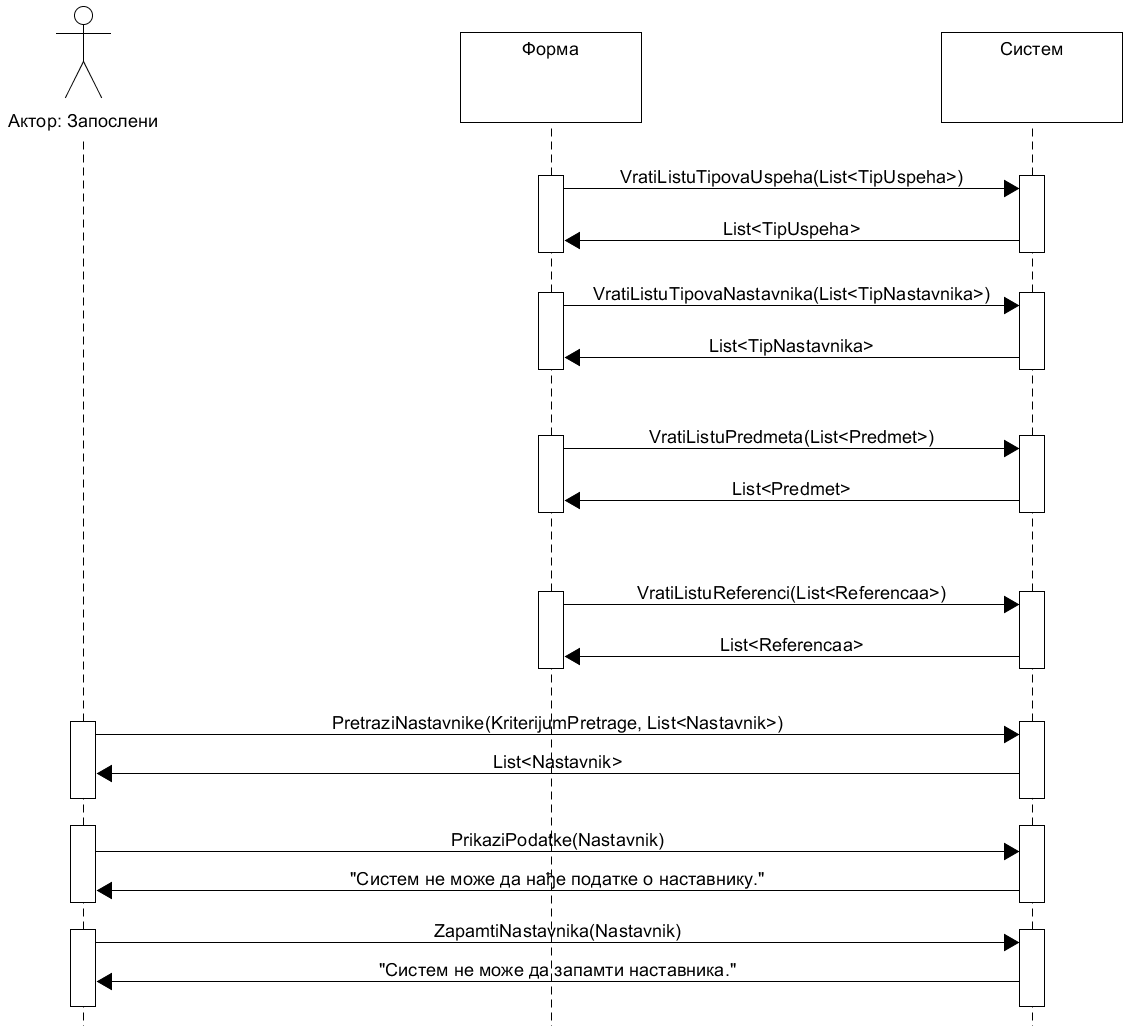
10.1 Уколико систем не може да нађе наставника он приказује запосленом поруку: “Систем не може да нађе наставника по задатој вредности”. Прекида се извршење сценариа. (ИА)



12.1 Уколико систем не може да нађе податке о наставнику он приказује запосленом поруку: “Систем не може да нађе податке о наставнику”. Прекида се извршење сценариа. (ИА)



14.1 Уколико систем не може да запамти податке о наставнику он приказује запосленом поруку “Систем не може да запамти наставника”. (ИА)

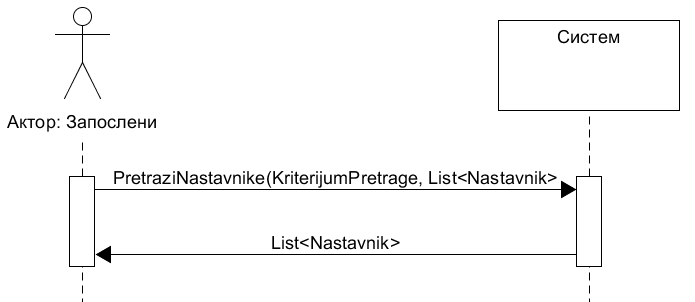


Са наведених секвенцних дијаграма уочавају се још 2 системске операције које треба пројектовати:

1. *signal* **PretraziNastavnike**(KriterijumPretrage, List<Nastavnik>);
2. *signal* **PrikaziPodatke**(Nastavnik);

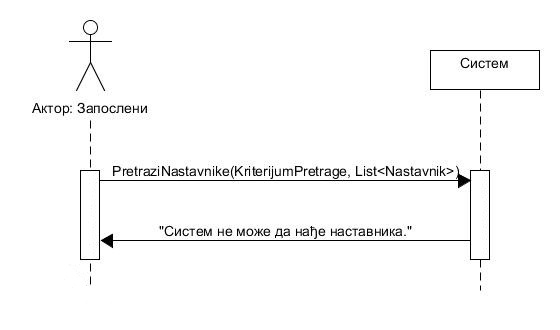
#### 3.2.1.4 ДС4: Дијаграм секвенци случаја коришћења – Претраживање наставника

1. Запослени **позива** систем да нађе наставника по задатој вредности. (АПСО)
2. Систем **приказује** запосленом податке о наставницима. (ИА)



**Алтернативна сценарија**

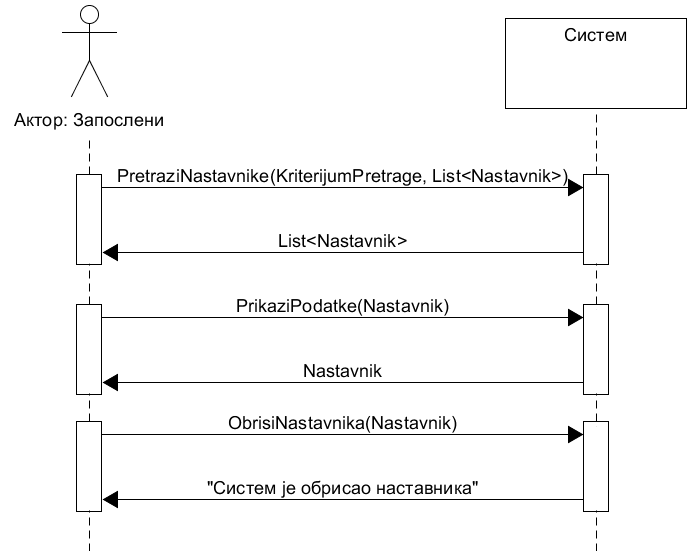
2.1 Уколико систем не може да нађе наставника он приказује запосленом поруку: “Систем не може да нађе наставника по задатој вредности”. (ИА)



Са наведених секвенцних дијаграма не уочава се ниједна додатна системска операција коју треба пројектовати.

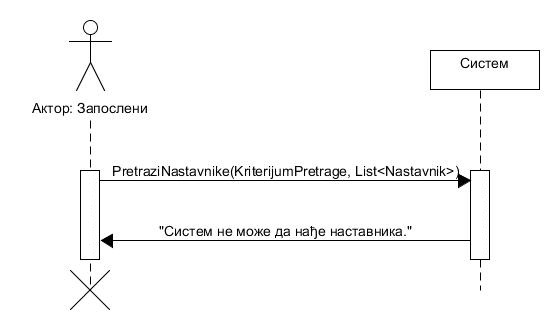
#### 3.2.1.5 ДС5: Дијаграм секвенци случаја коришћења – Брисање наставника

1. Запослени **позива** систем да нађе наставника по задатој вредности. (АПСО)
2. Систем **приказује** запосленом наставнике. (ИА)
3. Запослени **позива** систем да прикаже све податке о изабраном наставнику. (АПСО)
4. Систем **приказује** податке о наставнику. (ИА)
5. Запослени **позива** систем да обрише наставника. (АПСО)
6. Систем **приказује** запосленом поруку: “Систем је обрисао наставника.” (ИА)

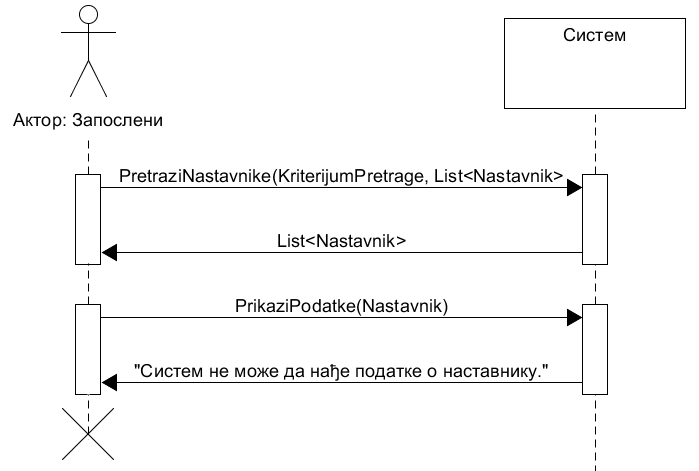


**Алтернативна сценарија**

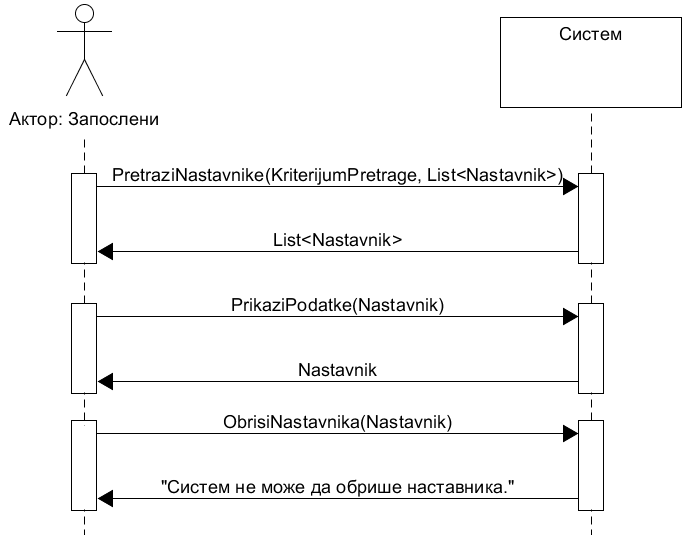
2.1 Уколико систем не може да нађе наставника он приказује запосленом поруку: “Систем не може да нађе наставника по задатој вредности”. Прекида се извршење сценариа. (ИА)



4.1 Уколико систем не може да нађе податке о наставнику он приказује запосленом поруку: “Систем не може да нађе податке о наставнику”. Прекида се извршење сценариа. (ИА)



6.1 Уколико систем не може да обрише наставника он приказује запосленом поруку: “Систем не може да обрише наставника”. (ИА)

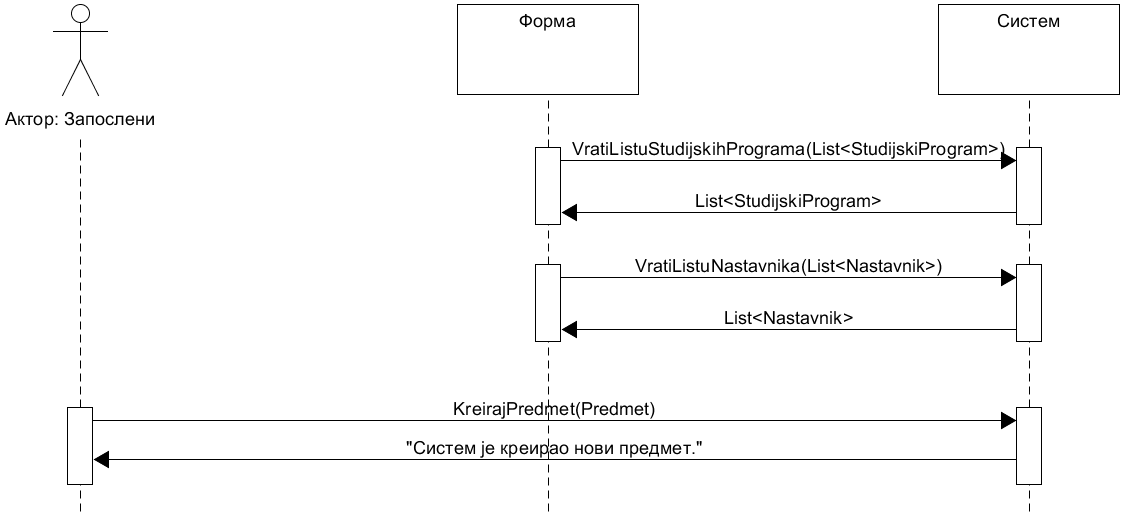


Са наведених секвенцних дијаграма уочава се још 1 системска операција коју треба пројектовати:

1. *signal* **ObrisiNastavnika**(Nastavnik);

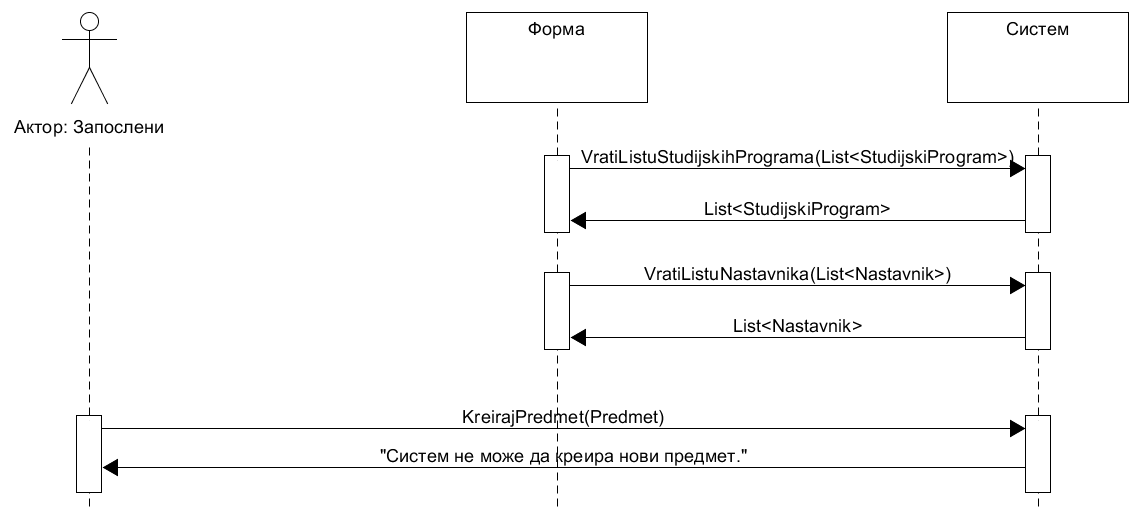
#### 3.2.1.6 ДС6: Дијаграм секвенци случаја коришћења – Креирање новог предмета

1. Форма **позива** систем да учита листу студијских програма. (АПСО)
2. Систем **учитава** листу студијских програма. (ИА)
3. Форма **позива** систем да учита листу наставника. (АПСО)
4. Систем **учитава** листу наставника. (ИА)
5. Запослени **позива** систем да креира нови предмет. (АПСО)
6. Систем **приказује** запосленом поруку: “Систем је креирао нови предмет“. (ИА)



**Алтернативна сценарија**

6.1 Уколико систем не може да креира нови предмет он приказује запосленом поруку: “Систем не може да креира нови предмет ”. (ИА)

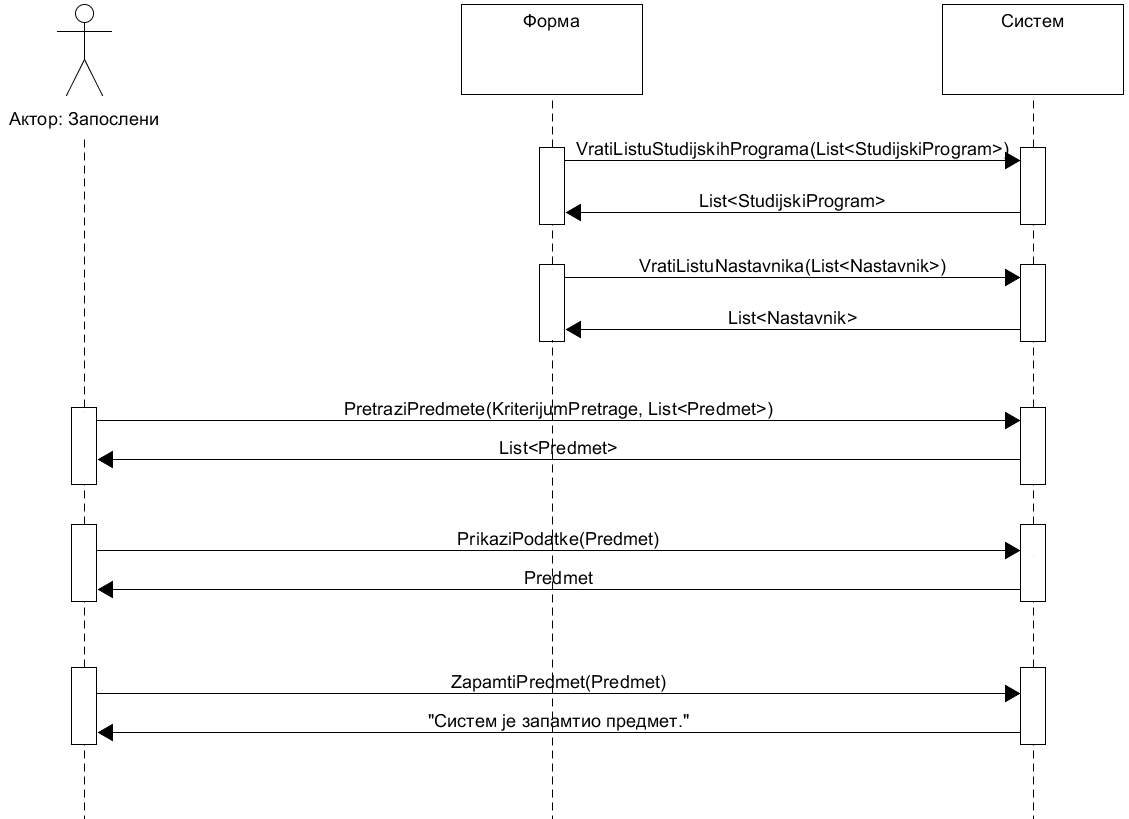


Са наведених секвенцних дијаграма уочавају се још 3 системске операције које треба пројектовати:

1. *signal* **VratiListuStudijskihPrograma**(List<StudijskiProgram>);
2. *signal* **VratiListuNastavnika**(List<Nastavnik>);
3. *signal* **KreirajPredmet**(Predmet);

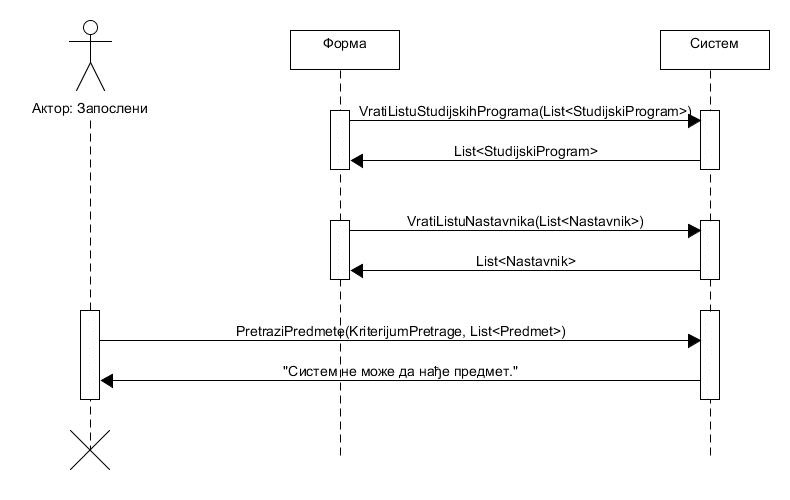
#### 3.2.1.7 ДС7: Дијаграм секвенци случаја коришћења – Промена предмета

1. Форма **позива** систем да учита листу студијских програма. (АПСО)
2. Систем **учитава** листу студијских програма. (ИА)
3. Форма **позива** систем да учита листу наставника. (АПСО)
4. Систем **учитава** листу наставника. (ИА)
5. Запослени **позива** систем да нађе предмет по задатој вредности. (АПСО)
6. Систем **приказује** запосленом предмете. (ИА)
7. Запослени **позива** систем да прикаже све податке о изабраном предмету. (АПСО)
8. Систем **приказује** податке о предмету. (ИА)
9. Запослени **позива** систем да запамти податке о предмету. (АПСО)
10. Систем **приказује** запосленом запамћени предмет и поруку: “Систем је запамтио предмет.” (ИА)

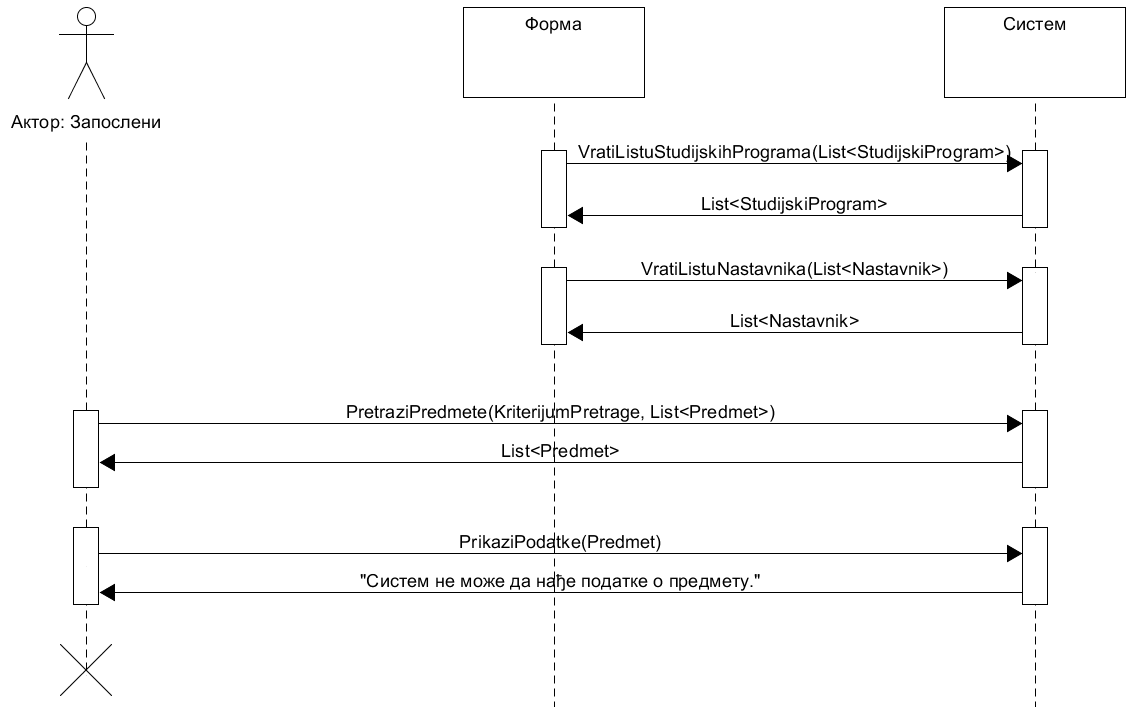


**Алтернативна сценарија**

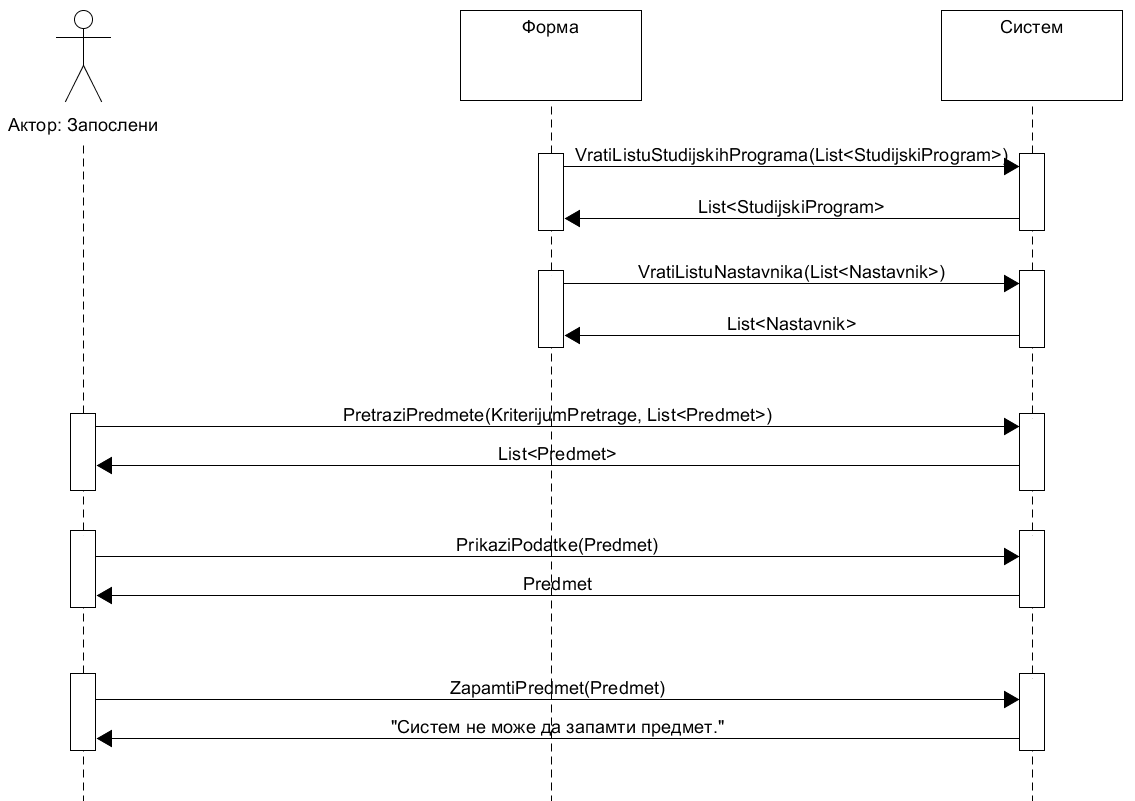
2.1 Уколико систем не може да нађе предмет он приказује запосленом поруку: “Систем не може да нађе предмет по задатој вредности”. Прекида се извршење сценариа. (ИА)



4.1 Уколико систем не може да нађе податке о предмету он приказује запосленом поруку: “Систем не може да нађе податке о предмету”. Прекида се извршење сценариа. (ИА)



6.1 Уколико систем не може да запамти податке о предмету он приказује запосленом поруку “Систем не може да запамти предмет.” (ИА)

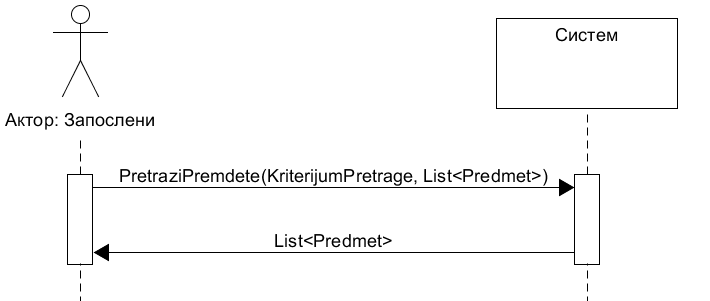


Са наведених секвенцних дијаграма уочавају се још 3 системске операције које треба пројектовати:

1. *signal* **PretraziPredmete**(KriterijumPretrage, List<Predmet>);
2. *signal* **PrikaziPodatke**(Predmet);
3. *signal* **ZapamtiPredmet**(Predmet);

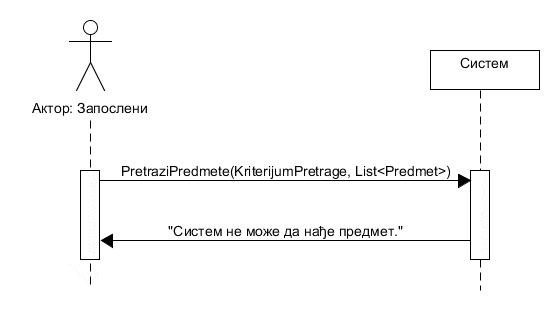
#### 3.2.1.8 ДС8: Дијаграм секвенци случаја коришћења – Претраживање предмета

1. Запослени **позива** систем да нађе предмет по задатој вредности. (АПСО)
2. Систем **приказује** запосленом податке о предметима. (ИА)



**Алтернативна сценарија**

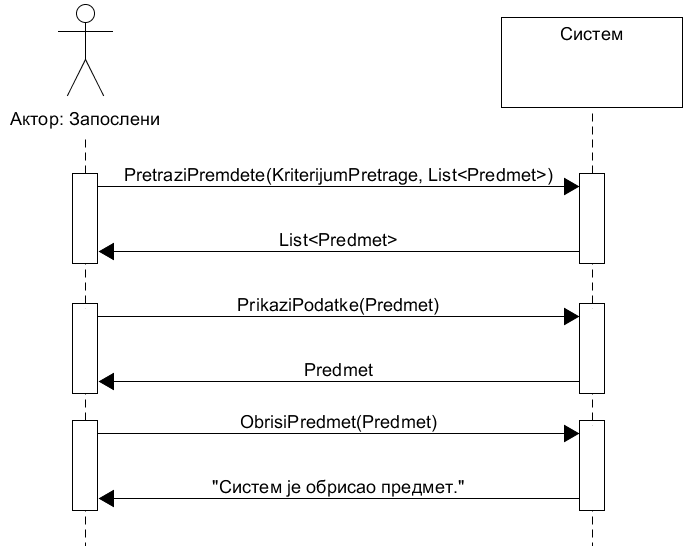
2.1 Уколико систем не може да нађе предмет он приказује запосленом поруку: “Систем не може да нађе предмет по задатој вредности”. (ИА)



Са наведених секвенцних дијаграма не уочава се ниједна додатна системска операција коју треба пројектовати.

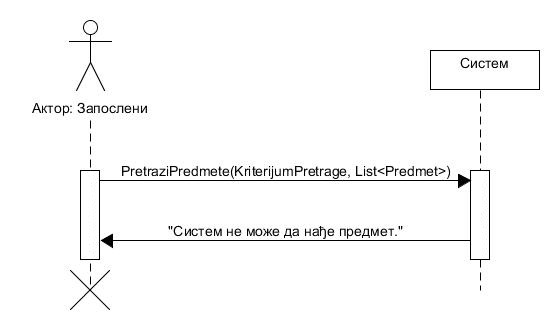
#### 3.2.1.9 ДС9: Дијаграм секвенци случаја коришћења – Брисање предмета

1. Запослени **позива** систем да нађе предмет по задатој вредности. (АПСО)
2. Систем **приказује** запосленом предмете. (ИА)
3. Запослени **позива** систем да прикаже све податке о изабраном предмету. (АПСО)
4. Систем **приказује** податке о предмету. (ИА)
5. Запослени **позива** систем да обрише предмет. (АПСО)
6. Систем **приказује** запосленом поруку: “Систем је обрисао предмет.” (ИА)

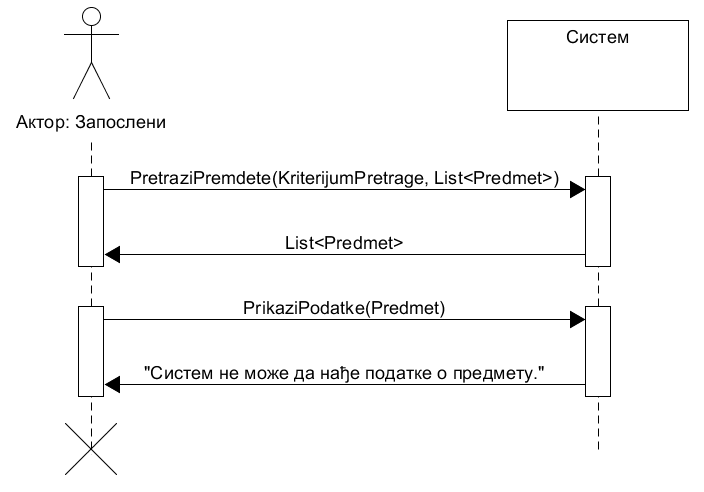


**Алтернативна сценарија**

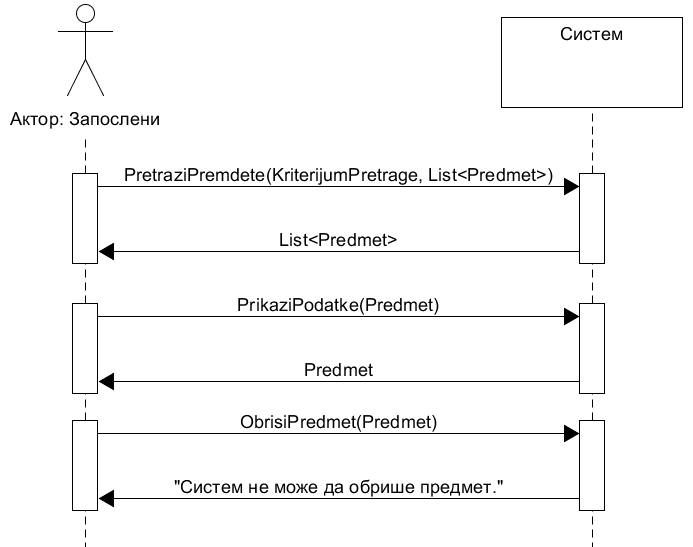
2.1 Уколико систем не може да нађе предмет он приказује запосленом поруку: “Систем не може да нађе предмет по задатој вредности”. Прекида се извршење сценариа. (ИА)



4.1 Уколико систем не може да нађе податке о предмету он приказује запосленом поруку: “Систем не може да нађе податке о предмету”. Прекида се извршење сценариа. (ИА)



6.1 Уколико систем не може да обрише предмет он приказује запосленом поруку “Систем не може да обрише предмет ”. (ИА)

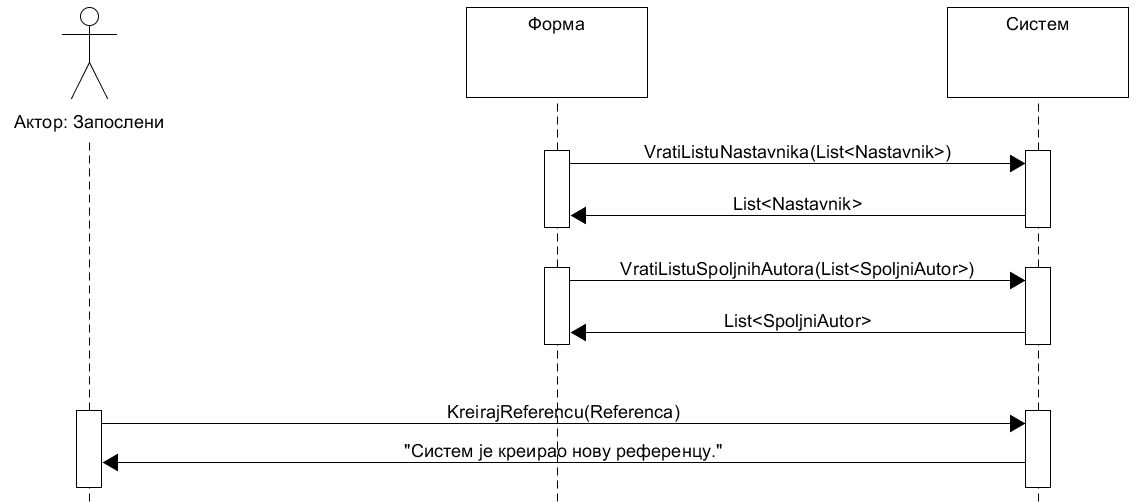


Са наведених секвенцних дијаграма уочава се још 1 системска операција коју треба пројектовати:

1. *signal* **ObrisiPredmet**(Predmet);

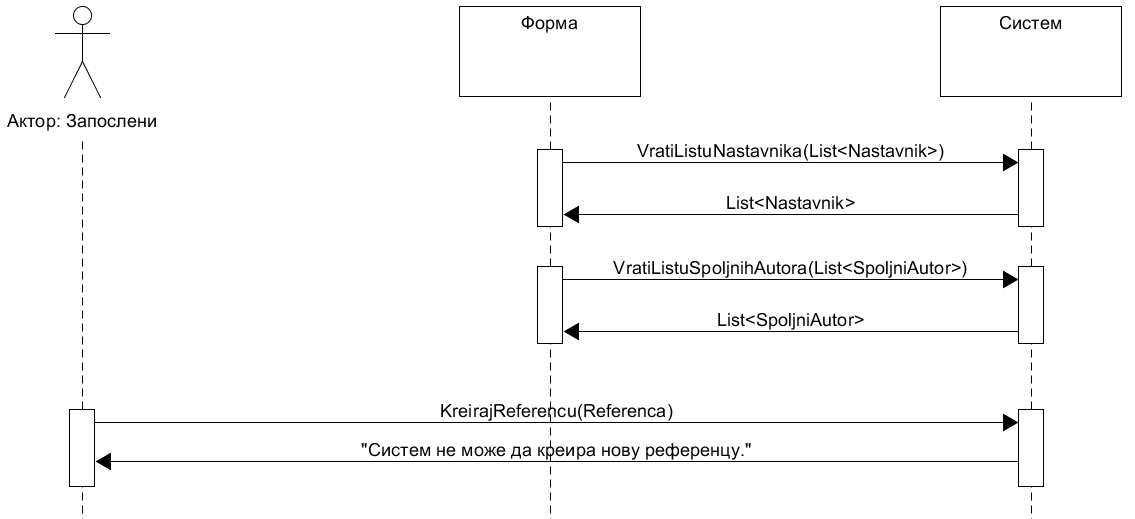
#### 3.2.1.10 ДС10: Дијаграм секвенци случаја коришћења – Креирање нове референце

1. Форма **позива** систем да учита листу наставника. (АПСО)
2. Систем **учитава** листу наставника. (ИА)
3. Форма **позива** систем да учита листу спољних аутора. (АПСО)
4. Систем **учитава** листу спољних аутора. (ИА)
5. Запослени **позива** систем да креира нову референцу. (АПСО)
6. Систем **приказује** поруку: “Систем је креирао нову референцу“. (ИА)



**Алтернативна сценарија**

6.1 Уколико систем не може да креира нову референцу он приказује запосленом поруку: “Систем не може да креира нову референцу”. (ИА)



Са наведених секвенцних дијаграма уочавају се још 2 системске операције које треба пројектовати:

1. *signal* **VratiListuSpoljnihAutora**(List<SpoljniAutor>);
2. *signal* **KreirajReferencu**(Referenca);

Као резултат анализе сценарија, добијено је укупно 19 системских операција које треба пројектовати:

1. *signal* **PronadjiZaposlenog**(Zaposleni);
2. *signal* **VratiListuTipovaUspeha**(List<TipUspeha>);
3. *signal* **VratiListuTipovaNastavnika**(List<TipNastavnika>);
4. *signal* **VratiListuPredmeta**(List<Predmet>);
5. *signal* **VratiListuReferenci**(List<Referenca>);
6. *signal* **KreirajNastavnika**(Nastavnik);
7. *signal* **ZapamtiNastavnika**(Nastavnik);
8. *signal* **PretraziNastavnike**(KriterijumPretrage, List<Nastavnik>);
9. *signal* **PrikaziPodatke**(Nastavnik);
10. *signal* **ObrisiNastavnika**(Nastavnik);
11. *signal* **VratiListuStudijskihPrograma**(List<StudijskiProgram>);
12. *signal* **VratiListuNastavnika**(List<Nastavnik>);
13. *signal* **KreirajPredmet**(Predmet);
14. *signal* **ZapamtiPredmet**(Predmet);
15. *signal* **PretraziPredmete**(KriterijumPretrage, List<Predmet>);
16. *signal* **PrikaziPodatke**(Predmet);
17. *signal* **ObrisiPredmet**(Predmet);
18. *signal* **VratiListuSpoljnihAutora**(List<SpoljniAutor>);
19. *signal* **KreirajReferencu**(Referenca);

### 3.2.2 Дефинисање уговора о системским операцијама

Уговори се праве за системске операције и описују њихово понашање. Они описују шта системска операција треба да ради, али не и како ће то да ради. Сваки уговор се везује за тачно једну системску операцију.

Уговори се састоје из следећих четири секција:

1. **Операција** – назив операције и њени аргументи (улазни и излазни)
2. **Веза** **са** **СК** – називи случајева коришћења у којима се јавља дата системска операција
3. **Предуслови** – услови који морају бити задовољени пре извршења системске операције
4. **Постуслови** – услови који морају бити задовољени након извршења системске операције

#### 3.2.2.1 Уговор UG1: PronadjiZaposlenog

**Операција: PronadjiZaposlenog**(Zaposleni):signal; **Веза са СК:** СК1  
**Предуслови:  
Постуслови:**

#### 3.2.2.2 Уговор UG2: VratiListuTipovaUspeha

**Операција: VratiListuTipovaUspeha**(List<TipUspeha>):signal; **Веза са СК:** СК2, СК3  
**Предуслови:  
Постуслови:**

#### 3.2.2.3 Уговор UG3: VratiListuTipovaNastavnika

**Операција: VratiListuTipovaNastavnika**(List<TipNastavnika>):signal; **Веза са СК:** СК2, СК3  
**Предуслови:  
Постуслови:**

#### 3.2.2.4 Уговор UG4: VratiListuPredmeta

**Операција: VratiListuPredmeta**(List<Predmet>):signal; **Веза са СК:** СК2, СК3  
**Предуслови:  
Постуслови:**

#### 3.2.2.5 Уговор UG5: VratiListuReferenci

**Операција: VratiListuReferenci**(List<Referenca>):signal; **Веза са СК:** СК2, СК3  
**Предуслови:  
Постуслови:**

#### 3.2.2.6 Уговор UG6: KreirajNastavnika

**Операција: KreirajNastavnika**(Nastavnik):signal; **Веза са СК:** СК2  
**Предуслови:** *Вредносна и структурна ограничења над објектом Nastavnik**морају бити задовољена.* **Постуслови:** *Креиран је нови наставник.*

#### 3.2.2.7 Уговор UG7: ZapamtiNastavnika

**Операција: ZapamtiNastavnika**(Nastavnik):signal; **Веза са СК:** СК2, СК3 **Предуслови:** *Вредносна и структурна ограничења над објектом Nastavnik**морају бити задовољена.* **Постуслови:** *Подаци о наставнику су запамћени.*

#### 3.2.2.8 Уговор UG8: PretraziNastavnike

**Операција: PretraziNastavnike**(KriterijumPretrage, List<Nastavnik>):signal; **Веза са СК:** СК3, СК4, СК5 **Предуслови:  
Постуслови:**

#### 3.2.2.9 Уговор UG9: PrikaziPodatke

**Операција: PrikaziPodatke**(Nastavnik):signal; **Веза са СК:** СК3, СК4, СК5 **Предуслови:  
Постуслови:**

#### 3.2.2.10 Уговор UG10: ObrisiNastavnika

**Операција: ObrisiNastavnika**(Nastavnik):signal; **Веза са СК:** СК5 **Предуслови:** *Структурно ограничење над објектом Nastavnik мора бити задовољено.* **Постуслови:** *Наставник је обрисан.*

#### 3.2.2.11 Уговор UG11: VratiListuStudijskihPrograma

**Операција: VratiListuStudijskihPrograma**(List<StudijskiProgram>):signal; **Веза са СК:** СК6, СК7  
**Предуслови:  
Постуслови:**

#### 3.2.2.12 Уговор UG12: VratiListuNastavnika

**Операција: VratiListuNastavnika**(List<Nastavnik>):signal; **Веза са СК:** СК6, СК7, CK10  
**Предуслови:  
Постуслови:**

#### 3.2.2.13 Уговор UG13: KreirajPredmet

**Операција: KreirajPredmet**(Predmet):signal; **Веза са СК:** СК6 **Предуслови:** *Вредносна и структурна ограничења над објектом Predmet**морају бити задовољена.* **Постуслови:** *Креиран је нови предмет.*

#### 3.2.2.14 Уговор UG14: ZapamtiPredmet

**Операција: ZapamtiPredmet**(Predmet):signal; **Веза са СК:** СК6, СК7 **Предуслови:** *Вредносна и структурна ограничења над објектом Predmet**морају бити задовољена.* **Постуслови:** *Подаци о предмету су запамћени.*

#### 3.2.2.15 Уговор UG15: PretraziPredmete

**Операција: PretraziPredmete**(KriterijumPretrage, List<Predmet>):signal; **Веза са СК:** СК7, СК8, СК9 **Предуслови:  
Постуслови:**

#### 3.2.2.16 Уговор UG16: PrikaziPodatke

**Операција: PrikaziPodatke**(Predmet):signal; **Веза са СК:** СК7, СК8, СК9 **Предуслови:  
Постуслови:**

#### 3.2.2.17 Уговор UG17: ObrisiPredmet

**Операција: ObrisiPredmet**(Predmet):signal; **Веза са СК:** СК9 **Предуслови:** *Структурна ограничења над објетком Predmet морају бити задовољена.* **Постуслови:** *Предмет је обрисан.*

#### 3.2.2.18 Уговор UG18: VratiListuSpoljnihAutora

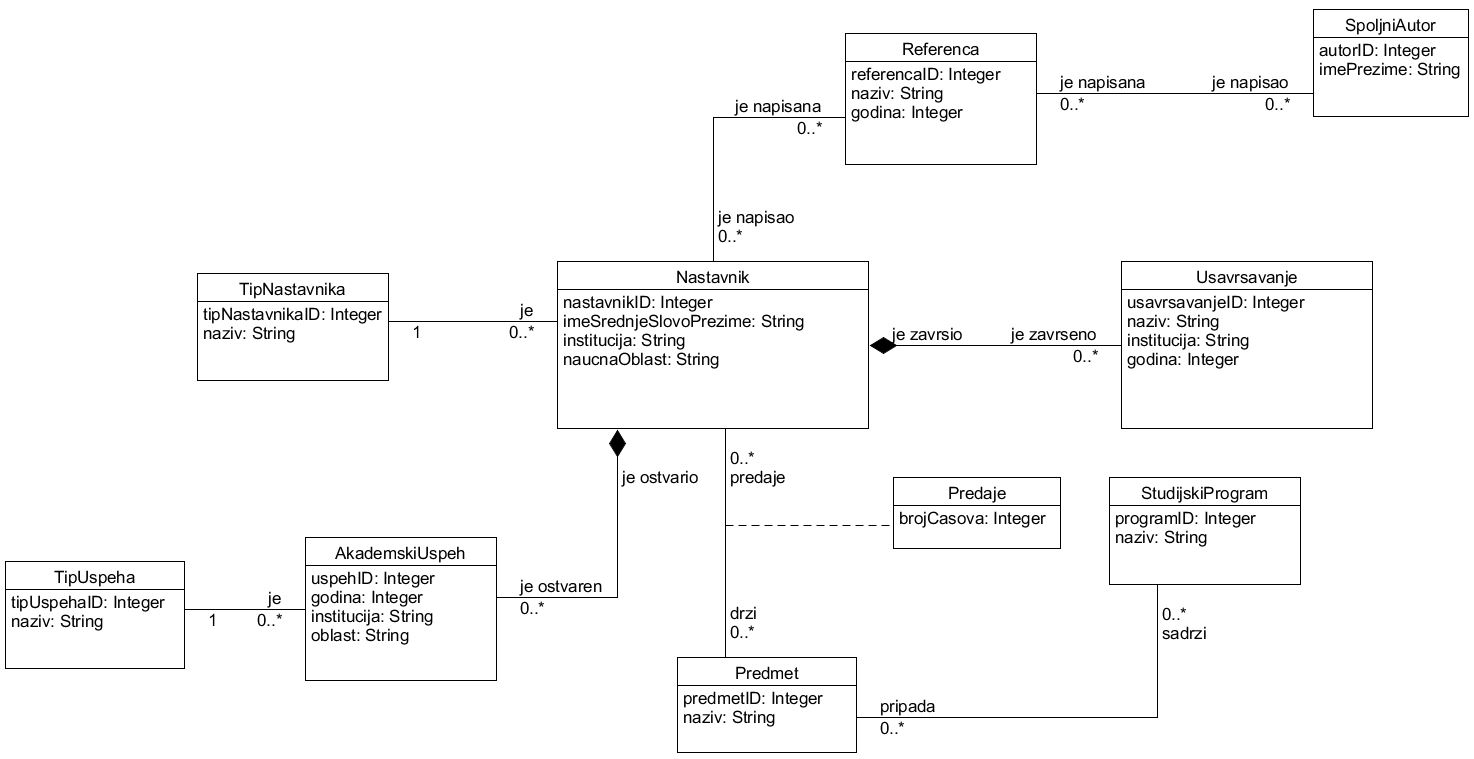
**Операција: VratiListuSpoljnihAutora**(List<SpoljniAutor>):signal; **Веза са СК:** СК10  
**Предуслови:  
Постуслови:**

#### 3.2.2.19 Уговор UG19: KreirajReferencu

**Операција: KreirajReferencu**(Referenca):signal; **Веза са СК:** СК10 **Предуслови:** *Вредносна и структурна ограничења над објектом Referenca**морају бити задовољена.* **Постуслови:** *Креирана је нова референца.*

### 3.2.3 Структура софтверског система – Концептуални (доменски) модел

Концептуални модел садржи доменске класе и везе између њих. Структура софтверског система се описује помоћу концептуалног модела.



### 3.2.4 Структура софтверског система – Релациони модел

На основу датог концептуалног модела, изводи се релациони модел, приказан испод.

**TipNastavnika**(TipNastavnikaID, Naziv) **Nastavnik**(NastavnikID, ImeSrednjeSlovoPrezime, Institucija, NaucnaOblast, *TipNastavnikaID*)  
**TipUspeha**(TipUspehaID, Naziv)   
**AkademskiUspeh**(NastavnikID, UspehID, Godina, Institucija, Oblast, *TipID*)  
**Usavrsavanje**(NastavnikID, UsavrsavanjeID, Naziv, Institucija, Godina)  
**Predmet**(PredmetID, Naziv)  
**Predaje**(NastavnikID, PredmetID, BrojCasova)  
**StudijskiProgram**(ProgramID, Naziv)  
**Referenca**(ReferencaID, Naslov, Godina)  
**SpoljniAutor**(AutorID, ImePrezime)  
**NastavnikReferenca**(NastavnikID, ReferencaID)  
**SpoljniAutorReferenca**(AutorID, ReferencaID)  
**PredmetStudijskiProgram**(PredmetID, ProgramID)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела TipNastavnika** | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT /  UPDATE  CASCADES  Nastavnik  DELETE  RESTRICTED  Nastavnik |
| TipNastavnikaID | Integer | not null |  |  |
| Naziv | String |  |  |  |

Табела . TipNastavnika

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела Nastavnik** | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT  RESTRICTED  TipNastavnika  UPDATE  CASCADES  AkademskiUspeh  Usavrsavanje  Predaje  NastavnikReferenca  RESTRICTED  TipNastavnika  DELETE  CASCADES  AkademskiUspeh  Usavrsavanje  Predaje  NastavnikReferenca |
| NastavnikID | Integer | not null |  |  |
| ImeSrednjeSlovoPrezime | String |  |  |  |
| Institucija | String |  |  |  |
| NaucnaObalst | String |  |  |  |
| TipNastavnikaID | Integer | not null |  |  |

Табела . Nastavnik

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела TipUspeha** | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT /  UPDATE  CASCADES  AkademskiUspeh  DELETE  RESTRICTED  AkademskiUspeh |
| TipUspehaID | Integer | not null |  |  |
| Naziv | String |  |  |  |

Табела . TipUspeha

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела AkademskiUspeh** | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT  RESTRICTED  TipUspeha  Nastavnik  UPDATE  RESTRICTED  TipUspeha  Nastavnik  DELETE / |
| NastavnikID | Integer | not null |  |  |
| UspehID | Integer | not null |  |  |
| Godina | Integer |  |  |  |
| Institucija | String |  |  |  |
| Oblast | String |  |  |  |
| TipID | Integer | not null |  |  |

Табела . AkademskiUspeh

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела Usavrsavanje** | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT  RESTRICTED  Nastavnik  UPDATE  RESTRICTED  Nastavnik  DELETE / |
| NastavnikID | Integer | not null |  |  |
| UsavrsavanjeID | Integer | not null |  |  |
| Naziv | String |  |  |  |
| Institucija | String |  |  |  |
| Godina | Integer |  |  |  |

Табела . Usavrsavanje

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела Predmet** | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT /  UPDATE  CASCADES  Predaje  PredmetStudijskiProgram  DELETE  CASCADES  Predaje  PredmetStudijskiProgram |
| PredmetID | Integer | not null |  |  |
| Naziv | String |  |  |  |

Табела . Predmet

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела Predaje** | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT  RESTRICTED  Nastavnik  Predmet  UPDATE  RESTRICTED  Nastavnik  Predmet  DELETE / |
| NastavnikID | Integer | not null |  |  |
| PredmetID | Integer | not null |  |  |
| BrojCasova | Integer |  |  |  |

Табела . Predaje

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела StudijskiProgram** | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT /  UPDATE  CASCADES  PredmetStudijskiProgram    DELETE  CASCADES  PredmetStudijskiProgram |
| ProgramID | Integer | not null |  |  |
| Naziv | String |  |  |  |

Табела . StudijskiProgram

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела Referenca** | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT /  UPDATE  CASCADES  NastavnikReferenca  SpoljniAutorReferenca  DELETE  CASCADES  NastavnikReferenca  SpoljniAutorReferenca |
| ReferencaID | Integer | not null |  |  |
| Naslov | String |  |  |  |
| Godina | Integer |  |  |  |

Табела . Referenca

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела SpoljniAutor** | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT /  UPDATE  CASCADES  SpoljniAutorReferenca  DELETE  CASCADES  SpoljniAutorReferenca |
| AutorID | Integer | not null |  |  |
| ImePrezime | String |  |  |  |

Табела . SpoljniAutor

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела NastavnikReferenca** | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT  RESTRICTED  Nastavnik  Referenca  UPDATE  RESTRICTED  Nastavnik  Referenca    DELETE / |
| NastavnikID | Integer | not null |  |  |
| ReferencaID | Integer | not null |  |  |

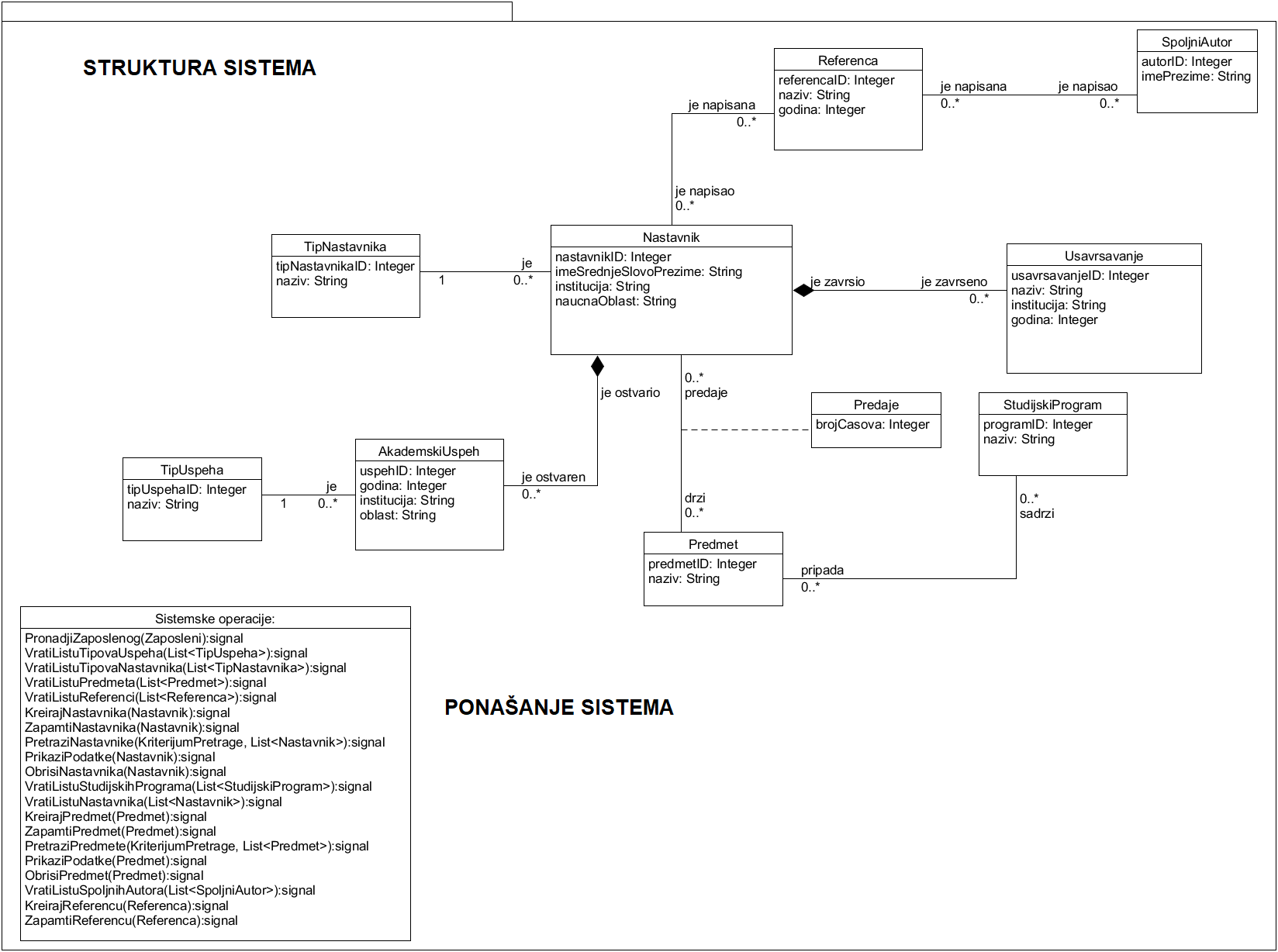
Табела . NastavnikReferenca

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела SpoljniAutorReferenca** | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT  RESTRICTED  SpoljniAutor  Referenca  UPDATE  RESTRICTED  SpoljniAutor  Referenca    DELETE / |
| AutorID | Integer | not null |  |  |
| ReferencaID | Integer | not null |  |  |

Табела . SpoljniAutorReferenca

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Табела PredmetStudijskiProgram** | | **Просто вредносно ограничење** | | **Сложено вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | **Међузав. атрибута једне табеле** | **Међузав. атрибута више табела** | INSERT  RESTRICTED  Predmet  StudijskiProgram  UPDATE  RESTRICTED  Predmet  StudijskiProgram    DELETE / |
| PredmetID | Integer | not null |  |  |
| ProgramID | Integer | not null |  |  |

Табела . PredmetStudijskiProgram



## 3.3 Пројектовање

Фаза пројектовања описује физичку структуру и понашање софтверског система (архитектуру софтверског система). Архитектура софтверског система је тронивојска и састоји се од следећих нивоа:

* Кориснички интерфејс
* Апликациона логика
* Складиште података

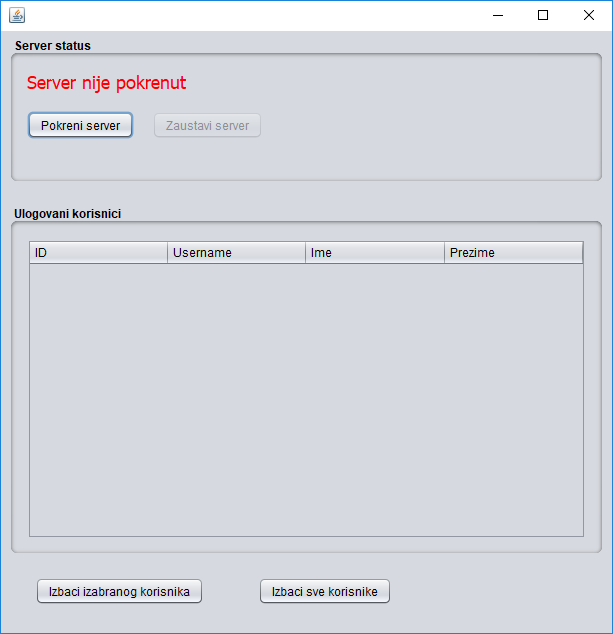
Ниво корисничког интерфејса је на страни клијента, а апликациона логика и складиште података на страни сервера.

### 3.3.1 Пројектовање корисничког интерфејса

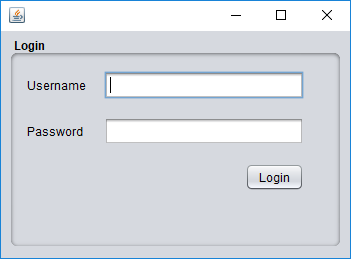
Кориснички интерфејс представља реализацију улаза и/или излаза софтверског система и састоји се од екранских форми и контролера корисничког интерфејса.

Кориснички интерфејс је дефинисан преко скупа екранских форми. Сценарија коришћења екранских форми су директно повезана са сценаријима случајева коришћења.

Изглед почетне клијентске и серверске екрансе форме је дат на сликама испод.



Слика . Почетна серверска екранска форма



Слика . Почетна клијентска екранска форма

#### 3.3.1.1 СК1: Случај коришћења - Пријављивање запосленог

**Назив СК**

Пријављивање запосленог

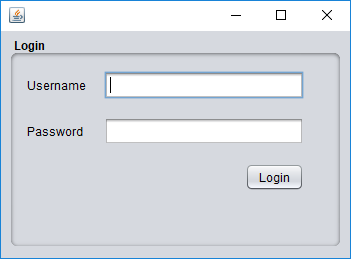
**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

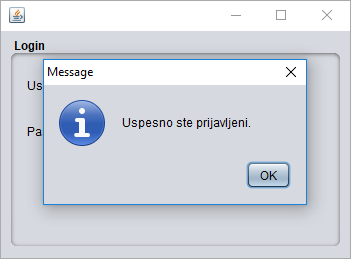
**Предуслов**: Систем је укључен и приказује форму за пријављивање запосленог.



Запослени уноси своје корисничко име и шифру у одговарајућа поља и кликом на дугме Login се пријављује на систем.

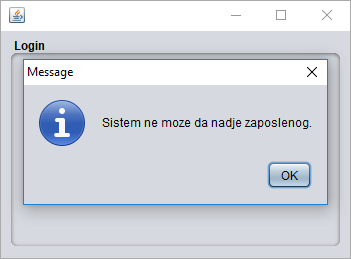
**Основни сценарио СК**

1. Запослени **уноси** податке за аутентификацију запосленог. (АПУСО)
2. Запослени **позива** систем да нађе запосленог по задатој вредности. (АПСО)
3. Систем **тражи** запосленог по задатој вредности. (СО)
4. Систем **приказује** запосленом поруку: “Успешно сте пријављени”. (ИА)



**Алтернативна сценарија**

4.1 Уколико систем не може да нађе запосленог он приказује запосленом поруку: “Систем не може да нађе запосленог по задатој вредности”. (ИА)



#### 3.3.1.2 СК2: Случај коришћења – Креирање новог наставника

**Назив СК**

Креирање новог наставника

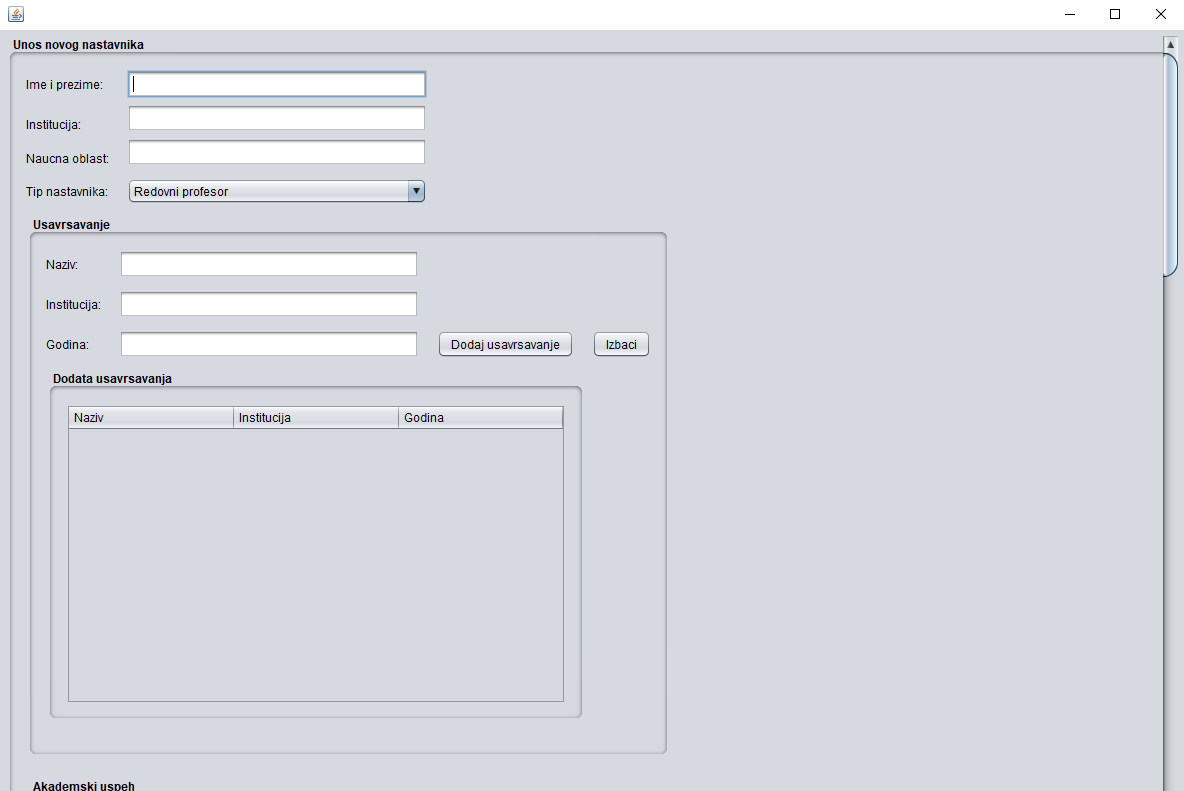
**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

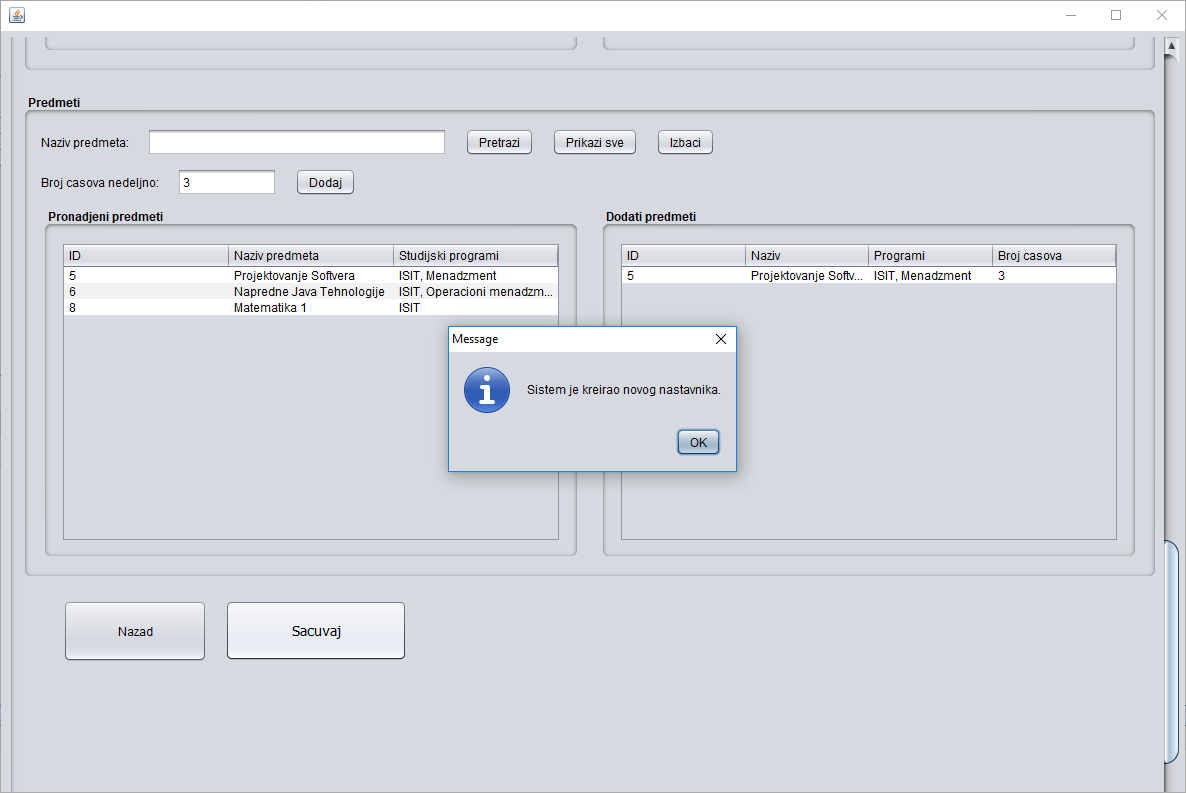
**Предуслов**: Систем је укључен, запослени je улогован под својом шифром и учитане су листе типова наставника, типова академских успеха, предмета и референци. Систем приказује форму за рад са наставником.



Запослени уноси податке о професору у одговарајућа поља и кликом на дугме Sacuvaj, систем креира новог професора у бази.

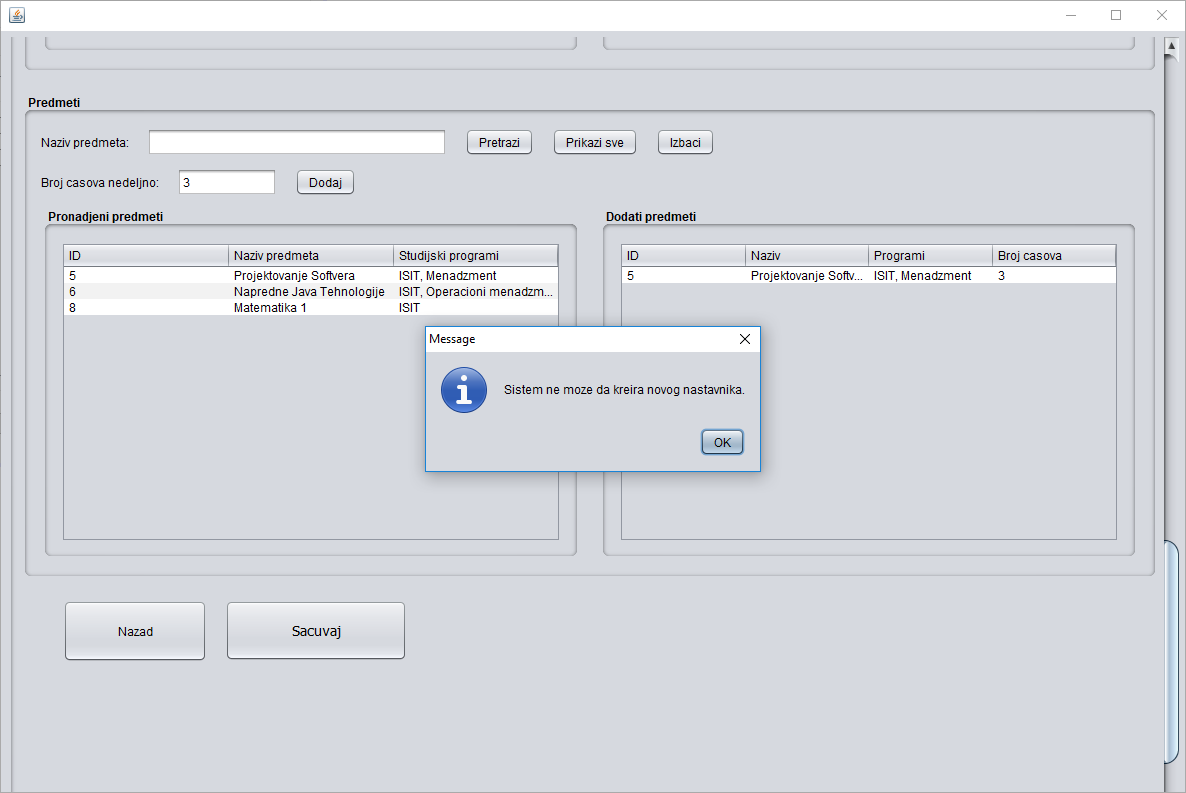
**Основни сценарио СК**

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



**Алтернативна сценарија**

5.1 Уколико систем не може да креира новог наставника он приказује запосленом поруку: “Систем не може да креира новог наставника”. (ИА)



#### 3.3.1.3 СК3: Случај коришћења – Промена наставника

**Назив СК**

Промена наставника

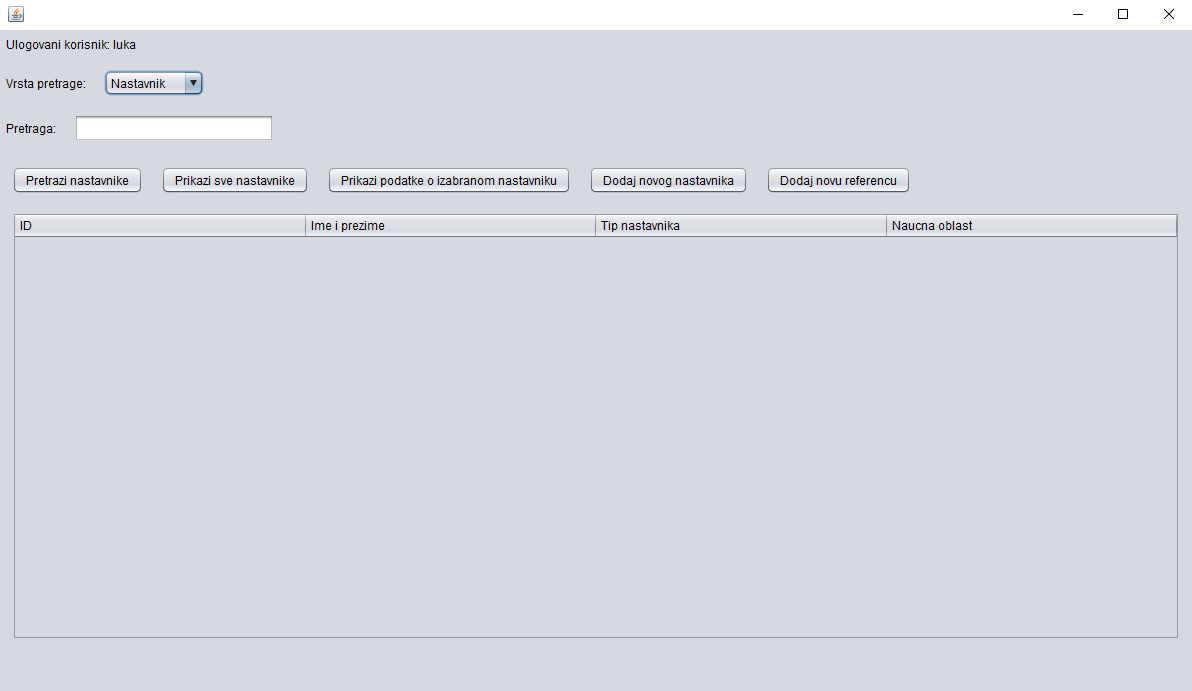
**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

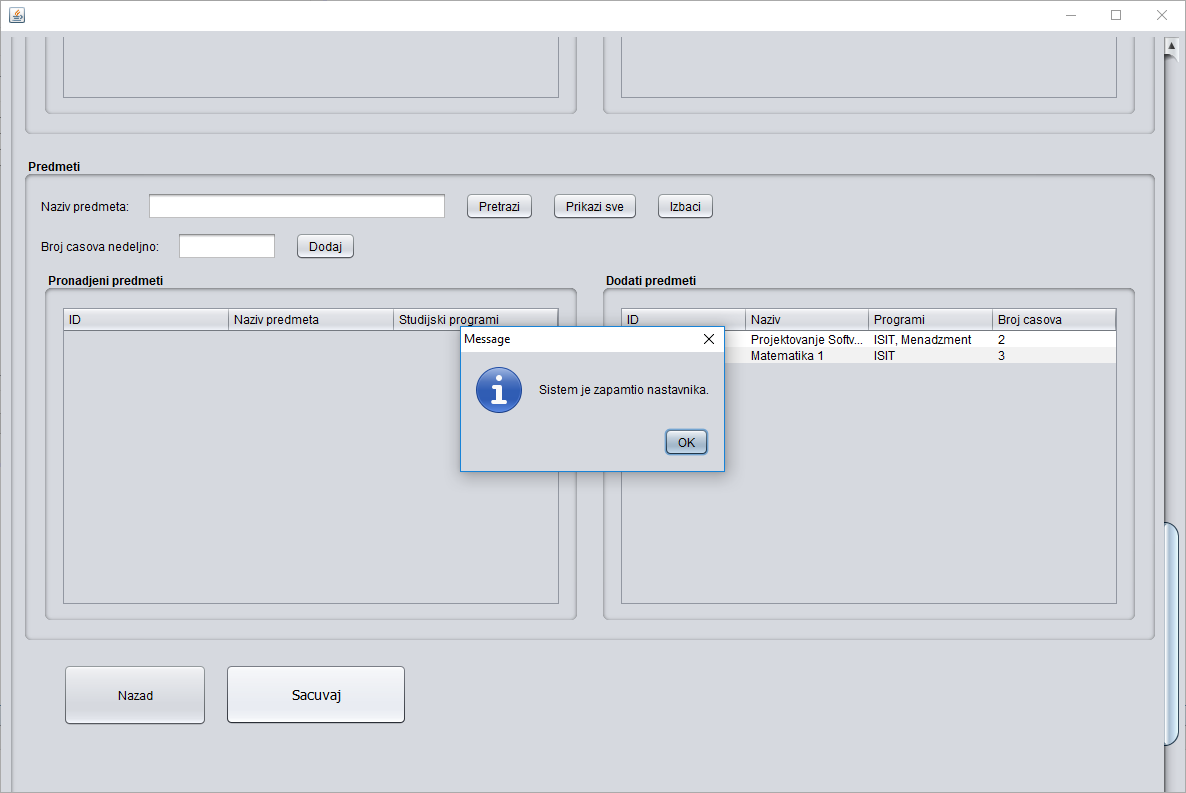
**Предуслов**: Систем је укључен, запослени je улогован под својом шифром и учитане су листе типова наставника, типова академских успеха, предмета и референци. Систем приказује форму за рад са наставником.



Запослени уноси име наставника у поље за претрагу, затим бира једног од пронађених наставника и кликом на дугме Prikazi podatke o izabranom nastavniku, отвара се форма на којој се приказују сви подаци о датом наставнику. Кликом на дугме Izmeni nastavnika, отвара се нова форма са постојећим подацима о наставнику које запослени може да мења. Кликом на дугме Sacuvaj, систем бележи све промене.

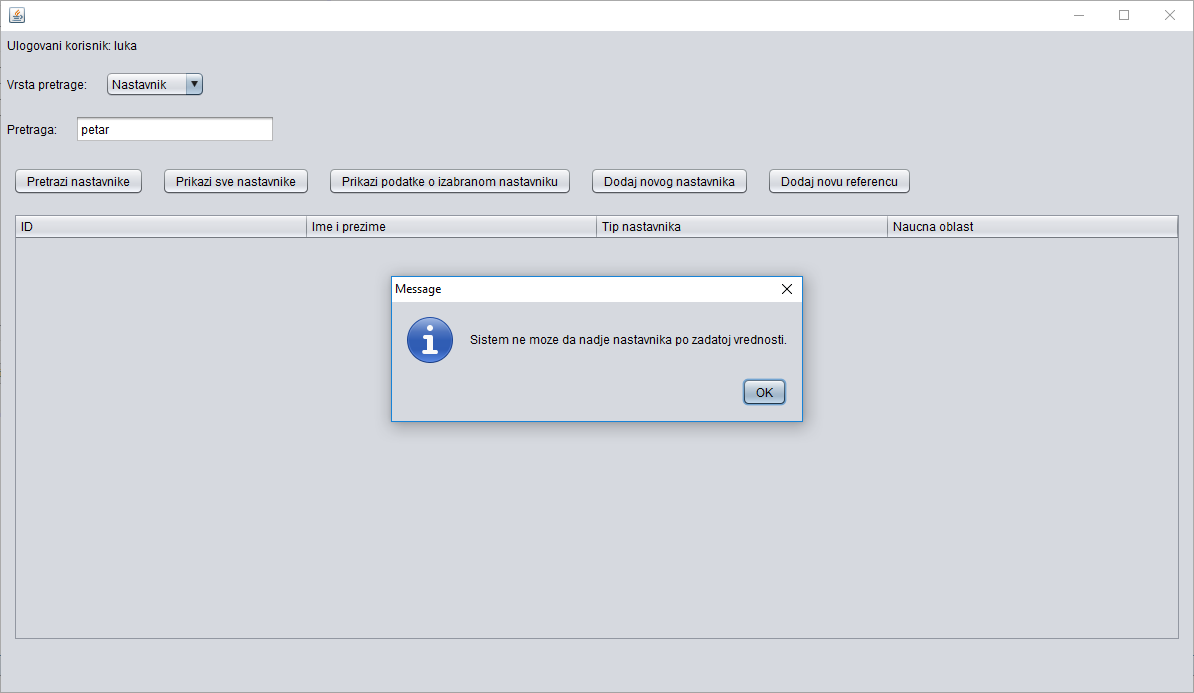
**Основни сценарио СК**

1. Запослени **уноси** вредностпо којој претражује наставника. (АПУСО)
2. Запослени **позива** систем да нађе наставника по задатој вредности. (АПСО)
3. Систем **тражи** наставника по задатој вредности. (СО)
4. Систем **приказује** запосленом наставнике. (ИА)
5. Запослени **бира** жељеног наставника из листе. (АПУСО)
6. Запослени **позива** систем да прикаже све податке о изабраном наставнику. (АПСО)
7. Систем **тражи** податке о изабраном наставнику. (СО)
8. Систем **приказује** податке о наставнику. (ИА)
9. Запослени **уноси** **(мења)** податке о наставнику. (АПУСО)
10. Запослени **контролише** да ли је коректно унео податке о наставнику. (АНСО)
11. Запослени **позива** систем да запамти податке о наставнику. (АПСО)
12. Систем **памти** податке о наставнику. (СО)
13. Систем **приказује** запосленом поруку: “Систем је запамтио наставника.” (ИА)

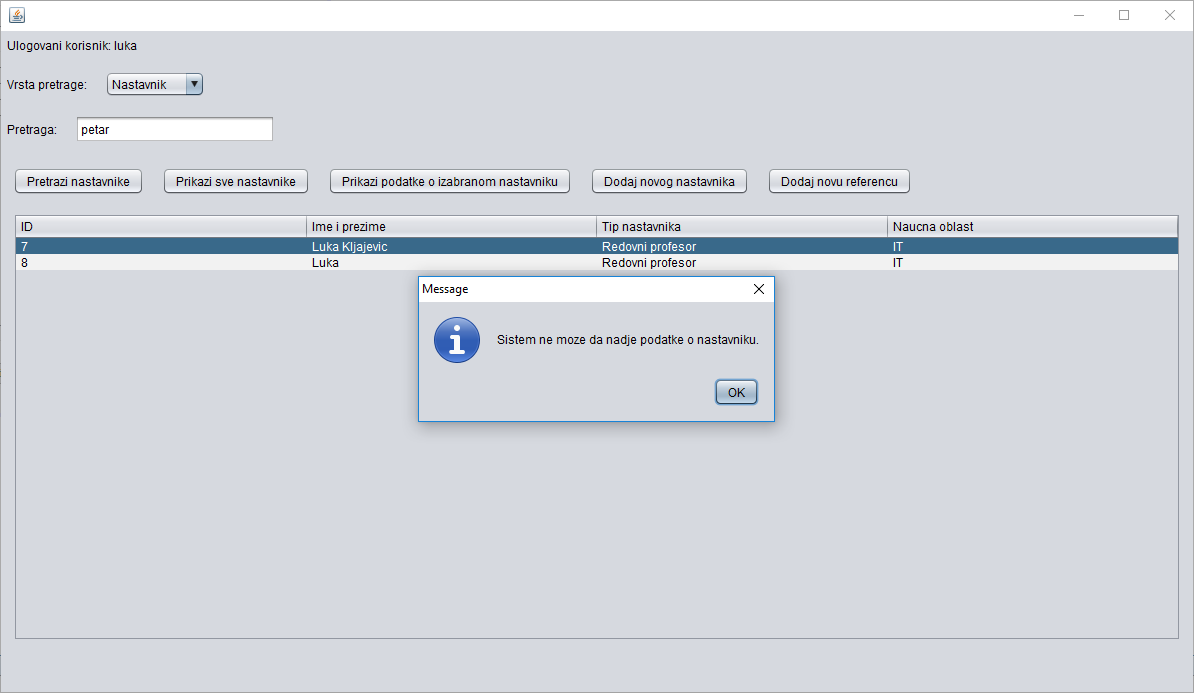


**Алтернативна сценарија**

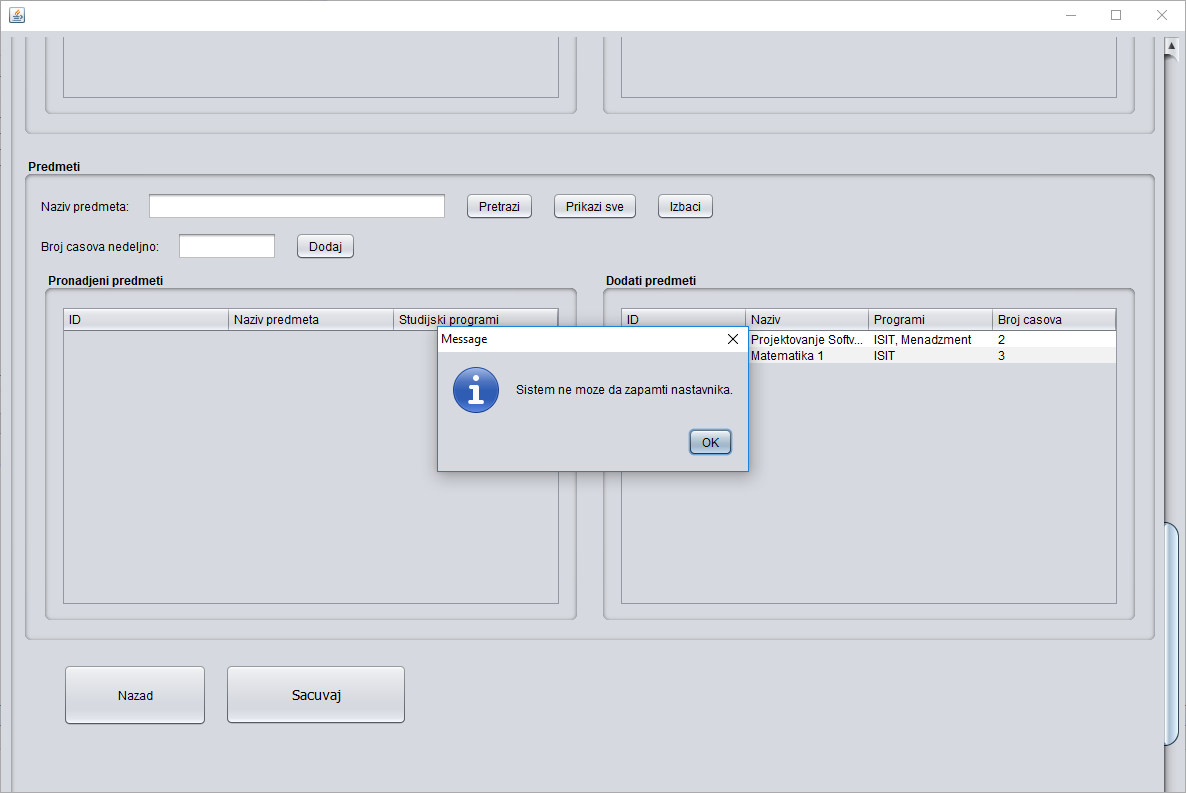
4.1 Уколико систем не може да нађе наставника он приказује запосленом поруку: “Систем не може да нађе наставника по задатој вредности”. Прекида се извршење сценариа. (ИА)



8.1 Уколико систем не може да нађе податке о наставнику он приказује запосленом поруку: “Систем не може да нађе податке о наставнику”. Прекида се извршење сценариа. (ИА)



13.1 Уколико систем не може да запамти податке о наставнику он приказује запосленом поруку: “Систем не може да запамти наставника”. (ИА)



#### 3.3.1.4 СК4: Случај коришћења – Претраживање наставника

**Назив СК**

Претраживање наставника

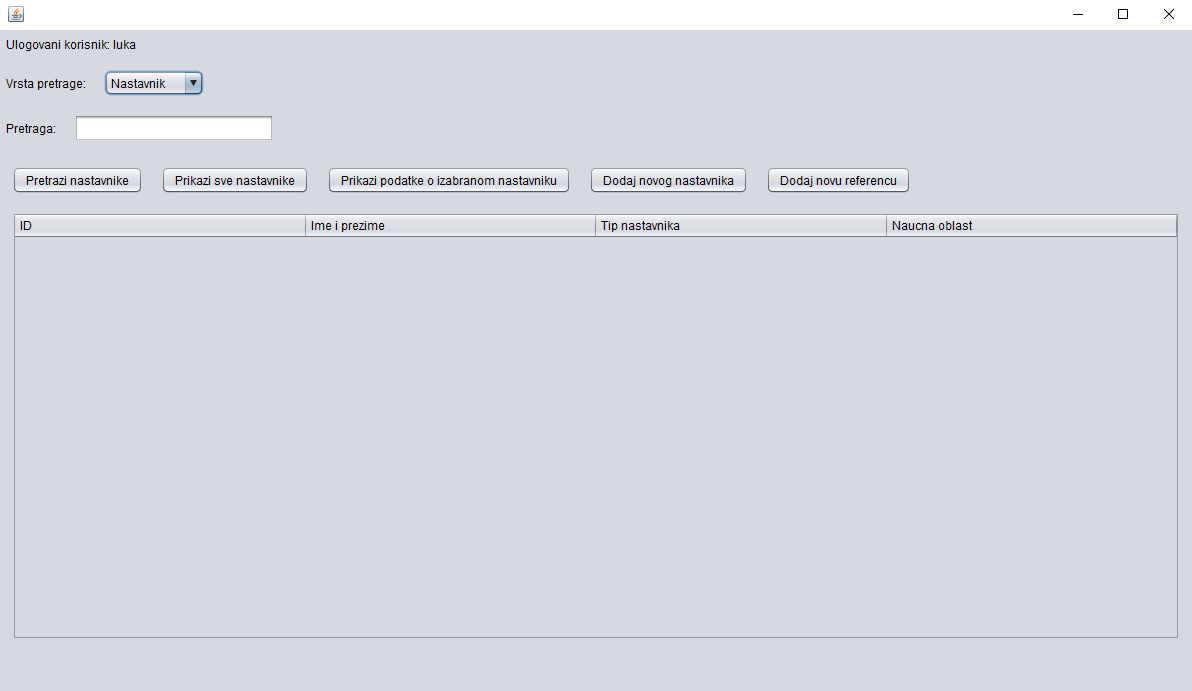
**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

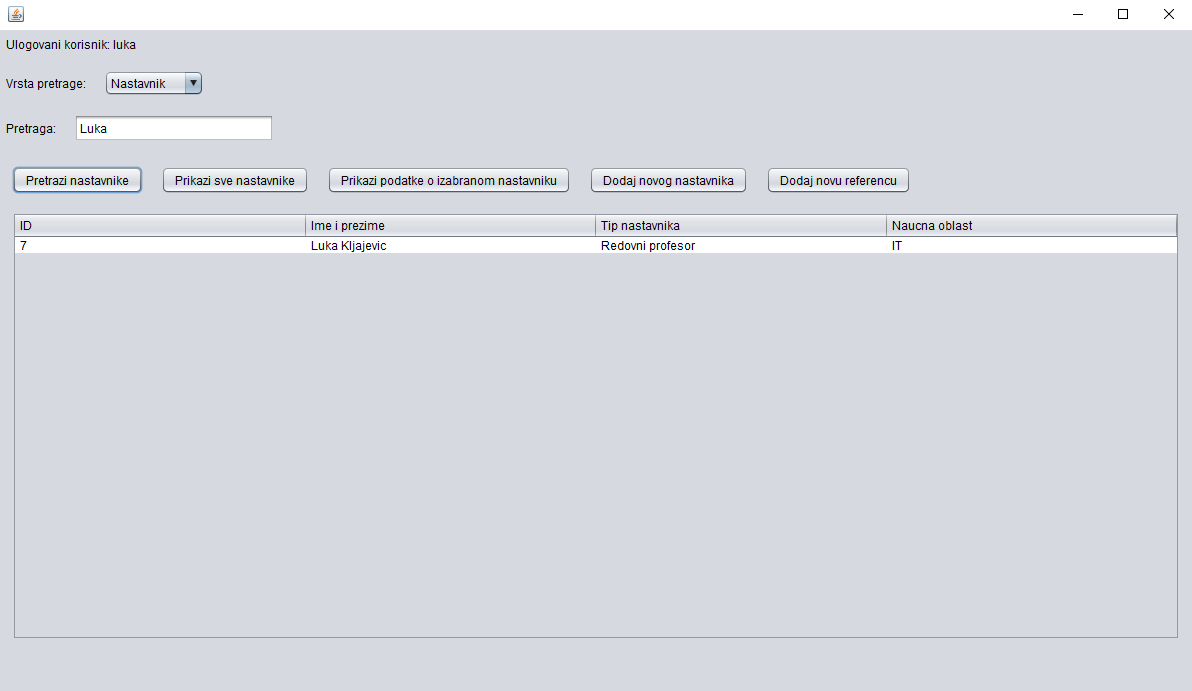
**Предуслов**: Систем је укључен и запослени je улогован под својом шифром. Систем приказује форму за рад са наставником.



Запослени уноси име и презиме наставника у поље за претрагу, а затим кликом на дугме Pretrazi nastavnike, позива систем да претражи наставнике по задатом критеријуму.

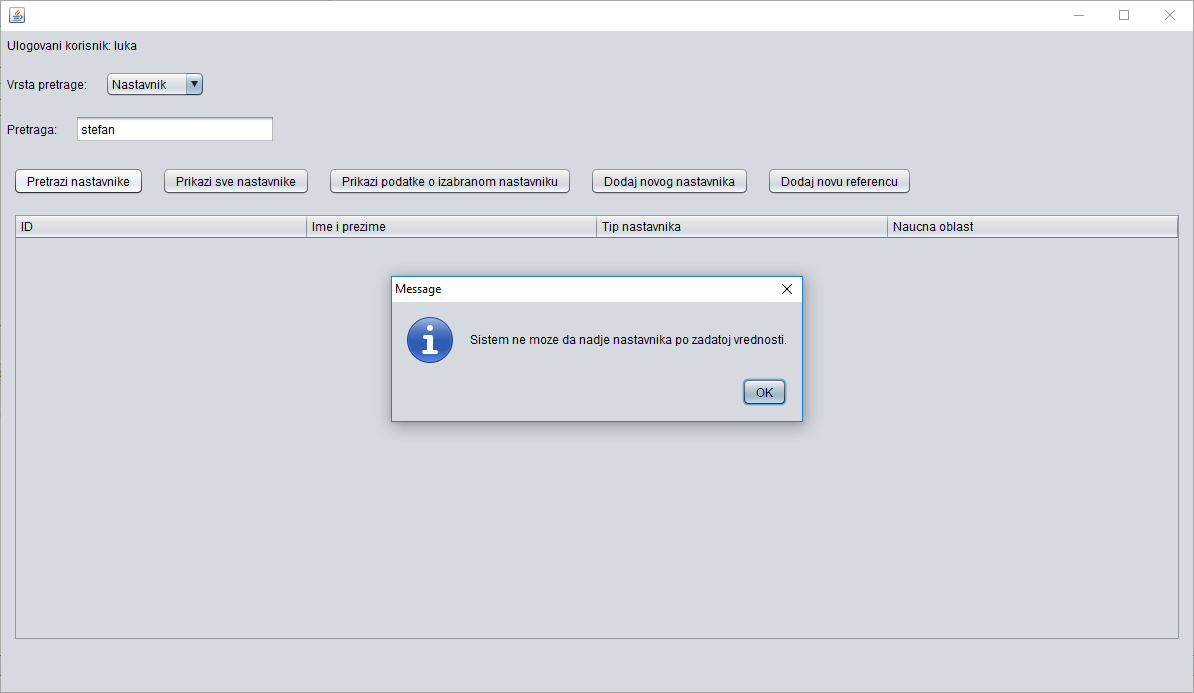
**Основни сценарио СК**

1. Запослени **уноси** вредностпо којој претражује наставника. (АПУСО)
2. Запослени **позива** систем да нађе наставника по задатој вредности. (АПСО)
3. Систем **тражи** наставника по задатој вредности. (СО)
4. Систем **приказује** запосленом податке о наставницима. (ИА)



**Алтернативна сценарија**

4.1 Уколико систем не може да нађе наставника он приказује запосленом поруку: “Систем не може да нађе наставника по задатој вредности”. (ИА)



#### 3.3.1.5 СК5: Случај коришћења – Брисање наставника

**Назив СК**

Брисање наставника

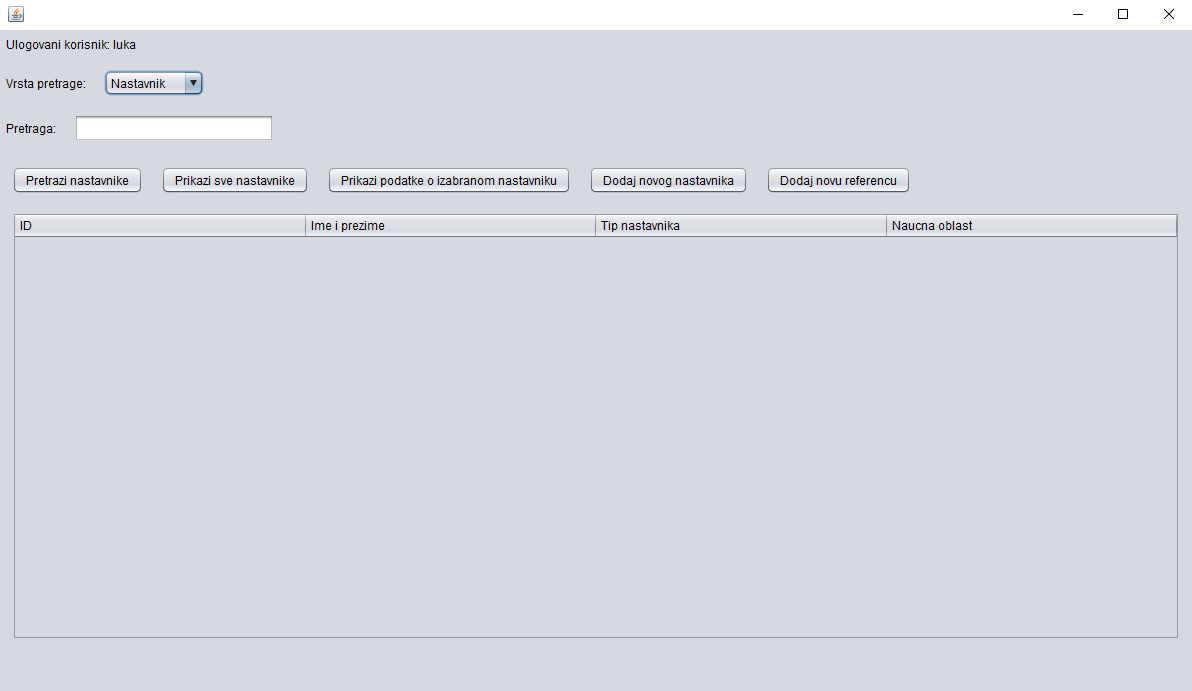
**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

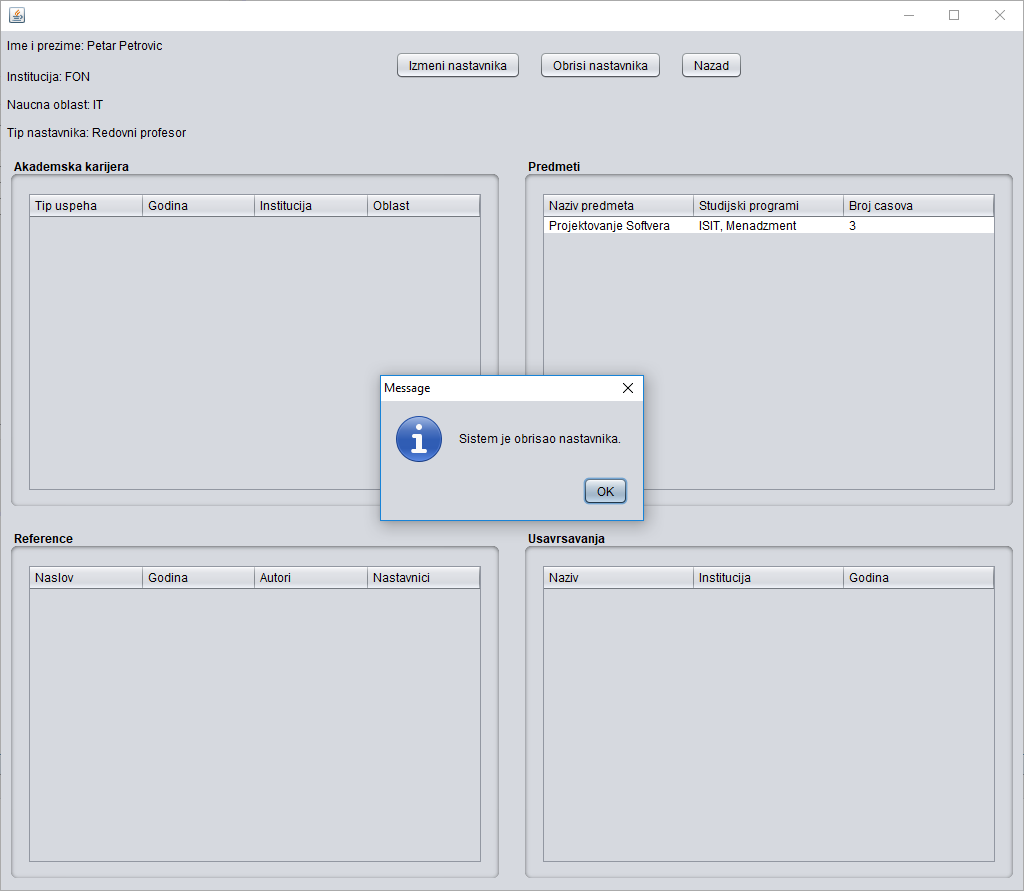
**Предуслов**: Систем је укључен и запослени je улогован под својом шифром. Систем приказује форму за рад са наставником.



Запослени уноси име наставника у поље за претрагу, затим бира једног од пронађених наставника и кликом на дугме Prikazi podatke o izabranom nastavniku, отвара се форма на којој се приказују сви подаци о датом наставнику. Кликом на дугме Obrisi nastavnika, позива систем да обрише све податке о том наставнику из базе.

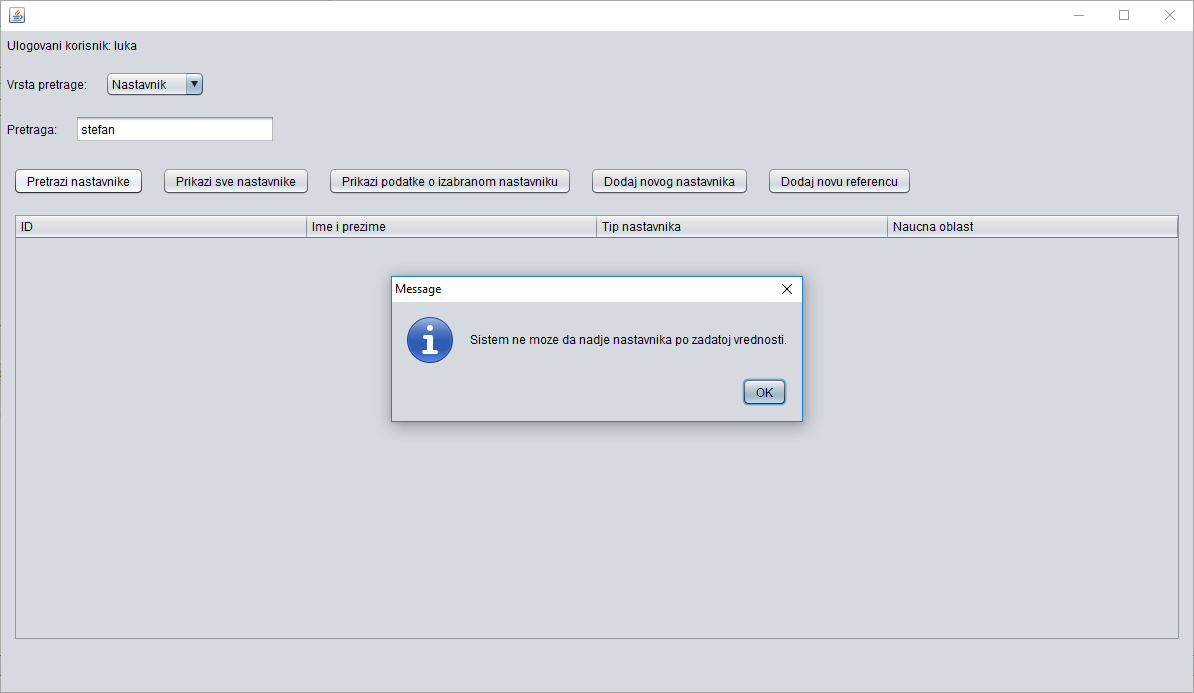
**Основни сценарио СК**

1. Запослени **уноси** вредностпо којој претражује наставника. (АПУСО)
2. Запослени **позива** систем да нађе наставника по задатој вредности. (АПСО)
3. Систем **тражи** наставника по задатој вредности. (СО)
4. Систем **приказује** запосленом наставнике. (ИА)
5. Запослени **бира** жељеног наставника из листе. (АПУСО)
6. Запослени **позива** систем да прикаже све податке о изабраном наставнику. (АПСО)
7. Систем **тражи** податке о изабраном наставнику. (СО)
8. Систем **приказује** податке о наставнику. (ИА)
9. Запослени **позива** систем да обрише наставника. (АПСО)
10. Систем **брише** наставника. (СО)
11. Систем **приказује** запосленом поруку: “Систем је обрисао наставника.” (ИА)

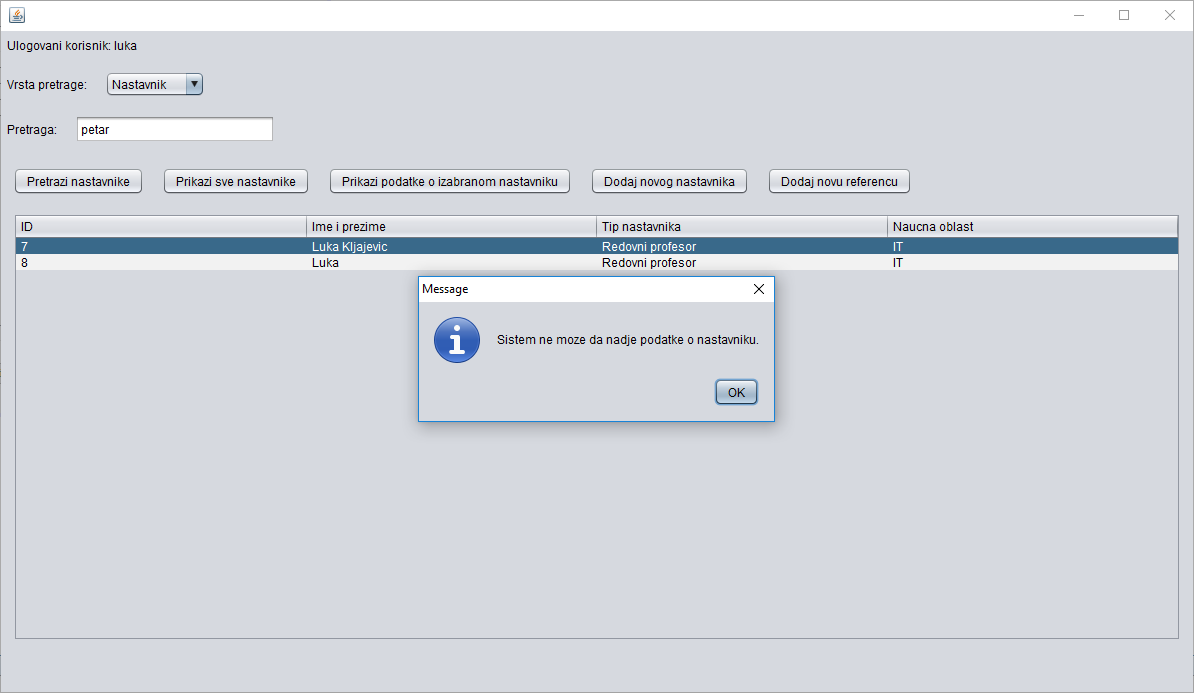


**Алтернативна сценарија**

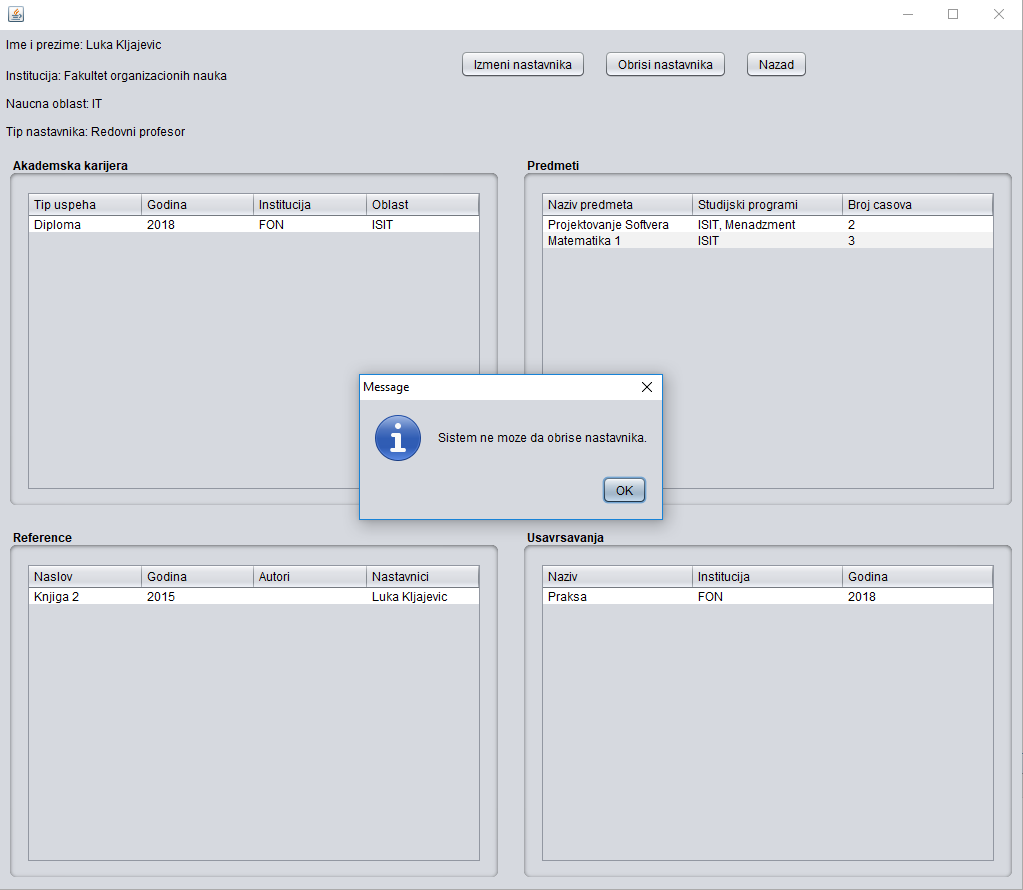
4.1 Уколико систем не може да нађе наставника он приказује запосленом поруку: “Систем не може да нађе наставника по задатој вредности”. Прекида се извршење сценариа. (ИА)



8.1 Уколико систем не може да нађе податке о наставнику он приказује запосленом поруку: “Систем не може да нађе податке о наставнику”. Прекида се извршење сценариа. (ИА)



11.1 Уколико систем не може да обрише наставника он приказује запосленом поруку: “Систем не може да обрише наставника”. (ИА)



#### 3.3.1.6 СК6: Случај коришћења – Креирање новог предмета

**Назив СК**

Креирање новог предмета

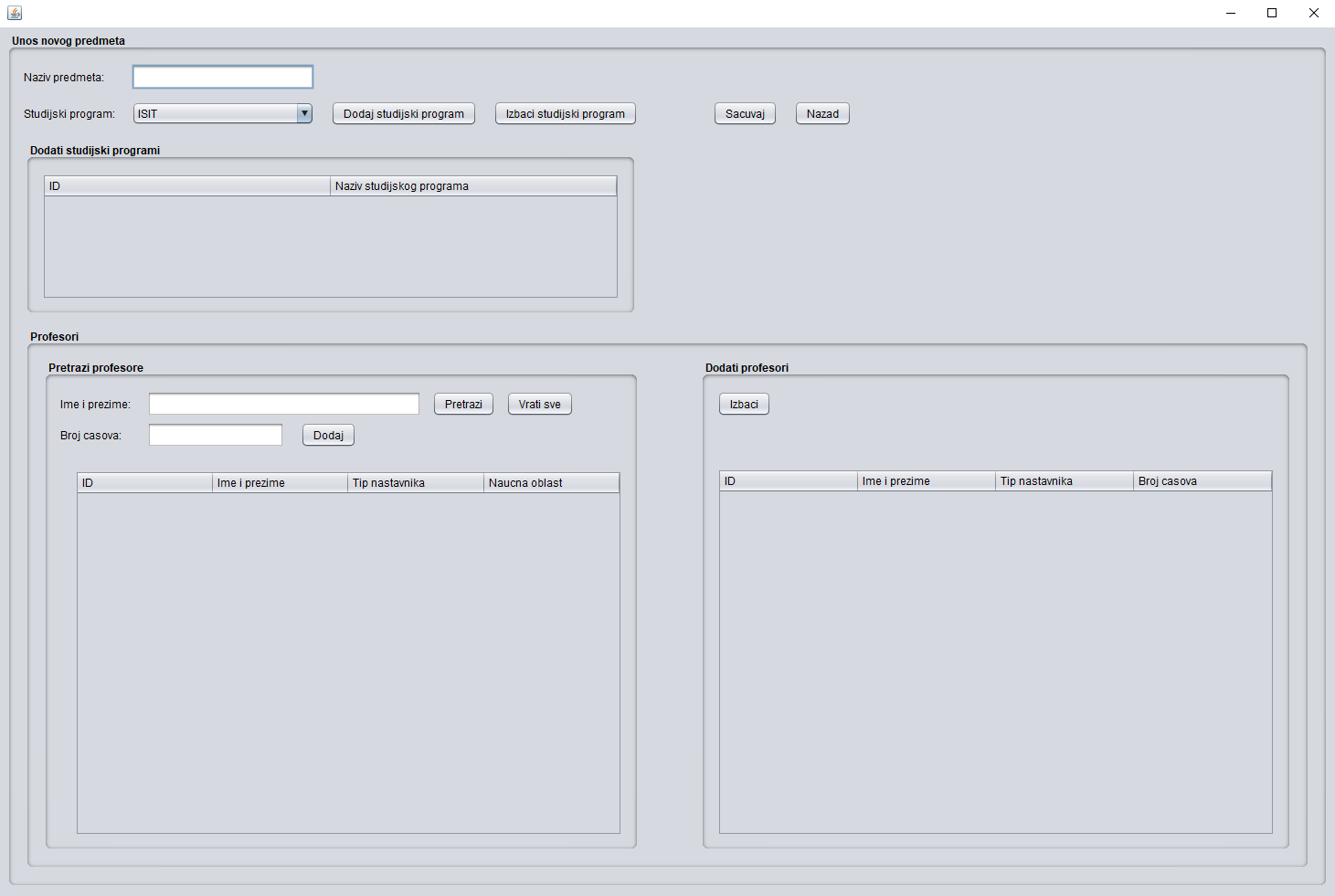
**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

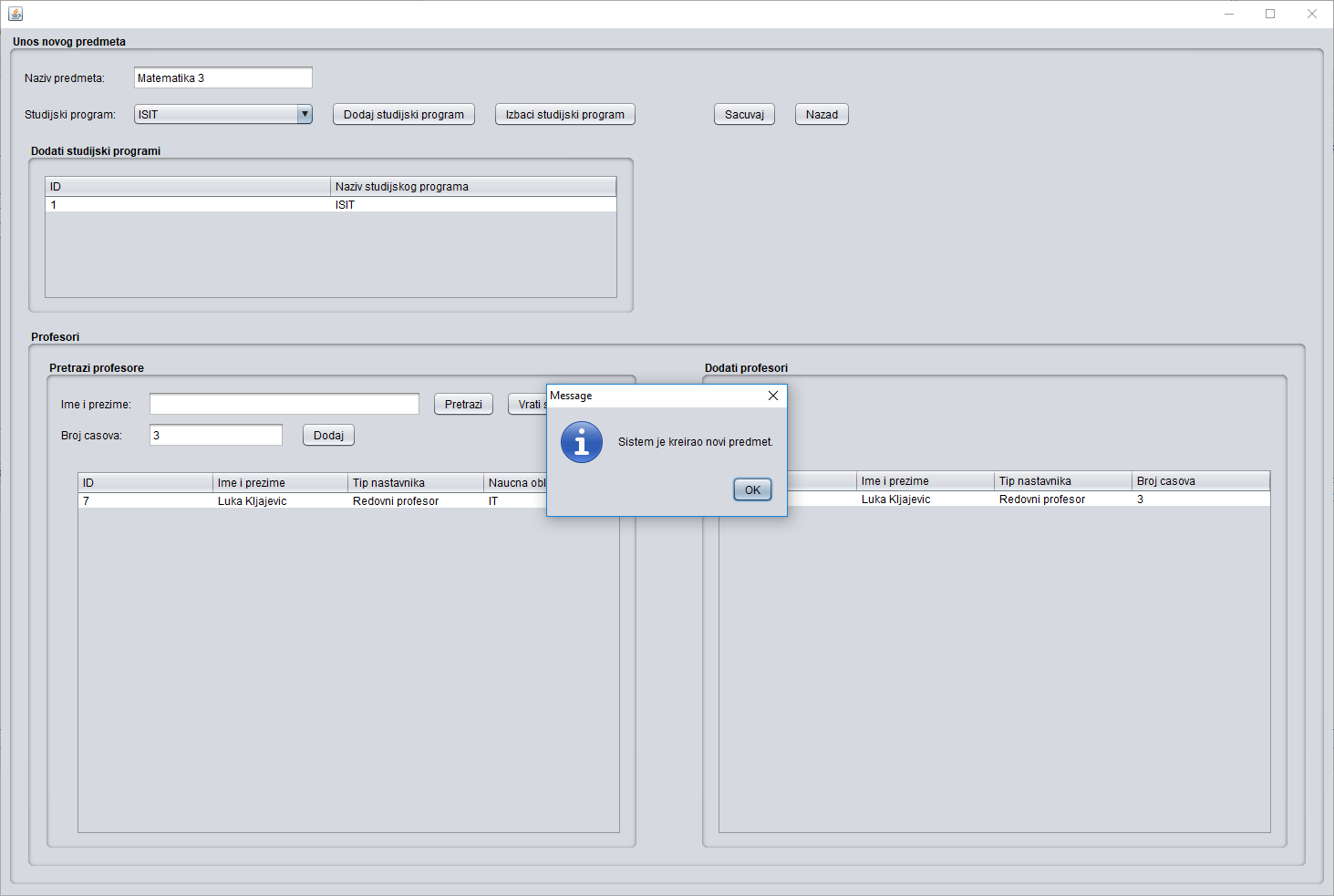
**Предуслов**: Систем је укључен, запослени je улогован под својом шифром и учитане су листе студијских програма и наставника. Систем приказује форму за рад са предметом.



Запослени уноси податке о предмету у одговарајућа поља и кликом на дугме Sacuvaj, систем креира нови предмет у бази.

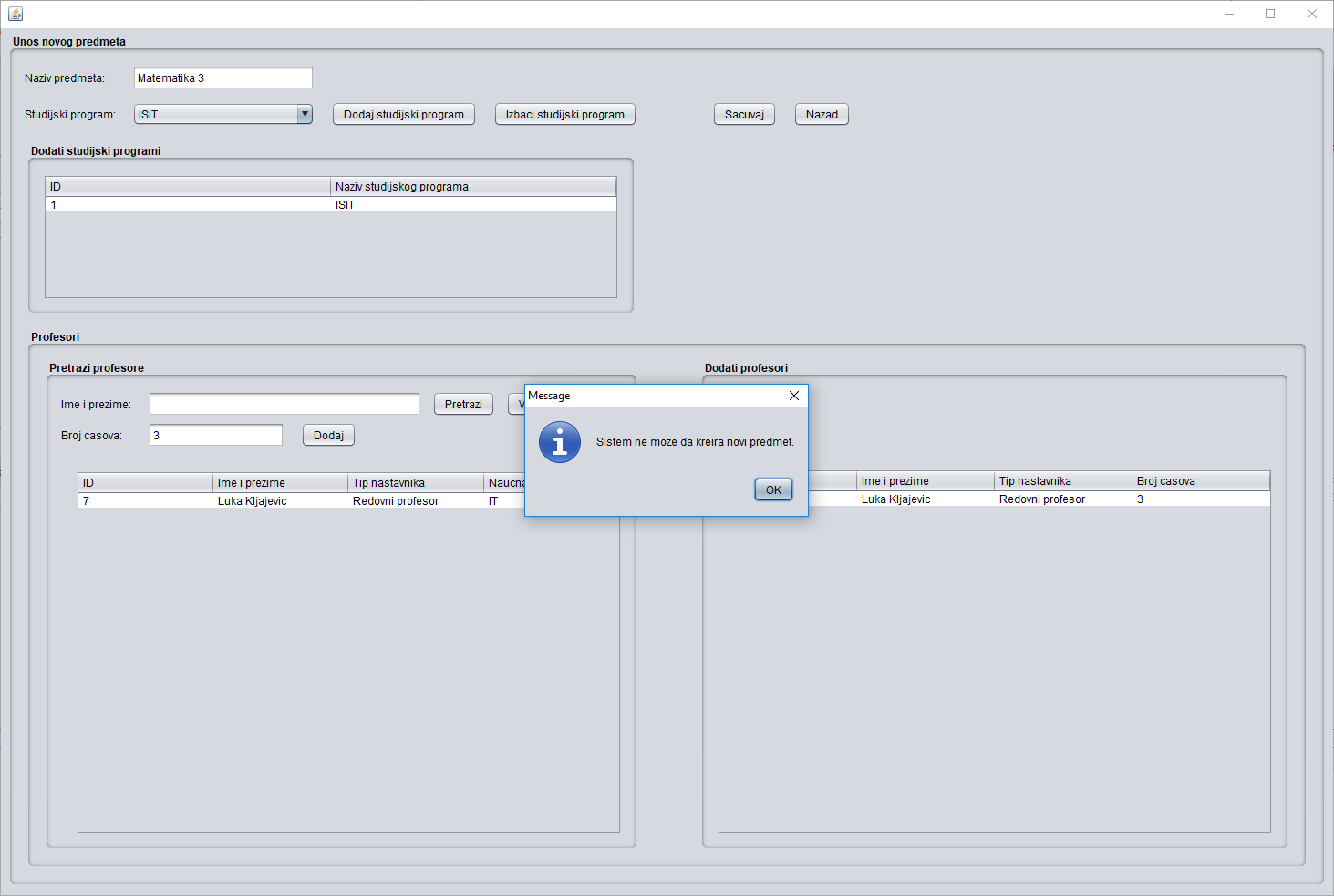
**Основни сценарио СК**

1. Запослени **уноси** податке у нови предмет. (АПУСО)
2. Запослени **контролише** да ли је коректно унео податке у нови предмет. (АНСО)
3. Запослени **позива** систем да креира нови предмет. (АПСО)
4. Систем **креира** нови предмет. (СО)
5. Систем **приказује** запосленом нови предмет и поруку: “Систем је креирао нови предмет“. (ИА)



**Алтернативна сценарија**

5.1 Уколико систем не може да креира нови предмет он приказује запосленом поруку: “Систем не може да креира нови предмет ”. Прекида се извршење сценариа. (ИА)



#### 3.3.1.7 СК7: Случај коришћења – Промена предмета

**Назив СК**

Промена предмета

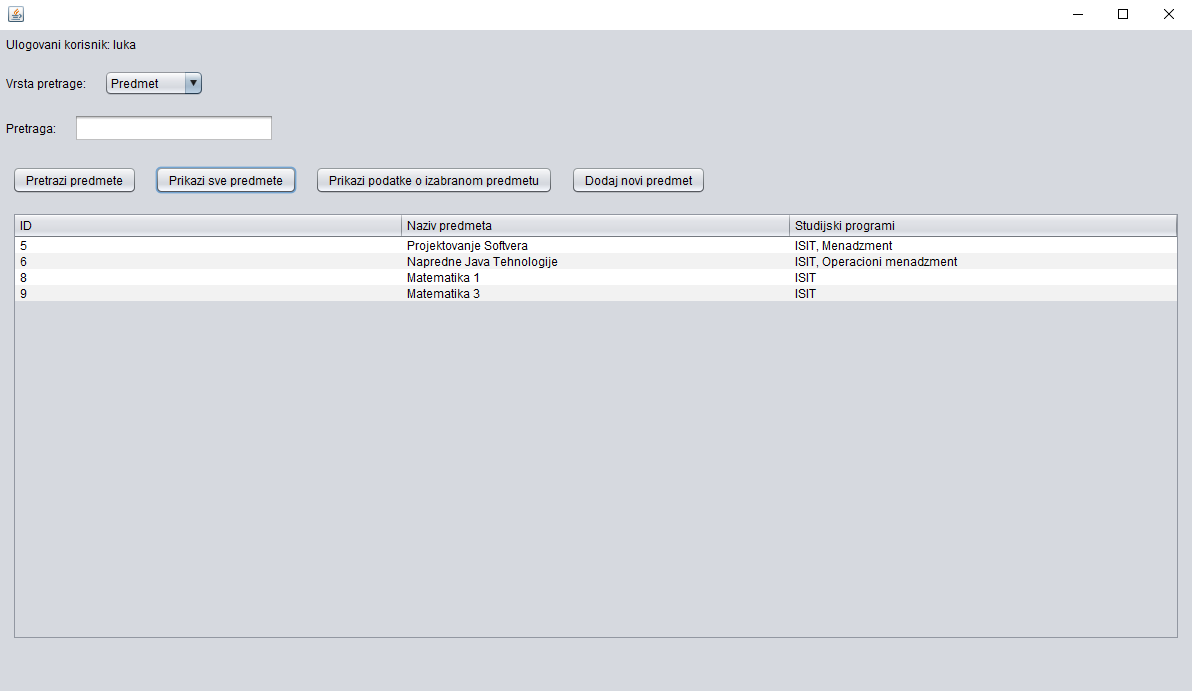
**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

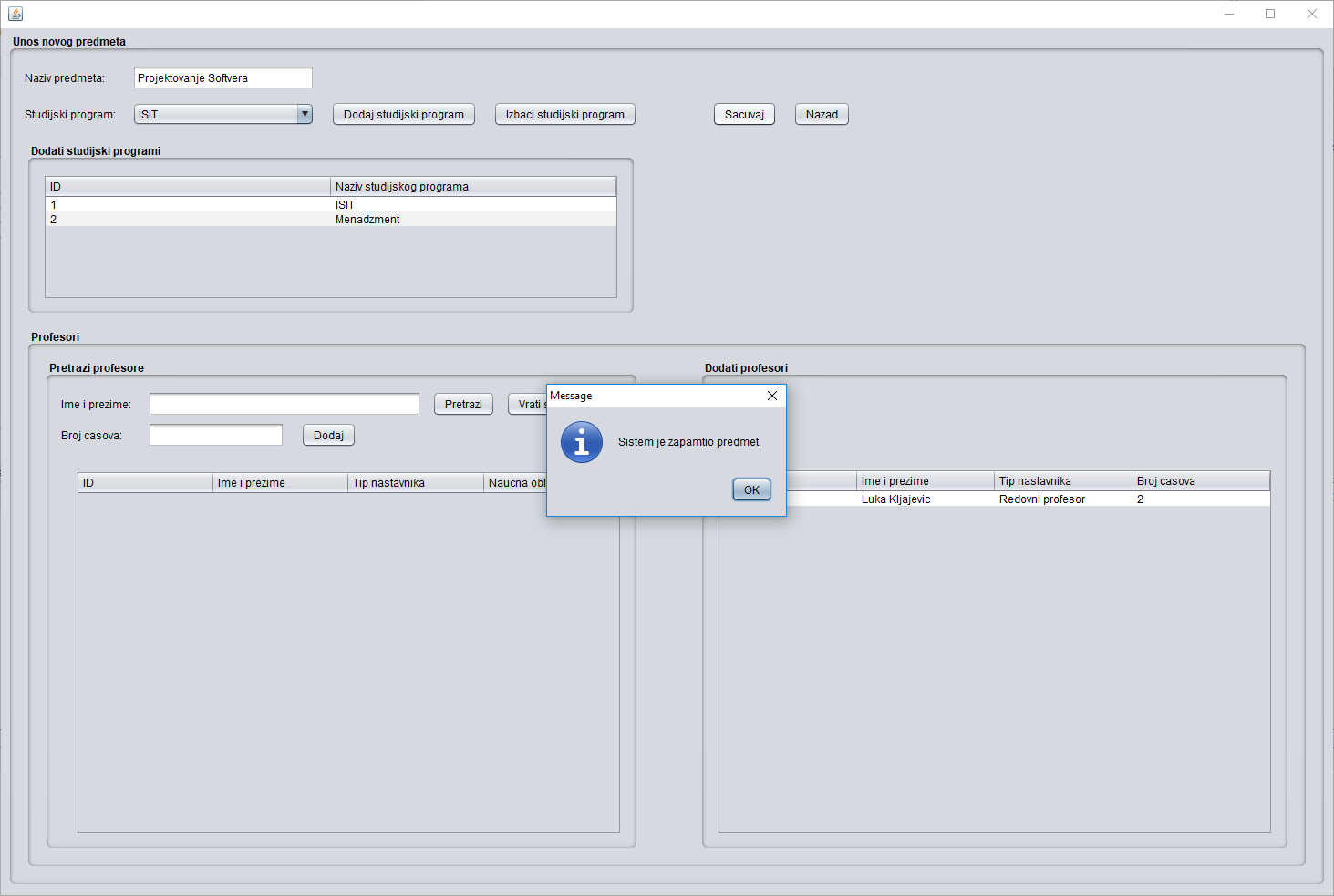
**Предуслов**: Систем је укључен, запослени je улогован под својом шифром и учитане су листе студијских програма и наставника. Систем приказује форму за рад са предметом.



Запослени уноси назив предмета у поље за претрагу, затим бира једног од пронађених предмета и кликом на дугме Prikazi podatke o izabranom predmetu, отвара се форма на којој се приказују сви подаци о датом предмету. Кликом на дугме Izmeni predmet, отвара се нова форма са постојећим подацима о предмету које запослени може да мења. Кликом на дугме Sacuvaj, систем бележи све промене.

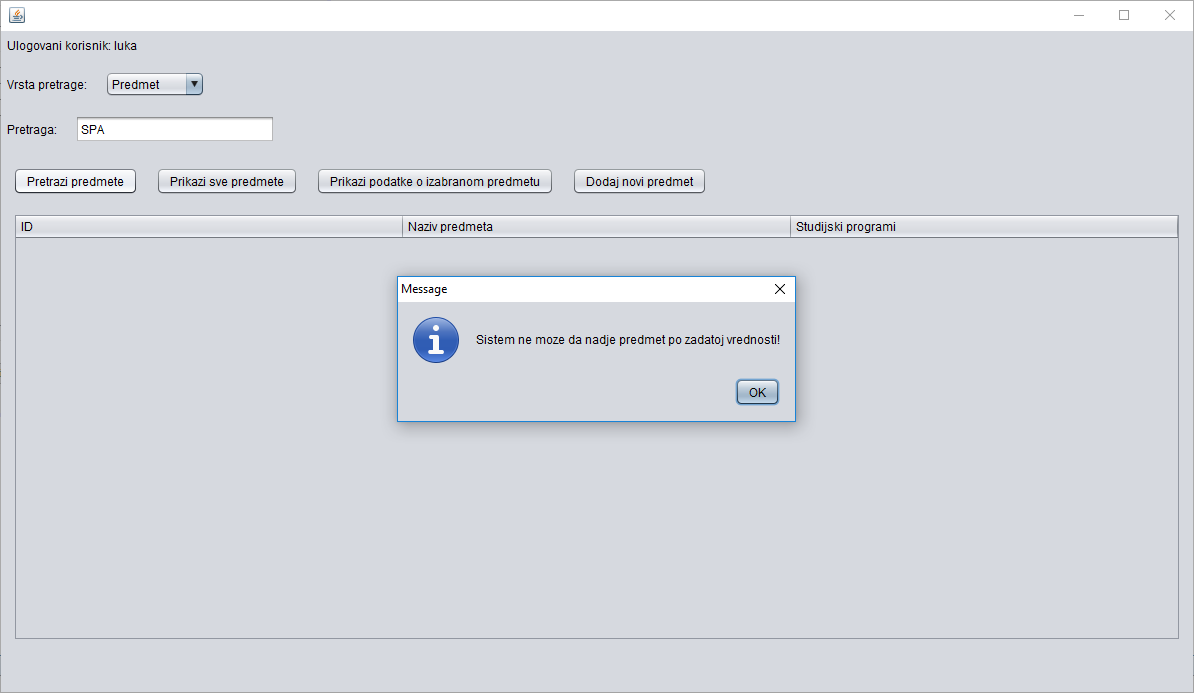
**Основни сценарио СК**

1. Запослени **уноси** вредностпо којој претражује предмет. (АПУСО)
2. Запослени **позива** систем да нађе предмет по задатој вредности. (АПСО)
3. Систем **тражи** предмет по задатој вредности. (СО)
4. Систем **приказује** запосленом предмете и поруку: “Систем је нашао предмете по задатој вредности”. (ИА)
5. Запослени **бира** жељени предмет из листе. (АПУСО)
6. Запослени **позива** систем да прикаже све податке о изабраном предмету. (АПСО)
7. Систем **тражи** податке о изабраном предмету. (СО)
8. Систем **приказује** податке о предмету. (ИА)
9. Запослени **уноси** **(мења)** податке о предмету. (АПУСО)
10. Запослени **контролише** да ли је коректно унео податке о предмету. (АНСО)
11. Запослени **позива** систем да запамти податке о предмету. (АПСО)
12. Систем **памти** податке о предмету. (СО)
13. Систем **приказује** запосленом поруку: “Систем је запамтио предмет.” (ИА)

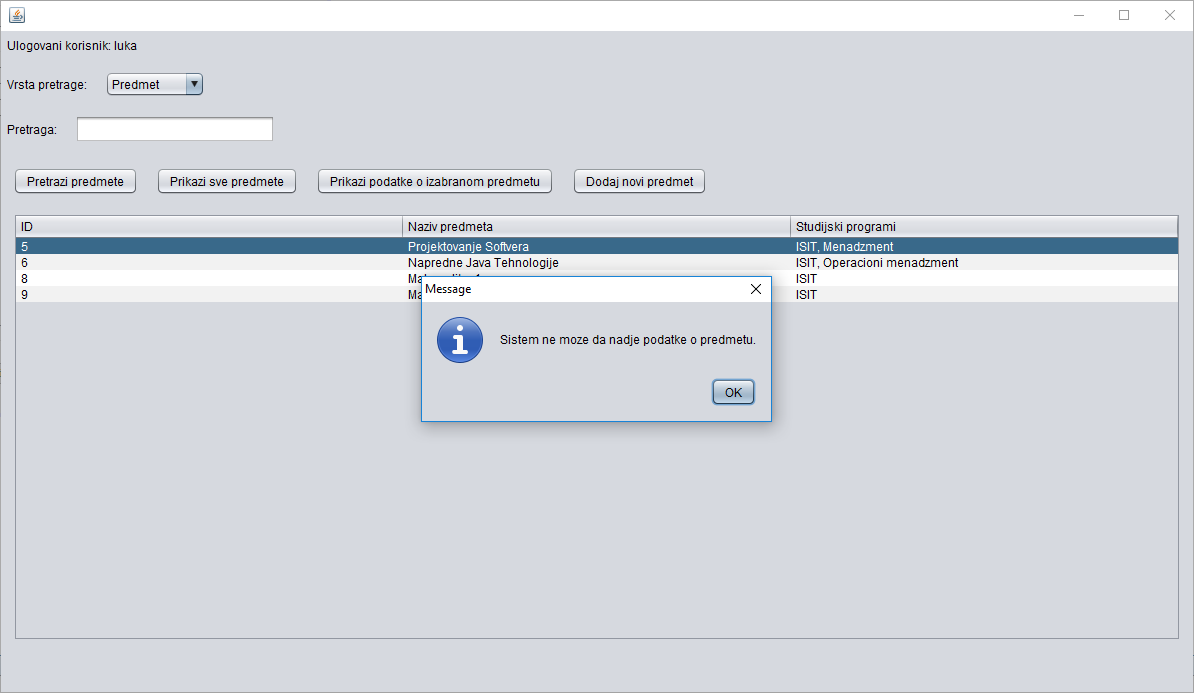


**Алтернативна сценарија**

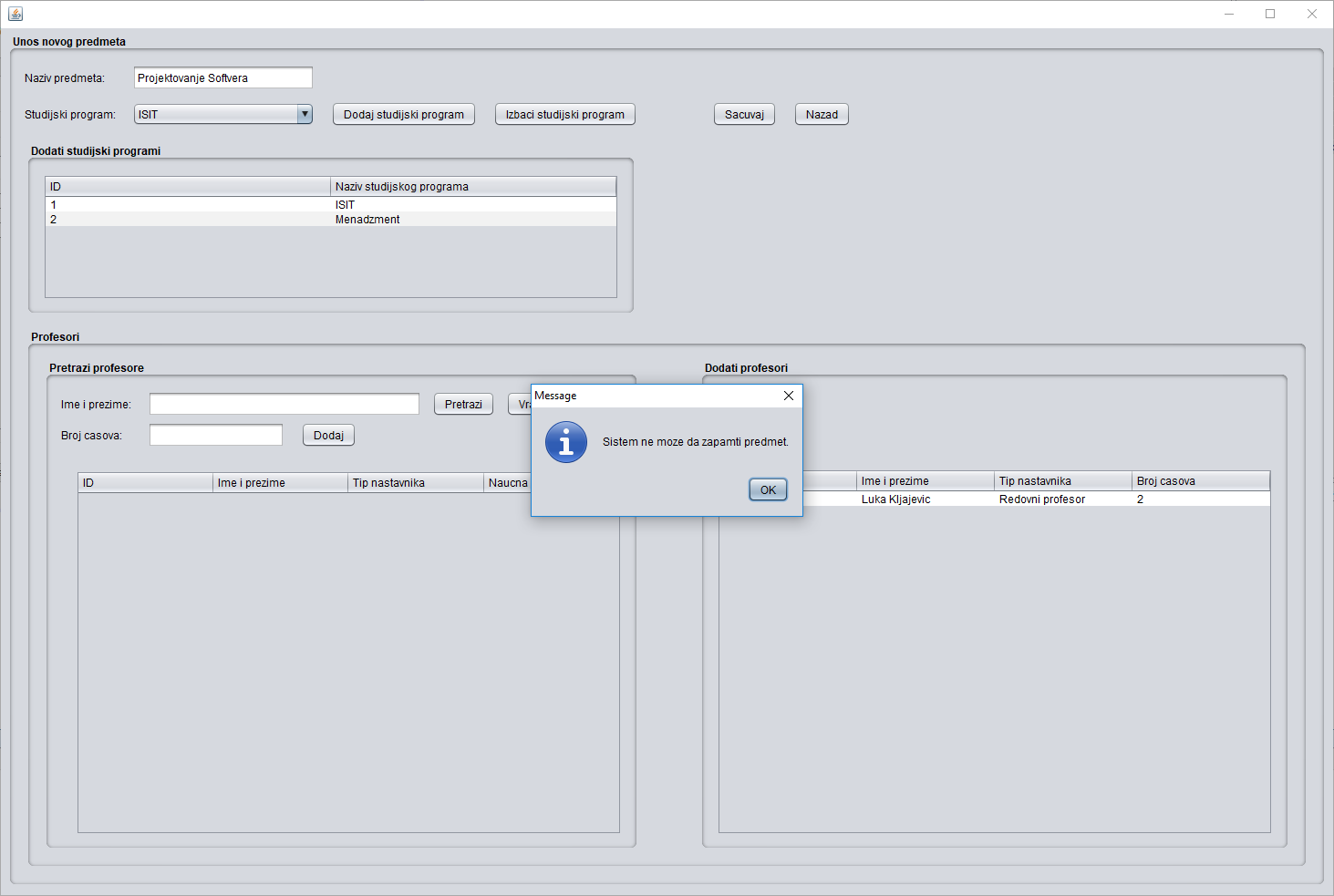
4.1 Уколико систем не може да нађе предмет он приказује запосленом поруку: “Систем не може да нађе предмет по задатој вредности”. Прекида се извршење сценариа. (ИА)



8.1 Уколико систем не може да нађе податке о предмету он приказује запосленом поруку: “Систем не може да нађе податке о предмету”. Прекида се извршење сценариа. (ИА)



13.1 Уколико систем не може да запамти податке о предмету он приказује запосленом поруку: “Систем не може да запамти предмет ”. (ИА)



#### 3.3.1.8 СК8: Случај коришћења – Претраживање предмета

**Назив СК**

Претраживање предмета

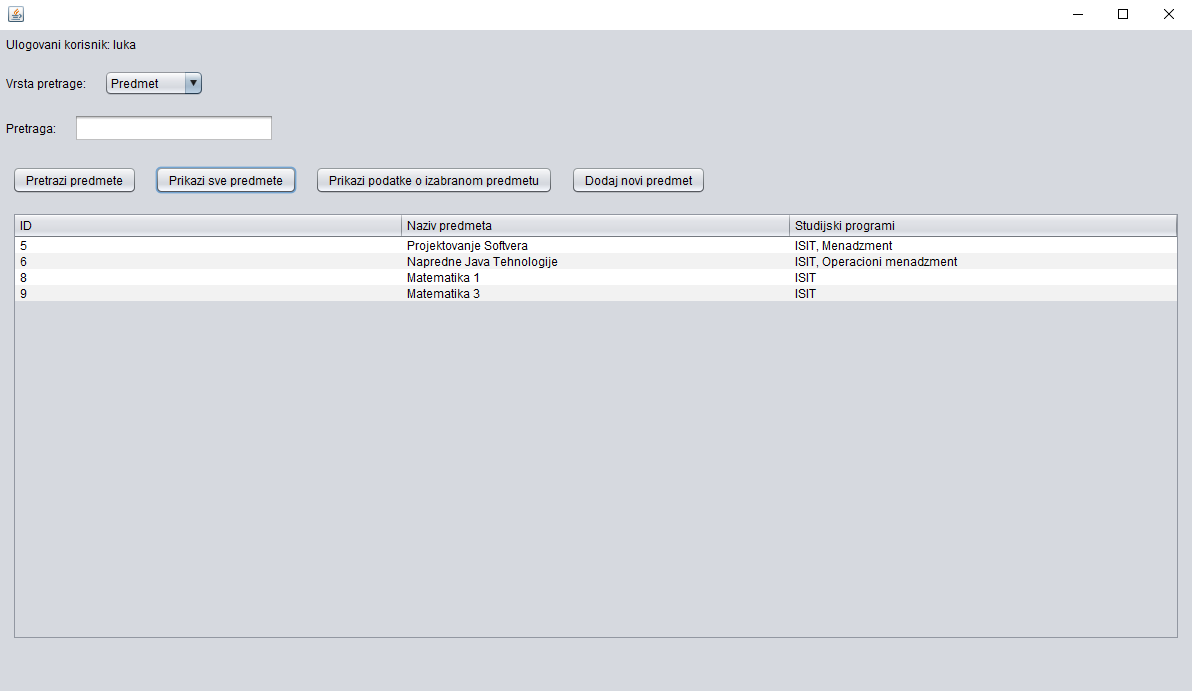
**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

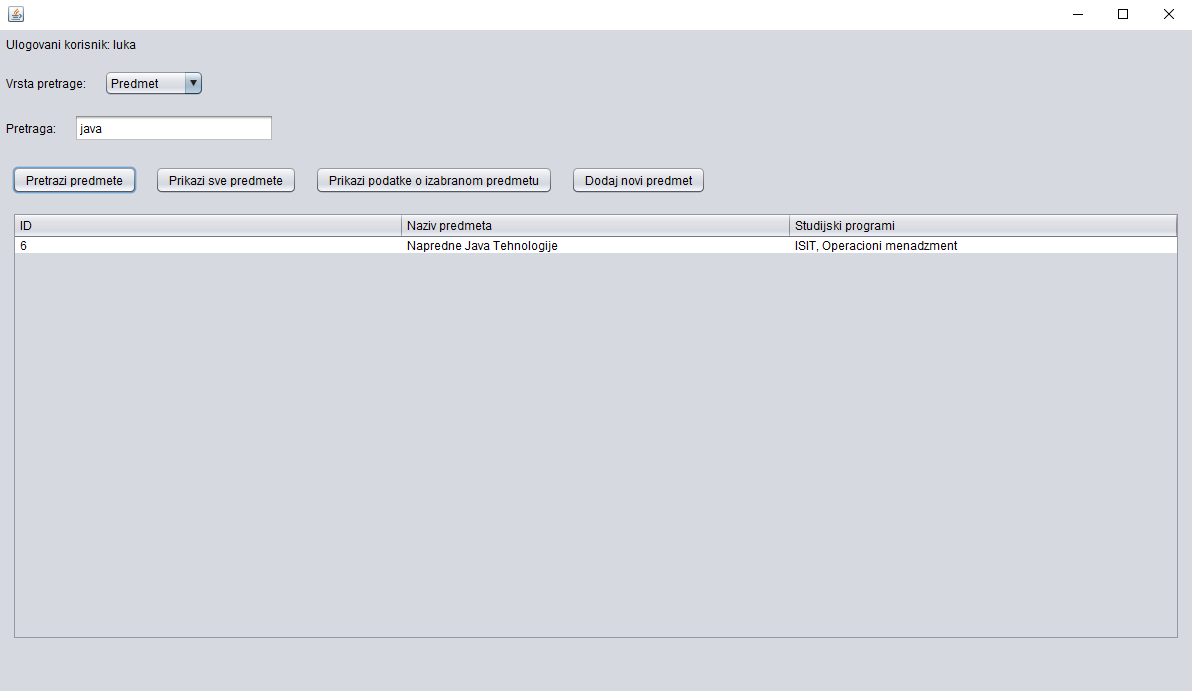
**Предуслов**: Систем је укључен и запослени je улогован под својом шифром. Систем приказује форму за рад са предметом.



Запослени уноси назив предмета у поље за претрагу, а затим кликом на дугме Pretrazi predmete, позива систем да претражи предмете по задатом критеријуму.

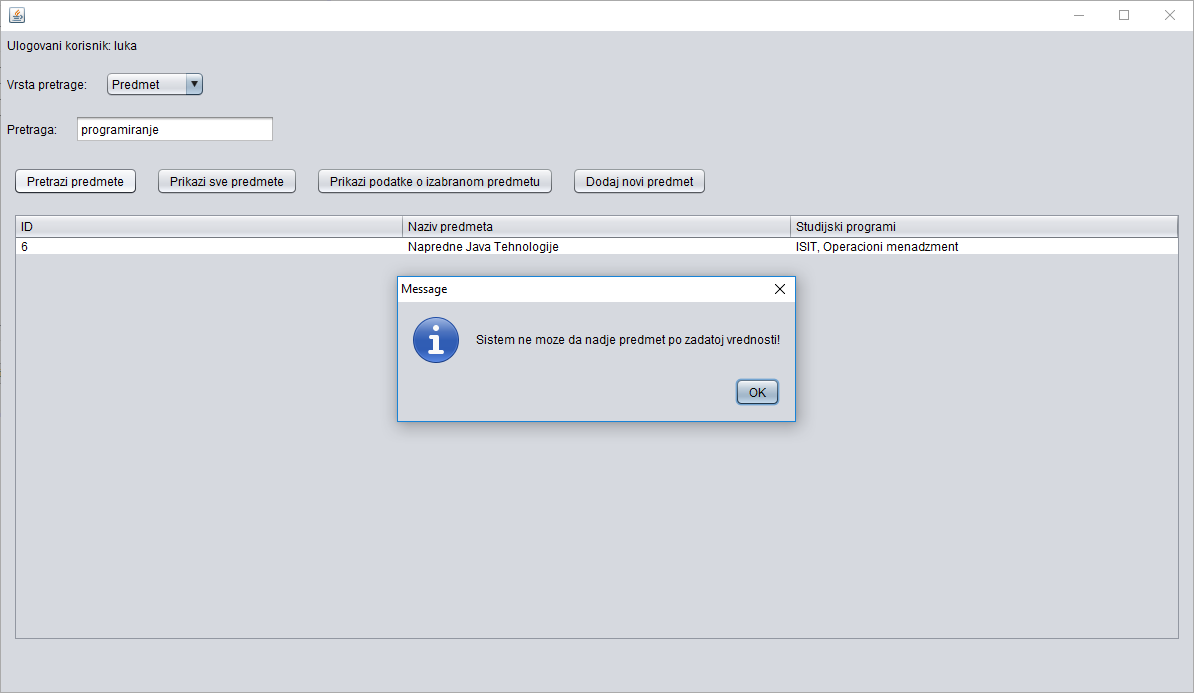
**Основни сценарио СК**

1. Запослени **уноси** вредностпо којој претражује предмет. (АПУСО)
2. Запослени **позива** систем да нађе предмет по задатој вредности. (АПСО)
3. Систем **тражи** предмет по задатој вредности. (СО)
4. Систем **приказује** запосленом податке о предметима и поруку: “Систем је нашао предмете по задатој вредности”. (ИА)



**Алтернативна сценарија**

4.1 Уколико систем не може да нађе предмет он приказује запосленом поруку: “Систем не може да нађе предмет по задатој вредности”. (ИА)



#### 3.3.1.9 СК9: Случај коришћења – Брисање предмета

**Назив СК**

Брисање предмета

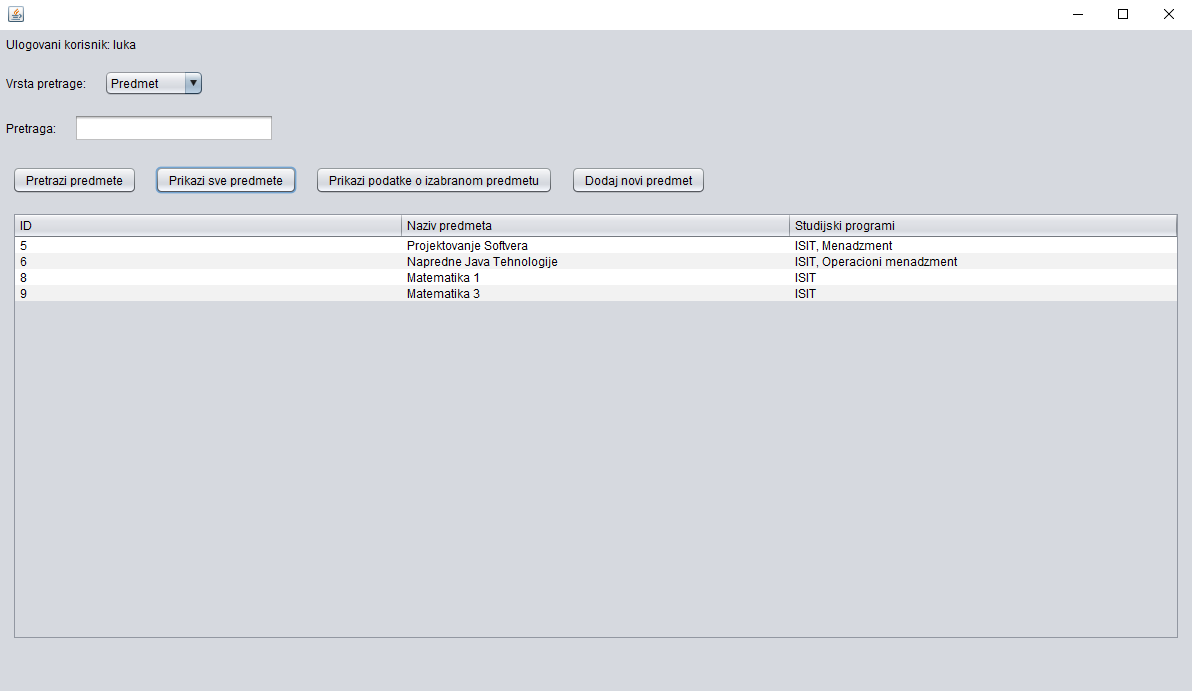
**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

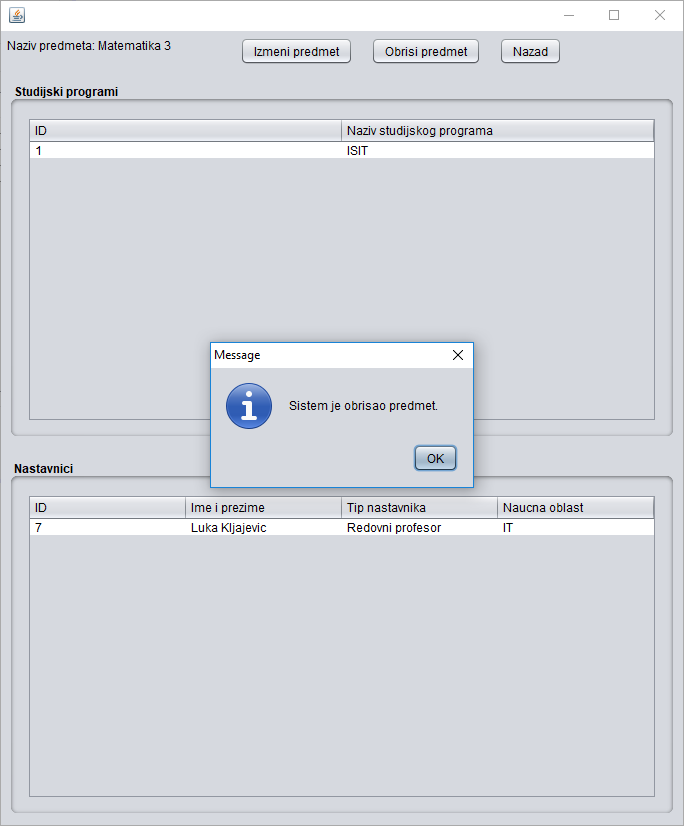
**Предуслов**: Систем је укључен и запослени je улогован под својом шифром. Систем приказује форму за рад са предметом.



Запослени уноси назив предмета у поље за претрагу, затим бира једног од пронађених предмета и кликом на дугме Prikazi podatke o izabranom predmetu, отвара се форма на којој се приказују сви подаци о датом предмету. Кликом на дугме Obrisi predmet, позива систем да обрише све податке о том предмету из базе.

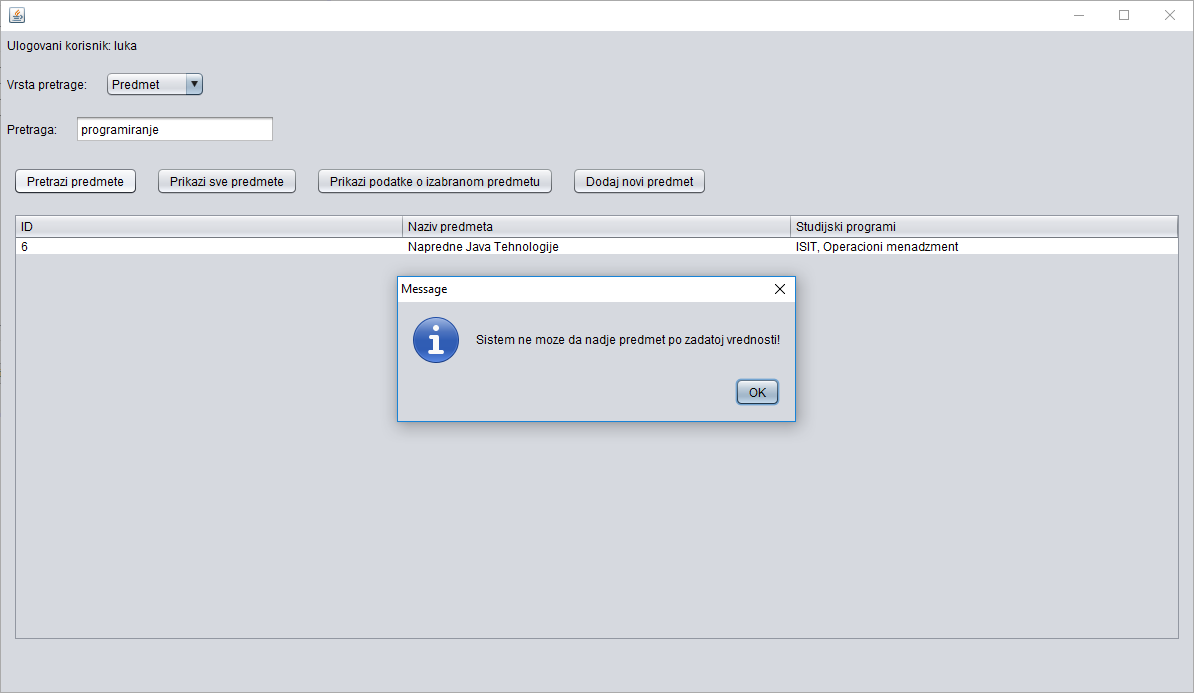
**Основни сценарио СК**

1. Запослени **уноси** вредностпо којој претражује предмет. (АПУСО)
2. Запослени **позива** систем да нађе предмет по задатој вредности. (АПСО)
3. Систем **тражи** предмет по задатој вредности. (СО)
4. Систем **приказује** запосленом предмете и поруку: “Систем је нашао предмете по задатој вредности”. (ИА)
5. Запослени **бира** жељени предмет из листе. (АПУСО)
6. Запослени **позива** систем да прикаже све податке о изабраном предмету. (АПСО)
7. Систем **тражи** податке о изабраном предмету. (СО)
8. Систем **приказује** податке о предмету. (ИА)
9. Запослени **позива** систем да обрише предмет. (АПСО)
10. Систем **брише** предмет. (СО)
11. Систем **приказује** запосленом поруку: “Систем је обрисао предмет.” (ИА)

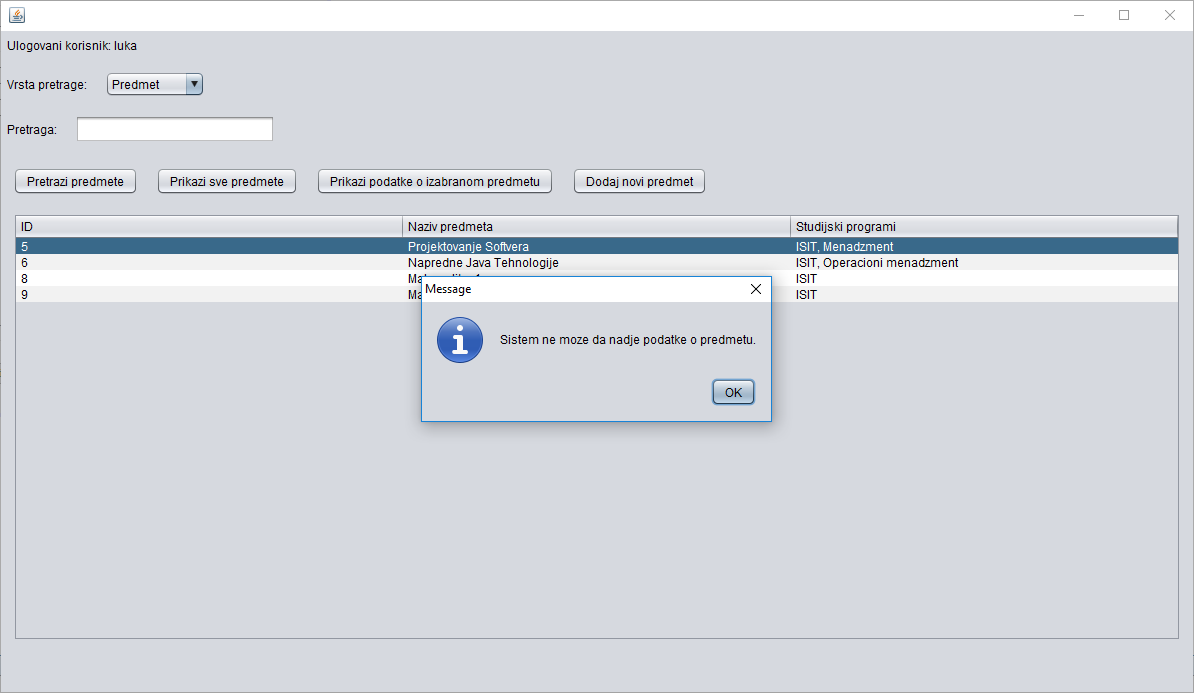


**Алтернативна сценарија**

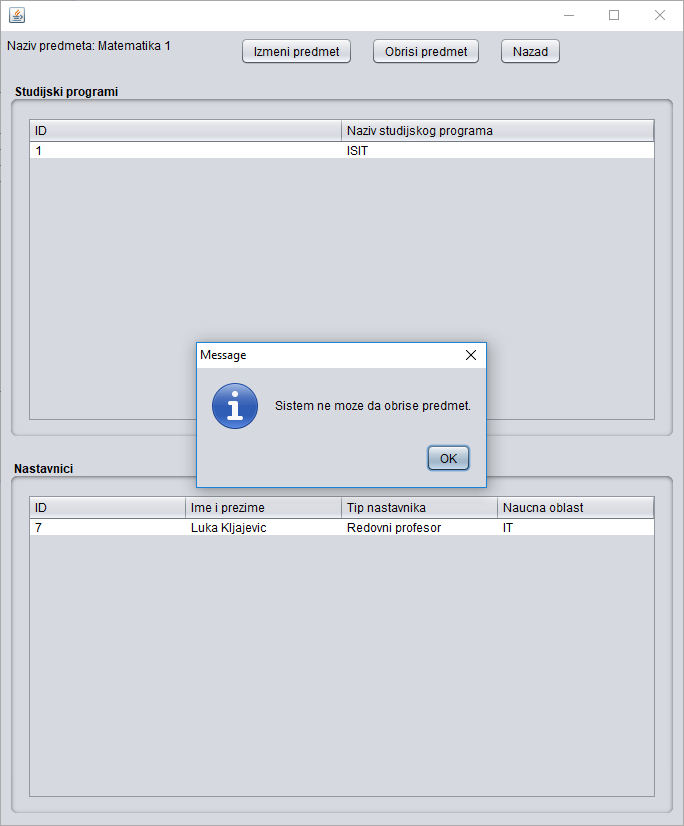
4.1 Уколико систем не може да нађе предмет он приказује запосленом поруку: “Систем не може да нађе предмет по задатој вредности”. Прекида се извршење сценариа. (ИА)



8.1 Уколико систем не може да нађе податке о предмету он приказује запосленом поруку: “Систем не може да нађе податке о предмету”. Прекида се извршење сценариа. (ИА)



11.1 Уколико систем не може да обрише предмет он приказује запосленом поруку: “Систем не може да обрише предмет ”. (ИА)



#### 3.3.1.10 СК10: Случај коришћења – Креирање нове референце

**Назив СК**

Креирање нове референце

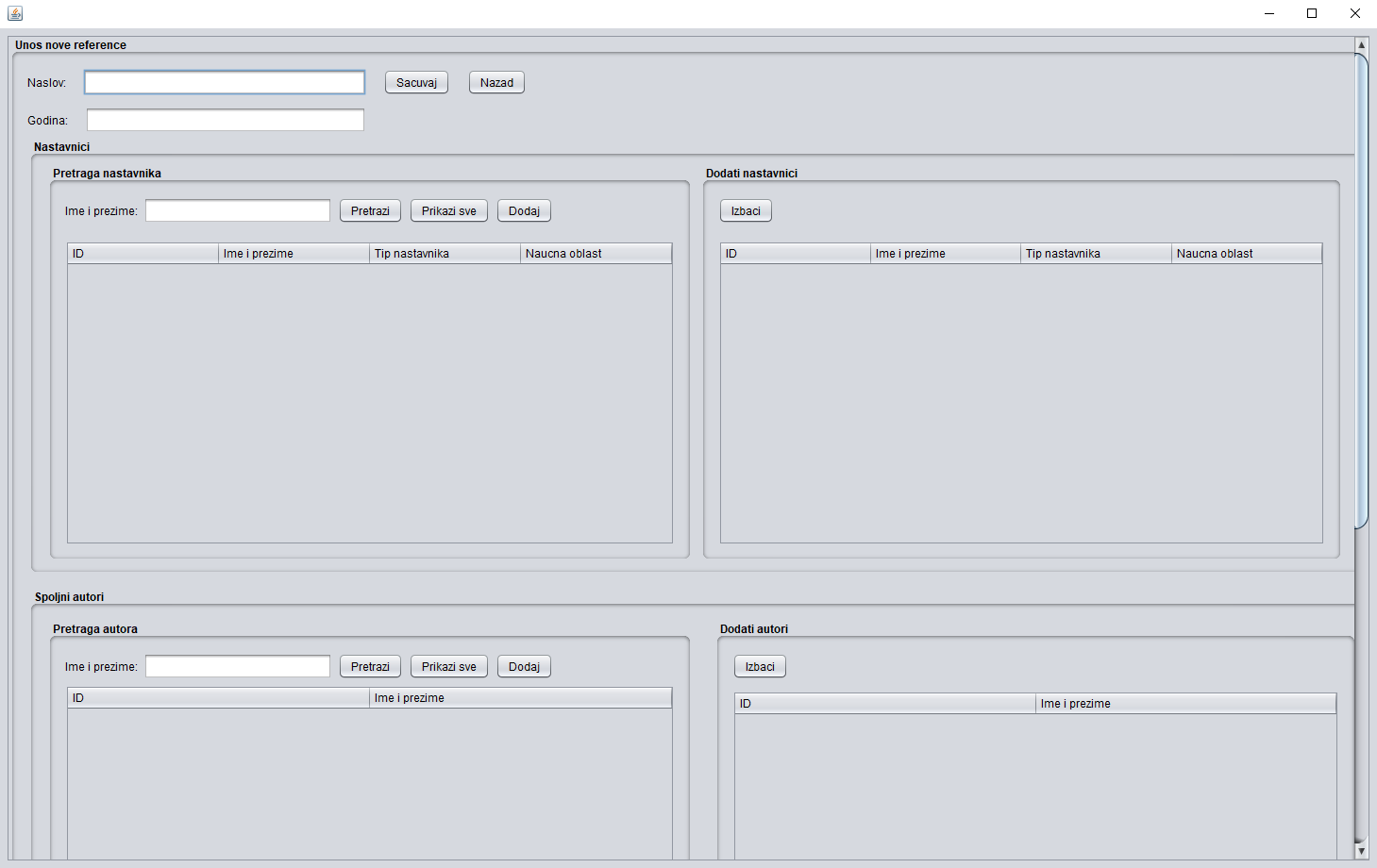
**Актори СК**

Запослени

**Учесници СК**

Запослени и систем (програм)

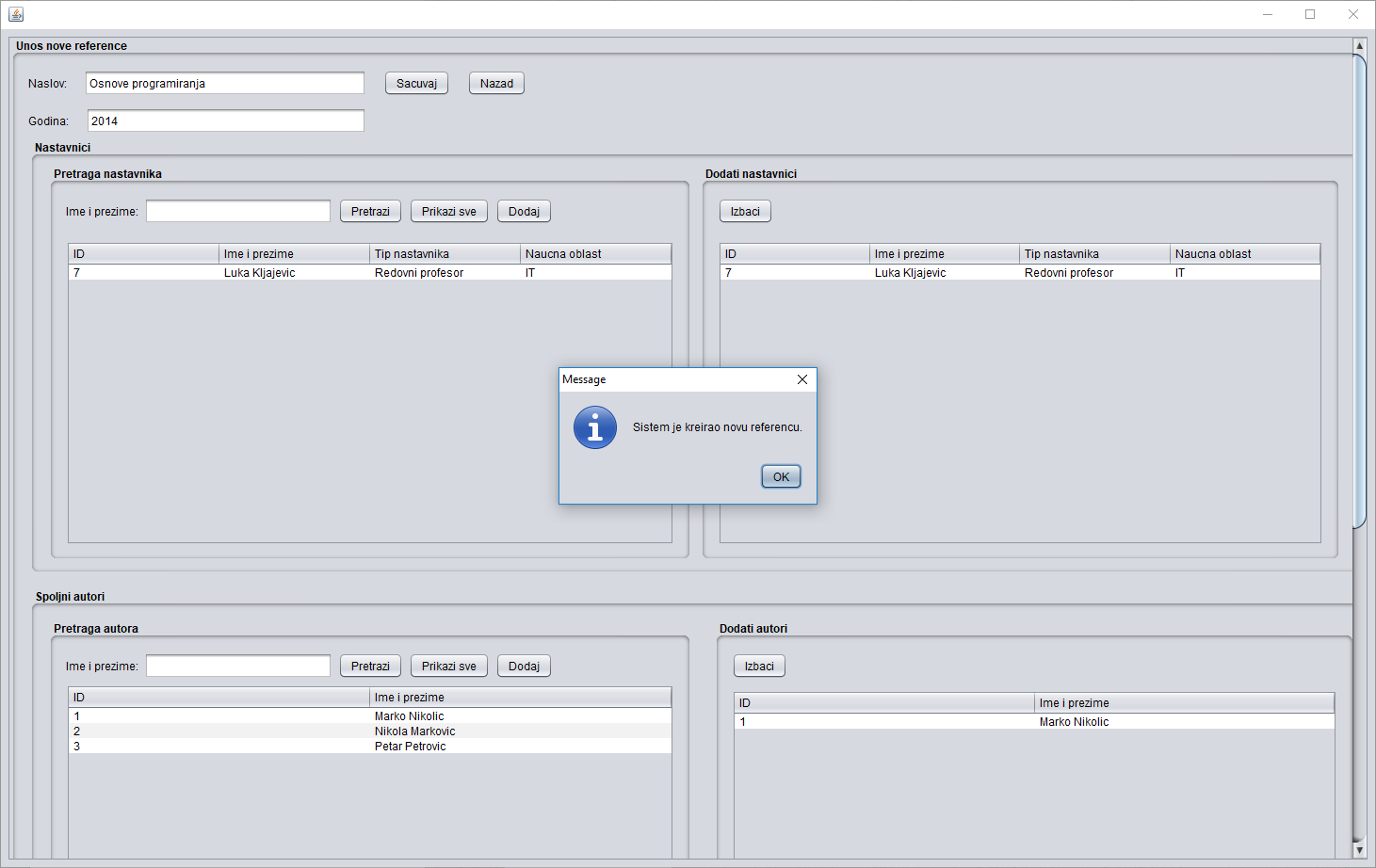
**Предуслов**: Систем је укључен, запослени je улогован под својом шифром и учитане су листе наставника и спољних аутора. Систем приказује форму за рад са референцом.



Запослени уноси податке о референци у одговарајућа поља и кликом на дугме Sacuvaj, систем креира нову референцу у бази.

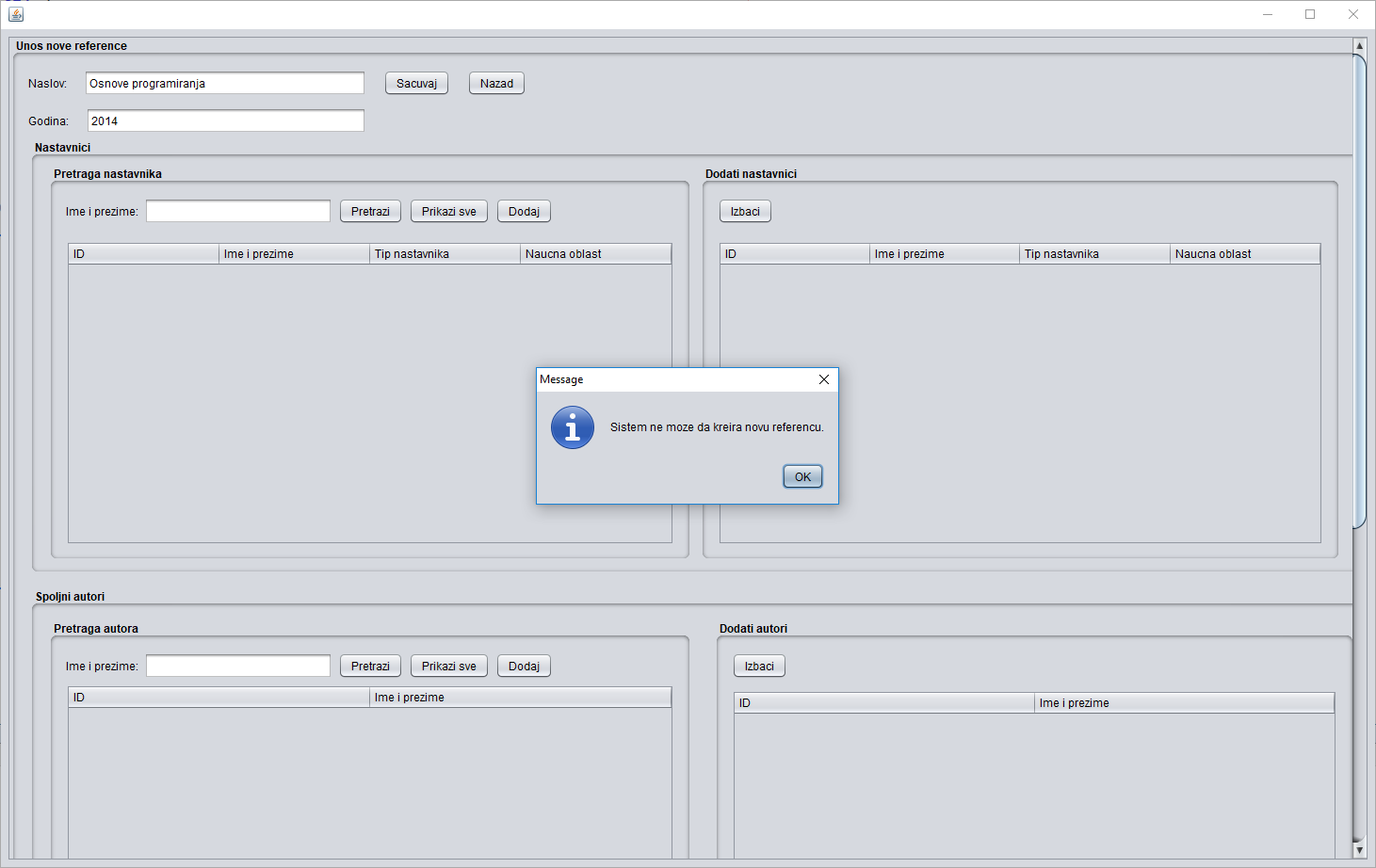
**Основни сценарио СК**

1. Запослени **уноси** податке о новој референци. (АПУСО)
2. Запослени **контролише** да ли је коректно унео податке о новој референци. (АНСО)
3. Запослени **позива** систем да запамти податке о референци. (АПСО)
4. Запослени **позива** систем да креира нову референцу. (АПСО)
5. Систем **креира** нову референцу. (СО)
6. Систем **приказује** запосленом поруку: “Систем је креирао нову референцу“. (ИА)



**Алтернативна сценарија**

6.1 Уколико систем не може да креира нову референцу он приказује запосленом поруку: “Систем не може да креира нову референцу”. (ИА)



### 3.3.2 Пројектовање апликационе логике

Апликациони сервери обезбеђују реализацију апликационе логике. Апликациони сервер садржи:

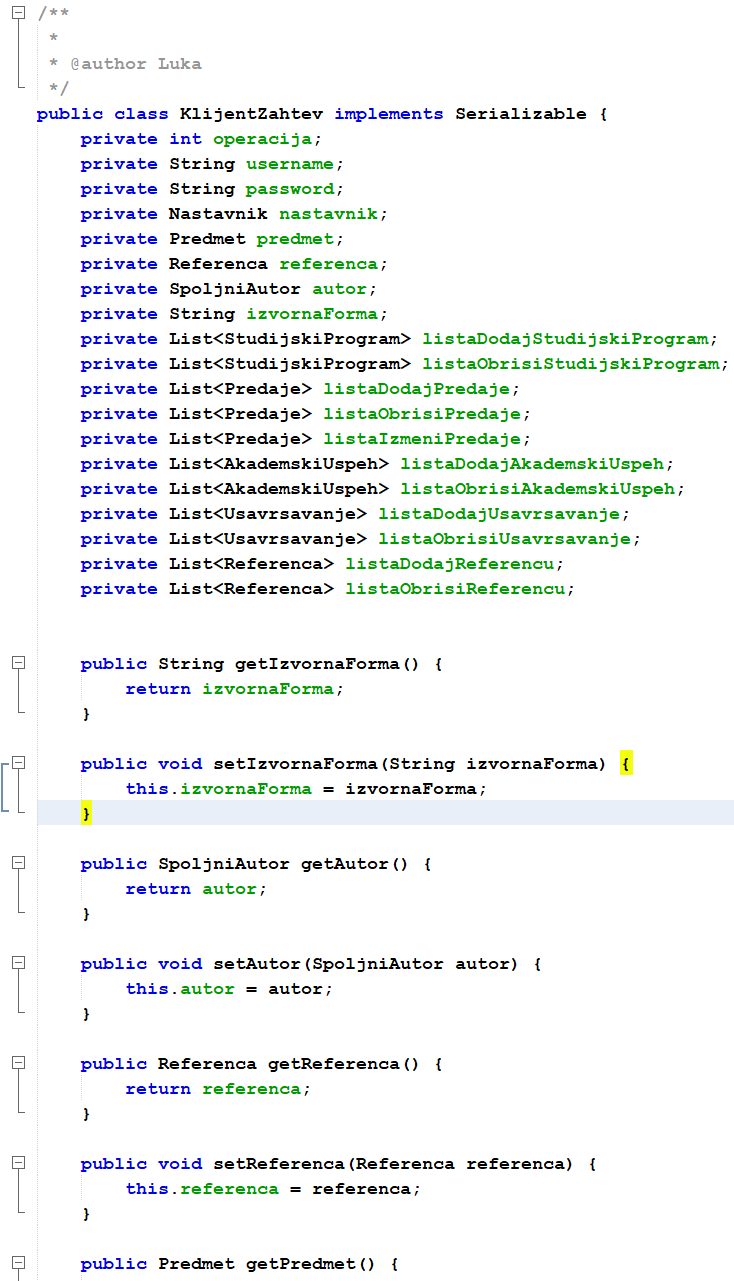
* Део за комуникацију са клијентима
* Контролер апликационе логике
* Део који садржи пословну логику
* Брокер базе података, који комуницира директно са базом

#### 3.3.2.1 Део за комуникацију са клијентима

Део за комуникацију са клијентом представља класа PokretanjeServera која подиже серверски сокет који ослушкује мрежу и чува листу повезаних клијената. Када клијент успостави везу, сервер генерише нит која ће бити одговорна за двосмерну везу са клијентом, убацује нит у мапу повезаних клијената.

Класа ObradaZahtevaKlijentaNit (додељена клијенту) прихвата захтеве које добија од клијентске апликације. У зависности од врсте захтева, позива одговарајуће системске операције. Након извршења системске операције, резултат извршења прослеђује одговор клијенту. Комуникација између клијента и сервера се обавља разменом објеката класа KlijentZahtev и ServerOdgovor.

На следећим сликама дат је приказ класа KlijentZahtev и ServerOdgovor.



Слика . Класа KlijentZahtev



Слика . Класа ServerOdgovor

#### 3.3.2.2 Контролер апликационе логике

Контролер апликационе логике обрађује прихваћене захтеве за извршавањем системских операција. Након завршетка системске операције, контролер прихвата резултат и враћа га клијенту.

За сваку системску операцију, прави се класа задужена за њено извршавање. Након прихватања захтева за извршењем неке од системских операција, контролер га прослеђује до одговарајуће класе системске операције.

Класе одговорне за извршење системских операција наслеђују класу OpstaSO, у којој је дефинисана веза са базом и омогућује праћење извршења СО као трансакција. Метода izvrsenjeSO(Object object) садржи позив три методе, од којих прва отвара конекцију са базом, друга проверава предуслове (ако их има) и трећа извршава операцију. Уколико дође до изузетка, поништава се трансакција, у супротном се потврђује.

У наставку је дат приказ апстрактне класе OpstaSO.

public abstract class OpstaSO {

protected Object object;

protected DBBroker db;

public OpstaSO(Object object, DBBroker db) {

this.object = object;

this.db = db;

}

public Object getObject() {

return object;

}

public void setObject(Object object) {

this.object = object;

}

public DBBroker getDb() {

return db;

}

public void setDb(DBBroker db) {

this.db = db;

}

private void uspostaviKonekciju() throws Exception {

db.uspostaviKonekciju();

}

private void prekiniKonekciju() throws Exception {

db.prekiniKonekciju();

}

private void commit() throws Exception {

db.commit();

}

private void rollback() throws Exception {

db.rollBack();

}

protected abstract void izvrsiOperaciju(Object object) throws Exception;

public final void izvrsenjeSO() throws Exception {

try {

uspostaviKonekciju();

izvrsiOperaciju(object);

commit();

} catch (Exception e) {

rollback();

throw e;

} finally {

prekiniKonekciju();

}

}

}

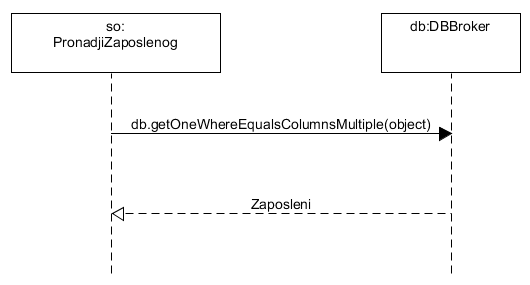
#### 3.3.2.3 Пројектовање системских операција

У почетку пројектовања СО треба треба направити концептуалне реализације (решења) за сваку СО.Концептуалне реализације требају да буду директно повезане са логиком проблема. Концептуалне реализације се могу описати преко објектног псеудокода, дијаграма сарадње, секвенцних дијаграма, дијаграма активности, дијаграма прелаза стања или дијаграма структура.[6]

За сваки од уговора пројектује се концептуално решење. У овом раду, коришћени су дијаграми секвенци за опис концептуалне реализације.

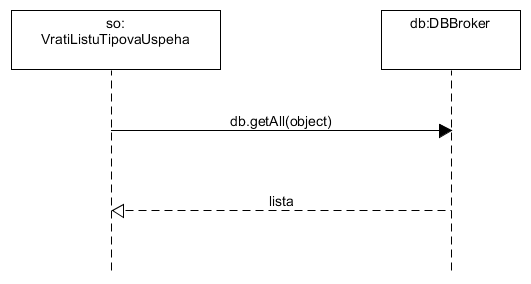
##### 3.3.2.3.1 Уговор UG1: PronadjiZaposlenog

**Операција: PronadjiZaposlenog**(Zaposleni):signal; **Веза са СК:** СК1  
**Предуслови:  
Постуслови:**

****

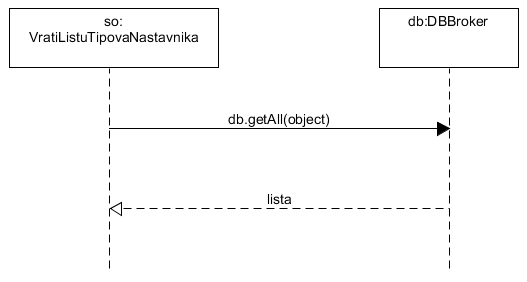
##### 3.3.2.3.2 Уговор UG2: VratiListuTipovaUspeha

**Операција: VratiListuTipovaUspeha**(List<TipUspeha>):signal; **Веза са СК:** СК2, СК3  
**Предуслови:  
Постуслови:**

****

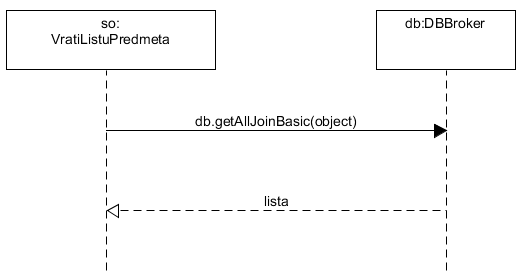
##### 3.3.2.3.3 Уговор UG3: VratiListuTipovaNastavnika

**Операција: VratiListuTipovaNastavnika**(List<TipNastavnika>):signal; **Веза са СК:** СК2, СК3  
**Предуслови:  
Постуслови:**

****

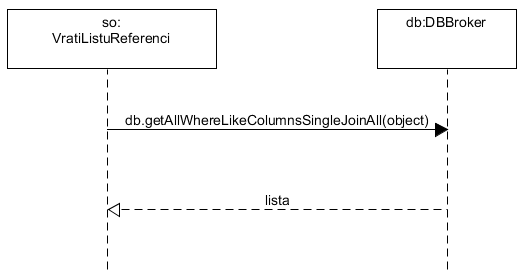
##### 3.3.2.3.4 Уговор UG4: VratiListuPredmeta

**Операција: VratiListuPredmeta**(List<Predmet>):signal; **Веза са СК:** СК2, СК3  
**Предуслови:  
Постуслови:**

****

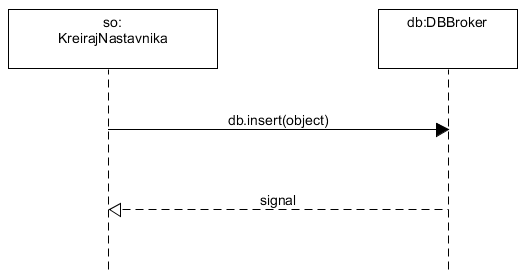
##### 3.3.2.3.5 Уговор UG5: VratiListuReferenci

**Операција: VratiListuReferenci**(List<Referenca>):signal; **Веза са СК:** СК2, СК3  
**Предуслови:  
Постуслови:**

****

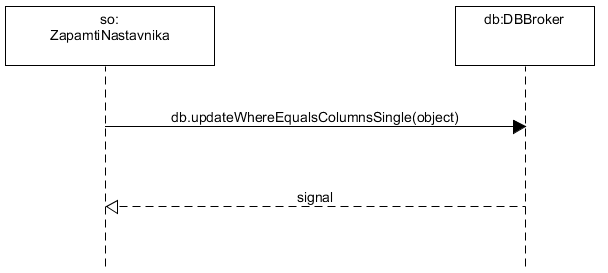
##### 3.3.2.3.6 Уговор UG6: KreirajNastavnika

**Операција: KreirajNastavnika**(Nastavnik):signal; **Веза са СК:** СК2  
**Предуслови:** *Вредносна и структурна ограничења над објектом Nastavnik**морају бити задовољена.* **Постуслови:** *Креиран је нови наставник*



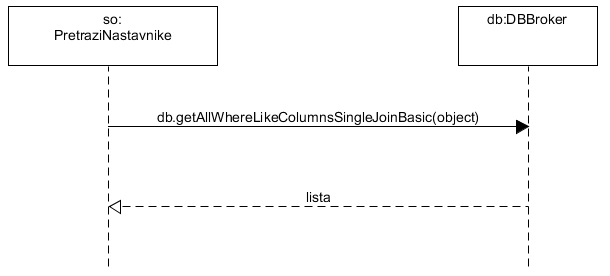
##### 3.3.2.3.7 Уговор UG7: ZapamtiNastavnika

**Операција: ZapamtiNastavnika**(Nastavnik):signal; **Веза са СК:** СК2, СК3 **Предуслови:** *Вредносна и структурна ограничења над објектом Nastavnik**морају бити задовољена.* **Постуслови:** *Подаци о наставнику су запамћени.*



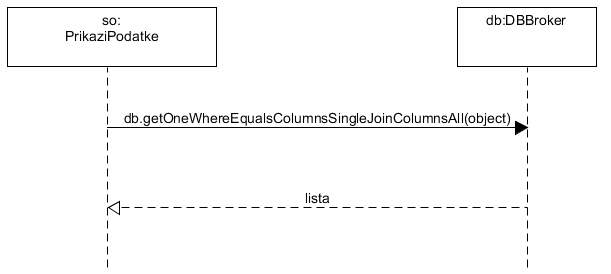
##### 3.3.2.3.8 Уговор UG8: PretraziNastavnike

**Операција: PretraziNastavnike**(KriterijumPretrage, List<Nastavnik>):signal; **Веза са СК:** СК3, СК4, СК5 **Предуслови:  
Постуслови:**

****

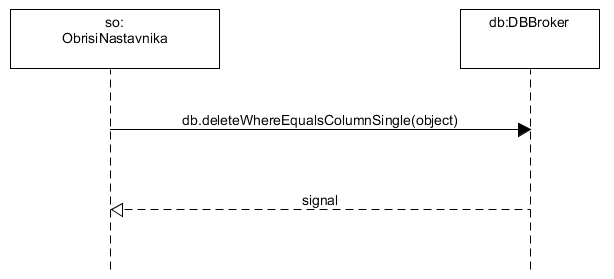
##### 3.3.2.3.9 Уговор UG9: PrikaziPodatke

**Операција: PrikaziPodatke**(Nastavnik):signal; **Веза са СК:** СК3, СК4, СК5 **Предуслови:  
Постуслови:**

****

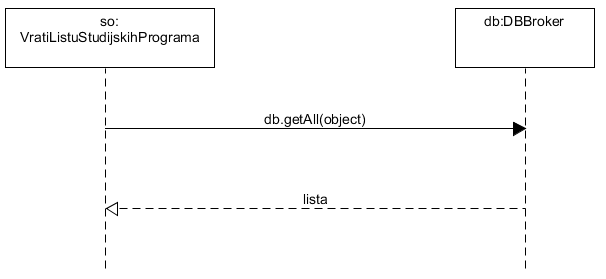
##### 3.3.2.3.10 Уговор UG10: ObrisiNastavnika

**Операција: ObrisiNastavnika**(Nastavnik):signal; **Веза са СК:** СК5 **Предуслови:** *Структурно ограничење над објектом Nastavnik мора бити задовољено.* **Постуслови:** *Наставник је обрисан.*



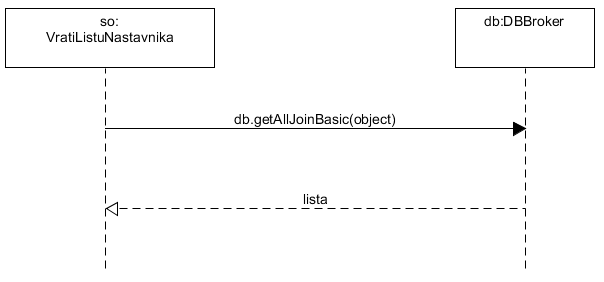
##### 3.3.2.3.11 Уговор UG11: VratiListuStudijskihPrograma

**Операција: VratiListuStudijskihPrograma**(List<StudijskiProgram>):signal; **Веза са СК:** СК6, СК7  
**Предуслови:  
Постуслови:**

****

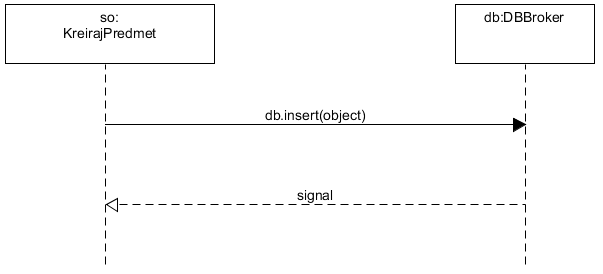
##### 3.3.2.3.12 Уговор UG12: VratiListuNastavnika

**Операција: VratiListuNastavnika**(List<Nastavnik>):signal; **Веза са СК:** СК6, СК7, CK10  
**Предуслови:  
Постуслови:**

****

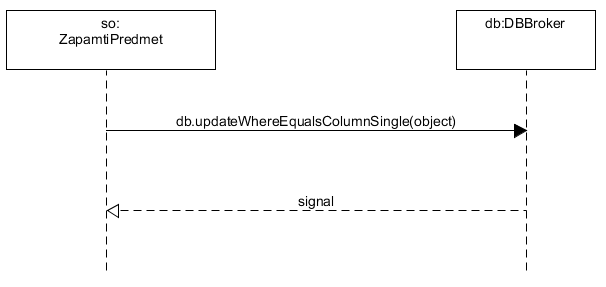
##### 3.3.2.3.13 Уговор UG13: KreirajPredmet

**Операција: KreirajPredmet**(Predmet):signal; **Веза са СК:** СК6 **Предуслови:** *Вредносна и структурна ограничења над објектом Predmet**морају бити задовољена.* **Постуслови:** *Креиран је нови предмет.*

****

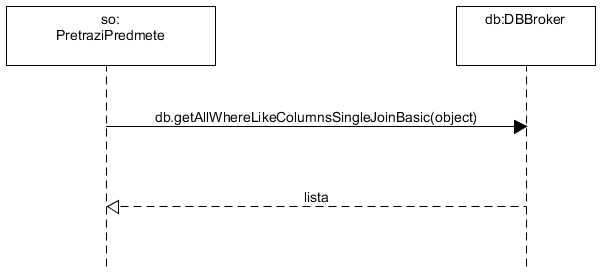
##### 3.3.2.3.14 Уговор UG14: ZapamtiPredmet

**Операција: ZapamtiPredmet**(Predmet):signal; **Веза са СК:** СК6, СК7 **Предуслови:** *Вредносна и структурна ограничења над објектом Predmet**морају бити задовољена.* **Постуслови:** *Подаци о предмету су запамћени.*



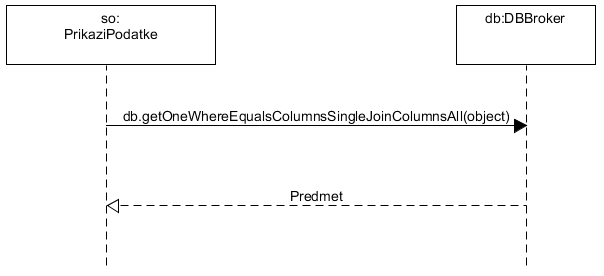
##### 3.3.2.3.15 Уговор UG15: PretraziPredmete

**Операција: PretraziPredmete**(KriterijumPretrage, List<Predmet>):signal; **Веза са СК:** СК7, СК8, СК9 **Предуслови:  
Постуслови:**

****

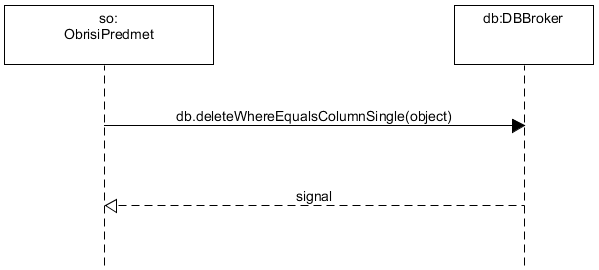
##### 3.3.2.3.16 Уговор UG16: PrikaziPodatke

**Операција: PrikaziPodatke**(Predmet):signal; **Веза са СК:** СК7, СК8, СК9 **Предуслови:  
Постуслови:**

****

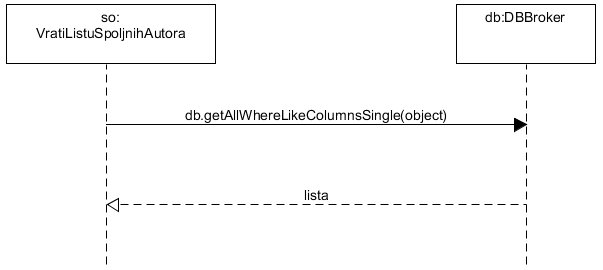
##### 3.3.2.3.17 Уговор UG17: ObrisiPredmet

**Операција: ObrisiPredmet**(Predmet):signal; **Веза са СК:** СК9 **Предуслови:** *Структурна ограничења над објетком Predmet морају бити задовољена.* **Постуслови:** *Предмет је обрисан.*

**

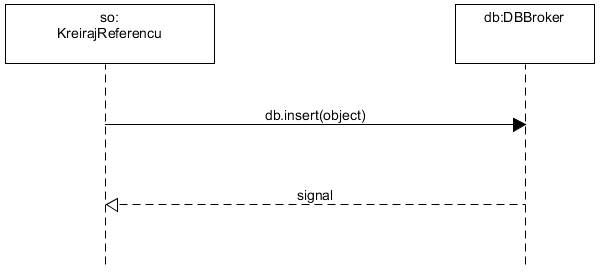
##### 3.3.2.3.18 Уговор UG18: VratiListuSpoljnihAutora

**Операција: VratiListuSpoljnihAutora**(List<SpoljniAutor>):signal; **Веза са СК:** СК10  
**Предуслови:  
Постуслови:**

****

##### 3.3.2.3.19 Уговор UG19: KreirajReferencu

**Операција: KreirajReferencu**(Referenca):signal; **Веза са СК:** СК10 **Предуслови:** *Вредносна и структурна ограничења над објектом Referenca**морају бити задовољена.* **Постуслови:** *Креирана је нова референца.*



#### 3.3.2.4 Брокер базе података

Брокер базе података, класа DBBroker, обезбеђује перзистентни оквир који посредује у свим операцијама над базом података и реализује следеће методе:

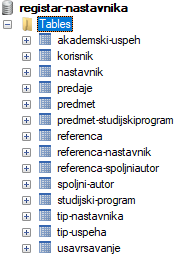
* void commit()
* void rollback()
* List<IDomen> getAllJoinBasic(IDomen domen)
* List<IDomen> getAllJoinBasic(IDomen domen)
* List<IDomen> getAllWhereLikeColumnsSingleJoinBasic(IDomen domen)
* IDomen getOneWhereEqualsColumnsSingleJoinColumnsAll(IDomen domen)
* IDomen getOneWhereEqualsColumnsSingleJoinColumnsBasic(IDomen domen)
* List<IDomen> getAllWhereEqualsColumnsSingleJoinColumnsBasic(IDomen domen)
* List<IDomen> getAllWhereEqualsColumnsSingle(IDomen domen)
* List<IDomen> getAllWhereLikeColumnsSingle(IDomen domen)
* List<IDomen> getAll(IDomen domen)
* int insert(IDomen domen)
* List<IDomen> getAllWhereLikeColumnsSingleJoinAll(IDomen domen)
* void deleteWhereEqualsColumnSingle(IDomen domen)
* void deleteWhereEqualsColumnSingle2(IDomen domen)
* void deleteWhereEqualsColumnMultiple(IDomen domen)
* void updateWhereEqualsColumnSingle(IDomen domen)
* void updateWhereEqualsColumnMultiple(IDomen domen)
* IDomen getOneWhereEqualsColumnsMultiple(IDomen domen)

Методе класе DBBroker су пројектоване као генеричке, што значи да могу да прихвате различите доменске објекте преко параметара. На тај начин постижемо да у DBBroker класи немамо имплементацију појединачних метода за сваку доменску класу.

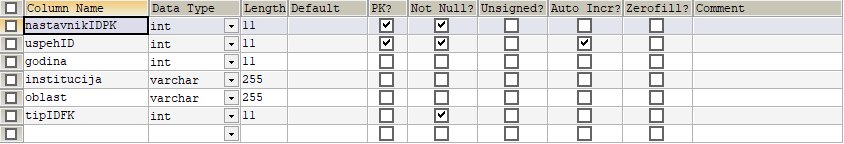
Ово је омогућено дефинисањем интерфејса IDomen који имплементирају доменске класе и чије методе, различито имплементиране у свакој доменској класи, омогућавају коришћење генеричких метода.

### 3.3.3 Пројектовање складишта података

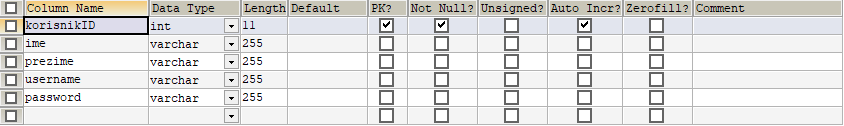
На основу релационог модела пројектоване су табеле (складишта података) релационог система за управљање базом података. У овом раду, коришћен је MySQL СУБП. На следећим сликама, приказане су све табеле базе података.



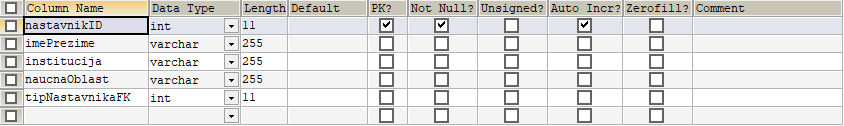
Слика . Табеле базе података



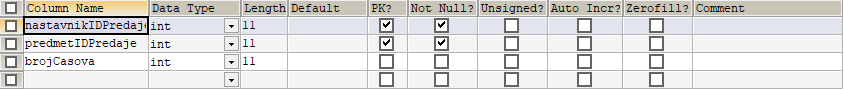
Слика . Табела akademski-uspeh



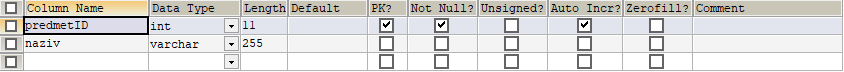
Слика . Табела korisnik



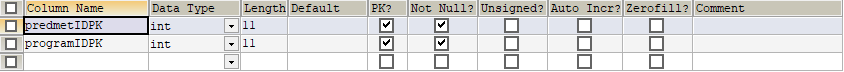
Слика . Табела nastavnik



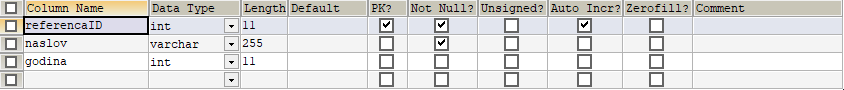
Слика . Табела predaje



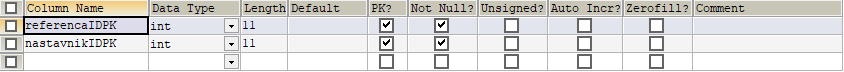
Слика . Табела predmet



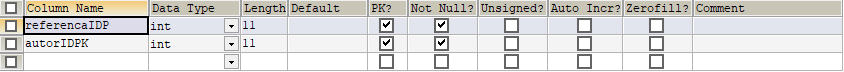
Слика . Табела predmet-studijskiprogram



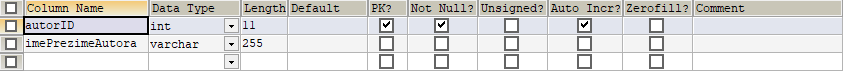
Слика . Табела referenca



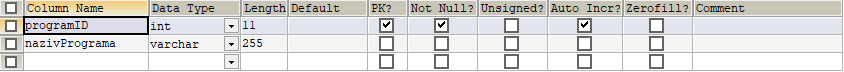
Слика . Табела referenca-nastavnik



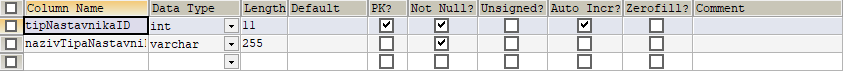
Слика . Табела referenca-spoljniautor



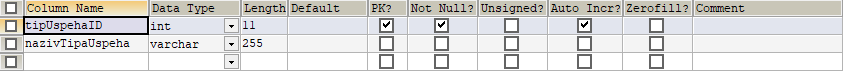
Слика . Табела spoljni-autor



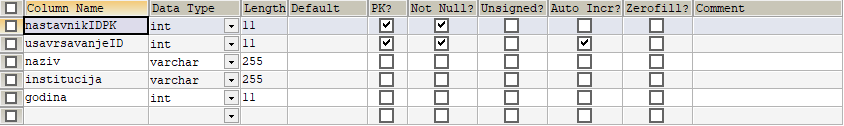
Слика . Табела studijski-program



Слика . Табела tip-nastavnika

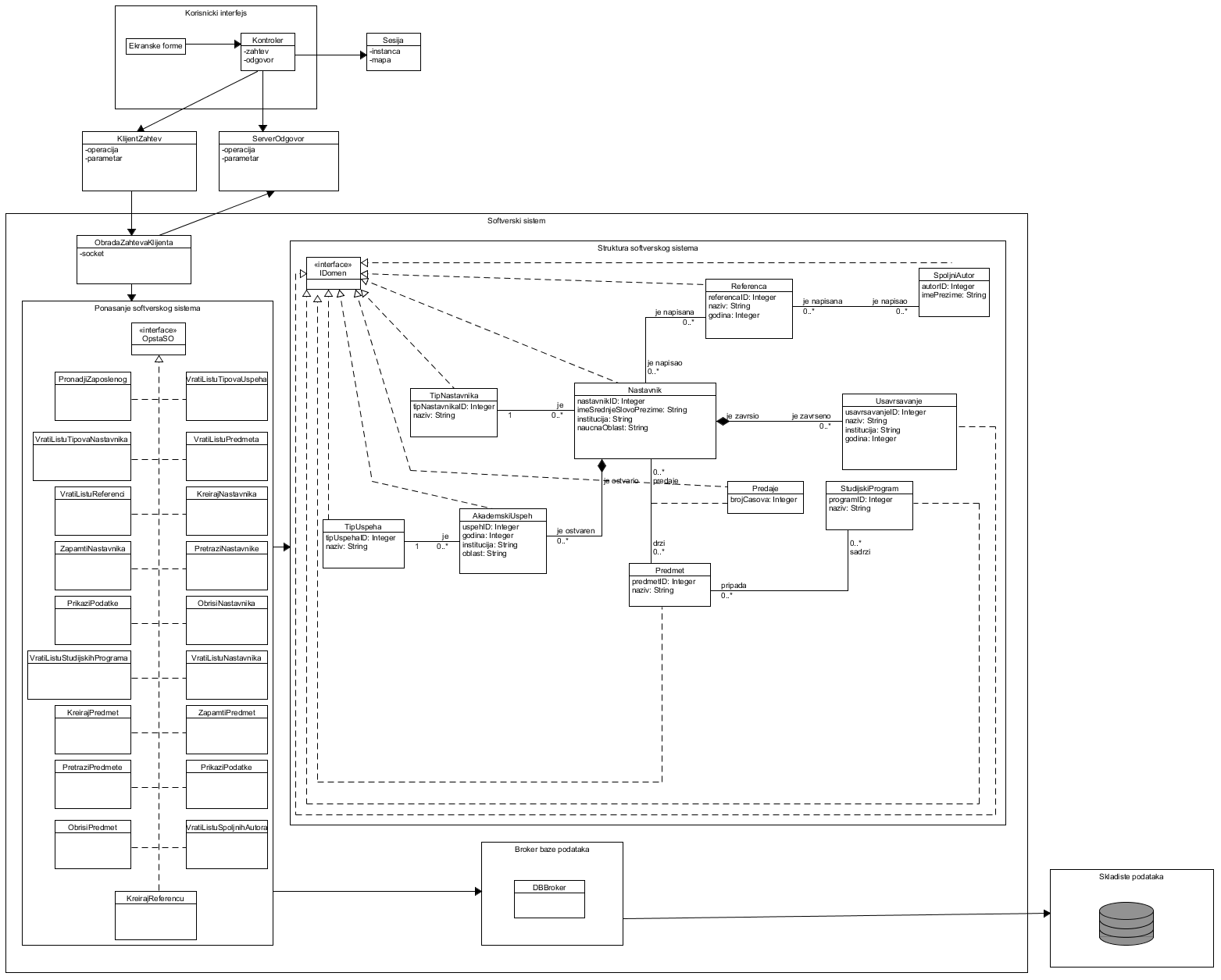


Слика . Табела tip-uspeha



Слика . Табела usavrsavanje

Након пројектовања складишта података, могуће је приказати целокупну архитектуру софтверског система. Архитектура је приказана на следећој слици.



Слика . Архитектура софтверског система

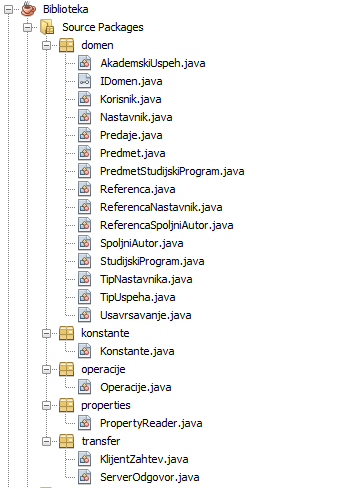
## 3.4 Имплементација

Софтверски систем је развијен у програмском језику Java и пројектован је као клијент-сервер апликација. Као развојно окружење коришћен је NetBeans IDE, а као систем за упраљање базом података коришћен је SQLYog.

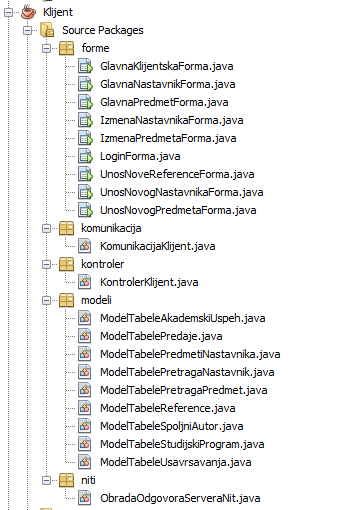
Систем је реализован у три пројекта:

* Biblioteka
* Klijent
* Server

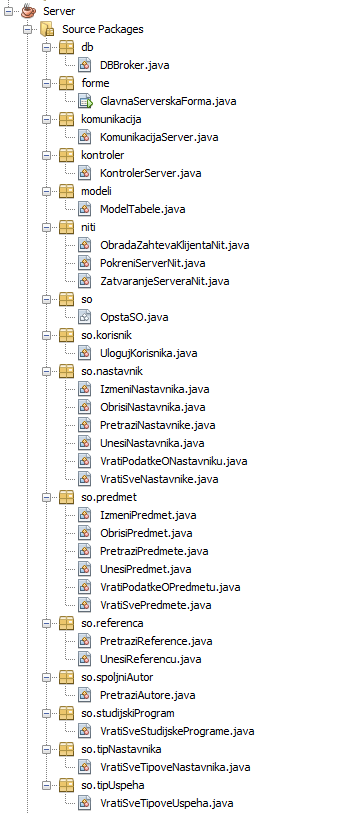
На основу архитектуре софтверског система, добијене су следеће класе:



Слика . Класе библиотеке



Слика . Класе клијентског програма



Слика . Класе серверског програма

# 4. Закључак

Главни циљ развоја софтверског система био је побољшање фунцкионалности студентске службе једног факултета.

Принцип објектно-оријентисаног програмирања (ООП) обезбеђује софтверски систем који има високу сигурност, хијерархијску структуру програмског кода, могућност лаког одржавања и надоградње. Такође, једна од главних предности овог принципа је могућност употребе једном написаног програмског кода на више различитих места у апликацији.

Софтверски систем приказан у студијском примеру је развијен коришћењем Јава програмског језика, заснован на принципу ООП. Могућност покретања на различитим оперативним системима је један од главних разлога зашто је Јава тренутно најпопуларнији програмски језик. За повезивање апликације са системом за управљање базом података коришћен је JDBC апликациони програмски интерфејс.

Развој софтверског система је заснован на упрошћеној Лармановој методи. Архитектура софтверског система је тронивојска и организована у складу са клијент-сервер архитектуром. Кориснички интерфејс се налази на страни клијента, док се пословна логика и складиште података налазе на страни сервера.

Овако постављен софтверски систем има добру основу за даљи развој и лаку имплементацију нових корисничких захтева у будућности.

# 5. Литература

[1] Java (programming language) – Wikipedia, <https://en.wikipedia.org/wiki/Java_(programming_language)>, датум приступа: Септембар 2018.

[2] History of Java – Javatpoint, <https://www.javatpoint.com/history-of-java>, датум приступа: Септембар 2018.

[3] Др Синиша Влајић, Душан Савић, Војислав Станојевић, Илија Антовић, Милош Милић: Пројектовање софтвера – Напредне Јава технологије, Факултет организационих наука, Београд 2008.

[4] TIOBE Index, <https://www.tiobe.com/tiobe-index/>, датум приступа: Септембар 2018.

[5] Вања Елчић, Развој апликације за обрачун личних доходака коришћењем Ларманове методе за развој софтвера, ИНФОТЕХ-ЈАХОРИНА, Vol. 10, Ref. F-2, p. 907-911, 2011, https://infoteh.etf.ues.rs.ba/zbornik/2011/radovi/F/F-2.pdf, датум приступа: Септембар 2018

[6] Др Синиша Влајић, Пројектовање софтвера(скрипта), Факултет организационих наука, Београд, 2015.

1. Слика преузета са [↑](#footnote-ref-1)
2. Слика преузета са https://1.bp.blogspot.com/-koz7AoOMRPQ/TsDnA35Qu6I/AAAAAAAAB5A/MezzxcoVxM4/s640/jasperreport\_overview.jpg [↑](#footnote-ref-2)