**УНИВЕРЗИТЕТ У БЕОГРАДУ**

**ФАКУЛТЕТ ОРГАНИЗАЦИОНИХ НАУКА**

**СЕМИНАРСКИ РАД**

**ТЕМА: Софтверски систем за музичку издавачку кућу**

|  |  |
| --- | --- |
| Ментор: | Студент: |
| **Проф др Синиша Влајић** | **Милош Михаиловић, 240/2013** |

Београд, 2018. године

Садржај:

[1. Прикупљање захтева 1](#_Toc528690705)

[1.1. Вербални опис модела 1](#_Toc528690706)

[1.2. Случајеви коришћења 2](#_Toc528690707)

[1.2.1. СК1: Креирање извођача 3](#_Toc528690708)

[1.2.2. СК2: Претраживање извођача 4](#_Toc528690709)

[1.2.3. СК3: Измена извођача 5](#_Toc528690710)

[1.2.4. СК4: Креирање песме 6](#_Toc528690711)

[1.2.5. СК5: Претраживање песме 7](#_Toc528690712)

[1.2.6. СК6: Измена песме 8](#_Toc528690713)

[1.2.7. СК7: Креирање албума (сложен СК) 9](#_Toc528690714)

[1.2.8. СК8: Претраживање албума 10](#_Toc528690715)

[1.2.9. СК10: Брисање албума 11](#_Toc528690717)

[2. Анализа 1](#_Toc528690718)2

[2.1. Системски дијаграм секвенци 1](#_Toc528690719)2

[2.1.1. ДС1: Креирање извођача](#_Toc528690720) 12

[2.1.2. ДС2: Претраживање извођача](#_Toc528690721) 15

[2.1.3. ДС3: Измена извођача](#_Toc528690722) 17

[2.1.4. ДС4: Креирање песме](#_Toc528690723) 21

[2.1.5. ДС5: Претраживање песме](#_Toc528690724) 24

[2.1.6. ДС6: Измена песме](#_Toc528690725) 26

[2.1.7. ДС7: Креирање жанра](#_Toc528690726) 28

[2.1.8. ДС8: Креирање албума (сложен)](#_Toc528690727) 31

[2.1.9. ДС9: Брисање албума](#_Toc528690728) 34

[2.2. Понашање софтверског система – дефинисање уговора о системским операцијама 37](#_Toc528690730)

2.3. Структура софтверског система – концептуални (доменски) модел 41

2.4 Структура софтверског система – релациони модел ....................41

[3. Пројектовање](#_Toc528690731) 44

[3.1. Архитектура софтверског система 44](#_Toc528690709)

[3.2. Пројектовање корисничког интерфејса 45](#_Toc528690710)

[3.3. Пројектовање екранских форми 46](#_Toc528690711)

[3.4. Пројектовање контролера корисничког интерфејса](#_Toc528690712) 69

[3.5. Пројектовање апликационе логике](#_Toc528690713) 70

[3.5.1 Контролер апликационе логике](#_Toc528690714) 70

[3.5.2. Системске операције 85](#_Toc528690715)

[3.5.3. Доменске класе](#_Toc528690717) 103

[3.5.4. Брокер базе података](#_Toc528690715) 103

[3.5.5. Пројектовање складишта података](#_Toc528690717) 107

[4. Имплементација](#_Toc528690732) 111

[5. Тестирање](#_Toc528690732) 115

[6. Литература](#_Toc528690732) 115

# Прикупљање захтева

### Вербални опис модела

Потребно је креирати софтверски систем за издавачку кућу. Овај систем треба да обезбеди менаџеру идавачке куће да воде евиденцију о свим извођачима са којима сарађују, албумима и песмама и њиховом жанру. Као такав, функционалности које овај систем треба да садржи су следеће:

* Креирање профила извођача
* Креирање песме са њиховим извођачем
* Евидентирање албума са жанром и песмама

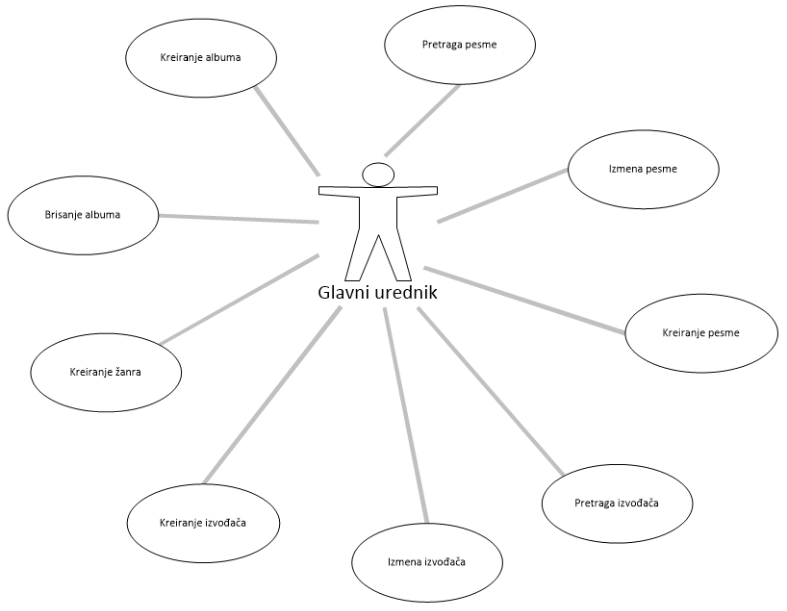
За извођача се могу издавати албуми и песме. Оно што је потребно нагласити када су у питању песме и албуми јесте следеће:

* Песма се може направити без обзира да ли постоји албум
* Свака песма има свог извођача
* Албум мора имати минимум једну песму
* Сваки албум мора да има свој жанр и листу песама

Сматра се да су текстописци део издавачке куће и као такви иницијално постоје у систему.

### 1.2. Случајеви коришћења

1. Креирање извођача
2. Претрага извођача
3. Измена извођача
4. Креирање песме
5. Претрага песме
6. Измена песме
7. Креирање жанра
8. Креирање албума
9. Брисање албума



Слика 1. Случајеви коришћења – Главни уредник

### СК1: Креирање извођача

**СК1: Случај коришћења – Креирање извођача**

**Назив СК**

Креирање извођача

**Актори СК**

Главни уредник

**Учесници СК**

Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник је улогован под својом шифром. Систем приказује форму за рад са извођачима.

**Основни сценарио СК**

1. Главни уредник **позива** систем да креира извођача. (АПСО)
2. Систем креира извођача.(СО)
3. Систем **приказује** главном уреднику извођача и поруку: “Систем је креирао извођача“. (ИА)
4. Главни уредник уноси податке у систем.(АПУСО)
5. Главни уредник **контролише** да ли је коректно унео податке у систем. (АНСО)
6. Главни уредник **позива** систем да запамти податке о извођачу. (АПСО)
7. Систем **памти** податке о извођачу. (СО)
8. Систем **приказује** главном уреднику запамћени извођач и поруку: “Систем је запамтио извођача“. (ИА)

**Алтернативна сценариа**

3.1 Уколико систем не може да креира извођача он приказује главном уреднику поруку: “Систем не може да креира извођача”. Прекида се извршење сценариа. (ИА)

8.1 Уколико систем не може да запамти податке о извођачу он приказује главном уреднику поруку: “Систем не може да запамти извођача”. (ИА)

### СК2: Претраживање извођача

**СК2: Случај коришћења – Претраживање извођача**

**Назив СК**

Претраживање извођача

**Актори СК**

Главни уредник

**Учесници СК**

Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник je улогован под својом шифром. Систем приказује форму за рад са извођачима.

**Основни сценарио СК**

1. Главни уредник **уноси** вредност по којој претражује извођача. (АПУСО)
2. Главни уредник **позива** систем да нађе извођача по задатој вредности. (АПСО)
3. Систем **тражи** извођача по задатој вредности. (СО)
4. Систем **приказује** главном уреднику податке о извођачу и поруку: “Систем је нашао извођача по задатој вредности”.(ИА)

**Алтернативна сценариа**

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

### СК3: Измена извођача

**СК3: Случај коришћења – Измена извођача**

**Назив СК**

Измена извођача

**Актори СК**

Главни уредник

**Учесници СК**

Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник je улогован под својом шифром. Систем приказује форму за рад са извођачем.

**Основни сценарио СК**

1. Главни уредник **уноси** вредност по којој претражује извођаче. (АПУСО)
2. Главни уредник **позива** систем да нађе извођаче по задатој вредности. (АПСО)
3. Систем **тражи** извођаче по задатој вредности. (СО)
4. Систем **приказује** главном уреднику извођача и поруку: “Систем је нашао извођача по задатој вредности”. (ИА)
5. Систем **бира** извођача. (АПУСО)
6. Главни уредник **позива** систем да нађе изабраног извођача. (АПСО)
7. Систем **тражи** изабраног извођача. (СО)
8. Систем **приказује** главном уреднику извођача и поруку: “Систем је нашао извођача по задатој вредности”. (ИА)
9. Главни уредник **уноси(мења)** податке о извођачу. (АПУСО)
10. Главни уредник **контролише** да ли је коректно унео податке о извођачу. (АНСО)
11. Главни уредник **позива** систем да запамти податке о извођачу. (АПСО)
12. Систем **памти** податке о извођачу. (СО)
13. Систем **приказује** главном уреднику запамћени извођач и поруку: “Систем је запамтио извођача”. (ИА)

**Алтернативна сценариа**

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

8.1 Уколико систем не може да нађе извођача он приказује главном уреднику поруку: “Систем не може да нађе извођача по задатој вредности”. Прекида се извршење сценариа. (ИА)

13.1 Уколико систем не може да запамти податке о извођачу он приказује главном уреднику поруку: “Систем не може да запамти извођача”. (ИА)

### СК4: Креирање песме

**СК4: Случај коришћења – Креирање песме**

**Назив СК**

Креирање песме

**Актори СК**

Главни уредник

**Учесници СК**

Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник је улогован под својом шифром. Систем приказује форму за рад са песмама.

**Основни сценарио СК**

1. Главни уредник **позива** систем да креира песму. (АПСО)
2. Систем креира песму.(СО)
3. Систем **приказује** главном уреднику песму и поруку: “Систем је креирао песму“. (ИА)
4. Главни уредник уноси податке у систем.(АПУСО)
5. Главни уредник **контролише** да ли је коректно унео податке у систем. (АНСО)
6. Главни уредник **позива** систем да запамти податке о песми. (АПСО)
7. Систем **памти** податке о песми. (СО)
8. Систем **приказује** главном уреднику запамћену песму и поруку: “Систем је запамтио песму“. (ИА)

**Алтернативна сценариа**

3.1 Уколико систем не може да креира песму он приказује главном уреднику поруку: “Систем не може да креира песму”. Прекида се извршење сценариа. (ИА)

8.1 Уколико систем не може да запамти податке о песми он приказује главном уреднику поруку: “Систем не може да запамти песму”. (ИА)

### СК5: Претраживање песме

**СК5: Случај коришћења – Претраживање песме**

**Назив СК**

Претраживање песме

**Актори СК**

Главни уредник

**Учесници СК**

Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник je улогован под својом шифром. Систем приказује форму за рад са песмама.

**Основни сценарио СК**

1. Главни уредник **уноси** вредност по којој претражује песму. (АПУСО)
2. Главни уредник **позива** систем да нађе песму по задатој вредности. (АПСО)
3. Систем **тражи** песму по задатој вредности. (СО)
4. Систем **приказује** главном уреднику податке о песми и поруку: “Систем је нашао песму по задатој вредности”.(ИА)

**Алтернативна сценариа**

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

### СК6: Измена песме

**СК6: Случај коришћења – Измена песме**

**Назив СК**

Измена песме

**Актори СК**

Главни уредник

**Учесници СК**

Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник je улогован под својом шифром. Систем приказује форму за рад са песмама.

**Основни сценарио СК**

1. Главни уредник **позива** систем да нађе песму. (АПСО)
2. Систем **тражи** песму. (СО)
3. Главни уредник **бира** песму. (АПУСО)
4. Главни уредник **позива** систем да нађе изабрану песму. (АПСО)
5. Систем **тражи** изабрану песму. (СО)
6. Главни уредник **уноси(мења)** податке о песми. (АПУСО)
7. Главни уредник **контролише** да ли је коректно унео податке о песми. (АНСО)
8. Главни уредник **позива** систем да измени податке о песми. (АПСО)
9. Систем **памти** податке о песми. (СО)
10. Систем **приказује** главном уреднику измењену песму и поруку: “Систем је запамтио песму”. (ИА)

**Алтернативна сценариа**

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

8.1 Уколико систем не може да нађе песму он приказује главном уреднику поруку: “Систем не може да нађе песму по задатој вредности”. Прекида се извршење сценариа. (ИА)

13.1 Уколико систем не може да запамти податке о песми он приказује главном уреднику поруку: “Систем не може да запамти песму”. (ИА)

### СК7: Креирање Жанра

**СК7: Случај коришћења – Креирање жанра**

**Назив СК**

Ккреирање жанра

**Актори СК**

Главни уредник

**Учесници СК**

Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник je улогован под својом шифром. Систем приказује форму за рад са жанровима. Учитана је листа жанрова.

.

**Основни сценарио СК**

1. Главни уредник **позива** систем да креира жанр. (АПСО)
2. Систем креира жанр.(СО)
3. Систем **приказује** главном уреднику жанр и поруку: “Систем је креирао жанр“. (ИА)
4. Главни уредник уноси податке у систем.(АПУСО)
5. Главни уредник **контролише** да ли је коректно унео податке у систем. (АНСО)
6. Главни уредник **позива** систем да запамти податке о жанру. (АПСО)
7. Систем **памти** податке о жанру. (СО)
8. Систем **приказује** главном уреднику запамћени жанр и поруку: “Систем је запамтио жанр“. (ИА)

**Алтернативна сценариа**

3.1 Уколико систем не може да креира жанр он приказује главном уреднику поруку: “Систем не може да креира жанр ”. Прекида се извршење сценариа. (ИА)

8.1 Уколико систем не може да запамти податке о жанру он приказује главном уреднику поруку: “Систем не може да запамти жанр”. (ИА)

### СК7: Креирање албума

**СК8: Случај коришћења – Креирање албума**

**Назив СК**

Ккреирањеалбума

**Актори СК**

Главни уредник

**Учесници СК**

Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник je улогован под својом шифром. Систем приказује форму за рад са албумима. Учитана је листа песама и извођача.

.

**Основни сценарио СК**

1. Главни уредник **позива** систем да креира албум. (АПСО)
2. Систем креира албум.(СО)
3. Систем **приказује** главном уреднику албум и поруку: “Систем је креирао албум“. (ИА)
4. Главни уредник **уноси** вредност по којој претражује песму. (АПУСО)
5. Систем **тражи** песму по задатој вредности. (СО)
6. Главни уредник **уноси** жанр и песме у систем.(АПУСО)
7. Главни уредник **контролише** да ли је коректно унео податке у систем. (АНСО)
8. Главни уредник **позива** систем да запамти податке о албуму. (АПСО)
9. Систем **памти** податке о албуму. (СО)
10. Систем **приказује** главном уреднику запамћени албум и поруку: “Систем је запамтио албум“. (ИА)

**Алтернативна сценариа**

3.1 Уколико систем не може да креира албум он приказује главном уреднику поруку: “Систем не може да креира албум ”. Прекида се извршење сценариа. (ИА)

10.1 Уколико систем не може да запамти податке о албуму он приказује главном уреднику поруку: “Систем не може да запамти албум”. (ИА)

### СК9: Брисање албума

**СК9: Случај коришћења – Брисање албума**

**Назив СК**

Брисање албума

**Актори СК**

Главни уредник

**Учесници СК**

Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник je улогован под својом шифром. Систем приказује форму за рад са албумом. Излистана је листа песама и извођача.

**Основни сценарио СК**

1. Главни уредник **позива** систем да обрише албум. (АПСО)
2. Систем **брише** албум. (СО)
3. Систем **приказује** главном уреднику поруку: “Систем је обрисао албум”. (ИА)

**Алтернативна сценариа**

3.1 Уколико систем не може да обрише албум он приказује главном уреднику поруку: “Систем не може да обрише албум”. (ИА)

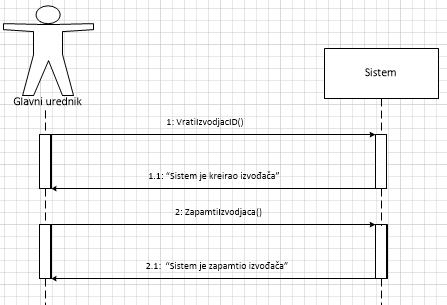
# Фаза анализе

## Системски дијаграм секвенци

### ДС1: Креирање извођача

**ДС1: Дијаграм секвенци случај коришћења – Креирање извођача**

1. Главни уредник **позива** систем да креира извођача. (АПСО)
2. Систем **приказује** главном уреднику извођача и поруку: “Систем је креирао извођача“. (ИА)
3. Главни уредник **позива** систем да запамти податке о извођачу. (АПСО)
4. Систем **приказује** главном уреднику запамћени извођач и поруку: “Систем је запамтио извођача“. (ИА)

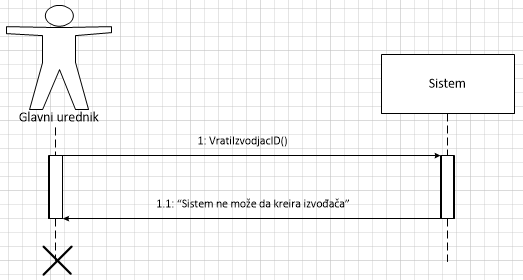


*Слика 3. - Дијаграм секвенци: СК1*

*Креирање извођача*

**Алтернативна сценариа**

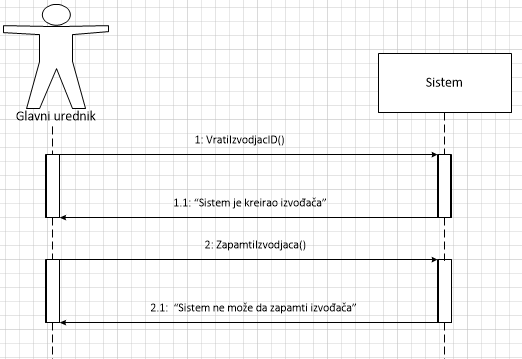
2.1 Уколико систем не може да креира извођача он приказује главном уреднику поруку: “Систем не може да креира извођача”. Прекида се извршење сценариа. (ИА)



*Слика 4. - Дијаграм секвенци: СК1*

*Креирање извођача – Алтернативни сценарио 1*

4.1 Уколико систем не може да запамти податке о извођачу он приказује главном уреднику поруку: “Систем не може да запамти извођача”. (ИА)



*Слика 5. - Дијаграм секвенци: СК1*

*Креирање извођача – Алтернативни сценарио 2*

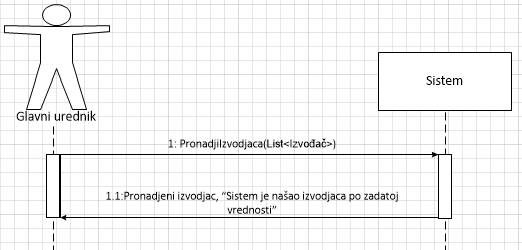
Са наведених секвенцних дијаграма уочавају се 2 системске операције које треба пројектовати:

1. Сигнал VratiIzvodjacID()
2. Сигнал ZapamtiIzvodjaca()

### ДС2: Претраживање извођача

**ДС2: Дијаграм секвенци случај коришћења – Претраживање извођача**

1. Главни уредник **позива** систем да нађе извођача по задатој вредности. (АПСО)
2. Систем **приказује** главном уреднику податке о извођачу и поруку: “ Систем је нашао извођача по задатој вредности”.(ИА)

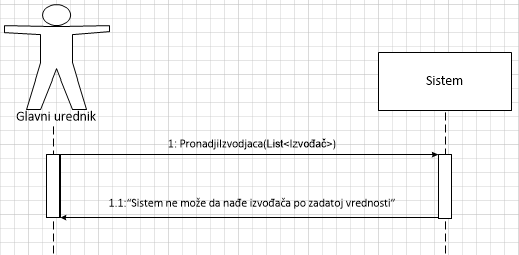


*Слика 6. - Дијаграм секвенци: СК2*

*Претраживање извођача*

**Алтернативна сценариа**

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



*Слика 7. - Дијаграм секвенци: СК2*

*Претраживање извођача – Алтернативни сценарио 1*

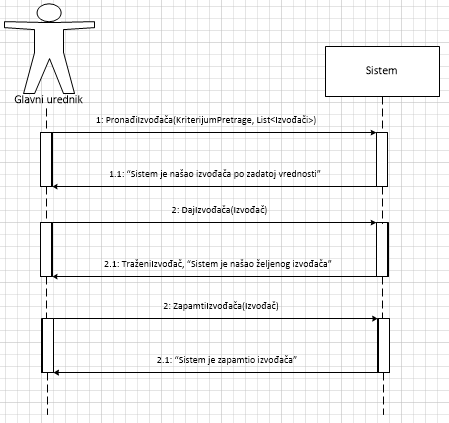
Са наведених секвенцних дијаграма уочава се 1 системскa операцијa којu треба пројектовати:

1. Сигнал PronadjiIzvođača(List<Izvođač>)

### ДС3: Измена извођача

**ДС3: Дијаграм секвенци случај коришћења – Измена извођача**

1. Главни уредник **позива** систем да нађе извођаче по задатој вредности. (АПСО)
2. Систем **приказује** главном уреднику извођача и поруку: “Систем је нашао извођача по задатој вредности”. (ИА)
3. Главни уредник **позива** систем да нађе изабраног извођача. (АПСО)
4. Систем **приказује** главном уреднику извођача и поруку: “Систем је нашао жељеног извођача”. (ИА)
5. Главни уредник **позива** систем да запамти податке о извођачу. (АПСО)
6. Систем **приказује** главном уреднику запамћени извођач и поруку: “Систем је запамтио извођача”. (ИА)

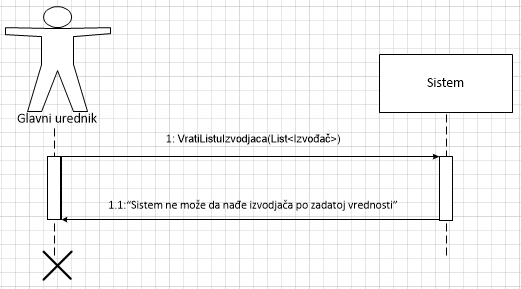


*Слика 8. – Дијаграм секвенци: СК3*

*Измена извођача*

**Алтернативна сценариа**

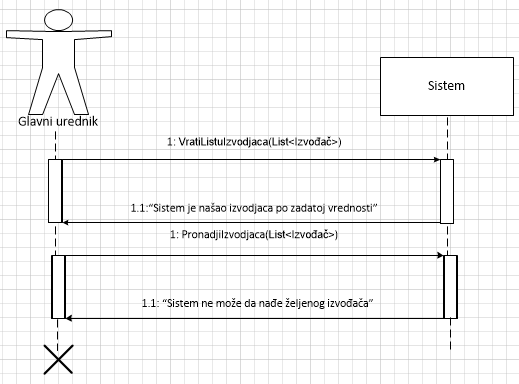
2.1.Уколико систем не може да нађе извођача он приказује главном уреднику поруку: “Систем не може да нађе извођача по задатој вредности”. Прекида се извршење сценариа. (ИА)



*Слика 9. - Дијаграм секвенци: СК3*

*Измена извођача – Алтернативни сценарио 1*

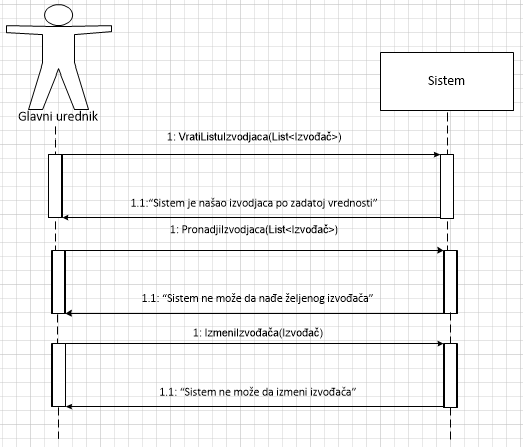
4.1.Уколико систем не може да нађе извођача он приказује главном уреднику поруку: “Систем не може да нађе жељеног извођача”. Прекида се извршење сценариа. (ИА)



*Слика 10. - Дијаграм секвенци: СК3*

*Измена извођача – Алтернативни сценарио 2*

* 1. Уколико систем не може да запамти податке о извођачу он приказује главном уреднику поруку: “Систем не може да запамти извођача”. (ИА)



*Слика 11. - Дијаграм секвенци: СК3*

*Измена извођача – Алтернативни сценарио 3*

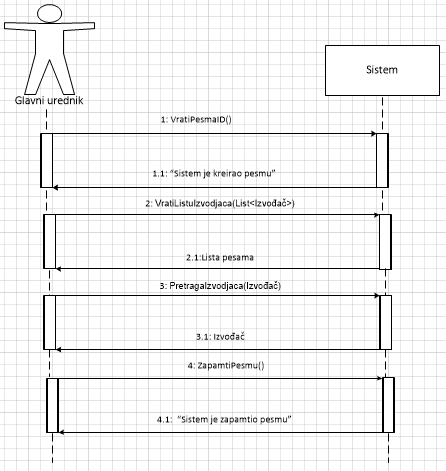
Са наведених секвенцних дијаграма уочавају се 2 системске операције које треба пројектовати:

1. Сигнал PronadjiIzvodjaca(List<Izvođač>)
2. Сигнал VratiListuIzvodjaca(List<Izvođač>)
3. Сигнал IzmeniIzvođača(Izvođač)

### ДС4: Креирање песме

**ДС4: Дијаграм секвенци случај коришћења – Креирање песме**

1. Главни уредник **позива** систем да креира песму. (АПСО)
2. Систем **приказује** главном уреднику песму и поруку: “Систем је креирао песму“. (ИА)
3. Главни уредник **позива** систем да запамти податке о песми. (АПСО)
4. Систем **приказује** главном уреднику запамћену песму и поруку: “Систем је запамтио песму“. (ИА)

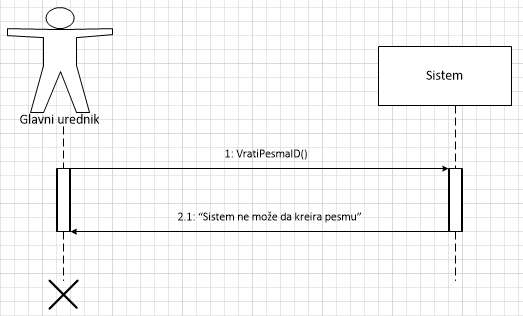


*Слика 12. – Дијаграм секвенци: СК4*

*Креирање песме*

**Алтернативна сценариа**

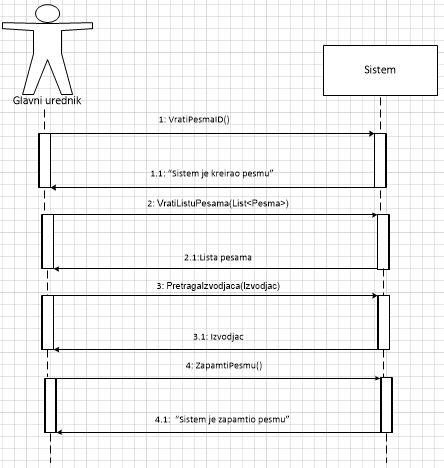
2.1 Уколико систем не може да креира песму он приказује главном уреднику поруку: “Систем не може да креира песму”. Прекида се извршење сценариа. (ИА)



*Слика 13. - Дијаграм секвенци: СК4*

*Креирање песме – Алтернативни сценарио 1*

4.1 Уколико систем не може да запамти податке о песми он приказује главном уреднику поруку: “Систем не може да запамти песму”. (ИА)



*Слика 14. - Дијаграм секвенци: СК4*

*Креирање песме – Алтернативни сценарио 2*

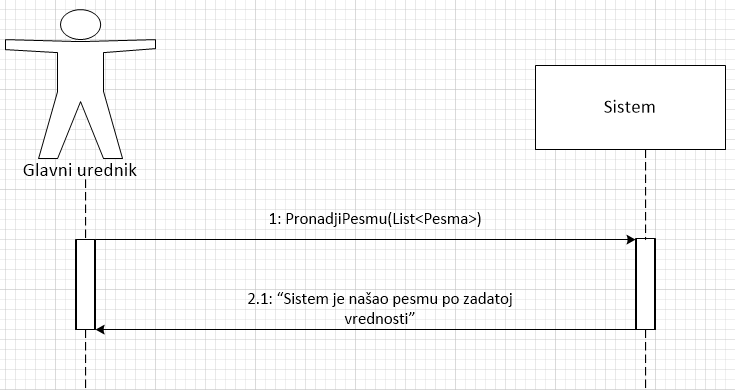
Са наведених секвенцних дијаграма уочавају се 2 системске операције које треба пројектовати:

1. Сигнал VratiPesmaID()
2. Сигнал VratiListuPesama(List<Pesma>)
3. Сигнал PretragaIzvodjaca(Izvodjac)
4. Сигнал ZapamtiPesmu()

### ДС5: Претраживање песме

**ДС5: Дијаграм секвенци случај коришћења – Претраживање песме**

1. Главни уредник **позива** систем да нађе песму по задатој вредности. (АПСО)
2. Систем **приказује** главном уреднику податке о песми и поруку: “Систем је нашао песму по задатој вредности”.(ИА)

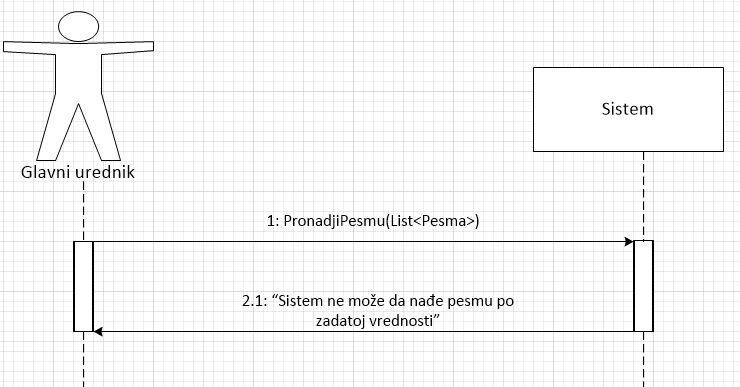


*Слика 15. – Дијаграм секвенци: СК5*

*Претраживање песме*

**Алтернативна сценариа**

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



*Слика 16. - Дијаграм секвенци: СК5*

*Претраживање песме – Алтернативни сценарио 1*

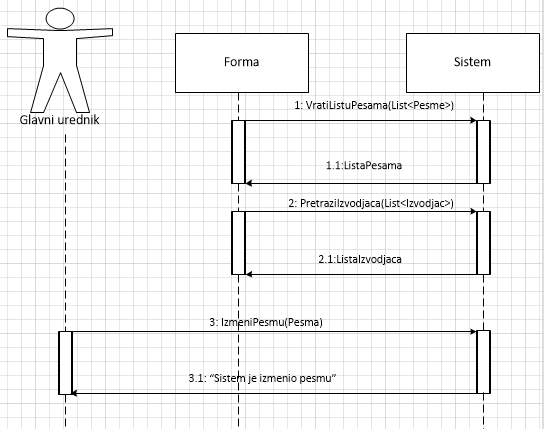
Са наведених секвенцних дијаграма уочавају се 2 системске операције које треба пројектовати:

1. Сигнал PronadjiPesmu(List<Pesma>)

### ДС6: Измена песме

**ДС6: Дијаграм секвенци случај коришћења – Измена песме**

1. Главни уредник **позива** систем да измени податке о песми. (АПСО)
2. Систем **приказује** главном уреднику запамћену песму и поруку: “Систем је изменио песму”. (ИА)

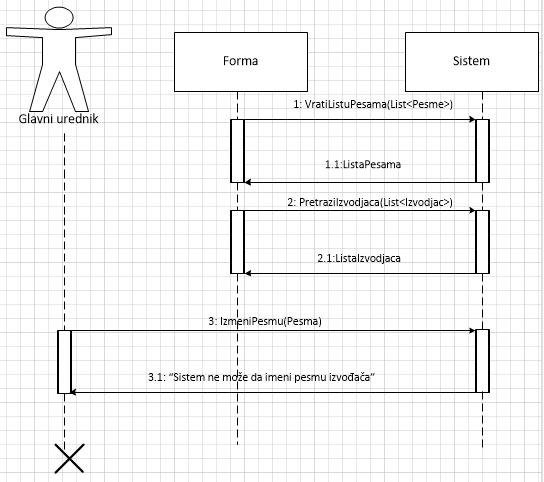


*Слика 17. – Дијаграм секвенци: СК6*

*Измена песме*

**Алтернативна сценариа**

10.1.Уколико систем не може да запамти податке о песми он приказује главном уреднику поруку: “Систем не може да имени песму извођача”. Прекида се извршење сценариа. (ИА)



*Слика 18. - Дијаграм секвенци: СК6*

*Измена песме – Алтернативни сценарио 1*

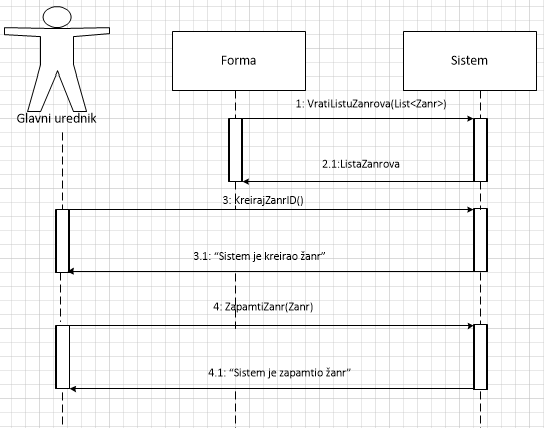
Са наведених секвенцних дијаграма уочавају се 2 системске операције које треба пројектовати:

1. Сигнал VratiListuPesama(List<Pesme>)
2. Сигнал PretraziIzvodjaca(List<Izvodjac>)
3. Сигнал IzmeniPesmu(Pesma)

### ДС7: Креирање жанра

**ДС7: Дијаграм секвенци случај коришћења – Креирање жанра**

1. Главни уредник **позива** систем да креира жанр. (АПСО)
2. Систем **приказује** главном уреднику жанр и поруку: “Систем је креирао жанр“. (ИА)
3. Главни уредник **позива** систем да запамти податке о жанру. (АПСО)
4. Систем **приказује** главном уреднику запамћену песму и поруку: “Систем је запамтио жанр“. (ИА)

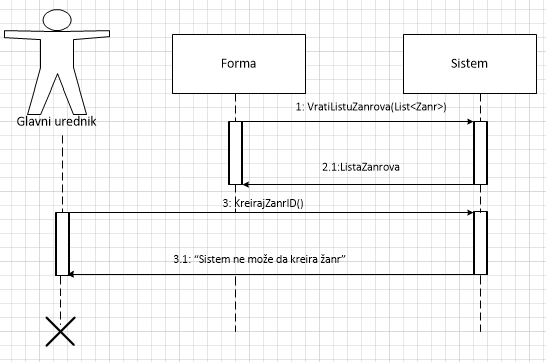


*Слика 19. – Дијаграм секвенци: СК7*

*Креирање жанра*

**Алтернативна сценариа**

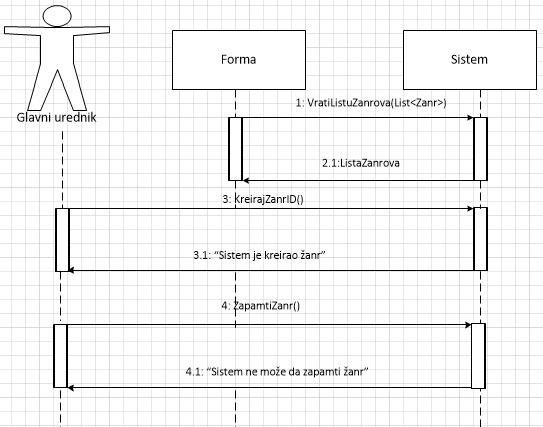
2.1 Уколико систем не може да креира жанр он приказује главном уреднику поруку: “Систем не може да креира жанр”. Прекида се извршење сценариа. (ИА)



*Слика 20. - Дијаграм секвенци: СК7*

*Креирање жанра – Алтернативни сценарио 1*

4.1 Уколико систем не може да запамти податке о жанру он приказује главном уреднику поруку: “Систем не може да запамти жанр”. (ИА)



*Слика 21. - Дијаграм секвенци: СК7*

*Креирање жанра – Алтернативни сценарио 2*

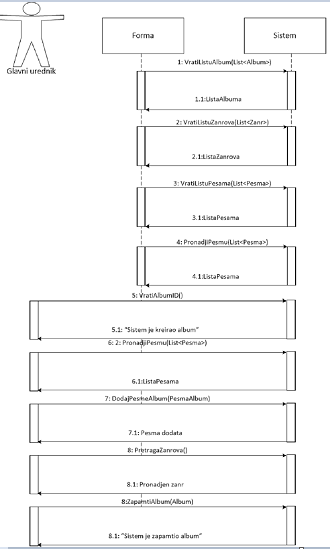
Са наведених секвенцних дијаграма уочавају се 2 системске операције које треба пројектовати:

1. Сигнал VratiListuZanrova(List<Zanr>)
2. Сигнал KreirajZanrID()
3. Сигнал ZapamtiZanr()

### ДС8: Креирање албума

**ДС7: Дијаграм секвенци случај коришћења – Креирање албума**

1. Главни уредник **позива** систем да креира албум. (АПСО)
2. Систем **приказује** главном уреднику албум и поруку: “ Систем је креирао албум“. (ИА)
3. Главни уредник **позива** систем да запамти податке о албуму. (АПСО)
4. Систем **приказује** главном уреднику запамћени албум и поруку: “Систем је запамтио албум“. (ИА)

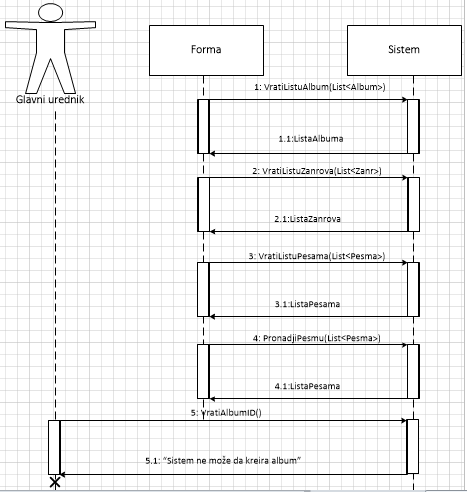


*Слика 22. – Дијаграм секвенци: СК7*

*Креирање албума*

**Алтернативна сценариа**

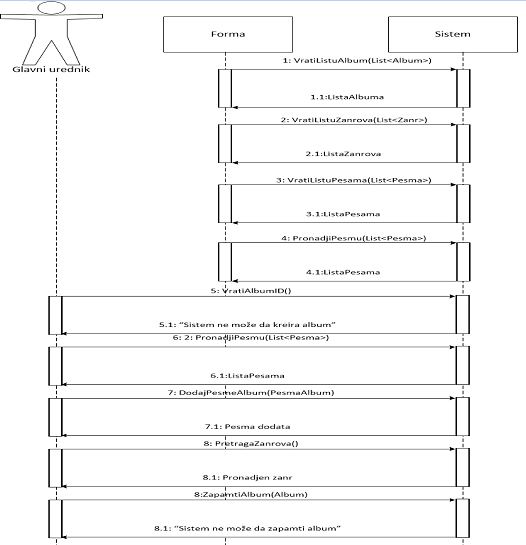
2.1 Уколико систем не може да креира албум он приказује главном уреднику поруку: “Систем не може да креира албум ”. Прекида се извршење сценариа. (ИА)



*Слика 23 - Дијаграм секвенци: СК7*

*Креирање албума – Алтернативни сценарио 1*

4.1 Уколико систем не може да запамти податке о албуму он приказује главном уреднику поруку: “Систем не може да запамти албум”. (ИА)



*Слика 24. - Дијаграм секвенци: СК7*

*Креирање албума – Алтернативни сценарио 2*

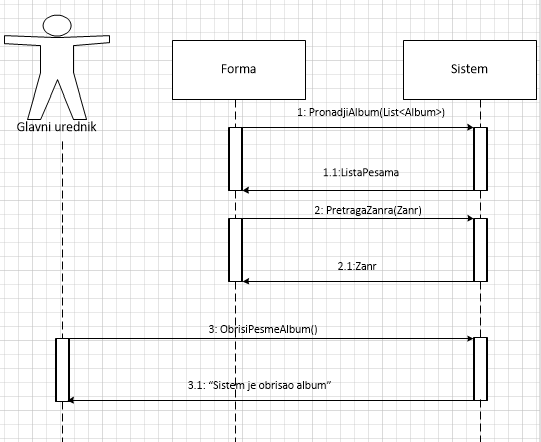
Са наведених секвенцних дијаграма уочавају се 4 системске операције које треба пројектовати:

1. Сигнал VratiListuAlbum(List<Album>)
2. Сигнал VratiListuZanrova(List<Zanr>)
3. Сигнал VratiListuPesama(List<Pesme>)
4. Сигнал PronadjiPesmu(List<Pesme>)
5. Сигнал VratiAlbumID()
6. Сигнал DodajPesmuAlbum(PesmaAlbum)
7. Сигнал PretragaZanrova()
8. СигналZapamtiAlbum(Album)

### ДС9: Брисање албума

**ДС10: Дијаграм секвенци случај коришћења – Брисање албума**

1. Главни уредник **позива** систем да обрише албум. (АПСО)
2. Систем **приказује** главном уреднику поруку: “Систем је обрисао албум”. (ИА)

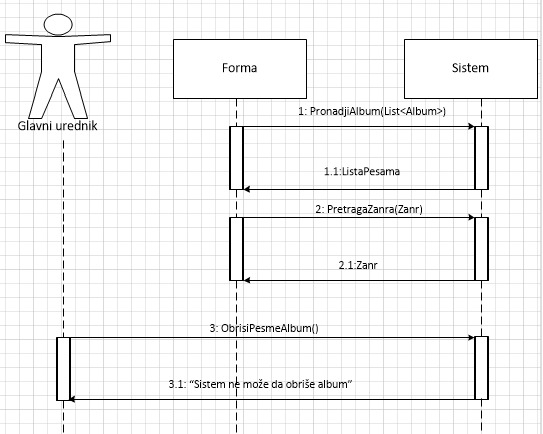


*Слика 25. – Дијаграм секвенци: СК10*

*Брисање албума*

**Алтернативна сценариа**

4.1. Уколико систем не може да обрише албум он приказује главном уреднику поруку: “Систем не може да обрише албум”. (ИА)



*Слика 26. - Дијаграм секвенци: СК9*

*Брисање албума – Алтернативни сценарио 1*

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

1. Сигнал PronadjiAlbum(List<Album>)
2. Сигнал PretragaZanra(Album)
3. Сигнал ObrisiPesmeAlbum()

Као резултат анализе сценарија добијено је укупно 26 системских операција које је потребно пројектовати:

1. Сигнал VratiIzvodjacID(Izvodjac)
2. Сигнал VratiListuIzvodjaca(List<Izvodjac>)
3. Сигнал PronadjiIzvodjaca(List<Izvodjac>))
4. Сигнал PretragaIzvodjaca(Izvodjac)
5. Сигнал IzmeniIzvodjaca(Izvodjac)
6. Сигнал ZapamtiIzvodjaca(Izvodjac)
7. Сигнал VratiPesmaID(Pesma)
8. Сигнал VratiListuPesme(List<Izvodjac>)
9. Сигнал PronadjiPesmu(List<Izvodjac>)
10. Сигнал PretragaPesme (Pesma)
11. Сигнал IzmeniPesmu(Pesma)
12. Сигнал Zapamti Pesmu(Pesma)
13. Сигнал VratiZanrID(Zanr)
14. Сигнал VratiListuZanrova(List<Zanr>)
15. Сигнал PretragaZanra(Zanr)
16. Сигнал ZapamtiZanr(Zanr)
17. Сигнал VratiAlbumID(Album)
18. Сигнал VratiListuAlbuma(List<Album>)
19. Сигнал VratiPesmaAlbumID(PesmaAlbum)
20. Сигнал PretragaAlbuma(Album)
21. Сигнал PronadjiAlbum(List<Album>)
22. Сигнал PretragaPesmeAlbuma(PesmaAlbum)
23. Сигнал PronadjiPesmeAlbuma(List<PesmaAlbum>)
24. Сигнал DodajPesmaAlbuma(PesmaAlbum)
25. Сигнал ObrisiPesmeAbum(PesmaAlbum)
26. Сигнал ZapamtiAlbum(Album)

## 

## 2.2. Понашање софтверског система – дефинисање уговора о системским операцијама

**Уговор УГ1:IzmeniIzvođača**

Операција:IzmeniIzvodjaca(Izvodjac) :сигнал

Веза са СК: СК1

Предуслови: -

Постуслови: -

**Уговор УГ2: ZapamtiIzvođača**

Операција: ZapamtiIzvođača(Izvodjac) :сигнал

Веза са СК: СК1, СК3

Предуслови: Вредносна и структурна ограничења над објектом извођач морају бити задовољена

Постуслови: Подаци о извођачу су запамћени

**Уговор УГ3: PretragaIzvođača**

Операација: PretragaIzvođača(Izvodjac) :сигнал

Веза са СК: СК2

Предуслови: -

Постуслови:-

**Уговор УГ4: PronađiIzvođača**

Операција: PronađiIzvođača(List<Izvodjac>) :сигнал

Веза са СК:CK3

Предуслови: -  
Постуслови: -

**Уговор УГ5:VratiIzvođačID**

Операција: VratiIzvodjacID(Izvodjac) :сигнал

Веза са СК:CK3

Предуслови: -  
Постуслови: -

**Уговор УГ6:VratiListuIzvođača**

Операција: VratiListuIzvodjaca(Izvodjac) :сигнал

Веза са СК:CK3

Предуслови: -  
Постуслови: -

**Уговор УГ7: IzmeniPesmu**

Операција: IzmeniPesmu(Pesma) :сигнал

Веза са СК:CK4

Предуслови: -  
Постуслови: -

**Уговор УГ8: ZapamtiPesmu**

Операција: ZapamtiPesmu(Pesma) :сигнал

Веза са СК:CK4, СК6

Предуслови:Вредносна и структурна ограничења над објектом песма морају бити задовољена  
Постуслови: Подаци о песми су запамћени

**Уговор УГ9: PretragaPesme**

Операција: PretragaPesme(Pesma) :сигнал

Веза са СК:CK5

Предуслови: -   
Постуслови: -

**Уговор УГ10:PronađiPesmu**

Операција: PronađiPesmu(List<Pesma>) :сигнал

Веза са СК:CK6

Предуслови: -  
Постуслови: -

**Уговор УГ11: VratiPesmuID**

Операција: VratiPesmuID(Pesma) :сигнал

Веза са СК:CK6

Предуслови: -  
Постуслови: -

**Уговор УГ12:VratiListuPesama**

Операција: VratiListuPesama(List<Pesma>) :сигнал

Веза са СК:CK7

Предуслови: -  
Постуслови: -

**Уговор УГ13: PretragaŽanra**

Операција: PretragaZanra(Zanr) :сигнал

Веза са СК:CK5

Предуслови: -   
Постуслови: -

**Уговор УГ14: VratiŽanrID**

Операција: VratiZanrID(Zanr) :сигнал

Веза са СК:CK6

Предуслови: -  
Постуслови: -

**Уговор УГ15:VratiListuŽanrova**

Операција: VratiListuZanrova(List<Zanr>) :сигнал

Веза са СК:CK7

Предуслови: -  
Постуслови: -

**Уговор УГ16: ZapamtiŽanr**

Операција: ZapamtiZanr(Zanr) :сигнал

Веза са СК:CK4, СК6

Предуслови:Вредносна и структурна ограничења над објектом жанр морају бити задовољена  
Постуслови: Подаци о жанру су запамћени

**Уговор УГ17: DodajPesmaAlbum**

Операција: DodajPesmaAlbum(PesmaAlbum) :сигнал

Веза са СК:CK7

Предуслови: -  
Постуслови: -

**Уговор УГ18: ZapamtiAlbum**

Операција: ZapamtiAlbum(Album) :сигнал

Веза са СК:CK7, СК9

Предуслови:Вредносна и структурна ограничења над објектом албум морају бити задовољена

Постуслови:Подаци о албуму су запамћени

**Уговор УГ19:PretragaAlbuma**

Операција: PretragaAlbuma(Album) :сигнал

Веза са СК:CK8

Предуслови:-  
Постуслови: -

**Уговор УГ20:PronađiAlbum**

Операција: PronadjiAlbum(List<Album>) :сигнал

Веза са СК:CK9

Предуслови:-  
Постуслови: -

**Уговор УГ21:VratiAlbumID**

Операција: VratiAlbumID(Album) :сигнал

Веза са СК:CK9

Предуслови:-  
Постуслови: -

**Уговор УГ22: VratiListuAlbuma**

Операција: VratiListuAlbuma(Album) :сигнал

Веза са СК:CK10

Предуслови:-  
Постуслови: -

**Уговор УГ23:ObrišiPesmaAlbum**

Операција: ObrisiPesmaAlbum(PesmaAlbum) :сигнал

Веза са СК:CK10

Предуслови:Структурно ограничење над објектом ПесмаАлбум мора бити задовољено  
Постуслови: ПесмаАлбум је обрисан

**Уговор УГ24:PretragaPesmaAlbuma**

Операција: PretragaPesmaAlbuma(PesmaAlbum) :сигнал

Веза са СК:CK8

Предуслови:-  
Постуслови: -

**Уговор УГ25:PronađiAlbum**

Операција: PronadjiPesmaAlbum(List<PesmaAlbum>) :сигнал

Веза са СК:CK9

Предуслови:-  
Постуслови: -

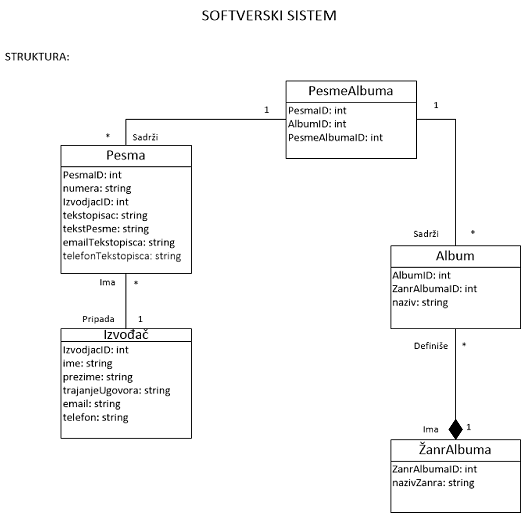
**Уговор УГ26:VratiAlbumID**

Операција: VratiPesmaAlbumID(PesmaAlbum) :сигнал

Веза са СК:CK9

Предуслови:-  
Постуслови: -

## 2.3. Структура софтверског система – концептуални (доменски) модел



## 2.4. Структура софтверског система – релациони модел

Izvodjac(IzvodjacID, ime, prezime, trajanjeUgovora, email, telefon)

Album(AlbumID, naziv, *ZanrID*)

Pesma(PesmaID, numera, *IzvodjacID,* tekstopisac, tekstPesme, emailTekstopisca, telefonTekstopisca)

Zanr(ZanrID, nazivZanra)

PesmeAlbum(PesmaAlbumID, *PesmaID*, *AlbumID*,)

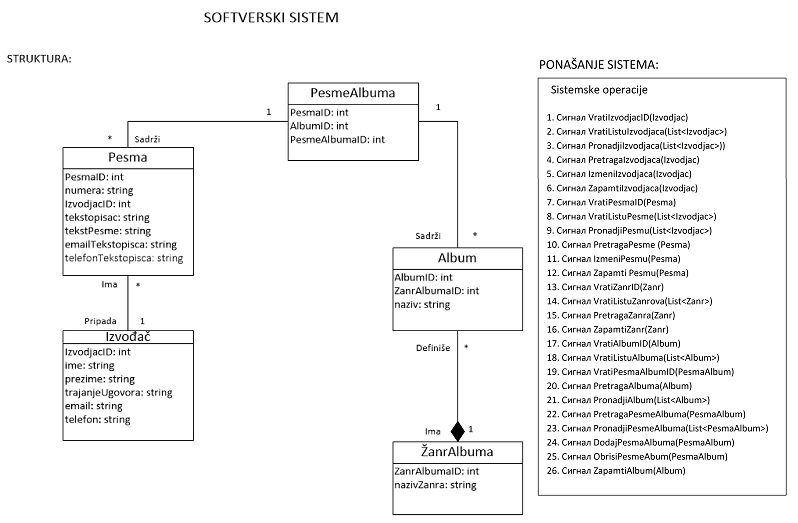
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Tabela Izvodjac | | Prosto vrednosno ograničenje | | Složeno vrednosno ograničenje | | Strukturno ograničenje |
| Atributi | Ime | Tip atributa | Vrednost atributa | Međuzavinost atributa jedne tabele | Međuzavisnost atributa više tabela | INSERT  UPDATE CASCADE Album  DELETE RESTRICTED Album |
| IzvodjacID | int | not null and >0 | \ | \ |
| Ime | String | not null | \ | \ |
| prezime | String | not null | \ | \ |
| trajanjeUgovora | date | not null | \ | \ |
| email | String | not null | \ | \ |
| telefon | String | not null | \ | \ |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Tabela Žanr | | Prosto vrednosno ograničenje | | Složeno vrednosno ograničenje | | Strukturno ograničenje |
| Atributi | Ime | Tip atributa | Vrednost atributa | Međuzavinost atributa jedne tabele | Međuzavisnost atributa više tabela | INSERT RESTRICTED Izvođač  UPDATE RESTRICTED Izvođač  DELETE / |
| ZanrID | int | not null and >0 | \ | \ |
| nazivZanra | int | not null | \ | \ |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Tabela Album | | Prosto vrednosno ograničenje | | Složeno vrednosno ograničenje | | Strukturno ograničenje |
| Atributi | Ime | Tip atributa | Vrednost atributa | Međuzavinost atributa jedne tabele | Međuzavisnost atributa više tabela | INSERT RESTRICTED Izvodjac  UPDATE RESTRICTED Izvodjac  DELETE / |
| AlbumID | int | not null and >0 | \ | \ |
| naziv | String | not null | \ | \ |
| ZanrID | int | not null | \ | \ |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Tabela Pesma | | Prosto vrednosno ograničenje | | Složeno vrednosno ograničenje | | Strukturno ograničenje |
| Atributi | Ime | Tip atributa | Vrednost atributa | Međuzavinost atributa jedne tabele | Međuzavisnost atributa više tabela | INSERT RESTRICTED  Album, Autor  UPDATE CASCADE Album, Autor |
| PesmaID | int | not null and >0 | \ | \ |
| numera | String | not null | \ | \ |
| IzvodjacID | int | not null | \ | \ |
| tekstopisac | String | not null | \ | \ |
| tekstPesme | String | not null | \ | \ |
| emailTekstopisca | String | not null | \ | \ |
| telefonTekstopisca | String | not null | \ | \ |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Tabela PesmeAlbuma | | Prosto vrednosno ograničenje | | Složeno vrednosno ograničenje | | Strukturno ograničenje |
| Atributi | Ime | Tip atributa | Vrednost atributa | Međuzavinost atributa jedne tabele | Međuzavisnost atributa više tabela | INSERT /  UPDATE CASCADE Pesma  DELETE / |
| PesmaAlbumID | int | not null and >0 | \ | \ |
| PesmaID | String | not null | \ | \ |
| AlbumID | String | not null | \ | \ |



# 3 Пројектовање

Фаза пројектовања описује физичку структуру и понашање софтверског система архитектуру софтверског система. Пројектовање архитектуре софтверског система обухвата пројектовање корисничког интерфејса, апликационе логике и складишта података. Пројектовање корисничког интерфејса обухвата пројектовање екранских форми и контролера корисничког интерфејса. У оквиру апликационе логике се пројектују контролер апликационе логике, пословна логика И брокер базе података. Пројектовање пословне логика обухвата пројектовање логичке структуре и понашање софтверског система.

## 3.1 Архитектура софтверског система

У оквиру фазе пројектовања описује се физичка структура и понашање софтверског система тј. архитектура софтверског система. У овом семинарском раду је коришћена класична трослојна архитектура, која се састоји од:

• Корисничког интерфејса

• Апликационе логике

• Складишта података

Korisnički interfejs

Aplikaciona logika

Skladište podataka

**I sloj**

**II sloj**

**III sloj**

**Softverski sistem**

*Слика 2 : Трослојна архитектура*

Skladište podataka

**Korisnički interfejs**

Ekranska forma

KontrolerKI

**Aplikaciona logika**

KontrolerAL

Poslovna logika

Database broker

*Слика 3 : Трослојна архитектура*

*– детаљнији приказ*

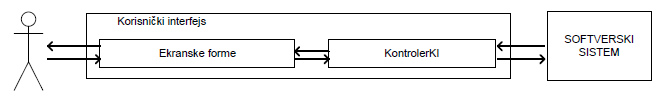
## 

## 3.2. Пројектовање корисничког интерфејса

Контролер корисничког интерфејса и екранске форме не представљају софтверски систем, већ реализацију улаза и излаза из софтверског система. Контролер корисничког интерфејса има улога да конвертује податке у графичке елементе екранских форми и да податке или пошаље у софтверски систем или да прими податке из софтверског система и исте конвертује у елементе екранске форме.

Кориснички интерфејс се састоји из:

• Екранске форме,

• Контролера корисничког интерфејса.

## 3.3 Пројектовање екранских форми

**СК1: Случај коришћења – Креирање извођача**

**Назив СК**

Креирање извођача

**Актори СК**

Главни уредник

**Учесници СК**

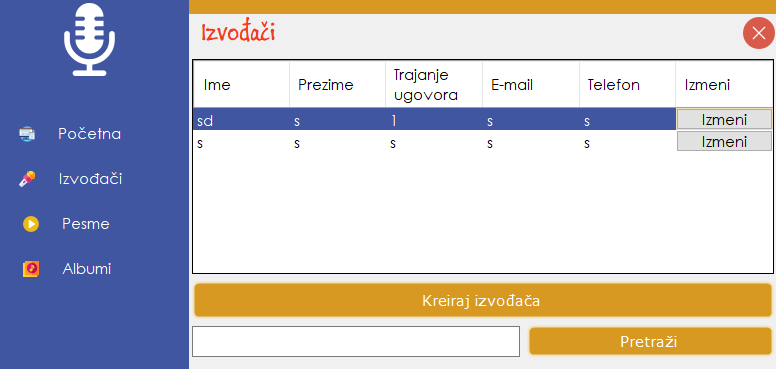
Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник је улогован под својом шифром. Систем приказује форму за рад са извођачима.

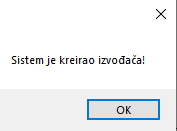


**Основни сценарио СК**

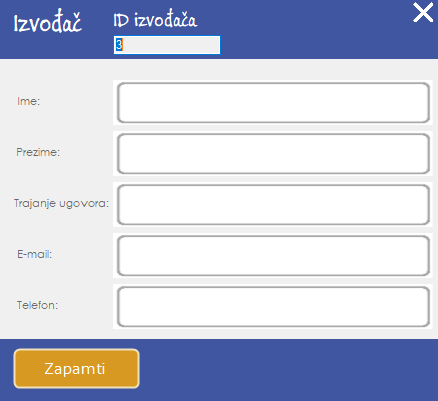
1. Главни уредник **позива** систем да креира извођача. (АПСО)



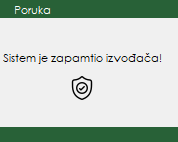
1. Систем креира извођача.(СО)
2. Систем **приказује** главном уреднику извођача и поруку: “Систем је креирао извођача“. (ИА)



1. Главни уредник уноси податке у систем.(АПУСО)
2. Главни уредник **контролише** да ли је коректно унео податке у систем. (АНСО)
3. Главни уредник **позива** систем да запамти податке о извођачу. (АПСО)

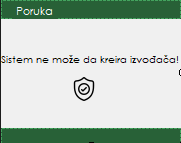


1. Систем **памти** податке о извођачу. (СО)
2. Систем **приказује** главном уреднику запамћени извођач и поруку: “Систем је запамтио извођача“. (ИА)

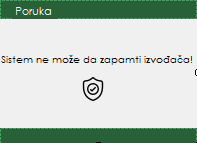


**Алтернативна сценариа**

* 1. Уколико систем не може да креира извођача он приказује главном уреднику поруку: “Систем не може да креира извођача”. Прекида се извршење сценариа. (ИА)



8.1 Уколико систем не може да запамти податке о извођачу он приказује главном уреднику поруку: “Систем не може да запамти извођача”. (ИА)



**СК2: Случај коришћења – Претраживање извођача**

**Назив СК**

Претраживање извођача

**Актори СК**

Главни уредник

**Учесници СК**

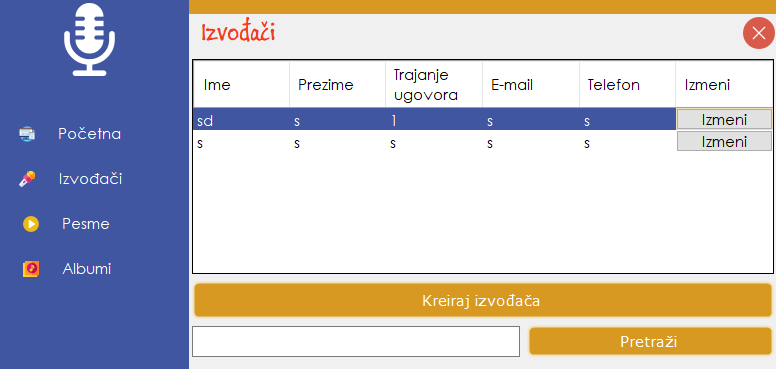
Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник je улогован под својом шифром. Систем приказује форму за рад са извођачима.

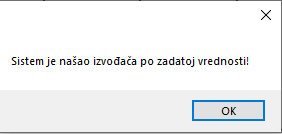


**Основни сценарио СК**

1. Главни уредник **уноси** вредност по којој претражује извођача. (АПУСО)
2. Главни уредник **позива** систем да нађе извођача по задатој вредности. (АПСО)

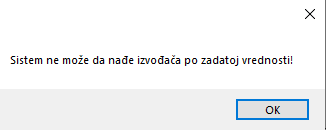


1. Систем **тражи** извођача по задатој вредности. (СО)
2. Систем **приказује** главном уреднику податке о извођачу и поруку: “Систем је нашао извођача по задатој вредности”.(ИА)



**Алтернативна сценариа**

* 1. Уколико систем не може да нађе извођача он приказује главном уреднику поруку: “Систем не може да нађе извођача по задатој вредности”. (ИА)



**СК3: Случај коришћења – Измена извођача**

**Назив СК**

Измена извођача

**Актори СК**

Главни уредник

**Учесници СК**

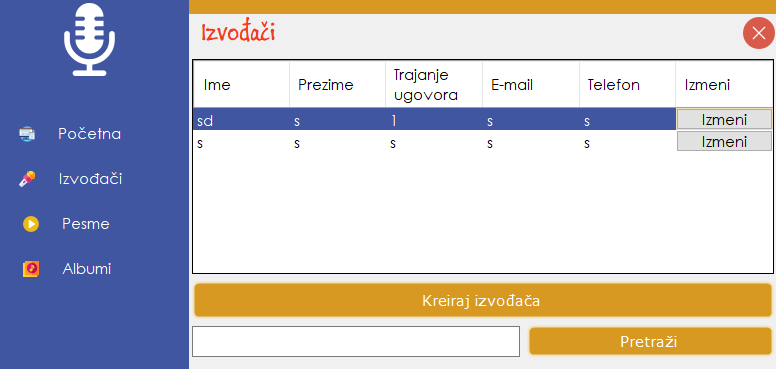
Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник je улогован под својом шифром. Систем приказује форму за рад са извођачем.

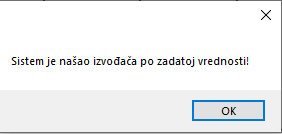


**Основни сценарио СК**

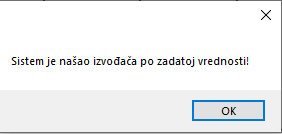
1. Главни уредник **уноси** вредност по којој претражује извођаче. (АПУСО)
2. Главни уредник **позива** систем да нађе извођаче по задатој вредности. (АПСО)



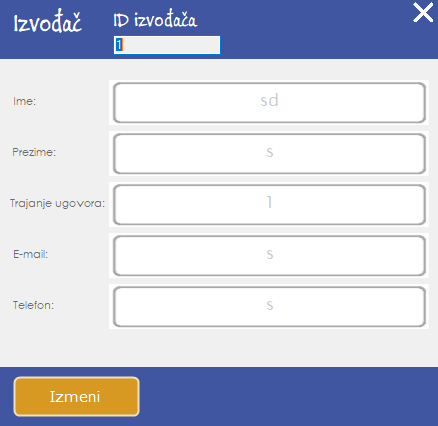
1. Систем **тражи** извођаче по задатој вредности. (СО)
2. Систем **приказује** главном уреднику извођача и поруку: “Систем је нашао извођача по задатој вредности”. (ИА)



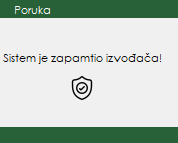
1. Систем **бира** извођача. (АПУСО)
2. Главни уредник **позива** систем да нађе изабраног извођача. (АПСО)
3. Систем **тражи** изабраног извођача. (СО)
4. Систем **приказује** главном уреднику извођача и поруку: “Систем је нашао извођача по задатој вредности”. (ИА)



1. Главни уредник **уноси(мења)** податке о извођачу. (АПУСО)

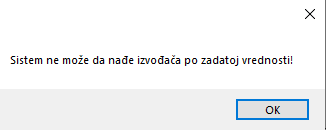


1. Главни уредник **контролише** да ли је коректно унео податке о извођачу. (АНСО)
2. Главни уредник **позива** систем да запамти податке о извођачу. (АПСО)
3. Систем **памти** податке о извођачу. (СО)
4. Систем **приказује** главном уреднику запамћени извођач и поруку: “Систем је запамтио извођача”. (ИА)

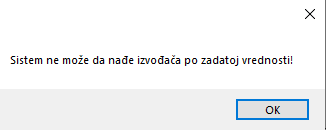


**Алтернативна сценариа**

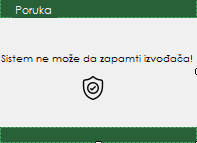
* 1. Уколико систем не може да нађе извођача он приказује главном уреднику поруку: “Систем не може да нађе извођача по задатој вредности”. Прекида се извршење сценариа. (ИА)



* 1. Уколико систем не може да нађе извођача он приказује главном уреднику поруку: “Систем не може да нађе извођача по задатој вредности”. Прекида се извршење сценариа. (ИА)



13.1 Уколико систем не може да запамти податке о извођачу он приказује главном уреднику поруку: “Систем не може да запамти извођача”. (ИА)



**СК4: Случај коришћења – Креирање песме**

**Назив СК**

Креирање песме

**Актори СК**

Главни уредник

**Учесници СК**

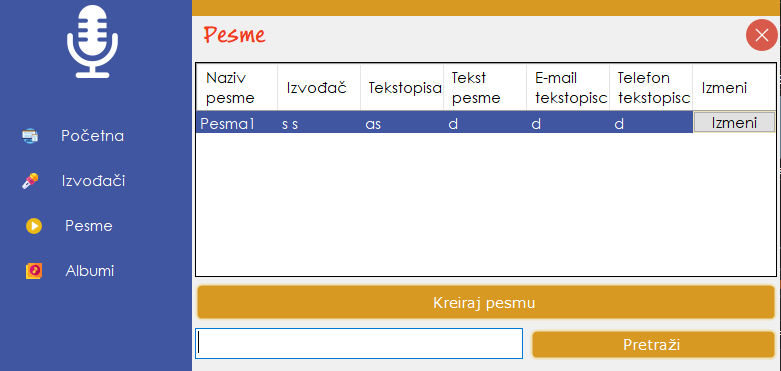
Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник је улогован под својом шифром. Систем приказује форму за рад са песмама.

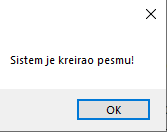


**Основни сценарио СК**

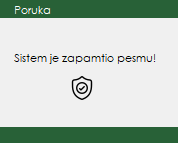
1. Главни уредник **позива** систем да креира песму. (АПСО)



1. Систем креира песму.(СО)
2. Систем **приказује** главном уреднику песму и поруку: “Систем је креирао песму“. (ИА)

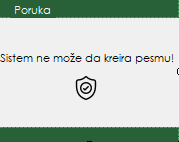


1. Главни уредник уноси податке у систем.(АПУСО)
2. Главни уредник **контролише** да ли је коректно унео податке у систем. (АНСО)
3. Главни уредник **позива** систем да запамти податке о песми. (АПСО)
4. Систем **памти** податке о песми. (СО)
5. Систем **приказује** главном уреднику запамћену песму и поруку: “Систем је запамтио песму“. (ИА)

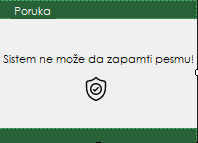


**Алтернативна сценариа**

* 1. Уколико систем не може да креира песму он приказује главном уреднику поруку: “Систем не може да креира песму”. Прекида се извршење сценариа. (ИА)



* 1. Уколико систем не може да запамти податке о песми он приказује главном уреднику поруку: “Систем не може да запамти песму”. (ИА)



**СК5: Случај коришћења – Претраживање песме**

**Назив СК**

Претраживање песме

**Актори СК**

Главни уредник

**Учесници СК**

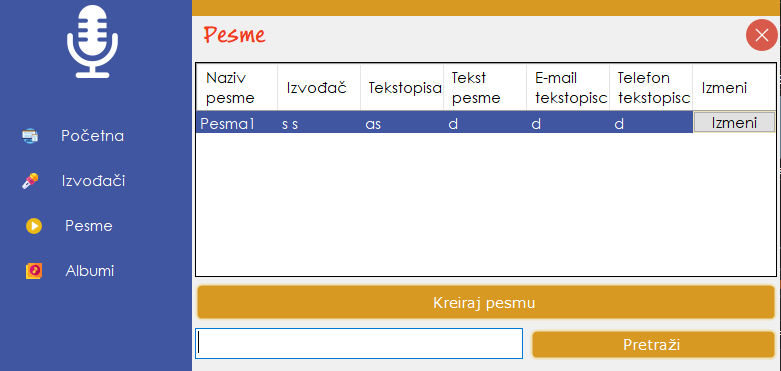
Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник je улогован под својом шифром. Систем приказује форму за рад са песмама.

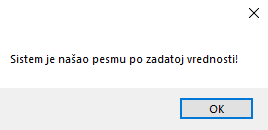


**Основни сценарио СК**

1. Главни уредник **уноси** вредност по којој претражује песму. (АПУСО)
2. Главни уредник **позива** систем да нађе песму по задатој вредности. (АПСО)

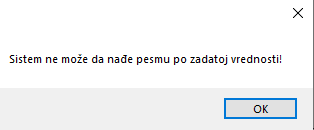


1. Систем **тражи** песму по задатој вредности. (СО)
2. Систем **приказује** главном уреднику податке о песми и поруку: “Систем је нашао песму по задатој вредности”.(ИА)



**Алтернативна сценариа**

* 1. Уколико систем не може да нађе песму он приказује главном уреднику поруку: “Систем не може да нађе песму по задатој вредности”. (ИА)



**СК6: Случај коришћења – Измена песме**

**Назив СК**

Измена песме

**Актори СК**

Главни уредник

**Учесници СК**

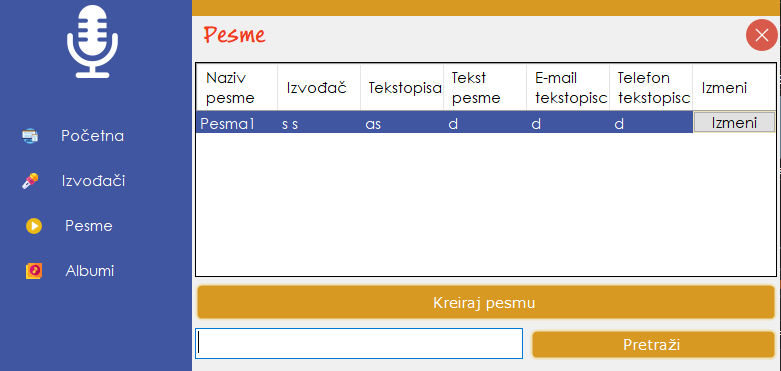
Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник je улогован под својом шифром. Систем приказује форму за рад са песмама.

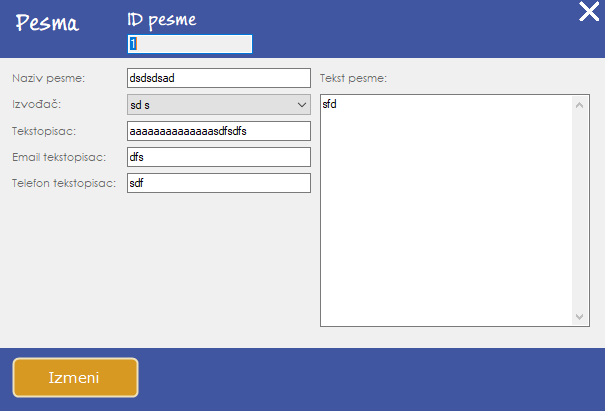


**Основни сценарио СК**

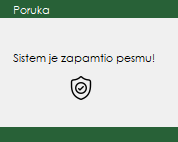
1. Главни уредник **позива** систем да нађе песму. (АПСО)



1. Систем **тражи** песму. (СО)
2. Главни уредник **бира** песму. (АПУСО)
3. Главни уредник **позива** систем да нађе изабрану песму. (АПСО)
4. Систем **тражи** изабрану песму. (СО)
5. Главни уредник **уноси(мења)** податке о песми. (АПУСО)

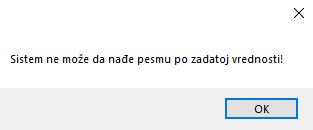


1. Главни уредник **контролише** да ли је коректно унео податке о песми. (АНСО)
2. Главни уредник **позива** систем да измени податке о песми. (АПСО)
3. Систем **памти** податке о песми. (СО)
4. Систем **приказује** главном уреднику измењену песму и поруку: “Систем је запамтио песму”. (ИА)

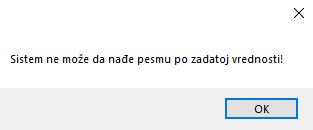


**Алтернативна сценариа**

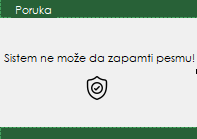
* 1. Уколико систем не може да нађе песму он приказује главном уреднику поруку: “Систем не може да нађе песму по задатој вредности”. Прекида се извршење сценариа. (ИА)



* 1. Уколико систем не може да нађе песму он приказује главном уреднику поруку: “Систем не може да нађе песму по задатој вредности”. Прекида се извршење сценариа. (ИА)



13.1 Уколико систем не може да запамти податке о песми он приказује главном уреднику поруку: “Систем не може да запамти песму”. (ИА)



**СК7: Случај коришћења – Креирање жанра**

**Назив СК**

Ккреирање жанра

**Актори СК**

Главни уредник

**Учесници СК**

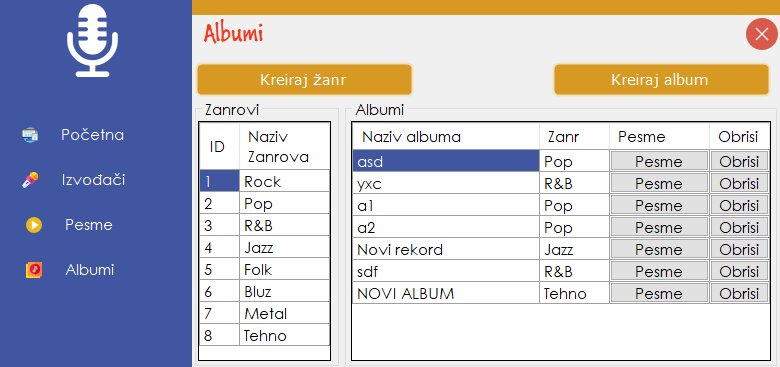
Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник je улогован под својом шифром. Систем приказује форму за рад са жанровима. Учитана је листа жанрова.

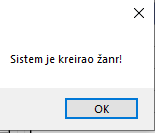
. 

**Основни сценарио СК**

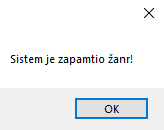
1. Главни уредник **позива** систем да креира жанр. (АПСО)



1. Систем креира жанр.(СО)
2. Систем **приказује** главном уреднику жанр и поруку: “Систем је креирао жанр“. (ИА)

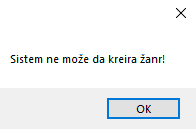


1. Главни уредник уноси податке у систем.(АПУСО)
2. Главни уредник **контролише** да ли је коректно унео податке у систем. (АНСО)
3. Главни уредник **позива** систем да запамти податке о жанру. (АПСО)
4. Систем **памти** податке о жанру. (СО)
5. Систем **приказује** главном уреднику запамћени жанр и поруку: “Систем је запамтио жанр“. (ИА)

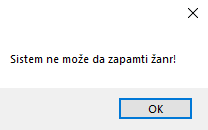


**Алтернативна сценариа**

* 1. Уколико систем не може да креира жанр он приказује главном уреднику поруку: “Систем не може да креира жанр ”. Прекида се извршење сценариа. (ИА)



* 1. Уколико систем не може да запамти податке о жанру он приказује главном уреднику поруку: “Систем не може да запамти жанр”. (ИА)



**СК8: Случај коришћења – Креирање албума**

**Назив СК**

Ккреирањеалбума

**Актори СК**

Главни уредник

**Учесници СК**

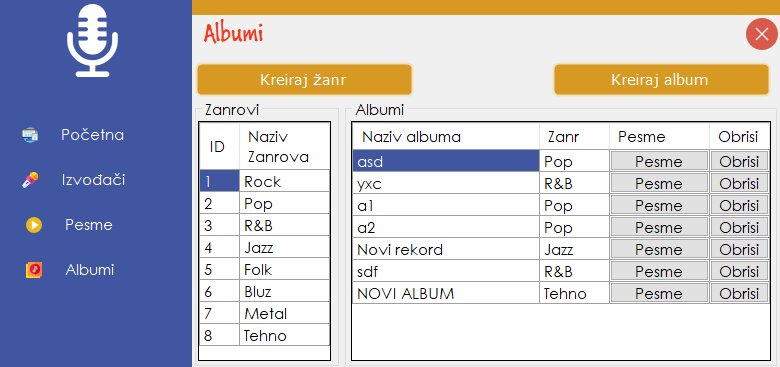
Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник je улогован под својом шифром. Систем приказује форму за рад са албумима. Учитана је листа песама и извођача.

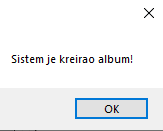
. 

**Основни сценарио СК**

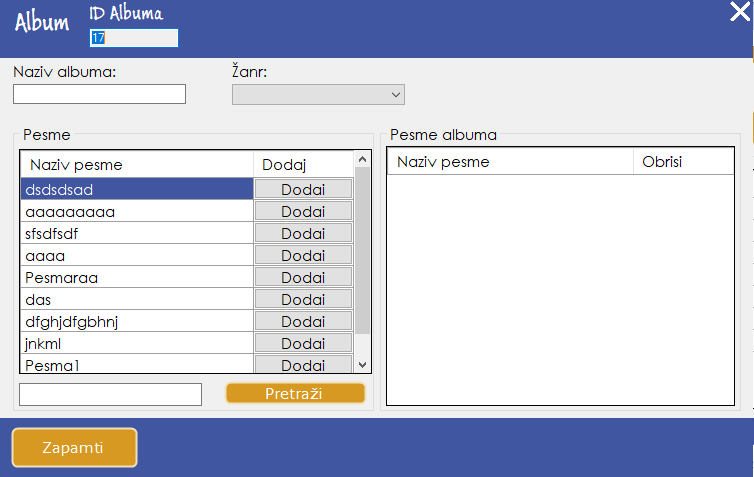
1. Главни уредник **позива** систем да креира албум. (АПСО)



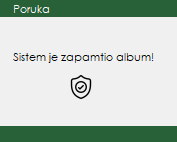
1. Систем креира албум.(СО)
2. Систем **приказује** главном уреднику албум и поруку: “Систем је креирао албум“. (ИА)



1. Главни уредник **уноси** вредност по којој претражује песму. (АПУСО)
2. Систем **тражи** песму по задатој вредности. (СО)
3. Главни уредник **уноси** жанр и песме у систем.(АПУСО)
4. Главни уредник **контролише** да ли је коректно унео податке у систем. (АНСО)
5. Главни уредник **позива** систем да запамти податке о албуму. (АПСО)

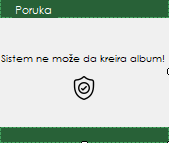


1. Систем **памти** податке о албуму. (СО)
2. Систем **приказује** главном уреднику запамћени албум и поруку: “Систем је запамтио албум“. (ИА)

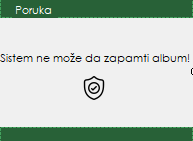


**Алтернативна сценариа**

* 1. Уколико систем не може да креира албум он приказује главном уреднику поруку: “Систем не може да креира албум ”. Прекида се извршење сценариа. (ИА)



* 1. Уколико систем не може да запамти податке о албуму он приказује главном уреднику поруку: “Систем не може да запамти албум”. (ИА)



**СК9: Случај коришћења – Брисање албума**

**Назив СК**

Брисање албума

**Актори СК**

Главни уредник

**Учесници СК**

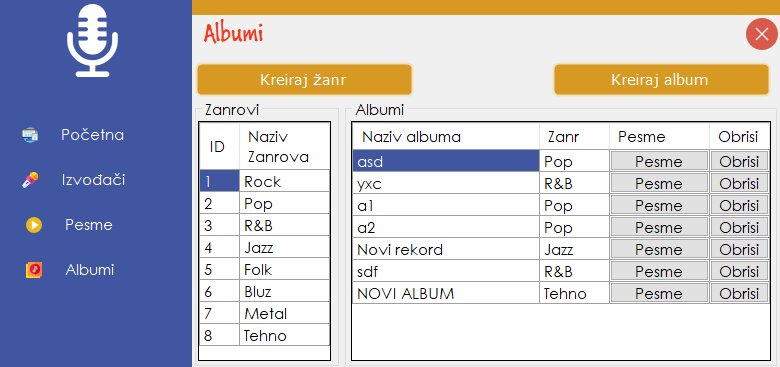
Главни уредник и систем (програм)

**Предуслов**: Систем је укључен и главни уредник je улогован под својом шифром. Систем приказује форму за рад са албумом. Излистана је листа песама и извођача.

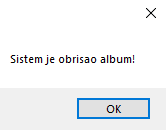


**Основни сценарио СК**

1. Главни уредник **позива** систем да обрише албум. (АПСО)

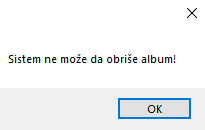


1. Систем **брише** албум. (СО)
2. Систем **приказује** главном уреднику поруку: “Систем је обрисао албум”. (ИА)



**Алтернативна сценариа**

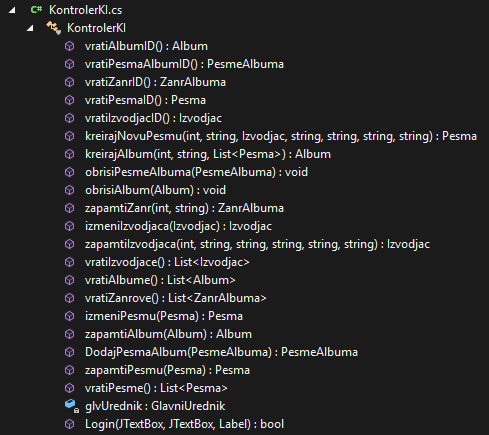
* 1. Уколико систем не може да обрише албум он приказује главном уреднику поруку: “Систем не може да обрише албум”. (ИА)

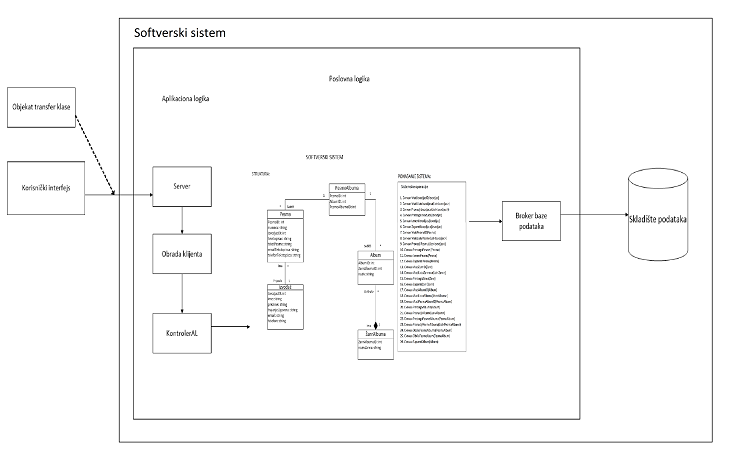


## 3.4 Пројектовање контролера корисничког интерфејса

Апликациона логика служи за описивање структуре и понашања софтверског система и пројектује се независно од корисничког интерфејса и обрнуто. Другим речима, апликациона логика (која представља Модел у МВЦ патерну) нема знања о томе где се налази кориснички интерфејс (која представља Виењ у МВЦ патерну).

Контролер је одговоран да прихвати захтев за извршење системске операције од клијента и да га проследи до пословне логике која је одговорна за извршење системске операције.





## *Дијаграм: Слојеви софтверског система*

## 3.5 Пројектовање апликационе логике

3.5.1. Контролер апликационе логике

Контролер апликационе логике треба да подигне серверски сокет који ће да ослушкује мрежу. Када клијент (клијентски сокет) успостави конекцију са контролером (серверским сокетом), тада контролер треба да генерише нит која ће успоставити двосмерну везу са клијентом (улазну и излазну). Слање и примање података од клијента се остварује преко сокета.

Клијент шаље захтев за извршење неке од СО до одговарајуће нити (коју смо назвали “ОбрадаКлијента”), која је повезана са тим клијентом. “ОбрадаКлијента” прима захтев и даље га преусмерава до класа које су одговорне за извршење СО. Након извршења СО резултат се враћа до апликационе логике, односно до “ОбрадаКлијента”, која тај резултат шаље назад до клијента.

Пример специфичног контролера апликационе логике

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Domen;

using System.Net;

using System.Net.Sockets;

using System.Runtime.Serialization.Formatters.Binary;

using System.Threading;

using SistemskeOperacije;

namespace Server

{

public class ObradaKlijenta

{

private NetworkStream tok;

BinaryFormatter formater;

public ObradaKlijenta(NetworkStream tok)

{

this.tok = tok;

formater = new BinaryFormatter();

}

internal void obradi()

{

try

{

bool kraj = false;

while (!kraj)

{

TransferKlasa transfer = formater.Deserialize(tok) as TransferKlasa;

switch (transfer.operacija)

{

case Operacija.LOGIN:

OpstaSistemskaOperacija operacija = Kontroler.Instance.Login(transfer.TransferObjekat as GlavniUrednik);

TransferKlasa odgovor = new TransferKlasa

{

TransferObjekat = operacija.Rezultat,

Signal = operacija.Signal

};

formater.Serialize(tok, odgovor);

break;

case Operacija.KREIRAJ\_ALBUM:

OpstaSistemskaOperacija kreirajAlbumSO = Kontroler.Instance.KreirajAlbum(transfer.TransferObjekat as Album);

try

{

transfer.Signal = kreirajAlbumSO.Signal;

transfer.Rezultat = kreirajAlbumSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.VRATI\_ALBUM\_ID:

OpstaSistemskaOperacija vratiAlbumIDSO = Kontroler.Instance.VratiAlbumID(transfer.TransferObjekat as Album);

try

{

transfer.Signal = vratiAlbumIDSO.Signal;

transfer.Rezultat = vratiAlbumIDSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.VRATI\_PESMA\_ALBUM\_ID:

OpstaSistemskaOperacija vratiPesmaAlbumIDSO = Kontroler.Instance.VratiPesmaAlbumID(transfer.TransferObjekat as PesmeAlbuma);

try

{

transfer.Signal = vratiPesmaAlbumIDSO.Signal;

transfer.Rezultat = vratiPesmaAlbumIDSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.VRATI\_ZANR\_ID:

OpstaSistemskaOperacija vratiZanrIDSO = Kontroler.Instance.VratiZanrID(transfer.TransferObjekat as ZanrAlbuma);

try

{

transfer.Signal = vratiZanrIDSO.Signal;

transfer.Rezultat = vratiZanrIDSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.VRATI\_PESMA\_ID:

OpstaSistemskaOperacija vratiPesmaIDSO = Kontroler.Instance.VratiPesmaID(transfer.TransferObjekat as Pesma);

try

{

transfer.Signal = vratiPesmaIDSO.Signal;

transfer.Rezultat = vratiPesmaIDSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.VRATI\_IZVODJAC\_ID:

OpstaSistemskaOperacija vratiIZvodjacIDSO = Kontroler.Instance.VratiIzvodjacID(transfer.TransferObjekat as Izvodjac);

try

{

transfer.Signal = vratiIZvodjacIDSO.Signal;

transfer.Rezultat = vratiIZvodjacIDSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.OBRISI\_PESME\_ALBUM:

OpstaSistemskaOperacija ObrisiPesmeAlbumSO = Kontroler.Instance.ObrisiPesmeAlbum(transfer.TransferObjekat as PesmeAlbuma);

OpstaSistemskaOperacija ObrisiAlbumSO = Kontroler.Instance.ObrisiAlbum((transfer.TransferObjekat as PesmeAlbuma).Album);

try

{

transfer.Signal = ObrisiPesmeAlbumSO.Signal;

transfer.Rezultat = ObrisiPesmeAlbumSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.PRETRAGA\_ALBUMA:

OpstaSistemskaOperacija pretragaAlbumaSO = Kontroler.Instance.PretragaAlbuma(transfer.TransferObjekat as Album);

try

{

transfer.Signal = pretragaAlbumaSO.Signal;

transfer.Rezultat = pretragaAlbumaSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.PRETRAGA\_PESME\_ALBUMA:

OpstaSistemskaOperacija pretragaPesmeAlbumaSO = Kontroler.Instance.PretragaPesmeAlbuma(transfer.TransferObjekat as PesmeAlbuma);

try

{

transfer.Signal = pretragaPesmeAlbumaSO.Signal;

transfer.Rezultat = pretragaPesmeAlbumaSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.PRETRAGA\_ZANRA:

OpstaSistemskaOperacija pretragaZanraSO = Kontroler.Instance.PretragaZanra(transfer.TransferObjekat as ZanrAlbuma);

try

{

transfer.Signal = pretragaZanraSO.Signal;

transfer.Rezultat = pretragaZanraSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.PRETRAGA\_IZVODJACA:

OpstaSistemskaOperacija pretragaIzvodjacaSO = Kontroler.Instance.PretragaIzvodjaca(transfer.TransferObjekat as Izvodjac);

try

{

transfer.Signal = pretragaIzvodjacaSO.Signal;

transfer.Rezultat = pretragaIzvodjacaSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.PRETRAGA\_PESME:

OpstaSistemskaOperacija pretragaPesmeSO = Kontroler.Instance.PretragaPesme(transfer.TransferObjekat as Pesma);

try

{

transfer.Signal = pretragaPesmeSO.Signal;

transfer.Rezultat = pretragaPesmeSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.PRONADJI\_ALBUM:

OpstaSistemskaOperacija pronadjiAlbumSO = Kontroler.Instance.PronadjiAlbum(transfer.TransferObjekat as Album);

try

{

transfer.Signal = pronadjiAlbumSO.Signal;

transfer.Rezultat = pronadjiAlbumSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.PRONADJI\_IZVODJACA:

OpstaSistemskaOperacija pronadjiIzvodjacaSO = Kontroler.Instance.PronadjiIzvodjaca(transfer.TransferObjekat as Izvodjac);

try

{

transfer.Signal = pronadjiIzvodjacaSO.Signal;

transfer.Rezultat = pronadjiIzvodjacaSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.PRONADJI\_PESMU\_ALBUMA:

OpstaSistemskaOperacija pronadjiPesmaAlbumSO = Kontroler.Instance.PronadjiPesmaAlbum(transfer.TransferObjekat as PesmeAlbuma);

try

{

transfer.Signal = pronadjiPesmaAlbumSO.Signal;

transfer.Rezultat = pronadjiPesmaAlbumSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.PRONADJI\_PESMU:

OpstaSistemskaOperacija pronadjiPesmuSO = Kontroler.Instance.PronadjiPesmu(transfer.TransferObjekat as Pesma);

try

{

transfer.Signal = pronadjiPesmuSO.Signal;

transfer.Rezultat = pronadjiPesmuSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.VRATI\_LISTU\_ALBUMA:

OpstaSistemskaOperacija vratiListuAlbumaSO = Kontroler.Instance.VratiListuAlbuma(transfer.TransferObjekat as Album);

try

{

transfer.Signal = vratiListuAlbumaSO.Signal;

transfer.Rezultat = vratiListuAlbumaSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.VRATI\_LISTU\_ZANROVA:

OpstaSistemskaOperacija vratiListuZanrovaSO = Kontroler.Instance.VratiListuZanrova(transfer.TransferObjekat as ZanrAlbuma);

try

{

transfer.Signal = vratiListuZanrovaSO.Signal;

transfer.Rezultat = vratiListuZanrovaSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.VRATI\_LISTU\_IZVODJACA:

OpstaSistemskaOperacija vratiListuIzvodjacaSO = Kontroler.Instance.VratiListuIzvodjaca(transfer.TransferObjekat as Izvodjac);

try

{

transfer.Signal = vratiListuIzvodjacaSO.Signal;

transfer.Rezultat = vratiListuIzvodjacaSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.VRATI\_LISTU\_PESAMA:

OpstaSistemskaOperacija vratiListuPesamaSO = Kontroler.Instance.VratiListuPesama(transfer.TransferObjekat as Pesma);

try

{

transfer.Signal = vratiListuPesamaSO.Signal;

transfer.Rezultat = vratiListuPesamaSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.ZAPAMTI\_ALBUM:

OpstaSistemskaOperacija zapamtiAlbumSO = Kontroler.Instance.ZapamtiAlbum(transfer.TransferObjekat as Album);

try

{

transfer.Signal = zapamtiAlbumSO.Signal;

transfer.Rezultat = zapamtiAlbumSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.DODAJ\_PESMA\_ALBUM:

OpstaSistemskaOperacija dodajPesmaAlbumSO = Kontroler.Instance.DodajPesmaAlbum(transfer.TransferObjekat as PesmeAlbuma);

try

{

transfer.Signal = dodajPesmaAlbumSO.Signal;

transfer.Rezultat = dodajPesmaAlbumSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.ZAPAMTI\_ZANR:

OpstaSistemskaOperacija zapamtiZanrSO = Kontroler.Instance.ZapamtiZanr(transfer.TransferObjekat as ZanrAlbuma);

try

{

transfer.Signal = zapamtiZanrSO.Signal;

transfer.Rezultat = zapamtiZanrSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.IZMENI\_PESMU:

OpstaSistemskaOperacija IzmeniPesmuSO = Kontroler.Instance.IzmeniPesmu(transfer.TransferObjekat as Pesma);

try

{

transfer.Signal = IzmeniPesmuSO.Signal;

transfer.Rezultat = IzmeniPesmuSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.IZMENI\_IZVODJACA:

OpstaSistemskaOperacija IzmeniIzvodjacaSO = Kontroler.Instance.IzmeniIzvodjaca(transfer.TransferObjekat as Izvodjac);

try

{

transfer.Signal = IzmeniIzvodjacaSO.Signal;

transfer.Rezultat = IzmeniIzvodjacaSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.ZAPAMTI\_IZVODJACA:

OpstaSistemskaOperacija zapamtiIzvodjacaSO = Kontroler.Instance.ZapamtiIzvodjaca(transfer.TransferObjekat as Izvodjac);

try

{

transfer.Signal = zapamtiIzvodjacaSO.Signal;

transfer.Rezultat = zapamtiIzvodjacaSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.ZAPAMTI\_PESMU:

OpstaSistemskaOperacija zapamtiPesmuSO = Kontroler.Instance.ZapamtiPesmu(transfer.TransferObjekat as Pesma);

try

{

transfer.Signal = zapamtiPesmuSO.Signal;

transfer.Rezultat = zapamtiPesmuSO.Rezultat;

}

catch (Exception)

{

transfer.Signal = false;

}

formater.Serialize(tok, transfer);

break;

case Operacija.KRAJ: kraj = true;

break;

default:

break;

}

}

}

catch (Exception e)

{

}

}

}

}

3.5.2. Системске операције

У претходној фази, фази анализе, одредили смо уговоре о системским операцијама, при чему смо рекли да један уговор описује понашање једне системске операције, тако што описује ШТА операција треба да ради, али не и како. Сада се у фази пројектовања за сваки од уговора пројектује концептуално решење (реализација) СО. То значи да ћемо за сваку класу одговорну за извршење СО дефинисати КАКО ће се системска операција извршити.

На самом почетку пројектовања СО избећи ћемо аспекте реализације који су везани за конекцију са базом, перзистентност и трансакције.

using Domen;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SistemskeOperacije

{

public abstract class OpstaSistemskaOperacija

{

public object Rezultat { get; set; }

public bool Signal { get; set; }

//public string Poruka { get; set; }

public object IzvrsiSO(IDomenskiObjekat domenskiObjekat)

{

//template izvrsavanja

Broker.BrokerBaze.dajSesiju().OtvoriKonekciju();

Broker.BrokerBaze.dajSesiju().PocniTransakciju();

IzvrsiKonkretnuOperaciju(domenskiObjekat);

if (Signal)

{

Broker.BrokerBaze.dajSesiju().PotvrdiTransakciju();

}

else

{

Broker.BrokerBaze.dajSesiju().PonistiTransakciju();

}

Broker.BrokerBaze.dajSesiju().ZatvoriKonekciju();

return Signal;

}

protected abstract object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat);

}

}

**Уговор УГ1:IzmeniIzvođača**

Операција:IzmeniIzvodjaca(Izvodjac) :сигнал

Веза са СК: СК1

Предуслови: -

Постуслови: -

namespace SistemskeOperacije

{

public class IzmeniIzvodjacaSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Izvodjac izvodjac = (Izvodjac)domenskiObjekat;

izvodjac.Id = Broker.BrokerBaze.dajSesiju().Izmena(domenskiObjekat);

Rezultat = izvodjac.Id;

Signal = true;

return true;

}

catch (Exception)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ2: ZapamtiIzvođača**

Операција: ZapamtiIzvođača(Izvodjac) :сигнал

Веза са СК: СК1, СК3

Предуслови: Вредносна и структурна ограничења над објектом извођач морају бити задовољена

Постуслови: Подаци о извођачу су запамћени

namespace SistemskeOperacije

{

public class ZapamtiIzvodjacaSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Izvodjac izvodjac = (Izvodjac)domenskiObjekat;

izvodjac.Id = Broker.BrokerBaze.dajSesiju().Sacuvaj(domenskiObjekat);

Rezultat = izvodjac.Id;

Signal = true;

return true;

}

catch (Exception)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ3: PretragaIzvođača**

Операација: PretragaIzvođača(Izvodjac) :сигнал

Веза са СК: СК2

Предуслови: -

Постуслови:-

namespace SistemskeOperacije

{

public class PretragaIzvodjacaSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Izvodjac izvodjac = (Izvodjac)domenskiObjekat;

Rezultat = Broker.BrokerBaze.dajSesiju().VratiTacanObjekat(izvodjac);

Signal = true;

return true;

}

catch (Exception e)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ4: PronađiIzvođača**

Операција: PronađiIzvođača(List<Izvodjac>) :сигнал

Веза са СК:CK3

Предуслови: -  
Постуслови: -

namespace SistemskeOperacije

{

public class PronadjiIzvodjacaSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Izvodjac izvodjac = (Izvodjac)domenskiObjekat;

Rezultat = Broker.BrokerBaze.dajSesiju().VratiPoKriterijumu(izvodjac).OfType<Izvodjac>().ToList<Izvodjac>();

Signal = true;

return true;

}

catch (Exception e)

{

Signal = true;

return true;

}

}

}

}

**Уговор УГ5:VratiIzvođačID**

Операција: VratiIzvodjacID(Izvodjac) :сигнал

Веза са СК:CK3

Предуслови: -  
Постуслови: -

namespace SistemskeOperacije

{

public class VratiIzvodjacIDSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Izvodjac izvodjac = (Izvodjac)domenskiObjekat;

izvodjac.Id = Broker.BrokerBaze.dajSesiju().DajSifru(domenskiObjekat);

Rezultat = izvodjac.Id;

Signal = true;

return true;

}

catch (Exception e)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ6:VratiListuIzvođača**

Операција: VratiListuIzvodjaca(Izvodjac) :сигнал

Веза са СК:CK3

Предуслови: -  
Постуслови: -

namespace SistemskeOperacije

{

public class VratiListuIzvodjacaSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Rezultat = Broker.BrokerBaze.dajSesiju().VratiSve(domenskiObjekat).OfType<Izvodjac>().ToList<Izvodjac>();

Signal = true;

return true;

}

catch (Exception)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ7: IzmeniPesmu**

Операција: IzmeniPesmu(Pesma) :сигнал

Веза са СК:CK4

Предуслови: -  
Постуслови: -

namespace SistemskeOperacije

{

public class IzmeniPesmuSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Pesma pesma = (Pesma)domenskiObjekat;

pesma.Id = Broker.BrokerBaze.dajSesiju().Izmena(domenskiObjekat);

Rezultat = pesma.Id;

Signal = true;

return true;

}

catch (Exception)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ8: ZapamtiPesmu**

Операција: ZapamtiPesmu(Pesma) :сигнал

Веза са СК:CK4, СК6

Предуслови:Вредносна и структурна ограничења над објектом песма морају бити задовољена  
Постуслови: Подаци о песми су запамћени

namespace SistemskeOperacije

{

public class ZapamtiPesmuSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Pesma pesma = (Pesma)domenskiObjekat;

pesma.Id = Broker.BrokerBaze.dajSesiju().Sacuvaj(domenskiObjekat);

Rezultat = pesma.Id;

Signal = true;

return true;

}

catch (Exception)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ9: PretragaPesme**

Операција: PretragaPesme(Pesma) :сигнал

Веза са СК:CK5

Предуслови: -   
Постуслови: -

namespace SistemskeOperacije

{

public class PretragaPesmeSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Rezultat = Broker.BrokerBaze.dajSesiju().VratiTacanObjekat(domenskiObjekat);

Signal = true;

return true;

}

catch (Exception e)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ10:PronađiPesmu**

Операција: PronađiPesmu(List<Pesma>) :сигнал

Веза са СК:CK6

Предуслови: -  
Постуслови: -

namespace SistemskeOperacije

{

public class PronadjiPesmuSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Pesma pesma = (Pesma)domenskiObjekat;

Rezultat = Broker.BrokerBaze.dajSesiju().VratiPoKriterijumu(pesma).OfType<Pesma>().ToList<Pesma>();

Signal = true;

return true;

}

catch (Exception e)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ11: VratiPesmuID**

Операција: VratiPesmuID(Pesma) :сигнал

Веза са СК:CK6

Предуслови: -  
Постуслови: -

namespace SistemskeOperacije

{

public class VratiPesmaIDSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Pesma pesma = (Pesma)domenskiObjekat;

pesma.Id = Broker.BrokerBaze.dajSesiju().DajSifru(domenskiObjekat);

Rezultat = pesma.Id;

Signal = true;

return true;

}

catch (Exception)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ12:VratiListuPesama**

Операција: VratiListuPesama(List<Pesma>) :сигнал

Веза са СК:CK7

Предуслови: -  
Постуслови: -

namespace SistemskeOperacije

{

public class VratiListuPesamaSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Rezultat = Broker.BrokerBaze.dajSesiju().VratiSve(domenskiObjekat).OfType<Pesma>().ToList<Pesma>();

Signal = true;

return true;

}

catch (Exception)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ13: PretragaŽanra**

Операција: PretragaZanra(Zanr) :сигнал

Веза са СК:CK5

Предуслови: -   
Постуслови: -

namespace SistemskeOperacije

{

public class PretragaZanraSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Rezultat = Broker.BrokerBaze.dajSesiju().VratiTacanObjekat(domenskiObjekat);

Signal = true;

return false;

}

catch (Exception e)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ14: VratiŽanrID**

Операција: VratiZanrID(Zanr) :сигнал

Веза са СК:CK6

Предуслови: -  
Постуслови: -

namespace SistemskeOperacije

{

public class VratiZanrIDSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

ZanrAlbuma zanr = (ZanrAlbuma)domenskiObjekat;

zanr.Id = Broker.BrokerBaze.dajSesiju().DajSifru(domenskiObjekat);

Rezultat = zanr.Id;

Signal = true;

return true;

}

catch (Exception e)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ15:VratiListuŽanrova**

Операција: VratiListuZanrova(List<Zanr>) :сигнал

Веза са СК:CK7

Предуслови: -  
Постуслови: -

namespace SistemskeOperacije

{

public class VratiListuZanrovaSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Rezultat = Broker.BrokerBaze.dajSesiju().VratiSve(domenskiObjekat).OfType<ZanrAlbuma>().ToList<ZanrAlbuma>();

Signal = true;

return true;

}

catch (Exception)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ16: ZapamtiŽanr**

Операција: ZapamtiZanr(Zanr) :сигнал

Веза са СК:CK4, СК6

Предуслови:Вредносна и структурна ограничења над објектом жанр морају бити задовољена  
Постуслови: Подаци о жанру су запамћени

namespace SistemskeOperacije

{

public class ZapamtiZanrSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

ZanrAlbuma zanr = (ZanrAlbuma)domenskiObjekat;

zanr.Id = Broker.BrokerBaze.dajSesiju().Sacuvaj(domenskiObjekat);

Rezultat = zanr.Id;

Signal = true;

return true;

}

catch (Exception)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ17: DodajPesmaAlbum**

Операција: DodajPesmaAlbum(PesmaAlbum) :сигнал

Веза са СК:CK7

Предуслови: -  
Постуслови: -

namespace SistemskeOperacije

{

public class DodajPesmaAlbumSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

PesmeAlbuma pa = (PesmeAlbuma)domenskiObjekat;

Console.WriteLine("XXX :"+pa.Id);

pa.Id = Broker.BrokerBaze.dajSesiju().Sacuvaj(domenskiObjekat);

Rezultat = pa.Id;

Signal = true;

return true;

}

catch (Exception)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ18: ZapamtiAlbum**

Операција: ZapamtiAlbum(Album) :сигнал

Веза са СК:CK7, СК9

Предуслови:Вредносна и структурна ограничења над објектом албум морају бити задовољена

Постуслови:Подаци о албуму су запамћени

namespace SistemskeOperacije

{

public class ZapamtiAlbumSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Album album = (Album)domenskiObjekat;

album.Id = Broker.BrokerBaze.dajSesiju().Sacuvaj(domenskiObjekat);

Rezultat = album.Id;

Signal = true;

return true;

}

catch (Exception)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ19:PretragaAlbuma**

Операција: PretragaAlbuma(Album) :сигнал

Веза са СК:CK8

Предуслови:-  
Постуслови: -

namespace SistemskeOperacije

{

public class PretragaAlbumaSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Rezultat = Broker.BrokerBaze.dajSesiju().VratiTacanObjekat(domenskiObjekat);

Signal = true;

return false;

}

catch (Exception e)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ20:PronađiAlbum**

Операција: PronadjiAlbum(List<Album>) :сигнал

Веза са СК:CK9

Предуслови:-  
Постуслови: -

namespace SistemskeOperacije

{

public class PronadjiAlbumSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Album album = (Album)domenskiObjekat;

Rezultat = Broker.BrokerBaze.dajSesiju().VratiPoKriterijumu(album).OfType<Album>().ToList<Album>();

Signal = true;

return true;

}

catch (Exception e)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ21:VratiAlbumID**

Операција: VratiAlbumID(Album) :сигнал

Веза са СК:CK9

Предуслови:-  
Постуслови: -

namespace SistemskeOperacije

{

public class VratiAlbumIDSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Album album = (Album)domenskiObjekat;

album.Id = Broker.BrokerBaze.dajSesiju().DajSifru(domenskiObjekat);

Rezultat = album.Id;

Signal = true;

return true;

}

catch (Exception e)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ22: VratiListuAlbuma**

Операција: VratiListuAlbuma(Album) :сигнал

Веза са СК:CK10

Предуслови:-  
Постуслови: -

namespace SistemskeOperacije

{

public class VratiListuAlbumaSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Rezultat = Broker.BrokerBaze.dajSesiju().VratiSve(domenskiObjekat).OfType<Album>().ToList<Album>();

Signal = true;

return true;

}

catch (Exception)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ23:ObrišiPesmaAlbum**

Операција: ObrisiPesmaAlbum(PesmaAlbum) :сигнал

Веза са СК:CK10

Предуслови:Структурно ограничење над објектом ПесмаАлбум мора бити задовољено  
Постуслови: ПесмаАлбум је обрисан

namespace SistemskeOperacije

{

public class ObrisiPesmeAlbumSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Rezultat = Broker.BrokerBaze.dajSesiju().Obrisi(domenskiObjekat);

Signal = true;

return true;

}

catch (Exception)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ24:PretragaPesmaAlbuma**

Операција: PretragaPesmaAlbuma(PesmaAlbum) :сигнал

Веза са СК:CK8

Предуслови:-  
Постуслови: -

namespace SistemskeOperacije

{

public class PretragaPesmeAlbumaSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Rezultat = Broker.BrokerBaze.dajSesiju().VratiTacanObjekat(domenskiObjekat);

Signal = true;

return false;

}

catch (Exception e)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ25:PronađiAlbum**

Операција: PronadjiPesmaAlbum(List<PesmaAlbum>) :сигнал

Веза са СК:CK9

Предуслови:-  
Постуслови: -

namespace SistemskeOperacije

{

public class PronadjiAlbumSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Album album = (Album)domenskiObjekat;

Rezultat = Broker.BrokerBaze.dajSesiju().VratiPoKriterijumu(album).OfType<Album>().ToList<Album>();

Signal = true;

return true;

}

catch (Exception e)

{

Signal = false;

return false;

}

}

}

}

**Уговор УГ26:VratiAlbumID**

Операција: VratiPesmaAlbumID(PesmaAlbum) :сигнал

Веза са СК:CK9

Предуслови:-  
Постуслови: -

namespace SistemskeOperacije

{

public class VratiAlbumIDSO : OpstaSistemskaOperacija

{

protected override object IzvrsiKonkretnuOperaciju(IDomenskiObjekat domenskiObjekat)

{

try

{

Album album = (Album)domenskiObjekat;

album.Id = Broker.BrokerBaze.dajSesiju().DajSifru(domenskiObjekat);

Rezultat = album.Id;

Signal = true;

return true;

}

catch (Exception e)

{

Signal = false;

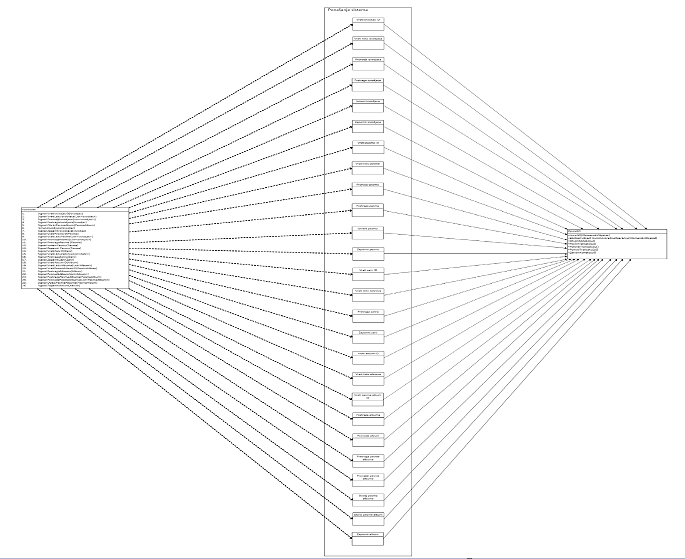
return false;

}

}

}

}



Слика 5: ОСО Класе које су одговорне за СО наслеђују класу OpstaSistemskaOperacija

3.5.3. Доменске класе

На основу концептуалних класа праве се софтверске класе структуре. Свака класа садржи приватна поља атрибута, getter-е, setter-е за исте, конструкторе (било безпараметризовани било параметризовани).

Доменске класе имплементирају интерфејс OpstiDomenskiObjekat како би омогућили лакше имплементирање метода DatabaseBroker-а. На тај начин DatabaseBroker прима интерфејс уместо самих класа и олакшава креирање генеричких упита.

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Domen

{

public interface IDomenskiObjekat

{

List<IDomenskiObjekat> ListaObjekata(SqlDataReader citac);

object NazivTabele();

object PodesiParametreZaIzmenu();

object VratiKljuc();

object VratiUslov();

object VratiUslovZaBrisanje();

string ImeKljuca();

object VratiVrednostiZaInsert();

}

}

3.5.4 Брокер базе података

Брокер базе података је софтверска класа одговорна за комуникацју између пословне логике и складишта података. Другим речима, пројектује се како би обезбедио перзистентни сервис објектима доменских класа који се чувају у бази података. Класа DatabaseBroker представља перзистентни оквир који посредује у свим операцијама над базом података и реализује следеће методе:

namespace Broker

{

public class BrokerBaze

{

SqlCommand komanda;

SqlConnection konekcija;

SqlTransaction transakcija;

void konektujSe()

{

konekcija = new SqlConnection(@"Data Source=MILOS-NOTE;Initial Catalog=MuzickaIzdavackaKuca;Integrated Security=True");

komanda = konekcija.CreateCommand();

}

BrokerBaze()

{

konektujSe();

}

static BrokerBaze instanca;

public static BrokerBaze dajSesiju()

{

if (instanca == null)

{

instanca = new BrokerBaze();

}

return instanca;

}

public List<IDomenskiObjekat> VratiSve(IDomenskiObjekat ido)

{

string upit = $"SELECT \* FROM {ido.NazivTabele()}";

SqlCommand komanda = new SqlCommand(upit, konekcija, transakcija);

SqlDataReader citac = komanda.ExecuteReader();

List<IDomenskiObjekat> rezultat = ido.ListaObjekata(citac);

citac.Close();

try

{

return rezultat;

}

catch (Exception)

{

throw new Exception("Greška!");

}

}

public List<IDomenskiObjekat> VratiPoKriterijumu(IDomenskiObjekat ido)

{

string upit = $"select \* from {ido.NazivTabele()} where {ido.VratiUslov()}";

SqlCommand komanda = new SqlCommand(upit, konekcija, transakcija);

SqlDataReader citac = komanda.ExecuteReader();

List<IDomenskiObjekat> rezultat = ido.ListaObjekata(citac);

citac.Close();

try

{

return rezultat;

}

catch (Exception)

{

throw new Exception("Greška!");

}

}

public IDomenskiObjekat VratiTacanObjekat(IDomenskiObjekat ido)

{

string upit = $"Select \* from {ido.NazivTabele()} where {ido.VratiKljuc()}";

SqlCommand komanda = new SqlCommand(upit, konekcija, transakcija);

SqlDataReader citac = komanda.ExecuteReader();

List<IDomenskiObjekat> rez = ido.ListaObjekata(citac);

citac.Close();

return rez.FirstOrDefault();

}

public IDomenskiObjekat Vratiizvodjaca(IDomenskiObjekat ido, string vrednost)

{

komanda.CommandText = $"Select \* from {ido.NazivTabele()} where {vrednost}";

komanda.CommandType = CommandType.Text;

SqlDataReader citac = komanda.ExecuteReader();

List<IDomenskiObjekat> rez = ido.ListaObjekata(citac);

citac.Close();

return rez.FirstOrDefault();

}

public int Sacuvaj(IDomenskiObjekat ido)

{

string upit = $"insert into {ido.NazivTabele()} values ({ido.VratiVrednostiZaInsert()})";

SqlCommand komanda = new SqlCommand(upit, konekcija, transakcija);

try

{

return komanda.ExecuteNonQuery();

}

catch (Exception e)

{

throw new Exception("Greška!");

}

}

public int Izmena(IDomenskiObjekat ido)

{

Console.WriteLine("sdasd: "+ ido.PodesiParametreZaIzmenu());

string upit = $"update {ido.NazivTabele()} set {ido.PodesiParametreZaIzmenu()} where {ido.VratiKljuc()}";

SqlCommand komanda = new SqlCommand(upit, konekcija, transakcija);

try

{

return komanda.ExecuteNonQuery();

}

catch (Exception)

{

throw new Exception("Greška!");

}

}

public int Obrisi(IDomenskiObjekat ido)

{

string upit = $"delete from {ido.NazivTabele()} where {ido.VratiUslovZaBrisanje()}";

SqlCommand komanda = new SqlCommand(upit, konekcija, transakcija);

try

{

return komanda.ExecuteNonQuery();

}

catch (Exception)

{

throw new Exception("Greška!");

}

}

public int DajSifru(IDomenskiObjekat ido)

{

string upit = @"select MAX("+ido.ImeKljuca()+") from "+ido.NazivTabele()+"";

SqlCommand komanda = new SqlCommand(upit, konekcija, transakcija);

try

{

try

{

int sifra = Convert.ToInt32(komanda.ExecuteScalar());

return sifra + 1;

}

catch (Exception)

{

return 1;

}

}

catch (Exception)

{

throw new Exception("Greška!");

}

}

public void OtvoriKonekciju()

{

konekcija.Open();

}

public void ZatvoriKonekciju()

{

konekcija.Close();

}

public void PocniTransakciju()

{

transakcija = konekcija.BeginTransaction();

}

public void PotvrdiTransakciju()

{

transakcija.Commit();

}

public void PonistiTransakciju()

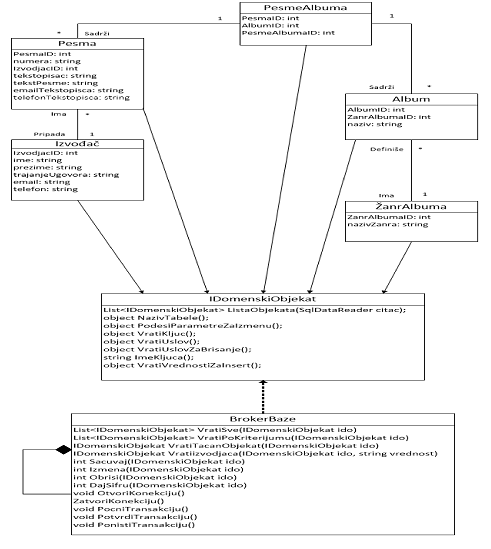
{

transakcija.Rollback();

}

}

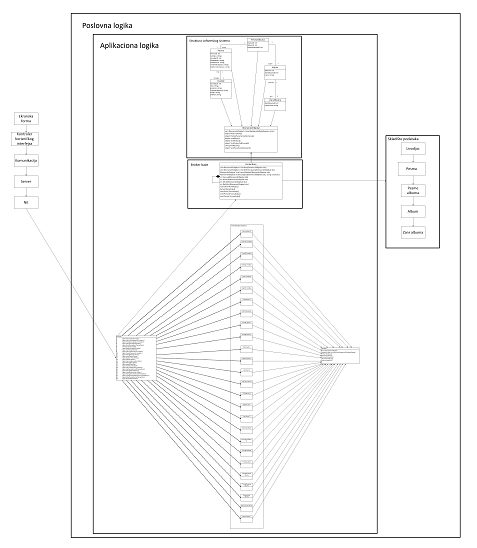
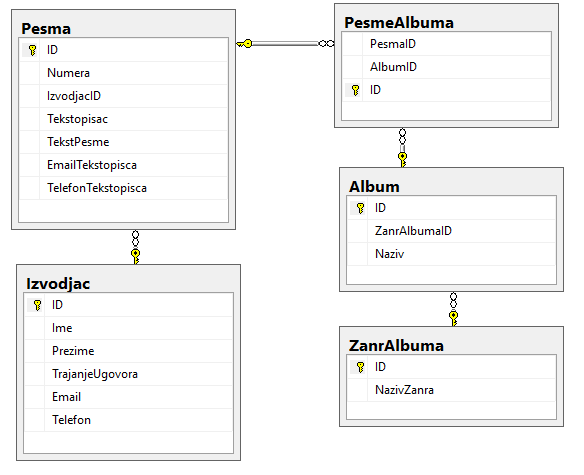
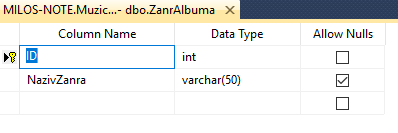
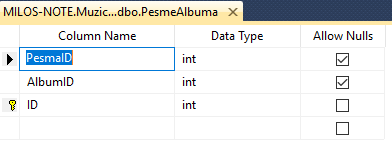
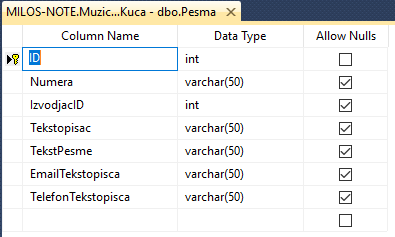
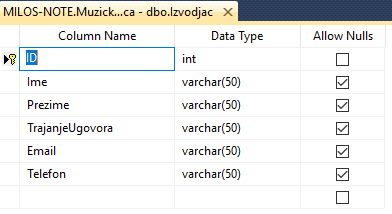
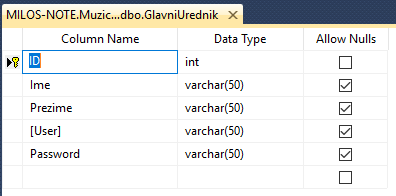
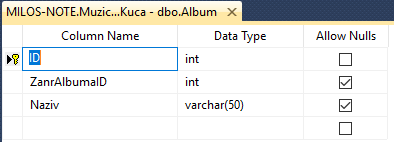
}



*Slika 6: BrokerBaze класа се повезује са класом IDomenskiObjekat*

3.5.5. Пројектовање складишта података

На основу структуре софтверских класа пројектоване су табеле (складишта података) релационог система за управљање базом података. У овом раду је коришћен Microsoft SQL Server Management Studio.

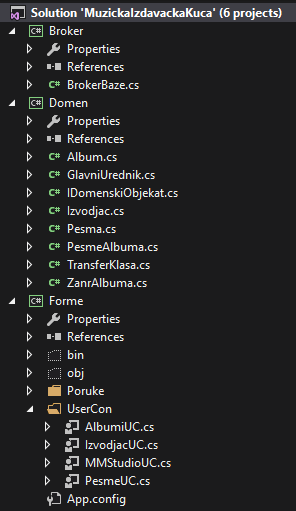


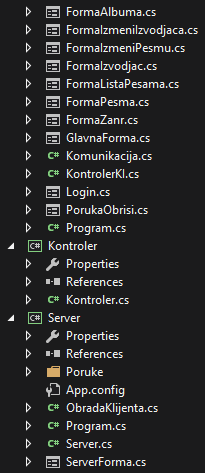
*Слика: Архитектура софт. система након пројектовања табела базе података*

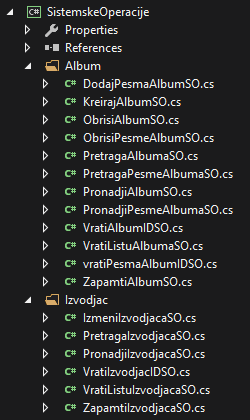
# Имплементација

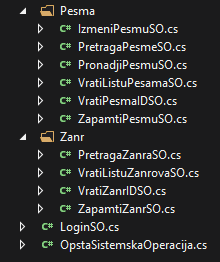
Софтверски систем, резултат овог рада, развијен је у програмском језику C#. Систем је пројектован као клијент-сервер. Као развојно окружење коришћен је Microsoft Visual Studio 2017. Као систем за управљањем базом података коришћен је Microsoft SQL Server Management Studio.

На основу архитектуре софтверског система добијене су следеће софтверске класе:









# 5. ТЕСТИРАЊЕ

Сваки од имплементираних случајева коришћења је тестиран. Приликом тестирања сваког случаја коришћења поред унетих правилних података уношени су и неправилни подаци да би се утврдило какав ће резултат извршења бити. На основу извршених тестирања отклоњени су уочени недостаци.

# 6. ЛИТЕРАТУРА

Пројектовање софтвера – Скрипта, др Синиша Влајић, Београд - 2015.