Univerzitet u Beogradu

Fakultet organizacionih nauka

Laboratorija za softversko inženjerstvo

Seminarski rad iz predmeta

Projektovanje softvera

Tema: Softverski sistem za poslovanje Rent a Car agencije

|  |  |
| --- | --- |
| Profesor:  dr Siniša Vlajić | Student:  Ivana Živaljević 114/14 |
|  |  |

Beograd,

Mart 2019.

Sadržaj

[1. Korisnički zahtev 5](#_Toc4721710)

[1.1 Verbalni opis 5](#_Toc4721711)

[1.2 Slučajevi korisćenja 6](#_Toc4721712)

[SK 1: Slučaj korisćenja – Unos novog vozila 7](#_Toc4721713)

[SK 2: Slučaj korisćenja – Pregled vozila 7](#_Toc4721714)

[SK 3: Slučaj korisćenja – Izmena vozila 8](#_Toc4721715)

[SK 4: Slučaj korisćenja – Brisanje vozila 9](#_Toc4721716)

[SK 5: Slučaj korisćenja – Unos klijenta 10](#_Toc4721717)

[SK 6: Slučaj korisćenja – Pregled klijenata 11](#_Toc4721718)

[SK 7: Slučaj korisćenja – Izmena klijenta 12](#_Toc4721719)

[SK 8: Slučaj korisčenja – Brisanje klijenta 13](#_Toc4721720)

[2.Analiza 15](#_Toc4721721)

[2.1. Ponašanje softverskog sistema - Dijagram sekvenci slučaja koriščenja 15](#_Toc4721722)

[DS1: Dijagram sekvenci slučaja klijenta - Unos novog vozila 15](#_Toc4721723)

[DS 2: Dijagram sekvenci slučaja klijenta – Pregled vozila 17](#_Toc4721724)

[DS 3: Dijagram sekvenci slučaja klijenta – Izmena vozila 18](#_Toc4721725)

[DS 4: Dijagram sekvenci slučaja klijenta – Brisanje vozila 21](#_Toc4721726)

[DS 5: Dijagram sekvenci slučaja klijenta – Unos klijenta 23](#_Toc4721727)

[DS 6: Dijagram sekvenci slučaja klijenta – Pregled klijenata 25](#_Toc4721728)

[DS 7: Dijagram sekvenci – Izmena klijenta 26](#_Toc4721729)

[DS 8: Dijagram sekvenci – Brisanje klijenta 29](#_Toc4721730)

[2.2. Ponašanje softverskog sistema - Definisanje ugovora o sistemskim operacijama 34](#_Toc4721731)

[2.3. Struktura softverskog sistema – Konceptualni (domenski) model 37](#_Toc4721732)

[2.4. Struktura softverskog sistema – Relacioni model 38](#_Toc4721733)

[3. PROJEKTOVANJE 41](#_Toc4721734)

[3.1 Arhitektura Softverskog Sistema 41](#_Toc4721735)

[3.2. Projektovanje korisničkog interfejsa 42](#_Toc4721736)

[1.2.1. Projektovanje ekranskih formi 43](#_Toc4721737)

[SK 1: Slučaj korisćenja – Unos novog vozila 43](#_Toc4721738)

[SK 2: Slučaj korisćenja – Pregled vozila 45](#_Toc4721739)

[SK 3: Slučaj korisćenja – Izmena vozila 46](#_Toc4721740)

[SK 4: Slučaj korisćenja – Brisanje vozila 50](#_Toc4721741)

[SK 5: Slučaj korisćenja – Unos klijenta 53](#_Toc4721742)

[SK 6: Slučaj korisćenja – Pregled klijenata 55](#_Toc4721743)

[SK 7: Slučaj korisćenja – Izmena klijenta 56](#_Toc4721744)

[SK 8: Slučaj korisčenja – Brisanje klijenta 59](#_Toc4721745)

[3.2.3. Projektovanje kontrolera korisničkog interfejsa 62](#_Toc4721746)

[3.3. Projektovanje aplikacione logike 64](#_Toc4721747)

[3.3.1. Kontroler aplikacione logike 64](#_Toc4721748)

[3.3.2. Sistemske operacije 68](#_Toc4721749)

[3.3.3. Domenske klase 78](#_Toc4721750)

[3.3.4 Broker baze podataka 78](#_Toc4721751)

[3.3.5. Projektovanje skladišta podataka 80](#_Toc4721752)

[4. IMPLEMENTACIJA 83](#_Toc4721753)

[5. TESTIRANJE 84](#_Toc4721754)

[6. LITERATURA 85](#_Toc4721755)

# 1. Korisnički zahtev

## 1.1 Verbalni opis

Preduzeće rent-a-car se bavi iznajmiljivanjem vozila klijentima, različitog tipa(malo putničko vozilo, srednje putničko vozilo, veliko putničko vozilo, kombi, mini bus, bus). Svoj vozni park mora da vremenom osveži, pa prema tome je korisniku omogućen uos novih vozila za rentiranje. Primljena vozila unosi u bazu podataka za vozni park.

Svaka firma rent a car mora da ima bazu podataka u koju će da upisuje klijente. Na osnovu toga firma stalnim mušterijama daje određene pogodnosti i popuste. To se radi tako što službenik svaku mušteriju upiše u bazu podataka. Pored toga svaka nova mušterija se upisuje u bazu podataka da bi se imala evidencija o iznajmljenim vozilima.

Kada mušterija dođe u firmu rent a car i zatraži vozilo, službenik ga upoznaje sa voznim parkom firme, dajući mu katalog vozila. Posle odabira, obe strane se dogovore o dodatnim troškovima i načinu plaćanja za iznajmljeni automobil. Prilikom vraćanja automobila službenik pregleda stanje automobila posle korišćenja. Ako je sve u redu, službenik naplaćuje samo cenu korišćenja automobila.

Ako mušterija želi da produži rentu za vozilo ili promeni tip vozila ili datum rentiranja mora da se najavi. On tada zove rent a car kuću i službeniku daje osnovne podatke. Na osnovu čega službenik pronalazi rentiranje u bazi i pomera datum za vraćanje, broj rentiranih dana ili tip vozila. Tada se mušterija obaveštava o promeni rentiranja. Svakom klijentu službenik je dužan uzeti lične podatke i upisati ih u bazu podataka.

Da bi se efikasno izašlo u susret poslovnim izazovima, potrebno je realizovati softver koji će pratiti rad ovog Rent a car centra. Korisniku sistema treba obezbediti sledeće mogućnosti: Unos novog vozila, Pregled vozila, Izmena vozila, Brisanje vozila, Unos klijenta, Pregled klijenta, Izmena i Brisanje klijenta .

Aplikacija treba da vodi evidenciju o svim članovima Rent a car centra, da omogući njihovu pretragu po određenim vrednostima, ali i da omogući unos novih i brisanje postojećih članova, kao i izmenu njihovih podataka.

Neophodno je i voditi evidenciju o vozilima i omogućiti pretragu vozila prema zadatim vrednostima(registarski broj, marka, model), kao i izmenu i brisanje.

Sistemom upravlja korisnik (zaposleni u Rent a car agenciji).

## 1.2 Slučajevi korisćenja

U ovom slučaju uočeni su sledeći slučajevi korisćenja i oni su prikazani na slici 1:

1. Unos novog vozila
2. Pregled vozila
3. Izmena vozila
4. Brisanje vozila
5. Unos klijenta (složen SK)
6. Pregled klijenata
7. Izmena klijenata (složen SK)
8. Brisanje klijenata



Slika 1. Dijagram slučajeva korisčenja

### 

### SK 1: Slučaj korisćenja – Unos novog vozila

**Naziv SK**

Unos *novog vozila*

**Aktori SK**

**Zaposleni u agenciji**

**Učesnici SK**

**Zaposleni** i **sistem** (program)

**Preduslov:Sistem** je uključen . **Sistem** prikazuje formu za unos novog vozila. Učitana je lista tipova.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da kreira *novo vozilo*. (APSO)
2. **Sistem** kreira *novo vozilo*. (SO)
3. **Sistem** prikazuje **korisniku** *novo vozilo* i poruku: „Sistem je kreirao *novo vozilo*“. (IA)
4. **Korisnik** unosi podatke o *novom vozilu*. (APUSO)
5. **Korisnik** proverava da li je korektno uneo podatke o *novom vozilu*. (ANSO)
6. **Korisnik** poziva **sistem** da zapamti podatke o *novom vozilu*. (APSO)
7. **Sistem** pamti podatke o *novom vozilu*. (SO)
8. **Sistem** prikazuje **korisniku** poruku: „**Sistem** je uspešno zapamtio *vozilo*”. (IA)

**Alternativna scenarija**

3.1. Ukoliko **sistem** ne može da omogući kreiranje novog vozila on prikazuje **korisniku** poruku: „**Sistem** ne može da kreira *novo vozilo*“. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da izvrši unos novog vozila on prikazuje **korisniku** poruku : „**Sistem** ne može da zapamti novo *vozilo.*“ (IA)

### SK 2: Slučaj korisćenja – Pregled vozila

**Naziv SK**

Pregled *vozila*

**Aktori SK**

**Zaposleni**

**Učesnici SK**

**Zaposleni** i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za pregled *vozila*.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednost po kojoj pretražuje *vozila*. (APUSO)
2. **Korisnik** poziva **sistem** da nađe *vozila*  po zadatoj vrednosti. (APSO)
3. **Sistem** traži *vozila* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** *vozila* i poruku: „Sistem je uspešno pronašao *vozila*“. (IA)

**Alternativna scenarija:**

4.1. Ukoliko **sistem** ne može da nađe *vozila* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *vozila* ”. (IA)

### SK 3: Slučaj korisćenja – Izmena vozila

**Naziv SK**

Izmena *vozila*

**Aktori SK**

**Zaposleni**

**Učesnici SK**

**Zaposleni** i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za izmenu *vozila*. Učitana je lista tipova.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednost po kojoj pretražuje *vozila*. (APUSO)
2. **Korisnik** poziva **sistem** da nađe *vozila* po zadatoj vrednosti. (APSO)
3. **Sistem** traži *vozila* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** *vozila* i poruku: „Sistem je uspešno pronašao *vozila*“.(IA)
5. **Korisnik** bira vozilo koje želi da izmeni. (APUSO)
6. **Korisnik** poziva **sistem** da nađe *vozilo*. (APSO)
7. **Sistem** traži *vozilo*. (SO)
8. **Sistem** prikazuje **korisniku** *vozilo* i poruku: „Sistem je uspešno pronašao *vozilo*“.(IA)
9. **Korisnik** menja podatke o *vozilu*. (APUSO)
10. **Korisnik** proverava da li je korektno uneo podatke o *vozilu*. (ANSO)
11. **Korisnik** poziva **sistem** da zapamti podatke o *vozilu*. (APSO)
12. **Sistem** pamti podatke o *vozilu*. (SO)
13. **Sistem** prikazuje **korisniku** poruku: “**Sistem** je uspešno izmenio *vozila*.” (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *vozila*  on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *vozila*”. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *vozilo*  on prikazuje **korisniku** poruku: “**Sistem** ne može da pronađe *vozilo*”. Prekida se izvršenje scenarija. (IA)

13.1. Ukoliko **sistem** ne može da zapamti podatke o *vozilu* on prikazuje **korisniku** poruku: “**Sistem** ne može da zapamti *vozilo*”. (IA)

### 

### SK 4: Slučaj korisćenja – Brisanje vozila

**Naziv SK**

Brisanje *vozila*

**Aktori SK**

**Zaposleni**

**Učesnici SK**

**Zaposleni** i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu brisanje *vozila*.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednost po kojoj pretražuje *vozila*. (APUSO)
2. **Korisnik** poziva **sistem** da nađe *vozila* po zadatoj vrednosti. (APSO)
3. **Sistem** traži *vozila* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** *vozila* i poruku: „Sistem je uspešno pronašao *vozila*“.(IA)
5. **Korisnik** bira vozilo koje želi da izmeni. (APUSO)
6. **Korisnik** poziva **sistem** da nađe *vozilo*. (APSO)
7. **Sistem** traži *vozilo*. (SO)
8. **Sistem** prikazuje **korisniku** *vozilo* i poruku: „Sistem je uspešno pronašao *vozilo*“.(IA)
9. **Korisnik** poziva **sistem** da obriše *vozilo*. (APSO)
10. **Sistem** briše *vozilo*. (SO)
11. **Sistem** prikazuje **korisniku** poruku: “**Sistem** je uspešno obrisao *vozilo*.” (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *vozila* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *vozila*”. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *vozilo* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *vozilo*”. Prekida se izvršenje scenarija. (IA)

11.1. Ukoliko **sistem** ne može da obriše *vozilo* on prikazuje **korisniku** poruku: “**Sistem** ne može da obriše *vozilo*”. (IA)

### SK 5: Slučaj korisćenja – Unos klijenta

**Naziv SK**

Unos *klijenta*

**Aktori SK**

**Zaposleni**

**Učesnici SK**

**Zaposleni** i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za unos novog *klijenta*. Učitane su sve zemlje i vozila.

**Osnovni scenario SK**

**Preduslov:Sistem** je uključen . **Sistem** prikazuje formu za unos novog vozila. Učitana je lista tipova.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da kreira *novog klijenta*. (APSO)
2. **Sistem** kreira *novog klijenta*. (SO)
3. **Sistem** prikazuje **korisniku** *novog klijenta* i poruku: „Sistem je kreirao *novog klijenta*“. (IA)
4. **Korisnik** unosi podatke o *novom klijentu*. (APUSO)
5. **Korisnik** proverava da li je korektno uneo podatke o *novom klijentu*. (ANSO)
6. **Korisnik** poziva **sistem** da zapamti podatke o *novom klijentu*. (APSO)
7. **Sistem** pamti podatke o *novom klijentu*. (SO)
8. **Sistem** prikazuje **korisniku** poruku: “**Sistem** je uspešno zapamtio *klijenta* ”. (IA)

**Alternativna scenarija**

3.1. Ukoliko **sistem** ne može da kreira *novog klijenta* on prikazuje **Korisniku** poruku “**Sistem** ne može da kraira *novog klijenta* ”. (IA)

8.1. Ukoliko **sistem** ne može da zapamti podatke o *novom klijentu* on prikazuje **Korisniku** poruku “**Sistem** ne može da zapamti *novog klijenta* ”. (IA)

### SK 6: Slučaj korisćenja – Pregled klijenata

**Naziv SK**

Pregled *klijenata*

**Aktori SK**

**Zaposleni**

**Učesnici SK**

**Zaposleni** i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za pregled *klijenata*.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednosti po kojima pretražuje *klijente*. (APUSO)
2. **Korisnik** poziva **sistem** da nađe *klijente* po zadatim vrednostima. (APSO)
3. **Sistem** traži *klijente* po zadatim vrednostima. (SO)
4. **Sistem** prikazuje **korisniku** nađene *klijente* i poruku: „Sistem je uspešno pronašao *klijente*“. (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *klijente* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *klijente* po zadatim vrednostima”. (IA)

### 

### SK 7: Slučaj korisćenja – Izmena klijenta

**Naziv SK**

Izmena *klijenta*

**Aktori SK**

**Zaposleni**

**Učesnici SK**

**Zaposleni** i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za izmenu *klijenta.* Učitane su sve zemlje i sva vozila.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednosti po kojima pretražuje *klijente*. (APUSO)
2. **Korisnik** poziva **sistem** da nađe *klijente* po zadatim vrednostima. (APSO)
3. **Sistem** traži *klijente* po zadatim vrednostima. (SO)
4. **Sistem** prikazuje **korisniku** nađene *klijente* i poruku: „Sistem je uspešno pronašao *klijente*“. (IA)
5. **Korisnik** bira klijenta kojeg želi da izmeni. (APUSO)
6. **Korisnik** poziva **sistem** da nađe *klijenta*. (APSO)
7. **Sistem** traži *klijenta*. (SO)
8. **Sistem** prikazuje **korisniku** pronađenog *klijenta* i poruku: „Sistem je uspešno pronašao *klijenta*“. (IA)
9. **Korisnik** unosi nove podatke o *klijentu*. (APUSO)
10. **Korisnik** proverava da li je korektno uneo podatke o *klijentu*. (ANSO)
11. **Korisnik** poziva **sistem** da izmeni *klijenta*. (APSO)
12. **Sistem** menja *klijenta*. (SO)
13. **Sistem** prikazuje **korisniku** poruku: „Sistem je uspešno izmenio *klijenta*“. (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *klijente* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *klijente* po zadatim vrednostima”. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *klijenta* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *klijenta* po zadatim vrednostima”. Prekida se izvršenje scenarija. (IA)

13.1. Ukoliko **sistem** ne može da izmeni izabranog *klijenta* on prikazuje **korisniku** poruku: “**Sistem** ne može da zapamti *klijenta* ”. (IA)

### SK 8: Slučaj korisčenja – Brisanje klijenta

**Naziv SK**

Brisanje *klijenta*

**Aktori SK**

**Zaposleni**

**Učesnici SK**

**Zaposleni** i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za brisanje *klijenta*.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednosti po kojima pretražuje *klijente*. (APUSO)
2. **Korisnik** poziva **sistem** da nađe *klijente* po zadatim vrednostima. (APSO)
3. **Sistem** traži *klijente* po zadatim vrednostima. (SO)
4. **Sistem** prikazuje **korisniku** nađene *klijente* i poruku: „Sistem je uspešno pronašao *klijente*“. (IA)
5. **Korisnik** bira klijenta kojeg želi da izmeni. (APUSO)
6. **Korisnik** poziva **sistem** da nađe *klijenta*. (APSO)
7. **Sistem** traži *klijenta*. (SO)
8. **Sistem** prikazuje **korisniku** pronađenog *klijenta* i poruku: „Sistem je uspešno pronašao *klijenta*“. (IA)
9. **Korisnik** poziva **sistem** da obriše *klijenta*. (APSO)
10. **Sistem** briše *klijenta*. (SO)
11. **Sistem** prikazuje **korisniku** poruku: „Sistem je uspešno obrisao izabrano *klijenta*“. (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *klijente* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *klijente* po zadatim vrednostima”. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *klijenta* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *klijenta* po zadatim vrednostima”. Prekida se izvršenje scenarija. (IA)

11.1. Ukoliko **sistem** ne može da obriše izabrano *klijenta* on prikazuje **korisniku** poruku: “**Sistem** ne može da obriše izabrano *klijenta* ”. (IA)

# 2.Analiza

Nakon faze prikupljanja zahteva, prelazi se na sledeću fazu, fazu analize. Ova faza opisuje logičku strukturu i ponašanje sofvterskog sistema, tj. poslovnu logiku softverskog sistema.

* Ponašanje opisujemo pomoću:
* sistemskih dijagrama sekvenci
* ugovora o sistemskim operacijama
* Strukturu opisujemo pomoću:
* konceptualnog modela
* relacionog modela

## 2.1. Ponašanje softverskog sistema - Dijagram sekvenci slučaja koriščenja

Ponašanje sistema se može opisati preko UML-ovih **sekvencnih dijagrama** [Larman]**,** odnosno preko **dijagrama saradnje**[JPRS]**.**

**Sistemski dijagram sekvenci** prikazuje, za izdvojeni scenario SK, događaje u određenom redosledu, koji uspostavljaju interakciju izmenu aktora i softverskog sistema.

**Događaj** koji **napravi** aktor je pobuda za poziv sistemske operacije. Preciznije rečeno, događaj koji napravi aktor prihvata primalac događaja koji nakon toga poziva sistemsku operaciju. To znači da aktor ne poziva sistemsku operaciju neposredno već to čini preko posrednika (primaoca događaja). Poziv sistemske operacije ukazuje na interakciju izmenu aktora i sistema. Za događaj koji predstavlja pobudu za poziv SO se često kaže da je to **sistemski događaj.**

### DS1: Dijagram sekvenci slučaja klijenta - Unos novog vozila

**Preduslov:Sistem** je uključen . **Sistem** prikazuje formu za unos novog vozila. Učitana je lista tipova.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da kreira *novo vozilo*. (APSO)
2. **Sistem** prikazuje **korisniku** *novo vozilo* i poruku: „Sistem je kreirao *novo vozilo*“. (IA)
3. **Korisnik** poziva **sistem** da zapamti podatke o *novom vozilu*. (APSO)
4. **Sistem** prikazuje **korisniku** poruku: „**Sistem** je uspešno zapamtio *vozilo*”. (IA)



**Alternativna scenarija**

3.1. Ukoliko **sistem** ne može da omogući kreiranje novog vozila on prikazuje **korisniku** poruku: „**Sistem** ne može da kreira *novo vozilo*“. Prekida se izvršenje scenarija. (IA)



8.1. Ukoliko **sistem** ne može da izvrši unos novog vozila on prikazuje **korisniku** poruku : „**Sistem** ne može da zapamti novo *vozilo.*“ (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

1. *signal* **KreirajVozilo**(*Vozilo*)
2. *signal* **ZapamtiVozilo**(*Vozilo*)
3. *signal* **vratiSveTipove**(*List<TipVozila>)*

### DS 2: Dijagram sekvenci slučaja klijenta – Pregled vozila

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za pregled *vozila*.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da nađe *vozila*  po zadatoj vrednosti. (APSO)
2. **Sistem** prikazuje **korisniku** *vozila* i poruku: „Sistem je uspešno pronašao *vozila*“. (IA)



**Alternativna scenarija:**

4.1. Ukoliko **sistem** ne može da nađe *vozila* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *vozila* ”. (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

1. *signal* **PretraziVozila**(*Vozilo, List<Vozilo>*)

### DS 3: Dijagram sekvenci slučaja klijenta – Izmena vozila

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za izmenu *vozila*. Učitana je lista tipova.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da nađe *vozila* po zadatoj vrednosti. (APSO)
2. **Sistem** prikazuje **korisniku** *vozila* i poruku: „Sistem je uspešno pronašao *vozila*“.(IA)
3. **Korisnik** poziva **sistem** da nađe *vozilo*. (APSO)
4. **Sistem** prikazuje **korisniku** *vozilo* i poruku: „Sistem je uspešno pronašao *vozilo*“.(IA)
5. **Korisnik** poziva **sistem** da zapamti podatke o *vozilu*. (APSO)
6. **Sistem** prikazuje **korisniku** poruku: “**Sistem** je uspešno izmenio *vozila*.” (IA)



**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *vozila*  on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *vozila*”. Prekida se izvršenje scenarija. (IA)



8.1. Ukoliko **sistem** ne može da nađe *vozilo*  on prikazuje **korisniku** poruku: “**Sistem** ne može da pronađe *vozilo*”. Prekida se izvršenje scenarija. (IA)



13.1. Ukoliko **sistem** ne može da zapamti podatke o *vozilu* on prikazuje **korisniku** poruku: “**Sistem** ne može da zapamti *vozilo*”. (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

1. *signal* **PretraziVozila**(*Vozilo, List<Vozilo>*)
2. *signal* **PronadjiVozilo**(*Vozilo*)
3. *signal* **ZapamtiVozilo**(*Vozilo)*
4. *signal* **VratiSveTipove**(*List<TipVozila>)*

### 

### DS 4: Dijagram sekvenci slučaja klijenta – Brisanje vozila

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu brisanje *vozila*.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da nađe *vozila* po zadatoj vrednosti. (APSO)
2. **Sistem** prikazuje **korisniku** *vozila* i poruku: „Sistem je uspešno pronašao *vozila*“.(IA)
3. **Korisnik** poziva **sistem** da nađe *vozilo*. (APSO)
4. **Sistem** prikazuje **korisniku** *vozilo* i poruku: „Sistem je uspešno pronašao *vozilo*“.(IA)
5. **Korisnik** poziva **sistem** da obriše *vozilo*. (APSO)
6. **Sistem** prikazuje **korisniku** poruku: “**Sistem** je uspešno obrisao *vozilo*.” (IA)



**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *vozila* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *vozila*”. Prekida se izvršenje scenarija. (IA)



8.1. Ukoliko **sistem** ne može da nađe *vozilo* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *vozilo*”. Prekida se izvršenje scenarija. (IA)



11.1. Ukoliko **sistem** ne može da obriše *vozilo* on prikazuje **korisniku** poruku: “**Sistem** ne može da obriše *vozilo*”. (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

1. *signal* **PretraziVozila**(*Vozilo, List<Vozilo>*)
2. *signal* **PronadjiVozilo**(*Vozilo*)
3. *signal* **ObrisiVozilo**(*Vozilo)*

### DS 5: Dijagram sekvenci slučaja klijenta – Unos klijenta

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za unos novog *klijenta*. Učitane su sve zemlje i vozila.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da kreira *novog klijenta*. (APSO)
2. **Sistem** prikazuje **korisniku** *novog klijenta* i poruku: „Sistem je kreirao *novog klijenta*“. (IA)
3. **Korisnik** poziva **sistem** da zapamti podatke o *novom klijentu*. (APSO)
4. **Sistem** prikazuje **korisniku** poruku: “**Sistem** je uspešno zapamtio *klijenta* ”. (IA)



**Alternativna scenarija**

3.1. Ukoliko **sistem** ne može da kreira *novog klijenta* on prikazuje **Korisniku** poruku “**Sistem** ne može da kraira *novog klijenta* ”. (IA)



8.1. Ukoliko **sistem** ne može da zapamti podatke o *novom klijentu* on prikazuje **Korisniku** poruku “**Sistem** ne može da zapamti *novog klijenta* ”. (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

1. *signal* **KreirajKlijenta** (*Klijent*)
2. *signal* **ZapamtiKlijenta** (*Klijent*)
3. *signal* **vratiSvaVozila**(*List<Vozilo>)*
4. *signal* **vratiSveZemlje**(*List<Zemlja>)*

### DS 6: Dijagram sekvenci slučaja klijenta – Pregled klijenata

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za pregled *klijenata*.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da nađe *klijente* po zadatim vrednostima. (APSO)
2. **Sistem** prikazuje **korisniku** nađene *klijente* i poruku: „Sistem je uspešno pronašao *klijente*“. (IA)



**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *klijente* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *klijente* po zadatim vrednostima”. (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

1. *signal* **PretraziKlijente**(*Klijent, List<Klijent>*)

### DS 7: Dijagram sekvenci – Izmena klijenta

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za izmenu *klijenta.* Učitane su sve zemlje i sva vozila.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da nađe *klijente* po zadatim vrednostima. (APSO)
2. **Sistem** prikazuje **korisniku** nađene *klijente* i poruku: „Sistem je uspešno pronašao *klijente*“. (IA)
3. **Korisnik** poziva **sistem** da nađe *klijenta*. (APSO)
4. **Sistem** prikazuje **korisniku** pronađenog *klijenta* i poruku: „Sistem je uspešno pronašao *klijenta*“. (IA)
5. **Korisnik** poziva **sistem** da izmeni *klijenta*. (APSO)
6. **Sistem** prikazuje **korisniku** poruku: „Sistem je uspešno izmenio *klijenta*“. (IA)



**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *klijente* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *klijente* po zadatim vrednostima”. Prekida se izvršenje scenarija. (IA)



8.1. Ukoliko **sistem** ne može da nađe *klijenta* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *klijenta* po zadatim vrednostima”. Prekida se izvršenje scenarija. (IA)



13.1. Ukoliko **sistem** ne može da izmeni izabranog *klijenta* on prikazuje **korisniku** poruku: “**Sistem** ne može da zapamti *klijenta* ”. (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

1. *signal* **PretraziKlijente**(*Klijent,List<Klijent>*)
2. *signal* **PronadjiKlijenta**(*Klijent*)
3. *signal* **ZapamtiKlijenta**(*Klijent*)
4. *signal* **VratiSveZemlje**(*List<Zemlja>)*
5. *signal* **VratiSvaVozila**(*List<Vozilo>)*

### DS 8: Dijagram sekvenci – Brisanje klijenta

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za izmenu *klijenta.*

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da nađe *klijente* po zadatim vrednostima. (APSO)
2. **Sistem** prikazuje **korisniku** nađene *klijente* i poruku: „Sistem je uspešno pronašao *klijente*“. (IA)
3. **Korisnik** poziva **sistem** da nađe *klijenta*. (APSO)
4. **Sistem** prikazuje **korisniku** pronađenog *klijenta* i poruku: „Sistem je uspešno pronašao *klijenta*“. (IA)
5. **Korisnik** poziva **sistem** da izmeni *klijenta*. (APSO)
6. **Sistem** prikazuje **korisniku** poruku: „Sistem je uspešno obrisa *klijenta*“. (IA)



**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *klijente* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *klijente* po zadatim vrednostima”. Prekida se izvršenje scenarija. (IA)



8.1. Ukoliko **sistem** ne može da nađe *klijenta* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *klijenta* po zadatim vrednostima”. Prekida se izvršenje scenarija. (IA)



13.1. Ukoliko **sistem** ne može da izmeni izabranog *klijenta* on prikazuje **korisniku** poruku: “**Sistem** ne može da zapamti *klijenta* ”. (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

1. *signal* **PretraziKlijente**(*Klijent,List<Klijent>*)
2. *signal* **PronadjiKlijenta**(*Klijent*)
3. *signal* **ObrisiKlijenta**(*List<Zemlja>)*

Kao rezultat analize scenarija dobijeno je ukupno 13 sistemskih operacija koje treba projektovati:

1. *signal* **KreirajVozilo**(*Vozilo*)
2. *signal* **ZapamtiVozilo**(*Vozilo*)
3. *signal* **vratiSveTipove**(*List<TipVozila>)*
4. *signal* **PretraziVozila**(*Vozilo, List<Vozilo>*)
5. *signal* **PronadjiVozilo**(*Vozilo*)
6. *signal* **ObrisiVozilo**(*Vozilo)*
7. *signal* **KreirajKlijenta** (*Klijent*)
8. *signal* **ZapamtiKlijenta** (*Klijent*)
9. *signal* **vratiSvaVozila**(*List<Vozilo>)*
10. *signal* **vratiSveZemlje**(*List<Zemlja>)*
11. *signal* **PretraziKlijente**(*Klijent, List<Klijent>*)
12. *signal* **PronadjiKlijenta**(*Klijent*)
13. *signal* **ObrisiKlijenta**(*List<Zemlja>)*

## 2.2. Ponašanje softverskog sistema - Definisanje ugovora o sistemskim operacijama

**Ugovor UG1: KreirajVozilo**

Operacija: KreirajVozilo(*Vozilo*):signal;

Veza sa SK: SK1

Preduslovi: *Prosto* *vrednosno ograničenje nad objektom Vozilo mora biti zadovoljeno.*

*Strukturno ograničenje nad objektom Vozilo mora biti zadovoljeno.*

Postuslovi: *Kreirano je novo vozilo.*

**Ugovor UG2: ZapamtiVozilo**

Operacija: ZapamtiVozilo(*Vozilo*):signal;

Veza sa SK: SK1, SK3

Preduslovi: *Prosto vrednosno ograničenje nad objektom Vozilo mora biti zadovoljeno.*

*Strukturno ograničenje nad objektom Vozilo mora biti zadovoljeno.*

Postuslovi: *Podaci o vozilu su zapamćeni.*

**Ugovor UG3: PretraziVozila**

Operacija: PretraziVozila(*Vozilo,List<Vozilo>*):signal;

Veza sa SK: SK2, SK3, SK4

Preduslovi: -

Postuslovi: -

**Ugovor UG4:ObrisiVozilo**

Operacija: ObrisiVozilo(*Vozilo*):signal;

Veza sa SK: SK4

Preduslovi: *Strukturno ograničenje nad objektom Vozilo mora biti zadovoljeno.*

Postuslovi: *Vozilo je obrisano.*

**Ugovor UG5: PronadjiVozilo**

Operacija: PronadjiVozilo(*Vozilo*):signal;

Veza sa SK: SK5, SK6

Preduslovi: -

Postuslovi: *-*

**Ugovor UG6: vratiSveZemlje**

Operacija: vratiSveZemlje (*List<Zemlja>*):signal;

Veza sa SK: SK5, SK7

Preduslovi: */*

Postuslovi: */*

**Ugovor UG7: PretraziKlijente**

Operacija: PretraziKlijente(*Klijent, List<Klijent>*):signal;

Veza sa SK: SK5,SK6,SK7

Preduslovi: -

Postuslovi: -

**Ugovor UG8: vratiTipove**

Operacija: vratiTipove (*List<TipVozila>*):signal;

Veza sa SK: SK5,SK7

Preduslovi: */*

Postuslovi: /

**Ugovor UG9: KreirajKlijenta**

Operacija: KreirajKlijenta (*Klijent*):signal;

Veza sa SK: SK5, SK7

Preduslovi: *Prosto vrednosno ograničenje nad objektom Klijent mora biti zadovoljeno.*

*Strukturno ograničenje nad objektom Klijent mora biti zadovoljeno.*

Postuslovi: *Podaci o klijentu su zapamćeni.*

**Ugovor UG10: ZapamtiKlijenta**

Operacija: ZapamtiKlijenta (*Klijent*):signal;

Veza sa SK: SK5, SK7

Preduslovi: *Prosto vrednosno ograničenje nad objektom Klijent mora biti zadovoljeno.*

*Strukturno ograničenje nad objektom Klijent mora biti zadovoljeno.*

Postuslovi: *Podaci o klijentu su zapamćeni.*

**Ugovor UG11: PronadjiKlijenta**

Operacija: PronadjiKlijenta (*Klijent*):signal;

Veza sa SK: SK7,SK8

Preduslovi: -

Postuslovi: -

**Ugovor UG12: ObrisiKlijenta**

Operacija: ObrisiKlijenta(*Klijent*):signal;

Veza sa SK: SK8

Preduslovi: *Strukturno ograničenje nad objektom Klijent mora biti zadovoljeno.*

Postuslovi: *Klijent je obrisan.*

**Ugovor UG13: VratiSvaVozila**

Operacija: VratiSvaVozila (*List<Vozilo>*):signal;

Veza sa SK: SK5, SK7

Preduslovi: */*

Postuslovi: */*

## 2.3. Struktura softverskog sistema – Konceptualni (domenski) model



## 2.4. Struktura softverskog sistema – Relacioni model

Klijent(ID, Ime, Prezime, JMBG, Kontakt, Mail, IznosZaUplatu, *ZemljaID*)

Zemlja(ID, Naziv)

Vozilo(ID, RegBroj, Model, Marka, CenaPoDanu, *TipVozilaID*)

TipVozila(ID, Naziv, Opis, BrojSedista)

Rentiranje( ID, Datum, BrojDana, Klijent*ID, VoziloID*)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tabela Klijent** | | **Prosto vrednosno ograničenje** | | **Složeno vrednosno ograničenje** | | **Strukturno ograničenje** |
| **Atributi** | **Ime** | **Tip atributa** | **Vrednost atributa** | **Međuzav. atributa jedne tabele** | **Međuzav. atributa više tabela** | INSERT RESTRICTED TipVozila  UPDATE  CASCADES Rentiranje  RESTRICTED  TipVozila  DELETE RESTRICTED  Rentiranje |
| ID | Integer | not null and >0 |  |  |
| Ime | String | not null |  |  |
| Prezime | String | not null |  |  |
| JMBG | String | not null |  |  |
| Kontakt | String |  |  |  |
| Mail | String |  |  |  |
| IznosZaNaplatu | Double | not null  and >0 |  |  |
| ZemljaID | Integer | not null  and >0 |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tabela Zemlja** | | **Prosto vrednosno ograničenje** | | **Složeno vrednosno ograničenje** | | **Strukturno ograničenje** |
| **Atributi** | **Ime** | **Tip atributa** | **Vrednost atributa** | **Međuzav. atributa jedne tabele** | **Međuzav. atributa više tabela** | INSERT /  UPDATE CASCADES Klijent  DELETE  RESTRICTED Klijent |
| ID | Integer | not null and >0 |  |  |
| Naziv | String | not null |  |  |
|  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tabela Vozilo** | | **Prosto vrednosno ograničenje** | | **Složeno vrednosno ograničenje** | | **Strukturno ograničenje** |
| **Atributi** | **Ime** | **Tip atributa** | **Vrednost atributa** | **Međuzav. atributa jedne tabele** | **Međuzav. atributa više tabela** | INSERT RESTRICTED  TipVozila  UPDATE CASCADES Rentiranje  RESTRICTED  TipVozila  DELETE RESTRICTED Rentiranje |
| ID | Integer | not null and >0 |  |  |
| Naziv | String | not null |  |  |
| Model | String | not null |  |  |
| Marka | String |  |  |  |
| CenaPoDanu | Double | >0 |  |  |
| TipVozilaID | Integer | not null and >0 |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tabela TipVozila** | | **Prosto vrednosno ograničenje** | | **Složeno vrednosno ograničenje** | | **Strukturno ograničenje** |
| **Atributi** | **Ime** | **Tip atributa** | **Vrednost atributa** | **Međuzav. atributa jedne tabele** | **Međuzav. atributa više tabela** | INSERT /  UPDATE CASCADES Vozilo  DELETE  RESTRICTED Vozilo |
| ID | Integer | not null and >0 |  |  |
| Naziv | String | not null |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tabela Rentiranje** | | **Prosto vrednosno ograničenje** | | **Složeno vrednosno ograničenje** | | **Strukturno ograničenje** |
| **Atributi** | **Ime** | **Tip atributa** | **Vrednost atributa** | **Međuzav. atributa jedne tabele** | **Međuzav. atributa više tabela** | INSERT RESTRICTED  Vozilo  Klijent  UPDATE RESTRICTED  Vozilo  Klijent  DELETE / |
| ID | Integer | not null  and >0 |  |  |
| Datum | Date | not null |  |  |
| BrDana | Integer | not null and >0 |  |  |
| VoziloID | Integer | not null and >0 |  |  |
| KlijentID | Integer | not null and >0 |  |  |

­­

# 3. PROJEKTOVANJE

Faza projektovanja opisuje fizičku strukturu i ponašanje softverskog sistema arhitekturu softverskog sistema). Projektovanje arhitekture softverskog sistema obuhvata projektovanje korisničkog interfejsa, aplikacione logike i skladišta podataka. Projektovanje korisničkog interfejsa obuhvata projektovanje ekranskih formi i kontrolera korisničkog interfejsa. U okviru aplikacione logike se projektuju kontroler aplikacione logike, poslovna logika I broker baze podataka. Projektovanje poslovne logika obuhvata projektovanje logičke structure i ponašanja softverskog sistema.

## 3.1 Arhitektura Softverskog Sistema

U okviru faze projektovanja opisuje se fizička struktura i ponašanje softverskog sistema tj. arhitektura softverskog sistema. U ovom seminarskom radu je korišćena klasična troslojna arhitektura, koja se sastoji od:

* Korisničkog interfejsa
* Aplikacione logike
* Skladišta podataka

Korisnički interfejs

Aplikaciona logika

Skladište podataka

**I sloj**

**II sloj**

**III sloj**

**Softverski sistem**

*Slika: Troslojna arhitektura*

Skladište podataka

**Korisnički interfejs**

Ekranska forma

KontrolerKI

**Aplikaciona logika**

KontrolerAL

Poslovna logika

Database broker

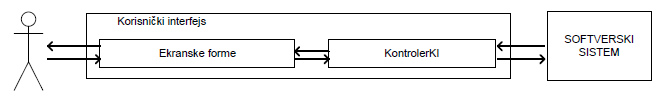
*Slika: Troslojna arhitektura – detaljniji prikaz*

## 3.2. Projektovanje korisničkog interfejsa

Kontroler korisničkog interfejsa i ekranske forme ne predstavljaju softverski sistem, već realizaciju ulaza i izlaza iz softverskog sistema. Kontroler korisničkog interfejsa ima uloga da konvertuje podatke u grafičke elemente ekranskih formi i da podatke ili pošalje u softverski sistem ili da primi podatke iz softverskog sistema i iste konvertuje u elemente ekranske forme.

Korisnički interfejs se sastoji iz:

* Ekranske forme,
* Kontrolera korisničkog interfejsa.



### Projektovanje ekranskih formi

### SK 1: Slučaj korisćenja – Unos novog vozila

**Naziv SK**

Unos *novog vozila*

**Aktori SK**

**Zaposleni u agenciji**

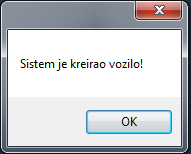
**Učesnici SK**

**Zaposleni** i **sistem** (program)

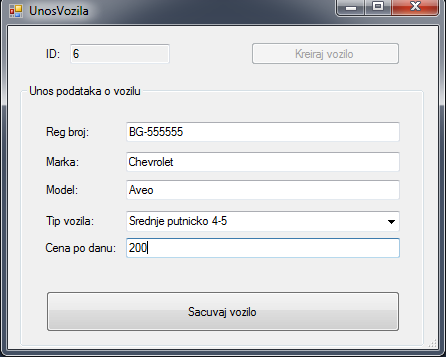
**Preduslov:Sistem** je uključen . **Sistem** prikazuje formu za unos novog vozila. Učitana je lista tipova.

**Osnovni scenario SK**

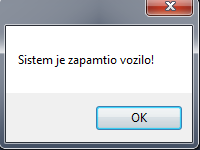
1. **Korisnik** poziva **sistem** da kreira *novo vozilo*. (APSO)
2. **Sistem** kreira *novo vozilo*. (SO)
3. **Sistem** prikazuje **korisniku** *novo vozilo* i poruku: „Sistem je kreirao *novo vozilo*“. (IA)



1. **Korisnik** unosi podatke o *novom vozilu*. (APUSO)



1. **Korisnik** proverava da li je korektno uneo podatke o *novom vozilu*. (ANSO)
2. **Korisnik** poziva **sistem** da zapamti podatke o *novom vozilu*. (APSO)
3. **Sistem** pamti podatke o *novom vozilu*. (SO)
4. **Sistem** prikazuje **korisniku** poruku: „**Sistem** je uspešno zapamtio *vozilo*”. (IA)



**Alternativna scenarija**

3.1. Ukoliko **sistem** ne može da omogući kreiranje novog vozila on prikazuje **korisniku** poruku: „**Sistem** ne može da kreira *novo vozilo*“. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da izvrši unos novog vozila on prikazuje **korisniku** poruku : „**Sistem** ne može da zapamti novo *vozilo.*“ (IA)

### SK 2: Slučaj korisćenja – Pregled vozila

**Naziv SK**

Pregled *vozila*

**Aktori SK**

**Zaposleni**

**Učesnici SK**

**Zaposleni** i **sistem** (program)

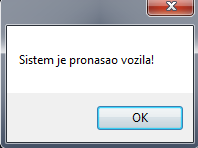
**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za pregled *vozila*.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednost po kojoj pretražuje *vozila*. (APUSO)



1. **Korisnik** poziva **sistem** da nađe *vozila*  po zadatoj vrednosti. (APSO)
2. **Sistem** traži *vozila* po zadatoj vrednosti. (SO)
3. **Sistem** prikazuje **korisniku** *vozila* i poruku: „Sistem je uspešno pronašao *vozila*“. (IA)



**Alternativna scenarija:**

4.1. Ukoliko **sistem** ne može da nađe *vozila* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *vozila* ”. (IA)

### SK 3: Slučaj korisćenja – Izmena vozila

**Naziv SK**

Izmena *vozila*

**Aktori SK**

**Zaposleni**

**Učesnici SK**

**Zaposleni** i **sistem** (program)

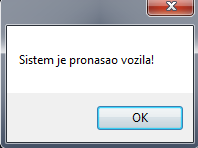
**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za izmenu *vozila*. Učitana je lista tipova.

**Osnovni scenario SK**

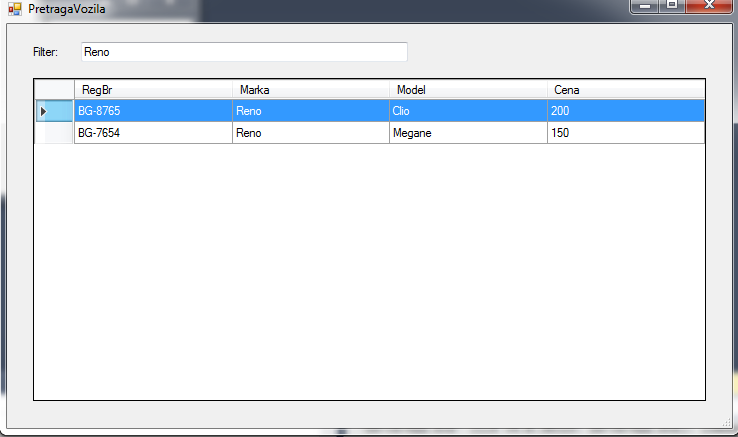
1. **Korisnik** unosi vrednost po kojoj pretražuje *vozila*. (APUSO)



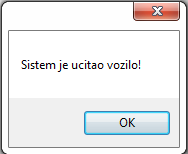
1. **Korisnik** poziva **sistem** da nađe *vozila* po zadatoj vrednosti. (APSO)
2. **Sistem** traži *vozila* po zadatoj vrednosti. (SO)
3. **Sistem** prikazuje **korisniku** *vozila* i poruku: „Sistem je uspešno pronašao *vozila*“.(IA)



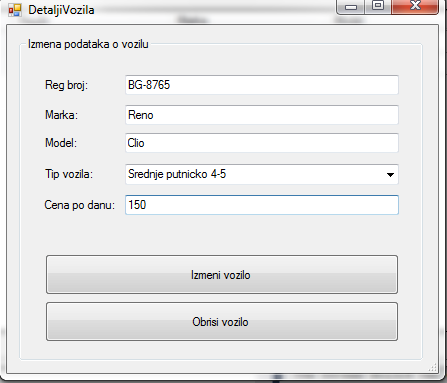
1. **Korisnik** bira vozilo koje želi da izmeni. (APUSO)



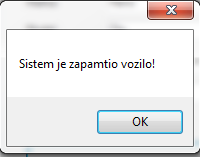
1. **Korisnik** poziva **sistem** da nađe *vozilo*. (APSO)
2. **Sistem** traži *vozilo*. (SO)
3. **Sistem** prikazuje **korisniku** *vozilo* i poruku: „Sistem je uspešno pronašao *vozilo*“.(IA)



1. **Korisnik** menja podatke o *vozilu*. (APUSO)



1. **Korisnik** proverava da li je korektno uneo podatke o *vozilu*. (ANSO)
2. **Korisnik** poziva **sistem** da zapamti podatke o *vozilu*. (APSO)
3. **Sistem** pamti podatke o *vozilu*. (SO)
4. **Sistem** prikazuje **korisniku** poruku: “**Sistem** je uspešno izmenio *vozila*.” (IA)



**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *vozila*  on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *vozila*”. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *vozilo*  on prikazuje **korisniku** poruku: “**Sistem** ne može da pronađe *vozilo*”. Prekida se izvršenje scenarija. (IA)

13.1. Ukoliko **sistem** ne može da zapamti podatke o *vozilu* on prikazuje **korisniku** poruku: “**Sistem** ne može da zapamti *vozilo*”. (IA)

### SK 4: Slučaj korisćenja – Brisanje vozila

**Naziv SK**

Brisanje *vozila*

**Aktori SK**

**Zaposleni**

**Učesnici SK**

**Zaposleni** i **sistem** (program)

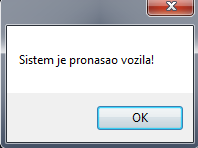
**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu brisanje *vozila*.

**Osnovni scenario SK**

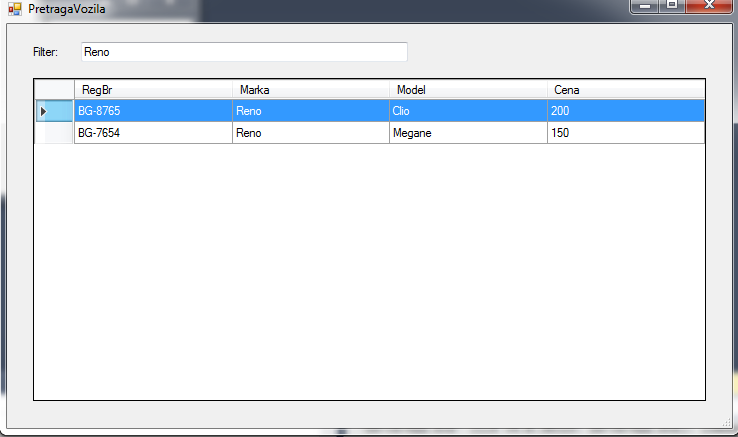
1. **Korisnik** unosi vrednost po kojoj pretražuje *vozila*. (APUSO)



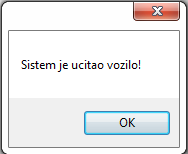
1. **Korisnik** poziva **sistem** da nađe *vozila* po zadatoj vrednosti. (APSO)
2. **Sistem** traži *vozila* po zadatoj vrednosti. (SO)
3. **Sistem** prikazuje **korisniku** *vozila* i poruku: „Sistem je uspešno pronašao *vozila*“.(IA)



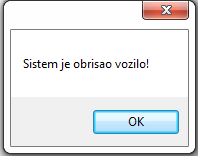
1. **Korisnik** bira vozilo koje želi da izmeni. (APUSO)



1. **Korisnik** poziva **sistem** da nađe *vozilo*. (APSO)
2. **Sistem** traži *vozilo*. (SO)
3. **Sistem** prikazuje **korisniku** *vozilo* i poruku: „Sistem je uspešno pronašao *vozilo*“.(IA)



1. **Korisnik** poziva **sistem** da obriše *vozilo*. (APSO)
2. **Sistem** briše *vozilo*. (SO)
3. **Sistem** prikazuje **korisniku** poruku: “**Sistem** je uspešno obrisao *vozilo*.” (IA)



**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *vozila* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *vozila*”. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *vozilo* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *vozilo*”. Prekida se izvršenje scenarija. (IA)

11.1. Ukoliko **sistem** ne može da obriše *vozilo* on prikazuje **korisniku** poruku: “**Sistem** ne može da obriše *vozilo*”. (IA)

### SK 5: Slučaj korisćenja – Unos klijenta

**Naziv SK**

Unos *klijenta*

**Aktori SK**

**Zaposleni**

**Učesnici SK**

**Zaposleni** i **sistem** (program)

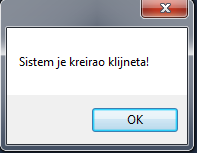
**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za unos novog *klijenta*. Učitane su sve zemlje i vozila.

**Osnovni scenario SK**

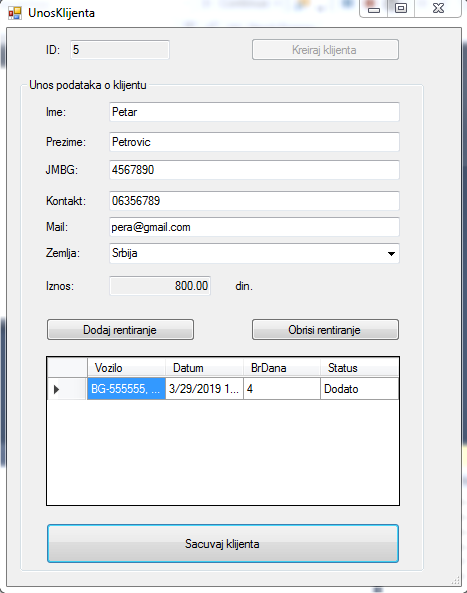
**Preduslov:Sistem** je uključen . **Sistem** prikazuje formu za unos novog vozila. Učitana je lista tipova.

**Osnovni scenario SK**

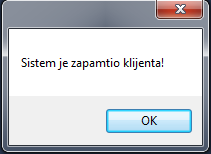
1. **Korisnik** poziva **sistem** da kreira *novog klijenta*. (APSO)
2. **Sistem** kreira *novog klijenta*. (SO)
3. **Sistem** prikazuje **korisniku** *novog klijenta* i poruku: „Sistem je kreirao *novog klijenta*“. (IA)



1. **Korisnik** unosi podatke o *novom klijentu*. (APUSO)



1. **Korisnik** proverava da li je korektno uneo podatke o *novom klijentu*. (ANSO)
2. **Korisnik** poziva **sistem** da zapamti podatke o *novom klijentu*. (APSO)
3. **Sistem** pamti podatke o *novom klijentu*. (SO)
4. **Sistem** prikazuje **korisniku** poruku: “**Sistem** je uspešno zapamtio *klijenta* ”. (IA)



**Alternativna scenarija**

3.1. Ukoliko **sistem** ne može da kreira *novog klijenta* on prikazuje **Korisniku** poruku “**Sistem** ne može da kraira *novog klijenta* ”. (IA)

8.1. Ukoliko **sistem** ne može da zapamti podatke o *novom klijentu* on prikazuje **Korisniku** poruku “**Sistem** ne može da zapamti *novog klijenta* ”. (IA)

### SK 6: Slučaj korisćenja – Pregled klijenata

**Naziv SK**

Pregled *klijenata*

**Aktori SK**

**Zaposleni**

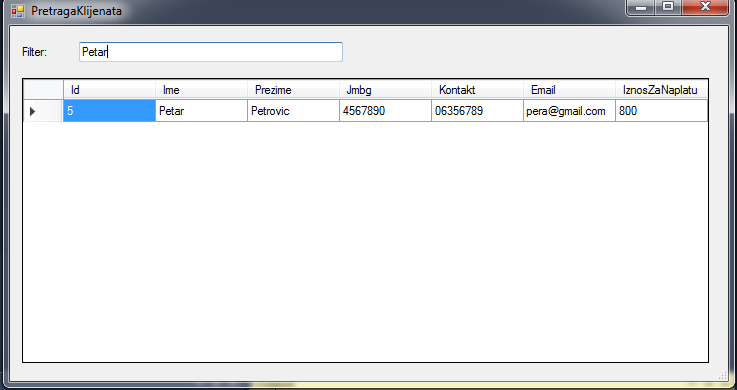
**Učesnici SK**

**Zaposleni** i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za pregled *klijenata*.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednosti po kojima pretražuje *klijente*. (APUSO)



1. **Korisnik** poziva **sistem** da nađe *klijente* po zadatim vrednostima. (APSO)
2. **Sistem** traži *klijente* po zadatim vrednostima. (SO)
3. **Sistem** prikazuje **korisniku** nađene *klijente* i poruku: „Sistem je uspešno pronašao *klijente*“. (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *klijente* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *klijente* po zadatim vrednostima”. (IA)

### SK 7: Slučaj korisćenja – Izmena klijenta

**Naziv SK**

Izmena *klijenta*

**Aktori SK**

**Zaposleni**

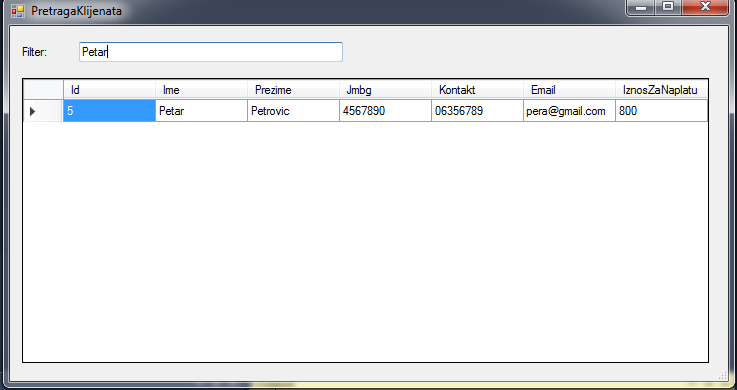
**Učesnici SK**

**Zaposleni** i **sistem** (program)

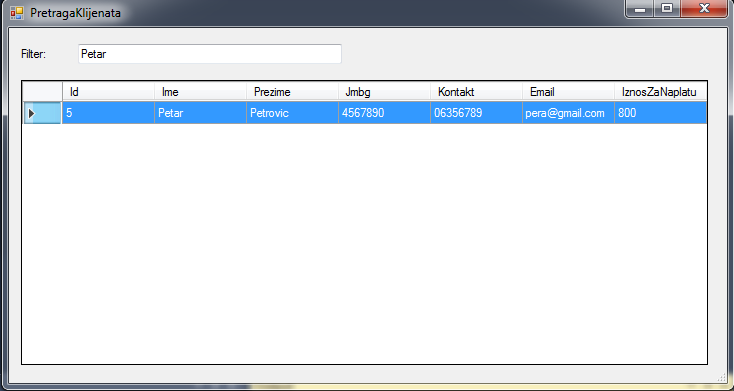
**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za izmenu *klijenta.* Učitane su sve zemlje i sva vozila.

**Osnovni scenario SK**

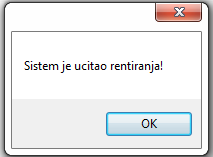
1. **Korisnik** unosi vrednosti po kojima pretražuje *klijente*. (APUSO)



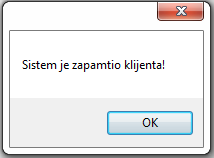
1. **Korisnik** poziva **sistem** da nađe *klijente* po zadatim vrednostima. (APSO)
2. **Sistem** traži *klijente* po zadatim vrednostima. (SO)
3. **Sistem** prikazuje **korisniku** nađene *klijente* i poruku: „Sistem je uspešno pronašao *klijente*“. (IA)
4. **Korisnik** bira klijenta kojeg želi da izmeni. (APUSO)



1. **Korisnik** poziva **sistem** da nađe *klijenta*. (APSO)
2. **Sistem** traži *klijenta*. (SO)
3. **Sistem** prikazuje **korisniku** pronađenog *klijenta* i poruku: „Sistem je uspešno pronašao *klijenta*“. (IA)



1. **Korisnik** unosi nove podatke o *klijentu*. (APUSO)
2. **Korisnik** proverava da li je korektno uneo podatke o *klijentu*. (ANSO)
3. **Korisnik** poziva **sistem** da izmeni *klijenta*. (APSO)
4. **Sistem** menja *klijenta*. (SO)
5. **Sistem** prikazuje **korisniku** poruku: „Sistem je uspešno izmenio *klijenta*“. (IA)

****

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *klijente* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *klijente* po zadatim vrednostima”. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *klijenta* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *klijenta* po zadatim vrednostima”. Prekida se izvršenje scenarija. (IA)

13.1. Ukoliko **sistem** ne može da izmeni izabranog *klijenta* on prikazuje **korisniku** poruku: “**Sistem** ne može da zapamti *klijenta* ”. (IA)

### SK 8: Slučaj korisčenja – Brisanje klijenta

**Naziv SK**

Brisanje *klijenta*

**Aktori SK**

**Zaposleni**

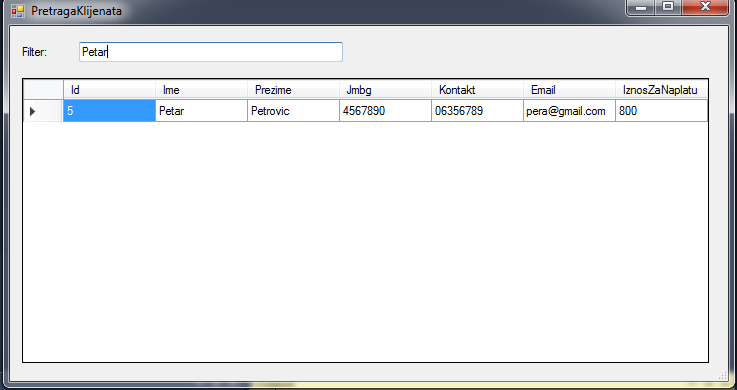
**Učesnici SK**

**Zaposleni** i **sistem** (program)

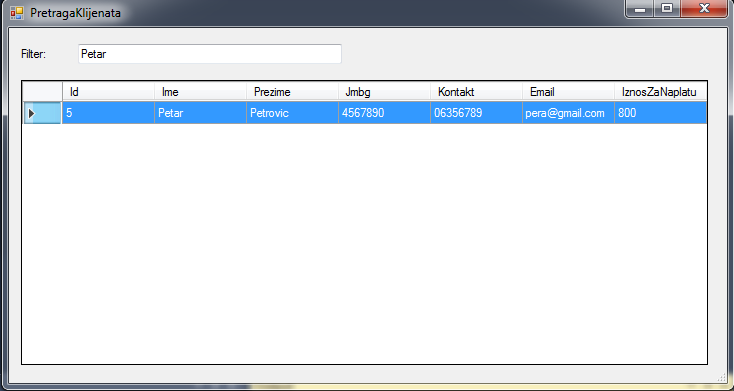
**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za brisanje *klijenta*.

**Osnovni scenario SK**

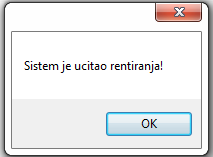
1. **Korisnik** unosi vrednosti po kojima pretražuje *klijente*. (APUSO)



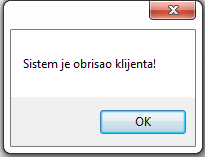
1. **Korisnik** poziva **sistem** da nađe *klijente* po zadatim vrednostima. (APSO)
2. **Sistem** traži *klijente* po zadatim vrednostima. (SO)
3. **Sistem** prikazuje **korisniku** nađene *klijente* i poruku: „Sistem je uspešno pronašao *klijente*“. (IA)
4. **Korisnik** bira klijenta kojeg želi da izmeni. (APUSO)



1. **Korisnik** poziva **sistem** da nađe *klijenta*. (APSO)
2. **Sistem** traži *klijenta*. (SO)
3. **Sistem** prikazuje **korisniku** pronađenog *klijenta* i poruku: „Sistem je uspešno pronašao *klijenta*“. (IA)



1. **Korisnik** poziva **sistem** da obriše *klijenta*. (APSO)
2. **Sistem** briše *klijenta*. (SO)
3. **Sistem** prikazuje **korisniku** poruku: „Sistem je uspešno obrisao izabrano *klijenta*“. (IA)



**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *klijente* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *klijente* po zadatim vrednostima”. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *klijenta* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *klijenta* po zadatim vrednostima”. Prekida se izvršenje scenarija. (IA)

11.1. Ukoliko **sistem** ne može da obriše izabrano *klijenta* on prikazuje **korisniku** poruku: “**Sistem** ne može da obriše izabrano *klijenta* ”. (IA)

### 3.2.3. Projektovanje kontrolera korisničkog interfejsa

Aplikaciona logika služi za opisivanje strukture i ponašanja softverskog sistema i projektuje se nezavisno od korisničkog interfejsa i obrnuto. Drugim rečima, aplikaciona logika (koja predstavlja *Model* u MVC paternu) nema znanja o tome gde se nalazi korisnički interfejs (koja predstavlja *View* u MVC paternu).

Kontroler je odgovoran da prihvati zahtev za izvršenje sistemske operacije od klijenta i da ga prosledi do poslovne logike koja je odgovorna za izvršenje sistemske operacije.



Slika KKI Korisnički interfejs u kontekstu arhitekture softverskog sistema

## 3.3. Projektovanje aplikacione logike

### 3.3.1. Kontroler aplikacione logike

Kontroler aplikacione logike treba da podigne serverski soket koji će da osluškuje mrežu. Kada klijent (klijentski soket) uspostavi konekciju sa kontrolerom (serverskim soketom), tada kontroler treba da generiše nit koja će uspostaviti dvosmernu vezu sa klijentom (ulaznu i izlaznu). Slanje i primanje podataka od klijenta se ostvaruje preko soketa.

Klijent šalje zahtev za izvršenje neke od SO do odgovarajuće niti (koju smo nazvali “Obrada”), koja je povezana sa tim klijentom. “Obrada” prima zahtev i dalje ga preusmerava do klasa koje su odgovorne za izvršenje SO. Nakon izvršenja SO rezultat se vraća do aplikacione logike, odnosno do “Obrade”, koja taj rezultat šalje nazad do klijenta.

**Primer specificnog kontrolera aplikacione logike**

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;

namespace KlijentApp

{

public class Komunikacija

{

TcpClient klijent;

BinaryFormatter formater;

NetworkStream tok;

public bool poveziSeNaServer()

{

try

{

klijent = new TcpClient("localhost", 11000);

tok = klijent.GetStream();

formater = new BinaryFormatter();

return true;

}

catch (Exception)

{

return false;

}

}

public void kraj()

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.Kraj;

formater.Serialize(tok, transfer);

}

public Object kreirajVozilo()

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.KreirajVozilo;

transfer.TransferObjekat = new Vozilo();

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object vratiSveTipove()

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.VratiSveTipove;

transfer.TransferObjekat = new TipVozila();

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object zapamtiVozilo(Vozilo v)

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.ZapamtiVozilo;

transfer.TransferObjekat = v;

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object obrisiVozilo(Vozilo v)

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.ObrisiVozilo;

transfer.TransferObjekat = v;

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object pretraziVozila(Vozilo v)

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.PretraziVozila;

transfer.TransferObjekat = v;

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object ucitajVozilo(Vozilo v)

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.UcitajVozilo;

transfer.TransferObjekat = v;

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object vratiSvaVozila()

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.VratiSvaVozila;

transfer.TransferObjekat = new Vozilo();

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object vratiSveKlijente()

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.VratiSveKlijente;

transfer.TransferObjekat = new Klijent();

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object zapamtiRentiranja(Klijent k)

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.ZapamtiRentiranje;

transfer.TransferObjekat = k;

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object pretraziRentiranja(Klijent r)

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.PretraziRentiranja;

transfer.TransferObjekat = r;

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object ucitajRentiranje(Klijent r)

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.UcitajRentiranje;

transfer.TransferObjekat = r;

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object obrisiKlijenta(Klijent k)

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.ObrisiKlijenta;

transfer.TransferObjekat = k;

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object kreirajKlijenta()

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.KreirajKlijenta;

transfer.TransferObjekat = new Klijent();

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

}

}

### 3.3.2. Sistemske operacije

U prethodnoj fazi, fazi analize, odredili smo ugovore o sistemskim operacijama, pri čemu smo rekli da jedan ugovor opisuje ponašanje jedne sistemske operacije, tako što opisuje ŠTA operacija treba da radi, ali ne i kako. Sada se u fazi projektovanja za svaki od ugovora projektuje konceptualno rešenje (realizacija) SO. To znači da ćemo za svaku klasu odgovornu za izvršenje SO definisati KAKO će se sistemska operacija izvršiti.

Na samom početku projektovanja SO izbeći ćemo aspekte realizacije koji su vezani za konekciju sa bazom, perzistentnost i transakcije.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using Biblioteka;

using Sesija;

namespace SistemskeOperacije

{

public abstract class OpstaSO

{

public Object izvrsiSO(OpstiDomenskiObjekat odo)

{

Object rezultat = null;

Broker.dajSesiju().otvoriKonekciju();

Broker.dajSesiju().zapocniTransakciju();

try

{

rezultat = Izvrsi(odo);

Broker.dajSesiju().potvrdiTransakciju();

}

catch (Exception)

{

Broker.dajSesiju().ponistiTransakciju();

}

finally

{

Broker.dajSesiju().zatvoriKonekciju();

}

return rezultat;

}

public abstract object Izvrsi(OpstiDomenskiObjekat odo);

}

}

**Ugovor UG1: KreirajVozilo**

Operacija: KreirajVozilo(*Vozilo*):signal;

Veza sa SK: SK1

Preduslovi: *Prosto* *vrednosno ograničenje nad objektom Vozilo mora biti zadovoljeno.*

*Strukturno ograničenje nad objektom Vozilo mora biti zadovoljeno.*

Postuslovi: *Kreirano je novo vozilo.*



**Ugovor UG2: ZapamtiVozilo**

Operacija: ZapamtiVozilo(*Vozilo*):signal;

Veza sa SK: SK1, SK3

Preduslovi: *Prosto vrednosno ograničenje nad objektom Vozilo mora biti zadovoljeno.*

*Strukturno ograničenje nad objektom Vozilo mora biti zadovoljeno.*

Postuslovi: *Podaci o vozilu su zapamćeni.*



**Ugovor UG3: PretraziVozila**

Operacija: PretraziVozila(*Vozilo,List<Vozilo>*):signal;

Veza sa SK: SK2, SK3, SK4

Preduslovi: -

Postuslovi: -



**Ugovor UG4:ObrisiVozilo**

Operacija: ObrisiVozilo(*Vozilo*):signal;

Veza sa SK: SK4

Preduslovi: *Strukturno ograničenje nad objektom Vozilo mora biti zadovoljeno.*

Postuslovi: *Vozilo je obrisano.*



**Ugovor UG5: PronadjiVozilo**

Operacija: PronadjiVozilo(*Vozilo*):signal;

Veza sa SK: SK5, SK6

Preduslovi: -

Postuslovi: *-*



**Ugovor UG6: vratiSveZemlje**

Operacija: vratiSveZemlje (*List<Zemlja>*):signal;

Veza sa SK: SK5, SK7

Preduslovi: */*

Postuslovi: */*



**Ugovor UG7: PretraziKlijente**

Operacija: PretraziKlijente(*Klijent, List<Klijent>*):signal;

Veza sa SK: SK5,SK6,SK7

Preduslovi: -

Postuslovi: -



**Ugovor UG8: vratiTipove**

Operacija: vratiTipove (*List<TipVozila>*):signal;

Veza sa SK: SK5,SK7

Preduslovi: */*

Postuslovi: /



**Ugovor UG9: KreirajKlijenta**

Operacija: KreirajKlijenta (*Klijent*):signal;

Veza sa SK: SK5, SK7

Preduslovi: *Prosto vrednosno ograničenje nad objektom Klijent mora biti zadovoljeno.*

*Strukturno ograničenje nad objektom Klijent mora biti zadovoljeno.*

Postuslovi: *Podaci o klijentu su zapamćeni.*



**Ugovor UG10: ZapamtiKlijenta**

Operacija: ZapamtiKlijenta (*Klijent*):signal;

Veza sa SK: SK5, SK7

Preduslovi: *Prosto vrednosno ograničenje nad objektom Klijent mora biti zadovoljeno.*

*Strukturno ograničenje nad objektom Klijent mora biti zadovoljeno.*

Postuslovi: *Podaci o klijentu su zapamćeni.*



**Ugovor UG11: PronadjiKlijenta**

Operacija: PronadjiKlijenta (*Klijent*):signal;

Veza sa SK: SK7,SK8

Preduslovi: -

Postuslovi: -



**Ugovor UG12: ObrisiKlijenta**

Operacija: ObrisiKlijenta(*Klijent*):signal;

Veza sa SK: SK8

Preduslovi: *Strukturno ograničenje nad objektom Klijent mora biti zadovoljeno.*

Postuslovi: *Klijent je obrisan.*



**Ugovor UG13: VratiSvaVozila**

Operacija: VratiSvaVozila (*List<Vozilo>*):signal;

Veza sa SK: SK5, SK7

Preduslovi: */*

Postuslovi: */*





Slika OSO: Klase koje su odgovorne za SO nasleđuju klasu OpstaSO

### 3.3.3. Domenske klase

Na osnovi konceptualnih klasa prave se softverske klase strukture. Svaka klasa sadrži privatna polja atributa, getter-e, setter-e za iste, konstruktore (bilo bezparametrizovani bilo parametrizovani).

Domenske klase implementiraju interfejs *BazniObjekat* kako bi omogućili lakše implementiranje metoda *DatabaseBroker*-a. Na taj način *DatabaseBroker* prima interfejs umesto samih klasa i olakšava kreiranje generičkih upita.

using System;

using System.Collections.Generic;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Domen

{

public interface OpstiDomenskiObjekat

{

string Tabela { get; }

string ID { get; }

string UslovID { get; }

string UslovOpsti { get; }

string Insert { get; }

string Update { get; }

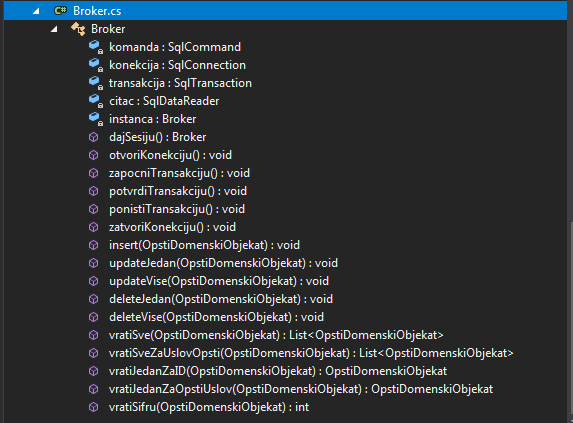
OpstiDomenskiObjekat procitajRed(DataRow red);

}

}

### 3.3.4 Broker baze podataka

Broker baze podataka je softverska klasa odgovorna za komunikacju između poslovne logike i skladišta podataka. Drugim rečima, projektuje se kako bi obezbedio perzistentni servis objektima domenskih klasa koji se čuvaju u bazi podataka. Klasa *DatabaseBroker* predstavlja perzistentni okvir koji posreduje u svim operacijama nad bazom podataka i realizuje sledeće metode:

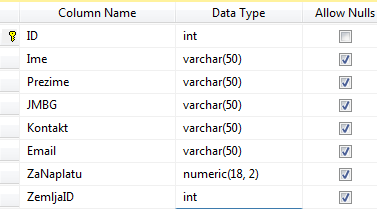




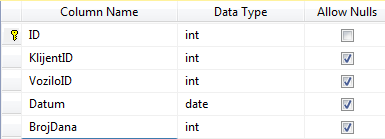
Slika DBBR: Database broker klasa se povezuje sa klasom OpstiDomenskiObjekat

### 3.3.5. Projektovanje skladišta podataka

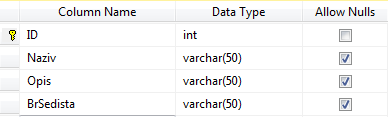
Na osnovi strukture softverskih klasa projektovane su tabele (skladišta podataka) relacionog sistema za upravljanje bazom podataka. U ovom radu je korišćen MS SQL.

****

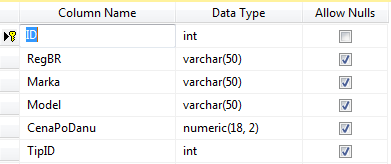
**KLIJENT**

****

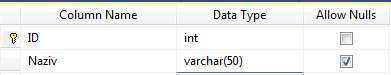
**RENTIRANJE**

****

**TIP VOZILA**

****

**VOZILO**

****

**ZEMLJA**



Slika ASSTBP: Arhitektura soft. sistema nakon projektovanja tabela baze podataka

# 4. IMPLEMENTACIJA

Softverski sistem, rezultat ovog rada, razvijen je u programskom jeziku C#, a izvorni kod je priložen na pratećem CD-u. Sistem je projektovan kao klijent-server. Kao razvojno okruženje korišćen je Microsoft Visual Studio 2017. Kao sistem za upravljanjem bazom podataka korišćen je MS SQL.

Na osnovi arhitekture softverskog sistema dobijene su sledeće softverske klase:

# 

# 

# 5. TESTIRANJE

Svaki od implementiranih slučajeva korišćenja je testiran. Prilikom testiranja svakog slučaja korišćenja pored unetih pravilnih podataka unošeni su i nepravilni podaci da bi se utvrdilo kakav će rezultat izvršenja biti. Na osnovi izvršenih testiranja otklonjeni su uočeni nedostaci.

# 6. LITERATURA

Projektovanje softvera – Skripta, dr Siniša Vlajić, Beograd - 2018.