

FAKULTET TEHNIČKIH NAUKA

UNIVERZITET U NOVOM SADU

WPF PROJEKAT

SALON LEPOTE

JOVANA ZEČEVIĆ IT41/2022

Sadržaj

1. Opis realnog sistema.....	3
2. Zadatak.....	3
3. Opis tehnologija	3
3.1 Microsoft Structured Query Language (SQL).....	3
3.2 Microsoft Visual Studio IDE.....	4
3.3 Microsoft ActiveX Data Objects (ADO).NET.....	4
3.4 Windows Presentation Foundation (WPF)	4
4. UML.....	5
4.1. Dijagram slučajeva upotrebe	5
4.2. Dijagram klasa.....	9
4.3. Dijagram sekvenci	10
5. Baza podataka.....	11
5.1. Kreiranje baze podataka	11
5.2 Upiti i DML naredbe	22
6. Kreiranje WPF aplikacije	22
7. Testiranje aplikacije	48
8. Zaključak.....	51

1. Opis realnog sistema

Ovaj projekat detaljno opisuje sistem rada u salonu lepote, fokusirajući se na ključne operacije kroz koje klijenti prolaze od trenutka dolaska do trenutka kada napuštaju salon. Neke od osnovnih operacija uključuju zakazivanje tretmana i pružanje usluga, dok dodatne uključuju praćenje stanja proizvoda i nabavka. Ovaj sistem je dizajniran tako da efikasno upravlja ovim operacijama.

2. Zadatak

U prvom delu zadatka vrši se modelovanje realnog sistema pomoću UML dijagrama koji na adekvatan način reprezentuju posmatrani sistem ili neki njegov deo. Zatim se, na osnovu tako modelovanog sistema, kreira WPF aplikacija koja simulira rad salona. Informacioni sistem treba da pruži mogućnost prikazivanja podataka o klijentima, zaposlenima, uslugama, vrstama usluga, terminima, zakazivanjima, proizvodima, nabavkama. Takođe, kroz aplikaciju je potrebno omogućiti dodavanje, ažuriranje i brisanje podataka nad klasama od interesa, definisanih dijagramom klasa.

3. Opis tehnologija

U ovom poglavlju predstavljene su tehnologije koje su korišćene u izradi projektnog zadatka.

3.1 Microsoft Structured Query Language (SQL)

Microsoft SQL Server je sistem za upravljanje bazom podataka.

Kada aplikacija postavlja upit, modifikuje i dodaje podatke u bazu podataka, ona koristi Structured Query Language (SQL). SQL je standardni jezik koji se koristi za komunikaciju sa bazama podataka. Kada je SQL server instaliran, uobičajeno postoji jedna instanca. Ova instanca nema ime i pristupa joj se koristeći ime servera. Unutar svake instance može postojati više baza podataka.

3.2 Microsoft Visual Studio IDE

Microsoft Visual Studio je programsko okruženje kreirano za izradu računarskih programa za Microsoft Windows. Takođe se koristi i za izradu veb-sajtova kao i veb i mobilnih aplikacija. Visual Studio koristi platforme kao što su: Windows API, Windows Forms, Windows Presentation Foundation, Windows Store i Microsoft Silverlight. Visual Studio sadrži deo za pisanje izvornog koda programa sa integrisanim debugger-om. Druge alatke su code profiler, forms designer za izradu aplikacija, web designer aplikaciju i druge. Pored alatki i osnovnih funkcionalnosti, moguće je instalirati i dodatke za poboljšanje funkcionalnosti.

3.3 Microsoft ActiveX Data Objects (ADO).NET

ADO.NET je skup klasa za rad sa podacima. Snabdevači podataka su klase koje obezbeđuju mogućnost konektovanja na izvor podataka. Osnovni izvori podataka su SQL Server izvor podataka i OLE DB izvor podataka. U ADO.NET-u postoje dva osnovna načina rada: konektovani i diskonektovani.

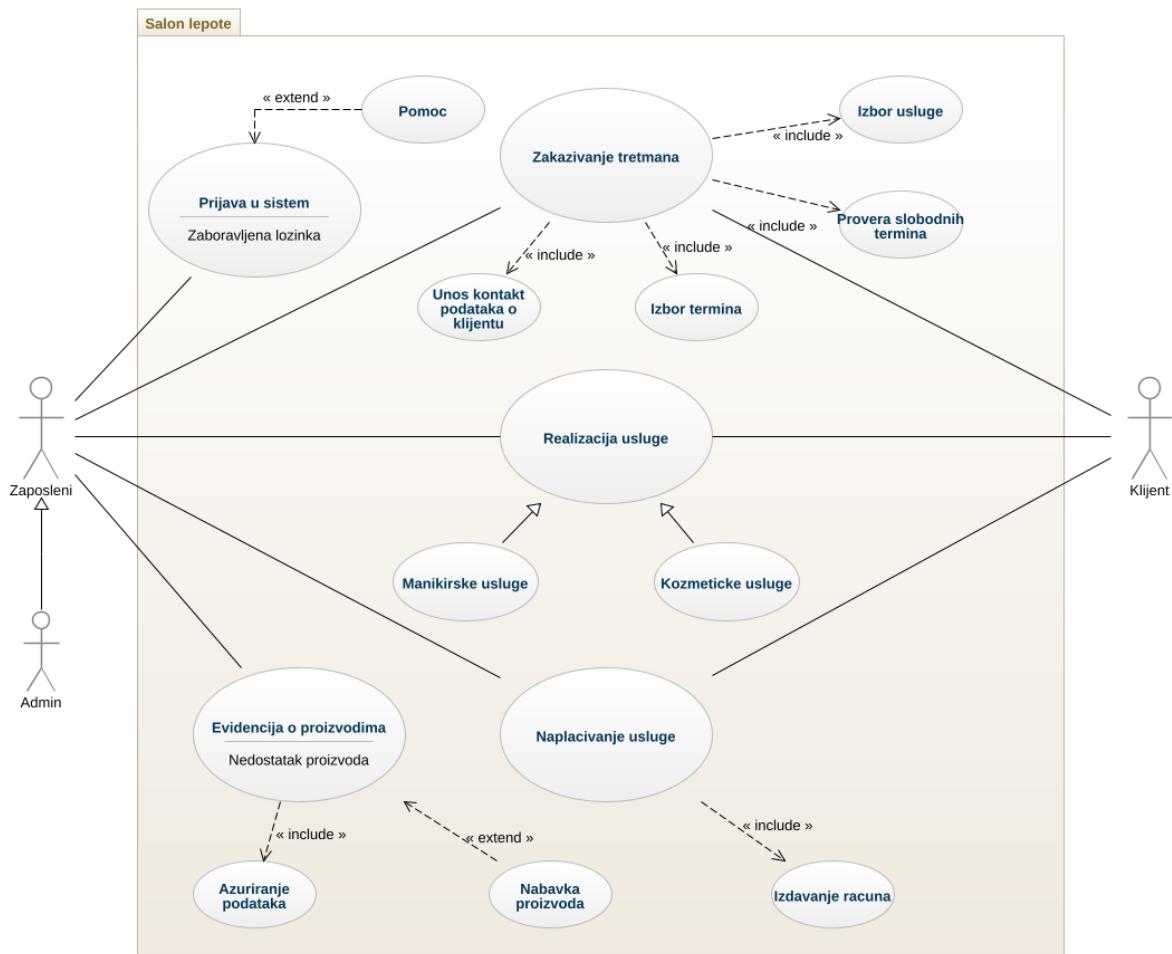
3.4 Windows Presentation Foundation (WPF)

Windows Presentation WPF je grafički podsistem (deo .NET platforme) koji služi za prikazivanje korisničkog interfejsa u aplikacijama koje su zasnovane na Windows operativnom sistemu. Ovaj grafički podsistem daje mogućnost izrade desktop aplikacija ili veb aplikacija. Glavne komponente arhitekture WPF-a su:

- Presentation Framework,
- Presentation Core i
- Milcore

4. UML

4.1. Dijagram slučajeva upotrebe



Slika 4.1 – Dijagram slučajeva upotrebe za informacioni sistem salona lepote

1) **Slučaj upotrebe:** Prijavljivanje u sistem.

Kratak opis: Prijava zaposlenog u sistem.

Učesnici: Zaposleni.

Uslovi koji moraju biti zadovoljeni pre izvršavanja: Zaposleni mora imati validne pristupne podatke (korisničko ime i lozinku).

Opis: Zaposleni unosi svoje korisničko ime i lozinku, zatim bira opciju za prijavu u sistem. Sistem proverava unete podatke kako bi utvrdio njihovu tačnost. **[Izuzetak:** Pogrešno uneto korisničko ime ili pogrešna lozinka]. Ukoliko su podaci tačni, zaposleni dobija pristup funkcionalnostima sistema.

Izuzeci: [Pogrešno uneto korisničko ime ili pogrešna lozinka] Ukoliko zaposleni unese netačno korisničko ime ili lozinku, sistem će obavestiti zaposlenog o neuspešnoj prijavi i zatražiti ponovni unos tačnih podataka.

Uslovi koji moraju biti zadovoljeni nakon izvršavanja: Zaposleni ima pristup funkcionalnostima sistema.

2) **Slučaj upotrebe:** Zakazivanje tretmana.

Kratak opis: Zaposleni zakazuje tretman za klijenta.

Učesnici: Zaposleni, Klijent.

Uslovi koji moraju biti zadovoljeni pre izvršavanja: Zaposleni je prijavljen u sistem.

Opis: Zaposleni pruža klijentu pregled dostupnih usluga, omogućujući mu da odabere željenu. Zatim, zaposleni proverava slobodne termine u rasporedu i prikazuje klijentu. Nakon što klijent izabere željeni termin, zaposleni unosi kontakt podatke kako bi potvrdio zakazivanje.**[Izuzetak:** Nema slobodnih termina za odabranu uslugu].

Izuzeci: [Nema slobodnih termina za odabranu uslugu] U slučaju da nema slobodnih termina za odabranu uslugu, zaposleni će obavestiti klijenta o tome i ponuditi alternative.

Uslovi koji moraju biti zadovoljeni nakon izvršavanja: Klijent je uspešno zakazao tretman, a kontakt podaci su tačno uneti u sistem kako bi se omogućila dalja komunikacija.

3) **Slučaj upotrebe:** Realizacija usluge.

Kratak opis: Zaposleni pruža odabrani tretman klijentu.

Učesnici: Zaposleni, Klijent.

Uslovi koji moraju biti zadovoljeni pre izvršavanja: Zaposleni je prijavljen u sistem, a klijent je prethodno zakazao tretman.

Opis: Nakon što je klijent uspešno zakazao željeni tretman, zaposleni preuzima ključnu ulogu u pružanju visokokvalitetne usluge u skladu sa vrstom tretmana koji je odabran.

[Izuzetak: Nedostatak neophodnih proizvoda ili materijala].

Izuzeci: [Nedostatak neophodnih proizvoda ili materijala] U nekim situacijama, određeni proizvodi ili materijali potrebni za tretman mogu biti privremeno nedostupni. To zahteva prilagođavanje procedure ili može zahtevati alternativni tretman.

Uslovi koji moraju biti zadovoljeni nakon izvršavanja: Klijent mora biti zadovoljan pruženom uslugom.

4) **Slučaj upotrebe:** Naplaćivanje usluge.

Kratak opis: Zaposleni izdaje račun za pruženu uslugu.

Učesnici: Zaposleni, Klijent.

Uslovi koji moraju biti zadovoljeni pre izvršavanja: Tretman je uspešno realizovan.

Opis: Zaposleni izdaje račun koji uključuje cenu tretmana. Klijent plaća iznos koji je naveden na računu.[**Izuzetak: Specifični promotivni događaji ili akcije**].

Izuzeci: [Specifični promotivni događaji ili akcije] Salon može organizovati posebne promotivne događaje ili akcije gde određene usluge mogu biti besplatne ili imati znatno smanjenu cenu za određeno vreme.

Uslovi koji moraju biti zadovoljeni nakon izvršavanja: Tretman je plaćen, a račun evidentiran.

5) ***Slučaj upotrebe:*** Evidencija o proizvodima.

Kratak opis: Vođenje evidencije o dostupnim proizvodima u salonu lepote.

Učesnici: Zaposleni.

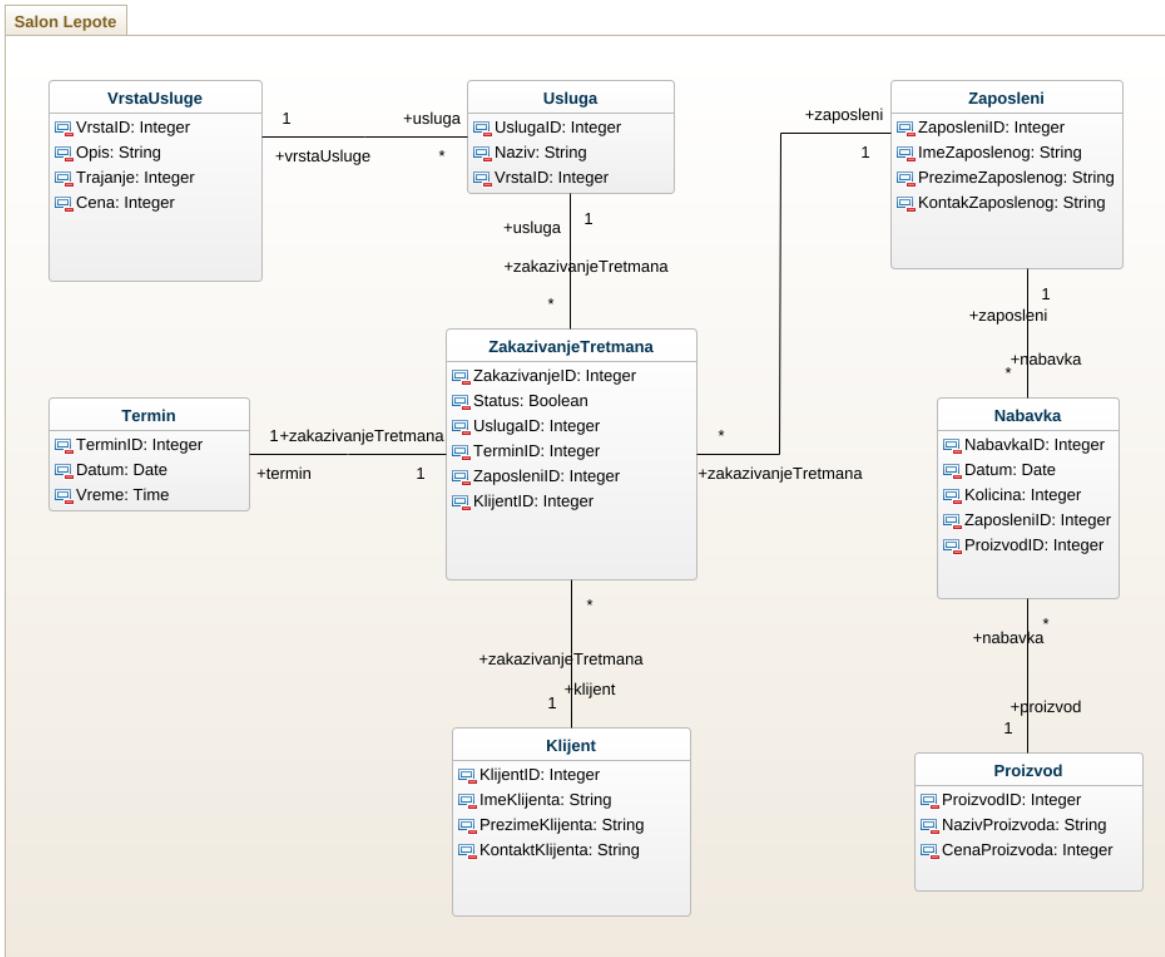
Uslovi koji moraju biti zadovoljeni pre izvršavanja: Proizvodi su tačno identifikovani.

Opis: Zaposleni prati dostupnost proizvoda, ažurira stanje i, u slučaju nedostatka, pokreće nabavku.**[Izuzetak: Nisu dostupne informacije o proizvodu].**

Izuzeci: **[Nisu dostupne informacije o proizvodu]** Proizvod nije unet u sistem, nema dovoljno informacija o njegovom trenutnom stanju ili postoji greška u sistemu.

Uslovi koji moraju biti zadovoljeni nakon izvršavanja: Evidencija proizvoda je ažurirana i tačna.

4.2. Dijagram klasa



Slika 4.2 – Dijagram klasa za informacioni sistem salona lepote

U okviru sistema za salon lepote, identifikovano je nekoliko ključnih klasa koje imaju važnu ulogu u upravljanju tretmanima, klijentima, zaposlenima, uslugama i proizvodima. Svaka od ovih klasa ima specifične atribute koji sadrže relevantne informacije o entitetima u sistemu. Na primer, klasa **ZakazivanjeTretmana** sadrži atribute koji omogućavaju identifikaciju i praćenje svakog zakazivanja. Klasa **Klijent** čuva informacije o klijentima, uključujući ime, prezime, kontakt itd.

U salonu lepote, svako zakazivanje tretmana je povezano sa jednim klijentom i jednim zaposlenim, dok oni mogu imati više zakazivanja tretmana. Takođe, povezano je sa jednom

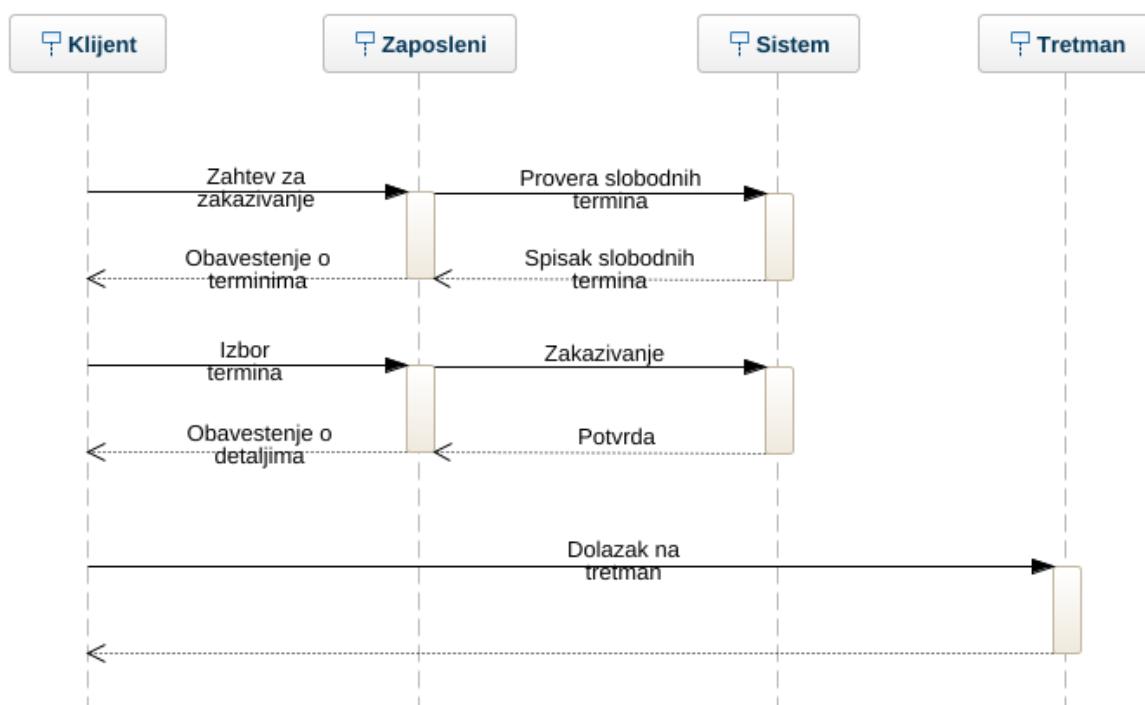
određenom uslугом. Međutim, svaka usluga može biti povezana sa više zakazivanja tretmana. Ona je specifična i pripada tačno jednoj vrsti usluge, dok jedna vrsta usluge može biti povezana sa različitim uslugama.

Zakazivanje je vremenski određeno i vezano za tačno jedan termin, i svaki termin može biti povezan sa tačno jednim zakazivanjem tretmana.

Što se tiče nabavki, jedan zaposleni može biti povezan sa više nabavki, a svaka nabavka ima tačno jednog zaposlenog koji je izvršio nabavku. Nabavka se vrši za određeni proizvod, a za taj proizvod može da se izvrši više nabavki.

Ovaj sistem omogućava efikasnu organizaciju i praćenje svih aktivnosti u salonu lepote.

4.3. Dijagram sekvenci

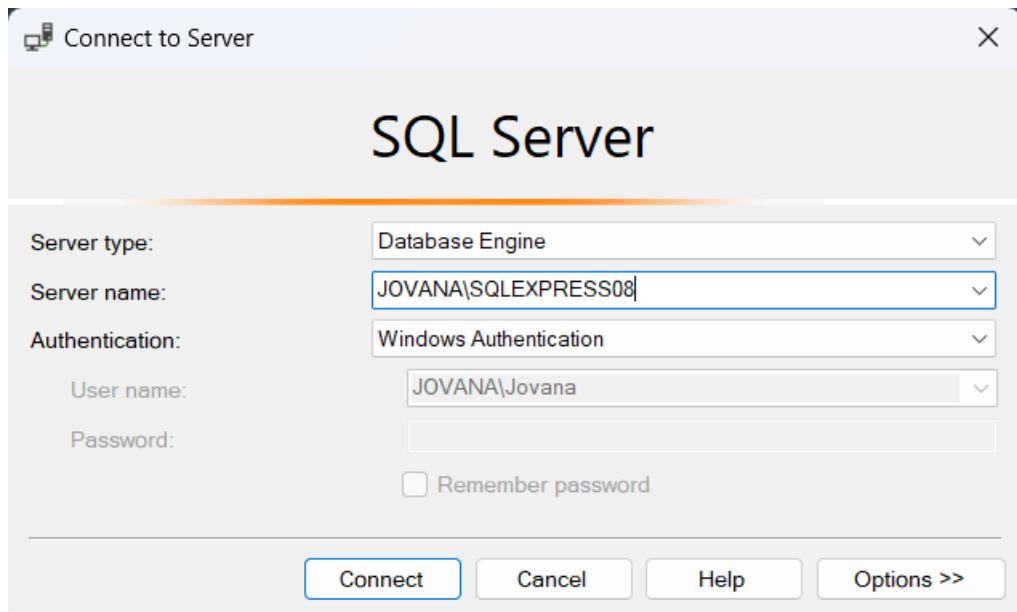


Slika 4.3 – Dijagram sekvenci za informacioni sistem salona lepote

5. Baza podataka

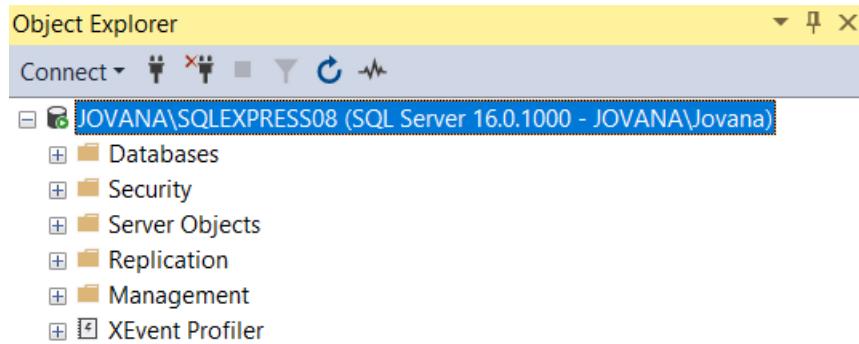
5.1. Kreiranje baze podataka

Sledeći korak u izradi informacionog sistema uključuje stvaranje baze podataka. Na početku, treba otvoriti SQL Server Management Studio Express okruženje. Pojaviće se prozor sa dijalogom, kako je prikazano na slici 5.1. U tom dijalogu, neophodno je podešavanje parametara *Server type* i *Authentication*.



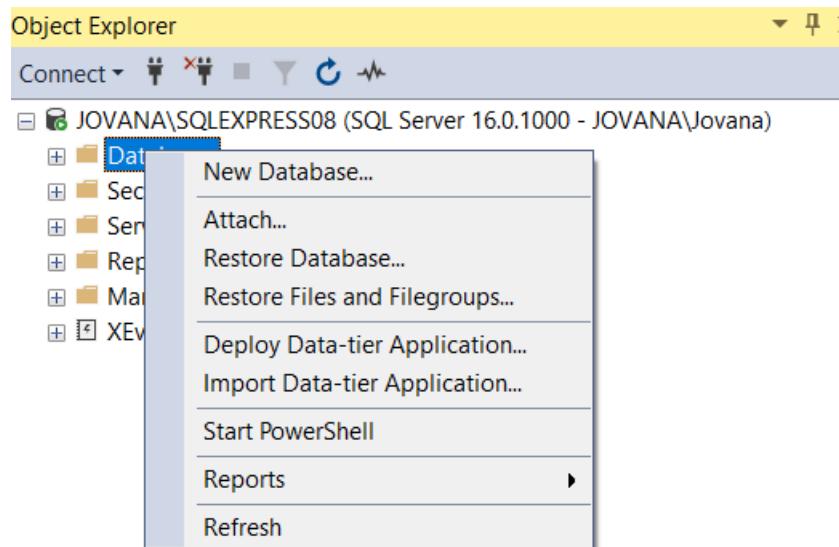
Slika 5.1 – Ostvarivanje konekcije na server

Nakon što se podešavanja unesu, klikom na dugme **Connect**, dijalog prozor će se zatvoriti, a pojaviće se novi prozor čiji izgled treba da odgovara slici 5.2.



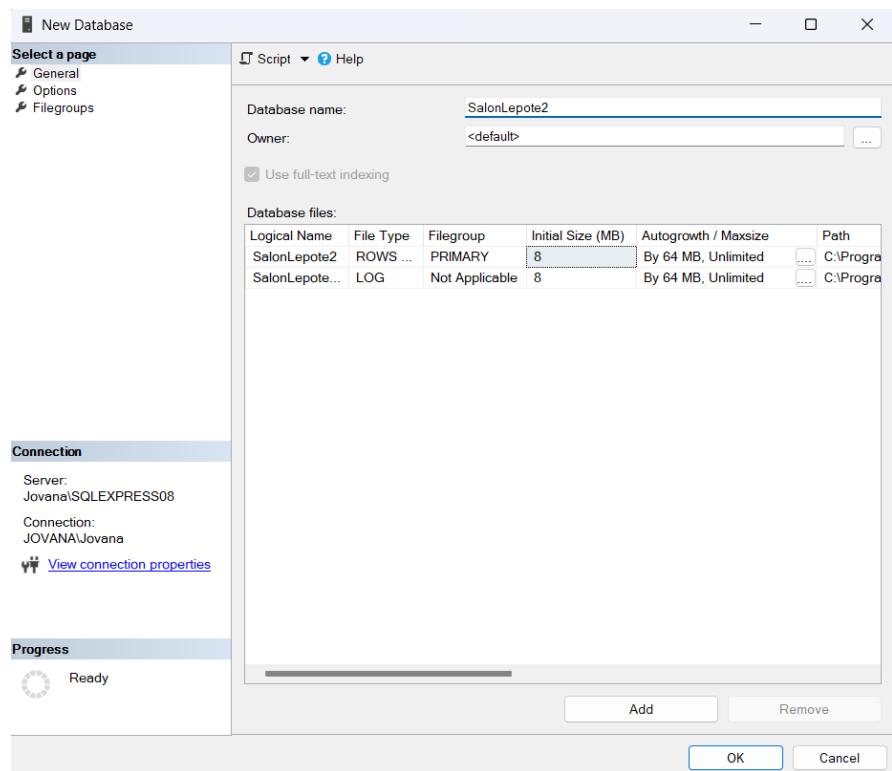
Slika 5.2 – Izgled Object Explorer-a nakon ostvarivanja konekcije sa bazom podataka

Baza podataka se kreira desnim klikom tastera miša na **Database** u treeviewu-u koji se nalazi sa leve strane prozora u okviru Object Explorer-a. Otvoriće se lista u kojoj je potrebno označiti **New Database**.



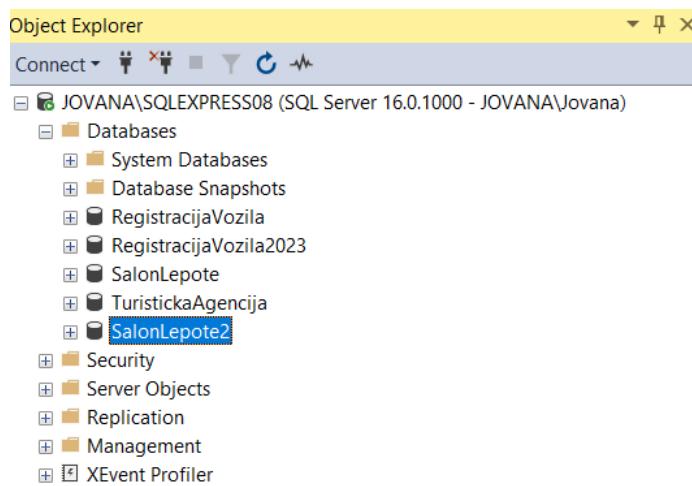
Slika 5.3 – Kreiranje baze podataka

Novi prozor ponudiće opciju za kreiranje baze podataka. Potrebno je dati naziv bazi podataka, u ovom slučaju naziv će biti *Salon lepote*. Ostale parameter nije potrebno menjati.



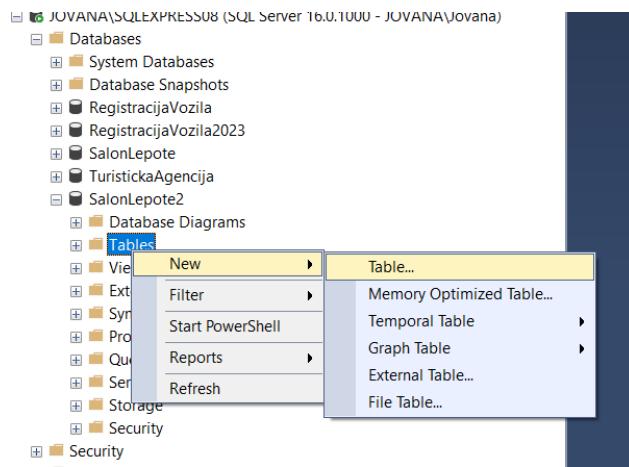
Slika 5.4 – Podešavanje parametara pri kreiranju baze podataka

Posle uspešnog kreiranja baze podataka u Object Explorer-u u folderu Database se pojavljuje kreirana baza podataka.



Slika 5.5 – Prikaz kreirane baze podataka

Proširivanjem baze podataka, u stablu Object Explorer-a, pojavljuje se više datoteka. Desnim klikom tastera miša kliknuti na datoteku Tables i odabratи Table (slika 5.6). Nakon klika na ovu opciju, sa desne strane otvara se prozor za definisanje tabele, pri čemu će tabela imati predefinisani naziv Table_1.



Slika 5.6 – Kreiranje tabele unutar baze podataka

Sledeće što je potrebno uraditi jeste kreirati kolone tabele tako što će se popuniti polja ColumnName, Data Type, Allow Nulls. U kolonu Column Name se upisuje naziv atributa, u koloni Data Type se bira tip podatka za željeni atribut, dok se u koloni Allow Nulls označava da li određeno polje može imat null vrednosti. Preporučuje se da Allow Nulls bude označeno za sve kolone, osim za primarni ključ, kako bi se izbegli određeni problemi prilikom upisa rekorda u tabelu (slika 5.7)

Column Name	Data Type	Allow Nulls
		<input type="checkbox"/>

Slika 5.7 – Dodavanje obeležja unutar tabele

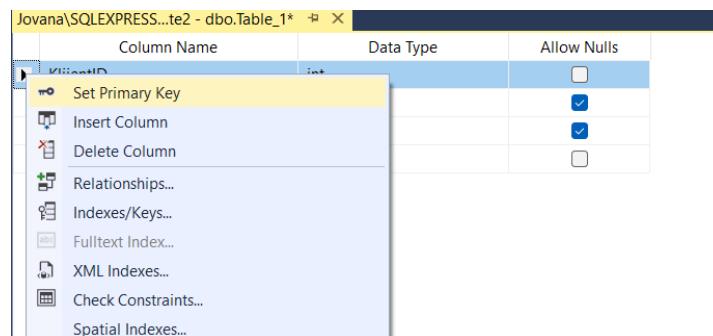
Svaka tabela određene baze podataka mora posedovati primarni ključ - ID i bilo koja kolona može da se deklariše kao primarni ključ. U ovom slučaju, kolona KlijentID treba da se deklariše kao

primarni ključ. Kao takva, ova kolona će zbog svoje jedinstvene vrednosti imati tip podatka int i neće imati null vrednosti (slika 5.8).

Column Name	Data Type	Allow Nulls
KlijentID	int	<input type="checkbox"/>
ImeKlijenta	nchar(10)	<input checked="" type="checkbox"/>
PrezimeKlijenta	nchar(10)	<input checked="" type="checkbox"/>

Slika 5.8 – Prikaz kreiranih obeležja

Da bi se kolona KlijentID deklarisala kao primarni ključ, potrebno je kliknuti desnim klikom tastera miša na crnu strelicu koja se nalazi pored naziva ove kolone. Otvoriće lista u kojoj treba odabrati opciju Set Primary Key (slika 5.9). Ovom komandom se kolona KlijentID deklariše kao primarni ključ.



Slika 5.9 – Definisanje kolone letID za primarni ključ

Da bi sistem za upravljanje bazom podataka - SUBP automatski kreirao primarni ključ, potrebno je u Column Properties pronaći opciju Identity Specification. Proširivanjem opcije Identity Specification dobija se opcija Is Identity, koju treba promeniti na Yes. Ovde je, takođe, moguće podesiti parametre Identity Increment i Identity Seed. Identity Increment-om se definiše za koliko raste sledeća vrednost ključa, dok Identity Seed označava od koje vrednosti počinju da se memorišu ključevi (slika 5.10).

Jovana\SQLEXPRESS0...e - dbo.tblKlijent

Column Name	Data Type	Allow Nulls
KlijentID	int	<input type="checkbox"/>
ImeKlijenta	nvarchar(20)	<input checked="" type="checkbox"/>
PrezimeKlijenta	nvarchar(20)	<input checked="" type="checkbox"/>
KontaktKlijenta	nvarchar(20)	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

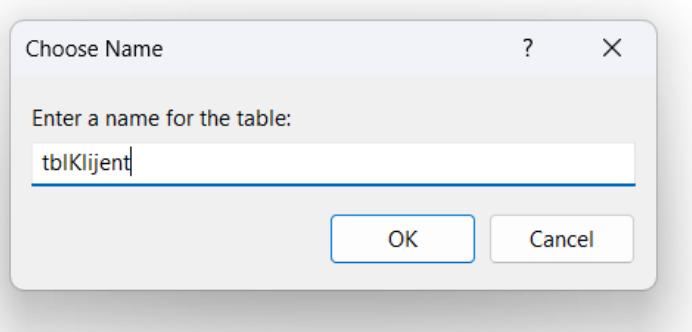
Slika 5.10 – Prikazana kolona KlijentID kao primarni ključ

Column Properties

Property	Value
Identity Specification	Yes
(Is Identity)	Yes
Identity Increment	1
Identity Seed	1
Indexable	Yes
Is Columnset	No

Slika 5.11 – Podešavanje automatske inkrementacije vrednosti obeležja primarnog ključa

Čuvanje tabele vrši se pritiskanjem tastera Ctrl i S nakon čega će SQL Server Management Studio zahtevati unos naziva tabele. Naziv tabele se najčešće navodi sa prefiksom „tbl“, u ovom slučaju tblKlijent.



Slika 5.12 – Čuvanje kreirane tabele

Isti proces treba ponoviti za kreiranje tabela tblZaposleni, tblProizvod, tblNabavka, tblUsluga, tblVrstaUsluge, tblTermin, tblZakazivanjaTretmana.

Column Name	Data Type	Allow Nulls
ZaposleniID	int	<input type="checkbox"/>
ImeZaposlenog	nvarchar(20)	<input checked="" type="checkbox"/>
PrezimeZaposlenog	nvarchar(20)	<input checked="" type="checkbox"/>
KontaktZaposlenog	nvarchar(20)	<input checked="" type="checkbox"/>

Slika 5.13 – Kreiranje tabele tblZaposleni

Column Name	Data Type	Allow Nulls
ProizvodID	int	<input type="checkbox"/>
NazivProizvoda	nvarchar(30)	<input checked="" type="checkbox"/>
CenaProizvoda	int	<input checked="" type="checkbox"/>

Slika 5.14 – Kreiranje tabele tblProizvod

Column Name	Data Type	Allow Nulls
NabavkaID	int	<input type="checkbox"/>
Datum	date	<input checked="" type="checkbox"/>
Kolicina	int	<input checked="" type="checkbox"/>
ZaposleniID	int	<input checked="" type="checkbox"/>
ProizvodID	int	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

Slika 5.15 – Kreiranje tabele tblNabavka

Jovana\SQLEXPRESS...te - dbo.tblUsluga

	Column Name	Data Type	Allow Nulls
PK	UslugaID	int	<input type="checkbox"/>
	Naziv	nvarchar(50)	<input checked="" type="checkbox"/>
	VrstalID	int	<input checked="" type="checkbox"/>

Slika 5.16 – Kreiranje tabele tblUsluga

Jovana\SQLEXPRESS...bo.tbIVrstaUsluga

	Column Name	Data Type	Allow Nulls
PK	VrstalID	int	<input type="checkbox"/>
	Opis	nvarchar(50)	<input checked="" type="checkbox"/>
	Trajanje	int	<input checked="" type="checkbox"/>
	Cena	int	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Slika 5.17 – Kreiranje tabele tbIVrsta

Jovana\SQLEXPRES...kazivanjeTretmana

	Column Name	Data Type	Allow Nulls
PK	ZakazivanjeID	int	<input type="checkbox"/>
	UslugaID	int	<input checked="" type="checkbox"/>
	TerminID	int	<input checked="" type="checkbox"/>
	ZaposleniID	int	<input checked="" type="checkbox"/>
	KlijentID	int	<input checked="" type="checkbox"/>

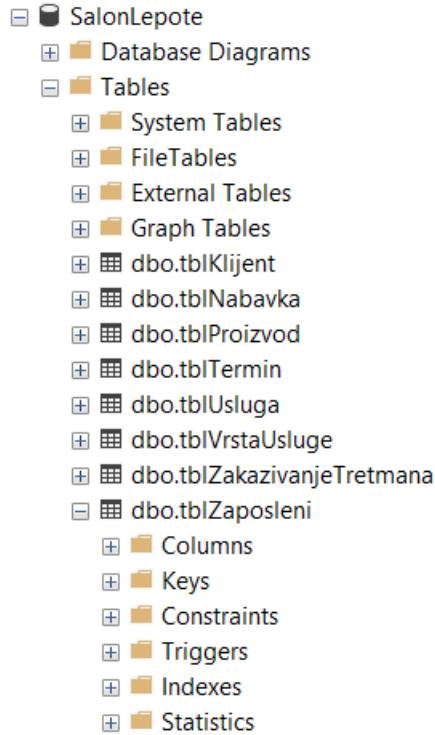
Slika 5.18 – Kreiranje tabele tbIZakazivanjeTretmana

Jovana\SQLEXPRESS...te - dbo.tblTermin

	Column Name	Data Type	Allow Nulls
PK	TerminID	int	<input type="checkbox"/>
	Datum	date	<input checked="" type="checkbox"/>
	Vreme	time(7)	<input checked="" type="checkbox"/>

Slika 5.19 – Kreiranje tabele tbITermin

Nakon toga bi u treeview-u, koji se nalazi sa leve strane prozora u okviru Object Explorer-a, trebala da se nalazi baza podataka Salon Lepote, u kojoj će se, pored već kreirane tblKlijent, naći i novokreirane tabele sa odgovarajućim atributima (slika 5.20).



Slika 5.20 – Object Explorer prikaz kreiranih tabela i kolona

Kao što se može videti na slici 5.20, u tblNabavka se nalaze dva strana ključa (ZaposleniID, PrizvodID). Prilikom unosa atributa u tabelu tblNabavka, potrebno je uneti i atribute koji imaju nazive ZaposleniID i PrizvodID, a vrednost DataType-a za njih staviti na int. Zatim je potrebno kliknuti desnim klikom tastera miša na crnu strelicu koja se nalazi pored naziva ove kolone. Otvoriće se lista u kojoj treba odabrati opciju Relationships (slika 5.21).

Jovana\SQLEXPRESS... - dbo.tblNabavka Jovana\SQLEXPRESS...te - dbo.tblUsluga

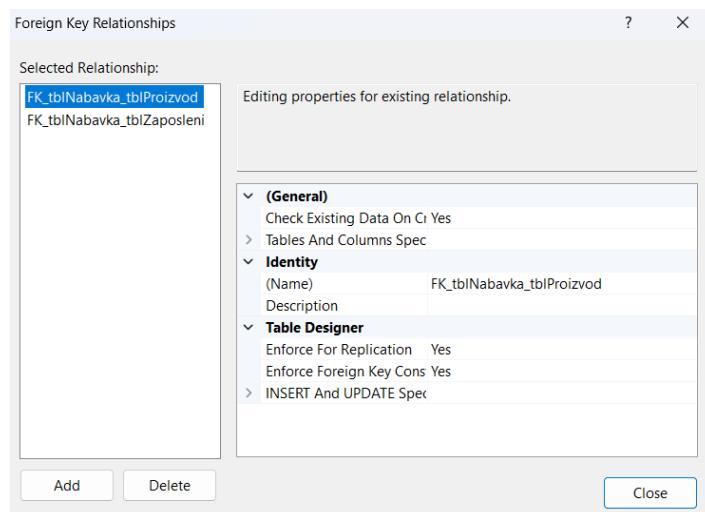
	Column Name	Data Type	Allow Nulls
!	NabavkaID	int	<input type="checkbox"/>
	Datum	date	<input checked="" type="checkbox"/>
	Kolicina	int	<input checked="" type="checkbox"/>
▶	ZaposlenID	int	<input checked="" type="checkbox"/>
	ProizvodID		<input checked="" type="checkbox"/>

▼ Relationships... (highlighted)

- Insert Column
- Delete Column
- Relationships...
- Indexes/Keys...
- Fulltext Index...
- XML Indexes...
- Check Constraints...
- Spatial Indexes...
- Generate Change Script...
- Properties

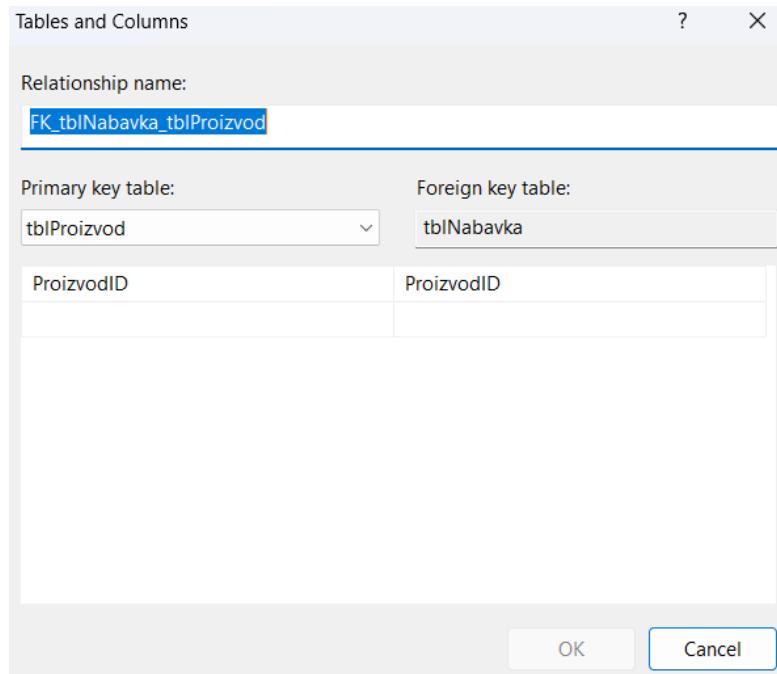
Slika 5.21 – Dodavanje veza

Nakon odabirice opcije Relationships, otvara se prozor Foreign key Relationships (slika 5.22) u kojem je potrebno prvo kliknuti tasterom miša na dugme Add, a zatim proširiti polje Tables and Columns Specifications i kliknuti na tri tačke koje se nalaze desno.



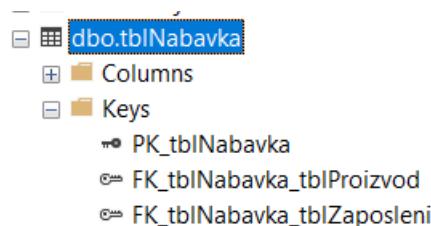
Slika 5.22 – Dodavanje stranog ključa

U otvorenom prozoru Tables and Columns potrebno je u okviru Foreign key table, iz padajuće liste izabrati atribut definisan prilikom kreiranja tabele, a koji je potrebno povezati sa drugom tabelom. Zatim, u okviru Primary key table treba izabrati tabelu sa kojom se vrši povezivanje, i iz padajuće liste odabrati ključ te tabele sa kojim se povezuje željeni atribut. Nakon što je sve ovo urađeno, prozor treba da izgleda kao na sledećoj slici (slika 5.23).



Slika 5.23 – Kreiranje veza između tabela

Dodavanjem Relationships-a atributima ZaposleniID i PID u tabeli tblNabavka, oni postaju strani ključevi. Da li je povezivanje tabele uspešno urađeno znaće se proverom postojanja u treeview-u, u okviru tabele tblNabavka, pod kategorijom Keys (slika 5.24). Isti postputak potrebno je ponoviti za sve tabele koje u sebi sadrže strane ključeve koji su primarni ključevi u drugoj tabeli.



Slika 5.24 – TreeView prikaz kreiranih ključeva

Baza podataka kreirana je na način koji je prikazan kroz celo poglavlje 5. Sledеći korak u izradi projektnog zadatka je kreiranje WPF aplikacije.

5.2 Upiti i DML naredbe

Na slici 5.25 prikazan je kod SQL naredbi sa kojim je izvršen unos inicijalnih pet vrednosti u svaku tabelu u bazu podataka Salon Lepote.

```
--Insert

--Klijenti
insert into tblKlijent (ImeKlijenta, PrezimeKlijenta, KontaktKlijenta)
values ('Ana', 'Jovanovic', '+381641234567');

insert into tblKlijent (ImeKlijenta, PrezimeKlijenta, KontaktKlijenta)
values ('Mila', 'Petrovic', '+381601112233');

insert into tblKlijent (ImeKlijenta, PrezimeKlijenta, KontaktKlijenta)
values ('Jovana', 'Nikolic', '+381641234567');

insert into tblKlijent (ImeKlijenta, PrezimeKlijenta, KontaktKlijenta)
values ('Milica', 'Stojanovic', '+381631122334');

insert into tblKlijent (ImeKlijenta, PrezimeKlijenta, KontaktKlijenta)
values ('Tamara', 'Djordjevic', '+381641122538');

--Zaposleni
insert into tblZaposleni (ImeZaposlenog, PrezimeZaposlenog, KontaktZaposlenog)
values ('Marija', 'Petrovic', '+381641234567');

insert into tblZaposleni (ImeZaposlenog, PrezimeZaposlenog, KontaktZaposlenog)
values ('Ivana', 'Nikolic', '+381601112233');

insert into tblZaposleni (ImeZaposlenog, PrezimeZaposlenog, KontaktZaposlenog)
values ('Teodora', 'Jovanovic', '+381641234567');

insert into tblZaposleni (ImeZaposlenog, PrezimeZaposlenog, KontaktZaposlenog)
values ('Katarina', 'Stojanovic', '+381631122334');

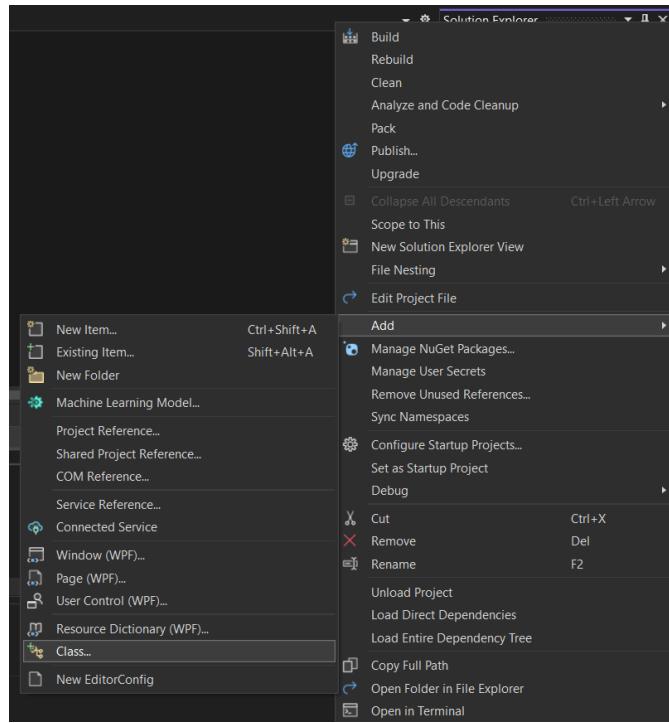
insert into tblZaposleni (ImeZaposlenog, PrezimeZaposlenog, KontaktZaposlenog)
values ('Tamara', 'Ivanovic', '+381631122334');
```

Slika 5.25 – Unos podataka u bazu podataka

6. Kreiranje WPF aplikacije

U okviru ovog poglavlja biće prikazano na koji način je kreirana WPF aplikacija za informacioni sistem Salon Lepote. Nakon pokretanja razvojnog okruženja *Visual Studio*, potrebno je tasterom miša kliknuti na *File menu*, a zatim odabratи *New Project*. Odabratи *Windows Classic Desktop*, zatim *WPF Application*. Navesti ime aplikacije i kliknutи **OK**. U Solution Explorer-u pojaviće se napravljena aplikacija.

Sledeći korak je kreiranje konekcije u aplikaciji, ka bazi podataka. Za početak, kreira se klasa u kojoj će se nalaziti metoda putem koje će se ostvarivati veza na prethodno kreiranu bazu. Klasa se kreira desnim klikom tastera miša na naziv projekta, a zatim **Add/Class** (slika 6.1).

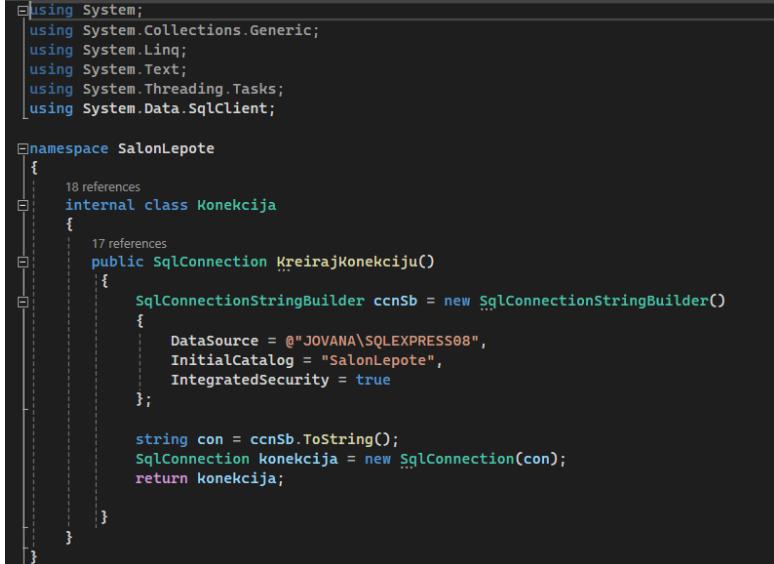


Slika 6.1 – Kreiranje klase

Novokreiranoj klasi se dodeljuje naziv „Konekcija“ i njen sadržaj prikazan je u listingu 6.1. Na samom početku potrebno je uključiti imenski prostor *System.Data.SqlClient*. Telo klase sadrži samo jednu metodu pod nazivom *"KreirajKonekciju"*, koja će se u kasnijem razvoju aplikacije pozivati po potrebi, odnosno svaki put kada je potrebno ostvariti vezu sa bazom podataka, stoga je povratni tip ove metode instanca klase *SqlConnection*. Unutar same metode prikazana su podešavanja parametara konekcije i *Add to Source Control*.

- 1) *DataSource* - naziv servera na kojem je baza podataka smeštena,
- 2) *Initial Catalog* - naziv baze podataka kojoj je potrebno pristupiti i
- 3) *IntegratedSecurity* - ukoliko se baza nalazi na lokalnoj mašini, ovaj parametar postavlja se na true.

Nakon podešavanja ovih parametara, ceo string se prosleđuje instanci klase *SqlConnection*.



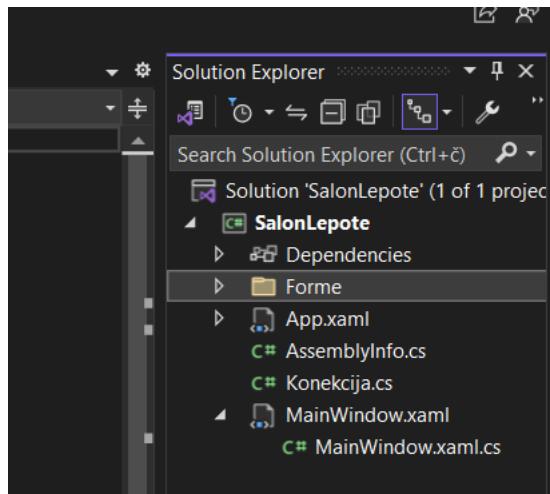
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.SqlClient;

namespace SalonLepote
{
    internal class Konekcija
    {
        public SqlConnection KreirajKonekciju()
        {
            SqlConnectionStringBuilder ccnSb = new SqlConnectionStringBuilder()
            {
                DataSource = @"JOVANA\SQLEXPRESS08",
                InitialCatalog = "SalonLepote",
                IntegratedSecurity = true
            };

            string con = ccnSb.ToString();
            SqlConnection konekcija = new SqlConnection(con);
            return konekcija;
        }
    }
}
```

Listing 6.1 – Klasa Konekcija

Sledeći korak u izradi WPF aplikacije je kreiranje ekranskih formi (eng. Forms) - prozora, za dodavanje objekata u bazu podataka. Kako bi sve forme bile uredno smeštene na jednom mestu, potrebno je kreirati folder pod nazivom Forme u kom će se nalaziti svih osam formi za, prethodno kreiranih, osam tabela u bazi podataka. Kreiranje foldera se vrši na sličan način kao kreiranje klase, desnim klikom tastera miša na projekat, a zatim **Add/New Folder**.

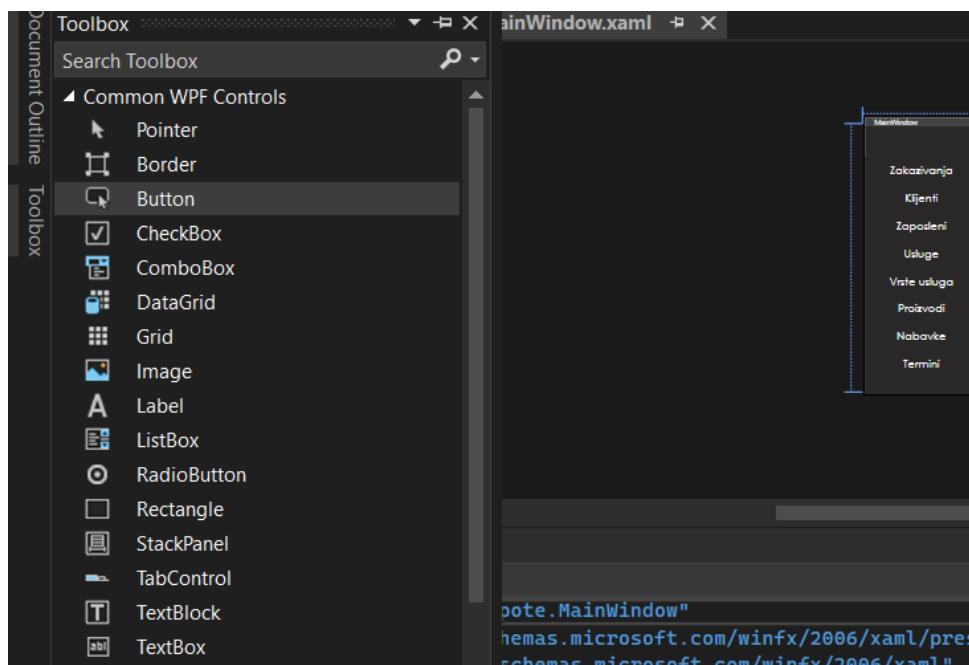


Slika 6.2 – Kreiran folder Forme

Nakon kreiranja foldera, potrebno je dodati novi prozor za svaku od formi. WPF prozor, unutar foldera, kreira se desnim klikom tastera miša na folder, a zatim **Add/Window**.

Nakon toga, prikazaće se prozor za dodavanje novog *Item-a* unutar kog je potrebno podesiti naziv i zatim kliknuti na **Add**.

Kada je novi prozor izgenerisan, potrebno je, unutar njega, dodati određene elemente putem kojih će se ostvarivati potrebne funkcionalnosti. Na levoj strani prozora, klikom tastera miša na Toolbox, prikazaće se lista svih ugrađenih WPF komandi (slika 6.3). Jednostavnim prevlačenjem tih komandi na centralni prozor, izgenerisaće se kod u XAML deklarativnom jeziku unutar kojeg će se vršiti detaljna podešavanja svih komandi unutar prozora.



Slika 6.3 – Dodavanje komandi iz ToolBox-a

Forma za dodavanje novog kljirnta u bazu podataka treba da sadrži elemente prikazane u listingu 6.2. To je osnovni TextBox unutar kog će biti upisana vrednost obeležja. Takođe, uz svaki TextBox, mora postojati naziv (Label) koji će opisivati njegovu namenu. Na kraju, svaka forma mora imati dugme Sačuvaj, na čiji klik tasterom miša će se vrednosti upisivati u bazu podataka, i dugme Poništi, koje korisniku omogućava zatvaranje forme. Ove funkcionalnosti dugmića postižu

se dvostrukim klikom tastera miša na odgovarajuće dugme u okviru forme, čime se u Code Behind-u generišu metode BtnSacuvaj_Click i BtnZatvori_Click.

```
<Window x:Class="SalonLepote.FrmKlijent"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:SalonLepote.Forme"
    mc:Ignorable="d"
    Title="FrmKlijent" Height="450" Width="800">
<Grid>
    <Grid.Background>
        <ImageBrush ImageSource="/Forme/adb816b0f969aa79ca061f351d935ce8.jpg"/>
    </Grid.Background>
    <Label Content="Ime" HorizontalAlignment="Left" Height="37" Margin="186,78,0,0" VerticalAlignment="Top" Width="156" FontFamily="Century Gothic" FontSize="20"/>
    <Label Content="Prezime" HorizontalAlignment="Left" Height="51" Margin="186,150,0,0" VerticalAlignment="Top" Width="177" FontFamily="Century Gothic" FontSize="20"/>
    <Label Content="Kontakt" HorizontalAlignment="Left" Height="53" Margin="186,230,0,0" VerticalAlignment="Top" Width="156" FontFamily="Century Gothic" FontSize="20"/>
    <TextBox Name="txtIme" HorizontalAlignment="Left" Height="40" Margin="363,78,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="160" SelectionBrush="#FFFFAD7FA" FontFamily="Century Gothic" FontSize="18" />
    <TextBox Name="txtPrezime" HorizontalAlignment="Left" Height="40" Margin="363,155,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="160" SelectionBrush="#FFFFAD7FA" />
    <TextBox Name="txtKontakt" HorizontalAlignment="Left" Height="40" Margin="363,230,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="160" SelectionBrush="#FFFFAD7FA" />
    <Button Name="Sacuvaj" Content="Sacuvaj" HorizontalAlignment="Left" Height="51" Margin="140,350,0,0" VerticalAlignment="Top" Width="180" Click="Sacuvaj_Click" FontFamily="Century Gothic" FontSize="20" Background="White" />
    <Button Name="Otkazi" Content="Otkazi" HorizontalAlignment="Left" Height="51" Margin="443,350,0,0" VerticalAlignment="Top" Width="180" Click="Otkazi_Click" FontFamily="Century Gothic" FontSize="18" Background="#FFFFFFCPC" />
</Grid>
</Window>
```

Listing 6.2 – XAML kod kreiranih komponenti

Vizuelni prikaz forme za unos novog klijenta, izgledaće kao na slici 6.4.



Slika 6.4 – Vizuelni prikaz forme za dodavanje/izmenu klijenta

U *Code Behind*-u ove forme, nalazi se sledeća poslovna logika (listing 6.3). Najpre je potrebno uključiti imenski proctor *System.Data.SqlClient*. Zatim, potrebno je napraviti novi objekat *SqlConnection* klase, putem kojeg će se ostvarivati veza ka prethodno kreiranoj bazi podataka, nakon klika tasterom miša na dugme *Sačuvaj*. Takođe, u konstruktoru ovog prozora poželjno je podesiti metodu *Focus()*, koja pozicionira kurzor na prvo polje koje korisnik popunjava pri pokretanju forme. U slučaju forme *frmKlijent*, to je tekst polje za unos imena klijenta.

```

6 references
public partial class FrmKlijent : Window
{
    Konekcija kon = new Konekcija();
    SqlConnection konekcija = new SqlConnection();
    private bool azuriraj;
    private DataRowView red;
    1 reference
    public FrmKlijent()
    {
        InitializeComponent();
        konekcija = kon.KreirajKonekciju();
        txtIme.Focus();
    }

    1 reference
    public FrmKlijent(bool azuriraj, DataRowView red)
    {
        InitializeComponent();
        konekcija = kon.KreirajKonekciju();
        txtIme.Focus();
        this.red = red;
        this.azuriraj = azuriraj;
    }
    1 reference
    private void Otkazi_Click(object sender, RoutedEventArgs e)
    {
        this.Close();
    }
}

```

Listing 6.3 – Sadržaj CodeBehind-a forme Klijent

Ovakav prikaz pozadinskog koda gotovo je identičan za svaku od osam kreiranih formi. Ono u čemu je svaka od ovih formi jedinstvena jeste pozadinska funkcionalnost koja se dešava u trenutku kada korisnik tasterom miša klikne na dugme *Sačuvaj*. U listingu 6.4 prikazan je sadržaj izgenerisane metode *Btn_Sacuvaj_Click*. Na samom početku, potrebno je otvoriti konekciju ka bazi podataka. Zatim, kreira se string koji u sebi sadrži insert into naredbu putem koje se dodaje novi objekat u bazu podataka. U odeljak values ove naredbe, prosleđuju se vrednosti koje su unete u TextBox-ove ove forme. Potrebno je voditi računa da se vrednosti string-ova ograde apostrofima. Nakon uspešno napravljenе insert komande, ovaj string se prosleđuje objektu klase *SqlCommand*.

Ovaj objekat, kao parametre prima prethodno kreirani string, ali i objekat klase Connection putem kojeg će znati u koju bazu podataka treba da upiše prosleđeni zapis. Nakon toga, komanda se izvršava putem metode *ExecuteNonQuery()*, nakon čega se prozor zatvara. Ovaj deo koda potrebno jeograditi try blokom iz razloga što postoji mogućnost nastanka greške usled prosleđivanja unosa pogrešnog tipa podatka od strane korisnika. Kako prilikom ovakvog unosa, program ne bi "pukao", potrebno je dodati catch blok unutar kojeg će se "hvati" izuzetak tipa *SqlException*. Na kraju, u finally bloku, potrebno je postarati se da se prethodno otvorena konekcija sigurno zatvori. Ukoliko konekcija ostane otvorena, niko drugi neće moći da pristupi bazi podataka na isti način dok se ova instanca konekcije ne zatvori.

```

    1 reference
private void Sacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();
        SqlCommand cmd = new SqlCommand
        {
            Connection = konekcija
        };
        cmd.Parameters.Add("@ImeKlijenta", SqlDbType.NVarChar).Value = txtIme.Text;
        cmd.Parameters.Add("@PrezimeKlijenta", SqlDbType.NVarChar).Value = txtPrezime.Text;
        cmd.Parameters.Add("@KontaktKlijenta", SqlDbType.NVarChar).Value = txtKontakt.Text;

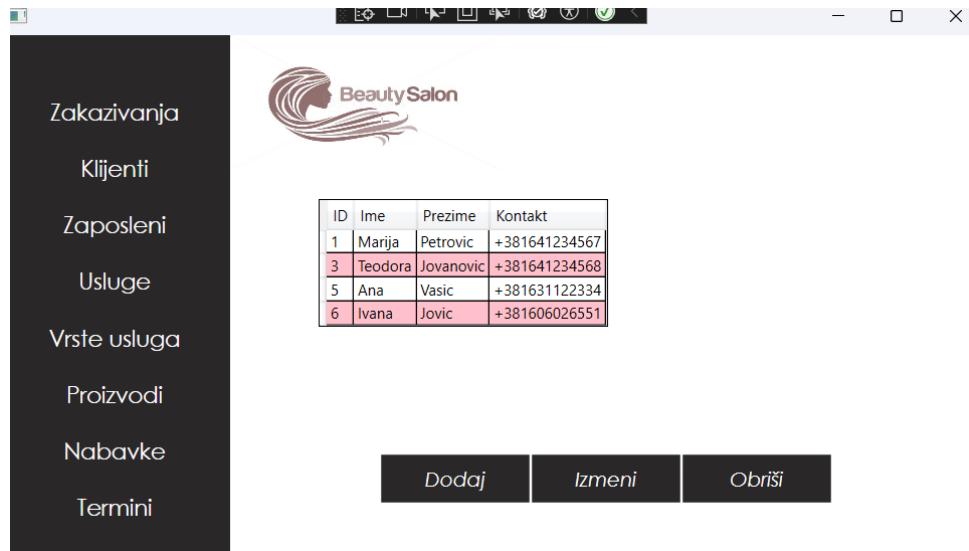
        if (azuriraj)
        {
            cmd.Parameters.Add("@id", SqlDbType.Int).Value = red["ID"];
            cmd.CommandText = @"update tblKlijent set ImeKlijenta=@ImeKlijenta, PrezimeKlijenta=@PrezimeKlijenta, KontaktKlijenta=@KontaktKlijenta
                                where KlijentID = @id";
            red = null;
        }
        else
        {
            cmd.CommandText = @"insert into tblKlijent(ImeKlijenta, PrezimeKlijenta, KontaktKlijenta)
                                values (@ImeKlijenta, @PrezimeKlijenta, @KontaktKlijenta)";
        }

        cmd.ExecuteNonQuery();
        cmd.Dispose(); //sluzi za oslobođenje trenutno zauzetih resursa
        this.Close();
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrijednosti nije validan", "Greska", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    catch (FormatException)
    {
        MessageBox.Show("Doslo je do greske prilikom konverzije podataka", "Greska", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
        {
            konekcija.Close();
        }
    }
}

```

Listing 6.4 – Sadržaj metode BtnSacuvaj_Click za unos novog klijenta

U nastavku će biti prikazan izgled i ostalih sedam formi.



Slika 6.4 – Dizajn forme za dodavanje/izmenu zaposlenog

```

<Window x:Class="SalonLepote.FrmZaposleni"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        xmlns:local="clr-namespace:SalonLepote.Forme"
        mc:Ignorable="d"
        Height="450" Width="800">
    <Grid HorizontalAlignment="Left" Height="434" Margin="449,0,0,0" VerticalAlignment="Center" Width="351">
        <Grid.Background>
            <ImageBrush ImageSource="/Forme/adb816b0f969aa79ca61f351d935ce8.jpg"/>
        </Grid.Background>
        <Button x:Name="Otkazi" Content="Otkaži" HorizontalAlignment="Left" Height="40" Margin="118,325,0,0"
                VerticalAlignment="Top" Width="142" Click="Otkazi_Click" FontFamily="Century Gothic" FontSize="20"
                Background="White" Foreground="Black" FontWeight="Bold" BorderBrush="Black" Padding="5"/>
        <Label Content="Dodajte ili izmenite podatke" HorizontalAlignment="Left" Height="61" Margin="92,92,0,0"
              VerticalAlignment="Top" Width="194" FontFamily="Century Gothic"/>
        <Label Content="Zaposleni" HorizontalAlignment="Center" Height="72" Margin="0,20,0,0"
              VerticalAlignment="Top" Width="27" FontSize="36" FontFamily="Century Gothic" Foreground="Black" FontWeight="Bold"
              HorizontalContentAlignment="Center" VerticalContentAlignment="Center"/>
        <Label Content="Sacuvaj" HorizontalAlignment="Left" Height="40" Margin="142,219,0,0"
              VerticalAlignment="Top" Width="142" Click="Sacuvaj_Click" FontFamily="Century Gothic" FontSize="20" Background="White"/>
        <Label Content="Prezime" HorizontalAlignment="Left" Height="31" Margin="35,130,0,0" VerticalAlignment="Top" Width="178" FontFamily="Century Gothic" FontSize="20"/>
        <Label Content="Ime" HorizontalAlignment="Left" Height="37" Margin="35,61,0,0" VerticalAlignment="Top" Width="156" FontFamily="Century Gothic" FontSize="20"/>
        <Label Content="Kontakt" HorizontalAlignment="Left" Height="53" Margin="35,197,0,0" VerticalAlignment="Top" Width="156" FontFamily="Century Gothic" FontSize="20" RenderTransformOrigin="0.532,0.859"/>
        <TextBox x:Name="txtImeZaposlenog" HorizontalAlignment="Left" Height="40" Margin="232,61,0,0" TextWrapping="Wrap"
                 VerticalAlignment="Top" Width="160" FontFamily="Century Gothic" FontSize="20"/>
        <TextBox x:Name="txtPrezimeZaposlenog" HorizontalAlignment="Left" Height="40" Margin="232,130,0,0" TextWrapping="Wrap"
                 VerticalAlignment="Top" Width="160" FontFamily="Century Gothic" FontSize="20"/>
        <TextBox x:Name="txtKontaktZaposlenog" HorizontalAlignment="Left" Height="40" Margin="232,0,0,0" TextWrapping="Wrap"
                 VerticalAlignment="Center" Width="160" FontFamily="Century Gothic" FontSize="20"/>
    </Grid>
</Grid>

```

Listing 6.5 – XAML kod forme za dodavanje/izmenu zaposlenog

```

private void Sacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();
        SqlCommand cmd = new SqlCommand
        {
            Connection = konekcija
        };
        cmd.Parameters.Add("@ImeZaposlenog", SqlDbType.NVarChar).Value = txtImeZaposlenog.Text; //Označava da je to ta varijabla koja cuva odredjenu vr
        cmd.Parameters.Add("@PrezimeZaposlenog", SqlDbType.NVarChar).Value = txtPrezimeZaposlenog.Text;
        cmd.Parameters.Add("@KontaktZaposlenog", SqlDbType.NVarChar).Value = txtKontaktZaposlenog.Text;

        if (azuriraj)
        {
            cmd.Parameters.Add("@id", SqlDbType.Int).Value = red["ID"];
            cmd.CommandText = @"update tblZaposleni set ImeZaposlenog=@ImeZaposlenog, PrezimeZaposlenog=@PrezimeZaposlenog, KontaktZaposlenog=@KontaktZaposlenog
                                where ZaposleniID = @id";
            red = null;
        }
        else
        {
            cmd.CommandText = @"insert into tblZaposleni(ImeZaposlenog, PrezimeZaposlenog, KontaktZaposlenog)
                                values (@ImeZaposlenog, @PrezimeZaposlenog, @KontaktZaposlenog)";
        }

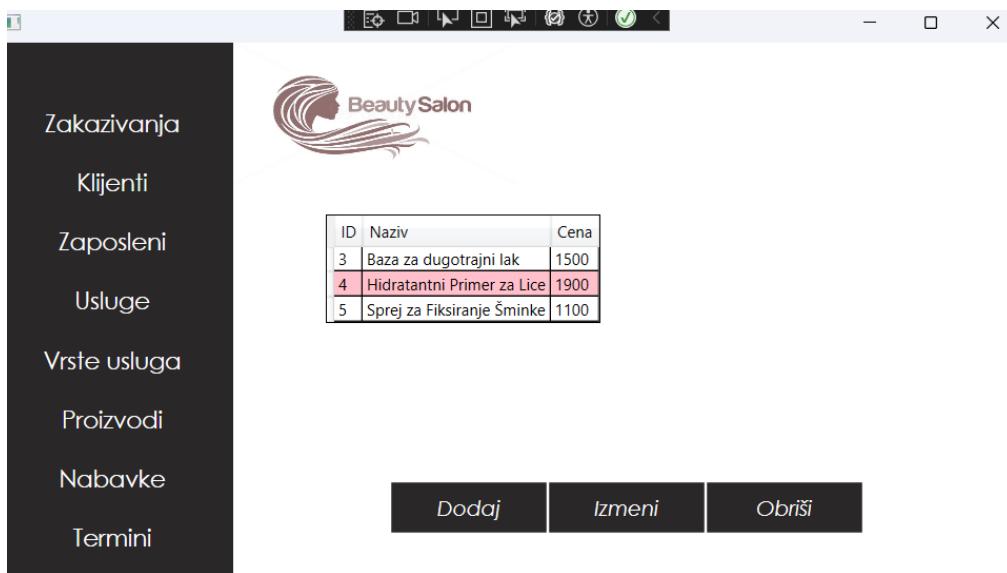
        cmd.ExecuteNonQuery();
        cmd.Dispose(); //sluzi za oslobadjanje trenutno zauzetih resursa
        this.Close();
    }
    catch (SqlException ex)
    {
        MessageBox.Show(ex.Message, "Greska", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    catch (FormatException)
    {
        MessageBox.Show("Doslo je do greske prilikom konverzije podataka", "Greska", MessageBoxButton.OK, MessageBoxImage.Error);
    }

    finally
    {
        if (konekcija != null)
        {
            konekcija.Close();
        }
    }
}

//reference
private void Otkazi_Click(object sender, RoutedEventArgs e)
{
    this.Close();
}

```

Listing 6.6 - Sadržaj metode Sacuvaj_Click za unos novog zaposlenog



Slika 6.5 – Dizajn forme za dodavanje/izmenu proizvoda

```

<Window x:Class="SalonLepote.FrmProizvod"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    Height="450" Width="800">
<Grid HorizontalAlignment="Left" Height="431" Margin="403,0,0,0" VerticalAlignment="Center" Width="357">
    <Grid.Background>
        <ImageBrush ImageSource="/Freme/adb816bf96aa77ca0e1f351d935ce8.jpg"/>
    </Grid.Background>
    <Button x:Name="Otkazi" Content="Otkazi" HorizontalAlignment="Center" Height="40" Margin="0,330,0,0" VerticalAlignment="Top" Width="142" Click="Otkazi_Click"
        BorderBrush="White" Background="White" Foreground="White" FontFamily="Century Gothic" FontSize="20" FontWeight="Bold"/>
    <Label Content="Dodajte ili izmenite podatke" HorizontalAlignment="Center" Height="66" Margin="0,98,0,0" VerticalAlignment="Top" Width="244"
        FontFamily="Century Gothic" HorizontalContentAlignment="Center"/>
    <Label Content="Proizvod" HorizontalAlignment="Left" Height="58" Margin="65,26,0,0" VerticalAlignment="Top" Width="220"
        FontSize="36" FontFamily="Century Gothic" Foreground="White" FontWeight="Bold" HorizontalContentAlignment="Center"/>
    </Grid>
    <Grid HorizontalAlignment="Left" Height="435" Margin="-1,0,0,0" VerticalAlignment="Center" Width="461">
        <Label Content="Naziv" HorizontalAlignment="Left" Height="46" Margin="68,89,0,0" VerticalAlignment="Top"
            Width="175" FontFamily="Century Gothic" FontSize="20"/>
        <Label Content="Cena" HorizontalAlignment="Left" Height="52" Margin="68,166,0,0" VerticalAlignment="Top"
            Width="175" FontFamily="Century Gothic" FontSize="20"/>
        <TextBox x:Name="txtNaziv" HorizontalAlignment="Left" Height="40" Margin="243,89,0,0" TextWrapping="Wrap"
            VerticalAlignment="Top" Width="160" FontFamily="Century Gothic" FontSize="20"/>
        <TextBox x:Name="txtCena" HorizontalAlignment="Left" Height="40" Margin="243,166,0,0" TextWrapping="Wrap"
            VerticalAlignment="Top" Width="160" FontFamily="Century Gothic" FontSize="20"/>
        <Button x:Name="Sacuvaj" Content="Sacuvaj" HorizontalAlignment="Center" Height="40" Margin="0,334,0,0"
            VerticalAlignment="Top" Width="142" Click="Sacuvaj_Click" Background="White" BorderBrush="Black" FontFamily="Century Gothic" FontSize="20"/>
    </Grid>
</Grid>
</Window>

```

Listing 6.7 – XAML kod forme za dodavanje/izmenu proizvoda

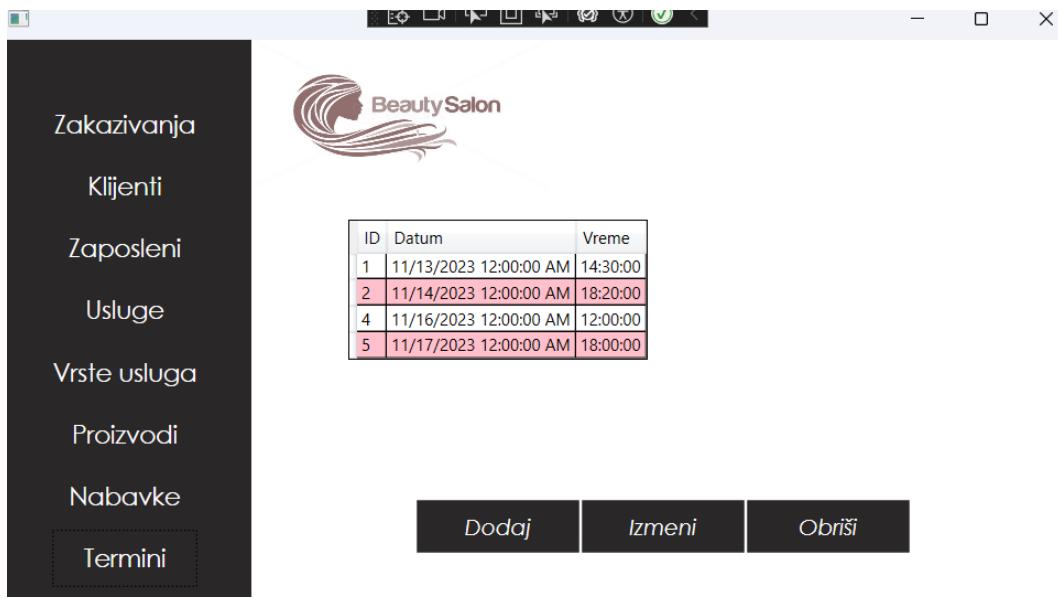
```

private void Sacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();
        SqlCommand cmd = new SqlCommand
        {
            Connection = konekcija
        };
        cmd.Parameters.Add("@NazivProizvoda", SqlDbType.NVarChar).Value = txtNaziv.Text;
        cmd.Parameters.Add("@CenaProizvoda", SqlDbType.Int).Value = txtCena.Text;
        if (cazaraj)
        {
            cmd.Parameters.Add("@id", SqlDbType.Int).Value = red["ID"];
            cmd.CommandText = @"update tblProizvod set NazivProizvoda = @NazivProizvoda, CenaProizvoda=@CenaProizvoda
                                where ProizvodID = @id";
            red = null;
        }
        else
        {
            cmd.CommandText = @"insert into tblProizvod(NazivProizvoda, CenaProizvoda)
                                values (@NazivProizvoda, @CenaProizvoda)";
        }

        cmd.ExecuteNonQuery();
        cmd.Dispose(); //stazi za oslobođanje trenutno zauzetih resursa
        this.Close();
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrijednosti nije validan", "Greska", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    catch (FormatException)
    {
        MessageBox.Show("Doslo je do greske prilikom konverzije podataka", "Greska", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    catch (InvalidOperationException)
    {
        MessageBox.Show("Odaberite datum!", "Greska", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
        {
            konekcija.Close();
        }
    }
}

```

Listing 6.8 - Sadržaj metode Sacuvaj_Click za unos novog proizvoda



Slika 6.6 – Dizajn forme za dodavanje/izmenu termina

```

<Window x:Class="SalonLepote.Frme.FrmTermin"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:SalonLepote.Frme"
    mc:Ignorable="d"
    Height="458" Width="800">
<Grid>
    <Grid HorizontalAlignment="Left" Height="434" Margin="462,0,0,0" VerticalAlignment="Center" Width="338">
        <Grid.Background>
            <ImageBrush ImageSource="/Forme/adb816b0f969aa79ca061f351d935ce8.jpg"/>
        </Grid.Background>
        <Grid.RowDefinitions>
            <RowDefinition Height="215*"/>
            <RowDefinition Height="219*"/>
        </Grid.RowDefinitions>
        <Button x:Name="Otkazi" Content="Otkazi" HorizontalAlignment="Left" Height="40" Margin="107,109,0,0" VerticalAlignment="Top" Width="142" Click="Otkazi_Click" Grid.Row="1" FontFamily="Century Gothic" FontSize="20" Background="#{Null}" BorderBrush="White" Foreground="White" FontWeight="Bold"/>
        <Label Content="Dodajte ili izmenite podatke" HorizontalAlignment="Left" Height="66" Margin="56,90,0,0" VerticalAlignment="Top" Width="244" FontFamily="Century Gothic" HorizontalContentAlignment="Center"/>
        <Label Content="Termin" HorizontalAlignment="Left" Height="58" Margin="65,26,0,0" VerticalAlignment="Top" Width="220" FontSize="36" FontFamily="Century Gothic" Foreground="White" FontWeight="Bold" HorizontalContentAlignment="Center"/>
    </Grid>
    <Grid HorizontalAlignment="Left" Height="434" VerticalAlignment="Center" Width="462">
        <Label Content="Vreme" HorizontalAlignment="Left" Height="42" Margin="68,167,0,0" VerticalAlignment="Top" Width="174" FontFamily="Century Gothic" FontSize="20"/>
        <DatePicker x:Name="dpDatum" HorizontalAlignment="Left" Height="40" Margin="182,89,0,0" VerticalAlignment="Top" Width="160" FontFamily="Century Gothic" FontSize="16"/>
        <Label Content="Datum" HorizontalAlignment="Left" Height="48" Margin="68,89,0,0" VerticalAlignment="Top" FontSize="20" FontFamily="Century Gothic"/>
        <TextBox x:Name="txtVreme" HorizontalAlignment="Left" Height="40" Margin="182,167,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="160" FontFamily="Century Gothic" FontSize="20"/>
        <Button x:Name="Sacuvaj" Content="Sacuvaj" HorizontalAlignment="Center" Height="40" Margin="0,324,0,0" VerticalAlignment="Top" Width="142" Click="Sacuvaj_Click" Background="#{Null}" BorderBrush="Black" FontFamily="Century Gothic" FontSize="20"/>
    </Grid>
</Grid>
</Window>

```

Listing 6.9 – XAML kod forme za dodavanje/izmenu termina

```

    1 reference
private void Sacuvaj_Click(object sender, RoutedEventArgs e)
{
    DateTime date = (DateTime)dpDatum.SelectedDate;
    string datum = date.ToString("yyyy-MM-dd");

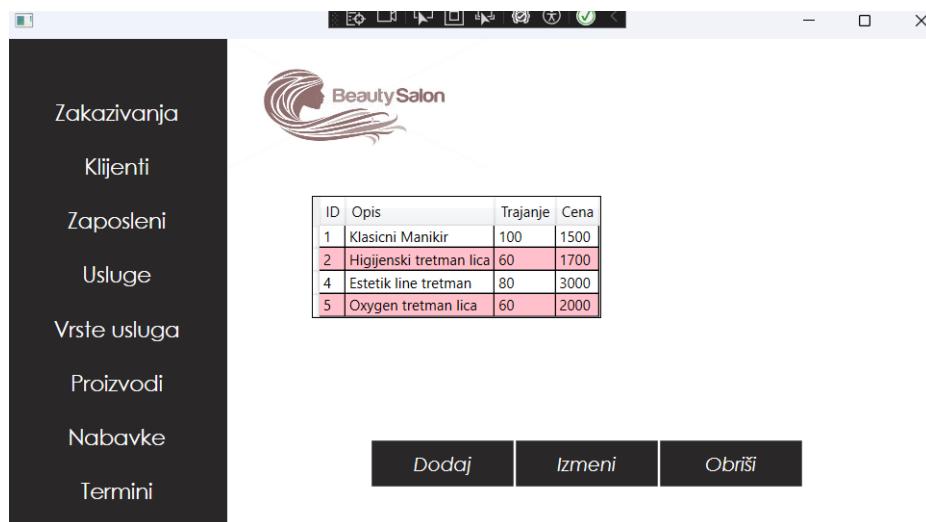
    try
    {
        konekcija.Open();
        SqlCommand cmd = new SqlCommand
        {
            Connection = konekcija
        };
        cmd.Parameters.Add("@Datum", SqlDbType.DateTime).Value = datum;
        cmd.Parameters.Add("@Vreme", SqlDbType.Time).Value = txtVreme.Text; //?????????

        //@ označava da je to ta varijabla koja cuva odredjenu vr
        if (azuriraj)
        {
            cmd.Parameters.Add("@id", SqlDbType.Int).Value = red["ID"]; //ovo u zagradi mora biti isto kao u sql
            cmd.CommandText = @"update tblTermin set Datum=@Datum, Vreme=@Vreme where TerminID = @id";
            red = null;
        }
        else
        {
            cmd.CommandText = @"insert into tblTermin(Datum, Vreme)
                                values (@Datum, @Vreme)";
        }
        cmd.ExecuteNonQuery();
        cmd.Dispose(); //sluzi za oslobođanje trenutno zauzetih resursa
        this.Close();
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrijednosti nije validan", "Greska", MessageBoxButton.OK);
    }
    finally
    {
        if (konekcija != null)
        {
            konekcija.Close();
        }
    }
}

    1 reference
private void Otkazi_Click(object sender, RoutedEventArgs e)
{
    this.Close();
}

```

Listing 6.10 - Sadržaj metode Sacuvaj_Click za unos novog termina



Slika 6.7 - Dizajn forme za dodavanje/izmenu vrste usluge

```

<Window x:Class="SalonLepte.Forme.FrmVrstaUsluge"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:SalonLepte.Forme"
    mc:Ignorable="d"
    Height="450" Width="800">
    <Grid>
        <Grid HorizontalAlignment="Left" Height="434" Margin="436,0,0,0" VerticalAlignment="Center" Width="364">
            <Grid.Background>
                <ImageBrush ImageSource="/Forme/adbb16b0f969aa79ca061f351d935ce8.jpg"/>
            </Grid.Background>
            <Button x:Name="Otkazi" Content="Otkazi" HorizontalAlignment="Left" Height="40" Margin="104,331,0,0" VerticalAlignment="Top" Width="142" Click="Otkazi_Click" Background="#{x:Null}" BorderBrush="White" Foreground="White" FontWeight="Bold" FontFamily="Century Gothic" FontSize="20"/>
            <Label Content="Dodataj ili izmenite podatke" HorizontalAlignment="Left" Height="66" Margin="53,89,0,0" VerticalAlignment="Top" Width="142" Click="Sacuvaj_Click" Background="#{x:Null}" BorderBrush="Black" FontFamily="Century Gothic" FontSize="20"/>
            <Label Content="Usluga" HorizontalAlignment="Left" Height="58" Margin="65,26,0,0" VerticalAlignment="Top" Width="220" FontSize="36" FontFamily="Century Gothic" Foreground="White" FontWeight="Bold" HorizontalContentAlignment="Center"/>
        </Grid>
        <Grid HorizontalAlignment="Left" Height="426" Margin="2,-2,0,0" VerticalAlignment="Top" Width="502">
            <Button x:Name="Sacuvaj" Content="Sacuvaj" HorizontalAlignment="Left" Height="40" Margin="133,333,0,0" VerticalAlignment="Top" Width="142" Click="Sacuvaj_Click" Background="#{x:Null}" BorderBrush="Black" FontFamily="Century Gothic" FontSize="20"/>
            <TextBox x:Name="txtCena" HorizontalAlignment="Center" Height="40" Margin="0,198,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="160" FontFamily="Century Gothic" FontSize="20"/>
            <Label Content="Trajanje" HorizontalAlignment="Left" Height="40" Margin="50,120,0,0" VerticalAlignment="Top" Width="154" FontFamily="Century Gothic" FontSize="20"/>
            <Label Content="Opis" HorizontalAlignment="Left" Height="36" Margin="50,56,0,0" VerticalAlignment="Top" Width="154" FontFamily="Century Gothic" FontSize="20"/>
            <Label Content="Cena" HorizontalAlignment="Left" Height="45" Margin="45,193,0,0" VerticalAlignment="Top" Width="154" FontFamily="Century Gothic" FontSize="20"/>
            <TextBox x:Name="txtOpis" HorizontalAlignment="Center" Height="40" Margin="0,56,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="160" FontFamily="Century Gothic" FontSize="20"/>
            <TextBox x:Name="txtTrajanje" HorizontalAlignment="Center" Height="40" Margin="0,120,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="160" FontFamily="Century Gothic" FontSize="20"/>
        </Grid>
    </Grid>
</Window>

```

Listing 6.11 – XAML kod forme za dodavanje/izmenu vrste usluge

```

    private void Sacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();
        SqlCommand cmd = new SqlCommand
        {
            Connection = konekcija
        };
        cmd.Parameters.AddWithValue("@Opis", SqlDbType.NVarChar).Value = txtOpis.Text; // oznacava da je to ta varijabla koja cuva odredjenu vrstu usluge
        cmd.Parameters.AddWithValue("@Trajanje", SqlDbType.Int).Value = txtTrajanje.Text;
        cmd.Parameters.AddWithValue("@Cena", SqlDbType.Int).Value = txtCena.Text;

        if (azuriraj)
        {
            cmd.Parameters.AddWithValue("@id", SqlDbType.Int).Value = red["ID"];
            cmd.CommandText = @"update tblVrstavUsluge set Opis = @Opis, Trajanje = @Trajanje, Cena = @Cena
                                where VrstavUslugeID = @id";
            red = null;
        }
        else
        {
            cmd.CommandText = @"insert into tblVrstavUsluge(Opis, Trajanje, Cena)
                                values (@Opis, @Trajanje, @Cena)";
        }

        cmd.ExecuteNonQuery();
        cmd.Dispose(); // sluzi za oslobadjanje trenutno zauzetih resursa
        this.Close();
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos odredjenih vrijednosti nije validan", "Greska", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    catch (FormatException)
    {
        MessageBox.Show("Doslo je do greske prilikom konverzije podataka", "Greska", MessageBoxButton.OK, MessageBoxImage.Error);
    }

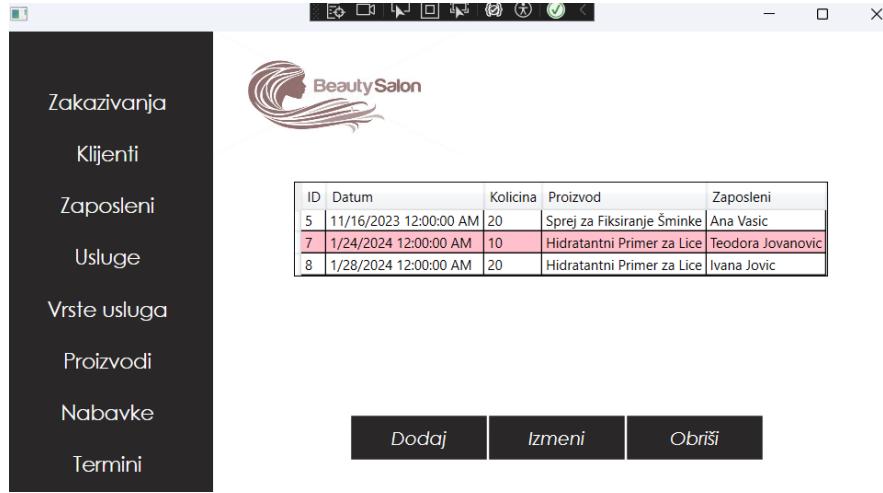
    finally
    {
        if (konekcija != null)
        {
            konekcija.Close();
        }
    }
}
}

```

Listing 6.12 - Sadržaj metode Sacuvaj_Click za unos nove vrste usluge

Tabele tblNabavka, tblZakazivanjeTretmana i tblUsluga, u sebi sadrže strane ključeve drugih tabela na osnovu kojih se pristupa sadržajima tih tabela. U samoj aplikaciji, potrebno je korisniku

obezbediti vizuelni prikaz željenih obeležja neke od povezanih tabela. U WPF tehnologiji se to postiže korišćenjem ComboBox-a. Vizuelni prikaz forme za dodavanje nove nabavke prikazan je na slici 6.7.



Slika 6.7 - Dizajn forme za dodavanje/izmenu nabavke

```

<Window x:Class="SalonLepote.FrmNabavke"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        mc:Ignorable="d"
        Height="450" Width="800">

    <Grid>
        <Grid HorizontalAlignment="Left" Height="434" Margin="450,0,0,0" VerticalAlignment="Center" Width="350">
            <Grid Background="Black">
                <ImageBrush ImageSource="/Forme/adb816b0ff969aa79ca961f351d935ce8.jpg"/>
            </Grid>
            <Button x:Name="Otkazi" Content="Otkazi" HorizontalAlignment="Center" Height="40" Margin="0,337,0,0" VerticalAlignment="Top" Width="142" Click="Otkazi_Click" BorderBrush="White" Foreground="White" FontWeight="Bold" Background="White" FontFamily="Century Gothic" FontSize="20"/>
            <Label Content="Nabavka" HorizontalAlignment="Center" Height="58" Margin="0,26,0,0" VerticalAlignment="Top" Width="142" FontSize="36" FontFamily="Century Gothic" Foreground="White" FontWeight="Bold" HorizontalContentAlignment="Center"/>
            <Label Content="Dodajte ili izmenite podatke" HorizontalAlignment="Center" Height="66" Margin="0,84,0,0" VerticalAlignment="Top" Width="244" FontFamily="Century Gothic" HorizontalContentAlignment="Center"/>
        </Grid>
        <Grid HorizontalAlignment="Left" Height="41" Margin="1,3,0,0" VerticalAlignment="Top" Width="469">
            <Button x:Name="Sacuvaj" Content="Sacuvaj" HorizontalAlignment="Center" Height="40" Margin="0,328,0,0" VerticalAlignment="Top" Width="142" Click="Sacuvaj_Click" BorderBrush="Black" Foreground="Black" FontFamily="Century Gothic" FontSize="20"/>
            <Label Content="Datum" HorizontalAlignment="Left" Height="45" Margin="32,30,0,0" VerticalAlignment="Top" Width="159" FontFamily="Century Gothic" FontSize="20"/>
            <Label Content="Kolicina" HorizontalAlignment="Left" Height="44" Margin="32,100,0,0" VerticalAlignment="Top" Width="159" FontFamily="Century Gothic" FontSize="16"/>
            <DatePicker x:Name="dpDatum" HorizontalAlignment="Left" Height="40" Margin="169,30,0,0" VerticalAlignment="Top" Width="160" FontSize="16" FontFamily="Century Schoolbook"/>
            <TextBox x:Name="txtKolicina" HorizontalAlignment="Left" Height="40" Margin="169,100,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="160" FontFamily="Century Gothic" FontSize="20"/>
            <ComboBox x:Name="cbZaposleni" HorizontalAlignment="Left" Height="40" Margin="169,176,0,0" VerticalAlignment="Top" Width="160" SelectedValuePath="ZaposleniID" DisplayMemberPath="zaposleni" FontFamily="Century Gothic" FontSize="16"/>
            <ComboBox x:Name="cbProizvod" HorizontalAlignment="Left" Height="40" Margin="169,244,0,0" VerticalAlignment="Top" Width="160" SelectedValuePath="ProizvodID" DisplayMemberPath="NazivProizvoda" FontSize="16" FontFamily="Century Gothic">
                <ComboBox.Background>
                    <LinearGradientBrush EndPoint="0,1">
                        <GradientStop Color="#FFFF00FF" />
                        <GradientStop Color="White" Offset="1" />
                    </LinearGradientBrush>
                </ComboBox.Background>
            </ComboBox>
        </Grid>
    </Grid>
</Window>
    
```

Listing 6.13 – XAML kod forme za dodavanje/izmenu nabavke

```

    references
private void PopuniPadajuceListe()
{
    try
    {

        konekcija.Open();
        string vratiZaposlene = @"select ZaposleniID, ImeZaposlenog + ' ' + PrezimeZaposlenog as zaposleni, KontaktZaposlenog from tblZaposleni";
        SqlDataAdapter daZaposleni = new SqlDataAdapter(vratiZaposlene, konekcija);
        DataTable dtZaposleni = new DataTable();
        daZaposleni.Fill(dtZaposleni);
        cbZaposleni.ItemsSource = dtZaposleni.DefaultView;
        daZaposleni.Dispose();
        dtZaposleni.Dispose();

        string vratiProizvode = @"select ProizvodID, NazivProizvoda, CenaProizvoda from tblProizvod";
        SqlDataAdapter daProizvodi = new SqlDataAdapter(vratiProizvode, konekcija);
        DataTable dtProizvodi = new DataTable();
        daProizvodi.Fill(dtProizvodi);
        cbProizvod.ItemsSource = dtProizvodi.DefaultView;
        daProizvodi.Dispose();
        dtProizvodi.Dispose();

    }
    catch (SqlException)
    {
        MessageBox.Show("Padajuce liste nisu popunjene", "Greska", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
        {
            konekcija.Close();
        }
    }
}

```

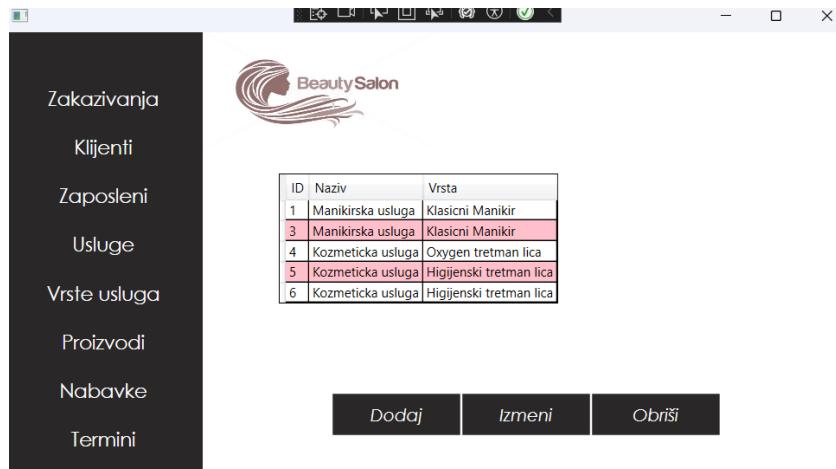
Listing 6.14 - Kod metode PopuniPadajuceListe

```

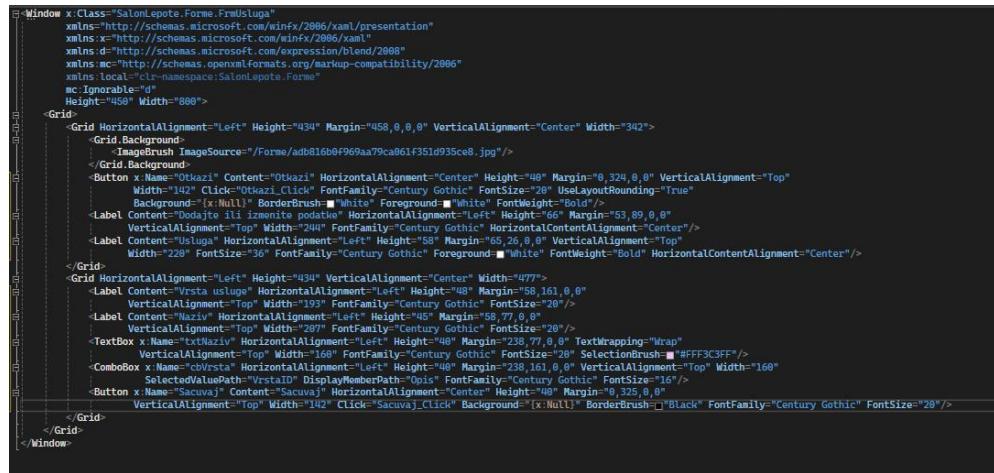
    references
private void Sacuvaj_Click(object sender, RoutedEventArgs e)
{
    DateTime date = (DateTime)dpDatum.SelectedDate;
    string datum = date.ToString("yyyy-MM-dd");
    try
    {
        konekcija.Open();
        SqlCommand cmd = new SqlCommand
        {
            Connection = konekcija
        };
        cmd.Parameters.Add("@Datum", SqlDbType.DateTime).Value = datum;
        cmd.Parameters.Add("@Kolicina", SqlDbType.Int).Value = txtKolicina.Text;
        cmd.Parameters.Add("@ZaposleniID", SqlDbType.Int).Value = cbZaposleni.SelectedValue;
        cmd.Parameters.Add("@ProizvodID", SqlDbType.Int).Value = cbProizvod.SelectedValue;
        if (azuriraj)
        {
            cmd.Parameters.Add("@id", SqlDbType.Int).Value = red["ID"];
            cmd.CommandText = @"update tblNabavka set Datum=@Datum,Kolicina=@Kolicina, ZaposleniID=@ZaposleniID, ProizvodID=@ProizvodID where NabavkaID = @id";
            red = null;
        }
        else
        {
            cmd.CommandText = @"insert into tblNabavka(Datum, Kolicina, ZaposleniID, ProizvodID)
                                values (@Datum, @Kolicina, @ZaposleniID, @ProizvodID)";
        }
        cmd.ExecuteNonQuery();
        cmd.Dispose(); //sluzi za oslobadjanje trenutno zauzetih resursa
        this.Close();
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos odredjenih vrijednosti nije validan", "Greska", MessageBoxButton.OK);
    }
    finally
    {
        if (konekcija != null)
        {
            konekcija.Close();
        }
    }
}

```

Listing 6.15 - Sadržaj metode Sacuvaj_Click za unos nove nabavke



Slika 6.8 - Dizajn forme za dodavanje/izmenu usluge



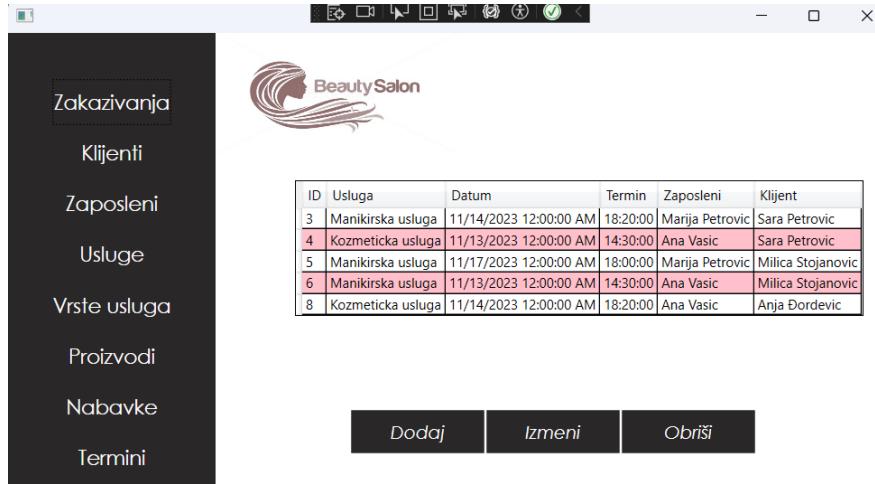
Listing 6.16 – XAML kod forme za dodavanje/izmenu usluge

```

private void Sacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();
        SqlCommand cmd = new SqlCommand
        {
            Connection = konekcija
        };
        cmd.Parameters.AddWithValue("@Naziv", SqlDbType.VarChar).Value = txtNaziv.Text;
        cmd.Parameters.AddWithValue("@VrstaID", SqlDbType.Int).Value = intVrstaID;
        cmd.CommandText = "UPDATE tblUsluga SET Naziv = @Naziv, VrstaID = @VrstaID
        WHERE UslugaID = @ID";
        cmd.Parameters.AddWithValue("@ID", SqlDbType.Int).Value = refID;
        cmd.ExecuteNonQuery();
        cmd.Dispose();
        this.Close();
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrijednosti nije validan.", "Greška", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    catch (FormatException)
    {
        MessageBox.Show("Unos je do greske prilikom konverzije podataka.", "Greška", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    catch (InvalidOperationException)
    {
        MessageBox.Show("Unaberite datum.", "Greška", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
        {
            konekcija.Close();
        }
    }
}

```

Listing 6.17 - Sadržaj metode Sacuvaj_Click za unos nove usluge



Slika 6.9 - Dizajn forme za dodavanje/izmenu zakazivanja tretmana

```

<Window x:Class="SalonLepote.FrmeZakazivanjeTretmana"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    Height="558" Width="800">
    <Grid>
        <Grid HorizontalAlignment="Left" Height="434" Margin="460,0,0,0" VerticalAlignment="Center" Width="340">
            <Grid.Background>
                <ImageBrush ImageSource="/Forme/adb816b0f969aa79ca061f351d935ce8.jpg"/>
            </Grid.Background>
            <Button x:Name="Otkazi" Content="Otkazi" HorizontalAlignment="Center" Height="40" Margin="0,367,0,0"
                    VerticalAlignment="Top" Width="142" Click="Otkazi_Click" FontFamily="Century Gothic" FontSize="20"/>
            <Label Content="Dodajte ili izmenite podatke" HorizontalAlignment="Left" Height="66" Margin="41,84,0,0"
                  VerticalAlignment="Top" Width="244" FontFamily="Century Gothic" HorizontalContentAlignment="Center"/>
            <Label Content="Zakazivanje" HorizontalAlignment="Left" Height="58" Margin="65,26,0,0" VerticalAlignment="Top"
                  Width="220" FontSize="36" FontFamily="Century Gothic" Foreground="White" FontWeight="Bold" HorizontalContentAlignment="Center"/>
        </Grid>
        <Grid HorizontalAlignment="Left" Height="432" Margin="4,0,0,0" VerticalAlignment="Center" Width="990">
            <Button x:Name="Sacuvaj" Content="Sacuvaj" HorizontalAlignment="Left" Height="40" Margin="134,369,0,0"
                    VerticalAlignment="Top" Width="142" Click="Sacuvaj_Click" FontFamily="Century Gothic" FontSize="20" Background="{x:Null}" BorderBrush="Black"/>
            <ComboBox x:Name="cbKlijent" HorizontalAlignment="Left" Height="40" Margin="221,283,0,0"
                      VerticalAlignment="Top" Width="160" SelectedValuePath="KlijentID" DisplayMemberPath="Klijent" FontFamily="Century Gothic" FontSize="16"/>
            <ComboBox x:Name="cbTermin" HorizontalAlignment="Left" Height="40" Margin="221,115,0,0"
                      VerticalAlignment="Top" Width="160" SelectedValuePath="TerminID" DisplayMemberPath="Datum" FontFamily="Century Gothic" FontSize="16"/>
            <Label Content="Usluga" HorizontalAlignment="Left" Height="40" Margin="83,35,0,0"
                  VerticalAlignment="Top" FontFamily="Century Gothic" FontSize="20"/>
            <Label Content="Termin" HorizontalAlignment="Left" Height="40" Margin="83,115,0,0"
                  VerticalAlignment="Top" Width="133" FontFamily="Century Gothic" FontSize="20"/>
            <Label Content="Zaposleni" HorizontalAlignment="Left" Height="40" Margin="83,196,0,0"
                  VerticalAlignment="Top" Width="133" FontFamily="Century Gothic" FontSize="20"/>
            <Label Content="Naziv" HorizontalAlignment="Left" Height="40" Margin="83,279,0,0" VerticalAlignment="Top" Width="133" FontFamily="Century Gothic" FontSize="20"/>
            <ComboBox x:Name="cbUsluga" HorizontalAlignment="Left" Height="40" Margin="221,48,0,0"
                      VerticalAlignment="Top" Width="160" SelectedValuePath="UslugaID" DisplayMemberPath="Naziv" FontFamily="Century Gothic" FontSize="16"/>
            <ComboBox x:Name="cbZaposleni" HorizontalAlignment="Left" Height="40" Margin="221,8,0,0"
                      VerticalAlignment="Center" Width="160" SelectedValuePath="ZaposleniID" DisplayMemberPath="ImeZaposlenog" FontFamily="Century Gothic" FontSize="16"/>
        </Grid>
    </Grid>
</Window>

```

Listing 6.18 – XAML kod forme za dodavanje/izmenu zakazivanja tretmana

```

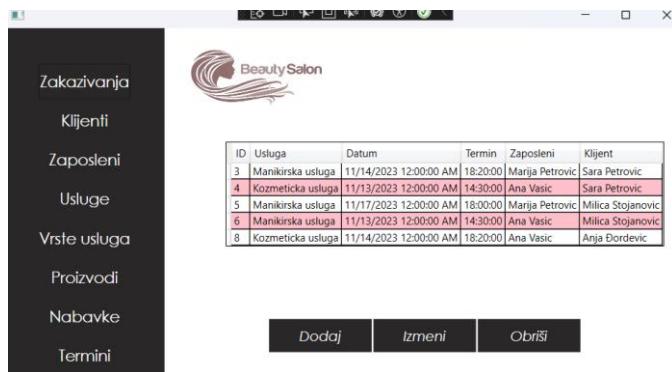
private void Sacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();
        SqlCommand cmd = new SqlCommand
        {
            Connection = konekcija
        };
        cmd.Parameters.AddWithValue("@UslugaID", SqlDbType.Int).Value = cbUsluga.SelectedValue;
        cmd.Parameters.AddWithValue("@TerminID", SqlDbType.Int).Value = cbTermin.SelectedValue;
        cmd.Parameters.AddWithValue("@ZaposleniID", SqlDbType.Int).Value = cbZaposleni.SelectedValue;
        cmd.Parameters.AddWithValue("@KlijentID", SqlDbType.Int).Value = cbKlijent.SelectedValue;

        if (azurira)
        {
            cmd.CommandText = @"update tblZakazivanjeTretmana set UslugaID=@UslugaID, TerminID=@TerminID, ZaposleniID=@ZaposleniID, KlijentID=@KlijentID
                           where ZakazivanjeID = @id";
            red = null;
        }
        else
        {
            cmd.CommandText = @"insert into tblZakazivanjeTretmana(UslugaID, TerminID, ZaposleniID, KlijentID)
                               values (@UslugaID, @TerminID, @ZaposleniID, @KlijentID)";
        }
        cmd.ExecuteNonQuery();
        cmd.Dispose();
        this.Close();
    }
    catch (SqlException ex)
    {
        MessageBox.Show(ex.Message, "Greska", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    catch (FormatException)
    {
        MessageBox.Show("Došlo je do greske prilikom konverzije podataka", "Greska", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
        {
            konekcija.Close();
        }
    }
}

```

Listing 6.19 - Sadržaj metode Sacuvaj_Click za unos novog zakazivanja tretmana

Nakon kreiranja formi za svaku od tabele, potrebno je nešto slično napraviti i za glavni prozor (eng. Main Window). Glavni prozor predstavlja prvi, inicijalni, kontakt korisnika sa aplikacijom i ujedno i osnovni prozor iz kojeg će se otvarati prethodno kreirane forme i u koji će se korisnik vraćati nakon zatvaranja tih formi. Na njemu bi trebalo da se nalazi dugme za svaku od osam kreiranih tabela u bazi, čijim će se klikom tastera miša, na sredini forme iščitavati ti podaci. Takođe, klikom tastera miša na dugme određene tabele, potrebno je da se korisniku obezbede dugmići za dodavanje, izmenu i brisanje prikazanog sadržaja.



Slika 6.10 - Vizuelni prikaz glavnog prozora

```

<Window x:Class="SalonLepte.MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:local="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    Height="450" Width="800">
    <Grid Background="#White">
        <StackPanel HorizontalAlignment="Left" Height="44" VerticalAlignment="Top" Width="378" Orientation="Horizontal" Margin="303,331,0,0">
            <Button x:Name="btnDodaj" Content="Dodaj" Width="120" Click="btnDodaj_Click" FontSize="16" FontFamily="Century Gothic" Height="40" Foreground="#White" BorderBrush="#White" FontStyle="Italic">
                <Button.Background>
                    <SolidColorBrush Color="#FF292728"/>
                </Button.Background>
            </Button>
            <Button x:Name="btnIzmeni" Content="Izmeni" Width="120" Click="btnIzmeni_Click" FontFamily="Century Gothic" FontSize="16" Height="40" Foreground="#White" BorderBrush="#White" FontStyle="Italic">
                <Button.Background>
                    <SolidColorBrush Color="#FF292728"/>
                </Button.Background>
            </Button>
            <Button x:Name="btnObrisi" Content="Obriši" Width="120" Click="btnObrisi_Click" FontFamily="Century Gothic" FontSize="16" Height="40" Foreground="#White" BorderBrush="#White" FontStyle="Italic">
                <Button.Background>
                    <SolidColorBrush Color="#FF292728"/>
                </Button.Background>
            </Button>
        </StackPanel>
        <Grid HorizontalAlignment="Left" Height="434" VerticalAlignment="Center" Width="184" Background="#FF292728">
            <StackPanel HorizontalAlignment="Center" Height="436" VerticalAlignment="Top" Width="298" Background="#FF292728" Margin="0,49,0,0">
                <Button x:Name="btnZakazivanja" Content="Zakazivanja" HorizontalAlignment="Center" Height="45" Width="110" Click="btnZakazivanja_Click" FontSize="18" FontFamily="Century Gothic" VerticalAlignment="Center" Background="#(x:Null)" Foreground="#White" BorderBrush="#(x:Null)"/>
                <Button x:Name="btnKlijenti" Content="Klijenti" HorizontalAlignment="Center" Height="45" Width="110" Click="btnKlijenti_Click" FontSize="18" FontFamily="Century Gothic" Foreground="#White" Background="#(x:Null)" BorderBrush="#(x:Null)"/>
                <Button x:Name="btnZaposleni" Content="Zaposleni" HorizontalAlignment="Center" Height="45" Width="110" Click="btnZaposleni_Click" FontSize="18" FontFamily="Century Gothic" FontStyle="Italic" BorderBrush="#(x:Null)" Background="#(x:Null)" Foreground="#White" VerticalAlignment="Center" Width="110" />
                <Button x:Name="btnUsluge" Content="Usluge" HorizontalAlignment="Center" Height="45" Width="110" Click="btnUsluge_Click" FontSize="18" FontFamily="Century Gothic" FontStyle="Italic" BorderBrush="#(x:Null)" Background="#(x:Null)" Foreground="#White" VerticalAlignment="Center" Width="110" />
                <Button x:Name="btnVrstiUsluga" Content="Vrsti usluga" HorizontalAlignment="Center" Height="45" Width="110" Click="btnVrstiUsluga_Click" FontSize="18" FontFamily="Century Gothic" FontStyle="Italic" BorderBrush="#(x:Null)" Background="#(x:Null)" Foreground="#White" VerticalAlignment="Center" Width="110" />
                <Button x:Name="btnProizvodi" Content="Proizvodi" HorizontalAlignment="Center" Height="45" Width="110" Click="btnProizvodi_Click" FontSize="18" FontFamily="Century Gothic" FontStyle="Italic" BorderBrush="#(x:Null)" Background="#(x:Null)" Foreground="#White" VerticalAlignment="Center" Width="110" />
                <Button x:Name="btnNabavke" Content="Nabavke" HorizontalAlignment="Center" Height="45" Width="110" Click="btnNabavke_Click" FontSize="18" FontFamily="Century Gothic" FontStyle="Italic" BorderBrush="#(x:Null)" Background="#(x:Null)" Foreground="#White" VerticalAlignment="Center" Width="110" />
                <Button x:Name="btnTermini" Content="Termini" HorizontalAlignment="Center" Height="45" Width="110" Click="btnTermini_Click" FontSize="18" FontFamily="Century Gothic" FontStyle="Italic" BorderBrush="#(x:Null)" Background="#(x:Null)" Foreground="#White" VerticalAlignment="Center" Width="110" />
            </StackPanel>
        </Grid>
        <ImageBrush ImageSource="/1909b88e75b08925fe4Ha0ee0ccc0cc0.jpg" Fill="White" HorizontalAlignment="Center" Margin="0,0,0,100" VerticalAlignment="Bottom" Width="616" Height="216" />
    </Grid>
    <DataGrid x:Name="dataGridCentralni" HorizontalAlignment="Left" VerticalAlignment="Top" Height="auto" Width="auto" IsReadOnly="True" AlternatingRowBackground="#D9E1F2" CanUserAddRows="False" SelectionUnit="FullRow" SelectionMode="Single" Margin="254,130,0,0" BorderBrush="Black" />

```

Listing 6.19 – XAML kod centralnog prozora i StackPanaela glavnog prozora

Kako bi elementi unutar ovog prozora bili uredno raspoređeni, koristiće se poseban panel za njihovo smeštanje - StackPanel. U okviru gornjeg StackPanel-a, biće smešteni dugmići za izlistavanje tabela na centralnom gridu. Potrebno je podesiti poravnjanja kao što je prikazano u listingu 6.19, a zatim izgenerisati metode koje će sprovoditi odgovarajuće akcije klikom tastera miša na to dugme. U nastavku će biti prikazana CodeBehind sadržina glavnog prozora, gde je smešten najveći deo poslovne logike. U sledećem listingu prikazani su korišteni imenski prostori.

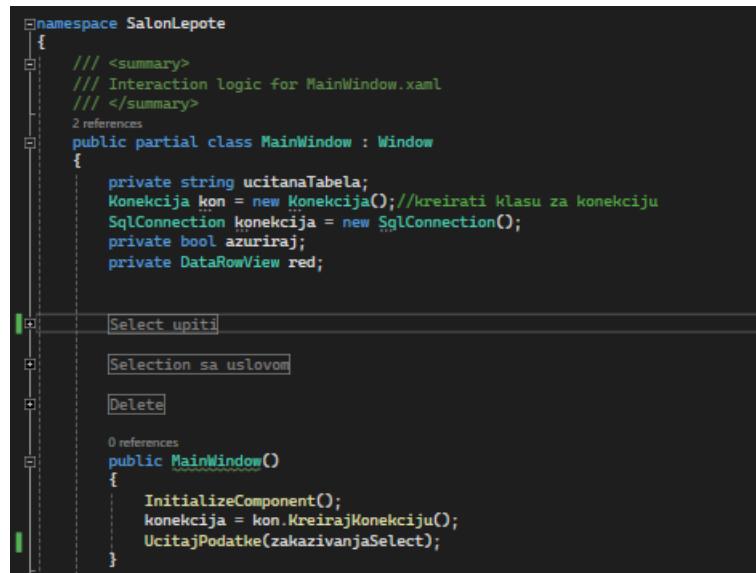
```

using System.Data.SqlClient;
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;
using System.Data.SqlClient;
using System.Data;
using SalonLepte.Forme;

```

Listing 6.20 – Imenski prostori za glavni prozor

Za početak, sadržina glavnog prozora trebalo bi da izgleda kao u listingu 6.21. Sadržaće instancu konekcije ka bazi podataka, metodu za popunjavanje početnog grida pri pokretanju prozora, konstruktor za kreiranje novog prozora, kao i izgenerisane metode iz XAML koda, putem kojih će se, klikom tastera miša na odgovarajuće dugme, obavljati određene funkcionalnosti.



Listing 6.21 – Sadržaj CodeBehind-a glavnog prozora

```

#region Select upiti
private static string klijentiSelect = @"select KlijentID as ID, ImeKlijenta as Ime, PrezimeKlijenta as Prezime,
                                         KontaktKlijenta as Kontakt from tblKlijent";
private static string zaposleniSelect = @"select ZaposleniID as ID, ImeZaposlenog as Ime, PrezimeZaposlenog as Prezime,
                                         KontaktZaposlenog as Kontakt from tblZaposleni";
private static string proizvodiSelect = @"select ProizvodID as ID, NazivProizvoda as Naziv,
                                         CenaProizvoda as Cena from tblProizvod";
private static string uslugeSelect = @"select UslugaID as ID, Naziv as Naziv, Opis as Vrsta
                                         from tblUsluga join tblVrstauSluge on tblUsluga.VrstaID = tblVrstauSluge.VrstaID";
private static string vrsteUslugaSelect = @"select VrstaID as ID, Opis, Trajanje, Cena from tblVrstauSluge";
private static string terminiSelect = @"select TerminID as ID, Datum, Vreme from tblTermin";
private static string nabavkeSelect = @"select NabavkaID as ID, Datum, Kolicina, NazivProizvoda as Proizvod,
                                         ImeZaposlenog + ' ' + PrezimeZaposlenog as Zaposleni
                                         from tblNabavka join tblProizvod on tblNabavka.ProizvodID = tblProizvod.ProizvodID
                                         join tblZaposleni on tblNabavka.ZaposleniID = tblZaposleni.ZaposleniID";
private static string zakazivanjaSelect = @"select ZakazivanjeID as ID, Naziv as Usluga, Datum, Vreme as Termin,
                                         ImeZaposlenog + ' ' + PrezimeZaposlenog as Zaposleni, ImeKlijenta + ' ' + PrezimeKlijenta as Klijent
                                         from tblZakazivanjeTretmana join tblUsluga on tblZakazivanjeTretmana.UslugaID = tblUsluga.UslugaID
                                         join tblTermin on tblZakazivanjeTretmana.TerminID = tblTermin.TerminID
                                         join tblZaposleni on tblZakazivanjeTretmana.ZaposleniID = tblZaposleni.ZaposleniID
                                         join tblKlijent on tblZakazivanjeTretmana.KlijentID = tblKlijent.KlijentID";

```

Listing 6.22 – SQL kod Select upita

```

#region Selection sa uslovom
private static string selectUslovKlijenti = @"select * from tblKlijent where KlijentID=";
private static string selectUslovZaposleni = @"select * from tblZaposleni where ZaposleniID=";
private static string selectUslovProizvodi = @"select * from tblProizvod where ProizvodID=";
private static string selectUslovNabavke = @"select * from tblNabavka where NabavkaID=";
private static string selectUslovUsluge = @"select * from tblUsluga where UslugaID=";
private static string selectUslovVrsteUsluga = @"select * from tblVrstaUsluga where VrstaID=";
private static string selectUslovTermini = @"select * from tblTermin where TerminID=";
private static string selectUslovZakazivanja = @"select * from tblZakazivanjeTretmana where ZakazivanjeID=";
#endregion

```

Listing 6.23 – SQL kod Select upita sa uslovom

```

#region Delete
private static string klijentiDelete = @"delete from tblKlijent where KlijentID=";
private static string zaposleniDelete = @"delete from tblZaposleni where ZaposleniID=";
private static string proizvodiDelete = @"delete from tblProizvod where ProizvodID=";
private static string nabavkeDelete = @"delete from tblNabavka where NabavkaID=";
private static string uslugeDelete = @"delete from tblUsluga where UslugaID=";
private static string vrsteUslugaDelete = @"delete from tblVrstaUsluga where VrstaID=";
private static string terminiDelete = @"delete from tblTermin where TerminID=";
private static string zakazivanjaDelete = @"delete from tblZakazivanjeTretmana where ZakazivanjeID=";
#endregion

```

Listing 6.24 – SQL kod Delete upita

U prethodna tri listinga je prikazana komunikacija sa bazom podataka.

Nakon instanciranja konekcije, sledeća stvar je kreiranje metode koja će popunjavati početni DataGrid pri kreiranju novog prozora. U ovom primeru, tabela tblZakazivanjeTretmana će biti podrazumeumevana tabela, odnosno tabela čiji će se prikaz generisati pri pokretanju ovog prozora. U ovom slučaju, nakon otvaranja konekcije, prikazani upit se kreira na osnovu povezivanja pet tabela iz baze podataka. Zatim, na isti način kao kod popunjavanja ComboBoxova pri instanciranju prozora neke od formi, ovi podaci se smeštaju u odgovarajući DataSet kojeg zatim popunjava SqlDataAdapter. U konstruktoru se poziva ova metoda i prosleđuje joj se select upit za tabelu tblZakazivanjeTretmana.

```

33 references
private void UcitajPodatke(string selectUpit)
{
    try
    {
        konekcija.Open();
        SqlDataAdapter dataAdapter = new SqlDataAdapter(selectUpit, konekcija);
        DataTable dataTable = new DataTable();
        dataAdapter.Fill(dataTable);
        if (dataGridCentralni != null)
        {
            dataGridCentralni.ItemsSource = dataTable.DefaultView;
        }
        ucitanaTabela = selectUpit;
        dataAdapter.Dispose();
        dataTable.Dispose();
    }
    catch (SqlException ex)
    {
        MessageBox.Show(ex.Message, "Greska", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
        {
            konekcija.Close();
        }
    }
}

```

Listing 6.25 – Sadržaj metode UcitajPodatke

Sledi pregled svake od posebnih metoda, koje se pozivaju nakon klika tasterom miša na odgovarajuće dugme iz gornjeg StackPanel-a, koje služe za ispis podataka iz tabele na centralnom gridu. Na osnovu prethodno kreirane metode UcitajPodatke i select upita, za sve druge metode, koje predstavljaju odgovor na klik korisnika, poziva se upravo ova metoda i prosleđuje joj se odgovarajući upit (listing 6.26).

```

1 reference
private void btnZakazivanja_Click(object sender, RoutedEventArgs e)
{
    UcitajPodatke(zakazivanjaSelect);
}

1 reference
private void btnKlijenti_Click(object sender, RoutedEventArgs e)
{
    UcitajPodatke(klijentiSelect);
}

1 reference
private void btnZaposleni_Click(object sender, RoutedEventArgs e)
{
    UcitajPodatke(zaposleniSelect);
}

1 reference
private void btnUsluge_Click(object sender, RoutedEventArgs e)
{
    UcitajPodatke(uslugeSelect);
}

1 reference
private void btnVrsteUsluga_Click(object sender, RoutedEventArgs e)
{
    UcitajPodatke(vrsteUslugaSelect);
}

1 reference
private void btnProizvodi_Click(object sender, RoutedEventArgs e)
{
    UcitajPodatke(proizvodiSelect);
}

1 reference
private void btnNabavke_Click(object sender, RoutedEventArgs e)
{
    UcitajPodatke(nabavkeSelect);
}

1 reference
private void btnTermini_Click(object sender, RoutedEventArgs e)
{
    UcitajPodatke(terminiSelect);
}

```

Listing 6.26 – Prikaz svih metoda koje učitavaju podatke

Sadržaj metoda generisanih na klik tasterom miša na dugme Dodaj je poprilično jednostavan. Klikom tastera miša na dugme Dodaj, instancira se prozor odgovarajuće forme i prikazuje korisniku na ekranu. Nakon zatvaranja tog prozora forme, osvežava se centralni grid sa aktuelnim podacima nakon eventualnog dodavanja novog zapisa u bazu. U listingu 6.26 prikazan je sadržaj metode za dodavanje nove torke u odgovarajuću tabelu u bazi podataka.

```

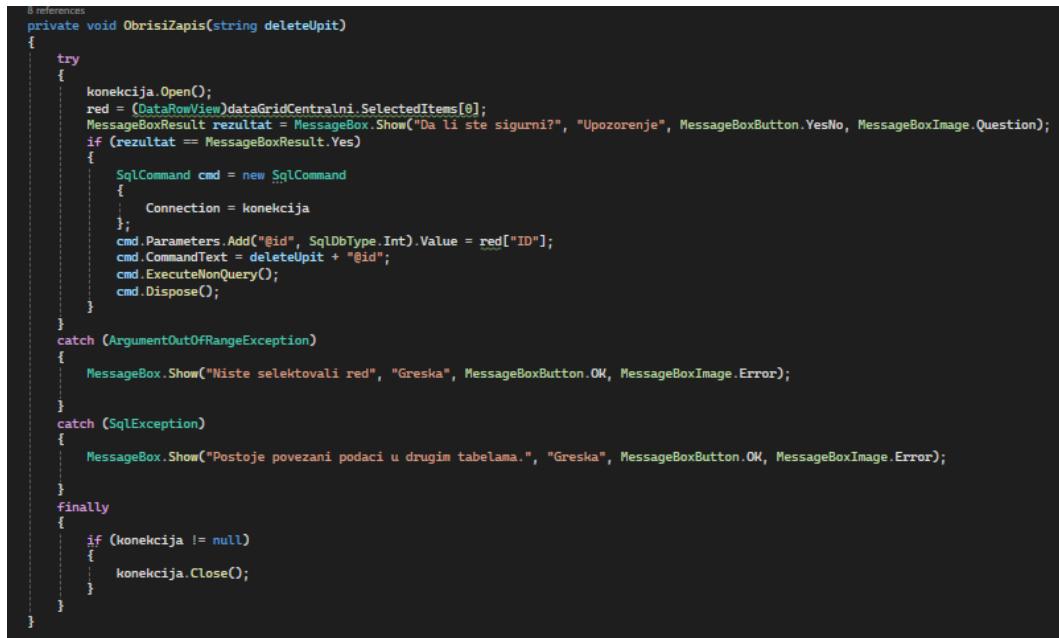
1 reference
private void btnDodaj_Click(object sender, RoutedEventArgs e)
{
    Window prozor;
    if (ucitanaTabela.Equals(klijentiSelect))
    {
        prozor = new FrmKlijent();
        prozor.ShowDialog();
        UcitajPodatke(klijentiSelect);
    }
    else if (ucitanaTabela.Equals(zaposleniSelect))
    {
        prozor = new FrmZaposleni();
        prozor.ShowDialog();
        UcitajPodatke(zaposleniSelect);
    }
    else if (ucitanaTabela.Equals(proizvodiSelect))
    {
        prozor = new FrmProizvod();
        prozor.ShowDialog();
        UcitajPodatke(proizvodiSelect);
    }
    else if (ucitanaTabela.Equals(uslugeSelect))
    {
        prozor = new FrmUsluga();
        prozor.ShowDialog();
        UcitajPodatke(uslugeSelect);
    }
    else if (ucitanaTabela.Equals(vrstеUslugaSelect))
    {
        prozor = new FrmVrstaUsluge();
        prozor.ShowDialog();
        UcitajPodatke(vrstеUslugaSelect);
    }
    else if (ucitanaTabela.Equals(terminiSelect))
    {
        prozor = new FrmTermin();
        prozor.ShowDialog();
        UcitajPodatke(terminiSelect);
    }
    else if (ucitanaTabela.Equals(nabavkeSelect))
    {
        prozor = new FrmNabavka();
        prozor.ShowDialog();
        UcitajPodatke(nabavkeSelect);
    }
    else
    {
        prozor = new FrmZakazivanjeTretmana();
        prozor.ShowDialog();
        UcitajPodatke(zakazivanjaSelect);
    }
}
8 references

```

Listing 6.27 – Sadržaj metode btnDodaj_Click

Brisanje podataka iz baze podataka obavlja se putem klika tasterom miša na dugme Obriši na način prikazan u listingu 6.28. Nakon otvaranja konekcije, kreira se objekat klase DataRowView koji u sebi čuva informaciju o indeksu reda koji je korisnik prethodno označio. Nakon toga kreira se Delete SQL komanda putem koje će se u samoj bazi obrisati željena stavka. Pre samog izvršavanja

komande, pred korisnikom se pojavljuje dijalog prozor (MessageBox) putem kojeg on potvrđuje ili odustaje od brisanja. Dva osnovna izuzetka koja se ovde mogu javiti su: ArgumentOutOfRangeException i SQLException. Prvi se javlja pri pokušaju brisanja ukoliko nije označen nijedan red, a pritisnuto je dugme za brisanje, a drugi ukoliko se pokuša brisanje entiteta iz tabele koji je primarnim ključem povezan sa nekom drugom tabelom.



```
8 references
private void ObrisizZapis(string deleteUpit)
{
    try
    {
        konekcija.Open();
        red = (DataRowView)dataGridCentralni.SelectedItems[0];
        MessageBoxResult rezultat = MessageBox.Show("Da li ste sigurni?", "Upozorenje", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (rezultat == MessageBoxResult.Yes)
        {
            SqlCommand cmd = new SqlCommand
            {
                Connection = konekcija
            };
            cmd.Parameters.Add("@id", SqlDbType.Int).Value = red["ID"];
            cmd.CommandText = deleteUpit + "@id";
            cmd.ExecuteNonQuery();
            cmd.Dispose();
        }
    }
    catch (ArgumentOutOfRangeException)
    {
        MessageBox.Show("Niste selektovali red", "Greska", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    catch (SQLException)
    {
        MessageBox.Show("Postoje povezani podaci u drugim tabelama.", "Greska", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    finally
    {
        if (konekcija != null)
        {
            konekcija.Close();
        }
    }
}
```

Listing 6.28 – Sadržaj metode ObrisizZapis

Kada je tasterom miša kliknuto na dugme Obriši, u zavisnosti od toga koja tabela je trenutno učitana i koju stavku tabele je korisnik izabrao, metodi ObrisizZapis se prosleđuju parametri i nakon toga se ponovo učitavaju podaci iz baze podataka, kako bi oni bili ažurni nakon brisanja označenog reda (listing 6.29).

```

    T reference
    private void btnObrisi_Click(object sender, RoutedEventArgs e)
    {
        if (ucitanaTabela.Equals(klijentiSelect))
        {
            ObrisizZapis(klijentiDelete);
            UcitajPodatke(klijentiSelect);
        }
        else if (ucitanaTabela.Equals(nabavkeSelect))
        {
            ObrisizZapis(nabavkeDelete);
            UcitajPodatke(nabavkeSelect);
        }
        else if (ucitanaTabela.Equals(zaposleniSelect))
        {
            ObrisizZapis(zaposleniDelete);
            UcitajPodatke(zaposleniSelect);
        }
        else if (ucitanaTabela.Equals(uslugeSelect))
        {
            ObrisizZapis(uslugeDelete);
            UcitajPodatke(uslugeSelect);
        }
        else if (ucitanaTabela.Equals(proizvodiSelect))
        {
            ObrisizZapis(proizvodiDelete);
            UcitajPodatke(proizvodiSelect);
        }
        else if (ucitanaTabela.Equals(vrsteUslugaSelect))
        {
            ObrisizZapis(vrsteUslugaDelete);
            UcitajPodatke(vrsteUslugaSelect);
        }
        else if (ucitanaTabela.Equals(terminiSelect))
        {
            ObrisizZapis(terminiDelete);
            UcitajPodatke(terminiSelect);
        }
        else
        {
            ObrisizZapis(zakazivanjaDelete);
            UcitajPodatke(zakazivanjaSelect);
        }
    }
}

```

Listing 6.29 – Sadržaj metode btnObrisi_Click

Za modifikaciju podataka, kao i za dodavanje, potrebno je da postoji odgovarajuća forma u kojoj će korisnik menjati, odnosno unositi željene vrednosti. U ovom primeru, za svaku od tabele već je kreirana forma putem koje se novi zapisi upisuju u bazu podataka. Kako bi se izbeglo kreiranje novih formi specijalno za modifikaciju podataka, može se iskoristiti postojeća forma za dodavanje, s tim što će jedina razlika biti u tome koju akciju izvršava dugme Sačuvaj te forme kada korisnik tasterom miša klikne na njega. Na samom početku potrebno je, u okviru glavnog prozora deklarisati tri promenljive. Prva je tipa podatka boolean i koristiće se kao uslov koji će dugmetu Sačuvaj govoriti da li je potrebno podatke upisati u bazu podataka kao novi entitet ili izmeniti. Druga je tipa DataRowView i služi za pamćenje vrednosti označenog reda, kada iz glavnog prozora pređemo u neku drugu formu. Treća promenljiva, tipa podatka string, koja govori koja je trenutno tabela učitana. U naredna dva listinga prikazana je metoda koja se koristi za izmenu. Na samom početku, deklarisana promenljiva se postavlja na true. Nakon toga instancira se prozor njene forme i otvara konekciju. Pomoću DataRowView objekta, kao i kod brisanja, pamti se koji red je označen. Razlika je u tome što se sada, u drugu promenljivu (označen) dodaje ta vrednost kako bi se iskoristila u drugoj formi. Nakon toga se, pomoću select upita kojem se u where uslovu

zadaje vrednost primarnog ključa zapisa koji se modifikuje, potrebni podaci smeštaju u odgovarajuća polja forme, pomoću SqlDataReader-a. Nakon toga, poziva se metoda prozora ShowDialog i on se prikazuje na ekranu sa popunjениm vrednostima traženog zapisa. U okviru ove metode, u finally bloku, konekcija se zatvara, a promenljiva ažuriraj, vraća na false, u slučaju da, nakon izmene, korisnik želi da doda novi zapis u bazu podataka.

```
private void PopuniFormu(string selectUslov)
{
    try
    {
        konekcija.Open();
        ažuriraj = true;
        red = (DataTable)view.GridViewCentralni.SelectedItems[0];
        SqlCommand cmd = new SqlCommand
        {
            Connection = konekcija
        };
        cmd.Parameters.Add("@id", SqlDbType.Int).Value = red["ID"];
        cmd.CommandText = selectUslov + " @id";
        SqlDataReader citac = cmd.ExecuteReader();
        if (citac.Read())
        {
            if (ucitanaTabela.Equals(klijentiSelect))
            {
                FrmKlijent prozorKlijent = new FrmKlijent(azuriraj, red);
                prozorKlijent.txtIme.Text = citac["imeKlijenta"].ToString();
                prozorKlijent.txtPrezime.Text = citac["PrezimeKlijenta"].ToString();
                prozorKlijent.txtKontakt.Text = citac["KontaktKlijenta"].ToString();
                prozorKlijent.ShowDialog();
            }
            else if (ucitanaTabela.Equals(nabavkeSelect))
            {
                FrmNabavka prozorNabavka = new FrmNabavka(azuriraj, red);
                prozorNabavka.dateTimePicker.SelectedDate = DateTime.Now;
                prozorNabavka.txtNadziv.Text = citac["NadzivNabavke"].ToString();
                prozorNabavka.cbZaposleni.SelectedValue = citac["ZaposlenID"].ToString();
                prozorNabavka.cbProizvod.SelectedValue = citac["ProizvodID"].ToString();
                prozorNabavka.ShowDialog();
            }
            else if (ucitanaTabela.Equals(proizvodiSelect))
            {
                FrmProizvod prozorProizvod = new FrmProizvod(azuriraj, red);
                prozorProizvod.txtNaziv.Text = citac["NazivProizvoda"].ToString();
                prozorProizvod.txtCena.Text = citac["CenaProizvoda"].ToString();
                prozorProizvod.ShowDialog();
            }
            else if (ucitanaTabela.Equals(zaposleniSelect))
            {
                FrmZaposleni prozorZaposleni = new FrmZaposleni(azuriraj, red);
                prozorZaposleni.txtImeZaposlenog.Text = citac["ImeZaposlenog"].ToString();
                prozorZaposleni.txtPrezimeZaposlenog.Text = citac["PrezimeZaposlenog"].ToString();
                prozorZaposleni.txtKontaktZaposlenog.Text = citac["KontaktZaposlenog"].ToString();
                prozorZaposleni.ShowDialog();
            }
            else if (ucitanaTabela.Equals(uslugeSelect))
            {
                FrmUsluga prozorUsluga = new FrmUsluga(azuriraj, red);
                prozorUsluga.txtNaziv.Text = citac["Naziv"].ToString();
                prozorUsluga.cbVrsta.Text = citac["VrstaID"].ToString();
                prozorUsluga.ShowDialog();
            }
        }
    }
}
```

Listing 6.30.1 – Sadržaj metode PopuniFormu

```

        }
        else if (ucitanaTabela.Equals(vrsteUslugaSelect))
        {
            FrmVrsteUsluga prozorVrsta = new FrmVrsteUsluga(azuriraj, red);
            prozorVrsta.txtOpis.Text = citac["Opis"].ToString();
            prozorVrsta.txtTrajanje.Text = citac["Trajanje"].ToString();
            prozorVrsta.txtCena.Text = citac["Cena"].ToString();
            prozorVrsta.ShowDialog();
        }
        else if (ucitanaTabela.Equals(zakazivanjaSelect))
        {
            FrmZakazivanjeTrtmana prozorZakazivanja = new FrmZakazivanjeTrtmana(azuriraj, red);
            prozorZakazivanja.dateTimePicker.SelectedDate = DateTime.Now;
            prozorZakazivanja.cbTermin.SelectedValue = citac["TerminID"].ToString();
            prozorZakazivanja.cbZaposleni.SelectedValue = citac["ZaposlenID"].ToString();
            prozorZakazivanja.cbKlijent.SelectedValue = citac["KlijentID"].ToString();
            prozorZakazivanja.ShowDialog();
        }
        else
        {
            FrmTermin prozorTermin = new FrmTermin(azuriraj, red);
            prozorTermin.dateTimePicker.Text = citac["Datum"].ToString();
            prozorTermin.txtVreme.Text = citac["Vreme"].ToString();
            prozorTermin.ShowDialog();
        }
    }
}
catch (ArgumentOutOfRangeException)
{
    MessageBox.Show("Niste selektovali red!", "Greška", MessageBoxButtons.OK, MessageBoxIcon.Error);
}
finally
{
    if (konekcija != null)
    {
        konekcija.Close();
    }
    ažuriraj = false;
}
}
```

Listing 6.30.2 – Sadržaj metode PopuniFormu

Kada Korisnik izabere stavku koju želi da izmeni, poziva se metoda BtnIzmeni_Click koja u zavisnosti od učitane tabele prosleđuje odgovarajući parameter metodi PopuniFormu i nakon izvršene ismene ponovo učitava podatke kako bi podaci bili ažurni.

```

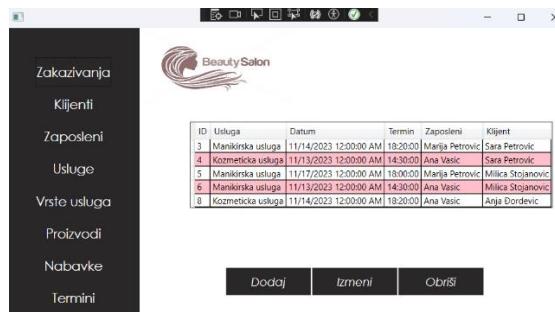
1 reference
private void btnIzmeni_Click(object sender, RoutedEventArgs e)
{
    if (ucitanaTabela.Equals(klijentiSelect))
    {
        PopuniFormu(selectUlovKlijenti);
        UcitajPodatke(klijentiSelect);
    }
    else if (ucitanaTabela.Equals(nabavkeSelect))
    {
        PopuniFormu(selectUlovNabavke);
        UcitajPodatke(nabavkeSelect);
    }
    else if (ucitanaTabela.Equals(zaposleniSelect))
    {
        PopuniFormu(selectUlovZaposleni);
        UcitajPodatke(zaposleniSelect);
    }
    else if (ucitanaTabela.Equals(uslugeSelect))
    {
        PopuniFormu(selectUlovUsluge);
        UcitajPodatke(uslugeSelect);
    }
    else if (ucitanaTabela.Equals(proizvodiSelect))
    {
        PopuniFormu(selectUlovProizvodi);
        UcitajPodatke(proizvodiSelect);
    }
    else if (ucitanaTabela.Equals(vrsteUslugaSelect))
    {
        PopuniFormu(selectUlovVrsteUsluga);
        UcitajPodatke(vrsteUslugaSelect);
    }
    else if (ucitanaTabela.Equals(termeniSelect))
    {
        PopuniFormu(selectUlovTermeni);
        UcitajPodatke(termeniSelect);
    }
    else
    {
        PopuniFormu(selectUlovZakazivanja);
        UcitajPodatke(zakazivanjaSelect);
    }
}

```

Listing 6.31 – Sadržaj metode btnIzmeni_Click

7. Testiranje aplikacije

Kada je aplikacija kreirana i kada su implementirane sve neophodne funkcionalnosti koje ona treba da podrži, neophodno je testirati aplikaciju kako bi se proverila ispravnost tih funkcionalnosti. U ovom poglavljju dat je prikaz testiranja WPF aplikacije za podršku poslovanja agencije za registraciju vozila. Pri pokretanju aplikacije, generiše se početni prozor kao na slici 7.1.



Slika 7.1 – Početni prozor pri pokretanju aplikacije i prikaz podataka iz tabele Zakazivanje Tretmana

Klikom tastera miša na dugmiće iz centralnog menija vrši se prikaz podataka željene tabele na centralnom gridu.

ID	Ime	Prezime	Kontakt
2	Sara	Petrovic	+381601112233
4	Milica	Stojanovic	+381631122334
5	Anja	Dordevic	+381641122534
6	Tijana	Maric	+381606026551
7	Mila	Bikar	+381606026553
9	Jovana	Zecevic	+38163155364

Dodaj | **Izmeni** | **Obrisi**

Slika 7.2 – Prikaz podataka iz tabele klijent

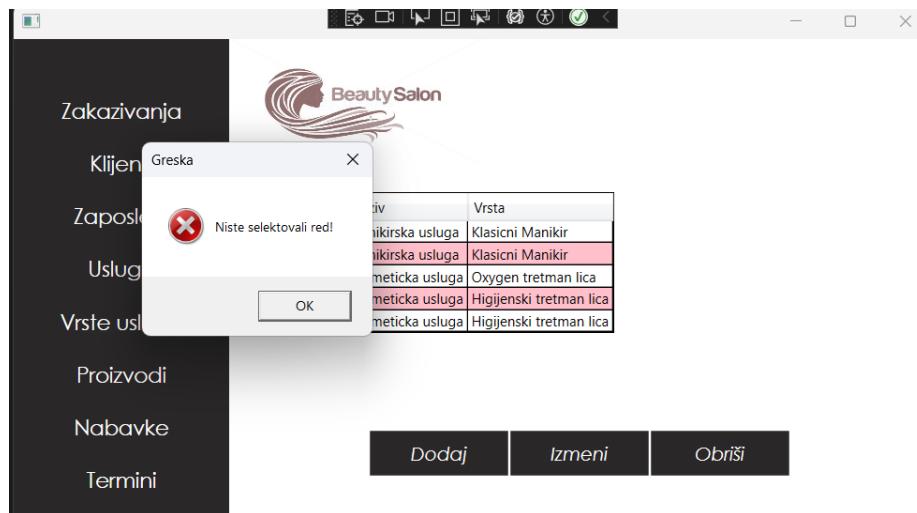
Klikom tastera miša na dugme Dodaj, otvara se forma za dodavanje novog zapisa u bazu podataka. U formi se klikom na taster Sacuvaj, sačuvavaju odnosno upisuju podaci u bazu i tabela se, nakon zatvaranja forme, automatski ažurira.

ID	Ime	Prezime	Kontakt
2	Sara	Petrovic	+381601112233
4	Milica	Stojanovic	+381631122334
5	Anja	Dordevic	+381641122534
6	Tijana	Maric	+381606026551
7	Mila	Bikar	+381606026553
9	Jovana	Zecevic	+38163155364

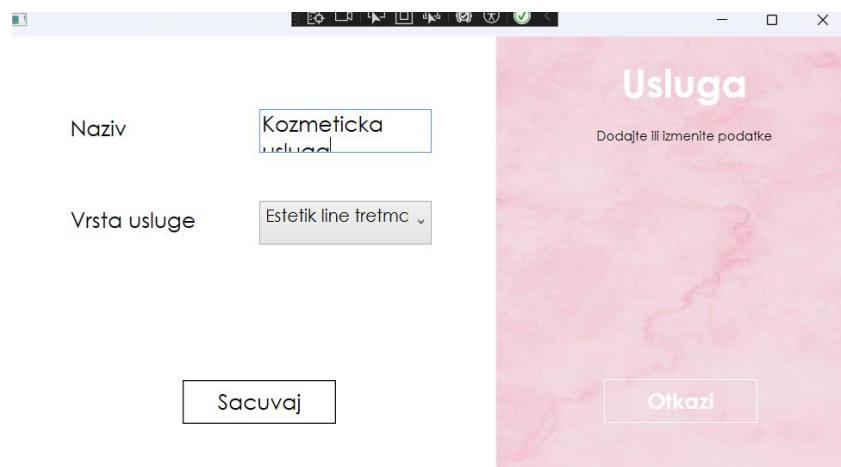
Dodaj | **Izmeni** | **Obrisi**

Slika 7.3 – Dodavanje zapisa u bazu podataka

Označavanjem odgovarajućeg reda otvara se forma sa popunjениm vrednostima prethodnog zapisa koje korisnik može modifikovati. Ukoliko Korisnik klikne na dugme Izmeni, a pritom nije izabrao red, onda će mu program ukazati na tu grešku.

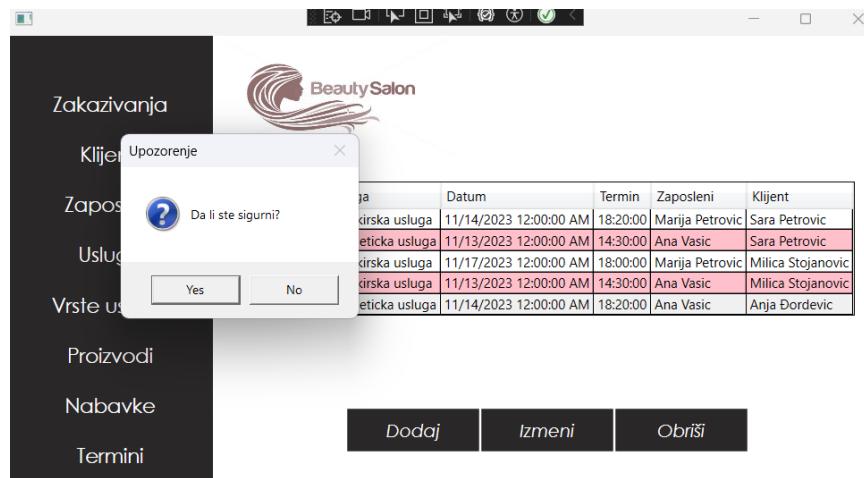


Slika 7.4 – Prikaz greške prilikom neselektovanja reda



Slika 7.5 – Prikaz forme sa prethodno učitanim podacima

Brisanje podataka iz tabele se vrši tako što se prvo selektuje red, pa potom klikne na dugme Obrisi. Nakon toga će program pitati korisnika da li je siguran da želi da izbriše podatak, i onda će završiti radnju u zavisnosti od odgovora korisnika.



Slika 7.6 – Prikaz pokušaja brisanja forme

8. Zaključak

Kroz proces izrade ovog WPF projekta, uspešno sam primenila koncepte WPF tehnologije, poboljšala svoje veštine u dizajniranju korisničkog interfejsa i implementirala funkcionalnosti koje su odgovarale zahtevima projekta. Osim što sam stekla bolje razumevanje Windows Presentation Foundation tehnologije i njenih principa, takođe proširila sam svoje znanje o principima korisničkog iskustva i važnosti prilagođavanja aplikacije potrebama korisnika. Tokom rada na projektu, intenzivno sam unapredila svoje programerske veštine u C# jeziku, istovremeno bolje razumevajući arhitekturu i strukturu WPF aplikacija.