



Univerzitet u Novom Sadu

Fakultet tehničkih nauka

Inženjerstvo informacionih sistema

Uputstvo za razvoj WPF aplikacije

Novi Sad, 2017. god.

Sadržaj

• Uvod	-3
• Zadatak	-3
• Kreiranje UML dijagrama.....	- 4
• Dijagram slučajeva upotrebe.....	- 4
• Dijagram klasa	- 8
• Kreiranje baze podataka	-9
• Kreiranje Studentske WPF aplikacije (desktop aplikacije)	-21

1. Uvod

U okviru ovog uputstva opisani su svi potrebni koraci koje neko treba da prođe kako bi razvio **WPF** (engl. *Windows Presentation Foundation*) **aplikaciju** i upoznao se sa osnovnim principima **UML modelovanja**, kao i rada u **Microsoft SQL Server-u** i **Microsoft Visual Studiju**.

2. Zadatak

Cilj zadatka je da se kreira **WPF aplikacija** koja simulira rad **trafike**. Njena komunikacija sa **bazom podataka** ostvarivaće se putem ADO.NET tehnologije. **Aplikacija** treba da pruži mogućnost prikazivanja podataka o obavljenim kupovinama u **trafici**, kupcima koji su te kupovine obavljali, prodavcima koji su ih opsluživali, proizvodima koji su bili predmet tih kupovina (njihovim proizvođačima, vrstama i dobavljačima) kao i o fiskalnim računima koji se izrađuju nakon uspešno obavljene kupovine. Takođe, treba da pruža mogućnost *dodavanja, izmene i brisanja* podataka o navedenim entitetima.

3. Kreiranje UML dijagrama

3.1. Dijagram slučajeve upotrebe

Na samom početku pristupamo inicijalnom modelovanju sistema odnosno kreiranju **dijagrama slučajeve upotrebe** na osnovu datog opisa za primer **trafike**.

Slučaj upotrebe: Logovanje prilikom preuzimanja smene

Kratak opis: Prodavac se prilikom preuzimanja smene prijavljuje na sistem

Učesnici: Prodavac

Uslovi koji moraju biti zadovoljeni pre izvršavanja: Prodavac je započeo svoju smenu.

Opis: Prodavac se prilikom preuzimanja smene prijavljuje na sistem unosom svog username- a i password- a. Ukoliko se prvi put prijavljuje na sistem ili je eventualno zaboravio svoju lozinku iskoristiće pomoć koju sistem pruža u takvim situacijama. [*Izuzetak:* Prodavac je pogrešno uneo svoj username ili password].

Izuzeci:

[**Prodavac je pogrešno uneo svoj username ili password**]. Ukoliko je prodavac pogrešno uneo svoj username ili password zatražiće se od njega da ponovo unese podatke.

Uslovi koji moraju biti zadovoljeni nakon izvršavanja: Prodavac se uspešno ulogovao na sistem.

Slučaj upotrebe: Pregled stanja zaliha

Kratak opis: Pregled stanja zaliha robe u trafici

Učesnici: Prodavac

Uslovi koji moraju biti zadovoljeni pre izvršavanja: Prodavac je uspešno započeo prvu smenu.

Opis: Prodavac koji radi u prvoj smeni pregleda trenutno stanje zaliha u trafici i u koliko postoji potreba, naručuje potrebnu robu u potrebnim količinama. [**Izuzetak:** Ne postoji potreba za naručivanjem robe].

Izuzeci:

[**Ne postoji potreba za naručivanjem robe**]. Ukoliko trenutno ne postoji potreba za naručivanjem robe, porudžbina se neće obaviti.

Uslovi koji moraju biti zadovoljeni nakon izvršavanja: Utvrđeno je stanje zaliha i porudžbina je uspešno poslata odgovarajućem dobavljaču.

Slučaj upotrebe: Kupovina proizvoda

Kratak opis: Kupovina proizvoda na trafici od strane kupca

Učesnici: Prodavac, Kupac

Uslovi koji moraju biti zadovoljeni pre izvršavanja: U trafici se nalazi se traženi proizvod i kupac ima dovoljno novca za njegovu kupovinu.

Opis: Kupac potražuje određeni proizvod. Ukoliko se dati proizvod nalazi u ponudi i ukoliko kupac ima dovoljno novca obavlja se kupovina. U slučaju da kupac vrši elektronsku dopunu od njega će se tražiti broj mobilnog telefona. [**Izuzetak:** Traženi proizvod se ne nalazi u ponudi ili kupac nema dovoljno novca]. [**Izuzetak:** Prilikom elektronske dopune prodavac je pogrešno uneo broj mobilnog telefona].

Izuzeci:

[**Traženi proizvod se ne nalazi u ponudi ili kupac nema dovoljno novca**]. Ukoliko se traženi proizvod ne nalazi u ponudi ili kupac nema dovoljno novca kupovina se neće obaviti.

[**Prilikom elektronske dopune prodavac je pogrešno uneo broj mobilnog telefona**]. Ukoliko je prodavac pogrešno uneo broj mobilnog telefona kupca dopuna neće biti ispravna.

Uslovi koji moraju biti zadovoljeni nakon izvršavanja: Kupovina je uspešno obavljena, odnosno kupac je dobio traženi proizvod, ili je uspešno dopunio kredit.

Slučaj upotrebe: Izdavanje fiskalnog računa

Kratak opis: Izdavanje fiskalnog računa nakon kupovine

Učesnici: Prodavac, Kupac

Uslovi koji moraju biti zadovoljeni pre izvršavanja: Kupovina je uspešno obavljena i izvršeno je plaćanje.

Opis: Ukoliko je uspešno izvršena kupovina, kupcu se nakon izvršenog plaćanja izdaje fiskalni račun od strane prodavca. [**Izuzetak:** Neuspešno izvršena kupovina].

Izuzeci:

[**Neuspešno izvršena kupovina**]. Ukoliko kupovina nije izvršena fiskalni račun neće biti izdat.

Uslovi koji moraju biti zadovoljeni nakon izvršavanja: Kupcu je izdat fiskalni račun za datu kupovinu.

Slučaj upotrebe: Evidencija dnevne zarade

Kratak opis: Evidencija dnevne zarade na kraju radnog dana

Učesnici: Prodavac

Uslovi koji moraju biti zadovoljeni pre izvršavanja: Kraj je radnog dana i prodavac je za svaku kupovinu uredno izdao fiskalni račun.

Opis: Prodavac koji radi u drugoj smeni na kraju radnog dana računa i vrši evidenciju zarade za taj dan i podatke beleži u sistem. [**Izuzetak:** Prodavac je pogrešno izračunao ili uneo podatke]. [**Izuzetak:** Prodavac za neku kupovinu nije izdao fiskalni račun].

Izuzeci:

[**Pogrešno izračunati ili uneti podaci**]. Ukoliko je prodavac pogrešno izračunao ili uneo neke podatke evidencija o prihodima će biti neispravna.

[**Prodavac za neku kupovinu nije izdao fiskalni račun**]. Ukoliko prodavac za neku kupovinu nije izdao fiskalni račun evidencija o prihodima će takođe biti neispravna.

Uslovi koji moraju biti zadovoljeni nakon izvršavanja: Uspešno je evidentirana dnevna zarada i podaci su uneti u sistem.

Slučaj upotrebe: Isporuka i prijem dnevne štampe

Kratak opis: Isporuka i prijem dnevne štampe za potrebe trafike
Učesnici: Prodavac, Dobavljač

Uslovi koji moraju biti zadovoljeni pre izvršavanja: Dobavljač je isporučio naručene proizvode.

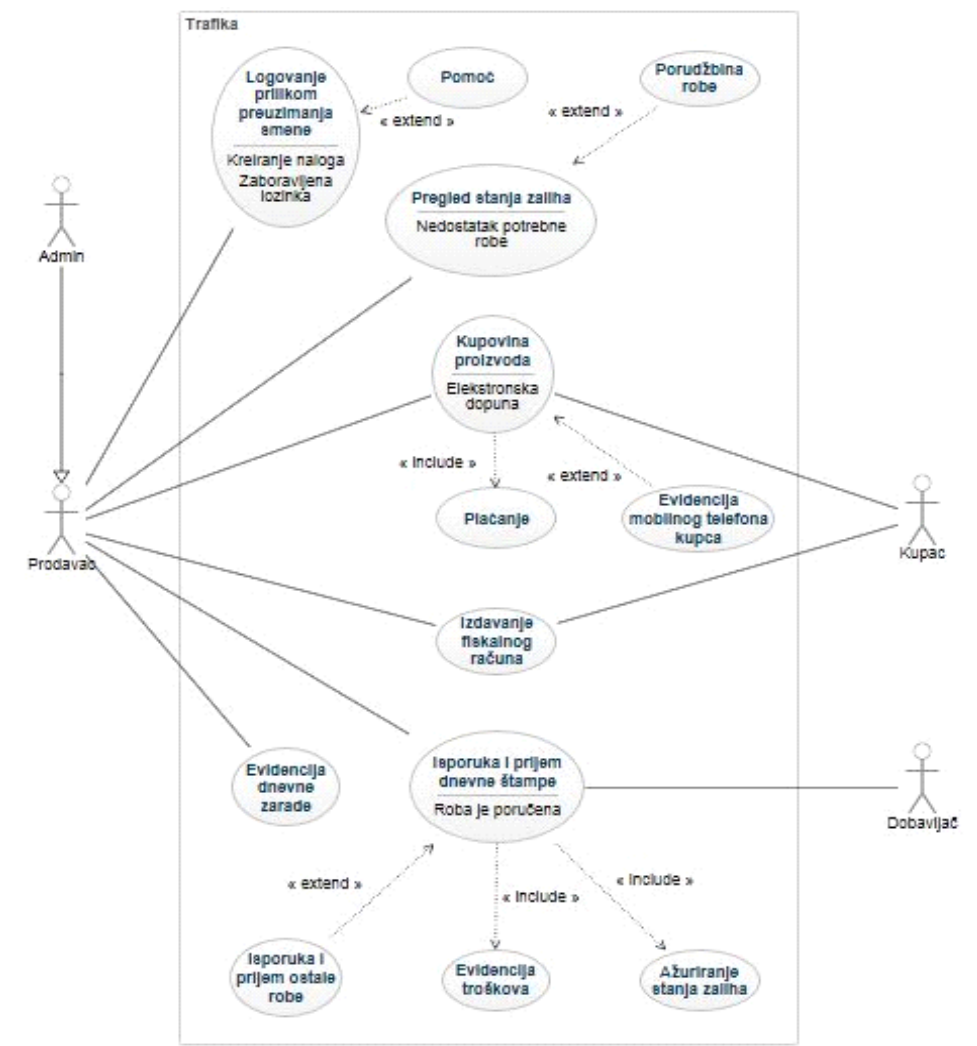
Opis: Dobavljač isporučuje dnevnu štampu, a ukoliko su naručeni ostali proizvodi oni će takođe biti isporučeni od strane istog ili drugog dobavljača. Pri prijemu robe evidentiraju se troškovi nabavke i ažurira se stanje zaliha robe. [***Izuzetak:*** Isporučena je neispravna roba].

Izuzeci:

[**Isporučena je neispravna roba**]. Ukoliko je dobavljač isporučio neispravnu robu i ona nije prošla kontrolu ne vrši se prijem robe, ona se vraća dobavljaču i kreira se reklamacioni zapisnik.

Uslovi koji moraju biti zadovoljeni nakon izvršavanja: Roba je uspešno isporučena i primljena, evidentirani su troškovi nabavke i ažurirano je stanje zaliha.

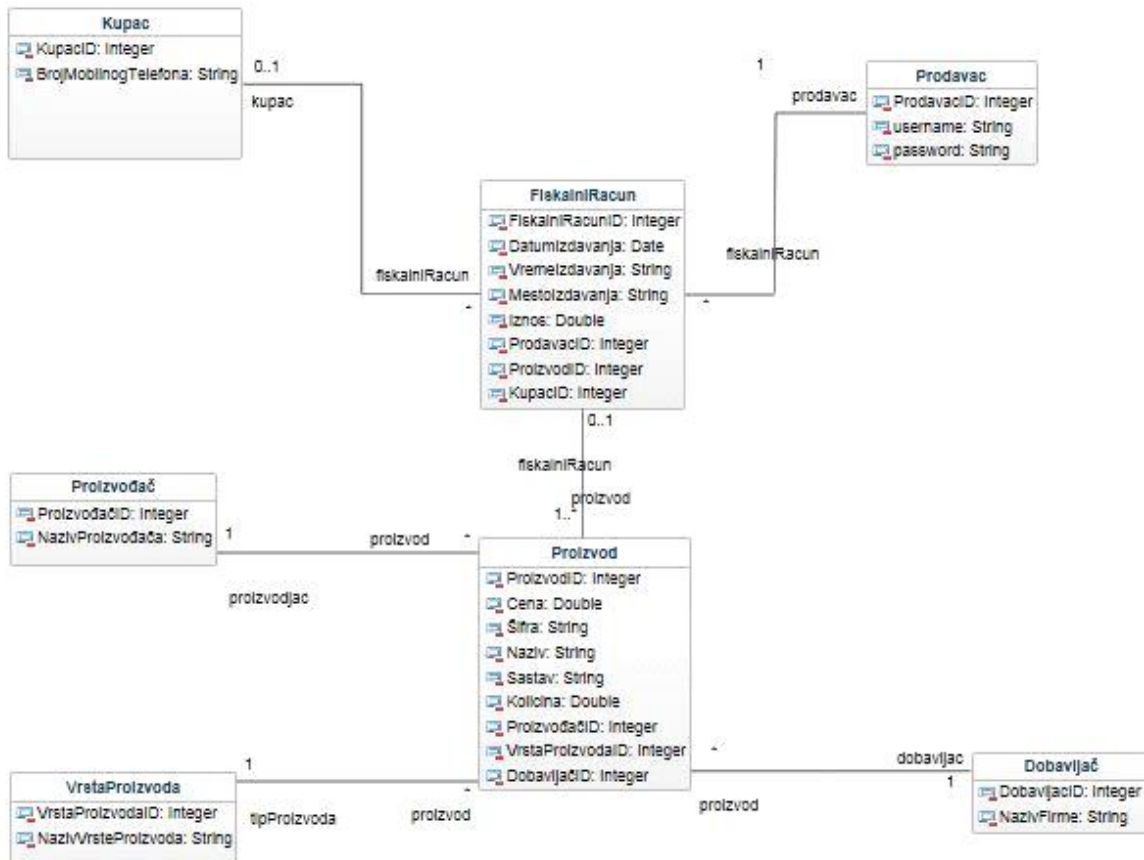
Na osnovu datih opisa slučajeva upotrebe kreiran je dijagram prikazan na *Slici 1*.



Slika 1 – Dijagram slučajeva upotrebe

3.2. Dijagram klasa

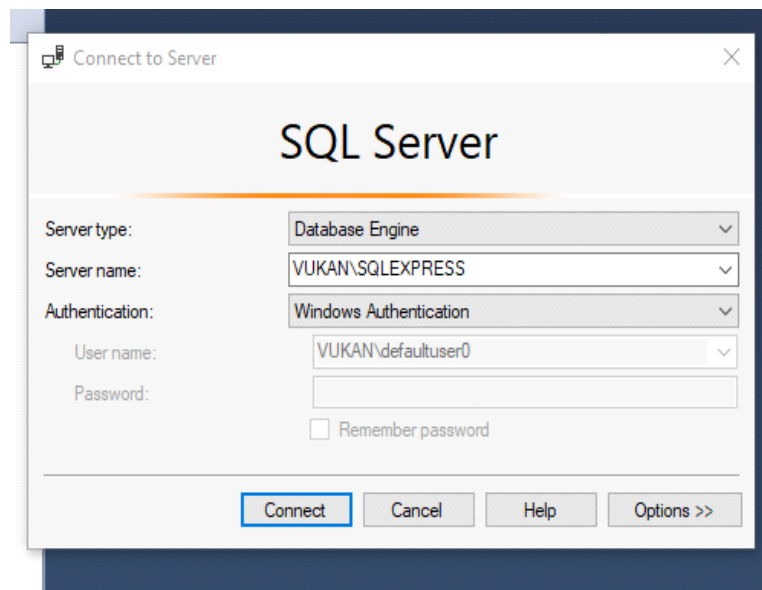
Nakon uspešno kreiranog dijagrama slučajeva upotrebe, pristupamo izradi **dijagrama klasa** koji predstavlja vizuelni prikaz modela statičke strukture sistema. U slučaju primera **trafike** kreiran je dijagram klasa prikazan na *Slici 2*.



Slika 2 – Dijagram klasa

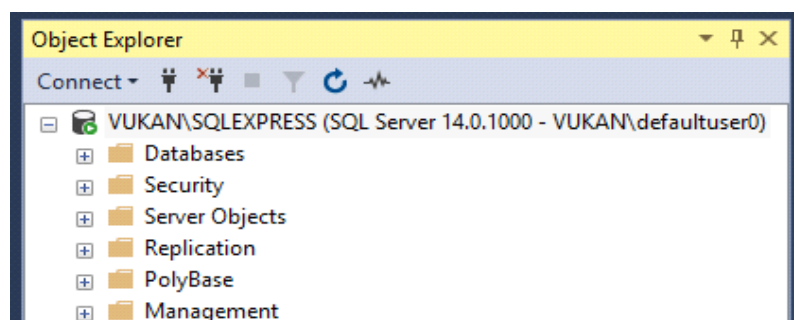
4. Kreiranje baze podataka

Sledeći korak u izradi aplikacije jeste kreiranje **baze podataka**. Na samom početku, potrebno je pokrenuti **SQL Server Management Studio Express**. Kada se to uradi, pojaviće se prozor sa dijalogom, kao što je prikazano na *Slici 3*. U prikazanom dijalogu, potrebno je podesiti parametre **Server type** i **Authentication**, kao što je prikazano na *Slici 3*. Napomena: **Server name** nije potrebno menjati, ono predstavlja naziv mašine na kojoj se radi.



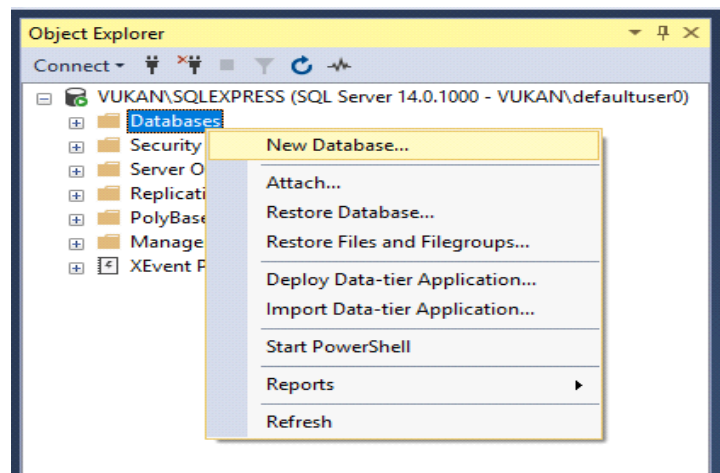
Slika 3 – Ostvarivanje konekcije na server

Nakon podešavanja parametara, klikom na dugme *Connect*, dijalog će se zatvoriti, a otvoriće se novi prozor koji treba da izgleda kao što je prikazano na *Slici 4*.



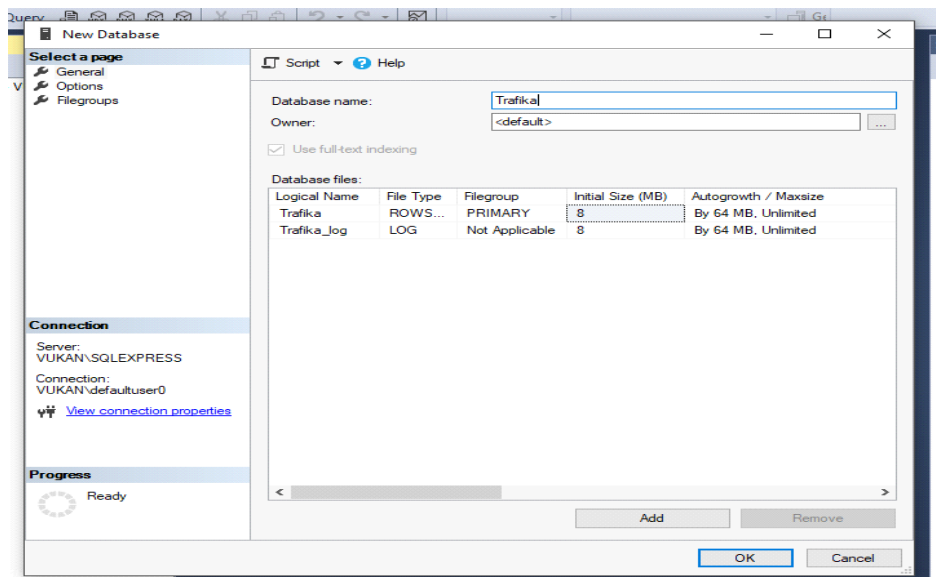
Slika 4 – Izgled Object Explorera nakon ostvarivanja konekcije na bazu

Sada je potrebno kreirati bazu podataka za **Studentsku aplikaciju**. Baza se kreira tako što se klikne *desnim klikom* na **Database** u *treeview*-u koji se nalazi sa leve strane prozora u okviru *Object Explorer*-a. Otvoriće se lista u kojoj je potrebno selektovati **New Database** (Slika 5).



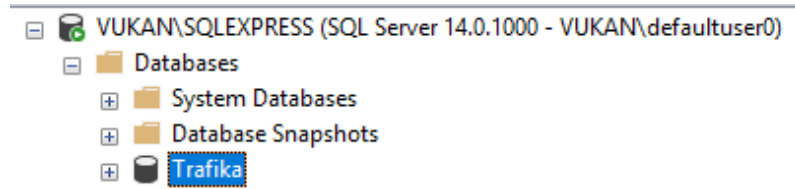
Slika 5 – Kreiranje baze podataka

Otvoriće se prozor u kom će se kreirati **baza podataka**. Potrebno je dati naziv bazi, u ovom slučaju naziv će biti *Trafika*. Ostale parametre nije potrebno menjati. Kliknuti **OK** (Slika 6).



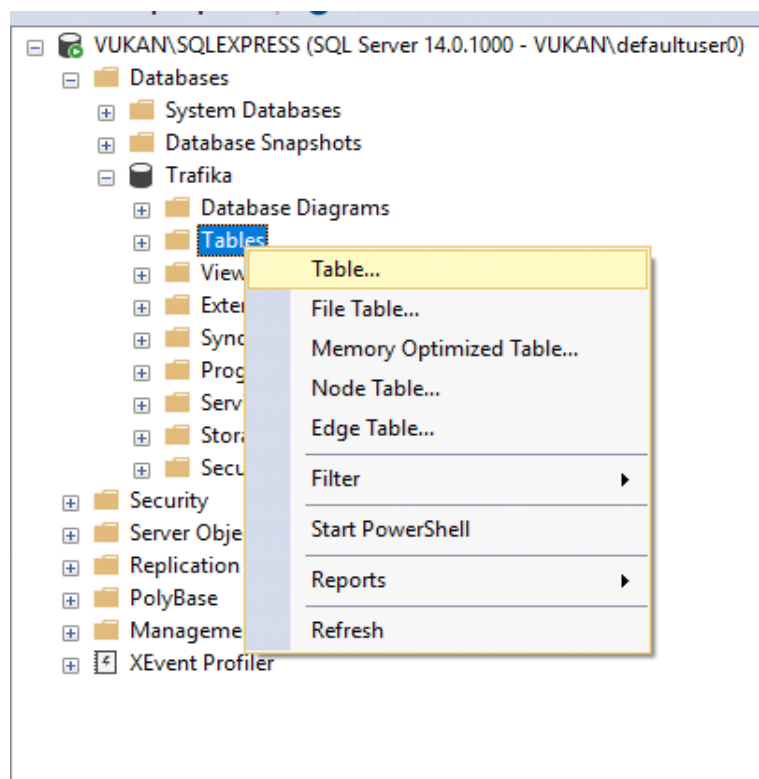
Slika 6 – Podešavanje parametara pri kreiranju baze podataka

Posle uspešnog kreiranja baze podataka u *Object explorer-u*, u folderu *Database* se pojavljuje kreirana baza podataka *Trafika* (Slika 7).



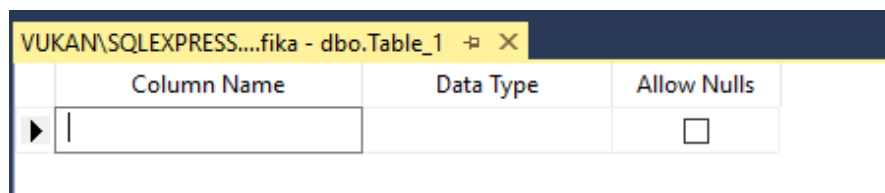
Slika 7 – Prikaz kreirane baze podataka

Proširivanjem baze *Trafika* u stablu *Object Explorer-a*, pojavljuje se više datoteka. Desnim klikom kliknuti na datoteku *Tables* i odabrati *Table* (Slika 8). Nakon klika na ovu opciju, sa desne strane otvara se prozor za definisanje tabele, pri čemu će tabela imati predefinisani naziv *Table_1* (Slika 9).



Slika 8 – Kreiranje tabele unutar baze podataka

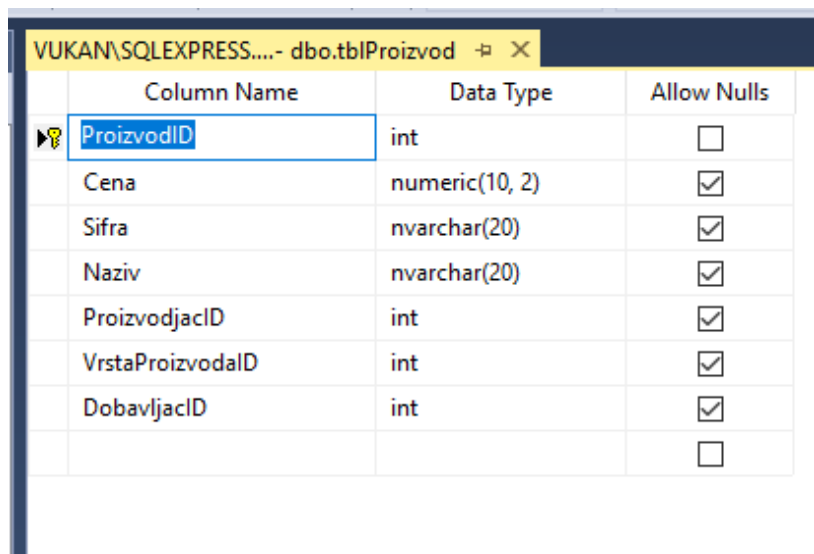
Sledeće što je potrebno uraditi jeste kreirati kolone tabele tako što će se popuniti polja **Column Name**, **Data Type**, **Allow Nulls**. U kolonu *Column Name* se upisuje naziv atributa. U koloni *Data Type* se bira tip podatka za željeni atribut, dok se u koloni *Allow Nulls* selektuje da li određeno polje može imat *null* vrednosti. Preporučuje se da *Allow Nulls* bude check-irano za sve kolone osim za primarni ključ kako bi se izbegli određeni problemi prilikom upisa rekorda u tabelu (Slika 9).



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

Slika 9 – Dodavanje obeležja unutar tabele

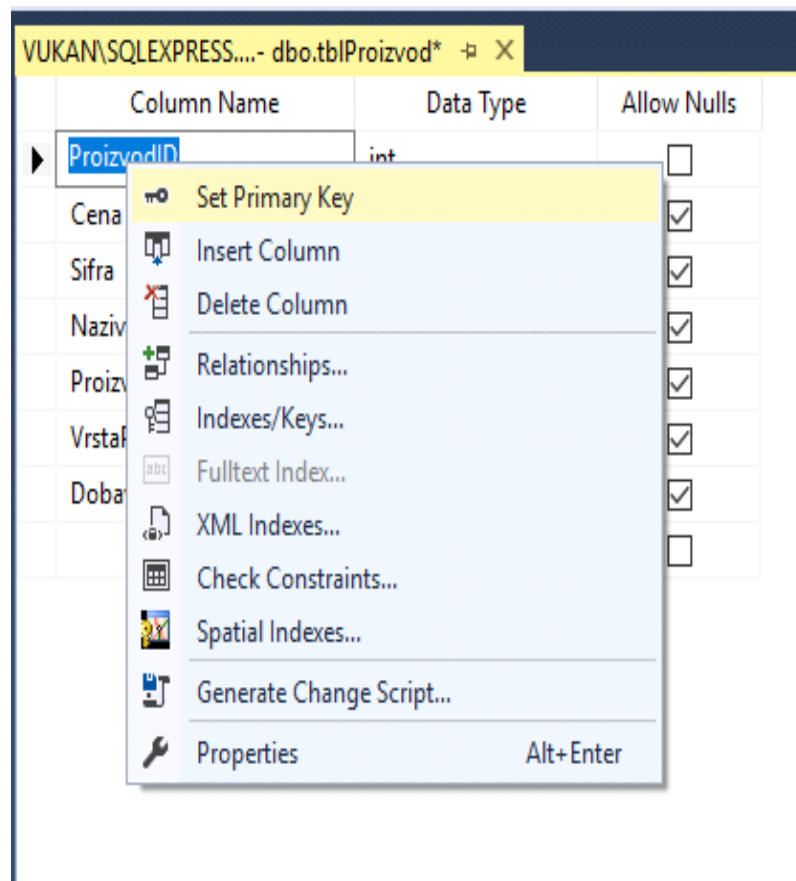
Svaka tabela neke baze podataka mora posedovati **primarni ključ - ID** i bilo koja kolona može da se deklariše kao primarni ključ. U ovom slučaju, kolona *ProizvodID* treba da se deklariše kao primarni ključ. Kao takva, ova kolona će zbog svoje jedinstvene vrednosti imati tip podatka *int* i neće imati *null* vrednosti (Slika 10).



Column Name	Data Type	Allow Nulls
ProizvodID	int	<input type="checkbox"/>
Cena	numeric(10, 2)	<input checked="" type="checkbox"/>
Sifra	nvarchar(20)	<input checked="" type="checkbox"/>
Naziv	nvarchar(20)	<input checked="" type="checkbox"/>
ProizvodjacID	int	<input checked="" type="checkbox"/>
VrstaProizvodaID	int	<input checked="" type="checkbox"/>
DobavljacID	int	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

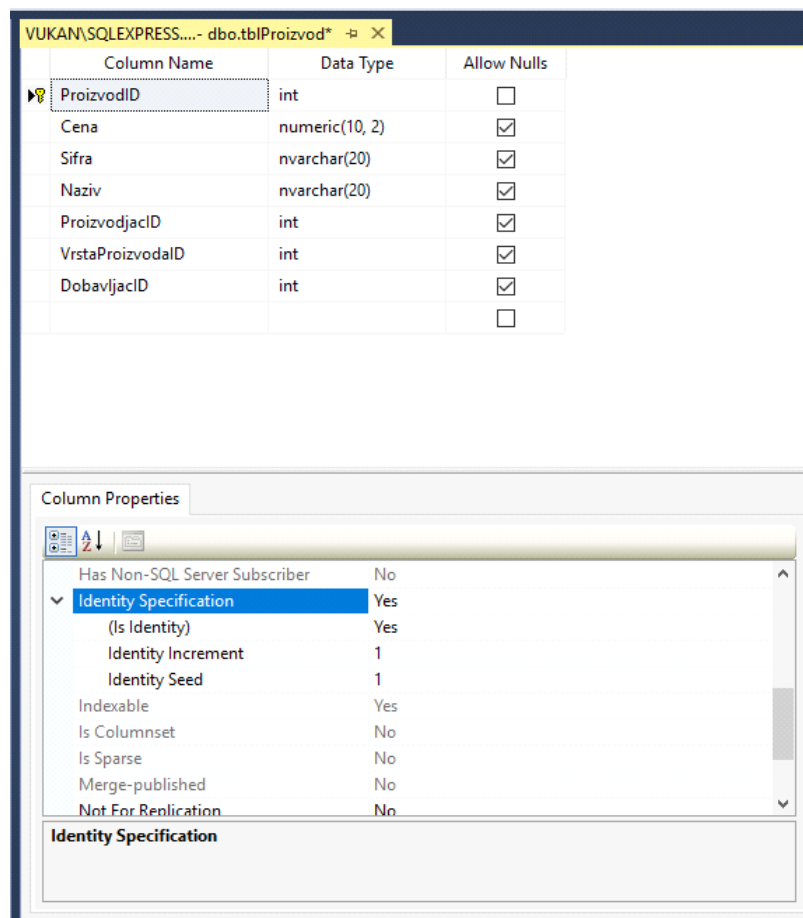
Slika 10 – Prikaz kreiranih obeležja

Da bi se kolona *ProizvodID* deklarirala kao primarni ključ, potrebno je kliknuti desnim klikom na crnu strelicu koja se nalazi pored naziva ove kolone. Otvoriće se lista u kojoj treba odabrati opciju *Set Primary Key* (Slika 11). Ovom komandom se kolona *ProizvodID* deklarira kao primarni ključ.



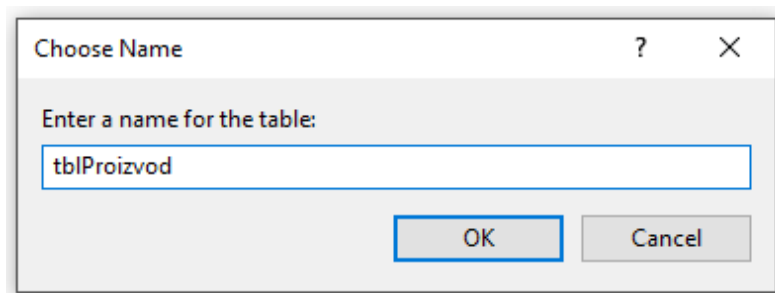
Slika 11 – Definisanje primarnog ključa

Da bi **SUBP (Sistem za upravljanje bazom podataka)** 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, *Identity Seed* od koje vrednosti počinju da se memorišu ključevi (Slika 12).



Slika 12 – Podešavanja automatske inkrementacije vrednosti obeležja primarnog ključa

Čuvanje tabele se vrši pritiskanjem tastera **Ctrl** i **S** (Strl+S). SQL Server Managment Studio Express će pitati pod kojim nazivom se želi sačuvati tabela. Naziv tabele se najčešće navodi sa prefiksom "tbl", u ovom slučaju *tblProizvod* (Slika 13).



Slika 13 – Čuvanje kreirane tabele

Isti proces bi se trebao ponoviti za kreiranje tabela *tblKupac*, *tblProdavac*, *tblVrstaProizvoda*, *tblProizvodjac*, *tblDobavljac* i *tblFiksalniRacun* (Slike 14 – 19).

Column Name	Data Type	Allow Nulls
KupacID	int	<input type="checkbox"/>
BrojMobilnogTelefona	nvarchar(10)	<input checked="" type="checkbox"/>

Slika 14 – Kreiranje tabele Kupac

Column Name	Data Type	Allow Nulls
ProdavacID	int	<input type="checkbox"/>
Username	nvarchar(20)	<input checked="" type="checkbox"/>
Password	nvarchar(20)	<input checked="" type="checkbox"/>

Slika 15 – Kreiranje tabele Prodavac

Column Name	Data Type	Allow Nulls
VrstaProizvodaID	int	<input type="checkbox"/>
NazivVrsteProizvoda	nvarchar(20)	<input checked="" type="checkbox"/>

Slika 16 – Kreiranje tabele VrstaProizvoda

VUKAN\SQLEXPRESS....bo.tblProizvodjac			
	Column Name	Data Type	Allow Nulls
PK	ProizvodjacID	int	<input type="checkbox"/>
	NazivProizvodjaca	nvarchar(30)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Slika 17 – Kreiranje tabele Proizvodjac

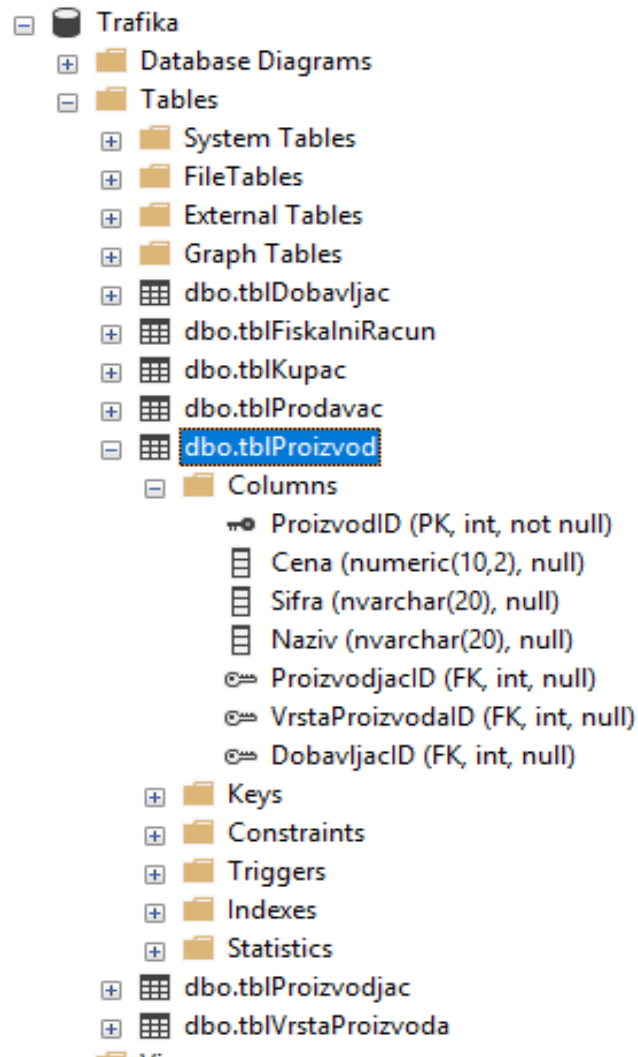
VUKAN\SQLEXPRESS....dbo.tblDobavljac			
	Column Name	Data Type	Allow Nulls
PK	DobavljacID	int	<input type="checkbox"/>
	NazivFirme	nvarchar(30)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Slika 18 – Kreiranje tabele Dobavljac

VUKAN\SQLEXPRESS....tblFiskalniRacun			
	Column Name	Data Type	Allow Nulls
PK	FiskalniRacunID	int	<input type="checkbox"/>
	DatumIzdavanja	date	<input checked="" type="checkbox"/>
	Vremelzdavanja	nvarchar(10)	<input checked="" type="checkbox"/>
	Mestolzdavanja	nvarchar(10)	<input checked="" type="checkbox"/>
	Iznos	numeric(10, 2)	<input checked="" type="checkbox"/>
	Kolicina	int	<input checked="" type="checkbox"/>
	ProdavacID	int	<input checked="" type="checkbox"/>
	ProizvodID	int	<input checked="" type="checkbox"/>
	KupacID	int	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

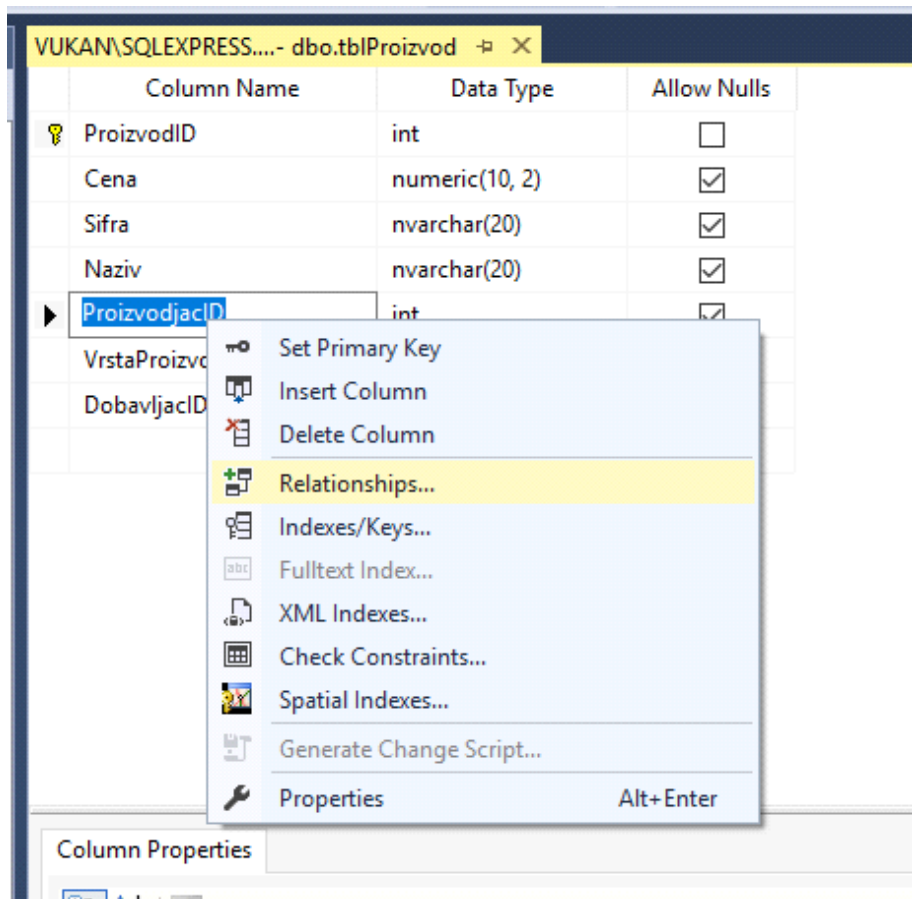
Slika 19 – Kreiranje tabele FiskalniRacun

Nakon toga bi u *treeview*-u koji se nalazi sa leve strane prozora u okviru *Object Explorer*-a trebala da se nalazi baza podataka *Trafika* u kojoj se pored već kreirane *tblProizvod* nalaze i novokreirane tabele sa odgovarajućim atributima (*Slika 20*).



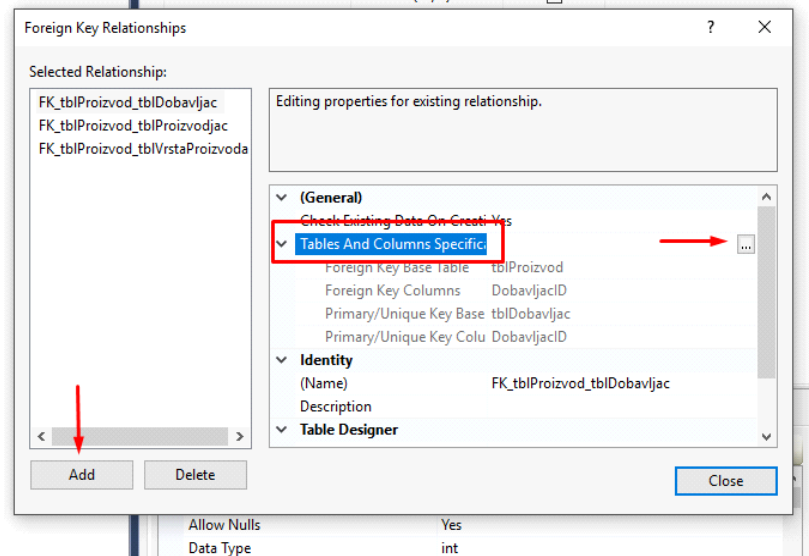
Slika 20 – Object Explorer prikaz kreiranih tabela

Kao što se može videti na slici 12, u *tblProizvod* se nalaze tri *Foreign key*-a (*ProizvodjacID*, *VrstaProizvodaID* i *DobavljacID*). Prilikom unosa atributa u tabelu *tblProizvod* potrebno je uneti i attribute koji imaju nazive *ProizvodjacID*, *VrstaProizvodaID* i *DobavljacID*, a vrednost *DataType*-a za njih staviti na *int*. Zatim je potrebno kliknuti desnim klikom na crnu strelicu koja se nalazi pored naziva ove kolone. Otvoriće se lista u kojoj treba odabrati opciju *Relationships* (Slika 21).



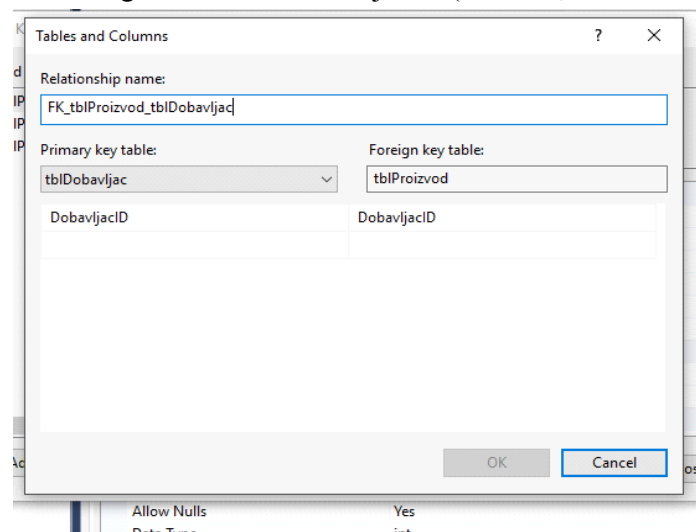
Slika 21 – Dodavanje veza

Otvara se prozor *Foreign key Relationships* (Slika 22) u kojem je potrebno prvo kliknuti na dugme *Add* a zatim proširiti polje *Tables and Columns Specification* i kliknuti na tri tačke koje se nalaze desno.



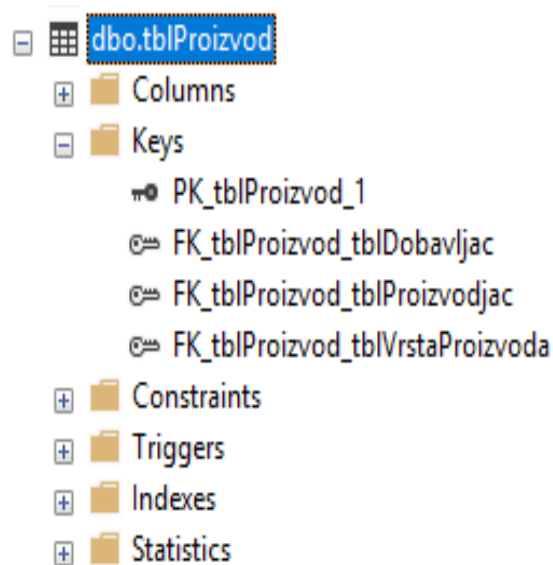
Slika 22 – Dodavanje stranog ključa

U otvorenom prozoru *Tables and Columns* potrebno je u okviru *Foreign key table*, konkretno iz tabele *tblProizvod*, iz padajuće liste izabrati atribut koji smo mi definisali a koji želimo da povežemo sa drugom tabelom. Zatim u okviru *Primary key table* biramo tabelu sa kojom vršimo povezivanje i iz padajuće liste biramo ključ te tabele sa kojim ćemo povezati željeni atribut. Nakon što smo sve ovo odradili, prozor treba da nam izgleda kao na sledećoj slici (Slika 23).



Slika 23 – Kreiranje veza između tabela

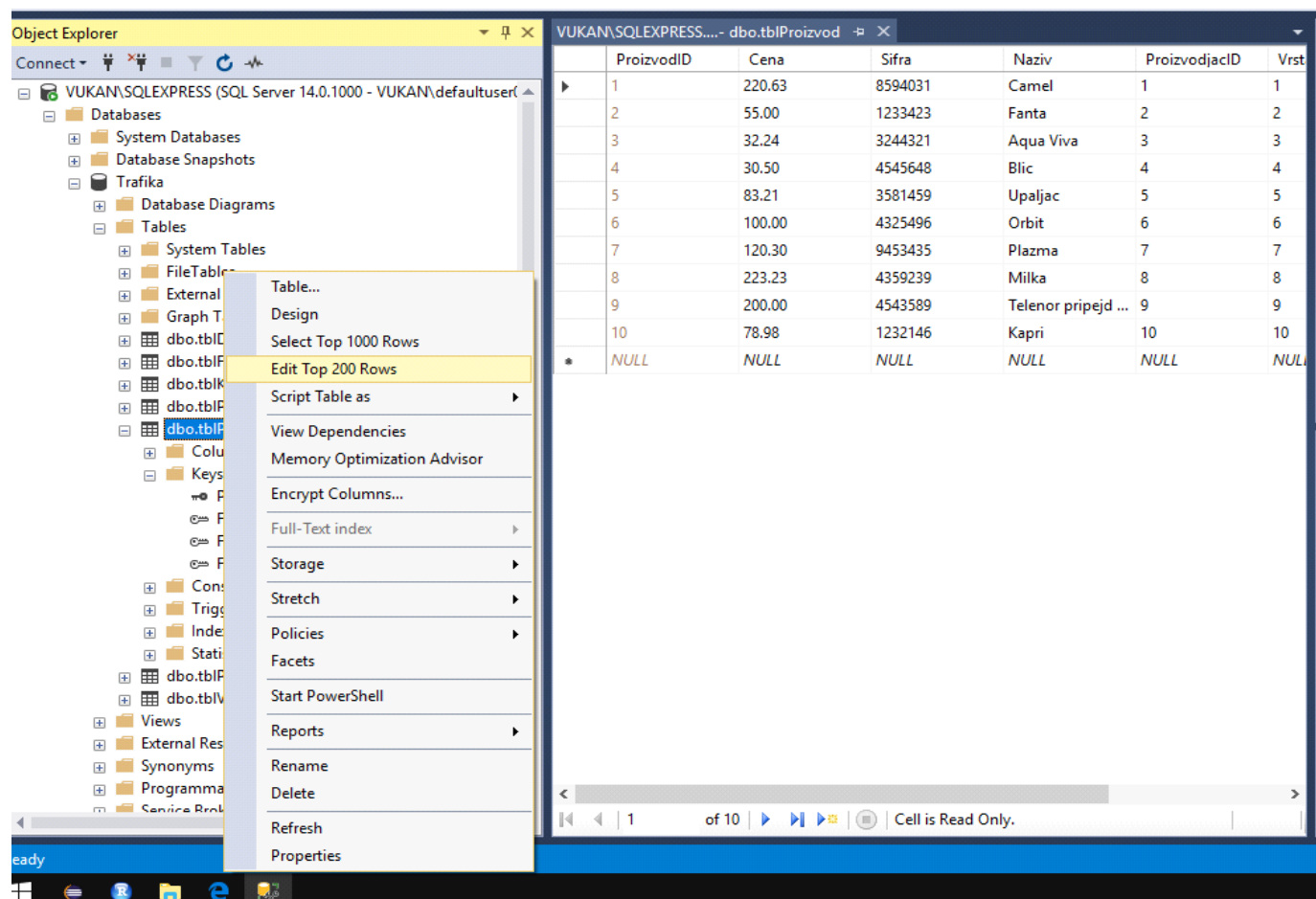
Dodavanjem *relationships*-a atributima *ProizvodjacID*, *VrstaProizvodaID* i *DobavljacID* u tabeli *tblProizvod* oni postaju strani ključevi. Da li smo uspešno odradili povezivanje tabela znaćemo tako što ćemo proveriti da li se oni nalaze u *treeview*-u u okviru tabele *tblProizvod* pod kategorijom *Keys* (Slika 24).



Slika 24 – TreeView prikaz kreiranih ključeva

Isti postupak potrebno je ponoviti za sve tabele koje u sebi sadrže strane ključeve koji su primarni ključevi u drugim tabelama, a to je u našem slučaju još tabela *tblFiskalniRacun* (*ProdavacID*, *ProizvodID*, *KupacID*).

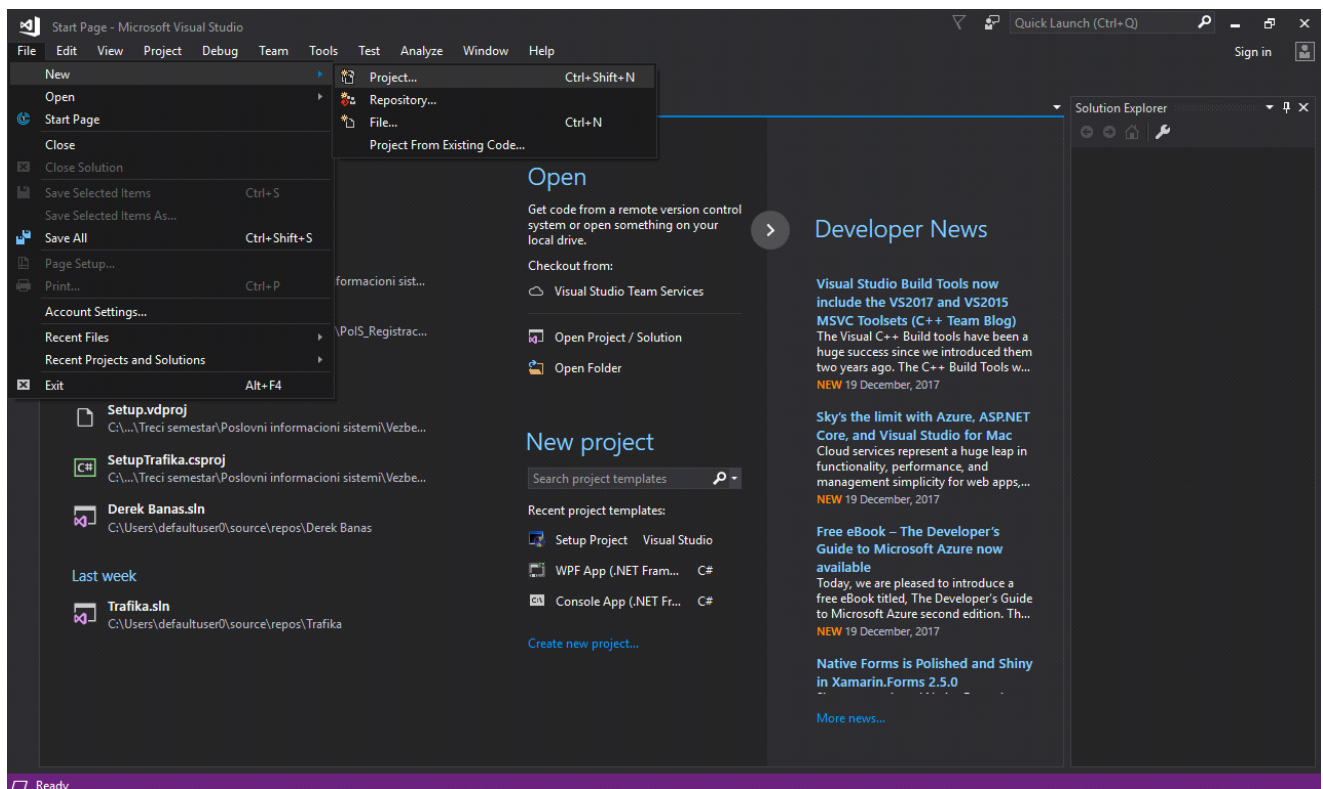
Sada su tabele kreirane i povezane, a dodavanje novih podataka u ovu bazu se može izvršiti desnim klikom na naziv tabele, gde se iz liste opcija treba odabrati **Edit Top 200 Rows** (Slika 25).



Slika 25 – Prikaz kreirane tabele i dodavanje podataka

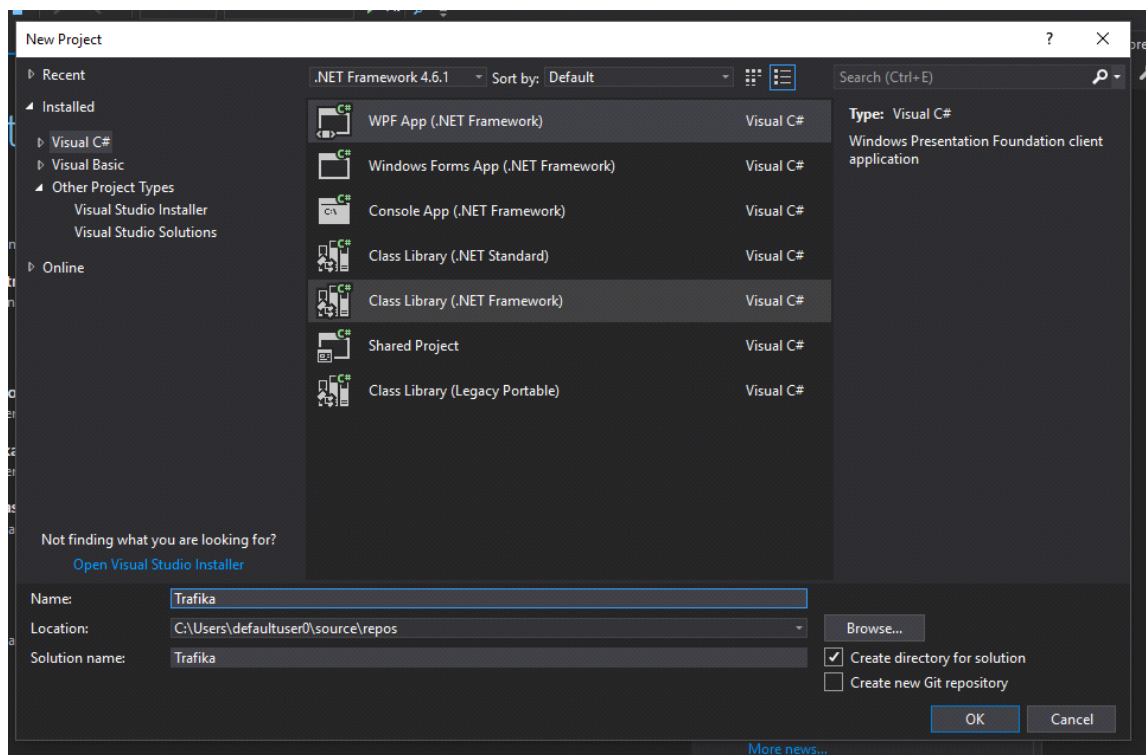
5. Kreiranje studentske WPF aplikacije (desktop aplikacije)

U okviru ovog poglavlja objasniće se kako se kreira **WPF** (*Windows Presentation Foundation*) aplikacija. Nakon pokretanja **Visual Studija** potrebno je kliknuti na *File* meni a zatim odabrati *New Project* (Slika 26).



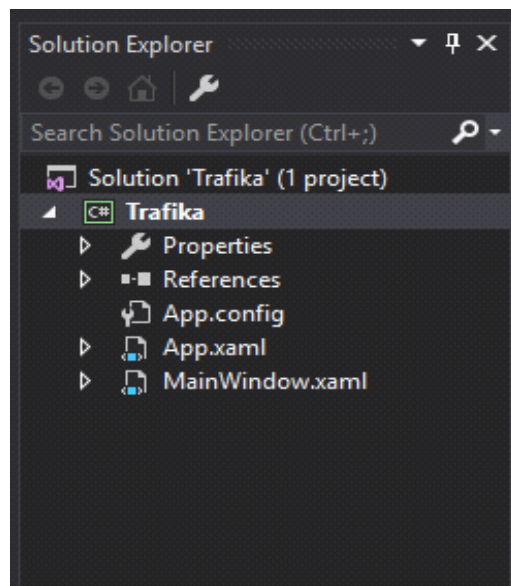
Slika 26 – Kreiranje novog projekta u Visual Studiju

Odaberi *Windows Classic Desktop*, zatim *WPF Application*. Navesti ime aplikacije i kliknuti **OK** (Slika 27).



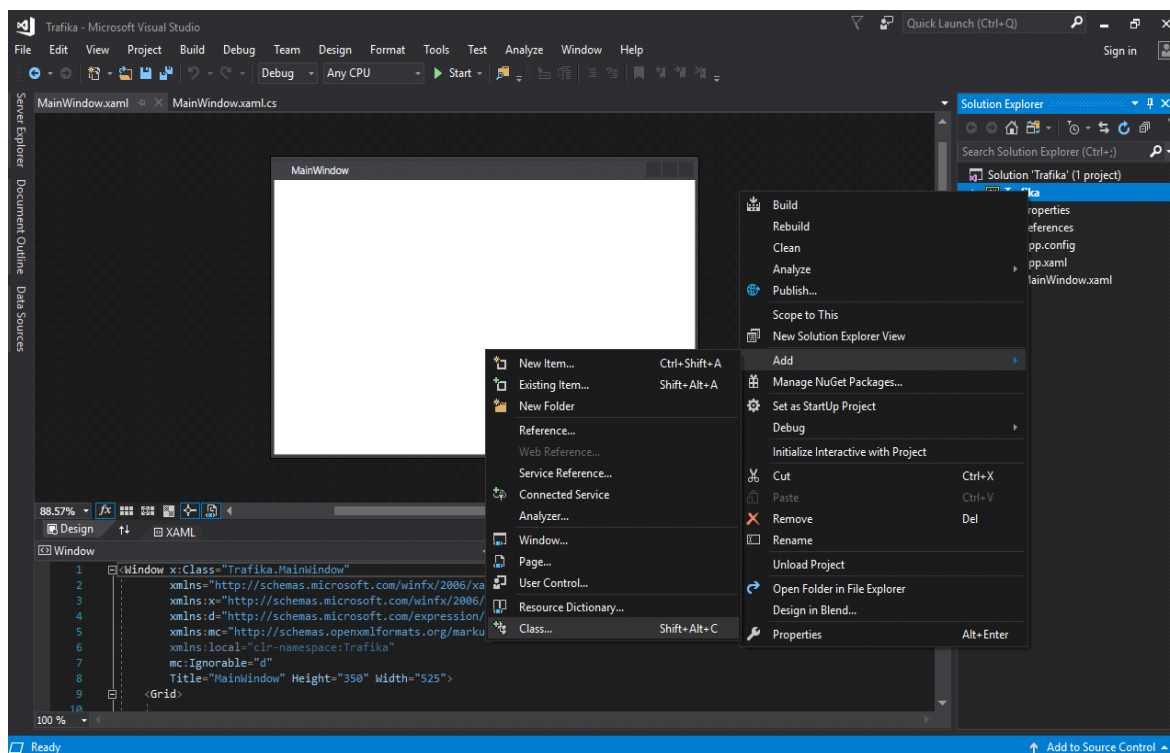
Slika 27 – Kreiranje WPF aplikacije

U *Solution Explorer*-u će se pojaviti napravljena aplikacija (*Slika 28*).



Slika 28 – Solution Explorer kreirane WPF aplikacije

Sledeći korak je kreiranje konekcije u aplikaciji, ka bazi podataka. Za početak, kreira se klasa u kojoj će se nalaziti statička metoda putem koje će se ostvarivati veza na prethodno kreiranu bazu. Klasa se kreira desnim klikom na naziv projekta (*Trafika*), a zatim *Add / Class...* (*Slika 29*).



Slika 29 – Kreiranje klase

Novokreiranoj klasi se dodeljuje naziv “*Konekcija*” i njen sadržaj prikazan je na *Slici 30*. Na samom početku uključujemo imenski prostor *System.Data.SqlClient*. Telo klase sadrži samo jednu statičku metodu pod nazivom “*KreirajKonekciju*”, koja će se u kasnijem razvoju aplikacije pozivati po potrebi odnosno svaki put kada budemo želeli da ostvarimo vezu sa bazom podataka, stoga je povratni tip ove metode instanca klase *SqlConnection*. Unutar same metode prikazana su podešavanja parametara konekcije i to:

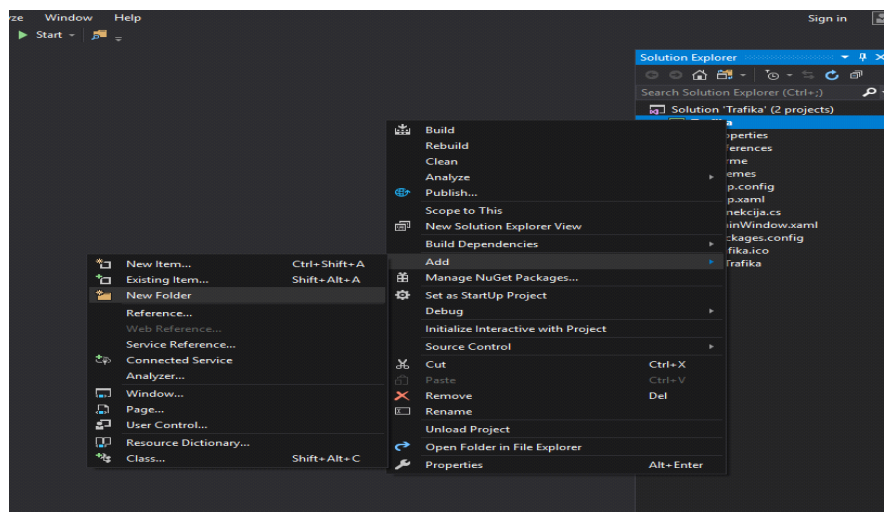
- DataSource – naziv servera na kojem je baza podataka smeštena.
- InitialCatalog – naziv baze podataka kojoj želimo da pristupimo.
- IntegratedSecurity – ukoliko se baza nalazi na lokalnoj mašini ovaj parametar postavljamo na true.

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

```
1  using System.Data.SqlClient; // dodajemo
2
3  namespace Trafika
4  {
5      class Konekcija
6      {
7          public static SqlConnection KreirajKonekciju()
8          {
9              SqlConnectionStringBuilder connectionStringBuilder = new SqlConnectionStringBuilder();
10
11              connectionStringBuilder.DataSource = @"VUKAN\SQLEXPRESS";
12              connectionStringBuilder.InitialCatalog = @"Trafika";
13              connectionStringBuilder.IntegratedSecurity = true;
14
15              string connection = connectionStringBuilder.ToString();
16              SqlConnection Connection = new SqlConnection(connection);
17
18              return Connection;
19          }
20      }
21  }
22
```

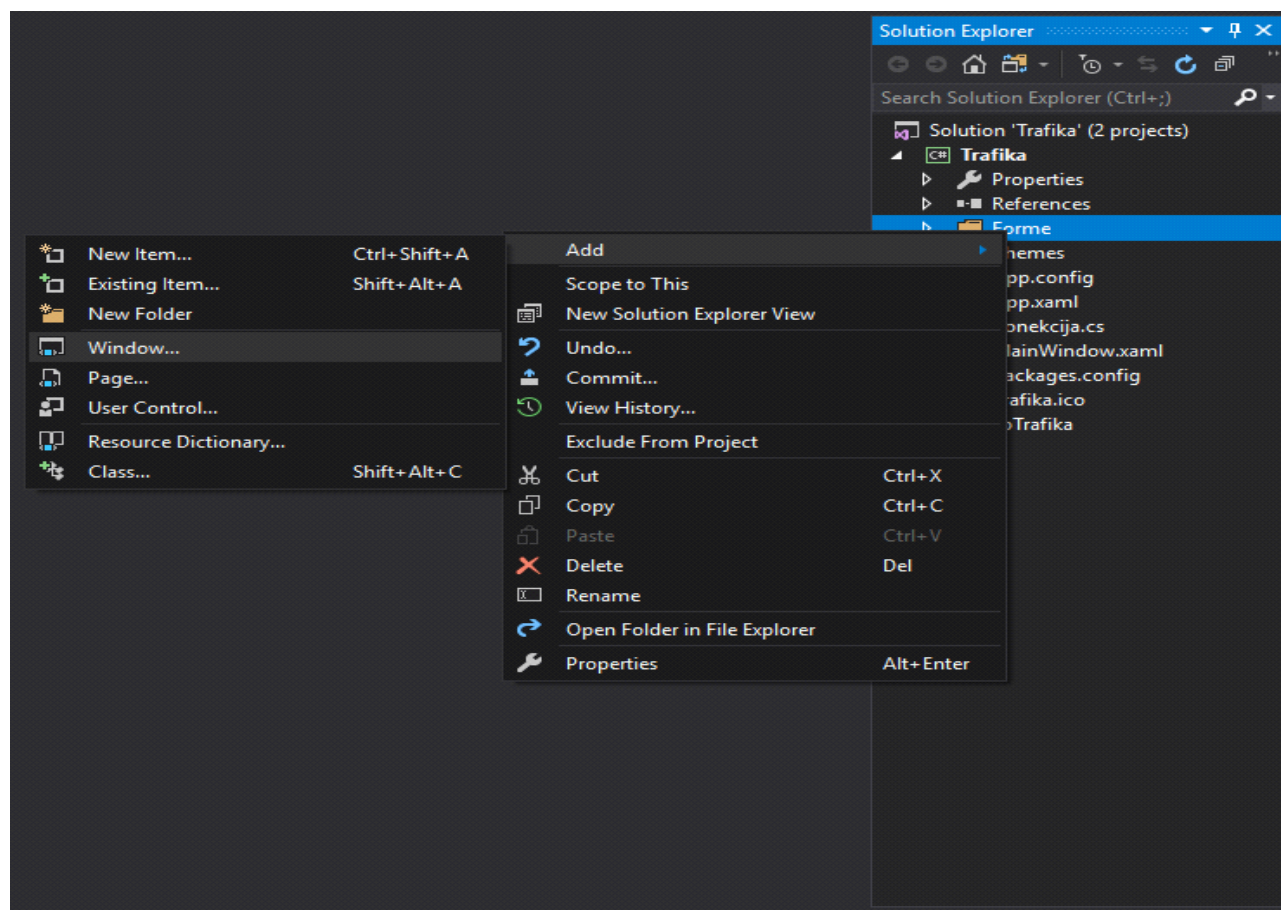
Slika 30 – Klasa Konekcija

Sledeći korak u izradi aplikacije biće kreiranje **formi** 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 7 formi za, prethodno kreiranih, 7 tabela u bazi podataka. Kreiranje foldera se vrši na sličan način kao kreiranje klase, desnim klikom na projekat, a zatim *Add/ New Folder*.



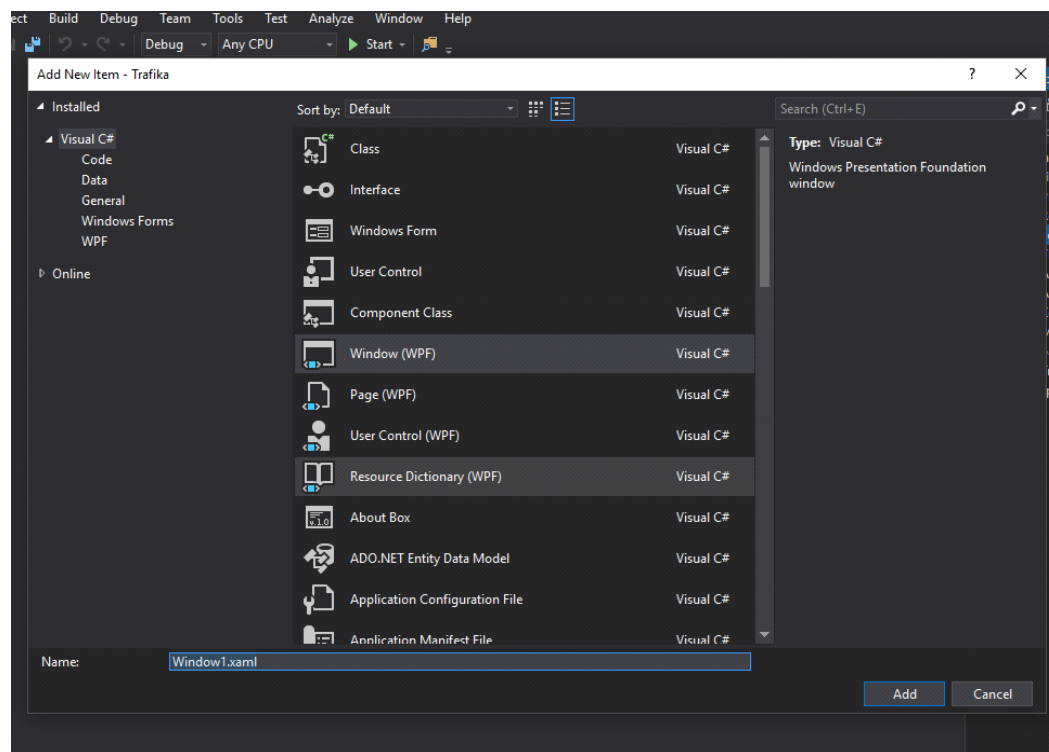
Slika 31 – Kreiranje novog foldera

Nakon kreiranja foldera, potrebno je dodati novi prozor za svaku od formi. WPF prozor, unutar folder, kreira se desnim klikom na folder, a zatim *Add/Window* (Slika 32).



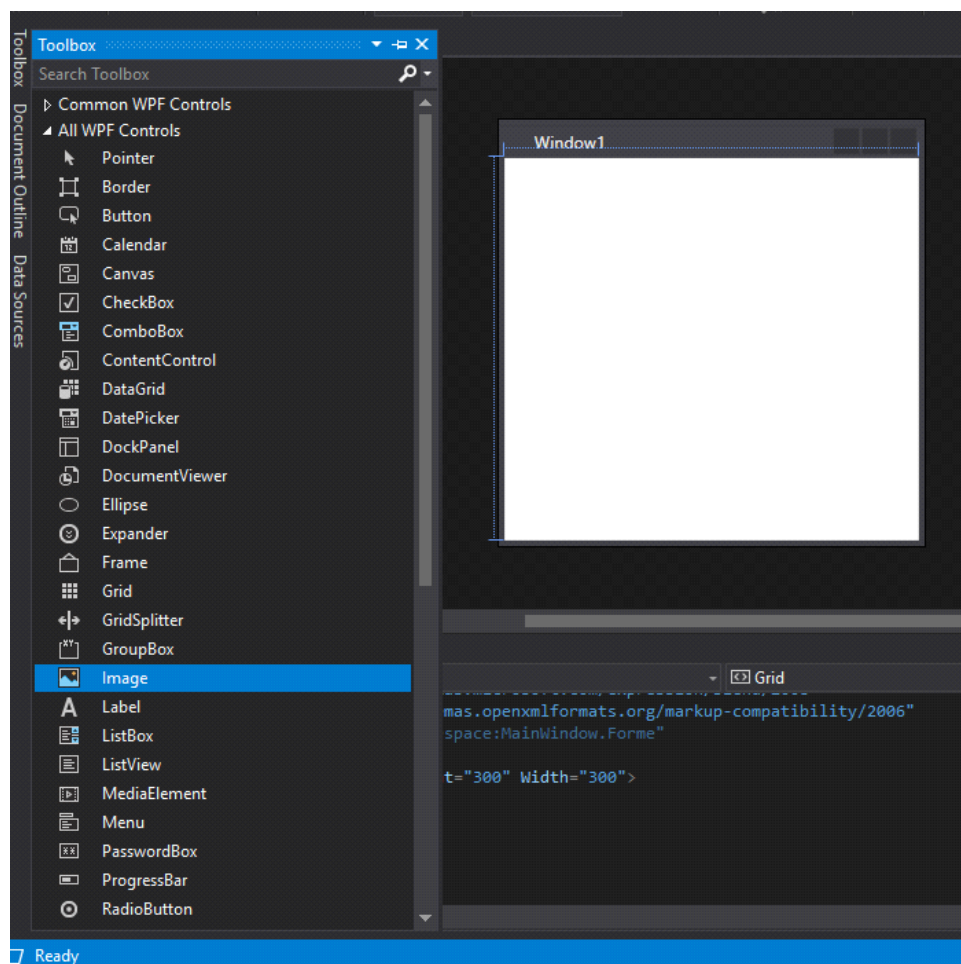
Slika 32 – Dodavanje formi unutar foldera

Nakon toga prikazaće se prozor da dodavanje novog *Item*-a unutar kog je potrebno podesiti naziv i zatim kliknuti na *Add* (Slika 33).



Slika 33 – Dodavanje novog prozora

Kada je novi prozor izgenerisan, potrebno je, unutar njega, dodati određene elemente putem kojih će se ostvarivati potrebne funkcionalnosti. Na levoj strani prozora, klikom na *Toolbox*, prikazaće se lista svih ugrađenih **WPF komandi** (Slika 34). 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 35).



Slika 34 – Dodavanje komandi iz Toolbox-a

Forma za dodavanje novog *Prodavca* u bazu podataka, treba da sadrži sledeće elemente (Slika 34). Dva osnovna *TextBox*-ova unutar kojih će biti upisane vrednosti obeležja. Takođe, uz svaki *TextBox*, mora postojati naziv (*Label*) koji će opisivati njegovu namenu. Na kraju, svaka forma morati imati ugrađeno dugme **Sačuvaj**, putem kojih će se vrednosti upisivati u bazu, i dugme **Otkazi** - koje korisniku omogućava zatvaranje forme (Slika 35). Ove funkcionalnosti dugmića postižu se klikom miša na odgovarajuće dugme čime se u *Code Behind*-u generišu metode *btnSačuvaj_Click* i *btnZatvori*.

```

<Window x:Class="Trafika.Forme.frmProdavac"
        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:Trafika.Forme"
        mc:Ignorable="d"
        Title="Prodavac" Height="223.482" Width="458.863"
        Icon="C:\Users\defaultuser0\OneDrive\2. godina\Treci semestar\Poslovni informacioni sistemi\Vezbe\Trafika\Ikonice\Prodavac.png"
        ResizeMode="CanMinimize" WindowStartupLocation="CenterScreen" FontWeight="DemiBold">

    <Grid Background="LightSkyBlue">

        <TextBox x:Name="txtUserName" HorizontalAlignment="Left" Height="26" Margin="187,35,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="250"/>
        <Label Content="Username" HorizontalAlignment="Left" Margin="10,35,0,0" VerticalAlignment="Top" Width="160"/>
        <TextBox x:Name="txtPassword" HorizontalAlignment="Left" Height="26" Margin="187,80,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="250"/>
        <Label Content="Password" HorizontalAlignment="Left" Margin="10,80,0,0" VerticalAlignment="Top" Width="160"/>
        <Button Content="Otkazi" HorizontalAlignment="Left" Margin="266,143,0,0" VerticalAlignment="Top" Width="171" Height="26" Click="btnOtkaci_Click"
            Background="Red" Cursor="Hand"/>
        <Button Content="Sačuvaj" HorizontalAlignment="Left" Margin="10,143,0,0" VerticalAlignment="Top" Width="171" Height="26" Click="btnSacuvaj_Click"
            Background="LawnGreen" Cursor="Hand"/>

    </Grid>

</Window>

```

Slika 35 – XAML kod kreiranih komponenti

Vizuelni prikaz forme za unos novog *Prodavca*, izgledaće kao na *Slici 36*.

Slika 36 – Vizuelni prikaz forme za dodavanje novog proizvoda

U *Code Behind*-u ove forme, trebalo bi da se nalazi sledeća poslovna logika (*Slika 37*). Najpre je potrebno importovati imenski prostor *System.Data.SqlClient*. Zatim, potrebno je napraviti novi objekat *SqlConnection* klase, putem kojeg će se ostvarivati veza ka bazi podataka nakon klika na dugme *Sačuvaj*. Takođe, u konstruktoru ovog prozora poželjno je podesiti metodu *Focus()*, koja pozicionira kursor na prvo polje koje korisnik popunjava pri pokretanju forme. U slučaju prodavca to je tekst polje za unos njegovog *Username* - a.

```

using System.Windows;
using System.Data;
using System.Data.SqlClient;

namespace Trafika.Forme
{
    public partial class frmProdavac : Window
    {
        public SqlConnection konekcija = Konekcija.KreirajKonekciju();

        public frmProdavac()
        {
            InitializeComponent();
            txtUserName.Focus();
        }

        private void btnSacuvaj_Click(object sender, RoutedEventArgs e) ...

        private void btnOtkaci_Click(object sender, RoutedEventArgs e)
        {
            this.Close();
        }
    }
}

```

Slika 37 – Sadržaj CodeBehind-a forme frmProdavac

Ovakav prikaz pozadinskog koda gotovo je identičan za svaku od 7 kreiranih formi. Ono u čemu je svaka od ovih formi jedinstvena jeste pozadinska funkcionalnost koja se dešava u trenutku kada korisnik klikne na dugme *Sačuvaj*. Na Slici 38 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 nove objekat u bazu podataka. U *values* ove naredbe prosledjuju se pokupljene unete vrednosti iz *TextBox*-ova ove forme. Potrebno je voditi računa da se vrednosti stringova ograde apostrofima. Nakon upešno napravljene *insert* komande, ovaj string se prosleđuje objektu klase *SqlCommand*. Ovaj objekat, kao parametre prima prethodno kreirani string ali i konekciju putem koje 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 je ograditi *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 “hvatati” 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.

```
private void btnSacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();

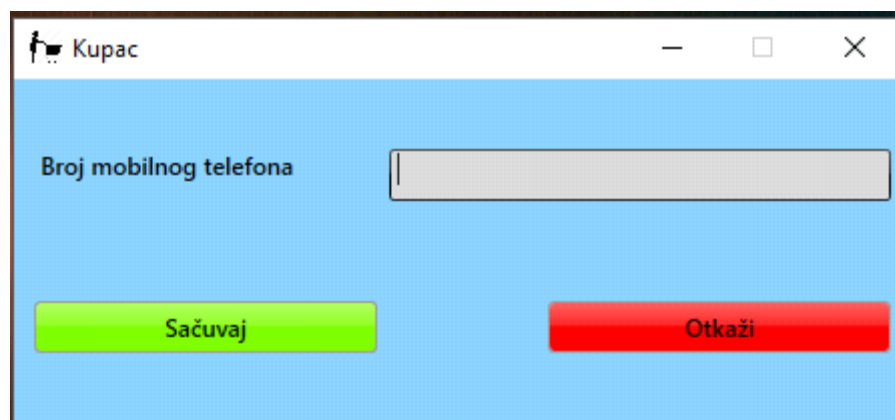
        string insert = @"insert into tblProdavac(Username, Password)
                        values('" + txtUserName.Text + "', '" + txtPassword.Text + "')";

        SqlCommand cmd = new SqlCommand(insert, konekcija);
        cmd.ExecuteNonQuery();
        this.Close();
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrednosti nije validan.", "Greška!", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
}
```

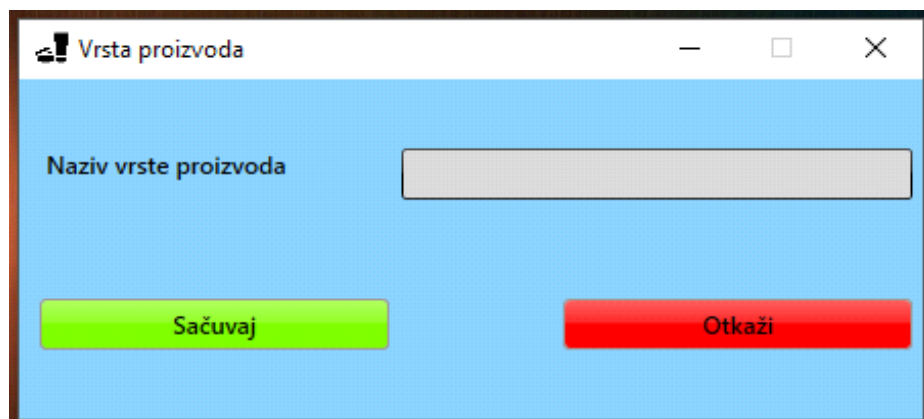
Slika 38 – Sadržaj metode btnSacuvaj_Click za insert novog prodavca

U nastavku će biti prikazan izgled i kod ostalih 6 formi.

Izgled formi za dodavanje kupca, vrste proizvoda, proizvođača i dobavljača takođe ima veoma sličan dizajn (Slika 39., Slika 40., Slika 41, Slika 42, Slika 43).



Slika 39 – Dizajn forme za dodavanje kupca

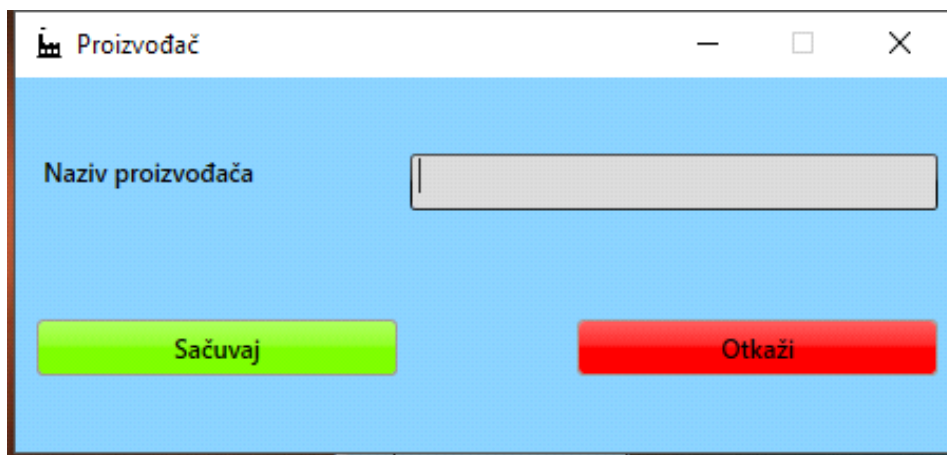


Vrsta proizvoda

Naziv vrste proizvoda

Sačuvaj Otkazi

Slika 40 - Dizajn forme za dodavanje nove vrste proizvoda

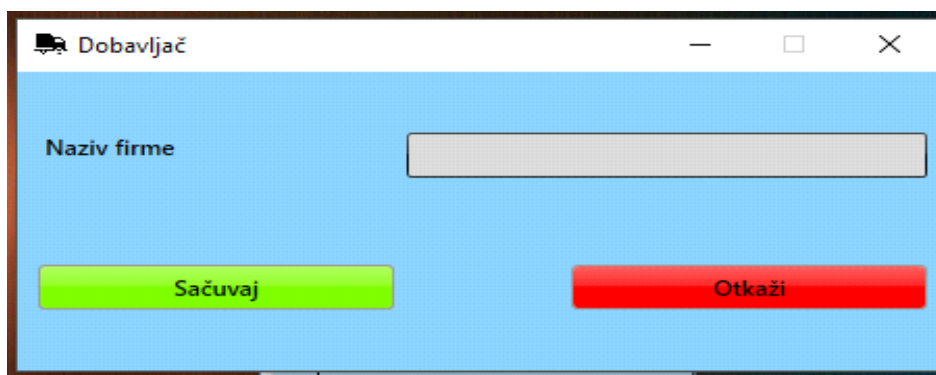


Proizvođač

Naziv proizvođača

Sačuvaj Otkazi

Slika 41 - Dizajn forme za dodavanje novog proizvođača



Dobavljač

Naziv firme

Sačuvaj Otkazi

Slika 42 - Dizajn forme za dodavanje novog dobavljača

```

<Window x:Class="Trafika.Forme.frmKupac"
    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:Trafika.Forme"
    mc:Ignorable="d"
    Title="Kupac" Height="210.902" Width="457.895"
    Icon="C:\Users\defaultuser0\OneDrive\2. godina\Treci semestar\Poslovni informacijski sistemi\Vezbe\Trafika\Ikonice\Kupac.png"
    ResizeMode="CanMinimize" WindowStartupLocation="CenterScreen" FontWeight="DemiBold">

    <Grid Background="LightSkyBlue">

        <TextBox x:Name="txtBrojMobilnogTelefona" HorizontalAlignment="Left" Height="26" Margin="187,35,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
            Width="250"/>
        <Label Content=" Broj mobilnog telefona" HorizontalAlignment="Left" Margin="10,35,0,0" VerticalAlignment="Top" Width="160"/>
        <Button x:Name="btnOtkazi" Content="Otkazi" HorizontalAlignment="Left" Margin="266,111,-145,0" VerticalAlignment="Top" Width="171" Height="26"
            Click="btnOtkazi_Click" Background="Red" Cursor="Hand"/>
        <Button x:Name="btnSacuvaj" Content="Sačuvaj" HorizontalAlignment="Left" Margin="10,111,0,0" VerticalAlignment="Top" Width="171" Height="26"
            Click="btnSacuvaj_Click" Background="LawnGreen" Cursor="Hand"/>

    </Grid>

</Window>

```

Slika 43 – XAML kod forme za dodavanje novog kupca

```

<Window x:Class="Trafika.Forme.frmVrstaProizvoda"
    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:Trafika.Forme"
    mc:Ignorable="d"
    Title="Vrsta proizvoda" Height="210.902" Width="457.895"
    Icon="C:\Users\defaultuser0\OneDrive\2. godina\Treci semestar\Poslovni informacijski sistemi\Vezbe\Trafika\Ikonice\Vrsta Proizvoda.png"
    ResizeMode="CanMinimize" WindowStartupLocation="CenterScreen" FontWeight="DemiBold">

    <Grid Background="LightSkyBlue">

        <TextBox x:Name="txtNazivVrsteProizvoda" HorizontalAlignment="Left" Height="26" Margin="187,35,-145,0" TextWrapping="Wrap"
            VerticalAlignment="Top" Width="250"/>
        <Label Content=" Naziv vrste proizvoda" HorizontalAlignment="Left" Margin="10,35,0,0" VerticalAlignment="Top" Width="160"/>
        <Button Content="Otkazi" HorizontalAlignment="Left" Margin="266,111,-145,0" VerticalAlignment="Top" Width="171" Height="26"
            Click="btnOtkazi_Click" Background="Red" Cursor="Hand"/>
        <Button Content="Sačuvaj" HorizontalAlignment="Left" Margin="10,111,0,0" VerticalAlignment="Top" Width="171" Height="26"
            Click="btnSacuvaj_Click" Background="LawnGreen" Cursor="Hand"/>

    </Grid>

</Window>

```

Slika 42 – XAML kod forme za dodavanje nove vrste proizvoda

```

<Window x:Class="Trafika.Forme.frmProizvodjac"
        xmlns:mas="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:Trafika.Forme"
        mc:Ignorable="d"
        Title="Proizvođač" Height="210.902" Width="457.895"
        Icon="C:\Users\defaultuser0\OneDrive\2. godina\Treci semestar\Poslovni informacioni sistemi\Vezbe\Trafika\Ikonice\Proizvodjac.png"
        ResizeMode="CanMinimize" WindowStartupLocation="CenterScreen" FontWeight="DemiBold">

    <Grid Background="LightSkyBlue">

        <TextBox x:Name="txtNazivProizvodjaca" HorizontalAlignment="Left" Height="26" Margin="187,35,-145,0" TextWrapping="Wrap" VerticalAlignment="Top"
                Width="250"/>
        <Label Content="Naziv proizvođača" HorizontalAlignment="Left" Margin="10,35,0,0" VerticalAlignment="Top" Width="160"/>
        <Button Content="Otkazi" HorizontalAlignment="Left" Margin="266,111,-145,0" VerticalAlignment="Top" Width="171" Height="26" Click="btnOtkaci_Click"
                Background="Red" Cursor="Hand"/>
        <Button Content="Sačuvaj" HorizontalAlignment="Left" Margin="10,111,0,0" VerticalAlignment="Top" Width="171" Height="26" Click="btnSacuvaj_Click"
                Background="LawnGreen" Cursor="Hand"/>

    </Grid>

</Window>

```

Slika 43 – XAML kod forme za dodavanje novog proizvođača

```

<Window x:Class="Trafika.Forme.frmDobavljac"
        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:Trafika.Forme"
        mc:Ignorable="d"
        Title="Dobavljač" Height="210.902" Width="457.895"
        Icon="C:\Users\defaultuser0\OneDrive\2. godina\Treci semestar\Poslovni informacioni sistemi\Vezbe\Trafika\Ikonice\Dobavljac.png"
        ResizeMode="CanMinimize" WindowStartupLocation="CenterScreen" FontWeight="DemiBold">

    <Grid Background="LightSkyBlue">

        <TextBox x:Name="txtNazivFirme" HorizontalAlignment="Left" Height="26" Margin="187,35,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
                Width="250"/>
        <Label Content="Naziv firme" HorizontalAlignment="Left" Margin="10,35,0,0" VerticalAlignment="Top" Width="160"/>
        <Button x:Name="btnOtkazi" Content="Otkazi" HorizontalAlignment="Left" Margin="266,111,-145,0" VerticalAlignment="Top" Width="171" Height="26"
                Click="btnOtkazi_Click" Background="Red" Cursor="Hand"/>
        <Button x:Name="btnSacuvaj" Content="Sačuvaj" HorizontalAlignment="Left" Margin="10,111,0,0" VerticalAlignment="Top" Width="171" Height="26"
                Click="btnSacuvaj_Click" Background="LawnGreen" Cursor="Hand"/>

    </Grid>

</Window>

```

Slika 44 – XAML kod forme za dodavanje novog dobavljača

Razlika se ponovo ogleda u *insert* upitu, smeštenom unutar metode *Sačuvaj*.

```
private void btnSacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();

        string insert = @"insert into tblKupac(BrojMobilnogTelefona)
                        values ('" + txtBrojMobilnogTelefona.Text + "')";

        SqlCommand cmd = new SqlCommand(insert, konekcija);
        cmd.ExecuteNonQuery();

        this.Close();
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrednosti nije validan.", "Greška!", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
}
```

Slika 45 – Sadržaj metode *btnSacuvaj_Click* za insert novog kupca

```
private void btnSacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();
        string insert = @"insert into tblVrstaProizvoda(NazivVrsteProizvoda)
                        values('" + txtNazivVrsteProizvoda.Text + "')";
        SqlCommand cmd = new SqlCommand(insert, konekcija);
        cmd.ExecuteNonQuery();
        this.Close();
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrednosti nije validan.", "Greška!", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
}
```

Slika 46 – Sadržaj metode *btnSacuvaj_Click* za insert nove vrste proizvoda

```

private void btnSacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();

        string insert = @"insert into tblProizvodjac(NazivProizvodjaca)
                        values('" + txtNazivProizvodjaca.Text + "')";

        SqlCommand cmd = new SqlCommand(insert, konekcija);
        cmd.ExecuteNonQuery();

        this.Close();
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrednosti nije validan.", "Greška!", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
}

```

Slika 47 – Sadržaj metode btnSacuvaj_Click za insert novog proizvođača

```

public void btnSacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();

        string insert = @"insert into tblDobavljac(NazivFirme)
                        values ('" + txtNazivFirme.Text + "')";

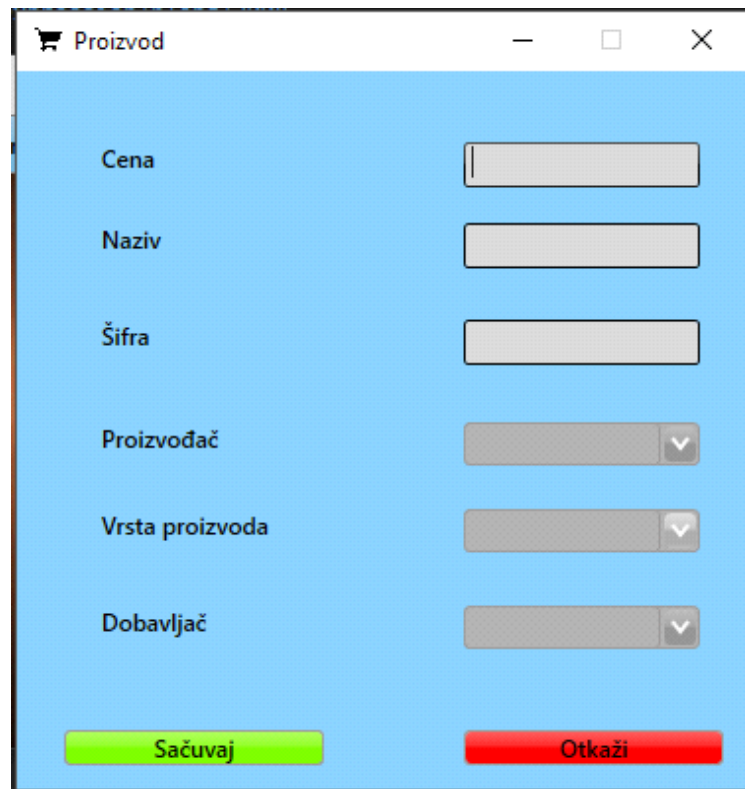
        SqlCommand cmd = new SqlCommand(insert, konekcija);
        cmd.ExecuteNonQuery();

        this.Close(); // zatvori prozor
    }
    catch(SqlException)
    {
        MessageBox.Show("Unos određenih vrednosti nije validan.", "Greška!", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
}

```

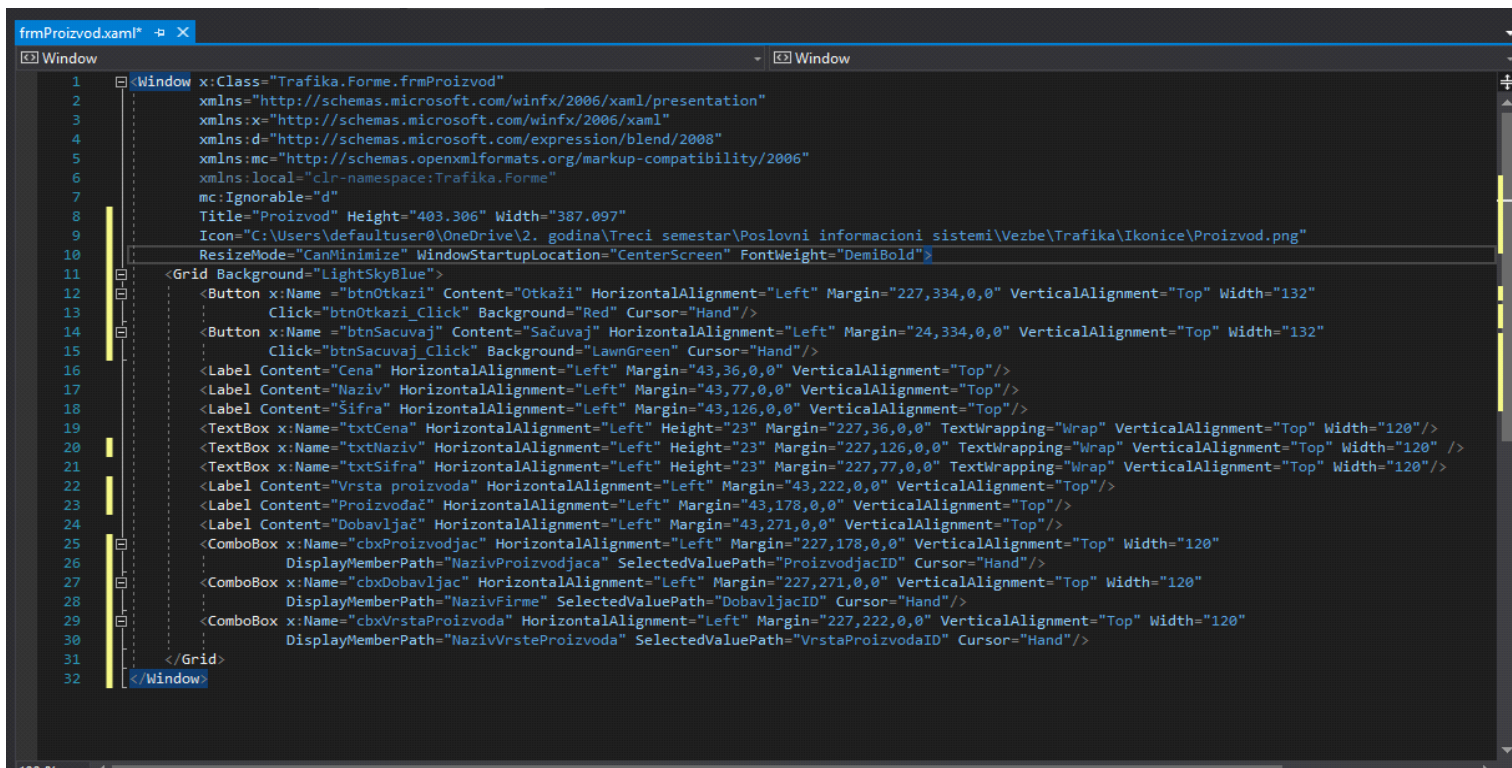
Slika 48 – Sadržaj metode btnSacuvaj_Click za insert novog dobavljača

Tabele *tblProizvod* i *tblFiskalniRacun* 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 to se najjednostavniji način postiže korišćenjem *ComboBox*-a. Vizuelni prikaz forme za dodavanje novog proizvoda prikazan je na *Slici 49*.



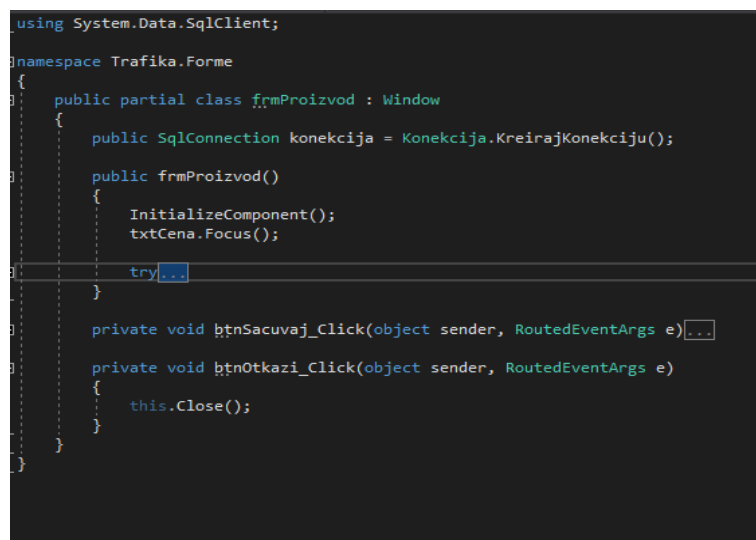
Slika 49 – Vizuelni prikaz forme za dodavanje novog proizvoda

XAML kod ove forme prikazan je na *Slici 50*. Prilikom kreiranja novog *ComboBox*-a, potrebno je da definišemo *DisplayMemberPath*, koji predstavlja obeležje povezane tabele koje će se prikazati u padajućoj listi kada korisnik klikne na taj *ComboBox*, kao i *SelectedValuePath* koji predstavlja obeležje primarnog ključa na osnovu kog će se uneta vrednost iz prikazanog *ComboBox*-a povezati sa svojim identifikacionim obeležjem i kao takva proslediti u bazu podataka.



Slika 50 – XAML kod forme za dodavanje novog proizvoda

CodeBehind formi koje u sebi sadrže *ComboBox*-ove razlikovaće se od ostalih. U nastavku sagledaćemo svrhu i sadržaj označenog *try* bloka u konstruktoru ove klase (Slika 51).



Slika 51 – Sadržaj CodeBehind-a forme frmProizvod

Try blok prikazan na *Slici 52*, smešten je unutar konstruktora ove klase s obzirom da se, pri instanciranju novog prozora ove forme, podaci koji će biti prikazani u *ComboBox*-ovima moraju negde privremeno uskladištiti. Za postizanje toga, koristi se objekat *DataTable* klase koji predstavlja kopiju tabele iz baze podataka (sa obeležjima koja se proslede unutar *select* upita), sačuvanu u memoriji aplikacije. Objekat klase *DataTable* se popunjava pomoću objekta klase *SqlDataAdapter* kojem prosleđujemo odgovarajući *select* upit kao i konekciju ka odgovarajućoj bazi podataka. Zatim se taj *DataTable* postavlja kao *ItemsSource* za odgovarajući *ComboBox*. Na samom kraju, u *finally* bloku, potrebno je obezbediti da se konekcija ka bazi bezbedno zatvori.

```
try
{
    konekcija.Open();
    //cbxProizvodjac
    string vratiProizvodjace = "select ProizvodjacID, NazivProizvodjaca from tblProizvodjac";
    DataTable dataTableProizvodjaci = new DataTable();
    SqlDataAdapter dataAdapterProizvodjaci = new SqlDataAdapter(vratiProizvodjace, konekcija);

    dataAdapterProizvodjaci.Fill(dataTableProizvodjaci);
    cbxProizvodjac.ItemsSource = dataTableProizvodjaci.DefaultView;

    //cbxVrstaProizvoda

    string vratiVrsteProizvoda = "select VrstaProizvodaID, NazivVrsteProizvoda from tblVrstaProizvoda";
    DataTable dataTableVrsteProizvoda = new DataTable();
    SqlDataAdapter dataAdapterVrsteProizvoda = new SqlDataAdapter(vratiVrsteProizvoda, konekcija);

    dataAdapterVrsteProizvoda.Fill(dataTableVrsteProizvoda);
    cbxVrstaProizvoda.ItemsSource = dataTableVrsteProizvoda.DefaultView;

    //cbxDobavljac

    string vratiDobavljace = "select DobavljacID, NazivFirme from tblDobavljac";
    DataTable dataTableDobavljaci = new DataTable();
    SqlDataAdapter dataAdapterDobavljaci = new SqlDataAdapter(vratiDobavljace, konekcija);

    dataAdapterDobavljaci.Fill(dataTableDobavljaci);
    cbxDobavljac.ItemsSource = dataTableDobavljaci.DefaultView;
}
finally
{
    if (konekcija != null)
        konekcija.Close();
}
```

Slika 52 – Sadržaj try - finally bloka unutar konstruktora forme frmProizvod

Sadržaj metode koja se aktivira na klik dugmeta ***Sačuvaj***, sličan je kao i u prethodnim formama s tim što se za razliku od tekstualnih polja, vrednost iz ComboBox-a kupi pomoću *property*-ja *SelectedValue* (Slika 53).

```
private void btnSacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();

        string insert = @"insert into tblProizvod(Cena, Sifra, Naziv, ProizvodjacID, VrstaProizvodaID, DobavljacID)
        values(" + txtCena.Text + ", " + txtSifra.Text + ", " + txtNaziv.Text + ", " +
        cbxProizvodjac.SelectedValue +
        ", " + cbxVrstaProizvoda.SelectedValue + ", " + cbxDobavljac.SelectedValue + ")";

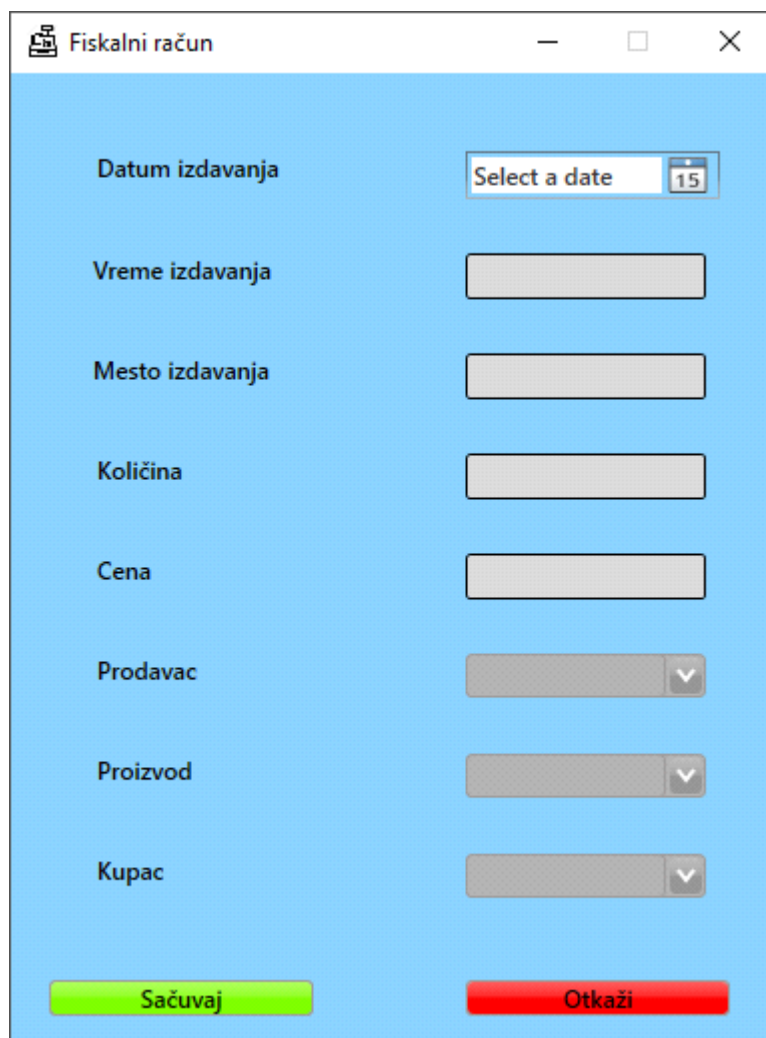
        SqlCommand cmd = new SqlCommand(insert, konekcija);

        cmd.ExecuteNonQuery();

        this.Close();
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrednosti nije validan.", "Greška!", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
}
```

Slika 53 – Sadržaj metode *btnSacuvaj_Click* za insert novog proizvoda

Poslednja forma koju je potrebno kreirati je forma za dodavanje novog *Fiskalnog računa* u bazu podataka, koja je veoma slična formi za dodavanje novog *Proizvoda*. Na *Slici 54*. prikazan je njen sadržaj koji se sastoji od po četiri *TextBox*-a i tri *ComboBox*a, i jednog *DatePicker*-a. XAML kod ove forme prikazan je na *Slici 55*.



The image shows a Windows application window titled "Fiskalni račun". The window has a standard Windows title bar with a minimize button, a maximize button, and a close button. The main content area has a light blue background. It contains several input fields and buttons. The fields are arranged in a list-like structure. The first field is "Datum izdavanja" with a date picker showing "15". The other fields are "Vreme izdavanja", "Mesto izdavanja", "Količina", "Cena", "Prodavac", "Proizvod", and "Kupac". At the bottom, there are two buttons: "Sačuvaj" (green) and "Otkazi" (red).

Slika 54 – Vizuelni prikaz forme za dodavanje novog fiskalnog računa

```

<Window x:Class="Trafika.Forme.frmFiskalniRacun"
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:Trafika.Forme"
mc:Ignorable="d"
Title="Fiskalni račun" Height="522.806" Width="398.097"
Icon="C:\Users\defaultuser0\OneDrive\2. godina\Treci semestar\Poslovni informacioni sistemi\Vezbe\Trafika\Ikonice\Fiskalni racun.png"
ResizeMode="CanMinimize" WindowStartupLocation="CenterScreen" FontWeight="DemiBold">
<Grid Background="LightSkyBlue" Margin="0,0,-6,-3">
<Label Content="Datum izdavanja" HorizontalAlignment="Left" Margin="43,39,0,0" VerticalAlignment="Top"/>
<Label Content="Vreme izdavanja" HorizontalAlignment="Left" Margin="41,90,0,0" VerticalAlignment="Top"/>
<Label Content="Mesto izdavanja" HorizontalAlignment="Left" Margin="41,140,0,0" VerticalAlignment="Top"/>
<TextBox x:Name="txtVremeIzdavanja" HorizontalAlignment="Left" Height="23" Margin="227,90,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"/>
<TextBox x:Name="txtMestoIzdavanja" HorizontalAlignment="Left" Height="23" Margin="227,140,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"/>
<TextBox x:Name="txtKolicina" HorizontalAlignment="Left" Height="23" Margin="227,190,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"/>
<Button x:Name="btnOtkazi" Content="Otkazi" HorizontalAlignment="Left" Margin="227,453,0,0" VerticalAlignment="Top" Width="132"
Click="btnOtkazi_Click" Background="Red" Cursor="Hand"/>
<Button x:Name="btnSacuvaj" Content="Sačuvaj" HorizontalAlignment="Left" Margin="19,453,0,0" VerticalAlignment="Top" Width="132"
Click="btnSacuvaj_Click" Background="LawnGreen" Cursor="Hand"/>
<DatePicker x:Name="DatumIzdavanja" Margin="227,39,34,0" VerticalAlignment="Top"/>
<Label Content="Proizvod" HorizontalAlignment="Left" Margin="43,340,0,0" VerticalAlignment="Top"/>
<Label Content="Kupac" HorizontalAlignment="Left" Margin="43,390,0,0" VerticalAlignment="Top"/>
<Label Content="Količina" HorizontalAlignment="Left" Margin="43,190,0,0" VerticalAlignment="Top"/>
<ComboBox x:Name="cbxKupac" HorizontalAlignment="Left" Margin="227,390,0,0" VerticalAlignment="Top" Width="120"
DisplayMemberPath="BrojMobilnogTelefona" SelectedValuePath="KupacID" Cursor="Hand"/>
<ComboBox x:Name="cbxProdavac" HorizontalAlignment="Left" Margin="227,290,0,0" VerticalAlignment="Top" Width="120"
DisplayMemberPath="Prodavac" SelectedValuePath="ProdavacID" Cursor="Hand" RenderTransformOrigin="0.558,0"/>
<ComboBox x:Name="cbxProizvod" HorizontalAlignment="Left" Margin="227,340,0,0" VerticalAlignment="Top" Width="120" DisplayMemberPath="Proizvod"
SelectedValuePath="ProizvodID" Cursor="Hand" RenderTransformOrigin="0.558,-0.273"/>
<Label Content="Prodavac" HorizontalAlignment="Left" Margin="43,290,0,0" VerticalAlignment="Top" Width="69"/>
<Label Content="Cena" HorizontalAlignment="Left" Margin="43,240,0,0" VerticalAlignment="Top"/>
<TextBox x:Name="txtCena" HorizontalAlignment="Left" Height="23" Margin="227,240,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
TextChanged="txtIznos_TextChanged"/>
</Grid>

```

Slika 55 – XAML kod forme za dodavanje novo fiskalnog računa

Konstruktor ove forme, prikazan na Slici 56, u sebi sadrži tri *DataTable*-a, koji snabdevaju podacima tri *ComboBox*-a.

```

public frmFiskalniRacun()
{
    InitializeComponent();
    DatumIzdavanja.Focus();

    try
    {
        konekcija.Open();

        //cbxProdavac
        string vratiProdavce = "select ProdavacID, Username + ' ' + Password as Prodavac from tblProdavac";
        DataTable dataTableProdavci = new DataTable();
        SqlDataAdapter dataAdapterProdavci = new SqlDataAdapter(vratiProdavce, konekcija);
        dataAdapterProdavci.Fill(dataTableProdavci);
        cbxProdavac.ItemsSource = dataTableProdavci.DefaultView;

        //cbxKupac
        string vratiKupce = "select KupacID, BrojMobilnogTelefona from tblKupac";
        DataTable dataTableKupci = new DataTable();
        SqlDataAdapter dataAdapterKupci = new SqlDataAdapter(vratiKupce, konekcija);
        dataAdapterKupci.Fill(dataTableKupci);
        cbxKupac.ItemsSource = dataTableKupci.DefaultView;

        //cbxProizvod
        string vratiProizvode = "select ProizvodID, Naziv + ' ' + ltrim(str(Cena)) as 'Proizvod' from tblProizvod";
        DataTable dataTableProizvodi = new DataTable();
        SqlDataAdapter dataAdapterProizvodi = new SqlDataAdapter(vratiProizvode, konekcija);
        dataAdapterProizvodi.Fill(dataTableProizvodi);
        cbxProizvod.ItemsSource = dataTableProizvodi.DefaultView;
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
}

```

Slika 56 – Konstruktor forme za dodavanje novog fiskalnog računa

Metoda za čuvanje zapisa, odnosno njegovo prosleđivanje u bazu podataka, funkcioniše na identitičan način kao i kod prethodnih formi, s'tim što se vrednost izabrana vrednost datuma iz komande *DatePicker* ubacuje u select komandu pomoću *property*-ja *SelectedDate* (Slika 57).

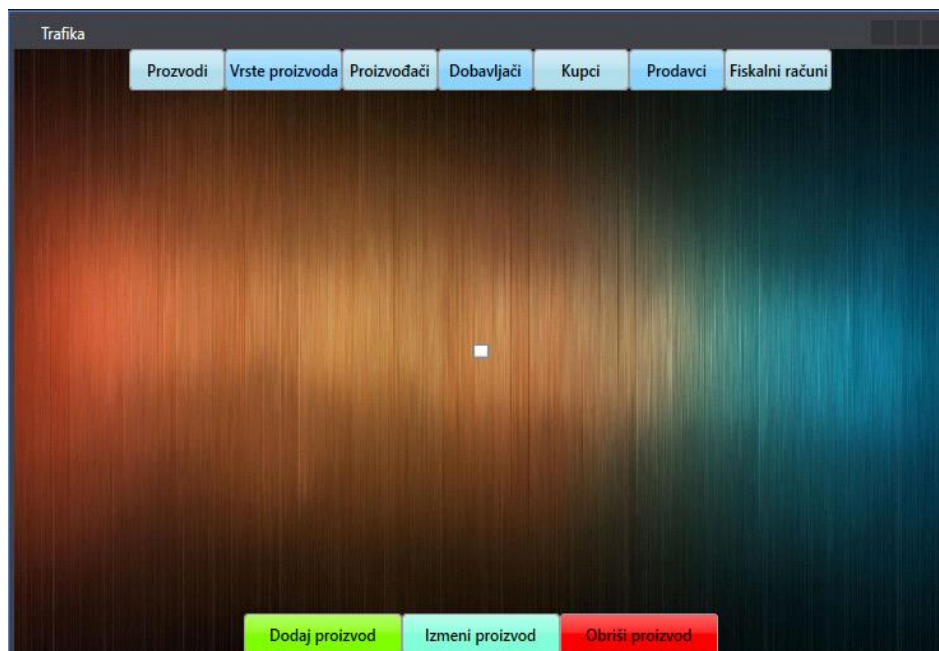
```
private void btnSacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();

        string insert = @"insert into tblFiskalniRacun(DatumIzdavanja, VremeIzdavanja, MestoIzdavanja, Iznos, Kolicina, ProdavacID, ProizvodID, KupacID)
        values('" + DatumIzdavanja.SelectedDate + "', '" + txtVremeIzdavanja.Text + "', '" + txtMestoIzdavanja.Text + "', " +
        (int.Parse(txtCena.Text) * int.Parse(txtKolicina.Text)).ToString() + ", " + txtKolicina.Text + ", " + cbxProdavac.SelectedValue +
        ", " + cbxProizvod.SelectedValue + ", " + cbxKupac.SelectedValue + "));";

        SqlCommand cmd = new SqlCommand(insert, konekcija);
        cmd.ExecuteNonQuery();
        this.Close();
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrednosti nije validan.", "Greška!", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
}
```

Slika 57 – Sadržaj metode *btnSacuvaj_Click* za dodavanje novog fiskalnog računa

Nakon kreiranja formi za svaku od tabela, potrebno je nešto slično napraviti i za glavni (*Main*) prozor. Main prozor predstavlja prvi, inicijalni, kontakt korisnika sa aplikacijom i ujedno i bazni 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 7 kreiranih tabela u bazi, čijim će se klikom, na sredini forme iščitavati ti podaci. Takođe, klikom na dugme određene tabele, potrebno je da se korisniku obezbede dugmići za *Dodavanje*, *Izmenu* i *Brisanje* prikazanog sadržaja. Dodavanjem odgovarajućih funkcionalnosti tim dugmićima u okviru glavne forme, obezbediće se izvršavanje osnovnih CRUD operacija, što je i bio cilj ovog zadatka. Prikaz izgleda ove forme vidimo na *Slici* 58.



Slika 58 – Vizuelni prikaz glavnog prozora

U nastavku, bliže ćemo sagledati XAML kod ove forme.

Na samom početku, potrebno je definisati centralni *DataGrid* u okviru kojeg će biti ispisani učitani podaci iz baze. Podešavanja ovog grid-a takođe su prikazana u okviru *Slike 59*.

```
<Window x:Class="Trafika.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:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        xmlns:local="clr-namespace:Trafika"
        mc:Ignorable="d"
        Title="Trafika" Height="360" Width="630"
        Icon="C:\Users\defaultuser0\OneDrive\2. godina\Treci semestar\Poslovni informacioni sistemi\Vezbe\Trafika\Ikonice\Trafika.ico"
        WindowStartupLocation="CenterScreen" FontWeight="DemiBold">

    <Grid>

        <Grid.Background>

            <ImageBrush ImageSource=
                "C:\Users\defaultuser0\OneDrive\2. godina\Treci semestar\Poslovni informacioni sistemi\Vezbe\Trafika\Ikonice\Pozadina.png"
                Stretch="UniformToFill"/>

        </Grid.Background>

        <DataGrid x:Name="dataGridCentralni" HorizontalAlignment="Center" Height="auto" VerticalAlignment="Center" Width="auto"
            AlternatingRowBackground="LightBlue" IsReadOnly="True" CanUserAddRows="False" SelectionUnit="FullRow" SelectionMode="Single"
            CanUserResizeRows="False" CanUserDeleteRows="False" CanUserReorderColumns="False" CanUserResizeColumns="False"
            CanUserSortColumns="False"/>

    </Grid>
```

Slika 59 – XAML kod centralnog grid-a

Kako bi elementi unutar ovog prozora bili uredno raspoređeni, koristiće se poseban panel za njihovo smeštanje – *StackPanel*. U okviru gornjeg *StackPanel*-a (Slika 60.), biće smešteni dugmići za izlistavanje tabela na centralnom gridu. Potrebno je podesiti poravnanja kao što je prikazano na slici, a zatim izgenerisati metode koje će sprovoditi odgovarajuće akcije klikom na to dugme, a čija će implementacija biti dodata nešto kasnije.

```
<StackPanel HorizontalAlignment="Center" Height="30" VerticalAlignment="Top" Width="auto" Orientation="Horizontal">

    <Button x:Name="btnProizvodi" Content="Proizvodi" HorizontalAlignment="Left" Width="75" Height="30" VerticalAlignment="Top"
        Click="btnProizvodi_Click" Background="LightBlue" Cursor="Hand"/>
    <Button x:Name="btnVrsteProizvoda" Content="Vrste proizvoda" HorizontalAlignment="Left" VerticalAlignment="Bottom" Width="92"
        Height="30" Click="btnVrsteProizvoda_Click" Background="LightSkyBlue" Cursor="Hand"/>
    <Button x:Name="btnProizvodjaci" Content="Proizvođači" HorizontalAlignment="Left" VerticalAlignment="Top" Width="75" Height="30"
        Click="btnProizvodjaci_Click" Background="LightBlue" Cursor="Hand"/>
    <Button x:Name="btnDobavljac" Content="Dobavljači" HorizontalAlignment="Left" VerticalAlignment="Top" Width="75" Height="30"
        Click="btnDobavljac_Click" Background="LightSkyBlue" Cursor="Hand"/>
    <Button x:Name="btnKupci" Content="Kupci" HorizontalAlignment="Left" VerticalAlignment="Top" Width="75" Height="30" Click="btnKupci_Click"
        Background="LightBlue" Cursor="Hand"/>
    <Button x:Name="btnProdavci" Content="Prodavci" HorizontalAlignment="Left" VerticalAlignment="Top" Width="75" Height="30"
        Click="btnProdavci_Click" Background="LightSkyBlue" Cursor="Hand"/>
    <Button x:Name="btnFiskalniRacuni" Content="Fiskalni računi" HorizontalAlignment="Left" VerticalAlignment="Top" Width="84" Height="30"
        Click="btnFiskalniRacuni_Click" Background="LightBlue" Cursor="Hand"/>

</StackPanel>
```

Slika 60 – Gornji *StackPanel* glavnog prozora


```

<StackPanel HorizontalAlignment="Center" Height="30" VerticalAlignment="Bottom" Width="372" Orientation="Horizontal" Cursor="Hand">
  <Button x:Name="btnDodajProizvod" Content="Dodaj proizvod" Width="124" Click="btnDodajProizvod_Click" Background="LawnGreen" Cursor="Hand"/>
  <Button x:Name="btnIzmeniProizvod" Content="Izmeni proizvod" Width="124" Click="btnIzmeniProizvod_Click" Background="Aquamarine" Cursor="Hand"/>
  <Button x:Name="btnObrisiProizvod" Content="Obriši proizvod" Width="124" Click="btnObrisiProizvod_Click" Background="Red" Cursor="Hand"/>
  <Button x:Name="btnDodajKupca" Content="Dodaj kupca" Width="124" Click="btnDodajKupca_Click" Background="LawnGreen" Cursor="Hand"/>
  <Button x:Name="btnIzmeniKupca" Content="Izmeni kupca" Width="124" Click="btnIzmeniKupca_Click" Background="Aquamarine" Cursor="Hand"/>
  <Button x:Name="btnObrisiKupca" Content="Obriši kupca" Width="124" Click="btnObrisiKupca_Click" Background="Red" Cursor="Hand"/>
  <Button x:Name="btnDodajDobavljacka" Content="Dodaj dobavljača" Width="124" Click="btnDodajDobavljacka_Click" Background="LawnGreen"
    Cursor="Hand"/>
  <Button x:Name="btnIzmeniDobavljacka" Content="Izmeni dobavljača" Width="124" Click="btnIzmeniDobavljacka_Click" Background="Aquamarine"
    Cursor="Hand"/>
  <Button x:Name="btnObrisiDobavljacka" Content="Obriši dobavljača" Width="124" Click="btnObrisiDobavljacka_Click" Background="Red"
    Cursor="Hand"/>
  <Button x:Name="btnDodajProdavca" Content="Dodaj prodavca" Width="124" Click="btnDodajProdavca_Click" Background="LawnGreen" Cursor="Hand"/>
  <Button x:Name="btnIzmeniProdavca" Content="Izmeni prodavca" Width="124" Click="btnIzmeniProdavca_Click" Background="Aquamarine" Cursor="Hand"/>
  <Button x:Name="btnObrisiProdavca" Content="Obriši prodavca" Width="124" Click="btnObrisiProdavca_Click" Background="Red" Cursor="Hand"/>
  <Button x:Name="btnDodajFiskalniRacun" Content="Dodaj fiskalni račun" Width="124" Click="btnDodajFiskalniRacun_Click" Background="LawnGreen"
    Cursor="Hand"/>
  <Button x:Name="btnIzmeniFiskalniRacun" Content="Izmeni fiskalni račun" Width="124" Click="btnIzmeniFiskalniRacun_Click"
    Background="Aquamarine" Cursor="Hand"/>
  <Button x:Name="btnObrisiFiskalniRacun" Content="Obriši fiskalni račun" Width="124" Click="btnObrisiFiskalniRacun_Click" Background="Red"
    Cursor="Hand"/>
  <Button x:Name="btnDodajVrstuProizvoda" Content="Dodaj vrstu proizvoda" Width="124" Click="btnDodajVrstuProizvoda_Click" Background="LawnGreen"
    Cursor="Hand"/>
  <Button x:Name="btnIzmeniVrstuProizvoda" Content="Izmeni vrstu proizvoda" Width="124" Click="btnIzmeniVrstuProizvoda_Click"
    Background="Aquamarine" Cursor="Hand"/>
  <Button x:Name="btnObrisiVrstuProizvoda" Content="Obriši vrstu proizvoda" Width="124" Click="btnObrisiVrstuProizvoda_Click" Background="Red"
    Cursor="Hand"/>
  <Button x:Name="btnDodajProizvodjaca" Content="Dodaj proizvođača" Width="124" Click="btnDodajProizvodjaca_Click" Background="LawnGreen"
    Cursor="Hand"/>
  <Button x:Name="btnIzmeniProizvodjaca" Content="Izmeni proizvođača" Width="124" Click="btnIzmeniProizvodjaca_Click" Background="Aquamarine"
    Cursor="Hand"/>
  <Button x:Name="btnObrisiProizvodjaca" Content="Obriši proizvođača" Width="124" Click="btnObrisiProizvodjaca_Click" Background="Red"
    Cursor="Hand"/>
</StackPanel>
</Grid>

```

Slika 61 – Donji StackPanel glavnog prozora

Na Slici 61, vidimo prikaz donjeg *StackPanel*-a u okviru kog su kreirani dugmići za *Dodavanje*, *Izmenu* i *Brisanje* zapisa iz svake od posebnih tabela, i za svaki od dugmića kreirana je metoda koja će obavljati neku funkcionalnost klikom na to dugme. Prilikom kreiranja ovog *StackPanela*, potrebno je obratiti pažnju na širinu i visinu *Panel*-a ali i svakog posebnog dugmeta. U okviru širine *StackPanel*-a, potrebno je da se nađu tačno tri dugmeta za odgovarajuću formu. Takođe, visina *StackPanel*-a treba da bude jednaka visini dugmeta. Na ovaj način, dugmići će biti preklapljeni jedni preko drugih i kasnije će se prikazivati u zavisnosti od toga koje nam je dugme potrebno. Naravno, ovo je samo jedno od dizajnerskih rešenja za ovakav raspored dugmića i nije nužno pridržavati se istog prilikom kreiranja i rasporeda ovih komponenti u XAML kodu.

U nastavku će biti prikazan *CodeBehind* sadržina glavnog prozora gde je smešten najveći deo poslovne logike.

Na samom početku, potrebno je importovati odgovarajuće imenske prostore kao što je prikazano na *Slici 62*.

```
using System;  
using System.Data;  
using System.Data.SqlClient;  
using System.Windows;  
using System.Windows.Controls;  
using Trafika.Forme;
```

Slika 62 – Imenski prostori za glavni prozor

Za početak, sadržina glavnog prozora trebalo bi da izgleda kao na *Slici 63*. Sadržaće instancu konekcije kao bazi podataka, metodu za popunjavanje početnog grid-a pri pokretanju prozora, konstruktor za kreiranje novog prozora kao izgenerisane metode iz XAML koda, putem kojih će se, klikom na odgovarajuće dugme, obavljati određene funkcionalnosti.

```
public partial class MainWindow : Window  
{  
    private void PocetniDataGrid(DataGrid grid)...  
    public MainWindow()...  
    private void btnProizvodi_Click(object sender, RoutedEventArgs e)...  
    private void btnVrsteProizvoda_Click(object sender, RoutedEventArgs e)...  
    private void btnProizvodjaci_Click(object sender, RoutedEventArgs e)...  
  
    private void btnDobavljacki_Click(object sender, RoutedEventArgs e)...  
    private void btnKupci_Click(object sender, RoutedEventArgs e)...  
    private void btnProdavci_Click(object sender, RoutedEventArgs e)...  
    private void btnFiskalniRacuni_Click(object sender, RoutedEventArgs e)...  
    // Akcija dodaj  
    private void btnDodajProizvod_Click(object sender, RoutedEventArgs e)...  
    private void btnDodajVrstuProizvoda_Click(object sender, RoutedEventArgs e)...  
    private void btnDodajProizvodjaca_Click(object sender, RoutedEventArgs e)...  
    private void btnDodajDobavljacka_Click(object sender, RoutedEventArgs e)...  
    private void btnDodajKupca_Click(object sender, RoutedEventArgs e)...  
    private void btnDodajProdavca_Click(object sender, RoutedEventArgs e)...  
    private void btnDodajFiskalniRacun_Click(object sender, RoutedEventArgs e)...
```



```

private void btnIzmeniProizvod_Click(object sender, RoutedEventArgs e)
private void btnIzmeniVrstuProizvoda_Click(object sender, RoutedEventArgs e)
private void btnIzmeniProizvodjaca_Click(object sender, RoutedEventArgs e)
private void btnIzmeniDobavljacka_Click(object sender, RoutedEventArgs e)
private void btnIzmeniKupca_Click(object sender, RoutedEventArgs e)
private void btnIzmeniProdavca_Click(object sender, RoutedEventArgs e)
private void btnIzmeniFiskalniRacun_Click(object sender, RoutedEventArgs e)
private void btnObrisiProizvod_Click(object sender, RoutedEventArgs e)
private void btnObrisiVrstuProizvoda_Click(object sender, RoutedEventArgs e)
private void btnObrisiProizvodjaca_Click(object sender, RoutedEventArgs e)
private void btnObrisiDobavljacka_Click(object sender, RoutedEventArgs e)
private void btnObrisiKupca_Click(object sender, RoutedEventArgs e)
private void btnObrisiProdavca_Click(object sender, RoutedEventArgs e)
private void btnObrisiFiskalniRacun_Click(object sender, RoutedEventArgs e)
}

```

Slika 63 – Sadržaj CodeBehind-a glavnog prozora

Nakon instanciranja konekcije, sledeća stvar je kreiranje metode koja će popunjavati početni *DataGrid* pri kreiranju novog prozora. U ovom primeru, tabela *Proizvod* će biti podrazumevana tabela, odnosno tabela čiji će se prikaz generisati pri pokretanju ovog prozora. Na osnovu toga, potrebno je kreirati odgovarajući *Select* upit koji će izlistavati, korisniku bitne informacije o svakom proizvodu. U ovom slučaju, nakon otvaranja konekcije, prikazani upit se kreira na osnovu povezivanja 4 tabele iz baze podataka (*tblProizvod*, *tblProizvodjac*, *tblVrstaProizvoda* i *tblDobavljac*). Zatim, na isti način kao kod popunjavanja *ComboBox*-ova pri instanciranju prozora neke od formi, ovi podaci se smeštaju u odgovarajući *DataSet* kojeg zatim popunjava *SqlDataAdapter*. Razlika je u tome što, ove podatke, prosledjujemo kao *ItemsSource* gridu (parametru ove metode), umesto *ComboBox*-u kao u prethodno objašnjenom primeru (Slika 64).

```
private void PocetniDataGrid(DataGrid grid)
{
    try
    {
        string upit = @"select ProizvodID as ID, Cena, Sifra as Šifra, Naziv from tblProizvod
            inner join tblProizvodjac on tblProizvod.ProizvodjacID = tblProizvodjac.ProizvodjacID
            inner join tblVrstaProizvoda on tblProizvod.VrstaProizvodaID = tblVrstaProizvoda.VrstaProizvodaID
            inner join tblDobavljac on tblProizvod.DobavljacID = tblDobavljac.DobavljacID";

        SqlDataAdapter dataAdapter = new SqlDataAdapter(upit, konekcija);
        DataTable dataTable = new DataTable("Proizvod");
        dataAdapter.Fill(dataTable);
        dataGridCentralni.ItemsSource = dataTable.DefaultView;
    }
    catch (Exception exception)
    {
        System.Diagnostics.Debug.WriteLine(exception.Message.ToString());
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
}
```

Slika 64 – Sadržaj metode *PocetniDataGrid*

Sadržaj konstruktora ovog prozora je prilično jednostavan. Prilikom instanciranja novog prozora, pomoću prethodno kreirane metode, popunjava se *CentralniGrid* (Slika 65).

```
public MainWindow()
{
    InitializeComponent();
    PocetniDataGrid(dataGridCentralni);
}
```

Slika 65 – Konstruktor glavnog prozora

Sledi pregled svake od posebnih metoda vezanih za klik na dugme koje služi za ispis podataka iz tabele na centralnom gridu. Prva, stvar koju ova metoda treba da obezbedi jeste da se, pored ispisa podataka iz baze, omogući prikaz dugmića za *Dodavanje*, *Izmenu* i *Brisanje* podataka. Tako nešto postiže se pomoću komand *Visibility*, na način prikazan na *Slici 66*. Za ispis podataka o proizvodima, koristiće se prethodno kreirana metoda *PocetniDataGrid* kojoj se kao argument prosleđuje kreirani *CentralniGrid*.

```
private void btnProizvodi_Click(object sender, RoutedEventArgs e)
{
    btnDodajProizvod.Visibility = Visibility.Visible;

    btnDodajProdavca.Visibility = Visibility.Collapsed;
    btnDodajKupca.Visibility = Visibility.Collapsed;
    btnDodajDobavljacka.Visibility = Visibility.Collapsed;
    btnDodajVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnDodajFiskalniRacun.Visibility = Visibility.Collapsed;
    btnDodajProizvodjaca.Visibility = Visibility.Collapsed;

    btnIzmeniProizvod.Visibility = Visibility.Visible;

    btnIzmeniProdavca.Visibility = Visibility.Collapsed;
    btnIzmeniKupca.Visibility = Visibility.Collapsed;
    btnIzmeniDobavljacka.Visibility = Visibility.Collapsed;
    btnIzmeniVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnIzmeniFiskalniRacun.Visibility = Visibility.Collapsed;
    btnIzmeniProizvodjaca.Visibility = Visibility.Collapsed;

    btnObrisiProizvod.Visibility = Visibility.Visible;

    btnObrisiProdavca.Visibility = Visibility.Collapsed;
    btnObrisiKupca.Visibility = Visibility.Collapsed;
    btnObrisiDobavljacka.Visibility = Visibility.Collapsed;
    btnObrisiVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnObrisiFiskalniRacun.Visibility = Visibility.Collapsed;
    btnObrisiProizvodjaca.Visibility = Visibility.Collapsed;

    PocetniDataGrid(dataGridCentralni);
}
```

Slika 66 – Sadržaj metode btnProizvodi_Click

Ispis podataka iz preostalih 6 tabela, vrši se na gotovo identičan način kao za tabelu *Proizvod*, sa tom razlikom da će se *Select* upit pisati direktno u metodi. Na ostalim slikama, prikazani su ispisi i za ostalih 6 tabela: Vrste proizvoda (*Slika 67*), Proizvođači (*Slika 68.*), Dobavljači (*Slika 69*), Kupci (*Slika 70*), Prodavci (*Slika 71*) i Fiskalni Računi (*Slika 72*).

```
private void btnVrsteProizvoda_Click(object sender, RoutedEventArgs e)
{
    btnDodajVrstuProizvoda.Visibility = Visibility.Visible;

    btnDodajProdavca.Visibility = Visibility.Collapsed;
    btnDodajKupca.Visibility = Visibility.Collapsed;
    btnDodajDobavljacka.Visibility = Visibility.Collapsed;
    btnDodajProizvod.Visibility = Visibility.Collapsed;
    btnDodajFiskalniRacun.Visibility = Visibility.Collapsed;
    btnDodajProizvodjaca.Visibility = Visibility.Collapsed;

    btnIzmeniVrstuProizvoda.Visibility = Visibility.Visible;

    btnIzmeniProdavca.Visibility = Visibility.Collapsed;
    btnIzmeniKupca.Visibility = Visibility.Collapsed;
    btnIzmeniDobavljacka.Visibility = Visibility.Collapsed;
    btnIzmeniProizvod.Visibility = Visibility.Collapsed;
    btnIzmeniFiskalniRacun.Visibility = Visibility.Collapsed;
    btnIzmeniProizvodjaca.Visibility = Visibility.Collapsed;

    btnObrisiVrstuProizvoda.Visibility = Visibility.Visible;

    btnObrisiProdavca.Visibility = Visibility.Collapsed;
    btnObrisiKupca.Visibility = Visibility.Collapsed;
    btnObrisiDobavljacka.Visibility = Visibility.Collapsed;
    btnObrisiProizvod.Visibility = Visibility.Collapsed;
    btnObrisiFiskalniRacun.Visibility = Visibility.Collapsed;
    btnObrisiProizvodjaca.Visibility = Visibility.Collapsed;
}
```

```
try
{
    konekcija.Open();

    string upit =
        @"select VrstProizvodaID as ID, NazivVrsteProizvoda as 'Naziv vrste proizvoda' from tblVrstaProizvoda";
    SqlDataAdapter dataAdapter = new SqlDataAdapter(upit, konekcija);
    DataTable dataTable = new DataTable("VrstaProizvoda");

    dataAdapter.Fill(dataTable);

    dataGridCentralni.ItemsSource = dataTable.DefaultView;
}
catch (Exception exception)
{
    System.Diagnostics.Debug.WriteLine(exception.Message.ToString());
}
finally
{
    if (konekcija != null)
        konekcija.Close();
}
}
```

Slika 67 - Sadržaj metode *btnVrsteProizvoda_Click*

```

private void btnProizvodjaci_Click(object sender, RoutedEventArgs e)
{
    btnDodajProizvodjaca.Visibility = Visibility.Visible;

    btnDodajProdavca.Visibility = Visibility.Collapsed;
    btnDodajKupca.Visibility = Visibility.Collapsed;
    btnDodajDobavljacka.Visibility = Visibility.Collapsed;
    btnDodajVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnDodajFiskalniRacun.Visibility = Visibility.Collapsed;
    btnDodajProizvod.Visibility = Visibility.Collapsed;

    btnIzmeniProizvodjaca.Visibility = Visibility.Visible;

    btnIzmeniProdavca.Visibility = Visibility.Collapsed;
    btnIzmeniKupca.Visibility = Visibility.Collapsed;
    btnIzmeniDobavljacka.Visibility = Visibility.Collapsed;
    btnIzmeniVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnIzmeniFiskalniRacun.Visibility = Visibility.Collapsed;
    btnIzmeniProizvod.Visibility = Visibility.Collapsed;

    btnObrisiProizvodjaca.Visibility = Visibility.Visible;

    btnObrisiProdavca.Visibility = Visibility.Collapsed;
    btnObrisiKupca.Visibility = Visibility.Collapsed;
    btnObrisiDobavljacka.Visibility = Visibility.Collapsed;
    btnObrisiVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnObrisiFiskalniRacun.Visibility = Visibility.Collapsed;
    btnObrisiProizvod.Visibility = Visibility.Collapsed;
}

```

```

try
{
    konekcija.Open();

    string upit = @"select ProizvodjacID as ID, NazivProizvodjaca as 'Naziv proizvođača' from tblProizvodjac";
    SqlDataAdapter dataAdapter = new SqlDataAdapter(upit, konekcija);
    DataTable dataTable = new DataTable("Proizvodjac");

    dataAdapter.Fill(dataTable);

    dataGridCentralni.ItemsSource = dataTable.DefaultView;
}
catch (Exception exception)
{
    System.Diagnostics.Debug.WriteLine(exception.Message.ToString());
}
finally
{
    if (konekcija != null)
        konekcija.Close();
}
}

```

Slika 68 - Sadržaj metode btnProizvodjaci_Click

```

private void btnDobavljaci_Click(object sender, RoutedEventArgs e)
{
    btnDodajDobavljaca.Visibility = Visibility.Visible;

    btnDodajProdavca.Visibility = Visibility.Collapsed;
    btnDodajKupca.Visibility = Visibility.Collapsed;
    btnDodajProizvod.Visibility = Visibility.Collapsed;
    btnDodajVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnDodajFiskalniRacun.Visibility = Visibility.Collapsed;
    btnDodajProizvodjaca.Visibility = Visibility.Collapsed;

    btnIzmeniDobavljaca.Visibility = Visibility.Visible;

    btnIzmeniProdavca.Visibility = Visibility.Collapsed;
    btnIzmeniKupca.Visibility = Visibility.Collapsed;
    btnIzmeniProizvod.Visibility = Visibility.Collapsed;
    btnIzmeniVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnIzmeniFiskalniRacun.Visibility = Visibility.Collapsed;
    btnIzmeniProizvodjaca.Visibility = Visibility.Collapsed;

    btnObrisiDobavljaca.Visibility = Visibility.Visible;

    btnObrisiProdavca.Visibility = Visibility.Collapsed;
    btnObrisiKupca.Visibility = Visibility.Collapsed;
    btnObrisiProizvod.Visibility = Visibility.Collapsed;
    btnObrisiVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnObrisiFiskalniRacun.Visibility = Visibility.Collapsed;
    btnObrisiProizvodjaca.Visibility = Visibility.Collapsed;
}

```

```

try
{
    konekcija.Open();

    string upit = @"select DobavljacID as ID, NazivFirme as 'Naziv firme' from tblDobavljac";
    SqlDataAdapter dataAdapter = new SqlDataAdapter(upit, konekcija);
    DataTable dataTable = new DataTable("Dobavljac");

    dataAdapter.Fill(dataTable);

    dataGridCentralni.ItemsSource = dataTable.DefaultView;
}
catch (Exception exception)
{
    System.Diagnostics.Debug.WriteLine(exception.Message.ToString());
}
finally
{
    if (konekcija != null)
        konekcija.Close();
}
}

```

Slika 69 - Sadržaj metode btnDobavljaci_Click

```

private void btnKupci_Click(object sender, RoutedEventArgs e)
{
    btnDodajKupca.Visibility = Visibility.Visible;

    btnDodajProdavca.Visibility = Visibility.Collapsed;
    btnDodajProizvod.Visibility = Visibility.Collapsed;
    btnDodajDobavljacka.Visibility = Visibility.Collapsed;
    btnDodajVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnDodajFiskalniRacun.Visibility = Visibility.Collapsed;
    btnDodajProizvodjaca.Visibility = Visibility.Collapsed;

    btnIzmeniKupca.Visibility = Visibility.Visible;

    btnIzmeniProdavca.Visibility = Visibility.Collapsed;
    btnIzmeniProizvod.Visibility = Visibility.Collapsed;
    btnIzmeniDobavljacka.Visibility = Visibility.Collapsed;
    btnIzmeniVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnIzmeniFiskalniRacun.Visibility = Visibility.Collapsed;
    btnIzmeniProizvodjaca.Visibility = Visibility.Collapsed;

    btnObrisiKupca.Visibility = Visibility.Visible;

    btnObrisiProdavca.Visibility = Visibility.Collapsed;
    btnObrisiProizvod.Visibility = Visibility.Collapsed;
    btnObrisiDobavljacka.Visibility = Visibility.Collapsed;
    btnObrisiVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnObrisiFiskalniRacun.Visibility = Visibility.Collapsed;
    btnObrisiProizvodjaca.Visibility = Visibility.Collapsed;
}

```

```

try
{
    konekcija.Open();

    string upit = @"select KupacID as ID, BrojMobilnogTelefona as 'Broj mobilnog telefona' from tblKupac";
    SqlDataAdapter dataAdapter = new SqlDataAdapter(upit, konekcija);
    DataTable dataTable = new DataTable("Kupac");

    dataAdapter.Fill(dataTable);

    dataGridCentralni.ItemsSource = dataTable.DefaultView;
}
catch (Exception exception)
{
    System.Diagnostics.Debug.WriteLine(exception.Message.ToString());
}
finally
{
    if (konekcija != null)
        konekcija.Close();
}
}

```

Slika 70 - Sadržaj metode btnKupci_Click

```

private void btnProdavci_Click(object sender, RoutedEventArgs e)
{
    btnDodajProdavca.Visibility = Visibility.Visible;

    btnDodajProizvod.Visibility = Visibility.Collapsed;
    btnDodajKupca.Visibility = Visibility.Collapsed;
    btnDodajDobavljacka.Visibility = Visibility.Collapsed;
    btnDodajVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnDodajFiskalniRacun.Visibility = Visibility.Collapsed;
    btnDodajProizvodjaca.Visibility = Visibility.Collapsed;

    btnIzmeniProdavca.Visibility = Visibility.Visible;

    btnIzmeniProizvod.Visibility = Visibility.Collapsed;
    btnIzmeniKupca.Visibility = Visibility.Collapsed;
    btnIzmeniDobavljacka.Visibility = Visibility.Collapsed;
    btnIzmeniVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnIzmeniFiskalniRacun.Visibility = Visibility.Collapsed;
    btnIzmeniProizvodjaca.Visibility = Visibility.Collapsed;

    btnObrisiProdavca.Visibility = Visibility.Visible;

    btnObrisiProizvod.Visibility = Visibility.Collapsed;
    btnObrisiKupca.Visibility = Visibility.Collapsed;
    btnObrisiDobavljacka.Visibility = Visibility.Collapsed;
    btnObrisiVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnObrisiFiskalniRacun.Visibility = Visibility.Collapsed;
    btnObrisiProizvodjaca.Visibility = Visibility.Collapsed;
}

```

```

try
{
    konekcija.Open();

    string upit = @"select ProdavacID as ID, Username, Password from tblProdavac";
    SqlDataAdapter dataAdapter = new SqlDataAdapter(upit, konekcija);
    DataTable dataTable = new DataTable("Prodavac");

    dataAdapter.Fill(dataTable);

    dataGridCentralni.ItemsSource = dataTable.DefaultView;
}
catch (Exception exception)
{
    System.Diagnostics.Debug.WriteLine(exception.Message.ToString());
}
finally
{
    if (konekcija != null)
        konekcija.Close();
}
}

```

Slika 71 - Sadržaj metode btnProdavci_Click


```

private void btnFiskalniRacuni_Click(object sender, RoutedEventArgs e)
{
    btnDodajFiskalniRacun.Visibility = Visibility.Visible;

    btnDodajProdavca.Visibility = Visibility.Collapsed;
    btnDodajKupca.Visibility = Visibility.Collapsed;
    btnDodajDobavljacka.Visibility = Visibility.Collapsed;
    btnDodajVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnDodajProizvod.Visibility = Visibility.Collapsed;
    btnDodajProizvodjaca.Visibility = Visibility.Collapsed;

    btnIzmeniFiskalniRacun.Visibility = Visibility.Visible;

    btnIzmeniProdavca.Visibility = Visibility.Collapsed;
    btnIzmeniKupca.Visibility = Visibility.Collapsed;
    btnIzmeniDobavljacka.Visibility = Visibility.Collapsed;
    btnIzmeniVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnIzmeniProizvod.Visibility = Visibility.Collapsed;
    btnIzmeniProizvodjaca.Visibility = Visibility.Collapsed;

    btnObrisiFiskalniRacun.Visibility = Visibility.Visible;

    btnObrisiProdavca.Visibility = Visibility.Collapsed;
    btnObrisiKupca.Visibility = Visibility.Collapsed;
    btnObrisiDobavljacka.Visibility = Visibility.Collapsed;
    btnObrisiVrstuProizvoda.Visibility = Visibility.Collapsed;
    btnObrisiProizvod.Visibility = Visibility.Collapsed;
    btnObrisiProizvodjaca.Visibility = Visibility.Collapsed;
}

```

```

try
{
    konekcija.Open();

    string upit = @"select FiskalniRacunID as ID, DatumIzdavanja as 'Datum izdavanja', VremeIzdavanja as
                    'Vreme izdavanja', MestoIzdavanja as 'Mesto izdavanja',
                    Iznos, Kolicina as Količina from tblFiskalniRacun
                    inner join tblProdavac on tblFiskalniRacun.ProdavacID = tblProdavac.ProdavacID
                    inner join tblKupac on tblFiskalniRacun.KupacID = tblKupac.KupacID
                    inner join tblProizvod on tblFiskalniRacun.ProizvodID = tblProizvod.ProizvodID";

    SqlDataAdapter dataAdapter = new SqlDataAdapter(upit, konekcija);
    DataTable dataTable = new DataTable("FiskalniRacun");

    dataAdapter.Fill(dataTable);

    dataGridCentralni.ItemsSource = dataTable.DefaultView;
}
catch (Exception exception)
{
    System.Diagnostics.Debug.WriteLine(exception.Message.ToString());
}
finally
{
    if (konekcija != null)
        konekcija.Close();
}
}

```

Slika 72 - Sadržaj metode btnFiskalniRacuni_Click

Sadržaj metoda generisanih na klik dugmeta *Dodaj* je poprilično jednostavan. Pri kliku na to dugme, 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. Na narednim slikama prikazan je sadržaj metode za dodavanje za svih 7 tabela.

```
private void btnDodajProizvod_Click(object sender, RoutedEventArgs e)
{
    frmProizvod prozor = new frmProizvod();
    prozor.ShowDialog();
    PocetniDataGrid(dataGridCentralni); // refresh grida
}
```

Slika 73 - Sadržaj metode btnDodajProizvod_Click

```
private void btnDodajVrstuProizvoda_Click(object sender, RoutedEventArgs e)
{
    frmVrstaProizvoda prozor = new frmVrstaProizvoda();
    prozor.ShowDialog();
    btnVrsteProizvoda_Click(sender, e); // refresh
}
```

Slika 74 - Sadržaj metode btnDodajVrstuProizvoda_Click

```
private void btnDodajProizvodjaca_Click(object sender, RoutedEventArgs e)
{
    frmProizvodjac prozor = new frmProizvodjac();
    prozor.ShowDialog();
    btnProizvodjaci_Click(sender, e);
}
```

Slika 75 - Sadržaj metode btnDodajProizvodjaca_Click

```
private void btnDodajDobavljacka_Click(object sender, RoutedEventArgs e)
{
    frmDobavljac prozor = new frmDobavljac();
    prozor.ShowDialog();
    btnDobavljacki_Click(sender, e);
}
```

Slika 76 - Sadržaj metode btnDodajDobavljacka_Click

```
private void btnDodajKupca_Click(object sender, RoutedEventArgs e)
{
    frmKupac prozor = new frmKupac();
    prozor.ShowDialog();
    btnKupci_Click(sender, e);
}
```

Slika 77 - Sadržaj metode btnDodajKupca_Click

```
private void btnDodajProdavca_Click(object sender, RoutedEventArgs e)
{
    frmProdavac prozor = new frmProdavac();
    prozor.ShowDialog();
    btnProdavci_Click(sender, e);
}
```

Slika 78 - Sadržaj metode btnDodajProdavca_Click

```
private void btnDodajFiskalniRacun_Click(object sender, RoutedEventArgs e)
{
    frmFiskalniRacun prozor = new frmFiskalniRacun();
    prozor.ShowDialog();
    btnFiskalniRacuni_Click(sender, e);
}
```

Slika 79 - Sadržaj metode btnDodajFikslaniRacun_Click

Brisanje zapisa iz baze podataka obavlja se putem klika na dugme *Obriši* na način prikazan na *Slici 80*. Nakon otvaranja konekcije, kreira se objekat klase *DataRowView* koji u sebi čuva informaciju o indeksu reda koji je korisnik prethodno selektovao. Nakon toga kreira se *Delete* SQL komanda putem koje će se u samoj bazi obrisati željena stavka. Pre samog izvršavanja komande, kreira 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 ukoliko korisnik nije selektovao ni jedan red a pritisnuo je dugme za brisanje, a drugi ukoliko pokuša da obriše entitet iz tabele čiji je primarni ključ distribuiran u nekoj od drugih tabela. Nakon toga, konekcija se zatvara i početna strana osvežava sa izmenjenim podacima.

```
private void btnObrisiProizvod_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();
        DataRowView red = (DataRowView) dataGridCentralni.SelectedItems[0];
        string upit = @"Delete from tblProizvod Where ProizvodID = " + red["ID"];
        SqlCommand komanda = new SqlCommand(upit, konekcija);
        MessageBoxResult poruka = MessageBox.Show("Da li ste sigurni da želite da izbrišete selektovani red?", "Upozorenje!", MessageBoxButton.YesNo,
            MessageBoxImage.Question);

        if (poruka == MessageBoxResult.Yes)
            komanda.ExecuteNonQuery();
    }
    catch (ArgumentException)
    {
        MessageBox.Show("Niste selektovali odgovarajući red!", "Obaveštenje", MessageBoxButton.OK, MessageBoxImage.Warning);
    }
    catch (SQLException)
    {
        MessageBox.Show("Podaci koje želite da izbrišete su povezani sa podacima u drugim tabelama.", "Upozorenje!", MessageBoxButton.OK,
            MessageBoxImage.Information);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
    PocetniDataGrid(dataGridCentralni);
}
```

Slika 80 - Sadržaj metode btnObrisiProizvod_Click

Na ostalim slikama, prikazana su brisanja zapisa i iz ostalih 6 tabela: Vrste proizvoda (*Slika 81*), Proizvođači (*Slika 82*), Dobavljači (*Slika 83*), Kupci (*Slika 84*), Prodavci (*Slika 85*) i Fiskalni računi (*Slika 86*).

```

private void btnObrisiVrstuProizvoda_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();
        DataRowView red = (DataRowView)dataGridCentralni.SelectedItems[0];
        string upit = @"Delete from tblVrstaProizvoda Where VrstaProizvodaID = " + red["ID"];
        SqlCommand komanda = new SqlCommand(upit, konekcija);
        MessageBoxResult poruka =
            MessageBox.Show("Da li ste sigurni da želite da izbrišete selektovani red?", "Upozorenje!", MessageBoxButton.YesNo, MessageBoxImage.Question);

        if (poruka == MessageBoxResult.Yes)
            komanda.ExecuteNonQuery();
    }
    catch (ArgumentException)
    {
        MessageBox.Show("Niste selektovali odgovarajući red!", "Obaveštenje", MessageBoxButton.OK, MessageBoxImage.Warning);
    }
    catch (SQLException)
    {
        MessageBox.Show("Podaci koje želite da izbrišete su povezani sa podacima u drugim tabelama.", "Upozorenje!", MessageBoxButton.OK,
            MessageBoxImage.Information);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
    btnVrsteProizvoda_Click(sender, e);
}

```

Slika 81 - Sadržaj metode btnObrisiVrstuProizvoda_Click

```

private void btnObrisiProizvodjaca_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();
        DataRowView red = (DataRowView)dataGridCentralni.SelectedItems[0];
        string upit = @"Delete from tblProizvodjac Where ProizvodjacID = " + red["ID"];
        SqlCommand komanda = new SqlCommand(upit, konekcija);
        MessageBoxResult poruka =
            MessageBox.Show("Da li ste sigurni da želite da izbrišete selektovani red?", "Upozorenje!", MessageBoxButton.YesNo, MessageBoxImage.Question);

        if (poruka == MessageBoxResult.Yes)
            komanda.ExecuteNonQuery();
    }
    catch (ArgumentException)
    {
        MessageBox.Show("Niste selektovali odgovarajući red!", "Obaveštenje", MessageBoxButton.OK, MessageBoxImage.Warning);
    }
    catch (SQLException)
    {
        MessageBox.Show("Podaci koje želite da izbrišete su povezani sa podacima u drugim tabelama.", "Upozorenje!", MessageBoxButton.OK,
            MessageBoxImage.Information);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
    btnProizvodjaci_Click(sender, e);
}

```

Slika 82 - Sadržaj metode btnObrisiProizvodjaca_Click

```

private void btnObrisiDobavljacka_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();
        DataRowView red = (DataRowView)dataGridCentralni.SelectedItems[0];
        string upit = @"Delete from tblDobavljac Where DobavljacID = " + red["ID"];
        SqlCommand komanda = new SqlCommand(upit, konekcija);
        MessageBoxResult poruka =
            MessageBox.Show("Da li ste sigurni da želite da izbrišete selektovani red?", "Upozorenje!", MessageBoxButton.YesNo, MessageBoxImage.Question);

        if (poruka == MessageBoxResult.Yes)
            komanda.ExecuteNonQuery();
    }
    catch (ArgumentException)
    {
        MessageBox.Show("Niste selektovali odgovarajući red!", "Obaveštenje", MessageBoxButton.OK, MessageBoxImage.Warning);
    }
    catch (SqlException)
    {
        MessageBox.Show("Podaci koje želite da izbrišete su povezani sa podacima u drugim tabelama.", "Upozorenje!", MessageBoxButton.OK,
            MessageBoxImage.Information);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
    btnDobavljacki_Click(sender, e);
}

```

Slika 83 - Sadržaj metode btnObrisiDobavljacka_Click

```

private void btnObrisiKupca_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();
        DataRowView red = (DataRowView)dataGridCentralni.SelectedItems[0];
        string upit = @"Delete from tblKupac Where KupacID = " + red["ID"];
        SqlCommand komanda = new SqlCommand(upit, konekcija);
        MessageBoxResult poruka =
            MessageBox.Show("Da li ste sigurni da želite da izbrišete selektovani red?", "Upozorenje!", MessageBoxButton.YesNo, MessageBoxImage.Question);

        if (poruka == MessageBoxResult.Yes)
            komanda.ExecuteNonQuery();
    }
    catch (ArgumentException)
    {
        MessageBox.Show("Niste selektovali odgovarajući red!", "Obaveštenje", MessageBoxButton.OK, MessageBoxImage.Warning);
    }
    catch (SqlException)
    {
        MessageBox.Show("Podaci koje želite da izbrišete su povezani sa podacima u drugim tabelama.", "Upozorenje!", MessageBoxButton.OK,
            MessageBoxImage.Information);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
    btnKupci_Click(sender, e);
}

```

Slika 84 - Sadržaj metode btnObrisiKupca_Click

```

private void btnObrisiProdavca_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();
        DataRowView red = (DataRowView)dataGridCentralni.SelectedItems[0];
        string upit = @"Delete from tblProdavac Where ProdavacID = " + red["ID"];
        SqlCommand komanda = new SqlCommand(upit, konekcija);
        MessageBoxResult poruka =
            MessageBox.Show("Da li ste sigurni da želite da izbrišete selektovani red?", "Upozorenje!", MessageBoxButton.YesNo, MessageBoxImage.Question);

        if (poruka == MessageBoxResult.Yes)
            komanda.ExecuteNonQuery();
    }
    catch (ArgumentException)
    {
        MessageBox.Show("Niste selektovali odgovarajući red!", "Obaveštenje", MessageBoxButton.OK, MessageBoxImage.Warning);
    }
    catch (SqlException)
    {
        MessageBox.Show("Podaci koje želite da izbrišete su povezani sa podacima u drugim tabelama.", "Upozorenje!", MessageBoxButton.OK,
            MessageBoxImage.Information);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
    btnProdavci_Click(sender, e);
}

```

Slika 85 - Sadržaj metode btnObrisiProdavca_Click

```

private void btnObrisiFiskalniRacun_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();
        DataRowView red = (DataRowView)dataGridCentralni.SelectedItems[0];
        string upit = @"Delete from tblFiskalniRacun Where FiskalniRacunID = " + red["ID"];
        SqlCommand komanda = new SqlCommand(upit, konekcija);
        MessageBoxResult poruka =
            MessageBox.Show("Da li ste sigurni da želite da izbrišete selektovani red?", "Upozorenje!", MessageBoxButton.YesNo, MessageBoxImage.Question);

        if (poruka == MessageBoxResult.Yes)
            komanda.ExecuteNonQuery();
    }
    catch (ArgumentException)
    {
        MessageBox.Show("Niste selektovali odgovarajući red!", "Obaveštenje", MessageBoxButton.OK, MessageBoxImage.Warning);
    }
    catch (SqlException)
    {
        MessageBox.Show("Podaci koje želite da izbrišete su povezani sa podacima u drugim tabelama.", "Upozorenje!", MessageBoxButton.OK,
            MessageBoxImage.Information);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
    btnFiskalniRacuni_Click(sender, e);
}

```

Slika 87 - Sadržaj metode btnObrisiFiskalniRacun_Click

Za modifikaciju podataka, kao i za dodavanje, potrebno je da postoji odgovarajuća forma u koju će korisnik menjati odnosno unositi željene vrednosti. U ovom primeru, za svaku od tabela već je kreirana forma putem koje se novi zapisi upisuju u bazu podataka.

Kako ne bismo morali da kreiramo novu formu specijalno za modifikaciju podataka može se iskoristiti postojeća za dodavanje s tim što će jedina razlika biti u tome koju akciju izvršava dugme *Sačuvaj* te forme kada ga korisnik klikne. Na samom početku potrebno je, u okviru glavnog prozora deklarirati dve promenljive (*Slika 88*). Prva je tipa *boolean* i koristiće se kao uslov koji će dugmetu *Sačuvaj* govoriti da li je potrebno podatke upisati u bazu kao novi entitet ili izmeniti. Druga je tipa *object* i služi za pamćenje vrednosti selektovanog reda kada iz glavnog prozora pređemo u neku drugu formu.

```
public static bool izmena;  
public static object pomocniRed;
```

Slika 88 – Deklaracija promenljivih

Na *Slici 89* prikazana je metoda za izmenu *proizvođača*. Na samom početku, deklarirana promenljiva se postavlja na *true*. Nakon toga instancira se prozor njene forme i otvara konekcija. Pomoću *DataRowView* objekta, kao i kod brisanja, pamti se koji je red je selektovan. Razlika je u tome što se sada, u drugu promenljivu (*pomocniRed*) 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 popunjenim vrednostima traženog zapisa. Dalja funkcionalnost obavlja se unutar te forme i dugmeta *Sačuvaj* i biće objašnjena kasnije. U okviru ove metode, u *finally* bloku, konekcija se zatvara, osvežava se početna strana sa izmenjenim proizvođačima, a promenljiva *izmena* vraća na *false* u slučaju da nakon izmene korisnik želi da doda novi zapis u bazu podataka.


```

private void btnIzmeniProizvodjaca_Click(object sender, RoutedEventArgs e)
{
    try
    {
        izmena = true;

        konekcija.Open();

        DataRowView red = (DataRowView)dataGridCentralni.SelectedItems[0];
        pomocniRed = red;
        string upit = "select * from tblProizvodjac where ProizvodjacID = " + red["ID"];
        SqlCommand komanda = new SqlCommand(upit, konekcija);
        SqlDataReader citac = komanda.ExecuteReader();
        frmProizvodjac prozor = new frmProizvodjac();

        while (citac.Read())
            prozor.txtNazivProizvodjaca.Text = citac["NazivProizvodjaca"].ToString();

        prozor.ShowDialog();
    }
    catch (ArgumentOutOfRangeException)
    {
        MessageBox.Show("Niste selektovali odgovarajući red.", "Obaveštenje", MessageBoxButton.OK, MessageBoxImage.Warning);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }

    btnProizvodjaci_Click(sender, e);

    izmena = false;
}

```

Slika 89 – Sadržaj metode btnIzmeniProizvodjaca_Click

Na ostalim slikama, prikazane su metode za izmenu zapisa i iz ostalih 6 tabela: Dobavljači (*Slika 90*), Kupci (*Slika 91.*), Prodavci (*Slika 92*), Fiskalni računi (*Slika 93*), Proizvodi (*Slika 94*) i Vrste proizvoda (*Slika 95*).

```

private void btnIzmeniDobavljacka_Click(object sender, RoutedEventArgs e)
{
    try
    {
        izmena = true;

        konekcija.Open();

        DataRowView red = (DataRowView)dataGridCentralni.SelectedItems[0];

        pomocniRed = red;

        string upit = "select * from tblDobavljac where DobavljacID = " + red["ID"];

        SqlCommand komanda = new SqlCommand(upit, konekcija);
        SqlDataReader citac = komanda.ExecuteReader();
        frmDobavljac prozor = new frmDobavljac();

        while (citac.Read())
            prozor.txtNazivFirme.Text = citac["NazivFirme"].ToString();

        prozor.ShowDialog();
    }
    catch (ArgumentOutOfRangeException)
    {
        MessageBox.Show("Niste selektovali odgovarajući red.", "Obaveštenje", MessageBoxButton.OK, MessageBoxImage.Warning);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }

    btnDobavljacki_Click(sender, e);

    izmena = false;
}

```

Slika 90 – Sadržaj metode btnIzmeniDobavljacka_Click

```

private void btnIzmeniKupca_Click(object sender, RoutedEventArgs e)
{
    try
    {
        izmena = true;

        konekcija.Open();

        DataRowView red = (DataRowView)dataGridCentralni.SelectedItems[0];
        pomocniRed = red;
        string upit = "select * from tblKupac where KupacID = " + red["ID"];
        SqlCommand komanda = new SqlCommand(upit, konekcija);
        SqlDataReader citac = komanda.ExecuteReader();
        frmKupac prozor = new frmKupac();

        while (citac.Read())
            prozor.txtBrojMobilnogTelefona.Text = citac["BrojMobilnogTelefona"].ToString();

        prozor.ShowDialog();
    }
    catch (ArgumentOutOfRangeException)
    {
        MessageBox.Show("Niste selektovali odgovarajući red.", "Obaveštenje", MessageBoxButton.OK, MessageBoxImage.Warning);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }

    btnKupci_Click(sender, e);

    izmena = false;
}

```

Slika 91 – Sadržaj metode btnIzmeniKupca_Click

```

private void btnIzmeniProdavca_Click(object sender, RoutedEventArgs e)
{
    try
    {
        izmena = true;

        konekcija.Open();

        DataRowView red = (DataRowView)dataGridCentralni.SelectedItems[0];

        pomocniRed = red;
        string upit = "select * from tblProdavac where ProdavacID = " + red["ID"];
        SqlCommand komanda = new SqlCommand(upit, konekcija);
        SqlDataReader citac = komanda.ExecuteReader();
        frmProdavac prozor = new frmProdavac();

        while (citac.Read())
        {
            prozor.txtUserName.Text = citac["Username"].ToString();
            prozor.txtPassword.Text = citac["Password"].ToString();
        }

        prozor.ShowDialog();
    }
    catch (ArgumentOutOfRangeException)
    {
        MessageBox.Show("Niste selektovali odgovarajući red.", "Obaveštenje", MessageBoxButton.OK, MessageBoxImage.Warning);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }

    btnProdavci_Click(sender, e);

    izmena = false;
}

```

Slika 92 – Sadržaj metode btnIzmeniProdavca_Click

```

private void btnIzmeniFiskalniRacun_Click(object sender, RoutedEventArgs e)
{
    try
    {
        izmena = true;
        konekcija.Open();
        DataRowView red = (DataRowView)dataGridCentralni.SelectedItems[0];
        pomocniRed = red;
        string upit = "select * from tblFiskalniRacun where FiskalniRacunID = " + red["ID"];
        SqlCommand komanda = new SqlCommand(upit, konekcija);
        SqlDataReader citac = komanda.ExecuteReader();
        frmFiskalniRacun prozor = new frmFiskalniRacun();
        while (citac.Read())
        {
            prozor.DatumIzdavanja.Text = citac["DatumIzdavanja"].ToString();
            prozor.txtVremeIzdavanja.Text = citac["VremeIzdavanja"].ToString();
            prozor.txtMestoIzdavanja.Text = citac["MestoIzdavanja"].ToString();
            prozor.txtKolicina.Text = citac["Kolicina"].ToString();
            prozor.txtCena.Text = citac["Iznos"].ToString();
            prozor.cbxDobavljac.SelectedValue = citac["DobavljacID"].ToString();
            prozor.cbxProizvod.SelectedValue = citac["ProizvodID"].ToString();
            prozor.cbxDobavljac.SelectedValue = citac["DobavljacID"].ToString();
        }
        prozor.ShowDialog();
    }
    catch (ArgumentOutOfRangeException)
    {
        MessageBox.Show("Niste selektovali odgovarajući red.", "Obaveštenje", MessageBoxButton.OK, MessageBoxImage.Warning);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
    btnFiskalniRacuni_Click(sender, e);
    izmena = false;
}

```

Slika 93 – Sadržaj metode btnIzmeniFiskalniRacun_Click

```

private void btnIzmeniProizvod_Click(object sender, RoutedEventArgs e)
{
    try
    {
        izmena = true;
        konekcija.Open();
        DataRowView red = (DataRowView)dataGridCentralni.SelectedItems[0];
        pomocniRed = red;
        string upit = "select * from tblProizvod where ProizvodID = " + red["ID"];
        SqlCommand komanda = new SqlCommand(upit, konekcija);
        SqlDataReader citac = komanda.ExecuteReader();
        frmProizvod prozor = new frmProizvod();

        while (citac.Read())
        {
            prozor.txtCena.Text = citac["Cena"].ToString();
            prozor.txtSifra.Text = citac["Sifra"].ToString();
            prozor.txtNaziv.Text = citac["Naziv"].ToString();
            prozor.cbxDobavljac.SelectedValue = citac["DobavljacID"].ToString();
            prozor.cbxProizvodja.SelectedValue = citac["ProizvodjaID"].ToString();
            prozor.cbxVrstaProizvoda.SelectedValue = citac["VrstaProizvodaID"].ToString();
            prozor.cbxDobavljac.SelectedValue = citac["DobavljacID"].ToString();
        }

        prozor.ShowDialog();
    }
    catch (ArgumentOutOfRangeException)
    {
        MessageBox.Show("Niste selektovali odgovarajući red.", "Obaveštenje", MessageBoxButton.OK, MessageBoxImage.Warning);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
    PocetniDataGrid(dataGridCentralni);
    izmena = false; // pri izmeni je true ali kada se izmeni postavlja se na false
}

```

Slika 94 – Sadržaj metode btnIzmeniProizvod_Click

```

private void btnIzmeniVrstuProizvoda_Click(object sender, RoutedEventArgs e)
{
    try
    {
        izmena = true;

        konekcija.Open();

        DataRowView red = (DataRowView)dataGridCentralni.SelectedItems[0];
        pomocniRed = red;
        string upit = "select * from tblVrstaProizvoda where VrstaProizvodaID = " + red["ID"];
        SqlCommand komanda = new SqlCommand(upit, konekcija);
        SqlDataReader citac = komanda.ExecuteReader();
        frmVrstaProizvoda prozor = new frmVrstaProizvoda();

        while (citac.Read())
            prozor.txtNazivVrsteProizvoda.Text = citac["NazivVrsteProizvoda"].ToString();

        prozor.ShowDialog();
    }
    catch (ArgumentOutOfRangeException)
    {
        MessageBox.Show("Niste selektovali odgovarajući red.", "Obaveštenje", MessageBoxButton.OK, MessageBoxImage.Warning);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }

    btnVrsteProizvoda_Click(sender, e);

    izmena = false;
}

```

Slika 95 – Sadržaj metode *btnIzmeniVrstuProizvoda_Click*

Nakon što su željene vrednosti prikupljene unutar same forme, korisnik klikom na dugme *Sačuvaj* prosleđuje te zapise u bazu. U okviru *try* bloka metode ovog dugmeta, nalazi se *if* uslov na osnovu kojeg se, ukoliko je promenljiva *izmena* postavljena na *true* – zapis modifikuje odnosno ako je *false* – upise u bazu kao novi zapis. Ključ selektovanog reda nalazi se u promenljivoj *pomocniRed* i njegova vrednost se smešta u objekat *DataRowView*-a. Zatim se kreira odgovarajući *Update* upit na osnovu kojeg se podaci u bazi modifikuju. Na kraju, promenljiva *pomocniRed* se postavlja na *NULL*, kako bi sledeća modifikacija mogla da se dogodi na isti način, a prozor forme zatvara.

Sledi primer za Dobavljača (Slika 96). Na ostalim slikama, prikazane su izmene metode *Sačuvaj* i za ostalih 6 tabela: Fiskalni račun (Slika 97), Kupac (Slika 98), Prodavac (Slika 99), Proizvod (Slika 100), Proizvođač (Slika 101) i Vrsta proizvoda (Slika 102).

```

public void btnSacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();

        if (MainWindow.izmena)
        {
            DataRowView red = (DataRowView)MainWindow.pomocniRed;
            string update = @"update tblDobavljac Set NazivFirme = '" + txtNazivFirme.Text + "' Where DobavljacID = " + red["ID"];
            SqlCommand cmd = new SqlCommand(update, konekcija);
            cmd.ExecuteNonQuery();
            MainWindow.pomocniRed = null;
            this.Close();
        }
        else
        {
            string insert = @"insert into tblDobavljac(NazivFirme)
                            values ('" + txtNazivFirme.Text + "')";

            SqlCommand cmd = new SqlCommand(insert, konekcija);

            cmd.ExecuteNonQuery();

            this.Close(); // zatvori prozor
        }
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrednosti nije validan.", "Greška!", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
}

```

Slika 96 – Izmene metode btnSacuvaj_Click u formi frmDobavljac

```

private void btnSacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();
        if (MainWindow.izmena)
        {
            DataRowView red = (DataRowView)MainWindow.pomocniRed;
            string update = @"update tblFiskalniRacun Set DatumIzdavanja = '" + DatumIzdavanja.SelectedDate + "', VremeIzdavanja = '" + txtVremeIzdavanja.Text
                            + "', MestoIzdavanja = '" + txtMestoIzdavanja.Text + "', Iznos = " + (double.Parse(txtCena.Text) * int.Parse(txtKolicina.Text)).ToString() +
                            ", Kolicina = " + txtKolicina.Text + ", ProdavacID= " + cbxProdavac.SelectedValue + ", ProizvodID = " + cbxProizvod.SelectedValue +
                            ", KupacID = " + cbxKupac.SelectedValue + " Where FiskalniRacunID = " + red["ID"];
            SqlCommand cmd = new SqlCommand(update, konekcija);
            cmd.ExecuteNonQuery();
            MainWindow.pomocniRed = null;
            this.Close();
        }
        else
        {
            string insert = @"insert into tblFiskalniRacun(DatumIzdavanja, VremeIzdavanja, MestoIzdavanja, Iznos, Kolicina, ProdavacID, ProizvodID, KupacID)
                            values('" + DatumIzdavanja.SelectedDate + "', '" + txtVremeIzdavanja.Text + "', '" + txtMestoIzdavanja.Text + "', " +
                            (int.Parse(txtCena.Text) * int.Parse(txtKolicina.Text)).ToString() + ", " + txtKolicina.Text + ", " + cbxProdavac.SelectedValue +
                            ", " + cbxProizvod.SelectedValue + ", " + cbxKupac.SelectedValue + "));";
            SqlCommand cmd = new SqlCommand(insert, konekcija);
            cmd.ExecuteNonQuery();
            this.Close();
        }
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrednosti nije validan.", "Greška!", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
}

```

Slika 97 – Izmene metode btnSacuvaj_Click u formi frmFiskalniRacun

```

private void btnSacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();

        if (MainWindow.izmena)
        {
            DataRowView red = (DataRowView)MainWindow.pomocniRed;

            string update = @"update tblKupac Set BrojMobilnogTelefona = '" + txtBrojMobilnogTelefona.Text + "' Where KupacID = " + red["ID"];
            SqlCommand cmd = new SqlCommand(update, konekcija);
            cmd.ExecuteNonQuery();
            MainWindow.pomocniRed = null;
            this.Close();
        }
        else
        {
            string insert = @"insert into tblKupac(BrojMobilnogTelefona)
                values ('" + txtBrojMobilnogTelefona.Text + "')";

            SqlCommand cmd = new SqlCommand(insert, konekcija);
            cmd.ExecuteNonQuery();

            this.Close();
        }
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrednosti nije validan.", "Greška!", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
}

```

Slika 98 – Izmene metode `btnSacuvaj_Click` u formi `frmKupac`

```

private void btnSacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();

        if (MainWindow.izmena)
        {
            DataRowView red = (DataRowView)MainWindow.pomocniRed;

            string update = @"update tblProdavac Set Username = '" + txtUserName.Text + "', Password = '" + txtPassword.Text + "' Where ProdavacID = " +
                red["ID"];
            SqlCommand cmd = new SqlCommand(update, konekcija);
            cmd.ExecuteNonQuery();
            MainWindow.pomocniRed = null;
            this.Close();
        }
        else
        {
            string insert = @"insert into tblProdavac(Username, Password)
                values('" + txtUserName.Text + "', '" + txtPassword.Text + "')";

            SqlCommand cmd = new SqlCommand(insert, konekcija);
            cmd.ExecuteNonQuery();
            this.Close();
        }
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrednosti nije validan.", "Greška!", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
}

```

Slika 99 – Izmene metode `btnSacuvaj_Click` u formi `frmProdavac`


```

private void btnSacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();

        if (MainWindow.izmena)
        {
            DataRowView red = (DataRowView)MainWindow.pomocniRed;
            string update = @"update tblProizvod Set Cena = " + txtCena.Text + " , Sifra = '" + txtSifra.Text + "' , Naziv = '" + txtNaziv.Text + 
            "' , ProizvodjacID = " + cbxProizvodjac.SelectedValue + " , VrstaProizvodaID = " + cbxVrstaProizvoda.SelectedValue + " , DobavljacID = " 
            + cbxDobavljac.SelectedValue + " Where ProizvodID = " + red["ID"];
            SqlCommand cmd = new SqlCommand(update, konekcija);
            cmd.ExecuteNonQuery();
            MainWindow.pomocniRed = null;
            this.Close();
        }
        else
        {
            string insert = @"insert into tblProizvod(Cena, Sifra, Naziv, ProizvodjacID, VrstaProizvodaID, DobavljacID)
            values(" + txtCena.Text + " , " + txtSifra.Text + " , " + txtNaziv.Text + " , " + cbxProizvodjac.SelectedValue + 
            " , " + cbxVrstaProizvoda.SelectedValue + " , " + cbxDobavljac.SelectedValue + ")";
            SqlCommand cmd = new SqlCommand(insert, konekcija);
            cmd.ExecuteNonQuery();
            this.Close();
        }
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrednosti nije validan.", "Greška!", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
}

```

Slika 100 - Izmene metode btnSacuvaj_Click u formi frmProizvod

```

private void btnSacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();

        if (MainWindow.izmena)
        {
            DataRowView red = (DataRowView)MainWindow.pomocniRed;

            string upit = @"update tblProizvodjac Set NazivProizvodjaca = '" + txtNazivProizvodjaca.Text + "' Where ProizvodjacID = " + red["ID"];
            SqlCommand cmd = new SqlCommand(upit, konekcija);
            cmd.ExecuteNonQuery();
            MainWindow.pomocniRed = null;
            this.Close();
        }
        else
        {
            string insert = @"insert into tblProizvodjac(NazivProizvodjaca)
            values('" + txtNazivProizvodjaca.Text + "')";

            SqlCommand cmd = new SqlCommand(insert, konekcija);
            cmd.ExecuteNonQuery();

            this.Close();
        }
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrednosti nije validan.", "Greška!", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
}

```

Slika 101 - Izmene metode btnSacuvaj_Click u formi frmProizvodjac

```

private void btnSacuvaj_Click(object sender, RoutedEventArgs e)
{
    try
    {
        konekcija.Open();

        if (MainWindow.izmena)
        {
            DataRowView red = (DataRowView)MainWindow.pomocniRed;

            string upit = @"update tblVrstaProizvoda Set NazivVrsteProizvoda = '" + txtNazivVrsteProizvoda.Text +
                "' Where VrstaProizvodaID = " + red["ID"];
            SqlCommand cmd = new SqlCommand(upit, konekcija);

            cmd.ExecuteNonQuery();
            MainWindow.pomocniRed = null;
            this.Close();
        }
        else
        {
            string insert = @"insert into tblVrstaProizvoda(NazivVrsteProizvoda)
                values('" + txtNazivVrsteProizvoda.Text + "')";
            SqlCommand cmd = new SqlCommand(insert, konekcija);
            cmd.ExecuteNonQuery();
            this.Close();
        }
    }
    catch (SqlException)
    {
        MessageBox.Show("Unos određenih vrednosti nije validan.", "Greška!", MessageBoxButton.OK, MessageBoxImage.Error);
    }
    finally
    {
        if (konekcija != null)
            konekcija.Close();
    }
}

```

Slika 102 - Izmene metode btnSacuvaj_Click u formi frmVrstaProizvoda

5. Testiranje aplikacije

Pri pokretanju aplikacije, generiše se početni prozor kao na *Slici 103*.



Slika 103- Početni prozor pri pokretanju aplikacije

Klikom na dugmiće iz gornjeg StackPanel-a, vrši se prikaz podataka željene tabele na centralnom gridu (*Slika 104*).



Slika 104 – Izlistavanje podataka o prodavcima na početnom gridu

Klikom na dugme *Dodaj*, otvara se forma za dodavanje novog zapisa u bazu podataka (*Slika 105*).

The screenshot shows a window titled 'Prodavac' with a light blue background. It contains two input fields: 'Username' and 'Password'. At the bottom, there are two buttons: 'Sačuvaj' (green) and 'Otkazi' (red).

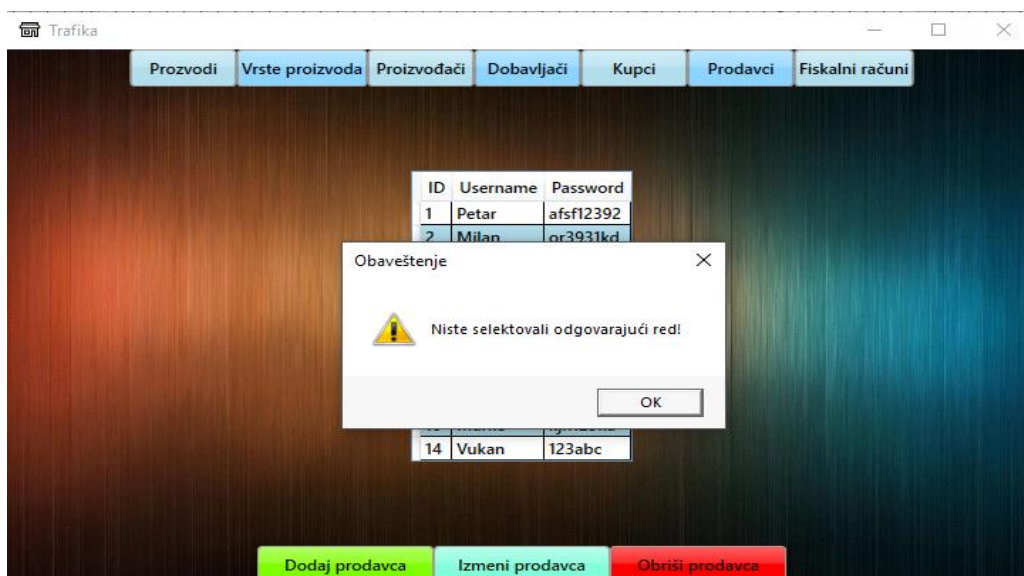
Slika 105 – Dodavanje novog prodavca u bazu podataka

Klikom na dugme *Sačuvaj*, novi zapis se dodaje u bazu podataka i prikazuje na centralnom gridu (*Slika 106*).



Slika 106 – Ispis dodatog prodavca

Klik na dugme *Izmeni* ili *Obriši*, a da pritom nije selektovan ni jedan red, uzrokuje sledeću grešku/obaveštenje (*Slika 107*).



Slika 107 – Pokušaj klika na dugme izmeni ili obriši bez selektovanja reda

Selektovanjem odgovarajućeg reda otvara se forma sa popunjenim vrednostima prethodnog zapisa koje korisnik dalje može modifikovati (*Slika 108*).

Prodavac

Username: Vukan

Password: 123abc

Sačuvaj Otkazi

14	Vukan	123abc
----	-------	--------

Dodaj prodavca Izmeni prodavca Obriši prodavca

Slika 108 – Otvaranje forme za modifikaciju

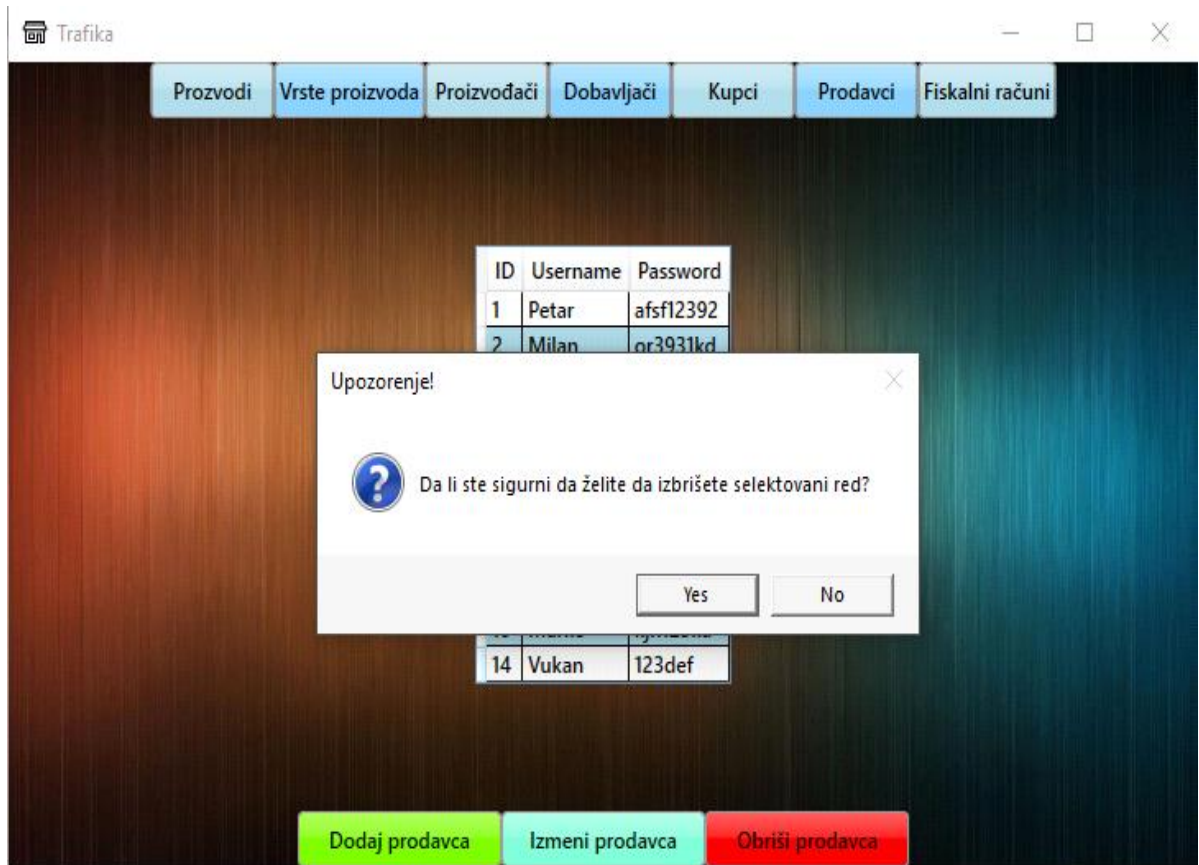
Nakon izmene željenih podataka, ažurirani zapis se prikazuje na ekranu (*Slika 109*).

ID	Username	Password
1	Petar	afsf12392
2	Milan	or3931kd
3	Nebojša	342341af
4	Goran	32423432
5	Dragana	03rjfsjff3
6	Ivana	312irudo
7	David	324erk32
8	Sandra	foj3nf2kj
9	Marijana	324rgfhk
10	Marko	kjh123ka
14	Vukan	123def

Dodaj prodavca Izmeni prodavca Obriši prodavca

Slika 109 – Ažuriran sadržaj nakon promene podataka

Brisanje podataka vrši se klikom na dugme *Obriši*, nakon što je korisnik prethodno selektovao željeni zapis za brisanje (*Slika 110*). Klikom na dugme *Yes*, korisnik potvrđuje brisanje, a na ekranu se generiše ažužiran ispis.



Slika 110 – Brisanje zapisa