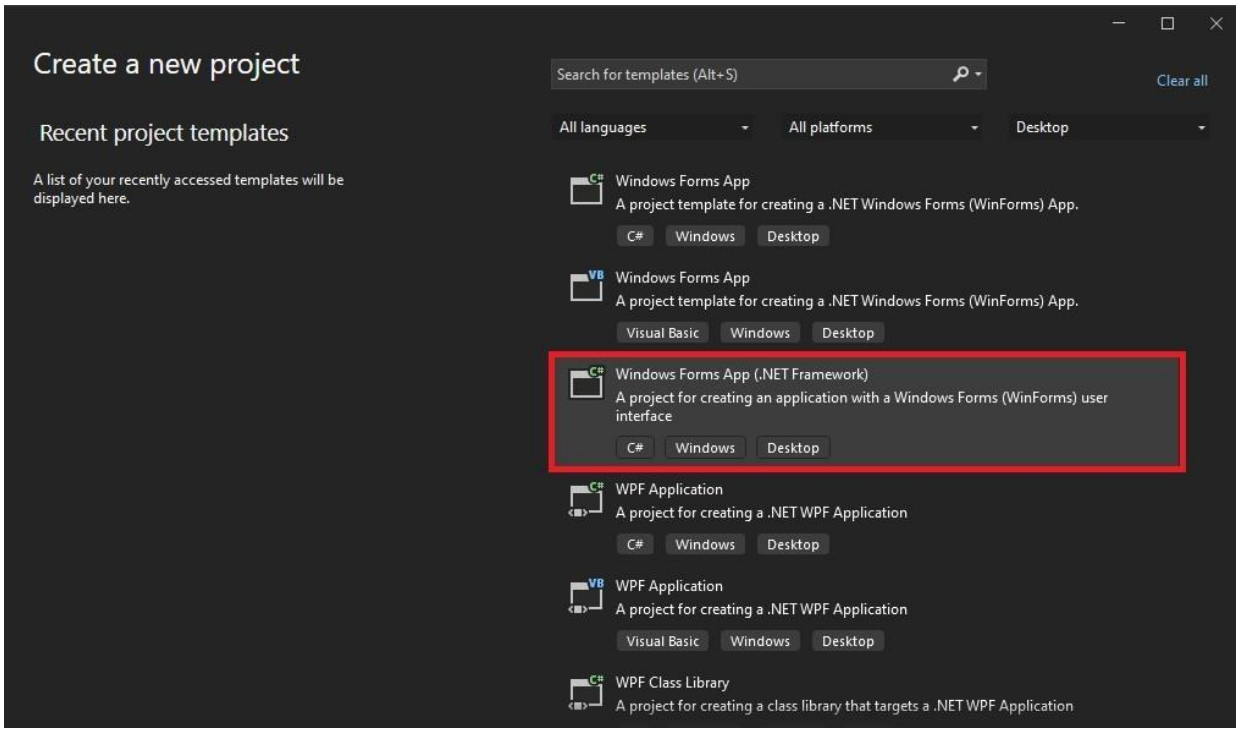
Praktični deo maturskog ispita (Programiranje) – Uputstvo i primeri

Uvod

Prilikom kreiranja projekta birati opciju Windows Forms App (.NET Framework).



Napomena

Kontrola sa vidljivim kolonama je *ListView*. Kolone se dodaju preko svojstva *Columns*, a da bi bile vidljive potrebno je svojstvo *View* promeniti na *Details*. Ukoliko je potrebno obezbediti da se selektuje ceo red vrednost svojstva *FullRowSelect* se menja na *True*.

Kontrola sa sivom pozadinom je *DataGridView*.

Kontrola sa grafikom je *Chart*. Tip grafika se bira unutar svojstva *Series* i opcije *ChartType*.

Kontrola sa tabovima je *TabControl*. Tabovi se dodaju i imenuju preko svojstva *TabPages*.

Kontrola sa dugmadima u vrhu je *ToolStrip*. Bitna svojstva za postavljanje i pozicioniranje teksta i slike su *DisplayStyle* i *TextImageRelation*. Za veličinu slike *ImageScalingSize*. Za pozicioniranje samog *ToolStrip*-a svojstvo *Dock*.

Kontrola za brojeve je *NumericUpDown*. Bitna svojstva su *Minimum*, *Maximum* i *Value*.

Kontrola padajućeg menija je *ComboBox*, a kontrola za unos je *TextBox* (svojstvo za maksimalan broj karaktera za unos je *MaxLength*).

Dodavanje nove forme u projekat se vrši desnim klikom na ime projekta u *Solution Explorer*-u i izborom opcije *Add – New – Form*.

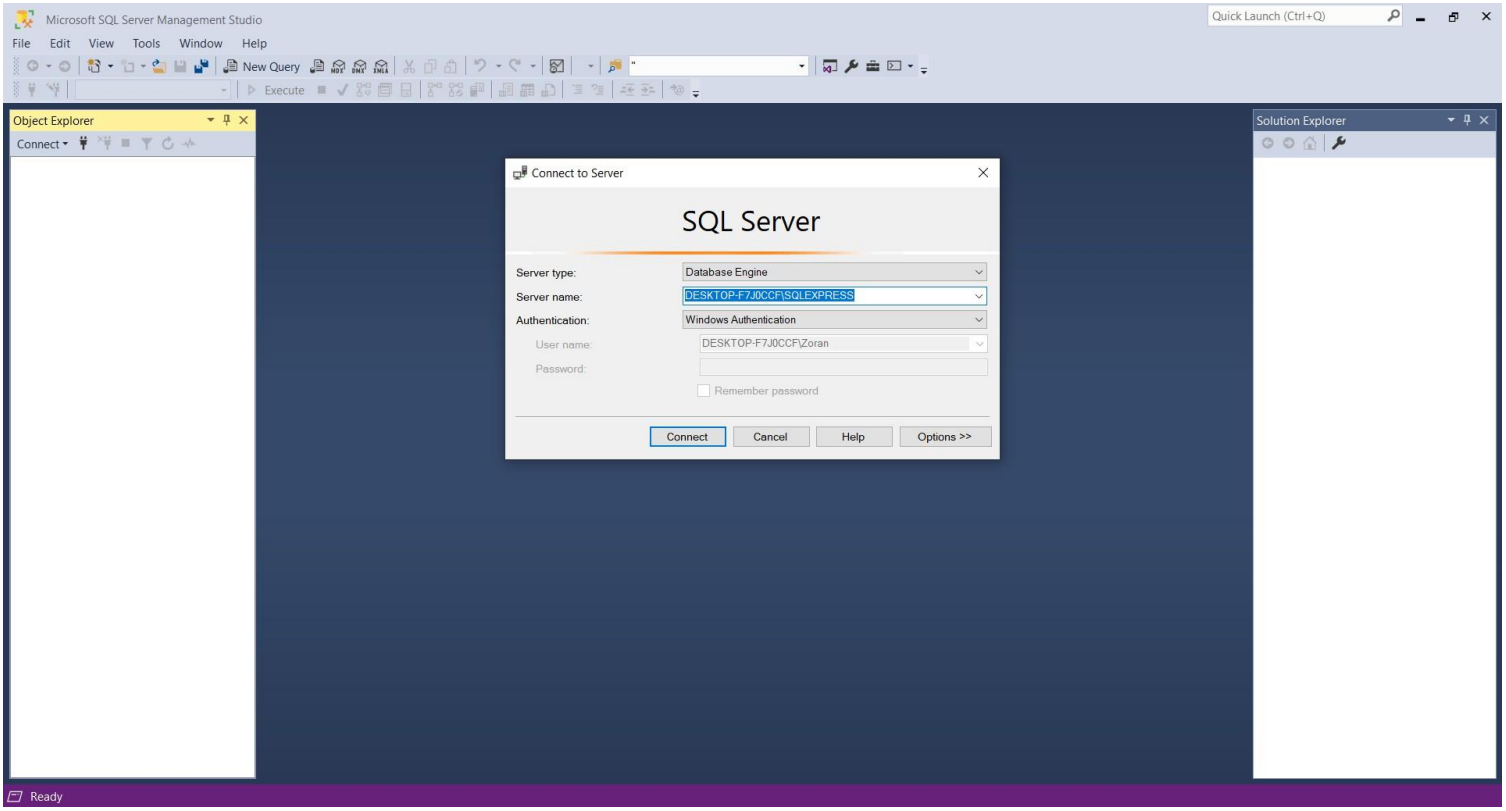
Slike koje se koriste (preko svojstava *BackgroundImage* i *Image*) ubacuju se u resurse programa izborom opcije *Project resource file* i *Import...*. Podešavanja veličine slike i uklapanja u kontrolu se vrše preko svojstava *BackgroundImageLayout* (za pozadinsku sliku) i *SizeMode* za običnu sliku.

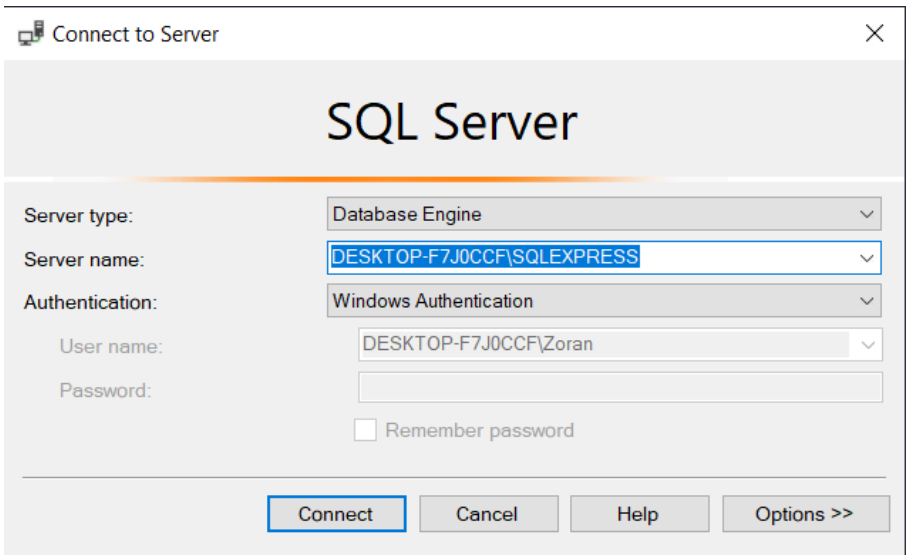
Rad sa bazama podataka

SQL Server Management Studio – početni koraci

Pokretanje Microsoft SQL Server Management Studio okruženja:

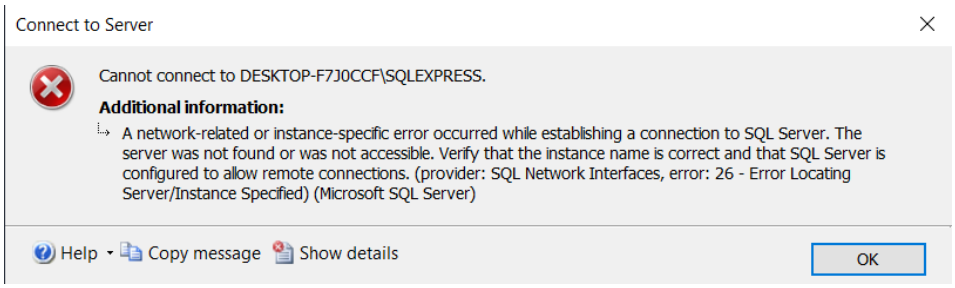
Nakon pokretanja programa dobija se radno okruženje i otvoren prozor *Connect to Server*:

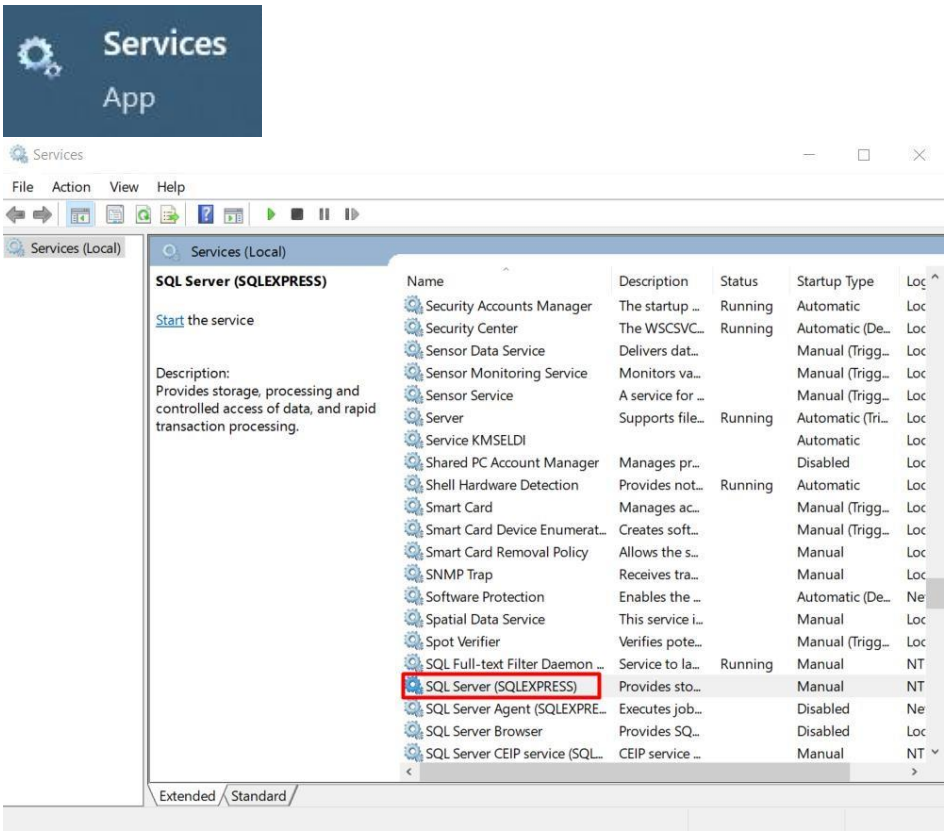




U prozoru *Connect to Server* potrebno je izabrati opciju *Connect*. Na mestu *Server name* će stojati ime_računara\SQLEXPRESS ako prilikom instalacije nije drugačije naznačeno.

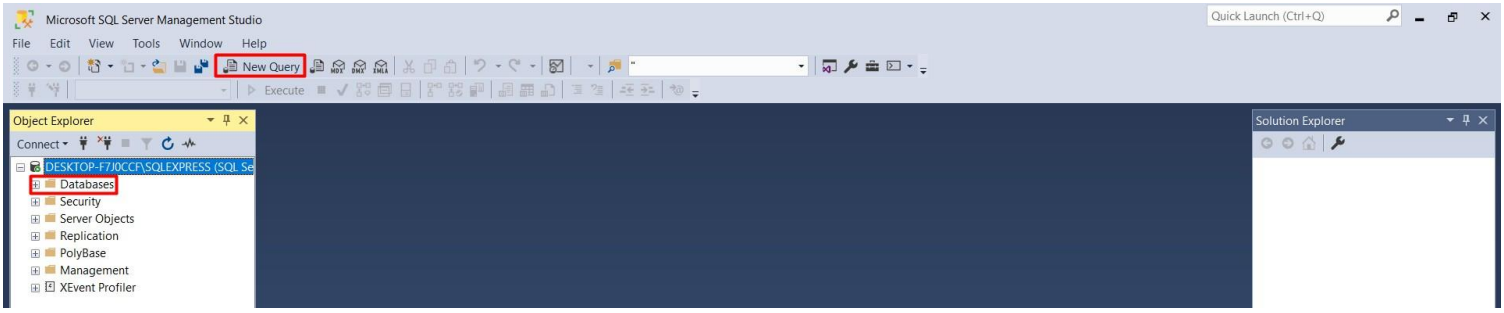
Ukoliko prilikom konektovanja na server dođe do greške prikazane na slici ispod, to je znak da SQL Server nije pokrenut na računaru.





Da bi se SQL Server pokrenuo pristupa se servisima (*Services*) i pronalazi se servis *SQL Server (SQLEXPRESS)* i pokreće se desnim klikom na njega i izborom opcije *Start*.

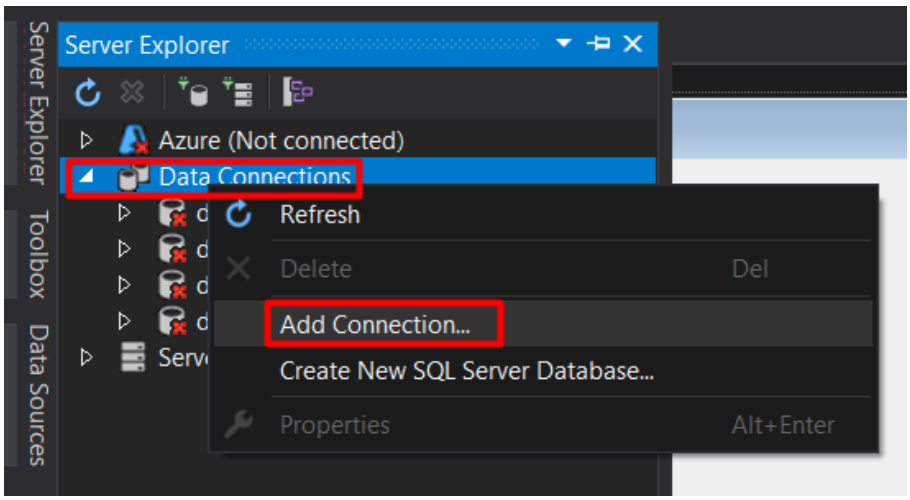
Nakon pokretanja potrebnog servisa ponovo biramo opciju *Connect* u prozoru *Connect to Server* i dobijamo radno okruženje sa *Object Explorer*-om na levoj, *Solution Explorer*-om na desnoj strani i praznim mestom u sredini. Upite kreiramo izborom opcije *New Query* iz gornje trake sa opcijama (ili kombinacijom tastera *CTRL* i *N*). Unutar *Object Explorer*-a će se nalaziti sve baze podataka koje kreirate, zajedno sa njihovim tabelama, pogledima itd.



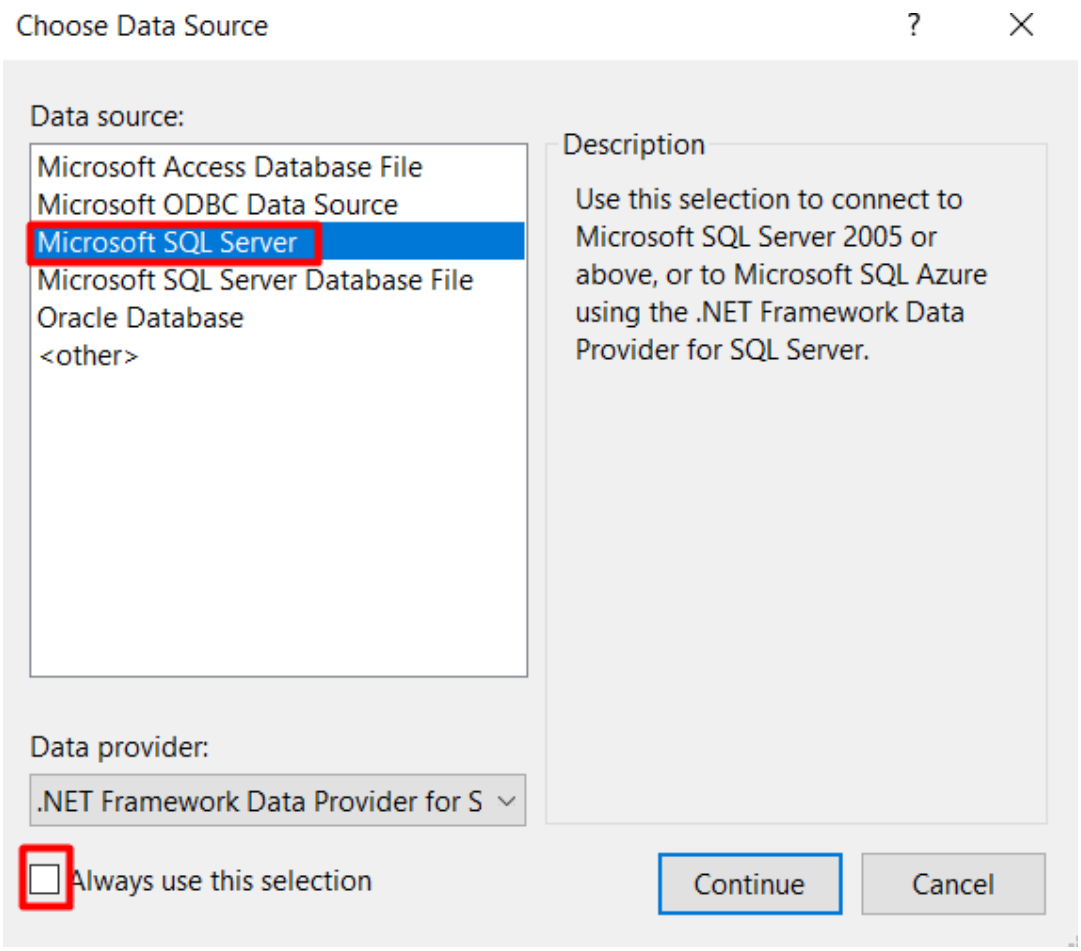
Nakon kreiranja novog upita možemo birati nad kojom bazom podataka želimo da ga izvršimo, izborom baze iz padajućeg menija u gornjoj traci sa alatima. Izborom opcije Execute izvršavamo kreirani upit.

Povezivanje baze podataka sa projektom – početni koraci

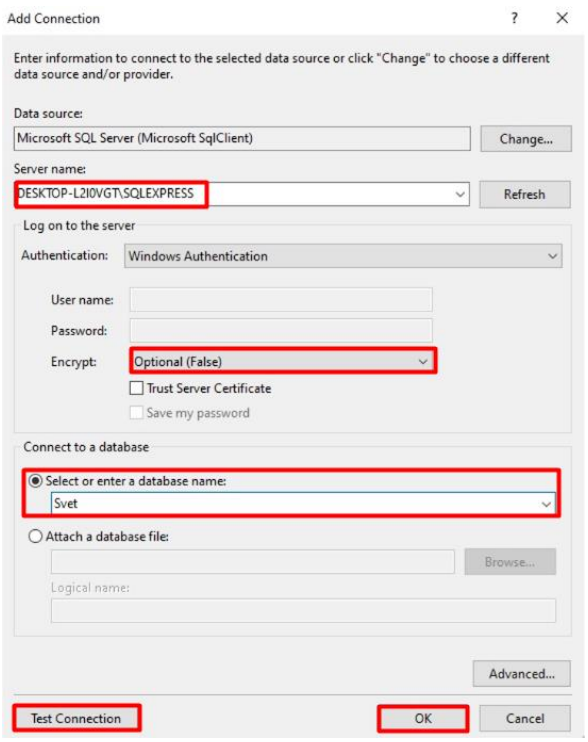
Unutar *Server Explorer*-a desnim klikom na *Data Connections* biramo opciju *Add Connection…* (ukoliko se *Server Explorer* ne nalazi na levoj strani radne površine otvaramo ga iz gornjeg menija biranjem opcija *View -> Server Explorer*).



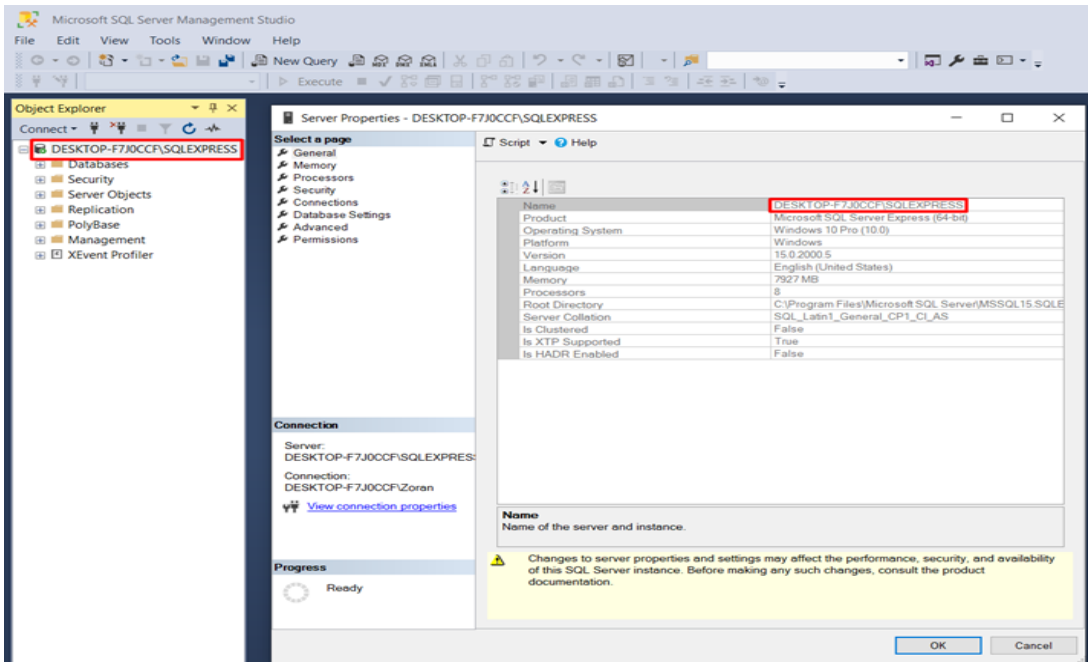
U prozoru *Choose Data Source* biramo opciju *Microsoft SQL Server*, a polje *Always use this selection* ostavljamo nečekirano. Zatim biramo opciju *Continue*.



U polje *Server name:* upisujemo ime servera (isto koje je i u SQL Management Studio-u). Obavezno je izabrati opciju *Optional (False)* unutar *Encrypt* dela. Zatim u padajućem meniju biramo bazu podataka za koju želimo da dodamo konekciju. Nakon toga pritiskom na dugme *Test Connection* dobijamo povratnu informaciju da li je konekcija uspešna (*Test connection succeeded* ako jeste). Ukoliko je konekcija uspešna biramo dugme *OK*.



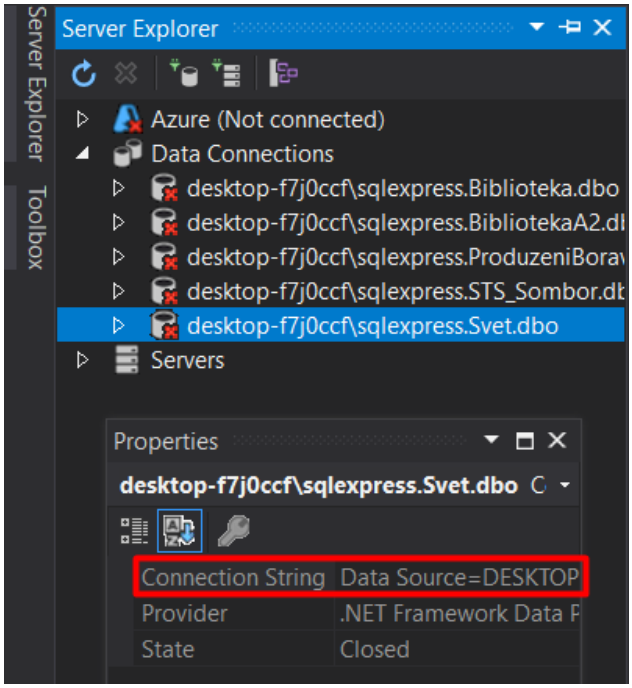
Desnim klikom na server u *Object Explorer*-u (leva slika) i izborom opcije *Properties* možemo doći do imena servera.



Konekcioni string

Konekcioni string koji nam je kasnije potreban za povezivanje sa bazom možemo naći u *Properties*-u od kreirane konekcije na bazu podataka.

Konekcioni string nam koristi da bi projekat znao na koji server i koju bazu podataka je potrebno da se poveže.



Kopirani tekst konekcionog stringa čuvamo unutar tekstualne promenljive proizvoljnog naziva (u primerima i priručniku promenljiva će biti nazivana *konekcioniString*).

```
string konekcioniString = @"Data Source=DESKTOP-F7J0CCF\SQLEXPRESS;Initial Catalog=Biblioteka;Integrated Security=True";
```

Klase za rad sa bazama podataka

Da bi koristili klase i metode za rad sa SQL Server bazom podataka na vrhu našeg projekta moramo dodati biblioteku *System.Data.SqlClient*.

```
using System.Data.SqlClient;
```

SqlConnection

SqlConnection klasa u C# predstavlja vezu između aplikacije napisane u C# i baze podataka u kojoj se nalaze podaci koje aplikacija treba koristiti. Ova klasa omogućava otvaranje, zatvaranje i upravljanje vezom između aplikacije i baze podataka.

Da bi se koristila SqlConnection klasa, prvo se uključuje biblioteka za rad sa bazama podataka (*System.Data.SqlClient*). Zatim se kreira nova instanca SqlConnection klase i prosleđuje joj se string koji sadrži informacije o žejlenoj bazi podataka, kao što su naziv servera, naziv baze podataka, korisničko ime i lozinka (konekcioni string).

```
            // Kreiranje instance klase SqlConnection

            SqlConnection konekcija = new SqlConnection(konekcioniString);

            // Otvaranje veze ka bazi podataka
            konekcija.Open();

            // Zatvaranje veze ka bazi podataka
            konekcija.Close();
```

SqlCommand

SqlCommand klasa u C# je deo ADO.NET biblioteke i koristi se za izvršavanje SQL naredbi nad bazom podataka. SqlCommand klasa omogućuje da izvršimo različite vrste SQL naredbi, kao što su SELECT, INSERT, UPDATE, DELETE, itd.

```
            /* Kreiranje instance klase SqlCommand (s tim da konekcija predstavlja instancu
            SqlConnection klase) */

            SqlCommand komanda = new SqlCommand("SELECT * FROM Tabela", konekcija);

            // SQL upit može da se čuva i unutar promenljive

            string select = "SELECT * FROM Tabela";

            SqlCommand komanda = new SqlCommand(select, konekcija);
```

Ukoliko se unutar upita komande nalaze i parametri njih dodajemo na sledeći način :

```
            SqlCommand komanda = new SqlCommand(SQLupit, konekcija);


            komanda.Parameters.AddWithValue("@ImeParametra", vrednost);
```

SqlDataReader

SqlDataReader klasa u C# omogućava programerima da pročitaju podatke iz SQL Server baze podataka. Ova klasa se koristi u kombinaciji sa SqlCommand klasom koja se koristi za izvršavanje SQL upita na bazi podataka. Kada se izvrši SQL upit korištenjem SqlCommand klase, SqlDataReader

klasa se koristi za čitanje podataka iz rezultujućeg skupa podataka. SqlDataReader klasa čita podatke u jednom smeru, red po red, tako da omogućava brzo i efikasno čitanje velikih skupova podataka.

```
string select = "SELECT * FROM Tabela";

SqlCommand komanda = new SqlCommand(select, konekcija);

// Izvršavanje čitanja podataka nad upitom iz komande

SqlDataReader citac = komanda.ExecuteReader();

while (citac.Read())

{

    // Kod koji se izvršava dok se čitaju podaci

}

// Zatvaranje čitača - prekid čitanja podataka iz komande

citac.Close();
```

SqlDataAdapter

SqlDataAdapter klasa u C# se koristi za preuzimanje podataka iz baze podataka i njihovo skladištenje u *DataSet* ili *DataTable* objekat. *DataSet* objekat može sadržati više *DataTable* objekata koji su usklađeni s tabelama u bazi podataka. Nakon što su podaci učitani u *DataSet* ili *DataTable* objekat, aplikacija ih može koristiti na razne načine.

```
// Kreiranje instance klase SqlDataAdapter
SqlDataAdapter adapter = new SqlDataAdapter(SQLupit, konekcija);

// Kreiranje tabele - instance klase DataTable
DataTable podaci = new DataTable();


// Popunjavanje kreirane tabele podacima iz adapter-ovog upita
adapter.Fill(podaci);
```

Rad sa kontrolama

Rad sa ListView-om

ListView kontrola se na formu dodaje preuzimanjem iz *Toolbox*-a.

- ***Dodavanje kolona*** - Kolone se unutar ListView-a mogu dodavati ili preko svojstva **Columns** unutar

Properties-a ili unutar koda forme (imeListView.Columns.Add("ImeKolone");).

- ***Prikaz kolona*** – Da bi videli naslove i podatke u kreiranim kolonama svojstvo **View** menjamo na vrednost **Details** (takođe ili preko *Properties*-a ili unutar koda (imeListView.View = View.Details;).
- ***Selektovanje celog reda*** – Ukoliko želimo da se klikom na red u ListView-u selektuju sve kolone tog reda menjamo svojstvo **FullRowSelect** u vrednost **True**.
- ***Selektovanje većeg broja redova*** – Ukoliko želimo da omogućimo/onemogućimo selektovanje većeg broja redova unutar ListView-a menjamo svojstvo **MultiSelect** u vrednost **True/False**.

Ukoliko želimo da određeni red bude selektovan:

```
imeListViewa.SelectedIndices.Add(indeks od željenog reda);
```

Ukoliko želimo da određeni red bude vidljiv (ukoliko je lista dugačka i potrebno je skrolovanje metoda *EnsureVisible()* će automatski dovesti do željenog reda):

```
imeListViewa.Items[indeks od željenog reda].EnsureVisible();
```

Popunjavanje kolona podacima

Da bi popunili kolone podacima koristimo ranije pomenute klase za rad sa bazama podataka. Primer :

```
// Dok čitač prolazi kroz podatke na osnovu prosleđenog upita kod unutar petlje će se izvršavati
            while (citac.Read())
            {
                /* Kreira se stavka ListView-a, a u konstruktor se postavlja prva željena vrednost u
                tekstualnoj vrednosti */
                ListViewItem prikaz = new ListViewItem(citac[0].ToString());
/* Na kreiranu stavku se dodaju SubItem-i, onoliko njih koliko podataka želimo da izvučemo iz upita
                Broj polja ne sme biti veći od broja polja koje vraća upit */
                prikaz.SubItems.Add(citac[1].ToString());
                prikaz.SubItems.Add(citac[2].ToString());
                prikaz.SubItems.Add(citac[3].ToString());
                prikaz.SubItems.Add(citac[4].ToString());
                // Kreirana stavka sa svojim SubItem-ima se dodaje ListView-u
                imeListViewa.Items.Add(prikaz);
            }
```

Rad sa ComboBox-om

Popunjavanje *ComboBox*-a podacima iz baze podataka se vrši pomoću korišćenja klasa *SqlDataAdapter* i *DataTable*.

```
SqlDataAdapter adapter = new SqlDataAdapter(SQLupit, konekcija);

DataTable podaci = new DataTable();
adapter.Fill(podaci);

// Popunjena tabela se dodaje kao DataSource
imeComboBoxa.DataSource = podaci;

// Za ValueMember se uzima unikatna vrednost iz upita
imeComboBoxa.ValueMember = "UnikatnoPoljeIzTabele";
```

Rad sa DataGridView-om

DataGridView kontrola se na formu dodaje preuzimanjem iz *Toolbox*-a.

```
SqlDataAdapter dataAdapter = new SqlDataAdapter(SQLupit);

DataTable podaci = new DataTable();
dataAdapter.Fill(podaci);

// Podatke iz tabele dodeljujemo DataGridView-u
imeDataGridViewa.DataSource = podaci;
```

Preko svojstva *ReadOnly* se omogućava/onemogućava da se vrši unos u polja *DataGridView*-a.

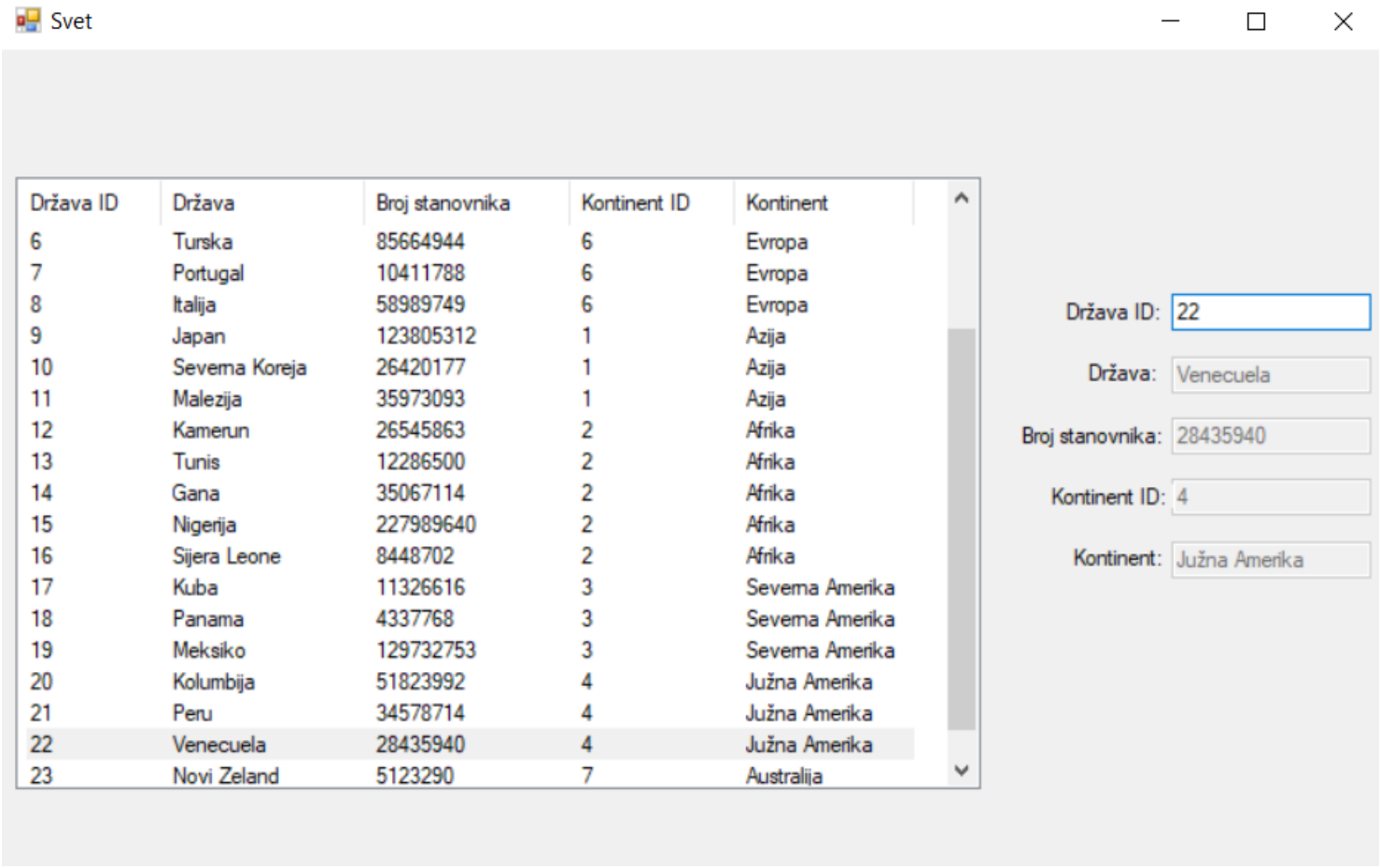
Rad sa Chart-om

Chart kontrola se na formu dodaje preuzimanjem iz *Toolbox*-a. U zavisnosti od tipa *Chart*-a (Line, Pie, Column...) zavisiće i izgled koda.

```
// Dodeljujemo izvor podataka našem chart-u
imeCharta.DataSource = dataAdapter;

// Na x osu postavljamo kolonu po kojoj želimo da se mere podaci

imeCharta.Series[0].XValueMember = "Kolona X ose";

// Na y osu postavljamo kolonu koja po kojoj će se meriti podaci sa x ose

imeCharta.Series[0].YValueMembers = "Kolona Y ose";

// Unosimo podatke unutar chart-a
imeCharta.DataBind();
```

Primeri

Svet



Država ID	Država	Broj stanovnika	Kontinent ID	Kontinent
6	Turska	85664944	6	Evropa
7	Portugal	10411788	6	Evropa
8	Italija	58989749	6	Evropa
9	Japan	123805312	1	Azija
10	Severna Koreja	26420177	1	Azija
11	Malezija	35973093	1	Azija
12	Kamerun	26545863	2	Afrika
13	Tunis	12286500	2	Afrika
14	Gana	35067114	2	Afrika
15	Nigerija	227989640	2	Afrika
16	Sijera Leone	8448702	2	Afrika
17	Kuba	11326616	3	Severna Amerika
18	Panama	4337768	3	Severna Amerika
19	Meksiko	129732753	3	Severna Amerika
20	Kolumbija	51823992	4	Južna Amerika
21	Peru	34578714	4	Južna Amerika
22	Venecuela	28435940	4	Južna Amerika
23	Novi Zeland	5123290	7	Australija

Baza podataka za zadatak: link

Pri pokretanju aplikacije *ListView* se popunjava podacima kao što je prikazano na gornjoj slici.

Jedini *TextBox* u kom je omogućen unos je onaj za unos ID-ja države. Onemogućen je unos slova i simbola dozvoljene su samo cifre u tom *TextBox*-u.

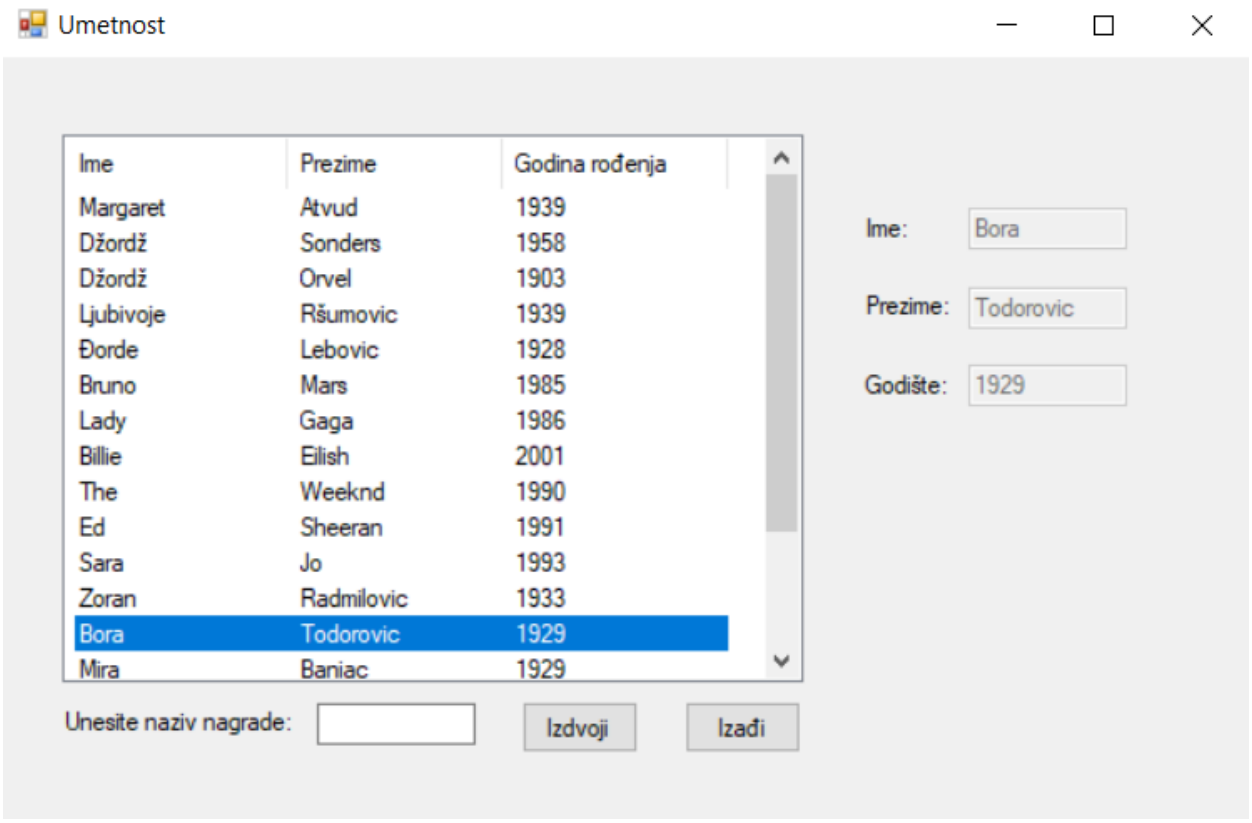
Unosom broja, ukoliko postoji država sa tim ID-jem u bazi, ta država se selektuje (ceo red), a ostali *TextBox*-ovi se popunjavaju odgovarajućim podacima.

```
using System;
using System.Data.SqlClient;
using System.Windows.Forms;

namespace Svet
{
    public partial class Form1 : Form
    {
        /* Konekcioni string možemo izvući iz Server Explorer-a desnim klikom na kreiranu
konekciju,
i on nam služi za povezivanje sa bazom podataka */
        string konekcioniString = @"Data Source=DESKTOP-F7J0CCF\SQLEXPRESS;Initial
Catalog=Svet;Integrated Security=True";
        /* SQL upite možemo skladištiti unutar tekstualnih promenljivih */
```

```
        string select = "SELECT DrzavaID, Naziv, BrojStanovnika, Drzava.KontinentID,
NazivKontinenta " +
        "FROM Drzava JOIN Kontinent ON Drzava.KontinentID = Kontinent.KontinentID " +
        "ORDER BY DrzavaID";

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            /* Kreiramo konekciju ka bazi podataka */
            SqlConnection konekcija = new SqlConnection(konekcioniString);
            /* Kreiramo komandu koja će izvršavati željeni upit nad željenom konekcijom */
            SqlCommand izaberiSve = new SqlCommand(select, konekcija);
            /* Da bi mogli da radimo sa bazom podataka potrebno je da kreiranu konekciju
,,otvorimo'' */
            konekcija.Open();
            /* Promenljiva klase SqlDataReader nam služi za čitanje podataka iz baze
            Koje podatke ćemo čitati zavisi od komande nad kojom pozovemo metodu
ExecuteReader() */
            SqlDataReader citac = izaberiSve.ExecuteReader();
            /* Sprečavamo dupliranje redova, “čišćenjem” ListView-a */
            DrzaveListView.Items.Clear();
            /* Dok čitač prolazi kroz podatke na osnovu prosleđenog upita kod unutar petlje će
se izvršavati */
            while (citac.Read())
            {
                /* Kreira se stavka ListView-a, a u konstruktor se postavlja prva željena
vrednost u tekstualnoj vrednosti */
                ListViewItem prikaz = new ListViewItem(citac[0].ToString());
                /* Na kreiranu stavku se dodaju SubItem-i, onoliko njih koliko podataka želimo
da izvučemo iz upita
                Broj polja ne sme biti veći od broja polja koje vraća upit */
                prikaz.SubItems.Add(citac[1].ToString());
                prikaz.SubItems.Add(citac[2].ToString());
                prikaz.SubItems.Add(citac[3].ToString());
                prikaz.SubItems.Add(citac[4].ToString());
                /* Kreirana stavka sa svojim SubItem-ima se dodaje ListView-u */
                DrzaveListView.Items.Add(prikaz);
            }
            /* Zatvaranje čitača i konekcije */
            citac.Close();
            konekcija.Close();
        }

        private void DrzavaIDTextBox_TextChanged(object sender, EventArgs e)
        {
            /* Metodom Clear() brišemo sve iz navedenih TextBox-ova */
            DrzavaTextBox.Clear();
            BrojStanovnikaTextBox.Clear();
            KontinentIDTextBox.Clear();
            KontinentTextBox.Clear();
            /* Takođe obezbeđujemo da nijedan red u ListView-u ne bude selektovan */
            DrzaveListView.SelectedIndices.Clear();
```

```
            /* foreach petljom prolazimo kroz sve stavke unutar ListView-a */
            foreach (ListViewItem lv in DrzaveListView.Items)
            {
                /* Provera da li se uneti tekst u TextBox DrzavaIDTextBox poklapa sa nekom od
vrednosti polja ListView-a */
                if (DrzavaIDTextBox.Text == lv.Text)
                {
                    /* Ostali TextBox-ovi se popunjavaju SubItem-ima iz ListView-a */
                    DrzavaTextBox.Text = lv.SubItems[1].Text;
                    BrojStanovnikaTextBox.Text = lv.SubItems[2].Text;
                    KontinentIDTextBox.Text = lv.SubItems[3].Text;
                    KontinentTextBox.Text = lv.SubItems[4].Text;
                    /* Selektujemo red u kom se nalazi uneta vrednost za polje DrzavaID */
                    DrzaveListView.SelectedIndices.Add(lv.Index);
                    /* Omogućujemo da se selektovani red prikaže ukoliko je potrebno da se do
njega skroluje */
                    DrzaveListView.Items[lv.Index].EnsureVisible();
                    /* Prekidamo foreach petlju jer smo našli traženi ID */
                    break;
                }
            }
        }

        private void DrzavaIDTextBox_KeyPress(object sender, KeyPressEventArgs e)
        {
            /* Linije koda kojima onemogućujemo unos karaktera u željeni TextBox */
            if (!char.IsDigit(e.KeyChar) && !char.IsControl(e.KeyChar))
            {
                e.Handled = true;
            }
        }
    }
}
```

Umetnost



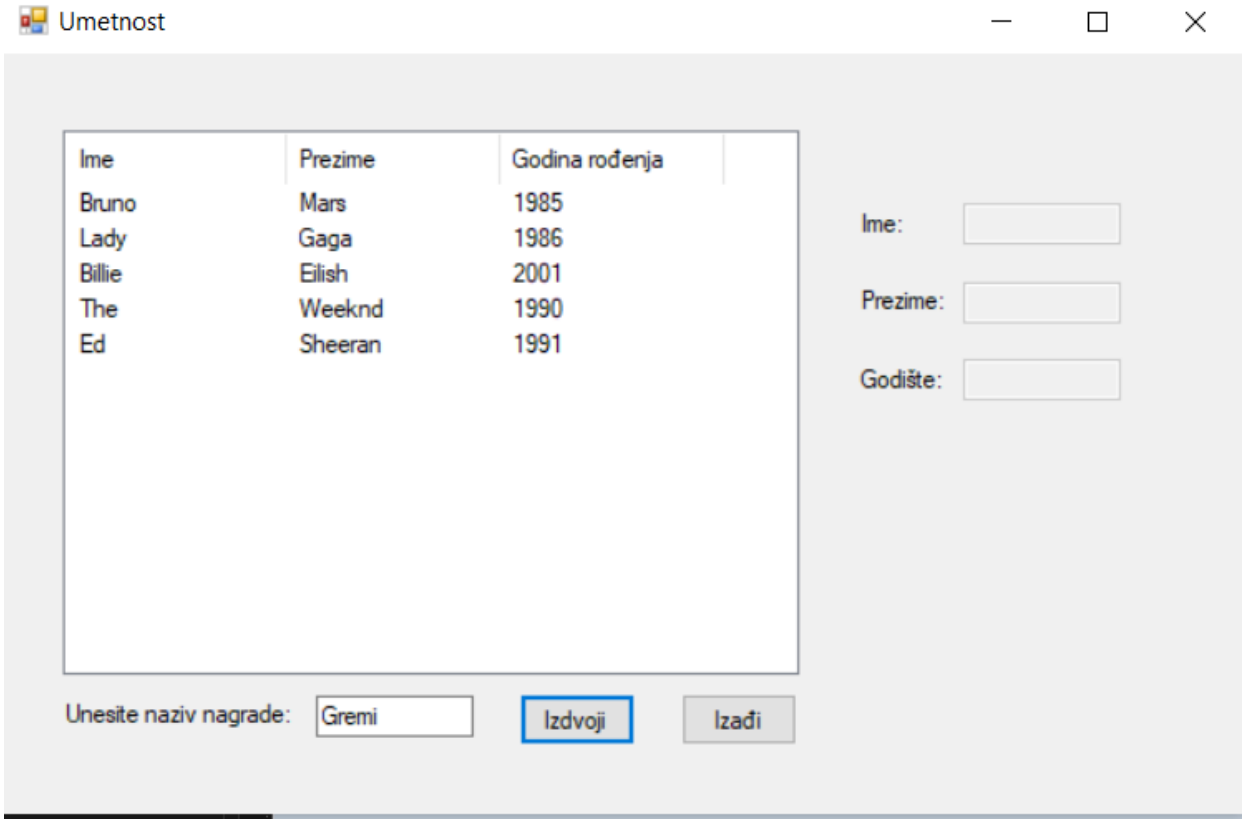
Baza podataka za zadatak: link

Pri pokretanju aplikacije *ListView* se popunjava podacima kao što je prikazano na gornjoj slici.

Jedini *TextBox* u kom je omogućen unos je onaj za unos naziva nagrade.

Selektovanjem bilo kog reda u *ListView*-u, selektuje se ceo red a *TextBox*-ovi sa desne strane se popunjavaju odgovarajućim podacima.

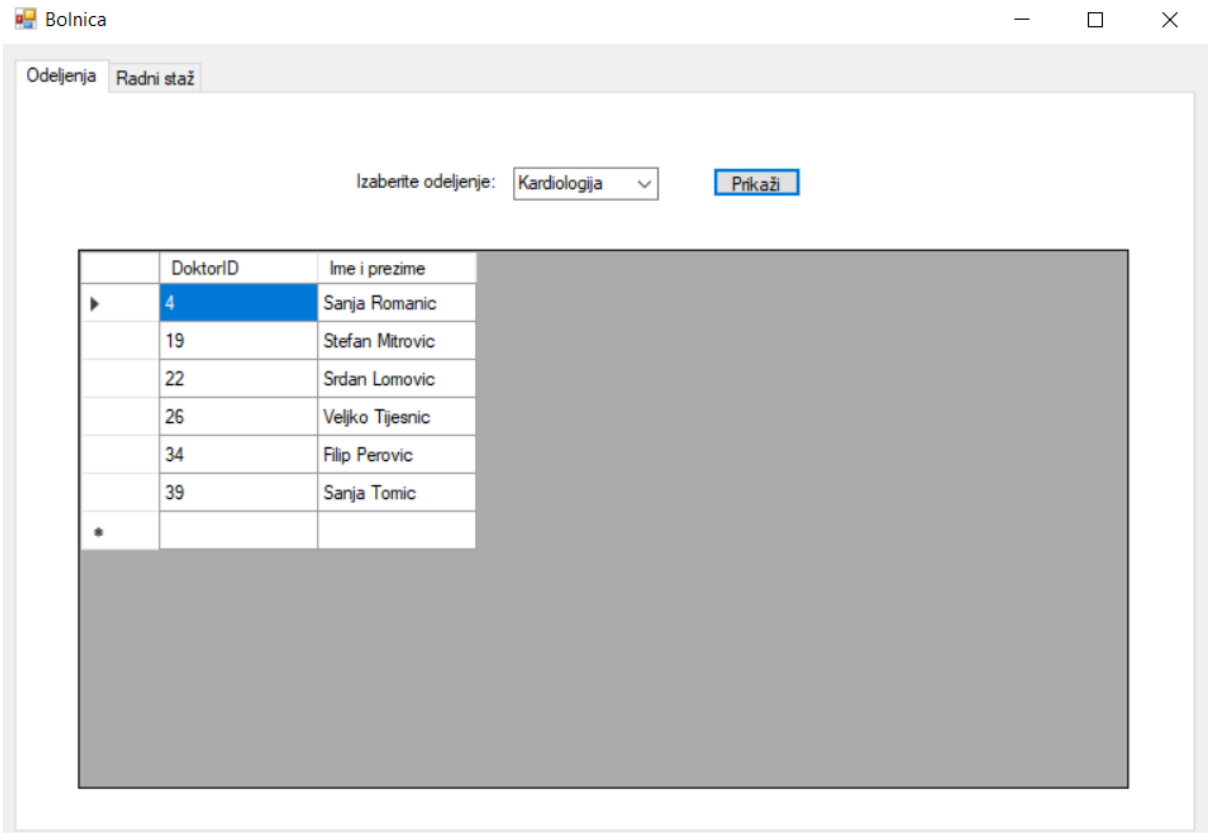
Unosom naziva nagrade i klikom na dugme *Izdvoji* u *ListView*-u ostaju samo podaci onih umetnika koji su osvojili navedenu nagradu.



```
using System;
using System.Data.SqlClient;
using System.Windows.Forms;

namespace Umetnost
{
    public partial class Form1 : Form
    {
        string konekcioniString = @"Data Source=DESKTOP-F7J0CCF\SQLEXPRESS;Initial
Catalog=Umetnost;Integrated Security=True";
        string select = "SELECT Ime, Prezime, YEAR(DatumRodjenja) AS [Godina rođenja] FROM
Umetnik";
        string selectNagrada = "SELECT Ime, Prezime, YEAR(DatumRodjenja) AS GodinaRodjenja
FROM Umetnik JOIN Nagrada ON Umetnik.NagradaID = Nagrada.NagradaID WHERE Naziv = @Naziv;";

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            SqlConnection konekcija = new SqlConnection(konekcioniString);
            SqlCommand izaberi = new SqlCommand(select, konekcija);
            konekcija.Open();
            SqlDataReader citac = izaberi.ExecuteReader();

            while (citac.Read())
            {
                ListViewItem prikaz = new ListViewItem(citac[0].ToString());
                prikaz.SubItems.Add(citac[1].ToString());
                prikaz.SubItems.Add(citac[2].ToString());
                UmetniciListView.Items.Add(prikaz);
            }

            citac.Close();
            konekcija.Close();
        }

        private void UmetniciListView_SelectedIndexChanged(object sender, EventArgs e)
        {
            /* Proveravamo da li je bilo šta iz ListView-a selektovano */
            if (UmetniciListView.SelectedItems.Count > 0)
            {
                /* Preuzimamo podatke iz polja koje je selektovano */
                ListViewItem umetnici = UmetniciListView.SelectedItems[0];
                /* Dodeljujemo vrednosti iz ListView polja TextBox-ovima */
                ImeTextBox.Text = umetnici.SubItems[0].Text;
                PrezimeTextBox.Text = umetnici.SubItems[1].Text;
                GodisteTextBox.Text = umetnici.SubItems[2].Text;
            }
            else
            {
                /* Ukoliko ništa nije selektovano TextBox-ovi se prazne */
                ImeTextBox.Clear();
                PrezimeTextBox.Clear();
```

```
                GodisteTextBox.Clear();
            }
        }


        private void IzdvojiButton_Click(object sender, EventArgs e)
        {
            /* Prvo uklanjamo sve podatke iz ListView-a */
            UmetniciListView.Items.Clear();
            SqlConnection konekcija = new SqlConnection(konekcioniString);
            SqlCommand ispisCitalaca = new SqlCommand(selectNagrada, konekcija);
            /* Dodajemo parametar koji je potreban selectNagrada SQL upitu */
            ispisCitalaca.Parameters.AddWithValue("@Naziv", NagradaTextBox.Text);
            konekcija.Open();

            SqlDataReader citac = ispisCitalaca.ExecuteReader();

            while (citac.Read())
            {
                ListViewItem prikaz = new ListViewItem(citac[0].ToString());
                prikaz.SubItems.Add(citac[1].ToString());
                prikaz.SubItems.Add(citac[2].ToString());
                UmetniciListView.Items.Add(prikaz);
            }

            citac.Close();
            konekcija.Close();
        }

        private void IzadjiButton_Click(object sender, EventArgs e)
        {
            /* Poziv metode koja zatvara aplikaciju */
            Application.Exit();
        }
    }
}
```

Bolnica

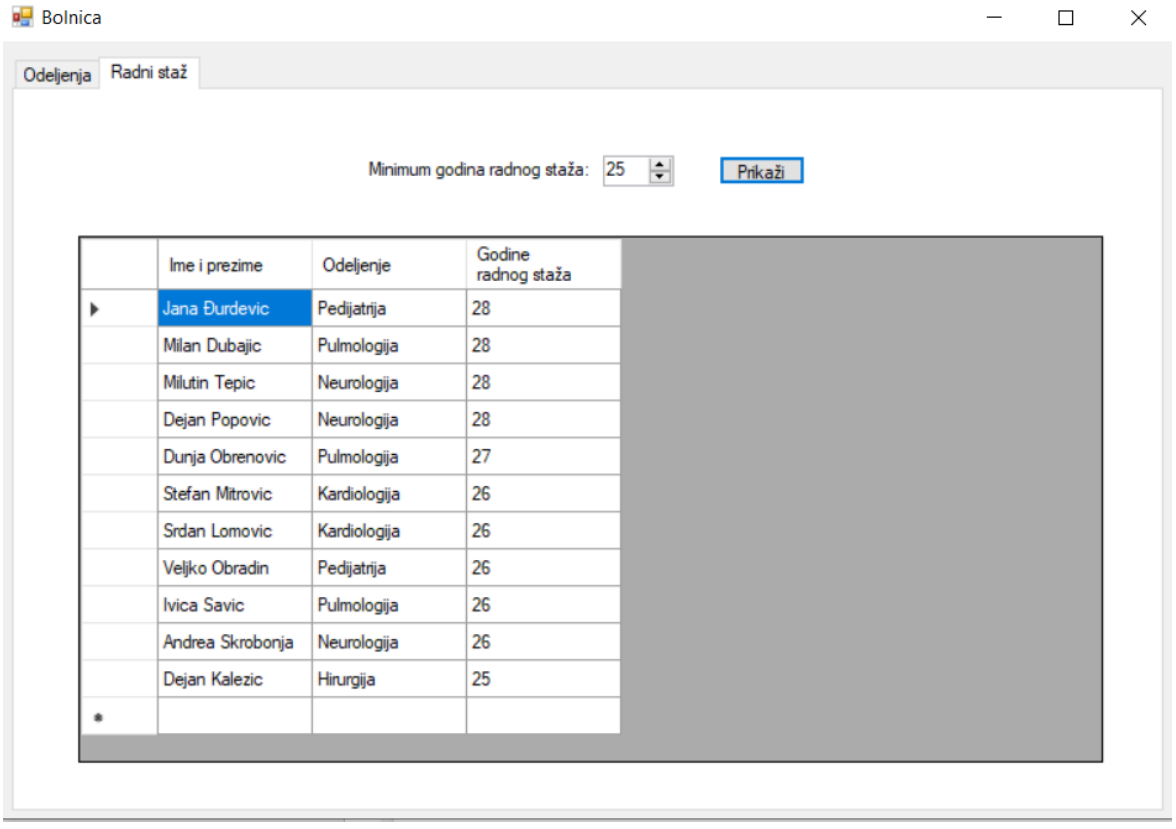


Baza podataka za zadatak: link

Pri pokretanju aplikacije na prvom tabu je *ComboBox* popunjen odeljenjima iz baze podataka.

Klikom na dugme *Prikaži* na prvom tabu *DataGridView* prvog taba se popunjava podacima o doktorima koji pripadaju tom odeljenju.

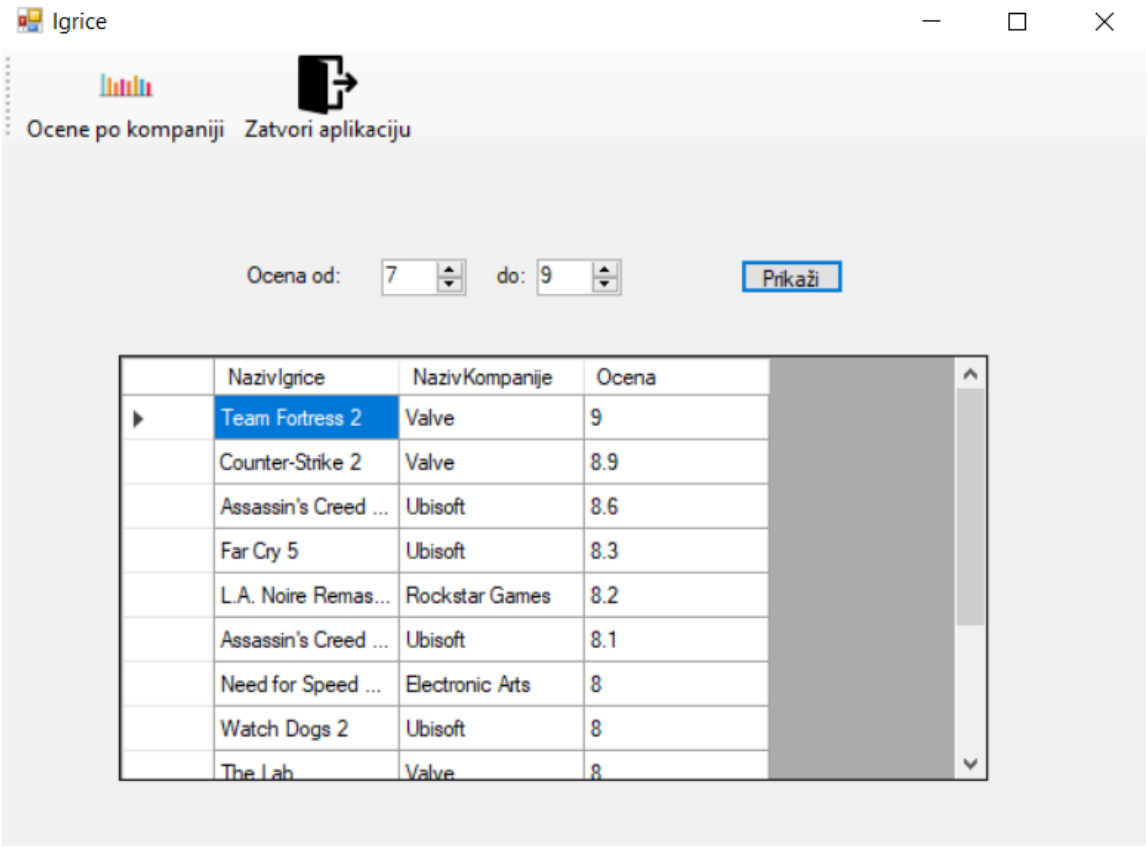
Klikom na dugme *Prikaži* na drugom tabu *DataGridView* drugog taba se popunjava podacima o doktorima koji imaju minimum izabrani broj godina staža.



```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Windows.Forms;

namespace Bolnica
{
    public partial class Form1 : Form
    {
        string konekcioniString = @"Data Source=DESKTOP-F7J0CCF\SQLEXPRESS;Initial
Catalog=Bolnica;Integrated Security=True";
        string insertComboBox = "SELECT OdeljenjeID, Naziv FROM Odeljenje";
        string insertGridViewOdeljenja = "SELECT DoktorID, ImeDoktora + ' ' + PrezimeDoktora "
+
            "AS [Ime i prezime] FROM Doktor WHERE OdeljenjeID = @OdeljenjeID;";
        string insertGridViewStaz = "SELECT ImeDoktora + ' ' + PrezimeDoktora AS [Ime i
prezime], " +
            "Odeljenje.Naziv AS [Odeljenje], RadniStaz AS [Godine radnog staža] " +
            "FROM Doktor INNER JOIN Odeljenje ON Doktor.OdeljenjeID = Odeljenje.OdeljenjeID "
+
            "WHERE RadniStaz >= @RadniStaz ORDER BY RadniStaz DESC;";

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            SqlConnection konekcija = new SqlConnection(konekcioniString);
            /* Kreiramo adapter koji će preko SQL upita izvući željene podatke */
            SqlDataAdapter adapter = new SqlDataAdapter(insertComboBox, konekcija);
            /* Kreiramo DataTable koji preko adaptera popunjavamo podacima iz našeg SQL upita
*/
            DataTable podaci = new DataTable();
            adapter.Fill(podaci);
            /* Podešavamo izvor podataka za ComboBox,
               kao i vrednosnog (Value) odnosno pokaznog (Display) član */
            OdeljenjaComboBox.DataSource = podaci;
            OdeljenjaComboBox.ValueMember = "OdeljenjeID";
            OdeljenjaComboBox.DisplayMember = "Naziv";
        }

        private void PrikaziOdeljenjaButton_Click(object sender, EventArgs e)
        {
            SqlConnection konekcija = new SqlConnection(konekcioniString);
            SqlCommand upis = new SqlCommand(insertGridViewOdeljenja, konekcija);
            upis.Parameters.AddWithValue("@OdeljenjeID",
(int)OdeljenjaComboBox.SelectedValue);
            SqlDataAdapter adapter = new SqlDataAdapter(upis);
            DataTable doktori = new DataTable();
            adapter.Fill(doktori);
            /* Podešavamo izvor podataka za DataGridView */
            OdeljenjaDataGridView.DataSource = doktori;
        }
```

```
        private void PrikaziStazButton_Click(object sender, EventArgs e)
        {
            SqlConnection konekcija = new SqlConnection(konekcioniString);
            SqlCommand upis = new SqlCommand(insertGridViewStaz, konekcija);
            upis.Parameters.AddWithValue("@RadniStaz", (int)StazNumericUpDown.Value);
            SqlDataAdapter adapter = new SqlDataAdapter(upis);
            DataTable doktori = new DataTable();
            adapter.Fill(doktori);
            StazDataGridView.DataSource = doktori;
        }
    }
}
```

Igrice



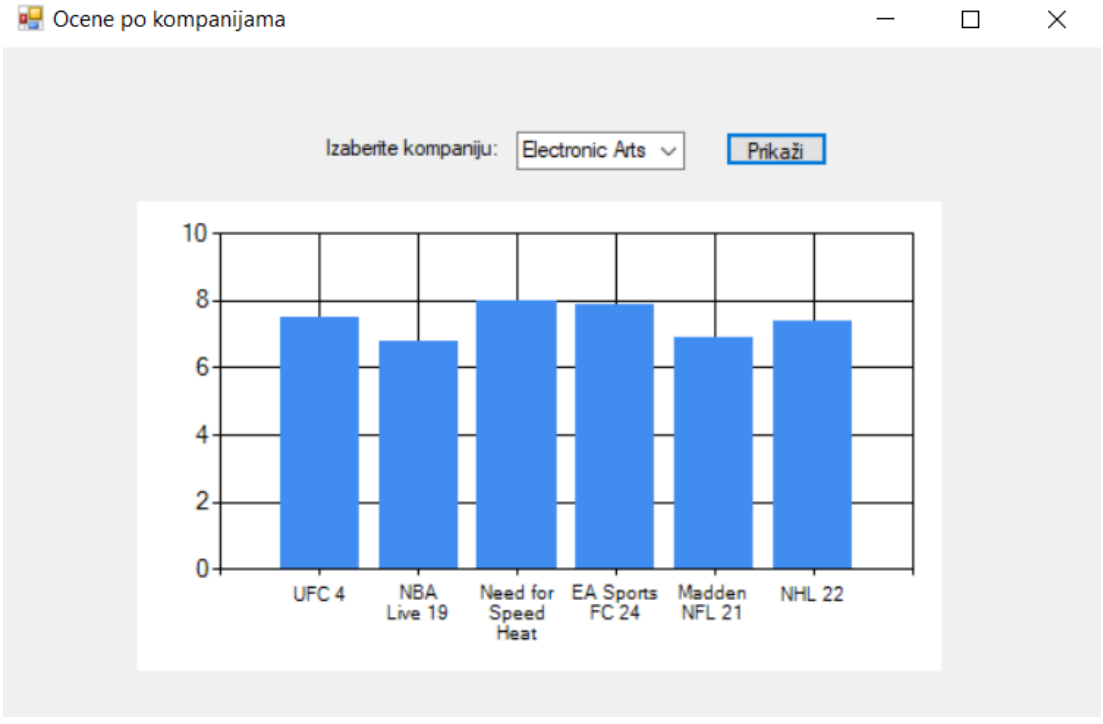
Baza podataka za zadatak: link

Izborom početne i krajnje vrednosti (od i do) i klikom na dugme *Prikaži DataGridView* se popunjava podacima o igricama koje odgovaraju izabranim ocenama.

Klikom na dugme *Ocene po kompaniji* otvara se druga forma, a klikom na dugme *Zatvori aplikaciju* aplikacija se zatvara.

Nakon pokretanja druge forme *ComboBox* na njoj se popunjava podacima o kompanijama iz baze podataka.

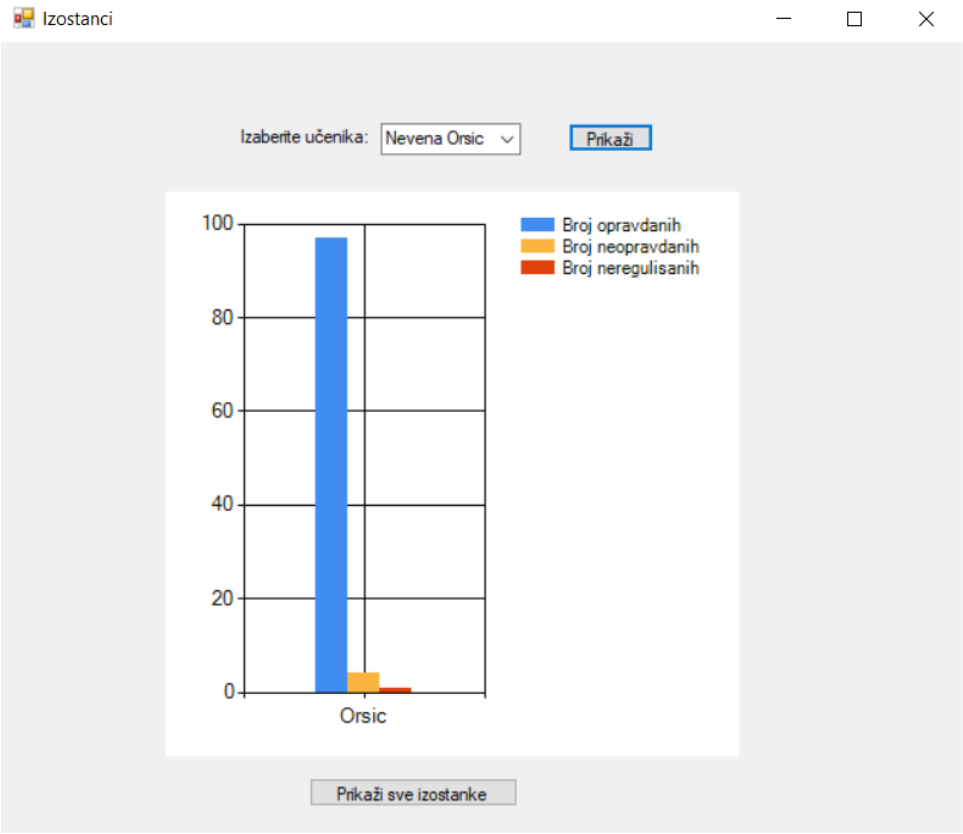
Klikom na dugme *Prikaži* na drugoj formi popunjava se *Chart* sa igricama te kompanije i njihovim ocenama.



```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Windows.Forms;

namespace Igrice
{
    public partial class Form1 : Form
    {
        string konekcioniString = @"Data Source=DESKTOP-F7J0CCF\SQLEXPRESS;Initial
Catalog=Igrice;Integrated Security=True";
        string insertGridView = "SELECT NazivIgrice, NazivKompanije, Ocena " +
        "FROM Igrica INNER JOIN Kompanija ON Igrica.KompanijaID = Kompanija.KompanijaID " +
        "WHERE Ocena >= @Od AND Ocena <= @Do ORDER BY Ocena DESC";

        public Form1()
        {
            InitializeComponent();
        }

        private void PrikaziButton_Click(object sender, EventArgs e)
        {
            SqlConnection konekcija = new SqlConnection(konekcioniString);
            SqlCommand upis = new SqlCommand(insertGridView, konekcija);

            if (OdNumericUpDown.Value > DoNumericUpDown.Value)
            {
               MessageBox.Show("Vrednost Od ne može biti veća od Do");
               return;
            }

            upis.Parameters.AddWithValue("@Od", OdNumericUpDown.Value);
            upis.Parameters.AddWithValue("@Do", DoNumericUpDown.Value);

            SqlDataAdapter dataAdapter = new SqlDataAdapter(upis);

            DataTable podaci = new DataTable();
            dataAdapter.Fill(podaci);
            IgriceDataGridView.DataSource = podaci;
        }

        private void OceneToolStripButton_Click(object sender, EventArgs e)
        {
            /* Linije koda za otvaranje nove forme */
            Form2 forma = new Form2();
            forma.Show();
        }

        private void ZatvoriToolStripButton_Click(object sender, EventArgs e)
        {
            /* Još jedan način za zatvaranje prozora */
            Close();
        }
    }
}
```

```
using System;
using System.Data.SqlClient;
using System.Data;
using System.Windows.Forms;

namespace Igrice
{
    public partial class Form2 : Form
    {
        string konekcioniString = @"Data Source=DESKTOP-F7J0CCF\SQLEXPRESS;Initial
Catalog=Igrice;Integrated Security=True";
        string insertComboBox = "SELECT KompanijaID, NazivKompanije FROM Kompanija";
        string insertChart = "SELECT NazivIgrice, Ocena FROM Igrica WHERE KompanijaID =
@KompanijaID";

        public Form2()
        {
            InitializeComponent();
        }

        private void Form2_Load(object sender, EventArgs e)
        {
            SqlConnection konekcija = new SqlConnection(konekcioniString);
            SqlDataAdapter adapter = new SqlDataAdapter(insertComboBox, konekcija);
            DataTable podaci = new DataTable();
            adapter.Fill(podaci);
            KompanijeComboBox.DataSource = podaci;
            KompanijeComboBox.ValueMember = "KompanijaID";
            KompanijeComboBox.DisplayMember = "NazivKompanije";
        }

        private void GrafikButton_Click(object sender, EventArgs e)
        {
            SqlConnection konekcija = new SqlConnection(konekcioniString);
            SqlCommand upis = new SqlCommand(insertChart, konekcija);
            upis.Parameters.AddWithValue("@KompanijaID",
(int)KompanijeComboBox.SelectedValue);
            SqlDataAdapter adapter = new SqlDataAdapter(upis);
            /* Popunjavanje grafika (Chart-a) željenim podacima */
            DataTable podaciGrafik = new DataTable();
            adapter.Fill(podaciGrafik);
            OceneChart.DataSource = podaciGrafik;

            OceneChart.Series[0].XValueMember = "NazivIgrice";
            OceneChart.Series[0].YValueMembers = "Ocena";
            OceneChart.DataBind();
        }
    }
}
```

Izostanci

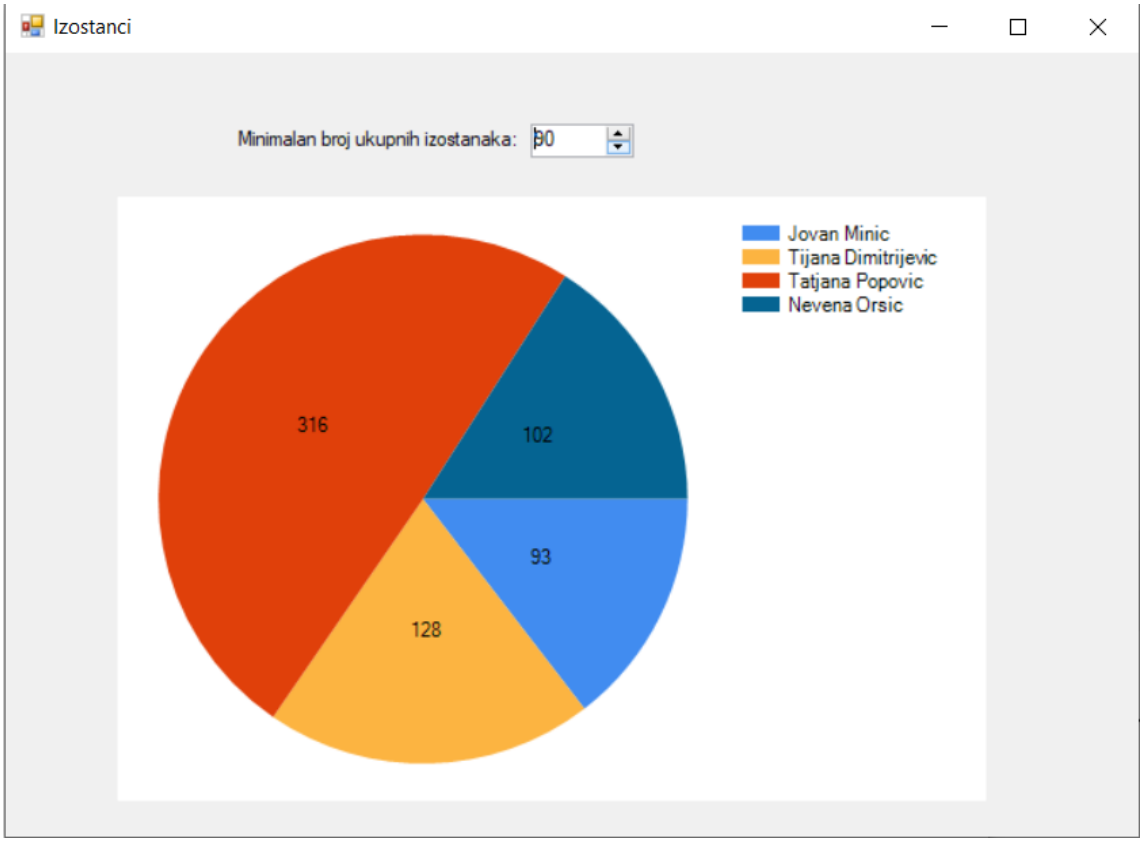


Baza podataka za zadatak: link

Nakon pokretanja aplikacije *ComboBox* se popunjava imenima i prezimenima učenika iz baze podataka.

Nakon izbora učenika i klika na dugme *Prikaži* popunjava se *Chart* sa podacima o broju opravdanih, neopravdanih i neregulisanih časova tog učenika.

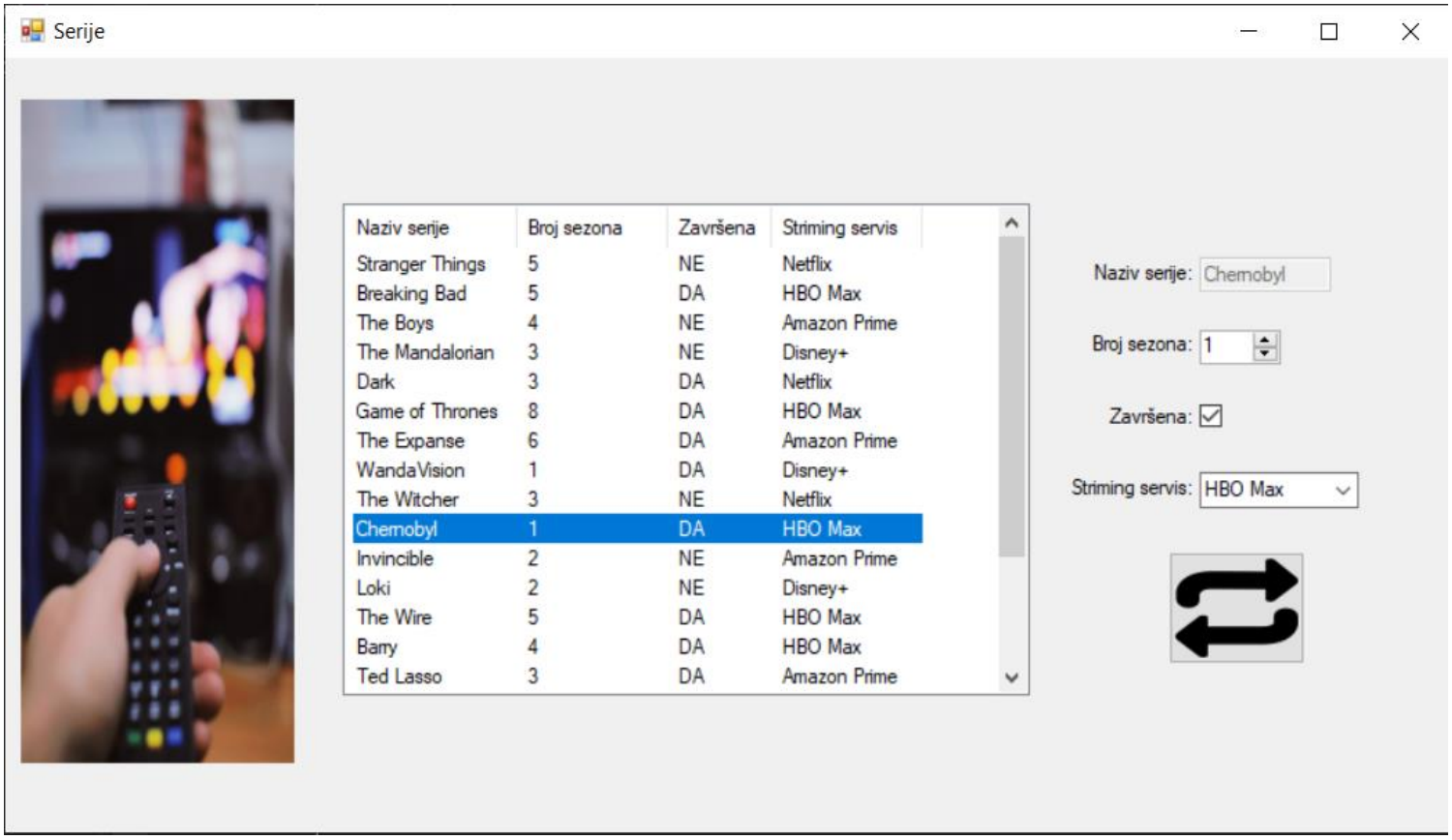
Klikom na dugme *Prikaži sve izostanke* otvara se druga forma na kojoj se *Chart* popunjava nakon promene vrednosti u *NumericUpDown*-u.



```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Windows.Forms;

namespace Izostanci
{
    public partial class Form1 : Form
    {
        string konekcioniString = @"Data Source=DESKTOP-F7J0CCF\SQLEXPRESS;Initial
Catalog=Izostanci;Integrated Security=True";
        string insertComboBox = "SELECT UcenikID, Ime + ' ' + Prezime AS PunoIme FROM Ucenik";
        string insertColumnChart = "SELECT BrojOpravdanih, BrojNeopravdanih,
BrojNeregulisanih, Prezime FROM Ucenik " +
        "WHERE UcenikID = @UcenikID";

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            SqlConnection konekcija = new SqlConnection(konekcioniString);
            SqlDataAdapter adapter = new SqlDataAdapter(insertComboBox, konekcija);
            DataTable podaci = new DataTable();
            adapter.Fill(podaci);
            UceniciComboBox.DataSource = podaci;
            UceniciComboBox.ValueMember = "UcenikID";
            UceniciComboBox.DisplayMember = "PunoIme";
        }

        private void PrikaziColumnButton_Click(object sender, EventArgs e)
        {
            SqlConnection konekcija = new SqlConnection(konekcioniString);
            SqlCommand upis = new SqlCommand(insertColumnChart, konekcija);
            upis.Parameters.AddWithValue("@UcenikID",
UceniciComboBox.SelectedValue.ToString());
            SqlDataAdapter adapter = new SqlDataAdapter(upis);
            /* Popunjavanje grafika (Chart-a) sa više kolona (Series) */
            DataTable podaciGrafik = new DataTable();
            adapter.Fill(podaciGrafik);
            IzostanciColumnChart.DataSource = podaciGrafik;

            IzostanciColumnChart.Series[0].YValueMembers = "BrojOpravdanih";
            IzostanciColumnChart.Series[1].YValueMembers = "BrojNeopravdanih";
            IzostanciColumnChart.Series[2].YValueMembers = "BrojNeregulisanih";
            IzostanciColumnChart.Series[0].XValueMember = "Prezime";
            IzostanciColumnChart.DataBind();
        }

        private void SviIzostanciButton_Click(object sender, EventArgs e)
        {
            Form2 forma = new Form2();
            forma.Show();
        } } }
```

```
using System;
using System.Data.SqlClient;
using System.Windows.Forms;

namespace Izostanci
{
    public partial class Form2 : Form
    {
        string konekcioniString = @"Data Source=DESKTOP-F7J0CCF\SQLEXPRESS;Initial
Catalog=Izostanci;Integrated Security=True";
        string insertPieChart = "SELECT Ime + ' ' + Prezime AS [Ucenik], (BrojOpravdanih +
BrojNeopravdanih + BrojNeregulisanih) AS [Ukupno izostanaka] FROM Ucenik WHERE (BrojOpravdanih
+ BrojNeopravdanih + BrojNeregulisanih) >= @Minimum;";

        public Form2()
        {
            InitializeComponent();
        }

        private void IzostanciNumericUpDown_ValueChanged(object sender, EventArgs e)
        {
            SqlConnection konekcija = new SqlConnection(konekcioniString);
            SqlCommand upis = new SqlCommand(insertPieChart, konekcija);

            upis.Parameters.AddWithValue("@Minimum", (int)IzostanciNumericUpDown.Value);

            SqlDataAdapter adapter = new SqlDataAdapter(upis);

            DataTable podaciGrafik = new DataTable();
            adapter.Fill(podaciGrafik);
            IzostanciPieChart.DataSource = podaciGrafik;

            IzostanciPieChart.Series[0].XValueMember = "Ucenik";
            IzostanciPieChart.Series[0].YValueMembers = "Ukupno izostanaka";
            IzostanciPieChart.DataBind();
        }
    }
}
```

Serije



Baza podataka za zadatak: link

Nakon pokretanja aplikacije *ListView* se popunjava podacima o serijama, preuzetim iz baze podataka, isto kao i *ComboBox* koji se popunjava podacima o striming servisima iz baze podataka.

Nakon selektovanja serije u *ListView*-u selektuje se ceo red a kontrole sa desne strane se popunjavaju odgovarajućim podacima. *TextBox* sa nazivom serije je onemogućen za unos/izmenu.

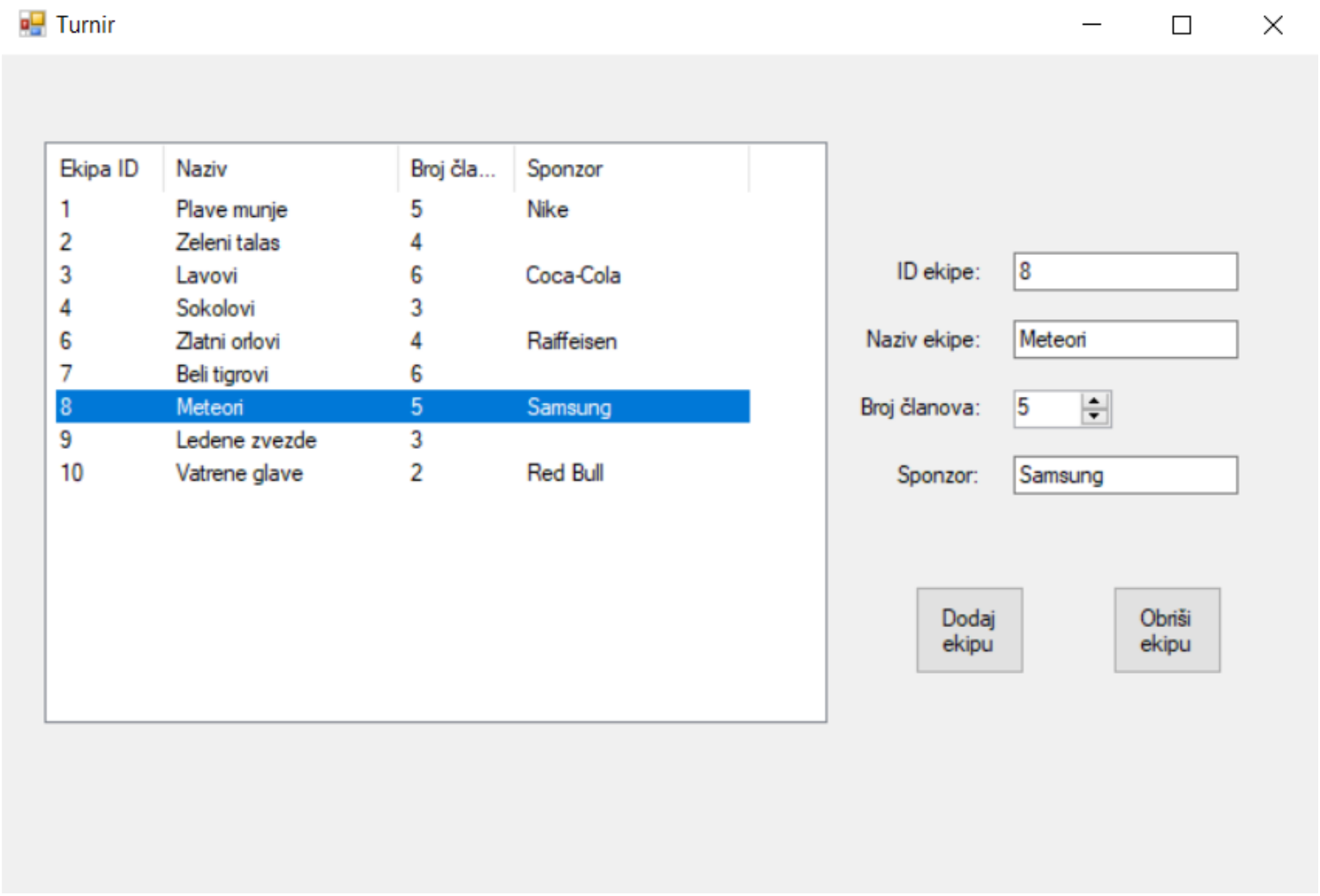
Klikom na jedino dugme eventualne izmene se unose u bazu podataka, a *ListView* se osvežava (refresh-uje) i u njemu se prikazuju podaci nakon izvršene izmene.

```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Windows.Forms;

namespace Serije
{
    public partial class Form1 : Form
    {
        string konekcioniString = "Data Source=DESKTOP-F7J0CCF\\SQLEXPRESS;Initial
Catalog=Serije;Integrated Security=True;Encrypt=False";
        string select = "SELECT SerijaID, Serija.NazivSerije, Serija.BrojSezona,
Serija.Zavrsena, StrimingServis.NazivServisa FROM Serija INNER JOIN StrimingServis ON
Serija.StrimingServisID = StrimingServis.StrimingServisID;";
        string insertComboBox = "SELECT StrimingServisID, NazivServisa FROM StrimingServis;";
        string update = "UPDATE Serija SET BrojSezona = @BrojSezona, Zavrsena = @Zavrsena,
StrimingServisID = @StrimingServisID WHERE SerijaID = @SerijaID;";

        public Form1()
        {
            InitializeComponent();
```

```
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            SqlConnection konekcija = new SqlConnection(konekcioniString);
            SqlCommand izaberi = new SqlCommand(select, konekcija);
            konekcija.Open();
            SqlDataReader citac = izaberi.ExecuteReader();
            while (citac.Read())
            {
                ListViewItem prikaz = new ListViewItem(citac[1].ToString());
                prikaz.SubItems.Add(citac[2].ToString());
                prikaz.SubItems.Add(citac[3].ToString());
                prikaz.SubItems.Add(citac[4].ToString());
                prikaz.Tag = (int)citac[0];
                SerijeListView.Items.Add(prikaz);
            }
            citac.Close();

            SqlDataAdapter adapter = new SqlDataAdapter(insertComboBox, konekcija);

            DataTable podaci = new DataTable();
            adapter.Fill(podaci);

            StrimingComboBox.DataSource = podaci;
            StrimingComboBox.ValueMember = "StrimingServisID";
            StrimingComboBox.DisplayMember = "NazivServisa";

            konekcija.Close();
        }

        private void SerijeListView_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (SerijeListView.SelectedItems.Count > 0)
            {
                ListViewItem serije = SerijeListView.SelectedItems[0];
                SerijaTextBox.Text = serije.SubItems[0].Text;
                SezoneNumericUpDown.Value = int.Parse(serije.SubItems[1].Text);
                if (serije.SubItems[2].Text == "DA") ZavrsenaCheckBox.Checked = true;
                else ZavrsenaCheckBox.Checked = false;
                StrimingComboBox.Text = serije.SubItems[3].Text;

            }
        }

        private void IzmeniButton_Click(object sender, EventArgs e)
        {
            int serijaID = (int)SerijeListView.SelectedItems[0].Tag;
            SqlConnection konekcija = new SqlConnection(konekcioniString);
            SqlCommand promeni = new SqlCommand(update, konekcija);
            promeni.Parameters.AddWithValue("@StrimingServisID",
StrimingComboBox.SelectedValue);
            promeni.Parameters.AddWithValue("@BrojSezona", (int)SezoneNumericUpDown.Value);
            if (ZavrsenaCheckBox.Checked == true) promeni.Parameters.AddWithValue("@Zavrsena",
"DA");
            else promeni.Parameters.AddWithValue("@Zavrsena", "NE");
```

```
            promeni.Parameters.AddWithValue("@SerijaID", serijaID);
            konekcija.Open();
            int i = promeni.ExecuteNonQuery();
            konekcija.Close();
            /* Provera da li je upit izvršen */
            if (i != 0)
            {
                /* Kod koji se izvršava ako je upit izvršen */
                MessageBox.Show("Podaci su uspešno izmenjeni");
                SqlCommand izaberi = new SqlCommand(select, konekcija);
                SerijeListView.Items.Clear();
                konekcija.Open();
                SqlDataReader citac = izaberi.ExecuteReader();
                while (citac.Read())
                {
                    ListViewItem prikaz = new ListViewItem(citac[1].ToString());
                    prikaz.SubItems.Add(citac[2].ToString());
                    prikaz.SubItems.Add(citac[3].ToString());
                    prikaz.SubItems.Add(citac[4].ToString());
                    prikaz.Tag = (int)citac[0];
                    SerijeListView.Items.Add(prikaz);
                }
                citac.Close();
                konekcija.Close();
            }
        }
    }
}
```

Turnir



Baza podataka za zadatak: link

Nakon pokretanja aplikacije *ListView* se popunjava podacima iz baze podataka.

Klikom na bilo koji red u *ListView*-u selektuje se ceo red a odgovarajući podaci se prikazuju u kontrolama sa desne strane. Unosom broja u *TextBox* za ID ekipe selektuje se red u *ListView*-u koji odgovara tom ID-ju (onemogućen je unos slova i simbola dozvoljene su samo cifre u tom *TextBox*-u).

Klikom na dugme *Dodaj ekipu* dodaje se nova ekipa u bazu podataka (sva polja sem sponzora su obavezna), a *ListView* se osvežava i prikazuje ažurne podatke.

Klikom na dugme *Obriši ekipu* selektovana ekipa se briše iz baze podataka, a *ListView* se osvežava i prikazuje ažurne podatke.

```
using System;
using System.Data.SqlClient;
using System.Windows.Forms;

namespace Turnir
{
    public partial class Form1 : Form
    {
        string konekcioniString = "Data Source=DESKTOP-F7J0CCF\\SQLEXPRESS;Initial
Catalog=Turnir;Integrated Security=True;Encrypt=False";
        string select = "SELECT EkipaID, Naziv, BrojClanova, Sponzor FROM Ekipa;";
        string insert = "INSERT INTO Ekipa (EkipaID, Naziv, BrojClanova, Sponzor) VALUES (@ID,
@Naziv, @BrojClanova, @Sponzor)";
```

```
        string delete = "DELETE FROM Ekipa WHERE EkipaID = @ID";

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            SqlConnection konekcija = new SqlConnection(konekcioniString);
            SqlCommand ispisEkipa = new SqlCommand(select, konekcija);
            konekcija.Open();

            SqlDataReader citac = ispisEkipa.ExecuteReader();

            while (citac.Read())
            {
                ListViewItem prikaz = new ListViewItem(citac[0].ToString());
                prikaz.SubItems.Add(citac[1].ToString());
                prikaz.SubItems.Add(citac[2].ToString());
                prikaz.SubItems.Add(citac[3].ToString());
                EkipeListView.Items.Add(prikaz);
            }

            citac.Close();
            konekcija.Close();
        }

        private void EkipeListView_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (EkipeListView.SelectedItems.Count > 0)
            {
                ListViewItem item = EkipeListView.SelectedItems[0];
                EkipaIDTextBox.Text = item.SubItems[0].Text;
                NazivTextBox.Text = item.SubItems[1].Text;
                ClanoviNumericUpDown.Value = int.Parse(item.SubItems[2].Text);
                SponzorTextBox.Text = item.SubItems[3].Text;
            }
            else
            {
                NazivTextBox.Clear();
                ClanoviNumericUpDown.Value = 2;
                SponzorTextBox.Clear();
            }
        }

        private void EkipaIDTextBox_TextChanged(object sender, EventArgs e)
        {
            NazivTextBox.Clear();
            ClanoviNumericUpDown.Value = 2;
            SponzorTextBox.Clear();
            foreach (ListViewItem lv in EkipeListView.Items)
            {
                if (EkipaIDTextBox.Text == lv.Text)
                {
                    NazivTextBox.Text = lv.SubItems[1].Text;
```

```
                ClanoviNumericUpDown.Value = int.Parse(lv.SubItems[2].Text);
                SponzorTextBox.Text = lv.SubItems[3].Text;
                EkipeListView.SelectedIndices.Clear();
                EkipeListView.SelectedIndices.Add(lv.Index);
                EkipeListView.Items[lv.Index].EnsureVisible();
                break;
            }
        }
    }

    private void EkipaIDTextBox_KeyPress(object sender, KeyPressEventArgs e)
    {
        if (!char.IsDigit(e.KeyChar) && !char.IsControl(e.KeyChar))
        {
            e.Handled = true;
        }
    }

    private void DodajButton_Click(object sender, EventArgs e)
    {
if (EkipaIDTextBox.Text == "" || NazivTextBox.Text == "")
{
    MessageBox.Show("Sva polja (sem sponzora) su obavezna!");
    return;
}

try
{
    SqlConnection konekcija = new SqlConnection(konekcioniString);
    SqlCommand dodaj = new SqlCommand(insert, konekcija);

    dodaj.Parameters.AddWithValue("@ID", int.Parse(EkipaIDTextBox.Text));
    dodaj.Parameters.AddWithValue("@Naziv", NazivTextBox.Text);
    dodaj.Parameters.AddWithValue("@BrojClanova", (int)ClanoviNumericUpDown.Value);
    dodaj.Parameters.AddWithValue("@Sponzor", SponzorTextBox.Text);

    konekcija.Open();
    int i = dodaj.ExecuteNonQuery();
    konekcija.Close();

    if (i != 0)
    {
        MessageBox.Show("Podaci su uspešno dodati");

        SqlCommand izaberi = new SqlCommand(select, konekcija);
        EkipeListView.Items.Clear();

        konekcija.Open();
        SqlDataReader citac = izaberi.ExecuteReader();
        while (citac.Read())
        {
            ListViewItem prikaz = new ListViewItem(citac[0].ToString());
            prikaz.SubItems.Add(citac[1].ToString());
            prikaz.SubItems.Add(citac[2].ToString());
            prikaz.SubItems.Add(citac[3].ToString());
            EkipeListView.Items.Add(prikaz);
```

```
            }
            citac.Close();
            konekcija.Close();

            foreach (ListViewItem lv in EkipeListView.Items)
            {
                if (EkipaIDTextBox.Text == lv.Text)
                {
                    lv.Selected = true;
                    lv.EnsureVisible();
                    break;
                }
            }
        }
    }
    catch (FormatException)
    {
        MessageBox.Show("ID mora biti broj.");
    }
    catch (SqlException)
    {
        MessageBox.Show("Ekipa sa tim ID-jem već postoji ili je došlo do greške u bazi.");
    }

        private void ObrisiButton_Click(object sender, EventArgs e)
        {
            int i = 0;
            SqlConnection konekcija = new SqlConnection(konekcioniString);
            SqlCommand upis = new SqlCommand(delete, konekcija);
            SqlCommand ispisCitalaca = new SqlCommand(select, konekcija);
            upis.Parameters.AddWithValue("@ID", int.Parse(EkipaIDTextBox.Text));
            try
            {
                konekcija.Open();
                i = upis.ExecuteNonQuery();
                konekcija.Close();
            }
            catch (SqlException sql)
            {
                MessageBox.Show("Došlo je do greške: " + sql.Message);
                konekcija.Close();
            }
            if (i != 0)
            {
                konekcija.Open();
                MessageBox.Show("Podaci su izbrisani!");
                EkipaIDTextBox.Clear();
                NazivTextBox.Clear();
                ClanoviNumericUpDown.Value = 2;
                SponzorTextBox.Clear();
                EkipeListView.Items.Clear();
                SqlDataReader citac = ispisCitalaca.ExecuteReader();
                while (citac.Read())
                {
                    ListViewItem lv = new ListViewItem(citac[0].ToString());
                    lv.SubItems.Add(citac[1].ToString());
```

```
                    lv.SubItems.Add(citac[2].ToString());
                    lv.SubItems.Add(citac[3].ToString());
                    EkipeListView.Items.Add(lv);
                }
                citac.Close();
                konekcija.Close();
            }
        }
    }
}
```