***Klasa*Repository Metode**

1. **Metoda za čitanje iz tabele**

public List<**Auto**> GetAutos()

{

\*List**<Auto>** **list** = new List<**Auto**>();

\*string cons = "**Data Source=(localdb)\\MSSQLLocalD**[**B;Initial Catalog=Auto**](https://ftncacak-my.sharepoint.com/personal/68-2018_ftn_edu_rs/Documents/B;Initial%20Catalog=Auto)**DB;Integrated Security=True;Connect Timeout=30;Encrypt=False;TrustServerCertificate=False;ApplicationIntent=ReadWrite;MultiSubnetFailover=False**"; *-*

SqlConnection con = new SqlConnection(cons);

con.Open();

\*string command = "Select \* FROM **Automobili**";

SqlCommand com = new SqlCommand(command, con);

SqlDataReader dr = com.ExecuteReader();

while (dr.Read())

{

\*list.Add(new Auto(**dr.GetString(1), dr.GetString(2), dr.GetInt32(3),dr.GetString(4)**));

}

return **list**;

}

Umesto <**Auto**> piše se <*ImeKlase*>

\*u string con = “ovde” ovde – konekcioni link ka **BAZI** (ne ka tabeli), on se nalazi kad se klikne na bazu pa dole desno ispod Properties **piše connection string**

\*unutar “Select \* FROM **Automobili**” - umesto Automobili piše se ime tabele, voditi računa o malom i velikom slovu!

\*kod list.Add(new **Auto**(**dr.GetString(1), dr.GetString(2), dr.GetInt32(3),dr.GetString(4)**));

Umesto **Auto** pišemo ime naše klase, kao kod kreiranja liste, unutar zagrada stavljamo Get**Tip(br. kolone)** – brojevi kolone kreću od 1 jer je ID broj 0.

1. **Metoda za unos u tabelu**

public int Insert**Auto**(**Auto a**)

{

int result;

string cons = "**Data Source=(localdb)**[**\\MSSQLLocalDB;Initial**](file:///\\MSSQLLocalDB;Initial) **Catalog=AutoDB;Integrated Security=True;Connect Timeout=30;Encrypt=False;TrustServerCertificate=False;ApplicationIntent=ReadWrite;MultiSubnetFailover=False**";

using (SqlConnection con = new SqlConnection(cons))

{

con.Open();

string command = "Insert Into Automobili (**Marka,Model,Godiste,Boja) VALUES (@Marka,@Model,@Godiste,@Boja**)";

SqlCommand com = new SqlCommand(command, con);

com.Parameters.AddWithValue("**@Marka", a.marka**);

com.Parameters.AddWithValue("**@Model", a.model**);

com.Parameters.AddWithValue("**@Godiste", a.godiste**);

com.Parameters.AddWithValue("**@Boja", a.boja**);

**\*\***result = com.ExecuteNonQuery();

}

return result;

}

Prvo u metodu šaljemo objekat tipa koji odgovara podatku iz tabele, tj. objekat klase koja se nalazi u **Model** folderu

Kod stringa cons, ponovo stavljamo link ka **bazi** podataka.

kod komandnog stringa, u ovom slučaju command ide text “Insert Into **IMETABELE (tip1,tip2,tip3…)** VALUES **(@tip1,@tip2,@tip3…)**”;

kod linije sa **- com.Parameters.AddWithValue(“@tipRedniBroj”, slovo.tipRedniBroj);**

linija com.Parameters…. se piše onliko puta koliko tipova imamo.

Na kraju **obavezno - return result;**

**3. Metoda za brisanje iz tabele**

public int Delete**Student**(int **sid**)

{

int result;

string constring = "**Data Source=(localdb)\\MSSQLLocalDB;Initial Catalog=FacultyDB;Integrated Security=True;Connect Timeout=30;Encrypt=False;TrustServerCertificate=False;ApplicationIntent=ReadWrite;MultiSubnetFailover=False**";

using (SqlConnection con = new SqlConnection(constring))

{

con.Open();

string command = "Delete From **Students** where Id=@Id";

SqlCommand com = new SqlCommand(command, con);

com.Parameters.AddWithValue("@Id", **sid**);

**\***result = com.ExecuteNonQuery();

}

return result;

}

}

U ovoj metodi se vrši brisanje iz tabele na osnovu prosleđenog ID-a. On se prosleđuje unutar - public int DeleteStudent(**int sid**).

Ponovo se prosleđuje konekcioni string ka **BAZI.**

Unutar komande se piše - “Delete From **ImeTabele** Id=@Id”;

kod com.Parameters.AddWithValue(“ID”, **sid**); sid – je prosleđeni ID tog člana tabele koji se briše.

Na kraju se vraća result.