SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

VARAŽDIN

Nikola Švenda

Izvanredni student

Broj indeksa: 000829

Smjer: Primjena informacijske tehnologije u poslovanju

Stručni preddiplomski studij

**Primjer baze podataka u sustavu PostgreSQL**

**ZAVRŠNI RAD**

Mentor:

Doc. Dr. Sc. Markus Schatten

Varaždin, lipanj 2014.

[1.UVOD 3](#_Toc398221219)

[2. KORIŠTENA TEHNOLOGIJA 4](#_Toc398221220)

[2.1. PostgreSQL 4](#_Toc398221221)

[2.2. Microsoft C# programski jezik 4](#_Toc398221222)

[2.3. SQL jezik 4](#_Toc398221223)

[2.3.1. Naredbe u SQL-u 5](#_Toc398221224)

[2.3.2. Operatori u SQL-u 6](#_Toc398221225)

[2.3.3. Matematičke funkcije u SQL-u 6](#_Toc398221226)

[3. OPIS APLIKACIJSKE DOMENE 7](#_Toc398221227)

[3.1. ERA model 7](#_Toc398221228)

[4. IMPLEMENTACIJA 10](#_Toc398221229)

[4.1. Baza Podataka 10](#_Toc398221230)

[4.2. Djelovi koda 12](#_Toc398221231)

[5.PRIMJER KORIŠTENJA 17](#_Toc398221232)

[5.1. Glavni Ekran 17](#_Toc398221233)

[5.2. Baza dijelova 18](#_Toc398221234)

[5.3. Grupa 19](#_Toc398221235)

[5.4. Baza automobila 20](#_Toc398221236)

[5.5. Baza stranaka 21](#_Toc398221237)

[5.6. Vozila stranke 23](#_Toc398221238)

[5.7. Popis svih naloga 24](#_Toc398221239)

[5.7.1. Ispis naloga 25](#_Toc398221240)

[5.7.2 Ispis računa 25](#_Toc398221241)

[5.8. Serviseri 26](#_Toc398221242)

[5.9. Novi radni nalog 27](#_Toc398221243)

[6. KRITIČKI OSVRT 29](#_Toc398221244)

[7. ZAKLJUČAK 30](#_Toc398221245)

[8. LITERATURA 31](#_Toc398221246)

[9. PRILOG 32](#_Toc398221247)

[Forma „Baza\_auta“ 32](#_Toc398221248)

[Forma „Grupa“ 52](#_Toc398221249)

[Forma „Novi\_nalog“ 53](#_Toc398221250)

[Forma „Serviseri“ 61](#_Toc398221251)

[Forma „svi\_nalozi“ 62](#_Toc398221252)

[Forma „vozila“ 64](#_Toc398221253)

[Forma „Nalog\_frm“ 66](#_Toc398221254)

1.UVOD

Tokom traženja teme završnog rada želio sam se približiti temi koja me najviše interesira, a to su baze podataka. Kako sam tokom studia dolazio u susret sa različitim bazama podataka, ova tema me zainteresirala jer stavlja bazu podataka u praktičnu upotrebu pomoću aplikacije.

Samim time što se tema mog rada bazira na PostgreSQL-u opisat ću teoriju baze podataka te koda kojeg sam koristio. Nakon toga opisat ću aplikacijsku domenu odnosno područje djelovanja moje aplikacije. Potom slijedi ERA model baze podataka koji grafički prikazuje entitete i veze u bazi podataka. Zatim ću opisati implementaciju baze podataka te njeno korištenje na primjeru. Za lakše razumjevanje bit će prikazano pomoću slika. Također ću prokomentirati bazu podataka i kritički se osvrnuti. Na kraju ovog rada prilažem cijeloviti kod aplikacije.

2. KORIŠTENA TEHNOLOGIJA

2.1. PostgreSQL

PostgreSQL je generalno objektno-relacijski sustav za upravljanje bazama podataka, najnapredniji sustav baze podataka otvorenog koda. PostgreSQL se bazira na POSTGRES 4.2 koji je razvijen na Berkley-evoj katedri za računalne znanosti, Sveučilište Kalifornija.

PostgreSQL je dizajniran za rad na UNIX platformama. Dizajniran je tako da bude potpuno prenosiv te se može pokretati na različitim platformama kao što su Mac OS X, Solaris i Windows.

PostgreSQL je besplatan i otvorenog je koda. Njegov izvorni kod je dostupan pod PostgreSQL licencom, liberalna licenca otvorenog koda. Slobodan je za korištenje, modificiranje i distribuiranje u bilo kojem obliku. [1]

PostgreSQL sadržava mnogo modernih odlika:

* Kompleksne upite
* Vanjske ključeve
* Okidače
* Ažurirane poglede
* Transakcijski integritet

2.2. Microsoft C# programski jezik

C# je dizajniran kao platformno neovisan jezik u tradiciji Jave (iako je primarno implementiran na Windows platformi). Sintaksa C# veoma je slična C i C++ programskom jeziku te je C# dizajniran da bude objektno-orjentirani programski jezik.

Vrlo malo se razlikuje u sintaksi od C++. Čini ga mješavina C sintakse sa nekoliko drugih različitih jezika, a ponajviše Jave. Potencial C# programskog jezika je ogroman te je dizajniran da koristi prednosti dizajna .NET framework-a. [2]

2.3. SQL jezik

SQL (Structured Query Language) je računalni jezik korišten pri izradi baza podataka te manipulacijom podataka u bazi podataka. Napravljen je po uzoru na relacijski model Edgar F. Codda predstavljen u njegovom radu „A Relational Model of Data for Large Shared Data Banks“ koji je objavio časopis „Association for Computing Machinery“ 1970. godine. Razvila ga je američka tvrtka IBM, gdje je razvijena baza podataka pod imenom „System R“, a zadatak nekad zvanog Structured Enghlish Query Language („SEQUEL“) je bio upravljanje tom bazom podataka. Kratica „SEQUEL“ je kasnije promjenjena u „SQL“, zato što je tada naziv „SEQUEL“ koristila jedna zrakoplovna tvrtka u Ujedinjenom Kraljevstvu. SQL je 1987. godine postao referentni jezik za relacijske baze podataka.

SQL je poprilično jednostavan jezik i ne zahtjeva preveliko znanje korisnika za upotrebu. Deklarativan je što bi značilo da korisnik sam određuje što SQL treba napraviti. [3]

Tipovi podatka u SQL-u:

- NUMBER – brojevni tip podatka (cijeli, negativni, decimalni)

- INTEGER – cijeli brojevi

- CHAR – niz podataka fiksne duljine

- VARCHAR – niz podataka varijabilne duljine

- DATE/TIME – tip podatka koji se koristi za označavanje datuma/vremena

2.3.1. Naredbe u SQL-u

Postoje četiri kategorije na koje se mogu podjeliti naredbe:

* DDL(data definition language) naredbe
* DML(data manipulation language) naredbe
* DCL(data control language) naredbe
* TCL(transaction control language) naredbe

**DDL** naredbe se koriste kod kreiranja/brisanja tablica, indeksiranja i pogleda. Naredbe tipa (CREATE TABLE, CREATE INDEX, CREATE VIEW, ALTER TABLE, DROP VIEW i DROP INDEX)

**DML** naredbe koje se koriste za dodavanje redaka, izmjenu i brisanje podatka u tablicama. Naredbe tipa (SELECT, INSERT, UPDATE i DELETE). [3]

INSERT naredba koristi se za dodavanje novih redaka u tablicu. Sintaksa INSERT naredbe:

INSERT INTO *tablica* VALUES(*vrijednost\_1, vrijednost\_2..)*

INSERT INTO *tablica(atr1, atr2....)* VALUES*(vr1, vr2....)*

UPDATE se koristi za izmjenu postojećih podataka u tablici. Sintaksa UPDATE naredbe:

UPDATE *tablica* SET atr1 = *vrijednost\_1 , atr2 = vrijednost\_2..* WHERE *atr\_X= vrijednost\_X*

DELETE se koristi za brisanje redaka iz tablica. Sintaksa DELETE naredbe:

DELETE FROM *tablica* WHERE atr\_X = vrijednost\_X

**DCL** naredbe se koristi kod dodjeljivanja dozvola za određene operacija nad bazom podataka.

**TCL** naredbe se koriste kod upravljanja transakcijama u SQL-u. Transakcija predstavlja skup naredbi koje se izvršavaju u bazi podataka.

2.3.2. Operatori u SQL-u

**Aritmetički operatori** – koriste se u aritmetičkim operacijama (+, -, /, \*, %)

**Operatori uspoređivanja** - koriste se kod uspoređivanja pomoću uvijeta upita ( =, <>, <, >, <=, >=)

**Logički operatori** – uspoređuju dva uvjeta istovremeno kako bi se utvrdilo može li redak biti dohvaćen.

OR – minimalno jedan uvjet mora biti zadovoljen da bi bila istina

AND – oba moraju biti istinita da bi bila istina

NOT – svi uvjeti moraju biti lažni da bi bila istina

2.3.3. Matematičke funkcije u SQL-u

SQL ima agregirajuće funkcije koje olakšavaju rad korisnika sa podacima u bazi. A to su sljedeće:

* COUNT – zbroj redova s nekom vrijednosti
* MIN – najmanja vrijednost označenog skupa
* MAX – najveća vrijednost označenog skupa
* SUM – zbroj svih vrijednosti označenog skupa
* AVG – srednja vrijednost označenog skupa

3. OPIS APLIKACIJSKE DOMENE

Prilikom redovnog servisa automobila uočio sam nedostake programa kojeg su koristili moji serviseri. Potaknut tom činjenicom pokušao sam osmisliti vlastitu aplikaciju koja bi bila prikladnija. Aplikacija kao takva sadržava bazu dijelova, bazu automobila, bazu (dosje) stranaka i njihovih vlastitih automobila, također ima mogućnost stvaranja radnih naloga, pregled te ispis istih.

3.1. ERA model

Era model skraćeno od entity-relationship-attribute model. Model koji sadržava podatke te veze između njih u smislu entiteta, veza i atributa.

Entiteti imaju atribute te imaju veze sa ostalim entitetima. Entiteti imaju imena, najćešče su to jedinstvena imena. Entiteti se ćesto implementiraju kao zapisi koji sadržavaju broj polja. Entitet predstavlja stvarni ili apstraktni predmet, događaj ili osobu kojom se pamte podaci u sustavu.

Atributi najćešče opisuju karakteristike entiteta. Svaki atribut sadržava ime, većinom su to jedinstvena imena. Atributi se najćešče koriste kao polja sa nekakvom vrijednosti. Atribut je svojstvo ili obilježje entiteta koje također spremamo u bazu podataka.

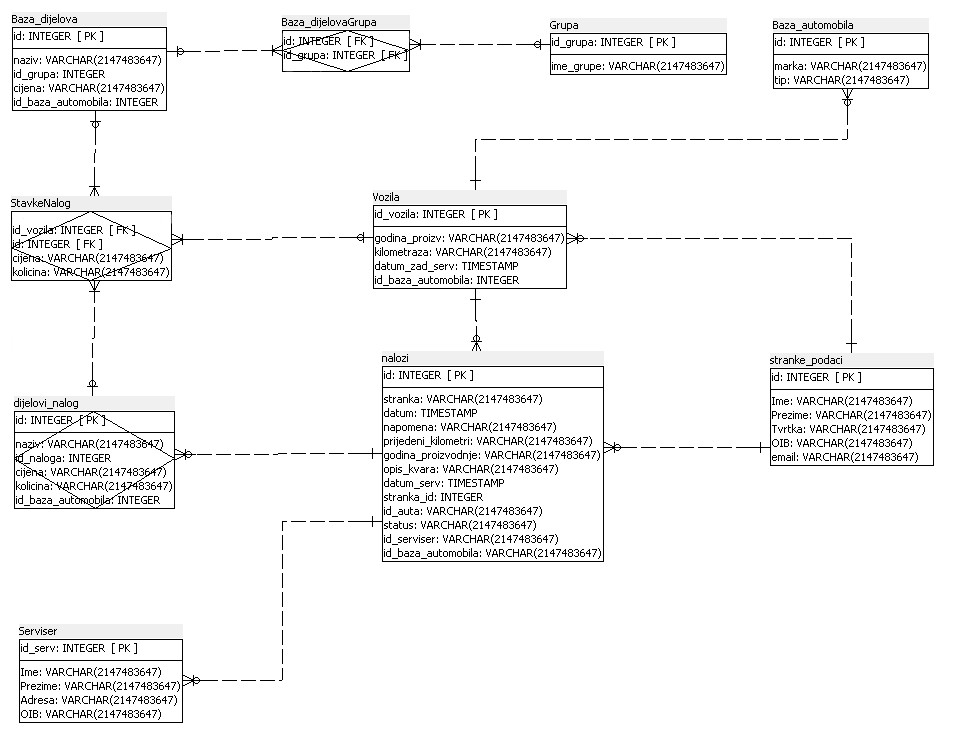
Svaka veza ima dva imena (okrenuto te normalno ime) te sadržavaju imena koja su u većini slučajeva jedinstvena. Može postojati više od jedne veze između dva entiteta. Veza ili relacija je odnos između dva entiteta. Ime veze opisuje ulogu entiteta u vezi.

Ograničenje ili kardinalnost, govori o tome koliko entiteta jednog tipa može biti u vezi s entitetom drugog tipa. [4]

Moguće kardinalnosti su:

* Jedan na jedan (1:1)
* Jedan na više (1:M)
* Više na jedan (M:1)
* Više na više (M:M)

U nastavku je prikazan ERA model moje baze podataka:



*Slika 1. ERA model*

**Baza\_automobila – vozila**; jedno vozilo može sadržavati nijedan ili više automobila, a jedan automobil može pripadati samo jednom vozilu.

**Nalozi – vozila**; nalog može sadržavti samo jedno vozilo, a jedno vozilo se može nalaziti na nijednom ili više naloga.

**Dijelovi\_nalog – nalozi**; stavke naloga mogu pripadati samo jednom nalogu, a jedan nalog može sadržavati nitijednu ili više stavki naloga.

**Grupa – baza\_dijelova**; jedna grupa može sadržavati nijedan ili više dijelova, a jedan dio može pripadati nijednoj ili više grupa.

**Baza\_dijelova – dijelovi\_nalog**; jedan dio može se nalaziti na nijednoj ili više stavnki naloga, a jedna stavka naloga može sadržavati samo jedan dio.

**Dijelovi\_nalog – vozila**; stavke naloga mogu sadržavati samo jedno vozilo, a jedno vozilo može pripadati nijednoj ili više stavki naloga.

**Serviser – nalozi**; jedan serviser može pripadati nijednom ili više naloga, a jedan nalog može sadržavati samo jednog servisera.

**Vozila – stranke\_podaci**; jedno vozilo može pripadati samo jednoj stranci, a jedna stranka može sadržavati nitijedno ili više vozila.

**Nalozi – stranke\_podaci**; jedna stranka može pripadati nijednom ili više naloga, a jedan nalog može sadržavti samo jednu stranku.

4. IMPLEMENTACIJA

4.1. Baza Podataka

Koristi se za unos i uređivanje vrsta automobila koji se koriste kod vozila stranke te kod stvaranja novih naloga. Primarni ključ ove tablice je „id“.

Tablica „Baza\_automobila“

|  |  |  |  |
| --- | --- | --- | --- |
| Ključ | Atribut | Tip podatka | Opis |
| PK | id | INTEGER | identifikator |
|  | marka | VARCHAR | Marka automobila |
|  | tip | VARCHAR | Tip automobila |

Koristi se za unos automobila stranke u bazu, veže se na tablicu „stranke\_podaci“.

Tablica „Vozila“

|  |  |  |  |
| --- | --- | --- | --- |
| Ključ | Atribut | Tip podatka | Opis |
| PK | id\_vozila | INTEGER | Identifikator vozila stranke |
| FK | id | INTEGER | Identifikator stranke |
|  | godina\_proizv | VARCHAR | Godina proizvodnje |
|  | kilometraza | VARCHAR | Prijeđeni kilometri |
|  | datum\_zad\_serv | TIMESTAMP | Datum zadnjeg servisa |
|  | Id\_baza\_automobila | INTEGER | Identifikator automobila |

Koristi se za unos novih stranaka.

Tablica „stranke\_podaci“

|  |  |  |  |
| --- | --- | --- | --- |
| Ključ | Atribut | Tip podatka | Opis |
| PK | id | INTEGER | Identifikator stranke |
|  | Ime | VARCHAR | Ime stranke |
|  | Prezime | VARCHAR | Prezime stranke |
|  | Tvrtka | VARCHAR | Tvrtka |
|  | Oib | VARCHAR | Oib |
|  | email | VARCHAR | Email stranke |

|  |  |  |  |
| --- | --- | --- | --- |
| Ključ | Atribut | Tip podatka | Opis |
| PK | id | INTEGER | Identifikator naloga |
|  | stranka | VARCHAR | Ime i prezime stranke |
|  | datum | TIMESTAMP | Datum naloga |
|  | napomena | VARCHAR | Napomena |
|  | prijedeni\_kilometri | VARCHAR | Prijedjeni kilometri |
|  | godina\_proizvodnje | VARCHAR | Godina proizvodinje |
|  | opis\_kvara | VARCHAR | Opis kvara automobila |
|  | datum\_serv | TIMESTAMP | Datum servisa |
|  | stranka\_id | INTEGER | Identifikator stranke |
|  | id\_auta | VARCHAR | Identifikator auta |
|  | status | VARCHAR | Status naloga |
| FK | id\_vozila | INTEGER | Identifikator\_vozila |
|  | Id\_baza\_automobila | INTEGER | Identifikator automobila |

Koristi se za unos novih naloga za servis, veže se na vanjski ključ tablice „dijelovi\_nalog“ atributi entiteta „id\_nalog“.

Tablica „Nalozi“

Koristi se za unos grupe za grupiranje dijelova.

Tablica „Grupa“

|  |  |  |  |
| --- | --- | --- | --- |
| Ključ | Atribut | Tip podatka | Opis |
| PK | id\_grupa | INTEGER | identifikator |
|  | ime\_grupe | VARCHAR | Ime grupe |

Koristi se za spremanje novih dijelova.

Tablica „baza dijelova“

|  |  |  |  |
| --- | --- | --- | --- |
| Ključ | Atribut | Tip podatka | Opis |
| PK | id | INTEGER | Identifikator dijela |
|  | naziv | VARCHAR | Naziv automobila |
| FK | id\_grupa | INTEGER | Identifikator grupe |
|  | cijena | VARCHAR | Cjena dijela |
|  | Id\_baza\_automobila | INTEGER | Identifikator automobila |

Koristi se za spremanje dijelova u nalozima.

Tablica „djelovi\_nalog“

|  |  |  |  |
| --- | --- | --- | --- |
| Ključ | Atribut | Tip podatka | Opis |
| PK | id | INTEGER | Identifikator djela |
|  | naziv | VARCHAR | Naziv dijela |
| FK | id\_naloga | INTEGER | Identifikator naloga |
|  | cijena | VARCHAR | Cjena |
|  | kolicina | VARCHAR | Kolicina |
|  | Id\_baza\_automobila | INTEGER | Identifikator automobila |

Tablica „Serviser“

|  |  |  |  |
| --- | --- | --- | --- |
| Ključ | Atribut | Tip podatka | Opis |
| PK | Id\_serv | INTEGER | Identifikator servisera |
|  | Ime | VARCHAR | Ime servisera |
|  | Prezime | VARCHAR | Prezime servisera |
|  | Adresa | VARCHAR | Adresa servisera |
|  | OIB | VARCHAR | OIB servisera |

Koristi se za spremanje servisera u tablicu.

4.2. Djelovi koda

U ovoj aplikaciji postoje samo tri klase koje se koriste:

**Program.cs** – integrirana klasa koja se pokreće prva te određuje koja forma se pokreće prva kod pokretanja aplikacije

**classConnectDatabase.cs** – klasa koja sadržava funkciju u kojoj su upisane informacije poput imena servera, broja porta, korisničkog imena, imena baze podataka te funkciju za provjeru je li povezivanje na bazu podataka otvoreno

**classSQL.cs** – klasa sadržava funkciju povezivanja na bazu te funkcije za izvršavanje SQL naredbi nad bazom podataka

Povezivanje na PostgreSQL bazu podataka:

Klasa „classConnectDatabase“ sadržava funkciju „GetRemoteConnectionString()“ koja se poziva u klasi „classSQL“ koja se poziva kod svakog upita.

public static string GetRemoteConnectionString()

{

string server = "", port = "", username = "", password = "", database = "";

string path = Path.GetDirectoryName(Application.ExecutablePath) + "\\settings.xml";

DataSet DS = new DataSet();

DS.ReadXml(path);

server = "localhost";

port = "5432";

username = "postgres";

database = "Glavna";

return "Server=" + server + ";Port=" + port + ";" + "User Id=" + username + ";Password=" + password + ";Database=" + database + ";";

}

Klasa „classSQL“

Spajanje na bazu podataka :

public static DataSet dataSet = new DataSet();

public static NpgsqlDataAdapter Npgadapter = new NpgsqlDataAdapter();

public static string remoteConnectionString = SQL.claasConnectDatabase.GetRemoteConnectionString();

public static NpgsqlConnection remoteConnection = new NpgsqlConnection(remoteConnectionString);

Ovaj dio koda stvara sql data adaptere koje se koriste za sql insert, update, delete da bi popunio tablicu, zatim ulazi u klasu gdje isčitava potake za spajanje na bazu, zatim se spaja na bazu.

Izvršavanje SQL naredbe SELECT :

//SELECT

public static DataSet select(string sql, string table)

{

dataSet = new DataSet();

if (remoteConnection.State.ToString() == "Closed") { remoteConnection.Open(); }

NpgsqlDataAdapter da = new NpgsqlDataAdapter(sql.Replace("+", "||").Replace("[", "\"").Replace("]", "\"").Replace("zbroj", "+").Replace("?", "\""), remoteConnection);

dataSet.Reset();

da.Fill(dataSet);

return dataSet;

}

Ovaj dio koda provjerava ako je konekcija na bazu otvorena, ako nije otvara je te nakon toga stvara adaptere za punjenje tablice naredbom select, zatim se mjenjaju neki znakovi zbog sintakse izvršavanja SQL koda, pa slijedi punjene tablice dobivenim dobivenim rezultatom. Nakon svega iznad odrađenog funkcija vraća tablicu kao podatak.

Izvršavanje SQL naredbe INSERT :

//INSERT

public static string insert(string sql)

{

try

{

if (remoteConnection.State.ToString() == "Closed") { remoteConnection.Open(); }

NpgsqlCommand comm = new NpgsqlCommand(sql.Replace("+", "||").Replace("[", "\"").Replace("]", "\"").Replace("zbroj", "+").Replace("?", "\""), remoteConnection);

comm.ExecuteNonQuery();

return "";

}

catch (Exception ex)

{

remoteConnection.Close();

return ex.ToString();

}

}

Za razliku od funkcije SELECT koja vraća vrijednost, za funkciju INSERT to nije potrebno.

Izvršavanje SQL naredbe UPDATE :

//UPDATE

public static string update(string sql)

{

try

{

if (remoteConnection.State.ToString() == "Closed") { remoteConnection.Open(); }

NpgsqlCommand comm = new NpgsqlCommand(sql.Replace("+", "||").Replace("[", "\"").Replace("]", "\"").Replace("zbroj", "+").Replace("?", "\""), remoteConnection);

comm.ExecuteNonQuery();

return "";

}

catch (Exception ex)

{

remoteConnection.Close();

return ex.ToString();

}

}

Također funkcija UPDATE ne vrača vrijednost već se samo izvršava.

Izvršavanje SQL naredbe DELETE :

//DELETE

public static string delete(string sql)

{

try

{

if (remoteConnection.State.ToString() == "Closed") { remoteConnection.Open(); }

NpgsqlCommand comm = new NpgsqlCommand(sql.Replace("+", "||").Replace("[", "\"").Replace("]", "\"").Replace("zbroj", "+").Replace("?", "\""), remoteConnection);

comm.ExecuteNonQuery();

return "";

}

catch (Exception ex)

{

remoteConnection.Close();

return ex.ToString();

}

}

Također funkcija DELETE ne vrača vrijednost već se samo izvršava.

Svaka forma kod pokretanja ima inicijalizaciju podataka:

private void puni\_grid()

{

DateTime datum = Convert.ToDateTime(monthCalendar1.SelectionRange.Start.ToShortDateString());

string sql1 = "select ?id? as [Broj], ?stranka? as [Stranka], ?datum? as [Datum] from ?nalozi? where ?datum? >= '" + datum.Date.ToString("dd-MM-yyyy 00:00:00") + "' and ?datum? <= '" + datum.Date.ToString("dd-MM-yyyy 24:00:00") + "'";

DTgrupa = classSQL.select(sql1, "Table").Tables[0];

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

dataGridView1.DataSource = DTgrupa;

DataGridViewColumn column = dataGridView1.Columns[0];

DataGridViewColumn column1 = dataGridView1.Columns[1];

DataGridViewColumn column2 = dataGridView1.Columns[2];

column.Width = 29;

column1.Width = 120;

column2.Width = 150;

}

Ovaj dio koda se izvršava na „Load eventu“ forme „Form1“ koja je ujedno i glavna forma kod pokretanja programa te služi tome da pupuni element za prikaz podataka.

private void puni\_combo()

{

DataTable DTSK = new DataTable("odabir");

DataTable BDDT = new DataTable("odabir1");

string sql1 = "select \* from ?Grupa?";

DTSK = classSQL.select(sql1, "tablice").Tables[0];

comboBox3.DataSource = DTSK;

comboBox3.DisplayMember = "ime\_grupe";

comboBox3.ValueMember = "id\_grupa";

string sql2 = "select \* from ?Baza\_automobila?";

BDDT = classSQL.select(sql2, "djelovi").Tables[0];

comboBox1.DataSource = BDDT;

comboBox1.DisplayMember = "marka";

comboBox1.ValueMember = "id";

DataTable BDDT2 = new DataTable();

try

{

if (comboBox1.Text != "" && comboBox1.SelectedValue != null)

{

string sql3 = "select \* from ?Baza\_automobila? where ?id? = '" + comboBox1.SelectedValue.ToString() + "'";

BDDT2 = classSQL.select(sql3, "auti").Tables[0];

comboBox2.DataSource = BDDT2;

comboBox2.DisplayMember = "tip";

comboBox2.ValueMember = "id";

}

}

catch

{ }

}

Ovaj dio koda se također izvršava kod pokretanja forme, a služi isključivo tome da popuni element padajući izbornik podacima iz baze.

Ovaj dio koda se izvršava u dijelu programa za manipulaciju podacima u tablici „Baza\_dijelova“, ovisno o varijabli „edit“ mijenja se već postojeći zapis u bazi ili se dodaje novi podatak.

Za upis novog podatak u bazu koristi se funkcija INSERT , pošto je u tablici identifikator „id“ serijski tip podatka što bi značilo da je autoinkrementalni, ostala imena kolona u koje redove želimo zapisati podatak su upisana u prvom dijelu te nakon toga su upisane vrijednosti koje se zapisuju.

Za promjenu podatka u bazi podataka korisi se funkcija UPDATE. Kod ove vrste manipulacije bazom upisuju se imena kolona te se izjednačuju sa novom vrijednosti. Da UPDATE ne bi promjenio sve podatke u tablici koristi se naredba WHERE gdje su zapisani uvjeti koji određuju u koji red ili redove će podatak biti ažurirani.

if (edit == false)

{

try

{

int vrj\_id = vrati\_id\_auta();

string sql = "INSERT INTO ?Baza\_dijelova?(?naziv?,?id\_grupa?, ?cijena?, ?id\_baza\_automobila?) " +

" VALUES('" + txtnaziv\_dijela.Text + "', '" + comboBox3.SelectedValue.ToString() + "', '" + txtcijena.Text + "', '" + vrj\_id + "')";

classSQL.insert(sql);

MessageBox.Show("Uspješno dodan zapis !");

if (System.Windows.Forms.Application.OpenForms["Baza\_djelova"] != null)

{

(System.Windows.Forms.Application.OpenForms["Baza\_djelova"] as Baza\_djelova).popuni\_grid();

}

this.Close();

}

catch

{

MessageBox.Show("Greška kod spremanja !");

}

}

else if (edit == true)

{

try

{

int vrj\_id1 = vrati\_id\_auta();

string sql1 = "update ?Baza\_dijelova? SET ?naziv? = '" + txtnaziv\_dijela.Text + "', ?id\_grupa? = '" + comboBox3.SelectedValue.ToString() + "', ?cijena? = '" + txtcijena.Text + "', ?id\_baza\_automobila? = '" + vrj\_id1 + "' WHERE ?id? = '" + id\_dijela + "'";

classSQL.update(sql1);

MessageBox.Show("Uspješno promijenjen zapis !");

if (System.Windows.Forms.Application.OpenForms["Baza\_djelova"] != null)

{

(System.Windows.Forms.Application.OpenForms["Baza\_djelova"] as Baza\_djelova).popuni\_grid();

}

this.Close();

}

catch

{

MessageBox.Show("Greška kod spremanja !");

}

}

U ovoj aplikaciji se koristi obavještavanje stranke pomoću slanja emaila, stoga se u ovom dijelu koda definira e-mail server, korisničko ime, lozinka, port te se definira da li želimo koristiti SSL („secure socket layer“) da bi smo spriječili treću stranu da nam čita poruke, drugim riječima da spriječimo napad.

public void email\_send(string stranka\_email,string stranka\_podaci)

{

MailMessage mail = new MailMessage();

SmtpClient SmtpServer = new SmtpClient("smtp.gmail.com");

mail.From = new MailAddress("zavrsni.postgres@gmail.com");

mail.To.Add(stranka\_email);

mail.Subject = "Obavijest o servisu vozila";

mail.Body = "Poštovani "+ stranka\_podaci +", Vaše vozilo je servisirano te je spremno za podizanje ! ";

//System.Net.Mail.Attachment attachment;

//attachment = new System.Net.Mail.Attachment("c:/textfile.txt");

//mail.Attachments.Add(attachment);

SmtpServer.Port = 587;

SmtpServer.Credentials = new System.Net.NetworkCredential("zavrsni.postgres@gmail.com", "Zavr12345");

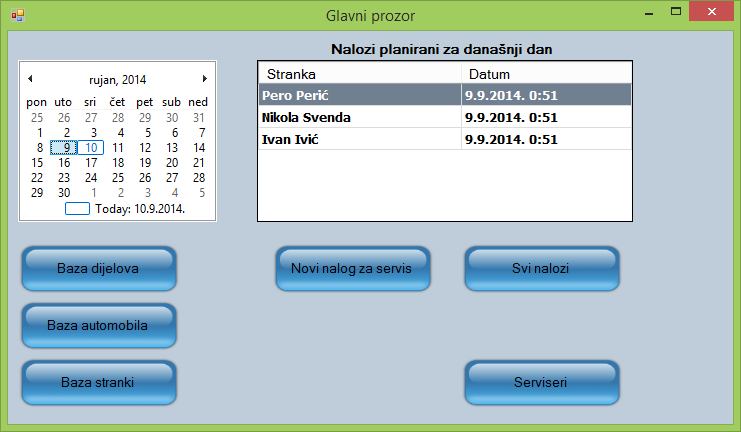
SmtpServer.EnableSsl = true;

SmtpServer.Send(mail);

}

5.PRIMJER KORIŠTENJA

5.1. Glavni Ekran



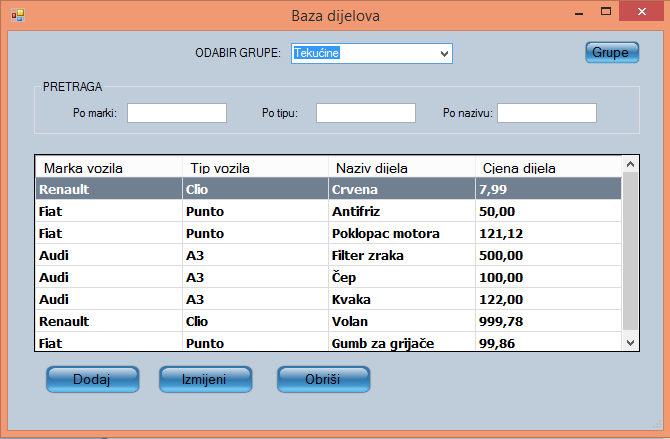
Slika 2. Glavni prozor

Prilikom pokretanja aplikacije prvo se pokreće forma „Glavni prozor“. Ona prikazuje podatke iz baze podataka o nalozima za trenutni dan, a ovisno o odabiru datuma na kalendaru u formi mijenjaju se nalozi. Na „Glavnom prozoru“ nalaze se sve kontrole za otvaranje ostalih formi.

Na glavnom prozoru nalaze se slijedeći gumbi:

|  |  |
| --- | --- |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.12.46.png | * pritiskom na taj gumb otvara se forma koja sadržava popis djelova te kontrole za manipulaciju podataka |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.12.58.png | * pritiskom na taj gumb otvara se forma koja sadržava popis automobila te kontrole za manipulaciju podataka |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.06.png | * pritiskom na taj gumb otvara se forma koja sadržava popis stranaka te kontrole za manipulaciju podataka |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.12.35.png | * pritiskom na taj gumb otvara se forma za stvaranje novog servisnog naloga |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.12.11.png | * pritiskom na taj gumb otvara se forma koja sadržava popis svih naloga te kontrole za manipulaciju podataka |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.12.21.png | * pritiskom na taj gumb otvara se forma koja sadržava popis servisera te kontrole za manipulaciju podataka |

5.2. Baza dijelova

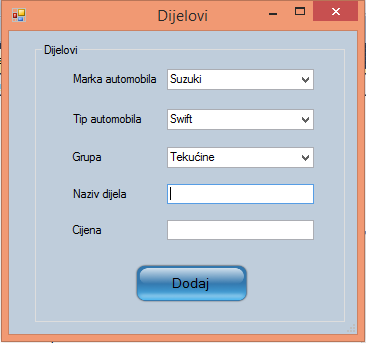


Slika 3. Baza dijelova

U ovoj formi nalazi se popis dijelova potrebnih za servisiranje i popravak vozila. Za lakše ponalaženje željenog dijela nalazi se pretraživač iznad samog popisa dijelova. Pretraživanje je moguće po marki vozila, tipu vozila i nazivu dijela. Tu se nalaze i gumbi:

|  |  |
| --- | --- |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.16.png | * pritiskom na ovaj gumb otvara se forma grupa koja je opisana u daljnjem tekstu |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.28.png | * pritiskom na ovaj gumb otvara se forma za dodavanje novog dijela |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.39.png | * pritiskom na ovaj gumb otvara se forma za promijenu podataka prethodno odabranog dijela |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.50.png | * pritiskom na ovaj gumb brišemo dio iz baze podataka |

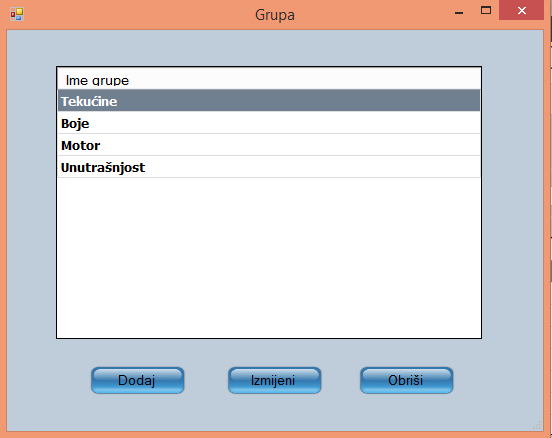
Pritiskom na gumb „Dodaj“ ili „Izmijeni“ otvara se forma na Slici 4.



Slika 4. Dijelovi

Ona služi za dodavanje i mijenjanje podataka o dijelovima u bazi.

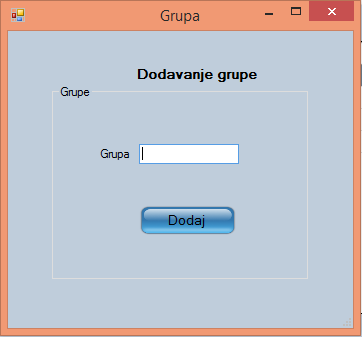
5.3. Grupa



Slika 5. Grupa

U ovoj formi nalazi se popis grupa pomoću kojih se grupiraju dijelovi. Tu se nalaze slijedeći gumbi:

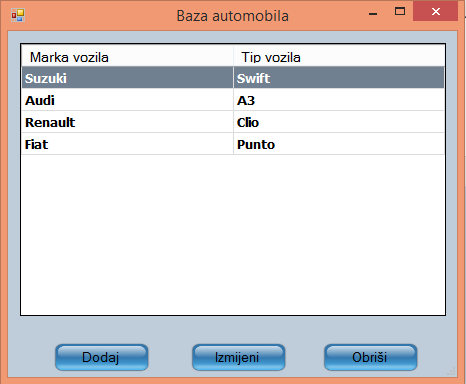
|  |  |
| --- | --- |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.28.png | * pritiskom na ovaj gumb otvara se forma za dodavanje nove grupe |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.39.png | * pritiskom na ovaj gumb otvara se forma za promijenu podataka prethodno odabrane grupe |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.50.png | * pritiskom na ovaj gumb brišemo grupu iz baze podataka |



Slika 6. Dodavanje grupe

U ovoj formi se dodaju ili mijenjaju podaci o grupi u bazi podataka.

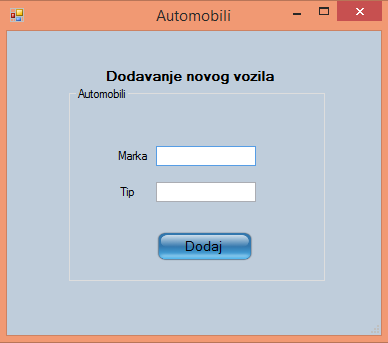
5.4. Baza automobila



Slika 7. Baza automobila

U ovoj formi nalazi se popis automobila koje smo do sad unjeli. Nalaze se i slijedeći gumbi:

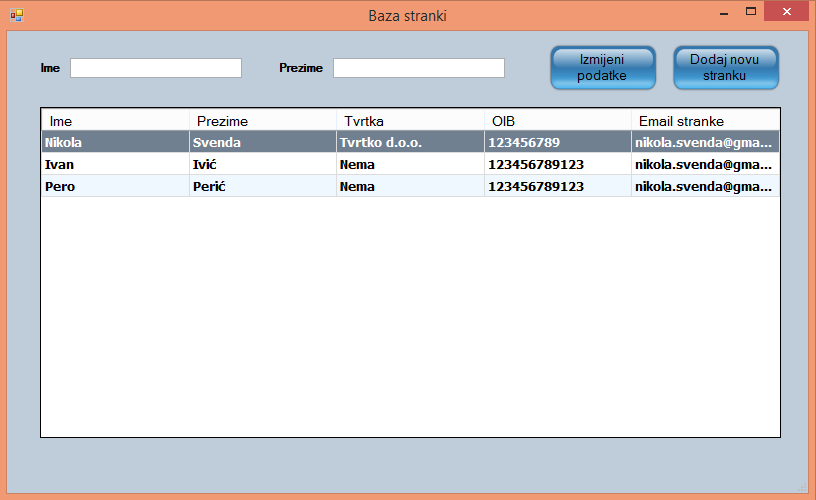
|  |  |
| --- | --- |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.28.png | * pritiskom na ovaj gumb otvara se forma za dodavanje novog automobila |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.39.png | * pritiskom na ovaj gumb otvara se forma za promijenu podataka prethodno odabranog automobila |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.50.png | * pritiskom na ovaj gumb brišemo odabrani automobil iz baze podataka |



Slika 8. Dodavanje automobila

U ovoj formi se dodaju ili mijenjaju podaci o automobilima u bazi podataka.

5.5. Baza stranaka



Slika 9. Baza stranaka

U ovoj formi nalazi se popis svih stranaka koje smo do sad unijeli u bazu podataka. Nalazi se i pretraživač zapisa s time da možemo pretraživati po imenu ili po prezimenu stranke. Nalaze se i slijedeći gumbi:

|  |  |
| --- | --- |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.14.17.png | * pritiskom na ovaj gumb otvara se forma sa slike 10. koja služi za dodavanje nove stranke u bazu podataka |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.14.09.png | * pritiskom na ovaj gumb otvara se forma sa slike 10. koja služi za promjenu podataka o stranci |



Slika 10. Forma za promjenu ili dodavanje stranke

U ovoj formi se mijenjaju ili dodaju podaci o strankama i njihovim vozilima.

|  |  |
| --- | --- |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.14.35.png | * pritiskom na ovaj gumb spremaju se promjene podataka o stranci i dodaju se nove stranke |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.14.41.png | * pritiskom na ovaj gumb briše se prethodno odabrana stranka |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.14.25.png | * pritiskom na ovaj gumb otvara se forma u kojoj se nalaze sva vozila koja pripadaju pethodno odabranoj stranci |

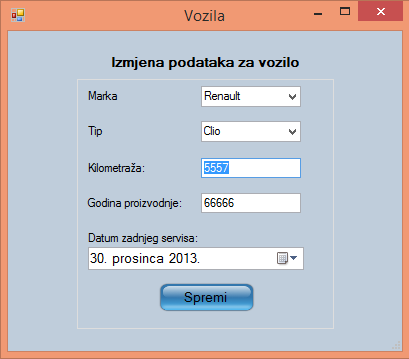
5.6. Vozila stranke



Slika 11. Forma za pregled vozila stranke

U ovoj formi nalazi se popis svih vozila prethodno odabrane stranke. Stranku smo odabrali u formi na slici 9. Tu se nalaze i slijedeći gumbi:

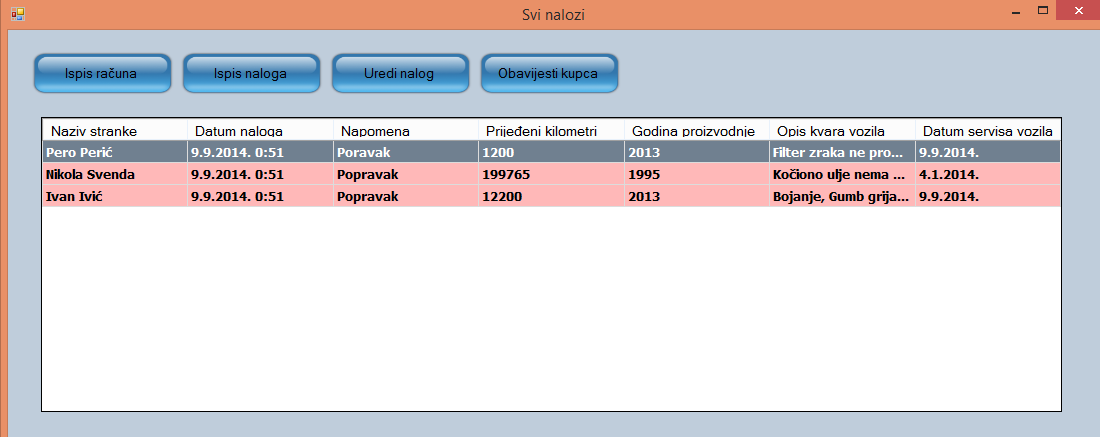
|  |  |
| --- | --- |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.28.png | * pritiskom na ovaj gumb otvara se forma za dodavanje novog automobila odabrane stranke |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.39.png | * pritiskom na ovaj gumb otvara se forma za promijenu podataka odabranog automobila stranke |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.50.png | * pritiskom na ovaj gumb brišemo odabrani automobil stranke iz baze podataka |



Slika 12. Forma za uređivanje vozila stranke

U toj formi se dodaju ili mijenjaju vozila koja pipadaju prethodno odabranoj stranci.

5.7. Popis svih naloga

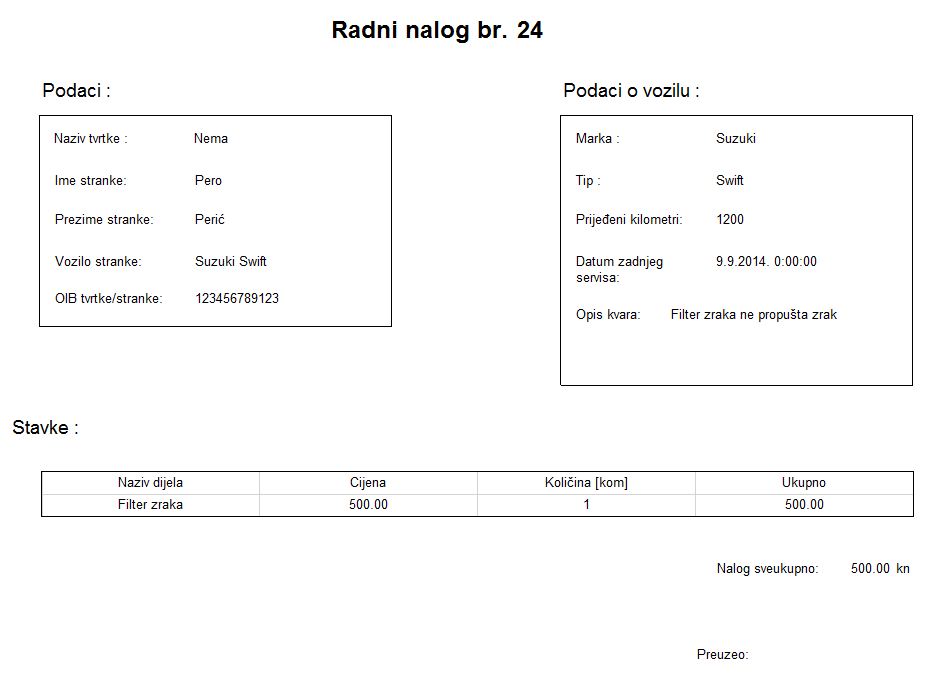


Slika 13. Forma za prikaz svih naloga

U ovoj formi se nalazi popis svih naloga koji su do sad stvoreni sa svim važnim podacima za taj nalog. Nakon odabira naloga moguće je koristiti jednu od slijedećih kontrola:

|  |  |
| --- | --- |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.15.01.png | * pritiskom na ovaj gumb otvara se izvješće sa prikazom računa za odabrani nalog |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.15.16.png | * pritiskom na ovaj gumb otvara se izvješće sa prikazom odabranog radonog naloga |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.15.23.png | * pritiskom na ovaj gumb otvara se forma za uređivanje odabranog naloga |
| \\psf\Home\\Desktop\Screen Shot 2014-09-11 at 01.01.19.png | * pritiskom na ovaj gumb obavještava se stranku o završenom servisu putem e-mail-a čime redak u popisu naloga mijenja boju iz crvene u zelenu kao znak završenog naloga |

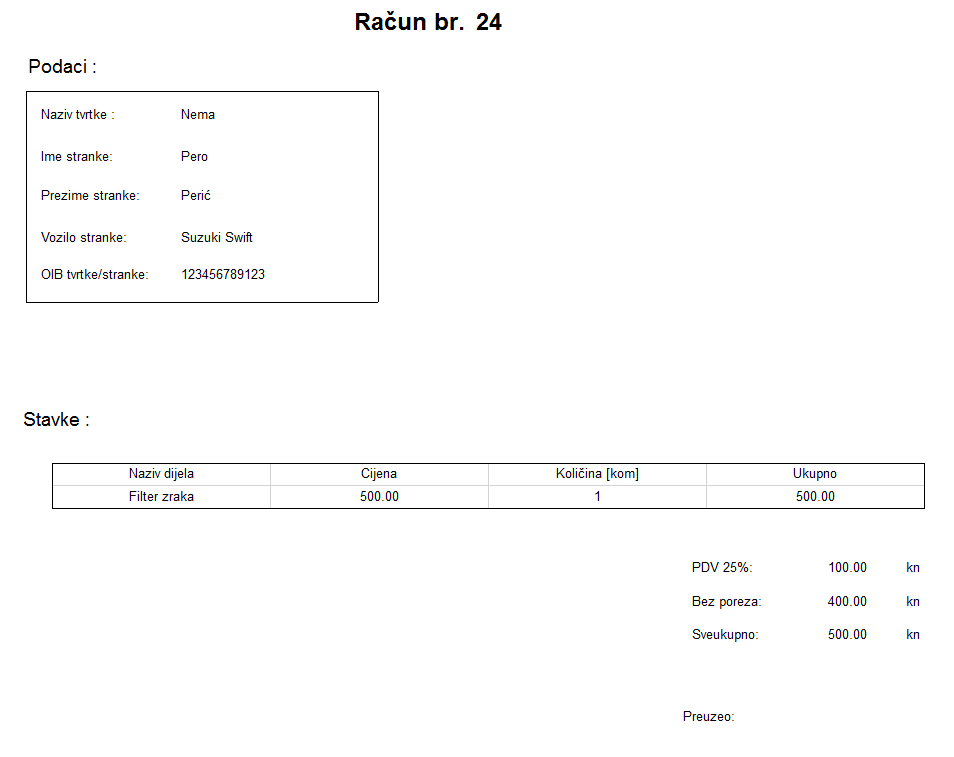
5.7.1. Ispis naloga



Slika 14. Izvješće radnog naloga

Slika 14. prikazuje izvješće odabranog radnog naloga spremnog za ispis.

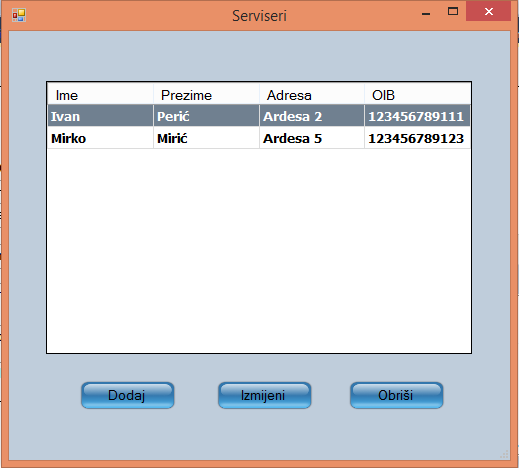
5.7.2 Ispis računa



Slika 15. Račun

Slika 15. prikazuje račun spreman za ispis.

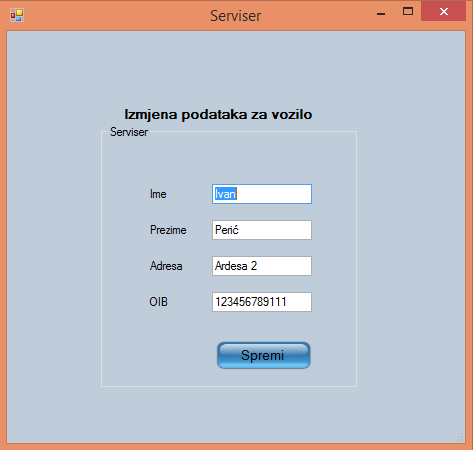
5.8. Serviseri



Slika 16. Forma prikaza svih servisera

U ovoj formi se nalazi popis svih zaposlenih servisera sa svim potrebnim podacima. Dodavanje, mijenjanje i brisanje vrši se pomoću slijedećih gumbi:

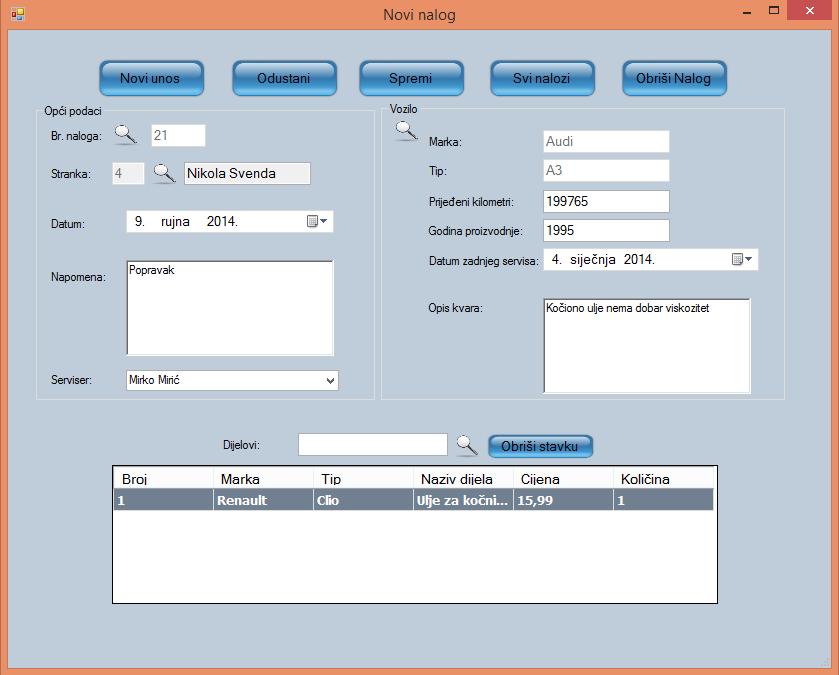
|  |  |
| --- | --- |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.28.png | * pritiskom na ovaj gumb otvara se forma za dodavanje novog servisera |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.39.png | * pritiskom na ovaj gumb otvara se forma za promijenu podataka odabranog servisera |
| \\psf\Home\\Desktop\Screen Shot 2014-09-10 at 23.13.50.png | * pritiskom na ovaj gumb brišemo odabranog servisera iz baze podataka |



Slika 17. Forma za dodavanje i promjenu podataka servisera

Ova forma otvara se prilikom pritiska na gumb „Dodaj“ ili „Izmijeni“ u formi „Serviseri“ te se pomoću nje dodaje ili mijenja neki od podataka pridodanih odabranom serviseru.

5.9. Novi radni nalog



Slika 18. Forma za unos novog radnog naloga

U ovoj formi serviser stvara novi radni nalog ili mijenja postojeći. Forma sadrži slijedeće kontrole:

|  |  |
| --- | --- |
| \\psf\Home\\Desktop\Screen Shot 2014-09-11 at 01.25.02.png | * pritiskom na ovaj gumb trenutnom nalogu dodjeljuje se novi broj |
| \\psf\Home\\Desktop\Screen Shot 2014-09-11 at 01.25.21.png | * nakon pritiska na gumb „Novi unos“ počinje se stvarati novi nalog te se tim gumbom odustaje od stvaranja |
| \\psf\Home\\Desktop\Screen Shot 2014-09-11 at 01.25.34.png | * klikom na ovaj gumb sprema se novi nalog u bazu podataka |
| \\psf\Home\\Desktop\Screen Shot 2014-09-11 at 01.25.43.png | * pritiskom na ovaj gumb otvara se forma sa prikazom svih dosadašnjih radnih naloga |
| \\psf\Home\\Desktop\Screen Shot 2014-09-11 at 01.25.54.png | * pritiskom na ovaj gumb briše se radni nalog koji je prethodno učitan u formu |
| \\psf\Home\\Desktop\Screen Shot 2014-09-11 at 01.29.20.png | * klikom na ovaj gumb briše se stavka naloga koju smo dodali |
| \\psf\Home\\Desktop\Screen Shot 2014-09-11 at 01.26.42.pngpopis stranke | * pritiskom na povečalo otvara se forma sa popisom stranaka za automatsko popunjavanje podataka |
| \\psf\Home\\Desktop\Screen Shot 2014-09-11 at 01.26.42.pngpopis vozila | * pritiskom na povečalo otvara se forma sa popisom svi vozila stranke za koju se stvara nalog |
| \\psf\Home\\Desktop\Screen Shot 2014-09-11 at 01.26.42.pngpopis dijelova | * pritiskom na povečalo otvara se forma sa popisom svi dijelova |

6. KRITIČKI OSVRT

Smatram da je ovo dobra aplikacija zato što možemo povezati i pogledati stare stranke, automobile te ispisati radne naloge i račune. Baza sadrži dosje svake stranke te se može pogledati što se sve radilo na dotičnim automobilima.

Dakako aplikacija se može unaprijediti tako da je pretvorimo u web-aplikaciju gdje bi svaki od korisnika imao svoju lozinku za pristup. Primjerice stranka bi mogla pogledati slobodne termine za servis te ako joj neki od termina odgovara mogla bi se predbilježiti za određeni termin. On-line pristupom u svakom trenu stranka bi mogla vidjeti u kojem je stadiju servis njenog vozila. Također bi se aplikacija mogla poboljšati na način da se omogući uvoz podataka u bazu pomoću raznih tipova dokumenata.

Mana aplikacije je ručni unos svih stranaka jer ne postoji unificirana baza korisnika i njihovih automobila.

7. ZAKLJUČAK

Svakodnevno se povećava broj tvrtki kod kojih je izražena potreba za velikom količinom podataka koji su međusobno povezani i sortirani na određeni željeni način. Kako to u papirnatom obliku predstavlja veliki problem rješenje nalaze u informatizaciji poslovanja. Sve više je popularan ERP (Enterprise Resource Planning) te CRM (Customer Relatioship Management) sustav zbog potrebe za kontrolom velike količine robe na skladištu ili/i velika količina klijenata u nekom poslovanju.

Ovim završnim radom prikazana je jednostavna aplikacija koja upravlja s bazom podataka. Može se primijeniti u nekolicini obrta i poduzeća sa sličnim potrebama koje pruža ova aplikacija. PostgreSQL kao besplatni sustav za upravljanje bazom podataka uvelike je popularizirao korištenje baza kod običnih korisnika te malih tvrtki. Tvrtke s vrlo malim budžetom i kvalitetnim kadrom mogu na vrlo lak način modelirati te izgraditi vlastiti informacijski sustav. Tako nešto osigurava minimalne troškove uz visku sigurnost podataka, brzinu, učinkovitost te ponajviše integritet.

8. LITERATURA

1. [www.postgrestutorial.com](http://www.postgrestutorial.com)
2. [www.cprogramming.com/tutorial/csharp.html](http://www.cprogramming.com/tutorial/csharp.html)
3. [www.encyclopedia.com/doc/1011-ERAmodel.html](http://www.encyclopedia.com/doc/1011-ERAmodel.html)
4. C.Coronel, S.Morris, P.Rob, Database systems, Boston, SAD: Cengage Learning, 2011. pp. 7-247

9. PRILOG

Forma „Baza\_auta“

namespace Aplikacija\_za\_zavrsni

{

public partial class Baza\_auta : Form

{

public Baza\_auta()

{

InitializeComponent();

}

private void Baza\_auta\_Load(object sender, EventArgs e)

{

popuni\_grid();

}

DataTable DTgrid = new DataTable();

public void popuni\_grid()

{

//dataGridView1.Rows.Clear();

string sql1 = "select \* from ?Baza\_automobila?";

DTgrid = classSQL.select(sql1, "Table").Tables[0];

dataGridView1.DataSource = DTgrid;

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

DataGridViewColumn column\_id = dataGridView1.Columns[0];

column\_id.Visible = false;

DataGridViewColumn column\_marka = dataGridView1.Columns[1];

column\_marka.HeaderText = "Marka vozila";

DataGridViewColumn column\_tip = dataGridView1.Columns[2];

column\_tip.HeaderText = "Tip vozila";

}

int col1;

int row1;

private void button1\_Click(object sender, EventArgs e)

{

int row = dataGridView1.CurrentCell.RowIndex;

string id = dataGridView1.Rows[row].Cells[0].FormattedValue.ToString();

string marka = dataGridView1.Rows[row].Cells[1].FormattedValue.ToString();

string tip = dataGridView1.Rows[row].Cells[2].FormattedValue.ToString();

if (MessageBox.Show("Da li ste sigurni da želite obrisati automobil " + marka + " " + tip + " ", "Upozorenje", MessageBoxButtons.YesNo, MessageBoxIcon.Information) == DialogResult.Yes)

{

string sql = "Delete from ?Baza\_automobila? where ?id? = '" + id + "'";

classSQL.delete(sql);

MessageBox.Show("Zapis obrisan !");

popuni\_grid();

}

}

private void btndodaj\_Click(object sender, EventArgs e)

{

edit\_auti ed\_dj = new edit\_auti();

ed\_dj.ShowDialog();

}

private void btnizmijeni\_Click(object sender, EventArgs e)

{

int col = dataGridView1.CurrentCell.ColumnIndex;

int row = dataGridView1.CurrentCell.RowIndex;

edit\_auti eddj = new edit\_auti();

eddj.edit = true;

eddj.id\_auta = Convert.ToInt32(dataGridView1.Rows[row].Cells[0].FormattedValue);

eddj.ShowDialog();

}

private void pic\_ButtonEnter(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w + 3, h + 3);

}

private void pic\_ButtonLeave(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w - 3, h - 3);

}

}

}

Forma „Baza\_dijelova“

namespace Aplikacija\_za\_zavrsni

{

public partial class Baza\_djelova : Form

{

public Baza\_djelova()

{

InitializeComponent();

}

public bool edit\_nalog { get; set; }

private void Baza\_djelova\_Load(object sender, EventArgs e)

{

puni\_Combo();

popuni\_grid();

}

private void PaintRows(DataGridView dg)

{

int br = 0;

for (int i = 0; i < dg.Rows.Count; i++)

{

if (br == 0)

{

dg.Rows[i].DefaultCellStyle.BackColor = Color.AliceBlue;

br++;

}

else

{

dg.Rows[i].DefaultCellStyle.BackColor = Color.White;

br = 0;

}

}

DataGridViewRow row = dg.RowTemplate;

row.Height = 25;

}

DataTable DTgrid = new DataTable();

public void popuni\_grid()

{

string sql1 = "SELECT ?Baza\_dijelova?.id, ?Baza\_automobila?.marka, ?Baza\_automobila?.tip,?Baza\_dijelova?.naziv, ?Baza\_dijelova?.id\_grupa, " +

" ?Baza\_dijelova?.cijena, ?Baza\_dijelova?.id\_baza\_automobila FROM ?Baza\_dijelova?, ?Baza\_automobila? WHERE " +

" ?Baza\_dijelova?.id\_baza\_automobila = ?Baza\_automobila?.id ";

DTgrid = classSQL.select(sql1, "Table").Tables[0];

dataGridView1.DataSource = DTgrid;

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

DataGridViewColumn column\_id = dataGridView1.Columns[0];

column\_id.Visible = false;

DataGridViewColumn column\_marka = dataGridView1.Columns[1];

column\_marka.HeaderText = "Marka vozila";

DataGridViewColumn column\_tip = dataGridView1.Columns[2];

column\_tip.HeaderText = "Tip vozila";

DataGridViewColumn column\_naziv = dataGridView1.Columns[3];

column\_naziv.HeaderText = "Naziv dijela";

DataGridViewColumn column\_id\_grupa = dataGridView1.Columns[4];

column\_id\_grupa.Visible = false;

DataGridViewColumn column\_cijena = dataGridView1.Columns[5];

column\_cijena.HeaderText = "Cjena dijela";

}

public void puni\_Combo()

{

DataTable DTSK = new DataTable("odabir");

string sql2 = "select \* from ?Grupa?";

DTSK = classSQL.select(sql2, "tablice").Tables[0];

comboBox1.DataSource = DTSK;

comboBox1.DisplayMember = "ime\_grupe";

comboBox1.ValueMember = "id\_grupa";

}

private void comboBox1\_SelectedValueChanged(object sender, EventArgs e)

{

if (comboBox1.ValueMember != "")

{

string sql1 = "SELECT ?Baza\_dijelova?.id, ?Baza\_automobila?.marka, ?Baza\_automobila?.tip,?Baza\_dijelova?.naziv, ?Baza\_dijelova?.id\_grupa, " +

" ?Baza\_dijelova?.cijena, ?Baza\_dijelova?.id\_baza\_automobila FROM ?Baza\_dijelova?, ?Baza\_automobila? WHERE " +

" ?Baza\_dijelova?.id\_baza\_automobila = ?Baza\_automobila?.id AND ?id\_grupa? = '" + comboBox1.SelectedValue + "'";

DTgrid = classSQL.select(sql1, "Table").Tables[0];

dataGridView1.DataSource = DTgrid;

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

DataGridViewColumn column\_id = dataGridView1.Columns[0];

column\_id.Visible = false;

DataGridViewColumn column\_marka = dataGridView1.Columns[1];

column\_marka.HeaderText = "Marka vozila";

DataGridViewColumn column\_tip = dataGridView1.Columns[2];

column\_tip.HeaderText = "Tip vozila";

DataGridViewColumn column\_naziv = dataGridView1.Columns[3];

column\_naziv.HeaderText = "Naziv dijela";

DataGridViewColumn column\_id\_grupa = dataGridView1.Columns[4];

column\_id\_grupa.Visible = false;

DataGridViewColumn column\_cijena = dataGridView1.Columns[5];

column\_cijena.HeaderText = "Cjena dijela";

DataGridViewColumn column\_id\_baza\_auta = dataGridView1.Columns[6];

column\_id\_baza\_auta.HeaderText = "Baza automobila";

column\_id\_baza\_auta.Visible = false;

dataGridView1.Visible = true;

if (DTgrid.Rows.Count <= 0)

{

dataGridView1.Visible = false;

}

}

}

private int vrati\_id\_auta\_priblizno(string tablica, string marka)

{

int id = 0;

try

{

string sql3 = "select ?id? from ?Baza\_automobila? where ?"+tablica+"? ~\* '" + marka + "'";

id = Convert.ToInt32(classSQL.select(sql3, "id").Tables[0].Rows[0][0].ToString());

if (marka == "")

{

id = 0;

}

}

catch { }

return id;

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

string top = "";

string remote = "";

string where = "";

if (classSQL.remoteConnectionString != "")

{

remote = " LIMIT 500";

where = " AND ?id\_baza\_automobila? = '" + vrati\_id\_auta\_priblizno("marka", textBox1.Text) + "' ";

}

else

{

where = "AND ?id\_baza\_automobila? LIKE '%" + vrati\_id\_auta\_priblizno("marka", textBox1.Text) + "%'";

top = " TOP(500) ";

}

string sql = "";

if (vrati\_id\_auta\_priblizno("marka", textBox1.Text) != 0)

{

sql = "SELECT ?Baza\_dijelova?.id, ?Baza\_automobila?.marka, ?Baza\_automobila?.tip,?Baza\_dijelova?.naziv, ?Baza\_dijelova?.id\_grupa, " +

" ?Baza\_dijelova?.cijena, ?Baza\_dijelova?.id\_baza\_automobila FROM ?Baza\_dijelova?, ?Baza\_automobila? WHERE " +

" ?Baza\_dijelova?.id\_baza\_automobila = ?Baza\_automobila?.id " + where + " ORDER BY 1 " + remote + "";

}

else

{

sql = "SELECT ?Baza\_dijelova?.id, ?Baza\_automobila?.marka, ?Baza\_automobila?.tip,?Baza\_dijelova?.naziv, ?Baza\_dijelova?.id\_grupa, " +

" ?Baza\_dijelova?.cijena, ?Baza\_dijelova?.id\_baza\_automobila FROM ?Baza\_dijelova?, ?Baza\_automobila? WHERE " +

" ?Baza\_dijelova?.id\_baza\_automobila = ?Baza\_automobila?.id ";

}

DataTable trazi = classSQL.select(sql, "marka").Tables[0];

dataGridView1.DataSource = trazi;

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

DataGridViewColumn column\_id = dataGridView1.Columns[0];

column\_id.Visible = false;

DataGridViewColumn column\_marka = dataGridView1.Columns[1];

column\_marka.HeaderText = "Marka vozila";

DataGridViewColumn column\_tip = dataGridView1.Columns[2];

column\_tip.HeaderText = "Tip vozila";

DataGridViewColumn column\_naziv = dataGridView1.Columns[3];

column\_naziv.HeaderText = "Naziv dijela";

DataGridViewColumn column\_id\_grupa = dataGridView1.Columns[4];

column\_id\_grupa.Visible = false;

DataGridViewColumn column\_cijena = dataGridView1.Columns[5];

column\_cijena.HeaderText = "Cjena dijela";

PaintRows(dataGridView1);

}

private void textBox2\_TextChanged(object sender, EventArgs e)

{

string top = "";

string remote = "";

string where = "";

if (classSQL.remoteConnectionString != "")

{

remote = " LIMIT 500";

where = " AND ?id\_baza\_automobila? = '" + vrati\_id\_auta\_priblizno("tip",textBox2.Text) + "' ";

}

else

{

where = "AND ?id\_baza\_automobila? LIKE '%" + vrati\_id\_auta\_priblizno("tip", textBox2.Text) + "%'";

top = " TOP(500) ";

}

string sql = "";

if (vrati\_id\_auta\_priblizno("tip", textBox2.Text) != 0)

{

sql = "SELECT ?Baza\_dijelova?.id, ?Baza\_automobila?.marka, ?Baza\_automobila?.tip,?Baza\_dijelova?.naziv, ?Baza\_dijelova?.id\_grupa, " +

" ?Baza\_dijelova?.cijena, ?Baza\_dijelova?.id\_baza\_automobila FROM ?Baza\_dijelova?, ?Baza\_automobila? WHERE " +

" ?Baza\_dijelova?.id\_baza\_automobila = ?Baza\_automobila?.id " + where + " ORDER BY 1 " + remote + "";

}

else

{

sql = "SELECT ?Baza\_dijelova?.id, ?Baza\_automobila?.marka, ?Baza\_automobila?.tip,?Baza\_dijelova?.naziv, ?Baza\_dijelova?.id\_grupa, " +

" ?Baza\_dijelova?.cijena, ?Baza\_dijelova?.id\_baza\_automobila FROM ?Baza\_dijelova?, ?Baza\_automobila? WHERE " +

" ?Baza\_dijelova?.id\_baza\_automobila = ?Baza\_automobila?.id ";

}

DataTable trazi = classSQL.select(sql, "marka").Tables[0];

dataGridView1.DataSource = trazi;

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

DataGridViewColumn column\_id = dataGridView1.Columns[0];

column\_id.Visible = false;

DataGridViewColumn column\_marka = dataGridView1.Columns[1];

column\_marka.HeaderText = "Marka vozila";

DataGridViewColumn column\_tip = dataGridView1.Columns[2];

column\_tip.HeaderText = "Tip vozila";

DataGridViewColumn column\_naziv = dataGridView1.Columns[3];

column\_naziv.HeaderText = "Naziv dijela";

DataGridViewColumn column\_id\_grupa = dataGridView1.Columns[4];

column\_id\_grupa.Visible = false;

DataGridViewColumn column\_cijena = dataGridView1.Columns[5];

column\_cijena.HeaderText = "Cjena dijela";

PaintRows(dataGridView1);

}

private void textBox4\_TextChanged(object sender, EventArgs e)

{

string top = "";

string remote = "";

string where = "";

if (classSQL.remoteConnectionString != "")

{

remote = " LIMIT 500";

where = "?naziv? ~\* '" + textBox4.Text + "'";

}

else

{

where = "?naziv? LIKE '%" + textBox4.Text + "%'";

top = " TOP(500) ";

}

DataTable trazi = classSQL.select("SELECT ?Baza\_dijelova?.id, ?Baza\_automobila?.marka, ?Baza\_automobila?.tip,?Baza\_dijelova?.naziv, ?Baza\_dijelova?.id\_grupa, " +

" ?Baza\_dijelova?.cijena, ?Baza\_dijelova?.id\_baza\_automobila FROM ?Baza\_dijelova?, ?Baza\_automobila? WHERE " +

" ?Baza\_dijelova?.id\_baza\_automobila = ?Baza\_automobila?.id AND " + where + " ORDER BY ?naziv? " + remote + "", "naziv").Tables[0];

dataGridView1.DataSource = trazi;

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

DataGridViewColumn column\_id = dataGridView1.Columns[0];

column\_id.Visible = false;

DataGridViewColumn column\_marka = dataGridView1.Columns[1];

column\_marka.HeaderText = "Marka vozila";

DataGridViewColumn column\_tip = dataGridView1.Columns[2];

column\_tip.HeaderText = "Tip vozila";

DataGridViewColumn column\_naziv = dataGridView1.Columns[3];

column\_naziv.HeaderText = "Naziv dijela";

DataGridViewColumn column\_id\_grupa = dataGridView1.Columns[4];

column\_id\_grupa.Visible = false;

DataGridViewColumn column\_cijena = dataGridView1.Columns[5];

column\_cijena.HeaderText = "Cjena dijela";

PaintRows(dataGridView1);

}

private void btndodaj\_Click(object sender, EventArgs e)

{

edit\_djelovi ed\_dj = new edit\_djelovi();

ed\_dj.ShowDialog();

}

private void btnizmijeni\_Click(object sender, EventArgs e)

{

int col = dataGridView1.CurrentCell.ColumnIndex;

int row = dataGridView1.CurrentCell.RowIndex;

edit\_djelovi eddj = new edit\_djelovi();

eddj.edit = true;

eddj.id\_dijela = Convert.ToInt32(dataGridView1.Rows[row].Cells[0].FormattedValue) ;

eddj.ShowDialog();

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

int col = dataGridView1.CurrentCell.ColumnIndex;

int row = dataGridView1.CurrentCell.RowIndex;

string celija\_id = dataGridView1.Rows[row].Cells[3].FormattedValue.ToString();

string celija\_prezime = dataGridView1.Rows[row].Cells[1].FormattedValue.ToString();

}

private void button1\_Click(object sender, EventArgs e)

{

int row = dataGridView1.CurrentCell.RowIndex;

string id = dataGridView1.Rows[row].Cells[0].FormattedValue.ToString();

string naziv = dataGridView1.Rows[row].Cells[3].FormattedValue.ToString();

string marka = dataGridView1.Rows[row].Cells[1].FormattedValue.ToString();

string tip = dataGridView1.Rows[row].Cells[2].FormattedValue.ToString();

if (MessageBox.Show("Da li ste sigurni da želite obrisati dio " + naziv + " za automobil " + marka + " " + tip + " ", "Upozorenje", MessageBoxButtons.YesNo, MessageBoxIcon.Information) == DialogResult.Yes)

{

string sql = "Delete from ?Baza\_dijelova? where ?id? = '" + id + "'";

classSQL.delete(sql);

MessageBox.Show("Zapis obrisan !");

}

}

private void dataGridView1\_CellClick(object sender, DataGridViewCellEventArgs e)

{

if (edit\_nalog == true)

{

try

{

int br = e.RowIndex;

Properties.Settings.Default.id\_dio = dataGridView1.Rows[br].Cells[0].FormattedValue.ToString();

Properties.Settings.Default.Save();

this.Close();

}

catch (Exception)

{

}

}

}

private void pic\_ButtonEnter(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w + 3, h + 3);

}

private void pic\_ButtonLeave(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w - 3, h - 3);

}

private void button2\_Click(object sender, EventArgs e)

{

Grupa frp = new Grupa();

frp.ShowDialog();

}

}

Forma „Baza\_stranki“

namespace Aplikacija\_za\_zavrsni

{

public partial class baza\_stranki : Form

{

public baza\_stranki()

{

InitializeComponent();

}

DataTable DTgrid = new DataTable();

public bool edit\_nalog { get; set; }

public void puni\_grid()

{

string sql1 = "select \* from stranke\_podaci";

DTgrid = classSQL.select(sql1, "Table").Tables[0];

dataGridView1.DataSource = DTgrid;

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

DataGridViewColumn column\_id = dataGridView1.Columns[0];

column\_id.Visible = false;

DataGridViewColumn column\_email = dataGridView1.Columns[5];

column\_email.HeaderText = "Email stranke";

}

private void PaintRows(DataGridView dg)

{

int br = 0;

for (int i = 0; i < dg.Rows.Count; i++)

{

if (br == 0)

{

dg.Rows[i].DefaultCellStyle.BackColor = Color.AliceBlue;

br++;

}

else

{

dg.Rows[i].DefaultCellStyle.BackColor = Color.White;

br = 0;

}

}

DataGridViewRow row = dg.RowTemplate;

row.Height = 25;

}

private void baza\_stranki\_Load(object sender, EventArgs e)

{

puni\_grid();

PaintRows(dataGridView1);

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

string top = "";

string remote = "";

string where = "";

if (classSQL.remoteConnectionString != "")

{

remote = " LIMIT 500";

where = "?Ime? ~\* '" + textBox1.Text + "'";

}

else

{

where = "?Ime? LIKE '%" + textBox1.Text + "%'";

top = " TOP(500) ";

}

DataTable trazi = classSQL.select("SELECT \* FROM stranke\_podaci WHERE " + where + " ORDER BY ?Ime? " + remote + "", "Imena").Tables[0];

dataGridView1.DataSource = trazi;

DataGridViewColumn column\_id = dataGridView1.Columns[0];

column\_id.Visible = false;

DataGridViewColumn column\_email = dataGridView1.Columns[5];

column\_email.HeaderText = "Email stranke";

PaintRows(dataGridView1);

}

private void textBox2\_TextChanged(object sender, EventArgs e)

{

string top = "";

string remote = "";

string where = "";

if (classSQL.remoteConnectionString != "")

{

remote = " LIMIT 500";

where = "?Prezime? ~\* '" + textBox2.Text + "'";

}

else

{

where = "?Prezime? LIKE '%" + textBox2.Text + "%'";

top = " TOP(500) ";

}

DataTable trazi = classSQL.select("SELECT \* FROM stranke\_podaci WHERE " + where + " ORDER BY ?Prezime? " + remote + "", "Imena").Tables[0];

dataGridView1.DataSource = trazi;

DataGridViewColumn column\_id = dataGridView1.Columns[0];

column\_id.Visible = false;

DataGridViewColumn column\_email = dataGridView1.Columns[5];

column\_email.HeaderText = "Email stranke";

PaintRows(dataGridView1);

}

private void button1\_Click(object sender, EventArgs e)

{

int col = dataGridView1.CurrentCell.ColumnIndex;

int row = dataGridView1.CurrentCell.RowIndex;

string celija\_ime = dataGridView1.Rows[row].Cells[0].FormattedValue.ToString();

string celija\_prezime = dataGridView1.Rows[row].Cells[1].FormattedValue.ToString();

Edit\_form edit = new Edit\_form();

edit.id\_stranke = Convert.ToInt32(celija\_ime);

edit.edit = true;

edit.ShowDialog();

}

private void dataGridView1\_CellMouseClick(object sender, DataGridViewCellMouseEventArgs e)

{

}

private void button2\_Click(object sender, EventArgs e)

{

Edit\_form edit1 = new Edit\_form();

edit1.edit = false;

edit1.ShowDialog();

}

private void dataGridView1\_CellClick(object sender, DataGridViewCellEventArgs e)

{

if (edit\_nalog == true)

{

try

{

int br = e.RowIndex;

Properties.Settings.Default.id\_stranka = Convert.ToInt32(dataGridView1.Rows[br].Cells[0].FormattedValue.ToString());

Properties.Settings.Default.Save();

this.Close();

}

catch (Exception)

{

}

}

}

private void pic\_ButtonEnter(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w + 3, h + 3);

}

private void pic\_ButtonLeave(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w - 3, h - 3);

}

}

}

Klasa „claasConnectDatabase.cs“

namespace Aplikacija\_za\_zavrsni.SQL

{

class claasConnectDatabase

{

public static string GetRemoteConnectionString()

{

string server = "", port = "", username = "", password = "", database = "";

string path = Path.GetDirectoryName(Application.ExecutablePath) + "\\settings.xml";

DataSet DS = new DataSet();

DS.ReadXml(path);

server = "localhost";

port = "5432";

username = "postgres";

database = "Glavna";

return "Server=" + server + ";Port=" + port + ";" + "User Id=" + username + ";Password=" + password + ";Database=" + database + ";";

}

public static bool TestRemoteConnection()

{

NpgsqlConnection conn = new NpgsqlConnection(GetRemoteConnectionString());

try

{

conn.Open();

return true;

}

catch (Exception) { return false; }

}

}

}

Klasa „classSQL.cs“

namespace Aplikacija\_za\_zavrsni

{

class classSQL

{

public static DataSet dataSet = new DataSet();

public static NpgsqlDataAdapter Npgadapter = new NpgsqlDataAdapter();

public static string remoteConnectionString = SQL.claasConnectDatabase.GetRemoteConnectionString();

public static NpgsqlConnection remoteConnection = new NpgsqlConnection(remoteConnectionString);

//SELECT

public static DataSet select(string sql, string table)

{

dataSet = new DataSet();

if (remoteConnection.State.ToString() == "Closed") { remoteConnection.Open(); }

NpgsqlDataAdapter da = new NpgsqlDataAdapter(sql.Replace("+", "||").Replace("[", "\"").Replace("]", "\"").Replace("zbroj", "+").Replace("?", "\""), remoteConnection);

dataSet.Reset();

da.Fill(dataSet);

return dataSet;

}

//INSERT

public static string insert(string sql)

{

try

{

if (remoteConnection.State.ToString() == "Closed") { remoteConnection.Open(); }

NpgsqlCommand comm = new NpgsqlCommand(sql.Replace("+", "||").Replace("[", "\"").Replace("]", "\"").Replace("zbroj", "+").Replace("?", "\""), remoteConnection);

comm.ExecuteNonQuery();

return "";

}

catch (Exception ex)

{

remoteConnection.Close();

return ex.ToString();

}

}

//UPDATE

public static string update(string sql)

{

try

{

if (remoteConnection.State.ToString() == "Closed") { remoteConnection.Open(); }

NpgsqlCommand comm = new NpgsqlCommand(sql.Replace("+", "||").Replace("[", "\"").Replace("]", "\"").Replace("zbroj", "+").Replace("?", "\""), remoteConnection);

comm.ExecuteNonQuery();

return "";

}

catch (Exception ex)

{

remoteConnection.Close();

return ex.ToString();

}

}

//DELETE

public static string delete(string sql)

{

try

{

if (remoteConnection.State.ToString() == "Closed") { remoteConnection.Open(); }

NpgsqlCommand comm = new NpgsqlCommand(sql.Replace("+", "||").Replace("[", "\"").Replace("]", "\"").Replace("zbroj", "+").Replace("?", "\""), remoteConnection);

comm.ExecuteNonQuery();

return "";

}

catch (Exception ex)

{

remoteConnection.Close();

return ex.ToString();

}

}

}

}

Forma „edit\_auti“

namespace Aplikacija\_za\_zavrsni

{

public partial class edit\_auti : Form

{

public edit\_auti()

{

InitializeComponent();

}

public bool edit { get; set; }

public int id\_auta { get; set; }

private void btnspremi\_Click(object sender, EventArgs e)

{

if (edit == true)

{

try

{

string sql = "Update ?Baza\_automobila? Set ?marka? = '" + txtmarka.Text + "'," +

" ?tip? = '" + txttip.Text + "' WHERE ?id? = '" + id\_auta + "'";

classSQL.update(sql);

MessageBox.Show("Uspješno spremljeno !");

if (System.Windows.Forms.Application.OpenForms["Baza\_auta"] != null)

{

(System.Windows.Forms.Application.OpenForms["Baza\_auta"] as Baza\_auta).popuni\_grid();

}

this.Close();

}

catch

{

MessageBox.Show("Greška kod spremanja !");

}

}

else

{

try

{

if (txtmarka.Text != "" || txttip.Text != "")

{

string sql = "INSERT INTO ?Baza\_automobila?(?marka?, ?tip?) VALUES('" + txtmarka.Text + "', " +

"'" + txttip.Text + "')";

classSQL.insert(sql);

txtmarka.Text = "";

txttip.Text = "";

MessageBox.Show("Uspješno ste spremili zapis !");

if (System.Windows.Forms.Application.OpenForms["Baza\_auta"] != null)

{

(System.Windows.Forms.Application.OpenForms["Baza\_auta"] as Baza\_auta).popuni\_grid();

}

this.Close();

}

else

{

MessageBox.Show("Niste popunili potrebna polja !");

}

}

catch

{

MessageBox.Show("Greška kod spremanja !");

}

}

}

private void edit\_auti\_Load(object sender, EventArgs e)

{

if (edit == true)

{

lblnaslov.Text = "Izmjena podataka za vozilo";

btnspremi.Text = "Spremi";

puni\_edit();

}

else

{

lblnaslov.Text = "Dodavanje novog vozila";

btnspremi.Text = "Dodaj";

}

}

private void puni\_edit()

{

DataTable DTtemp = new DataTable();

string sql = "select \* from ?Baza\_automobila? WHERE ?id? = '" + id\_auta + "'";

DTtemp = classSQL.select(sql, "temp").Tables[0];

txtmarka.Text = DTtemp.Rows[0]["marka"].ToString();

txttip.Text = DTtemp.Rows[0]["tip"].ToString();

}

private void pic\_ButtonEnter(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w + 3, h + 3);

}

private void pic\_ButtonLeave(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w - 3, h - 3);

}

}

}

Forma „edit\_djelovi“

namespace Aplikacija\_za\_zavrsni

{

public partial class edit\_djelovi : Form

{

public edit\_djelovi()

{

InitializeComponent();

}

public bool edit { get; set; }

public int id\_dijela { get; set; }

private void edit\_djelovi\_Load(object sender, EventArgs e)

{

puni\_combo();

if (edit == true)

{

btnspremi.Text = "Spremi";

puni\_edit();

}

else

{

btnspremi.Text = "Dodaj";

}

}

private void puni\_edit()

{

DataTable DTtemp = new DataTable();

string sql = "SELECT ?Baza\_dijelova?.id, ?Baza\_automobila?.marka, ?Baza\_automobila?.tip,?Baza\_dijelova?.naziv, ?Baza\_dijelova?.id\_grupa, " +

" ?Baza\_dijelova?.cijena, ?Baza\_dijelova?.id\_baza\_automobila FROM ?Baza\_dijelova?, ?Baza\_automobila? WHERE " +

" ?Baza\_dijelova?.id\_baza\_automobila = ?Baza\_automobila?.id AND ?Baza\_dijelova?.id = '" + id\_dijela + "'";

DTtemp = classSQL.select(sql,"temp").Tables[0];

comboBox1.Text = DTtemp.Rows[0]["marka"].ToString();

comboBox2.Text = DTtemp.Rows[0]["tip"].ToString();

comboBox3.SelectedValue = Convert.ToInt32(DTtemp.Rows[0]["id\_grupa"].ToString());

txtnaziv\_dijela.Text = DTtemp.Rows[0]["naziv"].ToString();

txtcijena.Text = DTtemp.Rows[0]["cijena"].ToString();

}

private void puni\_combo()

{

DataTable DTSK = new DataTable("odabir");

DataTable BDDT = new DataTable("odabir1");

string sql1 = "select \* from ?Grupa?";

DTSK = classSQL.select(sql1, "tablice").Tables[0];

comboBox3.DataSource = DTSK;

comboBox3.DisplayMember = "ime\_grupe";

comboBox3.ValueMember = "id\_grupa";

string sql2 = "select \* from ?Baza\_automobila?";

BDDT = classSQL.select(sql2, "djelovi").Tables[0];

comboBox1.DataSource = BDDT;

comboBox1.DisplayMember = "marka";

comboBox1.ValueMember = "id";

DataTable BDDT2 = new DataTable();

try

{

if (comboBox1.Text != "" && comboBox1.SelectedValue != null)

{

string sql3 = "select \* from ?Baza\_automobila? where ?id? = '" + comboBox1.SelectedValue.ToString() + "'";

BDDT2 = classSQL.select(sql3, "auti").Tables[0];

comboBox2.DataSource = BDDT2;

comboBox2.DisplayMember = "tip";

comboBox2.ValueMember = "id";

}

}

catch

{ }

}

private int vrati\_id\_auta()

{

string sql3 = "select ?id? from ?Baza\_automobila? where ?marka? = '" + comboBox1.Text + "' and ?tip? = '" + comboBox2.Text + "'";

int id = Convert.ToInt32(classSQL.select(sql3, "id").Tables[0].Rows[0][0].ToString());

return id;

}

private void btnspremi\_Click(object sender, EventArgs e)

{

if (edit == false)

{

try

{

int vrj\_id = vrati\_id\_auta();

string sql = "INSERT INTO ?Baza\_dijelova?(?naziv?,?id\_grupa?, ?cijena?, ?id\_baza\_automobila?) " +

" VALUES('" + txtnaziv\_dijela.Text + "', '" + comboBox3.SelectedValue.ToString() + "', '" + txtcijena.Text + "', '" + vrj\_id + "')";

classSQL.insert(sql);

MessageBox.Show("Uspješno dodan zapis !");

if (System.Windows.Forms.Application.OpenForms["Baza\_djelova"] != null)

{

(System.Windows.Forms.Application.OpenForms["Baza\_djelova"] as Baza\_djelova).popuni\_grid();

}

this.Close();

}

catch

{

MessageBox.Show("Greška kod spremanja !");

}

}

else if (edit == true)

{

try

{

int vrj\_id1 = vrati\_id\_auta();

string sql1 = "update ?Baza\_dijelova? SET ?naziv? = '" + txtnaziv\_dijela.Text + "', ?id\_grupa? = '" + comboBox3.SelectedValue.ToString() + "', ?cijena? = '" + txtcijena.Text + "', ?id\_baza\_automobila? = '" + vrj\_id1 + "' WHERE ?id? = '" + id\_dijela + "'";

classSQL.update(sql1);

MessageBox.Show("Uspješno promijenjen zapis !");

if (System.Windows.Forms.Application.OpenForms["Baza\_djelova"] != null)

{

(System.Windows.Forms.Application.OpenForms["Baza\_djelova"] as Baza\_djelova).popuni\_grid();

}

this.Close();

}

catch

{

MessageBox.Show("Greška kod spremanja !");

}

}

}

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

DataTable BDDT2 = new DataTable();

try

{

if (comboBox1.Text != "" && comboBox1.SelectedValue != null)

{

string sql3 = "select \* from ?Baza\_automobila? where ?id? = '" + comboBox1.SelectedValue.ToString() + "'";

BDDT2 = classSQL.select(sql3, "auti").Tables[0];

comboBox2.DataSource = BDDT2;

comboBox2.DisplayMember = "tip";

comboBox2.ValueMember = "id";

}

}

catch

{ }

}

private void pic\_ButtonEnter(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w + 3, h + 3);

}

private void pic\_ButtonLeave(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w - 3, h - 3);

}

private void txtcijena\_KeyPress(object sender, KeyPressEventArgs e)

{

if (sender.ToString().Length == 0) return;

if (sender.ToString()[0] == '+')

{

if ((char)('+') == (e.KeyChar)) return;

}

if (sender.ToString()[0] == '-')

{

if ((char)('-') == (e.KeyChar)) return;

}

if ((char)(',') == (e.KeyChar))

{

e.Handled = false; return;

}

if (!Char.IsNumber(e.KeyChar) && !Char.IsControl(e.KeyChar))

{

e.Handled = true;

}

}

private void txtcijena\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Oemcomma || e.KeyCode == (Keys)110)

{

if (txtcijena.Text.Count(x => x == ',') > 0)

{

e.SuppressKeyPress = true;

}

}

}

private void txtcijena\_Validating(object sender, CancelEventArgs e)

{

if (txtcijena.Text.Count(x => x == ',') > 0)

{

bool val = IsValidNumber(txtcijena.Text);

if (val == false)

{

MessageBox.Show("Cijena mora biti formata '19,99'");

txtcijena.Focus();

}

if (txtcijena.Text.Substring(0, 1) == ",")

{

MessageBox.Show("Cijena mora biti formata '19,99'");

txtcijena.Focus();

}

}

else

{

txtcijena.Text = txtcijena.Text + ",00";

}

}

private bool IsValidNumber(string number)

{

int position = number.IndexOf(",");

if (position != -1)

{

int decdigits = number.Substring(position + 1, number.Length - number.IndexOf(',') - 1).Length;

if (decdigits > 0)

{

return true;

}

else

{

return false;

}

}

return false;

}

}

}

Forma „Edit\_form“

namespace Aplikacija\_za\_zavrsni

{

public partial class Edit\_form : Form

{

public Edit\_form()

{

InitializeComponent();

}

public int id\_stranke { get; set; }

public bool edit { get; set; }

DataTable podaci = new DataTable ();

private void Edit\_form\_Load(object sender, EventArgs e)

{

if (edit == true)

{

btndodaj.Text = "Spremi";

btnobrisi.Visible = true;

btnvozila.Visible = true;

}

else

{

btndodaj.Text = "Dodaj novu stranku";

btnobrisi.Visible = false;

btnvozila.Visible = false;

}

string sql = "Select \* from stranke\_podaci where id = '"+id\_stranke+"'";

podaci = classSQL.select(sql, "podaci").Tables[0];

if (edit == true)

{

popuni\_polja();

}

}

private void popuni\_polja()

{

id.Text = podaci.Rows[0]["id"].ToString();

txtime.Text = podaci.Rows[0]["Ime"].ToString();

txtprezime.Text = podaci.Rows[0]["Prezime"].ToString();

txttvrtka.Text = podaci.Rows[0]["Tvrtka"].ToString();

txtoib.Text = podaci.Rows[0]["OIB"].ToString();

txtemail.Text = podaci.Rows[0]["email"].ToString();

}

private void button1\_Click(object sender, EventArgs e)

{

DataTable vozila = new DataTable();

string sql = "Select \* from ?Vozila? where id = '" + id\_stranke + "'";

vozila = classSQL.select(sql,"vozila").Tables[0];

vozila voz = new vozila();

voz.id\_vozila = Convert.ToInt32(id.Text);

voz.ShowDialog();

}

private void button3\_Click(object sender, EventArgs e)

{

if (edit == false)

{

try

{

if (txtime.Text != "" || txtprezime.Text != "" || txtoib.Text != "")

{

string sql = "INSERT INTO stranke\_podaci(?Ime?, ?Prezime?, ?Tvrtka?, ?OIB?, ?email?) VALUES('" + txtime.Text + "', " +

"'" + txtprezime.Text + "','" + txttvrtka.Text + "','" + txtoib.Text + "','" + txtemail.Text + "')";

classSQL.insert(sql);

txtime.Text = "";

txtprezime.Text = "";

txttvrtka.Text = "";

txtoib.Text = "";

txtemail.Text = "";

MessageBox.Show("Uspješno ste spremili zapis !");

if (System.Windows.Forms.Application.OpenForms["baza\_stranki"] != null)

{

(System.Windows.Forms.Application.OpenForms["baza\_stranki"] as baza\_stranki).puni\_grid();

}

this.Close();

}

else

{

MessageBox.Show("Niste popunili potrebna polja ! (Ime, Prezime ili Oib)");

}

}

catch

{

MessageBox.Show("Greška kod spremanja !");

}

}

else

{

string sql = "Update stranke\_podaci SET ?Ime? = '" + txtime.Text + "', ?Prezime? = '" + txtprezime.Text + "' "+

", ?Tvrtka? = '" + txttvrtka.Text + "', ?OIB? = '" + txtoib.Text + "', ?email? = '" + txtemail.Text + "' WHERE ?id? = '" + id\_stranke + "'";

classSQL.update(sql);

MessageBox.Show("Uspiješno promijenjeni podaci ! ");

if (System.Windows.Forms.Application.OpenForms["baza\_stranki"] != null)

{

(System.Windows.Forms.Application.OpenForms["baza\_stranki"] as baza\_stranki).puni\_grid();

}

this.Close();

}

}

private void btnobrisi\_Click(object sender, EventArgs e)

{

if (id.Text != "")

{

if (MessageBox.Show("Da li ste sigurni da želite obrisati stranku "+ txtime.Text +" "+ txtprezime.Text +"", "Upozorenje", MessageBoxButtons.YesNo, MessageBoxIcon.Information) == DialogResult.Yes)

{

string sql = "Delete from stranke\_podaci where id = '" + id\_stranke + "'";

classSQL.delete(sql);

MessageBox.Show("Zapis obrisan !");

if (System.Windows.Forms.Application.OpenForms["baza\_stranki"] != null)

{

(System.Windows.Forms.Application.OpenForms["baza\_stranki"] as baza\_stranki).puni\_grid();

}

this.Close();

}

}

}

private void pic\_ButtonEnter(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w + 3, h + 3);

}

private void pic\_ButtonLeave(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w - 3, h - 3);

}

}

}

Forma „edit\_grupa“

namespace Aplikacija\_za\_zavrsni

{

public partial class Edit\_grupa : Form

{

public Edit\_grupa()

{

InitializeComponent();

}

public bool edit { get; set; }

public int id\_grupa { get; set; }

private void Edit\_grupa\_Load(object sender, EventArgs e)

{

if (edit == true)

{

lblnaslov.Text = "Izmjena grupe";

btnspremi.Text = "Spremi";

puni\_edit();

}

else

{

lblnaslov.Text = "Dodavanje grupe";

btnspremi.Text = "Dodaj";

}

}

private void btnspremi\_Click(object sender, EventArgs e)

{

if (edit == true)

{

try

{

string sql = "Update ?Grupa? Set ?ime\_grupe? = '" + txtgrupa.Text + "'" +

" WHERE ?id\_grupa? = '" + id\_grupa + "'";

classSQL.update(sql);

MessageBox.Show("Uspješno spremljeno !");

if (System.Windows.Forms.Application.OpenForms["Grupa"] != null)

{

(System.Windows.Forms.Application.OpenForms["Grupa"] as Grupa).popuni\_grid();

}

this.Close();

}

catch

{

MessageBox.Show("Greška kod spremanja !");

}

}

else

{

try

{

if (txtgrupa.Text != "" )

{

string sql = "INSERT INTO ?Grupa?(?ime\_grupe?) VALUES('" + txtgrupa.Text + "')";

classSQL.insert(sql);

txtgrupa.Text = "";

MessageBox.Show("Uspješno ste spremili zapis !");

if (System.Windows.Forms.Application.OpenForms["Grupa"] != null)

{

(System.Windows.Forms.Application.OpenForms["Grupa"] as Grupa).popuni\_grid();

}

this.Close();

}

else

{

MessageBox.Show("Niste popunili potrebna polja !");

}

}

catch

{

MessageBox.Show("Greška kod spremanja !");

}

}

}

private void puni\_edit()

{

DataTable DTtemp = new DataTable();

string sql = "select \* from ?Grupa? WHERE ?id\_grupa? = '" + id\_grupa + "'";

DTtemp = classSQL.select(sql, "temp").Tables[0];

txtgrupa.Text = DTtemp.Rows[0]["ime\_grupe"].ToString();

}

private void pic\_ButtonEnter(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w + 3, h + 3);

}

private void pic\_ButtonLeave(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w - 3, h - 3);

}

}

}

Forma „edit\_serviser“

namespace Aplikacija\_za\_zavrsni

{

public partial class edit\_serviser : Form

{

public edit\_serviser()

{

InitializeComponent();

}

public bool edit { get; set; }

public int id\_servis { get; set; }

private void edit\_serviser\_Load(object sender, EventArgs e)

{

if (edit == true)

{

lblnaslov.Text = "Izmjena podataka za vozilo";

btnspremi.Text = "Spremi";

puni\_edit();

}

else

{

lblnaslov.Text = "Dodavanje novog vozila";

btnspremi.Text = "Dodaj";

}

}

private void btnspremi\_Click(object sender, EventArgs e)

{

if (edit == true)

{

try

{

string sql = "Update ?Serviser? Set ?Ime? = '" + txtime.Text + "'," +

" ?Prezime? = '" + txtprezime.Text + "', ?Adresa? = '" + txtadresa.Text + "', ?OIB? = '" + txtoib.Text + "' WHERE ?id\_serv? = '" + id\_servis + "'";

classSQL.update(sql);

MessageBox.Show("Uspješno spremljeno !");

if (System.Windows.Forms.Application.OpenForms["Serviseri"] != null)

{

(System.Windows.Forms.Application.OpenForms["Serviseri"] as Serviseri).popuni\_grid();

}

this.Close();

}

catch

{

MessageBox.Show("Greška kod spremanja !");

}

}

else

{

try

{

if (txtime.Text != "" || txtprezime.Text != "")

{

string sql = "INSERT INTO ?Serviser?(?Ime?, ?Prezime?, ?Adresa?, ?OIB?) VALUES('" + txtime.Text + "', " +

"'" + txtprezime.Text + "', '" + txtadresa.Text + "', '" + txtoib.Text + "')";

classSQL.insert(sql);

txtime.Text = "";

txtprezime.Text = "";

MessageBox.Show("Uspješno ste spremili zapis !");

if (System.Windows.Forms.Application.OpenForms["Serviseri"] != null)

{

(System.Windows.Forms.Application.OpenForms["Serviseri"] as Serviseri).popuni\_grid();

}

this.Close();

}

else

{

MessageBox.Show("Niste popunili potrebna polja !");

}

}

catch

{

MessageBox.Show("Greška kod spremanja !");

}

}

}

private void puni\_edit()

{

DataTable DTtemp = new DataTable();

string sql = "select \* from ?Serviser? WHERE ?id\_serv? = '" + id\_servis + "'";

DTtemp = classSQL.select(sql, "temp").Tables[0];

txtime.Text = DTtemp.Rows[0]["Ime"].ToString();

txtprezime.Text = DTtemp.Rows[0]["Prezime"].ToString();

txtadresa.Text = DTtemp.Rows[0]["Adresa"].ToString();

txtoib.Text = DTtemp.Rows[0]["OIB"].ToString();

}

private void pic\_ButtonEnter(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w + 3, h + 3);

}

private void pic\_ButtonLeave(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w - 3, h - 3);

}

}

}

Forma „edit\_vozila“

namespace Aplikacija\_za\_zavrsni

{

public partial class Edit\_vozila : Form

{

public Edit\_vozila()

{

InitializeComponent();

}

public bool edit { get; set; }

public string id\_vozila { get; set; }

public string id\_stranka { get; set; }

private void Edit\_vozila\_Load(object sender, EventArgs e)

{

if (edit == true)

{

puni\_combo();

eidt\_fill();

}

else

{

puni\_combo();

}

}

private void comboBox1\_SelectedValueChanged(object sender, EventArgs e)

{

DataTable BDDT2 = new DataTable();

try

{

if (comboBox1.Text != "" && comboBox1.SelectedValue != null)

{

string sql3 = "select \* from ?Baza\_automobila? where ?id? = '" + comboBox1.SelectedValue.ToString() + "'";

BDDT2 = classSQL.select(sql3, "auti").Tables[0];

comboBox2.DataSource = BDDT2;

comboBox2.DisplayMember = "tip";

comboBox2.ValueMember = "id";

}

}

catch

{ }

}

private void puni\_combo()

{

DataTable BDDT2 = new DataTable("odabir1");

DataTable BDDT = new DataTable("odabir1");

string sql2 = "select \* from ?Baza\_automobila?";

BDDT = classSQL.select(sql2, "auti").Tables[0];

if (comboBox1.SelectedValue != "" && comboBox1.SelectedValue != null)

{

string sql3 = "select \* from ?Baza\_automobila? where ?id? = '" + comboBox1.SelectedValue.ToString() + "'";

BDDT2 = classSQL.select(sql3, "auti").Tables[0];

comboBox2.DataSource = BDDT2;

comboBox2.DisplayMember = "tip";

comboBox2.ValueMember = "id";

}

comboBox1.DataSource = BDDT;

comboBox1.DisplayMember = "marka";

comboBox1.ValueMember = "id";

}

private int vrati\_id\_auta()

{

string sql3 = "select ?id? from ?Baza\_automobila? where ?marka? = '" + comboBox1.Text + "' and ?tip? = '" + comboBox2.Text + "'";

int id = Convert.ToInt32(classSQL.select(sql3,"id").Tables[0].Rows[0][0].ToString());

return id;

}

private void eidt\_fill()

{

lblnaslov.Visible = true;

lblnaslov.Text = "Izmjena podataka za vozilo";

groupBox1.Visible = true;

btnspremi.Text = "Spremi";

string sql1 = "SELECT ?Vozila?.id, ?Baza\_automobila?.marka, ?Baza\_automobila?.tip,?Vozila?.id\_vozila, ?Vozila?.godina\_proizv, " +

" ?Vozila?.kilometraza, ?Vozila?.datum\_zad\_serv FROM ?Vozila?, ?Baza\_automobila? WHERE " +

" ?Vozila?.id\_baza\_automobila = ?Baza\_automobila?.id AND ?Vozila?.id\_vozila = '" + id\_vozila + "' ";

DataTable DTselektirano = classSQL.select(sql1, "vozilo").Tables[0];

comboBox1.Text = DTselektirano.Rows[0]["marka"].ToString();

comboBox2.Text = DTselektirano.Rows[0]["tip"].ToString();

txtgodpro.Text = DTselektirano.Rows[0]["godina\_proizv"].ToString();

txtkm.Text = DTselektirano.Rows[0]["kilometraza"].ToString();

dtpDatum.Text = DTselektirano.Rows[0]["datum\_zad\_serv"].ToString();

}

private void btnspremi\_Click(object sender, EventArgs e)

{

if (edit == true)

{

try

{

int vrj = vrati\_id\_auta();

string sql = "Update ?Vozila? Set ?id? = '" + id\_stranka + "', " +

" ?godina\_proizv?= '" + txtgodpro.Text + "', ?kilometraza?= '" + txtkm.Text + "'," +

" ?datum\_zad\_serv?= '" + dtpDatum.Value.Date.ToString("dd-MM-yyyy 0:00:00") + "', ?id\_baza\_automobila? = '"+ vrj +"' WHERE ?id\_vozila? = '" + id\_vozila + "'";

classSQL.update(sql);

MessageBox.Show("Uspješno spremljeno !");

if (System.Windows.Forms.Application.OpenForms["vozila"] != null)

{

(System.Windows.Forms.Application.OpenForms["vozila"] as vozila).popuni\_grid();

}

this.Close();

}

catch

{

MessageBox.Show("Greška kod spremanja !");

}

}

else

{

try

{

if (comboBox1.DisplayMember.ToString() != "" || comboBox2.DisplayMember.ToString() != "" || txtkm.Text != "")

{

int vrj1 = vrati\_id\_auta();

string sql = "INSERT INTO ?Vozila?(?id?, ?godina\_proizv?, ?kilometraza?, ?datum\_zad\_serv?, ?id\_baza\_automobila?) VALUES('" + id\_stranka + "', " +

"'" + txtgodpro.Text + "','" + txtkm.Text + "', '" + dtpDatum.Value.Date.ToString("dd-MM-yyyy 0:00:00") + "', '" + vrj1 + "')";

classSQL.insert(sql);

txtgodpro.Text = "";

txtkm.Text = "";

MessageBox.Show("Uspješno ste spremili zapis !");

lblnaslov.Visible = false;

if (System.Windows.Forms.Application.OpenForms["vozila"] != null)

{

(System.Windows.Forms.Application.OpenForms["vozila"] as vozila).popuni\_grid();

}

this.Close();

}

else

{

MessageBox.Show("Niste popunili potrebna polja !");

}

}

catch

{

MessageBox.Show("Greška kod spremanja !");

}

}

}

private void pic\_ButtonEnter(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w + 3, h + 3);

}

private void pic\_ButtonLeave(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w - 3, h - 3);

}

}

}

Forma „Form1“

namespace Aplikacija\_za\_zavrsni

{

public partial class Main : Form

{

public Main()

{

InitializeComponent();

}

DataTable DTgrupa = new DataTable();

private void Form1\_Load(object sender, EventArgs e)

{

puni\_grid();

string sql = "SELECT \* FROM stranke\_podaci";

DataTable DTs = classSQL.select(sql, "podaci").Tables[0];

}

private void puni\_grid()

{

DateTime datum = Convert.ToDateTime(monthCalendar1.SelectionRange.Start.ToShortDateString());

string sql1 = "select ?id? as [Broj], ?stranka? as [Stranka], ?datum? as [Datum] from ?nalozi? where ?datum? >= '" + datum.Date.ToString("dd-MM-yyyy 00:00:00") + "' and ?datum? <= '" + datum.Date.ToString("dd-MM-yyyy 24:00:00") + "'";

DTgrupa = classSQL.select(sql1, "Table").Tables[0];

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

dataGridView1.DataSource = DTgrupa;

DataGridViewColumn column = dataGridView1.Columns[0];

DataGridViewColumn column1 = dataGridView1.Columns[1];

DataGridViewColumn column2 = dataGridView1.Columns[2];

column.Width = 29;

column1.Width = 120;

column2.Width = 150;

}

private void button1\_Click(object sender, EventArgs e)

{

baza\_stranki baz = new baza\_stranki();

baz.ShowDialog();

}

private void button3\_Click(object sender, EventArgs e)

{

Baza\_djelova baz = new Baza\_djelova();

baz.ShowDialog();

}

private void button4\_Click(object sender, EventArgs e)

{

Baza\_auta BA = new Baza\_auta();

BA.ShowDialog();

}

private void button2\_Click(object sender, EventArgs e)

{

novi\_nalog nn = new novi\_nalog();

nn.ShowDialog();

}

private void button5\_Click(object sender, EventArgs e)

{

svi\_nalozi svi = new svi\_nalozi();

svi.ShowDialog();

}

private void monthCalendar1\_DateChanged(object sender, DateRangeEventArgs e)

{

DateTime datum = Convert.ToDateTime(monthCalendar1.SelectionRange.Start.ToShortDateString());

string sql1 = "select ?stranka? as [Stranka], ?datum? as [Datum] from ?nalozi? where ?datum? >= '" + datum.Date.ToString("dd-MM-yyyy 00:00:00") + "' and ?datum? <= '" + datum.Date.ToString("dd-MM-yyyy 24:00:00") + "'";

DTgrupa = classSQL.select(sql1, "Table").Tables[0];

dataGridView1.DataSource = DTgrupa;

}

private void pic\_ButtonEnter(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w + 3, h + 3);

}

private void pic\_ButtonLeave(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w - 3, h - 3);

}

private void dataGridView1\_CellMouseDoubleClick(object sender, DataGridViewCellMouseEventArgs e)

{

}

private void btnserviseri\_Click(object sender, EventArgs e)

{

Serviseri serv = new Serviseri();

serv.ShowDialog();

}

}

}

Forma „Grupa“

namespace Aplikacija\_za\_zavrsni

{

public partial class Grupa : Form

{

public Grupa()

{

InitializeComponent();

}

DataTable DTgrid = new DataTable();

private void Grupa\_Load(object sender, EventArgs e)

{

popuni\_grid();

}

public void popuni\_grid()

{

string sql1 = "select \* from ?Grupa?";

DTgrid = classSQL.select(sql1, "Table").Tables[0];

dataGridView1.DataSource = DTgrid;

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

DataGridViewColumn column\_id = dataGridView1.Columns[0];

column\_id.Visible = false;

DataGridViewColumn column\_marka = dataGridView1.Columns[1];

column\_marka.HeaderText = "Ime grupe";

}

int col1;

int row1;

private void button1\_Click(object sender, EventArgs e)

{

}

private void btndodaj\_Click(object sender, EventArgs e)

{

Edit\_grupa ed\_dj = new Edit\_grupa();

ed\_dj.ShowDialog();

}

private void btnizmijeni\_Click(object sender, EventArgs e)

{

int col = dataGridView1.CurrentCell.ColumnIndex;

int row = dataGridView1.CurrentCell.RowIndex;

Edit\_grupa eddj = new Edit\_grupa();

eddj.edit = true;

eddj.id\_grupa = Convert.ToInt32(dataGridView1.Rows[row].Cells[0].FormattedValue);

eddj.ShowDialog();

}

private void pic\_ButtonEnter(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w + 3, h + 3);

}

private void pic\_ButtonLeave(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w - 3, h - 3);

}

private void button1\_Click\_1(object sender, EventArgs e)

{

int row = dataGridView1.CurrentCell.RowIndex;

string id = dataGridView1.Rows[row].Cells[0].FormattedValue.ToString();

string grupa = dataGridView1.Rows[row].Cells[1].FormattedValue.ToString();

if (MessageBox.Show("Da li ste sigurni da želite obrisati grupu " + grupa + " ", "Upozorenje", MessageBoxButtons.YesNo, MessageBoxIcon.Information) == DialogResult.Yes)

{

string sql = "Delete from ?Grupa? where ?id\_grupa? = '" + id + "'";

classSQL.delete(sql);

MessageBox.Show("Zapis obrisan !");

popuni\_grid();

}

}

}

}

Forma „Novi\_nalog“

namespace Aplikacija\_za\_zavrsni

{

public partial class novi\_nalog : Form

{

public novi\_nalog()

{

InitializeComponent();

}

public bool edit { get; set; }

public string broj\_naloga { get; set; }

public bool svi\_nalozi\_edit { get; set; }

private void novi\_nalog\_Load(object sender, EventArgs e)

{

puni\_combo();

edit\_sa\_svih\_naloga();

}

private void edit\_sa\_svih\_naloga()

{

if (svi\_nalozi\_edit == true)

{

DataTable DT = classSQL.select("SELECT \* FROM ?nalozi? WHERE ?id? = '" + broj\_naloga + "'", "nalozi").Tables[0];

clear\_all();

if (DT.Rows.Count == 0)

{

MessageBox.Show("Odabrani nalog ne postoji.", "Greška");

}

else if (DT.Rows.Count == 1)

{

edit = true;

btnObrisi.Enabled = true;

EnableDisable(true);

txtbrnalog.Enabled = false;

txtbrnalog.Text = DT.Rows[0]["id"].ToString();

txtPartnerNaziv1.Text = DT.Rows[0]["stranka"].ToString();

txtid\_stranka.Text = DT.Rows[0]["stranka\_id"].ToString();

dtpDatum.Text = DT.Rows[0]["datum"].ToString();

rtbxnapomena.Text = DT.Rows[0]["napomena"].ToString();

txtkm.Text = DT.Rows[0]["prijedeni\_kilometri"].ToString();

txtgodpro.Text = DT.Rows[0]["godina\_proizvodnje"].ToString();

rtbxopis.Text = DT.Rows[0]["opis\_kvara"].ToString();

dateTimePicker1.Text = DT.Rows[0]["datum\_serv"].ToString();

lblkontrola.Text = DT.Rows[0]["id\_baza\_automobila"].ToString();

if (DT.Rows[0]["id\_serviser"].ToString() != "")

{

comboserviser.SelectedValue = DT.Rows[0]["id\_serviser"].ToString();

}

else

{

comboserviser.SelectedValue = "1";

}

string sql3 = "select ?marka? from ?Baza\_automobila? where ?id? = '" + lblkontrola.Text + "'";

string sql4 = "select ?tip? from ?Baza\_automobila? where ?id? = '" + lblkontrola.Text + "'";

string marka = classSQL.select(sql3, "marka").Tables[0].Rows[0][0].ToString();

string tip = classSQL.select(sql4, "tip").Tables[0].Rows[0][0].ToString();

txtmarka.Text = marka;

txttip.Text = tip;

if (DT.Rows[0]["id\_auta"].ToString() != "")

{

txtmarka.Enabled = false;

txttip.Enabled = false;

}

pupuni\_dgw();

}

}

}

private void EnableDisable(bool x)

{

dtpDatum.Enabled = x;

txtPartnerNaziv1.Enabled = x;

rtbxnapomena.Enabled = x;

rtbxopis.Enabled = x;

txtmarka.Enabled = x;

txtdijelovi.Enabled = x;

txtgodpro.Enabled = x;

txtkm.Enabled = x;

txttip.Enabled = x;

btnPartner1.Enabled = x;

pictureBox2.Enabled = x;

btnOpenRoba.Enabled = x;

btnSpremi.Enabled = x;

dgw.Enabled = x;

dateTimePicker1.Enabled = x;

btnObrisi.Enabled = x;

btnObrisiNalog.Enabled = x;

comboserviser.Enabled = x;

}

private void clear\_all()

{

txtPartnerNaziv1.Text = "";

rtbxnapomena.Text = "";

rtbxopis.Text = "";

txtmarka.Text = "";

txtdijelovi.Text = "";

txtgodpro.Text = "";

txtkm.Text = "";

txttip.Text = "";

txtid\_stranka.Text = "";

lblkontrola.Text = "kontrola";

txtbrnalog.Text = "";

}

private void btnOdustani\_Click(object sender, EventArgs e)

{

EnableDisable(false);

txtbrnalog.Enabled = true;

clear\_all();

pictureBox1.Enabled = true;

if (Properties.Settings.Default.id\_vozilo != 0)

{

Properties.Settings.Default.id\_vozilo = 0;

}

dgw.Rows.Clear();

}

private void btnNoviUnos\_Click(object sender, EventArgs e)

{

EnableDisable(true);

txtbrnalog.Enabled = false;

pictureBox1.Enabled = false;

edit = false;

clear\_all();

string sql = "Select max(?id?) from ?nalozi?";

int broj = Convert.ToInt32(classSQL.select(sql,"broj").Tables[0].Rows[0][0].ToString());

txtbrnalog.Text = (broj + 1).ToString();

if (Properties.Settings.Default.id\_vozilo != 0)

{

Properties.Settings.Default.id\_vozilo = 0;

}

}

private void puni\_combo()

{

DataTable DTSK = new DataTable("odabir");

DataTable BDDT = new DataTable("odabir1");

string sql1 = "select ?id\_serv?, ?Ime? || ' ' || ?Prezime? as Ime from ?Serviser?";

DTSK = classSQL.select(sql1, "tablice").Tables[0];

comboserviser.DataSource = DTSK;

comboserviser.DisplayMember = "Ime";

comboserviser.ValueMember = "id\_serv";

}

private void btnPartner1\_Click(object sender, EventArgs e)

{

baza\_stranki baz\_Str = new baza\_stranki();

baz\_Str.edit\_nalog = true;

baz\_Str.ShowDialog();

if (Properties.Settings.Default.id\_stranka != 0)

{

DataSet partner = new DataSet();

partner = classSQL.select("SELECT ?Tvrtka?,?Ime?, ?Prezime?,?id? FROM ?stranke\_podaci? WHERE ?id? ='" + Properties.Settings.Default.id\_stranka + "'", "partners");

if (partner.Tables[0].Rows.Count > 0)

{

txtPartnerNaziv1.Text = partner.Tables[0].Rows[0]["Ime"].ToString() + " " + partner.Tables[0].Rows[0]["Prezime"].ToString();

txtid\_stranka.Text = partner.Tables[0].Rows[0]["id"].ToString();

}

else

{

MessageBox.Show("Upisana šifra ne postoji.", "Greška");

}

}

}

private void pictureBox2\_Click(object sender, EventArgs e)

{

if (txtid\_stranka.Text != "")

{

vozila voz = new vozila();

voz.edit\_nalog = true;

voz.id\_vozila = Convert.ToInt32(txtid\_stranka.Text);

voz.ShowDialog();

if (Properties.Settings.Default.id\_stranka != 0)

{

DataSet vozila = new DataSet();

vozila = classSQL.select("SELECT ?Vozila?.id, ?Baza\_automobila?.marka, ?Baza\_automobila?.tip,?Vozila?.id\_vozila, ?Vozila?.godina\_proizv, " +

" ?Vozila?.kilometraza, ?Vozila?.datum\_zad\_serv FROM ?Vozila?, ?Baza\_automobila? WHERE " +

" ?Vozila?.id\_baza\_automobila = ?Baza\_automobila?.id AND ?Vozila?.id\_vozila = '" + Properties.Settings.Default.id\_vozilo + "'", "vozilo");

if (vozila.Tables[0].Rows.Count > 0)

{

lblkontrola.Text = vozila.Tables[0].Rows[0]["id\_vozila"].ToString();

txtmarka.Text = vozila.Tables[0].Rows[0]["marka"].ToString();

txtmarka.Enabled = false;

txttip.Enabled = false;

txttip.Text = vozila.Tables[0].Rows[0]["tip"].ToString();

txtgodpro.Text = vozila.Tables[0].Rows[0]["godina\_proizv"].ToString();

txtkm.Text = vozila.Tables[0].Rows[0]["kilometraza"].ToString();

dateTimePicker1.Text = vozila.Tables[0].Rows[0]["datum\_zad\_serv"].ToString();

}

else

{

MessageBox.Show("Ne postoji vozilo !", "Greška");

}

}

}

else

{

MessageBox.Show("Odaberite stranku da bi ste mogli birati vozila iste !");

}

}

private int vrati\_id\_auta(string marka, string tip)

{

string sql3 = "select ?id? from ?Baza\_automobila? where ?marka? = '" + marka + "' and ?tip? = '" + tip + "'";

int id = Convert.ToInt32(classSQL.select(sql3, "id").Tables[0].Rows[0][0].ToString());

return id;

}

private void btnSpremi\_Click(object sender, EventArgs e)

{

bool promjena = false;

if (edit == false)

{

try

{

string id\_aut = "";

if (lblkontrola.Text != "kontrola")

{ id\_aut = lblkontrola.Text; }

else { id\_aut = ""; }

string sql = "Insert into ?nalozi?(" +

"?stranka?, ?datum?, ?napomena?, ?marka?, ?tip?, ?prijedeni\_kilometri?, " +

"?godina\_proizvodnje?, ?opis\_kvara?, ?datum\_serv?, ?stranka\_id?, ?id\_auta?, ?id\_serviser?, ?id\_baza\_automobila?) VALUES ('" + txtPartnerNaziv1.Text + "', '" + dtpDatum.Value + "', '" + rtbxnapomena.Text + "'," +

" '" + txtmarka.Text + "', '" + txttip.Text + "', '" + txtkm.Text + "', '" + txtgodpro.Text + "', '" + rtbxopis.Text + "', '" + dateTimePicker1.Value + "'," +

" '" + txtid\_stranka.Text + "', '" + id\_aut + "', '" + comboserviser.SelectedValue + "', '"+ lblkontrola.Text +"')";

classSQL.insert(sql);

for (int i = 0; i < dgw.Rows.Count; i++)

{

string marka = dgw.Rows[i].Cells["marka"].Value.ToString();

string tip = dgw.Rows[i].Cells["tip"].Value.ToString();

string sql\_dgw = "INSERT INTO ?dijelovi\_nalog?(?naziv?, ?id\_naloga?, ?cijena?, ?kolicina?, ?id\_baza\_automobila?) VALUES ("+

" '" + dgw.Rows[i].Cells["naziv"].Value + "', " +

" '" + txtbrnalog.Text + "', '"

+ dgw.Rows[i].Cells["cijena"].Value + "', '"

+ dgw.Rows[i].Cells["kolicina"].Value + "', '" + vrati\_id\_auta(marka, tip) + "')";

classSQL.insert(sql\_dgw);

}

MessageBox.Show("Uspješno spremljeno !");

clear\_all();

EnableDisable(false);

dgw.ClearSelection();

dgw.Rows.Clear();

}

catch

{

MessageBox.Show("Greška kod spremanja !");

}

}

else

{

string sql\_update = "update ?nalozi? set ?stranka? = '" + txtPartnerNaziv1.Text + "', ?datum? = '" + dtpDatum.Value + "', ?napomena? = '" + rtbxnapomena.Text + "' " +

" , ?prijedeni\_kilometri? = '" + txtkm.Text + "', " +

"?godina\_proizvodnje? = '" + txtgodpro.Text + "', ?opis\_kvara? = '" + rtbxopis.Text + "', ?datum\_serv? = '" + dateTimePicker1.Value + "', ?stranka\_id? = '" + txtid\_stranka.Text + "', ?id\_serviser? = '" + comboserviser.SelectedValue + "', " +

" ?id\_baza\_automobila? = '"+ lblkontrola.Text +"' WHERE ?id? = '"+ txtbrnalog.Text +"'";

classSQL.update(sql\_update);

string sql\_provjera = "SELECT ?dijelovi\_nalog?.id, ?Baza\_automobila?.marka, ?Baza\_automobila?.tip,?dijelovi\_nalog?.naziv, ?dijelovi\_nalog?.id\_naloga, " +

" ?dijelovi\_nalog?.cijena, ?dijelovi\_nalog?.kolicina, ?dijelovi\_nalog?.id\_baza\_automobila FROM ?dijelovi\_nalog?, ?Baza\_automobila? WHERE " +

" ?dijelovi\_nalog?.id\_baza\_automobila = ?Baza\_automobila?.id AND ?id\_naloga? = '" + txtbrnalog.Text + "'";

DataTable provjera = classSQL.select(sql\_provjera, "provjera").Tables[0];

for (int y = 0; y < provjera.Rows.Count; y++)

{

string cijena = provjera.Rows[y]["cijena"].ToString();

string kolicina = provjera.Rows[y]["kolicina"].ToString();

for (int z = 0; z < dgw.Rows.Count; z++)

{

string cijena\_dgw = dgw.Rows[z].Cells["cijena"].FormattedValue.ToString();

string kolicina\_dgw = dgw.Rows[z].Cells["kolicina"].FormattedValue.ToString();

if (cijena != cijena\_dgw)

{

promjena = true;

}

else if (kolicina != kolicina\_dgw)

{

promjena = true;

}

}

}

if (promjena == true)

{

classSQL.delete("Delete from ?dijelovi\_nalog? where ?id\_naloga? = '" + txtbrnalog.Text + "'");

for (int i = 0; i < dgw.Rows.Count; i++)

{

string marka = dgw.Rows[i].Cells["marka"].Value.ToString();

string tip = dgw.Rows[i].Cells["tip"].Value.ToString();

string sql\_dgw = "INSERT INTO ?dijelovi\_nalog?(?naziv?, ?id\_naloga?, ?cijena?, ?kolicina?, ?id\_baza\_automobila?) VALUES ("+

" '" + dgw.Rows[i].Cells["naziv"].Value + "', " +

" '" + txtbrnalog.Text + "', '" + dgw.Rows[i].Cells["cijena"].Value + "', '" + dgw.Rows[i].Cells["kolicina"].Value + "', '"+vrati\_id\_auta(marka, tip)+"' )";

classSQL.insert(sql\_dgw);

}

}

if (dgw.Rows.Count != provjera.Rows.Count)

{

classSQL.delete("Delete from ?dijelovi\_nalog? where ?id\_naloga? = '" + txtbrnalog.Text + "'");

for (int i = 0; i < dgw.Rows.Count; i++)

{

string marka = dgw.Rows[i].Cells["marka"].Value.ToString();

string tip = dgw.Rows[i].Cells["tip"].Value.ToString();

string sql\_dgw = "INSERT INTO ?dijelovi\_nalog?(?naziv?, ?id\_naloga?, ?cijena?, ?kolicina?, ?id\_baza\_automobila?) VALUES ("+

" '" + dgw.Rows[i].Cells["naziv"].Value + "', " +

" '" + txtbrnalog.Text + "', '" + dgw.Rows[i].Cells["cijena"].Value + "', '" + dgw.Rows[i].Cells["kolicina"].Value + "', '" + vrati\_id\_auta(marka, tip) + "' )";

classSQL.insert(sql\_dgw);

}

}

if (lblkontrola.Text != "kontrola" && lblkontrola.Text != "0")

{

string sql1 = "update ?Vozila? set ?godina\_proizv? = '" + txtgodpro.Text + "', ?kilometraza? = '" + txtkm.Text + "'" +

", ?datum\_zad\_serv? = '" + dateTimePicker1.Text + "' WHERE ?id\_vozila? = '" + lblkontrola.Text + "'";

classSQL.update(sql1);

}

MessageBox.Show("Uspiješno izmijenjeni podaci !");

clear\_all();

EnableDisable(false);

txtbrnalog.Enabled = true;

dgw.Rows.Clear();

}

}

private void txtbrnalog\_KeyPress(object sender, KeyPressEventArgs e)

{

if (sender.ToString().Length == 0) return;

if (sender.ToString()[0] == '+')

{

if ((char)('+') == (e.KeyChar)) return;

}

if (sender.ToString()[0] == '-')

{

if ((char)('-') == (e.KeyChar)) return;

}

if (!Char.IsNumber(e.KeyChar) && !Char.IsControl(e.KeyChar))

{

e.Handled = true;

}

}

private void txtbrnalog\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

DataTable DT = classSQL.select("SELECT \* FROM ?nalozi? WHERE ?id? = '" + txtbrnalog.Text + "'", "nalozi").Tables[0];

clear\_all();

if (DT.Rows.Count == 0)

{

MessageBox.Show("Odabrani nalog ne postoji.", "Greška");

}

else if (DT.Rows.Count == 1)

{

edit = true;

btnObrisi.Enabled = true;

EnableDisable(true);

txtbrnalog.Enabled = false;

txtbrnalog.Text = DT.Rows[0]["id"].ToString();

txtPartnerNaziv1.Text = DT.Rows[0]["stranka"].ToString();

txtid\_stranka.Text = DT.Rows[0]["stranka\_id"].ToString();

dtpDatum.Text = DT.Rows[0]["datum"].ToString();

rtbxnapomena.Text = DT.Rows[0]["napomena"].ToString();

txtkm.Text = DT.Rows[0]["prijedeni\_kilometri"].ToString();

txtgodpro.Text = DT.Rows[0]["godina\_proizvodnje"].ToString();

rtbxopis.Text = DT.Rows[0]["opis\_kvara"].ToString();

dateTimePicker1.Text = DT.Rows[0]["datum\_serv"].ToString();

lblkontrola.Text = DT.Rows[0]["id\_baza\_automobila"].ToString();

if (DT.Rows[0]["id\_serviser"].ToString() != "")

{

comboserviser.SelectedValue = DT.Rows[0]["id\_serviser"].ToString();

}

else

{

comboserviser.SelectedValue = "1";

}

string sql3 = "select ?marka? from ?Baza\_automobila? where ?id? = '" + lblkontrola.Text + "'";

string sql4 = "select ?tip? from ?Baza\_automobila? where ?id? = '" + lblkontrola.Text + "'";

string marka = classSQL.select(sql3, "marka").Tables[0].Rows[0][0].ToString();

string tip = classSQL.select(sql4, "tip").Tables[0].Rows[0][0].ToString();

txtmarka.Text = marka;

txttip.Text = tip;

if (DT.Rows[0]["id\_auta"].ToString() != "")

{

txtmarka.Enabled = false;

txttip.Enabled = false;

}

pupuni\_dgw();

}

}

}

private void pupuni\_dgw()

{

DataTable dtR = classSQL.select("SELECT ?dijelovi\_nalog?.id, ?Baza\_automobila?.marka, ?Baza\_automobila?.tip,?dijelovi\_nalog?.naziv, ?dijelovi\_nalog?.id\_naloga, " +

" ?dijelovi\_nalog?.cijena, ?dijelovi\_nalog?.kolicina, ?dijelovi\_nalog?.id\_baza\_automobila FROM ?dijelovi\_nalog?, ?Baza\_automobila? WHERE " +

" ?dijelovi\_nalog?.id\_baza\_automobila = ?Baza\_automobila?.id AND ?id\_naloga? ='" + txtbrnalog.Text + "'", "nalog").Tables[0];

if (dtR.Rows.Count > 0)

{

for (int i = 0; i < dtR.Rows.Count; i++)

{

int br = dgw.Rows.Count ;

dgw.Rows.Add();

dgw.Rows[br].Cells[0].Value = i + 1;

dgw.Rows[br].Cells["marka"].Value = dtR.Rows[i]["marka"].ToString();

dgw.Rows[br].Cells["tip"].Value = dtR.Rows[i]["tip"].ToString();

dgw.Rows[br].Cells["naziv"].Value = dtR.Rows[i]["naziv"].ToString();

dgw.Rows[br].Cells["cijena"].Value = dtR.Rows[i]["cijena"].ToString();

dgw.Rows[br].Cells["kolicina"].Value = dtR.Rows[i]["kolicina"].ToString();

dgw.Rows[br].Cells["id\_dijela"].Value = dtR.Rows[i]["id"].ToString();

}

}

}

private void btnOpenRoba\_Click(object sender, EventArgs e)

{

if (txtid\_stranka.Text != "")

{

Baza\_djelova dj = new Baza\_djelova();

dj.edit\_nalog = true;

dj.ShowDialog();

DataTable djelovi = new DataTable();

string sql = "SELECT ?Baza\_dijelova?.id, ?Baza\_automobila?.marka, ?Baza\_automobila?.tip,?Baza\_dijelova?.naziv, ?Baza\_dijelova?.id\_grupa, " +

" ?Baza\_dijelova?.cijena, ?Baza\_dijelova?.id\_baza\_automobila FROM ?Baza\_dijelova?, ?Baza\_automobila? WHERE " +

" ?Baza\_dijelova?.id\_baza\_automobila = ?Baza\_automobila?.id AND ?Baza\_dijelova?.id = '" + Properties.Settings.Default.id\_dio + "'";

djelovi = classSQL.select(sql, "djelovi").Tables[0];

djelovi = classSQL.select(sql, "djelovi").Tables[0];

if (djelovi.Rows.Count > 0)

{

dgw.Rows.Add(

dgw.RowCount + 1,

djelovi.Rows[0]["marka"].ToString(),

djelovi.Rows[0]["tip"].ToString(),

djelovi.Rows[0]["naziv"].ToString(),

djelovi.Rows[0]["cijena"].ToString(),

'1',

djelovi.Rows[0]["id"].ToString()

);

int br = dgw.Rows.Count - 1;

dgw.ClearSelection();

txtdijelovi.Text = "";

txtdijelovi.Select();

return;

}

else

{

MessageBox.Show("Tražena šifra ne postoji!");

txtdijelovi.Text = "";

}

}

else

{

MessageBox.Show("Odaberite stranku da bi ste mogli dodavati dijelove !");

}

}

private void txtdijelovi\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

e.SuppressKeyPress = true;

if (txtdijelovi.Text == "")

{

Baza\_djelova roba = new Baza\_djelova();

roba.ShowDialog();

if (Properties.Settings.Default.id\_dio != "")

{

txtdijelovi.Text = Properties.Settings.Default.id\_dio;

txtdijelovi.Select();

}

else

{

return;

}

}

if (txtdijelovi.Text.Length > 0)

{

DataTable djelovi = new DataTable();

string sql = "SELECT ?Baza\_dijelova?.id, ?Baza\_automobila?.marka, ?Baza\_automobila?.tip,?Baza\_dijelova?.naziv, ?Baza\_dijelova?.id\_grupa, " +

" ?Baza\_dijelova?.cijena, ?Baza\_dijelova?.id\_baza\_automobila FROM ?Baza\_dijelova?, ?Baza\_automobila? WHERE " +

" ?Baza\_dijelova?.id\_baza\_automobila = ?Baza\_automobila?.id AND ?Baza\_dijelova?.id = '" + txtdijelovi.Text + "'";

djelovi = classSQL.select(sql, "djelovi").Tables[0];

if (djelovi.Rows.Count > 0)

{

dgw.Rows.Add(

dgw.RowCount + 1,

djelovi.Rows[0]["marka"].ToString(),

djelovi.Rows[0]["tip"].ToString(),

djelovi.Rows[0]["naziv"].ToString(),

djelovi.Rows[0]["id"].ToString()

);

int br = dgw.Rows.Count - 1;

dgw.ClearSelection();

txtdijelovi.Text = "";

txtdijelovi.Select();

return;

}

else

{

MessageBox.Show("Tražena šifra ne postoji!");

txtdijelovi.Text = "";

}

}

}

}

private void txtdijelovi\_KeyPress(object sender, KeyPressEventArgs e)

{

if (sender.ToString().Length == 0) return;

if (sender.ToString()[0] == '+')

{

if ((char)('+') == (e.KeyChar)) return;

}

if (sender.ToString()[0] == '-')

{

if ((char)('-') == (e.KeyChar)) return;

}

if (!Char.IsNumber(e.KeyChar) && !Char.IsControl(e.KeyChar))

{

e.Handled = true;

}

}

private void btnObrisi\_Click(object sender, EventArgs e)

{

if (dgw.RowCount == 0)

{

return;

}

if (dgw.Rows[dgw.CurrentRow.Index].Cells["id\_dijela"].Value != null)

{

if (MessageBox.Show("Da li ste sigurni da želite obrisati stavku " + dgw.Rows[dgw.CurrentRow.Index].Cells["id"].Value + " ?", "Upozorenje", MessageBoxButtons.YesNo, MessageBoxIcon.Information) == DialogResult.Yes)

{

classSQL.delete("DELETE FROM ?dijelovi\_nalog? WHERE id\_naloga='" + txtbrnalog.Text + "' AND id = '" + dgw.Rows[dgw.CurrentRow.Index].Cells["id\_dijela"].Value + "'");

dgw.Rows.RemoveAt(dgw.CurrentRow.Index);

}

}

else

{

dgw.Rows.RemoveAt(dgw.CurrentRow.Index);

MessageBox.Show("Obrisano.");

}

}

private void btnObrisiNalog\_Click(object sender, EventArgs e)

{

if (MessageBox.Show("Da li ste sigurni da želite obrisati nalog broj '"+ txtbrnalog.Text +"' ?", "Upozorenje", MessageBoxButtons.YesNo, MessageBoxIcon.Information) == DialogResult.Yes)

{

classSQL.delete("DELETE FROM ?nalozi? WHERE ?id?='" + txtbrnalog.Text + "'");

classSQL.delete("DELETE FROM ?dijelovi\_nalog? WHERE ?id\_naloga?='" + txtbrnalog.Text + "'");

EnableDisable(false);

dgw.Rows.Clear();

clear\_all();

txtbrnalog.Enabled = true;

}

}

private void btnSviNalozi\_Click(object sender, EventArgs e)

{

CheckOpened("svi\_nalozi");

this.Hide();

svi\_nalozi svi\_n = new svi\_nalozi();

svi\_n.ShowDialog();

this.Close();

}

private void pic\_ButtonEnter(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w + 3, h + 3);

}

private bool CheckOpened(string name)

{

FormCollection fc = Application.OpenForms;

foreach (Form frm in fc)

{

if (frm.Text == name)

{

return true;

}

}

return false;

}

private void pic\_ButtonLeave(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w - 3, h - 3);

}

private void pictureBox1\_Click(object sender, EventArgs e)

{

svi\_nalozi nal = new svi\_nalozi();

nal.edit\_nalog = true;

nal.ShowDialog();

if (Properties.Settings.Default.id\_nalog != 0)

{

DataTable nalog = new DataTable();

nalog = classSQL.select("SELECT \* FROM ?nalozi? WHERE ?id? ='" + Properties.Settings.Default.id\_nalog + "'", "nalog").Tables[0];

if (nalog.Rows.Count > 0)

{

btnObrisi.Enabled = true;

EnableDisable(true);

txtbrnalog.Enabled = false;

txtbrnalog.Text = nalog.Rows[0]["id"].ToString();

txtPartnerNaziv1.Text = nalog.Rows[0]["stranka"].ToString();

txtid\_stranka.Text = nalog.Rows[0]["stranka\_id"].ToString();

dtpDatum.Text = nalog.Rows[0]["datum"].ToString();

rtbxnapomena.Text = nalog.Rows[0]["napomena"].ToString();

txtkm.Text = nalog.Rows[0]["prijedeni\_kilometri"].ToString();

txtgodpro.Text = nalog.Rows[0]["godina\_proizvodnje"].ToString();

rtbxopis.Text = nalog.Rows[0]["opis\_kvara"].ToString();

dateTimePicker1.Text = nalog.Rows[0]["datum\_serv"].ToString();

lblkontrola.Text = nalog.Rows[0]["id\_baza\_automobila"].ToString();

if (nalog.Rows[0]["id\_serviser"].ToString() != "")

{

comboserviser.SelectedValue = nalog.Rows[0]["id\_serviser"].ToString();

}

else

{

comboserviser.SelectedValue = "1";

}

string sql3 = "select ?marka? from ?Baza\_automobila? where ?id? = '" + lblkontrola.Text + "'";

string sql4 = "select ?tip? from ?Baza\_automobila? where ?id? = '" + lblkontrola.Text + "'";

string marka = classSQL.select(sql3, "marka").Tables[0].Rows[0][0].ToString();

string tip = classSQL.select(sql4, "tip").Tables[0].Rows[0][0].ToString();

txtmarka.Text = marka;

txttip.Text = tip;

if (nalog.Rows[0]["id\_auta"].ToString() != "")

{

txtmarka.Enabled = false;

txttip.Enabled = false;

}

}

}

}

}

}

Forma „Serviseri“

namespace Aplikacija\_za\_zavrsni

{

public partial class Serviseri : Form

{

public Serviseri()

{

InitializeComponent();

}

private void Serviseri\_Load(object sender, EventArgs e)

{

popuni\_grid();

}

DataTable DTgrid = new DataTable();

public void popuni\_grid()

{

string sql1 = "select \* from ?Serviser?";

DTgrid = classSQL.select(sql1, "Table").Tables[0];

dataGridView1.DataSource = DTgrid;

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

DataGridViewColumn column\_id = dataGridView1.Columns[0];

column\_id.Visible = false;

DataGridViewColumn column\_ime = dataGridView1.Columns[1];

column\_ime.HeaderText = "Ime";

DataGridViewColumn column\_prezime = dataGridView1.Columns[2];

column\_prezime.HeaderText = "Prezime";

DataGridViewColumn column\_adresa = dataGridView1.Columns[3];

column\_adresa.HeaderText = "Adresa";

DataGridViewColumn column\_OIB = dataGridView1.Columns[4];

column\_OIB.HeaderText = "OIB";

}

int col1;

int row1;

private void button1\_Click(object sender, EventArgs e)

{

int row = dataGridView1.CurrentCell.RowIndex;

string id = dataGridView1.Rows[row].Cells[0].FormattedValue.ToString();

string Ime = dataGridView1.Rows[row].Cells[1].FormattedValue.ToString();

string Prezime = dataGridView1.Rows[row].Cells[2].FormattedValue.ToString();

if (MessageBox.Show("Da li ste sigurni da želite obrisati Servisera " + Ime + " " + Prezime + " ", "Upozorenje", MessageBoxButtons.YesNo, MessageBoxIcon.Information) == DialogResult.Yes)

{

string sql = "Delete from ?Serviser? where ?id\_serv? = '" + id + "'";

classSQL.delete(sql);

MessageBox.Show("Zapis obrisan !");

popuni\_grid();

}

}

private void btndodaj\_Click(object sender, EventArgs e)

{

edit\_serviser ed\_dj = new edit\_serviser();

ed\_dj.ShowDialog();

}

private void btnizmijeni\_Click(object sender, EventArgs e)

{

int col = dataGridView1.CurrentCell.ColumnIndex;

int row = dataGridView1.CurrentCell.RowIndex;

edit\_serviser eddj = new edit\_serviser();

eddj.edit = true;

eddj.id\_servis = Convert.ToInt32(dataGridView1.Rows[row].Cells[0].FormattedValue);

eddj.ShowDialog();

}

private void pic\_ButtonEnter(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w + 3, h + 3);

}

private void pic\_ButtonLeave(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w - 3, h - 3);

}

}

}

Forma „svi\_nalozi“

namespace Aplikacija\_za\_zavrsni

{

public partial class svi\_nalozi : Form

{

public int id\_nalog { get; set; }

public bool edit\_nalog { get; set; }

public svi\_nalozi()

{

InitializeComponent();

}

private void svi\_nalozi\_Load(object sender, EventArgs e)

{

popuni\_grid();

}

DataTable DTgrid = new DataTable();

private void popuni\_grid()

{

string sql1 = "select \* from ?nalozi?";

DTgrid = classSQL.select(sql1, "Table").Tables[0];

dataGridView1.DataSource = DTgrid;

for (int i = 0; i < DTgrid.Rows.Count; i++)

{

string status = dataGridView1.Rows[i].Cells["status"].FormattedValue.ToString();

if (status != "YES")

{

dataGridView1.Rows[i].DefaultCellStyle = new DataGridViewCellStyle { BackColor = Color.FromArgb(255,184,184) };

}

else

{

dataGridView1.Rows[i].DefaultCellStyle = new DataGridViewCellStyle { BackColor = Color.FromArgb(215, 255, 184) };

}

}

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

DataGridViewColumn column\_id = dataGridView1.Columns[0];

column\_id.Visible = false;

DataGridViewColumn column\_stranka = dataGridView1.Columns[1];

column\_stranka.HeaderText = "Naziv stranke";

DataGridViewColumn column\_datum = dataGridView1.Columns[2];

column\_datum.HeaderText = "Datum naloga";

DataGridViewColumn column\_napomena = dataGridView1.Columns[3];

column\_napomena.HeaderText = "Napomena";

DataGridViewColumn column\_kilometri = dataGridView1.Columns[4];

column\_kilometri.HeaderText = "Prijeđeni kilometri";

DataGridViewColumn column\_god\_pro = dataGridView1.Columns[5];

column\_god\_pro.HeaderText = "Godina proizvodnje";

DataGridViewColumn column\_opis\_kvara = dataGridView1.Columns[6];

column\_opis\_kvara.HeaderText = "Opis kvara vozila";

DataGridViewColumn column\_datum\_serv = dataGridView1.Columns[7];

column\_datum\_serv.HeaderText = "Datum servisa vozila";

DataGridViewColumn column\_id\_stranke = dataGridView1.Columns[8];

column\_id\_stranke.Visible = false;

DataGridViewColumn column\_id\_auta = dataGridView1.Columns[9];

column\_id\_auta.Visible = false;

DataGridViewColumn column\_status = dataGridView1.Columns[10];

column\_status.Visible = false;

DataGridViewColumn column\_serv = dataGridView1.Columns[11];

column\_serv.Visible = false;

DataGridViewColumn column\_id\_baza\_auta = dataGridView1.Columns[12];

column\_id\_baza\_auta.Visible = false;

}

private void btnUrediNalog\_Click(object sender, EventArgs e)

{

this.Hide();

int col = dataGridView1.CurrentCell.ColumnIndex;

int row = dataGridView1.CurrentCell.RowIndex;

string nalog\_br = dataGridView1.Rows[row].Cells["id"].FormattedValue.ToString();

novi\_nalog nn = new novi\_nalog();

nn.broj\_naloga = nalog\_br;

nn.edit = true;

nn.svi\_nalozi\_edit = true;

nn.ShowDialog();

this.Close();

}

private void button2\_Click(object sender, EventArgs e)

{

int col = dataGridView1.CurrentCell.ColumnIndex;

int row = dataGridView1.CurrentCell.RowIndex;

string id\_stranke = dataGridView1.Rows[row].Cells[8].FormattedValue.ToString();

string nalog\_br = dataGridView1.Rows[row].Cells["id"].FormattedValue.ToString();

DataTable email\_podaci = classSQL.select("select \* from stranke\_podaci where ?id? = '"+ id\_stranke +"'", "").Tables[0];

string email\_stranke = email\_podaci.Rows[0]["email"].ToString();

string stanka\_podaci = email\_podaci.Rows[0]["Ime"].ToString() + " " + email\_podaci.Rows[0]["Prezime"].ToString();

if (email\_stranke != "")

{

email\_send(email\_stranke, stanka\_podaci);

string sql = "update ?nalozi? set ?status? = 'YES' where ?id? = '" + nalog\_br + "'";

classSQL.update(sql);

MessageBox.Show("Poslano !");

}

else

{

MessageBox.Show("Nevažeći email : " + email\_stranke + " !");

}

popuni\_grid();

}

private void pic\_ButtonEnter(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w + 3, h + 3);

}

private void pic\_ButtonLeave(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w - 3, h - 3);

}

private void oznaci\_na\_netu()

{

string MyConnectionString = "Server=www.rotovilux.hr;PORT=3306;Database=rotovilu\_nix;Uid=rotovilu;Pwd=m4X1P5owq8;";

MySql.Data.MySqlClient.MySqlConnection connection = new MySql.Data.MySqlClient.MySqlConnection(MyConnectionString);

connection.Open();

int col = dataGridView1.CurrentCell.ColumnIndex;

int row = dataGridView1.CurrentCell.RowIndex;

string nalog\_br = dataGridView1.Rows[row].Cells["id"].FormattedValue.ToString();

MySql.Data.MySqlClient.MySqlCommand command1 = new MySql.Data.MySqlClient.MySqlCommand("SET SQL\_SAFE\_UPDATES=0", connection);

command1.ExecuteNonQuery();

MySql.Data.MySqlClient.MySqlCommand command = new MySql.Data.MySqlClient.MySqlCommand("Update Satus\_servisa SET status\_serv = '"+ 1 +"' " +

"WHERE id\_nalog = '" + nalog\_br + "'", connection);

command.ExecuteNonQuery();

connection.Close();

MessageBox.Show("Označeno na internetu !");

}

public void email\_send(string stranka\_email,string stranka\_podaci)

{

MailMessage mail = new MailMessage();

SmtpClient SmtpServer = new SmtpClient("smtp.gmail.com");

mail.From = new MailAddress("zavrsni.postgres@gmail.com");

mail.To.Add(stranka\_email);

mail.Subject = "Obavijest o servisu vozila";

mail.Body = "Poštovani "+ stranka\_podaci +", Vaše vozilo je servisirano te je spremno za podizanje ! ";

SmtpServer.Port = 587;

SmtpServer.Credentials = new System.Net.NetworkCredential("zavrsni.postgres@gmail.com", "Zavr12345");

SmtpServer.EnableSsl = true;

SmtpServer.Send(mail);

}

private void btnispis\_Click(object sender, EventArgs e)

{

Report.Nalog.Nalog\_frm lis = new Report.Nalog.Nalog\_frm();

int broj\_naloga = Convert.ToInt32(dataGridView1.CurrentRow.Cells[0].FormattedValue.ToString());

lis.broj\_naloga = broj\_naloga;

lis.ShowDialog();

}

private void dataGridView1\_CellClick(object sender, DataGridViewCellEventArgs e)

{

if (edit\_nalog == true)

{

try

{

int br = e.RowIndex;

Properties.Settings.Default.id\_nalog = Convert.ToInt32(dataGridView1.Rows[br].Cells[0].FormattedValue.ToString());

Properties.Settings.Default.Save();

this.Close();

}

catch (Exception)

{

}

}

}

private void btnracun\_Click(object sender, EventArgs e)

{

Report.Nalog.Racun lis = new Report.Nalog.Racun();

int broj\_naloga = Convert.ToInt32(dataGridView1.CurrentRow.Cells[0].FormattedValue.ToString());

lis.broj\_naloga = broj\_naloga;

lis.ShowDialog();

}

}

}

Forma „vozila“

namespace Aplikacija\_za\_zavrsni

{

public partial class vozila : Form

{

public int id\_vozila { get; set; }

public bool edit\_nalog { get; set; }

public vozila()

{

InitializeComponent();

}

private void vozila\_Load(object sender, EventArgs e)

{

id\_stranka.Text = id\_vozila.ToString();

string sql1 = "select \* from ?stranke\_podaci? where id= '" + id\_vozila + "'";

string ime = classSQL.select(sql1,"Ime").Tables[0].Rows[0][1].ToString();

string prezime = classSQL.select(sql1, "Ime").Tables[0].Rows[0][2].ToString();

lblime.Text = ime + " "+ prezime;

if (lblime.Text != "")

{

lblime.Visible = true;

}

popuni\_grid();

}

DataTable DTgrid = new DataTable();

public void popuni\_grid()

{

string sql1 = "SELECT ?Vozila?.id, ?Baza\_automobila?.marka, ?Baza\_automobila?.tip,?Vozila?.id\_vozila, ?Vozila?.godina\_proizv, "+

" ?Vozila?.kilometraza, ?Vozila?.datum\_zad\_serv FROM ?Vozila?, ?Baza\_automobila? WHERE "+

" ?Vozila?.id\_baza\_automobila = ?Baza\_automobila?.id AND ?Vozila?.id = '" + id\_vozila + "' ";

DTgrid = classSQL.select(sql1, "Table").Tables[0];

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

dataGridView1.DataSource = DTgrid;

DataGridViewColumn column\_id = dataGridView1.Columns[0];

column\_id.Visible = false;

DataGridViewColumn column\_marka = dataGridView1.Columns[1];

column\_marka.HeaderText = "Marka vozila";

DataGridViewColumn column\_tip = dataGridView1.Columns[2];

column\_tip.HeaderText = "Tip vozila";

DataGridViewColumn column\_id\_vozila = dataGridView1.Columns[3];

column\_id\_vozila.Visible = false;

DataGridViewColumn column\_god\_pro = dataGridView1.Columns[4];

column\_god\_pro.HeaderText = "Godina proizvodnje";

DataGridViewColumn column\_kilometraza = dataGridView1.Columns[5];

column\_kilometraza.HeaderText = "Prijeđeni kilometri vozila";

DataGridViewColumn column\_datum\_zad\_serv = dataGridView1.Columns[6];

column\_datum\_zad\_serv.HeaderText = "Datum zadnjeg servisa vozila";

}

int col1;

int row1;

private void dataGridView1\_CellClick(object sender, DataGridViewCellEventArgs e)

{

if (edit\_nalog == true)

{

try

{

int br = e.RowIndex;

Properties.Settings.Default.id\_vozilo = Convert.ToInt32(dataGridView1.Rows[br].Cells[3].FormattedValue.ToString());

Properties.Settings.Default.Save();

this.Close();

}

catch (Exception)

}

}

}

private void btnobrisi\_Click(object sender, EventArgs e)

{

int row = dataGridView1.CurrentCell.RowIndex;

string x = dataGridView1.Rows[row].Cells[3].FormattedValue.ToString();

string vozilo = dataGridView1.Rows[row].Cells[1].FormattedValue.ToString() + " " + dataGridView1.Rows[row].Cells[2].FormattedValue.ToString(); ;

if (MessageBox.Show("Jeste li sigurni da želite obrisati vozilo " + vozilo + " ?", "Spremljeno.", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

string sql = "Delete from ?Vozila? where ?id\_vozila? = '" + x + "'";

classSQL.delete(sql);

popuni\_grid();

}

}

private void btnizmijeni\_Click(object sender, EventArgs e)

{

int col = dataGridView1.CurrentCell.ColumnIndex;

int row = dataGridView1.CurrentCell.RowIndex;

Edit\_vozila eddj = new Edit\_vozila();

eddj.edit = true;

eddj.id\_stranka = dataGridView1.Rows[row].Cells[0].FormattedValue.ToString();

eddj.id\_vozila = dataGridView1.Rows[row].Cells[3].FormattedValue.ToString();

eddj.ShowDialog();

}

private void btndodaj\_Click(object sender, EventArgs e)

{

Edit\_vozila edvoz = new Edit\_vozila();

edvoz.id\_stranka = id\_stranka.Text;

edvoz.ShowDialog();

}

private void pic\_ButtonEnter(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w + 3, h + 3);

}

private void pic\_ButtonLeave(object sender, EventArgs e)

{

Button PB = ((Button)sender);

PB.BackColor = Color.Transparent;

PB.FlatStyle = FlatStyle.Flat;

PB.FlatAppearance.BorderSize = 0;

int w = PB.Size.Width;

int h = PB.Size.Height;

PB.Size = new Size(w - 3, h - 3);

}

}

Forma „Nalog\_frm“

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using Microsoft.Reporting.WinForms;

namespace Aplikacija\_za\_zavrsni.Report.Nalog

{

public partial class Nalog\_frm : Form

{

public Nalog\_frm()

{

InitializeComponent();

}

public int broj\_naloga { get; set; }

private void Nalog\_frm\_Load(object sender, EventArgs e)

{

this.reportViewer1.RefreshReport();

}

private void reportViewer1\_Load(object sender, EventArgs e)

{

podaci();

popuni\_dijelove();

ReportParameter p1 = new ReportParameter("broj\_naloga", broj\_naloga.ToString());

this.reportViewer1.LocalReport.EnableExternalImages = true;

this.reportViewer1.LocalReport.SetParameters(new ReportParameter[] { p1});

}

private void popuni\_dijelove()

{

string sql\_dijelovi = "Select \* from ?dijelovi\_nalog? where ?id\_naloga? = '"+ broj\_naloga +"'";

DataTable DT\_dijelovi = classSQL.select(sql\_dijelovi, "dijelovi").Tables[0];

decimal ukupno\_zbroj = 0;

for (int i = 0; i < DT\_dijelovi.Rows.Count; i++)

{

DataRow DTrow = dataSet1.Tables[1].NewRow();

DTrow["naziv\_dijela"] = DT\_dijelovi.Rows[i]["naziv"].ToString();

DTrow["cijena"] = DT\_dijelovi.Rows[i]["cijena"].ToString();

DTrow["kolicina"] = DT\_dijelovi.Rows[i]["kolicina"].ToString();

try

{

DTrow["ukupno"] = Convert.ToDecimal(DT\_dijelovi.Rows[i]["kolicina"].ToString()) \* Convert.ToDecimal(DT\_dijelovi.Rows[i]["cijena"].ToString());

ukupno\_zbroj += Convert.ToDecimal(DT\_dijelovi.Rows[i]["kolicina"].ToString()) \* Convert.ToDecimal(DT\_dijelovi.Rows[i]["cijena"].ToString());

}

catch

{

}

if (DT\_dijelovi.Rows.Count == i + 1)

{

DTrow["ukupno\_stavke"] = ukupno\_zbroj;

}

dataSet1.Tables[1].Rows.Add(DTrow);

}

}

private void podaci()

{

string sql = "Select ?stranka\_id?,?id\_baza\_automobila? from ?nalozi? where ?id? = '"+ broj\_naloga +"'";

DataTable DT\_nalog = classSQL.select(sql,"nalog").Tables[0];

string sql3 = "select ?marka? from ?Baza\_automobila? where ?id? = '" + DT\_nalog.Rows[0]["id\_baza\_automobila"].ToString() + "'";

string sql4 = "select ?tip? from ?Baza\_automobila? where ?id? = '" + DT\_nalog.Rows[0]["id\_baza\_automobila"].ToString() + "'";

string marka = classSQL.select(sql3, "marka").Tables[0].Rows[0][0].ToString();

string tip = classSQL.select(sql4, "tip").Tables[0].Rows[0][0].ToString();

string sql1 = "Select \* from ?stranke\_podaci? where ?id? = '"+ DT\_nalog.Rows[0]["stranka\_id"].ToString() +"'";

DataTable DT\_stranka = classSQL.select(sql1, "stranka").Tables[0];

DataRow DTrow = dataSet1.Tables[0].NewRow();

DTrow["Ime\_tvrtke"] = DT\_stranka.Rows[0]["Tvrtka"].ToString();

DTrow["Ime\_stranke"] = DT\_stranka.Rows[0]["Ime"].ToString();

DTrow["Prezime\_stranke"] = DT\_stranka.Rows[0]["Prezime"].ToString();

DTrow["Vozilo\_stranke"] = marka + " " + tip;

DTrow["OIB\_stranke"] = DT\_stranka.Rows[0]["OIB"].ToString();

dataSet1.Tables[0].Rows.Add(DTrow);

string sql5 = "Select \* from ?nalozi? where ?id? = '" + broj\_naloga + "'";

DataTable DT\_auto = classSQL.select(sql5, "nalog").Tables[0];

DataRow DTrow1 = dataSet1.Podaci\_vozilo.NewRow();

DTrow1["marka"] = marka;

DTrow1["tip"] = tip;

DTrow1["kilometri"] = DT\_auto.Rows[0]["prijedeni\_kilometri"].ToString();

DTrow1["zadnji\_serv"] = DT\_auto.Rows[0]["datum\_serv"].ToString(); ;

DTrow1["opis\_kvara"] = DT\_auto.Rows[0]["opis\_kvara"].ToString();

DTrow1["napomena"] = DT\_auto.Rows[0]["napomena"].ToString();

dataSet1.Tables[2].Rows.Add(DTrow1);

}

}

}

Forma „Racun“

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using Microsoft.Reporting.WinForms;

namespace Aplikacija\_za\_zavrsni.Report.Nalog

{

public partial class Racun : Form

{

public Racun()

{

InitializeComponent();

}

public int broj\_naloga { get; set; }

private void Racun\_Load(object sender, EventArgs e)

{

this.reportViewer1.RefreshReport();

}

private void reportViewer1\_Load(object sender, EventArgs e)

{

podaci();

popuni\_dijelove();

ReportParameter p1 = new ReportParameter("broj\_naloga", broj\_naloga.ToString());

this.reportViewer1.LocalReport.EnableExternalImages = true;

this.reportViewer1.LocalReport.SetParameters(new ReportParameter[] { p1 });

}

private void popuni\_dijelove()

{

string sql\_dijelovi = "Select \* from ?dijelovi\_nalog? where ?id\_naloga? = '" + broj\_naloga + "'";

DataTable DT\_dijelovi = classSQL.select(sql\_dijelovi, "dijelovi").Tables[0];

decimal ukupno\_zbroj = 0;

decimal sa\_pdv = 0;

decimal bez\_pdv = 0;

for (int i = 0; i < DT\_dijelovi.Rows.Count; i++)

{

DataRow DTrow = dataSet1.Tables[1].NewRow();

DTrow["naziv\_dijela"] = DT\_dijelovi.Rows[i]["naziv"].ToString();

DTrow["cijena"] = DT\_dijelovi.Rows[i]["cijena"].ToString();

DTrow["kolicina"] = DT\_dijelovi.Rows[i]["kolicina"].ToString();

try

{

sa\_pdv = Convert.ToDecimal(DT\_dijelovi.Rows[i]["kolicina"].ToString()) \* Convert.ToDecimal(DT\_dijelovi.Rows[i]["cijena"].ToString());

bez\_pdv += sa\_pdv - (sa\_pdv \* Convert.ToDecimal(0.20));

DTrow["ukupno"] = Convert.ToDecimal(DT\_dijelovi.Rows[i]["kolicina"].ToString()) \* Convert.ToDecimal(DT\_dijelovi.Rows[i]["cijena"].ToString());

ukupno\_zbroj += Convert.ToDecimal(DT\_dijelovi.Rows[i]["kolicina"].ToString()) \* Convert.ToDecimal(DT\_dijelovi.Rows[i]["cijena"].ToString());

}

catch

{

}

if (DT\_dijelovi.Rows.Count == i + 1)

{

DTrow["ukupno\_stavke"] = ukupno\_zbroj;

DTrow["string1"] = bez\_pdv;

DTrow["dec2"] = ukupno\_zbroj - bez\_pdv;

}

dataSet1.Tables[1].Rows.Add(DTrow);

}

}

private void podaci()

{

string sql = "Select ?stranka\_id?,?id\_baza\_automobila? from ?nalozi? where ?id? = '" + broj\_naloga + "'";

DataTable DT\_nalog = classSQL.select(sql, "nalog").Tables[0];

string sql3 = "select ?marka? from ?Baza\_automobila? where ?id? = '" + DT\_nalog.Rows[0]["id\_baza\_automobila"].ToString() + "'";

string sql4 = "select ?tip? from ?Baza\_automobila? where ?id? = '" + DT\_nalog.Rows[0]["id\_baza\_automobila"].ToString() + "'";

string marka = classSQL.select(sql3, "marka").Tables[0].Rows[0][0].ToString();

string tip = classSQL.select(sql4, "tip").Tables[0].Rows[0][0].ToString();

string sql1 = "Select \* from ?stranke\_podaci? where ?id? = '" + DT\_nalog.Rows[0]["stranka\_id"].ToString() + "'";

DataTable DT\_stranka = classSQL.select(sql1, "stranka").Tables[0];

DataRow DTrow = dataSet1.Tables[0].NewRow();

DTrow["Ime\_tvrtke"] = DT\_stranka.Rows[0]["Tvrtka"].ToString();

DTrow["Ime\_stranke"] = DT\_stranka.Rows[0]["Ime"].ToString();

DTrow["Prezime\_stranke"] = DT\_stranka.Rows[0]["Prezime"].ToString();

DTrow["Vozilo\_stranke"] = marka + " " + tip;

DTrow["OIB\_stranke"] = DT\_stranka.Rows[0]["OIB"].ToString();

dataSet1.Tables[0].Rows.Add(DTrow);

}

}

}