**Višja strokovna šola v Kranju**

**INFORMATIKA**

**Seminarska naloga pri predmetu Razvoj programskih aplikacij:**

**HEALTH TRACKER**

**Avtor: Gašper Črtalič**

**Mentor: Srečo Uranič**

**Kranj, Junij 2018**

Kazalo vsebine:

[1. UVOD 3](#_Toc517134039)

[2. PODATKOVNA BAZA 4](#_Toc517134040)

[3. PRIJAVNI OBRAZEC 5](#_Toc517134041)

[3.1 GUMB PRIJAVA 5](#_Toc517134042)

[4. REGISTRACIJA 7](#_Toc517134043)

[4.1 GUMB REGISTRACIJA 7](#_Toc517134044)

[4.2 VALIDNOST UPORABNIKA 8](#_Toc517134045)

[4.3 METODA ZabeleziMetaUser( ) 8](#_Toc517134046)

[4.4 METODA ZabeležiLogin( ) 9](#_Toc517134047)

[5. GLAVNI OBRAZEC 10](#_Toc517134048)

[5.1 DINAMIČEN OBRAZEC INFO 11](#_Toc517134049)

[5.2 POGOVORNO OKNO 12](#_Toc517134050)

[6. FCUKR 13](#_Toc517134051)

[6.1 GUMB SHRANI - insert 14](#_Toc517134052)

[6.2 GUMB IZBRIŠI - delete 14](#_Toc517134053)

[6.3 GUMB PONASTAVI – update 15](#_Toc517134055)

[6.4 ISKANJE SLADKORJEV 16](#_Toc517134057)

[7. FSPORT 18](#_Toc517134058)

[7.1 METODA: PridobiSliko() 18](#_Toc517134059)

[7.2 METODA: ZapisiPodatke() 19](#_Toc517134060)

[7.3 FSport\_Load 20](#_Toc517134061)

[7.4 Izvoz podatkov v Excel 20](#_Toc517134062)

[7.5 TISKANJE 22](#_Toc517134063)

[7.6 METODA: Uredi() 23](#_Toc517134064)

[8. FZnanje 24](#_Toc517134065)

[8.1 METODA: PokaziPodatke() 24](#_Toc517134066)

[9. FDodajKnjigo 25](#_Toc517134067)

[9.1 GUMB Dodaj 26](#_Toc517134068)

[9.2 KLIC METODE PreveriCeSoPoljaPrazna() 26](#_Toc517134069)

[10. ZAKLJUČEK 27](#_Toc517134070)

**KAZALO SLIK:**

[Slika 1: Health Tracker 4](#_Toc517134076)

[Slika 2: baza 5](file:///C:\Users\GEP\Desktop\N1\C%23\HealthTracker\Seminarska%20naloga%20HealthTracker.docx#_Toc517134077)

[Slika 3: FStart 6](file:///C:\Users\GEP\Desktop\N1\C%23\HealthTracker\Seminarska%20naloga%20HealthTracker.docx#_Toc517134078)

[Slika 4: FPrijava 6](#_Toc517134079)

[Slika 5: Error sporočilo 7](#_Toc517134080)

[Slika 6: FRegistracija 8](file:///C:\Users\GEP\Desktop\N1\C%23\HealthTracker\Seminarska%20naloga%20HealthTracker.docx#_Toc517134081)

[Slika 7: Pogovorno okno 13](#_Toc517134082)

[Slika 8: Referenca na FCukr 14](file:///C:\Users\GEP\Desktop\N1\C%23\HealthTracker\Seminarska%20naloga%20HealthTracker.docx#_Toc517134083)

[Slika 9: FCukr 14](file:///C:\Users\GEP\Desktop\N1\C%23\HealthTracker\Seminarska%20naloga%20HealthTracker.docx#_Toc517134084)

[Slika 10: Gumb izbriši 15](#_Toc517134085)

[Slika 11: Gumb Ponastavi 16](#_Toc517134086)

[Slika 12: Update dokončan 16](#_Toc517134087)

[Slika 13: Iskanje sladkorjev 18](#_Toc517134088)

[Slika 14: FSport 19](#_Toc517134089)

[Slika 15: Tiskanje 24](#_Toc517134090)

[Slika 16: FZnanje 25](#_Toc517134091)

# UVOD

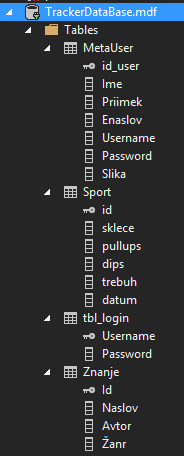
Za svojo seminarsko nalogo pri predmetu razvoj programskih aplikacij, sem se odločil, da bom izdelal program, ki bo skrbel za naše vsakodnevne dejavnosti. Program je zasnovan iz treh različnih področij: zdravje, šport, znanje. Program, izdelan v ogrodju Microsoft Visual Studio Basic 2015, skrbi da si lahko naše rezultate dnevnih dosežkov zabeležimo in tako pregledno lahko vidimo naš napredek.



Slika : Health Tracker

# PODATKOVNA BAZA

V projekt je vključena baza podatkov z imenom TrackerDataBase.mdf, kjer so shranjeni vsi podatki, ki jih program potrebuje za optimalno delovanje.



Slika : baza

Tabela MetaUser je namenjena shranjevanju podatkov ob potrditvi registracije. V njo se shranjujejo podatki o uporabniku: ime, priimek, e-naslov, uporabniško ime, geslo ter profilna slika.

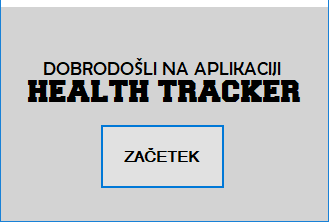
V tabelo Sport se shranjujejo uporabnikove ponovitve sklec, vzdigov, dipsov, trebusnih vaj ter datum vadbe.

Tabela Znanje ima podobno nalogo kot tabela Sport, samo da shranjuje podatke o prebranih knjigah, oz. o knjigah, ki jih trenutno beremo.

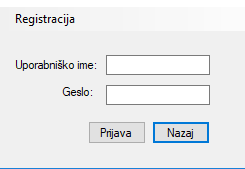
Tbl\_Login vsebuje 2 atributa: uporabnika ter geslo. Tabela je namenjena shranjevanju novih uporabnikov, in s pomočjo te tabele tudi preverimo, pravilnost podatkov, o uporabniku, ki se želi prijaviti.

# PRIJAVNI OBRAZEC

Ko se program zažene, se najprej pojavi pozdravno okno, za kateremu sledi prijava.



Slika : FStart



Slika : FPrijava

Uporabnik vnese uporabniško ime in geslo, in če se zapisa ujemata z zapisom v bazi podatkov, se uporabnik uspešno prijavi. Ob tretji neuspešni prijavi se obrazec zapre. V datoteki Program.cs sem ustvaril nov objekt tipa FStart in ga prikazal ob zagonu programa.

# GUMB PRIJAVA

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

{

FSport fs = new FSport();

SqlConnection dataConnection = new SqlConnection(ConnectionString);

dataConnection.Open(); //odpremo povezavo

string poizvedba = "Select Count(\*) from tbl\_login where Username='" + txtUser.Text + "' and Password='" + txtPass.Text + "'";

using (SqlCommand dataCommand = new SqlCommand(poizvedba, dataConnection))

{

if (System.Convert.ToInt32(dataCommand.ExecuteScalar()) == 1) //preverimo, če je bila poizvedba uspešna

{

\_username = txtUser.Text;

this.Hide();//skrijemo prijavni obrazec

FMeni meni = new FMeni(); //nov objekt za glavni obrazec

meni.Show();

}

else

{

MessageBox.Show("Napačno uporabniško ime ali geslo!");

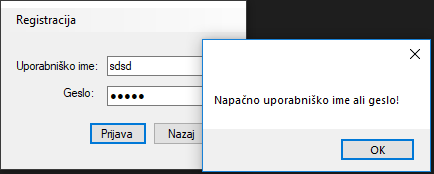
Ponastavi();

}

}

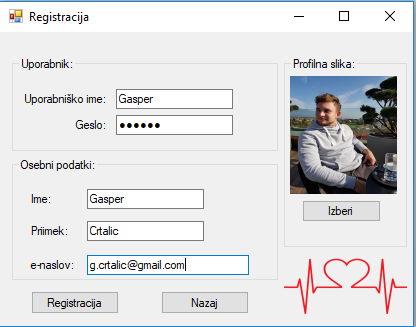
}

Ob kliku na gumb Prijava se izvede zgoraj navedena koda. Najpej vzpostavimo povezavo z podatkovno bazo. Nato pa s poizvedbo *“Select Count(\*) from tbl\_login where Uporabnik=@upIme and Geslo=@Geslo"* preverimo, če uporabnik obstaja ki ima to uporabniško ime in geslo. Poizvedba mora vrniti 1, kar pomeni da je natanko en zapis s tem uporabniškim imenom in geslom, kar pomeni, da je prijava uspela. V kolikor uporabniško ime ali geslo ni pravilno se odpre sporočilno okno z ustreznim sporočilom.



Slika : Error sporočilo

# REGISTRACIJA

Če uporabnik še nima računa, lahko le tega ustvari z izpolnjenim obrazcem Fregistracija.

Slika : FRegistracija

Podatki se ob kliku na registracijo preverijo ali ustrezajo tipu in če niso vnosna polja prazna.

Vsak uporabnik pa si lahko izbere profilno sliko, ki je prikazana kasneje v nadaljevanju. Podatke, ki jih uporabnik posreduje se zabeležijo v tabelo MetaUser.

# GUMB REGISTRACIJA

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

{

try

{

//spremenljivka ki nam pove, če je uporabnik vsa polja izpolnil

bool vsaPolja = true;

//preverimo, če je vnesel vsa polja

if (txtIme.Text == "" || txtPriimek.Text == "" || txtUser.Text == "" || txtGeslo.Text == "" || txtMail.Text == "")

{

MessageBox.Show("Izpolniti morate vsa polja!");

DialogResult = DialogResult.Retry;

vsaPolja = false;

}

else if (vsaPolja)

{

PreveriValidnostUsername();

ZabeležiMetaUser();

ZabeležiLogin();

MessageBox.Show("Hvala za vašo registracijo " + txtIme.Text + " " + txtPriimek.Text + "\nSedaj lahko nadaljujete s prijavo");

}

}catch(Exception ex) { MessageBox.Show(ex.Message); } }

# VALIDNOST UPORABNIKA

public void PreveriValidnostUsername()

{

SqlConnection povezava = new SqlConnection(\_connectionString);

povezava.Open();

bool obstaja = false;

//če uporabnik obstaja vrnemo obstaja = true

using (SqlCommand sql = new SqlCommand())

{

sql.Connection = povezava;

sql.Parameters.Add(new SqlParameter("@upIme", txtUser.Text));

sql.CommandText = "Select Count(\*) from tbl\_login where Username=@upIme";

if (System.Convert.ToInt32(sql.ExecuteScalar()) == 1)

//preverimo ali uporabnik s tem imenom že obstaja

obstaja = true;

}

//Če uporabnik že obstaja to povemo uporabniku, in dialogResult nastavimo na Retry

if (obstaja)

{

MessageBox.Show("Uporabnik s tem imenom že obstaja!");

DialogResult = DialogResult.Retry;

txtUser.Clear();

txtGeslo.Clear();

}

}

# METODA ZabeleziMetaUser( )

public void ZabeležiMetaUser()

{

using (SqlConnection connection = new SqlConnection(\_connectionString))

{

byte[] slika = null;

FileStream fs = new FileStream(naslovSlike, FileMode.Open, FileAccess.Read);

BinaryReader br = new BinaryReader(fs);

slika = br.ReadBytes((int)fs.Length); //branje binarnih podatkob iz file streama, kjer se nahaja kompleksen niz

String query = "INSERT INTO MetaUser (Ime,Priimek,Enaslov,Username,Password,Slika) VALUES (@ime,@priimek,@enaslov,@user,@pass,@slika)";

try

{

using (SqlCommand command = new SqlCommand(query, connection))

{

if (connection.State != ConnectionState.Open)

connection.Open();

command.Parameters.AddWithValue("@ime", txtIme.Text);

command.Parameters.AddWithValue("@priimek", txtPriimek.Text);

command.Parameters.AddWithValue("@enaslov", txtMail.Text);

command.Parameters.AddWithValue("@user", txtUser.Text);

command.Parameters.AddWithValue("@pass", txtGeslo.Text);

command.Parameters.Add(new SqlParameter("@slika", slika));

int result = command.ExecuteNonQuery();

connection.Close();

// Check Error

if (result < 0)

Console.WriteLine("Prišlo je do napake zabeleževanja v SQL bazo!");

}

}

catch (Exception x) { MessageBox.Show(x.Message); }

}

# METODA ZabeležiLogin( )

public void ZabeležiLogin()

{

using (SqlConnection connection = new SqlConnection(\_connectionString))

{

String query1 = "INSERT INTO tbl\_login (Username,Password) VALUES (@ime,@geslo)";

try

{

using (SqlCommand command = new SqlCommand(query1, connection))

{

command.Parameters.AddWithValue("@ime", txtUser.Text);

command.Parameters.AddWithValue("@geslo", txtGeslo.Text);

connection.Open();

int result = command.ExecuteNonQuery();

// Check Error

if (result < 0)

Console.WriteLine("Prišlo je do napake zabeleževanja v SQL bazo !");

}

}

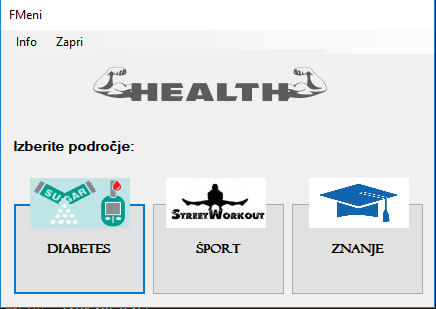
catch (Exception x) { MessageBox.Show(x.Message); }

}

}

Metodi ZabeležiMetaUser() in ZabeležiLogin() skrbita za zapisovanje v podatkovno bazo, ko uporabnik potrdi registracijo. Ker je v odzadju še slika uporabnika, sem to sliko moral spremeniti v podatkovni tip byte[], nato file streamu povedati naslov slike, ter z uporabo tehnologije BinaryReader to sliko zapisati v naš parameter slika.

# GLAVNI OBRAZEC



Ko je uporabnik logiran v svoj račun, ima na voljo izbiro treh tematik: Diabetes oz. zdravje, šport ter znanje. Ko izbere možnost za katero želi shraniti zapise, se mu odpre novo okno z ustreznimi vsebinami dodajanja in ažuriranja podatkov. Glavni obrazec ima tudi dva zavihka, in sicer Info, ki je popolnoma dinamičen obrazec, ter Zapri, ki s pomočjo pogovornega okna zapre aplikacijo.

Izbire uporabnika so predstavljene s kodo napisano spodaj.

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

{

FCukr window = new FCukr();

window.Show();

}

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

{

FSport window = new FSport();

window.Show();

}

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

{

FZnanje window = new FZnanje();

window.Show();

}

# DINAMIČEN OBRAZEC INFO

Form FInfo = new Form();

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

{

FInfo.StartPosition = FormStartPosition.CenterScreen;

FInfo.Load += FInfo\_Load;

FInfo.Text = "Info";

FInfo.Width = 600;

FInfo.Height = 600;

FInfo.ControlBox = false;

FInfo.FormBorderStyle = FormBorderStyle.None;

FInfo.ShowDialog();

}

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

{

PictureBox slika = new PictureBox();

slika.Parent = FInfo;

slika.Width = 560;

slika.Height = 225;

slika.Left = 20;

slika.Top = 20;

slika.ImageLocation = @"C:\Users\GEP\Desktop\N1\C#\HealthTracker\HealthTracker\slika1.jpg";

slika.SizeMode = PictureBoxSizeMode.StretchImage;

ClassLibraryGC.UtripajocaOznaka lblNaslov = new ClassLibraryGC.UtripajocaOznaka();

lblNaslov.Parent = FInfo;

lblNaslov.Top = 290;

lblNaslov.Left = 130;

lblNaslov.Besedilo = "HEALTH TRACKER";

lblNaslov.Height = 50;

lblNaslov.Width = 300;

FInfo.Controls.Add(lblNaslov);

Label lblText = new Label();

lblText.Parent = FInfo;

lblText.Font = new Font("Verdana", 15);

lblText.Width = 560;

lblText.Height = 80;

lblText.Top = 350;

lblText.Left = 20;

lblText.Text = "Porgram je namenjen vsem, ki želijo spremljati svojim trendom kot so zdravje (diabetes) \r\n" +

"šport ter znanje";

Label lblLink = new Label();

lblLink.Parent = FInfo;

lblLink.Font = new Font("Verdana", 15);

lblLink.Width = 400;

lblLink.Height = 30;

lblLink.Top = 445;

lblLink.Left = 20;

lblLink.Text = "Spletna stran z vsebino:";

FInfo.Controls.Add(lblLink);

LinkLabel link = new LinkLabel();

link.Parent = FInfo;

link.Font = new Font("Verdana", 10);

link.Width = 350;

link.Height = 30;

link.Top = 475;

link.Left = 20;

link.Text = "https://fitnesmotivacija.webs.com/";

link.Click += Link\_Click;

Button Zapri = new Button();

Zapri.Parent = FInfo;

Zapri.Text = "Zapri";

Zapri.Font = new Font("Verdana", 10);

Zapri.Height = 30;

Zapri.Width = 100;

Zapri.Left = 250;

Zapri.Top = 550;

Zapri.Click += Zapri\_Click;

}

Dinamično okno je zgrajeno iz različnik gradnikov kot so form, label, picturebox, ter uvoz knjižnice (label). Vsem gradnikom sem nastavil nastavitve za postavitev gradnikov na oknu in različne druge specifikacije.

# POGOVORNO OKNO

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

{

DialogResult dialogResult = MessageBox.Show("Ali res želite zapreti aplikacijo?", "Zapri?", MessageBoxButtons.YesNo);

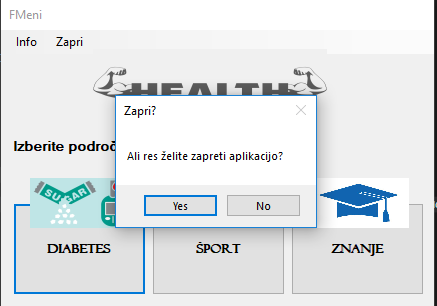
if (dialogResult == DialogResult.Yes)

{

this.Close();

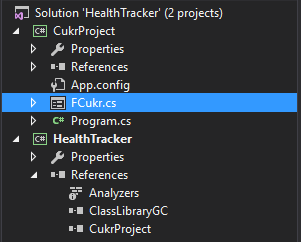
}

}



Slika : Pogovorno okno

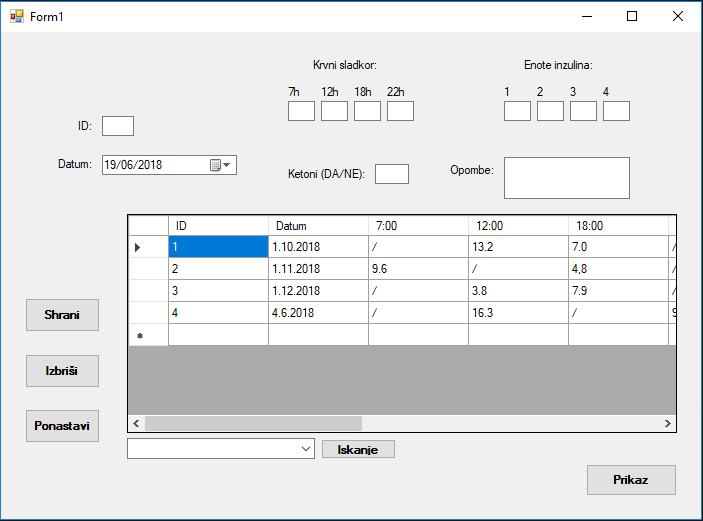
# FCUKR

Za uporabo prve možnosti oziroma tematike skrbi okno FCukr, ki prvotno pripada projektu CukrProject.

Slika : Referenca na FCukr

Za to sem moral najprej uvoziti projekt v moj solution HealthTracker, ter dodati referenco, ki kaže na CukrProject. Slika ob strani kaže zgoraj napisano.

Ob kliku na okno Diabetes se odpre novo okno z vsebino, kot je spodaj predstavljena.



Slika : FCukr

# GUMB SHRANI - insert

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

{

try

{

if (txtID.Text != "" & txt7.Text != "" & txt12.Text != "" & txt18.Text != "" & txt22.Text != "" & txt1.Text != "" & txt2.Text != "" & txt3.Text != "" & txt4.Text != "" & txtKetoni.Text != "" & txtOpombe.Text != "")

{

connectionString.Open();

cmd.CommandText = "insert into sladkorji values ('" + txtID.Text + "','" + dateTimePicker1.Value.ToString("dd/MM/yyyy") + "','" + txt7.Text + "','" + txt12.Text + "','" + txt18.Text + "','" + txt22.Text + "','" + txt1.Text + "','" + txt2.Text + "','" + txt3.Text + "','" + txt4.Text + "','" + txtKetoni.Text + "','" + txtOpombe.Text + "')";

cmd.ExecuteNonQuery();

connectionString.Close();

PokaziPodatke();

MessageBox.Show("Podatki so bili uspešno shranjeni!");

PocistiPodatke();

//ko se podatki uporabniku shranijo se počistijo txtboxi

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Prišlo je do napake!");

}

finally { ZapriPovezavo(); }

}

# GUMB IZBRIŠI - delete

# 

Slika : Gumb izbriši

# GUMB PONASTAVI – update

# 

Slika : Gumb Ponastavi

public void Ponastavi()

{

try

{

connectionString.Open();

cmd.CommandText = "UPDATE sladkorji SET Ketoni = '" + txtKetoni.Text + "', Opombe = '"+txtOpombe.Text+"', [1] = '"+txt1.Text+"', [2] = '"+txt2.Text+"', [3] = '"+txt3.Text+ "', [4] = '" + txt4.Text + "', [7:00] = '"+ txt7.Text + "', [12:00] = '" + txt12.Text + "', [18:00] = '" + txt18.Text + "', [22:00] = '" + txt22.Text + "' WHERE ID = " + int.Parse(txtID.Text);

cmd.ExecuteNonQuery();

connectionString.Close();

PokaziPodatke();

PocistiPodatke();

}

catch (Exception ex)

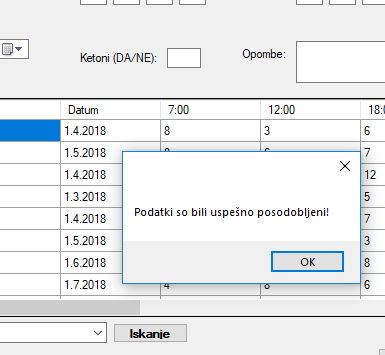
{

MessageBox.Show(ex.Message, "Prišlo je do napake!");

}

finally { ZapriPovezavo();

}



Slika : Update dokončan

# ISKANJE SLADKORJEV

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

{

try

{

if (comboBox1.SelectedItem.ToString() == "Vsi sladkorji")

{

try

{

connectionString.Open();

cmd.CommandText = "SELECT \* FROM sladkorji";

cmd.ExecuteNonQuery();

connectionString.Close();

PokaziPodatke();

PocistiPodatke();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Prišlo je do napake!");

}

}

if (comboBox1.SelectedItem.ToString() == "Sladkorji pod 8,0")

{

try

{

connectionString.Open();

cmd.CommandText = "SELECT [7:00], [12:00], [18:00], [22:00] FROM sladkorji WHERE [7:00] < 8.0, [12:00] < 8.0, [18:00] < 8.0, [22:00] < 8.0";

cmd.ExecuteNonQuery();

connectionString.Close();

PokaziPodatke();

PocistiPodatke();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Prišlo je do napake!");

}

}

if (comboBox1.SelectedItem.ToString() == "Sladkorji nad 8,0")

{

try

{

connectionString.Open();

cmd.CommandText = "SELECT [7:00], [12:00], [18:00], [22:00] FROM sladkorji WHERE [7:00] > 8.0, [12:00] > 8.0, [18:00] > 8.0, [22:00] > 8.0";

cmd.ExecuteNonQuery();

connectionString.Close();

PocistiPodatke();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Prišlo je do napake!");

}

}

if (comboBox1.SelectedItem.ToString() == "Pozitivni ketoni")

{

try

{

connectionString.Open();

cmd.CommandText = "SELECT \* FROM sladkorji WHERE Ketoni = 'DA'";

cmd.ExecuteNonQuery();

connectionString.Close();

PocistiPodatke();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Prišlo je do napake!");

}

}

}

catch(Exception ex)

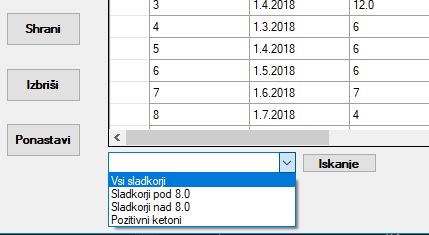
{

MessageBox.Show(ex.Message, "Prišlo je do napake!");

}

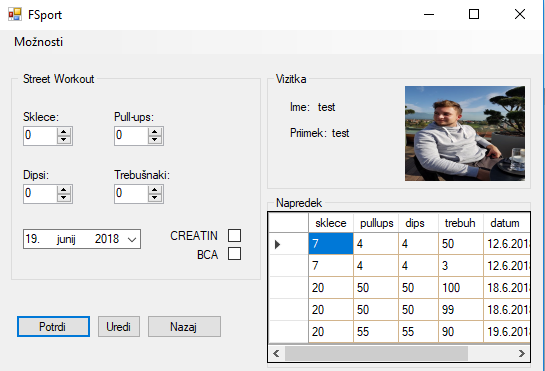
finally { ZapriPovezavo(); }

}



Slika : Iskanje sladkorjev

# FSPORT



Slika : FSport

Če v meniju izberemo možnost šport se nam odpre okno, v katerem lahko dnevno vnašamo število ponovitev za določene vaje. Poleg tega imamo možnost urejejanja podatkov s pomočjo gumba Uredi. V desnem zgornjem kotu je poseben del obrazca, v katerem je ime ter priimek logirane osebe, ter profilna slika.

V zavihku Možnosti se nam ponudita dve opcije: Natisni s predogledom ter Izvozi v Excel.

# METODA: PridobiSliko()

public void PridobiSliko(string user)

{

using (SqlConnection conn = new SqlConnection(\_connectionString))

{

conn.Open();

using (SqlCommand cmd = new SqlCommand("SELECT Slika FROM MetaUser WHERE Username = '"+user+"'", conn))

using (SqlDataReader reader = cmd.ExecuteReader())

{

if (reader.Read())

{

byte[] slika = reader["Slika"] as byte[] ?? null;

if (slika != null)

{

MemoryStream ms = new MemoryStream(slika);

pictureBox1.Image = Image.FromStream(ms);

pictureBox1.SizeMode = PictureBoxSizeMode.StretchImage;

}

else { pictureBox1.Image = null; }

}

}

}

}

Namen te funkcije je, da pridobi sliko iz baze podatkov, konkretno iz tabele MetaUser in jo prikaže v picturebox-u.

# METODA: ZapisiPodatke()

public void ZapisiPodatke()

{

try

{

using (SqlConnection connection = new SqlConnection(\_connectionString))

{

String query = "INSERT INTO Sport (sklece,pullups,dips,trebuh,datum) VALUES (@param1,@param2,@param3,@param4,@param5)";

using (SqlCommand command = new SqlCommand(query, connection))

{

command.Parameters.AddWithValue("@param1", Convert.ToInt32(numSklece.Value));

command.Parameters.AddWithValue("@param2", Convert.ToInt32(numPull.Value));

command.Parameters.AddWithValue("@param3", Convert.ToInt32(numDips.Value));

command.Parameters.AddWithValue("@param4", Convert.ToInt32(numTrebuh.Value));

command.Parameters.AddWithValue("@param5", Convert.ToDateTime(dateTimePicker1.Value));

connection.Open();

int result = command.ExecuteNonQuery();

// Check Error

if (result < 0)

Console.WriteLine("Prišlo je do napake zabeleževanja v SQL bazo!");

}

}

}

catch (Exception ex) { MessageBox.Show(ex.Message); }

}

# FSport\_Load

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

{

FPrijava fp = new FPrijava();

user = FPrijava.\_username;

lblime.Text += " " + fp.VrniTrenutnoImeUporabnika(user);

lblpriimek.Text += " " + fp.VrniTrenutniPriimekUporabnika(user);

PridobiSliko(user); //klic metode PridobiSliko()

}

# Izvoz podatkov v Excel

public static void VExcel(DataGridView dgv)

{

try

{

//Preverimo ali je sploh kaj podatkov v dataGridView-u

if (dgv.Rows.Count != 0)

{

/\*dialog za shranjevanje: lokacijo Excel datoteke bo določil

uporabnik\*/

SaveFileDialog sFD = new SaveFileDialog();

//napis na pogovornem oknu

sFD.Title = "Shranjevanje podatkov v \"EXCEL\" datoteko!";

sFD.Filter = "Excel datoteke (\*.xls)|\*.xls|Excel datoteke (\*.xlsx)|\*.xlsx"; //filter za dialog.

string imeExcelDat;

if (sFD.ShowDialog() == DialogResult.OK)

{

imeExcelDat = sFD.FileName;

//ustvarimo nov objekt tipa Excel.Application

Excel.Application xlApp = new Excel.Application();

/\*s pomočjo metode Add ustvarimo nov delovni zvezek.

Metodi Add lahko v oklepaju navedemo ime obstoječega Excel zvezka in nova

stran se bo vanj dodala\*/

Excel.Workbook xlWorkBook = xlApp.Workbooks.Add();

/\*ustvarimo novo stran: število strani določimo s pomočjo

metode get\_Item (številka 1 pomeni eno stran)\*/

Excel.Worksheet xlWorkSheet = (Excel.Worksheet)xlWorkBook.Worksheets.get\_Item(1);

//Shranimo zapise v glavi vsakega stolpca v excel.

string nizGlava = "";

int stCelice = 0; /\*zaporedna številka celice v vrstici\*/

/\*v preglednici ustvarimo toliko vrstic in celic, kolikor

jih je v dGVDrzave\*/

for (int vrstica = 0; vrstica < dgv.Columns.Count; vrstica++)

{

//Shranimo vrednost iz celice v glavi v niz.

nizGlava = dgv.Columns[vrstica].HeaderText.ToString();

//Font nastavimo na krepko.

xlWorkSheet.Cells[1, stCelice + 1].Font.Bold = true;

//določimo barvo pisave v glavah stolpcev.

xlWorkSheet.Cells[1, stCelice + 1].Font.Color = ColorTranslator.ToOle(System.Drawing.Color.Black);

xlWorkSheet.Cells[1, stCelice + 1].Interior.Color = ColorTranslator.ToOle(Color.FromArgb(204, 255, 255));

xlWorkSheet.Cells[1, stCelice + 1] = nizGlava;

stCelice++; /\*povečamo zaporedno številko celice v

tekoči vrstici\*/

nizGlava = ""; /\*resetiramo niz za shranjevanje

vrednosti celice\*/

}

/\*z dvojno zanko se sprehodimo skozi podatke in

shranimo vrednosti celic v excel\*/

for (int i = 0; i <= dgv.RowCount - 1; i++)

{

for (int j = 0; j <= dgv.ColumnCount - 1; j++)

{

/\*vsebino tekoče celice shranimo v objekt

ustreznega tipa\*/

DataGridViewCell cell = dgv[j, i];

//zapis v preglednico

xlWorkSheet.Cells[i + 2, j + 1] = cell.Value;

/\*ali pa hitreje kar takole:

xlWorkSheet.Cells[i + 2, j + 1]=dgv[j,i].Value;\*/

}

}

//Preverimo končnico excel-ove datoteke.

string koncnicaExcel = Path.GetExtension(imeExcelDat);

/\*Če je excel datoteka s končnico ".xlsx", potem jo

shranimo v tem formatu\*/

if (koncnicaExcel == ".xlsx")

{

/\*preglednico shranimo z metodo SaveAs: večino parametrov te metode ni potrebno navajati (npr. Password, ReadOnlyrecommended, CreateBackup, ...)\*/

xlWorkBook.SaveAs(imeExcelDat, Excel.XlFileFormat.xlOpenXMLWorkbook, null, null, null, null, Excel.XlSaveAsAccessMode.xlNoChange, Excel.XlSaveConflictResolution.xlUserResolution, null, null, null, null);

}

else //sicer pa bo shranjena v formatu ".xls"

xlWorkBook.SaveAs(imeExcelDat, Excel.XlFileFormat.xlWorkbookNormal, null, null, null, null, Excel.XlSaveAsAccessMode.xlNoChange, null, null, null, null, null);

xlWorkBook.Close();//zapremo delovni zvezek

xlApp.Quit();//zaključek dela z Excel-om

/\*ker objektov za delo z Excel-om ne potrebujemo več,

lahko sami poskrbimo za sproščanje ustvarjenih objektov

ReleaseObject(xlWorkSheet);//ReleaseObject(xlWorkBook);

ReleaseObject(xlApp);\*/

MessageBox.Show("Excel datoteka je ustvarjena in shranjena v: " + imeExcelDat, "Podatki o ustvarjeni \"EXCEL\" datoteki!");

}

else

{

MessageBox.Show("Pretvorba v \"Excel\" zavrnjena, ker je tabela s podatki PRAZNA!");

}

}

}

catch(Exception ex)

{

MessageBox.Show(ex.Message);

}

}

# TISKANJE

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

{

PrintDialog pD = new PrintDialog(); //dialog za izbiro tiskalnika brez predogleda, njegovih lastnosti, števila in obseg strani

PrintPreviewDialog pPD = new PrintPreviewDialog(); //dialog za izbiro tiskalnika s predogledom, njegovih lastnosti, števila in obseg strani

//pDocje objekt tipa PrintDocument, ki ga na obrazec potegnemo iz orodjarne in ga poimenujemo

System.Drawing.Printing.PrintDocument pDoc = new System.Drawing.Printing.PrintDocument();

pDoc.PrintPage += pDoc\_PrintPage;

if (pD.ShowDialog() == DialogResult.OK)

{

pDoc.PrinterSettings = pD.PrinterSettings;// privzamemo nastavitve tiskalnika za naš dokument

pPD.Document = pDoc; //pred tiskanjem določimo kako bo izgledala stran, ki jo bomo tiskali

pPD.ShowDialog();

}

}

void pDoc\_PrintPage(object sender, System.Drawing.Printing.PrintPageEventArgs e)

{

e.Graphics.DrawString(VsebinaTiskanja(), this.Font, Brushes.Blue, new PointF(30, 30));//stiskali bomo napis na obrazcu

}

public string VsebinaTiskanja()

{

string elementi = "Datum Sklece Pullups Dips Trebuh";

string data = "";

try

{

SqlConnection con = new SqlConnection(\_connectionString);

SqlCommand cmd = new SqlCommand("SELECT \* FROM Sport", con);

SqlDataReader myreader;

con.Open();

myreader = cmd.ExecuteReader();

while (myreader.Read())

{

data += myreader[5].ToString() + " " + myreader[1].ToString() + " " + myreader[2].ToString() + " " + myreader[3].ToString() + " " + myreader[4].ToString() + "\n";

}

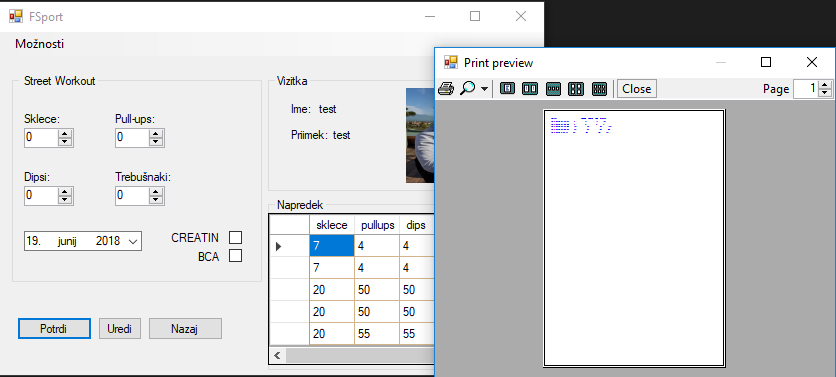
con.Close();

}

catch (Exception ex) { MessageBox.Show(ex.Message); }

return elementi + "\n" + data;

}



Slika : Tiskanje

# METODA: Uredi()

public void Uredi()

{

try

{

using (var connection1 = new SqlConnection(\_connectionString))

{

SqlDataAdapter cmd = new SqlDataAdapter();

using (var insertCommand = new SqlCommand("UPDATE Sport SET sklece = '" + numSklece.Value + "', pullups = '" + numPull.Value + "', dips = '" + numDips.Value + "', trebuh = '" + numTrebuh.Value + "' WHERE ID = " + dataGridView1.Columns["id"].Index))

{

insertCommand.Connection = connection1;

cmd.InsertCommand = insertCommand;

connection1.Open();

insertCommand.ExecuteNonQuery();

NapolniGrid();

Ponastavi();

}

}

}

catch (Exception ex)

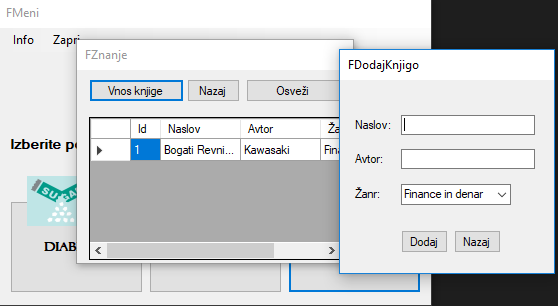
{

MessageBox.Show(ex.Message, "Prišlo je do napake!");

}

finally { ZapriPovezavo(); } }

# FZnanje



Slika : FZnanje

Okno v katerega lahko vnašamo različne knjige, katere smo že prebrali oziroma jih še bomo. Če odpremo gumb Vnos Knjige se nam odpre novo okno, v katerega lahko vstavimo podatke in dodamo knjigo v datagridview, kot tudi v podatkovno bazo.

# METODA: PokaziPodatke()

public void PokaziPodatke()

{

try

{

SqlConnection con = new SqlConnection(\_connectionString);

SqlCommand sqlCmd = new SqlCommand();

sqlCmd.Connection = con;

sqlCmd.CommandType = CommandType.Text;

sqlCmd.CommandText = "Select \* from Znanje";

SqlDataAdapter adpt = new SqlDataAdapter(sqlCmd);

DataTable dt = new DataTable();

adpt.Fill(dt);

dataGridView1.DataSource = dt;

}

catch (Exception ex) { MessageBox.Show(ex.Message); }

}

# FDodajKnjigo

**Zapis podtkov v bazo:**

public void ZapisiVBazo()

{

try

{

using (SqlConnection connection = new SqlConnection(\_connection))

{

String query = "INSERT INTO Znanje (Naslov,Avtor,Žanr) VALUES (@param1,@param2,@param3)";

using (SqlCommand command = new SqlCommand(query, connection))

{

command.Parameters.AddWithValue("@param1", txtNaslov.Text);

command.Parameters.AddWithValue("@param2", txtAvtor.Text);

command.Parameters.AddWithValue("@param3", comboBox1.GetItemText(comboBox1.SelectedItem));

connection.Open();

int result = command.ExecuteNonQuery();

// Check Error

if (result < 0)

Console.WriteLine("Prišlo je do napake zabeleževanja v SQL bazo!");

}

}

}

catch (Exception ex) { MessageBox.Show(ex.Message); }

}

# GUMB Dodaj

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

{

ZapisiVBazo();

NastaviVrednosti();

MessageBox.Show("Čestitke, kmalu boste MODER!!");

}

# KLIC METODE PreveriCeSoPoljaPrazna()

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

{

PreveriCeSoPraznaPolja();

}

//metoda, ki preveri, če sta polji prazni, in če sta vrne error provider

bool PreveriCeSoPraznaPolja()

{

string niz = "";

string niz1 = "";

try

{

niz = txtNaslov.ToString();

niz1 = txtAvtor.ToString();

}

catch

{

errorProvider1.SetError(txtNaslov, "Vnesili ste prazno polje!");

return false;

}

if (niz1.Length < 0)

{

errorProvider1.SetError(txtAvtor, "Vnesili ste prazno polje!");

return false;

}

else

{

errorProvider1.SetError(txtAvtor, "");

return true;

}

}

# ZAKLJUČEK

Izdelava programa se mi je zdela zelo zanimiva in poučna. Med izdelavo sem se naučil veliko novih stvari, utrdil sem tudi preteklo znanje, ki mi bo koristilo za naprej. Ker se sam zelo veliko ukvarjam s sladkorno boleznijo, športom ter knjigam, mi bo aplikacija dobronamerno služila in jo bom uporabljal za zasebne namene.