ШИФРА РАДНОГ ЗАДАТКА: 4**ЕИТ–А2**

НАЗИВ РАДНОГ ЗАДАТКА: **Израда десктоп апликације:**Б**иблиотека**

Пројектована је база за памћење и обраду података о расположивим ресурсима библиотеке и њеним корисницима. Структура базе је дата на слици у прилогу. Управа библиотеке се труди да стално набавља нове наслове, али и расходује старе – похабане или изгубљене наслове. Да не би дошло до претрпавања података, тражено је да се омогући преглед и брисање аутора (у ситуацији када библиотека не поседује ни један наслов тог аутора). Исказана је потреба да се у бази чувају и додатни подаци o формату у коме је књига штампана.

На основу спецификације дате у **Прилогу задатка** урадити следеће:

* Дизајнирати десктоп апликацију у циљу задовољења захтева;
* Повезати десктоп апликацију са базом података;
* Креирати упите за манипулацију подацима у бази података;
* Тестирати функционалности десктоп апликације;
* Модификовати структуру базе података у складу са захтевима корисника;
* Написати упутство корисницима о коришћењу апликације;
* ЕР дијаграм, бекап базе и апликацију запамтити на десктопу рачунара, у фолдеру **Матурски\_испит\_4ЕИТ\Име\_Презиме\4ЕИТ\_1**, где се као име и презиме уноси име ученика.

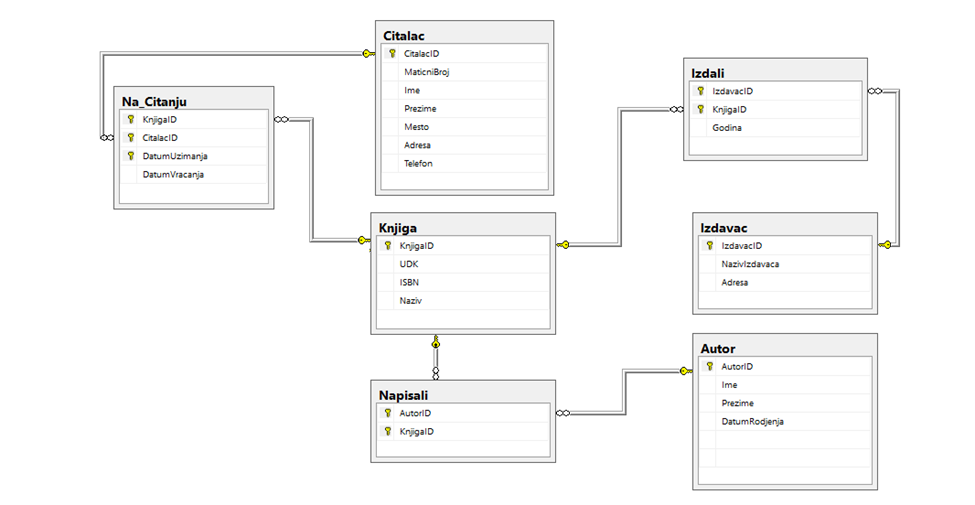
Предвиђено време за израду задатка је 120 минута.

По истеку максималног времена задатак се прекида и бодује се оно што је до тада урађено.

У оквиру времена за израду задатка ученик може да одустане од даљег рада, при чему се бодује оно што је до тада урађено.

Дијаграм почетне структуре базе, резервна копија почетног стања базе и архива икона снимљени су на радну површину рачунара (desktop), у фолдеру **Матурски\_испит\_4ЕИТ\Име\_Презиме\4ЕИТ\_А2**\ **Потребни ресурси**.

**ПРИЛОГ:**



Развојним планом апликације предвиђено је да се уради следеће:

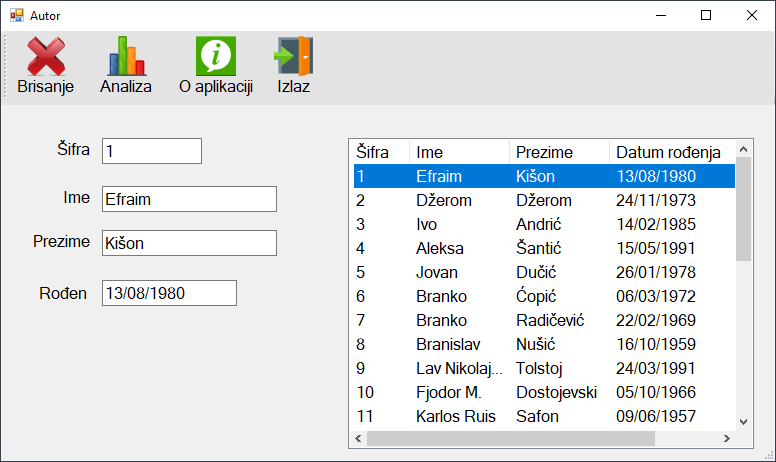
Креиратиформукоја податке о свим ауторима табеларно приказује у ***listView*** контроли.

Кликом на неки од редова, податке о аутору треба приказати у осталим пољима форме.

Кликом на дугме **Брисање**у линији са алатима, обрисати изабраног аутораиз базе, а потом освежити приказ у ***listView-у***.

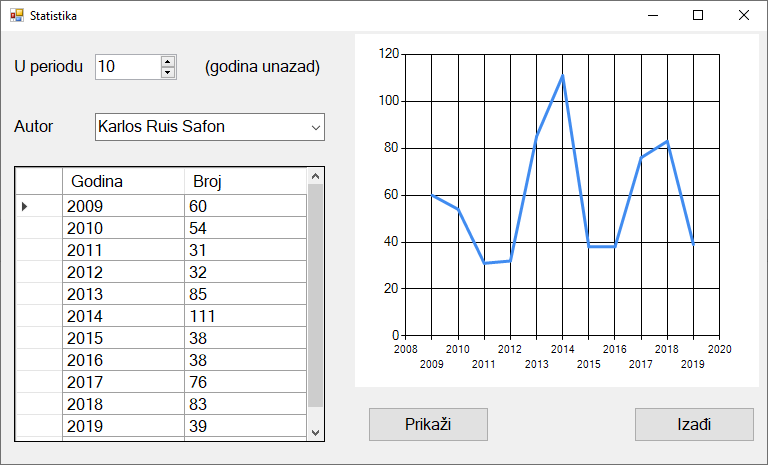
Кликом на дугме **Излаз** у линији алата прве форме, затворити целу апликацију.

Отворити кратко корисничко упутствокликом на дугме „**О апликацији**“.



Дугме **Анализа** отвара другу форму на којој је потребно приказати колико су пута дела изабраног аутора била изнајмљена током задатог временског периода. Временски период се задаје у односу на текући датум, као изабрани број година уназад. Табела и график се приказују кликом на дугме **Прикажи**.

Кликом на **Изађи** вратити корисника на почетну форму.



Осмислити проширење базе којим се за сваки примерак књиге омогућава памћење формата у коме је књига издата. Формат може да буде тврди повез, меки повез, илустовано издање...

Проширити базу тако да се за читаоца памтити датум последњег плаћања чланарине и ограничити да се чланарина не може да плаћа ретроактивно (датум плаћања није старији од текућег датума).

Креирати дијаграм и бекап стања проширене базе.

АУТОРЗАДАТКА: Милена Крстић

ЗАДАТАК ПРЕГЛЕДАЛИ:

RESENJE:

FORMA1:

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 System.Data.SqlClient;

namespace Matura\_A\_2\_Biblioteka

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void toolStripButton2\_Click(object sender, EventArgs e)

{

Form2 analiza = new Form2();

analiza.Show();

}

SqlConnection kon = new SqlConnection(@"Data Source=MILICA-PC\SQLEXPRESS;Initial Catalog=EIT-A02;Integrated Security=True");

private void ucitaj()

{

try

{

listView1.Items.Clear();

SqlCommand kom = new SqlCommand();

kom.Connection = kon;

kom.CommandText = "SELECT \*FROM AUTOR ";

kon.Open();

SqlDataReader dr = kom.ExecuteReader();

while (dr.Read())

{

ListViewItem red = new ListViewItem(dr[0].ToString());

red.SubItems.Add(dr[1].ToString());

red.SubItems.Add(dr[2].ToString());

red.SubItems.Add(((DateTime)dr[3]).ToString("dd.MM.yyyy"));

listView1.Items.Add(red);

}

kon.Close();

}

catch (Exception er)

{

MessageBox.Show(er.Message);

}

finally

{

kon.Close();

}

}

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

{

ucitaj();

}

int id = -1;

private void listView1\_SelectedIndexChanged(object sender, EventArgs e)

{

foreach (ListViewItem el in listView1.SelectedItems)

{

id = Convert.ToInt32(el.SubItems[0].Text);

textBox1.Text = el.SubItems[0].Text;

textBox2.Text = el.SubItems[1].Text;

textBox3.Text = el.SubItems[2].Text;

textBox4.Text = el.SubItems[3].Text;

}

}

private void resetuj()

{

textBox1.Clear();

textBox2.Clear();

textBox3.Clear();

textBox4.Clear();

}

private void toolStripButton1\_Click(object sender, EventArgs e)

{

try

{

if (id != 0)

{

SqlCommand kom = new SqlCommand();

kom.Connection = kon;

kom.CommandText = "DELETE FROM Autor WHERE AutorID=@i";

kom.Parameters.Clear();

kom.Parameters.AddWithValue("@i", id);

kon.Open();

kom.ExecuteNonQuery();

kon.Close();

MessageBox.Show("Uspesno ste obrisali izabranog autora");

ucitaj();

resetuj();

}

}

catch (Exception er)

{

MessageBox.Show(er.Message);

}

finally

{

kon.Close();

}

}

private void toolStripButton4\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void toolStripButton3\_Click(object sender, EventArgs e)

{

MessageBox.Show("Upustvo za korišćenje aplikacije");

}

}

}

FORMA2:

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 System.Data.SqlClient;

namespace Matura\_A\_2\_Biblioteka

{

public partial class Form2 : Form

{

public Form2()

{

InitializeComponent();

}

SqlConnection kon = new SqlConnection(@"Data Source=MILICA-PC\SQLEXPRESS;Initial Catalog=EIT-A02;Integrated Security=True");

private void Form2\_Load(object sender, EventArgs e)

{

//Ovo je za tab Analiza o iznajmljivanjima

// popunjavanje comboBox-a sa imenima i prezimenima autora

SqlCommand kom = new SqlCommand();

kom.Connection = kon;

kom.CommandText = "SELECT DISTINCT Ime, Prezime FROM Autor";

kon.Open();

SqlDataReader dr = kom.ExecuteReader();

while (dr.Read())

{

string s = dr[0].ToString() + " " + dr[1].ToString();

comboBox1.Items.Add(s);

}

kon.Close();

}

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

{

this.Close();

}

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

{

if (comboBox1.Text!="" && numericUpDown1.Value!=0 )

{

try

{

SqlCommand kom = new SqlCommand();

kom.Connection = kon;

kom.Parameters.Clear();

int g = (int)numericUpDown1.Value;

int gg = 2022 - g;

string s = comboBox1.Text;

var delovi = s.Split(' ');

// label4.Text = delovi[0].ToString();

/\* kom.CommandText = "select year(n\_c.DatumUzimanja)as godina, count(n\_c.knjigaID)as broj\_knjiga " +

" from na\_citanju n\_c " +

" where n\_c.knjigaID in (select k.knjigaID from knjiga k, napisali n " +

" where k.knjigaID=n.knjigaID and n.autorId=(select a.autorID from autor a where ime=@i and prezime=@p)) " +

" and year(n\_c.DatumUzimanja)>=2020-@g " +

" group by year(DatumUzimanja) " +

" having count(n\_c.knjigaID)>0 ";\*/

kom.CommandText = " SELECT YEAR(dbo.Na\_Citanju.DatumUzimanja) AS Godina, COUNT(dbo.Na\_Citanju.KnjigaID) AS Broj " +

" FROM dbo.Na\_Citanju INNER JOIN " +

" dbo.Knjiga ON dbo.Na\_Citanju.KnjigaID = dbo.Knjiga.KnjigaID INNER JOIN " +

" dbo.Napisali ON dbo.Knjiga.KnjigaID = dbo.Napisali.KnjigaID INNER JOIN " +

" dbo.Autor ON dbo.Napisali.AutorID = dbo.Autor.AutorID " +

" WHERE (dbo.Autor.Prezime = @p) AND (dbo.Autor.Ime =@i) AND (YEAR(dbo.Na\_Citanju.DatumUzimanja) BETWEEN 2022-@gg AND 2022) " +

" GROUP BY YEAR(dbo.Na\_Citanju.DatumUzimanja) ";

kon.Open();

kom.Parameters.AddWithValue("@i", delovi[0]);

kom.Parameters.AddWithValue("@p", delovi[1]);

kom.Parameters.AddWithValue("@gg",gg);

SqlDataAdapter da = new SqlDataAdapter(kom);

DataTable dt = new DataTable();

da.Fill(dt);

dataGridView1.DataSource = dt;

kon.Close();

chart1.Series["Prikaz"].Points.Clear();

foreach (DataRow item in dt.Rows)

{

chart1.Series["Prikaz"].Points.AddXY(item[0].ToString(), item[1]);

}

}

catch (Exception er)

{

MessageBox.Show(er.Message);

}

finally

{

kon.Close();

}

}

}

}

}