Федеральное государственное автономное образовательное учреждение высшего образования Российской Федерации

«Российский университет транспорта (РУТ (МИИТ))»

ИНСТИТУТ ЭКОНОМИКИ И ФИНАНСОВ

КАФЕДРА «ИНФОРМАЦИОННЫЕ СИСТЕМЫ ЦИФРОВОЙ ЭКОНОМИКИ»

**Курсовая работа**

на тему: «Ознакомление с работой базы данных и c#»

Выполнил: студент группы ЭБИ-211

Богушевич Антон Максимович

Проверил: Шационок П.В.

Москва 2021г

Оглавление

[Введение 3](#_Toc86884582)

[1. Обзор средств программирования 4](#_Toc86884583)

[1.1. Обзор Visual Studio 4](#_Toc86884584)

[1.2. Обзор C# 4](#_Toc86884585)

[1.3. Обзор XAML 4](#_Toc86884586)

[1.4. Обзор SQlManager 4](#_Toc86884587)

[1.5. Обзор SQLServer 5](#_Toc86884588)

[2. Формулировка задания 6](#_Toc86884589)

[3. Требования к программе 7](#_Toc86884590)

[4. UML диаграммы вариантов использования интерфейса управления пользователями для приложения 8](#_Toc86884591)

[5. Блок схема 10](#_Toc86884592)

[6. Скриншоты работы программы 19](#_Toc86884593)

[7. Листинг программы 22](#_Toc86884594)

[Заключение 41](#_Toc86884595)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 42](#_Toc86884596)

# **Введение**

Курсовая работа посвящена разработке приложения, по хранению студентов в базе данных. А также их редактирование, удаление и создание.

Данная работа позволит учебным заведениям хранить информацию о студентов в базе данных, а также вести простую статистику по ним.

Задачи по работе с данным приложением нацелены на легкую доступность к информации о студентах, а также ее редактированию.

# **Обзор средств программирования**

Проект был создан на языке программирования C#. Данный проект реализовывает все данные по заданию возможности и функционал.

# **Обзор Visual Studio**

Visual Studio — это многофункциональная программа, которая поддерживает многие аспекты разработки программного обеспечения. Интегрированная среда разработки Visual Studio — это стартовая площадка для написания, отладки и сборки кода, а также последующей публикации приложений. Помимо стандартного редактора и отладчика, которые есть в большинстве сред IDE, Visual Studio включает в себя компиляторы, средства автозавершения кода, графические конструкторы и многие другие функции для улучшения процесса разработки. Для работы будет выбран язык C#.

# **Обзор C#**

C# — объектно-ориентированный язык программирования. Разработан в 1998—2001 годах группой инженеров компании Microsoft под руководством Андерса Хейлсберга и Скотта Вильтаумота как язык разработки приложений для платформы Microsoft .NET Framework.

# **Обзор XAML**

XAML — расширяемый язык разметки для приложений, основанный на XML язык разметки для декларативного программирования приложений, разработанный Microsoft.

# **Обзор SQlManager**

 SQlManager — утилита из Microsoft SQL Server 2005 и более поздних версий для конфигурирования, управления и администрирования всех компонентов Microsoft SQL Server.

# **Обзор SQLServer**

 SQLServer - система управления реляционными базами данных, разработанная корпорацией Microsoft.

# **Формулировка задания**

Создать программу, реализующую обработку данных, введенных пользователем по следующим алгоритмам:

* 1. Запись набора введенных пользователем данных в Базу данных;
  2. Создание возможности авторизации и регистрации новых пользователей;
  3. Разделение ролей между пользователями;
  4. Редактирование информации о пользователях и студентах;
  5. Отбор данных (вводятся пользователем) с последующей сортировкой по указанным признакам (по дате рождения, платная форма обучения);
  6. Вывод всех элементов базы данных;
  7. Создание связей между таблицами базы данных;

# **Требования к программе**

* 1. Начальная форма приложения должна иметь выбор между регистрацией нового пользователя и авторизацией. А также два поля ввода данных (логин и пароль) на каждом из вариантов.
  2. Вторая форма приложения должна содержать 9 полей для ввода данных, поле для вывода текущего значения количества записей, кнопку по которой создается новый элемент таблицы базы данных student из введенных пользователем данных и отдельную кнопку для реализации каждого действия из указанных в задании.
  3. Заголовок начальной формы – ФИО студента и номер группы.
  4. Результат выполнения каждого из действий выводится на новой форме, имеющей табличное поле для вывода результата и кнопку ОК, реализующей закрытие формы. Предусмотреть блокирование начальной формы до закрытия форм с выводом результатов. На форме демонстрирующей все текущие элементы массива предусмотреть возможность редактирования и удаления записей.
  5. Для ввода полей с ограниченным набором значений (пол, основа обучения) использовать элемент comboBox, для даты рождения – dateTimePicker.
  6. Предусмотреть проверку введенных пользователем данных с генерацией сообщения об ошибке в случае необходимости. Обязательна проверка на уникальность № студенческого билета
  7. Обеспечить базовое наполнение таблицы базы данных 10 записями.

# **UML диаграммы вариантов использования интерфейса управления пользователями для приложения**

На данный момент в системе существует три типа пользователей – это гость, пользователь и администратор.

На Рисунке 1 предоставлена UML - диаграмма вариантов использования для гостя.



Рисунок 1 – Гость

На Рисунке 2 предоставлена UML- диаграмма вариантов использования для пользователя.



Рисунок 2 – Пользователь

На Рисунке 3 предоставлена UML- диаграмма вариантов использования для администратора.



Рисунок 3 – Администратор

# **Блок схема**

MainWindow.xaml.cs:







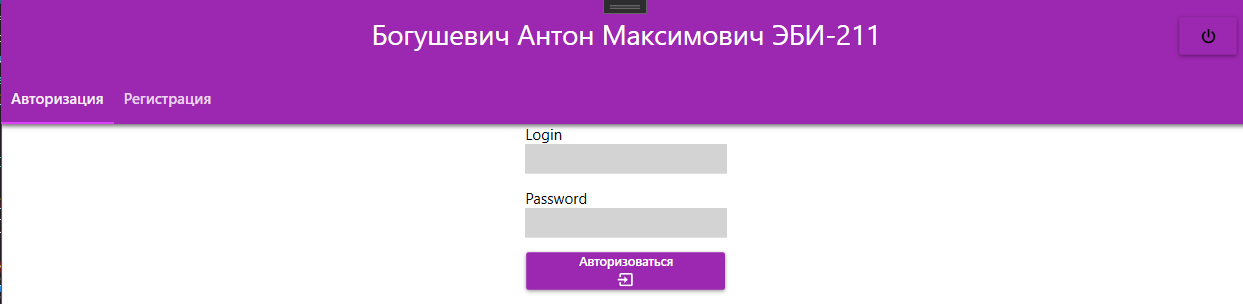
 

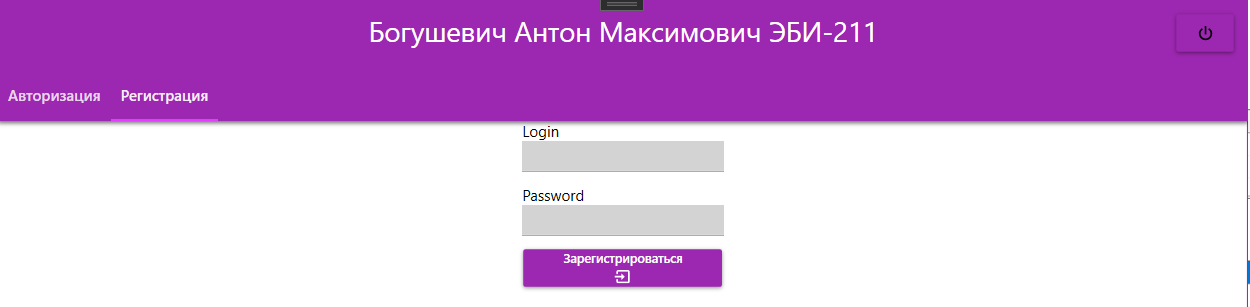


Auth.xaml.cs:

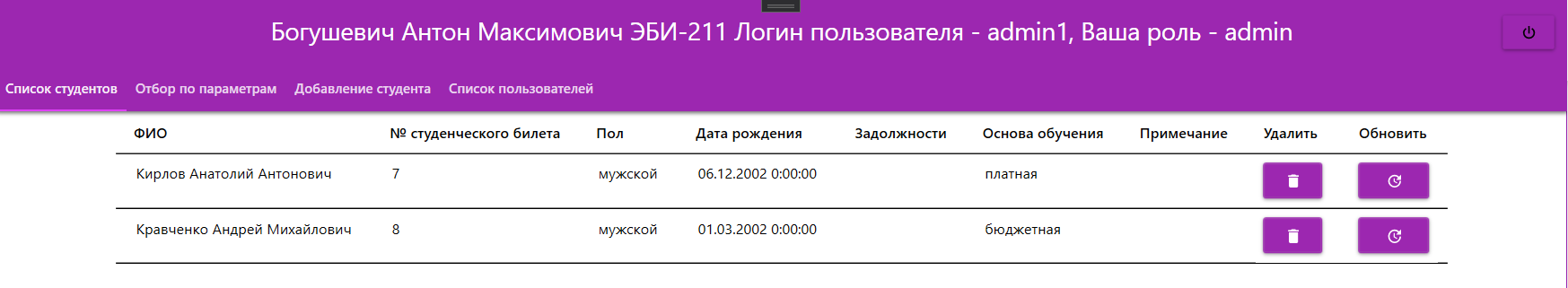
# **Скриншоты работы программы**

Начальная форма:

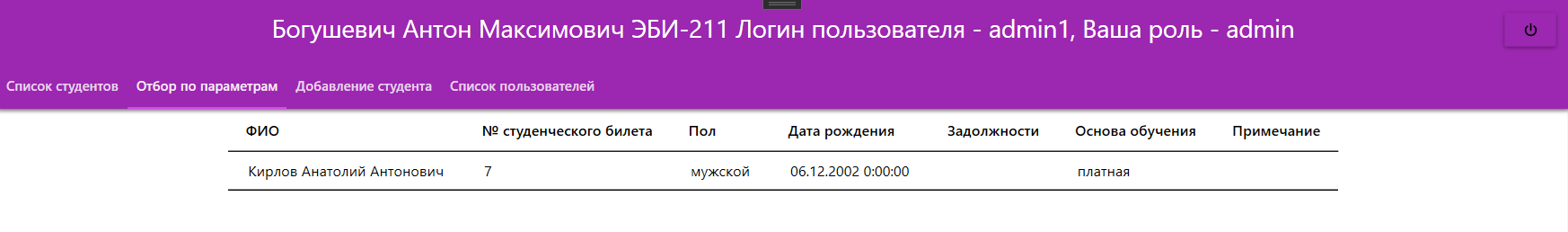




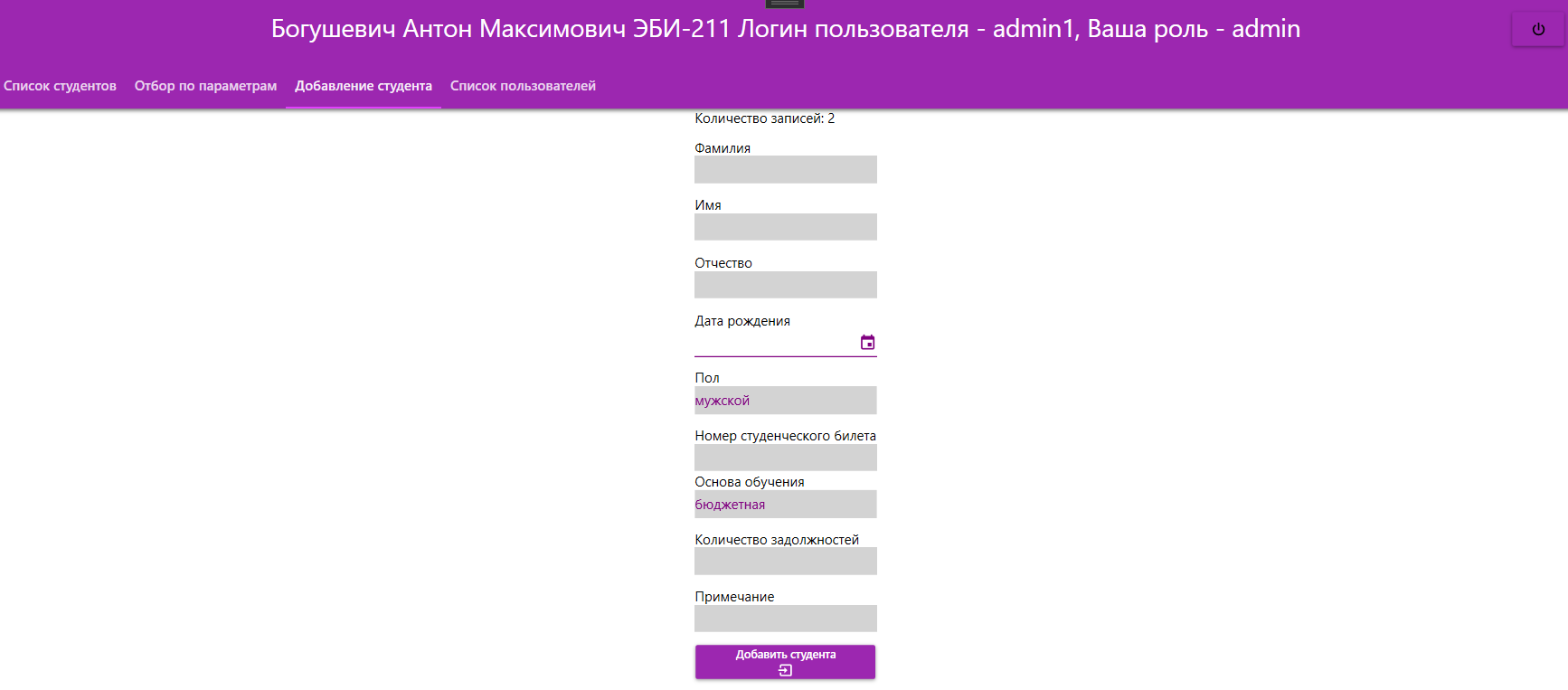
Форма отображения списка группы:



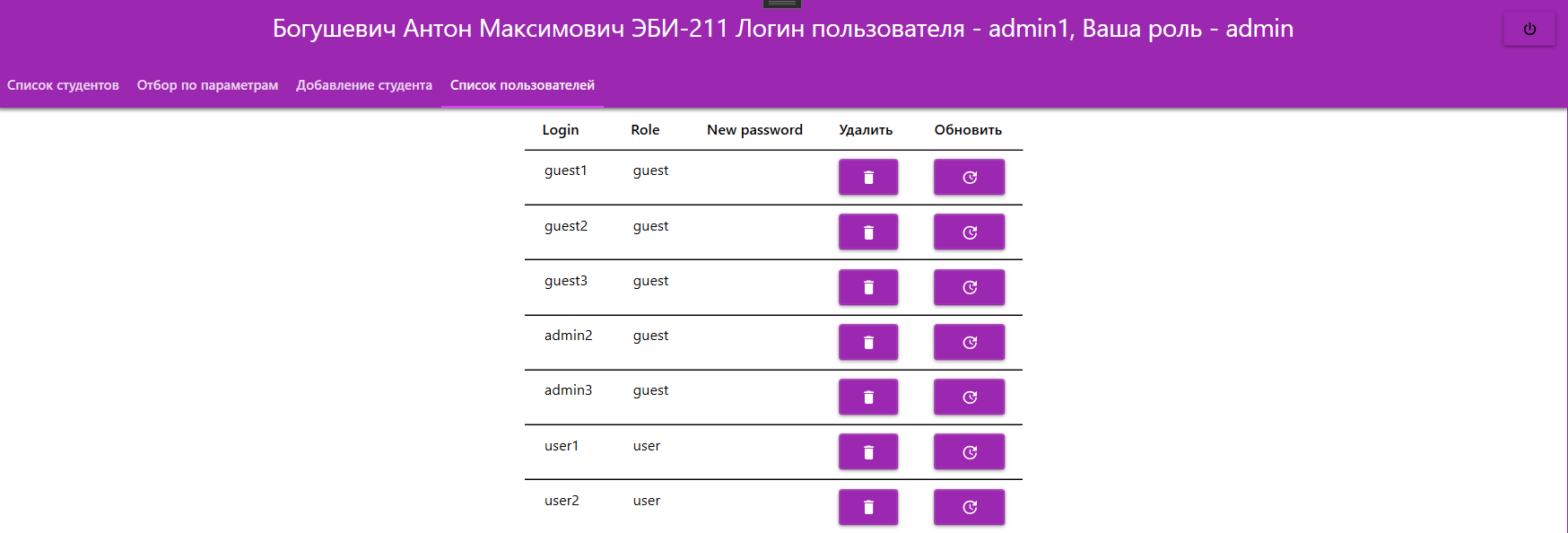
Форма обработки по параметрам:



Форма добавления студентов:



Форма отображения списка пользователей:



Скриншот базы данных:

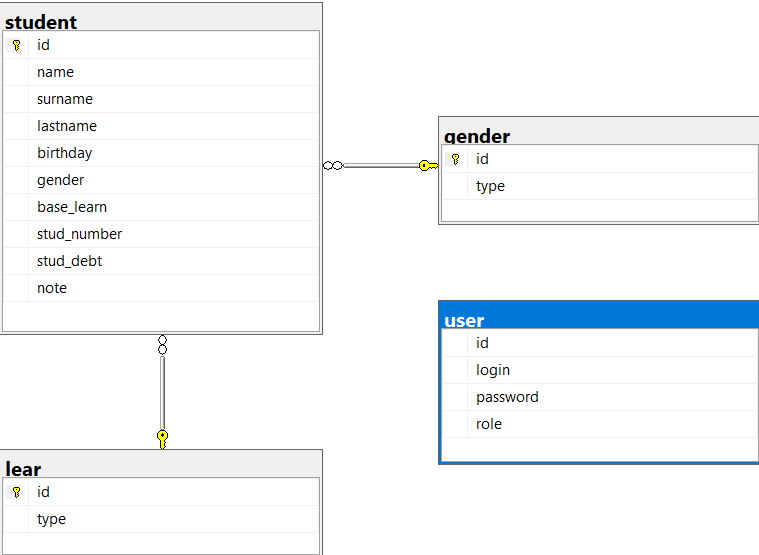


Таблица student:

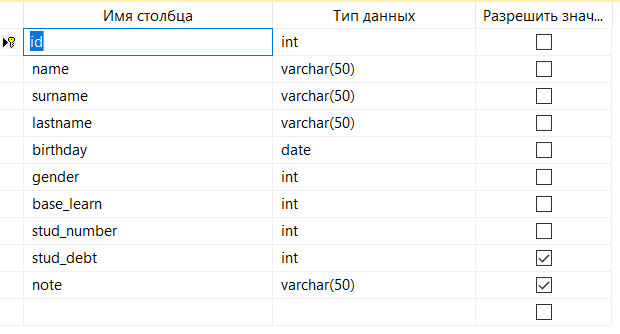


Таблица lear:

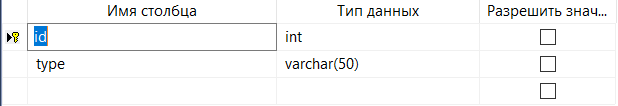


Таблица gender:

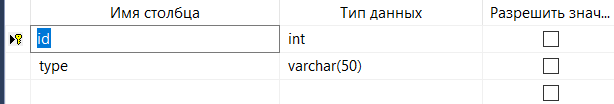
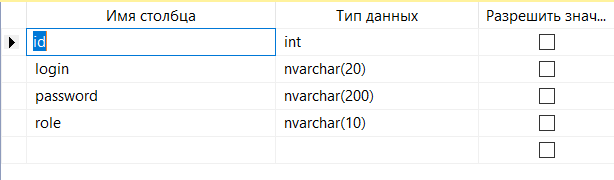


Таблица user:



# **Листинг программы**

Auth.xaml.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using System.Data;

using System.Collections.ObjectModel;

using System.Data.SqlClient;

using System.Text.RegularExpressions;

namespace coursework

{

/// <summary>

/// Логика взаимодействия для Auth.xaml

/// </summary>

public partial class Auth : Window

{

public Auth()

{

InitializeComponent();

}

void Close(object sender, RoutedEventArgs e) {

this.Close();

}

void auth\_reg(object sender, RoutedEventArgs e) {

string login = login\_box.Text;

string password = password\_box.Password.ToString();

getHash.GetHash get = new getHash.GetHash();

password = get.GetHashString(password);

sqlCon.SqlConnect connect = new sqlCon.SqlConnect();

connect.conOpen();

DataTable dt\_user1 = connect.getpassword("SELECT \* FROM[dbo].[user] WHERE login = @Login AND password = @Password",login, password);

connect.conClose();

User[] user = new User[1];

if (dt\_user1.Rows.Count != 0)

{

for (int i = 0; i < dt\_user1.Rows.Count; i++)

{

user[0] = new User()

{

login = Convert.ToString(dt\_user1.Rows[0][1]),

role = Convert.ToString(dt\_user1.Rows[0][3])

};

}

var MainWindow = new MainWindow(user); //create your new form.

MainWindow.Show(); //show the new form.

this.Close(); //only if you want to close the current form.

}

else { MessageBox.Show("Неверный логин или пароль!"); }

}

void Register(object sender, RoutedEventArgs e) {

string login = loginBoxReg.Text;

string input = passwordBoxReg.Password.ToString();

string ErrorMessage = string.Empty;

if (string.IsNullOrWhiteSpace(input))

{

ErrorMessage += "Пароль не должен быть пустым \n";

}

var hasNumber = new Regex(@"[0-9]+");

var hasUpperChar = new Regex(@"[A-Z]+");

var hasMiniMaxChars = new Regex(@".{6,20}");

var hasLowerChar = new Regex(@"[a-z]+");

var hasSymbols = new Regex(@"[!@#$%^&\*()\_+=\[{\]};:<>|./?,-]");

if (!hasLowerChar.IsMatch(input))

{

ErrorMessage += "Пароль должен содержать хотя бы одну строчную букву\n";

}

else if (!hasUpperChar.IsMatch(input))

{

ErrorMessage += "Пароль должен содержать хотя бы одну заглавную букву\n";

}

else if (!hasMiniMaxChars.IsMatch(input))

{

ErrorMessage += "Пароль должен содержать хотя бы 6 символов\n";

}

else if (!hasNumber.IsMatch(input))

{

ErrorMessage += "Пароль должен содержать хотя бы одно числовое значение\n";

}

else if (!hasSymbols.IsMatch(input))

{

ErrorMessage += "Пароль должен содержать хотя бы 1 спец. символ\n";

}

else

{

sqlCon.SqlConnect connect = new sqlCon.SqlConnect();

connect.conOpen();

DataTable dt\_user1 = connect.select\_query("SELECT \* FROM[dbo].[user] WHERE login = '" + login + "'");

connect.conClose();

if (dt\_user1.Rows.Count > 0)

{

MessageBox.Show("Такой логин уже есть в базе, введите другой");

}

else {

getHash.GetHash get = new getHash.GetHash();

input = get.GetHashString(input);

string role = "guest";

connect.conOpen();

connect.insert\_command\_user(login, input, role);

connect.conClose();

MessageBox.Show("Пользователь успешно зарегистрирован! \n Теперь вам необходимо авторизоваться.");

}

}

if (ErrorMessage != "") { MessageBox.Show(ErrorMessage); }

}}}

MainWindow.xaml.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using System.Data;

using System.Collections.ObjectModel;

namespace coursework

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public Student[] student = new Student[1];

DataTable dt\_user = new DataTable();

DataTable dt\_user\_user = new DataTable();

public string user\_login = "";

public string user\_role = "";

public MainWindow(User[] user)

{

InitializeComponent();

user\_login = user[0].login;

user\_role = user[0].role;

if (user[0].role != "admin") {

if (user[0].role == "user")

{

Tab\_Mom.Items.Remove(full\_users);

gridStudents.IsReadOnly = true;

btn\_del.Visibility = Visibility.Hidden;

btn\_res.Visibility = Visibility.Hidden;

gridStudents1.IsReadOnly = true;

}

else

{

Tab\_Mom.Items.Remove(name);

Tab\_Mom.Items.Remove(otbor\_name);

Tab\_Mom.Items.Remove(full\_users);

gridStudents.IsReadOnly = true;

btn\_del.Visibility = Visibility.Hidden;

btn\_res.Visibility = Visibility.Hidden;

}

}

gridStudents.CanUserSortColumns = false;

gridStudents1.CanUserSortColumns = false;

gridUsers.CanUserSortColumns = false;

dateTime1.DisplayDateEnd = DateTime.Now;

name\_block.Text += " Логин пользователя - " + user[0].login;

name\_block.Text += ", Ваша роль - " + user[0].role;

sqlCon.SqlConnect connect = new sqlCon.SqlConnect();

connect.conOpen();

dt\_user = connect.select\_query("SELECT \* FROM[dbo].[student] AS st LEFT JOIN[dbo].[gender] AS gr ON(st.gender = gr.id) LEFT JOIN[dbo].[lear] ler ON(st.base\_learn = ler.id)"); // получаем данные из таблицы

dt\_user\_user = connect.select\_query("SELECT \* FROM[dbo].[user]"); // получаем данные из таблицы

connect.conClose();

Person\_for\_tables[] persons = new Person\_for\_tables[dt\_user.Rows.Count];

Users\_for\_tables[] users = new Users\_for\_tables[dt\_user\_user.Rows.Count - 1];

for (int i = 0; i < dt\_user.Rows.Count; i++)

{

persons[i] = new Person\_for\_tables() { FIO = dt\_user.Rows[i][2] + " " + dt\_user.Rows[i][1] + " " + dt\_user.Rows[i][3], st\_number = Convert.ToInt32(dt\_user.Rows[i][7]), gender\_table = Convert.ToString(dt\_user.Rows[i][11]), date = Convert.ToString(dt\_user.Rows[i][4]), score = Convert.ToString(dt\_user.Rows[i][8]), osnova = Convert.ToString(dt\_user.Rows[i][13]), prim = Convert.ToString(dt\_user.Rows[i][9]) };

}

int new\_index = 0;

for (int i = 0; i < dt\_user\_user.Rows.Count; i++)

{

if (Convert.ToString(dt\_user\_user.Rows[i][1]) != user\_login)

{

users[new\_index] = new Users\_for\_tables() { Login = Convert.ToString(dt\_user\_user.Rows[i][1]), role = Convert.ToString(dt\_user\_user.Rows[i][3]), password = "" };

new\_index += 1;

}

}

gridStudents.ItemsSource = persons;

gridUsers.ItemsSource = users;

textBlock1.Text = dt\_user.Rows.Count.ToString();

}

void Load\_number2(object sender, RoutedEventArgs e) //Вывод всех

{

sqlCon.SqlConnect connect = new sqlCon.SqlConnect();

connect.conOpen();

dt\_user = connect.select\_query("SELECT \* FROM[dbo].[student] AS st LEFT JOIN[dbo].[gender] AS gr ON(st.gender = gr.id) LEFT JOIN[dbo].[lear] ler ON(st.base\_learn = ler.id)"); // получаем данные из таблицы

connect.conClose();

Person\_for\_tables[] persons = new Person\_for\_tables[dt\_user.Rows.Count];

for (int i = 0; i < dt\_user.Rows.Count; i++)

{

persons[i] = new Person\_for\_tables() { FIO = dt\_user.Rows[i][2] + " " + dt\_user.Rows[i][1] + " " + dt\_user.Rows[i][3], st\_number = Convert.ToInt32(dt\_user.Rows[i][7]), gender\_table = Convert.ToString(dt\_user.Rows[i][11]), date = Convert.ToString(dt\_user.Rows[i][4]), score = Convert.ToString(dt\_user.Rows[i][8]), osnova = Convert.ToString(dt\_user.Rows[i][13]), prim = Convert.ToString(dt\_user.Rows[i][9]) };

}

gridStudents.ItemsSource = persons;

textBlock1.Text = dt\_user.Rows.Count.ToString();

gridStudents.Height = 410;

gridStudents.MaxHeight = 410;

}

void Load\_number3(object sender, RoutedEventArgs e) //Подбор

{

sqlCon.SqlConnect connect = new sqlCon.SqlConnect();

connect.conOpen();

DataTable dt\_user1 = connect.select\_query("SELECT \* FROM[dbo].[student] AS st INNER JOIN[dbo].[gender] AS gr ON(st.gender = gr.id) INNER JOIN[dbo].[lear] ler ON(st.base\_learn = ler.id) where ler.type = 'платная' order by st.birthday DESC"); // получаем данные из таблицы // получаем данные из таблицы

connect.conClose();

Person\_for\_tables[] persons = new Person\_for\_tables[dt\_user1.Rows.Count];

for (int i = 0; i < dt\_user1.Rows.Count; i++)

{

persons[i] = new Person\_for\_tables() { FIO = dt\_user1.Rows[i][2] + " " + dt\_user1.Rows[i][1] + " " + dt\_user1.Rows[i][3], st\_number = Convert.ToInt32(dt\_user1.Rows[i][7]), gender\_table = Convert.ToString(dt\_user1.Rows[i][11]), date = Convert.ToString(dt\_user1.Rows[i][4]), score = Convert.ToString(dt\_user1.Rows[i][8]), osnova = Convert.ToString(dt\_user1.Rows[i][13]), prim = Convert.ToString(dt\_user1.Rows[i][9]) };

}

gridStudents1.ItemsSource = persons;

}

void OnClick1(object sender, RoutedEventArgs e)

{

var Auth = new Auth();

Auth.Show();

this.Close();

}

void OnClick2(object sender, RoutedEventArgs e)

{

int k;

if (textBox1.Text == "" || Int32.TryParse(textBox1.Text, out k) == true)

{

textBox1.Background = Brushes.Red; //Color.FromRgb(237, 105, 105);

textBox1.Foreground = Brushes.White;

}

else if (textBox2.Text == "" || Int32.TryParse(textBox2.Text, out k) == true)

{

textBox2.Background = Brushes.Red; //Color.FromRgb(237, 105, 105);

textBox2.Foreground = Brushes.White;

}

else if (textBox3.Text == "" || Int32.TryParse(textBox3.Text, out k) == true)

{

textBox3.Background = Brushes.Red; //Color.FromRgb(237, 105, 105);

textBox3.Foreground = Brushes.White;

}

else if (textBox4.Text == "" || Int32.TryParse(textBox4.Text, out k) == false)

{

textBox4.Background = Brushes.Red; //Color.FromRgb(237, 105, 105);

textBox4.Foreground = Brushes.White;

}

else if (textBox5.Text != "") //Для задолжностей

{

if (Int32.TryParse(textBox5.Text, out k) == false)

{

textBox5.Background = Brushes.Red; //Color.FromRgb(237, 105, 105);

textBox5.Foreground = Brushes.White;

}

}

else

{

sqlCon.SqlConnect connect = new sqlCon.SqlConnect();

connect.conOpen();

dt\_user = connect.select\_query("SELECT \* FROM[dbo].[student] WHERE stud\_number = '" + textBox4.Text + "'"); // получаем данные из таблицы

connect.conClose();

if (dt\_user.Rows.Count > 0)

{

MessageBox.Show("Номер студ. билета уже есть в базе!", "ERROR!");

}

else

{

connect = new sqlCon.SqlConnect();

connect.conOpen();

int key = gender.SelectedIndex + 1;

int key\_2 = base\_learn.SelectedIndex + 1;

student[0] = new Student

{

Name = textBox2.Text,

Surname = textBox1.Text,

LName = textBox3.Text,

DateD = dateTime1.SelectedDate.Value.Day,

DateM = dateTime1.SelectedDate.Value.Month,

DateY = dateTime1.SelectedDate.Value.Year,

Gender = Convert.ToString(key),

S\_number = Convert.ToInt32(textBox4.Text),

L\_base = key\_2,

Score = textBox5.Text,

Note = textBox6.Text

};

int check = connect.insert\_command\_student(student);

Load\_number2(gridStudents, e);

Load\_number3(gridStudents1, e);

}

}

}

public class DB\_ID

{

public int id;

}

void OnClick3(object sender, RoutedEventArgs e) // delete

{

sqlCon.SqlConnect connect = new sqlCon.SqlConnect();

connect.conOpen();

dt\_user = connect.select\_query("SELECT \* FROM[dbo].[student]"); // получаем данные из таблицы

connect.conClose();

DB\_ID[] ids = new DB\_ID[dt\_user.Rows.Count];

for (int i = 0; i < dt\_user.Rows.Count; i++)

{

ids[i] = new DB\_ID

{

id = Convert.ToInt32(dt\_user.Rows[i][0])

};

}

var index = gridStudents.SelectedIndex;

gridStudents.ClearValue(ItemsControl.ItemsSourceProperty);

connect.conOpen();

connect.delete\_command\_studentById(ids[index].id);

dt\_user = connect.select\_query("SELECT \* FROM[dbo].[student] AS st LEFT JOIN[dbo].[gender] AS gr ON(st.gender = gr.id) LEFT JOIN[dbo].[lear] ler ON(st.base\_learn = ler.id)"); // получаем данные из таблицы

connect.conClose();

Person\_for\_tables[] persons = new Person\_for\_tables[dt\_user.Rows.Count];

for (int i = 0; i < dt\_user.Rows.Count; i++)

{

persons[i] = new Person\_for\_tables() { FIO = dt\_user.Rows[i][2] + " " + dt\_user.Rows[i][1] + " " + dt\_user.Rows[i][3], st\_number = Convert.ToInt32(dt\_user.Rows[i][7]), gender\_table = Convert.ToString(dt\_user.Rows[i][11]), date = Convert.ToString(dt\_user.Rows[i][4]), score = Convert.ToString(dt\_user.Rows[i][8]), osnova = Convert.ToString(dt\_user.Rows[i][13]), prim = Convert.ToString(dt\_user.Rows[i][9]) };

}

gridStudents.ItemsSource = persons;

Load\_number3(gridStudents1, e);

connect.conOpen();

dt\_user = connect.select\_query("SELECT \* FROM[dbo].[student] AS st LEFT JOIN[dbo].[gender] AS gr ON(st.gender = gr.id) LEFT JOIN[dbo].[lear] ler ON(st.base\_learn = ler.id)"); // получаем данные из таблицы

connect.conClose();

textBlock1.Text = dt\_user.Rows.Count.ToString();

Load\_number3(gridStudents1, e);

Load\_number2(gridStudents, e);

}

void OnClick4(object sender, RoutedEventArgs e) // edit

{

var index = gridStudents.SelectedIndex;

Person\_for\_tables newPerson = new Person\_for\_tables();

Person\_for\_tables obj = gridStudents.SelectedItem as Person\_for\_tables;

string Full\_name = obj.FIO;

int st\_numb = obj.st\_number;

string gender = obj.gender\_table;

string date = obj.date;

string score = obj.score;

string osn = obj.osnova;

string prim = obj.prim;

sqlCon.SqlConnect connect = new sqlCon.SqlConnect();

connect = new sqlCon.SqlConnect();

connect.conOpen();

dt\_user = connect.select\_query("SELECT \* FROM[dbo].[student] AS st LEFT JOIN[dbo].[gender] AS gr ON(st.gender = gr.id) LEFT JOIN[dbo].[lear] ler ON(st.base\_learn = ler.id)"); // получаем данные из таблицы

DB\_ID[] ids = new DB\_ID[dt\_user.Rows.Count];

for (int i = 0; i < dt\_user.Rows.Count; i++)

{

ids[i] = new DB\_ID

{

id = Convert.ToInt32(dt\_user.Rows[i][0])

};

}

connect.delete\_command\_studentById(ids[index].id);

int key = 0;

if (gender == "мужской")

{

key = 1;

}

else {

key = 2;

}

int key\_2 = 0;

if (osn == "бюджетная")

{

key\_2 = 1;

}

else {

key\_2 = 2;

}

string[] fullName = Convert.ToString(Full\_name).Split(' ');

DateTime date1 = Convert.ToDateTime(date);

student[0] = new Student

{

Name = fullName[1],

Surname = fullName[0],

LName = fullName[2],

DateD = date1.Day,

DateM = date1.Month,

DateY = date1.Year,

Gender = Convert.ToString(key),

S\_number = st\_numb,

L\_base = key\_2,

Score = score,

Note = prim

};

int check = connect.insert\_command\_student(student);

connect.conClose();

Load\_number3(gridStudents1, e);

Load\_number2(gridStudents, e);

}

public class DB\_ID\_Users

{

public int id;

}

void delete\_users(object sender, RoutedEventArgs e) { //удаление пользователей

sqlCon.SqlConnect connect = new sqlCon.SqlConnect();

connect.conOpen();

dt\_user\_user = connect.select\_query("SELECT \* FROM[dbo].[user]"); // получаем данные из таблицы

connect.conClose();

DB\_ID\_Users[] ids\_user = new DB\_ID\_Users[dt\_user\_user.Rows.Count - 1];

int new\_index = 0;

for (int i = 0; i < dt\_user\_user.Rows.Count; i++)

{

if (Convert.ToString(dt\_user\_user.Rows[i][1]) != user\_login)

{

ids\_user[new\_index] = new DB\_ID\_Users

{

id = Convert.ToInt32(dt\_user\_user.Rows[i][0])

};

new\_index += 1;

}

}

var index = gridUsers.SelectedIndex;

gridUsers.ClearValue(ItemsControl.ItemsSourceProperty);

connect.conOpen();

connect.delete\_command\_userById(ids\_user[index].id);

dt\_user\_user = connect.select\_query("SELECT \* FROM[dbo].[user]"); // получаем данные из таблицы

connect.conClose();

Users\_for\_tables[] users = new Users\_for\_tables[dt\_user\_user.Rows.Count - 1];

new\_index = 0;

for (int i = 0; i < dt\_user\_user.Rows.Count; i++)

{

if (Convert.ToString(dt\_user\_user.Rows[i][1]) != user\_login)

{

users[new\_index] = new Users\_for\_tables() { Login = Convert.ToString(dt\_user\_user.Rows[i][1]), role = Convert.ToString(dt\_user\_user.Rows[i][3]), password = "" };

new\_index += 1;

}

}

gridUsers.ItemsSource = users;

}

void update\_users(object sender, RoutedEventArgs e) { //редактирование пользователей

var index = gridUsers.SelectedIndex;

Users\_for\_tables obj = gridUsers.SelectedItem as Users\_for\_tables;

string login = obj.Login;

string password = obj.password;

string role = obj.role;

sqlCon.SqlConnect connect = new sqlCon.SqlConnect();

connect = new sqlCon.SqlConnect();

connect.conOpen();

dt\_user\_user = connect.select\_query("SELECT \* FROM[dbo].[user]"); // получаем данные из таблицы

DB\_ID\_Users[] ids\_user = new DB\_ID\_Users[dt\_user\_user.Rows.Count - 1];

int new\_index = 0;

for (int i = 0; i < dt\_user\_user.Rows.Count; i++)

{

if (Convert.ToString(dt\_user\_user.Rows[i][1]) != user\_login)

{

ids\_user[new\_index] = new DB\_ID\_Users

{

id = Convert.ToInt32(dt\_user\_user.Rows[i][0])

};

new\_index += 1;

}

}

gridUsers.ClearValue(ItemsControl.ItemsSourceProperty);

connect.delete\_command\_userById(ids\_user[index].id);

getHash.GetHash get = new getHash.GetHash();

password = get.GetHashString(password);

int x = connect.insert\_command\_user(login, password, role);

dt\_user\_user = connect.select\_query("SELECT \* FROM[dbo].[user]"); // получаем данные из таблицы

connect.conClose();

Users\_for\_tables[] users = new Users\_for\_tables[dt\_user\_user.Rows.Count - 1];

new\_index = 0;

for (int i = 0; i < dt\_user\_user.Rows.Count; i++)

{

if (Convert.ToString(dt\_user\_user.Rows[i][1]) != user\_login)

{

users[new\_index] = new Users\_for\_tables() { Login = Convert.ToString(dt\_user\_user.Rows[i][1]), role = Convert.ToString(dt\_user\_user.Rows[i][3]), password = "" };

new\_index += 1;

}

}

gridUsers.ItemsSource = users;

MessageBox.Show("Пользователь успешно обновлен!");

}

void TextChecker1(object sender, RoutedEventArgs e)

{

textBox1.Background = Brushes.LightGray; //Color.FromRgb(237, 105, 105);

textBox1.Foreground = Brushes.Black;

}

void TextChecker2(object sender, RoutedEventArgs e)

{

textBox2.Background = Brushes.LightGray; //Color.FromRgb(237, 105, 105);

textBox2.Foreground = Brushes.Black;

}

void TextChecker3(object sender, RoutedEventArgs e)

{

textBox3.Background = Brushes.LightGray; //Color.FromRgb(237, 105, 105);

textBox3.Foreground = Brushes.Black;

}

void TextChecker4(object sender, RoutedEventArgs e)

{

textBox4.Background = Brushes.LightGray; //Color.FromRgb(237, 105, 105);

textBox4.Foreground = Brushes.Black;

}

void TextChecker5(object sender, RoutedEventArgs e)

{

textBox5.Background = Brushes.LightGray; //Color.FromRgb(237, 105, 105);

textBox5.Foreground = Brushes.Black;

}

}}

SqlConnect.cs

using System;

using System.Data;

using System.Data.SqlClient;

namespace coursework.sqlCon

{

public class SqlConnect

{

public SqlConnection Con { get; set; }//the object

private string conString { get; set; }//the string to store your connection parameters

public void conOpen()

{

conString = "Data Source=.\\NEMOOO;Initial Catalog=students;Integrated Security=True"; //the same as you post in your post

Con = new SqlConnection(conString);//

try

{

Con.Open();//try to open the connection

}

catch (Exception ex)

{

//MessageBox.Show("Ошибка подключения к бд!", "ERROR", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

public void conClose()

{

Con.Close();//close the connection

}

public DataTable select\_query(string query)

{

DataTable dataTable = new DataTable("dataBase");

SqlCommand sqlCommand = Con.CreateCommand(); // создаём команду

sqlCommand.CommandText = query;

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(sqlCommand); // создаём обработчик

sqlDataAdapter.Fill(dataTable); // возращаем таблицу с результатом

return dataTable;

}

public DataTable getpassword(string query,string login, string password)

{

DataTable dataTable = new DataTable("dataBase");

SqlCommand sqlCommand = Con.CreateCommand(); // создаём команду

sqlCommand.CommandText = query;

sqlCommand.Parameters.Add("@Login", SqlDbType.NVarChar, 20);

sqlCommand.Parameters.Add("@Password", SqlDbType.NVarChar, 200);

sqlCommand.Parameters["@Login"].Value = login;

sqlCommand.Parameters["@Password"].Value = password;

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(sqlCommand); // создаём обработчик

sqlDataAdapter.Fill(dataTable); // возращаем таблицу с результатом

return dataTable;

}

public int insert\_command\_student(coursework.Student[] student)

{

string key = student[0].Name;

string stmt = "INSERT INTO [dbo].[student] ([name],[surname],[lastname],[birthday],[gender],[base\_learn],[stud\_number],[stud\_debt],[note]) VALUES (@Name , @Surname, @Lastname, @Birthday, @Gender, @Base\_Learn, @Stud\_Number, @Stud\_Debt, @Note)";

SqlCommand cmd = new SqlCommand(stmt, Con);

cmd.Parameters.Add("@Name", SqlDbType.VarChar, 50);

cmd.Parameters.Add("@Surname", SqlDbType.VarChar, 50);

cmd.Parameters.Add("@Lastname", SqlDbType.VarChar, 50);

cmd.Parameters.Add("@Birthday", SqlDbType.Date);

cmd.Parameters.Add("@Gender", SqlDbType.Int);

cmd.Parameters.Add("@Base\_Learn", SqlDbType.Int);

cmd.Parameters.Add("@Stud\_Number", SqlDbType.Int);

cmd.Parameters.Add("@Stud\_Debt", SqlDbType.Int);

cmd.Parameters.Add("@Note", SqlDbType.VarChar, 50);

cmd.Parameters["@Name"].Value = student[0].Name;

cmd.Parameters["@Surname"].Value = student[0].Surname;

cmd.Parameters["@Lastname"].Value = student[0].LName;

string check\_date = Convert.ToString(student[0].DateD.ToString() + '.' + student[0].DateM.ToString() + '.' + student[0].DateY.ToString());

cmd.Parameters["@Birthday"].Value = check\_date;

cmd.Parameters["@Gender"].Value = student[0].Gender;

cmd.Parameters["@Base\_Learn"].Value = student[0].L\_base;

cmd.Parameters["@Stud\_Number"].Value = student[0].S\_number;

if (student[0].Score == "")

{

cmd.Parameters["@Stud\_Debt"].Value = DBNull.Value;

}

else { cmd.Parameters["@Stud\_Debt"].Value = student[0].Score; }

cmd.Parameters["@Note"].Value = student[0].Note;

cmd.ExecuteNonQuery();

return 0;

}

public int insert\_command\_user(string login, string password, string role){

string stmt = "INSERT INTO [dbo].[user] ([login],[password],[role]) VALUES (@Login , @Password, @Role)";

SqlCommand cmd = new SqlCommand(stmt, Con);

cmd.Parameters.Add("@Login", SqlDbType.NVarChar, 20);

cmd.Parameters.Add("@Password", SqlDbType.NVarChar, 200);

cmd.Parameters.Add("@Role", SqlDbType.NVarChar, 10);

cmd.Parameters["@Login"].Value = login;

cmd.Parameters["@Password"].Value = password;

cmd.Parameters["@Role"].Value = role;

cmd.ExecuteNonQuery();

return 0;}

public void delete\_command\_studentById(int id){

string stmt = "DELETE FROM [dbo].[student] WHERE id = " + id;

SqlCommand cmd = new SqlCommand(stmt, Con);

cmd.ExecuteNonQuery();}

public void delete\_command\_userById(int id){

string stmt = "DELETE FROM [dbo].[user] WHERE id = " + id;

SqlCommand cmd = new SqlCommand(stmt, Con);

cmd.ExecuteNonQuery();}

public void delete\_command\_All()

{

string stmt = "DELETE FROM [dbo].[student]";

SqlCommand cmd = new SqlCommand(stmt, Con);

cmd.ExecuteNonQuery();}}}

GetHash.cs:

using System;

using System.Collections.Generic;

using System.Text;

using System.Security.Cryptography;

namespace coursework.getHash

{

public class GetHash

{

public string GetHashString(string s)

{

//переводим строку в байт-массим

byte[] bytes = Encoding.Unicode.GetBytes(s);

//создаем объект для получения средст шифрования

MD5CryptoServiceProvider CSP =

new MD5CryptoServiceProvider();

//вычисляем хеш-представление в байтах

byte[] byteHash = CSP.ComputeHash(bytes);

string hash = string.Empty;

//формируем одну цельную строку из массива

foreach (byte b in byteHash)

hash += string.Format("{0:x2}", b);

return hash;

}

}

}

Person\_for\_tables.cs:

using System;

using System.Collections.Generic;

using System.Text;

namespace coursework

{

public class Person\_for\_tables

{

public string FIO { get; set; }

public int st\_number { get; set; }

public string gender\_table { get; set; }

public string date { get; set; }

public string score { get; set; }

public string osnova { get; set; }

public string prim { get; set; }

}

}

Student.cs:

using System;

using System.Collections.Generic;

using System.Text;

namespace coursework

{

public class Student

{

public string Name; // Имя

public string Surname; // Фамилия

public string LName; // Отчество

public int DateD; // Число ДР

public int DateM; // Месяц ДР

public int DateY; // Год ДР

public string Gender; // Пол

public int S\_number; // № студенческого билета

public int L\_base; // основа обучения

public string Score; // количество задолженностей

public string Note; // примечание

}

}

User.cs:

using System;

using System.Collections.Generic;

using System.Text;

namespace coursework

{

public class User

{

public string login;

public string role;

}

}

Users\_for\_tables.cs:

using System;

using System.Collections.Generic;

using System.Text;

namespace coursework

{

public class Users\_for\_tables

{

public string Login { get; set; }

public string role { get; set; }

public string password { get; set; }

}

}

Auth.xaml:

<Window x:Class="coursework.Auth"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:coursework"

mc:Ignorable="d"

xmlns:dragablz="clr-namespace:Dragablz;assembly=Dragablz"

xmlns:materialDesign="http://materialdesigninxaml.net/winfx/xaml/themes"

Height="300" Width="1000" ResizeMode="NoResize" WindowStartupLocation="CenterScreen" WindowStyle="None">

<Grid>

<Grid Height="60" VerticalAlignment="Top" Background="#FF9C27B0">

<TextBlock Text="Богушевич Антон Максимович ЭБИ-211" Foreground="White" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="22" FontFamily="Champagne &amp; Limousines" />

<Button Margin="0,0,10,0" HorizontalAlignment="Right" VerticalAlignment="Center" Click="Close" Foreground="Black" BorderBrush="{x:Null}" Cursor="Hand" >

<materialDesign:PackIcon Kind="Power"/>

</Button>

</Grid>

<Grid Margin="0 60 0 0" >

<dragablz:TabablzControl>

<dragablz:TabablzControl.InterTabController>

<dragablz:InterTabController/>

</dragablz:TabablzControl.InterTabController>

<TabItem Header="Авторизация">

<StackPanel VerticalAlignment="Top" HorizontalAlignment="Center" Height="450" Margin="0,0,0,10" Orientation = "Vertical">

<TextBlock Text="Login"/>

<StackPanel Orientation="Horizontal" Margin="0,0,0,10">

<TextBox x:Name="login\_box" Width="161" Background="LightGray" Cursor="IBeam" />

</StackPanel>

<TextBlock Text="Password" />

<StackPanel Orientation="Horizontal" Margin="0,0,0,10">

<PasswordBox x:Name="password\_box" PasswordChar="\*" Width="161" Background="LightGray"/>

</StackPanel>

<Button HorizontalAlignment="Center" Width="161" VerticalAlignment="Center" Foreground="White" BorderBrush="{x:Null}" Cursor="Hand" Click="auth\_reg">

<StackPanel VerticalAlignment="Center">

<TextBlock FontSize="10">Авторизоваться</TextBlock>

<materialDesign:PackIcon HorizontalAlignment="Center" VerticalAlignment="Center" Kind="Import"/>

</StackPanel>

</Button>

</StackPanel>

</TabItem>

<TabItem Header="Регистрация">

<StackPanel VerticalAlignment="Top" HorizontalAlignment="Center" Height="450" Margin="0,0,0,10" Orientation = "Vertical">

<TextBlock Text="Login"/>

<StackPanel Orientation="Horizontal" Margin="0,0,0,10">

<TextBox x:Name="loginBoxReg" Width="161" Background="LightGray" />

</StackPanel>

<TextBlock Text="Password" />

<StackPanel Orientation="Horizontal" Margin="0,0,0,10">

<PasswordBox x:Name="passwordBoxReg" PasswordChar="\*" Width="161" Background="LightGray"/>

</StackPanel>

<Button HorizontalAlignment="Center" Width="161" VerticalAlignment="Center" Foreground="White" BorderBrush="{x:Null}" Click="Register" Cursor="Hand">

<StackPanel VerticalAlignment="Center">

<TextBlock FontSize="10">Зарегистрироваться</TextBlock>

<materialDesign:PackIcon HorizontalAlignment="Center" VerticalAlignment="Center" Kind="Import"/>

</StackPanel>

</Button>

</StackPanel>

</TabItem>

</dragablz:TabablzControl>

</Grid>

</Grid>

</Window>

MainWindow.xaml:

<Window x:Class="coursework.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:coursework"

mc:Ignorable="d"

xmlns:dragablz="clr-namespace:Dragablz;assembly=Dragablz"

xmlns:materialDesign="http://materialdesigninxaml.net/winfx/xaml/themes"

Height="650" Width="1400" ResizeMode="NoResize" WindowStartupLocation="CenterScreen" WindowStyle="None">

<Grid>

<Grid Height="60" VerticalAlignment="Top" Background="#FF9C27B0">

<TextBlock x:Name ="name\_block" Text="Богушевич Антон Максимович ЭБИ-211" Foreground="White" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="22" FontFamily="Champagne &amp; Limousines" />

<Button Margin="0,0,10,0" HorizontalAlignment="Right" VerticalAlignment="Center" Foreground="Black" BorderBrush="{x:Null}" Click="OnClick1" Cursor="Hand">

<materialDesign:PackIcon Kind="Power"/>

</Button>

</Grid>

<Grid Margin="0 60 0 0" >

<dragablz:TabablzControl x:Name="Tab\_Mom">

<dragablz:TabablzControl.InterTabController>

<dragablz:InterTabController/>

</dragablz:TabablzControl.InterTabController>

<TabItem Header="Список студентов">

<StackPanel VerticalAlignment="Top" HorizontalAlignment="Center" Height="450" Margin="0,0,0,10" Orientation = "Vertical">

<DataGrid x:Name="gridStudents" ItemsSource="{Binding Person}" AutoGenerateColumns="False" Height="410" SelectionMode="Single" SelectionUnit="FullRow" CanUserReorderColumns="True" SelectedItem="{Binding Person}">

<DataGrid.Columns>

<DataGridTextColumn Header="ФИО" Binding="{Binding FIO}" />

<DataGridTextColumn Header="№ студенческого билета" Binding="{Binding st\_number}" />

<DataGridTextColumn Header="Пол" Binding="{Binding gender\_table}" />

<DataGridTextColumn Header="Дата рождения" Binding="{Binding date}" />

<DataGridTextColumn Header="Задолжности" Binding="{Binding score}" />

<DataGridTextColumn Header="Основа обучения" Binding="{Binding osnova}" />

<DataGridTextColumn Header="Примечание" Binding="{Binding prim}" />

<DataGridTemplateColumn x:Name="btn\_del" Header="Удалить" Width="85">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Click="OnClick3">

<materialDesign:PackIcon Kind="Delete" Foreground="White" HorizontalAlignment="Center"/>

</Button>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<DataGridTemplateColumn x:Name="btn\_res" Header="Обновить" Width="95">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Click="OnClick4">

<materialDesign:PackIcon Kind="Update" Foreground="White" HorizontalAlignment="Center"/>

</Button>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

</StackPanel>

</TabItem>

<TabItem x:Name="otbor\_name" Header="Отбор по параметрам">

<StackPanel x:Name="otbor" Orientation="Horizontal" VerticalAlignment="Top" HorizontalAlignment="Center" Height="450" Margin="0,0,0,10">

<DataGrid x:Name="gridStudents1" AutoGenerateColumns="False" Loaded="Load\_number3">

<DataGrid.Columns>

<DataGridTextColumn Header="ФИО" Binding="{Binding FIO}" />

<DataGridTextColumn Header="№ студенческого билета" Binding="{Binding st\_number}" />

<DataGridTextColumn Header="Пол" Binding="{Binding gender\_table}" />

<DataGridTextColumn Header="Дата рождения" Binding="{Binding date}" />

<DataGridTextColumn Header="Задолжности" Binding="{Binding score}" />

<DataGridTextColumn Header="Основа обучения" Binding="{Binding osnova}" />

<DataGridTextColumn Header="Примечание" Binding="{Binding prim}" />

</DataGrid.Columns>

</DataGrid>

</StackPanel>

</TabItem>

<TabItem x:Name="name" Header="Добавление студента">

<StackPanel x:Name="addStudent" HorizontalAlignment="Center" VerticalAlignment="Top" >

<StackPanel Orientation="Horizontal" Margin="0,0,0,10">

<TextBlock Text="Количество записей: "/>

<TextBlock x:Name="textBlock1" Text=""/>

</StackPanel>

<TextBlock Text="Фамилия"/>

<StackPanel Orientation="Horizontal" Margin="0,0,0,10">

<TextBox x:Name="textBox1" Width="161" Background="LightGray" Cursor="IBeam" TextChanged="TextChecker1"/>

</StackPanel>

<TextBlock Text="Имя"/>

<StackPanel Orientation="Horizontal" Margin="0,0,0,10">

<TextBox x:Name="textBox2" Width="161" Background="LightGray" TextChanged="TextChecker2"/>

</StackPanel>

<TextBlock Text="Отчество"/>

<StackPanel Orientation="Horizontal" Margin="0,0,0,10">

<TextBox x:Name="textBox3" Width="161" Background="LightGray" TextChanged="TextChecker3"/>

</StackPanel>

<TextBlock Text="Дата рождения"/>

<StackPanel Orientation="Horizontal" Margin="0,0,0,10">

<DatePicker x:Name="dateTime1" Width="161" Foreground="Black" BorderBrush="Purple" Background="#FFF90101" Margin="0,0,-3,0" >

</DatePicker>

</StackPanel>

<TextBlock Text="Пол"/>

<StackPanel Orientation="Horizontal" Margin="0,0,0,10">

<ComboBox x:Name="gender" Width="161" Background="LightGray" Foreground="Purple" SelectedIndex="0" OpacityMask="Black">

<ComboBoxItem Content="мужской" Opacity="100" Background="White"/>

<ComboBoxItem Content="женский" Opacity="100" Background="White"/>

</ComboBox>

</StackPanel>

<TextBlock Text="Номер студенческого билета"/>

<StackPanel Orientation="Horizontal" Margin="0 0 0 0">

<TextBox x:Name="textBox4" Width="161" Background="LightGray" TextChanged="TextChecker4"/>

</StackPanel>

<TextBlock Text="Основа обучения"/>

<StackPanel Orientation="Horizontal" Margin="0,0,0,10">

<ComboBox x:Name="base\_learn" Width="161" Background="LightGray" Foreground="Purple" SelectedIndex="0" OpacityMask="Black">

<ComboBoxItem Content="бюджетная" Opacity="100" Background="White"/>

<ComboBoxItem Content="платная" Opacity="100" Background="White"/>

</ComboBox>

</StackPanel>

<TextBlock Text="Количество задолжностей"/>

<StackPanel Orientation="Horizontal" Margin="0,0,0,10">

<TextBox x:Name="textBox5" Width="161" Background="LightGray" TextChanged="TextChecker5"/>

</StackPanel>

<TextBlock Text="Примечание"/>

<StackPanel Orientation="Horizontal" Margin="0,0,0,10">

<TextBox x:Name="textBox6" Width="161" Background="LightGray"/>

</StackPanel>

<Button HorizontalAlignment="Center" Width="161" VerticalAlignment="Center" Foreground="White" BorderBrush="{x:Null}" Click="OnClick2" Cursor="Hand">

<StackPanel VerticalAlignment="Center">

<TextBlock FontSize="10">Добавить студента</TextBlock>

<materialDesign:PackIcon HorizontalAlignment="Center" VerticalAlignment="Center" Kind="Import"/>

</StackPanel>

</Button>

</StackPanel>

</TabItem>

<TabItem x:Name="full\_users" Header="Список пользователей">

<StackPanel VerticalAlignment="Top" HorizontalAlignment="Center" Height="450" Margin="0,0,0,10" Orientation = "Vertical">

<DataGrid x:Name="gridUsers" ItemsSource="{Binding Users}" AutoGenerateColumns="False" Height="410" SelectionMode="Single" SelectionUnit="FullRow" CanUserReorderColumns="True" SelectedItem="{Binding Users}">

<DataGrid.Columns>

<DataGridTextColumn Header="Login" Binding="{Binding Login}" />

<DataGridTextColumn Header="Role" Binding="{Binding role}" />

<DataGridTextColumn Header="New password" Binding="{Binding password}" />

<DataGridTemplateColumn x:Name="btn\_delete\_user" Header="Удалить" Width="85">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Click="delete\_users">

<materialDesign:PackIcon Kind="Delete" Foreground="White" HorizontalAlignment="Center"/>

</Button>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<DataGridTemplateColumn x:Name="btn\_res\_user" Header="Обновить" Width="95">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button x:Name="update\_users" Click="update\_users">

<materialDesign:PackIcon Kind="Update" Foreground="White" HorizontalAlignment="Center"/>

</Button>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

</StackPanel>

</TabItem>

</dragablz:TabablzControl>

</Grid>

</Grid>

</Window>

# **Заключение**

В ходе курсовой работы было разработано приложение, посвященное хранению, удалению, созданию и редактирования студентов в базе данных. В данном приложении были выполнены все необходимые функции, а именно:

* Запись набора введенных пользователем данных в Базу данных;
* Создание возможности авторизации и регистрации новых пользователей;
* Разделение ролей между пользователями;
* Редактирование информации о пользователях и студентах;
* Отбор данных (вводятся пользователем) с последующей сортировкой по указанным признакам (по вариантам);
* Вывод всех элементов базы данных;
* Создание связей между таблицами базы данных;
* Вывод данных о студентах в отсортированном виде (по дате рождения, платная форма обучения).

Данное приложение готово к работе в учебных заведениях, а также других сферах, которые предусматривают хранения списка людей в базе данных.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Рихтер, Д. Программирование на платформе Microsoft .NET Framework 4.5 на языке C# [Текст] / Д. Рихтер – М.: Изд-во Символ-Плюс, 2018. – 896 с.
2. Ликнесс, Д. Приложения для Windows на C# и XAML [Текст] / Д. Ликнесс – М.: Изд-во Символ-Плюс, 2013. – 896 с.
3. Мюллер, Д. C# для чайников [Текст] / Д. Мюллер – М.: Изд-во Диалектика-Вильямс, 2019. – 608 с.
4. Программирование для Microsoft Windows [Электронный ресурс] https://www.piter.com/product/programmirovanie-dlya-microsoft-windows-8-6-e-izd-3
5. Полное руководство по языку программирования C# и платформе .NET [Электронный ресурс] https://metanit.com/sharp/tutorial/