СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 2](#_Toc120203511)

[1 РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 3](#_Toc120203512)

[1.1. Вводный инструктаж по технике безопасности во время прохождения практики 3](#_Toc120203513)

[1.2. Анализ предметной области 3](#_Toc120203514)

[1.3. Выработка требований к программному модулю. 3](#_Toc120203515)

[1.3.1. Функциональные требования 3](#_Toc120203516)

[1.3.2. Нефункциональные требования 3](#_Toc120203517)

[1.4. Разработка технического задания в соответствии с индивидуальным заданием. 3](#_Toc120203518)

[1.5. Проектирование программного обеспечения 3](#_Toc120203519)

[1.6. Проектирование логики программного модуля 3](#_Toc120203520)

[2 СРЕДСТВА РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 4](#_Toc120203521)

[2.1 Программная реализация приложения. 4](#_Toc120203522)

[2.1.1 Проектирование интерфейса приложения 4](#_Toc120203523)

[2.1.2 Программирование и интерпретация результатов 4](#_Toc120203524)

[2.1.3 Реализация алгоритма интеграции данных. 4](#_Toc120203525)

[2.2 Отладка приложения 4](#_Toc120203526)

[2.3 Тестирование приложения 4](#_Toc120203527)

[2.3.1 Выбор и обоснование методики проведения тестирования 4](#_Toc120203528)

[2.3.2 Проведение комплексного тестирования приложения 4](#_Toc120203529)

[2.3.3 Документирование результатов тестирования 4](#_Toc120203530)

[3 МОДЕЛИРОВАНИЕ В ПРОГРАММНЫХ СИСТЕМАХ 5](#_Toc120203531)

[3.1 Нахождение кратчайших путей на графе. 5](#_Toc120203532)

[3.2 Решение задачи об оптимальных ресурсах 5](#_Toc120203533)

[Приложение 1 8](#_Toc120203534)

[**1.** **ОБЩИЕ ПОЛОЖЕНИЕ** 9](#_Toc120203535)

[1.1 Полное наименование разработки 9](#_Toc120203536)

[1.2 Назначение и область применения 9](#_Toc120203537)

# ВВЕДЕНИЕ

В рамках учебной практики по модулю ПМ 02, осваиваются практические навыки общих и профессиональных компетенций. В результате прохождения практической подготовки по модулю ПМ 02 «Осуществление интеграции программных модулей», предоставляются работы в соответствии с полученным индивидуальным заданием

Целью учебной практики является закрепление общих и профессиональных компетенций, определяющих готовность обучающихся к виду профессиональной деятельности Информационные системы и программирование.

Для выполнения поставленных целей поставлены следующие задачи:

* Разработать требования к программному модулю.
* Разработать техническое задание
* Спроектировать диаграммы UML
* Разработать графический интерфейс приложения
* Разработать логику приложения
* Провести тестирование приложения

# РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

В данном разделе

## Вводный инструктаж по технике безопасности во время прохождения практики

Общее положения:

* К работе в компьютерном классе допускаются лица, ознакомленные с данной инструкцией по технике безопасности и правилам поведения.
* Работа обучающихся в компьютерном классе разрешается только в присутствии преподавателя (лаборанта).
* Во время занятий посторонние лица могут находиться в классе только с

разрешения преподавателя.

* Во время перемен между уроками проводится обязательное проветривание компьютерного кабинета с обязательным выходом обучающихся из класса.
* Помните, что каждый обучающихся в ответе за состояние своего рабочего места и сохранность размещенного на нем оборудования.

Перед началом работы необходимо:

* Убедиться в отсутствии видимых повреждений на рабочем месте;
* Разместить на столе тетради, учебные пособия так, что бы они не мешали работе на компьютере;
* Принять правильною рабочую позу.
* Посмотреть на индикатор монитора и системного блока и определить, включён или выключен компьютер. Переместите мышь, если компьютер

находится в энергосберегающем состоянии или включить монитор, если он был выключен.

При работе в компьютерном классе категорически запрещается:

* Находиться в классе в верхней одежде;
* Класть одежду и сумки на столы;
* Находиться в классе с напитками и едой;
* Располагаться сбоку или сзади от включенного монитора;
* Присоединять или отсоединять кабели, трогать разъемы, провода и розетки;
* Передвигать компьютеры и мониторы;
* Открывать системный блок;
* Включать и выключать компьютеры самостоятельно.
* Пытаться самостоятельно устранять неисправности в работе аппаратуры;
* Перекрывать вентиляционные отверстия на системном блоке и мониторе;
* Ударять по клавиатуре, нажимать бесцельно на клавиши;
* Класть книги, тетради идругие вещи на клавиатуру, монитор и системный блок;
* Удалять и перемещать чужие файлы;
* Приносить и запускать компьютерные игры.

# Находясь в компьютерном классе, обучающиеся обязаны:

* Соблюдать тишину и порядок;
* Выполнять требования преподавателя и лаборанта;
* Находясь в сети работать только под своим именем и паролем;
* Соблюдать режим работы (согласно п. 9.4.2. Санитарных правил и норм);

## Анализ предметной области

Образование является неотъемлемой частью нашего общества поэтому важно уделять особое внимание всестороннему развитию обучающихся. Образовательный процесс должен включать в себя гуманитарные и точные науки, но не стоит забывать о физической активности обучающихся в виде уроков/пар предмета физическая культура на которой обучающийся занимаются, физическими активностями для поддержания их здоровья.

В организации образовательного процесса необходимо также позаботится об удобстве педагогов образовательного учреждения, поэтому необходимо внедрять в организацию системы оптимизации документооборота. Одна из таких систем является мой программный модуль **«Система ведения электронного журнала и расчет среднего балла обучающихся»**

Проанализировав предметную область был найден следующий аналог моего программного модуля:

* Дневник.ру.

Дневник.ру является многофункциональной системой проанализировав данную разработку были найдены следующие недоработки в виде большой избыточности функционала, а так как заказчику был не нужен столь большой функционал принято решение создать новый программный модуль **«Система ведения электронного журнала и расчет среднего балла обучающихся»**

## Выработка требований к программному модулю.

### Функциональные требования

Необходимо обеспечить поиск нужной информации по одному или нескольким критериям.

Приложение должно иметь минимум одну печатную форму (в зависимости от условий задачи) для вывода информации в виде законченного отформатированного документа.

Приложение должно сохранять следующую информацию о студентах первого курса:

* фамилия,
* группа,
* оценка1 - экзамен по высшей математике,
* оценка2 - по физике,
* оценка3 - по истории России.

Программа должна выдавать средний балл по каждой оценке

### Нефункциональные требования

Для корректной работы программного модуля требуется установленный Net Framework Runtime не ниже версии 8

## Разработка технического задания в соответствии с индивидуальным заданием

Для уточнения более точных требований заказчика и юридической защиты как исполнителя, так и заказчика, было разработано техническое задание в соответствии с ГОСТ-19.201-78. Расположен данный документ в приложении А.

## Проектирование программного обеспечения

Спроектированы 4 диаграммы для программного модуля

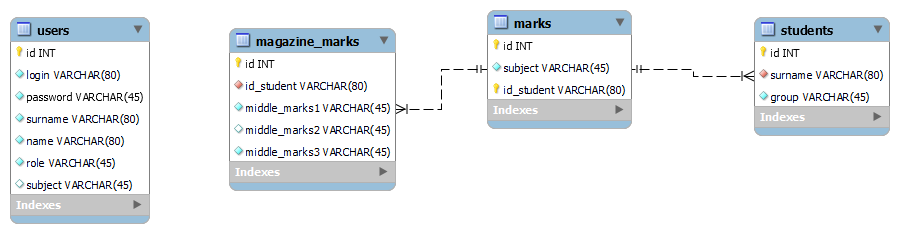


Рисунок 1 Диаграмма классов

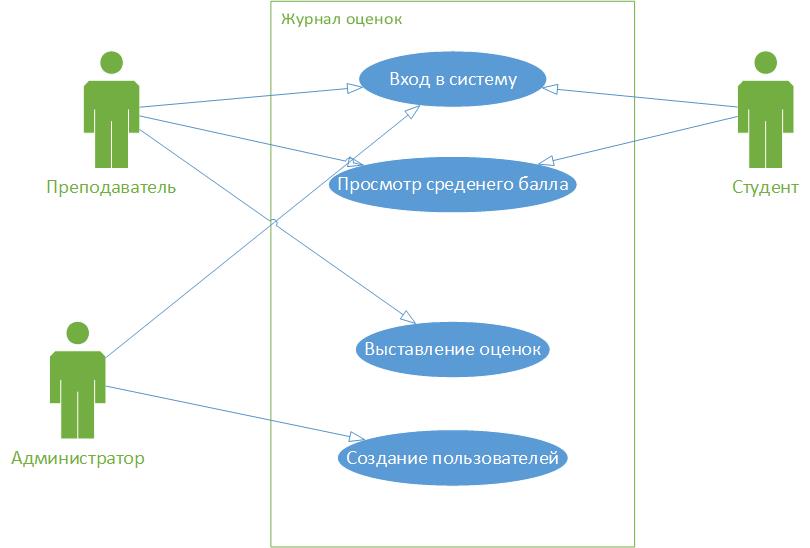


Рисунок 2 Диаграмма вариантов использования

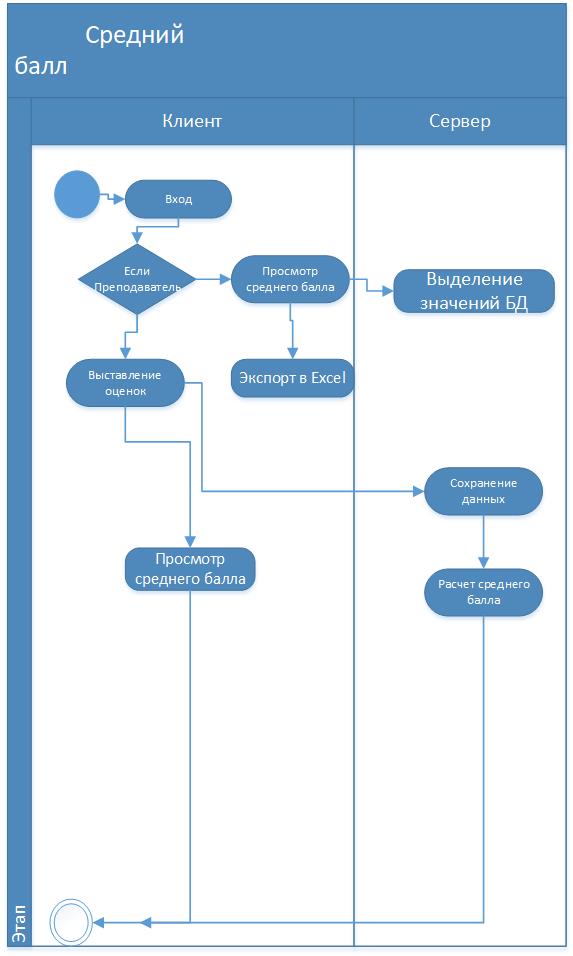


Рисунок 3 Диаграмма деятельности

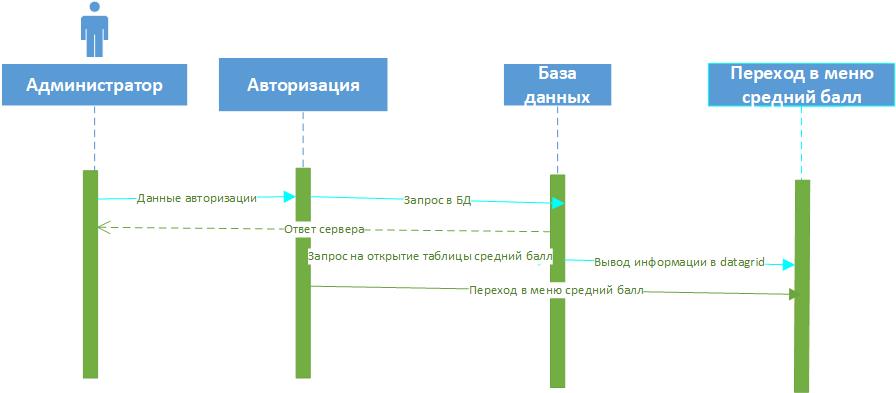


Рисунок 4 Диаграмма последовательности

## Проектирование логики программного модуля

Программный модуль разработан на языке программирования C# платформе WPF. Логика моего приложения представлена в виде схемы.

Рисунок 5 Схема логики приложения

Авторизация

Средний балл

Оценки обучающихся

Обучающиеся

Основной модуль

**Модуль** – отдельная часть какого-либо приложения.

**Модуль авторизация** – в данном модуле пользователь системы входит в свою учетную запись введя свои логин и пароль. Система при вводе данных проверяет есть ли пользователь с введенными данными в базе данных системы. В случае если пользователь ввел правильные данные для входа, то система открывает модуль «Средний балл» в противном случае система выводит ошибку в которой говорится: «Пользователь не найден. Проверьте логин или пароль.

**Модуль «Средний балл»** - в данном модуле пользователю выводится таблица со списком обучающихся и их средним баллом по трем предметам:

* Высшая математика
* История России
* Физика

Также в данном модуле расположен экспорт в таблицу Excel. Связи с чем пользователь имеет возможность распечатать таблицу Excel или отправить её файлом.

**Оценки обучающихся** – в данном модуле педагог может выставлять оценки обучающимся по своему предмету, а также просматривать оценки. Доступ к данному модулю имеют только педагоги

**Модуль «Обучающиеся»** - в данном модуле Администраторы системы добавляют обучающихся в систему в последующем преподаватели выставляют им оценки, а система рассчитывает средний балл.

# СРЕДСТВА РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

## Программная реализация приложения.

## Проектирование интерфейса приложения

Модуль “Авторизация”

**Таблица 1 Модуль «Авторизация»**

|  |  |  |  |
| --- | --- | --- | --- |
| Объект | Название | Свойства | Значение |
| textBox | Login\_box | Size | 242,43 |
| PasswordBox | password\_box | Size | 242,43 |

**Таблица 2 Модуль «Средний балл»**

|  |  |  |  |
| --- | --- | --- | --- |
| Объект | Название | Свойства | Значение |
| dataGrid | Middle\_mar | Size | 1080,553 |
| AutoGenerateColumns | False |
| Button | btn\_Print | Content | Экспорт |
|  |  |

**Таблица 3 Модуль «Оценки обучающийся»**

|  |  |  |  |
| --- | --- | --- | --- |
| Объект | Название | Свойства | Значение |
| dataGrid | db\_marks | Size | 1084,366 |
| AutoGenerateColumns | False |
| ComboBox | Students\_combo | Size | 120, 27.28 |
| TextBox | mark\_text | Size | 120,24 |
| Button |  | Size | 77.65, 27.94 |
| Content | Записать |
| ComboBox | Search\_students\_marks | Size | 120,27.28 |
| ComboBox | Search\_subject\_marks1 | Size | 120,27.28 |
| Button |  | Content | Поиск |
| Size | 69,27 |

**Таблица 4 Модуль «Обучающийся»**

|  |  |  |  |
| --- | --- | --- | --- |
| Объект | Название | Свойства | Значение |
| TextBox | surname\_box | Size | 190,37 |
| TextBox | name\_box | Size | 190,37 |
| TextBox | group\_box | Size | 190,37 |
| Button |  | Content | Создать |
| Size | 148, 45 |

## Программирование и интерпретация результатов

Листинг 1 «Программный код приложения»

|  |  |
| --- | --- |
| 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.SQLite;  using System.Data;  using static System.Net.Mime.MediaTypeNames;  using System.Xml.Linq;  using iTextSharp.text;  using iTextSharp.text.pdf;  using iTextSharp.text.html;  using iTextSharp;  using ClosedXML;  using ClosedXML.Excel;  namespace magazine\_marks  {  /// <summary>  /// Логика взаимодействия для MainWindow.xaml  /// </summary>  public partial class MainWindow : Window  {  public MainWindow()  {  InitializeComponent();  }  public string role;    public string user;  public string logins;  public string passwords;    //Кнопка входа проверка логина и пароля  Jonner j = new Jonner();  public void Button\_Click(object sender, RoutedEventArgs e)  {    if(j.logining(login\_box.Text, password\_box.Password) == "OK")  {  j.logining(login\_box.Text, password\_box.Password);  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  log\_in\_.Visibility = Visibility.Hidden;  back\_gr1.Visibility = Visibility.Hidden;  GUI.Visibility = Visibility.Visible;  SQLiteCommand command1 = new SQLiteCommand();  command1.Connection = conn;  command1.CommandText = "SELECT \* FROM magazine\_marks";  command1.ExecuteNonQuery();  DataTable dt1 = new DataTable();  SQLiteDataAdapter adap1 = new SQLiteDataAdapter(command1);  adap1.Fill(dt1);  middle\_mar.ItemsSource = dt1.DefaultView;    MessageBox.Show(j.roles() + " " + j.subjects());  conn.Close();  username\_text.Content = j.users() + " ( " + j.subjects() + " ) ";  if (j.roles() == "Администратор")  {  cr\_students.Visibility = Visibility.Visible;    }  else  {  cr\_students.Visibility = Visibility.Hidden;  }  }  }  private void Window\_Activated(object sender, EventArgs e)  {  //Выводить таблицу с средними баллами когда при активной форме  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  SQLiteCommand command1 = new SQLiteCommand();  command1.Connection = conn;  command1.CommandText = "SELECT \* FROM magazine\_marks";  command1.ExecuteNonQuery();  DataTable dt1 = new DataTable();  SQLiteDataAdapter adap1 = new SQLiteDataAdapter(command1);  adap1.Fill(dt1);  middle\_mar.ItemsSource = dt1.DefaultView;  conn.Close();  if (mark\_text.Text == "1")  {  mark\_text.Text = "1";  }  if (mark\_text.Text == "2")  {  mark\_text.Text = "2";  }  if (mark\_text.Text == "3")  {  mark\_text.Text = "3";  }  if (mark\_text.Text == "4")  {  mark\_text.Text = "4";  }  if (mark\_text.Text == "5")  {  mark\_text.Text = "5";  }  else  {  mark\_text.Clear();  }  if (j.roles() == "Администратор")  {  cr\_students.Visibility = Visibility.Visible;      }  else  {  cr\_students.Visibility = Visibility.Hidden;  }  }  private void Window\_Deactivated(object sender, EventArgs e)  {  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  SQLiteCommand command1 = new SQLiteCommand();  command1.Connection = conn;  command1.CommandText = "SELECT \* FROM magazine\_marks";  command1.ExecuteNonQuery();  DataTable dt1 = new DataTable();  SQLiteDataAdapter adap1 = new SQLiteDataAdapter(command1);  adap1.Fill(dt1);  middle\_mar.ItemsSource = dt1.DefaultView;  conn.Close();  conn.Open();  SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "SELECT name || ' ' || surname FROM students";  command.ExecuteNonQuery();  SQLiteDataAdapter adapter = new SQLiteDataAdapter(command);  DataSet ds = new DataSet();  adapter.Fill(ds);  Students\_combo.ItemsSource = ds.Tables[0].DefaultView;  Students\_combo.DisplayMemberPath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  Students\_combo.SelectedValuePath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  Search\_students\_marks.ItemsSource = ds.Tables[0].DefaultView;  Search\_students\_marks.DisplayMemberPath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  Search\_students\_marks.SelectedValuePath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  conn.Close();  if (j.roles() == "Администратор")  {  cr\_students.Visibility = Visibility.Visible;  }  else  {  cr\_students.Visibility = Visibility.Hidden;  }  }  private void Window\_SizeChanged(object sender, SizeChangedEventArgs e)  {  }  private void Label\_MouseDown(object sender, MouseButtonEventArgs e)  {  db\_middle\_marks.Visibility = Visibility.Visible;  marks\_panel.Visibility = Visibility.Hidden;  students\_panel.Visibility = Visibility.Hidden;  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  SQLiteCommand command1 = new SQLiteCommand();  command1.Connection = conn;  command1.CommandText = "SELECT \* FROM magazine\_marks";  command1.ExecuteNonQuery();  DataTable dt1 = new DataTable();  SQLiteDataAdapter adap1 = new SQLiteDataAdapter(command1);  adap1.Fill(dt1);  middle\_mar.ItemsSource = dt1.DefaultView;  conn.Close();  }  private void Label\_MouseDown\_1(object sender, MouseButtonEventArgs e)  {    if(j.roled != "Преподаватель")  {  MessageBox.Show("У вас нет прав доступа к этому компоненту");  }  else  {  db\_middle\_marks.Visibility = Visibility.Hidden;  marks\_panel.Visibility = Visibility.Visible;  students\_panel.Visibility = Visibility.Hidden;  }  //Заполнение таблицы db\_marks базой данных  }  private void Win1\_Loaded(object sender, RoutedEventArgs e)  {  try  {  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "SELECT name || ' ' || surname FROM students";  command.ExecuteNonQuery();  SQLiteDataAdapter adapter = new SQLiteDataAdapter(command);  DataSet ds = new DataSet();  adapter.Fill(ds);  Students\_combo.ItemsSource = ds.Tables[0].DefaultView;  Students\_combo.DisplayMemberPath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  Students\_combo.SelectedValuePath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  Search\_students\_marks.ItemsSource = ds.Tables[0].DefaultView;  Search\_students\_marks.DisplayMemberPath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  Search\_students\_marks.SelectedValuePath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  conn.Close();    }  catch (Exception ex)  {  MessageBox.Show(ex.ToString());  }  }  private void Button\_Click\_1(object sender, RoutedEventArgs e)  {  int st\_fail = 0;  try  {    string mark = mark\_text.Text;  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  if (mark == "5")  {      SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "INSERT INTO marks(subject,id\_student,mark) VALUES (@s,@is,@m)";  command.Parameters.AddWithValue("@s", j.subjects());  command.Parameters.AddWithValue("@is", Students\_combo.Text.ToString());  command.Parameters.AddWithValue("@m", mark);  command.ExecuteNonQuery();  }  else if (mark == "4")  {    SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "INSERT INTO marks(subject,id\_student,mark) VALUES (@s,@is,@m)";  command.Parameters.AddWithValue("@s", j.subjects());  command.Parameters.AddWithValue("@is", Students\_combo.Text.ToString());  command.Parameters.AddWithValue("@m", mark);  command.ExecuteNonQuery();  }  else if (mark == "3")  {    SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "INSERT INTO marks(subject,id\_student,mark) VALUES (@s,@is,@m)";  command.Parameters.AddWithValue("@s", j.subjects());  command.Parameters.AddWithValue("@is", Students\_combo.Text.ToString());  command.Parameters.AddWithValue("@m", mark);  command.ExecuteNonQuery();  }  else if (mark == "2")  {    SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "INSERT INTO marks(subject,id\_student,mark) VALUES (@s,@is,@m)";  command.Parameters.AddWithValue("@s", j.subjects());  command.Parameters.AddWithValue("@is", Students\_combo.Text.ToString());  command.Parameters.AddWithValue("@m", mark);  command.ExecuteNonQuery();  }  else if (mark == "1")  {    SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "INSERT INTO marks(subject,id\_student,mark) VALUES (@s,@is,@m)";  command.Parameters.AddWithValue("@s", j.subjects());  command.Parameters.AddWithValue("@is", Students\_combo.Text.ToString());  command.Parameters.AddWithValue("@m", mark);  command.ExecuteNonQuery();  }  //if (mark\_text.Text != "1" && mark\_text.Text != "2" && mark\_text.Text != "3" && mark\_text.Text != "4" && mark\_text.Text != "5")  else  {  st\_fail++;  mark\_text.Text = " ";  MessageBox.Show("Внимание не корректное значение");  }  SQLiteCommand command1 = new SQLiteCommand();  command1.Connection = conn;  command1.CommandText = "SELECT avg(mark) FROM marks WHERE id\_student =@studentval AND subject = @sub";  command1.Parameters.AddWithValue("@studentval", Students\_combo.Text.ToString());  command1.Parameters.AddWithValue("@sub", j.subjects());  command1.ExecuteNonQuery();  SQLiteDataAdapter adapter = new SQLiteDataAdapter(command1);  DataSet ds = new DataSet();  adapter.Fill(ds, "marks");  DataRow row = ds.Tables[0].Rows[0];    SQLiteCommand command2 = new SQLiteCommand();  command2.Connection = conn;  if (j.subjects() == "Физика")  {  command2.CommandText = "UPDATE magazine\_marks SET middle\_marks2 = @mm WHERE id\_student = @is";  command2.Parameters.AddWithValue("@is", Students\_combo.Text.ToString());  command2.Parameters.AddWithValue("@mm", Math.Round(Convert.ToDouble(row[0]),2));  }  else if (j.subjects() == "Высшая математика")  {  command2.CommandText = "UPDATE magazine\_marks SET middle\_marks1 = @mm WHERE id\_student = @is";  command2.Parameters.AddWithValue("@is", Students\_combo.Text.ToString());  command2.Parameters.AddWithValue("@mm", Math.Round(Convert.ToDouble(row[0]),2));  }  else if (j.subjects() == "История России")  {  command2.CommandText = "UPDATE magazine\_marks SET middle\_marks3 = @mm WHERE id\_student = @is";  command2.Parameters.AddWithValue("@is", Students\_combo.Text.ToString());  command2.Parameters.AddWithValue("@mm", Math.Round(Convert.ToDouble(row[0]),2));  }  command2.ExecuteNonQuery();  if(st\_fail== 0)  {  MessageBox.Show("OK");  }    conn.Close();    }  catch (Exception ex)  {  MessageBox.Show("Ошибка возможно у вас нет необходимых прав для данной операции " + ex.Message);  }  }  private void Button\_Click\_2(object sender, RoutedEventArgs e)  {  string students = Search\_students\_marks.Text;  string subjects = Search\_subject\_marks1.Text;  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "SELECT \* FROM marks WHERE subject = @sub AND id\_student = @is";  command.Parameters.AddWithValue("@sub", subjects);  command.Parameters.AddWithValue("@is", students);  command.ExecuteNonQuery();  DataTable dt = new DataTable();  SQLiteDataAdapter adap = new SQLiteDataAdapter(command);  adap.Fill(dt);    db\_marks.ItemsSource = dt.DefaultView;  }  private void btn\_Print\_Click(object sender, RoutedEventArgs e)  {    }  private void btn\_Print\_Click\_1(object sender, RoutedEventArgs e)  {  try  {  var wb = new XLWorkbook();  var sh = wb.Worksheets.Add("Export");  for (int j = 0; j < middle\_mar.Columns.Count; j++)  {  sh.Cell(1, j + 1).SetValue(middle\_mar.Columns[j].Header);  sh.Cell(1, j + 1).Style.Font.Bold = true;  sh.Columns().AdjustToContents();  sh.Rows().AdjustToContents();  }  int q = 1;  for (int i = 0; i < middle\_mar.Columns.Count; i++)  {    for (int j = 0; j < middle\_mar.Items.Count; j++)  {  DataRowView row = middle\_mar.Items[j] as DataRowView;  if(row != null)  sh.Cell(j + 2, i + 1).SetValue(row[i+1].ToString());  }  }  wb.SaveAs("First.xlsx");  }  catch(Exception ex)  {  MessageBox.Show( ex.Message);  }  }  private void mark\_text\_PreviewKeyDown(object sender, KeyEventArgs e)  {  if(mark\_text.Text == "1")  {  mark\_text.Text = "1";  }  if(mark\_text.Text == "2")  {  mark\_text.Text = "2";  }  if (mark\_text.Text == "3")  {  mark\_text.Text = "3";  }  if (mark\_text.Text == "4")  {  mark\_text.Text = "4";  }  if (mark\_text.Text == "5")  {  mark\_text.Text = "5";  }  else  {  mark\_text.Text = " ";  }  }  private void mark\_text\_KeyDown(object sender, KeyEventArgs e)  {  if (mark\_text.Text == "1")  {  mark\_text.Text = "1";  }  if (mark\_text.Text == "2")  {  mark\_text.Text = "2";  }  if (mark\_text.Text == "3")  {  mark\_text.Text = "3";  }  if (mark\_text.Text == "4")  {  mark\_text.Text = "4";  }  if (mark\_text.Text == "5")  {  mark\_text.Text = "5";  }  else if (mark\_text.Text != null)  {  mark\_text.Text = "";  }  }  private void TextBlock\_MouseDown(object sender, MouseButtonEventArgs e)  {  if (j.roles() == "Администратор")  {  db\_middle\_marks.Visibility = Visibility.Hidden;  marks\_panel.Visibility = Visibility.Hidden;  students\_panel.Visibility = Visibility.Visible;  }  else  {  MessageBox.Show("У вас нет прав доступа к этому разделу");  }  }  private void Button\_Click\_3(object sender, RoutedEventArgs e)  {  try  {  string surn = surname\_box.Text;  string names = name\_box.Text;  string group = group\_box.Text;  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  string com = "INSERT INTO students (surname,name,st\_group) VALUES (@sn,@n,@gr)";  SQLiteCommand cmd = new SQLiteCommand();  cmd.CommandText = com;  cmd.Parameters.AddWithValue("@sn", surn);  cmd.Parameters.AddWithValue("@n", names);  cmd.Parameters.AddWithValue("@gr", group);  cmd.Connection = conn;  cmd.ExecuteNonQuery();  SQLiteCommand command = new SQLiteCommand();  string coms = "INSERT INTO magazine\_marks(id\_student) VALUES (@is)";  command.Connection = conn;  command.CommandText = coms;  command.Parameters.AddWithValue("@is", names + " " + surn);  command.ExecuteNonQuery();  conn.Close();  }  catch(Exception ex)  {  MessageBox.Show(ex.ToString());  }  }  private void Win1\_Activated(object sender, EventArgs e)  {  try  {  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "SELECT name || ' ' || surname FROM students";  command.ExecuteNonQuery();  SQLiteDataAdapter adapter = new SQLiteDataAdapter(command);  DataSet ds = new DataSet();  adapter.Fill(ds);  Students\_combo.ItemsSource = ds.Tables[0].DefaultView;  Students\_combo.DisplayMemberPath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  Students\_combo.SelectedValuePath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  Search\_students\_marks.ItemsSource = ds.Tables[0].DefaultView;  Search\_students\_marks.DisplayMemberPath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  Search\_students\_marks.SelectedValuePath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  conn.Close();  }  catch(Exception ex)  {  MessageBox.Show(ex.ToString());  }  }    private void surname\_box\_PreviewTextInput(object sender, TextCompositionEventArgs e)//Ограничение ввода. Активен ввод только букв  {  if (int.TryParse(e.Text, out int i))  {  e.Handled = true;  }    }  private void name\_box\_PreviewTextInput(object sender, TextCompositionEventArgs e) //Ограничение ввода. Активен ввод только букв  {  if (int.TryParse(e.Text, out int i))  {  e.Handled = true;  }  }  }  }    Листинг 2 «Класс Jonner»   |  | | --- | | using System;  using System.Collections.Generic;  using System.Data.SQLite;  using System.Data;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows;  using System.Xml.Linq;  using magazine\_marks;  using Xunit;  using System.ComponentModel.DataAnnotations;  using System.Dynamic;  namespace magazine\_marks  {  public class Jonner  {  public string roled;  public string subj;  public string us;    public string logining(string login, string password)  {  string status = "0";  try  {  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "SELECT role,subject,surname ||' '|| name FROM users WHERE login = @log AND password = @pass";  command.Parameters.Add("@log", DbType.String).Value = login;  command.Parameters.Add("@pass", DbType.String).Value = password;  command.ExecuteNonQuery();  DataTable dt = new DataTable();  SQLiteDataAdapter adap = new SQLiteDataAdapter(command);  adap.Fill(dt);  if (dt.Rows.Count > 0)  {    status = "OK";  roled = Convert.ToString(dt.Rows[0].ItemArray[0]);  subj = Convert.ToString(dt.Rows[0].ItemArray[1]);  us = Convert.ToString(dt.Rows[0].ItemArray[2]);  }  else  {  conn.Close();  MessageBox.Show("Пользователь не найден. Проверьте логин или пароль");  status = "NO";  }  return status;  }  catch (Exception ex)  {  MessageBox.Show(ex.ToString());  return status;  }  }  public string roles()  {  return roled;  }  public string subjects()  {  return subj;  }  public string users()  {  return us;  }  public string GetVal(string Y)  {  return Y;  }  }  } | |

Листинг 3

|  |
| --- |
| <Window x:Class="magazine\_marks.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:magazine\_marks"  mc:Ignorable="d"  Title="MainWindow" Height="720" Width="1080" Deactivated="Window\_Deactivated" SizeChanged="Window\_SizeChanged"  x:Name="Win1" Loaded="Win1\_Loaded" Activated="Win1\_Activated"  >  <Grid>  <Grid x:Name="back\_gr1" Visibility="Visible">  <Image x:Name ="image" HorizontalAlignment="Left" Height="697" Margin="10,10,0,0" VerticalAlignment="Top" Width="1920" Source="/p1.jpg" Stretch="Fill" MaxWidth="1920" OpacityMask="#FFE5CF07" MaxHeight="2000"/>  </Grid>  <StackPanel x:Name="log\_in\_" HorizontalAlignment="Center" Width="1080" Visibility="Visible">  <StackPanel.Background>  <ImageBrush ImageSource="/back.jpg" Opacity="0.1"/>  </StackPanel.Background>  <TextBlock TextWrapping="Wrap" Text="Добро пожаловать!" HorizontalAlignment="Center" VerticalAlignment="Top" FontSize="40" FontFamily="Courier New" Foreground="Black" Margin="0,100,0,0"/>  <TextBox x:Name="login\_box" TextWrapping="NoWrap" Width="242" HorizontalAlignment="Center" Height="43" Background="#FFF1F1F1" Margin="0,82,0,0" FontSize="25" />  <PasswordBox x:Name="password\_box" Width="242" HorizontalAlignment="Center" Height="43" Background="#FFF4EFEF" Margin="0,60,-1,0" FontSize="25" />  <Button Content="Войти" HorizontalAlignment="Center" Foreground="White" FontSize="40" Background="#FF9100FE" BorderBrush="#FF6F14A4" Padding="1,-1,1,1" UseLayoutRounding="True" Height="83" BorderThickness="0,0,0,0" Margin="0,60,0,0" Width="144" FontFamily="Courier New" Click="Button\_Click"/>  <Label Content="Логин" HorizontalAlignment="Center" VerticalAlignment="Top" FontSize="20" Margin="0,-326,162,0" FontFamily="Courier New" Foreground="Black"/>  <Label Content="Пароль" FontSize="20" HorizontalAlignment="Center" VerticalAlignment="Top" Margin="0,-221,157,0" FontFamily="Courier New" Foreground="Black"/>  </StackPanel>  <StackPanel Visibility="Hidden" x:Name="GUI">  <StackPanel.Background>  <ImageBrush ImageSource="/back.jpg" Opacity="0.3"/>  </StackPanel.Background>  <StackPanel x:Name="menu" Background="#FF1414DD">  <Label Content="Средний балл" Foreground="White" MouseDown="Label\_MouseDown"/>  <Label Content="Оценки обучающихся" Foreground="White" Margin="92,-27,0,0" HorizontalAlignment="Left" MouseDown="Label\_MouseDown\_1" x:Name="marks\_label"/>  <Label x:Name="username\_text" Content="" Foreground="White" HorizontalAlignment="Right" FontSize="14" Margin="0,-27,0,0"/>  <TextBlock TextWrapping="Wrap" Text="Обучающийся" Foreground="White" Margin="240,-23,0,0" HorizontalAlignment="Left" MouseDown="TextBlock\_MouseDown" x:Name ="cr\_students"/>  </StackPanel>  <StackPanel Height="664" Visibility="Hidden" x:Name="db\_middle\_marks" VerticalAlignment="Top">  <StackPanel.Background>  <SolidColorBrush Color="#FFF3F0F0" Opacity="0.01"/>  </StackPanel.Background>  <TextBlock TextWrapping="Wrap" Text="Средний балл обучающихся" HorizontalAlignment="Left" FontSize="25" FontWeight="Bold" Margin="0,9,0,0"/>  <DataGrid x:Name="middle\_mar" Height="553" d:ItemsSource="{d:SampleData ItemCount=5}" AutoGenerateColumns="False" FontSize="14">  <DataGrid.Background>  <SolidColorBrush Color="#FFF0F0F0" Opacity="0"/>  </DataGrid.Background>  <DataGrid.Columns>  <DataGridTextColumn Header="Студент" Binding ="{Binding Path=id\_student}" IsReadOnly="True" FontSize="16"/>  <DataGridTextColumn Header="Высшая математика" Binding ="{Binding Path=middle\_marks1}" IsReadOnly="True" FontSize="16"/>  <DataGridTextColumn Header="Физика" Binding ="{Binding Path=middle\_marks2}" IsReadOnly="True" FontSize="16"/>  <DataGridTextColumn Header="История России" Binding ="{Binding Path=middle\_marks3}" IsReadOnly="True" FontSize="16"/>  </DataGrid.Columns>  </DataGrid>  <Button x:Name="btn\_Print" Content="Экспорт" HorizontalAlignment="Right" FontSize="20" Margin="0,5,38,0" Height="55" Width="97" Background="#FF7239BA" Foreground="White" Click="btn\_Print\_Click\_1"/>  </StackPanel>  <StackPanel Height="664" Margin="-4,-660,0,0" Visibility="Hidden" x:Name="marks\_panel">  <ComboBox x:Name="Students\_combo" Width="120" Margin="-19,8,-178,0" HorizontalAlignment="Center" FontSize="16"/>  <TextBlock TextWrapping="Wrap" Text="Обучающийся" HorizontalAlignment="Center" Margin="-63,-20,21,0" FontSize="16"/>  <TextBox x:Name ="mark\_text" TextWrapping="Wrap" Width="120" Margin="158,23,0,0" Height="24" FontSize="18" PreviewKeyDown="mark\_text\_PreviewKeyDown" KeyDown="mark\_text\_KeyDown"/>  <TextBlock TextWrapping="Wrap" Text="Оценка" HorizontalAlignment="Center" Margin="-26,-24,0,0" FontSize="16"/>  <Button Content="Записать" FontSize="18" HorizontalAlignment="Center" Margin="108,24,0,0" VerticalAlignment="Top" Click="Button\_Click\_1"/>  <DataGrid x:Name="db\_marks" Height="366" d:ItemsSource="{d:SampleData ItemCount=4}" Margin="0,103.8,0,0" AutoGenerateColumns="False" ItemsSource="{Binding Test}">  <DataGrid.Background>  <SolidColorBrush Color="#FFF0F0F0" Opacity="0"/>  </DataGrid.Background>  <DataGrid.Columns>  <DataGridTextColumn Header="Предмет" Binding ="{Binding Path=subject}" IsReadOnly="True" FontSize="16" MaxWidth="300"/>  <DataGridTextColumn Header="Обучающийся" Binding ="{Binding Path=id\_student}" IsReadOnly="True" FontSize="16" MaxWidth="300"/>  <DataGridTextColumn Header="Оценки" Binding ="{Binding Path=mark}" IsReadOnly="True" FontSize="16" MaxWidth="300"/>  </DataGrid.Columns>  </DataGrid>  <StackPanel Height="45" Margin="0,-829,0,0" Background="#FF121078">  <TextBlock TextWrapping="Wrap" Text="Обучающийся" FontSize="18" HorizontalAlignment="Right" Foreground="White" Margin="-12,9,675,0"/>  <ComboBox x:Name="Search\_students\_marks" Width="119" Margin="-4,-22,521,0" HorizontalAlignment="Right" FontSize="16"/>  <TextBlock TextWrapping="Wrap" Text="Предмет" HorizontalAlignment="Right" VerticalAlignment="Top" Foreground="White" FontSize="18" Margin="0,-29,368,0"/>  <ComboBox x:Name="Search\_subject\_marks1" Width="120" HorizontalAlignment="Right" Margin="0,-30,220,0" ItemsSource="{Binding}" FontSize="16" VerticalAlignment="Top">  <TextBlock Text="Физика"/>  <TextBlock Text="Высшая математика"/>  <TextBlock Text="История России"/>  </ComboBox>  <Button Content="Поиск" HorizontalAlignment="Right" Margin="0,-26,36,0" Width="57" Height="27" FontSize="16" Background="#FF7239BA" Foreground="White" Click="Button\_Click\_2"/>  </StackPanel>  </StackPanel>  <StackPanel Height="664" HorizontalAlignment="Left" Width="1078" Margin="0,-662,0,0" Opacity="100" x:Name="students\_panel" Visibility="Visible" >  <TextBlock TextWrapping="Wrap" Text="Добавление обучающихся" HorizontalAlignment="Left" Foreground="Black" Width="393" Height="55" Margin="0,5,0,0" FontSize="30"/>  <TextBox x:Name="surname\_box" TextWrapping="Wrap" Width="187" FontSize="20" Height="37" PreviewTextInput="surname\_box\_PreviewTextInput"/>  <TextBox x:Name="name\_box" TextWrapping="Wrap" Width="190" FontSize="20" Height="37" Margin="0,20,0,0" PreviewTextInput="name\_box\_PreviewTextInput"/>  <TextBox x:Name="group\_box" TextWrapping="Wrap" Width="190" FontSize="20" Height="37" Margin="0,33,0,0"/>  <TextBlock TextWrapping="Wrap" Text="Фамилия" VerticalAlignment="Center" HorizontalAlignment="Center" FontSize="16" Padding="0,3,0,0" Margin="0,-293,281,0"/>  <TextBlock TextWrapping="Wrap" Text="Имя" VerticalAlignment="Center" HorizontalAlignment="Center" FontSize="16" Padding="0,3,0,0" Margin="0,-178,281,0"/>  <TextBlock TextWrapping="Wrap" Text="Группа  " VerticalAlignment="Center" HorizontalAlignment="Center" FontSize="16" Padding="0,3,0,0" Margin="0,-42,281,0"/>  <Button Content="Создать" HorizontalAlignment="Center" Margin="0,21,0,0" Width="148" Height="45" VerticalAlignment="Top" FontSize="15" Click="Button\_Click\_3"/>  </StackPanel>    </StackPanel>  </Grid>  </Window> |

|  |  |
| --- | --- |
| Объекты | Событийно-управляемые процедуры |
| Button | public void Button\_Click(object sender, RoutedEventArgs e)  {    if(j.logining(login\_box.Text, password\_box.Password) == "OK")  {  j.logining(login\_box.Text, password\_box.Password);  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  log\_in\_.Visibility = Visibility.Hidden;  back\_gr1.Visibility = Visibility.Hidden;  GUI.Visibility = Visibility.Visible;  SQLiteCommand command1 = new SQLiteCommand();  command1.Connection = conn;  command1.CommandText = "SELECT \* FROM magazine\_marks";  command1.ExecuteNonQuery();  DataTable dt1 = new DataTable();  SQLiteDataAdapter adap1 = new SQLiteDataAdapter(command1);  adap1.Fill(dt1);  middle\_mar.ItemsSource = dt1.DefaultView;    MessageBox.Show(j.roles() + " " + j.subjects());  conn.Close();  username\_text.Content = j.users() + " ( " + j.subjects() + " ) ";  if (j.roles() == "Администратор")  {  cr\_students.Visibility = Visibility.Visible;    }  else  {  cr\_students.Visibility = Visibility.Hidden;  }  }  }  private void Button\_Click\_1(object sender, RoutedEventArgs e)  {  int st\_fail = 0;  try  {    string mark = mark\_text.Text;  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  if (mark == "5")  {      SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "INSERT INTO marks(subject,id\_student,mark) VALUES (@s,@is,@m)";  command.Parameters.AddWithValue("@s", j.subjects());  command.Parameters.AddWithValue("@is", Students\_combo.Text.ToString());  command.Parameters.AddWithValue("@m", mark);  command.ExecuteNonQuery();  }  else if (mark == "4")  {    SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "INSERT INTO marks(subject,id\_student,mark) VALUES (@s,@is,@m)";  command.Parameters.AddWithValue("@s", j.subjects());  command.Parameters.AddWithValue("@is", Students\_combo.Text.ToString());  command.Parameters.AddWithValue("@m", mark);  command.ExecuteNonQuery();  }  else if (mark == "3")  {    SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "INSERT INTO marks(subject,id\_student,mark) VALUES (@s,@is,@m)";  command.Parameters.AddWithValue("@s", j.subjects());  command.Parameters.AddWithValue("@is", Students\_combo.Text.ToString());  command.Parameters.AddWithValue("@m", mark);  command.ExecuteNonQuery();  }  else if (mark == "2")  {    SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "INSERT INTO marks(subject,id\_student,mark) VALUES (@s,@is,@m)";  command.Parameters.AddWithValue("@s", j.subjects());  command.Parameters.AddWithValue("@is", Students\_combo.Text.ToString());  command.Parameters.AddWithValue("@m", mark);  command.ExecuteNonQuery();  }  else if (mark == "1")  {    SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "INSERT INTO marks(subject,id\_student,mark) VALUES (@s,@is,@m)";  command.Parameters.AddWithValue("@s", j.subjects());  command.Parameters.AddWithValue("@is", Students\_combo.Text.ToString());  command.Parameters.AddWithValue("@m", mark);  command.ExecuteNonQuery();  }  //if (mark\_text.Text != "1" && mark\_text.Text != "2" && mark\_text.Text != "3" && mark\_text.Text != "4" && mark\_text.Text != "5")  else  {  st\_fail++;  mark\_text.Text = " ";  MessageBox.Show("Внимание не корректное значение");  }  SQLiteCommand command1 = new SQLiteCommand();  command1.Connection = conn;  command1.CommandText = "SELECT avg(mark) FROM marks WHERE id\_student =@studentval AND subject = @sub";  command1.Parameters.AddWithValue("@studentval", Students\_combo.Text.ToString());  command1.Parameters.AddWithValue("@sub", j.subjects());  command1.ExecuteNonQuery();  SQLiteDataAdapter adapter = new SQLiteDataAdapter(command1);  DataSet ds = new DataSet();  adapter.Fill(ds, "marks");  DataRow row = ds.Tables[0].Rows[0];    SQLiteCommand command2 = new SQLiteCommand();  command2.Connection = conn;  if (j.subjects() == "Физика")  {  command2.CommandText = "UPDATE magazine\_marks SET middle\_marks2 = @mm WHERE id\_student = @is";  command2.Parameters.AddWithValue("@is", Students\_combo.Text.ToString());  command2.Parameters.AddWithValue("@mm", Math.Round(Convert.ToDouble(row[0]),2));  }  else if (j.subjects() == "Высшая математика")  {  command2.CommandText = "UPDATE magazine\_marks SET middle\_marks1 = @mm WHERE id\_student = @is";  command2.Parameters.AddWithValue("@is", Students\_combo.Text.ToString());  command2.Parameters.AddWithValue("@mm", Math.Round(Convert.ToDouble(row[0]),2));  }  else if (j.subjects() == "История России")  {  command2.CommandText = "UPDATE magazine\_marks SET middle\_marks3 = @mm WHERE id\_student = @is";  command2.Parameters.AddWithValue("@is", Students\_combo.Text.ToString());  command2.Parameters.AddWithValue("@mm", Math.Round(Convert.ToDouble(row[0]),2));  }  command2.ExecuteNonQuery();  if(st\_fail== 0)  {  MessageBox.Show("OK");  }    conn.Close();    }  catch (Exception ex)  {  MessageBox.Show("Ошибка возможно у вас нет необходимых прав для данной операции " + ex.Message);  }  }  private void Button\_Click\_2(object sender, RoutedEventArgs e)  {  string students = Search\_students\_marks.Text;  string subjects = Search\_subject\_marks1.Text;  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "SELECT \* FROM marks WHERE subject = @sub AND id\_student = @is";  command.Parameters.AddWithValue("@sub", subjects);  command.Parameters.AddWithValue("@is", students);  command.ExecuteNonQuery();  DataTable dt = new DataTable();  SQLiteDataAdapter adap = new SQLiteDataAdapter(command);  adap.Fill(dt);    db\_marks.ItemsSource = dt.DefaultView;  }  private void btn\_Print\_Click\_1(object sender, RoutedEventArgs e)  {  try  {  var wb = new XLWorkbook();  var sh = wb.Worksheets.Add("Export");  for (int j = 0; j < middle\_mar.Columns.Count; j++)  {  sh.Cell(1, j + 1).SetValue(middle\_mar.Columns[j].Header);  sh.Cell(1, j + 1).Style.Font.Bold = true;  sh.Columns().AdjustToContents();  sh.Rows().AdjustToContents();  }  int q = 1;  for (int i = 0; i < middle\_mar.Columns.Count; i++)  {    for (int j = 0; j < middle\_mar.Items.Count; j++)  {  DataRowView row = middle\_mar.Items[j] as DataRowView;  if(row != null)  sh.Cell(j + 2, i + 1).SetValue(row[i+1].ToString());  }  }  wb.SaveAs("First.xlsx");  }  catch(Exception ex)  {  MessageBox.Show( ex.Message);  }  }  private void Button\_Click\_3(object sender, RoutedEventArgs e)  {  try  {  string surn = surname\_box.Text;  string names = name\_box.Text;  string group = group\_box.Text;  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  string com = "INSERT INTO students (surname,name,st\_group) VALUES (@sn,@n,@gr)";  SQLiteCommand cmd = new SQLiteCommand();  cmd.CommandText = com;  cmd.Parameters.AddWithValue("@sn", surn);  cmd.Parameters.AddWithValue("@n", names);  cmd.Parameters.AddWithValue("@gr", group);  cmd.Connection = conn;  cmd.ExecuteNonQuery();  SQLiteCommand command = new SQLiteCommand();  string coms = "INSERT INTO magazine\_marks(id\_student) VALUES (@is)";  command.Connection = conn;  command.CommandText = coms;  command.Parameters.AddWithValue("@is", names + " " + surn);  command.ExecuteNonQuery();  conn.Close();  }  catch(Exception ex)  {  MessageBox.Show(ex.ToString());  }  } |
| Window | private void Window\_Activated(object sender, EventArgs e)  {  //Выводить таблицу с средними баллами когда при активной форме  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  SQLiteCommand command1 = new SQLiteCommand();  command1.Connection = conn;  command1.CommandText = "SELECT \* FROM magazine\_marks";  command1.ExecuteNonQuery();  DataTable dt1 = new DataTable();  SQLiteDataAdapter adap1 = new SQLiteDataAdapter(command1);  adap1.Fill(dt1);  middle\_mar.ItemsSource = dt1.DefaultView;  conn.Close();  if (mark\_text.Text == "1")  {  mark\_text.Text = "1";  }  if (mark\_text.Text == "2")  {  mark\_text.Text = "2";  }  if (mark\_text.Text == "3")  {  mark\_text.Text = "3";  }  if (mark\_text.Text == "4")  {  mark\_text.Text = "4";  }  if (mark\_text.Text == "5")  {  mark\_text.Text = "5";  }  else  {  mark\_text.Clear();  }  if (j.roles() == "Администратор")  {  cr\_students.Visibility = Visibility.Visible;      }  else  {  cr\_students.Visibility = Visibility.Hidden;  }  }  private void Window\_Deactivated(object sender, EventArgs e)  {  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  SQLiteCommand command1 = new SQLiteCommand();  command1.Connection = conn;  command1.CommandText = "SELECT \* FROM magazine\_marks";  command1.ExecuteNonQuery();  DataTable dt1 = new DataTable();  SQLiteDataAdapter adap1 = new SQLiteDataAdapter(command1);  adap1.Fill(dt1);  middle\_mar.ItemsSource = dt1.DefaultView;  conn.Close();  conn.Open();  SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "SELECT name || ' ' || surname FROM students";  command.ExecuteNonQuery();  SQLiteDataAdapter adapter = new SQLiteDataAdapter(command);  DataSet ds = new DataSet();  adapter.Fill(ds);  Students\_combo.ItemsSource = ds.Tables[0].DefaultView;  Students\_combo.DisplayMemberPath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  Students\_combo.SelectedValuePath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  Search\_students\_marks.ItemsSource = ds.Tables[0].DefaultView;  Search\_students\_marks.DisplayMemberPath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  Search\_students\_marks.SelectedValuePath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  conn.Close();  if (j.roles() == "Администратор")  {  cr\_students.Visibility = Visibility.Visible;  }  else  {  cr\_students.Visibility = Visibility.Hidden;  }  }  private void Win1\_Loaded(object sender, RoutedEventArgs e)  {  try  {  SQLiteConnection conn = new SQLiteConnection("Data Source = mm.db; Version = 3");  conn.Open();  SQLiteCommand command = new SQLiteCommand();  command.Connection = conn;  command.CommandText = "SELECT name || ' ' || surname FROM students";  command.ExecuteNonQuery();  SQLiteDataAdapter adapter = new SQLiteDataAdapter(command);  DataSet ds = new DataSet();  adapter.Fill(ds);  Students\_combo.ItemsSource = ds.Tables[0].DefaultView;  Students\_combo.DisplayMemberPath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  Students\_combo.SelectedValuePath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  Search\_students\_marks.ItemsSource = ds.Tables[0].DefaultView;  Search\_students\_marks.DisplayMemberPath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  Search\_students\_marks.SelectedValuePath = ds.Tables[0].Columns["name || ' ' || surname"].ToString();  conn.Close();    }  catch (Exception ex)  {  MessageBox.Show(ex.ToString());  }  } |
| TextBox | private void mark\_text\_PreviewKeyDown(object sender, KeyEventArgs e)  {  if(mark\_text.Text == "1")  {  mark\_text.Text = "1";  }  if(mark\_text.Text == "2")  {  mark\_text.Text = "2";  }  if (mark\_text.Text == "3")  {  mark\_text.Text = "3";  }  if (mark\_text.Text == "4")  {  mark\_text.Text = "4";  }  if (mark\_text.Text == "5")  {  mark\_text.Text = "5";  }  else  {  mark\_text.Text = " ";  }  }  private void mark\_text\_KeyDown(object sender, KeyEventArgs e)  {  if (mark\_text.Text == "1")  {  mark\_text.Text = "1";  }  if (mark\_text.Text == "2")  {  mark\_text.Text = "2";  }  if (mark\_text.Text == "3")  {  mark\_text.Text = "3";  }  if (mark\_text.Text == "4")  {  mark\_text.Text = "4";  }  if (mark\_text.Text == "5")  {  mark\_text.Text = "5";  }  else if (mark\_text.Text != null)  {  mark\_text.Text = "";  }  }  private void surname\_box\_PreviewTextInput(object sender, TextCompositionEventArgs e)//Ограничение ввода. Активен ввод только букв  {  if (int.TryParse(e.Text, out int i))  {  e.Handled = true;  }    }  private void name\_box\_PreviewTextInput(object sender, TextCompositionEventArgs e) //Ограничение ввода. Активен ввод только букв  {  if (int.TryParse(e.Text, out int i))  {  e.Handled = true;  }  } |
| TextBlock | private void TextBlock\_MouseDown(object sender, MouseButtonEventArgs e)  {  if (j.roles() == "Администратор")  {  db\_middle\_marks.Visibility = Visibility.Hidden;  marks\_panel.Visibility = Visibility.Hidden;  students\_panel.Visibility = Visibility.Visible;  }  else  {  MessageBox.Show("У вас нет прав доступа к этому разделу");  }  } |

## Реализация алгоритма интеграции данных.

Алгоритм интеграции данных реализовывался с помощью реализационной базы данных SQLite. База данных хранится в виде файла с расширением \*.db

Ниже приведены таблицы из базы данных.

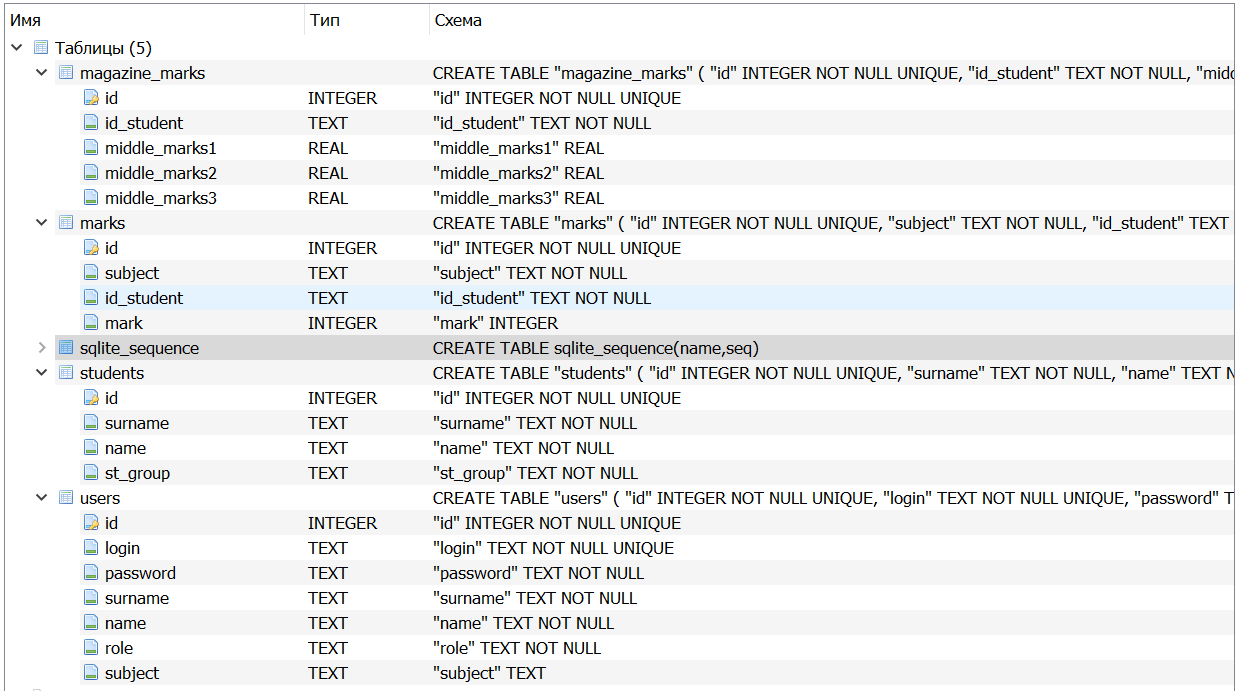
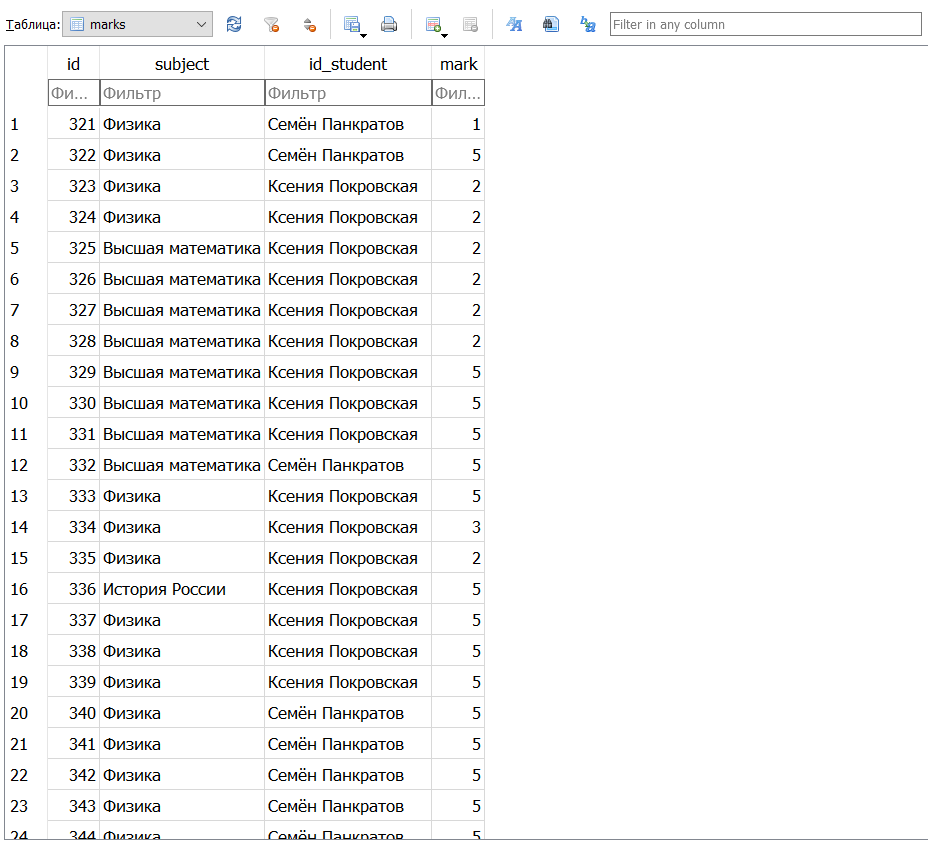


Рисунок 6 Иерархия базы данных

## 

Рисунок 7 Записи в таблице magazine\_marks



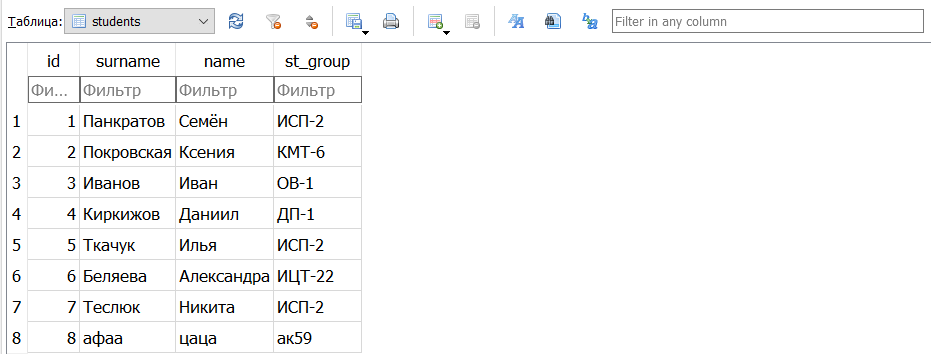
Рисунок 8 Записи в таблице marks

Рисунок 9 Записи в таблице students

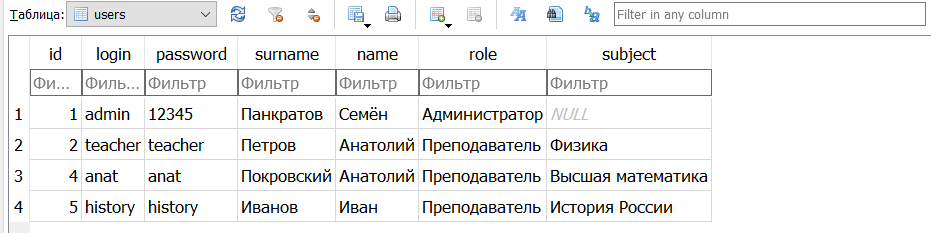


Рисунок 10 Таблица users

## Отладка приложения

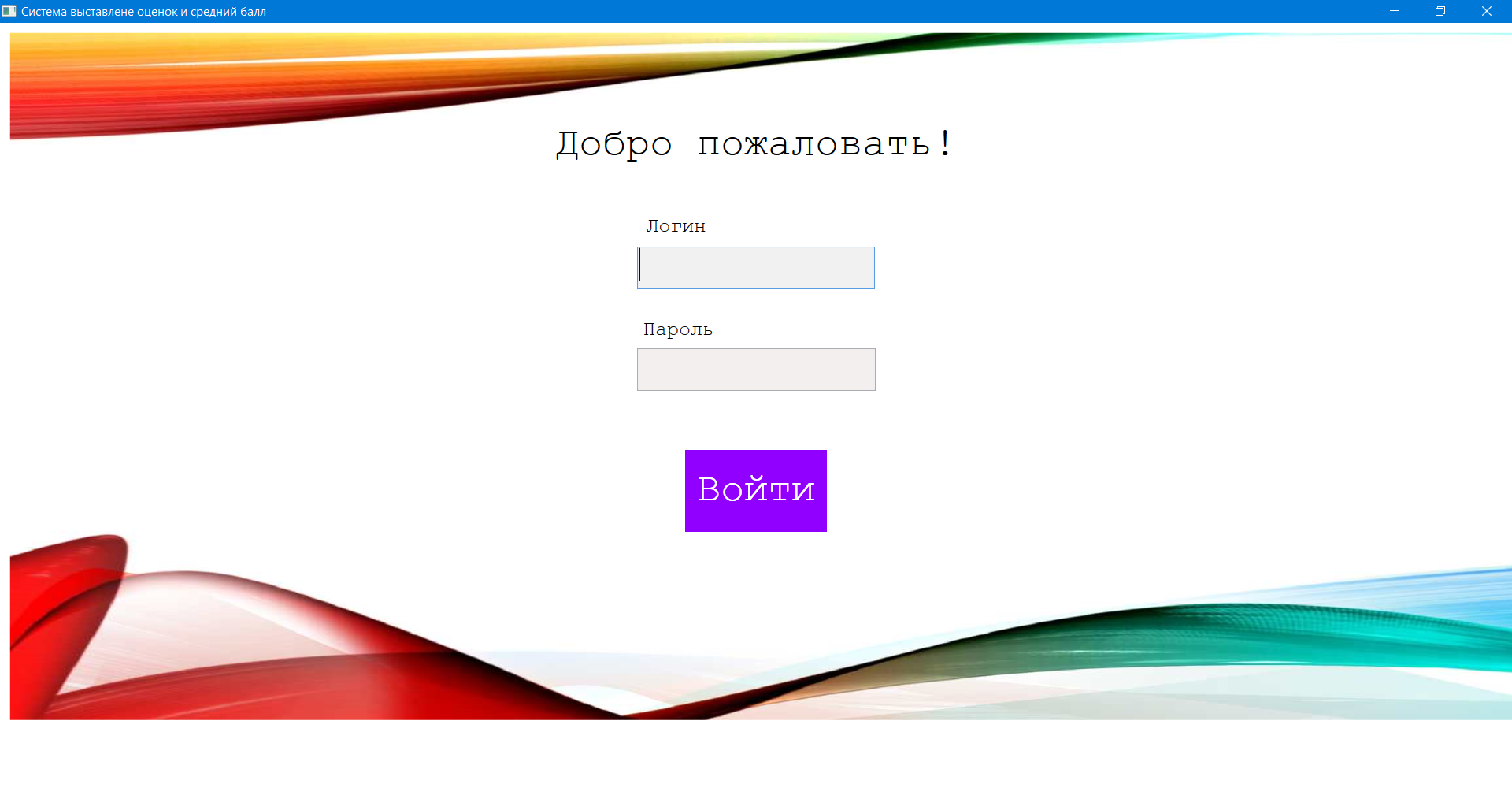


Рисунок 11 Авторизация в системе

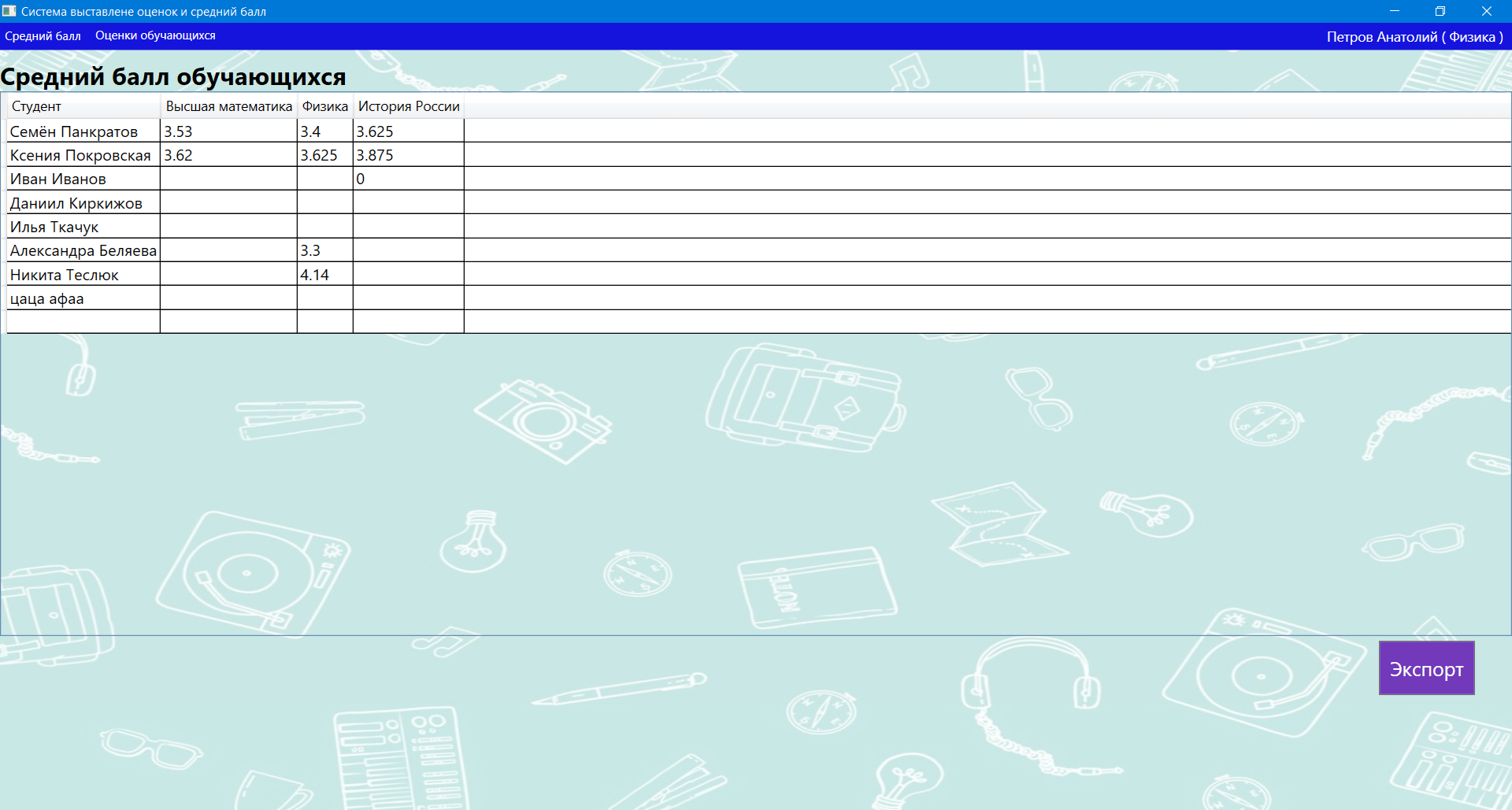


Рисунок 12 Модуль «Средний балл»

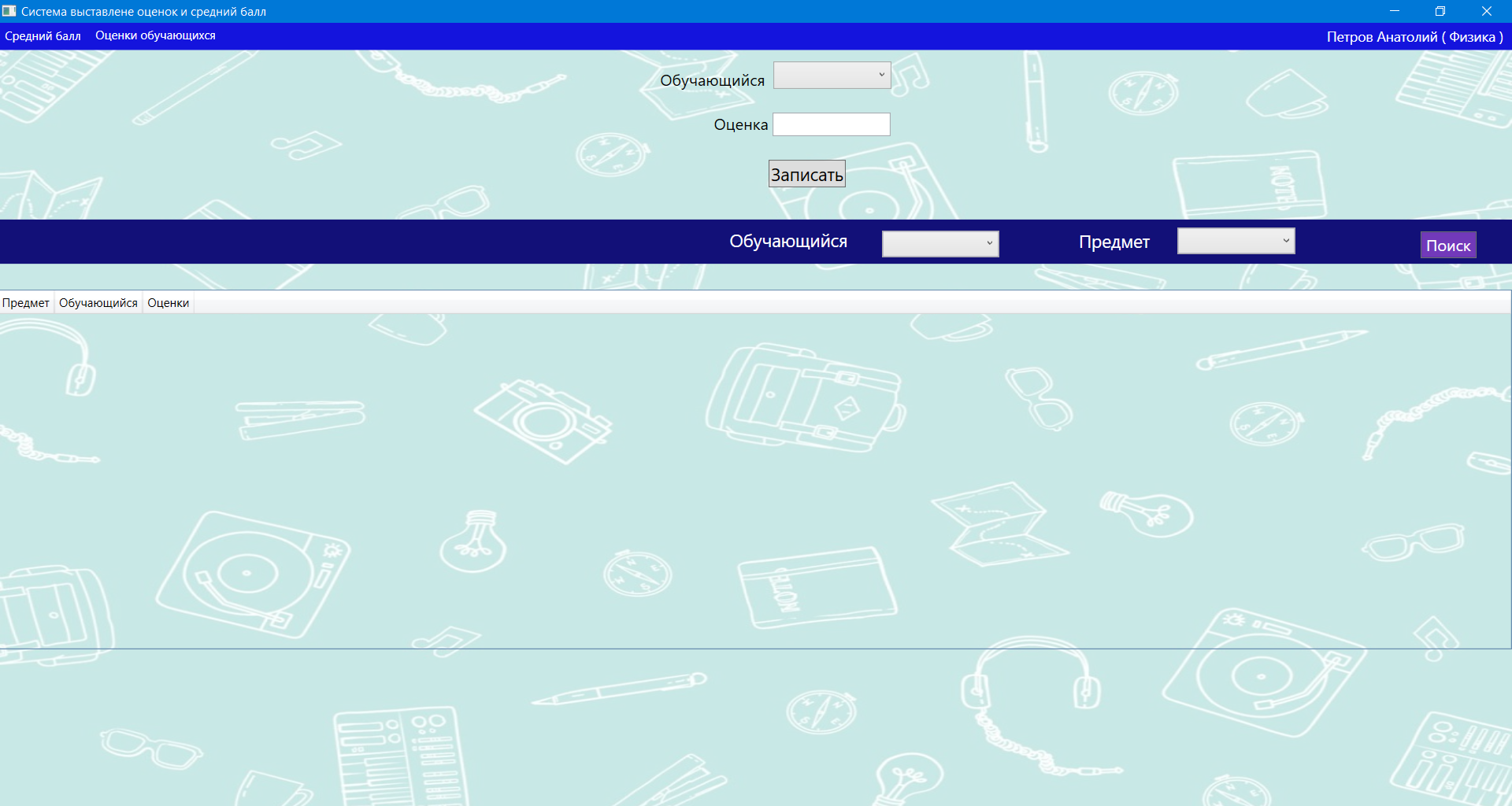


Рисунок 13 Модуль «Оценки обучающихся»

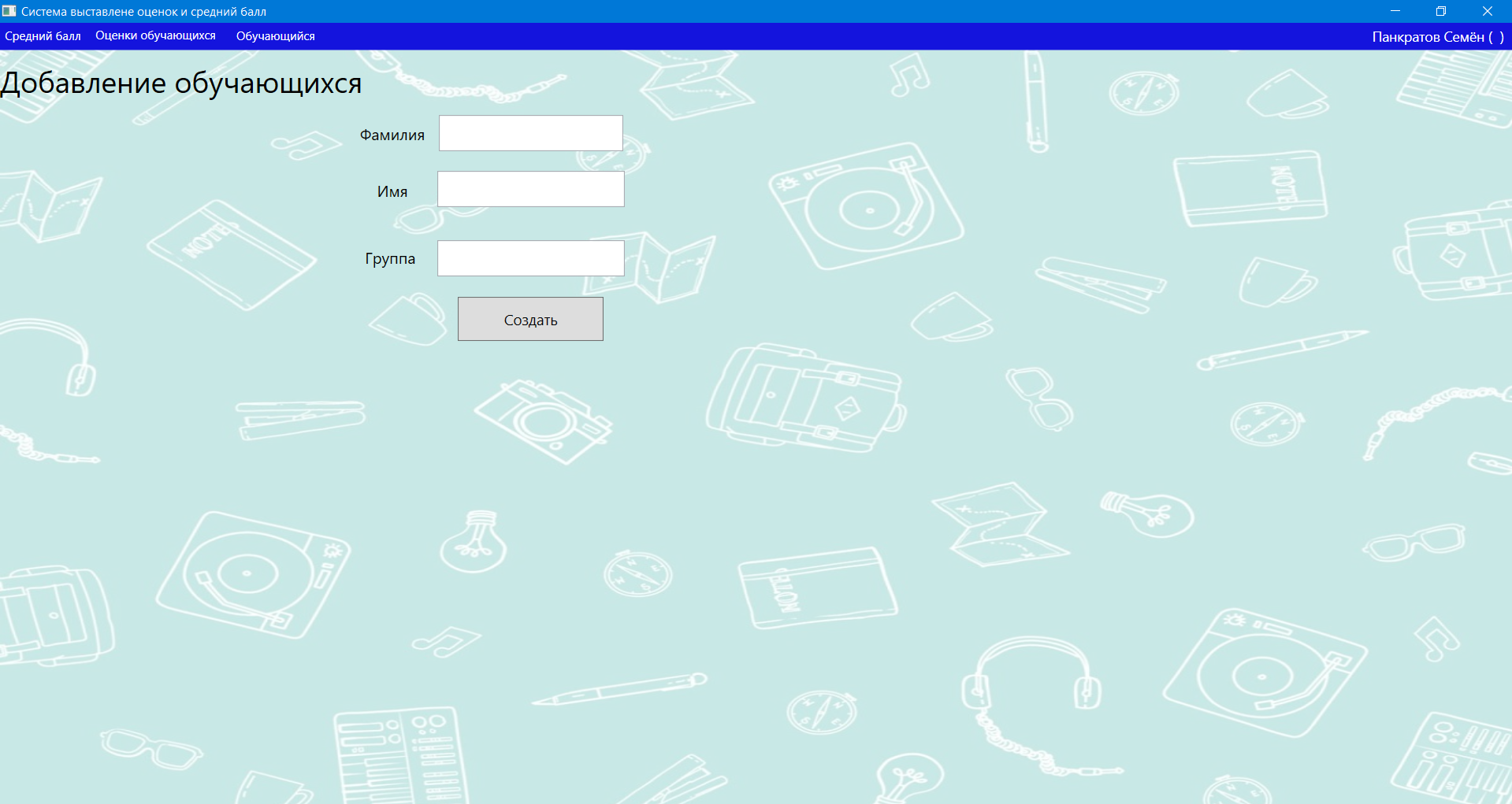


Рисунок 14 Модуль "Обучающийся"

## Тестирование приложения

**Тестирование программного обеспечения**—проверка соответствия реальных и ожидаемых результатов поведения программы, проводимая на конечном наборе тестов, выбранном определённым образом.

**Модульное тестирование** - проверяет функциональность конкретного куска кода, обычно по одной функции за раз.

**Нагрузочное тестирование -** подвид тестирования производительности, сбор показателей и определение производительности и времени отклика программно-технической системы или устройства в ответ на внешний запрос с целью установления соответствия требованиям, предъявляемым к данной системе.

**Цель:** Целью тестирования программного обеспечения, является выявление багов в приложении для последующего их устранения.

## Выбор и обоснование методики проведения тестирования

Тестирование приложения будет проходить с помощью следующих видов тестирования:

* Модульное тестирование
* Нагрузочное тестирование
* Ручное тестирование

Данные виды тестов позволяют протестировать отдельные модули приложения. Общий тест на работу приложения под нагрузкой, а также самостоятельное выявление несоответствий требованиям к приложению.

## Проведение комплексного тестирования приложения

Приложение протестировано модульным и нагрузочным тестами.

Создан unit-тест, который тестирует класс авторизации. Все тесты завершились с пометкой **«Успешно»**

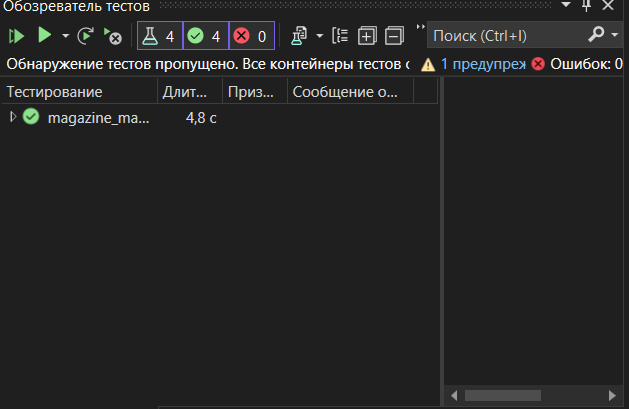


Рисунок 15 Результат Unit теста

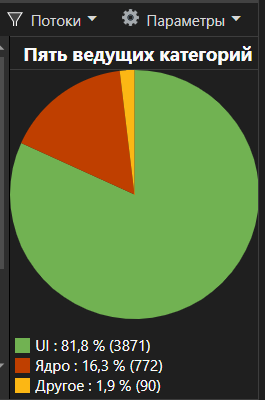


Рисунок 16 Результат нагрузочного тестирования

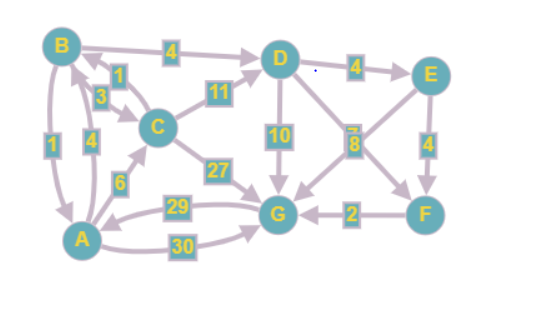
## Документирование результатов тестирования

|  |  |
| --- | --- |
| Описание тестируемого модуля | Кол-во тестов |
| Модуль **«Авторизация»** - данный модуль выполняет проверку авторизации пользователя на вход метода принимается логин и пароль. Если данные авторизации правильные метод возвращает “OK” в противном случае “No” | 4 теста |
| Нагрузочное тестирование | 1 тест |

# МОДЕЛИРОВАНИЕ В ПРОГРАММНЫХ СИСТЕМАХ

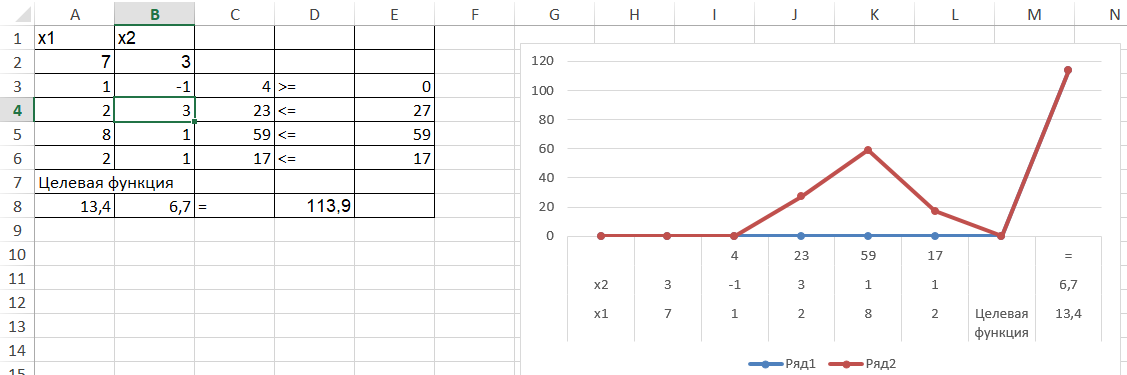
## Нахождение кратчайших путей на графе.

Найден кратчайший путь на графе. Граф построен с помощью сервиса: www.graphonline.ru



С помощью анализа графов было выявлено 12 маршрутов удовлетворяющие условию поставленной задачи.

## Решение задачи об оптимальных ресурсах

Решил задание в соответствии с индивидуальным заданием.

# ЗАКЛЮЧЕНИЕ

В результате прохождения учебной практики выполнены следующие задачи.

Разработаны требования к программному модулю.

Разработано техническое задание

Спроектированы диаграммы UML

* Разработан графический интерфейс приложения

Разработана логику приложения

Проведено тестирование приложения

В ходе прохождения учебной практики успешно освоена практическая подготовка по модулю ПМ 02.

## Приложение А

Министерство образования по Иркутской области

Государственное бюджетное профессиональное образовательное учреждение

Иркутской области

«Ангарский промышленно-экономический техникум»

СОГЛАСОВАНО

Фурман К.М

26.11.2022

**Техническое задание**

на разработку программного модуля

для информационной системы

«Журнал оценок»

ИСПОЛНИТЕЛЬ

Студент группы ИСП-2

Панкратов С.Н.

26.11.2022

г. Ангарск

2022г