МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Программное обеспечение информационных технологий»

РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ

Курсовая работа

по дисциплине «Основы алгоритмизации и программирования»

КР.1-53 01 02.10031796.18.ПЗ

Исполнитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бубнов Ярослав Николаевич

(подпись) АСОИ-201

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кашпар Александр Иванович

(подпись)

Дата допуска к защите \_\_\_\_\_\_\_\_\_\_\_\_\_

Дата защиты \_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_

Могилев 2022

МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Факультет Электротехнический

«Утверждаю»

Заведующий кафедрой\_\_\_\_\_\_\_\_\_\_

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

З А Д А Н И Е

на курсовую работу по дисциплине «Основы алгоритмизации и

программирования»

студенту Бубнова Я.Н гр. АСОИ-201

1. Тема курсовой работы: Разработка программы планирования факультативных учебных дисциплин для студентов

2. Срок сдачи законченной работы: 15.05.2022

3. Исходные данные к курсовой работе:

Сведения об пользователях информацию следующего вида: Ф.И.О. студента, номер группы, средний балл успеваемости, пять возможных для факультативного посещения дисциплин.

4. Перечень подлежащих разработке вопросов:

Введение

1 Техническое задание

1.1 Анализ предметной области

1.2 Структура программы

2 Описание программы

3 Руководство оператора

4 Тестирование

Заключение

Список использованных источников

5. На проверку предоставляются пояснительная записка, исходные тексты

программ и исполняемые файлы на электронном носителе.

Руководитель курсовой работы /Кашпар И.А./

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Бубнов Я.Н. /

Содержание

[Введение 3](#_Toc8310767)

1 Техническое задание 4

[2 Описание программы 7](#_Toc8310768)

[2.1 Общие сведения 7](#_Toc8310769)

[2.2 Функциональное назначение 7](#_Toc8310772)

[2.3 Описание логической структуры 7](#_Toc8310774)

[2.4 Используемые технические средства 13](#_Toc8311032)

2.5 Вызов и загрузка……………………………………………………….…...………13

2.6 Входные данные.…… …………………………………….………….……………13

2.7Выходные данные……………………..………………………..………..…………14

[3 Руководство оператора 15](#_Toc8311033)

[3.1 Назначение программы 15](#_Toc8311034)

[3.2 Условия выполнения программы 15](#_Toc8311036)

[3.3 Выполнение программы 15](#_Toc8311037)

4 Тестирование …………………………………………………………………………..21

Заключение [25](#_Toc8311038)

[Список литературы](#_Toc8311038) 26

[Приложения 2](#_Toc514187279)7

[ПРИЛОЖЕНИЕ А 2](#_Toc514187280)7

Введение

Разработка программы планирования факультативных учебных дисциплин для студентов. Данная программа в результате реализации должна помочь в планировании факультативных учебных дисциплин.

Так же, в процессе разработки данной программы необходимо создать систему авторизации, модули администратора и пользователя, предусмотреть действия пользователя, которые могут стать причиной аварийного завершения программы.

Основная часть пояснительной записки включает:

* Техническое задание – в данном разделе приведены развёрнутое описание к программе, структура меню;
* Описание программы – данный раздел содержит общие сведения о программе;
* Руководство оператора – в данном разделе указываются наименование программы и основные положения по её эксплуатации;
* Тестирование – в данном разделе приведены тесты для каждой возможной ситуации (вход учителя, вход ученика);
* Дано заключение и приведён список использованных источников;
* Приложения – данный раздел содержит код различных блоков программы.

Таким образом, данная работа позволит не только получить теоретические знания, но и освоить практические навыки программирования на языке С#.

# 1 Техническое задание

Тема курсовой работы «Разработка программного модуля банковского приложения».

Курсовая работа разработана на языке программирования C# на базе WPF с подключенной базой данной SQLite.

Система содержит базу данных с различными таблицами пользователей, и факультативных занятий. Также практически все таблицы можно изменять через окно администратора.

На рисунке 1 приводится графическое описание логической структуры программы.

При запуске программы первым открывается окно MainWindow у пользователя и у админа. Далее, в зависимости от введённых пароля, вместо роли программа разделена на 2 части, пользователя и админа.

При переходе в окно AdminWind, можно добавить пользователя, можно удалить пользователя, просмотр слушателей определенных, просмотр списка студентов, поиск студентов по успеваемости, ФИО и группе, создание отчета по слушателям предметов.

При переходе в окно AddUser, нужно вписать ФИО, группу, средний балл, логин, пароль и добавить его в базу данных нажатием на кнопку. После этого вернется окно администратора.

При переходе в окно DeleteUser, нужно вписать ФИО и удалить его из базы данных нажатием на кнопку. После этого вернется окно администратора.

При переходе в окно ShowTable, выводятся данные из базы, согласно указанному параметру. После этого вернется окно администратора.

При переходе в окно Block user, нужно вписать имя и логин и добавить значение блокировки в базу данных нажатием на кнопку. После этого вернется окно администратора.

При переходе в окно FindWindow, выбрать один из параметров поиска и ввести строку для поиска и начать поиск нажатием на кнопку, после чего выведется таблица результатов.

При переходе в окно LessonsListWind, нужно выбрать один из учебных предметов нажатием на кнопку, после чего выведется таблица с информацией об этом предмет. После этого вернется окно администратора.

При переходе в окно MakeReport, выбрать один из параметров, по какому предмету делать отчет, и создать его нажатием на кнопку. После этого вернется окно администратора.

При переходе в окно UserMenu, можно выбрать функционал юзера, список учащихся, выбор предметов, поиск преметов.

При переходе в окно ChooseLessons, нужно выбрать предметы для факультативных занятий и после нажать кнопку для сохранения.

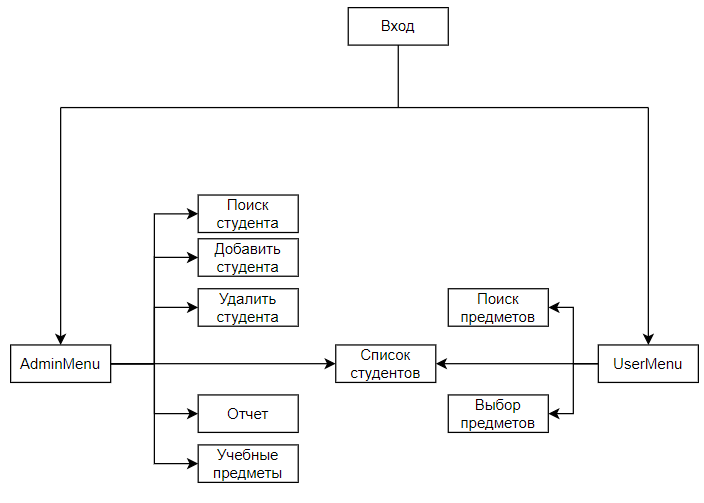


Рисунок 1 – Графическое представление структуры программы

2 Описание программы

2.1 Общие сведения

Данная программа планирования факультативных учебных дисциплин для студентов. Данная программа реализована на платформе .NET с использованием технологии WPF. Программа написана на языке C# с использованием компилятора .NET Framework.

Автор программы: Бубнов Я.Н., дата создания 26.03.2021, версия программы: 1.0.

2.2 Функциональное назначение

Программа предназначена для планирования факультативных учебных дисциплин для студентов. Программа прошла тестирования на студентов БРУ группы АСОИ-201.

2.3 Описание логической структуры

Программный модуль состоит из 9 основных окон.

AddUser – окно добавления студентов, AdminWind– главное окно админа, DeleteUser– окно удаления студента, FindWindow– окно поиска студентов, LessonsListWind– окно выбора предмета, MainWindow– окно входа, MakeReport– окно создания текстового отчета, ShowTable– окно вывода таблиц, UserMenu– окно главное окно студента.

На рисунке 2 представлено графическое представление схемы классов.

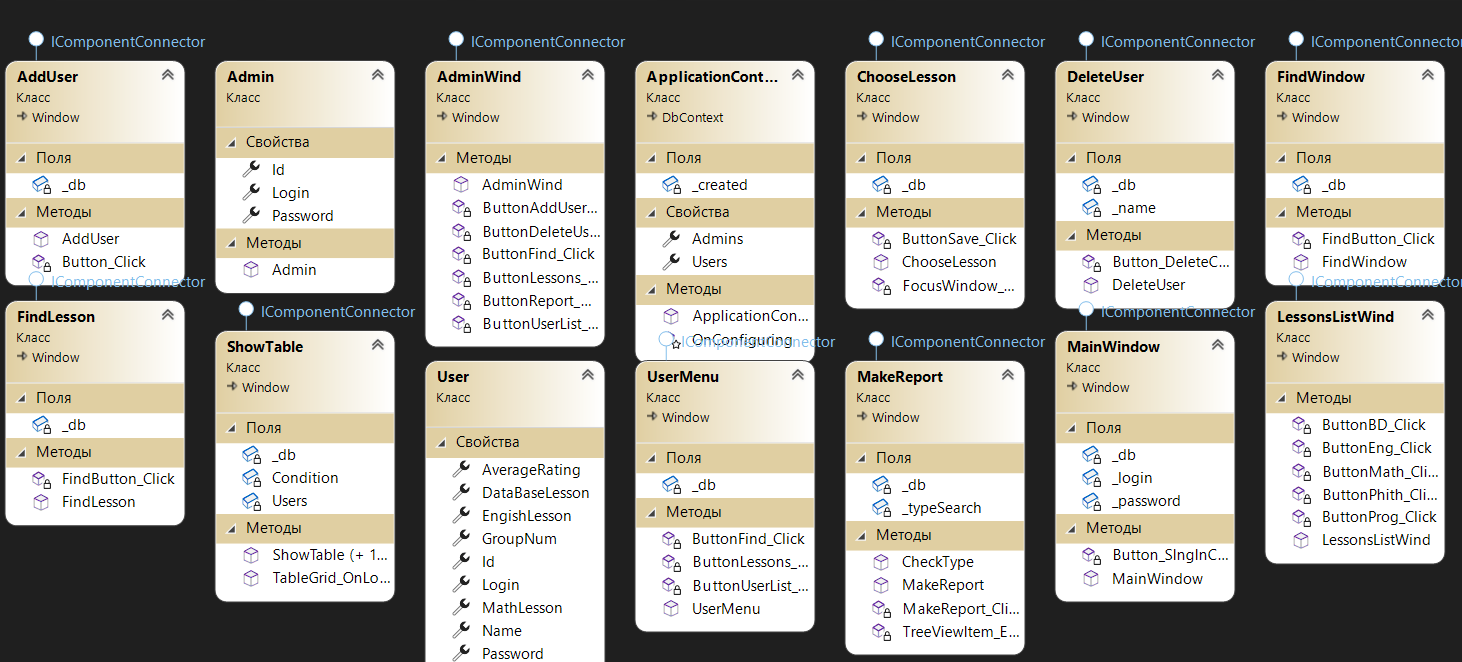


Рисунок 2 – Схема классов

Таблица 1 – Описание методов, применённых при реализации элементов управления

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя формы | Имя элемента управления | Метод | Краткое описание | |
| 1 | 2 | 3 | 4 | |
| AddUser | AddStudent | Button\_Click(object sender, RoutedEventArgs e) | Добавление студента | |
| AdminWind | UserList | ButtonUserList\_Click(object sender, RoutedEventArgs e) | Вывод списка студентов | |
| DeleteUser | ButtonDeleteUser\_Click(object sender, RoutedEventArgs e) | Вызов окна удаление студента | |
| FindUser | ButtonFind\_Click(object sender, RoutedEventArgs e) | Поиск студентов | |
| Lessons | ButtonLessons\_Click(object sender, RoutedEventArgs e) | Вывод окна с уроками | |
| Report | ButtonReport\_Click(object sender, RoutedEventArgs e) | Вызов окна создание отчета | |
| AddUser | ButtonAddUser\_Click(object sender, RoutedEventArgs e) | Вызов окна дабавление студента | |
| DeleteUser | DeleteButton | Button\_DeleteClick(object sender, RoutedEventArgs e) | Удаление студента | |
| FindWindow | FindButton | FindButton\_Click(object sender, RoutedEventArgs e) | Поиск студента | |
| LessonsListWind | ButtonMath | ButtonMath\_Click(object sender, RoutedEventArgs e) | Вызов таблицы по предмету математика | |
|  | ButtonPhith | ButtonPhith\_Click(object sender, RoutedEventArgs e) | Вызов таблицы по предмету физика | |
| ButtonEng | ButtonEng\_Click(object sender, RoutedEventArgs e) | Вызов таблицы по предмету английский | |
| ButtonProg | ButtonProg\_Click(object sender, RoutedEventArgs e) | Вызов таблицы по предмету программирование | |
| ButtonBD | ButtonBD\_Click(object sender, RoutedEventArgs e) | Вызов таблицы по предмету программирование | |
| Продолжение таблицы 1 | | | | |
| 1 | 2 | 3 | 4 | |
| MainWindow | Button\_SIngInClicks | Button\_SIngInClick(object sender, RoutedEventArgs e) | | Вход в систему |
| MakeReport | MakeReport | MakeReport\_Click(object sender, RoutedEventArgs e) | | Создание отчета |
| ShowTable |  | TableGrid\_OnLoaded(object sender, RoutedEventArgs e) | | Загрузка таблицы |
| ChooseLesson | SaveButton | ButtonSave\_Click(object sender, RoutedEventArgs e) | | Сохраняет выбранные предметы |
| UserMenu | LessonsButton | ButtonLessons\_Click(object sender, RoutedEventArgs e) | | Открытие окна выбора предметов |
| FindButton | ButtonFind\_Click(object sender, RoutedEventArgs e) | | Открытие окна поиска |
| UserList | ButtonUserList\_Click(object sender, RoutedEventArgs e) | | Открытия списка студентов |
| LessonsListWind | FindButton | FindButton\_Click(object sender, RoutedEventArgs e) | | Поиск предметов |

2.4 Используемые технические средства

Программа эксплуатируется на персональном ноутбуке Lenovo IdeaPad L340. Для работы в диалоговом режиме используется экран дисплея, клавиатура и манипулятор типа «мышь». Входные данные хранятся на жестком диске. Программа работает под управлением ОC семейства Windows версии 7 и выше.

2.5 Вызов и загрузка

Вызов программы осуществляется с помощью двойного щелчка левой кнопкой мыши по ярлыку программы, предварительно перейдя по пути его расположения, либо с помощью ярлыка, который находится на рабочем столе ПК. Так же запуск программы может осуществляться с помощью командной строки.

2.6 Входные данные

Входными данными являются логин, пароль и ФИО студента, группа и его успеваемость.

Таблица 2 – Описание входных данных

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Обозначение | Тип данных | Диапазон  значений |
| ФИО студента | Name | string | Ограничений  нет |
| Группа студента | GroupeName | string | Ограничений  нет |
| Логин | Login | string | Ограничений  нет |
| Пароль | Password | string | Ограничений  нет |
| Подписка на математику | MathLesson | int | 0..1 |
| Подписка на физику | PhysicsLesson | int | 0..1 |
| Подписка на английский | EngishLesson | int | 0..1 |
| Подписка на программирование | ProgramLesson | int | 0..1 |
| Подписка на базы данных | DataBaseLesson | int | 0..1 |
| Средний бал | AverageRating | double | 0..10 |

2.7 Выходные данные

Выходными данными будут: отсортированная таблица студентов, текстовые отчеты о слушателях предметов.

3 Руководство оператора

3.1 Назначение программы

Программа предназначена для планирования факультативных учебных дисциплин для студентов. В ходе использования программы, можно подписать на определенные предметы, и отслеживать это со стороны админа.

3.2 Условия выполнения программы

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять следующим требованиям: температура от 15 до 24 °С, влажность воздуха 45–80%и т.д. В состав технических средств должен входить IBM-совместимый персональный компьютер (ПЭВМ) или Mac, включающий в себя:

* Процессор не хуже Intel Celeron 4205U 1800 МГц;
* Объем оперативной памяти не менее 2 ГБ;
* Жесткий диск объемом не менее 120 ГБ;
* Видеопроцессор не хуже Intel HD Graphics 500;
* Сетеваякарта Intel PRO-1000;
* и т.д.

3.3 Выполнение программы

Загрузка и запуск программы осуществляется несколькими способами. В случае успешного запуска, как пользователь, программы на рабочем столе будет отображено стартовое окно, которое изображено на рисунке 2.

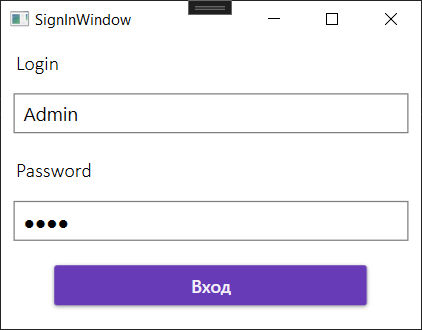


Рисунок 2 – Окно входа

При нажатии кнопки вход, при правильно введенных данных, открывается окно админа или юзера, в зависимости от логина, изображенное на рисунке 3.

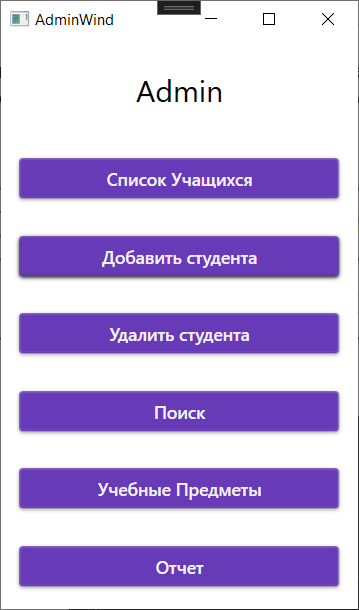


Рисунок 3 – Окно админа

При нажатии кнопки список учащихся, открывается таблица, изображенное на рисунке 4.

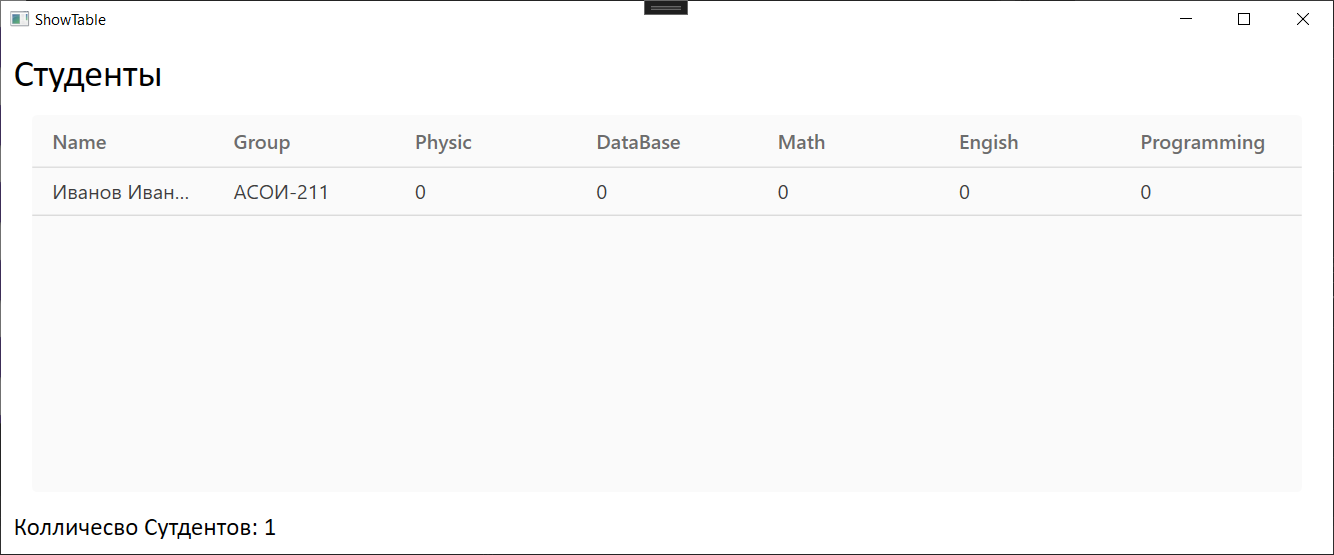


Рисунок 4 – Окно списка студентов

При нажатии кнопки добавить студента, открывается окно, изображенное на рисунке 5.

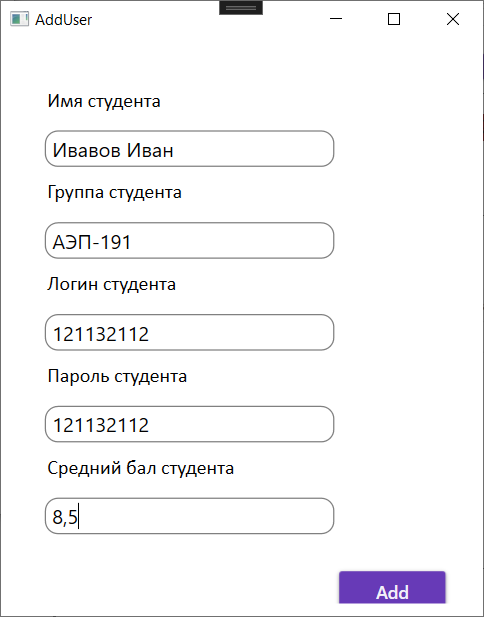


Рисунок 5 – Окно добавления студента

При нажатии кнопки удалить студента, открывается окно, изображенное на рисунке 6.

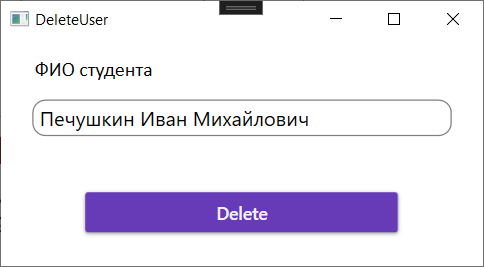


Рисунок 6 – Окно удаления

При нажатии кнопку поиск, открывается окно, изображенное на рисунке 7.

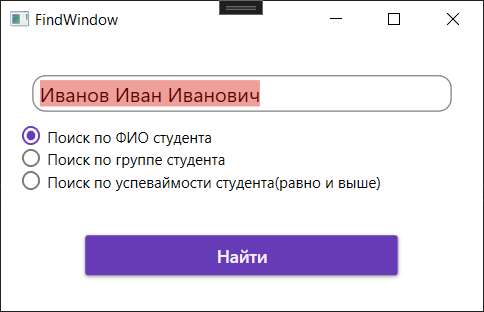


Рисунок 7 – Окно поиска студента

При нажатии кнопку найти и выбора параметра, открывается таблица, изображенное на рисунке 7.

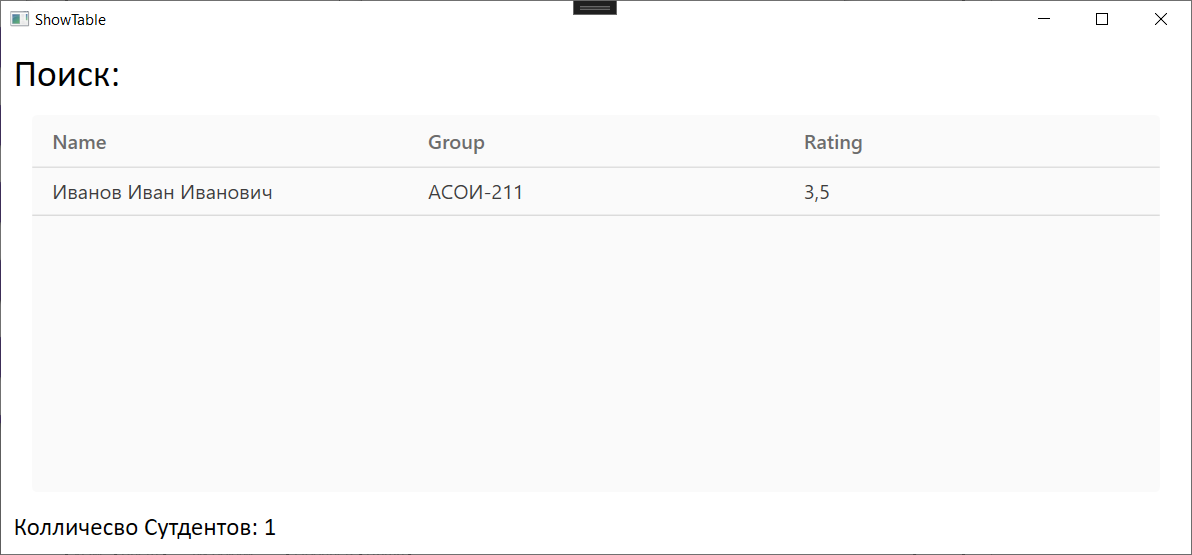


Рисунок 7 – Окно списка найденных студентов

При нажатии кнопку Учебные предметы и выбора, открывается окно, изображенное на рисунке 8.

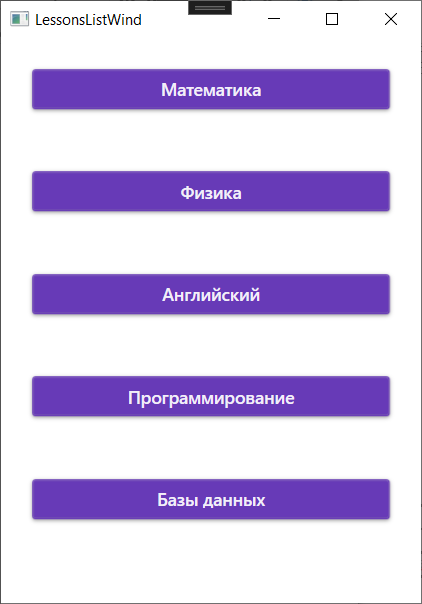


Рисунок 8– Окно выбора предмета

При нажатии на одну из кнопок, открывается окно для этого предмета, изображенное на рисунке 9.

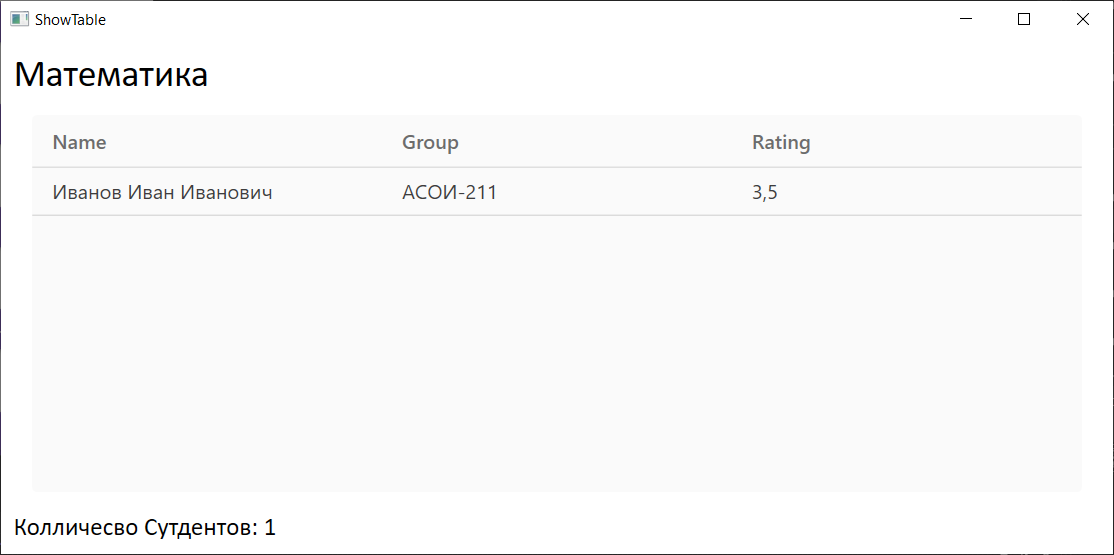


Рисунок 9– Окно предмета

При нажатии кнопку Отчет, открывается окно, изображенное на де 10.

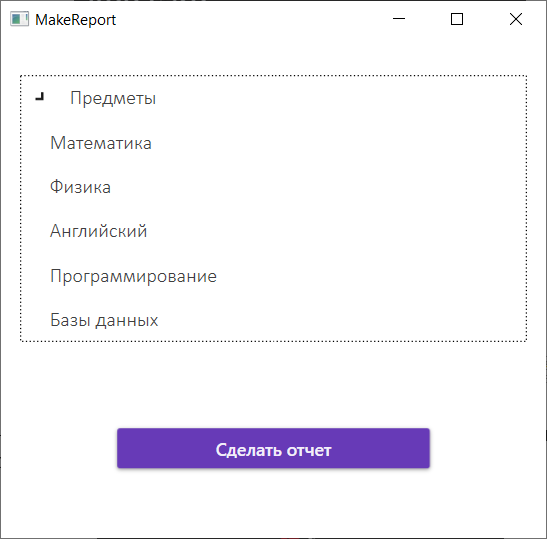


Рисунок 10– Окно создания отчета

При входе как студент, открывается окно, изображенное на рисунке 11.

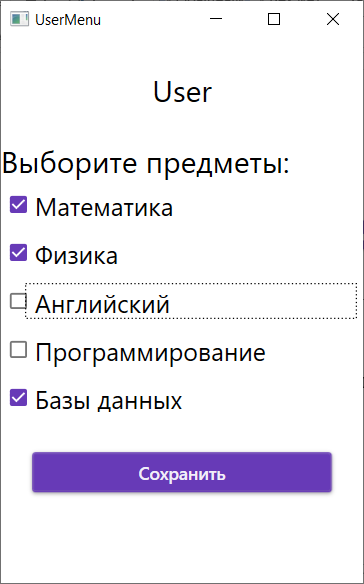


Рисунок 11– Окно меню студента

В случае не корректного ввода данных программ предупредит надписью на текстовым полем, изображенное на рисунке 12.

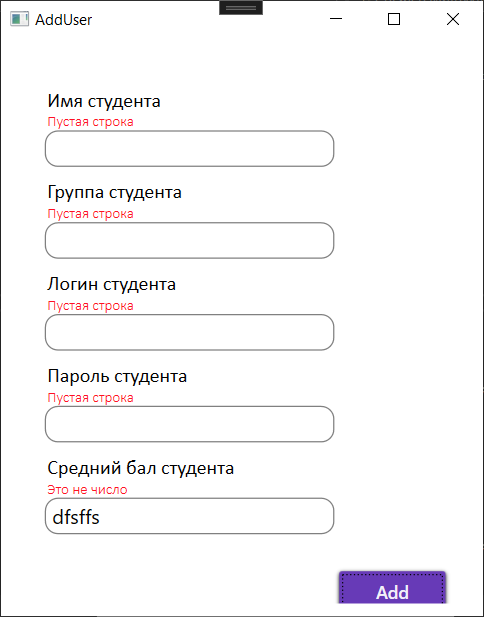


Рисунок 12– Окно c примером оповещение об ошибке

# 4 Тестирование

Для проверки корректности работы программы были приготовлены тесты, которые содержали различные варианты продолжения работы программы.

Таблица 3– Журнал тестирования программного продукта

|  |  |  |  |
| --- | --- | --- | --- |
| Объект | Команда пользователя | Ожидаемая реакция | Отметка об отклике системы |
| Кнопка «Вход» | Нажатие | Появление окна одного из модулей | Авторизация осуществлена успешно |
| Кнопка «Список учащихся» | Нажатие | Появление списка студентов | Список успешно появилось |
| Кнопка «Добавление студента» | Нажатие | Появление окна добавления студента | Окно успешно появилось |
| Кнопка «Удалить студента» | Нажатие | Появление окна удаления студента | Окно успешно появилось |
| Кнопка «Поиск» | Нажатие | Появление окна поиска студента | Окно успешно появилось |
| Кнопка «Учебные предметы » | Нажатие | Появление окна со списком предметов | Окно успешно появилось |
| Кнопка «Add» | Нажатие | Добавление студента | Студент успешно добавлен |
| Кнопка «Delete» | Нажатие | Удаления студента | Студент успешно удален |
| Кнопка «Найти» | Нажатие | Появление списка студентов | Список успешно появилось |
| Кнопка «Математика» | Нажатие | Появление списка студентов подписанных на этот предмет | Список успешно появилось |
| Кнопка «Физика» | Нажатие | Появление списка студентов подписанных на этот предмет | Список успешно появилось |
| Кнопка «Английский» | Нажатие | Появление списка студентов подписанных на этот предмет | Список успешно появилось |
| Кнопка «Программирование» | Нажатие | Появление списка студентов подписанных на этот предмет | Список успешно появилось |

Продолжение таблицы 3

|  |  |  |  |
| --- | --- | --- | --- |
| Объект | Команда пользователя | Ожидаемая реакция | Отметка об отклике системы |
| Кнопка «Базы данных» | Нажатие | Появление списка студентов подписанных на этот предмет | Список успешно появилось |
| Кнопка «Сделать отчет» | Нажатие | Создание отчета | Отчет успешно создан |
| Кнопка «Сохранить» | Нажатие | Сохранения изменений подписок студента | Изменения успешно сохраниилсь |

# Заключение

В ходе выполнения курсовой работы была разработана программа для планирования факультативных учебных дисциплин для студентов.

При создании программного продукта использовались программный продукт от Microsoft (MicrosoftVisualStudio 2022), язык программирования C# и база данных SQLite.

Было произведено тестирование разработанного программного модуля при различных вариантах входных данных, предусмотрен поиск ошибок при вводе неверных данных. Результаты тестирования показали, что программа работает корректно. Все пункты курсового задания были выполнены, разработанный программный продукт позволяет составить и просмотреть расписание.

Список литературы

1. WPF4 Подробное руководство / Адам Натан - СПб.: Символ-Плюс, 2011. - 880 с
2. Введение в системы баз данных / К. Дж. Дейт - Пер. с англ. — М.: Издательский дом "Вильяме", 2005. — 1328 с
3. Как программировать на Visual C# 2012 / Х. Дейтел. – СПб. : Питер. 2014. 864 с.
4. Троелсен, Э. Язык программирования С# 4.0 и платформа .NET 4.0 : пер. с англ. / Э. Троелсен. – М. : Вильямс, 2011. – 1392 с.
5. Фленов, М. Библия C# / М. Фленов. – СПб.: БХВ-Петербург, 2013. — 560 с.
6. Шилдт, Г. С#. Учебный курс / Г. Шилдт. – СПб.: Питер, 2003. – 512 с.: ил.
7. Шилдт, Г. C# 4.0. Полное руководство : пер. с англ. / Г. Шилдт. – М. : Вильямс, 2011. – 1056 с.
8. Visual C# 2010. Полный курс / К.Уотсон [и др.]. – М. : Вильямс, 2011. – 955 с.

Приложения

ПРИЛОЖЕНИЕ А

(обязательное)

Код программы

using System.Linq;

using System.Windows;

namespace Cursach

{

/// <summary>

/// Добавление студента

/// </summary>

public partial class AddUser : Window

{

private readonly ApplicationContext \_db;

public AddUser()

{

InitializeComponent();

\_db = new ApplicationContext();

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

NameError.Text = "";

GroupeError.Text = "";

LoginError.Text = "";

PasswordError.Text = "";

RatingError.Text = "";

double rating = 0;

if (!double.TryParse(Rating.Text, out rating))

{

RatingError.Text = "Это не число";

return;

}

if (\_db.Users.ToList().FindAll(t => t.Login.Equals(Login.Text)).Count != 0)

{

LoginError.Text = "Логин занят";

return;

}

var user = new User(Name.Text, Password.Text, Login.Text, GroupeName.Text, Rating.Text);

\_db.Users.Add(user);

\_db.SaveChanges();

Close();

}

}

}

namespace Cursach

{

public class Admin

{

public int Id { get; set; }

public string Password { get; set; }

public string Login { get; set; }

public Admin()

{

}

}

}

using System.Windows;

namespace Cursach

{

/// <summary>

/// Меню админа

/// </summary>

public partial class AdminWind : Window

{

public AdminWind()

{

InitializeComponent();

}

private void ButtonUserList\_Click(object sender, RoutedEventArgs e)

{

var showUsersList = new ShowTable("Студенты");

showUsersList.Owner = this;

showUsersList.Show();

}

private void ButtonAddUser\_Click(object sender, RoutedEventArgs e)

{

var wind = new AddUser();

wind.Owner = this;

wind.Show();

}

private void ButtonDeleteUser\_Click(object sender, RoutedEventArgs e)

{

var wind = new DeleteUser();

wind.Owner = this;

wind.Show();

}

private void ButtonFind\_Click(object sender, RoutedEventArgs e)

{

var wind = new FindWindow();

wind.Owner = this;

wind.Show();

}

private void ButtonLessons\_Click(object sender, RoutedEventArgs e)

{

var showUsersList = new LessonsListWind();

showUsersList.Owner = this;

showUsersList.Show();

}

private void ButtonReport\_Click(object sender, RoutedEventArgs e)

{

var showUsersList = new MakeReport();

showUsersList.Owner = this;

showUsersList.Show();

}

}

}

using Microsoft.EntityFrameworkCore;

namespace Cursach

{

//файл подключения к базе данных

public class ApplicationContext : DbContext

{

public DbSet<User> Users { get; set; }

private static bool \_created = false;

public ApplicationContext()

{

if (!\_created)

{

\_created = true;

Database.EnsureCreated();

}

}

protected override void OnConfiguring(DbContextOptionsBuilder optionbuilder)

{

optionbuilder.UseSqlite(@"Data Source=D:\Cursach\Sample.db");

}

}

}

using System.Windows;

namespace Cursach

{

/// <summary>

/// Окно подписки на предметы

/// </summary>

public partial class ChooseLesson : Window

{

private readonly ApplicationContext \_db;

public ChooseLesson()

{

InitializeComponent();

\_db = new ApplicationContext();

}

private void FocusWindow\_OnLoaded(object sender, RoutedEventArgs e)

{

var user = \_db.Users.ToList().Find(t => t.Id.Equals(App.userNow));

if (user.MathLesson.Equals(1))

{

MathButton.IsChecked = true;

}

if (user.PhysicsLesson.Equals(1))

{

PhithButton.IsChecked = true;

}

if (user.EngishLesson.Equals(1))

{

EngButton.IsChecked = true;

}

if (user.ProgramLesson.Equals(1))

{

ProgButton.IsChecked = true;

}

if (user.DataBaseLesson.Equals(1))

{

DdButton.IsChecked = true;

}

}

private void ButtonSave\_Click(object sender, RoutedEventArgs e)

{

var user = \_db.Users.ToList().Find(t => t.Id.Equals(App.userNow));

if ((bool)MathButton.IsChecked)

{

user.MathLesson = 1;

}

else

{

user.MathLesson = 0;

}

if ((bool)PhithButton.IsChecked)

{

user.PhysicsLesson = 1;

}

else

{

user.PhysicsLesson = 0;

}

if ((bool)EngButton.IsChecked)

{

user.EngishLesson = 1;

}

else

{

user.EngishLesson = 0;

}

if ((bool)ProgButton.IsChecked)

{

user.ProgramLesson = 1;

}

else

{

user.ProgramLesson = 0;

}

if ((bool)DdButton.IsChecked)

{

user.DataBaseLesson = 1;

}

else

{

user.DataBaseLesson = 0;

}

\_db.SaveChanges();

}

}

}

using System.Linq;

using System.Windows;

namespace Cursach

{

/// <summary>

/// Окно поиска предметов

/// </summary>

public partial class FindLesson : Window

{

ApplicationContext \_db;

public FindLesson()

{

InitializeComponent();

\_db = new ApplicationContext();

}

private void FindButton\_Click(object sender, RoutedEventArgs e)

{

NameError.Text = "";

if (ByName.IsChecked.Equals(true))

{

if (NameTextBox.Text.Equals("Математика"))

{

var users = \_db.Users.ToList().FindAll(t => t.MathLesson.Equals(1));

var showTable = new ShowTable(users);

showTable.Show();

}

else if (NameTextBox.Text.Equals("Физика"))

{

var users = \_db.Users.ToList().FindAll(t => t.PhysicsLesson.Equals(1));

var showTable = new ShowTable(users);

showTable.Show();

}

else if (NameTextBox.Text.Equals("Английкий"))

{

var users = \_db.Users.ToList().FindAll(t => t.EngishLesson.Equals(1));

var showTable = new ShowTable(users);

showTable.Show();

}

else if (NameTextBox.Text.Equals("Программирование"))

{

var users = \_db.Users.ToList().FindAll(t => t.ProgramLesson.Equals(1));

var showTable = new ShowTable(users);

showTable.Show();

}

else if (NameTextBox.Text.Equals("Базы данных"))

{

var users = \_db.Users.ToList().FindAll(t => t.DataBaseLesson.Equals(1));

var showTable = new ShowTable(users);

showTable.Show();

}

else

{

NameError.Text = "Не корректный ввод";

}

}

if (ByAverRating.IsChecked.Equals(true))

{

double doubleRating = 0;

if (double.TryParse(NameTextBox.Text, out doubleRating) || doubleRating < 0)

{

var users = \_db.Users.ToList().FindAll(t => t.ProgramLesson.Equals(1));

if (users.Count != 0 && users.Average(t => t.AverageRating) >= doubleRating)

{

MessageBox.Show("Прогаммирование");

}

users = \_db.Users.ToList().FindAll(t => t.EngishLesson.Equals(1));

if (users.Count != 0 && users.Average(t => t.AverageRating) >= doubleRating)

{

MessageBox.Show("Английский");

}

users = \_db.Users.ToList().FindAll(t => t.PhysicsLesson.Equals(1));

if (users.Count != 0 && users.Average(t => t.AverageRating) >= doubleRating)

{

MessageBox.Show("Физика");

}

users = \_db.Users.ToList().FindAll(t => t.MathLesson.Equals(1));

if (users.Count != 0 && users.Average(t => t.AverageRating) >= doubleRating)

{

MessageBox.Show("Матеиатика");

}

users = \_db.Users.ToList().FindAll(t => t.DataBaseLesson.Equals(1));

if (users.Count != 0 && users.Average(t => t.AverageRating) >= doubleRating)

{

MessageBox.Show("Базы данныйх");

}

}

else

{

NameError.Text = "Неверное число";

}

}

if (ByCount.IsChecked.Equals(true))

{

int doubleCount = 0;

if (int.TryParse(NameTextBox.Text, out doubleCount) || doubleCount < 0)

{

if (\_db.Users.ToList().FindAll(t => t.ProgramLesson.Equals(1)).Count >= doubleCount)

{

MessageBox.Show("Програмирование");

}

if (\_db.Users.ToList().FindAll(t => t.EngishLesson.Equals(1)).Count >= doubleCount)

{

MessageBox.Show("Английский");

}

if (\_db.Users.ToList().FindAll(t => t.PhysicsLesson.Equals(1)).Count >= doubleCount)

{

MessageBox.Show("Физика");

}

if (\_db.Users.ToList().FindAll(t => t.MathLesson.Equals(1)).Count >= doubleCount)

{

MessageBox.Show("Матеиатика");

}

if (\_db.Users.ToList().FindAll(t => t.DataBaseLesson.Equals(1)).Count >= doubleCount)

{

MessageBox.Show("Базы данныйх");

}

}

else

{

NameError.Text = "Неверное число";

}

}

}

}

}

using System.Linq;

using System.Windows;

namespace Cursach

{

/// <summary>

/// Удаление студента

/// </summary>

public partial class DeleteUser : Window

{

private readonly ApplicationContext \_db;

private string \_name;

public DeleteUser()

{

InitializeComponent();

\_db = new ApplicationContext();

}

private void Button\_DeleteClick(object sender, RoutedEventArgs e)

{

\_name = NameTextBox.Text.Trim();

NameError.Text = "";

var user = \_db.Users.ToList().Find(t => t.Name.Equals(\_name));

if (user != null)

{

\_db.Users.Remove(user);

\_db.SaveChanges();

Close();

}

else

{

NameError.Text = "Нет такого студента";

}

}

}

}

using System.Collections.Generic;

using System.Linq;

using System.Windows;

namespace Cursach

{

/// <summary>

/// Окно поиска студента

/// </summary>

public partial class FindWindow : Window

{

ApplicationContext \_db;

public FindWindow()

{

InitializeComponent();

\_db = new ApplicationContext();

}

private void FindButton\_Click(object sender, RoutedEventArgs e)

{

NameError.Text = "";

if (ByName.IsChecked.Equals(true))

{

var users = \_db.Users.ToList().FindAll(t => t.Name.Equals(NameTextBox.Text));

var showTable = new ShowTable(users);

showTable.Show();

}

if (ByGroupe.IsChecked.Equals(true))

{

var users = \_db.Users.ToList().FindAll(t => t.GroupNum.Equals(NameTextBox.Text));

var showTable = new ShowTable(users);

showTable.Show();

}

if (ByRating.IsChecked.Equals(true))

{

List<User> users = new List<User>();

double doubleRating;

if (double.TryParse(NameTextBox.Text, out doubleRating))

{

foreach (User user in \_db.Users.ToList())

{

if (double.Parse(user.AverageRating) >= doubleRating)

{

users.Add(user);

}

}

var showTable = new ShowTable(users);

showTable.Show();

}

else

{

NameError.Text = "Это не число";

}

}

}

}

}

using System.Windows;

namespace Cursach

{

/// <summary>

/// окно выбора предмета

/// </summary>

public partial class LessonsListWind : Window

{

public LessonsListWind()

{

InitializeComponent();

}

private void ButtonMath\_Click(object sender, RoutedEventArgs e)

{

var showUsersList = new ShowTable("Математика");

showUsersList.Owner = this;

showUsersList.Show();

}

private void ButtonPhith\_Click(object sender, RoutedEventArgs e)

{

var showUsersList = new ShowTable("Физика");

showUsersList.Owner = this;

showUsersList.Show();

}

private void ButtonEng\_Click(object sender, RoutedEventArgs e)

{

var showUsersList = new ShowTable("Английский");

showUsersList.Owner = this;

showUsersList.Show();

}

private void ButtonProg\_Click(object sender, RoutedEventArgs e)

{

var showUsersList = new ShowTable("Программирование");

showUsersList.Owner = this;

showUsersList.Show();

}

private void ButtonBD\_Click(object sender, RoutedEventArgs e)

{

var showUsersList = new ShowTable("Базы данных");

showUsersList.Owner = this;

showUsersList.Show();

}

}

}

using System.Linq;

using System.Windows;

namespace Cursach

{

/// <summary>

/// Окно входа всистему

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

\_db = new ApplicationContext();

}

private readonly ApplicationContext \_db;

private string \_login;

private string \_password;

private void Button\_SIngInClick(object sender, RoutedEventArgs e)

{

\_login = LoginTextbox.Text;

\_password = PasswordTextbox.Password;

Admin admin = new Admin();

var users = \_db.Users.ToList();

var user = users.Find(item => item.Login.Equals(\_login));

LoginError.Text = "";

PasswordError.Text = "";

if (\_password.Length == 0) PasswordError.Text = "Blank field";

if (\_login == "Admin")

{

if (admin.Password.Equals(\_password))

{

AdminWind adminWind = new AdminWind();

adminWind.Show();

Close();

}

else

{

PasswordError.Text = "Wrong password";

}

}

else if (user == null)

{

LoginError.Text = "Unknown login";

}

else

{

if (user.Password.Equals(\_password))

{

App.userNow = user.Id;

var userMenu = new UserMenu();

userMenu.Show();

Close();

}

else

{

PasswordError.Text = "Wrong password";

}

}

}

}

}

using System;

using System.IO;

using System.Linq;

using System.Text;

using System.Windows;

using System.Windows.Controls;

namespace Cursach

{

/// <summary>

/// Окно созадния очета

/// </summary>

public partial class MakeReport : Window

{

private readonly ApplicationContext \_db;

private string \_typeSearch;

public MakeReport()

{

InitializeComponent();

\_db = new ApplicationContext();

}

public bool CheckType(string \_typeCard)

{

if (\_typeCard == null)

{

TypeError.Text = "Choose period";

return false;

}

return true;

}

private void TreeViewItem\_Expanded(object sender, RoutedEventArgs e)

{

var tvItem = (TreeViewItem)sender;

\_typeSearch = tvItem.Name;

}

private void MakeReport\_Click(object sender, RoutedEventArgs e)

{

var user = \_db.Users.ToList().Find(t => t.Id.Equals(App.userNow));

string path = @"D:\Cursach";

string textReport = "";

if (CheckType(\_typeSearch))

{

if (\_typeSearch == "Math")

{

var usersSelect = \_db.Users.ToList().FindAll(t => t.MathLesson.Equals(1));

textReport = "Математика";

foreach (var report in usersSelect)

{

textReport += $"\n{report.Name} {report.GroupNum} {report.AverageRating}";

}

DirectoryInfo dirInfo = new DirectoryInfo(path);

if (!dirInfo.Exists)

{

dirInfo.Create();

}

string dataNowString = DateTime.Now.ToLongTimeString().ToString();

using (FileStream fstream = new FileStream($@"{path}\report\_Математика.txt", FileMode.OpenOrCreate))

{

byte[] array = Encoding.Default.GetBytes(textReport);

fstream.Write(array, 0, array.Length);

}

}

else if (\_typeSearch == "Phith")

{

var usersSelect = \_db.Users.ToList().FindAll(t => t.MathLesson.Equals(1));

textReport = "Физика";

foreach (var report in usersSelect)

{

textReport += $"\n{report.Name} {report.GroupNum} {report.AverageRating}";

}

DirectoryInfo dirInfo = new DirectoryInfo(path);

if (!dirInfo.Exists)

{

dirInfo.Create();

}

string dataNowString = DateTime.Now.ToLongTimeString().ToString();

using (FileStream fstream = new FileStream($@"{path}\report\_Физика.txt", FileMode.OpenOrCreate))

{

byte[] array = Encoding.Default.GetBytes(textReport);

fstream.Write(array, 0, array.Length);

}

}

else if (\_typeSearch == "English")

{

var usersSelect = \_db.Users.ToList().FindAll(t => t.MathLesson.Equals(1));

textReport = "Английский";

foreach (var report in usersSelect)

{

textReport += $"\n{report.Name} {report.GroupNum} {report.AverageRating}";

}

DirectoryInfo dirInfo = new DirectoryInfo(path);

if (!dirInfo.Exists)

{

dirInfo.Create();

}

string dataNowString = DateTime.Now.ToLongTimeString().ToString();

using (FileStream fstream = new FileStream($@"{path}\report\_Английский.txt", FileMode.OpenOrCreate))

{

byte[] array = Encoding.Default.GetBytes(textReport);

fstream.Write(array, 0, array.Length);

}

}

else if (\_typeSearch == "Prog")

{

var usersSelect = \_db.Users.ToList().FindAll(t => t.MathLesson.Equals(1));

textReport = "Программирование";

foreach (var report in usersSelect)

{

textReport += $"\n{report.Name} {report.GroupNum} {report.AverageRating}";

}

DirectoryInfo dirInfo = new DirectoryInfo(path);

if (!dirInfo.Exists)

{

dirInfo.Create();

}

string dataNowString = DateTime.Now.ToLongTimeString().ToString();

using (FileStream fstream = new FileStream($@"{path}\report\_Программирование.txt", FileMode.OpenOrCreate))

{

byte[] array = Encoding.Default.GetBytes(textReport);

fstream.Write(array, 0, array.Length);

}

}

else if (\_typeSearch == "BD")

{

var usersSelect = \_db.Users.ToList().FindAll(t => t.MathLesson.Equals(1));

textReport = "Базы данных";

foreach (var report in usersSelect)

{

textReport += $"\n{report.Name} {report.GroupNum} {report.AverageRating}";

}

DirectoryInfo dirInfo = new DirectoryInfo(path);

if (!dirInfo.Exists)

{

dirInfo.Create();

}

string dataNowString = DateTime.Now.ToLongTimeString().ToString();

using (FileStream fstream = new FileStream($@"{path}\report\_БазыДанных.txt", FileMode.OpenOrCreate))

{

byte[] array = Encoding.Default.GetBytes(textReport);

fstream.Write(array, 0, array.Length);

}

}

Close();

}

}

}

}

using System.Collections.Generic;

using System.Linq;

using System.Windows;

namespace Cursach

{

/// <summary>

/// окно вывода таблиц

/// </summary>

public partial class ShowTable : Window

{

ApplicationContext \_db;

string Condition;

List<User> Users;

public ShowTable(string cond)

{

InitializeComponent();

\_db = new ApplicationContext();

Condition = cond;

}

public ShowTable(List<User> users)

{

InitializeComponent();

\_db = new ApplicationContext();

Condition = "Поиск:";

Users = users;

}

public void TableGrid\_OnLoaded(object sender, RoutedEventArgs e)

{

Cond.Text = Condition;

var users = \_db.Users.ToList();

if (Condition.Equals("Студенты"))

{

var usersSelect = users.Select(t => new

{

Name = t.Name,

Group = t.GroupNum,

Physic = t.PhysicsLesson,

DataBase = t.DataBaseLesson,

Math = t.MathLesson,

Engish = t.EngishLesson,

Programming = t.ProgramLesson,

}).ToList();

TableGrid.ItemsSource = usersSelect;

CountUser.Text += $"{usersSelect.Count}";

}

else if (Condition == "Математика")

{

var usersSelect = users.FindAll(t => t.MathLesson.Equals(1));

if (usersSelect.Count > 15)

{

usersSelect.OrderBy(t => (double.Parse(t.AverageRating)));

var t = usersSelect;

for (int i = 0; i < 15; i++)

{

usersSelect[i] = t[i];

}

}

var usersSelectGrid = usersSelect.Select(t => new

{

Name = t.Name,

Group = t.GroupNum,

Rating = t.AverageRating

}).ToList();

CountUser.Text += $"{usersSelectGrid.Count}";

TableGrid.ItemsSource = usersSelectGrid;

}

else if (Condition == "Физика")

{

var usersSelect = users.FindAll(t => t.PhysicsLesson.Equals(1));

if (usersSelect.Count > 15)

{

usersSelect.OrderBy(t => (double.Parse(t.AverageRating)));

var t = usersSelect;

for (int i = 0; i < 15; i++)

{

usersSelect[i] = t[i];

}

}

var usersSelectGrid = usersSelect.Select(t => new

{

Name = t.Name,

Group = t.GroupNum,

Rating = t.AverageRating

}).ToList();

CountUser.Text += $"{usersSelectGrid.Count}";

TableGrid.ItemsSource = usersSelectGrid;

}

else if (Condition == "Английский")

{

var usersSelect = users.FindAll(t => t.EngishLesson.Equals(1));

if (usersSelect.Count > 15)

{

usersSelect.OrderBy(t => (double.Parse(t.AverageRating)));

var t = usersSelect;

for (int i = 0; i < 15; i++)

{

usersSelect[i] = t[i];

}

}

var usersSelectGrid = usersSelect.Select(t => new

{

Name = t.Name,

Group = t.GroupNum,

Rating = t.AverageRating

}).ToList();

TableGrid.ItemsSource = usersSelectGrid;

CountUser.Text += $"{usersSelectGrid.Count}";

}

else if (Condition == "Программирование")

{

var usersSelect = users.FindAll(t => t.ProgramLesson.Equals(1));

if (usersSelect.Count > 15)

{

usersSelect.OrderBy(t => (double.Parse(t.AverageRating)));

var t = usersSelect;

for (int i = 0; i < 15; i++)

{

usersSelect[i] = t[i];

}

}

var usersSelectGrid = usersSelect.Select(t => new

{

Name = t.Name,

Group = t.GroupNum,

Rating = t.AverageRating

}).ToList();

TableGrid.ItemsSource = usersSelectGrid;

CountUser.Text += $"{usersSelectGrid.Count}";

}

else if (Condition == "Базы данных")

{

var usersSelect = users.FindAll(t => t.DataBaseLesson.Equals(1));

if (usersSelect.Count > 15)

{

usersSelect.OrderBy(t => (double.Parse(t.AverageRating)));

var t = usersSelect;

for (int i = 0; i < 15; i++)

{

usersSelect[i] = t[i];

}

}

var usersSelectGrid = usersSelect.Select(t => new

{

Name = t.Name,

Group = t.GroupNum,

Rating = t.AverageRating

}).ToList();

TableGrid.ItemsSource = usersSelectGrid;

CountUser.Text += $"{usersSelectGrid.Count}";

}

else if (Condition == "Поиск:")

{

var usersSelectGrid = Users.Select(t => new

{

Name = t.Name,

Group = t.GroupNum,

Rating = t.AverageRating

}).ToList();

TableGrid.ItemsSource = usersSelectGrid;

CountUser.Text += $"{usersSelectGrid.Count}";

}

}

}

}

namespace Cursach

{

public class User

{

public int Id { get; set; }

public string Name { get; set; }

public string Password { get; set; }

public string Login { get; set; }

public string GroupNum { get; set; }

public double AverageRating { get; set; }

public int MathLesson { get; set; }

public int PhysicsLesson { get; set; }

public int ProgramLesson { get; set; }

public int EngishLesson { get; set; }

public int DataBaseLesson { get; set; }

public User(string name, string pasword, string login, string groupNum, double averageRating)

{

Name = name;

Password = pasword;

Login = login;

GroupNum = groupNum;

AverageRating = averageRating;

}

public User()

{

}

}

}

using System.Linq;

using System.Windows;

namespace Cursach

{

/// <summary>

/// Меню юзера

/// </summary>

public partial class UserMenu : Window

{

private readonly ApplicationContext \_db;

public UserMenu()

{

InitializeComponent();

\_db = new ApplicationContext();

}

private void ButtonUserList\_Click(object sender, RoutedEventArgs e)

{

var showUsersList = new ShowTable("Студенты");

showUsersList.Owner = this;

showUsersList.Show();

}

private void ButtonFind\_Click(object sender, RoutedEventArgs e)

{

var wind = new FindLesson();

wind.Owner = this;

wind.Show();

}

private void ButtonLessons\_Click(object sender, RoutedEventArgs e)

{

var wind = new ChooseLesson();

wind.Owner = this;

wind.Show();

}

}

}

Удалить Примечание

Примечание

В строке подключения и в окне создания текстового файла путь к папке указан абсолютна

На своем компе нужно будет поменять путь на путь папки где лежит проект или куда хочешь сохранять + уточнить у кашпара нужно ли это менять на относительный путь