МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий

Направление специальности 1-40 01 01 10 Программное обеспечение информационных технологий (программирование интернет приложений)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА:**

по дисциплине «Объектно-ориентированные технологии программирования и стандарты проектирования»

Тема Программное средство «Студенты университета» (Student Hub)

Исполнитель

студент 2 курса группы 6 Истомин Илья Олегович

(Ф.И.О.)

Руководитель работы ассистент Северинчик Н. А.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

Председатель Северинчик Н. А.

(подпись)

Минск 2020

# **Содержание**

[Содержание 2](#_Toc41329224)

[ВВЕДЕНИЕ 3](#_Toc41329225)

[1 Аналитический обзор аналогов 4](#_Toc41329226)

[2 Разработка архитектуры проекта 5](#_Toc41329227)

[2.1 Обобщенная структура 5](#_Toc41329228)

[2.2 Проектирование логической структуры базы данных 6](#_Toc41329229)

[2.3 Структура проекта в виде схемы 7](#_Toc41329230)

[2.4 Пример использования 8](#_Toc41329231)

[3 Разработка архитектуры проекта 9](#_Toc41329232)

[3.1 Система для проектирования 9](#_Toc41329233)

[3.2 Схема и описание авторизации 9](#_Toc41329234)

[3.3 Работа с дизайном 9](#_Toc41329235)

[3.4 Подключение к БД 10](#_Toc41329236)

[4 Тестирование 11](#_Toc41329237)

[ЗАКЛЮЧЕНИЕ 16](#_Toc41329238)

[Список литературных источников 17](#_Toc41329239)

[ПРИЛОЖЕНИЕ А 18](#_Toc41329240)

[ПРИЛОЖЕНИЕ Б 19](#_Toc41329241)

[ПРИЛОЖЕНИЕ В 21](#_Toc41329242)

[ПРИЛОЖЕНИЕ Г 22](#_Toc41329243)

# **ВВЕДЕНИЕ**

В настоящее время роль высшего образования в Беларуси занимает одно из ключевых мест в создании конкурентоспособного поколения. Порой бывает трудно уследить за всеми процессами, которые происходят в университете. Для облегчения работы со студентами было создано программное средство «Student Hub».

В данной работе разработано ПО, которое позволяет студенту выполнить запрос на отработку, пересдачу, просматривать свою успеваемость. В университете не малую роль играет староста группы. Староста следит за работой своей группы и выставляет пропуски и оценки для поддержания порядка в учебном процессе.

Целью данной работы является создание приложения, которое работает с личными данными пользователя и упрощает работу с администратором университета. А также функционал приложения позволяет:

* Принять\отклонить запрос на отработку, пересдачу;
* Выполнять поисковые запросы;
* Просматривать текущую успеваемость студента;
* Выполнять отчеты;
* Генерировать электронную почту для студента, если ему это необходимо.

Структура приложения: приложение, написанное на С#. Оно устанавливается на ПК и представляет собой интерфейс.

ПК приложение для оптимальной работы требует:

* ОС Windows 10;
* установленный Microsoft SQL Server 2019;
* установленный .Net framework 4.7.2 или более поздней версии.

# **Аналитический обзор аналогов**

На сегодняшний день существует мало программ для работы со студентами. Системы программирования дают возможность удобно работать с базами данных.

Использование современных технологий – естественный выбор для создания приложений на современной технологической базе.

В качестве одного из примеров для анализа возьмем приложение популярное ПО для работы с информацией студентов «Microsoft Access» [1]. Данное приложение было разработано для персональных компьютеров. Имеет следующие возможности:

* Широкий набор инструментов для создания и работы с БД;
* Большое количество предустановленных шаблонов;
* Использование макросов;
* Поддержка аппаратного ускорения.

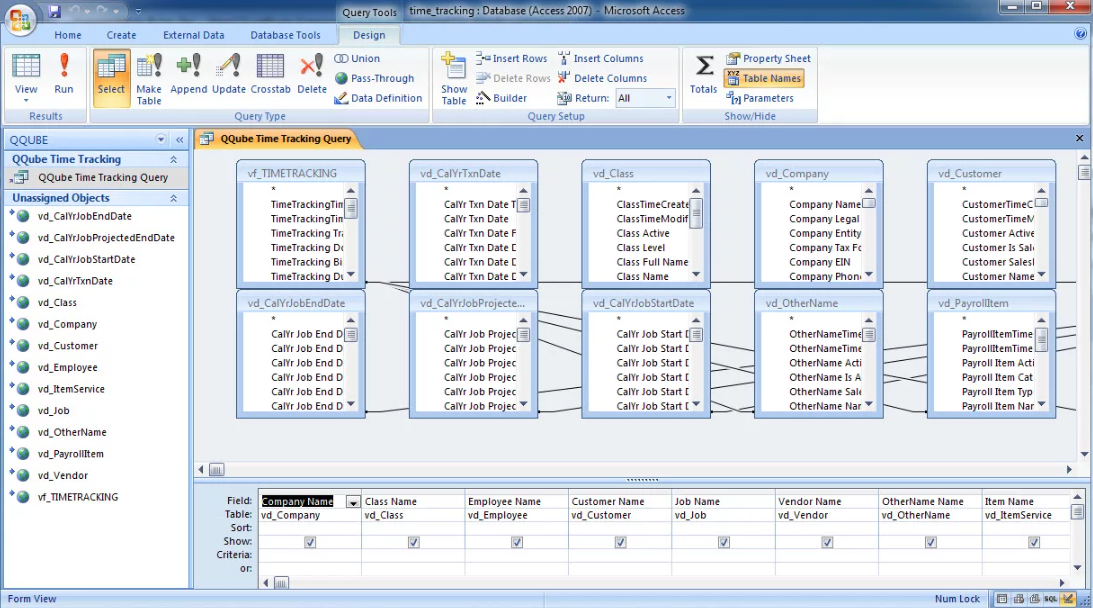
Интерфейс приложения представлен на рисунке 1.1.

Рис. 1.1 – «Microsoft Access»

Недостатки «Microsoft Access»:

* Ограничение размера базы данных в 2 ГБ;
* Отсутствие триггеров и хранимых процедур;
* Стоит довольно дорого.

# **Разработка архитектуры проекта**

## **Обобщенная структура**

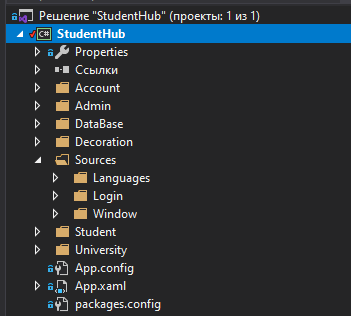
Решение представлено одним проектом StudentHub, имеющим структуру, представленную на рисунке 2.1.1.

Рисунок 2.1.1 – Обобщенная структура проекта

Описание обобщенной структуры проекта представлено в таблице 2.1.1.

Таблица 2.1.1 – Обобщенная структура проекта

|  |  |
| --- | --- |
| Объект | Описание |
| Properties | Свойства проекта, содержит информацию о сборке, используемых ресурсах и настройках |
| Ссылки | Перечень сборок, используемых в проекте |
| Account | Содержит в себе окна для работы с аккаунтом, а именно авторизация и регистрация |
| Admin | Содержит в себе окна для администратора |
| DataBase | Содержит в себе класс для работы с базой данных |
| Decoration | Содержит в себе словари ресурсов (XAML) для оформления ЭУ и не только |
| Sources | Содержит в себе 3 подпапки, в которых находятся необходимые объекты для оформления приложения (иконки, рисунки, словари ресурсов) |
| Student | Содержит в себе окна для студента |
| University | Содержит в себе классы, необходимые для работы с сущностями университета |

Продолжение таблицы 2.1.1

## **Проектирование логической структуры базы данных**

Для реализации поставленной задачи была создана база данных. Для её создания использовались система управления реляционными базами данных Microsoft SQL Server 2019.

База данных – совокупность данных, организованных по определенным правилам, предусматривающим общие принципы описания, хранения и манипулирования данными, независимо от прикладных программ.

Система управления базами данных – совокупность программ и языковых средств, предназначенных для управления данными в базе данных, ведения базы данных и обеспечения взаимодействия ее с прикладными программами. База данных нашей программной подсистемы состоит из 11 таблиц, ее схема изображена в Приложении А.

Таблица Users используется для хранения данных о пользователях. Включает 3 столбца: UserId – id пользователя, UserName – имя пользователя, UserPassword – пароль пользователя (при записи в БД хешируется)

Таблица Adjustment используется для хранения информации об отработках. Включает 5 столбцов: AdjustmentId – id отработки, StudentId – id студента, SubjectName – название предмета, по которому стоит отработка, AdjustmentStatus – статус отработки (0 – в обработке, 1 – отклонено, 2 – принято), ADate – дата отработки.

Таблица Admin используется для хранения информации об администраторе. Включает 3 столбца: AdminId – id администратора, AdminName – имя администратора, UserId – id пользователя.

Таблица BadStudent используется для хранения информации о студентах, у которых пропусков по предметам больше, чем 30. Включает 2 столбца: StudentId – id студента, Gaps – количество пропусков на данный момент.

Таблица Faculty используется для хранения информации о факультетах. Включает 2 столбца: Faculty – название факультета в сокращенном виде, FacultyName – название факультета в полном виде.

Таблица Progress используется для хранения данных об успеваемости студентов. Включает 5 столбцов: ProgressId – id успеваемости, StudentId – id студента, SubjectName – название предмета, по которому студент получил оценку, Note – оценка, PDate – дата, когда была выставлена оценка.

Таблица Retake используется для хранения информации о пересдачах. Включает 5 столбцов: RetakeId – id пересдачи, StudentId – id студента, SubjectName – название предмета, по которому стоит пересдача, RetakeStatus – статус пересдачи (0 – в обработке, 1 – отклонено, 2 – принято), RDate – дата пересдачи.

Таблица Specialization хранит информацию об специальностях на факультетах. Включает 2 столбца: Specialization – название специальности, Faculty – название факультета, к которому принадлежит специальность.

Таблица Student используется для хранения информации об студентах. Включает 10 столбцов: StudentId ­– id студента, UserId – id пользователя, StudentName – имя студента, StudentStatus – статус студента (Student – обычный студент, Elder – староста), Course – курс студента, на котором он учится, GroupId – группа студента, в которой он учится, Specialization – специальность студента, на которой он учится, Faculty – факультет студента, на котором он учится, Birthday – дата дня рождения студента, Email – генерированный от администратора почтовый ящик (none – не существует). Почтовый ящик записывается в виде: Электронная\_почта@belstu.by;пароль

Таблица Subject используется для хранения информации об предметах. Включает 3 столбца: Subject – название предмета в сокращенном виде, SubjectName – название предмета в полном виде, Faculty – факультет, на котором преподают предмет.

Таблица SubjectGaps используется для хранения информации об пропусках по предметам. Включает 3 столбца: StudentId – id студента, которому выставили пропуски, SubjectName – название предмета в сокращенном виде, по которому стоят пропуски, GapsCount – количество пропусков.

## **Структура проекта в виде схемы**

Схема связи между окнами проекта представлена на рисунке 2.3.1

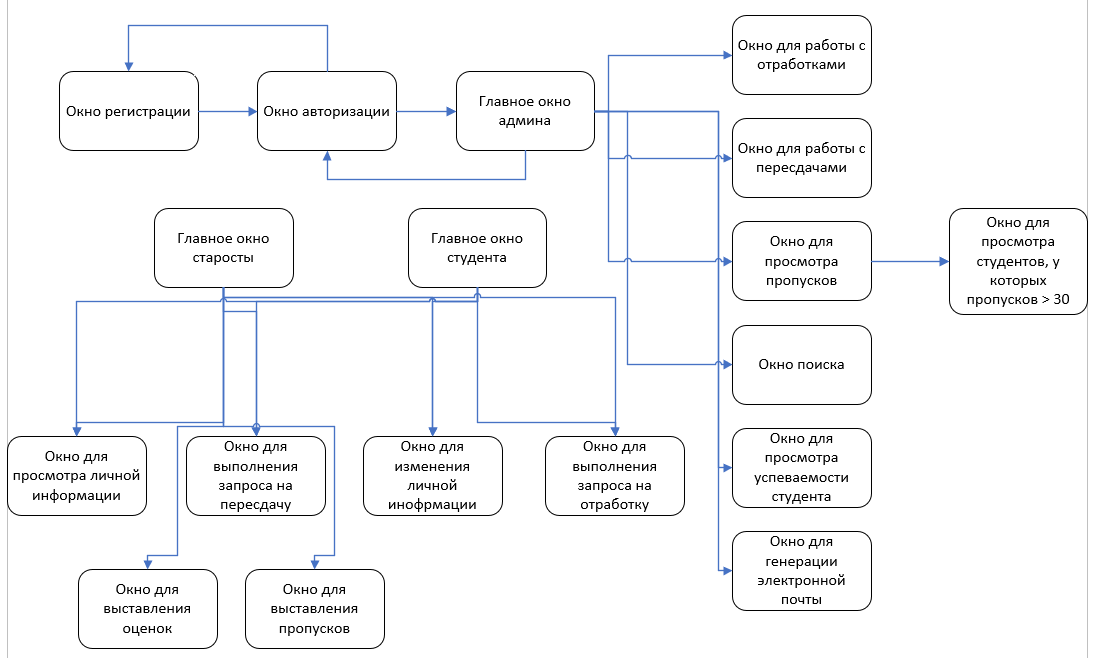


Рис. 2.3.1 – Схема связи между окнами

## **Пример использования**

Диаграмма использования “Авторизованный пользователь”, “Неавторизованный пользователь”, “Администратор” предоставлены в Приложении Б.

# **Разработка архитектуры проекта**

## **Система для проектирования**

Для проектирования проекта использовалась система WPF. Windows Presentation Foundation (WPF) — аналог WinForms, система для построения клиентских приложений [Windows](https://ru.wikipedia.org/wiki/Windows" \o "Windows) с визуально привлекательными возможностями взаимодействия с пользователем, графическая (презентационная) подсистема в составе [.NET Framework](https://ru.wikipedia.org/wiki/.NET_Framework) (начиная с версии [3.0](https://ru.wikipedia.org/wiki/.NET_Framework_3.0)), использующая язык [XAML](https://ru.wikipedia.org/wiki/XAML).

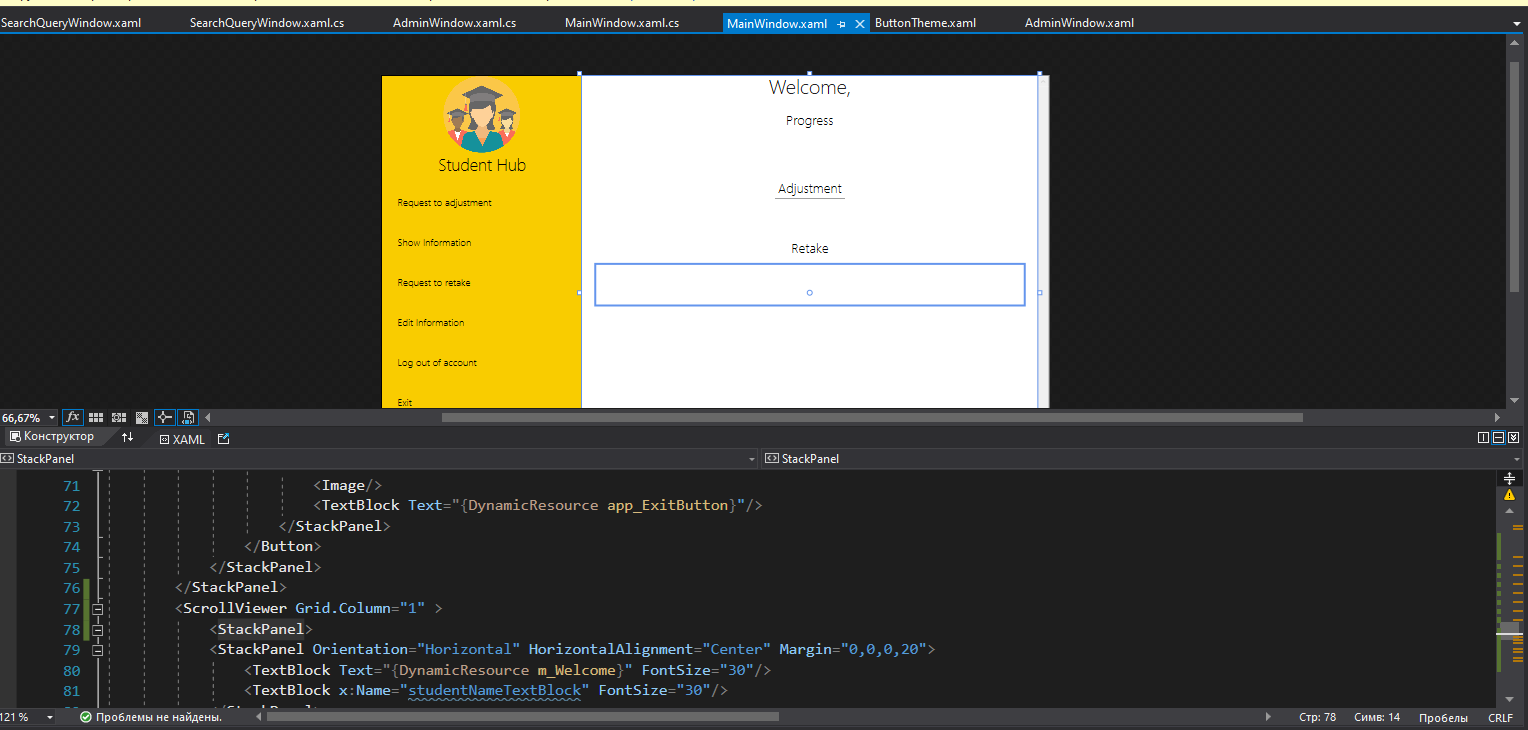
Пример работы на WPF показан на рисунке 3.1.1 

Рис. 3.1.1 – Пример работы на WPF

## **Схема и описание авторизации**

Схема авторизации представлена в приложении B. Сразу после запуска программу пользователь попадает в окно авторизации. Если поля логина и пароля будут пустыми, в окне будет выведено сообщение об ошибке. После программа обращается к базе данных и проверяет существует ли пользователь с соответствующим логином и паролем. Если пользователь не найден, будет выведено сообщение об ошибке, в противном случае вход в приложение.

## **Работа с дизайном**

NuGet предоставляет достаточное количество библиотек для работы с дизайном. Мой выбор упал на библиотеки MaterialDesignColors и MaterialDesignThemes. Пример работы с Material Design представлен на рисунке 3.3.1.

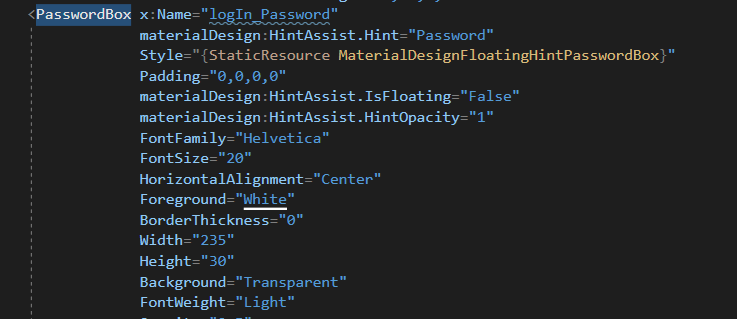


Рис. 3.3.1 – Пример использования Material Design

## **Подключение к БД**

Если различные механизмы работы с БД в С#. Т.к. моя база данных небольшая, мой выбор пал на ADO.Net. ADO.Net предоставляет самый прямой способ доступа к данным в .Net Framework. В Приложении Г продемонстрирован листинг одного из запросов к БД.

# **Тестирование**

* 1. **Вход под пользователем**

После запуска приложения мы попадаем в окно авторизации. Если данные введены неверно либо такого пользователя не существует будет выведено сообщение. На рисунке 4.1.1 представлено окно авторизации.

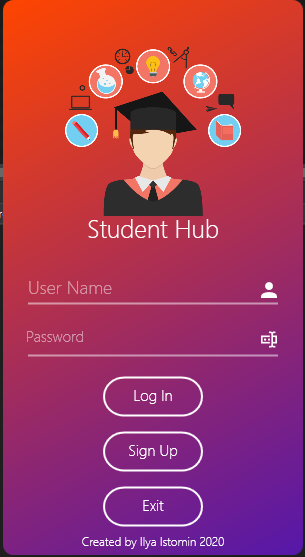
****

Рис. 4.1.1 – Авторизация

После успешной авторизации мы попадем на главную страницу пользователя. На рисунке 4.1.2 мы можем ее видеть. Интерфейс удобный и понятный. Реализован проект с множеством окон, чтобы было удобнее пользоваться. С главной страницы мы можем отправить запрос на отработку, пересдачу, заполнить информацию о себе, а также ее изменить. На рисунке 4.1.3 и 4.1.4 продемонстрированы скриншоты приложения.

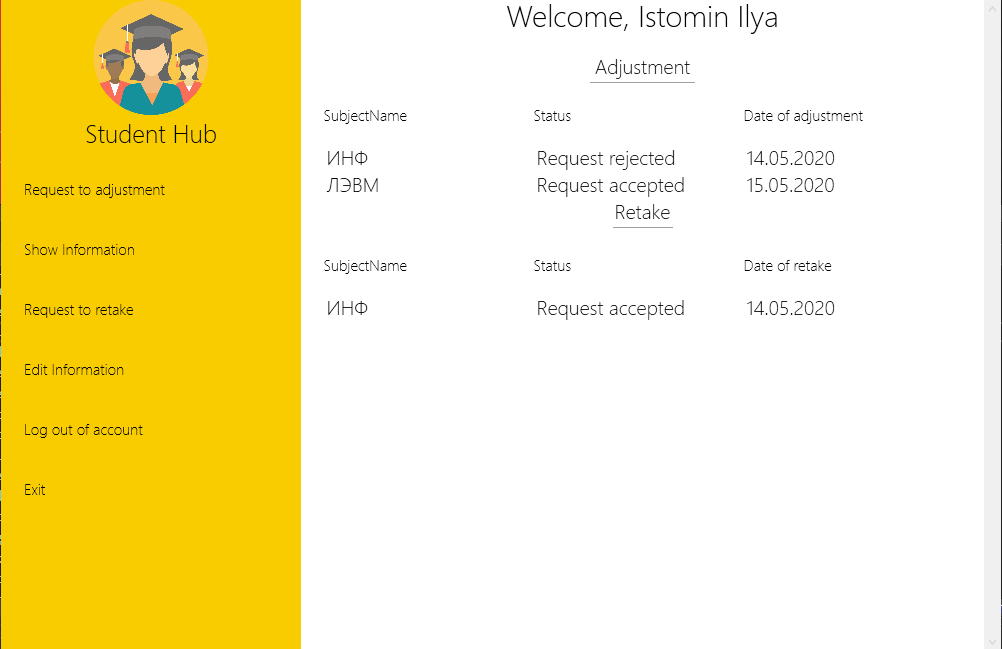


Рис. 4.1.2 – Главная страница

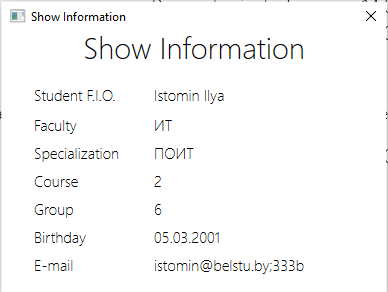


Рис. 4.1.3 – Кабинет пользователя

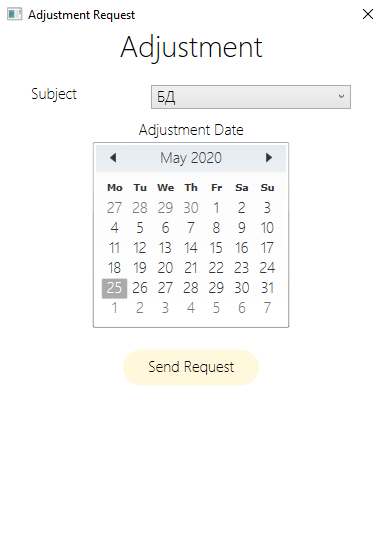


Рис. 4.1.4 – Окно запроса на отработку

* 1. **Вход под админом**

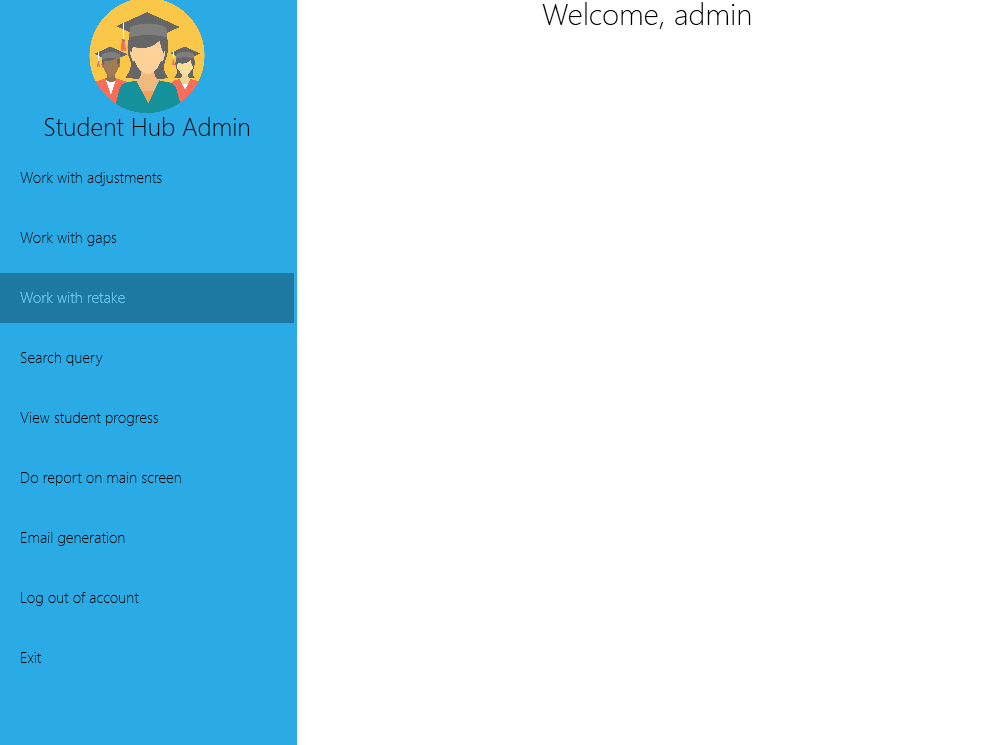
Админ контролирует базу отработок, пересдач и пропусков. Соответственно он может работать с ними. На рисунке 4.2.1 продемонстрировано окно админа.

Рис. 4.2.1 – Окно админа

На рисунке 4.2.2 продемонстрирована форма для работы с пересдачами.

Рис. 4.2.2 – Форма для работы с пересдачами

* 1. **Валидация**

Каждый ошибается. Поэтому в моем приложении предусмотрены некоторые варианты обхода глупых ошибок.

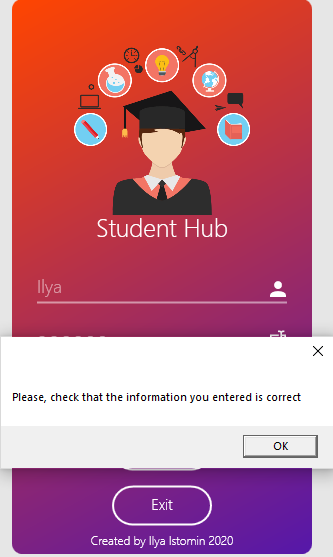
****Например пользователь ввел неверный пароль или логин (результат на рис 4.3.1):

Рис. 4.3.1

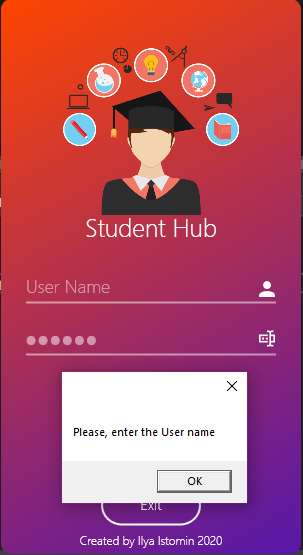
****Или вы упустили пустое поле и нажали войти (результат на рис. 4.3.2):

Рис. 4.3.2 – Сообщение об ошибке при входе

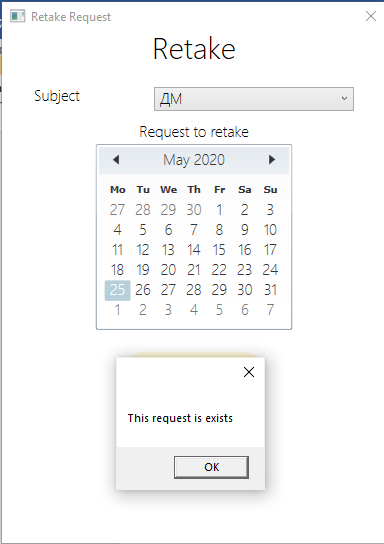
И вот еще пример, где мы хотели добавить два раза одну и ту же отработку на запрос

Рисунок 4.3.3 – Сообщение об ошибке при добавлении

## **ЗАКЛЮЧЕНИЕ**

В ходе выполнения курсовой работы был разработан проект, являющийся помощником в управлении учебным процессом. Во время выполнения данной курсовой работы было изучено немало теоретического материала, а также просмотрено и разобрано большое количество уже готовых решений тех или иных задач.

В ходе выполнения курсовой работы было выполнено следующее:

- разработка базы данных для хранения информации;

- разработка архитектуры приложения;

- разработка функциональных части приложения;

- разработка пользовательского интерфейса;

- написание исходного кода приложения;

- тестирование приложения.

В соответствии с полученным результатом работы программы можно сделать вывод, что разработанная программа работает, верно, а требования технического задания выполнены в полном объеме.

Само приложение – первый опыт в данной области, поэтому не факт, что все баги исправлены. Естественно, существует ряд приложений, имеющих функционал и дизайн в разы лучше. Подводя итог всей курсовой работе, можно сделать выводы, что в разработке приложений мне предстоит еще долгий путь.

## **Список литературных источников**

1. «CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. 4-е издение». Автор Джеффри Рихтер, 2013

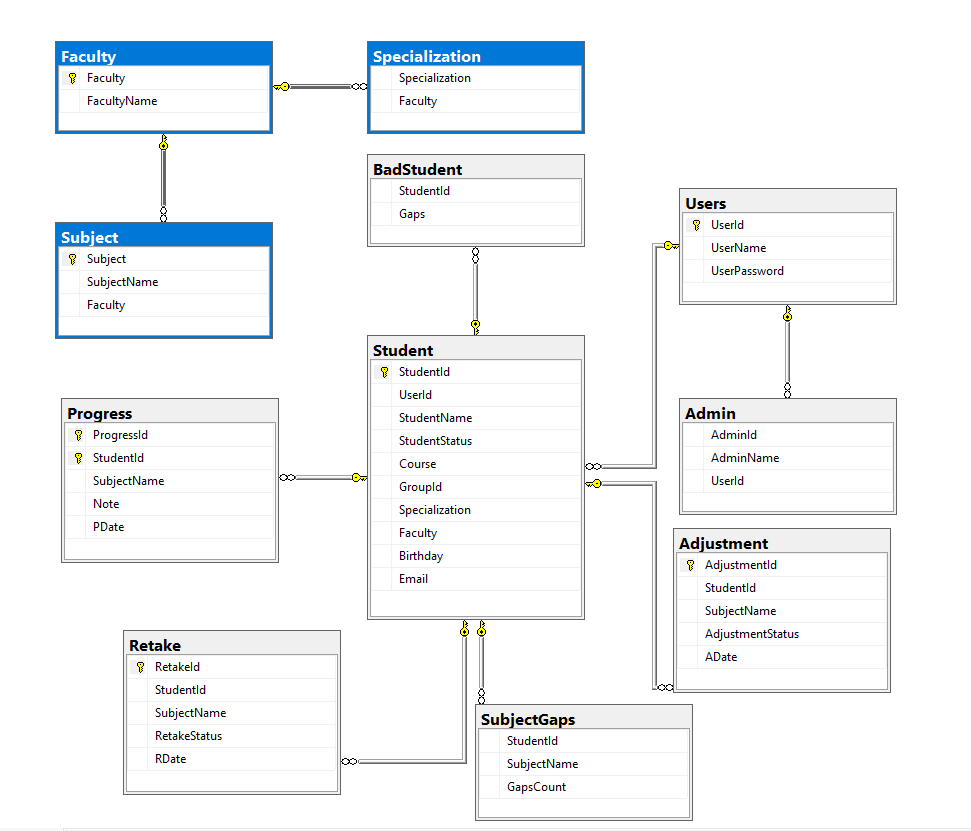
2.METANIT SQL – <https://metanit.com/sql/>.

3.METANIT C# – <https://metanit.com/sharp/>.

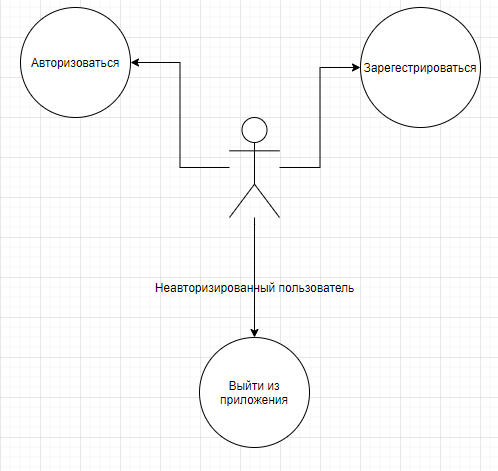
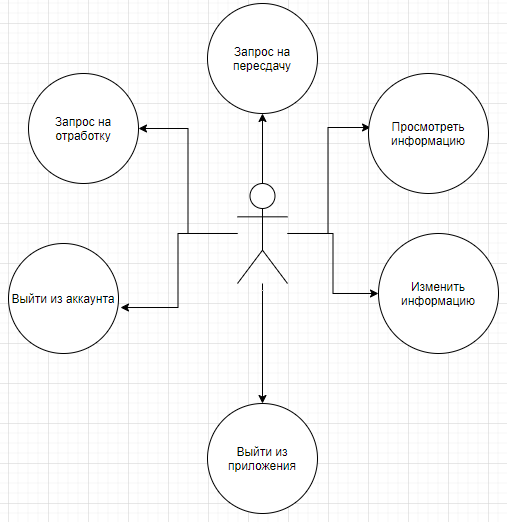
4. Помощник в построении базы данных – <https://ondras.zarovi.cz/sql/demo/>

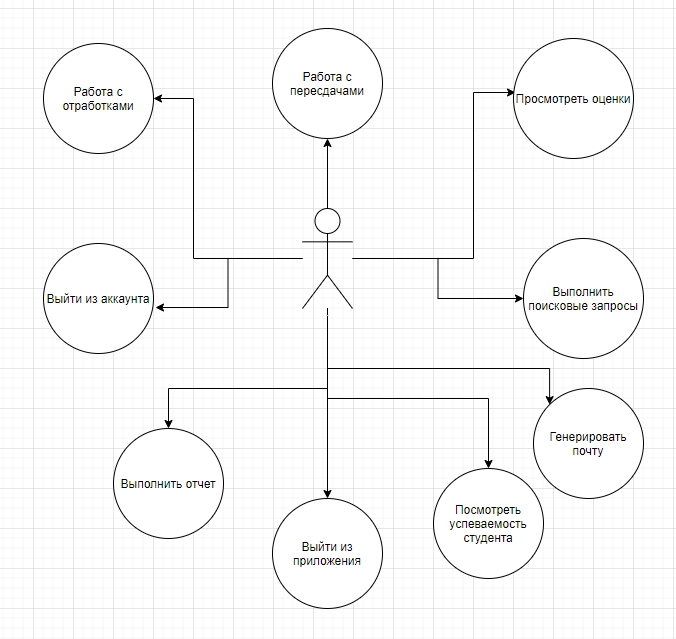
**ПРИЛОЖЕНИЕ А**

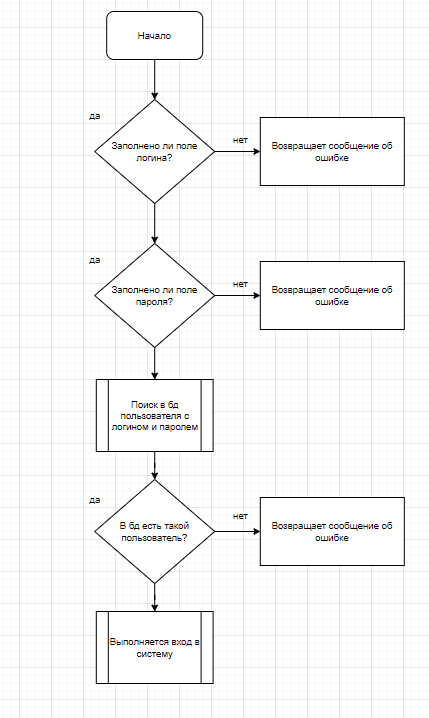
Логическая структура базы данных

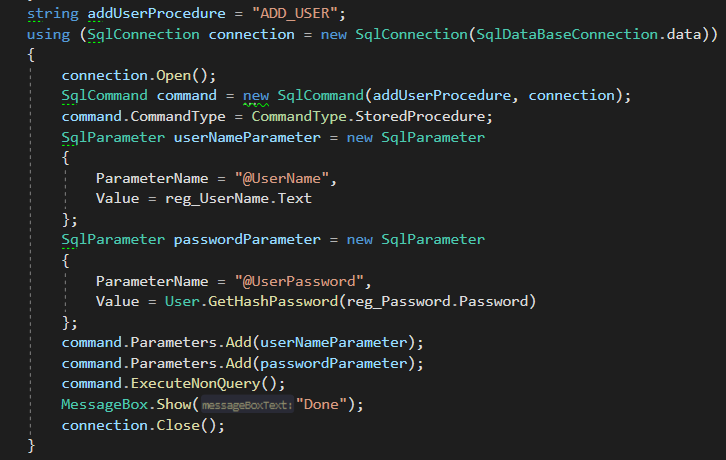


**ПРИЛОЖЕНИЕ Б**





**ПРИЛОЖЕНИЕ В**

**ПРИЛОЖЕНИЕ Г**