Министерство образования Республики Беларусь

Учреждение образования «Минский государственный колледж электроники»

Отчёт

По учебной практике по разработке и сопровождению программного обеспечения

Разработчик: Жгуновский О.Б.

Руководитель практики: Потоцкий Д.С.

Минск, 2021

Оглавление

[ВВЕДЕНИЕ 3](#_Toc74729864)

[1. Постановка задачи 4](#_Toc74729865)

[1.1 Описание предметной области 4](#_Toc74729866)

[1.2 Обзор существующих аналогов 4](#_Toc74729867)

[1.3 Информационная база задачи 4](#_Toc74729868)

[1.3.1 Входная и выходная информация 5](#_Toc74729869)

[1.4 Формулировка задачи 5](#_Toc74729870)

[2. Проектирование задачи 7](#_Toc74729871)

[2.1 Обоснование выбора средств для разработки 7](#_Toc74729872)

[2.2 Разработка алгоритма решения 7](#_Toc74729873)

[2.3 Проектирование интерфейса 8](#_Toc74729874)

[3. Реализация задач 9](#_Toc74729875)

[3.1 Описание разработанных модулей 9](#_Toc74729876)

[3.2 Описание реализованных методов и событий модулей 9](#_Toc74729877)

[3.3 Компоненты WFP 9](#_Toc74729878)

[4. Тестирование 11](#_Toc74729879)

[4.1 Основная информация о тестировании 11](#_Toc74729880)

[4.2 Функциональное тестирование 11](#_Toc74729881)

[5. Применение 13](#_Toc74729882)

[5.1 Установка и системные требования приложения 13](#_Toc74729883)

[5.2 Демонстрационный пример работы 13](#_Toc74729884)

[Заключение 18](#_Toc74729885)

[Список использованных источников 19](#_Toc74729886)

[Приложение А (обязательное) 20](#_Toc74729887)

[Приложение Б (обязательное) 32](#_Toc74729888)

[Приложение В (обязательное) 33](#_Toc74729889)

[Приложение Г (обязательное) 34](#_Toc74729890)

[Приложение Д (обязательное) 35](#_Toc74729891)

[Приложение Е (обязательное) 36](#_Toc74729892)

[Приложение Ж (обязательное) 37](#_Toc74729893)

[Приложение К (обязательное) 38](#_Toc74729894)

[Приложение Л (обязательное) 39](#_Toc74729895)

# ВВЕДЕНИЕ

Существует большое количество различных программ, создание для упрощения определённых задач. Из всех возможных вариантов, мне выпала задача создать программу, которая управляет базой данных учебного заведения. Данный тип программ, весьма популярен в учебных заведениях, ведь он позволяет работать с информацией в колледже быстрее, нежели делать это вручную. Так же, данные программы удобны и для учащихся и их родителей, которые хотят посмотреть информацию об их учебном заведении.

Целью данной работы, является создание приложения с использованием технологии Windows Presentation Foundation (далее – WPF), на языке программирования C#, в среде программирования Visual Studio 2019, с помощью СУБД ACCESS. Данное приложение должно позволять пользователю работать с базой данных учебного заведения. Так же, целью является закрепление знаний полученных в ходе практики, и в процессе изучения дисциплин «Конструирование программ и языки программирования», «Технологии разработки ПО», «Базы данных».

Данная программа должна быть:

* Простая в освоении;
* С эргономичным интерфейсом;
* Не переполнена избыточным функционалом.

Данная пояснительная записка является описанием разработанного программного продукта. Она разделена на разделы и подразделы, и каждый раздел или подраздел посвящён определённому аспекту разработки приложения.

# 1. Постановка задачи

# 1.1 Описание предметной области

Целью данной работы является разработка программы, предназначенной для автоматизации процесса работы учебного заведения. Программа должна предоставлять возможность добавления, удаления, редактирования полей, а также должна присутствовать возможность сортировки информации.

# 1.2 Обзор существующих аналогов

На данный момент существует множество аналогов данного приложения. Однако, они либо находятся в закрытом доступе, либо распространяются с помощью платной модели распространения. Как пример платного решения, можно привести решение от студии 1С «1С:Предприятие 8. Колледж», функциональные возможности которого состоят из:

* комплексно автоматизировать управление бизнес-процессами, в частности работу приемной комиссии, оперативное управление учебно-методическим процессом, студенческим контингентом;
* предоставить возможность накопления информации для анализа и дальнейшего принятия эффективных управленческих решений, что повысит качество предоставляемых услуг;
* создать в учебном заведении «базу знаний», обеспечив сохранение и использование имеющихся «know-how» в области образовательного процесса;
* обеспечить «прозрачность» управления учебным и вспомогательными бизнес-процессами;
* предоставить учащимся и их родителям дополнительные информационные услуги.

Однако, всё это удовольствие стоит 2016 бел. руб. за 1 рабочее место, что для нашего заказчика – непозволительная роскошь.

Данное решение представляет собой Open Source проект с бесплатной моделью распространения. В связи с этим, он может при необходимости свободно дорабатываться заказчиком и без текущего разработчика, а так же, передаваться бесплатно сторонним лицам.

# 1.3 Информационная база задачи

Основной частью внутримашинного информационного обеспечения является информационная база.

Информационная база (ИБ) ─ это определенным способом организованная совокупность данных, хранимых в памяти вычислительной системы в виде файлов, с помощью которых удовлетворяются информационные потребности управленческих процессов и решаемых задач.

Файл ─ это некоторое множество записей однородной структуры, предназначенное для решения экономических задач. Запись ─ это набор полей определенного формата, объединенных по общему ключевому полю.

Данная программа производит операции над входными данными, выводит результаты в виде выходных данных. Так же, данная программа сохраняет результаты своей работы по желанию пользователя, и позволяет их выводить.

# 1.3.1 Входная и выходная информация

Входными данными для разработанного ПП являются:

* Нажатия пользователя;
* Поля таблиц.

Выходными данными разработанного ПП являются:

* Информация отображаемая на экран по запросу пользователя.
* Постоянными данными является база данных.

# 1.4 Формулировка задачи

Создать приложение с использованием технологии WPF на языке программирования C#. Программа должна предоставлять возможность добавления, удаления, изменения и поиска в БД.

Разрабатываемый ПП должен выполнять функции:

1. Ведение БД. Добавление, удаление и редактирование информации в БД.
2. Сортировка. Сортировка информации в БД.

Разработать для проектируемого программного средства:

* техническое задание;
* функциональную диаграмму;
* диаграмму потоков данных;
* диаграмму сущность-связь;
* диаграмму вариантов использования;
* диаграмму деятельности;
* диаграмму последовательности;
* диаграмму состояний;
* диаграмму классов.

Произвести тестирование разработанного программного продукта. Для проведения тестирования на основании разработанных требований написать чек-лист и тест-кейсы.

Чек-лист и тест-кейсы должны быть разработаны, как для смоук- теста, так и для теста критического пути.

# 2. Проектирование задачи

# 2.1 Обоснование выбора средств для разработки

В качестве платформы была взята платформа NET, потому что платформа NET обладает достаточным функционалом для реализации данной программы на ней, а так же, проста в обращении.

Visual Studio – среда разработки, поставляемая вместе с .NET, предоставляет необходимый инструментарий для эффективного и быстрого создания приложений с графическим интерфейсом.

Язык программирования С# объектно-ориентированный язык программирования. Разработан в 1998—2001 годах группой инженеров компании Microsoft под руководством Андерса Хейлсберга и Скотта Вильтаумота как язык разработки приложений для платформы Microsoft .NET Framework. Впоследствии был стандартизирован как ECMA-334 и ISO/IEC 23270.

C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, переменные, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML.

WPF - Windows Presentation Foundation (WPF) — аналог WinForms, система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая (презентационная) подсистема в составе .NET Framework (начиная с версии 3.0), использующая язык XAML.

WPF предустановлена в Windows Vista (.NET Framework 3.0), Windows 7 (.NET Framework 3.5 SP1), Windows 8 (.NET Framework 4.0 и 4.5), Windows 8.1 (.NET Framework 4.5.1) и Windows 10 (.NET Framework 4.7). С помощью WPF можно создавать широкий спектр как автономных, так и запускаемых в браузере приложений.

# 2.2 Разработка алгоритма решения

Работа программы будет начинаться с меню входа в программу. В данном меню будет пользователю будет предоставлен выбор: войти как преподаватель, или как ученик.

Если пользователь входит как ученик, то у него отсутствует часть функционала, но ему не нужно вводить логин и пароль.

Если пользователь входит как подаватель, ему доступен весь функционал, но ему нужно ввести свои инициалы (логин) и пароль.

После входа, пользователю предоставляется меню из множества кнопок. На данном меню, все кнопки кроме двух ведут к следующей форме. В случае, если пользователь нажал на кнопку «Разлогиниться», то происходит выход из его учётной записи, и его возвращает на предыдущую форму. В случае, если пользователь нажал на кнопку «Выйти», производится выход из программы. В случае, если пользователь нажал на любую другую кнопку на форме, то открывается другая форма.

Открывшаяся форма состоит из таблицы, кнопки «Сохранить» и кнопки «Назад». Таблица заполнена информацией из БД, в соответствии с тем, что выбрал пользователь. При нажатии кнопки «Сохранить», происходит сохранение изменений из таблицы в БД. Если пользователь нажал на кнопку «Назад», происходит возврат на предыдущую форму.

# 2.3 Проектирование интерфейса

Программа будет состоять из 3 форм, включая главную. А именно:

* форма логина;
* форма выбора таблицы;
* форма таблицы.

Формы построены с помощью Grid.

Они содержат Button, Label, DataGrid, TextBox.

# 3. Реализация задач

# 3.1 Описание разработанных модулей

В разработанной программе имеется 3 модулей.

Главный модуль «MainWindow.xaml» предназначен для входа в программу.

Модуль «Menu.xaml» предназначен для выбора таблицы.

Модуль «Tables.xaml» предназначен для просмотра и редактирования таблицы.

У каждого модуля есть одноимённый модуль, оканчивающийся на «.cs», в котором содержится логика формы.

# 3.2 Описание реализованных методов и событий модулей

Методы и события формы «MainWindow.xaml.cs»:

private void Button\_Click(object sender, RoutedEventArgs e) – записывает, что пользователь пытается войти как преподаватель.

private void Button\_Back(object sender, RoutedEventArgs e) – записывает, что пользователь больше не пытается войти как преподаватель.

private void Button\_Loin(object sender, RoutedEventArgs e) – производит предоставление прав (если пользователь вошёл как преподаватель) и производит переход на другую форму.

Методы и события формы «Menu.xaml.cs»:

private void Button\_Click(object sender, RoutedEventArgs e) – производит возврат на предыдущую форму, выход из программы или перемещение на другую форму.

Методы и события формы «Tables.xaml.cs»:

private void Button\_Return(object sender, RoutedEventArgs e) – возвращает на предыдущую форму.

private void Close(object sender, EventArgs e) – закрывает поток базы данных.

private void Button\_Save(object sender, RoutedEventArgs e) – сохраняет изменённую таблицу.

# 3.3 Компоненты WFP

При разработке программы использовались следующие компоненты:

* TextBox – компонент предназначен для ввода информации.
* Button – компонент представляет собой элемент управления «Кнопка Windows». В моем случае использовался для запуска процессов реализации методов;
* Label – компонент представляет собой стандартную метку Windows для создания текстовых пояснений;
* DataGrid – компонент для отображения таблицы;
* Grid – контейнер для компонентов.

# 4. Тестирование

# 4.1 Основная информация о тестировании

Тестирование программного обеспечения — процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и её ожидаемым поведением на конечном наборе тестов, выбранных определённым образом.

# 4.2 Функциональное тестирование

Функциональное тестирование — это тестирование ПО в целях проверки реализуемости функциональных требований, то есть способности ПО в определённых условиях решать задачи, нужные пользователям. Функциональные требования определяют, что именно делает ПО, какие задачи оно решает.

В ходе функционального тестирования, была выявлена необходимость в добавлении некоторых кнопок, а также, в удалении существующих и/или изменении их функционала.

После произведения изменений, программа была протестирована повторно. В ходе повторного тестирования недочёты обнаружены небыли.

Результаты проведенного тестирования приведены в таблице 4.1.

Таблица 4.1 - Результаты тестирования программного продукта

|  |  |  |
| --- | --- | --- |
| Название теста | Последовательность действий | Результат |
| Проверка исполняемого файла программы. | Запуск файла “RPO\_college.exe». | Запуск программы. |
| Проверка работоспособности кнопок возврата на предыдущую форму. | Переход к форме “tables»;  Возврат на кнопку «Назад». | Кнопка “Назад” работает корректно. |
| Проверка работоспособности кнопки “Войти как…”. Проверка кнопки “Разлогинится” | Нажать кнопку “Войти как ученик”;  Нажать любую кнопку, которая подразумевает открытие формы с талблицей;  Вернутся на форму логина;  Нажать кнопку “Войти как преподаватель”;  Ввести логин и пароль;  Нажать любую кнопку, которая подразумевает открытие формы с талблицей. | Кнопка логина работает корретно. Кнопка “Разлогинится” работает корректно. |
| Проверка корректности работы таблицы. | Открытие формы с таблицей;  Изменение её содержимого;  Нажатие на кнопку “Сохранить”;  Возврат на предыдущую форму;  Возврат на форму с таблицей. | Таблица изменяется и сохраняется корректно. |
| Проверка корректности работы кнопки “Выход” | Открытие формы “menu”;  Нажатие на кнопку «Выход» | Все элементы функционируют |

# 5. Применение

# 5.1 Установка и системные требования приложения

Установка программы не требуется. Для открытия программы необходимо открыть файл «RPO\_college.exe»

Программа предназначена для работы всех лиц, имеющих доступ к программному обеспечению.

Минимальные системные требования:

* ОС: Windows 7 / 8.1 / 10;
* Процессор: IBM PC with a 233Mhz 486 processor;
* Оперативная память: 64 MB ОЗУ;
* Видеокарта: A VESA compatible Super VGA card;
* DirectX: Версии 9.0c;
* Место на диске: 2 GB;
* Звуковая карта: Sound Blaster AWE32 sound card;

# 5.2 Демонстрационный пример работы

При запуске приложения открывается главная форма. Она предназначена для входа в программу.

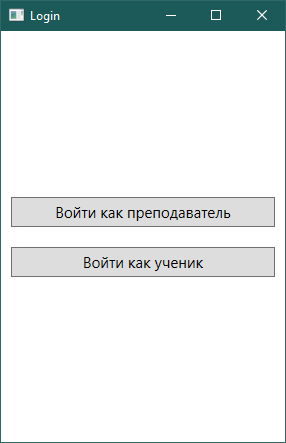


Рис 5.1 – форма «Login»

При нажатии на кнопку «Войти как преподаватель», содержимое формы изменится на представленное на Рис 5.2.

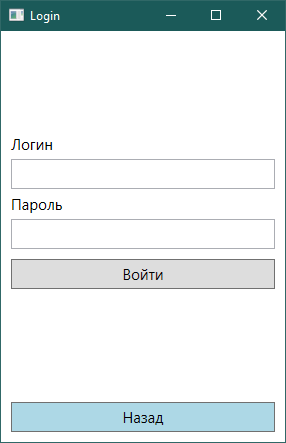


Рис 5.2 - форма «Login» после нажатия на кнопку «Войти как преподаватель»

После ввода корректного логина и пароля, происходит переход на форму, представленную на рисунке 5.3.

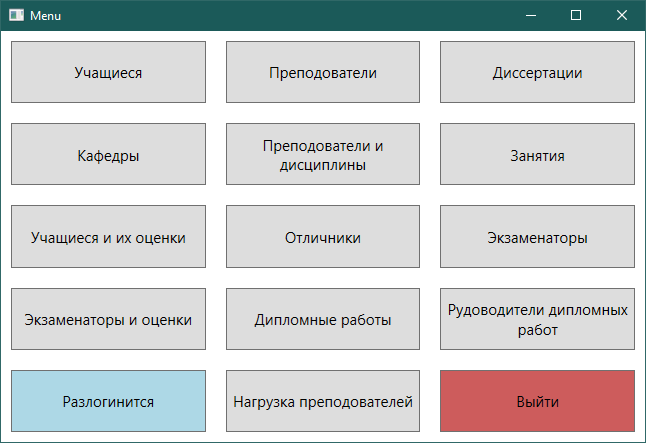


Рис 5.3 – форма «Menu»

В случае нажатия кнопки «Войти как ученик», пользователя сразу перекинет на форму «Menu», без необходимости вводить логин и пароль.

После выбора любой из кнопок белого цвета, пользователя перекинет на форму представленную на рисунке 5.4

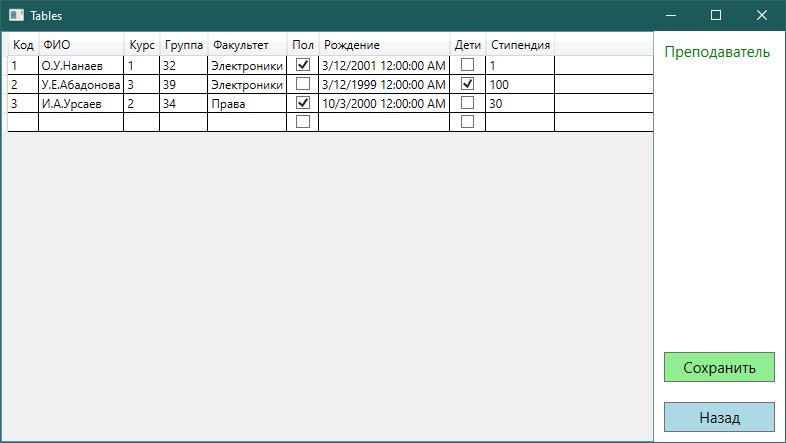


Рис 5.4 – форма «Tables»

В случае, если пользователь произвёл вход в программу как «Ученик», то форма будет иметь вид представленный на рисунке 5.5

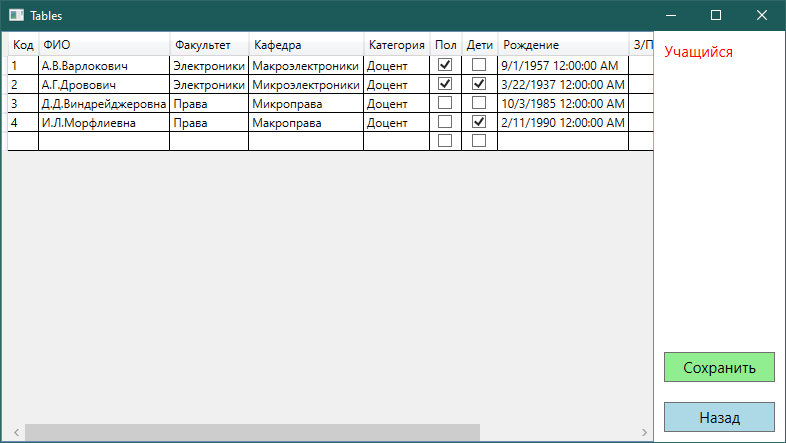


Рис 5.5 – форма «Tables» под ученическим аккаунтом

Содержимое таблиц зависит от выбранного пункта на форме «Menu»

# Заключение

В итоге, было создано приложение позволяющее редактировать и просматривать базу данных учебного заведения, с разграничением ролей в зависимости уровня от уровня доступа, с удобным и эргономичным интерфейсом, который лёгок в освоении и понимании.

В ходе решения поставленной задачи, была изучена технология WFP, изучено управление БД Access, улучшены навыки работы с C#. Были закреплены знания в области дисциплин «Конструирование программ и языки программирования», «Технологии разработки ПО», «Базы данных».

Так же, были улучшены навыки тестирования программ, описания найденных ошибок, и их исправления.

Данная программа является законченным продуктом, который, однако, она может быть расширена путём изменения интерфейса и добавления нового функционала.

# Список использованных источников

1. C# 4.0: полное руководство.: Пер. с англ. — М.: ООО "И.Д. Вильямс", 2011. — 1056 с.: ил. — Парал. тит. англ.
2. Троелсен, Эндрю. Язык программирования C# 2010 и платформа .NET 4, Изд. APress, 2017 г. — 1392 c.
3. Албахари, Джозеф, Албахари, Бен – C# 6.0. Справочник. Полное описание языка – Вильямс, 2015 – 1040 с.
4. Зиборов, В. В. – Visual C# 2012 на примерах – БХВ-Петербург, 2017 – 475 с.
5. Дейтел, Пол, Дейтел, Харви – Как программировать на Visual C# 2012 – Спб.:Питер, 2018 – 858 с.

Список Интернет-ресурсов

1. GitHub.com.
2. Metanit.com.
3. Youtube.com.
4. MSDN.Microsoft.com.
5. eservice.by/1s-kolledzh.

# Приложение А (обязательное)

Текст программы

<Window x:Class="RPO\_college.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:RPO\_college"

mc:Ignorable="d"

Title="Login" Width="300" Height="450" MinWidth="300" MinHeight="350">

<Window.Resources>

<Style TargetType="Button">

<Setter Property="Margin" Value="10"/>

<Setter Property="Height" Value="30"/>

<Setter Property="FontSize" Value="15"/>

</Style>

<Style TargetType="Label">

<Setter Property="FontSize" Value="15"/>

<Setter Property="Margin" Value="5,5,5,0"/>

</Style>

<Style TargetType="TextBox">

<Setter Property="Margin" Value="10,0,10,0"/>

<Setter Property="Height" Value="30"/>

<Setter Property="FontSize" Value="19"/>

<Setter Property="TextWrapping" Value="Wrap"/>

</Style>

</Window.Resources>

<!--А дальше код-->

<Grid>

<Grid x:Name="login">

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Button Grid.Row="1" Content="Войти как преподаватель" Click="Button\_Click"/>

<Button Grid.Row="2" Content="Войти как ученик" Click="Button\_Loin"/>

</Grid>

<Grid x:Name="thcr" Visibility="Hidden">

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<Label Grid.Row="1" Content="Логин"/>

<TextBox Grid.Row="2" Name="Поле\_Логин"/>

<Label Grid.Row="3" Content="Пароль" Margin="5,0,5,0" />

<TextBox Grid.Row="4" x:Name="Поле\_Пароль"/>

<Button Grid.Row="5" Content="Войти" Click="Button\_Loin" IsDefault="True"/>

<Button Grid.Row="9" Content="Назад" Background="LightBlue" Click="Button\_Back"/>

</Grid>

</Grid>

</Window>

<Window x:Class="RPO\_college.Menu"

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:RPO\_college"

mc:Ignorable="d"

Title="Menu" Height="450" Width="550" MinHeight="350" MinWidth="660">

<Window.Resources>

<Style TargetType="Button">

<Setter Property="Margin" Value="10"/>

<Setter Property="MinHeight" Value="35"/>

<Setter Property="FontSize" Value="15"/>

<Setter Property="TextBlock.TextAlignment" Value="Center"/>

</Style>

</Window.Resources>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<!--Получить перечень и общее число студентов указанных групп либоуказанного курса (курсов) факультета полностью, по половому признаку,

году рождения, возрасту, признаку наличия детей, по признаку получения

и размеру стипендии.-->

<Button Grid.Column="0" Grid.Row="0" Content="Учащиеся" Click="Button\_Click"/>

<!--Получить список и общее число преподавателей указанных кафедр либо

указанного факультета полностью, либо указанных категорий

(ассистенты, доценты, профессора и т.д.) по половому признаку, году

рождения, возрасту, признаку наличия и количеству детей, размеру

заработной платы, являющихся аспирантами, защитивших кандидатские,

докторские диссертации.-->

<Button Grid.Column="1" Grid.Row="0" Content="Преподователи" Click="Button\_Click"/>

<!--Получить перечень тем кандидатских и докторских

диссертаций, защитивших сотрудниками-->

<Button Grid.Column="2" Grid.Row="0" Content="Диссертации" Click="Button\_Click"/>

<!--Получить перечень кафедр,и факультетов проводящих занятия в указанной группе-->

<Button Grid.Column="0" Grid.Row="1" Content="Кафедры" Click="Button\_Click"/>

<!--Получить список и общее число преподавателей, проводивших

(проводящих) занятия по указанной дисциплине в указанной группе-->

<Button Grid.Column="1" Grid.Row="1" Content="Преподователи и&#xa;дисциплины" Click="Button\_Click"/>

<!--Получить перечень и общее число преподавателей проводивших

(проводящих) лекционные, семинарские и другие виды занятий в

указанной группе-->

<Button Grid.Column="2" Grid.Row="1" Content="Занятия" Click="Button\_Click"/>

<!--Получить список и общее число студентов указанных групп, сдавших

зачет либо экзамен по указанной дисциплине с указанной оценкой.-->

<Button Grid.Column="0" Grid.Row="2" Content="Учащиеся и их оценки" Click="Button\_Click"/>

<!--Получить список и общее число студентов указанных групп или

указанного курса указанного факультета, сдавших указанную сессию на

отлично, без троек, без двоек.-->

<Button Grid.Column="1" Grid.Row="2" Content="Отличники" Click="Button\_Click"/>

<!--Получить перечень преподавателей, принимающих (принимавших)

экзамены в указанных группах.-->

<Button Grid.Column="2" Grid.Row="2" Content="Экзаменаторы" Click="Button\_Click"/>

<!--Получить список студентов указанных групп, либо которым заданный

преподаватель поставил некоторую оценку за экзамен по определенным

дисциплинам-->

<Button Grid.Column="0" Grid.Row="3" Content="Экзаменаторы и оценки" Click="Button\_Click"/>

<!--Получить список студентов и тем дипломных работ, выполняемых ими на

указанной кафедре либо у указанного преподавателя.-->

<Button Grid.Column="1" Grid.Row="3" Content="Дипломные работы" Click="Button\_Click"/>

<!--Получить список руководителей дипломных работ с указанной кафедры,

либо факультета полностью и раздельно по некоторым категориям

преподавателей-->

<Button Grid.Column="2" Grid.Row="3" Content="Рудоводители дипломных&#xa;работ" Click="Button\_Click"/>

<!--Получить нагрузку преподавателей (название дисциплины, количество

часов), ее объем по отдельным видам занятий-->

<Button Grid.Column="1" Grid.Row="4" Content="Нагрузка преподователей" Click="Button\_Click"/>

<Button Grid.Column="0" Grid.Row="4" Content="Разлогинится" Background="LightBlue" Click="Button\_Click"/>

<Button Grid.Column="2" Grid.Row="4" Content="Выйти" Background="IndianRed" Click="Button\_Click"/>

</Grid>

</Window>

<Window x:Class="RPO\_college.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:RPO\_college"

mc:Ignorable="d"

Title="Login" Width="300" Height="450" MinWidth="300" MinHeight="350">

<Window.Resources>

<Style TargetType="Button">

<Setter Property="Margin" Value="10"/>

<Setter Property="Height" Value="30"/>

<Setter Property="FontSize" Value="15"/>

</Style>

<Style TargetType="Label">

<Setter Property="FontSize" Value="15"/>

<Setter Property="Margin" Value="5,5,5,0"/>

</Style>

<Style TargetType="TextBox">

<Setter Property="Margin" Value="10,0,10,0"/>

<Setter Property="Height" Value="30"/>

<Setter Property="FontSize" Value="19"/>

<Setter Property="TextWrapping" Value="Wrap"/>

</Style>

</Window.Resources>

<!--А дальше код-->

<Grid>

<Grid x:Name="login">

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Button Grid.Row="1" Content="Войти как преподаватель" Click="Button\_Click"/>

<Button Grid.Row="2" Content="Войти как ученик" Click="Button\_Loin"/>

</Grid>

<Grid x:Name="thcr" Visibility="Hidden">

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<Label Grid.Row="1" Content="Логин"/>

<TextBox Grid.Row="2" Name="Поле\_Логин"/>

<Label Grid.Row="3" Content="Пароль" Margin="5,0,5,0" />

<TextBox Grid.Row="4" x:Name="Поле\_Пароль"/>

<Button Grid.Row="5" Content="Войти" Click="Button\_Loin" IsDefault="True"/>

<Button Grid.Row="9" Content="Назад" Background="LightBlue" Click="Button\_Back"/>

</Grid>

</Grid>

</Window>

using System;

using System.Data;

using System.Data.OleDb;

using System.Windows;

using System.Windows.Media;

namespace RPO\_college

{

/// <summary>

/// Логика взаимодействия для Tables.xaml

/// </summary>

public partial class Tables : Window

{

private readonly string getInfo;//readonly

private readonly OleDbDataAdapter da;

private readonly DataSet ds;

private OleDbCommandBuilder cmd;

private readonly OleDbConnection DataBase = new OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data Source=DB.mdb;Persist Security Info=True"); //readonly

public Tables(string WhatWasCho)

{

InitializeComponent();

if (MainWindow.IsTeacher)

{

NotifyIfTeacher.Foreground = Brushes.Green;

NotifyIfTeacher.Content = "Преподаватель";

}

else

{

NotifyIfTeacher.Foreground = Brushes.Red;

NotifyIfTeacher.Content = "Учащийся";

}

Closed += Close;

DataBase.Open();

switch (WhatWasCho)

{

case "Учащиеся":

getInfo = "SELECT \* FROM Учащиеся";

da = new OleDbDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Учащиеся]");

ds.Tables[0].TableName = "MyTable";

MyTable.ItemsSource = ds.Tables["MyTable"].DefaultView;

break;

case "Преподователи":

if (MainWindow.IsTeacher)

{

getInfo = "SELECT \* FROM Преподователи";

}

else

{

getInfo = "SELECT Код, ФИО, Факультет, Кафедра, Категория, Пол, Дети, Рождение, [З/П], [Тип занятий], [Руководитель курсовой] FROM Преподователи";

}

da = new OleDbDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Преподователи]");

ds.Tables[0].TableName = "MyTable";

MyTable.ItemsSource = ds.Tables["MyTable"].DefaultView;

break;

case "Диссертации":

getInfo = "SELECT \* FROM Диссертации";

da = new OleDbDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Диссертации]");

ds.Tables[0].TableName = "MyTable";

MyTable.ItemsSource = ds.Tables["MyTable"].DefaultView;

break;

case "Кафедры":

getInfo = "SELECT Группа, Кафедра FROM Занятия";

da = new OleDbDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Занятия]");

ds.Tables[0].TableName = "MyTable";

MyTable.ItemsSource = ds.Tables["MyTable"].DefaultView;

break;

case "Преподователи и\nдисциплины":

getInfo = "SELECT ФИО, Кафедра FROM Преподователи";

da = new OleDbDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Преподователи]");

ds.Tables[0].TableName = "MyTable";

MyTable.ItemsSource = ds.Tables["MyTable"].DefaultView;

break;

case "Занятия":

getInfo = "SELECT \* FROM Занятия";

da = new OleDbDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Занятия]");

ds.Tables[0].TableName = "MyTable";

MyTable.ItemsSource = ds.Tables["MyTable"].DefaultView;

break;

case "Учащиеся и их оценки":

getInfo = "SELECT \* FROM Оценки";

da = new OleDbDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Оценки]");

ds.Tables[0].TableName = "MyTable";

MyTable.ItemsSource = ds.Tables["MyTable"].DefaultView;

break;

case "Отличники":

getInfo = "SELECT \* FROM Оценки WHERE Оценка > 7";

da = new OleDbDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Учащиеся]");

ds.Tables[0].TableName = "MyTable";

MyTable.ItemsSource = ds.Tables["MyTable"].DefaultView;

break;

case "Экзаменаторы":

if (MainWindow.IsTeacher)

getInfo = "SELECT \* FROM Преподователи WHERE [Руководитель курсовой] = true";

else

getInfo = "SELECT Код, ФИО, Факультет, Кафедра, Категория, Пол, Дети, Рождение, [З/П], [Тип занятий], [Руководитель курсовой] FROM Преподователи WHERE [Руководитель курсовой] = true";

da = new OleDbDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Преподователи]");

ds.Tables[0].TableName = "MyTable";

MyTable.ItemsSource = ds.Tables["MyTable"].DefaultView;

break;

case "Экзаменаторы и оценки":

getInfo = "SELECT \* FROM [Экзамены и курсачи]";

da = new OleDbDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Учащиеся]");

ds.Tables[0].TableName = "MyTable";

MyTable.ItemsSource = ds.Tables["MyTable"].DefaultView;

break;

case "Дипломные работы":

getInfo = "SELECT Оценка, ФИО, Дициплина FROM[Дипломные работы]";

da = new OleDbDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Дипломные работы]");

ds.Tables[0].TableName = "MyTable";

MyTable.ItemsSource = ds.Tables["MyTable"].DefaultView;

break;

case "Рудоводители дипломных\nработ":

getInfo = "SELECT [ФИО Руководителя] FROM [Дипломные работы]";

da = new OleDbDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Дипломные работы]");

ds.Tables[0].TableName = "MyTable";

MyTable.ItemsSource = ds.Tables["MyTable"].DefaultView;

break;

case "Нагрузка преподователей":

getInfo = "SELECT \* FROM Нагрузка";

da = new OleDbDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Нагрузка]");

ds.Tables[0].TableName = "MyTable";

MyTable.ItemsSource = ds.Tables["MyTable"].DefaultView;

break;

default:

MessageBox.Show("Вы всё сломали!");

break;

}

}

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

{

Menu window = new Menu();

window.Show();

this.Close();

}

private void Close(object sender, EventArgs e)

{

try

{

DataBase.Close();

}

catch (Exception err)

{

MessageBox.Show(err.Message, "err with close bd", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

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

{

if (MainWindow.IsTeacher)

{

cmd = new OleDbCommandBuilder(da);

da.Update(ds, "MyTable");

MessageBox.Show("Saved");

}

else

{

MessageBox.Show("Вы не преподаватель", "Не удалось", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

}

}

//WhatWasClosed

/\*

<Button Grid.Column="0" Grid.Row="0" Content="Учащиеся" Click="Button\_Click"/>

<Button Grid.Column="1" Grid.Row="0" Content="Преподователи" Click="Button\_Click"/>

<Button Grid.Column="2" Grid.Row="0" Content="Диссертации" Click="Button\_Click"/>

<Button Grid.Column="0" Grid.Row="1" Content="Кафедры" Click="Button\_Click"/>

<Button Grid.Column="1" Grid.Row="1" Content="Преподователи и&#xa;дисциплины" Click="Button\_Click"/>

<Button Grid.Column="2" Grid.Row="1" Content="Занятия" Click="Button\_Click"/>

<Button Grid.Column="0" Grid.Row="2" Content="Учащиеся и их оценки" Click="Button\_Click"/>

<Button Grid.Column="1" Grid.Row="2" Content="Отличники" Click="Button\_Click"/>

<Button Grid.Column="2" Grid.Row="2" Content="Экзаменаторы" Click="Button\_Click"/>

<Button Grid.Column="0" Grid.Row="3" Content="Экзаменаторы и оценки" Click="Button\_Click"/>

<Button Grid.Column="1" Grid.Row="3" Content="Дипломные работы" Click="Button\_Click"/>

<Button Grid.Column="2" Grid.Row="3" Content="Рудоводители дипломных&#xa;работ" Click="Button\_Click"/>

<Button Grid.Column="1" Grid.Row="4" Content="Нагрузка преподователей" Click="Button\_Click"/>

<Button Grid.Column="0" Grid.Row="4" Content="Разлогинится" Background="LightBlue" Click="Button\_Click"/>

<Button Grid.Column="2" Grid.Row="4" Content="Выйти" Background="IndianRed" Click="Button\_Click"/>

\*/

using System;

using System.Windows;

using System.Windows.Controls;

namespace RPO\_college

{

/// <summary>

/// Логика взаимодействия для Menu.xaml

/// </summary>

public partial class Menu : Window

{

public Menu()

{

InitializeComponent();

}

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

{

Button btn = sender as Button;

Tables;

switch (btn.Content)

{

default:

tables = new Tables(btn.Content.ToString());

tables.Show();

this.Close();

break;

case "Выйти":

Environment.Exit(0);

break;

case "Разлогинится":

MainWindow window = new MainWindow();

window.Show();

this.Close();

break;

}

}

}

}

using System.Data;

using System.Data.OleDb;

using System.Windows;

using System.Windows.Controls;

using System.Linq;

namespace RPO\_college

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

string getInfo;

DataSet ds;

OleDbConnection con;

OleDbDataAdapter da;

static bool \_isTeacher;

static bool \_isTryLoginAsTeacher;

public static bool IsTeacher { get => \_isTeacher; set => \_isTeacher = value; }//t = вошёл как учитель

public static bool IsTryLoginAsTeacher { get => \_isTryLoginAsTeacher; set => \_isTryLoginAsTeacher = value; }//t = пытается войти как учитель

public MainWindow()

{

InitializeComponent();

}

private void Button\_Click(object sender, RoutedEventArgs e)//задейстуется если чел входит как учитель

{

login.Visibility = Visibility.Hidden;

thcr.Visibility = Visibility.Visible;

con = new OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data Source=DB.mdb;Persist Security Info=True");//подключаем БД

getInfo = "SELECT Пароль, ФИО FROM Преподователи";//Выбираем что получать

ds = new DataSet();//создаём датасет

da = new OleDbDataAdapter(getInfo, con);//создаём датаадаптер (отправляем запрос в БД?)

da.Fill(ds,"Преподователи");//заполняем то что хранится в таблице "Преподователи" в ds

con.Close();//закрываем подключение

IsTryLoginAsTeacher = true;

}

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

{

thcr.Visibility = Visibility.Hidden;

login.Visibility = Visibility.Visible;

IsTryLoginAsTeacher = false;

}

private void Button\_Loin(object sender, RoutedEventArgs e)//тут проверять

{

if (IsTryLoginAsTeacher)

{

foreach(DataTable dt in ds.Tables)

{

foreach(DataRow dr in dt.Rows)

{

var cells = dr.ItemArray;

if (cells[0].ToString() == Поле\_Пароль.Text && cells[1].ToString() == Поле\_Логин.Text)//passw, FIO А.В.Варлокович

{

IsTeacher = true;

Menu = new Menu();

menu.Show();

this.Close();

return;

}

}

MessageBox.Show("Неправильный логин или пароль", "Ошибка", MessageBoxButton.OK);

}

}

else

{

Menu = new Menu();

menu.Show();

this.Close();

}

}

}

}

# Приложение Б (обязательное)

Диаграмма вариантов использования



# Приложение В (обязательное)

Диаграмма состояний



# Приложение Г (обязательное)

Функциональная диаграмма

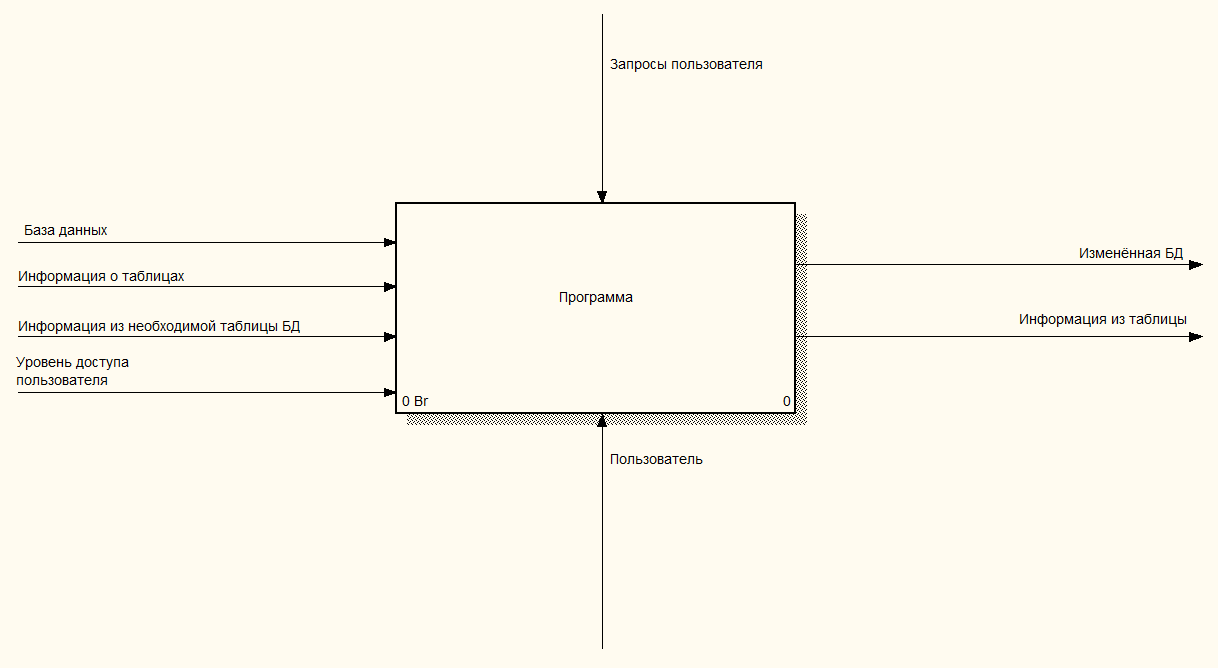


Рисунок Г1 – Контекстная диаграмма

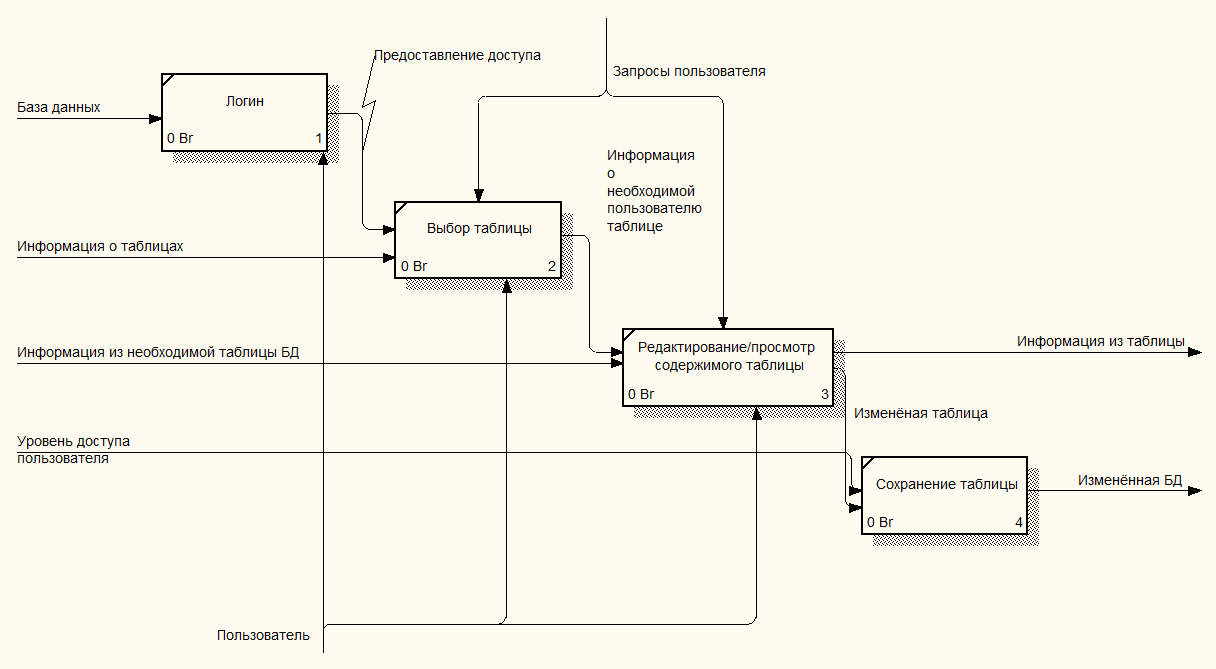


Рисунок Г2 - Декомпозиция первого уровня функциональной модели

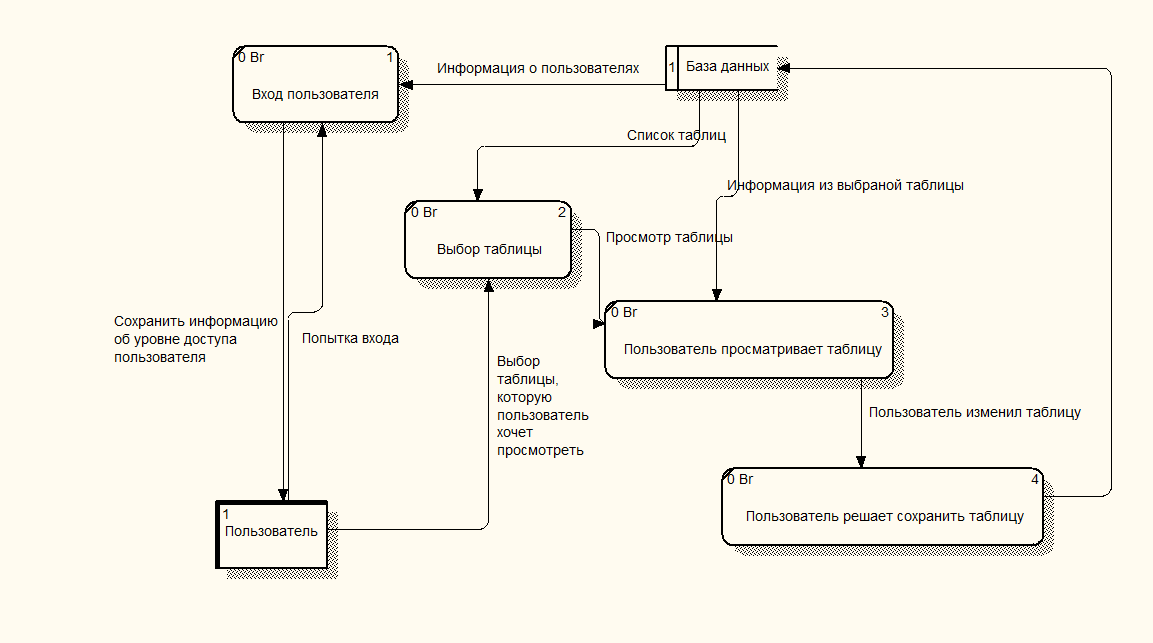
# Приложение Д (обязательное)

Диаграмма деятельности



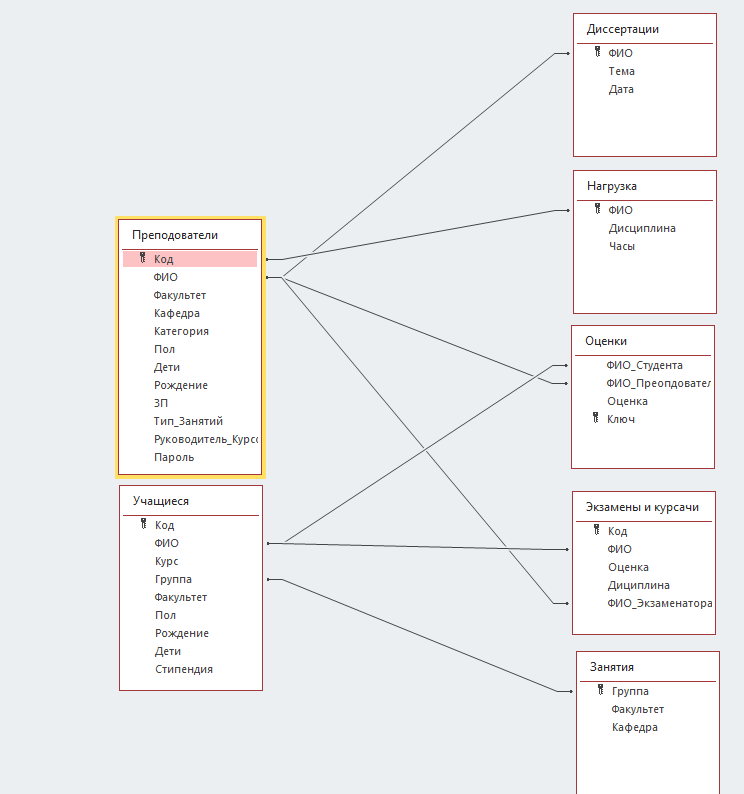
# Приложение Е (обязательное)

Диаграмма потоков данных



# Приложение Ж (обязательное)

Диаграмма «сущность-связь»



# Приложение К (обязательное)

Диаграмма последовательности



# Приложение Л (обязательное)

Диаграмма классов



ещено на Allbest.ru