КОМИТЕТ ПО ОБРАЗОВАНИЮ МИНГОРИСПОЛКОМА

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«МИНСКИЙ ГОСУДАРСТВЕННЫЙ КОЛЛЕДЖ ЭЛЕКТРОНИКИ»

Специальность: 2-40 01 01

«Программное обеспечение информационных технологий»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Разработка приложения «Колледж»

МГКЭ КП38ТП.011109.081 ПЗ081 ПЗ

Автор проекта (О.Б.Жгуновский )

Руководитель проекта (И.М.Вайтович )

2021

СОДЕРЖАНИЕ

Изм.

Лист

№ документа

Подпись

Дата

Лист

2

МГКЭ КП38ТП.011109.081 ПЗ

Разраб.

Жгуновский О.Б

Провер.

Вайиович И.М

.

*Реценз.*

Н. контр.

Утв.

Вайиович И.М

*Разработка приложения «Колледж»*

*Пояснительная записка*

Лит.

Листов

41

УО «МГКЭ»

[ВВЕДЕНИЕ 5](#_Toc88592485)

[1 Постановка задачи 7](#_Toc88592486)

[1.1 Анализ предметной области 7](#_Toc88592487)

[1.2 Обзор проектных решений 7](#_Toc88592488)

[2 Проектирование базы данных 9](#_Toc88592489)

[2.1 Выбор средств (методологий) проектирования. Выбор СУБД 9](#_Toc88592490)

[2.2 Построение концептуальной и логической модели БД 10](#_Toc88592491)

[2.3 Физическая модель БД 13](#_Toc88592492)

[2.4 Разграничение прав доступа 18](#_Toc88592493)

[3 Реализация базы данных 19](#_Toc88592494)

[3.1 Реализация запросов к БД 19](#_Toc88592495)

[3.2 Программная реализация объектов БД 19](#_Toc88592496)

[4 Применение разработанной базы данных в программном приложении 21](#_Toc88592497)

[4.1 Общие сведения 21](#_Toc88592498)

[4.2 Функции: логическая и физическая организация 21](#_Toc88592499)

[4.3 Входные и выходные данные 25](#_Toc88592500)

[4.4 Демонстрационный пример использования 25](#_Toc88592501)

[ЗАКЛЮЧЕНИЕ 29](#_Toc88592502)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 30](#_Toc88592503)

[ПРИЛОЖЕНИЕ А 31](#_Toc88592504)

[ПРИЛОЖЕНИЕ Б 37](#_Toc88592505)

# ВВЕДЕНИЕ

Существует большое количество различных программ, созданных 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 MSSQL. Данное приложение должно позволять пользователю работать с базой данных учебного заведения. Так же, целью является закрепление знаний, полученных в процессе изучения дисциплины «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.

Данная пояснительная записка, включает в себя четыре раздела, содержащих необходимую и достаточную информацию по использованию данного программного средства, и приводит подробное описание программы и вычислительной системы.

Первый раздел пояснительной записки «Постановка задачи» должен обосновать выбор области решения бизнес-задачи и применяемой технологии. Данный раздел пояснительной записки включает следующие подразделы: «Анализ предметной области», «Обзор проектных решений».

В подразделе «Анализ предметной области» описывается предметная область и определяется перечень выполняемых функций проекта, использующего базу данных.

В подразделе «Обзор проектных решений» описываются существующие аналоги разрабатываемой базы данных, предоставляется краткое описание аналогичных проектов. Приводятся достоинства и недостатки существующих решений.

Второй раздел «Проектирование базы данных» содержит выбор средств (методологий) проектирования и обоснование выбора СУБД, а также описывает построение концептуальной и логической модели базы данных. Описывается физическая модель базы данных и разграничение прав доступа.

Третий раздел «Реализация БД» содержит описание реализации запросов к базе данных, а также программную реализацию объектов базы данных, которые были спроектированы для базы данных.

Четвертый раздел «Применение разработанной БД в программном приложении» содержит общие сведения о приложении, описание проекта в среде разработки. Раздел включает в себя описание функций приложения, а также структуру входных и выходных данных. В конце раздела приводится демонстрационный пример использования разработанного приложения, содержащего базу данных.

В заключении описывается выполнение поставленной задачи, степень соответствия проектных решений задания, причины несоответствия, если таковые имеются.

В приложении А приведен текст исходных программ.

В приложении Б приведены диаграммы

К пояснительной записке прилагается графическая часть, которая состоит из двух листов и содержит схему базы данных и диаграмму вариантов использования, спроектированную в ходе объектно-ориентированного анализа и проектирования приложения.

# 1 Постановка задачи

## 1.1 Анализ предметной области

Основной задачей рассматриваемой предметной области является автоматизация работы учебного заведения

Рассмотрим предметную область задачи. Любой пользователь без введения личной информации имеет возможность войти в программу и просматривать информацию, которая в ней находится (за исключением паролей преподавателей).

Преподавателю же необходимо ввести свой логин и пароль для просмотра и редактирования информации.

При традиционной организации деятельность колледжа включает в себя следующее:

* список преподавателей;
* список учащихся, их оценок;
* список работ;
* прочие списки, которые служат для информирования, такие как: отличники, руководители дипломных работ, экзаменаторы и прочее.

Исходя из исследования предметной области требуется создать систему программу, которая позволяет просматривать и редактировать необходимую пользователю информацию.

Ручная обработка всей информации без использования вычислительной техники не может выполняться оперативно, а также требует значительных временных и трудозатрат.

## 1.2 Обзор проектных решений

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 также, передаваться бесплатно сторонним лицам.

# 2 Проектирование базы данных

## 2.1 Выбор средств (методологий) проектирования. Выбор СУБД

При проектировании базы данных после описания предметной области необходимо выбрать метод построения инфологической модели (ER-модели) и СУБД, в которой будет реализован курсовой проект.

Исходя из анализа предметной области для построения ER-модели была выбрана программа CA ERwin Data Modeler 7.3 (ERwin). В качестве обоснования выбора можно привести несколько аргументов.

ERwin Data Modeler позволяет наглядно отображать сложные структуры данных. Удобная в использовании графическая среда упрощает разработку базы данных и автоматизирует множество трудоемких задач, уменьшая сроки создания высококачественных и высокопроизводительных транзакционных баз данных. Наиболее важными для разработки курсового проекта являются следующие возможности данного CASE-средства:

* поддержка нотации IDEF1x;
* возможность совместной работы группы проектировщиков;
* описание структуры БД с помощью визуальных средств;
* автоматическая генерация файлов данных для любого типа СУБД;
* уменьшение числа рутинных операций, что облегчает и сокращает работу.

Выбор системы управления баз данных (СУБД) представляет собой сложную многопараметрическую задачу и является одним из важных этапов при разработке приложений баз данных. Выбранная СУБД должна удовлетворять как текущим, так и будущим потребностям пользователя.

Перечень требований к СУБД, используемых при анализе той или иной информационной системы, может изменяться в зависимости от поставленных целей. Тем не менее можно выделить несколько групп критериев:

* моделирование данных;
* особенности архитектуры и функциональные возможности;
* контроль работы системы;
* особенности разработки приложений;
* производительность;
* надежность;
* требования к рабочей среде.

Исходя из анализа вышеперечисленных критериев, принимая во внимание то, что необходима реляционная база данных, а так же особенности поддержки языков программирования, мной был осуществлен выбор в пользу СУБД Microsoft SQL Server.

СУБД Microsoft SQL Server является одной из наиболее популярных систем управления базами данных (СУБД) в мире. Данная СУБД подходит для самых различных проектов: от небольших приложений до больших высоконагруженных проектов. SQL Server характеризуется такими особенностями как: производительность (SQL Server работает очень быстро), надежность и безопасность (SQL Server предоставляет шифрование данных), простота (с данной СУБД относительно легко работать и вести администрирование).

Для организации баз данных MS SQL Server использует реляционную модель, которая на сегодняшний день она фактически является стандартом для организации баз данных.

Реляционная модель предполагает хранение данных в виде таблиц, каждая из которых состоит из строк и столбцов. Каждая строка хранит отдельный объект, а в столбцах размещаются атрибуты этого объекта.

Для идентификации каждой строки в рамках таблицы применяется первичный ключ (primary key). В качестве первичного ключа может выступать один или несколько столбцов. Используя первичный ключ, мы можем ссылаться на определенную строку в таблице. Соответственно две строки не могут иметь один и тот же первичный ключ.

Через ключи одна таблица может быть связана с другой, то есть между двумя таблицами могут быть организованы связи. А сама таблица может быть представлена в виде отношения («relation»).

Для взаимодействия с базой данных применяется язык T-SQL (Transact - Structured Query Language). Клиент (например, внешняя программа) отправляет запрос на языке T-SQL. СУБД должным образом интерпретирует и выполняет запрос, а затем посылает клиенту результат выполнения.

## 2.2 Построение концептуальной и логической модели БД

Модель отображает функциональную структуру объектов, то есть производимые ими действия и связи между этими действиями в виде диаграммы вариантов использования, диаграммы классов, диаграммы последовательности, диаграммы деятельности, диаграммы компонентов.

Функциональная модель данных разрабатываемого программного средства представлена в виде диаграммы вариантов использования, смоделированной на унифицированном языке моделирования UML средствами Rational Rose Enterprise Edition, которая предназначена для определения общих границ и контекста моделируемой предметной области на начальных этапах проектирования программы. [2]

Главной целью проектирования моделей является планирование разработки ПС.

Наиболее распространенным средством моделирования данных являются диаграммы «сущность-связь» (ERD), которые предназначены для графического представления моделей данных разрабатываемой программной системы и предлагают некоторый набор стандартных обозначений для определения данных и отношений между ними. С помощью этого вида диаграмм можно описать отдельные компоненты концептуальной модели данных и совокупность взаимосвязей между ними, имеющих важное значение для разрабатываемой системы.

Основными понятиями данной нотации являются понятия сущности и связи. При этом под сущностью понимается произвольное множество реальных или абстрактных объектов, каждый из которых обладает одинаковыми свойствами и характеристиками.

В этом случае каждый рассматриваемый объект может являться экземпляром одной и только одной сущности, должен иметь уникальное имя или идентификатор, а также отличаться от других экземпляров данной сущности. Связь определяется как отношение или некоторая ассоциация между отдельными сущностями.

Примерами связей могут являться родственные отношения типа «отец-сын» или производственные отношения типа «начальник-подчиненный». Другой тип связей задается отношениями «иметь в собственности» или «обладать свойством».

Исходя из исследования предметной области, можно выделить следующие сущности разработки: «Дипломные работы», «Диссертации», «Занятия», «Нагрузка», «Оценки», «Преподаватели», «Учащиеся», «Экзамены и курсовые проекты».

Для сущности «Дипломные работы» атрибутами будут являться «ФИО», «Оценка», «Дисциплина» и «ФИО руководителя». Для сущности «Диссертации», атрибутами будут являться «ФИО», «Тема» и «Дата». Для сущности «Занятия» - «Группа», «Факультет», «Кафедра». Для сущности «Нагрузка» - «ФИО», «Дисциплина» и «Часы». Для сущности «Оценки» - «ФИО студента», «ФИО преподавателя» и атрибут «Оценка». Сущность «Преподаватель», будет иметь атрибуты «ФИО», «Факультет», «Категория», «Кафедра», «Пол», «Дети», «Дата рождения», «Заработная плата», «Типы занятий», «Пароль», а так же «Руководитель курсовой», которая будет отображать является ли данный преподаватель руководителем курсовых проектов. Сущность «Учащиеся» содержит «ФИО», «Курс», «Группа», «Факультет», «Пол», «Дата рождения» и «Стипендия». «Экзамены и курсовые работы» содержит «ФИО», «Оценка», «Дисциплина», «ФИО экзаменатора», и отвечает за курсовые работы и экзамены.

Информационная модель базы данных представлена на диаграмме «Сущность-связь». Данная диаграмма представлена на рисунке 1.1.



Рисунок 2.1 – Диаграмма «Сущность-связь.

В рамках языка UML все представления о модели сложной системы фиксируются в виде специальных графических конструкций – диаграмм. В терминах языка UML определены следующие виды диаграмм: диаграмма вариантов использования, диаграмма состояний, диаграмма деятельности.

Перечень этих диаграмм представляет собой неотъемлемую часть графической нотации языка UML, сам процесс объектно-ориентированного программирования (ООП) неразрывно связан с процессом построения этих диаграмм. Совокупность построенных таким образом диаграмм содержит всю информацию, необходимую для реализации проекта сложной системы.

Варианты использования описывают не только взаимодействия между пользователями и сущностью, но также реакции сущности на получение отдельных сообщений от пользователей и восприятие этих сообщений за пределами системы.

Актер представляет собой любую внешнюю по отношению к моделируемой системе сущность, которая взаимодействует с системой и использует ее функциональные возможности для достижения определенных целей или решения частных задач.

В данной проектируемой системе в качестве актера выступает сотрудник салона проката, который служит источником воздействия на моделируемую систему.

К основным функциям разрабатываемой программы относятся следующие функции:

* осуществить работу с базой данных;
* регистрировать изменения в таблицах;
* показывать содержание таблиц.

Диаграмма вариантов использования представлена на листе 1 графической части.

## 2.3 Физическая модель БД

Физическая модель БД определяет способ размещения данных в среде хранения и способы доступа к этим данным, которые поддерживаются на физическом уровне. Исторически первыми системами хранения и доступа были файловые структуры и системы управления файлами (СУФ), которые фактически являлись частью операционных систем. СУБД создавала над этими файловыми моделями свою надстройку, которая позволяла организовать всю совокупность файлов таким образом, чтобы она выглядела как единое целое и получала централизованное управление от СУБД.

Механизмы буферизации и управления файловыми структурами не приспособлены для решения задач собственно СУБД, эти механизмы разрабатывались просто для традиционной обработки файлов, и с ростом объемов хранимых данных они стали неэффективными для использования СУБД. Тогда постепенно произошёл переход от базовых файловых структур к непосредственному управлению размещением данных на внешних носителях самой СУБД.

Соответствие основных категорий моделей БД представлено в таблице 2.1

Таблица 2.1 – Соответствие основных категорий моделей БД

|  |  |  |  |
| --- | --- | --- | --- |
| Сущность | Отношение | Таблица | Файл |
| Экземпляр | Кортеж | Строка | Запись |
| Атрибут | Домен | Столбец | Поле |

Построение физической модели БД производится на основе логической модели, представленной в подразделе 2.2.

Все данные проекта будут храниться в базе данных «Database1.mdf».

База данных соответствует реляционной модели данных, где каждый выделенный в ходе проектировании сущности соответствует таблица.

В таблице 2.2 описано соответствие сущностей логической модели и таблиц физической.

Таблица 2.2 – Соответствие сущностей логической и физической моделей

|  |  |
| --- | --- |
| Сущность | Таблица |
| Учащиеся | Учащиеся |
| Дипломные работы | Дипломные работы |
| Оценки | Оценки |
| Экзамены и курсовые работы | Экзамены и курсовые работы |
| Нагрузка | Нагрузка |
| Занятия | Занятия |
| Диссертации | Диссертации |
| Преподаватели | Преподаватели |
| Диссертации | Диссертации |

Структура базы данных разрабатываемого программного средства включает четыре таблицы. Структура данных и их краткое описание приводится в таблицах 3.2-3.5.

На рисунке 3.2 представлена схема базы данных разрабатываемого приложения.

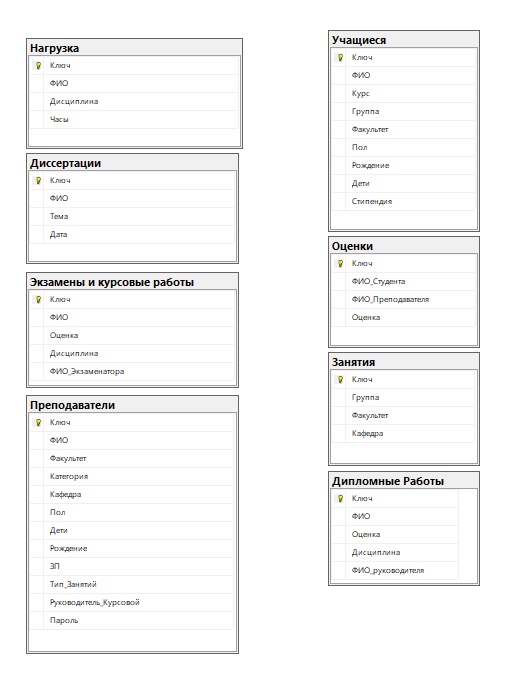


Рисунок 2.2-Схема базы данных

Таблица «Нагрузка» хранит информацию о количестве часов которые отводятся под определённый предмет. Структура данных таблицы приведена в таблице 2.3

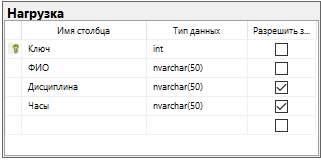


Таблица 2.3— Структура таблицы «Нагрузка»

Таблица «Преподаватели» хранит информацию о преподавателях, их личную информацию, заработную плату, а также пароли от их учётных записей. Структура приведена в таблице 2.4.

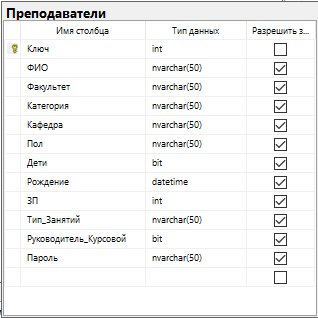


Таблица 2.4— Структура таблицы «Преподаватели»

Таблица «Дипломные работы» хранит информацию о дипломных работах, оценках за них, их исполнителей и проверяющих, структура данных приведена в таблице 2.5.

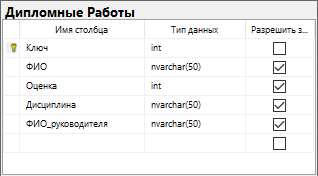


Таблица 2.5— Структура таблицы «Дипломные работы»

Таблица «Занятия» хранит информацию о занятиях, группах у которые эти занятия проводятся и более подробная информация, структура данных приведена в таблице 2.6.

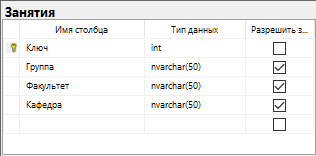


Таблица 2.6— Структура таблицы «Занятия»

Таблица «Оценки» хранит информацию о товарах, хранящихся в каталоге, которые предоставляются клиенту для выбора на прокат, структура данных для товаров приведена в таблице 2.7.

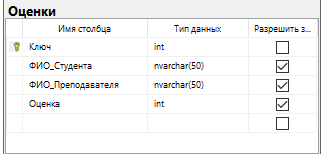


Таблица 2.7— Структура таблицы «Оценки»

Таблица «Диссертации» хранит информацию о диссертациях преподавателей, структура данных приведена в таблице 2.8.

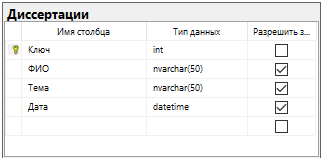


Таблица 2.8— Структура таблицы «Диссертации»

Таблица «Экзамены и курсовые работы хранит информацию о экзаменах и курсовых проектах учащихся, оценку, дисциплину и ФИО как сдающего, так и принимающего. Структура данных приведена в таблице 2.9.

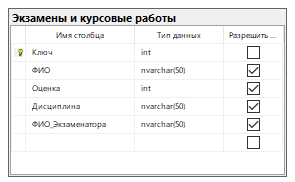


Таблица 2.9— Структура таблицы «Экзамены и курсовые работы»

Таблица «Учащиеся» хранит информацию об учащихся, их личную информацию. Структура данных приведена в таблице 2.10.

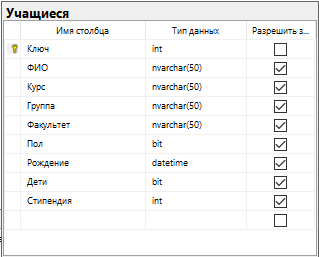


Таблица 2.10— Структура таблицы «Учащиеся»

## 2.4 Разграничение прав доступа

Права доступа (permissions) — это права, дающие воз­можность доступа к объекту (например, таблице) базы данных. Право доступа предоставляется пользователю или группе для выполнения таких функций, как выборка дан­ных, добавление но­вых строк или обновление данных.

При создании приложения базы данных часто возникает необходимость разграничить права пользователей для работы с данными. Кроме этого, ограничение доступа к базе, выполняет, по сути, еще и функцию защиты данных от несанкционированного доступа и изменения. Ведь не зарегистрированному пользователю невозможно будет открыть базу данных даже для чтения.

Право доступа косвенно предоставляется владельцу или создателю объекта. Владелец объекта может затем сам принимать решение о предоставлении прав доступа другим пользователям или группам, как он считает нужным. Существует несколько ти­пов прав доступа к объектам базы данных.

В данном курсовом проекте были созданы две группы пользователей, которые наделены разными правами, то есть им присущи разные роли.

В первой группе «Преподаватели» содержатся все преподаватели. У данной роли есть все привилегии на корректировку, удаление, восстановление, администрирование базы данных.

Во второй группе «Учащиеся» содержатся все пользователи, которые вошли без учётной записи «Преподаватель». Данные пользователи могут только просматривать базу данных.

# 3 Реализация базы данных

## 3.1 Реализация запросов к БД

В курсовом проекте реализованы разные типы запросов к базе данных:

* запрос на обновление
* запрос на выборку.

Рассмотрим для каждого типа запроса по одному примеру. Остальные однотипные запросы реализуются аналогично.

Запрос на обновление всех таблиц формируется при нажатии на форме кнопки «Сохранить»

Код запроса представлен ниже:

da.Update(ds, "MyTable");

Запрос на выборку данных из таблицы формируется при входе в программу в качестве учителя, для аутентификации.

Код запроса представлен ниже:

getInfo = "SELECT Пароль, ФИО FROM Преподаватели"

## 3.2 Программная реализация объектов БД

В рамках разработки базы данных для курсового проектирования были созданы следующие объекты базы данных:

− представления;

− хранимые процедуры и функции, объединенные в пакеты;

В таблице 3.1 описаны все представления, реализованные в базе данных

Таблица 3.1 Представления БД

|  |  |
| --- | --- |
| Название представления | Назначение представления |
| View | Вывод таблицы «Преподаватели» для учащегося (без столбца «Пароль»)  Код: CREATE VIEW [dbo].[View] AS SELECT [Ключ], [ФИО], [Факультет], [Категория], [Кафедра], [Пол], [Дети], [Рождение], [ЗП], [Тип\_Занятий], [Руководитель\_Курсовой] FROM [Преподаватели] |
| View2 | Вывод таблицы «Учащиеся». Но выводит информацию только о тех учениках, которые считаются отличниками (балл выше 7) |

В таблице 3.2 описаны хранимые процедуры и назначенные права на доступ к хранимым процедурам.

Таблица 3.2 Процедуры

|  |  |  |  |
| --- | --- | --- | --- |
| Хранимая процедура | Назначение хранимой процедуры | Права преподавателя | Учащегося |
| procedure | Демонстрация экзаменаторов без пароля (для учеников) | Недоступна | Разрешено |

Код: CREATE PROCEDURE [dbo].[Procedure]

AS

SELECT Ключ, ФИО, Факультет, Категория, Кафедра, Пол, Дети, Рождение, ЗП, Тип\_Занятий, Руководитель\_Курсовой FROM Преподаватели WHERE ([Руководитель\_Курсовой] = 1)

# 4 Применение разработанной базы данных в программном приложении

## 4.1 Общие сведения

Программное приложение имеет название «RPO\_college.exe» разработано в визуальной среде разработки Microsoft Visual Studio 2019 на языке программирования C#. Программный продукт может работать в средах операционных систем семейства Microsoft Windows, начиная с Windows 7.

Для того чтобы начать работу с программой, нужно запустить файл «RPO\_college.exe». Его объем составляет 552 килобайт. Для функционирования программы необходимо установить библиотеку .NET Framework 4.7.2 и выше.

Файл базы данных имеет название «Database1.mdf» и хранится в папке с приложением.

На рисунке 4.1 представлена структура проекта приложения.

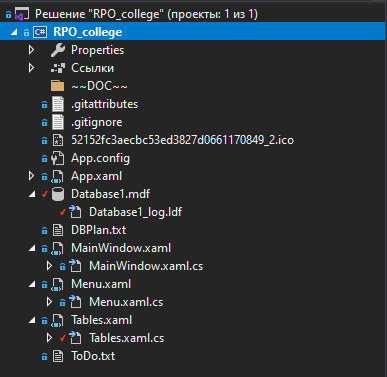


Рисунок 4.1-Проект приложения

Программное приложение не требовательно к системным ресурсам, обладает развитым интерактивным интерфейсом, просто в использовании.

## 4.2 Функции: логическая и физическая организация

На основании диаграммы вариантов использования в программном приложении на тему Разработка приложения «Колледж», были реализованы следующие функции:

* получить перечень и общее число студентов указанных групп либо указанного курса (курсов) факультета полностью, по половому признаку, году рождения, возрасту, признаку наличия детей, по признаку получения и размеру стипендии;
* 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 в указанный период;
* получить перечень и общее число тем кандидатских и докторских диссертаций, защитивших сотрудниками указанной кафедры либо указанного факультета;
* получить перечень кафедр, проводящих занятия в указанной группе либо на указанном курсе указанного факультета в указанном семестре, либо за указанный период;
* получить список и общее число преподавателей, проводивших (проводящих) занятия по указанной дисциплине в указанной группе либо на указанном курсе указанного факультета;
* получить перечень и общее число преподавателей, проводивших (проводящих) лекционные, семинарские и другие виды занятий в указанной группе либо на указанном курсе указанного факультета в указанном семестре, либо за указанный период;
* получить список и общее число студентов указанных групп, сдавших зачет либо экзамен по указанной дисциплине с указанной оценкой.
* получить список и общее число студентов указанных групп или указанного курса указанного факультета, сдавших указанную сессию на отлично, без троек, без двоек;
* получить перечень преподавателей, принимающих (принимавших) экзамены в указанных группах, по указанным дисциплинам, в указанном семестре;
* получить список студентов указанных групп, либо которым заданный преподаватель поставил некоторую оценку за экзамен по определенным дисциплинам, в указанных семестрах, за некоторый период;
* получить список студентов и тем дипломных работ, выполняемых ими на указанной кафедре либо у указанного преподавателя;
* получить список руководителей дипломных работ с указанной кафедры, либо факультета полностью и раздельно по некоторым категориям преподавателей;
* получить нагрузку преподавателей (название дисциплины, количество часов), ее объем по отдельным видам занятий и общую нагрузку в указанном семестре для конкретного преподавателя либо для преподавателей указанной кафедры.

Для организации в приложении графического пользовательского интерфейса используются элементы управления в виде кнопок, текстбоксов, ярлыков, гридов и датагридов. Пользователю предоставляется возможность просматривать, вводить/изменять и выводить данные. При нажатии на активный элемент управления, событие выполняет соответствующее действие.

Для выполнения вышеперечисленных функций необходимо запустить на компьютере пользователя программу. В окне программы отобразится главная форма программного приложения с кнопочным меню, предоставляющим пользователю доступ к основным функциям проекта.

Функция корректировки базы данных реализована с помощью кнопки «Сохранить», а также с помощью компонента DataGrid для вывода информации из базы данных на экран и её редактирования.

Рассмотрим основные функции проекта.

Функция входа обрабатывается несколькими функциями подвешенными на событие Button\_Click закрепленным за элементами управления Button. Код функций представлен ниже.

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

{

login.Visibility = Visibility.Hidden;

thcr.Visibility = Visibility.Visible;

con = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=Database1.mdf;Integrated Security=True");//подключаем БД

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

ds = new DataSet();//создаём датасет

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

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

con.Close();//закрываем подключение

IsTryLoginAsTeacher = true;

}

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

{

Button button = sender as Button;

if(button.Content.ToString() == "Войти как ученик")

{

IsTryLoginAsTeacher = false;

IsTeacher = false;

}

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();

}

}

Функция кнопки «Назад» на форме MainWindow лежит на кнопке «Назад», на событии Button\_Back(), и производит возвращение на главную форму (данная кнопка появляется при вводе пароля и логина при попытке войти как преподаватель).

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

{

thcr.Visibility = Visibility.Hidden;

login.Visibility = Visibility.Visible;

IsTryLoginAsTeacher = false;

}

Функции всех кнопок с формы Menu лежат на обработчике события Button\_Click(). Они выполняют следующие функции: возврат на главную форму, выход из программы и открытие следующей формы.

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;

}

}

Функция которая заполняет столбцы вызывается автоматически при открытии формы.

private void MyTable\_AutoGeneratingColumn(object sender, DataGridAutoGeneratingColumnEventArgs e)//ЭТО МОЖЕТ ВСЁ СЛОМАТЬ

{

string headername = e.Column.Header.ToString();

if (headername == "Ключ")

{

e.Cancel = true;

}

}

Функция, которая возвращает на форму назад, подвешена на события button\_return, и активируется при нажатии кнопки «Назад» с формы tables.

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

{

Menu window = new Menu();

window.Show();

this.Close();

}

Функция, которая активируется при закрытии программы подвешена на событие 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);

}

}

Функция, которая сохраняет изменённую таблицу висит на кнопке «Сохранить» и подвешена на событие Button\_save.

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

{

if (MainWindow.IsTeacher)

{

da.Update(ds, "MyTable");

MessageBox.Show("Saved");

}

else

{

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

}

## 4.3 Входные и выходные данные

В разработанном программном средстве входные данные носят разнообразный характер и зависят от выбора пользователя. Так, для входа указывается информация, представленная в таблице 4.1

Таблица 4.1 – Структура входных данных для входа в программу.

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип поля | Размерность | Наименование  (смысл отображаемой информации) |
| Логин | string | 255 | Логин |
| Пароль | string | 255 | Пароль |

Входными данными является получение доступа к следующей форме с правами преподавателя.

Входными данными для таблиц является изменённая пользователем таблица.

Выходными данными для таблиц является неизменённая пользователем таблица.

Выходные данные разнообразны и зависят от уровня доступа пользователя.

## 4.4 Демонстрационный пример использования

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

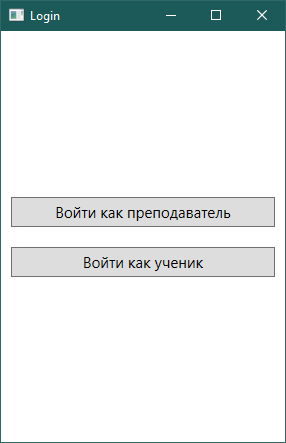


Рис 4.2 – форма «Login»

При нажатии на кнопку «Войти как преподаватель», содержимое формы изменится на представленное на Рис 4.3.

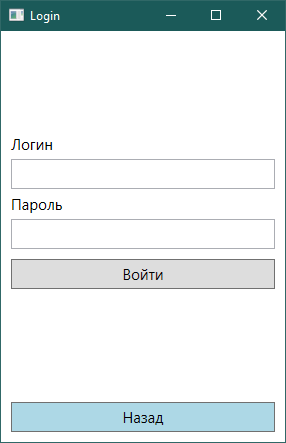


Рис 4.3 - форма «Login» после нажатия на кнопку «Войти как преподаватель»

После ввода корректного логина и пароля, происходит переход на форму, представленную на рисунке 4.4.

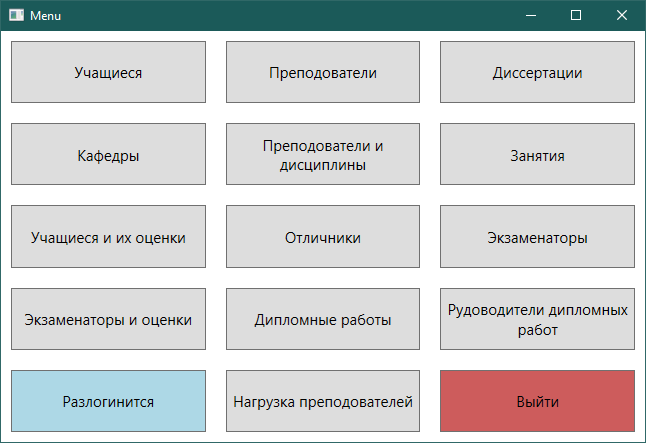


Рис 4.4 – форма «Menu»

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 4.5

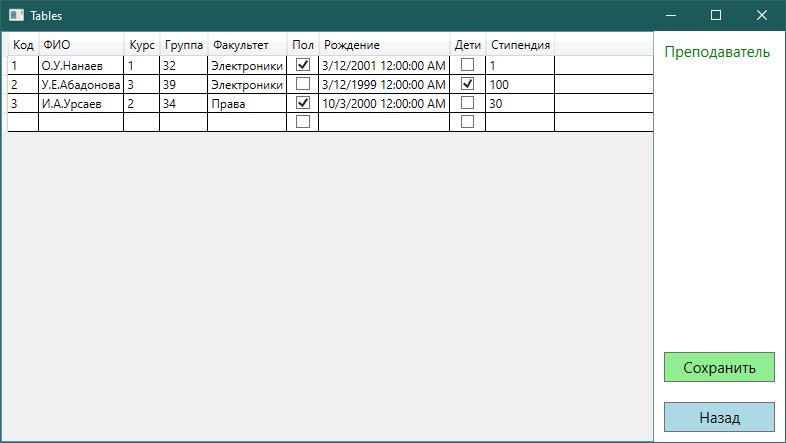


Рис 4.5 – форма «Tables»

В случае, если пользователь произвёл вход в программу как «Ученик», то форма будет иметь вид, представленный на рисунке 4.6

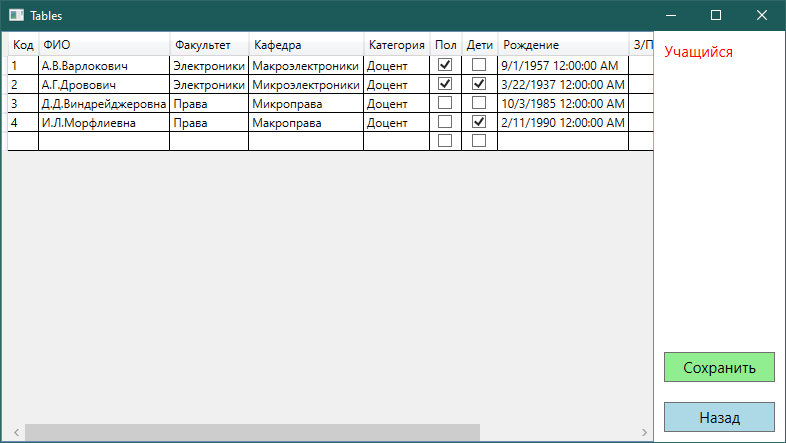


Рис 4.6 – форма «Tables» под ученическим аккаунтом

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 MSSQL, 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.
6. GitHub.com.
7. Metanit.com.
8. Youtube.com.
9. MSDN.Microsoft.com.
10. eservice.by/1s-kolledzh.
11. Описание программы: ГОСТ 19.402-2000 ЕСПД. – М. : Изд-во стандартов, 2000
12. Текст программы.: ГОСТ 19.401-2000. ЕСПД. – М. : Изд-во стандартов, 2000

# ПРИЛОЖЕНИЕ А

(справочное)

Фрагменты текста программы

using System.Data;

using System.Data.Sql;

using System.Windows;

using System.Windows.Controls;

using System.Linq;

using System.Data.SqlClient;

namespace RPO\_college

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

string getInfo;

DataSet ds;

SqlConnection con;

SqlDataAdapter 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 SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=Database1.mdf;Integrated Security=True");//подключаем БД

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

ds = new DataSet();//создаём датасет

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

da.Fill(ds,"Преподаватели");//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

{

Button button = sender as Button;

if(button.Content.ToString() == "Войти как ученик")

{

IsTryLoginAsTeacher = false;

IsTeacher = false;

}

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 menu = new Menu();

menu.Show();

this.Close();

return;

}

}

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

}

}

else

{

Menu menu = new Menu();

menu.Show();

this.Close();

}

}

}

}

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 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;

using System.Data;

using System.Data.SqlClient;

using System.Windows;

using System.Windows.Controls;

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 SqlDataAdapter da;

private readonly DataSet ds;

private SqlCommandBuilder cmd;

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

private readonly SqlConnection DataBase = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\boris\Source\Repos\olegzig\RPO\_college\Database1.mdf;Integrated Security=True");

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 SqlDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Учащиеся]");

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

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

break;

case "Преподаватели":

if (MainWindow.IsTeacher)

{

getInfo = "SELECT \* FROM Преподаватели";

}

else

{

getInfo = "select \* from [View]";

}

da = new SqlDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Преподаватели]");

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

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

break;

case "Диссертации":

getInfo = "SELECT \* FROM Диссертации";

da = new SqlDataAdapter(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 \* FROM Занятия";

da = new SqlDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Занятия]");

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

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

break;

case "Преподаватели и\nдисциплины":

getInfo = "SELECT \* FROM Преподаватели";

da = new SqlDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Преподаватели]");

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

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

break;

case "Занятия":

getInfo = "SELECT \* FROM Занятия";

da = new SqlDataAdapter(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 SqlDataAdapter(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 [View2]";

da = new SqlDataAdapter(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 Преподаватели WHERE ([Руководитель\_Курсовой] = 1)";

}

else

{

getInfo = "exec [Procedure]";//тут мб из-за бита будет ошибка

}

da = new SqlDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Преподаватели]");

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

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

break;

case "Экзаменаторы и оценки":

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

da = new SqlDataAdapter(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 \* from [Дипломные Работы]";

da = new SqlDataAdapter(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\* FROM[Дипломные работы]";

da = new SqlDataAdapter(getInfo, DataBase);

ds = new DataSet();

da.Fill(ds, "[Дипломные работы]");

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

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

break;

case "Нагрузка преподавателей":

getInfo = "SELECT \* FROM Нагрузка";

da = new SqlDataAdapter(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;

}

cmd = new SqlCommandBuilder(da);

}

private void MyTable\_AutoGeneratingColumn(object sender, DataGridAutoGeneratingColumnEventArgs e)//ЭТО МОЖЕТ ВСЁ СЛОМАТЬ

{

string headername = e.Column.Header.ToString();

if (headername == "Ключ")

{

e.Cancel = true;

}

}

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)

{

da.Update(ds, "MyTable");

MessageBox.Show("Saved");

}

else

{

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

}

}

private void RemoveNullColumnFromDataset()//пусть пока висит

{

for (int i = ds.Tables[0].Rows.Count - 1; i >= 0; i--)

{

if (ds.Tables[0].Rows[i][1] == DBNull.Value)

ds.Tables[0].Rows[i].Delete();

}

ds.AcceptChanges();

}

}

}

//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"/>

\*/

# ПРИЛОЖЕНИЕ Б

ДИАГРАММА ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ

ДИАГРАММА СОСТОЯНИЙ

ДИАГРАММА ДЕЯТЕЛЬНОСТИ



*Разработка приложения «Колледж»*

МГКЭ КП38ТП.011109.081 ПЗ

Вайтович И.М

*38*

Жгуновский О.Б

Вайтович И.М

*CD-RW*

Утв.

Н. контр.

Т. контр.

Провер.

Разраб.

*УО «МГКЭ»*