Министерство науки и высшего образования Российской Федерации

ФГАОУ ВО «Уральский федеральный университет   
имени первого Президента России Б.Н. Ельцина»

Институт новых материалов и технологий

Кафедра «Теплофизика и информатика в металлургии»

**Курсовая работа**

на тему «Разработка программы для контроля доступа в организацию на платформе Windows Forms»

**по дисциплине «Теория алгоритмов»**

Образовательная программа бакалавриата   
09.03.02/03.01 «Информационные системы и

технологии в металлургии» (ФГОС ВО)

Руководитель

к.т.н., доцент И.А. Гурин

звание, должность подпись расшифровка подписи

Студент

НМТ-293907 Е.Ю.Мазлова

номер группы подпись расшифровка подписи

Екатеринбург

2020

ФГАОУ ВО «Уральский федеральный университет   
имени первого Президента России Б.Н. Ельцина»

Институт новых материалов и технологий

Кафедра «Теплофизика и информатика в металлургии»

УТВЕРЖДАЮ:

Зав. кафедрой\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (Н.А. Спирин)

«20» сентября 2020 г.

**Задание**

**на выполнение курсовой работы**

**по дисциплине «Теория алгоритмов»**

Студент группы **НМТ-293907**. Направление 09.03.02 «Информационные системы и технологии» (уровень бакалавриата). ОП 09.03.02/03.01 «Информационные системы и технологии в металлургии» (ФГОС)

Фамилия: *Мазлова* Имя: *Елена* Отчество: *Юрьевна*

Руководитель работы: к.т.н., доцент Гурин И.А.

Срок выполнения работы: с «20» сентября 2020 г. по «31» декабря 2020 г.

1. Тема курсовой работы:

***Разработка программы для контроля доступа в организацию на платформе Windows Forms***

2. Содержание работы

2.1. Пояснительная записка:

* титульный лист;
* бланк задания преподавателя на выполнение курсовой работы;
* оглавление;
* постановка задачи, требования пользователей к системе;
* архитектура системы;
* схема базы данных;
* библиотека классов;
* пользовательские формы для работы с базой данных;
* заключение;
* библиографический список;

2.2. Требования к приложению

* загрузка и отображение данных из базы данных Microsoft Access на пользовательской форме. Фильтрация данных на пользовательской форме;
* возможность сопровождения базы данных (вставка, корректировка и удаление данных) из программного приложения через пользовательские формы;
* контроль корректности вводимых данных на пользовательских формах. Сообщения об ошибках ввода;
* формирование графика (диаграммы) в программном приложении;
* формирование отчета в программном приложении с возможностью его предварительного просмотра;
* наличие файла справочной помощи по работе с приложением.

2.3. Компьютерные версии

* реализация: приложение – в среде Microsoft Visual Studio (проект с пользовательской формой и библиотекой классов); базы данных – в СУБД Microsoft Access.

3. К защите предоставляются следующие материалы

* программная реализация;
* пояснительная записка (ПЗ), выполненная в соответствии с требованиями по оформлению курсовых и дипломных работ.

4. Выполнение курсовой работы закончено «15» января 2021 г.

Руководитель работы, доцент, к.т.н. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (И.А. Гурин)

Задание к исполнению принял \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (Е.Ю.Мазлова)

# **СОДЕРЖАНИЕ**

[Введение 3](#_Toc59800505)

[1 Постановка задачи. Требования пользователей к информационной системе 4](#_Toc59800506)

[2 Архитектура построения информационной системы 5](#_Toc59800507)

[3 Разработка и описание базы данных 6](#_Toc59800508)

[3.1 Разработка схемы базы данных 6](#_Toc59800509)

[3.2 Наполнение базы данных 9](#_Toc59800510)

[4 Разработка и описание клиентского приложения 11](#_Toc59800511)

[4.1 Интерфейс взаимодействия базы данных с клиентским приложением 11](#_Toc59800512)

[4.2 Разработка и описание работы клиентского приложения 11](#_Toc59800513)

[Заключение 23](#_Toc59800514)

[Библиографический список 24](#_Toc59800515)

[Приложение. Фрагмент листинга программного кода 26](#_Toc59800516)

# **Введение**

Язык программирования C# — объектно-ориентированный язык программирования, который был 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. C# относится к семье языков с C-подобным синтаксисом, 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. Всё это и стало главной причиной выбора данного языка программирования для написания данной курсовой работы.

В современном мире существует большое количество различных организаций, одним из главных приоритетов которых является обеспечение безопасности сотрудников и защита их личной информации. Для этих целей существует множество программ, содержащих личную информацию о сотрудниках компаний и контролирующих их передвижения по офисам и другим рабочим помещениям данного предприятия. В нашем случае, требуется создать программу, которая позволит отслеживать передвижения работников некой организации, а также предоставит пользователю возможность просмотра и редактирования информации о сотрудниках и рабочих помещениях предприятия.

# **1 Постановка задачи. Требования пользователей к информационной системе**

Требуется создать программную среду, предназначенную для специалистов безопасности предприятия, без возможности доступа в систему посторонних лиц. Данная разработка позволит мониторить все передвижения сотрудников организации, а также просматривать личную информацию работников (ФИО, дата рождения, телефон, адрес электронной почты), занимаемые ими должности и информацию о помещениях компании (название, площадь, адрес).

Принцип работы программы: с неких устройств контроля доступа (турникеты, либо двери со специальными замками) в программу поступают сведения о доступе в помещения работников компании. Программа позволит просматривать, добавлять и редактировать информацию о перемещении работников, их личные данные, занимаемые должности и информацию о помещениях организации, а также формировать отчет и диаграмму.

# **2 Архитектура построения информационной системы**

Архитектура системы контроля реализована в среде Microsoft Visual Studio 2019. Для более простого восприятия ее можно представить в виде простой блок-схемы (рисунок 2.1).

Взаимодействие пользователя с программой осуществляется по средствам графического интерфейса. Работа начинается с окна авторизации пользователя, без возможности регистрации нового пользователя (для защиты хранящихся в приложении данных). Интерфейс позволяет просматривать, добавлять и редактировать содержащиеся там данные, открывать новые формы отображения информации. Программа отображает информацию не только в виде таблиц с возможностью фильтрации данных, но и выводит ее в виде диаграммы и создает отчет.

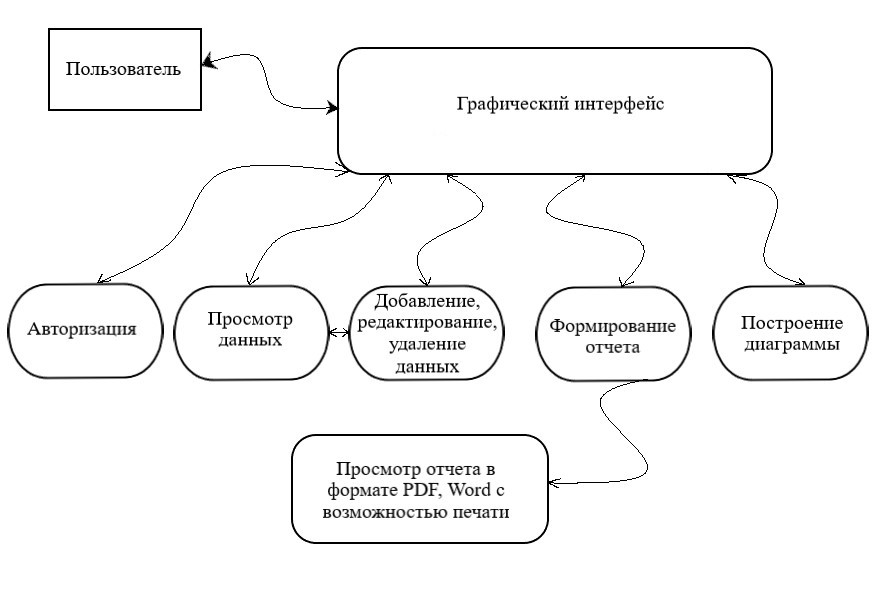


Рисунок 2.1 – Архитектура информационной системы

# **3 Разработка и описание базы данных**

**3.1 Разработка схемы базы данных**

В курсовой работе с помощью Microsoft Access была разработана модель данных с набором схем отношений, с указанием первичных ключей, а также связей между отношениями, представляющих собой внешние ключи (рисунок 3.1).

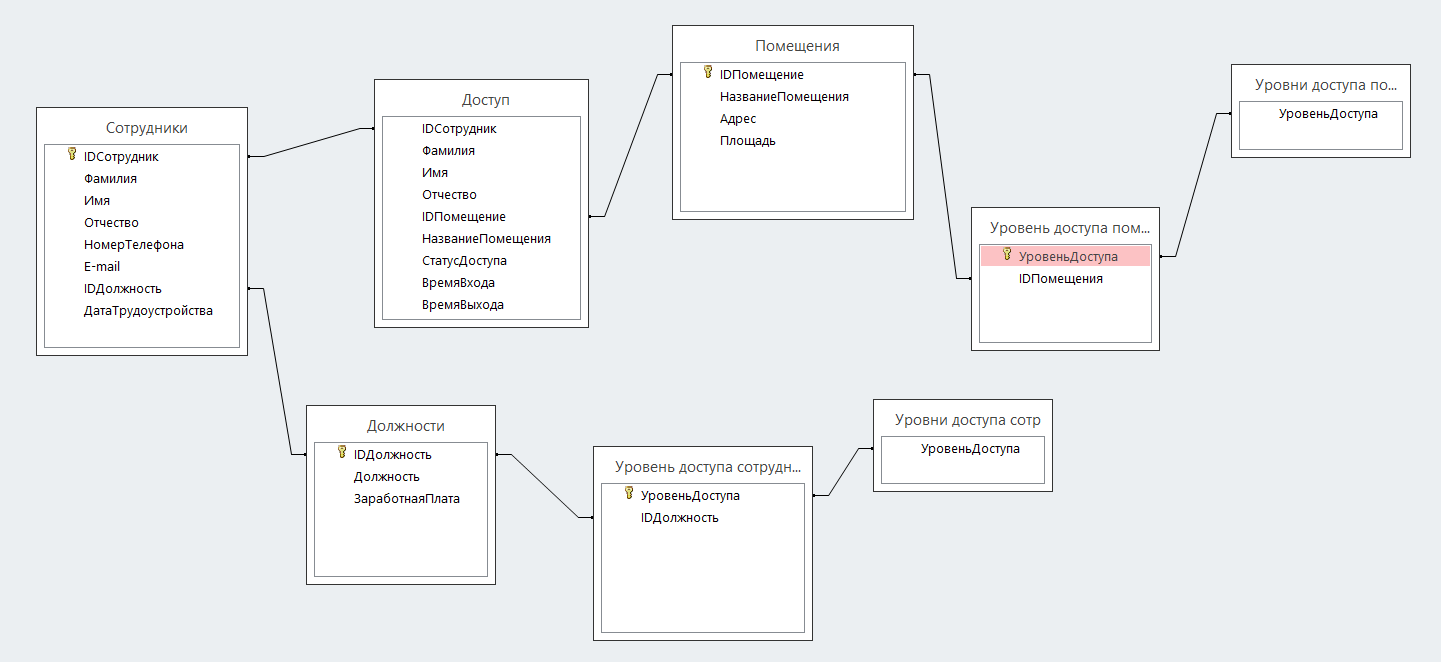


Рисунок 3.1 – Схема разработанной базы данных

В соответствие с заданием, была создана база данных, содержащая следующие таблицы: Доступ, Сотрудники, Должности, Помещения, Уровень доступа сотрудники, Уровень доступа помещения, Уровни доступа сотр, Уровни доступа помещ. Структура таблиц базы данных представлена в таблицах 3.1-3.8.

Таблица 3.1 – Структура таблицы «Помещения»

|  |  |  |
| --- | --- | --- |
| **Наименование столбца** | **Тип данных** | **Примечание** |
| IDПомещение | int | PK |
| НазваниеПомещения | string |  |
| Адрес | string |  |
| Площадь | string |  |

В таблице «Помещения» хранится информация об идентификационном номере помещения (столбец IDПомещения), названии помещения (столбец НазваниеПомещения), его местоположении (столбец Адрес) и размерах (столбец Площадь).

Таблица 3.2 – Структура таблицы «Доступ»

|  |  |  |
| --- | --- | --- |
| **Наименование столбца** | **Тип данных** | **Примечания** |
| IDСотрудник | int | FK |
| Фамилия | string |  |
| Имя | string |  |
| Отчество | string |  |
| IDПомещение | int | FK |
| НазваниеПомещения | string |  |
| СтатусДоступа | string |  |
| ВремяВхода | datetime |  |
| ВремяВыхода | datetime |  |

В таблице «Доступ» хранится информация об идентификационных номерах сотрудников компании и служебных помещений (столбцы IDСотрудник и IDПомещение), информация о сотрудниках (столбцы Фамилия, Имя, Отчество), название рабочего помещения (столбец НазваниеПомещения), статус доступа сотрудника (столбец СтатусДоступа), а также дата и время входа и выхода каждого сотрудника (столбцы ВремяВхода и ВремяВыхода).

Таблица 3.3 – Структура таблицы «Должности»

|  |  |  |
| --- | --- | --- |
| **Название столбца** | **Тип данных** | **Примечание** |
| IDДолжность | int | PK |
| Должность | string | FK |
| ЗаработнаяПлата | int |  |

В таблице «Должности» находится информация об идентификационном номере должности (столбец IDДолжность), наименовании должности (столбец Должность) и заработной плате, соответствующей каждой должности (столбец ЗаработнаяПлата).

Таблица 3.4 – Структура таблицы «Сотрудники»

|  |  |  |
| --- | --- | --- |
| **Наименование столбца** | **Тип данных** | **Примечания** |
| IDСотрудник | int | PK |
| Фамилия | string |  |
| Имя | string |  |
| Отчество | string |  |
| НомерТелефона | string |  |
| E-mail | string |  |
| IDДолжность | int | FK |
| ДатаТрудоустройства | date |  |

В таблице «Сотрудники» содержится личная информация о сотрудниках организации: идентификационный номер сотрудника и занимаемой им должности (столбцы IDСотрудник и IDДолжность), личные данные сотрудника (столбцы Фамилия, Имя, Отчество), контактная информация (столбцы НомерТелефона и E-mail) и дата устройства на работу (столбец ДатаТрудоустройства).

Таблица 3.5 – «Уровень доступа сотрудники»

|  |  |  |
| --- | --- | --- |
| **Наименование столбца** | **Тип данных** | **Примечание** |
| УровеньДоступа | string | K |
| IDДолжность | int | FK |

В таблице «Уровень доступа сотрудники» хранится информация об уровнях доступа сотрудников (столбец УровеньДоступа) соответствующих занимаемой должности (столбец IDДолжность).

Таблица 3.6 – Структура таблицы «Уровень доступа помещения»

|  |  |  |
| --- | --- | --- |
| **Наименование столбца** | **Тип данных** | **Примечание** |
| УровеньДоступа | string | K |
| IDПомещение | int | FK |

В таблице «Уровень доступа помещения» находятся данные об уровнях доступа помещений организации (столбец УровеньДоступа), соответствующий каждому служебному помещению (столбец IDПомещение).

Таблица 3.7 – Структура таблицы «Уровни доступа сотр»

|  |  |  |
| --- | --- | --- |
| **Наименование столбца** | **Тип данных** | **Примечание** |
| УровеньДоступа | string | FK |

Таблица «Уровни доступа сотр» является справочной и содержит уровни доступа, соответствующие количеству сотрудников организации.

Таблица 3.8 – Структура таблицы «Уровни доступа помещ»

|  |  |  |
| --- | --- | --- |
| **Наименование столбца** | **Тип данных** | **Примечание** |
| УровеньДоступа | string | FK |

Таблица «Уровни доступа помещ» как и предыдущая таблица является справочной и содержит информацию об уровнях доступа всех рабочих помещений предприятия.

**3.2 Наполнение базы данных**

Наполнение базы данных осуществлялось как с помощью приложения Microsoft Access, так и с помощью внутренних средств, разработанного мной приложения. Наполнение таблиц базы данных представлено на рисунках 3.2-3.9.

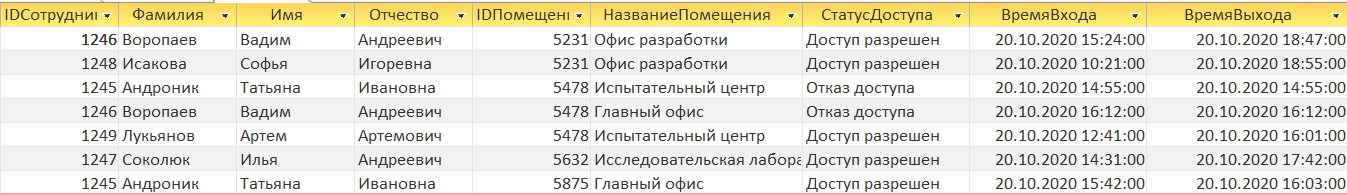


Рисунок 3.2 – Наполнение таблицы «Доступ»

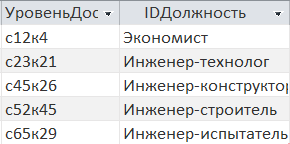


Рисунок 3.3 – Наполнение таблицы «Уровень доступа сотрудники»

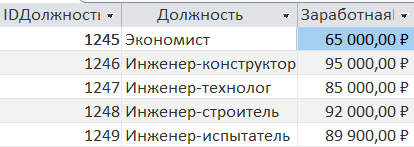


Рисунок 3.4 – Наполнение таблицы «Должности»

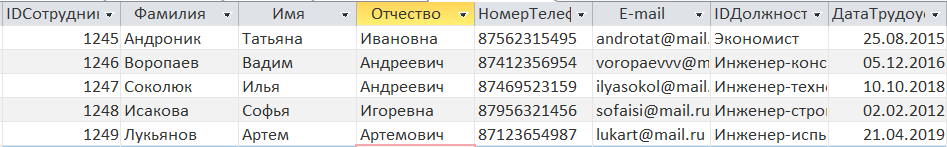


Рисунок 3.5 – Наполнение таблицы «Сотрудники»

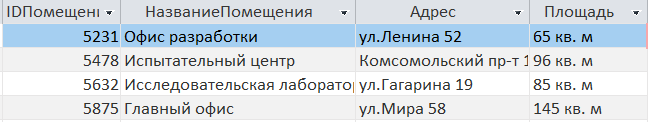


Рисунок 3.6 – Наполнение таблицы «Помещения»

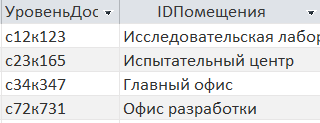


Рисунок 3.7 – Наполнение таблицы «Уровень доступа помещения»

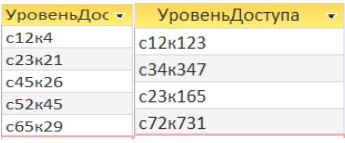


Рисунок 3.8 – Наполнение таблиц «Уровни доступа сотрудники»

# **4 Разработка и описание клиентского приложения**

**4.1 Интерфейс взаимодействия базы данных с клиентским приложением**

Взаимодействие приложения с созданной базой данных Microsoft Access осуществляется с помощью ADO.NET DataSet, который является одним из компонентов рабочего пространства Visual Studio 2019. ADO.NET предоставляет возможность доступа и управления данными, хранящимися в базе данных или других источниках, основанных на платформе .NET Framework. DataSet представляет полный набор данных, включая таблицы с данными, а также связи между таблицами.

Подключение базы данных выполнено через пространство имен System.Data.OleDb. Строка подключения создается через переменную для удобства пользования программой с других устройств (рисунок 4.1).

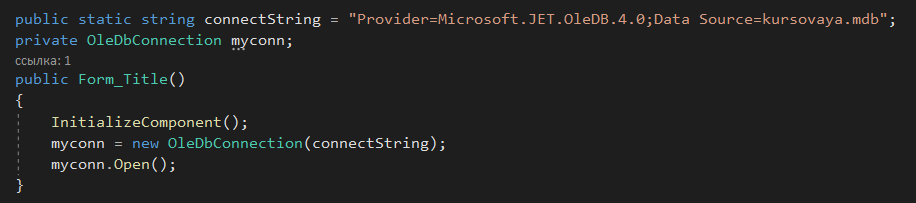


Рисунок 4.1 – Подключение базы данных к проекту

**4.2 Разработка и описание работы клиентского приложения**

Работа в приложении начинается с формы авторизации (рисунок 4.2).

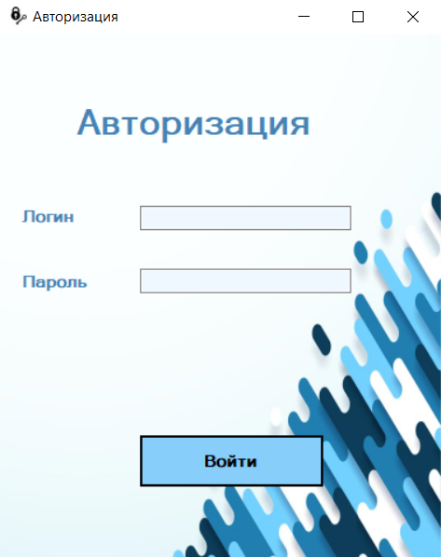


Рисунок 4.2 – Форма авторизации

Форма не имеет возможности регистрации для обеспечения безопасности данных приложения. Для этой формы была создана отдельная база данных (рисунок 4.3), в которой хранятся данные логина и пароля для входа.



Рисунок 4.3 – Таблица базы данных «Auth»

На форме были использованы стандартные элементы, такие как label, textbox, button. Если пользователь вводит неверный логин или пароль, программа дает об этом знать: на форме загорается ярко-красная надпись - «Неверный логин и(или) пароль!» (рисунок 4.4).

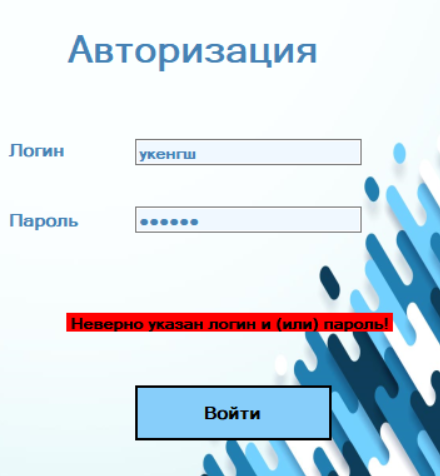


Рисунок 4.4 – Ошибка неверного ввода пароля

Обработка ошибки неверного ввода была выполнена с помощью оператора цикла while (предназначен для организации циклического процесса, в котором выполнение каждой следующей итерации определяется на основе истинности некоторого условия) и конструкции if/else if (проверяет истинность некоторого условия и в зависимости от результатов проверки выполняет определенный код; в качестве условий выступают операции сравнения). Пример кода приведен на рисунке 4.5.

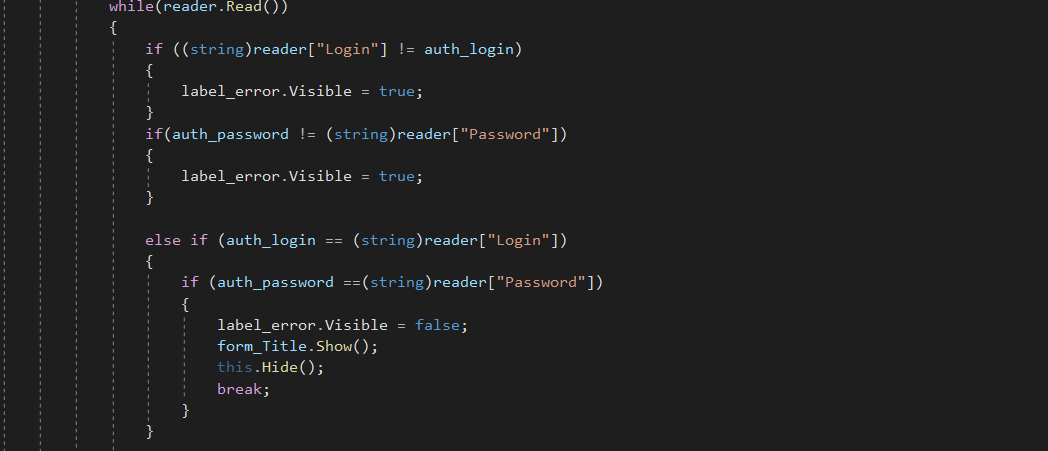


Рисунок 4.5 – Пример кода программы

После успешной авторизации, программа перенаправляет пользователя на главную форму (рисунок 4.6). Для разработки дизайна главной формы использовались такие стандартные инструменты, как menustrip, tabcontrol, datagrigview, button, groupbox, checkbox, textbox, label. Основа главной формы – элемент управления tabcontrol (позволяет создать элемент управления с несколькими вкладками; каждая вкладка будет хранить некоторый набор других элементов управления, как кнопки, текстовые поля и др.). Он содержит 4 вкладки с таблицами данных и фильтрацией.

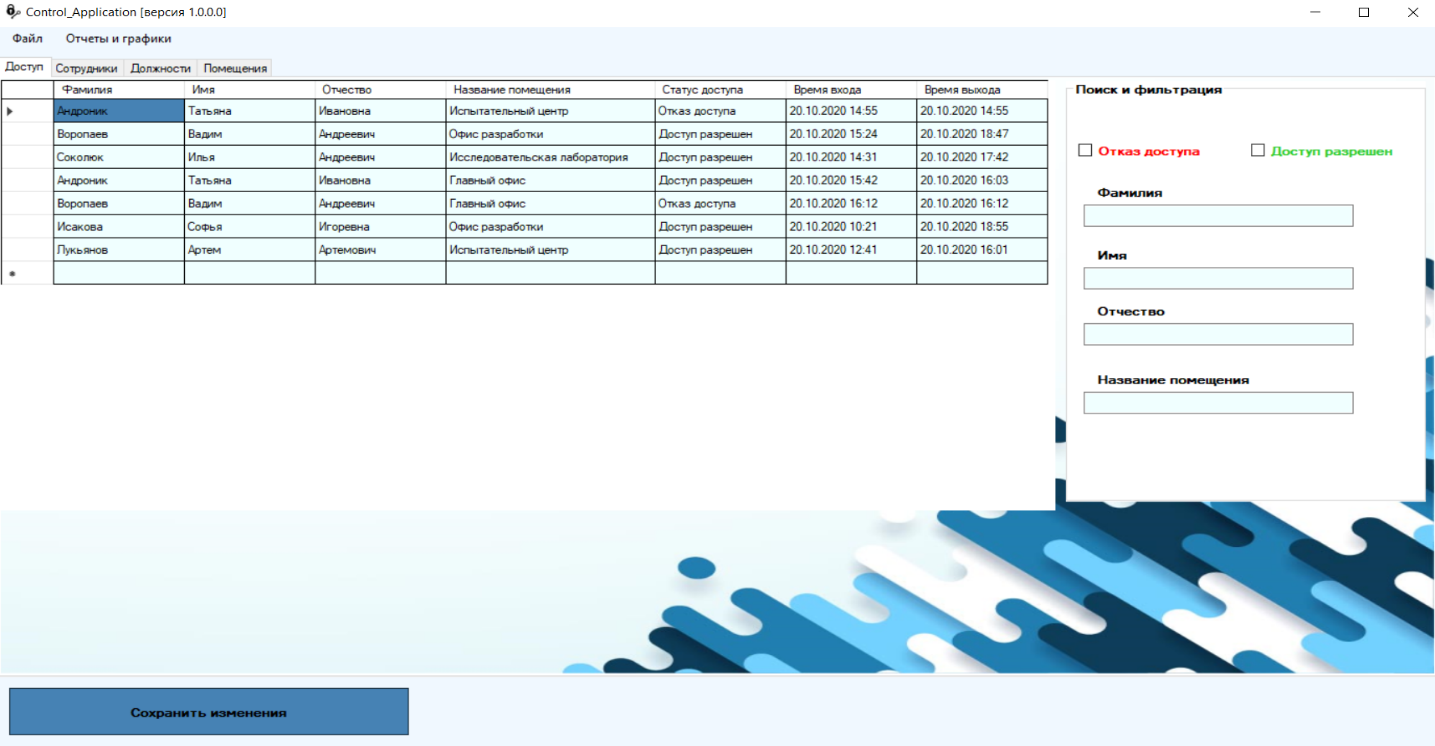


Рисунок 4.6 – Главная форма приложения

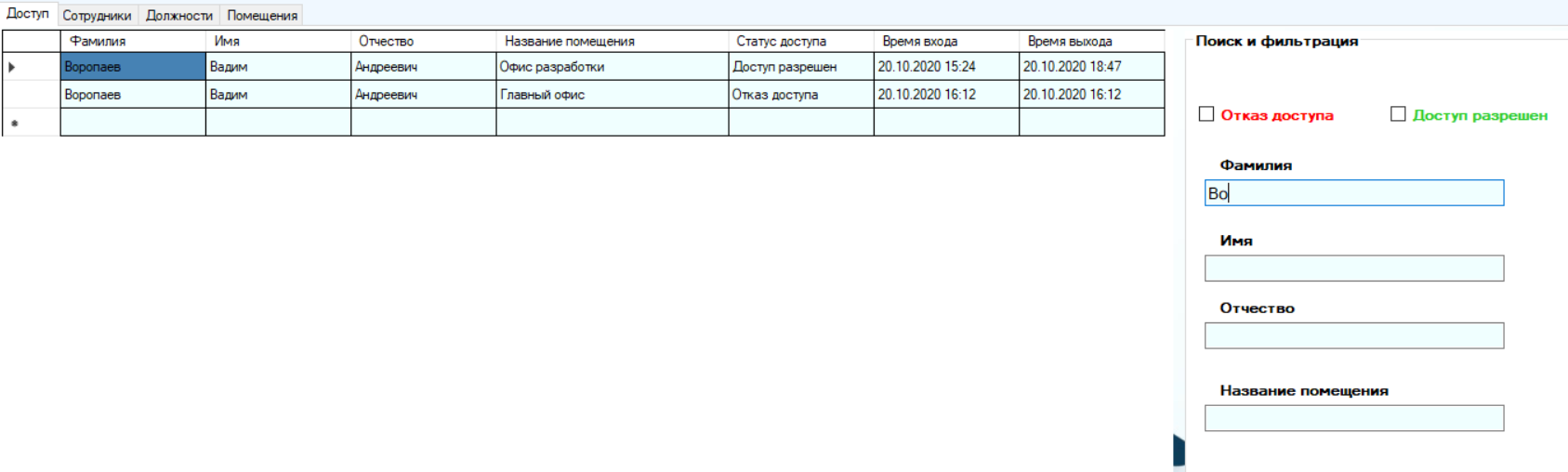
Первая вкладка называется «Доступ». На вкладке расположен datagridview, содержащий таблицу с данными о передвижении сотрудников. В данном элементе отключены возможности редактирования, удаления и добавления данных, так как он носит информацию, поступающую только со внешних источников. Справа от таблицы расположен groupbox «Поиск и фильтрация», содержащий texbox, combobox и label. Данный элемент управления помогает осуществлять поиск и фильтрацию по следующем параметрам: Фамилия, Имя, Отчество сотрудника, Название помещения, куда был совершен вход. Фильтрация через textbox выполнена с помощью метода string BindingSource.Filter, позволяющего отфильтровать значения в таблице по заданному параметру (рисунки 4.7-4.10). 

Рисунок 4.7 – Результат фильтрации данных по параметру «Фамилия»

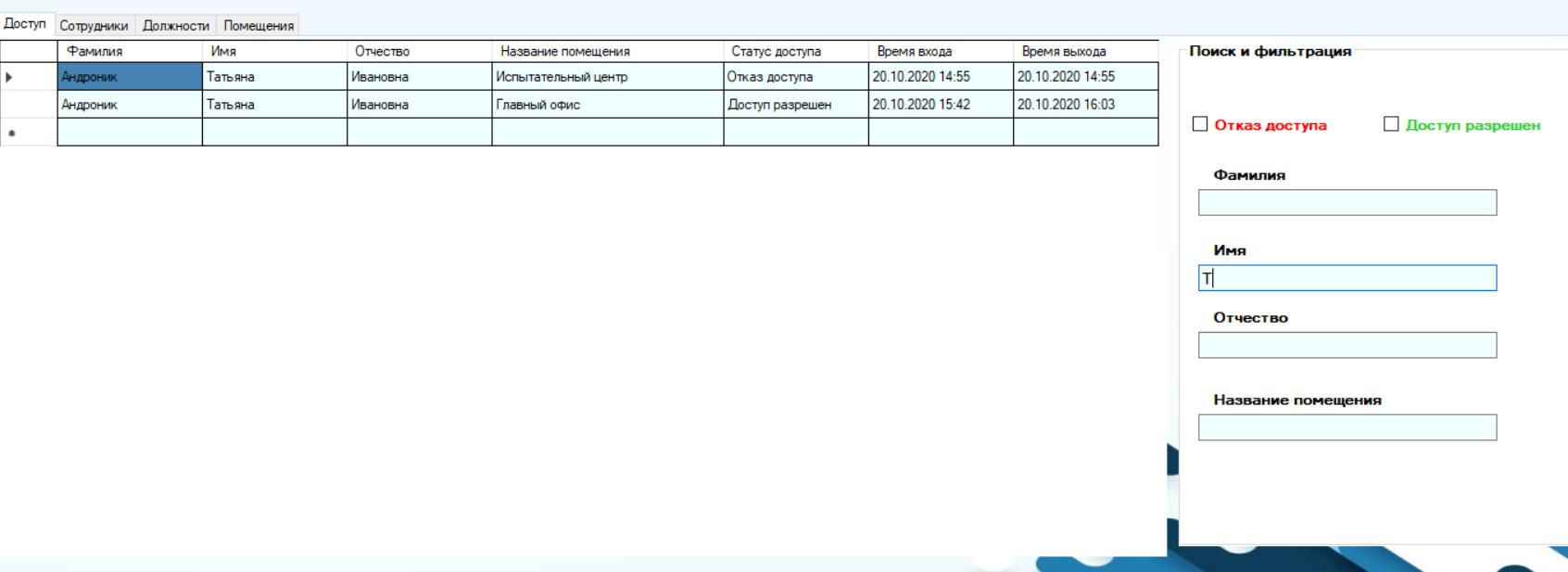


Рисунок 4.8 – Результат фильтрации по параметру «Имя»



Рисунок 4.9 – Результат фильтрации по параметру «Отчество»



Рисунок 4.10 – Результат фильтрации по параметру «Название помещения»

Фильтрация данных по параметру «Статус доступа» осуществляется с помощью элементов combobox, а также по средствам оператора цикла foreach (предназначен для перебора элементов коллекции или массива) и конструкции if. Если в столбце «Статус доступа» содержится значение «Доступ разрешен», строка данных с этим значением подсвечивается зеленым. Если же значение в столбце – «Отказ доступа», строка выделяется красным (рисунок 4.11).



Рисунок 4.11 – Результат выполнения фильтрации по параметру «Статус доступа»

Вторая вкладка tabcontrol – «Сотрудники» – включает datagridview, содержащий таблицу с личной информацией о сотрудниках организации и groupbox с textbox и label, для осуществления фильтрации данных (рисунок 4.12). Фильтрация данных выполнена по тому же методу Filter, что и во вкладке «Доступ».



Рисунок 4.12 – Вкладка «Сотрудники»

Вкладка «Должности» содержит два datagridview: первый – информация о должностях и соответствующих окладах, второй – информация об уровнях доступа с возможностью выбора уровня доступа для каждой должности (рисунок 4.13).

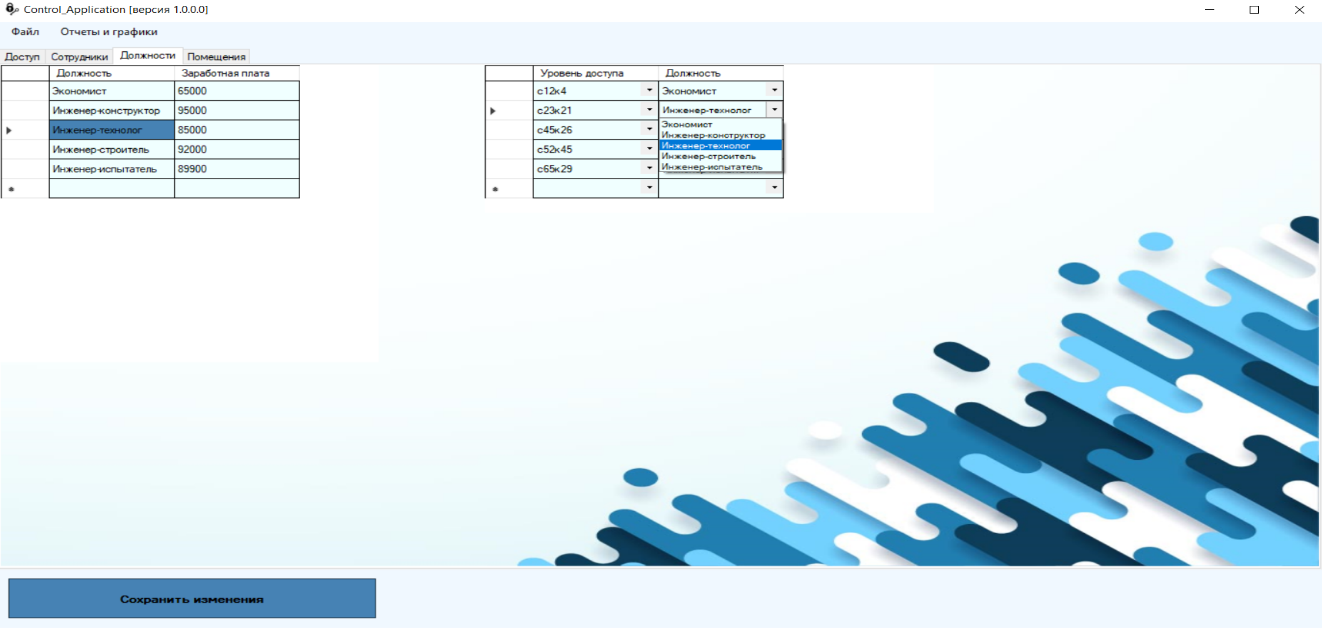


Рисунок 4.13 – Вкладка «Должности»

Последняя вкладка tabcontrol «Помещения» включает в себя два datagridview: в первом находится таблица с информацией о помещениях организации, второй же содержит таблицу с информацией о возможных уровнях доступа с возможностью выбора для каждого помещения (рисунок 4.14).

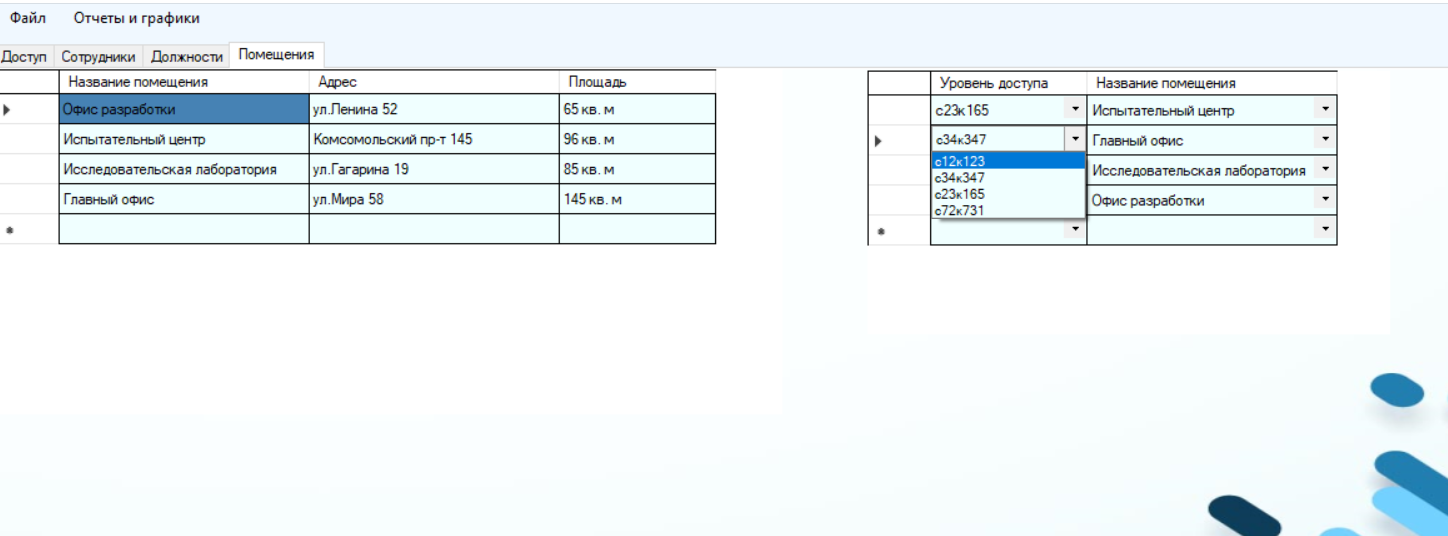


Рисунок 4.14 – Вкладка «Помещения»

В левом нижнем углу главной формы расположена кнопка «Сохранить изменения». Она позволяет сохранить внесенные во все таблицы формы изменения, так как расположена на общей форме и не привязана к определенной вкладке. Для сохранения данных следует с начала заполнить все ключевые поля таблиц, в противном случае программа выдаст исключение и сообщит о необходимости заполнения. Кнопка работает в результате привязки к ней события Click и обработки исключений с помощью блоков try-catch (вначале выполняются все инструкции в блоке try. Если в этом блоке не возникло исключений, то блок catch игнорируется и конструкция try-catch завершает свою работу. Если же в блоке try вдруг возникает исключение, то обычный порядок выполнения останавливается, и среда начинает искать блок catch, который может обработать данное исключение. Если нужный блок catch найден, то он выполняется. Если нужный блок catch не найден, то при возникновении исключения программа аварийно завершает свое выполнение). Внутри блока try прописывается заполнение и обновление данных всех таблиц приложения, в блоке catch прописывается вывод MessageBox с инструкцией устранения возникших ошибок сохранения данных. Фрагмент кодирования кнопки представлен на рисунке 4.15.

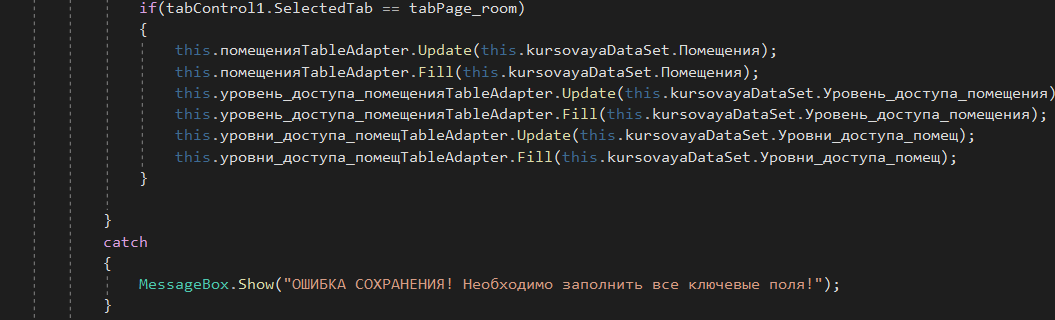


Рисунок 4.15 – Фрагмент кодирования кнопки сохранения

Одна из возможностей разработанной мной программы – отображение данных в виде диаграммы и формирование отчета. Эти возможности достигаются по средствам вывода на экран новых форм. Открыть новые формы можно с помощью стандартного элемента menustrip в верхней части главной формы (рисунок 4.16).

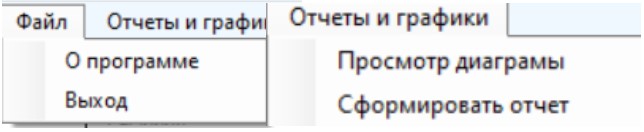


Рисунок 4.16 – Элемент главной формы MenuStrip

MenuStrip содержит две команды: «Файл», содержащий информацию о программе и предоставляющий возможность выхода из приложения, и «Отчеты и графики», с помощью которой можно перейти в новые формы для просмотра диаграммы и формирования отчета.

Форма «Просмотр диаграммы» (рисунок 4.17) открывается с помощью нажатия на одноименную кнопку в menustrip. «Просмотр диаграммы» содержит стандартный элемент chart, с помощью которого можно легко визуализировать информацию из баз данных. Chart содержит огромное количество различных видов диаграмм, от простых линейных до 3д моделей. Для подключения к графику необходимой таблицы из базы данных использовались свойства диаграммы DataSource и Series. Диаграмма носит исключительно справочный характер – показывает зависимость заработной платы сотрудника от занимаемой им должности. Цветовая схема диаграммы изменена с помощью Series -> Palette.

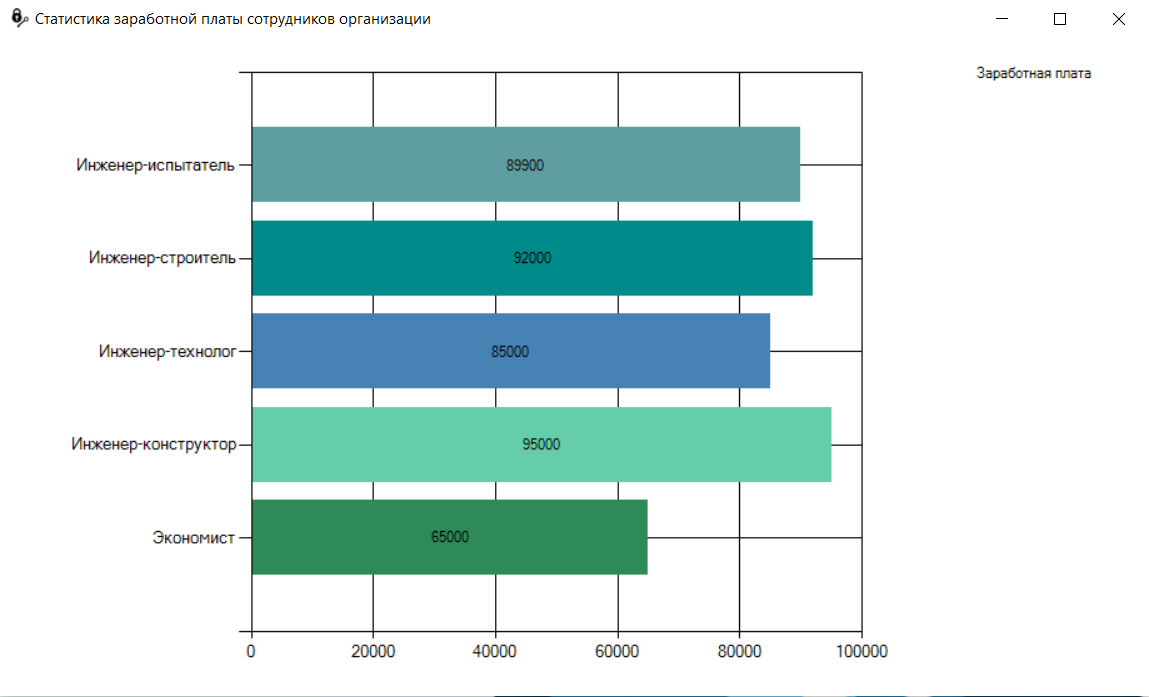


Рисунок 4.17 – Форма «Просмотр диаграммы»

Для формирования отчета было установлено расширение Microsoft RDLC Report Desinger, позволяющее формировать отчет не только в формате .rdlc, но и конвертировать полученный отчет в Word, PDF, Exel и выводить отчет на печать. Конструктор отчета расположен в среде Visual Studio (рисунок 4.18). В конструкторе можно добавить элементы отчета, такие как текстовое поле, строка, таблица, матрица и многие другие. Также конструктор отчета содержит встроенные поля, с помощью которых можно, например, задать в отчете текущую дату и время, либо изменить язык отображения отчета.

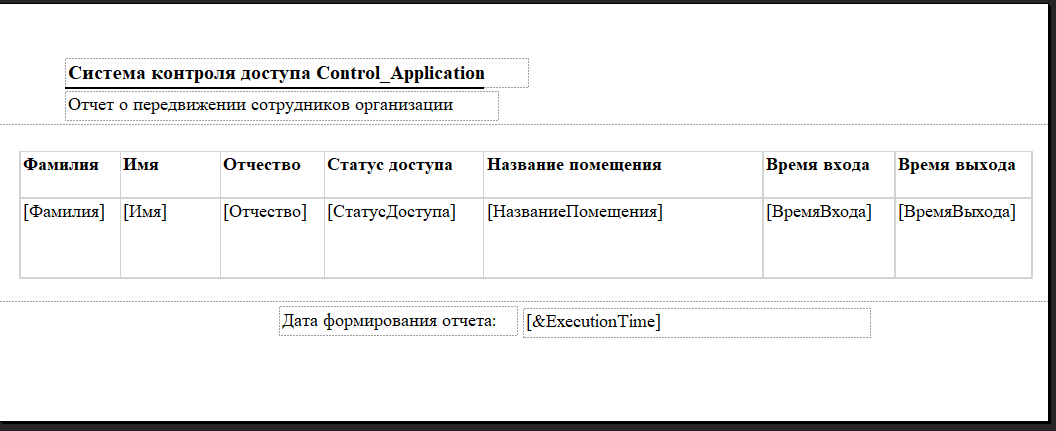


Рисунок 4.18 – Конструктор отчетов

Просмотреть отчет в программе пользователь может с помощью формы «Просмотр отчета» (рисунок 4.19). Она содержит элемент просмотра отчета reportViewer.

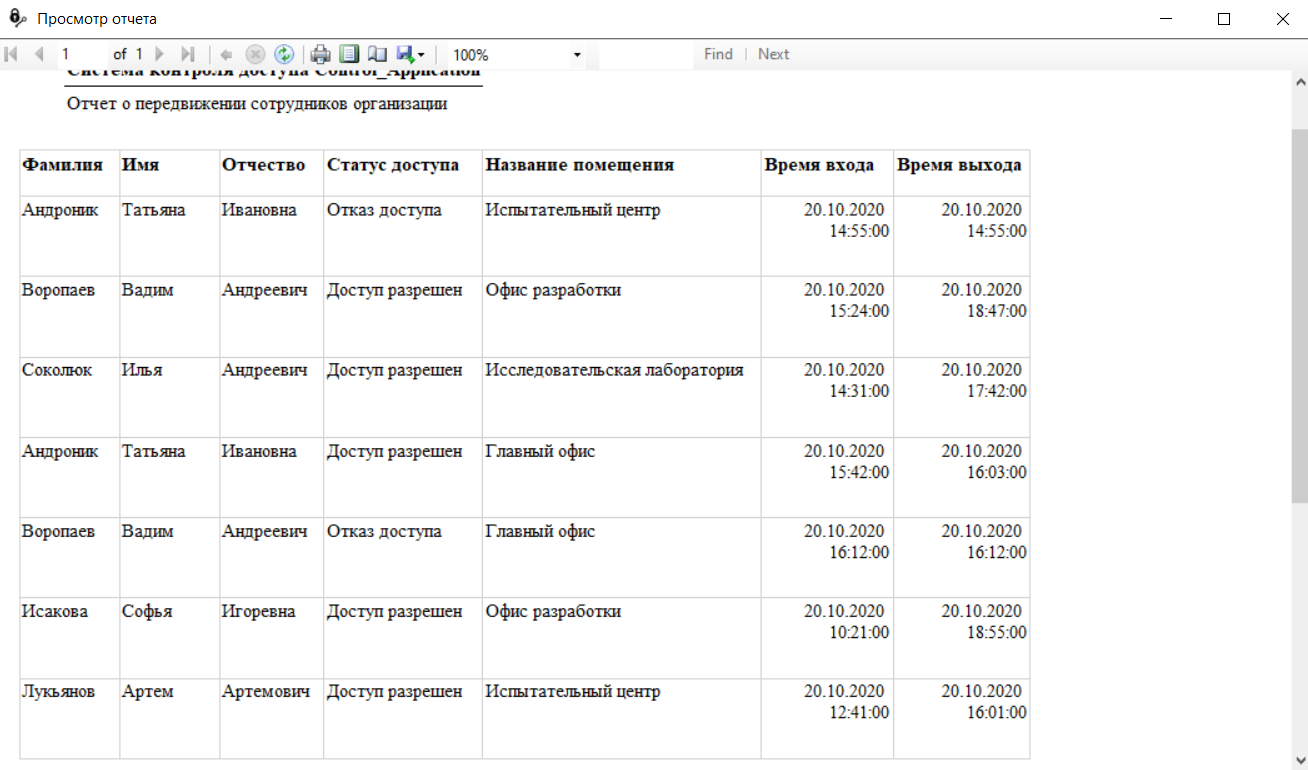


Рисунок 4.19 – Форма «Просмотр отчета»

Узнать информацию о программе и ее авторе можно с помощью формы «О программе» (рисунок 4.20). Открыть данную форму можно нажатием одноименной кнопки в menustrip в разделе «Файл».

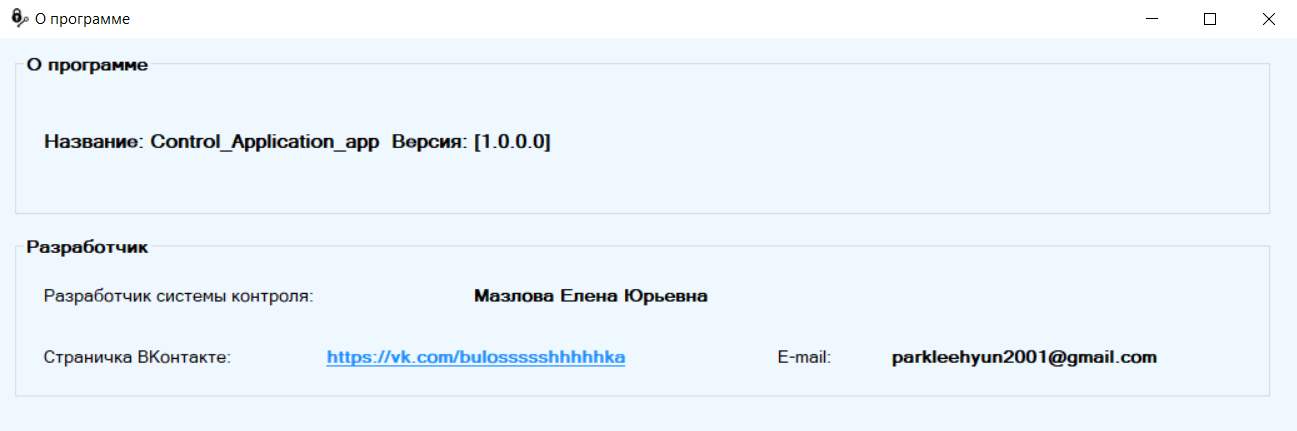


Рисунок 4.20 – Окно формы «О программе»

Форма содержит стандартные элементы label, groupbox и linklabel. Окно формы условно разделено на две части с помощью groupbox: верхняя часть формы содержит информацию о приложении, его полное название и версию, а нижняя часть содержит информацию о разработчике, его ФИО и контактную информацию. Рабочая ссылка на страничку ВКонтакте создана с помощью linklabel и пространства имен System.Diagnostics. Название приложения и его версия выводятся в lable по средствам ссылки на класс System.Reflection.Assembly. Фрагмент программного кода представлен на рисунке 4.21.

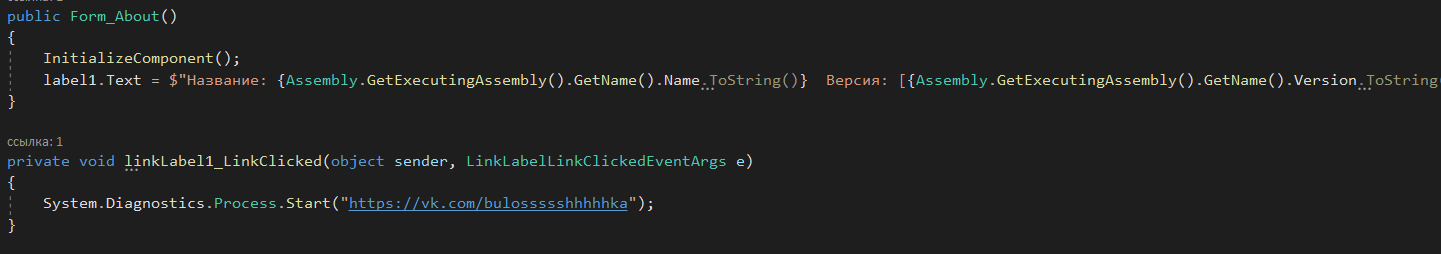


Рисунок 4.21 – Фрагмент программного кода

Приложение имеет свою иконку, которая отображается в панели задач компьютера, когда приложение запущено. Иконка приложения имеет тип .ico. Для отображения иконки в приложении она была добавлена в папку «Resouces» в среде Visual Studio, а также прикреплена к каждой форме через Свойства -> Значок.

# **Заключение**

В ходе выполнения курсовой работы были реализованы основные этапы проекта: разработка и наполнение баз данных, разработка архитектуры и проектирование пользовательского интерфейса, реализация рабочего функционала программы. Помимо этого, в процессе выполнения работы, мной были закреплены полученные по дисциплине знания.

Результат выполнения работы – удобное и приятное для работы приложение, с широким функционалом, позволяющее не только просматривать, редактировать и удалять данные в таблицах, но и представляющее данные в графическом формате и в формате отчета с возможностью последующей конвертации в удобный для хранения и печати формат.

Ожидается, что данное приложение позволит улучшить контроль за безопасностью любой организации.

# **Библиографический список**

1. Коцюба И.Ю., Чунаев А.В., Шиков А.Н. Основы проектирования информационных систем. Учебное пособие. – СПб: Университет ИТМО, 2015. – 206 с.
2. Стасышин В.М. Проектирование информационных систем и баз данных: учебное пособие. – Новосибирск: НГТУ, 2012. – 100 с.   
   <http://biblioclub.ru/index.php?page=book_red&id=228774&sr=1>
3. Горлушкина Н.Н. Системный анализ и моделирование информационных процессов и систем. Учебное пособие. – СПб: Университет ИТМО, 2016. – 120 с.
4. Грекул В.И., Денищенко Г.Н., Коровкина Н.Л. – Проектирование информационных систем: учебное пособие / 2-е изд., испр. – М.: Интернет-Университет информационных технологий (ИНТУИТ.РУ): БИНОМ. Лаборатория знаний, 2010. – 299 c.
5. Дейт К.Дж. Введение в системы баз данных. Пер. с англ. 8-е изд. / К.Дж. Дейт. – М.: Издательский дом «Вильямс», 2006. – 1328 с.
6. Кузнецов С. Введение в реляционные базы данных. – М.: Национальный Открытый Университет «ИНТУИТ», 2016. – 248 с.   
   <http://biblioclub.ru/index.php?page=book_red&id=429088&sr=1>
7. Кириллов В.В. Введение в реляционные базы данных / В.В. Кириллов, Г.Ю. Громов. – СПб.: БХВ-Петербург, 2009. – 464 с.
8. Гущин А. Н. Базы данных: учебник. – М.: Директ-Медиа, 2014. – 266 с.   
   <http://biblioclub.ru/index.php?page=book_red&id=222149&sr=1>
9. Гущин А.Н. Базы данных: учебно-методическое пособие. – М., Берлин: Директ-Медиа, 2015. – 311 с.   
   <http://biblioclub.ru/index.php?page=book_red&id=278093&sr=1>
10. Щелоков С.А. Базы данных: учебное пособие. – Оренбург: Оренбургский государственный университет, 2014. – 298 с.   
    <http://biblioclub.ru/index.php?page=book_red&id=260752&sr=1>
11. Щелоков С.А. Разработка и создание баз данных средствами СУБД Access и SQL Server. – Оренбург: Оренбургский государственный университет, 2014. – 109 c. <http://biblioclub.ru/index.php?page=book_red&id=260754&sr=1>
12. Баженова И.Ю. Основы проектирования приложений баз данных. – М.: Национальный Открытый Университет «ИНТУИТ», 2016. – 238 с.  
    <http://biblioclub.ru/index.php?page=book_red&id=428933&sr=1>
13. Грабер М. SQL. – М.: Лори, 2007. – 643 с.
14. Троелсен Э. Язык программирования C# 5.0 и платформа .NET 4.5. 6-е изд. – М.: ООО «И.Д. Вильямс», 2015. – 1312 с.
15. Флёнов М.Е. Библия С#. 2-е изд. – СПб.: БХВ-Петербург, 2011. – 560 с.
16. Лавров В.В. Методические указания к выполнению курсовой работы для студентов бакалавриата, обучающихся по направлению «Информационные системы и технологии» и студентов магистратуры, обучающихся по программе «Информационные системы в металлургии» / В.В. Лавров, К.А. Щипанов, А.А. Бурыкин. – Екатеринбург: УрФУ, 2014. – 49 с.
17. Указания к оформлению дипломных и курсовых проектов и работ. Методические указания / Н.Б. Лошкарев, А.Н. Лошкарев, Л.А. Зайнуллин. – Екатеринбург: УГТУ–УПИ, 2007. – 49 с.

# **Приложение. фрагмент листинга программного кода**

Форма «Title»

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.OleDb;

using System.Drawing;

using System.Linq;

using System.Reflection;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Control\_Application\_app

{

public partial class Form\_Title : Form

{

public static string connectString = "Provider=Microsoft.JET.OleDB.4.0;Data Source=kursovaya.mdb";

private OleDbConnection myconn;

public Form\_Title()

{

InitializeComponent();

myconn = new OleDbConnection(connectString);

myconn.Open();

}

private void оПрограммеToolStripMenuItem\_Click(object sender, EventArgs e)

{

Form\_About form\_About = new Form\_About();

form\_About.Show();

}

private void просмотрДиаграмыToolStripMenuItem\_Click(object sender, EventArgs e)

{

Form\_Chart form\_Chart = new Form\_Chart();

form\_Chart.Show();

}

private void выходToolStripMenuItem\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void Form\_Title\_FormClosing(object sender, FormClosingEventArgs e)

{

Application.Exit();

}

private void Form\_Title\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "kursovayaDataSet.Уровни\_доступа\_помещ". При необходимости она может быть перемещена или удалена.

this.уровни\_доступа\_помещTableAdapter.Fill(this.kursovayaDataSet.Уровни\_доступа\_помещ);

// TODO: данная строка кода позволяет загрузить данные в таблицу "kursovayaDataSet.Уровень\_доступа\_помещения". При необходимости она может быть перемещена или удалена.

this.уровень\_доступа\_помещенияTableAdapter.Fill(this.kursovayaDataSet.Уровень\_доступа\_помещения);

// TODO: данная строка кода позволяет загрузить данные в таблицу "kursovayaDataSet.Уровни\_доступа\_сотр". При необходимости она может быть перемещена или удалена.

this.уровни\_доступа\_сотрTableAdapter.Fill(this.kursovayaDataSet.Уровни\_доступа\_сотр);

// TODO: данная строка кода позволяет загрузить данные в таблицу "kursovayaDataSet.Уровень\_доступа\_сотрудники". При необходимости она может быть перемещена или удалена.

this.уровень\_доступа\_сотрудникиTableAdapter.Fill(this.kursovayaDataSet.Уровень\_доступа\_сотрудники);

// TODO: данная строка кода позволяет загрузить данные в таблицу "kursovayaDataSet.Помещения". При необходимости она может быть перемещена или удалена.

this.помещенияTableAdapter.Fill(this.kursovayaDataSet.Помещения);

// TODO: данная строка кода позволяет загрузить данные в таблицу "kursovayaDataSet.Должности". При необходимости она может быть перемещена или удалена.

this.должностиTableAdapter.Fill(this.kursovayaDataSet.Должности);

// TODO: данная строка кода позволяет загрузить данные в таблицу "kursovayaDataSet.Сотрудники". При необходимости она может быть перемещена или удалена.

this.сотрудникиTableAdapter.Fill(this.kursovayaDataSet.Сотрудники);

// TODO: данная строка кода позволяет загрузить данные в таблицу "kursovayaDataSet.Доступ". При необходимости она может быть перемещена или удалена.

this.доступTableAdapter.Fill(this.kursovayaDataSet.Доступ);

this.Text = this.Text + " [версия " + Assembly.GetExecutingAssembly().GetName().Version.ToString() + "]";

}

private void TB\_e\_Fname\_TextChanged(object sender, EventArgs e)

{

доступBindingSource.Filter = "[Фамилия] LIKE'" + TB\_e\_Fname.Text + "%'";

}

private void TB\_e\_name\_TextChanged(object sender, EventArgs e)

{

доступBindingSource.Filter = "[Имя] LIKE'" + TB\_e\_name.Text + "%'";

}

private void TB\_e\_Lname\_TextChanged(object sender, EventArgs e)

{

доступBindingSource.Filter = "[Отчество] LIKE'" + TB\_e\_Lname.Text + "%'";

}

private void TB\_e\_Room\_TextChanged(object sender, EventArgs e)

{

доступBindingSource.Filter = "[НазваниеПомещения] LIKE'" + TB\_e\_Room.Text + "%'";

}

private void TB\_s\_Fname\_TextChanged(object sender, EventArgs e)

{

сотрудникиBindingSource.Filter = "[Фамилия] LIKE'" + TB\_s\_Fname.Text + "%'";

}

private void TB\_s\_name\_TextChanged(object sender, EventArgs e)

{

сотрудникиBindingSource.Filter = "[Имя] LIKE'" + TB\_s\_name.Text + "%'";

}

private void TB\_s\_Lname\_TextChanged(object sender, EventArgs e)

{

сотрудникиBindingSource.Filter = "[Отчество] LIKE'" + TB\_s\_Lname.Text + "%'";

}

private void checkBox\_enterfalse\_CheckedChanged(object sender, EventArgs e)

{

if (checkBox\_enterfalse.Checked == true)

{

DataGridViewCellStyle RedcellStyle = null;

RedcellStyle = new DataGridViewCellStyle();

RedcellStyle.BackColor = Color.Red;

int index = dataGridView\_enter.Columns["статусДоступаDataGridViewTextBoxColumn"].Index;

foreach (DataGridViewRow viewRow in dataGridView\_enter.Rows)

{

if (viewRow.IsNewRow)

break;

foreach (DataGridViewCell viewCell in viewRow.Cells)

if (viewRow.Cells[index].Value.ToString().Contains("Отказ доступа"))

{

viewRow.DefaultCellStyle = RedcellStyle;

}

}

}

else if (checkBox\_enterfalse.Checked == false)

{

DataGridViewCellStyle AzurecellStyle = null;

AzurecellStyle = new DataGridViewCellStyle();

AzurecellStyle.BackColor = Color.Azure;

int index = dataGridView\_enter.Columns["статусДоступаDataGridViewTextBoxColumn"].Index;

foreach (DataGridViewRow Viewrow in dataGridView\_enter.Rows)

{

if (Viewrow.IsNewRow)

break;

foreach (DataGridViewCell viewCell in Viewrow.Cells)

if (Viewrow.Cells[index].Value.ToString().Contains("Отказ доступа"))

{

Viewrow.DefaultCellStyle = AzurecellStyle;

}

}

}

}

private void checkBox\_entertrue\_CheckedChanged(object sender, EventArgs e)

{

if (checkBox\_entertrue.Checked == true)

{

DataGridViewCellStyle GreencellStyle = null;

GreencellStyle = new DataGridViewCellStyle();

GreencellStyle.BackColor = Color.LimeGreen;

int index = dataGridView\_enter.Columns["статусДоступаDataGridViewTextBoxColumn"].Index;

foreach (DataGridViewRow Viewrow in dataGridView\_enter.Rows)

{

if (Viewrow.IsNewRow)

break;

foreach (DataGridViewCell viewCell in Viewrow.Cells)

if (Viewrow.Cells[index].Value.ToString().Contains("Доступ разрешен"))

{

Viewrow.DefaultCellStyle = GreencellStyle;

}

}

}

else if (checkBox\_entertrue.Checked == false)

{

DataGridViewCellStyle AzurecellStyle = null;

AzurecellStyle = new DataGridViewCellStyle();

AzurecellStyle.BackColor = Color.Azure;

int index = dataGridView\_enter.Columns["статусДоступаDataGridViewTextBoxColumn"].Index;

foreach (DataGridViewRow Viewrow in dataGridView\_enter.Rows)

{

if (Viewrow.IsNewRow)

break;

foreach (DataGridViewCell viewCell in Viewrow.Cells)

if (Viewrow.Cells[index].Value.ToString().Contains("Доступ разрешен"))

{

Viewrow.DefaultCellStyle = AzurecellStyle;

}

}

}

}

private void button\_save\_Click(object sender, EventArgs e)

{

try

{

if(tabControl1.SelectedTab == tabPage\_enter)

{

this.доступTableAdapter.Update(this.kursovayaDataSet.Доступ);

this.доступTableAdapter.Fill(this.kursovayaDataSet.Доступ);

checkBox\_enterfalse.Enabled = true;

checkBox\_entertrue.Enabled = true;

доступBindingSource.Filter = null;

}

if(tabControl1.SelectedTab == tabPage\_staff)

{

this.сотрудникиTableAdapter.Update(this.kursovayaDataSet.Сотрудники);

this.сотрудникиTableAdapter.Fill(this.kursovayaDataSet.Сотрудники);

сотрудникиBindingSource.Filter = null;

}

if(tabControl1.SelectedTab == tabPage\_post)

{

this.должностиTableAdapter.Update(this.kursovayaDataSet.Должности);

this.должностиTableAdapter.Fill(this.kursovayaDataSet.Должности);

this.уровень\_доступа\_сотрудникиTableAdapter.Update(this.kursovayaDataSet.Уровень\_доступа\_сотрудники);

this.уровень\_доступа\_сотрудникиTableAdapter.Fill(this.kursovayaDataSet.Уровень\_доступа\_сотрудники);

this.уровни\_доступа\_сотрTableAdapter.Update(this.kursovayaDataSet.Уровни\_доступа\_сотр);

this.уровни\_доступа\_сотрTableAdapter.Fill(this.kursovayaDataSet.Уровни\_доступа\_сотр);

}

if(tabControl1.SelectedTab == tabPage\_room)

{

this.помещенияTableAdapter.Update(this.kursovayaDataSet.Помещения);

this.помещенияTableAdapter.Fill(this.kursovayaDataSet.Помещения);

this.уровень\_доступа\_помещенияTableAdapter.Update(this.kursovayaDataSet.Уровень\_доступа\_помещения);

this.уровень\_доступа\_помещенияTableAdapter.Fill(this.kursovayaDataSet.Уровень\_доступа\_помещения);

this.уровни\_доступа\_помещTableAdapter.Update(this.kursovayaDataSet.Уровни\_доступа\_помещ);

this.уровни\_доступа\_помещTableAdapter.Fill(this.kursovayaDataSet.Уровни\_доступа\_помещ);

}

}

catch

{

MessageBox.Show("ОШИБКА СОХРАНЕНИЯ! Необходимо заполнить все ключевые поля!");

}

myconn.Close();

}

private void dataGridView\_enter\_DataError(object sender, DataGridViewDataErrorEventArgs anError)

{

MessageBox.Show("Error happened" + anError.Context.ToString());

if (anError.Context == DataGridViewDataErrorContexts.Commit)

{

MessageBox.Show("Commit error");

}

if (anError.Context == DataGridViewDataErrorContexts.CurrentCellChange)

{

MessageBox.Show("Cell change");

}

if (anError.Context == DataGridViewDataErrorContexts.Parsing)

{

MessageBox.Show("parsing error");

}

if (anError.Context == DataGridViewDataErrorContexts.LeaveControl)

{

MessageBox.Show("leave control error");

}

if ((anError.Exception) is ConstraintException)

{

DataGridView view = (DataGridView)sender;

view.Rows[anError.RowIndex].ErrorText = "an error";

view.Rows[anError.RowIndex].Cells[anError.ColumnIndex].ErrorText = "an error";

anError.ThrowException = false;

}

}

private void сформироватьОтчетToolStripMenuItem\_Click(object sender, EventArgs e)

{

Form\_Report form\_Report = new Form\_Report();

form\_Report.Show();

}

}

}

Форма «Auth»

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.OleDb;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Control\_Application\_app

{

public partial class Form\_Auth : Form

{

public static string connectString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=Auth.mdb";

private OleDbConnection myConnection;

public Form\_Auth()

{

InitializeComponent();

myConnection = new OleDbConnection(connectString);

myConnection.Open();

}

private void Form\_Auth\_Load(object sender, EventArgs e)

{

label\_error.Visible = false;

}

private void but\_auth\_Click(object sender, EventArgs e)

{

Form\_Title form\_Title = new Form\_Title();

string query = "SELECT Login, Password FROM Auth ORDER BY ID";

OleDbCommand command\_1 = new OleDbCommand(query, myConnection);

OleDbDataReader reader = command\_1.ExecuteReader();

string auth\_login = Convert.ToString(TB\_login.Text);

string auth\_password = Convert.ToString(TB\_password.Text);

while(reader.Read())

{

if ((string)reader["Login"] != auth\_login)

{

label\_error.Visible = true;

}

if(auth\_password != (string)reader["Password"])

{

label\_error.Visible = true;

}

else if (auth\_login == (string)reader["Login"])

{

if (auth\_password ==(string)reader["Password"])

{

label\_error.Visible = false;

form\_Title.Show();

this.Hide();

break;

}

}

}

reader.Close();

}

}

}

Форма «About»

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Reflection;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Control\_Application\_app

{

public partial class Form\_About : Form

{

public Form\_About()

{

InitializeComponent();

label1.Text = $"Название: {Assembly.GetExecutingAssembly().GetName().Name.ToString()} Версия: [{Assembly.GetExecutingAssembly().GetName().Version.ToString()}]";

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

System.Diagnostics.Process.Start("https://vk.com/bulossssshhhhhka");

}

private void Form\_About\_Load(object sender, EventArgs e)

{

}

}

}