Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Пермский национальный** **исследовательский политехнический университет»**

Факультет: Прикладной математики и механики

Кафедра: Вычислительной математики, механики и биомеханики

Направление: 09.03.02 «Информационные системы и технологии»

Профиль бакалавриата: «Информационные системы и технологии»

У Т В Е Р Ж Д А Ю

**Зав. кафедрой ВММБ**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В.Ю. Столбов

“\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_ г.

***ЗАДАНИЕ***

***НА КУРСОВУЮ РАБОТУ***

***по дисциплине***

**«КОРПОРАТИВНЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ»**

Рачев Руслан Игоревич; ИСТ-19-2б

(фамилия, имя, отчество студента; группа)

1. **Тема курсовой работы**

Корпоративная информационная система PC-Master

2. **Срок сдачи студентом отчета:** « » \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_г.

3. **Содержание отчета:**

Введение, средства реализации, структура базы данных,

реализация программы, заключение, список литературы

Руководитель курсовой работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата) (расшифровка)

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата) (расшифровка)

Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Пермский национальный** **исследовательский политехнический университет»**

Факультет: Прикладной математики и механики

Кафедра: Вычислительной математики, механики и биомеханики

Направление: 09.03.02 «Информационные системы и технологии»

Профиль бакалавриата: «Информационные системы и технологии»

**КУРСОВАЯ РАБОТА**

по дисциплине

**«КОРПОРАТИВНЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ»**

Тема: **«Корпоративная информационная система PC-Master»**

Выполнил:

студент гр. ИСТ-19-2б

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Рачев Руслан Игоревич

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

(Ф.И.О.)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

*(подпись)*

Принял:

ст. преподаватель, Банников Р.Ю.

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(должность, ФИО руководителя)*

*\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(оценка) (подпись)*

*\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(дата)*

**Пермь 2023**

**Оглавление**

[ВВЕДЕНИЕ 1](#_Toc132799924)

[СРЕДСТВА РЕАЛИЗАЦИИ 2](#_Toc132799925)

[Система управления базами данных 2](#_Toc132799926)

[Технологии разработки 2](#_Toc132799927)

[СТРУКТУРА БАЗЫ ДАННЫХ 4](#_Toc132799928)

[Логическая модель базы данных 4](#_Toc132799929)

[Физическая модель базы данных 5](#_Toc132799930)

[Описание основных сущностей и типы полей 5](#_Toc132799931)

[РЕАЛИЗАЦИЯ ПРОГРАММЫ 7](#_Toc132799932)

[ЗАКЛЮЧЕНИЕ 11](#_Toc132799933)

[СПИСОК ЛИТЕРАТУРЫ 12](#_Toc132799934)

ВВЕДЕНИЕ

Корпоративные информационные системы (КИС) являются центральным элементом современного бизнеса. Они обеспечивают автоматизацию и интеграцию различных бизнес-процессов, которые позволяют компаниям эффективно управлять своими ресурсами и достигать своих бизнес-целей.

КИС включают в себя различные компоненты, такие как системы управления складом, системы управления производством, системы управления клиентскими отношениями и другие. Они работают совместно, обмениваясь информацией и обеспечивая ее централизованное хранение.

Корпоративная информационная система – важный инструмент для магазина, который может обеспечить его эффективную работу.

Причины, почему КИС важна для магазина:

1. Управление складом и запасами. КИС позволяет магазину отслеживать свои запасы, контролировать приход и расход товаров, а также управлять складом. Это позволяет избежать избыточного запаса товаров или, наоборот, дефицита, что в свою очередь может привести к потере клиентов.
2. Управление продажами. КИС может помочь магазину отслеживать продажи, управлять заказами и возвращаемыми товарами. Это позволяет управлять инвентаризацией, а также анализировать популярность определенных товаров.
3. Управление персоналом. КИС может быть использована для управления работой персонала, включая учет рабочего времени, расписание смен и оплату труда. Это помогает магазину управлять своими ресурсами более эффективно и повышает производительность.
4. Управление финансами. КИС может помочь магазину управлять своими финансами, включая учет доходов и расходов, налоговую отчетность и управление бюджетом. Это может помочь магазину повысить свою прибыльность и уменьшить затраты.
5. Улучшение взаимодействия с клиентами. КИС может помочь магазину лучше понимать своих клиентов и улучшить их опыт покупок. Например, магазин может использовать данные из КИС для персонализации предложений и улучшения обслуживания.

Корпоративная информационная система является важным инструментом для магазина, который позволяет управлять бизнес-процессами более эффективно, повышать производительность и улучшать опыт покупателей.

**Назначение системы**

Система предназначена для повышения эффективности и качества деятельности магазина «PC-Master». Основным назначением системы является автоматизация рабочего места пользователей.

**Цели создания системы**

Информационно-справочная система PC-Master создается с целью:

1. Уменьшение возможности ошибок при редактировании и добавлении информации о товарах и информации о поставщиках;
2. Демонстрации данных, обеспечивающие практичность в работе с информацией;
3. Автоматизированная отчетность сведений о сотрудниках, товарах и поставщиках.

СРЕДСТВА РЕАЛИЗАЦИИ

Система управления базами данных

Для разработки базы данных программного продукта была выбрана СУБД MySQL и среда разработки MySQL Workbench.

MySQL – это свободная реляционная система управления базами данных (СУБД), которая является одной из самых популярных в мире. MySQL разработана и поддерживается компанией Oracle Corporation.

Одним из главных преимуществ MySQL является его открытый исходный код. Благодаря этому, он доступен для бесплатного использования и модификации, что делает его привлекательным выбором для многих компаний и разработчиков. Кроме того, MySQL обладает высокой производительностью, надежностью и безопасностью, что делает его идеальным выбором для многих веб-приложений и систем.

Еще одним преимуществом MySQL является его гибкость и масштабируемость. Он может быть использован для различных типов приложений, от малых веб-сайтов до крупных корпоративных систем. MySQL поддерживает многие операционные системы, включая Windows, Linux и macOS, что делает его универсальным и доступным для широкой аудитории.

Кроме того, MySQL обладает широкой поддержкой сторонних инструментов и технологий, таких как PHP, Perl, Java, Python, Ruby и многие другие. Это делает его более гибким и удобным для разработчиков и интеграторов.

Наконец, MySQL имеет большое сообщество пользователей и разработчиков, что обеспечивает высокий уровень поддержки и развития системы. Это позволяет компаниям и организациям использовать MySQL в своих проектах с уверенностью в его качестве и долгосрочной поддержке.

MySQL Workbench - это интегрированная среда разработки баз данных, предназначенная для работы с базами данных MySQL. Эта программа обеспечивает разработчикам и администраторам баз данных инструменты для создания, проектирования, моделирования, визуализации и управления базами данных MySQL.

Функциональные возможности MySQL Workbench определены следующим образом:

1. Интерпретация баз данных в виртуальной среде
2. Изменение таблиц в графическом редакторе
3. Создание связей между таблицами
4. Реализация построения ER-диаграмм из скриптов SQL, которые создавались ранее

MySQL Workbench позволяет выполнять простейшие операции с данными:

1. Добавить в таблицу одну или несколько записей;
2. Удалить из таблицы одну или несколько записей;
3. Обновить значения некоторых полей в одной или нескольких записях;
4. Найти одну или несколько записей, удовлетворяющих заданному условию.

Технологии разработки

Для разработки базы данных программного продукта был выбран язык программирования C# и среда разработки Visual Studio.

C# – это объектно-ориентированный язык программирования, разработанный компанией Microsoft. Он был создан для разработки приложений для платформы .NET Framework и быстро стал одним из наиболее популярных языков программирования в мире.

Преимущества C# перед другими языками программирования:

* Разработан компанией Microsoft, что обеспечивает его высокую степень совместимости с операционными системами Windows и другими продуктами Microsoft. Это делает C# идеальным выбором для разработки приложений для Windows.
* Имеет сильную типизацию, что означает, что компилятор проверяет типы данных во время компиляции, что позволяет обнаруживать ошибки в коде на ранних стадиях. Это уменьшает количество ошибок в коде и улучшает качество приложений.
* Имеет обширную библиотеку классов .NET Framework, которая предоставляет разработчикам широкий выбор функциональности для разработки приложений. Это ускоряет разработку приложений и уменьшает объем необходимого кода.
* Поддерживает многопоточность, что означает, что он может эффективно использовать несколько процессоров и ядер для обработки задач. Это улучшает производительность приложений и обеспечивает более быстрый отклик.
* Обладает хорошей поддержкой различных парадигм программирования, включая процедурное, объектно-ориентированное и функциональное программирование. Это позволяет разработчикам использовать подход, который лучше всего подходит для решения конкретной задачи.
* Имеет высокую степень безопасности благодаря встроенной системе управления памятью и проверки типов. Это уменьшает возможность появления ошибок в коде, которые могут привести к небезопасным ситуациям.

С учетом всех этих преимуществ, C# является одним из наиболее популярных и востребованных языков программирования в мире. Он может быть использован для разработки многих видов программного обеспечения.

Visual Studio – это интегрированная среда разработки (IDE) от компании Microsoft, которая используется для разработки различных типов приложений.

Данная среда разработки является лучшим выбором для разработки приложений для Windows на языке программирования C# по многим причинам:

1. Интеграция с платформой Windows: Visual Studio интегрирован с платформой Windows и позволяет создавать приложения, которые максимально используют возможности операционной системы. Благодаря этому, приложения, созданные в Visual Studio, могут работать более эффективно и иметь лучшую производительность.
2. Отличная поддержка для языка C#: Visual Studio предоставляет отличную поддержку для языка C#, включая подсветку синтаксиса, автодополнение, отладчик и другие инструменты, что делает процесс разработки более эффективным.
3. Множество инструментов разработки: Visual Studio предоставляет широкий выбор инструментов разработки, таких как отладчик, анализатор кода, инструменты для тестирования и другие. Это позволяет разработчикам создавать приложения более эффективно и с меньшим количеством ошибок.
4. Большое сообщество разработчиков: Visual Studio имеет большое сообщество разработчиков, которые создают и поддерживают различные расширения и плагины для IDE. Это обеспечивает большой выбор возможностей и инструментов для разработчиков.

СТРУКТУРА БАЗЫ ДАННЫХ

Для повышения эффективности деятельности любого магазина, необходим строгий учет и контроль всех процессов. Магазину PC-Master необходима информационно-справочная система, содержащая полную информацию о сотрудниках, товарах, а также его поставках и продажах.

Помимо хранения, нужна возможность для ввода, редактирования и удаления информации из базы данных. Также для администраторов системы необходима функция формирования следующих итоговых отчетов:

1. Отчет по продажам;
2. Отчет по закупкам;
3. Отчет по остаткам товаров;
4. Отчет по продажам сотрудников;
5. Отчет по прибыли.

Без возможности хранения и обработки всей информации по функционированию магазина PC-Master есть малая вероятность успешности данного предприятия. Автоматизация процесса учета в магазине PC-Master позволит уменьшить ручной труд, погасить процент ошибок, снизить время на обработку информации и работу с документацией.

Логическая модель базы данных

На основе данной предметной области было составлена логическая модель со связями (рис.1).

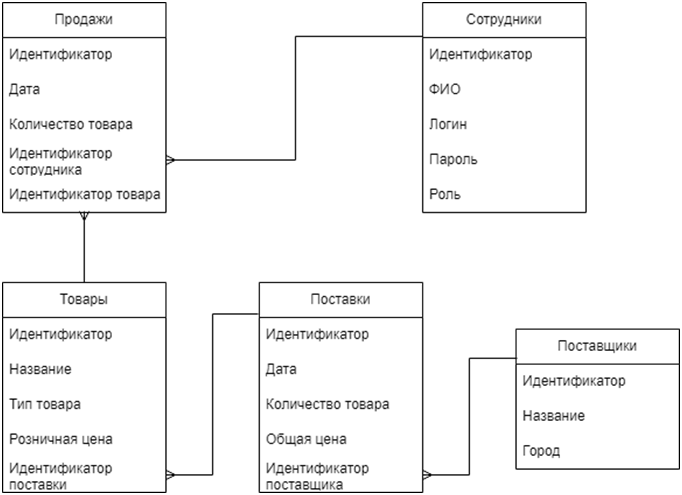


Рисунок 1 – Логическая модель базы данных

Физическая модель базы данных

На основе предметной области и логической модели базы данных была составлена физическая модель базы данных. Физическая модель диаграммы базы данных (рис.2).

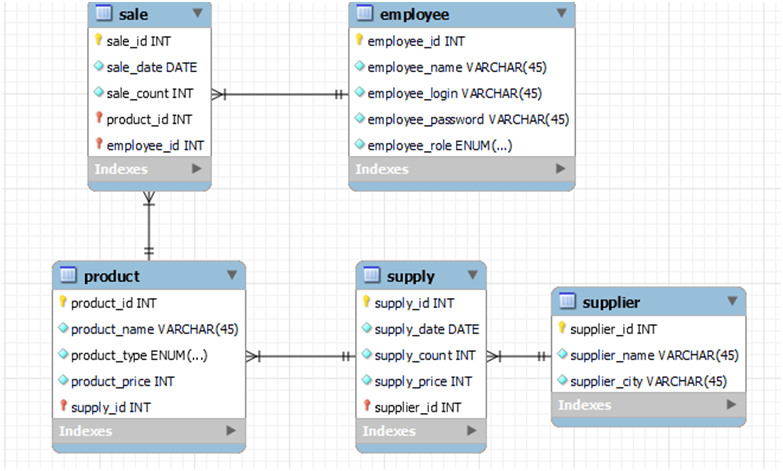


Рисунок 2 – Физическая модель базы данных

Описание основных сущностей и типы полей

1. Таблица «Поставщики» – supplier. Таблица включает в себя такие поля как идентификатор, название, город поставщика.

|  |  |
| --- | --- |
| **Столбец** | **Тип данных** |
| supplier\_id | PK, INT |
| supplier\_name | VARCHAR (45) |
| supplier\_city | VARCHAR (45) |

1. Таблица «Поставки» – supply. Таблица включает в себя такие поля как идентификатор, дата поставки, количество товара, стоимость поставки и идентификатор поставщика.

|  |  |
| --- | --- |
| **Столбец** | **Тип данных** |
| supply\_id | PK, INT |
| supply\_date | DATE |
| supply\_count | INT |
| supply\_price | INT |
| supplier\_id | FK, INT |

1. Таблица «Товары» – product. Таблица включает в себя такие поля как идентификатор, название товара, тип товара, цена, идентификатор поставки.

|  |  |
| --- | --- |
| **Столбец** | **Тип данных** |
| product\_id | PK, INT |
| product\_name | VARCHAR (45) |
| product\_type | ENUM |
| product\_price | INT |
| supply\_id | FK, INT |

1. Таблица «Продажи» – sale. Таблица является информацией о продажах товаров, которые связаны с продавцами. Таблица включает в себя столбцы идентификатор, дата продажи, количество, идентификатор товара, идентификатор продавца.

|  |  |
| --- | --- |
| **Столбец** | **Тип данных** |
| sale\_id | PK, INT |
| sale\_date | DATE |
| sale\_count | INT |
| product\_id | FK, INT |
| employee\_id | FK, INT |

1. Таблица «Сотрудники» – employee. Таблица включает в себя такие поля как идентификатор, ФИО, логин, пароль, роль сотрудника.

|  |  |
| --- | --- |
| **Столбец** | **Тип данных** |
| employee\_id | PK, INT |
| employee\_name | VARCHAR (100) |
| employee\_login | VARCHAR (100) |
| employee\_password | VARCHAR (100) |
| employee\_role | ENUM |

Тип данных для идентификаторов (первичные и внешние ключи) выбран INT, который занимает 8 бит памяти. Выбран такой тип данных т.к. записей в таблице может быть огромное количество. Для строковых типов данных заданы размеры – 45 и 100, например, для наименования поставщика, товара, роли пользователя т.к. названия могут быть длинными. Для дат выбран тип данных DATE, содержащий день, месяц, год. Для информации «о количестве» задан целочисленный тип данных INT, т.к. количество может быть до нескольких тысяч.

Связи между сущностями в таблице базы данных:

|  |  |  |
| --- | --- | --- |
| **Название таблицы 1** | **Название таблицы 2** | **Связь между таблицами** |
| supplier | supply | Один ко многим идентифицирующая связь |
| supply | product | Один ко многим идентифицирующая связь |
| product | sale | Один ко многим идентифицирующая связь |
| employee | sale | Один ко многим идентифицирующая связь |

РЕАЛИЗАЦИЯ ПРОГРАММЫ

В системе должен быть реализован ряд функций, для корректной работы с базой данных, при этом они должны отвечать всем требованиям технического задания.

Функции, которые должны быть реализованы в системе:

1. **Функция авторизации**

Внешний интерфейс для функции авторизации представлен на рисунке 1. В данном окне пользователю необходимо ввести логин и пароль для того, чтобы войти в приложение.

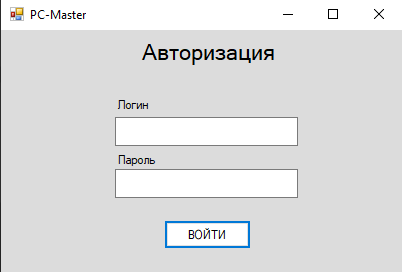


Рисунок 1 – Интерфейс функции авторизации

1. **Функция для просмотра записей в базе данных**

В системе должна быть функция для просмотра данных в базе по всем товарам, продажам, закупкам, поставщикам и сотрудникам магазина. Интерфейс просмотра справочника представлен на рисунке 2.

1. **Функция добавления, изменения и удаления записей из базы данных**

В системе должна быть функция, обеспечивающая добавление, изменение и удаление записей в каждой таблице. При закупке нового товара или при устройстве нового сотрудника, его данные должны быть добавлены в базу, также и при его увольнении и прочих событиях данные должны быть актуализированы.

Интерфейс ИС «PC-Master» для добавления, удаления и изменения записей в базе данных (рис. 2).

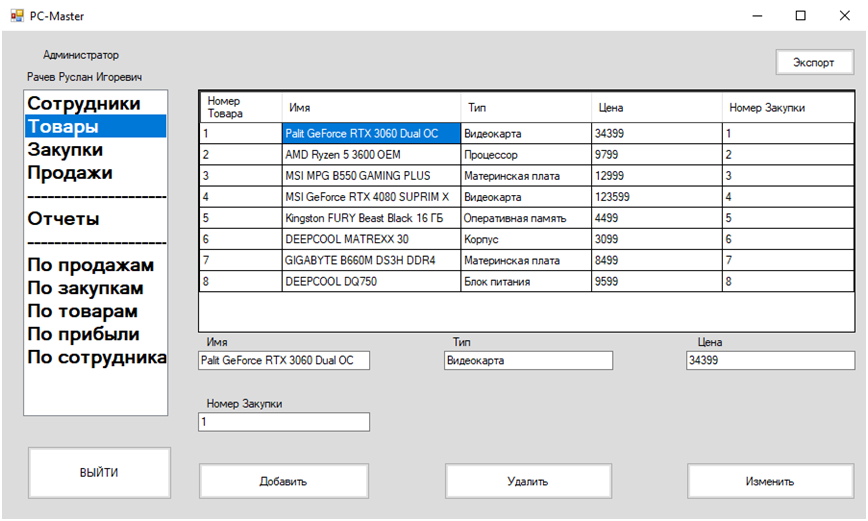


Рисунок 2 – Пример интерфейса для раздела «Товары»

Для добавления записи в базу необходимо ввести данные во все предложенные поля, после чего нажать кнопку «Добавить». После этого отображаемая таблица обновится и в ней появится новая строка.

Для удаления нужно щёлкнуть по необходимой строке и нажать на кнопку «Удалить». После этого отображаемая таблица обновится и из нее исчезнет удаленная строка.

Для изменения строки таблицы, необходимо кликнуть на необходимую строку, данные ее столбцов отобразятся в полях снизу. После этого нужно изменить данные строки на необходимые данные и нажать кнопку «Изменить».

1. **Функция генерации отчетов**

В системе присутствует 5 видов отчета (рис.3). Каждый из них предназначен для отображения своего набора данных.

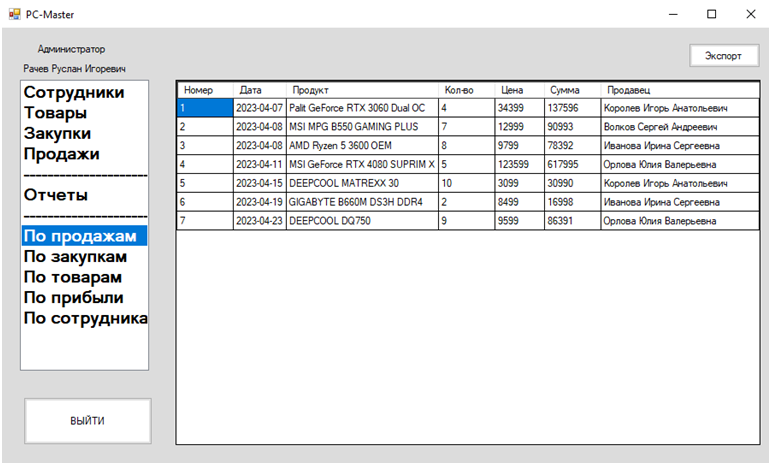


Рисунок 3 – Пример интерфейса для отображения отчета

Отображаемые отчетами данные:

1. Отчет по продажам предоставляет данные по всем совершенным продажам.
2. Отчет по закупкам предоставляет данные по всем совершенным закупкам.
3. Отчет по товарам предоставляет данные по всем товарам на складе.
4. Отчет по прибыли предоставляет данные по прибыли, полученной с продаж каждого товара.
5. Отчет по сотрудникам предоставляет данные обо всех работниках магазина.

Каждый из разработанных отчетов необходимо отображать в приложении.

1. **Функция выгрузки отчетов в формат Excel**

В программе должна быть функция выгрузки отчета в формат xlsx. Для переноса необходимого отчета необходимо нажать кнопку «Экспорт».

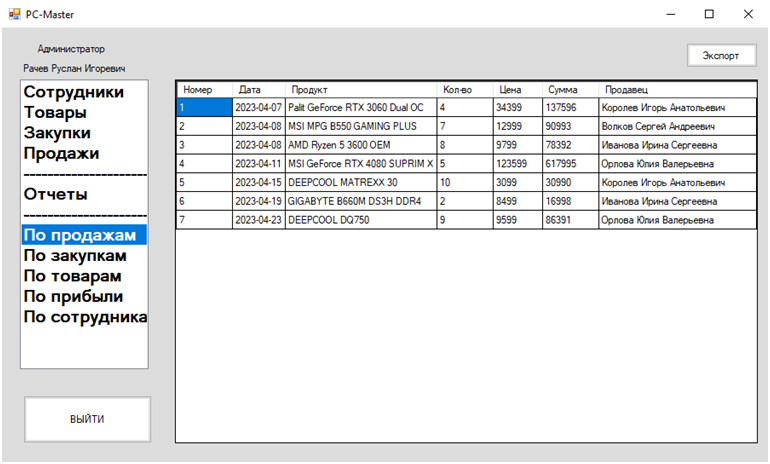


Рисунок 4 – Интерфейс для выгрузки отчета

После выгрузки открывается Excel и отчет автоматически копируется в новый документ (рис.5). После этого, сделав все необходимые изменения, пользователь может сохранить отчет в формате xlsx.

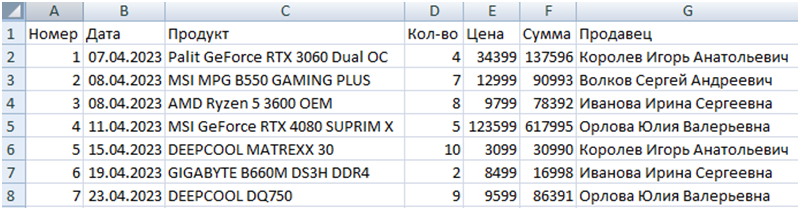


Рисунок 5 – Просмотр отчета в Excel

1. **Реакция системы на ошибочный ввод.**

При попытке входа в систему с неверными или пустыми данными авторизации на экран выводится информация об ошибке (рис.6). Данная функция важна для того, чтобы к системе не получили доступ лица, не являющиеся сотрудниками магазина.

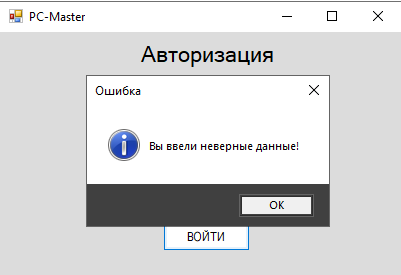


Рисунок 6 – Вывод ошибки при авторизации

При добавлении или изменении строк справочников, введенные пользователями данные также проверяются на корректность. Если сотрудник магазина вводит данные, которые не соответствуют своему типу, на экран выводится ошибка об этом (рис.7).

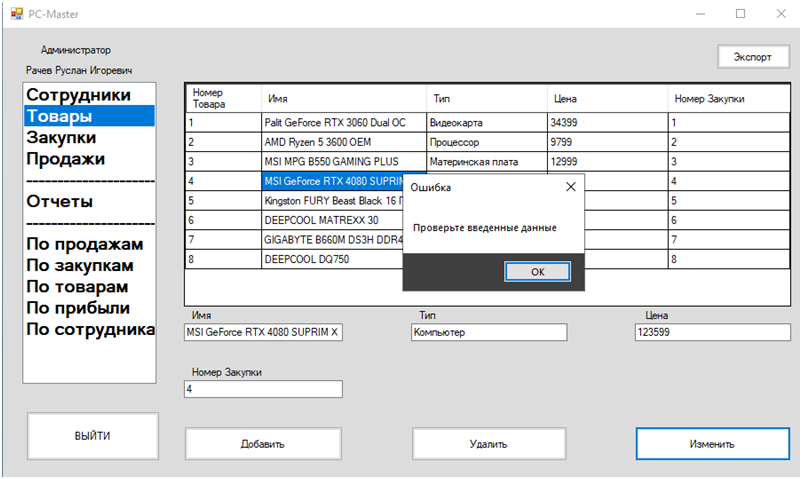


Рисунок 7 – Вывод ошибки в типе данных

ЗАКЛЮЧЕНИЕ

В данной работе была спроектирована база данных, а также реализован основной функционал системы: аутентификация, вид под разными пользователями (в зависимости от роли), редактировании базы данных, формирование отчетов, их выгрузка, а также реакция системы на ошибочный ввод.

Данная корпоративная система выполняет все поставленные задачи, благодаря чему может использоваться для автоматизации процессов магазина, повышая при этом эффективность его работы.

В настоящее время, КИС являются неотъемлемой частью бизнеса, и их использование будет продолжать расти в будущем.

СПИСОК ЛИТЕРАТУРЫ

1. Ищенко К.А. Обеспечение качества и эффективности корпоративных информационных систем.//Тамбовский государственный технический университет. –2022. –С. 266-268.
2. Кушбакова Б.К. Факторы формирования корпоративных информационных систем.//Башкирский государственный университет. –2020. –С. 195-198.
3. Мартишин С.А., Симонов В.Л., Храпченко М.В. Проектирование и реализация баз данных в СУБД MySQL с использованием MySQL Workbench.// ИД "Форум". –2012.
4. Ефремова А.Н., Полячкова М.А., Васильева Л.В., Гаранина И.А.   
   MySQL Workbench как инструментальное средство информационной системы.// ФГБОУ ВО «БрГУ», г. Братск, ФГБОУ ВО «ТГМУ», г. Владивосток. –2021. –С. 183-186.
5. Ильясова Ф.С., Валиева Н.Э. Разработка схемы данных информационной системы c использованием MySQL Workbench.//Крымский инженерно-педагогический университет. –2018. – №1(19). –С. 135-142.
6. Марченко А.Л. Основы программирования на C# 2.0.//Интернет-Университет Информационных Технологий. –2017. –522с.
7. Галлини Н.И., Филимоненкова Т.Н. Обзор средств разработки программного обеспечения с помощью Microsoft Visual Studio 2013.//Крымский гуманитарный университет. –2014. – №45-1. –С. 316-321.

Приложение 1

SQL-код заполнения базы данных:

INSERT INTO `mydb`.`employee` (`employee\_name`, `employee\_login`, `employee\_password`, `employee\_role`) VALUES ('Королев Игорь Анатольевич', 'korolevigor', '12345igorek', 'Продавец');

INSERT INTO `mydb`.`employee` (`employee\_name`, `employee\_login`, `employee\_password`, `employee\_role`) VALUES ('Рачев Руслан Игоревич', 'rouslan', '870125', 'Администратор');

INSERT INTO `mydb`.`employee` (`employee\_id`, `employee\_name`, `employee\_login`, `employee\_password`, `employee\_role`) VALUES ('3', 'Орлова Юлия Валерьевна', 'orlovaa', '412423', 'Продавец');

INSERT INTO `mydb`.`employee` (`employee\_id`, `employee\_name`, `employee\_login`, `employee\_password`, `employee\_role`) VALUES ('4', 'Волков Сергей Андреевич', 'volk473', '9993seriy', 'Продавец');

INSERT INTO `mydb`.`employee` (`employee\_id`, `employee\_name`, `employee\_login`, `employee\_password`, `employee\_role`) VALUES ('5', 'Иванова Ирина Сергеевна', 'ivanovai', 'ivan18325irina', 'Продавец');

INSERT INTO `mydb`.`supplier` (`supplier\_name`, `supplier\_city`) VALUES ('ООО Техника-Дорого', 'Москва');

INSERT INTO `mydb`.`supplier` (`supplier\_name`, `supplier\_city`) VALUES ('Лаборатория Компьютеров', 'Санкт-Петербург');

INSERT INTO `mydb`.`supplier` (`supplier\_name`, `supplier\_city`) VALUES ('Cheap computers', 'Тверь');

INSERT INTO `mydb`.`supplier` (`supplier\_name`, `supplier\_city`) VALUES ('АйТи Технологии', 'Пермь');

INSERT INTO `mydb`.`supplier` (`supplier\_name`, `supplier\_city`) VALUES ('SunDzin Tech', 'Гуанчжоу');

INSERT INTO `mydb`.`supply` (`supply\_date`, `supply\_count`, `supply\_price`, `supplier\_id`) VALUES ('2023-03-28', '25', '700000', '5');

INSERT INTO `mydb`.`supply` (`supply\_date`, `supply\_count`, `supply\_price`, `supplier\_id`) VALUES ('2023-04-05', '30', '210000', '1');

INSERT INTO `mydb`.`supply` (`supply\_date`, `supply\_count`, `supply\_price`, `supplier\_id`) VALUES ('2023-04-05', '20', '200000', '1');

INSERT INTO `mydb`.`supply` (`supply\_date`, `supply\_count`, `supply\_price`, `supplier\_id`) VALUES ('2023-04-02', '13', '1300000', '3');

INSERT INTO `mydb`.`supply` (`supply\_date`, `supply\_count`, `supply\_price`, `supplier\_id`) VALUES ('2023-04-10', '40', '108000', '2');

INSERT INTO `mydb`.`supply` (`supply\_date`, `supply\_count`, `supply\_price`, `supplier\_id`) VALUES ('2023-04-10', '20', '38000', '2');

INSERT INTO `mydb`.`supply` (`supply\_date`, `supply\_count`, `supply\_price`, `supplier\_id`) VALUES ('2023-04-10', '35', '239750', '2');

INSERT INTO `mydb`.`supply` (`supply\_date`, `supply\_count`, `supply\_price`, `supplier\_id`) VALUES ('2023-04-21', '45', '346500', '4');

INSERT INTO `mydb`.`product` (`product\_name`, `product\_type`, `product\_price`, `supply\_id`) VALUES ('Palit GeForce RTX 3060 Dual OC', 'Видеокарта', '34399', '1');

INSERT INTO `mydb`.`product` (`product\_name`, `product\_type`, `product\_price`, `supply\_id`) VALUES ('AMD Ryzen 5 3600 OEM', 'Процессор', '9799', '2');

INSERT INTO `mydb`.`product` (`product\_name`, `product\_type`, `product\_price`, `supply\_id`) VALUES ('MSI MPG B550 GAMING PLUS', 'Материнская плата', '12999', '3');

INSERT INTO `mydb`.`product` (`product\_name`, `product\_type`, `product\_price`, `supply\_id`) VALUES ('MSI GeForce RTX 4080 SUPRIM X', 'Видеокарта', '123599', '4');

INSERT INTO `mydb`.`product` (`product\_name`, `product\_type`, `product\_price`, `supply\_id`) VALUES ('Kingston FURY Beast Black 16 ГБ', 'Оперативная память', '4499', '5');

INSERT INTO `mydb`.`product` (`product\_name`, `product\_type`, `product\_price`, `supply\_id`) VALUES ('DEEPCOOL MATREXX 30', 'Корпус', '3099', '6');

INSERT INTO `mydb`.`product` (`product\_name`, `product\_type`, `product\_price`, `supply\_id`) VALUES ('GIGABYTE B660M DS3H DDR4', 'Материнская плата', '8499', '7');

INSERT INTO `mydb`.`product` (`product\_name`, `product\_type`, `product\_price`, `supply\_id`) VALUES ('DEEPCOOL DQ750', 'Блок питания', '9599', '8');

INSERT INTO `mydb`.`sale` (`sale\_date`, `sale\_count`, `product\_id`, `employee\_id`) VALUES ('2023-04-07', '4', '1', '1');

INSERT INTO `mydb`.`sale` (`sale\_date`, `sale\_count`, `product\_id`, `employee\_id`) VALUES ('2023-04-08', '7', '3', '4');

INSERT INTO `mydb`.`sale` (`sale\_date`, `sale\_count`, `product\_id`, `employee\_id`) VALUES ('2023-04-08', '8', '2', '5');

INSERT INTO `mydb`.`sale` (`sale\_date`, `sale\_count`, `product\_id`, `employee\_id`) VALUES ('2023-04-11', '5', '4', '3');

INSERT INTO `mydb`.`sale` (`sale\_date`, `sale\_count`, `product\_id`, `employee\_id`) VALUES ('2023-04-15', '10', '6', '1');

INSERT INTO `mydb`.`sale` (`sale\_date`, `sale\_count`, `product\_id`, `employee\_id`) VALUES ('2023-04-19', '2', '7', '5');

INSERT INTO `mydb`.`sale` (`sale\_date`, `sale\_count`, `product\_id`, `employee\_id`) VALUES ('2023-04-23', '9', '8', '3');

SQL-код для отчета по продажам:

CREATE VIEW sales\_report

as SELECT

sale.sale\_id as "Номер",

sale.sale\_date as "Дата",

product.product\_name as "Продукт",

sale.sale\_count as "Кол-во",

product.product\_price as "Цена",

(sale.sale\_count\*product.product\_price) as "Сумма",

employee.employee\_name as "Продавец"

FROM sale

JOIN product ON sale.product\_id = product.product\_id

JOIN employee ON sale.employee\_id = employee.employee\_id;

SQL-код для отчета по закупкам:

CREATE VIEW supplies\_report

as SELECT

supply.supply\_id as "Номер",

supply.supply\_date as "Дата",

product.product\_name as "Продукт",

supply.supply\_count as "Кол-во",

REPLACE(FORMAT((supply.supply\_price/supply.supply\_count), '######'), ",", "") as "Цена закупки",

supply.supply\_price as "Сумма",

supplier.supplier\_name as "Поставщик",

supplier.supplier\_city as "Город"

FROM supply

JOIN product ON supply.supply\_id = product.supply\_id

JOIN supplier ON supply.supplier\_id = supplier.supplier\_id

ORDER BY supply.supply\_date;

SQL-код для отчета по товарам:

CREATE VIEW products\_report

as SELECT

product.product\_id as "Номер",

product.product\_name as "Продукт",

product.product\_type as "Тип",

supplier.supplier\_name as "Поставщик",

supply.supply\_date as "Дата поставки",

supply.supply\_count as "Кол-во",

SUM(sale.sale\_count) as "Продано",

(supply.supply\_count-SUM(sale.sale\_count)) as "На складе"

FROM product

JOIN supply ON supply.supply\_id = product.supply\_id

JOIN supplier ON supply.supplier\_id = supplier.supplier\_id

JOIN sale ON sale.product\_id = product.product\_id

GROUP BY product.product\_id, supply.supply\_id, supplier.supplier\_id, sale.sale\_id;

SQL-код для отчета по прибыли:

CREATE VIEW profits\_report

as SELECT

product.product\_id as "Номер",

product.product\_name as "Продукт",

product.product\_type as "Тип",

REPLACE(FORMAT((supply.supply\_price/supply.supply\_count), '######'), ",", "") as "Закупочная цена",

product.product\_price as "Розничная цена",

SUM(sale.sale\_count) as "Продано",

(SUM(sale.sale\_count)\*product.product\_price) as "Сумма продаж",

(SUM(sale.sale\_count)\*(product.product\_price-REPLACE(FORMAT((supply.supply\_price/supply.supply\_count), '######'), ",", ""))) as "Прибыль"

FROM product

JOIN supply ON supply.supply\_id = product.supply\_id

JOIN supplier ON supply.supplier\_id = supplier.supplier\_id

JOIN sale ON sale.product\_id = product.product\_id

GROUP BY product.product\_id, supply.supply\_id, supplier.supplier\_id, sale.sale\_id;

SQL-код для отчета по сотрудникам:

CREATE VIEW employees\_report as

SELECT

abc.employee\_id as "Номер",

abc.employee\_name as "ФИО",

SUM(abc.summ\_count) as "Кол-во продаж",

SUM(abc.summ\_price) as "Сумма продаж",

SUM(abc.summ\_profit) as "Заработал"

from(

SELECT

employee.employee\_id,

employee.employee\_name,

SUM(sale.sale\_count) as summ\_count,

(SUM(sale.sale\_count)\*product.product\_price) as summ\_price,

(SUM(sale.sale\_count)\*(product.product\_price-REPLACE(FORMAT((supply.supply\_price/supply.supply\_count), '######'), ",", ""))) as summ\_profit

FROM employee

JOIN sale ON sale.employee\_id = employee.employee\_id

JOIN product ON sale.product\_id = product.product\_id

JOIN supply ON supply.supply\_id = product.supply\_id

GROUP BY product.product\_id, sale.sale\_id, supply.supply\_id, supply.supply\_price, supply.supply\_count, employee.employee\_id) as abc

GROUP BY abc.employee\_id;

Программный код для аутентификации с помощью запроса к базе данных:

private bool checkPassword()

{

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

string sql = "Select \* from employee WHERE employee\_login='"+loginInput.Text+"' AND employee\_password='"+passwordInput.Text+"'";

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

using (DbDataReader reader = cmd.ExecuteReader())

{

if (reader.HasRows)

{

while (reader.Read())

{

name = Convert.ToString(reader.GetValue(1));

role = Convert.ToString(reader.GetValue(4));

}

}

}

Console.WriteLine(name);

conn.Close();

conn.Dispose();

if (name == "") return false;

else return true;

}

Программный код для отображения базы:

private void queryTable(String table\_name)

{

itemId = 0;

dataGridView1.Columns.Clear();

dataGridView1.Rows.Clear();

columns.Clear();

clearTextBoxes();

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

table\_name = translateTable(table\_name);

string sql = "Select \* from " + table\_name;

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

columns = new List<string>();

using (DbDataReader reader = cmd.ExecuteReader())

{

if (reader.HasRows)

{

int count = 0;

DataTable schema = null;

schema = reader.GetSchemaTable();

foreach (DataRow col in schema.Rows)

{

string сname = translateColumn(col.Field<String>("ColumnName"));

dataGridView1.Columns.Add(col.Field<String>("ColumnName").ToUpper(), сname);

columns.Add(col.Field<String>("ColumnName"));

count++;

}

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.None;

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

dataGridView1.Columns[i].Width = (int)(dataGridView1.Width \* 1 / count);

dataGridView1.Columns[0].Width -= 50;

dataGridView1.Columns[1].Width += 50;

dataGridView1.Columns[count – 1].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

int row = 0;

while (reader.Read())

{

dataGridView1.Rows.Add();

for (int column = 0; column < count; column++)

{

string value = "";

if (reader.GetValue(column).GetType() == typeof(DateTime))

{

DateTime time = (DateTime) reader.GetValue(column);

value = time.ToString("yyyy-MM-dd");

}

else value = Convert.ToString(reader.GetValue(column));

dataGridView1.Rows[row].Cells[column].Value = value;

}

row++;

}

rows = row;

}

}

conn.Close();

conn.Dispose();

showControl();

}

Программный код для добавления записи:

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

{

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

String table\_name = translateTable(tableChoice.SelectedItem.ToString());

String clmns = "";

for (int i = 1; i < columns.Count; i++)

{

clmns += columns[i] + ", ";

}

clmns = clmns.Substring(0, clmns.Length – 2);

String text = "";

TextBox[] textBoxes = new TextBox[6] { textBox1, textBox2, textBox3, textBox4, textBox5, textBox6 };

for (int i = 0; i < columns.Count – 1; i++)

{

text += "'" + textBoxes[i].Text + "', ";

}

text = text.Substring(0, text.Length – 2);

string sql = "Insert into " + table\_name + " (" + clmns + ") values (" + text + ")";

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

try

{

cmd.ExecuteNonQuery();

MessageBox.Show("Строка добавлена!", "Message");

}

catch (Exception)

{

MessageBox.Show("Проверьте введенные данные", "Ошибка");

}

conn.Close();

conn.Dispose();

queryTable(table\_name);

}

Программный код для удаления записи:

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

{

if (itemId != 0)

{

String table\_name = translateTable(tableChoice.SelectedItem.ToString());

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

string sql = "DELETE FROM " + table\_name + " WHERE " + columns[0] + "=" + itemId;

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

try

{

cmd.ExecuteNonQuery();

MessageBox.Show("Строка удалена!", "Message");

}

catch (Exception)

{

MessageBox.Show("Проверьте введенные данные", "Ошибка");

}

conn.Close();

queryTable(table\_name);

} else MessageBox.Show("Ни одна строка не выбрана!", "Ошибка");

}

Программный код для изменения записи:

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

{

if (itemId != 0)

{

String table\_name = translateTable(tableChoice.SelectedItem.ToString());

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

String text = "";

TextBox[] textBoxes = new TextBox[6] { textBox1, textBox2, textBox3, textBox4, textBox5, textBox6 };

for (int i = 1; i < columns.Count; i++)

{

text += columns[i] + "='" + textBoxes[i – 1].Text + "',";

}

text = text.Substring(0, text.Length – 2);

string sql = "UPDATE " + table\_name + " SET " + text +

" WHERE " + columns[0] + "=" + itemId;

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

try

{

cmd.ExecuteNonQuery();

MessageBox.Show("Строка изменена!", "Message");

}

catch (Exception)

{

MessageBox.Show("Проверьте введенные данные", "Ошибка");

}

conn.Close();

queryTable(table\_name);

} else MessageBox.Show("Ни одна строка не выбрана!", "Ошибка");

}

Программный код для отображения отчетов:

private void queryReport(String report\_name)

{

dataGridView1.Columns.Clear();

dataGridView1.Rows.Clear();

columns.Clear();

clearTextBoxes();

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

report\_name = translateReport(report\_name);

string sql = "Select \* from " + report\_name;

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

columns = new List<string>();

using (DbDataReader reader = cmd.ExecuteReader())

{

if (reader.HasRows)

{

int count = 0;

DataTable schema = null;

schema = reader.GetSchemaTable();

foreach (DataRow col in schema.Rows)

{

string сname = col.Field<String>("ColumnName");

dataGridView1.Columns.Add(col.Field<String>("ColumnName").ToUpper(), сname);

count++;

}

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

dataGridView1.Columns[i].Width = (int)(dataGridView1.Width \* 1 / count);

if (report\_name == "sales\_report" || report\_name == "employees\_report")

dataGridView1.Columns[count – 1].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.AllCells;

int row = 0;

while (reader.Read())

{

dataGridView1.Rows.Add();

for (int column = 0; column < count; column++)

{

string value = "";

if (reader.GetValue(column).GetType() == typeof(DateTime))

{

DateTime time = (DateTime)reader.GetValue(column);

value = time.ToString("yyyy-MM-dd");

}

else value = Convert.ToString(reader.GetValue(column));

dataGridView1.Rows[row].Cells[column].Value = value;

}

row++;

}

rows = row;

}

}

conn.Close();

conn.Dispose();

}

Программный код для выгрузки отчетов:

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

{

Excel.Application ExcelApp = new Excel.Application();

Excel.Workbook ExcelWorkBook;

Excel.Worksheet ExcelWorkSheet;

ExcelWorkBook = ExcelApp.Workbooks.Add(System.Reflection.Missing.Value);

ExcelWorkSheet = (Excel.Worksheet)ExcelWorkBook.Worksheets.get\_Item(1);

for (int i = 0; i < dataGridView1.Rows.Count; i++)

{

for (int j = 0; j < dataGridView1.ColumnCount; j++)

{

ExcelApp.Cells[1, j + 1] = dataGridView1.Columns[j].HeaderText.ToString();

ExcelApp.Cells[i + 2, j + 1] = dataGridView1.Rows[i].Cells[j].Value;

}

}

ExcelApp.Visible = true;

ExcelApp.UserControl = true;

ExcelApp.Rows.AutoFit();

ExcelApp.Columns.AutoFit();

}

Программный код реакции на ошибку при авторизации:

if (loginInput.Text != "" && passwordInput.Text != "")

{

bool userExists = checkPassword();

if (userExists)

{

loginInput.Text = "";

passwordInput.Text = "";

this.Hide();

Main mainForm = new Main(name, role, this);

mainForm.Show();

}

else

{

MessageBox.Show("Вы ввели неверные данные!",

"Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Information,

MessageBoxDefaultButton.Button1,

MessageBoxOptions.DefaultDesktopOnly);

}

}

else

{

MessageBox.Show("Введите логин и пароль!",

"Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Information,

MessageBoxDefaultButton.Button1,

MessageBoxOptions.DefaultDesktopOnly);

}

Программный код реакции на ошибку в типе данных:

try

{

cmd.ExecuteNonQuery();

MessageBox.Show("Строка добавлена!", "Message");

}

catch (Exception)

{

MessageBox.Show("Проверьте введенные данные", "Ошибка");

}