

СОДЕРЖАНИЕ

Введение…………………………………………………………………….6

1 Постановка задачи……………………………...………………………...7

2 Структура программы………………………...……………………….....9

3 Схема данных…………...…………………………………………….…13

4 Оптимизация программы ………………………..……………………..17

1. Тестирование программы………………………………………………19

6 Руководство пользователя………………………….…………………..25

Заключение…………………………………………………………….…..34

Список использованных источников.…………………………………....35

Приложение А Листинг программы..……………….…………………....37

Приложение Б Результат выполнения программы...…………………....60

Приложение 1 Аттестационный лист…………………………………….67

Приложение 2 Дневник…………………………………………………...69

Приложение 3 Характеристика…………………………………………..74

ВВЕДЕНИЕ

База данных (БД) “sklad” создана в рамках прохождения производственной практики ПП.04.

Для ее реализации необходимо решить следующие задачи:

1. Изучить предметную область, т.е. изучить сферу деятельности организации.
2. Проанализировав информацию, полученную в ходе выполнения первого пункта, построить необходимые схемы для дальнейшей эффективной разработки базы данных.
3. Выбрать средство для разработки и приступить к началу работы.

После достижения результатов по вышеперечисленным пунктам заполнить созданную базу данных и создать запросы для корректного извлечения информации и удобства дальнейшей ее эксплуатации.

Для разработки были выбраны сервис MySQL и язык программирования C#.

MySQL— это система управления реляционными базами данных, разработанная корпорацией Microsoft. Эта СУБД учитывает все современные требования к работе с данными различных форматов. MySQL упрощает развертывание, передачу и интеграцию больших данных.

Для разработки приложения будет использоваться Microsoft Visual Studio.

1 Постановка задачи

Разработать приложение для работы с базой данных (БД) sklad. Приложение должно обеспечивать возможности внесения, изменения и удаления данных, разделение уровня доступа для администратора БД и сотрудника.

Для реализации нужно:

а) Создать базу данных, используя наиболее подходящую к выданному заданию платформу (MySQL / Microsoft SQL Server).

б) Создать приложение, используя наиболее приемлемую для решения задачи платформу: .NET (или Java).

в) Создать форму для авторизации зарегистрированных пользователей.

г) Реализовать разделение уровня доступа для администратора БД и сотрудников.

д) Пользователи должны входить в систему, используя форму авторизации. Форма авторизации должна соответствовать следующим требованиям:

1. Для успешной авторизации необходимо ввести свой логин и пароль.
2. При попытке входа введенные данные проверяются на совпадение в базе данных. Должно выдаваться сообщение об ошибке в случае неправильного ввода связки логин/пароль.
3. После успешного входа пользователя в зависимости от его роли происходит перенаправление на конкретные формы, предназначенные для работы этого типа пользователя.
4. При нажатии на кнопку "Выход" в любом окне/форме системы, пользователи должны быть отключены от системы и совершен переход на Форму авторизации.
5. Необходимо предусмотреть возможность регистрации новых пользователей определенной роли. Поэтому нужно добавить в систему форму регистрации. При регистрации пароль должен отвечать следующим требованиям:

– должен содержать от 6 до 16 символов;

– не должно быть символов из набора: \* & { } | +;

– должны встречаться заглавные буквы;

– должны встречаться цифры.

е) Должны быть реализованы основные формы для работы с сущностями, базы данных.

ж) Необходимо реализовать автоматизацию не менее 2 бизнес-процессов для каждого из видов пользователей и необходимых для этого форм.

з) Реализовать форму вывода отчета с возможностью печати документа.

и) Для выполнения процедуры тестирования корректного ввода пароля при регистрации заказчика вам нужно создать два модульных теста, которые будут проверять различные комбинации условий ввода пароля.

к) Необходимо, чтобы модульные тесты демонстрировали как положительный, так и отрицательный результаты при разных входных данных. Набор модульных тестов должен быть реализован в отдельном проекте. Все тесты должны выполняться последовательно.

2 Структура программы

На рисунках 2.1-2.3 представлена структура программы.



Рисунок 2.1 – Структура программы



Рисунок 2.2 – Структура программы



Рисунок 2.3 – Структура программы

Формы, используемые в основной программе, приведены в таблице 2.1.

Таблица 2.1 – Формы, используемые в основной программе

| **Форма** | **Назначение** |
| --- | --- |
| Form1 | Форма авторизации для пользователя |
| reg | Форма регистрации |
| tables | Форма выбора таблиц для пользователя |
| inventary | Пользовательская форма для работы с таблицей «инвентарь» |
| sklady | Пользовательская форма для работы с таблицей «склад» |
| otchety | Пользовательская форма для работы с таблицей «отчёты» |
| admintables | Форма авторизации для администратора |
| admin | Форма выбора таблиц для администратора |
| inventaryadmin | Форма администратора для работы с таблицей «инвентарь» |
| personaladmin | Форма администратора для работы с таблицей «персонал» |
| skladadmin | Форма администратора для работы с таблицей «склад» |
| adminotchety | Форма администратора для работы с таблицей «отчёты» |

3 Схема данных

Для создания базы данных склада нужно знать данные: о предмете, находящемся в распоряжении отдела, количество предметов, его дату поставки и цену. Конечным этапом работы программы будет таблица со всеми предметами, его расположением, в виде таблиц.

Исходя из этого, можно составить контекстную диаграмму, которая представлена на рисунке 3.1.



Рисунок 3.1 – Контекстная диаграмма

Можно выделить три основные сущности: отчёты о поставках, инвентарь, склад. Данные сущности представлены на рисунке 3.2.



Рисунок 3.2 – Основные сущности базы данных

Логическая схема (рисунок 3.3).

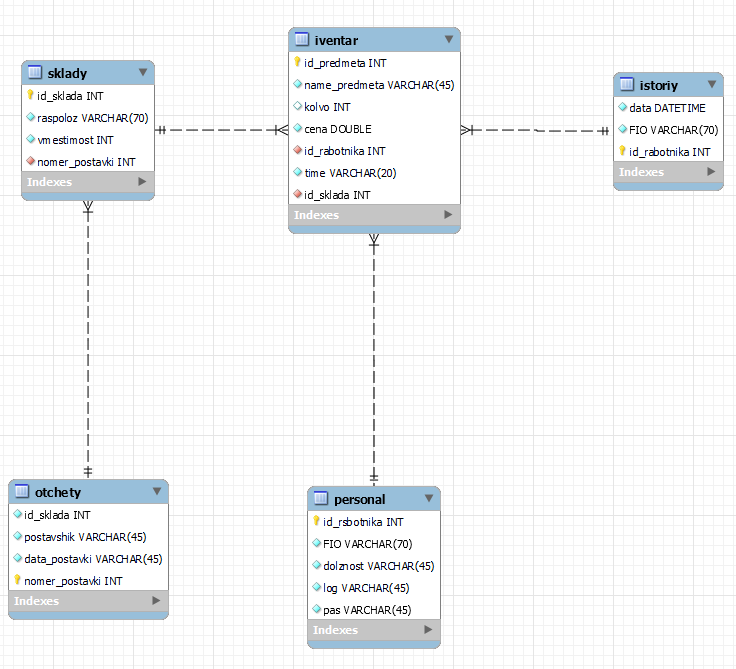


Рисунок 3.3 – Логическая схема

Структура таблиц базы данных представлена в таблицах 3.1 – 3.5.

Таблица 3.1 – Таблица «istoriy»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Описание | Тип данного | Размер |
| data | Дата входа | Дата | – |
| FIO | ФИО входившего | Текстовый | 70 |
| id\_rabotnika(PK) | ID входишвшего | Числовой | - |

Таблица 3.2 – Таблица «iventar»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Описание | Тип данного | Размер |
| id\_predmeta(PK) | ID предмета | Числовой | – |
| name\_predmeta | Наименование предмета | Текстовый | 45 |
| kolvo | Количество предметов | Числовой | - |
| cena | Цена предмета | Денежный | - |
| id\_rabotnika(FK) | ID работника | Числовой | - |
| time | Время добавления предмета | Дата | - |
| id\_sklada(FK) | ID склада в котором храниться предмет | Числовой | - |

Таблица 3.3 – Таблица «otchety»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Описание | Тип данного | Размер |
| id\_sklada(FK) | ID склада куда была поставка | Числовой | – |
| postavshik | Поставщик | Текстовый | 45 |
| data\_postavki | Дата поставки | Дата | - |
| nomer\_postavki(PK) | Номер поставки | Числовой | - |

Таблица 3.4 – Таблица «personal»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Описание | Тип данного | Размер |
| id\_rsbotnika(PK) | ID работника | Числовой | – |
| FIO | ФИО работника | Текстовый | 45 |
| dolznost | Должность работника | Тестовый | 45 |
| log | Логин работника | Тестовый | 45 |
| pas | Пароль работника | Тестовый | 45 |

Таблица 3.5 – Таблица «sklady»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Описание | Тип данного | Размер |
| id\_sklada | ID склада | Числовой | – |
| raspoloz | Месторасположение склада | Текстовый | 70 |
| vmestimost | Оставшееся свободное место | Числовой | - |
| nomer\_postavki | Номера поставок | Числовой | - |

4 Оптимизация программы

Оптимизация программы – это процесс модификации кода программы с целью улучшения её эффективности.

Основной задачей программирования является создание правильных, а не эффективных программ. Зачастую для повышения производительности программ большую значимость имеют такие факторы как:

четко выработанные требования к программе до начала ее разработки;

– хорошая архитектура программы;

– хорошо спроектированные модули;

– правильно выбранный компилятор со встроенными средствами оптимизации.

Ввиду данных факторов основной целью разработки являлось создание правильной, хорошо спроектированной программы. В большинстве случаев к оптимизации следует приступать только после того, как программа создана и выдает требуемые результаты. В противном случае можно получить не просто неудобочитаемый код и неработающую программу, но еще и неоптимизированное решение. Однако неизвестно, насколько эффективной будет программа, оптимизированная после ее разработки, поэтому определять требования к эффективности следует на стадии проектирования.

Несмотря на вышеперечисленные факты, оптимизация оказывается очень полезной, когда программа работает правильно. Производя небольшие изменения в некоторых фрагментах кода, называемых критическими областями, оптимизация помогает повысить эффективность программы в несколько раз.

Так как современные ЭВМ отличаются высоким быстродействием, то оптимизация отдельных, редко встречающихся операторов является бесполезной тратой времени. Однако повышение эффективности на мощных машинах можно получить за счет правильной оптимизации циклов и операторов, находящихся внутри тела циклов.

Программа, разрабатываемая в ходе практической работы, была оптимизирована по памяти. Программа не имеет постоянного подключения к базе данных, а подключается к ней только когда необходимо произвести манипуляции с данными из базы данных, после чего отключается от неё.

Вышеприведённый приём помог существенно повысить эффективность работы программы.

5 Тестирование программы

Тестирование — это набор процедур и действий, предназначенных для демонстрации правильности работы программы в заданных режимах и внешних условиях. Цель тестирования – выявить наличие ошибок или убедительно продемонстрировать их отсутствие. Процесс тестирования проходит в 3 этапа:

* проверка поведения программы в нормальных условиях;
* проверка поведения программы в экстремальных условиях;
* проверка поведения программы в исключительных ситуациях.

Каждый из этапов предполагает задание определенного, характерного для данного этапа набора данных.

5.1 Тестирование программы в нормальных условиях

Тестирование в нормальных условиях было проведено со всеми формами. Были протестированы переходы от формы к форме, корректная выборка данных из базы данных, автоматическое обновление форм. Особое внимание было уделено проверке регистрации нового пользователя, при котором обязательна проверка введенного пароля на соответствие требованиям безопасности. Тестирование процедуры регистрации нового пользователя представлено на рисунках 5.1, 5.2, 5.3.

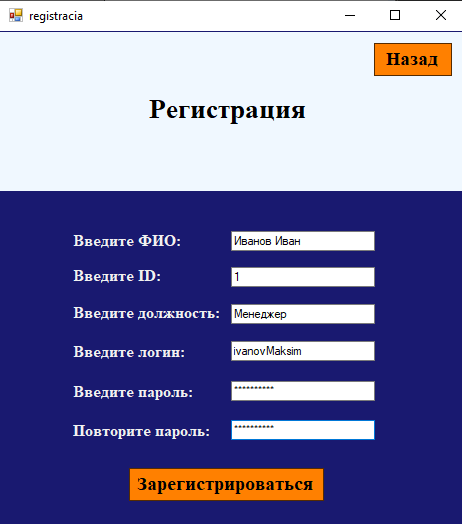


Рисунок 5.1 – Окно «Регистрация»

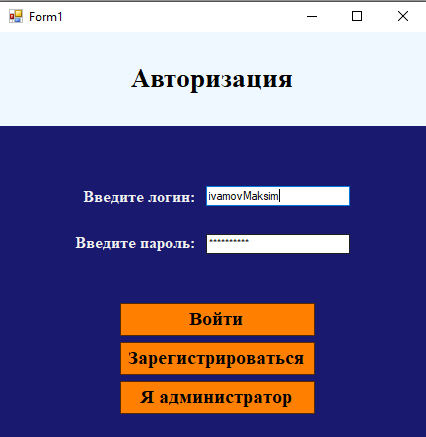


Рисунок 5.2 – Окно «Авторизация»

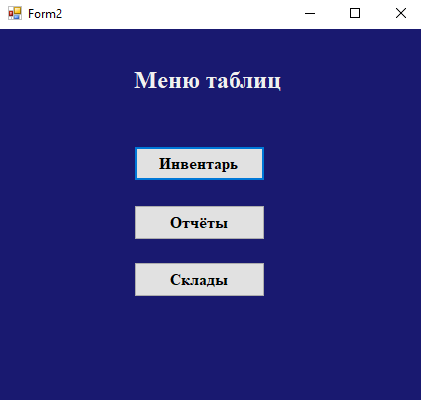


Рисунок 5.3 – Главное меню

Во время тестирования программы в нормальных условиях ошибок не выявлено, так как форма с главным меню, представленная на рисунке 5.3, появляется только в случае правильности ввода данных. Тестирование было подтверждено визуально, и оно показало, что программа работает правильно.

* 1. Тестирование программы в экстремальных условиях

Тестирование в экстремальных условиях было проведено со всеми формами. Была протестирована функция предотвращения ошибки при попытке добавить уже существующую запись в базу данных. Особое внимание было уделено проверке входа пользователя при котором обязательна проверка занятости логина и пароля. При попытке ввести в окно авторизации неправильный логин и пароль (рисунок 5.4) появляется сообщение об ошибке (рисунок 5.5).

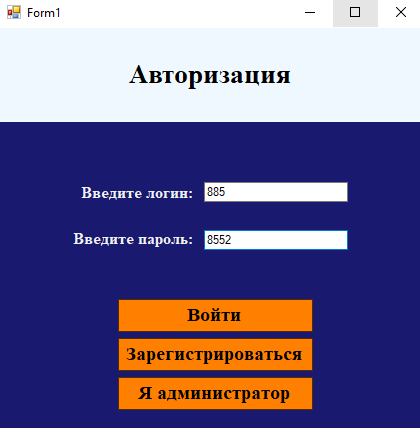


Рисунок 5.4 – Форма авторизации

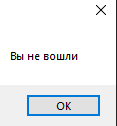


Рисунок 5.5 – Сообщение об ошибке

5.3 Тестирование программы в исключительных ситуациях

Тестирование в исключительных ситуациях было проведено со всеми формами. Была протестирована функция предотвращения ошибок при попытке отправить на сервер пустые поля. Особое внимание было уделено проверке входа пользователя при котором обязательна проверка полей. При попытке нажать на кнопку регистрации с пустыми полями ввода появляется сообщение об ошибке (рисунок 5.6).

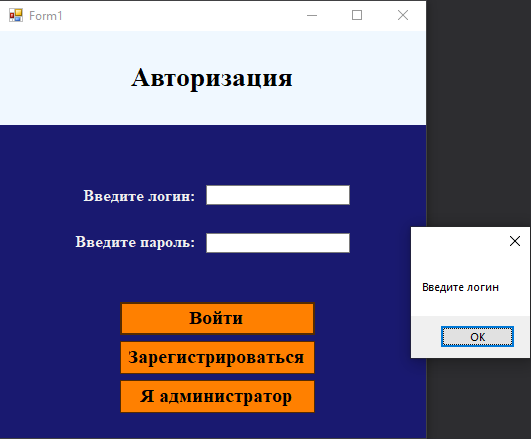


Рисунок 5.6 – Сообщение об ошибке

Так как тестирование в нормальных, экстремальных условиях и исключительных ситуациях не выявило наличие ошибок, можно сказать, что программа работает правильно. Листинг программы представлен в приложении А, а результаты программы – в приложении Б.

6 Руководство пользователя

Минимальные системные требования:

* Windows 7;
* процессор от 1,2 ГГц;
* ОЗУ от 64 МБ;
* свободное место на жестком диске 5 МБ;
* .NET Framework 4.5 и выше.

Для работы с программой следуйте пошаговому руководству, представленному ниже:

1. для запуска приложения необходимо запустить файл proga1.sln;
2. при запуске на экране появится форма для авторизации пользователя (рисунок 6.1) с возможностью переключения на форму регистрации (рисунок 6.2);

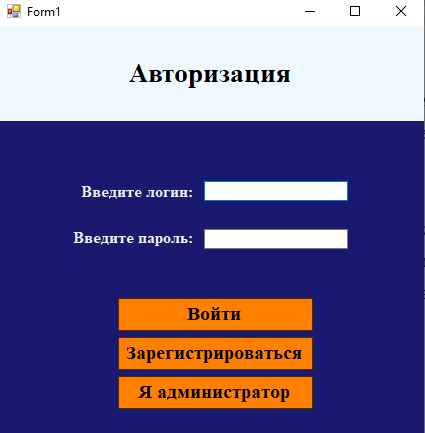


Рисунок 6.1 – Форма авторизации

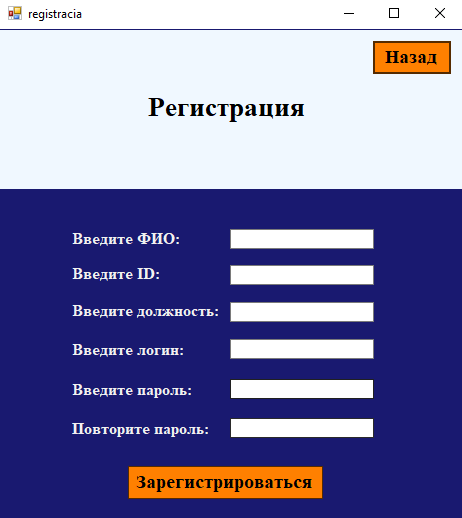


Рисунок 6.2 – Форма регистрации

1. после авторизации пользователь попадет в главное меню (рисунок 6.3), где можно будет перейти к нужной таблице;

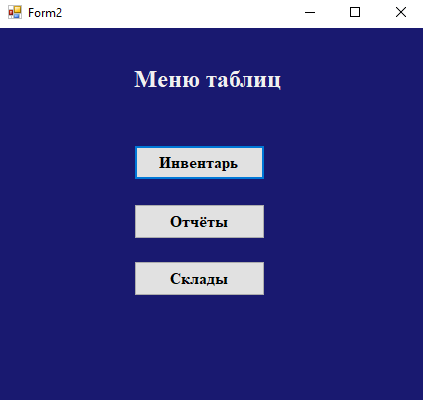
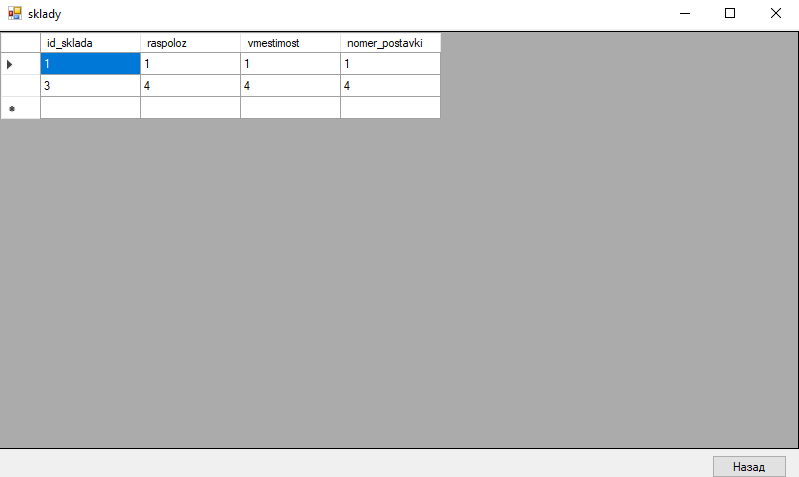


Рисунок 6.3 – Главное меню

1. при выборе одной из таблиц откроется соответствующее окно (рисунок 6.4);

 Рисунок 6.4 – Таблица «sklad»

1. после авторизации в роли администратора пользователь попадет в главное меню (рисунок 6.5), где можно будет перейти к нужной таблице;

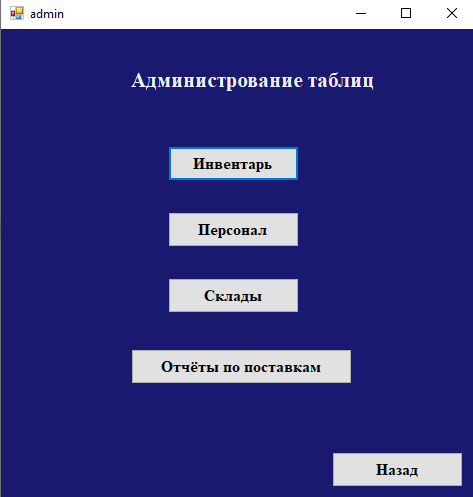


Рисунок 6.5 – Главное администратора

1. при выборе одной из таблиц откроется соответствующее окно (рисунок 6.6);

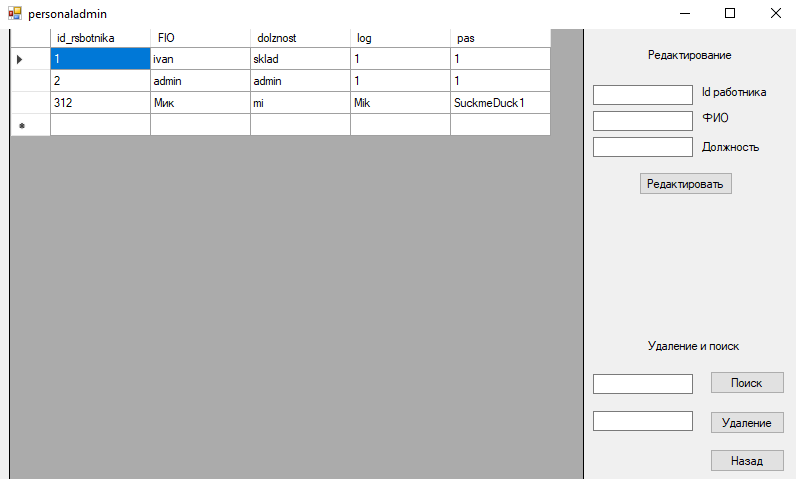


Рисунок 6.6 – Таблица «personal»

1. для добавления записей в таблицу необходимо заполнить соответствующие поля для ввода данных и нажать кнопку “Добавить” (рисунок 6.7);

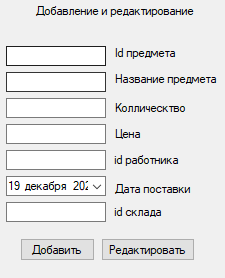


Рисунок 6.7 – Добавление данных

1. при нажатии на кнопку “Добавить” появится соответствующее сообщение (рисунок 6.8);

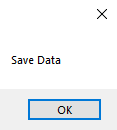


Рисунок 6.8 – Сообщение об успешном добавлении

1. при наличии пустых полей программа потребует их заполнения (рисунок 6.9);

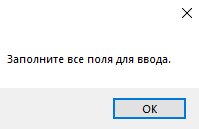


Рисунок 6.9 – Сообщение о наличии незаполненных полей для ввода

1. для поиска нужных данных необходимо ввести данные в соответствующей графе (рисунок 6.10);



Рисунок 6.10 – Поиск данных

1. после нажатия кнопки «поиск» будут выделены результаты поиска (рисунок 6.11);

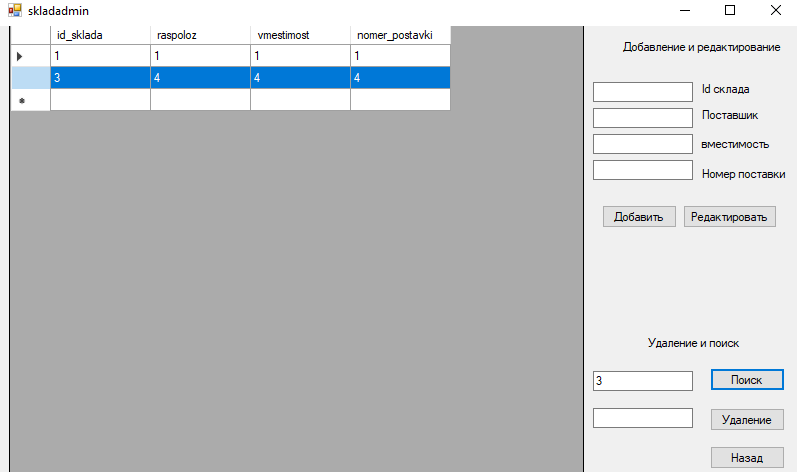


Рисунок 6.11 – Результаты поиска

1. для обновления данных необходимо заполнить поля для ввода данных, ввести id редактируемой строки в специальное поле и кликнуть на кнопку “Редактировать по ID” (рисунок 6.12);

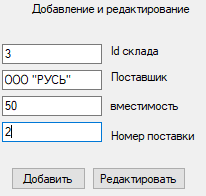


Рисунок 6.12 – Обновление данных

1. после нажатия кнопки «Редактировать по ID» появится сообщение (рисунок 6.11);

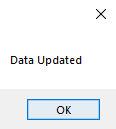


Рисунок 6.13 – Сообщение об успешном обновлении данных

1. при нажатии кнопки «Удаление» появится сообщение об успешном удалении (рисунок 6.14);



Рисунок 6.12 – Удаление данных

1. при нажатии кнопки «Удаление» появится сообщение об успешном удалении (рисунок 6.15);

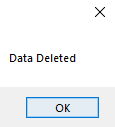


Рисунок 6.15 – Сообщение об успешном удалении

1. при нажатии кнопки «Назад» откроется главное меню (рисунок 6.16);



Рисунок 6.16 – Кнопка «Назад»

1. для выхода из программы нажмите кнопку «X» в любой форме приложения (рисунок 6.17).



Рисунок 6.17 – Кнопка закрытия приложения

ЗАКЛЮЧЕНИЕ

Основная задача практической работы – создание информационной системы для работы с базой данных. Поставленная задача была полностью выполнена в процессе реализации данного проекта.

В ходе создания данной программы были получены практические навыки в области проектирования программы для базы данных. Опыт, полученный при написании работы подтвердил необходимость в первую очередь выработки правильных требований к программе и грамотного подхода к её проектированию перед началом написания кода.

Также были получены практические навыки в области оптимизации и тестирования программы.

Результатом проведенной работы является правильно спроектированное приложение, а также работающий, эффективный, отлаженный и корректно отвечающий тестам проект.

При разработке программы были изучены и применены такие элементы управления как: TextBox, DataGridView и другие.

Программа имеет надлежащую структуру и необходимые комментарии.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Верзух Э. [пер. с англ. О. Л. Пелявского] Управление проектами: ускоренный курс по программе MBA / М., С-П: Диалектика, 2017
2. Грашина М. Н. Основы управления проектами / М.: БИНОМ. Лаборатория знаний, 2015
3. Желязны Д. Говори на языке диаграмм. – М: Манн, Иванов и Фербер, 3-е издание, 2016
4. Иванов П. В. Управление проектами / учеб. Пособие / Издательство: Феникс, Ростов-на-Дону, 2016
5. Кремлев Н.Д. Управление инновационными проектами / Учебное пособие М.: НИЦ ИНФРА-М, 2017
6. Лич Л. [пер. с англ. У. Саламатовой] Вовремя и в рамках бюджета: Управление проектами по методу критической цепи / М.: Альпина Паблишер, 2016
7. Мороз О. А. Управление проектами в ProjectLibre [учебно-практическое пособие] / Издательство: Феникс, Ростов-на-Дону, 2015
8. Ньютон Р. Управление проектами от А до Я [Пер. с англ.] / М.: Альпина Паблишер, 2016
9. Петин В. А. Проекты с использованием контроллера Arduino / С-П: БХВ-Петербург, 2017
10. Снедакер С. Управление IT-проектом, или Как стать полноценным CIO [пер. с англ.] / М.: ДМК Пресс, 2016
11. Тихомирова О. Г. Управление проектом: комплексный подход и системный анализ / Монография М.: ИНФРА-М, 2017
12. Тихомирова О. Г. Управление проектами: практикум / Учебное пособие / М.: Инфра-М, 2017
13. Управление проектом: основы проектного управления / учебник для студентов вузов под. ред. М. Л. Разу /М.: КноРус, 2018
14. Управление проектами: учебник и практикум для среднего профессионального образования / А. И. Балашов, Е. М. Рогова, М. В. Тихонова, Е. А. Ткаченко ; под общей редакцией Е. М. Роговой. — Москва : Издательство Юрайт, 2019
15. Управление проектом в сфере графического дизайна [Пер. с англ.] / М.: Альпина Паблишер, 2016
16. Хелдман К [Пер. с англ. А. В. Шаврина] Профессиональное управление проектом / М.: Лаборатория знаний, Москва, 2016
17. Шаврин А. В. Руководство по управлению проектами на основе стандарта ISO 21500 / М.: Лаборатория знаний, 2017
18. Краткая документация по ProjectLibre / [Электронный ресурс] – Режим доступа: http://kostigoff.ru/projectlibre.15/ (проверено 19.08.2017)
19. ООО «Спутникс» https://sputnix.ru/ru/
20. Открытая замена коммерческому продукту Microsoft Project / ProjectLibre [Электронный ресурс] – Режим доступа: http://www.projectlibre.com/ (проверено 19.08.2017)
21. Основы управления проектами / [Электронный ресурс] – Режим доступа: http://www.studfiles.ru/preview/6325721/ (проверено 19.08.2017)
22. Разработка проекта / [Электронный ресурс] – Режим доступа: http://www.studfiles.ru/preview/2905944/page:3/(проверено 19.08.2017)
23. Создание проекта в ProjectLibre, видеоурок / [Электронный ресурс] – Режим доступа: https://www.youtube.com/watch?v=aTuCTUjZ5WM (проверено 19.08.2017)
24. Управление проектами, видеоуроки / [Электронный ресурс] – Режим доступа: https://www.youtube.com/watch?v=F7yMJos3SRk&app=d.. (проверено 19.08.2017)
25. ЭБС biblio-online.ru

ПРИЛОЖЕНИЕ А

(обязательное)

Листинг программы

Листинг программы «Практика»

/\*

Практическая работа по профессиональному модулю ПМ.02 «Осуществление интеграции программных модулей».

Название: «Практика».

Разработал: Бобылёв Сергей Сергеевич.

Группа: ТИП-61.

Дата: 12.06.2021год.

Краткое описание: данная программа является информационной системой для отдела ТСО.

Задание:

а) Создать базу данных, используя наиболее подходящую к выданному заданию платформу (MySQL / Microsoft SQL Server).

б) Создать приложение, используя наиболее приемлемую для решения задачи платформу: .NET (или Java).

в) Создать форму для авторизации зарегистрированных пользователей.

г) В системе, в зависимости от предметной области должны быть выделены не менее 2 различных типов пользователей.

д) Пользователи должны входить в систему, используя форму Авторизации:

1) Для успешной авторизации необходимо ввести свой логин и пароль.

2) При попытке входа введенные данные проверяются на совпадение в базе данных. Должно выдаваться сообщение об ошибке в случае неправильного ввода связки логин/пароль.

3) После успешного входа пользователя в зависимости от его роли происходит перенаправление на конкретные формы, предназначенные для работы этого типа пользователя.

4) При нажатии на кнопку "Выход" в любом окне/форме системы, пользователи должны быть отключены от системы и совершен переход на Форму авторизации.

5) Необходимо предусмотреть возможность регистрации Новых пользователей определенной роли. Поэтому нужно добавить в систему форму регистрации. При регистрации пароль должен отвечать следующим требованиям:

– должен содержать от 6 до 16 символов;

– не должно быть символов из набора: \* & { } | +;

– должны встречаться заглавные буквы;

– должны встречаться цифры.

е) Должны быть реализованы основные формы для работы с сущностями, реализованными в базе данных.

\*/

//листинг начальной формы

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace производственная\_практика

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

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

{

if (avtlog.Text == "")

{

MessageBox.Show("Введите логин");

return;

}

if (avtpas.Text == "")

{

MessageBox.Show("Введите пароль");

return;

}

string login = avtlog.Text;

string password = avtpas.Text;

db DB = new db();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT \* FROM personal WHERE log = @userlogin AND pas = @userpassword", DB.GetConnection());

command.Parameters.Add("@userlogin", MySqlDbType.VarChar).Value = login;

command.Parameters.Add("@userpassword", MySqlDbType.VarChar).Value = password;

adapter.SelectCommand = command;

adapter.Fill(table);

if (table.Rows.Count > 0)

{

MessageBox.Show("Вы авторизоывались");

this.Hide();

tables tables = new tables();

tables.Show();

}

else

MessageBox.Show("Вы не вошли");

}

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

{

this.Hide();

reg reg= new reg();

reg.Show();

}

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

{

this.Hide();

admintables admintables = new admintables();

admintables.Show();

}

}

}

//листинг формы регистрации

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace производственная\_практика

{

public partial class reg : Form

{

public reg()

{

InitializeComponent();

}

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

{

if (zaregfio.Text == "")

{

MessageBox.Show("Введите ФИО");

return;

}

if (zaregid.Text == "")

{

MessageBox.Show("Введите ID");

return;

}

if (zaregdolz.Text == "")

{

MessageBox.Show("Введите вашу должность");

return;

}

if (zareglog.Text == "")

{

MessageBox.Show("Введите логин");

return;

}

if (zaregpas.Text == "")

{

MessageBox.Show("Введите пароль");

return;

}

if (zaregpas.Text != repitpas.Text)

{

MessageBox.Show("Вы повторили пароль не верно");

return;

}

if (zaregpas.Text == zareglog.Text)

{

MessageBox.Show("Ошибка - Логин не может быть равень паролю");

zareglog.Text = "";

zaregpas.Text = "";

return;

}

{

if (zaregpas.MaxLength < 20 && zareglog.MaxLength < 20)

{

MessageBox.Show("Ошибка - Логин или пароль больше 20 символов");

zareglog.Text = "";

zaregpas.Text = "";

return;

}

{

if (zaregpas.Text.Length < 5 && zareglog.Text.Length < 5)

{

MessageBox.Show("Ошибка - Логин или пароль меньше 5 символов");

zareglog.Text = "";

zaregpas.Text = "";

return;

}

{

int p = 0;

for (int i = 0; i < zaregpas.Text.Length; i++)

{

if (char.IsLower(zaregpas.Text[i]))

{

p = 1;

}

}

if (p == 0)

{

MessageBox.Show("Должна быть хоть одна маленькая буква");

zareglog.Text = "";

zaregpas.Text = "";

return;

}

{

int l = 0;

for (int i = 0; i < zaregpas.Text.Length; i++)

{

if (char.IsUpper(zaregpas.Text[i]))

{

l = 1;

}

}

if (l == 0)

{

MessageBox.Show("Должна быть хоть одна заглавная буква");

zareglog.Text = "";

zaregpas.Text = "";

return;

}

}

}

}

}

if (checkUser())

return;

db DB = new db();

MySqlCommand command = new MySqlCommand ("INSERT INTO personal (id\_rsbotnika, FIO, dolznost, log, pas ) VALUES (@id, @fio, @dolz, @login, @pass)", DB.GetConnection());

command.Parameters.Add("@fio", MySqlDbType.VarChar).Value = zaregfio.Text;

command.Parameters.Add("@id", MySqlDbType.VarChar).Value = zaregid.Text;

command.Parameters.Add("@dolz", MySqlDbType.VarChar).Value = zaregdolz.Text;

command.Parameters.Add("@login", MySqlDbType.VarChar).Value = zareglog.Text;

command.Parameters.Add("@pass", MySqlDbType.VarChar).Value = zaregpas.Text;

DB.openConnection();

if (command.ExecuteNonQuery() == 1)

{

MessageBox.Show("Аккаунт создан");

this.Hide();

Form1 Form1 = new Form1();

Form1.Show();

}

else

MessageBox.Show("Аккаунт не создан");

DB.closeConnection();

}

public Boolean checkUser()

{

db DB = new db();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT \* FROM personal WHERE id\_rsbotnika = @id", DB.GetConnection());

command.Parameters.Add("@id", MySqlDbType.VarChar).Value = zaregid.Text;

adapter.SelectCommand = command;

adapter.Fill(table);

if (table.Rows.Count > 0)

{

MessageBox.Show("Данный ID уже занят");

return true;

}

else

return false;

}

private void button1\_Click(object sender, EventArgs e)

{

this.Hide();

Form1 Form1 = new Form1();

Form1.Show();

}

}

}

//листинг формы входа для администратора

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace производственная\_практика

{

public partial class admintables : Form

{

public admintables()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

if (adminlog.Text == "")

{

MessageBox.Show("Введите логин");

return;

}

if (adminpass.Text == "")

{

MessageBox.Show("Введите пароль");

return;

}

string login = adminlog.Text;

string password = adminpass.Text;

string dolz = "admin";

db DB = new db();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT \* FROM personal WHERE log = @userlogin AND pas = @userpassword AND dolznost = @dolznost", DB.GetConnection());

command.Parameters.Add("@userlogin", MySqlDbType.VarChar).Value = login;

command.Parameters.Add("@userpassword", MySqlDbType.VarChar).Value = password;

command.Parameters.Add("@dolznost", MySqlDbType.VarChar).Value = dolz;

adapter.SelectCommand = command;

adapter.Fill(table);

if (table.Rows.Count > 0)

{

MessageBox.Show("Вы авторизоывались");

this.Hide();

admin admin = new admin();

admin.Show();

}

else

MessageBox.Show("Вы не вошли");

}

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

{

this.Hide();

Form1 Form1 = new Form1();

Form1.Show();

}

private void admintables\_Load(object sender, EventArgs e)

{

}

}

}

//листинг формы выбора таблицы пользователя

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace производственная\_практика

{

public partial class tables : Form

{

public tables()

{

InitializeComponent();

}

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

{

this.Hide();

inventary inventary = new inventary();

inventary.Show();

}

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

{

this.Hide();

otchety otchety = new otchety();

otchety.Show();

}

private void button1\_Click(object sender, EventArgs e)

{

this.Hide();

sklady sklady = new sklady();

sklady.Show();

}

private void tables\_Load(object sender, EventArgs e)

{

}

}

}

//листинг просмотра таблицы инвентарь от лица пользователя

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace производственная\_практика

{

public partial class inventary : Form

{

public inventary()

{

InitializeComponent();

}

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

{

this.Hide();

tables tables = new tables();

tables.Show();

}

private void inventary\_Load(object sender, EventArgs e)

{

db DB = new db();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command = new MySqlCommand("SELECT \* FROM iventar ", DB.GetConnection());

adapter.SelectCommand = command;

adapter.Fill(table);

otobraz.DataSource = table;

}

}

}

//листинг просмотра таблицы

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace производственная\_практика

{

public partial class otchety : Form

{

public otchety()

{

InitializeComponent();

}

private void otchety\_Load(object sender, EventArgs e)

{

db DB = new db();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command = new MySqlCommand("SELECT \* FROM otchety ", DB.GetConnection());

adapter.SelectCommand = command;

adapter.Fill(table);

otobraz.DataSource = table;

}

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

{

this.Hide();

tables tables = new tables();

tables.Show();

}

}

}

//листинг просмотра таблицы склад от лица пользователя

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace производственная\_практика

{

public partial class sklady : Form

{

public sklady()

{

InitializeComponent();

}

private void sklady\_Load(object sender, EventArgs e)

{

db DB = new db();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command = new MySqlCommand("SELECT \* FROM sklady ", DB.GetConnection());

adapter.SelectCommand = command;

adapter.Fill(table);

otobraz.DataSource = table;

}

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

{

this.Hide();

tables tables = new tables();

tables.Show();

}

}

}

//просмотр таблицы инвентарь от лица администратора

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace производственная\_практика

{

public partial class invetryadmin : Form

{

public invetryadmin()

{

InitializeComponent();

}

private void invetryadmin\_Load(object sender, EventArgs e)

{

db DB = new db();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command = new MySqlCommand("SELECT \* FROM iventar ", DB.GetConnection());

adapter.SelectCommand = command;

adapter.Fill(table);

otobraz.DataSource = table;

}

private void button2\_Click(object sender, EventArgs e)

{

db DB = new db();

try

{ // Формирование запроса

MySqlCommand command = new MySqlCommand("insert into iventar (id\_predmeta,name\_predmeta,kolvo,cena,id\_rabotnika,time,id\_sklada) values(@id,@name,@kol,@cen,@rab,@data,@sklad);", DB.GetConnection());

command.Parameters.Add("@id", MySqlDbType.VarChar).Value = idsk.Text;

command.Parameters.Add("@name", MySqlDbType.VarChar).Value = post.Text;

command.Parameters.Add("@kol", MySqlDbType.VarChar).Value = textBox1.Text;

command.Parameters.Add("@cen", MySqlDbType.VarChar).Value = nomer.Text;

command.Parameters.Add("@rab", MySqlDbType.VarChar).Value = textBox2.Text;

command.Parameters.Add("@data", MySqlDbType.VarChar).Value = data.Text;

command.Parameters.Add("@sklad", MySqlDbType.VarChar).Value = textBox3.Text;

MySqlDataReader MyReader2;

DB.openConnection();

MyReader2 = command.ExecuteReader();

MessageBox.Show("Save Data");

while (MyReader2.Read())

{

}

DB.closeConnection();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

} // Добавление данных

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command2 = new MySqlCommand("Select \* From iventar ", DB.GetConnection());

adapter.SelectCommand = command2;

adapter.Fill(table);

otobraz.DataSource = table;

}

private void button1\_Click(object sender, EventArgs e)

{

db DB = new db();

try

{

// Формирование запроса

MySqlCommand command = new MySqlCommand("update iventar set name\_predmeta = @name, kolvo = @kol, cena=@cen, id\_rabotnika=@rab,time=@data, id\_sklada=@sklad where id\_predmeta = @id", DB.GetConnection());

command.Parameters.Add("@id", MySqlDbType.VarChar).Value = idsk.Text;

command.Parameters.Add("@name", MySqlDbType.VarChar).Value = post.Text;

command.Parameters.Add("@kol", MySqlDbType.VarChar).Value = textBox1.Text;

command.Parameters.Add("@cen", MySqlDbType.VarChar).Value = nomer.Text;

command.Parameters.Add("@rab", MySqlDbType.VarChar).Value = textBox2.Text;

command.Parameters.Add("@data", MySqlDbType.VarChar).Value = data.Text;

command.Parameters.Add("@sklad", MySqlDbType.VarChar).Value = textBox3.Text;

MySqlDataReader MyReader2;

DB.openConnection();

MyReader2 = command.ExecuteReader();

MessageBox.Show("Data Updated");

while (MyReader2.Read())

{

}

DB.closeConnection();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

} // Редактирование данных

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command2 = new MySqlCommand("Select \* From iventar ", DB.GetConnection());

adapter.SelectCommand = command2;

adapter.Fill(table);

otobraz.DataSource = table;

}

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

{

for (int i = 0; i < otobraz.RowCount; i++)

{

otobraz.Rows[i].Selected = false;

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

if (otobraz.Rows[i].Cells[j].Value != null)

if

(otobraz.Rows[i].Cells[j].Value.ToString().Contains(idseasrch.Text))

{

otobraz.Rows[i].Selected = true;

break;

}

}

}

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

{

{//Проверка на наличие незаполненных полей

if (iddel.Text == "")

{

MessageBox.Show("Заполните поле ID.");

return;

}

db DB = new db();

try

{

// Формирование запроса

MySqlCommand command = new MySqlCommand("DELETE FROM iventar WHERE id\_predmeta=@nomer", DB.GetConnection());

command.Parameters.Add("@nomer", MySqlDbType.Int32).Value = iddel.Text;

MySqlDataReader MyReader2;

DB.openConnection();

MyReader2 = command.ExecuteReader();

MessageBox.Show("Data Deleted");

while (MyReader2.Read())

{

}

DB.closeConnection();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

} // Удаление данных

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command2 = new MySqlCommand("SELECT \* FROM iventar ", DB.GetConnection());

adapter.SelectCommand = command2;

adapter.Fill(table);

otobraz.DataSource = table;

}

}

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

{

this.Hide();

admin admin = new admin();

admin.Show();

}

}

}

//листинг таблицы выбора таблиц от лица администратора

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace производственная\_практика

{

public partial class admin : Form

{

public admin()

{

InitializeComponent();

}

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

{

this.Hide();

invetryadmin invetryadmin = new invetryadmin();

invetryadmin.Show();

}

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

{

this.Hide();

personaladmin personaladmin = new personaladmin();

personaladmin.Show();

}

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

{

this.Hide();

adminotch adminotch = new adminotch();

adminotch.Show();

}

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

{

this.Hide();

Form1 Form1 = new Form1();

Form1.Show();

}

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

{

this.Hide();

skladadmin skladadmin = new skladadmin();

skladadmin.Show();

}

private void admin\_Load(object sender, EventArgs e)

{

}

}

}

//листинг таблицы персонал от лица администратора

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace производственная\_практика

{

public partial class personaladmin : Form

{

public personaladmin()

{

InitializeComponent();

}

private void personaladmin\_Load(object sender, EventArgs e)

{

db DB = new db();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command = new MySqlCommand("SELECT \* FROM personal ", DB.GetConnection());

adapter.SelectCommand = command;

adapter.Fill(table);

otobraz.DataSource = table;

}

private void button1\_Click(object sender, EventArgs e)

{

db DB = new db();

try

{

// Формирование запроса

MySqlCommand command = new MySqlCommand("update personal set FIO = @fio, dolznost = @post where id\_rsbotnika = @id", DB.GetConnection());

command.Parameters.Add("@id", MySqlDbType.Int32).Value = idsk.Text;

command.Parameters.Add("@fio", MySqlDbType.VarChar).Value = post.Text;

command.Parameters.Add("@post", MySqlDbType.VarChar).Value = data.Text;

MySqlDataReader MyReader2;

DB.openConnection();

MyReader2 = command.ExecuteReader();

MessageBox.Show("Data Updated");

while (MyReader2.Read())

{

}

DB.closeConnection();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

} // Редактирование данных

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command2 = new MySqlCommand("Select \* From personal ", DB.GetConnection());

adapter.SelectCommand = command2;

adapter.Fill(table);

otobraz.DataSource = table;

}

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

{

for (int i = 0; i < otobraz.RowCount; i++)

{

otobraz.Rows[i].Selected = false;

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

if (otobraz.Rows[i].Cells[j].Value != null)

if

(otobraz.Rows[i].Cells[j].Value.ToString().Contains(idseasrch.Text))

{

otobraz.Rows[i].Selected = true;

break;

}

}

}

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

{

{//Проверка на наличие незаполненных полей

if (iddel.Text == "")

{

MessageBox.Show("Заполните поле ID.");

return;

}

db DB = new db();

try

{

// Формирование запроса

MySqlCommand command = new MySqlCommand("DELETE FROM personal WHERE id\_rsbotnika=@nomerpost", DB.GetConnection());

command.Parameters.Add("@nomerpost", MySqlDbType.Int32).Value = iddel.Text;

MySqlDataReader MyReader2;

DB.openConnection();

MyReader2 = command.ExecuteReader();

MessageBox.Show("Data Deleted");

while (MyReader2.Read())

{

}

DB.closeConnection();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

} // Удаление данных

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command2 = new MySqlCommand("SELECT \* FROM personal ", DB.GetConnection());

adapter.SelectCommand = command2;

adapter.Fill(table);

otobraz.DataSource = table;

}

}

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

{

this.Hide();

admin admin = new admin();

admin.Show();

}

}

}

//листинг таблицы склад от лица администратора

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace производственная\_практика

{

public partial class skladadmin : Form

{

public skladadmin()

{

InitializeComponent();

}

private void skladadmin\_Load\_1(object sender, EventArgs e)

{

db DB = new db();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command = new MySqlCommand("SELECT \* FROM sklady ", DB.GetConnection());

adapter.SelectCommand = command;

adapter.Fill(table);

otobraz.DataSource = table;

}

private void search\_Click\_1(object sender, EventArgs e)

{

for (int i = 0; i < otobraz.RowCount; i++)

{

otobraz.Rows[i].Selected = false;

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

if (otobraz.Rows[i].Cells[j].Value != null)

if

(otobraz.Rows[i].Cells[j].Value.ToString().Contains(idseasrch.Text))

{

otobraz.Rows[i].Selected = true;

break;

}

}

}

private void button2\_Click\_1(object sender, EventArgs e)

{

db DB = new db();

try

{ // Формирование запроса

MySqlCommand command = new MySqlCommand("insert into sklady (id\_sklada,raspoloz,vmestimost,nomer\_postavki) values(@id,@postav,@data,@nomerpost);", DB.GetConnection());

command.Parameters.Add("@id", MySqlDbType.VarChar).Value = idsk.Text;

command.Parameters.Add("@postav", MySqlDbType.VarChar).Value = post.Text;

command.Parameters.Add("@data", MySqlDbType.VarChar).Value = data.Text;

command.Parameters.Add("@nomerpost", MySqlDbType.VarChar).Value = nomer.Text;

MySqlDataReader MyReader2;

DB.openConnection();

MyReader2 = command.ExecuteReader();

MessageBox.Show("Save Data");

while (MyReader2.Read())

{

}

DB.closeConnection();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

} // Добавление данных

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command2 = new MySqlCommand("Select \* From sklady ", DB.GetConnection());

adapter.SelectCommand = command2;

adapter.Fill(table);

otobraz.DataSource = table;

}

private void button1\_Click\_1(object sender, EventArgs e)

{

db DB = new db();

try

{

// Формирование запроса

MySqlCommand command = new MySqlCommand("update sklady set raspoloz = @id, vmestimost = @post, nomer\_postavki = @data where id\_sklada = @nomer", DB.GetConnection());

command.Parameters.Add("@id", MySqlDbType.Int32).Value = idsk.Text;

command.Parameters.Add("@post", MySqlDbType.VarChar).Value = post.Text;

command.Parameters.Add("@data", MySqlDbType.VarChar).Value = data.Text;

command.Parameters.Add("@nomer", MySqlDbType.Int32).Value = nomer.Text;

MySqlDataReader MyReader2;

DB.openConnection();

MyReader2 = command.ExecuteReader();

MessageBox.Show("Data Updated");

while (MyReader2.Read())

{

}

DB.closeConnection();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

} // Редактирование данных

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command2 = new MySqlCommand("Select \* From sklady ", DB.GetConnection());

adapter.SelectCommand = command2;

adapter.Fill(table);

otobraz.DataSource = table;

}

private void del\_Click\_1(object sender, EventArgs e)

{

{//Проверка на наличие незаполненных полей

if (iddel.Text == "")

{

MessageBox.Show("Заполните поле ID.");

return;

}

db DB = new db();

try

{

// Формирование запроса

MySqlCommand command = new MySqlCommand("DELETE FROM sklady WHERE id\_sklada=@nomerpost", DB.GetConnection());

command.Parameters.Add("@nomerpost", MySqlDbType.Int32).Value = iddel.Text;

MySqlDataReader MyReader2;

DB.openConnection();

MyReader2 = command.ExecuteReader();

MessageBox.Show("Data Deleted");

while (MyReader2.Read())

{

}

DB.closeConnection();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

} // Удаление данных

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command2 = new MySqlCommand("SELECT \* FROM sklady ", DB.GetConnection());

adapter.SelectCommand = command2;

adapter.Fill(table);

otobraz.DataSource = table;

}

}

private void back\_Click\_1(object sender, EventArgs e)

{

this.Hide();

admin admin = new admin();

admin.Show();

}

}

}

//листинг таблицы отчёты от лица администратора

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace производственная\_практика

{

public partial class adminotch : Form

{

public adminotch()

{

InitializeComponent();

}

private void adminotch\_Load(object sender, EventArgs e)

{

db DB = new db();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command = new MySqlCommand("SELECT \* FROM otchety ", DB.GetConnection());

adapter.SelectCommand = command;

adapter.Fill(table);

otobraz.DataSource = table;

}

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

{

this.Hide();

admin admin = new admin();

admin.Show();

}

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

{

{//Проверка на наличие незаполненных полей

if (iddel.Text == "")

{

MessageBox.Show("Заполните поле ID.");

return;

}

db DB = new db();

try

{

// Формирование запроса

MySqlCommand command = new MySqlCommand("DELETE FROM otchety WHERE nomer\_postavki=@nomer", DB.GetConnection());

command.Parameters.Add("@nomer", MySqlDbType.Int32).Value = iddel.Text;

MySqlDataReader MyReader2;

DB.openConnection();

MyReader2 = command.ExecuteReader();

MessageBox.Show("Data Deleted");

while (MyReader2.Read())

{

}

DB.closeConnection();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

} // Удаление данных

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command2 = new MySqlCommand("SELECT \* FROM otchety ", DB.GetConnection());

adapter.SelectCommand = command2;

adapter.Fill(table);

otobraz.DataSource = table;

}

}

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

{

for (int i = 0; i < otobraz.RowCount; i++)

{

otobraz.Rows[i].Selected = false;

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

if (otobraz.Rows[i].Cells[j].Value != null)

if

(otobraz.Rows[i].Cells[j].Value.ToString().Contains(idseasrch.Text))

{

otobraz.Rows[i].Selected = true;

break;

}

}

}

private void button2\_Click(object sender, EventArgs e)

{

db DB = new db();

try

{ // Формирование запроса

MySqlCommand command = new MySqlCommand("insert into otchety (id\_sklada,postavshik,data\_postavki,nomer\_postavki) values(@id,@postav,@data,@nomerpost);", DB.GetConnection());

command.Parameters.Add("@id", MySqlDbType.VarChar).Value = idsk.Text;

command.Parameters.Add("@postav", MySqlDbType.VarChar).Value = post.Text;

command.Parameters.Add("@data", MySqlDbType.VarChar).Value = data.Text;

command.Parameters.Add("@nomerpost", MySqlDbType.VarChar).Value = nomer.Text;

MySqlDataReader MyReader2;

DB.openConnection();

MyReader2 = command.ExecuteReader();

MessageBox.Show("Save Data");

while (MyReader2.Read())

{

}

DB.closeConnection();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

} // Добавление данных

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command2 = new MySqlCommand("Select \* From otchety ", DB.GetConnection());

adapter.SelectCommand = command2;

adapter.Fill(table);

otobraz.DataSource = table;

}

private void button1\_Click(object sender, EventArgs e)

{

db DB = new db();

try

{

// Формирование запроса

MySqlCommand command = new MySqlCommand("update otchety set id\_sklada = @id, postavshik = @post, data\_postavki = @data where nomer\_postavki = @nomer", DB.GetConnection());

command.Parameters.Add("@id", MySqlDbType.VarChar).Value = idsk.Text;

command.Parameters.Add("@post", MySqlDbType.VarChar).Value = post.Text;

command.Parameters.Add("@data", MySqlDbType.VarChar).Value = data.Text;

command.Parameters.Add("@nomer", MySqlDbType.VarChar).Value = nomer.Text;

MySqlDataReader MyReader2;

DB.openConnection();

MyReader2 = command.ExecuteReader();

MessageBox.Show("Data Updated");

while (MyReader2.Read())

{

}

DB.closeConnection();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

} // Редактирование данных

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

// Формирование запроса

MySqlCommand command2 = new MySqlCommand("Select \* From otchety ", DB.GetConnection());

adapter.SelectCommand = command2;

adapter.Fill(table);

otobraz.DataSource = table;

}

}

}

ПРИЛОЖЕНИЕ Б

(обязательное)

Результаты выполнения программы

На рисунке Б.1 представлена форма авторизации пользователя.

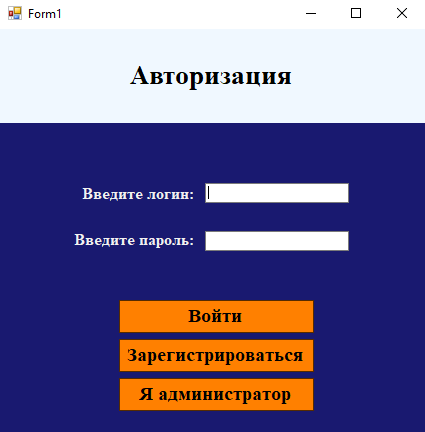


Рисунок Б.1 – Форма авторизации

На рисунке Б.2 представлена форма регистрации нового пользователя.

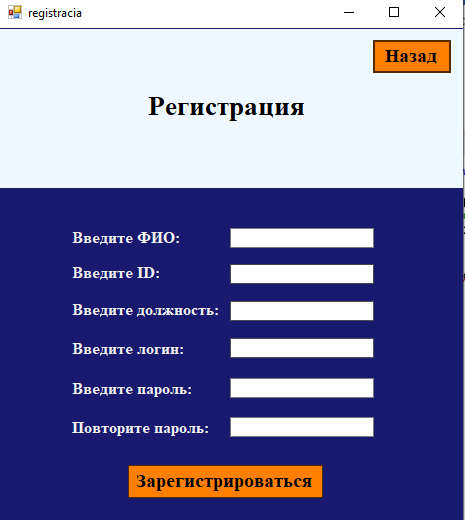


Рисунок Б.2 – Форма регистрации

На рисунке Б.3 представлена форма авторизации для администратора.

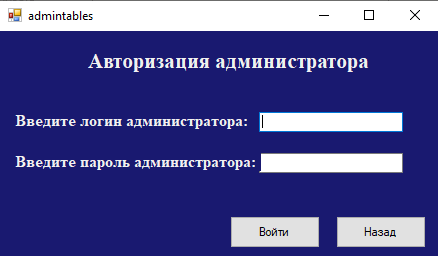


Рисунок Б.3 – Форма авторизации администратора

На рисунке Б.4 представлена форма главного меню пользователя.

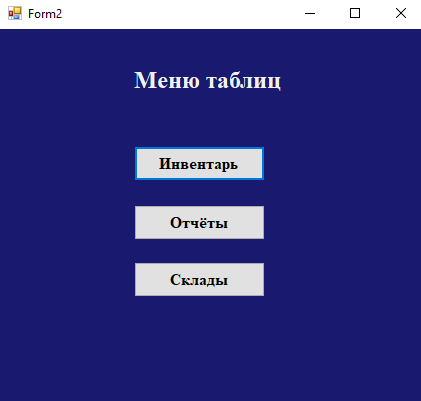


Рисунок Б.4 – Форма главного меню

На рисунке Б.5 представлена форма таблицы «инвентарь».

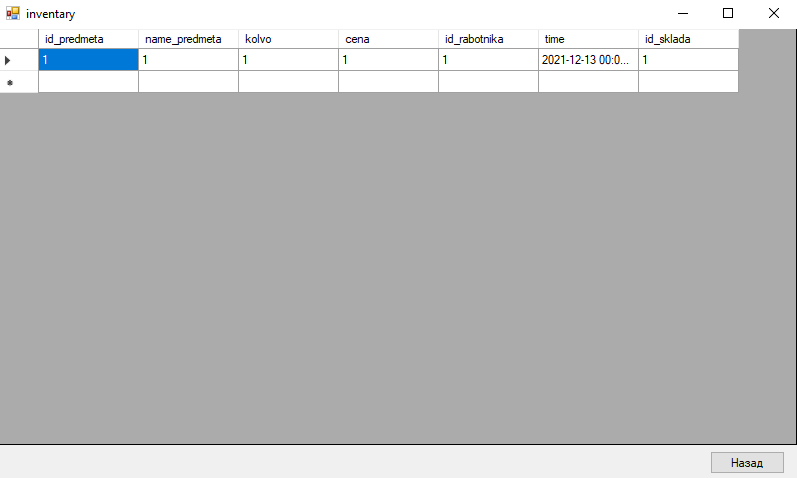
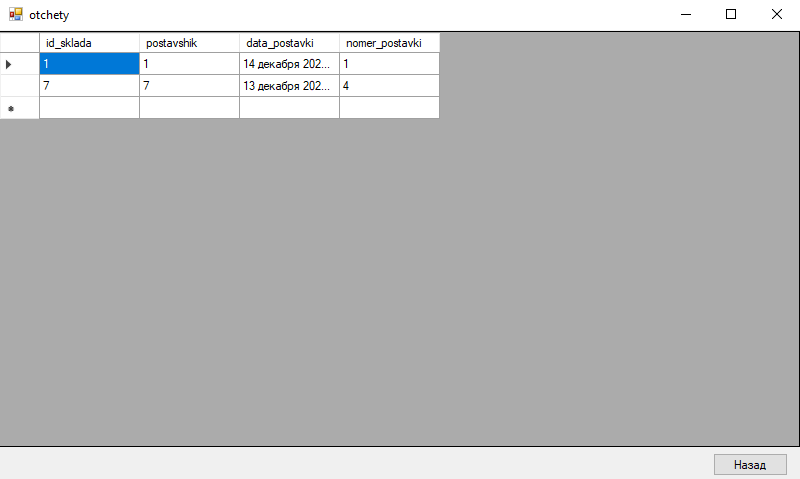
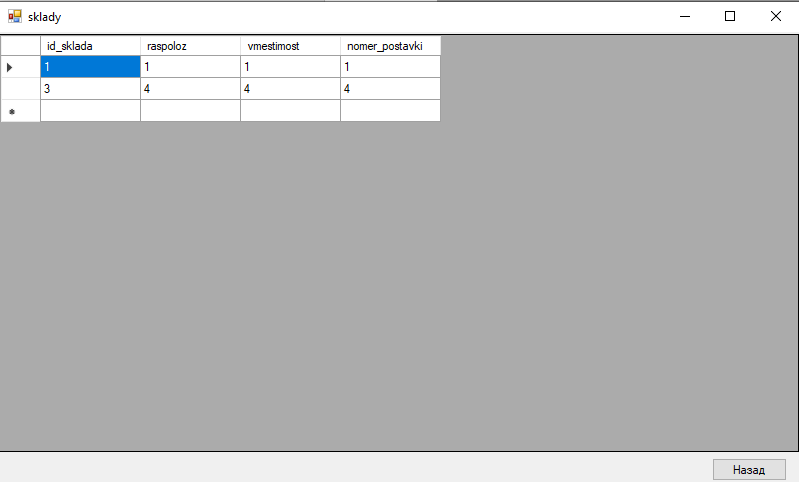


Рисунок Б.5 – Форма таблицы «инвентарь»

На рисунке Б.6 представлена форма таблицы «отчёты».

 Рисунок Б.6 – Форма таблицы «отчёты»

На рисунке Б.7 представлена форма таблицы «склады».

 Рисунок Б.7 – Форма таблицы «склады»

На рисунке Б.8 представлена форма главного администратора.

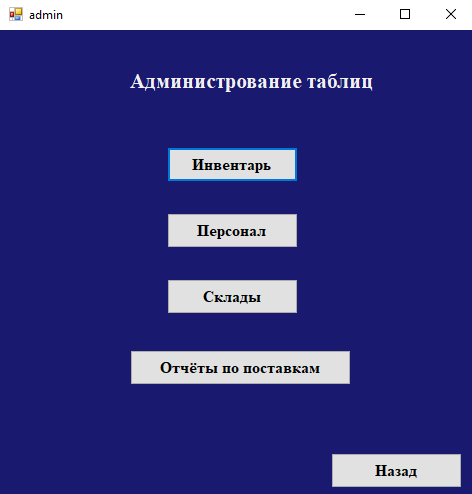


Рисунок Б.8 – Форма главного администратора

На рисунке Б.9 представлена форма таблицы «инвентарь».

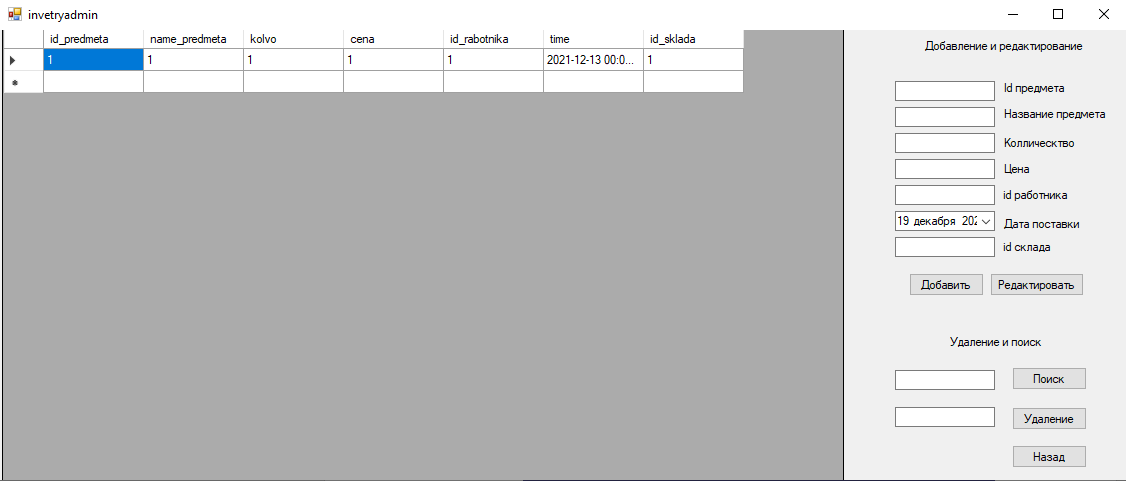


Рисунок Б.9 – Форма таблицы «инвентарь»

На рисунке Б.10 представлена форма таблицы «персонал».

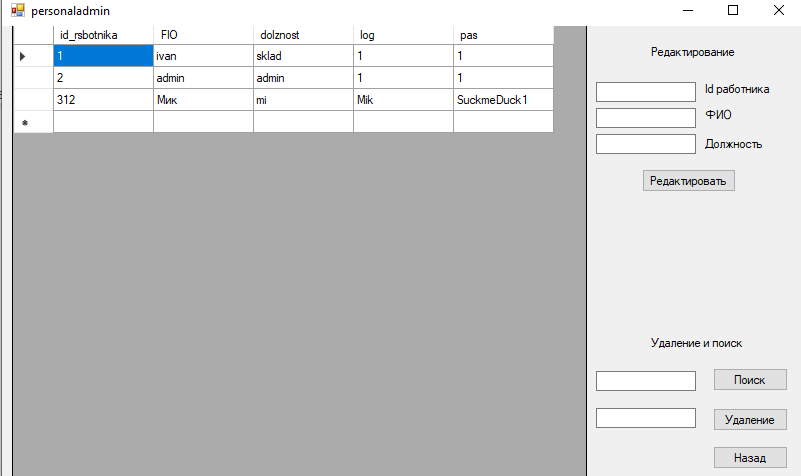


Рисунок Б.10 – Форма таблицы «персонал»

На рисунке Б.11 представлена форма таблицы «склады».

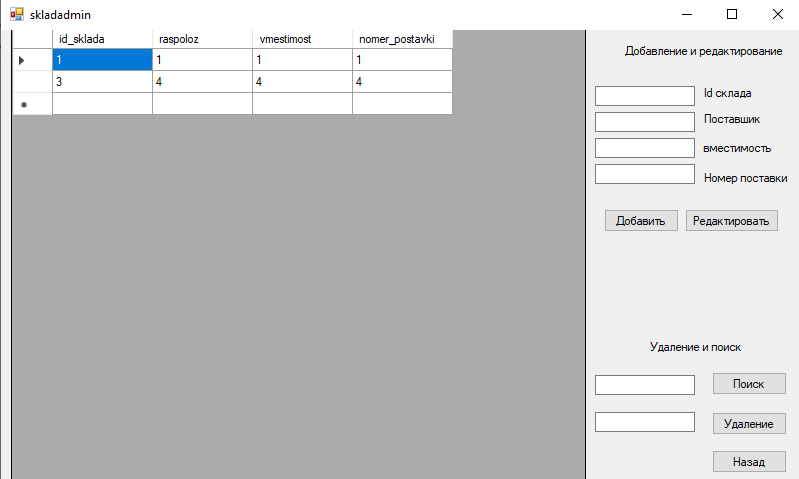


Рисунок Б.11 – Форма таблицы «склады»

На рисунке Б.12 представлена форма таблицы «отчёты по поставкам».

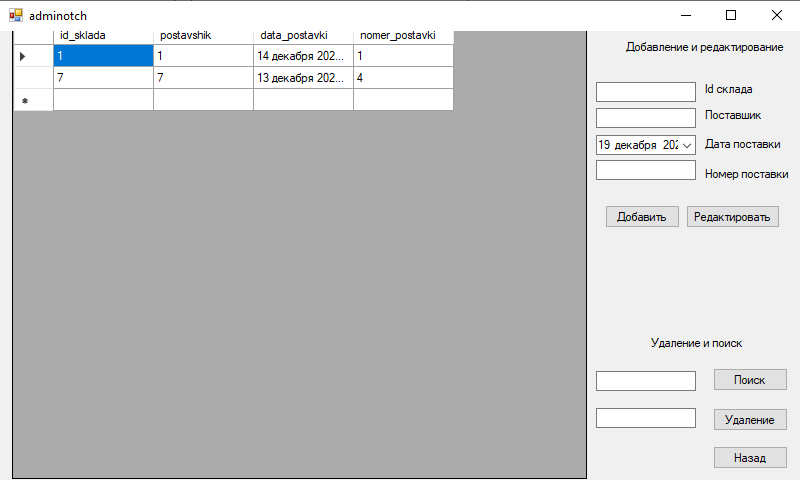


Рисунок Б.12 – Форма таблицы «отчёты по поставкам»