АННОТАЦИЯ

Курсовой проект на тему: АРМ менеджера по продажам программного обеспечения, состоит из программного средства, пояснительной записки и графической части.

Программное средство предназначено для ведения учета, находящихся на складе, программных продуктов.

Программное средство разработано в среде программирования Microsoft Visual Studio 2022 с использованием языка программирования C#.

Пояснительная записка выполнена на 49 листах, содержит 4 раздела и одно приложение.

Графическая часть выполнена на двух листах, включает диаграмму вариантов использования и диаграмму последовательности.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc106117569)

[1. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ 6](#_Toc106117570)

[2. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 7](#_Toc106117571)

[2.1. Постановка задачи 7](#_Toc106117572)

[2.2. Описание организации данных 7](#_Toc106117573)

[2.3. Описание программных средств 8](#_Toc106117574)

[2.4. Проектирование интерфейса 8](#_Toc106117575)

[2.5 Описание логической структуры программы 11](#_Toc106117576)

[3. ТЕСТИРОВАНИЕ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ 13](#_Toc106117577)

[4. РУКОВОДСТВО ПО ИСПОЛЬЗОВАНИЮ ПРОГРАММНОГО СРЕДСТВА 25](#_Toc106117578)

[ЗАКЛЮЧЕНИЕ 27](#_Toc106117579)

[ИСТОЧНИКИ, ИСПОЛЬЗУЕМЫЕ ПРИ РАЗРАБОТКЕ 28](#_Toc106117580)

[ПРИЛОЖЕНИЕ 1 (ОБЯЗАТЕЛЬНОЕ) – ЛИСТИНГ ПРОГРАММЫ 29](#_Toc106117581)

ВВЕДЕНИЕ

На сегодняшний день применение баз данных приобрело важное значение для многих организаций, которые для упрощения своей работы применяют компьютерные технологии.

База данных – это совокупность сведений, относящихся к определенной теме или задаче, организованная таким образом, чтобы обеспечить удобное представление этой совокупности, как целой, так и любой ее части.

Базы данных стали основой информационных систем и в корне изменили методы работы многих организаций. В частности, в последние годы развитие технологии баз данных привело к созданию весьма мощных и удобных в эксплуатации систем. Благодаря этому системы баз данных стали доступными широкому кругу пользователей

В данном курсовом проекте необходимо разработать автоматизированное рабочее место для менеджера по продажам программного обеспечения. Разрабатываемое приложение предназначено для облегчения работы менеджеров по продажам, связанной с ведением баз данных товаров.

Пояснительная записка содержит следующие разделы:

1. «Назначение и область применения» описывается назначение программы, краткая характеристика области применения программы. Проводится анализ предметной области и инфологическое проектирование разрабатываемой базы данных. Приводится обзор существующих подходов к решению задачи, их достоинства и недостатки, отличие разработанной программы от аналогов. Рассматриваются существующие аналоги.
2. «Технические характеристики» – рассматриваются следующие вопросы: формирование требований к системе в целом; требования к функциям (раздел «Постановка задачи»); требования к организации данных в проекте, описание их организации, структуры, построение концептуальной и логической модели данных, способа кодирования (при необходимости) и предварительной подготовки (раздел «Описание организации данных»); обоснование выбора программного обеспечения, используемого для разработки программного средства (раздел «Описание программных средств»); проектирование программного средства. (разделы «Описание физической структуры базы данных», «Проектирование интерфейса», «Описание логической структуры программы»).
3. «Тестирование и анализ полученных результатов». Описывается проверка работы программы на различных тестах и приводится анализ ожидаемых и полученных результатов, выводы по результатам тестирования.
4. «Руководство по использованию программного средства» – описывается последовательность действий пользователя, обеспечивающих загрузку, запуск, выполнение и завершение программы, приведены описание функций, формата и возможных вариантов команд, с помощью которых пользователь управляет выполнением программы, а также ответы программы на эти команды.

1. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ

Целью курсового проекта является разработка автоматизированного рабочего места менеджера по продажам программного обеспечения.

База данных может быть использована в различных сферах жизни. Например: использоваться продавцами программного обеспечения для более быстрой и точной работы с клиентами.

Аналогом данного проекта является сайт quintadb.ru. Достоинством данного сайта является упрощенный конструктор баз данных. Недостатками сайта являются: всплывающие рекламные блоки, ограничение по объему информации при работе на бесплатном тарифе, для использования всех возможностей сайта требуется покупка определенного тарифа.

Отличиями разрабатываемой программы от аналогов являются: офлайн-доступ к информации по запросу пользователя, доступность функций без регистрации и СМС.

Данное программное средство рассчитано на людей, работающих в сфере компьютерных услуг, которые занимаются формированием списков программных средств и продуктов.

Требования к аппаратным и операционным ресурсам для разработки:

1) Операционная система Windows 7 и новее.

2) Процессор Intel Celeron и мощнее.

4) Оперативная память 2 GB и более.

5) HDD 5GB и более.

2. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

2.1. Постановка задачи

Разработать автоматизированное рабочее место менеджера по продажам программного обеспечения, предназначенное для ведения учета программных средств и продуктов, находящихся на складе компании.

В приложении реализовать меню с пунктами:

* все продукты;
* продукты на складе.

По нажатию на пункт «все продукты» запрашивается код доступа. После ввода корректного кода доступа, открывается таблица товаров, находящихся в базе данных товаров.

Реализовать возможность добавления, редактирования параметров и удаления товаров, находящихся в базе данных товаров.

По нажатию на пункт «продукты на складе» открывается таблица товаров, имеющихся в наличии на складе.

Реализовать возможность продажи определенного количества товара, находящегося в наличии на складе.

Реализовать возможность поиска и фильтрации товаров по различным параметрам.

2.2. Описание организации данных

Входные данные:

* значения, используемые по умолчанию;
* данные, вводимые пользователем.

Выходные данные:

* таблица данных в соответствии с запросом пользователя.

2.3. Описание программных средств

В данном проекте используется язык программирования C# плюсами которого являются:

1) Подлинная объектная ориентированность.

2) Компонентно-ориентированное программирование.

3) Безопасный (по сравнению с языками C и C++) код.

4) Унифицированная система типизации.

5) Поддержка событийно-ориентированного программирования.

6) Более простая обработка ошибок за счет исключений.

7) Возможность создания обобщенных алгоритмов для разных типов данных, их специализация, и вычисления на этапе компиляции, используя шаблоны.

К минусам C# можно отнести:

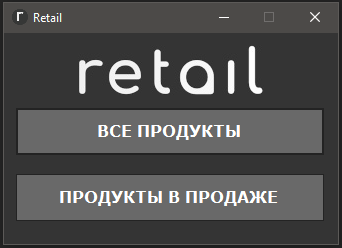
1) Приоритетная ориентированность на платформу Windows.

2) Язык бесплатен только для небольших фирм, индивидуальных программистов, стартапов и учащихся. Крупной компании покупка лицензионной версии этого языка обойдётся в круглую сумму.

2.4. Проектирование интерфейса

Проект состоит из трех форм.

На форме главного меню (рис. 2.1) пользователь может открыть таблицу со всеми продуктами или только с продуктами в наличии.



All Button

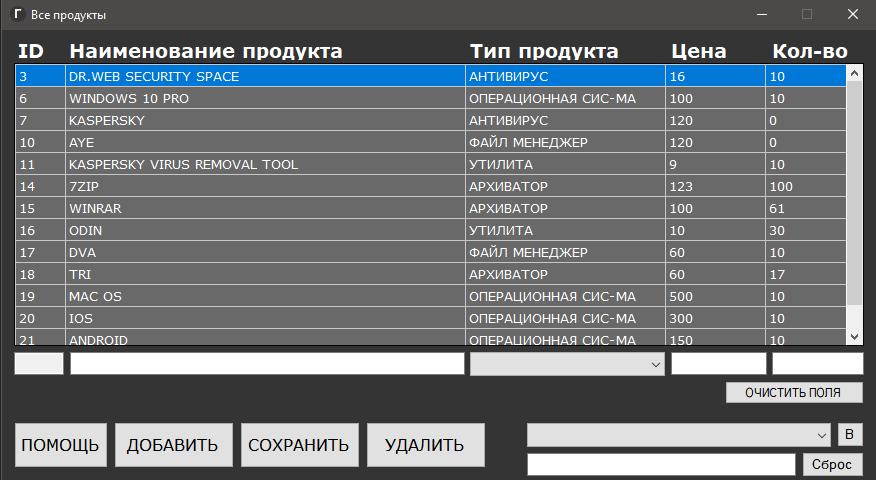
StockButton

Рис 2.1. Форма меню

Таблица 2.1 – Элементы управления сцены меню

|  |  |  |
| --- | --- | --- |
| Элементы управления окна регистрации | Свойства | Назначение |
| AllButton | Name = AllButton | Кнопка для перехода на форму со всеми товарами |
| StockButton | Name = StockButton | Кнопка перехода на форму с товарами на складе |

На форме со всеми продуктами (рис 2.2), пользователь может добавлять, редактировать, удалять продукты, выполнять фильтрацию и поиск по параметрам.



DeleteButton

SaveButton



HelpButton

AddButton



Reset

Button



OrderButton



ClearButton

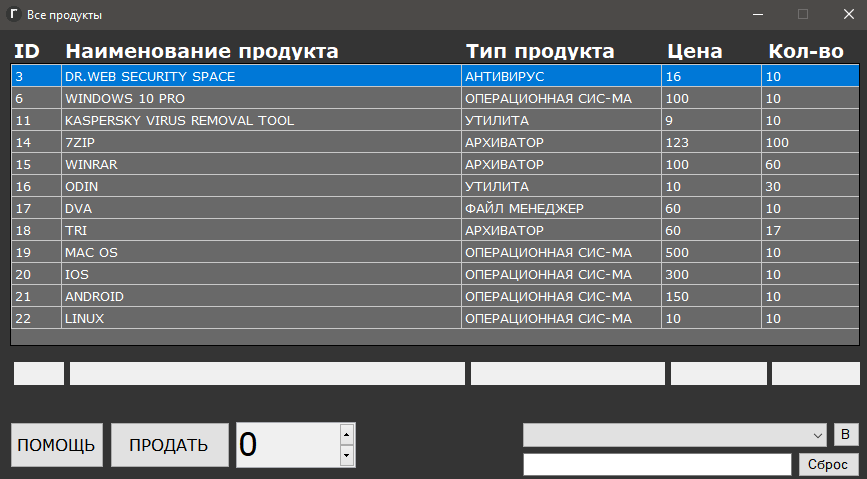


Рис 2.2. Форма всех продуктов

Таблица 2.2 –Элементы управления формы всех продуктов

|  |  |  |
| --- | --- | --- |
| Элементы управления окна навигации | Свойства | Назначение |
| HelpButton | Name = HelpButton | Открывает помощь |
| AddButton | Name = AddButton | Добавляет в таблицу запись |
| SaveButton | Name = SaveButton | Обновляет запись в таблице |
| DeleteButton | Name = DeleteButton | Удаляет запись из таблицы |
| ClearButton | Name = ClearButton | Очищает поля ввода |
| OrderButton | Name = OrderButton | Устанавливает порядок сортировки |
| ResetButton | Name = ResetButton | Сбрасывает параметры сортировки |

На форме с продуктами в продаже (рис 2.3), пользователь может производить продажи, выполнять фильтрацию и поиск по параметрам.



HelpButton

SellButton



OrderButton



Reset

Button



Рис 2.3. Форма продуктов в продаже

Таблица 2.3 –Элементы управления формы продуктов в продаже

|  |  |  |
| --- | --- | --- |
| Элементы управления окна навигации | Свойства | Назначение |
| HelpButton | Name = HelpButton | Открывает помощь |
| SellButton | Name = SellButton | Производит продажу товара из таблицы |
| OrderButton | Name = OrderButton | Устанавливает порядок сортировки |
| ResetButton | Name = ResetButton | Сбрасывает параметры сортировки |

2.5 Описание логической структуры программы

В автоматизированном рабочем месте менеджера по продажам программного обеспечения реализованы следующие функции:

– loadData –загружает информацию из базы данных в таблицу;

– authCheck – авторизация пользователя для доступа к таблице;

– helpClick – вывод помощи по форме;

– sortChanged – сортирует данные в таблице по заданному параметру;

– sellClick – производит продажу товара со склада;

– orderClick – изменяет порядок сортировки;

– clearFields – очищает поля для ввода;

– resetSort – сбрасывает сортировку;

– searchChanged – производит выборку из таблицы по названию товара;

– digitsOnly – разрешает ввод только цифр;

– addClick – добавляет информацию в таблицу;

– updateClick – обновляет информацию в таблице;

– deleteClick – удаляет информацию из таблицы;

– applicationLoad – производит загрузку основной части приложения.

3. ТЕСТИРОВАНИЕ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ

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

Метод «черного ящика». Основан на принципе «вход-выход». Программе подаются некоторые данные на вход и проверяются результаты, в надежде найти несоответствия. При этом как именно работает программа, считается несущественным.

Метод «белого ящика». В этом методе тестовые данные получают путем анализа логики программы.

Этапы процесса тестирования:

1) Проверка в нормальных условиях. Предполагает тестирование на основе данных, которые характерны для реальных условий функционирования системы.

2) Проверка в экстремальных условиях. Тестовые данные включают граничные значения области изменения входных переменных, граничные объемы данных.

3) Проверка в исключительных ситуациях. Проводится с использованием данных, значения которых лежат за пределами допустимой области. Программа должна отвергать любые данные, которые она не в состоянии обработать правильно. Наихудшая ситуация возникает тогда, когда программа принимает данные, выходящие за допустимые пределы, и выдает неверный результат.

В проекте использовался метод тестирования «черного ящика». Программа была протестирована на следующих тестах:

1) Тест №1. Проверка работы добавления.

2) Тест №2. Проверка работы обновления.

3) Тест №3. Проверка работы удаления.

4) Тест №4. Проверка работы сортировки.

5) Тест №5. Проверка работы поиска.

6) Тест №6. Проверка работы очистки полей.

7) Тест №7. Проверка работы сброса сортировки.

8) Тест №8. Проверка загрузки основной части приложения.

9) Тест №9. Проверка переноса информации в поля ввода.

10) Тест №10. Проверка работы продажи.

Тест №1. После нажатия на кнопку «Добавить», информация из полей вносится в таблицу базы данных. Этот тест представлен на рисунках 3.1 и 3.2. После сообщения об успешности операции, происходит обновление информации в таблице. В результате тестирования программа работает правильно.

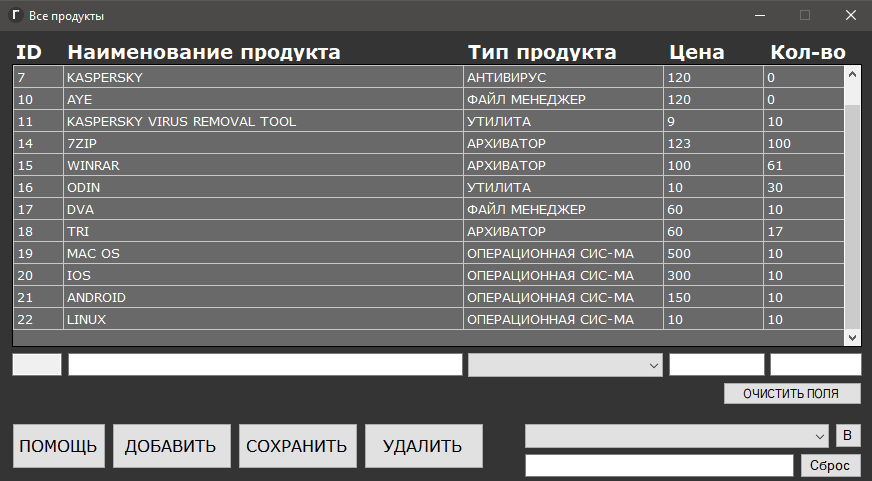


Рис. 3.1. Форма всех товаров до добавления

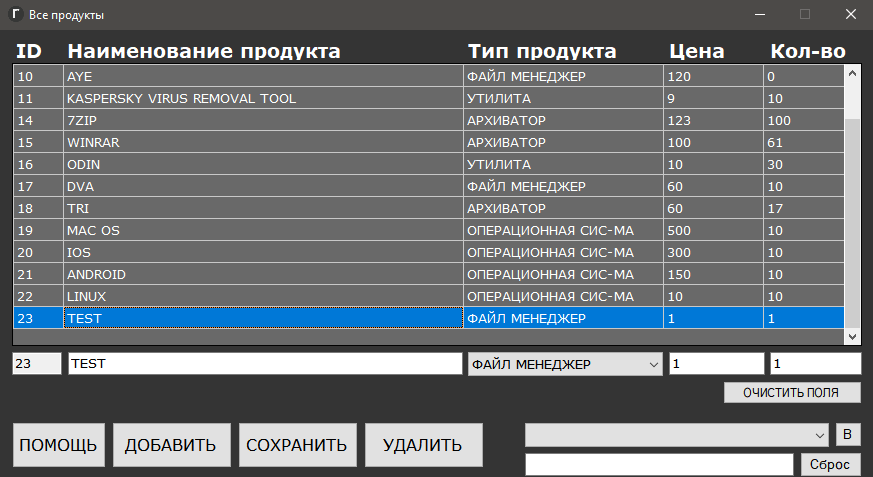


Рис. 3.2. Форма всех товаров после добавления

Тест №2. После нажатия на кнопку «Сохранить», информация из полей вносится в таблицу базы данных. Этот тест представлен на рисунках 3.3 и 3.4. После сообщения об успешности операции, происходит обновление информации в таблице. В результате тестирования программа работает правильно.

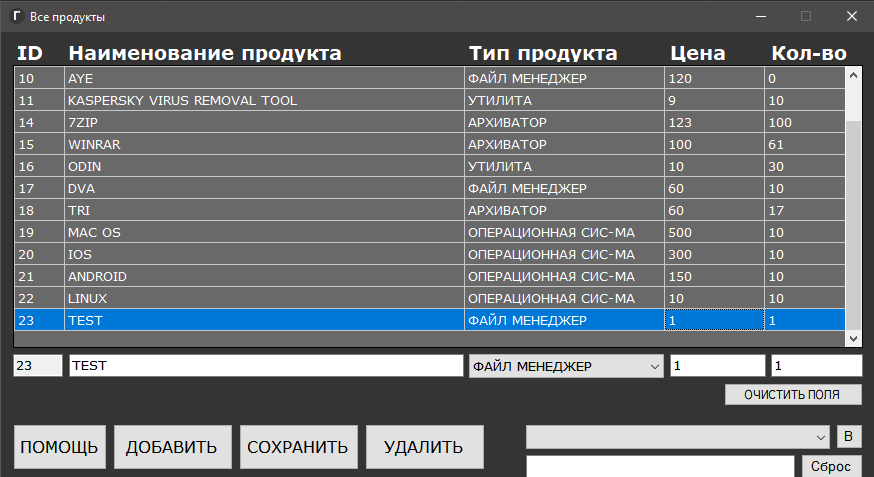


Рис 3.3. Форма всех товаров до обновления

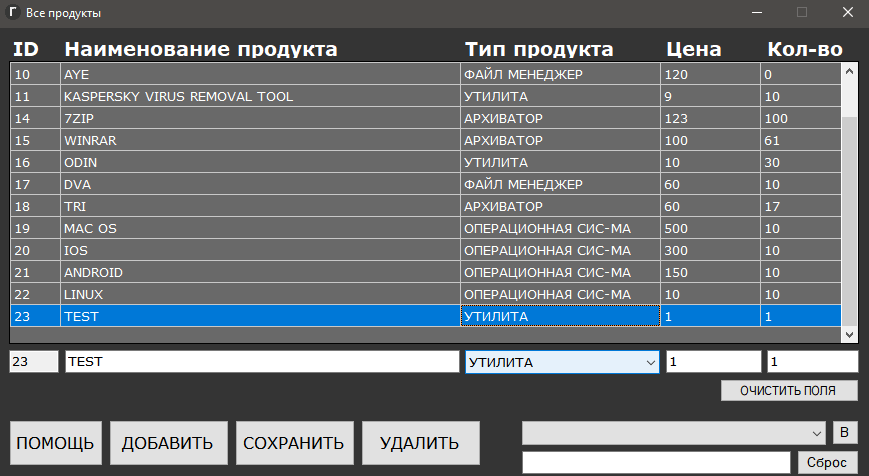


Рис 3.4. Форма всех товаров после обновления

Тест №3. После нажатия на кнопку «Удалить», происходит удаление из таблицы базы данных по параметрам из полей ввода. Этот тест представлен на рисунках 3.5 и 3.6. После сообщения об успешности операции, происходит обновление информации в таблице. В результате тестирования программа работает правильно.

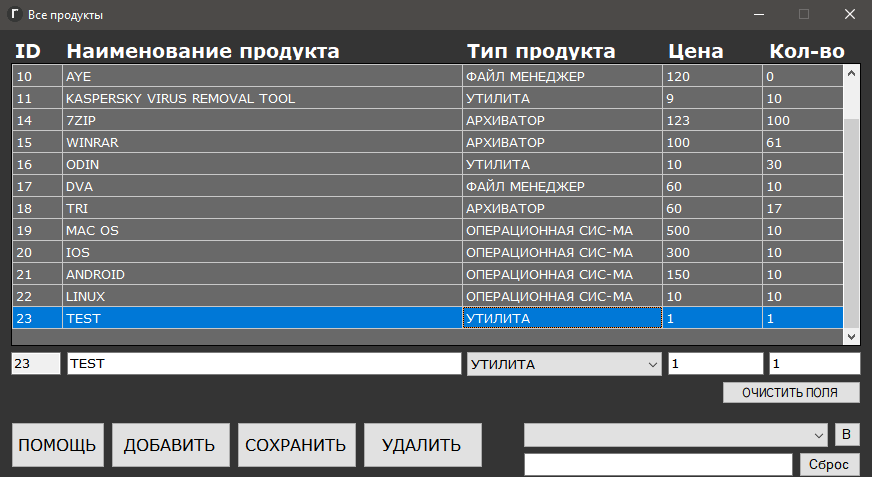


Рис 3.5. Форма всех товаров до удаления

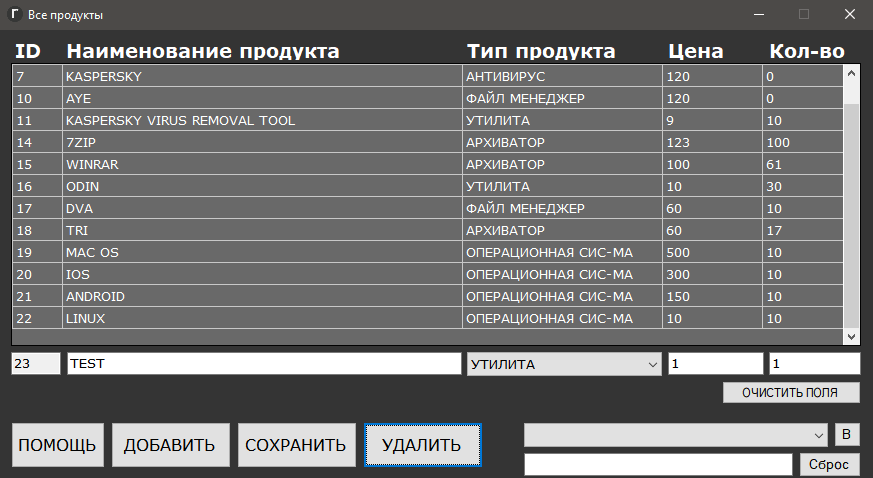


Рис 3.6. Форма всех товаров после удаления

Тест №4. После выбора вида сортировки, записи в таблице перестраиваются в соответствии с параметрами. Этот тест представлен на рисунках 3.7 и 3.8. В результате тестирования программа работает правильно.

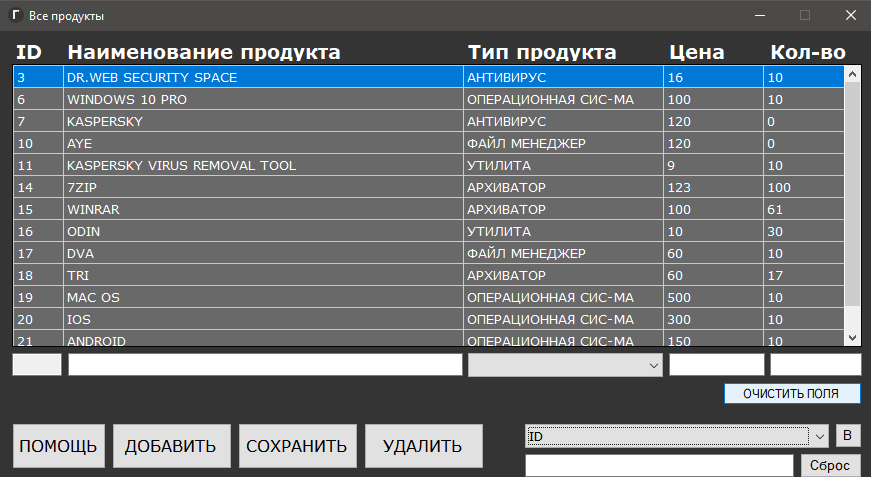


Рис 3.7. Форма всех товаров при сортировке по умолчанию

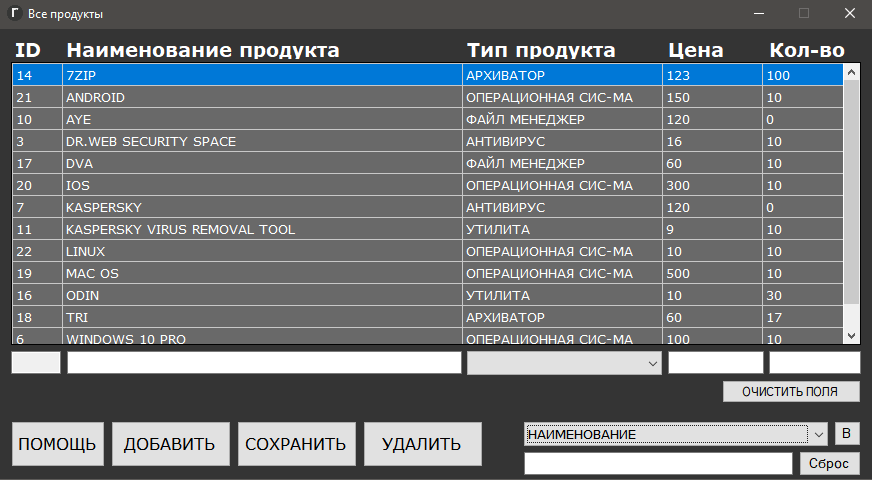


Рис 3.7. Форма всех товаров при сортировке по названию

Тест №5. После начала ввода наименования в поле поиска происходит выборка информации по неполному совпадению. Этот тест представлен на рисунках 3.9 и 3.10. В результате тестирования программа работает правильно.

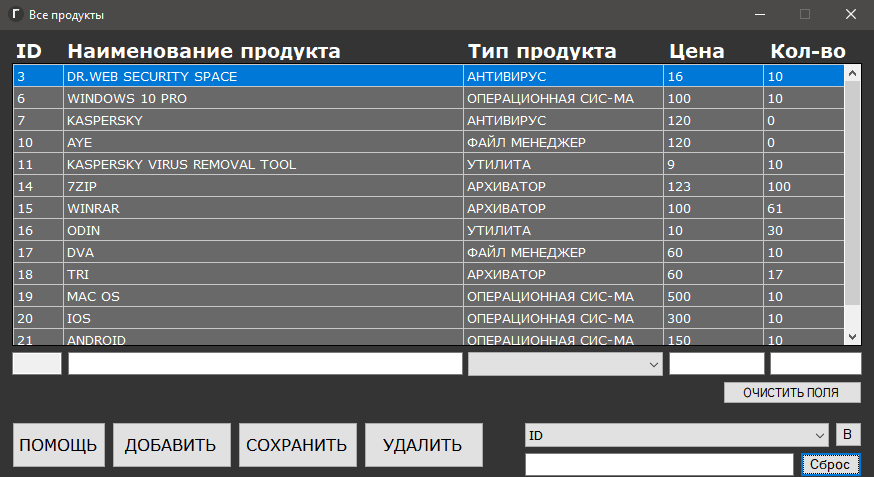


Рис 3.9. Форма всех товаров до ввода параметров поиска

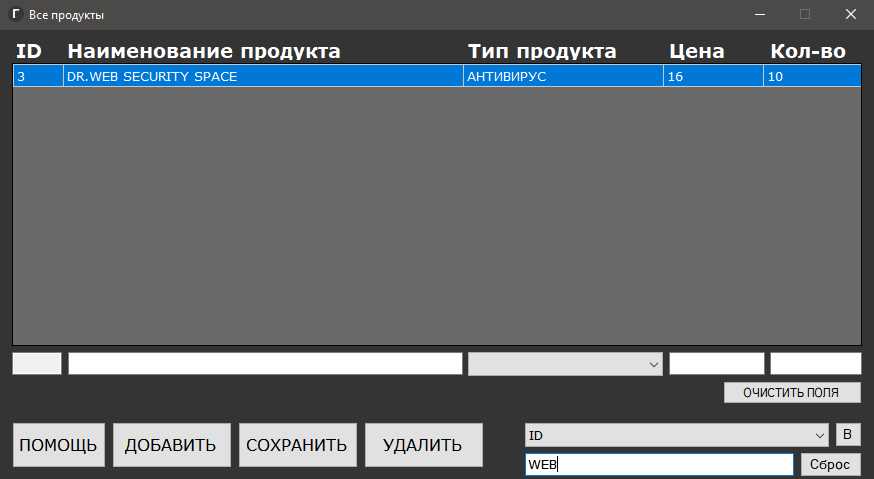


Рис 3.10. Форма всех товаров после ввода параметров поиска

Тест №6. При нажатии на кнопку «Очистить поля» происходит очистка полей ввода информации. Этот тест представлен на рисунках 3.11 и 3.12. В результате тестирования программа работает правильно

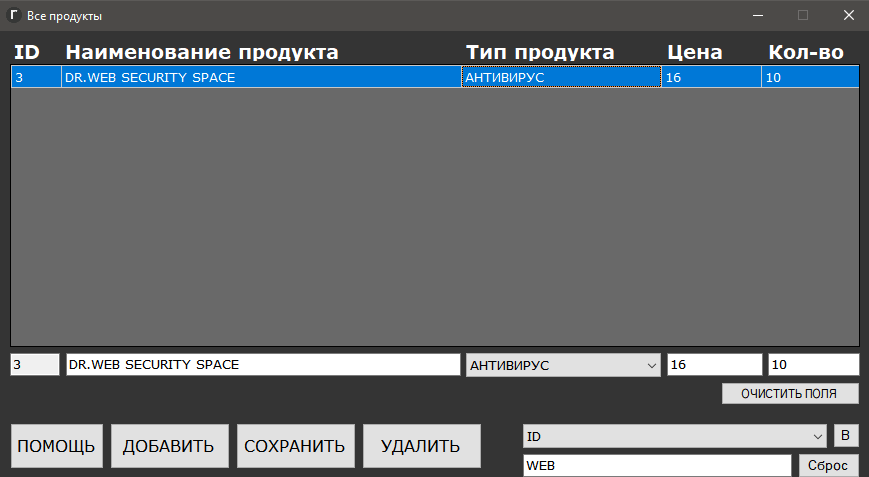


Рис 3.11. Форма всех товаров до очистки полей ввода

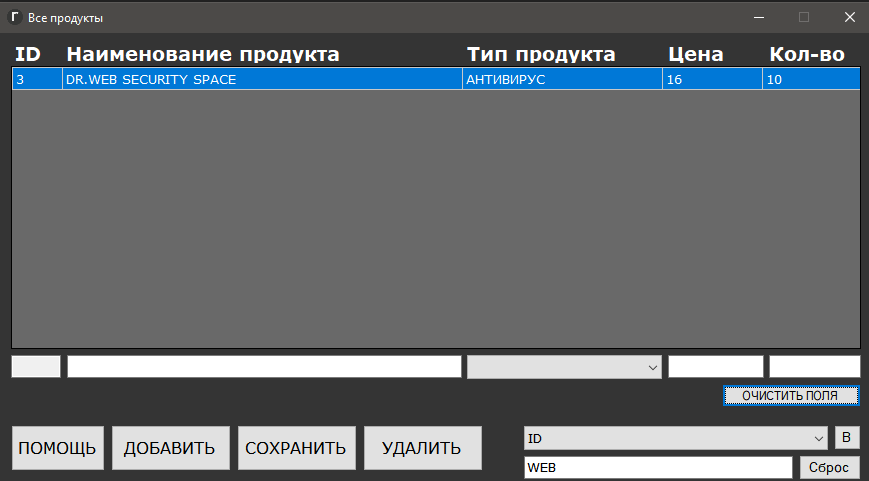


Рис 3.12. Форма всех товаров после очистки полей ввода

Тест №7. При нажатии на кнопку «Сброс» происходит сброс параметров сортировки на параметры по умолчанию. Этот тест представлен на рисунках 3.13 и 3.14. В результате тестирования программа работает правильно.

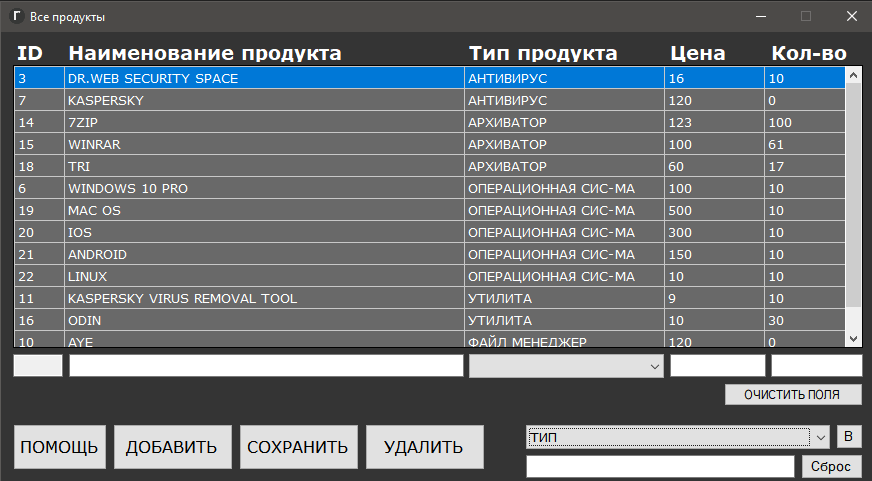


Рис 3.13. Форма всех товаров до сброса сортировки

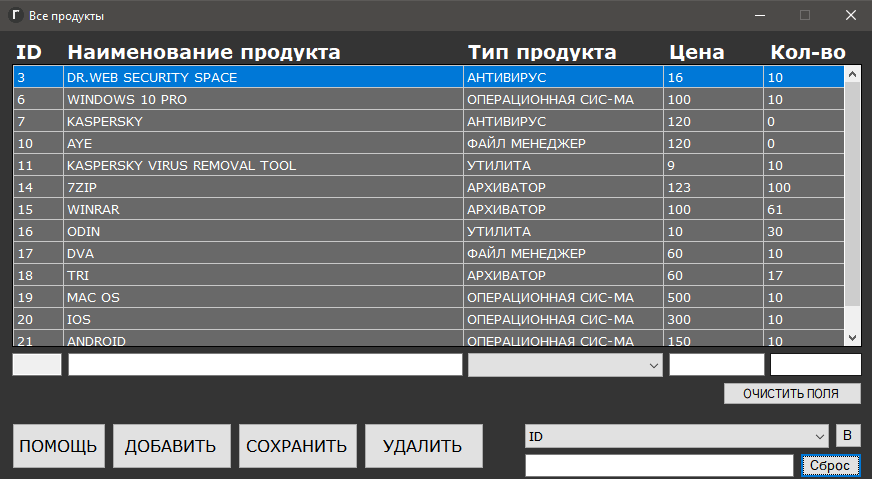


Рис 3.14. Форма всех товаров после сброса сортировки

Тест №8. После запуска приложения двойным нажатием на исполняемый файл появляется загрузочный экран и происходит загрузка основной части приложения. Этот тест представлен на рисунках 3.15 и 3.16. В результате тестирования программа работает правильно.



Рис 3.15. Загрузочный экран приложения

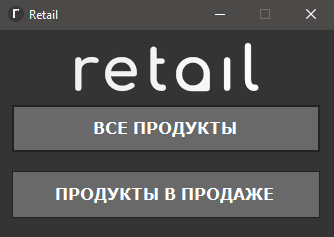


Рис 3.16. Форма главного меню основной части приложения

Тест №9. После нажатия на поле в таблице, информация перенесется в поля ввода, что представлено на рисунках 3.17 и 3.18. В результате тестирования программа работает правильно.

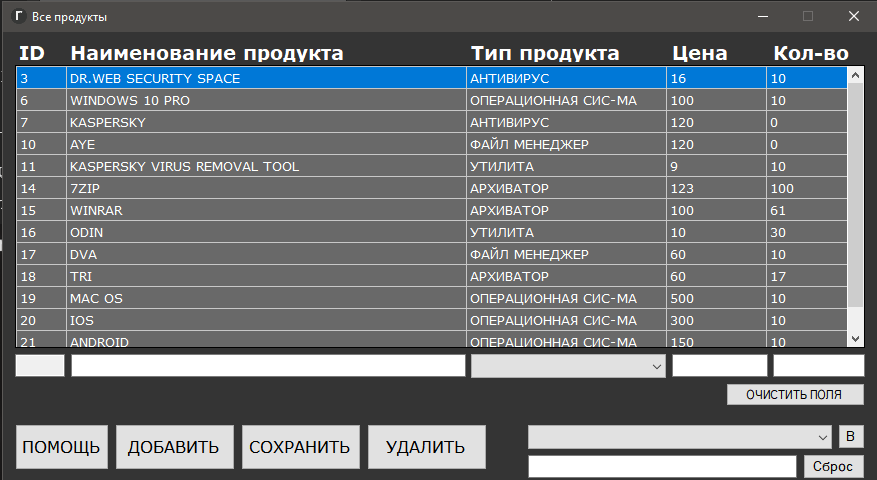


Рис 3.17. Форма всех товаров до переноса данных

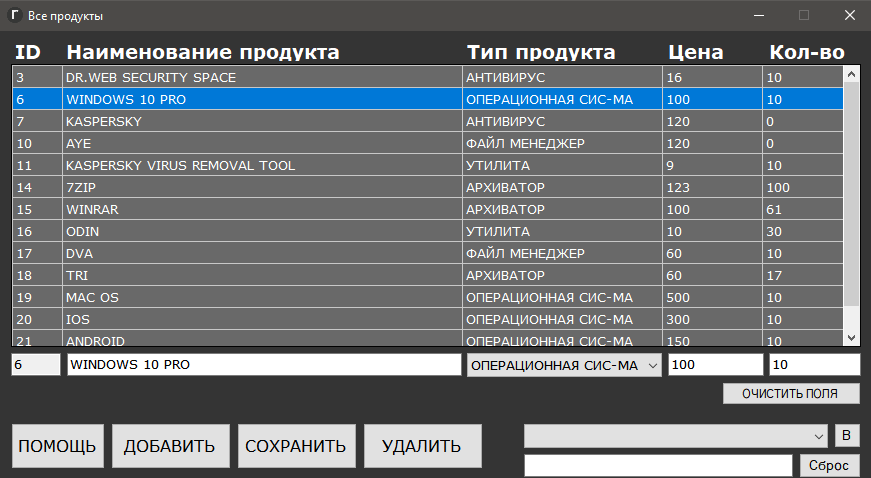


Рис 3.18. Форма всех товаров после переноса данных

Тест №10. После нажатия на кнопку «Продать», информация из поля количества отнимается из поля количества таблицы базы данных. Этот тест представлен на рисунках 3.19 и 3.20. После сообщения об успешности операции, происходит обновление информации в таблице. В результате тестирования программа работает правильно.

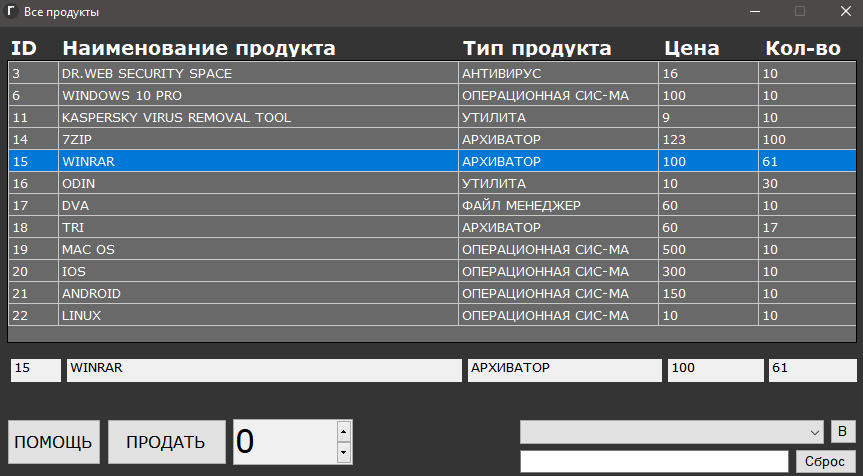


Рис 3.19. Форма товаров в продаже до продажи товара

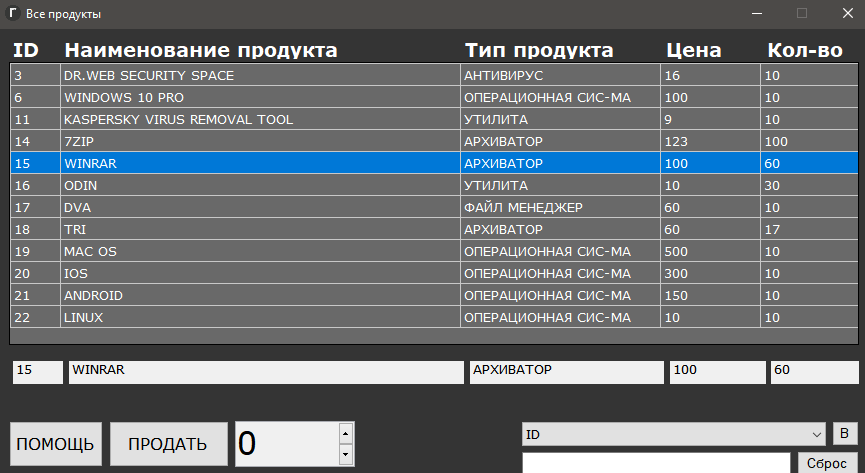


Рис 3.20. Форма товаров в продаже после продажи товара

4. РУКОВОДСТВО ПО ИСПОЛЬЗОВАНИЮ ПРОГРАММНОГО СРЕДСТВА

Чтобы запустить игру надо запустить файл «coursach.exe». Чтобы произвести продажу необходимо нажать «Продукты в продаже», представлено на рисунке 4.1.

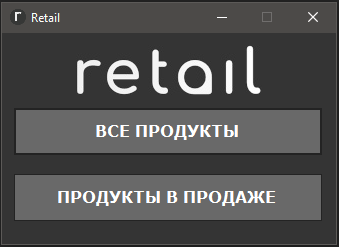


Рис 4.1. Главное меню

После открывается форма продуктов в продаже на которой можно произвести продажи программных продуктов, что представлено на рисунке 4.2.

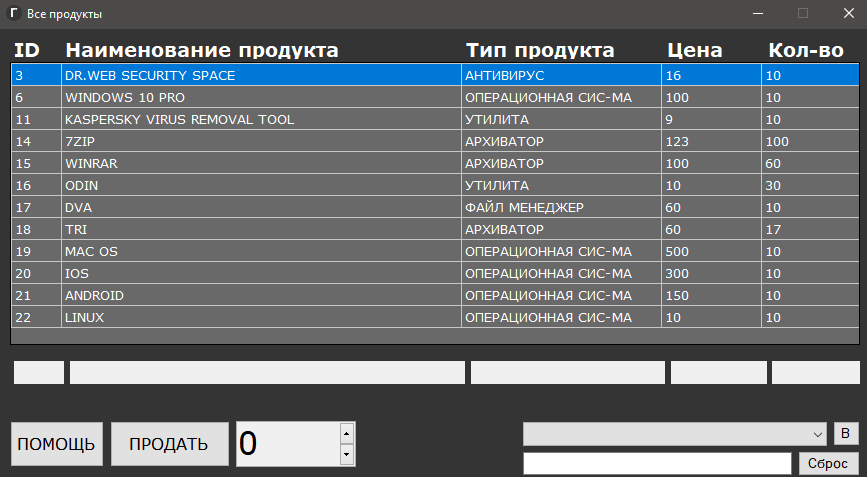


Рис 4.2. Форма продуктов в продаже

С формы главного меню также можно перейти на форму всех продуктов. Переход на форму всех продуктов с формы главного меню происходит через ввод пароля, что представлено на рисунке 4.3.

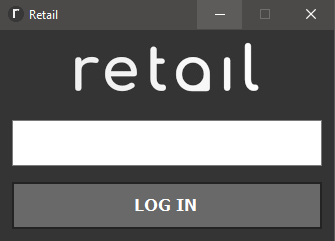


Рис 4.3. Диалоговое окно для ввода пароля для доступа к форме всех продуктов

После ввода верного пароля происходит закрытие диалогового окна и открытие формы всех продуктов, что представлено на рисунке 4.4.

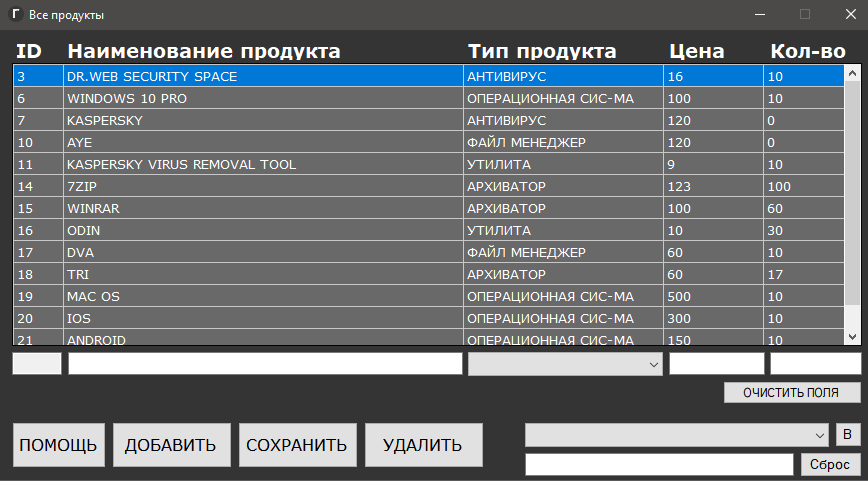


Рис 4.4. Форма всех продуктов

ЗАКЛЮЧЕНИЕ

В ходе данного курсового проекта было разработано автоматизированное рабочее место менеджера по продажам программного обеспечения. Суть приложения заключается в облегчении работы менеджеров по продажам программного обеспечения и автоматизации некоторых процессов, касающихся работ по ведению учета товара.

Были разработаны 3 формы: «Меню», «Все товары», «Товары на складе».

В качестве языка программирования используется C#.

В качестве выбранной среды разработки используется Microsoft Visual Studio 2022.

Программные средства были протестированы. Тестирование прошло успешно. В дальнейшем программу можно модифицировать, расширением функционала программы. Можно добавить смену языка, добавить новый способ авторизации для доступа к редактированию базы товаров.

ИСТОЧНИКИ, ИСПОЛЬЗУЕМЫЕ ПРИ РАЗРАБОТКЕ

1. Habr [Электронный ресурс] / Памятка/шпаргалка по SQL. Режим доступа: https://habr.com/ru/post/564390/ – Дата доступа 20.05.2022.
2. CyberForum [Электронный ресурс] / [Форум программистов и сисадминов Киберфорум](https://www.cyberforum.ru/) / [Форум программистов](https://www.cyberforum.ru/programming/) /  Программирование пользовательских интерфейсов с использованием Windows Forms / C# Windows Forms. Режим доступа: https://www.cyberforum.ru/ – Дата доступа 15.05.2022.

ПРИЛОЖЕНИЕ 1 (ОБЯЗАТЕЛЬНОЕ) – ЛИСТИНГ ПРОГРАММЫ

namespace coursach

{

public partial class All : Form

{

public All()

{

InitializeComponent();

}

private void All\_FormClosed(object sender, FormClosedEventArgs e)

{

MainMenu mainMenu = new MainMenu();

mainMenu.Show();

}

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

{

LoadData();

}

private void LoadData()

{

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT \* FROM retail", db.getConnection());

adapter.SelectCommand = command;

adapter.Fill(table);

dataGridView1.DataSource = table;

dataGridView1.Columns[0].Width = 50;

dataGridView1.Columns[1].Width = 400;

dataGridView1.Columns[2].Width = 200;

dataGridView1.Columns[3].Width = 100;

dataGridView1.Columns[4].Width = 100;

}

private void dataGridView1\_CellMouseClick(object sender, DataGridViewCellMouseEventArgs e)

{

int row = Convert.ToInt32(dataGridView1.CurrentCell.RowIndex.ToString());

textBox1.Text = dataGridView1[0, row].Value.ToString();

textBox2.Text = dataGridView1[1, row].Value.ToString();

textBox4.Text = dataGridView1[3, row].Value.ToString();

textBox5.Text = dataGridView1[4, row].Value.ToString();

comboBox1.Text = dataGridView1[2, row].Value.ToString();

}

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

{

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("UPDATE `retail` SET `name` = @Name, `type` = @Type, `price` = @Price, `quantity` = @Quantity WHERE `id` = @ID", db.getConnection());

if (textBox2.Text == "" || comboBox1.SelectedIndex==-1 || textBox4.Text == "" || textBox5.Text == "")

{

MessageBox.Show(

"Заполните все поля",

"Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Error

);

}

else

{

command.Parameters.Add("@ID", MySqlDbType.Int32).Value = textBox1.Text;

command.Parameters.AddWithValue("@Name", textBox2.Text.Trim());

command.Parameters.AddWithValue("@Type", comboBox1.Text.Trim());

command.Parameters.AddWithValue("@Price", textBox4.Text.Trim());

command.Parameters.AddWithValue("@Quantity", textBox5.Text.Trim());

db.openConnection();

adapter.SelectCommand = command;

adapter.Fill(table);

if (command.ExecuteNonQuery() == 1)

{

MessageBox.Show(

"Запись обновлена успешно",

"Успех",

MessageBoxButtons.OK,

MessageBoxIcon.Information

);

LoadData();

}

else

{

MessageBox.Show(

"Ошибка обновления записи",

"Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Error

);

}

}

db.closeConnection();

}

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

{

MessageBox.Show(

"ДОБАВЛЕНИЕ. Для того чтобы добавить информаицю о позиции, введите данные в соответствующие поля (КРОМЕ ПОЛЯ ID), затем нажмите кнопку ДОБАВИТЬ." +

"\n\nРЕДАКТИРОВАНИЕ. Для того чтобы обновить информацию о позиции, нажмите на нее в таблице, затем введите новые данные в соответствующие поля и нажмите кнопку СОХРАНИТЬ. " +

"\n\nУДАЛЕНИЕ. Для того чтобы удалить информацию о позиции, нажмите на нее в таблице, затем нажмите кнопку УДАЛИТЬ." +

"\n\nСОРТИРОВКА. Для того чтобы отсортировать информацию в таблице и выберите параметр для сортировки (по умолчанию ID)." +

"\n Для того чтобы изменить порядок сортировки - нажмите на кнопку рядом со списком. В - возрастание | У - убывание.",

"Помощь",

MessageBoxButtons.OK,

MessageBoxIcon.Information

);

}

private void button3\_Click(object sender, EventArgs e)

{

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("DELETE FROM retail WHERE id = @ID", db.getConnection());

if (textBox2.Text == "" || comboBox1.SelectedIndex==-1 || textBox4.Text == "" || textBox5.Text == "")

{

MessageBox.Show(

"Заполните все поля",

"Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Error

);

}

else

{

command.Parameters.Add("@ID", MySqlDbType.Int32).Value = textBox1.Text;

db.openConnection();

adapter.SelectCommand = command;

adapter.Fill(table);

MessageBox.Show

(

"Запись удалена успешно",

"Успех",

MessageBoxButtons.OK,

MessageBoxIcon.Information

);

}

LoadData();

db.closeConnection();

}

private void button4\_Click(object sender, EventArgs e)

{

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("INSERT INTO `retail` (`name`, `type`, `price`, `quantity`) VALUES (@Name, @Type, @Price, @Quantity)", db.getConnection());

if (textBox2.Text == "" || comboBox1.SelectedIndex==-1 || textBox4.Text == "" || textBox5.Text == "")

{

MessageBox.Show(

"Заполните все поля",

"Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Error

);

}

else

{

command.Parameters.AddWithValue("@Name", textBox2.Text.Trim());

command.Parameters.AddWithValue("@Type", comboBox1.Text.Trim());

command.Parameters.AddWithValue("@Price", textBox4.Text.Trim());

command.Parameters.AddWithValue("@Quantity", textBox5.Text.Trim());

db.openConnection();

adapter.SelectCommand = command;

adapter.Fill(table);

MessageBox.Show(

"Запись добавлена успешно",

"Успех",

MessageBoxButtons.OK,

MessageBoxIcon.Information

);

}

LoadData();

db.closeConnection();

}

private void button5\_Click(object sender, EventArgs e)

{

textBox1.Clear();

textBox2.Clear();

textBox4.Clear();

textBox5.Clear();

comboBox1.SelectedIndex = -1;

}

private void textBox5\_KeyPress(object sender, KeyPressEventArgs e)

{

//ввод только цифр

if ((e.KeyChar <= 47 || e.KeyChar >= 58) && e.KeyChar != 8)

{

e.Handled = true;

}

}

private void textBox4\_KeyPress(object sender, KeyPressEventArgs e)

{

//ввод только цифр

if ((e.KeyChar <= 47 || e.KeyChar >= 58) && e.KeyChar != 8)

{

e.Handled = true;

}

}

private void comboBox2\_SelectedIndexChanged(object sender, EventArgs e)

{

if(button6.Text.Trim() == "В")

dataGridView1.Sort(dataGridView1.Columns[comboBox2.SelectedIndex], ListSortDirection.Ascending);

if (button6.Text.Trim() == "У")

dataGridView1.Sort(dataGridView1.Columns[comboBox2.SelectedIndex], ListSortDirection.Descending);

}

private void button6\_Click(object sender, EventArgs e)

{

if (comboBox2.SelectedIndex == -1)

{

MessageBox.Show(

"Выберите колонку для сортировки",

"Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Error

);

}

else

{

if (button6.Text.Trim() == "В")

{

button6.Text = "У";

dataGridView1.Sort(dataGridView1.Columns[comboBox2.SelectedIndex], ListSortDirection.Descending);

}

else

{

button6.Text = "В";

dataGridView1.Sort(dataGridView1.Columns[comboBox2.SelectedIndex], ListSortDirection.Ascending);

}

}

}

private void button7\_Click(object sender, EventArgs e)

{

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT \* FROM retail WHERE `name` LIKE '%@Name%' ", db.getConnection());

command.CommandText = "SELECT\* FROM retail WHERE `name` LIKE '%" + textBox3.Text.Trim() + "%';";

//command.Parameters.AddWithValue("@Name", textBox3.Text.Trim());

db.openConnection();

adapter.SelectCommand = command;

adapter.Fill(table);

dataGridView1.DataSource = table;

db.closeConnection();

}

private void button8\_Click(object sender, EventArgs e)

{

textBox3.Clear();

comboBox2.SelectedIndex = 0;

LoadData();

}

private void textBox3\_TextChanged(object sender, EventArgs e)

{

if (textBox3.Text.Trim() != "")

{

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT \* FROM retail WHERE `name` LIKE '%@Name%' ", db.getConnection());

command.CommandText = "SELECT\* FROM retail WHERE `name` LIKE '%" + textBox3.Text.Trim() + "%';";

//command.Parameters.AddWithValue("@Name", textBox3.Text.Trim());

db.openConnection();

adapter.SelectCommand = command;

adapter.Fill(table);

dataGridView1.DataSource = table;

db.closeConnection();

}

else

{

LoadData();

}

}

}

}

public partial class Stock : Form

{

public Stock()

{

InitializeComponent();

}

private void Stock\_FormClosed(object sender, FormClosedEventArgs e)

{

MainMenu mainMenu = new MainMenu();

mainMenu.Show();

}

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

{

LoadData();

}

private void LoadData()

{

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT \* FROM retail where quantity>0", db.getConnection());

adapter.SelectCommand = command;

adapter.Fill(table);

dataGridView1.DataSource = table;

dataGridView1.Columns[0].Width = 50;

dataGridView1.Columns[1].Width = 400;

dataGridView1.Columns[2].Width = 200;

dataGridView1.Columns[3].Width = 100;

dataGridView1.Columns[4].Width = 100;

}

private void dataGridView1\_CellMouseClick(object sender, DataGridViewCellMouseEventArgs e)

{

int row = Convert.ToInt32(dataGridView1.CurrentCell.RowIndex.ToString());

label6.Text = dataGridView1[0, row].Value.ToString();

label7.Text = dataGridView1[1, row].Value.ToString();

label9.Text = dataGridView1[3, row].Value.ToString();

label10.Text = dataGridView1[4, row].Value.ToString();

label8.Text = dataGridView1[2, row].Value.ToString();

}

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

{

}

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

{

MessageBox.Show(

"ПРОДАЖА. Для того чтобы выполнить продажу, кликните на нужный товар в таблице, затем введите количество товара и нажмите кнопку ПРОДАТЬ." +

"\n\nСОРТИРОВКА. Для того чтобы отсортировать информацию в таблице и выберите параметр для сортировки (по умолчанию ID)." +

"\n Для того чтобы изменить порядок сортировки - нажмите на кнопку рядом со списком. В - возрастание | У - убывание.",

"Помощь",

MessageBoxButtons.OK,

MessageBoxIcon.Information

);

}

private void button4\_Click(object sender, EventArgs e)

{

if (label7.Text == "" || label8.Text == "" || label9.Text == "" || label10.Text == "")

{

MessageBox.Show(

"Выберите товар для продажи",

"Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Error

);

}

else

{

if ((Convert.ToInt32(label10.Text) - Convert.ToInt32(numericUpDown1.Value) < 0))

{

MessageBox.Show(

"Недостаточно товара для продажи введенного количества",

"Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Error

);

}

else

{

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("UPDATE `retail` SET `quantity` = @Quantity WHERE `id` = @ID", db.getConnection());

command.Parameters.Add("@ID", MySqlDbType.Int32).Value = label6.Text;

command.Parameters.AddWithValue("@Quantity", Convert.ToInt32(label10.Text) - Convert.ToInt32(numericUpDown1.Value));

db.openConnection();

adapter.SelectCommand = command;

adapter.Fill(table);

if (command.ExecuteNonQuery() == 1)

{

MessageBox.Show(

"Сумма: " + numericUpDown1.Value \* Convert.ToInt32(label9.Text) + ".",

"Подытог",

MessageBoxButtons.OK,

MessageBoxIcon.Information

);

MessageBox.Show(

"Успешно продано.",

"Успех",

MessageBoxButtons.OK,

MessageBoxIcon.Information

);

numericUpDown1.Value = 0;

button8.PerformClick();

LoadData();

}

else

{

MessageBox.Show(

"Ошибка продажи",

"Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Error

);

}

LoadData();

db.closeConnection();

}

}

}

private void comboBox2\_SelectedIndexChanged(object sender, EventArgs e)

{

if (button6.Text.Trim() == "В")

dataGridView1.Sort(dataGridView1.Columns[comboBox2.SelectedIndex], ListSortDirection.Ascending);

if (button6.Text.Trim() == "У")

dataGridView1.Sort(dataGridView1.Columns[comboBox2.SelectedIndex], ListSortDirection.Descending);

}

private void button6\_Click(object sender, EventArgs e)

{

if (comboBox2.SelectedIndex == -1)

{

MessageBox.Show(

"Выберите колонку для сортировки",

"Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Error

);

}

else

{

if (button6.Text.Trim() == "В")

{

button6.Text = "У";

dataGridView1.Sort(dataGridView1.Columns[comboBox2.SelectedIndex], ListSortDirection.Descending);

}

else

{

button6.Text = "В";

dataGridView1.Sort(dataGridView1.Columns[comboBox2.SelectedIndex], ListSortDirection.Ascending);

}

}

}

private void button8\_Click(object sender, EventArgs e)

{

label6.Text = "";

label7.Text = "";

label8.Text = "";

label9.Text = "";

label10.Text = "";

textBox3.Clear();

comboBox2.SelectedIndex = 0;

LoadData();

}

private void textBox3\_TextChanged(object sender, EventArgs e)

{

if (textBox3.Text.Trim() != "")

{

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT \* FROM retail WHERE `name` LIKE '%@Name%' ", db.getConnection());

command.CommandText = "SELECT\* FROM retail WHERE `name` LIKE '%" + textBox3.Text.Trim() + "%';";

//command.Parameters.AddWithValue("@Name", textBox3.Text.Trim());

db.openConnection();

adapter.SelectCommand = command;

adapter.Fill(table);

dataGridView1.DataSource = table;

db.closeConnection();

}

else

{

LoadData();

}

}

private void textBox5\_KeyPress(object sender, KeyPressEventArgs e)

{

if ((e.KeyChar <= 47 || e.KeyChar >= 58) && e.KeyChar != 8)

{

e.Handled = true;

}

}

}

public partial class Auth : Form

{

public Auth()

{

InitializeComponent();

}

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

{

if(textBox1.Text.Trim() == "admin")

{

this.Hide();

All all = new All();

all.Show();

}

else

{

MessageBox.Show(

"Неправильный пароль! " +

"\nАварийное завершение работы.",

"Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Error

);

Application.Exit();

}

}

private void Auth\_FormClosed(object sender, FormClosedEventArgs e)

{

this.Hide();

MainMenu mm = new MainMenu();

mm.Show();

}

}