**Министерство образования Московской области**

**ГОУ ВО МО «Государственный гуманитарно-технологический университет» (ГГТУ) Ликино-Дулевский политехнический колледж- филиал ГГТУ**

Наименование колледжа/техникума

**О Т Ч Ё Т**

**ПО КУРСОВОЙ РАБОТЕ**

по МДК 01.01 Курсовой проект

(код и наименование ПМ)

Обучающегося\_\_\_\_\_ Гришина Романа Алексеевича \_

Курс \_\_\_\_3\_\_\_\_\_ группа \_\_\_ИСП.19А\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Специальность 09.02.07 Информационные системы и программирование \_\_\_

Место курсовой Ликино-Дулевский политехнический колледж – филиал ГГТУ

Руководители курсовой

от колледжа Кузьмина Елена Евгеньевна\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

г. Ликино-Дулево

2022

Содержание

[Введение 3](#_Toc99921495)

[1. Назначение разработки 3](#_Toc99921496)

[2. Требования к программному модулю 5](#_Toc99921497)

[2.1 Требования к функциональным характеристикам 5](#_Toc99921498)

[2.2 Требования к составу и параметрам технических средств 6](#_Toc99921499)

[2.3 Требования к информационной и программной совместимости 7](#_Toc99921500)

[3. Руководство программиста 9](#_Toc99921501)

[3.1 Разработка базы данных 9](#_Toc99921502)

[3.2 Разработка программного модуля 17](#_Toc99921503)

[3.3 Текст программы 27](#_Toc99921504)

[4. Руководство пользователя 45](#_Toc99921505)

[4.1 Выполнение программного модуля 45](#_Toc99921506)

[4.2 Сообщения пользователю 55](#_Toc99921507)

[5. Тестирование и отладка программного модуля 56](#_Toc99921508)

[Заключение 58](#_Toc99921509)

[Список литературы 58](#_Toc99921510)

# Введение

Мебельный цех занимается производством мягкой мебели – диванов, кресел, кушеток и т.д.

У цеха имеется небольшой магазинчик, в котором работают два продавца. В магазине представлены образцы типовых моделей мебели, которую производит цех. Продавцы принимают заказы на производство мебели от покупателей. Покупатель выбирает модель, расцветку обивки и комплектацию (существуют комплекты мебели – например, диван и два кресла в едином стиле, но покупатель может заказать только диван или диван и одно кресло). Продавец оформляет договор и принимает предоплату. Срок исполнения заказа – обычно одна-две недели.

В цехе работают 5 мастеров, каждый из которых «ведет» несколько заказов. Мастер занимается всеми операциями – от распилки пиломатериалов до обтяжки мебели тканью. Часть материалов – фурнитура (ручки, гвозди, шурупы, шарниры, подвижные блоки) и матрацы - закупается на стороне.

Заказы распределяются между мастерами директором цеха. Он также занимается закупкой материалов (дерева, фанеры, ткани и фурнитуры) и координирует работу.

Когда заказ готов, продавец связывается с покупателем и договаривается о времени доставки. Затем создает гарантийный талон, документы на получение товара, прием оставшейся оплаты и передает их водителю. Водитель доставляет заказ клиенту, забирает оставшуюся часть оплаты заказа и подписывает документы у клиента. После доставки заказа водитель сдает документы и деньги бухгалтеру, который контролирует правильность расчетов и оформления.

У цеха также есть собственный дизайнер-технолог, который занимается разработкой новых моделей мебели.

# **Назначение** **разработки**

Разработка проводится на основании приказа о курсовом проектировании.

Программный продукт «Учёт продажи офисной мебели» – программа, созданная для работы с базой данных по предметной области.

Автоматизация продукта «Учёт продажи офисной мебели» - процесс внедрения программно-аппаратных комплексов для облегчения и автоматизации процесса ведения учёта по продажи, а также добавление в базу новых покупателей, новых моделей мебели, договоров, продаж и просмотр информации по ним.

Приложение должно выполнять следующие функции:

* Вход в программу при запуске
* Просмотр информации из базы данных
* Корректно выводить данные из базы данных.
* Добавление новой информации
* Редактирование информации
* Фильтрация
* Сортировка
* Переходы по страницам
* Просмотр информации из БД,

# Требования к программному модулю

## **Требования** **к функциональным характеристикам**

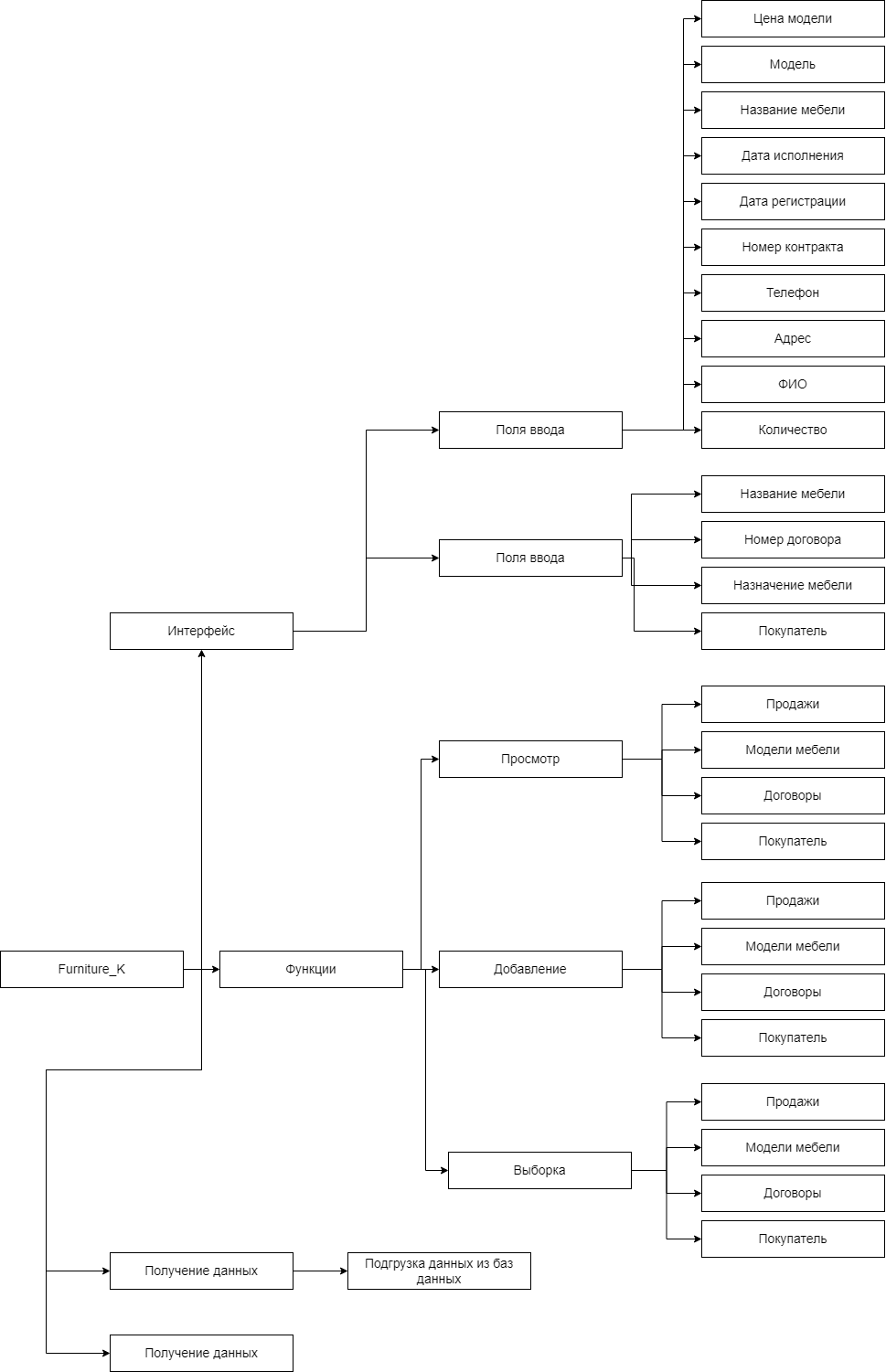


Рис.1 «Схема приложения»

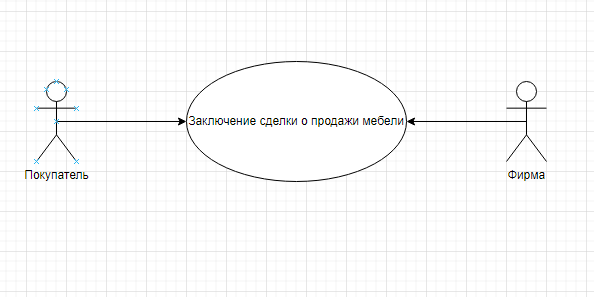


Рис.2 «Схема приложения1»

## **Требования** **к составу и параметрам технических средств**

Минимальные характеристики компьютера

|  |  |
| --- | --- |
| Процессор | AMD Athlon |
| Память | От 4 гб |
| Разрешение экрана | От 1280x1024 |
| Устройства ввода | Клавиатура, мышь |
| Дисковое пространство | От 1 Гб |
| Дополнительно | Постоянное подключение к сети, в которой находится база данных |
|  |  |

Оптимальные характеристики компьютера

|  |  |
| --- | --- |
| Процессор | Intel Core i5 3ГГц |
| Память | От 8 гб |
| Разрешение экрана | От 1920x1080 |
| Устройства ввода | Клавиатура, мышь |
| Дисковое пространство | От 1 Гб |
| Дополнительно | Постоянное подключение к сети, в которой находится база данных |
|  |  |

## **Требования** **к информационной и программной совместимости**

Для корректной работы программы необходимо: **ОС Windows 7/8/8.1/10 32х/64x**, установленный дистрибутив **.Net Framework 4.7.2.**

**Word 2016** – текстовый процессор, предназначенный для создания, просмотра, редактирования и форматирования текстов статей, деловых бумаг, а также иных документов, с локальным применением простейших форм таблично-матричных алгоритмов. Используется для ведения хода разработки и составления отчета для подведения итога работоспособности программы.

**Excel 2016** – программа для работы с электронными таблицами. Используется для импорта данных в БД с помощью мастера импорта.

**MS SQL Server Management Studio 2017** (и выше) - утилита из Microsoft SQL Server 2005 и более поздних версий для конфигурирования, управления и администрирования всех компонентов Microsoft SQL Server. Утилита включает скриптовый редактор и графическую программу, которая работает с объектами и настройками сервера.

**MS SQL Server** - система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных.

**Visual Studio 2017** (и выше) – продукт включающая интегрированную среду разработки программного обеспечения и ряд других инструментальных инструментов. Нужна для реализации основного кода программы «Учёт продажи офисной мебели» написанная на языке C#

**Язык программирования C#** - компилируемый статически типизированный язык программирования общего назначения, разработанный в 1969—1973 годах сотрудником Bell Labs Деннисом Ритчи как развитие языка Си. Первоначально был разработан для реализации операционной системы UNIX, но впоследствии был перенесён на множество других платформ

**ADO.NET** - технология, предоставляющая доступ и управление данными, хранящимся в базе данных или других источниках.

**ADO.NET Entity Framework** - объектно-ориентированная технология доступа к данным. Предоставляет возможность взаимодействия с объектами как посредством LINQ в виде LINQ to Entities, так и с использованием Entity SQL. Для облегчения построения web-решений используется как ADO.NET Data Services, так и связка из Windows Communication Foundation и Windows Presentation Foundation, позволяющая строить многоуровневые приложения.

# **Руководство** **программиста**

## **Разработка** **базы данных**

Создание БД

Нажать на «Базы данных» 🡪 «Создать базу данных»

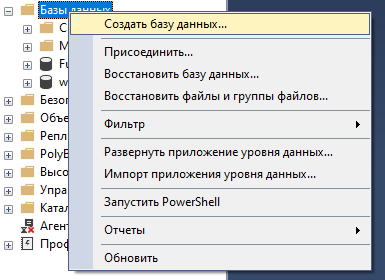


Рис. 3 «Создание базы данных»

В поле «Имя базы данных» ввести название базы, в этом случаем «Furniture» 🡪 нажать «ОК»

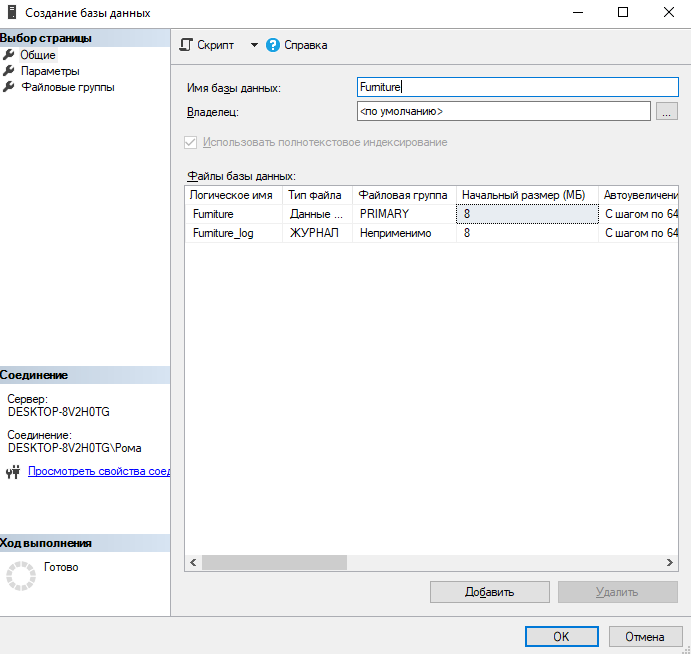


Рис.4 «Имя БД»

Создание Таблиц

Таблица Buyer

нажать ПКМ по «Таблицы» 🡪 «Создать» 🡪 «Таблица»

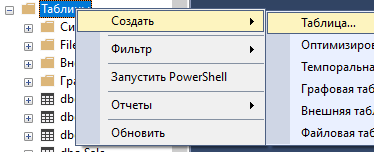


Рис5 «Создание таблиц»

Заполняем таблицу, создаем столбцы и задаем типы

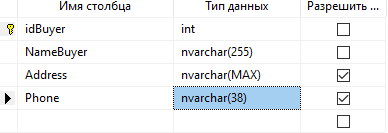


Рис.6 «Таблица Buyer»

Таблица Contract

Заполняем столбцы и типы

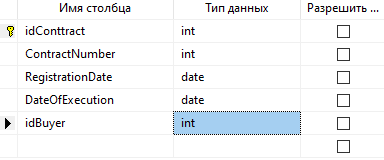


Рис.7 «Таблица Contract»

Таблица Model

Заполняем столбцы и типы

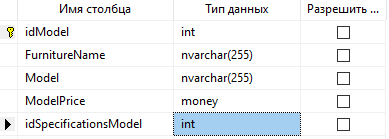


Рис.8 «Таблица Model»

Таблица Sale

Заполняем столбцы и типы

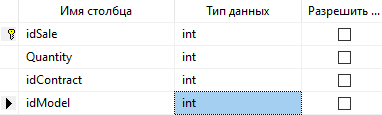


Рис.9 «Таблица Sale»

Таблица SpecificationsModel

Заполняем столбцы и типы

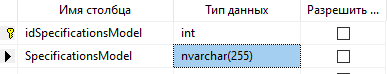


Рис.10 «Таблица SpecificationsModel»

Теперь осуществляем импорт данных из приложения MS Excel

нажать ПКМ на «Название БД» 🡪 «Задачи» 🡪 «Импорт данных»

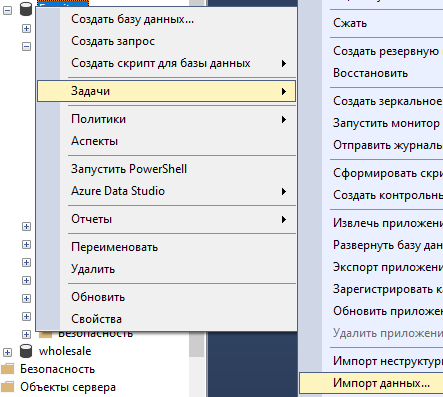


Рис.11 «Импорт данных»

В появившемся окне, в поле «Источник данных» выбрать Excel

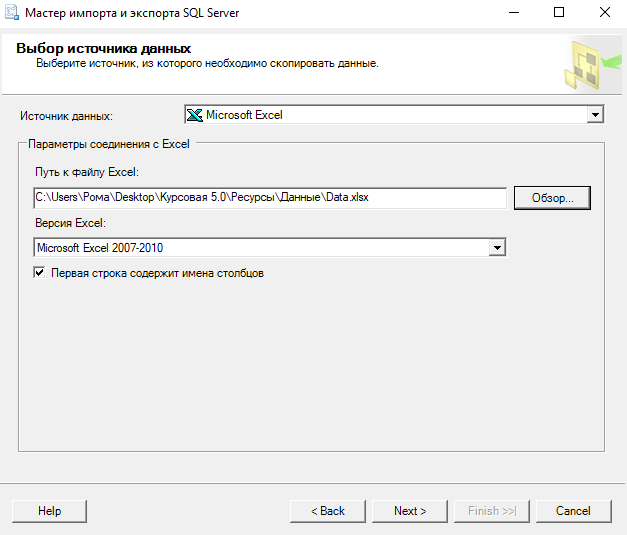


Рис.12 «Выбор откуда производиться импорт данных»

нажать кнопку «Обзор» и выбираем нужный Excel файл и нажать на кнопку «Next»

Выбрать в поле Destinatio «SQL server Native» и в поле Database выбрать нужную БД

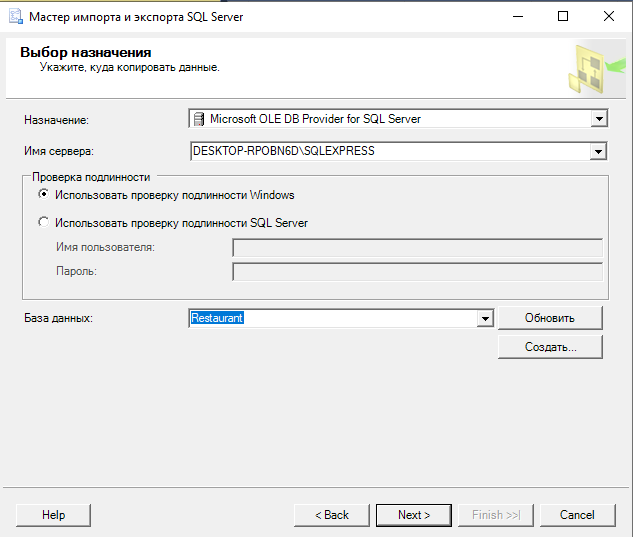


Рис.13 «Выбор куда импортировать данные»

выбрать как именно импортировать данные и нажать «Next»

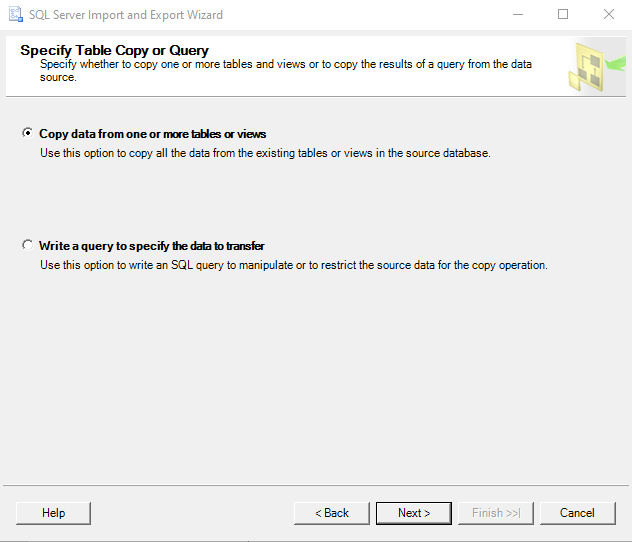


Рис.14 «выбор того, как будет импортировать данные»

выбрать нужные нам таблицы, и ждем «Next»

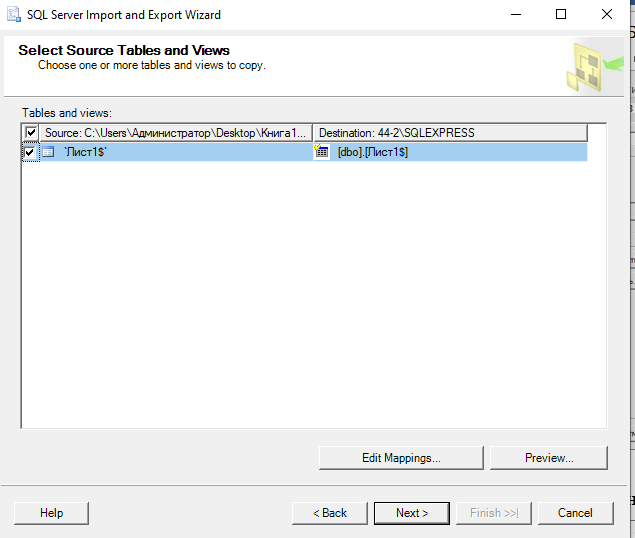


Рис.15 «Выбор таблиц»

Перенос данных произведет успешно

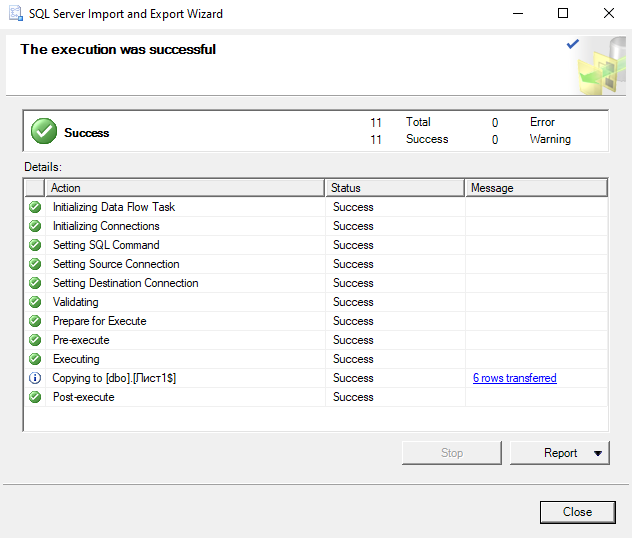


Рис.16 «Успех»

Диаграмма

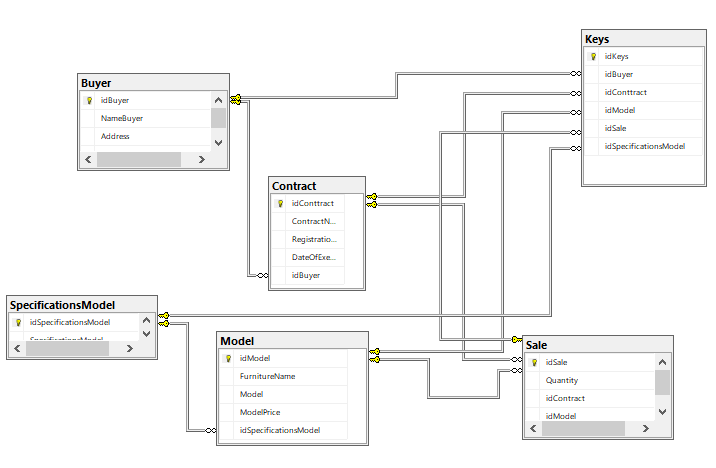


Рис.17 «Диаграмма»

## **Разработка** **программного модуля**

Программа была разработана на Visual Studio 2019 - 2022

Шаги создания программы:

1. **Создание проекта.**

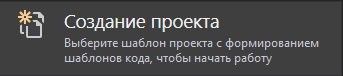


Рис.18 «Создание проекта»

1. **Создание типа проекта.**

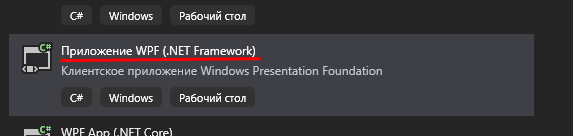


Рис.19 «Выбор типа проекта»

1. **Подключения к базе данных (ADO NET).**
   1. **Подключение ADO NET**

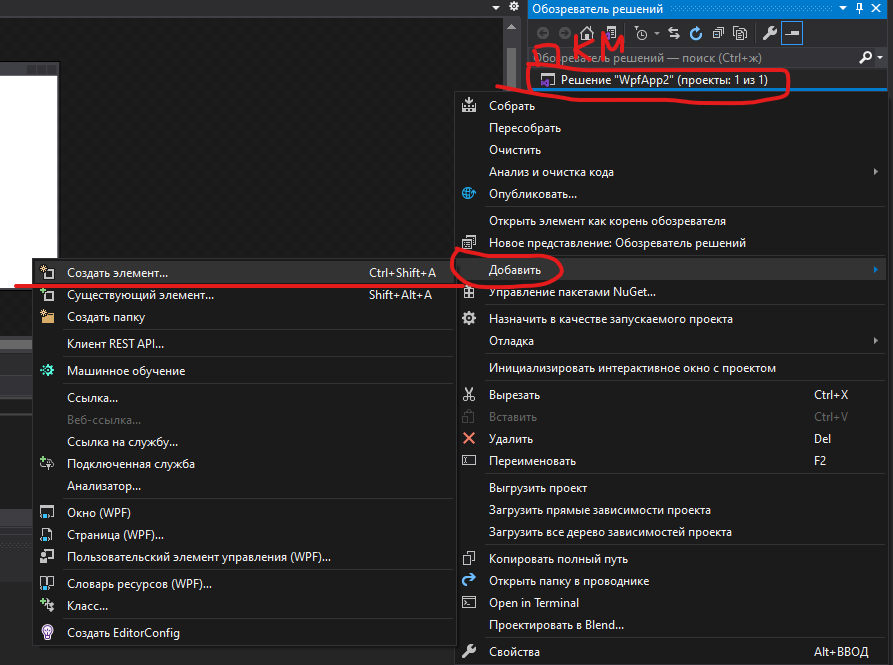


Рис.20 «Создание Элемента ADO NET»

Выбираем в левом столбце выбираем «Данные» и нажать на Модель ADO.NET

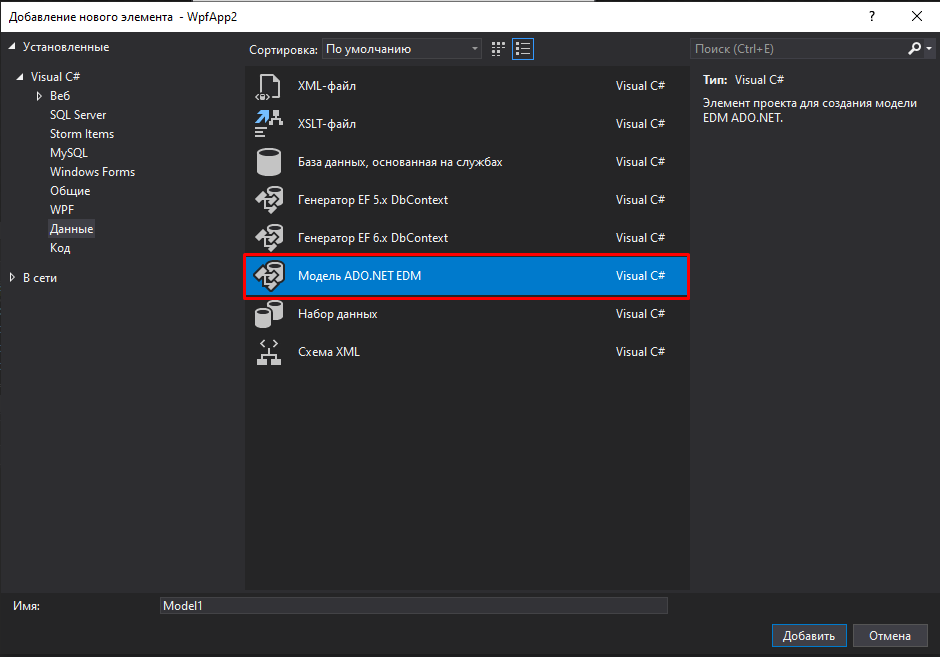


Рис.21 «Выбор ADO.NET»

* 1. **Подключение БД**

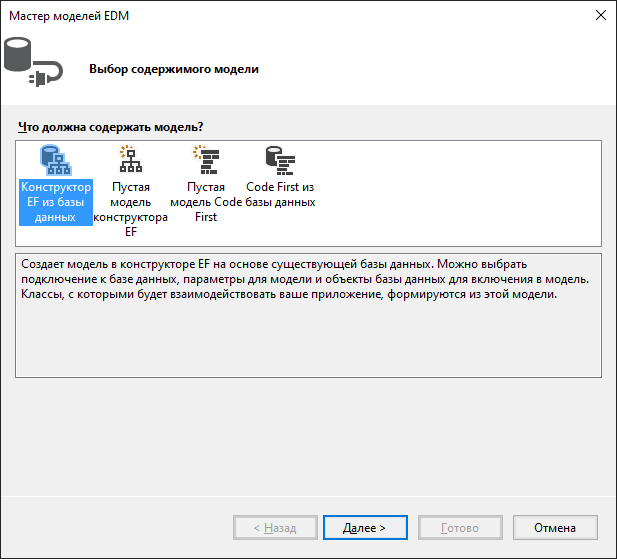


Рис.22 «Выбор конструктора»

Нажать на «Создать соединение»

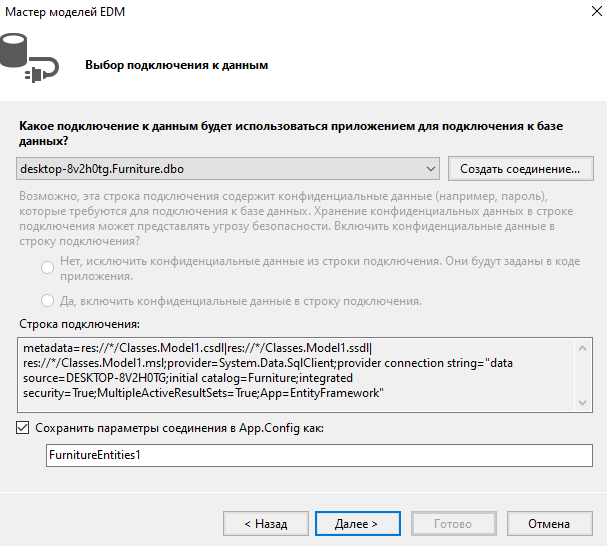


Рис.23 «Создание создании»

Вводим «Имя сервера» и выбираем нужную БД

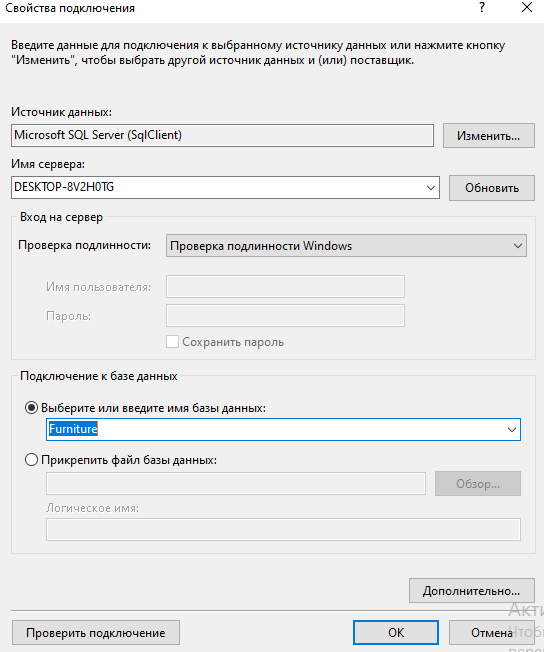


Рис.24 «Выбор БД»

Выбираем нужные таблицы и нажать «Готово»

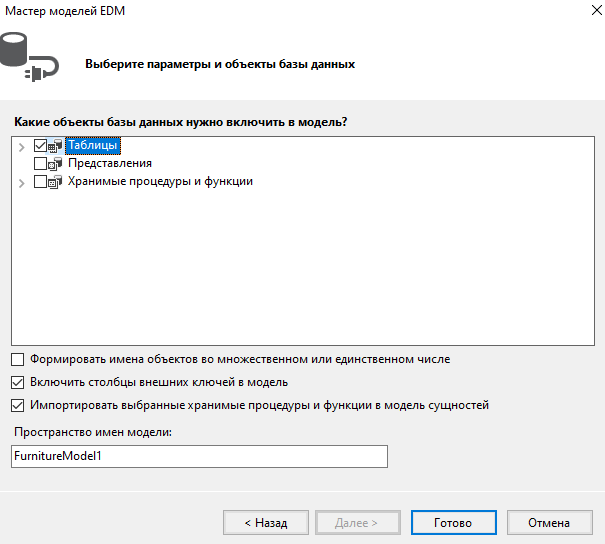


Рис.25 «Выбор таблиц»

Получившиеся модель данных

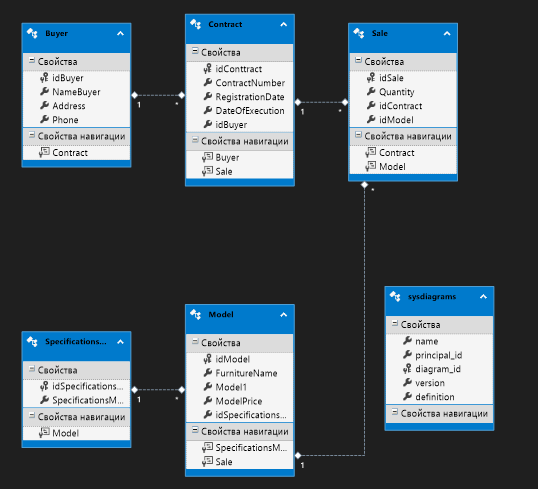


Рис.26 «Модель данных»

Модель БД в обозревателе решений

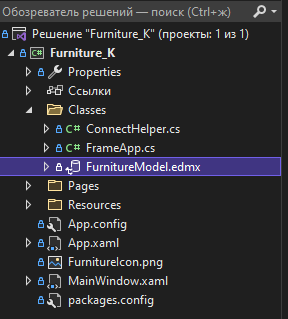


Рис.27 «Обозреватель Решения»

1. **Создание Класса.**

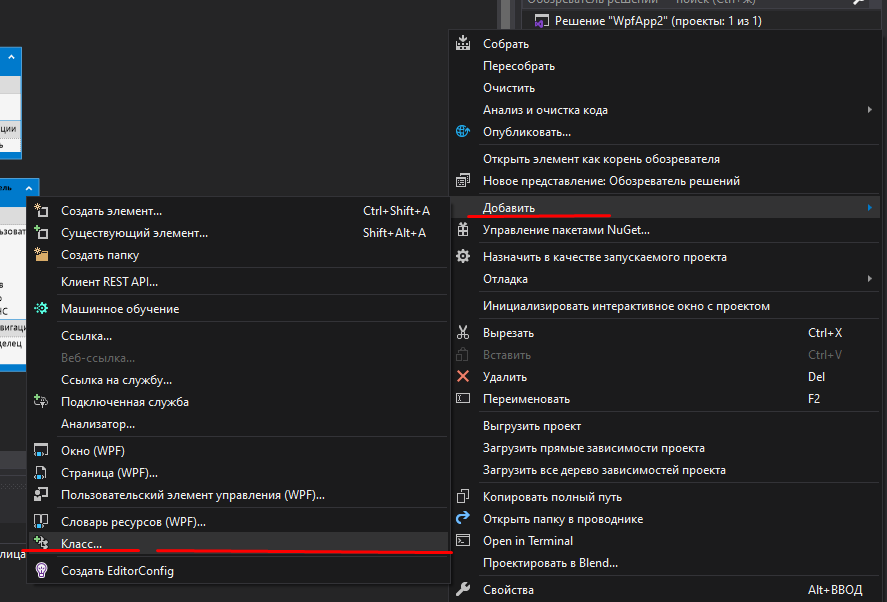


Рис.28 «Создание класса»

Класс ConnectHelper.cs - Помогает при переходе на следующие страницы

class ConnectHelper

{

public static FurnitureEntities FurnitureOBJ; // Строка подключения БД //

}

**Создание Страницы.**

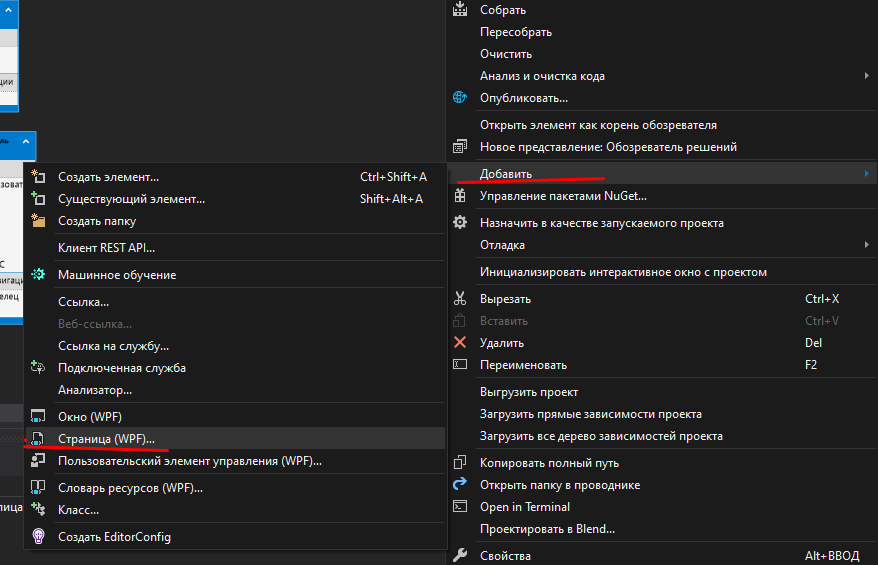


Рис.29 «Создание страницы»

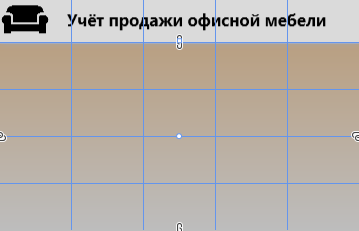
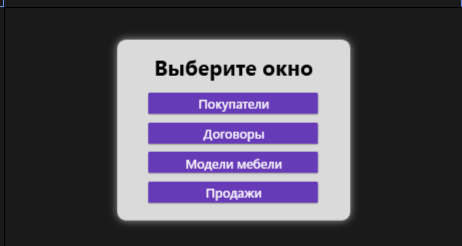


Рис.30 «Страница MainWindow»

  
Рис.31 «Страница MainPage »

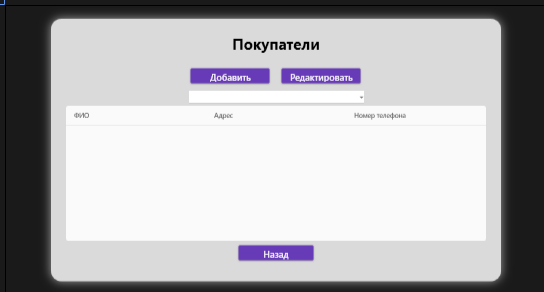


Рис.32 «Страница BuyerPage»

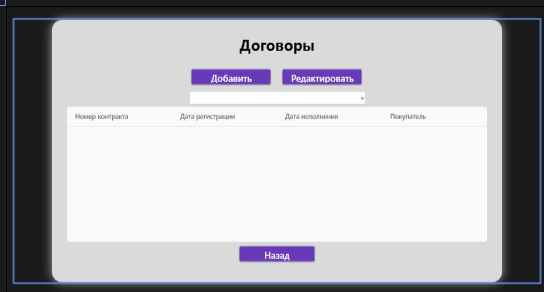


Рис.33 «Страница ContractPage»

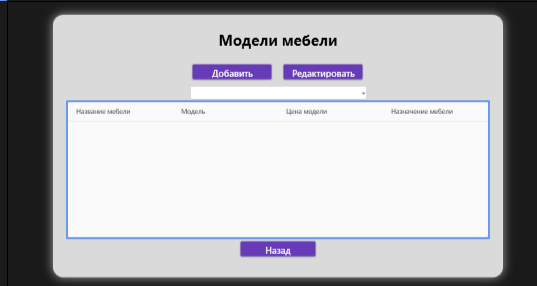


Рис.34 «Страница ModelPage»

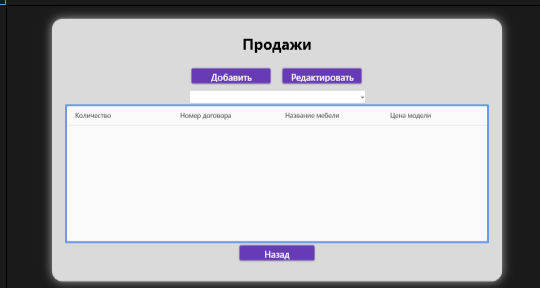


Рис.35 «Страница SalePage»

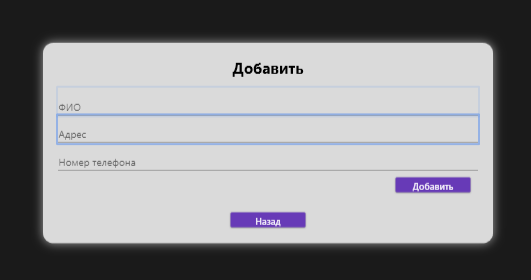


Рис.36 «Страница BuyerAddPage»

Для других страниц, перечисленных выше, сделаны похожие страницы на добавление

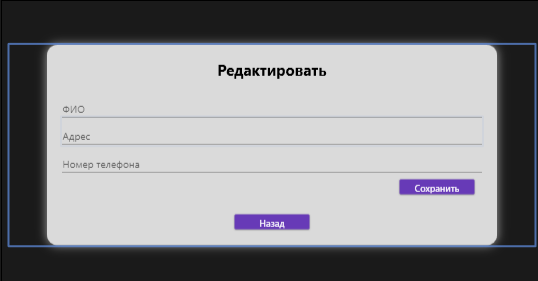


Рис.37 «Страница BuyerEditPage»

Для других страниц, перечисленных выше, сделаны похожие страницы на редактирование

## **Текст** **программы**

Страница MainWindows – Главное окно, в которое подгружается разная информация из базы данных

**Xaml код:**

<Window x:Class="Furniture\_K.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:Furniture\_K"

mc:Ignorable="d"

Title="Учёт продажи офисной мебели" Height="720" Width="1080" ResizeMode="NoResize" WindowStyle="ThreeDBorderWindow" Icon="/FurnitureIcon.png" WindowStartupLocation="CenterScreen">

<Window.Background>

<LinearGradientBrush EndPoint="0.5,1" StartPoint="0.5,0">

<GradientStop Color="#FFB89771"/>

<GradientStop Color="#BDBDBD" Offset="1"/>

</LinearGradientBrush>

</Window.Background>

<Grid>

<Frame x:Name="MainFrame" NavigationUIVisibility="Hidden" Grid.Column="0" Grid.Row="1" Grid.ColumnSpan="5" Grid.RowSpan="4"

Source="Pages/MainPage.xaml"/>

<Border Grid.Column="0" Grid.Row="0" Grid.ColumnSpan="5" Margin="0" Background="#ffdadada">

<Border.Effect>

<DropShadowEffect BlurRadius="30" Color="LightGray" ShadowDepth="0"/>

</Border.Effect>

<StackPanel>

<Grid>

<Image Source="/Resources/FurnitureIcon.png" Height="140" Width="140" Margin="5 0 0 0" HorizontalAlignment="Left"/>

<TextBlock Text="Учёт продажи офисной мебели" VerticalAlignment="Center" HorizontalAlignment="Center" FontSize="50" Margin="100 0 0 0" FontWeight="Bold"/>

</Grid>

</StackPanel>

</Border>

<Grid.RowDefinitions>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

</Grid.ColumnDefinitions>

</Grid>

</Window>

**Cs код:**

using Furniture\_K.Classes; // Использую папку "Classes" для "MainWindow" //

namespace Furniture\_K

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

FrameApp.FrameOBJ = MainFrame; // Используем класс Frame в MainWindow //

ConnectHelper.FurnitureOBJ = new FurnitureEntities(); // Строка подключения БД для MainWindow //

}

}

}

Страница MainPage – Используется в качестве главного меню пользователя

Xaml код:

<Page x:Class="Furniture\_K.Pages.MainPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:Furniture\_K.Pages"

mc:Ignorable="d"

d:DesignHeight="576" d:DesignWidth="1080"

Title="MainPage">

<Grid>

<Border MinWidth="400" Margin="15" Background="#ffdadada" VerticalAlignment="Center" Padding="30" MaxWidth="550" CornerRadius="20">

<Border.Effect>

<DropShadowEffect BlurRadius="30" Color="LightGray" ShadowDepth="0"/>

</Border.Effect>

<StackPanel>

<TextBlock Text="Выберите окно" HorizontalAlignment="Center" FontSize="50" FontWeight="Bold" Margin="0 0 0 20"/>

<Button x:Name="btnShowBuyer" HorizontalAlignment="Center" FontSize="30" Content="Покупатели" Margin="10" Width="400" Height="50" Cursor="Hand" Click="btnShowBuyer\_Click"/>

<Button x:Name="btnShowContract" HorizontalAlignment="Center" FontSize="30" Content="Договоры" Margin="10" Width="400" Height="50" Cursor="Hand" Click="btnShowContract\_Click"/>

<Button x:Name="btnShowModel" HorizontalAlignment="Center" FontSize="30" Content="Модели мебели" Margin="10" Width="400" Height="50" Cursor="Hand" Click="btnShowModel\_Click"/>

<Button x:Name="btnShowSale" HorizontalAlignment="Center" FontSize="30" Content="Продажи" Margin="10" Width="400" Height="50" Cursor="Hand" Click="btnShowSale\_Click"/>

</StackPanel>

</Border>

</Grid>

</Page>

Cs код:

using Furniture\_K.Classes;

namespace Furniture\_K.Pages

{

/// <summary>

/// Логика взаимодействия для MainPage.xaml

/// </summary>

public partial class MainPage : Page

{

public MainPage()

{

InitializeComponent();

}

// ////////// //

// Покупатели //

// ////////// //

private void btnShowBuyer\_Click(object sender, RoutedEventArgs e)

{

FrameApp.FrameOBJ.Navigate(new Pages.BuyerPage());

}

// //////// //

// Договоры //

// //////// //

private void btnShowContract\_Click(object sender, RoutedEventArgs e)

{

FrameApp.FrameOBJ.Navigate(new Pages.ContractPage());

}

// ///////////// //

// Модели мебели //

// ///////////// //

private void btnShowModel\_Click(object sender, RoutedEventArgs e)

{

FrameApp.FrameOBJ.Navigate(new Pages.ModelPage());

}

// /////// //

// Продажи //

// /////// //

private void btnShowSale\_Click(object sender, RoutedEventArgs e)

{

FrameApp.FrameOBJ.Navigate(new Pages.SalePage());

}

}

}

Страница BuyerPage – Используется для вывода информации о покупателях.

Xaml код:

<Page x:Class="Furniture\_K.Pages.BuyerPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:Furniture\_K.Pages"

mc:Ignorable="d"

d:DesignHeight="576" d:DesignWidth="1080"

Title="BuyerPage">

<Grid>

<Border MinWidth="400" Margin="15" Background="#ffdadada" VerticalAlignment="Center" Padding="30" MaxWidth="900" CornerRadius="20">

<Border.Effect>

<DropShadowEffect BlurRadius="30" Color="LightGray" ShadowDepth="0"/>

</Border.Effect>

<StackPanel>

<TextBlock Text="Покупатели" HorizontalAlignment="Center" FontSize="30" FontWeight="Bold" Margin="0 0 0 20"/>

<Grid>

<Button x:Name="btnAddBuyer" HorizontalAlignment="Left" FontSize="18" Content="Добавить" Margin="250 10 10 10" Width="158" Height="30" Cursor="Hand" Click="btnAddBuyer\_Click"/>

<Button x:Name="btnEditBuyer" HorizontalAlignment="Right" FontSize="18" Content="Редактировать" Margin="10 10 250 10" Width="158" Height="30" Cursor="Hand" Click="btnEditBuyer\_Click"/>

</Grid>

<ComboBox x:Name="cmbFilt" Width="350" HorizontalAlignment="Center" VerticalAlignment="Center" Background="#ffff" Margin="5" SelectionChanged="cmbFilt\_SelectionChanged"/>

<DataGrid x:Name="dgBuyer" AutoGenerateColumns="False" IsReadOnly="True" Height="270">

<DataGrid.Columns>

<DataGridTextColumn Header="ФИО" Binding="{Binding NameBuyer}" Width="\*"/>

<DataGridTextColumn Header="Адрес" Binding="{Binding Address}" Width="\*"/>

<DataGridTextColumn Header="Номер телефона" Binding="{Binding Phone}" Width="\*"/>

</DataGrid.Columns>

</DataGrid>

<Button x:Name="btnBack" HorizontalAlignment="Center" FontSize="18" Content="Назад" Margin="10" Width="150" Height="30" Cursor="Hand" Click="btnBack\_Click"/>

</StackPanel>

</Border>

</Grid>

</Page>

Cs код:

using Furniture\_K.Classes;

namespace Furniture\_K.Pages

{

/// <summary>

/// Логика взаимодействия для BuyerPage.xaml

/// </summary>

public partial class BuyerPage : Page

{

public BuyerPage()

{

InitializeComponent();

dgBuyer.ItemsSource = ConnectHelper.FurnitureOBJ.Buyer.ToList(); // Берёт данные из таблицы "Buyer"

cmbFilt.SelectedValuePath = "idBuyer";

cmbFilt.DisplayMemberPath = "Address";

cmbFilt.ItemsSource = ConnectHelper.FurnitureOBJ.Buyer.ToList();

}

// ////////// //

// Фильтрация //

// ////////// //

private void cmbFilt\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

int BuyerID = (int)cmbFilt.SelectedValue;

dgBuyer.ItemsSource = ConnectHelper.FurnitureOBJ.Buyer.Where(x => x.idBuyer == BuyerID).ToList();

}

// ///////////////// //

// Возвращене в меню //

// ///////////////// //

private void btnBack\_Click(object sender, RoutedEventArgs e)

{

FrameApp.FrameOBJ.Navigate(new Pages.MainPage());

}

// ///////////////// //

// Добавление данных //

// ///////////////// //

private void btnAddBuyer\_Click(object sender, RoutedEventArgs e)

{

FrameApp.FrameOBJ.Navigate(new BuyerAddPage());

}

// ///////////////////// //

// Редактирование данных //

// ///////////////////// //

private void btnEditBuyer\_Click(object sender, RoutedEventArgs e)

{

Buyer buyer = dgBuyer.SelectedItem as Buyer;

if(buyer == null)

{

MessageBox.Show("Покупатель не выбран");

}

else

{

FrameApp.FrameOBJ.Navigate(new BuyerEditPage(buyer));

}

}

}

}

Страница ContractPage – Используется для вывода информации о договорах.

Xaml код:

<Page x:Class="Furniture\_K.Pages.ContractPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:Furniture\_K.Pages"

mc:Ignorable="d"

d:DesignHeight="576" d:DesignWidth="1080"

Title="ContractPage">

<Grid>

<Border MinWidth="400" Margin="15" Background="#ffdadada" VerticalAlignment="Center" Padding="30" MaxWidth="900" CornerRadius="20">

<Border.Effect>

<DropShadowEffect BlurRadius="30" Color="LightGray" ShadowDepth="0"/>

</Border.Effect>

<StackPanel>

<TextBlock Text="Договоры" HorizontalAlignment="Center" FontSize="30" FontWeight="Bold" Margin="0 0 0 20"/>

<Grid>

<Button x:Name="btnAddContract" HorizontalAlignment="Left" FontSize="18" Content="Добавить" Margin="250 10 10 10" Width="158" Height="30" Cursor="Hand" Click="btnAddContract\_Click"/>

<Button x:Name="btnEditContract" HorizontalAlignment="Right" FontSize="18" Content="Редактировать" Margin="10 10 250 10" Width="158" Height="30" Cursor="Hand" Click="btnEditContract\_Click"/>

</Grid>

<ComboBox x:Name="cmbFilt" Width="350" HorizontalAlignment="Center" VerticalAlignment="Center" Background="#ffff" Margin="5" SelectionChanged="cmbFilt\_SelectionChanged"/>

<DataGrid x:Name="dgContract" AutoGenerateColumns="False" IsReadOnly="True" Height="270">

<DataGrid.Columns>

<DataGridTextColumn Header="Номер контракта" Binding="{Binding ContractNumber}" Width="\*"/>

<DataGridTextColumn Header="Дата регистрации" Binding="{Binding RegistrationDate}" Width="\*"/>

<DataGridTextColumn Header="Дата исполнения" Binding="{Binding DateOfExecution}" Width="\*"/>

<DataGridTextColumn Header="Покупатель" Binding="{Binding Buyer.NameBuyer}" Width="\*"/>

</DataGrid.Columns>

</DataGrid>

<Button x:Name="btnBack" HorizontalAlignment="Center" FontSize="18" Content="Назад" Margin="10" Width="150" Height="30" Cursor="Hand" Click="btnBack\_Click"/>

</StackPanel>

</Border>

</Grid>

</Page>

Cs код:

using Furniture\_K.Classes;

namespace Furniture\_K.Pages

{

/// <summary>

/// Логика взаимодействия для ContractPage.xaml

/// </summary>

public partial class ContractPage : Page

{

public ContractPage()

{

InitializeComponent();

dgContract.ItemsSource = ConnectHelper.FurnitureOBJ.Contract.ToList(); // Берёт данные из таблицы "Contract"

cmbFilt.SelectedValuePath = "idConttract";

cmbFilt.DisplayMemberPath = "DateOfExecution";

cmbFilt.ItemsSource = ConnectHelper.FurnitureOBJ.Contract.ToList();

}

// ///////////////// //

// Возвращене в меню //

// ///////////////// //

private void btnBack\_Click(object sender, RoutedEventArgs e)

{

FrameApp.FrameOBJ.Navigate(new Pages.MainPage());

}

// ////////// //

// Фильтрация //

// ////////// //

private void cmbFilt\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

int ContractID = (int)cmbFilt.SelectedValue;

dgContract.ItemsSource = ConnectHelper.FurnitureOBJ.Contract.Where(x => x.idConttract == ContractID).ToList();

}

// ////////// //

// Добавление //

// ////////// //

private void btnAddContract\_Click(object sender, RoutedEventArgs e)

{

FrameApp.FrameOBJ.Navigate(new ContractAddPage());

}

// ////////////// //

// Редактирование //

// ////////////// //

private void btnEditContract\_Click(object sender, RoutedEventArgs e)

{

Contract contract = dgContract.SelectedItem as Contract;

if (contract == null)

{

MessageBox.Show("Договор не выбран");

}

else

{

FrameApp.FrameOBJ.Navigate(new ContractEditPage(contract));

}

}

}

}

Страница ModelPage – Используется для вывода информации о моделях мебели.

Xaml код:

<Page x:Class="Furniture\_K.Pages.ModelPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:Furniture\_K.Pages"

mc:Ignorable="d"

d:DesignHeight="576" d:DesignWidth="1080"

Title="ModelPage">

<Grid>

<Border MinWidth="400" Margin="15" Background="#ffdadada" VerticalAlignment="Center" Padding="30" MaxWidth="900" CornerRadius="20">

<Border.Effect>

<DropShadowEffect BlurRadius="30" Color="LightGray" ShadowDepth="0"/>

</Border.Effect>

<StackPanel>

<TextBlock Text="Модели мебели" HorizontalAlignment="Center" FontSize="30" FontWeight="Bold" Margin="0 0 0 20"/>

<Grid>

<Button x:Name="btnAddModel" HorizontalAlignment="Left" FontSize="18" Content="Добавить" Margin="250 10 10 10" Width="158" Height="30" Cursor="Hand" Click="btnAddModel\_Click"/>

<Button x:Name="btnEditModel" HorizontalAlignment="Right" FontSize="18" Content="Редактировать" Margin="10 10 250 10" Width="158" Height="30" Cursor="Hand" Click="btnEditModel\_Click"/>

</Grid>

<ComboBox x:Name="cmbFilt" Width="350" HorizontalAlignment="Center" VerticalAlignment="Center" Background="#ffff" Margin="5" SelectionChanged="cmbFilt\_SelectionChanged"/>

<DataGrid x:Name="dgModel" AutoGenerateColumns="False" IsReadOnly="True" Height="270">

<DataGrid.Columns>

<DataGridTextColumn Header="Название мебели" Binding="{Binding FurnitureName}" Width="\*"/>

<DataGridTextColumn Header="Модель" Binding="{Binding Model1}" Width="\*"/>

<DataGridTextColumn Header="Цена модели" Binding="{Binding ModelPrice}" Width="\*"/>

<DataGridTextColumn Header="Назначение мебели" Binding="{Binding SpecificationsModel.SpecificationsModel1}" Width="\*"/>

</DataGrid.Columns>

</DataGrid>

<Button x:Name="btnBack" HorizontalAlignment="Center" FontSize="18" Content="Назад" Margin="10" Width="150" Height="30" Cursor="Hand" Click="btnBack\_Click"/>

</StackPanel>

</Border>

</Grid>

</Page>

Cs код:

using Furniture\_K.Classes;

namespace Furniture\_K.Pages

{

/// <summary>

/// Логика взаимодействия для ModelPage.xaml

/// </summary>

public partial class ModelPage : Page

{

public ModelPage()

{

InitializeComponent();

dgModel.ItemsSource = ConnectHelper.FurnitureOBJ.Model.ToList(); // Берёт данные из таблицы "Model"

cmbFilt.SelectedValuePath = "idSpecificationsModel";

cmbFilt.DisplayMemberPath = "SpecificationsModel1";

cmbFilt.ItemsSource = ConnectHelper.FurnitureOBJ.SpecificationsModel.ToList();

}

// ///////////////// //

// Возвращене в меню //

// ///////////////// //

private void btnBack\_Click(object sender, RoutedEventArgs e)

{

FrameApp.FrameOBJ.Navigate(new Pages.MainPage());

}

// ////////// //

// Добавление //

// ////////// //

private void btnAddModel\_Click(object sender, RoutedEventArgs e)

{

FrameApp.FrameOBJ.Navigate(new ModelAddPage());

}

// ////////////// //

// Редактирование //

// ////////////// //

private void btnEditModel\_Click(object sender, RoutedEventArgs e)

{

Model model = dgModel.SelectedItem as Model;

if (model == null)

{

MessageBox.Show("Модель мебели не выбрана");

}

else

{

FrameApp.FrameOBJ.Navigate(new ModelEditPage(model));

}

}

// ////////// //

// Фильтрация //

// ////////// //

private void cmbFilt\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

int ModelID = (int)cmbFilt.SelectedValue;

dgModel.ItemsSource = ConnectHelper.FurnitureOBJ.Model.Where(x => x.idSpecificationsModel == ModelID).ToList();

}

}

}

Страница SalePage – Используется для вывода информации о продажах.

Код Xaml:

<Page x:Class="Furniture\_K.Pages.SalePage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:Furniture\_K.Pages"

mc:Ignorable="d"

d:DesignHeight="576" d:DesignWidth="1080"

Title="SalePage">

<Grid>

<Border MinWidth="400" Margin="15" Background="#ffdadada" VerticalAlignment="Center" Padding="30" MaxWidth="900" CornerRadius="20">

<Border.Effect>

<DropShadowEffect BlurRadius="30" Color="LightGray" ShadowDepth="0"/>

</Border.Effect>

<StackPanel>

<TextBlock Text="Продажи" HorizontalAlignment="Center" FontSize="30" FontWeight="Bold" Margin="0 0 0 20"/>

<Grid>

<Button x:Name="btnAddSale" HorizontalAlignment="Left" FontSize="18" Content="Добавить" Margin="250 10 10 10" Width="158" Height="30" Cursor="Hand" Click="btnAddSale\_Click"/>

<Button x:Name="btnEditSale" HorizontalAlignment="Right" FontSize="18" Content="Редактировать" Margin="10 10 250 10" Width="158" Height="30" Cursor="Hand" Click="btnEditSale\_Click"/>

</Grid>

<ComboBox x:Name="cmbFilt" Width="350" HorizontalAlignment="Center" VerticalAlignment="Center" Background="#ffff" Margin="5" SelectionChanged="cmbFilt\_SelectionChanged"/>

<DataGrid x:Name="dgSale" AutoGenerateColumns="False" IsReadOnly="True" Height="270">

<DataGrid.Columns>

<DataGridTextColumn Header="Количество" Binding="{Binding Quantity}" Width="\*"/>

<DataGridTextColumn Header="Номер договора" Binding="{Binding Contract.ContractNumber}" Width="\*"/>

<DataGridTextColumn Header="Название мебели" Binding="{Binding Model.FurnitureName}" Width="\*"/>

<DataGridTextColumn Header="Цена модели" Binding="{Binding Model.ModelPrice}" Width="\*"/>

</DataGrid.Columns>

</DataGrid>

<Button x:Name="btnBack" HorizontalAlignment="Center" FontSize="18" Content="Назад" Margin="10" Width="150" Height="30" Cursor="Hand" Click="btnBack\_Click"/>

</StackPanel>

</Border>

</Grid>

</Page>

Код Cs:

using Furniture\_K.Classes;

namespace Furniture\_K.Pages

{

/// <summary>

/// Логика взаимодействия для SalePage.xaml

/// </summary>

public partial class SalePage : Page

{

public SalePage()

{

InitializeComponent();

dgSale.ItemsSource = ConnectHelper.FurnitureOBJ.Sale.ToList(); // Берёт данные из таблицы "Sale"

cmbFilt.SelectedValuePath = "idModel";

cmbFilt.DisplayMemberPath = "FurnitureName";

cmbFilt.ItemsSource = ConnectHelper.FurnitureOBJ.Model.ToList();

}

// ///////////////// //

// Возвращене в меню //

// ///////////////// //

private void btnBack\_Click(object sender, RoutedEventArgs e)

{

FrameApp.FrameOBJ.Navigate(new Pages.MainPage());

}

// ////////// //

// Добавление //

// ////////// //

private void btnAddSale\_Click(object sender, RoutedEventArgs e)

{

FrameApp.FrameOBJ.Navigate(new SaleAddPage());

}

// ////////////// //

// Редактирование //

// ////////////// //

private void btnEditSale\_Click(object sender, RoutedEventArgs e)

{

Sale sale = dgSale.SelectedItem as Sale;

if (sale == null)

{

MessageBox.Show("Продажа не выбрана");

}

else

{

FrameApp.FrameOBJ.Navigate(new SaleEditPage(sale));

}

}

// ////////// //

// Фильтрация //

// ////////// //

private void cmbFilt\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

int SaleID = (int)cmbFilt.SelectedValue;

dgSale.ItemsSource = ConnectHelper.FurnitureOBJ.Sale.Where(x => x.idModel == SaleID).ToList();

}

}

}

Страница BuyerAddPage – Используется для добавления записей с информацией о покупателях

Код Xaml:

<Page x:Class="Furniture\_K.Pages.BuyerAddPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:Furniture\_K.Pages"

xmlns:materialDesign="http://materialdesigninxaml.net/winfx/xaml/themes"

mc:Ignorable="d"

d:DesignHeight="576" d:DesignWidth="1080"

Title="BuyerAddPage">

<Grid>

<Border MinWidth="400" Margin="15" Background="#ffdadada" VerticalAlignment="Center" Padding="30" MaxWidth="900" CornerRadius="20">

<Border.Effect>

<DropShadowEffect BlurRadius="30" Color="LightGray" ShadowDepth="0"/>

</Border.Effect>

<StackPanel Cursor="Arrow">

<TextBlock Text="Добавить" HorizontalAlignment="Center" FontSize="30" FontWeight="Bold" Margin="0 0 0 20"/>

<TextBox x:Name="FIO" materialDesign:HintAssist.Hint="ФИО" Style="{StaticResource MaterialDesignFloatingHintTextBox}" FontSize="20"/>

<TextBox x:Name="Address" materialDesign:HintAssist.Hint="Адрес" Style="{StaticResource MaterialDesignFloatingHintTextBox}" FontSize="20"/>

<TextBox x:Name="Phone" materialDesign:HintAssist.Hint="Номер телефона" Style="{StaticResource MaterialDesignFloatingHintTextBox}" FontSize="20"/>

<Button x:Name="btnAdd" HorizontalAlignment="Right" FontSize="18" Content="Добавить" Margin="15" Width="150" Height="30" Cursor="Hand" Click="btnAdd\_Click"/>

<Button x:Name="btnBack" HorizontalAlignment="Center" VerticalAlignment="Bottom" FontSize="18" Content="Назад" Margin="0 25 0 0" Width="150" Height="30" Cursor="Hand" Click="btnBack\_Click"/>

</StackPanel>

</Border>

</Grid>

</Page>

Код Cs:

using Furniture\_K.Classes;

namespace Furniture\_K.Pages

{

/// <summary>

/// Логика взаимодействия для BuyerAddPage.xaml

/// </summary>

public partial class BuyerAddPage : Page

{

public BuyerAddPage()

{

InitializeComponent();

}

// //////////////////////////// //

// Возвращене в таблицу "Buyer" //

// //////////////////////////// //

private void btnBack\_Click(object sender, RoutedEventArgs e)

{

FrameApp.FrameOBJ.Navigate(new Pages.BuyerPage());

}

// ///////////////// //

// Добавление данных //

// ///////////////// //

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

{

// Проверка на пустое и нулевое значение в полях

if (String.IsNullOrEmpty(FIO.Text) || String.IsNullOrEmpty(Address.Text) || String.IsNullOrEmpty(Phone.Text))

{

MessageBox.Show("Все поля должны быть заполнены!");

}

else

{

// Проверка на наличие данных в БД

if (ConnectHelper.FurnitureOBJ.Buyer.Where(u => u.NameBuyer == FIO.Text && u.Address == Address.Text && u.Phone == Phone.Text || u.Phone == Phone.Text).FirstOrDefault()!=null)

{

MessageBox.Show("Такой покупатель уже есть!");

}

else

{

Buyer buyer = new Buyer()

{

NameBuyer = FIO.Text,

Address = Address.Text,

Phone = Phone.Text

};

ConnectHelper.FurnitureOBJ.Buyer.Add(buyer); // Добавление данных в таблицу "Buyer"

ConnectHelper.FurnitureOBJ.SaveChanges();

MessageBox.Show("Данные успешно добавлены");

}

}

}

}

}

Страница BuyerEditPage – Используется для редактирования или корректирования уже имеющихся записей о покупателях

Код Xaml:

<Page x:Class="Furniture\_K.Pages.BuyerEditPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:Furniture\_K.Pages"

xmlns:materialDesign="http://materialdesigninxaml.net/winfx/xaml/themes"

mc:Ignorable="d"

d:DesignHeight="576" d:DesignWidth="1080"

Title="BuyerEditPage">

<Grid>

<Border MinWidth="400" Margin="15" Background="#ffdadada" VerticalAlignment="Center" Padding="30" MaxWidth="900" CornerRadius="20">

<Border.Effect>

<DropShadowEffect BlurRadius="30" Color="LightGray" ShadowDepth="0"/>

</Border.Effect>

<StackPanel>

<TextBlock Text="Редактировать" HorizontalAlignment="Center" FontSize="30" FontWeight="Bold" Margin="0 0 0 20"/>

<TextBox x:Name="FIO" materialDesign:HintAssist.Hint="ФИО" Style="{StaticResource MaterialDesignFloatingHintTextBox}" FontSize="20"/>

<TextBox x:Name="Address" materialDesign:HintAssist.Hint="Адрес" Style="{StaticResource MaterialDesignFloatingHintTextBox}" FontSize="20"/>

<TextBox x:Name="Phone" materialDesign:HintAssist.Hint="Номер телефона" Style="{StaticResource MaterialDesignFloatingHintTextBox}" FontSize="20"/>

<Button x:Name="btnEdit" HorizontalAlignment="Right" FontSize="18" Content="Сохранить" Margin="15" Width="150" Height="30" Cursor="Hand" Click="btnEdit\_Click"/>

<Button x:Name="btnBack" HorizontalAlignment="Center" VerticalAlignment="Bottom" FontSize="18" Content="Назад" Margin="0 25 0 0" Width="150" Height="30" Cursor="Hand" Click="btnBack\_Click"/>

</StackPanel>

</Border>

</Grid>

</Page>

Код Cs:

using Furniture\_K.Classes;

namespace Furniture\_K.Pages

{

/// <summary>

/// Логика взаимодействия для BuyerEditPage.xaml

/// </summary>

public partial class BuyerEditPage : Page

{

private Buyer br = new Buyer();

public BuyerEditPage(Buyer buyer) // Вписываем название таблицы и задаём новую переменную

{

InitializeComponent();

br = buyer;

FIO.Text = br.NameBuyer;

Address.Text = br.Address;

Phone.Text = br.Phone;

}

// //////////////////////////// //

// Возвращене в таблицу "Buyer" //

// //////////////////////////// //

private void btnBack\_Click(object sender, RoutedEventArgs e)

{

FrameApp.FrameOBJ.Navigate(new Pages.BuyerPage());

}

// ///////////////////// //

// Редактирование данных //

// ///////////////////// //

private void btnEdit\_Click(object sender, RoutedEventArgs e)

{

br.NameBuyer = FIO.Text;

br.Address = Address.Text;

br.Phone = Phone.Text;

if(br.idBuyer == 0)

{

ConnectHelper.FurnitureOBJ.Buyer.Add(br);

}

try

{

ConnectHelper.FurnitureOBJ.SaveChanges();

MessageBox.Show("Данные успешно изменены");

}

catch(Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

}

}

# 4. **Руководство пользователя**

## 4.1 Выполнение программного модуля

Для запуска программы надо запустить Furniture\_K.exe, который находится по пути:

C:\Users\Рома\Desktop\Курсовая 5.0\VISUAL STUDIO\_Проект\Furniture\_K\Furniture\_K\bin\Debug

Существует только 1 роль: Администратор – он ведёт учёт по всей базе данных.

Администратор – может добавлять, редактировать и просматривать все таблицы, которые есть в базе данных через приложение.

Главное меню Администратора, где он может выбрать нужное ему окно с информацией по магазину:

:

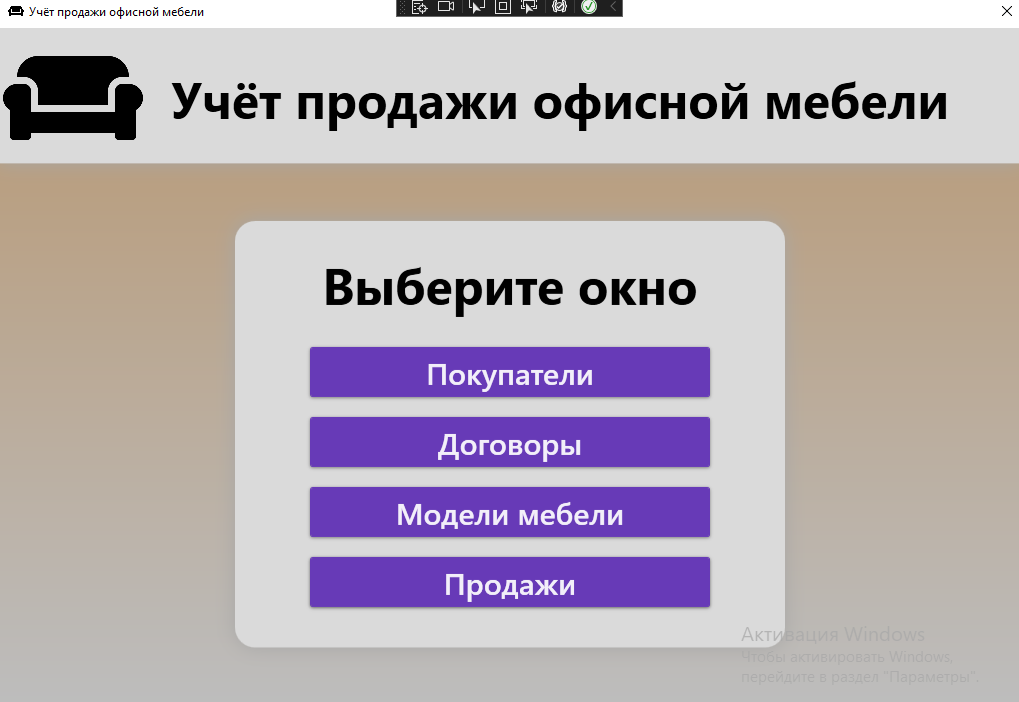


Рис. 38 «Главное меню»

На эту страницу попадаем после нажатия кнопки «Покупатели», здесь так же выбор того, что нужно сделать дальше:

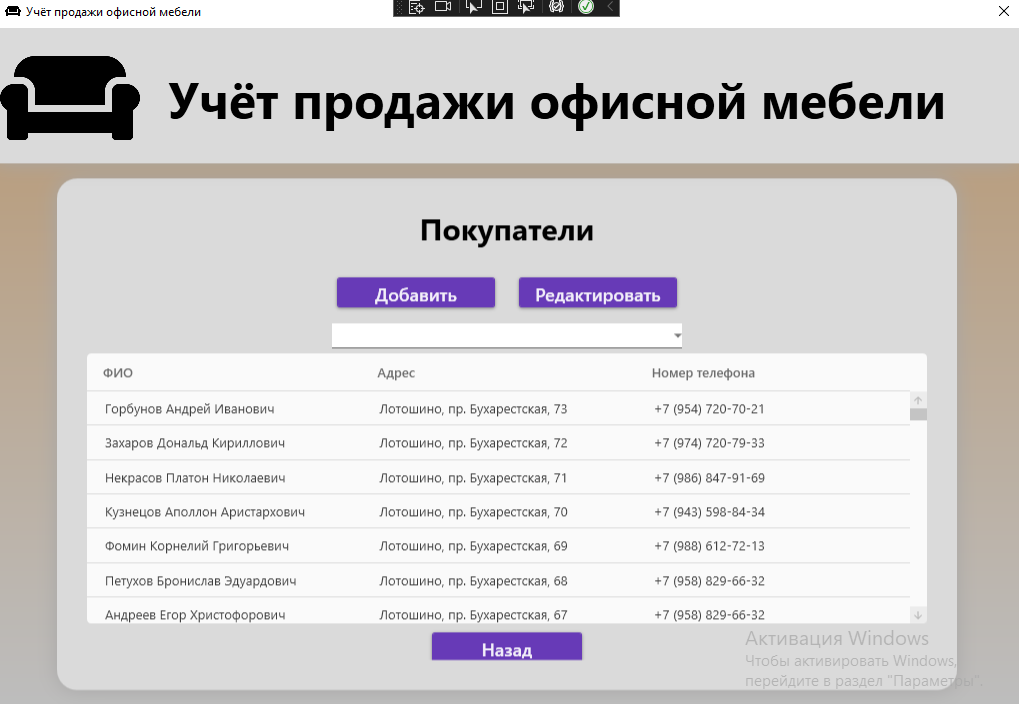


Рис.39 «Страница с информацией покупателей»

На этой странице Администратор может добавить новые записи о покупателе в БД:

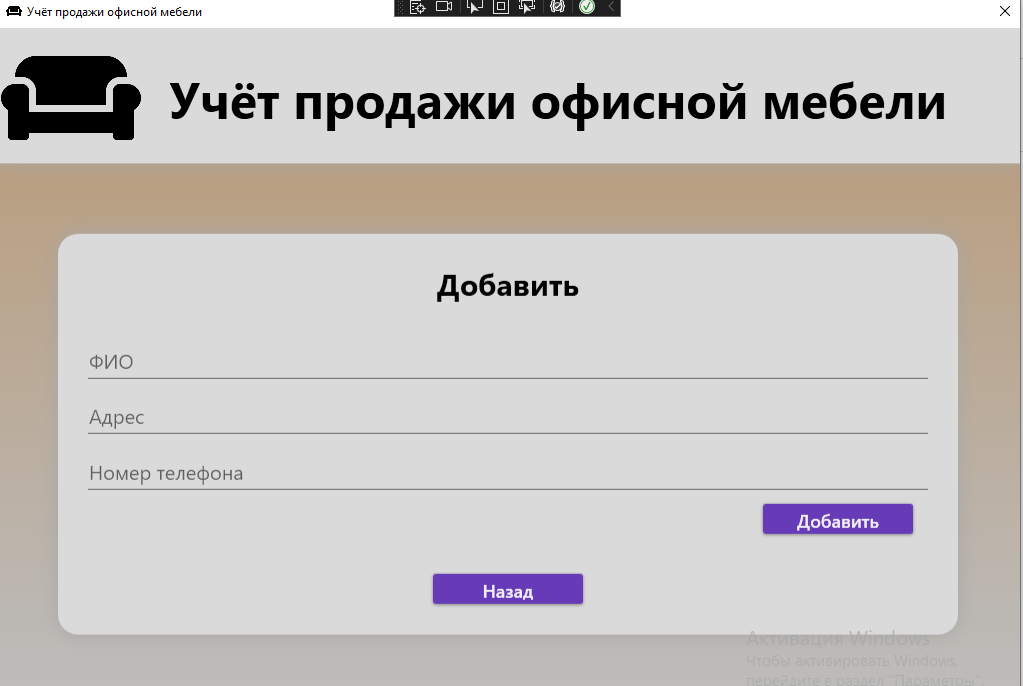


Рис.40 «Добавление покупателей»

На этой странице Администратор может редактировать и корректировать уже имеющиеся записи о покупателе в БД:

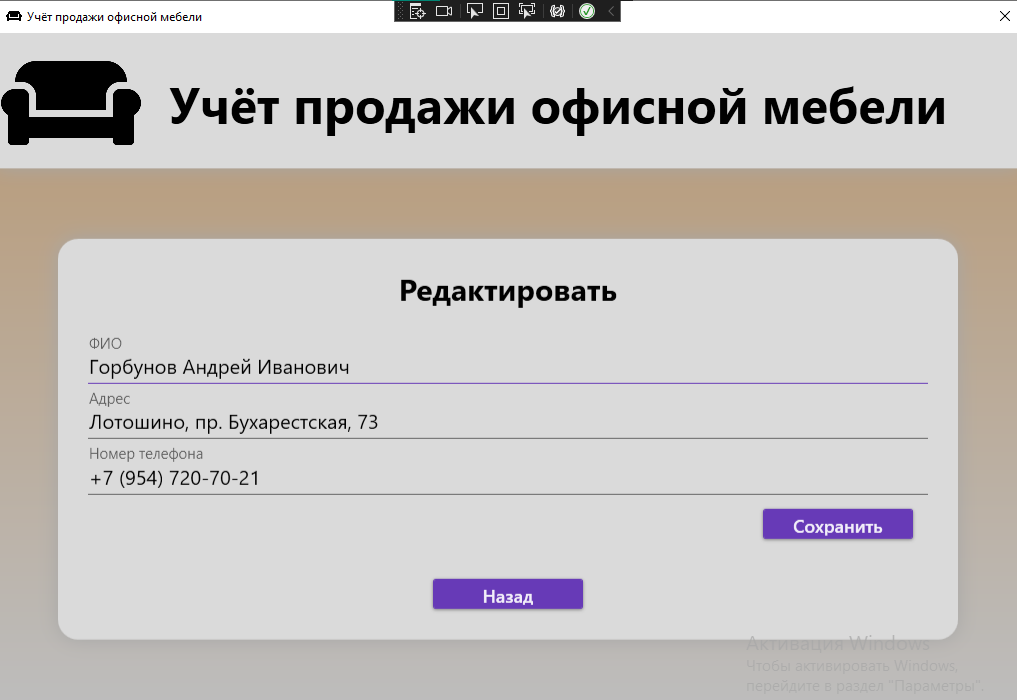


Рис.41 «Редактирование покупателей»

На эту страницу попадаем после нажатия кнопки «Договоры» из главного меню, здесь так же выбор того, что нужно сделать дальше:

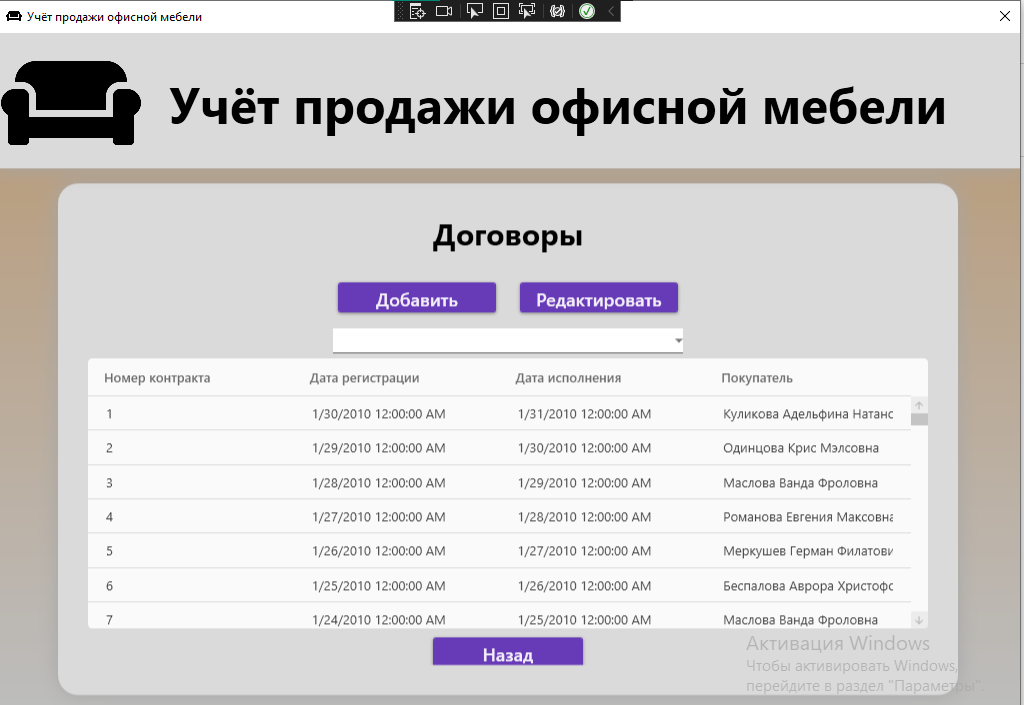


Рис.42 «Страница с информацией договоров»

На этой странице Администратор может добавить новые записи о договорах в БД:

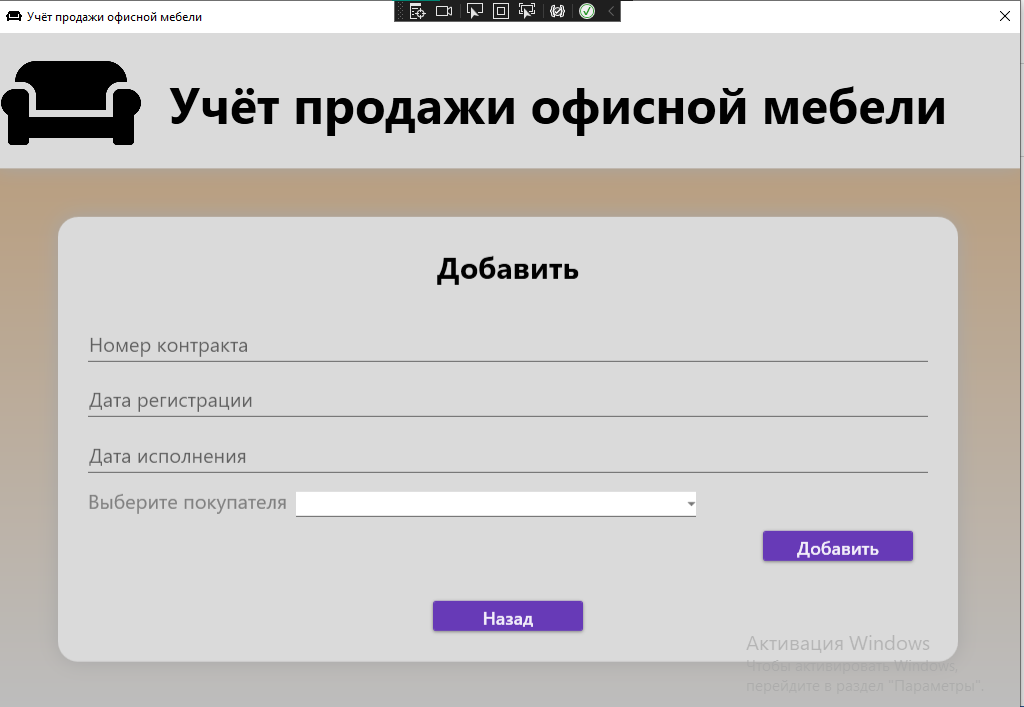


Рис.43 «Добавление покупателей»

На этой странице Администратор может редактировать и корректировать уже имеющиеся записи о договорах в БД:

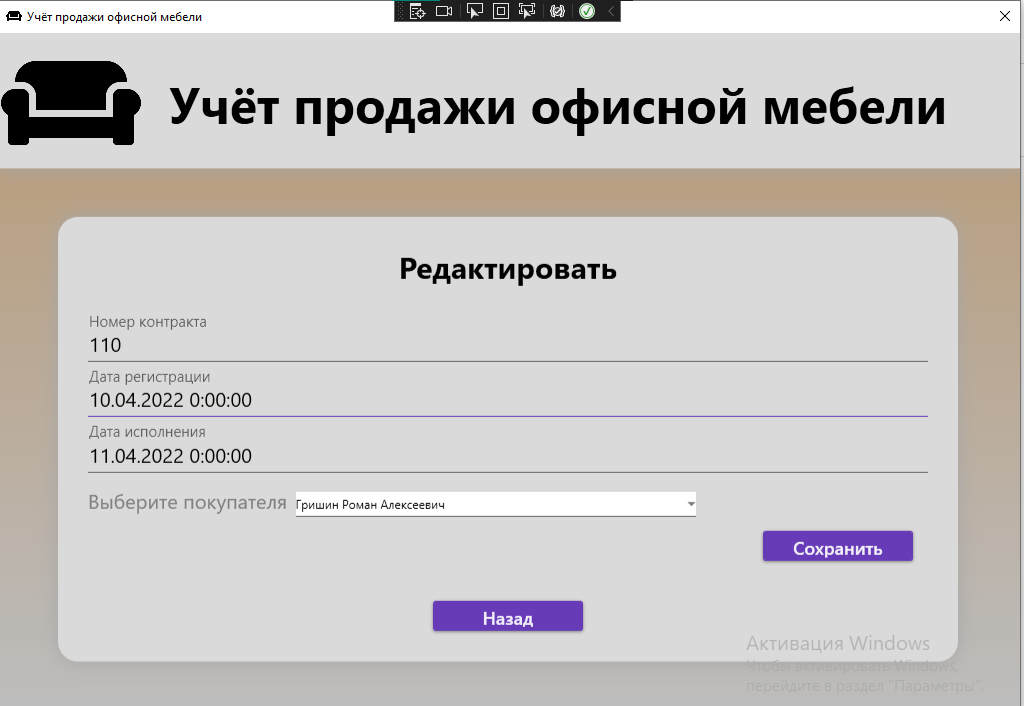


Рис.44 «Редактирование договоров»

На эту страницу попадаем после нажатия кнопки «Модели мебели», здесь так же выбор того, что нужно сделать дальше:

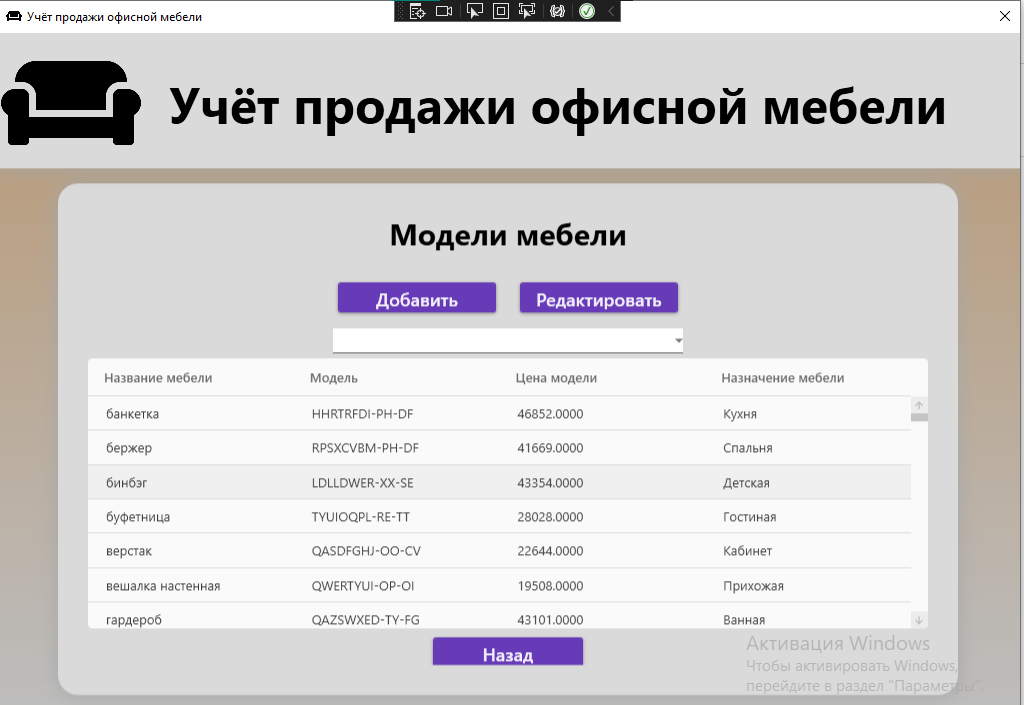


Рис.45 «Страница с информацией моделей мебели»

На этой странице Администратор может добавить новые записи о моделях мебели в БД:

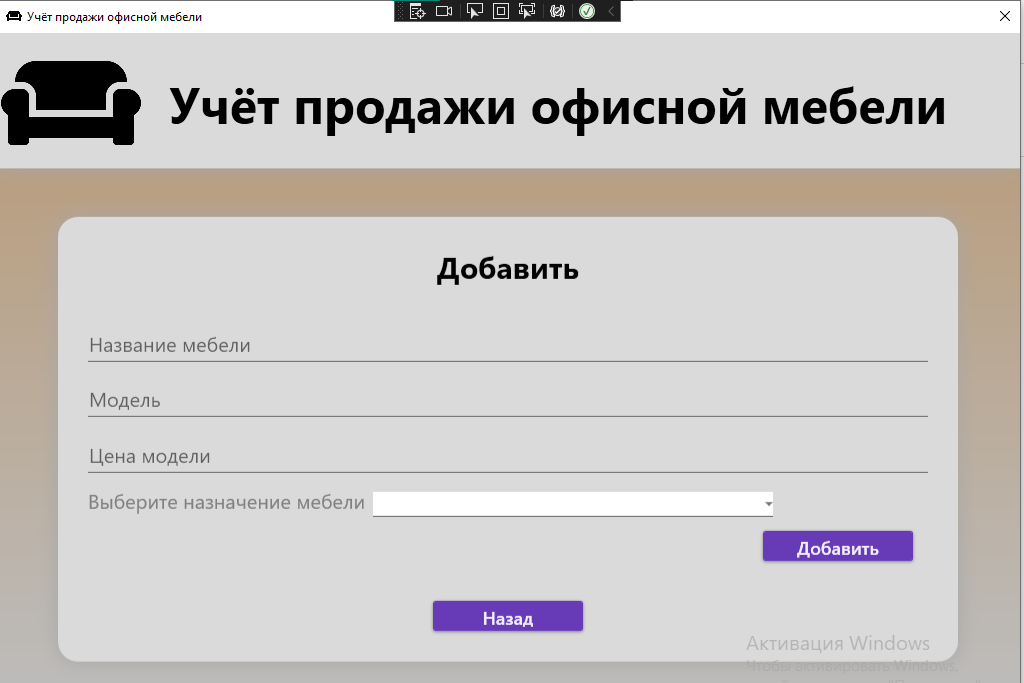


Рис.46 «Добавление моделей мебели»

На этой странице Администратор может редактировать и корректировать уже имеющиеся записи о моделях мебели в БД:

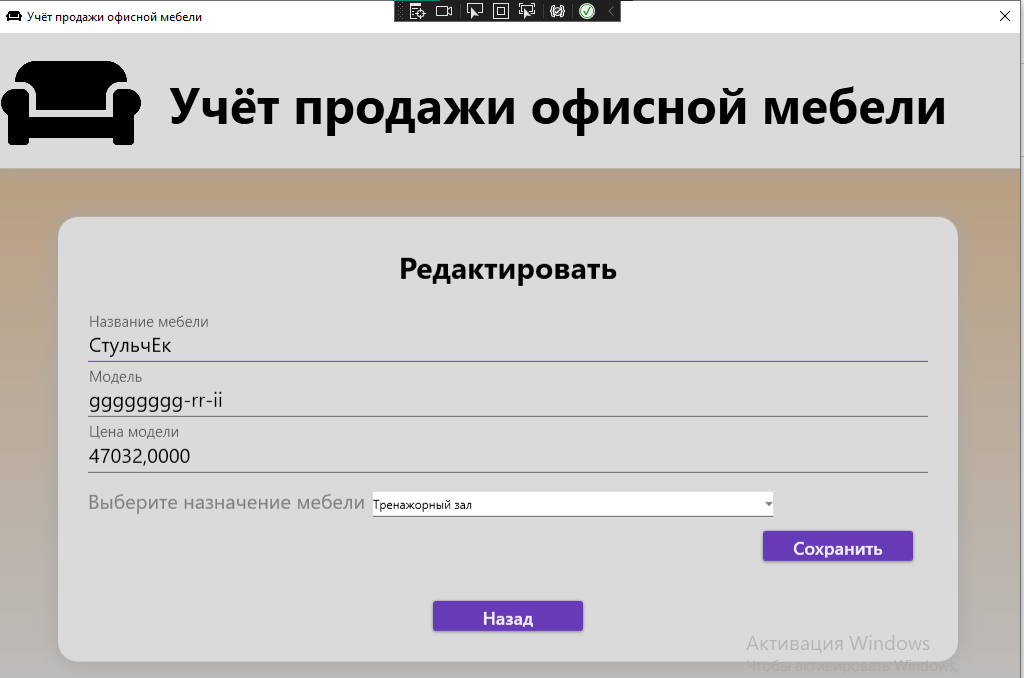


Рис.47 «Редактирование моделей мебели»

На эту страницу попадаем после нажатия кнопки «Продажи», здесь так же выбор того, что нужно сделать дальше:

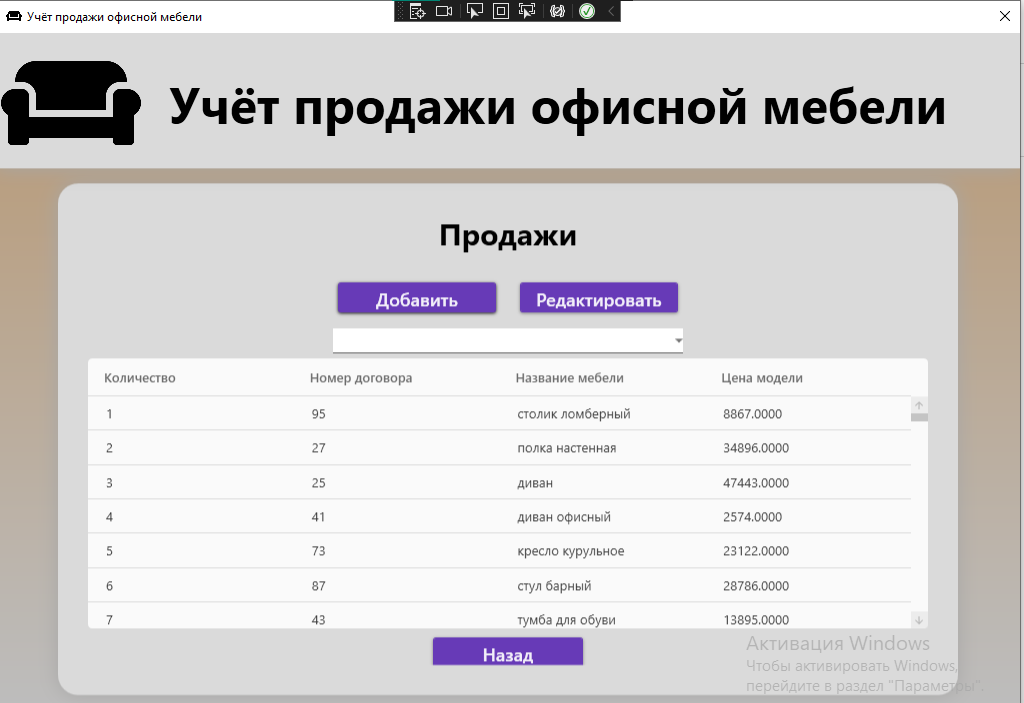


Рис.48 «Страница с информацией продаж»

На этой странице Администратор может добавить новые записи о продажах в БД:

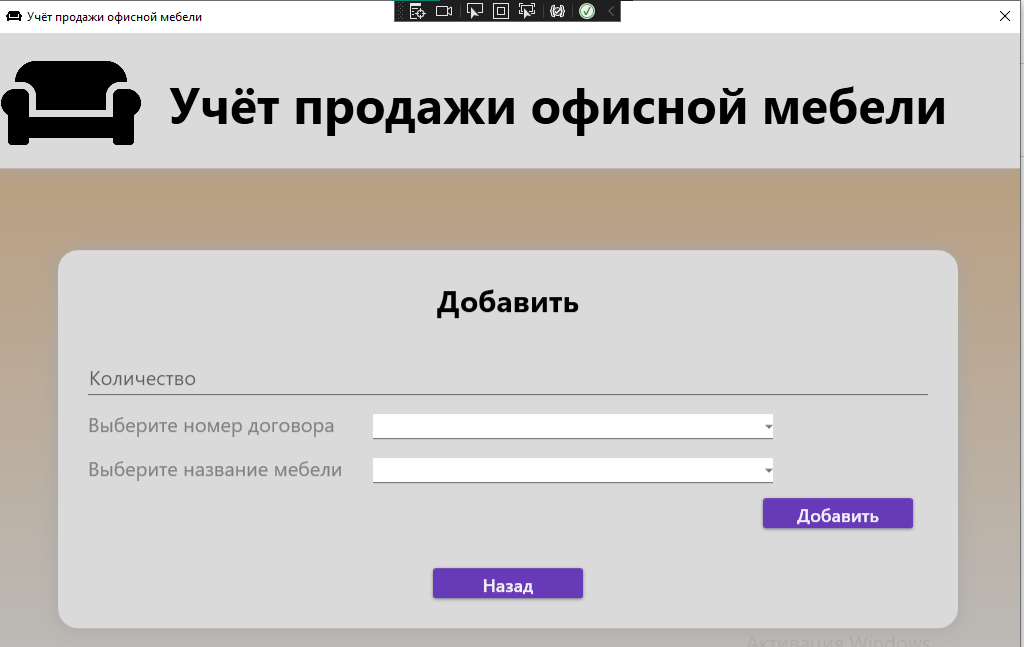


Рис.49 «Добавление продаж»

На этой странице Администратор может редактировать и корректировать уже имеющиеся записи о продажах в БД:

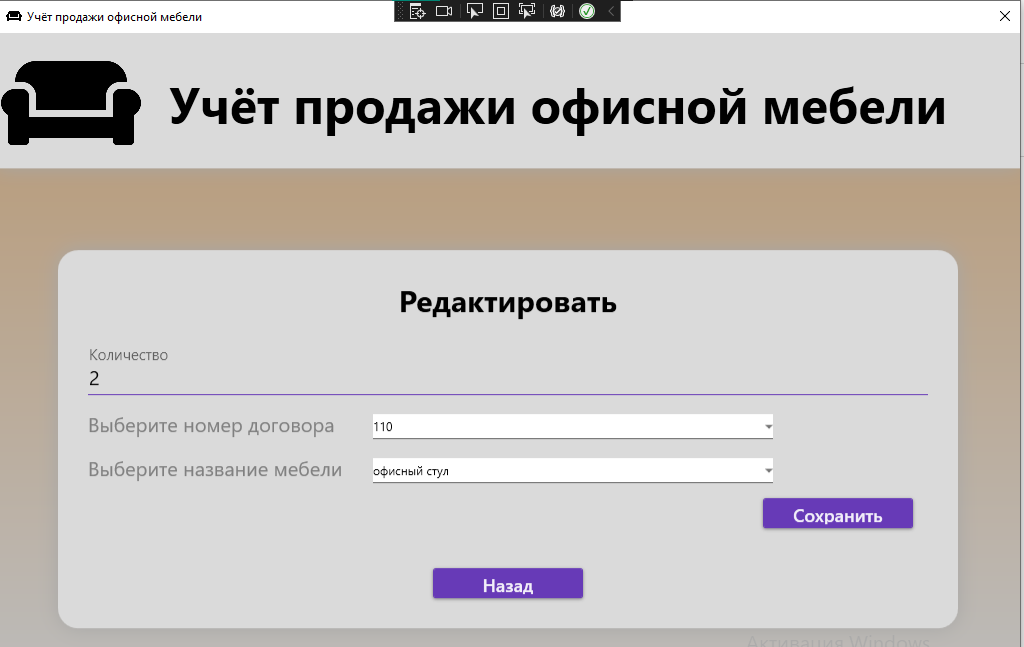


Рис.50 «Редактирование продаж»

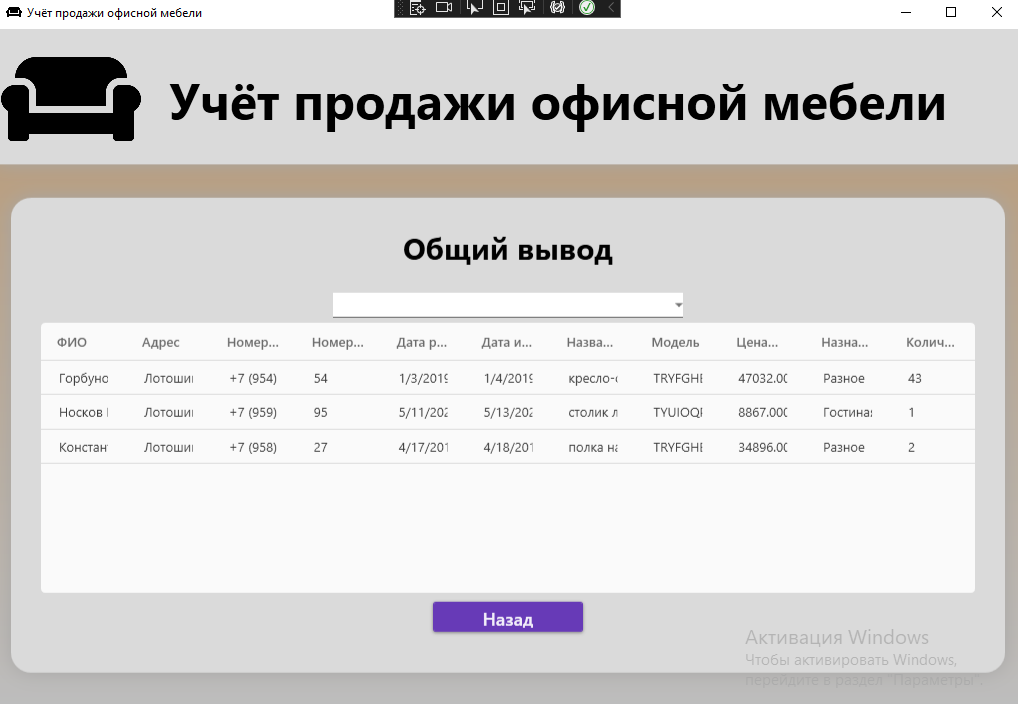


Рис.51 «Общий вывод данных»

## 4.2 Сообщения пользователю

Приведено описание сообщений пользователю и возможные действия пользователя в ответ на эти сообщения

* Сообщение о не выбранной записи покупателя в таблице

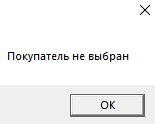


Рис.52 «Не выбрана запись покупателя»

* Сообщение о существующей записи какого-либо покупателя в БД

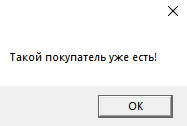


Рис.53 «Наличие записи покупателя в БД»

* Сообщение об успешном выполнении запроса на добавление записи покупателя в БД

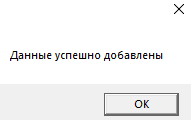


Рис.54 «Успешное добавление записи покупателя в БД»

* Сообщение об успешном выполнении запроса на обновление записи покупателя в БД

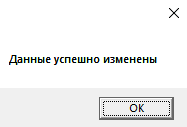


Рис.55 «Успешное обновление записи покупателя в БД»

# 5. **Тестирование** **и отладка программного модуля**

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 (ISO/IEC TR 19759:2005).

Таблица 1. Баги

|  |  |  |  |
| --- | --- | --- | --- |
| **Заголовок Ошибки** | **Действия** | **Ожидаемый результат** | **Фактический результат** |
| Добавление уже имеющейся записи договора в БД | 1. Открыть окно «Договоры» 🡪 «Добавить» 2. Ввести данные, например: (110, 2022.10.04, 2022.11.04, Гришин Роман Алексеевич) | Ошибка | Данные добавлены |
| Дата устанавливается по иностранной раскладке в зависимости от способа записи даты при редактировании или добавлении | 1. Открыть окно «Договоры» 🡪 «Добавить» или «Редактировать» 2. Ввести или отредактировать дату 3. Нажать на кнопку «Добавить» или «Сохранить» | Выполнит запрос на добавление или обновление | Дата в БД буде в формате той страны, для которой способ записи является правильным |

# Заключение

Для информационной системы по продаже офисной мебели был разработан программный модуль.

В результате выполненного исследования в рамках курсового проектирования разработан программный модуль для информационной системы по продаже офисной мебели. В процессе разработки использовалась СУБД SQL Server 2017, применена технология Entity Framework, язык программирования C# Visual Studio. Приложение соответствует поставленным задачам, реализованы функции: просмотр, добавление, фильтрация, выборка следующих таблиц: меню, заказ, рецепты. Программный модуль может быть установлен на любую операционную систему, начиная с Windows 7 x32.

В дальнейшем в приложении могут быть реализованы функции:

**1. Возможность удаления данных**

* 1. Открыть любую таблицу с просмотром информации
  2. Нажать на нужное поле
  3. Нажать на кнопку удалить

**2. Возможность иметь личный кабинет**

2.1 Открыть приложение и пройти регистрацию

2.2 Начать работать со своим окном администратора

# Список литературы

1. Федорова Г.Н. Разработка модулей программного обеспечения для компьютерных систем: учебник для студентов СПО /Г.Н. Федорова. 2-е изд., стер. - М.: Издательский центр «Академия»,2018. - 384с.
2. Федорова Г.Н. Осуществление интеграции программных модулей: учебник для студентов СПО / Г.Н. Федорова. 2-е изд., стер. - М.: Издательский центр «Академия»,2018. - 285 с.
3. Царёв, Р.Ю. Алгоритмы и структуры данных (CDIO): учебник /Р.Ю. Царёв, А.В. Прокопенко; Министерство образования и науки Российской Федерации, Сибирский Федеральный университет. - Красноярск: СФУ, 2016. - 204 с.: ил. - Библиогр. в кн. - ISBN 978-5-7638-3388-1; то же [Электронный ресурс]. - URL: <http://biblioclub.ru/index.php?page=book&id=497016>
4. Гниденко, И. Г. Технология разработки программного обеспечения: учебное пособие для среднего профессионального образования / И. Г. Гниденко, Ф. Ф. Павлов, Д. Ю. Федоров. — Москва: Издательство Юрайт, 2020. — 235 с.
5. C# 7.0. Справочник. Полное описание языка.: Пер. с англ. – СпБ.: ООО “Альфакнига”, 2018. – 1024 с.: ил. – Парал. тит. англ.
6. Microsoft Visual C#. Подробное руководство. 8-е изд. – СПб.: Питер, 2017. – 848 с.: ил. – (Серия «Библиотека программиста»).
7. Программирование на C# для начинающих. Основные сведения / Алексей Васильев. – Москва: Эксмо, 2018. – 592 с. – (Российский компьютерный бестселлер).
8. Язык программирования C# 7 и платформы .NET и .NET Core, 8-е изд.: Пер. с англ. – СПб.: ООО “Диалектика”, 2018 – 1328 с.: ил. – Парал. тит. англ.
9. Оппель, Эндрю Дж. SQL. Полное руководство / Оппель Эндрю Дж. - М.: Диалектика / Вильямс, 2016. - 902 c.