Министерство образования Московской области

Государственное образовательное учреждение высшего образования Московской области

«Государственный гуманитарно-технологический университет»

**Промышленно-экономический колледж**

**КУРСОВОЙ ПРОЕКТ**

«Разработка программных модулей для интернет-магазина ювелирных изделий»

МДК 01.01 «Разработка программных модулей»

**Выполнил:**

Жагрин Александр Денисович\_\_\_\_\_\_\_\_\_\_\_\_

студент группы ИСП.21.1А\_\_\_\_\_\_\_\_\_\_\_\_\_

09.02.07 Информационные системы и программирование)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

очной формы обучения\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Руководитель:**

Кузьмина Елена Евгеньевна­­­­­­­­­­­­­­­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка

\_\_\_\_\_\_\_\_\_/\_\_\_­­­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

Подпись руководителя

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Орехово-Зуево

2024 год

Оглавление

[**Введение** 3](#_Toc162290360)

[**1. Назначение разработки** 5](#_Toc162290361)

[**2.** **Требования к программному модулю.** 7](#_Toc162290362)

[**2.1 Требования к функциональным характеристикам.** 7](#_Toc162290363)

[**2.2 Требование к составу и параметрам технических средств.** 8](#_Toc162290364)

[**2.3 Требования к информационной и программной совместимости.** 8](#_Toc162290365)

[**3. Руководство программиста** 9](#_Toc162290366)

[**3.1 Разработка базы данных** 11](#_Toc162290367)

[**3.2 Разработка программного модуля** 13](#_Toc162290368)

[**3.3** **Текст программы.** 21](#_Toc162290369)

[**4. Руководство пользователя.** 27](#_Toc162290370)

[**4.1** **Выполнение программного модуля.** 27](#_Toc162290371)

[**4.2** **Сообщения пользователю.** 35](#_Toc162290372)

[**5.** **Тестирование и отладка программного модуля.** 39](#_Toc162290373)

[**Заключение.** 40](#_Toc162290374)

[**Список литературы:** 41](#_Toc162290375)

[**Приложение № 1 «Текст программы».** 42](#_Toc162290376)

**Введение**

Основная деятельность интернет магазина ювелирных изделий – это отслеживание финансовой составляющей работы компании, продажа ювелирных изделий.

**Интернет-магазин** — сайт, торгующий товарами посредством сети Интернет. Позволяет пользователям онлайн, в своём браузере или через мобильное приложение, сформировать заказ на покупку, выбрать способ оплаты и доставки заказа, оплатить заказ.

**Ювелирное изделие** – это изделие из драгоценных металлов с использованием различных видов художественной обработки со вставками из драгоценных, полудрагоценных, поделочных цветных камней и других материалов природного или искусственного происхождения или без них, применяемое в качестве различных украшений, различных утилитарных предметов быта и для декоративных целей.

**Покупатель** - юридические и физические лица, использующие, приобретающие, заказывающие либо имеющие намерение приобрести или заказать товары и услуги.

В приложении компании выставлены на продажу некоторые товары. Каждый из них имеет некоторое название, цену и единицу измерения.

Практичностью данного приложения является:

* Удобство использования - пользователи могут быстро и легко найти нужный товар, узнать его характеристики и оформить заказ.
* Широкий ассортимент - интернет-магазин предлагает большой выбор ювелирных изделий различных стилей и ценовых категорий.
* Безопасность - все транзакции проходят через защищенный сервер, что гарантирует сохранность личных данных и средств покупателей.
* Быстрая доставка - благодаря развитой сети курьеров и пунктов выдачи заказов, покупатели могут получить свой заказ в кратчайшие сроки.
* Возможность возврата и обмена товара - если товар не подошел покупателю, он может вернуть его или обменять на другой в течение 14 дней.
* Программа лояльности - для постоянных клиентов действует система скидок и бонусов.
* Профессиональная помощь - в случае возникновения вопросов, консультанты магазина всегда готовы помочь.
* Безопасность платежей - интернет-магазин использует современные технологии шифрования и защиты данных.
* Удобный поиск - пользователи могут легко найти нужный товар по названию, бренду или характеристикам.
* Регулярные акции и специальные предложения - магазин часто проводит распродажи, на которых можно приобрести товары со значительными скидками.

**1. Назначение разработки**

Разработка производится на основании приказа о курсовом проектировании.

Программный продукт «Ювелирный магазин Aurus» - программа, созданная для взаимодействия с базой данных по предметной области.

Автоматизация продукта «Ювелирный магазин Aurus» - процесс внедрения программно-аппаратных комплексов для облегчения и автоматизации процесса сбора, хранения и редактировании информации о ювелирных изделий.

Информация, хранимая в базе данных информационной системы магазина:

* Информация о заказах.
* Сведения о ювелирных изделий.
* Информация о пользователях.

Информация о пользователях:

* Фамилия.
* Имя.
* Телефон.
* Адрес.
* Способ оплаты.
* Пароль.
* Логин.
* Уровень Доступа.

Сведения о ювелирных изделиях:

* Название.
* Тип.
* Цена.
* Материал.
* Драгоценный кристалл.
* Вес.
* Фото.

Информация о заказах:

* Артикль изделия.
* Название изделия.
* Полная цена позиции.
* Количество.
* Дата заказа.
* Номер пользователя.

Приложение должно выполнять следующие функции:

* Вход в программу при запуске.
* Корректный просмотр информации из базы данных.
* Обеспечение авторизованным и аутентифицированным пользователям функционал по их ролям.
* Удаление, изменение информации в базе данных.
* Сортировка.
* Переходы по страницам.
* Печать итоговых данных.

**2.** **Требования к программному модулю.**

**2.1 Требования к функциональным характеристикам.**

Была поставлена задача разработать приложение для ювелирного магазина «Aurus». Главным условием было разделение прав. Пользователь может просматривать и добавлять товар в корзину, а Администратор должен добавлять, редактировать и удалять товары из каталога.

Таблица 1. «Таблица возможности пользователя и администратора»

|  |  |
| --- | --- |
| Возможности пользователя | Возможности администратора |
| Просматривать каталог | Просматривать каталог |
| Использовать фильтрацию | Использовать фильтрацию |
| Смотреть информацию о товаре | Смотреть информацию о товаре |
| Добавлять товар в корзину | Добавлять товар в корзину |
| Создавать заказ на покупку | Создавать заказ на покупку |
| Сменить тему приложения (светлая, темная) | Сменить тему приложения (светлая, темная) |
|  | Редактировать изделие |
|  | Добавить изделие |
|  | Удалить изделие |

**2.2 Требование к составу и параметрам технических средств.**

Таблица 2. «Рекомендуемые характеристики компьютера»

|  |  |
| --- | --- |
| Процессор | Ryzen 5 3600 |
| Память | 32 Гбайт |
| Разрешение экрана | 1920x1080 |
| Устройства ввода | Клавиатура, мышь |
| Дополнительно | Постоянное подключение к сети, в которой находится база данных |

**2.3 Требования к информационной и программной совместимости.**

Для корректной работы программы необходимо:

1. SQL Server Management Studio (SSMS), с установленной базой данных – это бесплатная систему управления базами данных, которая включает в себя набор инструментов для управления Microsoft SQL Server.
2. Windows 10 – Операционная система для персональных компьютеров и рабочих станций, разработанная корпорацией Microsoft в рамках семейства Windows NT.
3. Microsoft Word 2019 – популярный редактор для эффективной работы с текстом, который содержит все необходимые инструменты для создания и редактирования документов, отвечающих вашим задачам и предпочтениям.
4. Microsoft Excel 2019 – популярный программный продукт Microsoft, который используется для создания и анализа электронных таблиц.
5. Microsoft Visual Studio 2022 – это лучшая интегрированная среда разработки для создания многофункциональных, привлекательных кроссплатформенных приложений для Windows, Mac, Linux, iOS и Android.

**3. Руководство программиста**

При выполнении курсового проекта возникла необходимость выбора базы данных для использования. Рассматривались следующие варианты:

1. MySQL - свободная реляционная система управления базами данных. Разработку и поддержку MySQL осуществляет корпорация Oracle. Продукт распространяется под GNU GPL и коммерческой лицензией.

Плюсы:

* Производительность.
* Кроссплатформенность.
* Бесплатная.
* Встроенные инструменты безопасности.

Минусы:

* Проблемы при работе с большими объёмами данных.
* Несоответствие стандарту SQL.

1. PostgreSQL - свободная объектно-реляционная система управления базами данных. Существует в реализациях для множества UNIX-подобных платформ.

Плюсы:

* Поддержка различных языков программирования.
* Кроссплатформенность.
* Работа с большими объёмами данных.

Минусы:

* Низкая эффективность репликации.

Возможные повреждения таблиц при переносе информации.

1. SQLite - быстрая и легкая встраиваемая однофайловая СУБД на языке C, не требует сервера.

Плюсы:

* Хранение данных в одном файле.
* Не требует сложной настройки.
* Кроссплатформенность.
* Автономность.

Минусы:

* Ограниченная поддержка типов данных.
* Отсутствие хранимых процедур.
* Отсутствие поддержки UNICODE.

1. SQL Server Management Studio (SSMS) - система управления базами данных для Microsoft SQL Server.

Плюсы:

* Интеграция с ОС Windows.
* Производительность и отказоустойчивость.
* Поддержка многопользовательской среды.
* Расширенные функции резервного копирования данных.
* Работа с удалённым подключением.

Минусы:

* Работает только на ОС Windows.

При первичном выборе БД отбросили варианты с MySQL и PostgreSQL из-за отсутствия навыков работы и серьёзных минусов. Между SQLite и SSMS выбрали SQL Server Management Studio из-за ограниченной поддержки типов данных в SQLite.

**3.1 Разработка базы данных**

* Создание Таблиц

Таблица «Customers» со столбцами и типами данных:

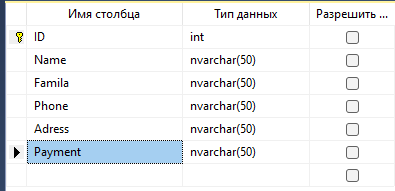


Рисунок 1. «Таблица Customers»

Таблица«Jewellery» со столбцами и типами данных:

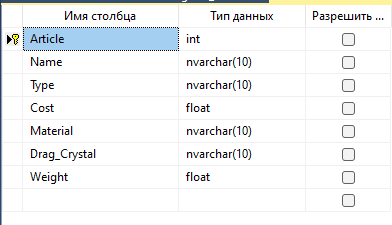
****

Рисунок 2. «Таблица Jewellery»

Таблица«Orders» со столбцами и типами данных:

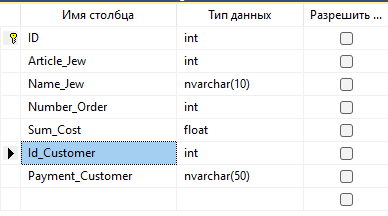
****

Рисунок 3. «Таблица Orders»

* Создание диаграммы базы данных:

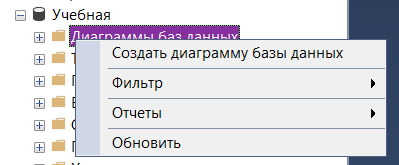
****

Рисунок 4. «Создание диаграммы базы данных»

Диаграмма базы данных со связями:

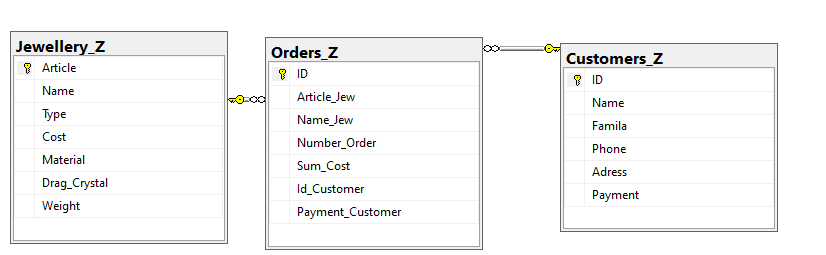
****

Рисунок 5. «Создание таблица»

**3.2 Разработка программного модуля**

Программа была разработана с использованием MS Visual Studio 2022.

Шаги создания программы:

1. Создание проекта:

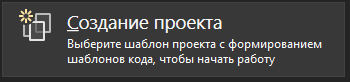


Рисунок 6. «Создание проекта»

1. Выбор типа проекта:

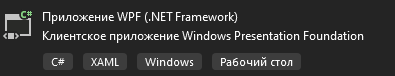


Рисунок 7. «Выбор типа проекта»

1. Подключил базу данных через Модель ADO.NET EDM

Модель данных.

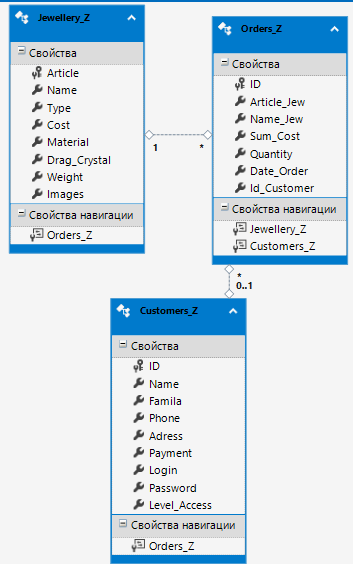
****

Рисунок 8. «Модель данных»

1. Создание класса ClassFrame:

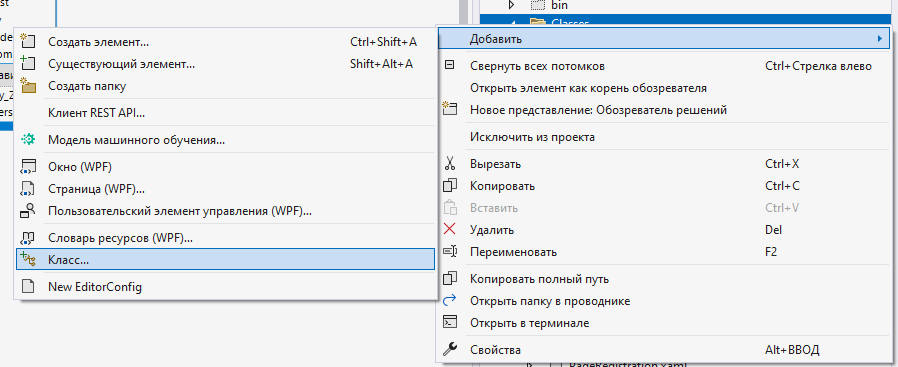


Рисунок 9. «Создание класса ClassFrame»

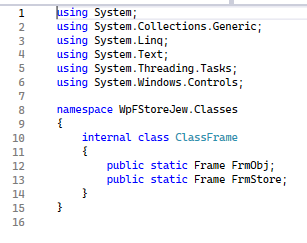


Рисунок 10. «Элементы класса ClassFrame»

1. Создание страницы.

Создал папку Pages.

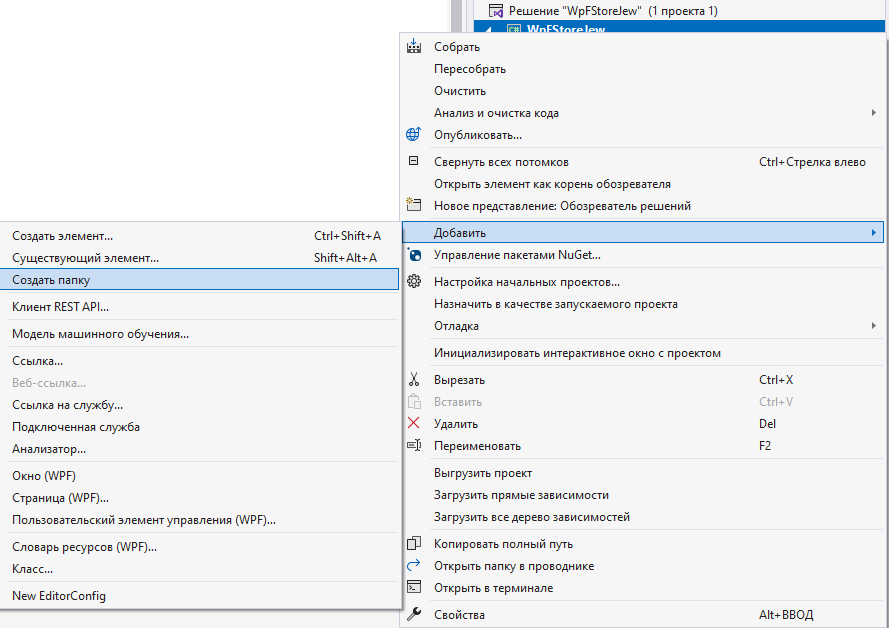


Рисунок 11. «Создание папки Pages»

Создал страницы:

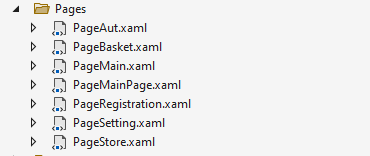


Рисунок 12. «Созданные страницы в папке Pages»

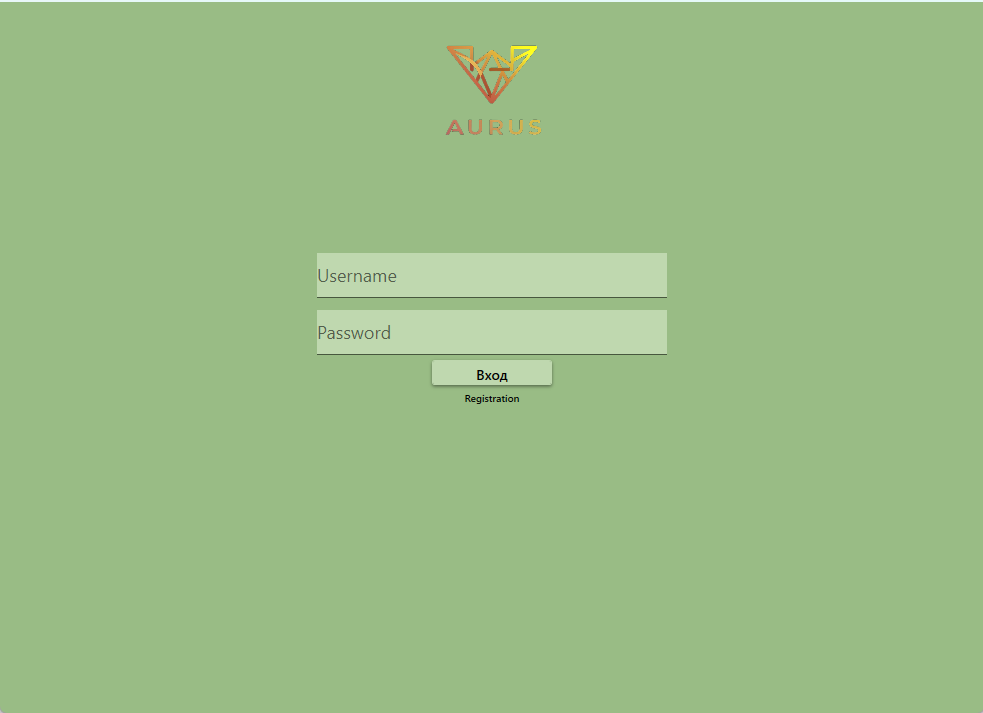


Рисунок 13. «Страница PageAut.xaml»

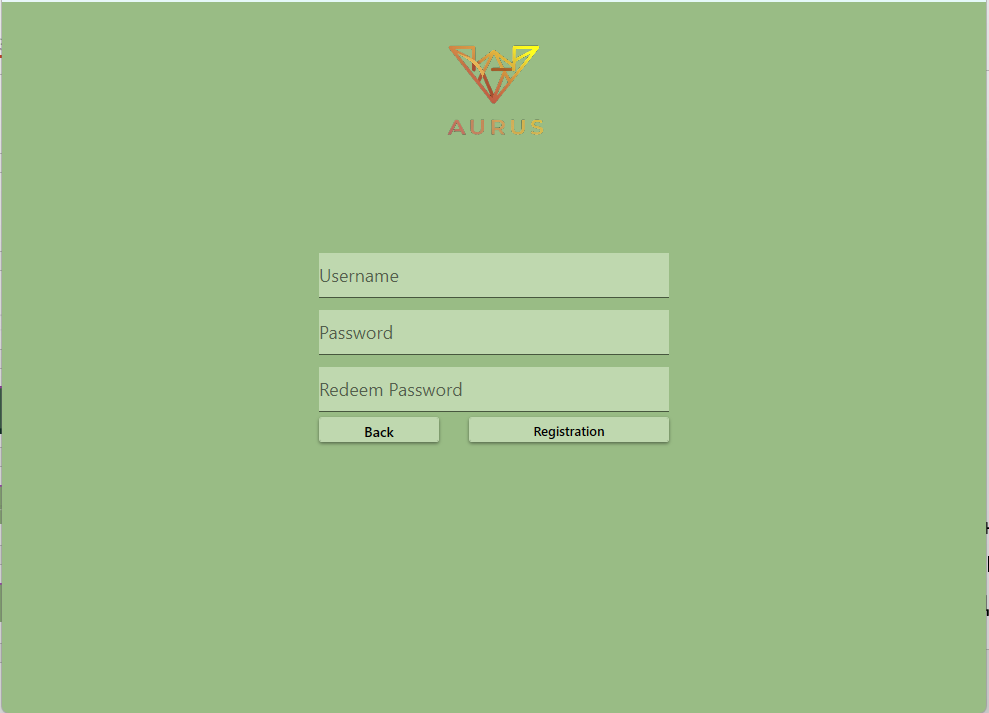


Рисунок 14. «Страница PageRegistration.xaml»



Рисунок 15. «Страница PageMainPage.xaml»

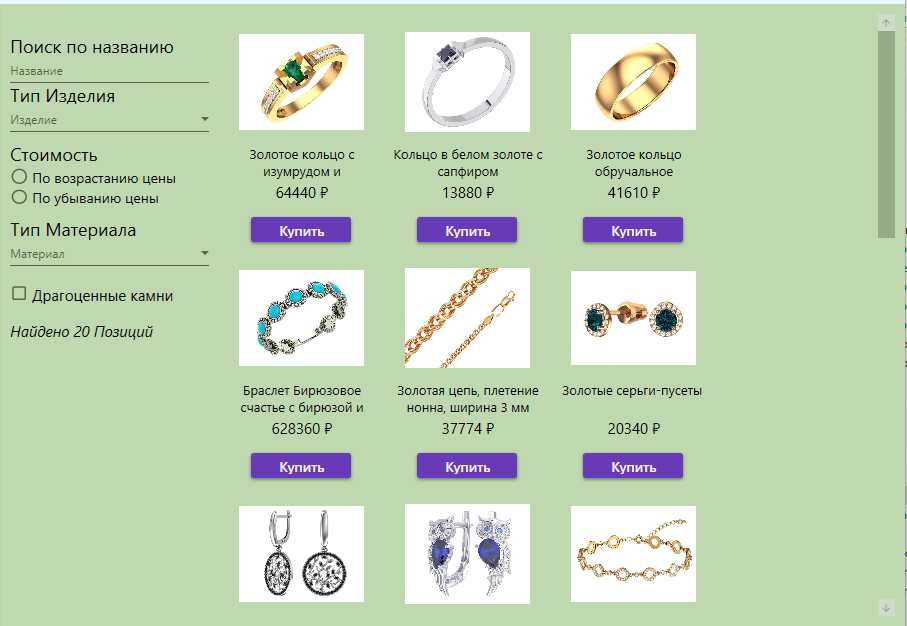


Рисунок 16. «Страница PageStore.xaml»

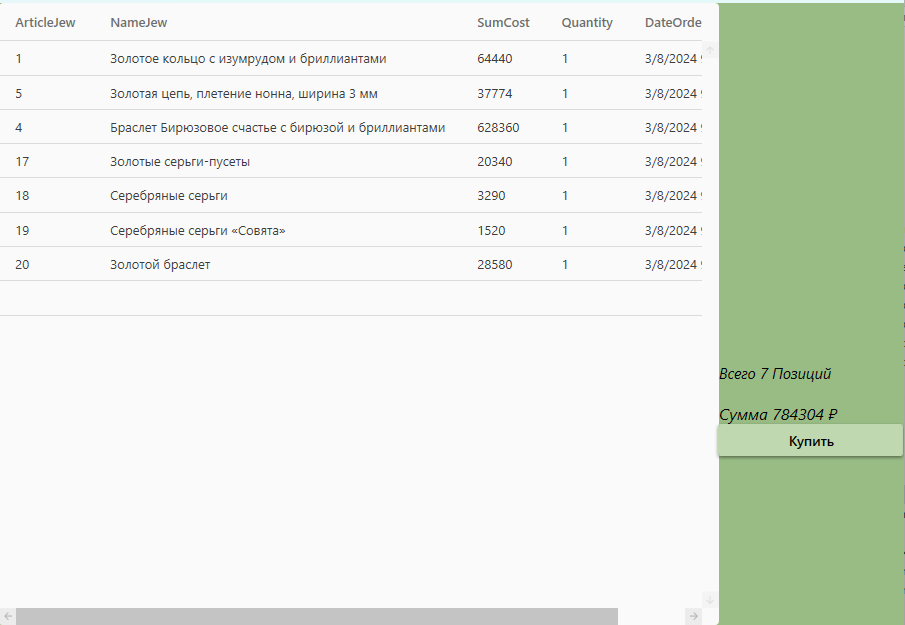


Рисунок 17. «Страница PageBasket.xaml»

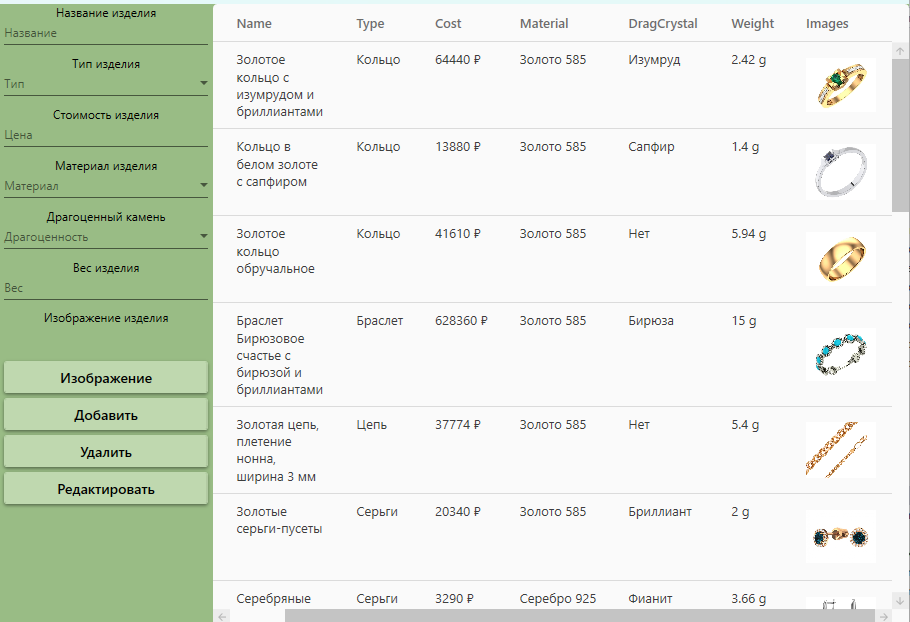


Рисунок 18. «Страница PageSetting.xaml»

**3.3** **Текст программы.**

Страница PageAut.xaml – используется для авторизации в программе:

<Page x:Class="WpFStoreJew.Pages.PageAut"

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:WpFStoreJew.Pages"

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

mc:Ignorable="d"

d:DesignHeight="750" d:DesignWidth="1000"

Title="PageAut"

xmlns:converters="clr-namespace:MaterialDesignThemes.Wpf.Converters;assembly=MaterialDesignThemes.Wpf"

xmlns:wpf="clr-namespace:MaterialDesignThemes.Wpf;assembly=MaterialDesignThemes.Wpf"

xmlns:internal="clr-namespace:MaterialDesignThemes.Wpf.Internal;assembly=MaterialDesignThemes.Wpf">

<Page.Resources>

<Style x:Key="ButtonStyle1" TargetType="{x:Type Button}">

<Setter Property="Background" Value="{DynamicResource SecondCol}"/>

<Setter Property="BorderBrush" Value="{DynamicResource SecondCol}"/>

<Setter Property="BorderThickness" Value="1"/>

<Setter Property="Cursor" Value="Hand"/>

<Setter Property="FocusVisualStyle" Value="{StaticResource FocusVisual}"/>

<Setter Property="Foreground" Value="Black"/>

<Setter Property="Height" Value="32"/>

<Setter Property="HorizontalContentAlignment" Value="Center"/>

<Setter Property="Padding" Value="16,4,16,4"/>

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="{x:Type Button}">

<Grid>

<AdornerDecorator CacheMode="{Binding (wpf:ShadowAssist.CacheMode), RelativeSource={RelativeSource Self}}">

<Grid>

<Border x:Name="border" Background="{TemplateBinding Background}" BorderBrush="{TemplateBinding BorderBrush}" BorderThickness="{TemplateBinding BorderThickness}" CornerRadius="{Binding (wpf:ButtonAssist.CornerRadius), RelativeSource={RelativeSource TemplatedParent}}" Effect="{Binding (wpf:ElevationAssist.Elevation), Converter={x:Static converters:ShadowConverter.Instance}, RelativeSource={RelativeSource TemplatedParent}}"/>

<ProgressBar x:Name="ProgressBar" Background="{Binding (wpf:ButtonProgressAssist.IndicatorBackground), RelativeSource={RelativeSource TemplatedParent}}" BorderBrush="{Binding (wpf:ButtonProgressAssist.IndicatorBackground), RelativeSource={RelativeSource TemplatedParent}}" Foreground="{Binding (wpf:ButtonProgressAssist.IndicatorForeground), RelativeSource={RelativeSource TemplatedParent}}" HorizontalAlignment="Left" Height="{TemplateBinding Height}" IsIndeterminate="{Binding (wpf:ButtonProgressAssist.IsIndeterminate), RelativeSource={RelativeSource TemplatedParent}}" Maximum="{Binding (wpf:ButtonProgressAssist.Maximum), RelativeSource={RelativeSource TemplatedParent}}" Minimum="{Binding (wpf:ButtonProgressAssist.Minimum), RelativeSource={RelativeSource TemplatedParent}}" Opacity="{Binding (wpf:ButtonProgressAssist.Opacity), RelativeSource={RelativeSource TemplatedParent}}" Style="{DynamicResource MaterialDesignLinearProgressBar}" Value="{Binding (wpf:ButtonProgressAssist.Value), RelativeSource={RelativeSource TemplatedParent}}" VerticalAlignment="Center" Visibility="{Binding (wpf:ButtonProgressAssist.IsIndicatorVisible), Converter={StaticResource BooleanToVisibilityConverter}, RelativeSource={RelativeSource TemplatedParent}}" Width="{Binding ActualWidth, RelativeSource={RelativeSource FindAncestor, AncestorType={x:Type ButtonBase}}}">

<ProgressBar.Clip>

<MultiBinding Converter="{StaticResource BorderClipConverter}">

<Binding ElementName="border" Path="ActualWidth"/>

<Binding ElementName="border" Path="ActualHeight"/>

<Binding ElementName="border" Path="CornerRadius"/>

<Binding ElementName="border" Path="BorderThickness"/>

</MultiBinding>

</ProgressBar.Clip>

</ProgressBar>

</Grid>

</AdornerDecorator>

<wpf:Ripple ContentStringFormat="{TemplateBinding ContentStringFormat}" ContentTemplate="{TemplateBinding ContentTemplate}" Content="{TemplateBinding Content}" Focusable="False" HorizontalContentAlignment="{TemplateBinding HorizontalContentAlignment}" Padding="{TemplateBinding Padding}" SnapsToDevicePixels="{TemplateBinding SnapsToDevicePixels}" VerticalContentAlignment="{TemplateBinding VerticalContentAlignment}">

<wpf:Ripple.Clip>

<MultiBinding Converter="{StaticResource BorderClipConverter}">

<Binding ElementName="border" Path="ActualWidth"/>

<Binding ElementName="border" Path="ActualHeight"/>

<Binding ElementName="border" Path="CornerRadius"/>

<Binding ElementName="border" Path="BorderThickness"/>

</MultiBinding>

</wpf:Ripple.Clip>

</wpf:Ripple>

</Grid>

<ControlTemplate.Triggers>

<Trigger Property="IsMouseOver" Value="true">

<Setter Property="wpf:ShadowAssist.Darken" TargetName="border" Value="True"/>

</Trigger>

<Trigger Property="IsKeyboardFocused" Value="true">

<Setter Property="wpf:ShadowAssist.Darken" TargetName="border" Value="True"/>

</Trigger>

<Trigger Property="IsEnabled" Value="false">

<Setter Property="Opacity" Value="0.38"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

<Setter Property="TextBlock.FontSize" Value="14"/>

<Setter Property="TextBlock.FontWeight" Value="Medium"/>

<Setter Property="VerticalContentAlignment" Value="Center"/>

<Setter Property="wpf:ButtonAssist.CornerRadius" Value="2"/>

<Setter Property="wpf:ButtonProgressAssist.IndicatorBackground" Value="{DynamicResource SecondCol}"/>

<Setter Property="wpf:ButtonProgressAssist.IndicatorForeground" Value="{DynamicResource PrimaryHueMidForegroundBrush}"/>

<Setter Property="wpf:ButtonProgressAssist.IsIndicatorVisible" Value="False"/>

<Setter Property="wpf:ButtonProgressAssist.Opacity" Value=".4"/>

<Setter Property="wpf:ElevationAssist.Elevation" Value="Dp2"/>

<Setter Property="wpf:RippleAssist.Feedback" Value="White"/>

</Style>

<Style x:Key="ButtonStyle2" TargetType="{x:Type Button}">

<Setter Property="Background" Value="{DynamicResource PrimaryHueMidBrush}"/>

<Setter Property="BorderBrush" Value="{DynamicResource PrimaryHueMidBrush}"/>

<Setter Property="BorderThickness" Value="1"/>

<Setter Property="Cursor" Value="Hand"/>

<Setter Property="FocusVisualStyle" Value="{StaticResource FocusVisual}"/>

<Setter Property="Foreground" Value="{DynamicResource Text}"/>

<Setter Property="Height" Value="32"/>

<Setter Property="HorizontalContentAlignment" Value="Center"/>

<Setter Property="Padding" Value="16,4,16,4"/>

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="{x:Type Button}">

<Grid>

<AdornerDecorator CacheMode="{Binding (wpf:ShadowAssist.CacheMode), RelativeSource={RelativeSource Self}}">

<Grid>

<Border x:Name="border" Background="{TemplateBinding Background}" BorderBrush="{TemplateBinding BorderBrush}" BorderThickness="{TemplateBinding BorderThickness}" CornerRadius="{Binding (wpf:ButtonAssist.CornerRadius), RelativeSource={RelativeSource TemplatedParent}}" Effect="{Binding (wpf:ElevationAssist.Elevation), Converter={x:Static converters:ShadowConverter.Instance}, RelativeSource={RelativeSource TemplatedParent}}"/>

<ProgressBar x:Name="ProgressBar" Background="{Binding (wpf:ButtonProgressAssist.IndicatorBackground), RelativeSource={RelativeSource TemplatedParent}}" BorderBrush="{Binding (wpf:ButtonProgressAssist.IndicatorBackground), RelativeSource={RelativeSource TemplatedParent}}" Foreground="{Binding (wpf:ButtonProgressAssist.IndicatorForeground), RelativeSource={RelativeSource TemplatedParent}}" HorizontalAlignment="Left" Height="{TemplateBinding Height}" IsIndeterminate="{Binding (wpf:ButtonProgressAssist.IsIndeterminate), RelativeSource={RelativeSource TemplatedParent}}" Maximum="{Binding (wpf:ButtonProgressAssist.Maximum), RelativeSource={RelativeSource TemplatedParent}}" Minimum="{Binding (wpf:ButtonProgressAssist.Minimum), RelativeSource={RelativeSource TemplatedParent}}" Opacity="{Binding (wpf:ButtonProgressAssist.Opacity), RelativeSource={RelativeSource TemplatedParent}}" Style="{DynamicResource MaterialDesignLinearProgressBar}" Value="{Binding (wpf:ButtonProgressAssist.Value), RelativeSource={RelativeSource TemplatedParent}}" VerticalAlignment="Center" Visibility="{Binding (wpf:ButtonProgressAssist.IsIndicatorVisible), Converter={StaticResource BooleanToVisibilityConverter}, RelativeSource={RelativeSource TemplatedParent}}" Width="{Binding ActualWidth, RelativeSource={RelativeSource FindAncestor, AncestorType={x:Type ButtonBase}}}">

<ProgressBar.Clip>

<MultiBinding Converter="{StaticResource BorderClipConverter}">

<Binding ElementName="border" Path="ActualWidth"/>

<Binding ElementName="border" Path="ActualHeight"/>

<Binding ElementName="border" Path="CornerRadius"/>

<Binding ElementName="border" Path="BorderThickness"/>

</MultiBinding>

</ProgressBar.Clip>

</ProgressBar>

</Grid>

</AdornerDecorator>

<wpf:Ripple ContentStringFormat="{TemplateBinding ContentStringFormat}" ContentTemplate="{TemplateBinding ContentTemplate}" Content="{TemplateBinding Content}" Focusable="False" HorizontalContentAlignment="{TemplateBinding HorizontalContentAlignment}" Padding="{TemplateBinding Padding}" SnapsToDevicePixels="{TemplateBinding SnapsToDevicePixels}" VerticalContentAlignment="{TemplateBinding VerticalContentAlignment}">

<wpf:Ripple.Clip>

<MultiBinding Converter="{StaticResource BorderClipConverter}">

<Binding ElementName="border" Path="ActualWidth"/>

<Binding ElementName="border" Path="ActualHeight"/>

<Binding ElementName="border" Path="CornerRadius"/>

<Binding ElementName="border" Path="BorderThickness"/>

</MultiBinding>

</wpf:Ripple.Clip>

</wpf:Ripple>

</Grid>

<ControlTemplate.Triggers>

<Trigger Property="IsMouseOver" Value="true">

<Setter Property="wpf:ShadowAssist.Darken" TargetName="border" Value="True"/>

</Trigger>

<Trigger Property="IsKeyboardFocused" Value="true">

<Setter Property="wpf:ShadowAssist.Darken" TargetName="border" Value="True"/>

</Trigger>

<Trigger Property="IsEnabled" Value="false">

<Setter Property="Opacity" Value="0.38"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

<Setter Property="TextBlock.FontSize" Value="14"/>

<Setter Property="TextBlock.FontWeight" Value="Medium"/>

<Setter Property="VerticalContentAlignment" Value="Center"/>

<Setter Property="wpf:ButtonAssist.CornerRadius" Value="2"/>

<Setter Property="wpf:ButtonProgressAssist.IndicatorBackground" Value="{DynamicResource PrimaryHueMidBrush}"/>

<Setter Property="wpf:ButtonProgressAssist.IndicatorForeground" Value="{DynamicResource PrimaryHueMidForegroundBrush}"/>

<Setter Property="wpf:ButtonProgressAssist.IsIndicatorVisible" Value="False"/>

<Setter Property="wpf:ButtonProgressAssist.Opacity" Value=".4"/>

<Setter Property="wpf:ElevationAssist.Elevation" Value="Dp2"/>

<Setter Property="wpf:RippleAssist.Feedback" Value="White"/>

<Style.Triggers>

<MultiTrigger>

<MultiTrigger.Conditions>

<Condition Property="IsMouseOver" Value="true"/>

</MultiTrigger.Conditions>

<MultiTrigger.Setters>

<Setter Property="Foreground" Value="Gray"/>

</MultiTrigger.Setters>

</MultiTrigger>

</Style.Triggers>

</Style>

</Page.Resources>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="1\*"/>

<ColumnDefinition Width="5\*"/>

<ColumnDefinition Width="1\*"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="2\*"/>

<RowDefinition Height="5\*"/>

<RowDefinition Height="1\*"/>

</Grid.RowDefinitions>

<Image Source="/Resources/Logo.png" HorizontalAlignment="Center" VerticalAlignment="Center"

Grid.Column="1" Stretch="UniformToFill" Width="200" Margin="0,20,0,0"/>

<StackPanel Grid.Row="1" Grid.Column="1" VerticalAlignment="Center" Height="210">

<TextBox x:Name="Txtbox1"

materialDesign:HintAssist.Hint="Username"

Margin="6"

Background="{DynamicResource SecondCol}"

Width="350"

Height="45"

FontSize="18"

HorizontalContentAlignment="Left"

VerticalContentAlignment="Center"

Foreground="{DynamicResource Text}"/>

<PasswordBox x:Name="Psbox1" Tag="Password"

materialDesign:HintAssist.Hint="Password"

Margin="6"

Background="{DynamicResource SecondCol}"

Width="350"

Height="45"

FontSize="18"

HorizontalContentAlignment="Left"

VerticalContentAlignment="Center"

Foreground="{DynamicResource Text}"/>

<Button Style="{DynamicResource ButtonStyle1}" x:Name="Btn1" Height="25" Width="120" Content="Вход"

Click="Btn1\_Click"

Foreground="{DynamicResource Text}"/>

<Button Style="{DynamicResource ButtonStyle2}" Background="Transparent" Content="Registration" x:Name="BtnReg"

FontSize="10"

Width="100" Height="23" Click="BtnReg\_Click"

BorderBrush="Transparent"/>

</StackPanel>

</Grid>

</Page>

Cs код:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using WpFStoreJew.Classes;

namespace WpFStoreJew.Pages

{

/// <summary>

/// Логика взаимодействия для PageAut.xaml

/// </summary>

public partial class PageAut : Page

{

public PageAut()

{

InitializeComponent();

}

private void Btn1\_Click(object sender, RoutedEventArgs e)

{

String Login = Txtbox1.Text;

String Password = Psbox1.Password;

if (Login != "" && Password != "")

{

var Level = StoreJewEntities.GetContext().Customers\_Z.FirstOrDefault(x => x.Login == Txtbox1.Text && x.Password == Psbox1.Password);

if (Level != null)

{

if (Level.Level\_Access == "Администратор")

{

MessageBox.Show($"Администратор авторизирован!");

ClassFrame.FrmObj.Navigate(new PageMain(Level));

Txtbox1.Text = null;

Psbox1.Password = null;

}

else if (Level.Level\_Access == "Пользователь")

{

MessageBox.Show($"Пользователь авторизирован!");

ClassFrame.FrmObj.Navigate(new PageMain(Level));

Txtbox1.Text = null;

Psbox1.Password = null;

}

}

else MessageBox.Show("Данный пользователь не существует");

}

else

{

MessageBox.Show("Поля не должны быть пустыми");

}

}

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

{

Classes.ClassFrame.FrmObj.Navigate(new PageRegistration());

}

}

}

Страница PageRegistration – используется для регистрации пользователя (см. Приложение №1 «Текст приложения»).

Страница PageMain – используется для навигации по приложению (см. Приложение №1 «Текст приложения»).

Страница PageMainPage – используется для вывода информации о компании (см. Приложение №1 «Текст приложения»).

Страница PageStore– используется для просмотра каталога магазина и добавления в корзину (см. Приложение №1 «Текст приложения»).

Страница PageBasket – используется для отображения списка покупок (см. Приложение №1 «Текст приложения»).

Страница PageSetting – используется для добавления, редактирования и удаления товара из каталога (см. Приложение №1 «Текст приложения»).

**4.** **Руководство пользователя.**

**4.1** **Выполнение программного модуля.**

Для запуска программы следует запустить WpfStoreJew.exe

Существует 2 роли:

* Администратор – главный пользователь системы, обладающий всеми полномочиями программы;
* Пользователь – потенциальный покупатель, у которого есть только возможности смотреть и покупать товар.

Страница авторизации при запуске:

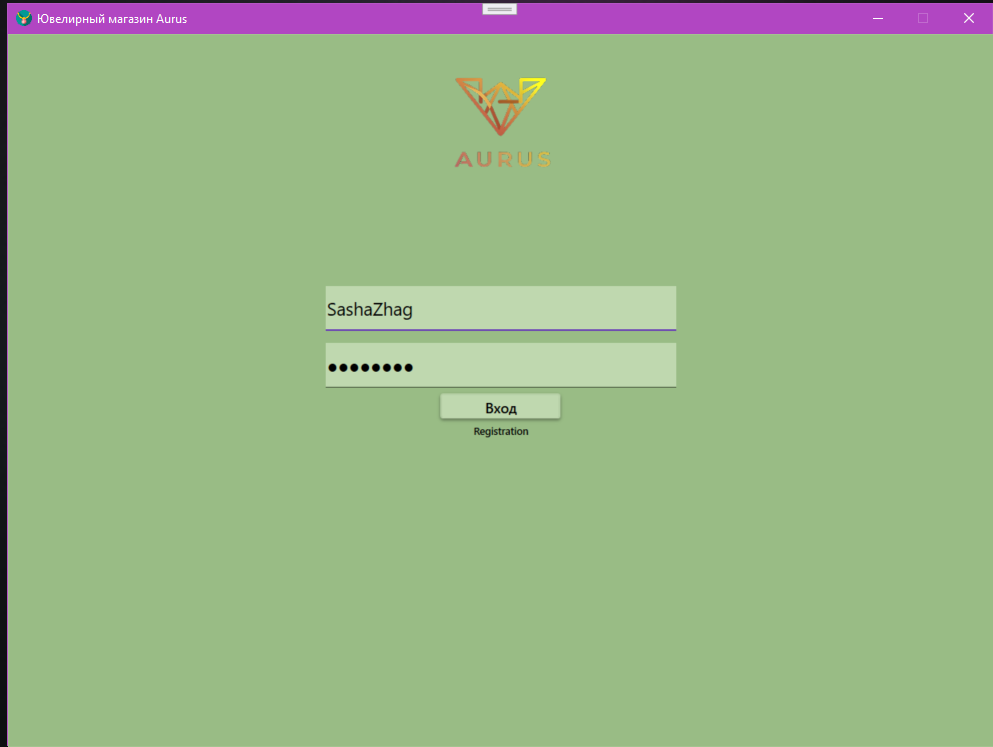


Рисунок 19. «Страница авторизации»

Основная страница от лица администратора после входа:

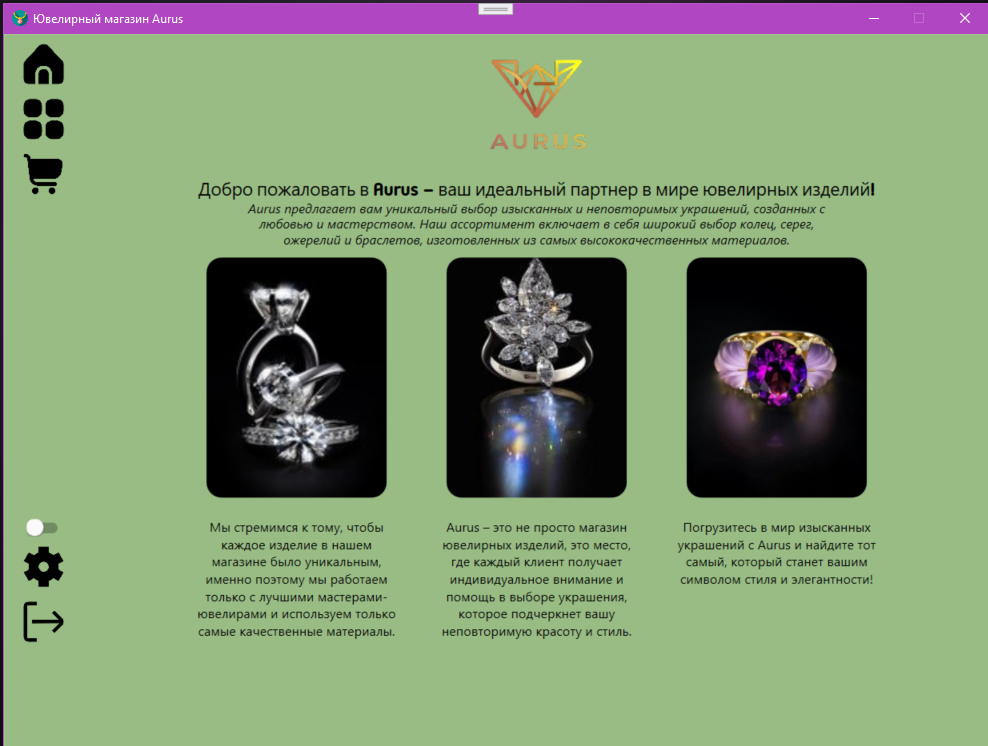


Рисунок 20. «Основная страница от лица администратора»

Страница добавления товара (только у администратора), после перехода на страницу «Редактирования каталога»:

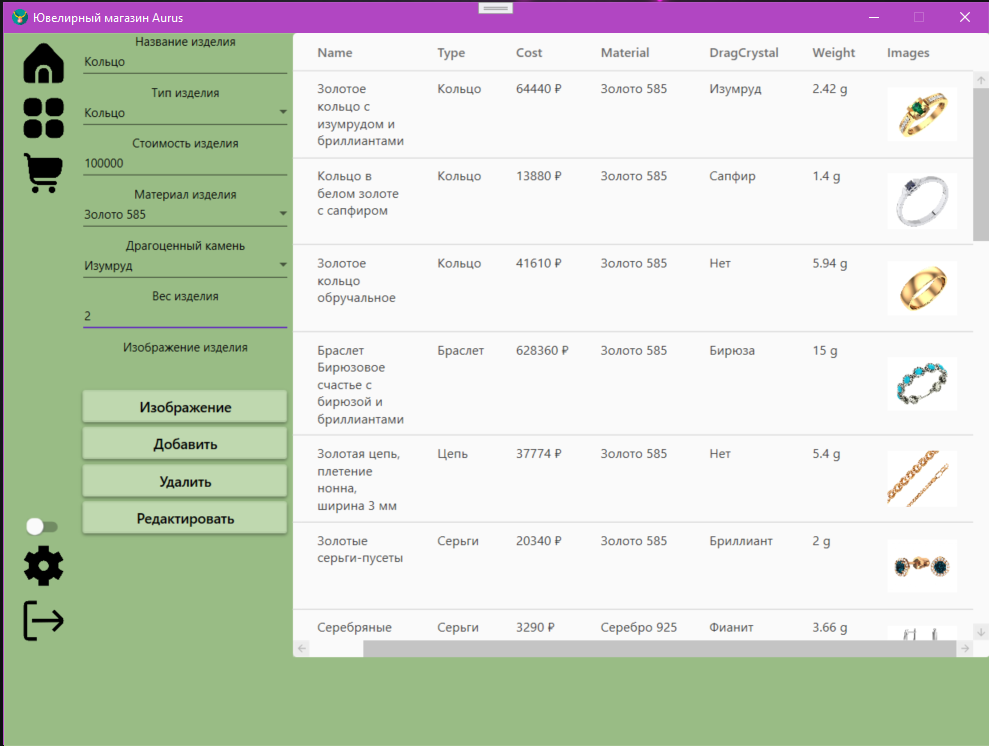


Рисунок 21. «Страница добавления товара (только у администратора)»

Страница просмотра каталога товаров, после перехода на страницу «Каталог»:

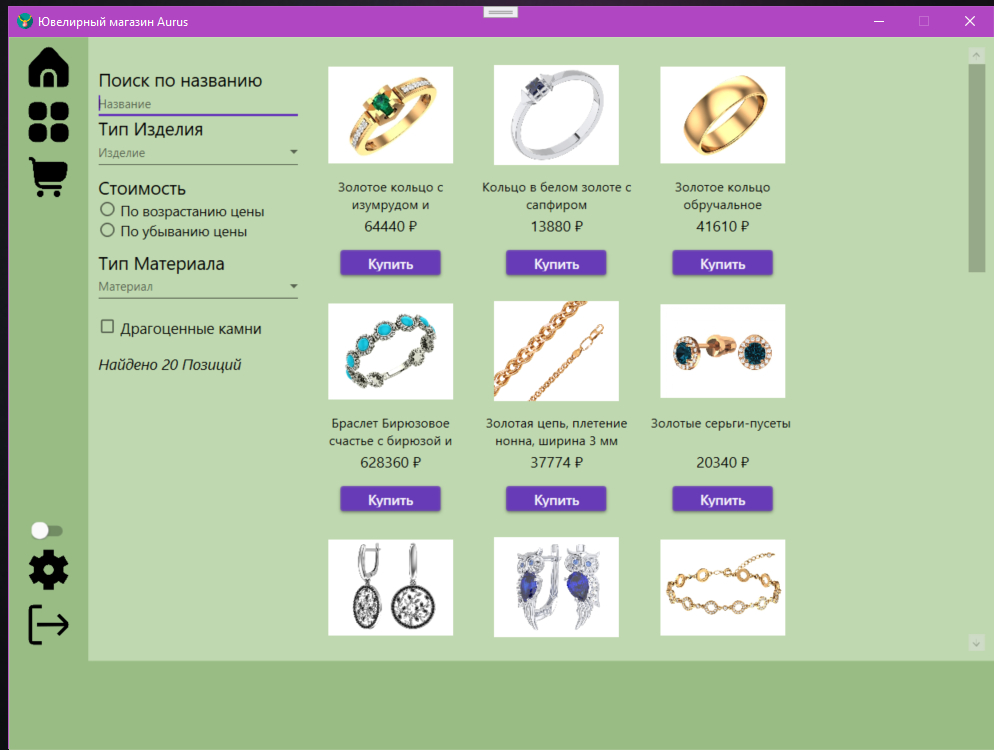


Рисунок 22. «Страница каталога»

Страница корзины, после перехода на страницу «Корзина»:

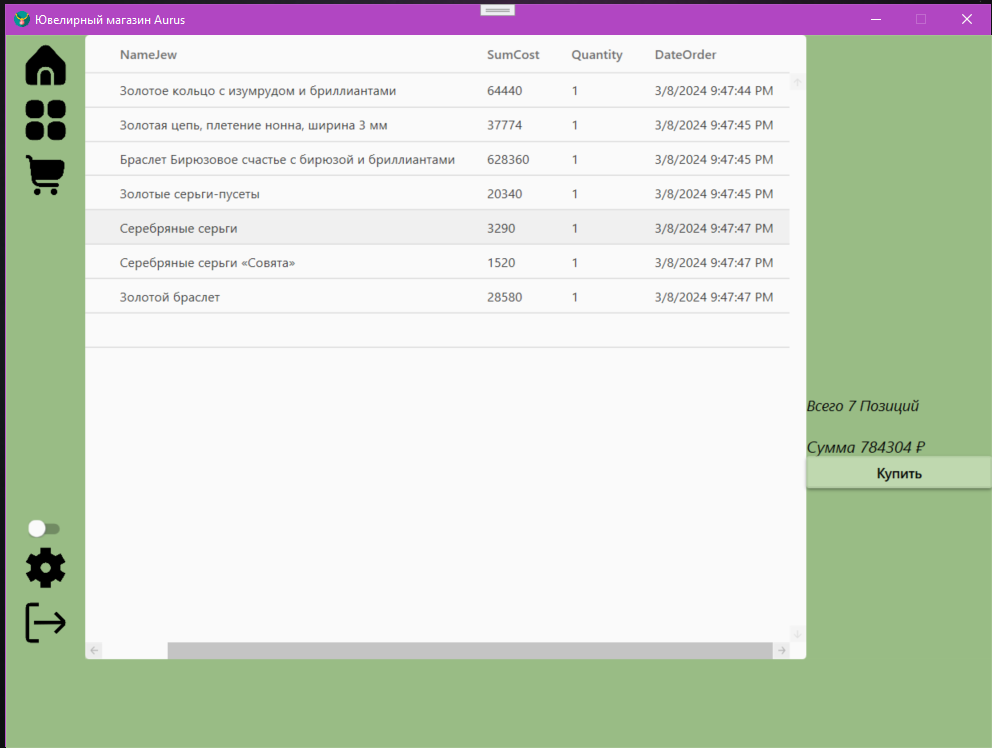


Рисунок 23. «Страница корзины»

Страница регистрации, после нажатия кнопки Registration на странице входа:

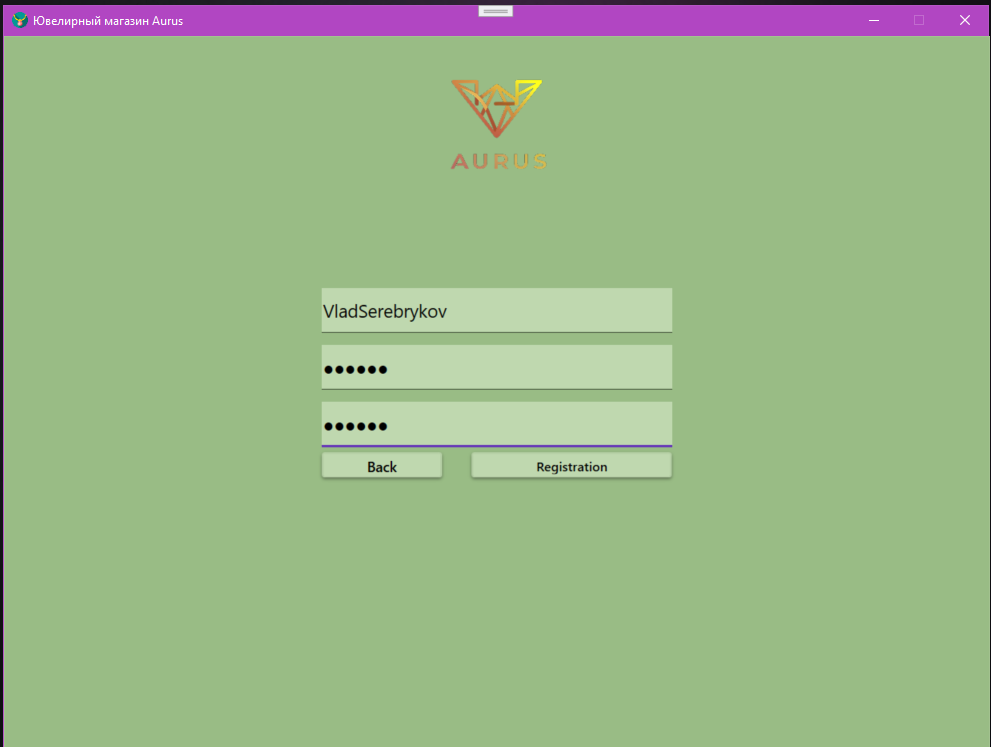


Рисунок 24. «Страница регистрации нового пользователя»

Также присутствует смена визуального оформления: светлая тема, темная тема. Они меняются переключателем:

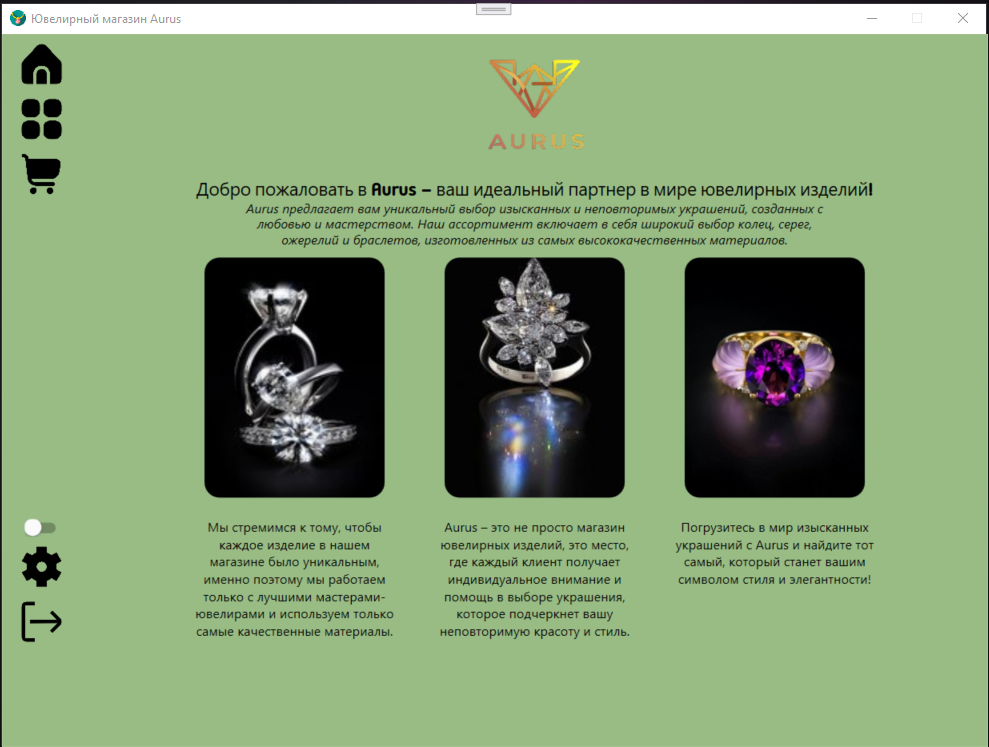


Рисунок 25. «Светлая тема приложения»

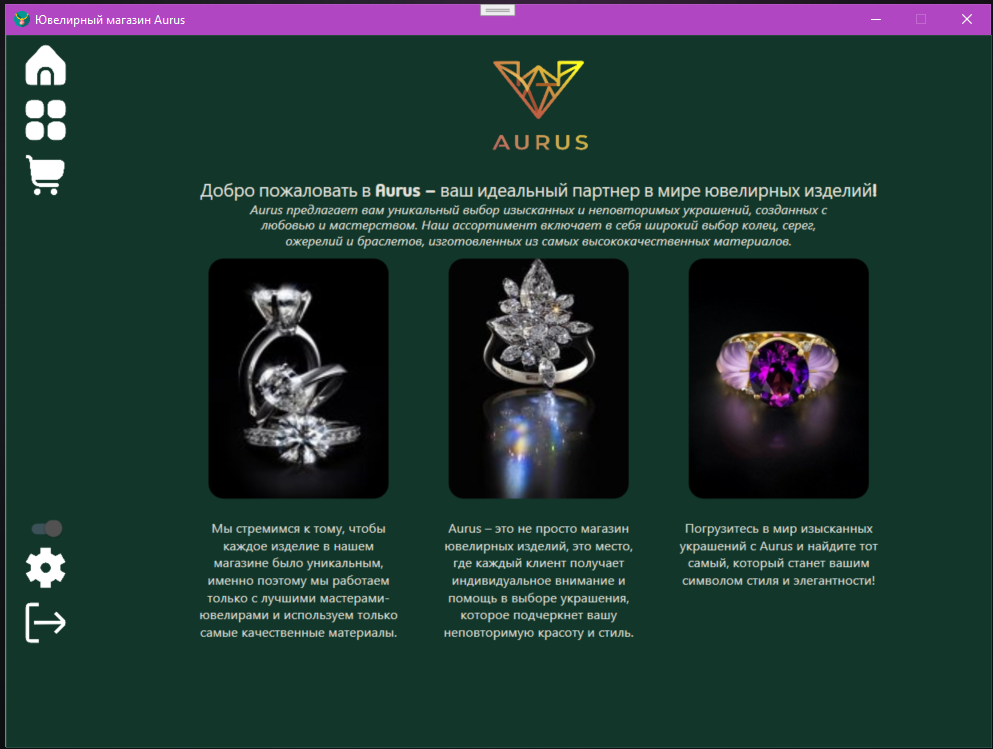


Рисунок 26. «Темная тема приложения»

* 1. **Сообщения пользователю.**
* Сообщение о пустых текстовых полях на странице авторизации.

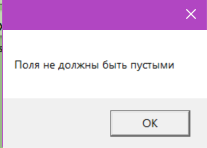


Рисунок 27. «Сообщение о пустых полях»

* Сообщение о неправильном логине/пароле.

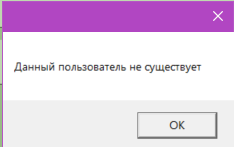


Рисунок 28. «Сообщение о неправильном логине/пароле»

* Сообщение об успешной авторизации пользователя

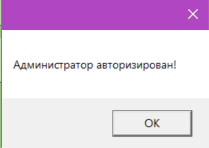


Рисунок 29. «Сообщение об успешной авторизации пользователя»

* Сообщение о пустых текстовых полях на странице добавления записи о приёме.

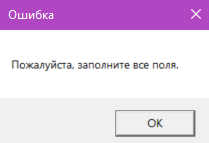


Рисунок 30. «Сообщение о пустых полях при добавлении товара»

* Сообщение об успешном добавлении товара в каталог.

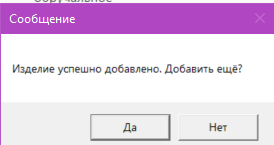


Рисунок 31. «Сообщение о успешном добавлении товара в каталог»

* Сообщение об удалении товара из каталога.

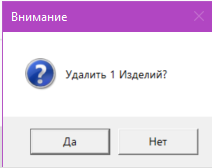


Рисунок 32. «Сообщение об удалении товара из каталога»

* Сообщение об успешном удалении товара из каталога.

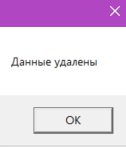


Рисунок 33. «Сообщение о успешном удалении товара из каталога»

* Сообщение о не выбранном товара перед редактированием.

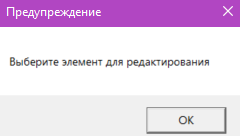


Рисунок 34. «Сообщение о не выбранном товара перед редактированием»

* Сообщение о пустых текстовых полях при редактировании товара в каталоге.

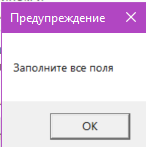


Рисунок 35. «Сообщение о пустых полях при редактировании»

* Сообщение об успешном редактировании товара в каталоге.

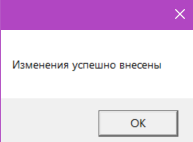


Рисунок 36. «Сообщение об успешном редактировании товара в каталоге»

* Сообщение о минимальной и максимальной длине пароля и логина.

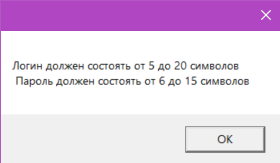


Рисунок 37. «Сообщение о минимальной и максимальной длине пароля и логина»

1. **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).

Ошибки:

1.

Как решил:

Было создано условие:

if (JewForRemoving.Count() != 0)

{

(код программы)

}

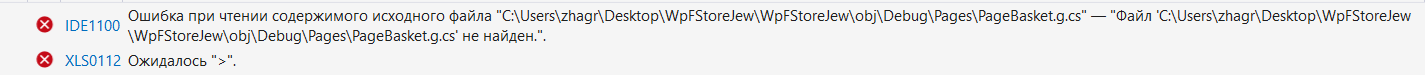
else

{

MessageBox.Show("Вы не выбрали кого удалить");

}

2.



Как решил:

Было импортировано пространство имён:

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

**Заключение.**

Для информационной системы «Ювелирный магазин» был разработан программный модуль.

В рамках разработки программного модуля для магазина ювелирных изделий был создан функциональный инструмент, предназначенный для эффективного управления информацией о товарах. Программный модуль позволяет осуществлять просмотр, добавление, удаление и редактирование данных о продукции, а также осуществлять поиск и сортировку по различным параметрам.

Основными целями разработки были обеспечение удобства использования, надежности и эффективности при работе с данными о товарах.

В дальнейшем возможно расширение функционала программного модуля, включая доработку функции редактирования личных данных сотрудника, что позволит улучшить пользовательский опыт и функциональные возможности программы.

**Список литературы:**

1. Гриффитс Иэн. Программируем на C# 8.0. Разработка приложений. - СПб.: Питер, 2021.
2. Арораа Гаурав, Чилберто Джеффри. Паттерны проектирования для C# и платформы .NET Core. - СПб.: Питер, 2021.
3. Лейтман М.У. Диагностическая офтальмология. Практическое руководство. - М.: ГЭОТАР-Медиа, 2023.
4. Клири Стивен. Конкурентность в C#. Асинхронное, параллельное и многопоточное программирование. Издательский дом «Питер», 2020.
5. "Современное ювелирное искусство: техники и материалы" / Мэттью Борн. - Издательство "Топкоттон", 2021.
6. "Искусство ювелирного дела: современные тенденции и технологии" / Хелен Дрисколл. - Издательство "АртПресс", 2020.
7. "Ювелирное искусство XXI века: тенденции и перспективы" / Под ред. Натальи Кузнецовой. - Издательство "АртМастер", 2020.
8. "Ювелирное искусство сегодня: новые технологии и идеи" / Дэвид Хэйс. - Издательство "АртЛайн", 2019.
9. "Современные тенденции в дизайне ювелирных изделий" / Под ред. Елены Смирновой. - Издательство "АртБук", 2019.

**Приложение № 1 «Текст программы».**

Код страницы «MainWindow»

Xaml код:

<Window x:Class="WpFStoreJew.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:WpFStoreJew"

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

mc:Ignorable="d"

Title="Ювелирный магазин Aurus" Height="750" Width="1000" ResizeMode="CanMinimize" WindowStartupLocation="CenterScreen"

>

<Grid Background="{DynamicResource Backdrop}">

<Grid.RowDefinitions>

<RowDefinition Height="7\*"/>

<RowDefinition Height="1\*"/>

</Grid.RowDefinitions>

<Frame Grid.Row="0" Grid.Column="1"

Name="FrmMain" NavigationUIVisibility="Hidden"/>

</Grid>

</Window>

Cs код:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using WpFStoreJew.Classes;

namespace WpFStoreJew

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

AppConnect.StoreJewEntities = new StoreJewEntities();

Classes.ClassFrame.FrmObj = FrmMain;

Classes.ClassFrame.FrmObj.Navigate(new Pages.PageAut());

}

}

}

Код страницы «PageAut » (см. рисунок 13)

<Page x:Class="WpFStoreJew.Pages.PageAut"

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:WpFStoreJew.Pages"

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

mc:Ignorable="d"

d:DesignHeight="750" d:DesignWidth="1000"

Title="PageAut"

xmlns:converters="clr-namespace:MaterialDesignThemes.Wpf.Converters;assembly=MaterialDesignThemes.Wpf"

xmlns:wpf="clr-namespace:MaterialDesignThemes.Wpf;assembly=MaterialDesignThemes.Wpf"

xmlns:internal="clr-namespace:MaterialDesignThemes.Wpf.Internal;assembly=MaterialDesignThemes.Wpf">

<Page.Resources>

<Style x:Key="ButtonStyle1" TargetType="{x:Type Button}">

<Setter Property="Background" Value="{DynamicResource SecondCol}"/>

<Setter Property="BorderBrush" Value="{DynamicResource SecondCol}"/>

<Setter Property="BorderThickness" Value="1"/>

<Setter Property="Cursor" Value="Hand"/>

<Setter Property="FocusVisualStyle" Value="{StaticResource FocusVisual}"/>

<Setter Property="Foreground" Value="Black"/>

<Setter Property="Height" Value="32"/>

<Setter Property="HorizontalContentAlignment" Value="Center"/>

<Setter Property="Padding" Value="16,4,16,4"/>

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="{x:Type Button}">

<Grid>

<AdornerDecorator CacheMode="{Binding (wpf:ShadowAssist.CacheMode), RelativeSource={RelativeSource Self}}">

<Grid>

<Border x:Name="border" Background="{TemplateBinding Background}" BorderBrush="{TemplateBinding BorderBrush}" BorderThickness="{TemplateBinding BorderThickness}" CornerRadius="{Binding (wpf:ButtonAssist.CornerRadius), RelativeSource={RelativeSource TemplatedParent}}" Effect="{Binding (wpf:ElevationAssist.Elevation), Converter={x:Static converters:ShadowConverter.Instance}, RelativeSource={RelativeSource TemplatedParent}}"/>

<ProgressBar x:Name="ProgressBar" Background="{Binding (wpf:ButtonProgressAssist.IndicatorBackground), RelativeSource={RelativeSource TemplatedParent}}" BorderBrush="{Binding (wpf:ButtonProgressAssist.IndicatorBackground), RelativeSource={RelativeSource TemplatedParent}}" Foreground="{Binding (wpf:ButtonProgressAssist.IndicatorForeground), RelativeSource={RelativeSource TemplatedParent}}" HorizontalAlignment="Left" Height="{TemplateBinding Height}" IsIndeterminate="{Binding (wpf:ButtonProgressAssist.IsIndeterminate), RelativeSource={RelativeSource TemplatedParent}}" Maximum="{Binding (wpf:ButtonProgressAssist.Maximum), RelativeSource={RelativeSource TemplatedParent}}" Minimum="{Binding (wpf:ButtonProgressAssist.Minimum), RelativeSource={RelativeSource TemplatedParent}}" Opacity="{Binding (wpf:ButtonProgressAssist.Opacity), RelativeSource={RelativeSource TemplatedParent}}" Style="{DynamicResource MaterialDesignLinearProgressBar}" Value="{Binding (wpf:ButtonProgressAssist.Value), RelativeSource={RelativeSource TemplatedParent}}" VerticalAlignment="Center" Visibility="{Binding (wpf:ButtonProgressAssist.IsIndicatorVisible), Converter={StaticResource BooleanToVisibilityConverter}, RelativeSource={RelativeSource TemplatedParent}}" Width="{Binding ActualWidth, RelativeSource={RelativeSource FindAncestor, AncestorType={x:Type ButtonBase}}}">

<ProgressBar.Clip>

<MultiBinding Converter="{StaticResource BorderClipConverter}">

<Binding ElementName="border" Path="ActualWidth"/>

<Binding ElementName="border" Path="ActualHeight"/>

<Binding ElementName="border" Path="CornerRadius"/>

<Binding ElementName="border" Path="BorderThickness"/>

</MultiBinding>

</ProgressBar.Clip>

</ProgressBar>

</Grid>

</AdornerDecorator>

<wpf:Ripple ContentStringFormat="{TemplateBinding ContentStringFormat}" ContentTemplate="{TemplateBinding ContentTemplate}" Content="{TemplateBinding Content}" Focusable="False" HorizontalContentAlignment="{TemplateBinding HorizontalContentAlignment}" Padding="{TemplateBinding Padding}" SnapsToDevicePixels="{TemplateBinding SnapsToDevicePixels}" VerticalContentAlignment="{TemplateBinding VerticalContentAlignment}">

<wpf:Ripple.Clip>

<MultiBinding Converter="{StaticResource BorderClipConverter}">

<Binding ElementName="border" Path="ActualWidth"/>

<Binding ElementName="border" Path="ActualHeight"/>

<Binding ElementName="border" Path="CornerRadius"/>

<Binding ElementName="border" Path="BorderThickness"/>

</MultiBinding>

</wpf:Ripple.Clip>

</wpf:Ripple>

</Grid>

<ControlTemplate.Triggers>

<Trigger Property="IsMouseOver" Value="true">

<Setter Property="wpf:ShadowAssist.Darken" TargetName="border" Value="True"/>

</Trigger>

<Trigger Property="IsKeyboardFocused" Value="true">

<Setter Property="wpf:ShadowAssist.Darken" TargetName="border" Value="True"/>

</Trigger>

<Trigger Property="IsEnabled" Value="false">

<Setter Property="Opacity" Value="0.38"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

<Setter Property="TextBlock.FontSize" Value="14"/>

<Setter Property="TextBlock.FontWeight" Value="Medium"/>

<Setter Property="VerticalContentAlignment" Value="Center"/>

<Setter Property="wpf:ButtonAssist.CornerRadius" Value="2"/>

<Setter Property="wpf:ButtonProgressAssist.IndicatorBackground" Value="{DynamicResource SecondCol}"/>

<Setter Property="wpf:ButtonProgressAssist.IndicatorForeground" Value="{DynamicResource PrimaryHueMidForegroundBrush}"/>

<Setter Property="wpf:ButtonProgressAssist.IsIndicatorVisible" Value="False"/>

<Setter Property="wpf:ButtonProgressAssist.Opacity" Value=".4"/>

<Setter Property="wpf:ElevationAssist.Elevation" Value="Dp2"/>

<Setter Property="wpf:RippleAssist.Feedback" Value="White"/>

</Style>

<Style x:Key="ButtonStyle2" TargetType="{x:Type Button}">

<Setter Property="Background" Value="{DynamicResource PrimaryHueMidBrush}"/>

<Setter Property="BorderBrush" Value="{DynamicResource PrimaryHueMidBrush}"/>

<Setter Property="BorderThickness" Value="1"/>

<Setter Property="Cursor" Value="Hand"/>

<Setter Property="FocusVisualStyle" Value="{StaticResource FocusVisual}"/>

<Setter Property="Foreground" Value="{DynamicResource Text}"/>

<Setter Property="Height" Value="32"/>

<Setter Property="HorizontalContentAlignment" Value="Center"/>

<Setter Property="Padding" Value="16,4,16,4"/>

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="{x:Type Button}">

<Grid>

<AdornerDecorator CacheMode="{Binding (wpf:ShadowAssist.CacheMode), RelativeSource={RelativeSource Self}}">

<Grid>

<Border x:Name="border" Background="{TemplateBinding Background}" BorderBrush="{TemplateBinding BorderBrush}" BorderThickness="{TemplateBinding BorderThickness}" CornerRadius="{Binding (wpf:ButtonAssist.CornerRadius), RelativeSource={RelativeSource TemplatedParent}}" Effect="{Binding (wpf:ElevationAssist.Elevation), Converter={x:Static converters:ShadowConverter.Instance}, RelativeSource={RelativeSource TemplatedParent}}"/>

<ProgressBar x:Name="ProgressBar" Background="{Binding (wpf:ButtonProgressAssist.IndicatorBackground), RelativeSource={RelativeSource TemplatedParent}}" BorderBrush="{Binding (wpf:ButtonProgressAssist.IndicatorBackground), RelativeSource={RelativeSource TemplatedParent}}" Foreground="{Binding (wpf:ButtonProgressAssist.IndicatorForeground), RelativeSource={RelativeSource TemplatedParent}}" HorizontalAlignment="Left" Height="{TemplateBinding Height}" IsIndeterminate="{Binding (wpf:ButtonProgressAssist.IsIndeterminate), RelativeSource={RelativeSource TemplatedParent}}" Maximum="{Binding (wpf:ButtonProgressAssist.Maximum), RelativeSource={RelativeSource TemplatedParent}}" Minimum="{Binding (wpf:ButtonProgressAssist.Minimum), RelativeSource={RelativeSource TemplatedParent}}" Opacity="{Binding (wpf:ButtonProgressAssist.Opacity), RelativeSource={RelativeSource TemplatedParent}}" Style="{DynamicResource MaterialDesignLinearProgressBar}" Value="{Binding (wpf:ButtonProgressAssist.Value), RelativeSource={RelativeSource TemplatedParent}}" VerticalAlignment="Center" Visibility="{Binding (wpf:ButtonProgressAssist.IsIndicatorVisible), Converter={StaticResource BooleanToVisibilityConverter}, RelativeSource={RelativeSource TemplatedParent}}" Width="{Binding ActualWidth, RelativeSource={RelativeSource FindAncestor, AncestorType={x:Type ButtonBase}}}">

<ProgressBar.Clip>

<MultiBinding Converter="{StaticResource BorderClipConverter}">

<Binding ElementName="border" Path="ActualWidth"/>

<Binding ElementName="border" Path="ActualHeight"/>

<Binding ElementName="border" Path="CornerRadius"/>

<Binding ElementName="border" Path="BorderThickness"/>

</MultiBinding>

</ProgressBar.Clip>

</ProgressBar>

</Grid>

</AdornerDecorator>

<wpf:Ripple ContentStringFormat="{TemplateBinding ContentStringFormat}" ContentTemplate="{TemplateBinding ContentTemplate}" Content="{TemplateBinding Content}" Focusable="False" HorizontalContentAlignment="{TemplateBinding HorizontalContentAlignment}" Padding="{TemplateBinding Padding}" SnapsToDevicePixels="{TemplateBinding SnapsToDevicePixels}" VerticalContentAlignment="{TemplateBinding VerticalContentAlignment}">

<wpf:Ripple.Clip>

<MultiBinding Converter="{StaticResource BorderClipConverter}">

<Binding ElementName="border" Path="ActualWidth"/>

<Binding ElementName="border" Path="ActualHeight"/>

<Binding ElementName="border" Path="CornerRadius"/>

<Binding ElementName="border" Path="BorderThickness"/>

</MultiBinding>

</wpf:Ripple.Clip>

</wpf:Ripple>

</Grid>

<ControlTemplate.Triggers>

<Trigger Property="IsMouseOver" Value="true">

<Setter Property="wpf:ShadowAssist.Darken" TargetName="border" Value="True"/>

</Trigger>

<Trigger Property="IsKeyboardFocused" Value="true">

<Setter Property="wpf:ShadowAssist.Darken" TargetName="border" Value="True"/>

</Trigger>

<Trigger Property="IsEnabled" Value="false">

<Setter Property="Opacity" Value="0.38"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

<Setter Property="TextBlock.FontSize" Value="14"/>

<Setter Property="TextBlock.FontWeight" Value="Medium"/>

<Setter Property="VerticalContentAlignment" Value="Center"/>

<Setter Property="wpf:ButtonAssist.CornerRadius" Value="2"/>

<Setter Property="wpf:ButtonProgressAssist.IndicatorBackground" Value="{DynamicResource PrimaryHueMidBrush}"/>

<Setter Property="wpf:ButtonProgressAssist.IndicatorForeground" Value="{DynamicResource PrimaryHueMidForegroundBrush}"/>

<Setter Property="wpf:ButtonProgressAssist.IsIndicatorVisible" Value="False"/>

<Setter Property="wpf:ButtonProgressAssist.Opacity" Value=".4"/>

<Setter Property="wpf:ElevationAssist.Elevation" Value="Dp2"/>

<Setter Property="wpf:RippleAssist.Feedback" Value="White"/>

<Style.Triggers>

<MultiTrigger>

<MultiTrigger.Conditions>

<Condition Property="IsMouseOver" Value="true"/>

</MultiTrigger.Conditions>

<MultiTrigger.Setters>

<Setter Property="Foreground" Value="Gray"/>

</MultiTrigger.Setters>

</MultiTrigger>

</Style.Triggers>

</Style>

</Page.Resources>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="1\*"/>

<ColumnDefinition Width="5\*"/>

<ColumnDefinition Width="1\*"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="2\*"/>

<RowDefinition Height="5\*"/>

<RowDefinition Height="1\*"/>

</Grid.RowDefinitions>

<Image Source="/Resources/Logo.png" HorizontalAlignment="Center" VerticalAlignment="Center"

Grid.Column="1" Stretch="UniformToFill" Width="200" Margin="0,20,0,0"/>

<StackPanel Grid.Row="1" Grid.Column="1" VerticalAlignment="Center" Height="210">

<TextBox x:Name="Txtbox1"

materialDesign:HintAssist.Hint="Username"

Margin="6"

Background="{DynamicResource SecondCol}"

Width="350"

Height="45"

FontSize="18"

HorizontalContentAlignment="Left"

VerticalContentAlignment="Center"

Foreground="{DynamicResource Text}"/>

<PasswordBox x:Name="Psbox1" Tag="Password"

materialDesign:HintAssist.Hint="Password"

Margin="6"

Background="{DynamicResource SecondCol}"

Width="350"

Height="45"

FontSize="18"

HorizontalContentAlignment="Left"

VerticalContentAlignment="Center"

Foreground="{DynamicResource Text}"/>

<Button Style="{DynamicResource ButtonStyle1}" x:Name="Btn1" Height="25" Width="120" Content="Вход"

Click="Btn1\_Click"

Foreground="{DynamicResource Text}"/>

<Button Style="{DynamicResource ButtonStyle2}" Background="Transparent" Content="Registration" x:Name="BtnReg"

FontSize="10"

Width="100" Height="23" Click="BtnReg\_Click"

BorderBrush="Transparent"/>

</StackPanel>

</Grid>

</Page>

Cs код:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using WpFStoreJew.Classes;

namespace WpFStoreJew.Pages

{

/// <summary>

/// Логика взаимодействия для PageAut.xaml

/// </summary>

public partial class PageAut : Page

{

public PageAut()

{

InitializeComponent();

}

private void Btn1\_Click(object sender, RoutedEventArgs e)

{

String Login = Txtbox1.Text;

String Password = Psbox1.Password;

if (Login != "" && Password != "")

{

var Level = StoreJewEntities.GetContext().Customers\_Z.FirstOrDefault(x => x.Login == Txtbox1.Text && x.Password == Psbox1.Password);

if (Level != null)

{

if (Level.Level\_Access == "Администратор")

{

MessageBox.Show($"Администратор авторизирован!");

ClassFrame.FrmObj.Navigate(new PageMain(Level));

Txtbox1.Text = null;

Psbox1.Password = null;

}

else if (Level.Level\_Access == "Пользователь")

{

MessageBox.Show($"Пользователь авторизирован!");

ClassFrame.FrmObj.Navigate(new PageMain(Level));

Txtbox1.Text = null;

Psbox1.Password = null;

}

}

else MessageBox.Show("Данный пользователь не существует");

}

else

{

MessageBox.Show("Поля не должны быть пустыми");

}

}

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

{

Classes.ClassFrame.FrmObj.Navigate(new PageRegistration());

}

}

}

Код страницы «PageBasket » (см. рисунок 17)

Xaml код:

<Page x:Class="WpFStoreJew.Pages.PageBasket"

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:WpFStoreJew.Pages"

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

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="PageBasket"

xmlns:converters="clr-namespace:MaterialDesignThemes.Wpf.Converters;assembly=MaterialDesignThemes.Wpf"

xmlns:wpf="clr-namespace:MaterialDesignThemes.Wpf;assembly=MaterialDesignThemes.Wpf">

<Page.Resources>

<Style x:Key="ButtonStyle1" TargetType="{x:Type Button}">

<Setter Property="Background" Value="{DynamicResource SecondCol}"/>

<Setter Property="BorderBrush" Value="{DynamicResource SecondCol}"/>

<Setter Property="BorderThickness" Value="1"/>

<Setter Property="Cursor" Value="Hand"/>

<Setter Property="FocusVisualStyle" Value="{StaticResource FocusVisual}"/>

<Setter Property="Foreground" Value="{DynamicResource Text}"/>

<Setter Property="Height" Value="32"/>

<Setter Property="HorizontalContentAlignment" Value="Center"/>

<Setter Property="Padding" Value="16,4,16,4"/>

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="{x:Type Button}">

<Grid>

<AdornerDecorator CacheMode="{Binding (wpf:ShadowAssist.CacheMode), RelativeSource={RelativeSource Self}}">

<Grid>

<Border x:Name="border" Background="{TemplateBinding Background}" BorderBrush="{TemplateBinding BorderBrush}" BorderThickness="{TemplateBinding BorderThickness}" CornerRadius="{Binding (wpf:ButtonAssist.CornerRadius), RelativeSource={RelativeSource TemplatedParent}}" Effect="{Binding (wpf:ElevationAssist.Elevation), Converter={x:Static converters:ShadowConverter.Instance}, RelativeSource={RelativeSource TemplatedParent}}"/>

<ProgressBar x:Name="ProgressBar" Background="{Binding (wpf:ButtonProgressAssist.IndicatorBackground), RelativeSource={RelativeSource TemplatedParent}}" BorderBrush="{Binding (wpf:ButtonProgressAssist.IndicatorBackground), RelativeSource={RelativeSource TemplatedParent}}" Foreground="{Binding (wpf:ButtonProgressAssist.IndicatorForeground), RelativeSource={RelativeSource TemplatedParent}}" HorizontalAlignment="Left" Height="{TemplateBinding Height}" IsIndeterminate="{Binding (wpf:ButtonProgressAssist.IsIndeterminate), RelativeSource={RelativeSource TemplatedParent}}" Maximum="{Binding (wpf:ButtonProgressAssist.Maximum), RelativeSource={RelativeSource TemplatedParent}}" Minimum="{Binding (wpf:ButtonProgressAssist.Minimum), RelativeSource={RelativeSource TemplatedParent}}" Opacity="{Binding (wpf:ButtonProgressAssist.Opacity), RelativeSource={RelativeSource TemplatedParent}}" Style="{DynamicResource MaterialDesignLinearProgressBar}" Value="{Binding (wpf:ButtonProgressAssist.Value), RelativeSource={RelativeSource TemplatedParent}}" VerticalAlignment="Center" Visibility="{Binding (wpf:ButtonProgressAssist.IsIndicatorVisible), Converter={StaticResource BooleanToVisibilityConverter}, RelativeSource={RelativeSource TemplatedParent}}" Width="{Binding ActualWidth, RelativeSource={RelativeSource FindAncestor, AncestorType={x:Type ButtonBase}}}">

<ProgressBar.Clip>

<MultiBinding Converter="{StaticResource BorderClipConverter}">

<Binding ElementName="border" Path="ActualWidth"/>

<Binding ElementName="border" Path="ActualHeight"/>

<Binding ElementName="border" Path="CornerRadius"/>

<Binding ElementName="border" Path="BorderThickness"/>

</MultiBinding>

</ProgressBar.Clip>

</ProgressBar>

</Grid>

</AdornerDecorator>

<wpf:Ripple ContentStringFormat="{TemplateBinding ContentStringFormat}" ContentTemplate="{TemplateBinding ContentTemplate}" Content="{TemplateBinding Content}" Focusable="False" HorizontalContentAlignment="{TemplateBinding HorizontalContentAlignment}" Padding="{TemplateBinding Padding}" SnapsToDevicePixels="{TemplateBinding SnapsToDevicePixels}" VerticalContentAlignment="{TemplateBinding VerticalContentAlignment}">

<wpf:Ripple.Clip>

<MultiBinding Converter="{StaticResource BorderClipConverter}">

<Binding ElementName="border" Path="ActualWidth"/>

<Binding ElementName="border" Path="ActualHeight"/>

<Binding ElementName="border" Path="CornerRadius"/>

<Binding ElementName="border" Path="BorderThickness"/>

</MultiBinding>

</wpf:Ripple.Clip>

</wpf:Ripple>

</Grid>

<ControlTemplate.Triggers>

<Trigger Property="IsMouseOver" Value="true">

<Setter Property="wpf:ShadowAssist.Darken" TargetName="border" Value="True"/>

</Trigger>

<Trigger Property="IsKeyboardFocused" Value="true">

<Setter Property="wpf:ShadowAssist.Darken" TargetName="border" Value="True"/>

</Trigger>

<Trigger Property="IsEnabled" Value="false">

<Setter Property="Opacity" Value="0.38"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

<Setter Property="TextBlock.FontSize" Value="14"/>

<Setter Property="TextBlock.FontWeight" Value="Medium"/>

<Setter Property="VerticalContentAlignment" Value="Center"/>

<Setter Property="wpf:ButtonAssist.CornerRadius" Value="2"/>

<Setter Property="wpf:ButtonProgressAssist.IndicatorBackground" Value="{DynamicResource PrimaryHueMidBrush}"/>

<Setter Property="wpf:ButtonProgressAssist.IndicatorForeground" Value="{DynamicResource PrimaryHueMidForegroundBrush}"/>

<Setter Property="wpf:ButtonProgressAssist.IsIndicatorVisible" Value="False"/>

<Setter Property="wpf:ButtonProgressAssist.Opacity" Value=".4"/>

<Setter Property="wpf:ElevationAssist.Elevation" Value="Dp2"/>

<Setter Property="wpf:RippleAssist.Feedback" Value="White"/>

</Style>

</Page.Resources>

<Grid Background="{DynamicResource Backdrop}">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="7\*"/>

<ColumnDefinition Width="1.8\*"/>

</Grid.ColumnDefinitions>

<DataGrid Name="DbCart" AutoGenerateColumns="False" ScrollViewer.CanContentScroll="True"

ScrollViewer.VerticalScrollBarVisibility="Visible" Height="auto" Grid.Column="0" >

<DataGrid.Columns>

<DataGridTextColumn Header="Article\_Jew"

Width="95"

Binding="{Binding Article\_Jew}" IsReadOnly="True"/>

<DataGridTextColumn Width="auto"

Header="Name\_Jew"

Binding="{Binding Name\_Jew}" IsReadOnly="True"/>

<DataGridTextColumn Header="Sum\_Cost"

Width="auto"

Binding="{Binding Sum\_Cost}" IsReadOnly="True"/>

<DataGridTextColumn Header="Quantity"

Width="auto"

Binding="{Binding Quantity}" IsReadOnly="True" />

<DataGridTextColumn Header="Date\_Order"

Width="auto"

Binding="{Binding Date\_Order}" IsReadOnly="True" />

</DataGrid.Columns>

</DataGrid>

<StackPanel Grid.Column="1">

<TextBlock x:Name="txtCount" FontSize="15" FontStyle="Italic" Foreground="{DynamicResource Text}" Margin="0,360,0,0" />

<TextBlock x:Name="txtSum" FontSize="16" FontStyle="Italic" Foreground="{DynamicResource Text}" Margin="0,20,0,0" TextWrapping="Wrap" />

<Button Style="{DynamicResource ButtonStyle1}" x:Name="btnBuy" Click="btnBuy\_Click"

Content="Купить"/>

</StackPanel>

</Grid>

</Page>

Cs код:

using Microsoft.Office.Interop.Excel;

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using WpFStoreJew.Classes;

using Excel = Microsoft.Office.Interop.Excel;

namespace WpFStoreJew.Pages

{

/// <summary>

/// Логика взаимодействия для PageBasket.xaml

/// </summary>

public partial class PageBasket : System.Windows.Controls.Page

{

private int ID;

public PageBasket(Customers\_Z level)

{

InitializeComponent();

ID = level.ID;

UpdateDataGrid(DbCart);

int Count = DbCart.Items.Count-1;

string Counttext = "Всего " + Count.ToString() + " Позиций"; // Количество записей после

txtCount.Text = Counttext.ToString();

double Sum = 0;

foreach (var item in DbCart.Items)

{

if (item is Orders\_Z order)

{

Sum += order.Sum\_Cost;

}

}

string Sumtext = "Сумма " + Sum.ToString() + " ₽"; // Количество записей после

txtSum.Text = Sumtext.ToString();

}

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

{

if (DbCart.Items.Count == 1)

{

MessageBox.Show("Корзина пуста. Пожалуйста, добавьте товары перед оплатой.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

return;

}

var app = new Excel.Application();

Excel.Workbook wb = app.Workbooks.Add();

Excel.Worksheet worksheet =

(Excel.Worksheet)wb.Worksheets[1];

int indexRows = 2;

double Count = 0;

var listcart = StoreJewEntities.GetContext().Orders\_Z.Where(x => x.Id\_Customer == ID).ToList();

worksheet.Cells[1][indexRows] = "Article";

worksheet.Cells[2][indexRows] = "Name";

worksheet.Cells[3][indexRows] = "Quantity";

worksheet.Cells[4][indexRows] = "Cost\_Jew";

worksheet.Cells[5][indexRows] = "Sum\_Cost\_Jew";

Excel.Range \_excelCells1 = (Excel.Range)worksheet.get\_Range("B1", "C1").Cells;

\_excelCells1.Merge(Type.Missing);

worksheet.Cells[2][1] = " Товарный чек";

worksheet.Cells[1].ColumnWidth = 6;

worksheet.Cells[2].ColumnWidth = 50;

worksheet.Cells[3].ColumnWidth = 8;

worksheet.Cells[4].ColumnWidth = 14;

worksheet.Cells[5].ColumnWidth = 14;

Excel.Range range = worksheet.Range["D1:E9999"];

range.NumberFormat = "0.00 ₽";

foreach (var jew in listcart)

{

Count = Count + jew.Sum\_Cost;

indexRows++;

worksheet.Cells[2][indexRows+1] =$" Дата Заказа: {DateTime.Now.ToString()}";

worksheet.Cells[1][indexRows] = jew.Article\_Jew;

worksheet.Cells[2][indexRows] = jew.Name\_Jew;

worksheet.Cells[3][indexRows] = jew.Quantity;

worksheet.Cells[4][indexRows] = jew.Sum\_Cost/jew.Quantity;

worksheet.Cells[5][indexRows] = jew.Sum\_Cost;

worksheet.Cells[4][indexRows + 1] = "Сумма заказа :";

worksheet.Cells[5][indexRows+1] = Count.ToString();

}

Excel.Range rangeBorders = worksheet.Range[worksheet.Cells[1][1], worksheet.Cells[5][indexRows +1]];

rangeBorders.Borders[Excel.XlBordersIndex.xlEdgeBottom].LineStyle =

rangeBorders.Borders[Excel.XlBordersIndex.xlEdgeLeft].LineStyle =

rangeBorders.Borders[Excel.XlBordersIndex.xlEdgeRight].LineStyle =

rangeBorders.Borders[Excel.XlBordersIndex.xlEdgeTop].LineStyle =

rangeBorders.Borders[Excel.XlBordersIndex.xlInsideVertical].LineStyle =

rangeBorders.Borders[Excel.XlBordersIndex.xlInsideHorizontal].LineStyle =

Excel.XlLineStyle.xlContinuous;

Excel.Range rangeCenter = worksheet.Range[worksheet.Cells[3][3], worksheet.Cells[3][indexRows]];

rangeCenter.HorizontalAlignment = Excel.XlHAlign.xlHAlignCenter;

app.Visible = true;

using (var dbContext = new StoreJewEntities())

{

var ordersToDelete = dbContext.Orders\_Z.Where(o => o.Id\_Customer == ID).ToList();

dbContext.Orders\_Z.RemoveRange(ordersToDelete);

dbContext.SaveChanges();

}

UpdateDataGrid(DbCart);

}

private void UpdateDataGrid(DataGrid dataGrid)

{

using (var dbContext = new StoreJewEntities())

{

var jew = dbContext.Jewellery\_Z.ToList();

var userOrders = dbContext.Orders\_Z.Where(o => o.Id\_Customer == ID).ToList();

dataGrid.ItemsSource = userOrders;

}

}

}

}

Код страницы «PageMain »

Xaml код:

<Page x:Class="WpFStoreJew.Pages.PageMain"

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

xmlns:local="clr-namespace:WpFStoreJew.Pages"

mc:Ignorable="d"

d:DesignHeight="680" d:DesignWidth="920"

Title="PageMain">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="0.35\*"/>

<ColumnDefinition Width="4\*"/>

</Grid.ColumnDefinitions>

<Frame Grid.Column="1" x:Name="frmStore" NavigationUIVisibility="Hidden"/>

<Grid>

<StackPanel>

<Button Style="{StaticResource btnMenu}" Click="btnMain\_Click" x:Name="btnMain">

<Image Style="{DynamicResource Home}" Width="40"/>

</Button>

<Button Style="{StaticResource btnMenu}" Click="btnCatalog\_Click" x:Name="btnCatalog">

<Image Style="{DynamicResource Store}" Width="40"/>

</Button>

<Button Style="{StaticResource btnMenu}" Click="btnBasket\_Click" x:Name="btnBasket">

<Image Style="{DynamicResource Cart}" Width="40"/>

</Button>

<ToggleButton Checked="ToggleButton\_Checked"

Unchecked="ToggleButton\_Unchecked"

Name="tgbThemes" Background="#FF515151" Margin="0,320,0,0"/>

<Button Style="{StaticResource btnMenu}"

x:Name="btnSetting" Click="btnSetting\_Click" >

<Image Style="{DynamicResource Setting}" Width="40"/>

</Button>

<Button Style="{StaticResource btnMenu}" x:Name="btnExit" Click="btnExit\_Click" >

<Image Style="{DynamicResource Exit}" Width="40"/>

</Button>

</StackPanel>

</Grid>

</Grid>

</Page>

Cs код:

using System;

using System.Collections.Generic;

using System.Data.Entity.Infrastructure;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using WpFStoreJew.Classes;

namespace WpFStoreJew.Pages

{

public partial class PageMain : Page

{

private Customers\_Z Level = new Customers\_Z();

public PageMain(Customers\_Z level)

{

InitializeComponent();

if (level != null)

Level = level;

DataContext = Level;

if (Level.Level\_Access == "Пользователь")

{

btnSetting.Visibility = Visibility.Collapsed;

}

Classes.ClassFrame.FrmStore = frmStore;

Classes.ClassFrame.FrmStore.Navigate(new PageMainPage());

}

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

{

if(frmStore.Content is PageStore)

{

return;

}

else Classes.ClassFrame.FrmStore.Navigate(new PageStore(Level));

}

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

{

if (frmStore.Content is PageMainPage)

{

return;

}

else Classes.ClassFrame.FrmStore.Navigate(new PageMainPage());

}

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

{

if (frmStore.Content is PageBasket)

{

return;

}

else Classes.ClassFrame.FrmStore.Navigate(new PageBasket(Level));

}

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

{

if (frmStore.Content is PageSetting)

{

return;

}

else Classes.ClassFrame.FrmStore.Navigate(new PageSetting());

}

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

{

if (frmStore.Content is PageAut)

{

return;

}

else Classes.ClassFrame.FrmObj.Navigate(new PageAut());

}

private void ToggleButton\_Checked(object sender, RoutedEventArgs e)

{

AppThemes.ChangeThemes(new Uri("Themes/Dark.xaml", UriKind.Relative));

}

private void ToggleButton\_Unchecked(object sender, RoutedEventArgs e)

{

AppThemes.ChangeThemes(new Uri("Themes/Light.xaml", UriKind.Relative));

}

}

}

Код страницы «PageMainPage » (см. рисунок 15)

Xaml код:

<Page x:Class="WpFStoreJew.Pages.PageMainPage"

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:WpFStoreJew.Pages"

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

mc:Ignorable="d"

d:DesignHeight="600" d:DesignWidth="920"

Title="PageMainPage">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition/>

<ColumnDefinition/>

<ColumnDefinition />

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="1.5\*"/>

<RowDefinition Height="3\*"/>

<RowDefinition Height="0.6\*"/>

</Grid.RowDefinitions>

<Image Source="/Resources/Logo.png" Height="200" Width="200" VerticalAlignment="Top"

Grid.Column="1" Stretch="UniformToFill" Margin="0,-30,0,0"/>

<StackPanel Grid.Row="1" VerticalAlignment="Top" Grid.ColumnSpan="3" Margin="0,-40,0,0" >

<TextBlock Style="{DynamicResource text}" Text="Добро пожаловать в Aurus – ваш идеальный партнер в мире ювелирных изделий!"

FontSize="18"

TextAlignment="Center"

FontFamily="Madimi One"

/>

<TextBlock Style="{DynamicResource text}" Text="Aurus предлагает вам уникальный выбор изысканных и неповторимых украшений, созданных с любовью и мастерством. Наш ассортимент включает в себя широкий выбор колец, серег, ожерелий и браслетов, изготовленных из самых высококачественных материалов."

FontSize="13" TextWrapping="Wrap"

Width="600" Height="auto"

TextAlignment="Center"

FontFamily="Open Sans"

FontStyle="Italic"/>

</StackPanel>

<StackPanel Grid.Row="1" Grid.ColumnSpan="3" VerticalAlignment="Top" Orientation="Horizontal" HorizontalAlignment="Center" Margin="0,40,0,0">

<Border CornerRadius="15" Height="240" Width="180" ToolTip="Кольцо в белом золоте с сапфиром" >

<Border.Background >

<ImageBrush Stretch="UniformToFill" ImageSource="/Resources/mainjew1.jpg"/>

</Border.Background>

</Border>

<TextBlock/>

<Border CornerRadius="15" Height="240" Width="180" Margin="60,0,0,0" ToolTip="Серебряное кольцо с бесцветными фианитами">

<Border.Background>

<ImageBrush Stretch="UniformToFill" ImageSource="/Resources/mainjew2.jpg"/>

</Border.Background>

</Border>

<Border CornerRadius="15" Height="240" Width="180" Margin="60,0,0,0" ToolTip="Золотое кольцо с аметистом">

<Border.Background>

<ImageBrush Stretch="UniformToFill" ImageSource="/Resources/mainjew3.jpg"/>

</Border.Background>

</Border>

</StackPanel>

<StackPanel Grid.Row="1" Grid.RowSpan="2" Grid.ColumnSpan="3" VerticalAlignment="Bottom" Orientation="Horizontal" HorizontalAlignment="Center" Margin="0,140,0,0">

<TextBlock Text="Мы стремимся к тому, чтобы каждое изделие в нашем магазине было уникальным, именно поэтому мы работаем только с лучшими мастерами-ювелирами и используем только самые качественные материалы." TextWrapping="Wrap" Width="200" Style="{DynamicResource text}" FontSize="13" Height="140" VerticalAlignment="Bottom" TextAlignment="Center"/>

<TextBlock Text="Aurus – это не просто магазин ювелирных изделий, это место, где каждый клиент получает индивидуальное внимание и помощь в выборе украшения, которое подчеркнет вашу неповторимую красоту и стиль." TextWrapping="Wrap" Width="200" Margin="40,0,0,0" Style="{DynamicResource text}" FontSize="13" Height="140" VerticalAlignment="Bottom" TextAlignment="Center"/>

<TextBlock Text="Погрузитесь в мир изысканных украшений с Aurus и найдите тот самый, который станет вашим символом стиля и элегантности!" TextWrapping="Wrap" Width="200" Margin="40,0,0,0" Style="{DynamicResource text}" FontSize="13" Height="140" VerticalAlignment="Bottom" TextAlignment="Center"/>

</StackPanel>

</Grid>

</Page>

Cs код:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace WpFStoreJew.Pages

{

/// <summary>

/// Логика взаимодействия для PageMainPage.xaml

/// </summary>

public partial class PageMainPage : Page

{

public PageMainPage()

{

InitializeComponent();

}

}

}

Код страницы «PageRegistration » (см. рисунок 14)

Xaml код:

<Page x:Class="WpFStoreJew.Pages.PageRegistration"

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

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

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

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

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

xmlns:local="clr-namespace:WpFStoreJew.Pages"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="PageRegistration"

xmlns:converters="clr-namespace:MaterialDesignThemes.Wpf.Converters;assembly=MaterialDesignThemes.Wpf"

xmlns:wpf="clr-namespace:MaterialDesignThemes.Wpf;assembly=MaterialDesignThemes.Wpf">

<Page.Resources>

<Style x:Key="ButtonStyle1" TargetType="{x:Type Button}">

<Setter Property="Background" Value="{DynamicResource SecondCol}"/>

<Setter Property="BorderBrush" Value="{DynamicResource SecondCol}"/>

<Setter Property="BorderThickness" Value="1"/>

<Setter Property="Cursor" Value="Hand"/>

<Setter Property="FocusVisualStyle" Value="{StaticResource FocusVisual}"/>

<Setter Property="Foreground" Value="{DynamicResource Text}"/>

<Setter Property="Height" Value="32"/>

<Setter Property="HorizontalContentAlignment" Value="Center"/>

<Setter Property="Padding" Value="16,4,16,4"/>

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="{x:Type Button}">

<Grid>

<AdornerDecorator CacheMode="{Binding (wpf:ShadowAssist.CacheMode), RelativeSource={RelativeSource Self}}">

<Grid>

<Border x:Name="border" Background="{TemplateBinding Background}" BorderBrush="{TemplateBinding BorderBrush}" BorderThickness="{TemplateBinding BorderThickness}" CornerRadius="{Binding (wpf:ButtonAssist.CornerRadius), RelativeSource={RelativeSource TemplatedParent}}" Effect="{Binding (wpf:ElevationAssist.Elevation), Converter={x:Static converters:ShadowConverter.Instance}, RelativeSource={RelativeSource TemplatedParent}}"/>

<ProgressBar x:Name="ProgressBar" Background="{Binding (wpf:ButtonProgressAssist.IndicatorBackground), RelativeSource={RelativeSource TemplatedParent}}" BorderBrush="{Binding (wpf:ButtonProgressAssist.IndicatorBackground), RelativeSource={RelativeSource TemplatedParent}}" Foreground="{Binding (wpf:ButtonProgressAssist.IndicatorForeground), RelativeSource={RelativeSource TemplatedParent}}" HorizontalAlignment="Left" Height="{TemplateBinding Height}" IsIndeterminate="{Binding (wpf:ButtonProgressAssist.IsIndeterminate), RelativeSource={RelativeSource TemplatedParent}}" Maximum="{Binding (wpf:ButtonProgressAssist.Maximum), RelativeSource={RelativeSource TemplatedParent}}" Minimum="{Binding (wpf:ButtonProgressAssist.Minimum), RelativeSource={RelativeSource TemplatedParent}}" Opacity="{Binding (wpf:ButtonProgressAssist.Opacity), RelativeSource={RelativeSource TemplatedParent}}" Style="{DynamicResource MaterialDesignLinearProgressBar}" Value="{Binding (wpf:ButtonProgressAssist.Value), RelativeSource={RelativeSource TemplatedParent}}" VerticalAlignment="Center" Visibility="{Binding (wpf:ButtonProgressAssist.IsIndicatorVisible), Converter={StaticResource BooleanToVisibilityConverter}, RelativeSource={RelativeSource TemplatedParent}}" Width="{Binding ActualWidth, RelativeSource={RelativeSource FindAncestor, AncestorType={x:Type ButtonBase}}}">

<ProgressBar.Clip>

<MultiBinding Converter="{StaticResource BorderClipConverter}">

<Binding ElementName="border" Path="ActualWidth"/>

<Binding ElementName="border" Path="ActualHeight"/>

<Binding ElementName="border" Path="CornerRadius"/>

<Binding ElementName="border" Path="BorderThickness"/>

</MultiBinding>

</ProgressBar.Clip>

</ProgressBar>

</Grid>

</AdornerDecorator>

<wpf:Ripple ContentStringFormat="{TemplateBinding ContentStringFormat}" ContentTemplate="{TemplateBinding ContentTemplate}" Content="{TemplateBinding Content}" Focusable="False" HorizontalContentAlignment="{TemplateBinding HorizontalContentAlignment}" Padding="{TemplateBinding Padding}" SnapsToDevicePixels="{TemplateBinding SnapsToDevicePixels}" VerticalContentAlignment="{TemplateBinding VerticalContentAlignment}">

<wpf:Ripple.Clip>

<MultiBinding Converter="{StaticResource BorderClipConverter}">

<Binding ElementName="border" Path="ActualWidth"/>

<Binding ElementName="border" Path="ActualHeight"/>

<Binding ElementName="border" Path="CornerRadius"/>

<Binding ElementName="border" Path="BorderThickness"/>

</MultiBinding>

</wpf:Ripple.Clip>

</wpf:Ripple>

</Grid>

<ControlTemplate.Triggers>

<Trigger Property="IsMouseOver" Value="true">

<Setter Property="wpf:ShadowAssist.Darken" TargetName="border" Value="True"/>

</Trigger>

<Trigger Property="IsKeyboardFocused" Value="true">

<Setter Property="wpf:ShadowAssist.Darken" TargetName="border" Value="True"/>

</Trigger>

<Trigger Property="IsEnabled" Value="false">

<Setter Property="Opacity" Value="0.38"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

<Setter Property="TextBlock.FontSize" Value="14"/>

<Setter Property="TextBlock.FontWeight" Value="Medium"/>

<Setter Property="VerticalContentAlignment" Value="Center"/>

<Setter Property="wpf:ButtonAssist.CornerRadius" Value="2"/>

<Setter Property="wpf:ButtonProgressAssist.IndicatorBackground" Value="{DynamicResource PrimaryHueMidBrush}"/>

<Setter Property="wpf:ButtonProgressAssist.IndicatorForeground" Value="{DynamicResource PrimaryHueMidForegroundBrush}"/>

<Setter Property="wpf:ButtonProgressAssist.IsIndicatorVisible" Value="False"/>

<Setter Property="wpf:ButtonProgressAssist.Opacity" Value=".4"/>

<Setter Property="wpf:ElevationAssist.Elevation" Value="Dp2"/>

<Setter Property="wpf:RippleAssist.Feedback" Value="White"/>

</Style>

</Page.Resources>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="1\*"/>

<ColumnDefinition Width="5\*"/>

<ColumnDefinition Width="1\*"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="2\*"/>

<RowDefinition Height="5\*"/>

<RowDefinition Height="1\*"/>

</Grid.RowDefinitions>

<Image Source="/Resources/Logo.png" HorizontalAlignment="Center" VerticalAlignment="Center"

Grid.Column="1" Stretch="UniformToFill" Width="200" Margin="0,20,0,0"/>

<StackPanel Grid.Row="1" Grid.Column="1" VerticalAlignment="Center" Height="210">

<TextBox x:Name="Txtbox1"

materialDesign:HintAssist.Hint="Username"

Margin="6"

Background="{DynamicResource SecondCol}"

Width="350"

Height="45"

FontSize="18"

HorizontalContentAlignment="Left"

VerticalContentAlignment="Center"

Foreground="{DynamicResource Text}"

PreviewKeyDown="Psbox1\_PreviewKeyDown"/>

<PasswordBox x:Name="Psbox1" Tag="Password"

materialDesign:HintAssist.Hint="Password"

Margin="6"

Background="{DynamicResource SecondCol}"

Width="350"

Height="45"

FontSize="18"

HorizontalContentAlignment="Left"

VerticalContentAlignment="Center"

Foreground="{DynamicResource Text}"

PreviewKeyDown="Psbox1\_PreviewKeyDown"/>

<PasswordBox x:Name="Psbox1\_Copy" Tag="Password\_Copy"

materialDesign:HintAssist.Hint="Redeem Password"

Margin="6"

Background="{DynamicResource SecondCol}"

Width="350"

Height="45"

FontSize="18"

HorizontalContentAlignment="Left"

VerticalContentAlignment="Center"

PasswordChanged="Psbox1\_Copy\_PasswordChanged"

Foreground="{DynamicResource Text}"

PreviewKeyDown="Psbox1\_PreviewKeyDown"/>

<StackPanel Orientation="Horizontal" Width="350">

<Button Style="{DynamicResource ButtonStyle1}" x:Name="btnBack" Click="btnBack\_Click" Content="Back"

Width="120" Height="25" Margin="0,0,30,0" Foreground="{DynamicResource Text}"/>

<Button Style="{DynamicResource ButtonStyle1}" x:Name="btnReg" Click="btnReg\_Click" Content="Registration"

Width="200" Height="25" FontSize="13" Foreground="{DynamicResource Text}"/>

</StackPanel>

</StackPanel>

</Grid>

</Page>

Cs код:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using WpFStoreJew.Classes;

namespace WpFStoreJew.Pages

{

/// <summary>

/// Логика взаимодействия для PageRegistration.xaml

/// </summary>

public partial class PageRegistration : Page

{

public PageRegistration()

{

InitializeComponent();

}

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

{

Classes.ClassFrame.FrmObj.GoBack();

}

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

{

if (AppConnect.StoreJewEntities.Customers\_Z.Count(x => x.Login == Txtbox1.Text) > 0)

{

Txtbox1.ToolTip = "Пользователь с таким логином уже есть";

MessageBox.Show("Пользователь с таким логином уже есть");

return;

}

try

{

if (Txtbox1.Text != "" && Psbox1.Password != "")

{

if (Txtbox1.Text.Length < 21 && Txtbox1.Text.Length >= 5 && Psbox1.Password.Length < 16 && Psbox1.Password.Length >= 6)

{

Customers\_Z customer = new Customers\_Z()

{

Login = Txtbox1.Text,

Password = Psbox1.Password,

Level\_Access = "Пользователь",

};

var Level = StoreJewEntities.GetContext().Customers\_Z.FirstOrDefault(x => x.Login == Txtbox1.Text && x.Password == Psbox1.Password);

AppConnect.StoreJewEntities.Customers\_Z.Add(customer);

AppConnect.StoreJewEntities.SaveChanges();

ClassFrame.FrmObj.Navigate(new PageMain(Level));

}

else MessageBox.Show("Логин должен состоять от 5 до 20 символов \n Пароль должен состоять от 6 до 15 символов");

}

else MessageBox.Show("Поля должны быть заполнены");

}

catch

{

MessageBox.Show("Не удалось добавить данные");

}

}

private void Psbox1\_Copy\_PasswordChanged(object sender, RoutedEventArgs e)

{

if (Psbox1\_Copy.Password != Psbox1.Password)

{

btnReg.IsEnabled = false;

}

else

{

btnReg.IsEnabled = true;

}

}

private void Psbox1\_PreviewKeyDown(object sender, KeyEventArgs e)

{

if(e.Key == Key.Space)

{

e.Handled = true;

}

}

}

}

Код страницы «PageSetting» (см. рисунок 18)

Xaml код:

<Page x:Class="WpFStoreJew.Pages.PageSetting"

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

xmlns:local="clr-namespace:WpFStoreJew.Pages"

mc:Ignorable="d"

d:DesignHeight="600" d:DesignWidth="920"

Title="PageSetting"

xmlns:converters="clr-namespace:MaterialDesignThemes.Wpf.Converters;assembly=MaterialDesignThemes.Wpf"

xmlns:wpf="clr-namespace:MaterialDesignThemes.Wpf;assembly=MaterialDesignThemes.Wpf">

<Page.Resources>

<Style TargetType="{x:Type TextBlock}" x:Key="WrapText">

<Setter Property="TextWrapping" Value="Wrap"/>

</Style>

<Style x:Key="ButtonStyle1" TargetType="{x:Type Button}">

<Setter Property="Background" Value="{DynamicResource SecondCol}"/>

<Setter Property="BorderBrush" Value="{DynamicResource SecondCol}"/>

<Setter Property="BorderThickness" Value="1"/>

<Setter Property="Cursor" Value="Hand"/>

<Setter Property="FocusVisualStyle" Value="{StaticResource FocusVisual}"/>

<Setter Property="Foreground" Value="{DynamicResource Text}"/>

<Setter Property="Height" Value="32"/>

<Setter Property="HorizontalContentAlignment" Value="Center"/>

<Setter Property="Padding" Value="16,4,16,4"/>

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="{x:Type Button}">

<Grid>

<AdornerDecorator CacheMode="{Binding (wpf:ShadowAssist.CacheMode), RelativeSource={RelativeSource Self}}">

<Grid>

<Border x:Name="border" Background="{TemplateBinding Background}" BorderBrush="{TemplateBinding BorderBrush}" BorderThickness="{TemplateBinding BorderThickness}" CornerRadius="{Binding (wpf:ButtonAssist.CornerRadius), RelativeSource={RelativeSource TemplatedParent}}" Effect="{Binding (wpf:ElevationAssist.Elevation), Converter={x:Static converters:ShadowConverter.Instance}, RelativeSource={RelativeSource TemplatedParent}}"/>

<ProgressBar x:Name="ProgressBar" Background="{Binding (wpf:ButtonProgressAssist.IndicatorBackground), RelativeSource={RelativeSource TemplatedParent}}" BorderBrush="{Binding (wpf:ButtonProgressAssist.IndicatorBackground), RelativeSource={RelativeSource TemplatedParent}}" Foreground="{Binding (wpf:ButtonProgressAssist.IndicatorForeground), RelativeSource={RelativeSource TemplatedParent}}" HorizontalAlignment="Left" Height="{TemplateBinding Height}" IsIndeterminate="{Binding (wpf:ButtonProgressAssist.IsIndeterminate), RelativeSource={RelativeSource TemplatedParent}}" Maximum="{Binding (wpf:ButtonProgressAssist.Maximum), RelativeSource={RelativeSource TemplatedParent}}" Minimum="{Binding (wpf:ButtonProgressAssist.Minimum), RelativeSource={RelativeSource TemplatedParent}}" Opacity="{Binding (wpf:ButtonProgressAssist.Opacity), RelativeSource={RelativeSource TemplatedParent}}" Style="{DynamicResource MaterialDesignLinearProgressBar}" Value="{Binding (wpf:ButtonProgressAssist.Value), RelativeSource={RelativeSource TemplatedParent}}" VerticalAlignment="Center" Visibility="{Binding (wpf:ButtonProgressAssist.IsIndicatorVisible), Converter={StaticResource BooleanToVisibilityConverter}, RelativeSource={RelativeSource TemplatedParent}}" Width="{Binding ActualWidth, RelativeSource={RelativeSource FindAncestor, AncestorType={x:Type ButtonBase}}}">

<ProgressBar.Clip>

<MultiBinding Converter="{StaticResource BorderClipConverter}">

<Binding ElementName="border" Path="ActualWidth"/>

<Binding ElementName="border" Path="ActualHeight"/>

<Binding ElementName="border" Path="CornerRadius"/>

<Binding ElementName="border" Path="BorderThickness"/>

</MultiBinding>

</ProgressBar.Clip>

</ProgressBar>

</Grid>

</AdornerDecorator>

<wpf:Ripple ContentStringFormat="{TemplateBinding ContentStringFormat}" ContentTemplate="{TemplateBinding ContentTemplate}" Content="{TemplateBinding Content}" Focusable="False" HorizontalContentAlignment="{TemplateBinding HorizontalContentAlignment}" Padding="{TemplateBinding Padding}" SnapsToDevicePixels="{TemplateBinding SnapsToDevicePixels}" VerticalContentAlignment="{TemplateBinding VerticalContentAlignment}">

<wpf:Ripple.Clip>

<MultiBinding Converter="{StaticResource BorderClipConverter}">

<Binding ElementName="border" Path="ActualWidth"/>

<Binding ElementName="border" Path="ActualHeight"/>

<Binding ElementName="border" Path="CornerRadius"/>

<Binding ElementName="border" Path="BorderThickness"/>

</MultiBinding>

</wpf:Ripple.Clip>

</wpf:Ripple>

</Grid>

<ControlTemplate.Triggers>

<Trigger Property="IsMouseOver" Value="true">

<Setter Property="wpf:ShadowAssist.Darken" TargetName="border" Value="True"/>

</Trigger>

<Trigger Property="IsKeyboardFocused" Value="true">

<Setter Property="wpf:ShadowAssist.Darken" TargetName="border" Value="True"/>

</Trigger>

<Trigger Property="IsEnabled" Value="false">

<Setter Property="Opacity" Value="0.38"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

<Setter Property="TextBlock.FontSize" Value="14"/>

<Setter Property="TextBlock.FontWeight" Value="Medium"/>

<Setter Property="VerticalContentAlignment" Value="Center"/>

<Setter Property="wpf:ButtonAssist.CornerRadius" Value="2"/>

<Setter Property="wpf:ButtonProgressAssist.IndicatorBackground" Value="{DynamicResource PrimaryHueMidBrush}"/>

<Setter Property="wpf:ButtonProgressAssist.IndicatorForeground" Value="{DynamicResource PrimaryHueMidForegroundBrush}"/>

<Setter Property="wpf:ButtonProgressAssist.IsIndicatorVisible" Value="False"/>

<Setter Property="wpf:ButtonProgressAssist.Opacity" Value=".4"/>

<Setter Property="wpf:ElevationAssist.Elevation" Value="Dp2"/>

<Setter Property="wpf:RippleAssist.Feedback" Value="White"/>

</Style>

</Page.Resources>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="0.3\*"/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<StackPanel x:Name="stkMenu" VerticalAlignment="Top" Orientation="Vertical" Margin="0,0,5,0">

<TextBlock x:Name="txtName"

Text="Название изделия"

TextAlignment="Center"

Foreground="{DynamicResource Text}"/>

<TextBox x:Name="txbName"

Foreground="{DynamicResource Text}"

materialDesign:HintAssist.Hint="Название"

Text="{Binding Name}"/>

<TextBlock x:Name="txtType"

Text="Тип изделия"

TextAlignment="Center"

Foreground="{DynamicResource Text}"

Margin="0,10,0,0"/>

<ComboBox x:Name="cmbType"

IsEditable="True"

Foreground="{DynamicResource TextCmb}"

materialDesign:HintAssist.Hint="Тип"

SelectedValue="{Binding Type}"

/>

<TextBlock x:Name="txtCost"

Text="Стоимость изделия"

TextAlignment="Center"

Foreground="{DynamicResource Text}"

Margin="0,10,0,0"/>

<TextBox x:Name="txbCost"

Foreground="{DynamicResource Text}"

materialDesign:HintAssist.Hint="Цена"

Text="{Binding Cost}"/>

<TextBlock x:Name="txtMaterial"

Text="Материал изделия"

TextAlignment="Center"

Foreground="{DynamicResource Text}"

Margin="0,10,0,0"/>

<ComboBox x:Name="cmbMaterial"

IsEditable="True"

Foreground="{DynamicResource TextCmb}"

materialDesign:HintAssist.Hint="Материал"

SelectedValue="{Binding Material}"

/>

<TextBlock x:Name="txtDragCrystal"

Text="Драгоценный камень"

TextAlignment="Center"

Foreground="{DynamicResource Text}"

Margin="0,10,0,0"/>

<ComboBox x:Name="cmbDrag\_Crystal"

IsEditable="True"

Foreground="{DynamicResource TextCmb}"

SelectedValue="{Binding Drag\_Cystal}"

materialDesign:HintAssist.Hint="Драгоценность"

/>

<TextBlock x:Name="txtWeight"

Text="Вес изделия"

TextAlignment="Center"

Foreground="{DynamicResource Text}"

Margin="0,10,0,0"/>

<TextBox x:Name="txbWeight"

Foreground="{DynamicResource Text}"

materialDesign:HintAssist.Hint="Вес"

Text="{Binding Weight}"/>

<TextBlock x:Name="txtImage"

Text="Изображение изделия"

TextAlignment="Center"

Foreground="{DynamicResource Text}"

Margin="0,10,0,0"/>

<TextBlock x:Name="txtCurImage"

TextAlignment="Left"

Foreground="{DynamicResource Text}"

Margin="0,10,0,10"

materialDesign:HintAssist.Hint="Выбранное изображение"/>

<Button x:Name="txbImage" Style="{DynamicResource ButtonStyle1}"

Foreground="{DynamicResource Text}"

materialDesign:HintAssist.Hint="Картинка"

Content="Изображение"

Click="AddImage\_Click"/>

<Button Style="{DynamicResource ButtonStyle1}"

Content="Добавить" x:Name="btnAdd"

Click="btnAdd\_Click" Margin="0,5,0,0"/>

<Button Style="{DynamicResource ButtonStyle1}"

Content="Удалить" x:Name="btnDel"

Click="btnDel\_Click" Margin="0,5,0,5"/>

<Button x:Name="btnEdit"

Click="btnEdit\_Click"

Content="Редактировать" Style="{DynamicResource ButtonStyle1}"

/>

</StackPanel>

<DataGrid Name="DbJew" AutoGenerateColumns="False" ScrollViewer.CanContentScroll="True"

ScrollViewer.VerticalScrollBarVisibility="Visible" Height="auto" Grid.Column="1"

SelectionChanged="DbJew\_SelectionChanged">

<DataGrid.Columns>

<DataGridTextColumn Header="Article"

Width="auto"

Binding="{Binding Article}" IsReadOnly="True" />

<DataGridTextColumn Width="120"

Header="Name"

Binding="{Binding Name}" IsReadOnly="True" ElementStyle="{StaticResource WrapText}" />

<DataGridTextColumn Header="Type"

Width="auto"

Binding="{Binding Type}" IsReadOnly="True"

/>

<DataGridTextColumn Header="Cost"

Width="auto"

Binding="{Binding Cost, StringFormat={}{0} ₽}" IsReadOnly="True"

/>

<DataGridTextColumn Header="Material"

Width="auto"

Binding="{Binding Material}" IsReadOnly="True"

/>

<DataGridTextColumn Header="Drag\_Crystal"

Width="103"

Binding="{Binding Drag\_Crystal}" IsReadOnly="True"

ElementStyle="{StaticResource WrapText}"/>

<DataGridTextColumn Header="Weight"

Width="auto"

Binding="{Binding Weight, StringFormat={}{0} g}"

IsReadOnly="True" />

<DataGridTemplateColumn Header="Images">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Image Width="70" Height="70">

<Image.Source>

<Binding Path="Images">

<Binding.TargetNullValue>

<ImageSource>/Resources/jew\_icon.jpg</ImageSource>

</Binding.TargetNullValue>

</Binding>

</Image.Source>

</Image>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

</Grid>

</Page>

Cs код:

using Microsoft.Win32;

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Media.Media3D;

using System.Windows.Navigation;

using System.Windows.Shapes;

using WpFStoreJew.Classes;

namespace WpFStoreJew.Pages

{

/// <summary>

/// Логика взаимодействия для PageSetting.xaml

/// </summary>

public partial class PageSetting : Page

{

string pathimages;

private Jewellery\_Z selectedJewellery;

public PageSetting()

{

InitializeComponent();

RefreshCmb();

DbJew.ItemsSource = StoreJewEntities.GetContext().Jewellery\_Z.OrderBy(x => x.Article).ToList();

}

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

{

OpenFileDialog = new OpenFileDialog();

openFileDialog.Filter = "Image files (\*.png;\*.jpeg;\*.jpg)|\*.png;\*.jpeg;\*.jpg|All files (\*.\*)|\*.\*";

if (openFileDialog.ShowDialog() == true)

{

string imagePath = openFileDialog.FileName;

// Получаем имя файла изображения

string imageFileName = System.IO.Path.GetFileName(imagePath);

string currentDirectory = System.IO.Directory.GetCurrentDirectory(); // Получаем текущую директорию

string projectDirectory = System.IO.Directory.GetParent(currentDirectory).Parent.FullName; // Получаем путь к главной папке проекта

string imagesDirectory = System.IO.Path.Combine(projectDirectory, "Images"); // Создаем путь к папке Images в главной папке проекта

// Создаем путь к файлу в папке Images

string targetPath = System.IO.Path.Combine(imagesDirectory, imageFileName);

// Создаем ссылку на файл в формате /images/название\_фото

string imageLink = $"/images/{imageFileName}";

try

{

// Копируем выбранный файл изображения в папку Images

File.Copy(imagePath, targetPath);

pathimages = imageLink;

MessageBox.Show("Фотография успешно добавлена в папку Photo.");

}

catch

{

pathimages = imageLink;

MessageBox.Show("Фото выбрано");

}

}

}

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

{

if (txbCost.Text == "" ||

txbName.Text == "" || txbWeight.Text == "" || cmbDrag\_Crystal.SelectedIndex == 0 || cmbMaterial.SelectedIndex == 0 || cmbType.SelectedIndex == 0)

{

MessageBox.Show("Пожалуйста, заполните все поля.", "Ошибка", MessageBoxButton.OK);

}

else

{

if (pathimages == null)

{

Jewellery\_Z jewelery = new Jewellery\_Z()

{

Name = txbName.Text,

Type = cmbType.Text,

Cost = double.Parse(txbCost.Text),

Material = cmbMaterial.Text,

Drag\_Crystal = cmbDrag\_Crystal.Text,

Weight = double.Parse(txbWeight.Text),

Images = null,

};

if (jewelery.Article == 0)

{

StoreJewEntities.GetContext().Jewellery\_Z.Add(jewelery);

}

}

else

{

Jewellery\_Z jewelery = new Jewellery\_Z()

{

Name = txbName.Text,

Type = cmbType.Text,

Cost = double.Parse(txbCost.Text),

Material = cmbMaterial.Text,

Drag\_Crystal = cmbDrag\_Crystal.Text,

Weight = double.Parse(txbWeight.Text),

Images = pathimages.ToString(),

};

if (jewelery.Article == 0)

{

StoreJewEntities.GetContext().Jewellery\_Z.Add(jewelery);

}

}

try

{

StoreJewEntities.GetContext().SaveChanges();

Refr();

RefreshCmb();

MessageBoxResult boxResult = MessageBox.Show("Изделие успешно добавлено. Добавить ещё?", "Сообщение",

MessageBoxButton.YesNo);

if (boxResult == MessageBoxResult.Yes)

{

txbCost.Clear();

txbName.Clear();

txbWeight.Clear();

cmbDrag\_Crystal.SelectedIndex = 0;

cmbMaterial.SelectedIndex = 0;

cmbType.SelectedIndex = 0;

}

}

catch (Exception ex)

{

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

}

}

}

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

{

var JewForRemoving = DbJew.SelectedItems.Cast<Jewellery\_Z>().ToList();

if (JewForRemoving.Count() != 0)

{

if (MessageBox.Show($"Удалить {JewForRemoving.Count()}" + " Изделий?",

"Внимание", MessageBoxButton.YesNo,

MessageBoxImage.Question) == MessageBoxResult.Yes)

try

{

StoreJewEntities.GetContext().Jewellery\_Z.RemoveRange(JewForRemoving);

StoreJewEntities.GetContext().SaveChanges();

MessageBox.Show("Данные удалены");

Refr();

}

catch (Exception ex)

{

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

}

}

else

{

MessageBox.Show("Вы не выбрали кого удалить");

}

}

public void Refr()

{

DbJew.ItemsSource = StoreJewEntities.GetContext().Jewellery\_Z.OrderBy(x => x.Article).ToList();

}

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

{

selectedJewellery = DbJew.SelectedItem as Jewellery\_Z;

if (selectedJewellery != null)

{

txbName.Text = selectedJewellery.Name;

txbCost.Text = selectedJewellery.Cost.ToString();

txbWeight.Text = selectedJewellery.Weight.ToString();

cmbDrag\_Crystal.SelectedValue = selectedJewellery.Drag\_Crystal;

cmbMaterial.SelectedValue = selectedJewellery.Material;

cmbType.SelectedValue = selectedJewellery.Type;

txtCurImage.Text = selectedJewellery.Images;

}

else

{

txbCost.Clear();

txbName.Clear();

txbWeight.Clear();

cmbDrag\_Crystal.SelectedIndex = 0;

cmbMaterial.SelectedIndex = 0;

cmbType.SelectedIndex = 0;

txtCurImage.Text = "";

pathimages = null;

}

}

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

{

if (selectedJewellery == null)

{

MessageBox.Show("Выберите элемент для редактирования", "Предупреждение");

return;

}

if (txbCost.Text == "" ||

txbName.Text == "" ||

txbWeight.Text == "" ||

cmbDrag\_Crystal.SelectedIndex == 0 ||

cmbMaterial.SelectedIndex == 0 ||

cmbType.SelectedIndex == 0 )

{

MessageBox.Show("Заполните все поля", "Предупреждение");

}

else

{

{

selectedJewellery.Name = txbName.Text;

selectedJewellery.Type = cmbType.Text;

selectedJewellery.Cost = double.Parse(txbCost.Text);

selectedJewellery.Material = cmbMaterial.Text;

selectedJewellery.Drag\_Crystal = cmbDrag\_Crystal.Text;

selectedJewellery.Weight = double.Parse(txbWeight.Text);

selectedJewellery.Images = pathimages;

};

try

{

StoreJewEntities.GetContext().SaveChanges();

MessageBox.Show("Изменения успешно внесены");

Refr();

RefreshCmb();

}

catch (Exception ex)

{

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

}

}

}

private void RefreshCmb()

{

var alltype = StoreJewEntities.GetContext().Jewellery\_Z.Select(x => x.Type).Distinct().ToList();

alltype.Insert(0, "");

cmbType.ItemsSource = alltype;

var allmat = StoreJewEntities.GetContext().Jewellery\_Z.Select(x => x.Material).Distinct().ToList();

allmat.Insert(0, "");

cmbMaterial.ItemsSource = allmat;

var alldrag = StoreJewEntities.GetContext().Jewellery\_Z.Select(x => x.Drag\_Crystal).Distinct().ToList();

alldrag.Insert(0, "");

cmbDrag\_Crystal.ItemsSource = alldrag;

}

}

}

Код страницы «PageStore» (см. рисунок 16)

Xaml код:

<Page x:Class="WpFStoreJew.Pages.PageStore"

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

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

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

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

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

xmlns:local="clr-namespace:WpFStoreJew.Pages"

mc:Ignorable="d"

d:DesignHeight="750" d:DesignWidth="1000"

Title="PageStore">

<Page.Resources>

</Page.Resources>

<Grid Background="{DynamicResource SecondCol}">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="0.3\*"/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<ListView Grid.Column="1" Name="listView" Margin="10" ScrollViewer.HorizontalScrollBarVisibility="Disabled" HorizontalContentAlignment="Center">

<ListView.ItemsPanel>

<ItemsPanelTemplate>

<WrapPanel Orientation="Horizontal" HorizontalAlignment="Left" />

</ItemsPanelTemplate>

</ListView.ItemsPanel>

<ListView.ItemTemplate>

<DataTemplate>

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="120"/>

<RowDefinition Height="40"/>

<RowDefinition Height="auto"/>

<RowDefinition Height="auto"/>

</Grid.RowDefinitions>

<Image Width="125" Grid.Row="0" Stretch="Uniform" HorizontalAlignment="Center">

<Image.Source>

<Binding Path="Images">

<Binding.TargetNullValue>

<ImageSource>/Resources/jew\_icon.jpg</ImageSource>

</Binding.TargetNullValue>

</Binding>

</Image.Source>

</Image>

<TextBlock Text="{Binding Name}" VerticalAlignment="Center" TextAlignment="Center" Width="150" Height="35" Grid.Row="1" TextWrapping="Wrap" Foreground="{DynamicResource Text}"/>

<Button Content="Купить" Click="btnBuy\_Click" x:Name="btnBuy" Height="25" Width="100" VerticalAlignment="Center" Grid.Row="3"/>

<TextBlock Text="{Binding Cost, StringFormat={}{0} ₽}" VerticalAlignment="Center" TextAlignment="Center" Width="100" Height="35" Grid.Row="2" FontSize="15" Foreground="{DynamicResource Text}"/>

</Grid>

</DataTemplate>

</ListView.ItemTemplate>

</ListView>

<StackPanel x:Name="stkFilt" Margin="10,30,0,0">

<TextBlock Text="Поиск по названию"

Style="{DynamicResource text}"

x:Name="txtSearch"/>

<TextBox x:Name="TxtBoxSearch"

materialDesign:HintAssist.Hint="Название"

TextChanged="TxtSearch\_TextChanged"

Foreground="{DynamicResource Text}"/>

<TextBlock Text="Тип Изделия"

Style="{DynamicResource text}"

x:Name="txtType"/>

<ComboBox

x:Name="CmbType"

SelectionChanged="CmbType\_SelectionChanged"

materialDesign:HintAssist.Hint="Изделие"

Background="{DynamicResource SecondCol}"

Foreground="{DynamicResource TextCmb}"/>

<TextBlock Text="Стоимость"

Style="{DynamicResource text}"

Name="txtCost"

Margin="0,10,0,0"/>

<RadioButton x:Name="RbtCostMin"

Content="По возрастанию цены"

Checked="ChkCostMin\_Checked"

FontSize="14"

Foreground="{DynamicResource Text}"/>

<RadioButton x:Name="RbtCostMax"

Content="По убыванию цены"

Checked="ChkCostMax\_Checked"

FontSize="14"

Foreground="{DynamicResource Text}"/>

<TextBlock Text="Тип Материала"

Style="{DynamicResource text}"

x:Name="TxtMaterial"

Margin="0,10,0,0"/>

<ComboBox materialDesign:HintAssist.Hint="Материал"

x:Name="CmbMaterial"

SelectionChanged="CmbMaterial\_SelectionChanged"

Foreground="{DynamicResource TextCmb}"/>

<CheckBox Height="35"

x:Name="ChbDrag\_Cryst"

FontSize="15"

Margin="0,10,0,0"

Content="Драгоценные камни"

Checked="ChbDrag\_Cryst\_Checked"

Unchecked="ChbDrag\_Cryst\_Unchecked"

Foreground="{DynamicResource Text}"/>

<TextBlock x:Name="txtCount" FontSize="15" FontStyle="Italic" Margin="0,10,0,0" Foreground="{DynamicResource Text}"/>

</StackPanel>

</Grid>

</Page>

Cs код:

using MaterialDesignColors;

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using WpFStoreJew.Classes;

namespace WpFStoreJew.Pages

{

/// <summary>

/// Логика взаимодействия для PageStore.xaml

/// </summary>

public partial class PageStore : Page

{

private int ID;

public PageStore(Customers\_Z level)

{

InitializeComponent();

ID = level.ID;

var currentJew = StoreJewEntities.GetContext().Jewellery\_Z.OrderBy(x=>x.Cost).ToList();

listView.ItemsSource = currentJew;

var listType =

StoreJewEntities.GetContext().Jewellery\_Z.Select(x => x.Type).Distinct().ToList();

CmbType.Items.Add("Все Типы");

foreach (var item in listType)

{

CmbType.Items.Add(item);

}

var listMaterial =

StoreJewEntities.GetContext().Jewellery\_Z.Select(x => x.Material).Distinct().ToList();

CmbMaterial.Items.Add("Все Материалы");

foreach (var item in listMaterial)

{

CmbMaterial.Items.Add(item);

}

UbdateDb();

}

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

{

UbdateDb();

}

private void ChkCostMin\_Checked(object sender, RoutedEventArgs e)

{

UbdateDb();

}

private void ChkCostMax\_Checked(object sender, RoutedEventArgs e)

{

UbdateDb();

}

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

{

UbdateDb();

}

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

{

Customers\_Z cus = new Customers\_Z();

Button = sender as Button;

if (button != null)

{

Jewellery\_Z Jew = (button.DataContext as Jewellery\_Z);

using (var dbContext = new StoreJewEntities())

{

Orders\_Z existingOrder = dbContext.Orders\_Z.FirstOrDefault(o => o.Article\_Jew == Jew.Article && o.Id\_Customer == ID);

if (existingOrder != null)

{

existingOrder.Quantity++;

existingOrder.Sum\_Cost = existingOrder.Quantity \* (float)Jew.Cost;

}

else

{

Orders\_Z newOrder = new Orders\_Z

{

Article\_Jew = Jew.Article,

Name\_Jew = Jew.Name,

Quantity = 1,

Sum\_Cost = (float)Jew.Cost,

Date\_Order = DateTime.Now,

Id\_Customer = ID

};

dbContext.Orders\_Z.Add(newOrder);

}

dbContext.SaveChanges();

}

}

}

private void ChbDrag\_Cryst\_Checked(object sender, RoutedEventArgs e)

{

UbdateDb();

}

private void ChbDrag\_Cryst\_Unchecked(object sender, RoutedEventArgs e)

{

UbdateDb();

}

private void TxtSearch\_TextChanged(object sender, TextChangedEventArgs e)

{

UbdateDb();

}

private void UbdateDb()

{

var currentJew = StoreJewEntities.GetContext().Jewellery\_Z.ToList();

if (TxtBoxSearch.Text.Length > 0)

currentJew = currentJew.Where(X => X.Name.ToLower().ToString().Contains(TxtBoxSearch.Text.ToLower())).ToList();

if (CmbMaterial.SelectedIndex > 0)

currentJew = currentJew.Where(x => x.Material == (CmbMaterial.SelectedValue.ToString())).ToList();

if (CmbType.SelectedIndex > 0)

currentJew = currentJew.Where(x => x.Type == (CmbType.SelectedValue.ToString())).ToList();

if (RbtCostMax.IsChecked.Value)

currentJew = currentJew.OrderByDescending(x => x.Cost).ToList();

if (RbtCostMin.IsChecked.Value)

currentJew = currentJew.OrderBy(x => x.Cost).ToList();

if (ChbDrag\_Cryst.IsChecked.Value)

currentJew= currentJew.Where(x => x.Drag\_Crystal != "Нет").ToList();

listView.ItemsSource = currentJew;

int Count = listView.Items.Count;

string Counttext = "Найдено " + Count.ToString() + " Позиций" ; // Количество записей после

txtCount.Text = Counttext.ToString();

}

}

}

Код словаря стилей «Dark» (см. рисунок 26)

Xaml код:

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

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

<SolidColorBrush Color="#12372A" x:Key="Backdrop"/>

<SolidColorBrush Color="#E3E1D9" x:Key="Text"/>

<SolidColorBrush Color="#FF9BD2" x:Key="TextCmb"/>

<SolidColorBrush Color="#436850" x:Key="SecondCol"/>

<Style TargetType="Button" x:Key="btnMenu">

<Setter Property="FontSize" Value="26"/>

<Setter Property="Height" Value="50"/>

<Setter Property="Margin" Value="0,5,0,0"/>

<Setter Property="Background" Value="Transparent"/>

<Setter Property="BorderThickness" Value="0"/>

</Style>

<Style TargetType="ComboBox" x:Key="cmbFiltr">

<Setter Property="Height" Value="35"/>

<Setter Property="FontSize" Value="20"/>

<Setter Property="Margin" Value="0,5,0,5"/>

</Style>

<Style TargetType="TextBlock" x:Key="text">

<Setter Property="FontSize" Value="18"/>

<Setter Property="Foreground" Value="#E3E1D9"/>

</Style>

<Style TargetType="Image" x:Key="Profile">

<Setter Property="Source" Value="/Resources/Profile\_Light.png"/>

</Style>

<Style TargetType="Image" x:Key="Home">

<Setter Property="Source" Value="/Resources/Home\_Light.png"/>

</Style>

<Style TargetType="Image" x:Key="Store">

<Setter Property="Source" Value="/Resources/Store\_Light.png"/>

</Style>

<Style TargetType="Image" x:Key="Cart">

<Setter Property="Source" Value="/Resources/Cart\_Light.png"/>

</Style>

<Style TargetType="Image" x:Key="Setting">

<Setter Property="Source" Value="/Resources/Setting\_Light.png"/>

</Style>

<Style TargetType="Image" x:Key="Exit">

<Setter Property="Source" Value="/Resources/Exit\_Light.png"/>

</Style>

</ResourceDictionary>

Код словаря стилей «Light» (см. рисунок 25)

Xaml код:

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

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

<SolidColorBrush Color="#99BC85" x:Key="Backdrop"/>

<SolidColorBrush Color="Black" x:Key="Text"/>

<SolidColorBrush Color="Black" x:Key="TextCmb"/>

<SolidColorBrush Color="#BFD8AF" x:Key="SecondCol"/>

<Style TargetType="Button" x:Key="btnMenu">

<Setter Property="FontSize" Value="26"/>

<Setter Property="Height" Value="50"/>

<Setter Property="Margin" Value="0,5,0,0"/>

<Setter Property="Background" Value="Transparent"/>

<Setter Property="BorderThickness" Value="0"/>

</Style>

<Style TargetType="ComboBox" x:Key="cmbFiltr">

<Setter Property="Height" Value="35"/>

<Setter Property="FontSize" Value="20"/>

<Setter Property="Margin" Value="0,5,0,5"/>

</Style>

<Style TargetType="TextBlock" x:Key="text">

<Setter Property="FontSize" Value="18"/>

<Setter Property="Foreground" Value="Black"/>

</Style>

<Style TargetType="Image" x:Key="Home">

<Setter Property="Source" Value="/Resources/Home.png"/>

</Style>

<Style TargetType="Image" x:Key="Store">

<Setter Property="Source" Value="/Resources/Store.png"/>

</Style>

<Style TargetType="Image" x:Key="Cart">

<Setter Property="Source" Value="/Resources/Cart.png"/>

</Style>

<Style TargetType="Image" x:Key="Profile">

<Setter Property="Source" Value="/Resources/Profile.png"/>

</Style>

<Style TargetType="Image" x:Key="Setting">

<Setter Property="Source" Value="/Resources/Setting.png"/>

</Style>

<Style TargetType="Image" x:Key="Exit">

<Setter Property="Source" Value="/Resources/Exit.png"/>

</Style>

</ResourceDictionary>

Код класса «AppThemes»

Cs код:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

namespace WpFStoreJew.Classes

{

internal class AppThemes

{

public static void ChangeThemes(Uri themeuri)

{

ResourceDictionary Theme = new ResourceDictionary() { Source = themeuri};

App.Current.Resources.Clear();

App.Current.Resources.MergedDictionaries.Add(Theme);

}

}

}