Создать бД запросом:

CREATE TABLE PRODUCT(

ID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

Name\_ NVARCHAR(50) NOT NULL,

Author NVARCHAR(50) NOT NULL,

Price MONEY,

Discount FLOAT,

Stock INT,

Desciption NVARCHAR(100),

PathToImage NVARCHAR(100),

)

CREATE TABLE ROLES(

ID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

ROLENAME NVARCHAR(50) NOT NULL

)

CREATE TABLE LOGINS(

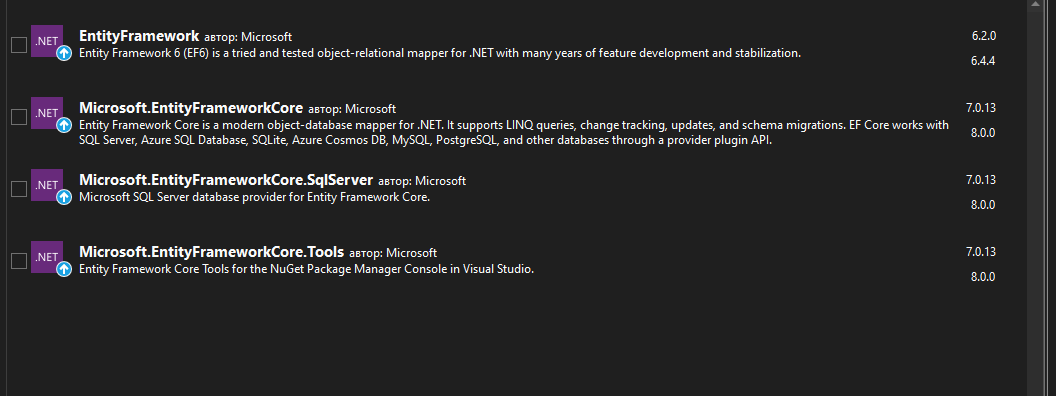
LOGIN\_ NVARCHAR(50) NOT NULL PRIMARY KEY,

PASSWORD\_ NVARCHAR(50) NOT NULL,

ROLE\_ INT NOT NULL REFERENCES ROLES(ID)

)

После этого установить через Nuget следующие пакеты



После этого в консоль диспетчера пакетов ввести команду

Scaffold-DbContext "Server=ИМЯСЕРВЕРА;Database=ИМЯБАЗЫДАННЫХ;Trusted\_Connection=True;TrustServerCertificate=True" Microsoft.EntityFrameworkCore.SqlServer

После создания контекста базы данных (файл ИМЯБАЗЫДАННЫХContext.cs) создать три окна – основное,окно авторизации и окно корзины, а так же 2 UserControl – один для товаров в каталоге, другой для товаров в корзине

# UserControl для продукта в каталоге создается со следующим содержанием Grid(Xaml)

<Grid x:Name="MainGrid">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="1\*"/>

<ColumnDefinition Width="1\*"/>

<ColumnDefinition Width="1\*"/>

<ColumnDefinition Width="1\*"/>

</Grid.ColumnDefinitions>

<Image x:Name="ProductImage" Grid.Column="0" Margin="5"/>

<Label x:Name="ProductNameLabel" Grid.Column="1" Margin="5"/>

<Label x:Name="ProductPriceLabel" Grid.Column="2" Margin="5"/>

<Button x:Name="ProductDeleteButton" Content="Удалить" Click="ProductDeleteButton\_Click" Grid.Column="4" Margin="5" IsEnabled="False"/>

<Button x:Name="ProductAddButton" Content="Добавить в корзину" Click="ProductAddButton\_Click" Grid.Column="3" Margin="5"/>

</Grid>

И следующим содержанием класса(.CS)

private Product ThisProduct;

private MainWindow EvokingWindow;

public ProductVisual(MainWindow MainWindow, int productId,bool admined)

{

InitializeComponent();

BookShopContext dbCont = new BookShopContext();

ThisProduct = dbCont.Products.Find(productId);//поиск в контексте БД текущей записи товара по ID переданному в конструкторе

EvokingWindow = MainWindow;//окно которое вызвало UserControl

ProductImage.Source = new BitmapImage(new Uri(ThisProduct.PathToImage, UriKind.RelativeOrAbsolute));

ProductNameLabel.Content = ThisProduct.Name;

ProductPriceLabel.Content = ThisProduct.Price;

if (admined) {//создание дополнительной колонки для отображения кнопки удаления

ColumnDefinition col = new ColumnDefinition();

col.Width = new GridLength(0.5, GridUnitType.Star);

MainGrid.ColumnDefinitions.Add(col);

ProductDeleteButton.IsEnabled = true;

}

if (ThisProduct.Stock < 1)//если колво товара меньше 1 то товар нельзя добавить в корзину и он окраивается в серый

{

ProductAddButton.IsEnabled = false;

this.Background = Brushes.Gray;

}

}

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

{//обработчик кнопки добавления товара в корзину

if (EvokingWindow.Basket.ContainsKey(ThisProduct.Id))

EvokingWindow.Basket[ThisProduct.Id] += 1;

else EvokingWindow.Basket.Add(ThisProduct.Id, 1);

BasketWindow.UpdateBasket();//метод обновления корзины

EvokingWindow.TurnOnBasketButton();//включить кнопку корзины в основном окне

MessageBox.Show("Товар успешно добавлен!");

}

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

{// обработчик кнопки удаления для адмнистратора

BookShopContext db = new BookShopContext();

db.Products.Remove(ThisProduct);//запрос в БД на удаление текущей записи

db.SaveChanges();

EvokingWindow.ProductPanel.Children.Remove(this);

}

# UserControl для товара в корзине создается со следующим содержанием разметки

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="1.5\*"/><!--картинка-->

<ColumnDefinition Width="1.5\*"/><!--название-->

<ColumnDefinition Width="1\*"/><!--колво-->

<ColumnDefinition Width="1\*"/><!--стоимость( + цена за штуку)-->

<ColumnDefinition Width="1\*"/><!--кнопка удаления-->

</Grid.ColumnDefinitions>

<Image x:Name="BasketProductImage" Grid.Column="0" Margin="5 5 5 5"/>

<Label x:Name="BasketProductNameLabel" Grid.Column="1" Margin="5 5 5 5"/>

<Grid Grid.Column="2" Margin="20" Height="50">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="0.2\*"/>

<ColumnDefinition Width="0.4\*"/>

<ColumnDefinition Width="0.2\*"/>

</Grid.ColumnDefinitions>

<Button x:Name="ProductCountUpButton" Content="+" Grid.Column="2" Margin="1 1 1 1" Click="ProductCountUpButton\_Click"/>

<Button x:Name="ProductCountDownButton" Content="-" Grid.Column="0" Margin="1 1 1 1" Click="ProductCountDownButton\_Click"/>

<TextBox x:Name="ProductCountTextBox" Grid.Column="1" Margin="1 1 1 1" IsEnabled="False" />

</Grid>

<Grid Grid.Column="3" Margin="5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<Label Grid.Row="0" x:Name="BasketProductSumLabel" Grid.Column="3"/>

<Label Grid.Row="1" x:Name="BasketProductPriceLabel" Grid.Column="3"/>

<Label Grid.Row="2" x:Name="BasketProductDiscountLabel" Grid.Column="3"/>

</Grid>

<Button x:Name="DeleteProductButton" Click="DeleteProductButton\_Click" Margin="5" Grid.Column="4" Content="Delete"/>

</Grid>

И следующим содержанием класса(.CS)

private BasketWindow EvokingWindow;

private int Quantity;

Product ThisProduct;

public BasketProductVisual(BasketWindow BasketWindow, int productId, int quantity)

{

InitializeComponent();

BookShopContext dbCont = new BookShopContext();

ThisProduct = dbCont.Products.Find(productId); );//поиск в контексте БД текущей записи товара по ID переданному в конструкторе

EvokingWindow = BasketWindow; //окно которое вызвало UserControl

if (ThisProduct.PathToImage != null) {

BasketProductImage.Source = new BitmapImage(new Uri(ThisProduct.PathToImage, UriKind.RelativeOrAbsolute));

}

BasketProductNameLabel.Content = ThisProduct.Name;

BasketProductSumLabel.Content = new string($"{ThisProduct.Price \* quantity \* (1-(Decimal)ThisProduct.Discount) }");//подсчет стоимости продуктов

BasketProductPriceLabel.Content = new string($"{ThisProduct.Price} за ед.");

BasketProductDiscountLabel.Content = new string($"Скидка {ThisProduct.Discount \* 100}%");

ProductCountTextBox.Text = quantity.ToString();

Quantity = quantity;

}

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

{//обработчик кнопки увелечения кол-ва товара

Quantity++;

EvokingWindow.Basket[ThisProduct.Id]++;

EvokingWindow.RefreshBasket();

UpdatePrice();

}

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

{//обработчик кнопки уменьшение кол-ва товара

Quantity--;

EvokingWindow.Basket[ThisProduct.Id]--;

EvokingWindow.RefreshBasket();

UpdatePrice();

}

private void UpdatePrice() {

//обновление цены

BasketProductPriceLabel.Content = ThisProduct.Price \* Quantity;

ProductCountTextBox.Text = Quantity.ToString();

}

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

{//обработчик кнопки удаление товара из корзины

EvokingWindow.Basket.Remove(ThisProduct.Id);

UpdatePrice();

EvokingWindow.RefreshBasket();

}

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

{//обработчик изменения текста кол-ва товара

Quantity = Convert.ToInt32(ProductCountTextBox.Text);

UpdatePrice();

EvokingWindow.RefreshBasket();

}

# Окно BasketWindow вызываемое кнопкой “Корзина”

<Window x:Class="WpfApp3.BasketWindow"

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:WpfApp3"

mc:Ignorable="d"

Title="BasketWindow" Height="450" Width="800" Closing="BasketWindow\_Closing">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="0.2\*" />

<RowDefinition Height="1\*" />

</Grid.RowDefinitions>

<Grid Grid.Row="0">

<Grid.ColumnDefinitions>

<ColumnDefinition/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<Label Content="Корзина" Grid.Row="0" Grid.Column="0" Grid.RowSpan="2" VerticalAlignment="Center" HorizontalAlignment="Left"/>><!--корзина-->

<Label Content="Сумма" Grid.Row="0" Grid.Column="1" x:Name="BasketWindowSumLabel"/><!—итоговая сумма-->

<Label Content="Скидка" Grid.Row="1" Grid.Column="1" x:Name="BasketWindowDiscountLabel"/><!— сумма скидки-->

</Grid>

<ScrollViewer Grid.Row="1"><!— контейнер для товаров-->

<StackPanel x:Name="BasketProductPanel">

</StackPanel>

</ScrollViewer>

</Grid>

</Window>

И следующим содержанием класса(.CS)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows;

namespace WpfApp3

{

public partial class BasketWindow : Window

{

public MainWindow EvokingWindow;

public Dictionary<int, int> Basket = new();

private int Sum = 0;

private int Discount = 0;

public BasketWindow(MainWindow evokingWindow,Dictionary<int, int> basket)

{

InitializeComponent();

EvokingWindow = evokingWindow;

Basket = basket;

CreateBasket();

}

public static void UpdateBasket()

{//метод обновления корзины

foreach (BasketWindow window in Application.Current.Windows.OfType<BasketWindow>())

window.RefreshBasket();

}

private void CreateBasket() {

//метод подсчета итоговой суммы и скидки

Sum = 0;Discount = 0;

BookShopContext dbCont = new BookShopContext();

foreach (var item in dbCont.Products)

if (Basket.ContainsKey(item.Id)) {

BasketProductPanel.Children.Add(new BasketProductVisual(this, item.Id, Basket[item.Id]));

Sum += (int)item.Price \* Basket[item.Id];

Discount += Convert.ToInt32(Basket[item.Id] \* ((decimal)item.Price \* (decimal)item.Discount));

}

BasketWindowSumLabel.Content = new string($"Сумма: {Sum}");

BasketWindowDiscountLabel.Content = new string($"Общая скидка: {Discount}");

}

public void RefreshBasket() {

foreach (var item in Basket)

if (item.Value < 1)

Basket.Remove(item.Key);

BasketProductPanel.Children.Clear();

CreateBasket();

}

private void BasketWindow\_Closing(object sender, System.ComponentModel.CancelEventArgs e)

//обработчик при закрытии окна

{

EvokingWindow.IsBasketWindowActive = false;

EvokingWindow.TurnOnBasketButton();

}

}

}

# Окно авторизации пользователя

<Window x:Class="WpfApp3.AuthWindow"

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:WpfApp3"

mc:Ignorable="d"

Title="AuthWindow" Height="450" Width="600">

<Grid>

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<Label Content="Login" Grid.Row="0" Margin="5" FontSize="30"/>

<Label Grid.Row="2" Margin="5" Content = "Password" FontSize="30"/>

<TextBox x:Name="LoginTextBox" Grid.Row="1" Margin="5" FontSize="30"/><!—текстбокс ввода логина-->

<TextBox x:Name="PasswordTextBox" Grid.Row="3" Margin="5" FontSize="30"/><!—тектсбокс ввода пароля-->

<Button x:Name="Authorize" Grid.Row="4" Margin="20" Content="Авторизоваться" FontSize="30" Click="Authorize\_Click"/><!—кнопка авторизации-->

</Grid>

</Window>

И следующим содержанием класса(.CS)

using System.Linq;

using System.Windows;

namespace WpfApp3

{

public partial class AuthWindow : Window

{

private MainWindow EvokingWindow;

public AuthWindow(MainWindow MainWindow)

{

InitializeComponent();

this.EvokingWindow = MainWindow;

}

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

{//обработчик кнопки авторизации и проверки логина/пароля

BookShopContext db = new BookShopContext();

bool auth = false,admined = false;

Login log = new Login();

foreach (Login item in db.Logins)

{

if (item.Login1 == this.LoginTextBox.Text && item.Password == this.PasswordTextBox.Text)

{

auth = true;

log = item;

break;

}

}

if (auth && log.Role == db.Roles.First(x => x.Rolename == "ADMIN").Id) {

admined = true;

}

EvokingWindow.IsAdmined = admined;

EvokingWindow.IsAuth = auth;

if (auth)

{

MessageBox.Show("авторизация успешна");

EvokingWindow.UpdateAuth();

this.Close();

}

else {

MessageBox.Show("логин или пароль не найдены");

}

}

}

}

# Основное окно(с каталогом товаров) создается со следующей разметкой

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height=".3\*"/>

<RowDefinition Height="2.1\*"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="0.2\*"/>

<ColumnDefinition Width="0.5\*"/>

<ColumnDefinition Width="0.1\*"/>

<ColumnDefinition Width="0.1\*"/>

</Grid.ColumnDefinitions>

<Button Grid.Column = "3" x:Name="BasketButton" Content="Корзина" IsEnabled="False" Margin="5" Click="BasketButton\_Click"/>

<Button Grid.Column = "2" x:Name="AuthButton" Content="Авторизация" Margin="5" Click="AuthButton\_Click"/>

</Grid>

<ScrollViewer Grid.Row="1">

<StackPanel x:Name="ProductPanel">

</StackPanel>

</ScrollViewer>

</Grid>

И со следующим кодом(Xaml.cs)

namespace WpfApp3

{

public partial class MainWindow : Window

{

public bool IsBasketWindowActive = false;

public bool IsAdmined = false;

public bool IsAuth = false;

public Login? authLogin = null;//ссылка на запись логина при авторизации

public Dictionary<int, int> Basket = new();

public MainWindow()

{

InitializeComponent();

UpdateProductsList();

}

public void UpdateProductsList()//пересоздать каталог

{

ProductPanel.Children.Clear();

BookShopContext dbCont = new BookShopContext();

foreach (var item in dbCont.Products)

ProductPanel.Children.Add(new ProductVisual(this, item.Id, IsAdmined));

}

public void UpdateAuth() {//обновить контент кнопки авторизации

AuthButton.Content = "Выйти из аккаунта";

UpdateProductsList();

}

public void TurnOnBasketButton() { //включение кнопки корзина

if (!IsBasketWindowActive)

BasketButton.IsEnabled = true;

}

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

{//при нажатии кнопки корзины создание окна с корзиной

IsBasketWindowActive = true;

BasketButton.IsEnabled = false;

BasketWindow basketWindow = new BasketWindow(this,Basket);

basketWindow.Show();

BasketWindow.UpdateBasket();

}

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

{

if (!IsAuth)

{

AuthWindow authWindow = new AuthWindow(this);

authWindow.Show();

}

else {

AuthButton.Content = "Авторизация";

IsAdmined = false;

IsAuth = false;

authLogin = null;

UpdateProductsList();

}

}

}

}