**НАПИСАНИЕ ПРОГРАММЫ**

**App.axaml:**

<Application xmlns="https://github.com/avaloniaui"

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

x:Class="DemoApp.App"

xmlns:local="using:DemoApp"

RequestedThemeVariant="Default">

<!-- "Default" ThemeVariant follows system theme variant. "Dark" or "Light" are other available options. -->

<Application.DataTemplates>

<local:ViewLocator />

</Application.DataTemplates>

<Application.Styles>

<FluentTheme />

<Style Selector="TextBlock.h1">

<Setter Property="FontSize" Value="24" />

<Setter Property="FontWeight" Value="Bold" />

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

</Style>

<Style Selector="StackPanel.main">

<Setter Property="Background" Value="#F4E8D3" />

</Style>

<Style Selector="Button">

<Setter Property="Background" Value="#67BA80" />

</Style>

</Application.Styles>

</Application>

**EditPartnerViewModel.cs:**

using System.Collections.ObjectModel;

using System.Linq;

using System.Threading.Tasks;

using CommunityToolkit.Mvvm.ComponentModel;

using CommunityToolkit.Mvvm.Input;

using DemoApp.Entities;

using MsBox.Avalonia;

using MsBox.Avalonia.Enums;

namespace DemoApp.ViewModels

{

public partial class EditPartnerViewModel : ViewModelBase

{

private readonly DbDemoFdbContext \_context;

private readonly PartnersViewModel? \_parentViewModel;

private readonly Partner \_partnerEntity;

[ObservableProperty]

private string? \_partnerName;

[ObservableProperty]

private string? \_partnerInn;

[ObservableProperty]

private string? \_director;

[ObservableProperty]

private string? \_directorPhone;

[ObservableProperty]

private string? \_directorMail;

[ObservableProperty]

private int? \_partnerRating;

[ObservableProperty]

private int? \_selectedPartnerType;

[ObservableProperty]

private ObservableCollection<PartnerType> \_partnerTypes = new();

[ObservableProperty]

private string? \_errorMessage;

public EditPartnerViewModel(DbDemoFdbContext context, Partner? partner = null, PartnersViewModel? parentViewModel = null)

{

\_context = context;

\_parentViewModel = parentViewModel;

\_partnerEntity = partner ?? new Partner();

LoadPartnerTypes();

LoadPartnerData();

}

private void LoadPartnerTypes()

{

var types = \_context.PartnerTypes.ToList();

PartnerTypes = new ObservableCollection<PartnerType>(types);

}

private void LoadPartnerData()

{

PartnerName = \_partnerEntity.PartnerName;

PartnerInn = \_partnerEntity.PartnerInn;

Director = \_partnerEntity.Director;

DirectorPhone = \_partnerEntity.DirectorPhone;

DirectorMail = \_partnerEntity.DirectorMail;

PartnerRating = \_partnerEntity.PartnerRating;

SelectedPartnerType = \_partnerEntity.PartnerType;

}

private bool ValidateData()

{

if (string.IsNullOrWhiteSpace(PartnerName))

{

ErrorMessage = "Название партнера обязательно для заполнения";

return false;

}

if (!ValidationHelper.IsValidInn(PartnerInn))

{

ErrorMessage = "ИНН должен содержать 10 или 12 цифр";

return false;

}

if (string.IsNullOrWhiteSpace(Director))

{

ErrorMessage = "Имя директора обязательно для заполнения";

return false;

}

if (!ValidationHelper.IsValidPhoneNumber(DirectorPhone))

{

ErrorMessage = "Неверный формат номера телефона. Пример: +7(999)999-99-99";

return false;

}

if (!ValidationHelper.IsValidEmail(DirectorMail))

{

ErrorMessage = "Неверный формат электронной почты";

return false;

}

if (!ValidationHelper.IsValidRating(PartnerRating))

{

ErrorMessage = "Рейтинг должен быть от 0 до 100";

return false;

}

if (SelectedPartnerType == null)

{

ErrorMessage = "Выберите тип партнера";

return false;

}

ErrorMessage = null;

return true;

}

[RelayCommand]

private async Task SaveAsync()

{

if (!ValidateData()) {

var box = MessageBoxManager

.GetMessageBoxStandard("Error", ErrorMessage,

ButtonEnum.Ok);

var result = await box.ShowAsync();

return;}

\_partnerEntity.PartnerName = PartnerName;

\_partnerEntity.PartnerInn = PartnerInn;

\_partnerEntity.Director = Director;

\_partnerEntity.DirectorPhone = DirectorPhone;

\_partnerEntity.DirectorMail = DirectorMail;

\_partnerEntity.PartnerRating = PartnerRating;

\_partnerEntity.PartnerType = SelectedPartnerType;

if (\_partnerEntity.PartnerId == 0)

{

\_context.Partners.Add(\_partnerEntity); // новая запись

}

else

{

\_context.Partners.Update(\_partnerEntity); // обновление существующей

}

\_context.SaveChanges();

\_parentViewModel?.LoadPartners();

Cancel();

}

[RelayCommand]

private void Cancel()

{

\_parentViewModel?.\_mainViewModel.ShowPartnersList();

}

}

}

**PartnerSalesView.axaml:**

<UserControl xmlns="https://github.com/avaloniaui"

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

xmlns:vm="using:DemoApp.ViewModels"

x:Class="DemoApp.Views.PartnerSalesView"

x:DataType="vm:PartnerSalesViewModel">

<Grid Margin="20">

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<StackPanel Grid.Row="0" Spacing="10">

<Button Content="← Вернуться к списку"

Command="{Binding BackCommand}"

HorizontalAlignment="Left"/>

<TextBlock Text="{Binding Title}"

FontSize="24"

FontWeight="Bold"/>

</StackPanel>

<StackPanel Grid.Row="1" Spacing="5" Margin="0,10">

<TextBlock Text="{Binding SummaryInfo}"

FontSize="16"/>

<TextBlock Text="{Binding DiscountInfo}"

FontSize="16"

Foreground="Green"/>

</StackPanel>

<Border Grid.Row="2"

BorderBrush="{StaticResource SystemBaseMediumColor}"

BorderThickness="1"

CornerRadius="4">

<Grid>

<ScrollViewer IsVisible="{Binding HasSales}">

<ItemsControl ItemsSource="{Binding Sales}">

<ItemsControl.ItemTemplate>

<DataTemplate>

<Border Margin="5"

Padding="10"

BorderBrush="{StaticResource SystemBaseLowColor}"

BorderThickness="1"

CornerRadius="4">

<Grid ColumnDefinitions="2\*,3\*,2\*,2\*">

<StackPanel Grid.Column="0" Margin="5">

<TextBlock Text="Дата продажи:"

Foreground="{StaticResource SystemBaseMediumColor}"/>

<TextBlock Text="{Binding SaleDateFormatted}"

FontWeight="SemiBold"/>

</StackPanel>

<StackPanel Grid.Column="1" Margin="5">

<TextBlock Text="Наименование:"

Foreground="{StaticResource SystemBaseMediumColor}"/>

<TextBlock Text="{Binding ProductName}"

FontWeight="SemiBold"/>

</StackPanel>

<StackPanel Grid.Column="2" Margin="5">

<TextBlock Text="Количество:"

Foreground="{StaticResource SystemBaseMediumColor}"/>

<TextBlock Text="{Binding \_sale.ProductCount}"

FontWeight="SemiBold"/>

</StackPanel>

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

<TextBlock Text="Стоимость:"

Foreground="{StaticResource SystemBaseMediumColor}"/>

<TextBlock Text="{Binding TotalCostFormatted}"

FontWeight="SemiBold"

Foreground="Green"/>

</StackPanel>

</Grid>

</Border>

</DataTemplate>

</ItemsControl.ItemTemplate>

</ItemsControl>

</ScrollViewer>

<TextBlock Text="{Binding NoSalesMessage}"

IsVisible="{Binding !HasSales}"

HorizontalAlignment="Center"

VerticalAlignment="Center"

FontSize="16"

Foreground="{StaticResource SystemBaseMediumColor}"/>

</Grid>

</Border>

<TextBlock Grid.Row="3"

Text="{Binding TotalInfo}"

FontSize="18"

FontWeight="Bold"

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

</Grid>

</UserControl>

**PartnerSalesView.axaml.cs:**

using Avalonia.Controls;

namespace DemoApp.Views

{

public partial class PartnerSalesView : UserControl

{

public PartnerSalesView()

{

InitializeComponent();

}

}

}

**PartnerSalesViewModel.cs:**

using System;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using System.Linq;

using CommunityToolkit.Mvvm.ComponentModel;

using CommunityToolkit.Mvvm.Input;

using DemoApp.Entities;

using DemoApp.Models;

using DemoApp.Views;

using Microsoft.EntityFrameworkCore;

namespace DemoApp.ViewModels

{

public partial class PartnerSalesViewModel : ViewModelBase

{

private readonly DbDemoFdbContext \_context;

private readonly MainWindowViewModel \_mainViewModel;

private readonly int \_partnerId;

[ObservableProperty]

private ObservableCollection<Models.Sale> \_sales = new();

[ObservableProperty]

private string \_title;

[ObservableProperty]

private string \_summaryInfo;

[ObservableProperty]

private string \_discountInfo;

[ObservableProperty]

private string \_totalInfo;

[ObservableProperty]

private bool \_hasSales;

[ObservableProperty]

private string \_noSalesMessage = "У данного партнера пока нет истории продаж";

public PartnerSalesViewModel(DbDemoFdbContext context, MainWindowViewModel mainViewModel, int partnerId, string partnerName)

{

\_context = context;

\_mainViewModel = mainViewModel;

\_partnerId = partnerId;

Title = $"История продаж: {partnerName}";

LoadSales();

}

private void LoadSales()

{

var partner = \_context.Partners

.Include(p => p.Sales)

.ThenInclude(s => s.ProductArticleNavigation)

.FirstOrDefault(p => p.PartnerId == \_partnerId);

if (partner == null) return;

var sales = partner.Sales?

.Where(s => s.ProductCount.HasValue && s.ProductArticleNavigation != null)

.OrderByDescending(s => s.SaleDate)

.ToList() ?? new List<Entities.Sale>();

Sales = new ObservableCollection<Models.Sale>(

sales.Select(s => new Models.Sale(s))

);

HasSales = Sales.Any();

if (HasSales)

{

var totalProducts = sales.Sum(s => s.ProductCount ?? 0);

var totalCost = sales.Sum(s => (s.ProductCount ?? 0) \* (s.ProductArticleNavigation?.MinimumCostForPartner ?? 0));

var discount = Models.Partner.CalculatePartnerDiscount(totalProducts);

SummaryInfo = $"Всего продаж: {sales.Count}";

DiscountInfo = $"Текущая скидка партнера: {discount:P0}";

TotalInfo = $"Общая стоимость продаж: {totalCost:C2}";

}

else

{

SummaryInfo = "Нет данных о продажах";

DiscountInfo = "Текущая скидка партнера: 0%";

TotalInfo = "Общая стоимость продаж: 0.00 ₽";

}

}

[RelayCommand]

private void Back()

{

\_mainViewModel.ShowPartnersList();

}

}

}

**PartnersView.axaml:**

<UserControl xmlns="https://github.com/avaloniaui"

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"

mc:Ignorable="d"

d:DesignWidth="800"

d:DesignHeight="450"

x:Class="DemoApp.Views.PartnersView"

xmlns:vm="using:DemoApp.ViewModels"

x:DataType="vm:PartnersViewModel">

<Grid Margin="20">

<StackPanel Spacing="10" Classes="main">

<TextBlock Text="Список партнеров" FontSize="24" FontWeight="Bold" Classes="h1" />

<StackPanel Orientation="Horizontal" Spacing="10">

<Button Content="Добавить партнера" Command="{Binding AddPartnerCommand}" />

<Button Content="Редактировать" Command="{Binding EditPartnerCommand}" />

</StackPanel>

<ListBox ItemsSource="{Binding Partners}"

SelectedItem="{Binding SelectedPartner}"

MaxHeight="800">

<ListBox.ItemTemplate>

<DataTemplate>

<StackPanel Orientation="Horizontal" Margin="5" HorizontalAlignment="Left">

<StackPanel Orientation="Vertical" Margin="5" HorizontalAlignment="Left">

<StackPanel Orientation="Horizontal" HorizontalAlignment="Left">

<TextBlock Text="{Binding PartnerTypeDescription}" TextWrapping="Wrap" />

<TextBlock Text=" | " TextWrapping="Wrap" />

<TextBlock Text="{Binding \_partner.PartnerName}" TextWrapping="Wrap" Width="250" />

</StackPanel>

<TextBlock Text="{Binding \_partner.Director}" TextWrapping="Wrap" Width="250" HorizontalAlignment="Left" />

<TextBlock Text="{Binding \_partner.DirectorPhone}" TextWrapping="Wrap" Width="250" HorizontalAlignment="Left" />

<TextBlock Text="{Binding RatingDisplay}" TextWrapping="Wrap" Width="250" HorizontalAlignment="Left" />

</StackPanel>

<StackPanel Orientation="Vertical" Spacing="10">

<TextBlock Text="{Binding DiscountDisplay}" TextWrapping="Wrap" Width="150" />

<Button Content="История продаж"

Command="{Binding $parent[ListBox].((vm:PartnersViewModel)DataContext).ViewSalesHistoryCommand}"

CommandParameter="{Binding}" />

</StackPanel>

</StackPanel>

</DataTemplate>

</ListBox.ItemTemplate>

</ListBox>

</StackPanel>

</Grid>

</UserControl>

**PartnersView.axaml.cs:**

using Avalonia;

using Avalonia.Controls;

using Avalonia.Markup.Xaml;

namespace DemoApp.Views;

public partial class PartnersView : UserControl

{

public PartnersView()

{

InitializeComponent();

}

}

**PartnersViewModel.cs:**

using System.Collections.ObjectModel;

using System.Linq;

using CommunityToolkit.Mvvm.ComponentModel;

using CommunityToolkit.Mvvm.Input;

using DemoApp.Entities;

using DemoApp.Views;

using Microsoft.EntityFrameworkCore;

namespace DemoApp.ViewModels

{

public partial class PartnersViewModel : ObservableObject

{

private readonly DbDemoFdbContext \_context;

internal readonly MainWindowViewModel \_mainViewModel;

[ObservableProperty]

private ObservableCollection<Models.Partner> \_partners = new();

[ObservableProperty]

private Models.Partner? \_selectedPartner;

public PartnersViewModel(DbDemoFdbContext context, MainWindowViewModel mainViewModel)

{

\_context = context;

\_mainViewModel = mainViewModel;

LoadPartners();

}

public void LoadPartners()

{

var entities = \_context.Partners

.Include(p => p.Sales)

.Include(p => p.PartnerTypeNavigation)

.ToList();

Partners = new ObservableCollection<Models.Partner>(

entities.Select(e => new Models.Partner(e))

);

}

[RelayCommand]

private void EditPartner()

{

if (SelectedPartner == null) return;

var partnerEntity = \_context.Partners.Find(SelectedPartner.\_partner.PartnerId);

if (partnerEntity == null) return;

var viewModel = new EditPartnerViewModel(\_context, partnerEntity, this);

var editControl = new EditPartnerView { DataContext = viewModel };

\_mainViewModel.ShowEditPartner(editControl);

}

[RelayCommand]

private void AddPartner()

{

var viewModel = new EditPartnerViewModel(\_context, parentViewModel: this);

var editControl = new EditPartnerView { DataContext = viewModel };

\_mainViewModel.ShowEditPartner(editControl);

}

[RelayCommand]

private void ViewSalesHistory(Models.Partner? partner)

{

if (partner is null) return;

\_mainViewModel.ShowContent(new PartnerSalesView

{

DataContext = new PartnerSalesViewModel(

\_context,

\_mainViewModel,

partner.\_partner.PartnerId ?? 0,

partner.\_partner.PartnerName

)

});

}

}

}

**Sale.cs:**

using System;

using DemoApp.Entities;

namespace DemoApp.Models

{

public class Sale

{

public Sale(Entities.Sale sale)

{

\_sale = sale ?? throw new ArgumentNullException(nameof(sale));

}

public Entities.Sale \_sale { get; set; }

public string ProductName => \_sale.ProductArticleNavigation?.ProductName ?? "Нет данных";

public decimal? TotalCost => \_sale.ProductCount \* (\_sale.ProductArticleNavigation?.MinimumCostForPartner ?? 0);

public string SaleDateFormatted => \_sale.SaleDate?.ToString("dd.MM.yyyy") ?? "Дата не указана";

public string TotalCostFormatted => TotalCost.HasValue ? TotalCost.Value.ToString("C2") : "0.00 ₽";

}

}