|  |  |  |  |
| --- | --- | --- | --- |
| Модули | Описание | Количество строк кода | Размер (в Кбайтах) |
| 1 | 2 | 3 | 4 |
| App.xaml |  | 15 | 1 |
| App.xaml.cs |  | 11 | 1 |
| AssemblyInfo.cs |  | 9 | 1 |
| Models\Client.cs |  | 27 | 1 |
| Models\Employee.cs |  | 33 | 1 |
| Models\EmployeeAccount.cs |  | 23 | 1 |
| Models\Operation.cs |  | 15 | 1 |
| Models\Order.cs |  | 39 | 1 |
| Models\OrderSparePart.cs |  | 17 | 1 |
| Models\Position.cs |  | 13 | 1 |
| Models\Qualification.cs |  | 13 | 1 |
| Models\Role.cs |  | 13 | 1 |
| Models\ServiceStation.cs |  | 21 | 1 |
| Models\SparePart.cs |  | 19 | 1 |
| Models\Status.cs |  | 13 | 1 |
| Pages\ClientsPage.xaml |  | 66 | 6 |
| Pages\ClientsPage.xaml.cs |  | 104 | 4 |
| Pages\OrdersMasterPage.xaml |  | 28 | 2 |
| Pages\OrdersMasterPage.xaml.cs |  | 77 | 3 |
| Pages\OrdersPage.xaml |  | 112 | 9 |
| Pages\OrdersPage.xaml.cs |  | 127 | 5 |
| Pages\ReportPage.xaml |  | 47 | 3 |
| Pages\ReportPage.xaml.cs |  | 153 | 6 |
| Pages\ServiceStationPage.xaml |  | 61 | 5 |
| Pages\ServiceStationPage.xaml.cs |  | 86 | 3 |
| Pages\StorePage.xaml |  | 61 | 5 |
| Pages\StorePage.xaml.cs |  | 104 | 4 |
| Support\ApiHelper.cs |  | 68 | 3 |
| Support\Serdeser.cs |  | 15 | 1 |
| View\AuthView.xaml |  | 40 | 3 |
| View\AuthView.xaml.cs |  | 51 | 2 |
| View\MainView.xaml |  | 46 | 4 |
| View\MainView.xaml.cs |  | 92 | 4 |
| WPF\_UP.csproj |  | 20 | 1 |

1. App.xaml

﻿<Application x:Class="WPF\_UP.App"

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

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

xmlns:local="clr-namespace:WPF\_UP"

xmlns:ui="http://schemas.lepo.co/wpfui/2022/xaml"

StartupUri="/View/AuthView.xaml">

<Application.Resources>

<ResourceDictionary>

<ResourceDictionary.MergedDictionaries>

<ui:ThemesDictionary Theme="Light" />

<ui:ControlsDictionary />

</ResourceDictionary.MergedDictionaries>

</ResourceDictionary>

</Application.Resources>

</Application>

2. App.xaml.cs

﻿using System.Configuration;

using System.Data;

using System.Windows;

namespace WPF\_UP;

/// <summary>

/// Interaction logic for App.xaml

/// </summary>

public partial class App : Application

{

}

3. AssemblyInfo.cs

using System.Windows;

[assembly: ThemeInfo(

ResourceDictionaryLocation.None, //where theme specific resource dictionaries are located

//(used if a resource is not found in the page,

// or application resource dictionaries)

ResourceDictionaryLocation.SourceAssembly //where the generic resource dictionary is located

//(used if a resource is not found in the page,

// app, or any theme specific resource dictionaries)

)]

4. Models\Client.cs

﻿using System;

using System.Collections.Generic;

namespace WPF\_UP.Models;

public partial class Client

{

public int Id { get; set; }

public string Surname { get; set; } = null!;

public string ClientName { get; set; } = null!;

public string Patronymic { get; set; } = null!;

public string TelephoneNumber { get; set; } = null!;

public string CarBrand { get; set; } = null!;

public string CarModel { get; set; } = null!;

public string GovNumber { get; set; } = null!;

public DateTime? LastVisitDate { get; set; }

public virtual ICollection<Order> Orders { get; set; } = new List<Order>();

}

5. Models\Employee.cs

﻿using System;

using System.Collections.Generic;

namespace WPF\_UP.Models;

public partial class Employee

{

public int Id { get; set; }

public string Surname { get; set; } = null!;

public string EmployeeName { get; set; } = null!;

public string Patronymic { get; set; } = null!;

public int PositionId { get; set; }

public int QualificationId { get; set; }

public int ServiceStationId { get; set; }

public int EmployeeAccountId { get; set; }

public virtual EmployeeAccount EmployeeAccount { get; set; } = null!;

public virtual ICollection<Order> Orders { get; set; } = new List<Order>();

public virtual Position Position { get; set; } = null!;

public virtual Qualification Qualification { get; set; } = null!;

public virtual ServiceStation ServiceStation { get; set; } = null!;

}

6. Models\EmployeeAccount.cs

﻿using System;

using System.Collections.Generic;

namespace WPF\_UP.Models;

public partial class EmployeeAccount

{

public int Id { get; set; }

public string Login { get; set; } = null!;

public string Password { get; set; } = null!;

public string Telephone { get; set; } = null!;

public string Email { get; set; } = null!;

public int RoleId { get; set; }

public virtual ICollection<Employee> Employees { get; set; } = new List<Employee>();

public virtual Role Role { get; set; } = null!;

}

7. Models\Operation.cs

﻿using System;

using System.Collections.Generic;

namespace WPF\_UP.Models;

public partial class Operation

{

public int Id { get; set; }

public string OperationName { get; set; } = null!;

public decimal Price { get; set; }

public virtual ICollection<Order> Orders { get; set; } = new List<Order>();

}

8. Models\Order.cs

﻿using System;

using System.Collections.Generic;

namespace WPF\_UP.Models;

public partial class Order

{

public int Id { get; set; }

public int ClientId { get; set; }

public DateTime DateReference { get; set; }

public string Description { get; set; } = null!;

public DateTime? RepairDate { get; set; }

public int StatusId { get; set; }

public int ServiceStationId { get; set; }

public decimal Price { get; set; }

public int EmployeeId { get; set; }

public int OperationId { get; set; }

public virtual Client Client { get; set; } = null!;

public virtual Employee Employee { get; set; } = null!;

public virtual Operation Operation { get; set; } = null!;

public virtual ICollection<OrderSparePart> OrderSpareParts { get; set; } = new List<OrderSparePart>();

public virtual ServiceStation ServiceStation { get; set; } = null!;

public virtual Status Status { get; set; } = null!;

}

9. Models\OrderSparePart.cs

﻿using System;

using System.Collections.Generic;

namespace WPF\_UP.Models;

public partial class OrderSparePart

{

public int Id { get; set; }

public int SparePartId { get; set; }

public int OrderId { get; set; }

public virtual Order Order { get; set; } = null!;

public virtual SparePart SparePart { get; set; } = null!;

}

10. Models\Position.cs

﻿using System;

using System.Collections.Generic;

namespace WPF\_UP.Models;

public partial class Position

{

public int Id { get; set; }

public string PositionName { get; set; } = null!;

public virtual ICollection<Employee> Employees { get; set; } = new List<Employee>();

}

11. Models\Qualification.cs

﻿using System;

using System.Collections.Generic;

namespace WPF\_UP.Models;

public partial class Qualification

{

public int Id { get; set; }

public string QualificationName { get; set; } = null!;

public virtual ICollection<Employee> Employees { get; set; } = new List<Employee>();

}

12. Models\Role.cs

﻿using System;

using System.Collections.Generic;

namespace WPF\_UP.Models;

public partial class Role

{

public int Id { get; set; }

public string RoleName { get; set; } = null!;

public virtual ICollection<EmployeeAccount> EmployeeAccounts { get; set; } = new List<EmployeeAccount>();

}

13. Models\ServiceStation.cs

﻿using System;

using System.Collections.Generic;

namespace WPF\_UP.Models;

public partial class ServiceStation

{

public int Id { get; set; }

public string Address { get; set; } = null!;

public string TelephoneNumber { get; set; } = null!;

public string Email { get; set; } = null!;

public int QuantityWorkPlaces { get; set; }

public virtual ICollection<Employee> Employees { get; set; } = new List<Employee>();

public virtual ICollection<Order> Orders { get; set; } = new List<Order>();

}

14. Models\SparePart.cs

﻿using System;

using System.Collections.Generic;

namespace WPF\_UP.Models;

public partial class SparePart

{

public int Id { get; set; }

public string PartName { get; set; } = null!;

public string Articul { get; set; } = null!;

public int Quantity { get; set; }

public decimal Price { get; set; }

public virtual ICollection<OrderSparePart> OrderSpareParts { get; set; } = new List<OrderSparePart>();

}

15. Models\Status.cs

﻿using System;

using System.Collections.Generic;

namespace WPF\_UP.Models;

public partial class Status

{

public int Id { get; set; }

public string StatusName { get; set; } = null!;

public virtual ICollection<Order> Orders { get; set; } = new List<Order>();

}

16. Pages\ClientsPage.xaml

﻿<Page x:Class="WPF\_UP.Pages.ClientsPage"

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:WPF\_UP.Pages"

xmlns:ui="http://schemas.lepo.co/wpfui/2022/xaml"

mc:Ignorable="d"

Title="" MinHeight="720" MinWidth="1280" Height="1080" Width="1920">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="3\*"/>

<RowDefinition Height="13\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="3\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<DockPanel Grid.Row="0" Grid.RowSpan="2" Background="#D9D9D9" Grid.Column="0" Margin="25 100 25 25" HorizontalAlignment="Stretch" VerticalAlignment="Stretch">

<DataGrid DockPanel.Dock="Top" Name="ClientsGrid" ItemsSource="{Binding Clients}" AutoGenerateColumns="False" Foreground="White" VerticalAlignment="Top" HorizontalAlignment="Stretch" Background="#363636" SelectionMode="Single" Margin="20" SelectionChanged="ClientsGrid\_OnSelectionChanged">

<DataGrid.Columns>

<DataGridTextColumn Header="ID" Binding="{Binding Id}"></DataGridTextColumn>

<DataGridTextColumn Header="Фамилия" Binding="{Binding Surname}"></DataGridTextColumn>

<DataGridTextColumn Header="Имя" Binding="{Binding ClientName}"></DataGridTextColumn>

<DataGridTextColumn Header="Отчество" Binding="{Binding Patronymic}"></DataGridTextColumn>

<DataGridTextColumn Header="Телефон" Binding="{Binding TelephoneNumber}"></DataGridTextColumn>

<DataGridTextColumn Header="Бренд машины" Binding="{Binding CarBrand}"></DataGridTextColumn>

<DataGridTextColumn Header="Модель машины" Binding="{Binding CarModel}"></DataGridTextColumn>

<DataGridTextColumn Header="Гос. Номер" Binding="{Binding GovNumber}"></DataGridTextColumn>

<DataGridTextColumn Header="Дата последнего посещения" Binding="{Binding LastVisitDate}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

</DockPanel>

<WrapPanel Grid.Row="0" Grid.Column="2" Margin="0 15 0 0" HorizontalAlignment="Center" Background="#D9D9D9">

<ui:Button Name="AddClientBtn" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="10" Background="White" Click="AddClientBtn\_OnClick">

<TextBlock FontSize="20">Добавить клиента</TextBlock>

</ui:Button>

<ui:Button Name="EditClientBtn" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="10" Background="White" Click="EditClientBtn\_OnClick">

<TextBlock FontSize="20">Изменить клиента</TextBlock>

</ui:Button>

<ui:Button Name="DelClientBtn" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="95 10 0 0" Background="White" Click="DelClientBtn\_OnClick">

<TextBlock FontSize="20">Удалить клиента</TextBlock>

</ui:Button>

</WrapPanel>

<Grid Name="CreateClientPanel" Grid.Row="1" Grid.Column="1" HorizontalAlignment="Stretch" Margin="10 20" Background="#D9D9D9">

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<TextBlock Margin="0 20" VerticalAlignment="Stretch" Grid.ColumnSpan="2" Grid.Row="0" Text="Создание" FontSize="40" HorizontalAlignment="Center"></TextBlock>

<ui:TextBox Name="ClientNameBox" Grid.Row="1" Margin="15 15" PlaceholderText="ФИО" FontSize="25"></ui:TextBox>

<ui:TextBox Name="PhoneBox" Grid.Row="2" Margin="15 15" PlaceholderText="Номер Телефона" FontSize="25"></ui:TextBox>

<ui:TextBox Name="CarBrandBox" Grid.Row="3" Margin="15 15" PlaceholderText="Марка авто" FontSize="25"></ui:TextBox>

<ui:TextBox Name="CarModelBox" Grid.Row="4" Margin="15 15" PlaceholderText="Модель авто" FontSize="25"></ui:TextBox>

<ui:TextBox Name="GovNumBox" Grid.Row="5" Margin="15 15" PlaceholderText="Гос. Номер" FontSize="25"></ui:TextBox>

</Grid>

</Grid>

</Page>

17. Pages\ClientsPage.xaml.cs

﻿using System.Windows;

using System.Windows.Controls;

using WPF\_UP.Models;

using WPF\_UP.Support;

namespace WPF\_UP.Pages;

public partial class ClientsPage : Page

{

public ClientsPage()

{

InitializeComponent();

DataContext = this;

updateData();

}

public List<Client> Clients { get; set; } = new List<Client>();

private void AddClientBtn\_OnClick(object sender, RoutedEventArgs e)

{

Client client = new Client();

string[] name = new string[3];

name = ClientNameBox.Text.Split(" " , StringSplitOptions.RemoveEmptyEntries);

client.Surname = name[0];

client.ClientName = name[1];

client.Patronymic = name[2];

client.TelephoneNumber = PhoneBox.Text;

client.CarBrand = CarBrandBox.Text;

client.CarModel = CarModelBox.Text;

client.GovNumber = GovNumBox.Text;

client.LastVisitDate = null;

bool check = ApiHelper.Post<Client>(Serdeser.Serialize(client), "client");

if (check)

{

MessageBox.Show(check.ToString());

}

else

{

MessageBox.Show(check.ToString());

}

updateData();

}

private void EditClientBtn\_OnClick(object sender, RoutedEventArgs e)

{

if (ClientsGrid.SelectedItem != null)

{

Client client = ClientsGrid.SelectedItem as Client;

string[] name = new string[3];

name = ClientNameBox.Text.Split(" " , StringSplitOptions.RemoveEmptyEntries);

client.Surname = name[0];

client.ClientName = name[1];

client.Patronymic = name[2];

client.TelephoneNumber = PhoneBox.Text;

client.CarBrand = CarBrandBox.Text;

client.CarModel = CarModelBox.Text;

client.GovNumber = GovNumBox.Text;

bool check = ApiHelper.Put<Client>(Serdeser.Serialize(client), "client", client.Id);

updateData();

}

else

{

MessageBox.Show("Элемент не выбран");

}

}

private void DelClientBtn\_OnClick(object sender, RoutedEventArgs e)

{

if (ClientsGrid.SelectedItem!=null)

{

Client client = ClientsGrid.SelectedItem as Client;

bool check = ApiHelper.Delete<Client>("client", client.Id);

updateData();

}

else

{

MessageBox.Show("Элемент не выбран");

}

}

private void ClientsGrid\_OnSelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (ClientsGrid.SelectedItem is Client selectedClient)

{

ClientNameBox.Text = selectedClient.Surname + " " + selectedClient.ClientName + " " + selectedClient.Patronymic;

PhoneBox.Text = selectedClient.TelephoneNumber;

CarBrandBox.Text = selectedClient.CarBrand;

CarModelBox.Text = selectedClient.CarModel;

GovNumBox.Text = selectedClient.GovNumber;

}

else

{

ClientNameBox.Text = string.Empty;

PhoneBox.Text = string.Empty;

CarBrandBox.Text = string.Empty;

CarModelBox.Text = string.Empty;

GovNumBox.Text = string.Empty;

}

}

private void updateData()

{

Clients = ApiHelper.Get<List<Client>>("client");

}

}

18. Pages\OrdersMasterPage.xaml

﻿<Page x:Class="WPF\_UP.Pages.OrdersMasterPage"

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:WPF\_UP.Pages"

xmlns:collections="clr-namespace:System.Collections;assembly=System.Runtime"

mc:Ignorable="d"

Title="OrdersMasterPage" Height="720" Width="1280">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="5\*"/>

<RowDefinition Height="5\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="3\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<Grid Grid.Row="1" Grid.RowSpan="2" Grid.Column="0">

<DataGrid Name="DataGrib" VerticalAlignment="Stretch" HorizontalAlignment="Stretch" AutoGenerateColumns="True" Background="#363636" SelectionMode="Single">

</DataGrid>

</Grid>

<Button Grid.Row="1" Grid.Column="1" VerticalAlignment="Stretch" HorizontalAlignment="Stretch" Click="ButtonBase\_OnClick">

<TextBlock Text="Выполнить"></TextBlock>

</Button>

</Grid>

</Page>

19. Pages\OrdersMasterPage.xaml.cs

﻿using System.Windows;

using System.Windows.Controls;

using WPF\_UP.Models;

using WPF\_UP.Support;

namespace WPF\_UP.Pages;

public partial class OrdersMasterPage : Page

{

public OrdersMasterPage()

{

InitializeComponent();

DataContext = this;

DataGrib.ItemsSource = GetOrders();

}

public List<OrderMasterDisplay> GetOrders()

{

var clients = ApiHelper.Get<List<Client>>("client");

var employees = ApiHelper.Get<List<Employee>>("employee");

var orders = ApiHelper.Get<List<Order>>("order");

var servicestations = ApiHelper.Get<List<ServiceStation>>("servicestation");

var statuses = ApiHelper.Get<List<Status>>("status");

var orderDisplays = orders.Select(order => new OrderMasterDisplay

{

OrderId = order.Id,

ClientInfo = $"{clients.First(c => c.Id == order.ClientId).Surname} {clients.First(c => c.Id == order.ClientId).ClientName}",

DateReference = order.DateReference,

RepairDate = order.RepairDate,

StatusName = $"{statuses.First(s => s.Id == order.StatusId).StatusName}",

Price = order.Price,

EmployeeInfo = $"{employees.First(e => e.Id == order.EmployeeId).Surname} {employees.First(e => e.Id == order.EmployeeId).EmployeeName}",

WorkName = order.Description,

ServiceStationAddress = servicestations.First(s => s.Id == employees.First(e => e.Id == order.EmployeeId).ServiceStationId).Address

}).ToList();

return orderDisplays;

}

private void ButtonBase\_OnClick(object sender, RoutedEventArgs e)

{

bool sel = DataGrib.SelectedItem != null;

if (sel)

{

Order order = ApiHelper.Get<Order>("order", (DataGrib.SelectedItem as OrderMasterDisplay).OrderId);

order.StatusId = 3;

if (ApiHelper.Put<Order>(Serdeser.Serialize(order), "order", order.Id))

{

MessageBox.Show("Данные обновлены");

}

else{MessageBox.Show("Ошибка");}

DataGrib.ItemsSource = GetOrders();

}

else

{

MessageBox.Show("Элемент не выбран");

}

}

}

public partial class OrderMasterDisplay

{

public long OrderId { get; set; }

public string ClientInfo { get; set; }

public DateTime DateReference { get; set; }

public DateTime? RepairDate { get; set; }

public decimal Price { get; set; }

public string StatusName { get; set; }

public string EmployeeInfo { get; set; }

public string WorkName { get; set; }

public string ServiceStationAddress { get; set; } // Новое поле для адреса станции

}

20. Pages\OrdersPage.xaml

﻿<Page x:Class="WPF\_UP.Pages.OrdersPage"

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:WPF\_UP.Pages"

xmlns:ui="http://schemas.lepo.co/wpfui/2022/xaml"

mc:Ignorable="d"

Title="OrdersPage" MinHeight="720" MinWidth="1280" Height="1080" Width="1920">

<Grid HorizontalAlignment="Stretch" VerticalAlignment="Stretch">

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="5\*"/>

<RowDefinition Height="5\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="3\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<DockPanel Grid.Row="1" Grid.RowSpan="2" Background="#D9D9D9" Grid.Column="0" Margin="25" HorizontalAlignment="Stretch" VerticalAlignment="Stretch">

<DataGrid DockPanel.Dock="Top" Name="DataGrib" AutoGenerateColumns="False" Foreground="White" VerticalAlignment="Top" HorizontalAlignment="Stretch" Background="#363636" SelectionMode="Single" Margin="20" SelectionChanged="DataGrib\_OnSelectionChanged">

<DataGrid.Columns>

<DataGridTextColumn Header="Заказ ID" Binding="{Binding OrderId}"></DataGridTextColumn>

<DataGridTextColumn Header="Клиент" Binding="{Binding ClientInfo}"></DataGridTextColumn>

<DataGridTextColumn Header="Дата обращения" Binding="{Binding DateReference}"></DataGridTextColumn>

<DataGridTextColumn Header="Дата ремонта" Binding="{Binding RepairDate}"></DataGridTextColumn>

<DataGridTextColumn Header="Статус" Binding="{Binding StatusName}"></DataGridTextColumn>

<DataGridTextColumn Header="Адрес" Binding="{Binding ServiceStationAddress}"></DataGridTextColumn>

<DataGridTextColumn Header="Стоимость" Binding="{Binding Price}"></DataGridTextColumn>

<DataGridTextColumn Header="Сотрудник" Binding="{Binding EmployeeInfo}"></DataGridTextColumn>

<DataGridTextColumn Header="Название работы" Binding="{Binding WorkName}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<TextBox DockPanel.Dock="Bottom" Height="100" Name="DescriptBox"></TextBox>

</DockPanel>

<Grid Margin="15,12,20,30" Grid.Row="0" Grid.RowSpan="3" Grid.Column="1">

<Grid.RowDefinitions>

<RowDefinition Height="2\*"/>

<RowDefinition Height="10\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<WrapPanel HorizontalAlignment="Center" Background="#D9D9D9">

<ui:Button Name="AddOrderBtn" HorizontalAlignment="Center" Margin="10" Background="White">

<TextBlock FontSize="20">Создать заказ</TextBlock>

</ui:Button>

<ui:Button Name="EditOrderBtn" HorizontalAlignment="Center" Margin="10" Background="White">

<TextBlock FontSize="20">Изменить заказ</TextBlock>

</ui:Button>

<ui:Button Name="DelOrderButton" HorizontalAlignment="Center" Margin="95 10 0 0" Background="White">

<TextBlock FontSize="20">Удалить заказ</TextBlock>

</ui:Button>

</WrapPanel>

<Grid Name="CreatePanel" Grid.Row="1" Grid.ColumnSpan="2" HorizontalAlignment="Stretch" Margin="0 20" Background="#D9D9D9">

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<TextBlock Margin="0 20" VerticalAlignment="Stretch" Grid.ColumnSpan="2" Grid.Row="0" Text="Создание" FontSize="40" HorizontalAlignment="Center"></TextBlock>

<ComboBox Margin="0 20" VerticalAlignment="Stretch" Grid.ColumnSpan="2" Grid.Row="1" Name="ClientBox"></ComboBox>

<TextBlock Grid.Column="0" Margin="0 20" Grid.Row="2" VerticalAlignment="Stretch" HorizontalAlignment="Center" Text="Дата обращения"></TextBlock>

<TextBlock Grid.Column="1" Margin="0 20" VerticalAlignment="Stretch" Grid.Row="2" HorizontalAlignment="Center" Text="Дата ремонта"></TextBlock>

<ui:CalendarDatePicker Margin="0 20" VerticalAlignment="Stretch" Name="ReferncePick" HorizontalAlignment="Stretch" Grid.Row="3" Grid.Column="0"></ui:CalendarDatePicker>

<ui:CalendarDatePicker Margin="0 20" VerticalAlignment="Stretch" Name="RepairPick" HorizontalAlignment="Stretch" Grid.Row="3" Grid.Column="1"></ui:CalendarDatePicker>

<ComboBox Margin="0 20" VerticalAlignment="Stretch" Name="ServiceStationBox" Grid.Row="4" Grid.ColumnSpan="2"></ComboBox>

<ComboBox Margin="0 20" VerticalAlignment="Stretch" Name="EmployeeBox" Grid.Row="5" Grid.ColumnSpan="2"></ComboBox>

<ComboBox Margin="0 20" VerticalAlignment="Stretch" Name="OperationBox" Grid.Row="6" Grid.ColumnSpan="2"></ComboBox>

<Grid Background="White" Grid.Row="8" Grid.ColumnSpan="2" VerticalAlignment="Stretch" HorizontalAlignment="Stretch" Margin="10">

<ListView Name="PartsListView" HorizontalAlignment="Stretch" VerticalAlignment="Top" Margin="10,10,10,50">

<ListView.View>

<GridView>

<GridViewColumn Width="150">

<GridViewColumn.CellTemplate>

<DataTemplate>

<ComboBox ItemsSource="{Binding PartNames}" SelectedItem="{Binding SelectedPart}" />

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Width="50">

<GridViewColumn.CellTemplate>

<DataTemplate>

<Button Content="X" Command="{Binding DataContext.RemoveCommand, RelativeSource={RelativeSource AncestorType=ListView}}"

CommandParameter="{Binding}" />

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

</GridView>

</ListView.View>

</ListView>

<Button Content="Добавить" HorizontalAlignment="Center" VerticalAlignment="Bottom" Margin="10"

Width="100" Click="AddButton\_Click"/>

</Grid>

</Grid>

<Grid Name="UpdPanel" Grid.Row="1">

</Grid>

<ui:Button Background="#D9D9D9" HorizontalAlignment="Stretch" Grid.Row="2">

<TextBlock FontSize="45" Text="Сохранить"></TextBlock>

</ui:Button>

</Grid>

</Grid>

</Page>

21. Pages\OrdersPage.xaml.cs

﻿using System.Windows;

using System.Windows.Controls;

using System.Windows.Input;

using WPF\_UP.Models;

using WPF\_UP.Support;

namespace WPF\_UP.Pages;

public partial class OrdersPage : Page

{

public OrdersPage()

{

InitializeComponent();InitializeComponent();

DataContext = this;

DataGrib.ItemsSource = GetOrders();

Parts = ApiHelper.Get<List<SparePart>>("sparepart");

PartsListView.ItemsSource = PartItems;

}

public List<SparePart> Parts { get; set; } = new List<SparePart>();

public List<PartItem> PartItems { get; set; } = new List<PartItem>();

public List<OrderDisplay> GetOrders()

{

var clients = ApiHelper.Get<List<Client>>("client");

var employees = ApiHelper.Get<List<Employee>>("employee");

var orders = ApiHelper.Get<List<Order>>("order");

var servicestations = ApiHelper.Get<List<ServiceStation>>("servicestation");

var statuses = ApiHelper.Get<List<Status>>("status");

var works = ApiHelper.Get<List<Operation>>("operation");

var orderDisplays = orders.Select(order => new OrderDisplay

{

OrderId = order.Id,

ClientInfo = $"{clients.First(c => c.Id == order.ClientId).Surname} {clients.First(c => c.Id == order.ClientId).ClientName}",

DateReference = order.DateReference,

RepairDate = order.RepairDate,

StatusName = $"{statuses.First(s => s.Id == order.StatusId).StatusName}",

Price = order.Price,

WorkName = $"{works.First((w=>w.Id == order.OperationId)).OperationName}",

EmployeeInfo = $"{employees.First(e => e.Id == order.EmployeeId).Surname} {employees.First(e => e.Id == order.EmployeeId).EmployeeName}",

ServiceStationAddress = servicestations.First(s => s.Id == employees.First(e => e.Id == order.EmployeeId).ServiceStationId).Address,

Description = order.Description

}).ToList();

return orderDisplays;

}

private void ButtonBase\_OnClick(object sender, RoutedEventArgs e)

{

bool sel = DataGrib.SelectedItem != null;

if (sel)

{

Order order = ApiHelper.Get<Order>("order", (DataGrib.SelectedItem as OrderDisplay).OrderId);

order.StatusId = 3;

if (ApiHelper.Put<Order>(Serdeser.Serialize(order), "order", order.Id))

{

MessageBox.Show("Данные обновлены");

}

else{ MessageBox.Show("Ошибка"); }

DataGrib.ItemsSource = GetOrders();

}

else

{

MessageBox.Show("Элемент не выбран");

}

}

private void DataGrib\_OnSelectionChanged(object sender, SelectionChangedEventArgs e)

{

DescriptBox.Visibility = Visibility.Visible;

DescriptBox.Text = (DataGrib.SelectedItem as OrderDisplay).Description;

}

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

{

PartItems.Add(new PartItem { Parts = Parts, SelectedPart = 0 });

PartsListView.ItemsSource = PartItems;

}

}

public partial class OrderDisplay

{

public long OrderId { get; set; }

public string ClientInfo { get; set; }

public DateTime DateReference { get; set; }

public DateTime? RepairDate { get; set; }

public decimal Price { get; set; }

public string StatusName { get; set; }

public string EmployeeInfo { get; set; }

public string WorkName { get; set; }

public string ServiceStationAddress { get; set; }

public string Description { get; set; }

}

public class PartItem

{

public List<SparePart> Parts { get; set; }

public int SelectedPart { get; set; }

}

public class RelayCommand : ICommand

{

private readonly Action<object> \_execute;

private readonly Predicate<object> \_canExecute;

public RelayCommand(Action<object> execute, Predicate<object> canExecute = null)

{

\_execute = execute;

\_canExecute = canExecute;

}

public bool CanExecute(object parameter) => \_canExecute?.Invoke(parameter) ?? true;

public void Execute(object parameter) => \_execute(parameter);

public event EventHandler CanExecuteChanged

{

add { CommandManager.RequerySuggested += value; }

remove { CommandManager.RequerySuggested -= value; }

}

}

22. Pages\ReportPage.xaml

﻿<Page x:Class="WPF\_UP.Pages.ReportPage"

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:sys="clr-namespace:System;assembly=mscorlib"

xmlns:ui="http://schemas.lepo.co/wpfui/2022/xaml"

xmlns:local="clr-namespace:WPF\_UP.Pages"

mc:Ignorable="d"

Title="ReportPage" Height="450" Width="800">

<Grid>

<Grid Row="1">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<ComboBox Name="typeBox" Grid.Row="0" Grid.ColumnSpan="2" Margin="25 10" Background="#D9D9D9">

<sys:String>Отчет о доходах</sys:String>

<sys:String>Отчет о запчастях на складе</sys:String>

<sys:String>Отчет о популярности работ</sys:String>

<sys:String>Отчет о популярности запчастей</sys:String>

</ComboBox>

<ui:Button Name="exportBtn" Grid.Column="2" HorizontalAlignment="Center" Grid.Row="0" Click="ExportBtn\_OnClick" Padding="10" Background="#D9D9D9">

<TextBlock Text="Экспорт" FontSize="15"></TextBlock>

</ui:Button>

<ui:Button Name="printBtn" Grid.Column="3" Grid.Row="0" Padding="10" Click="PrintBtn\_OnClick" Background="#D9D9D9">

<TextBlock Text="Печать" FontSize="15"></TextBlock>

</ui:Button>

<TextBlock Grid.Row="1" HorizontalAlignment="Center" Text="Дата с"></TextBlock>

<TextBlock Grid.Row="1" Grid.Column="1" HorizontalAlignment="Center" Text="Дата по"></TextBlock>

<ui:CalendarDatePicker Name="fromDate" Grid.Row="3" Margin="10" HorizontalAlignment="Stretch" VerticalAlignment="Stretch" IsTodayHighlighted="True"></ui:CalendarDatePicker>

<ui:CalendarDatePicker Name="toDate" Grid.Row="3" Grid.Column="1" Margin="10" HorizontalAlignment="Stretch" VerticalAlignment="Stretch" IsTodayHighlighted="True"></ui:CalendarDatePicker>

</Grid>

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition Height="4\*"/>

<RowDefinition Height="4\*"/>

</Grid.RowDefinitions>

</Grid>

</Page>

23. Pages\ReportPage.xaml.cs

﻿using System.Diagnostics;

using System.IO;

using System.Windows;

using System.Windows.Controls;

using System.Drawing.Printing;

using Microsoft.WindowsAPICodePack.Dialogs;

using WPF\_UP.Support;

namespace WPF\_UP.Pages;

public partial class ReportPage : Page

{

public ReportPage()

{

InitializeComponent();

}

private void ExportBtn\_OnClick(object sender, RoutedEventArgs e)

{

string file;

string model;

CommonOpenFileDialog fileDialog = new CommonOpenFileDialog() {IsFolderPicker = true};

byte[] excelBytes;

switch (typeBox.SelectedIndex)

{

case 0:

model = $"reports/income/excel?startDate={fromDate.Date:yyyy-MM-dd}&endDate={toDate.Date:yyyy-MM-dd}";

file = "\\Income.xlsx";

excelBytes = ApiHelper.GetFile(model);

File.WriteAllBytes(file, excelBytes);

try

{

// Открываем файл в приложении по умолчанию

Process.Start(new ProcessStartInfo(file) { UseShellExecute = true });

Console.WriteLine("Файл успешно открыт.");

}

catch (Exception ex)

{

Console.WriteLine($"Ошибка при открытии файла: {ex.Message}");

}

break;

case 1:

model = "reports/spareparts/stock/excel";

file = "\\Sparepart.xlsx";

excelBytes = ApiHelper.GetFile(model);

File.WriteAllBytes(file, excelBytes);

try

{

// Открываем файл в приложении по умолчанию

Process.Start(new ProcessStartInfo(file) { UseShellExecute = true });

Console.WriteLine("Файл успешно открыт.");

}

catch (Exception ex)

{

Console.WriteLine($"Ошибка при открытии файла: {ex.Message}");

}

break;

case 2:

model = "reports/operations/popular/excel";

file = "\\popularOper.xlsx";

excelBytes = ApiHelper.GetFile(model);

File.WriteAllBytes(file, excelBytes);

try

{

// Открываем файл в приложении по умолчанию

Process.Start(new ProcessStartInfo(file) { UseShellExecute = true });

Console.WriteLine("Файл успешно открыт.");

}

catch (Exception ex)

{

Console.WriteLine($"Ошибка при открытии файла: {ex.Message}");

}

break;

case 3:

model = "reports/spareparts/popular/excel";

file = "\\popelarParts.xlsx";

excelBytes = ApiHelper.GetFile(model);

File.WriteAllBytes(file, excelBytes);

try

{

// Открываем файл в приложении по умолчанию

Process.Start(new ProcessStartInfo(file) { UseShellExecute = true });

Console.WriteLine("Файл успешно открыт.");

}

catch (Exception ex)

{

Console.WriteLine($"Ошибка при открытии файла: {ex.Message}");

}

break;

}

}

private void PrintBtn\_OnClick(object sender, RoutedEventArgs e)

{

string file;

string model;

byte[] excelBytes;

try

{

switch (typeBox.SelectedIndex)

{

case 0:

model = $"reports/income/excel?startDate={fromDate.Date:yyyy-MM-dd}&endDate={toDate.Date:yyyy-MM-dd}";

file = "Income.xlsx";

break;

case 1:

model = "reports/spareparts/stock/excel";

file = "Sparepart.xlsx";

break;

case 2:

model = "reports/operations/popular/excel";

file = "popularOper.xlsx";

break;

case 3:

model = "reports/spareparts/popular/excel";

file = "popelarParts.xlsx";

break;

default:

Console.WriteLine("Неверный выбор.");

return;

}

// Получение данных файла и запись на диск

excelBytes = ApiHelper.GetFile(model);

File.WriteAllBytes(file, excelBytes);

// Печать файла

Process printProcess = new Process

{

StartInfo = new ProcessStartInfo

{

FileName = file,

Verb = "print",

CreateNoWindow = true,

UseShellExecute = true

}

};

printProcess.Start();

Console.WriteLine("Файл успешно отправлен на печать.");

}

catch (Exception ex)

{

Console.WriteLine($"Ошибка: {ex.Message}");

}

}

}

24. Pages\ServiceStationPage.xaml

﻿<Page x:Class="WPF\_UP.Pages.ServiceStationPage"

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:WPF\_UP.Pages"

xmlns:ui="http://schemas.lepo.co/wpfui/2022/xaml"

mc:Ignorable="d"

Title="" MinHeight="720" MinWidth="1280" Height="1080" Width="1920">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="3\*"/>

<RowDefinition Height="13\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="3\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<DockPanel Grid.Row="0" Grid.RowSpan="2" Background="#D9D9D9" Grid.Column="0" Margin="25 100 25 25" HorizontalAlignment="Stretch" VerticalAlignment="Stretch">

<DataGrid DockPanel.Dock="Top" Name="StationGrid" ItemsSource="{Binding ServiceStations}" AutoGenerateColumns="False" Foreground="White" VerticalAlignment="Top" HorizontalAlignment="Stretch" Background="#363636" SelectionMode="Single" Margin="20" SelectionChanged="StationGrid\_OnSelectionChangedGrid\_OnSelectionChanged">

<DataGrid.Columns>

<DataGridTextColumn Header="ID" Binding="{Binding Id}"></DataGridTextColumn>

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

<DataGridTextColumn Header="Телефон" Binding="{Binding TelephoneNumber}"></DataGridTextColumn>

<DataGridTextColumn Header="Почта" Binding="{Binding Email}"></DataGridTextColumn>

<DataGridTextColumn Header="Кол-во рабочих мест" Binding="{Binding QuantityWorkPlaces}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

</DockPanel>

<WrapPanel Grid.Row="0" Grid.Column="2" Margin="0 15 0 0" HorizontalAlignment="Center" Background="#D9D9D9">

<ui:Button Name="AddClientBtn" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="10" Background="White" Click="AddSTBtn\_OnClick">

<TextBlock FontSize="20">Добавить СТО</TextBlock>

</ui:Button>

<ui:Button Name="EditClientBtn" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="10" Background="White" Click="EditSTBtn\_OnClick">

<TextBlock FontSize="20">Изменить СТО</TextBlock>

</ui:Button>

<ui:Button Name="DelClientBtn" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="95 10 0 0" Background="White" Click="DelSTBtn\_OnClick">

<TextBlock FontSize="20">Удалить СТО</TextBlock>

</ui:Button>

</WrapPanel>

<Grid Name="CreateClientPanel" Grid.Row="1" Grid.Column="1" HorizontalAlignment="Stretch" Margin="10 20" Background="#D9D9D9">

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<TextBlock Margin="0 20" VerticalAlignment="Stretch" Grid.ColumnSpan="2" Grid.Row="0" Text="Создание" FontSize="40" HorizontalAlignment="Center"></TextBlock>

<ui:TextBox Name="AddressBox" Grid.Row="1" Margin="15 15" PlaceholderText="Адрес" FontSize="25"></ui:TextBox>

<ui:TextBox Name="PhoneBox" Grid.Row="2" Margin="15 15" PlaceholderText="Телефон" FontSize="25"></ui:TextBox>

<ui:TextBox Name="EmailBox" Grid.Row="3" Margin="15 15" PlaceholderText="Почта" FontSize="25"></ui:TextBox>

<ui:TextBox Name="QuantityBox" Grid.Row="4" Margin="15 15" PlaceholderText="Кол-во рабочих мест" FontSize="25"></ui:TextBox>

</Grid>

</Grid>

</Page>

25. Pages\ServiceStationPage.xaml.cs

﻿using System.Windows;

using System.Windows.Controls;

using WPF\_UP.Models;

using WPF\_UP.Support;

namespace WPF\_UP.Pages;

public partial class ServiceStationPage : Page

{

public ServiceStationPage()

{

InitializeComponent();

DataContext = this;

updateData();

}

public List<ServiceStation> ServiceStations { get; set; } = new List<ServiceStation>();

private void updateData()

{

ServiceStations = ApiHelper.Get<List<ServiceStation>>("servicestation");

}

private void AddSTBtn\_OnClick(object sender, RoutedEventArgs e)

{

ServiceStation serviceStation = new ServiceStation();

serviceStation.Address = AddressBox.Text;

serviceStation.TelephoneNumber = PhoneBox.Text;

serviceStation.Email = EmailBox.Text;

serviceStation.QuantityWorkPlaces = int.Parse(QuantityBox.Text);

bool check = ApiHelper.Post<ServiceStation>(Serdeser.Serialize(serviceStation), "servicestation");

if (check)

{

MessageBox.Show(check.ToString());

}

else

{

MessageBox.Show(check.ToString());

}

updateData();

}

private void EditSTBtn\_OnClick(object sender, RoutedEventArgs e)

{

if (StationGrid.SelectedItem != null)

{

ServiceStation serviceStation = StationGrid.SelectedItem as ServiceStation;

serviceStation.Address = AddressBox.Text;

serviceStation.TelephoneNumber = PhoneBox.Text;

serviceStation.Email = EmailBox.Text;

serviceStation.QuantityWorkPlaces = int.Parse(QuantityBox.Text);

bool check = ApiHelper.Put<ServiceStation>(Serdeser.Serialize(serviceStation), "servicestation", serviceStation.Id);

if (check)

{

MessageBox.Show(check.ToString());

}

else

{

MessageBox.Show(check.ToString());

}

updateData();

}

else

{

MessageBox.Show("Элемент не выбран");

}

}

private void DelSTBtn\_OnClick(object sender, RoutedEventArgs e)

{

if (StationGrid.SelectedItem != null)

{

ServiceStation serviceStation = StationGrid.SelectedItem as ServiceStation;

bool check = ApiHelper.Delete<ServiceStation>("servicestation", serviceStation.Id);

updateData();

}

}

private void StationGrid\_OnSelectionChangedGrid\_OnSelectionChanged(object sender, SelectionChangedEventArgs e)

{

ServiceStation serviceStation = StationGrid.SelectedItem as ServiceStation;

AddressBox.Text = serviceStation.Address;

PhoneBox.Text = serviceStation.TelephoneNumber;

EmailBox.Text = serviceStation.Email;

QuantityBox.Text = serviceStation.QuantityWorkPlaces.ToString();

}

}

26. Pages\StorePage.xaml

﻿<Page x:Class="WPF\_UP.Pages.StorePage"

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:WPF\_UP.Pages"

xmlns:ui="http://schemas.lepo.co/wpfui/2022/xaml"

mc:Ignorable="d"

Title="" MinHeight="720" MinWidth="1280" Height="1080" Width="1920">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="3\*"/>

<RowDefinition Height="13\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="3\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<DockPanel Grid.Row="0" Grid.RowSpan="2" Background="#D9D9D9" Grid.Column="0" Margin="25 100 25 25" HorizontalAlignment="Stretch" VerticalAlignment="Stretch">

<DataGrid DockPanel.Dock="Top" Name="StoreGrid" ItemsSource="{Binding spareParts}" AutoGenerateColumns="False" Foreground="White" VerticalAlignment="Top" HorizontalAlignment="Stretch" Background="#363636" SelectionMode="Single" Margin="20" SelectionChanged="DataGrib\_OnSelectionChanged">

<DataGrid.Columns>

<DataGridTextColumn Header="ID Запчасти" Binding="{Binding Id}"></DataGridTextColumn>

<DataGridTextColumn Header="Название" Binding="{Binding PartName}"></DataGridTextColumn>

<DataGridTextColumn Header="Артикул" Binding="{Binding Articul}"></DataGridTextColumn>

<DataGridTextColumn Header="Кол-во на складе" Binding="{Binding Quantity}"></DataGridTextColumn>

<DataGridTextColumn Header="Цена" Binding="{Binding Price}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

</DockPanel>

<WrapPanel Grid.Row="0" Grid.Column="2" Margin="0 15 0 0" HorizontalAlignment="Center" Background="#D9D9D9">

<ui:Button Name="AddPartBtn" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="10" Background="White" Click="AddPartBtn\_OnClick">

<TextBlock FontSize="20">Добавить запчасть</TextBlock>

</ui:Button>

<ui:Button Name="EditPartBtn" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="10" Background="White" Click="EditPartBtn\_OnClick">

<TextBlock FontSize="20">Изменить запчасть</TextBlock>

</ui:Button>

<ui:Button Name="DelPartBtn" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="95 10 0 0" Background="White" Click="DelPartBtn\_OnClick">

<TextBlock FontSize="20">Удалить запчасть</TextBlock>

</ui:Button>

</WrapPanel>

<Grid Name="CreatePartPanel" Grid.Row="1" Grid.Column="1" HorizontalAlignment="Stretch" Margin="10 20" Background="#D9D9D9">

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<TextBlock Margin="0 20" VerticalAlignment="Stretch" Grid.ColumnSpan="2" Grid.Row="0" Text="Создание" FontSize="40" HorizontalAlignment="Center"></TextBlock>

<ui:TextBox Name="partNameBox" Grid.Row="1" Margin="15 15" PlaceholderText="Название" FontSize="25"></ui:TextBox>

<ui:TextBox Name="articulBox" Grid.Row="2" Margin="15 15" PlaceholderText="Артикул" FontSize="25"></ui:TextBox>

<ui:TextBox Name="quantityBox" Grid.Row="3" Margin="15 15" PlaceholderText="Кол-во на складе" FontSize="25"></ui:TextBox>

<ui:TextBox Name="priceBox" Grid.Row="4" Margin="15 15" PlaceholderText="Цена" FontSize="25"></ui:TextBox>

</Grid>

</Grid>

</Page>

27. Pages\StorePage.xaml.cs

﻿using System.Windows;

using System.Windows.Controls;

using WPF\_UP.Models;

using WPF\_UP.Support;

namespace WPF\_UP.Pages;

public partial class StorePage : Page

{

public StorePage()

{

InitializeComponent();

DataContext = this;

updateData();

}

public List<SparePart> spareParts { get; set; } = new List<SparePart>();

private void DataGrib\_OnSelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (StoreGrid.SelectedItem is SparePart selectedPart)

{

partNameBox.Text = selectedPart.PartName;

articulBox.Text = selectedPart.Articul;

quantityBox.Text = selectedPart.Quantity.ToString();

priceBox.Text = selectedPart.Price.ToString();

}

else

{

// Если выделения нет, очищаем текстовые поля (опционально)

partNameBox.Text = string.Empty;

articulBox.Text = string.Empty;

quantityBox.Text = string.Empty;

priceBox.Text = string.Empty;

}

}

private void AddPartBtn\_OnClick(object sender, RoutedEventArgs e)

{

SparePart sparePart = new SparePart();

sparePart.PartName = partNameBox.Text;

sparePart.Articul = articulBox.Text;

sparePart.Quantity = int.Parse(quantityBox.Text);

sparePart.Price = decimal.Parse(priceBox.Text);

if (spareParts.FirstOrDefault(sp => sp.Articul == articulBox.Text) != null)

{

MessageBox.Show("Данный артикул занят");

articulBox.Text = "";

}

else if(sparePart!=null)

{

ApiHelper.Post<SparePart>(Serdeser.Serialize(sparePart), "sparepart");

updateData();

}

}

private void DelPartBtn\_OnClick(object sender, RoutedEventArgs e)

{

if (StoreGrid.SelectedItem != null)

{

ApiHelper.Delete<SparePart>("sparepart", (StoreGrid.SelectedItem as SparePart).Id);

updateData();

}

else

{

MessageBox.Show("Не выбран элемент");

}

}

private void EditPartBtn\_OnClick(object sender, RoutedEventArgs e)

{

if (StoreGrid.SelectedItem != null)

{

var selected = StoreGrid.SelectedItem as SparePart;

SparePart sparePart = new SparePart();

sparePart.PartName = partNameBox.Text;

sparePart.Articul = articulBox.Text;

sparePart.Quantity = int.Parse(quantityBox.Text);

sparePart.Price = decimal.Parse(priceBox.Text);

if(sparePart!=null)

{

bool check = ApiHelper.Put<SparePart>(Serdeser.Serialize(sparePart), "sparepart", selected.Id);

if (!check)

{

MessageBox.Show("Ошибка добавления");

}

else

{

updateData();

}

}

}

else

{

MessageBox.Show("Не выбран элемент");

}

}

private void updateData()

{

spareParts = ApiHelper.Get<List<SparePart>>("sparepart");

}

}

28. Support\ApiHelper.cs

﻿using System.Net;

using System.Net.Http;

using System.Text;

using Newtonsoft.Json;

namespace WPF\_UP.Support;

public static class ApiHelper

{

private static readonly string \_url = "http://localhost:5181/api";

public static T? Get<T>(string model, long id = 0)

{

var client = new HttpClient();

var request = id == 0 ? $"/{model}" : $"/{model}/{id}";

var response = client.GetAsync($"{\_url}/{model}/{(id != 0 ? id : string.Empty)}").Result;

if (response.StatusCode != HttpStatusCode.OK) return default;

return JsonConvert.DeserializeObject<T>(response.Content.ReadAsStringAsync().Result);

}

public static bool Put<T>(string json, string model, long id)

{

var client = new HttpClient();

HttpContent body = new StringContent(json, Encoding.UTF8, "application/json");

var response = client.PutAsync($"{\_url}/{model}/{id}", body).Result;

if (response.StatusCode != HttpStatusCode.NoContent) return false;

return true;

}

public static bool Post<T>(string json, string model)

{

var client = new HttpClient();

HttpContent body = new StringContent(json, Encoding.UTF8, "application/json");

// Логирование запроса

Console.WriteLine("Sending POST request to: " + $"{\_url}/{model}");

Console.WriteLine("Request body: " + json);

var response = client.PostAsync($"{\_url}/{model}", body).Result;

// Логирование ответа

Console.WriteLine("Response status code: " + response.StatusCode);

Console.WriteLine("Response content: " + response.Content.ReadAsStringAsync().Result);

if (response.StatusCode != HttpStatusCode.Created) return false;

return true;

}

public static bool Delete<T>(string model, long id)

{

var client = new HttpClient();

var response = client.DeleteAsync($"{\_url}/{model}/{id}").Result;

if (response.StatusCode == HttpStatusCode.NoContent) return true;

return false;

}

public static byte[] GetFile(string model)

{

var client = new HttpClient();

var requestUrl = $"{\_url}/{model}";

var response = client.GetAsync(requestUrl).Result;

if (response.StatusCode != HttpStatusCode.OK)

{

return null;

}

return response.Content.ReadAsByteArrayAsync().Result;

}

}

29. Support\Serdeser.cs

﻿using Newtonsoft.Json;

namespace WPF\_UP.Support;

public static class Serdeser

{

public static string Serialize<T>(T obj)

{

return JsonConvert.SerializeObject(obj);

}

public static T Deserialize<T>(string json)

{

return JsonConvert.DeserializeObject<T>(json);

}

}

30. View\AuthView.xaml

﻿<Window x:Class="WPF\_UP.View.AuthView"

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:WPF\_UP.View"

xmlns:ui="http://schemas.lepo.co/wpfui/2022/xaml"

mc:Ignorable="d"

Title="АИ «АвтоРемСервис»" MaxHeight="1080" MaxWidth="1920" MinHeight="720" MinWidth="1280">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="2\*"/>

<ColumnDefinition Width="6\*"/>

<ColumnDefinition Width="2\*"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition Height="4\*"/>

<RowDefinition/>

</Grid.RowDefinitions>

<Grid Grid.Row="1" Grid.Column="1" Background="#D9D9D9">

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="2\*"/>

<RowDefinition Height="2\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="2\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="2\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<TextBlock Grid.Row="1" FontSize="45" HorizontalAlignment="Center" Text="Добро пожаловать"/>

<ui:TextBox Grid.Row="2" Name="login" Margin="100,30" FontSize="20" PlaceholderText="Логин" Background="White"/>

<ui:PasswordBox Grid.Row="4" Name="password" Margin="100,30" FontSize="20" PlaceholderText="Пароль" Background="White"/>

<ui:Button Name="authButton" CornerRadius="10" Margin="5" HorizontalAlignment="Center" Grid.Row="6" Click="Authentificate" Background="White">

<TextBlock FontSize="35" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="50,0">Войти</TextBlock>

</ui:Button>

</Grid>

</Grid>

</Window>

31. View\AuthView.xaml.cs

﻿using System.ComponentModel;

using System.Runtime.CompilerServices;

using System.Windows;

using System.Security.Cryptography;

using System.Text;

using WPF\_UP.Models;

using WPF\_UP.Support;

namespace WPF\_UP.View;

public partial class AuthView : Window

{

public AuthView()

{

InitializeComponent();

}

private void Authentificate(object sender, RoutedEventArgs e)

{

List<EmployeeAccount> accounts = ApiHelper.Get<List<EmployeeAccount>>("employeeaccount");

string hashedPassword = ComputeSha256Hash(password.Password);

EmployeeAccount realaccount = (from acc in accounts

where acc.Login == login.Text && acc.Password == hashedPassword

select acc).FirstOrDefault();

if (realaccount!=null)

{

MainView mainView = new MainView(realaccount);

mainView.Show();

this.Close();

}

else

{

MessageBox.Show("Ошибка входа " + password.Password);

}

}

static string ComputeSha256Hash(string rawData)

{

using (SHA256 sha256Hash = SHA256.Create())

{

byte[] bytes = sha256Hash.ComputeHash(Encoding.UTF8.GetBytes(rawData));

StringBuilder builder = new StringBuilder();

for (int i = 0; i < bytes.Length; i++)

{

builder.Append(bytes[i].ToString("x2"));

}

return builder.ToString();

}

}

}

32. View\MainView.xaml

﻿<Window x:Class="WPF\_UP.View.MainView"

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:WPF\_UP.View"

xmlns:ui="http://schemas.lepo.co/wpfui/2022/xaml"

mc:Ignorable="d"

Title="АИ «АвтоРемСервис»" MaxHeight="1080" MaxWidth="1920" MinHeight="720" MinWidth="1280">

<Grid>

<ui:Button Name="MainButton" Margin="15" VerticalAlignment="Top" Visibility="Collapsed" Panel.ZIndex="10" Background="#D9D9D9" Click="MainButton\_OnClick">

<TextBlock Text="На главную"/>

</ui:Button>

<Frame Name="MainFrame" ></Frame>

<Grid>

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<WrapPanel Grid.Row="1" Orientation="Horizontal" VerticalAlignment="Center" HorizontalAlignment="Center">

<Button Name="ordersButton" Visibility="Collapsed" Background="#D9D9D9" Margin="100" Height="100" Width="250" Click="OrdersButton\_OnClick">

<TextBlock Text="Заказы" FontSize="35"></TextBlock>

</Button>

<Button Name="clientsButton" Visibility="Collapsed" Background="#D9D9D9" Margin="100" Height="100" Width="250" Click="ClientsButton\_OnClick">

<TextBlock Text="Клиенты" FontSize="35"></TextBlock>

</Button>

<Button Name="storeButton" Visibility="Collapsed" Background="#D9D9D9" Margin="100" Height="100" Width="250" Click="StoreButton\_OnClick">

<TextBlock Text="Склад" FontSize="35"></TextBlock>

</Button>

<Button Name="employeeButton" Visibility="Collapsed" Background="#D9D9D9" Margin="100" Height="100" Width="250">

<TextBlock Text="Сотрудники" FontSize="35"></TextBlock>

</Button>

<Button Name="serviceButton" Visibility="Collapsed" Background="#D9D9D9" Margin="100" Height="100" Width="250">

<TextBlock Text="СТО" FontSize="35"></TextBlock>

</Button>

<Button Name="reportsButton" Visibility="Collapsed" Background="#D9D9D9" Margin="100" Height="100" Width="250" Click="ReportsButton\_OnClick">

<TextBlock Text="Отчеты" FontSize="35"></TextBlock>

</Button>

<Button Name="scheduleButton" Visibility="Collapsed" Background="#D9D9D9" Margin="100" Height="100" Width="250">

<TextBlock Text="Расписание" FontSize="35"></TextBlock>

</Button>

</WrapPanel>

</Grid>

</Grid>

</Window>

33. View\MainView.xaml.cs

﻿using System.Windows;

using WPF\_UP.Models;

using WPF\_UP.Pages;

using WPF\_UP.Support;

namespace WPF\_UP.View;

public partial class MainView : Window

{

public MainView(EmployeeAccount employeeAccount)

{

InitializeComponent();

\_employeeAccount = employeeAccount;

ShowNeededButtons(employeeAccount.RoleId);

}

EmployeeAccount \_employeeAccount;

private void OrdersButton\_OnClick(object sender, RoutedEventArgs e)

{

MainFrame.Content = new OrdersPage();

MainButton.Visibility = Visibility.Visible;

HideAllButtons();

}

private void MainButton\_OnClick(object sender, RoutedEventArgs e)

{

MainFrame.Content = null;

MainButton.Visibility = Visibility.Collapsed;

ShowNeededButtons(\_employeeAccount.RoleId);

}

private void HideAllButtons()

{

ordersButton.Visibility = Visibility.Collapsed;

clientsButton.Visibility = Visibility.Collapsed;

storeButton.Visibility = Visibility.Collapsed;

scheduleButton.Visibility = Visibility.Collapsed;

reportsButton.Visibility = Visibility.Collapsed;

serviceButton.Visibility = Visibility.Collapsed;

employeeButton.Visibility = Visibility.Collapsed;

}

private void ShowNeededButtons(int role)

{

switch (role)

{

case 1:

ordersButton.Visibility = Visibility.Visible;

clientsButton.Visibility = Visibility.Visible;

storeButton.Visibility = Visibility.Visible;

break;

case 2:

ordersButton.Visibility = Visibility.Visible;

clientsButton.Visibility = Visibility.Visible;

scheduleButton.Visibility = Visibility.Visible;

break;

case 3:

ordersButton.Visibility = Visibility.Visible;

reportsButton.Visibility = Visibility.Visible;

break;

case 4:

employeeButton.Visibility = Visibility.Visible;

serviceButton.Visibility = Visibility.Visible;

scheduleButton.Visibility = Visibility.Visible;

break;

case 5:

OrdersMasterPage pg = new OrdersMasterPage();

MainFrame.Content = pg;

break;

}

}

private void StoreButton\_OnClick(object sender, RoutedEventArgs e)

{

MainFrame.Content = new StorePage();

MainButton.Visibility = Visibility.Visible;

HideAllButtons();

}

private void ClientsButton\_OnClick(object sender, RoutedEventArgs e)

{

MainFrame.Content = new ClientsPage();

MainButton.Visibility = Visibility.Visible;

HideAllButtons();

}

private void ReportsButton\_OnClick(object sender, RoutedEventArgs e)

{

MainFrame.Content = new ReportPage();

MainButton.Visibility = Visibility.Visible;

HideAllButtons();

}

}

34. WPF\_UP.csproj

﻿<Project Sdk="Microsoft.NET.Sdk">

<PropertyGroup>

<OutputType>WinExe</OutputType>

<TargetFramework>net9.0-windows</TargetFramework>

<Nullable>enable</Nullable>

<ImplicitUsings>enable</ImplicitUsings>

<UseWPF>true</UseWPF>

</PropertyGroup>

<ItemGroup>

<PackageReference Include="BCrypt" Version="1.0.0" />

<PackageReference Include="BCrypt.Net-Next" Version="4.0.3" />

<PackageReference Include="Microsoft.Office.Interop.Excel" Version="15.0.4795.1001" />

<PackageReference Include="Newtonsoft.Json" Version="13.0.3" />

<PackageReference Include="WindowsAPICodePack-Shell" Version="1.1.1" />

<PackageReference Include="WPF-UI" Version="4.0.0-rc.3" />

</ItemGroup>

</Project>