Федеральное государственное образовательное бюджетное учреждение высшего образования

«ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ

РОССИЙСКОЙ ФЕДЕРАЦИИ»

**Колледж информатики и программирования**

ПРАКТИЧЕСКАЯ РАБОТА №30

по МДК.03.01 Технология разработки программного обеспечения

по теме: «Реализация многостраничного wpf-приложения»

Предметная область: Оформление полиса ОСАГО

Выполнили:

студенты группы 4ПКС-220

Анисковцев И.А.

Беркун С.С.

Боднар А.В.

Иванова М.Ю.

Кравцова А.А.

Проверил:

преподаватель Аксёнова Т.Г.

Москва

2023

**Цель работы:** закрепить практические навыки разработки многостраничного WPF-приложения по описанию предметной области в соответствии с руководством по стилю.

**Ход работы**

Все страницы выполнены в соответствии требованиями по оформлению:

* RGB(0,0,204) (для акцентирования внимания) использован для фона всех кнопок;
* RGB(255, 255,255) (основной фон) использован на всех страницах.

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

Рисунок 1. Диаграмма базы, подключенной к WPF-приложению.

Ниже представлена страница авторизации для пользователей.

Изображение выглядит как кот, текст, снимок экрана

Автоматически созданное описание

Рисунок 2. Авторизация пользователей.

После успешного прохождения авторизации идёт разграничение прав доступа. Страница водителя представлена ниже.

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рисунок 3. Страница водителя.

Согласно диаграмме вариантов использования, представленной в предыдущей практической работе, у водителя нет иных функциональных возможностей, только просмотр своих данных. В связи с этим, пользователю с ролью «Водитель» доступна только одна страница.

Основные функциональные возможности системы «Полис ОСАГО» раскрыты в роли «Страховой агент».

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 4. Главная страница страхового агента.

Из данной страницы можно переместиться на страницу регистрации водителей, проиллюстрированной ниже:

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 5. Страница регистрации водителей.

Для регистрации водителей, необходим его id, которые можно посмотреть, нажав на кнопку «Водители» главной страницы страхового агента.

Изображение выглядит как текст, электроника, снимок экрана, дисплей

Автоматически созданное описание

Рисунок 6. Страница, описывающая данные водителей.

При редактировании или добавлении данных мы переносимся на следующую страницу:

Изображение выглядит как текст, снимок экрана, дисплей, число

Автоматически созданное описание

Рисунок 7. Страница изменения/добавления информации о водителях.

По аналогии с данными водителей, в системе описаны такие объекты как «Автомобили» и «Полисы ОСАГО».

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рисунок 8. Страница, описывающая информацию о зарегистрированных в системе автомобилях.

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рисунок 9. Страница изменения/добавления информации об автомобилях.

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рисунок 10. Страница, описывающая информацию об оформленных полисах ОСАГО.

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 11. Страница для оформления полиса ОСАГО.

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Рисунок 12. Успешное оформление полиса ОСАГО.

Для демонстрации листинга самой программы выбраны страницы «Авторизация», «Автомобили» и «Оформление страхового полиса», в достаточной мере демонстрирующие все функции системы в рамках предметной области.

XAML страницы авторизации:

|  |
| --- |
| <Page x:Class="OsaGo.Pages.Page1"  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:OsaGo.Pages"  mc:Ignorable="d"  d:DesignHeight="500" d:DesignWidth="800"  Title="Авторизация">  <Grid>  <Border Style="{StaticResource StyleBorder}">  <Image Style="{StaticResource Background}"/>  </Border>  <TextBox Text="Авторизация" Style="{StaticResource StyleTytle}" Margin="280,59,280,390"/>  <Label Content="Введите логин" Name="txtHintLogin" Style="{StaticResource StyleLabel}" Margin="96,165,569,305" RenderTransformOrigin="0.451,1.296"/>  <TextBox x:Name ="TextBoxLogin" Style="{StaticResource StyleBlock}" ToolTip="Введите логин аккаунта" Margin="85,160,465,300" TextChanged="TextBoxLogin\_Changed" Cursor="Hand"/>  <Label Content="Введите пароль" Name="txtHintPassword" Style="{StaticResource StyleLabel}" Margin="96,255,569,215"/>  <PasswordBox x:Name ="PasswordBox" Style="{StaticResource StylePassword}" PasswordChanged="PasswordBox\_Changed" ToolTip ="Введите пароль от аккаунта" Margin="85,250,465,210" Cursor="Hand"/>  <Button Content="Войти" Style="{StaticResource ButtonStyle}" Margin="85,335,465,125" Click="ButtonEnter\_Click"/>  </Grid>  </Page> |

XAML.CS страницы авторизации:

|  |
| --- |
| using OsaGo.DataBase;  using System;  using System.Collections.Generic;  using System.Data;  using System.Linq;  using System.Security.Cryptography;  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 OsaGo.Pages  {  public partial class Page1 : Page  {  public static Users UserUI;  public Page1()  {  InitializeComponent();  }  private void PasswordBox\_Changed(object sender, RoutedEventArgs e)  {  txtHintPassword.Visibility = Visibility.Visible;  if (PasswordBox.Password.Length > 0)  {  txtHintPassword.Visibility = Visibility.Hidden;  }  }  private void TextBoxLogin\_Changed(object sender, RoutedEventArgs e)  {  txtHintLogin.Visibility = Visibility.Visible;  if (TextBoxLogin.Text.Length > 0)  {  txtHintLogin.Visibility = Visibility.Hidden;  }  }  private void ButtonEnter\_Click(object sender, RoutedEventArgs e)  {  AuthTestSuccess(TextBoxLogin.Text, PasswordBox.Password);  }  public bool AuthTestSuccess(string login, string password)  {  if (string.IsNullOrEmpty(login) || string.IsNullOrEmpty(password))  {  MessageBox.Show(login, password);  return false;  }  using (var db = new Entitie())  {  var user = db.Users.  AsNoTracking().  FirstOrDefault(u => u.Login == login && u.Password == password);  if (user == null)  {  MessageBox.Show("Пользователь с такими данными не найден!");  return false;  }  MessageBox.Show("Пользователь успешно найден!");  TextBoxLogin.Clear();  PasswordBox.Clear();  UserUI = user;  switch (user.Role.Trim())  {  case "Страховой агент":  NavigationService?.Navigate(new Agenty());  break;  case "Водитель":  NavigationService?.Navigate(new Driver());  break;  }  return true;  }  }  internal static DataTable Select(object value)  {  throw new NotImplementedException();  }  }  } |

XAML страницы, выводящей информацию об автомобилях:

|  |
| --- |
| <Page x:Class="OsaGo.Pages.Agent.CarsAgent"  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:OsaGo.Pages.Agent"  mc:Ignorable="d"  d:DesignHeight="500" d:DesignWidth="800"  Title="Автомобили"  IsVisibleChanged="Page\_IsVisibleChanged">  <Grid>  <Grid.RowDefinitions>  <RowDefinition Height="377\*"/>  <RowDefinition Height="73\*"/>  </Grid.RowDefinitions>  <DataGrid Grid.Row="0" Name="DataGridCar" AutoGenerateColumns="False" IsReadOnly="True">  <DataGrid.Columns>  <DataGridTextColumn Header="Id" Binding="{Binding Id}" Width="20"/>  <DataGridTextColumn Header="Категория" Binding="{Binding Category}" Width="70"/>  <DataGridTextColumn Header="Марка" Binding="{Binding Mark}" Width="70"/>  <DataGridTextColumn Header="Модель" Binding="{Binding Model}" Width="80"/>  <DataGridTextColumn Header="Год выпуска" Binding="{Binding Year}" Width="90"/>  <DataGridTextColumn Header="Серийный номер" Binding="{Binding Number}" Width="125"/>  <DataGridTextColumn Header="СТС" Binding="{Binding STSNumber}" Width="70"/>  <DataGridTextColumn Header="ПТС" Binding="{Binding PTSNumber}" Width="70"/>  <DataGridTemplateColumn Width="\*">  <DataGridTemplateColumn.CellTemplate>  <DataTemplate>  <Button Name="ButtonEdit" Style="{StaticResource ButtonStyle}" Content="Редактировать" Click="ButtonEdit\_Click" Width="180" Height="35"></Button>  </DataTemplate>  </DataGridTemplateColumn.CellTemplate>  </DataGridTemplateColumn>  </DataGrid.Columns>  </DataGrid>  <Button Grid.Row="1" Name="ButtonAdd" Style="{StaticResource ButtonStyle}" Content="Добавить" HorizontalAlignment="Left" Click="ButtonAdd\_Click" Margin="100,20,0,21"/>  <Button Grid.Row="1" Name="ButtonDel" Style="{StaticResource ButtonStyle}" Content="Удалить" Click="ButtonDel\_Click" Margin="450,20,100,21"/>  <TextBlock Margin="747,15,23,15" Width="30" Grid.Row="1" FontSize="30">  <Hyperlink NavigateUri = "Agent.xaml" Cursor="Hand">  <TextBlock Text="&#10149;"/>  </Hyperlink>  </TextBlock>  </Grid>  </Page> |

XAML.CS страницы, выводящей информацию об автомобилях:

|  |
| --- |
| using OsaGo.DataBase;  using System;  using System.Collections.Generic;  using System.Linq;  using System.Runtime.ConstrainedExecution;  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 OsaGo.Pages.Agent  {  public partial class CarsAgent : Page  {  public CarsAgent()  {  InitializeComponent();  DataGridCar.ItemsSource = Entitie.GetContext().Cars.ToList();  }  private void ButtonDel\_Click(object sender, RoutedEventArgs e)  {  var carsForRemoving = DataGridCar.SelectedItems.Cast<Cars>().ToList();  if (MessageBox.Show($"Вы точно хотите удалить записи в количестве {carsForRemoving.Count()} элементов?", "Внимание",  MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)  try  {  Entitie.GetContext().Cars.RemoveRange(carsForRemoving);  Entitie.GetContext().SaveChanges();  MessageBox.Show("Данные успешно удалены!");  DataGridCar.ItemsSource = Entitie.GetContext().Cars.ToList();  }  catch (Exception ex)  {  MessageBox.Show(ex.Message.ToString());  }  }  private void ButtonAdd\_Click(object sender, RoutedEventArgs e)  {  NavigationService?.Navigate(new CarsAdd(null));  }  private void ButtonEdit\_Click(object sender, RoutedEventArgs e)  {  NavigationService.Navigate(new CarsAdd((sender as Button).DataContext as Cars));  }  private void Page\_IsVisibleChanged(object sender, DependencyPropertyChangedEventArgs e)  {  if (Visibility == Visibility.Visible)  {  Entitie.GetContext().ChangeTracker.Entries().ToList().ForEach(x => x.Reload());  DataGridCar.ItemsSource = Entitie.GetContext().Cars.ToList();  }  }  }  } |

XAML страницы для оформление страхового полиса:

|  |
| --- |
| <Page x:Class="OsaGo.Pages.Agent.PoliciesAdd"  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:OsaGo.Pages.Agent"  mc:Ignorable="d"  d:DesignHeight="500" d:DesignWidth="800"  Title="PoliciesAdd">  <Grid>  <Border Style="{StaticResource StyleBorder}">  </Border>  <TextBox Text="Оформление страхового полиса" Style="{StaticResource StyleTytle}" Margin="440,146,70,196" TextWrapping="Wrap"/>  <TextBox Style="{StaticResource StyleBlock}" Margin="88,61,462,399" IsReadOnly="True"/>  <Label Content="Выберите id водителя:" Style="{StaticResource StyleLabel}" Margin="88,66,462,404" Width="NaN" FontSize="12"/>  <ComboBox Text="{Binding DriverId}" Height="40" Margin="277,61,419,0" Name="CmbDr" DisplayMemberPath="Id" FontSize="15" VerticalAlignment="Top"/>  <TextBox Style="{StaticResource StyleBlock}" Margin="88,125,462,335" IsReadOnly="True"/>  <Label Content="Выберите id автомобиля:" Style="{StaticResource StyleLabel}" Margin="88,131,462,339" Width="NaN" FontSize="12"/>  <ComboBox Text="{Binding CarId}" Height="40" Margin="277,125,419,0" Name="CmbCar" DisplayMemberPath="Id" FontSize="15" VerticalAlignment="Top"/>  <Label Content="Введите страховую компанию" Style="{StaticResource StyleLabel}" Margin="88,196,460,274" Width="NaN" FontSize="12"/>  <TextBox Style="{StaticResource StyleBlock}" Margin="88,191,419,269" Width="NaN" IsReadOnly="True"/>  <ComboBox Text="{Binding InsuranceCompany}" Height="40" Margin="277,191,419,0" Name="CmbCompany" DisplayMemberPath="Title" FontSize="15" VerticalAlignment="Top"/>  <TextBox Style="{StaticResource StyleBlock}" Margin="88,254,419,206" Width="NaN" IsReadOnly="True"/>  <Label Content="Введите дату регистрации:" Style="{StaticResource StyleLabel}" Margin="88,257,460,213" Width="NaN" FontSize="12"/>  <DatePicker Text="{Binding RegistrationDate}" Name="ReggiDate" SelectedDateFormat="Short" Margin="277,254,419,206"/>  <TextBox Style="{StaticResource StyleBlock}" Margin="88,315,419,145" Width="NaN" IsReadOnly="True"/>  <Label Content="Введите срок действия:" Name="txtHintValidaty" Style="{StaticResource StyleLabel}" Margin="88,319,460,151" Width="NaN" FontSize="12"/>  <DatePicker Text="{Binding ValidityPeriod}" Name="VallyDate" SelectedDateFormat="Short" Margin="277,315,419,145"/>  <Label Content="Введите стоимость" Name="txtHintPrice" Style="{StaticResource StyleLabel}" Margin="88,384,460,86" Width="NaN" FontSize="12"/>  <TextBox x:Name ="TextBoxPrice" Text="{Binding Price}" Style="{StaticResource StyleBlock}" Margin="88,379,419,81" TextChanged="TextBoxPrice\_Changed" Cursor="Hand" Width="NaN"/>  <Button Content="Оформить полис" Style="{StaticResource ButtonStyle}" Margin="438,379,0,81" HorizontalAlignment="Left" Width="292" Click="Formalization\_Click"/>  <TextBlock Margin="732,34,38,433" Width="30" FontSize="30">  <Hyperlink NavigateUri = "\Pages\Agent\Policies.xaml" Cursor="Hand">  <TextBlock Text="&#10149;"/>  </Hyperlink>  </TextBlock>  </Grid>  </Page> |

XAML.CS страницы для оформление страхового полиса:

|  |
| --- |
| using OsaGo.DataBase;  using System;  using System.Collections.Generic;  using System.Data.Entity.Validation;  using System.Linq;  using System.Runtime.ConstrainedExecution;  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 Poly = OsaGo.DataBase.Policies;  namespace OsaGo.Pages.Agent  {  public partial class PoliciesAdd : Page  {  private Poly \_currentPoly = new Poly();  public PoliciesAdd(Poly selectedPoly)  {  InitializeComponent();  if (selectedPoly != null)  {  \_currentPoly = selectedPoly;  }  DataContext = \_currentPoly;  CmbCar.ItemsSource = Entitie.GetContext().Cars.ToList();  CmbDr.ItemsSource = Entitie.GetContext().Drivers.ToList();  CmbCompany.ItemsSource = Entitie.GetContext().InsuranceCopmanyes.ToList();  }  private void TextBoxPrice\_Changed(object sender, TextChangedEventArgs e)  {  txtHintPrice.Visibility = Visibility.Visible;  if (TextBoxPrice.Text.Length > 0)  {  txtHintPrice.Visibility = Visibility.Hidden;  }  }  private void Formalization\_Click(object sender, RoutedEventArgs e)  {  StringBuilder errors = new StringBuilder();  if (CmbDr.SelectedValue == null)  errors.AppendLine("Выберите Id водителя!");  if (CmbDr.SelectedValue == null)  errors.AppendLine("Выберите Id машины!");  if (CmbCompany.SelectedValue == null)  errors.AppendLine("Выберите страховую компанию!");  if (ReggiDate.SelectedDate.Value == default)  errors.AppendLine("Укажите дату регистрации!");  if (VallyDate.SelectedDate.GetValueOrDefault() == default)  errors.AppendLine("Укажите срок действия!");  if (TextBoxPrice.Text == "0" || string.IsNullOrEmpty(TextBoxPrice.Text))  errors.AppendLine("Введите стоимость!");  if (errors.Length > 0)  {  MessageBox.Show(errors.ToString());  return;  }  if (Convert.ToInt64(\_currentPoly.Number) != 0)  Entitie.GetContext().Policies.Add(\_currentPoly);  try  {  \_currentPoly.Number = (long.Parse(Entitie.GetContext().Policies.OrderByDescending(x => x.Number).First().Number) + 1).ToString();  var driver = Entitie.GetContext().Drivers.Where(x => x.Id == \_currentPoly.DriverId).FirstOrDefault();  var license = Entitie.GetContext().Licenses.Where(x => x.DriverId == driver.Id).FirstOrDefault();  \_currentPoly.LicenseNumber = license.Number;  \_currentPoly.LicenseSeries = license.Series;  Entitie.GetContext().Policies.Add(\_currentPoly);  Entitie.GetContext().SaveChanges();  MessageBox.Show("Данные успешно сохранены!");  }  catch (DbEntityValidationException ex)  {  var errorMessages = ex.EntityValidationErrors  .SelectMany(x => x.ValidationErrors)  .Select(x => x.ErrorMessage);  var fullErrorMessage = string.Join("; ", errorMessages);  var exceptionMessage = string.Concat(ex.Message, " The validation errors are: ", fullErrorMessage);  MessageBox.Show(exceptionMessage.ToString());  }  catch (Exception ex)  {  MessageBox.Show(ex.Message.ToString());  }  }  }  } |

Финальным штрихом работы является добавление иконки:

|  |
| --- |
| Icon="\Images\icon.png" |

Изображение выглядит как кот, текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 13. WPF-приложение с добавленной иконкой.

**Вывод:** былизакреплены практические навыки разработки многостраничного WPF-приложения по описанию предметной области в соответствии с руководством по стилю.