Федеральное государственное образовательное бюджетное учреждение  
высшего образования

«Финансовый университет при Правительстве Российской Федерации»

КОЛЛЕДЖ ИНФОРМАТИКИ И ПРОГРАММИРОВАНИЯ

**ОТЧЕТ**

**по результатам проектирования**

**и разработки программного приложения**

**по профессиональному модулю ПМ.03 Участие в интеграции программных модулей**

**специальности 09.02.03 Программирование в компьютерных системах**

Выполнил:

студент группы 4ПКС-220

Поволяев Д.А

Проверили:

преподаватели

\_\_\_\_\_\_\_\_ / Т.Г. Аксёнова /

\_\_\_\_\_\_\_\_/О.А. Тимакин/

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2023

ОГЛАВЛЕНИЕ

[**ПРОЕКТИРОВАНИЕ ПРОГРАМНОГО ПРИЛОЖЕНИЯ** 3](#_Toc154591076)

[**РУКОВОДСТВО ОПЕРАТОРА** 5](#_Toc154591077)

[**ЛИСТИНГ ПРОГРАММНЫХ МОДУЛЕЙ** 8](#_Toc154591078)

# **ПРОЕКТИРОВАНИЕ ПРОГРАМНОГО ПРИЛОЖЕНИЯ**

В ходе разработки программы были составлены 2 диаграммы ERD-диаграмма (рисунок 1) и диаграмма вариантов использования (рисунок 2)

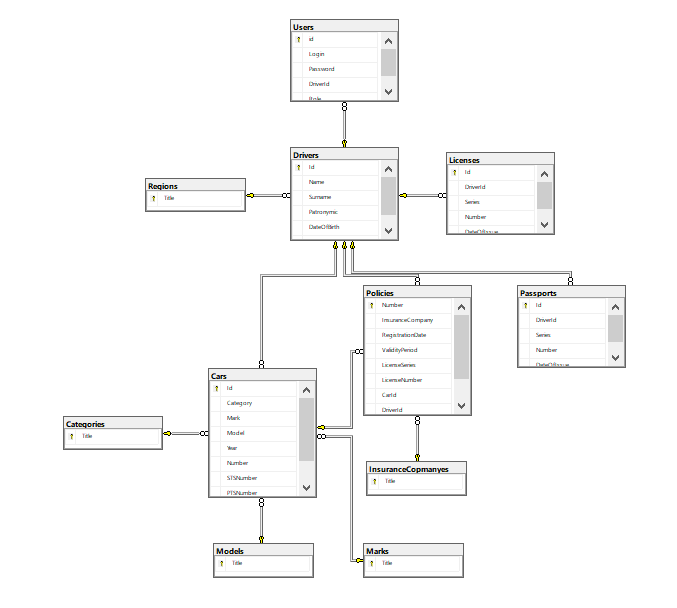


Рисунок 1. ERD-диаграмма

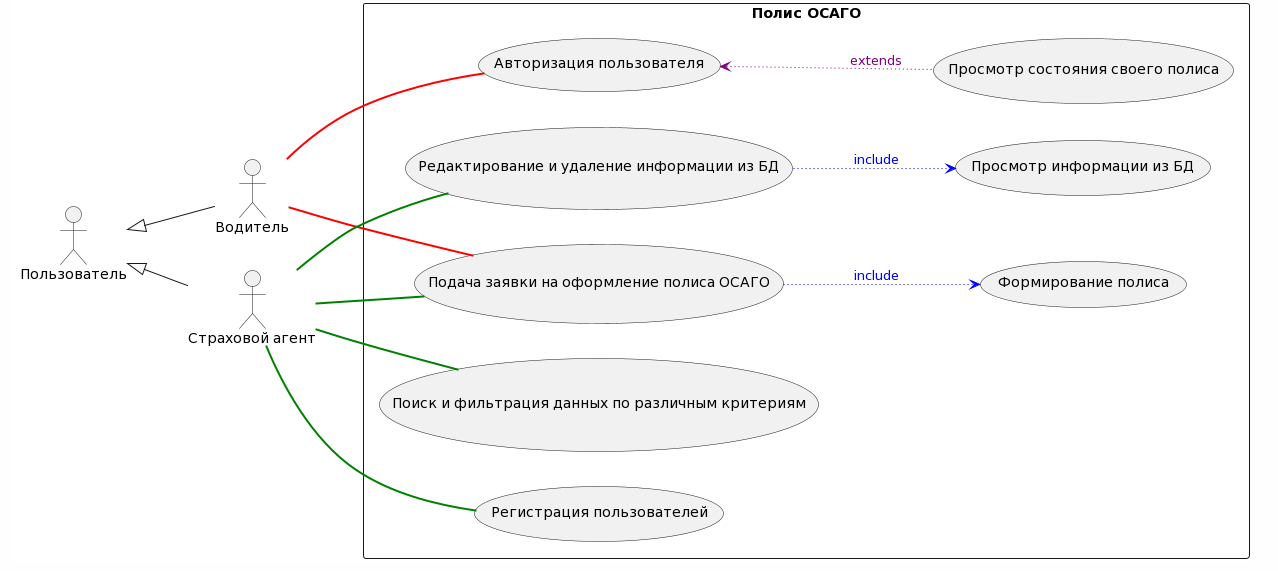


Рисунок 2. Диаграмма вариантов использования

Ниже приведён код для строения диаграммы вариантов использования при помощи платформы PlantUML

|  |
| --- |
| @startuml  left to right direction  rectangle "Полис ОСАГО" {  UC1 as (Авторизация пользователя)  UC2 as (Регистрация пользователей)  UC3 as (Редактирование и удаление информации из БД)  UC4 as (Поиск и фильтрация данных по различным критериям)  UC5 as (Формирование полиса)  UC6 as (Подача заявки на оформление полиса ОСАГО)  UC7 as (Просмотр информации из БД)  UC8 as (Просмотр состояния своего полиса)  }  actor "Пользователь" as user  actor "Водитель" as driver  actor "Страховой агент" as agent  user <|-- agent  user <|-- driver  driver--UC1 #line:red;line.bold;text:red  driver--UC6 #line:red;line.bold;text:red  UC6 ..>UC5 #blue;line.dotted;text:blue :include  agent--UC2 #line:green;line.bold;text:red  agent--UC3 #line:green;line.bold;text:red  agent--UC4 #line:green;line.bold;text:red  agent--UC6 #line:green;line.bold;text:red  UC3 ..>UC7 #blue;line.dotted;text:blue :include  UC1<.. UC8 #purple;line.dotted;text:purple :extends  @enduml |

# **РУКОВОДСТВО ОПЕРАТОРА**

**Назначение программы**

Программа назначена для облегчения процесса страхования ОСАГО, путём возможности предоставить страховому агенту действия по добавлению, редактированию и удалении данных о водителях, а водителям позволяет просмотреть все свои данные о ОСАГО.

**Условия выполнения программы**

Оптимальными условиями для выполнения программы будут являться:

* Установленная операционная система Windows 10;
* Персональный компьютер
* Установленная Visual Studio 2019
* Установленный SQL server Management studio 2019

**Выполнение программы**

Чтобы начать работать с программой нужно запустить программу это можно сделать через Visual Studio 2019 путём нажатия на кнопку пуска (Рисунок 3)



Рисунок 3. Кнопка пуск

Далее откроется программа где взору будет представлена страница авторизации (рисунок 4), на которой нужно будет ввести данные пользователя и нажать на кнопку войти.

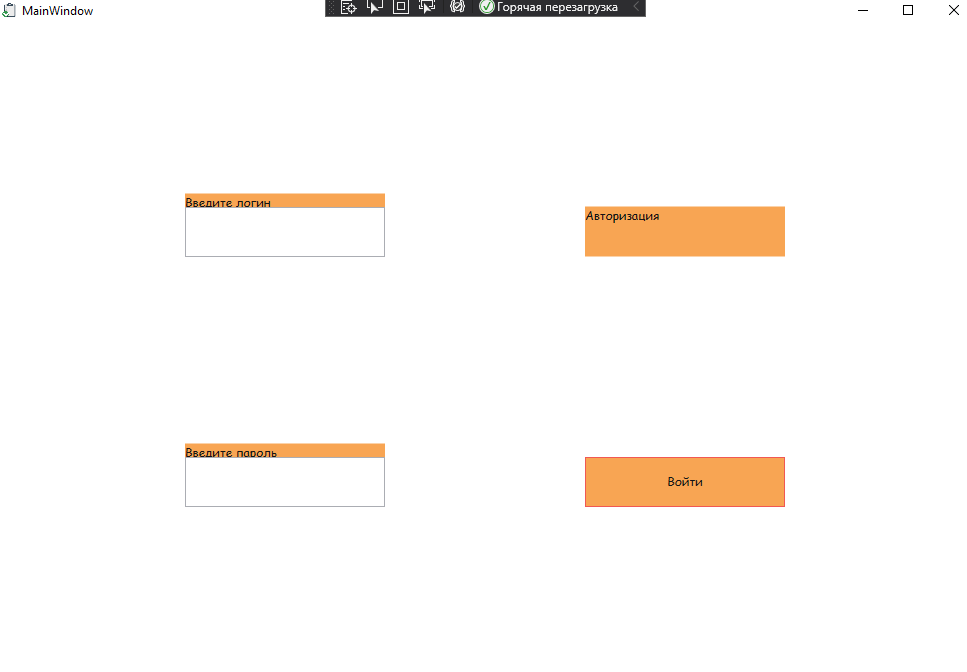


Рисунок 4. Окно авторизации

В зависимости от того что вы написали, будет выдана страница водителя (Рисунок 5) или страница страхового агента (Рисунок 6)

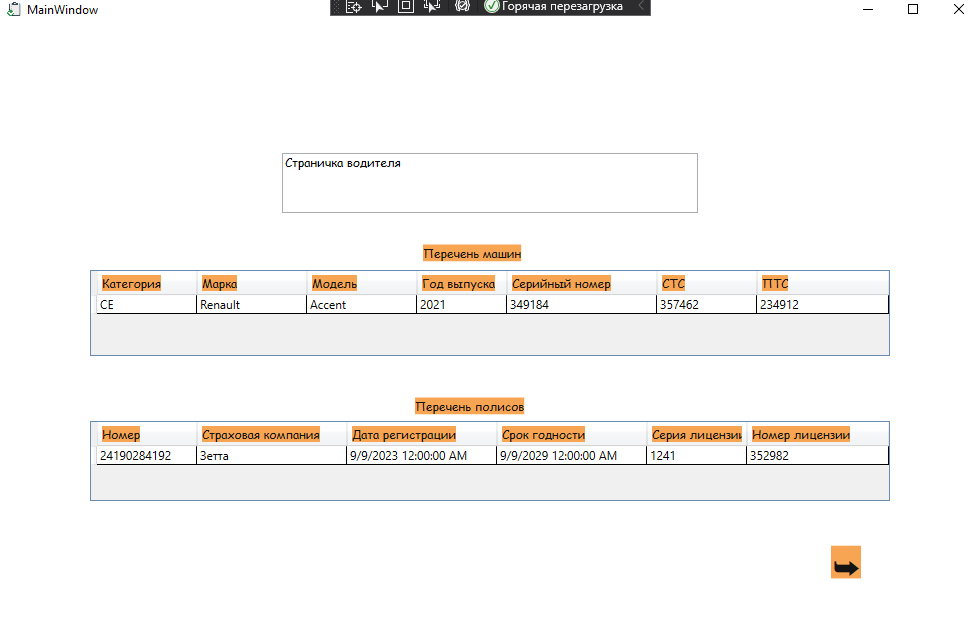


Рисунок 5. Страница водителя

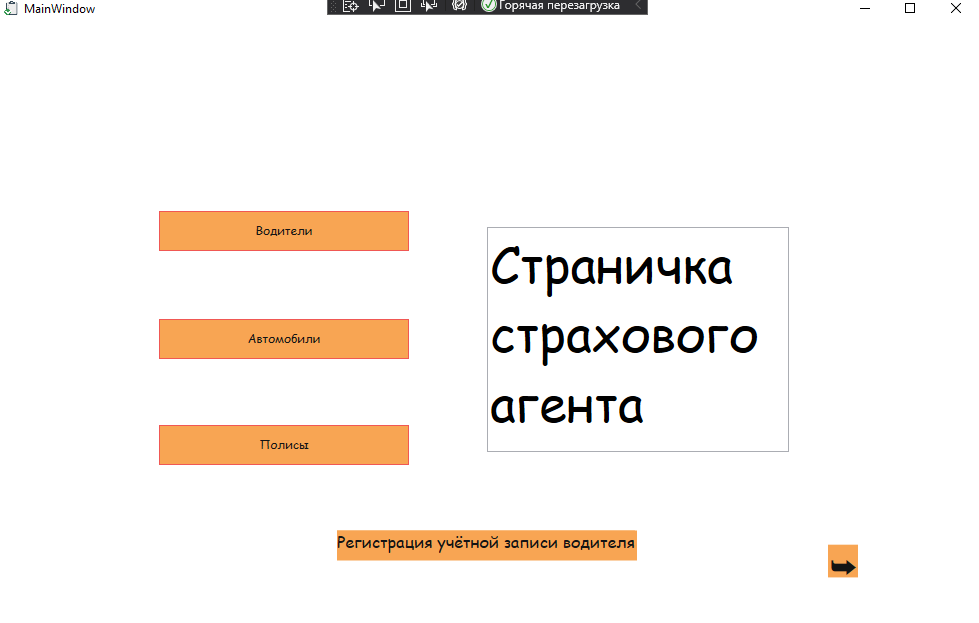


Рисунок 6. Страница страхового агента

При заходе за страхового агента будет возможность нажать на 3 кнопки Водители (Рисунок 7), Автомобили (Рисунок 8) и Полисы (Рисунок 9), тем самым при нажатии на одну из этих кнопок мы можем перейти на страницы который выдадут нам данные.

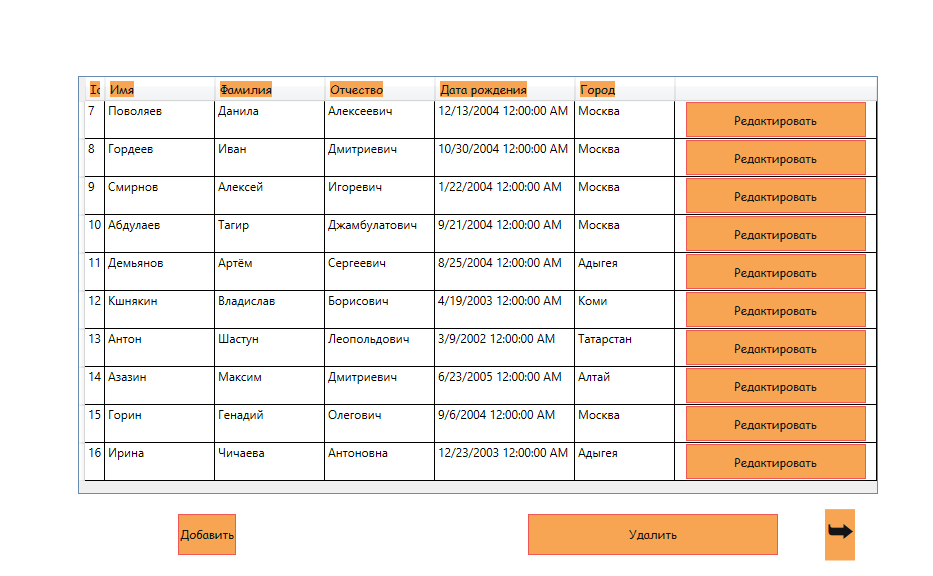


Рисунок 7. Страница данных Водителях

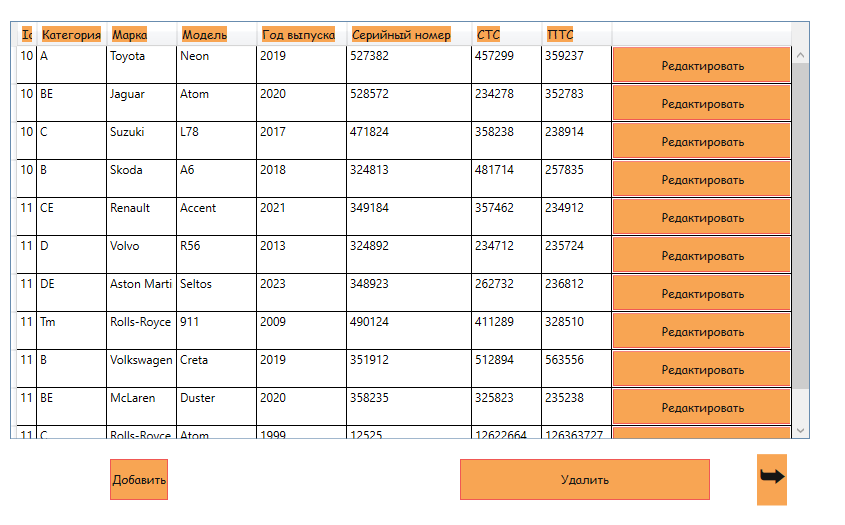


Рисунок 8. Страница данных Автомобилей

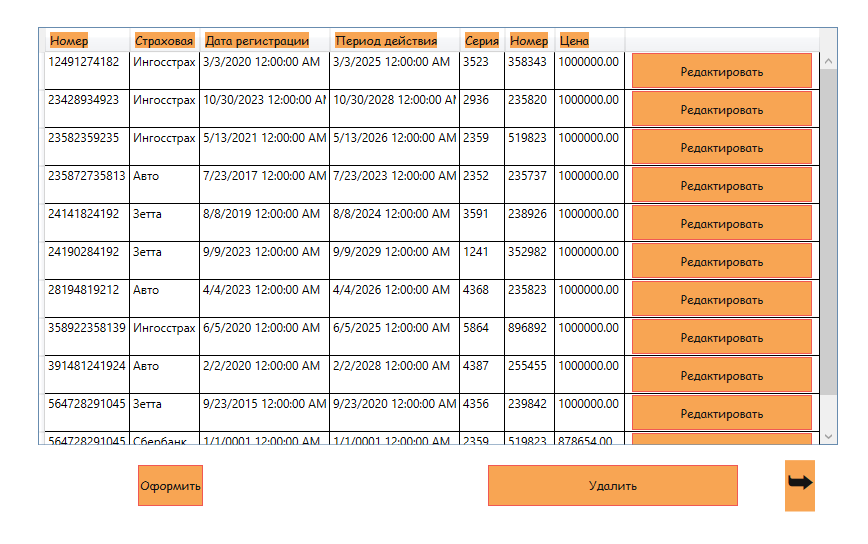


Рисунок 9. Страница данных Полюсов

**Сообщения оператору**

Представленные сообщения оператору – это “Пользователь найден” что обозначает что пользователь был найден, и произведён успешный вход в систему. “Пользователь не найден” обозначает что пользователь не был найден в системе , вам потребуется ввести данные о пользователе ещё раз.

# **ЛИСТИНГ ПРОГРАММНЫХ МОДУЛЕЙ**

Листинг кода Authorization.xaml.cs

|  |
| --- |
| public partial class Page1 : Page  {  public static Users UserUI;  public Page1()  {  InitializeComponent();  }  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();  }  } |

Листинг кода Driver.xaml.cs

|  |
| --- |
| public partial class Driver : Page  {  public Driver()  {  InitializeComponent();  DataGridCar.ItemsSource = Entitie.GetContext().Cars.Where(x => x.DriverId == Page1.UserUI.DriverId).ToList();  DataGridPolice.ItemsSource = Entitie.GetContext().Policies.Where(x => x.DriverId == Page1.UserUI.DriverId).ToList();  }  } |

Листинг кода Agent.xaml.cs

|  |
| --- |
| public partial class Agenty : Page  {  public Agenty()  {  InitializeComponent();  }  private void ButtonClickD(object sender, RoutedEventArgs e)  {  NavigationService.Navigate(new Uri("/Pages/Agent/Drivers.xaml", UriKind.Relative));  }  private void ButtonClickC(object sender, RoutedEventArgs e)  {  NavigationService.Navigate(new Uri("/Pages/Agent/CarsAgent.xaml", UriKind.Relative));  }  private void ButtonClickP(object sender, RoutedEventArgs e)  {  NavigationService.Navigate(new Uri("/Pages/Agent/Policies.xaml", UriKind.Relative));  }  } |

Листинг кода CarsAdd.xaml.cs

|  |
| --- |
| public partial class CarsAdd : Page  {  private Car \_currentCars = new Car();  public CarsAdd(Car selectedCars)  {  InitializeComponent();  if (selectedCars != null)  {  \_currentCars = selectedCars;  }  DataContext = \_currentCars;  CmbCat.ItemsSource = Entitie.GetContext().Categories.ToList();  CmbMark.ItemsSource = Entitie.GetContext().Marks.ToList();  CmbMod.ItemsSource = Entitie.GetContext().Models.ToList();  }  private void ButtonBack\_Click(object sender, RoutedEventArgs e)  {  NavigationService.Navigate(new Uri("/Pages/Agent/CarsAgent.xaml", UriKind.Relative));  }  private void ButtonSave\_Click(object sender, RoutedEventArgs e)  {  StringBuilder errors = new StringBuilder();  if (\_currentCars.Category == null)  errors.AppendLine("Выберите категорию!");  if (\_currentCars.Mark==null)  errors.AppendLine("Укажите название марки!");  if (\_currentCars.Model==null)  errors.AppendLine("Укажите название модели!");  if (string.IsNullOrWhiteSpace(\_currentCars.Year))  errors.AppendLine("Укажите дату выпуска автомобиля!");  if (string.IsNullOrWhiteSpace(\_currentCars.Number))  errors.AppendLine("Укажите гос.номер!");  if (string.IsNullOrWhiteSpace(\_currentCars.STSNumber))  errors.AppendLine("Укажите номер СТС");  if (string.IsNullOrWhiteSpace(\_currentCars.PTSNumber))  errors.AppendLine("Укажите номер ПТС");  if (errors.Length > 0)  {  MessageBox.Show(errors.ToString());  return;  }  if (\_currentCars.Id == 0)  Entitie.GetContext().Cars.Add(\_currentCars);  try  {  Entitie.GetContext().SaveChanges();  MessageBox.Show("Данные успешно сохранены!");  }  catch (Exception ex)  {  MessageBox.Show(ex.Message.ToString());  }  }  } |

Листинг кода CarsAgent.xaml.cs

|  |
| --- |
| 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();  }  }  } |

Листинг кода Drivers.xaml.cs

|  |
| --- |
| public partial class Drivers : Page  {  public Drivers()  {  InitializeComponent();  DataGridDriver.ItemsSource = Entitie.GetContext().Drivers.ToList();  }  private void ButtonAdd\_Click(object sender, RoutedEventArgs e)  {  NavigationService.Navigate(new DriversAdd(null));  }  private void ButtonEdit\_Click(object sender, RoutedEventArgs e)  {  NavigationService.Navigate(new DriversAdd((sender as Button).DataContext as DriverBase));  }  private void ButtonDel\_Click(object sender, RoutedEventArgs e)  {  var driversForRemoving = DataGridDriver.SelectedItems.Cast<DriverBase>().ToList();  if (MessageBox.Show($"Вы точно хотите удалить записи в количестве {driversForRemoving.Count()} элементов?", "Внимание",  MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)  try  {  Entitie.GetContext().Drivers.RemoveRange(driversForRemoving);  Entitie.GetContext().SaveChanges();  MessageBox.Show("Данные успешно удалены!");  DataGridDriver.ItemsSource = Entitie.GetContext().Drivers.ToList();  }  catch (Exception ex)  {  MessageBox.Show(ex.Message.ToString());  }  }  } |

Листинг кода DriversAdd.xaml.cs

|  |
| --- |
| public partial class DriversAdd : Page  {  private DriverBase \_currentDriver = new DriverBase();  public DriversAdd(DriverBase selectedDriver)  {  InitializeComponent();  if (selectedDriver != null)  {  \_currentDriver = selectedDriver;  }  DataContext = \_currentDriver;  CmbCity.ItemsSource = Entitie.GetContext().Regions.ToList();  }  private void ButtonBack\_Click(object sender, RoutedEventArgs e)  {  NavigationService.Navigate(new Uri("/Pages/Agent/Drivers.xaml", UriKind.Relative));  }  private void ButtonSave\_Click(object sender, RoutedEventArgs e)  {  StringBuilder errors = new StringBuilder();  if (string.IsNullOrWhiteSpace(\_currentDriver.Name))  errors.AppendLine("Введите имя!");  if (string.IsNullOrWhiteSpace(\_currentDriver.Surname))  errors.AppendLine("Введите фамилию!");  if (string.IsNullOrWhiteSpace(\_currentDriver.Patronymic))  errors.AppendLine("Введите отчество!");  if (\_currentDriver.DateOfBirth == null)  errors.AppendLine("Введите дату рождения!");  if (\_currentDriver.Region == null)  errors.AppendLine("Укажите город!");  if (errors.Length > 0)  {  MessageBox.Show(errors.ToString());  return;  }  if (\_currentDriver.Id == 0)  Entitie.GetContext().Drivers.Add(\_currentDriver);  try  {  Entitie.GetContext().SaveChanges();  MessageBox.Show("Данные успешно сохранены!");  }  catch (Exception ex)  {  MessageBox.Show(ex.Message.ToString());  }  }  } |

Листинг кода Policies.xaml.cs

|  |
| --- |
| public partial class Policies : Page  {  public Policies()  {  InitializeComponent();  DataGridPoly.ItemsSource = Entitie.GetContext().Policies.ToList();  }  private void ButtonEdit\_Click(object sender, RoutedEventArgs e)  {  NavigationService.Navigate(new PoliciesAdd((sender as Button).DataContext as Poly));  }  private void ButtonAdd\_Click(object sender, RoutedEventArgs e)  {  NavigationService.Navigate(new PoliciesAdd(null));  }  private void ButtonDel\_Click(object sender, RoutedEventArgs e)  {  var polyForRemoving = DataGridPoly.SelectedItems.Cast<Poly>().ToList();  if (MessageBox.Show($"Вы точно хотите удалить записи в количестве {polyForRemoving.Count()} элементов?", "Внимание",  MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)  try  {  Entitie.GetContext().Policies.RemoveRange(polyForRemoving);  Entitie.GetContext().SaveChanges();  MessageBox.Show("Данные успешно удалены!");  DataGridPoly.ItemsSource = Entitie.GetContext().Policies.ToList();  }  catch (Exception ex)  {  MessageBox.Show(ex.Message.ToString());  }  }  } |

Листинг кода PoliciesAdd.xaml.cs

|  |
| --- |
| 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());  }  }  } |