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

А.В. Боднар

Проверили:

преподаватели

\_\_\_\_\_\_\_\_ / Т.Г. Аксёнова /

\_\_\_\_\_\_\_\_/О.А. Тимакин/

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

2023

Содержание

[**Проектирование программного приложения** 3](#_Toc154589364)

[**Руководство оператора** 4](#_Toc154589365)

[**Листинг программных модулей** 10](#_Toc154589366)

# **Проектирование программного приложения**

На рисунке 1 представлена ER-диаграмма для разработки программного обеспечения предметной области «Оформление полиса ОСАГО».

Изображение выглядит как текст, снимок экрана, число, диаграмма

Автоматически созданное описание

Рисунок 1. ER-диаграмма базы данных «Полисы ОСАГО»

В инструментальном средстве «PlantText» в рамках предметной области была создана диаграмма вариантов использования, представленная на рисунке 2.

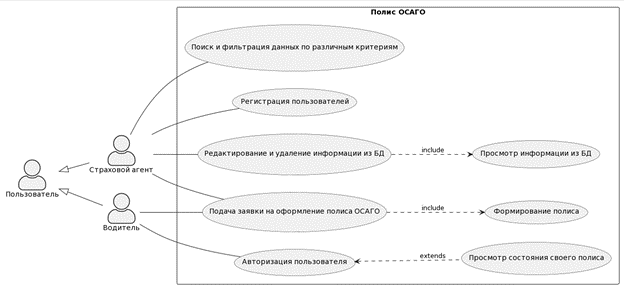


Рисунок 2. Диаграмма вариантов использования

# **Руководство оператора**

Назначение программы

Главное назначение программы, созданной в рамках выполнения задания «Оформление полиса ОСАГО», - это возможность для страховых агентов работать с данными о водителях (ФИО водителя, возраст, их машины, полисы), а также способность регистрировать новых водителей и их машины в системе, но главной задачей является само оформление полисов ОСАГО для водителей по ID водителя и его машины.

Водители сами же могут регистрироваться как пользователи для возможности отслеживания состояния своих полисов и привязанных к ним машин.

Условия выполнения программы

Рекомендуемые характеристики аппаратного и программного обеспечения компьютерной системы:

1. Процессор Intel Core i5 или AMD Ryzen 5 и последующие модели
2. Оперативная память 4 Гбайт и выше (рекомендуется 8 Гбайт)
3. Жесткий диск или твердотельный накопитель 40Гб и выше
4. Версия ОС: Windows 7, 10 и выше.

Выполнение программы

При запуске программы в первую очередь пользователя встречает окно авторизации, представленное на рисунке 3.

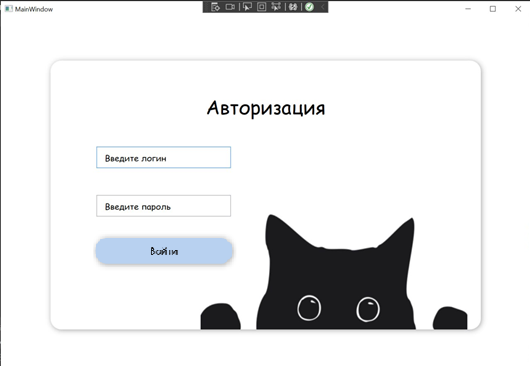


Рисунок 3. Окно авторизации

Если наш пользователь – страховой агент, то после авторизации ему открывается следующее стандартное окно для страхового агента:

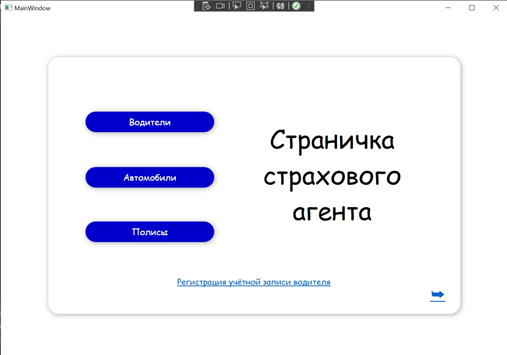


Рисунок 4. Окно страхового агента

Далее страховой агент может как просмотреть информацию о водителях, машинах и полисах, так и редактировать ее, удалять и добавлять. Дополнением так же является возможность регистрировать водителей как новых пользователей системы.

Пример того, как страховой просматривает информацию находится на рисунке 5.

Изображение выглядит как текст, электроника, снимок экрана, дисплей

Автоматически созданное описание

Рисунок 5. Страховой агент просматривает информацию о водителях

Например, так выглядят странички с добавлением новой информации о водителях, машинах и полисах на рисунках 6, 7, 8.

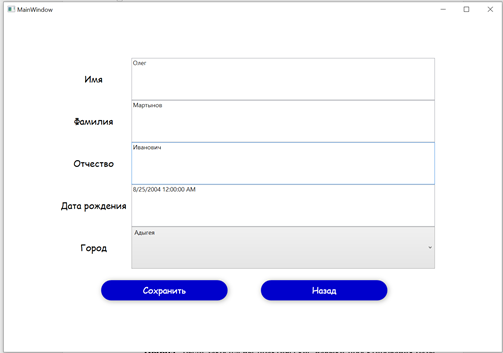


Рисунок 6. Добавление нового водителя

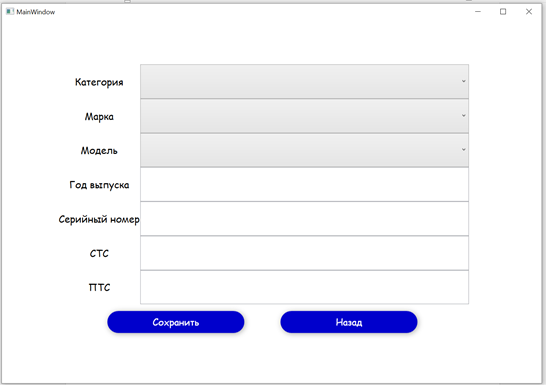


Рисунок 7. Добавление новой машины

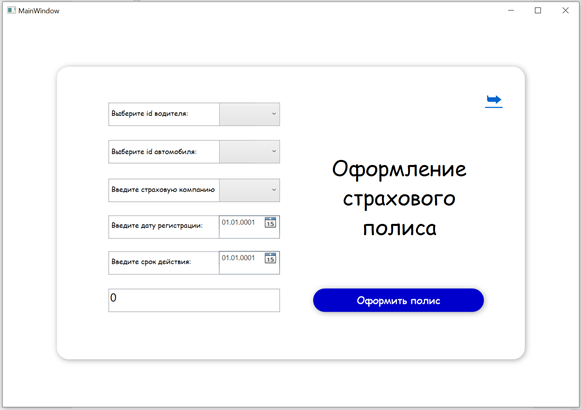


Рисунок 8. Оформление нового полиса ОСАГО

Регистрировать водителей как пользователей системы можно только после того, как внесена их информация страховым агентом. Так, к примеру, выглядит само окно регистрации водителя.

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

Автоматически созданное описание

Рисунок 9. Регистрация водителя в системе

Так же пользователь может войти как водитель и тогда у него будет только одна доступная страница для отслеживания информации о машинах и полисах:

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

Автоматически созданное описание

Рисунок 10. Страница водителя

Сообщения оператору

Пример сообщения оператору, а точнее страховому агенту представлен на рисунке 11. Данное сообщение вылезает только при успешном и правильном оформлении полиса ОСАГО.

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

Автоматически созданное описание

Рисунок 11. Сообщение об успешном оформлении

Так же во время выполнения программы могут встречаться и другие сообщения для оператора, но все они преимущественно только при добавлении/редактировании данных при несоблюдении требований (К примеру, для пароля, чья длина должна быть больше шести символов).

# **Листинг программных модулей**

Код авторизации:

|  |
| --- |
| namespace OsaGo.Pages  {  /// <summary>  /// Логика взаимодействия для Page1.xaml  /// </summary>  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 Entitie1())  {  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();  }  }  } |

Страница страхового агента:

|  |
| --- |
| namespace OsaGo.Pages  {  /// <summary>  /// Логика взаимодействия для Agent.xaml  /// </summary>  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));  }  }  } |

Страница просмотра информации о машинах:

|  |
| --- |
| namespace OsaGo.Pages.Agent  {  /// <summary>  /// Логика взаимодействия для CarsAgent.xaml  /// </summary>  public partial class CarsAgent : Page  {  public CarsAgent()  {  InitializeComponent();  DataGridCar.ItemsSource = Entitie1.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  {  Entitie1.GetContext().Cars.RemoveRange(carsForRemoving);  Entitie1.GetContext().SaveChanges();  MessageBox.Show("Данные успешно удалены!");  DataGridCar.ItemsSource = Entitie1.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)  {  Entitie1.GetContext().ChangeTracker.Entries().ToList().ForEach(x => x.Reload());  DataGridCar.ItemsSource = Entitie1.GetContext().Cars.ToList();  }  }  }  } |

Страница добавления машины:

|  |
| --- |
| namespace OsaGo.Pages.Agent  {  /// <summary>  /// Логика взаимодействия для CarsAdd.xaml  /// </summary>  public partial class CarsAdd : Page  {  private Car \_currentCars = new Car();  public CarsAdd(Car selectedCars)  {  InitializeComponent();  if (selectedCars != null)  {  \_currentCars = selectedCars;  }  DataContext = \_currentCars;  CmbCat.ItemsSource = Entitie1.GetContext().Categories.ToList();  CmbMark.ItemsSource = Entitie1.GetContext().Marks.ToList();  CmbMod.ItemsSource = Entitie1.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)  Entitie1.GetContext().Cars.Add(\_currentCars);  try  {  Entitie1.GetContext().SaveChanges();  MessageBox.Show("Данные успешно сохранены!");  }  catch (Exception ex)  {  MessageBox.Show(ex.Message.ToString());  }  }  }  } |

Страница просмотра информации о водителях:

|  |
| --- |
| namespace OsaGo.Pages.Agent  {  /// <summary>  /// Логика взаимодействия для Drivers.xaml  /// </summary>  public partial class Drivers : Page  {  public Drivers()  {  InitializeComponent();  DataGridDriver.ItemsSource = Entitie1.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  {  Entitie1.GetContext().Drivers.RemoveRange(driversForRemoving);  Entitie1.GetContext().SaveChanges();  MessageBox.Show("Данные успешно удалены!");  DataGridDriver.ItemsSource = Entitie1.GetContext().Drivers.ToList();  }  catch (Exception ex)  {  MessageBox.Show(ex.Message.ToString());  }  }  }  } |

Страница добавления информации о водителе:

|  |
| --- |
| namespace OsaGo.Pages.Agent  {  /// <summary>  /// Логика взаимодействия для DriversAdd.xaml  /// </summary>  public partial class DriversAdd : Page  {  private DriverBase \_currentDriver = new DriverBase();  public DriversAdd(DriverBase selectedDriver)  {  InitializeComponent();  if (selectedDriver != null)  {  \_currentDriver = selectedDriver;  }  DataContext = \_currentDriver;  CmbCity.ItemsSource = Entitie1.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)  Entitie1.GetContext().Drivers.Add(\_currentDriver);  try  {  Entitie1.GetContext().SaveChanges();  MessageBox.Show("Данные успешно сохранены!");  }  catch (Exception ex)  {  MessageBox.Show(ex.Message.ToString());  }  }  }  } |

Страница просмотра информации о полисах:

|  |
| --- |
| namespace OsaGo.Pages.Agent  {  /// <summary>  /// Логика взаимодействия для Policies.xaml  /// </summary>  public partial class Policies : Page  {  public Policies()  {  InitializeComponent();  DataGridPoly.ItemsSource = Entitie1.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  {  Entitie1.GetContext().Policies.RemoveRange(polyForRemoving);  Entitie1.GetContext().SaveChanges();  MessageBox.Show("Данные успешно удалены!");  DataGridPoly.ItemsSource = Entitie1.GetContext().Policies.ToList();  }  catch (Exception ex)  {  MessageBox.Show(ex.Message.ToString());  }  }  }  } |

Страница добавления нового полиса ОСАГО:

|  |
| --- |
| namespace OsaGo.Pages.Agent  {  /// <summary>  /// Логика взаимодействия для PoliciesAdd.xaml  /// </summary>  public partial class PoliciesAdd : Page  {  private Poly \_currentPoly = new Poly();  public PoliciesAdd(Poly selectedPoly)  {  InitializeComponent();  if (selectedPoly != null)  {  \_currentPoly = selectedPoly;  }  DataContext = \_currentPoly;  CmbCar.ItemsSource = Entitie1.GetContext().Cars.ToList();  CmbDr.ItemsSource = Entitie1.GetContext().Drivers.ToList();  CmbCompany.ItemsSource = Entitie1.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)  Entitie1.GetContext().Policies.Add(\_currentPoly);  try  {  \_currentPoly.Number = (long.Parse(Entitie1.GetContext().Policies.OrderByDescending(x => x.Number).First().Number) + 1).ToString();  var driver = Entitie1.GetContext().Drivers.Where(x => x.Id == \_currentPoly.DriverId).FirstOrDefault();  var license = Entitie1.GetContext().Licenses.Where(x => x.DriverId == driver.Id).FirstOrDefault();  \_currentPoly.LicenseNumber = license.Number;  \_currentPoly.LicenseSeries = license.Series;  Entitie1.GetContext().Policies.Add(\_currentPoly);  Entitie1.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());  }  }  }  } |

Страница регистрации нового водителя:

|  |
| --- |
| namespace OsaGo.Pages  {  /// <summary>  /// Логика взаимодействия для Page2.xaml  /// </summary>  public partial class Page2 : Page  {  public Page2()  {  InitializeComponent();  Cmb.ItemsSource = Entitie1.GetContext().Drivers.ToList();  }  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)  {  if (string.IsNullOrEmpty(TextBoxLogin.Text) ||  string.IsNullOrEmpty(PasswordBox.Password) || string.IsNullOrEmpty(PasswordBoxCheck.Password))  {  MessageBox.Show("Введите все данные!");  }    //DataTable dt\_user = Authorization.Select("SELECT \* FROM [dbo].[Users] WHERE [Login] = '" + TextBoxLogin.Text);  //if (dt\_user.Rows.Count > 0) // если такая запись существует  //{  // MessageBox.Show("Пользователь с таким логином уже есть");  //}  if (PasswordBox.Password.Length >= 6)  {  bool en = true; // английская раскладка  bool number = false; // цифра  for (int i = 0; i < PasswordBox.Password.Length; i++) // перебираем символы  {  if (PasswordBox.Password[i] >= 'А' && PasswordBox.Password[i] <= 'Я') en = false; // если русская раскладка  if (PasswordBox.Password[i] >= '0' && PasswordBox.Password[i] <= '9') number = true; // цифры  }  if (!en)  MessageBox.Show("Доступна только английская раскладка");// выводим сообщение  else if (!number)  MessageBox.Show("Добавьте хотя бы одну цифру"); // выводим сообщение  if (en && number) // проверяем соответствие  {  }  else { MessageBox.Show("Пароль слишком короткий, минимум 6 символов"); }  }  if (PasswordBox.Password == PasswordBoxCheck.Password) // проверка на совпадение паролей  {  MessageBox.Show("Пользователь зарегистрирован");  }  else MessageBox.Show("Пароли не совподают");    Entitie1 db = new Entitie1();  Users userObject = new Users  {  Login = TextBoxLogin.Text,  Password = PasswordBox.Password,  Role = "Водитель"  };    db.Users.Add(userObject);  db.SaveChanges();  }  }  } |

Страница водителя:

|  |
| --- |
| namespace OsaGo.Pages  {  /// <summary>  /// Логика взаимодействия для Driver.xaml  /// </summary>  public partial class Driver : Page  {  public Driver()  {  InitializeComponent();  DataGridCar.ItemsSource = Entitie1.GetContext().Cars.Where(x => x.DriverId == Page1.UserUI.DriverId).ToList();  DataGridPolice.ItemsSource = Entitie1.GetContext().Policies.Where(x => x.DriverId == Page1.UserUI.DriverId).ToList();  }  }  } |