## **2.7 Проектирование интерфейса приложения**

Благодаря ранее разработанной диаграмме вариантов использования, мы уже знаем, какие действия должен выполнять конечный пользователь. Таким образом, мы можем определить требуемый набор окон и их компонентов для решения задачи.

Проектирование интерфейса приложения — это процесс создания эффективного, удобного и привлекательного пользовательского интерфейса для программного приложения. Оно включает в себя разработку визуального оформления, расположение элементов управления, навигацию и взаимодействие пользователя с приложением.

Ниже будут представлены макеты приложения:

* разработка макета страницы регистрации представлена на рисунке 2.16.

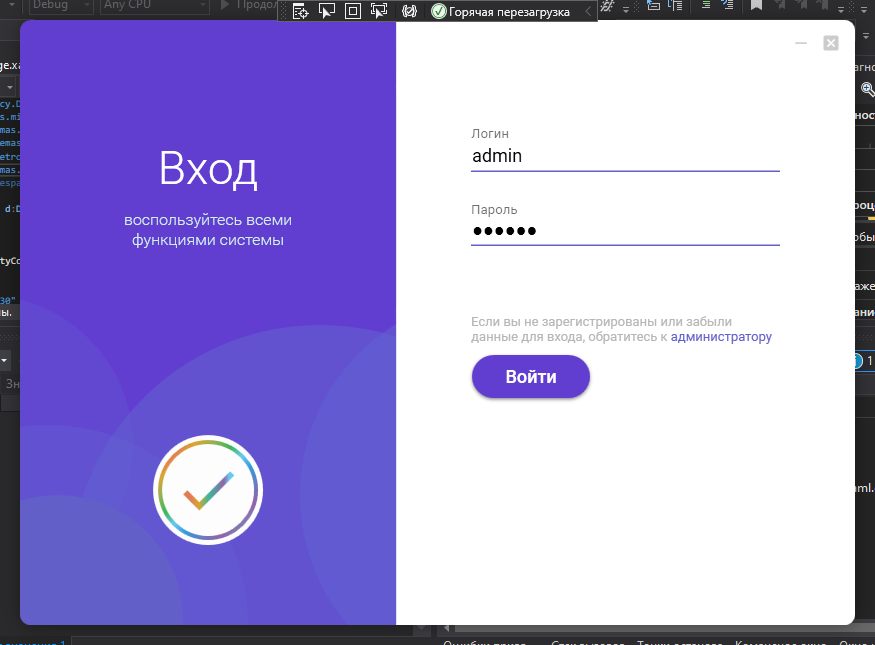


Рисунок 2.16 – Макет окна авторизации

* разработка макета главного окна представлена на рисунке 2.17.

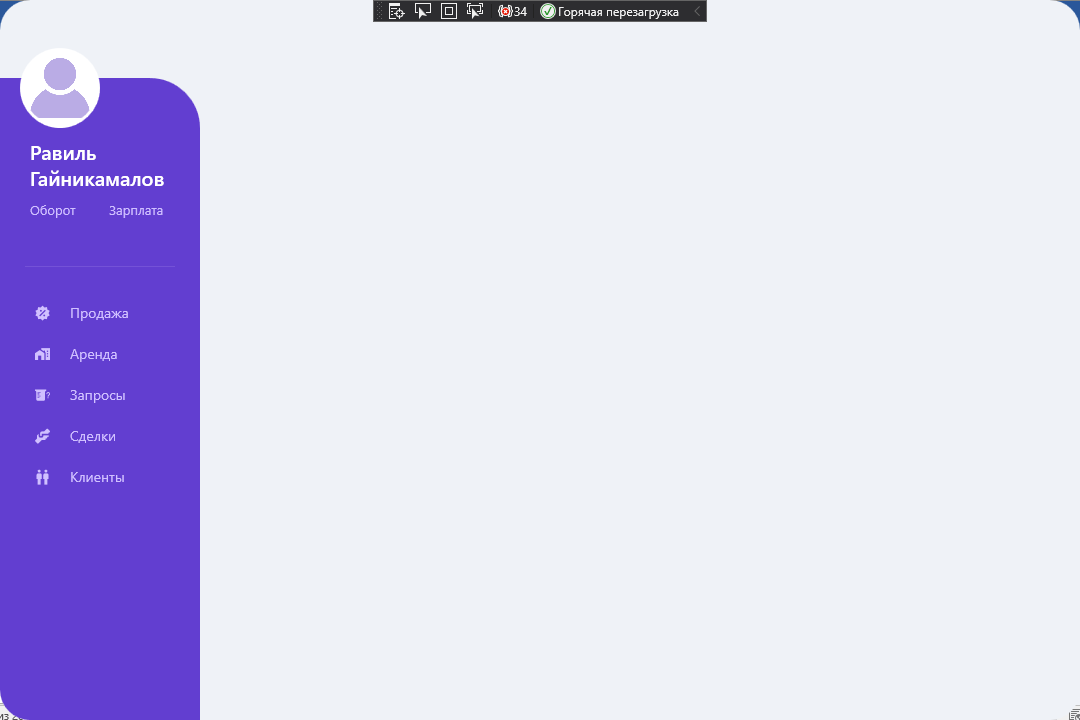


Рисунок 2.17 – Макет главного окна

* разработка макета страницы объектов представлена на рисунке 2.18.

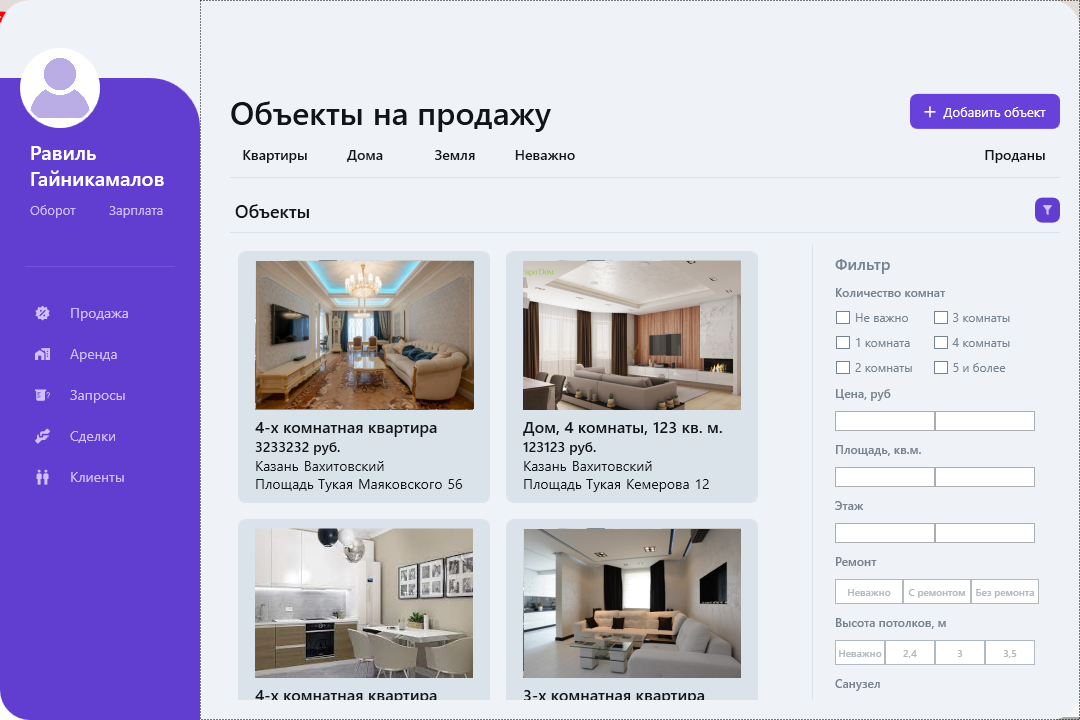


Рисунок 2.18 – Макет страницы объектов

* разработка макета окна добавления клиента представлена на рисунке 2.19.

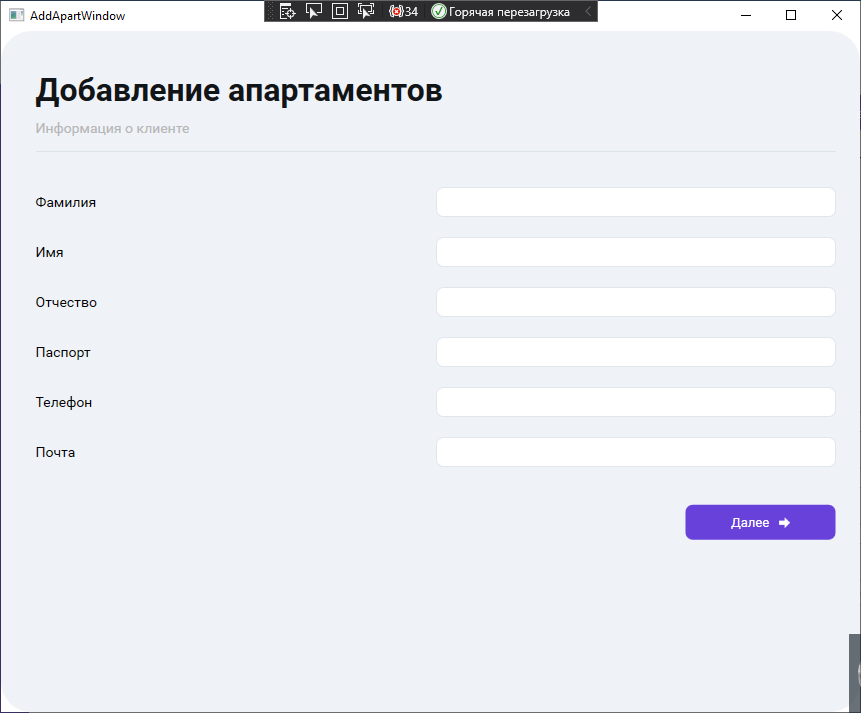


Рисунок 2.19 – Макет окна добавления клиента

* разработка окна добавления апартаментов представлена на рисунке 2.20

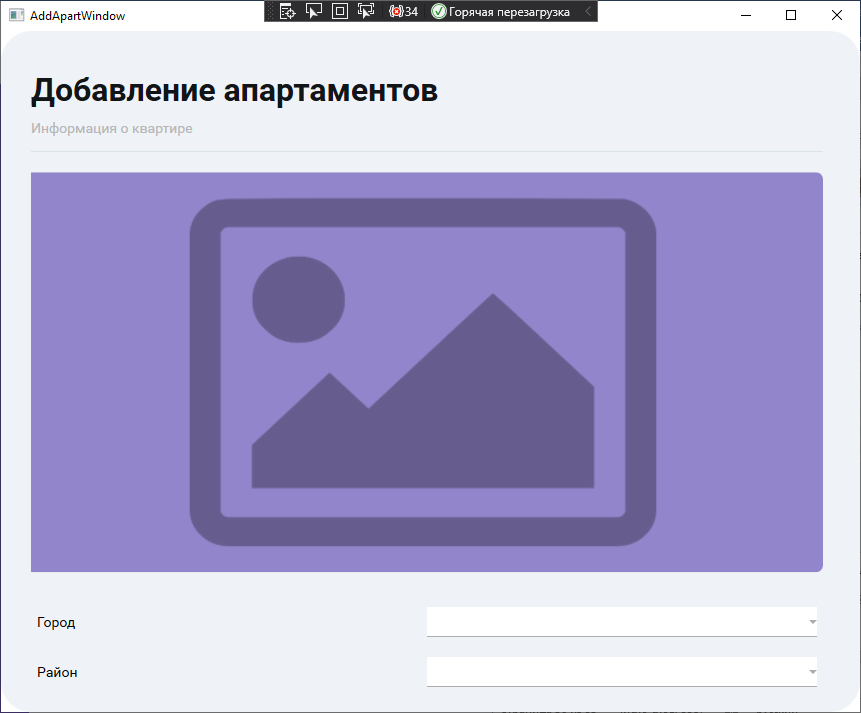


Рисунок 2.20 – Макет окна добавления апартаментов

* разработка окна добавления запросов представлена на рисунке 2.21

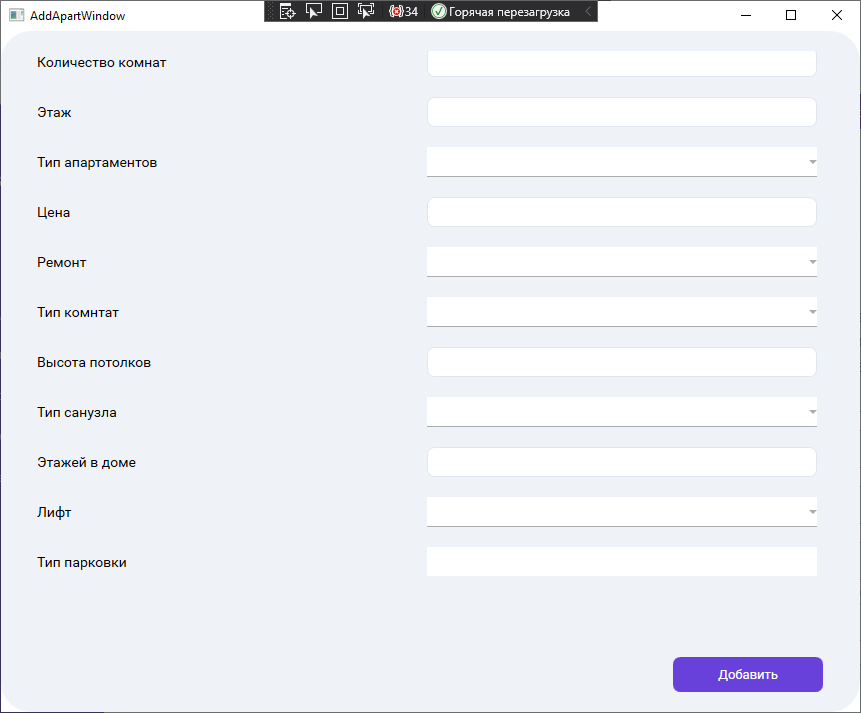


Рисунок 2.21 – Макет окна добавления запросов

* разработка окна клиентов представлена на рисунке 2.22

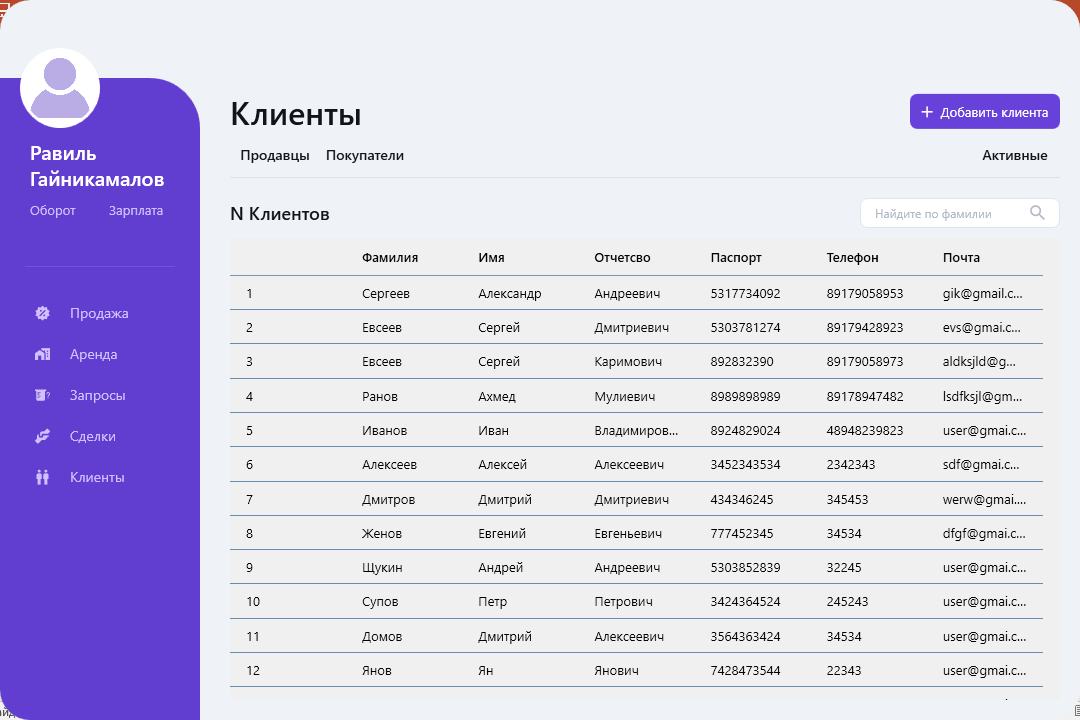


Рисунок 2.22 – Макет клиентов

## **ГЛАВА 3 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ**

MS SQL Server (Microsoft SQL Server) — это система управления реляционными базами данных (СУБД), разработанная компанией Microsoft. Она предназначена для хранения, организации и управления данными в корпоративных средах и широко используется в различных приложениях и системах.

Конечный выбор СУБД зависит от конкретных требований и контекста проекта. MS SQL Server имеет ряд преимуществ, особенно в экосистеме Microsoft, и широко применяется в организациях для управления данными и обеспечения надежности и производительности систем.

Начнём с таблицы «Apartament», которая представлена на рисунке 3.1. Содержание таблицы указано на рисунке 3.2.

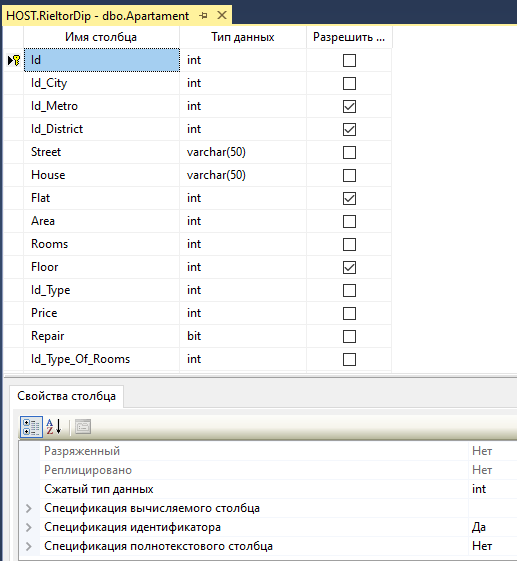


Рисунок 3.1 – Дизайнер таблицы «Apartament»

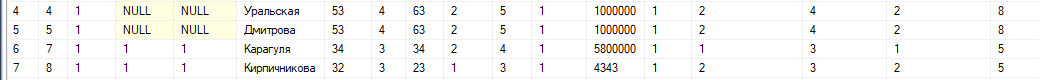


Рисунок 3.2 – Содержание таблицы «Apartament»

Таблица «Worker» представлена на рисунке 3.3. Содержание таблицы представлено на рисунке 3.4.

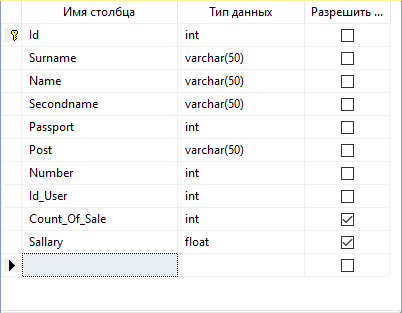


Рисунок 3.3 – Конструктор таблицы «Worker»



Рисунок 3.4 – Содержание таблицы «Worker»

Таблица «City» представлена на рисунке 3.5. Содержание таблицы представлено на рисунке 3.6.

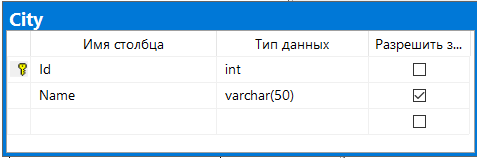


Рисунок 3.5 – Конструктор таблицы «City»

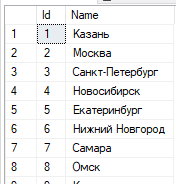


Рисунок 3.6 – Содержание таблицы «City»

Таблица «Client» представлена на рисунке 3.7. Содержание таблицы представлено на рисунке 3.8.

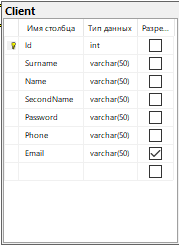


Рисунок 3.7 – Конструктор таблицы «Client»

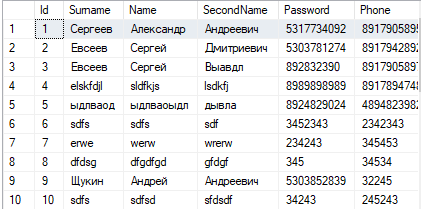


Рисунок 3.8 – Содержание таблицы «Client»

Таблица «Deal» представлена на рисунке 3.9. Содержание таблицы представлено на рисунке 3.10.



Рисунок 3.9 – Конструктор таблицы «Deal»

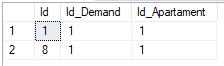


Рисунок 3.10 – Содержание таблицы «Deal»

Таблица «Demands» представлена на рисунке 3.11. Содержание таблицы представлено на рисунке 3.12.

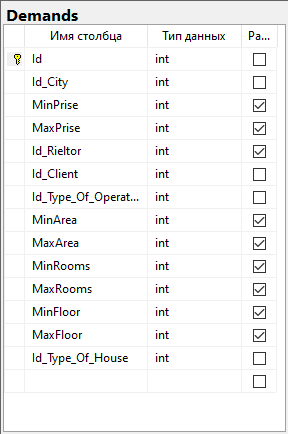


Рисунок 3.11 – Конструктор таблицы «Demands»



Рисунок 3.12 – Содержание таблицы «Demands»

Таблица «Districts» представлена на рисунке 3.13. Содержание таблицы представлено на рисунке 3.14.

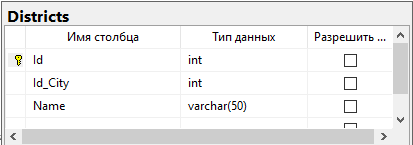


Рисунок 3.13 – Конструктор таблицы «Districts»



Рисунок 3.14 – Содержание таблицы «Districts»

Таблица «Metro» представлена на рисунке 3.15. Содержание таблицы представлено на рисунке 3.16

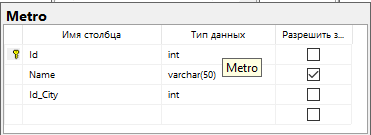


Рисунок 3.15 – Конструктор таблицы «Metro»



Рисунок 3.16 – Содержание таблицы «Metro»

Таблица «Type\_Of\_House» представлена на рисунке 3.17. Содержание таблицы представлено на рисунке 3.18



Рисунок 3.17 – Конструктор таблицы «Type\_Of\_House»

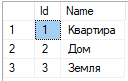


Рисунок 3.18 – Содержание таблицы «Type\_Of\_House»

Таблица «Type\_Of\_Operation» представлена на рисунке 3.19. Содержание таблицы представлено на рисунке 3.20

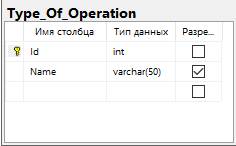


Рисунок 3.19 – Конструктор таблицы «Type\_Of\_Operation»



Рисунок 3.20 – Конструктор таблицы «Type\_Of\_Operation»

Таблица «Type\_Of\_Parking» представлена на рисунке 3.21. Содержание таблицы представлено на рисунке 3.22

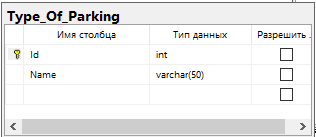


Рисунок 3.21 – Конструктор таблицы «Type\_Of\_Parking»

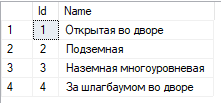


Рисунок 3.22 – Содержание таблицы «Type\_Of\_Parking»

Таблица «Type\_Of\_Rooms» представлена на рисунке 3.23. Содержание таблицы представлено на рисунке 3.24

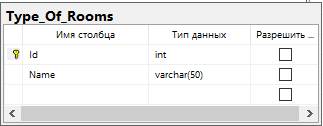


Рисунок 3.23 – Конструктор таблицы «Type\_Of\_Rooms»

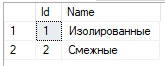


Рисунок 3.24 – Содержание таблицы «Type\_Of\_Rooms»

Таблица «Type\_Of\_Toilet» представлена на рисунке 3.25. Содержание таблицы представлено на рисунке 3.26

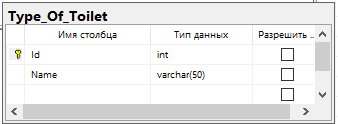


Рисунок 3.25 – Конструктор таблицы «Type\_Of\_Toilet»

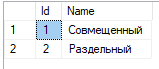


Рисунок 3.26 – Содержание таблицы «Type\_Of\_Toilet»

Собрав все таблицы и расставив связи, у нас, в итоге, получилась схема базы данных, показанная на рисунке 3.27.

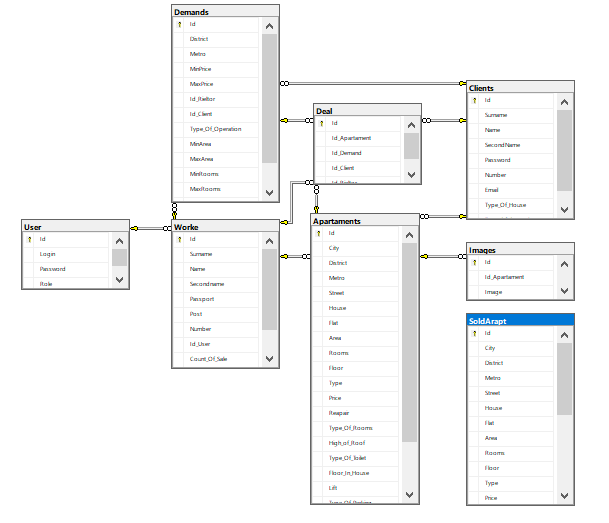


Рисунок 3.27 – Схема базы данных

# ГЛАВА 4 ПОЛЬЗОВАТЕЛЬСКАЯ ДОКУМЕНТАЦИЯ

Пользовательская документация представляет собой руководство пользователя и руководство по установке ПО, описывающее каждую функцию программы, а также шаги, которые нужно выполнить для использования этой функции.

## **4.1 Руководство программиста**

Руководство программиста — это документ или набор инструкций, которые предоставляются программистам для помощи и руководства во время разработки программного обеспечения. Руководство программиста может включать в себя рекомендации, стандарты кодирования, примеры использования, советы по отладке и другую полезную информацию, которая помогает программистам эффективно и правильно выполнять свою работу.

Общая информация о важных компонентах проекта:

1. Ключевым элементом в системе графического интерфейса в WPF является окно, которое содержит все необходимые элементы управления. Окно в WPF представлено классом Window, который является производным от класса ContentControl. Поэтому окно является элементом управления содержимым, и как, к примеру, кнопка, может содержать в себе один дочерний элемент. Как правило, в его качестве выступает один из элементов компоновки, например, Grid.

2. Конфигурационные файлы: определяют настройки приложения, а именно подключение к базе данных и настройка авторизации (JWT-авторизация).

3. Ресурсы: файлы, содержащие статические ресурсы, такие как изображения (фотографии профилей соискателя.

## **4.2 Руководство по установке программного обеспечения**

Руководство по установке программного обеспечения (installation guide) — это документ или набор инструкций, предназначенных для помощи пользователям в процессе установки и настройки определенного программного продукта на своем компьютере или сервере.

## Для того, что установить программу «RieltorAgency» необходимо запустить исполняемый файл RieltorAgency.exe.

## Запустив его, мы принимаем условия соглашение и нажимаем «Продолжить».

## Далее открывается окно, которое представлено на рисунке 4.1. Оно позволяет пользователю выбирать папку, куда сохранится наша программа.

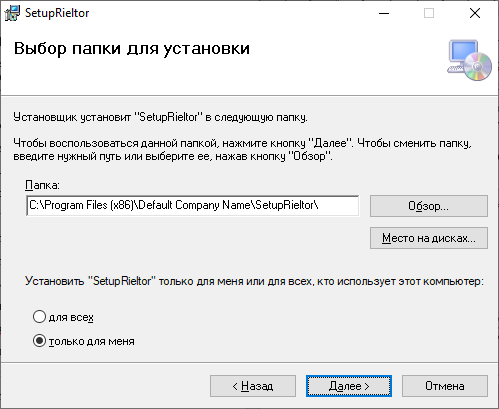


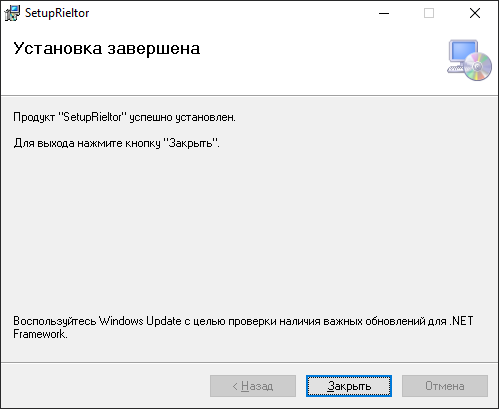
Рисунок 4.1 – Выбор папки для сохранения

## На рисунке 4.2 изображено окно, на котором нам нужно нажать на кнопку «Далее».

## 

Рисунок 4.2 – Подтверждение установки

## И последнее, что нам нужно сделать – подождать, пока программа установится на Ваш компьютер. После этого на рисунке 4.3 будет изображено окно с результатом установки.



## Рисунок 4.3 – Окно приветствия

## **4.3 Руководство пользователя**

Настоящий документ является руководством пользователя по эксплуатации автоматизированной информационной системы.

На рисунке 4.4 представлено окно входа. Введя валидные данные, пользователь может дальше работать с приложением.

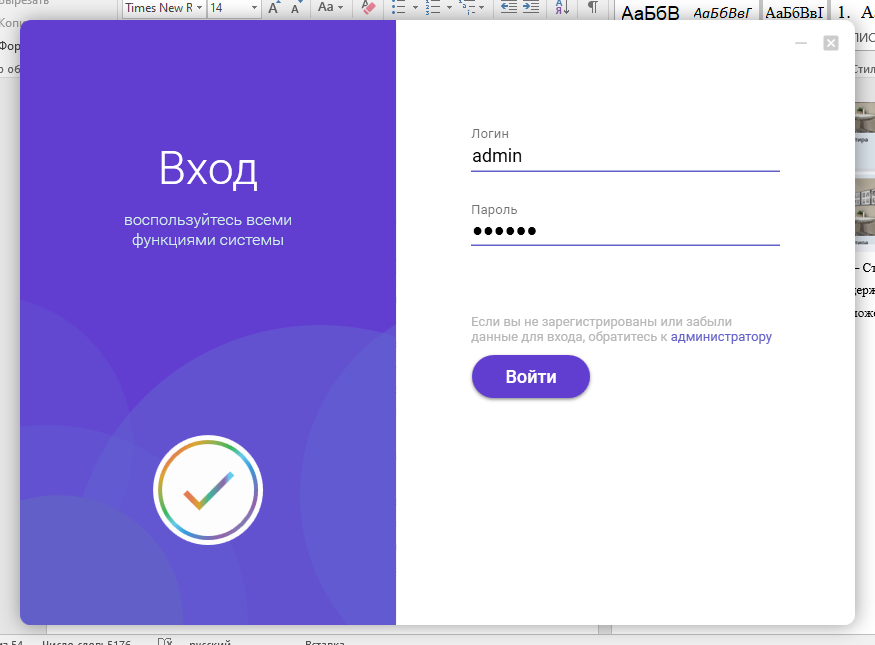


Рисунок 4.4 – Окно авторизации

На рисунке 4.5 представлена главная страница приложения, с которой пользователь сталкивается в первую очередь.

В основной части находится список объектов, состоящий из карточек объектов. В каждой карточке содержится количество комнат, цена объекта, метро, адрес, площадь объекта, фото объекта, а также кнопка для открытия содержимого карточки, с возможностью ее редактирования и удаления. Справа находится фильтр объектов, а также кнопка добавления объекта и перехода к списку проданных объектов.

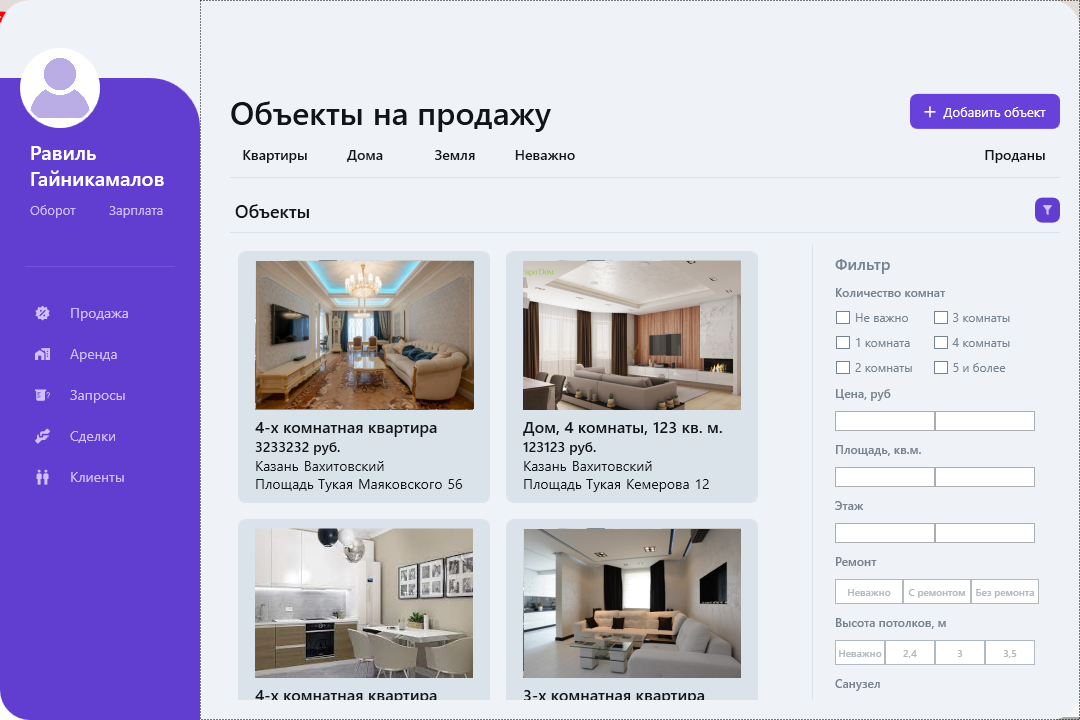


Рисунок 4.5. – Страница со списком объектов

Нажав на кнопку содержимого карточки выйдет следующее окно (рисунок 4.6). Пользователь может обновить информацию, удалить карточку.

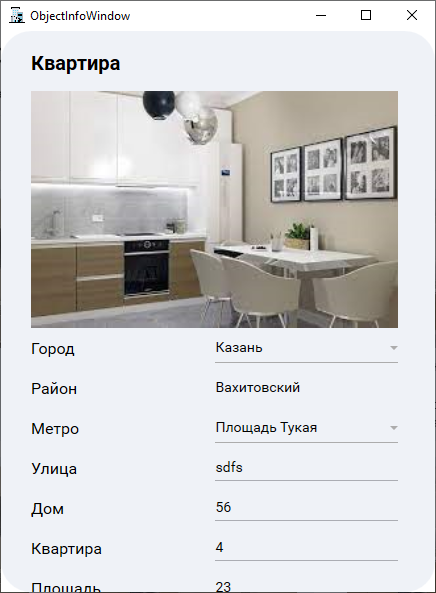


Рисунок 4.6. – Карточка определенного объекта

Далее сотрудник может продать объект и просмотреть таблицу с проданными объектами (рисунок 4.7).

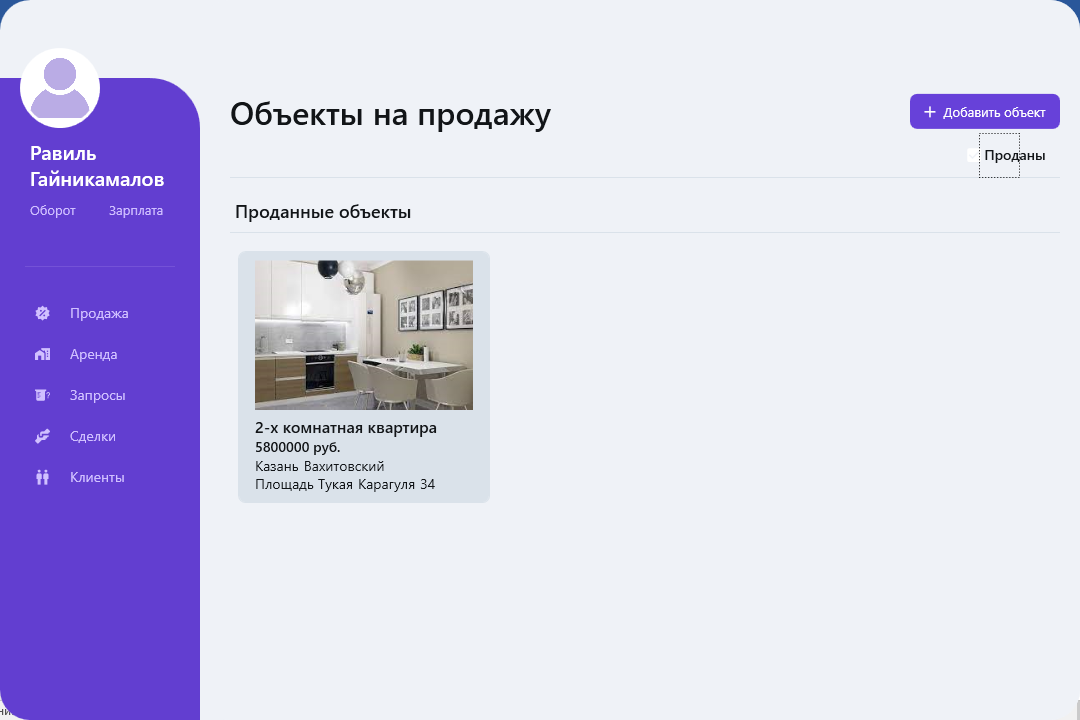


Рисунок 4.7 – Окно со списком проданных объектов

Пользователь может добавить объект. Пример продемонстрирован на рисунке 4.8 и на рисунке 4.9.

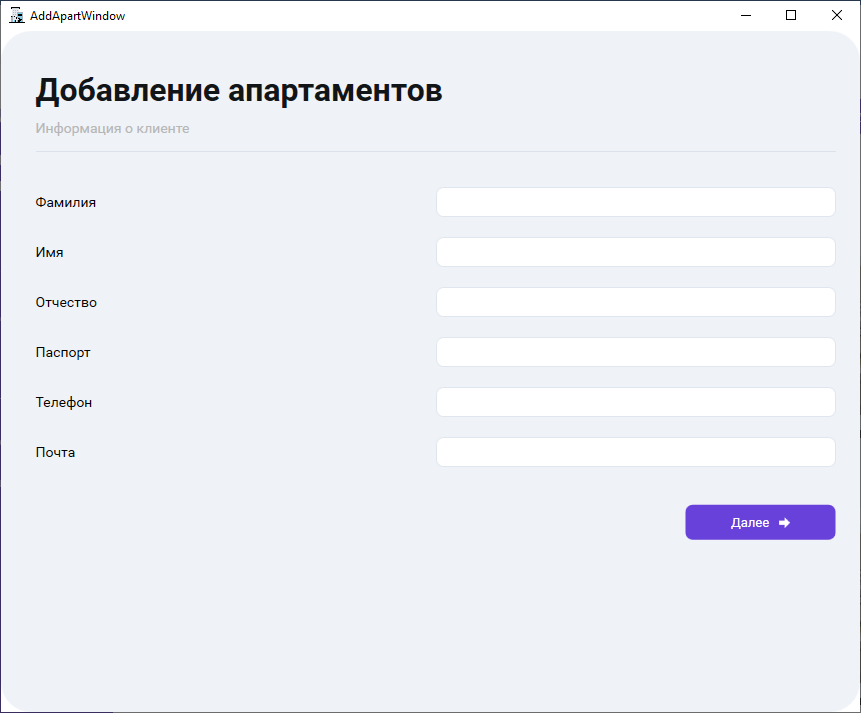


Рисунок 4.8 – Добавление объекта

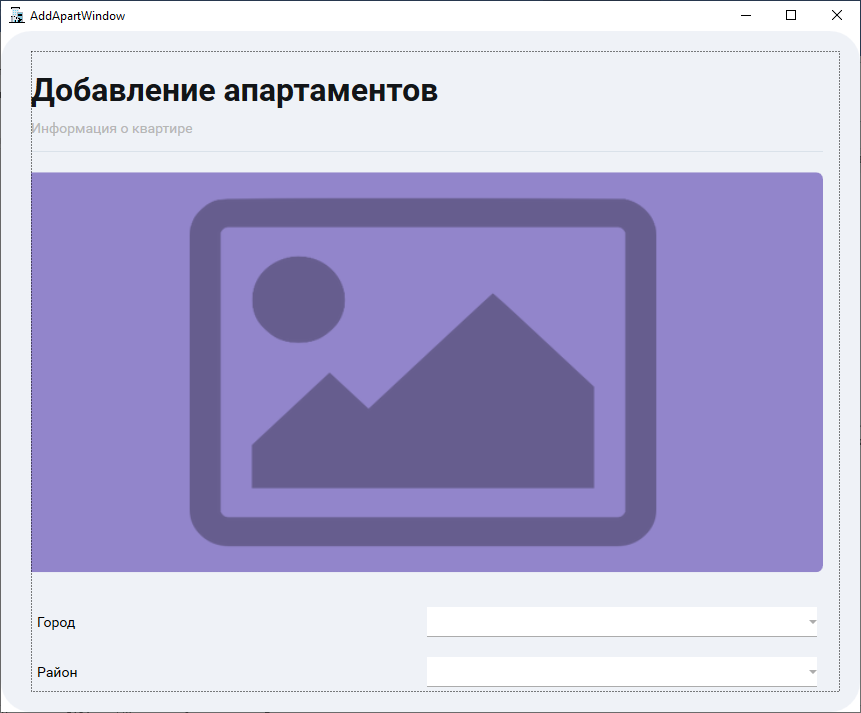


Рисунок 4.9 – Добавление объекта

Также сотрудник может просмотреть список клиентов. Пример показан на рисунке 4.10

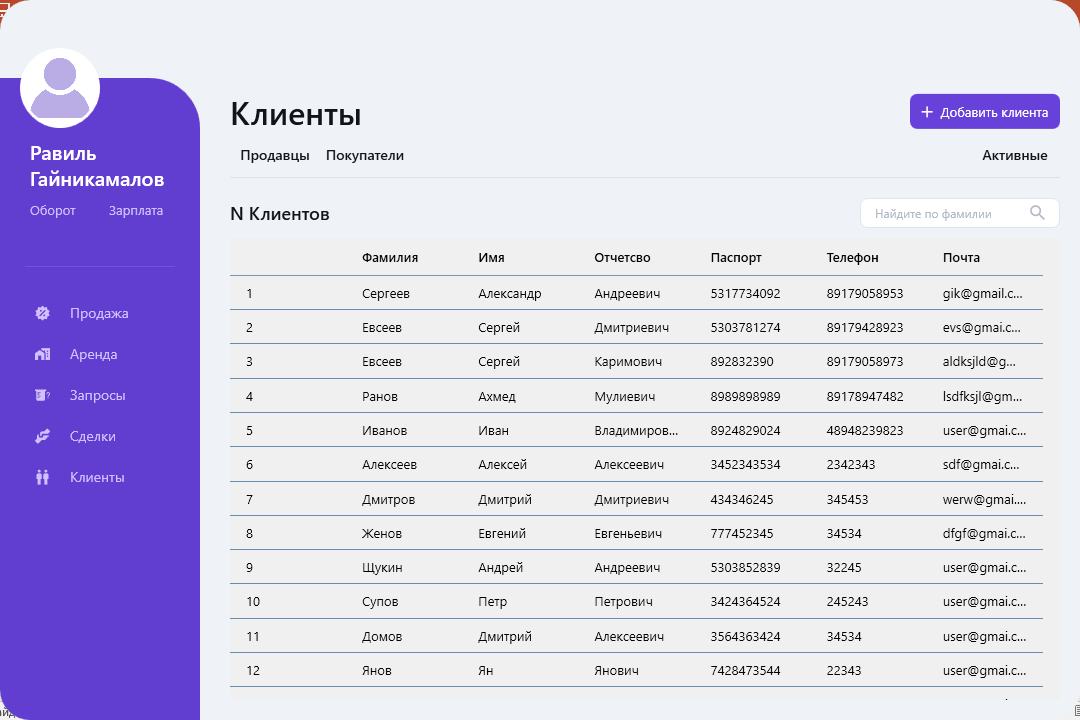


Рисунок 4.10 – Список клиентов

# ПРИЛОЖЕНИЕ A

**Список сокращений**

БД – база данных

СУБД – система управления базами данных

VS – Visual Studio – среда разработки

C# – C Sharp – язык программирования

SQL – Structured Query Language – язык структурированных запросов

.NET – Native Executable Translation – среда в процессе выполнения приложения сначала транслирует его в машинный код

UML – Unified Modeling Language – унифицированный язык моделирования

HTML – HyperText Markup Language – язык гипертекстовой разметки

WPF – cистема для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем

XAML –  расширяемый язык разметки для приложений

# ПРИЛОЖЕНИЕ Б

**Листинг программы**

Окно **«**MainWindow**»**:

namespace RieltorAgency

{

public partial class MainWindow : Window

{

public static RieltorDipEntities db = new RieltorDipEntities();

public MainWindow()

{

InitializeComponent();

}

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

{

var user = db.Worker.FirstOrDefault(q => q.Login == LoginTxt.Text && q.Password == PasswordTxt.Password.ToString());

if (user != null)

{

RieltorWindow worker = new RieltorWindow(user);

worker.Show();

this.Hide();

}

else

{

MessageBox.Show("Неудача");

}

}

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

{

this.Close();

}

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

{

Application.Current.MainWindow.WindowState = WindowState.Minimized;

}

private void Border\_MouseDown(object sender, MouseButtonEventArgs e)

{

if (e.ChangedButton == MouseButton.Left)

{

this.DragMove();

}

}

}

}

Окно «AddApartWindow»

namespace RieltorAgency

{

public partial class RieltorWindow : Window

{

private Worker \_worker;

public RieltorWindow(Worker worker)

{

InitializeComponent();

\_worker = worker;

SaleFrame.Navigate(new SalePage(worker));

RentFrame.Navigate(new RentPage());

ClientFrame.Navigate(new ClientsPage());

DemandFrame.Navigate(new DemandPage());

DealFrame.Navigate(new DealPage());

}

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

{

SaleFrame.Visibility = Visibility.Visible;

RentFrame.Visibility = Visibility.Hidden;

ClientFrame.Visibility = Visibility.Hidden;

DemandFrame.Visibility = Visibility.Hidden;

DealFrame.Visibility = Visibility.Hidden;

}

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

{

RentFrame.Visibility = Visibility.Visible;

SaleFrame.Visibility = Visibility.Hidden;

ClientFrame.Visibility = Visibility.Hidden;

DemandFrame.Visibility = Visibility.Hidden;

DealFrame.Visibility = Visibility.Hidden;

}

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

{

ClientFrame.Visibility = Visibility.Hidden;

RentFrame.Visibility = Visibility.Hidden;

SaleFrame.Visibility = Visibility.Hidden;

DemandFrame.Visibility = Visibility.Visible;

DealFrame.Visibility = Visibility.Hidden;

}

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

{

DemandFrame.Visibility = Visibility.Hidden;

ClientFrame.Visibility = Visibility.Visible;

RentFrame.Visibility = Visibility.Hidden;

SaleFrame.Visibility = Visibility.Hidden;

DealFrame.Visibility = Visibility.Hidden;

}

private void Border\_MouseDown(object sender, MouseButtonEventArgs e)

{

if (e.ChangedButton == MouseButton.Left)

{

this.DragMove();

}

}

private bool IsMaximized = false;

private void Border\_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)

{

if (e.ClickCount == 2)

{

if (IsMaximized)

{

this.WindowState = WindowState.Normal;

this.Height = 720;

this.Width = 1080;

IsMaximized = false;

}

else

{

this.WindowState = WindowState.Maximized;

IsMaximized = true;

}

{ }

}

}

private void Border\_Loaded(object sender, RoutedEventArgs e)

{

TxtName.Text = \_worker.Name;

TxtSurName.Text = \_worker.Surname;

TxtCountOfSale.Text = Convert.ToString(\_worker.Count\_Of\_Sale);

TxtSallary.Text = \_worker.Sallary.ToString();

}

}

}

Страница «SalePage»

namespace RieltorAgency

{

public partial class SalePage : Page

{

public RieltorDipEntities \_db = new RieltorDipEntities();

private List<Apartament> \_apartament = new List<Apartament>();

private List<Apartament> \_apartFlat = new List<Apartament>();

private List<Apartament> \_apartHouse = new List<Apartament>();

private List<Apartament> \_apartEarth = new List<Apartament>();

private Worker \_worker;

public SalePage(Worker worker)

{

InitializeComponent();

\_worker = worker;

this.\_apartament = \_db.Apartament.OrderByDescending(t => t.Id).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

this.\_apartFlat = \_db.Apartament.OrderByDescending(t => t.Id).Where(t => t.Id\_Type == 1).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

this.\_apartHouse = \_db.Apartament.OrderByDescending(t => t.Id).Where(t => t.Id\_Type == 2).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

this.\_apartEarth = \_db.Apartament.OrderByDescending(t => t.Id).Where(t => t.Id\_Type == 3).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

ListAparts.ItemsSource = \_apartament;

}

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

{

MunuClnm.Width = new GridLength(250);

filterBtnClose.Visibility = Visibility.Visible;

filterBtn.Visibility = Visibility.Hidden;

}

public void RefreshApart()

{

}

private void ListAparts\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

}

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

{

MunuClnm.Width = new GridLength(0);

filterBtn.Visibility = Visibility.Visible;

filterBtnClose.Visibility = Visibility.Hidden;

}

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

{

AddApartWindow addApartWindow = new AddApartWindow(\_worker);

addApartWindow.Show();

}

bool flagFlat, flagHouse, flagEarth;

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

{

flagFlat = true;

flagHouse = false;

flagEarth = false;

\_apartament = \_apartFlat;

ListAparts.ItemsSource = \_apartament;

runNameOfObject.Text = "Квартиры";

MunuClnm.Width = new GridLength(0);

filterBtn.Visibility = Visibility.Visible;

filterBtnClose.Visibility = Visibility.Hidden;

}

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

{

flagFlat = false;

flagHouse = true;

flagEarth = false;;

\_apartament = \_apartHouse;

ListAparts.ItemsSource = \_apartament;

runNameOfObject.Text = "Дома";

MunuClnm.Width = new GridLength(0);

filterBtn.Visibility = Visibility.Visible;

filterBtnClose.Visibility = Visibility.Hidden;

}

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

{

flagFlat = false;

flagHouse = false;

flagEarth = true;

\_apartEarth = \_apartFlat.OrderByDescending(t => t.Id).Where(t => t.Id\_Type == 3).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

\_apartament = \_apartEarth;

ListAparts.ItemsSource = \_apartEarth;

runNameOfObject.Text = "Земля";

MunuClnm.Width = new GridLength(0);

filterBtn.Visibility = Visibility.Visible;

filterBtnClose.Visibility = Visibility.Hidden;

}

int countSold = 1;

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

{

}

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

{

flagFlat = false;

flagHouse = false;

flagEarth = false;

\_apartament = \_db.Apartament.OrderByDescending(t => t.Id).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

ListAparts.ItemsSource = \_apartament;

runNameOfObject.Text = "Объекты";

MunuClnm.Width = new GridLength(0);

filterBtn.Visibility = Visibility.Visible;

filterBtnClose.Visibility = Visibility.Hidden;

}

private void ChBSold\_Checked(object sender, RoutedEventArgs e)

{

if (ChBSold.IsChecked == true)

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Sold == true).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

ListAparts.ItemsSource = \_apartament;

runNameOfObject.Text = "Проданные объекты";

MunuClnm.Width = new GridLength(0);

filterBtn.Visibility = Visibility.Hidden;

filterBtnClose.Visibility = Visibility.Hidden;

BtnFilterFlat.Visibility = Visibility.Hidden;

BtnFilterHouse.Visibility = Visibility.Hidden;

BtnFilterEarth.Visibility = Visibility.Hidden;

BntFilterAll.Visibility = Visibility.Hidden;

}

}

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

{

ObjectInfoWindow info = new ObjectInfoWindow(\_db, sender, this);

info.ShowDialog();

}

private void ChBSold\_Unchecked(object sender, RoutedEventArgs e)

{

if (ChBSold.IsChecked == false)

{

\_apartament = \_db.Apartament.OrderByDescending(t => t.Id).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

ListAparts.ItemsSource = \_apartament;

runNameOfObject.Text = "Oбъекты";

MunuClnm.Width = new GridLength(0);

filterBtn.Visibility = Visibility.Visible;

filterBtnClose.Visibility = Visibility.Hidden;

BtnFilterFlat.Visibility = Visibility.Visible;

BtnFilterHouse.Visibility = Visibility.Visible;

BtnFilterEarth.Visibility = Visibility.Visible;

BntFilterAll.Visibility = Visibility.Visible;

}

}

//Кнопка показать результаты фильтра

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

{

if (flagFlat == true && flagHouse == false && flagEarth == false)

\_apartament = \_apartFlat.OrderByDescending(t => t.Id).Where(t => t.Id\_Type == 1).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

else if (flagFlat == false && flagHouse == true && flagEarth == false)

\_apartament = \_apartHouse.OrderByDescending(t => t.Id).Where(t => t.Id\_Type == 2).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

else if (flagFlat == false && flagHouse == false && flagEarth == true)

\_apartament = \_apartEarth.OrderByDescending(t => t.Id).Where(t => t.Id\_Type == 3).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

else if (flagFlat == false && flagHouse == false && flagEarth == false)

\_apartament = \_db.Apartament.OrderByDescending(t => t.Id).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

if (ChB1.IsChecked == true)

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Rooms == 1).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

else if (ChB2.IsChecked == true)

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Rooms == 2).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

else if (ChB3.IsChecked == true)

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Rooms == 3).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

else if (ChB4.IsChecked == true)

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Rooms == 4).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

else if (ChB5.IsChecked == true)

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Rooms == 5).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

if (txtBoxMaxPrice.Text != "")

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Price <= Convert.ToInt32(txtBoxMaxPrice.Text)).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

if (txtBoxMinPrice.Text != "")

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Price >= Convert.ToInt32(txtBoxMinPrice.Text)).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

if (txtMinArea.Text != "")

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Area >= Convert.ToInt32(txtMinArea.Text)).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

if (txtMaxArea.Text != "")

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Area <= Convert.ToInt32(txtMaxArea.Text)).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

if (txtMinFloor.Text != "")

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Floor >= Convert.ToInt32(txtMinFloor.Text)).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

if (txtMaxFloor.Text != "")

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Floor <= Convert.ToInt32(txtMaxFloor.Text)).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

if (tgbfirst.IsChecked == true)

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.High\_Of\_Roof < 3 && t.High\_Of\_Roof >= 2.4).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

else if (tgbsec.IsChecked == true)

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.High\_Of\_Roof < 3.5 && t.High\_Of\_Roof >= 3).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

else if (tgbthird.IsChecked == true)

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.High\_Of\_Roof >= 3.5).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

if (tgbRepair.IsChecked == true)

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Repair == true).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

else if (tgbNonRepair.IsChecked == true)

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Repair == false).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

if (tgbTogether.IsChecked == true)

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Id\_Type\_Of\_Toilet == 1).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

else if (tgbApart.IsChecked == true)

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Id\_Type\_Of\_Toilet == 2).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

if (txtMinCountFloor.Text != "")

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Floor\_In\_House >= Convert.ToInt32(txtMinCountFloor.Text)).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

if (txtMaxCountFloor.Text != "")

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Floor\_In\_House <= Convert.ToInt32(txtMaxCountFloor.Text)).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

if (tgbLiftY.IsChecked == true)

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Lift == true).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

else if (tgbLiftN.IsChecked == true)

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Lift == false).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

if (ChBpark1.IsChecked == true)

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Id\_Type\_Of\_Parking == 1).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

else if (ChB2park.IsChecked == true)

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Id\_Type\_Of\_Parking == 2).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

else if (ChB3park.IsChecked == true)

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Id\_Type\_Of\_Parking == 3).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

else if (ChB4park.IsChecked == true)

{

\_apartament = \_apartament.OrderByDescending(t => t.Id).Where(t => t.Id\_Type\_Of\_Parking == 4).Where(p => p.Id\_Type\_Of\_Operation == 1).ToList();

}

ListAparts.ItemsSource = \_apartament;

}

}

}

Страница «ClientPage»

namespace RieltorAgency

{

public partial class ClientsPage : Page

{

public RieltorDipEntities \_db = new RieltorDipEntities();

public ClientsPage()

{

InitializeComponent();

DataGridClients.ItemsSource = \_db.Client.ToList();

}

}

}

Страница «RentPage»

namespace RieltorAgency

{

public partial class RentPage : Page

{

public RieltorDipEntities \_db = new RieltorDipEntities();

public RentPage()

{

InitializeComponent();

ListAparts.ItemsSource = \_db.Apartament.ToList();

}

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

{

MunuClnm.Width = new GridLength(250);

filterBtnClose.Visibility = Visibility.Visible;

filterBtn.Visibility = Visibility.Hidden;

}

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

{

MunuClnm.Width = new GridLength(0);

filterBtn.Visibility = Visibility.Visible;

filterBtnClose.Visibility = Visibility.Hidden;

}

private void ListAparts\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

}

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

{

}

}

Окно «AddApartWindow»

namespace RieltorAgency

{

public partial class AddApartWindow : Window

{

public RieltorDipEntities \_db = new RieltorDipEntities();

Client client;

private Worker \_worker;

public AddApartWindow(Worker worker)

{

InitializeComponent();

\_worker = worker;

}

public string url;

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

{

Apartament apartament = new Apartament()

{

City = CmbCity.SelectedItem as City,

Districts = CmbDistrict.SelectedItem as Districts,

Metro = CmbMetro.SelectedItem as Metro,

Street = TxtStreet.Text,

House = TxtHouse.Text,

Flat = Convert.ToInt32(TxtFlat.Text),

Area = Convert.ToInt32(TxtArea.Text),

Rooms = Convert.ToInt32(TxtCountOfRooms.Text),

Floor = Convert.ToInt32(TxtFloor.Text),

Type\_Of\_House = CmbTypeOFApart.SelectedItem as Type\_Of\_House,

Price = Convert.ToInt32(TxtPrice.Text),

Repair = CmbRepair.SelectedIndex == 0,

Type\_Of\_Rooms = CmbTypeOfRooms.SelectedItem as Type\_Of\_Rooms,

High\_Of\_Roof = float.Parse(TxtHighOfRoof.Text),

Type\_Of\_Toilet = CmbTypeOfToilet.SelectedItem as Type\_Of\_Toilet,

Floor\_In\_House = Convert.ToInt32(TxtFloorInHouse.Text),

Lift = CmbLiftYorN.SelectedIndex == 0,

Type\_Of\_Parking = CmbTypeOfParking.SelectedItem as Type\_Of\_Parking,

Id\_Client = client.Id,

Id\_Rieltor = \_worker.Id,

Type\_Of\_Operation = \_db.Type\_Of\_Operation.First(t => t.Name == "Продажа"),

Sold = false

};

\_db.Apartament.Add(apartament);

\_db.SaveChanges();

int count = 0;

foreach (var i in \_db.Apartament)

{

count++;

}

Image image = new Image()

{

Image1 = image\_byte,

Id\_Apartament = count

};

\_db.Image.Add(image);

\_db.SaveChanges();

if (apartament != null)

MessageBox.Show("Апартаменты добавлены");

else

MessageBox.Show("Ti loh");

}

private void CmbCity\_TextChanged(object sender, TextChangedEventArgs e)

{

}

private void ListImages\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

}

byte[] image\_byte;

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

{

InfoClient.Visibility = Visibility.Hidden;

InfoFlat.Visibility = Visibility.Visible;

client = new Client()

{

Surname = TxtSurname.Text,

Name = TxtName.Text,

SecondName = TxtSecondName.Text,

Password = Passport.Text,

Phone = TxtPhone.Text,

Email = TxtEmail.Text

};

\_db.Client.Add(client);

\_db.SaveChanges();

}

private void CmbCity\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

City city = CmbCity.SelectedItem as City;

CmbDistrict.ItemsSource = \_db.Districts.Where(d => d.City.Id == city.Id).ToList();

CmbMetro.ItemsSource = \_db.Metro.Where(c => c.City.Id == city.Id).ToList();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

CmbCity.ItemsSource = \_db.City.ToList();

CmbTypeOFApart.ItemsSource = \_db.Type\_Of\_House.ToList();

CmbTypeOfParking.ItemsSource = \_db.Type\_Of\_Parking.ToList();

CmbTypeOfRooms.ItemsSource = \_db.Type\_Of\_Rooms.ToList();

CmbTypeOfToilet.ItemsSource = \_db.Type\_Of\_Toilet.ToList();

}

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

{

OpenFileDialog openFileDialog = new OpenFileDialog();

openFileDialog.ShowDialog(); // показываем

if (openFileDialog.ShowDialog() == true)

{

Image image = new Image();

Uri fileUri = new Uri(openFileDialog.FileName);

url = fileUri.ToString();

Img.Source = new BitmapImage(fileUri);

image\_byte = File.ReadAllBytes(openFileDialog.FileName);

Img.Visibility = Visibility.Visible;

AddImage.Visibility = Visibility.Hidden;

}

}

}

Окно «ObjectInfoWindow»

namespace RieltorAgency

{

public partial class ObjectInfoWindow : Window

{

public RieltorDipEntities \_db = new RieltorDipEntities();

private Apartament \_apartament;

public City city;

private SalePage \_salePage;

public ObjectInfoWindow(RieltorDipEntities db, object o, SalePage salePage)

{

InitializeComponent();

\_apartament = (o as Button).DataContext as Apartament;

\_db = db;

\_salePage = salePage;

TxtType\_Of\_House.Text = \_apartament.Type\_Of\_House.Name;

CmbCity.SelectedIndex = \_apartament.Id\_City - 1;

CmbDistrict.SelectedIndex = Convert.ToInt32(\_apartament.Id\_District) -1;

CmbMetro.SelectedIndex = Convert.ToInt32(\_apartament.Id\_Metro)-1;

CmbTypeOfRooms.SelectedIndex = Convert.ToInt32(\_apartament.Id\_Type\_Of\_Rooms) - 1;

CmbTypeOfToilet.SelectedIndex = Convert.ToInt32(\_apartament.Id\_Type\_Of\_Toilet) - 1;

CmbTypeOfParking.SelectedIndex = Convert.ToInt32(\_apartament.Id\_Type\_Of\_Parking) - 1;

TxtStreet.Text = \_apartament.Street;

TxtHouse.Text = \_apartament.House;

TxtFlat.Text = \_apartament.Flat.ToString();

TxtArea.Text = \_apartament.Area.ToString();

TxtRooms.Text = \_apartament.Rooms.ToString();

TxtFloor.Text = \_apartament.Floor.ToString();

TxtPrice.Text = \_apartament.Price.ToString();

if (\_apartament.Repair == true)

{

CmbRepair.SelectedItem = RepairY;

}

else if(\_apartament.Repair == false)

{

CmbRepair.SelectedItem = RepairN;

}

TxtHighOfRoof.Text = \_apartament.High\_Of\_Roof.ToString();

TxtFloorInHouse.Text = \_apartament.Floor\_In\_House.ToString();

if (\_apartament.Lift)

CmbLiftYorN.SelectedItem = LiftY;

else if (\_apartament.Lift == false)

CmbLiftYorN.SelectedItem = LiftN;

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

CmbCity.ItemsSource = \_db.City.ToList();

CmbTypeOfRooms.ItemsSource = \_db.Type\_Of\_Rooms.ToList();

CmbTypeOfParking.ItemsSource = \_db.Type\_Of\_Parking.ToList();

CmbTypeOfToilet.ItemsSource = \_db.Type\_Of\_Toilet.ToList();

}

private void CmbCity\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

City city = CmbCity.SelectedItem as City;

CmbDistrict.ItemsSource = \_db.Districts.Where(d => d.City.Id == city.Id).ToList();

CmbMetro.ItemsSource = \_db.Metro.Where(c => c.City.Id == city.Id).ToList();

}

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

{

\_apartament.Street = TxtStreet.Text;

\_apartament.House = TxtHouse.Text;

\_apartament.Flat = Convert.ToInt32(TxtFlat.Text);

\_apartament.Area = Convert.ToInt32(TxtArea.Text);

\_apartament.Rooms = Convert.ToInt32(TxtRooms.Text);

\_apartament.Floor = Convert.ToInt32(TxtFloor.Text);

\_apartament.Price = Convert.ToInt32(TxtPrice.Text);

\_apartament.High\_Of\_Roof = float.Parse(TxtHighOfRoof.Text);

\_apartament.Floor\_In\_House = Convert.ToInt32(TxtFloorInHouse.Text);

\_db.SaveChanges();

this.Close();

}

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

{

\_db.Apartament.Remove(\_apartament);

\_db.SaveChanges();

this.Close();

}

private void CmbTypeOfRooms\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

}

private void CmbTypeOfToilet\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

}

private void CmbTypeOfParking\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

}

}

}

# ПРИЛОЖЕНИЕ В

**Презентация**

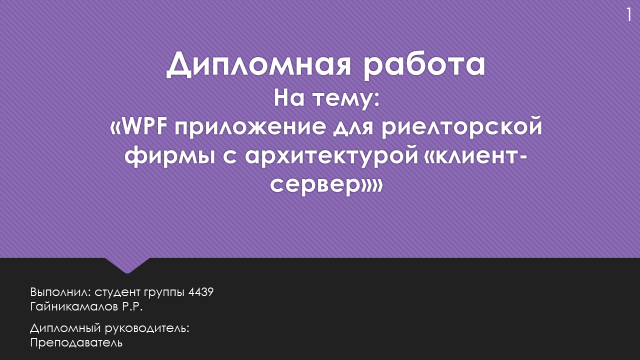
****

Рисунок В.1 – Слайд №1

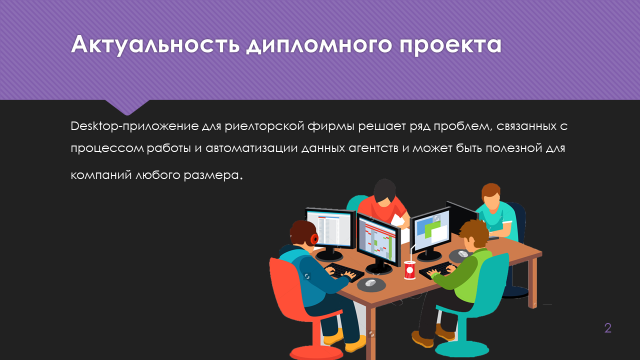
****

Рисунок В.2 – Слайд №2

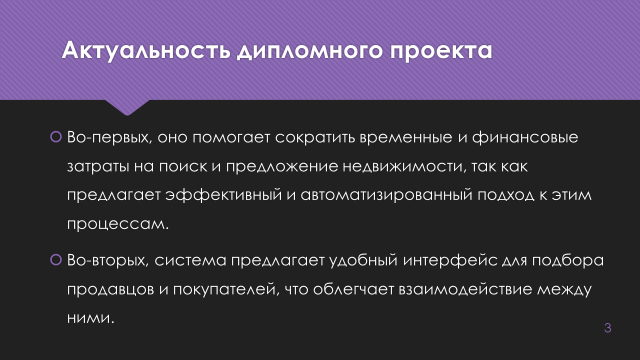
****

Рисунок В.3 – Слайд №3

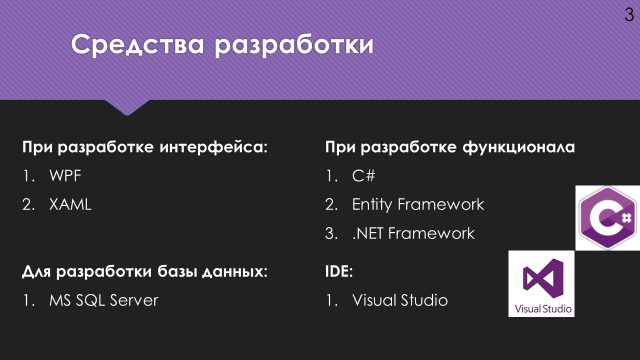


Рисунок В.4 – Слайд №4



Рисунок В.5 – Слайд №5

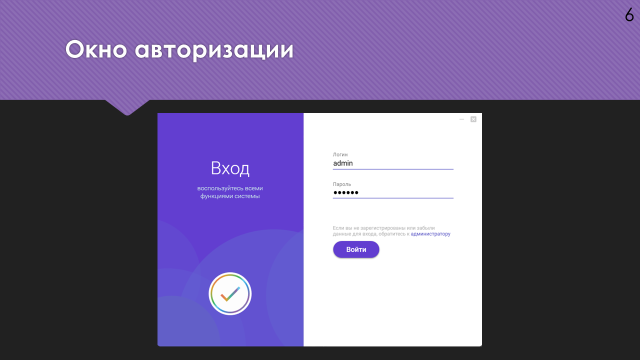


Рисунок В.6 – Слайд №6

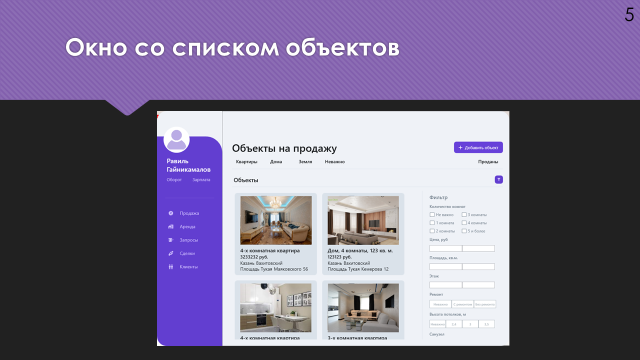


Рисунок В.7 – Слайд №7

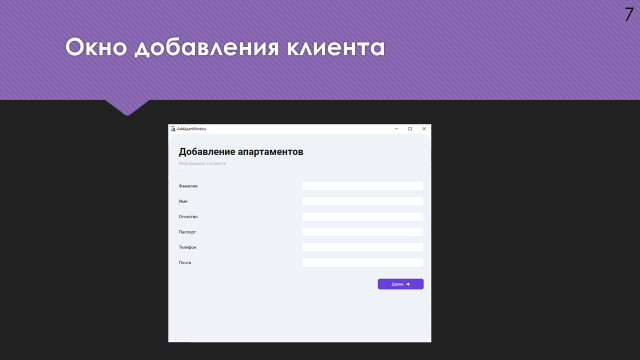


Рисунок В.8 – Слайд №8

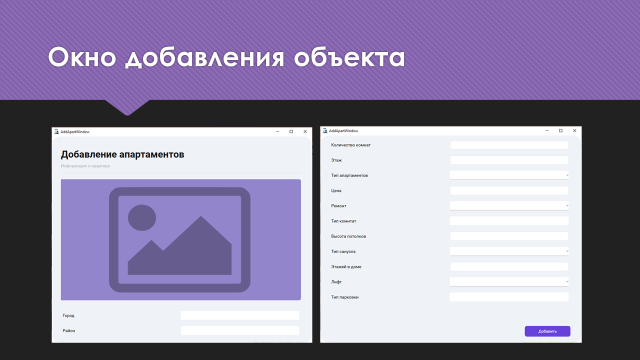


Рисунок В.9 – Слайд №9

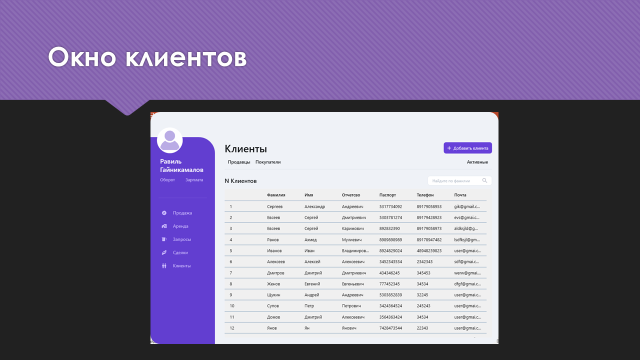


Рисунок В.10 – Слайд №10

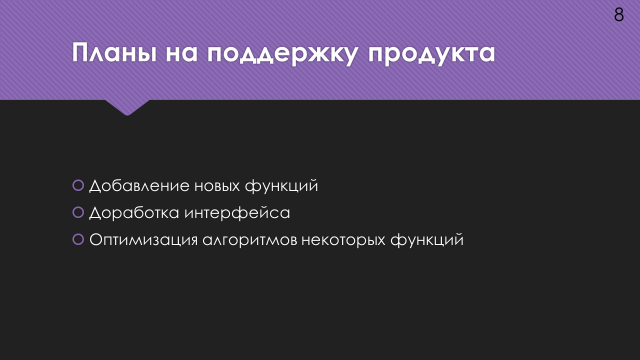


Рисунок В.11 – Слайд №11

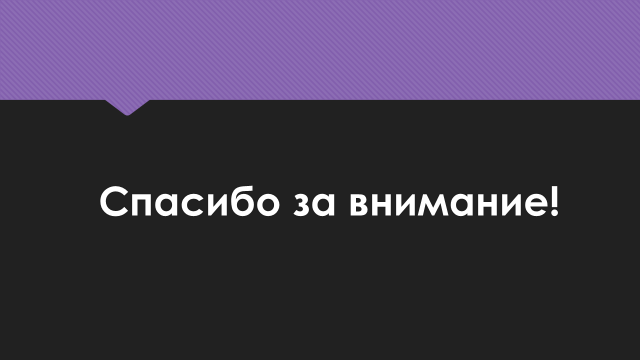


Рисунок В.12 – Слайд №12