Федеральное государственное образовательное бюджетное учреждение высшего образования
«Финансовый университет при Правительстве Российской Федерации»
(Финансовый университет)

Колледж информатики и программирования

ОТЧЁТ
По учебной практике

Специальность 09.02.07 «Информационные системы и программирование»
(код) (наименование)

Профессиональный модуль ПМ.01 Разработка модулей программного обеспечения для компьютерных систем
(код) (наименование)

Междисциплинарный курс МДК.01.03 Разработка мобильных приложений
(код) (наименование)

Выполнил:

Студент 4 курса 4ИСИП-222 учебной группы
(номер) (номер)
_____ В.Д. Очтова
(подпись) (инициалы, фамилия)

Проверил:

Руководитель практики от Колледжа информатики и программирования
Преподаватель Л.Д. Мальков
(квалификационная категория или звание, должность) (инициалы, фамилия)

Москва – 2026

Перечень работ, выполненных в ходе учебной практики

№ п/п	Виды работ	Оценка
1	Планирование проекта для разрабатываемого мобильного приложения	5 (отлично)
2	Разработка спецификаций мобильного приложения	5 (отлично)
3	Проектирование мобильного приложения	5 (отлично)
4	Разработка мобильного приложения	5 (отлично)
5	Отладка и тестирование мобильного приложения	5 (отлично)
6	Разработка технической документации мобильного приложения	5 (отлично)

Модуль № 1. Разработка, администрирование и защита баз данных

GitHub - [VioNo/HousingStockOchtova]

Создадим систему для управляющей компании, которая объединит данные по домам, платежам, задолженностям и заявкам. Приложение позволит администраторам вести учёт, а жильцам – удобно подавать обращения. Первое, что необходимо сделать – спроектировать базу данных. На рисунке 1 представлена ER-диаграмма базы данных

Рисунок 1 – ER-диаграмма базы данных

На листинге 1 представлен скрипт базы данных.

Листинг 1 – Скрипт создания базы данных

```
USE [Housing_stock]
GO
```

```
/****** Object:  Table [dbo].[Apartment]    Script Date: 22.01.2026 11:24:42
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Apartment](
	[ID] [int] IDENTITY(1,1) NOT NULL,
	[Adress] [nvarchar](255) NULL,
	[Beginning] [datetime] NULL,
	[Floors] [int] NULL,
	[Flats] [int] NULL,
	[Year] [int] NULL,
	[Area] [float] NULL,
 CONSTRAINT [PK_Apartment] PRIMARY KEY CLUSTERED
(
	[ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY =
OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Applications]    Script Date: 22.01.2026
11:24:42 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Applications](
	[ID] [int] IDENTITY(1,1) NOT NULL,
	[Address] [nvarchar](255) NOT NULL,
	[ApplicantName] [nvarchar](255) NOT NULL,
	[Phone] [nvarchar](50) NOT NULL,
	[Description] [nvarchar](1000) NOT NULL,
	[Responsible] [nvarchar](255) NOT NULL,
	[Status] [nvarchar](50) NOT NULL,
	[CreateDate] [datetime] NOT NULL,
	[CompleteDate] [datetime] NULL,
	[Priority] [nvarchar](50) NULL,
	[AssignedTo] [int] NULL,
	[AssignedEmployee] [nvarchar](255) NULL,
	[CategoryID] [int] NULL,
 CONSTRAINT [PK_Applications] PRIMARY KEY CLUSTERED
(
	[ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY =
OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Debt]    Script Date: 22.01.2026 11:24:42
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Debt](
	[ID] [int] IDENTITY(1,1) NOT NULL,
	[ID_owner] [int] NULL,
	[Water] [float] NULL,
	[Electric power] [float] NULL,
 CONSTRAINT [PK_Debt] PRIMARY KEY CLUSTERED
(
```

```
	[ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY =
OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Employees]    Script Date: 22.01.2026 11:24:42
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Employees](
	[EmployeeID] [int] IDENTITY(1,1) NOT NULL,
	[FullName] [nvarchar](255) NOT NULL,
	[Position] [nvarchar](100) NOT NULL,
	[Phone] [nvarchar](50) NULL,
	[Email] [nvarchar](255) NULL,
	[HireDate] [date] NULL,
	[Status] [nvarchar](50) NULL,
	[UserID] [int] NULL,
 CONSTRAINT [PK_Employees] PRIMARY KEY CLUSTERED
(
	[EmployeeID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY =
OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Owners]    Script Date: 22.01.2026 11:24:42
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Owners](
	[ID] [int] IDENTITY(1,1) NOT NULL,
	[Name_owner] [nvarchar](255) NULL,
	[Flat] [int] NULL,
	[Phone number] [float] NULL,
	[Adress] [int] NULL,
 CONSTRAINT [PK_Owners] PRIMARY KEY CLUSTERED
(
	[ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY =
OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Partners]    Script Date: 22.01.2026 11:24:42
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Partners](
	[PartnerID] [int] IDENTITY(1,1) NOT NULL,
	[PartnerName] [nvarchar](255) NOT NULL,
	[ContactPerson] [nvarchar](255) NULL,
	[Phone] [nvarchar](50) NULL,
	[Email] [nvarchar](255) NULL,
	[Services] [nvarchar](500) NULL,
	[ContractDate] [date] NULL,
```

```
	[ContractNumber] [nvarchar](50) NULL,
	[Status] [nvarchar](50) NULL,
 CONSTRAINT [PK_Partners] PRIMARY KEY CLUSTERED
(
	[PartnerID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY =
OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Payment]    Script Date: 22.01.2026 11:24:42
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Payment](
	[ID] [int] IDENTITY(1,1) NOT NULL,
	[ID_owner] [int] NULL,
	[Period] [nvarchar](255) NULL,
	[Accrued] [float] NULL,
	[Paid for] [float] NULL,
 CONSTRAINT [PK_Payment] PRIMARY KEY CLUSTERED
(
	[ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY =
OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Roles]    Script Date: 22.01.2026 11:24:42
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Roles](
	[RoleID] [int] IDENTITY(1,1) NOT NULL,
	[RoleName] [nvarchar](50) NULL,
 CONSTRAINT [PK_Roles] PRIMARY KEY CLUSTERED
(
	[RoleID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY =
OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[ServiceCategories]    Script Date: 22.01.2026
11:24:42 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[ServiceCategories](
	[ID] [int] IDENTITY(1,1) NOT NULL,
	[CategoryName] [nvarchar](255) NOT NULL,
	[Description] [nvarchar](1000) NULL,
	[IsActive] [bit] NOT NULL,
PRIMARY KEY CLUSTERED
(
	[ID] ASC
```

```
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY =
OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Users]    Script Date: 22.01.2026 11:24:42
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Users](
      [UserID] [int] IDENTITY(1,1) NOT NULL,
      [RoleID] [int] NULL,
      [FullName] [nvarchar](255) NULL,
      [Login] [nvarchar](50) NULL,
      [Password] [nvarchar](255) NULL,
      [Email] [nvarchar](255) NULL,
 CONSTRAINT [PK_Users] PRIMARY KEY CLUSTERED
(
      [UserID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY =
OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Applications] ADD  DEFAULT (getdate()) FOR [CreateDate]
GO
ALTER TABLE [dbo].[Applications] ADD  DEFAULT ('Средний') FOR [Priority]
GO
ALTER TABLE [dbo].[Employees] ADD  DEFAULT ('Активен') FOR [Status]
GO
ALTER TABLE [dbo].[Partners] ADD  DEFAULT ('Активен') FOR [Status]
GO
ALTER TABLE [dbo].[ServiceCategories] ADD  DEFAULT ((1)) FOR [IsActive]
GO
ALTER TABLE [dbo].[Applications]  WITH CHECK ADD  CONSTRAINT
[FK_Applications_ServiceCategories] FOREIGN KEY([CategoryID])
REFERENCES [dbo].[ServiceCategories] ([ID])
GO
ALTER TABLE [dbo].[Applications] CHECK CONSTRAINT
[FK_Applications_ServiceCategories]
GO
ALTER TABLE [dbo].[Debt]  WITH CHECK ADD  CONSTRAINT [FK_Debt_Owners] FOREIGN
KEY([ID_owner])
REFERENCES [dbo].[Owners] ([ID])
GO
ALTER TABLE [dbo].[Debt] CHECK CONSTRAINT [FK_Debt_Owners]
GO
ALTER TABLE [dbo].[Employees]  WITH CHECK ADD  CONSTRAINT
[FK_Employees_Users] FOREIGN KEY([UserID])
REFERENCES [dbo].[Users] ([UserID])
GO
ALTER TABLE [dbo].[Employees] CHECK CONSTRAINT [FK_Employees_Users]
GO
ALTER TABLE [dbo].[Owners]  WITH CHECK ADD  CONSTRAINT [FK_Owners_Apartment]
FOREIGN KEY([Adress])
REFERENCES [dbo].[Apartment] ([ID])
GO
ALTER TABLE [dbo].[Owners] CHECK CONSTRAINT [FK_Owners_Apartment]
GO
ALTER TABLE [dbo].[Payment]  WITH CHECK ADD  CONSTRAINT [FK_Payment_Owners]
FOREIGN KEY([ID_owner])
```

```
REFERENCES [dbo].[Owners] ([ID])
GO
ALTER TABLE [dbo].[Payment] CHECK CONSTRAINT [FK_Payment_Owners]
GO
ALTER TABLE [dbo].[Users]  WITH CHECK ADD  CONSTRAINT [FK_Users_Roles]
FOREIGN KEY([RoleID])
REFERENCES [dbo].[Roles] ([RoleID])
GO
ALTER TABLE [dbo].[Users] CHECK CONSTRAINT [FK_Users_Roles]
GO
```

Далее в листинге 2 представлен скрипт JOIN запроса, который показывает задолженности с информацией о собственниках и домах. Результат запроса представлен на рисунке 2.

Листинг 2 – JOIN запрос

```
-- Получение детального отчета о задолженностях
SELECT
    d.ID AS DebtID,
    o.Name_owner AS OwnerFIO,
    o.Flat,
    a.Adress AS FullAddress,
    o.[Phone number] AS ContactPhone,
    d.Water,
    d.[Electric power] AS ElectricPower,
    (ISNULL(d.Water, 0) + ISNULL(d.[Electric power], 0)) AS TotalDebt
FROM
    dbo.Debt AS d
INNER JOIN
    dbo.Owners AS o ON o.ID = d.ID_owner
LEFT JOIN
    dbo.Apartment AS a ON a.ID = o.Adress
WHERE
    (d.Water > 0 OR d.[Electric power] > 0) -- Только при наличии долга
ORDER BY
    TotalDebt DESC, o.Name_owner;
```

Results Messages

	DebtID	OwnerFIO	Flat	FullAddress	ContactPhone	Water	ElectricPower	TotalDebt
1	3	Байчорова Агата Рустамовна	8	ул. Матросова, 179а, Светлоград	89643300000	178451	257891,1	436342,1
2	2	Савельев Олег Иванович	4	ул. Матросова, 179а, Светлоград	89287800000	14567,56	48517,25	63084,81
3	7	Петров Станислав Игоревич	16	ул. Матросова, 179а, Светлоград	89189200000	7837,45	18991,79	26829,24
4	1	Мазалова Ирина Львовна	2	ул. Матросова, 179а, Светлоград	89185600000	2455,2	7541,81	9997,01
5	6	Лукин Илья Федорович	15	ул. Матросова, 179а, Светлоград	89634600000	438,87	1250,94	1689,81
6	9	Любяшева Галина Аркадьевна	23	ул. Матросова, 179а, Светлоград	89625700000	102,89	387,58	490,47
7	4	Тюренкова Наталья Сергеевна	9	ул. Матросова, 179а, Светлоград	89630000000	56,12	142,35	198,47
8	8	Мартыненко Александр Сергеевич	20	ул. Матросова, 179а, Светлоград	89183200000	1,27	0,84	2,11
9	5	Гусев Семен Петрович	13	ул. Матросова, 179а, Светлоград	89188600000	0,14	NULL	0,14

Рисунок 2 – Результат JOIN запрос

Ниже представлены таблицы с описанием полей и типов данных для каждой таблицы в базе данных.

Таблица 1 – Apartment (Многоквартирные дома)

Поле	Тип данных	NULL	Дополнительно

ID	int	NO	IDENTITY(1,1), PK
Adress	nvarchar(255)	YES	
Beginning	datetime	YES	
Floors	int	YES	
Flats	int	YES	
Year	int	YES	
Area	float	YES	

Таблица 2 – Applications (Заявки)

Поле	Тип данных	NULL	Дополнительно
ID	int	NO	IDENTITY(1,1), PK
Address	nvarchar(255)	NO	
ApplicantName	nvarchar(255)	NO	
Phone	nvarchar(50)	NO	
Description	nvarchar(1000)	NO	
Responsible	nvarchar(255)	NO	
Status	nvarchar(50)	NO	
CreateDate	datetime	NO	DEFAULT getdate()
CompleteDate	datetime	YES	
Priority	nvarchar(50)	YES	DEFAULT N'Средний'
AssignedTo	int	YES	
AssignedEmployee	nvarchar(255)	YES	
CategoryID	int	YES	FK → dbo.ServiceCategories(ID)

Таблица 3 – Debt (Задолженности)

Поле	Тип данных	NULL	Дополнительно
ID	int	NO	IDENTITY(1,1), PK
ID_owner	int	YES	FK → dbo.Owners(ID)
Water	float	YES	
Electric power	float	YES	Имя поля с пробелом

Таблица 4 – Employees (Сотрудники)

Поле	Тип данных	NULL	Дополнительно
EmployeeID	int	NO	IDENTITY(1,1), PK
FullName	nvarchar(255)	NO	
Position	nvarchar(100)	NO	
Phone	nvarchar(50)	YES	
Email	nvarchar(255)	YES	
HireDate	date	YES	
Status	nvarchar(50)	YES	DEFAULT N'Активен'
UserID	int	YES	FK → dbo.Users(UserID)

Таблица 5 – Owners (Собственники)

Поле	Тип данных	NULL	Примечание
ID	int	NO	IDENTITY(1,1), PK
Name_owner	nvarchar(255)	YES	
Flat	int	YES	
Phone number	float	YES	Имя поля с пробелом
Adress	int	YES	FK → dbo.Apartment(ID)

Таблица 6 – Partners (Партнеры)

Поле	Тип данных	NULL	Дополнительно
PartnerName	nvarchar(255)	NO	
ContactPerson	nvarchar(255)	YES	
Phone	nvarchar(50)	YES	
Email	nvarchar(255)	YES	
Services	nvarchar(500)	YES	
ContractDate	date	YES	
ContractNumber	nvarchar(50)	YES	
Status	nvarchar(50)	YES	DEFAULT N'Активен'

Таблица 7 – Payment (Платежи)

Поле	Тип данных	NULL	Дополнительно
ID	int	NO	IDENTITY(1,1), PK

ID_owner	int	YES	FK → dbo.Owners(ID)
Period	nvarchar(255)	YES	
Accrued	float	YES	
Paid for	float	YES	Имя поля с пробелом

Модуль № 2. Разработка модулей программного обеспечения для компьютерных систем

В ходе работы была создана блок-схема «Алгоритм разработки приложения» согласно стандарту ГОСТ 19.701-90. Она представлены ниже на рисунке 3.

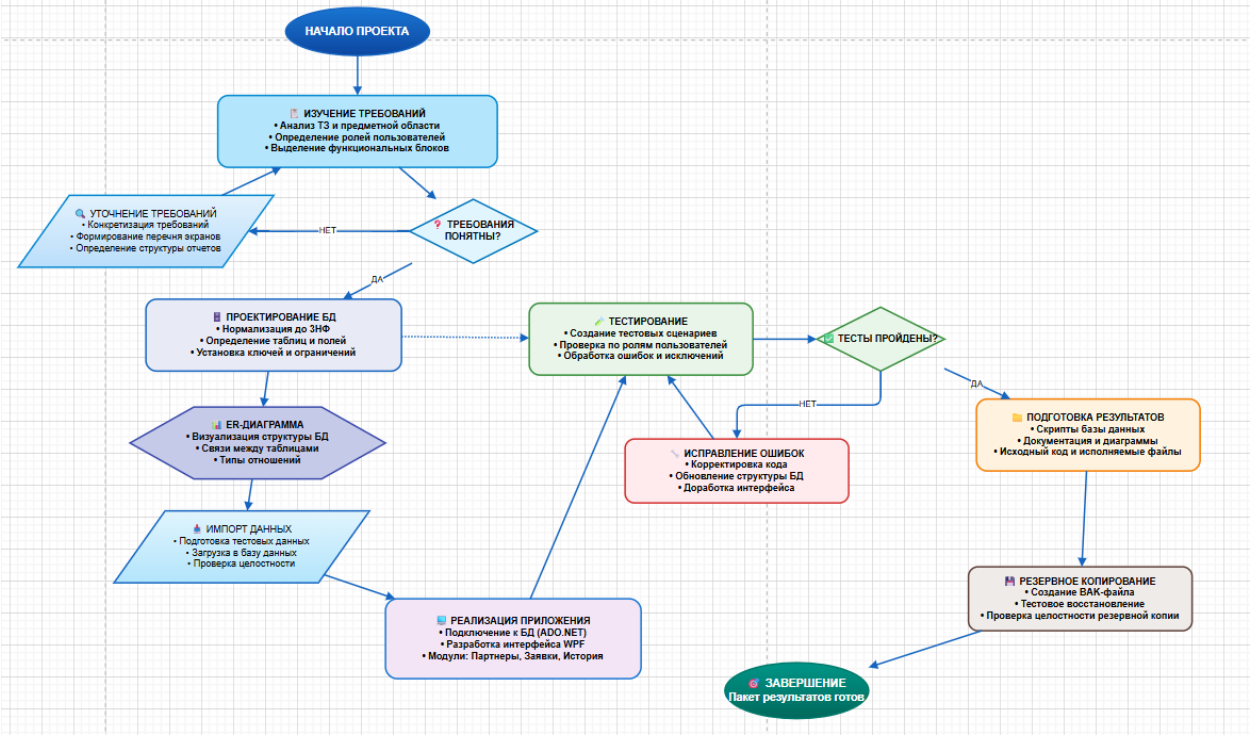

Рисунок 3 – Блок–схема алгоритма работы приложения

После создания базы данных разработано WPF приложение для управляющей компании, которое позволяет эффективно управлять жилым фондом, следить за заявками на ремонт и обслуживание.

Разработана подсистема для обработки заявок на ремонт и обслуживание от жильцов, обеспечивающая следующий функционал:

- просмотр списка адресов и сотрудников;
- добавление/редактирование/удаление данных о заявке с указанием адреса и ответственного исполнителя;
- просмотр истории выполнения заявок по сотрудникам и адресам.
- просмотр отчетов.

Итоги работы приложения представлены на рисунках 4 – 17.

На рисунке 4 представлена главная страница администратора. Данная страница является центральным узлом управления для администратора системы. Она содержит панель навигации со следующими разделами: Дашборд, Заявки на ремонт, История заявок, Партнеры, Финансовые отчеты, Управление персоналом и Выход. В верхней части отображается информация о текущем пользователе.

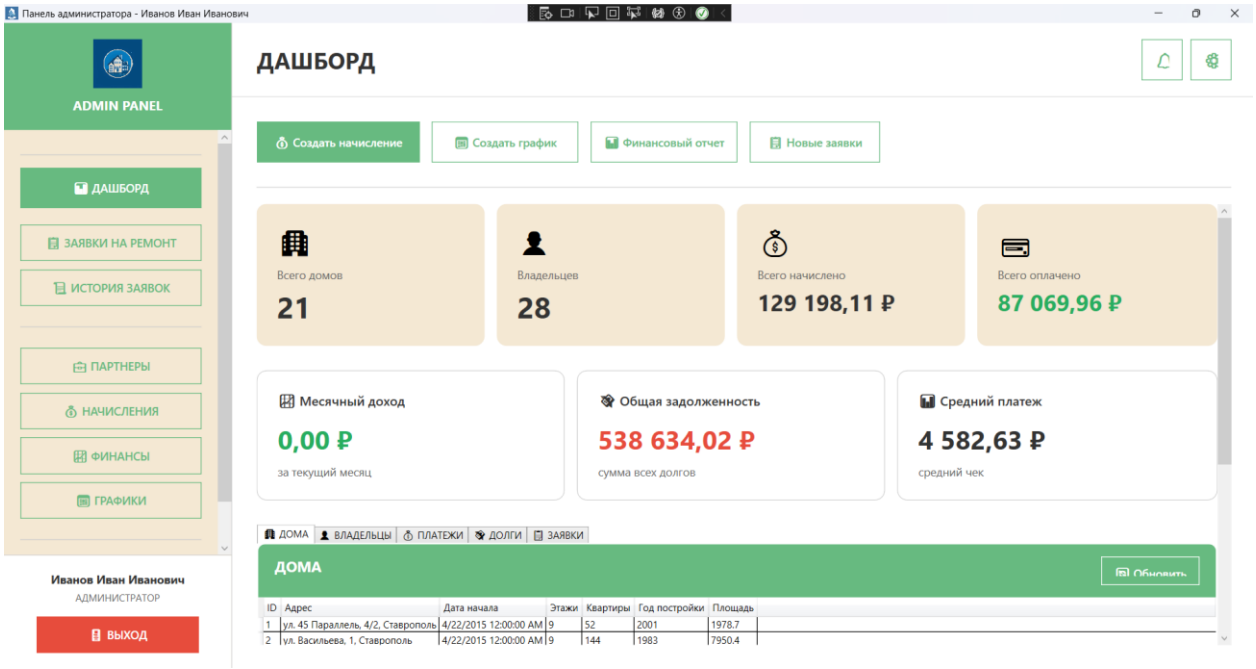

Рисунок 4 – Главная страница.

Ниже представлен код главной страницы в листингах 3 – 4.

Листинг 3 – Разметка главной страницы (AdminMainWindow.xaml)

```
<Window x:Class="HousingStock.AdminMainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="Панель администратора"
        Height="700"
        Width="1100"
        WindowStartupLocation="CenterScreen"
        Icon="Resources/icon.ico">

    <Window.Resources>
        <ResourceDictionary Source="Styles.xaml"/>
    </Window.Resources>

    <Grid Background="{StaticResource PrimaryBrush}">
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
        </Grid.RowDefinitions>
```

```
<!-- Шапка -->
<Border Grid.Row="0"
        Background="{StaticResource SecondaryBrush}"
        Padding="15">
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="Auto"/>
            <ColumnDefinition Width="*"/>
            <ColumnDefinition Width="Auto"/>
        </Grid.ColumnDefinitions>

        <!-- Логотип -->
        <Image Grid.Column="0"
               Source="Resources/logo.png"
               Height="40"
               Width="40"
               VerticalAlignment="Center"
               Margin="0,0,15,0"/>

        <TextBlock Grid.Column="1"
                   Text="Панель администратора"
                   Style="{StaticResource HeaderTextBlock}"
                   VerticalAlignment="Center"
                   HorizontalAlignment="Center"/>

        <StackPanel Grid.Column="2"
                    HorizontalAlignment="Right"
                    VerticalAlignment="Center">
            <TextBlock x:Name="UserNameText"
                       Style="{StaticResource LabelTextBlock}"
                       FontWeight="Bold"/>
            <TextBlock Text="Администратор"
                       Style="{StaticResource LabelTextBlock}"
                       Foreground="{StaticResource LightTextBrush}"/>
        </StackPanel>
    </Grid>
</Border>

<!-- Основное содержимое -->
<Grid Grid.Row="1">
    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="Auto"/>
        <ColumnDefinition Width="*"/>
    </Grid.ColumnDefinitions>

    <!-- Панель навигации -->
    <Border Grid.Column="0"
            Background="{StaticResource SecondaryBrush}"
            BorderBrush="{StaticResource BorderBrush}"
            BorderThickness="0,0,1,0"
            Width="220">

        <StackPanel Margin="15">
            <Button Content="📊 Дашборд"
                    Style="{StaticResource PrimaryButton}"
                    Height="45"
                    Margin="0,0,0,10"
                    Click="DashboardButton_Click"/>

            …
    </Border>
```

```
            <!-- Область данных -->
            <Border Grid.Column="1"
                    Background="{StaticResource PrimaryBrush}"
                    Padding="20">
                <Frame x:Name="MainFrame"
                       NavigationUIVisibility="Hidden"/>
            </Border>
        </Grid>
    </Grid>
</Window>
```

Листинг 4 – Код главной страницы (AdminMainWindow.xaml.cs)

```
using System.Windows;
using System.Windows.Controls;

namespace HousingStock
{
    public partial class AdminMainWindow : Window
    {
        public AdminMainWindow()
        {
            InitializeComponent();
            UserNameText.Text = CurrentUser.FullName;
            LoadDashboard();
        }

        private void LoadDashboard()
        {
            var dashboardPage = new AdminDashboardPage();
            MainFrame.Navigate(dashboardPage);
        }

        private void DashboardButton_Click(object sender, RoutedEventArgs e)
        {
            LoadDashboard();
        }

        private void ApplicationsButton_Click(object sender, RoutedEventArgs
e)
        {
            var applicationsPage = new ApplicationsPage();
            MainFrame.Navigate(applicationsPage);
        }

        private void HistoryButton_Click(object sender, RoutedEventArgs e)
        {
            var historyPage = new ApplicationHistoryPage();
            MainFrame.Navigate(historyPage);
        }

        private void PartnersButton_Click(object sender, RoutedEventArgs e)
        {
            var partnersPage = new PartnersPage();
            MainFrame.Navigate(partnersPage);
        }

        private void CreateAccrualButton_Click(object sender, RoutedEventArgs
e)
        {
            var createAccrualWindow = new CreateAccrualWindow();
            createAccrualWindow.Owner = this;
            createAccrualWindow.ShowDialog();
        }
```

```
        private void FinancialReportsButton_Click(object sender,
RoutedEventArgs e)
        {
            var financialReportsPage = new FinancialReportsPage();
            MainFrame.Navigate(financialReportsPage);
        }

        private void CreateScheduleButton_Click(object sender,
RoutedEventArgs e)
        {
            var window = new CreateScheduleWindow();
            window.Owner = this;
            window.ShowDialog();
        }

        private void StaffManagementButton_Click(object sender,
RoutedEventArgs e)
        {
            var page = new Page();
            var textBlock = new TextBlock
            {
                Text = "Управление персоналом:\n\n" +
                       "• Просмотр сотрудников\n" +
                       "• Назначение на работы\n" +
                       "• Распределение по заявкам\n" +
                       "• Учет рабочего времени",
                FontFamily = new System.Windows.Media.FontFamily("Segoe UI"),
                FontSize = 14,
                TextWrapping = TextWrapping.Wrap,
                HorizontalAlignment = HorizontalAlignment.Center,
                VerticalAlignment = VerticalAlignment.Center
            };
            page.Content = textBlock;
            MainFrame.Navigate(page);
        }

        private void LogoutButton_Click(object sender, RoutedEventArgs e)
        {
            var result = MessageBox.Show(
                "Вы действительно хотите выйти из системы?",
                "Подтверждение выхода",
                MessageBoxButton.YesNo,
                MessageBoxImage.Question);

            if (result == MessageBoxResult.Yes)
            {
                // Очистка статического класса CurrentUser - важно для
безопасности,
                // чтобы след. пользователь не получил доступ к данным
предыдущего
                CurrentUser.Clear();

                // Owner установлен для дочернего окна, но Parent не
определен у LoginWindow
                // Window.GetWindow(this) мог бы вернуть null, поэтому
используем просто Show()
                var loginWindow = new LoginWindow();
                loginWindow.Show();

                // Закрытие текущего окна с проверкой, не является ли оно
главным окном приложения
```

```
                // Если это главное окно - приложение завершится (поведение
по умолчанию)
                this.Close();
            }
        }
    }
}
```

На рисунке 5 представлена страница "Дашборд администратора". Эта страница содержит ключевые показатели системы: общее количество домов, владельцев, суммарные начисления и платежи. Также отображаются финансовые показатели: месячный доход, общая задолженность, средний платеж. Предоставлены быстрые действия: создание начисления, создание графика, просмотр отчетов.

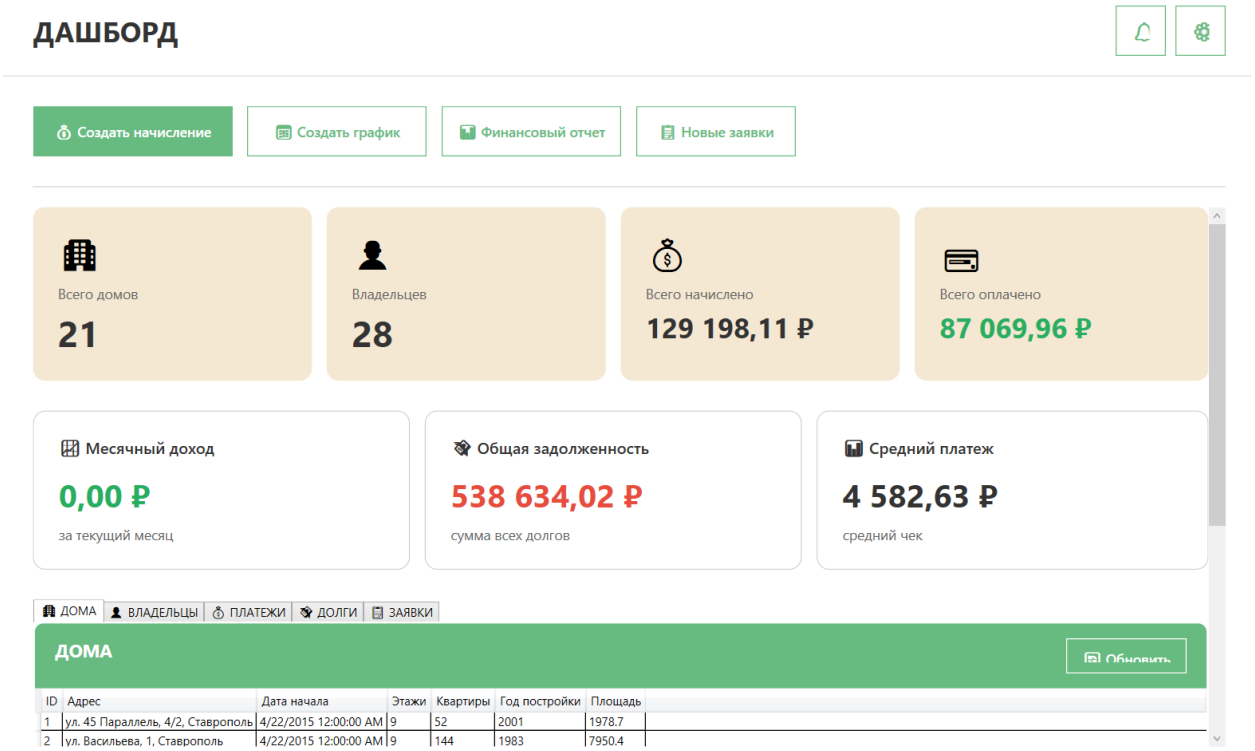

Рисунок 5 – Дашборд администратора

Ниже представлен код страницы "Дашборд администратора" в листингах 5 – 6.

Листинг 5 – Разметка страницы Дашборд администратора (AdminDashboardPage.xaml)

```
<Page x:Class="HousingStock.AdminDashboardPage"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      Title="Дашборд администратора"
      Background="{StaticResource PrimaryBrush}">
```

```
    <Page.Resources>
        <ResourceDictionary Source="Styles.xaml"/>
    </Page.Resources>

    <Grid>
        <TabControl>
            <!-- Вкладка 1: Статистика -->
            <TabItem Header="📊 Статистика">
                <ScrollViewer VerticalScrollBarVisibility="Auto">
                    <StackPanel Margin="20">
                        <!-- Заголовок -->
                        <TextBlock Text="Статистика системы"
                                   Style="{StaticResource HeaderTextBlock}"
                                   Margin="0,0,0,20"/>

                        <!-- Ключевые показатели -->
                        <Border Background="{StaticResource SecondaryBrush}"
                                CornerRadius="8"
                                Padding="20"
                                Margin="0,0,0,20">
                            <Grid>
                                <Grid.RowDefinitions>
                                    <RowDefinition Height="Auto"/>
                                    <RowDefinition Height="*"/>
                                </Grid.RowDefinitions>

                                <TextBlock Text="Ключевые показатели"
                                           Style="{StaticResource
SubheaderTextBlock}"
                                           Margin="0,0,0,15"/>

                                <Grid Grid.Row="1">
                                    <Grid.ColumnDefinitions>
                                        <ColumnDefinition Width="*"/>
                                        <ColumnDefinition Width="*"/>
                                        <ColumnDefinition Width="*"/>
                                        <ColumnDefinition Width="*"/>
                                    </Grid.ColumnDefinitions>

                                    <!-- Дома -->
                                    <StackPanel Grid.Column="0"
Margin="0,0,10,0">
                                        <TextBlock Text="Всего домов"
                                                   Style="{StaticResource
LabelTextBlock}"/>
                                        <TextBlock x:Name="TotalHousesText"
                                                   Text="0"
                                                   FontSize="28"
                                                   FontWeight="Bold"
                                                   FontFamily="Segoe UI"/>
                                    </StackPanel>

                                    <!-- Владельцы -->
                                    <StackPanel Grid.Column="1"
Margin="0,0,10,0">
                                        <TextBlock Text="Владельцев"
                                                   Style="{StaticResource
LabelTextBlock}"/>
                                        <TextBlock x:Name="TotalOwnersText"
                                                   Text="0"
                                                   FontSize="28"
                                                   FontWeight="Bold"
                                                   FontFamily="Segoe UI"/>
```

```
                                </StackPanel>

                                <!-- Всего начислено -->
                                <StackPanel Grid.Column="2"
Margin="0,0,10,0">
                                    <TextBlock Text="Всего начислено"
                                               Style="{StaticResource
LabelTextBlock}"/>
                                    <TextBlock x:Name="TotalAccruedText"
                                               Text="0 руб."
                                               FontSize="24"
                                               FontWeight="Bold"
                                               FontFamily="Segoe UI"/>
                                </StackPanel>

                                <!-- Всего оплачено -->
                                <StackPanel Grid.Column="3">
                                    <TextBlock Text="Всего оплачено"
                                               Style="{StaticResource
LabelTextBlock}"/>
                                    <TextBlock x:Name="TotalPaidText"
                                               Text="0 руб."
                                               FontSize="24"
                                               FontWeight="Bold"
                                               FontFamily="Segoe UI"
                                               Foreground="{StaticResource
SuccessBrush}"/>
                                </StackPanel>
                            </Grid>
                        </Grid>
                    </Border>

                    <!-- Финансовые показатели -->
                    <Border Background="{StaticResource SecondaryBrush}"
                            CornerRadius="8"
                            Padding="20"
                            Margin="0,0,0,20">
                        <StackPanel>
                            <TextBlock Text="Финансовые показатели"
                                       Style="{StaticResource
SubheaderTextBlock}"
                                       Margin="0,0,0,15"/>

                            <Grid>
                                <Grid.ColumnDefinitions>
                                    <ColumnDefinition Width="*"/>
                                    <ColumnDefinition Width="*"/>
                                    <ColumnDefinition Width="*"/>
                                </Grid.ColumnDefinitions>

                                <!-- Месячный доход -->
                                <StackPanel Grid.Column="0"
Margin="0,0,20,0">
                                    <TextBlock Text="Месячный доход"
                                               Style="{StaticResource
LabelTextBlock}"/>
                                    <TextBlock x:Name="MonthlyIncomeText"
                                               Text="0 руб."
                                               FontSize="24"
                                               FontWeight="Bold"
                                               FontFamily="Segoe UI"
                                               Foreground="{StaticResource
SuccessBrush}"/>
```

```
                                    </StackPanel>

                                    <!-- Общая задолженность -->
                                    <StackPanel Grid.Column="1"
Margin="0,0,20,0">
                                        <TextBlock Text="Общая задолженность"
                                                   Style="{StaticResource
LabelTextBlock}"/>
                                        <TextBlock x:Name="TotalDebtText"
                                                   Text="0 руб."
                                                   FontSize="24"
                                                   FontWeight="Bold"
                                                   FontFamily="Segoe UI"
                                                   Foreground="{StaticResource
ErrorBrush}"/>
                                    </StackPanel>

                                    <!-- Средний платеж -->
                                    <StackPanel Grid.Column="2">
                                        <TextBlock Text="Средний платеж"
                                                   Style="{StaticResource
LabelTextBlock}"/>
                                        <TextBlock
x:Name="AveragePaymentText"
                                                   Text="0 руб."
                                                   FontSize="24"
                                                   FontWeight="Bold"
                                                   FontFamily="Segoe UI"/>
                                    </StackPanel>
                                </Grid>
                            </StackPanel>
                        </Border>

                        <!-- Быстрые действия -->
                        <Border Background="{StaticResource SecondaryBrush}"
                                CornerRadius="8"
                                Padding="20">
                            <StackPanel>
                                <TextBlock Text="Быстрые действия"
                                           Style="{StaticResource
SubheaderTextBlock}"
                                           Margin="0,0,0,15"/>

                                <Grid>
                                    <Grid.ColumnDefinitions>
                                        <ColumnDefinition Width="*"/>
                                        <ColumnDefinition Width="*"/>
                                        <ColumnDefinition Width="*"/>
                                    </Grid.ColumnDefinitions>

                                    <Button Grid.Column="0"
                                            Content="💰 Создать начисление"
                                            Style="{StaticResource
PrimaryButton}"
                                            Height="45"
                                            Margin="0,0,10,0"

Click="CreateAccrualButton_Click"/>

                                    <Button Grid.Column="1"
                                            Content="📅 Создать график"
                                            Style="{StaticResource
PrimaryButton}"
```

```
                                                Height="45"
                                                Margin="10,0,10,0"

Click="CreateScheduleButton_Click"/>

                                        <Button Grid.Column="2"
                                                Content="📊 Посмотреть отчет"
                                                Style="{StaticResource
PrimaryButton}"
                                                Height="45"
                                                Margin="10,0,0,0"
                                                Click="ViewReportButton_Click"/>
                                    </Grid>
                                </StackPanel>
                            </Border>
                        </StackPanel>
                    </ScrollViewer>
                </TabItem>

                …….
    </Grid>
</Page>
```

Листинг 6 – Код страницы Дашборд администратора (AdminDashboardPage.xaml.cs)

```
using System;
using System.Data;
using System.Data.Entity;
using System.Linq;
using System.Windows;
using System.Windows.Controls;

namespace HousingStock
{
    public partial class AdminDashboardPage : Page
    {
        private Housing_stockEntities context;

        public AdminDashboardPage()
        {
            InitializeComponent();
            Loaded += AdminDashboardPage_Loaded;
            Unloaded += AdminDashboardPage_Unloaded;
        }

        private void AdminDashboardPage_Loaded(object sender, RoutedEventArgs
e)
        {
            try
            {
                context = new Housing_stockEntities();
                LoadStatistics();
                LoadHouses();
                LoadOwners();
                LoadPayments();
                LoadDebts();
                LoadApplications();
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Ошибка загрузки: {ex.Message}");
            }
```

```
        }

        private void AdminDashboardPage_Unloaded(object sender,
RoutedEventArgs e)
        {
            context?.Dispose();
        }

        private void LoadStatistics()
        {
            try
            {
                // Всего домов - просто Count
                int totalHouses = context.Apartment.Count();
                TotalHousesText.Text = totalHouses.ToString();

                // Всего владельцев
                int totalOwners = context.Owners.Count();
                TotalOwnersText.Text = totalOwners.ToString();

                // Всего начислено - Sum с проверкой null
                double totalAccrued = context.Payment.Sum(p => p.Accrued ??
0);
                TotalAccruedText.Text = $"{totalAccrued:C}";

                // Всего оплачено
                double totalPaid = context.Payment.Sum(p => p.Paid_for ?? 0);
                TotalPaidText.Text = $"{totalPaid:C}";

                // Месячный доход - текущий месяц
                string currentMonth = DateTime.Now.ToString("MM.yyyy");
                double monthlyIncome = context.Payment
                    .Where(p => p.Period != null &&
p.Period.Contains(currentMonth))
                    .Sum(p => p.Paid_for ?? 0);
                MonthlyIncomeText.Text = $"{monthlyIncome:C}";

                // Общая задолженность - Sum с ?? 0
                double totalDebt = context.Debt.Sum(d => (d.Water ?? 0) +
(d.Electric_power ?? 0));
                TotalDebtText.Text = $"{totalDebt:C}";

                // Средний платеж - DefaultIfEmpty чтобы избежать исключения
при пустой коллекции
                double avgPayment = context.Payment
                    .Where(p => p.Paid_for > 0)
                    .Select(p => p.Paid_for ?? 0)
                    .DefaultIfEmpty(0)
                    .Average();
                AveragePaymentText.Text = $"{avgPayment:C}";
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Ошибка статистики: {ex.Message}");
            }
        }

        private void LoadHouses()
        {
            try
            {
                // Проекция в анонимный тип
                var houses = context.Apartment
```

```
                    .Select(a => new
                    {
                        ID = a.ID,
                        Адрес = a.Adress ?? "",
                        Дата_начала = a.Beginning.HasValue ?
a.Beginning.Value.ToString("dd.MM.yyyy") : "",
                        Этажи = a.Floors.HasValue ? a.Floors.Value.ToString()
: "",
                        Квартиры = a.Flats.HasValue ?
a.Flats.Value.ToString() : "",
                        Год_постройки = a.Year.HasValue ?
a.Year.Value.ToString() : "",
                        Площадь = a.Area.HasValue ?
a.Area.Value.ToString("F1") : ""
                    })
                    .ToList();

                // Создаем DataTable вручную
                DataTable dt = new DataTable();

                // Добавляем колонки как строки
                dt.Columns.Add("ID", typeof(string));
                dt.Columns.Add("Адрес", typeof(string));
                dt.Columns.Add("Дата_начала", typeof(string));
                dt.Columns.Add("Этажи", typeof(string));
                dt.Columns.Add("Квартиры", typeof(string));
                dt.Columns.Add("Год_постройки", typeof(string));
                dt.Columns.Add("Площадь", typeof(string));
                foreach (var house in houses)
                {
                    dt.Rows.Add(
                        house.ID.ToString(),
                        house.Адрес,
                        house.Дата_начала,
                        house.Этажи,
                        house.Квартиры,
                        house.Год_постройки,
                        house.Площадь
                    );
                }

                HousesGrid.ItemsSource = dt.DefaultView;
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Ошибка домов: {ex.Message}");
            }
        }

        private void LoadOwners()
        {
            try
            {
                var owners = context.Owners
                    .Select(o => new
                    {
                        ID = o.ID,
                        ФИО_владельца = o.Name_owner ?? "",
                        Квартира = o.Flat.HasValue ? o.Flat.Value.ToString()
: "",
                        Телефон = o.Phone_number.HasValue ?
o.Phone_number.Value.ToString() : "",
```

```
                        Адрес_дома = o.Apartment != null ? o.Apartment.Adress
?? "" : ""
                    })
                    .ToList();
                DataTable dt = new DataTable();
                dt.Columns.Add("ID", typeof(string));
                dt.Columns.Add("ФИО_владельца", typeof(string));
                dt.Columns.Add("Квартира", typeof(string));
                dt.Columns.Add("Телефон", typeof(string));
                dt.Columns.Add("Адрес_дома", typeof(string));

                foreach (var owner in owners)
                {
                    dt.Rows.Add(
                        owner.ID.ToString(),
                        owner.ФИО_владельца,
                        owner.Квартира,
                        owner.Телефон,
                        owner.Адрес_дома
                    );
                }

                OwnersGrid.ItemsSource = dt.DefaultView;
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Ошибка владельцев: {ex.Message}");
            }
        }

        private void LoadPayments()
        {
            try
            {
                var payments = context.Payment
                    .Select(p => new
                    {
                        ID = p.ID,
                        Владелец = p.Owners != null ? p.Owners.Name_owner ??
"" : "",
                        Период = p.Period ?? "",
                        Начислено = p.Accrued.HasValue ?
$"{p.Accrued.Value:C}" : "0,00 ₽",
                        Оплачено = p.Paid_for.HasValue ?
$"{p.Paid_for.Value:C}" : "0,00 ₽",
                        Статус = GetPaymentStatus(p.Paid_for, p.Accrued)
                    })
                    .ToList();

                DataTable dt = new DataTable();
                dt.Columns.Add("ID", typeof(string));
                dt.Columns.Add("Владелец", typeof(string));
                dt.Columns.Add("Период", typeof(string));
                dt.Columns.Add("Начислено", typeof(string));
                dt.Columns.Add("Оплачено", typeof(string));
                dt.Columns.Add("Статус", typeof(string));

                foreach (var payment in payments)
                {
                    dt.Rows.Add(
                        payment.ID.ToString(),
                        payment.Владелец,
                        payment.Период,
```

```
                        payment.Начислено,
                        payment.Оплачено,
                        payment.Статус
                    );
                }

                PaymentsGrid.ItemsSource = dt.DefaultView;
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Ошибка платежей: {ex.Message}");
            }
        }

        ……..
        }
    }
}
```

На рисунке 6 представлена страница "Финансовые отчеты". Эта страница предназначена для анализа финансовых показателей компании. Отображаются: месячный доход, месячные расходы, чистая прибыль, рентабельность, общая задолженность, собрано платежей. Предоставлены быстрые отчеты: Ежемесячный, Квартальный, Годовой. Также отображаются списки последних платежей и должников.

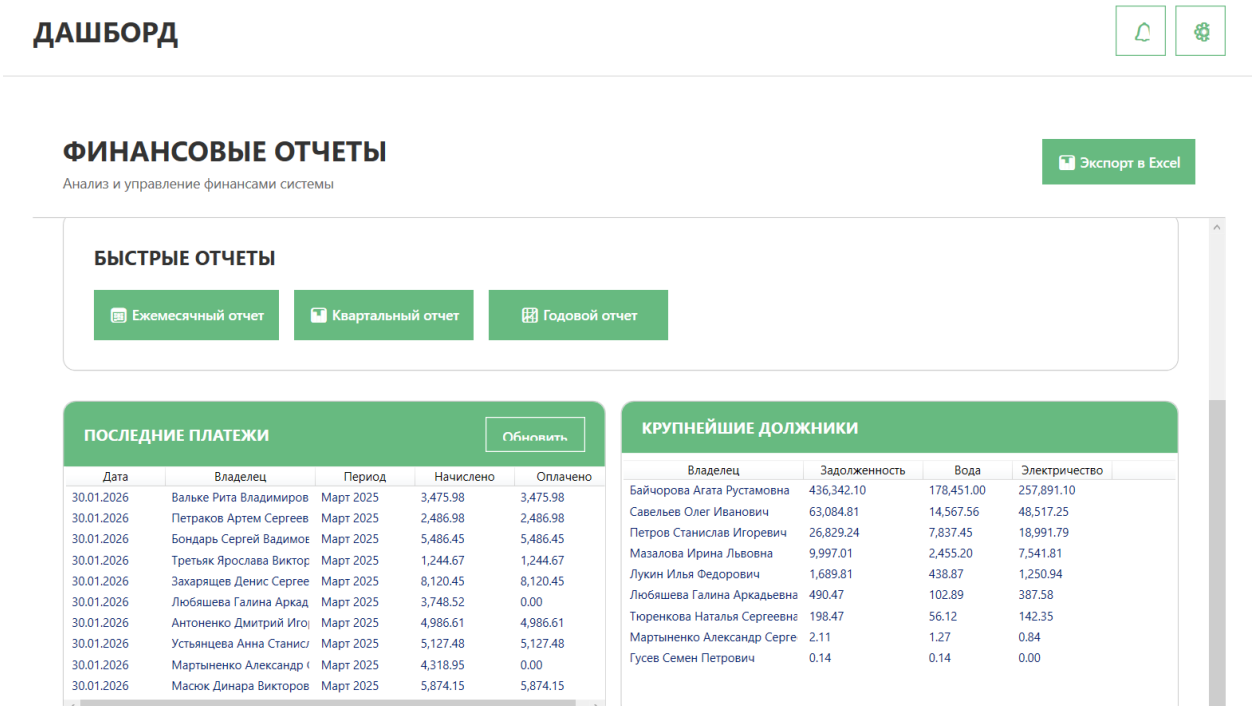

Рисунок 6 – Финансовые отчеты

Ниже представлен код страницы "Финансовые отчеты" в листингах 7 – 8.

Листинг 7 – Разметка страницы Финансовые отчеты (FinancialReportsPage.xaml)

```
<Page x:Class="HousingStock.FinancialReportsPage"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      Title="Финансовые отчеты"
      Background="White">

    <Page.Resources>
        <ResourceDictionary Source="Styles.xaml"/>
    </Page.Resources>

    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
        </Grid.RowDefinitions>

        <!-- Заголовок -->
        <Border Grid.Row="0"
                Background="#67BA80"
                Padding="15">
            <TextBlock Text="Финансовые отчеты"
                       FontSize="22"
                       FontWeight="Bold"
                       Foreground="White"
                       HorizontalAlignment="Center"/>
        </Border>

        <!-- Контент -->
        <Grid Grid.Row="1" Margin="20">
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width="*"/>
                <ColumnDefinition Width="*"/>
            </Grid.ColumnDefinitions>

            <!-- Левая колонка - Статистика -->
            <StackPanel Grid.Column="0" Margin="0,0,10,0">
                <Border Background="#F4E8D3"
                        CornerRadius="8"
                        Padding="20"
                        Margin="0,0,0,20">
                    <StackPanel>
                        <TextBlock Text="Финансовая статистика"
                                   FontSize="18"
                                   FontWeight="Bold"
                                   Margin="0,0,0,15"/>

                        <StackPanel                    Orientation="Horizontal"
Margin="0,0,0,10">
                            <TextBlock Text="Месячный доход:" Width="180"/>
                            <TextBlock x:Name="MonthlyIncomeText"
                                       FontWeight="SemiBold"
                                       Foreground="#27AE60"/>
                        </StackPanel>

                        <StackPanel                    Orientation="Horizontal"
Margin="0,0,0,10">
                            <TextBlock Text="Месячные расходы:" Width="180"/>
                            <TextBlock x:Name="MonthlyExpensesText"
                                       FontWeight="SemiBold"
```

```
                                Foreground="#E74C3C"/>
                    </StackPanel>

                    <StackPanel                    Orientation="Horizontal"
Margin="0,0,0,10">
                        <TextBlock Text="Чистая прибыль:" Width="180"/>
                        <TextBlock x:Name="NetProfitText"
                                FontWeight="SemiBold"/>
                    </StackPanel>

                    <StackPanel                    Orientation="Horizontal"
Margin="0,0,0,10">
                        <TextBlock Text="Рентабельность:" Width="180"/>
                        <TextBlock x:Name="ProfitabilityText"
                                FontWeight="SemiBold"/>
                    </StackPanel>

                    <StackPanel                    Orientation="Horizontal"
Margin="0,0,0,10">
                        <TextBlock       Text="Общая       задолженность:"
Width="180"/>
                        <TextBlock x:Name="TotalDebtText"
                                FontWeight="SemiBold"
                                Foreground="#E74C3C"/>
                    </StackPanel>

                    <StackPanel                    Orientation="Horizontal"
Margin="0,0,0,10">
                        <TextBlock Text="Собрано платежей:" Width="180"/>
                        <TextBlock x:Name="CollectedPaymentsText"
                                FontWeight="SemiBold"
                                Foreground="#27AE60"/>
                    </StackPanel>
                </StackPanel>
            </Border>

            <Border Background="#F4E8D3"
                    CornerRadius="8"
                    Padding="20">
                <StackPanel>
                    <TextBlock Text="Быстрые отчеты"
                               FontSize="18"
                               FontWeight="Bold"
                               Margin="0,0,0,15"/>

                    <Button x:Name="GenerateMonthlyReportButton"
                            Content="🖩 Ежемесячный отчет"
                            Background="#67BA80"
                            Foreground="White"
                            FontSize="14"
                            FontWeight="SemiBold"
                            BorderThickness="0"
                            Height="45"
                            Margin="0,0,0,10"
                            Click="GenerateMonthlyReportButton_Click"/>

                    <Button x:Name="GenerateQuarterlyReportButton"
                            Content="📊 Квартальный отчет"
                            Background="#67BA80"
                            Foreground="White"
                            FontSize="14"
                            FontWeight="SemiBold"
                            BorderThickness="0"
```

```
                        Height="45"
                        Margin="0,0,0,10"
                        Click="GenerateQuarterlyReportButton_Click"/>

                <Button x:Name="GenerateAnnualReportButton"
                        Content="☑ Годовой отчет"
                        Background="#67BA80"
                        Foreground="White"
                        FontSize="14"
                        FontWeight="SemiBold"
                        BorderThickness="0"
                        Height="45"
                        Margin="0,0,0,10"
                        Click="GenerateAnnualReportButton_Click"/>

                <Button x:Name="ExportToExcelButton"
                        Content="💾 Экспорт в Excel"
                        Background="#67BA80"
                        Foreground="White"
                        FontSize="14"
                        FontWeight="SemiBold"
                        BorderThickness="0"
                        Height="45"
                        Click="ExportToExcelButton_Click"/>
            </StackPanel>
        </Border>
    </StackPanel>

    <!-- Правая колонка - Платежи -->
    <StackPanel Grid.Column="1" Margin="10,0,0,0">
        <Border Background="#F4E8D3"
                CornerRadius="8"
                Padding="20"
                Margin="0,0,0,20">
            <StackPanel>
                <Grid>
                    <Grid.ColumnDefinitions>
                        <ColumnDefinition Width="*"/>
                        <ColumnDefinition Width="Auto"/>
                    </Grid.ColumnDefinitions>

                    <TextBlock Text="Недавние платежи"
                               FontSize="18"
                               FontWeight="Bold"
                               Grid.Column="0"/>

                    <Button x:Name="RefreshButton"
                            Content="🔄 Обновить"
                            Background="Transparent"
                            Foreground="#67BA80"
                            BorderBrush="#67BA80"
                            BorderThickness="1"
                            Height="35"
                            Width="100"
                            Grid.Column="1"
                            FontSize="13"
                            FontWeight="SemiBold"
                            Click="RefreshButton_Click"/>
                </Grid>

                <ListView x:Name="RecentPaymentsList"
                          Height="250"
                          Margin="0,15,0,0"
```

```
                                BorderThickness="1"
                                BorderBrush="#CCCCCC">
                            <ListView.View>
                                <GridView>
                                    <GridViewColumn Header="Дата" Width="120"
DisplayMemberBinding="{Binding PaymentDate, StringFormat=dd.MM.yyyy}"/>
                                    <GridViewColumn          Header="Владелец"
Width="150"
DisplayMemberBinding="{Binding OwnerName}"/>
                                    <GridViewColumn            Header="Период"
Width="100"
DisplayMemberBinding="{Binding Period}"/>
                                    <GridViewColumn         Header="Начислено"
Width="100"
DisplayMemberBinding="{Binding Accrued, StringFormat=N2}"/>
                                    <GridViewColumn          Header="Оплачено"
Width="100"
DisplayMemberBinding="{Binding Paid, StringFormat=N2}"/>
                                </GridView>
                            </ListView.View>
                        </ListView>
                    </StackPanel>
                </Border>

                <Border Background="#F4E8D3"
                        CornerRadius="8"
                        Padding="20">
                    <StackPanel>
                        <TextBlock Text="Должники"
                                   FontSize="18"
                                   FontWeight="Bold"
                                   Margin="0,0,0,15"/>

                        <ListView x:Name="DebtorsList"
                                  Height="200"
                                  BorderThickness="1"
                                  BorderBrush="#CCCCCC">
                            <ListView.View>
                                <GridView>
                                    <GridViewColumn          Header="Владелец"
Width="150" DisplayMemberBinding="{Binding OwnerName}"/>
                                    <GridViewColumn     Header="Задолженность"
Width="120"
DisplayMemberBinding="{Binding TotalDebt, StringFormat=N2}"/>
                                    <GridViewColumn Header="Вода" Width="80"
DisplayMemberBinding="{Binding WaterDebt, StringFormat=N2}"/>
                                    <GridViewColumn     Header="Электричество"
Width="100"
DisplayMemberBinding="{Binding ElectricityDebt, StringFormat=N2}"/>
                                </GridView>
                            </ListView.View>
                        </ListView>
                    </StackPanel>
                </Border>
            </StackPanel>
```

```
        </Grid>
    </Grid>
</Page>
```

Листинг 8 – Код страницы Финансовые отчеты (FinancialReportsPage.xaml.cs)

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Windows;
using System.Windows.Controls;

namespace HousingStock
{
   public partial class FinancialReportsPage : Page
   {
      …..
      private void RefreshButton_Click(object sender, RoutedEventArgs e)
      {
         try
         {
            LoadFinancialStatistics();
            LoadRecentPayments();
            LoadDebtors();
            MessageBox.Show("Данные обновлены", "Обновление",
               MessageBoxButton.OK, MessageBoxImage.Information);
         }
         catch (Exception ex)
         {
            MessageBox.Show($"Ошибка обновления: {ex.Message}", "Ошибка",
               MessageBoxButton.OK, MessageBoxImage.Error);
         }
      }
   }
}
```

На рисунке 7 представлена страница "Партнеры". Эта страница позволяет управлять информацией о партнерах компании: поставщиках материалов, подрядчиках, сервисных организациях. Реализованы операции: добавление, редактирование, удаление партнеров. Доступна фильтрация по статусу (Активен, Неактивен, Приостановлен).

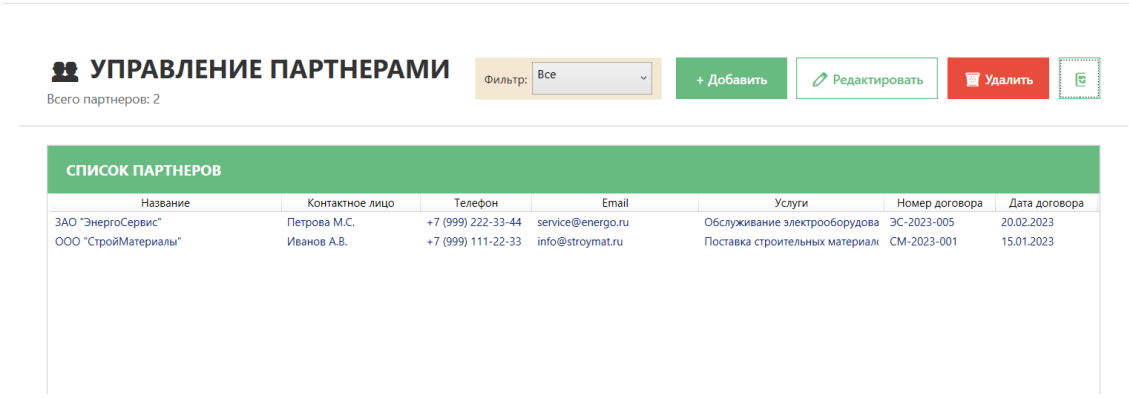

Рисунок 7 – Страница "Партнеры"

Ниже представлен код страницы "Партнеры" в листингах 9 – 10.

Листинг 9 – Разметка страницы Партнеры (PartnersPage.xaml)

```
<Page x:Class="HousingStock.PartnersPage"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      Title="Управление партнерами"
      Background="{StaticResource PrimaryBrush}">

    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
            <RowDefinition Height="Auto"/>
        </Grid.RowDefinitions>

        <!-- Панель инструментов -->
        <Border Grid.Row="0"
                Background="{StaticResource SecondaryBrush}"
                Padding="10">
            <StackPanel Orientation="Horizontal">
                <Button x:Name="AddButton"
                        Content="Добавить партнера"
                        Style="{StaticResource PrimaryButton}"
                        Width="171"
                        Height="35"
                        Margin="0,0,10,0"
                        Click="AddButton_Click"
                        ToolTip="Добавить нового партнера"/>

                <Button x:Name="EditButton"
                        Content="Редактировать"
                        Style="{StaticResource PrimaryButton}"
                        Width="148"
                        Height="35"
                        Margin="0,0,10,0"
                        Click="EditButton_Click"
                        ToolTip="Редактировать выбранного партнера"/>

                <Button x:Name="DeleteButton"
                        Content="Удалить"
                        Style="{StaticResource DangerButton}"
                        Width="100"
                        Height="35"
                        Margin="0,0,10,0"
```

```
                        Click="DeleteButton_Click"
                        ToolTip="Удалить выбранного партнера"/>
                <Button x:Name="RefreshButton"
                        Content="Обновить"
                        Style="{StaticResource PrimaryButton}"
                        Width="108"
                        Height="35"
                        Click="RefreshButton_Click"
                        ToolTip="Обновить список партнеров"/>

                <TextBlock Text="Фильтр по статусу:"
                           VerticalAlignment="Center"
                           Margin="0,0,5,0"/>
                <ComboBox x:Name="StatusFilter"
                          Width="154"
                          Height="30"
                          SelectedIndex="0"
                          SelectionChanged="StatusFilter_SelectionChanged">
                    <ComboBoxItem Content="Все"/>
                    <ComboBoxItem Content="Активен"/>
                    <ComboBoxItem Content="Неактивен"/>
                    <ComboBoxItem Content="Приостановлен"/>
                </ComboBox>
            </StackPanel>
        </Border>

……
    </Grid>
</Page>
```

Листинг 10 – Код страницы Партнеры (PartnersPage.xaml.cs)

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;
using System.Linq;
using System.Windows;
using System.Windows.Controls;

namespace HousingStock
{
    public partial class PartnersPage : Page
    {
        private string connectionString = "Data Source=(local);Initial
Catalog=Housing_stock;Integrated Security=True";
        private List<Partner> partners;

        public class Partner
        {
            public int PartnerID { get; set; }
            public string PartnerName { get; set; }
            public string ContactPerson { get; set; }
            public string Phone { get; set; }
            public string Email { get; set; }
            public string Services { get; set; }
            public DateTime? ContractDate { get; set; }
            public string ContractNumber { get; set; }
            public string Status { get; set; }
        }

        public PartnersPage()
        {
            InitializeComponent();
```

```
            Loaded += PartnersPage_Loaded;
        }

        private void PartnersPage_Loaded(object sender, RoutedEventArgs e)
        {
            LoadPartners();
        }

        private void LoadPartners()
        {
            try
            {
                partners = new List<Partner>();

                bool tableExists = CheckIfTableExists("Partners");

                if (tableExists)
                {
                    LoadPartnersFromDatabase();
                }
                else
                {
                    CreateMockPartners();

                    MessageBox.Show(
                        "Таблица Partners не найдена в базе данных.\n" +
                        "Используются тестовые данные для демонстрации.",
                        "Внимание",
                        MessageBoxButton.OK,
                        MessageBoxImage.Warning);
                }

                UpdatePartnersDisplay();
            }
            catch (Exception ex)
            {
                MessageBox.Show(
                    $"Ошибка при загрузке партнеров: {ex.Message}\n" +
                    "Будут использованы тестовые данные.",
                    "Ошибка",
                    MessageBoxButton.OK,
                    MessageBoxImage.Error);

                CreateMockPartners();
                UpdatePartnersDisplay();
            }
        }

        private bool CheckIfTableExists(string tableName)
        {
            try
            {
                using (SqlConnection connection = new
SqlConnection(connectionString))
                {
                    connection.Open();

                    string query = @"
                        SELECT COUNT(*)
                        FROM INFORMATION_SCHEMA.TABLES
                        WHERE TABLE_SCHEMA = 'dbo'
                        AND TABLE_NAME = @TableName";
```

```
                    using (SqlCommand command = new SqlCommand(query,
connection))
                    {
                        command.Parameters.AddWithValue("@TableName",
tableName);
                        int count = Convert.ToInt32(command.ExecuteScalar());
                        return count > 0;
                    }
                }
            }
            catch
            {
                return false;
            }
        }

        private void LoadPartnersFromDatabase()
        {
            try
            {
                using (SqlConnection connection = new
SqlConnection(connectionString))
                {
                    connection.Open();

                    string query = @"
                        SELECT
                            PartnerID,
                            PartnerName,
                            ContactPerson,
                            Phone,
                            Email,
                            Services,
                            ContractDate,
                            ContractNumber,
                            Status
                        FROM Partners
                        ORDER BY PartnerName";

                    using (SqlCommand command = new SqlCommand(query,
connection))
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            Partner partner = new Partner
                            {
                                PartnerID =
Convert.ToInt32(reader["PartnerID"]),
                                PartnerName =
reader["PartnerName"].ToString(),
                                ContactPerson =
reader["ContactPerson"].ToString(),
                                Phone = reader["Phone"].ToString(),
                                Email = reader["Email"].ToString(),
                                Services = reader["Services"].ToString(),
                                Status = reader["Status"].ToString(),
                                ContractNumber =
reader["ContractNumber"].ToString()
                            };

                            if (reader["ContractDate"] != DBNull.Value)
                            {
```

```
                                partner.ContractDate =
Convert.ToDateTime(reader["ContractDate"]);
                            }

                            partners.Add(partner);
                        }
                    }
                }
            }
            catch (Exception ex)
            {
                throw new Exception($"Ошибка загрузки данных: {ex.Message}",
ex);
            }
        }

        ……

        private void StatusFilter_SelectionChanged(object sender,
SelectionChangedEventArgs e)
        {
            try
            {
                UpdatePartnersDisplay();
            }
            catch (Exception ex)
            {
                MessageBox.Show(
                    $"Ошибка при изменении фильтра: {ex.Message}",
                    "Ошибка",
                    MessageBoxButton.OK,
                    MessageBoxImage.Error);
            }
        }
    }
}
```

На рисунке 8 представлено окно создания начисления. Это модальное окно позволяет администраторам создавать новые начисления для владельцев квартир. Содержит поля: Владелец (выпадающий список), Период начисления, Сумма начисления, Услуга (выпадающий список), Комментарий. Реализована валидация данных и обновление таблицы задолженностей.

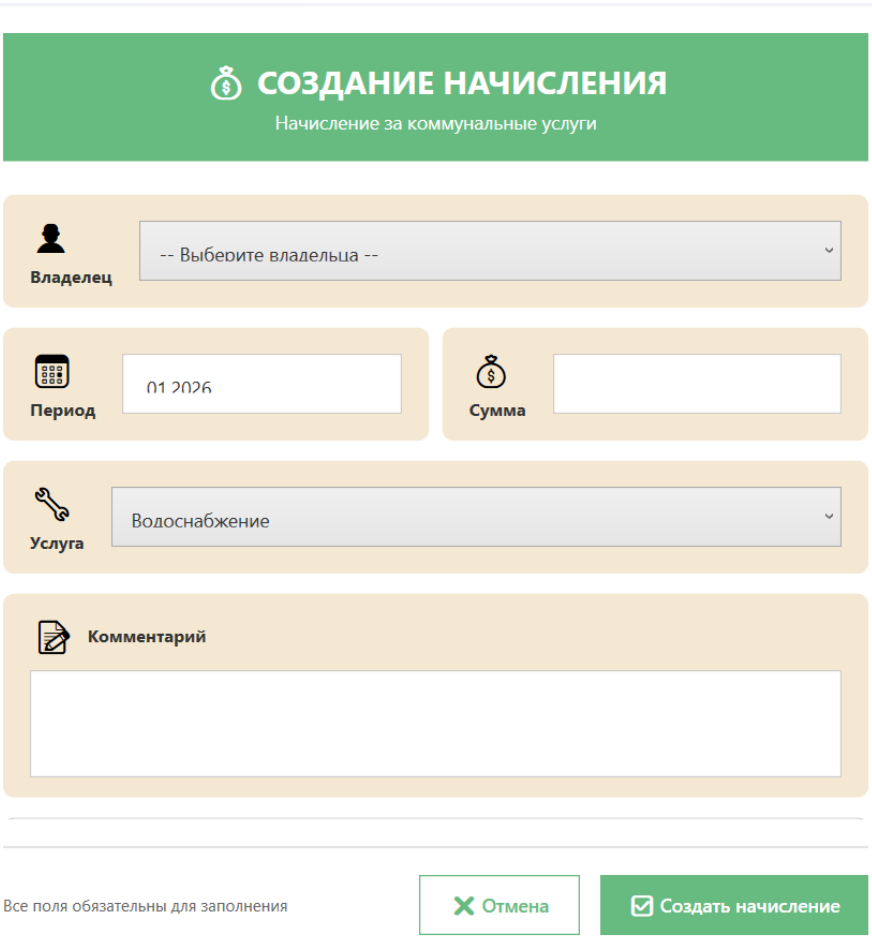

Рисунок 8 – Окно «Создание начисления»

Ниже представлен код окна создания начисления в листингах 11 – 12.

Листинг 11 – Разметка окна создания начисления (CreateAccrualWindow.xaml)

```
<Window x:Class="HousingStock.CreateAccrualWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="Создание начисления"
        Height="450"
        Width="500"
        WindowStartupLocation="CenterOwner"
        ResizeMode="NoResize">

    <Grid Margin="15">
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
            <RowDefinition Height="Auto"/>
        </Grid.RowDefinitions>

        <!-- Заголовок -->
        <Border Grid.Row="0"
                Background="{StaticResource AccentBrush}"
                CornerRadius="5"
                Padding="10"
                Margin="0,0,0,15">
            <TextBlock Text="Создание начисления за коммунальные услуги"
                       FontSize="18"
                       FontWeight="Bold"
```

```
                Foreground="White"
                HorizontalAlignment="Center"/>
        </Border>

        <!-- Поля формы -->
        <StackPanel Grid.Row="1" Margin="0,0,0,15">

            <!-- Владелец -->
            <StackPanel Margin="0,0,0,15">
                <Label Content="Владелец *"
                       FontWeight="Bold"
                       Foreground="#333333"/>
                <ComboBox x:Name="OwnerComboBox"
                          Height="35"
                          Margin="0,5,0,0"
                          FontSize="14"
                          Padding="5"
                          BorderThickness="1"
                          BorderBrush="{StaticResource BorderBrush}"
                          DisplayMemberPath="Name"/>
            </StackPanel>

            <!-- Период -->
            <StackPanel Margin="0,0,0,15">
                <Label Content="Период начисления *"
                       FontWeight="Bold"
                       Foreground="#333333"/>
                <TextBox x:Name="PeriodTextBox"
                         Height="35"
                         Margin="0,5,0,0"
                         FontSize="14"
                         Padding="5"
                         BorderThickness="1"
                         BorderBrush="{StaticResource BorderBrush}"
                         ToolTip="Формат: ММ.ГГГГ (например: 11.2024)"/>
            </StackPanel>

            <!-- Сумма -->
            <StackPanel Margin="0,0,0,15">
                <Label Content="Сумма начисления *"
                       FontWeight="Bold"
                       Foreground="#333333"/>
                <TextBox x:Name="AmountTextBox"
                         Height="35"
                         Margin="0,5,0,0"
                         FontSize="14"
                         Padding="5"
                         BorderThickness="1"
                         BorderBrush="{StaticResource BorderBrush}"
                         ToolTip="Введите сумму в рублях"/>
            </StackPanel>

            <!-- Услуга -->
            <StackPanel Margin="0,0,0,15">
                <Label Content="Услуга"
                       FontWeight="Bold"
                       Foreground="#333333"/>
                <ComboBox x:Name="ServiceComboBox"
                          Height="35"
                          Margin="0,5,0,0"
                          FontSize="14"
                          Padding="5"
                          BorderThickness="1"
```

```
                        BorderBrush="{StaticResource BorderBrush}"
                        SelectedIndex="0">
                    <ComboBoxItem Content="Водоснабжение"/>
                    <ComboBoxItem Content="Электроснабжение"/>
                    <ComboBoxItem Content="Отопление"/>
                    <ComboBoxItem Content="Содержание дома"/>
                </ComboBox>
            </StackPanel>

            <!-- Комментарий -->
            <StackPanel>
                <Label Content="Комментарий"
                       FontWeight="Bold"
                       Foreground="#333333"/>
                <TextBox x:Name="CommentTextBox"
                         Height="60"
                         Margin="0,5,0,0"
                         FontSize="14"
                         Padding="5"
                         BorderThickness="1"
                         BorderBrush="{StaticResource BorderBrush}"
                         TextWrapping="Wrap"
                         AcceptsReturn="True"
                         VerticalScrollBarVisibility="Auto"/>
            </StackPanel>
        </StackPanel>

        <!-- Кнопки -->
        <StackPanel Grid.Row="2"
                    Orientation="Horizontal"
                    HorizontalAlignment="Right"
                    Margin="0,10,0,0">

            <Button x:Name="SaveButton"
                    Content="Создать начисление"
                    Style="{StaticResource PrimaryButton}"
                    Width="180"
                    Height="40"
                    FontSize="14"
                    FontWeight="SemiBold"
                    Click="SaveButton_Click"/>

            <Button x:Name="CancelButton"
                    Content="Отмена"
                    Style="{StaticResource SecondaryButton}"
                    Width="120"
                    Height="40"
                    FontSize="14"
                    FontWeight="SemiBold"
                    Margin="10,0,0,0"
                    Click="CancelButton_Click"/>
        </StackPanel>
    </Grid>
</Window>
```

Листинг 12 – Код окна создания начисления (CreateAccrualWindow.xaml.cs)

```
using System;
using System.Data.SqlClient;
using System.Windows;

namespace HousingStock
{
    public partial class CreateAccrualWindow : Window
    {
```

```
        public CreateAccrualWindow()
        {
            InitializeComponent();
            LoadOwners();
            PeriodTextBox.Text = DateTime.Now.ToString("MM.yyyy");
        }

        private void LoadOwners()
        {
            try
            {
                using (SqlConnection connection = new
SqlConnection(connectionString))
                {
                    connection.Open();

                    string query = "SELECT ID, Name_owner FROM Owners ORDER BY
Name_owner";
                    using (SqlCommand command = new SqlCommand(query,
connection))
                    using (var reader = command.ExecuteReader())
                    {
                        OwnerComboBox.Items.Clear();
                        OwnerComboBox.Items.Add(new { Id = 0, Name = "--
Выберите владельца --" });

                        while (reader.Read())
                        {
                            OwnerComboBox.Items.Add(new
                            {
                                Id = reader["ID"],
                                Name = reader["Name_owner"].ToString()
                            });
                        }

                        OwnerComboBox.SelectedIndex = 0;
                    }
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Ошибка загрузки владельцев: {ex.Message}",
"Ошибка",
                    MessageBoxButton.OK, MessageBoxImage.Error);
            }
        }

        …..
                        using (SqlCommand insertCmd = new
SqlCommand(insertQuery, connection))
                        {
                            insertCmd.Parameters.AddWithValue("@OwnerId",
ownerId);
                            insertCmd.Parameters.AddWithValue("@Amount",
(double)amount);
                            insertCmd.Parameters.AddWithValue("@ServiceType",
serviceType ?? "Водоснабжение");
                            insertCmd.ExecuteNonQuery();
                        }
                    }
                }
```

```
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Ошибка обновления задолженности:
{ex.Message}", "Ошибка",
                    MessageBoxButton.OK, MessageBoxImage.Warning);
            }
        }

        private void CancelButton_Click(object sender, RoutedEventArgs e)
        {
            DialogResult = false;
            Close();
        }
    }
}
```

На листинге 13 представлен файл стилей. Этот файл содержит единые стили для всего приложения: цвета, стили текста, кнопок, полей ввода. Использование централизованных стилей обеспечивает единообразие интерфейса и упрощает его изменение.

Листинг 13 – Файл стилей (Styles.xaml)

```
<ResourceDictionary
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
                    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">

    <!-- Цвета -->
    <Color x:Key="PrimaryColor">#FFFFFFFF</Color>
    <Color x:Key="SecondaryColor">#FFF4E8D3</Color>
    <Color x:Key="AccentColor">#FF67BA80</Color>
    <Color x:Key="BorderColor">#FFCCCCCC</Color>
    <Color x:Key="ErrorColor">#FFE74C3C</Color>
    <Color x:Key="SuccessColor">#FF27AE60</Color>
    <Color x:Key="LightTextColor">#FF666666</Color>
    <Color x:Key="DarkTextColor">#FF333333</Color>

    <SolidColorBrush x:Key="PrimaryBrush" Color="{StaticResource
PrimaryColor}"/>
    <SolidColorBrush x:Key="SecondaryBrush" Color="{StaticResource
SecondaryColor}"/>
    <SolidColorBrush x:Key="AccentBrush" Color="{StaticResource
AccentColor}"/>
    <SolidColorBrush x:Key="BorderBrush" Color="{StaticResource
BorderColor}"/>
    <SolidColorBrush x:Key="ErrorBrush" Color="{StaticResource ErrorColor}"/>
    <SolidColorBrush x:Key="SuccessBrush" Color="{StaticResource
SuccessColor}"/>
    <SolidColorBrush x:Key="LightTextBrush" Color="{StaticResource
LightTextColor}"/>
    <SolidColorBrush x:Key="DarkTextBrush" Color="{StaticResource
DarkTextColor}"/>

    <!-- Простые стили (без сложных триггеров) -->
    <Style x:Key="HeaderTextBlock" TargetType="TextBlock">
        <Setter Property="FontFamily" Value="Segoe UI"/>
        <Setter Property="FontSize" Value="24"/>
        <Setter Property="FontWeight" Value="Bold"/>
        <Setter Property="Foreground" Value="{StaticResource
DarkTextBrush}"/>
```

```
    </Style>

    <Style x:Key="SubheaderTextBlock" TargetType="TextBlock">
        <Setter Property="FontFamily" Value="Segoe UI"/>
        <Setter Property="FontSize" Value="18"/>
        <Setter Property="FontWeight" Value="SemiBold"/>
        <Setter Property="Foreground" Value="{StaticResource
DarkTextBrush}"/>
    </Style>

    <Style x:Key="BodyTextBlock" TargetType="TextBlock">
        <Setter Property="FontFamily" Value="Segoe UI"/>
        <Setter Property="FontSize" Value="14"/>
        <Setter Property="Foreground" Value="{StaticResource
DarkTextBrush}"/>
    </Style>

    <Style x:Key="LabelTextBlock" TargetType="TextBlock">
        <Setter Property="FontFamily" Value="Segoe UI"/>
        <Setter Property="FontSize" Value="13"/>
        <Setter Property="FontWeight" Value="Medium"/>
        <Setter Property="Foreground" Value="{StaticResource
DarkTextBrush}"/>
    </Style>

    <!-- Простой стиль кнопки -->
    <Style x:Key="PrimaryButton" TargetType="Button">
        <Setter Property="FontFamily" Value="Segoe UI"/>
        <Setter Property="FontSize" Value="14"/>
        <Setter Property="FontWeight" Value="SemiBold"/>
        <Setter Property="Background" Value="{StaticResource AccentBrush}"/>
        <Setter Property="Foreground" Value="White"/>
        <Setter Property="BorderThickness" Value="0"/>
        <Setter Property="Padding" Value="15,10"/>
        <Setter Property="Height" Value="40"/>
        <Setter Property="Cursor" Value="Hand"/>
    </Style>

    <Style x:Key="SecondaryButton" TargetType="Button">
        <Setter Property="FontFamily" Value="Segoe UI"/>
        <Setter Property="FontSize" Value="14"/>
        <Setter Property="FontWeight" Value="SemiBold"/>
        <Setter Property="Background" Value="Transparent"/>
        <Setter Property="Foreground" Value="{StaticResource AccentBrush}"/>
        <Setter Property="BorderBrush" Value="{StaticResource AccentBrush}"/>
        <Setter Property="BorderThickness" Value="1"/>
        <Setter Property="Padding" Value="15,10"/>
        <Setter Property="Height" Value="40"/>
        <Setter Property="Cursor" Value="Hand"/>
    </Style>

    <Style x:Key="DangerButton" TargetType="Button">
        <Setter Property="FontFamily" Value="Segoe UI"/>
        <Setter Property="FontSize" Value="14"/>
        <Setter Property="FontWeight" Value="SemiBold"/>
        <Setter Property="Background" Value="{StaticResource ErrorBrush}"/>
        <Setter Property="Foreground" Value="White"/>
        <Setter Property="BorderThickness" Value="0"/>
        <Setter Property="Padding" Value="15,10"/>
        <Setter Property="Height" Value="40"/>
        <Setter Property="Cursor" Value="Hand"/>
    </Style>
```

```
    <!-- Стили полей ввода -->
    <Style x:Key="TextBoxStyle" TargetType="TextBox">
        <Setter Property="FontFamily" Value="Segoe UI"/>
        <Setter Property="FontSize" Value="14"/>
        <Setter Property="Foreground" Value="{StaticResource
DarkTextBrush}"/>
        <Setter Property="Background" Value="White"/>
        <Setter Property="BorderBrush" Value="{StaticResource BorderBrush}"/>
        <Setter Property="BorderThickness" Value="1"/>
        <Setter Property="Padding" Value="10"/>
        <Setter Property="Height" Value="40"/>
        <Setter Property="VerticalContentAlignment" Value="Center"/>
    </Style>

    <Style x:Key="PasswordBoxStyle" TargetType="PasswordBox">
        <Setter Property="FontFamily" Value="Segoe UI"/>
        <Setter Property="FontSize" Value="14"/>
        <Setter Property="Foreground" Value="{StaticResource
DarkTextBrush}"/>
        <Setter Property="Background" Value="White"/>
        <Setter Property="BorderBrush" Value="{StaticResource BorderBrush}"/>
        <Setter Property="BorderThickness" Value="1"/>
        <Setter Property="Padding" Value="10"/>
        <Setter Property="Height" Value="40"/>
        <Setter Property="VerticalContentAlignment" Value="Center"/>
    </Style>

    <!-- Стили ComboBox -->
    <Style x:Key="ComboBoxStyle" TargetType="ComboBox">
        <Setter Property="FontFamily" Value="Segoe UI"/>
        <Setter Property="FontSize" Value="14"/>
        <Setter Property="Foreground" Value="{StaticResource
DarkTextBrush}"/>
        <Setter Property="Background" Value="White"/>
        <Setter Property="BorderBrush" Value="{StaticResource BorderBrush}"/>
        <Setter Property="BorderThickness" Value="1"/>
        <Setter Property="Padding" Value="10"/>
        <Setter Property="Height" Value="40"/>
    </Style>

    <!-- Стили ListView -->
    <Style x:Key="ListViewStyle" TargetType="ListView">
        <Setter Property="FontFamily" Value="Segoe UI"/>
        <Setter Property="FontSize" Value="13"/>
        <Setter Property="Background" Value="White"/>
        <Setter Property="BorderBrush" Value="{StaticResource BorderBrush}"/>
        <Setter Property="BorderThickness" Value="1"/>
    </Style>

    <!-- Стили Label -->
    <Style x:Key="LabelStyle" TargetType="Label">
        <Setter Property="FontFamily" Value="Segoe UI"/>
        <Setter Property="FontSize" Value="14"/>
        <Setter Property="FontWeight" Value="Medium"/>
        <Setter Property="Foreground" Value="{StaticResource
DarkTextBrush}"/>
    </Style>

</ResourceDictionary>
```

Модуль № 3. Сопровождение и обслуживание программного обеспечения компьютерных систем

На рисунке 10 представлена страница "Заявки на ремонт". Эта страница позволяет администраторам и руководителям просматривать все заявки в системе. Для сотрудников и жителей отображаются только их собственные заявки. Реализована фильтрация по статусу (Все, Открыта, В работе, Завершена, Отменена). Доступны кнопки: Добавить заявку, Редактировать, Удалить, Обновить.

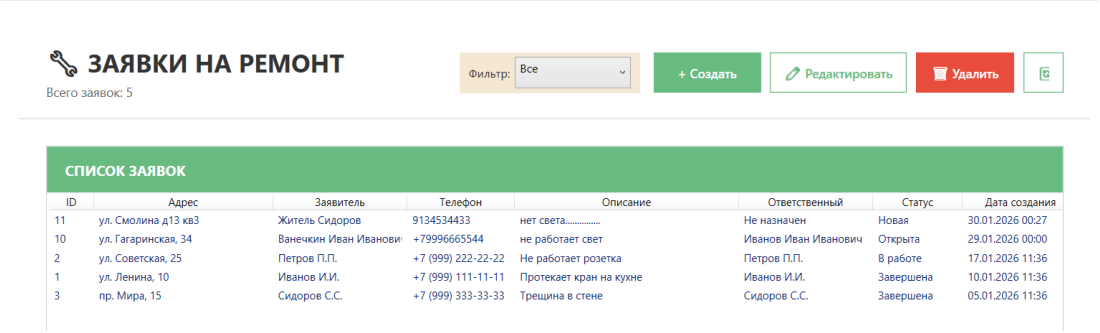

Рисунок 10 – Страница "Заявки на ремонт"

Ниже представлен код страницы "Заявки на ремонт" в листингах 14 – 15.

Листинг 14 – Разметка страницы Заявки на ремонт (ApplicationsPage.xaml)

```
<Page x:Class="HousingStock.ApplicationsPage"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      Title="Заявки на ремонт"
      Background="White">

    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
            <RowDefinition Height="Auto"/>
        </Grid.RowDefinitions>

        <!-- Панель инструментов -->
        <Border Grid.Row="0"
                Background="#F4E8D3"
                Padding="10">
            <StackPanel Orientation="Horizontal">
                <Button x:Name="AddButton"
                        Content="Добавить заявку"
                        Width="140"
                        Height="35"
                        Margin="0,0,10,0"
```

```
                    Click="AddButton_Click"
                    ToolTip="Создать новую заявку на ремонт"/>

            <Button x:Name="EditButton"
                    Content="Редактировать"
                    Width="120"
                    Height="35"
                    Margin="0,0,10,0"
                    Click="EditButton_Click"
                    ToolTip="Редактировать выбранную заявку"/>

            <Button x:Name="DeleteButton"
                    Content="Удалить"
                    Background="#E74C3C"
                    Width="100"
                    Height="35"
                    Margin="0,0,10,0"
                    Click="DeleteButton_Click"
                    ToolTip="Удалить выбранную заявку"/>

            <Button x:Name="RefreshButton"
                    Content="Обновить"
                    Width="100"
                    Height="35"
                    Margin="0,0,20,0"
                    Click="RefreshButton_Click"
                    ToolTip="Обновить список заявок"/>

            <TextBlock Text="Фильтр по статусу:"
                       VerticalAlignment="Center"
                       Margin="0,0,5,0"/>

            <ComboBox x:Name="StatusFilter"
                      Width="120"
                      Height="30"
                      SelectedIndex="0"
                      SelectionChanged="StatusFilter_SelectionChanged">
                <ComboBoxItem Content="Все"/>
                <ComboBoxItem Content="Открыта"/>
                <ComboBoxItem Content="В работе"/>
                <ComboBoxItem Content="Завершена"/>
                <ComboBoxItem Content="Отменена"/>
            </ComboBox>
        </StackPanel>
    </Border>

    <!-- Список заявок -->
    <ListView x:Name="ApplicationsList"
              Grid.Row="1"
              Margin="10"
              SelectionMode="Single"
              MouseDoubleClick="ApplicationsList_MouseDoubleClick">
        <ListView.View>
            <GridView>
                <GridViewColumn Header="ID" Width="50"
DisplayMemberBinding="{Binding Id}"/>
                <GridViewColumn Header="Адрес" Width="200"
DisplayMemberBinding="{Binding Address}"/>
                <GridViewColumn Header="Заявитель" Width="150"
DisplayMemberBinding="{Binding ApplicantName}"/>
                <GridViewColumn Header="Телефон" Width="120"
DisplayMemberBinding="{Binding Phone}"/>
```

```
                    <GridViewColumn Header="Описание" Width="250"
DisplayMemberBinding="{Binding Description}"/>
                    <GridViewColumn Header="Ответственный" Width="150"
DisplayMemberBinding="{Binding Responsible}"/>
                    <GridViewColumn Header="Статус" Width="100"
DisplayMemberBinding="{Binding Status}"/>
                    <GridViewColumn Header="Дата создания" Width="120"
                                    DisplayMemberBinding="{Binding CreateDate,
StringFormat='dd.MM.yyyy HH:mm'}"/>
                </GridView>
            </ListView.View>
        </ListView>

        <Border Grid.Row="2"
                Background="#E0E0E0"
                Height="30">
            <TextBlock x:Name="StatusText"
                       Text="Загружено заявок: 0"
                       VerticalAlignment="Center"
                       Margin="10,0,0,0"/>
        </Border>
    </Grid>
</Page>
```

Листинг 16 – Код страницы Заявки на ремонт (ApplicationsPage.xaml.cs)

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Windows;
using System.Windows.Controls;

namespace HousingStock
{
    public partial class ApplicationsPage : Page
    {
        private string connectionString = "Data Source=(local);Initial
Catalog=Housing_stock;Integrated Security=True";
        private List<RepairApplication> applications;

        public class RepairApplication
        {
            public int Id { get; set; }
            public string Address { get; set; }
            public string ApplicantName { get; set; }
            public string Phone { get; set; }
            public string Description { get; set; }
            public string Responsible { get; set; }
            public string Status { get; set; }
            public DateTime CreateDate { get; set; }
            public DateTime? CompleteDate { get; set; }
            public string AssignedEmployee { get; set; }
        }

        public ApplicationsPage()
        {
            InitializeComponent();
            Loaded += ApplicationsPage_Loaded;

            // Скрываем кнопки в зависимости от роли
            ConfigureButtonsForRole();
        }
```

```
        private void ConfigureButtonsForRole()
        {
            string role = CurrentUser.RoleName.ToLower();

            // Для жителей и сотрудников скрываем кнопки удаления
            if (role == "житель" || role == "сотрудник")
            {
                DeleteButton.Visibility = Visibility.Collapsed;
            }

            // Для жителей показываем только кнопку добавления
            if (role == "житель")
            {
                EditButton.Visibility = Visibility.Collapsed;
                StatusFilter.Visibility = Visibility.Collapsed;
                StatusText.Text = "Мои заявки";
            }
        }

        private void ApplicationsPage_Loaded(object sender, RoutedEventArgs
e)
        {
            LoadApplications();
        }

        private void LoadApplications()
        {
            try
            {
                applications = new List<RepairApplication>();

                using (SqlConnection connection = new
SqlConnection(connectionString))
                {
                    connection.Open();

                    // Проверяем, существует ли таблица Applications
                    string checkTableQuery = @"
                        SELECT COUNT(*)
                        FROM INFORMATION_SCHEMA.TABLES
                        WHERE TABLE_SCHEMA = 'dbo'
                        AND TABLE_NAME = 'Applications'";

                    SqlCommand checkCmd = new SqlCommand(checkTableQuery,
connection);
                    int tableExists =
Convert.ToInt32(checkCmd.ExecuteScalar());

                    if (tableExists > 0)
                    {
                        // Таблица существует, загружаем данные с фильтрацией
по роли
                        LoadApplicationsFromDatabase(connection);
                    }
                    else
                    {
                        // Таблицы нет, создаем тестовые данные
                        CreateMockApplications();
                        MessageBox.Show(
                            "Таблица Applications не найдена в базе
данных.\n" +
                            "Используются тестовые данные для демонстрации.",
                            "Внимание",
```

```
                            MessageBoxButton.OK,
                            MessageBoxImage.Warning);
                    }
                }

                UpdateApplicationsDisplay();
            }
            catch (Exception ex)
            {
                MessageBox.Show(
                    $"Ошибка при загрузке заявок: {ex.Message}\n" +
                    "Будут использованы тестовые данные.",
                    "Ошибка",
                    MessageBoxButton.OK,
                    MessageBoxImage.Error);

                CreateMockApplications();
                UpdateApplicationsDisplay();
            }
        }

        private void LoadApplicationsFromDatabase(SqlConnection connection)
        {
            try
            {
                string role = CurrentUser.RoleName.ToLower();
                string currentUserName = CurrentUser.FullName;

                // Базовый запрос
                string query = @"
                    SELECT
                        ID,
                        Address,
                        ApplicantName,
                        Phone,
                        Description,
                        Responsible,
                        Status,
                        CreateDate,
                        CompleteDate,
                        AssignedEmployee
                    FROM Applications";

                // Добавляем фильтрацию в зависимости от роли
                if (role == "сотрудник")
                {
                    query += " WHERE AssignedEmployee LIKE '%' + @UserName +
'%' OR Responsible LIKE '%' + @UserName + '%'";
                }
                else if (role == "житель")
                {
                    query += " WHERE ApplicantName LIKE '%' + @UserName +
'%'";
                }

                query += " ORDER BY CreateDate DESC";

                using (SqlCommand command = new SqlCommand(query,
connection))
                {
                    // Добавляем параметр только если нужно
                    if (role == "сотрудник" || role == "житель")
                    {
```

```
                        command.Parameters.AddWithValue("@UserName",
currentUserName);
                    }

                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            RepairApplication application = new
RepairApplication
                            {
                                Id = reader["ID"] != DBNull.Value ?
Convert.ToInt32(reader["ID"]) : 0,
                                Address = reader["Address"] != DBNull.Value ?
reader["Address"].ToString() : "Не указан",
                                ApplicantName = reader["ApplicantName"] !=
DBNull.Value ? reader["ApplicantName"].ToString() : "Неизвестно",
                                Phone = reader["Phone"] != DBNull.Value ?
reader["Phone"].ToString() : "Не указан",
                                Description = reader["Description"] !=
DBNull.Value ? reader["Description"].ToString() : "Нет описания",
                                Status = reader["Status"] != DBNull.Value ?
reader["Status"].ToString() : "Открыта",
                                Responsible = reader["Responsible"] !=
DBNull.Value ? reader["Responsible"].ToString() : "Не назначен",
                                AssignedEmployee = reader["AssignedEmployee"]
!= DBNull.Value ? reader["AssignedEmployee"].ToString() : "Не назначен"
                            };

                            if (reader["CreateDate"] != DBNull.Value)
                            {
                                application.CreateDate =
Convert.ToDateTime(reader["CreateDate"]);
                            }
                            else
                            {
                                application.CreateDate = DateTime.Now;
                            }

                            if (reader["CompleteDate"] != DBNull.Value)
                            {
                                application.CompleteDate =
Convert.ToDateTime(reader["CompleteDate"]);
                            }

                            applications.Add(application);
                        }
                    }
                }
            }
            catch (Exception ex)
            {
                throw new Exception($"Ошибка загрузки данных из базы:
{ex.Message}", ex);
            }
        }

        private void CreateMockApplications()
        {
            string role = CurrentUser.RoleName.ToLower();
            string currentUserName = CurrentUser.FullName;

            …….
```

```
        private void StatusFilter_SelectionChanged(object sender,
SelectionChangedEventArgs e)
        {
            try
            {
                UpdateApplicationsDisplay();
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Ошибка при изменении фильтра:
{ex.Message}",
                    "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);
            }
        }
    }
}
```

На рисунке 11 представлено окно редактирования заявки. Это модальное окно используется для добавления новых заявок и редактирования существующих. Окно содержит следующие поля: Адрес (обязательное), ФИО заявителя (обязательное), Телефон (обязательное), Описание проблемы (обязательное), Ответственный исполнитель (обязательное), Статус заявки (обязательное), Дата создания. Реализована валидация данных, включая проверку формата телефона.

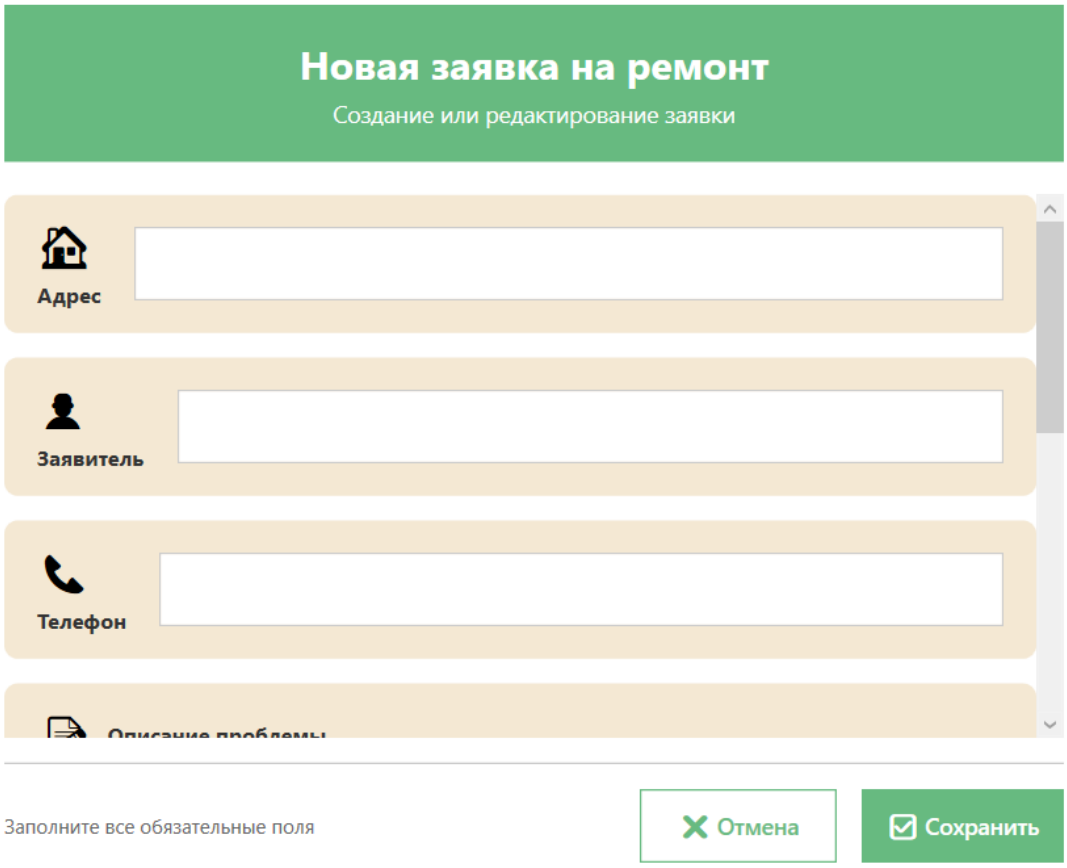

Рисунок 11 – Окно "Редактирование заявки"

Ниже представлен код окна редактирования заявки в листингах 16 – 17.

Листинг 16 – Разметка окна редактирования заявки (EditApplicationWindow.xaml)

```
<Window x:Class="HousingStock.EditApplicationWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="Заявка на ремонт"
        Height="550"
        Width="650"
        WindowStartupLocation="CenterOwner"
        ResizeMode="NoResize"
        Icon="Resources/icon.ico">

    <Grid Margin="15">
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
            <RowDefinition Height="Auto"/>
        </Grid.RowDefinitions>

        <!-- Заголовок -->
        <Border Grid.Row="0"
                Background="{StaticResource AccentBrush}"
                CornerRadius="5"
```

```
            Padding="10"
            Margin="0,0,0,15">
        <TextBlock x:Name="HeaderText"
                   Text="Форма заявки на ремонт"
                   FontSize="18"
                   FontWeight="Bold"
                   Foreground="White"
                   HorizontalAlignment="Center"/>
    </Border>

    <!-- Поля формы -->
    <ScrollViewer Grid.Row="1"
                  VerticalScrollBarVisibility="Auto"
                  Margin="0,0,0,10">
        <Grid>
            <Grid.RowDefinitions>
                <RowDefinition Height="Auto"/>
                <RowDefinition Height="Auto"/>
                <RowDefinition Height="Auto"/>
                <RowDefinition Height="Auto"/>
                <RowDefinition Height="Auto"/>
                <RowDefinition Height="Auto"/>
                <RowDefinition Height="Auto"/>
            </Grid.RowDefinitions>

            <!-- Адрес -->
            <StackPanel Grid.Row="0" Margin="0,0,0,15">
                <Label Content="Адрес *"
                       FontWeight="Bold"
                       Foreground="#333333"/>
                <TextBox x:Name="AddressBox"
                         Height="35"
                         Margin="0,5,0,0"
                         FontSize="14"
                         Padding="5"
                         BorderThickness="1"
                         BorderBrush="{StaticResource BorderBrush}"
                         ToolTip="Введите адрес дома или квартиры"/>
            </StackPanel>

            <!-- ФИО заявителя -->
            <StackPanel Grid.Row="1" Margin="0,0,0,15">
                <Label Content="ФИО заявителя *"
                       FontWeight="Bold"
                       Foreground="#333333"/>
                <TextBox x:Name="NameBox"
                         Height="35"
                         Margin="0,5,0,0"
                         FontSize="14"
                         Padding="5"
                         BorderThickness="1"
                         BorderBrush="{StaticResource BorderBrush}"
                         ToolTip="Введите полное имя заявителя"/>
            </StackPanel>

            <!-- Телефон -->
            <StackPanel Grid.Row="2" Margin="0,0,0,15">
                <Label Content="Контактный телефон *"
                       FontWeight="Bold"
                       Foreground="#333333"/>
                <TextBox x:Name="PhoneBox"
                         Height="35"
                         Margin="0,5,0,0"
```

```
                FontSize="14"
                Padding="5"
                BorderThickness="1"
                BorderBrush="{StaticResource BorderBrush}"
                ToolTip="Введите номер телефона для связи"/>
    </StackPanel>

    <!-- Описание проблемы -->
    <StackPanel Grid.Row="3" Margin="0,0,0,15">
        <Label Content="Описание проблемы *"
               FontWeight="Bold"
               Foreground="#333333"/>
        <TextBox x:Name="DescriptionBox"
                 Height="100"
                 Margin="0,5,0,0"
                 FontSize="14"
                 Padding="5"
                 BorderThickness="1"
                 BorderBrush="{StaticResource BorderBrush}"
                 TextWrapping="Wrap"
                 AcceptsReturn="True"
                 VerticalScrollBarVisibility="Auto"
                 ToolTip="Подробно опишите проблему"/>
    </StackPanel>

    <!-- Ответственный -->
    <StackPanel Grid.Row="4" Margin="0,0,0,15">
        <Label Content="Ответственный исполнитель *"
               FontWeight="Bold"
               Foreground="#333333"/>
        <ComboBox x:Name="ResponsibleBox"
                  Height="35"
                  Margin="0,5,0,0"
                  FontSize="14"
                  Padding="5"
                 BorderThickness="1"
                 BorderBrush="{StaticResource BorderBrush}"
                 ToolTip="Выберите ответственного сотрудника">
            <ComboBoxItem Content="Иванов И.И."/>
            <ComboBoxItem Content="Петров П.П."/>
            <ComboBoxItem Content="Сидоров С.С."/>
            <ComboBoxItem Content="Козлов К.К."/>
        </ComboBox>
    </StackPanel>

    <!-- Статус -->
    <StackPanel Grid.Row="5" Margin="0,0,0,15">
        <Label Content="Статус заявки *"
               FontWeight="Bold"
               Foreground="#333333"/>
        <ComboBox x:Name="StatusBox"
                  Height="35"
                  Margin="0,5,0,0"
                  FontSize="14"
                  Padding="5"
                 BorderThickness="1"
                 BorderBrush="{StaticResource BorderBrush}"
                 ToolTip="Выберите текущий статус заявки">
            <ComboBoxItem Content="Открыта"/>
            <ComboBoxItem Content="В работе"/>
            <ComboBoxItem Content="Завершена"/>
            <ComboBoxItem Content="Отменена"/>
        </ComboBox>
```

```
                </StackPanel>

                <!-- Дата создания -->
                <StackPanel Grid.Row="6">
                    <Label Content="Дата создания"
                           FontWeight="Bold"
                           Foreground="#333333"/>
                    <DatePicker x:Name="DateBox"
                                Height="35"
                                Margin="0,5,0,0"
                                FontSize="14"
                                Padding="5"
                                BorderThickness="1"
                                BorderBrush="{StaticResource BorderBrush}"
                                SelectedDateFormat="Short"/>
                </StackPanel>
            </Grid>
        </ScrollViewer>

        <!-- Кнопки -->
        <StackPanel Grid.Row="2"
                    Orientation="Horizontal"
                    HorizontalAlignment="Right"
                    Margin="0,10,0,0">

            <Button x:Name="SaveButton"
                    Content="Сохранить"
                    Style="{StaticResource PrimaryButton}"
                    Width="120"
                    Height="40"
                    FontSize="14"
                    FontWeight="SemiBold"
                    Click="SaveButton_Click"/>

            <Button x:Name="CancelButton"
                    Content="Отмена"
                    Style="{StaticResource SecondaryButton}"
                    Width="120"
                    Height="40"
                    FontSize="14"
                    FontWeight="SemiBold"
                    Margin="10,0,0,0"
                    Click="CancelButton_Click"/>
        </StackPanel>
    </Grid>
</Window>
```

Листинг 17 – Код окна редактирования заявки (EditApplicationWindow.xaml.cs)

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Text.RegularExpressions;
using System.Windows;
using System.Windows.Controls;

namespace HousingStock
{
    public partial class EditApplicationWindow : Window
    {
        private ApplicationsPage.RepairApplication application;
```

```
        private bool isEditMode = false;
        private List<Employee> employees;

        public class Employee
        {
            public int EmployeeID { get; set; }
            public string FullName { get; set; }
            public string Position { get; set; }
        }

        public EditApplicationWindow()
        {
            InitializeComponent();
            InitializeNewApplication();
            LoadEmployees();
        }

        public EditApplicationWindow(ApplicationsPage.RepairApplication
existingApplication)
        {
            InitializeComponent();
            isEditMode = true;
            application = existingApplication;
            LoadEmployees();
            LoadApplicationData();
        }

        private void LoadEmployees()
        {
            try
            {
                employees = new List<Employee>();

                using (SqlConnection connection = new
SqlConnection(connectionString))
                {
                    connection.Open();

                    string query = "SELECT EmployeeID, FullName, Position
FROM Employees WHERE Status = 'Активен' ORDER BY FullName";

                    using (SqlCommand command = new SqlCommand(query,
connection))
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            employees.Add(new Employee
                            {
                                EmployeeID =
Convert.ToInt32(reader["EmployeeID"]),
                                FullName = reader["FullName"].ToString(),
                                Position = reader["Position"].ToString()
                            });
                        }
                    }
                }

                // Очищаем ComboBox и добавляем сотрудников из базы данных
                ResponsibleBox.Items.Clear();
                foreach (var employee in employees)
                {
```

```
                    ResponsibleBox.Items.Add(new ComboBoxItem { Content =
employee.FullName, Tag = employee.EmployeeID });
                }

                if (ResponsibleBox.Items.Count > 0)
                    ResponsibleBox.SelectedIndex = 0;
            }
            catch (Exception ex)
            {
                // Если не удалось загрузить сотрудников, используем
статические данные
                ResponsibleBox.Items.Clear();
                ResponsibleBox.Items.Add(new ComboBoxItem { Content = "Иванов
И.И." });
                ResponsibleBox.Items.Add(new ComboBoxItem { Content = "Петров
П.П." });
                ResponsibleBox.Items.Add(new ComboBoxItem { Content =
"Сидоров С.С." });
                ResponsibleBox.Items.Add(new ComboBoxItem { Content = "Козлов
К.К." });

                if (ResponsibleBox.Items.Count > 0)
                    ResponsibleBox.SelectedIndex = 0;
            }
        }

        private void InitializeNewApplication()
        {
            HeaderText.Text = "Новая заявка на ремонт";
            DateBox.SelectedDate = DateTime.Now;
            StatusBox.SelectedIndex = 0;
        }

        private void LoadApplicationData()
        {
            try
            {
                HeaderText.Text = $"Редактирование заявки #{application.Id}";

                AddressBox.Text = application.Address;
                NameBox.Text = application.ApplicantName;
                PhoneBox.Text = application.Phone;
                DescriptionBox.Text = application.Description;
                DateBox.SelectedDate = application.CreateDate;

                // Устанавливаем ответственного
                foreach (ComboBoxItem item in ResponsibleBox.Items)
                {
                    if (item.Content.ToString() == application.Responsible)
                    {
                        ResponsibleBox.SelectedItem = item;
                        break;
                    }
                }

                // Устанавливаем статус
                foreach (ComboBoxItem item in StatusBox.Items)
                {
                    if (item.Content.ToString() == application.Status)
                    {
                        StatusBox.SelectedItem = item;
                        break;
                    }
```

```
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Ошибка загрузки данных заявки:
{ex.Message}",
                    "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);
                this.Close();
            }
        }

        private void SaveButton_Click(object sender, RoutedEventArgs e)
        {
            if (ValidateData())
            {
                try
                {
                    if (isEditMode)
                    {
                        UpdateApplication();
                    }
                    else
                    {
                        CreateApplication();
                    }

                    DialogResult = true;
                    this.Close();
                }
                catch (SqlException ex)
                {
                    if (ex.Number == 208) // Таблица не существует
                    {
                        DialogResult = true; // Для тестовых данных считаем
успешным
                        this.Close();
                    }
                    else
                    {
                        MessageBox.Show($"Ошибка базы данных при сохранении:
{ex.Message}",
                            "Ошибка базы данных", MessageBoxButton.OK,
MessageBoxImage.Error);
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show($"Ошибка при сохранении: {ex.Message}",
                        "Ошибка", MessageBoxButton.OK,
MessageBoxImage.Error);
                }
            }
        }

        private bool ValidateData()
        {
            string errorMessage = "";

            if (string.IsNullOrWhiteSpace(AddressBox.Text))
            {
                errorMessage += "• Введите адрес\n";
                AddressBox.Focus();
            }
```

```
            if (string.IsNullOrWhiteSpace(NameBox.Text))
            {
                errorMessage += "• Введите ФИО заявителя\n";
                if (string.IsNullOrEmpty(errorMessage)) NameBox.Focus();
            }

            if (string.IsNullOrWhiteSpace(PhoneBox.Text))
            {
                errorMessage += "• Введите контактный телефон\n";
                if (string.IsNullOrEmpty(errorMessage)) PhoneBox.Focus();
            }
            else if (!IsValidPhone(PhoneBox.Text))
            {
                errorMessage += "• Введите корректный номер телефона\n";
                if (string.IsNullOrEmpty(errorMessage)) PhoneBox.Focus();
            }

            if (string.IsNullOrWhiteSpace(DescriptionBox.Text))
            {
                errorMessage += "• Введите описание проблемы\n";
                if (string.IsNullOrEmpty(errorMessage))
DescriptionBox.Focus();
            }

            if (ResponsibleBox.SelectedItem == null)
            {
                errorMessage += "• Выберите ответственного исполнителя\n";
                if (string.IsNullOrEmpty(errorMessage))
ResponsibleBox.Focus();
            }

            if (StatusBox.SelectedItem == null)
            {
                errorMessage += "• Выберите статус заявки\n";
                if (string.IsNullOrEmpty(errorMessage)) StatusBox.Focus();
            }

            if (!string.IsNullOrEmpty(errorMessage))
            {
                MessageBox.Show($"Обнаружены
ошибки:\n\n{errorMessage}\nПожалуйста, исправьте указанные поля.",
                    "Ошибка валидации", MessageBoxButton.OK,
MessageBoxImage.Warning);
                return false;
            }

            return true;
        }

        private bool IsValidPhone(string phone)
        {
            try
            {
                string digitsOnly = new string(phone.Where(c =>
char.IsDigit(c)).ToArray());

                if (digitsOnly.Length >= 10)
                {
                    return true;
                }

                string pattern = @"^[\d\s\(\)\+-]+$";
```

```
                return Regex.IsMatch(phone, pattern);
            }
            catch
            {
                return false;
            }
        }



        private void CancelButton_Click(object sender, RoutedEventArgs e)
        {
            DialogResult = false;
            this.Close();
        }
    }
}
```

На рисунке 12 представлено окно авторизации. Это стартовое окно приложения, которое запрашивает логин и пароль пользователя. Реализована проверка учетных данных в базе данных. При успешной аутентификации открывается соответствующее главное окно в зависимости от роли пользователя (администратор, руководитель, сотрудник, житель).

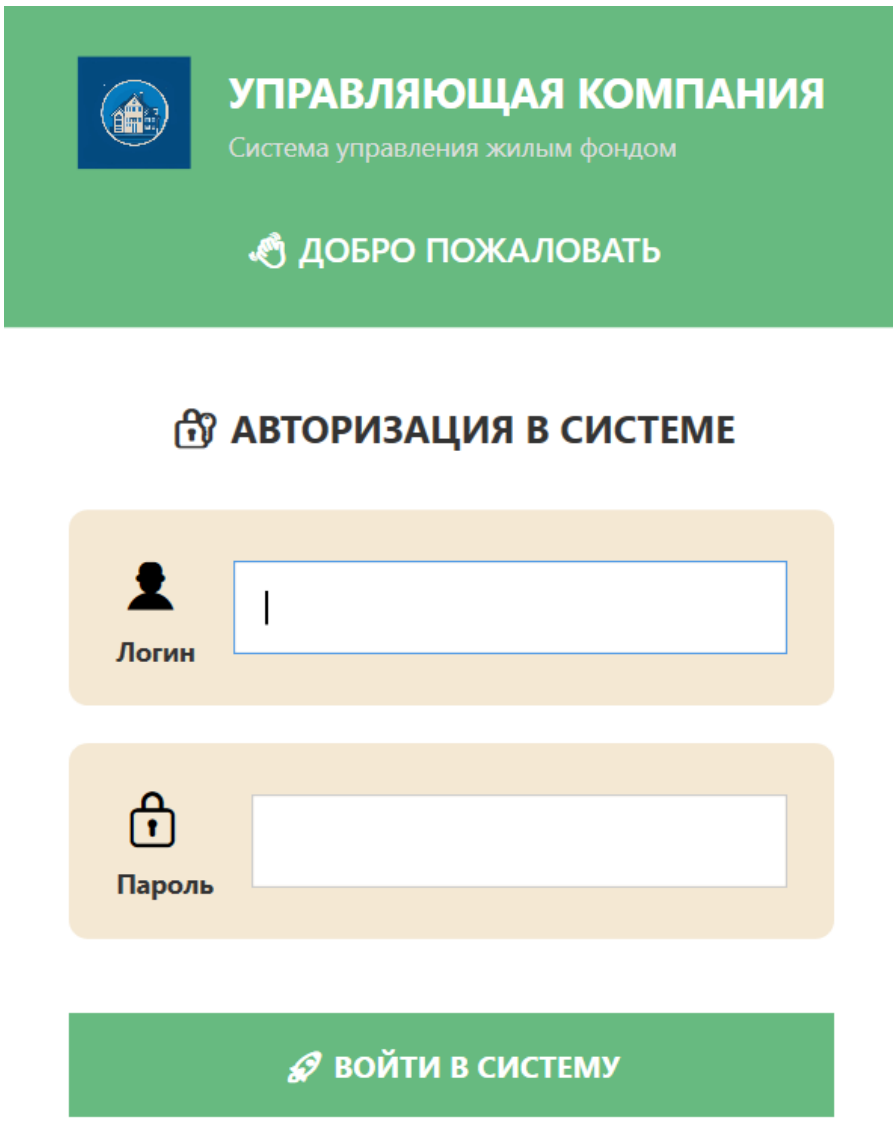

Рисунок 12 – Окно "Авторизация"

Ниже представлен код окна авторизации в листингах 18 – 19.

Листинг 18 – Разметка окна авторизации (LoginWindow.xaml)

```
<Window x:Class="HousingStock.LoginWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="Авторизация - Управляющая компания"
        Height="550"
        Width="450"
        WindowStartupLocation="CenterScreen"
        ResizeMode="NoResize"
        Icon="Resources/icon.ico">

    <Window.Resources>
        <ResourceDictionary Source="Styles.xaml"/>
    </Window.Resources>

    <Grid Background="{StaticResource PrimaryBrush}">
        <Grid.RowDefinitions>
            <RowDefinition Height="1.5*"/>
            <RowDefinition Height="2.5*"/>
        </Grid.RowDefinitions>
```

```
<!-- Верхний блок с логотипом -->
<Border Grid.Row="0"
        Background="{StaticResource SecondaryBrush}">
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="*"/>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
        </Grid.RowDefinitions>

        <StackPanel Grid.Row="1"
                    HorizontalAlignment="Center"
                    VerticalAlignment="Center">
            <Image Source="Resources/logo.png"
                   Height="80"
                   Width="80"
                   Margin="0,0,0,15"/>
            <TextBlock Text="Управляющая компания"
                       Style="{StaticResource HeaderTextBlock}"
                       HorizontalAlignment="Center"
                       Margin="0"/>
        </StackPanel>

        <TextBlock Grid.Row="2"
                   Text="Авторизация в системе"
                   FontSize="16"
                   HorizontalAlignment="Center"
                   Foreground="{StaticResource LightTextBrush}"
                   Margin="0,10,0,30"/>
    </Grid>
</Border>

<!-- Нижний блок с формой авторизации -->
<Border Grid.Row="1"
        Background="{StaticResource PrimaryBrush}"
        Margin="40,0,40,40"
        CornerRadius="8">
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
            <RowDefinition Height="Auto"/>
        </Grid.RowDefinitions>

        <!-- Отступ сверху -->
        <Rectangle Grid.Row="0" Height="20" Fill="Transparent"/>

        <!-- Логин -->
        <StackPanel Grid.Row="1" Margin="30,0,30,15">
            <Label Content="Логин"
                   Style="{StaticResource LabelStyle}"
                   Margin="0,0,0,8"/>
            <TextBox x:Name="LoginTextBox"
                     Style="{StaticResource TextBoxStyle}"
                     KeyDown="LoginTextBox_KeyDown"/>
        </StackPanel>

        <!-- Пароль -->
```

```
                <StackPanel Grid.Row="2" Margin="30,0,30,15">
                    <Label Content="Пароль"
                           Style="{StaticResource LabelStyle}"
                           Margin="0,0,0,8"/>
                    <PasswordBox x:Name="PasswordBox"
                                 Style="{StaticResource PasswordBoxStyle}"
                                 KeyDown="PasswordBox_KeyDown"/>
                </StackPanel>

                <!-- Сообщение об ошибке -->
                <Border Grid.Row="3"
                        Margin="30,5,30,15"
                        Visibility="Collapsed"
                        x:Name="ErrorBorder"
                        Background="#FFF5F5"
                        CornerRadius="4">
                    <TextBlock x:Name="ErrorText"
                               Text=""
                               Foreground="{StaticResource ErrorBrush}"
                               TextWrapping="Wrap"
                               FontSize="13"
                               Margin="10"/>
                </Border>

                <!-- Пробел вместо тестовых кнопок -->
                <Rectangle Grid.Row="4" Height="40" Fill="Transparent"/>

                <!-- Кнопка входа -->
                <Border Grid.Row="5"
                        VerticalAlignment="Bottom"
                        Margin="30,0,30,30">
                    <Button x:Name="LoginButton"
                            Content="Войти"
                            Style="{StaticResource PrimaryButton}"
                            Click="LoginButton_Click"
                            Height="45"
                            FontSize="15"
                            FontWeight="SemiBold"
                            HorizontalAlignment="Stretch"/>
                </Border>
            </Grid>
        </Border>
    </Grid>
</Window>
```

Листинг 19 – Код окна авторизации (LoginWindow.xaml.cs)

```
using System;
using System.Data.SqlClient;
using System.Windows;
using System.Windows.Input;

namespace HousingStock
{
    public partial class LoginWindow : Window
    {
        private string connectionString = "Data Source=(local);Initial
Catalog=Housing_stock;Integrated Security=True";

        public LoginWindow()
        {
            InitializeComponent();
            LoginTextBox.Focus();
        }
```

```
        private void LoginButton_Click(object sender, RoutedEventArgs e)
        {
            AuthenticateUser();
        }

        private void AuthenticateUser()
        {
            try
            {
                string login = LoginTextBox.Text.Trim();
                string password = PasswordBox.Password;

                if (string.IsNullOrEmpty(login))
                {
                    ShowError("Введите логин");
                    LoginTextBox.Focus();
                    return;
                }

                if (string.IsNullOrEmpty(password))
                {
                    ShowError("Введите пароль");
                    PasswordBox.Focus();
                    return;
                }

                LoginButton.Content = "Проверка...";
                LoginButton.IsEnabled = false;

                using (SqlConnection connection = new
SqlConnection(connectionString))
                {
                    connection.Open();

                    string query = @"
                        SELECT u.UserID, u.FullName, r.RoleName
                        FROM Users u
                        INNER JOIN Roles r ON u.RoleID = r.RoleID
                        WHERE u.Login = @Login AND u.Password = @Password";

                    using (SqlCommand command = new SqlCommand(query,
connection))
                    {
                        command.Parameters.AddWithValue("@Login", login);
                        command.Parameters.AddWithValue("@Password",
password);

                        using (SqlDataReader reader =
command.ExecuteReader())
                        {
                            if (reader.Read())
                            {
                                int userId =
Convert.ToInt32(reader["UserID"]);
                                string fullName =
reader["FullName"]?.ToString() ?? "Пользователь";
                                string roleName =
reader["RoleName"]?.ToString() ?? "Пользователь";

                                CurrentUser.Initialize(userId, fullName,
roleName);
```

```
                                OpenRoleSpecificWindow(userId, fullName,
roleName);
                                this.Close();
                            }
                            else
                            {
                                ShowError("Неверный логин или пароль");
                                PasswordBox.Password = "";
                                PasswordBox.Focus();
                            }
                        }
                    }
                }
            }
            catch (SqlException ex)
            {
                ShowError($"Ошибка подключения к базе данных: {ex.Message}");
            }
            catch (Exception ex)
            {
                ShowError($"Ошибка: {ex.Message}");
            }
            finally
            {
                LoginButton.Content = "Войти";
                LoginButton.IsEnabled = true;
            }
        }

        private void OpenRoleSpecificWindow(int userId, string fullName,
string roleName)
        {
            Window mainWindow;

            string cleanRoleName = roleName.Trim().ToLower();

            if (cleanRoleName == "administrator")
            {
                mainWindow = new AdminMainWindow();
                mainWindow.Title = $"Панель администратора - {fullName}";
            }
            else if (cleanRoleName == "руководитель" || cleanRoleName ==
"manager")
            {
                mainWindow = new ManagerMainWindow();
                mainWindow.Title = $"Панель руководителя - {fullName}";
            }
            else if (cleanRoleName == "employee")
            {
                mainWindow = new EmployeeMainWindow();
                mainWindow.Title = $"Панель сотрудника - {fullName}";
            }
            else if (cleanRoleName == "resident")
            {
                // Получаем RoleID для роли "resident"
                int roleId = GetRoleIdByRoleName(roleName);

                // Создаем объект пользователя для передачи в
ResidentMainWindow
                var user = new Users
                {
                    UserID = userId,
                    FullName = fullName,
```

```
                    RoleID = roleId
                };

                mainWindow = new ResidentMainWindow(user);
                mainWindow.Title = $"Панель жителя - {fullName}";
            }
            else
            {
                MessageBox.Show($"Роль '{roleName}' не распознана.
Используется общее окно.",
                    "Предупреждение", MessageBoxButton.OK,
MessageBoxImage.Warning);

                mainWindow = new MainWindow();
                mainWindow.Title = $"Управляющая компания - {fullName}";
            }

            mainWindow.Show();
        }

        ……

    public static class CurrentUser
    {
        public static int UserId { get; private set; }
        public static string FullName { get; private set; }
        public static string RoleName { get; private set; }
        public static bool IsAuthenticated { get; private set; }

        public static void Initialize(int userId, string fullName, string
roleName)
        {
            UserId = userId;
            FullName = fullName;
            RoleName = roleName;
            IsAuthenticated = true;
        }

        public static void Clear()
        {
            UserId = 0;
            FullName = string.Empty;
            RoleName = string.Empty;
            IsAuthenticated = false;
        }
    }
}
```

Модуль № 4. Осуществление интеграции программных модулей

На рисунке 13 представлена страница "История выполнения заявок". Эта страница позволяет просматривать историю всех заявок в системе с возможностью фильтрации по исполнителям и адресам. Реализованы три типа отчетов: Все заявки, По исполнителям, По адресам. Для фильтрации по исполнителям используется выпадающий список с именами сотрудников. Для фильтрации по адресам предусмотрено текстовое поле поиска.

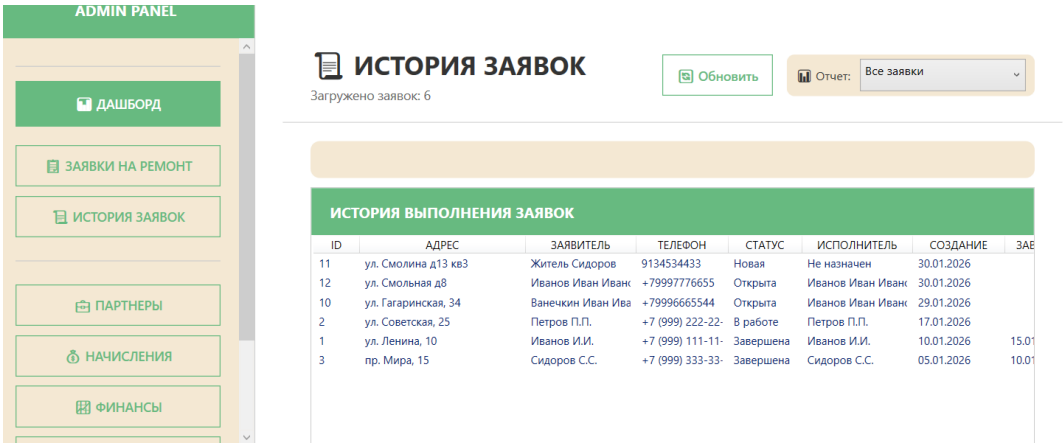

Рисунок 13 – Страница "История выполненных заявок"

Ниже представлен код страницы "История выполнения заявок" в листингах 20 – 21.

Листинг 20 – Разметка страницы История выполнения заявок (ApplicationHistoryPage.xaml)

```
<Page x:Class="HousingStock.ApplicationHistoryPage"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      Title="История выполнения заявок"
      Background="{StaticResource PrimaryBrush}">

    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
        </Grid.RowDefinitions>

        <!-- Заголовок -->
        <Border Grid.Row="0"
                Background="{StaticResource AccentBrush}"
                Padding="15">
            <TextBlock Text="История выполнения заявок"
                       FontSize="22"
                       FontWeight="Bold"
                       Foreground="White"
                       HorizontalAlignment="Center"/>
```

```
        </Border>

        <!-- Панель фильтров -->
        <Border Grid.Row="1"
                Background="{StaticResource SecondaryBrush}"
                Padding="10"
                Margin="0,0,0,10">
            <StackPanel>
                <!-- Выбор типа отчета -->
                <StackPanel Orientation="Horizontal" Margin="0,0,0,10">
                    <TextBlock Text="Тип отчета:"
                               VerticalAlignment="Center"
                               Margin="0,0,10,0"/>

                    <ComboBox x:Name="ReportTypeCombo"
                              Width="180"
                              Height="30"
                              SelectedIndex="0"
                              IsEditable="False"
                              IsTextSearchEnabled="True"

SelectionChanged="ReportTypeCombo_SelectionChanged">
                        <ComboBoxItem Content="Все заявки"/>
                        <ComboBoxItem Content="По исполнителям"/>
                        <ComboBoxItem Content="По адресам"/>
                    </ComboBox>
                </StackPanel>

                <!-- Поле поиска для исполнителей -->
                <StackPanel Orientation="Horizontal"
                            HorizontalAlignment="Left"
                            x:Name="EmployeeSearchPanel"
                            Visibility="Collapsed"
                            Margin="0,5,0,0">
                    <TextBlock Text="Исполнитель:"
                               VerticalAlignment="Center"
                               Margin="0,0,10,0"/>

                    <ComboBox x:Name="EmployeeCombo"
                              Width="200"
                              Height="30"
                              IsEditable="False"
                              IsTextSearchEnabled="True"/>

                    <Button x:Name="ShowEmployeeHistoryButton"
                            Content="Показать"
                            Style="{StaticResource PrimaryButton}"
                            Width="95"
                            Height="36"
                            Margin="10,0,0,0"
                            Click="ShowEmployeeHistoryButton_Click"/>
                </StackPanel>

                <!-- Поле поиска для адресов -->
                <StackPanel Orientation="Horizontal"
                            HorizontalAlignment="Left"
                            x:Name="AddressSearchPanel"
                            Visibility="Collapsed"
                            Margin="0,5,0,0">
                    <TextBlock Text="Адрес:"
                               VerticalAlignment="Center"
                               Margin="0,0,10,0"/>
```

```
                    <TextBox x:Name="AddressSearchBox"
                             Width="200"
                             Height="30"
                             Padding="5"
                             KeyDown="AddressSearchBox_KeyDown"/>

                    <Button x:Name="ShowAddressHistoryButton"
                             Content="Показать"
                             Style="{StaticResource PrimaryButton}"
                             Width="95"
                             Height="36"
                             Margin="10,0,0,0"
                             Click="ShowAddressHistoryButton_Click"/>
                </StackPanel>
            </StackPanel>
        </Border>

        <!-- Список истории -->
        <Grid Grid.Row="2" Margin="10">
            <Grid.RowDefinitions>
                <RowDefinition Height="*"/>
                <RowDefinition Height="Auto"/>
            </Grid.RowDefinitions>

            <!-- Основной список -->
            <ListView x:Name="HistoryList"
                      Grid.Row="0"
                      BorderThickness="1"
                      BorderBrush="{StaticResource BorderBrush}">
                <ListView.View>
                    <GridView>
                        <GridViewColumn Header="ID" Width="50"
DisplayMemberBinding="{Binding ApplicationID}"/>
                        <GridViewColumn Header="Адрес" Width="180"
DisplayMemberBinding="{Binding Address}"/>
                        <GridViewColumn Header="Заявитель" Width="120"
DisplayMemberBinding="{Binding ApplicantName}"/>
                        <GridViewColumn Header="Телефон" Width="100"
DisplayMemberBinding="{Binding Phone}"/>
                        <GridViewColumn Header="Статус" Width="80"
DisplayMemberBinding="{Binding Status}"/>
                        <GridViewColumn Header="Исполнитель" Width="120"
DisplayMemberBinding="{Binding EmployeeName}"/>
                        <GridViewColumn Header="Дата создания" Width="100"
                                        DisplayMemberBinding="{Binding
CreateDate, StringFormat='dd.MM.yyyy'}"/>
                        <GridViewColumn Header="Дата завершения" Width="100"
                                        DisplayMemberBinding="{Binding
CompleteDate, StringFormat='dd.MM.yyyy'}"/>
                        <GridViewColumn Header="Дней" Width="50"
DisplayMemberBinding="{Binding DaysToComplete}"/>
                    </GridView>
                </ListView.View>
            </ListView>

            <!-- Панель статуса с кнопкой обновить -->
            <Border Grid.Row="1"
                    Background="{StaticResource SecondaryBrush}"
                    Height="45"
                    Margin="0,10,0,0"
                    CornerRadius="5"
                    Padding="10">
                <Grid>
```

```
                    <Grid.ColumnDefinitions>
                        <ColumnDefinition Width="*"/>
                        <ColumnDefinition Width="Auto"/>
                    </Grid.ColumnDefinitions>

                    <TextBlock x:Name="StatusText"
                               Text="Загружено заявок: 0"
                               VerticalAlignment="Center"
                               FontSize="13"
                               Grid.Column="0"/>

                    <Button Content="🔄 Обновить"
                            Style="{StaticResource PrimaryButton}"
                            Width="120"
                            Height="35"
                            Grid.Column="1"
                            FontWeight="SemiBold"
                            Click="RefreshButton_Click"/>
                </Grid>
            </Border>
        </Grid>
    </Grid>
</Page>
```

Листинг 21 – Код страницы История выполнения заявок (ApplicationHistoryPage.xaml.cs)

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Input;

namespace HousingStock
{
    public partial class ApplicationHistoryPage : Page
    {

        private List<string> executors;

        public class ApplicationHistory
        {
            public int ApplicationID { get; set; }
            public string Address { get; set; }
            public string ApplicantName { get; set; }
            public string Phone { get; set; }
            public string Status { get; set; }
            public string EmployeeName { get; set; }
            public DateTime CreateDate { get; set; }
            public DateTime? CompleteDate { get; set; }
            public int? DaysToComplete { get; set; }
            public string Description { get; set; }
        }

        public ApplicationHistoryPage()
        {
            InitializeComponent();
            Loaded += ApplicationHistoryPage_Loaded;
        }
```

```
        private void ApplicationHistoryPage_Loaded(object sender,
RoutedEventArgs e)
        {
            LoadAllApplications();
            LoadExecutors();
        }

        private void LoadAllApplications()
        {
            try
            {
                List<ApplicationHistory> applications = new
List<ApplicationHistory>();

                using (SqlConnection connection = new
SqlConnection(connectionString))
                {
                    connection.Open();
                    string query = @"
                        SELECT
                            ID AS ApplicationID,
                            Address,
                            ApplicantName,
                            Phone,
                            Status,
                            AssignedEmployee AS EmployeeName,
                            CreateDate,
                            CompleteDate,
                            DATEDIFF(DAY, CreateDate, ISNULL(CompleteDate,
GETDATE())) AS DaysToComplete,
                            Description
                        FROM Applications
                        ORDER BY CreateDate DESC";

                    using (SqlCommand command = new SqlCommand(query,
connection))
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            ApplicationHistory app = new ApplicationHistory
                            {
                                ApplicationID =
reader.GetInt32(reader.GetOrdinal("ApplicationID")),
                                Address = reader["Address"].ToString(),
                                ApplicantName =
reader["ApplicantName"].ToString(),
                                Phone = reader["Phone"].ToString(),
                                Status = reader["Status"].ToString(),
                                CreateDate =
reader.GetDateTime(reader.GetOrdinal("CreateDate")),
                                Description =
reader["Description"].ToString()
                            };

                            // Обработка EmployeeName (может быть NULL)
                            int employeeNameOrdinal =
reader.GetOrdinal("EmployeeName");
                            if (!reader.IsDBNull(employeeNameOrdinal))
                            {
                                app.EmployeeName =
reader["EmployeeName"].ToString();
                            }
```

```
                            else
                            {
                                app.EmployeeName = "Не назначен";
                            }

                            // Обработка CompleteDate (может быть NULL)
                            int completeDateOrdinal =
reader.GetOrdinal("CompleteDate");
                            if (!reader.IsDBNull(completeDateOrdinal))
                            {
                                app.CompleteDate =
reader.GetDateTime(completeDateOrdinal);
                            }

                            // Обработка DaysToComplete (может быть NULL)
                            int daysOrdinal =
reader.GetOrdinal("DaysToComplete");
                            if (!reader.IsDBNull(daysOrdinal))
                            {
                                app.DaysToComplete =
reader.GetInt32(daysOrdinal);
                            }

                            applications.Add(app);
                        }
                    }
                }

                ……

                    case 2: // По адресам
                        if (AddressSearchBox != null &&
!string.IsNullOrWhiteSpace(AddressSearchBox.Text))
                        {

LoadAddressApplications(AddressSearchBox.Text.Trim());
                        }
                        else
                        {
                            LoadAllApplications();
                        }
                        break;

                    default:
                        LoadAllApplications();
                        break;
                }

                MessageBox.Show("Данные обновлены", "Обновление",
                    MessageBoxButton.OK, MessageBoxImage.Information);
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Ошибка обновления: {ex.Message}",
                    "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);
            }
        }
    }
}
```

На рисунке 14 представлена страница "Дашборд руководителя". Эта страница предназначена для руководителей компании и содержит ключевые

показатели эффективности: финансовые показатели, эффективность компании, информация о персонале, задачи. Отображаются: месячный доход, расходы, чистая прибыль, рентабельность, прибыль от ресурсов, затраты на ресурсы, рентабельность ресурсов.

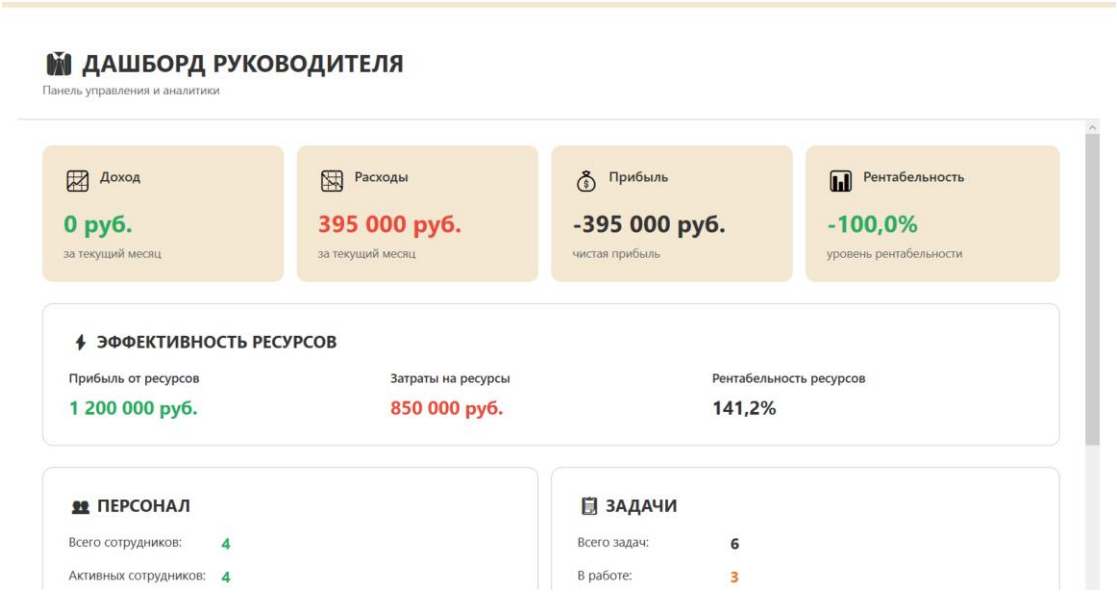

Рисунок 14 – Страница "Дашборд руководителя".

Ниже представлен код страницы "Дашборд руководителя" в листингах 22 – 23.

Листинг 22 – Разметка страницы Дашборд руководителя (ManagerDashboardPage.xaml)

```
<Page x:Class="HousingStock.ManagerDashboardPage"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      Title="Дашборд руководителя"
      Background="{StaticResource PrimaryBrush}">

    <Page.Resources>
        <ResourceDictionary Source="Styles.xaml"/>
    </Page.Resources>

    <Grid>
        <ScrollViewer VerticalScrollBarVisibility="Auto">
            <StackPanel Margin="20">

                <!-- Заголовок -->
                <TextBlock Text="Дашборд руководителя"
                           Style="{StaticResource HeaderTextBlock}"
                           Margin="0,0,0,20"/>

                <!-- Финансовые показатели -->
                <Border Background="{StaticResource SecondaryBrush}"
                        CornerRadius="8"
                        Padding="20"
                        Margin="0,0,0,20">
```

```
                    <StackPanel>
                        <TextBlock Text="Финансовые показатели"
                                   Style="{StaticResource SubheaderTextBlock}"
                                   Margin="0,0,0,15"/>

                        <Grid>
                            <Grid.ColumnDefinitions>
                                <ColumnDefinition Width="*"/>
                                <ColumnDefinition Width="*"/>
                                <ColumnDefinition Width="*"/>
                                <ColumnDefinition Width="*"/>
                            </Grid.ColumnDefinitions>

                            <!-- Месячный доход -->
                            <StackPanel Grid.Column="0" Margin="0,0,20,0">
                                <TextBlock Text="Месячный доход"
                                           Style="{StaticResource
LabelTextBlock}"/>
                                <TextBlock x:Name="MonthlyIncomeText"
                                           Text="0 руб."
                                           FontSize="24"
                                           FontWeight="Bold"
                                           FontFamily="Segoe UI"
                                           Foreground="{StaticResource
SuccessBrush}"/>
                            </StackPanel>

                            <!-- Месячные расходы -->
                            <StackPanel Grid.Column="1" Margin="0,0,20,0">
                                <TextBlock Text="Месячные расходы"
                                           Style="{StaticResource
LabelTextBlock}"/>
                                <TextBlock x:Name="MonthlyExpensesText"
                                           Text="0 руб."
                                           FontSize="24"
                                           FontWeight="Bold"
                                           FontFamily="Segoe UI"
                                           Foreground="{StaticResource
ErrorBrush}"/>
                            </StackPanel>

                            <!-- Чистая прибыль -->
                            <StackPanel Grid.Column="2" Margin="0,0,20,0">
                                <TextBlock Text="Чистая прибыль"
                                           Style="{StaticResource
LabelTextBlock}"/>
                                <TextBlock x:Name="NetProfitText"
                                           Text="0 руб."
                                           FontSize="24"
                                           FontWeight="Bold"
                                           FontFamily="Segoe UI"/>
                            </StackPanel>

                            <!-- Рентабельность -->
                            <StackPanel Grid.Column="3">
                                <TextBlock Text="Рентабельность"
                                           Style="{StaticResource
LabelTextBlock}"/>
                                <TextBlock x:Name="ProfitabilityText"
                                           Text="0%"
                                           FontSize="24"
                                           FontWeight="Bold"
                                           FontFamily="Segoe UI"
```

```
                                        Foreground="{StaticResource
SuccessBrush}"/>
                            </StackPanel>
                        </Grid>
                    </StackPanel>
                </Border>

                <!-- Эффективность -->
                <Border Background="{StaticResource SecondaryBrush}"
                        CornerRadius="8"
                        Padding="20"
                        Margin="0,0,0,20">
                    <StackPanel>
                        <TextBlock Text="Эффективность компании"
                                   Style="{StaticResource SubheaderTextBlock}"
                                   Margin="0,0,0,15"/>

                        <Grid>
                            <Grid.ColumnDefinitions>
                                <ColumnDefinition Width="*"/>
                                <ColumnDefinition Width="*"/>
                                <ColumnDefinition Width="*"/>
                            </Grid.ColumnDefinitions>

                            <!-- Прибыль от ресурсов -->
                            <StackPanel Grid.Column="0" Margin="0,0,20,0">
                                <TextBlock Text="Прибыль от ресурсов"
                                           Style="{StaticResource
LabelTextBlock}"/>
                                <TextBlock x:Name="ResourceProfitText"
                                           Text="0 руб."
                                           FontSize="20"
                                           FontWeight="Bold"
                                           FontFamily="Segoe UI"/>
                            </StackPanel>

                            <!-- Затраты на ресурсы -->
                            <StackPanel Grid.Column="1" Margin="0,0,20,0">
                                <TextBlock Text="Затраты на ресурсы"
                                           Style="{StaticResource
LabelTextBlock}"/>
                                <TextBlock x:Name="ResourceCostsText"
                                           Text="0 руб."
                                           FontSize="20"
                                           FontWeight="Bold"
                                           FontFamily="Segoe UI"/>
                            </StackPanel>

                            <!-- Рентабельность ресурсов -->
                            <StackPanel Grid.Column="2">
                                <TextBlock Text="Рентабельность ресурсов"
                                           Style="{StaticResource
LabelTextBlock}"/>
                                <TextBlock x:Name="ResourceProfitabilityText"
                                           Text="0%"
                                           FontSize="20"
                                           FontWeight="Bold"
                                           FontFamily="Segoe UI"/>
                            </StackPanel>
                        </Grid>
                    </StackPanel>
                </Border>
```

```
                <!-- Мониторинг -->
                <Border Background="{StaticResource SecondaryBrush}"
                        CornerRadius="8"
                        Padding="20"
                        Margin="0,0,0,20">
                    <StackPanel>
                        <Grid>
                            <Grid.ColumnDefinitions>
                                <ColumnDefinition Width="*"/>
                                <ColumnDefinition Width="*"/>
                            </Grid.ColumnDefinitions>

                            <!-- Персонал -->
                            <StackPanel Grid.Column="0" Margin="0,0,20,0">
                                <TextBlock Text="Персонал"
                                           Style="{StaticResource
SubheaderTextBlock}"
                                           Margin="0,0,0,10"/>

                                <StackPanel>
                                    <StackPanel Orientation="Horizontal"
Margin="0,0,0,5">
                                        <TextBlock Text="Всего сотрудников:"
Width="150"
                                                   Style="{StaticResource
LabelTextBlock}"/>
                                        <TextBlock x:Name="TotalStaffText"
                                                   Style="{StaticResource
BodyTextBlock}"
                                                   FontWeight="SemiBold"/>
                                    </StackPanel>

                                    <StackPanel Orientation="Horizontal"
Margin="0,0,0,5">
                                        <TextBlock Text="Активных:"
Width="150"
                                                   Style="{StaticResource
LabelTextBlock}"/>
                                        <TextBlock x:Name="ActiveStaffText"
                                                   Style="{StaticResource
BodyTextBlock}"
                                                   FontWeight="SemiBold"/>
                                    </StackPanel>

                                    <StackPanel Orientation="Horizontal">
                                        <TextBlock Text="Средний стаж:"
Width="150"
                                                   Style="{StaticResource
LabelTextBlock}"/>
                                        <TextBlock x:Name="AvgExperienceText"
                                                   Style="{StaticResource
BodyTextBlock}"
                                                   FontWeight="SemiBold"/>
                                    </StackPanel>
                                </StackPanel>
                            </StackPanel>

                            <!-- Задачи -->
                            <StackPanel Grid.Column="1">
                                <TextBlock Text="Задачи"
                                           Style="{StaticResource
SubheaderTextBlock}"
                                           Margin="0,0,0,10"/>
```

```
                                <StackPanel>
                                    <StackPanel Orientation="Horizontal"
Margin="0,0,0,5">
                                        <TextBlock Text="Всего задач:"
Width="150"
                                                   Style="{StaticResource
LabelTextBlock}"/>
                                        <TextBlock x:Name="TotalTasksText"
                                                   Style="{StaticResource
BodyTextBlock}"
                                                   FontWeight="SemiBold"/>
                                    </StackPanel>

                                    <StackPanel Orientation="Horizontal"
Margin="0,0,0,5">
                                        <TextBlock Text="В работе:"
Width="150"
                                                   Style="{StaticResource
LabelTextBlock}"/>
                                        <TextBlock
x:Name="InProgressTasksText"
                                                   Style="{StaticResource
BodyTextBlock}"
                                                   FontWeight="SemiBold"/>
                                    </StackPanel>

                                    <StackPanel Orientation="Horizontal">
                                        <TextBlock Text="Завершено:"
Width="150"
                                                   Style="{StaticResource
LabelTextBlock}"/>
                                        <TextBlock
x:Name="CompletedTasksText"
                                                   Style="{StaticResource
BodyTextBlock}"
                                                   FontWeight="SemiBold"/>
                                    </StackPanel>
                                </StackPanel>
                            </StackPanel>
                        </Grid>
                    </StackPanel>
                </Border>

                <!-- Действия -->
                <Border Background="{StaticResource SecondaryBrush}"
                        CornerRadius="8"
                        Padding="20">
                    <StackPanel>
                        <TextBlock Text="Быстрые действия"
                                   Style="{StaticResource SubheaderTextBlock}"
                                   Margin="0,0,0,15"/>

                        <Grid>
                            <Grid.ColumnDefinitions>
                                <ColumnDefinition Width="*"/>
                                <ColumnDefinition Width="*"/>
                                <ColumnDefinition Width="*"/>
                            </Grid.ColumnDefinitions>

                            <Button Grid.Column="0"
                                    Content="👥 Управление персоналом"
                                    Style="{StaticResource PrimaryButton}"
```

```
                                    Height="45"
                                    Margin="0,0,10,0"
                                    Click="ManageStaffButton_Click"/>

                            <Button Grid.Column="1"
                                    Content="📋 Планирование работ"
                                    Style="{StaticResource PrimaryButton}"
                                    Height="45"
                                    Margin="10,0,10,0"
                                    Click="PlanWorkButton_Click"/>

                            <Button Grid.Column="2"
                                    Content="📊 Аналитический отчет"
                                    Style="{StaticResource PrimaryButton}"
                                    Height="45"
                                    Margin="10,0,0,0"
                                    Click="AnalyticalReportButton_Click"/>
                        </Grid>
                    </StackPanel>
                </Border>

            </StackPanel>
        </ScrollViewer>
    </Grid>
</Page>
```

Листинг 23 – Код страницы Дашборд руководителя (ManagerDashboardPage.xaml.cs)

```
using System;
using System.Data.SqlClient;
using System.Windows;
using System.Windows.Controls;

namespace HousingStock
{
    public partial class ManagerDashboardPage : Page
    {
        public ManagerDashboardPage()
        {
            InitializeComponent();
            Loaded += ManagerDashboardPage_Loaded;
        }

        private void ManagerDashboardPage_Loaded(object sender,
RoutedEventArgs e)
        {
            LoadFinancialData();
            LoadStaffData();
            LoadTasksData();
            CalculateEfficiency();
        }

        private void LoadFinancialData()
        {
            try
            {
                using (SqlConnection connection = new
SqlConnection(connectionString))
                {
                    connection.Open();
```

```
                    // Месячный доход
                    string incomeQuery = @"
                        SELECT ISNULL(SUM([Paid for]), 0) as TotalIncome
                        FROM Payment
                        WHERE Period LIKE '%' + FORMAT(GETDATE(), 'MM.yyyy')
+ '%'";
        private void AnalyticalReportButton_Click(object sender,
RoutedEventArgs e)
        {
            try
            {
                string report = "АНАЛИТИЧЕСКИЙ ОТЧЕТ\n\n" +
                                $"Дата: {DateTime.Now:dd.MM.yyyy}\n" +
                                $"Пользователь: {CurrentUser.FullName}\n\n" +
                                "ФИНАНСОВЫЕ ПОКАЗАТЕЛИ:\n" +
                                $"• Месячный доход:
{MonthlyIncomeText.Text}\n" +
                                $"• Месячные расходы:
{MonthlyExpensesText.Text}\n" +
                                $"• Чистая прибыль: {NetProfitText.Text}\n" +
                                $"• Рентабельность:
{ProfitabilityText.Text}\n\n" +
                                "ЭФФЕКТИВНОСТЬ КОМПАНИИ:\n" +
                                $"• Прибыль от ресурсов:
{ResourceProfitText.Text}\n" +
                                $"• Затраты на ресурсы:
{ResourceCostsText.Text}\n" +
                                $"• Рентабельность ресурсов:
{ResourceProfitabilityText.Text}\n\n" +
                                "ПЕРСОНАЛ:\n" +
                                $"• Всего сотрудников:
{TotalStaffText.Text}\n" +
                                $"• Активных: {ActiveStaffText.Text}\n" +
                                $"• Средний стаж:
{AvgExperienceText.Text}\n\n" +
                                "ЗАДАЧИ:\n" +
                                $"• Всего задач: {TotalTasksText.Text}\n" +
                                $"• В работе: {InProgressTasksText.Text}\n" +
                                $"• Завершено: {CompletedTasksText.Text}\n\n"
+
                                "ВЫВОДЫ:\n" +
                                "• Показатели эффективности в пределах
нормы\n" +
                                "• Рекомендуется оптимизация расходов\n" +
                                "• Необходимо увеличить сбор платежей";

                MessageBox.Show(report, "Аналитический отчет",
                    MessageBoxButton.OK, MessageBoxImage.Information);
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Ошибка формирования отчета: {ex.Message}",
"Ошибка",
                    MessageBoxButton.OK, MessageBoxImage.Error);
            }
        }
    }
}
```

На рисунке 15 представлена страница "Мои заявки (житель)". Эта страница предназначена для жителей и отображает только заявки, созданные

текущим жителем. Реализована фильтрация по статусу (Все, Открыта, В работе, Завершена). Доступны функции: создание новой заявки, просмотр деталей заявки.

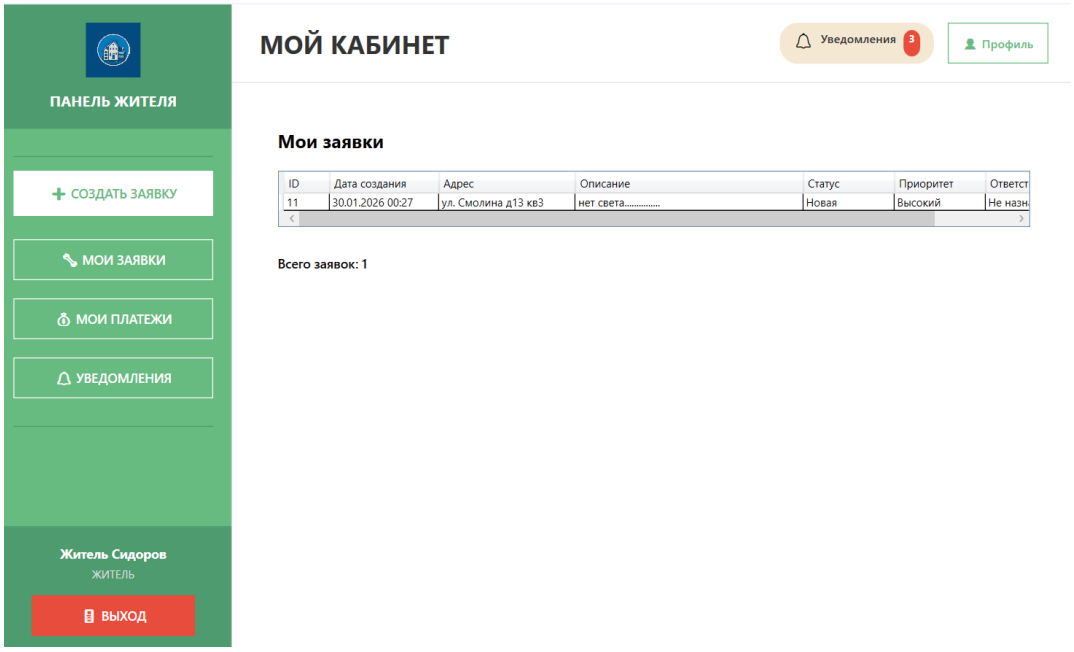

Рисунок 15 – Страница "Мои заявки (житель)"

Ниже представлен код страницы "Мои заявки (житель)" в листингах 24 – 25.

Листинг 24 – Разметка страницы Мои заявки жителя (ResidentApplicationsPage.xaml)

```
<Page x:Class="HousingStock.ResidentApplicationsPage"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      Title="Мои заявки"
      Background="White">

    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
            <RowDefinition Height="Auto"/>
        </Grid.RowDefinitions>

        <!-- Панель инструментов -->
        <Border Grid.Row="0"
                Background="#F4E8D3"
                Padding="10">
            <StackPanel Orientation="Horizontal">
                <Button x:Name="CreateButton"
                        Content="Создать заявку"
                        Width="140"
                        Height="35"
                        Margin="0,0,10,0"
                        Click="CreateButton_Click"
```

```
                    ToolTip="Создать новую заявку"/>

            <Button x:Name="RefreshButton"
                    Content="Обновить"
                    Width="100"
                    Height="35"
                    Margin="0,0,10,0"
                    Click="RefreshButton_Click"
                    ToolTip="Обновить список заявок"/>

            <Button x:Name="ViewDetailsButton"
                    Content="Просмотреть"
                    Width="120"
                    Height="35"
                    Margin="0,0,10,0"
                    Click="ViewDetailsButton_Click"
                    ToolTip="Просмотреть детали заявки"/>

            <TextBlock Text="Фильтр по статусу:"
                       VerticalAlignment="Center"
                       Margin="20,0,5,0"/>

            <ComboBox x:Name="StatusFilter"
                      Width="120"
                      Height="30"
                      SelectedIndex="0"
                      SelectionChanged="StatusFilter_SelectionChanged">
                <ComboBoxItem Content="Все"/>
                <ComboBoxItem Content="Открыта"/>
                <ComboBoxItem Content="В работе"/>
                <ComboBoxItem Content="Завершена"/>
            </ComboBox>
        </StackPanel>
    </Border>

    <!-- Список заявок -->
    <ListView x:Name="ApplicationsList"
              Grid.Row="1"
              Margin="10"
              SelectionMode="Single"
              MouseDoubleClick="ApplicationsList_MouseDoubleClick">
        <ListView.View>
            <GridView>
                <GridViewColumn Header="ID" Width="50"
DisplayMemberBinding="{Binding Id}"/>
                <GridViewColumn Header="Адрес" Width="200"
DisplayMemberBinding="{Binding Address}"/>
                <GridViewColumn Header="Категория" Width="180"
DisplayMemberBinding="{Binding CategoryName}"/>
                <GridViewColumn Header="Описание" Width="250"
DisplayMemberBinding="{Binding Description}"/>
                <GridViewColumn Header="Ответственный" Width="150"
DisplayMemberBinding="{Binding AssignedEmployee}"/>
                <GridViewColumn Header="Статус" Width="100"
DisplayMemberBinding="{Binding Status}"/>
                <GridViewColumn Header="Дата создания" Width="120"
                                DisplayMemberBinding="{Binding CreateDate,
StringFormat='dd.MM.yyyy HH:mm'}"/>
                <GridViewColumn Header="Дней" Width="50"
                                DisplayMemberBinding="{Binding
DaysInWork}"/>
            </GridView>
        </ListView.View>
```

```
        </ListView>

        <!-- Статус бар -->
        <Border Grid.Row="2"
                Background="#E0E0E0"
                Height="30">
            <TextBlock x:Name="StatusText"
                       Text="Загружено заявок: 0"
                       VerticalAlignment="Center"
                       Margin="10,0,0,0"/>
        </Border>
    </Grid>
</Page>
```

Листинг 25 – Код страницы Мои заявки жителя (ResidentApplicationsPage.xaml.cs)

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Windows;
using System.Windows.Controls;

namespace HousingStock
{
    public partial class ResidentApplicationsPage : Page
    {
        private List<ResidentApplication> applications;

        public class ResidentApplication
        {
            public int Id { get; set; }
            public string Address { get; set; }
            public string CategoryName { get; set; }
            public string Description { get; set; }
            public string AssignedEmployee { get; set; }
            public string Status { get; set; }
            public DateTime CreateDate { get; set; }
            public int? DaysInWork { get; set; }
        }

        public ResidentApplicationsPage()
        {
            InitializeComponent();
            Loaded += ResidentApplicationsPage_Loaded;
        }

        private void ResidentApplicationsPage_Loaded(object sender,
RoutedEventArgs e)
        {
            LoadResidentApplications();
        }

        private void LoadResidentApplications()
        {
            try
            {
                applications = new List<ResidentApplication>();

                // Получаем имя текущего пользователя
                string currentUserName = CurrentUser.FullName;
```

```
                ……

                    MessageBox.Show(details, "Детали заявки",
                        MessageBoxButton.OK, MessageBoxImage.Information);
                }
                else
                {
                    MessageBox.Show("Выберите заявку для просмотра деталей",
                        "Информация", MessageBoxButton.OK,
MessageBoxImage.Information);
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Ошибка: {ex.Message}",
                    "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);
            }
        }

        private void StatusFilter_SelectionChanged(object sender,
SelectionChangedEventArgs e)
        {
            UpdateApplicationsDisplay();
        }
    }
}
```

Модуль № 5. Вариативная часть

С целью минимизации риска потери данных и восстановления работоспособности системы в кратчайшие сроки в случае аварийных ситуаций сделаем резервное копирование базы данных.

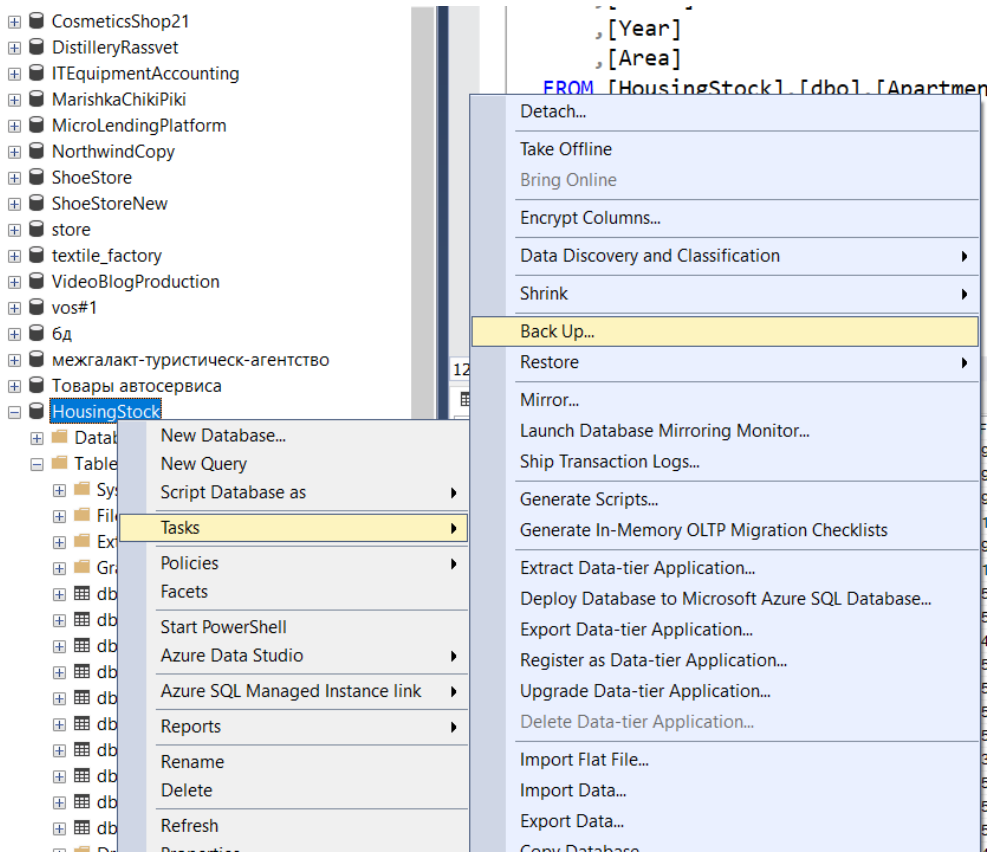

Рисунок 16 – Создаем резервную копию

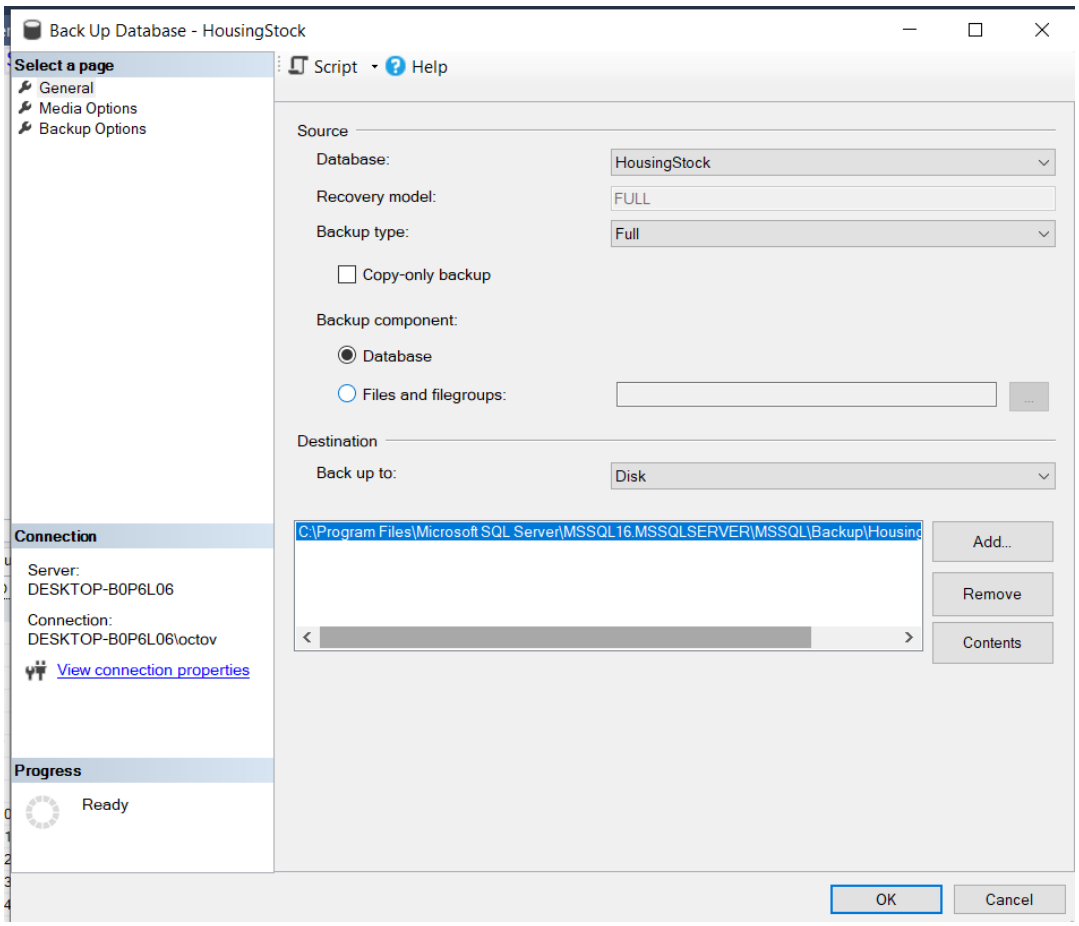

Рисунок 17 – Заполняем данными

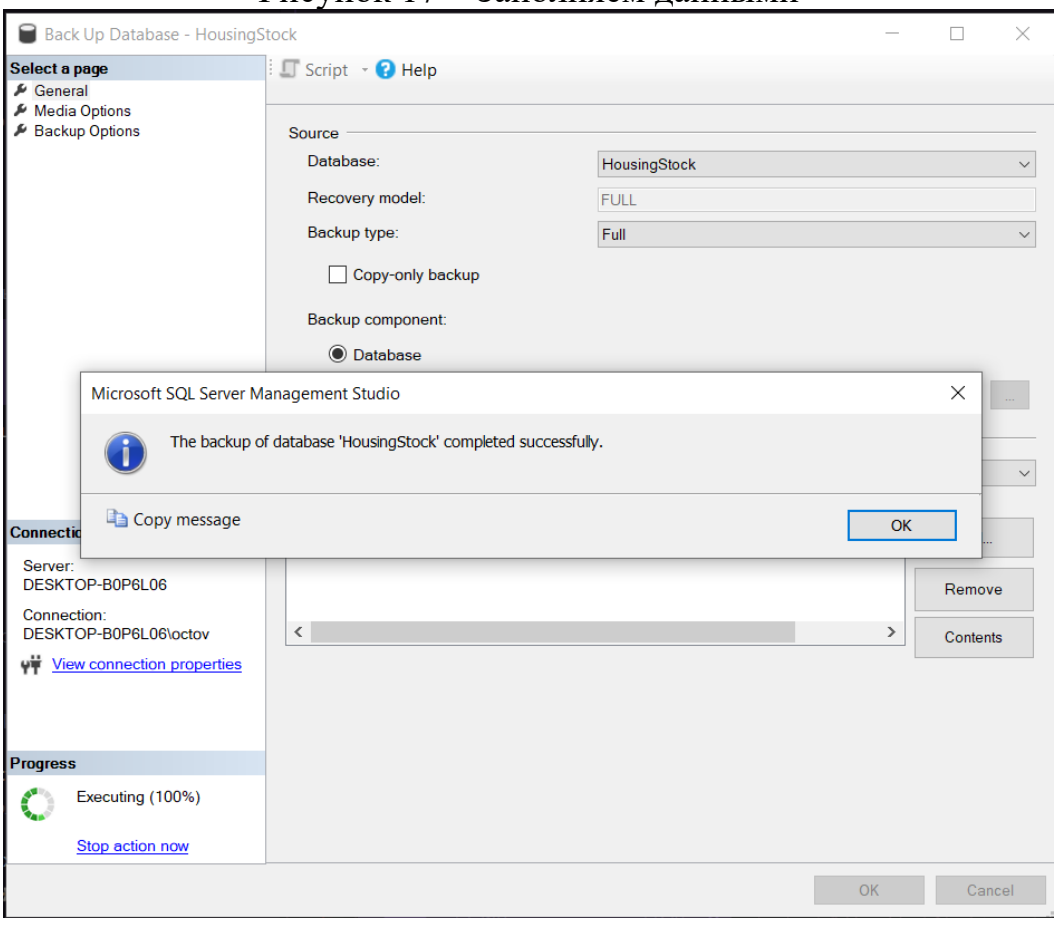

Рисунок 18 – Получаем сообщение об успешном резервном копировании БД

Далее проверяем целостность резервной копии путём тестового восстановления базы данных.

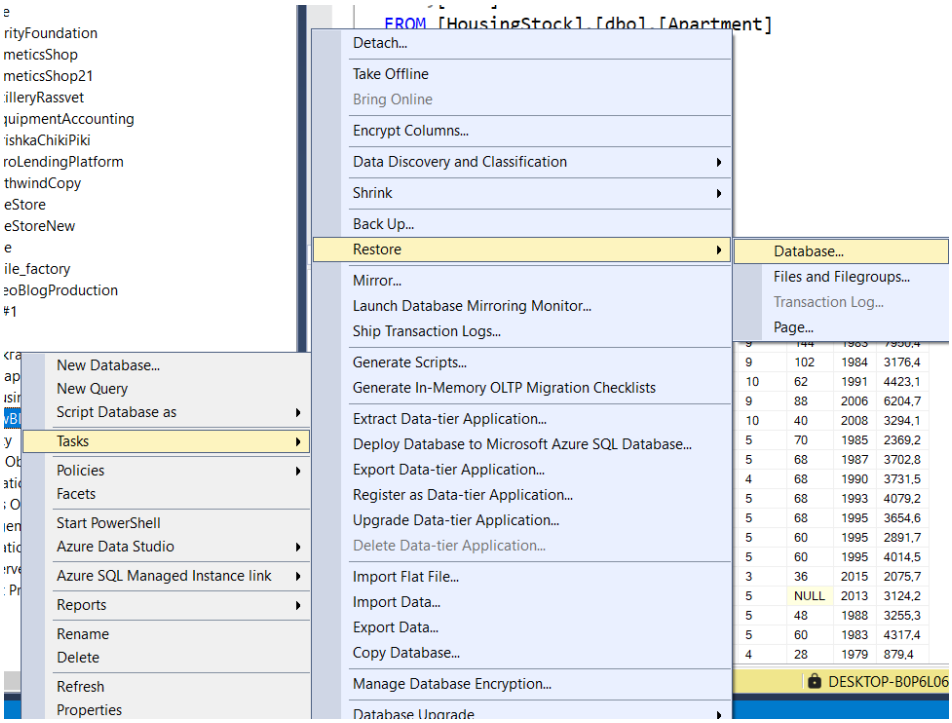

Рисунок 19 – Восстановление базы данных

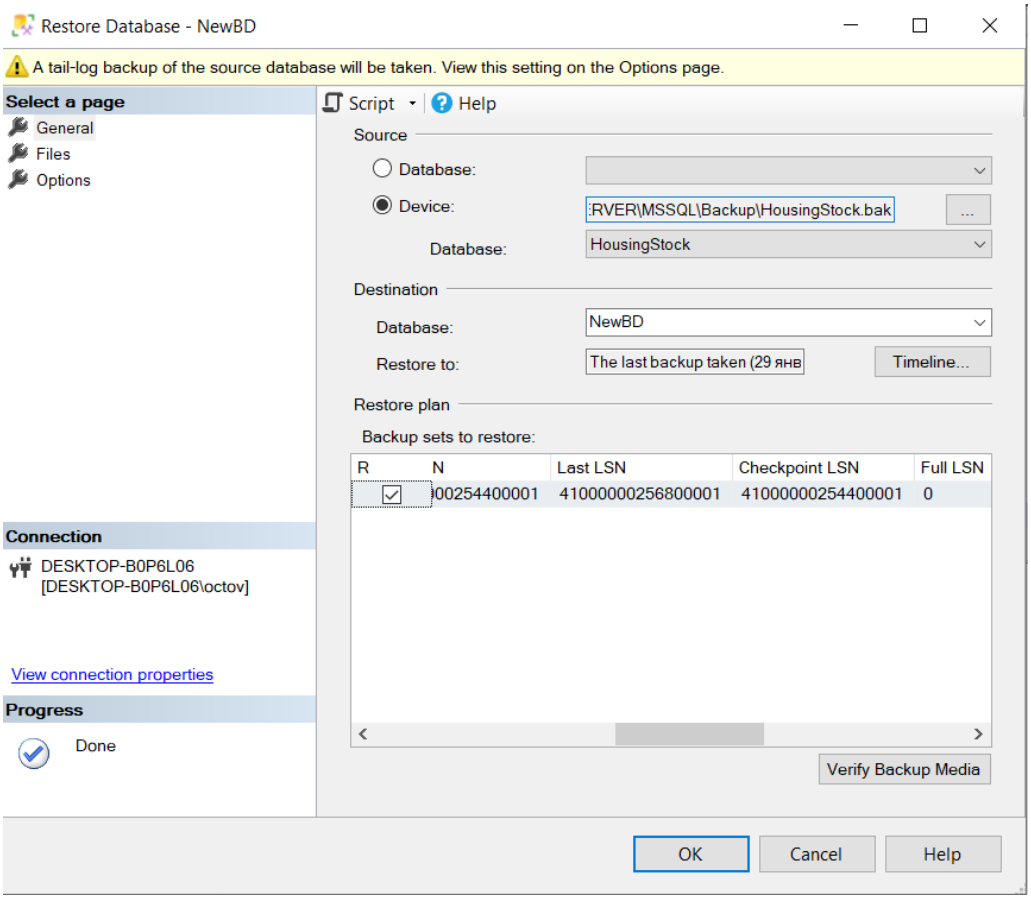

Рисунок 20 – Заполняем данные

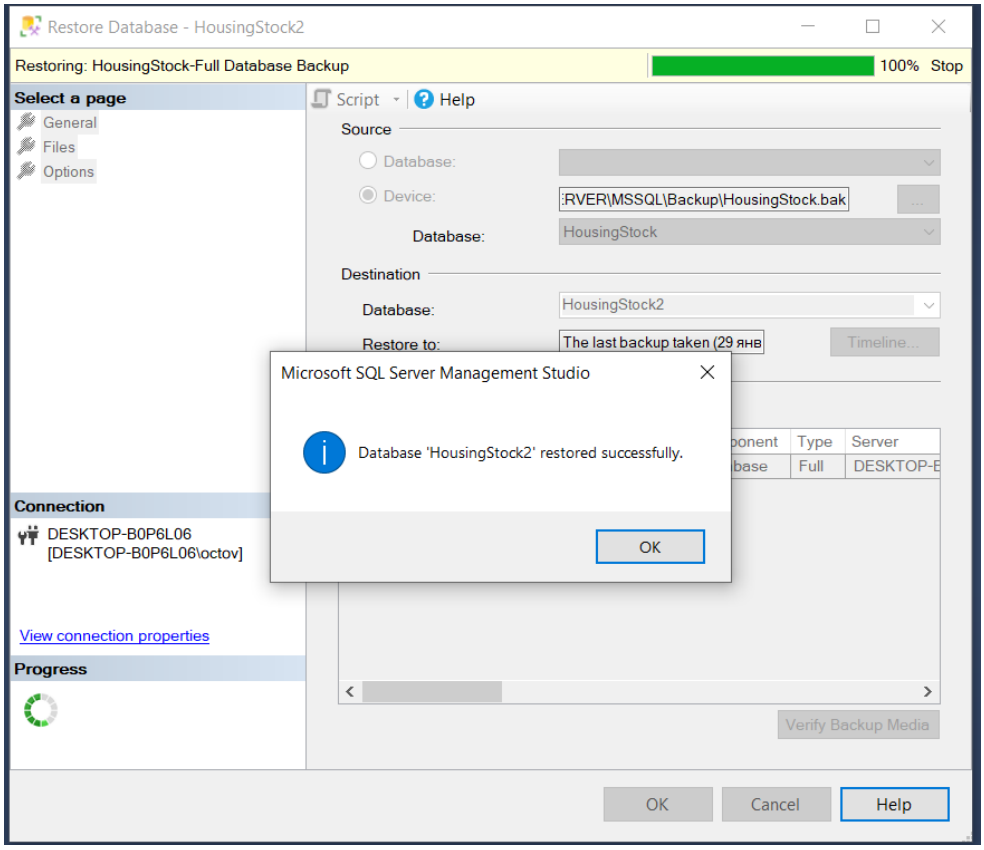

Рисунок 21 – Получаем сообщение об успешном восстановлении БД

После анализа требований к системе была построена диаграмма прецедентов (Use Case) для основных пользователей системы. Диаграмма прецедентов представлена на рисунке 23.

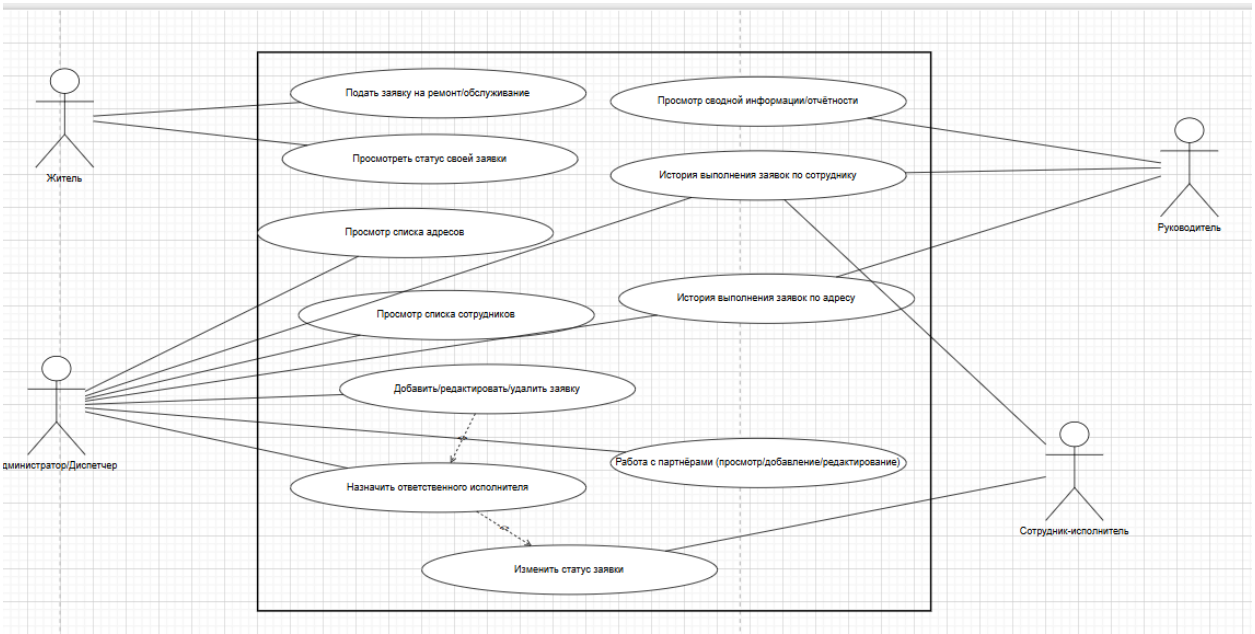

Рисунок 22 – Диаграмма прецедентов

В таблице 1 представлен тестовый сценарий для роли Руководитель.

Таблица 1 – Тестовый сценарий для роли Руководитель

Название проекта	Система управления жилым фондом для Управляющей компании (УК)
Рабочая версия	1.0.
Имя тестирующего	Виолетта
Дата(ы) теста	30.01.2026
Приоритет тестирования	Высокий
Заголовок/название теста	Анализ ключевого показателя эффективности (KPI) «Рентабельность использования ресурсов»
Краткое изложение теста	Проверка формирования и корректности расчета специфического финансового показателя для оценки эффективности работы компании.
Этапы теста	1. Руководитель авторизуется в системе. 2. Переходит в раздел «Аналитика» или «Финансовые показатели». 3. Выбирает отчет «Рентабельность использования ресурсов». 4. Задает период для анализа (например, предыдущий квартал). 5. Просматривает сформированную таблицу и график.
Тестовые данные	Учетные данные руководителя. Финансовые данные за выбранный период: доходы от услуг, затраты на ФОТ, материалы, подрядчиков.
Ожидаемый результат	1. Система отображает расчет: - Прибыль от использования ресурсов = [Доходы от услуг управления]. - Суммарные затраты на ресурсы = [ФОТ + Материалы + Амортизация + и т.д.]. - Рентабельность = (Прибыль / Затраты) * 100%. 2. Предоставляется возможность сравнения с предыдущими периодами в виде графика. 3. Данные в отчете консистентны с данными в основных финансовых отчетах (ОПиУ). 4. Руководитель НЕ может редактировать исходные финансовые данные из этого отчета.
Фактический результат	Система отобразила расчет, предоставила возможность сравнения, руководительне может редактировать, как и написано в задании
Постусловие	Руководитель получил данные для принятия управленческих решений.
Статус	Зачет
Примечания/комментарии	Критически важна точность агрегации финансовых данных из разных модулей (начисления, зарплата, закупки).

В таблице 2 представлен тестовый сценарий для ролей Руководитель и Администратор.

Таблица 2 – Тестовый сценарий для ролей Руководитель и Администратор

Название проекта	Система управления жилым фондом для Управляющей компании (УК)
Рабочая версия	1.0.
Имя тестирующего	Виолетта
Дата(ы) теста	30.01.2026
Приоритет тестирования	Высокий
Заголовок/название теста	Просмотр детальной аналитики по заявкам: среднее время закрытия, рейтинг жильцов
Краткое изложение теста	Проверка формирования статистики и отчетности по выполненным заявкам на ремонт для оценки качества работы и загрузки персонала.
Этапы теста	1. Пользователь переходит в раздел «Отчеты» -> «Аналитика по заявкам». 2. Задает период, фильтры (дом, тип неисправности, исполнитель). 3. Просматривает сформированные диаграммы и таблицы.
Тестовые данные	Учетные данные руководителя или администратора. История закрытых заявок за период.
Ожидаемый результат	Система отображает: 1. Количество заявок по статусам и категориям. 2. Среднее время выполнения заявки (от создания до закрытия). 3. Рейтинг/удовлетворенность жильцов на основе оценок (если функционал есть). 4. ТОП частых проблем по дому/подъезду. 5. Загрузку по исполнителям (количество выполненных заявок).
Фактический результат	Заявки можно сортировать, и просматривать исполнителей
Постусловие	Пользователь получил аналитическую сводку для оценки эффективности сервисной службы.
Статус	Зачет
Примечания/комментарии	Проверка корректности расчетов среднего времени и агрегации данных по фильтрам.

В таблице 3 представлен тестовый сценарий для роли Жилец.

Таблица 3 – Тестовый сценарий для роли Жилец

Название проекта	Система управления жилым фондом для Управляющей компании (УК)
Рабочая версия	1.0.
Имя тестирующего	Виолетта
Дата(ы) теста	30.01.2026
Приоритет тестирования	Высокий
Заголовок/название теста	Подача заявки на ремонт и отслеживание ее статуса
Краткое изложение теста	Проверка основного пользовательского сценария взаимодействия жильца с УК через мобильное приложение.
Этапы теста	1. Жилец авторизуется в мобильном приложении. 2. На главном экране нажимает «Подать заявку». 3. Выбирает тип заявки «Ремонт», категорию (сантехника), вводит описание проблемы. 4. Прикрепляет фотографию неисправности (опционально). 5. Отправляет заявку. 6. В разделе «Мои заявки» отслеживает смену статуса с «Новая» на «В работе», затем на «Выполнена».
Тестовые данные	Учетные данные жильца. Данные тестовой квартиры.
Ожидаемый результат	1. Заявка успешно создана и отображается в списке у жильца со статусом «Новая». 2. Жилец получает push-уведомление или видит изменение статуса в приложении при его смене администратором/исполнителем. 3. В статусе «Выполнена» есть возможность оставить комментарий или оценку работы. 4. Жилец НЕ может редактировать или удалять заявки после отправки, кроме статуса «Новая». 5. Жилец видит ТОЛЬКО свои заявки.
Фактический результат	Система успешно отобразила список жильца, жилец видит только свои заявки.
Постусловие	Заявка создана и прошла полный жизненный цикл.
Статус	Зачет
Примечания/комментарии	Необходимо проверить работу уведомлений и корректность отображения статусов, синхронизированных с веб-версией для администратора.

В таблице 4 представлен тестовый сценарий просмотра сотрудников и адресов

Таблица 4 – Тестовый сценарий просмотра сотрудников и адресов

Тестовый пример #	ТС_ПИ_2
Приоритет тестирования	Высокий
Заголовок/название теста	Просмотр адресов и сотрудников и возврат на главную
Краткое изложение теста	Проверить открытие окна адресов/сотрудников и корректную загрузку данных.
Этапы теста	1. На главном окне нажать «Адреса и сотрудники». 2. Убедиться, что отображаются таблицы адресов и сотрудников. 3. Нажать «Назад».
Тестовые данные	Не требуются
Ожидаемый результат	Открывается окно «Адреса и сотрудники». Вкладки/таблицы заполнены, внизу статус о количестве. Кнопка «Назад» возвращает на главное окно.
Фактический результат	Открывается окно «Адреса и сотрудники». Вкладки/таблицы заполнены, внизу статус о количестве. Кнопка «Назад» возвращает на главное окно.
Статус	Зачёт
Предварительное условие	БД доступна
Постусловие	Пользователь на главном окне
Примечания/комментарии	-

В таблице 5 представлен тестовый сценарий для роли Администратор.

Таблица 5 – Тестовый сценарий для роли Администратор

Название проекта	Система управления жилым фондом для Управляющей компании (УК)
Рабочая версия	1.0.
Имя тестирующего	Виолетта
Дата(ы) теста	30.01.2026
Приоритет тестирования	Высокий
Заголовок/название теста	Создание и управление графиком планового обслуживания домов
Краткое изложение теста	Проверка функции планирования регулярных работ (уборка территории, проверка систем) на месяц вперед с назначением ответственных.
Этапы теста	1. Администратор переходит в «График обслуживания». 2. Выбирает дом, период (месяц), тип работы «Уборка придомовой территории». 3. Добавляет повторяющееся событие (каждую неделю в понедельник). 4. Назначает ответственного сотрудника из списка дворников. 5. Сохраняет график. 6. В конце месяца отмечает выполненные работы и формирует отчет.
Тестовые данные	Учетные данные администратора. Список домов. Список сотрудников (персонал).
Ожидаемый результат	1. График создан, события отображаются в календаре системы. 2. Назначенный сотрудник видит задачу в своем личном кабинете/мобильном приложении в назначенный день. 3. Администратор может отслеживать статус выполнения (Запланировано/В работе/Выполнено). 4. На основе выполненных работ система автоматически формирует отчет «Выполненные работы за период» с фильтром по дому и типу работы. 5. Невыполненные работы выделяются в отчете.
Фактический результат	Назначенный сотрудник видит свою задачу, отчеты автоматически формируются
Постусловие	График создан, работы распределены, отчет сформирован.
Статус	Зачет
Примечания/комментарии	Важно проверить накладку задач на одного сотрудника и работу уведомлений для персонала.