**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.01.02 Поддержка и тестирование программных модулей

(код) (наименование)

Выполнил:

Студент ­4 курса 4ИСИП-421 учебной группы

(номер) (номер)

\_\_\_\_\_\_\_\_\_\_ Н.Д. Сакович

(подпись) (инициалы, фамилия)

Проверил:

Руководитель практики от Колледжа информатики и программирования

Преподаватель ВКК, к.п.н. Е.Л. Альшакова (квалификационная (инициалы, фамилия)

категория или звание,

должность)

Москва – 20­24

**Перечень работ, выполненных в ходе учебной практики**

|  |  |  |
| --- | --- | --- |
| **№ п/п** | **Виды работ** | **Оценка** |
| 1 | Проектирование базы данных и разработка объектов базы данных |  |
| 2 | Разработка модулей программного обеспечения для компьютерных систем |  |

**СОДЕРЖАНИЕ**

[**ОТЧЕТ О ВЫПОЛНЕНИИ ЗАДАНИЯ № 1** 4](#_Toc185523662)

[**ОТЧЕТ О ВЫПОЛНЕНИИ ЗАДАНИЯ № 2** 9](#_Toc185523663)

Федеральное государственное образовательное бюджетное

учреждение высшего образования

**«Финансовый университет при Правительстве Российской Федерации»**

**(Финансовый университет)**

Колледж информатики и программирования

**ОТЧЕТ О ВЫПОЛНЕНИИ ЗАДАНИЯ № 1**

**Тема:** **Проектирование базы данных и разработка объектов базы данных**

Студент: Сакович Н.Д.

Группа: 4ИСИП-421

Преподаватель: Альшакова Е.Л.

Дата: 19.12.2024

**ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ И РАЗРАБОТКА ОБЪЕКТОВ БАЗЫ**

**Цель работы:** разработать базу данных в соответствии с заданием, наполнить ее сущностями и атрибутами, сделать ER-диаграмму.

На рисунке 1 представлена ER-диаграмма БД RepairRequestsDB.

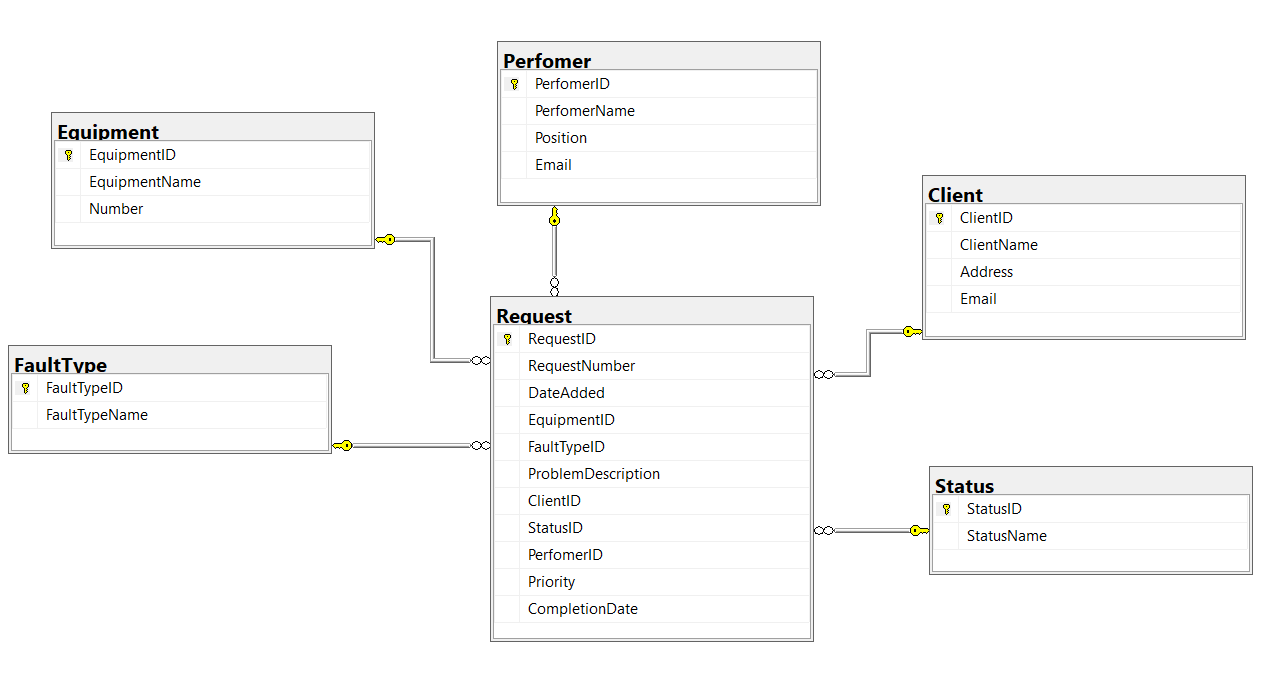


Рисунок 1 – ER-диаграмма

На рисунках 2-7 представлено заполнение таблиц базы данных.

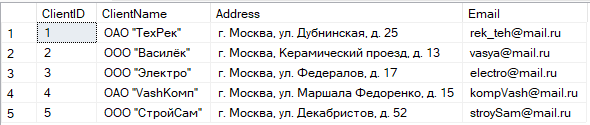


Рисунок 2 – Таблица Client

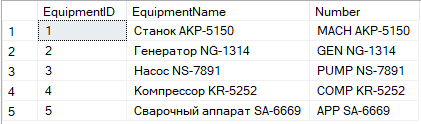


Рисунок 3 – Таблица Equipment

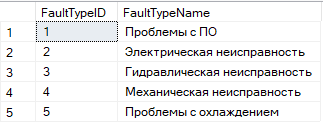


Рисунок 4 – Таблица FlautType

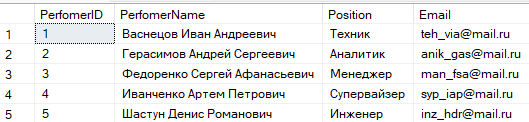


Рисунок 5 – Таблица Perfomer

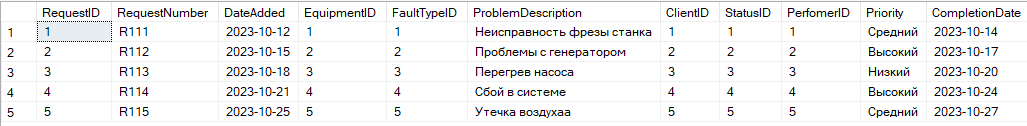


Рисунок 6 – Таблица Request

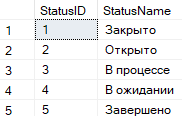


Рисунок 7 – Таблица Status

В листинге 1 представлен скрипт БД.

Листинг 1 – скрипт БД

|  |
| --- |
| -- Создание базы данных  CREATE DATABASE RepairRequestsDB;  GO  USE RepairRequestsDB;  GO  -- Таблица Users (Пользователи)  CREATE TABLE Users (  ID INT IDENTITY(1,1) PRIMARY KEY, -- Идентификатор пользователя  FullName NVARCHAR(150) NOT NULL, -- ФИО пользователя  Role NVARCHAR(50) NOT NULL -- Роль: Администратор, Менеджер, Исполнитель  );  -- Таблица Equipment (Оборудование)  CREATE TABLE Equipment (  ID INT IDENTITY(1,1) PRIMARY KEY, -- Идентификатор оборудования  Name NVARCHAR(100) NOT NULL, -- Название оборудования  SerialNumber NVARCHAR(100) NOT NULL -- Серийный номер оборудования  );  -- Таблица Requests (Заявки)  CREATE TABLE Requests (  ID INT IDENTITY(1,1) PRIMARY KEY, -- Идентификатор заявки  RequestNumber NVARCHAR(50) NOT NULL, -- Номер заявки  DateAdded DATETIME NOT NULL, -- Дата добавления заявки  EquipmentID INT NOT NULL FOREIGN KEY REFERENCES Equipment(ID), -- ID оборудования  FaultType NVARCHAR(100) NOT NULL, -- Тип неисправности  Description NVARCHAR(MAX), -- Описание проблемы  ClientID INT NOT NULL FOREIGN KEY REFERENCES Users(ID), -- Клиент, подающий заявку  Status NVARCHAR(50) NOT NULL -- Статус заявки: "В ожидании", "В работе", "Выполнено"  );  -- Таблица Repairs (Исполнения заявок)  CREATE TABLE Repairs (  ID INT IDENTITY(1,1) PRIMARY KEY, -- Идентификатор ремонта  RequestID INT NOT NULL FOREIGN KEY REFERENCES Requests(ID), -- Заявка на ремонт  ExecutorID INT NOT NULL FOREIGN KEY REFERENCES Users(ID), -- Исполнитель  StartDate DATETIME NOT NULL, -- Дата начала ремонта  EndDate DATETIME, -- Дата окончания ремонта  Comments NVARCHAR(MAX) -- Комментарии исполнителя  );  -- Таблица Feedback (Отзывы)  CREATE TABLE Feedback (  ID INT IDENTITY(1,1) PRIMARY KEY, -- Идентификатор отзыва  RequestID INT NOT NULL FOREIGN KEY REFERENCES Requests(ID), -- Заявка  Rating INT NOT NULL CHECK (Rating >= 1 AND Rating <= 5), -- Оценка работы  FeedbackText NVARCHAR(MAX) -- Текст отзыва  );  ALTER TABLE Users ADD Login NVARCHAR(100) NOT NULL UNIQUE;  ALTER TABLE Users ADD PasswordHash NVARCHAR(256) NOT NULL;  -- Заполнение таблицы Users  INSERT INTO Users (FullName, Role, Login, PasswordHash)  VALUES  ('Администратор Иванов', 'Администратор', 'admin', '40BD001563085FC35165329EA1FF5C5ECBDBBEEF'), -- пароль: 123  ('Менеджер Петров', 'Менеджер', 'manager', '40BD001563085FC35165329EA1FF5C5ECBDBBEEF'), -- пароль: admin  ('Исполнитель Сидоров', 'Исполнитель', 'executor', 'A94A8FE5CCB19BA61C4C0873D391E987982FBBD3'), -- пароль: test  ('Клиент Алексей Алексеев', 'Клиент', 'alexey', 'E99A18C428CB38D5F260853678922E03AB869F48'), -- пароль: password  ('Клиент Мария Марьянова', 'Клиент', 'maria', 'B1B3773A05C0ED0176787A4F1574FF0075F7521E'); -- пароль: welcome  -- Заполнение таблицы Equipment  INSERT INTO Equipment (Name, SerialNumber)  VALUES  ('Ноутбук ASUS', 'ASUS12345'),  ('Принтер Canon', 'CANON56789'),  ('Смартфон Samsung', 'SAMSUNG11223'),  ('Сервер HP', 'HP445566'),  ('Кофемашина Bosch', 'BOSCH99887');  -- Заполнение таблицы Requests  INSERT INTO Requests (RequestNumber, DateAdded, EquipmentID, FaultType, Description, ClientID, Status)  VALUES  ('REQ-001', GETDATE(), 1, 'Не включается', 'Ноутбук не реагирует на кнопку питания', 4, 'В ожидании'),  ('REQ-002', GETDATE(), 2, 'Не печатает', 'Принтер не выводит документы на печать', 5, 'В работе'),  ('REQ-003', GETDATE(), 3, 'Не заряжается', 'Смартфон не заряжается от зарядного устройства', 4, 'В ожидании'),  ('REQ-004', GETDATE(), 4, 'Перезагрузки', 'Сервер постоянно перезагружается', 5, 'В работе'),  ('REQ-005', GETDATE(), 5, 'Шумит', 'Кофемашина издает громкий шум при работе', 4, 'Выполнено');  -- Заполнение таблицы Repairs  INSERT INTO Repairs (RequestID, ExecutorID, StartDate, EndDate, Comments)  VALUES  (1, 3, GETDATE(), NULL, 'Диагностика начата, требуется замена платы'),  (2, 3, GETDATE(), NULL, 'Неисправен картридж, заказал новый'),  (3, 3, GETDATE(), NULL, 'Диагностика начата, требуется замена аккумулятора'),  (4, 3, GETDATE(), NULL, 'Проблема с блоком питания, требуется замена'),  (5, 3, GETDATE(), GETDATE(), 'Проблема устранена, заменён насос');  -- Заполнение таблицы Feedback  INSERT INTO Feedback (RequestID, Rating, FeedbackText)  VALUES  (5, 5, 'Отличная работа, проблема устранена быстро'),  (2, 4, 'Принтер починили, но пришлось долго ждать'),  (1, 3, 'Жду завершения ремонта'),  (4, 2, 'Сервер не починили, сроки нарушены'),  (3, 1, 'Не заряжается до сих пор, прошу ускорить ремонт'); |

Федеральное государственное образовательное бюджетное

учреждение высшего образования

**«Финансовый университет при Правительстве Российской Федерации»**

**(Финансовый университет)**

Колледж информатики и программирования

# **ОТЧЕТ О ВЫПОЛНЕНИИ ЗАДАНИЯ № 2**

**Тема: Разработка модулей программного обеспечения для компьютерных систем**

Студент: Сакович Н.Д.

Группа: 4ИСИП-421

Преподаватель: Альшакова Е.Л.

Дата: 19.12.2024

**РАЗРАБОТКА МОДУЛЕЙ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ КОМПЬЮТЕРНЫХ СИСТЕМ**

**Цель работы:** разработать WPF-приложение в соответствии с заданием, наполнить ее сущностями и атрибутами, написать листинг программ.

Для реализации графического интерфейса управления базой данных было создано WPF-приложение. Окно авторизации приложения представлено на рисунке 8.

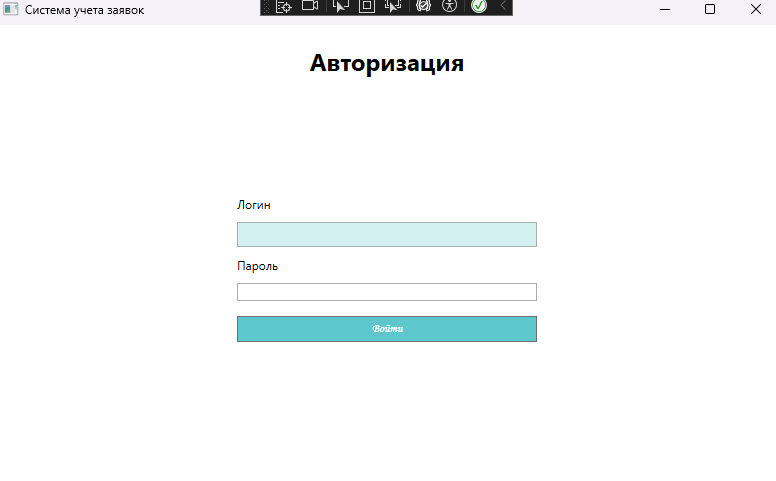


Рисунок 8 – Окно авторизации

В листинге 2 представлен код страницы AuthPage.xaml.

Листинг 2 – код MainWindow.xaml

|  |
| --- |
| <Page x:Class="WpfApp3.pages.AuthPage"  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"  xmlns:local="clr-namespace:WpfApp3.pages"  mc:Ignorable="d"  d:DesignHeight="450" d:DesignWidth="800"  Title="AuthPage">  <Grid Background="{StaticResource MainBackground}" Margin="10">  <Grid.RowDefinitions>  <RowDefinition Height="Auto" />  <RowDefinition Height="\*" />  </Grid.RowDefinitions>  <!-- Заголовок -->  <TextBlock Text="Авторизация" FontSize="24" FontWeight="Bold" HorizontalAlignment="Center" />  <!-- Форма авторизации -->  <StackPanel Grid.Row="1" HorizontalAlignment="Center" VerticalAlignment="Center" Width="300">  <!-- Логин -->  <TextBlock Text="Логин" Margin="0,5" />  <TextBox x:Name="TextBoxLogin" Margin="0,5" ToolTip="Введите логин" />  <!-- Пароль -->  <TextBlock Text="Пароль" Margin="0,5" />  <PasswordBox x:Name="PasswordBoxPassword" Margin="0,5" ToolTip="Введите пароль" />  <!-- Кнопка входа -->  <Button Content="Войти" Margin="0,10" Click="LoginButton\_Click" />  <!-- Сообщение об ошибке -->  <TextBlock x:Name="TextBlockError" Foreground="Red" Visibility="Collapsed" Text="Ошибка авторизации!" HorizontalAlignment="Center" />  </StackPanel>  </Grid>  </Page> |

В листинге 3 представлен код страницы AuthPage.xaml.cs.

Листинг 3 – код MainWindow.xaml.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Security.Cryptography;  using System.Text;  using System.Threading.Tasks;  using System.Windows;  using System.Windows.Controls;  using System.Windows.Data;  using System.Windows.Documents;  using System.Windows.Input;  using System.Windows.Media;  using System.Windows.Media.Imaging;  using System.Windows.Navigation;  using System.Windows.Shapes;  namespace WpfApp3.pages  {      /// <summary>      /// Логика взаимодействия для AuthPage.xaml      /// </summary>      public partial class AuthPage : Page      {          private readonly RepairRequestsDBEntities \_dbContext = new RepairRequestsDBEntities();          public AuthPage()          {              InitializeComponent();          }          // Метод для хэширования паролей          private static string GetHash(string input)          {              using (var sha1 = SHA1.Create())              {                  byte[] data = sha1.ComputeHash(Encoding.UTF8.GetBytes(input));                  return string.Concat(data.Select(b => b.ToString("X2"))); // Хэш в верхнем регистре              }          }          // Обработчик кнопки входа          private void LoginButton\_Click(object sender, RoutedEventArgs e)          {              // Получение данных из полей              string login = TextBoxLogin.Text.Trim();              string password = PasswordBoxPassword.Password.Trim();              // Проверка на пустые поля              if (string.IsNullOrEmpty(login) || string.IsNullOrEmpty(password))              {                  ShowErrorMessage("Введите логин и пароль!");                  return;              }              // Хэширование пароля              string passwordHash = GetHash(password);              // Поиск пользователя в базе данных              var user = \_dbContext.Users.FirstOrDefault(u => u.Login == login && u.PasswordHash == passwordHash);              if (user == null)              {                  ShowErrorMessage("Неверный логин или пароль!");                  return;              }              // Переход на главную страницу в зависимости от роли              switch (user.Role)              {                  case "Администратор":                      NavigationService.Navigate(new AdminPage());                      break;                  case "Менеджер":                      NavigationService.Navigate(new ManagerPage());                      break;                  case "Исполнитель":                      NavigationService.Navigate(new ExecutorPage(user.ID));                      break;                  default:                      ShowErrorMessage("Неизвестная роль пользователя!");                      break;              }          }          // Показать сообщение об ошибке          private void ShowErrorMessage(string message)          {              TextBlockError.Text = message;              TextBlockError.Visibility = Visibility.Visible;          }      }  } |

На рисунке 9 представлена страница редактирования.

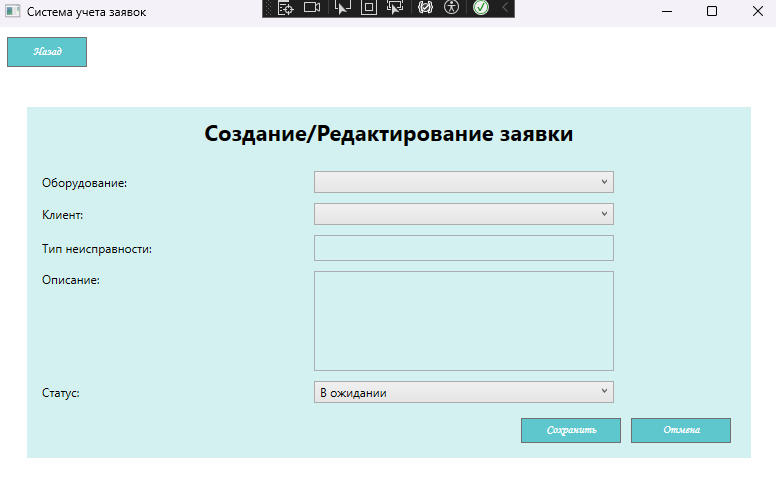


Рисунок 9 – страница редактирования

В листинге 4 представлен код странице AddRequestPage.xaml:

Листинг 4 – код AddRequestPage.xaml

|  |
| --- |
| <Page x:Class="WpfApp3.pages.AddRequestPage"  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"  xmlns:local="clr-namespace:WpfApp3.pages"  mc:Ignorable="d"  d:DesignHeight="450" d:DesignWidth="800"  Title="AddRequestPage">  <Grid Background="#D2F1F0" Margin="20">  <Grid.RowDefinitions>  <RowDefinition Height="Auto"/>  <RowDefinition Height="\*"/>  </Grid.RowDefinitions>  <!-- Заголовок страницы -->  <TextBlock Grid.Row="0" Text="Создание/Редактирование заявки"  HorizontalAlignment="Center" FontSize="22"  FontWeight="Bold" Margin="0,10"/>  <!-- Основная разметка формы -->  <Grid Grid.Row="1" Margin="10">  <Grid.RowDefinitions>  <RowDefinition Height="Auto"/>  <RowDefinition Height="Auto"/>  <RowDefinition Height="Auto"/>  <RowDefinition Height="Auto"/>  <RowDefinition Height="Auto"/>  <RowDefinition Height="Auto"/>  <RowDefinition Height="\*"/>  </Grid.RowDefinitions>  <Grid.ColumnDefinitions>  <ColumnDefinition Width="150"/>  <ColumnDefinition Width="\*"/>  </Grid.ColumnDefinitions>  <!-- Поля формы -->  <TextBlock Grid.Row="0" Grid.Column="0" Text="Оборудование:"  VerticalAlignment="Center" Margin="5"/>  <ComboBox x:Name="ComboBoxEquipment" Grid.Row="0" Grid.Column="1"  Margin="5" Width="300"/>  <TextBlock Grid.Row="1" Grid.Column="0" Text="Клиент:"  VerticalAlignment="Center" Margin="5"/>  <ComboBox x:Name="ComboBoxClient" Grid.Row="1" Grid.Column="1"  Margin="5" Width="300"/>  <TextBlock Grid.Row="2" Grid.Column="0" Text="Тип неисправности:"  VerticalAlignment="Center" Margin="5"/>  <TextBox x:Name="TextBoxFaultType" Grid.Row="2" Grid.Column="1"  Margin="5" Width="300"/>  <TextBlock Grid.Row="3" Grid.Column="0" Text="Описание:"  VerticalAlignment="Top" Margin="5"/>  <TextBox x:Name="TextBoxDescription" Grid.Row="3" Grid.Column="1"  Margin="5" Width="300" Height="100" TextWrapping="Wrap"  AcceptsReturn="True"/>  <TextBlock Grid.Row="4" Grid.Column="0" Text="Статус:"  VerticalAlignment="Center" Margin="5"/>  <ComboBox x:Name="ComboBoxStatus" Grid.Row="4" Grid.Column="1"  Margin="5" Width="300">  <ComboBoxItem Content="В ожидании" IsSelected="True"/>  <ComboBoxItem Content="В работе"/>  <ComboBoxItem Content="Выполнено"/>  </ComboBox>  <!-- Кнопки -->  <StackPanel Grid.Row="5" Grid.Column="1" Orientation="Horizontal"  HorizontalAlignment="Right" Margin="5">  <Button x:Name="ButtonSave" Content="Сохранить"  Width="100" Margin="5" Click="ButtonSave\_Click"/>  <Button x:Name="ButtonCancel" Content="Отмена"  Width="100" Margin="5" Click="ButtonCancel\_Click"/>  </StackPanel>  </Grid>  </Grid>  </Page> |

В листинге 5 представлен код RequestPage.xaml.cs.

Листинг 5 – код AddRequestPage.xaml.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows;  using System.Windows.Controls;  using System.Windows.Data;  using System.Windows.Documents;  using System.Windows.Input;  using System.Windows.Media;  using System.Windows.Media.Imaging;  using System.Windows.Navigation;  using System.Windows.Shapes;  namespace WpfApp3.pages  {      /// <summary>      /// Логика взаимодействия для AddRequestPage.xaml      /// </summary>      public partial class AddRequestPage : Page      {          private Requests \_currentRequest;          private readonly RepairRequestsDBEntities \_context = new RepairRequestsDBEntities();          public AddRequestPage(Requests selectedRequest = null)          {              InitializeComponent();              // Если передана заявка, редактируем её, иначе создаём новую              \_currentRequest = selectedRequest ?? new Requests();              DataContext = \_currentRequest;              LoadComboBoxes();          }          private void LoadComboBoxes()          {              // Заполнение ComboBoxEquipment              ComboBoxEquipment.ItemsSource = \_context.Equipment.ToList();              ComboBoxEquipment.DisplayMemberPath = "Name";              ComboBoxEquipment.SelectedValuePath = "ID";              // Заполнение ComboBoxClient              ComboBoxClient.ItemsSource = \_context.Users.ToList();              ComboBoxClient.DisplayMemberPath = "FullName"; // Отображаем ФИО              ComboBoxClient.SelectedValuePath = "ID";              // Если заявка существует, устанавливаем текущие значения              if (\_currentRequest.ID != 0)              {                  ComboBoxEquipment.SelectedValue = \_currentRequest.EquipmentID;                  ComboBoxClient.SelectedValue = \_currentRequest.ClientID;                  TextBoxFaultType.Text = \_currentRequest.FaultType;                  TextBoxDescription.Text = \_currentRequest.Description;                  ComboBoxStatus.Text = \_currentRequest.Status;              }          }          private void ButtonSave\_Click(object sender, RoutedEventArgs e)          {              try              {                  // Проверка на пустые поля                  if (ComboBoxEquipment.SelectedValue == null || ComboBoxClient.SelectedValue == null)                  {                      MessageBox.Show("Пожалуйста, выберите оборудование и клиента.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);                      return;                  }                  // Заполнение текущей заявки                  \_currentRequest.EquipmentID = (int)ComboBoxEquipment.SelectedValue;                  \_currentRequest.ClientID = (int)ComboBoxClient.SelectedValue;                  \_currentRequest.FaultType = TextBoxFaultType.Text.Trim();                  \_currentRequest.Description = TextBoxDescription.Text.Trim();                  \_currentRequest.Status = ComboBoxStatus.Text;                  \_currentRequest.DateAdded = \_currentRequest.ID == 0 ? DateTime.Now : \_currentRequest.DateAdded;                  if (\_currentRequest.ID == 0)                  {                      // Генерация RequestNumber                      int nextId = \_context.Requests.Any()                          ? \_context.Requests.Max(r => r.ID) + 1                          : 1;                      \_currentRequest.RequestNumber = $"REQ-{nextId:D3}";                      \_context.Requests.Add(\_currentRequest); // Добавляем новую заявку                  }                  else                  {                      // Обновляем существующую заявку                      var existingRequest = \_context.Requests.Find(\_currentRequest.ID);                      if (existingRequest != null)                      {                          existingRequest.EquipmentID = \_currentRequest.EquipmentID;                          existingRequest.ClientID = \_currentRequest.ClientID;                          existingRequest.FaultType = \_currentRequest.FaultType;                          existingRequest.Description = \_currentRequest.Description;                          existingRequest.Status = \_currentRequest.Status;                      }                  }                  \_context.SaveChanges();                  MessageBox.Show("Заявка успешно сохранена!", "Успех", MessageBoxButton.OK, MessageBoxImage.Information);                  NavigationService.GoBack();              }              catch (Exception ex)              {                  MessageBox.Show($"Ошибка при сохранении заявки: {ex.Message}", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);              }          }            private void ButtonCancel\_Click(object sender, RoutedEventArgs e)          {              NavigationService.GoBack();          }      }  } |

Так же была добавлена админская панель. Для этого была создана страница AdminPage, которая представлена на рисунке 10.

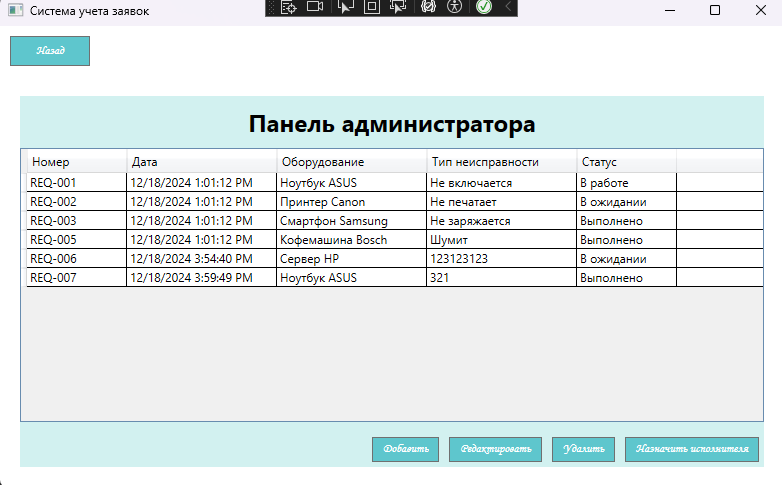


Рисунок 10 – страница AdminPage

В листинге 6 представлен код станицы AdminPage.xaml.

Листинг 6 – код AdminPage.xaml

|  |
| --- |
| <Page x:Class="WpfApp3.pages.AdminPage"  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"  xmlns:local="clr-namespace:WpfApp3.pages"  mc:Ignorable="d"  d:DesignHeight="450" d:DesignWidth="800"  Title="AdminPage"  IsVisibleChanged="Page\_IsVisibleChanged">  <Grid Background="#D2F1F0" Margin="10">  <Grid.RowDefinitions>  <RowDefinition Height="Auto" />  <RowDefinition Height="\*" />  </Grid.RowDefinitions>  <!-- Заголовок -->  <TextBlock Text="Панель администратора" FontSize="24" FontWeight="Bold" HorizontalAlignment="Center" Margin="0,10" />  <!-- Основной контент -->  <Grid Grid.Row="1">  <Grid.RowDefinitions>  <RowDefinition Height="\*" />  <RowDefinition Height="Auto" />  </Grid.RowDefinitions>  <!-- Таблица заявок -->  <DataGrid x:Name="DataGridRequests" AutoGenerateColumns="False" IsReadOnly="True" Margin="0,0,0,10">  <DataGrid.Columns>  <DataGridTextColumn Header="Номер" Binding="{Binding RequestNumber}" Width="100" />  <DataGridTextColumn Header="Дата" Binding="{Binding DateAdded}" Width="150" />  <DataGridTextColumn Header="Оборудование" Binding="{Binding Equipment.Name}" Width="150" />  <DataGridTextColumn Header="Тип неисправности" Binding="{Binding FaultType}" Width="150" />  <DataGridTextColumn Header="Статус" Binding="{Binding Status}" Width="100" />  </DataGrid.Columns>  </DataGrid>  <!-- Кнопки управления -->  <StackPanel Grid.Row="1" Orientation="Horizontal" HorizontalAlignment="Right">  <Button Content="Добавить" Click="ButtonAdd\_Click" Margin="5" />  <Button Content="Редактировать" Click="ButtonEdit\_Click" Margin="5" />  <Button Content="Удалить" Click="ButtonDelete\_Click" Margin="5" />  <Button Content="Назначить исполнителя" Click="ButtonAssignExecutor\_Click" Margin="5" />  </StackPanel>  </Grid>  </Grid>  </Page> |

В листинге 7 представлен код страницы AdminPage.xaml.cs.

Листинг 7 – код AdminPage.xaml.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows;  using System.Windows.Controls;  using System.Windows.Data;  using System.Windows.Documents;  using System.Windows.Input;  using System.Windows.Media;  using System.Windows.Media.Imaging;  using System.Windows.Navigation;  using System.Windows.Shapes;  namespace WpfApp3.pages  {      /// <summary>      /// Логика взаимодействия для AdminPage.xaml      /// </summary>      public partial class AdminPage : Page      {          private readonly RepairRequestsDBEntities \_context = new RepairRequestsDBEntities();          public AdminPage()          {              InitializeComponent();              LoadRequests();          }          private void LoadRequests()          {              DataGridRequests.ItemsSource = \_context.Requests.Include("Equipment").ToList();          }          private void ButtonAdd\_Click(object sender, RoutedEventArgs e)          {              NavigationService.Navigate(new AddRequestPage());          }          private void Page\_IsVisibleChanged(object sender, DependencyPropertyChangedEventArgs e)          {              if (Visibility == Visibility.Visible)              {                  \_context.ChangeTracker.Entries().ToList().ForEach(entry => entry.Reload());                  LoadRequests();              }          }          private void ButtonEdit\_Click(object sender, RoutedEventArgs e)          {              if (DataGridRequests.SelectedItem is Requests selectedRequest)              {                  NavigationService.Navigate(new AddRequestPage(selectedRequest));              }              else              {                  MessageBox.Show("Выберите заявку для редактирования!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);              }          }          private void ButtonDelete\_Click(object sender, RoutedEventArgs e)          {              var selectedRequests = DataGridRequests.SelectedItems.Cast<Requests>().ToList();              if (!selectedRequests.Any())              {                  MessageBox.Show("Выберите хотя бы одну заявку для удаления.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);                  return;              }              if (MessageBox.Show($"Удалить {selectedRequests.Count} заявок?", "Подтверждение", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)              {                  try                  {                      // Удаление связанных записей (если необходимо)                      foreach (var request in selectedRequests)                      {                          var relatedRecords = \_context.Repairs.Where(c => c.RequestID == request.ID).ToList();                          \_context.Repairs.RemoveRange(relatedRecords);                          var relatedFeedbackRecords = \_context.Feedback.Where(c => c.RequestID == request.ID).ToList();                          \_context.Feedback.RemoveRange(relatedFeedbackRecords);                          \_context.Requests.Remove(request);                      }                      \_context.SaveChanges();                      MessageBox.Show("Заявки успешно удалены!", "Успех", MessageBoxButton.OK, MessageBoxImage.Information);                      LoadRequests();                  }                  catch (Exception ex)                  {                      MessageBox.Show($"Ошибка при удалении заявок: {ex.InnerException?.Message ?? ex.Message}", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);                  }              }          }          private void ButtonAssignExecutor\_Click(object sender, RoutedEventArgs e)          {              if (DataGridRequests.SelectedItem is Requests selectedRequest)              {                  NavigationService.Navigate(new AssignExecutorPage(selectedRequest));              }              else              {                  MessageBox.Show("Выберите заявку для назначения исполнителя!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);              }          }      }  } |

На рисунке 11 предоставлена панель менеджера:

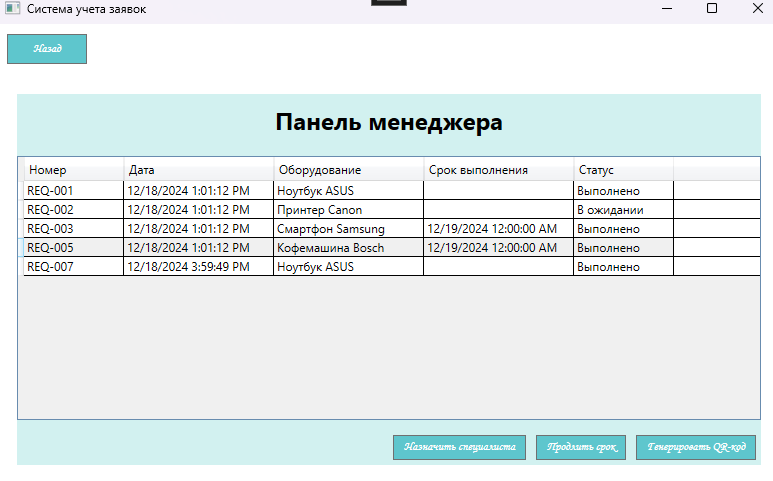


Рисунок 11. Панель менеджера

В листинге 8 предоставлен код страницы ManagerPage.xaml.

Листинг 8 – код ManagerPage.xaml

|  |
| --- |
| <Page x:Class="WpfApp3.pages.ManagerPage"  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"  xmlns:local="clr-namespace:WpfApp3.pages"  mc:Ignorable="d"  d:DesignHeight="450" d:DesignWidth="800"  Title="ManagerPage">  <Grid Background="#D2F1F0" Margin="10">  <Grid.RowDefinitions>  <RowDefinition Height="Auto" />  <RowDefinition Height="\*" />  <RowDefinition Height="Auto" />  </Grid.RowDefinitions>  <!-- Заголовок -->  <TextBlock Text="Панель менеджера" FontSize="24" FontWeight="Bold" HorizontalAlignment="Center" Margin="0,10" />  <!-- Таблица заявок -->  <DataGrid x:Name="DataGridRequests" Grid.Row="1" AutoGenerateColumns="False" IsReadOnly="True" Margin="0,10">  <DataGrid.Columns>  <DataGridTextColumn Header="Номер" Binding="{Binding RequestNumber}" Width="100" />  <DataGridTextColumn Header="Дата" Binding="{Binding DateAdded}" Width="150" />  <DataGridTextColumn Header="Оборудование" Binding="{Binding Equipment.Name}" Width="150" />  <DataGridTextColumn Header="Срок выполнения" Binding="{Binding Deadline}" Width="150" />  <DataGridTextColumn Header="Статус" Binding="{Binding Status}" Width="100" />  </DataGrid.Columns>  </DataGrid>  <!-- Кнопки управления -->  <StackPanel Grid.Row="2" Orientation="Horizontal" HorizontalAlignment="Right">  <Button Content="Назначить специалиста" Click="ButtonAssignSpecialist\_Click" Margin="5" />  <Button Content="Продлить срок" Click="ButtonExtendDeadline\_Click" Margin="5" />  <Button Content="Генерировать QR-код" Click="ButtonGenerateQR\_Click" Margin="5" />  </StackPanel>  </Grid>  </Page> |

В листинге 9 предоставлен код страницы ManagerPage.xaml.cs

|  |
| --- |
| using System;  using System.Linq;  using System.Windows;  using System.Windows.Controls;  namespace WpfApp3.pages  {  /// <summary>  /// Логика взаимодействия для ManagerPage.xaml  /// </summary>  public partial class ManagerPage : Page  {  private readonly RepairRequestsDBEntities \_context = new RepairRequestsDBEntities();  public ManagerPage()  {  InitializeComponent();  LoadRequests();  }  private void LoadRequests()  {  // Загружаем заявки с включением связанных данных  DataGridRequests.ItemsSource = \_context.Requests  .Include("Equipment")  .Include("Repairs.Users")  .ToList();  }  private void ButtonAssignSpecialist\_Click(object sender, RoutedEventArgs e)  {  if (DataGridRequests.SelectedItem is Requests selectedRequest)  {  // Открываем окно для выбора исполнителя  var assignPage = new AssignExecutorPage(selectedRequest);  NavigationService.Navigate(assignPage);  }  else  {  MessageBox.Show("Выберите заявку для назначения специалиста!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);  }  }  private void ButtonExtendDeadline\_Click(object sender, RoutedEventArgs e)  {  if (DataGridRequests.SelectedItem is Requests selectedRequest)  {  var inputDialog = new InputDialog("Введите новую дату окончания (ГГГГ-ММ-ДД):");  if (inputDialog.ShowDialog() == true)  {  if (DateTime.TryParse(inputDialog.Input, out DateTime newDeadline))  {  selectedRequest.Deadline = newDeadline;  \_context.SaveChanges();  LoadRequests();  MessageBox.Show("Срок выполнения продлен!", "Успех", MessageBoxButton.OK, MessageBoxImage.Information);  }  else  {  MessageBox.Show("Неправильный формат даты.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);  }  }  }  else  {  MessageBox.Show("Выберите заявку для продления срока!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);  }  }  private void ButtonGenerateQR\_Click(object sender, RoutedEventArgs e)  {    }  }  } |