-- Таблица пользователей

CREATE TABLE users (

userID INT PRIMARY KEY IDENTITY(1,1),

lastName NVARCHAR(100) NOT NULL,

firstName NVARCHAR(100) NOT NULL,

middleName NVARCHAR(100) NULL, -- Отчество может отсутствовать

phone NVARCHAR(20) NOT NULL UNIQUE,

login NVARCHAR(100) NOT NULL UNIQUE,

password NVARCHAR(255) NOT NULL,

userType NVARCHAR(50) CHECK (userType IN ('Менеджер', 'Специалист', 'Оператор', 'Заказчик')) NOT NULL

);

-- Таблица типов климатической техники

CREATE TABLE climate\_tech\_types (

techTypeID INT PRIMARY KEY IDENTITY(1,1),

techTypeName NVARCHAR(100) NOT NULL UNIQUE

);

-- Таблица моделей климатической техники

CREATE TABLE climate\_tech\_models (

modelID INT PRIMARY KEY IDENTITY(1,1),

modelName NVARCHAR(255) NOT NULL UNIQUE,

techTypeID INT NOT NULL,

FOREIGN KEY (techTypeID) REFERENCES climate\_tech\_types(techTypeID)

);

-- Таблица статусов заявок

CREATE TABLE statuses (

statusID INT PRIMARY KEY IDENTITY(1,1),

statusName NVARCHAR(100) NOT NULL UNIQUE

);

-- Таблица заявок на ремонт

CREATE TABLE requests (

requestID INT PRIMARY KEY IDENTITY(1,1),

startDate DATE NOT NULL,

modelID INT NOT NULL,

problemDescription NVARCHAR(MAX) NOT NULL,

requestStatus INT NOT NULL,

completionDate DATE NULL,

masterID INT NULL,

clientID INT NOT NULL,

FOREIGN KEY (modelID) REFERENCES climate\_tech\_models(modelID),

FOREIGN KEY (requestStatus) REFERENCES statuses(statusID),

FOREIGN KEY (masterID) REFERENCES users(userID),

FOREIGN KEY (clientID) REFERENCES users(userID)

);

-- Таблица комментариев к заявкам

CREATE TABLE comments (

commentID INT PRIMARY KEY IDENTITY(1,1),

message NVARCHAR(MAX) NOT NULL,

masterID INT NOT NULL,

requestID INT NOT NULL,

FOREIGN KEY (masterID) REFERENCES users(userID),

FOREIGN KEY (requestID) REFERENCES requests(requestID)

);

-- Таблица деталей для ремонта

CREATE TABLE repair\_parts (

partID INT PRIMARY KEY IDENTITY(1,1),

partName NVARCHAR(255) NOT NULL UNIQUE

);

-- Таблица связки заявок и деталей для ремонта

CREATE TABLE request\_parts (

requestID INT NOT NULL,

partID INT NOT NULL,

quantity INT NOT NULL CHECK (quantity > 0),

PRIMARY KEY (requestID, partID),

FOREIGN KEY (requestID) REFERENCES requests(requestID),

FOREIGN KEY (partID) REFERENCES repair\_parts(partID)

);

RegisterForm

<Window x:Class="demka.RegisterForm"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="Регистрация пользователя" Height="450" Width="400">

<Window.Resources>

<!-- Стили для улучшения внешнего вида элементов -->

<Style TargetType="TextBox">

<Setter Property="Height" Value="30"/>

<Setter Property="Margin" Value="10"/>

<Setter Property="Padding" Value="5"/>

<Setter Property="FontSize" Value="14"/>

</Style>

<Style TargetType="PasswordBox">

<Setter Property="Height" Value="30"/>

<Setter Property="Margin" Value="10"/>

<Setter Property="Padding" Value="5"/>

<Setter Property="FontSize" Value="14"/>

</Style>

<Style TargetType="ComboBox">

<Setter Property="Height" Value="30"/>

<Setter Property="Margin" Value="10"/>

<Setter Property="FontSize" Value="14"/>

</Style>

<Style TargetType="Button">

<Setter Property="Height" Value="40"/>

<Setter Property="Width" Value="200"/>

<Setter Property="FontSize" Value="16"/>

<Setter Property="Margin" Value="0,20,0,0"/>

<Setter Property="Background" Value="#4CAF50"/>

<Setter Property="Foreground" Value="White"/>

<Setter Property="Cursor" Value="Hand"/>

<Setter Property="BorderBrush" Value="#4CAF50"/>

<Setter Property="BorderThickness" Value="1"/>

<Setter Property="HorizontalAlignment" Value="Center"/>

</Style>

</Window.Resources>

<Grid Background="#F7F7F7">

<StackPanel VerticalAlignment="Top" HorizontalAlignment="Center" Margin="20">

<TextBlock Text="Регистрация пользователя" FontSize="20" FontWeight="Bold" Margin="0,0,0,30" HorizontalAlignment="Center"/>

<Label Content="Фамилия" HorizontalAlignment="Left" VerticalAlignment="Top"/>

<TextBox Name="LastNameTextBox" Width="300"/>

<Label Content="Имя" HorizontalAlignment="Left" VerticalAlignment="Top"/>

<TextBox Name="FirstNameTextBox" Width="300"/>

<Label Content="Отчество" HorizontalAlignment="Left" VerticalAlignment="Top"/>

<TextBox Name="MiddleNameTextBox" Width="300"/>

<Label Content="Телефон" HorizontalAlignment="Left" VerticalAlignment="Top"/>

<TextBox Name="PhoneTextBox" Width="300"/>

<Label Content="Логин" HorizontalAlignment="Left" VerticalAlignment="Top"/>

<TextBox Name="LoginTextBox" Width="300"/>

<Label Content="Пароль" HorizontalAlignment="Left" VerticalAlignment="Top"/>

<PasswordBox Name="PasswordBox" Width="300"/>

<Label Content="Роль" HorizontalAlignment="Left" VerticalAlignment="Top"/>

<ComboBox Name="RoleComboBox" Width="300">

<ComboBoxItem Content="Менеджер"/>

<ComboBoxItem Content="Специалист"/>

<ComboBoxItem Content="Оператор"/>

<ComboBoxItem Content="Заказчик"/>

</ComboBox>

<Button Name="RegisterButton" Content="Зарегистрироваться" Click="RegisterButton\_Click"/>

</StackPanel>

</Grid>

</Window>

using System;

using System.Configuration;

using System.Data.SqlClient;

using System.Windows;

using System.Windows.Controls;

namespace demka

{

public partial class RegisterForm : Window

{

string connectionString = "Server=localhost;Database=demka;Integrated Security=True;TrustServerCertificate=True;";

public RegisterForm()

{

InitializeComponent();

}

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

{

string lastName = LastNameTextBox.Text.Trim();

string firstName = FirstNameTextBox.Text.Trim();

string middleName = MiddleNameTextBox.Text.Trim();

string phone = PhoneTextBox.Text.Trim();

string login = LoginTextBox.Text.Trim();

string password = PasswordBox.Password.Trim();

string userType = (RoleComboBox.SelectedItem as ComboBoxItem)?.Content.ToString();

// Проверка обязательных полей

if (string.IsNullOrEmpty(lastName) || string.IsNullOrEmpty(firstName) ||

string.IsNullOrEmpty(phone) || string.IsNullOrEmpty(login) ||

string.IsNullOrEmpty(password) || string.IsNullOrEmpty(userType))

{

MessageBox.Show("Заполните все обязательные поля!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

return;

}

// Проверка выбора роли

if (userType == null)

{

MessageBox.Show("Выберите роль пользователя.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

return;

}

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "INSERT INTO Users (lastName, firstName, middleName, phone, login, password, userType) " +

"VALUES (@lastName, @firstName, @middleName, @phone, @login, @password, @userType)";

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@lastName", lastName);

cmd.Parameters.AddWithValue("@firstName", firstName);

cmd.Parameters.AddWithValue("@middleName", middleName);

cmd.Parameters.AddWithValue("@phone", phone);

cmd.Parameters.AddWithValue("@login", login);

cmd.Parameters.AddWithValue("@password", password);

cmd.Parameters.AddWithValue("@userType", userType);

try

{

conn.Open();

int rowsAffected = cmd.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Пользователь успешно зарегистрирован!", "Успех", MessageBoxButton.OK, MessageBoxImage.Information);

NavigateToUserPage(userType);

this.Close();

}

else

{

MessageBox.Show("Ошибка при добавлении пользователя.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

catch (Exception ex)

{

MessageBox.Show("Ошибка: " + ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

}

private void NavigateToUserPage(string userType)

{

Window targetWindow = null;

switch (userType)

{

case "Менеджер":

targetWindow = new RequestsListForm(); // Менеджер управляет заявками

break;

case "Специалист":

targetWindow = new ProcessRequestForm(); // Специалист обрабатывает заявки

break;

case "Оператор":

targetWindow = new AddRequestForm(); // Оператор добавляет заявки

break;

case "Заказчик":

int specialistID = 1; // Пример для передачи specialistID. Это значение нужно заменить на реальное.

targetWindow = new CompleteRequestForm(specialistID); // Заказчик завершает заявки

break;

default:

MessageBox.Show("Неизвестная роль пользователя.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

return;

}

if (targetWindow != null)

{

targetWindow.Show();

this.Close();

}

}

}

}

RequestsListForm

<Window x:Class="demka.RequestsListForm"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="Список заявок" Height="450" Width="600">

<Window.Resources>

<!-- Стиль для ComboBox -->

<Style TargetType="ComboBox">

<Setter Property="Height" Value="35"/>

<Setter Property="Margin" Value="10"/>

<Setter Property="FontSize" Value="14"/>

<Setter Property="Padding" Value="5"/>

</Style>

<!-- Стиль для DataGrid -->

<Style TargetType="DataGrid">

<Setter Property="Height" Value="250"/>

<Setter Property="Width" Value="580"/>

<Setter Property="Margin" Value="10"/>

<Setter Property="FontSize" Value="14"/>

<Setter Property="AutoGenerateColumns" Value="True"/>

<Setter Property="SelectionMode" Value="Single"/>

<Setter Property="GridLinesVisibility" Value="None"/>

<Setter Property="HeadersVisibility" Value="Column"/>

<Setter Property="Background" Value="#F5F5F5"/>

<Setter Property="BorderBrush" Value="#CCCCCC"/>

<Setter Property="BorderThickness" Value="1"/>

<Setter Property="AlternatingRowBackground" Value="#E9E9E9"/>

</Style>

<!-- Стиль для Label -->

<Style TargetType="Label">

<Setter Property="FontSize" Value="16"/>

<Setter Property="FontWeight" Value="Bold"/>

<Setter Property="Margin" Value="10"/>

</Style>

<!-- Стиль для Button -->

<Style TargetType="Button">

<Setter Property="Height" Value="40"/>

<Setter Property="Width" Value="150"/>

<Setter Property="FontSize" Value="16"/>

<Setter Property="Background" Value="#4CAF50"/>

<Setter Property="Foreground" Value="White"/>

<Setter Property="Cursor" Value="Hand"/>

<Setter Property="BorderBrush" Value="#4CAF50"/>

<Setter Property="BorderThickness" Value="1"/>

<Setter Property="HorizontalAlignment" Value="Center"/>

</Style>

</Window.Resources>

<Grid Background="#F7F7F7">

<!-- Верхняя панель с фильтром -->

<StackPanel VerticalAlignment="Top" HorizontalAlignment="Center" Margin="20">

<TextBlock Text="Список заявок" FontSize="22" FontWeight="Bold" Margin="0,0,0,20" HorizontalAlignment="Center"/>

<!-- Фильтр по статусу -->

<Label Content="Фильтр по статусу"/>

<ComboBox Name="StatusFilterComboBox" Width="250" SelectionChanged="StatusFilterComboBox\_SelectionChanged">

<ComboBoxItem Content="Все"/>

<ComboBoxItem Content="Новая заявка"/>

<ComboBoxItem Content="В работе"/>

<ComboBoxItem Content="Выполнено"/>

</ComboBox>

<!-- Таблица с заявками -->

<DataGrid Name="RequestsDataGrid" HorizontalAlignment="Stretch" VerticalAlignment="Top" Margin="0,50,0,0" />

</StackPanel>

</Grid>

</Window>

using System;

using System.Data;

using System.Data.SqlClient;

using System.Windows;

using System.Windows.Controls;

namespace demka

{

public partial class RequestsListForm : Window

{

string connectionString = "Server=localhost;Database=demka;Integrated Security=True;TrustServerCertificate=True;";

public RequestsListForm()

{

InitializeComponent();

LoadRequests();

}

private void LoadRequests(string statusFilter = "Все")

{

string query = @"

SELECT r.requestID AS '№',

tt.techTypeName AS 'Тип оборудования',

tm.modelName AS 'Модель',

r.problemDescription AS 'Описание',

s.statusName AS 'Статус',

(SELECT lastName + ' ' + firstName FROM Users WHERE Users.userID = r.masterID) AS 'Специалист'

FROM Requests r

JOIN climate\_tech\_models tm ON r.modelID = tm.modelID

JOIN climate\_tech\_types tt ON tm.techTypeID = tt.techTypeID

JOIN statuses s ON r.requestStatus = s.statusID";

if (statusFilter != "Все")

{

query += " WHERE r.requestStatus = @statusFilter";

}

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlCommand cmd = new SqlCommand(query, conn);

if (statusFilter != "Все")

{

cmd.Parameters.AddWithValue("@statusFilter", statusFilter);

}

SqlDataAdapter adapter = new SqlDataAdapter(cmd);

DataTable dt = new DataTable();

adapter.Fill(dt);

RequestsDataGrid.ItemsSource = dt.DefaultView;

}

}

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

{

string selectedStatus = (StatusFilterComboBox.SelectedItem as ComboBoxItem)?.Content.ToString();

LoadRequests(selectedStatus);

}

}

}

StatisticsForm

<Window x:Class="demka.StatisticsForm"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="Статистика" Height="300" Width="400">

<Grid>

<Label Content="Начальная дата" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="10,10,0,0"/>

<DatePicker Name="StartDatePicker" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="150,10,0,0"/>

<Label Content="Конечная дата" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="10,50,0,0"/>

<DatePicker Name="EndDatePicker" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="150,50,0,0"/>

<Button Name="GetStatisticsButton" Content="Получить статистику" HorizontalAlignment="Left" VerticalAlignment="Top" Width="200" Height="30" Margin="100,100,0,0" Click="GetStatisticsButton\_Click"/>

<Label Name="ResultLabel" Content="" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="10,150,0,0"/>

</Grid>

</Window>

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

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 demka

{

/// <summary>

/// Логика взаимодействия для StatisticsForm.xaml

/// </summary>

public partial class StatisticsForm : Window

{

string connectionString = "Server=localhost;Database=demka;Integrated Security=True;TrustServerCertificate=True;";

public StatisticsForm()

{

InitializeComponent();

}

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

{

DateTime startDate = StartDatePicker.SelectedDate ?? DateTime.MinValue;

DateTime endDate = EndDatePicker.SelectedDate ?? DateTime.MaxValue;

string query = "SELECT COUNT(\*) FROM Requests WHERE requestStatus = 'Выполнено' " +

"AND completionDate BETWEEN @startDate AND @endDate";

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@startDate", startDate);

cmd.Parameters.AddWithValue("@endDate", endDate);

try

{

conn.Open();

int count = (int)cmd.ExecuteScalar();

ResultLabel.Content = $"Выполнено заявок: {count}";

}

catch (Exception ex)

{

MessageBox.Show("Ошибка: " + ex.Message);

}

}

}

}

}

ProcessRequestForm

<Window x:Class="demka.ProcessRequestForm"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="Обработка заявки" Height="350" Width="450">

<Grid>

<Label Content="Выберите заявку" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="10,10,0,0"/>

<ComboBox Name="RequestComboBox" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="150,10,10,0" Width="250"/>

<Label Content="Назначить специалиста" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="10,50,0,0"/>

<ComboBox Name="SpecialistComboBox" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="150,50,10,0" Width="250"/>

<Button Name="AssignButton" Content="Назначить" HorizontalAlignment="Left" VerticalAlignment="Top" Width="150" Height="30" Margin="150,100,0,0" Click="AssignButton\_Click"/>

</Grid>

</Window>

using System;

using System.Data.SqlClient;

using System.Windows;

using System.Windows.Controls;

namespace demka

{

/// <summary>

/// Логика взаимодействия для ProcessRequestForm.xaml

/// </summary>

public partial class ProcessRequestForm : Window

{

string connectionString = "Server=localhost;Database=demka;Integrated Security=True;TrustServerCertificate=True;";

public ProcessRequestForm()

{

InitializeComponent();

LoadRequests();

LoadSpecialists();

}

private void LoadRequests()

{

string query = @"

SELECT r.requestID,

tt.techTypeName + ' (' + tm.modelName + ')' AS requestInfo

FROM Requests r

JOIN climate\_tech\_models tm ON r.modelID = tm.modelID

JOIN climate\_tech\_types tt ON tm.techTypeID = tt.techTypeID

WHERE r.requestStatus = (SELECT statusID FROM statuses WHERE statusName = 'Новая заявка')";

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlCommand cmd = new SqlCommand(query, conn);

conn.Open();

SqlDataReader reader = cmd.ExecuteReader();

while (reader.Read())

{

ComboBoxItem item = new ComboBoxItem

{

Content = reader["requestInfo"].ToString(),

Tag = reader["requestID"]

};

RequestComboBox.Items.Add(item);

}

}

}

private void LoadSpecialists()

{

string query = @"

SELECT userID,

lastName + ' ' + firstName AS specialistName

FROM Users

WHERE userType = 'Специалист'";

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlCommand cmd = new SqlCommand(query, conn);

conn.Open();

SqlDataReader reader = cmd.ExecuteReader();

while (reader.Read())

{

ComboBoxItem item = new ComboBoxItem

{

Content = reader["specialistName"].ToString(),

Tag = reader["userID"]

};

SpecialistComboBox.Items.Add(item);

}

}

}

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

{

int requestID = (int)((ComboBoxItem)RequestComboBox.SelectedItem).Tag;

int specialistID = (int)((ComboBoxItem)SpecialistComboBox.SelectedItem).Tag;

string query = @"

UPDATE Requests

SET masterID = @specialistID, requestStatus = (SELECT statusID FROM statuses WHERE statusName = 'В работе')

WHERE requestID = @requestID";

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@specialistID", specialistID);

cmd.Parameters.AddWithValue("@requestID", requestID);

try

{

conn.Open();

cmd.ExecuteNonQuery();

MessageBox.Show("Специалист успешно назначен!");

this.Close();

}

catch (Exception ex)

{

MessageBox.Show("Ошибка: " + ex.Message);

}

}

}

}

}

CompleteRequestForm

<Window x:Class="demka.CompleteRequestForm"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="Завершение заявки" Height="400" Width="450">

<Grid>

<Label Content="Выберите заявку" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="10,10,0,0"/>

<ComboBox Name="AssignedRequestsComboBox" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="150,10,10,0" Width="250"/>

<Label Content="Отчёт о проделанной работе" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="10,50,0,0"/>

<TextBox Name="ReportTextBox" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="10,90,10,0" Width="400" Height="100" TextWrapping="Wrap" AcceptsReturn="True"/>

<Button Name="CompleteButton" Content="Завершить заявку" HorizontalAlignment="Left" VerticalAlignment="Top" Width="150" Height="30" Margin="150,220,0,0" Click="CompleteButton\_Click"/>

</Grid>

</Window>

using System;

using System.Data.SqlClient;

using System.Windows;

using System.Windows.Controls;

namespace demka

{

/// <summary>

/// Логика взаимодействия для CompleteRequestForm.xaml

/// </summary>

public partial class CompleteRequestForm : Window

{

private int specialistID;

// Конструктор, который принимает specialistID

public CompleteRequestForm(int specialistID)

{

InitializeComponent();

this.specialistID = specialistID; // Присваиваем specialistID

LoadAssignedRequests(); // Загружаем заявки для специалиста

}

// Метод для загрузки заявок, назначенных данному специалисту

private void LoadAssignedRequests()

{

// Запрос для извлечения данных о заявках без конкатенации строк в SQL

string query = "SELECT r.requestID, t.techTypeName, m.modelName " +

"FROM requests r " +

"INNER JOIN climate\_tech\_models m ON r.modelID = m.modelID " +

"INNER JOIN climate\_tech\_types t ON m.techTypeID = t.techTypeID " +

"WHERE r.masterID = @specialistID AND r.requestStatus = 'В работе'";

using (SqlConnection conn = new SqlConnection("Server=localhost;Database=demka;Integrated Security=True;TrustServerCertificate=True;"))

{

SqlCommand cmd = new SqlCommand(query, conn);

// Убедимся, что specialistID - это int

if (specialistID <= 0)

{

MessageBox.Show("ID специалиста не установлен или неверен.");

return;

}

// Добавляем параметр для specialistID

cmd.Parameters.AddWithValue("@specialistID", specialistID);

try

{

conn.Open();

SqlDataReader reader = cmd.ExecuteReader();

// Очищаем ComboBox перед добавлением новых данных

AssignedRequestsComboBox.Items.Clear();

while (reader.Read())

{

// Получаем значения из базы данных

int requestID = Convert.ToInt32(reader["requestID"]);

string techTypeName = reader["techTypeName"].ToString();

string modelName = reader["modelName"].ToString();

// Формируем строку для отображения в ComboBox

string requestInfo = $"{techTypeName} ({modelName})";

// Добавляем элемент в ComboBox

ComboBoxItem item = new ComboBoxItem

{

Content = requestInfo,

Tag = requestID // Сохраняем requestID в Tag для дальнейшего использования

};

AssignedRequestsComboBox.Items.Add(item);

}

// Закрытие reader и соединения с базой данных

reader.Close();

}

catch (Exception ex)

{

MessageBox.Show("Ошибка подключения или выполнения запроса: " + ex.Message);

}

}

}

// Обработчик клика по кнопке "Завершить"

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

{

int requestID = (int)((ComboBoxItem)AssignedRequestsComboBox.SelectedItem).Tag;

string report = ReportTextBox.Text;

string query = "UPDATE Requests SET requestStatus = 'Выполнено', report = @report WHERE requestID = @requestID";

using (SqlConnection conn = new SqlConnection("Server=localhost;Database=demka;Integrated Security=True;TrustServerCertificate=True;"))

{

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@report", report);

cmd.Parameters.AddWithValue("@requestID", requestID);

try

{

conn.Open();

cmd.ExecuteNonQuery();

MessageBox.Show("Заявка успешно завершена!");

this.Close();

}

catch (Exception ex)

{

MessageBox.Show("Ошибка: " + ex.Message);

}

}

}

}

}

AddRequestForm

<Window x:Class="demka.AddRequestForm"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="Добавление заявки" Height="400" Width="450">

<Grid>

<Label Content="Тип оборудования" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="10,10,0,0"/>

<ComboBox Name="ClimateTechTypeComboBox" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="120,10,10,0" Width="250">

<ComboBoxItem Content="Кондиционер"/>

<ComboBoxItem Content="Увлажнитель воздуха"/>

<ComboBoxItem Content="Сушилка для рук"/>

</ComboBox>

<Label Content="Модель" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="10,50,0,0"/>

<ComboBox Name="ModelComboBox" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="120,50,10,0" Width="250">

<!-- Модели будут загружаться динамически -->

</ComboBox>

<Label Content="Описание проблемы" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="10,90,0,0"/>

<TextBox Name="ProblemDescriptionTextBox" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="120,90,10,0" Width="250"/>

<Label Content="Дата начала" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="10,130,0,0"/>

<DatePicker Name="StartDateDatePicker" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="120,130,10,0" Width="250"/>

<Label Content="Клиент" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="10,170,0,0"/>

<ComboBox Name="ClientComboBox" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="120,170,10,0" Width="250">

<!-- Клиенты будут загружаться из базы данных -->

</ComboBox>

<Button Name="AddRequestButton" Content="Добавить заявку" HorizontalAlignment="Left" VerticalAlignment="Top" Width="150" Height="30" Margin="150,250,0,0" Click="AddRequestButton\_Click"/>

</Grid>

</Window>

using System;

using System.Data.SqlClient;

using System.Windows;

using System.Windows.Controls;

namespace demka

{

/// <summary>

/// Логика взаимодействия для AddRequestForm.xaml

/// </summary>

public partial class AddRequestForm : Window

{

string connectionString = "Server=localhost;Database=demka;Integrated Security=True;TrustServerCertificate=True;";

public AddRequestForm()

{

InitializeComponent();

LoadClients();

LoadClimateTechTypes();

}

private void LoadClients()

{

// Загружаем клиентов с ролью "Заказчик"

string query = "SELECT userID, lastName + ' ' + firstName AS clientName FROM Users WHERE userType = 'Заказчик'";

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlCommand cmd = new SqlCommand(query, conn);

conn.Open();

SqlDataReader reader = cmd.ExecuteReader();

while (reader.Read())

{

ComboBoxItem item = new ComboBoxItem

{

Content = reader["clientName"].ToString(),

Tag = reader["userID"]

};

ClientComboBox.Items.Add(item);

}

}

}

private void LoadClimateTechTypes()

{

// Загружаем типы оборудования

string query = "SELECT techTypeID, techTypeName FROM climate\_tech\_types";

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlCommand cmd = new SqlCommand(query, conn);

conn.Open();

SqlDataReader reader = cmd.ExecuteReader();

while (reader.Read())

{

ComboBoxItem item = new ComboBoxItem

{

Content = reader["techTypeName"].ToString(),

Tag = reader["techTypeID"]

};

ClimateTechTypeComboBox.Items.Add(item);

}

}

}

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

{

if (ClimateTechTypeComboBox.SelectedItem != null)

{

int techTypeID = (int)((ComboBoxItem)ClimateTechTypeComboBox.SelectedItem).Tag;

LoadModelsForType(techTypeID);

}

}

private void LoadModelsForType(int techTypeID)

{

// Загружаем модели оборудования для выбранного типа

string query = "SELECT modelID, modelName FROM climate\_tech\_models WHERE techTypeID = @techTypeID";

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@techTypeID", techTypeID);

conn.Open();

SqlDataReader reader = cmd.ExecuteReader();

ModelComboBox.Items.Clear(); // Очищаем предыдущие модели

while (reader.Read())

{

ComboBoxItem item = new ComboBoxItem

{

Content = reader["modelName"].ToString(),

Tag = reader["modelID"]

};

ModelComboBox.Items.Add(item);

}

}

}

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

{

string climateTechType = ((ComboBoxItem)ClimateTechTypeComboBox.SelectedItem).Content.ToString();

string model = ((ComboBoxItem)ModelComboBox.SelectedItem)?.Content.ToString();

string problemDescription = ProblemDescriptionTextBox.Text;

string startDate = StartDateDatePicker.SelectedDate?.ToString("yyyy-MM-dd");

int clientID = (int)((ComboBoxItem)ClientComboBox.SelectedItem).Tag;

if (string.IsNullOrEmpty(model) || string.IsNullOrEmpty(problemDescription) || string.IsNullOrEmpty(startDate))

{

MessageBox.Show("Пожалуйста, заполните все обязательные поля!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

return;

}

// Вставка в базу данных

string query = @"

INSERT INTO Requests (startDate, climateTechType, climateTechModel, problemDescription, requestStatus, clientID, modelID, techTypeID)

VALUES (@startDate, @climateTechType, @climateTechModel, @problemDescription, 'Новая заявка', @clientID, @modelID, @techTypeID)";

int modelID = (int)((ComboBoxItem)ModelComboBox.SelectedItem).Tag;

int techTypeID = (int)((ComboBoxItem)ClimateTechTypeComboBox.SelectedItem).Tag;

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@startDate", startDate);

cmd.Parameters.AddWithValue("@climateTechType", climateTechType);

cmd.Parameters.AddWithValue("@climateTechModel", model);

cmd.Parameters.AddWithValue("@problemDescription", problemDescription);

cmd.Parameters.AddWithValue("@clientID", clientID);

cmd.Parameters.AddWithValue("@modelID", modelID);

cmd.Parameters.AddWithValue("@techTypeID", techTypeID);

try

{

conn.Open();

cmd.ExecuteNonQuery();

MessageBox.Show("Заявка успешно добавлена!");

this.Close();

}

catch (Exception ex)

{

MessageBox.Show("Ошибка: " + ex.Message);

}

}

}

}

}