Zadanie

Zadanie w C# polega na stworzeniu aplikacji WPF, która pozwala na przechowywanie i zarządzanie zgłoszeniami awarii użytkowników w bazie danych MySQL lub MS SQL. Aplikacja powinna umożliwiać użytkownikom:

- wyświetlanie wszystkich zgłoszeń z bazy danych,
- sprawdzenie, czy zadanie jest już przypisane do kogoś,
- możliwość usunięcia zadania,
- oznaczenie zadania jako wykonane.

Przykład funkcjonalnej bazy danych:

```
CREATE TABLE Zgloszenia (
ID int(11) NOT NULL AUTO_INCREMENT,
Opis varchar(255) DEFAULT NULL,
Przypisane varchar(255) DEFAULT NULL,
Wykonane tinyint(1) NOT NULL DEFAULT '0',
PRIMARY KEY (ID)
)
```

Szkielet kodu aplikacji WPF:

a) kod okna WPF (MainWindow.xaml):

```
<Window x:Class="ZgloszeniaAwarii.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="Zgłoszenia awarii" Height="350" Width="525">
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
        </Grid.RowDefinitions>

        <StackPanel Orientation="Horizontal" Margin="10">
            <Label Content="Nazwa użytkownika:"/>
            <TextBox Name="tbNazwaUzytkownika" Margin="10" Width="200"/>
        </StackPanel>

        <StackPanel Orientation="Horizontal" Margin="10">
            <Label Content="Opis awarii:"/>
            <TextBox Name="tbOpisAwarii" Margin="10" Width="200"/>
        </StackPanel>

        <Button Name="btnDodajZgloszenie" Margin="10" Width="100" Height="30"
Content="Dodaj zgłoszenie" Click="btnDodajZgloszenie_Click"/>

        <DataGrid Grid.Row="1" Name="dgZgloszenia" AutoGenerateColumns="False">
            <DataGrid.Columns>
                <DataGridTextColumn Header="ID" Binding="{Binding Path=Id}"/>
```

```
            <DataGridTextColumn Header="Nazwa użytkownika" Binding="{Binding
Path=NazwaUzytkownika}"/>
            <DataGridTextColumn Header="Opis awarii" Binding="{Binding Path=Opis}"/>
            <DataGridTextColumn Header="Data dodania" Binding="{Binding
Path=DataDodania}"/>
            <DataGridTextColumn Header="Przydzielone do" Binding="{Binding
Path=PrzydzieloneDo}"/>
            <DataGridCheckBoxColumn Header="Wykonane" Binding="{Binding
Path=Wykonane}"/>
            <DataGridTemplateColumn Header="Usuń">
               <DataGridTemplateColumn.CellTemplate>
                  <DataTemplate>
                     <Button Content="Usuń" Click="btnUsunZgloszenie_Click"/>
                  </DataTemplate>
               </DataGridTemplateColumn.CellTemplate>
            </DataGridTemplateColumn>
         </DataGrid.Columns>
      </DataGrid>
   </Grid>
</Window>
```

b) kod do aplikacji:

```
using System;
using System.Collections.Generic;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;

namespace ZgloszeniaAwarii
{
   public partial class MainWindow : Window
   {
      private string connectionString;

      public MainWindow()
      {
         InitializeComponent();
         connectionString =
ConfigurationManager.ConnectionStrings["DefaultConnection"].ConnectionString;
         LoadZgloszenia();
      }

      private void btnDodajZgloszenie_Click(object sender, RoutedEventArgs e)
      {
         string nazwaUzytkownika = tbNazwaUzytkownika.Text;
         string opisAwarii = tbOpisAwarii.Text;
```

```
            if (string.IsNullOrEmpty(nazwaUzytkownika) || string.IsNullOrEmpty(opisAwarii))
            {
                MessageBox.Show("Nazwa użytkownika i opis awarii nie mogą być puste.");
                return;
            }

            using (SqlConnection connection = new SqlConnection(connectionString))
            {
                connection.Open();
                SqlCommand command = new SqlCommand("INSERT INTO zgloszenia
(nazwa_uzytkownika, opis) VALUES (@nazwaUzytkownika, @opisAwarii)", connection);
                command.Parameters.AddWithValue("@nazwaUzytkownika", nazwaUzytkownika);
                command.Parameters.AddWithValue("@opisAwarii", opisAwarii);
                command.ExecuteNonQuery();
            }

            LoadZgloszenia();
        }

        private void btnUsunZgloszenie_Click(object sender, RoutedEventArgs e)
        {
            DataRowView row = (DataRowView)((Button)e.Source).DataContext;
            int id = (int)row["Id"];

            using (SqlConnection connection = new SqlConnection(connectionString))
            {
                connection.Open();
                SqlCommand command = new SqlCommand("DELETE FROM zgloszenia WHERE id =
@id", connection);
                command.Parameters.AddWithValue("@id", id);
                command.ExecuteNonQuery();
            }

            LoadZgloszenia();
        }

        private void LoadZgloszenia()
        {
            using (SqlConnection connection = new SqlConnection(connectionString))
            {
                connection.Open();
                SqlDataAdapter adapter = new SqlDataAdapter("SELECT * FROM zgloszenia",
connection);
                DataTable zgloszeniaTable = new DataTable();
                adapter.Fill(zgloszeniaTable);
                dgZgloszenia.ItemsSource = zgloszeniaTable.DefaultView;
            }
        }
    }
}
```

W pierwszej kolejności należy zaimplementować podany kod w taki sposób, by program był w pełni funkcjonalny. Wszelkie napotkane problemy muszą zostać wyeliminowane.

WYKONANIE TEJ CZĘŚCI ZADANIA W 100% - OCENA DOPUSZCZAJĄCY

Kod bazy należy zmienić w taki sposób by:

- pole Przypisane odpowiadało tabeli, w której będziemy mieli co najmniej pola imię, nazwisko
- dołożyć tabele Kategorie, w której powinny znaleźć się co najmniej pola: nazwa, opis
- następnie należy dokonać odpowiednich połączeń pomiędzy tabelami (relacje)
- należy poprawić odwołania do bazy danych w programie by zachował swoją funkcjonalność

WYKONANIE TEJ CZĘŚCI ZADANIA W 100% MAJĄC WYKONANE POPRZEDNIE CZĘŚCI - OCENA DOSTATECZNY

Aplikację należy zmodyfikować o następujące podpunkty:

- utworzyć nowe klasy (odpowiednio Kategoria, Osoba, Zgłoszenie), które będą odwzorowywały każdą z tabel
- należy tak zmodyfikować aplikację, by mogła korzystać z z utworzonych klas, zaś funkcjonalność aplikacji powinna zostać niezmienna

WYKONANIE TEJ CZĘŚCI ZADANIA W 100% MAJĄC WYKONANE POPRZEDNIE CZĘŚCI - OCENA DOBRY

Dodatkowe modyfikacje aplikacji:

- dodawanie nowych informacji do bazy danych powinno odbywać się w osobnych widokach/stanach okna aplikacji, w dedykowanych formularzach
- formularze powinny sprawdzać, czy podane dane są wprowadzane poprawnie (minimum sprawdzenia powinno tyczyć się wielkich liter imion i nazwisk pracowników)

WYKONANIE TEJ CZĘŚCI ZADANIA W 100% MAJĄC WYKONANE POPRZEDNIE CZĘŚCI - OCENA BARDZO DOBRY

Wszelkie dodatkowe modyfikacje kodu, jak np. implementacja czasu przyjęcia zgłoszenia, czasu rozwiązania, dodatnia osoby zgłaszającej itp. poskutkuje otrzymaniem oceny CELUJĄCY.