**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ**

**УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №6-7

по дисциплине: «Объектно-ориентированное проектирование и программирование»

на тему: **«**Коллекции. Обобщенные типы.

Технология доступа к данным *ADO.NET*.»

Выполнил: студент гр. ИТИ-21

Тарканов А. В.

Принял: преподаватель

Башаримов Ю. С.

Гомель 2024

**Цель работы**: изучение обобщений, итераторов, коллекций и технологии доступа к данным *ADO.NET*.

|  |  |
| --- | --- |
| № | Условие |
| 17. | Справочник в библиотеке: название книги, автор(ы), ***жанр***, год издания, количество страниц, количество книг в библиотеке |

* На основании предметной области разработать БД, состоящую из 3-5 таблиц (*кол-во таблиц и схему данных предварительно согласовать с преподавателем*). Заполнить каждую таблицу 20-30 записями.
* Реализовать CRUD операции используя технологию ADO.NET. ***В этой лабораторной работе запрещено использовать компоненты, которые скрывают работу с ADO.NET.***
* При реализации ***обязательно*** использовать **обобщенные интерфейсы методы и классы** (*в рамках темы 6ой лабораторной работы*)
* Разработать GUI. Приложение должно быть простым в использовании и включать в себя полную обработку исключений. Обязательно использовать подгрузку данных в элементы управления из справочных таблиц (Например: выпадающие списки).
* Написать *unit*-тесты для тестирования разработанных библиотечных классов, тестирование должно покрывать более 80% библиотечного кода.
* При написании и оформлении кода **обязательно** руководствоваться **Code Convention, принципами ООП, SOLID** и использовать элементы авто документирования с генерацией соответствующих файлов.

**Ход работы**

При запуске программы происходит создание таблиц в базе данных определенной структуры и пользователю вылазит окно.

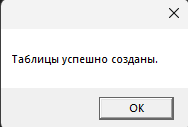


Рисунок 1 – Создание таблиц

Описание таблиц и их структура:

1. Таблица «*books*»:

• Идентификатор книги (*Primary* *Key*)

• Название книги

• Год издания

• Количество страниц

• Идентификатор жанра

• Айди автора

• Количество книг в библиотеке

2. Таблица «*authors*»:

• Идентификатор автора (*Primary* *Key*)

• Имя автора

• Фамилия автора

• Дата рождения автора

3. Таблица «*genres*»:

• Идентификатор жанра (*Primary* *Key*)

• Название жанра

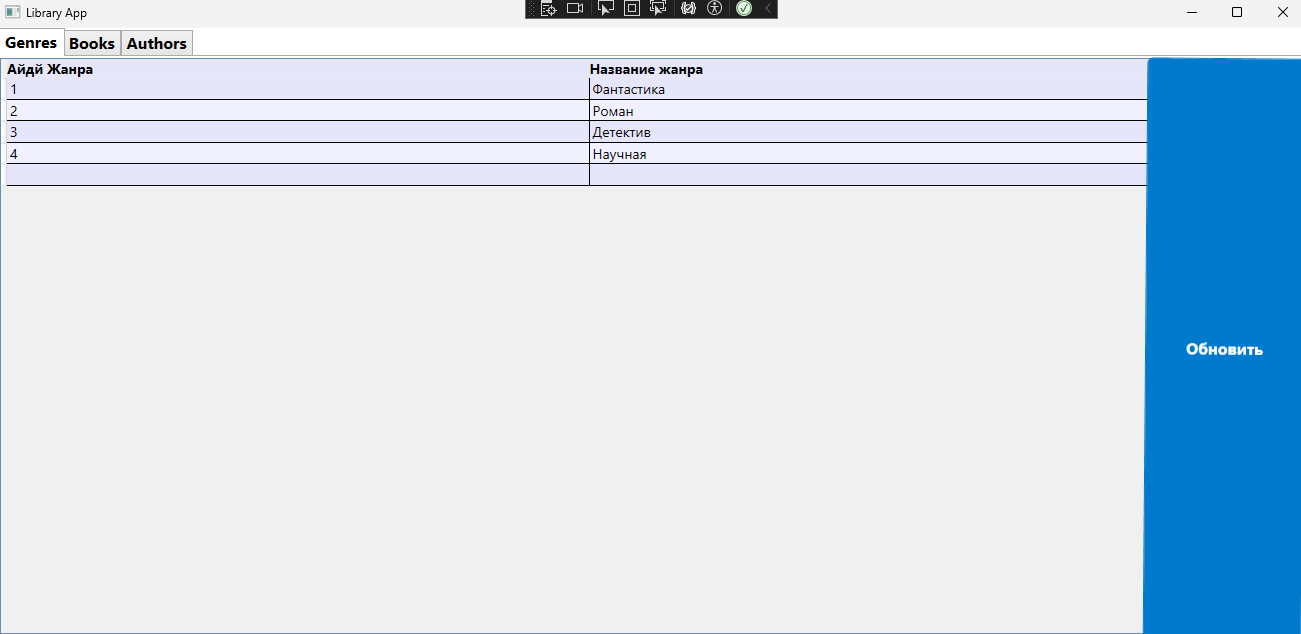


Рисунок 2 – Пользовательский интерфейс

На рисунке 2 изображен пользовательский интерфейс. В пользовательском интерфейсе мы видим, что у нас есть три вкладки с созданными таблицами, между которыми можно переключаться, также у нас есть кнопка «Обновить», при нажатии на которую происходит сохранение действий, которые вы сделали в таблицах. Таблицу можно редактировать, можно создавать и удалять новые записи, но перед тем, как добавить она должна быть валидна.

На рисунке 3 можно увидеть, что при вводе неправильного формата года ячейка подсвечивается красным, а также вылазит окно, которое предупреждает, что неверный формат года.

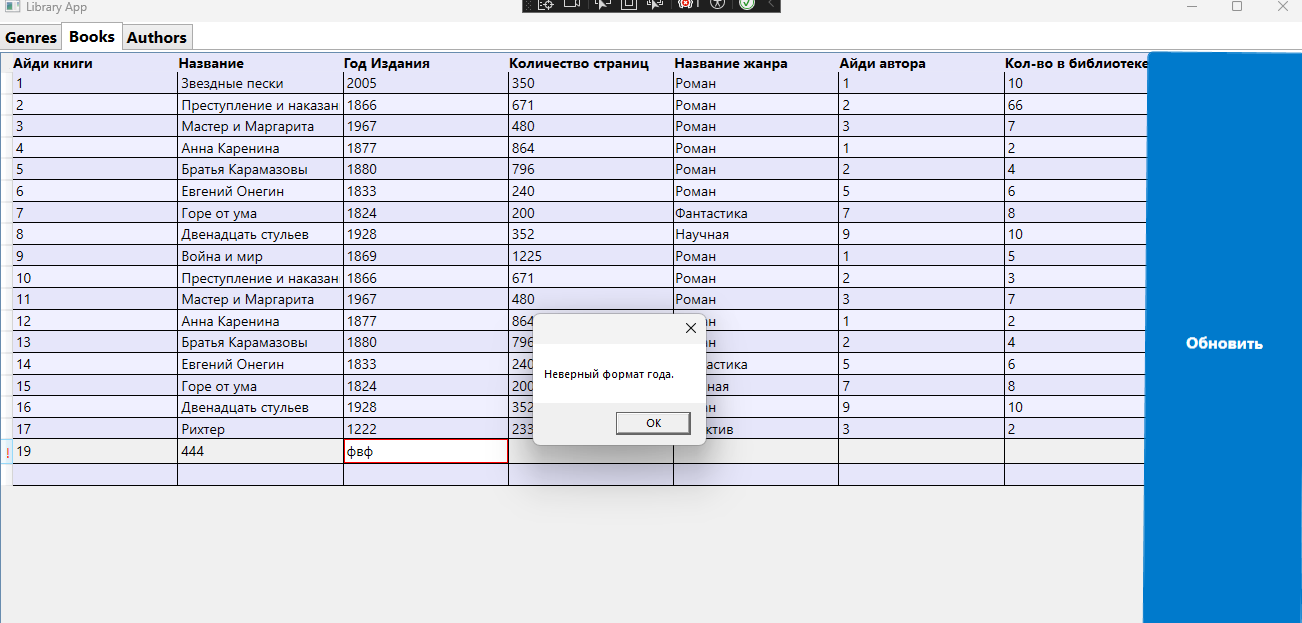


Рисунок 3 – Ошибка валидации

На рисунке 4 видно, что выбор жанра в программе представлен в виде комбо бокса с выбором четырех жанров на выбор.



Рисунок 4 – комбо бокс

На рисунке 5 можно увидеть, что при заполнении всех ячеек правильно, добавляется книга в нашу базу данных, чтобы понять что добавление успешно, вылазит окно о уведомлении , что все изменения сохранены.

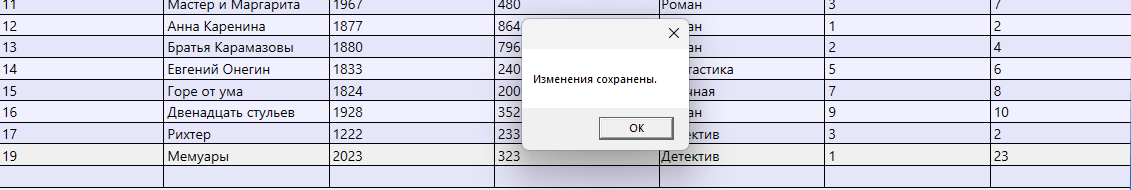


Рисунок 5 – Добавление книги

На рисунке 6 можно увидеть, что добавленная книга была добавлена в нашу базу данных *librarydb*.

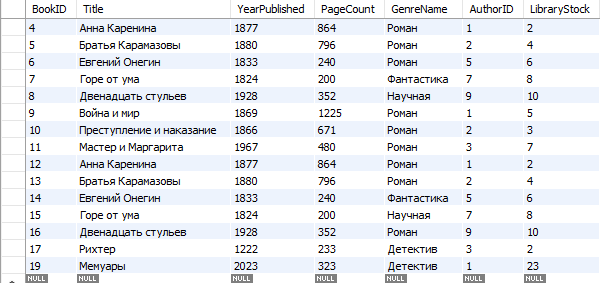


Рисунок 6 – Успешное изменение

Также можно удалять, редактировать текущие данные, как в таблице *books*, так и в таблице *authors* и *genres.*

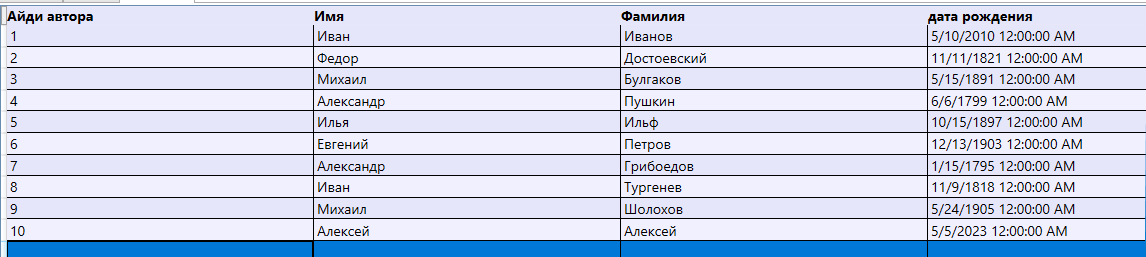


Рисунок 7– Добавление в таблицу авторов

На рисунке 8 представлены результаты *unit*-тестов.

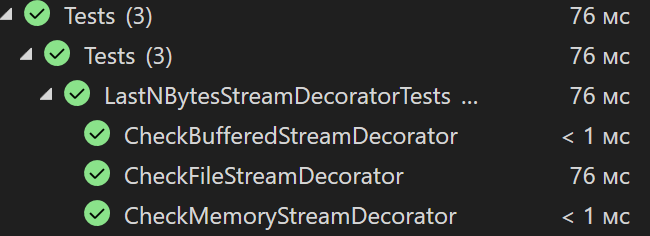


Рисунок 8 – Тестирование

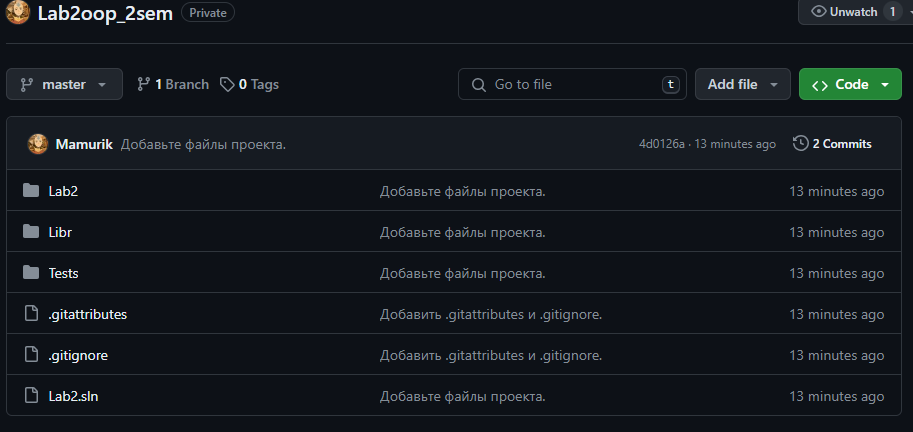


Рисунок 9 – *Github*

В приложении А представлен код программы.

**Вывод:** в результате выполнения лабораторной работы разработали приложение *wpf*, которое может взаимодействовать с базой данных и редактировать таблицы, а также углубили знания итераторов, обобщенных типов и методов.

**ПРИЛОЖЕНИЕ А**

(обязательное)

**Текст программы**

**MainWindow.Xaml.cs**

using MySql.Data.MySqlClient;

using System;

using System.Data;

using System.Windows;

using System.Windows.Controls;

namespace WpfApp2

{

public partial class MainWindow : Window

{

private const string connectionString = "server=localhost;database=librarydb;user=root;password=alex2005";

public MainWindow()

{

InitializeComponent();

CreateTables();

LoadGenres();

LoadBooks();

LoadAuthors();

}

private void CreateTables()

{

using (MySqlConnection connection = new MySqlConnection(connectionString))

{

try

{

connection.Open();

MySqlCommand cmd = connection.CreateCommand();

// Create the Genres table if it doesn't exist

cmd.CommandText = @"CREATE TABLE IF NOT EXISTS Genres (

GenreID INT PRIMARY KEY AUTO\_INCREMENT,

GenreName VARCHAR(255)

)";

cmd.ExecuteNonQuery();

// Create the Authors table if it doesn't exist

cmd.CommandText = @"CREATE TABLE IF NOT EXISTS Authors (

AuthorID INT PRIMARY KEY AUTO\_INCREMENT,

FirstName VARCHAR(255),

LastName VARCHAR(255),

BirthDate DATE

)";

cmd.ExecuteNonQuery();

// Create the Books table if it doesn't exist

cmd.CommandText = @"CREATE TABLE IF NOT EXISTS Books (

BookID INT PRIMARY KEY AUTO\_INCREMENT,

Title VARCHAR(255),

YearPublished INT,

PageCount INT,

GenreName VARCHAR(255),

FOREIGN KEY (AuthorID) REFERENCES Authors(AuthorID),

LibraryStock INT

)";

cmd.ExecuteNonQuery();

MessageBox.Show("Таблицы успешно созданы.");

}

catch (Exception ex)

{

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

}

}

}

private void LoadData<T>(string tableName, DataGrid dataGrid)

{

using (MySqlConnection connection = new MySqlConnection(connectionString))

{

connection.Open();

MySqlCommand cmd = new MySqlCommand($"SELECT \* FROM {tableName}", connection);

MySqlDataAdapter adapter = new MySqlDataAdapter(cmd);

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

dataGrid.ItemsSource = dataTable.DefaultView;

}

}

private void LoadGenres()

{

LoadData<Genre>("Genres", genresDataGrid);

}

private void LoadBooks()

{

LoadData<Book>("Books", booksDataGrid);

}

private void LoadAuthors()

{

LoadData<Author>("Authors", authorsDataGrid);

}

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

{

SaveChanges();

}

private void SaveChanges()

{

DataTable genresDataTable = ((DataView)genresDataGrid.ItemsSource).Table;

DataTable booksDataTable = ((DataView)booksDataGrid.ItemsSource).Table;

DataTable authorsDataTable = ((DataView)authorsDataGrid.ItemsSource).Table;

if (!DataValidator<Genre>.ValidateData(genresDataTable) ||

!DataValidator<Book>.ValidateData(booksDataTable) ||

!DataValidator<Author>.ValidateData(authorsDataTable))

{

return; // Stop saving changes if data is invalid

}

using (MySqlConnection connection = new MySqlConnection(connectionString))

{

try

{

connection.Open();

MySqlDataAdapter genresAdapter = new MySqlDataAdapter("SELECT \* FROM Genres", connection);

MySqlCommandBuilder genresCmdBuilder = new MySqlCommandBuilder(genresAdapter);

genresAdapter.Update(genresDataTable);

MySqlDataAdapter booksAdapter = new MySqlDataAdapter("SELECT \* FROM Books", connection);

MySqlCommandBuilder booksCmdBuilder = new MySqlCommandBuilder(booksAdapter);

booksAdapter.Update(booksDataTable);

MySqlDataAdapter authorsAdapter = new MySqlDataAdapter("SELECT \* FROM Authors", connection);

MySqlCommandBuilder authorsCmdBuilder = new MySqlCommandBuilder(authorsAdapter);

authorsAdapter.Update(authorsDataTable);

MessageBox.Show("изменения сохранены.");

}

catch (Exception ex)

{

MessageBox.Show("ошибка сохранения: " + ex.Message);

}

}

}

}

}

**book.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace WpfApp2

{

public class Book

{

public int BookID { get; set; }

public string Title { get; set; }

public int YearPublished { get; set; }

public int PageCount { get; set; }

public string GenreName { get; set; }

public int AuthorID { get; set; }

public int LibraryStock { get; set; }

}

**Datavalidator.cs:**

using System;

using System.Data;

using System.Windows;

namespace WpfApp2

{

public static class DataValidator<T>

{

public static bool ValidateData(DataTable dataTable)

{

if (dataTable != null)

{

foreach (DataRow row in dataTable.Rows)

{

if (row.RowState == DataRowState.Added || row.RowState == DataRowState.Modified)

{

if (!ValidateRow(row))

return false;

}

}

}

return true;

}

private static bool ValidateRow(DataRow row)

{

if (typeof(T) == typeof(Genre))

return ValidateGenresRow(row);

else if (typeof(T) == typeof(Book))

return ValidateBooksRow(row);

else if (typeof(T) == typeof(Author))

return ValidateAuthorsRow(row);

return true;

}

private static bool ValidateGenresRow(DataRow row)

{

if (string.IsNullOrEmpty(row["GenreName"] as string))

{

MessageBox.Show("Введите название жанра.");

return false;

}

string genreName = row["GenreName"].ToString();

// Проверка на допустимые значения жанров

if (genreName != "Роман" && genreName != "Детектив" && genreName != "Научная" && genreName != "Фантастика")

{

MessageBox.Show("Неверное значение жанра.");

return false;

}

return true;

}

private static bool ValidateBooksRow(DataRow row)

{

if (string.IsNullOrEmpty(row["Title"] as string))

{

MessageBox.Show("Введите название книги.");

return false;

}

if (!int.TryParse(row["YearPublished"].ToString(), out \_))

{

MessageBox.Show("Неверный формат года публикации.");

return false;

}

if (!int.TryParse(row["PageCount"].ToString(), out \_))

{

MessageBox.Show("Неверный формат количества страниц.");

return false;

}

if (row["GenreName"] == DBNull.Value)

{

MessageBox.Show("Выберите жанр.");

return false;

}

Genre genre;

if (!Enum.TryParse(row["GenreName"].ToString(), out genre))

{

MessageBox.Show("Неверное значение жанра.");

return false;

}

if (!int.TryParse(row["AuthorID"].ToString(), out \_))

{

MessageBox.Show("Неверный формат идентификатора автора.");

return false;

}

if (!int.TryParse(row["LibraryStock"].ToString(), out \_))

{

MessageBox.Show("Неверный формат количества книг в библиотеке.");

return false;

}

return true;

}

private static bool ValidateAuthorsRow(DataRow row)

{

if (string.IsNullOrEmpty(row["FirstName"] as string))

{

MessageBox.Show("Введите имя автора.");

return false;

}

if (string.IsNullOrEmpty(row["LastName"] as string))

{

MessageBox.Show("Введите фамилию автора.");

return false;

}

if (!DateTime.TryParse(row["BirthDate"].ToString(), out \_))

{

MessageBox.Show("Неверный формат даты рождения.");

return false;

}

return true;

}

}

}

**author.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace WpfApp2

{

public class Author

{

public int AuthorID { get; set; }

public string FirstName { get; set; }

public string LastName { get; set; }

public DateTime BirthDate { get; set; }

}

}

**genre.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace WpfApp2

{

public static class GenreValues

{

public static Genre[] Values => (Genre[])Enum.GetValues(typeof(Genre));

}

public enum Genre

{

Роман,

Детектив,

Научная,

Фантастика

}

}

**MainWindow.Xaml:**

<Window x:Class="WpfApp2.MainWindow"

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

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

xmlns:local="clr-namespace:WpfApp2"

xmlns:sys="clr-namespace:System;assembly=mscorlib"

Title="Library App" Height="523" Width="1386">

<Window.Resources>

<Style TargetType="TabItem">

<Setter Property="HeaderTemplate">

<Setter.Value>

<DataTemplate>

<TextBlock Text="{Binding}" FontSize="16" FontWeight="Bold" />

</DataTemplate>

</Setter.Value>

</Setter>

</Style>

<Style TargetType="Button" x:Key="UpdateButtonStyle">

<Setter Property="Margin" Value="548,298,86,81" />

<Setter Property="ContentTemplate">

<Setter.Value>

<DataTemplate>

<TextBlock Text="{Binding}" FontSize="14" FontWeight="Bold" />

</DataTemplate>

</Setter.Value>

</Setter>

<Setter Property="RenderTransform">

<Setter.Value>

<TransformGroup>

<ScaleTransform ScaleY="1" />

<SkewTransform />

<RotateTransform />

<TranslateTransform />

</TransformGroup>

</Setter.Value>

</Setter>

</Style>

<Style TargetType="DataGrid">

<Setter Property="Margin" Value="0,0,100,0" />

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

<Setter Property="RowBackground" Value="#FFE6E6FA" />

<Setter Property="AlternatingRowBackground" Value="#FFF0F0FF" />

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

<Setter Property="FontFamily" Value="Segoe UI" />

</Style>

<Style x:Key="BeautifulButtonStyle" TargetType="Button">

<Setter Property="Background" Value="#007ACC"/>

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

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

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

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

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

<Setter Property="BorderBrush" Value="#007ACC"/>

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

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="Button">

<Grid>

<Border Name="Border" Background="{TemplateBinding Background}" BorderBrush="{TemplateBinding BorderBrush}"

BorderThickness="{TemplateBinding BorderThickness}" CornerRadius="5"/>

<ContentPresenter HorizontalAlignment="Center" VerticalAlignment="Center"/>

</Grid>

<ControlTemplate.Triggers>

<Trigger Property="IsMouseOver" Value="True">

<Setter TargetName="Border" Property="Background" Value="#005A8D"/>

<Setter TargetName="Border" Property="BorderBrush" Value="#005A8D"/>

</Trigger>

<Trigger Property="IsPressed" Value="True">

<Setter TargetName="Border" Property="Background" Value="#003D61"/>

<Setter TargetName="Border" Property="BorderBrush" Value="#003D61"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

<Setter Property="RenderTransformOrigin" Value="0.5, 0.5"/>

<Setter Property="RenderTransform">

<Setter.Value>

<TransformGroup>

<ScaleTransform/>

<SkewTransform/>

<RotateTransform Angle="0.471"/>

<TranslateTransform/>

</TransformGroup>

</Setter.Value>

</Setter>

</Style>

<Style TargetType="DataGridColumnHeader">

<Setter Property="Background" Value="#FFE6E6FA" />

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

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

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

</Style>

<Style TargetType="TextBlock">

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

<Setter Property="FontFamily" Value="Segoe UI" />

</Style>

<Style TargetType="ComboBox">

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

<Setter Property="FontFamily" Value="Segoe UI" />

<Setter Property="Padding" Value="4,2" />

<Setter Property="MinWidth" Value="120" />

</Style>

</Window.Resources>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*" />

<ColumnDefinition Width="Auto" />

</Grid.ColumnDefinitions>

<TabControl Grid.Column="0">

<TabItem Header="Genres">

<DataGrid x:Name="genresDataGrid" Margin="0,0,130,0">

<DataGrid.Columns>

<DataGridTextColumn Header="Айдй Жанра" Binding="{Binding GenreID}" Width="\*" />

<DataGridTextColumn Header="Название жанра" Binding="{Binding GenreName}" Width="\*" />

</DataGrid.Columns>

</DataGrid>

</TabItem>

<TabItem Header="Books">

<DataGrid x:Name="booksDataGrid" Margin="0,0,217,0">

<DataGrid.Columns>

<DataGridTextColumn Header="Айди книги" Binding="{Binding BookID}" Width="\*" />

<DataGridTextColumn Header="Название" Binding="{Binding Title}" Width="\*" />

<DataGridTextColumn Header="Год Издания" Binding="{Binding YearPublished}" Width="\*" />

<DataGridTextColumn Header="Количество страниц" Binding="{Binding PageCount}" Width="\*" />

<DataGridTemplateColumn Header="Название жанра" Width="\*">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding GenreName}" />

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

<DataGridTemplateColumn.CellEditingTemplate>

<DataTemplate>

<ComboBox ItemsSource="{Binding Source={x:Static local:GenreValues.Values}}"

SelectedItem="{Binding GenreName}" />

</DataTemplate>

</DataGridTemplateColumn.CellEditingTemplate>

</DataGridTemplateColumn>

<DataGridTextColumn Header="Айди автора" Binding="{Binding AuthorID}" Width="\*" />

<DataGridTextColumn Header="Кол-во в библиотеке" Binding="{Binding LibraryStock}" Width="\*" />

</DataGrid.Columns>

</DataGrid>

</TabItem>

<TabItem Header="Authors">

<DataGrid x:Name="authorsDataGrid" Margin="0,0,128,0">

<DataGrid.Columns>

<DataGridTextColumn Header="Айди автора" Binding="{Binding AuthorID}" Width="\*" />

<DataGridTextColumn Header="Имя " Binding="{Binding FirstName}" Width="\*" />

<DataGridTextColumn Header="Фамилия" Binding="{Binding LastName}" Width="\*" />

<DataGridTextColumn Header="дата рождения" Binding="{Binding BirthDate }" Width="\*" />

</DataGrid.Columns>

</DataGrid>

</TabItem>

</TabControl>

<Button Content="Обновить" Click="Update\_Click" Style="{StaticResource BeautifulButtonStyle}" Margin="1148,30,2,-1"/>

</Grid>

</Window>