using System.Data.SqlClient;

namespace UI

{

class Program

{

private const String connectionString = @"Data Source=(localdb)\MSSQLLocalDB;Initial Catalog=BaseDeDatos-Agenda;Integrated Security=True;Connect Timeout=30;Encrypt=False;TrustServerCertificate=False;ApplicationIntent=ReadWrite;MultiSubnetFailover=False";

static void Main(string[] args)

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

#region Declara variablaes

DataSet dataSet;

SqlCommand select;

SqlDataAdapter dataAdapter;

SqlCommandBuilder commandBuilder;

// Crea un DataSet

dataSet = new DataSet();

// Genera el select que depues se le pasara al adapter

select = new SqlCommand("SELECT nombre, fechaCreacion, activo FROM Agendas", connection);

// Crea el adapter seguido del fill (metodo para refrescar el Dataset para que concuerdo con la Tabla en Base de Datos)

dataAdapter = new SqlDataAdapter(select);

dataAdapter.Fill(dataSet, "Agendas");

// Genera automaticamente comandos para la tabla

commandBuilder = new SqlCommandBuilder(dataAdapter);

#endregion

#region Comprueba las sentencias SQL generadas

/\*

Console.WriteLine("UPDATE : ");

Console.WriteLine("==================================================");

Console.WriteLine(commandBuilder.GetUpdateCommand().CommandText);

Console.WriteLine(" ");

Console.WriteLine("INSERT : ");

Console.WriteLine("==================================================");

Console.WriteLine(commandBuilder.GetInsertCommand().CommandText);

Console.WriteLine(" ");

Console.WriteLine("DELETE : ");

Console.WriteLine("==================================================");

Console.WriteLine(commandBuilder.GetDeleteCommand().CommandText);

Console.WriteLine(" ");

Console.ReadKey();

\*/

#endregion

#region Menu

Boolean salir = false;

do

{

UInt32 opcion;

int opcionModificar;

int opcionEliminar;

// Una variable para representar la Collection de Rows

DataRowCollection rowCollection = dataSet.Tables["Agendas"].Rows;

Console.WriteLine("1 -- Ver Agendas");

Console.WriteLine("2 -- Agregar Agenda");

Console.WriteLine("3 -- Modificar Agenda");

Console.WriteLine("4 -- Borrar Agenda");

Console.WriteLine("0 -- Salir");

Console.WriteLine("");

Console.WriteLine("Ingrese una opcion");

Boolean esValido = UInt32.TryParse(Console.ReadLine(), out opcion);

if (esValido && opcion >= 0 && opcion <= 4)

{

switch (opcion)

{

case 1:

Console.Clear();

#region Muestra las Agendas

do

{

for (int i = 1; i < rowCollection.Count; i++)

{

Console.WriteLine(i + " --- " + dataSet.Tables["Agendas"].Rows[i]["nombre"]);

}

Console.WriteLine("");

Console.WriteLine("Ingrese cualquier cosa para salir");

} while (Console.ReadLine() == "");

#endregion

Console.Clear();

break;

case 2:

Console.Clear();

#region Agregar una Fila

DataTable Agendas = dataSet.Tables["Agendas"];

DataRow row = Agendas.NewRow();

Console.WriteLine("Ingrese un Nombre para su Agenda");

String nombre;

Boolean existeNombre = false;

do

{

nombre = Console.ReadLine();

Console.WriteLine("");

foreach (DataRow rowBuscar in rowCollection)

{

if (rowBuscar["nombre"].ToString() == nombre)

{

existeNombre = true;

Console.WriteLine("Ese nombre ya existe, ingrese uno nuevo");

}

}

} while (existeNombre);

row["nombre"] = nombre;

row["fechaCreacion"] = DateTime.Now.Date;

row["activo"] = true;

Agendas.Rows.Add(row);

dataAdapter.Update(dataSet, "Agendas");

#endregion

Console.Clear();

break;

case 3:

Console.Clear();

#region Modificar Fila

#region Ver Agendas

for (int i = 1; i < rowCollection.Count; i++)

{

Console.WriteLine(i + " --- " + dataSet.Tables["Agendas"].Rows[i]["nombre"]);

}

#endregion

#region Seleccionar y Modificar Agenda

do

{

Console.WriteLine("");

Console.WriteLine("Ingrese la fila a modificar");

opcionModificar = int.Parse(Console.ReadLine());

DataRow selectedRowUpdate = rowCollection[opcionModificar];

Console.Clear();

Console.WriteLine("El nombre actual es " + selectedRowUpdate["nombre"]);

Console.WriteLine("");

Console.WriteLine("Ingrese un Nuevo nombre");

String nombreNuevo;

Boolean existe = false;

do

{

nombreNuevo = Console.ReadLine();

Console.WriteLine("");

foreach (DataRow rowBuscar in rowCollection)

{

if (rowBuscar["nombre"].ToString() == nombreNuevo)

{

existe = true;

Console.WriteLine("Ese nombre ya existe, ingrese uno nuevo");

}

}

} while (existe);

selectedRowUpdate.BeginEdit();

selectedRowUpdate["nombre"] = nombreNuevo;

selectedRowUpdate.EndEdit();

dataAdapter.Update(dataSet, "Agendas");

} while (opcionModificar < rowCollection.Count && opcionModificar >= rowCollection.Count);

#endregion

#endregion

Console.Clear();

break;

case 4:

Console.Clear();

#region Borrar Fila

#region Ver Agendas

for (int i = 1; i < rowCollection.Count; i++)

{

Console.WriteLine(i + " --- " + dataSet.Tables["Agendas"].Rows[i]["nombre"]);

}

#endregion

#region Seleccionar y Borrar Agenda

do

{

Console.WriteLine("");

Console.WriteLine("Ingrese la fila a borrar");

opcionEliminar = int.Parse(Console.ReadLine());

DataRow selectedRowDelete = rowCollection[opcionEliminar];

Console.WriteLine("Ingrese el nombre a colocar");

selectedRowDelete.Delete();

dataAdapter.Update(dataSet, "Agendas");

dataAdapter.Update(dataSet, "Agendas");

} while (opcionEliminar < rowCollection.Count && opcionEliminar >= rowCollection.Count);

#endregion

#endregion

Console.Clear();

break;

case 0:

salir = true;

break;

}

}

else

{

Console.WriteLine("");

Console.WriteLine("Seleccione una opcion correcta");

Console.WriteLine("");

}

} while (!salir);

#endregion

}

}

}

}