# Sistema Bibliotecario básico

Este es el primer formulario de menú del sistema bibliotecario

Texto

El contenido generado por IA puede ser incorrecto.

Este es su código:

namespace Biblioteca

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void groupBox1\_Enter(object sender, EventArgs e)

{

}

private void label2\_Click(object sender, EventArgs e)

{

}

private void button1\_Click(object sender, EventArgs e)

{

}

private void button5\_Click(object sender, EventArgs e)

{

}

private void btnLibros\_Click(object sender, EventArgs e)

{

frmLibros frmLibros = new frmLibros();

frmLibros.Show();

}

private void btnLectores\_Click(object sender, EventArgs e)

{

frmGestionLectores frmLectores = new frmGestionLectores();

frmLectores.Show();

}

private void btnPrestamos\_Click(object sender, EventArgs e)

{

frmPrestamos frmPrestamos = new frmPrestamos();

frmPrestamos.Show();

}

}

}

Cuando vamos a la sección de Libros nos lleva al registro de un nuevo Libro

Este es el diseño:  
Texto

El contenido generado por IA puede ser incorrecto.

Este es su código:  
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Configuration;

using Microsoft.Data.SqlClient;

namespace Biblioteca

{

public partial class frmLibros : Form

{

// La lógica del constructor y del botón 'btnGuardarLibro' no cambia

private string connectionString = ConfigurationManager.ConnectionStrings["MiConexion"].ConnectionString;

public frmLibros()

{

InitializeComponent();

}

private void frmLibros\_Load(object sender, EventArgs e)

{

CargarAutores();

CargarCategorias();

CargarEditoriales();

}

private void CargarAutores()

{

// Este método es casi identico al de la otra forma.

// En una app más grande, esto estaría en una clase separada (Capa de Datos).

DataTable dt = new DataTable();

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "SELECT id\_autor, nombre FROM tbAutor ORDER BY nombre;";

SqlDataAdapter da = new SqlDataAdapter(query, conn);

da.Fill(dt);

}

// Configuramos el ComboBox para que funcione con los datos

cmbAutores.DataSource = dt;

cmbAutores.DisplayMember = "nombre"; // El texto que ve el usuario

cmbAutores.ValueMember = "id\_autor"; // El valor oculto (el ID)

cmbAutores.SelectedItem = null;

}

private void CargarCategorias()

{

DataTable dt = new DataTable();

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "SELECT id\_categoria, categoria FROM tbCategoria ORDER BY categoria;";

SqlDataAdapter da = new SqlDataAdapter(query, conn);

da.Fill(dt);

}

cmbCategoria.DataSource = dt;

cmbCategoria.DisplayMember = "categoria";

cmbCategoria.ValueMember = "id\_categoria";

cmbCategoria.SelectedItem = null;

}

private void CargarEditoriales()

{

DataTable dt = new DataTable();

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "SELECT id\_editorial, editorial FROM tbEditorial ORDER BY editorial;";

SqlDataAdapter da = new SqlDataAdapter(query, conn);

da.Fill(dt);

}

cmbEditorial.DataSource = dt;

cmbEditorial.DisplayMember = "editorial";

cmbEditorial.ValueMember = "id\_editorial";

cmbEditorial.SelectedItem = null;

}

private void btnNuevoAutor\_Click(object sender, EventArgs e)

{

using (frmGestionAutores formAutores = new frmGestionAutores())

{

// 2. Lo abrimos como un diálogo modal

DialogResult resultado = formAutores.ShowDialog();

if (resultado == DialogResult.OK)

{

MessageBox.Show("!Regresamos al formulario de Libros! Actualizando autores...");

CargarAutores();

cmbAutores.SelectedItem = formAutores.AutorNuevo;

}

}

}

private void btnNuevaCategoria\_Click(object sender, EventArgs e)

{

using (frmGestionCategoria formCategoria = new frmGestionCategoria())

{

DialogResult resultadoCategoria = formCategoria.ShowDialog();

if (resultadoCategoria == DialogResult.OK)

{

MessageBox.Show("!Regresamos al formulario de Libros! Actualizando categorías...");

CargarCategorias();

cmbCategoria.SelectedItem = formCategoria.categoriaNueva;

}

}

}

private void btnNuevoEditorial\_Click(object sender, EventArgs e)

{

using (frmGestionEditorial formEditorial = new frmGestionEditorial())

{

DialogResult resultEditorial = formEditorial.ShowDialog();

if (resultEditorial == DialogResult.OK)

{

MessageBox.Show("!Regresamos al formulario de Libros! Actualizando editorial...");

CargarEditoriales();

cmbEditorial.SelectedItem = formEditorial.EditorialNuevo;

}

}

}

private void btnGuardarLibro\_Click(object sender, EventArgs e)

{

// 1. Validar que los campos no estén vacíos

if (string.IsNullOrEmpty(txtTituloLibro.Text) ||

cmbAutores.SelectedValue == null ||

cmbCategoria.SelectedValue == null ||

cmbEditorial.SelectedValue == null)

{

MessageBox.Show("Por favor, complete todos los campos obligatorios (Título, Autor, Categoría, Editorial).", "Datos Incompletos", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// 2. Obtener los valores de los controles

string titulo = txtTituloLibro.Text;

string ISBN = txtISBN.Text;

int idAutor = (int)cmbAutores.SelectedValue;

int idCategoria = (int)cmbCategoria.SelectedValue;

int idEditorial = (int)cmbEditorial.SelectedValue;

string idioma = txtIdioma.Text;

string paginas = txtPaginas.Text;

// 3. Crear y ejecutar el comando SQL

using (SqlConnection conn = new SqlConnection(connectionString))

{

// Usamos parámetros para seguridad

string query = "INSERT INTO tbLibros (ISBN, titulo, id\_categoria, id\_editorial, idioma, paginas, id\_autor) VALUES (@ISBN, @titulo, @id\_categoria, @id\_editorial, @idioma, @paginas, @id\_autor);";

using (SqlCommand cmd = new SqlCommand(query, conn))

{

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

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

cmd.Parameters.AddWithValue("@id\_categoria", idCategoria);

cmd.Parameters.AddWithValue("@id\_editorial", idEditorial);

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

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

cmd.Parameters.AddWithValue("@id\_autor", idAutor);

try

{

conn.Open();

cmd.ExecuteNonQuery();

MessageBox.Show("Libro guardado con éxito.", "Exito", MessageBoxButtons.OK, MessageBoxIcon.Information);

// Opcional: Limpiar los campos después de gaurdar

txtTituloLibro.Clear();

txtISBN.Clear();

txtIdioma.Clear();

txtPaginas.Clear();

cmbAutores.SelectedItem = null;

cmbCategoria.SelectedItem = null;

cmbEditorial.SelectedItem = null;

}

catch (Exception ex)

{

MessageBox.Show("Error al guardar el libro: " + ex.Message, "Error de Base de Datos", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

}

private void txtTituloLibro\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

txtISBN.Focus();

e.Handled = true;

e.SuppressKeyPress = true;

}

}

private void txtISBN\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

cmbAutores.Focus();

e.Handled = true;

e.SuppressKeyPress = true;

}

}

private void cmbAutores\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

cmbCategoria.Focus();

e.Handled = true;

e.SuppressKeyPress = true;

}

}

private void cmbCategoria\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

cmbEditorial.Focus();

e.Handled = true;

e.SuppressKeyPress = true;

}

}

private void cmbEditorial\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

txtIdioma.Focus();

e.Handled = true;

e.SuppressKeyPress = true;

}

}

private void txtIdioma\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

txtPaginas.Focus();

e.Handled = true;

e.SuppressKeyPress = true;

}

}

private void txtPaginas\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

btnGuardarLibro.PerformClick();

e.Handled = true;

e.SuppressKeyPress = true;

}

}

}

}



Este podemos darle al botón + alado del autor para seleccionar un nuevo autor que no haya sido registrado:

Este es el diseño de este formulario CRUD DE AUTOR:  
Imagen de la pantalla de un celular con letras

El contenido generado por IA puede ser incorrecto.

Y este es su código:  
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Configuration;

using System.Data.SqlClient;

using Microsoft.Data.SqlClient;

namespace Biblioteca

{

public partial class frmGestionAutores : Form

{

private string connectionString = ConfigurationManager.ConnectionStrings["MiConexion"].ConnectionString;

public string AutorNuevo { get; private set; }

private int idAutorSeleccionado = 0;

private string nombreDelAutor = "";

public frmGestionAutores()

{

InitializeComponent();

}

private void frmGestionAutores\_Load(object sender, EventArgs e)

{

CargarAutores();

}

private void CargarAutores()

{

// Este método ahora lee desde la base de datos

var dt = new DataTable();

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "SELECT id\_autor, nombre FROM tbAutor ORDER BY nombre;";

SqlCommand cmd = new SqlCommand(query, conn);

SqlDataAdapter da = new SqlDataAdapter(cmd);

da.Fill(dt);

}

// Asignamos la tabla de datos como fuente del ListBox

lstAutores.DataSource = dt;

lstAutores.DisplayMember = "nombre"; // La columna que se mostrara

}

private void btnGuardar\_Click(object sender, EventArgs e)

{

string nuevoAutor = txtNombreAutor.Text.Trim();

if (string.IsNullOrEmpty(nuevoAutor))

{

MessageBox.Show("Por favor, ingrese un nombre.");

return;

}

// Usamos 'using' para asegurar que la conexión se cierre correctamente

using (SqlConnection conn = new SqlConnection(connectionString))

{

// !Importante! usar parámetros para prevenir inyeccion SQL

string query = "INSERT INTO tbAutor (nombre) VALUES (@nombre);";

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@nombre", nuevoAutor);

try

{

conn.Open();

cmd.ExecuteNonQuery(); // Ejecuta la insercion

this.AutorNuevo = nuevoAutor;

this.DialogResult = DialogResult.OK;

this.Close();

}

catch (Exception ex)

{

MessageBox.Show("Error al guardar el autor: " + ex.Message);

}

}

}

private void lstAutores\_SelectedIndexChanged(object sender, EventArgs e)

{

// 1. Verifica que haya un elemento seleccionado.

if (lstAutores.SelectedItems != null)

{

// 2. Convierte el elemento seleccionado a un DataRowView

DataRowView drv = (DataRowView)lstAutores.SelectedItem;

// 3. Accede al valor de la columna especifica por su nombre.

// Usa el nombre de la columna que contiene el nombre del autor.

this.nombreDelAutor = drv["nombre"].ToString();

// -- Obtener y guardar el ID

this.idAutorSeleccionado = Convert.ToInt32(drv["id\_autor"]);

// 4. Asigna ese valor al TextBox.

txtNombreAutor.Text = nombreDelAutor;

}

else

{

// Si no hay nada seleccionado, reinicia el ID

this.idAutorSeleccionado = 0;

}

}

private void btnEliminar\_Click(object sender, EventArgs e)

{

if (this.idAutorSeleccionado == 0)

{

MessageBox.Show("Por favor, seleccione un autor de la lista para eliminar.");

return;

}

// Pide confirmación al usuario

var confirmacion = MessageBox.Show(

"Estas seguro de que quieres eliminar a este autor?",

"Confirmar Eliminación",

MessageBoxButtons.YesNo,

MessageBoxIcon.Warning

);

if (confirmacion == DialogResult.Yes)

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "DELETE from tbAutor WHERE id\_autor = @id";

using (SqlCommand cmd = new SqlCommand(query, conn))

{

// Usa la variable de la clase que guardo el ID

cmd.Parameters.AddWithValue("@id", this.idAutorSeleccionado);

conn.Open();

cmd.ExecuteNonQuery();

MessageBox.Show("Autor eliminado con éxito.");

}

}

CargarAutores();

}

}

private void btnEditar\_Click(object sender, EventArgs e)

{

if (this.idAutorSeleccionado == 0)

{

MessageBox.Show("Por favor, seleccione un autor de la lista para editar.");

return;

}

string txtAutorseleccionado = txtNombreAutor.Text;

// Pide confirmación al usuario

var confirmacion = MessageBox.Show(

"¿Estas seguro de que quieres editar a este autor?",

"Confirmar Edición",

MessageBoxButtons.YesNo,

MessageBoxIcon.Warning

);

if (confirmacion == DialogResult.Yes)

{

string query = "UPDATE tbAutor SET nombre = @nombre WHERE id\_autor = @id";

using (SqlConnection conn = new SqlConnection(connectionString))

using (SqlCommand cmd = new SqlCommand(query, conn))

{

// Usa la variable de la clase que guardo el ID

cmd.Parameters.AddWithValue("@id", this.idAutorSeleccionado);

cmd.Parameters.AddWithValue("@nombre", txtAutorseleccionado);

try

{

conn.Open();

cmd.ExecuteNonQuery();

MessageBox.Show("Autor editado con éxito.");

}

catch (Exception ex)

{

MessageBox.Show($"Error al actualizar Autor: {ex.Message}");

}

}

CargarAutores();

}

}

}

}

Este mismo podemos C: crear un nuevo autor, R leer todos los autores, U actualizar todos los autores, D eliminar todos los autores que queramos



En la parte de + de Categoría sirve para ingresar a un CRUD de las categorías

Este es su diseño:

Imagen de la pantalla de un celular con letras

El contenido generado por IA puede ser incorrecto.

Y este es su código:  
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Configuration;

using System.Data.SqlClient;

using Microsoft.Data.SqlClient;

namespace Biblioteca

{

public partial class frmGestionCategoria : Form

{

private string connectionString = ConfigurationManager.ConnectionStrings["MiConexion"].ConnectionString;

public string categoriaNueva { get; private set; }

private string nombreCategoria;

private int idCategoria = 0;

public frmGestionCategoria()

{

InitializeComponent();

}

private void CargarCategorias()

{

var dt = new DataTable();

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "SELECT id\_categoria, categoria FROM tbCategoria ORDER BY categoria;";

SqlCommand cmd = new SqlCommand(query, conn);

SqlDataAdapter da = new SqlDataAdapter(cmd);

da.Fill(dt);

}

lstCategoria.DataSource = dt;

lstCategoria.DisplayMember = "categoria";

}

private void btnGuardar\_Click(object sender, EventArgs e)

{

string nuevaCategoria = txtNombreCategoria.Text.Trim();

if (string.IsNullOrEmpty(nuevaCategoria))

{

MessageBox.Show("Por favor, ingrese una categoría");

return;

}

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "INSERT INTO tbCategoria (categoria) VALUES (@categoria);";

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@categoria", nuevaCategoria);

try

{

conn.Open();

cmd.ExecuteNonQuery();

this.categoriaNueva = nuevaCategoria;

this.DialogResult = DialogResult.OK;

this.Close();

}

catch (Exception ex)

{

MessageBox.Show("Error al guardar el autor: " + ex.Message);

}

}

}

private void frmGestionCategoria\_Load(object sender, EventArgs e)

{

CargarCategorias();

}

private void lstCategoria\_SelectedIndexChanged(object sender, EventArgs e)

{

if (lstCategoria.SelectedItems != null)

{

DataRowView drv = (DataRowView)lstCategoria.SelectedItem;

this.nombreCategoria = drv["categoria"].ToString();

this.idCategoria = Convert.ToInt32(drv["id\_categoria"]);

txtNombreCategoria.Text = nombreCategoria;

}

else

{

this.idCategoria = 0;

}

}

private void btnEliminar\_Click(object sender, EventArgs e)

{

if (this.idCategoria == 0)

{

MessageBox.Show("Por favor, seleccione una categoría de la lista para eliminar.");

return;

}

// Pide confirmación al usuario

var confirmacion = MessageBox.Show(

"Estas seguro de que quieres eliminar a esta categoría ?",

"Confirmar Eliminación",

MessageBoxButtons.YesNo,

MessageBoxIcon.Warning

);

if (confirmacion == DialogResult.Yes)

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "DELETE from tbCategoria WHERE id\_categoria = @id";

using (SqlCommand cmd = new SqlCommand(query, conn))

{

// Usa la variable de la clase que guardo el ID

cmd.Parameters.AddWithValue("@id", this.idCategoria);

conn.Open();

cmd.ExecuteNonQuery();

MessageBox.Show("Categoría eliminada con éxito.");

}

}

CargarCategorias();

}

}

private void btnEditar\_Click(object sender, EventArgs e)

{

if (this.idCategoria == 0)

{

MessageBox.Show("Por favor, seleccione una categoría de la lista para editar.");

return;

}

string txtCategoriaSeleccionada = txtNombreCategoria.Text;

var confirmacion = MessageBox.Show(

"¿Estas seguro que quieres editar esta categoría?",

"Confirmar Edición",

MessageBoxButtons.YesNo,

MessageBoxIcon.Warning

);

if (confirmacion == DialogResult.Yes)

{

string query = "UPDATE tbCategoria SET categoria = @categoria WHERE id\_categoria = @id";

using (SqlConnection conn = new SqlConnection(connectionString))

using (SqlCommand cmd = new SqlCommand(query, conn))

{

cmd.Parameters.AddWithValue("@id", this.idCategoria);

cmd.Parameters.AddWithValue("@categoria", txtCategoriaSeleccionada);

try

{

conn.Open();

cmd.ExecuteNonQuery();

MessageBox.Show("Categoría editada con éxito.");

}

catch (Exception ex)

{

MessageBox.Show($"Error al editar Categoría: {ex.Message}");

}

}

CargarCategorias();

}

}

}

}



El + nos lleva al CRUD DE EDITORIAL

Este es su diseño:  
Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Y este es su código:  
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Configuration;

using Microsoft.Data.SqlClient;

namespace Biblioteca

{

public partial class frmGestionEditorial : Form

{

private string connectionString = ConfigurationManager.ConnectionStrings["MiConexion"].ConnectionString;

private string nombreeditorial;

private int ideditorial = 0;

public string EditorialNuevo { get; private set; }

public frmGestionEditorial()

{

InitializeComponent();

}

private void btnGuardar\_Click(object sender, EventArgs e)

{

string nuevoEditorial = txtNombreEditorial.Text.Trim();

if (string.IsNullOrEmpty(nuevoEditorial))

{

MessageBox.Show("Por favor, ingrese un nombre.");

return;

}

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "INSERT INTO tbEditorial (editorial) VALUES (@editorial);";

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@editorial", nuevoEditorial);

try

{

conn.Open();

cmd.ExecuteNonQuery();

this.EditorialNuevo = nuevoEditorial;

this.DialogResult = DialogResult.OK;

this.Close();

}

catch (Exception ex)

{

MessageBox.Show("Error al guardar el autor: " + ex.Message);

}

}

}

private void CargarEditorial()

{

var dt = new DataTable();

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "SELECT id\_editorial, editorial FROM tbEditorial ORDER BY editorial;";

SqlCommand cmd = new SqlCommand(query, conn);

SqlDataAdapter da = new SqlDataAdapter(cmd);

da.Fill(dt);

}

lstEditorial.DataSource = dt;

lstEditorial.DisplayMember = "editorial";

}

private void frmGestionEditorial\_Load(object sender, EventArgs e)

{

CargarEditorial();

}

private void lstEditorial\_SelectedIndexChanged(object sender, EventArgs e)

{

// 1. Verifica que haya un elemento seleccionado.

if (lstEditorial.SelectedItems != null)

{

// 2. Convierte el elemento seleccionado a un DataRowView

DataRowView drv = (DataRowView)lstEditorial.SelectedItem;

// 3. Accede al valor de la columna especifica por su nombre.

// Usa el nombre de la columna que contiene el nombre del autor.

this.nombreeditorial = drv["editorial"].ToString();

// -- Obtener y guardar el ID

this.ideditorial = Convert.ToInt32(drv["id\_editorial"]);

// 4. Asigna ese valor al TextBox.

txtNombreEditorial.Text = nombreeditorial;

}

else

{

// Si no hay nada seleccionado, reinicia el ID

this.ideditorial = 0;

}

}

private void btnEliminar\_Click(object sender, EventArgs e)

{

if (this.ideditorial == 0)

{

MessageBox.Show("Por favor, seleccione una editorial de la lista para eliminar.");

return;

}

// Pide confirmación al usuario

var confirmacion = MessageBox.Show(

"Estas seguro de que quieres eliminar a esta editorial?",

"Confirmar Eliminación",

MessageBoxButtons.YesNo,

MessageBoxIcon.Warning

);

if (confirmacion == DialogResult.Yes)

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "DELETE from tbEditorial WHERE id\_editorial = @id";

using (SqlCommand cmd = new SqlCommand(query, conn))

{

// Usa la variable de la clase que guardo el ID

cmd.Parameters.AddWithValue("@id", this.ideditorial);

conn.Open();

cmd.ExecuteNonQuery();

MessageBox.Show("Autor eliminado con éxito.");

}

}

CargarEditorial();

}

}

private void btnEditar\_Click(object sender, EventArgs e)

{

if (this.ideditorial == 0)

{

MessageBox.Show("Por favor, seleccione una editorial de la lista para editar.");

return;

}

string txtEditorialSeleccionada = txtNombreEditorial.Text;

// Pide confirmación al usuario

var confirmacion = MessageBox.Show(

"¿Estas seguro de que quieres editar a esta editorial?",

"Confirmar Edición",

MessageBoxButtons.YesNo,

MessageBoxIcon.Warning

);

if (confirmacion == DialogResult.Yes)

{

string query = "UPDATE tbEditorial SET editorial = @editorial WHERE id\_editorial = @id";

using (SqlConnection conn = new SqlConnection(connectionString))

using (SqlCommand cmd = new SqlCommand(query, conn))

{

// Usa la variable de la clase que guardo el ID

cmd.Parameters.AddWithValue("@id", this.ideditorial);

cmd.Parameters.AddWithValue("@editorial", txtEditorialSeleccionada);

try

{

conn.Open();

cmd.ExecuteNonQuery();

MessageBox.Show("Editorial editada con éxito.");

}

catch (Exception ex)

{

MessageBox.Show($"Error al actualizar editorial: {ex.Message}");

}

}

CargarEditorial();

}

}

private void btnGuardar\_Click\_1(object sender, EventArgs e)

{

string nuevaEditorial = txtNombreEditorial.Text.Trim();

if (string.IsNullOrEmpty(nuevaEditorial))

{

MessageBox.Show("Por favor, ingrese una editorial.");

return;

}

// Usamos 'using' para asegurar que la conexión se cierre correctamente

using (SqlConnection conn = new SqlConnection(connectionString))

{

// !Importante! usar parámetros para prevenir inyeccion SQL

string query = "INSERT INTO tbEditorial (editorial) VALUES (@editorial);";

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@editorial", nuevaEditorial);

try

{

conn.Open();

cmd.ExecuteNonQuery(); // Ejecuta la insercion

this.EditorialNuevo = nuevaEditorial;

this.DialogResult = DialogResult.OK;

this.Close();

}

catch (Exception ex)

{

MessageBox.Show("Error al guardar la editorial: " + ex.Message);

}

}

}

}

}

Todo este formulario de libros nos ayuda a poder agregar los libros que queramos tener en nuestro sistema

Texto, Carta

El contenido generado por IA puede ser incorrecto.

El botón de LECTORES nos lleva a un CRUD de lectores

Este es su diseño:  
Tabla

El contenido generado por IA puede ser incorrecto.

Y este es su código:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Configuration;

using Microsoft.Data.SqlClient;

namespace Biblioteca

{

public partial class frmGestionLectores : Form

{

int posicion;

private int idLector = 0;

string nombre, app, apm, telefono, telefonopattern = @"^-?\d\*\.?\d+$";

private string connectionString = ConfigurationManager.ConnectionStrings["MiConexion"].ConnectionString;

public frmGestionLectores()

{

InitializeComponent();

llenarDataGridView();

limpiar();

}

private void frmGestionLectores\_Load(object sender, EventArgs e)

{

}

void eliminarbase()

{

posicion = dgvLectores.CurrentRow.Index;

string queryeliminar = "DELETE FROM tbLector WHERE id\_lector = @id\_lector";

using (SqlConnection conn = new SqlConnection(connectionString))

{

conn.Open();

using (SqlCommand cmd = new SqlCommand(queryeliminar, conn))

{

cmd.Parameters.AddWithValue("@id\_lector", this.idLector);

try

{

cmd.ExecuteNonQuery();

MessageBox.Show("Lector eliminado correctamente");

dgvLectores.Rows.RemoveAt(posicion);

}

catch (Exception ex)

{

MessageBox.Show($"Error al eliminar Lector: {ex.Message}");

}

}

}

}

void consultarID()

{

nombre = txtNombre.Text;

app = txtApellidoPaterno.Text;

apm = txtApellidoMaterno.Text;

telefono = txtTelefono.Text;

this.idLector = 0; // Inicializamos en 0 para saber si se encontró o no.

// 1. Consulta SQL corregida con AND

string queryconsultar = "SELECT id\_lector FROM tbLector WHERE nombre = @nombre AND app = @app AND apm = @apm AND telefono = @telefono";

using (SqlConnection conn = new SqlConnection(connectionString))

{

using (SqlCommand cmd = new SqlCommand(queryconsultar, conn))

{

// 2. Agregar solo los parámetros que si usa la consulta.

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

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

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

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

try

{

conn.Open();

// 3. Ejecutar la consulta y obtener el resultado

// ExecuteScalar() es perfecto para obtener un solo valor (como un ID).

object resultado = cmd.ExecuteScalar();

// 4. Verificar si se encontró un resultado antes de usarlo

if (resultado != null)

{

this.idLector = Convert.ToInt32(resultado);

MessageBox.Show("El ID del lector es: " + this.idLector);

}

else

{

this.idLector = 0;

MessageBox.Show("No se encontró un lector con esos datos.");

}

}

catch (Exception ex)

{

MessageBox.Show("Error al consultar la base de datos: " + ex.Message);

}

}

}

}

void agregarbase()

{

nombre = txtNombre.Text;

app = txtApellidoPaterno.Text;

apm = txtApellidoMaterno.Text;

telefono = txtTelefono.Text;

if (nombre == "" || app == "" || apm == "" || telefono == "")

{

MessageBox.Show("No debe hacer campos vacíos");

return;

}

if (!Regex.IsMatch(txtTelefono.Text, telefonopattern))

{

MessageBox.Show("El teléfono debe ser un número valido.", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Warning);

txtTelefono.Focus();

txtTelefono.Text = "";

return;

}

string query = "INSERT INTO tbLector (nombre, app, apm, telefono) " +

"VALUES (@nombre, @app, @apm, @telefono)";

using (SqlConnection conn = new SqlConnection(connectionString))

using (SqlCommand cmd = new SqlCommand(query, conn))

{

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

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

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

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

try

{

conn.Open();

cmd.ExecuteNonQuery();

MessageBox.Show("Lector agregado.");

dgvLectores.Rows.Add(nombre, app, apm, telefono);

}

catch (Exception ex)

{

MessageBox.Show($"Error al agregar cliente {ex.Message}");

}

}

limpiar();

txtNombre.Focus();

}

void modificarbase()

{

posicion = dgvLectores.CurrentRow.Index;

nombre = txtNombre.Text;

app = txtApellidoPaterno.Text;

apm = txtApellidoMaterno.Text;

telefono = txtTelefono.Text;

if (nombre == "" || app == "" || apm == "" || telefono == "")

{

MessageBox.Show("No debe haber campos vacíos", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

if (!Regex.IsMatch(txtTelefono.Text, telefonopattern))

{

MessageBox.Show("El teléfono debe ser un número valido.", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Warning);

txtTelefono.Focus();

txtTelefono.Text = "";

return;

}

if (this.idLector == 0)

{

MessageBox.Show("Por favor, selecciona un lector de la lista para modificar.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

string querymodificar = @"

UPDATE tbLector

SET nombre = @nombre,

app = @app,

apm = @apm,

telefono = @telefono

WHERE id\_lector = @id\_lector";

using (SqlConnection conn = new SqlConnection(connectionString))

using (SqlCommand cmd = new SqlCommand(querymodificar, conn))

{

cmd.Parameters.AddWithValue("@id\_lector", this.idLector);

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

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

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

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

try

{

conn.Open();

cmd.ExecuteNonQuery();

MessageBox.Show("Lector actualizado correctamente.");

dgvLectores[0, posicion].Value = txtNombre.Text;

dgvLectores[1, posicion].Value = txtApellidoPaterno.Text;

dgvLectores[2, posicion].Value = txtApellidoMaterno.Text;

dgvLectores[3, posicion].Value = txtTelefono.Text;

}

catch (Exception ex)

{

MessageBox.Show($"Error al actualizar Cliente: {ex.Message}");

}

}

limpiar();

txtNombre.Focus();

}

private void llenarDataGridView()

{

string query = "SELECT id\_lector, nombre, app, apm, telefono FROM tbLector";

using (SqlConnection conn = new SqlConnection(connectionString))

{

try

{

conn.Open();

SqlCommand cmd = new SqlCommand(query, conn);

using (SqlDataReader reader = cmd.ExecuteReader())

{

while (reader.Read())

{

int rowIndex = dgvLectores.Rows.Add();

dgvLectores.Rows[rowIndex].Cells["colNombre"].Value = reader["nombre"];

dgvLectores.Rows[rowIndex].Cells["colApellidoPaterno"].Value = reader["app"];

dgvLectores.Rows[rowIndex].Cells["colApellidoMaterno"].Value = reader["apm"];

dgvLectores.Rows[rowIndex].Cells["colTelefono"].Value = reader["telefono"];

}

}

}

catch (Exception ex)

{

MessageBox.Show($"Error al llenar el DataGridView: {ex.Message}", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

void limpiar()

{

btnCrear.Enabled = true;

btnEliminar.Enabled = false;

btnEditar.Enabled = false;

txtNombre.Text = "";

txtApellidoPaterno.Text = "";

txtTelefono.Text = "";

txtApellidoMaterno.Text = "";

}

private void btnNuevo\_Click(object sender, EventArgs e)

{

limpiar();

btnCrear.Enabled = true;

txtNombre.Focus();

}

private void btnCrear\_Click(object sender, EventArgs e)

{

agregarbase();

}

private void btnEditar\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("¿Estas seguro de modificar este cliente?", "Confirmación", MessageBoxButtons.YesNo);

if (result == DialogResult.Yes)

{

modificarbase();

}

}

private void dgvLectores\_CellClick(object sender, DataGridViewCellEventArgs e)

{

if (e.RowIndex >= 0)

{

DataGridViewRow filaSeleccionada = dgvLectores.Rows[e.RowIndex];

if (filaSeleccionada.Cells[0].Value != null && filaSeleccionada.Cells[1].Value != null)

{

posicion = dgvLectores.CurrentRow.Index;

txtNombre.Text = dgvLectores[0, posicion].Value.ToString();

txtApellidoPaterno.Text = dgvLectores[1, posicion].Value.ToString();

txtApellidoMaterno.Text = dgvLectores[2, posicion].Value.ToString();

txtTelefono.Text = dgvLectores[3, posicion].Value.ToString();

consultarID();

btnCrear.Enabled = false;

btnEditar.Enabled = true;

btnEliminar.Enabled = true;

txtNombre.Focus();

}

else

{

MessageBox.Show("La fila seleccionada contiene celdas vacías.", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

}

private void btnEliminar\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("¿Estas seguro que deseas eliminar este cliente?",

"Confirmacion", MessageBoxButtons.YesNo);

if (result == DialogResult.Yes)

{

eliminarbase();

limpiar();

txtNombre.Focus();

}

}

private void txtNombre\_Enter(object sender, EventArgs e)

{

}

private void txtNombre\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

txtApellidoPaterno.Focus();

e.Handled = true;

e.SuppressKeyPress = true;

}

}

private void txtApellidoPaterno\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

txtApellidoMaterno.Focus();

e.Handled = true;

e.SuppressKeyPress = true;

}

}

private void txtApellidoMaterno\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

txtTelefono.Focus();

e.Handled = true;

e.SuppressKeyPress = true;

}

}

private void txtTelefono\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

btnCrear.PerformClick();

e.Handled = true;

e.SuppressKeyPress = true;

}

}

}

}

Texto

El contenido generado por IA puede ser incorrecto.

Y si vamos al botón de PRESTAMOS iremos al CRUD de prestamos

Este es su diseño:

Tabla

El contenido generado por IA puede ser incorrecto.

Y este es su código:

using Microsoft.Data.SqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Configuration;

using Microsoft.Data.SqlClient;

namespace Biblioteca

{

public partial class frmPrestamos : Form

{

private string connectionString = ConfigurationManager.ConnectionStrings["MiConexion"].ConnectionString;

private string fechaEntrada, fechaSalida;

int lector, libro, posicion;

private int idPrestamo = 0;

public frmPrestamos()

{

InitializeComponent();

llenarDataGridView();

// Ajustar automáticamente el tamaño de las columnas para que se ajusten al contenido

dgvPrestamos.AutoResizeColumns(DataGridViewAutoSizeColumnsMode.AllCells);

// Ajustar automáticamente el tamaño de la última columna rellenando el espacio restante

dgvPrestamos.Columns[dgvPrestamos.Columns.Count - 1].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

}

void consultarID()

{

fechaSalida = dtpFechaSalida.Value.ToString("yyyy-MM-dd");

fechaEntrada = dtpFechaEntrada.Value.ToString("yyyy-MM-dd");

lector = (int)cmbLector.SelectedValue;

libro = (int)cmbLibro.SelectedValue;

this.idPrestamo = 0;

string queryconsultar = "SELECT id\_prestamo FROM tbPrestamo WHERE fecha\_salida = @fecha\_salida AND fecha\_entrada = @fecha\_entrada AND id\_lector = @idLector AND id\_libro = @idLibro";

using (SqlConnection conn = new SqlConnection(connectionString))

{

using (SqlCommand cmd = new SqlCommand(queryconsultar, conn))

{

cmd.Parameters.AddWithValue("@fecha\_salida", fechaSalida);

cmd.Parameters.AddWithValue("@fecha\_entrada", fechaEntrada);

cmd.Parameters.AddWithValue("@idLector", lector);

cmd.Parameters.AddWithValue("@idLibro", libro);

try

{

conn.Open();

object resultado = cmd.ExecuteScalar();

if (resultado != null)

{

this.idPrestamo = Convert.ToInt32(resultado);

MessageBox.Show("El ID del Préstamo es: " + this.idPrestamo);

}

else

{

this.idPrestamo = 0;

MessageBox.Show("No se encontró un préstamo con estos datos.");

}

}

catch (Exception ex)

{

MessageBox.Show("Error al consultar la base de datos: " + ex.Message);

}

}

}

}

void eliminarprestamo()

{

posicion = dgvPrestamos.CurrentRow.Index;

string queryeliminar = "DELETE FROM tbPrestamo WHERE id\_prestamo = @id\_prestamo";

using (SqlConnection conn = new SqlConnection(connectionString))

{

conn.Open();

using (SqlCommand cmd = new SqlCommand(queryeliminar, conn))

{

cmd.Parameters.AddWithValue("@id\_prestamo", this.idPrestamo);

try

{

cmd.ExecuteNonQuery();

MessageBox.Show("Préstamo eliminado correctamente");

dgvPrestamos.Rows.RemoveAt(posicion);

}

catch (Exception ex)

{

MessageBox.Show($"Error al eliminar el Préstamo: {ex.Message}");

}

}

}

}

private void agregarprestamos()

{

fechaEntrada = dtpFechaEntrada.Value.ToString("yyyy-MM-dd");

fechaSalida = dtpFechaSalida.Value.ToString("yyyy-MM-dd");

lector = (int)cmbLector.SelectedValue;

string lectortxt = cmbLector.Text;

libro = (int)cmbLibro.SelectedValue;

string librotxt = cmbLibro.Text;

if (string.IsNullOrEmpty(cmbLector.Text) || string.IsNullOrEmpty(cmbLibro.Text))

{

MessageBox.Show("No debe haber campos vacíos");

return;

}

string queryAgregar = "INSERT INTO tbPrestamo (fecha\_salida, fecha\_entrada, id\_lector, id\_libro) " +

"VALUES (@fecha\_salida, @fecha\_entrada, @id\_lector, @id\_libro)";

using (SqlConnection conn = new SqlConnection(connectionString))

using (SqlCommand cmd = new SqlCommand(queryAgregar, conn))

{

cmd.Parameters.AddWithValue("@fecha\_salida", fechaSalida);

cmd.Parameters.AddWithValue("@fecha\_entrada", fechaEntrada);

cmd.Parameters.AddWithValue("@id\_lector", lector);

cmd.Parameters.AddWithValue("@id\_libro", libro);

try

{

conn.Open();

cmd.ExecuteNonQuery();

MessageBox.Show("Préstamo guardado con éxito.", "Exito", MessageBoxButtons.OK, MessageBoxIcon.Information);

llenarDataGridView();

cmbLector.SelectedItem = null;

cmbLibro.SelectedItem = null;

}

catch (Exception ex)

{

MessageBox.Show("Error al guardar el prestamo" + ex.Message, "Error de Base de Dato", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

void modificarprestamos()

{

posicion = dgvPrestamos.CurrentRow.Index;

fechaSalida = dtpFechaSalida.Value.ToString("yyyy-MM-dd");

fechaEntrada = dtpFechaEntrada.Value.ToString("yyyy-MM-dd");

lector = (int)cmbLector.SelectedValue;

string lectortxt = cmbLector.Text;

libro = (int)cmbLibro.SelectedValue;

string librotxt = cmbLibro.Text;

if (string.IsNullOrEmpty(cmbLector.Text) || string.IsNullOrEmpty(cmbLibro.Text))

{

MessageBox.Show("No debe haber campos vacíos", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

string querymodificar = @"

UPDATE tbPrestamo

SET fecha\_salida = @fecha\_salida,

fecha\_entrada = @fecha\_entrada,

id\_lector = @idLector,

id\_libro = @idLibro

WHERE id\_prestamo = @idPrestamo";

using (SqlConnection conn = new SqlConnection(connectionString))

using (SqlCommand cmd = new SqlCommand(querymodificar, conn))

{

cmd.Parameters.AddWithValue("@idPrestamo", this.idPrestamo);

cmd.Parameters.AddWithValue("@fecha\_salida", fechaSalida);

cmd.Parameters.AddWithValue("@fecha\_entrada", fechaEntrada);

cmd.Parameters.AddWithValue("@idLector", lector);

cmd.Parameters.AddWithValue("@idLibro", libro);

try

{

conn.Open();

cmd.ExecuteNonQuery();

MessageBox.Show("Préstamo actualizado correctamente.");

dgvPrestamos[0, posicion].Value = fechaSalida;

dgvPrestamos[1, posicion].Value = fechaEntrada;

dgvPrestamos[2, posicion].Value = lectortxt;

dgvPrestamos[3, posicion].Value = librotxt;

}

catch (Exception ex)

{

MessageBox.Show($"Error al actualizar el Préstamo: {ex.Message}");

}

}

limpiar();

dtpFechaSalida.Focus();

}

private void frmPrestamos\_Load(object sender, EventArgs e)

{

CargarLibros();

CargarLectores();

}

private void CargarLectores()

{

DataTable dt = new DataTable();

using (SqlConnection conn = new SqlConnection(connectionString))

{

string queryLectores = "SELECT id\_lector, nombre FROM tbLector ORDER BY nombre;";

SqlDataAdapter da = new SqlDataAdapter(queryLectores, conn);

da.Fill(dt);

}

cmbLector.DataSource = dt;

cmbLector.DisplayMember = "nombre";

cmbLector.ValueMember = "id\_lector";

cmbLector.SelectedItem = null;

}

private void CargarLibros()

{

DataTable dt = new DataTable();

using (SqlConnection conn = new SqlConnection(connectionString))

{

string queryLibros = "SELECT id\_libro, titulo FROM tbLibros ORDER BY titulo;";

SqlDataAdapter d = new SqlDataAdapter(queryLibros, conn);

d.Fill(dt);

}

cmbLibro.DataSource = dt;

cmbLibro.DisplayMember = "titulo";

cmbLibro.ValueMember = "id\_libro";

cmbLibro.SelectedItem = null;

}

private void btnCrear\_Click(object sender, EventArgs e)

{

agregarprestamos();

}

private void llenarDataGridView()

{

string query = @"

SELECT p.fecha\_salida, p.fecha\_entrada, le.nombre, li.titulo

FROM tbPrestamo p

JOIN tbLector le ON p.id\_lector = le.id\_lector

JOIN tbLibros li ON p.id\_libro = li.id\_libro";

using (SqlConnection conn = new SqlConnection(connectionString))

{

try

{

conn.Open();

SqlCommand cmd = new SqlCommand(query, conn);

using (SqlDataReader reader = cmd.ExecuteReader())

{

while (reader.Read())

{

int rowIndex = dgvPrestamos.Rows.Add();

dgvPrestamos.Rows[rowIndex].Cells["colFechaSalida"].Value = reader["fecha\_salida"];

dgvPrestamos.Rows[rowIndex].Cells["colFechaEntrada"].Value = reader["fecha\_entrada"];

dgvPrestamos.Rows[rowIndex].Cells["colLector"].Value = reader["nombre"];

dgvPrestamos.Rows[rowIndex].Cells["colLibro"].Value = reader["titulo"];

}

}

}

catch (Exception ex)

{

MessageBox.Show($"Error al llenar el DataGridView: {ex.Message}", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

void limpiar()

{

btnCrear.Enabled = true;

btnEliminar.Enabled = false;

btnEditar.Enabled = false;

cmbLector.Text = "";

cmbLibro.Text = "";

}

private void dgvPrestamos\_CellClick(object sender, DataGridViewCellEventArgs e)

{

if (e.RowIndex >= 0)

{

DataGridViewRow filaSeleccionada = dgvPrestamos.Rows[e.RowIndex];

if (filaSeleccionada.Cells[0].Value != null && filaSeleccionada.Cells[1].Value != null)

{

posicion = dgvPrestamos.CurrentRow.Index;

dtpFechaSalida.Text = dgvPrestamos[0, posicion].Value.ToString();

dtpFechaEntrada.Text = dgvPrestamos[1, posicion].Value.ToString();

cmbLector.Text = dgvPrestamos[2, posicion].Value.ToString();

cmbLibro.Text = dgvPrestamos[3, posicion].Value.ToString();

consultarID();

btnCrear.Enabled = false;

btnEditar.Enabled = true;

btnEliminar.Enabled = true;

dtpFechaSalida.Focus();

}

else

{

MessageBox.Show("La fila seleccionada contiene celdas vacías.", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

}

private void btnNuevo\_Click(object sender, EventArgs e)

{

limpiar();

btnCrear.Enabled = true;

dtpFechaSalida.Focus();

}

private void btnEditar\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("¿Estas seguro de modificar este Prestamo?", "Confirmación", MessageBoxButtons.YesNo);

if (result == DialogResult.Yes)

{

modificarprestamos();

}

}

private void btnEliminar\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("¿Estas seguro que deseas eliminar este Prestamo?",

"Confirmacion", MessageBoxButtons.YesNo);

if (result == DialogResult.Yes)

{

eliminarprestamo();

limpiar();

dtpFechaSalida.Focus();

}

}

private void cmbLector\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

cmbLibro.Focus();

e.Handled = true;

e.SuppressKeyPress = true;

}

}

private void cmbLibro\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

btnCrear.PerformClick();

e.Handled = true;

e.SuppressKeyPress = true;

}

}

}

}

El archivo de conexión se usa esta variable global en todos los formulario que se necesite usar:

private string connectionString = ConfigurationManager.ConnectionStrings["MiConexion"].ConnectionString;

Esta conexión esta configurada a través de con archivo .config

Este es su código:

<?xml version="1.0" encoding="utf-8"?>

<configuration>

<startup>

<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />

</startup>

<connectionStrings>

<add name="MiConexion"

connectionString="Server=.\SQLEXPRESS;Database=BDBIBLIOTECA;Trusted\_Connection=True;TrustServerCertificate=True;"

providerName="System.Data.SqlClient" />

</connectionStrings>

</configuration>

Mediante el nombre que se le asigne en la connectionString ese se va a usar para todos los formulario. En este caso se configuró en connectionString

add name="MiConexion"

Y la variable connectionString es donde se va a nombrar el servidor y el nombre de la base de datos

"Server=.\SQLEXPRESS;Database=BDBIBLIOTECA;Trusted\_Connection=True;TrustServerCertificate=True;"