Materia: **Programación Visual**

Proyecto: **Parcial 2**

Fecha: **3 de Noviembre de 2025**

Semestre: **5**

Carrera: **Lic. Informática y tecnologías computacionales**

Alumno:

Liev Fabricio Torres Trinidad

Documentación de código del proyecto

```
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LFTT_ProyectoParcial1
{
    internal class conexion
    {
        public static MySqlConnection conexionp()
        {
            string servidor = "localhost";
            string bd = "playlist";
            string usuario = "root";
            string password = "hola";
            string cadenaConexion = "Database=" + bd + ";Data Source=" +
servidor + ";User Id=" + usuario + ";Password=" + password + "";
            try
            {
                MySqlConnection conexionBD = new
MySqlConnection(cadenaConexion);
                return conexionBD;
            }
            catch (MySqlException ex)
            {
                Console.WriteLine("Error: " + ex.Message);
                return null;
            }
        }
    }
}
```

Componente	Función
Clase conexion	Contiene la lógica para construir la cadena de conexión y abrir la conexión. Es una clase internal.
cadenaConexion	Almacena los parámetros necesarios (servidor, BD, usuario, contraseña) para que el conector MySQL pueda localizar y autenticar al usuario.
conexionp()	Método principal (asumo que se llama conexionp() por tu código de consulta) que intenta crear y devolver un objeto MySqlConnection. Si falla (ej. contraseña

Componente	Función
	incorrecta, servidor apagado), devuelve null y notifica el error en consola.

ConsultasBD

```
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace LFTT_ProyectoParcial1
{
    public static class ConsultasBD
    {
        public static bool InsertarCancion(string nomcan, string duracion,
string album, string artista, string anio, string genero)
        {
            string sql = "INSERT INTO playcan (nombre_cancion, Duracion, Album,
Artista, Anio, Genero) " +
                         "VALUES (@nombre, @duracion, @album, @artista, @anio,
@genero)";
            MySqlConnection conexionBD = conexion.conexionp();
            if (conexionBD == null) return false;

            try
            {
                conexionBD.Open();
                MySqlCommand comando = new MySqlCommand(sql, conexionBD);
                comando.Parameters.AddWithValue("@nombre", nomcan);
                comando.Parameters.AddWithValue("@duracion", duracion);
                comando.Parameters.AddWithValue("@album", album);
                comando.Parameters.AddWithValue("@artista", artista);
                comando.Parameters.AddWithValue("@anio", anio);
                comando.Parameters.AddWithValue("@genero", genero);
                int filasAfectadas = comando.ExecuteNonQuery();

                return filasAfectadas > 0;
            }
            catch (MySqlException ex)
            {
                MessageBox.Show("Error al insertar la canción: " + ex.Message,
"Error de BD", MessageBoxButtons.OK, MessageBoxIcon.Error);
                return false;
            }
            finally
            {
```

```
                if (conexionBD.State == ConnectionState.Open)
conexionBD.Close();
            }
        }

        public static DataTable BuscarCancion(string nombreCancion)
        {
            DataTable dt = new DataTable();
            string sql = "SELECT id, nombre_cancion, Duracion, Album, Artista,
Anio, Genero FROM playcan WHERE nombre_cancion = @nombre";

            MySqlConnection conexionBD = conexion.conexionp();
            if (conexionBD == null) return dt;

            try
            {
                conexionBD.Open();
                MySqlCommand comando = new MySqlCommand(sql, conexionBD);
                comando.Parameters.AddWithValue("@nombre", nombreCancion);
                MySqlDataAdapter da = new MySqlDataAdapter(comando);
                da.Fill(dt);
                return dt;
            }
            catch (MySqlException ex)
            {
                MessageBox.Show("Error al buscar la canción: " + ex.Message);
                return dt;
            }
            finally
            {
                if (conexionBD.State == ConnectionState.Open)
conexionBD.Close();
            }
        }
        public static bool EliminarCancion(string nombreCancion)
        {
            // SQL: DELETE FROM [nombre_tabla] WHERE [columna] = [valor]
            string sql = "DELETE FROM playcan WHERE nombre_cancion = @nombre";

            // Nota: Usando el método de conexión que has estado usando
            MySqlConnection conexionBD = conexion.conexionp();
            if (conexionBD == null) return false;

            try
            {
                conexionBD.Open();
                MySqlCommand comando = new MySqlCommand(sql, conexionBD);

                // Uso de parámetros
                comando.Parameters.AddWithValue("@nombre", nombreCancion);

                int filasAfectadas = comando.ExecuteNonQuery();

                // Si filasAfectadas es > 0, significa que se borró al menos un
registro
                return filasAfectadas > 0;
            }
            catch (MySqlException ex)
```

```
            {
                MessageBox.Show("Error al eliminar la canción: " + ex.Message,
"Error de BD", MessageBoxButtons.OK, MessageBoxIcon.Error);
                return false;
            }
            finally
            {
                if (conexionBD.State == ConnectionState.Open)
                {
                    conexionBD.Close();
                }
            }
        }
        public static bool ActualizarCancion(int id, string nombre, string
duracion, string album, string artista, string anio, string genero)
        {
            string sql = "UPDATE playcan SET nombre_cancion=@nombre,
Duracion=@duracion, Album=@album, Artista=@artista, Anio=@anio, Genero=@genero
" +
                         "WHERE id=@id";

            MySqlConnection conexionBD = conexion.conexionp();
            if (conexionBD == null) return false;

            try
            {
                conexionBD.Open();
                MySqlCommand comando = new MySqlCommand(sql, conexionBD);
                comando.Parameters.AddWithValue("@id", id);
                comando.Parameters.AddWithValue("@nombre", nombre);
                comando.Parameters.AddWithValue("@duracion", duracion);
                comando.Parameters.AddWithValue("@album", album);
                comando.Parameters.AddWithValue("@artista", artista);
                comando.Parameters.AddWithValue("@anio", anio);
                comando.Parameters.AddWithValue("@genero", genero);

                int filasAfectadas = comando.ExecuteNonQuery();

                return filasAfectadas > 0;
            }
            catch (MySqlException ex)
            {
                MessageBox.Show("Error al actualizar la canción: " +
ex.Message, "Error de BD", MessageBoxButtons.OK, MessageBoxIcon.Error);
                return false;
            }
            finally
            {
                if (conexionBD.State == ConnectionState.Open)
                {
                    conexionBD.Close();
                }
            }
        }
    }
}
```

Método	Función	SQL Ejecutado
InsertarCancion	CREAR. Recibe todos los datos de la canción (nombre, duración, etc.) y los inserta como un nuevo registro en la tabla playcan.	INSERT INTO playcan (...) VALUES (...)
BuscarCancion	READ (Consulta Específica). Busca una canción por su nombre_cancion. Devuelve un objeto DataTable con todos los detalles del registro encontrado. Se usa para cargar datos en el formulario Consultas.	SELECT * FROM playcan WHERE nombre_cancion = @nombre
ActualizarCancion	UPDATE. Recibe todos los campos, incluyendo el ID (clave primaria), para modificar los valores de la canción correspondiente en la base de datos.	UPDATE playcan SET ... WHERE id = @id
EliminarCancion	DELETE. Recibe el nombre_cancion y elimina el registro(s) que coincida(n) con ese nombre de la tabla playcan.	DELETE FROM playcan WHERE nombre_cancion = @nombre
Manejo de Transacciones	MySqlCommand.Parameters.AddWithValue. Es crucial para prevenir ataques de Inyección SQL al garantizar que los valores de entrada sean tratados como datos y no como parte del comando SQL.	N/A

Formulario agregar

```
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;

namespace LFTT_ProyectoParcial1
{
    public partial class Agregar : Form
    {
        public Agregar()
        {
            InitializeComponent();
        }

        private void Agregar_Load(object sender, EventArgs e)
        {
        }

        private void button1_Click(object sender, EventArgs e)
        {
            string nomcan = nomcantxt.Text;
            string duracion = duracion_can.Text;
            string album = album_can.Text;
            string artista = art_can.Text;
            string anio = anio_can.Text;
            string genero = gen_can.Text;

            if (string.IsNullOrWhiteSpace(nomcan))
            {
                MessageBox.Show("Por favor, introduce el nombre de una
canción.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
                return;
            }

            bool exito = ConsultasBD.InsertarCancion(nomcan, duracion, album,
artista, anio, genero);
            if (exito)
            {
                MessageBox.Show($"'{nomcan}' se añadió a tu playlist en la base
de datos.", "Canción Agregada");
            }
            else
            {
                MessageBox.Show("No se pudo añadir la canción. Revisa la
conexión y los datos.", "Error de Inserción", MessageBoxButtons.OK,
MessageBoxIcon.Error);
            }

            this.Close();
        }

        private void label1_Click(object sender, EventArgs e)
        {
```

```
        }

        private void album_can_TextChanged(object sender, EventArgs e)
        {

        }

        private void art_can_TextChanged(object sender, EventArgs e)
        {

        }

        private void anio_can_TextChanged(object sender, EventArgs e)
        {

        }
    }
}
```

Evento/Función	Código de BD Involucrado	Explicación
button1_Click	ConsultasBD.InsertarCancion(...)	Reemplaza la lógica de guardar en un array. Recolecta todos los datos de los TextBox y los envía al método de inserción de la base de datos. Muestra un mensaje de éxito o error.

Formulario Consultas/Actualizacion

```
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;

namespace LFTT_ProyectoParcial1
{
```

```
    public partial class Consultas : Form
    {
        public Consultas()
        {
            InitializeComponent();
        }

        private void Consultas_Load(object sender, EventArgs e)
        {

        }

        private void btnConsulta_Click(object sender, EventArgs e)
        {
            string nomcan = nomcantxt.Text;
            if (string.IsNullOrWhiteSpace(nomcan))
            {
                MessageBox.Show("Por favor, introduce el nombre de la canción a
buscar.", "Búsqueda", MessageBoxButtons.OK, MessageBoxIcon.Information);
                return;
            }
            id_cancion.Text = "";
            duracion_can.Text = "";
            Album_can.Text = "";
            art_can.Text = "";
            anio_can.Text = "";
            gen_can.Text = "";
            DataTable dtResultado = ConsultasBD.BuscarCancion(nomcan);
            if (dtResultado != null && dtResultado.Rows.Count > 0)
            {
                DataRow cancion = dtResultado.Rows[0];
                id_cancion.Text = cancion["id"].ToString();
                duracion_can.Text = cancion["Duracion"].ToString();
                Album_can.Text = cancion["Album"].ToString();
                art_can.Text = cancion["Artista"].ToString();
                anio_can.Text = cancion["Anio"].ToString();
                gen_can.Text = cancion["Genero"].ToString();

                MessageBox.Show($"¡Se encontró la canción '{nomcan}'!",
"Canción Encontrada");
            }
            else
            {
                MessageBox.Show($"No se encontró la canción '{nomcan}' en la
base de datos.", "Canción No Encontrada");
            }
        }

        private void btn_actualizar_Click(object sender, EventArgs e)
        {
            if (!int.TryParse(id_cancion.Text, out int idCancion))
            {
                MessageBox.Show("Primero debes buscar y cargar una canción para
actualizar (el campo ID no es válido).", "Error", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                return;
            }
            string nomcan = nomcantxt.Text;
```

```
            string duracion = duracion_can.Text;
            string album = Album_can.Text;
            string artista = art_can.Text;
            string anio = anio_can.Text;
            string genero = gen_can.Text;
            if (string.IsNullOrWhiteSpace(nomcan))
            {
                MessageBox.Show("El nombre de la canción no puede estar
vacío.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
                return;
            }
            bool exito = ConsultasBD.ActualizarCancion(idCancion, nomcan,
duracion, album, artista, anio, genero);
            if (exito)
            {
                MessageBox.Show($"La canción '{nomcan}' (ID: {idCancion}) se ha
actualizado correctamente.", "Actualización Exitosa");
                btnConsulta_Click(sender, e);
            }
            else
            {
                MessageBox.Show("No se pudo actualizar la canción. Revisa la
conexión o si los datos cambiaron.", "Error de Actualización");
            }
        }
    }
}
```

Evento/Función	Código de BD Involucrado	Explicación
btnConsulta_Click	ConsultasBD.BuscarCancion(nomcan)	Reemplaza la lógica de búsqueda en el array. Envía el nombre de la canción a la BD, recibe el DataTable, y rellena todos los TextBox del formulario con los datos de la canción.

Evento/Función	Código de BD Involucrado	Explicación
btnActualizar_Click	ConsultasBD.ActualizarCancion(...)	Recolecta el ID (clave) y todos los valores actualizados de los TextBox. Llama al método de actualización para guardar los cambios en la BD.

Formulario Borrar

```
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;

namespace LFTT_ProyectoParcial1
{
    public partial class Borrar : Form
    {
        public Borrar()
        {
            InitializeComponent();
        }

        private void Borrar_Load(object sender, EventArgs e)
        {

        }

        private void btnBorrar_Click(object sender, EventArgs e)
        {
            string nomcan = nomcantxt.Text;
            if (string.IsNullOrWhiteSpace(nomcan))
            {
```

```
                MessageBox.Show("Por favor, introduce el nombre de la canción a
eliminar.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
                return;
            }
            DialogResult confirmacion = MessageBox.Show($"¿Estás seguro de que
quieres eliminar la canción '{nomcan}'? Esta acción es irreversible.",
                                                        "Confirmar
Eliminación",

MessageBoxButtons.YesNo,

MessageBoxIcon.Warning);

            if (confirmacion == DialogResult.Yes)
            {
                bool exito = ConsultasBD.EliminarCancion(nomcan);
                if (exito)
                {
                    MessageBox.Show($"'{nomcan}' ha sido eliminada exitosamente
de la base de datos.", "Canción Eliminada");
                }
                else
                {
                    MessageBox.Show("No se encontró la canción o hubo un error
al intentar eliminarla.", "Error de Eliminación");
                }
            }
            this.Close();
        }
    }
}
```

Evento/Función	Código de BD Involucrado	Explicación
btnBorrar_Click	ConsultasBD.EliminarCancion(nomcan)	Reemplaza la lógica de poner null en el array. Pide confirmación al usuario y luego llama al método que ejecuta la instrucción DELETE en la base de datos, usando solo el

Evento/Función	Código de BD Involucrado	Explicación
		nombre de la canción.