


Laboratorio – CRUD de Registro y Autenticación de Alumnos con ADO.NET y SQL Server

Nombre: Celeste Pérez y Josael Zurita

1. Capturas de la Interfaz

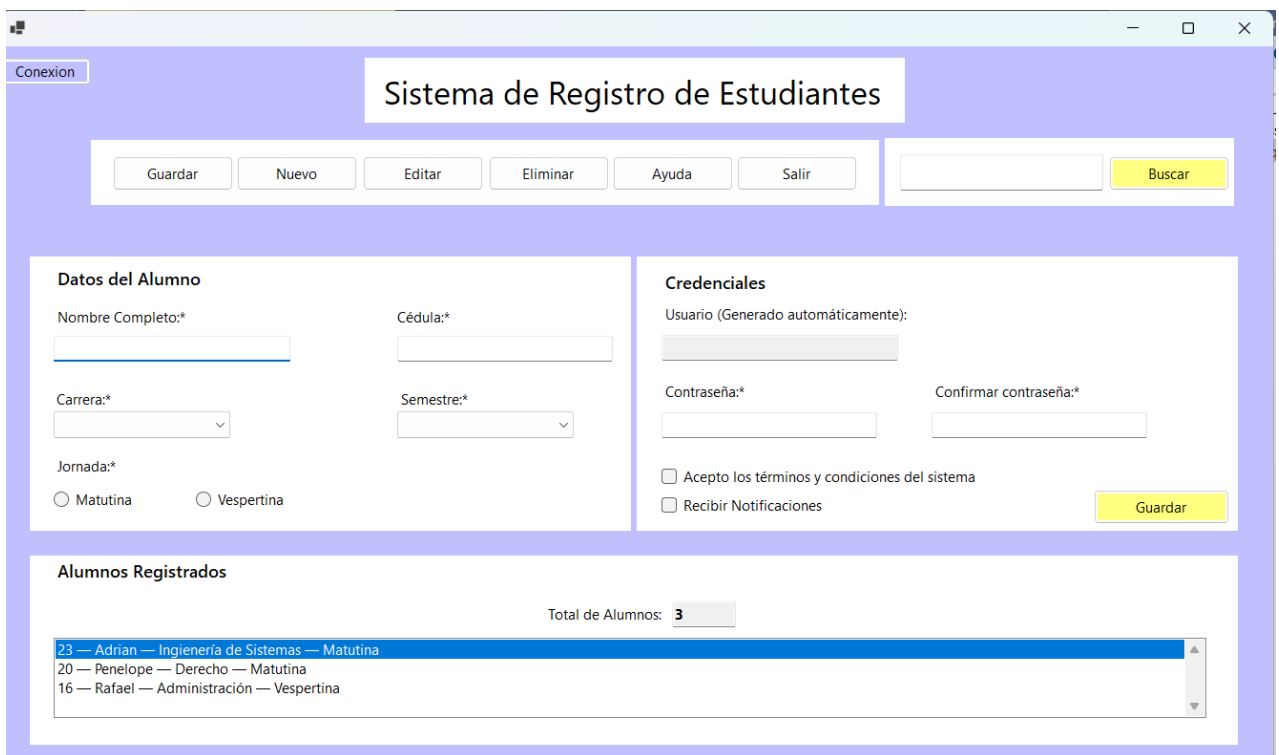


Form2

Acceso al Sistema

Ingresa el código de profesor

Ingresar **Salir**



Conexion

Sistema de Registro de Estudiantes

Guardar **Nuevo** **Editar** **Eliminar** **Ayuda** **Salir** **Buscar**

Datos del Alumno

Nombre Completo:*

Cédula:*

Carrera:*

Semestre:*

Jornada:*
☐ Matutina ☐ Vespertina

Credenciales

Usuario (Generado automáticamente):

Contraseña:* Confirmar contraseña:*

☐ Acepto los términos y condiciones del sistema
☐ Recibir Notificaciones

Guardar

Alumnos Registrados

Total de Alumnos: **3**

23	Adrian	Ingeniería de Sistemas	Matutina
20	Penelope	Derecho	Matutina
16	Rafael	Administración	Vespertina

2. Código del CRUD Completo

```
RegistroAlumnos_CelestePerezJosaelZurita  RegistroAlumnos_CelestePerezJosaelZurita EliminarAlumnoPorID(int id)
1  using Microsoft.Data.SqlClient;
2  using System;
3  using System.Collections.Generic;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7  using System.Windows.Forms;
8
9  namespace RegistroAlumnos_CelestePerezJosaelZurita
10 {
11     2 referencias
12     internal class CRUD
13     {
14
15         private Conexion cn = new Conexion();
16
17         1 referencia
18         public void InsertarAlumno(TextBox textNombre, TextBox textCedu, TextBox textCon,
19             TextBox textCon2, ComboBox combo1, ComboBox combo2,
20             RadioButton rbtMat, RadioButton rbtVis, CheckBox check2, TextBox textUser)
21         {
22             try
23             {
24                 SqlConnection conn = cn.Abrir();
25
26                 string sql = @"INSERT INTO Alumnos
27                     (Nombre, Cedula, Carrera, Semestre, Jornada, Usuario, Contraseña, RecibirNotific
28                     VALUES (@Nombre, @Cedula, @Carrera, @Semestre, @Jornada, @Usuario, @Contraseña,
29
30                 SqlCommand cmd = new SqlCommand(sql, conn);
31
32                 cmd.Parameters.AddWithValue("@Nombre", textNombre.Text.Trim());
33                 cmd.Parameters.AddWithValue("@Cedula", textCedu.Text.Trim());
34                 cmd.Parameters.AddWithValue("@Carrera", combo1.Text);
35                 cmd.Parameters.AddWithValue("@Semestre", combo2.Text);
36
37                 string jornada = rbtMat.Checked ? "Matutina" : "Vespertina";
38                 cmd.Parameters.AddWithValue("@Jornada", jornada);
39
40                 cmd.Parameters.AddWithValue("@Usuario", textUser.Text.Trim());
41                 cmd.Parameters.AddWithValue("@Contraseña", textCon.Text.Trim());
42                 cmd.Parameters.AddWithValue("@Noti", check2.Checked);
43
44                 cmd.ExecuteNonQuery();
45
46                 MessageBox.Show("Alumno guardado exitosamente ✓",
47                     "INSERT", MessageBoxButtons.OK, MessageBoxIcon.Information);
48
49                 cn.Cerrar();
50             }
51             catch (Exception ex)
52             {
53                 MessageBox.Show("Error INSERT ✗\n" + ex.Message);
54             }
55         }
56
57         // UPDATE
58         1 referencia
59         public void ActualizarAlumnoPorID(int id, TextBox textNombre, TextBox textCedu, TextBox textCon, TextBox
60             ComboBox combo1, ComboBox combo2, RadioButton rbtMat, RadioButton rbt
61             CheckBox check2, TextBox textUser)
62         {
63             try
64             {
65                 SqlConnection conn = cn.Abrir();
66
67                 string sql = @"UPDATE Alumnos SET
68                     Nombre=@Nombre,
69                     Cedula=@Cedula,
70                     Contraseña=@Contraseña,
71                     RecibirNotificaciones=@RecibirNotificaciones,
72                     Usuario=@Usuario,
73                     Carrera=@Carrera,
74                     Semestre=@Semestre,
75                     Jornada=@Jornada";
76
77                 cmd.Parameters.AddWithValue("@Nombre", textNombre.Text.Trim());
78                 cmd.Parameters.AddWithValue("@Cedula", textCedu.Text.Trim());
79                 cmd.Parameters.AddWithValue("@Contraseña", textCon.Text.Trim());
80                 cmd.Parameters.AddWithValue("@RecibirNotificaciones", check2.Checked);
81                 cmd.Parameters.AddWithValue("@Usuario", textUser.Text.Trim());
82                 cmd.Parameters.AddWithValue("@Carrera", combo1.Text);
83                 cmd.Parameters.AddWithValue("@Semestre", combo2.Text);
84                 cmd.Parameters.AddWithValue("@Jornada", jornada);
85
86                 cmd.ExecuteNonQuery();
87
88                 MessageBox.Show("Alumno actualizado exitosamente ✓",
89                     "UPDATE", MessageBoxButtons.OK, MessageBoxIcon.Information);
90
91                 cn.Cerrar();
92             }
93             catch (Exception ex)
94             {
95                 MessageBox.Show("Error UPDATE ✗\n" + ex.Message);
96             }
97         }
98     }
99 }
```

```
RegistoAlumnos_CelestePerezJosaelZurita
Cedula=@Cedula,
Carrera=@Carrera,
Semestre=@Semestre,
Jornada=@Jornada,
Usuario=@Usuario,
Contrasena=@Contrasena,
RecibirNotificaciones=@Noti
WHERE ID=@ID";

SqlCommand cmd = new SqlCommand(sql, conn);

cmd.Parameters.AddWithValue("@Nombre", textNombre.Text.Trim());
cmd.Parameters.AddWithValue("@Cedula", textCedu.Text.Trim());
cmd.Parameters.AddWithValue("@Carrera", combo1.Text);
cmd.Parameters.AddWithValue("@Semestre", combo2.Text);

string jornada = rbMat.Checked ? "Matutina" : "Vespertina";
cmd.Parameters.AddWithValue("@Jornada", jornada);

cmd.Parameters.AddWithValue("@Usuario", textUser.Text.Trim());
cmd.Parameters.AddWithValue("@Contrasena", textCon.Text.Trim());
cmd.Parameters.AddWithValue("@Noti", check2.Checked);
cmd.Parameters.AddWithValue("@ID", id);

cmd.ExecuteNonQuery();

MessageBox.Show("Alumno actualizado correctamente ✓", "UPDATE", MessageBoxButtons.OK, MessageB
cn.Cerrar();
}
catch (Exception ex)
{
    MessageBox.Show("Error UPDATE ✗\n" + ex.Message);
}
}

// DELETE
```

```
RegistoAlumnos_CelestePerezJosaelZurita
// DELETE
1 referencia
public bool EliminarAlumnoPorID(int id)
{
    try
    {
        SqlConnection conn = cn.Abrir();
        string sql = "DELETE FROM Alumnos WHERE ID=@ID";
        SqlCommand cmd = new SqlCommand(sql, conn);
        cmd.Parameters.AddWithValue("@ID", id);

        int filas = cmd.ExecuteNonQuery();
        cn.Cerrar();

        if (filas > 0)
        {
            MessageBox.Show("Alumno eliminado correctamente ✓", "DELETE", MessageBoxButtons.OK, Messag
            return true;
        }
        else
        {
            MessageBox.Show("No se encontró el alumno a eliminar.", "DELETE", MessageBoxButtons.OK, Mes
            return false;
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error DELETE ✗\n" + ex.Message);
        return false;
    }
}

0 referencias
public void BuscarPorCedula(TextBox cedula.
```

```

138 public void BuscarPorCedula(TextBox cedula,
139     TextBox textNombre, TextBox textCedu, ComboBox combo1, ComboBox combo2,
140     RadioButton rbtMat, RadioButton rbtVis, CheckBox check2, TextBox textUser, TextBox textCon)
141 {
142     try
143     {
144         SqlConnection conn = cn.Abrir();
145
146         string sql = "SELECT * FROM Alumnos WHERE Cedula=@Cedula";
147         SqlCommand cmd = new SqlCommand(sql, conn);
148         cmd.Parameters.AddWithValue("@Cedula", cedula.Text.Trim());
149
150         SqlDataReader dr = cmd.ExecuteReader();
151
152         if (dr.Read())
153         {
154             textNombre.Text = dr["Nombre"].ToString();
155             textCedu.Text = dr["Cedula"].ToString();
156             combo1.Text = dr["Carrera"].ToString();
157             combo2.Text = dr["Semestre"].ToString();
158             textUser.Text = dr["Usuario"].ToString();
159             textCon.Text = dr["Contraseña"].ToString();
160
161             string j = dr["Jornada"].ToString();
162             rbtMat.Checked = j == "Matutina";
163             rbtVis.Checked = j == "Vespertina";
164
165             check2.Checked = Convert.ToBoolean(dr["RecibirNotificaciones"]);
166         }
167         else
168         {
169             MessageBox.Show("No existe esa cédula ✕");
170         }
171
172         dr.Close();
173         cn.Cerrar();
174     }
175     catch (Exception ex)
176     {
177         MessageBox.Show("Error BUSCAR ✕\n" + ex.Message);
178     }
179 }
180
181
182
183
184
185

```

```

174     }
175     catch (Exception ex)
176     {
177         MessageBox.Show("Error BUSCAR ✕\n" + ex.Message);
178     }
179 }
180
181
182
183
184
185

```

5. Código de ADO.NET

6. Código de Validaciones y Atajos

```

1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Text.RegularExpressions;
6 using System.Threading.Tasks;
7 using System.Windows.Forms;
8
9 namespace RegistroAlumnos_CelestePerezJosaelZurita
10 {
11     internal class Validar
12     {
13         public bool ValidarCampos(TextBox textNombre, TextBox textCedu, TextBox textCon,
14             TextBox textCon2, CheckBox check1, ComboBox combo1,
15             ComboBox combo2, RadioButton rbtMat, RadioButton rbtVis, CheckBox check2)
16         {
17             // Validar que los campos estén completos
18             if (textNombre.Text.Trim() == "" || textCedu.Text.Trim() == "" ||
19                 combo1.SelectedIndex == -1 || combo2.SelectedIndex == -1 ||
20                 (!rbtMat.Checked && !rbtVis.Checked) || textCon.Text.Trim() == "" ||
21                 textCon2.Text.Trim() == "" || check1.Checked == false || check2.Checked == false)
22             {
23                 MessageBox.Show("Se necesita " + "Campos incompletos", "Mensaje de Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Warning);
24                 return false;
25             }
26
27             // Validar nombre solo letras
28             if (!SoloLetras(textNombre.Text))
29             {
30                 MessageBox.Show("El nombre solo permite letras.", "ERROR", MessageBoxButtons.OK, MessageBoxIcon.Error);
31                 textNombre.Focus();
32                 return false;
33             }
34         }
35     }
36 }

```

```

36  // ===== NUEVO: VALIDACION REAL DE CEDULA =====
37  if (!CedulaPanameniaValida(textCedu.Text.Trim()))
38  {
39      MessageBox.Show("La cédula no es válida. Debe tener el formato: 8-1017-808",
40          "Error",
41          MessageBoxButtons.OK,
42          MessageBoxIcon.Error);
43      textCedu.Focus();
44      return false;
45  }
46
47  // Validar confirmación de contraseña
48  if (textCon.Text != textCon2.Text)
49  {
50      MessageBox.Show("Las contraseñas no coinciden.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
51      return false;
52  }
53
54  return true;
55  } // fin ValidarCampos
56
57  1 referencia
58  private bool SoloLetras(string texto)
59  {
60      foreach (char c in texto)
61      {
62          if (!char.IsLetter(c) && c != ' ')
63              return false;
64      }
65      return true;
66  }
67
68  // Ya NO usamos SoloNumeros() para cédula, así que lo dejo por si te sirve en otra cosa
69  0 referencias
70  private bool SoloNumeros(string texto)
71  {
72      foreach (char c in texto)
73      {
74          if (!char.IsDigit(c))
75              return false;
76      }
77      return true;
78  }
79
80  //VALIDACIÓN DEL FORMATO DE CÉDULA PANAMEÑA
81
82  private bool CedulaPanameniaValida(string cedula)
83  {
84      // Provincias 1-13 | Tomo 1-4 dígitos | Asiento 1-4 dígitos
85      string patron = @"^(1[0-3]|1[1-9])-[0-9]{1,4}-[0-9]{1,4}$";
86      return Regex.IsMatch(cedula, patron);
87  }
88  } // fin clase

```

```

70  foreach (char c in texto)
71  {
72      if (!char.IsDigit(c))
73          return false;
74      }
75      return true;
76  }
77
78  //VALIDACIÓN DEL FORMATO DE CÉDULA PANAMEÑA
79
80  private bool CedulaPanameniaValida(string cedula)
81  {
82      // Provincias 1-13 | Tomo 1-4 dígitos | Asiento 1-4 dígitos
83      string patron = @"^(1[0-3]|1[1-9])-[0-9]{1,4}-[0-9]{1,4}$";
84      return Regex.IsMatch(cedula, patron);
85  }
86  } // fin clase
87
88

```

Atajos:

```

//Configurar atajos(limpiar y guardar)
1 referencia
private void Form1_KeyDown(object sender, KeyEventArgs e)
{
    // Ctrl + S: guardar
    if (e.Control && e.KeyCode == Keys.S)
    {
        btnGua2.PerformClick();
    }
    // ESC: limpiar
    if (e.KeyCode == Keys.Escape)
    {
        btnNuevo2.PerformClick();
    }
    if (e.Control && e.KeyCode == Keys.D)
    {
        btnEliminar.PerformClick();
    }
    if (e.Control && e.KeyCode == Keys.E)
    {
        btnEditar.PerformClick();
    }
}

```

7. Reportes del Sistema

Reporte General de Alumnos								
Sistema de Registro de Alumnos								
ID	Nombre Completo	Cédula	Carrera	Sem.	Jornada	Usuario	Notif.	F. Registro
1	Celeste Perez	8-1037-2060	Ingeniería de Sistemas	Cuarto Semestr e	Matutina	c8-1037-2060	True	11/26/2025 8:58:02 a. m.
2	Josael Zurita	8-1017-808	Ingeniería de Sistemas	Sexto Semestr e	Matutina	j8-1017-808	True	11/26/2025 8:58:36 a. m.
3	Adrian Perez	2-757-115	Ingeniería de Sistemas	Tercer Semestr e	Matutina	a2-757-115	True	11/26/2025 8:59:31 a. m.
4	Daniel Perez	2-356-456	Derecho	Cuarto Semestr e	Matutina	d2-356-456	True	11/26/2025 10:51:26 a. m.

Reporte Por Carrera		
Sistema de Registro de Alumnos		
ID	Nombre del Alumno	Carrera
1	Celeste Perez	Ingeniería de Sistemas
2	Josael Zurita	Ingeniería de Sistemas
3	Adrian Perez	Ingeniería de Sistemas
4	Daniel Perez	Derecho

8. Explicación del Flujo del Sistema

El sistema inicia mostrando la interfaz principal, donde el usuario ingresa su contraseña de administrador, luego si es correcta se abre la siguiente pantalla donde se selecciona la opción que desea realizar; al entrar a cada módulo, el sistema carga los datos necesarios desde SQL Server mediante ADO.NET, mostrando la información en tablas, cuadros o controles según corresponda. Cuando el usuario registra, edita, busca o elimina información, el sistema valida los datos ingresados y ejecuta las consultas necesarias en la base de datos, actualizando de inmediato la vista en pantalla. A medida que el usuario navega selecciona las diferentes secciones, el contenido cambia dinámicamente sin cerrar la aplicación, manteniendo un flujo continuo y ordenado. Finalmente, cuando se requieren reportes, el sistema genera un informe RDLC basado en las consultas realizadas, permitiendo visualizarlo. En conjunto, todo el proceso fluye desde la interacción del usuario con el formulario, pasa por la lógica del sistema y acaba en operaciones directas sobre la base de datos y la generación de reportes.

9. Conclusiones del Estudiante

1. Gracias al uso de C#, ADO.NET y SQL Server pude comprender mejor cómo una aplicación se comunica con su base de datos, y cómo se manejan los datos que el usuario ingresa o consulta.
2. Me di cuenta de que organizar el proyecto por partes (pantallas, lógica y consultas) hace que todo sea más fácil de mantener, ya que cualquier ajuste se puede hacer sin desordenar el resto del sistema.
3. Los controles y formularios de Windows Forms facilitaron diseñar una interfaz más amigable, permitiendo que el usuario interactúe con el sistema de forma sencilla y directa.
4. Implementar reportes con RDLC resultó bastante útil para generar documentos claros y presentables, que pueden imprimirse o exportarse cuando se necesita entregar información formal.
5. Trabajar directamente con los datos me enseñó lo importante que es validar bien la información, para evitar errores o registros incorrectos que puedan afectar el funcionamiento del sistema.
6. En conjunto, integrar las pantallas, la base de datos y los reportes me ayudó a entender cómo se construye un sistema real, desde la captura de datos hasta la generación de un informe final listo para usar.