

## 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



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

Este es un formulario para registrar estudiantes. La barra superior muestra "Conexion" y el título "Sistema de Registro de Estudiantes". Los botones de menú son Guardar, Nuevo, Editar, Eliminar, Ayuda, Salir y Buscar. La sección "Datos del Alumno" incluye campos para Nombre Completo, Cédula, Carrera, Semestre y Jornada (con radio buttons para Matutina y Vespertina). La sección "Credenciales" incluye campos para Usuario (generado automáticamente), Contraseña y Confirmar contraseña, así como checkboxes para aceptar los términos y condiciones y recibir notificaciones. Un botón "Guardar" está ubicado en la parte inferior derecha. La sección "Alumnos Registrados" muestra una lista con 3 alumnos: Adrian (Ingeniería de Sistemas, Matutina), Penelope (Derecho, Matutina) y Rafael (Administración, Vespertina).

## 2. Código del CRUD Completo

```
RegistroAlumnos_CelestePerezJosaelZurita    ▾ RegistroAlumnos_CelestePerezJosaelZurita.cs   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, Contrasena, RecibirNotific
28
29                    SqlCommand cmd = new SqlCommand(sql, conn);
30
31                    cmd.Parameters.AddWithValue("@Nombre", textNombre.Text.Trim());
32                    cmd.Parameters.AddWithValue("@Cedula", textCedu.Text.Trim());
33                    cmd.Parameters.AddWithValue("@Carrera", combo1.Text);
34                    cmd.Parameters.AddWithValue("@Semestre", combo2.Text);
35
36                    string jornada = rbtMat.Checked ? "Matutina" : "Vespertina";
37                    cmd.Parameters.AddWithValue("@Jornada", jornada);
38
39                    cmd.Parameters.AddWithValue("@Usuario", textUser.Text.Trim());
40                    cmd.Parameters.AddWithValue("@Contrasena", textCon.Text.Trim());
41                    cmd.Parameters.AddWithValue("@Noti", check2.Checked);
42
43                    cmd.ExecuteNonQuery();
44
45                    MessageBox.Show("Alumno guardado exitosamente ✓",
46                        "INSERT", MessageBoxButtons.OK, MessageBoxIcon.Information);
47
48                    cn.Cerrar();
49                }
50                catch (Exception ex)
51                {
52                    MessageBox.Show("Error INSERT X\n" + ex.Message);
53                }
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
99 %
```

```
RegistroAlumnos_CelestePerezJosaelZurita.cs  RegistroAlumnos_CelestePerezJosaelZurita.cs  EliminarAlumnoPorID(int id)

69     Cedula=@Cedula,
70     Carrera=@Carrera,
71     Semestre=@Semestre,
72     Jornada=@Jornada,
73     Usuario=@Usuario,
74     Contrasena=@Contrasena,
75     RecibirNotificaciones=@Noti
76     WHERE ID=@ID";
77
78     SqlCommand cmd = new SqlCommand(sql, conn);
79
80     cmd.Parameters.AddWithValue("@Nombre", textNombre.Text.Trim());
81     cmd.Parameters.AddWithValue("@Cedula", textCedu.Text.Trim());
82     cmd.Parameters.AddWithValue("@Carrera", combol.Text);
83     cmd.Parameters.AddWithValue("@Semestre", combo2.Text);
84
85     string jornada = rbtMat.Checked ? "Matutina" : "Vespertina";
86     cmd.Parameters.AddWithValue("@Jornada", jornada);
87
88     cmd.Parameters.AddWithValue("@Usuario", textUser.Text.Trim());
89     cmd.Parameters.AddWithValue("@Contrasena", textCon.Text.Trim());
90     cmd.Parameters.AddWithValue("@Noti", check2.Checked);
91     cmd.Parameters.AddWithValue("@ID", id);
92
93     cmd.ExecuteNonQuery();
94
95     MessageBox.Show("Alumno actualizado correctamente ✓", "UPDATE", MessageBoxButtons.OK, MessageBoxIcon.Information);
96
97     cn.Cerrar();
98 }
99 catch (Exception ex)
100 {
101     MessageBox.Show("Error UPDATE X\n" + ex.Message);
102 }
103 }
104 }
```

```
RegistroAlumnos_CelestePerezJosaelZurita.cs  RegistroAlumnos_CelestePerezJosaelZurita.cs  EliminarAlumnoPorID(int id)

104
105     // DELETE
106     1 referencia
107     public bool EliminarAlumnoPorID(int id)
108     {
109         try
110         {
111             SqlConnection conn = cn.Abrir();
112             string sql = "DELETE FROM Alumnos WHERE ID=@ID";
113             SqlCommand cmd = new SqlCommand(sql, conn);
114             cmd.Parameters.AddWithValue("@ID", id);
115
116             int filas = cmd.ExecuteNonQuery();
117             cn.Cerrar();
118
119             if (filas > 0)
120             {
121                 MessageBox.Show("Alumno eliminado correctamente ✓", "DELETE", MessageBoxButtons.OK, MessageBoxIcon.Information);
122                 return true;
123             }
124             else
125             {
126                 MessageBox.Show("No se encontró el alumno a eliminar.", "DELETE", MessageBoxButtons.OK, MessageBoxIcon.Information);
127                 return false;
128             }
129         }
130         catch (Exception ex)
131         {
132             MessageBox.Show("Error DELETE X\n" + ex.Message);
133             return false;
134         }
135
136
137     }
138 }
```

```
RegistroAlumnos_CelestePerezJosaelZurita    ▾ RegistroAlumnos_CelestePerezJosaelZurita.cs   EliminarAlumnoPorID(int id)

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 X");
170             }
171
172             dr.Close();
173             cn.Cerrar();
        }

Línea: 108   Carácter: 13   SPC   CRLF
```

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

## 5. Código de ADO.NET

## 6. Código de Validaciones y Atajos

```

36 // ===== NUERO: 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 private bool SoloLetras(string texto)
58 {
59     foreach (char c in texto)
60     {
61         if (!char.IsLetter(c) && c != ' ')
62             return false;
63     }
64     return true;
65 }
66
67 // Ya NO usamos SoloNumeros() para cédula, así que lo dejo por si te sirve en otra cosa
68 private bool SoloNumeros(string texto)
69 {
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 private bool CedulaPanameniaValida(string cedula)
80 {
81     // Provincias 1-13 | Tomo 1-4 digitos | Asiento 1-4 digitos
82     string patron = @"^(1[0-3]|1[1-9])-1[0-9]{1,4}-1[0-9]{1,4}$";
83     return Regex.IsMatch(cedula, patron);
84 }
85
86 } // fin clase
87
88

```

```

70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88

```

## Atajos:

```

//Configurar atajos(limpiar y guardar)
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	Ingiería de Sistemas	Cuarto Semestre	Matutina	c8-1037-2060	True	11/26/2025 8:58:02 a.m.	
2	Josael Zurita	8-1017-808	Ingiería de Sistemas	Sexto Semestre	Matutina	j8-1017-808	True	11/26/2025 8:58:36 a.m.	
3	Adrian Perez	2-757-115	Ingiería de Sistemas	Tercer Semestre	Matutina	a2-757-115	True	11/26/2025 8:59:31 a.m.	
4	Daniel Perez	2-356-456	Derecho	Cuarto Semestre	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	Ingiería de Sistemas
2	Josael Zurita	Ingiería de Sistemas
3	Adrian Perez	Ingiería 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.