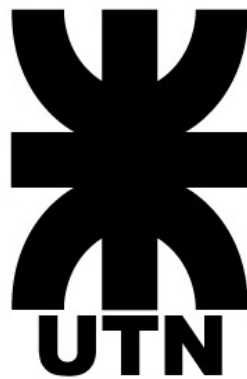


# **Tecnologías de Desarrollo de Software IDE**



**Trabajo práctico  
Academia**

**Grupo 31**

**Integrantes:**

Calvagna, Federico

Sánchez, Juan Manuel

Selva, Ignacio

## Academia

**Descripción:** Academia es un sistema ABM simple que simula manejar las inscripciones a examen similarmente al sistema que se usa actualmente en la facultad. Cada entidad tiene su respectiva alta, baja, modificación y consulta. Cuenta con control de permisos y diferentes tipos de usuarios. Tiene las mismas prestaciones en el entorno web y en escritorio. Con este sistema se puede realizar íntegramente la administración de todo lo necesario, desde cargar nuevos alumnos, asignarles un nuevo usuario hasta asignar permisos para cada usuario.

**Plataforma de Desarrollo:** Visual Studio 2010

**Lenguaje:** C#

**Versión del Framework:** 4.0

**Motor de Base de Datos:** Sql Server 2008 R2

**Entorno de Base de Datos:** Sql Management Studio

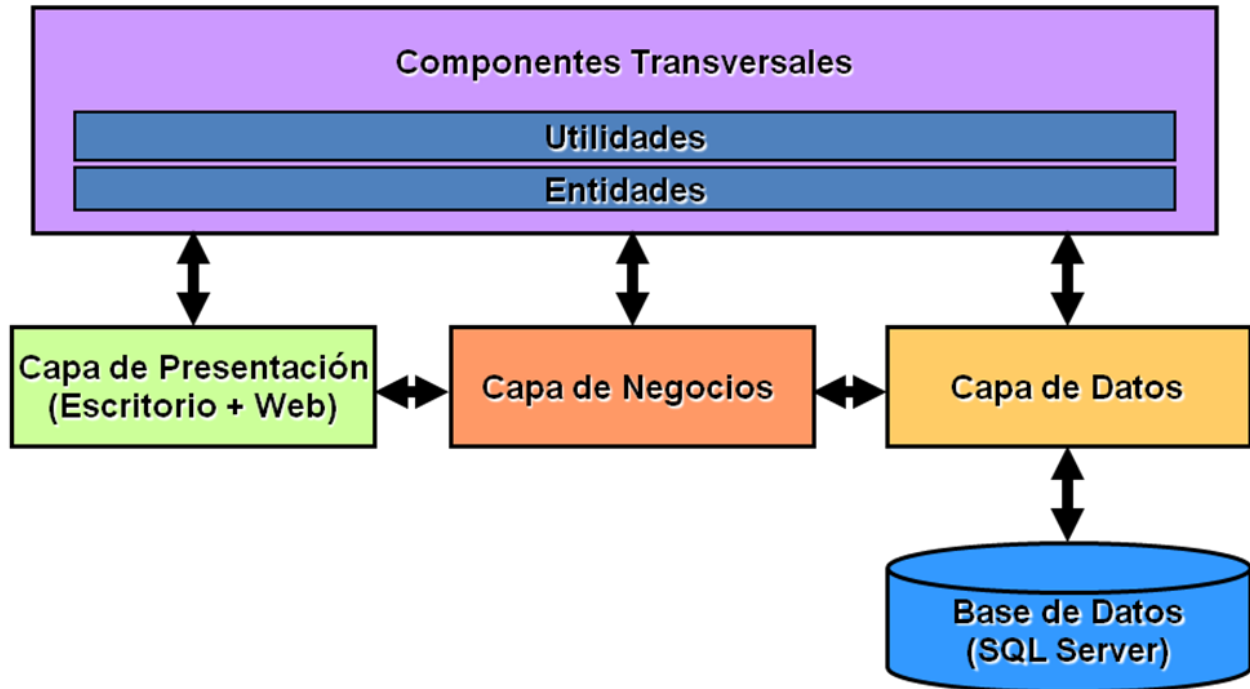
**Sistema de Versionado:** VisualSVN

**Repositorio:** Google Code

**Reporte:** Crystal Report

## Arquitectura

Utilizamos una arquitectura dividida en capas. Básicamente son 4 proyectos y contamos con 2 extras de apoyo. Estos 4 primeros son el proyecto de Acceso a Datos, el de Negocio y los de presentación (Web y Escritorio). Están referenciados para ser utilizados con el siguiente proyecto en orden. Los dos proyectos extras son transversales a todos y son Entidades y Utilidades. El primero cuenta con todas las clases necesarias que representan cada tipo de variable posible y el segundo cuenta con bibliotecas estáticas útiles que son necesarias en todo el proyecto, como por ejemplo, las verificaciones de formato.



**Acceso a Datos:** Se utilizó la biblioteca ADO .NET, lo que nos simplificó mucho las tareas correspondientes. El modelo de datos que decidimos usar fue el Conectado, recibiendo directo de la base de datos a un Data Reader y analizando este manualmente. Las consultas fueron escritas en el código, no utilizando vistas ni stored procedures. No se usaron Data Set para el mapeado de los datos. Se utilizó el patrón Singleton para acceder siempre a la misma instancia de los catálogos.

**Negocio:** En esta capa se hacen algunos cálculos y verificaciones puntuales. Al estar esta de por medio, la interfaz nunca se conecta directamente con la capa de datos, lo que hace nuestro modelo mucho más seguro.

**Presentación en Escritorio:** Utilizamos la biblioteca de Visual Studio "Winforms" con todas sus prestaciones por defecto. Se usaron componentes visuales básicos cargados en la biblioteca (text box, combo box, data grid view, entre otros). Todas las verificaciones de formato se realizaron de manera manual con el proyecto Utilidades.

Se agrega también a modo de prueba, una pequeño reporte de las materias a las que una determinada persona esta anotada. Este fue realizado en Crystal Reports V13\_0\_12. Esta capa se conecta a la de negocio, utiliza el proyecto Entidades y Utilidades, de la misma manera que lo hace la presentación web. Esto nos permite reutilizar mucho código.

**Presentación Web:** En este caso usamos ASP .NET con sus "Webforms". Vale aclarar que todo el código de servidor fue escrito en C#, ya que podría haber sido una combinación de varios. De la misma manera que en Escritorio se usaron los componentes brindados por la biblioteca estándar. Es conveniente contar que en este caso en algunos formularios

preferimos usar un Origen de Datos del tipo Objeto, cuestión que nos simplificó mucho la tarea, especialmente las actualizaciones.

Usamos una masterpage para todos los formularios con una barra de tareas y unas cuestiones visuales. No usamos estilos, skins y los formatos visuales fueron directamente sobre los webforms de manera visual.

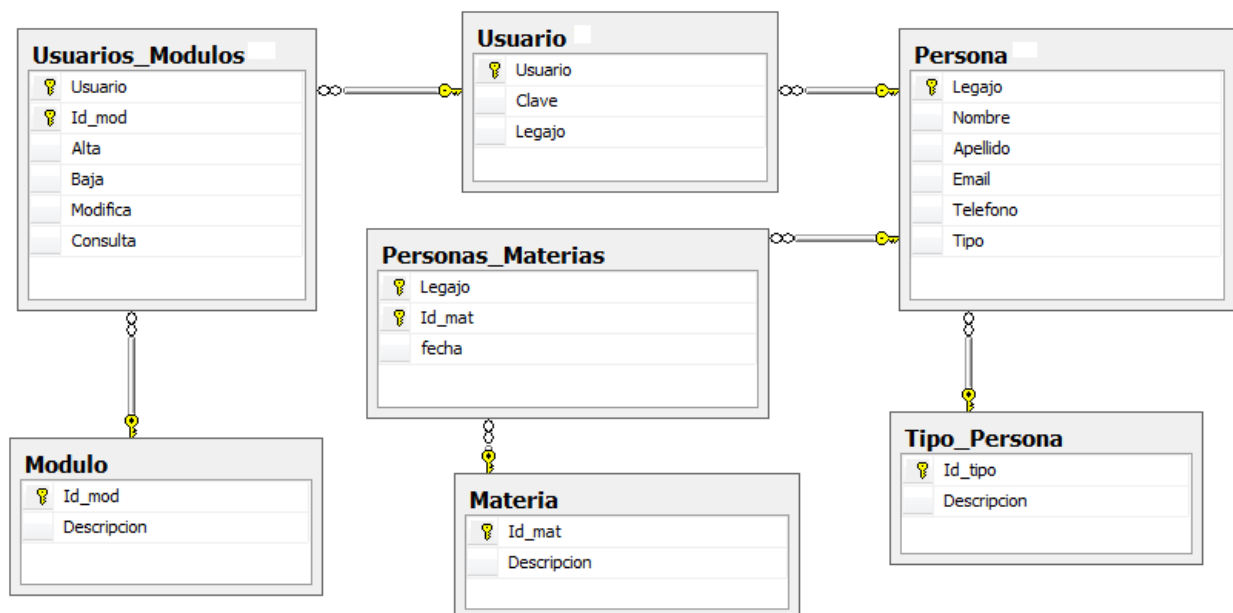
Se usaron variables de sesión para mantener al usuario conectado y páginas. También se usó la Query String para la transmisión de datos entre las páginas.

Esta capa se conecta a la de negocio, utiliza el proyecto Entidades y Utilidades, de la misma manera que lo hace la presentación de Escritorio. Esto nos permite reutilizar mucho código.

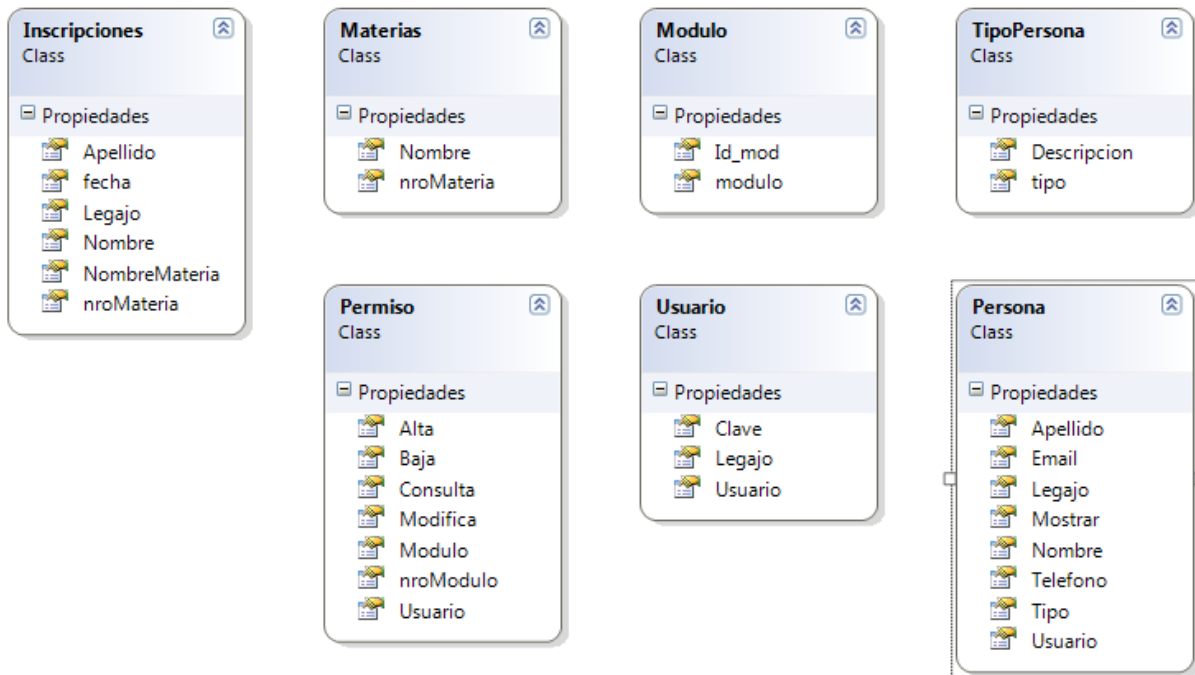
**Utilidades:** Es un proyecto que tiene una clase Formato, que realiza las verificaciones pertinentes, y una ventana modal para mostrar los errores, que es una personalización del MessageBox.

**Entidades:** Contiene todas las clases necesarias respetando los conceptos de POO. Las variables son privadas y cuentan con propiedades para acceder a las mismas. A su vez se aplica herencia, polimorfismo, sobrecarga, entre otros.

## Modelo de Datos



## Diagrama de Clases



## Reporte

Informe Materias

Personas: Admin, Admin

Informe principal

Informe de Inscripciones a Materias

24/11/2014

Nombre	Apellido	Id	Nombre Materia	Fecha
Admin	Admin	1	Analisis Matematico I	11/11/2014
Admin	Admin	3	Sistemas y Org.	29/10/2014
Admin	Admin	5	Algoritmos	27/10/2014
juanse	sanchete	1	Analisis Matematico I	20/11/2014
juanse	sanchete	3	Sistemas y Org.	20/11/2014
Juan Manuel	Sanchez	1	Analisis Matematico I	27/10/2014
Juan Manuel	Sanchez	2	Algebra y geometria Analitica	28/10/2014
Federico	Calvagna	1	Analisis Matematico I	11/10/2014
Federico	Calvagna	2	Algebra y geometria Analitica	23/10/2014
Federico	Calvagna	3	Sistemas y Org.	23/10/2014

Nº de página actual: 1      Nº total de páginas: 1      Factor de zoom: 100%

Informe

Informe

Admin, Admin

Informe principal

Informe de Inscripciones a Materias

24/11/2014

Nombre	Apellido	Id	Nombre Materia	Fecha
Admin	Admin	1	Analisis Matematico I	11/11/2014
Admin	Admin	3	Sistemas y Org.	29/10/2014
Admin	Admin	5	Algoritmos	27/10/2014
juanse	sanchete	1	Analisis Matematico I	20/11/2014
juanse	sanchete	3	Sistemas y Org.	20/11/2014
Juan Manuel	Sanchez	1	Analisis Matematico I	27/10/2014
Juan Manuel	Sanchez	2	Algebra y geometria Analitica	28/10/2014
Federico	Calvagna	1	Analisis Matematico I	11/10/2014
Federico	Calvagna	2	Algebra y geometria Analitica	23/10/2014
Federico	Calvagna	3	Sistemas y Org.	23/10/2014

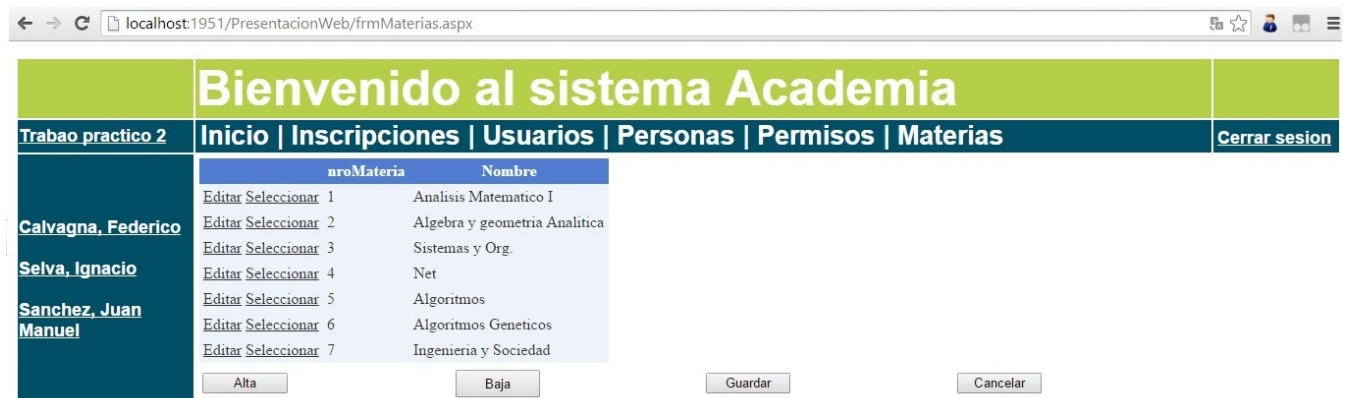
1 de 1      100%

## Ejecuciones de la aplicación

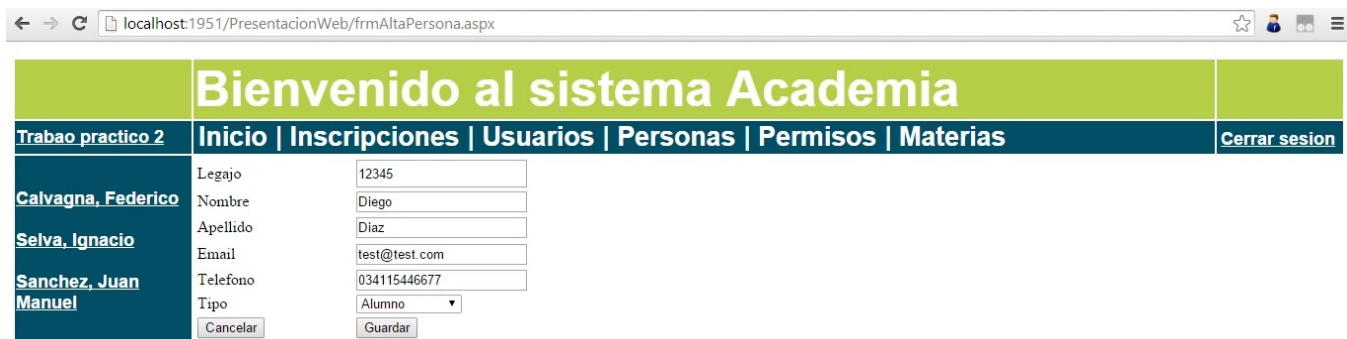
Web



*Pantalla de bienvenida de un usuario logueado.  
Esta vista proporciona un menú dinámico que muestra los módulos que dicho usuario tiene permiso.*



*Formulario de muestra de Materias.  
La grilla que contiene el listado de materias permite la selección y edición de un registro.  
Los botones siguientes a esta grilla, permiten crear una nueva materia (redireccionando al formulario de alta) o la baja de la misma.*



*Formulario de alta de Personas.*

Simple formulario para agregar una persona con sus datos personales y el tipo de usuario al que pertenece(Alumno, profesor, etc).

Bienvenido al sistema Academia						
Trabao practico 2	Inicio   Inscripciones   Usuarios   Personas   Permisos   Materias					Cerrar sesion
	Usuario	Modulo	Alta	Baja	Modifica	Consulta
	Seleccionar admin	Personas	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Calvagna, Federico	Seleccionar admin	Materias	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Selva, Ignacio	Seleccionar admin	Usuarios	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	Seleccionar admin	Permisos	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Sanchez, Juan Manuel	Seleccionar admin	Inscripciones	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	Seleccionar juan	Personas	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	Seleccionar nacho	Permisos	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="button" value="Alta"/> <input type="button" value="Baja"/> <input type="button" value="Guardar"/> <input type="button" value="Cancelar"/>						

### Formulario de Gestión de Permisos.

La gestión de permisos es fundamental ya que proporciona qué módulos puede manejar el usuario. A su vez, se configura si el usuario puede agregar, remover, modificar o consultar los registros.

Escritorio



Usu	Clave	Legajo
admin	admin	0

**Nuevo Usuario**

Seleccione Persona: Calvagna, Federico

Usuario: federiquito

Clave: 1234

Cancelar Aceptar

Alta Baja Modifica

*Formulario de gestión y alta de Usuarios.*

*En este formulario se despliega una lista con todos los usuarios. Seleccionando a uno de ellos, modificando los campos y haciendo click en “modificar” se guardan los datos cambiados. En caso de hacer click en la opción “baja” eliminamos el usuario, y en caso de la opción “alta” aparece una ventana emergente en la cual muestra las personas sin usuario, y sus campos requeridos para completar, que se usarán para la validación en el ingreso al sistema.*

Usuario	Modulo	Alta	Baja	Modifica	Consulta
admin	Personas	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
admin	Materias	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
admin	Usuarios	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
admin		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
admin		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
juan		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
nacho		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

**Nuevo Permiso**

Seleccione Usuario: amandito2

Seleccione Modulo: Usuarios

Seleccione Permisos: ☒ Alta ☒ Baja ☒ Modifica ☐ Consulta

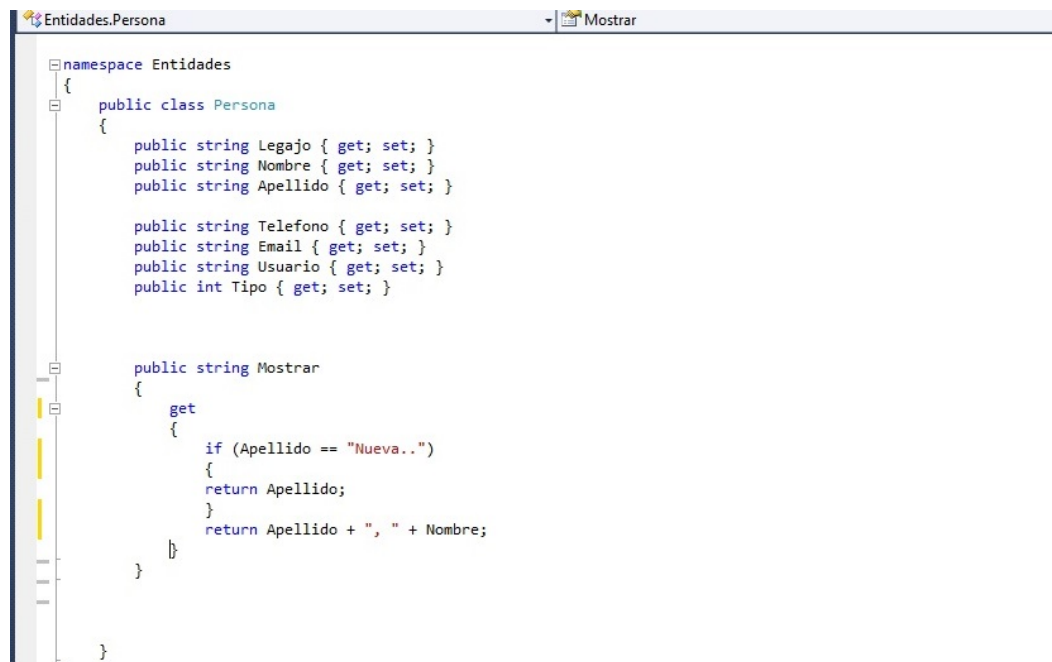
Aceptar Cancelar

Alta Baja Guardar Cancelar

*Formulario de gestión y alta de Permisos*

## Código

*A continuación se encuentran unas piezas de código a modo de ejemplo, las mismas no concluyentes respecto a la solución final, pueden sufrir algunas variaciones. Se recomienda, de todos modos, visualizar el código completo.*

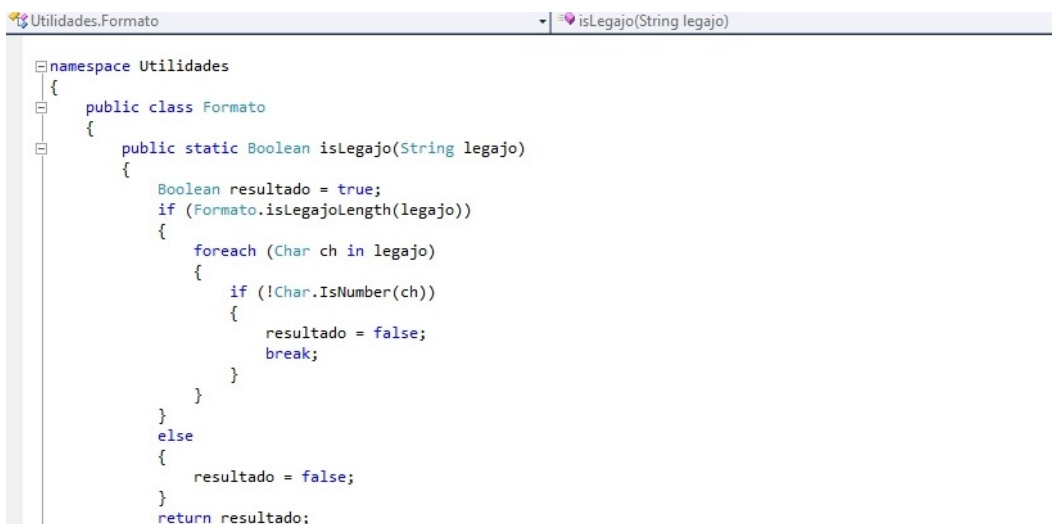


```
namespace Entidades
{
    public class Persona
    {
        public string Legajo { get; set; }
        public string Nombre { get; set; }
        public string Apellido { get; set; }

        public string Telefono { get; set; }
        public string Email { get; set; }
        public string Usuario { get; set; }
        public int Tipo { get; set; }

        public string Mostrar
        {
            get
            {
                if (Apellido == "Nueva..")
                {
                    return Apellido;
                }
                return Apellido + ", " + Nombre;
            }
        }
    }
}
```

*Clase Persona y sus Propiedades con sus get y set correspondientes*



```
namespace Utilidades
{
    public class Formato
    {
        public static Boolean isLegajo(String legajo)
        {
            Boolean resultado = true;
            if (Formato.isLegajoLength(legajo))
            {
                foreach (Char ch in legajo)
                {
                    if (!Char.IsNumber(ch))
                    {
                        resultado = false;
                        break;
                    }
                }
            }
            else
            {
                resultado = false;
            }
            return resultado;
        }
    }
}
```

*Del proyecto Utilidades, la clase Formato con un método estático que verifica que el parámetro sea un Legajo válido*

```

Presentacion.frmAltaUsuario frmAltaUsuario()
{
    public partial class frmAltaUsuario : Form
    {
        public frmAltaUsuario()
        {
            InitializeComponent();

            ControladorUsuarios cU = new ControladorUsuarios();
            bool activar = false;

            private void frmAltaUsuario_Load(object sender, EventArgs e)
            {
                cmbPersona.DataSource = cU.getPersonasSinUsu();
                cmbPersona.DisplayMember = "mostrar";

            }

            bool estaModificando = false;
    }
}

```

*Código de servidor de Formulario de Alta de Usuarios*

```

frmAltaPermisos btnCancelar_Click(object sender, EventArgs e)
{
    public partial class frmAltaPermisos : System.Web.UI.Page
    {
        Persona persona;
        ControladorPersonas controladorPersonas = new ControladorPersonas();
        ControladorPermisos controladorPermisos = new ControladorPermisos();
        ControladorUsuarios controladorUsuarios = new ControladorUsuarios();

        protected void Page_Load(object sender, EventArgs e)
        {
            persona = (Persona)Session["persona"];
            if (persona != null)
            {
                Usuario usuarioCorrespondiente = controladorPersonas.getUsuarioCorrespondiente(persona);
                Boolean permiso = controladorPermisos.getPermiso(usuarioCorrespondiente.Usu, "alta", "permisos");
                if (!permiso)
                {
                    Response.Redirect("~/frmPermisos.aspx");
                }
            }
        }
    }
}

```

*Código de servidor de Formulario de Alta de Permisos*

```

frmPermisos persona
{
    public partial class frmPermisos : System.Web.UI.Page
    {
        Persona persona;
        ControladorPermisos controladorPermisos = new ControladorPermisos();
        ControladorPersonas controladorPersonas = new ControladorPersonas();
        static Permiso[] permisos;

        protected void Page_Load(object sender, EventArgs e)
        {
            persona = (Persona)Session["persona"];
            if (persona != null)
            {
                Usuario usuarioCorrespondiente = controladorPersonas.getUsuarioCorrespondiente(persona);
                Boolean permiso = controladorPermisos.getPermiso(usuarioCorrespondiente.Usu, "consulta", "permisos");
                if (!permiso)
                {
                    Response.Redirect("~/frmPrincipal.aspx");
                }
            }
        }
    }
}

```

*Código de servidor de Formulario de Gestión de Permisos*

```

Eventos y objetos de cliente (No hay eventos)
<% Page Title="" Language="es" MasterPageFile="~/MasterPage.master" AutoEventWireup="true"
CodeFile="frmPermisos.aspx.cs" Inherits="frmPermisos" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
<style type="text/css">
.auto-style9 {
width: 80px;
}
.auto-style10 {
width: 83px;
}
</style>
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
<table style="width: 100%;">
<tr>
<td>
<asp:GridView ID="dgvPermisos" runat="server" CellPadding="4" ForeColor="#333333" GridLines="None" AutoGenerateColumns=
<AlternatingRowStyle BackColor="White" />
<Columns>
<asp:TemplateField HeaderText="Usuario">
<ItemTemplate>
<asp:Label ID="Usuario" runat="server" Text="<% Eval("Usuario") %>"></asp:Label>
</ItemTemplate>
</asp:TemplateField>
<asp:TemplateField HeaderText="Modulo">

```

### Código de cliente de Formulario de Gestión de Permisos

```

CapaDeDatos.CatalogoPersonas instancia
public void actualizarPersonas(Persona persona)
{
string actualizaString;
actualizaString = "UPDATE Persona SET ";
actualizaString += "Telefono=@Telefono, ";
actualizaString += "Nombre=@Nombre, Apellido=@Apellido, Email=@Email ";
actualizaString += "WHERE Legajo=@Legajo";
SqlCommand cmd = new SqlCommand(actualizaString, myCon);
cmd.Parameters.AddWithValue("@Legajo", persona.Legajo);
cmd.Parameters.AddWithValue("@Telefono", persona.Telefono);
cmd.Parameters.AddWithValue("@Nombre", persona.Nombre);
cmd.Parameters.AddWithValue("@Apellido", persona.Apellido);
cmd.Parameters.AddWithValue("@Email", persona.Email);
try
{
myCon.Open();
cmd.ExecuteNonQuery();
}
catch (Exception e)
{
Console.WriteLine(e.Message);
}
finally
{
myCon.Close();
}
}
}

```

### Consulta Actualizar Persona



```
CapaDeDatos.CatalogoMaterias instancia

    }

    public bool agregarMateria(string numero, string nombre)
    {
        int ok=-1;
        string query = "insert into Materia (Id_mat, Descripcion) ";
        query += "values (" + int.Parse(numero) + "," + nombre + " '));";

        SqlCommand cmd = new SqlCommand(query, myCon);
        cmd.CommandType = CommandType.Text;
        try
        {
            myCon.Open();
            ok = cmd.ExecuteNonQuery();
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
        }
        finally
        {
            myCon.Close();
        }
        if (ok > 0)
        {
            return true;
        }
        return false;
    }

    public bool eliminarMateria(string nombre)
    {
```

### *Consulta Agregar Materia*