

Trabajo Práctico N°3

Laboratorio 2

Hamer Thomas 2A

EXPLICACIÓN DEL PROYECTO

Este Proyecto simula ser un Sistema de Atención de Publico para un Gimnasio llamado Metro Flex Gym, en el que puede venir cualquier tipo de Publico para darse de Alta y comenzar a entrenar en el Gimnasio o Darse de Baja y dejar el mismo. También el Recepcionista podrá Modificar a estos Socios si ellos lo desean y podrá ver un Informe General del Gimnasio y Guardar estos datos en Distinto tipos de Archivos. El Socio podrá Elegir diferentes planes del Gimnasio y deberá dar todos sus datos para una Alta exitosa.

FUNCIONALIDAD

El Recepcionista deberá Seleccionar la Capacidad Máxima del Gimnasio para poder Cargar a sus Socios.



The screenshot shows a software window titled "Capacidad". Inside the window, there is a text prompt "Ingresa La Capacidad Maxima de Socios" (Enter the Maximum Capacity of Members). Below this prompt is a numeric input field containing the value "15". To the right of the input field is a small up/down arrow control. At the bottom of the window is a large button labeled "Aceptar" (Accept).

El Gimnasio ya viene por defecto con una Lista de Socios cargada.

MetroFlex Gym

Capacidad de Socios: 15
 Capacidad Libre: 9
 Total: \$ 15100

19:34:38
 4/6/2022

ID: 1 Nombre: Ronnie Apellido: Coleman Sexo: MASCULINO DNI: 24625100 Fecha de ingreso: 3/6/2022 Estatus: Activo Pase: Musculacion Pago: Credito
 ID: 2 Nombre: Jay Apellido: Cutler Sexo: MASCULINO DNI: 32782642 Fecha de ingreso: 3/6/2022 Estatus: Activo Pase: Gympass Pago: Efectivo
 ID: 3 Nombre: Tom Apellido: Platz Sexo: MASCULINO DNI: 17998120 Fecha de ingreso: 3/6/2022 Estatus: Activo Pase: Libre Pago: Debito
 ID: 4 Nombre: Chris Apellido: Bumstead Sexo: MASCULINO DNI: 35147890 Fecha de ingreso: 3/6/2022 Estatus: Activo Pase: Libre Pago: Debito
 ID: 5 Nombre: Larry Apellido: Wheels Sexo: MASCULINO DNI: 36529123 Fecha de ingreso: 3/6/2022 Estatus: Activo Pase: Gympass Pago: Efectivo
 ID: 6 Nombre: Noelia Apellido: Cvitanovic Sexo: FEMENINO DNI: 42897456 Fecha de ingreso: 3/6/2022 Estatus: Activo Pase: Libre Pago: Debito

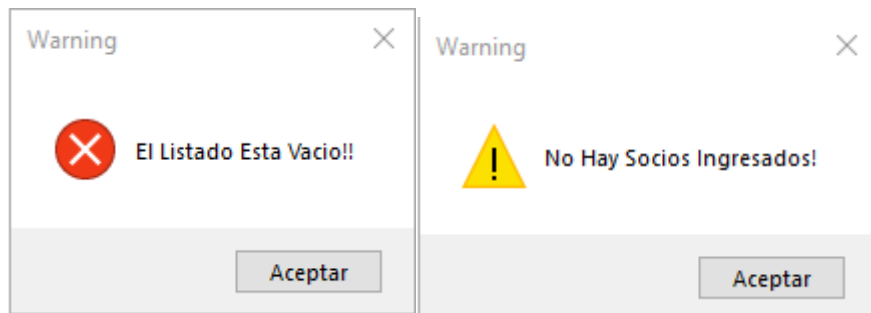
Añadir Socio Eliminar Socio Editar Socio Guardar Listado en XML Guardar Listado en Texto Informes Filtrar Datos
 Apellido/Nombre/Estatus/Pase

Si el Usuario del Proyecto desea Cargar su Propia lista de Socios, deberá Remplazar el Archivo "Socios.xml", el cual deberá tener mismo nombre y extensión.

↑ > Hamer.Thomas.2A.TPFinal > FormGimnasio

Nombre	Fecha de modificación	Tipo	Tamaño
bin	2/6/2022 01:42	Carpeta de archivos	
obj	4/6/2022 20:30	Carpeta de archivos	
Properties	4/6/2022 20:30	Carpeta de archivos	
FormGimnasio.csproj	3/6/2022 02:04	C# Project File	1 KB
FormGimnasio.csproj.user	2/6/2022 17:59	Per-User Project O...	1 KB
FrmAltaSocio.cs	4/6/2022 19:33	C# Source File	9 KB
FrmAltaSocio.Designer.cs	4/6/2022 19:33	C# Source File	15 KB
FrmAltaSocio.resx	4/6/2022 19:33	Microsoft .NET M...	345 KB
FrmCapacidad.cs	4/6/2022 18:55	C# Source File	1 KB
FrmCapacidad.Designer.cs	4/6/2022 18:54	C# Source File	5 KB
FrmCapacidad.resx	4/6/2022 18:54	Microsoft .NET M...	41 KB
FrmGym.cs	4/6/2022 19:43	C# Source File	15 KB
FrmGym.Designer.cs	4/6/2022 19:33	C# Source File	16 KB
FrmGym.resx	4/6/2022 19:33	Microsoft .NET M...	239 KB
FrmInformes.cs	4/6/2022 18:58	C# Source File	2 KB
FrmInformes.Designer.cs	3/6/2022 01:31	C# Source File	12 KB
FrmInformes.resx	3/6/2022 01:31	Microsoft .NET M...	196 KB
Program.cs	2/6/2022 18:00	C# Source File	1 KB
Socios.xml	3/6/2022 01:52	XML Document	2 KB

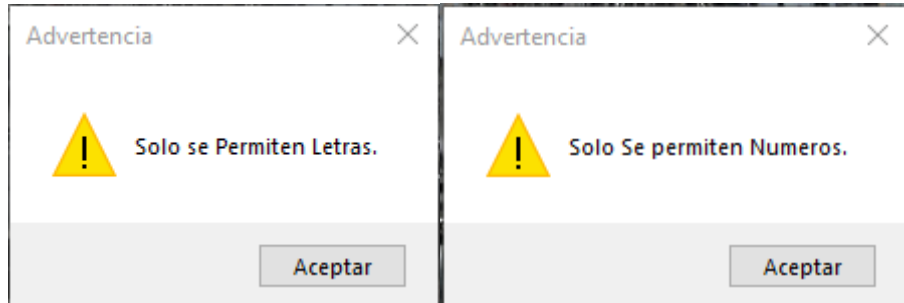
En el Caso que el Usuario intente Guardar una Lista Vacía en formato XML o Texto le saldrá una Advertencia informándole la Situación, También si el Usuario desea ver Informes del Gimnasio.



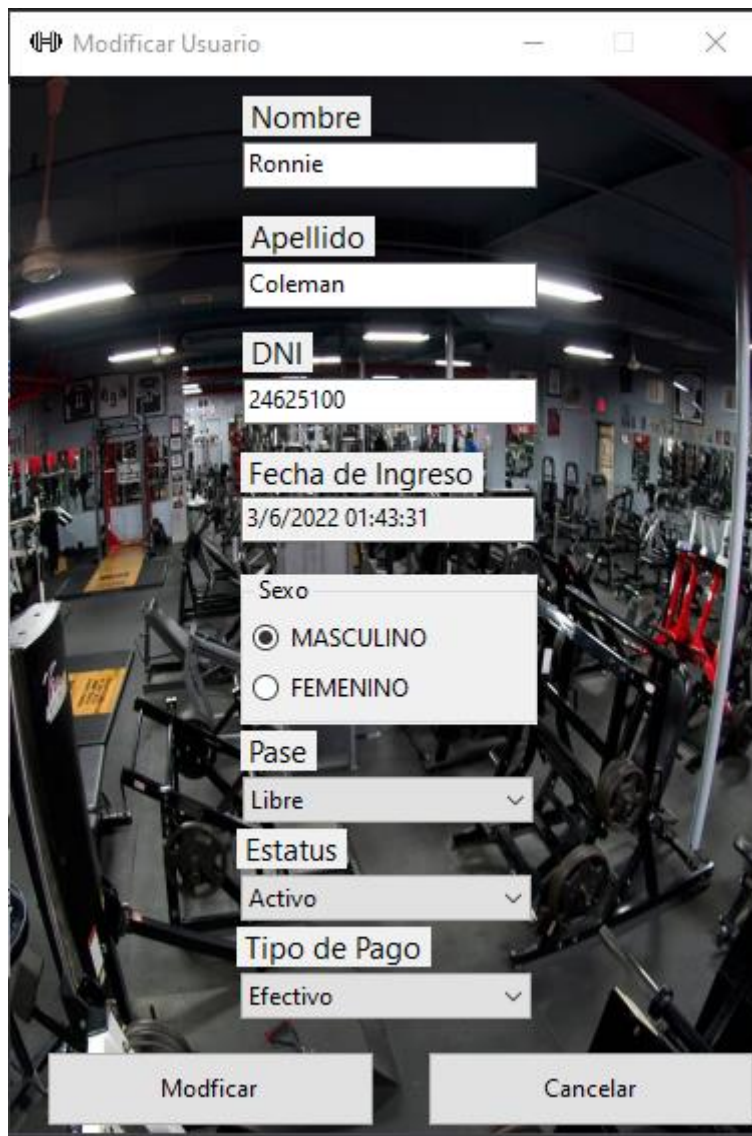
El Usuario del Proyecto Podrá dar el Alta de un Socio con sus datos.

A screenshot of a software window titled 'Alta Socio'. The window contains a form for adding a new member. The form fields are: 'Nombre' (text input), 'Apellido' (text input), 'DNI' (text input), 'Sexo' (radio buttons for 'MASCULINO' and 'FEMENINO'), 'Pase' (dropdown menu with 'Libre' selected), 'Estatus' (dropdown menu with 'Activo' selected), and 'Tipo de Pago' (dropdown menu with 'Efectivo' selected). At the bottom of the form are two buttons: 'Aceptar' and 'Cancelar'. The background of the window shows a gym interior with various exercise machines.

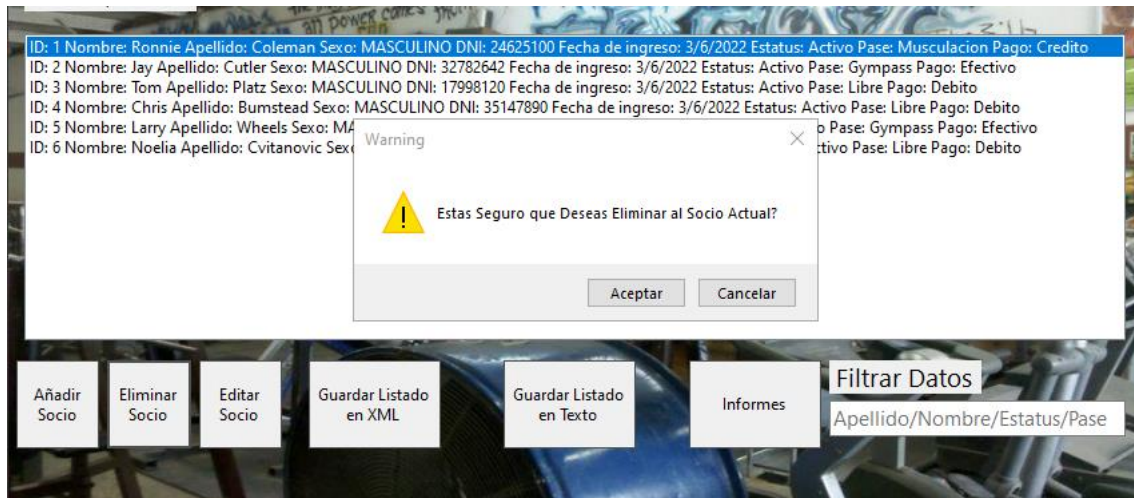
Si el Usuario de manera accidental ingresa un Numero en los Campos de Nombre y Apellido, Saldrá una Advertencia al Usuario informándole lo que debe ingresar, de la Misma manera pasa con el campo de DNI si se ingresa una Letra.



El Usuario podrá Modificar los datos de Cualquier Socio que desee.



Si el Usuario desea Dar de Baja a un Socio le saldrá una Advertencia para confirmar su Elección.



El Usuario podrá ver un Informe de los Socios del Gimnasio.



El Usuario podrá Filtrar Socio para facilitar su Búsqueda, a través del Apellido/Nombre/Estatus o Pase.

ID: 3 Nombre: Tom Apellido: Platz Sexo: MASCULINO DNI: 17998120 Fecha de ingreso: 3/6/2022 Estatus: Activo Pase: Libre Pago: Debito

Añadir Socio	Eliminar Socio	Editar Socio	Guardar Listado en XML	Guardar Listado en Texto	Informes	Filtrar Datos
						Tom

Clase 10 - Excepciones.

Implemento posibles Excepciones como por Ejemplo en caso que se sobrepase la Capacidad Máxima del Gimnasio.

```
namespace Entidades
{
    8 referencias
    public class CapacidadMaximaException : Exception
    {
        #region Constructores
        /// <summary> Crea un Mensaje con la Excepcion.
        1 referencia
        public CapacidadMaximaException(String mensaje) : this(mensaje, null)
        {
        }

        /// <summary> Crea un Mensaje con la Excepcion y la Inner Exception.
        1 referencia
        public CapacidadMaximaException(String mensaje, Exception inner) : base(mensaje, inner)
        {
        }
        #endregion
    }
}
```

```
        retorno = true;
    }
    else
    {
        throw new CapacidadMaximaException("No Hay Mas Lugares Disponibles!!!");
    }
    return retorno;
}
```

Clase 11 - Pruebas Unitarias.

Testeo los Principales Métodos del Proyecto y verifico que hagan lo correspondiente.

```
namespace TestUnitarios
{
    [TestClass]
    0 referencias
    public class TestGym
    {
        /// <summary> Valida que se puedan Agregar Socios Correctamente.
        [TestMethod]
        0 referencias
        public void AgregarSocios_Ok()...

        /// <summary> Valida que se Puedan Eliminar Socios Correctamente.
        [TestMethod]
        0 referencias
        public void EliminarSocios_Ok()...

        /// <summary> Valida que no se puedan Agregar Socios una vez Alcanzada La Capa ...
        [TestMethod]
        [ExpectedException(typeof(CapacidadMaximaException))]
        0 referencias
        public void AgregarSocios_Exception()...
    }
}
```

Clase 12 - Tipos Genéricos.

Implementado con Interfaces para Mayor practicidad.

```
namespace Entidades
{
    1 referencia
    public interface IAdministradorFiles<T>
    {
        2 referencias
        bool Exportar(T datos);
        2 referencias
        bool Importar(string ruta, out T datos);
    }
}
```

Clase 13 - Interfaces.

Implementada en la Serialización de Archivos.

```
namespace Entidades
{
    4 referencias
    public class Serializacion<T> : IAdministradorFiles<T>
        where T : class
    {
        private static string rutaArchivo;

        #region Metodos
        /// <summary>
        /// Arma la Ruta Donde se Guardara el Archivo.
        /// </summary>
        0 referencias
        static Serializacion()
        {
            string applicationData = Environment.GetFolderPath(Environment.SpecialFolder.Desktop);
            string nombreArchivo = "Socios.xml";
            rutaArchivo = Path.Combine(applicationData, nombreArchivo);
        }
        /// <summary>
        /// Exporta en XML Los Datos Pasados por Parametro.
        /// </summary>
        /// <param name="datos"></param>
        /// <returns></returns>
        2 referencias
    }
}
```

Clase 14 - Archivos y Serialización.

Utilizo Estos Principales Métodos para Serializar Archivos y poder Guardarlos tanto como XML o Txt.

```
private void ExportXml()
{
    serializador = new Serializacion<List<Socio>>();
    try
    {
        if (!string.IsNullOrEmpty(this.txtFiltro.Text))
        {
            this.gimnasio.lista = new List<Socio>(this.gimnasio.Filtrado.lista);
        }

        if (serializador.Exportar(this.gimnasio.lista))
        {
            MessageBox.Show("El Archivo ha Sido Generado con Exito." +
                " El Mismo se Encuentra en el Escritorio.", "Atencion", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            MessageBox.Show("Ocurrio un error.");
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Warning", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```



```

private void ExportTxt()
{
    SaveFileDialog saveFileDialog = new SaveFileDialog()
    {
        Filter = "TXT files|.txt",
        Title = "Guardar Archivo",
        InitialDirectory = Environment.GetFolderPath(Environment.SpecialFolder.Desktop),
        DefaultExt = ".txt",
        CheckPathExists = true,
        CheckFileExists = false,
        FileName = "Listado socios"
    };

    if (saveFileDialog.ShowDialog() == DialogResult.OK)
    {
        try
        {
            StreamWriter sw = new StreamWriter(saveFileDialog.FileName);
            sw.WriteLine("Listado Generado El: " + this.lblFecha.Text + "\n\n");

            foreach (var item in lstSocios.Items)
            {
                sw.WriteLine(item.ToString());
            }
            sw.WriteLine("\n\nTotal Facturado: " + this.lblTotalFacturado.Text);
            sw.Close();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message, "Warning", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
}

```

```

public bool Exportar(T datos)
{
    bool retorno = false;
    try
    {
        if (rutaArchivo != null)
        {
            using (StreamWriter streamWriter = new StreamWriter(rutaArchivo))
            {
                XmlSerializer nuevoXml = new XmlSerializer(typeof(T));
                nuevoXml.Serialize(streamWriter, datos);
                retorno = true;
            }
        }
    }
    catch (Exception)
    {
        throw new Exception("Error al Querer Guardar El Archivo: " + rutaArchivo);
    }
    return retorno;
}

```

```
public bool Importar(string ruta, out T datos)
{
    bool retorno = false;
    datos = default;

    try
    {
        if (ruta != null)
        {
            using (StreamReader auxReader = new StreamReader(ruta))
            {
                XmlSerializer nuevoXml = new XmlSerializer(typeof(T));
                datos = (T)nuevoXml.Deserialize(auxReader);
                retorno = true;
            }
        }
    }
    catch (Exception e)
    {
        throw new Exception(e.Message);
    }
    return retorno;
}
```