UNIVERSIDAD DE MÁLAGA
Departamento de Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA

Gestión de la Información
Práctica 7

Visual C#. Acceso a Base de datos con clases conectadas

Vamos a desarrollar una aplicación visual para trabajar con la base de datos del hotel que ya teníamos de las prácticas anteriores.

En esta práctica vamos a desarrollar la ventana de Login y el Mantenimiento de usuarios.

Se suministran la libreía **BDLibrary** y las clases:

- BDException: Clase para gestionar las excepciones de la aplicación.
- FLogin: Formulario para autentificarse como usuario
- FMenu: Menú principal de la aplicación.
- FUsuarios: Formulario de mantenimiento de usuarios.

SE RECOMIENDA mirar el código de dichas clases y se pide completar la clase Usuario para que el programa sea capaz de realizar la gestión de mantenimiento de usuarios.

```
public class Usuario
{
        public const int NO_REGISTRADO = -1;
        public const int INVITADO = 0;
        public const int USUARIO = 1;
        public const int ADMINISTRADOR = 2;

        private string _nombre;
        private string _pwd;
        private int _rol;

        public static List<string> ListaUsuarios()
        {
            // Método de clase que retorna la lista de todos los nombres de usuario

            List<string> lista = new List<string>();

            return lista;
        }

        public Usuario(string name)
        {
            // Constructor que rellena los atributos del usuario
            // leyéndolos de la base de datos a partir del nombre de usuario
            try
            {
            }
            catch (Exception ex)
            {
                _rol = Usuario.NO_REGISTRADO;
                throw new BDException(ex.Message);
            }
        }

        public Usuario(string name, string p)
        {
            // Constructor que rellena los atributos del usuario
            // leyéndolos de la base de datos a partir del nombre y contraseña de usuario
```

```
        try
        {
        }
        catch
        {
            throw new BDException("Nombre de Usuario o contraseña no válida");
        }
    }

    public void add(string n, string p, int r)
    {
        // Método que inserta un usuario en la base de datos.
        // Sólo se permite realizar esta operación a los administradores
        try
        {
            if (Rol != Usuario.ADMINISTRADOR)
                throw new BDException("Sólo los administradores pueden Insertar Usuarios");

        }
        catch (Exception ex)
        {
            throw new BDException(ex.Message);
        }
    }

    public string Nombre
    {
        // Propiedad que gestiona el nombre de usuario.
        // el get devuelve el valor del atributo
        // el set actualiza el atributo y lo modifica en la base de datos
        get
        {
            return _nombre;
        }
        set
        {
        }
    }

    public string Pwd
    {
        // Propiedad que gestiona la contraseña del usuario.
        // el get devuelve el valor del atributo
        // el set actualiza el atributo y lo modifica en la base de datos
        get
        {
            return _pwd;
        }
        set
        {
        }
    }

    public int Rol
    {
        // Propiedad que gestiona el rol de usuario.
        // el get devuelve el valor del atributo
        // no tiene set
        get
        {
            return _rol;
        }
    }

    public void modiRol(Usuario u, int newRol)
    {
        // Método que modifica el rol de un usuario en la base de datos.
        // Sólo se permite realizar esta operación a los administradores
```

```
    if (Rol != Usuario.ADMINISTRADOR)
            throw new BDException("El Rol sólo puede ser modificado por un administrador");


    u.reloadRol();
}

public void reloadRol()
{
    // Método que recarga el rol de un usuario leyendolo en la base de datos.
    try
    {

    }
    catch (Exception ex)
    {
        throw new BDException(ex.Message);
    }
}

public void del(Usuario u)
{
    // Método que borra un usuario en la base de datos.
    // Sólo se permite realizar esta operación a los administradores
    if (Rol != Usuario.ADMINISTRADOR)
      throw new BDException("Un usuario sólo puede ser borrado por un administrador");

}
```