Agregar =>Formulario;

Llama al formulario2 desde un botón en el formulario1.

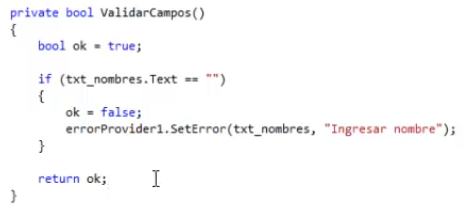
Crear una instancia de form2 y luego pone: nombredeinstancia.Show();

Mastil Visual

https://www.youtube.com/watch?v=kdEdKwgDduk&list=WL&index=38&t=31s

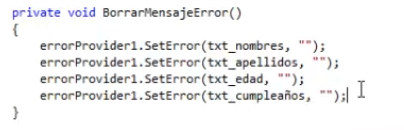
***Evento Error Provider***

Error Provider es un evento de Windows forms. Se lo manda al formulario, creo que en cualquier parte.

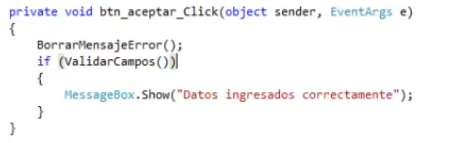


Si no ingresa ningún nombre, el booleano se convierte en falso y el error provider le pide que ingrese uno ¿dónde? En txt\_nombres.

Hecho eso (lo aplica a 4 campos mas), le da click al botón y saltan los errores al costado de los campos. Pero si después llena correctamente los campos, no desaparece el ícono del error. Para que desaparezca, hace:



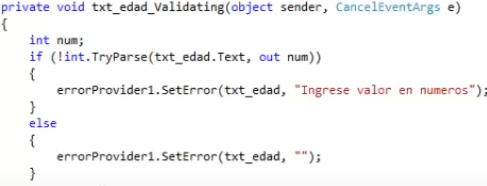
Pone un string en blanco. Luego llama al método, antes del otro método. En el evento del botón.



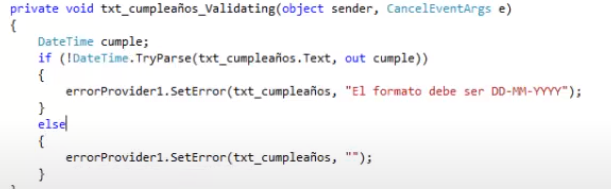
Ese if, hará que si todos los campos estén correctamente ingresados, aparezca el mensajito.

***Evento Validating***

El if, quiere decir que si el el texto ingresado en el campo no es un numero entero (con el out devuelve la variable num) lance el error provider



DateTime



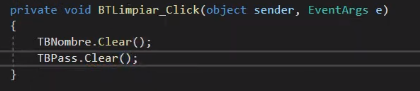
Lo mismo, pero con fechas.

Programacion FHOF

<https://www.youtube.com/watch?v=UkEx3vxNY4Q&list=WL&index=39>



Crea ese formulario para agregar sujetos al registro.

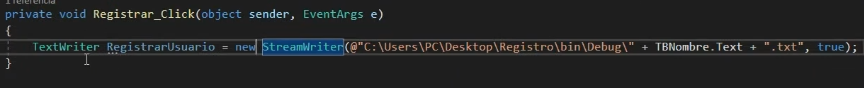


Con eso borra los campos.

***Agregar usuario***

Agrega librería using.system.IO

Usa textwriter. Crea una instancia. Luego decire con el streamwriter donde se lo va a almacenar. Elige la carpeta bin/debug. Copia todo y pone eso. Primero arroba y todo entrecomillado. Luego del debug va una barra diagonal. + nombre del campo que se quiere agregar+ entre comillas la extensión del archivo. Mas , true.

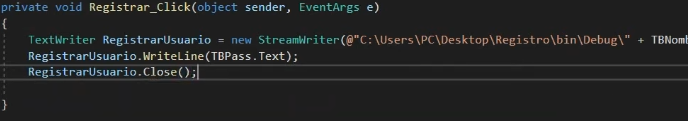


Con zoom:

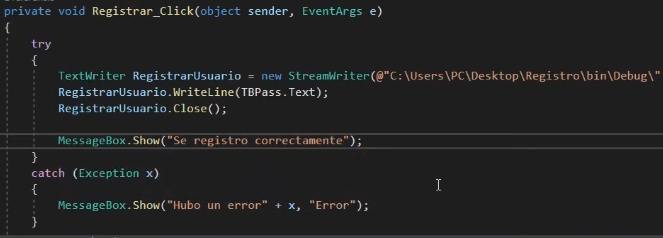


La otra línea dice que en registrar usuario, escriba el texto del campo contraseña.

Luego cierra el registro.



Luego pone todo dentro de un try/catch:

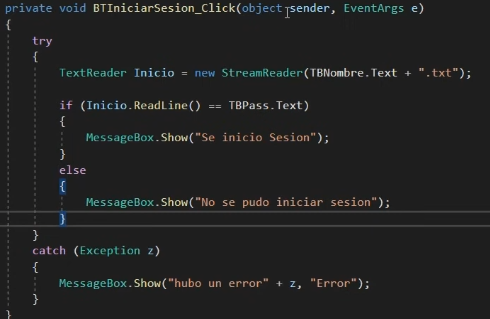


***Iniciar sesión***

Using system.IO

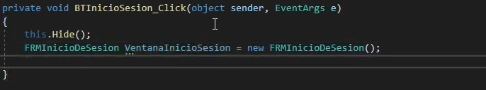
Acá se usa TextReader. Debe leer el archivo txt con el nombre que se le pasa

El if iguala lo que se escribió en el campo con lo que figura en el txt.  
También lo mete dentro del try/catch.



Crea el doc de texto en la carpeta mencionada y le pone como nombre Hola. En ese doc solo escribe 123. Luego, cuando va a iniciar sesión pone Hola y 123.

Luego, cuando haga un registro con la app será igual. Un doc llamado Pepe, con solo unos números en su interior.



***Con this.Hide(); se esconde el formulario actual.***

<https://www.youtube.com/watch?v=G05XNOIufM8>

Erick Recaman

Validar el usuario

Creo una tabla y le doy algunos valores

*picturebox*

Luego agrega a winforms un ***picturebox***. Ahí se agregan imágenes.

Con Sizemode se puede ajustar la imagen al tamaño del picturebox. Stretch le mandé.

Agrega los radiobuttons para seleccionar si se es administrador o empleados.

Luego crea otro formulario.

Les agrega tabcontrol con 2 pestañas. Cambia el text y el name. Toca la propiedad dock:fill. Agrega Panel, dentro del tabcontrol/consultar. También con dock.fill. Lo mismo con panEliminar.

Esto lo hace para facilitar que el empleado no pueda ver el la pestaña de tabcontrol, para eliminar empleados.

Hace planteo sobre la arquitectura de la app.

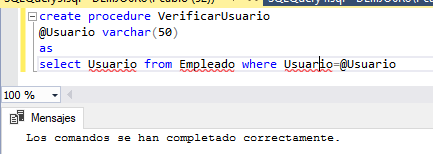
Afirma que todo lo hecho es la presentación. Entiendo yo que es vista.  
Crea un 3 nuevos proyectos. A los que llama BL (bussiness), DAL y EN (entity).

BL es la capa de enlace de las otras 2.

Dice que lo primero en crearse es entity/modelo.

Crea clase empleado desde el explorador de soluciones. A la clase Empleado la lla ENEmpleado. Luego crea DALEmpleado.

Va a SQLMANAGEMENT y desde la consulta crea un procedimiento llamado verificarusuario.



Luego eso puede revisarse en el explorador de objetos=>base de datos=>Programación=>Procedimientos almacenados

Agrega a DALEmpleados librerías de SQL:

using System.Data.Sql;

using System.Data.SqlClient;

También hace falta:

using System.Data;

Luego, abre la conexión:

SqlConnection Conexion = new SqlConnection("Data Source=DESKTOP-MSJO6R8; Initial catalog HC-Empleado; Integrated Security=true");

Conexion.Open();

Luego va al picLogin. Hace click, y en su evento codea:

String usuarioDIgitado = txtUsuario.Text;

Luego ese string se le pasa por parámetro al método verificarUsuario, donde se abre la conexión.

public String VerificarUsuario(String usuarioDigitado)

{

SqlConnection Conexion = new SqlConnection("Data Source=DESKTOP-MSJO6R8; Initial catalog HC-Empleado; Integrated Security=true");

Conexion.Open();

//Ahí se abre la conexión.

//Para verificar el usuario hacen falta las variables.

SqlDataAdapter adaptador;

//para que se necesita el adaptador?

//La BBDD habla lenguaje SQL, y el programa este habla C#

//El adaptador es el intermediario

DataSet datos = new DataSet();

adaptador = new SqlDataAdapter("execute VerificarUsuario " + usuarioDigitado, Conexion);

// primero se pone lo que se va a hacer. Despues se pasa el parámetro

//y despues se da el nombre de la conexión, la ruta.

//Ahí ya el adaptador está guardando la respuesta. Falta que esa respuesta la entienda C#

adaptador.Fill(datos, "Empleado");

//ahora, adaptador llena datos. Se agrega de qué tabla viene esa informacion

String usuarioValidado = datos.Tables[0].Rows[0]["Usuario"].ToString();

//Pone esto ya que es una matriz. Tables es la columna, rows es la fila.

return usuarioValidado;

//Recibe como parámetro usuarioDigitado y retorna usuariovalidado

}

Crea la clase BLEmpleado

Enlazar proyectos

Se hace click en el proyecto. Agregar referencia=>Solución y se marca el proyecto con el que se enlaza.

En este caso enlaza BL con DAL.

Antes de eso, no se podía llamar a las clases del otro proyecto.

Tambien enlaza BL con EN

class BLEmpleado

{

DAL.DALEmpleado dalempleado = new DAL.DALEmpleado();

public String VerificarUsuario (EN.ENEmpleado enempleado)

{

return dalempleado.VerificarUsuario(enempleado)

}

}

Luego cambia el código de DALEmpleado, ya que no recibe usuariodigitado sino todo el objeto ENEmpleado.

Antes hay que agregar a DAL la referencia EN. Así queda:

public String VerificarUsuario(EN.ENEmpleado enEmpleado)

{

SqlConnection Conexion = new SqlConnection("Data Source=DESKTOP-MSJO6R8; Initial catalog HC-Empleado; Integrated Security=true");

Conexion.Open();

SqlDataAdapter adaptador;

DataSet datos = new DataSet();

adaptador = new SqlDataAdapter("execute VerificarUsuario " + enEmpleado.EmpleadoUsuario, Conexion);

adaptador.Fill(datos, "Empleado");

String usuarioValidado = datos.Tables[0].Rows[0]["Usuario"].ToString();

return usuarioValidado;

}

Luego va al primer proyecto de los 4 creados y agrega referencias a BL y a EN

namespace WF\_\_ValidacionesYSQL

{

public partial class Form1 : Form

{

BL.BLEmpleado blempleado;

EN.ENEmpleado enEmpleado;

public Form1()

{

InitializeComponent();

blempleado = new BL.BLEmpleado();

enEmpleado = new EN.ENEmpleado();

}

Crea e inicia esos objetos y agrega esto en el botón del gatito:

private void picLogin\_Click(object sender, EventArgs e)

{

enEmpleado.EmpleadoUsuario = txtUsuario.Text;

String usuarioValidado = blempleado.VerificarUsuario(enEmpleado);

MessageBox.Show("El usuario digitado es: "+ enEmpleado.EmpleadoUsuario + " y el usuario validado es: "+ usuarioValidado);

}

No es que esté genial eso.

Ya funciona con esto. Le agrega los try/catch a DALEmplempleado.

public class DALEmpleado

{

public String VerificarUsuario(EN.ENEmpleado enEmpleado)

{

SqlConnection Conexion = new SqlConnection("Data Source= DESKTOP-MSJO6R8; Initial Catalog= HC-Empleados; Integrated Security= true");

Conexion.Open();

SqlDataAdapter adaptador;

DataSet datos = new DataSet();

try

{

adaptador = new SqlDataAdapter("execute VerificarUsuario " + enEmpleado.EmpleadoUsuario, Conexion);

adaptador.Fill(datos, "Empleado");

String usuarioValidado = datos.Tables[0].Rows[0]["Usuario"].ToString();

return usuarioValidado;

}

catch (Exception error)

{

// MessageBox.Show("El error es: " + error);

return null;

}

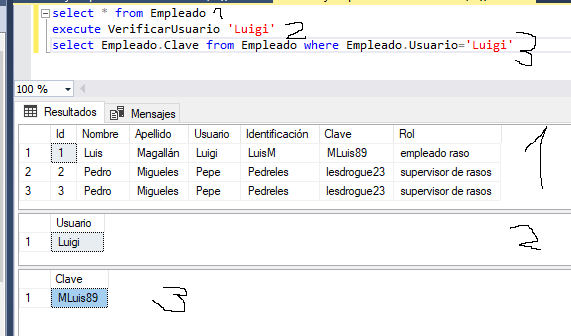
}

}

https://www.youtube.com/watch?v=6BWNAH7Hag8

Validar la clave

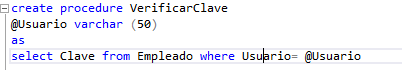
No es igual que validar el usuario. Para validar el usuario lo que se hace es fijarse en la base de datos si hay un usuario que coincida con lo que el usuario escribió. Acá no se puede hacer así. Primero se valida el usuario y después se fija si la clave escrita coincide con él.



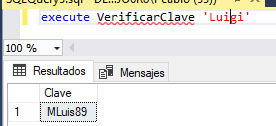
Ahí con lo escrito se obtiene el usuario y la clave de un empleado de la tabla Empleado en que el usuario sea (123456). Posteriormente este último valor sería el escrito en WF por el usuario.

Esa tercera línea es la que sirve para validar la clave. Luego la convierte en un procedimiento almacenado.

Recuerdalo papu. @ significa parámetro.



Ahí se crea el procedimiento. Luego se lo puede probar.



DALEmpleado

public String VerificarClave(EN.ENEmpleado enempleado)

{

SqlConnection Conexion = new SqlConnection("Data Source= DESKTOP-MSJO6R8; Initial Catalog= HC-Empleados; Integrated Security= true");

try

{

Conexion.Open();

}

catch (Exception error)

{

MessageBox.Show("Error al tratar de conectar con la base de datos \n"+ error);

}

//Pongo este try/catch, aunque parece al pedo.

SqlDataAdapter adaptador;

DataSet datos = new DataSet();

try

{

adaptador = new SqlDataAdapter("execute VerificarClave " + VerificarUsuario(enempleado), Conexion);

adaptador.Fill(datos, "Empleado");

}

catch (Exception error)

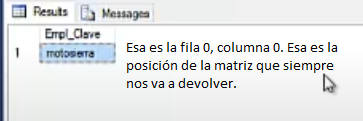
{

return null;

}

String ClaveValidada = datos.Tables[0].Rows[0]["Clave"].ToString();

return ClaveValidada;

}

BLEmpleado

public String VerificarClave (EN.ENEmpleado enempleado)

{

string ClaveVerifiqueta = dalempleado.VerificarClave(enempleado);

return ClaveVerifiqueta;

}

private void picLogin\_Click(object sender, EventArgs e)

{

enEmpleado.EmpleadoUsuario = txtUsuario.Text;

enEmpleado.EmpleadoClave = txtClave.Text;

String usuarioValidado = blempleado.VerificarUsuario(enEmpleado);

String claveValidada = blempleado.VerificarClave(enEmpleado);

MessageBox.Show("El usuario digitado es: "+ enEmpleado.EmpleadoUsuario + " y el usuario validado es: "+ usuarioValidado);

MessageBox.Show("La clave digitada es: "+ enEmpleado.EmpleadoClave + " y la clave validada es :"+ claveValidada);

}

A continuación de esto, Codea el tema del rol, que se establece en el radiobutton.

if (rbtAdmin.Checked == true) enEmpleado.EmpleadoRol = "Administrador";

if (rbtEmpleado.Checked == true) enEmpleado.EmpleadoRol = "Empleado";

Luego, finalmente, procede a llevar a cabo la verificación para ir al otro form.

if (usuarioValidado==enEmpleado.EmpleadoUsuario && claveValidada==enEmpleado.EmpleadoClave && enEmpleado.EmpleadoRol=="Administrador")

{

MenuPrincipal entrar = new MenuPrincipal();

entrar.Show();

this.Hide();

}

Ahí, si se selecciona administrador, se entra como administrador, lo mismo con empleado. El tema del rol, está manejado muy precariamente.

Evento Form closed

Genera el evento formclosed:

private void MenuPrincipal\_FormClosed(object sender, FormClosedEventArgs e)

//Funca cuando se le da a la X

{

Form1 VolverALogin = new Form1();

VolverALogin.Show();

this.Hide();

}

No se cierra la app, sino que se vuelve al formulario anterior.

Luego pasa a diferenciar el menúprincipal si sos administrador o empleado: si sos empleado, se te oculta la posibilidad de eliminar registros.

if (usuarioValidado==enEmpleado.EmpleadoUsuario && claveValidada==enEmpleado.EmpleadoClave && enEmpleado.EmpleadoRol=="Administrador")

{

MenuPrincipal entrar = new MenuPrincipal("Administrador");

entrar.Show();

this.Hide();

}

if (usuarioValidado == enEmpleado.EmpleadoUsuario && claveValidada == enEmpleado.EmpleadoClave && enEmpleado.EmpleadoRol == "Empleado")

{

MenuPrincipal entrar = new MenuPrincipal("Empleado");

entrar.Show();

this.Hide();

}

else MessageBox.Show("Error en el inicio de sesión");

En menú principal se ve como se oculta un panel si el parámetro que se le pasa desde form1 es “empleado”.

public partial class MenuPrincipal : Form

{

public MenuPrincipal(String rol)

{

InitializeComponent();

if (rol == "empleado") panEliminar.Visible = false;

}

private void MenuPrincipal\_FormClosed(object sender, FormClosedEventArgs e)

//Funca cuando se le da a la X

{

Form1 VolverALogin = new Form1();

VolverALogin.Show();

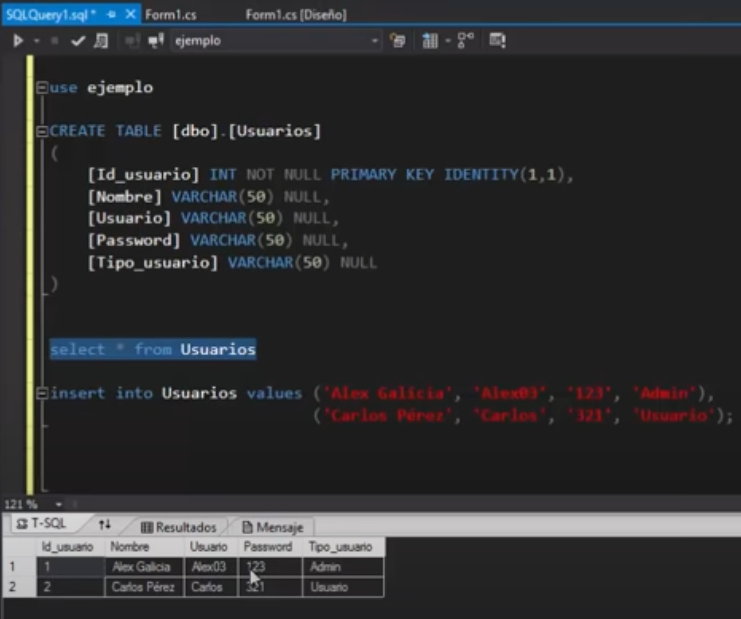
this.Hide();

}

}

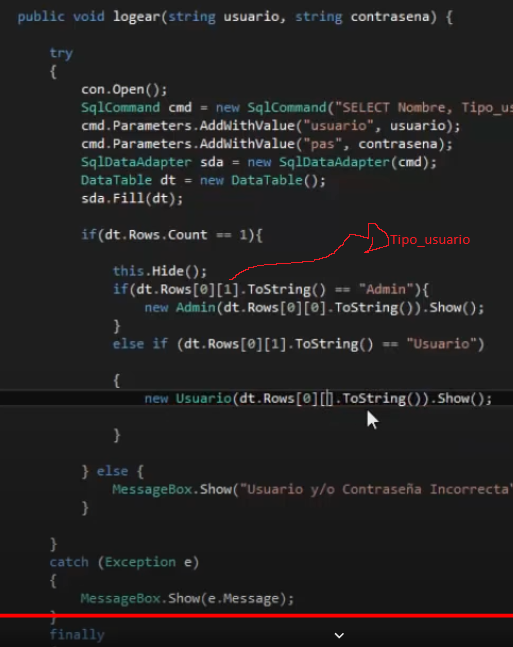
Crear tabla desde Visual Studio

El proyecto que se necesita no es de winforms ni de c#. Es SQL, lenguaje de consulta creo que se llama.



Este logueo devuelve menos información que el que hice antes. Además no están desacopladas las capas. Tampoco utiliza los procedimientos almacenados de SQL.

En el finally puse con.Close();



@= Parámetro