Guía técnica del proyecto Casa del Libro de la segunda evaluación del segundo curso de DAM

**Malik Korrich y Tomás Praga**

Contenido

[Introducción 4](#_Toc65098180)

[1.- Requisitos mínimos y software usado 5](#_Toc65098181)

[2.- Instalación y desinstalación 5](#_Toc65098182)

[2.1.- Instalación 5](#_Toc65098183)

[2.2.- Desinstalación 9](#_Toc65098184)

[3.- Definición del problema 9](#_Toc65098185)

[3.1.- Base de datos 9](#_Toc65098186)

[3.2.- Formulario de gestión de libros 13](#_Toc65098187)

[3.3.- Formulario de búsqueda o consultas 15](#_Toc65098188)

[3.4.- Formulario de gestión de stock 16](#_Toc65098189)

[3.5.- Formulario de gestión de clientes/socios 17](#_Toc65098190)

[3.6.- Formulario de gestión de empleados 18](#_Toc65098191)

[3.7.- Formulario de gestión de usuarios 19](#_Toc65098192)

[3.8.- Formulario de inicio de sesión 20](#_Toc65098193)

[3.9.- Formulario pantalla principal 21](#_Toc65098194)

[3.10.- Aspectos generales a los Formularios 22](#_Toc65098195)

[4.- Requisitos funcionales 22](#_Toc65098196)

[4.1.- Gestión de libros 22](#_Toc65098197)

[4.-2.- Gestión de Stock 22](#_Toc65098198)

[4.3.- Gestión de búsquedas 23](#_Toc65098199)

[5.- Requisitos no funcionales 23](#_Toc65098200)

[5.1.- Gestión de empleados 23](#_Toc65098201)

[5.2.- Gestión de socios 23](#_Toc65098202)

[5.3.- Gestión de usuarios 23](#_Toc65098203)

[6.- Variables 24](#_Toc65098204)

[7.- Procedimientos y funciones 26](#_Toc65098205)

[8.- Definición de ficheros 32](#_Toc65098206)

[9.- Bibliotecas/Librerías 32](#_Toc65098207)

[9.1.- Librería para todo lo relacionado con la base de datos: *libConexion* 32](#_Toc65098208)

[9.2.- Librería para gestión de validación de datos: *libValidacionDatos* 34](#_Toc65098209)

[10.- Diseño de entrada y salida 35](#_Toc65098210)

[11.- Diagramas 37](#_Toc65098211)

[11.1.- Diagrama Entidad relación de la base de datos 37](#_Toc65098212)

[11.2.- Diagrama de clases 37](#_Toc65098213)

[Librería de Validación de datos 37](#_Toc65098214)

[Librería de conexión a la base de datos 38](#_Toc65098215)

[12.- Casos de uso 38](#_Toc65098216)

[Tests generales a los formularios 39](#_Toc65098217)

[Validación de campos en formularios e inicio de sesión 40](#_Toc65098218)

[Pantalla principal 40](#_Toc65098219)

[13.- How to 41](#_Toc65098220)

[Validaciones 41](#_Toc65098221)

[Pantalla de inicio 41](#_Toc65098222)

[Pantalla principal 42](#_Toc65098223)

[Gestiones 42](#_Toc65098224)

[Terminología 42](#_Toc65098225)

[Anexo 43](#_Toc65098226)

[Modelo de informe de Libro, clientes y empleados 43](#_Toc65098227)

[Código fuente 43](#_Toc65098228)

# Introducción

En este manual, dirigido a personas con conocimiento técnico en la materia, se encuentra toda la información necesaria para conocer y poder usar el software Casa del Libro.

Se describe cuál es la petición del cliente (llamada en la guía técnica *definición del problema*), la solución al problema y los elementos de dicha solución.

Si tiene una duda sobre el funcionamiento o una parte del software, o tiene problemas a la hora de utilizarlo, le sugerimos que consulte este documento, ya que todos los elementos del programa están comprendidos aquí, desde la definición de tales elementos, su funcionalidad y su relación con otros elementos si la hubiera.

# 1.- Requisitos mínimos y software usado

Para instalar el siguiente software es necesario cumplir los siguientes requisitos mínimos de hardware y software:

* Sistema operativo Windows 8.1 o más reciente.
* Disco duro: 100MB de espacio libre.
* Memoria: 1GB RAM
* Procesador: 1.8 GHz Dual Core Procesador.
* Teclado y ratón.
* Pantalla de resolución 1920 \* 1080

El software usado para crear este proyecto ha sido:

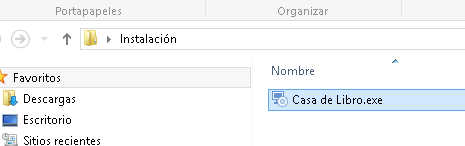
* Visual Studio 2017 - .Net Framework 4.6.1
* Microsoft Access 2007

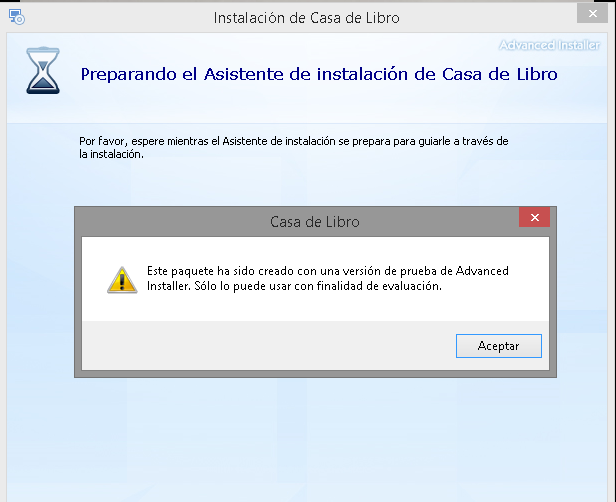
# 2.- Instalación y desinstalación

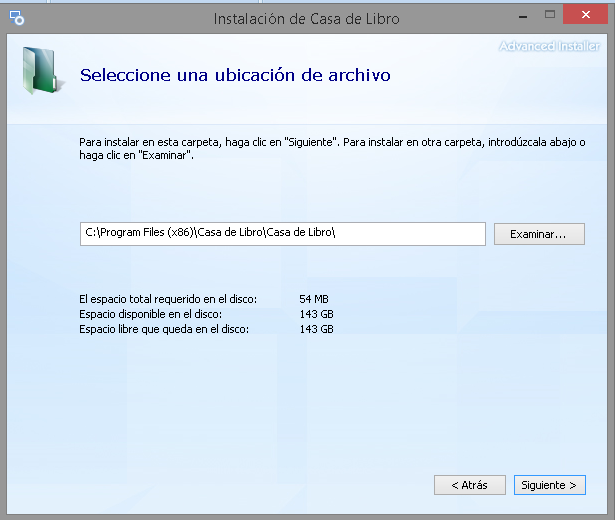
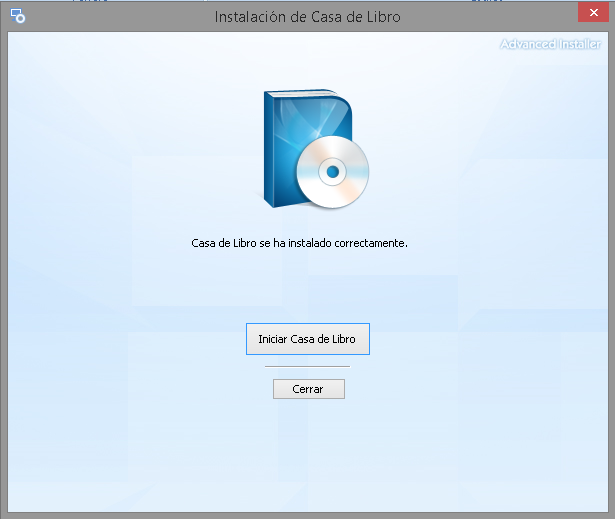
## 2.1.- Instalación

En esta sección se explica el proceso de instalación del software. Si desea salir de la instalación en cualquier momento, pulse la cruz en parte superior derecha de la ventana.

Los pasos de la instalación son los siguientes:

1. Haga doble click en el archivo ejecutable (de extensión .exe) llamado "Casa del Libro". Es posible que su antivirus le dé un aviso. Ignórelo, o deshabilite el antivirus y continúe con la instalación.
2. Es posible que le aparezca esta ventana, pulse aceptar:



1. Haga click en siguiente.
2. Para aceptar los términos debe por lo menos mover la barra de desplazamiento hasta abajo del todo. **Haga click en la casilla para aceptar** los términos de la licencia y pulse aceptar.
3. Elija la ruta donde desea instalar el programa. Cuando haya escogido la ruta, pulse siguiente.****
4. Pulse instalar. La instalación puede llevar más o menos tiempo dependiendo de las características de su equipo. Una vez acabe la instalación, puede ejecutar el programa.
5. **Si hace falta instalar una versión más reciente de .NET Framework el instalador se ocupará de eso. Este software usa la versión 4.6.1.**

Si no desea ejecutar el programa en ese momento, para hacerlo en otro momento haga doble click en el ejecutable que se haya en el directorio que se halla en la ruta de instalación indicada previamente.

Para acceder la primera vez se le provee con unos perfiles de empleado y usuario para que pueda crear otros según necesite. Solo se usa el nombre del usuario y una clave para acceder a la aplicación, pero se crea un empleado por defecto porque los usuarios precisan de uno para existir.

**El usuario es admin, y la contraseña es 123456.**

Si por algún motivo ha perdido acceso al programa, pulse la tecla alt (a la izquierda de la barra espaciadora) y la letra A al mismo tiempo. Se accederá al menú principal y se podrán crear empleados y usuarios para tener acceso otra vez a la aplicación.

## 2.2.- Desinstalación

Para desinstalar el programa siga los pasos siguientes:

1. Abra el panel de control (si no lo encuentra, abra la barra de windows y realice una búsqueda "panel de control" o directamente "desinstalar" -y seguramente le aparezca la ventana a la que debe ir)
2. Vaya a programas y luego a programas y características.
3. Seleccione el software Casa del libro, haga botón derecho y pulse desinstalar.

El programa se encargará de borrar todos los datos.

# 3.- Definición del problema

Según las especificaciones del cliente, Casa del Libro tiene que cumplir unos objetivos:

El software tiene que servir para consultar libros en una base de datos. Las búsquedas deben tener la opción de usar filtros.

El software debe dar la opción de gestionar los libros, empleados, usuarios y clientes. Gestión significa poder dar de alta (crear algo y guardar sus datos en la base de datos), baja(borrar de la base de datos) y modificar (modificar los datos de un elemento en la base de datos) un elemento deseado.

Para llegar a la solución del problema se han tomado las siguientes decisiones.

## 3.1.- Base de datos

Para este proyecto se ha usado una base de datos Access. Para su implementación se ha seguido el siguiente proceso:

1. Elección de un nombre para la base de datos: En este caso el nombre ha sido dado por el cliente: *cas\_lib\_bd.*
2. Diseño de la base de datos: La base de datos contiene tablas que guardan información de los libros, los empleados, los usuarios relacionados con esos empleados, los socios/clientes y el stock.

La base de datos se estructura de la siguiente forma (**entre paréntesis se escribe el nombre de la tabla según aparece en la base de datos**):

* Libros (*Libros*): Esta tabla almacena datos sobre los libros que hay en la tienda o en stock. La tabla está compuesta por los siguiente campos (salvo que se diga lo contrario, no admiten espacios):
  + *Id:* Un identificador de tipo entero(número sin puntos ni comas), positivo y único para cada libro. El campo en el formulario acepta hasta 7 dígitos (enteros, sin puntos ni comas).
  + *Isbn:* Tipo texto (varchar, String)El número ISBN de cada libro según la convención establecida. En la aplicación se valida al entrada de datos correcta de este dato.

**Ejemplos válidos:**

0672317249

9780672317248

Para más información.

(https://www.google.com/search?client=firefox-b- d&q=isbn+code)

* + *titulo*: Tipo texto (varchar, String).El título del libro.Tiene una longitud máxima de 25 caracteres y acepta todo tipo de caracteres y espacios.
  + *precio:* Tipo entero.Tiene un límite de 6 dígitos y acepta solo dígitos y puntos y comas
  + *autor:* Tipo texto (varchar, String). Tiene una longitud máxima de 25 caracteres y acepta todo tipo de caracteres y espacios.
  + *Editorial:* Tipo texto (varchar, String).Tiene un límite de 25 caracteres. Acepta todo tipo de caracteres y espacios.
  + *Tapa:* Tipo texto (varchar, String). Tiene un límite de 25 caracteres.Acepta todo tipo de caracteres y espacios.
  + *Stock:* Tipo entero(dígito sin puntos ni comas) y positivo .Aquí se introduce la cantidad en stock actual. Debe introducirse un número entero (sin comas ni puntos), y se permiten 4 cifras.
  + *genero:* Tipo texto (varchar, String).Tiene un límite de 60 caracteres.Acepta todo tipo de caracteres.
  + *Categoria*: Tipo texto (varchar, String).Tiene un límite de 50 caracteres.Acepta todo tipo de caracteres.
  + *Estado:* Tipo texto (varchar, String).Sirve para describir el estado nuevo, seminuevo de un libro. Tiene un límite de 30 caracteres. Acepta todo tipo de caracteres y espacios.
  + *Idioma:* Tipo texto (varchar, String).Tiene un límite de 30 caracteres.Acepta todo tipo de caracteres y espacios.
  + *Imagen:* Una imagen del libro. Las extensiones soportadas son .bmp (bitmap) y .jpg.
* Usuarios (*Usuarios*): Esta tabla comprende los datos de los usuarios con los que los empleados acceden a la aplicación. Esta tabla tiene una relación 1:1 (una a una) con la tabla empleados, ya que un empleado tiene un único usuario y viceversa. Los campos son los siguientes:
  + *Id:* Es un identificador único (no puede haber repetidos)Acepta un máximo de 4 dígitos (sin comas ni puntos)
  + *Usuario:*Debe ser un nombre en minúsculas o mayúsculas (no caracteres especiales, ni símbolos ni números) de hasta 10 caracteres de longitud.

No pueden repetirse los nombres de usuario.

* + *Clave:* Admite enteros positivos (sin comas ni puntos). Admite hasta 6 caracteres.

*Rol:* Dado que el cliente no ha especificado los permisos que desea dar a cada rol, se han establecido unos roles que tienen accesos a diversas partes del programa, a discreción de los programadores.

* El rol de Administrador tiene acceso a todas las partes del programa.
* El rol de usuario accede solamente a la gestión de libros, stock y -consultas de libros.
* El rol de Guest o invitado solo tiene acceso a la consulta de libros.
  + Fk\_empleado: debe ser un Id de la lista que no tenga ya asociado un usuario. Si quiere desligar un usuario de un empleado, deberá borrarlo o modificarlo (asignar el usuario a un empleado diferente).
* Empleados(*Empleados*): La tabla guarda datos sobre los empleados de la tienda. Cada empleado solo puede tener un usuario. Cada empleado tiene un Id y DNI únicos.
  + *Id:* Tipo entero(Sin comas ni puntos). Es el número identificador de cada empleado. Acepta un máximo de 4 dígitos (sin comas ni puntos).
  + *DNI:* Tipo texto (varchar, String)Este campo admite 8 números y una letra, al final. (no se permiten caracteres especiales y símbolos)

Es único, igual que el campo id.

* + *Nombre*: Tipo texto (varchar,String)Admite has 25 caracteres. No permite dígitos. Se permiten minúsculas, mayúsculas (ambos con tildes), espacios y guiones(-).
  + *Apellidos*: Tipo texto. Admite has 25 caracteres. No permite dígitos. Se permiten minúsculas, mayúsculas (ambos con tildes), espacios y guiones(-).
  + *Teléfono:* Tipo entero positivo.Admite hasta 9 dígitos (no permite comas ni puntos).
* Socios(*ClientesSocios*): Esta tabla contiene datos de clientes o socios. Los clientes pueden ser clientes que hagan una reserva, y el empleado guarda sus datos para poder contactar con la persona una vez que el libro o producto esté en la tienda. Si el cliente se convierte en socio, se marca la caja/checkbox (nombre de variable: *checkSocio* en el formulario *GestionClienteSocio)*en el formulario para indicar que tendrá los descuentos correspondientes.
  + Id: Identificador único. Acepta un máximo de 4 dígitos (sin comas ni puntos)
  + Nombre: El nombre del cliente o socio.permite un máximo de 25 caracteres, y permite minúsculas, mayúsculas, algunos caracteres especiales y espacios y guiones(-).
  + Apellidos: Los apellidos de socios. Igual que nombre pero permite 30 caracteres de longitud.
  + Teléfono: El número de contacto. Permite 9 dígitos (no comas ni puntos)
  + Email: Se sigue la convención normal. Una primera sección antes del arroba, una segunda sección después del arroba y la parte final de hasta 3 caracteres. La primera y segunda sección no admiten muchos símbolos (sí el guión y guión bajo) ni caracteres especiales (como al ñ), pero sí dígitos y caracteres en minúscula o mayúscula.
  + Fecha de alta: Se abre un calendario para que el usuario pueda seleccionar una fecha.
  + Socio(checkbox/caja para marcar): Debería marcarse si la persona tiene carnet de socio. En la base de datos pueden quedar guardados ex socios(sin la cajita marcada) que se hayan dado de baja, futuros socios a los que les falte información por cumplimentar, o quizás se desee tener la información de un cliente por una reserva, sin que llegue llegue a ser socio.
* Reservas (Reservas):

1. Hace falta importar unas librerías para poder usar la base de datos con el programa.

En este proyecto la conexión con la base de datos se realiza en la librería *libConexion*, situada en el directorio ProyectoDIDos(dentro de la carpeta con el mismo nombre).

Por tanto es en la biblioteca *libConexion* donde escribimos la sentencia:

*imports System.Data.OleDb*.

1. Hace falta un **conector** para conectar la base de datos con el programa.

En este caso usamos la clase *OleDbConnection* con un objeto de esa clase de nombre *conexion* y lo instanciamos pasándole la variable *conexionUrl* ,de dimensión local en la biblioteca, que contiene una cadena de texto: "Provider=Microsoft.ACE.OLEDB.12.0;DATA SOURCE=./cas\_lib\_db.accdb".

En este caso usamos "./" para indicar una ruta relativa y el nombre de la base de datos es el fichero usado como base de datos, en este caso *cas\_lib\_db.accdb*.

Notamos que se usa la versión 12 de OLEDB, si usamos una anterior, el programa no funcionará.

1. Para pode realizar consultas necesitamos un **adaptador** de datos. Seguimos en la librería *libConexion* donde ya tenemos la conexión a la base de datos.

Para el adaptador instanciamos un objeto OleDbDataAdapter y le pasamos por parámetro una consulta de tipo SQL., y como segundo parámetro le pasamos la variable que contiene la conexión. En este caso el objeto *OleDbDataAdapter* se llama *adaptador*.

1. Hace falta un **conjunto de datos** que se encontrará en la memoria caché para poder realizar operaciones en la base de datos original. Es decir, las operaciones que hacemos las realizamos con ese conjunto de datos y no directamente con la base de datos.

Este conjunto de datos es de tipo *DataSet* y en el proyecto las variables objeto de tipo *DataSet* se llaman *midataset*, *midataset2* y *midataset3*.(Ver apartado X sobre variables)

1. Con esto ya estamos preparados para realizar operaciones en la base de datos.

En este punto solo nos queda cargar el espacio de memoria reservado (el **dataset**) con los datos definidos en el **adaptador**. Para eso utilizamos el método *fill* del objeto de tipo *OleDbDataAdapter* que tenemos definido (en este caso llamado *adaptador*, consultar capitulo X de variables) y le pasamos por parámetro la variable de tipo *DataSet* que mencionamos anteriormente, y como segundo parámetro le pasamos un texto (tipo String), con el nombre que tendrá ese DataSet, esa tabla en la memoria.

1. Finalmente, para mostrar los datos de la base de datos se utiliza en todos los formularios un control de Visual Studio llamado *DataGridView*.

En este apartado 8 trabajamos ahora en los formularios, no en la biblioteca *libConexion*.

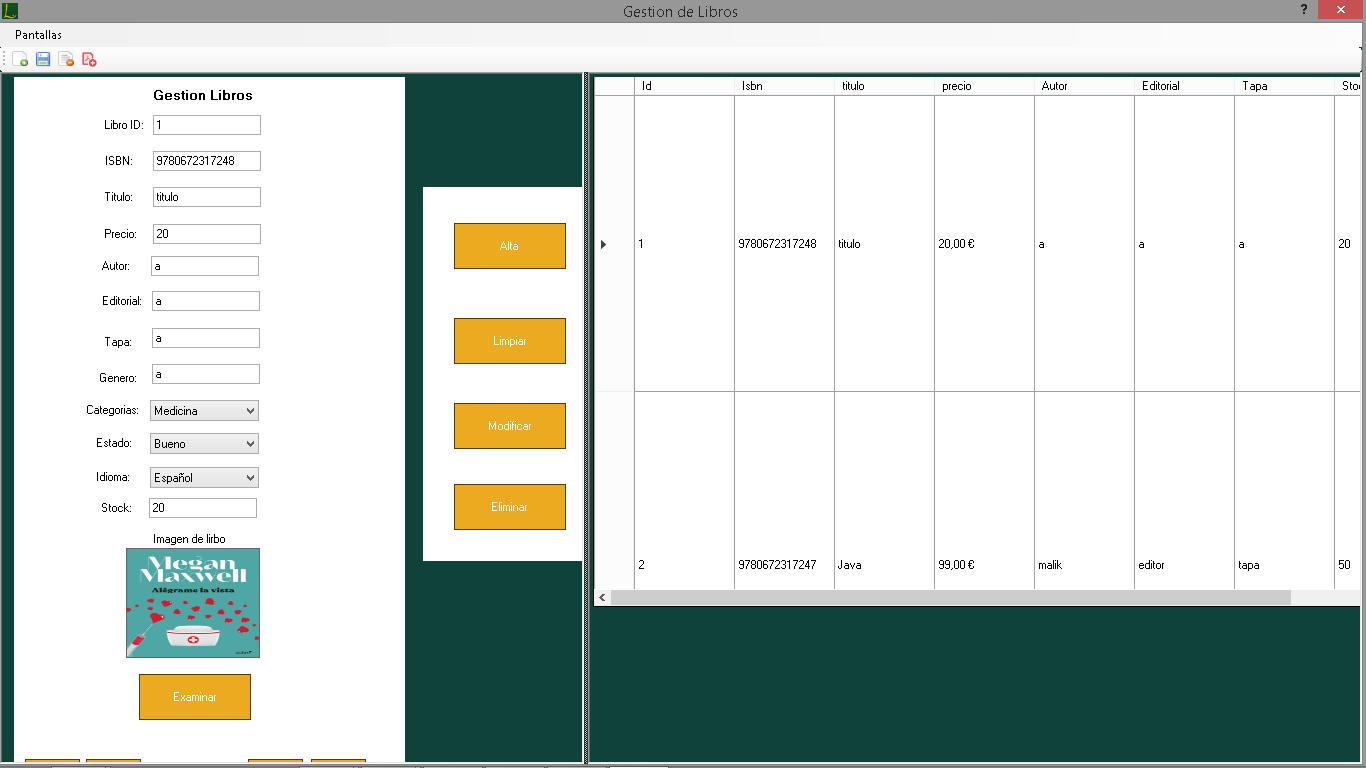
Para cargar en el *DataGridView* los datos que acabamos de guardar en el *DataSet*(el espacio de memoria que reservamos para la consulta que realizamos previamente), debemos asignar los datos del *DataSet* a la propiedad *DataSource* del control (el *DataGridView*), para ello usamos un objeto creado de tipo *libConexion* y llamamos al método *.getDataSource.*

Como el *DataSet* puede tener varias tablas, debemos indicar qué tabla queremos cargar para que se muestre. Para ello asignamos a la propiedad *DataMember* un texto (tipo *String*) con el nombre de la tabla (el que le dimos cuando creamos el *DataSet* no el de la tabla de la base de datos).

## 3.2.- Formulario de gestión de libros

En el formulario de gestión de libros, llamado *GestionLibro.vb*, se pretende lograr los siguientes objetivos: poder dar de alta, baja y modificar los libros de la base de datos.

**Los controles más importantes son:** la tabla que se ve en pantalla, que es un DataGridView (llamado *DataGridView1* en la *Solucion*, es decir en el proyecto*)*, los botones alta, modificar, eliminar y limpiar, llamados *btnAlta, btnModificar, btnEliminar y btnLimpiar* respectivamente.



Formulario *GestionLibro.vb*

**Si se necesita ayuda extra sobre algún control, el usuario puede pulsar el signo "?" en la parte superior derecha y luego hacer click en el elemento donde quiera recibir ayuda.**

El formulario tiene también una barra en el medio, dada por el componente *SplitContainer* llamado *SplitContainer1*. Con ella se puede arrastrar haciendo click con el raton en ella, para tener más visión de una de las mitades de la pantalla.

Funcionamiento del formulario:

Se proporciona un ejemplo y luego una explicación más detallada para realizar operaciones en esta ventana. Si desean introducir datos:

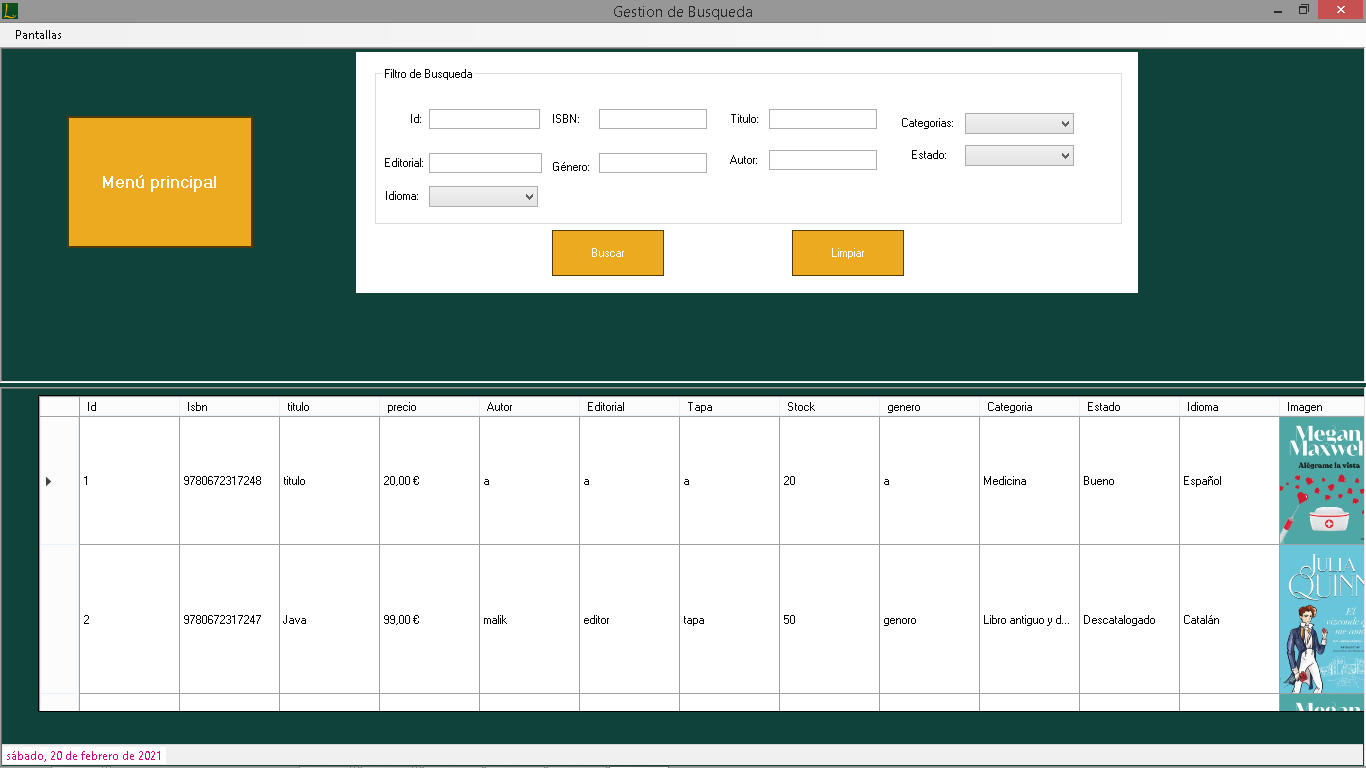
* Hacer doble click en la fila deseada en la tabla que se ve en pantalla.
* Introducir datos en los *textbox* para dar de alta o modificar un elemento.
* Pulsar el botón alta o modificar.
* Se recibirá un mensaje que notificará al usuario del resultado de la operación.

Pasamos a explicar las funcionalidades concretas halladas en este formulario:

* Dar de alta: Si se rellenan todos los campos y se pulsa el botón *alta* (en el proyecto su nombre es *btnAlta*) se añadirá una fila nueva en la tabla *Libros* de la base de datos *cas\_lib\_db.accdb*.
* Modificar: Si el usuario hace doble click en la tabla (*DataGridView1*) y modifica alguno de los campos (por ejemplo el llamado *txtId*) y luego pulsa el botón *Modificar*, se modificará la fila en tabla en la tabla *Libros*.
* Eliminar: Si se hace doble click en una fila en la tabla que se ve en pantalla y se pulsa el botón *Eliminar*, aparecerá un mensaje (*Messagebox* o *msgBox*) que preguntará al usuario si está seguro si quiere borrar la fila. En caso de pulsar *sí*, la fila se eliminará de la base de datos **permanentemente.**
* Limpiar: Limpia todos los campos de los valores que contengan para que el usuario pueda introducir los datos que desee.

## 3.3.- Formulario de búsqueda o consultas

El objetivo del formulario de consultas, llamado *GestionBusqueda*, es devolver información al usuario una vez que éste introduce datos en los campos *texbox* de búsqueda.



Formulario *GestionBusquedo.vb*

**Los controles más importantes son** el botón de Buscar (llamado *btnBuscar)* los *textbox* (todos), los *combobox*, y el *DataGridView* llamado *DataGridView1* que muestra el contenido de la tabla.

El formulario tiene también una barra en el medio, dada por el componente *SplitContainer* llamado *SplitContainer1*. Con ella se puede arrastrar haciendo click con el raton en ella, para tener más visión de una de las mitades de la pantalla.

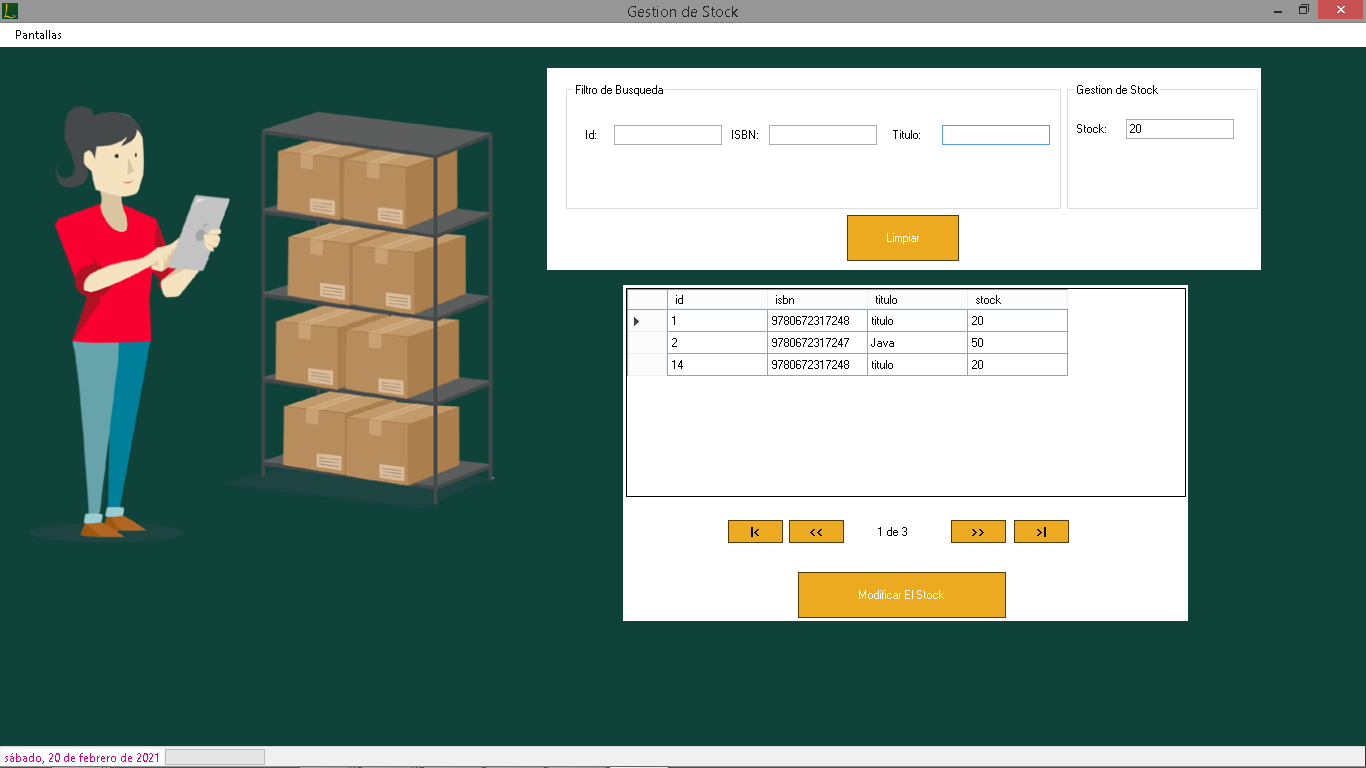
Funcionamiento del formulario:

* La búsqueda se realiza "en tiempo real", es decir, según se van introduciendo caracteres o dígitos el proceso de búsqueda se lleva a cabo y se actualiza el *DataGridView*(llamado *DataGridView1)*.
* El caso del campo del Id (el identificador único de los libros) es diferente, el proceso de búsqueda se lleva a cabo solamente cuando se pulsa el botón de *Buscar*, (llamadoo *btnBuscar)*.
* Se pueden introducir datos en varios campos para filtrar una búsqueda en más detalle.

## 3.4.- Formulario de gestión de stock

En este formulario, llamado *GestionStock.vb*, se puede modificar el stock de cada libro.

**Es importante subrayar que solo en este formulario se puede modificar el stock, aunque en el Formulario *GestionLibro.vb* haya un *textbox* en el que se pueden introducir datos.**



Formulario *GestionStock.vb*

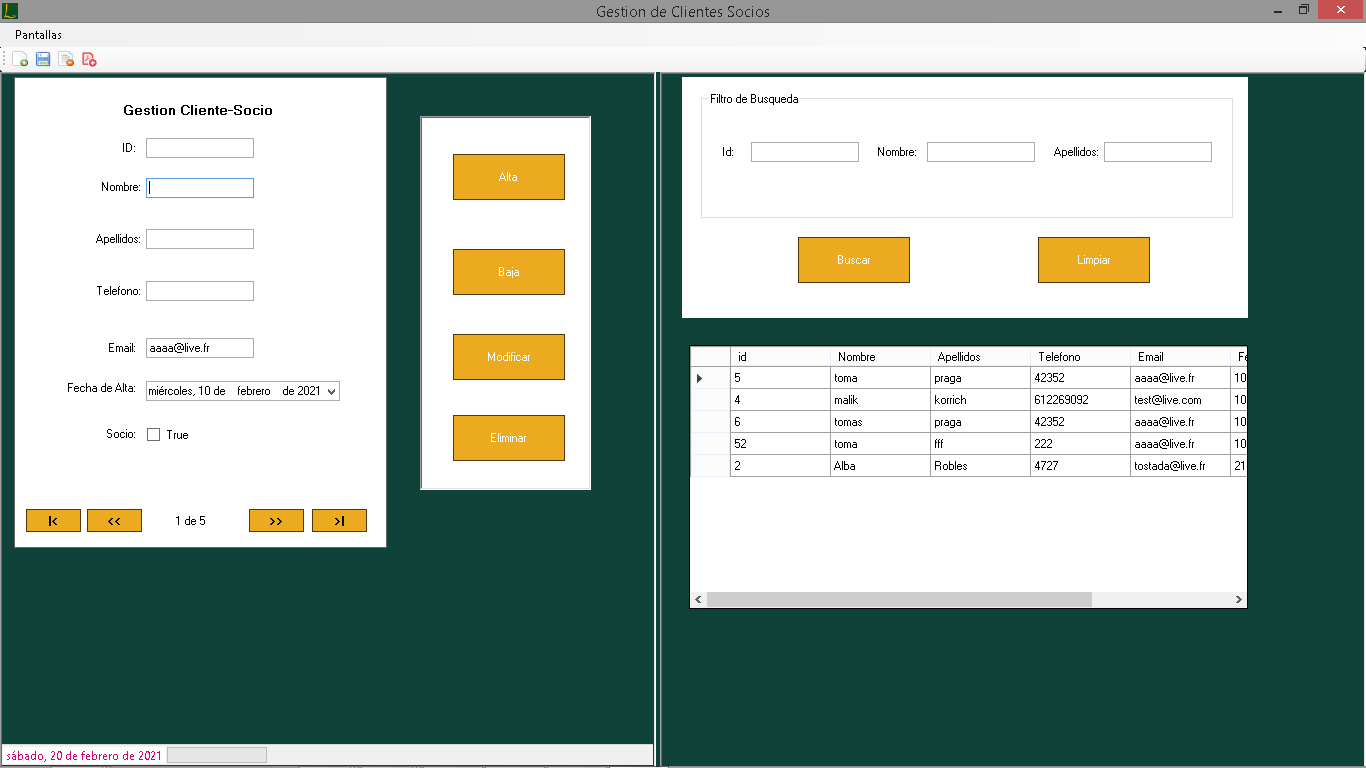
**Los componentes más importantes** del Formulario son los *textbox* de búsqueda, el *textbox* llamado txtStock que es donde se introduce la cantidad de stock que se desea modificar, el botón llamado *btnModificarStock* que hay que pulsar para que la modificación se lleve a cabo, y el *DataGridView* llamado *DataGridView1* que muestra el contenido de la tabla.

Funcionamiento del formulario:

* Se debe hacer doble click en la fila de la tabla que nos interese.
* Modificamos el valor del campo de Stock (el *textbox* llamado *txtStock)*
* Presionamos el botón Modificar stock.
* Un ventana nueva aparecerá con texto notificando al usuario sobre el resultado de la operación.

## 3.5.- Formulario de gestión de clientes/socios

En este Formulario llamado *GestionClienteSocio.vb* se gestionan los clientes y socios de la tienda. Se pueden realizar operaciones como alta, modificación y baja.



Formulario *GestionClienteSocio.vb*

**Los controles más importantes son**: la tabla que se ve en pantalla, que es un DataGridView (llamado *DataGridView1* en la *Solucion*, es decir en el proyecto*)*, los botones alta, modificar, eliminar y limpiar, llamados *btnAlta, btnModificar, btnEliminar y btnLimpiar* respectivamente.

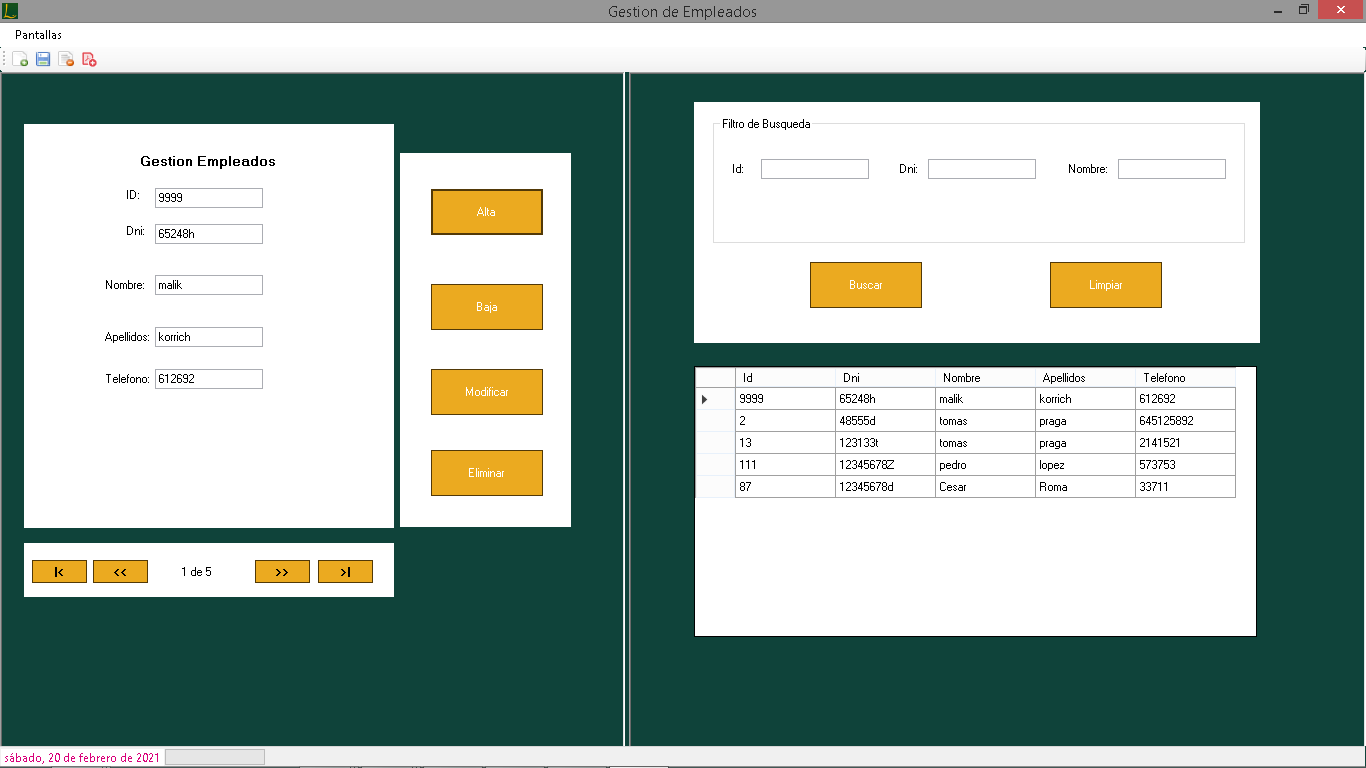
El formulario tiene también una barra en el medio, dada por el componente *SplitContainer* llamado *SplitContainer1*. Con ella se puede arrastrar haciendo click con el raton en ella, para tener más visión de una de las mitades de la pantalla.

Funcionamiento del formulario:

* Para gestionar la distinción entre clientes y socios se usa un *checkbox* llamado *checkSocio*. Al estar marcado este *checkbox* (o sea, true) el cliente es socio y obtiene un descuento. Si no está marcado, es un cliente que puede haber hecho una reserva o un futuro socio al cual le faltan campos por rellenar.
* Para realizar una búsqueda simplemente se deben introducir datosen los *textbox.*
* Para dar de alta hay que poner valores en los *textbox*  y dar a botón Alta (*btnAlta*).
* Para modificar hace falta dar doble click en el *DataGridView* en la fila que nos interese, cambiar los valores de los *textbox* y presionar el botón modificar.
* Para eliminar hace falta dar doble click para seleccionar la fila que nos interese en el *DataGridView* y presionar el botón Eliminar (*btnEliminar)*.

## 3.6.- Formulario de gestión de empleados

El formulario de gestion de empleados, llamado *GestionEmpleados.vb*, busca, como en otros Formularios de gestión de este proyecto, dar la posibilidad de llevar a cabo operaciones de alta, modificación y eliminación en la base de datos *cas\_lib\_db.accdb*.



Formulario *GestionEmpleados.vb*

**Los componentes más importantes son:** la tabla que se ve en pantalla, que es un DataGridView (llamado *DataGridView1* en la *Solucion*, es decir en el proyecto*)*, los botones alta, modificar, eliminar y limpiar, llamados *btnAlta, btnModificar, btnEliminar y btnLimpiar* respectivamente.

El formulario tiene también una barra en el medio, dada por el componente *SplitContainer* llamado *SplitContainer1*. Con ella se puede arrastrar haciendo click con el raton en ella, para tener más visión de una de las mitades de la pantalla.

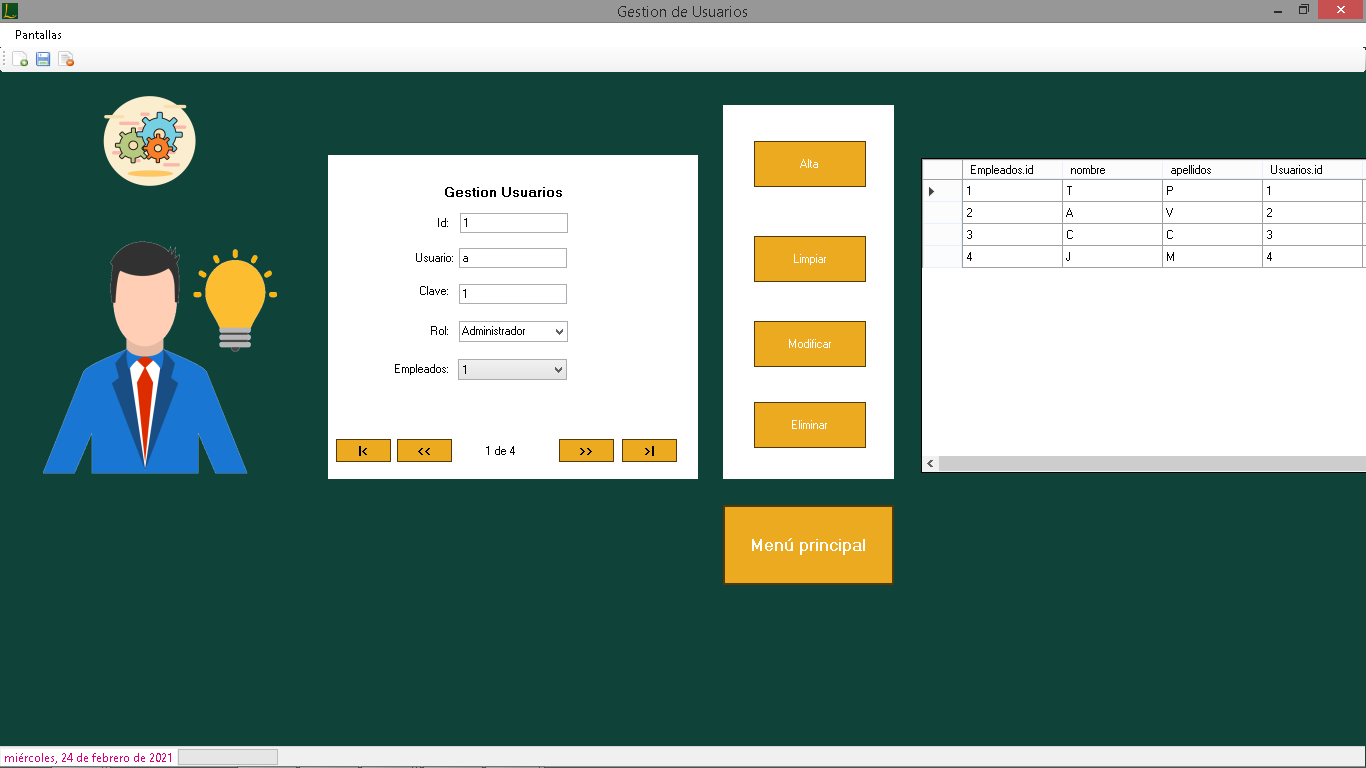
Funcionamiento del Formulario:

* Para realizar una búsqueda simplemente se deben introducir datosen los *textbox.*
* Para dar de alta hay que poner valores en los *textbox*  y dar a botón Alta (*btnAlta*).
* Para modificar hace falta **dar doble click** en el *DataGridView* en la fila que nos interese, cambiar los valores de los *textbox* y presionar el botón modificar.
* Para eliminar hace falta dar doble click para seleccionar la fila que nos interese en el *DataGridView* y presionar el botón Eliminar (*btnEliminar)*.

## 3.7.- Formulario de gestión de usuarios

En la gestión de usuarios, llamado *GestionUsuarios.vb* , se puede crear el usuario que usará el empleado para acceder a la aplicación.

**Los componentes más importantes son:** la tabla que se ve en pantalla, que es un DataGridView (llamado *DataGridView1* en la *Solucion*, es decir en el proyecto*)*, los botones alta, modificar, eliminar y limpiar, llamados *btnAlta, btnModificar, btnEliminar y btnLimpiar* respectivamente.



*Ventana de gestión de usuarios*

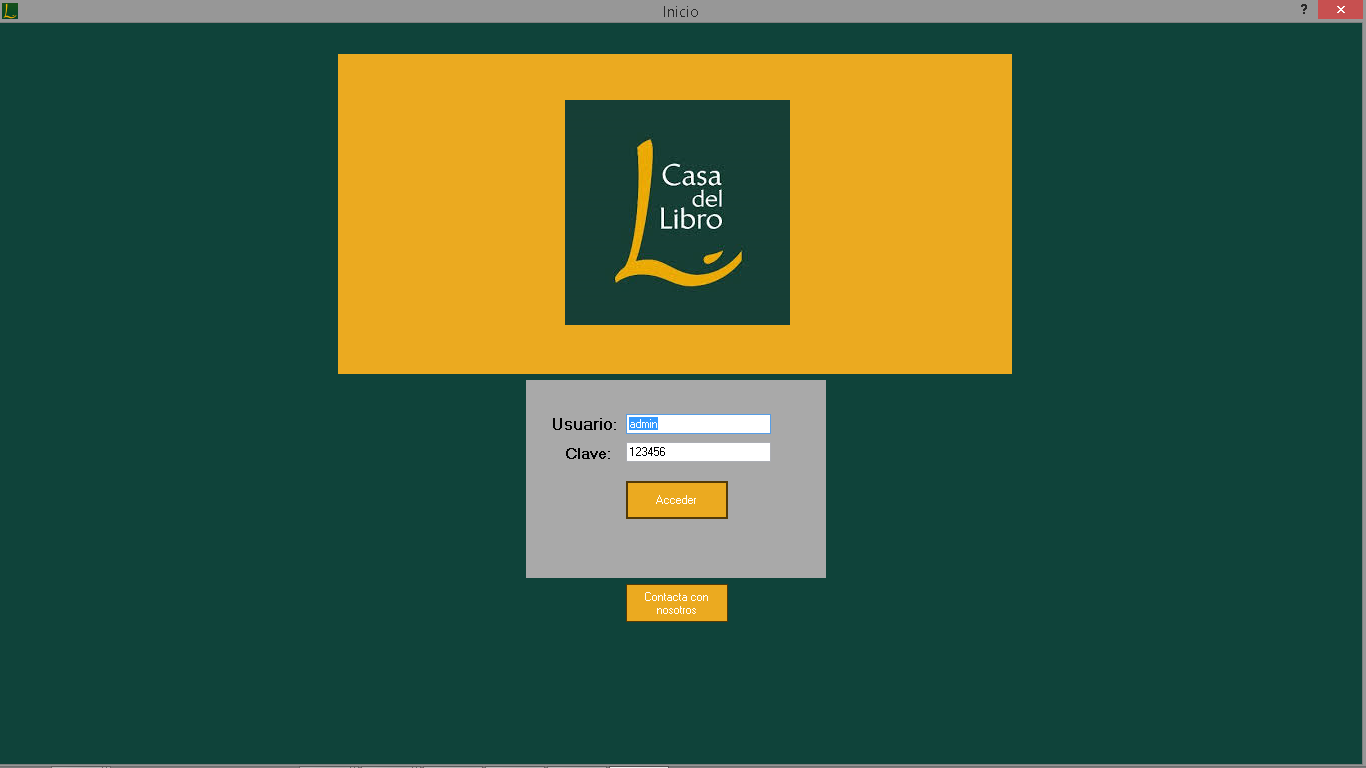
Funcionamiento del Formulario:

* Para realizar una búsqueda simplemente se deben introducir datosen los *textbox.*
* Para dar de alta hay que poner valores en los *textbox*  y dar a botón Alta (*btnAlta*).
* Para modificar hace falta **dar doble click** en el *DataGridView* en la fila que nos interese, cambiar los valores de los *textbox* y presionar el botón modificar.
* Para eliminar hace falta dar doble click para seleccionar la fila que nos interese en el *DataGridView* y presionar el botón Eliminar (*btnEliminar)*.

## 3.8.- Formulario de inicio de sesión

En este Formulario, llamado *PantallaInicio.vb*, el usuario debe poner unos datos que deben coincidir con unos existentes en la base de datos para poder pasar a la siguiente ventana, la del menú principal, y por tanto tener acceso al resto de la aplicación.

Si se pierden todos los usuarios por el motivo que sea, se puede acceder a la aplicación para recuperar el control de la misma usando: alt + a, en la pantalla de inicio.



Pantalla de inicio *PantallaInicio.vb*

Los componentes más importantes de este Formulario son el los *textbox* llamados *txtUsuario* y *txtClave*, y el botón para acceder a la aplicación llamado *btnAcceder*.

Funcionamiento del Formulario:

El usuario debe introducir datos en ambos campos (y que coincidan con datos existentes en la abse de datos *cas\_lib\_db.accdb*) para poder acceder a la pantalla siguiente, que es la del menú que contiene el acceso al resto de funcionalidades del software.

Dependiendo del rol que tenga el usuario, en esta pantalla se decide con los métodos .*enable* para los controles qué botones del menú principal funcionan o no.

## 3.9.- Formulario pantalla principal

En este Formulario, llamado *PantallaPrincipal.vb* podemos acceder al resto de funcionalidades del programa. Por tanto esta pantalla sirve, sobre todo, para acceder a otros Formularios.



Pantalla *PantallaPrincipal.vb*

Principales componentes: botones llamados *btnGestionLibros* (para ir a la pantalla de gestion de libros),*picUsuarios*(que al ser pulsado llevar a la pantalla de gestión de usuarios)*, picEmple*(que al ser pulsado lleva a la pantalla de gestión de empleados)*, picStock*(que al ser pulsado lleva a la pantalla de gestión de Stock), *picSocios*(que al ser pulsado lleva a la ventanan de gestión de socios/clientes), *btnGestionBusqueda*(para ir a la pantalla de búsqueda de libros).

Funcionamiento del Formulario:

* Para ir a otro Formulario (gestión de stock, usuarios, etcétera) solo hay que pulsar en los botones (en realidad, son componentes *PictureBox*, para no tener problemas con las dimensiones de las imágenes).
* Para cerrar sesión para poder entrar con otro usuario, se debe pinchar en el *MenuStrip* llamado *Menustrip1* y pinchar en cerrar sesión. De esa forma se podrá acceder con otro usuario, que puede tener otros roles diferentes y la aplicación se actualizará acorde a los nuevos permisos concedidos.

## 3.10.- Aspectos generales a los Formularios

Aquí se mencionan elementos comunes a todos los formularios (excepto en las pantallas de informes, que tienen en común que tienen una barra de herramientas con un botón que los lleva a la pantalla anterior correspondiente).

* Barra de herramientas *MenuStrip*  llamada *MenuStrip1*.

Contiene un comando llamado *OpcionesToolStripMenuItem* que tiene más *ToolStripMenuItem*, uno por cada Formulario para que el usuario pueda ir de un Formulario a otro, sin tener que pasar por el menú principal en su ruta.

La excepción es el menú principal (*PantallaPrincipal.vb*) que en su barra de herramientas tiene dos comandos y son para cerrar sesión y para cerrar sesión y salir de la aplicación.

* Barra de herramientas *ToolStrip* llamada *ToolStrip1* (Excepto en los siguientes formularios: Pantalla de inicio (*PantallaInicio.vb*) y Pantalla principal o menú principal (*PantallaPrincipal.vb)*).

Esta barra de herramientas tiene uno botones cuya funcionalidad es la misma que los botones (por ejemplo *btnAlta* para dar de alta un libro) pero que se han establecido en el proyecto para facilitar al usuario maneras diferentes de usar la aplicación.

# 4.- Requisitos funcionales

Se han considerado como requisitos funcionales los siguientes:

1. Búsqueda o consulta de libros en la base de datos
2. Gestión de libros en la base de datos (altas, bajas, modificaciones).
3. Gestión de stock de los libros en la base de datos.

## 4.1.- Gestión de libros

L a gestión (alta, eliminación, modificación) de los libros es una parte esencial del programa, dado que son la base sobre la cual se apoyan el resto de características funcionales.

Sin libros no hay búsquedas. Aunque las búsquedas también son esenciales.

## 4.-2.- Gestión de Stock

Ir y venir continuamente del almacén para comprobar el stock requeriría de un tiempo valioso para los empleados. Por tanto se considera que el control del stock es una funcionalidad imprescindible en el programa.

La gestión de stock significa modificar el valor del stock de cada libro.

## 4.3.- Gestión de búsquedas

La búsqueda es una consulta a la base de datos que devuelve información al usuario.

Es imprescindible que se puedan buscar libros en la base de datos y obtener la información de forma clara y rápida.

Es la funcionalidad más importante, dado que aparece en varios Formularios e incluso tiene uno propio.

# 5.- Requisitos no funcionales

La aplicación cumple su objetivo con los elementos mencionados en los requisitos funcionales. Por tanto, se han considerado como requisitos no funcionales los siguientes:

1. Gestión de empleados
2. Gestión de socios
3. Gestión de usuarios

## 5.1.- Gestión de empleados

La aplicación podría cumplir un objetivo por sí misma con la búsqueda de libros en la base de datos y el control del stock. Por lo que la gestión del personal es un añadido.

La gestión de los empleados es una ayuda más para el cliente, que puede saber en todo momento quién ha realizado qué operación con el programa.

## 5.2.- Gestión de socios

No es imprescindible para poder buscar libros y gestioanrlos el controlar qué clientes son socios, o siquiera llevar una cuenta de los clientes.

La función principal de la aplicación es poder acceder y manipular datos de forma rápida, eficiente y con una presentación que sea fácil de leer.

## 5.3.- Gestión de usuarios

Podría debatirse que la gestión de usuarios es un requisito funcional, dado que resulta muy útil llevar un control de quién accede al programa y realiza operaciones.

Pero para consultar los libros disponibles y realizar operaciones en la base de datos no hace falta tener usuarios con diferentes roles.

Una alternativa clara es dar una clave común a todos los empleados, y con eso se evitaría la necesidad de una gestión (altas, bajas, modificaciones) aunque a costa de un nivel mínimo de protección de acceso a la aplicación.

Como características de la gestión de usuarios notamos:

* Se ha decidido que no pueden existir usuarios sin empleados, y que los empleados solo pueden tener un usuario.
* Hay varios roles, cada uno da diferentes permisos de acceso a funcionalidades dentro de la aplicación.

# 6.- Variables

Este capítulo organiza en tablas (ordenadas en orden alfabético) las variables que hay en el proyecto. Se ha decidido usar varias tablas con algunas variables en ellas para un resultado más ordenado y legible del documento.

Para saber dónde aparece una variable hay que consultar la columna *Formulario*.

**Tabla A**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nombre | Tipo | Dimensión | Formulario | Descripción |
| queryAdaptador | Texto(String) | Local(dim) | GestiónBusqueda.vb, GestionClienteSocio.vb, GestionEmpleados.vb, GestionClienteSocio, GestionLibro.vb, GestionLibro.vb, GestionUsuarios.vb, PantallaInicio, vb | Es la la consulta que queremos hacer a la base de datos. |
| datasetTableName | Texto(String) | Local(dim) | GestiónBusqueda.vb, GestionClienteSocio.vb, GestionEmpleados.vb, GestionClienteSocio, GestionLibro.vb, GestionLibro.vb, GestionUsuarios.vb, PantallaInicio.vb | El nombre que queremos darle a una tabla de las que hay en el *dataset.* |
| Gestion | Objeto (GestionConexion) | Local(dim) | GestiónBusqueda.vb,GestionClienteSocio.vb, GestionEmpleados.vb, GestionLibro.vb, GestionStock.vb, GestionLibro.vb | Objeto con el cual tenemos acceso a los métodos de la clase GestionBusqueda de la lbrería de conexión a la base de datos (*libConexion*) |
| getIdFila | Texto (string) | pública | GestionClienteSocio.vb, GestionEmpleados.vb, GestionLibros.vb, GestionStock.vb, GestionLibro.vb, GestionUsuarios.vb | Usado para obtener el id de la fila seleccionada.Esta información se usa durante procesos como eliminar una fila o modificarla. |
| validDatos | Objecto (Validacion) de la librería *libValidacionDatos* | Local | GestionEmpleados.vb, GestionLibro.vb, | Usado para acceder a los métodos de la librería de validación de datos. |

**Tabla B**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nombre | | Tipo | Dimensión | Formulario | Descripción |
| queryUsr | Texto(String) | | Local | GestionUsuarios.vb | Usado para obtener el elemento seleccionado del DataGridView. |
| queryEmp | Text(String) | | Local | GestionUsuarios.vb | Usado para obtener el elemento seleccionado del DataGridView. |
| fkEmpleado | Texto (String) | | Pública | GestionUsuarios.vb | Usado para obtener el elemento seleccionado del DataGridView. |
| getUsuario | Texto(String) | | Pública | GestionUsuarios.vb | Usado para obtener el elemento seleccionado del DataGridView. |
| Validador | Validacion(objeto de la librería libValidacionDatos | | Local | PantallaInicio.vb | Usada para obtener acceso a los métodos de la clase Validacion, para validar datos. |

**Tabla C**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nombre | | Tipo | Dimensión | Formulario | Descripción |
| validCon | GestionConextion(objeto de la libreria libConexion) | | Local | PantallaInicio.vb | Usado para realizar consultas a la base de datos. |
| Validador | Validacion(objeto de la librería libValidacionDatos | | Local | PantallaInicio.vb | Usada para obtener acceso a los métodos de la clase Validacion, para validar datos. |
| Conexion | OleDbConnecion | | Local | GestionConexion.vb(librería libConexion), Validacion.vb (librería libValidacionDatos) | Objeto para poder conectar el programa con la base de datos. |
| adaptador | OleDbDataAdapter | | Local | GestionConexion.vb(librería libConexion), Validacion.vb (libValidacionDatos) | Objeto usado para albergar la consulta que se desea realizar. |
| Midataset, midataset2, mitadaset 3 | Todos objetos Dataset | | local | GestionConexion.vb(librería libConexion), y midataset también se halla en Validacion.vb (libValidacionDatos) | Sirven para crear espacio en memoria para albergar las tablas obtenidas de la base de datos. |

**Tabla D**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nombre | | Tipo | Dimensión | Formulario | Descripción |
| datatable | DataTable | | Púbilica | GestionEmpleados.vb | Usado para buscar por ID en la tabla empleados. |
| conexionUrl | Texto(String) | | Local | GestionConexion(librería lib Conexion), Validacion(librería libValidacionDatos) | Usada para albergar los datos de la conexión. |

# 7.- Procedimientos y funciones

En este capítulo se muestran, en tablas, los procedimientos y funciones. Se han organizado en tablas ordenadas alfabéticamente (A-Z) para una fácil lectura. En la columna de Formulario se especifica dónde se usa cada procedimiento o función.

**Tabla A**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nombre | Tipo | Parámetros | Devuelve | Formulario | Descripción |
| cargarPantalla | Procedimiento(sub) | - | - | GestionBusqueda.vb, GestionClienteSocio.vb, GestionEmpleados.vb, GestionLibro.vb, GestionStock.vb, | Carga el DataGridView en el formulario donde se use. |
| llimpiarPantalla | Procedimiento(sub) | - | - | GestionBusqueda.vb, GestionClienteSocio.vb, GestionEmpleados.vb, GestionLibro.vb,GestionStock.vb, GestionUsuarios.vb | Limpia todos los campos en el formulario donde se use. |
| new | constructor | vacío | - | GestionConexion.vb, Validacion.vb (librerías libConexion y libValidacionDatos) | Usado cuando se instancia un obeto de la clase GestionConexion y Validacion respectivamente. |
| new | constructor | queryAdaptador(String), datasetTableName(String) (Consultar variables) |  | GestionConexion.vb | Usado cuando se instancia un obeto de la clase GestionConexion |
| getAdapter, getConxion, getDataset | Función | - | Objeto adaptador, objeto conexion y objeto dataset (consultar tabla e variables C) | GestionConexion.vb | Métodos usados para conseguir los objetos mencionados. |

**Tabla B**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nombre | Tipo | Parámetros | Devuelve | Formulario | Descripción |
| buscarEmpleado | Función, | Cadena(texto), table(texto), whColumnName(texto) | DataTable | GestionConexion.cv | Devuelve el contenido de la tabla Empleados. |
| buscarporCaracter | Función | Cadena(texto), table(texto), whColumnName(texto) | - | GestionConexion.vb | Utilizado para realizar búsquedas por carácter. Llena la variable dataset con el resultado filtrado por el carácter introducido. |
| eliminarFila | función | idFila(Entero), table(texto),whColumnName(texto) | boolean | GestionConexion.vb | Usado para borrar la fila seleccionada del datagridview. El valor devuelto indica si la operación tuvo éxito o no. |
| modificarEmpleado | Función | idFila(tenero), Dni(texto), nombre(texto), apellidos(texto), telefono(entero), tabla(texto)whColumnName(texto) | boolean | GestionConexion.vb | Usado para modificar una fila de la tabla empleados. El valor devuelto indica si la operación tuvo éxito o no. |
| insertEmpleado | Función | Id(entero),dni, (texto), nombre(texto), apellidos(texto), telefono(entero) | boolean | GestionConexion | Método usado para insertar una fila en la tabla Empleados. El valor devuelto indica si la operación tuvo éxito o no. |

**Tabla C**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nombre | Tipo | Parámetros | Devuelve | Formulario | Descripción |
| insertEmpleado | Función | Id(entero),dni, (texto), nombre(texto), apellidos(texto), telefono(entero) | boolean | GestionConexion | Método usado para insertar una fila en la tabla Empleados. El valor devuelto indica si la operación tuvo éxito o no. |
| insertClienteSocio | Función | Id(entero), nombre(texto), apellidos(texto), telefono(entero), email(texto), fechaAlta(teto), socio(boolean) | boolean | GestionConexion.vb | Método usado para insertar una fila en la tabla ClienteSocio. El valor devuelto indica si la operación tuvo éxito o no. |
| modificarCliente | Función | Id(entero), nombre(texto), apellidos(texto), telefono(entero), email(texto), fechaAlta(teto), socio(boolean), whColumnName(texto) | boolean | GestionConexion.vb | Usada para modificar datos en la tabla ClienteSocio. El valor devuelto indica si la operación tuvo éxito o no. |
| insertLibro | función | id(entero), ISBN(texto), Titulo(texto), precio(Single, número decimal), Autor(Texto), editorial(texto), tapa(texto), stock(texto), genero(texto),categoria(texto), estado(texto), idioma(texto), imagen(objeto envoltorio Byte) | boolean | GestionConexion.vb | Método usado para añadir datos a la tabla Libros. El valor devuelto indica si la operación tuvo éxito o no. |
| ModificarLibro | Función | id(entero), ISBN(texto), Titulo(texto), precio(Single, número decimal), Autor(Texto), editorial(texto), tapa(texto), stock(texto), genero(texto),categoria(texto), estado(texto), idioma(texto), imagen(objeto envoltorio Byte), table(texto), whColumnName(texto) | booleano | GestionConexion | Método usado para modificar datos en lal tabla Libros. El valor devuelto indica si la operación tuvo éxito o no. |

**Tabla D**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nombre | Tipo | Parámetros | Devuelve | Formulario | Descripción |
| ModificarLibro | Función | id(entero), ISBN(texto), Titulo(texto), precio(Single, número decimal), Autor(Texto), editorial(texto), tapa(texto), stock(texto), genero(texto),categoria(texto), estado(texto), idioma(texto), imagen(objeto envoltorio Byte), table(texto), whColumnName(texto) | booleano | GestionConexion | Método usado para modificar datos en lal tabla Libros. El valor devuelto indica si la operación tuvo éxito o no. |
| modificarStock | Función | idFIla(entero), stock(entero), table(texto), whColumnName(texto) | boolean | GestionConexion.vb | Usado para modificar stock. El valor devuelto indica si la operación tuvo éxito o no. |
| insertarUsuario | Función | Id(entero), usuario(texto), clave(entero), rol(texto), fkEmpelado(entero) | boolean | GestionConexion.vb | Usada para insertar una fila en la tabla Usuarios. El valor devuelto indica si la operación tuvo éxito o no. |
| modificarUsuario | Función | getFila(entero), usuario(texto), clave (entero), rol(texto), fkEmpelado(entero), table (texto), whColumnName(texto) | Boolean | GestionConexion.vb | Usada para modificar una fila en la tabla Usuarios. El valor devuelto indica si la operación tuvo éxito o no. |
| getEmpleados | Función | queryAdpter(texto(, datasetTableName(texto) | DataSet | GestionConexion.vb | Usada para obtener ID de empleado y poder llenar el campo ID en el Formulario GestionUsuarios. |

**Tabla E**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nombre | Tipo | Parámetros | Devuelve | Formulario | Descripción |
| gerRelacionEmpUsr | Función | queryAdaptadoRelacion(texto), datasetTableRelacionName(texto),queryAdaptadorEmp(texto), queryAdaptadorUsr (texto), datasetTableNameEmp(texto), datastTableNemUsr(texto) | Dataset | GestionConexion.vb | Método usado para obtener la relación entre empleados y usuarios. Devuelve un dataset con varias tablas que se usan para mostrar en la pantalla. |
| buscarStock | Función | Cadena(Texto), table(texto), whColumnName(texto), datasetTableName(texto) | - | GestionConexion.vb | Método usado para buscar por caracteres |
| selectTabla | Función | Campo(texto), tabla(texto) | DataTable | GestionConexion.vb | Devuelve la consulta sobre una tabla que se pasa por parámetro. |
| selectWhere | Función | Campo(texto), table(texto), cond1(texto), cond2(texto) | DataTable | GestionConexion.vb | Devuelve la consulta filtrada con una condición. |
| selectDosWhere | Función | Campo(texto), tabla(texto), cond1(texto), cond2(texto), cond3(texto), cond4(texto) | Boolean | GestionConexion | Devuelve un booleano indicando si se ha encontrado una fila con los parámetros recibidos. |

**Tabla F**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nombre | Tipo | Parámetros | Devuelve | Formulario | Descripción |
| getRol | Función | cond1(texto), cond2(texto), cond3(texto), cond4(texto) | Entero | GestionConexion.vb | Devuelve un entero, que indica qué rol tiene la persona que ha efectuado el acceso a la aplicación. |
| errorLogWrite | Fución | - | Boolean | GestionConexion.vb(también en Validacion.vb en libValidacionDatos) | Usado para escribir un fichero de errores con información sobre las excepciones. El archivo es creado en el directorio Debug. |
| mensajeErrorDatos | Función | Método sobrecargado, puede recibir ningún parámetro, o texto, o texto y un objeto Exception. | String | GestionConexion.vb y también Validacion.vb (en la biblioteca libValidacionDatos) | Usado para mostrar por pantalla un mensaje de error. |
| validarUsuario | Función | stringRecibido(texto, tipo(entero) | Boolean | Validacion.vb | Usado para validar que en el *textbox* donde el usuario introduce datos de usuario, introduce los datos permitidos. |
| validarPrecioString | Funcion | Precio(texto) | Boolean | Validacion.vb | Usado para validar la introducción de datos en el campo precio. |

**Tabla G**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nombre | Tipo | Parámetros | Devuelve | Formulario | Descripción |
| validarCodigo | Funcion | codigo(texto), tipo(entero) | Boolean | Validacion.vb | Usado para validar la introducción de datos en el campo clave y otros. El entero indica al método el tipo de dato que se va a validar. |
| mensajeInfoDatoInvlalido | Función | Campo(texto) | Texto(String) | Validacion.vb | Usado para mostrar un mensaje al usuario de que ha introducido un dato inválido en un camo de texto. |
| validarEmail | Función | emailAddress(texto) | Boolean | Validacion.vb | Usado para validar la introducción de datos en el campo email en el Formulario ClienteSocio.vb |
| validarDNI | Función | Dni(texto) | Boolean | Validacion.vb | Usado para validar el DNI en el Formulario GestionEmpleados.vb |
| validarISBN | Función | Isbn(texto) | boolean | Validacion.vb | Usado para validar el campo ISBN en el formulario GestionLibros.vb |

# 8.- Definición de ficheros

Los ficheros utilizados en este proyecto son los siguientes:

|  |  |  |
| --- | --- | --- |
| Nombre | Extensión | Descripción |
| cas\_lib\_db | .accdb: | Se encuentra en el directorio Debug. La base de datos con extensión .accdb. Es una base de dato Access, y es donde se guardan todos los datos de la aplicación. |
| ErrorLog | .txt (texto plano) | Se encuentra en el directorio Debug. Es un archivo de texto plano donde se detalla la información y el número de identificación de los errores que puedan suceder durante algún proceso del programa. |
| logAcceso | .txt(texto plano) | Se encuentra en el directorio Debug. Es un archivo de texto plano donde se detallan los accesos de los usuarios. Se pone la fecha y el nombre del usuario. |
| Debug | Directorio | Es el directorio donde se encuentra todo lo necesario para ejecutar el programa. |

# 9.- Bibliotecas/Librerías

En este capítulo se tratan las librerías usadas en este proyecto.

Las bibliotecas usadas son dos.

## 9.1.- Librería para todo lo relacionado con la base de datos: *libConexion*

En este sub-apartado tratamos la librería con la que se lleva a cabo la gestión de conexión a la base de datos y gestión de operaciones con la base de datos.

**Cuando en un formulario se desea integrar la función de alta, baja, modificación o búsqueda de algún elemento es cuando debemos usar esta biblioteca.**

Incluye además dos métodos (que son iguales que los que están en la librería de validación de datos - *libValidacionDatos*) que sirven para escribir un *log* de errores en el directorio *Debug* cuando surge una excepción, y para gestionar el mostrar un mensaje de error al usuario cuando surja una excepción.

*Para ver los métodos que pertenecen a esta librería por favor vaya al capítulo X.*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Para usar los métodos de esta librería se debe hacer lo siguiente:**

* Es necesario incluir en el formulario un objeto de la clase *GestionConexion*, pasándole dos valores de texto (tipo String), tal que así:

Dim gestion As New LibConexion.GestionConexion(queryAdaptador, datasetTableName)

En el ejemplo *gestion* es el nombre del objeto de clase GestionConexion. Por parámetro se le pasan dos variables que tienen un texto (String) como valor. El primer parámetro es el *adaptador* (la consulta que deseamos hacer) y el segundo parámetro es el nombre que vamos a darle a la tabla dentro del *DataSet* (como hemos mencionado, en el *dataset*, el espacio de memoria caché reservada para los datos de la base de datos, se pueden guardar varias tablas).

* + En el constructor de la clase primero se instancia la *conexión* (pasándole una variable que contiene un valor texto (String). No se ha usado aún ninguno de los parámetros que pasamos cuando llamamos al constructor.
  + Luego se instancia el *adaptador*, pasándole uno de los parámetros que recibió el constructor (el *adaptador)* y la conexión creada previamente.
  + Y finalmente se instancia un nuevo dataset, para poder llenarlo llamando al método *fill* del *adaptador*, pasándole el atributo de la clase *midataset* que acabamos de instanciar(ver apartado variables) y el nombre de la tabla, también recibido por parámetro cuando invocamos el constructor desde el formulario (el nombre de la tabla identifica la tabla, el miembro del *dataset* que luego se selecciona cuando se cargan los datos desde el *dataset* al *DataGridView)*.

Public Sub New(queryAdaptador As String, datasetTableName As String)

conexion = New OleDbConnection(conexionUrl)

'adaptador de la tabla Emplado

adaptador = New OleDbDataAdapter(queryAdaptador, conexion)

midataset = New DataSet

adaptador.Fill(midataset, datasetTableName)

conexion.Close()

End Sub

* Una vez que tenemos el objeto declarado e instanciado, podemos invocar los métodos necesarios para realizar consultas, altas, bajas y modificaciones.
* Se adjunta un ejemplo tomado del Formulario *GestionLibro*:

' Se carga un control textbox en el formulario con el dato de la columna id de la tabla libros.

txtLibroID.DataBindings.Add("text", gestion.getDataSet, "Libros.id")

* Otro ejemplo, esta vez de modificación de un libro:

If gestion.ModificarLibro(getIdFila, txtISBN.Text, txtTitulo.Text, CSng(txtPrecio.Text), txtAutor.Text, txtEditorial.Text, txtTapa.Text, txtGenero.Text, cbxCategorias.Text, cbxEstado.Text, cbxIdioma.Text, arrImage, " Libros ", " id ") Then

Vemos cómo el método *ModificarLibro* recibe varios valores por parámetro. El método en este caso devuelve un valor booleano, y según el valor se mostrará al usuario un mensaje describiendo éxito o fracaso de la operación.

## 9.2.- Librería para gestión de validación de datos: *libValidacionDatos*

En esta librería se incluyen todos los métodos para la gestión de la validación de los datos en todo el proyecto.

**Si se desean validar los datos introducidos por el usuario en los campos dedicados a tal fin en los formularios, entonces se debe usar esta librería.**

Incluye además dos métodos (que son iguales que los que están en la librería de gestión de conexión a la base de datos -*libConexion*) que sirven para escribir un *log* de errores en el directorio *Debug* cuando surge una excepción, y para gestionar el mostrar un mensaje de error al usuario cuando surja una excepción.

Además incluye un método para escribir un archivo (*logAcceso.txt*) donde se registran los accesos de los usuarios.

*Para ver los métodos que pertenecen a esta librería por favor vaya al capítulo X.*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

La librería incluye, por necesidad, otras librerías a su vez para su funcionamiento. Estas son:

Imports System.IO

Imports System.Text.RegularExpressions

Se precisa de la librería IO para poder escribir el archivo de errores y el de accesos.

Y se precisa la librería de *RegularExpressions* para poder usar métodos útiles a la hora de usar expresiones regulares en los métodos que lo necesiten (por ejemplo el de validación de DNI).

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Los pasos para poder usar los métodos incluidos en esta librería son los siguientes:

* No es necesario importar la librería *System.IO* en el formulario donde se deseen usar los métodos de esta clase, pero es recomendable.
* Es necesario crear un objeto de la clase *Validacion* en el formulario. El constructor no requiere que se le pasen parámetros.
* Con el objeto ya tenemos acceso a los métodos.
  + Se adjunta un ejemplo

If (validDatos.validarUsuario(txtAutor.Text, 1)) Then...

Vemos el objeto validDatos de la clase Validación que nos da acceso a los métodos para validar los datos que introduce el usuario.

Vemos que se le pasa por parámetro un valor texto contenido en el *control* *TextBox* de nombre *txtAutor* (es el campo donde el usuario rellena el nombre del autor del libro, en el Formulario *GestionLibro*).

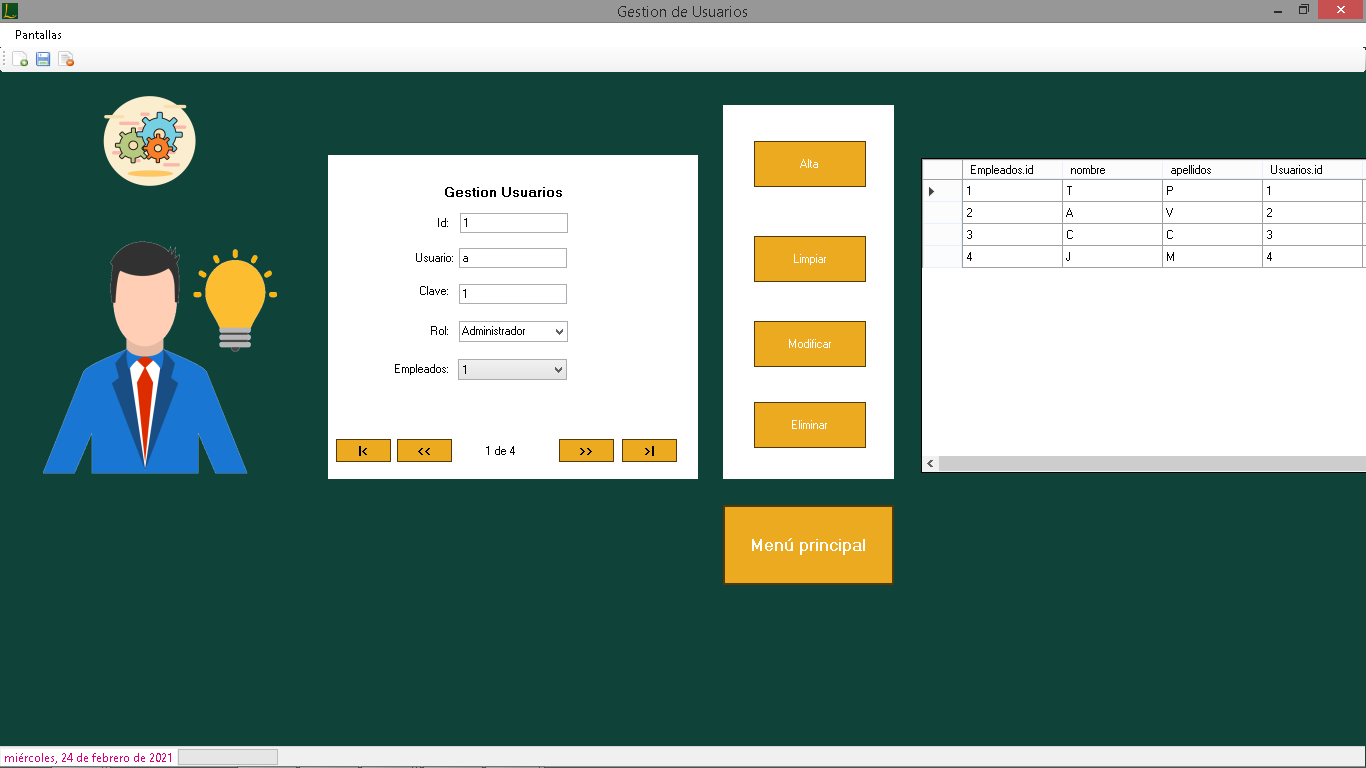
También se le pasa un entero, que sirve para distinguir qué tipo de dato se va a validar.

Para más información sobre la función, consulte el apartado de procedimientos y funciones de este documento.

# 10.- Diseño de entrada y salida

Ejemplos de entrada de datos:

**Ejemplo 1:**

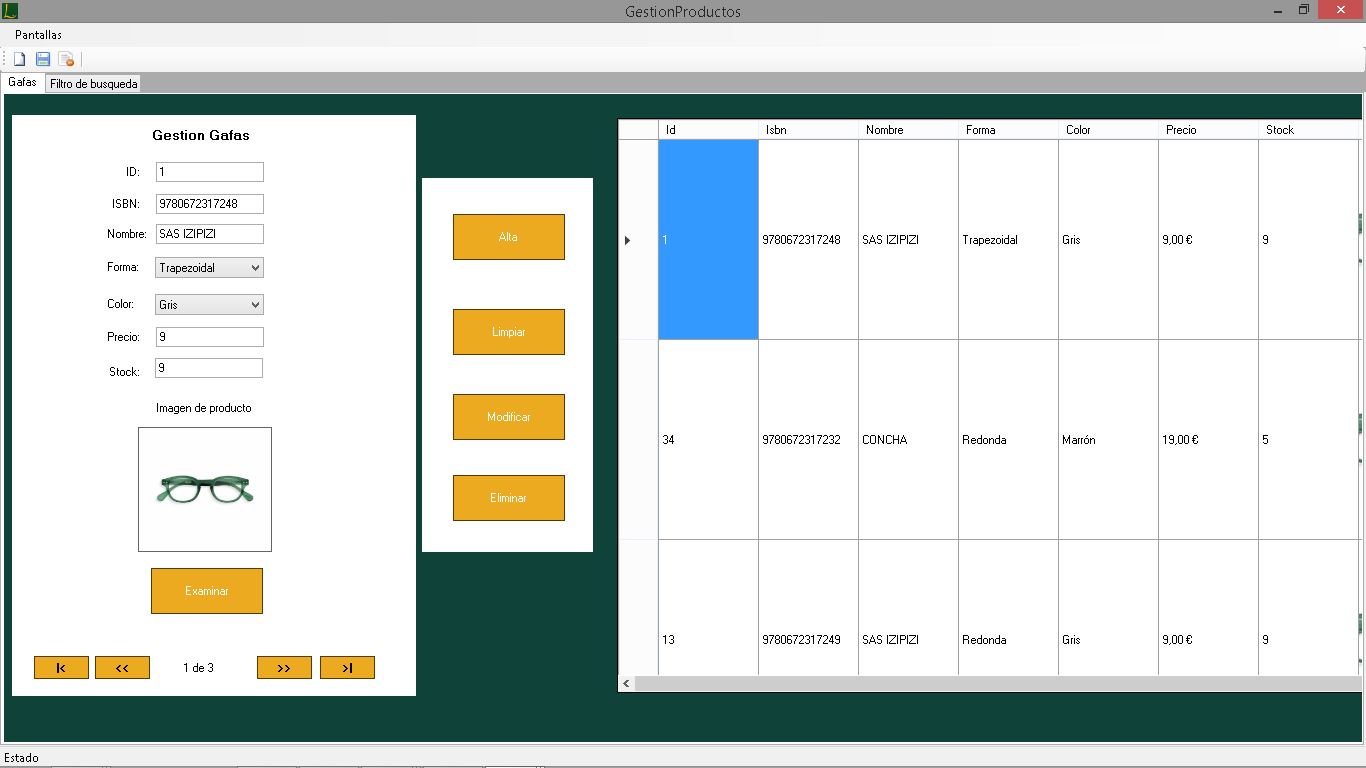


El circulo indica un campo de texto, uno de los principales métodos de introducción de datos

En todos los formularios donde se introducen datos el sistema es el mismo.

* Para modificar un dato es necesario hacer doble click en la fila del *DataGridView*.

**Ejemplo 2:**

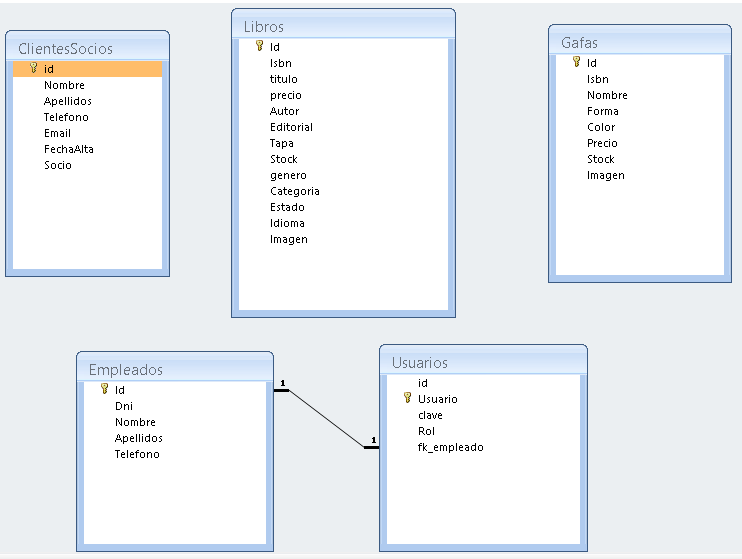


*Formulario de gestión de productos*

La segunda forma de introducción de datos es mediante la selección de un archivo .bmp o .jpg para establecer la imagen de un libro o producto.

# 11.- Diagramas

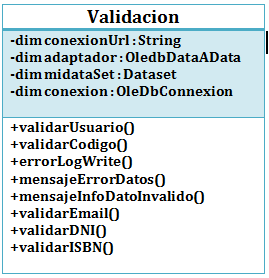
## 11.1.- Diagrama Entidad relación de la base de datos



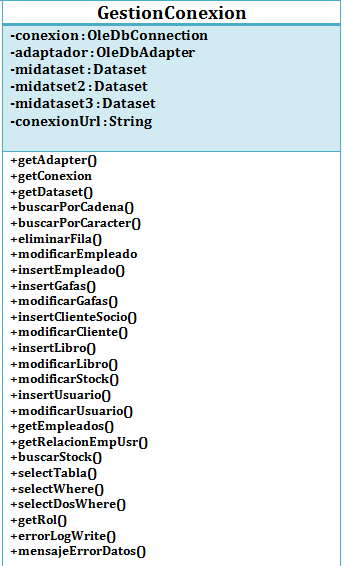
Modelo entidad relación de la base de datos

## 11.2.- Diagrama de clases

### Librería de Validación de datos



### Librería de conexión a la base de datos



# 12.- Casos de uso

En este proyecto se han realizado pruebas unitarias, de integración y de sistema.

Sin embargo, en este documento se reflejan solamente las de sistema, realizadas mediante pruebas de caja negra y blanca.

Para llevar a cabo los casos de uso se han introducido combinaciones de valores diferentes en cada control donde se pudiese recibir información de cualquier modo(por ejemplo en la gestión de empleados, rellenando los campos para dar de alta un libro, o en la consulta de libros), y se han probado combinaciones de rutas con cada control que tuviese código asociado(por ejemplo cambiar de pantalla desde el menú, desde el generador de informes, desde la gestión de usuario, etcétera).

A continuación se demuestran los casos de prueba.

Tenga en consideración que en muchos casos en la columna "Input" se separan con comas un grupo de casos individuales.

## Tests generales a los formularios

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Descripción | Input | Esperado | Resultado |
| 1 | Navegación entre formularios usando barra de herramientas | Click en botones de menustrip | Funcionamiento normal. | El programa no validaba el rol con el que se había accedido al cambiar de pantallas. Se ha añadido un método para que al iniciarse cada formulario, se llama a éste y se comprueba el rol del usuario que ha hecho acceso. |
| 2 | Gestión (Altas, bajas, modificaciones) ( NO inserción de datos, ese es otro caso de prueba) | Presionar botones (alta, eliminar, modificar) | Funcionamiento correcto | Funcionamiento correcto |
| 5 | Cerrar la aplicación | Cerrar cualquier formulario abierto que no sea el de inicio. | Hay que cerrar todos los formularios abiertos antes de cerrar el principal. | Funcionamiento correcto. |
| 6 | Aparición del tooltp (mensaje de ayuda) | Dejar el ratón sobre los componentes durante un tiempo determinado | Aparición del tooltip. | Funcionamiento normal: aparición del tooltip de ayuda. |
| 7 | Realizar búsqueda y en la mitad de esta realizar una operación de gestión | Introducción de carácter en campo de texto de búsqueda y luego dar de alta, modificar o eliminar una fila antes de acabar con la búsqueda. | Funcionamiento normal. | Funcionamiento normal. |
| 8 | Limpiar campos | Presionar botón "Limpiar" (Llamar al método *llimpiar*) campos | Funcionamiento normal | Al seleccionar el DataGridView se borraban las filas (aunque reiniciando el programa se arreglaba el problema - solo era un problema visual). Pero fue el error que nos llevó a realizar el doble click para poder seleccionar una fila. |

## Validación de campos en formularios e inicio de sesión

En todos los formularios donde hay campos para introducidr datos y en la Pantalla inicio se han realizado las siguientes pruebas de caja negra:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Descripción | Input | Esperado | Resultado |
| 1 | En textbox donde se permiten caracteres | "a" | Funcionamiento normal. | Funcionamiento normal. |
| 2 | En textbox donde se permiten caracteres | "a", "@", "ç", "a2", "a@", "aç" y sobrepasar el máximo de 10 dígitos. | Mostrado mensaje de error. | Se muestra mensaje de error. |
| 5 | En textbox donde se permiten dígitos | "a", "@", "ç", ,"a@","2a", "2@" y sobrepasar el máximo de 6 dígitos. | Mostrado mensaje de error. | Se muestra mensaje de error. |
| 6 | En textbox donde se permiten dígitos | 2 | Funcionamiento normal | Funcionamiento normal. |
| 7 | En inicio de sesión: Insertar combinación de usuario y clave que no existe en la base de datos. | "zzzz", "871776" | Mensaje de usuario inexistente. | Mensaje de usuario inexistente. |
| 8 | En inicio de sesión: Insertar combinación de usuario y clave que sí existen en la base de datos. | "admin", "123456" |  |  |

## Pantalla principal

En el formulario de pantalla principal se han llevado a cabo los siguientes tests:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Descripción | Input | Esperado | Resultado |
| 1 | Cerrar sesión | Click botón | Funcionamiento normal. | Funcionamiento normal. |
| 2 | Navegación a formularios | Click en botones | Funcionamiento normal. | Funcionamiento normal. |
| 3 | Intento de acceso a partes del formulario con diferentes roles/permisos | Click en botones diferentes | Funcionamiento normal, acceso a roles con permisos y denegacion a roles sin permisos. | Funcionamiento normal, acceso a roles con permisos y denegacion a roles sin permisos. |

# 13.- How to

Los métodos explicados en este apartado pueden consultarse en el capítulo de métodos.

## Validaciones

Se han usado dos formas de validar datos introducidos en campos texto:

1. Usando expresiones regulares y métodos de la librería disponible para estas operaciones,
2. y usando estructuras de control donde se itera por cada pieza de información de los datos introducidos y se comparan con los datos permitidos.

* El primer caso es más sencillo dado que se usa un texto (String) donde se ponen los caracters y el patrón permitidos (por ejemplo en el ISBN).
* El segundo caso requiere de explicacion algo más detallada: los métodos que validan de este modo reciben dos parámetros, los datos, y un entero que indica qué dato se va a validar. Según el dato que se va a validar, se itera por un texto (String) de caracteres permitidos y se comparan ambas cadenas.

Si todos los caracteres coinciden, significa que el número de aciertos/coincidencias de caracteres (entre caracteres permitidos y datos introducidos) es igual a la longitud de los datos introducidos. Por tanto, se validan como buenos.

## Pantalla de inicio

Para el inicio de sesión se ha decidido lo siguiente:

En la pantalla inicial el usuario introduce datos y pincha el botón acceder. Entonces se realiza una consulta con un filtro, con los datos introducidos. Si la consulta retorna una fila, significa que el usuario existe y tiene permiso para entrar.Si no, significa que se hayan introducido mal los datos, o el usuario no exista.

Si los datos existen y la combinación de nombre de usuario y clave son válidos, se cambia el contenido de texto de un control label en la pantalla principal, para cambiarlo a Administrador, Usuario o Guest según corresponda.

Esto es importante porque el método que se halla en la pantalla principal lee este label, y según el valor se darán permisos a uno u otros formularios (bloqueando o liberando botones en el menustrip).

## Pantalla principal

Se ha decidido que para navegar hacia los diferentes formularios desde el menú principal se pondrá a la vista del usuario varios botones (algunos de ellos son controles PictureBox pero al ser pinchados se lanza el evento de Visual Basic .onClick, que los hace reaccionar).

Cada botón lleva una estructura de control que comprueba el contenido del control label que se encuentra en esta pantalla (PantallaPrincipal.vb). Según el contenido del label, se dará permiso o no al usuario para ir a otro formulario.

## Gestiones

Las gestiones de alta, eliminación y modificación se llevan a cabo cuando el usuario pincha los botones alta, eliminar o modificar que se ven en la pantalla.

Si los campos necesarios están llenos y están validados, se ejecutará el código correspondiente. En todos los casos, en el código, se usa un objeto de la librería *libConexion* para realizar todo el trabajo. La conexión a la base de datos, y la consulta.

Luego se actualiza el DataGridView, cargándose en el DataGridView del formulario los datos recibidos tras invocar el método que nos interese de la librería de gestión de conexión a la base de datos.

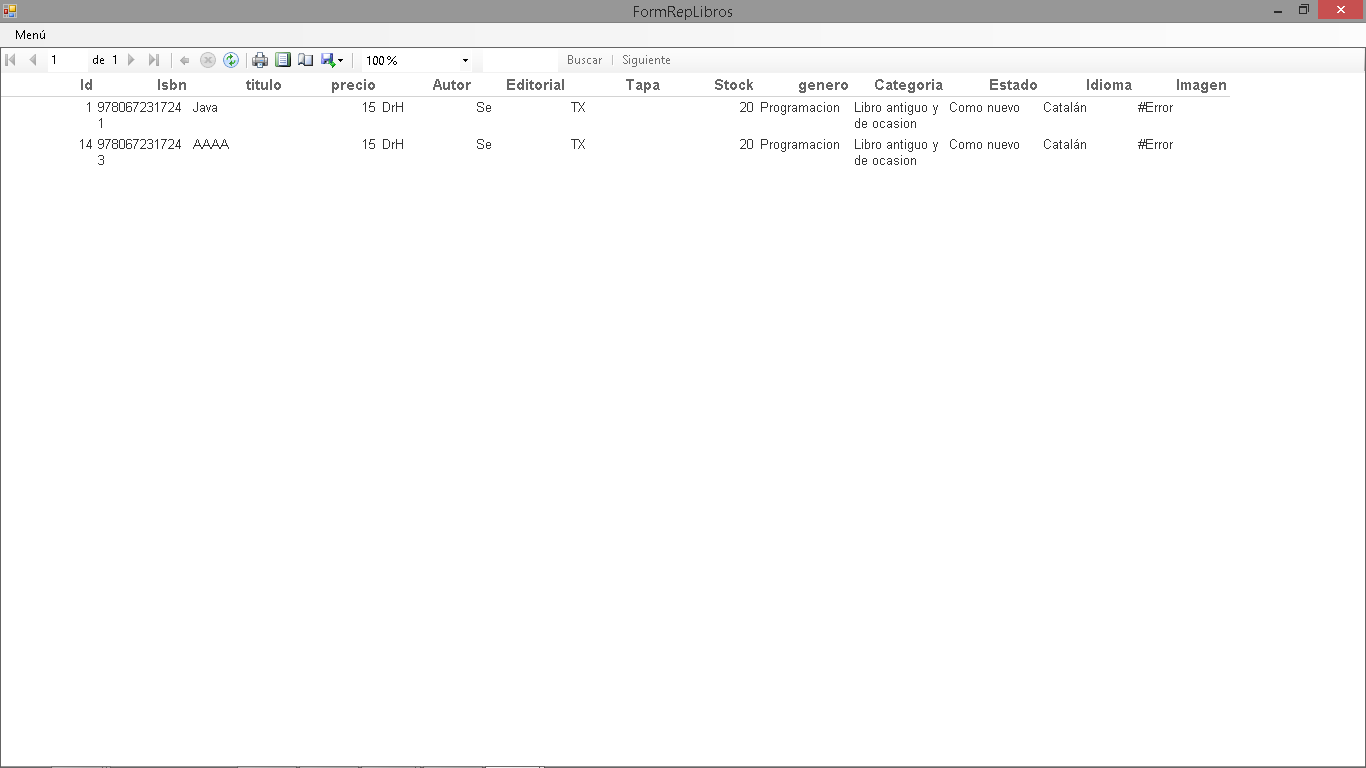
Como resultado de un comportamiento no deseado por parte del DataGridView al llevar a cabo la acción de limpiar todos los campos de un formulario, se ha decidido que para poder seleccionar una fila para realizar cualquier operación de gestión, se debe hacer doble click en lugar de un solo click (que es lo que normalmente basta para que los datos de una fila se carguen en los campos que el programador desee).

# Terminología

* Function Función, método: Subprograma que realiza una tarea específica y devuelve un valor.
* Procedimiento, Sub: Subprograma que realiza una tarea específica y no devuelve un valor.
* Alta: Añadir una fila, un conjunto de datos a la base de datos.
* Modificación: Modificar una fila, un conjunto de datos en la base de datos.
* Eliminar/baja: eliminar una fila, un conjunto de datos en la base de datos.
* Fila: conjunto de columnas.
* Columnas: Compuestas por campos. Estos a su vez son la "unidad mínima" que contiene valores de todo tipo.
* Inicio de sesion/Acceso: Cuando el usuario mete una combinación de nombre de usuario y contraseña puede usar las funcionalidades de la aplicacion.

# Anexo

## Modelo de informe de Libro, clientes y empleados



\*Solo varían los campos mostrados, el modelo para cada informe es el mismo.

# Código fuente

**Librería libValidacionDatos**

Imports System.IO

Imports System.Data.OleDb

Imports System.Text.RegularExpressions

Public Class Validacion

Dim conexionUrl As String = "Provider=Microsoft.ACE.OLEDB.12.0;DATA SOURCE=./cas\_lib\_db.accdb"

Dim adaptador As OleDbDataAdapter

Dim midataset As DataSet

Dim conexion As OleDbConnection

Public Sub New()

conexion = New OleDbConnection(conexionUrl)

adaptador = New OleDbDataAdapter("SELECT \* FROM Usuarios", conexion)

midataset = New DataSet

adaptador.Fill(midataset, "Tabla")

End Sub

' Constructor parametrizado.

Public Sub New(consulta As String, tabla As String)

conexion = New OleDbConnection(conexionUrl)

adaptador = New OleDbDataAdapter("SELECT " & consulta & " FROM " & tabla, conexion)

midataset = New DataSet

adaptador.Fill(midataset, tabla)

End Sub

' Función para ver si los datos (String) introducidos por el usuario son válidos para lo que se exige en

' el campo.

Public Function validarUsuario(stringRecibido As String, tipo As Integer) As Boolean

' Si es 0, se está recibiendo que se va a validar un usuario.

' Si es 1, es nombre.

' Si es 2 es apellido.

' Si es 3 es dirección. -> Ejemmplo

' Ejemplos abajo

' 4

' 5

' 6 etc...

' Entorno:

' Creamos una lista de caracteres permitidos.

Dim charsPosibles As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZ"

' Se permiten espacios y otros caracteres en los nombres, apellidos y direcciones.

' Para el nombre y apellidos es el mismo rango de caracteres posible.

Dim charsPosiblesNombre As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZáéíóúÁÉÍÓÚç- "

Dim charsposiblesDireccion As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZ ,'ªº\-0123456789"

' Las variables donde guardaremos los caracteres para compararlos.

Dim chDeLista As Char

Dim chRecibido As Char

' El buchón/chivato para ver si el número de aciertos (caracteres equivalentes) es el mismo al número

' de caracteres del string pasado por parámetro.

Dim coincidencias As Integer = 0

' Finalmente la variable boolean valido para que en el formulario se pueda saber si

' el string es válido o no.

Dim valido As Boolean = False

Dim valorMaximo As Integer

If tipo = 0 Then

valorMaximo = 10

ElseIf tipo = 1 Then

valorMaximo = 25

ElseIf tipo = 2 Then

valorMaximo = 30

ElseIf tipo = 3 Then

valorMaximo = 50

End If

' Inicio

' Primero comprobamos el tamaño del string

If stringRecibido.Length > valorMaximo Then

MsgBox("No se permiten más de" & valorMaximo & " caracteres.", MsgBoxStyle.OkOnly + MsgBoxStyle.Exclamation, "Longitud incorrecta")

Else

' Iteramos un número de veces igual al tamaño del string pasado por parámetro.

For i = 1 To stringRecibido.Length

' Guardamos el carácter en chRecibido.

chRecibido = GetChar(stringRecibido, i)

' Comprobamos si el carácter que acabamos de guardar es igual a los caracteres del string(charsPosibles).

' Iteramos un número de veces igual al tamaño de charsPosibles (dos veces el abecedario, una en minúsculas y otra mayúsculas)

' 0 Es usuario

If tipo = 0 Then

For j = 1 To charsPosibles.Length

If chRecibido = GetChar(charsPosibles, j) Then

' Comparamos los caracteres. Si son iguales, se suma 1 al buchón

coincidencias = coincidencias + 1

Else

' MsgBox("No ha introducido un carácter válido.", MsgBoxStyle.Exclamation, "Error al introducir datos para el acceso.")

End If

Next j

ElseIf tipo = 1 Then

For j = 1 To charsPosiblesNombre.Length

If chRecibido = GetChar(charsPosiblesNombre, j) Then

' Comparamos los caracteres. Si son iguales, se suma 1 al buchón

coincidencias = coincidencias + 1

Else

' MsgBox("No ha introducido un carácter válido.", MsgBoxStyle.Exclamation, "Error al introducir datos para el acceso.")

End If

Next j

ElseIf tipo = 2 Then

' Con el apellido usamos el mismo rango de caracteres válidos que con el nombre.

For j = 1 To charsPosiblesNombre.Length

If chRecibido = GetChar(charsPosiblesNombre, j) Then

' Comparamos los caracteres. Si son iguales, se suma 1 al buchón

coincidencias = coincidencias + 1

Else

' MsgBox("No ha introducido un carácter válido.", MsgBoxStyle.Exclamation, "Error al introducir datos para el acceso.")

End If

Next j

ElseIf tipo = 3 Then

For j = 1 To charsposiblesDireccion.Length

If chRecibido = GetChar(charsposiblesDireccion, j) Then

' Comparamos los caracteres. Si son iguales, se suma 1 al buchón

coincidencias = coincidencias + 1

Else

' MsgBox("No ha introducido un carácter válido.", MsgBoxStyle.Exclamation, "Error al introducir datos para el acceso.")

End If

Next j

End If

Next i

' Si el buchón tiene un valor igual al tamaño del string pasado, significa que todos los caracteres

' son válidos, pues fueron comparados con los caracteres del String y fueron dados por buenos.

If coincidencias = stringRecibido.Length Then

valido = True

End If

End If

Return valido

End Function

' Método usado en la pantalla de inicio de sesión para ver que se introducen solo números en el campo de contraseña.

Public Function validarCodigo(codigo As String, tipo As Integer) As Boolean

' Si se recibe 0 se está validando contraseña en inicio de sesión. (se permiten 6 dígitos)

' Si se recibe 1 se está validando teléfono. (se permite 9 dígitos).

' Si se recibe 2 se está validando precios en productos

' vvv validaciones en proyecto Casa del libro:

' Si se recibe 3 se está validando un ID.

' Si recibe 4 es Stock

' Se opta de nuevo por la solución de poner en un String los valores permitidos.

Dim charsPosibles As String = "0123456789"

' Para el campo de teléfono se permite usar el carácter especial +.

' Se ha quitado el carácter especial + por ahora.

Dim charsPosiblesTelefono As String = "0123456789"

Dim charsPosiblesPrecio As String = "0123456789."

Dim valido As Boolean = False

Dim coincidencias As Integer = 0

Dim valorMaximo As Integer

' Clave/contraseña de inicio sesión

If tipo = 0 Then

valorMaximo = 6

' Número de teléfono

ElseIf tipo = 1 Then

valorMaximo = 9

' Precios de productos

ElseIf tipo = 2 Then

valorMaximo = 5

' ID de libro

ElseIf tipo = 3 Then

valorMaximo = 6

ElseIf tipo = 4 Then

valorMaximo = 6

End If

' Solo se permiten códigos de hasta 4 números.

' La validación de que sean 4 se hace en otro método, porque eso depende de la creación de usuarios.

If codigo.Length > valorMaximo Then

MsgBox("Por favor, introduzca una contraseña/código de 4 dígitos", MsgBoxStyle.OkOnly + MsgBoxStyle.Exclamation, "Longitud incorrecta.")

Else

For i = 1 To codigo.Length

If tipo = 0 Then

For j = 1 To charsPosibles.Length

' Chars posibles solo 0 a 9

If (GetChar(codigo, i) <> GetChar(charsPosibles, j)) Then

Else

' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay

' entonces los datos se validan.

coincidencias = coincidencias + 1

End If

Next j

ElseIf tipo = 1 Then

' Chars posibles para telefono

For j = 1 To charsPosiblesTelefono.Length

If (GetChar(codigo, i) <> GetChar(charsPosiblesTelefono, j)) Then

Else

' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay

' entonces los datos se validan.

coincidencias = coincidencias + 1

End If

Next j

ElseIf tipo = 2 Then

For j = 1 To charsPosiblesPrecio.Length

' Chars posibles para precio.

If (GetChar(codigo, i) <> GetChar(charsPosiblesPrecio, j)) Then

Else

' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay

' entonces los datos se validan.

coincidencias = coincidencias + 1

End If

Next j

ElseIf tipo = 3 Then

For j = 1 To charsPosibles.Length

' Solo usamos de 0 al 9

If (GetChar(codigo, i) <> GetChar(charsPosibles, j)) Then

Else

' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay

' entonces los datos se validan.

coincidencias = coincidencias + 1

End If

Next j

ElseIf tipo = 4 Then

For j = 1 To charsPosibles.Length

' Solo usamos de 0 al 9

If (GetChar(codigo, i) <> GetChar(charsPosibles, j)) Then

Else

' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay

' entonces los datos se validan.

coincidencias = coincidencias + 1

End If

Next j

End If

Next i

End If

' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.

If coincidencias = codigo.Length Then

valido = True

End If

Return valido

End Function

Public Function errorLogWrite() As Boolean

Dim correcto As Boolean = False

' Escribir en errorLog

Dim fichero = "Errorlog.txt"

Try

Dim fich As New FileStream(fichero, FileMode.Append, FileAccess.Write)

Dim rs As New StreamWriter(fich)

rs.WriteLine(Now & "---> " & "Error en la comprobación de datos." & Chr(13) & "Detalle: " & Err.Description & Chr(13) & "Número de error: " & Err.Number)

rs.Close()

fich.Close()

correcto = True

Catch ex As Exception

MsgBox(ex.Message)

End Try

Return correcto

End Function

Public Function mensajeErrorDatos() As String

Dim s As String

s = "Error en la comprobación de datos." & Chr(13) & "Detalle: " & Err.Description & Chr(13) & "Número de error: " & Err.Number

' MsgBox("Error en la comprobación de datos." & Chr(13) & "Detalle: " & Err.Description & Chr(13) & "Número de error: " & Err.Number)

errorLogWrite()

Return s

End Function

' Mensaje de información al usuario de que el dato intrducido no es válido

' se retorna un string, hace falta usar msgbox o equivalente en el programa principal para que se vea.

Public Function mensajeInfoDatoInvalido(ByVal campo As String) As String

' Chr(13) sirve para salto de linea, es como <br> en html

Dim s As String

s = "El dato que ha introducido en el campo " & campo & " no es válido."

' MsgBox("Error en la comprobación de datos." & Chr(13) & "Detalle: " & Err.Description & Chr(13) & "Número de error: " & Err.Number)

Return s

End Function

' Validar/Validación email

Function validarEmail(ByVal emailAddress As String) As Boolean

Dim regExPattern As String = "^[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,4}$"

Dim emailAddressMatch As Match = Regex.Match(emailAddress, regExPattern)

If emailAddressMatch.Success Then

Return True

Else

' mensajeInfoDatoInvalido("Correo eletrónico")

Return False

End If

End Function

' Validar/Validación DNI

Function validarDNI(ByVal dni As String) As Boolean

Dim regExPattern As String = "^[0-9]+[a-zA-Z]{1,2}$"

Dim dnimatch As Match = Regex.Match(dni, regExPattern)

If dnimatch.Success Then

Return True

Else

' mensajeInfoDatoInvalido("DNI")

Return False

End If

End Function

' Validar/Validación ISBN

' Matches y non matches para el primer regex pattern.

' matches 0672317249 | 9780672317248

' Non-Matches 0-672-31724-9 | 5555555555555

Function validarISBN(ByVal isbn As String) As Boolean

'Dim regExPattern As String = "/(978[\--– ])?[0-9][0-9\--– ]{10}[\--– ][0-9xX])|((978)?[0-9]{9}[0-9Xx])/"

'Dim regExPattern As String = "^(?:ISBN(?:-13)?:?●)?(?=[0-9]{13}$|(?=(?:[0-9]+[-●]){4})[-●0-9]{17}$) 97[89][-●]?[0-9]{1,5}[-●]?[0-9]+[-●]?[0-9]+[-●]?[0-9]$"

Dim regExPattern As String = "^(97(8|9))?\d{9}(\d|X)$"

Dim isbnMatch As Match = Regex.Match(isbn, regExPattern)

If isbnMatch.Success Then

Return True

Else

' mensajeInfoDatoInvalido("DNI")

Return False

End If

End Function

End Class

**Librería libConexion**

Imports System.Data.OleDb

Imports System.IO

Public Class GestionConexion

'Atributos de Conexion

Dim conexion As OleDbConnection

Dim adaptador As OleDbDataAdapter

Dim midataset, midataset2, midataset3 As DataSet

Dim conexionUrl As String = "Provider=Microsoft.ACE.OLEDB.12.0;DATA SOURCE=./cas\_lib\_db.accdb"

' Constructor vacío

' Usos: en el inicio de sesión

Public Sub New()

End Sub

'Constructor

' en el constructor se establce una conexion con la base de datos le entra 3 parametros

'url de conxion que es unico para todos formularios

'queryAdaptador y datasetTableName son especificos por cada formulario

Public Sub New(queryAdaptador As String, datasetTableName As String)

conexion = New OleDbConnection(conexionUrl)

'adaptador de la tabla Emplado

adaptador = New OleDbDataAdapter(queryAdaptador, conexion)

midataset = New DataSet

adaptador.Fill(midataset, datasetTableName)

conexion.Close()

End Sub

'Metodos Getters

Public Function getAdapter() As OleDbDataAdapter

Return adaptador

End Function

Public Function getConxion() As OleDbConnection

Return conexion

End Function

Public Function getDataSet() As DataSet

Return midataset

End Function

'Metodo Buscar por nombre Empleado (por palabra )

Public Function buscarPorCadena(cadena As String, Table As String, whColumnName As String) As Integer

Dim pos As Integer = 0

Dim flag, flag2, flag3 As Integer

Try

conexion.Open()

Dim cmd As New OleDbCommand("SELECT \* FROM " & Table & "'", conexion)

'Aqui se crea un obj de la clase OldDBDataRereader que necsita un queriy para ejecutar la

Dim dr As OleDbDataReader = cmd.ExecuteReader

'recorremos la tabla

While dr.Read

If dr.Item("id") = cadena Then

flag2 = pos

flag = 1

Exit While

Else

pos += 1

End If

End While

If flag = 0 Then

MsgBox("Busqueda fallida id no existe . ", 64, "Gestion " & Table)

flag2 = -1

End If

conexion.Close()

Return flag2

Catch ex As Exception

mensajeErrorDatos()

errorLogWrite()

End Try

End Function

' Tomas -> Metodo Buscar por nombre Empleado (por palabra )

Public Function TbuscarEmpleado(nombre As String) As DataTable

Dim midatatable As New DataTable

midataset.Tables.Add(midatatable)

adaptador = New OleDbDataAdapter("SELECT \* FROM Empleado where nombre like '%" & nombre & "%'", conexion)

adaptador.Fill(midatatable)

Return midatatable

End Function

'Metodo Buscar por caracteres

Public Function buscarporCaracter(cadena As String, Table As String, whColumnName As String, datasetTableName As String)

'definir nuestra query con el parametro que es caracter que se escribe en el textbox

Dim comando As New OleDbCommand(("select \* from " & Table & " where " & whColumnName & " LIKE '%" & cadena & "%'"), conexion)

adaptador.SelectCommand = comando

'se limpia y carga el dataset nuevo

midataset.Clear()

adaptador.Fill(midataset, datasetTableName)

End Function

'Metodo Elminar una fila que elimina por id

Public Function eliminarFila(idFila As Integer, Table As String, whColumnName As String)

Dim result As Boolean

Try

'aqui se crear un objeto de la clase OleDbCommand con la query pereparada

Dim comando As New OleDbCommand(("delete from " & Table & " where " & whColumnName & " =@id"), conexion)

'aqui se anade el parametro recibido el parametro

comando.Parameters.AddWithValue("@id", idFila)

'aqui se abre la conexion

conexion.Open()

'aqui se ejecuta la query el metodo ExecuteNonQuery devulve un entero que es numero de filas affectadas

Dim filas\_affectas As Integer = comando.ExecuteNonQuery()

'aqui tenemos una condicion por si numero de filas es mayor que 0 signfica que se ha eliminda una fila

'ponemos el var flag igual a a true si no false

If filas\_affectas > 0 Then

result = True

Else

result = False

End If

'se limpia y carga el dataset nuevo

midataset.Clear()

adaptador.Fill(midataset, "Empleados")

'aqui se cierra la conexion con la base de datos

conexion.Close()

Catch ex As Exception

' MsgBox(ex)

mensajeErrorDatos()

errorLogWrite()

'aqui por si habra una excepcion cerramos la conxion con la base de datos

conexion.Close()

End Try

Return result

End Function

'Metodo Elminar una fila que modificar por id

Public Function modificarEmpleado(idFila As Integer, Dni As String, Nombre As String, Apellidos As String, Telefono As Integer, Table As String, whColumnName As String)

Dim result As Boolean

Try

'aqui se crear un objeto de la clase OleDbCommand con la query pereparada

Dim comando As New OleDbCommand(("update " & Table & "set dni=@dni ,nombre=@nombre,apellidos=@apellidos,telefono=@telefono Where " & whColumnName & " =@id"), conexion)

'aqui se anade el parametro recibido el parametro

comando.Parameters.AddWithValue("@dni", Dni)

comando.Parameters.AddWithValue("@nombre", Nombre)

comando.Parameters.AddWithValue("@apellidos", Apellidos)

comando.Parameters.AddWithValue("@telefono", Telefono)

comando.Parameters.AddWithValue("@id", idFila)

'aqui se abre la conexion

conexion.Open()

'aqui se ejecuta la query el metodo ExecuteNonQuery devulve un entero que es numero de filas affectadas

Dim filas\_affectas As Integer = comando.ExecuteNonQuery()

'aqui tenemos una condicion por si numero de filas es mayor que 0 signfica que se ha eliminda una fila

'ponemos el var flag igual a a true si no false

If filas\_affectas > 0 Then

result = True

Else

result = False

End If

'se limpia y carga el dataset nuevo

midataset.Clear()

adaptador.Fill(midataset, "Empleados")

'aqui se cierra la conexion con la base de datos

conexion.Close()

Catch ex As Exception

MsgBox("Error de Modificacion , ha intentado introducir un valor duplicado " & Chr(13), 64, "Gestion Empleados")

'mensajeErrorDatos()

errorLogWrite()

'aqui por si habra una excepcion cerramos la conxion con la base de datos

conexion.Close()

End Try

Return result

End Function

'Metodo insert Empleado

Public Function insertEmpleado(id As Integer, Dni As String, Nombre As String, Apellidos As String, Telefono As Integer) As Boolean

Dim result As Boolean

Try

Dim pos As Integer = 0

Dim flag As Integer

conexion.Open()

'select query del empleado

Dim cmd As New OleDbCommand("Select \* from Empleados", conexion)

'Aqui se crea un obj de la clase OldDBDataRereader que necsita un queriy para ejecutar la

Dim dr As OleDbDataReader = cmd.ExecuteReader

'recorremos la tabla

While dr.Read

'si id existe en la tabla no se hace nada

If dr.Item("id") = id Then

flag = 1

MsgBox("ID Empleado ya existe por favor introduce un nuevo id . ", 64, "Gestion Empleados")

ElseIf dr.Item("dni") = Dni Then

flag = 1

MsgBox("DNI Empleado ya existe por favor introduce un nuevo dni . ", 64, "Gestion Empleados")

Else

pos += 1

End If

End While

'si no si inserta nuevo empleado

If flag = 0 Then

'aqui se crea un objeto de clas OledbCommandBuilder para perparar el commando insert

Dim comando As New OleDbCommandBuilder(adaptador)

'recuperamos el insert command desde adaptador

adaptador.InsertCommand = comando.GetInsertCommand

'aqui objeto de clase DataRowCollection generar un hueco en dataset empleado

Dim dataRowCollection As DataRowCollection = midataset.Tables("Empleados").Rows

dataRowCollection.Add(id, Dni, Nombre, Apellidos, Telefono)

adaptador.Update(midataset.Tables("Empleados"))

result = True

End If

conexion.Close()

Catch ex As Exception

' MsgBox(ex.Message)

mensajeErrorDatos()

errorLogWrite()

result = False

conexion.Close()

End Try

Return result

End Function

'Metodo insert Gafas

Public Function insertGafas(id As Integer, ISBN As String, Nombre As String, Forma As String, Color As String, Precio As Single, Stock As Integer, Imagen As Byte()) As Boolean

Dim result As Boolean

Try

Dim pos As Integer = 0

Dim flag As Integer

conexion.Open()

Dim cmd As New OleDbCommand("Select \* from Gafas", conexion)

'Aqui se crea un obj de la clase OldDBDataRereader que necsita un queriy para ejecutar la

Dim dr As OleDbDataReader = cmd.ExecuteReader

'recorremos la tabla

While dr.Read

'si id existe en la tabla no se hace nada

If dr.Item("id") = id Then

flag = 1

MsgBox("ID Gafas ya existe por favor introduce un nuevo id . ", 64, "Gestion Gafas")

ElseIf dr.Item("isbn") = ISBN Then

flag = 2

MsgBox("ISBN Gafas ya existe por favor introduce un nuevo ISBN . ", 64, "Gestion Gafas")

Else

pos += 1

End If

End While

'si no si inserta nuevo empleado

If flag = 0 Then

'aqui se crea un objeto de clas OledbCommandBuilder para perparar el commando insert

Dim comando As New OleDbCommandBuilder(adaptador)

'recuperamos el insert command desde adaptador

adaptador.InsertCommand = comando.GetInsertCommand

'aqui objeto de clase DataRowCollection generar un hueco en dataset empleado

Dim dataRowCollection As DataRowCollection = midataset.Tables("Gafas").Rows

dataRowCollection.Add(id, ISBN, Nombre, Forma, Color, Precio, Stock, Imagen)

adaptador.Update(midataset.Tables("Gafas"))

result = True

End If

conexion.Close()

Catch ex As Exception

' MsgBox(ex.Message)

mensajeErrorDatos()

errorLogWrite()

result = False

conexion.Close()

End Try

Return result

End Function

'Metodo modificar Libro

Public Function ModificarGafas(idFila As Integer, ISBN As String, Nombre As String, Forma As String, Color As String, Precio As Single, Stock As Integer, Imagen As Byte(), Table As String, whColumnName As String) As Boolean

Dim result As Boolean

Try

'aqui se crear un objeto de la clase OleDbCommand con la query pereparada

Dim comando As New OleDbCommand(("update " & Table & "set isbn=@isbn ,nombre=@nombre,forma=@forma,color=@color,precio=@precio,stock=@stock,imagen=@imagen Where " & whColumnName & " =@id"), conexion)

'aqui se anade el parametro recibido el parametro

comando.Parameters.AddWithValue("@isbn", ISBN)

comando.Parameters.AddWithValue("@nombre", Nombre)

comando.Parameters.AddWithValue("@forma", Forma)

comando.Parameters.AddWithValue("@color", Color)

comando.Parameters.AddWithValue("@precio", Precio)

comando.Parameters.AddWithValue("@stock", Stock)

comando.Parameters.AddWithValue("@imagen", Imagen)

comando.Parameters.AddWithValue("@id", idFila)

'aqui se abre la conexion

conexion.Open()

'aqui se ejecuta la query el metodo ExecuteNonQuery devulve un entero que es numero de filas affectadas

Dim filas\_affectas As Integer = comando.ExecuteNonQuery()

'aqui tenemos una condicion por si numero de filas es mayor que 0 signfica que se ha eliminda una fila

'ponemos el var flag igual a a true si no false

If filas\_affectas > 0 Then

result = True

Else

result = False

End If

'se limpia y carga el dataset nuevo

midataset.Clear()

adaptador.Fill(midataset, "Gafas")

'aqui se cierra la conexion con la base de datos

conexion.Close()

Catch ex As Exception

MsgBox("Error de Modificacion , ha intentado introducir un valor duplicado " & Chr(13), 64, "Gestion Gafas")

'mensajeErrorDatos()

errorLogWrite()

'aqui por si habra una excepcion cerramos la conxion con la base de datos

conexion.Close()

End Try

Return result

End Function

'Metodo insert Cliente

Public Function insertClienteSocio(id As Integer, Nombre As String, Apellidos As String, Telefono As Integer, Email As String, fechaAlta As String, Socio As Boolean) As Boolean

Dim result As Boolean

Try

Dim pos As Integer = 0

Dim flag As Integer

conexion.Open()

Dim cmd As New OleDbCommand("Select \* from ClientesSocios", conexion)

'Aqui se crea un obj de la clase OldDBDataRereader que necsita un queriy para ejecutar la

Dim dr As OleDbDataReader = cmd.ExecuteReader

'recorremos la tabla

While dr.Read

'si id existe en la tabla no se hace nada

If dr.Item("id") = id Then

flag = 1

MsgBox("ID Cliente Socio ya existe por favor introduce un nuevo id . ", 64, "Gestion Clientes Socios")

Else

pos += 1

End If

End While

'si no si inserta nuevo empleado

If flag = 0 Then

'aqui se crea un objeto de clas OledbCommandBuilder para perparar el commando insert

Dim comando As New OleDbCommandBuilder(adaptador)

'recuperamos el insert command desde adaptador

adaptador.InsertCommand = comando.GetInsertCommand

'aqui objeto de clase DataRowCollection generar un hueco en dataset empleado

Dim dataRowCollection As DataRowCollection = midataset.Tables("ClientesSocios").Rows

dataRowCollection.Add(id, Nombre, Apellidos, Telefono, Email, fechaAlta, Socio)

adaptador.Update(midataset.Tables("ClientesSocios"))

result = True

End If

conexion.Close()

Catch ex As Exception

'MsgBox(ex.Message)

mensajeErrorDatos()

errorLogWrite()

result = False

conexion.Close()

End Try

Return result

End Function

'Metodo Modificar una fila que modificar por id

Public Function modificarCliente(idFila As Integer, Nombre As String, Apellidos As String, Telefono As Integer, Email As String, fechaAlta As String, Socio As Boolean, Table As String, whColumnName As String) As Boolean

Dim result As Boolean

Try

'aqui se crear un objeto de la clase OleDbCommand con la query pereparada

Dim comando As New OleDbCommand(("update " & Table & "set nombre=@nombre ,apellidos=@apellidos,telefono=@telefono,email=@email,fechaalta=@fechaAlta,socio=@socio Where " & whColumnName & " =@id"), conexion)

'aqui se anade el parametro recibido el parametro

comando.Parameters.AddWithValue("@nombre", Nombre)

comando.Parameters.AddWithValue("@apellidos", Apellidos)

comando.Parameters.AddWithValue("@telefono", Telefono)

comando.Parameters.AddWithValue("@email", Email)

comando.Parameters.AddWithValue("@fechaAlta", fechaAlta)

comando.Parameters.AddWithValue("@socio", Socio)

comando.Parameters.AddWithValue("@id", idFila)

'aqui se abre la conexion

conexion.Open()

'aqui se ejecuta la query el metodo ExecuteNonQuery devulve un entero que es numero de filas affectadas

Dim filas\_affectas As Integer = comando.ExecuteNonQuery()

'aqui tenemos una condicion por si numero de filas es mayor que 0 signfica que se ha eliminda una fila

'ponemos el var flag igual a a true si no false

If filas\_affectas > 0 Then

result = True

Else

result = False

End If

'se limpia y carga el dataset nuevo

midataset.Clear()

adaptador.Fill(midataset, "ClientesSocios")

'aqui se cierra la conexion con la base de datos

conexion.Close()

Catch ex As Exception

' MsgBox(ex)

mensajeErrorDatos()

errorLogWrite()

'aqui por si habra una excepcion cerramos la conxion con la base de datos

conexion.Close()

End Try

Return result

End Function

'Metodo insert Libro

Public Function insertLibro(id As Integer, ISBN As String, Titulo As String, Precio As Single, Autor As String, Editorial As String, Tapa As String, Stock As Integer, Genero As String, Categoria As String, Estado As String, Idioma As String, Imagen As Byte()) As Boolean

Dim result As Boolean

Try

Dim pos As Integer = 0

Dim flag As Integer

conexion.Open()

Dim cmd As New OleDbCommand("Select \* from Libros", conexion)

'Aqui se crea un obj de la clase OldDBDataRereader que necsita un queriy para ejecutar la

Dim dr As OleDbDataReader = cmd.ExecuteReader

'recorremos la tabla

While dr.Read

'si id existe en la tabla no se hace nada

If dr.Item("id") = id Then

flag = 1

MsgBox("ID Libro ya existe por favor introduce un nuevo id . ", 64, "Gestion Libros")

ElseIf dr.Item("isbn") = ISBN Then

flag = 2

MsgBox("ISBN Libro ya existe por favor introduce un nuevo ISBN . ", 64, "Gestion Libros")

Else

pos += 1

End If

End While

'si no si inserta nuevo empleado

If flag = 0 Then

'aqui se crea un objeto de clas OledbCommandBuilder para perparar el commando insert

Dim comando As New OleDbCommandBuilder(adaptador)

'recuperamos el insert command desde adaptador

adaptador.InsertCommand = comando.GetInsertCommand

'aqui objeto de clase DataRowCollection generar un hueco en dataset empleado

Dim dataRowCollection As DataRowCollection = midataset.Tables("Libros").Rows

dataRowCollection.Add(id, ISBN, Titulo, Precio, Autor, Editorial, Tapa, Stock, Genero, Categoria, Estado, Idioma, Imagen)

adaptador.Update(midataset.Tables("Libros"))

result = True

End If

conexion.Close()

Catch ex As Exception

' MsgBox(ex.Message)

mensajeErrorDatos()

errorLogWrite()

result = False

conexion.Close()

End Try

Return result

End Function

'Metodo modificar Libro

Public Function ModificarLibro(idFila As Integer, ISBN As String, Titulo As String, Precio As Single, Autor As String, Editorial As String, Tapa As String, Genero As String, Categoria As String, Estado As String, Idioma As String, Imagen As Byte(), Table As String, whColumnName As String) As Boolean

Dim result As Boolean

Try

'aqui se crear un objeto de la clase OleDbCommand con la query pereparada

Dim comando As New OleDbCommand(("update " & Table & "set isbn=@isbn ,titulo=@titulo,precio=@precio,autor=@autor,editorial=@editorial,tapa=@tapa,genero=@genero,categoria=@categoria,estado=@estado,idioma=@idioma,imagen=@imagen Where " & whColumnName & " =@id"), conexion)

'aqui se anade el parametro recibido el parametro

comando.Parameters.AddWithValue("@isbn", ISBN)

comando.Parameters.AddWithValue("@titulo", Titulo)

comando.Parameters.AddWithValue("@precio", Precio)

comando.Parameters.AddWithValue("@autor", Autor)

comando.Parameters.AddWithValue("@editorial", Editorial)

comando.Parameters.AddWithValue("@tapa", Tapa)

comando.Parameters.AddWithValue("@genero", Genero)

comando.Parameters.AddWithValue("@categoria", Categoria)

comando.Parameters.AddWithValue("@estado", Estado)

comando.Parameters.AddWithValue("@Idioma", Idioma)

comando.Parameters.AddWithValue("@imagen", Imagen)

comando.Parameters.AddWithValue("@id", idFila)

'aqui se abre la conexion

conexion.Open()

'aqui se ejecuta la query el metodo ExecuteNonQuery devulve un entero que es numero de filas affectadas

Dim filas\_affectas As Integer = comando.ExecuteNonQuery()

'aqui tenemos una condicion por si numero de filas es mayor que 0 signfica que se ha eliminda una fila

'ponemos el var flag igual a a true si no false

If filas\_affectas > 0 Then

result = True

Else

result = False

End If

'se limpia y carga el dataset nuevo

midataset.Clear()

adaptador.Fill(midataset, "Libros")

'aqui se cierra la conexion con la base de datos

conexion.Close()

Catch ex As Exception

MsgBox("Error de Modificacion , ha intentado introducir un valor duplicado " & Chr(13), 64, "Gestion Libros")

'mensajeErrorDatos()

errorLogWrite()

'aqui por si habra una excepcion cerramos la conxion con la base de datos

conexion.Close()

End Try

Return result

End Function

'Metodo modificarStock modfica el stock de la tabla libros por id

Public Function modificarStock(idFila As Integer, Stock As Integer, Table As String, whColumnName As String) As Boolean

Dim result As Boolean

Try

'aqui se crear un objeto de la clase OleDbCommand con la query pereparada

Dim comando As New OleDbCommand(("update " & Table & "set stock=@stock Where " & whColumnName & " =@id"), conexion)

'aqui se anade el parametro recibido el parametro

comando.Parameters.AddWithValue("@stock", Stock)

comando.Parameters.AddWithValue("@id", idFila)

'aqui se abre la conexion

conexion.Open()

'aqui se ejecuta la query el metodo ExecuteNonQuery devulve un entero que es numero de filas affectadas

Dim filas\_affectas As Integer = comando.ExecuteNonQuery()

'aqui tenemos una condicion por si numero de filas es mayor que 0 signfica que se ha eliminda una fila

'ponemos el var flag igual a a true si no false

If filas\_affectas > 0 Then

result = True

Else

result = False

End If

'se limpia y carga el dataset nuevo

midataset.Clear()

adaptador.Fill(midataset, "Libros")

'aqui se cierra la conexion con la base de datos

conexion.Close()

Catch ex As Exception

' MsgBox(ex)

mensajeErrorDatos()

errorLogWrite()

'aqui por si habra una excepcion cerramos la conxion con la base de datos

conexion.Close()

End Try

Return result

End Function

'Metodo insert Usuario

Public Function insertUsuario(id As Integer, Usuario As String, Clave As Integer, Rol As String, fkEmpleado As Integer) As Boolean

Dim result As Boolean

Try

Dim pos As Integer = 0

Dim flag As Integer

conexion.Open()

Dim cmd As New OleDbCommand("Select \* from Usuarios", conexion)

'Aqui se crea un obj de la clase OldDBDataRereader que necsita un queriy para ejecutar la

Dim dr As OleDbDataReader = cmd.ExecuteReader

'recorremos la tabla

While dr.Read

'si id existe en la tabla no se hace nada

If dr.Item("Usuario") = Usuario Then

'comprobacion si usuario existe en la tabla usuario

flag = 1

MsgBox(" Usuario ya existe por favor introduce un nuevo usuario . ", 64, "Gestion Usuarios")

'comprobacion si id existe en la tabla usuario

ElseIf dr.Item("id") = id Then

flag = 2

MsgBox(" id ya existe por favor introduce un nuevo id . ", 64, "Gestion Usuarios")

'comprobacion si usario esta relacionado con empleado

ElseIf dr.Item("fk\_empleado") = fkEmpleado Then

flag = 3

MsgBox(" Empleado ya tiene un usuario por favor seleccione un nuevo empleado . ", 64, "Gestion Usuarios")

Else

pos += 1

End If

End While

'si no si inserta nuevo usuario

If flag = 0 Then

'aqui se crea un objeto de clas OledbCommandBuilder para perparar el commando insert

Dim comando As New OleDbCommandBuilder(adaptador)

'recuperamos el insert command desde adaptador

adaptador.InsertCommand = comando.GetInsertCommand

'aqui objeto de clase DataRowCollection generar un hueco en dataset empleado

Dim dataRowCollection As DataRowCollection = midataset.Tables("Usuarios").Rows

dataRowCollection.Add(id, Usuario, Clave, Rol, fkEmpleado)

adaptador.Update(midataset.Tables("Usuarios"))

result = True

End If

conexion.Close()

Catch ex As Exception

MsgBox(ex.Message)

result = False

conexion.Close()

End Try

Return result

End Function

'Metodo modificarUsuario modfica el stock de la tabla libros por id

Public Function modificarUsuario(getIdFila As Integer, Usuario As String, Clave As Integer, Rol As String, fkEmpleado As Integer, Table As String, whColumnName As String) As Boolean

Dim result As Boolean

Try

Dim pos As Integer = 0

Dim flag As Integer = 0

conexion.Open()

Dim cmd As New OleDbCommand("Select \* from Usuarios", conexion)

'Aqui se crea un obj de la clase OldDBDataRereader que necsita un queriy para ejecutar la

Dim dr As OleDbDataReader = cmd.ExecuteReader

'recorremos la tabla

While dr.Read

'si id existe en la tabla no se hace nada

If dr.Item("Usuario") <> Usuario Then 'And dr.Item("fk\_empleado") <> fkEmpleado Then

flag = 0

'MsgBox(" Usuario ya existe por favor usa un nuevo usuario . ", 64, "Gestion Usuarios")

'MsgBox(" Este Empleado ya tiene un Usuario por favor Elige un nuevo Empleado . ", 64, "Gestion Usuarios")

Else

pos += 1

End If

End While

'si no si modifica nuevo usuario

If flag = 0 Then

' MsgBox("update")

'aqui se crear un objeto de la clase OleDbCommand con la query pereparada

Dim comando As New OleDbCommand(("update " & Table & "set usuario=@usuario,clave=@clave,rol=@rol,fk\_empleado=@fkEmpleado Where " & whColumnName & " =@id"), conexion)

'aqui se anade el parametro recibido el parametro

comando.Parameters.AddWithValue("@usuario", Usuario)

comando.Parameters.AddWithValue("@clave", Clave)

comando.Parameters.AddWithValue("@rol", Rol)

comando.Parameters.AddWithValue("@fkEmpleado", fkEmpleado)

comando.Parameters.AddWithValue("@id", getIdFila)

'aqui se ejecuta la query el metodo ExecuteNonQuery devulve un entero que es numero de filas affectadas

Dim filas\_affectas As Integer = comando.ExecuteNonQuery()

'aqui tenemos una condicion por si numero de filas es mayor que 0 signfica que se ha eliminda una fila

'ponemos el var flag igual a a true si no false

If filas\_affectas > 0 Then

result = True

Else

result = False

End If

' MsgBox(result)

'se limpia y carga el dataset nuevo

midataset.Clear()

adaptador.Fill(midataset, "Usuarios")

End If

'aqui se cierra la conexion con la base de datos

conexion.Close()

Catch ex As Exception

MsgBox("Error de Modificacion , ha intentado introducir un valor duplicado " & Chr(13), 64, "Gestion Usuarios")

' mensajeErrorDatos()

errorLogWrite()

'aqui por si habra una excepcion cerramos la conxion con la base de datos

conexion.Close()

End Try

Return result

End Function

'Metodo getEmpleados que sirver para rellenar id de los empleados en formulario Gestion Usuaios

Public Function getEmpleados(queryAdaptador As String, datasetTableName As String) As DataSet

conexion = New OleDbConnection(conexionUrl)

'adaptador de la tabla Emplado

Dim adaptador2 As New OleDbDataAdapter(queryAdaptador, conexion)

midataset2 = New DataSet

adaptador2.Fill(midataset2, datasetTableName)

conexion.Close()

Return midataset2

End Function

'metodo para obtener los datos de la relacion entre empleados y usuarios

Public Function getRelacionEmpUsr(queryAdaptadoRelacion As String, datasetTableRelacionName As String, queryAdaptadorEmp As String, queryAdaptadorUsr As String, datasetTableNameEmp As String, datasetTableNameUsr As String) As DataSet

'aqui se crea una conexion con la base de datos

conexion = New OleDbConnection(conexionUrl)

'preparar los adpaters

Dim adaptadorEmp As New OleDbDataAdapter(queryAdaptadorEmp, conexion)

Dim adaptadorUsr As New OleDbDataAdapter(queryAdaptadorUsr, conexion)

Dim adaptadorRelaciones As New OleDbDataAdapter(queryAdaptadoRelacion, conexion)

midataset3 = New DataSet

'cargar data set con los 3 tablas

adaptadorEmp.Fill(midataset3, datasetTableNameEmp)

adaptadorUsr.Fill(midataset3, datasetTableNameUsr)

adaptadorRelaciones.Fill(midataset3, datasetTableRelacionName)

Return midataset3

End Function

'Metodo Buscar por caracteres

Public Function buscarStock(cadena As String, Table As String, whColumnName As String, datasetTableName As String)

'definir nuestra query con el parametro que es caracter que se escribe en el textbox

Dim comando As New OleDbCommand(("select id,isbn,titulo,stock from " & Table & " where " & whColumnName & " LIKE '%" & cadena & "%'"), conexion)

adaptador.SelectCommand = comando

'se limpia y carga el dataset nuevo

midataset.Clear()

adaptador.Fill(midataset, datasetTableName)

' cmd = New OleDbCommand("SELECT " & campo & " FROM " & tabla & " where " & cond1 & " = '" & cond2 & "' and " & cond3 & " = " & Integer.Parse(cond4), conexion)

End Function

Public Function selectTabla(campo As String, tabla As String)

Dim midatatable As New DataTable

midataset.Tables.Add(midatatable)

adaptador = New OleDbDataAdapter("SELECT " & campo & " FROM " & tabla, conexion)

adaptador.Fill(midatatable)

Return midatatable

End Function

Public Function selectWhere(campo As String, tabla As String, cond1 As String, cond2 As String) As DataTable

Dim midatatable As New DataTable

midataset.Tables.Add(midatatable)

adaptador = New OleDbDataAdapter("SELECT " & campo & " FROM " & tabla & " where usuario = " & cond1 & " = '" & cond2 & "'", conexion)

adaptador.Fill(midatatable)

Return midatatable

End Function

Public Function selectDosWhere(campo As String, tabla As String, cond1 As String, cond2 As String, cond3 As String, cond4 As String) As Boolean

Dim m As Integer = 0

conexion.Open()

Dim correcto As Boolean = False

Dim numero As Integer

Dim cmd As OleDbCommand

If Integer.TryParse(cond4, numero) Then

cmd = New OleDbCommand("SELECT " & campo & " FROM " & tabla & " where " & cond1 & " = '" & cond2 & "' and " & cond3 & " = " & Integer.Parse(cond4), conexion)

Else

cmd = New OleDbCommand("SELECT " & campo & " FROM " & tabla & " where " & cond1 & " = '" & cond2 & "' and " & cond3 & " = '" & cond4 & "'", conexion)

End If

' la idea es:

' Select \* From Usuarios Where usuario = 'admin' and clave = 123456

' midataset.Tables.Add(midatatable)

' adaptador = New OleDbDataAdapter("SELECT " & campo & " FROM " & tabla & " where " & cond1 & " = '" & cond2 & "' and " & cond3 & " = '" & cond4 & "'", conexion)

'adaptador.Fill(midatatable)

Dim dr As OleDbDataReader = cmd.ExecuteReader

While dr.Read 'And m = 0

If dr.Item(cond1) = cond2 Then

correcto = True

m = 1

Else

End If

End While

If m = 0 Then

MsgBox("El usuario no existe, por favo asegúrese de haber introducido datos correctos.", 64, "Inicio de sesión")

End If

conexion.Close()

Return correcto

End Function

Public Function getRol(cond1 As String, cond2 As String, cond3 As String, cond4 As String) As Integer

Dim m As Integer = 0

conexion.Open()

Dim valorColumna As String

Dim numero As Integer

Dim cmd As OleDbCommand

' Depende el rol se usa un integer u otro

' 1 es Amin

' 2 es Usuario

' 3 es Guest

Dim queRol As Integer

cmd = New OleDbCommand("SELECT \* from usuarios where " & cond1 & " = '" & cond2 & "' and " & cond3 & " = " & Integer.Parse(cond4), conexion)

' la idea es:

' Select \* From Usuarios Where usuario = 'admin' and clave = 123456

' midataset.Tables.Add(midatatable)

' adaptador = New OleDbDataAdapter("SELECT " & campo & " FROM " & tabla & " where " & cond1 & " = '" & cond2 & "' and " & cond3 & " = '" & cond4 & "'", conexion)

'adaptador.Fill(midatatable)

Dim dr As OleDbDataReader = cmd.ExecuteReader

While dr.Read And m = 0

If dr.Item("rol") = "Administrador" Then

queRol = 1

m = 1

ElseIf dr.Item("rol").Equals("Usuario") Then

queRol = 2

m = 1

ElseIf dr.Item("rol").Equals("Guest") Then

' Se entra como guest

queRol = 3

m = 1

End If

End While

If m = 0 Then

MsgBox("El usuario no existe, por favo asegúrese de haber introducido datos correctos.", 64, "Inicio de sesión")

End If

conexion.Close()

Return queRol

End Function

' Método traido de libValidacionDatos

' Escribe un archivo en el debug con los errores. (Fecha, info. del error, etc.)

Public Function errorLogWrite() As Boolean

Dim correcto As Boolean = False

' Escribir en errorLog

Dim fichero = "Errorlog.txt"

Try

Dim fich As New FileStream(fichero, FileMode.Append, FileAccess.Write)

Dim rs As New StreamWriter(fich)

rs.WriteLine(Now & "---> " & "Error en la comprobación de datos." & Chr(13) & "Detalle: " & Err.Description & Chr(13) & "Número de error: " & Err.Number)

rs.Close()

fich.Close()

correcto = True

Catch ex As Exception

MsgBox(ex.Message)

End Try

Return correcto

End Function

' Método traido de libValidacionDatos

' Métodos sobrecargados: uno sin recibir parámetros, otro recibe un string, y otro recibe string y objeto exception.

' Retorna un string para mostrar en un msgbox o donde sea.

Public Function mensajeErrorDatos() As String

Dim s As String

s = "Error en la comprobación de datos." & Chr(13) & "Detalle: " & Err.Description & Chr(13) & "Número de error: " & Err.Number

MsgBox("Error en la comprobación de datos." & Chr(13) & "Detalle: " & Err.Description & Chr(13) & "Número de error: " & Err.Number)

errorLogWrite()

Return s

End Function

' Falta completar

Public Function mensajeErrorDatos(mensaje As String) As String

Dim s As String

s = "Error en la comprobación de datos." & Chr(13) & "Detalle: " & Err.Description & Chr(13) & "Número de error: " & Err.Number

' MsgBox("Error en la comprobación de datos." & Chr(13) & "Detalle: " & Err.Description & Chr(13) & "Número de error: " & Err.Number)

errorLogWrite()

Return s

End Function

' Método traido de libValidacionDatos

' Retorna un string para mostrar en un msgbox o donde sea.

Public Function mensajeErrorDatos(mensaje As String, ex As Exception) As String

Dim s As String

s = "Error en la comprobación de datos." & Chr(13) & "Detalle: " & Err.Description & Chr(13) & "Número de error: " & Err.Number

' MsgBox("Error en la comprobación de datos." & Chr(13) & "Detalle: " & Err.Description & Chr(13) & "Número de error: " & Err.Number)

errorLogWrite()

Return s

End Function

End Class

**PantallaInicio.vb**

Imports libValidacionDatos.Validacion

' Vamos a usar esta librería para validar datos (usuarios, nombres de empleados, libros, etc.)

Imports LibConexion

Imports System.IO

Public Class PantallaInicio

Dim validador As New libValidacionDatos.Validacion

Dim queryAdaptador As String = "SELECT \* FROM Usuarios"

Dim datasetTableName As String = "Usuarios"

Dim validCon As New LibConexion.GestionConexion(queryAdaptador, datasetTableName)

' Se usara en todos los formularios para saber con qué rol se ha accedido.

' Cada vez que cambia el texto del campo de usuario se ejecuta este método.

Private Sub txtUsuario\_TextChanged(sender As Object, e As EventArgs) Handles txtUsuario.TextChanged

Try

ProgressBar1.Visible = False

' Si se recibe 0, se está recibiendo que se va a validar un usuario. máximo 10 caracteres (minuscula, sin tildes, etc)

' Si es 1, es nombre.

' Si es 2 es apellido.

' Si es 3 es dirección.

If validador.validarUsuario(txtUsuario.Text, 0) Then

Else

MsgBox("Ha introducido carácter no válido o se ha excedido en la longitud.")

txtUsuario.Clear()

End If

Catch ex As Exception

validCon.mensajeErrorDatos()

validCon.errorLogWrite()

End Try

End Sub

' Cada vez que cambia el texto del campo de clave se ejecuta este método.

Private Sub txtClave\_TextChanged(sender As Object, e As EventArgs) Handles txtClave.TextChanged

' Si se recibe 0 se está validando contraseña en inicio de sesión. (se permiten 6 dígitos)

' Las dos lineas de abajo cambiarán porque lo que está es del proyecto anterior.

' Si se recibe 1 se está validando teléfono. (se permite 9 dígitos).

' Si se recibe 2 se está validando precios en productos

Try

If validador.validarCodigo(txtClave.Text, 0) Then

Else

MsgBox("Ha introducido carácter no válido o se ha excedido en la longitud.")

txtClave.Clear()

End If

Catch ex As Exception

validCon.mensajeErrorDatos()

validCon.errorLogWrite()

End Try

End Sub

' Botón acceder

' Cuando se pulsa el botón acceder se comprobará si los datos existen en la base de datos

' y si se corresponden con los datos introducidos.

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles btnAcceder.Click

Dim queRol As Integer

Try

If Not txtUsuario.Text = "" And Not txtClave.Text = "" Then

' Ralizamos query para ver si existen estos datos. Se pasan como parámetros los valores de los textBox

' Solo si se encuentra una fila se procederá a ver si el usuario encontrado es admin, usuario, etc...

If validCon.selectDosWhere("\*", "usuarios", "usuario", txtUsuario.Text, "clave", txtClave.Text) Then

queRol = validCon.getRol("usuario", txtUsuario.Text, "clave", txtClave.Text)

' Controlar con otra query si el rol es admin o usuario.

' Entramos como Admin

If queRol = 1 Then

MsgBox("Bienvenido Admin", 64, "Acceso")

iniciarProgessBar()

PantallaPrincipal.Show()

Me.Hide()

PantallaPrincipal.lbtxtRol.Text = "Administrador"

' Guardamos la hora de acceso del usuario (con sus datos):

Try

Dim datosAcceso As New FileStream("logAcceso.txt", FileMode.Append, FileAccess.Write)

Dim sw As New StreamWriter(datosAcceso)

sw.WriteLine("El usuario:" & txtUsuario.Text)

sw.WriteLine("La clave:" & txtClave.Text)

sw.WriteLine("Acceso en fecha: " & Now)

' Cerramos los flujos para escribir en el log de acceso.

sw.Close()

datosAcceso.Close()

Catch ex As Exception

validador.mensajeErrorDatos()

validador.errorLogWrite()

End Try

' Entramos como Usuario

ElseIf queRol = 2 Then

MsgBox("Bienvenido Usuario", 64, "Acceso")

iniciarProgessBar()

PantallaPrincipal.Show()

Me.Hide()

PantallaPrincipal.lbtxtRol.Text = "Usuario"

' Podemos decidir: según el formulario vamos poniendo qué permisos hay segun el contenido del label

' ponemos aqui qué accesos tiene cada rol.

' Aquí tiene la ventaja de que está todo en el mismo sitio, mientras que comprobar el label

' en cada formulario requiere justamente eso, ir form por form viendo qué está deshabilitado

' según el contenido del label. (Admin, usuario, etc).

' Vamos a probar la solución de Malik y a poner todo aquí de una:

'PantallaPrincipal.picUsuarios.Enabled = False

'PantallaPrincipal.picEmple.Enabled = False

GestionBusqueda.GestionDeUsuariosToolStripMenuItem1.Enabled = False

GestionBusqueda.GestionEmpleadosToolStripMenuItem.Enabled = False

GestionLibro.GestionDeEmpleadosToolStripMenuItem.Enabled = False

GestionLibro.GestionDeUsuariosToolStripMenuItem1.Enabled = False

GestionClienteSocio.GestionDeUsuariosToolStripMenuItem1.Enabled = False

GestionClienteSocio.GestionDeEmpleadosToolStripMenuItem1.Enabled = False

GestionProductos.GestionDeUsuariosToolStripMenuItem1.Enabled = False

GestionProductos.GestionEmpleadosToolStripMenuItem.Enabled = False

GestionStock.GestionEmpleadosToolStripMenuItem.Enabled = False

GestionStock.GestionDeUsuariosToolStripMenuItem1.Enabled = False

' Guardamos la hora de acceso del usuario (con sus datos):

Try

Dim datosAcceso As New FileStream("logAcceso.txt", FileMode.Append, FileAccess.Write)

Dim sw As New StreamWriter(datosAcceso)

sw.WriteLine("El usuario:" & txtUsuario.Text)

sw.WriteLine("La clave:" & txtClave.Text)

sw.WriteLine("Acceso en fecha: " & Now)

' Cerramos los flujos para escribir en el log de acceso.

sw.Close()

datosAcceso.Close()

Catch ex As Exception

validador.mensajeErrorDatos()

validador.errorLogWrite()

End Try

' Se entra como Guest /invitado.

ElseIf queRol = 3 Then

MsgBox("Bienvenido Guest", 64, "Acceso")

PantallaPrincipal.lbtxtRol.Text = "Guest"

iniciarProgessBar()

PantallaPrincipal.Show()

Me.Hide()

' Al guest solo se le permite buscar libros/realizar consultas.

'PantallaPrincipal.picUsuarios.Enabled = False

'PantallaPrincipal.picEmple.Enabled = False

'PantallaPrincipal.picStock.Enabled = False

'PantallaPrincipal.picSocios.Enabled = False

'PantallaPrincipal.btnGestionLibros.Enabled = False

GestionBusqueda.GestionDeClientesToolStripMenuItem1.Enabled = False

GestionBusqueda.GestionEmpleadosToolStripMenuItem.Enabled = False

GestionBusqueda.GestionDeUsuariosToolStripMenuItem1.Enabled = False

GestionBusqueda.GestiónProductosToolStripMenuItem.Enabled = False

GestionBusqueda.GestionDeLibrosToolStripMenuItem1.Enabled = False

GestionBusqueda.GestionDeStockToolStripMenuItem.Enabled = False

' Guardamos la hora de acceso del usuario (con sus datos):

Try

Dim datosAcceso As New FileStream("logAcceso.txt", FileMode.Append, FileAccess.Write)

Dim sw As New StreamWriter(datosAcceso)

sw.WriteLine("El usuario:" & txtUsuario.Text)

sw.WriteLine("La clave:" & txtClave.Text)

sw.WriteLine("Acceso en fecha: " & Now)

' Cerramos los flujos para escribir en el log de acceso.

sw.Close()

datosAcceso.Close()

Catch ex As Exception

validador.mensajeErrorDatos()

validador.errorLogWrite()

End Try

End If

End If

Else

MsgBox("Por favor, rellene los campos antes de intentar acceder a la aplicación.", 64, "Ausencia de datos.")

End If

Catch ex As Exception

validCon.mensajeErrorDatos()

validCon.errorLogWrite()

End Try

End Sub

' metodo para iniciar el progressbar primero se hace visible y luego se comprueba su valor

'luego se incrementa valor de progressbar con un boucle

Public Sub iniciarProgessBar()

Try

ProgressBar1.Visible = True

If ProgressBar1.Value >= 10000 Then

ProgressBar1.Value = 0

Else

For i = 1 To 10000

ProgressBar1.Value = i

Next

ProgressBar1.Visible = False

ProgressBar1.Value = ProgressBar1.Minimum

End If

Catch ex As Exception

validCon.mensajeErrorDatos()

validCon.errorLogWrite()

End Try

End Sub

'Metodo que responde al evento click del linklabel contacto

Private Sub linkContacto\_LinkClicked(sender As Object, e As LinkLabelLinkClickedEventArgs)

System.Diagnostics.Process.Start("https://www.casadellibro.com/atencion-al-clienteNF/consulta-dudas")

End Sub

Private Sub Button1\_Click\_1(sender As Object, e As EventArgs) Handles Button1.Click

System.Diagnostics.Process.Start("https://www.casadellibro.com/atencion-al-clienteNF/consulta-dudas")

End Sub

Private Sub PantallaInicio\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

End Sub

Private Sub btnAccederBackdoor\_Click(sender As Object, e As EventArgs) Handles btnAccederBackdoor.Click

PantallaPrincipal.Show()

PantallaPrincipal.lbtxtRol.Text = "Administrador"

End Sub

End Class

**GestionBusqueda.vb**

Imports System.ComponentModel

Public Class GestionBusqueda

Dim queryAdaptador As String = "SELECT \* FROM Libros"

Dim datasetTableName As String = "Libros"

Dim gestion As New LibConexion.GestionConexion(queryAdaptador, datasetTableName)

Public datatable As New DataTable

' Variables usadas para validar datos

Dim validDatos As New libValidacionDatos.Validacion

Private Sub GestionBusqueda\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

'cargar en el datagridview

'donde sacar los datos pues de la memoria cache dataset

cargarPantalla()

'añadir la fecha en status label

ToolStripStatusLabel1.Text = DateTime.Now.ToLongDateString.ToString

' Llamamos al método para ver qué botones de la barra de herramientas están disponibles para

' este usuario.

PantallaPrincipal.darPermisos()

End Sub

'metodo para Actualizar y cargar los datos en data grid view

Private Sub cargarPantalla()

Try

gestion.getDataSet.Clear()

gestion.getAdapter.Fill(gestion.getDataSet, "Libros")

'aqui se carga el DatagridView

DataGridView1.DataSource = gestion.getDataSet

DataGridView1.DataMember = "Libros"

'metodo DefaultCellStyle permite de definir la forma que van aparecer los valores de la columna precio en este caso C como currency

'docs.microsoft.com/en-us/previous-versions/dotnet/netframework-4.0/dwhawy9k(v=vs.100)?WT.mc\_id=DT-MVP-5003235

DataGridView1.Columns("Precio").DefaultCellStyle.Format = "C"

DataGridView1.ClearSelection()

'establacer el tamaño automatic de todoas la celdas del datagridview

'DataGridView1.Columns.Item("imagen").Width = 100

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento textChange del campo txtBusID se busca por caracter

'se usa el metodo buscarporCaracter() que recibe dos parametros la caracter y nombre de la columna correspondiente en la tabla

Private Sub txtBusID\_TextChanged(sender As Object, e As EventArgs) Handles txtBusID.TextChanged

Try

If (validDatos.validarCodigo(txtBusID.Text, 3)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("ID"), 64, "Datos")

txtBusID.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento textChange del campo txtBusISBN se busca por caracter

'se usa el metodo buscarporCaracter() que recibe dos parametros la caracter y nombre de la columna correspondiente en la tabla

Private Sub txtBusISBN\_TextChanged(sender As Object, e As EventArgs) Handles txtBusISBN.TextChanged

Try

gestion.buscarporCaracter(txtBusISBN.Text, "Libros", " isbn", "Libros")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView1.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento textChange del campo txtBusTitulo se busca por caracter

'se usa el metodo buscarporCaracter() que recibe dos parametros la caracter y nombre de la columna correspondiente en la tabla

Private Sub txtBusTitulo\_TextChanged(sender As Object, e As EventArgs) Handles txtBusTitulo.TextChanged

Try

gestion.buscarporCaracter(txtBusTitulo.Text, "Libros", " titulo", "Libros")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView1.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento textChange del campo txtBusEditorial se busca por caracter

'se usa el metodo buscarporCaracter() que recibe dos parametros la caracter y nombre de la columna correspondiente en la tabla

Private Sub txtBusEditorial\_TextChanged(sender As Object, e As EventArgs) Handles txtBusEditorial.TextChanged

Try

gestion.buscarporCaracter(txtBusEditorial.Text, "Libros", " editorial", "Libros")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView1.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento textChange del campo txtBusGenero se busca por caracter

'se usa el metodo buscarporCaracter() que recibe dos parametros la caracter y nombre de la columna correspondiente en la tabla

Private Sub txtBusGenero\_TextChanged(sender As Object, e As EventArgs) Handles txtBusGenero.TextChanged

Try

gestion.buscarporCaracter(txtBusGenero.Text, "Libros", " genero", "Libros")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView1.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento textChange del campo txtBusGenero se busca por caracter

'se usa el metodo buscarporCaracter() que recibe dos parametros la caracter y nombre de la columna correspondiente en la tabla

Private Sub txtBusAutor\_TextChanged(sender As Object, e As EventArgs) Handles txtBusAutor.TextChanged

Try

gestion.buscarporCaracter(txtBusAutor.Text, "Libros", " autor", "Libros")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView1.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'Metodo para limpiar los campos del formulario

Private Sub llimpiarPantalla()

txtBusID.Clear()

txtBusISBN.Clear()

txtBusTitulo.Clear()

txtBusEditorial.Clear()

txtBusGenero.Clear()

txtBusAutor.Clear()

cbxCategorias.Text = Nothing

cbxEstado.Text = Nothing

cbxIdioma.Text = Nothing

End Sub

' Volver a Menu principal

Private Sub MenúPrincipalToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles MenúPrincipalToolStripMenuItem.Click

Me.Close()

PantallaPrincipal.Show()

End Sub

Private Sub btnLimpiar\_Click(sender As Object, e As EventArgs) Handles btnLimpiar.Click

Try

llimpiarPantalla()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub GestionDeLibrosToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeLibrosToolStripMenuItem1.Click

Me.Close()

GestionLibro.Show()

End Sub

Private Sub GestionDeUsuariosToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeUsuariosToolStripMenuItem1.Click

Me.Close()

GestionUsuarios.Show()

End Sub

Private Sub GestionDeClientesToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeClientesToolStripMenuItem1.Click

Me.Close()

GestionClienteSocio.Show()

End Sub

Private Sub GestionDeStockToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestionDeStockToolStripMenuItem.Click

Me.Close()

GestionStock.Show()

End Sub

Private Sub GestionEmpleadosToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestionEmpleadosToolStripMenuItem.Click

Me.Close()

GestionEmpleados.Show()

End Sub

Private Sub cbxCategorias\_SelectedIndexChanged(sender As Object, e As EventArgs) Handles cbxCategorias.SelectedIndexChanged

Try

gestion.buscarporCaracter(cbxCategorias.Text, "Libros", "Categoria", "Libros")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView1.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub cbxEstado\_SelectedIndexChanged(sender As Object, e As EventArgs) Handles cbxEstado.SelectedIndexChanged

Try

gestion.buscarporCaracter(cbxEstado.Text, "Libros", "Estado", "Libros")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView1.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub cbxIdioma\_SelectedIndexChanged(sender As Object, e As EventArgs) Handles cbxIdioma.SelectedIndexChanged

Try

gestion.buscarporCaracter(cbxIdioma.Text, "Libros", "Idioma", "Libros")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView1.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click

Me.Close()

PantallaPrincipal.Show()

End Sub

'Metodo de evento cerrando fourmulario

Private Sub Form1\_Closing(Sender As Object, e As FormClosingEventArgs) Handles Me.FormClosing

PantallaPrincipal.Show()

Me.Dispose()

End Sub

Private Sub GestiónProductosToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestiónProductosToolStripMenuItem.Click

Me.Close()

GestionProductos.Show()

End Sub

Private Sub btnBuscar\_Click(sender As Object, e As EventArgs) Handles btnBuscar.Click

Try

'buscar por id

If txtBusID.Text = "" Then

MsgBox("Por favor rellene El campo id . ", 64, "Gestion Busqueda")

Else

Dim pos As Integer = gestion.buscarPorCadena(txtBusID.Text, "Libros", " id")

If pos <> -1 Then

DataGridView1.Rows(pos).Selected = True

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

End Class

**GestionClienteSocio.vb**

' No hace falta, en principio, importar la librería porque está en la carpeta del proyecto

' pero por las dudas, dejamos al línea.

Imports libValidacionDatos

Public Class GestionClienteSocio

'aqui se crea un objeto de la clase GestionConexion de la liberia

Dim queryAdaptador As String = "SELECT \* FROM ClientesSocios"

Dim datasetTableName As String = "ClientesSocios"

Dim gestion As New LibConexion.GestionConexion(queryAdaptador, datasetTableName)

Public datatable As New DataTable

Public getIdFila As String

' vvv Variables para la validación de datos vvvv

Dim validDatos As New libValidacionDatos.Validacion

Private Sub GestionClienteSocio\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

Try

' releacionar los campos de la tabla con los textbox se usando DataBindi txtID.DataBindings.Add("text", gestion.getDataSet, "ClientesSocios.id")

txtID.DataBindings.Add("text", gestion.getDataSet, "ClientesSocios.Id")

txtNombre.DataBindings.Add("text", gestion.getDataSet, "ClientesSocios.Nombre")

txtApellidos.DataBindings.Add("text", gestion.getDataSet, "ClientesSocios.Apellidos")

txtTelefono.DataBindings.Add("text", gestion.getDataSet, "ClientesSocios.Telefono")

txtEmail.DataBindings.Add("text", gestion.getDataSet, "ClientesSocios.Email")

fechaAlta.DataBindings.Add("text", gestion.getDataSet, "ClientesSocios.FechaAlta")

checkSocio.DataBindings.Add("checked", gestion.getDataSet, "ClientesSocios.Socio", True)

cargarPantalla()

'metodo para mostrar donde esta cursor y en datagridview del total de la filas

getNumeroRegistros()

'añadir la fecha en status label

ToolStripStatusLabel1.Text = DateTime.Now.ToLongDateString.ToString

' Llamamos al método para ver qué botones de la barra de herramientas están disponibles para

' este usuario.

PantallaPrincipal.darPermisos()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub MenúPrincipalToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles MenúPrincipalToolStripMenuItem.Click

Me.Close()

PantallaPrincipal.Show()

End Sub

' Alta de cliente/socio

Private Sub btnAlta\_Click(sender As Object, e As EventArgs) Handles btnAlta.Click

Try

Timer1.Enabled = True

If (txtID.Text = "" Or txtNombre.Text = "" Or txtApellidos.Text = "" Or txtTelefono.Text = "" Or txtEmail.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Clientes Socios")

Else

If fechaAlta.Value = Now.Date Then

' Validamos el correo

If (validDatos.validarEmail(txtEmail.Text)) Then

' Si es correcto seguimos con los insert

If gestion.insertClienteSocio(CInt(txtID.Text), txtNombre.Text, txtApellidos.Text, CInt(txtTelefono.Text), txtEmail.Text, fechaAlta.Value.ToShortDateString, Convert.ToBoolean(checkSocio.CheckState)) Then

cargarPantalla()

MsgBox("Cliente Insertado con éxito . ", 64, "Gestion Clientes Socios")

llimpiarPantalla()

cargarPantalla()

getNumeroRegistros()

Else

cargarPantalla()

MsgBox("Error Cliente no Insertado . ", MsgBoxStyle.Exclamation, "Gestion Clientes Socios")

End If

Else

' El correo no es válido y hay que cambiarlo.

txtEmail.Clear()

MsgBox(validDatos.mensajeInfoDatoInvalido("Correo electrónico"), 64, "Datos")

End If

Else

MsgBox("Por favor seleccione una fecha valida . ", 64, "Gestion Clientes Socios")

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'ojo todavia falta Arreglarla

'En el evento buttonBuscar busca por id es por

Private Sub Button2\_Click(sender As Object, e As EventArgs) Handles Button2.Click

Try

'buscar por id

If txtBusID.Text = "" Then

MsgBox("Por favor rellene El campo id . ", 64, "Gestion Clientes Socios")

Else

Dim pos As Integer = gestion.buscarPorCadena(txtBusID.Text, "clientessocios", " id")

If pos <> -1 Then

DataGridView1.Rows(pos).Selected = True

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento textChange del campo txtBusNombre se busca por caracter

'se usa el metodo buscarporCaracter() que recibe dos parametros la caracter y nombre de la columna correspondiente en la tabla

Private Sub txtBusNombre\_TextChanged(sender As Object, e As EventArgs) Handles txtBusNombre.TextChanged

Try

gestion.buscarporCaracter(txtBusNombre.Text, "clientessocios", " nombre", "clientessocios")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView1.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub txtBusID\_TextChanged(sender As Object, e As EventArgs) Handles txtBusID.TextChanged

Try

If (validDatos.validarCodigo(txtBusID.Text, 3)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("ID"), 64, "Datos")

txtBusID.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento textChange del campo txtBusNombre se busca por caracter

'se usa el metodo buscarporCaracter() que recibe dos parametros la caracter y nombre de la columna correspondiente en la tabla

Private Sub txtBusApellidos\_TextChanged(sender As Object, e As EventArgs) Handles txtBusApellidos.TextChanged

Try

gestion.buscarporCaracter(txtBusApellidos.Text, "clientessocios", " apellidos", "clientessocios")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView1.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento button Modificar para modifica una fila

Private Sub btnModificar\_Click(sender As Object, e As EventArgs) Handles btnModificar.Click

Try

If (txtID.Text = "" Or txtNombre.Text = "" Or txtApellidos.Text = "" Or txtTelefono.Text = "" Or txtEmail.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Clientes Socios")

Else

'esa condicion nos permite assegurar que una celda de la fila en datagridview es seleccionda para poder eleminar la fila

' y una vez seleccionda se recupera el id de la tabla para poder modifcar la fila

If DataGridView1.SelectedCells.Count > 0 Then

Dim valor As Integer = MsgBox("¿Esta seguro de que desea Modificar Cliente?", 36, "Gestion Clientes Socios")

If valor = MsgBoxResult.Yes Then

' MsgBox(getIdFila)

Timer1.Enabled = True

If fechaAlta.Value > Now.Date Then

MsgBox("Por favor seleccione una fecha valida . ", 64, "Gestion Clientes Socios")

Else

If gestion.modificarCliente(getIdFila, txtNombre.Text, txtApellidos.Text, txtTelefono.Text, txtEmail.Text, fechaAlta.Value.ToShortDateString, Convert.ToBoolean(checkSocio.CheckState), " ClientesSocios ", " id ") Then

MsgBox("Cliente Modificao con éxito . ", 64, "Gestion Clientes Socios")

cargarPantalla()

Else

MsgBox("Error Cliente no modificado . ", MsgBoxStyle.Exclamation, "Gestion Clientes Socios")

'aqui se deselecta las celdas de datagridview

cargarPantalla()

End If

End If

End If

Else

MsgBox("Por favor seleccione un cliente . ", 64, "Gestion Clientes Socios")

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento button Eliminar para eliminar una fila

Private Sub btnEliminar\_Click(sender As Object, e As EventArgs) Handles btnEliminar.Click

'esa condicion nos permite assegurar que una celda de la fila en datagridview es seleccionda para poder eleminar la fila

' y una vez seleccionda se recupera el id de la tabla para poder eliminar la fila

Try

If DataGridView1.SelectedCells.Count > 0 Then

Dim valor As Integer = MsgBox("¿Esta seguro de que desea Elminar Cliente?", 36, "Gestion Clientes Socios")

If valor = MsgBoxResult.Yes Then

Timer1.Enabled = True

' MsgBox(getIdFila)

If gestion.eliminarFila(getIdFila, " clientessocios ", " id ") Then

MsgBox("Cliente Eliminado con éxito . ", 64, "Gestion Clientes Socios")

llimpiarPantalla()

cargarPantalla()

getNumeroRegistros()

Else

MsgBox("Error Cliente no Eliminado . ", MsgBoxStyle.Exclamation, "Gestion Clientes Socios")

'aqui se deselecta las celdas de datagridview

cargarPantalla()

End If

End If

Else

MsgBox("Por favor seleccione un cliente . ", 64, "Gestion Clientes Socios")

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'metodo para Actualizar y cargar los datos en data grid view

Private Sub cargarPantalla()

Try

gestion.getDataSet.Clear()

gestion.getAdapter.Fill(gestion.getDataSet, "ClientesSocios")

'aqui se carga el DatagridView

DataGridView1.DataSource = gestion.getDataSet

DataGridView1.DataMember = "ClientesSocios"

DataGridView1.ClearSelection()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'Metodo para limpiar los campos del formulario

Private Sub llimpiarPantalla()

txtID.Clear()

txtNombre.Clear()

txtApellidos.Clear()

txtTelefono.Clear()

txtEmail.Clear()

checkSocio.Checked = False

'campos de busqueda

txtBusID.Clear()

txtBusNombre.Clear()

End Sub

'metodo para obtener elemento seleccionado del datagridview para poder programar el metodo Eliminar sin necesitar al biding

Private Sub DataGridView1\_CellClick(sender As Object, e As DataGridViewCellEventArgs) Handles DataGridView1.CellClick

'aqui se declara una var de tipo entero que va guardar el indice de la fila seleccionada usando evento e.RowIndex

Try

Dim indice As Integer

indice = e.RowIndex

'aqui se crea un objeto de la clase DataGridViewRow que nos va permitir obtener el valor

Dim selectedFila As DataGridViewRow

'aqui se obtiene la fila pasandole el indice

Try

selectedFila = DataGridView1.Rows(indice)

getIdFila = selectedFila.Cells(0).Value.ToString

'actualizar datagrid view

cargarPantalla()

Catch ex As Exception

MsgBox("Por favor selecciona una fila valida")

End Try

'MsgBox(getIdFila)

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' ------------------------------ VALIDACIÓN DE CAMPOS -------------------------------------------------------------

' Validación del textbox ID.

Private Sub txtID\_TextChanged(sender As Object, e As EventArgs) Handles txtID.TextChanged

Try

' Si se recibe 0 se está validando contraseña en inicio de sesión. (se permiten 6 dígitos)

' Si se recibe 1 se está validando teléfono. (se permite 9 dígitos).

' Si se recibe 2 se está validando precios en productos

' Si recibe 3 es Id de cliente.

If (validDatos.validarCodigo(txtID.Text, 3)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("ID"), 64, "Datos")

txtID.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Validación tel textbox Nombre.

Private Sub txtNombre\_TextChanged(sender As Object, e As EventArgs) Handles txtNombre.TextChanged

' Si es 0, se está recibiendo que se va a validar un usuario.

' Si es 1, es nombre.

' Si es 2 es apellido.

' Si es 3 es dirección. -> Ejemmplo

' Ejemplos abajo

' 4

' 5

' 6 etc...

' Pasamos el valor 1 porque es un nombre

Try

If (validDatos.validarUsuario(txtNombre.Text, 1)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Nombre"), 64, "Datos")

txtNombre.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Validar apellidos

Private Sub txtApellidos\_TextChanged(sender As Object, e As EventArgs) Handles txtApellidos.TextChanged

' Si es 0, se está recibiendo que se va a validar un usuario.

' Si es 1, es nombre.

' Si es 2 es apellido.

' Si es 3 es dirección. -> Ejemmplo

' Ejemplos abajo

' 4

' 5

' 6 etc...

' Pasamos el valor 1 porque es un nombre

Try

If (validDatos.validarUsuario(txtApellidos.Text, 2)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Apellidos"), 64, "Datos")

txtApellidos.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Validar teléfono

Private Sub txtTelefono\_TextChanged(sender As Object, e As EventArgs) Handles txtTelefono.TextChanged

' Si se recibe 0 se está validando contraseña en inicio de sesión. (se permiten 6 dígitos)

' Si se recibe 1 se está validando teléfono. (se permite 9 dígitos).

' Si se recibe 2 se está validando precios en productos

' vvv validaciones en proyecto Casa del libro:

' Si se recibe 3 se está validando un ID.

Try

If (validDatos.validarCodigo(txtTelefono.Text, 1)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Teléfono"), 64, "Datos")

txtTelefono.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'metodo para mostrar donde esta cursor y en datagridview del total de la filas

Private Sub getNumeroRegistros()

Try

lblNumero.Text = (BindingContext(gestion.getDataSet, "ClientesSocios").Position + 1).ToString + " de " +

(BindingContext(gestion.getDataSet, "ClientesSocios").Count).ToString

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnUltimo\_Click(sender As Object, e As EventArgs) Handles btnUltimo.Click

'BindingContext(midataset, "Tabla").Position = DataGridView1.RowCount - 1

Try

BindingContext(gestion.getDataSet, "ClientesSocios").Position = BindingContext(gestion.getDataSet, "ClientesSocios").Count - 1

btnUltimo.Enabled = False

btnSiguiete.Enabled = False

btnPrimero.Enabled = True

btnAnterior.Enabled = True

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnSiguiete\_Click(sender As Object, e As EventArgs) Handles btnSiguiete.Click

'colacrtese con bindingContext en la dataset dandlole nombre en la posicion

Try

BindingContext(gestion.getDataSet, "ClientesSocios").Position += 1

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnAnterior\_Click(sender As Object, e As EventArgs) Handles btnAnterior.Click

'colacrtese con bindingContext en la dataset dandlole nombre en la posicion

Try

BindingContext(gestion.getDataSet, "ClientesSocios").Position -= 1

btnSiguiete.Enabled = True

btnUltimo.Enabled = True

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnPrimero\_Click(sender As Object, e As EventArgs) Handles btnPrimero.Click

Try

BindingContext(gestion.getDataSet, "ClientesSocios").Position = 0

btnPrimero.Enabled = False

btnAnterior.Enabled = False

btnUltimo.Enabled = True

btnSiguiete.Enabled = True

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Guardar/alta

' Mismo código que botón alta

Private Sub toolsbtnGuardar\_Click(sender As Object, e As EventArgs) Handles toolsbtnGuardar.Click

Try

If (txtID.Text = "" Or txtNombre.Text = "" Or txtApellidos.Text = "" Or txtTelefono.Text = "" Or txtEmail.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Clientes Socios")

Else

' Validamos el correo

If (validDatos.validarEmail(txtEmail.Text)) Then

' Si es correcto seguimos con los insert

If gestion.insertClienteSocio(CInt(txtID.Text), txtNombre.Text, txtApellidos.Text, CInt(txtTelefono.Text), txtEmail.Text, fechaAlta.Value.ToShortDateString, Convert.ToBoolean(checkSocio.CheckState)) Then

cargarPantalla()

MsgBox("Cliente Insertado con éxito . ", 64, "Gestion Clientes Socios")

llimpiarPantalla()

cargarPantalla()

getNumeroRegistros()

Else

cargarPantalla()

MsgBox("Error Cliente no Insertado . ", MsgBoxStyle.Exclamation, "Gestion Clientes Socios")

End If

Else

' El correo no es válido y hay que cambiarlo.

txtEmail.Clear()

MsgBox(validDatos.mensajeInfoDatoInvalido("Correo electrónico"), 64, "Datos")

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Toolstrip Eliminar

' Misma función que botón eliminar.

Private Sub toolsbtnEliminar\_Click(sender As Object, e As EventArgs) Handles toolsbtnEliminar.Click

'esa condicion nos permite assegurar que una celda de la fila en datagridview es seleccionda para poder eleminar la fila

' y una vez seleccionda se recupera el id de la tabla para poder eliminar la fila

Try

If DataGridView1.SelectedCells.Count > 0 Then

Dim valor As Integer = MsgBox("¿Esta seguro de que desea Elminar Cliente?", 36, "Gestion Clientes Socios")

If valor = MsgBoxResult.Yes Then

' MsgBox(getIdFila)

If gestion.eliminarFila(getIdFila, " clientessocios ", " id ") Then

MsgBox("Cliente Eliminado con éxito . ", 64, "Gestion Clientes Socios")

llimpiarPantalla()

cargarPantalla()

getNumeroRegistros()

Else

MsgBox("Error Cliente no Eliminado . ", MsgBoxStyle.Exclamation, "Gestion Clientes Socios")

'aqui se deselecta las celdas de datagridview

cargarPantalla()

End If

End If

Else

MsgBox("Por favor seleccione un cliente . ", 64, "Gestion Clientes Socios")

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnLimpiar\_Click(sender As Object, e As EventArgs) Handles btnLimpiar.Click

llimpiarPantalla()

End Sub

Private Sub toolsbtnNuevo\_Click(sender As Object, e As EventArgs) Handles toolsbtnNuevo.Click

llimpiarPantalla()

End Sub

Private Sub ToolStripButton1\_Click(sender As Object, e As EventArgs) Handles ToolStripButton1.Click

Me.Close()

FormRepSocios.Show()

End Sub

'metodo timer

Private Sub Timer1\_Tick(sender As Object, e As EventArgs) Handles Timer1.Tick

Try

If ToolStripProgressBar1.Value <= ToolStripProgressBar1.Maximum - 1 Then

ToolStripProgressBar1.Value = ToolStripProgressBar1.Value + 50

Else

Timer1.Enabled = False

ToolStripProgressBar1.Value = 0

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub GestionDeLibrosToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeLibrosToolStripMenuItem1.Click

Me.Close()

GestionLibro.Show()

End Sub

Private Sub GestionDeUsuariosToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeUsuariosToolStripMenuItem1.Click

Me.Close()

GestionUsuarios.Show()

End Sub

Private Sub GestionDeBusquedaToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestionDeBusquedaToolStripMenuItem.Click

Me.Close()

GestionBusqueda.Show()

End Sub

Private Sub GestionDeClientesToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeEmpleadosToolStripMenuItem1.Click

Me.Close()

GestionEmpleados.Show()

End Sub

Private Sub MenúPrincipaToolStripMenuItem\_Click(sender As Object, e As EventArgs)

Me.Close()

PantallaPrincipal.Show()

End Sub

Private Sub GestionDeStockToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestionDeStockToolStripMenuItem.Click

Me.Close()

GestionStock.Show()

End Sub

'Metodo de evento cerrando fourmulario

Private Sub Form1\_Closing(Sender As Object, e As FormClosingEventArgs) Handles Me.FormClosing

PantallaPrincipal.Show()

Me.Dispose()

End Sub

Private Sub GestiónProductosToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestiónProductosToolStripMenuItem.Click

Me.Close()

GestionProductos.Show()

End Sub

End Class

**GestionEmpleados.vb**

Imports LibConexion

Public Class GestionEmpleados

'aqui se crea un objeto de la clase GestionConexion de la liberia

Dim queryAdaptador As String = "SELECT \* FROM Empleados"

Dim datasetTableName As String = "Empleados"

Dim gestion As New LibConexion.GestionConexion(queryAdaptador, datasetTableName)

Public datatable As New DataTable

Public getIdFila As String

Public getDNI As String

' Variables usadas para validar datos

Dim validDatos As New libValidacionDatos.Validacion

'Constructor formulario Gestion Empleados

Private Sub GestionEmpleados\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

Try

' releacionar los campos de la tabla con los textbox se usando DataBindings

txtID.DataBindings.Add("text", gestion.getDataSet, "Empleados.id")

txtDni.DataBindings.Add("text", gestion.getDataSet, "Empleados.Dni")

txtNombre.DataBindings.Add("text", gestion.getDataSet, "Empleados.Nombre")

txtApellidos.DataBindings.Add("text", gestion.getDataSet, "Empleados.Apellidos")

txtTelefono.DataBindings.Add("text", gestion.getDataSet, "Empleados.Telefono")

'txtUsuario.DataBindings.Add("text", gestion.getDataSet, "Empleados.Usuario")

'txtClave.DataBindings.Add("text", gestion.getDataSet, "Empleados.Clave")

'cargar en el datagridview

'donde sacar los datos pues de la memoria cache dataset

cargarPantalla()

' DataGridView1.DataSource = gestion.getDataSet

'y le damos nombre

' DataGridView1.DataMember = "Empleados"

'aqui se deselect los elementos en el data gridview

' DataGridView1.ClearSelection()

'metodo para mostrar donde esta cursor y en datagridview del total de la filas

getNumeroRegistros()

'añadir la fecha en status label

ToolStripStatusLabel1.Text = DateTime.Now.ToLongDateString.ToString

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

' Llamamos al método para ver qué botones de la barra de herramientas están disponibles para

' este usuario.

PantallaPrincipal.darPermisos()

End Sub

Private Sub MenúPrincipaToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles MenúPrincipaToolStripMenuItem.Click

Me.Close()

PantallaPrincipal.Show()

End Sub

Private Sub Button3\_Click\_1(sender As Object, e As EventArgs) Handles btnAlta.Click

Try

Timer1.Enabled = True

If (txtID.Text = "" Or txtDni.Text = "" Or txtNombre.Text = "" Or txtApellidos.Text = "" Or txtTelefono.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Empleados")

Else

' Comprobamos si el dni es válido.

If (validDatos.validarDNI(txtDni.Text)) Then

If gestion.insertEmpleado(CInt(txtID.Text), txtDni.Text, txtNombre.Text, txtApellidos.Text, CInt(txtTelefono.Text)) Then

cargarPantalla()

MsgBox("Empleado Insertado con éxito . ", 64, "Gestion Empleados")

llimpiarPantalla()

cargarPantalla()

getNumeroRegistros()

Else

cargarPantalla()

MsgBox("Error Empleado no Insertado . ", MsgBoxStyle.Exclamation, "Gestion Empleados")

End If

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("DNI"), 64, "Datos")

txtDni.Clear()

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'ojo todavia falta Arreglarla

'En el evento buttonBuscar busca por id es por

Private Sub Button2\_Click(sender As Object, e As EventArgs) Handles btnBuscar.Click

Try

'buscar por id

If txtBusID.Text = "" Then

MsgBox("Por favor rellene El campo id . ", 64, "Gestion Empleados")

Else

Dim pos As Integer = gestion.buscarPorCadena(txtBusID.Text, "Empleados", " id")

If pos <> -1 Then

DataGridView1.Rows(pos).Selected = True

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento textChange del campo txtBusNombre se busca por caracter

'se usa el metodo buscarporCaracter() que recibe dos parametros la caracter y nombre de la columna correspondiente en la tabla

Private Sub txtBusNombre\_TextChanged(sender As Object, e As EventArgs) Handles txtBusNombre.TextChanged

Try

gestion.buscarporCaracter(txtBusNombre.Text, "empleados", " nombre", "Empleados")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView1.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento textChange del campo txtBusDni se busca por caracter

'se usa el metodo buscarporCaracter() que recibe dos parametros la caracter y nombre de la columna correspondiente en la tabla

Private Sub txtBusDni\_TextChanged(sender As Object, e As EventArgs) Handles txtBusDni.TextChanged

Try

gestion.buscarporCaracter(txtBusDni.Text, "empleados", "dni", "Empleados")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView1.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento button Modificar para modifica una fila

Private Sub btnModificar\_Click(sender As Object, e As EventArgs) Handles btnModificar.Click

Try

If (txtID.Text = "" Or txtID.Text = "" Or txtNombre.Text = "" Or txtApellidos.Text = "" Or txtTelefono.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Empleados")

Else

'esa condicion nos permite assegurar que una celda de la fila en datagridview es seleccionda para poder eleminar la fila

' y una vez seleccionda se recupera el id de la tabla para poder modifcar la fila

If DataGridView1.SelectedCells.Count > 0 Then

Dim valor As Integer = MsgBox("¿Esta seguro de que desea Modificar Empleado?", 36, "Gestion Empleados")

If valor = MsgBoxResult.Yes Then

Timer1.Enabled = True

' MsgBox(getIdFila)

' Comprobamos si el dni es valido aun

If (validDatos.validarDNI(txtDni.Text)) Then

If gestion.modificarEmpleado(getIdFila, txtDni.Text, txtNombre.Text, txtApellidos.Text, txtTelefono.Text, " Empleados ", " id ") Then

MsgBox("Empleado Modificado con éxito . ", 64, "Gestion Empleados")

cargarPantalla()

Else

MsgBox("Error Empleado no modificado . ", MsgBoxStyle.Exclamation, "Gestion Empleados")

'aqui se deselecta las celdas de datagridview

cargarPantalla()

End If

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("DNI"), 64, "Datos")

txtTelefono.Clear()

End If

End If

Else

MsgBox("Por favor seleccione una un empleado . ", 64, "Gestion Empleados")

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento button Eliminar para eliminar una fila

Private Sub Button6\_Click(sender As Object, e As EventArgs) Handles btnEliminar.Click

'esa condicion nos permite assegurar que una celda de la fila en datagridview es seleccionda para poder eleminar la fila

' y una vez seleccionda se recupera el id de la tabla para poder eliminar la fila

Try

If (txtID.Text = "" Or txtID.Text = "" Or txtNombre.Text = "" Or txtApellidos.Text = "" Or txtTelefono.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Empleados")

Else

If DataGridView1.SelectedCells.Count > 0 Then

Dim valor As Integer = MsgBox("¿Esta seguro de que desea Elminar Empleado?", 36, "Gestion Empleados")

If valor = MsgBoxResult.Yes Then

' MsgBox(getIdFila)

Timer1.Enabled = True

If gestion.eliminarFila(getIdFila, " Empleados ", " id ") Then

MsgBox("Empleado Eliminado con éxito . ", 64, "Gestion Empleados")

llimpiarPantalla()

cargarPantalla()

getNumeroRegistros()

Else

MsgBox("Error Empleado no Eliminado . ", MsgBoxStyle.Exclamation, "Gestion Empleados")

'aqui se deselecta las celdas de datagridview

cargarPantalla()

getNumeroRegistros()

End If

End If

Else

MsgBox("Por favor seleccione un empleado . ", 64, "Gestion Empleados")

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'Metodo para limpiar los campos del formulario

Private Sub llimpiarPantalla()

txtID.Clear()

txtDni.Clear()

txtNombre.Clear()

txtApellidos.Clear()

txtTelefono.Clear()

'campos de busqueda

txtBusDni.Clear()

txtBusID.Clear()

txtBusNombre.Clear()

End Sub

'En el evento button limpiar se limpia todos los campos del forumalario

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles btnLimpiar.Click

llimpiarPantalla()

End Sub

'metodo para Actualizar y cargar los datos en data grid view

Private Sub cargarPantalla()

Try

gestion.getDataSet.Clear()

gestion.getAdapter.Fill(gestion.getDataSet, "Empleados")

'aqui se carga el DatagridView

DataGridView1.DataSource = gestion.getDataSet

DataGridView1.DataMember = "Empleados"

DataGridView1.ClearSelection()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'metodo para obtener elemento seleccionado del datagridview para poder programar el metodo Eliminar sin necesitar al biding

Private Sub DataGridView1\_CellClick(sender As Object, e As DataGridViewCellEventArgs) Handles DataGridView1.CellClick

'aqui se declara una var de tipo entero que va guardar el indice de la fila seleccionada usando evento e.RowIndex

Try

Dim indice As Integer

indice = e.RowIndex

'aqui se crea un objeto de la clase DataGridViewRow que nos va permitir obtener el valor

Dim selectedFila As DataGridViewRow

'aqui se obtiene la fila pasandole el indice

Try

selectedFila = DataGridView1.Rows(indice)

getIdFila = selectedFila.Cells(0).Value.ToString

getDNI = selectedFila.Cells(1).Value.ToString

'actualizar datagrid view

cargarPantalla()

Catch ex As Exception

MsgBox("Por favor seleccione un fila valida . ", 64, "Gestion Empleados")

End Try

'aqui se recupera el valor de la primera celda que id

' MsgBox(getIdFila)

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Comprobar que id es correcto

Private Sub txtID\_TextChanged(sender As Object, e As EventArgs) Handles txtID.TextChanged

' Si se recibe 0 se está validando contraseña en inicio de sesión. (se permiten 6 dígitos)

' Si se recibe 1 se está validando teléfono. (se permite 9 dígitos).

' Si se recibe 2 se está validando precios en productos

' Si recibe 3 es Id de cliente.

Try

If (validDatos.validarCodigo(txtID.Text, 3)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("ID"), 64, "Datos")

txtID.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Validación tel textbox Nombre.

Private Sub txtNombre\_TextChanged(sender As Object, e As EventArgs) Handles txtNombre.TextChanged

' Si es 0, se está recibiendo que se va a validar un usuario.

' Si es 1, es nombre.

' Si es 2 es apellido.

' Si es 3 es dirección. -> Ejemmplo

' Ejemplos abajo

' 4

' 5

' 6 etc...

' Pasamos el valor 1 porque es un nombre

Try

If (validDatos.validarUsuario(txtNombre.Text, 1)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Nombre"), 64, "Datos")

txtNombre.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Validar apellidos

Private Sub txtApellidos\_TextChanged(sender As Object, e As EventArgs) Handles txtApellidos.TextChanged

' Si es 0, se está recibiendo que se va a validar un usuario.

' Si es 1, es nombre.

' Si es 2 es apellido.

' Si es 3 es dirección. -> Ejemmplo

' Ejemplos abajo

' 4

' 5

' 6 etc...

' Pasamos el valor 1 porque es un nombre

Try

If (validDatos.validarUsuario(txtApellidos.Text, 2)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Apellidos"), 64, "Datos")

txtApellidos.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Validar teléfono

Private Sub txtTelefono\_TextChanged(sender As Object, e As EventArgs) Handles txtTelefono.TextChanged

' Si se recibe 0 se está validando contraseña en inicio de sesión. (se permiten 6 dígitos)

' Si se recibe 1 se está validando teléfono. (se permite 9 dígitos).

' Si se recibe 2 se está validando precios en productos

' vvv validaciones en proyecto Casa del libro:

' Si se recibe 3 se está validando un ID.

Try

If (validDatos.validarCodigo(txtTelefono.Text, 1)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Teléfono"), 64, "Datos")

txtTelefono.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'metodo para mostrar donde esta cursor y en datagridview del total de la filas

Private Sub getNumeroRegistros()

Try

lblNumero.Text = (BindingContext(gestion.getDataSet, "Empleados").Position + 1).ToString + " de " +

(BindingContext(gestion.getDataSet, "Empleados").Count).ToString

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnUltimo\_Click(sender As Object, e As EventArgs) Handles btnUltimo.Click

'BindingContext(midataset, "Tabla").Position = DataGridView1.RowCount - 1

Try

BindingContext(gestion.getDataSet, "Empleados").Position = BindingContext(gestion.getDataSet, "Empleados").Count - 1

btnUltimo.Enabled = False

btnSiguiete.Enabled = False

btnPrimero.Enabled = True

btnAnterior.Enabled = True

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnSiguiete\_Click(sender As Object, e As EventArgs) Handles btnSiguiete.Click

'colacrtese con bindingContext en la dataset dandlole nombre en la posicion

Try

BindingContext(gestion.getDataSet, "Empleados").Position += 1

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnAnterior\_Click(sender As Object, e As EventArgs) Handles btnAnterior.Click

'colacrtese con bindingContext en la dataset dandlole nombre en la posicion

Try

BindingContext(gestion.getDataSet, "Empleados").Position -= 1

btnSiguiete.Enabled = True

btnUltimo.Enabled = True

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnPrimero\_Click(sender As Object, e As EventArgs) Handles btnPrimero.Click

Try

BindingContext(gestion.getDataSet, "Empleados").Position = 0

btnPrimero.Enabled = False

btnAnterior.Enabled = False

btnUltimo.Enabled = True

btnSiguiete.Enabled = True

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub ToolStripButton1\_Click(sender As Object, e As EventArgs) Handles ToolStripButton1.Click

Me.Close()

FormRepEmple.Show()

End Sub

'metodo timer

Private Sub Timer1\_Tick(sender As Object, e As EventArgs) Handles Timer1.Tick

Try

If ToolStripProgressBar1.Value <= ToolStripProgressBar1.Maximum - 1 Then

ToolStripProgressBar1.Value = ToolStripProgressBar1.Value + 50

Else

Timer1.Enabled = False

ToolStripProgressBar1.Value = 0

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub GestionDeLibrosToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeLibrosToolStripMenuItem1.Click

Me.Close()

GestionLibro.Show()

End Sub

Private Sub GestionDeUsuariosToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeUsuariosToolStripMenuItem1.Click

Me.Close()

GestionUsuarios.Show()

End Sub

Private Sub GestionDeClientesToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeClientesToolStripMenuItem1.Click

Me.Close()

GestionClienteSocio.Show()

End Sub

Private Sub GestionDeBusquedaToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestionDeBusquedaToolStripMenuItem.Click

Me.Close()

GestionBusqueda.Show()

End Sub

Private Sub GestionDeStoclToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestionDeStoclToolStripMenuItem.Click

Me.Close()

GestionStock.Show()

End Sub

Private Sub toolsbtnNuevo\_Click(sender As Object, e As EventArgs) Handles toolsbtnNuevo.Click

llimpiarPantalla()

End Sub

' Botón toolStrip guardar, misma función que botón guardar.

Private Sub toolsbtnGuardar\_Click(sender As Object, e As EventArgs) Handles toolsbtnGuardar.Click

Try

Timer1.Enabled = True

If (txtID.Text = "" Or txtDni.Text = "" Or txtNombre.Text = "" Or txtApellidos.Text = "" Or txtTelefono.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Empleados")

Else

' Comprobamos si el dni es válido.

If (validDatos.validarDNI(txtDni.Text)) Then

If gestion.insertEmpleado(CInt(txtID.Text), txtDni.Text, txtNombre.Text, txtApellidos.Text, CInt(txtTelefono.Text)) Then

cargarPantalla()

MsgBox("Empleado Insertado con éxito . ", 64, "Gestion Empleados")

llimpiarPantalla()

cargarPantalla()

getNumeroRegistros()

Else

cargarPantalla()

MsgBox("Error Empleado no Insertado . ", MsgBoxStyle.Exclamation, "Gestion Empleados")

End If

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("DNI"), 64, "Datos")

txtDni.Clear()

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Botón toolStrip eliminar

' misma función que botón eliminar.

Private Sub toolsbtnEliminar\_Click(sender As Object, e As EventArgs) Handles toolsbtnEliminar.Click

'esa condicion nos permite assegurar que una celda de la fila en datagridview es seleccionda para poder eleminar la fila

' y una vez seleccionda se recupera el id de la tabla para poder eliminar la fila

Try

If (txtID.Text = "" Or txtID.Text = "" Or txtNombre.Text = "" Or txtApellidos.Text = "" Or txtTelefono.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Empleados")

Else

If DataGridView1.SelectedCells.Count > 0 Then

Dim valor As Integer = MsgBox("¿Esta seguro de que desea Elminar Empleado?", 36, "Gestion Empleados")

If valor = MsgBoxResult.Yes Then

' MsgBox(getIdFila)

Timer1.Enabled = True

If gestion.eliminarFila(getIdFila, " Empleados ", " id ") Then

MsgBox("Empleado Eliminado con éxito . ", 64, "Gestion Empleados")

llimpiarPantalla()

cargarPantalla()

getNumeroRegistros()

Else

MsgBox("Error Empleado no Eliminado . ", MsgBoxStyle.Exclamation, "Gestion Empleados")

'aqui se deselecta las celdas de datagridview

cargarPantalla()

getNumeroRegistros()

End If

End If

Else

MsgBox("Por favor seleccione un empleado . ", 64, "Gestion Empleados")

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'Metodo de evento cerrando fourmulario

Private Sub Form1\_Closing(Sender As Object, e As FormClosingEventArgs) Handles Me.FormClosing

PantallaPrincipal.Show()

Me.Dispose()

End Sub

Private Sub SplitContainer1\_Panel1\_Paint(sender As Object, e As PaintEventArgs) Handles SplitContainer1.Panel1.Paint

End Sub

Private Sub GestiónProductosToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestiónProductosToolStripMenuItem.Click

Me.Close()

GestionProductos.Show()

End Sub

Private Sub txtBusID\_TextChanged(sender As Object, e As EventArgs) Handles txtBusID.TextChanged

Try

If (validDatos.validarCodigo(txtBusID.Text, 3)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("ID"), 64, "Datos")

txtBusID.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

End Class

**GestionLibre.vb**

Imports System.IO

Imports libValidacionDatos

Public Class GestionLibro

Dim queryAdaptador As String = "SELECT \* FROM Libros"

Dim datasetTableName As String = "Libros"

Dim gestion As New LibConexion.GestionConexion(queryAdaptador, datasetTableName)

Public datatable As New DataTable

Public getIdFila As String

Dim ofd As OpenFileDialog

' Variables usadas para validar datos

Dim validDatos As New libValidacionDatos.Validacion

Private Sub GestionLibro\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

Try

' releacionar los campos de la tabla con los textbox se usando DataBindings

txtLibroID.DataBindings.Add("text", gestion.getDataSet, "Libros.id")

txtISBN.DataBindings.Add("text", gestion.getDataSet, "Libros.isbn")

txtTitulo.DataBindings.Add("text", gestion.getDataSet, "Libros.titulo")

txtPrecio.DataBindings.Add("text", gestion.getDataSet, "Libros.precio")

txtAutor.DataBindings.Add("text", gestion.getDataSet, "Libros.autor")

txtEditorial.DataBindings.Add("text", gestion.getDataSet, "Libros.Editorial")

txtTapa.DataBindings.Add("text", gestion.getDataSet, "Libros.tapa")

txtStock.DataBindings.Add("text", gestion.getDataSet, "Libros.stock")

txtGenero.DataBindings.Add("text", gestion.getDataSet, "Libros.genero")

cbxCategorias.DataBindings.Add("text", gestion.getDataSet, "Libros.categoria")

cbxEstado.DataBindings.Add("text", gestion.getDataSet, "Libros.estado")

cbxIdioma.DataBindings.Add("text", gestion.getDataSet, "Libros.idioma")

'picturebox databingdings se crea un objeto anonimo de la clase Binding que recibe 4 parametros

'el tipo que es Image, nombre del dataset , nombre del campo , un boolean que permite aplicar formato

imagenLibro.DataBindings.Add(New Binding("Image", gestion.getDataSet, "Libros.imagen", True))

'cargar en el datagridview

'donde sacar los datos pues de la memoria cache dataset

cargarPantalla()

'desabilitar campo de stock (no se peude gestion el stock , la gestion de libro)

txtStock.Enabled = True

'metodo para mostrar donde esta cursor y en datagridview del total de la filas

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

' Llamamos al método para ver qué botones de la barra de herramientas están disponibles para

' este usuario.

PantallaPrincipal.darPermisos()

End Sub

' Alta de Libro

Private Sub btnAlta\_Click(sender As Object, e As EventArgs) Handles btnAlta.Click

'damos nombre de imagen como el id del libro como es identificador unico

Try

If (txtLibroID.Text = "" Or txtISBN.Text = "" Or txtTitulo.Text = "" Or txtPrecio.Text = "" Or txtAutor.Text = "" Or txtEditorial.Text = "" Or txtTapa.Text = "" Or txtStock.Text = "" Or txtGenero.Text = "" Or cbxCategorias.Text = "" Or cbxEstado.Text = "" Or cbxIdioma.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Libros")

Else

If (validDatos.validarISBN(txtISBN.Text)) Then

If Not imagenLibro.Image Is Nothing Then

'se crea un objeto de clase Memorystream que va coger el tamaño de la imagen

Dim arrImage() As Byte

Dim mstream As New System.IO.MemoryStream()

'SPECIFIES THE FILE FORMAT OF THE IMAGE

imagenLibro.Image.Save(mstream, System.Drawing.Imaging.ImageFormat.Jpeg)

'RETURNS THE ARRAY OF UNSIGNED BYTES FROM WHICH THIS STREAM WAS CREATED

arrImage = mstream.GetBuffer()

'GET THE SIZE OF THE STREAM IN BYTES

Dim FileSize As UInt32

FileSize = mstream.Length

'CLOSES THE CURRENT STREAM AND RELEASE ANY RESOURCES ASSOCIATED WITH THE CURRENT STREAM

mstream.Close()

If gestion.insertLibro(CInt(txtLibroID.Text), txtISBN.Text, txtTitulo.Text, CSng(txtPrecio.Text), txtAutor.Text, txtEditorial.Text, txtTapa.Text, CInt(txtStock.Text), txtGenero.Text, cbxCategorias.Text, cbxEstado.Text, cbxIdioma.Text, arrImage) Then

MsgBox("Libro Insertado con éxito . ", 64, "Gestion Libros")

llimpiarPantalla()

cargarPantalla()

getNumeroRegistros()

Else

MsgBox("Error Libro no Insertado . ", MsgBoxStyle.Exclamation, "Gestion Libros")

End If

Else

MsgBox("Por favor seleccione un imagen . ", MsgBoxStyle.Exclamation, "Gestion Libros")

End If

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("ISBN"), 64, "Datos")

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub llimpiarPantalla()

txtISBN.Clear()

txtLibroID.Clear()

txtTitulo.Clear()

txtAutor.Clear()

txtEditorial.Clear()

txtTapa.Clear()

txtStock.Clear()

txtPrecio.Clear()

txtGenero.Clear()

cbxCategorias.Text = Nothing

cbxEstado.Text = Nothing

cbxIdioma.Text = Nothing

imagenLibro.Image = New PictureBox().Image

End Sub

Private Sub btnLimpiar\_Click(sender As Object, e As EventArgs) Handles btnLimpiar.Click

llimpiarPantalla()

End Sub

Private Sub btnExaminar\_Click(sender As Object, e As EventArgs) Handles btnExaminar.Click

Try

'objeto de openfiledialog

Dim odf As New OpenFileDialog()

odf.Title = "Seleccione una imagen ..."

'tipo de fichiero

odf.Filter = "Image Files(\*.BMP;\*.JPG;\*.GIF;\*.PNG)|\*.BMP;\*.JPG;\*.GIF;\*.PNG"

'inicio de la ruta

odf.InitialDirectory = Environment.GetFolderPath(Environment.SpecialFolder.Desktop)

If odf.ShowDialog() = DialogResult.OK Then

imagenLibro.Image = Image.FromFile(odf.FileName)

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento button Modificar para modifica una fila

Private Sub btnModificar\_Click(sender As Object, e As EventArgs) Handles btnModificar.Click

Try

If (txtLibroID.Text = "" Or txtISBN.Text = "" Or txtTitulo.Text = "" Or txtPrecio.Text = "" Or txtAutor.Text = "" Or txtEditorial.Text = "" Or txtTapa.Text = "" Or txtStock.Text = "" Or txtGenero.Text = "" Or cbxCategorias.Text = "" Or cbxEstado.Text = "" Or cbxIdioma.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Libros")

Else

'esa condicion nos permite assegurar que una celda de la fila en datagridview es seleccionda para poder eleminar la fila

' y una vez seleccionda se recupera el id de la tabla para poder modifcar la fila

If DataGridView1.SelectedCells.Count > 0 Then

Dim valor As Integer = MsgBox("¿Esta seguro de que desea Modificar Libro?", 36, "Gestion Libros")

If valor = MsgBoxResult.Yes Then

' MsgBox(getIdFila)

'se crea un objeto de clase Memorystream que va coger el tamaño de la imagen

Dim arrImage() As Byte

Dim mstream As New System.IO.MemoryStream()

'SPECIFIES THE FILE FORMAT OF THE IMAGE

imagenLibro.Image.Save(mstream, System.Drawing.Imaging.ImageFormat.Jpeg)

'RETURNS THE ARRAY OF UNSIGNED BYTES FROM WHICH THIS STREAM WAS CREATED

arrImage = mstream.GetBuffer()

'GET THE SIZE OF THE STREAM IN BYTES

Dim FileSize As UInt32

FileSize = mstream.Length

'CLOSES THE CURRENT STREAM AND RELEASE ANY RESOURCES ASSOCIATED WITH THE CURRENT STREAM

mstream.Close()

' Comprobamos que el ISBN no fue modificado incorrectamente.

If (validDatos.validarISBN(txtISBN.Text)) Then

If gestion.ModificarLibro(getIdFila, txtISBN.Text, txtTitulo.Text, CSng(txtPrecio.Text), txtAutor.Text, txtEditorial.Text, txtTapa.Text, txtGenero.Text, cbxCategorias.Text, cbxEstado.Text, cbxIdioma.Text, arrImage, " Libros ", " id ") Then

MsgBox("Libro Modificado con éxito . ", 64, "Gestion Libros")

cargarPantalla()

Else

MsgBox("Error Libro no modificado . ", MsgBoxStyle.Exclamation, "Gestion Libros")

'aqui se deselecta las celdas de datagridview

cargarPantalla()

End If

Else

' Error al modificar el isbn

MsgBox(validDatos.mensajeInfoDatoInvalido("ISBN"), 64, "Datos")

End If

End If

Else

MsgBox("Por favor seleccione un Libro . ", 64, "Gestion Libros")

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'metodo para Actualizar y cargar los datos en data grid view

Private Sub cargarPantalla()

Try

gestion.getDataSet.Clear()

gestion.getAdapter.Fill(gestion.getDataSet, "Libros")

'aqui se carga el DatagridView

DataGridView1.DataSource = gestion.getDataSet

DataGridView1.DataMember = "Libros"

'metodo DefaultCellStyle permite de definir la forma que van aparecer los valores de la columna precio en este caso C como currency

'docs.microsoft.com/en-us/previous-versions/dotnet/netframework-4.0/dwhawy9k(v=vs.100)?WT.mc\_id=DT-MVP-5003235

DataGridView1.Columns("Precio").DefaultCellStyle.Format = "C"

DataGridView1.ClearSelection()

'DataGridView1.Columns.Item("imagen").Width = 100

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'metodo para obtener elemento seleccionado del datagridview para poder programar el metodo Eliminar sin necesitar al biding

Private Sub DataGridView1\_CellClick(sender As Object, e As DataGridViewCellEventArgs) Handles DataGridView1.CellClick

Try

'aqui se declara una var de tipo entero que va guardar el indice de la fila seleccionada usando evento e.RowIndex

Dim indice As Integer

indice = e.RowIndex

'aqui se crea un objeto de la clase DataGridViewRow que nos va permitir obtener el valor

Dim selectedFila As DataGridViewRow

'aqui se obtiene la fila pasandole el indice

'aqui se obtiene la fila pasandole el indice

Try

selectedFila = DataGridView1.Rows(indice)

getIdFila = selectedFila.Cells(0).Value.ToString

'actualizar el datagridviwe

cargarPantalla()

Catch ex As Exception

MsgBox("Por favor selecciona una fila valida")

End Try

' MsgBox(getIdFila)

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'Modificar o establecer el tamano del imagen

'Private Sub DataGridView1\_CellPainting(sender As Object, e As DataGridViewCellPaintingEventArgs) Handles DataGridView1.CellPainting

' Try

' If (e.RowIndex < 0 Or e.ColumnIndex < 0) Then Return

' If DataGridView1.Rows(e.RowIndex).Cells(e.ColumnIndex).ValueType = GetType(Byte()) Then

' CType(DataGridView1.Rows(e.RowIndex).Cells(e.ColumnIndex),

' DataGridViewImageCell).ImageLayout = DataGridViewImageCellLayout.Zoom

' End If

' Catch ex As Exception

' gestion.mensajeErrorDatos()

' gestion.errorLogWrite()

' End Try

'End Sub

'En el evento button Eliminar para eliminar una fila

Private Sub btnEliminar\_Click(sender As Object, e As EventArgs) Handles btnEliminar.Click

Try

'esa condicion nos permite assegurar que una celda de la fila en datagridview es seleccionda para poder eleminar la fila

' y una vez seleccionda se recupera el id de la tabla para poder eliminar la fila

If DataGridView1.SelectedCells.Count > 0 Then

Dim valor As Integer = MsgBox("¿Esta seguro de que desea Elminar Libro?", 36, "Gestion Libros")

If valor = MsgBoxResult.Yes Then

' MsgBox(getIdFila)

If gestion.eliminarFila(getIdFila, " Libros ", " id ") Then

MsgBox("Libro Eliminado con éxito . ", 64, "Gestion Libros")

llimpiarPantalla()

cargarPantalla()

getNumeroRegistros()

Else

MsgBox("Error Libro no Eliminado . ", MsgBoxStyle.Exclamation, "Gestion Libros")

'aqui se deselecta las celdas de datagridview

cargarPantalla()

End If

End If

Else

MsgBox("Por favor seleccione un Libro . ", 64, "Gestion Libros")

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Volver al menú principal

Private Sub MenúPrincipalToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles MenúPrincipalToolStripMenuItem.Click

Me.Close()

PantallaPrincipal.Show()

End Sub

Private Sub txtTitulo\_TextChanged(sender As Object, e As EventArgs) Handles txtTitulo.TextChanged

' Si es 0, se está recibiendo que se va a validar un usuario.

' Si es 1, es nombre.

' Si es 2 es apellido.

' Si es 3 es dirección. -> Ejemmplo

' Ejemplos abajo

' 4

' 5

' 6 etc...

' Pasamos el valor 1 porque el título permite caracteres parecidos a los de un nombre de persona.

Try

If (validDatos.validarUsuario(txtTitulo.Text, 1)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Título"), 64, "Datos")

txtTitulo.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub txtPrecio\_TextChanged(sender As Object, e As EventArgs) Handles txtPrecio.TextChanged

' Si se recibe 0 se está validando contraseña en inicio de sesión. (se permiten 6 dígitos)

' Si se recibe 1 se está validando teléfono. (se permite 9 dígitos).

' Si se recibe 2 se está validando precios en productos

' Si recibe 3 es Id (cliente, libro, producto es igual).

Try

If (validDatos.validarCodigo(txtPrecio.Text, 2)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Precio"), 64, "Precio")

txtPrecio.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Validar autor

Private Sub txtAutor\_TextChanged(sender As Object, e As EventArgs) Handles txtAutor.TextChanged

Try

If (validDatos.validarUsuario(txtAutor.Text, 1)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Autor"), 64, "Datos")

txtAutor.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'metodo para mostrar donde esta cursor y en datagridview del total de la filas

Private Sub getNumeroRegistros()

Try

lblNumero.Text = (BindingContext(gestion.getDataSet, "Libros").Position + 1).ToString + " de " +

(BindingContext(gestion.getDataSet, "Libros").Count).ToString

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnUltimo\_Click(sender As Object, e As EventArgs) Handles btnUltimo.Click

'BindingContext(midataset, "Tabla").Position = DataGridView1.RowCount - 1

Try

BindingContext(gestion.getDataSet, "Libros").Position = BindingContext(gestion.getDataSet, "Libros").Count - 1

btnUltimo.Enabled = False

btnSiguiete.Enabled = False

btnPrimero.Enabled = True

btnAnterior.Enabled = True

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnSiguiete\_Click(sender As Object, e As EventArgs) Handles btnSiguiete.Click

Try

'colacrtese con bindingContext en la dataset dandlole nombre en la posicion

BindingContext(gestion.getDataSet, "Libros").Position += 1

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnAnterior\_Click(sender As Object, e As EventArgs) Handles btnAnterior.Click

'colacrtese con bindingContext en la dataset dandlole nombre en la posicion

Try

BindingContext(gestion.getDataSet, "Libros").Position -= 1

btnSiguiete.Enabled = True

btnUltimo.Enabled = True

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnPrimero\_Click(sender As Object, e As EventArgs) Handles btnPrimero.Click

Try

BindingContext(gestion.getDataSet, "Libros").Position = 0

btnPrimero.Enabled = False

btnAnterior.Enabled = False

btnUltimo.Enabled = True

btnSiguiete.Enabled = True

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Misma funcionalidad que botón limpiar

Private Sub toolsbtnNuevo\_Click(sender As Object, e As EventArgs) Handles toolsbtnNuevo.Click

llimpiarPantalla()

End Sub

' Guardar

' igual que botón alta.

Private Sub toolsbtnGuardar\_Click(sender As Object, e As EventArgs) Handles toolsbtnGuardar.Click

'damos nombre de imagen como el id del libro como es identificador unico

Try

If (txtLibroID.Text = "" Or txtISBN.Text = "" Or txtTitulo.Text = "" Or txtPrecio.Text = "" Or txtAutor.Text = "" Or txtEditorial.Text = "" Or txtTapa.Text = "" Or txtStock.Text = "" Or txtGenero.Text = "" Or cbxCategorias.Text = "" Or cbxEstado.Text = "" Or cbxIdioma.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Libros")

Else

If (validDatos.validarISBN(txtISBN.Text)) Then

'se crea un objeto de clase Memorystream que va coger el tamaño de la imagen

Dim arrImage() As Byte

Dim mstream As New System.IO.MemoryStream()

'SPECIFIES THE FILE FORMAT OF THE IMAGE

imagenLibro.Image.Save(mstream, System.Drawing.Imaging.ImageFormat.Jpeg)

'RETURNS THE ARRAY OF UNSIGNED BYTES FROM WHICH THIS STREAM WAS CREATED

arrImage = mstream.GetBuffer()

'GET THE SIZE OF THE STREAM IN BYTES

Dim FileSize As UInt32

FileSize = mstream.Length

'CLOSES THE CURRENT STREAM AND RELEASE ANY RESOURCES ASSOCIATED WITH THE CURRENT STREAM

mstream.Close()

If gestion.insertLibro(CInt(txtLibroID.Text), txtISBN.Text, txtTitulo.Text, CSng(txtPrecio.Text), txtAutor.Text, txtEditorial.Text, txtTapa.Text, CInt(txtStock.Text), txtGenero.Text, cbxCategorias.Text, cbxEstado.Text, cbxIdioma.Text, arrImage) Then

MsgBox("Libro Insertado con éxito . ", 64, "Gestion Libros")

llimpiarPantalla()

cargarPantalla()

getNumeroRegistros()

Else

MsgBox("Error Libro no Insertado . ", MsgBoxStyle.Exclamation, "Gestion Libros")

End If

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("ISBN"), 64, "Datos")

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Toolstrip Eliminar

' Igual que botón eliminar

Private Sub toolsbtnEliminar\_Click(sender As Object, e As EventArgs) Handles toolsbtnEliminar.Click

'esa condicion nos permite assegurar que una celda de la fila en datagridview es seleccionda para poder eleminar la fila

' y una vez seleccionda se recupera el id de la tabla para poder eliminar la fila

Try

If DataGridView1.SelectedCells.Count > 0 Then

Dim valor As Integer = MsgBox("¿Esta seguro de que desea Elminar Libro?", 36, "Gestion Libros")

If valor = MsgBoxResult.Yes Then

' MsgBox(getIdFila)

If gestion.eliminarFila(getIdFila, " Libros ", " id ") Then

MsgBox("Libro Eliminado con éxito . ", 64, "Gestion Libros")

llimpiarPantalla()

cargarPantalla()

getNumeroRegistros()

Else

MsgBox("Error Libro no Eliminado . ", MsgBoxStyle.Exclamation, "Gestion Libros")

'aqui se deselecta las celdas de datagridview

cargarPantalla()

End If

End If

Else

MsgBox("Por favor seleccione un Libro . ", 64, "Gestion Libros")

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Método para llamar al generador de informes para libros.

Private Sub ToolStripButton1\_Click(sender As Object, e As EventArgs) Handles ToolStripButton1.Click

Me.Close()

FormRepLibros.Show()

End Sub

Private Sub GestionDeStockToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestionDeStockToolStripMenuItem.Click

Me.Close()

GestionStock.Show()

End Sub

Private Sub GestionDeBusquedaToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestionDeBusquedaToolStripMenuItem.Click

Me.Close()

GestionBusqueda.Show()

End Sub

Private Sub GestionDeEmpleadosToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestionDeEmpleadosToolStripMenuItem.Click

Me.Close()

GestionEmpleados.Show()

End Sub

Private Sub GestionDeUsuariosToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeUsuariosToolStripMenuItem1.Click

Me.Close()

GestionUsuarios.Show()

End Sub

Private Sub GestionDeClientesToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeClientesToolStripMenuItem1.Click

Me.Close()

GestionClienteSocio.Show()

End Sub

Private Sub txtISBN\_TextChanged(sender As Object, e As EventArgs) Handles txtISBN.TextChanged

End Sub

'Metodo de evento cerrando fourmulario

Private Sub Form1\_Closing(Sender As Object, e As FormClosingEventArgs) Handles Me.FormClosing

PantallaPrincipal.Show()

Me.Dispose()

End Sub

Private Sub txtEditorial\_TextChanged(sender As Object, e As EventArgs) Handles txtEditorial.TextChanged

' Si es 0, se está recibiendo que se va a validar un usuario.

' Si es 1, es nombre.

' Si es 2 es apellido.

' Si es 3 es dirección. -> Ejemmplo

' Ejemplos abajo

' 4

' 5

' 6 etc...

' Pasamos el valor 1 porque el título permite caracteres parecidos a los de un nombre de persona.

Try

If (validDatos.validarUsuario(txtEditorial.Text, 1)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Título"), 64, "Datos")

txtEditorial.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub txtTapa\_TextChanged(sender As Object, e As EventArgs) Handles txtTapa.TextChanged

' Si es 0, se está recibiendo que se va a validar un usuario.

' Si es 1, es nombre.

' Si es 2 es apellido.

' Si es 3 es dirección. -> Ejemmplo

' Ejemplos abajo

' 4

' 5

' 6 etc...

' Pasamos el valor 1 porque el título permite caracteres parecidos a los de un nombre de persona.

Try

If (validDatos.validarUsuario(txtTapa.Text, 1)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Título"), 64, "Datos")

txtTapa.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub txtGenero\_TextChanged(sender As Object, e As EventArgs) Handles txtGenero.TextChanged

' Si es 0, se está recibiendo que se va a validar un usuario.

' Si es 1, es nombre.

' Si es 2 es apellido.

' Si es 3 es dirección. -> Ejemmplo

' Ejemplos abajo

' 4

' 5

' 6 etc...

' Pasamos el valor 1 porque el título permite caracteres parecidos a los de un nombre de persona.

Try

If (validDatos.validarUsuario(txtGenero.Text, 1)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Título"), 64, "Datos")

txtGenero.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub SplitContainer1\_Panel1\_Paint(sender As Object, e As PaintEventArgs) Handles SplitContainer1.Panel1.Paint

End Sub

Private Sub GestiónProductosToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestiónProductosToolStripMenuItem.Click

Me.Close()

GestionProductos.Show()

End Sub

Private Sub txtLibroID\_TextChanged(sender As Object, e As EventArgs) Handles txtLibroID.TextChanged

' Si se recibe 0 se está validando contraseña en inicio de sesión. (se permiten 6 dígitos)

' Si se recibe 1 se está validando teléfono. (se permite 9 dígitos).

' Si se recibe 2 se está validando precios en productos

' Si recibe 3 es Id (cliente, libro, producto es igual).

Try

If (validDatos.validarCodigo(txtLibroID.Text, 3)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("ID"), 64, "ID")

txtLibroID.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub txtStock\_TextChanged(sender As Object, e As EventArgs) Handles txtStock.TextChanged

' Si se recibe 0 se está validando contraseña en inicio de sesión. (se permiten 6 dígitos)

' Si se recibe 1 se está validando teléfono. (se permite 9 dígitos).

' Si se recibe 2 se está validando precios en productos

' Si recibe 3 es Id (cliente y libro es igual).

' Si recibe 4 es stock

Try

If (validDatos.validarCodigo(txtStock.Text, 3)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Stock"), 64, "Stock")

txtStock.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

End Class

**GestionProductos.vb**

Imports System.Windows.Forms

Public Class GestionProductos

'aqui se crea un objeto de la clase GestionConexion de la liberia

Dim queryAdaptador As String = "SELECT \* FROM Gafas"

Dim datasetTableName As String = "Gafas"

Dim gestion As New LibConexion.GestionConexion(queryAdaptador, datasetTableName)

Public datatable As New DataTable

Public getIdFila As String

Public getISBN As String

' Variables usadas para validar datos

Dim validDatos As New libValidacionDatos.Validacion

Private Sub ShowNewForm(ByVal sender As Object, ByVal e As EventArgs) Handles LimpiarToolStripButton.Click

limpiarPantalla()

End Sub

Private Sub OpenFile(ByVal sender As Object, ByVal e As EventArgs) Handles AltaToolStripButton.Click

'damos nombre de imagen como el id del libro como es identificador unico

Try

If (txtID.Text = "" Or txtISBN.Text = "" Or txtNombre.Text = "" Or cbxForma.Text = "" Or cbxColor.Text = "" Or txtPrecio.Text = "" Or txtStock.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Libros")

Else

If (validDatos.validarISBN(txtISBN.Text)) Then

If Not imagenGafas.Image Is Nothing Then

'se crea un objeto de clase Memorystream que va coger el tamaño de la imagen

Dim arrImage() As Byte

Dim mstream As New System.IO.MemoryStream()

'SPECIFIES THE FILE FORMAT OF THE IMAGE

imagenGafas.Image.Save(mstream, System.Drawing.Imaging.ImageFormat.Jpeg)

'RETURNS THE ARRAY OF UNSIGNED BYTES FROM WHICH THIS STREAM WAS CREATED

arrImage = mstream.GetBuffer()

'GET THE SIZE OF THE STREAM IN BYTES

Dim FileSize As UInt32

FileSize = mstream.Length

'CLOSES THE CURRENT STREAM AND RELEASE ANY RESOURCES ASSOCIATED WITH THE CURRENT STREAM

mstream.Close()

If gestion.insertGafas(CInt(txtID.Text), txtISBN.Text, txtNombre.Text, cbxForma.Text, cbxColor.Text, CSng(txtPrecio.Text), CInt(txtStock.Text), arrImage) Then

MsgBox("Gafa Insertado con éxito . ", 64, "Gestion Gafas")

limpiarPantalla()

cargarTab1()

getNumeroRegistros()

Else

MsgBox("Error Gafa no Insertado . ", MsgBoxStyle.Exclamation, "Gestion Gafas")

End If

Else

MsgBox("Por favor seleccione un imagen . ", MsgBoxStyle.Exclamation, "Gestion Gafas")

End If

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("ISBN"), 64, "Datos")

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub ExitToolsStripMenuItem\_Click(ByVal sender As Object, ByVal e As EventArgs)

Me.Close()

End Sub

Private Sub CascadeToolStripMenuItem\_Click(ByVal sender As Object, ByVal e As EventArgs)

Me.LayoutMdi(MdiLayout.Cascade)

End Sub

Private Sub TileVerticalToolStripMenuItem\_Click(ByVal sender As Object, ByVal e As EventArgs)

Me.LayoutMdi(MdiLayout.TileVertical)

End Sub

Private Sub TileHorizontalToolStripMenuItem\_Click(ByVal sender As Object, ByVal e As EventArgs)

Me.LayoutMdi(MdiLayout.TileHorizontal)

End Sub

Private Sub ArrangeIconsToolStripMenuItem\_Click(ByVal sender As Object, ByVal e As EventArgs)

Me.LayoutMdi(MdiLayout.ArrangeIcons)

End Sub

Private Sub CloseAllToolStripMenuItem\_Click(ByVal sender As Object, ByVal e As EventArgs)

' Cierre todos los formularios secundarios del principal.

For Each ChildForm As Form In Me.MdiChildren

ChildForm.Close()

Next

End Sub

Private m\_ChildFormNumber As Integer

'--------------------------------------------------------------------------------------------------

Private Sub GestionProductos\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

' releacionar los campos de la tabla con los textbox se usando DataBindings

txtID.DataBindings.Add("text", gestion.getDataSet, "Gafas.Id")

txtISBN.DataBindings.Add("text", gestion.getDataSet, "Gafas.isbn")

txtNombre.DataBindings.Add("text", gestion.getDataSet, "Gafas.nombre")

txtStock.DataBindings.Add("text", gestion.getDataSet, "Gafas.stock")

txtPrecio.DataBindings.Add("text", gestion.getDataSet, "Gafas.precio")

cbxForma.DataBindings.Add("text", gestion.getDataSet, "Gafas.forma")

cbxColor.DataBindings.Add("text", gestion.getDataSet, "Gafas.color")

'picturebox databingdings se crea un objeto anonimo de la clase Binding que recibe 4 parametros

'el tipo que es Image, nombre del dataset , nombre del campo , un boolean que permite aplicar formato

imagenGafas.DataBindings.Add(New Binding("Image", gestion.getDataSet, "Gafas.imagen", True))

'cargar en el datagridview de la tab1

'donde sacar los datos pues de la memoria cache dataset

cargarTab1()

'cargar en el datagridview de la tab2

cargarTab2()

'desabilitar campo de stock (no se peude gestion el stock , la gestion de libro)

txtStock.Enabled = True

'metodo para mostrar donde esta cursor y en datagridview del total de la filas

getNumeroRegistros()

' Llamamos al método para ver qué botones de la barra de herramientas están disponibles para

' este usuario.

PantallaPrincipal.darPermisos()

End Sub

'Metodo para limpiar los campos del formulario

Private Sub limpiarPantalla()

txtID.Clear()

txtISBN.Clear()

txtNombre.Clear()

txtPrecio.Clear()

txtStock.Clear()

cbxForma.Text = Nothing

cbxColor.Text = Nothing

imagenGafas.Image = New PictureBox().Image

End Sub

'metodo para Actualizar y cargar los datos en data grid view

Private Sub cargarTab1()

Try

gestion.getDataSet.Clear()

gestion.getAdapter.Fill(gestion.getDataSet, "Gafas")

'aqui se carga el DatagridView

DataGridView1.DataSource = gestion.getDataSet

DataGridView1.DataMember = "Gafas"

'metodo DefaultCellStyle permite de definir la forma que van aparecer los valores de la columna precio en este caso C como currency

'docs.microsoft.com/en-us/previous-versions/dotnet/netframework-4.0/dwhawy9k(v=vs.100)?WT.mc\_id=DT-MVP-5003235

DataGridView1.Columns("Precio").DefaultCellStyle.Format = "C"

DataGridView1.ClearSelection()

'DataGridView1.Columns.Item("imagen").Width = 100

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'metodo para Actualizar y cargar los datos en data grid view

Private Sub cargarTab2()

Try

gestion.getDataSet.Clear()

gestion.getAdapter.Fill(gestion.getDataSet, "Gafas")

'aqui se carga el DatagridView

DataGridView2.DataSource = gestion.getDataSet

DataGridView2.DataMember = "Gafas"

'metodo DefaultCellStyle permite de definir la forma que van aparecer los valores de la columna precio en este caso C como currency

'docs.microsoft.com/en-us/previous-versions/dotnet/netframework-4.0/dwhawy9k(v=vs.100)?WT.mc\_id=DT-MVP-5003235

DataGridView2.Columns("Precio").DefaultCellStyle.Format = "C"

DataGridView2.ClearSelection()

'DataGridView1.Columns.Item("imagen").Width = 100

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'metodo para obtener elemento seleccionado del datagridview para poder programar el metodo Eliminar sin necesitar al biding

Private Sub DataGridView1\_CellClick(sender As Object, e As DataGridViewCellEventArgs) Handles DataGridView1.CellClick

'aqui se declara una var de tipo entero que va guardar el indice de la fila seleccionada usando evento e.RowIndex

Try

Dim indice As Integer

indice = e.RowIndex

'aqui se crea un objeto de la clase DataGridViewRow que nos va permitir obtener el valor

Dim selectedFila As DataGridViewRow

'aqui se obtiene la fila pasandole el indice

Try

selectedFila = DataGridView1.Rows(indice)

getIdFila = selectedFila.Cells(0).Value.ToString

getISBN = selectedFila.Cells(1).Value.ToString

'actualizar datagrid view

cargarTab1()

Catch ex As Exception

MsgBox("Por favor seleccione un fila valida . ", 64, "Gestion Gafas")

End Try

'aqui se recupera el valor de la primera celda que id

' MsgBox(getIdFila)

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'metodo para mostrar donde esta cursor y en datagridview del total de la filas

Private Sub getNumeroRegistros()

Try

lblNumero.Text = (BindingContext(gestion.getDataSet, "Gafas").Position + 1).ToString + " de " +

(BindingContext(gestion.getDataSet, "Gafas").Count).ToString

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnUltimo\_Click(sender As Object, e As EventArgs) Handles btnUltimo.Click

'BindingContext(midataset, "Tabla").Position = DataGridView1.RowCount - 1

Try

BindingContext(gestion.getDataSet, "Gafas").Position = BindingContext(gestion.getDataSet, "Gafas").Count - 1

btnUltimo.Enabled = False

btnSiguiete.Enabled = False

btnPrimero.Enabled = True

btnAnterior.Enabled = True

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnSiguiete\_Click(sender As Object, e As EventArgs) Handles btnSiguiete.Click

'colocarse con bindingContext en la dataset dandlole nombre en la posicion

Try

BindingContext(gestion.getDataSet, "Gafas").Position += 1

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnAnterior\_Click(sender As Object, e As EventArgs) Handles btnAnterior.Click

'colacarse con bindingContext en la dataset dandlole nombre en la posicion

Try

BindingContext(gestion.getDataSet, "Gafas").Position -= 1

btnSiguiete.Enabled = True

btnUltimo.Enabled = True

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnPrimero\_Click(sender As Object, e As EventArgs) Handles btnPrimero.Click

Try

BindingContext(gestion.getDataSet, "Gafas").Position = 0

btnPrimero.Enabled = False

btnAnterior.Enabled = False

btnUltimo.Enabled = True

btnSiguiete.Enabled = True

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'Metodo de evento cerrando fourmulario

Private Sub Form1\_Closing(Sender As Object, e As FormClosingEventArgs) Handles Me.FormClosing

PantallaPrincipal.Show()

Me.Dispose()

End Sub

Private Sub btnAlta\_Click(sender As Object, e As EventArgs) Handles btnAlta.Click

'damos nombre de imagen como el id del libro como es identificador unico

Try

If (txtID.Text = "" Or txtISBN.Text = "" Or txtNombre.Text = "" Or cbxForma.Text = "" Or cbxColor.Text = "" Or txtPrecio.Text = "" Or txtStock.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Libros")

Else

If (validDatos.validarISBN(txtISBN.Text)) Then

If Not imagenGafas.Image Is Nothing Then

'se crea un objeto de clase Memorystream que va coger el tamaño de la imagen

Dim arrImage() As Byte

Dim mstream As New System.IO.MemoryStream()

'Especifica el formato de imagen imagenGafas.Image.Save(mstream, System.Drawing.Imaging.ImageFormat.Jpeg)

'Devuelve el array de bytes

arrImage = mstream.GetBuffer()

'Coge el tamaño del stream en bytes Dim FileSize As UInt32

FileSize = mstream.Length

'Cierra el stream actual

mstream.Close()

If gestion.insertGafas(CInt(txtID.Text), txtISBN.Text, txtNombre.Text, cbxForma.Text, cbxColor.Text, CSng(txtPrecio.Text), CInt(txtStock.Text), arrImage) Then

MsgBox("Gafa Insertado con éxito . ", 64, "Gestion Gafas")

limpiarPantalla()

cargarTab1()

getNumeroRegistros()

Else

MsgBox("Error Gafa no Insertado . ", MsgBoxStyle.Exclamation, "Gestion Gafas")

End If

Else

MsgBox("Por favor seleccione un imagen . ", MsgBoxStyle.Exclamation, "Gestion Gafas")

End If

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("ISBN"), 64, "Datos")

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnExaminar\_Click(sender As Object, e As EventArgs) Handles btnExaminar.Click

Try

'objeto de openfiledialog

Dim odf As New OpenFileDialog()

odf.Title = "Seleccione una imagen ..."

'tipo de fichiero

odf.Filter = "Image Files(\*.BMP;\*.JPG;\*.GIF;\*.PNG)|\*.BMP;\*.JPG;\*.GIF;\*.PNG"

'inicio de la ruta

odf.InitialDirectory = Environment.GetFolderPath(Environment.SpecialFolder.Desktop)

If odf.ShowDialog() = DialogResult.OK Then

imagenGafas.Image = Image.FromFile(odf.FileName)

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnModificar\_Click(sender As Object, e As EventArgs) Handles btnModificar.Click

Try

If (txtID.Text = "" Or txtISBN.Text = "" Or txtNombre.Text = "" Or cbxForma.Text = "" Or cbxColor.Text = "" Or txtPrecio.Text = "" Or txtStock.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Gafas")

Else

'esa condicion nos permite assegurar que una celda de la fila en datagridview es seleccionda para poder eleminar la fila

' y una vez seleccionda se recupera el id de la tabla para poder modifcar la fila

If DataGridView1.SelectedCells.Count > 0 Then

Dim valor As Integer = MsgBox("¿Esta seguro de que desea Modificar Libro?", 36, "Gestion Gafas")

If valor = MsgBoxResult.Yes Then

'MsgBox(getIdFila)

'se crea un objeto de clase Memorystream que va coger el tamaño de la imagen

Dim arrImage() As Byte

Dim mstream As New System.IO.MemoryStream()

'especifica el formato de la imagen imagenGafas.Image.Save(mstream, System.Drawing.Imaging.ImageFormat.Jpeg)

'devuelve array de bytes

arrImage = mstream.GetBuffer()

'GET THE SIZE OF THE STREAM IN BYTES

Dim FileSize As UInt32

FileSize = mstream.Length

' Cierra el flujo

mstream.Close()

' Comprobamos que el ISBN no fue modificado incorrectamente.

If (validDatos.validarISBN(txtISBN.Text)) Then

If gestion.ModificarGafas(getIdFila, txtISBN.Text, txtNombre.Text, cbxForma.Text, cbxColor.Text, CSng(txtPrecio.Text), CInt(txtStock.Text), arrImage, " Gafas ", " id ") Then

MsgBox("Gafa Modificado con éxito . ", 64, "Gestion Gafas")

cargarTab1()

Else

MsgBox("Error Gafa no modificado . ", MsgBoxStyle.Exclamation, "Gestion Gafas")

'aqui se deselecta las celdas de datagridview

cargarTab1()

End If

Else

' Error al modificar el isbn

MsgBox(validDatos.mensajeInfoDatoInvalido("ISBN"), 64, "Datos")

End If

End If

Else

MsgBox("Por favor seleccione una Gafa . ", 64, "Gestion Gafas")

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnEliminar\_Click(sender As Object, e As EventArgs) Handles btnEliminar.Click

Try

'esa condicion nos permite assegurar que una celda de la fila en datagridview es seleccionda para poder eleminar la fila

' y una vez seleccionda se recupera el id de la tabla para poder eliminar la fila

If DataGridView1.SelectedCells.Count > 0 Then

Dim valor As Integer = MsgBox("¿Esta seguro de que desea Elminar Libro?", 36, "Gestion Gafas")

If valor = MsgBoxResult.Yes Then

' MsgBox(getIdFila)

If gestion.eliminarFila(getIdFila, " Gafas ", " id ") Then

MsgBox("Gafa Eliminada con éxito . ", 64, "Gestion Gafas")

limpiarPantalla()

cargarTab1()

getNumeroRegistros()

Else

MsgBox("Error gafa no Eliminada . ", MsgBoxStyle.Exclamation, "Gestion Gafas")

'aqui se deselecta las celdas de datagridview

cargarTab1()

End If

End If

Else

MsgBox("Por favor seleccione una Gafa . ", 64, "Gestion Gafas")

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento textChange del campo txtBusID se busca por caracter

'se usa el metodo buscarporCaracter() que recibe dos parametros la caracter y nombre de la columna correspondiente en la tabla

Private Sub txtBusID\_TextChanged(sender As Object, e As EventArgs) Handles txtBusID.TextChanged

Try

gestion.buscarporCaracter(txtBusID.Text, "Gafas", " id", "Gafas")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView2.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub txtBusISBN\_TextChanged(sender As Object, e As EventArgs) Handles txtBusISBN.TextChanged

Try

gestion.buscarporCaracter(txtBusISBN.Text, "Gafas", " isbn", "Gafas")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView2.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub txtBusNombre\_TextChanged(sender As Object, e As EventArgs) Handles txtBusNombre.TextChanged

Try

gestion.buscarporCaracter(txtBusNombre.Text, "Gafas", " nombre", "Gafas")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView2.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub cbxBusColor\_SelectedIndexChanged(sender As Object, e As EventArgs) Handles cbxBusColor.SelectedIndexChanged

Try

gestion.buscarporCaracter(cbxBusColor.Text, "Gafas", " color", "Gafas")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView2.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub cbxBusForma\_SelectedIndexChanged(sender As Object, e As EventArgs) Handles cbxBusForma.SelectedIndexChanged

Try

gestion.buscarporCaracter(cbxBusForma.Text, "Gafas", " forma", "Gafas")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView2.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnLimpiar\_Click(sender As Object, e As EventArgs) Handles btnLimpiarTab2.Click

'limpiar campos campos de busqueda de la tab2

txtBusID.Clear()

txtBusISBN.Clear()

txtBusNombre.Clear()

cbxBusColor.Text = Nothing

cbxBusForma.Text = Nothing

End Sub

Private Sub btnLimpiarTab1\_Click(sender As Object, e As EventArgs) Handles btnLimpiarTab1.Click

limpiarPantalla()

End Sub

Private Sub txtID\_TextChanged(sender As Object, e As EventArgs) Handles txtID.TextChanged

' Si se recibe 0 se está validando contraseña en inicio de sesión. (se permiten 6 dígitos)

' Si se recibe 1 se está validando teléfono. (se permite 9 dígitos).

' Si se recibe 2 se está validando precios en productos

' Si recibe 3 es Id (cliente, libro, producto es igual).

Try

If (validDatos.validarCodigo(txtID.Text, 3)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("ID"), 64, "ID")

txtID.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub GestionDeLibrosToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeLibrosToolStripMenuItem1.Click

Me.Close()

GestionLibro.Show()

End Sub

Private Sub GestionDeUsuariosToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeUsuariosToolStripMenuItem1.Click

Me.Close()

GestionUsuarios.Show()

End Sub

Private Sub GestionDeClientesToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeClientesToolStripMenuItem1.Click

Me.Close()

GestionClienteSocio.Show()

End Sub

Private Sub GestionDeStockToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestionDeStockToolStripMenuItem.Click

Me.Close()

GestionStock.Show()

End Sub

Private Sub GestionEmpleadosToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestionEmpleadosToolStripMenuItem.Click

Me.Close()

GestionEmpleados.Show()

End Sub

Private Sub GestionDeBusquedaToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles BúsquedaToolStripMenuItem.Click

Me.Close()

GestionBusqueda.Show()

End Sub

Private Sub EliminarToolStripButton\_Click(sender As Object, e As EventArgs) Handles EliminarToolStripButton.Click

Try

'esa condicion nos permite assegurar que una celda de la fila en datagridview es seleccionda para poder eleminar la fila

' y una vez seleccionda se recupera el id de la tabla para poder eliminar la fila

If DataGridView1.SelectedCells.Count > 0 Then

Dim valor As Integer = MsgBox("¿Esta seguro de que desea Elminar Libro?", 36, "Gestion Gafas")

If valor = MsgBoxResult.Yes Then

' MsgBox(getIdFila)

If gestion.eliminarFila(getIdFila, " Gafas ", " id ") Then

MsgBox("Gafa Eliminada con éxito . ", 64, "Gestion Gafas")

limpiarPantalla()

cargarTab1()

getNumeroRegistros()

Else

MsgBox("Error gafa no Eliminada . ", MsgBoxStyle.Exclamation, "Gestion Gafas")

'aqui se deselecta las celdas de datagridview

cargarTab1()

End If

End If

Else

MsgBox("Por favor seleccione una Gafa . ", 64, "Gestion Gafas")

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub MenúPrincipalToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles MenúPrincipalToolStripMenuItem.Click

Me.Close()

PantallaPrincipal.Show()

End Sub

Private Sub txtStock\_TextChanged(sender As Object, e As EventArgs) Handles txtStock.TextChanged

' Si se recibe 0 se está validando contraseña en inicio de sesión. (se permiten 6 dígitos)

' Si se recibe 1 se está validando teléfono. (se permite 9 dígitos).

' Si se recibe 2 se está validando precios en productos

' Si recibe 3 es Id (cliente y libro es igual).

' si recibe 4 es para validar stock.

Try

If (validDatos.validarCodigo(txtStock.Text, 4)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Stock"), 64, "Stock")

txtStock.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub txtPrecio\_TextChanged(sender As Object, e As EventArgs) Handles txtPrecio.TextChanged

' Si se recibe 0 se está validando contraseña en inicio de sesión. (se permiten 6 dígitos)

' Si se recibe 1 se está validando teléfono. (se permite 9 dígitos).

' Si se recibe 2 se está validando precios en productos

' Si recibe 3 es Id (cliente, libro, producto es igual).

Try

If (validDatos.validarCodigo(txtPrecio.Text, 2)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Precio"), 64, "Precio")

txtPrecio.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

End Class

**GestionStock.vb**

Public Class GestionStock

'aqui se crea un objeto de la clase GestionConexion de la liberia

Dim queryAdaptador As String = "SELECT id,isbn,titulo,stock FROM Libros"

Dim datasetTableName As String = "Libros"

Dim gestion As New LibConexion.GestionConexion(queryAdaptador, datasetTableName)

Public datatable As New DataTable

Public getIdFila As String

Private Sub GestionStock\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

' releacionar los campos de la tabla con los textbox se usando DataBindings

Try

txtStock.DataBindings.Add("text", gestion.getDataSet, "Libros.stock")

'cargar en el datagridview

'donde sacar los datos pues de la memoria cache dataset

cargarPantalla()

'metodo para mostrar donde esta cursor y en datagridview del total de la filas

getNumeroRegistros()

'añadir la fecha en status label

ToolStripStatusLabel1.Text = DateTime.Now.ToLongDateString.ToString

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

' Llamamos al método para ver qué botones de la barra de herramientas están disponibles para

' este usuario.

PantallaPrincipal.darPermisos()

End Sub

'metodo para Actualizar y cargar los datos en data grid view

Private Sub cargarPantalla()

Try

gestion.getDataSet.Clear()

gestion.getAdapter.Fill(gestion.getDataSet, "Libros")

'aqui se carga el DatagridView

DataGridView1.DataSource = gestion.getDataSet

DataGridView1.DataMember = "Libros"

DataGridView1.ClearSelection()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'Metodo para limpiar los campos del formulario

Private Sub llimpiarPantalla()

txtBusID.Clear()

txtBusISBN.Clear()

txtBusTitulo.Clear()

' txtStock.Clear()

End Sub

'metodo para obtener elemento seleccionado del datagridview para poder programar el metodo Eliminar sin necesitar al biding

Private Sub DataGridView1\_CellClick(sender As Object, e As DataGridViewCellEventArgs) Handles DataGridView1.CellClick

Try

'aqui se declara una var de tipo entero que va guardar el indice de la fila seleccionada usando evento e.RowIndex

Dim indice As Integer

indice = e.RowIndex

'aqui se crea un objeto de la clase DataGridViewRow que nos va permitir obtener el valor

Dim selectedFila As DataGridViewRow

'aqui se obtiene la fila pasandole el indice

'aqui se obtiene la fila pasandole el indice

Try

selectedFila = DataGridView1.Rows(indice)

getIdFila = selectedFila.Cells(0).Value.ToString

Catch ex As Exception

MsgBox("Por favor selecciona una fila valida")

End Try

'MsgBox(getIdFila)

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento button Modificar stock para modificar el campo stock de la tabla libros

Private Sub btnModificarStock\_Click(sender As Object, e As EventArgs) Handles btnModificarStock.Click

Try

'esa condicion nos permite assegurar que una celda de la fila en datagridview es seleccionda para poder eleminar la fila

' y una vez seleccionda se recupera el id de la tabla para poder modifcar la fila

If DataGridView1.SelectedCells.Count > 0 Then

Dim valor As Integer = MsgBox("¿Esta seguro de que desea modificar el stock ?", 36, "Gestion Stock")

If valor = MsgBoxResult.Yes Then

' MsgBox(getIdFila)

Timer1.Enabled = True

If gestion.modificarStock(getIdFila, CInt(txtStock.Text), " Libros ", " id ") Then

MsgBox("Stock Modificao con éxito . ", 64, "Gestion Stock")

cargarPantalla()

Else

MsgBox("Error Stock no modificado . ", MsgBoxStyle.Exclamation, "Gestion Stock")

'aqui se deselecta las celdas de datagridview

cargarPantalla()

End If

End If

Else

MsgBox("Por favor seleccione el Stock . ", 64, "Gestion Stock")

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento textChange del campo txtBusID se busca por caracter

'se usa el metodo buscarporCaracter() que recibe dos parametros la caracter y nombre de la columna correspondiente en la tabla

Private Sub txtBusID\_TextChanged(sender As Object, e As EventArgs) Handles txtBusID.TextChanged

Try

gestion.buscarStock(txtBusID.Text, "Libros", " id", "Libros")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView1.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento textChange del campo txtBusISBN se busca por caracter

'se usa el metodo buscarporCaracter() que recibe dos parametros la caracter y nombre de la columna correspondiente en la tabla

Private Sub txtBusISBN\_TextChanged(sender As Object, e As EventArgs) Handles txtBusISBN.TextChanged

Try

gestion.buscarStock(txtBusISBN.Text, "Libros", " isbn", "Libros")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView1.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento textChange del campo txtBusTitulo se busca por caracter

'se usa el metodo buscarporCaracter() que recibe dos parametros la caracter y nombre de la columna correspondiente en la tabla

Private Sub txtBusTitulo\_TextChanged(sender As Object, e As EventArgs) Handles txtBusTitulo.TextChanged

Try

gestion.buscarStock(txtBusTitulo.Text, "Libros", " titulo", "Libros")

'aqui se recupera midataset desde metodo get de la clase gestionConexion

DataGridView1.DataSource = gestion.getDataSet

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'metodo para mostrar donde esta cursor y en datagridview del total de la filas

Private Sub getNumeroRegistros()

Try

lblNumero.Text = (BindingContext(gestion.getDataSet, "Libros").Position + 1).ToString + " de " +

(BindingContext(gestion.getDataSet, "Libros").Count).ToString

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnUltimo\_Click(sender As Object, e As EventArgs) Handles btnUltimo.Click

'BindingContext(midataset, "Tabla").Position = DataGridView1.RowCount - 1

Try

BindingContext(gestion.getDataSet, "Libros").Position = BindingContext(gestion.getDataSet, "Libros").Count - 1

btnUltimo.Enabled = False

btnSiguiete.Enabled = False

btnPrimero.Enabled = True

btnAnterior.Enabled = True

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnSiguiete\_Click(sender As Object, e As EventArgs) Handles btnSiguiete.Click

Try

'colacrtese con bindingContext en la dataset dandlole nombre en la posicion

BindingContext(gestion.getDataSet, "Libros").Position += 1

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnAnterior\_Click(sender As Object, e As EventArgs) Handles btnAnterior.Click

'colacrtese con bindingContext en la dataset dandlole nombre en la posicion

Try

BindingContext(gestion.getDataSet, "Libros").Position -= 1

btnSiguiete.Enabled = True

btnUltimo.Enabled = True

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnPrimero\_Click(sender As Object, e As EventArgs) Handles btnPrimero.Click

Try

BindingContext(gestion.getDataSet, "Libros").Position = 0

btnPrimero.Enabled = False

btnAnterior.Enabled = False

btnUltimo.Enabled = True

btnSiguiete.Enabled = True

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'metodo timer

Private Sub Timer1\_Tick(sender As Object, e As EventArgs) Handles Timer1.Tick

Try

If ToolStripProgressBar1.Value <= ToolStripProgressBar1.Maximum - 1 Then

ToolStripProgressBar1.Value = ToolStripProgressBar1.Value + 50

Else

Timer1.Enabled = False

ToolStripProgressBar1.Value = 0

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub GestionDeLibrosToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeLibrosToolStripMenuItem1.Click

Me.Close()

GestionLibro.Show()

End Sub

Private Sub GestionDeBusquedaToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestionDeBusquedaToolStripMenuItem.Click

Me.Close()

GestionBusqueda.Show()

End Sub

Private Sub GestionEmpleadosToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestionEmpleadosToolStripMenuItem.Click

Me.Close()

GestionEmpleados.Show()

End Sub

Private Sub GestionDeUsuariosToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeUsuariosToolStripMenuItem1.Click

Me.Close()

GestionUsuarios.Show()

End Sub

Private Sub GestionDeClientesToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeClientesToolStripMenuItem1.Click

Me.Close()

GestionClienteSocio.Show()

End Sub

Private Sub MenúPrincipaToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles MenúPrincipaToolStripMenuItem.Click

Me.Close()

PantallaPrincipal.Show()

End Sub

'Metodo de evento cerrando fourmulario

Private Sub Form1\_Closing(Sender As Object, e As FormClosingEventArgs) Handles Me.FormClosing

PantallaPrincipal.Show()

Me.Dispose()

End Sub

Private Sub TrackBar1\_Scroll(sender As Object, e As EventArgs) Handles TrackBar1.Scroll

txtStock.Text = TrackBar1.Value.ToString

End Sub

Private Sub GestiónProductosToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestiónProductosToolStripMenuItem.Click

Me.Close()

GestionProductos.Show()

End Sub

Private Sub btnLimpiar\_Click(sender As Object, e As EventArgs) Handles btnLimpiar.Click

limpiarPantalla()

End Sub

'metodo limpiarPantalla() permite limpiar los campos de buesqueda

Private Sub limpiarPantalla()

txtBusID.Clear()

txtBusISBN.Clear()

txtBusTitulo.Clear()

End Sub

End Class

**GestionUsuarios.vb**

Public Class GestionUsuarios

'aqui se crea un objeto de la clase GestionConexion de la liberia

Dim queryAdaptador As String = "SELECT \* FROM Usuarios"

Dim datasetTableName As String = "Usuarios"

Dim queryRelacionEmpUsr As String = "Select Empleados.id, Empleados.nombre, Empleados.apellidos, Usuarios.id, Usuarios.usuario, Usuarios.clave, Usuarios.rol from Empleados,Usuarios where Empleados.id = Usuarios.fk\_empleado"

Dim queryUsr As String = "SELECT \* FROM Usuarios"

Dim queryEmp As String = "SELECT \* FROM Empleados"

Dim gestion As New LibConexion.GestionConexion(queryAdaptador, datasetTableName)

Public datatable As New DataTable

Public getIdFila As String

Public fkEmpleado As String

Public getUsuario As String

Public dataset As DataSet

' Variables para validar datos vvvv

Dim validDatos As New libValidacionDatos.Validacion

Private Sub GestionUsuarios\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

Try

'aqui se recupera la tabla Empleados del dataset que se returna el metodo getEmpleados

' Cargamos el combobox con el contenido del campo

Dim datasetEmp = gestion.getEmpleados("select id from Empleados", "Empleados")

cbxEmpleados.DataSource = datasetEmp.Tables("Empleados")

'aqui se visualiza solamente la columna id de la tabla empleados usando metodo DisplayMember

cbxEmpleados.DisplayMember = "id"

'cargar en el datagridview

'donde sacar los datos pues de la memoria cache dataset

dataset = gestion.getRelacionEmpUsr(queryRelacionEmpUsr, "relacion", queryEmp, queryUsr, "Empleados", "Usuarios")

DataGridView1.DataSource = dataset

DataGridView1.DataMember = "relacion"

dataset.Relations.Add("relacion", dataset.Tables("Empleados").Columns("id"), dataset.Tables("Usuarios").Columns("fk\_empleado"))

'releacionar los campos de la tabla con los textbox se usando DataBindings

txtID.DataBindings.Add("text", dataset, "Usuarios.id")

txtUsuario.DataBindings.Add("text", dataset, "Usuarios.usuario")

txtClave.DataBindings.Add("text", dataset, "Usuarios.Clave")

cbxRol.DataBindings.Add("text", dataset, "Usuarios.Rol")

cbxEmpleados.DataBindings.Add("text", dataset, "Empleados.id")

cargarPantalla()

'añadir la fecha en status label

ToolStripStatusLabel1.Text = DateTime.Now.ToLongDateString.ToString

'metodo para mostrar donde esta cursor y en datagridview del total de la filas

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

' Comprobarmos permisos

If PantallaPrincipal.lbtxtRol.Text.Equals("Usuario") Then

GestionBusqueda.GestionDeUsuariosToolStripMenuItem1.Enabled = False

GestionBusqueda.GestionEmpleadosToolStripMenuItem.Enabled = False

GestionLibro.GestionDeEmpleadosToolStripMenuItem.Enabled = False

GestionLibro.GestionDeUsuariosToolStripMenuItem1.Enabled = False

GestionClienteSocio.GestionDeUsuariosToolStripMenuItem1.Enabled = False

GestionClienteSocio.GestionDeEmpleadosToolStripMenuItem1.Enabled = False

GestionProductos.GestionDeUsuariosToolStripMenuItem1.Enabled = False

GestionProductos.GestionEmpleadosToolStripMenuItem.Enabled = False

GestionStock.GestionEmpleadosToolStripMenuItem.Enabled = False

GestionStock.GestionDeUsuariosToolStripMenuItem1.Enabled = False

' Se entra como Guest /invitado.

ElseIf PantallaPrincipal.lbtxtRol.Text.Equals("Guest") Then

GestionBusqueda.GestionDeClientesToolStripMenuItem1.Enabled = False

GestionBusqueda.GestionEmpleadosToolStripMenuItem.Enabled = False

GestionBusqueda.GestionDeUsuariosToolStripMenuItem1.Enabled = False

GestionBusqueda.GestiónProductosToolStripMenuItem.Enabled = False

GestionBusqueda.GestionDeLibrosToolStripMenuItem1.Enabled = False

GestionBusqueda.GestionDeStockToolStripMenuItem.Enabled = False

' Guardamos la hora de acceso del usuario (con sus datos):

End If

End Sub

'metodo para dar alta usaurio

Private Sub btnAlta\_Click(sender As Object, e As EventArgs) Handles btnAlta.Click

Try

Timer1.Enabled = True

If (txtID.Text = "" Or txtUsuario.Text = "" Or txtClave.Text = "" Or cbxRol.Text = "" Or cbxEmpleados.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Usuarios")

Else

If gestion.insertUsuario(CInt(txtID.Text), txtUsuario.Text, CInt(txtClave.Text), cbxRol.Text, CInt(cbxEmpleados.Text)) Then

cargarPantalla()

MsgBox("Usario Insertado con éxito . ", 64, "Gestion Usuarios")

llimpiarPantalla()

cargarPantalla()

getNumeroRegistros()

Else

cargarPantalla()

MsgBox("Error Usario no Insertado . ", MsgBoxStyle.Exclamation, "Gestion Usuarios")

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'Metodo para limpiar los campos del formulario

Private Sub llimpiarPantalla()

txtID.Clear()

txtClave.Clear()

txtUsuario.Clear()

cbxEmpleados.Text = Nothing

cbxRol.Text = Nothing

End Sub

'metodo para Actualizar y cargar los datos en data grid view

Private Sub cargarPantalla()

Try

'aqui se carga el DatagridView

dataset = gestion.getRelacionEmpUsr(queryRelacionEmpUsr, "relacion", queryEmp, queryUsr, "Empleados", "Usuarios")

DataGridView1.DataSource = dataset

DataGridView1.DataMember = "relacion"

dataset.Relations.Add("relacion", dataset.Tables("Empleados").Columns("id"), dataset.Tables("Usuarios").Columns("fk\_empleado"))

DataGridView1.ClearSelection()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'metodo para obtener elemento seleccionado del datagridview para poder programar el metodo Eliminar sin necesitar al biding

Private Sub DataGridView1\_CellClick(sender As Object, e As DataGridViewCellEventArgs) Handles DataGridView1.CellClick

Try

'aqui se declara una var de tipo entero que va guardar el indice de la fila seleccionada usando evento e.RowIndex

Dim indice As Integer

indice = e.RowIndex

'aqui se crea un objeto de la clase DataGridViewRow que nos va permitir obtener el valor

Dim selectedFila As DataGridViewRow

'aqui se obtiene la fila pasandole el indice

Try

selectedFila = DataGridView1.Rows(indice)

getIdFila = selectedFila.Cells(3).Value.ToString

fkEmpleado = selectedFila.Cells(0).Value.ToString

getUsuario = selectedFila.Cells(4).Value.ToString

Catch ex As Exception

MsgBox("Por favor selecciona una fila valida")

End Try

'MsgBox(getIdFila)

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnLimpiar\_Click(sender As Object, e As EventArgs) Handles btnLimpiar.Click

llimpiarPantalla()

End Sub

' En el evento button Modificar para modifica una fila

Private Sub btnModificar\_Click(sender As Object, e As EventArgs) Handles btnModificar.Click

'esa condicion nos permite assegurar que una celda de la fila en datagridview es seleccionda para poder eleminar la fila

' y una vez seleccionda se recupera el id de la tabla para poder modifcar la fila

Try

If (txtID.Text = "" Or txtUsuario.Text = "" Or txtClave.Text = "" Or cbxRol.Text = "" Or cbxEmpleados.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Usuarios")

Else

If DataGridView1.SelectedCells.Count > 0 Then

Dim valor As Integer = MsgBox("¿Esta seguro de que desea Modificar Empleado?", 36, "Gestion Empleados")

If valor = MsgBoxResult.Yes Then

'MsgBox(getIdFila)

Timer1.Enabled = True

If gestion.modificarUsuario(CInt(txtID.Text), txtUsuario.Text, CInt(txtClave.Text), cbxRol.Text, CInt(cbxEmpleados.Text), " Usuarios ", " id ") Then

MsgBox("Usuario Modificao con éxito . ", 64, "Gestion Usuarios")

cargarPantalla()

Else

MsgBox("Error Usuario no modificado . ", MsgBoxStyle.Exclamation, "Gestion Usuarios")

'aqui se deselecta las celdas de datagridview

cargarPantalla()

End If

End If

Else

MsgBox("Por favor seleccione un Usuario . ", 64, "Gestion Usuario")

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'En el evento button Eliminar para eliminar una fila

Private Sub Button6\_Click(sender As Object, e As EventArgs) Handles btnEliminar.Click

'esa condicion nos permite assegurar que una celda de la fila en datagridview es seleccionda para poder eleminar la fila

' y una vez seleccionda se recupera el id de la tabla para poder eliminar la fila

Try

If DataGridView1.SelectedCells.Count > 0 Then

Dim valor As Integer = MsgBox("¿Esta seguro de que desea Elminar Empleado?", 36, "Gestion Usuarios")

If valor = MsgBoxResult.Yes Then

' MsgBox(getIdFila)

Timer1.Enabled = True

If gestion.eliminarFila(getIdFila, " Usuarios ", " id ") Then

MsgBox("Usuario Eliminado con éxito . ", 64, "Gestion Usuarios")

llimpiarPantalla()

cargarPantalla()

getNumeroRegistros()

Else

MsgBox("Error Usuario no Eliminado . ", MsgBoxStyle.Exclamation, "Gestion Usuarios")

'aqui se deselecta las celdas de datagridview

cargarPantalla()

End If

End If

Else

MsgBox("Por favor seleccione un Usuario . ", 64, "Gestion Usuarios")

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Validar campo ID

Private Sub txtID\_TextChanged(sender As Object, e As EventArgs) Handles txtID.TextChanged

' Si se recibe 0 se está validando contraseña en inicio de sesión. (se permiten 6 dígitos)

' Si se recibe 1 se está validando teléfono. (se permite 9 dígitos).

' Si se recibe 2 se está validando precios en productos

' Si recibe 3 es Id de cliente.

Try

If (validDatos.validarCodigo(txtID.Text, 3)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("ID"), 64, "Datos")

txtID.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Validar campo usuario

Private Sub txtUsuario\_TextChanged(sender As Object, e As EventArgs) Handles txtUsuario.TextChanged

Try

If (validDatos.validarUsuario(txtUsuario.Text, 0)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Usuario"), 64, "Datos")

txtUsuario.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Validar campo clave

Private Sub txtClave\_TextChanged(sender As Object, e As EventArgs) Handles txtClave.TextChanged

Try

If (validDatos.validarCodigo(txtClave.Text, 0)) Then

Else

MsgBox(validDatos.mensajeInfoDatoInvalido("Clave"), 64, "Datos")

txtClave.Clear()

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'get los valors de la filas seleccionadas en datagridview usando evento DataGridView1.Click

Private Sub DataGridView1\_Click(sender As Object, e As EventArgs) Handles DataGridView1.Click

Try

'aqui se recupera los valores pasando al datagridview las celdas que se necesita

txtID.Text = DataGridView1.CurrentRow.Cells(3).Value.ToString

txtUsuario.Text = DataGridView1.CurrentRow.Cells(4).Value.ToString

txtClave.Text = DataGridView1.CurrentRow.Cells(5).Value.ToString

cbxRol.Text = DataGridView1.CurrentRow.Cells(6).Value.ToString

cbxEmpleados.Text = DataGridView1.CurrentRow.Cells(0).Value.ToString

Catch ex As Exception

MsgBox("Porfavor seleccione una fila valida " & Chr(13), 64, "Gestion Usuarios")

' gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'metodo para mostrar donde esta cursor y en datagridview del total de la filas

Private Sub getNumeroRegistros()

Try

lblNumero.Text = (BindingContext(dataset, "relacion").Position + 1).ToString + " de " +

(BindingContext(dataset, "relacion").Count).ToString

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnUltimo\_Click(sender As Object, e As EventArgs) Handles btnUltimo.Click

'BindingContext(midataset, "Tabla").Position = DataGridView1.RowCount - 1

Try

BindingContext(dataset, "relacion").Position = BindingContext(dataset, "relacion").Count - 1

btnUltimo.Enabled = False

btnSiguiete.Enabled = False

btnPrimero.Enabled = True

btnAnterior.Enabled = True

'aqui se recupera los valores pasando al datagridview las celdas que se necesita

txtID.Text = DataGridView1.CurrentRow.Cells(3).Value.ToString

txtUsuario.Text = DataGridView1.CurrentRow.Cells(4).Value.ToString

txtClave.Text = DataGridView1.CurrentRow.Cells(5).Value.ToString

cbxRol.Text = DataGridView1.CurrentRow.Cells(6).Value.ToString

cbxEmpleados.Text = DataGridView1.CurrentRow.Cells(0).Value.ToString

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnSiguiete\_Click(sender As Object, e As EventArgs) Handles btnSiguiete.Click

'colacrtese con bindingContext en la dataset dandlole nombre en la posicion

Try

BindingContext(dataset, "relacion").Position += 1

'aqui se recupera los valores pasando al datagridview las celdas que se necesita

txtID.Text = DataGridView1.CurrentRow.Cells(3).Value.ToString

txtUsuario.Text = DataGridView1.CurrentRow.Cells(4).Value.ToString

txtClave.Text = DataGridView1.CurrentRow.Cells(5).Value.ToString

cbxRol.Text = DataGridView1.CurrentRow.Cells(6).Value.ToString

cbxEmpleados.Text = DataGridView1.CurrentRow.Cells(0).Value.ToString

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnAnterior\_Click(sender As Object, e As EventArgs) Handles btnAnterior.Click

'colacrtese con bindingContext en la dataset dandlole nombre en la posicion

Try

BindingContext(dataset, "relacion").Position -= 1

btnSiguiete.Enabled = True

btnUltimo.Enabled = True

'aqui se recupera los valores pasando al datagridview las celdas que se necesita

txtID.Text = DataGridView1.CurrentRow.Cells(3).Value.ToString

txtUsuario.Text = DataGridView1.CurrentRow.Cells(4).Value.ToString

txtClave.Text = DataGridView1.CurrentRow.Cells(5).Value.ToString

cbxRol.Text = DataGridView1.CurrentRow.Cells(6).Value.ToString

cbxEmpleados.Text = DataGridView1.CurrentRow.Cells(0).Value.ToString

'llamar al metodo

getNumeroRegistros()

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnPrimero\_Click(sender As Object, e As EventArgs) Handles btnPrimero.Click

Try

BindingContext(dataset, "relacion").Position = 0

btnPrimero.Enabled = False

btnAnterior.Enabled = False

btnUltimo.Enabled = True

btnSiguiete.Enabled = True

'llamar al metodo

getNumeroRegistros()

'aqui se recupera los valores pasando al datagridview las celdas que se necesita

txtID.Text = DataGridView1.CurrentRow.Cells(3).Value.ToString

txtUsuario.Text = DataGridView1.CurrentRow.Cells(4).Value.ToString

txtClave.Text = DataGridView1.CurrentRow.Cells(5).Value.ToString

cbxRol.Text = DataGridView1.CurrentRow.Cells(6).Value.ToString

cbxEmpleados.Text = DataGridView1.CurrentRow.Cells(0).Value.ToString

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

'metodo timer

Private Sub Timer1\_Tick(sender As Object, e As EventArgs) Handles Timer1.Tick

Try

If ToolStripProgressBar1.Value <= ToolStripProgressBar1.Maximum - 1 Then

ToolStripProgressBar1.Value = ToolStripProgressBar1.Value + 50

Else

Timer1.Enabled = False

ToolStripProgressBar1.Value = 0

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub GestionDeLibrosToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeLibrosToolStripMenuItem1.Click

Me.Close()

GestionLibro.Show()

End Sub

Private Sub GestionDeBusquedaToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestionDeBusquedaToolStripMenuItem.Click

Me.Close()

GestionBusqueda.Show()

End Sub

Private Sub GestionDeStockToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestionDeStockToolStripMenuItem.Click

Me.Close()

GestionStock.Show()

End Sub

Private Sub GestionDeEmpleadosToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestionDeEmpleadosToolStripMenuItem.Click

Me.Close()

GestionEmpleados.Show()

End Sub

Private Sub GestionDeClientesToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestionDeClientesToolStripMenuItem1.Click

Me.Close()

GestionClienteSocio.Show()

End Sub

Private Sub MenúPrincipaToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles MenúPrincipaToolStripMenuItem.Click

Me.Close()

PantallaPrincipal.Show()

End Sub

Private Sub toolsbtnNuevo\_Click(sender As Object, e As EventArgs) Handles toolsbtnNuevo.Click

llimpiarPantalla()

End Sub

' Botón de toolStrip guardar

' Misma funcion que alta

Private Sub toolsbtnGuardar\_Click(sender As Object, e As EventArgs) Handles toolsbtnGuardar.Click

Try

Timer1.Enabled = True

If (txtID.Text = "" Or txtUsuario.Text = "" Or txtClave.Text = "" Or cbxRol.Text = "" Or cbxEmpleados.Text = "") Then

MsgBox("Por favor rellene los campos . ", 64, "Gestion Usuarios")

Else

If gestion.insertUsuario(CInt(txtID.Text), txtUsuario.Text, CInt(txtClave.Text), cbxRol.Text, CInt(cbxEmpleados.Text)) Then

cargarPantalla()

MsgBox("Usario Insertado con éxito . ", 64, "Gestion Usuarios")

llimpiarPantalla()

cargarPantalla()

getNumeroRegistros()

Else

cargarPantalla()

MsgBox("Error Usario no Insertado . ", MsgBoxStyle.Exclamation, "Gestion Usuarios")

End If

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

' Botón toolStrip eliminar

' misma función que el botón eliminar

Private Sub toolsbtnEliminar\_Click(sender As Object, e As EventArgs) Handles toolsbtnEliminar.Click

'esa condicion nos permite assegurar que una celda de la fila en datagridview es seleccionda para poder eleminar la fila

' y una vez seleccionda se recupera el id de la tabla para poder eliminar la fila

Try

If DataGridView1.SelectedCells.Count > 0 Then

Dim valor As Integer = MsgBox("¿Esta seguro de que desea Elminar Empleado?", 36, "Gestion Usuarios")

If valor = MsgBoxResult.Yes Then

' MsgBox(getIdFila)

Timer1.Enabled = True

If gestion.eliminarFila(getIdFila, " Usuarios ", " id ") Then

MsgBox("Usuario Eliminado con éxito . ", 64, "Gestion Usuarios")

llimpiarPantalla()

cargarPantalla()

getNumeroRegistros()

Else

MsgBox("Error Usuario no Eliminado . ", MsgBoxStyle.Exclamation, "Gestion Usuarios")

'aqui se deselecta las celdas de datagridview

cargarPantalla()

End If

End If

Else

MsgBox("Por favor seleccione un Usuario . ", 64, "Gestion Usuarios")

End If

Catch ex As Exception

gestion.mensajeErrorDatos()

gestion.errorLogWrite()

End Try

End Sub

Private Sub btnVolverMenu\_Click(sender As Object, e As EventArgs) Handles btnVolverMenu.Click

Me.Close()

PantallaPrincipal.Show()

End Sub

'Metodo de evento cerrando fourmulario

Private Sub Form1\_Closing(Sender As Object, e As FormClosingEventArgs) Handles Me.FormClosing

PantallaPrincipal.Show()

Me.Dispose()

End Sub

Private Sub GestiónProductosToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestiónProductosToolStripMenuItem.Click

Me.Close()

GestionProductos.Show()

End Sub

End Class