## Planeación.

## Objetivos

El objetivo del proyecto es crear un programa en cualquier lenguaje de programación que permita gestionar los datos de un cliente y asignarles una categoría.

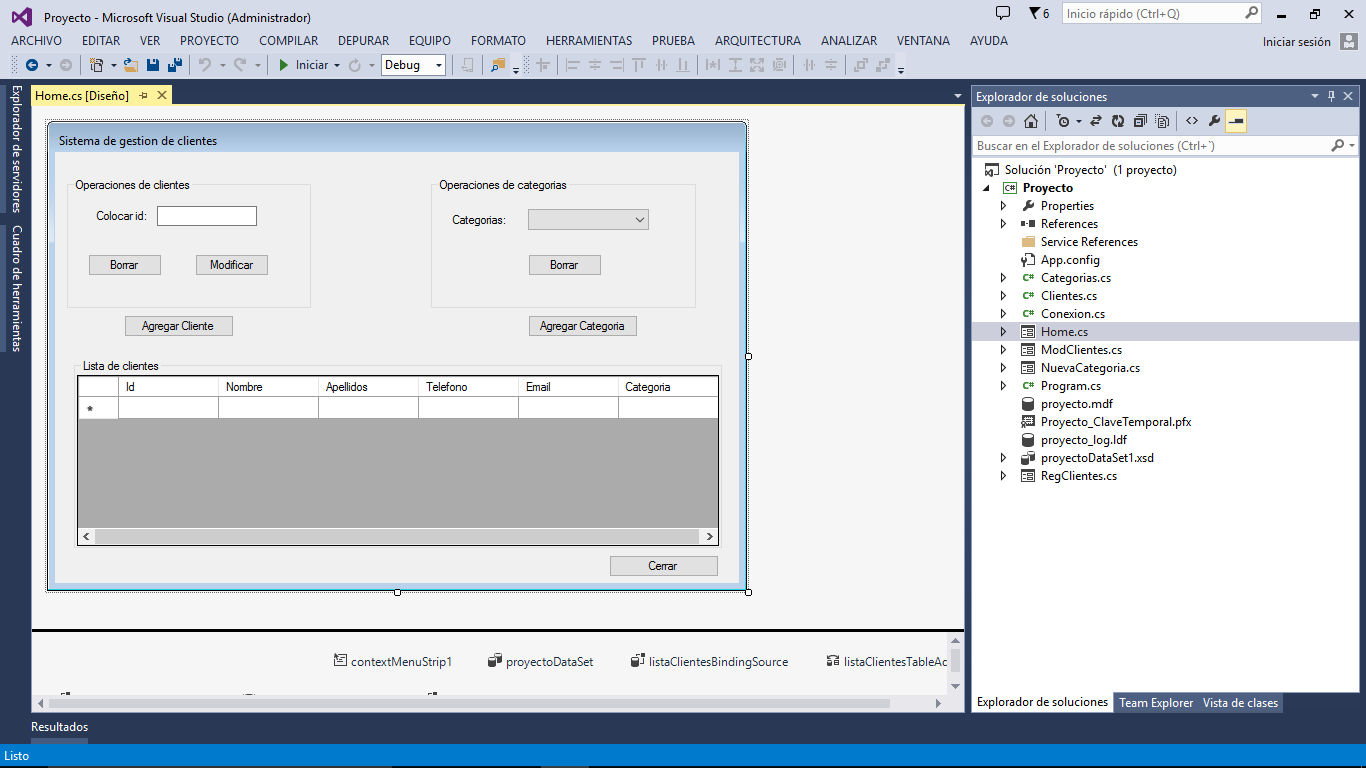
## Alcance

Esta aplicación permitirá visualizar una lista de usuarios agregados a la base de datos así mismo permitirá agregar, eliminar y modificar los usuarios ya registrados previamente.

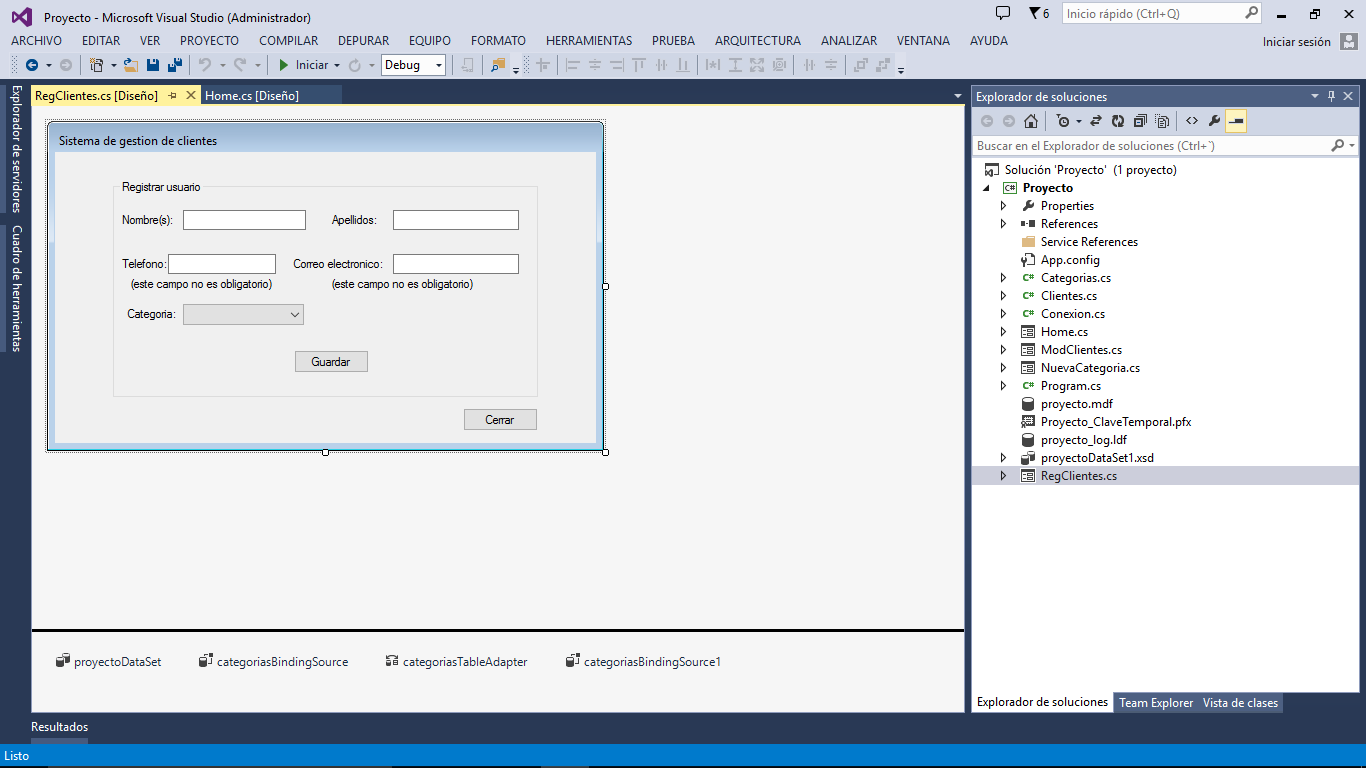
## Requerimientos

* Determinar si el correo electrónico ingresado valido
* Determinar si los valore de los campos son validos
* No permitir ingresar nombre y apellidos nulos
* No permitir que el campo de categorías este vacío
* Los correos electrónicos no deben repetirse

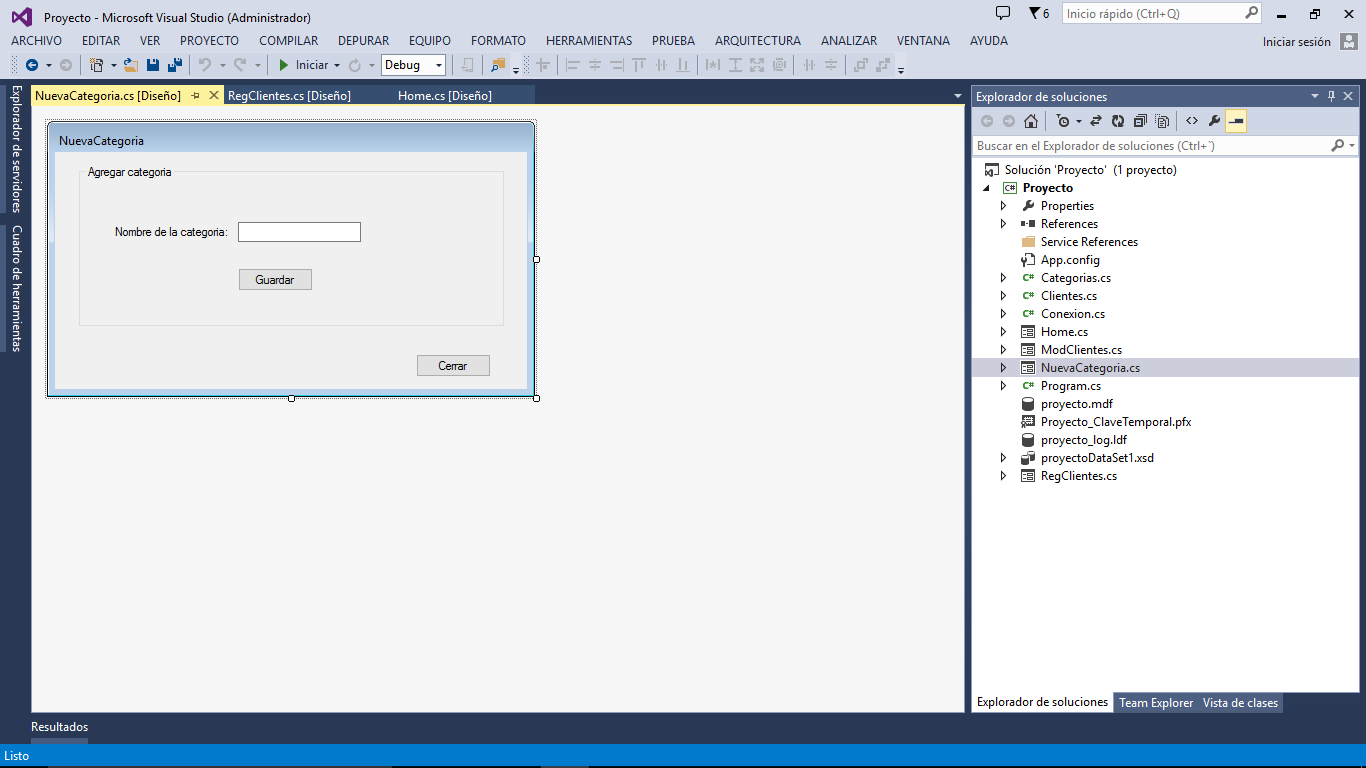
# Diseño



Interfaz de inicio



Ventana de registro de clientes



Ventana de agregación de categorías

**Convención de nombres**

Los nombres de los archivos en generados por Visual Studio se mantendrán en su forma predeterminada, pero cuando se cree algún otro archivo la primera letra deberá ser mayúscula y el resto deberá ser minúsculas.

**Diseño de archivos**

Las carpetas necesarias ya son creadas por Visual Studio y los archivos que estarán dentro serán compuestos por una palabra y el resto minúsculas, en caso de que se requiera usar más palabras se colocara la primera palabra en minúsculas, y la inicial de las siguientes palabras serán mayúsculas.

**Diseño de clases**

Las clases serán colocadas en un archivo.cs que es la extensión que les genera Visual Studio y dentro se almacenará el código correspondiente. El nombre del archivo será el mismo del de la clase y siempre empezará en mayúscula.

Las clases declaradas la primera letra de la palabra será mayúscula y si está compuesta de otras palabras, la primera inicial de cada palabra será mayúscula.

**Diseño de funciones**

El nombre de los métodos se declarara sin espacios en caso de tener más de una palabra, se colocara la primera letra de cada palabra en mayúsculas.

Las funciones no deben devolver más de 1 solo valor.

**Diseño de variables**

Las variables serán minúsculas.

Si la variable contiene más de una palabra será mayúscula la inicial de cada palabra.

**Diseño de las tablas (DB)**

La base de datos tendrá por nombre proyecto y los nombres de las tablas en la base de datos se escribirán de forma plural y los campos se escribirán de forma singular, si es necesario escribir el nombre usando 2 palabras se separara a través de un guion bajo (\_).

**Comentarios**

Los comentarios se harán arriba de cada método.

## Estructuras de control

### If

If(Numero<0)

{

MessageBox(“Numero es menor que 0”);

}

### Switch

La estructura de control switch no será requerida en el proyecto.

### Do

La estructura de control Do no será requerida en el proyecto.

### Do while

La estructura de control Do While no será requerida en el proyecto.

### While

La estructura de control While no será requerida en el proyecto.

### For

for(int x=0 ; x <10 ; x++)

{

}

## Clases

class Categorias

{

}

## Método

public int ObtenerId(string nombreCategoria)

{

}

## Variables

String nombre;

## Constantes

No se utilizaran constantes en el proyecto

## Comentarios

/\*

Establece una conexión con la base de datos y hace la inserción de un cliente

\*/

public void Agregar(String nombre)

{

Conexion objConexion = new Conexion();

String cadena = "Insert into categorias(nombre)" + "values ('" + nombre+"')";

objConexion.Conectar();

objConexion.EjecutarSql(cadena);

objConexion.Desconectar();

}

# Codificación

Clase Categorias:

class Categorias

{

public void Agregar(String nombre)

{

Conexion objConexion = new Conexion();

String cadena = "Insert into categorias(nombre)" + "values ('" + nombre+"')";

objConexion.Conectar();

objConexion.EjecutarSql(cadena);

objConexion.Desconectar();

}

public void Eliminar(int id)

{

Conexion objConexion = new Conexion();

var resultado = MessageBox.Show("¿Desear realmente eliminar este registro?", "Confirmaciòn de borrado", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (resultado == DialogResult.Yes)

{

objConexion.Conectar();

String Cadenasql = "delete from categorias where id= " + id + "; ";

objConexion.EjecutarSql(Cadenasql);

objConexion.Desconectar();

}

}

}

Clase Clientes:

class Clientes

{

public void Agregar(String nombre,String apellidos,String telefono,String email, int categoria) {

Conexion objConexion =new Conexion();

String cadena = "Insert into clientes(nombre,apellidos,telefono,correo\_electronico,categoria\_id)" + "values ('" +

nombre + "','" + apellidos + "','" + telefono + "','" + email + "','" + categoria + "')";

objConexion.Conectar();

objConexion.EjecutarSql(cadena);

objConexion.Desconectar();

}

public Boolean Buscar(int id) {

try{

Conexion objConexion = new Conexion();

int idEncontrado;

String query = "Select id from clientes where id='"+id+"'";

SqlDataAdapter da = new SqlDataAdapter(query, "Data Source=DESKTOP-PNHNJ5R;Initial Catalog=proyecto;Integrated Security=True");

da.SelectCommand.Connection.Open();

idEncontrado = Convert.ToInt32(da.SelectCommand.ExecuteScalar().ToString());

da.SelectCommand.Connection.Close();

if (idEncontrado == id)

{

return true;

}

}

catch (System.NullReferenceException)

{

}

return false;

}

public void Eliminar(int id)

{

Conexion objConexion = new Conexion();

var resultado = MessageBox.Show("¿Desear realmente eliminar este registro?", "Confirmaciòn de borrado", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (resultado == DialogResult.Yes)

{

objConexion.Conectar();

String Cadenasql = "delete from clientes where id= " + id + "; ";

objConexion.EjecutarSql(Cadenasql);

objConexion.Desconectar();

}

}

public string ObtenerNombre(int id)

{

String nombre;

String query = "select nombre from clientes where id='" + id + "'";

SqlDataAdapter da = new SqlDataAdapter(query, "Data Source=DESKTOP-PNHNJ5R;Initial Catalog=proyecto;Integrated Security=True");

da.SelectCommand.Connection.Open();

nombre = da.SelectCommand.ExecuteScalar().ToString();

da.SelectCommand.Connection.Close();

return nombre;

}

public string ObtenerApellidos(int id)

{

String apellidos;

String query = "select apellidos from clientes where id='" + id + "'";

SqlDataAdapter da = new SqlDataAdapter(query, "Data Source=DESKTOP-PNHNJ5R;Initial Catalog=proyecto;Integrated Security=True");

da.SelectCommand.Connection.Open();

apellidos = da.SelectCommand.ExecuteScalar().ToString();

da.SelectCommand.Connection.Close();

return apellidos;

}

public string ObtenerTelefono(int id)

{

String telefono;

String query = "select telefono from clientes where id='" + id + "'";

SqlDataAdapter da = new SqlDataAdapter(query, "Data Source=DESKTOP-PNHNJ5R;Initial Catalog=proyecto;Integrated Security=True");

da.SelectCommand.Connection.Open();

telefono = da.SelectCommand.ExecuteScalar().ToString();

da.SelectCommand.Connection.Close();

return telefono;

}

public string ObtenerEmail(int id)

{

String email;

String query = "select correo\_electronico from clientes where id='" + id + "'";

SqlDataAdapter da = new SqlDataAdapter(query, "Data Source=DESKTOP-PNHNJ5R;Initial Catalog=proyecto;Integrated Security=True");

da.SelectCommand.Connection.Open();

email = da.SelectCommand.ExecuteScalar().ToString();

da.SelectCommand.Connection.Close();

return email;

}

public void Modificar(int id,String nombre,String apellidos,String telefono,String email,int categoria )

{

Conexion objConexion = new Conexion();

objConexion.Conectar();

String Cadenasql = "UPDATE clientes set nombre ='"+ nombre + "',apellidos='" + apellidos + "',telefono='" + telefono + "',correo\_electronico='" + email +"',categoria\_id='"+ categoria+"' WHERE id = " + id + "; ";

objConexion.EjecutarSql(Cadenasql);

objConexion.Desconectar();

}

public Boolean RevisarParametros(String nombre, String apellidos, String telefono, String email)

{

int correoPar1 = email.IndexOf(".");

int correoPar2 = email.IndexOf("@");

Boolean parametros=false;

if(nombre.Equals(""))

{

MessageBox.Show("el campo nombre se encuentra vacio");

return false;

}

if (apellidos.Equals(""))

{

MessageBox.Show("el campo apellidos se encuentra vacio");

return false;

}

if(nombre.Equals("")&&apellidos.Equals(""))

{

MessageBox.Show("No se llenaron todos los campos");

return false;

}

if(parametros==true && email.Equals(""))

{

return true;

}

if (correoPar1 == -1 )

{

MessageBox.Show("El correo que ingreso no es valido");

return false;

}

if (correoPar2 == -1)

{

MessageBox.Show("El correo que ingreso no es valido");

return false;

}

if (correoPar1==-1 && correoPar2==-1)

{

MessageBox.Show("El correo que ingreso no es valido");

return false;

}

else{

parametros = true;

return parametros;

}

}

}

Clase conexión:

class Conexion

{

SqlConnection Miconexion;

public void Conectar()

{

Miconexion = new SqlConnection("Data Source=DESKTOP-PNHNJ5R;Initial Catalog=proyecto;Integrated Security=True");

Miconexion.Open();

}

public void Desconectar()

{

Miconexion.Close();

}

public int ConsultarIdCategoria(String NombreCategoria){

int categoria;

String query = "select id from categorias where nombre='" + NombreCategoria + "'";

SqlDataAdapter da = new SqlDataAdapter(query, "Data Source=DESKTOP-PNHNJ5R;Initial Catalog=proyecto;Integrated Security=True");

da.SelectCommand.Connection.Open();

categoria=Convert.ToInt32(da.SelectCommand.ExecuteScalar().ToString());

da.SelectCommand.Connection.Close();

return categoria;

}

public Boolean ConsultarCliente(int categoria)

{

try

{

Conexion objConexion = new Conexion();

int idEncontrado;

String query = "Select categoria\_id from clientes where categoria\_id='" + categoria + "'";

SqlDataAdapter da = new SqlDataAdapter(query, "Data Source=DESKTOP-PNHNJ5R;Initial Catalog=proyecto;Integrated Security=True");

da.SelectCommand.Connection.Open();

idEncontrado = Convert.ToInt32(da.SelectCommand.ExecuteScalar().ToString());

da.SelectCommand.Connection.Close();

if (idEncontrado == categoria)

{

return true;

}

else

{

return false;

}

}

catch (System.NullReferenceException)

{

return false;

}

}

public Boolean Buscar(int id) {

try{

Conexion objConexion = new Conexion();

int idEncontrado;

String query = "Select id from clientes where id='"+id+"'";

SqlDataAdapter da = new SqlDataAdapter(query, "Data Source=DESKTOP-PNHNJ5R;Initial Catalog=proyecto;Integrated Security=True");

da.SelectCommand.Connection.Open();

idEncontrado = Convert.ToInt32(da.SelectCommand.ExecuteScalar().ToString());

da.SelectCommand.Connection.Close();

if (idEncontrado == id)

{

return true;

}

}

catch (System.NullReferenceException)

{

}

return false;

}

public void EjecutarSql(String Cadenasql)

{

SqlCommand Misentencia = new SqlCommand(Cadenasql,Miconexion);

int FilasAfectadas = Misentencia.ExecuteNonQuery();

if (FilasAfectadas > 0)

MessageBox.Show("Operación realizada correctamente", "La base de Datos ha sido afectada", MessageBoxButtons.OK, MessageBoxIcon.Information);

else

MessageBox.Show("La operacion no pudo concretarse", "Error del sistema", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

public void ActualizarGrid(DataGridView dg, String Cadenasql)

{

this.Conectar();

System.Data.DataSet MiDataSet = new System.Data.DataSet();

SqlDataAdapter MiDataAdapter = new SqlDataAdapter(Cadenasql, Miconexion);

MiDataAdapter.Fill(MiDataSet, "ListaClientes");

dg.DataSource = MiDataSet;

dg.DataMember = "ListaClientes";

this.Desconectar();

}

}

Clase Home:

public partial class Home : Form

{

public Home()

{

InitializeComponent();

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void button4\_Click(object sender, EventArgs e)

{

NuevaCategoria objCategoria = new NuevaCategoria();

this.Hide();

objCategoria.Show();

}

private void Form1\_Load(object sender, EventArgs e)

{

// TODO: esta línea de código carga datos en la tabla 'proyectoDataSet.categorias' Puede moverla o quitarla según sea necesario.

this.categoriasTableAdapter.Fill(this.proyectoDataSet.categorias);

// TODO: esta línea de código carga datos en la tabla 'proyectoDataSet.ListaClientes' Puede moverla o quitarla según sea necesario.

this.listaClientesTableAdapter.Fill(this.proyectoDataSet.ListaClientes);

}

private void button1\_Click(object sender, EventArgs e){

RegClientes Registro = new RegClientes();

this.Hide();

Registro.Show();

}

private void button2\_Click(object sender, EventArgs e)

{

Clientes objCliente = new Clientes();

if (this.txtId.Text.Equals(""))

{

MessageBox.Show("El campo id se encuentra vacio");

}

else

{

try

{

int id = Convert.ToInt32(this.txtId.Text);

if(objCliente.Buscar(id)==false){

MessageBox.Show("No se encontro al cliente");

this.txtId.Clear();

}

else

{

Conexion actualizar = new Conexion();

objCliente.Eliminar(id);

actualizar.ActualizarGrid(this.dataGridView1, "Select \* from ListaClientes");

this.txtId.Clear();

}

}

catch(System.FormatException)

{

MessageBox.Show("El campo id solo acepta valores numericos");

}

}

}

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

{

ModClientes modificar = new ModClientes();

Clientes objCliente = new Clientes();

String idCliente = txtId.Text;

if (idCliente.Equals(""))

{

MessageBox.Show("El campo id se encuentra vacio");

}

else

{

try

{

int id = Convert.ToInt32(this.txtId.Text);

if (objCliente.Buscar(id) == false)

{

MessageBox.Show("No se encontro al cliente");

this.txtId.Clear();

}

else

{

modificar.txtIdModClientes.Text = idCliente;

this.Hide();

modificar.Show();

}

}

catch(System.FormatException)

{

MessageBox.Show("El campo id solo acepta valores numericos");

}

}

}

private void button1\_Click\_1(object sender, EventArgs e)

{

Application.Exit();

}

private void button5\_Click(object sender, EventArgs e)

{

String nombreCategoria= this.cbCategorias.Text;

Conexion objConexion= new Conexion();

int categoria=objConexion.ConsultarIdCategoria(nombreCategoria);

bool vinculado=objConexion.ConsultarCliente(categoria);

if (vinculado==true)

{

MessageBox.Show("No se permite eliminar una categoria que se encuentre vincualada a un cliente","Error al eliminar");

}

else

{

Categorias eliminar = new Categorias();

eliminar.Eliminar(categoria);

this.Close();

Home objHome = new Home();

objHome.Show();

}

}

private void txtId\_TextChanged(object sender, EventArgs e)

{

}

private void txtId\_KeyPress(object sender, KeyPressEventArgs e)

{

if (!(char.IsNumber(e.KeyChar)) && (e.KeyChar != (char)Keys.Back))

{

MessageBox.Show("Solo se permiten numeros", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

e.Handled = true;

return;

}

}

}

Clase ModClientes:

public partial class ModClientes : Form

{

public ModClientes()

{

InitializeComponent();

}

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

{

Home objHome = new Home();

this.Hide();

objHome.Show();

}

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

{

Clientes objClientes=new Clientes();

Conexion objConexion=new Conexion();

String nombre=this.txtNombre.Text;

String apellidos=this.txtApellidos.Text;

String telefono=this.txtTelefono.Text;

String email=this.txtEmail.Text;

int idCliente =Convert.ToInt32(this.txtIdModClientes.Text);

int categoria=objConexion.ConsultarIdCategoria(cbCategoria.Text);

if(objClientes.RevisarParametros(nombre,apellidos,telefono,email)==true)

{

objClientes.Modificar(idCliente,nombre,apellidos,telefono,email,categoria);

this.Hide();

Home ObjHome = new Home();

ObjHome.Show();

}

}

private void groupBox1\_Enter(object sender, EventArgs e)

{

}

private void ModClientes\_Load(object sender, EventArgs e)

{

// TODO: esta línea de código carga datos en la tabla 'proyectoDataSet.categorias' Puede moverla o quitarla según sea necesario.

this.categoriasTableAdapter.Fill(this.proyectoDataSet.categorias);

Clientes objClientes=new Clientes();

this.txtIdModClientes.Visible = false;

int id = Convert.ToInt32(this.txtIdModClientes.Text);

this.txtNombre.Text=objClientes.ObtenerNombre(id);

this.txtApellidos.Text=objClientes.ObtenerApellidos(id);

this.txtTelefono.Text = objClientes.ObtenerTelefono(id) ;

this.txtEmail.Text=objClientes.ObtenerEmail(id);

}

private void txtTelefono\_TextChanged(object sender, EventArgs e)

{

}

private void txtTelefono\_KeyPress(object sender, KeyPressEventArgs e)

{

if (Char.IsDigit(e.KeyChar))

{

e.Handled = false;

}

else if (Char.IsControl(e.KeyChar))

{

e.Handled = false;

}

else if (Char.IsSeparator(e.KeyChar))

{

e.Handled = false;

}

else

{

e.Handled = true;

}

}

private void txtNombre\_TextChanged(object sender, EventArgs e)

{

}

private void txtNombre\_KeyPress(object sender, KeyPressEventArgs e)

{

if (!(char.IsLetter(e.KeyChar)) && (e.KeyChar != (char)Keys.Back))

{

MessageBox.Show("Solo se permiten letras", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

e.Handled = true;

return;

}

}

private void txtApellidos\_KeyPress(object sender, KeyPressEventArgs e)

{

if (!(char.IsLetter(e.KeyChar)) && (e.KeyChar != (char)Keys.Back))

{

MessageBox.Show("Solo se permiten letras", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

e.Handled = true;

return;

}

}

}

Clase NuevaCategoria:

public partial class NuevaCategoria : Form

{

public NuevaCategoria()

{

InitializeComponent();

}

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

{

Categorias objCategorias = new Categorias();

String nombre = this.txtNombre.Text;

objCategorias.Agregar(nombre);

this.txtNombre.Clear();

}

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

{

Home objHome = new Home();

this.Hide();

objHome.Show();

}

private void txtNombre\_KeyPress(object sender, KeyPressEventArgs e)

{

if (!(char.IsLetter(e.KeyChar)) && (e.KeyChar != (char)Keys.Back))

{

MessageBox.Show("Solo se permiten letras", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

e.Handled = true;

return;

}

}

private void groupBox1\_Enter(object sender, EventArgs e)

{

}

private void NuevaCategoria\_Load(object sender, EventArgs e)

{

}

}

Clase RegClientes:

public partial class RegClientes : Form

{

public RegClientes()

{

InitializeComponent();

}

private void Form3\_Load(object sender, EventArgs e)

{

// TODO: esta línea de código carga datos en la tabla 'proyectoDataSet.categorias' Puede moverla o quitarla según sea necesario.

this.categoriasTableAdapter.Fill(this.proyectoDataSet.categorias);

}

public void LimpiarRegForm()

{

this.txtNombre.Clear();

this.txtApellidos.Clear();

this.txtTelefono.Clear();

this.txtEmail.Clear();

}

private void button1\_Click(object sender, EventArgs e)

{

Clientes objCliente = new Clientes();

Conexion objConexion = new Conexion();

String nombre=this.txtNombre.Text;

String apellidos=this.txtApellidos.Text;

String telefono=this.txtTelefono.Text;

String email=this.txtEmail.Text;

int categoria = objConexion.ConsultarIdCategoria( this.cbCategoria.Text);

if (objCliente.RevisarParametros(nombre, apellidos, telefono, email) == true)

{

objCliente.Agregar(nombre, apellidos, telefono, email, categoria);

LimpiarRegForm();

}

}

private void label4\_Click(object sender, EventArgs e)

{

}

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

{

Home objHome = new Home();

objHome.Show();

this.Hide();

}

private void txtTelefono\_KeyPress(object sender, KeyPressEventArgs e)

{

if (Char.IsDigit(e.KeyChar))

{

e.Handled = false;

}

else if (Char.IsControl(e.KeyChar))

{

e.Handled = false;

}

else if (Char.IsSeparator(e.KeyChar))

{

e.Handled = false;

}

else

{

e.Handled = true;

}

}

private void txtNombre\_KeyPress(object sender, KeyPressEventArgs e)

{

if (!(char.IsLetter(e.KeyChar)) && (e.KeyChar != (char)Keys.Back))

{

MessageBox.Show("Solo se permiten letras", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

e.Handled = true;

return;

}

}

private void txtApellidos\_KeyPress(object sender, KeyPressEventArgs e)

{

if (!(char.IsLetter(e.KeyChar)) && (e.KeyChar != (char)Keys.Back))

{

MessageBox.Show("Solo se permiten letras", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

e.Handled = true;

return;

}

}

}

## Compilación

No aplica para este proyecto ya que se hace desde Visual Studio.

## Pruebas

Las pruebas fueron realizadas conforme se compilaba el código.

## Postmortem

## Reportes del software Process Dashboard.

