

Calidad en el desarrollo del software

MiAppClientes

Bernal Ramírez Cristian

SAENZ DOMINGUEZ ANDRES

1502273

TI51 5°

TECNOLOGIAS DE LA INFORMACION Y LA COMUNICACIÓN

UNIVERSIDAD TECNOLÓGICA DE RIVIERA MAYA

15/02/2016

Tabla de contenido

**PLANEACIÓN**

**Objetivo**

Crear una aplicación conectada a una base de datos la cual permite realizar registros, modificaciones y eliminado de clientes relacionarlos a una determinada categoría.

**Alcance**

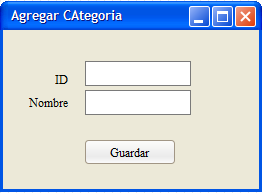
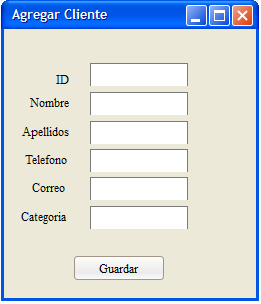
Una aplicación con una interfaz gráfica que permita realizar un mantenimiento de un sistema permitiendo agregar, eliminar y modificar los datos de clientes.

**Requerimientos**

* No permitir el borrado de categorías, si está relacionado con un cliente.
* Nombre y apellido son obligatorios.
* Correo electrónico valido.
* Nombre de categoría obligatorio.
* Un cliente debe estar relacionado con una categoría.
* Que el nombre de la categoría no puede quedar vacía.
* No permitir duplicado de correo electrónico.

DISEÑO





DEFINICIÓN DE LÍNEA DE CÓDIGO

Convención de nombres

* Nombre de archivos
* Nombre de clases

Los nombres de las clases deberán estar ligadas a un objeto abstracto del mundo real y debe de seguir la siguiente estructura:

* Primera letra en mayúscula.
* El nombre debe de estar en singular
* No deben de existir otra clase con el mismo nombre.
* Nombre de funciones/métodos

El nombre de las funciones o de los métodos deberán estar relacionadas a la función que realizarán en el programa. Por ejemplo, método agregar, es el método que agregara un nuevo usuario.

Se declarará el tipo de método (público, privado o protegido).

Se declara el tipo de dato que regresa.

La primera letra del nombre del método debe estar en mayúscula.

Después del nombre se declaran entre paréntesis las variables que usara.

La llave de apertura debe de estar en la misma línea de declaración del método.

La llave de cierre ira al final del método.

* Nombre de variables/propiedades

El nombre de variables se realizará de la siguiente manera:

Se declara la visibilidad del atributo.

Se declara el tipo de atributo (int, string, boolean).

El nombre del atributo debe de estar en minúscula y singular.

El nombre de la variable debe de relacionarse de acuerdo al dato que almacenara, por ejemplo: nombre almacenara nombre del usuario.

* Nombre de constantes

El nombre de las constantes

* Nombre de tablas

El nombre de las tablas

Estructuras de control

* If

if(expresión-booleana){

la expresión se evaluó verdadera

}

* Else

if(expresión-booleana) {

la expresión se evaluó verdadera

}else{

la expresión se evaluó falsa

}

* Switch

switch(variable) {

case valor1:

Instrucciones

break;

case valor2:

Instrucciones

break;

case valor3:

Instrucciones

break;

default:

Instrucciones

* Go to

goto <nombre\_de\_la\_etiqueta>

Estructuras loops/iterativas

* Do

do {

sentencia;

}

* Do while

do {

sentencia;

}while (condición);

* While

while (condición){

sentencia;

}

* For

for (valorInicial; CondiciónRepetición; Incremento){

Sentencia;

}

* Foreach

foreach (‘Tipo’ Nombre in Listado)

{

// Instrucciones.

}

Clases

class <nombreClase>

{

<miembros>

}

Método/función

<tipoAcceso> <tipoDevuelto> <nombre del método>()

{

<instrucciones>

}

Métodos con parámetros:

<tipoAcceso> <tipoDevuelto> <nombre del método>([parámetros]))

{

<instrucciones>

}

Variables/propiedades

[visibilidad] [tipo dato] [nombreVariable]

Constantes

[visibilidad] [tipo dato] [nombreConstante] = [dato asignado];

Comentarios

Multilinea:

/\*<texto>\*/

Una línea:

// <texto>

Codificación

//Metodo para el listado de clientes

public void listarclientes(DataGridView clientes, string cadenadatos)

{

this.conectar();

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

MySqlDataAdapter adaptador = new MySqlDataAdapter(cadenadatos, conexion);

adaptador.Fill(dataset, "clientes");

clientes.DataSource = dataset;

clientes.DataMember = "clientes";

clientes.ClearSelection();

this.desconectar();

}

//Metodo para agregar un nuevo cliente

public void agregarcliente(string nombre, string apellidos, string telefono, string correo, int cat\_id)

{

try

{

this.conectar();

string cadenadatos = "insert into clientes(id,nombre,apellidos,telefono,correo,categoria\_id) values(NULL, '" + nombre + "','" + apellidos + "','" + telefono + "','" + correo + "'," + cat\_id + ");";

MySqlCommand cmd = new MySqlCommand(cadenadatos, conexion);

MySqlDataReader reader = cmd.ExecuteReader();

this.desconectar();

}

catch (Exception n)

{

MessageBox.Show("Erros al guardar los datos \nLos datos no se almacenaran", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//Metodo para eliminar un cliente

public void eliminar(int id)

{

try

{

this.conectar();

string cadena = "delete from clientes where id=" + id + "";

MySqlCommand cmd = new MySqlCommand(cadena, conexion);

MySqlDataReader reader = cmd.ExecuteReader();

this.desconectar();

}

catch (Exception)

{

MessageBox.Show("No se puede eliminar", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//Metotodo para el listado de una categoria

public void listarcategoria(DataGridView categorias, string cadenadatos)

{

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

MySqlDataAdapter adaptador = new MySqlDataAdapter(cadenadatos, cn);

adaptador.Fill(dataset, "categoria");

categorias.DataSource = dataset;

categorias.DataMember = "categoria";

categorias.ClearSelection();

this.desconectar();

}

//Agrega una nueva categoria

public void guardarcategoria(string nombre)

{

this.conectar();

try

{

string cadenadatos = "insert into categoria(id,nombre) values (NULL,'" + nombre + "')";

this.conectar();

MySqlCommand cmd = new MySqlCommand(cadenadatos, cn);

MySqlDataReader reader = cmd.ExecuteReader();

this.desconectar();

}

catch (Exception)

{

MessageBox.Show("Erros al guardar el datos \nLos datos no se almacenaran", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

} }

Postmortem

