# SOURCE CODE FOR LOGIN

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

namespace NetworkMonitoring

{

public partial class frmLogin : Form

{

public bool ok;

public frmLogin()

{

InitializeComponent();

}

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

{

if (cls.con.State != ConnectionState.Open)

{

cls.con.Open();

}

cls.cmd.CommandText = "select count(\*) from admin where username='" + txtUser.Text + "' and Password='" + txtPass.Text + "'";

if (Convert.ToInt16(cls.cmd.ExecuteScalar().ToString()) > 0)

{

ok = true;

cls.con.Close();

this.Close();

}

else

{

ok = false;

cls.con.Close();

MessageBox.Show("Invalid UserName/Password.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

{

Application.Exit();

}

private void GroupBox1\_Enter(object sender, EventArgs e)

{

}

}

}

PROCESS

The Process module is used to ensure whether only added to the Process Module:

This process module fields are:

* Process ID
* Process Name
* Process PATH

Browse

Save

Delete

Clear

Exit

# SOURCE CODE PROCESS

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

using Microsoft.VisualBasic;

namespace NetworkMonitoring

{

public partial class frmAddProcess : Form

{

bool Searching = false;

public frmAddProcess()

{

InitializeComponent();

}

private void Button4\_Click(object sender, EventArgs e)

{

this.Close();

}

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

{

try

{

if (cls.con.State != ConnectionState.Open) cls.con.Open();

cls.cmd.CommandText = "Select ProcessId,ProcessName From Processes Order By ProcessId";

SqlDataReader r = null; //default(SqlDataReader);

SqlDataAdapter da = new SqlDataAdapter(cls.cmd);

DataTable dt = new DataTable();

DataView dv = null;// default(DataView);

r = cls.cmd.ExecuteReader();

while (r.Read())

{

//ListBox1.Items.Add(r(0))

ListBox1.Items.Add(r["ProcessId"] + ":" + r["ProcessName"]);

}

r.Close();

dt.Rows.Clear();

da.Fill(dt);

dv = dt.DefaultView;

}

catch (Exception ex)

{

Interaction.MsgBox(ex.Message, MsgBoxStyle.Critical, "");

}

finally

{

if (cls.con.State != ConnectionState.Closed) cls.con.Close();

}

}

private void Button1\_Click(object sender, EventArgs e)

{

txtProcessID.Text = "";

txtProcessName.Text = "";

// ClearControls();

try

{

if (cls.con.State != ConnectionState.Open)

{

cls.con.Open();

}

cls.cmd.CommandText = "Select Max(ProcessId) From Processes";

int sid;

if (cls.cmd.ExecuteScalar() == null)

sid = 1;

else if (cls.cmd.ExecuteScalar() == DBNull.Value )

sid = 1;

else

{

sid = int.Parse(cls.cmd.ExecuteScalar().ToString()) + 1;

}

txtProcessID.Text = sid.ToString();

}

catch (Exception ex)

{

Interaction.MsgBox(ex.Message, MsgBoxStyle.Critical, "");

}

finally

{

if (cls.con.State != ConnectionState.Closed)

{

cls.con.Close();

}

}

txtProcessName.Focus();

}

private void Button2\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(txtProcessName.Text.Trim()))

{

Interaction.MsgBox("Please Enter Process Name.", MsgBoxStyle.Information, "");

txtProcessName.Focus();

return;

}

SqlTransaction tran = null;

try

{

if (cls.con.State != ConnectionState.Open) cls.con.Open();

tran = cls.con.BeginTransaction();

cls.cmd.Transaction = tran;

cls.cmd.CommandText = "Delete From Processes Where ProcessId=" + txtProcessID.Text.Trim().Replace("'", "''") + "";

cls.cmd.ExecuteNonQuery();

cls.cmd.CommandText = "Insert into Processes Values(" + txtProcessID.Text.Trim().Replace("'", "''") + ",'" + txtProcessName.Text + "','" + textBox1.Text + "')";

if (Convert.ToInt32(cls.cmd.ExecuteNonQuery()) > 0)

{

//Data Affected

tran.Commit();

ListBox1.Items.Add(txtProcessID.Text.Trim().Replace("'", "''") + ":" + txtProcessName.Text.Trim().Replace("'", "''"));

Interaction.MsgBox("Process Details Saved...", MsgBoxStyle.Information, "");

}

else

{

Interaction.MsgBox("Process Details Could Not Be Saved...", MsgBoxStyle.Information, "");

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message.ToString());

try

{

tran.Rollback();

}

catch (Exception ex2)

{

Console.WriteLine(ex2.Message.ToString());

}

Interaction.MsgBox(ex.Message, MsgBoxStyle.Critical, "");

}

finally

{

if ((tran != null)) tran = null;

if (cls.con.State != ConnectionState.Closed) cls.con.Close();

}

}

private void Button5\_Click(object sender, EventArgs e)

{

Searching = true;

if (ListBox1.SelectedIndex >= 0)

{

try

{

if (cls.con.State != ConnectionState.Open) cls.con.Open();

string[] s = Strings.Split(ListBox1.Text, ":", -1, CompareMethod.Text);

cls.cmd.CommandText = "Delete From Processes Where ProcessId=" + s[0] + "";

if (Convert.ToInt32(cls.cmd.ExecuteNonQuery()) > 0)

{

ListBox1.Items.Remove(ListBox1.Text);

Interaction.MsgBox("Process Details Successfully Deleted", MsgBoxStyle.Exclamation, "");

}

else

{

Interaction.MsgBox("Process Details Could Not Be Deleted", MsgBoxStyle.Exclamation, "");

}

}

catch (Exception ex)

{

Interaction.MsgBox(ex.Message, MsgBoxStyle.Critical, "");

}

finally

{

if (cls.con.State != ConnectionState.Closed) cls.con.Close();

}

}

Searching = false;

}

private void ListBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

Searching = true;

if (ListBox1.SelectedIndex >= 0)

{

try

{

if (cls.con.State != ConnectionState.Open) cls.con.Open();

string[] s = Strings.Split(ListBox1.Text, ":", -1, CompareMethod.Text);

cls.cmd.CommandText = "Select \* From Processes Where ProcessId=" + s[0] + "";

SqlDataReader r = cls.cmd.ExecuteReader();

if (r.Read())

{

txtProcessID.Text = r[0].ToString();

txtProcessName.Text = r[1].ToString();

textBox1.Text = r[2].ToString();

}

r.Close();

}

catch (Exception ex)

{

Interaction.MsgBox(ex.Message, MsgBoxStyle.Critical, "");

}

finally

{

if (cls.con.State != ConnectionState.Closed) cls.con.Close();

}

}

Searching = false;

}

private void button3\_Click(object sender, EventArgs e)

{

openFileDialog1.Filter = "Executable Files|\*.exe";

if (openFileDialog1.ShowDialog() == DialogResult.OK)

textBox1.Text = openFileDialog1.FileName;

}

}

}

}

# SOURCE CODE PROCESS

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

using Microsoft.VisualBasic;

namespace NetworkMonitoring

{

public partial class frmAddProcess : Form

{

bool Searching = false;

public frmAddProcess()

{

InitializeComponent();

}

private void Button4\_Click(object sender, EventArgs e)

{

this.Close();

}

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

{

try

{

if (cls.con.State != ConnectionState.Open) cls.con.Open();

cls.cmd.CommandText = "Select ProcessId,ProcessName From Processes Order By ProcessId";

SqlDataReader r = null; //default(SqlDataReader);

SqlDataAdapter da = new SqlDataAdapter(cls.cmd);

DataTable dt = new DataTable();

DataView dv = null;// default(DataView);

r = cls.cmd.ExecuteReader();

while (r.Read())

{

//ListBox1.Items.Add(r(0))

ListBox1.Items.Add(r["ProcessId"] + ":" + r["ProcessName"]);

}

r.Close();

dt.Rows.Clear();

da.Fill(dt);

dv = dt.DefaultView;

}

catch (Exception ex)

{

Interaction.MsgBox(ex.Message, MsgBoxStyle.Critical, "");

}

finally

{

if (cls.con.State != ConnectionState.Closed) cls.con.Close();

}

}

private void Button1\_Click(object sender, EventArgs e)

{

txtProcessID.Text = "";

txtProcessName.Text = "";

// ClearControls();

try

{

if (cls.con.State != ConnectionState.Open)

{

cls.con.Open();

}

cls.cmd.CommandText = "Select Max(ProcessId) From Processes";

int sid;

if (cls.cmd.ExecuteScalar() == null)

sid = 1;

else if (cls.cmd.ExecuteScalar() == DBNull.Value )

sid = 1;

else

{

sid = int.Parse(cls.cmd.ExecuteScalar().ToString()) + 1;

}

txtProcessID.Text = sid.ToString();

}

catch (Exception ex)

{

Interaction.MsgBox(ex.Message, MsgBoxStyle.Critical, "");

}

finally

{

if (cls.con.State != ConnectionState.Closed)

{

cls.con.Close();

}

}

txtProcessName.Focus();

}

private void Button2\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(txtProcessName.Text.Trim()))

{

Interaction.MsgBox("Please Enter Process Name.", MsgBoxStyle.Information, "");

txtProcessName.Focus();

return;

}

SqlTransaction tran = null;

try

{

if (cls.con.State != ConnectionState.Open) cls.con.Open();

tran = cls.con.BeginTransaction();

cls.cmd.Transaction = tran;

cls.cmd.CommandText = "Delete From Processes Where ProcessId=" + txtProcessID.Text.Trim().Replace("'", "''") + "";

cls.cmd.ExecuteNonQuery();

cls.cmd.CommandText = "Insert into Processes Values(" + txtProcessID.Text.Trim().Replace("'", "''") + ",'" + txtProcessName.Text + "','" + textBox1.Text + "')";

if (Convert.ToInt32(cls.cmd.ExecuteNonQuery()) > 0)

{

//Data Affected

tran.Commit();

ListBox1.Items.Add(txtProcessID.Text.Trim().Replace("'", "''") + ":" + txtProcessName.Text.Trim().Replace("'", "''"));

Interaction.MsgBox("Process Details Saved...", MsgBoxStyle.Information, "");

}

else

{

Interaction.MsgBox("Process Details Could Not Be Saved...", MsgBoxStyle.Information, "");

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message.ToString());

try

{

tran.Rollback();

}

catch (Exception ex2)

{

Console.WriteLine(ex2.Message.ToString());

}

Interaction.MsgBox(ex.Message, MsgBoxStyle.Critical, "");

}

finally

{

if ((tran != null)) tran = null;

if (cls.con.State != ConnectionState.Closed) cls.con.Close();

}

}

private void Button5\_Click(object sender, EventArgs e)

{

Searching = true;

if (ListBox1.SelectedIndex >= 0)

{

try

{

if (cls.con.State != ConnectionState.Open) cls.con.Open();

string[] s = Strings.Split(ListBox1.Text, ":", -1, CompareMethod.Text);

cls.cmd.CommandText = "Delete From Processes Where ProcessId=" + s[0] + "";

if (Convert.ToInt32(cls.cmd.ExecuteNonQuery()) > 0)

{

ListBox1.Items.Remove(ListBox1.Text);

Interaction.MsgBox("Process Details Successfully Deleted", MsgBoxStyle.Exclamation, "");

}

else

{

Interaction.MsgBox("Process Details Could Not Be Deleted", MsgBoxStyle.Exclamation, "");

}

}

catch (Exception ex)

{

Interaction.MsgBox(ex.Message, MsgBoxStyle.Critical, "");

}

finally

{

if (cls.con.State != ConnectionState.Closed) cls.con.Close();

}

}

Searching = false;

}

private void ListBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

Searching = true;

if (ListBox1.SelectedIndex >= 0)

{

try

{

if (cls.con.State != ConnectionState.Open) cls.con.Open();

string[] s = Strings.Split(ListBox1.Text, ":", -1, CompareMethod.Text);

cls.cmd.CommandText = "Select \* From Processes Where ProcessId=" + s[0] + "";

SqlDataReader r = cls.cmd.ExecuteReader();

if (r.Read())

{

txtProcessID.Text = r[0].ToString();

txtProcessName.Text = r[1].ToString();

textBox1.Text = r[2].ToString();

}

r.Close();

}

catch (Exception ex)

{

Interaction.MsgBox(ex.Message, MsgBoxStyle.Critical, "");

}

finally

{

if (cls.con.State != ConnectionState.Closed) cls.con.Close();

}

}

Searching = false;

}

private void button3\_Click(object sender, EventArgs e)

{

openFileDialog1.Filter = "Executable Files|\*.exe";

if (openFileDialog1.ShowDialog() == DialogResult.OK)

textBox1.Text = openFileDialog1.FileName;

}

}

}

}

# SOURCE CODE TIMEINTERVAL

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

using Microsoft.VisualBasic;

namespace NetworkMonitoring

{

public partial class frmAddTimeInterval : Form

{

bool Searching = false;

public frmAddTimeInterval()

{

InitializeComponent();

}

private void Button4\_Click(object sender, EventArgs e)

{

this.Close();

}

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

{

}

private void Button1\_Click(object sender, EventArgs e)

{

txtTimeIntervalID.Text = "";

}

private void Button2\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(txtTimeIntervalID.Text.Trim()))

{

Interaction.MsgBox("Please Enter TimeInterval .", MsgBoxStyle.Information, "");

txtTimeIntervalID.Focus();

return;

}

SqlTransaction tran = null;

try

{

if (cls.con.State != ConnectionState.Open) cls.con.Open();

tran = cls.con.BeginTransaction();

cls.cmd.Transaction = tran;

cls.cmd.CommandText = "Delete From TimeInterval";

cls.cmd.ExecuteNonQuery();

cls.cmd.CommandText = "Insert into TimeInterval Values(" + txtTimeIntervalID.Text.Trim().Replace("'", "''") + ")";

if (Convert.ToInt32(cls.cmd.ExecuteNonQuery()) > 0)

{

//Data Affected

tran.Commit();

Interaction.MsgBox("TimeInterval Details Saved...", MsgBoxStyle.Information, "");

}

else

{

Interaction.MsgBox("TimeInterval Details Could Not Be Saved...", MsgBoxStyle.Information, "");

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message.ToString());

try

{

tran.Rollback();

}

catch (Exception ex2)

{

Console.WriteLine(ex2.Message.ToString());

}

Interaction.MsgBox(ex.Message, MsgBoxStyle.Critical, "");

}

finally

{

if ((tran != null)) tran = null;

if (cls.con.State != ConnectionState.Closed) cls.con.Close();

}

}

}

}

# SOURCE CODE SNAPSHOT

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient ;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

namespace NetworkMonitoring

{

public partial class frmSnapshot : Form

{

public frmSnapshot()

{

InitializeComponent();

}

string ip = "";

string compname = "";

public void loadform()

{

if (cls.con.State != ConnectionState.Open) cls.con.Open();

cls.cmd.CommandText = "Select TimeIntervalinSeconds From TimeInterval";

int time = 60;

if (cls.cmd.ExecuteScalar() == null)

{

}

else

{

time = int.Parse(cls.cmd.ExecuteScalar().ToString());

}

ip = System.Net.Dns.GetHostByName(System.Net.Dns.GetHostName()).AddressList[0].ToString();

compname = System.Net.Dns.GetHostName();

timer1.Interval = time \* 1000;

timer1.Enabled = true;

}

Bitmap bmp = null;

private void timer1\_Tick(object sender, EventArgs e)

{

bmp = CaptureScreen.CaptureDesktop();

if (cls.con.State != ConnectionState.Open) cls.con.Open();

cls.cmd.CommandText = "Insert into SnapShots (EntryTime,IPAddress,SystemName) Values ('" + string.Format("{0:MM-dd-yyyy hh:mm:ss tt}", DateTime.Now) + "','" + ip + "','" + compname + "')";

cls.cmd.ExecuteNonQuery();

cls.cmd.CommandText = "Select Max(SNo) From SnapShots";

int sno = int.Parse(cls.cmd.ExecuteScalar().ToString());

cls.cmd.CommandText = "Update SnapShots Set ScreenshotImageData=@ScreenshotImageData Where SNo=" + sno;

cls.cmd.Parameters.Clear();

SqlParameter p1 = new SqlParameter("@ScreenshotImageData", SqlDbType.Image);

System.IO.MemoryStream ms = new System.IO.MemoryStream();

bmp.Save(ms, System.Drawing.Imaging.ImageFormat.Jpeg);

byte[] b = ms.ToArray();

p1.Value = b;

cls.cmd.Parameters.Add(p1);

cls.cmd.ExecuteNonQuery();

//ScreenshotImageData

}

}

}

# SOURCE CODE SHOWPROCESS

using Microsoft.VisualBasic;

using System;

using System.Collections;

using System.Collections.Generic;

using System.Data;

using System.Diagnostics;

using System.Data.SqlClient;

namespace NetworkMonitoring

{

public partial class showprocess

{

DataSet ds = new DataSet();

SqlDataAdapter da2 = new SqlDataAdapter();

//Dim con As New SqlConnection("Provider=Microsoft.jet.Sql.4.0; data source=" + AppDomain.CurrentDomain.BaseDirectory + "process.mdb")

private void showprocess\_Load(System.Object sender, System.EventArgs e)

{

}

private void Button1\_Click(System.Object sender, System.EventArgs e)

{

this.Close();

}

string ip="";

string compname="";

private void showprocess\_Load\_1(object sender, EventArgs e)

{

// ds.Clear();

// string c1 = "Select \* from processes";

}

public void loadform()

{

if(cls.con.State!= ConnectionState.Open) cls.con.Open();

ip=System.Net.Dns.GetHostByName(System.Net.Dns.GetHostName()).AddressList[0].ToString();

compname =System.Net.Dns.GetHostName();

// cmd1.CommandText = c1;

// da2 = new SqlDataAdapter(cmd1);

// da2.Fill(ds);

// DataGridView1.DataSource = ds.Tables[0];

//// cls.con.Close();

}

private void timer1\_Tick(object sender, EventArgs e)

{

Process [] Prc = Process.GetProcesses();

if (cls.con.State != ConnectionState.Open) cls.con.Open();

int pid;

foreach (Process p in Prc)

{

cls.cmd.CommandText = "Select Count(\*) From Processes Where ProcessName='" + p.ProcessName + "'";

pid = int.Parse(cls.cmd.ExecuteScalar().ToString());

if (pid > 0)

{

cls.cmd.CommandText = "Insert into ProcessLog (ProcessId,EntryTime,IPAddress,SystemName) Values (" + p.Id + ",'" + string.Format("{0:MM-dd-yyyy hh:mm:ss tt}", DateTime.Now) + "','" + ip + "','" + compname + "')";

cls.cmd.ExecuteNonQuery();

}

}

}

private void Button1\_Click\_1(object sender, EventArgs e)

{

this.Close();

}

}

}