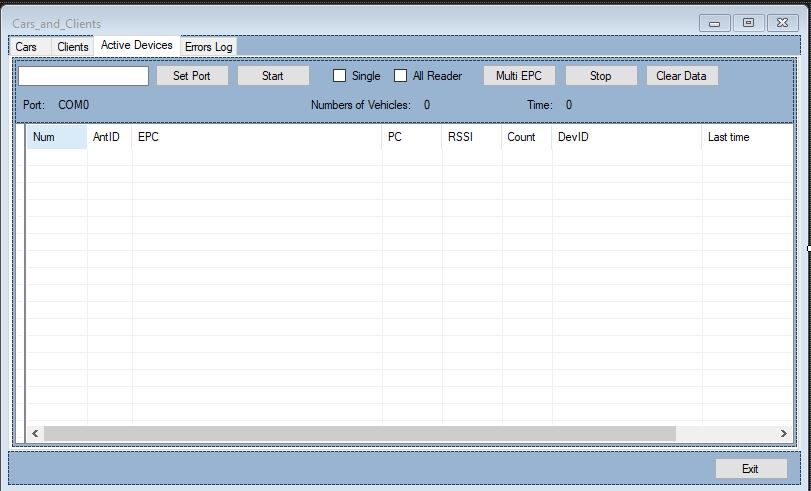
**Car Management Codes and Forms**

Display data from the device-Form:



Codes:

using System;

using System.Collections.Generic;

using System.Data;

using System.Windows.Forms;

using System.Data.SqlClient;

using NetFrame.Net.TCP.Sock.Asynchronous;

using System.Threading;

using System.IO.Ports;

using System.IO;

using System.Reflection;

using System.Text;

namespace Car\_Management

{

public partial class Cars\_and\_Clients : Form

{

private const int cash= 100;

private long totalnum1 = 0x00;

private long totalnum2 = 0x00;

private long totaltime = 0x00;

private const int listView\_md\_epc\_Num = 0;

private const int listView\_md\_epc\_AntID = 1;

private const int listView\_md\_epc\_EPC = 2;

private const int listView\_md\_epc\_PC = 3;

private const int listView\_md\_epc\_Rssi = 4;

private const int listView\_md\_epc\_Count = 5;

private const int listView\_md\_epc\_IP = 6;

private const int listView\_md\_epc\_Last\_Time = 7;

private const int listView\_md\_epc\_Direction = 8;

private const int listView\_md\_State = 3;

private volatile List<\_epc\_t> epcs\_list = new List<\_epc\_t>(1000);

private string portname = "";

private int baudRate = 230400;

private int dataBits = 8;

private Parity parity = Parity.None;

private StopBits stopbits = StopBits.One;

string error;

List<AsyncSocketState> clients;

public Cars\_and\_Clients()

{

InitializeComponent();

Control.CheckForIllegalCrossThreadCalls = false;

ReaderControllor.cmd.MultiEPC\_Event += ShowEPC;

this.listView\_md\_addr.Columns.Add("Num", 30, HorizontalAlignment.Left);

this.listView\_md\_addr.Columns.Add("IP", 100, HorizontalAlignment.Left);

this.listView\_md\_addr.Columns.Add("Port", 50, HorizontalAlignment.Left);

this.listView\_md\_addr.Columns.Add("ID", 50, HorizontalAlignment.Left);

this.listView\_md\_addr.Columns.Add("State", 50, HorizontalAlignment.Left);

this.listView\_md\_addr.GridLines = true;

this.listView\_md\_addr.FullRowSelect = true;

this.listView\_md\_addr.MultiSelect = false;

}

private void Cars\_and\_Clients\_Load(object sender, EventArgs e)

{

loadData();

}

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

{

New\_Car frmNewCar = new New\_Car();

frmNewCar.Show();

}

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

{

this.Close();

Form1 frm1 = new Form1();

frm1.Show();

}

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

{

New\_Client frmNewClient = new New\_Client();

frmNewClient.Show();

}

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

{

try

{

PortConfig SerialPortForm = new PortConfig();

SerialPortForm.ShowDialog();

if (SerialPortForm.result == true)

{

textBox1.Text = SerialPortForm.PortName;

portname = textBox1.Text;

baudRate = SerialPortForm.BuadRate;

dataBits = SerialPortForm.dataBits;

parity = SerialPortForm.parity;

stopbits = SerialPortForm.stopbits;

}

}

catch(Exception ex)

{

new LogWriter(ex);

}

}

bool serialisstart = false;

bool serverisstart = false;

private Reader ReaderControllor = new Reader();

private AsyncSocketState currentclient;

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

{

if (btnStartPort.Text == "Start")

{

portname = textBox1.Text;

try

{

ReaderControllor.ComStart(portname, baudRate, dataBits, parity, stopbits);

if (timer\_md\_query\_Tick.Enabled == false)

{

timer\_md\_query\_Tick.Enabled = true;

}

serialisstart = true;

lblPort.Text = textBox1.Text;

btnStartPort.Text = "Stop";

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

new LogWriter(ex);

}

}

else

{

serialisstart = false;

ReaderControllor.SerialPortClose();

if (serverisstart == false && serialisstart == false && timer\_md\_query\_Tick.Enabled == true)

{

timer\_md\_query\_Tick.Enabled = false;

btnStop.PerformClick();

}

btnStartPort.Text = "Start";

}

}

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

{

try

{

if (checkBoxMulti.Checked == true)

{

if (checkBoxSingle.Checked == true)

{

ReaderControllor.SingleEPC();

}

else

{

ReaderControllor.SatrtMultiEPC();

}

}

else

{

if (checkBoxSingle.Checked == true)

{

ReaderControllor.SingleEPC(currentclient);

}

else

{

ReaderControllor.SatrtMultiEPC(currentclient);

}

}

btnMultiEPC.Enabled = false;

}

catch (Exception ex)

{

new LogWriter(ex);

}

}

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

{

try

{

if (checkBoxMulti.Checked == true)

{

if (checkBoxSingle.Checked == true)

{

;

}

else

{

ReaderControllor.StopMultiEPC();

}

}

else

{

if (checkBoxSingle.Checked == true)

{

;

}

else

{

ReaderControllor.StopMultiEPC(currentclient);

}

}

btnMultiEPC.Enabled = true;

}

catch (Exception ex)

{

new LogWriter(ex);

}

}

public void loadData()

{

string connected;

DatabaseConnection check = new DatabaseConnection();

connected = check.checkDatabase();

try

{

if (connected == "true")

{

using (SqlConnection conn = new SqlConnection(DatabaseConnection.connectionStr))

{

var select = "SELECT \* FROM Contacts ";

var dataAdapter = new SqlDataAdapter(select, conn);

var select2 = "SELECT \* FROM Clients ";

var dataAdapter2 = new SqlDataAdapter(select2, conn);

var commandBuilder = new SqlCommandBuilder(dataAdapter);

var commandBuilder2 = new SqlCommandBuilder(dataAdapter2);

var ds = new DataSet();

dataAdapter.Fill(ds);

dataGridView1.ReadOnly = true;

dataGridView1.DataSource = ds.Tables[0];

dataGridView1.DefaultCellStyle.WrapMode = DataGridViewTriState.True;

var ds2 = new DataSet();

dataAdapter2.Fill(ds2);

dgvClients.ReadOnly = true;

dgvClients.DataSource = ds2.Tables[0];

dgvClients.DefaultCellStyle.WrapMode = DataGridViewTriState.True;

}

}

else

{

throw new Exception("Connection to the database was not established.");

//MessageBox.Show(error, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

//new LogWriter(error);

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

new LogWriter(ex);

}

}

public void ShowEPC(object sender, Command.ShowEPCEventArgs e)

{

try

{

\_epc\_t MultiID = e.MultiEPC;

bool isexit = false;

for (int index = 0; index < epcs\_list.Count; index++)

{

if ((epcs\_list[index].epc == MultiID.epc) && (epcs\_list[index].dev == MultiID.dev))

{

MultiID.count = epcs\_list[index].count + 1;

epcs\_list[index] = MultiID;

isexit = true;

break;

}

}

if (!isexit)

{

epcs\_list.Add(MultiID);

}

}

catch(Exception ex)

{

new LogWriter(ex);

}

}

string str\_epc = "";

string str\_pc = "";

string str\_read\_cnt ="" ;

string str\_ant\_id = "";

string str\_dev = "";

string str\_ip = "";

string str\_time = "";

string str\_rssi = "";

string direction = "";

const int price= 100;

private void timer\_md\_query\_Tick\_Tick\_1(object sender, EventArgs e)

{

try

{

string connected;

DatabaseConnection check = new DatabaseConnection();

connected = check.checkDatabase();

totalnum1 = 0;

totaltime++;

lblTime.Text = totaltime.ToString();

epcs\_list = ReaderControllor.GetMultiEPC();

lblNumVhcls.Text = epcs\_list.Count.ToString();

for (int index = 0; index < epcs\_list.Count; index++)

{

str\_epc = epcs\_list[index].epc;

str\_pc = epcs\_list[index].PC.ToString("X2");

str\_read\_cnt = epcs\_list[index].count.ToString();

str\_ant\_id = epcs\_list[index].antID.ToString();

str\_dev = epcs\_list[index].dev;

//str\_ip = epcs\_list[index].ClientIP;

str\_time = epcs\_list[index].time;

str\_rssi = epcs\_list[index].RSSI.ToString("f1");

direction = epcs\_list[index].direction.ToString();

totalnum1 += epcs\_list[index].count;

string scanTime;

double pri;

bool Exist = false;

int item\_index = 0;

string count2;

foreach (ListViewItem viewitem in this.listView\_md\_epc.Items)

{

using (SqlConnection conn = new SqlConnection(DatabaseConnection.connectionStr))

{

conn.Open();

string querry = "UPDATE Contacts SET Count=@str\_read\_cnt,Account=@pric where SerialNumber = '"+ str\_epc+"'";

string querry2 = "UPDATE DeviceData SET AntID=@str\_ant\_id,PC=@str\_pc,RSSI=@str\_rssi,Count=@str\_read\_cnt,Dir=@direction,LastTime=@str\_time,DevID=@str\_dev where SerialNumber = '" + str\_epc + "'";

string querry3 = "SELECT Account from CONTACTS where SerialNumber = '" + str\_epc + "'";

string querry4 = "SELECT LastTime from DeviceData where SerialNumber = '" + str\_epc + "'";

using (SqlCommand cmd3=new SqlCommand(querry3, conn))

{

pri = Convert.ToDouble(cmd3.ExecuteScalar());

}

using (SqlCommand cmd4 = new SqlCommand(querry4, conn))

{

scanTime = cmd4.ExecuteScalar().ToString();

}

var results = (Convert.ToDateTime(DateTime.Now)-Convert.ToDateTime(scanTime)).TotalMinutes;

if (results >= 1)

{

count2 = (Convert.ToInt32(str\_read\_cnt) + 1).ToString();

using (SqlCommand cmd = new SqlCommand(querry, conn))

{

cmd.Parameters.AddWithValue("@str\_read\_cnt", count2);

cmd.Parameters.AddWithValue("@pric", pri - (Convert.ToDouble(count2) \* price));

cmd.ExecuteNonQuery();

}

conn.Close();

conn.Open();

using (SqlCommand cmd2 = new SqlCommand(querry2, conn))

{

cmd2.Parameters.AddWithValue("@str\_read\_cnt", count2);

cmd2.Parameters.AddWithValue("@str\_ant\_id", @str\_ant\_id);

cmd2.Parameters.AddWithValue("@str\_pc", str\_pc);

cmd2.Parameters.AddWithValue("@direction", direction);

cmd2.Parameters.AddWithValue("@str\_time", str\_time);

cmd2.Parameters.AddWithValue("@str\_dev", str\_dev);

cmd2.Parameters.AddWithValue("@str\_rssi", str\_rssi);

cmd2.ExecuteNonQuery();

conn.Close();

}

}

else

{

count2 = "1";

using (SqlCommand cmd = new SqlCommand(querry, conn))

{

cmd.Parameters.AddWithValue("@str\_read\_cnt", count2);

cmd.Parameters.AddWithValue("@pric", pri - (Convert.ToDouble(count2) \* price));

cmd.ExecuteNonQuery();

}

conn.Close();

conn.Open();

using (SqlCommand cmd2 = new SqlCommand(querry2, conn))

{

cmd2.Parameters.AddWithValue("@str\_read\_cnt", count2);

cmd2.Parameters.AddWithValue("@str\_ant\_id", @str\_ant\_id);

cmd2.Parameters.AddWithValue("@str\_pc", str\_pc);

cmd2.Parameters.AddWithValue("@direction", direction);

cmd2.Parameters.AddWithValue("@str\_time", str\_time);

cmd2.Parameters.AddWithValue("@str\_dev", str\_dev);

cmd2.Parameters.AddWithValue("@str\_rssi", str\_rssi);

cmd2.ExecuteNonQuery();

conn.Close();

}

}

}

if ((viewitem.SubItems[listView\_md\_epc\_EPC].Text == str\_epc) && (viewitem.SubItems[listView\_md\_epc\_IP].Text == str\_dev))

{

viewitem.SubItems[listView\_md\_epc\_AntID].Text = str\_ant\_id;

viewitem.SubItems[listView\_md\_epc\_Count].Text = count2;

viewitem.SubItems[listView\_md\_epc\_Last\_Time].Text = str\_time;

viewitem.SubItems[listView\_md\_epc\_PC].Text = str\_pc;

viewitem.SubItems[listView\_md\_epc\_Rssi].Text = str\_rssi;

viewitem.SubItems[listView\_md\_epc\_Direction].Text = direction;

Exist = true;

}

item\_index++;

timer\_md\_query\_Tick.Stop();

timer\_md\_query\_Tick.Start();

}

if (!Exist)

{

ListViewItem item = new ListViewItem((this.listView\_md\_epc.Items.Count + 1).ToString());

item.SubItems.Add(str\_ant\_id);

item.SubItems.Add(str\_epc);

item.SubItems.Add(str\_pc);

item.SubItems.Add(str\_rssi);

item.SubItems.Add(str\_read\_cnt);

item.SubItems.Add(str\_dev);

item.SubItems.Add(str\_time);

item.SubItems.Add(direction);

this.listView\_md\_epc.Items.Add(item);

this.listView\_md\_epc.Items[this.listView\_md\_epc.Items.Count - 1].EnsureVisible();

this.listView\_md\_epc.Items[this.listView\_md\_epc.Items.Count - 1].Selected = true;

this.listView\_md\_epc.Items[this.listView\_md\_epc.Items.Count - 1].BackColor = System.Drawing.Color.FromArgb(red:200,blue:200,green:200);

break;

}

}

totalnum2 = totalnum1;

}

catch(Exception ex)

{

new LogWriter(ex);

}

}

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

{

ReaderControllor.GetMultiEPC().Clear();

epcs\_list.Clear();

listView\_md\_epc.Items.Clear();

lblNumVhcls.Text = "0";

totalnum1 = 0;

totalnum2 = 0;

totaltime = 0;

lblTime.Text = "0";

//label8.Text = "0";

}

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

{

loadData();

dgvClients.Update();

dgvClients.Refresh();

}

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

{

try

{

string path = Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments) + "\\" + @"Car Management\Logs\ErrorLogs.txt";

using (StreamReader streamReader = new StreamReader(path, Encoding.UTF8))

{

txtLog.Text = streamReader.ReadToEnd();

}

}

catch(Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private void Cars\_and\_Clients\_FormClosing(object sender, FormClosingEventArgs e)

{

btnStop.PerformClick();

timer\_md\_query\_Tick.Enabled = false;

}

private void listView\_md\_addr\_SelectedIndexChanged(object sender, EventArgs e)

{

int row = 0;

if (listView\_md\_addr.SelectedItems.Count > 0)

{

row = listView\_md\_addr.SelectedIndices[0];

}

currentclient = clients[row];

if (currentclient.types == connect.net)

{

lblPort.Text = "设备：" + currentclient.dev;

}

else

{

lblPort.Text = "设备：" + currentclient.com;

}

}

private delegate void mcListviewDelegate(int index, string text);

private void mcListviewUpdate(int index, string text)

{

if (listView\_md\_addr.InvokeRequired)

{

mcListviewDelegate d = new mcListviewDelegate(mcListviewUpdate);

listView\_md\_addr.Invoke(d, new object[] { index, text });

}

else

{

//int idx = Int32.Parse(index);

listView\_md\_addr.Items[index].SubItems[listView\_md\_State].Text = text;

}

}

private void timer\_scan\_Tick(object sender, EventArgs e)

{

listView\_md\_addr.Items.Clear();

clients = ReaderControllor.GetClientInfo();

foreach (AsyncSocketState client in clients)

{

ListViewItem item = new ListViewItem((this.listView\_md\_addr.Items.Count + 1).ToString());

if (client.types == connect.net)

{

item.SubItems.Add(client.ip\_addr);

item.SubItems.Add(client.port);

item.SubItems.Add(client.dev);

item.SubItems.Add(client.state);

this.listView\_md\_addr.Items.Add(item);

this.listView\_md\_addr.Items[this.listView\_md\_addr.Items.Count - 1].EnsureVisible();

}

else if (client.types == connect.com)

{

item.SubItems.Add(client.com);

item.SubItems.Add(" -- ");

item.SubItems.Add(client.dev);

item.SubItems.Add(client.state);

this.listView\_md\_addr.Items.Add(item);

this.listView\_md\_addr.Items[this.listView\_md\_addr.Items.Count - 1].EnsureVisible();

}

}

}

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

{

loadData();

dataGridView1.Update();

dataGridView1.Refresh();

}

}

}