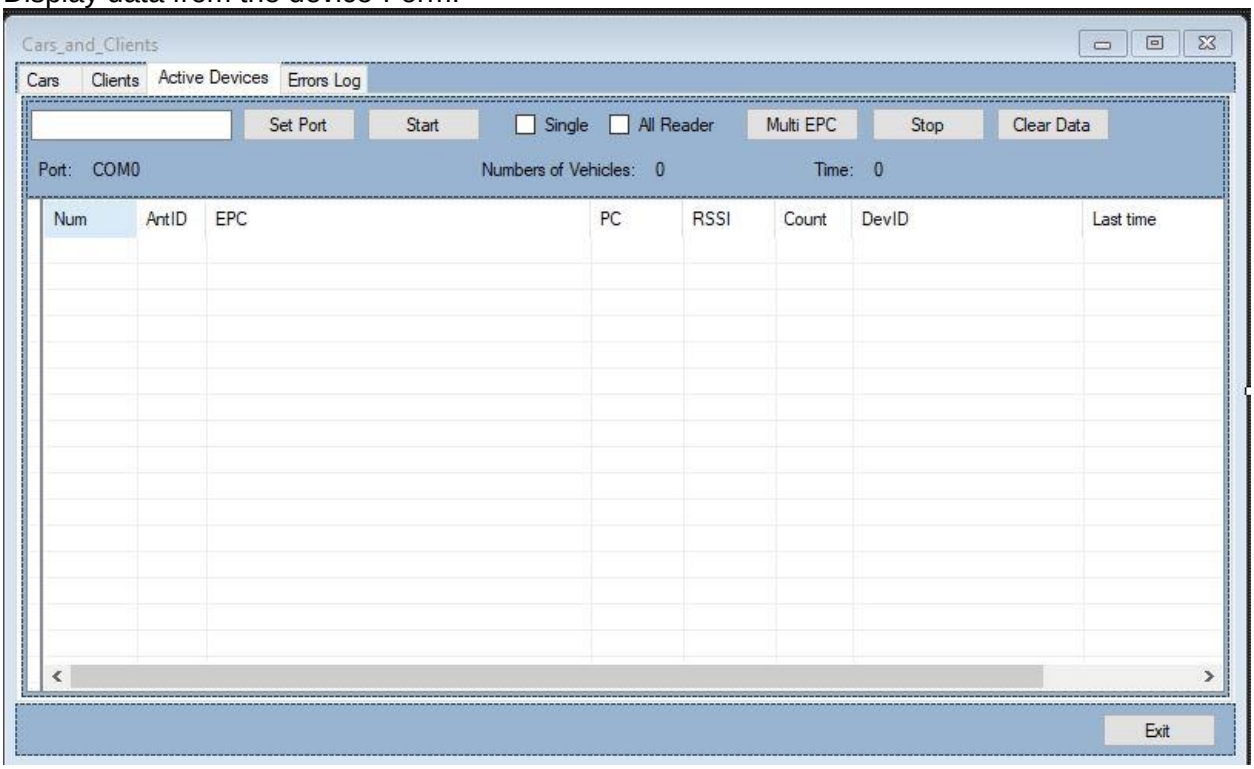
Car Management Codes and Forms

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();
        }
    }
}
```