1. **UDP:**

**A screenshot of a computer screen

Description automatically generated**

**UDP Client:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Net;

using System.Net.Sockets;

namespace UDP\_chat

{

public partial class UDP\_Client : Form

{

public UDP\_Client()

{

InitializeComponent();

}

private void button\_send\_Click(object sender, EventArgs e)

{

UdpClient client = new UdpClient();

IPAddress ipadd = IPAddress.Parse(textBox\_IP.Text);

int port = Convert.ToInt32(textBox\_port.Text);

IPEndPoint ipendpoint = new IPEndPoint(ipadd, port);

byte[] stringByte = Encoding.UTF8.GetBytes(richTextBox\_msg.Text);

client.Send(stringByte, stringByte.Length, ipendpoint);

richTextBox\_msg.Text = "";

}

}

}

**UDP Server:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Net;

using System.Net.Sockets;

namespace UDP\_chat\_server

{

public partial class UDP\_Server : Form

{

public UDP\_Server()

{

InitializeComponent();

}

private async void button\_listen\_Click(object sender, EventArgs e)

{

try

{

int port;

if (!int.TryParse(textBox\_port\_server.Text, out port))

{

MessageBox.Show("Port không hợp lệ");

return;

}

UdpClient udpClient = new UdpClient(port);

while (true)

{

UdpReceiveResult result = await udpClient.ReceiveAsync();

string data = Encoding.UTF8.GetString(result.Buffer);

string message = result.RemoteEndPoint.Address.ToString() + ":" + result.RemoteEndPoint.Port.ToString() + ":" + data;

richTextBox\_rmsg.Text = message;

}

}

catch (Exception ex)

{

MessageBox.Show("Error: " + ex.Message);

}

}

}

}

1. **TCP:**

A screenshot of a computer

Description automatically generated

**TCP Client:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Net;

using System.Net.Sockets;

namespace TCP\_chat\_client

{

public partial class TCP\_Client : Form

{

public TCP\_Client()

{

InitializeComponent();

}

private void button\_send\_Click(object sender, EventArgs e)

{

int port = Convert.ToInt32(textBox\_port.Text);

IPAddress ipadd = IPAddress.Parse(textBox\_IP.Text);

try

{

TcpClient tcpClient = new TcpClient(textBox\_IP.Text, port);

NetworkStream stream = tcpClient.GetStream();

byte[] stringBytes = Encoding.UTF8.GetBytes(richTextBox\_msg.Text);

stream.Write(stringBytes, 0, stringBytes.Length);

stream.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

}

}

**TCP Server:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Net.Sockets;

using System.Net;

using System.Threading;

namespace TCP\_chat\_server

{

public partial class Form1 : Form

{

private TcpListener listener;

public Form1()

{

InitializeComponent();

}

private void DisplayMessage(string message)

{

if (richTextBox\_rmsg.InvokeRequired)

{

richTextBox\_rmsg.Invoke((MethodInvoker)delegate {

richTextBox\_rmsg.Text = message;

});

}

else

{

richTextBox\_rmsg.Text = message;

}

}

private void HandleClientComm(object clientObj)

{

TcpClient client = (TcpClient)clientObj;

NetworkStream stream = client.GetStream();

byte[] data = new byte[1024];

try

{

while (true)

{

int bytesRead = stream.Read(data, 0, data.Length);

if (bytesRead == 0)

{

break; // Kết thúc khi client đóng kết nối

}

string message = Encoding.UTF8.GetString(data, 0, bytesRead);

DisplayMessage(message);

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi trong xử lý dữ liệu từ client: " + ex.Message);

}

finally

{

client.Close();

}

}

private void ListenForClients()

{

try

{

while (true)

{

TcpClient client = listener.AcceptTcpClient();

Thread clientThread = new Thread(HandleClientComm);

clientThread.IsBackground = true;

clientThread.Start(client);

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi: " + ex.Message);

}

}

private void button\_listen\_Click(object sender, EventArgs e)

{

int port = Convert.ToInt32(textBox\_port.Text);

try

{

listener = new TcpListener(IPAddress.Any, port);

listener.Start();

// Khởi tạo một luồng riêng biệt để lắng nghe kết nối từ client

Thread listenThread = new Thread(ListenForClients);

listenThread.IsBackground = true;

listenThread.Start();

MessageBox.Show("Đã bắt đầu lắng nghe kết nối từ client.");

}

catch (Exception ex)

{

MessageBox.Show("Lỗi: " + ex.Message);

}

}

}

}