

Tên: Trần Dương Minh Đại

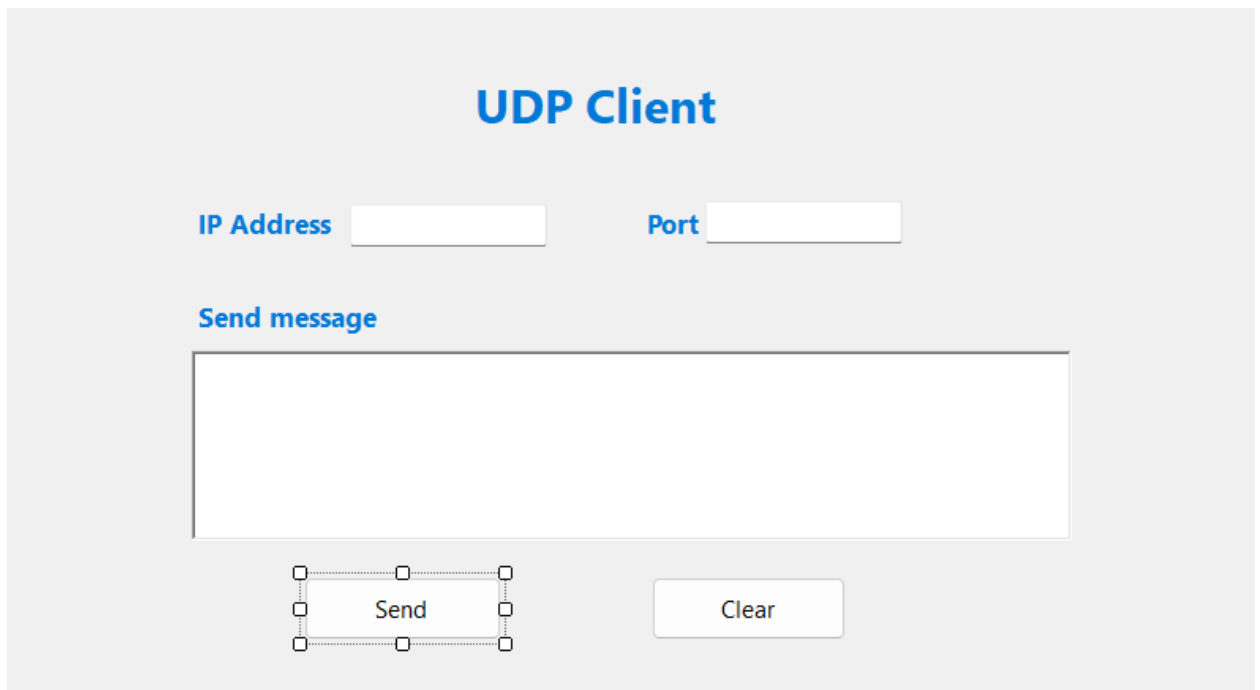
MSSV: 22520183

Báo cáo Lab 3

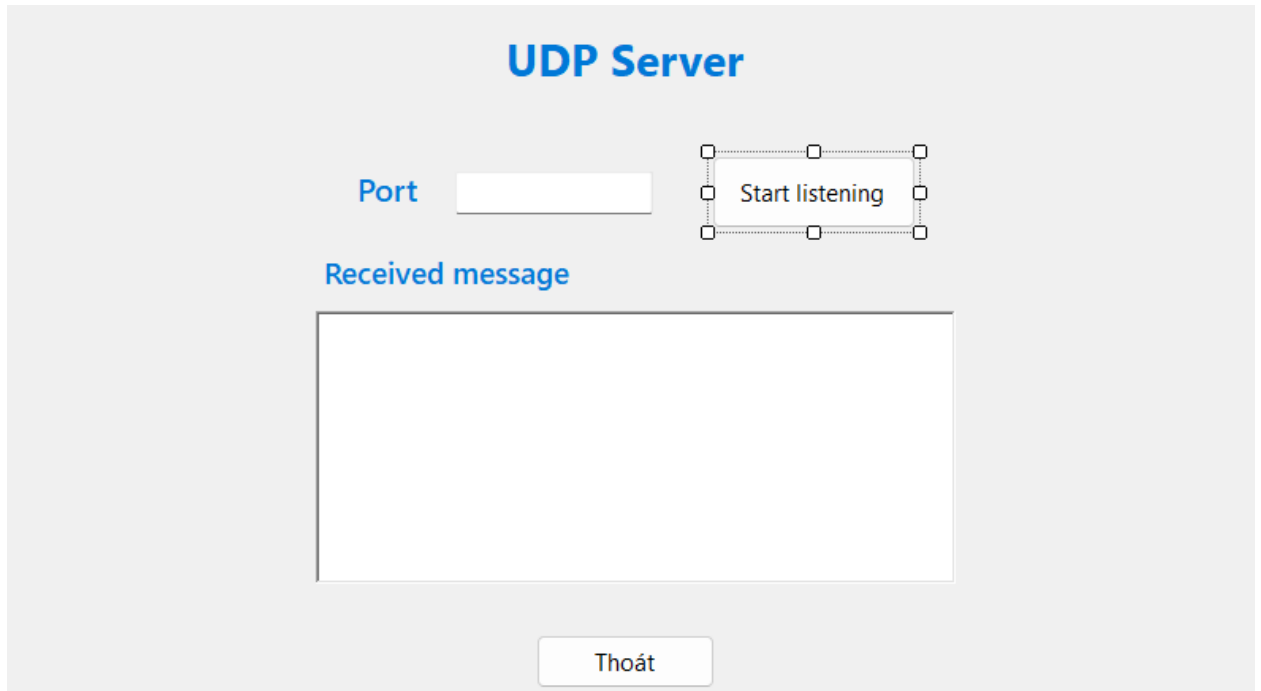
Lập trình mạng căn bản

1. Gửi nhận dữ liệu với UDP

- Screenshot bài 1:



The screenshot displays a web application titled "UDP Client" in blue text. Below the title, there are two input fields: "IP Address" and "Port", both with blue labels. Underneath these is a "Send message" label followed by a large, empty text area for input. At the bottom of the interface, there are two buttons: a "Send" button with a dashed border and a "Clear" button with a solid border. The entire interface is set against a light gray background.



- Đối với UDP Client, ta sẽ dùng class `UdpClient` để gửi dữ liệu đến

```
private void button1_Click(object sender, EventArgs e)
{
    UdpClient udpClient = new UdpClient();
    IPAddress ipadd = IPAddress.Parse(textBox1.Text);
    int port = Convert.ToInt32(textBox2.Text);
    IPEndPoint ipend = new IPEndPoint(ipadd, port);
    Byte[] sendBytes = Encoding.UTF8.GetBytes(richTextBox1.Text);
    udpClient.Send(sendBytes, sendBytes.Length, ipend);
}
```

IpEndpoint:

- Đối với UDP Server, ta sẽ dùng `Socket` để nhận dữ liệu. Ở đây ta sẽ xử lý bất đồng bộ bằng cách dùng phương thức `ReceiveCallback` luôn sẵn sàng nhận dữ liệu mới. Khi có dữ liệu được gửi đến, hàm callback này được gọi tự động, giúp xử lý dữ liệu và tiếp tục nhận dữ liệu tiếp theo mà không cần chờ đợi.

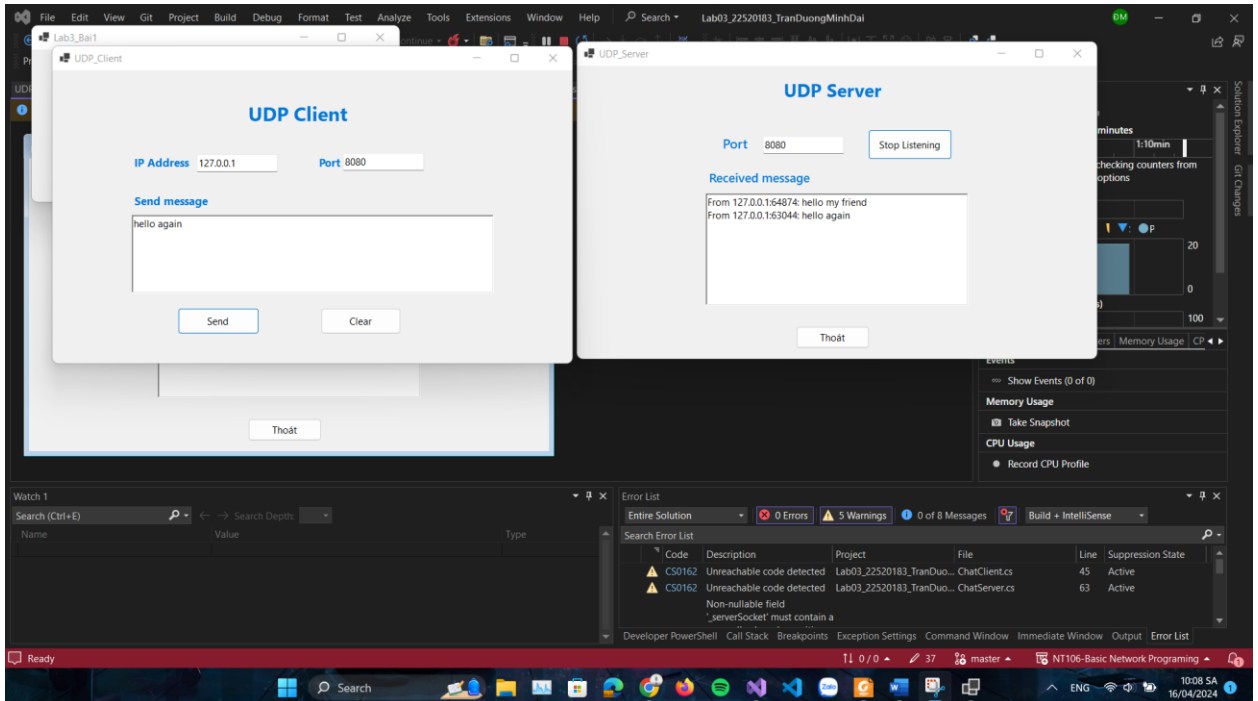
```
private void StartListening()
{
    udpSocket = new Socket(AddressFamily.InterNetwork, SocketType.Dgram, ProtocolType.Udp);
    int port = Convert.ToInt32(textBox1.Text);
    udpSocket.Bind(new IPEndPoint(IPAddress.Any, port));
    EndPoint ipRec = new IPEndPoint(IPAddress.Any, 0);
    udpSocket.BeginReceiveFrom(buffer, 0, buffer.Length, SocketFlags.None, ref ipRec, ReceiveCallback);
}

2 references
private void ReceiveCallback(IAsyncResult ar)
{
    try
    {
        EndPoint ipRec = new IPEndPoint(IPAddress.Any, 0);
        int bytesRead = udpSocket.EndReceiveFrom(ar, ref ipRec);

        string mess = Encoding.ASCII.GetString(buffer, 0, bytesRead);
        this.Invoke((MethodInvoker)delegate
        {
            richTextBox1.AppendText($"From {ipRec}: {mess}\r\n");
        });

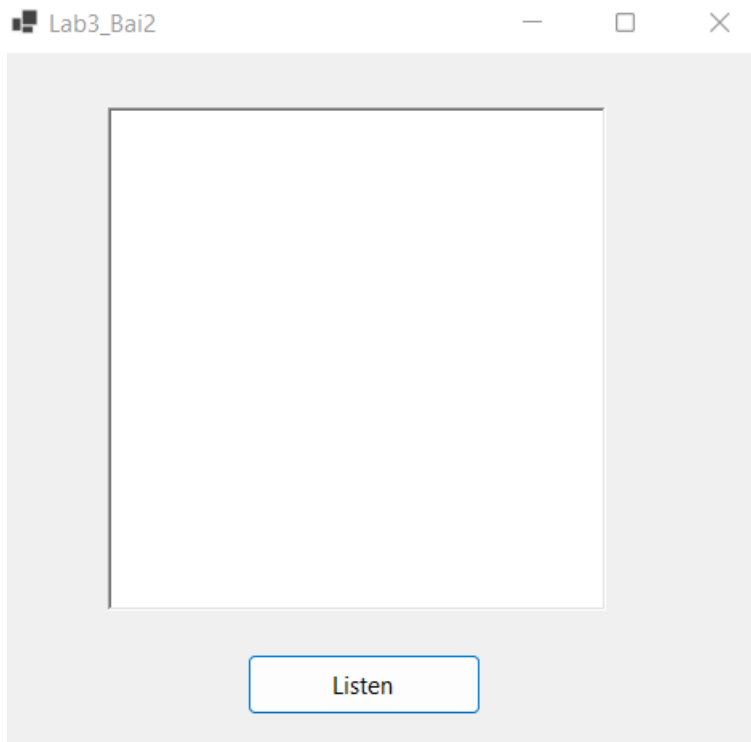
        EndPoint newIpRec = new IPEndPoint(IPAddress.Any, 0);
        udpSocket.BeginReceiveFrom(buffer, 0, buffer.Length, SocketFlags.None, ref newIpRec, ReceiveCallback);
    }
    catch { }
}
```

- Kết quả:



2. Gửi dữ liệu với Socket

- Screenshot bài 2:



- Ta dùng socket để tạo 2 socket là clientSocket và listenerSocket, với listenerSocket (AddressFamily là ipv4, dạng stream và protocol là tcp) sẽ gán tới IpEndpoint còn client socket sẽ accept kết nối.

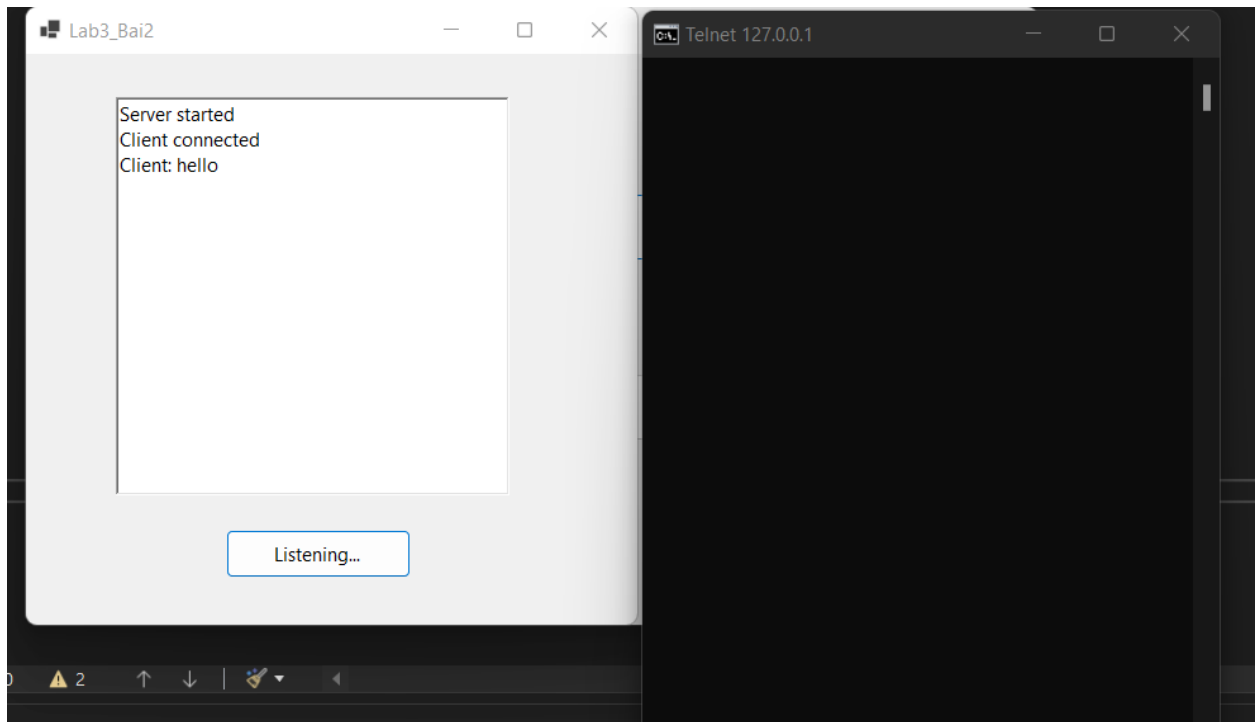
```
private void StartThread()
{
    richTextBox1.Text = "Server started\n";
    int bytesRecv = 0;
    byte[] recv = new byte[1];
    Socket clientSocket;

    Socket listenerSocket = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
    IPEndPoint ipepSV = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 8080);
    listenerSocket.Bind(ipepSV);
    listenerSocket.Listen(-1);
    clientSocket = listenerSocket.Accept();

    richTextBox1.Text += "Client connected\n";
    while (clientSocket.Connected)
    {
        string text = "";
        do
        {
            bytesRecv = clientSocket.Receive(recv);
            text += Encoding.ASCII.GetString(recv);
        }
        while (text[text.Length - 1] != '\n');
        richTextBox1.AppendText("Client: " + text);
    }
    listenerSocket.Close();
}
```

- Kết quả:

Sau khi telnet 127.0.0.1 8080 thì gõ hello thì bên server sẽ nhận được kết quả



-

-

3. – Gửi nhận dữ liệu với TCP (1S – 1C)

- Screenshot bài 3:

TCP Client

IP Port

TCP Listener

Port

- Đối với TcpClient, sau khi khởi tạo đối tượng TcpClient thì ta sẽ dùng method connect() tới IpEndpoint với port là user khai báo. Sau đó, ta tạo thread để nhận dữ liệu và gửi message đã kết nối tới server.

```
private void button1_Click(object sender, EventArgs e)
{
    client = new TcpClient();
    string ipAddress = textBox1.Text;
    int port = int.Parse(textBox2.Text);
    IPAddress ip = IPAddress.Parse(ipAddress);
    IPEndPoint endPoint = new IPEndPoint(ip, port);
    client.Connect(endPoint);
    stream = client.GetStream();
    button1.Text = "Connected";
    Thread Recv = new Thread(Receive);
    Recv.IsBackground = true;
    Recv.Start();
    byte[] data = Encoding.UTF8.GetBytes("Hello Server");
    stream.Write(data, 0, data.Length);
}

1 reference
void Receive(Object obj)
{
    while (true)
    {
        byte[] recv = new byte[1024];
        stream.Read(recv, 0, recv.Length);
        string str = Encoding.UTF8.GetString(recv);
        richTextBox2.AppendText(str+"\n");
    }
}
```

- Cuối cùng là dùng nút send để gửi dữ liệu với stream.write:

```
private void button2_Click(object sender, EventArgs e)
{
    try
    {
        string message = richTextBox1.Text;
        byte[] data = Encoding.UTF8.GetBytes(message);
        stream.Write(data, 0, data.Length);

        richTextBox2.AppendText($"{message}\n");
        richTextBox1.Clear();
    }
    catch (Exception ex)
    {
        MessageBox.Show($"Error: {ex.Message}");
    }
}
```

- Còn với TcpListener, ta sẽ gán IpEndpoint với TcpListener sau đó thì start để bắt đầu server, đồng thời là tạo thread để lắng nghe dữ liệu từ client:

```

private void StartServer()
{
    try
    {
        int port = int.Parse(textBox2.Text.Trim());

        server = new TcpListener(IPAddress.Any, port);
        server.Start();

        isServerRunning = true;
        serverThread = new Thread(new ThreadStart(ListenForClients));
        serverThread.Start();
        richTextBox1.AppendText($"Server running on port {port}\n");
        ListenForClients();
    }
    catch (Exception ex)
    {
        MessageBox.Show($"Error: {ex.Message}");
    }
}

```

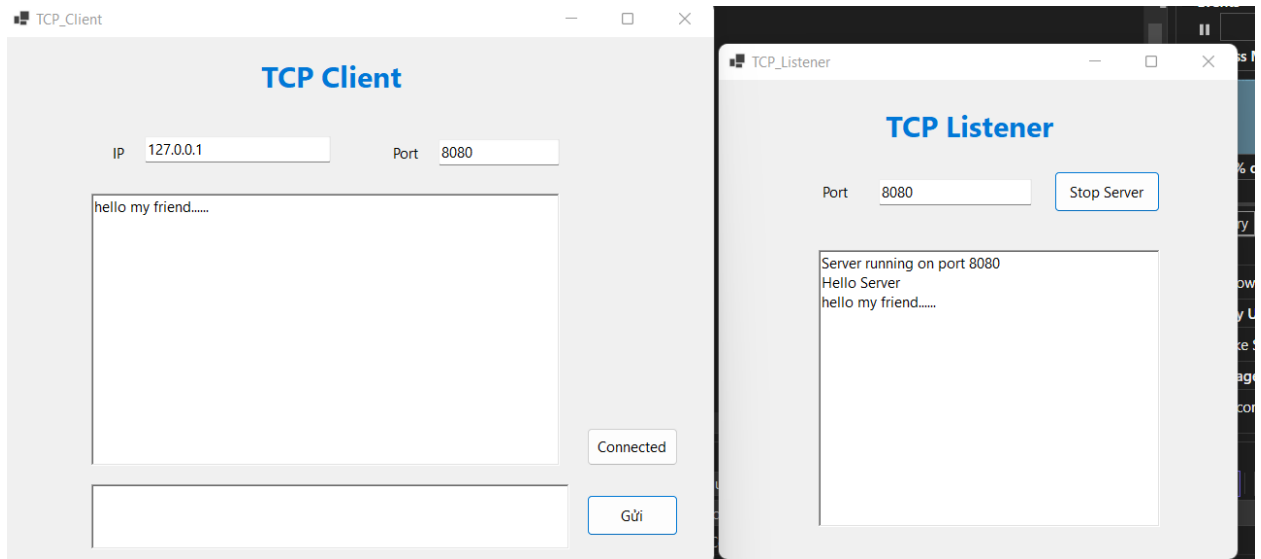
```

private async void ListenForClients()
{
    while (isServerRunning)
    {
        try
        {
            TcpClient client = await server.AcceptTcpClientAsync();
            NetworkStream stream = client.GetStream();
            byte[] buffer = new byte[1024];
            int bytesRead;

            while ((bytesRead = stream.Read(buffer, 0, buffer.Length)) > 0)
            {
                string message = Encoding.UTF8.GetString(buffer, 0, bytesRead);
                richTextBox1.AppendText(message+"\n");
            }
            client.Close();
        }
        catch (Exception ex)
        {
            MessageBox.Show($"Error: {ex.Message}");
        }
    }
}

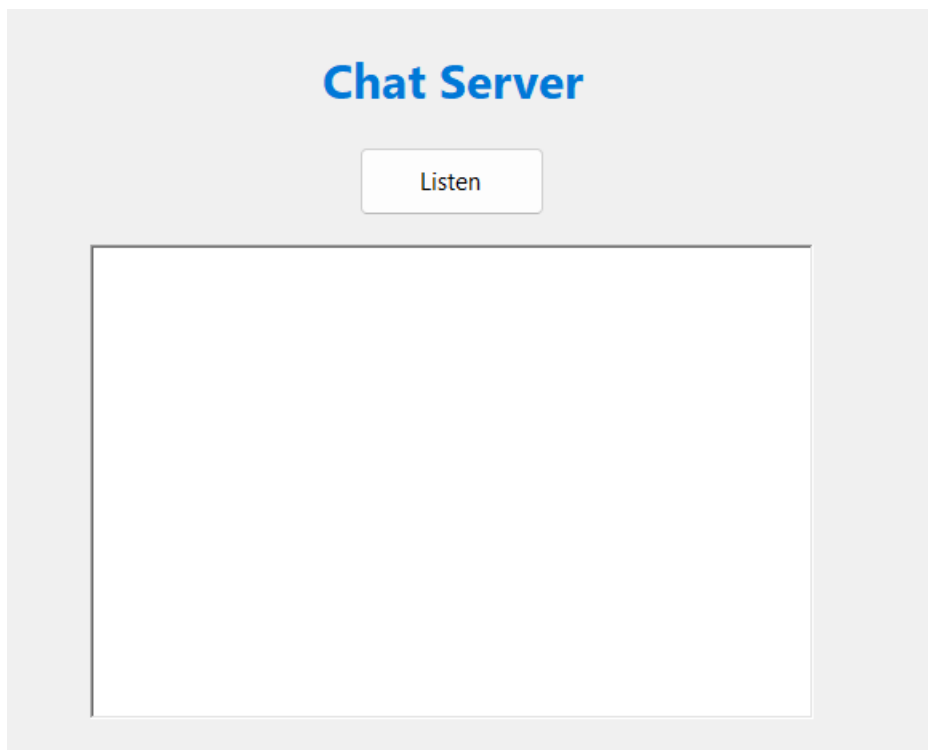
```

- Kết quả:



4. Gửi nhận dữ liệu (1S – N C)

- Screenshot bài 4:



Chat Client

Your name

Message

- Phần client sẽ tương tự như TcpClient khi ta cũng connect tới endpoint và tạo 1 thread nhận dữ liệu:

```
private void ReceiveMessages()
{
    int bytesRead;

    while (true)
    {
        bytesRead = clientStream.Read(message, 0, 4096);
        string receivedMessage = Encoding.ASCII.GetString(message, 0, bytesRead);
        richTextBox1.AppendText(receivedMessage + "\n");
    }
}

1 reference
private void button2_Click(object sender, EventArgs e)
{
    client = new TcpClient();
    client.Connect("127.0.0.1", 8080);
    clientStream = client.GetStream();

    richTextBox1.AppendText("Connected to server\n");

    Thread receiveThread = new Thread(ReceiveMessages);
    receiveThread.Start();
}
```

- Với nút gửi tin nhắn bên client:

```
private void button1_Click(object sender, EventArgs e)
{
    string sendMessage = textBox1.Text+": "+ richTextBox2.Text ;
    byte[] sendMessageBytes = Encoding.ASCII.GetBytes(sendMessage);
    clientStream.Write(sendMessageBytes, 0, sendMessageBytes.Length);
}
```

- Đối với server, ta tạo một list các TcpClient và thực hiện lắng nghe đến các clients:

```
private TcpListener server;
private List<TcpClient> clients = new List<TcpClient>();
private Thread listenThread;
1 reference
public ChatServer()
{
    InitializeComponent();
}

1 reference
private void button1_Click(object sender, EventArgs e)
{
    server = new TcpListener(IPAddress.Any, 8080);
    server.Start();
    listenThread = new Thread(ListenForClients);
    listenThread.Start();
    button1.Text = "Listening";
    richTextBox1.AppendText("Server is listening on port 8080...\n");
}
```

- Khi 1 client mới kết nối tới, ta sẽ một thread tương ứng với client đó:

```
1 reference
private void ListenForClients()
{
    while (true)
    {
        TcpClient client = server.AcceptTcpClient();
        clients.Add(client);

        string clientInfo = "New client connected from " + ((IPEndPoint)client.Client.RemoteEndPoint).ToString();
        richTextBox1.AppendText(clientInfo + "\n");
        Thread clientThread = new Thread(HandleClient);
        clientThread.Start(client);
    }
}
```

```

1 reference
private void HandleClient(object client)
{
    TcpClient tcpClient = (TcpClient)client;
    NetworkStream clientStream = tcpClient.GetStream();
    byte[] message = new byte[4096];
    int bytesRead;

    while (true)
    {
        bytesRead = clientStream.Read(message, 0, 4096);
        string receivedMessage = Encoding.ASCII.GetString(message, 0, bytesRead);
        BroadcastMessage(receivedMessage);
        richTextBox1.AppendText(receivedMessage + "\n");
    }
}

```

- Sau đó là broadcast message đến toàn bộ clients:

```

1 reference
private void BroadcastMessage(string message)
{
    foreach (TcpClient client in clients)
    {
        NetworkStream clientStream = client.GetStream();
        byte[] broadcastMessage = Encoding.ASCII.GetBytes(message);
        clientStream.Write(broadcastMessage, 0, broadcastMessage.Length);
    }
}

```

- Kết quả:

