BÁO CÁO THỰC HÀNH

Môn học: Lập trình mạng căn bản

Buổi báo cáo: Lab 03

Tên chủ đề: Lập trình Sockets trong C# (Sockets Programming in C#)

GVHD: Đỗ Thị Hương Lan

Ngày thực hiện: 23/10/2024

Ngày nộp báo cáo: 30/10/2024

THÔNG TIN CHUNG:

Lớp: NT106.P11.1

STT	Họ và tên	MSSV	Email
1	Nguyễn Lê Nhật Đăng	23520231	23520231@gm.uit.edu.vn
2	Trần Hải Đăng	23520237	23520237@gm.uit.edu.vn
3	Huỳnh Minh Đạt	23520249	23520249@gm.uit.edu.vn

ĐÁNH GIÁ KHÁC:

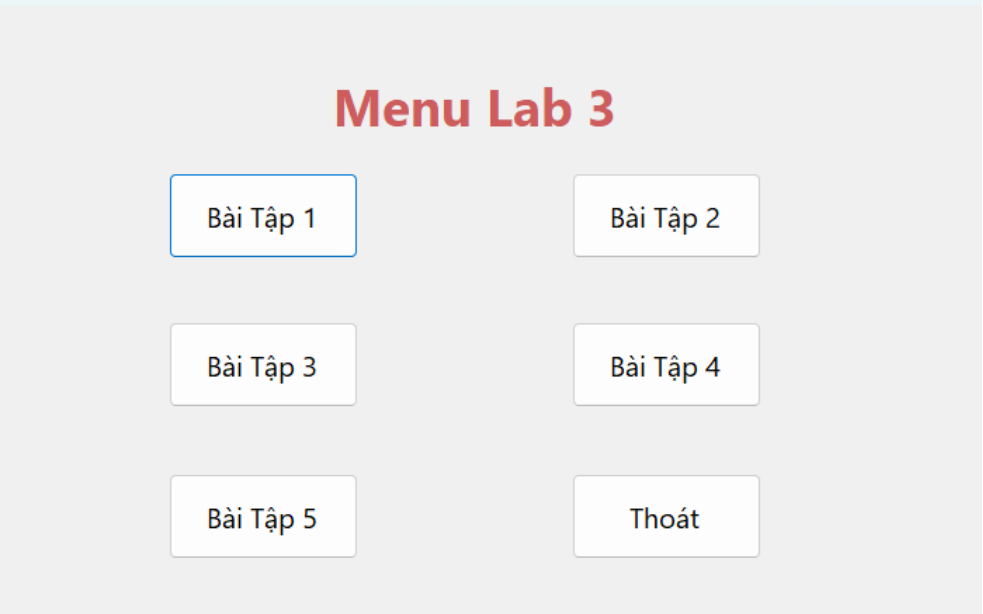
Nội dung	Kết quả
Tổng thời gian thực hiện bài thực hành trung bình	1 tuần
Link Video thực hiện *(nếu có)*	
Ý kiến *(nếu có)* + Khó khăn + Đề xuất …	

Phần bên dưới của báo cáo này là báo cáo chi tiết của nhóm thực hiện

MỤC LỤC

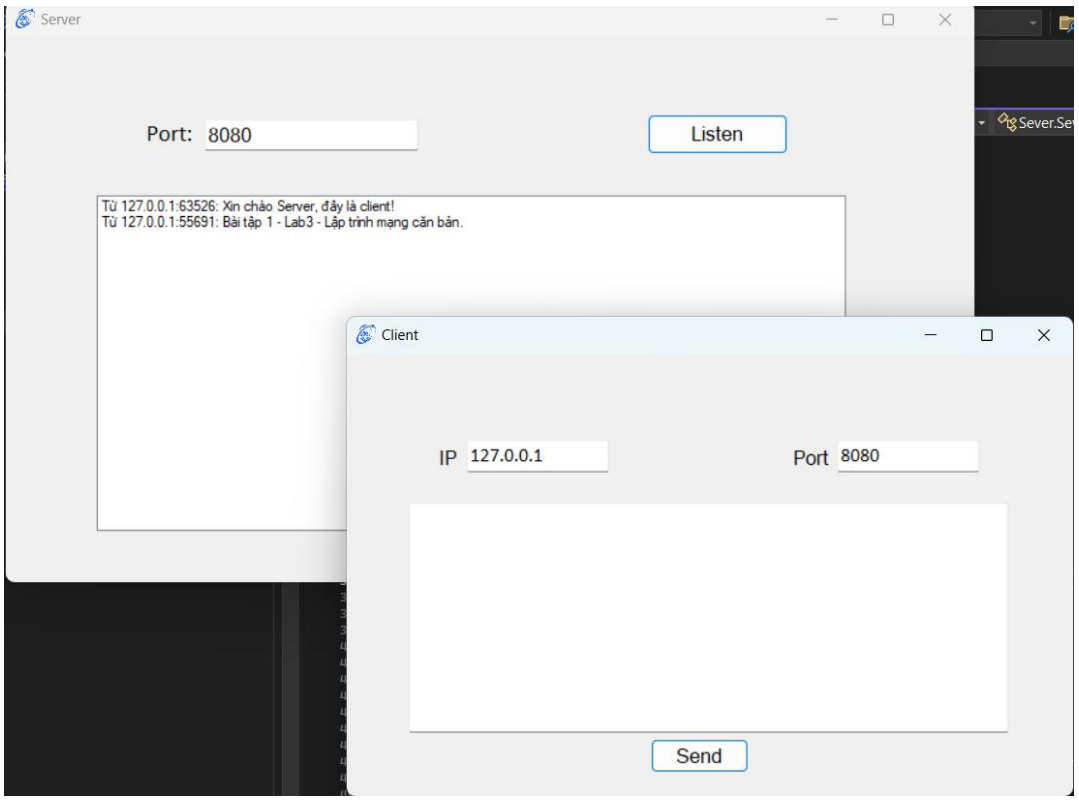
BÁO CÁO CHI TIẾT

1. *Form "Menu" của Lab03:*



2. *Bài tập 1 – Chương trình UDP (Client-Server):*

a) Tổng quan:



b) Chi tiết:

- Client – Server thuộc hai project khác nhau và được chạy song song.
- Client sử dụng địa chỉ IP loopback và Port phù hợp để kết nối và gửi thông điệp được nhập trong textbox đến Server.

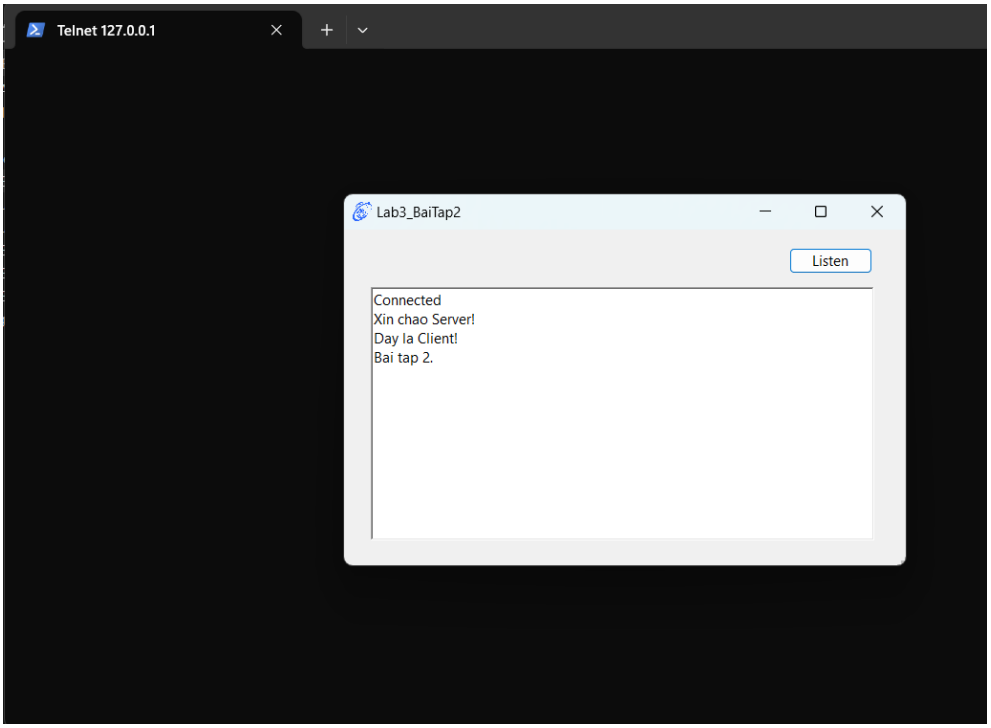
```
private void btnSend_Click(object sender, EventArgs e)
{
    try
    {
        client = new UdpClient();
        byte[] sendBytes = Encoding.Unicode.GetBytes(message_tb.Text);

        IPEndPoint remoteIpEndPoint = new IPEndPoint(IPAddress.Parse(ip_tb.Text), int.Parse(port_tb.Text));
        client.Send(sendBytes, sendBytes.Length, remoteIpEndPoint);
    }
    catch (Exception ex)
    {
        MessageBox.Show("Lỗi: " + ex.Message);// Báo lỗi nếu có.
    }
    finally
    {
        client.Close();
    }
    message_tb.Text = "";// Làm rỗng textbox thông điệp sau khi gửi.
}
```

- Server hiển thị thông điệp nhận được từ Client qua ListView.

```
private UdpClient server;
private IPEndPoint endpoint;

1 reference
private void listen_Click(object sender, EventArgs e)// Tạo thread để nhận thông điệp.
{
    Thread listenThread = new Thread(Listen);
    listenThread.Start();
}

1 reference
private void Listen()// Đón thông điệp từ Client.
{
    int port = Int32.Parse(port_tb.Text);
    server = new UdpClient(port);

    try
    {
        while (true)
        {
            IPEndPoint groupEP = new IPEndPoint(IPAddress.Any, port);
            byte[] data_byte = server.Receive(ref groupEP);
            string data = Encoding.Unicode.GetString(data_byte);
            Invoke((MethodInvoker)delegate
            {
                message_lbx.Items.Add($"Từ {groupEP.Address}:{groupEP.Port}: {data}{Environment.NewLine}");// Hiển thị thông điệp gửi từ Client.
            });
        }
    }
    catch (Exception e)
    {
        Console.WriteLine(e.ToString());// Thông báo lỗi nếu có.
    }
    finally
    {
        server.Close();
    }
}
```

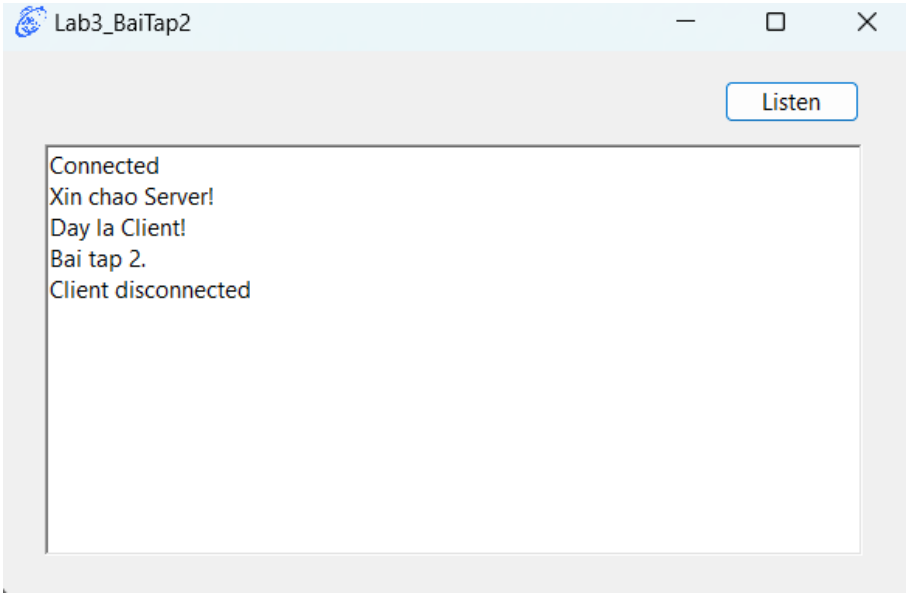
3. *Bài tập 2 – Chương trình TCP Server đơn giản:*

a) Tổng quan:



b) Chi tiết:

- Server mở kết nối Port 8080 khi sử dụng nút Listen và thông báo "Connected" khi kết nối thành công sau đó bắt đầu nhận thông điệp được nhập từ Terminal và sẽ thông báo "Disconnected" khi đóng Terminal hay ngắt kết nối.

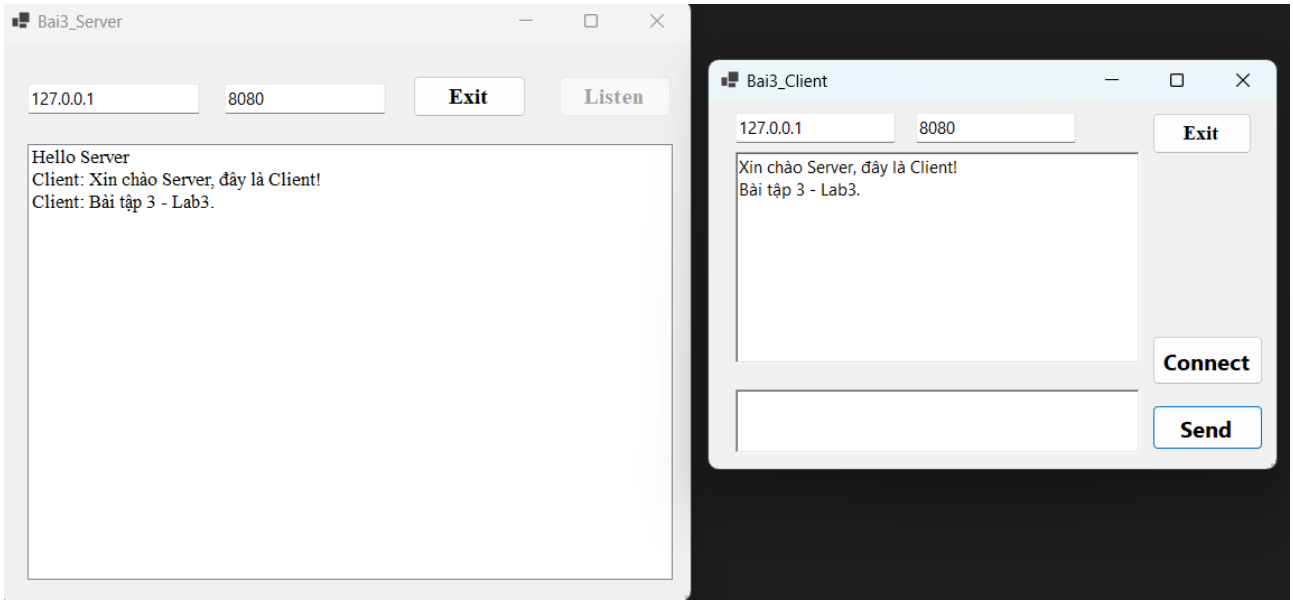


- Ở cửa sổ Terminal sử dụng lệnh telnet với địa chỉ IP loopback và Port 8080 để kết nối với Server, sau khi kết nối thành công có thể bắt đầu nhập thông điệp và ấn Enter để gửi.

```
private void btn_Listen_Click(object sender, EventArgs e)
{
    // Xử lý lỗi InvalidOperationException
    CheckForIllegalCrossThreadCalls = false;
    Thread serverThread = new Thread(new ThreadStart(ServerThread));
    serverThread.Start();
}
1 reference
void ServerThread()
{
    int bytesReceived = 0;
    byte[] recv = new byte[1];
    Socket clientSocket = null; // Khởi tạo socket client
    Socket listener = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
    IPEndPoint ipEnd = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 8080);
    listener.Bind(ipEnd);
    listener.Listen(-1);
    clientSocket = listener.Accept();
    rtb_Output.Text += "Connected\n";
    bool isConnected = true;
    while (clientSocket.Connected && isConnected)
    {
        string text = "";
        do
        {
            try
            {
                bytesReceived = clientSocket.Receive(recv);
                if (bytesReceived == 0) // Kiểm tra nếu client bị ngắt kết nối
                {
                    isConnected = false;
                    break; // Thoát khỏi vòng lặp
                }
                text += Encoding.ASCII.GetString(recv);
            }
            catch (InvalidOperationException ex)
            {
                // Xử lý lỗi InvalidOperationException
                Console.WriteLine("Error: " + ex.Message);
            }
        } while (text[text.Length - 1] != '\n');

        if (isConnected == false)
        {
            rtb_Output.Text += "Client disconnected\n";
            break;
        }
        else
        {
            rtb_Output.Text += text;
        }
```

4. *Bài tập 3 – Chương trình gửi nhận dữ liệu với TCP (1S-1C):*

a) Tổng quan:



b) Chi tiết:

- Server bắt đầu mở kết nối theo địa chỉ IP và Port xác định khi sử dụng nút Listen, nếu Client kết nối thành công sẽ hiển thị thông báo “Hello Server” và Server bắt đầu nhận thông điệp gửi từ Client.

```
public LAB3_BaiTap3_Server()
{
    InitializeComponent();
    Control.CheckForIllegalCrossThreadCalls = false;
    Connect();
}
Socket Client;
IPEndPoint IPEP;
TcpListener Listener;

1 reference
private void btn_Listen_Click(object sender, EventArgs e)
{
    btn_Listen.Enabled = false;
}
1 reference
void Connect()
{
    IPEP = new IPEndPoint(IPAddress.Any, 8080);
    Listener = new TcpListener(IPEP);
    Thread thread = new Thread(() =>
    {
        while (true)
        {
            Listener.Start();
            Client = Listener.AcceptSocket();
            Thread receive = new Thread(Receive);
            receive.IsBackground = true;
            receive.Start(Client);
        }
    });
    thread.IsBackground = true;
    thread.Start();
    void Receive(Object obj)
    {
        while (true)
        {
            Socket client = obj as Socket;
            byte[] recv = new byte[1000];
            Client.Receive(recv);
            string str = Encoding.UTF8.GetString(recv);
            Addmessage(str);

        }
    }
}
```

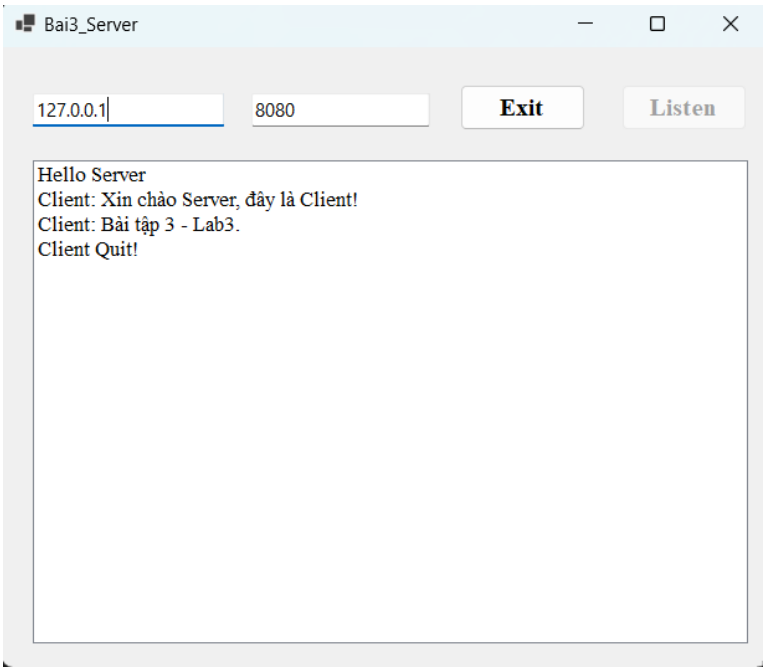
```
void Addmessage(string message)
{
    listView1.Items.Add(message);
}

1 reference
private void btn_Exit_Click(object sender, EventArgs e)
{
    this.Hide();
}

1 reference
private void Bai3_Server_FormClosed(object sender, FormClosedEventArgs e)
{
    this.Hide();
    Listener.Stop();
    Client.Close();
}
```

- Client sẽ kết nối với Server có IP và Port đã nhập khi sử dụng nút Connect, nhập thông điệp muốn gửi vào ô bên dưới và sử dụng nút Send để gửi đến Server.

```
public LAB3_BaiTap3_Client()
{
    InitializeComponent();
    Control.CheckForIllegalCrossThreadCalls = false;
}
TcpClient tcpClient;
IPEndPoint iPEndPoint;
NetworkStream stream;
1 reference
private void btn_Connect_Click(object sender, EventArgs e)
{
    iPEndPoint = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 8080);
    tcpClient = new TcpClient();
    tcpClient.Connect(iPEndPoint);
    stream = tcpClient.GetStream();

    byte[] data = Encoding.UTF8.GetBytes("Hello Server" +
        " ");
    stream.Write(data, 0, data.Length);
}
1 reference
private void btn_Send_Click(object sender, EventArgs e)
{
    try
    {
        // Lấy tin nhắn tùy ý từ rtb_message
        string message = rtb_Input.Text;
        rtb_Input.Clear();
        rtb_Output.Text += message + "\n";
        byte[] data = Encoding.UTF8.GetBytes("Client: " + message + "\n");
        stream.Write(data, 0, data.Length);
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message);
    }
}
```

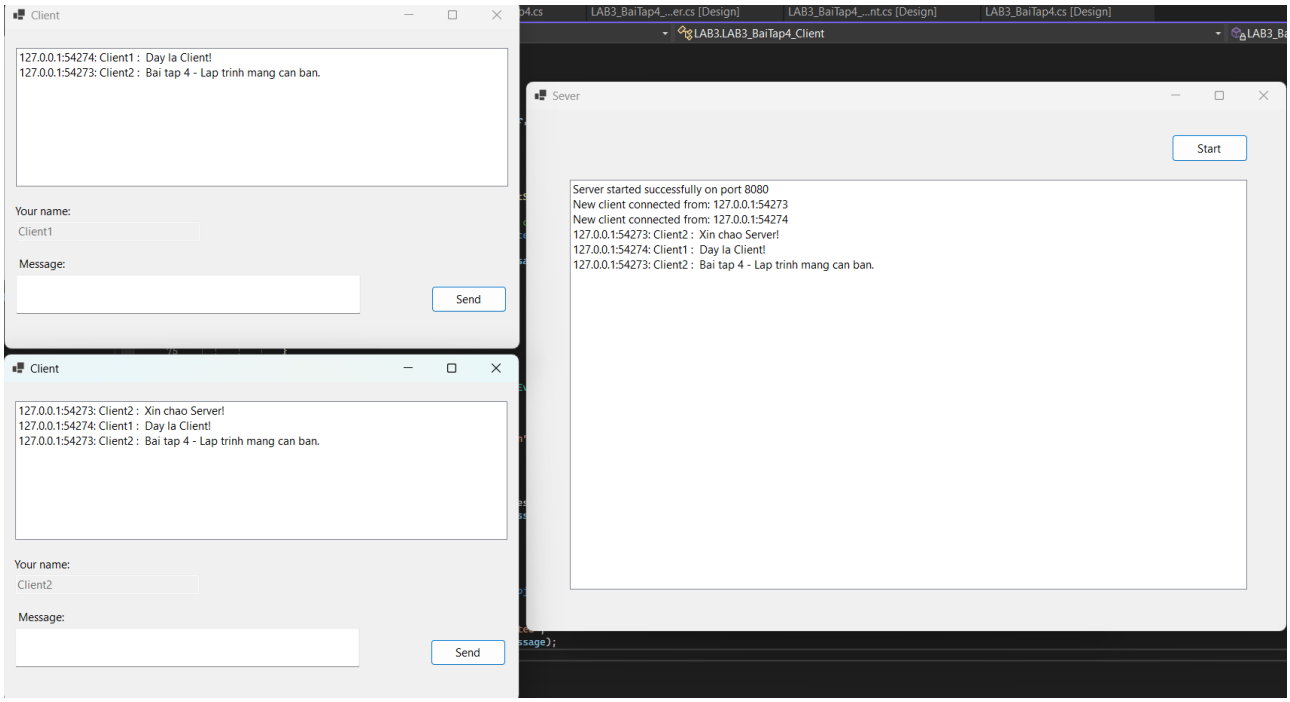
```
private void btn_exit_Click(object sender, EventArgs e)
{
    try
    {
        Byte[] data = System.Text.Encoding.UTF8.GetBytes("Client Quit!\n");
        stream.Write(data, 0, data.Length);
        this.Hide();
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message);
    }
}

1 reference
private void Bai3_Client_FormClosed(object sender, FormClosedEventArgs e)
{
    Byte[] data = System.Text.Encoding.UTF8.GetBytes("Quit\n");
    stream.Write(data, 0, data.Length);
    this.Hide();
}
```

- Khi muốn kết thúc kết nối, sử dụng nút Exit ở form Bai3_Client và Server sẽ nhận thông báo “Client Quit!”.



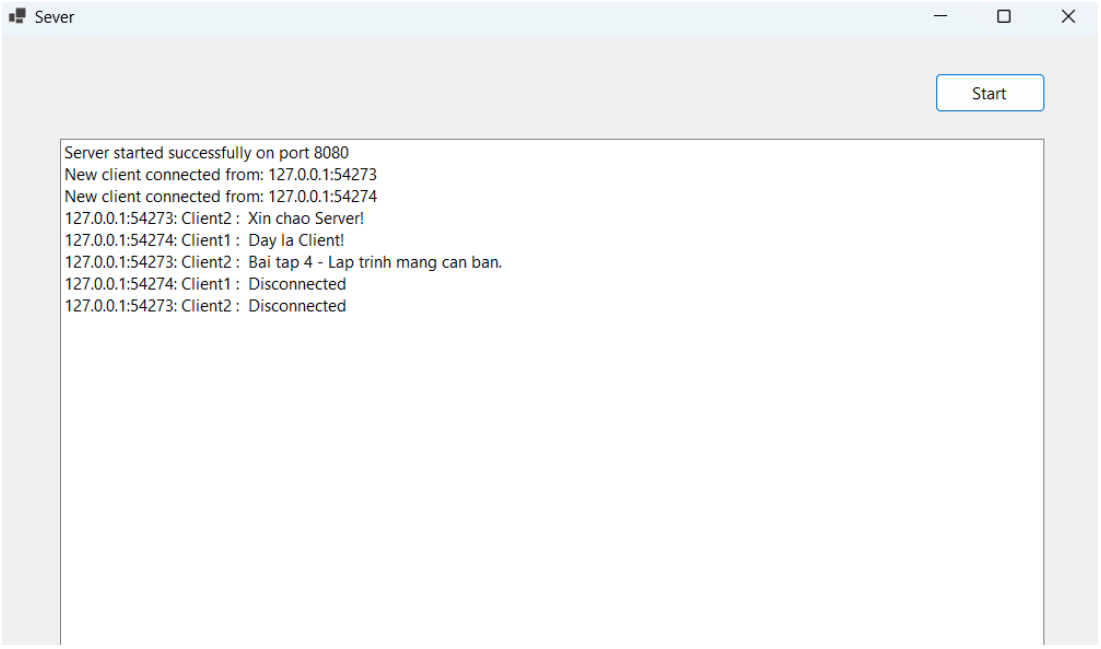
5. *Bài tập 4 – Chương trình Chatroom đơn giản (1S-nC):*

a) Tổng quan:



b) Chi tiết:

- Server mở kết nối khi sử dụng nút Start, nhận thông báo khi có Client kết nối thành công, nhận thông điệp và hiển thị ở ListView, khi đóng cửa sổ Client thì Server nhận thông báo Client nào đã ngắt kết nối.



```
public LAB3_BaiTap4_Server()
{
    InitializeComponent();
}
private TcpListener server;
private Dictionary<TcpClient, string> clients = new Dictionary<TcpClient, string>(); // Lưu trữ client và tên phòng

1 reference
private void buttonStart_Click(object sender, EventArgs e)
{
    try
    {
        server = new TcpListener(IPAddress.Any, 8080);
        server.Start();
        Thread listenThread = new Thread(ListenForClients);
        listenThread.Start();

        // Update form to show server started successfully
        Invoke((MethodInvoker)delegate
        {
            listBoxMessage.Items.Add($"Server started successfully on port 8080");
        });
    }
    catch (Exception ex)
    {
        MessageBox.Show("Có lỗi khi sử dụng port này " + ex);
    }
}
1 reference
private void ListenForClients()
{
    while (true)
    {
        try
        {
            TcpClient client = server.AcceptTcpClient();
            Thread clientThread = new Thread(() => HandleClient(client));
            clientThread.Start();
        }
        catch (SocketException ex)
        {
            break;
        }
    }
}
```

```
private void HandleClient(TcpClient client)
{
    NetworkStream stream = client.GetStream();
    byte[] buffer = new byte[1024];
    int bytesRead;
    IPEndPoint clientEndPoint = (IPEndPoint)client.Client.RemoteEndPoint;
    Invoke((MethodInvoker)delegate
    {
        listBoxMessage.Items.Add($"New client connected from: {clientEndPoint}");
    });
    try
    {
        while ((bytesRead = stream.Read(buffer, 0, buffer.Length)) > 0)
        {
            string info = Encoding.UTF8.GetString(buffer, 0, bytesRead);
            string[] username = info.Split('~');
            if (!clients.ContainsKey(client)) clients.Add(client, username[0]);
            string message = username[1];
            // Nhắn tin dạng IPEndPoint + UserName + Message
            string formattedMessage = $"{clientEndPoint}: {username[0]}: {message}";
            Invoke((MethodInvoker)delegate
            {
                listBoxMessage.Items.Add(formattedMessage);
            });
            BroadcastMessage(formattedMessage);
        }
    }
    catch (Exception ex)
    {
        //MessageBox.Show("Lỗi: " + ex.Message);
        clients.Remove(client);
    }
    finally
    {
        client.Close();
    }
}



private void BroadcastMessage(string message)
{
    byte[] data = Encoding.UTF8.GetBytes(message);
    foreach (var pair in clients)
    {
        pair.Key.GetStream().Write(data, 0, data.Length);
    }
}
```

```
private void SendToClient(TcpClient client, string message)
{
    byte[] data = Encoding.UTF8.GetBytes(message);
    client.GetStream().Write(data, 0, data.Length);
}


private void LAB3_BaiTap4_Server_FormClosed(object sender, FormClosedEventArgs e)
{
    // Tắt Server
    server.Stop();
}
```

- Client sẽ thực hiện kết nối với Server khi sử dụng nút Send với tên hiển thị là tên đã nhập theo địa chỉ IP và Port đã cài đặt sẵn, thông điệp sẽ được gửi và hiển thị chung ở cửa sổ Server cũng như các Client khác đang kết nối.

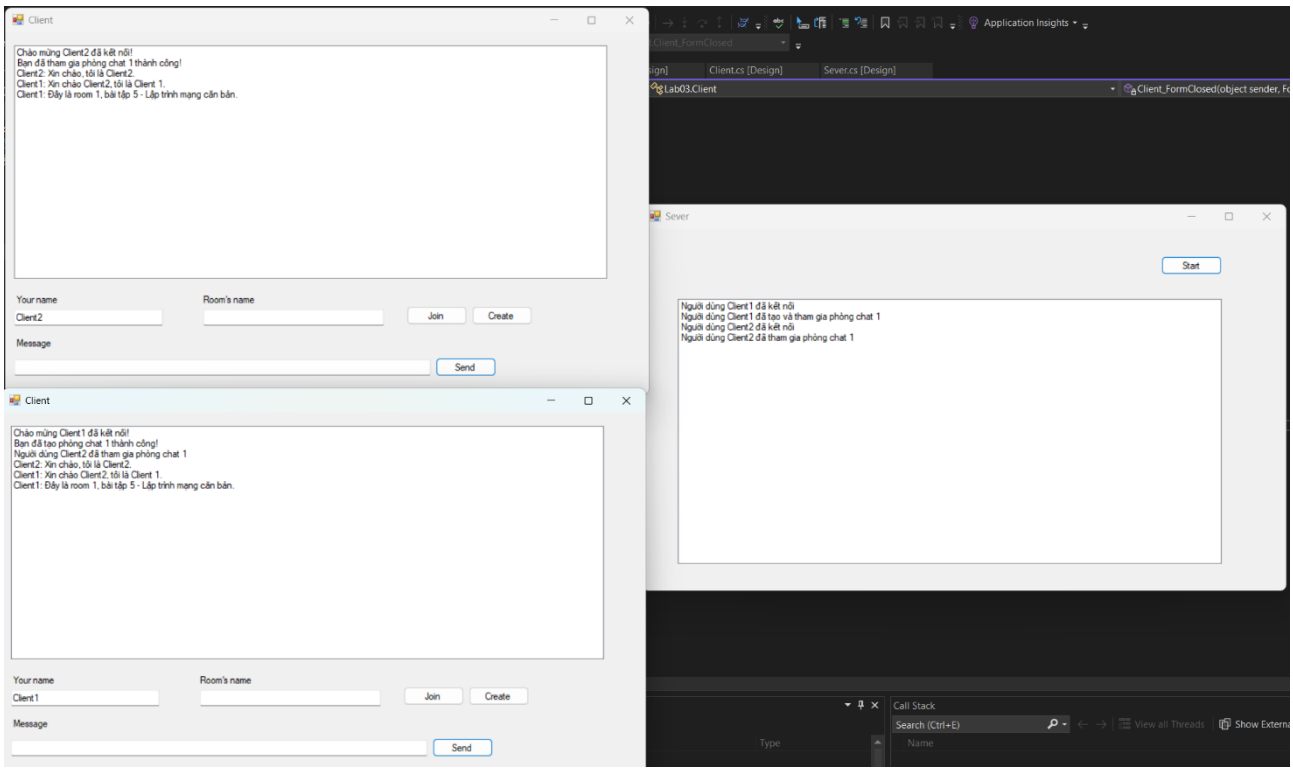
```
public partial class LAB3_BaiTap4_Client : Form
{
    private string username;
    private TcpClient client;
    private NetworkStream stream;
    private BackgroundWorker bgWorker;

    public LAB3_BaiTap4_Client()
    {
        InitializeComponent();
        bgWorker = new BackgroundWorker();
        bgWorker.DoWork += BgWorker_DoWork;
        ConnectToServer("localhost", 8080);

        // Prompt user for a username
        //username = Microsoft.VisualBasic.Interaction.InputBox("Enter your name:", "Username", "Guest");
        if (!string.IsNullOrEmpty(username) && stream != null)
        {
            byte[] buffer = Encoding.UTF8.GetBytes(username);
            stream.Write(buffer, 0, buffer.Length);
        }
    }

    private void ConnectToServer(string server, int port)
    {
        try
        {
            client = new TcpClient(server, port);
            stream = client.GetStream();
            bgWorker.RunWorkerAsync(); // Bắt đầu nhận dữ liệu từ server
        }
        catch (Exception ex)
        {
            MessageBox.Show("Lỗi kết nối: " + ex.Message);
        }
    }
```

```
private void BgWorker_DoWork(object sender, DoWorkEventArgs e)
{
    byte[] buffer = new byte[1024];
    while (true)
    {
        try
        {
            int bytesRead = stream.Read(buffer, 0, buffer.Length);
            if (bytesRead == 0)
            {
                // Kết nối bị ngắt
                break;
            }
            string message = Encoding.UTF8.GetString(buffer, 0, bytesRead);

            // Cập nhật giao diện trên thread chính
            this.Invoke((MethodInvoker)delegate
            {
                listBoxMessages.Items.Add(message);
            });
        }
        catch (Exception ex)
        {
            break;
        }
    }
}
1 reference
private void buttonSend_Click(object sender, EventArgs e)
{
    if (nameTb.Text.Trim() == "")
    {
        MessageBox.Show("Vui lòng nhập tên bạn");
        return;
    }
    nameTb.Enabled = false;
    username = nameTb.Text.ToString();
    string message = $"{username} ~ {textBoxMessage.Text}";
    byte[] buffer = Encoding.UTF8.GetBytes(message);
    stream.Write(buffer, 0, buffer.Length);
    textBoxMessage.Clear();
}
```

```
1 reference
private void LAB3_BaiTap4_Client_FormClosed(object sender, FormClosedEventArgs e)
{
    //Thông báo Client Disconnected
    string message = $"{username} ~ Disconnected";
    byte[] buffer = Encoding.UTF8.GetBytes(message);
    stream.Write(buffer, 0, buffer.Length);
    stream.Close();
    client.Close();
}
```

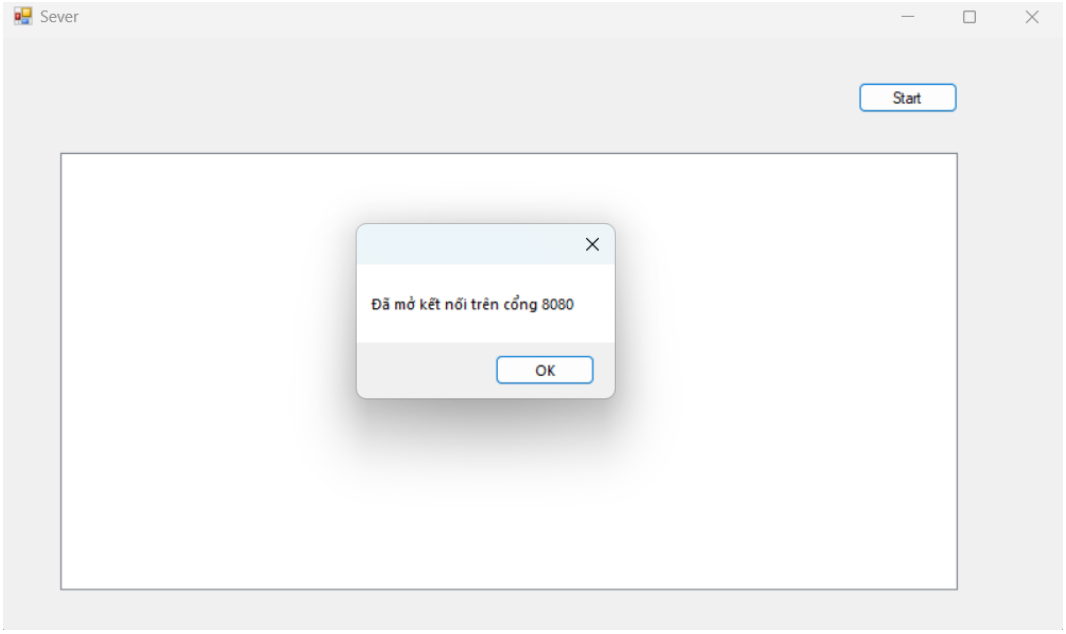
6. *Bài tập 5 – Chương trình Chatroom nâng cao:*

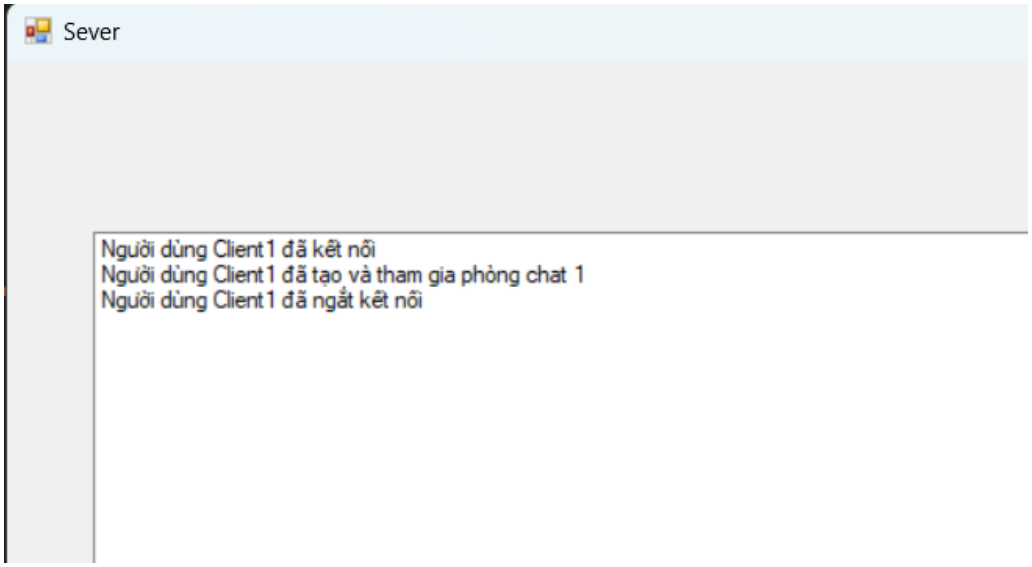
a) Tổng quan:



b) Chi tiết:

- Server mở kết nối và thông báo cho người dùng khi sử dụng nút Start, hiển thị thông báo Client nào đã kết nối đến Server cũng như thông tin phòng và thông tin Client nào vừa tạo/tham gia phòng đó. Khi đóng cửa sổ Client, Server hiển thị thông báo Client nào đã ngắt kết nối.






```
public partial class Sever : Form
{

    public Sever()
    {
        InitializeComponent();
    }

    private TcpListener server;
    private Dictionary<TcpClient, string> clients = new Dictionary<TcpClient, string>(); // Lưu trữ client và tên phòng
    private Dictionary<string, List<TcpClient>> rooms = new Dictionary<string, List<TcpClient>>();


    private void buttonStart_Click(object sender, EventArgs e)
    {

        try
        {
            server = new TcpListener(IPAddress.Any, 8080);
            server.Start();
            Thread listenThread = new Thread(ListenForClients);
            listenThread.Start();
            MessageBox.Show("Đã mở kết nối trên cổng " + 8080);
        }
        catch (Exception ex)
        {
            MessageBox.Show("Có lỗi khi sử dụng port này " + ex);
        }

    }

    private void ListenForClients()
    {
        while (true)
        {
            TcpClient client = server.AcceptTcpClient();
            Thread clientThread = new Thread(() => HandleClient(client));
            clientThread.Start();
        }
    }
```

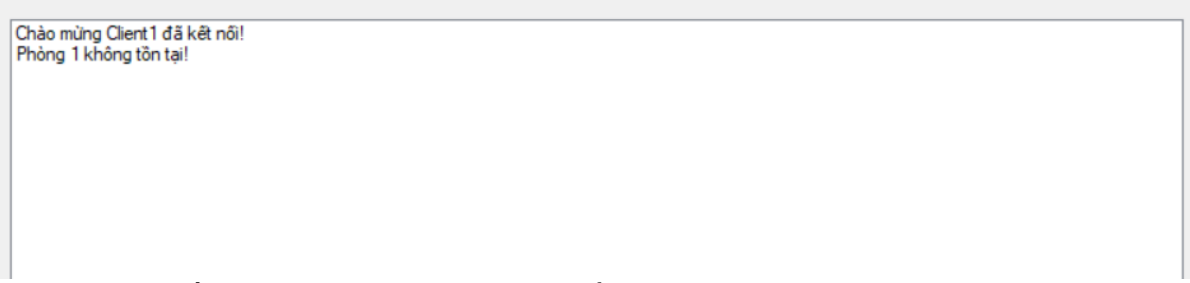
```
private void HandleClient(TcpClient client)
{
    NetworkStream stream = client.GetStream();
    byte[] buffer = new byte[1024];
    int bytesRead;
    try
    {
        // Nhận tên người dùng từ client
        bytesRead = stream.Read(buffer, 0, buffer.Length);
        string username = Encoding.UTF8.GetString(buffer, 0, bytesRead);
        // Lưu trữ tên người dùng và client vào dictionary
        clients.Add(client, username);

        // Gửi thông báo chào mừng đến client
        string welcomeMessage = $"Chào mừng {username} đã kết nối!";
        byte[] welcomeData = Encoding.UTF8.GetBytes(welcomeMessage);
        stream.Write(welcomeData, 0, welcomeData.Length);
        Invoke((MethodInvoker)delegate
        {
            listBoxMessage.Items.Add($"Người dùng {username} đã kết nối");
        });

        // Vòng lặp để nhận và gửi tin nhắn
        while ((bytesRead = stream.Read(buffer, 0, buffer.Length)) > 0)
        {
            string message = Encoding.UTF8.GetString(buffer, 0, bytesRead);
            if (message == "/leaveroom")
            {
                // Xóa client khỏi phòng
                clients.Remove(client);

                // Gửi thông báo đến các client còn lại trong phòng
                byte[] messageJoin = Encoding.UTF8.GetBytes($"Người dùng {username} đã ngắt kết nối");
                foreach (var pair in clients)
                {
                    if (pair.Value == clients[client])
                    {
                        pair.Key.GetStream().Write(messageJoin, 0, messageJoin.Length);
                    }
                }
                Invoke((MethodInvoker)delegate
                {
                    listBoxMessage.Items.Add($"Người dùng {username} đã ngắt kết nối");
                });
            }
```

```
else if(message == "/login")
{
    SendToClient(client, "/aceptlogin");
}
else if (message.StartsWith("/joinRoom"))
{
    // Lấy tên phòng chat từ tin nhắn (ví dụ: /createRoom room1)
    string[] parts = message.Split(' ');
    if (parts.Length > 1)
    {
        string roomName = parts[1];

        // Kiểm tra phòng chat có tồn tại
        if (!rooms.ContainsKey(roomName))
        {
            // Thông báo khi phòng không tồn tại
            SendToClient(client, $"Phòng {roomName} không tồn tại!");
        }
        else
        {
            // Thêm client vào phòng
            rooms[roomName].Add(client);
            clients[client] = roomName;

            //Thông báo cho các thành viên khác trong phòng
            SendToClient(client, $"Bạn đã tham gia phòng chat {roomName} thành công!");
            byte[] messageJoin = Encoding.UTF8.GetBytes($"Người dùng {username} đã tham gia phòng chat {roomName}");
            foreach (var pair in clients)
            {
                if (pair.Value == clients[client] && pair.Key != client)
                {
                    pair.Key.GetStream().Write(messageJoin, 0, messageJoin.Length);
                }
            }
            Invoke((MethodInvoker)delegate
            {
                listBoxMessage.Items.Add($"Người dùng {username} đã tham gia phòng chat {roomName}");
            });
        }
    }
}
```

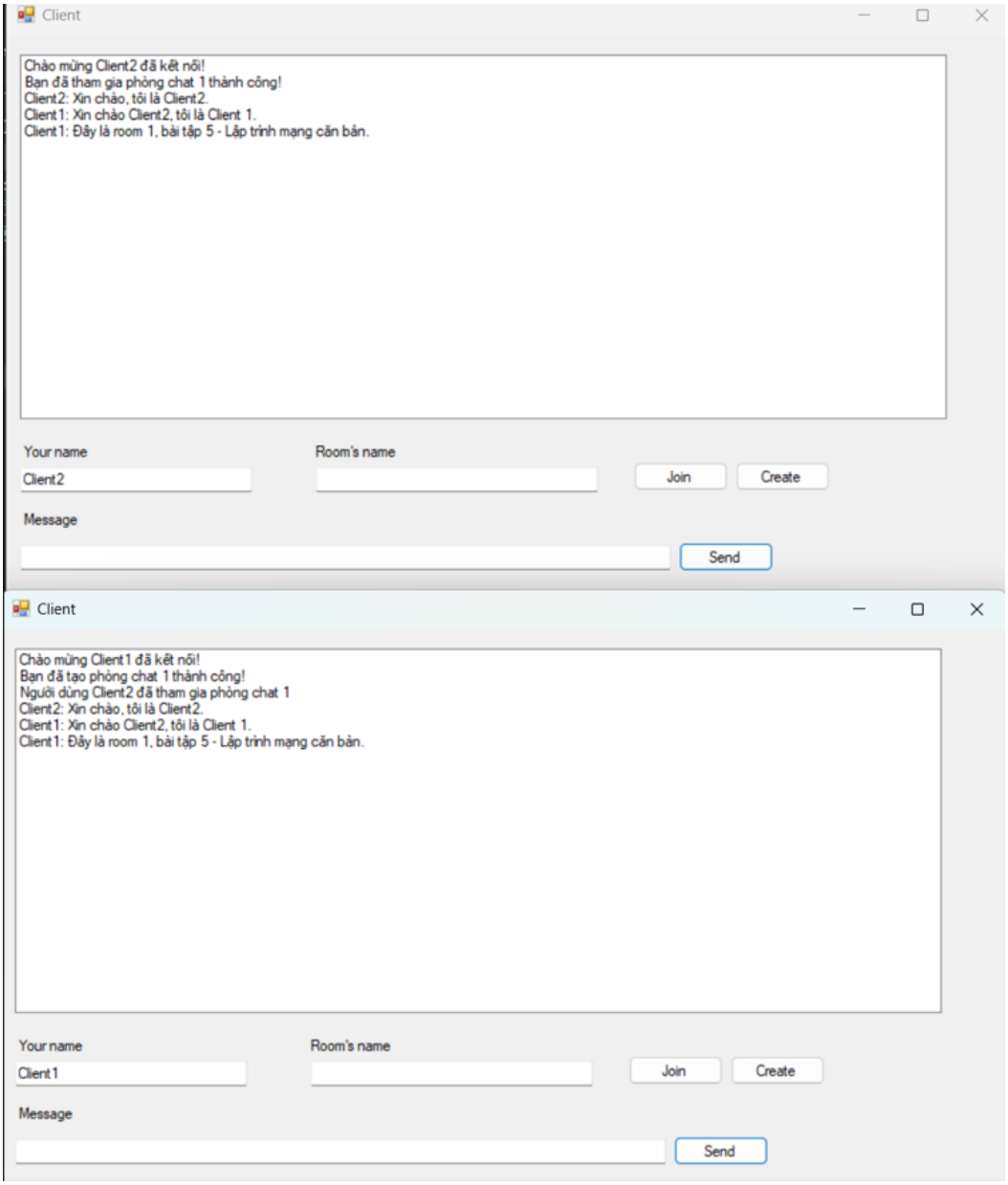
```
else if (message.StartsWith("/createRoom"))
{
    // Lấy tên phòng chat từ tin nhắn (ví dụ: /createRoom room1)
    string[] parts = message.Split(' ');
    if (parts.Length > 1)
    {
        string roomName = parts[1];

        // Kiểm tra xem phòng chat đã tồn tại chưa
        if (!rooms.ContainsKey(roomName))
        {
            // Tạo một phòng chat mới
            List<TcpClient> newRoom = new List<TcpClient>();
            newRoom.Add(client);
            rooms.Add(roomName, newRoom);
            // Thêm client vào room
            rooms[roomName].Add(client);
            clients[client] = roomName;

            // Gửi thông báo thành công
            SendToClient(client, $"Bạn đã tạo phòng chat {roomName} thành công!");
            Invoke((MethodInvoker)delegate
            {
                listBoxMessage.Items.Add($"Người dùng {username} đã tạo và tham gia phòng chat {roomName}");
            });
        }
        else
        {
            // Gửi thông báo lỗi
            SendToClient(client, $"Phòng chat {roomName} đã tồn tại!");
        }
    }
    else
    {
        // Gửi thông báo lỗi định dạng tin nhắn
        SendToClient(client, "Định dạng lệnh tạo phòng không hợp lệ. Vui lòng dùng: /createRoom <tên_phòng>");
    }
}
else
{
    string mes = $"{username}: {message}";
    byte[] mesin4 = Encoding.UTF8.GetBytes(mes);
    foreach (var pair in clients)
    {
        if (pair.Value == clients[client])
        {
            pair.Key.GetStream().Write(mesin4, 0, mesin4.Length);
        }
    }
```

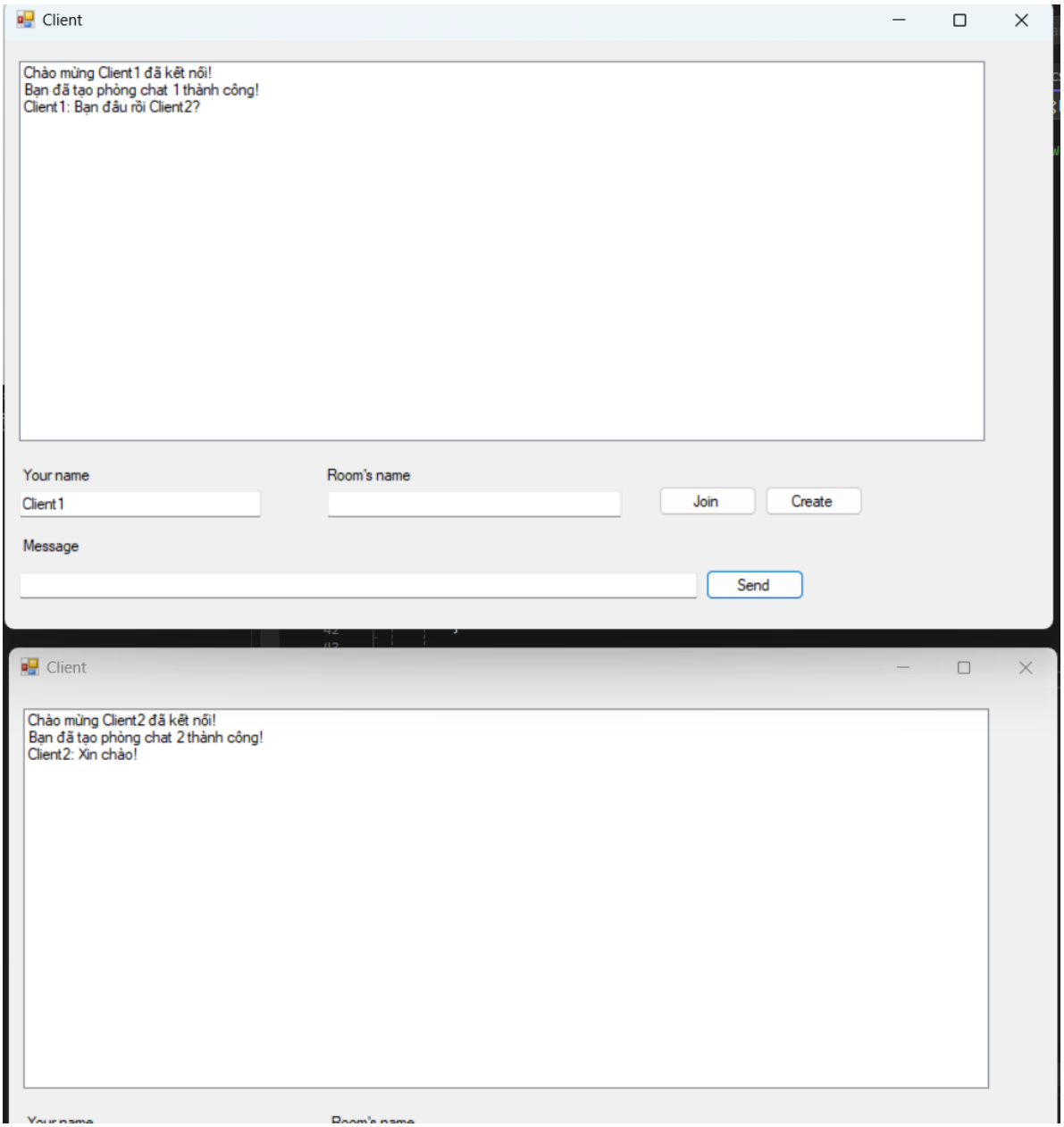
```
        }
        catch (Exception ex)
        {
            MessageBox.Show("Lỗi: " + ex.Message);
            // Gửi thông báo khi client ngắt kết nối
            string username = clients[client];
            BroadcastMessage($"Người dùng {username} đã rời khỏi phòng chat.");
            Invoke((MethodInvoker)delegate
            {
                listBoxMessage.Items.Add($"Người dùng {username} đã rời khỏi phòng chat.");
            });
            // Loại bỏ client khỏi danh sách
            clients.Remove(client);
        }
        finally
        {
            client.Close();
        }
    }
    1 reference
    private void BroadcastMessage(string message)
    {
        byte[] data = Encoding.UTF8.GetBytes(message);
        foreach (var pair in clients)
        {
            pair.Key.GetStream().Write(data, 0, data.Length);
        }
    }
    6 references
    private void SendToClient(TcpClient client, string message)
    {
        byte[] data = Encoding.UTF8.GetBytes(message);
        client.GetStream().Write(data, 0, data.Length);
    }
}
```

- Client thực hiện kết nối với Client với tên được khai báo qua địa chỉ IP và Port được cài đặt sẵn, Client có thể tạo chatroom với tên mong muốn và các Client khác có thể tham gia vào chatroom có sẵn khi nhập đúng tên chatroom, nếu không Client đó sẽ nhận thông báo chatroom không tồn tại.



- Tin nhắn giữa các Client sẽ được hiển thị qua lại thông qua ListView thuộc cửa sổ của các Client khác nhau ở cùng chatroom, nếu không tin nhắn sẽ không được hiển thị.




Client
Chào mừng Client1 đã kết nối!
Bạn đã tạo phòng chat 1 thành công!
Client1: Bạn đâu rồi Client2?
Your name
Client1
Room's name
Join
Create
Message
Send
Client
Chào mừng Client2 đã kết nối!
Bạn đã tạo phòng chat 2 thành công!
Client2: Xin chào!

```
public Client()
{
    InitializeComponent();
    bgWorker = new BackgroundWorker();
    bgWorker.DoWork += BgWorker_DoWork;

    //Kết nối đến server khi form load
    ConnectToServer("localhost", 8080);
}
1 reference
private void ConnectToServer(string server, int port)
{
    try
    {
        client = new TcpClient(server, port);
        stream = client.GetStream();
        bgWorker.RunWorkerAsync(); // Bắt đầu nhận dữ liệu từ server
    }
    catch (Exception ex)
    {
        MessageBox.Show("Lỗi kết nối: " + ex.Message);
    }
}

1 reference
private void BgWorker_DoWork(object sender, DoWorkEventArgs e)
{
    byte[] buffer = new byte[1024];
    while (true)
    {
        try
        {
            int bytesRead = stream.Read(buffer, 0, buffer.Length);
            if (bytesRead == 0)
            {
                // Kết nối bị ngắt
                break;
            }
            string message = Encoding.UTF8.GetString(buffer, 0, bytesRead);

            // Cập nhật giao diện trên thread chính
            this.Invoke((MethodInvoker)delegate
            {
                listBoxMessages.Items.Add(message);
            });
        }
        catch (Exception ex)
        {
            // Xử lý lỗi đọc dữ liệu
            MessageBox.Show("Lỗi nhận dữ liệu: " + ex.Message);
            break;
        }
    }
```

```
private void buttonSend_Click(object sender, EventArgs e)
{
    string message = textBoxMessage.Text;
    byte[] buffer = Encoding.UTF8.GetBytes(message);
    stream.Write(buffer, 0, buffer.Length);
    textBoxMessage.Clear();
}
0 references
private void buttonConnect_Click(object sender, EventArgs e)
{

}

1 reference
private void buttonJoin_Click(object sender, EventArgs e)
{
    string message = "/joinRoom " + textBoxRoomName.Text;
    byte[] buffer = Encoding.UTF8.GetBytes(message);
    stream.Write(buffer, 0, buffer.Length);
    textBoxRoomName.Clear();
}

1 reference
private void Client_FormClosed(object sender, FormClosedEventArgs e)
{
    // Gửi yêu cầu đến server
    string message = "/leaveroom";
    byte[] buffer = Encoding.UTF8.GetBytes(message);
    stream.Write(buffer, 0, buffer.Length);
}

1 reference
private void buttonCreateRoom_Click(object sender, EventArgs e)
{
    string message = "/createRoom " + textBoxRoomName.Text;
    byte[] buffer = Encoding.UTF8.GetBytes(message);
    stream.Write(buffer, 0, buffer.Length);
    textBoxRoomName.Clear();
}
```