Bộ môn An toàn Thông tin – Khoa MMT&TT
Trường Đại học Công nghệ Thông tin (UIT)

BÁO CÁO THỰC HÀNH

Môn học: Lập trình mạng căn bản

Kỳ báo cáo: Buổi 03 (Session 03)

Tên chủ đề:

GVHD: Nghi Hoàng Khoa

Ngày báo cáo: 03/04/2022

Nhóm: NH (ghi số thứ tự nhóm)

1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT106.M21.ATCL

STT	Họ và tên	MSSV	Email
1	Võ Khoa Nam	19521877	19521877@gm.uit.edu.vn
2	Trịnh Minh Hoàng	19521547	19521547@gm.uit.edu.vn
3	Phạm Trung Hiếu	19521512	19521512@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:[1]

STT	Công việc	Kết quả tự đánh giá
1	UDP Client – UDP Server	100%
2	Lắng nghe dữ liệu Telnet sử dụng TCP	100%
3	TCP Client – TCP Server	100%
4	Đọc và ghi file sử dụng BinaryFormatter	100%
5	Server MultiClient	100%

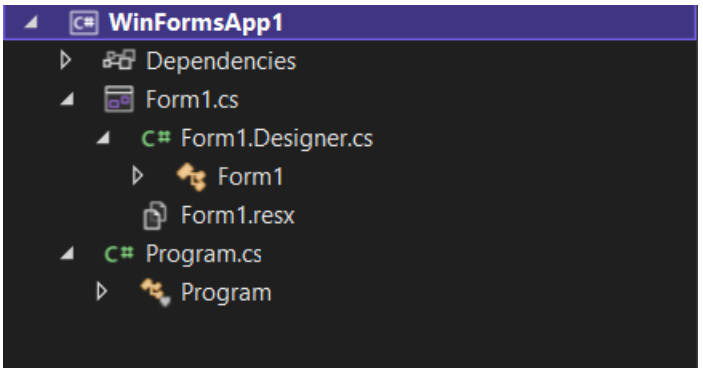
Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.

[1] Ghi nội dung công việc, các kịch bản trong bài Thực hành

BÁO CÁO CHI TIẾT

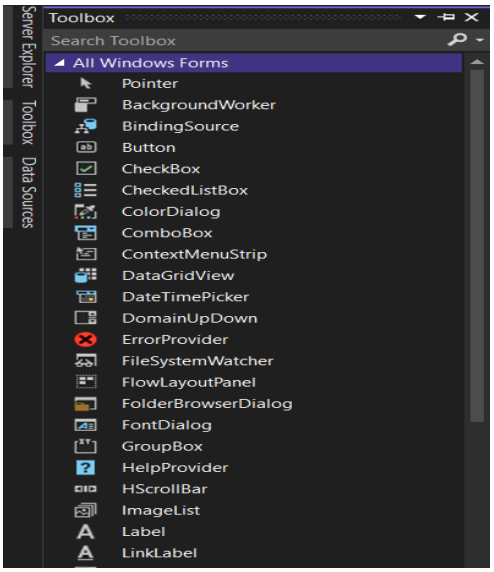
1. UDP Client – UDP Server

- Tài nguyên:
- Mô tả/mục tiêu:
  - Mô tả: ứng dụng gửi và nhận dữ liệu hai bên sử dụng giao thức UDP. Sử dụng IP và Port để kết nối
  - Mục tiêu: kết nối được Client – Server, thực hiện gửi, nhận tin và xuất ra màn hình.
- Các bước thực hiện/ Phương pháp thực hiện (Ảnh chụp màn hình, có giải thích)
  - Bước 1: Tạo project trong Visual Code bằng cách chọn File -> New -> Project -> Visual C# -> Windows Form Application. Đặt tên cho cho project.

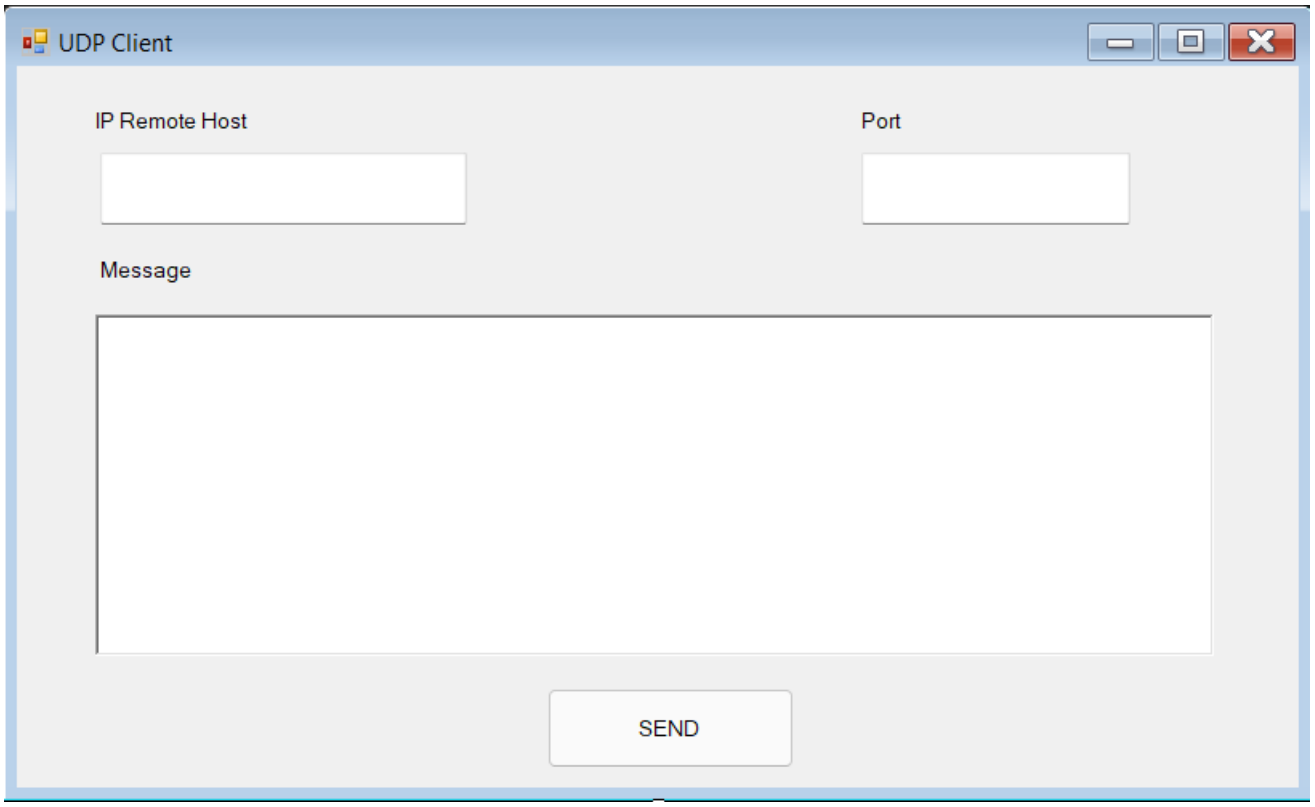


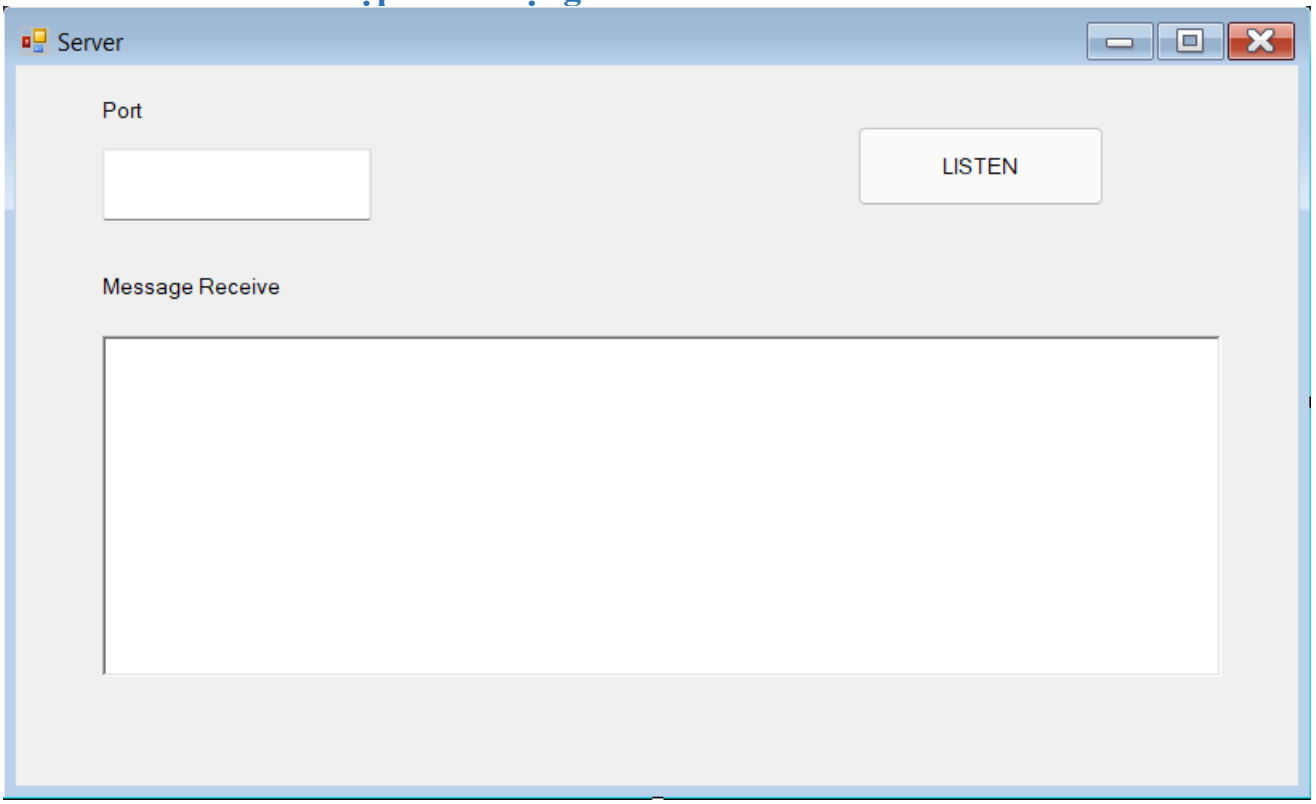
Hình 1: Sau khi tạo project sẽ được tổ chức gồm các file trong hình.

  - Bước 1: Tạo form giao diện theo yêu cầu bằng cách sử dụng toolbox trong Visual Studio. Sử dụng button và richBoxtext để tạo ra giao. Sử dụng button để làm nút khi nhấn vào sẽ thực hiện việc gửi và nhận dữ liệu.



Hình 2: Thanh toolbox chứa các thẻ label, input text, button,... Để tạo giao diện form





Hình 3: Giao diện windowsform UDP Client và UDP Server.

- Bước 2: Lập trình sự kiện và chức năng theo yêu cầu bài toán gửi và nhận dữ liệu.
  - Để gửi dữ liệu từ client cần click vào nút "SEND" trên giao diện. Vì vậy ta phải bắt sự kiện click cho button này.
  - Nhấp đúp chuột vào button "SEND". Chương trình sẽ khởi tạo 1 hàm tương ứng với sự kiện click vào button này.

```
private void button1_Click(object sender, EventArgs e)
{

}
```

Hình 4: Hàm khởi tạo sao khi click vào nút button tính tổng trên giao diện.

- Thêm code để thực hiện yêu cầu của bài toán vào

```
private void button1_Click(object sender, EventArgs e)
{
    Int32 port = Int32.Parse(textBox2.Text);
    IPAddress ip = IPAddress.Parse(textBox1.Text);
    IPEndPoint iped = new IPEndPoint(ip, port);

    byte[] data = UTF32Encoding.UTF32.GetBytes(richTextBox1.Text);

    try
    {
        int count = udpClient.Send(data, data.Length, iped);
        if (count > 0)
        {
            richTextBox1.Text = "";
        }
    }
    catch
    {
        MessageBox.Show("Error!", "Hoang ",MessageBoxButtons.OK, MessageBoxIcon.Error);
    }

}
```

Hinh 5: Đoạn mã thực hiện việc gửi dữ liệu sau khi nhấn vào button send.

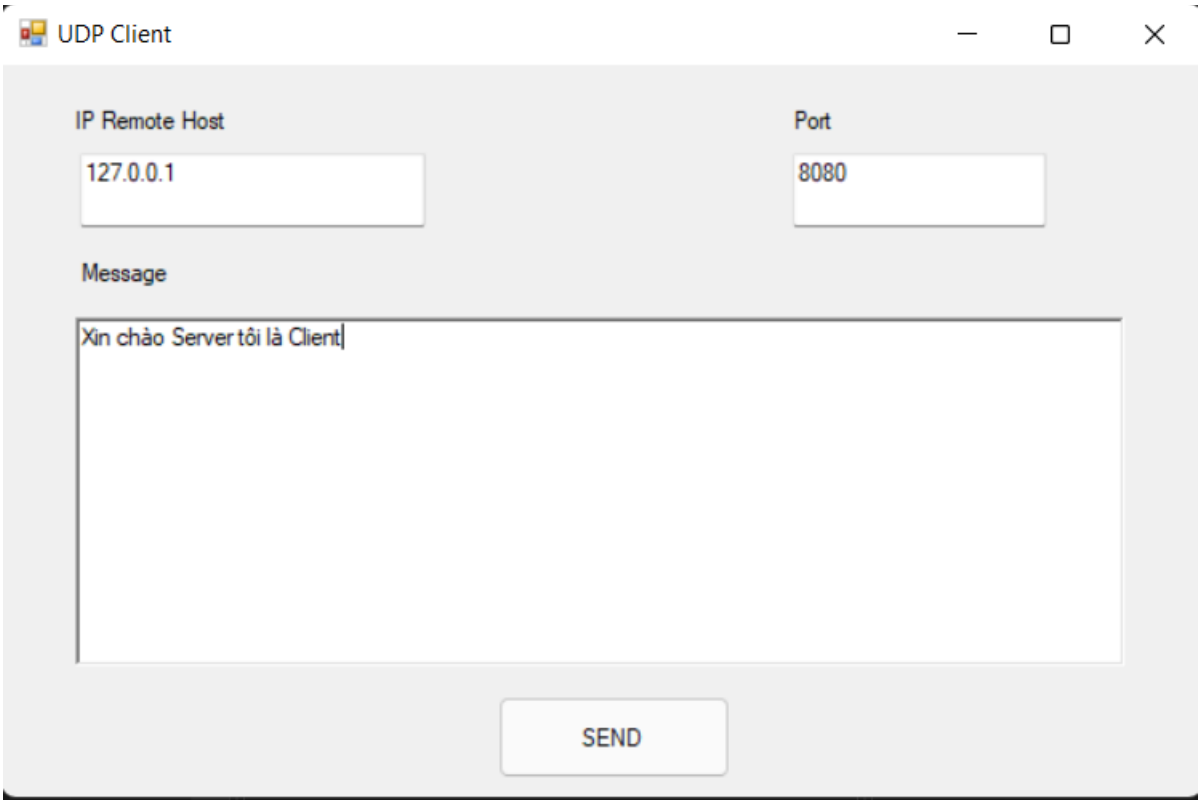
- Tiến hành khởi tạo biến "port" để lưu giá trị port từ textBox, "ip" lưu giá trị IP nhập vào của người dùng từ textBox
- Tiếp theo tiến hành khởi tao một mảng để lưu toàn bộ nội dung người dùng nhập từ richTextBox.
- Thực hiện việc gửi dữ liệu bằng giao thức udpClient thông qua hàm send, bao gồm dữ liệu (tin nhắn của người dùng nhập vào, địa chỉ ip và port).
- Kiểm tra nếu dữ liệu từ richTextBox > 0 có nghĩa là nếu richTextBox có nội dung thì tiến hành gửi và xoá nội dung đó trên richTextBox của Client.
- Để nhận dữ liệu tại server cần click vào button "Listen" tại window form của Server.
- Nhấp chuột vào nút "LISTEN" tại server chương trình bắt sự kiện và khởi tạo ra một hàm. Thêm mã vào hàm đó để server có thể nhận được tin nhắn.

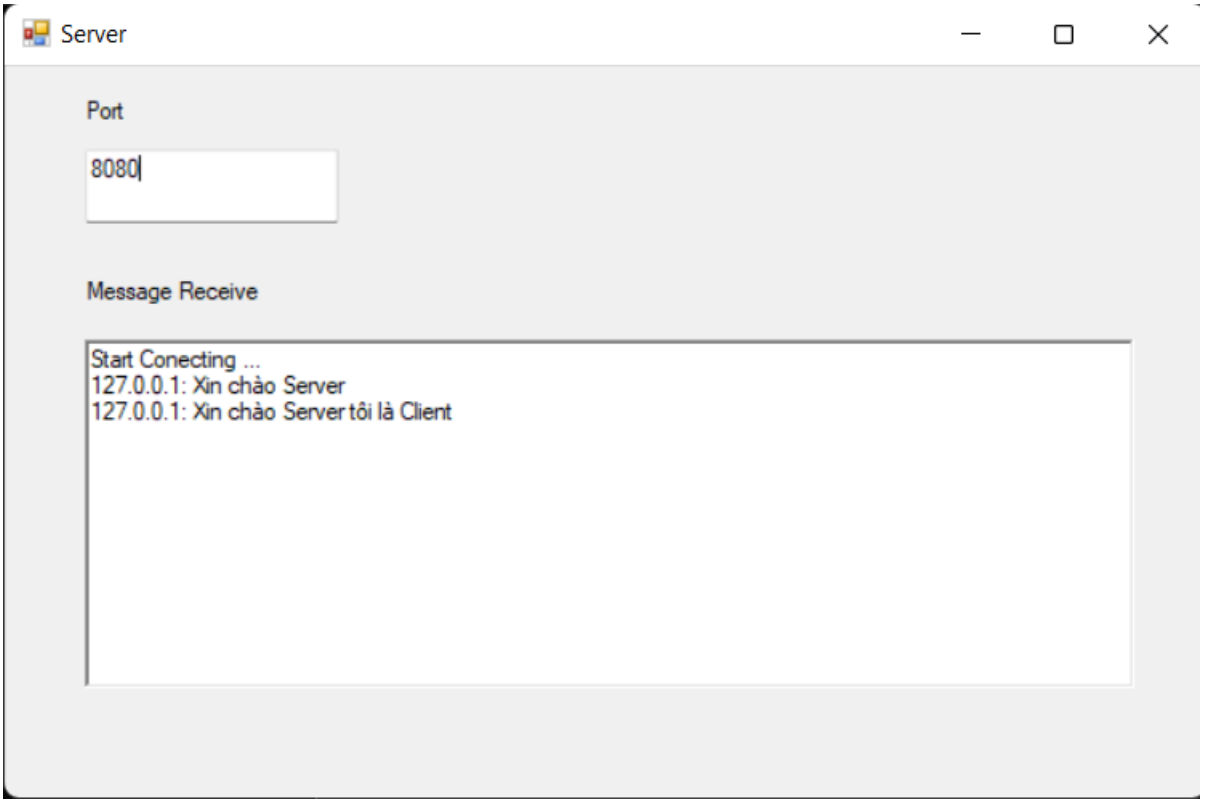
```
private void button1_Click(object sender, EventArgs e)
{
    Button btn = sender as Button;
    btn.Hide();
    richTextBox1.Text = "Start Conecting ..." + "\n";
    myDelegate = new ShowMessage(Show);
    Thread thread = new Thread(Receive);
    thread.IsBackground = true;
    thread.Start();
}

1 reference
public void Receive()
{
    while (true)
    {
        IPEndPoint iped = new IPEndPoint(IPAddress.Any, port);
        byte[] receiveIped = udpClient.Receive(ref iped);
        if (receiveIped.Length > 0)
        {
            string message = iped.Address.ToString() + ": " + UTF32Encoding.UTF32.GetString(receiveIped);
            this.Invoke(myDelegate, new object[] { message });
        }
    }
}
```

Hình 6: Hàm nhận tin và sự kiện khi click vào button LISTEN tại server

- Tiến hành xây dựng hàm Receive để nhận tin từ client.
- Sử dụng vòng lặp while nếu điều kiện đúng, sẽ khởi tạo một mảng để nhận dữ liệu gửi qua từ client kết nối cùng port.
- Sau đó tiến hành kiểm tra nếu độ dài của dữ liệu lớn hơn 0 (tức có tin nhắn) sẽ tiến hành lưu vào biến message và hiển thị ra màn hình.



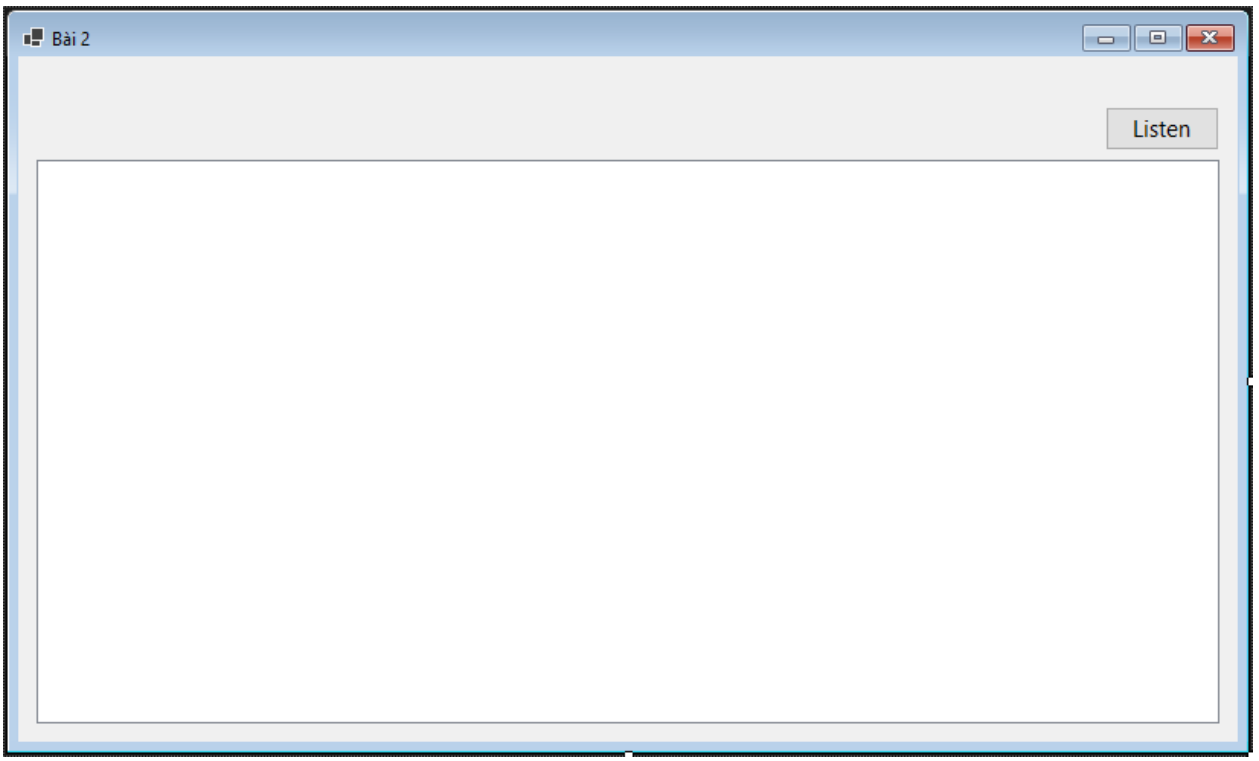


Hình 7: Kết quả sau khi chạy chương trình

2. Lắng nghe dữ liệu Telnet sử dụng TCP

- Tài nguyên:
- Mô tả/mục tiêu:
  - Mô tả: viết chương trình lắng nghư dữ liệu Telnet bằng TCP.
  - Mục tiêu: viết được chương trình lắng nghe dữ liệu từ dịch vụ Telnet sử dụng kết nối TCP(sử dụng lớp Socket)

- Các bước thực hiện/ Phương pháp thực hiện (Ảnh chụp màn hình, có giải thích)
  - Bước 1: Tạo project trong Visual Code bằng cách chọn File -> New -> Project -> Visual C# -> Windows Form Application. Đặt tên cho cho project.
  - Bước 2: Tạo form giao diện theo yêu cầu bằng cách sử dụng toolbox trong Visual Studio. Sử dụng button để tạo nút "Listen" sau khi chạy chương trình ta Click và nút Listen chương trình sẽ lắng nghe dữ liệu từ dihcj vụ Telner. Sử dụng ListView để hiện thị dữ liệu lắng nghe được từ Telnet.



Hình 9: Tạo form giao diện theo yêu cầu

- Bước 2: Lập trình sự kiện và chức năng theo yêu cầu
  - Để thực hiện lắng nghe và hiển thị các nội dung khi dữ liệu được gửi từ Telnet thì người dùng phải tiến hành click vào bút "Listen" vì vậy ta tiến hành bắt sự kiện khi người dùng click vào nút này.
  - Nhấp đúp chuột vào button "Listen". Chương trình sẽ khởi tạo 1 hàm tương ứng với sự kiện click vào button này.

```
1 reference
private void btnListen_Click(object sender, EventArgs e)
{

}
```

Hình 10: Hàm khởi tạo sao khi click vào nút button tính tổng trên giao diện.

- Thêm code để thực hiện yêu cầu của bài toán vào

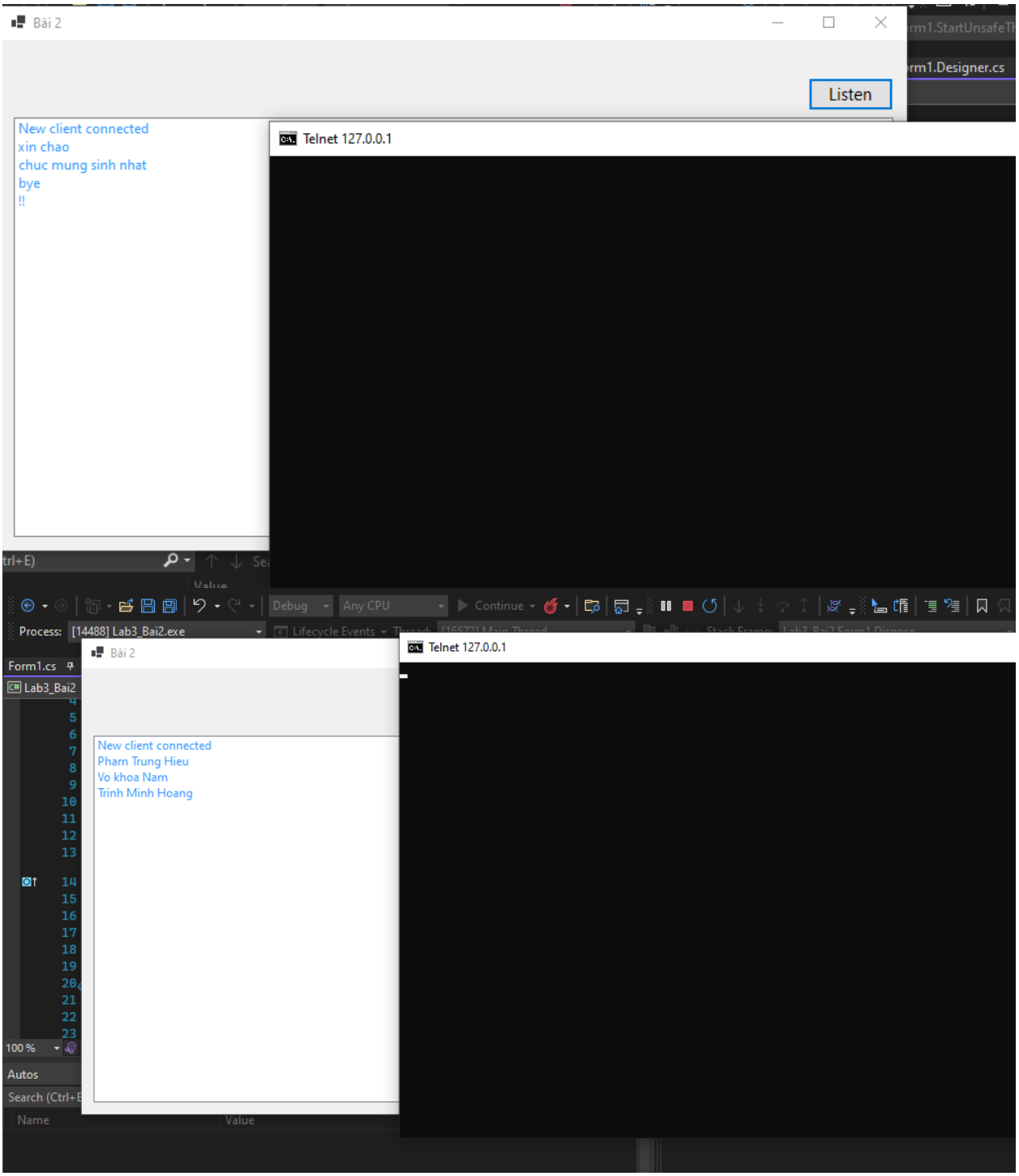
```
private void btnListen_Click(object sender, EventArgs e)
{
    CheckForIllegalCrossThreadCalls = false;
    Thread serverThread = new Thread(new ThreadStart(StartUnsafeThread));
    serverThread.Start();
}
1 reference
private void StartUnsafeThread()
{
    int bytesReceived = 0;
    byte[] recv = new byte[1];
    Socket clientSocket;
    Socket listenerSocket = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
    IPEndPoint ipepServer = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 8080);
    listenerSocket.Bind(ipepServer);
    listenerSocket.Listen(-1);
    clientSocket = listenerSocket.Accept();
    listView1.Items.Add(new ListViewItem("New client connected"));
    while(clientSocket.Connected)
    {
        string text = "";
        do
        {
            bytesReceived = clientSocket.Receive(recv);
            text += Encoding.ASCII.GetString(recv);

        }
        while (text[text.Length - 1] != '\n');
        listView1.Items.Add(new ListViewItem(text));

    }
    listenerSocket.Close();
}
```

Hình 11: Đoạn mã thực hiện yêu cầu

- Để telnet đến địa chỉ IP của máy tại cổng 8080, ta phải mở port 8080 lắng nghe kết nối TCP đến port 8080, khi nhất nút Listen để thực hiện lắng nghe kết nối tại địa chỉ IP của máy và cổng là 8080.
- Tạo socket bên nhận, socket này lắng nghe kết nối tới địa chỉ 127.0.0.1 và port 8080. Đây là 1 TCP/IP socket.
- AddresFamily.InterNetwork trả về địa chỉ của IPv4 hiện hành.
- SocketType kết nối socket sử dụng luồng Stream để nhận dữ liệu.
- ProtocolType: sử dụng giao thức TCP để kết nối.



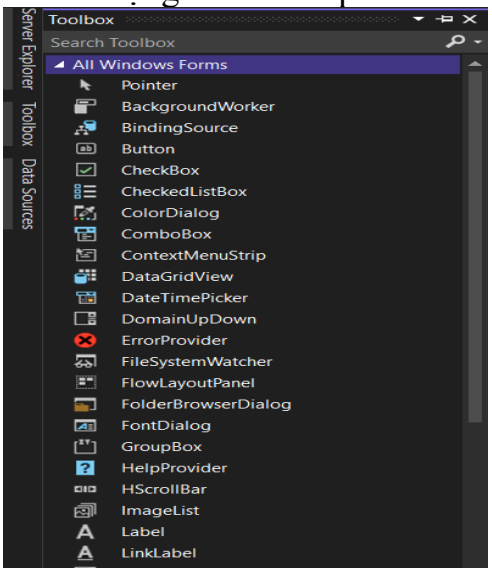
Hình 12: Kết quả hoạt động của chương trình

3. TCP Client – TCP Server

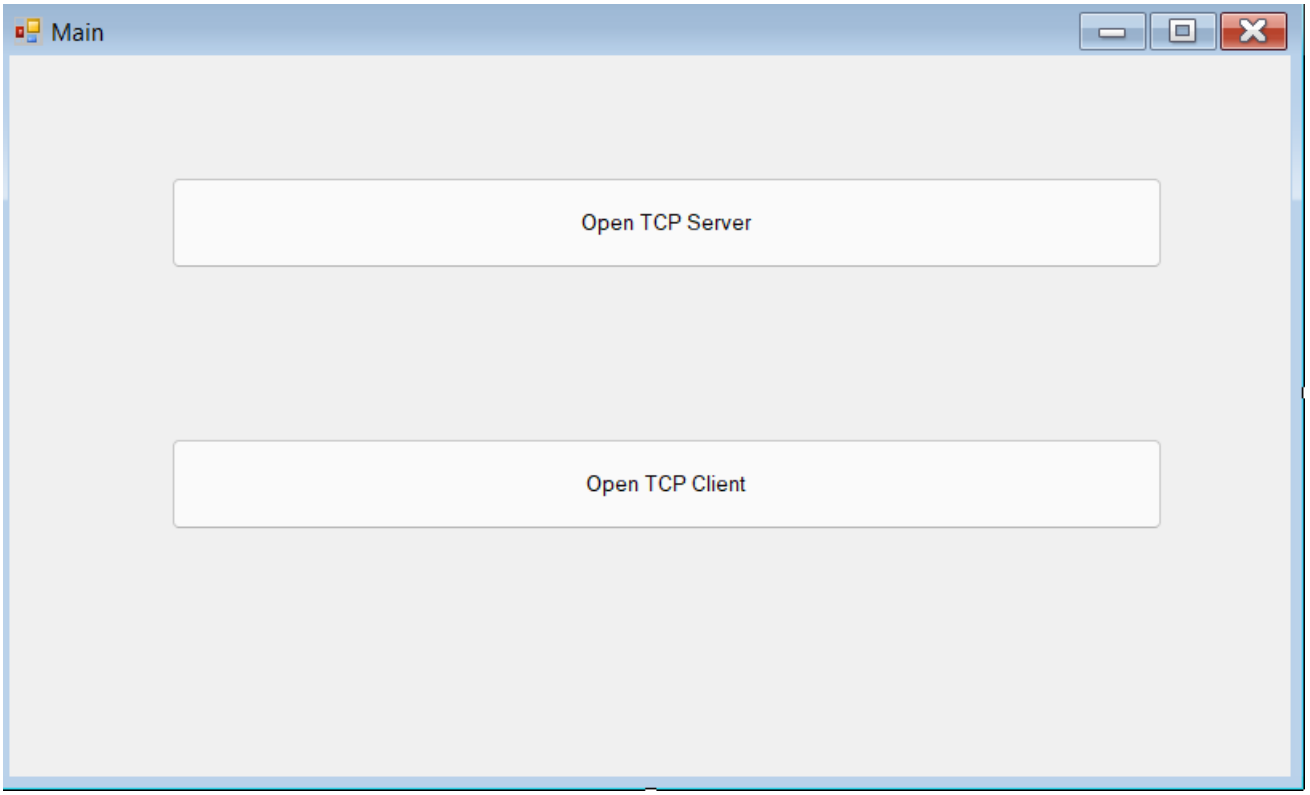
- Tài nguyên:
- Mô tả/mục tiêu:

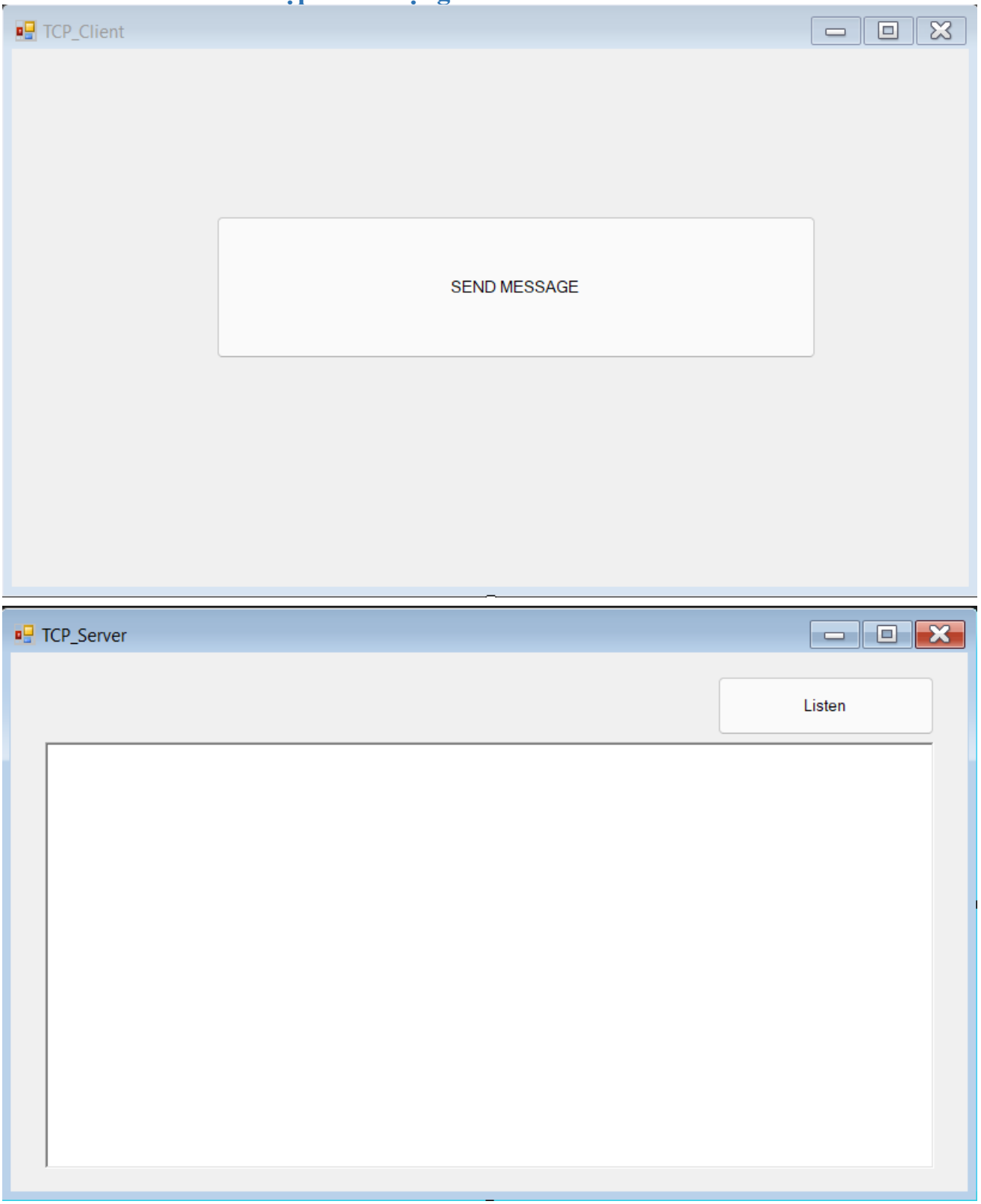
  - Mô tả: viết ứng dụng gửi và nhận dữ liệu sử dụng giao thức TCP. Server lắng nghe kết nối và thông điệp từ Client.
  - Mục tiêu: kết nối được Client và Server thông qua giao thức TCP. Gửi và nhận được tin nhắn từ Server và Client.

- Các bước thực hiện/ Phương pháp thực hiện (Ảnh chụp màn hình, có giải thích)
  - Bước 1: Bước 1: Tạo form giao diện theo yêu cầu bằng cách sử dụng toolbox trong Visual Studio. Sử dụng label và input text để tạo ra giao diện.



Hình 13: Thanh tool box chứa các thành phần để tạo giao diện form





Hình 14: Giao diện windows form khởi tạo theo yêu cầu

- Bước 2: Lập trình sự kiện và chức năng theo yêu cầu bài toán thực hiện các phép tính
  - Để thực hiện mở Server và Client ta thực hiện bắt sự kiện cho các button
  - Nhấp đúp chuột vào button "Open TCP Client " và "Open TCP Server". Chương trình sẽ khởi tạo 1 hàm tương ứng với sự kiện click vào button này.

```
private void button1_Click(object sender, EventArgs e)
{


}
```

Hình 15: Hàm được khởi tạo sao khi click vào nút đọc file

- Bắt sự kiện cho nút "Open TCP Server" và "Open TCP Client"..

```
1 reference
private void button1_Click(object sender, EventArgs e)
{
    Form tcpServer = new TCP_Server();
    tcpServer.Show();
}

1 reference
private void button2_Click(object sender, EventArgs e)
{
    Form tcpClient = new TCP_Client();
    tcpClient.Show();
}
```

Hình 16: Cliick vào nút để mở giao diện server và client

- Bước 3: Thực hiện xây dựng Client và Server

```
private void button1_Click(object sender, EventArgs e)
{
    TcpClient tcpClient = new TcpClient();

    IPAddress ipAdress = IPAddress.Parse("127.0.0.1");
    IPEndPoint ipEndpoint = new IPEndPoint(ipAdress, 8080);
    tcpClient.Connect(ipEndpoint);

    NetworkStream ns = tcpClient.GetStream();

    byte[] data = System.Text.Encoding.ASCII.GetBytes("Hello server\n");
    ns.Write(data, 0, data.Length);

    tcpClient.Close();


}
```

Hình 17: Nút gửi tin bên Client

- Thực hiện bắt sự kiện sau khi click vào nút gửi tin nhắn tại client.
- Tiến hành khởi tạo các biến để lưu các giá trị gồm IP của server và port để kết nối với server
- Mảng để lưu tin nhắn gửi "Hello server"

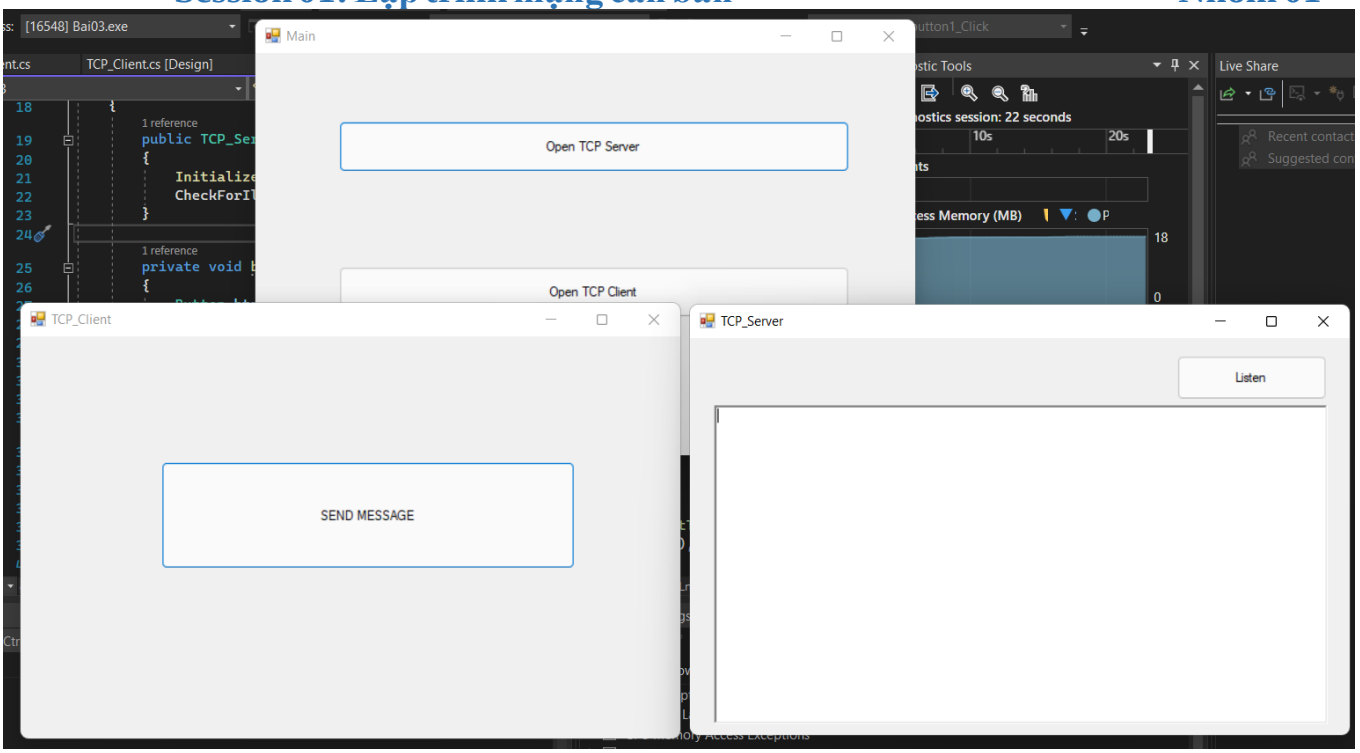
```
private void StartUnsafeThread()
{
    byte[] recv = new byte[1024];
    Socket client;
    Socket listen = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
    IPEndPoint ipep = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 8080);
    listen.Bind(ipep);
    listen.Listen(-1);
    client = listen.Accept();
    richTextBox1.Text += "New client onnected" + "\n";
    while (client.Connected)
    {
        client.Receive(recv);
        string s = Encoding.UTF8.GetString(recv);
        richTextBox1.Text += s + "\n";
    }
    listen.Close();
}
```

```
private void button1_Click(object sender, EventArgs e)
{
    Button btn = sender as Button;
    btn.Hide();
    richTextBox1.Text = "Server running on 127.0.0.1:8080" + "\n";
    Thread thread = new Thread(new ThreadStart(StartUnsafeThread));
    thread.Start();
}
```

- Bên phía Server khởi tạo hàm để tiến hành kết nối và nhận tin nhắn từ client
- Sau khi nhận tin nhắn từ client tiến hành in ra màn hình.

- Bước 4:

- Khi chương trình hoạt động:

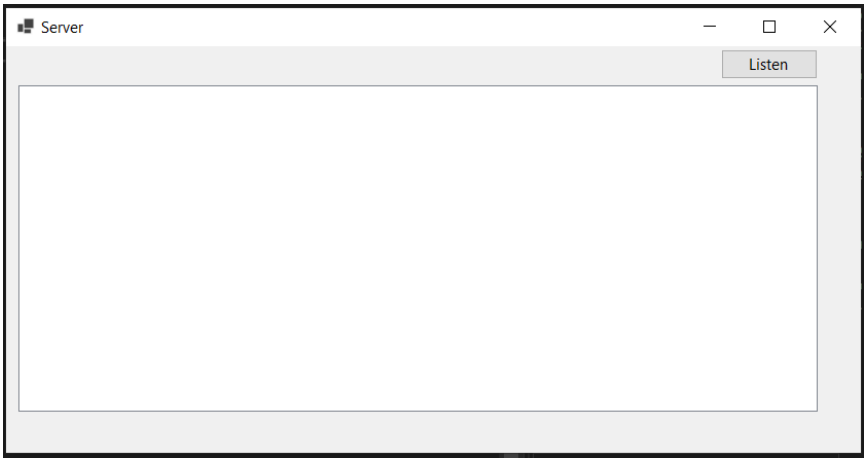


Hình 21: Kết quả chương trình hoạt động

4. Server MultiClient

- Tài nguyên:
- Mô tả/mục tiêu:
  - Server quản lý kết nối và broadcast tin nhắn đến các client khác cùng kết nối vào Server
  - Khi một người dùng gửi tin thì tất cả mọi người còn lại đề nhận được tin nhắn
- Các bước thực hiện/ Phương pháp thực hiện (Ảnh chụp màn hình, có giải thích)
  - Bước 1: Tạo form

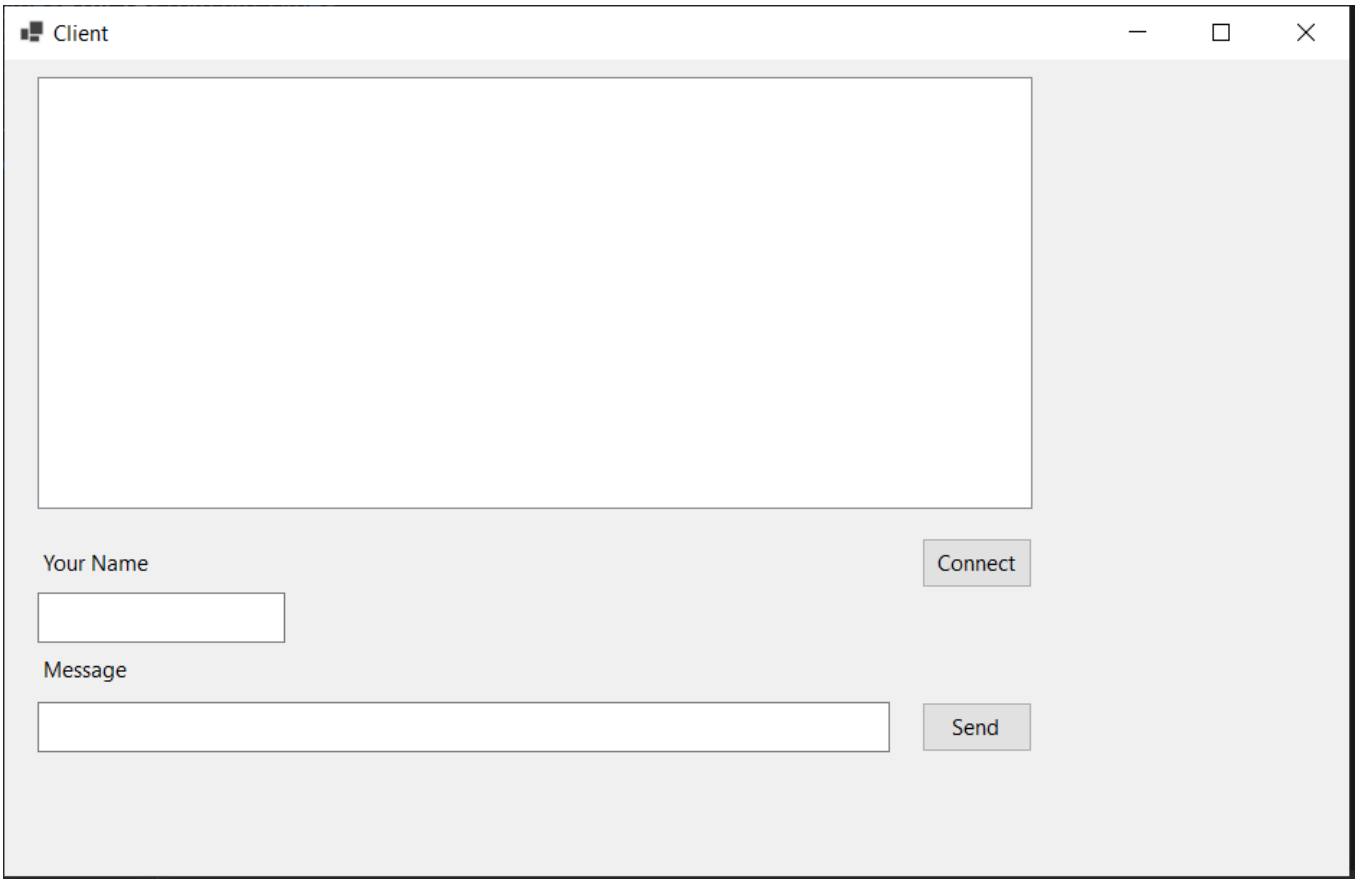
    + Server



Hình 1

- Click listen Server bắt đầu nắng nghe kết nối từ các client

+ Client



Hình 2

- Click connect để connect với Server. Nếu Server chưa listen thì hiện thông báo "Không thể kết nối Server!"
- Send để gửi tin nhắn

- Bước 2: Lập trình chức năng

+ Server

```
1 reference
private void btnListen_Click(object sender, EventArgs e)
{
    Connect();
    AddMessage("Server runing on 127.0.0.1:9999");
}

IPEndPoint IP;
Socket server;
List<Socket> clientList;
1 reference
void Connect()
{
    clientList = new List<Socket>();
    IP = new IPEndPoint(IPAddress.Any, 9999);
    server = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.IP);

    server.Bind(IP);

    Thread Listen = new Thread(() =>
    {
        try
        {
            while (true)
            {
                server.Listen(100);
                Socket client = server.Accept();
                clientList.Add(client);

                IPEndPoint ip = client.RemoteEndPoint as IPEndPoint;
                string ipAddress = "New client connectted form: " + ip.Address.ToString() + ":" + ip.Port.ToString();
```

Hình 3

```
                string ipAddress = "New client connectted form: " + ip.Address.ToString() + ":" + ip.Port.ToString();
                AddMessage(ipAddress);

                Thread receive = new Thread(Receive);
                receive.IsBackground = true;
                receive.Start(client);
            }
        }
        catch
        {
            IP = new IPEndPoint(IPAddress.Any, 9999);
            server = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.IP);
        }


    });
    Listen.IsBackground = true;
    Listen.Start();
}
```

- Khi nhấn click Connect server sẽ lắng nghe tất cả các client kết nối với server ở port 9999 và thêm vào list "clientList"

```
void Receive(object obj)
{
    Socket client = obj as Socket;


    try
    {
        while (true)
        {
            byte[] data = new byte[1024 * 5000];
            client.Receive(data);

            string message = (string)Deserialize(data);

            foreach (Socket item in clientList)
            {
                if (item != client)
                    item.Send(Serialize(message));
            }
            IPEndPoint ip = client.RemoteEndPoint as IPEndPoint;
            string ipAddress = ip.Address.ToString() + ":" + ip.Port.ToString();

            AddMessage(ipAddress + ": " +message);
        }
    }
    catch
    {
        clientList.Remove(client);
        client.Close();
    }

}
```

- Server nhận tin từ client dưới dạng byte, sử dụng Deserialize để chuyển mảng byte về string.
- Server sẽ gửi tin vừa nhận được cho tất cả client trong "clientList"
- Hàm Serialize để chuyển string sang mảng byte.

```
// Phân mảnh
1 reference
byte[] Serialize(object obj)
{
    MemoryStream stream = new MemoryStream();
    BinaryFormatter formatter = new BinaryFormatter();

    formatter.Serialize(stream, obj);

    return stream.ToArray();
}

// gom mảnh
1 reference
object Deserialize(byte[] data)
{
    MemoryStream stream = new MemoryStream(data);
    BinaryFormatter formatter = new BinaryFormatter();

    return formatter.Deserialize(stream);
}
```

Hình 4

- Serialize chuyển đổi string sang mảng byte
- Deserialize chuyển đổi từ mảng byte sang string

+ Client

```
1 reference
private void btnConnect_Click(object sender, EventArgs e)
{
    Connect();
}

IPEndPoint IP;
Socket client;
1 reference
void Connect()
{
    IP = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 9999);
    client = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.IP);

    try
    {
        client.Connect(IP);
    }
    catch
    {
        MessageBox.Show("Không thể kết nối Server!", " Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    Thread listen = new Thread(Receive);
    listen.IsBackground = true;
    listen.Start();

}
```

Hình 5

- Khi nhấn Connect client gửi yêu cầu kết nối với server

```
1 reference
private void btnSend_Click(object sender, EventArgs e)
{
    Send();
    AddMessage(txtName.Text+": "+txtMessage.Text);
}

2 references
void Close()
{
    client.Close();
}

1 reference
void Send()
{
    if (txtMessage.Text != String.Empty)
    {
        string mess = txtName.Text +": "+ txtMessage.Text;
        client.Send(Serialize(mess));
    }
}
```

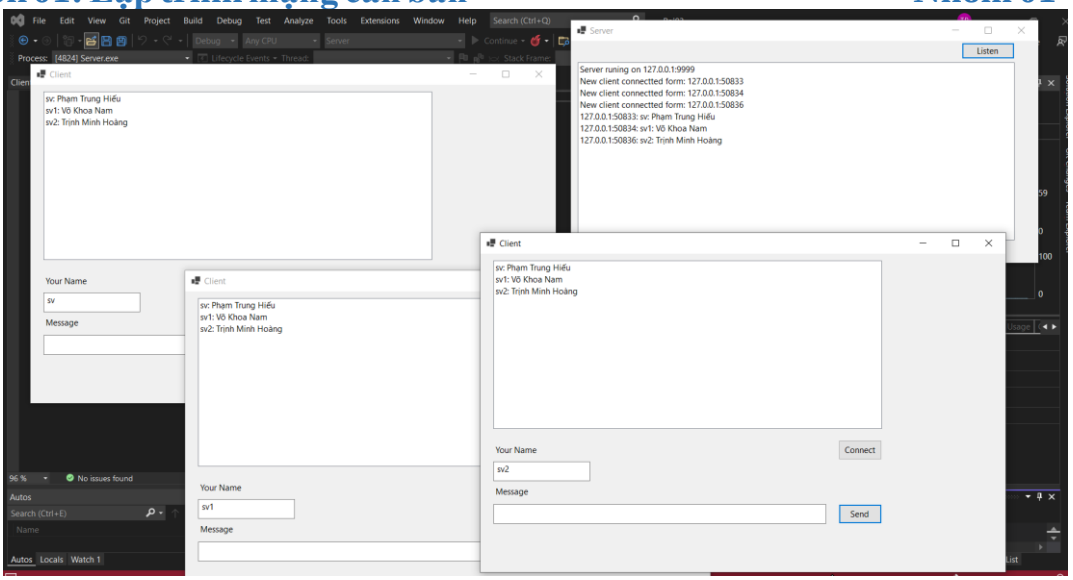
- Khi nhấn send client sẽ tin cho server dưới dạng mảng byte

```
1 reference
void Receive()
{
    try
    {
        while (true)
        {
            byte[] data = new byte[1024 * 5000];
            client.Receive(data);

            string message = (string)Deserialize(data);
            AddMessage(message);
        }
    }
    catch
    {
        Close();
    }

}
```

- Client nhận tin từ Server sau đó chuyển từ mảng byte sang kiểu string
  - Kết quả


Server
Listen
Server runing on 127.0.0.1:9999
New client connectted form: 127.0.0.1:50833
New client connectted form: 127.0.0.1:50834
New client connectted form: 127.0.0.1:50836
127.0.0.1:50833: sv: Pham Trung Hiếu
127.0.0.1:50834: sv1: Võ Khoa Nam
127.0.0.1:50836: sv2: Trịnh Minh Hoàng
Client
sv: Pham Trung Hiếu
sv1: Võ Khoa Nam
sv2: Trịnh Minh Hoàng
Your Name
sv
Message
Client
sv: Pham Trung Hiếu
sv1: Võ Khoa Nam
sv2: Trịnh Minh Hoàng
Your Name
sv1
Message
Client
sv: Pham Trung Hiếu
sv1: Võ Khoa Nam
sv2: Trịnh Minh Hoàng
Your Name
Connect
sv2
Message
Send

YÊU CẦU CHUNG

- Sinh viên tìm hiểu và thực hành theo hướng dẫn.
- Nộp báo cáo kết quả chi tiết những việc (**Report**) bạn đã thực hiện, quan sát thấy và kèm ảnh chụp màn hình kết quả (nếu có); giải thích cho quan sát (nếu có).
- Sinh viên báo cáo kết quả thực hiện và nộp bài.

Báo cáo:

- File .PDF. Tập trung vào nội dung, không mô tả lý thuyết.
- Nội dung trình bày bằng Font chữ Times New Romans/ hoặc font chữ của mẫu báo cáo này (UTM Neo Sans Intel/UTM Viet Sach)– cỡ chữ 13. Canh đều (Justify) cho văn bản. Canh giữa (Center) cho ảnh chụp.
- Đặt tên theo định dạng: [Mã lớp]-SessionX_GroupY. (trong đó X là Thứ tự buổi Thực hành, Y là số thứ tự Nhóm Thực hành đã đăng ký với GVHD-TH).

  *Ví dụ: [*NT101.K11.ANTT*]-Session1_Group3.*
- Nếu báo cáo có nhiều file, nén tất cả file vào file .ZIP với cùng tên file báo cáo.
- Không đặt tên đúng định dạng – yêu cầu, sẽ **KHÔNG** chấm điểm bài Lab.
- Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

Đánh giá: Sinh viên hiểu và tự thực hiện được bài thực hành. Khuyến khích:

- Chuẩn bị tốt.
- Có nội dung mở rộng, ứng dụng trong kịch bản phức tạp hơn, có đóng góp xây dựng bài thực hành.

Bài sao chép, trễ, … sẽ được xử lý tùy mức độ vi phạm.

HẾT