BÁO CÁO THỰC HÀNH

Môn học: Lập trình mạng căn bản

Lab 2: Lập trình Socket trong C#

GVHD: Đoàn Minh Trung

1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT106.N22.ATCL

STT	Họ và tên	MSSV	Email
1	Phạm Thanh Tâm	21522573	21522573@gm.uit.edu.vn
2	Phạm Đô Thành	21522603	21522603@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:[1]

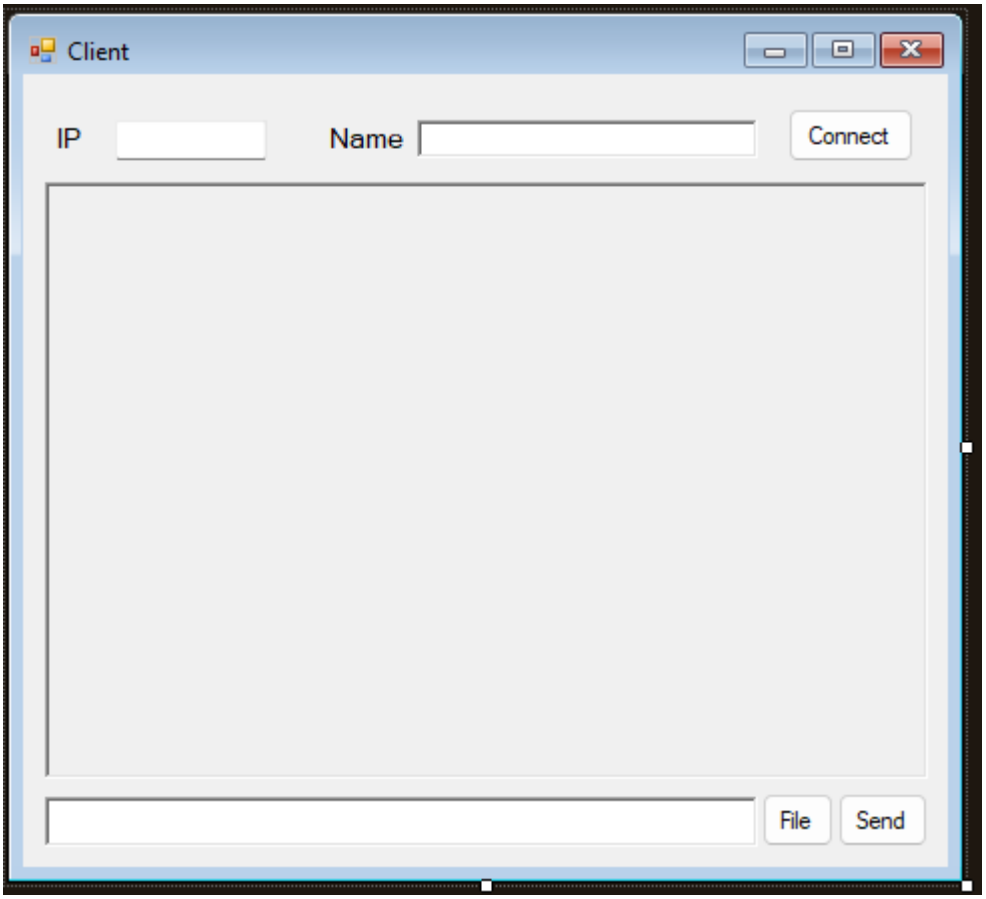
STT	Công việc	Kết quả tự đánh giá
1	Bài tập 4	100%

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

Bài 4: 1 Server – Multi Client Viết chương trình Chat Room/ Gửi và nhận dữ liệu sử dụng TCP Client và TCP Listener. Mỗi người dùng sẽ có một tài khoản, khi một người dùng gửi tin nhắn thì tất cả mọi người còn lại đều sẽ nhận được tin nhắn đó. Thêm vào đó là tính năng nhắn tin riêng, người dùng có thể chọn một người dùng khác để nhắn tin riêng. Cho phép gửi tin nhắn dạng hình ảnh (.jpg và .png) và file (.txt).



Hình 1: Giao diện Client

- Về Client: Ta có sự kiện buttonConnect như sau :

```
1 reference
private void buttonConnect_Click(object sender, EventArgs e)
{
    client = new TcpClient(textBoxIP.Text, 9494);
    stream = client.GetStream();

    richTextBox.AppendText("Connected to server.\n");
    richTextBox.AppendText("You've joined.\n");
    clientName = richTextBoxName.Text;

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

- Khi Click vào button này nó sẽ tạo ra 1 object TcpClient, địa chỉ IP sẽ được lấy từ textBoxIP và cổng sẽ là 9494. Tạo luồng mới ReceiveMessages để lắng nghe thông điệp từ máy chủ.

- Sự kiện buttonSend :

```
1 reference
private void buttonSend_Click(object sender, EventArgs e)
{
    string message = richTextBoxMessage.Text;
    if (message.StartsWith("@")) // Kiểm tra xem tin nhắn có phải là riêng tư hay không
    {
        string[] parts = message.Split(' ');
        int recipientPort = int.Parse(parts[0].Substring(1)); // Lấy cổng (port) người nhận
        string privateMessage = string.Join(" ", parts.Skip(1)); // Lấy nội dung tin nhắn riêng tư

        message = "@" + recipientPort + " " + privateMessage;

    }
    message = $"[{client.Client.LocalEndPoint}]{clientName}: {message}"; // Thêm ip:port của client vào message
    richTextBox.AppendText(message + "\n");
    richTextBoxMessage.Clear();

    byte[] data = Encoding.Unicode.GetBytes(message);
    stream.Write(data, 0, data.Length);

}
```

- Như đã chú thích ở trong code: Đầu tiên khi click vào ButtonSend nó sẽ kiểm tra tin nhắn có bắt đầu bằng ký tự « @ » hay không để nhận biết đó là tin nhắn riêng tư. Nếu có, giải mã tin nhắn riêng tư bằng cách tách cổng (port) của người nhận và nội dung tin nhắn riêng tư ra từ message, và sau đó gán message bằng @" + recipientPort + " " + privateMessage để có được một chuỗi tin nhắn riêng tư đầy đủ thông tin.

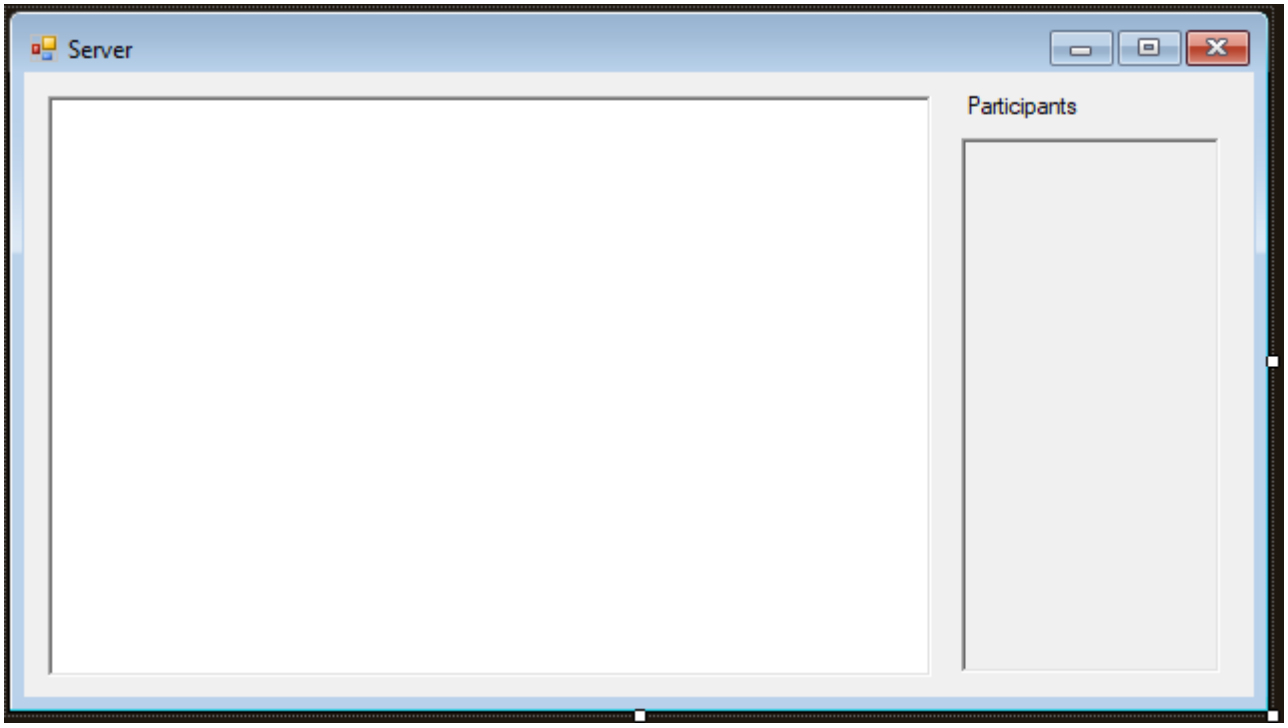
- Sử dụng client.Client.LocalEndPoint để lấy địa chỉ IP và Port của Client.

- Hàm ReceiveMessages :

```
private void ReceiveMessages()
{
    try
    {
        byte[] buffexr = new byte[1024];
        int bytesRead;

        while ((bytesRead = stream.Read(buffer, 0, buffer.Length)) != 0)
        {
            string message = Encoding.Unicode.GetString(buffer, 0, bytesRead);


            if (message.StartsWith("[file]")) // Kiểm tra xem tin nhắn gửi đi có phải file hay không
            {
                // Tin nhắn chứa tập tin được gửi từ server
                string fileName = message.Substring(6);
                string filePath = Path.Combine(Application.StartupPath, fileName);

                // Lưu tập tin vào thư mục của ứng dụng
                using (FileStream fileStream = new FileStream(filePath, FileMode.Create))
                {
                    while ((bytesRead = stream.Read(buffer, 0, buffer.Length)) > 0)
                    {
                        fileStream.Write(buffer, 0, bytesRead);
                    }
                }
            }
            else
            {
                if (message.StartsWith("@")) // Kiểm tra xem tin nhắn có phải là riêng tư hay không
                {
                    string[] parts = message.Split(' ');
                    int senderPort = int.Parse(parts[0].Substring(1)); // Lấy cổng (port) người gửi
                    string privateMessage = string.Join(" ", parts.Skip(1)); // Lấy nội dung tin nhắn riêng tư

                    richTextBox.Invoke(new Action(() => richTextBox.AppendText($"[{senderPort}] {privateMessage}\n")));

                }
                else
                {
                    richTextBox.Invoke(new Action(() => richTextBox.AppendText(message + "\n")));
                }
            }
        }
    }
    catch
    {

    }
```

- Phương thức này sẽ chạy 1 luồng riêng biệt để lắng nghe tin nhắn từ Server và xử lý chúng trong Client.
- Đầu tiên nó sẽ chạy vòng while nếu có bytes được đọc từ NetworkStream
- Buffer[] dùng để trữ byte đã nhận và đọc từ NetworkStream vào buffer với Read()
- Khi vào buffer sẽ được chuyển thành chuỗi sau đó check tin nhắn private.

- Về Server :



Hình 2 : Giao điện Server

- Hàm LoadServer :

```
1 reference
private void Server_Load(object sender, EventArgs e)
{
    server = new TcpListener(IPAddress.Any, 9494);
    server.Start();
    richTextBox1.AppendText("Server started.\n");

    Thread listenThread = new Thread(Listen);
    listenThread.Start();
}
```

- Khởi tạo 1 object TcpClient để lắng nghe kết nối từ Clients tới Server. Sau đó khởi chạy 1 luồng Listen mới để lắng nghe kết nối tới Server. Cuối cùng hiện « Server started » để thông báo cho người dùng biết Server đã được kết nối.

- Hàm Listen :

```
1 reference
private void Listen()
{
    while (true)
    {
        TcpClient client = server.AcceptTcpClient();
        clients.Add(client);
        int clientPort = ((IPEndPoint)client.Client.RemoteEndPoint).Port;
        connectedClientsDict.Add(clientPort, client); // Thêm cổng kết nối và TcpClient tương ứng vào connectedClientsDict
        connectedClients.Add(client.Client.RemoteEndPoint.ToString());
        UpdateConnectedClients();

        Thread clientThread = new Thread(() => HandleClient(client));
        clientThread.Start();
    }
}
```

- Chạy 1 luồng riêng biệt để lắng nghe kết nối từ các Clients tới Server. Sử dụng vòng lặp vô hạn để chấp nhận các kết nối từ các Client bằng cách sử dụng server.AcceptTcpClient(). Khi 1 kết nối được chấp nhận ta sẽ lưu chúng vào trong 1 danh sách Clients và 1 từ điển connectedClientsDict.

- Sau đó tạo 1 luồng mới để xử lý tin nhắn từ Client qua phương thức HandleClient

- Hàm HandleClient :

```
1 reference
private void HandleClient(TcpClient client)
{
    NetworkStream stream = client.GetStream();
    byte[] buffer = new byte[1024];
    int bytesRead;

    string clientName = "";

    while ((bytesRead = stream.Read(buffer, 0, buffer.Length)) != 0)
    {
        string message = Encoding.Unicode.GetString(buffer, 0, bytesRead);
        richTextBox1.Invoke(new Action(() => richTextBox1.AppendText(message + "\n")));
        if (message.StartsWith("[file]"))
        {
            // Tin nhắn chứa tập tin được gửi từ client
            string fileName = message.Substring(6);
            byte[] fileData = new byte[1024];
            int totalBytesRead = 0;
            int bytesToRead;

            // Đọc dữ liệu tập tin từ client
            while ((bytesToRead = stream.Read(fileData, 0, fileData.Length)) > 0)
            {
                // Ghi dữ liệu vào tập tin trên server
                using (FileStream fileStream = new FileStream(fileName, FileMode.Append))
                {
                    fileStream.Write(fileData, 0, bytesToRead);
                }

                totalBytesRead += bytesToRead;
                if (totalBytesRead >= bytesRead)
                {
                    break;
                }
            }

            // Gửi tập tin đến các client khác
            foreach (TcpClient otherClient in clients)
            {
                if (otherClient != client)
                {
                    NetworkStream otherStream = otherClient.GetStream();

                    // Gửi tin nhắn chứa tên tập tin đến các client khác
                    byte[] response = Encoding.Unicode.GetBytes(message);
                    otherStream.Write(response, 0, response.Length);

                    // Gửi dữ liệu tập tin đến các client khác
                    byte[] fileBuffer = File.ReadAllBytes(fileName);
                    otherStream.Write(fileBuffer, 0, fileBuffer.Length);
                }
            }
        }
```

```
        }
        else
        {
            string[] test = message.Split(' ');  // Tin nhắn được gửi từ client có format "[Ip][Name]: [@recipientPort] [message] " nếu là chat riêng
            string check = test[1]; // Ta cắt các khoảng trắng trong tin nhắn từ client và lấy phần thứ hai là [@recipientPort]
            if (check.StartsWith("@")) // Kiểm tra xem tin nhắn có phải là riêng tư hay không
            {
                string[] parts = message.Split(' '); // Cắt tin nhắn được gửi từ client
                string senderName = parts[0]; // Lấy ip và tên người gửi từ phần tử đầu tiên của mảng parts [Ip][Name]
                int recipientPort = int.Parse(parts[1].Substring(1)); // Lấy cổng (port) người nhận [@recipientPort]
                string privateMessage = string.Join(" ", parts.Skip(2)); // Lấy nội dung tin nhắn riêng tư [message]

                TcpClient recipientClient;
                if (connectedClientsDict.TryGetValue(recipientPort, out recipientClient))
                {
                    string senderPort = ((IPEndPoint)client.Client.RemoteEndPoint).Port.ToString();
                    string privateLog = $"private from {senderPort} to {recipientPort}: {privateMessage}";
                    // Ghi thông điệp từ client lên server vào richTextBox
                    richTextBox1.Invoke(new Action(() => richTextBox1.AppendText(privateLog + "\n")));

                    // Gửi tin nhắn riêng đến người nhận
                    NetworkStream recipientStream = recipientClient.GetStream();
                    privateMessage = senderName + " " + privateMessage;
                    byte[] privateResponse = Encoding.Unicode.GetBytes(privateMessage);
                    recipientStream.Write(privateResponse, 0, privateResponse.Length);
                }
            }
            else
            {
                if (clientName == "") // Nếu tên người dùng chưa được đặt, sử dụng tin nhắn đầu tiên làm tên
                {
                    clientName = message;
                    clientNames.Add(client, clientName);
                }

                foreach (TcpClient otherClient in clients)
                {
                    if (otherClient != client)
                    {
                        NetworkStream otherStream = otherClient.GetStream();
                        byte[] response = Encoding.Unicode.GetBytes(message);
                        otherStream.Write(response, 0, response.Length);
                    }
                }
            }
        }

    }

    clients.Remove(client);
    int disconnectedPort = ((IPEndPoint)client.Client.RemoteEndPoint).Port;
    connectedClientsDict.Remove(disconnectedPort); // Xóa cổng kết nối khi client ngắt kết nối
    connectedClients.Remove(client.Client.RemoteEndPoint.ToString());
    clientNames.Remove(client); // Xóa thông tin người dùng khi client ngắt kết nối
    UpdateConnectedClients();
}
```

- Phương thức HandleClient được khởi chạy trong một luồng riêng để xử lý tin nhắn từ một client được kết nối tới máy chủ. Trong phương thức này, bạn sử dụng đối tượng TcpClient để lấy đối tượng NetworkStream để có thể đọc và ghi dữ liệu từ client.
- Sử dụng một vòng lặp vô hạn để đọc các tin nhắn từ client thông qua đối tượng NetworkStream. Nếu tin nhắn chứa tên tập tin được gửi từ client, bạn lưu trữ tập tin đó trên máy chủ và gửi tập tin đó tới tất cả các client khác ngoại trừ client gửi tin nhắn. Nếu tin nhắn không chứa tên tập tin, bạn kiểm tra xem tin nhắn có phải là tin nhắn riêng tư hay không. Nếu là tin nhắn riêng, bạn gửi tin nhắn đó tới client nhận được tin nhắn riêng tư. Nếu không phải là tin nhắn riêng, bạn gửi tin nhắn đó tới tất cả các client khác.
- Nếu tên người dùng của client chưa được đặt, bạn sử dụng tin nhắn đầu tiên của client làm tên người dùng và lưu trữ thông tin về tên người dùng đó trong clientNames.
- Cuối cùng, khi client ngắt kết nối, bạn xóa thông tin liên quan đến client đó khỏi danh sách và từ điển lưu trữ thông tin về các client đã kết nối, và cập nhật giao diện người dùng thông qua phương thức UpdateConnectedClients.

- Hàm UpdateConnectedClients() :

```
private void UpdateConnectedClients()
{
    StringBuilder sb = new StringBuilder();
    foreach (string client in connectedClients)
    {
        sb.AppendLine(client);
    }
    richTextBox3.Invoke(new Action(() => richTextBox3.Text = sb.ToString()));

}
```

- Hàm này tạo StringBuilder object mới để lưu danh sác h các client và nối các endpoint của mỗi client vào StringBuilder, cuối cùng update tin nhắn ở khung richTextBox3

Sinh viên đọc kỹ yêu cầu trình bày bên dưới trang này

YÊU CẦU CHUNG

- Sinh viên tìm hiểu và thực hiện bài tập theo yêu cầu, 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 .DOCX và .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]-ExeX_GroupY. (trong đó X là Thứ tự Bài tập, Y là mã số thứ tự nhóm trong danh sách mà GV phụ trách công bố).

  *Ví dụ: [*NT101.K11.ANTT*]-Exe01_Group03.*
- 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 nộp.
- Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

Đánh giá:

- Hoàn thành tốt yêu cầu được giao.
- Có nội dung mở rộng, ứng dụng.

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

HẾT