**Báo cáo Lab3**

**Nhóm: 2**

**Lớp: NT106.M21.1**

**Giảng Viên Hướng Dẫn: Lê Minh Khánh Hội**

Thành viên nhóm:

1. Phan Nguyễn Văn Duy: 20520168

2. Vi Minh Tiến: 20520810

3. Nguyễn Đàm Nhật Anh: 20520880

4. Khưu Minh Phong: 20521742

**Báo cáo**

**Câu 1: Viết ứng dụng thực hiện gởi và nhận dữ liệu giữa 2 bên sử dụng giao thức UDP (UDP Client và UDP Server). Người dùng ở Client sẽ chỉ định IP, port cần kết nối và thông điệp gửi đến Server. Tại Server sẽ nhận được thông điệp từ Client**

Giao diện:

Graphical user interface, application

Description automatically generated

Hoạt động:

+ Server chỉ định port

+ Bên client điền port và ip (sử dụng ip loopback).

+ Server nhấn nút listen

+ Client điền và gửi thông điệp

Code sự kiện cho nút Listen ở Server:

CheckForIllegalCrossThreadCalls = false;

Thread thdUDPServer = new Thread(new ThreadStart(serverThread));

thdUDPServer.Start();

Hàm xử lý:

public void InfoMessage(string mess)

{

tbReceived.Text += mess + "\r\n";

}

public void serverThread()

{

int port = 0;

try

{

port = int.Parse(tbPort.Text);

lbListen.Text = "Listen: ON";

}

catch(Exception)

{

MessageBox.Show("Vui lòng nhập port");

}

UdpClient udpClient = new UdpClient(port);

while (true)

{

IPEndPoint RemoteIpEndPoint = new IPEndPoint(IPAddress.Any, 0);

Byte[] receiveBytes = udpClient.Receive(ref RemoteIpEndPoint);

string returnData = Encoding.ASCII.GetString(receiveBytes);

string mess = RemoteIpEndPoint.Address.ToString() + " : " + returnData.ToString();

InfoMessage(mess);

}

}

**Câu 2: Viết chương trình lắng nghe từ dịch vụ Telnet sử dụng kết nối TCP (sử dụng lớp Socket) với mô tả sau:**

1. Chạy chương trình
2. Nhấn nút Listen
3. Mở CMD gõ lệnh: telnet <IP của máy> 8080
4. Vào màn hình telnet, gõ thông điệp tùy ý, chương trình sẽ nhận và hiển thị lên form.
5. Xem hình mẫu

Giao diện:

Graphical user interface, application

Description automatically generated

Sự kiện cho nút Listen:

private void button1\_Click(object sender, EventArgs e)

{

Thread serverThread = new Thread(new ThreadStart(StartUnsafeThread));

serverThread.Start();

}

Hàm xử lý:

public void StartUnsafeThread()

{

int recv;

byte[] data = new byte[1024];

IPAddress serverIP = IPAddress.Parse("127.0.0.1");

IPEndPoint ipep = new IPEndPoint(serverIP, 8080);

Socket newsock = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);

newsock.Bind(ipep);

newsock.Listen(10);

Socket client = newsock.Accept();

while (true)

{

data = new byte[1024];

recv = client.Receive(data);

if (recv == 0) break;

string rcvMessage = Encoding.ASCII.GetString(data, 0, recv);

richTextBox1.Text += rcvMessage;

}

}

**Câu 3: Viết ứng dụng thực hiện gởi và nhận dữ liệu sử dụng giao thức TCP (TCP Client và TCP Listener). Server lắng nghe kết nối và thông điệp từ Client.**

1. Chạy chương trình
2. Nhấn nút Listen
3. Khởi tạo Client
4. Gửi thông điệp từ Client đến Server
5. Server nhận thông điệp và hiện lên form.
6. Xem hình mẫu.

*Note: Mở 2 lần Visual Studio.*

Giao diện:

Graphical user interface, application, Word

Description automatically generated

Sự kiện cho nút Listen:

private void btnListen\_Click(object sender, EventArgs e)

{

tbServer.Text = "Server running on 127.0.0.1:8080\n";

try

{

IPAddress IPAddress = IPAddress.Parse("127.0.0.1");

TcpListener tcpListener = new TcpListener(IPAddress, 8080);

// 1. Khoi dong server

tcpListener.Start();

tbServer.Text += "New client connected\n";

Socket socket = tcpListener.AcceptSocket();

var stream = new NetworkStream(socket);

var reader = new StreamReader(stream);

var writer = new StreamWriter(stream);

writer.AutoFlush = true;

// 2. Nhan message

string message = reader.ReadLine();

tbServer.Text += message + "\n";

// 3. Ngat ket noi

stream.Close();

socket.Close();

tcpListener.Stop();

}

catch (Exception ex)

{

MessageBox.Show("Lỗi");

}

}

Sự kiện cho nút Send:

private void btnSend\_Click(object sender, EventArgs e)

{

try

{

TcpClient tcpClient = new TcpClient();

// 1. Ket noi

tcpClient.Connect("127.0.0.1", 8080);

Stream stream = tcpClient.GetStream();

MessageBox.Show("Gửi thành công");

// 2. Khoi tao message

string message = "Hello server";

var reader = new StreamReader(stream);

var writer = new StreamWriter(stream);

writer.AutoFlush = true;

// 3. Gui message

writer.WriteLine(message);

// 4. Dong ket noi

stream.Close();

tcpClient.Close();

}

catch (Exception ex)

{

MessageBox.Show("Có lỗi xảy ra, vui lòng kiểm tra server");

}

}

**Câu 4: 1 Server – Multi Client – Viết chương trình Chat Room/ Gửi và nhận dữ liệu sử dụng giao thức TCP Client và TCP Listener. Mỗi người 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.**

Giao diện:

Graphical user interface, application, Word

Description automatically generated

Hoạt động:

+ Bên Server khởi tạo kết nối:

+ Các client yêu cầu kết nối với server

+ Server chấp nhận kết nối

+ Các client điền tên và thông điệp sau đó ấn gửi.

Code của Client:

public Lab03\_Bai4\_Client()

{

InitializeComponent();

CheckForIllegalCrossThreadCalls = false;

Connect();

}

private void sendBtn\_Click(object sender, EventArgs e)

{

Send();

AddMessage("Me: " + sendBox.Text);

}

IPEndPoint IP;

Socket client;

void Connect()

{

IP = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 8088);

client = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.IP);

try

{

client.Connect(IP);

}

catch

{

MessageBox.Show("Không kết nối được server", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

Thread listen = new Thread(Receive);

listen.IsBackground = true;

listen.Start();

}

void Send()

{

if (sendBox.Text != string.Empty)

client.Send(Serialize(nameBox.Text + ": " + sendBox.Text));

}

void Receive()

{

try

{

while (true)

{

byte[] data = new byte[4096];

client.Receive(data);

string message = (string)Deserialize(data);

AddMessage(message);

}

}

catch

{

Close();

}

}

void AddMessage(string s)

{

chatBox.Items.Add(new ListViewItem() { Text = s });

sendBox.Clear();

}

byte[] Serialize(object obj)

{

MemoryStream stream = new MemoryStream();

BinaryFormatter formatter = new BinaryFormatter();

formatter.Serialize(stream, obj);

return stream.ToArray();

}

object Deserialize(byte[] data)

{

MemoryStream stream = new MemoryStream(data);

BinaryFormatter formatter = new BinaryFormatter();

return formatter.Deserialize(stream);

}

Code của Server:

public Lab03\_Bai4\_Server()

{

InitializeComponent();

CheckForIllegalCrossThreadCalls = false;

Connect();

}

IPEndPoint IP;

Socket server;

List<Socket> clientList;

void Connect()

{

clientList = new List<Socket>();

IP = new IPEndPoint(IPAddress.Any, 8088);

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);

Thread receive = new Thread(Receive);

receive.IsBackground = true;

receive.Start(client);

}

}

catch

{

IP = new IPEndPoint(IPAddress.Any, 8080);

server = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.IP);

}

});

Listen.IsBackground = true;

Listen.Start();

}

void Close()

{

server.Close();

}

void Receive(Object obj)

{

Socket client = obj as Socket;

try

{

while (true)

{

byte[] data = new byte[4096];

client.Receive(data);

string message = (string)Deserialize(data);

foreach (var item in clientList)

{

if (item != null && item != client)

item.Send(Serialize(message));

}

AddMessage(message);

}

}

catch

{

clientList.Remove(client);

client.Close();

}

}

void AddMessage(string s)

{

rcvBox.Items.Add(new ListViewItem() { Text = s });

}

byte[] Serialize(object obj)

{

MemoryStream stream = new MemoryStream();

BinaryFormatter formatter = new BinaryFormatter();

formatter.Serialize(stream, obj);

return stream.ToArray();

}

object Deserialize(byte[] data)

{

MemoryStream stream = new MemoryStream(data);

BinaryFormatter formatter = new BinaryFormatter();

return formatter.Deserialize(stream);

}