

public partial class udpServer2 : Form

{

Socket serverSocket;

bool isRunning;

public udpServer2()

{

InitializeComponent();

}

private void udpServer2\_Load(object sender, EventArgs e)

{

Thread serverThread = new Thread(RunServer);

serverThread.IsBackground = true;

serverThread.Start();

}

private void RunServer()

{

isRunning = true;

try

{

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

serverSocket.Bind(new IPEndPoint(IPAddress.Any, 8080));

serverSocket.Listen(5);

MessageBox.Show("Server đang chạy");

while (isRunning) // Kiểm tra biến isRunning

{

if (serverSocket.Poll(1000, SelectMode.SelectRead)) // Kiểm tra có kết nối không

continue;

Socket clientsSocket = serverSocket.Accept();

Thread clientThread = new Thread(() => HandleClient(clientsSocket));

clientThread.Start();

}

}

catch (Exception ex)

{

if (isRunning) // Chỉ báo lỗi nếu server chưa đóng

MessageBox.Show("Lỗi Server: " + ex.Message);

}

}

private void HandleClient(Socket clientsSocket)

{

byte[] buffes = new byte[1024];

int receivedBytes;

try

{

while ((receivedBytes = clientsSocket.Receive(buffes)) > 0)

{

string s = Encoding.UTF8.GetString(buffes, 0, receivedBytes);

if (string.IsNullOrEmpty(s)) break;

if (s.ToUpper().Equals("QUIT")) break;

string[] arr = s.Split('#');

if (arr.Length != 3) continue;

int a, b, kq = 0;

if (!int.TryParse(arr[0], out a) || !int.TryParse(arr[2], out b)) continue;

switch (arr[1])

{

case "cộng":

kq = a + b;

break;

case "trừ":

kq = a - b;

break;

case "nhân":

kq = a \* b;

break;

case "chia":

kq = (b != 0) ? (a / b) : int.MaxValue;

break;

default:

kq = 0;

break;

}

string response = kq.ToString(); // Gửi kết quả thay vì dữ liệu thô

clientsSocket.Send(Encoding.UTF8.GetBytes(response));

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi Client: " + ex.Message);

}

finally

{

clientsSocket.Close();

}

}

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

{

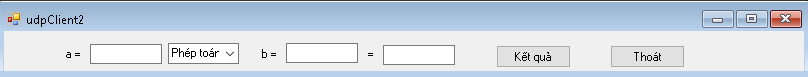
isRunning = false; // Dừng vòng lặp trong RunServer

serverSocket?.Close(); // Đóng socket server nếu có

Application.Exit(); // Thoát ứng dụng

}

}



public partial class udpForm1 : Form

{

// 1⃣ Tạo socket client

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

string s;

public udpForm1()

{

InitializeComponent();

}

private void udpForm1\_Load(object sender, EventArgs e)

{

// 2⃣ Kết nối đến server

try

{

clientSocket.Connect(new IPEndPoint(IPAddress.Parse("192.168.43.17"), 8080));

MessageBox.Show("Kết nối thành công!");

}

catch (Exception ex)

{

MessageBox.Show("Không tìn thấy server!");

}

}

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

{

if (lable1.Text != "" && cbbpheptoan.Text != "" && lable2.Text != "") send();

}

public void send()

{

s = txta.Text + '#' + cbbpheptoan.Text + '#' + txtb.Text;

clientSocket.Send(Encoding.UTF8.GetBytes(s));

byte[] buffer = new byte[1024];

int receivedBytes = clientSocket.Receive(buffer);

string response = Encoding.UTF8.GetString(buffer, 0, receivedBytes);

txtkq.Text = response.ToString();

}

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

{

try

{

s = "QUIT";

clientSocket.Send(Encoding.UTF8.GetBytes(s));

// Gửi thông báo đóng kết nối đến server (nếu cần)

clientSocket.Shutdown(SocketShutdown.Both);

clientSocket.Close();

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi đóng kết nối: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

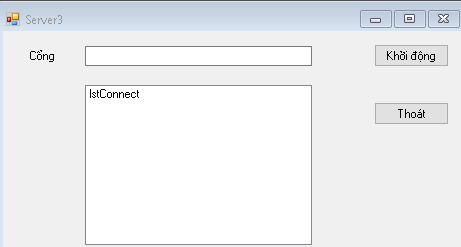
}

// Đóng ứng dụng

this.Close();

}

}



public partial class Server3 : Form

{

int BufferSize = 1024;

private delegate void dlgAddItem(String str);

private void AddItem(String str)

{

if (this.lstConnect.InvokeRequired)

{

this.Invoke(new dlgAddItem(AddItem), str);

}

else { this.lstConnect.Items.Add(str); }

}

public Server3()

{

InitializeComponent();

}

private void Server3\_Load(object sender, EventArgs e)

{

}

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

{

Thread thdListener = new Thread(new ThreadStart(ListenerThead));

thdListener.Start();

AddItem("Server đã khởi động");

}

public void ListenerThead()

{

try

{

ReceiveTCP(int.Parse(txtPort.Text));

}

catch (Exception ex)

{

MessageBox.Show("Lỗi đọc cổng");

}

}

public void ReceiveTCP(int portN)

{

TcpListener Listener = null;

try

{

Listener = new TcpListener(IPAddress.Any, portN);

Listener.Start();

byte[] RecData = new byte[BufferSize];

int RecBytes;

while (true)

{

TcpClient client = null;

NetworkStream networkStream = null;

if (Listener.Pending())

{

client = Listener.AcceptTcpClient();

networkStream = client.GetStream();

AddItem("Kết nối với client");

string SaveFileName = "Z:/TH3/Server3/test01.txt";

int totalrecbytes = 0;

FileStream Fs = new FileStream(SaveFileName, FileMode.OpenOrCreate, FileAccess.Write);

while ((RecBytes = networkStream.Read(RecData, 0, RecData.Length)) > 0)

{

Fs.Write(RecData, 0, RecBytes);

totalrecbytes += RecBytes;

}

Fs.Close();

networkStream.Close();

client.Close();

AddItem("Đã lưu tập tin");

}

}

}

catch (Exception ex)

{

MessageBox.Show("Lưu file thất bại");

}

}

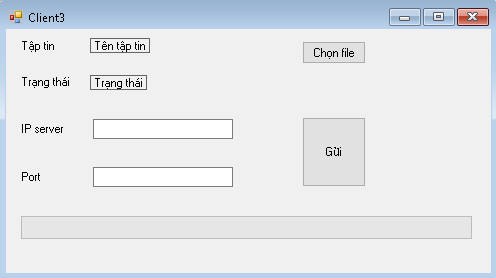
private void buttonThoat\_Click(object sender, EventArgs e)

{

Application.Exit();

}

}



public partial class Client3 : Form

{

int BufferSize = 1024;

public Client3()

{

InitializeComponent();

}

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

{

if (fdlg.ShowDialog() == DialogResult.OK)

{

lblFile.Text = fdlg.FileName;

}

}

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

{

try

{

if (lblFile.Text != "")

SendTCP(lblFile.Text, txtIP.Text, int.Parse(txtPort.Text.Trim()));

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

public void SendTCP(string M, string IPA, Int32 PortN)

{

byte[] SendingBuffer = null;

TcpClient client = null;

NetworkStream netstream = null;

try

{

client = new TcpClient(IPA, PortN);

MessageBox.Show("Kết đến server thành công");

lblState.Text = "Kết nối server...\n";

netstream = client.GetStream();

FileStream Fs = new FileStream(M, FileMode.Open, FileAccess.Read);

int NoOfPackets = Convert.ToInt32(Math.Ceiling(Convert.ToDouble(Fs.Length) / Convert.ToDouble(BufferSize)));

progressBar1.Maximum = NoOfPackets;

int TotalLength = (int)Fs.Length, CurrentPacketLength;

for (int i = 0; i < NoOfPackets; i++)

{

if (TotalLength > BufferSize)

{

CurrentPacketLength = BufferSize;

TotalLength = TotalLength - CurrentPacketLength;

}

else CurrentPacketLength = TotalLength;

SendingBuffer = new byte[CurrentPacketLength];

Fs.Read(SendingBuffer, 0, CurrentPacketLength);

netstream.Write(SendingBuffer, 0, (int)SendingBuffer.Length);

if (progressBar1.Value >= progressBar1.Maximum)

progressBar1.Value = progressBar1.Minimum;

progressBar1.PerformStep();

}

lblState.Text = lblState.Text + " Đã gửi " + Fs.Length.ToString() + " bytes đến Server";

Fs.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

finally

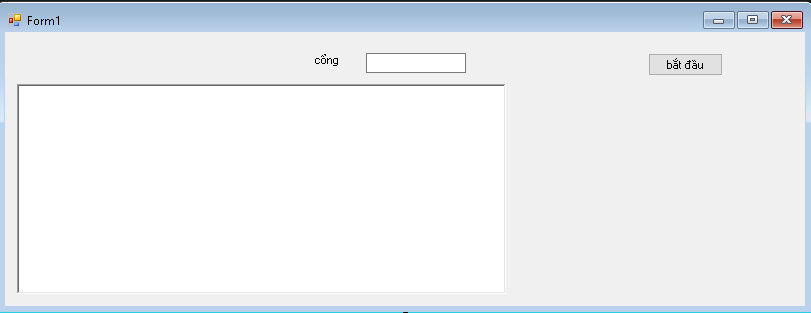
{

netstream.Close(); client.Close();

}

}

}



public partial class Form1 : Form

{

TcpListener listener = null;

NetworkStream netStream = null;

TcpClient client = null;

Thread listenThread = null;

Thread receiveThread = null;

const int BUFFER\_SIZE = 8192;

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

CheckForIllegalCrossThreadCalls = false;

this.FormClosed += (s, ev) => StopServer();

}

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

{

int port = int.Parse(txtPort.Text.Trim());

listener = new TcpListener(IPAddress.Any, port);

listener.Start();

listenThread = new Thread(() =>

{

AppendLog($"🟢 Đang chờ client kết nối trên port {port}...");

client = listener.AcceptTcpClient(); // Block đến khi có client

netStream = client.GetStream();

AppendLog("✅ Client đã kết nối!");

receiveThread = new Thread(ReceiveData);

receiveThread.Start();

});

listenThread.Start();

}

private void ReceiveData()

{

try

{

byte[] buffer = new byte[BUFFER\_SIZE];

MemoryStream ms = null;

string fileName = "";

bool isReceivingFile = false;

while (true)

{

int bytesRead = netStream.Read(buffer, 0, buffer.Length);

if (bytesRead == 0) break;

// Nếu đang không ở chế độ nhận file, ta kiểm tra header

if (!isReceivingFile)

{

string header = Encoding.UTF8.GetString(buffer, 0, bytesRead);

if (header.StartsWith("<FILE>:"))

{

fileName = header.Substring(7).Trim();

ms = new MemoryStream();

isReceivingFile = true;

AppendLog($"📥 Bắt đầu nhận file: {fileName}");

continue;

}

if (header == "<EOF>")

{

// Đề phòng nếu nhận EOF mà chưa có dữ liệu

continue;

}

// Nếu không phải là file, xem là tin nhắn

string msg = Encoding.UTF8.GetString(buffer, 0, bytesRead);

AppendLog("💬 Client: " + msg);

}

else

{

// Kiểm tra nếu buffer chứa EOF

string maybeEOF = Encoding.UTF8.GetString(buffer, 0, bytesRead);

if (maybeEOF.Contains("<EOF>"))

{

// Cắt bỏ phần "<EOF>" nếu có

int eofIndex = maybeEOF.IndexOf("<EOF>");

if (eofIndex > 0)

ms.Write(buffer, 0, eofIndex);

// Lưu file

string saveDir = @"C:\ReceivedFiles";

Directory.CreateDirectory(saveDir);

string savePath = Path.Combine(saveDir, fileName);

File.WriteAllBytes(savePath, ms.ToArray());

ms.Close();

ms = null;

isReceivingFile = false;

AppendLog($"✅ Đã lưu file: {fileName}");

}

else

{

// Ghi dữ liệu nhị phân vào MemoryStream

ms.Write(buffer, 0, bytesRead);

}

}

}

}

catch (Exception ex)

{

AppendLog("❌ Lỗi nhận dữ liệu: " + ex.Message);

}

}

private void AppendLog(string text)

{

rtxLog.AppendText(text + "\r\n");

}

private void StopServer()

{

try

{

receiveThread?.Abort();

listenThread?.Abort();

netStream?.Close();

client?.Close();

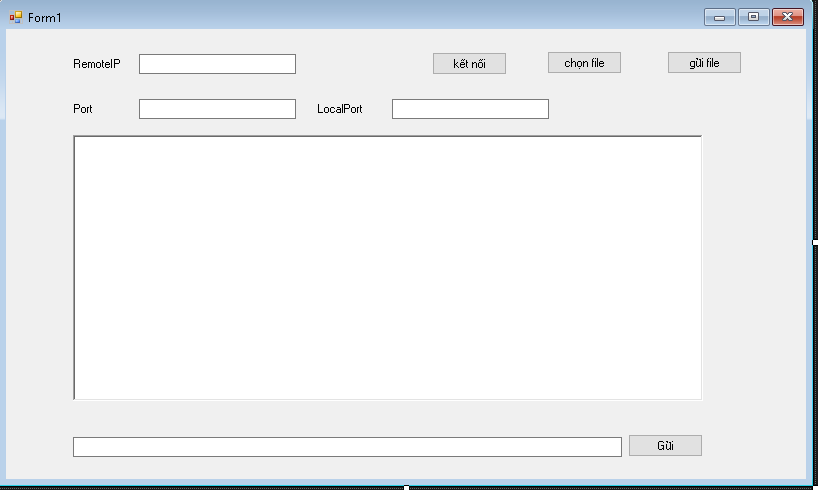
listener?.Stop();

}

catch { }

}

}



public partial class Form1: Form

{

TcpClient tcpClient = null;

NetworkStream netStream = null;

Thread receiveThread = null;

const int BUFFER\_SIZE = 8192;

string filePath = "", fileName = "";

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

CheckForIllegalCrossThreadCalls = false;

this.FormClosed += new FormClosedEventHandler(closeForm);

}

private void closeForm(object sender, EventArgs e)

{

try

{

receiveThread?.Abort();

netStream?.Close();

tcpClient?.Close();

}

catch { }

}

private void openinput(bool state)

{

btnGui.Enabled = !state;

btnSendFile.Enabled = !state;

btnBrowseFile.Enabled = !state;

txtIPR.ReadOnly = !state;

txtPortL.ReadOnly = !state;

txtPortR.ReadOnly = !state;

btntKN.Enabled = state;

}

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

{

try

{

string ip = txtIPR.Text.Trim();

int port = int.Parse(txtPortR.Text.Trim());

tcpClient = new TcpClient();

tcpClient.Connect(ip, port);

netStream = tcpClient.GetStream();

openinput(false);

receiveThread = new Thread(new ThreadStart(ReceiveData));

receiveThread.Start();

}

catch (Exception ex)

{

MessageBox.Show("Lỗi kết nối: " + ex.Message);

}

}

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

{

try

{

string message = txtMsg.Text;

byte[] data = Encoding.UTF8.GetBytes(message);

netStream.Write(data, 0, data.Length);

rtxMsg.AppendText("Send: " + message + "\r\n");

}

catch (Exception ex)

{

MessageBox.Show("Lỗi gửi tin nhắn: " + ex.Message);

}

}

private void ReceiveData()

{

try

{

byte[] buffer = new byte[BUFFER\_SIZE];

MemoryStream ms = null;

string receivedFileName = "";

while (true)

{

int bytesRead = netStream.Read(buffer, 0, buffer.Length);

if (bytesRead == 0) break; // Ngắt kết nối

string header = Encoding.UTF8.GetString(buffer, 0, bytesRead);

if (header.StartsWith("<FILE>:"))

{

receivedFileName = header.Substring(7);

ms = new MemoryStream();

continue;

}

if (header == "<EOF>")

{

string saveDir = @"C:\ReceivedFiles";

Directory.CreateDirectory(saveDir);

string savePath = Path.Combine(saveDir, receivedFileName);

File.WriteAllBytes(savePath, ms.ToArray());

rtxMsg.AppendText($"[Đã nhận file: {receivedFileName}]\r\n");

ms.Close();

ms = null;

continue;

}

if (ms != null)

{

ms.Write(buffer, 0, bytesRead);

}

else

{

string text = Encoding.UTF8.GetString(buffer, 0, bytesRead);

rtxMsg.AppendText("Receive: " + text + "\r\n");

}

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi nhận dữ liệu: " + ex.Message);

}

}

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

{

OpenFileDialog ofd = new OpenFileDialog();

if (ofd.ShowDialog() == DialogResult.OK)

{

filePath = ofd.FileName;

fileName = Path.GetFileName(filePath);

txtMsg.Text = fileName;

}

}

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

{

try

{

if (string.IsNullOrEmpty(filePath)) return;

// Gửi tên file

string header = "<FILE>:" + fileName;

byte[] headerBytes = Encoding.UTF8.GetBytes(header);

netStream.Write(headerBytes, 0, headerBytes.Length);

Thread.Sleep(10);

// Gửi nội dung file

using (FileStream fs = new FileStream(filePath, FileMode.Open, FileAccess.Read))

{

byte[] buffer = new byte[BUFFER\_SIZE];

int bytesRead;

while ((bytesRead = fs.Read(buffer, 0, buffer.Length)) > 0)

{

netStream.Write(buffer, 0, bytesRead);

Thread.Sleep(1); // tránh nghẽn buffer

}

}

// Gửi EOF

byte[] eof = Encoding.UTF8.GetBytes("<EOF>");

netStream.Write(eof, 0, eof.Length);

rtxMsg.AppendText($"[Đã gửi file: {fileName}]\r\n");

}

catch (Exception ex)

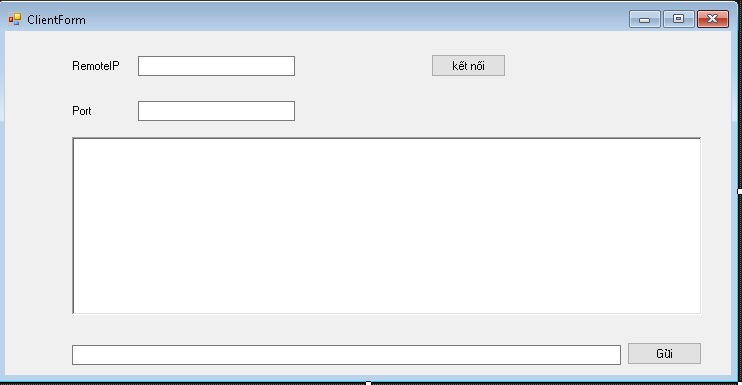
{

MessageBox.Show("Lỗi gửi file: " + ex.Message);

}

}

}



public partial class ClientForm : Form

{

private TcpClient client;

private NetworkStream stream;

public ClientForm()

{

InitializeComponent();

}

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

{

string ip = txtServerIP.Text.Trim();

int port = int.Parse(txtPort.Text.Trim());

try

{

client = new TcpClient();

client.Connect(ip, port);

stream = client.GetStream();

txtChat.AppendText("Đã kết nối đến server!\r\n");

Thread receiveThread = new Thread(ReceiveData);

receiveThread.IsBackground = true;

receiveThread.Start();

}

catch (Exception ex)

{

MessageBox.Show("Kết nối thất bại: " + ex.Message);

}

}

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

{

if (stream != null && client.Connected)

{

byte[] data = Encoding.UTF8.GetBytes(txtMessage.Text);

stream.Write(data, 0, data.Length);

txtChat.AppendText("Bạn: " + txtMessage.Text + "\r\n");

txtMessage.Clear();

}

}

private void ClientForm\_Load(object sender, EventArgs e)

{

}

private void ReceiveData()

{

byte[] buffer = new byte[1024];

int bytesRead;

while (client.Connected)

{

try

{

bytesRead = stream.Read(buffer, 0, buffer.Length);

string msg = Encoding.UTF8.GetString(buffer, 0, bytesRead);

txtChat.Invoke(new Action(() =>

{

txtChat.AppendText("Server: " + msg + "\r\n");

}));

}

catch

{

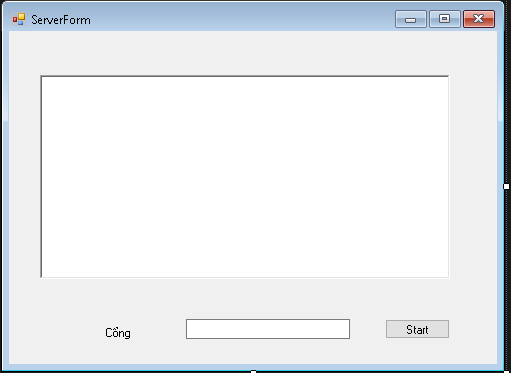
break;

}

}

}

}



public partial class ServerForm : Form

{

private TcpListener listener;

private TcpClient client;

private NetworkStream stream;

public ServerForm()

{

InitializeComponent();

}

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

{

int port = int.Parse(txtPort.Text);

listener = new TcpListener(IPAddress.Any, port);

listener.Start();

txtLog.AppendText("Server đang lắng nghe trên cổng " + port + "\r\n");

Thread acceptThread = new Thread(() =>

{

client = listener.AcceptTcpClient();

stream = client.GetStream();

txtLog.Invoke(new Action(() =>

{

txtLog.AppendText("Client đã kết nối!\r\n");

}));

ReceiveData();

});

acceptThread.IsBackground = true;

acceptThread.Start();

}

private void ReceiveData()

{

byte[] buffer = new byte[1024];

int bytesRead;

while (client.Connected)

{

try

{

bytesRead = stream.Read(buffer, 0, buffer.Length);

string msg = Encoding.UTF8.GetString(buffer, 0, bytesRead);

txtLog.Invoke(new Action(() =>

{

txtLog.AppendText("Client: " + msg + "\r\n");

}));

}

catch

{

break;

}

}

}

}

### 📌 3.1. Server-side TCP

csharp

CopyEdit

// 1. Khai báo listener

TcpListener listener = new TcpListener(IPAddress.Any, 1234);

listener.Start();

// 2. Chấp nhận kết nối từ client

TcpClient client = listener.AcceptTcpClient(); // blocking

NetworkStream stream = client.GetStream();

// 3. Đọc/Gửi dữ liệu

byte[] buffer = new byte[1024];

int bytesRead = stream.Read(buffer, 0, buffer.Length); // nhận

stream.Write(buffer, 0, bytesRead); // gửi lại

// 4. Đóng

stream.Close();

client.Close();

listener.Stop();

### 📌 3.2. Client-side TCP

csharp

CopyEdit

// 1. Kết nối đến server

TcpClient client = new TcpClient();

client.Connect("127.0.0.1", 1234); // IP, Port

// 2. Gửi/nhận dữ liệu

NetworkStream stream = client.GetStream();

byte[] data = Encoding.UTF8.GetBytes("Hello Server");

stream.Write(data, 0, data.Length); // gửi

byte[] buffer = new byte[1024];

int bytesRead = stream.Read(buffer, 0, buffer.Length); // nhận

string response = Encoding.UTF8.GetString(buffer, 0, bytesRead);

// 3. Đóng

stream.Close();

client.Close();

## **Mã C# – Server nhận file và lưu**

public partial class ServerForm : Form

{

TcpListener listener;

public ServerForm()

{

InitializeComponent();

}

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

{

Thread serverThread = new Thread(StartServer);

serverThread.IsBackground = true;

serverThread.Start();

}

void StartServer()

{

listener = new TcpListener(IPAddress.Any, 9000);

listener.Start();

AppendLog("Server đã khởi động...");

while (true)

{

TcpClient client = listener.AcceptTcpClient();

Thread t = new Thread(() => ReceiveFile(client));

t.IsBackground = true;

t.Start();

}

}

void ReceiveFile(TcpClient client)

{

NetworkStream ns = client.GetStream();

BinaryReader reader = new BinaryReader(ns);

try

{

string fileName = reader.ReadString(); // Đọc tên file

long fileLength = reader.ReadInt64(); // Đọc kích thước file

string savePath = Path.Combine(Application.StartupPath, "Received\_" + fileName);

using (FileStream fs = new FileStream(savePath, FileMode.Create, FileAccess.Write))

{

byte[] buffer = new byte[1024];

long totalBytesRead = 0;

while (totalBytesRead < fileLength)

{

int bytesRead = ns.Read(buffer, 0, buffer.Length);

fs.Write(buffer, 0, bytesRead);

totalBytesRead += bytesRead;

}

}

AppendLog("Đã nhận file: " + fileName);

}

catch (Exception ex)

{

AppendLog("Lỗi: " + ex.Message);

}

ns.Close();

client.Close();

}

void AppendLog(string msg)

{

if (txtLog.InvokeRequired)

{

txtLog.Invoke(new Action(() => txtLog.AppendText(msg + Environment.NewLine)));

}

else

{

txtLog.AppendText(msg + Environment.NewLine);

}

}

}

## ✅ **4. Mã C# – Client gửi file**

csharp

CopyEdit

using System;

using System.IO;

using System.Net.Sockets;

using System.Windows.Forms;

public partial class ClientForm : Form

{

public ClientForm()

{

InitializeComponent();

}

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

{

if (openFileDialog1.ShowDialog() == DialogResult.OK)

{

string filePath = openFileDialog1.FileName;

Thread sendThread = new Thread(() => SendFile(filePath));

sendThread.IsBackground = true;

sendThread.Start();

}

}

void SendFile(string filePath)

{

try

{

TcpClient client = new TcpClient("127.0.0.1", 9000);

NetworkStream ns = client.GetStream();

BinaryWriter writer = new BinaryWriter(ns);

FileInfo fi = new FileInfo(filePath);

writer.Write(fi.Name); // Gửi tên file

writer.Write(fi.Length); // Gửi kích thước

byte[] buffer = new byte[1024];

using (FileStream fs = fi.OpenRead())

{

int bytesRead;

while ((bytesRead = fs.Read(buffer, 0, buffer.Length)) > 0)

{

ns.Write(buffer, 0, bytesRead);

}

}

AppendLog("Đã gửi file: " + fi.Name);

ns.Close();

client.Close();

}

catch (Exception ex)

{

AppendLog("Lỗi: " + ex.Message);

}

}

void AppendLog(string msg)

{

if (txtLog.InvokeRequired)

{

txtLog.Invoke(new Action(() => txtLog.AppendText(msg + Environment.NewLine)));

}

else

{

txtLog.AppendText(msg + Environment.NewLine);

}

}

}

* .

// TCP Server dùng đa luồng

TcpListener server = new TcpListener(IPAddress.Any, 5000);

server.Start();

Console.WriteLine("Server started...");

while (true)

{

TcpClient client = server.AcceptTcpClient();

Console.WriteLine("New client connected.");

// Tạo luồng mới để xử lý client này

Thread clientThread = new Thread(() => HandleClient(client));

clientThread.Start();

}

void HandleClient(TcpClient client)

{

NetworkStream stream = client.GetStream();

StreamReader reader = new StreamReader(stream);

StreamWriter writer = new StreamWriter(stream) { AutoFlush = true };

try

{

while (true)

{

string msg = reader.ReadLine();

if (msg == null) break;

Console.WriteLine("Client said: " + msg);

writer.WriteLine("Server received: " + msg);

}

}

catch (Exception ex)

{

Console.WriteLine("Error: " + ex.Message);

}

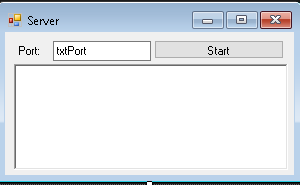
finally

{

client.Close();

}

}



public partial class Server : Form

{

TcpListener Listener;

int portN;

String rootDir="F:/";

public Server()

{

InitializeComponent();

}

private delegate void dlgAddInfo(string str);

private void AddInfo(string str)

{

if (this.rtxInfo.InvokeRequired)

{ this.Invoke(new dlgAddInfo(AddInfo), str); }

else

{ this.rtxInfo.AppendText(str+"\n\r"); }

}

private void CreJob(String[] request, StreamWriter sw)

{ //acc key pass message

String s = "100. Tin nhan luu thanh cong";

string fileName = rootDir +request[1]+"\_"+ request[2] + ".txt";

try

{

File.WriteAllText(fileName, request[3]);

}

catch(Exception ex)

{

s = "203. Khong the luu tin nhan.";

}

sw.WriteLine(s);

sw.Flush();

}

private void ReaJob(String[] request, StreamWriter sw)

{

string[] files = Directory.GetFiles(rootDir);

String s = "205. Tin nhan khong ton tai.";

foreach (string file in files)

{

if (Path.GetFileName(file).StartsWith(request[1]))

{

s = "204. Sai mat khau.";

string fileName = rootDir + request[1] + "\_" + request[2] + ".txt";

if (File.Exists(fileName))

{

String[] lines = File.ReadAllLines(fileName);

s = String.Join(" ", lines);

break;

}

}

}

sw.WriteLine(s);

sw.Flush();

}

private void UndefinedCommand(String[] request, StreamWriter sw)

{

String s = "Lỗi. Lệnh không tồn tại:"+ request[0];

byte[] data = new byte[s.Length];

sw.Write(s, 0, s.Length);

sw.Flush();

}

private void ThreadProc(object obj)

{

try {

var client = (TcpClient)obj;

StreamReader sr = new StreamReader(client.GetStream());

StreamWriter sw = new StreamWriter(client.GetStream());

sw.WriteLine("Chao mung ket noi toi SecretBox");

sw.Flush();

while (true)

{

string raw = sr.ReadLine();

string[] request = raw.Split('#');

AddInfo("Client Command:"+raw);

string command = "";

if (request.Length != 0)

command = request[0];

switch (command.ToUpper().Trim())

{

case "CRE"://tạo tin nhắn

{

CreJob(request, sw);

break;

}

case "REA"://đọc tin tin

{

ReaJob(request, sw);

break;

}

default:

{

UndefinedCommand(request, sw);

break;

}

}

}

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

public void ListenerThread()

{

Listener = null;

try

{

Listener = new TcpListener(IPAddress.Any, portN);

Listener.Start();

while (true)

{

TcpClient client = null;

NetworkStream netstream = null;

if (Listener.Pending())

{

client = Listener.AcceptTcpClient();

netstream = client.GetStream();

AddInfo("Kết nối với Client.");

ThreadPool.QueueUserWorkItem(ThreadProc, client);

// netstream.Close();

// client.Close();

}

}

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

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

{

Thread thdListener = new Thread(new ThreadStart(ListenerThread));

portN = int.Parse(txtPort.Text);

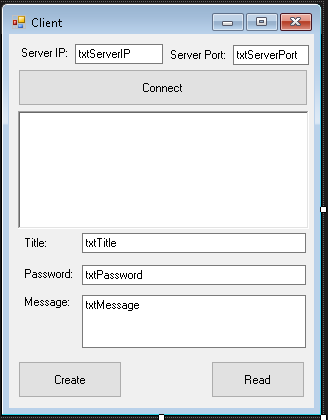
thdListener.Start();

thdListener.IsBackground=true;

AddInfo("Server đã khởi động");

}

}



public partial class Client : Form

{

IPEndPoint iep;

TcpClient client;

StreamReader sr;

StreamWriter sw;

public Client()

{

try

{

InitializeComponent();

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

private void AddInfo(String cnt)

{

rtxInfo.AppendText(cnt + "\n\r");

}

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

{

iep = new IPEndPoint(IPAddress.Parse(txtServerIP.Text), int.Parse(txtServerPort.Text));

client = new TcpClient();

client.Connect(iep);

sr = new StreamReader(client.GetStream());

sw = new StreamWriter(client.GetStream());

AddInfo(sr.ReadLine());

}

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

{

sw.WriteLine("REA#" + txtTitle.Text + "#" + txtPassword.Text);

sw.Flush();

AddInfo(sr.ReadLine());

}

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

{

sw.WriteLine("CRE#" + txtTitle.Text+"#" + txtPassword.Text + "#"+txtMessage.Text);

sw.Flush();

AddInfo(sr.ReadLine());

}

}