	COLLEGE OF COMPUTING AND INFORMATION SCIENCES		
	Final-Term Assessment Spring 2021 Semester		
Class Id	106383,4,5	**Course Title**	Network programming
Program	BSCS	**Campus / Shift**	North Campus / Morning
Date	29th – April 2021	**Total Marks**	40
Duration	03 hours	**Faculty Name**	Misbah Anwer
Student Id	63758	**Student Name**	Ali Salman Hassan

Instructions:

- Filling out Student-ID and Student-Name on exam header is mandatory.
- Do not remove or change any part of exam header or question paper.
- Write down your answers in given space or at the end of exam paper with proper title "Answer for Question# _ _".
- Answers should be formatted correctly (font size, alignment and etc.)
- Handwritten text or image should be on A4 size page with clear visibility of contents.
- Only PDF format is accepted (Student are advise to install necessary software)
- In case of CHEATING, COPIED material or any unfair means would result in negative marking or ZERO.
- A mandatory recorded viva session will be conducted to ascertain the quality of answer scripts were deemed necessary.

- **Caution:** Duration to perform Final-Term Assessment is **03 hours only**. **if you failed to upload answer sheet on LMS (in PDF format) within 03 hours limit, you would be considered as ABSENT/FAILED.**
- Mention complete code with output screenshots

QUESTION NO. 1 **[5+5 Marks]**

a. Explain the following code snippet also explain text in bold from 1 to 5. Rewrite this code for helper classes.

IPHostEntry local = Dns.GetHostByName(Dns.GetHostName());
IPEndPoint iep = new IPEndPoint(local.AddressList[0],**8000**); **(1)**
Socket **newserver (2)** = new Socket(AddressFamily.InterNetwork,
SocketType.Stream, ProtocolType.Tcp);
newserver.Bind(**iep**);**(3)**
newserver.Listen**(5);(4)**
Socket **newclient(5)** = newserver.Accept();

Answer for Question# 1a:

1 – 8000: Here the port number is defined as 8000. We define port number as a ProcessID to the certain instance.

2 – newserver: It is initializing a socket with the help of it`s class and we are passing default parameters in it for initializing the socket successfully.

3 – iep: We are using Bind method for Binding our End Point (IP END POINT) denoted as iep in the code.

4 – 5: We are using Listen method for listening the client from the server.

5 – newclient: We are using Accept method so we can accept the upcoming connections from the client.

HELPER CLASS CODE:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Net;
using System.Net.Sockets;

namespace q1ausinghelperclass
{
    class Program
    {
        static void Main(string[] args)
        {
            //SERVER (LISTENER)
            TcpListener newserver = new TcpListener(9050);
            newserver.Start();
            TcpClient newclient = newserver.AcceptTcpClient();
            NetworkStream ns = newclient.GetStream();
```

```
byte[] outbytes = Encoding.ASCII.GetBytes("Testing");
ns.Write(outbytes, 0, outbytes.Length);
byte[] inbytes = new byte[1024];
ns.Read(inbytes, 0, inbytes.Length);
string instring = Encoding.ASCII.GetString(inbytes);
Console.WriteLine(instring);
ns.Close();
newclient.Close();
Console.ReadLine();
        }
    }
}
```

b. Suppose you are tasked to improve the throughput and reliability of real-time video feeds collection from the surveillance cameras inside the city. You incorporate this using a network of intelligent nodes with storage to collect the video feed within an appropriate geographical area. Now, ignoring how cameras storing their feeds on intelligent node, you rather focus on how to collect video feeds from the intelligent nodes in real-time to a central control room datacenter. Create an application using C# network programming for above given scenario, Make suitable assumptions.

Answer for Question# 1b:

I will implement TCP according to the given scenario because here we can see that our task is to collect that video feeds from the intelligent nodes in real-time to a central control room datacenter. To improve the throughput and reliability of real-time video feeds collection, we use TCP instead of UDP. So, I am creating an application using helper class to implement above scenario.

CODE:

SERVER (CENTRAL CONTROL ROOM DATACENTER):

```
using System;
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;


namespace CENTRAL_CONTROL_ROOM_DATACENTER
{

    class Program
    {

        static void Main(string[] args)
        {

            IPEndPoint iep = new IPEndPoint(IPAddress.Loopback, 8000);

            int recv;
            byte[] data = new byte[1024];
            TcpListener newsock = new TcpListener(iep);
```

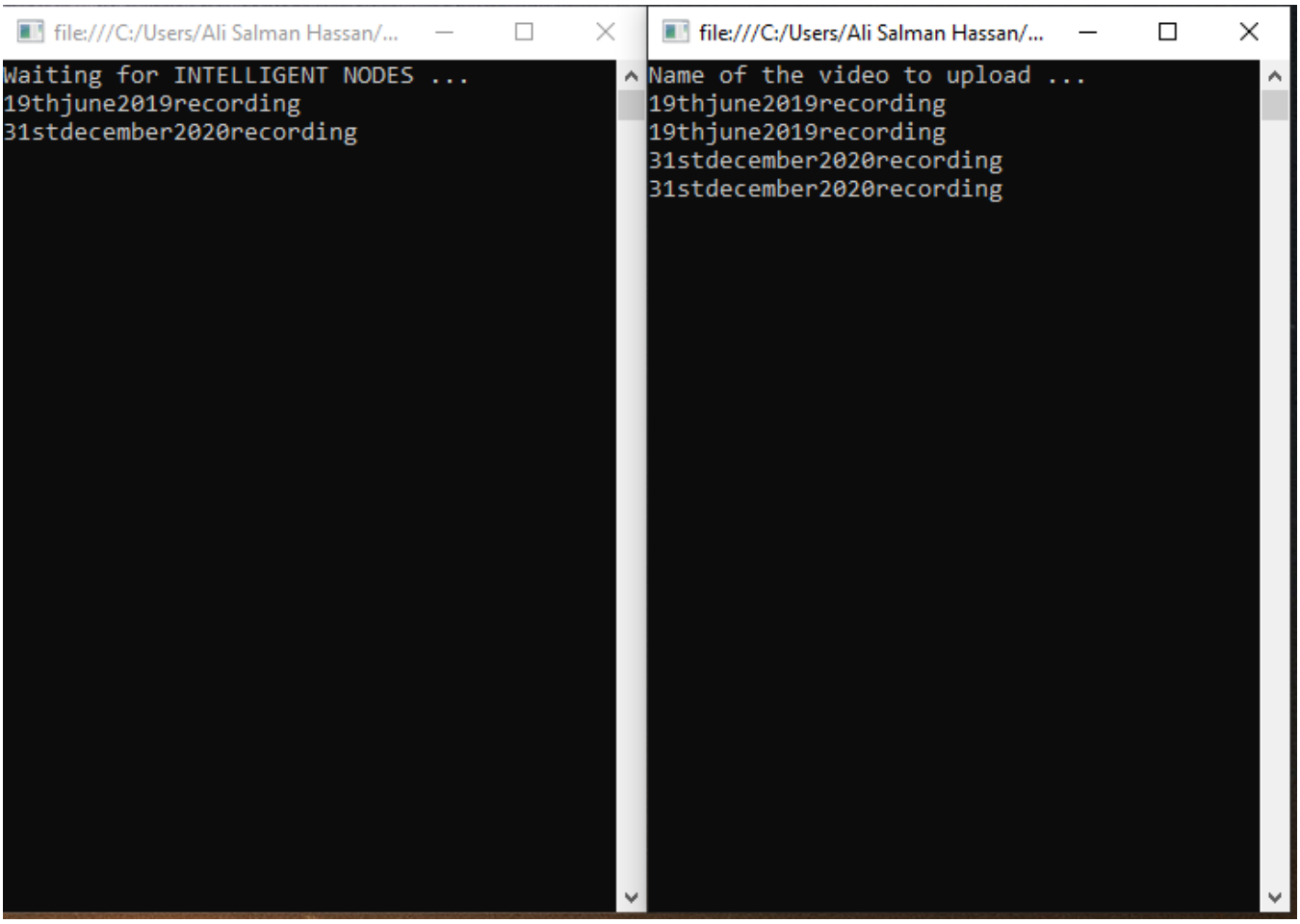
```
            newsock.Start();
            Console.WriteLine("Waiting for INTELLIGENT NODES ...");
            TcpClient client = newsock.AcceptTcpClient();
            NetworkStream ns = client.GetStream();
            string welcome = "Name of the video to upload ...";
            data = Encoding.ASCII.GetBytes(welcome);
            ns.Write(data, 0, data.Length);

            while (true)
            {

                data = new byte[1024];
                recv = ns.Read(data, 0, data.Length);

                if (recv == 0)
                {

                    break;

                }

                Console.WriteLine(
                Encoding.ASCII.GetString(data, 0, recv));
                ns.Write(data, 0, recv);

            }

            ns.Close();
            client.Close();
            newsock.Stop();

        }

    }

}
```

CLIENT (INTELLIGENT NODES):

```
using System;
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;

namespace INTELLIGENT_NODES
{

    class Program
    {

        static void Main(string[] args)
        {

            byte[] data = new byte[1024];
            string input, stringData;
            TcpClient server;

            try
            {
```

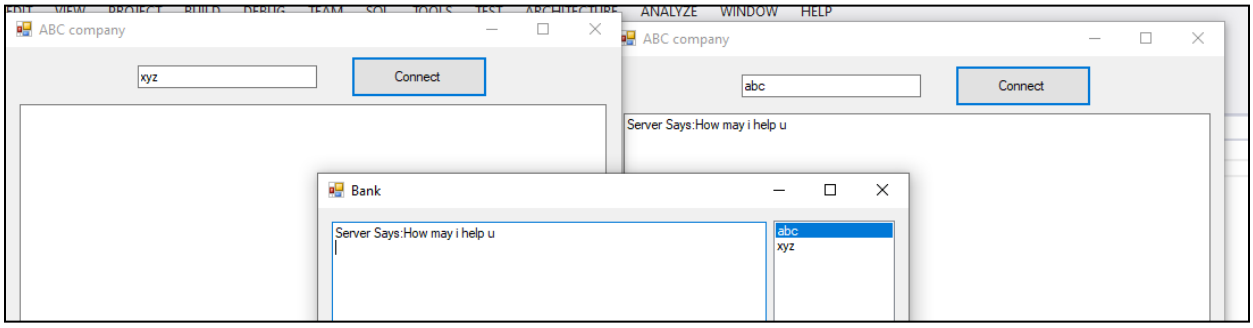
```
                server = new TcpClient("127.0.0.1", 8000);

            }

            catch (SocketException)
            {

                Console.WriteLine("Unable to connect to the CENTRAL CONTROL ROOM DATACENTER ...
Can`t upload the video ...");
                return;

            }

            NetworkStream ns = server.GetStream();
            int recv = ns.Read(data, 0, data.Length);
            stringData = Encoding.ASCII.GetString(data, 0, recv);
            Console.WriteLine(stringData);

            while (true)
            {

                input = Console.ReadLine();

                if (input == "end")
                {

                    break;

                }

                ns.Write(Encoding.ASCII.GetBytes(input), 0, input.Length);
                ns.Flush();
                data = new byte[1024];
                recv = ns.Read(data, 0, data.Length);
                stringData = Encoding.ASCII.GetString(data, 0, recv);
                Console.WriteLine(stringData);

            }

            Console.WriteLine("Disconnecting from CENTRAL CONTROL ROOM DATACENTER ...");
            ns.Close();
            server.Close();

        }

    }

}
```

OUTPUT:



QUESTION NO. 2 **[6+3+3 Marks]**

a. The CEO of organization need to have control on company from anywhere, the organization has three branches, branch one in Islamabad, branch two in Karachi and branch three in Lahore. You need to create asynchronous socket communication using Asynchronous methods.



Answer for Question# 2a:

CODE:

SERVER:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Net.Sockets;
using System.Net;

namespace ServerAsync
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            CheckForIllegalCrossThreadCalls = false;

            TcpListener listner = new TcpListener(new IPEndPoint(IPAddress.Loopback, 8001));
            listner.Start(10);

            listner.BeginAcceptTcpClient(new AsyncCallback(ClientConnected), listner);
        }

        Dictionary<string, TcpClient> clientlist = new Dictionary<string, TcpClient>();

        private void ClientConnected(IAsyncResult ar)
        {

            TcpListener listner = (TcpListener)ar.AsyncState;

            TcpClient client = listner.EndAcceptTcpClient(ar);

            NetworkStream ns = client.GetStream();

            byte[] arr = new byte[1024];

            int count = ns.Read(arr, 0, arr.Length);
            string clientName = ASCIIEncoding.ASCII.GetString(arr, 0, count);

            clientlist.Add(clientName, client);

            listBox1.Items.Add(clientName);
            arr = new byte[1024];

            ns.BeginRead(arr, 0, arr.Length, new AsyncCallback(ReadData), new object[] { ns, arr
});

            listner.BeginAcceptTcpClient(new AsyncCallback(ClientConnected), listner);


        }

        private void ReadData(IAsyncResult ar)
        {
```

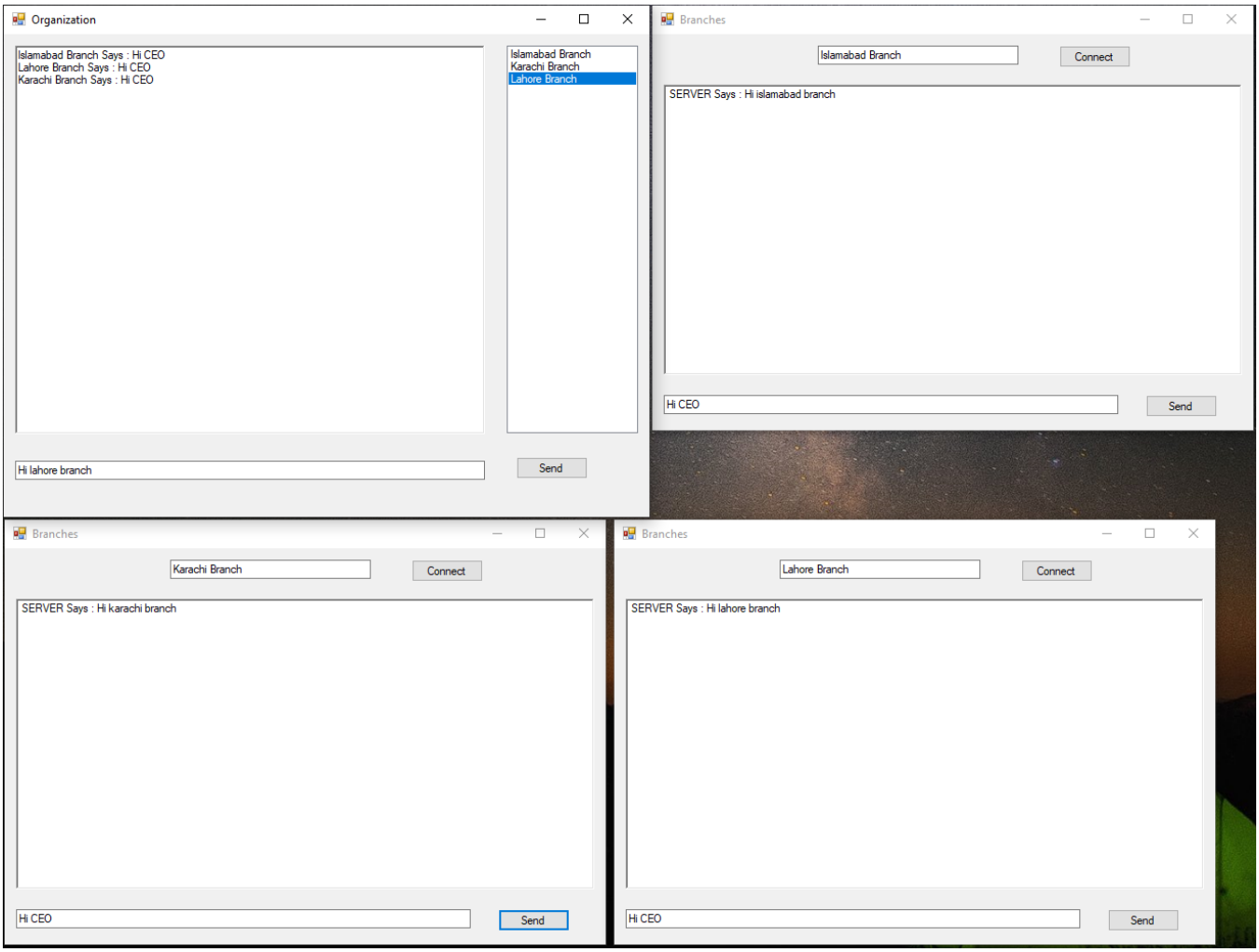
```
            object[] obj = (object[])ar.AsyncState;

            NetworkStream ns = (NetworkStream)obj[0];

            byte[] arr = (byte[])obj[1];

            int size = ns.EndRead(ar);

            richTextBox1.Text += ASCIIEncoding.ASCII.GetString(arr, 0, size) + "\n";

            ns.BeginRead(arr, 0, arr.Length, new AsyncCallback(ReadData), new object[] { ns, arr
});

        }

        private void SendMsg(String msg, string str)
        {
            NetworkStream ns = clientlist[str].GetStream();
            byte[] arr = ASCIIEncoding.ASCII.GetBytes(" SERVER Says : " + textBox1.Text);
            ns.Write(arr, 0, arr.Length);
            ns.Flush();

        }

        private void button1_Click_1(object sender, EventArgs e)
        {

                SendMsg(textBox1.Text, listBox1.SelectedItem.ToString());

        }
    }
}
```

CLIENT:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Net.Sockets;
using System.Net;

namespace ClientAsync
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
            CheckForIllegalCrossThreadCalls = false;
        }

        TcpClient client = new TcpClient();
        private void Form1_Load(object sender, EventArgs e)
        {
            CheckForIllegalCrossThreadCalls = false;
        }

        private void ReadData(IAsyncResult ar)
        {
```

```
            object[] obj = (object[])ar.AsyncState;
            NetworkStream ns = (NetworkStream)obj[0];
            byte[] arr = (byte[])obj[1];

            int size = ns.EndRead(ar);
            richTextBox1.Text += ASCIIEncoding.ASCII.GetString(arr, 0, size) + "\n";
            ns.BeginRead(arr, 0, arr.Length, new AsyncCallback(ReadData), new object[] { ns, arr
});

        }

        private void button1_Click_1(object sender, EventArgs e)
        {
            client.Connect(new IPEndPoint(IPAddress.Loopback, 8001));
            NetworkStream ns = client.GetStream();
            byte[] arr = ASCIIEncoding.ASCII.GetBytes(textBox1.Text);

            ns.Write(arr, 0, arr.Length);
            arr = new byte[1024];

            ns.BeginRead(arr, 0, arr.Length, new AsyncCallback(ReadData), new object[] { ns, arr
});
        }

        private void button2_Click_1(object sender, EventArgs e)
        {
            NetworkStream ns = client.GetStream();
            byte[] arr = ASCIIEncoding.ASCII.GetBytes(textBox1.Text + " Says : " +
textBox2.Text);
            ns.Write(arr, 0, arr.Length);
            ns.Flush();
        }
    }
}
```

OUTPUT:



b. Update part a for group communication among branches.

Answer for Question# 2b:

CODE:

SERVER:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Net.Sockets;
using System.Net;

namespace ServerAsync
{
    public partial class Form1 : Form
    {
        public Form1()
```

```
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            CheckForIllegalCrossThreadCalls = false;

            listBox1.Items.Add("Group Message");

            listBox1.SetSelected(0, true);

            TcpListener listner = new TcpListener(new IPEndPoint(IPAddress.Loopback, 8001));
            listner.Start(10);

            listner.BeginAcceptTcpClient(new AsyncCallback(ClientConnected), listner);
        }

        Dictionary<string, TcpClient> clientlist = new Dictionary<string, TcpClient>();

        private void ClientConnected(IAsyncResult ar)
        {

            TcpListener listner = (TcpListener)ar.AsyncState;

            TcpClient client = listner.EndAcceptTcpClient(ar);

            NetworkStream ns = client.GetStream();

            byte[] arr = new byte[1024];

            int count = ns.Read(arr, 0, arr.Length);
            string clientName = ASCIIEncoding.ASCII.GetString(arr, 0, count);

            clientlist.Add(clientName, client);

            listBox1.Items.Add(clientName);
            arr = new byte[1024];

            ns.BeginRead(arr, 0, arr.Length, new AsyncCallback(ReadData), new object[] { ns, arr
});

            listner.BeginAcceptTcpClient(new AsyncCallback(ClientConnected), listner);


        }

        private void ReadData(IAsyncResult ar)
        {

            object[] obj = (object[])ar.AsyncState;

            NetworkStream ns = (NetworkStream)obj[0];

            byte[] arr = (byte[])obj[1];

            int size = ns.EndRead(ar);

            richTextBox1.Text += ASCIIEncoding.ASCII.GetString(arr, 0, size) + "\n";

            ns.BeginRead(arr, 0, arr.Length, new AsyncCallback(ReadData), new object[] { ns, arr
});

        }
```

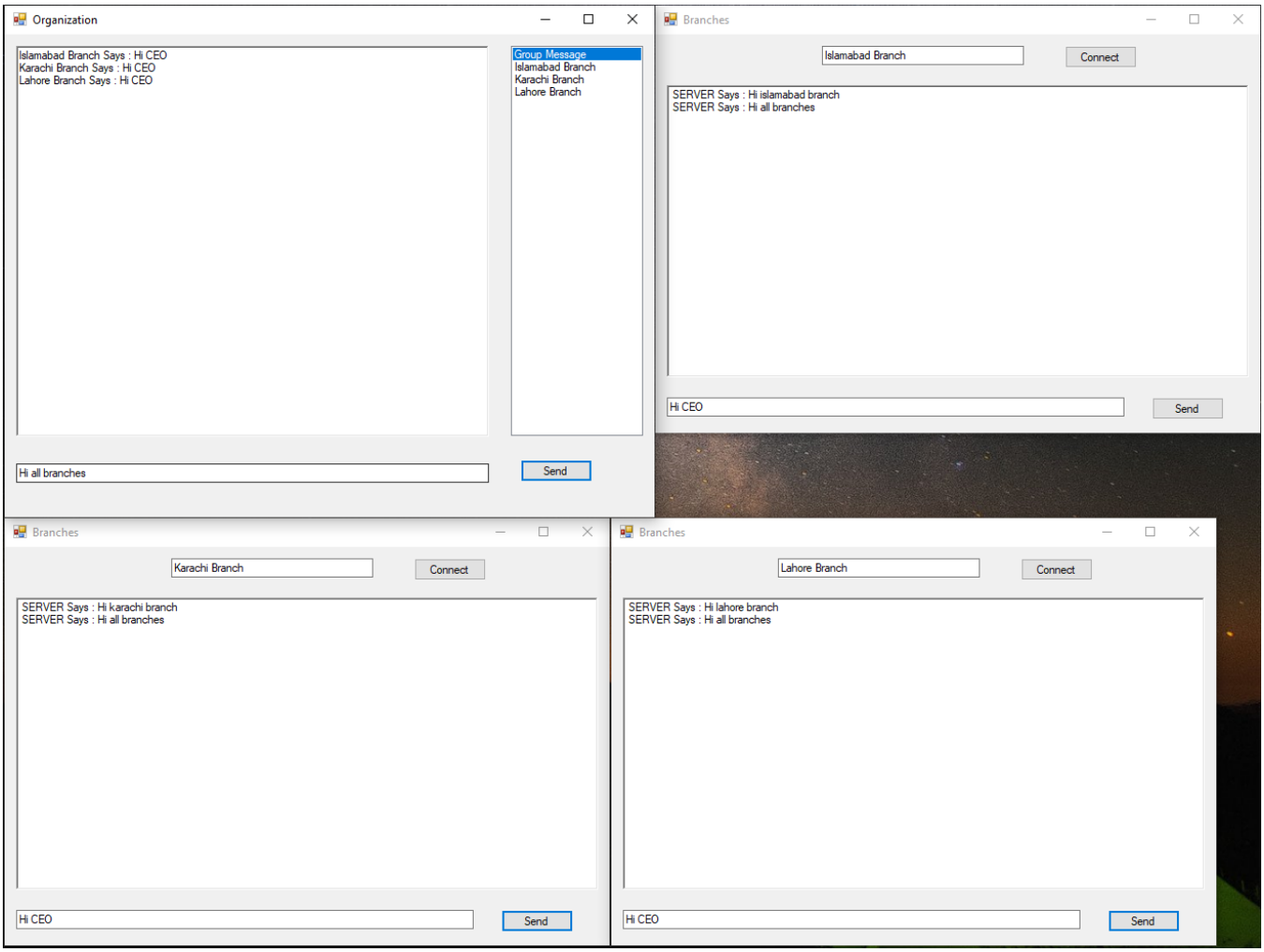
```
        private void GroupCommunication(String msg)
        {
            foreach (string str in clientlist.Keys)
            {
                NetworkStream ns = clientlist[str].GetStream();
                byte[] arr = ASCIIEncoding.ASCII.GetBytes(" SERVER Says : " + msg);
                ns.Write(arr, 0, arr.Length);
                ns.Flush();
            }
        }

        private void SendMsg(String msg, string str)
        {
            NetworkStream ns = clientlist[str].GetStream();
            byte[] arr = ASCIIEncoding.ASCII.GetBytes(" SERVER Says : " + textBox1.Text);
            ns.Write(arr, 0, arr.Length);
            ns.Flush();

        }

        private void button1_Click_1(object sender, EventArgs e)
        {
            if (listBox1.SelectedIndex.Equals(0))
            {
                GroupCommunication(textBox1.Text);
            }
            else
            {
                SendMsg(textBox1.Text, listBox1.SelectedItem.ToString());
            }
        }
    }
}
```

CLIENT:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Net.Sockets;
using System.Net;

namespace ClientAsync
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
            CheckForIllegalCrossThreadCalls = false;
        }

        TcpClient client = new TcpClient();
        private void Form1_Load(object sender, EventArgs e)
        {
            CheckForIllegalCrossThreadCalls = false;
```

```
        }

        private void ReadData(IAsyncResult ar)
        {
            object[] obj = (object[])ar.AsyncState;
            NetworkStream ns = (NetworkStream)obj[0];
            byte[] arr = (byte[])obj[1];

            int size = ns.EndRead(ar);
            richTextBox1.Text += ASCIIEncoding.ASCII.GetString(arr, 0, size) + "\n";
            ns.BeginRead(arr, 0, arr.Length, new AsyncCallback(ReadData), new object[] { ns, arr
});

        }

        private void button1_Click_1(object sender, EventArgs e)
        {
            client.Connect(new IPEndPoint(IPAddress.Loopback, 8001));
            NetworkStream ns = client.GetStream();
            byte[] arr = ASCIIEncoding.ASCII.GetBytes(textBox1.Text);

            ns.Write(arr, 0, arr.Length);
            arr = new byte[1024];

            ns.BeginRead(arr, 0, arr.Length, new AsyncCallback(ReadData), new object[] { ns, arr
});
        }

        private void button2_Click_1(object sender, EventArgs e)
        {
            NetworkStream ns = client.GetStream();
            byte[] arr = ASCIIEncoding.ASCII.GetBytes(textBox1.Text + " Says : " +
textBox2.Text);
            ns.Write(arr, 0, arr.Length);
            ns.Flush();
        }
    }
}
```

OUTPUT:



c. Define blocking, write a C# code that shows blocking also explain how to avoid blocking with the help of code

Answer for Question# 2c:

In case of TCP, server will wait until client generates request but if it fails to generate request, server will go in its dead or expire state also known as blocking state. This scenario does not occur in UDP because there is no need of any prior connection in UDP.

BLOCKING CODE:

SERVER:

```
using System;
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;
```

```
namespace Server
{

    class Program
    {

        static void Main(string[] args)
        {

            int recv;
            byte[] data = new byte[1024];
            IPEndPoint ipep = new IPEndPoint(IPAddress.Any, 9050);
            Socket newsock = new Socket(AddressFamily.InterNetwork, SocketType.Stream,
ProtocolType.Tcp);
            newsock.Bind(ipep);
            newsock.Listen(10);
            Console.WriteLine("Waiting for client ...");
            Socket client = newsock.Accept();
            IPEndPoint clientep =
            (IPEndPoint)client.RemoteEndPoint;
            Console.WriteLine("Connected with {0} at port {1}", clientep.Address,
clientep.Port);

            string welcome = "Welcome to my test server ...";
            data = Encoding.ASCII.GetBytes(welcome);
            client.Send(data, data.Length,
            SocketFlags.None);

            while (true)
            {

                data = new byte[1024];
                recv = client.Receive(data);
                if (recv == 0)
                {

                    break;

                }

                Console.WriteLine(
                Encoding.ASCII.GetString(data, 0, recv));
                client.Send(data, recv, SocketFlags.None);

            }

            Console.WriteLine("Disconnected from {0}",
            clientep.Address);
            client.Close();
            newsock.Close();

        }

    }

}
```

CLIENT:

```
using System;
using System;
```

```
using System.Net;
using System.Net.Sockets;
using System.Text;

namespace Client
{

    class Program
    {

        static void Main(string[] args)
        {

            byte[] data = new byte[1024];
            string input, stringData;
            IPEndPoint ipep = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 9050);
            Socket server = new Socket(AddressFamily.InterNetwork,SocketType.Stream,
ProtocolType.Tcp);

            try
            {

                server.Connect(ipep);

            }

            catch (SocketException e)
            {

                Console.WriteLine("Unable to connect to the server ...");
                Console.WriteLine(e.ToString());
                return;

            }
            int recv = server.Receive(data);
            stringData = Encoding.ASCII.GetString(data, 0, recv);
            Console.WriteLine(stringData);

            while (true)
            {

                input = Console.ReadLine();

                if (input == "end"){

                    break;

                }

                server.Send(Encoding.ASCII.GetBytes(input));
                data = new byte[1024];
                recv = server.Receive(data);
                stringData = Encoding.ASCII.GetString(data, 0, recv);
                Console.WriteLine(stringData);

            }

            Console.WriteLine("Disconnecting from server ...");
            server.Shutdown(SocketShutdown.Both);
            server.Close();

        }

    }
```

```
}
```

So, to overcome this blocking problem in TCP, we use Non-blocking I/0 methods like Non-blocking Sockets, Multiplexed Sockets and Winsock Non-blocking Socket Functions, etc. The Async Functions also use their methods to overcome the blocking problem.

NON-BLOCKING CODE:

SERVER:

```
using System;
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;
using System.Collections;


namespace Server
{

    class Program
    {

        static void Main(string[] args)
        {

            ArrayList socketList = new ArrayList(2);
            ArrayList copyList = new ArrayList(2);
            Socket main = new Socket(AddressFamily.InterNetwork, SocketType.Stream,
ProtocolType.Tcp);
            IPEndPoint iep = new IPEndPoint(IPAddress.Any, 9050);
            byte[] data = new byte[1024];
            string stringData;
            int recv;
            main.Bind(iep);
            main.Listen(2);
            Console.WriteLine("Waiting for clients ...");
            Socket client1 = main.Accept();
            IPEndPoint iep1 = (IPEndPoint)client1.RemoteEndPoint;
            client1.Send(Encoding.ASCII.GetBytes("Welcome to my server ..."));
            Console.WriteLine("Connected to {0}", iep1.ToString());
            socketList.Add(client1);
            Console.WriteLine("Waiting for other client ...");
            Socket client2 = main.Accept();
            IPEndPoint iep2 = (IPEndPoint)client2.RemoteEndPoint;
            client2.Send(Encoding.ASCII.GetBytes("Welcome to my server ..."));
            Console.WriteLine("Connected to {0}", iep2.ToString());
            socketList.Add(client2);
            main.Close();

            while (true)
            {
```

```
                copyList = new ArrayList(socketList);
                Console.WriteLine("Monitoring {0} sockets...", copyList.Count);
                Socket.Select(copyList, null, null, 10000000);

                foreach (Socket client in copyList)
                {

                    data = new byte[1024];
                    recv = client.Receive(data);
                    stringData = Encoding.ASCII.GetString(data, 0, recv);
                    Console.WriteLine("Received: {0}", stringData);

                    if (recv == 0)
                    {

                        iep = (IPEndPoint)client.RemoteEndPoint;
                        Console.WriteLine("Client {0} disconnected.", iep.ToString());
                        client.Close();
                        socketList.Remove(client);

                        if (socketList.Count == 0)
                        {

                            Console.WriteLine("Last client disconnected, bye");
                            return;

                        }

                    }

                    else
                    {

                        client.Send(data, recv, SocketFlags.None);

                    }

                }

            }

        }

    }

}
```

CLIENT:

```
using System;
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;

namespace Client
{

    class Program
    {
```

```
        static void Main(string[] args)
        {

            Socket socket = new Socket(AddressFamily.InterNetwork, SocketType.Stream,
ProtocolType.Tcp);
            IPEndPoint iep = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 9050);
            byte[] data = new byte[1024];
            string stringData;
            int recv;
            socket.Connect(iep);
            Console.WriteLine("Connected to server");
            recv = socket.Receive(data);
            stringData = Encoding.ASCII.GetString(data, 0, recv);
            Console.WriteLine("Received: {0}", stringData);

            while (true)
            {

                stringData = Console.ReadLine();

                if (stringData == "end")
                {

                    break;

                }

                data = Encoding.ASCII.GetBytes(stringData);
                socket.Send(data, data.Length, SocketFlags.None);
                data = new byte[1024];
                recv = socket.Receive(data);
                stringData = Encoding.ASCII.GetString(data, 0, recv);
                Console.WriteLine("Received: {0}", stringData);

            }

            socket.Close();

        }

    }

}
```

QUESTION NO. 3 **[4+4 Marks]**

a. How TCP ensure packet reliability? How we can acknowledge the UDP packets? Explain and write code. What are the parameters needed for four synchronous socket methods defined in Dns class?

Answer for Question# 3a:

It ensures its reliability by using sequence and acknowledgement numbers.

We can`t acknowledge UDP because it does not have any type of acknowledgement but we can create a system by creating a packet retransmission system with a timer.

In socket, we define domain type and protocol in its parameters it uses IP-Address from the DNS class and gethostbyname() method easily.

b. Explain the issue in the following code also fix the issue (if any) and rewrite the client code

```
byte[] data = new byte[1024];
string stringData;
IPEndPoint ipep = new IPEndPoint(
IPAddress.Parse("127.0.0.1"), 9050);
Socket server = new Socket(AddressFamily.InterNetwork,
SocketType.Stream, ProtocolType.Tcp);
try
{
    server.Connect(ipep);
}
catch (SocketException e)
{
    Console.WriteLine("Unable to connect to server.");
    Console.WriteLine(e.ToString());
    return;
}
int recv = server.Receive(data);
stringData = Encoding.ASCII.GetString(data, 0, recv);
Console.WriteLine(stringData);
server.Send(Encoding.ASCII.GetBytes("message 1"));
server.Send(Encoding.ASCII.GetBytes("message 2"));
server.Send(Encoding.ASCII.GetBytes("message 3"));
server.Send(Encoding.ASCII.GetBytes("message 4"));
server.Send(Encoding.ASCII.GetBytes("message 5"));
Console.WriteLine("Disconnecting from server...");
server.Shutdown(SocketShutdown.Both);
server.Close();
```

Answer for Question# 3b:

CODE:

```
using System;
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;

namespace Code
```

```
{
    class Program
    {
        private static int SendData(Socket s, byte[] data)
        {
            int total = 0;
            int size = data.Length;
            int dataleft = size;
            int sent;
            while (total < size)
            {
                sent = s.Send(data, total, dataleft, SocketFlags.None);
                total += sent;
                dataleft -= sent;
            }
            return total;
        }
        private static byte[] ReceiveData(Socket s, int size)
        {
            int total = 0;
            int dataleft = size;
            byte[] data = new byte[size];
            int recv;
            while (total < size)
            {
                recv = s.Receive(data, total, dataleft, 0);
                if (recv == 0)
                {
                    data = Encoding.ASCII.GetBytes("end ");
                    break;
                }
                total += recv;
                dataleft -= recv;
            }
            return data;
        }

        static void Main(string[] args)
        {
            byte[] data = new byte[1024];
            int sent;
            IPEndPoint ipep = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 9050);
            Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            try
            {
                server.Connect(ipep);
            }
            catch (SocketException e)
            {
                Console.WriteLine("Unable to connect to the server ...");
                Console.WriteLine(e.ToString());
                return;
            }
            int recv = server.Receive(data);
            string stringData = Encoding.ASCII.GetString(data, 0, recv);
            Console.WriteLine(stringData);
            sent = SendData(server, Encoding.ASCII.GetBytes("message 1"));
            sent = SendData(server, Encoding.ASCII.GetBytes("message 2"));
            sent = SendData(server, Encoding.ASCII.GetBytes("message 3"));
            sent = SendData(server, Encoding.ASCII.GetBytes("message 4"));
            sent = SendData(server, Encoding.ASCII.GetBytes("message 5"));
            Console.WriteLine("Disconnecting from the server ...");
            server.Shutdown(SocketShutdown.Both);
```

```
            server.Close();

        }
    }
}
```

QUESTION NO. 4 **[3+4+3 Marks]**

a. You are required to create a multimedia application for NP online Exam query session that include the following features.
   i. Instructor can upload the paper (File sharing .doc or . pdf)

Answer for Question# 4a(i):

In this scenario, I will use the logic of my application of my project of Network Programming "DROPBOX SYNCHORONIZATION MECHANISM" in which we can make the backup of the selected folder in any drive in our PC. So, the scenario will be that instructor will copy the paper in that folder and will upload the paper (Backup logic) on the drive from which the students can access the paper.

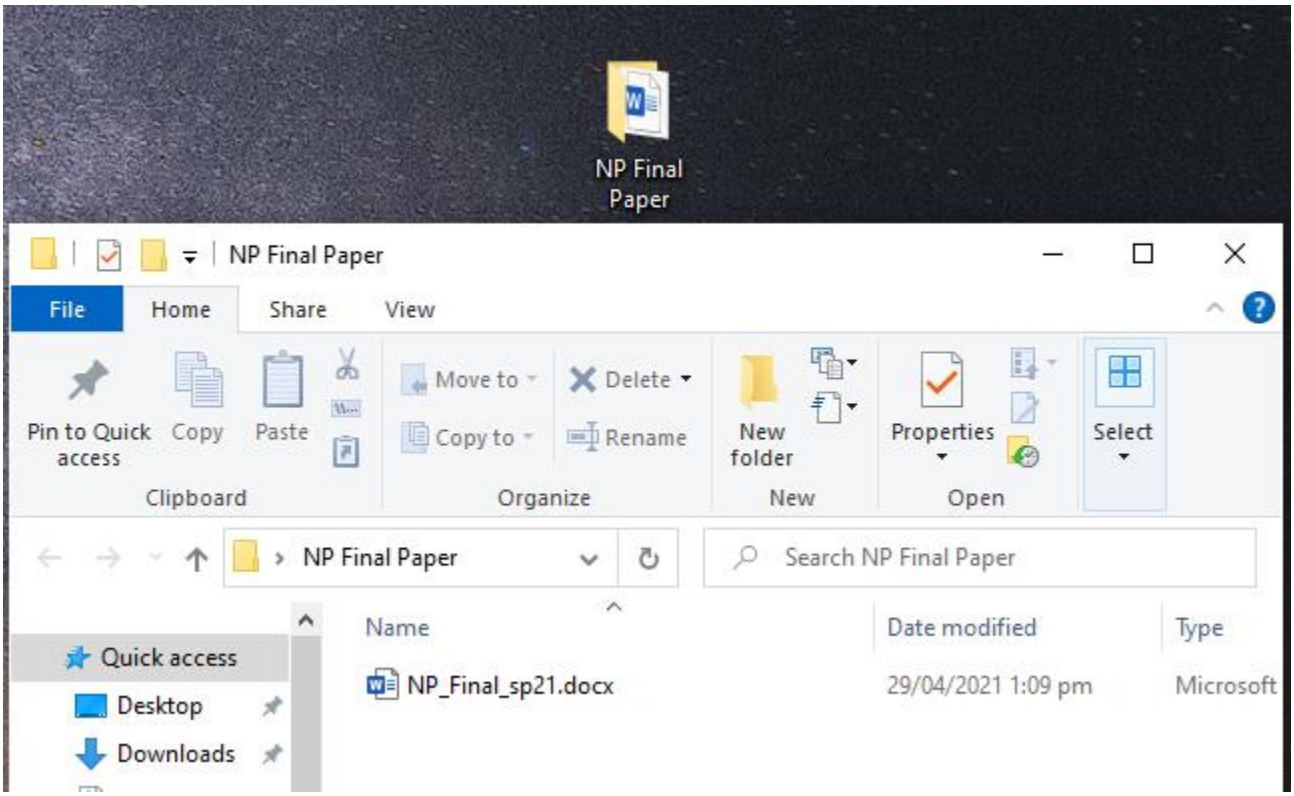
CODE:

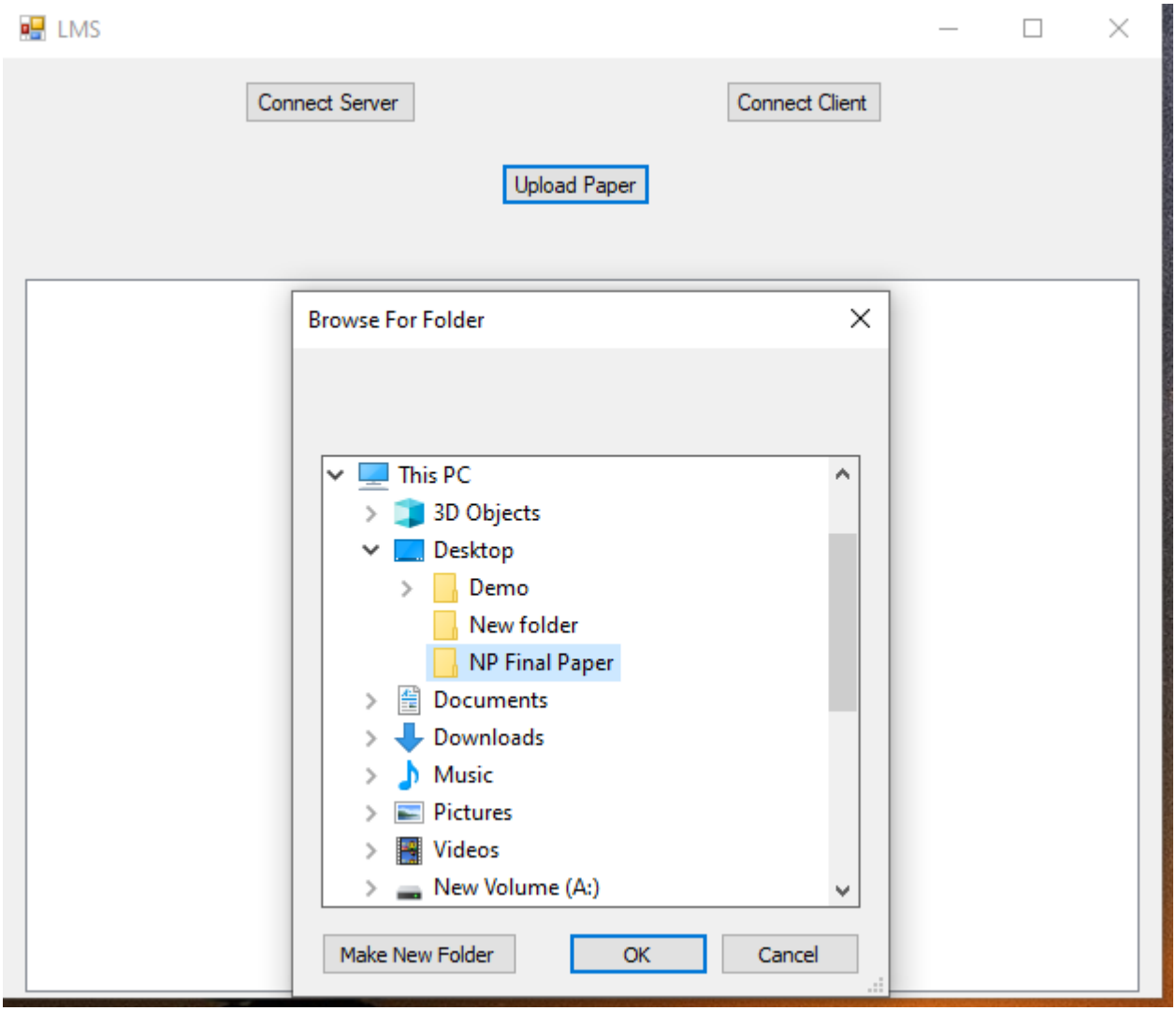
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Net;
using System.Net.Sockets;
using System.Text;
using System.Threading;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace LMS{

    public partial class Form1 : Form{


        static TcpListener server;
        static TcpClient tc;


        public Form1(){

            InitializeComponent();
            server = new TcpListener(IPAddress.Loopback, 8001);
            connect_client.Enabled = false;
            select_folder.Enabled = false;

            if (!Directory.Exists(strDesPath)){
```

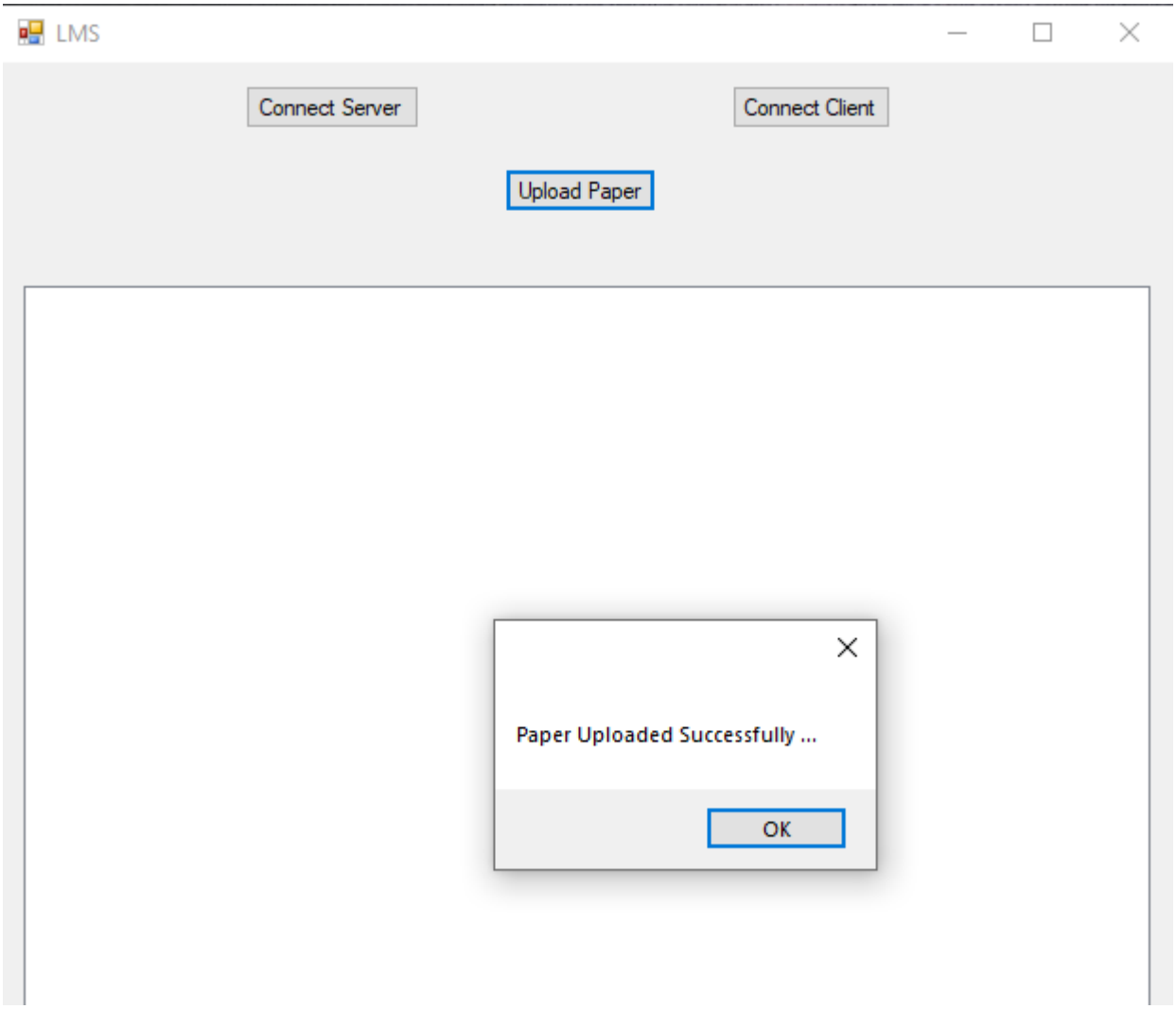
```
                Directory.CreateDirectory(strDesPath);

            }

        }


        private string strDesPath = @"A:\LMS DATABASE";
        private static string strSourcePath;


        private void connect_server_Click(object sender, EventArgs e){

            CheckForIllegalCrossThreadCalls = false;
            server.Start(10);
            connect_client.Enabled = true;

        }


        private void connect_client_Click(object sender, EventArgs e){

            IPEndPoint point = new IPEndPoint(IPAddress.Loopback, 8002);
            tc = new TcpClient(point);
            tc.Connect(IPAddress.Loopback, 8001);
            select_folder.Enabled = true;

        }


        private void upload_paper_Click(object sender, EventArgs e){

            if (tc.Connected){

                select_folder.Enabled = true;

                syncAllFilesFirstTime();

                MessageBox.Show("Paper Uploaded Successfully ...");

            }

        }


        private void syncAllFilesFirstTime(){

            if (folderBrowserDialog1.ShowDialog() == DialogResult.OK){

                strSourcePath = folderBrowserDialog1.SelectedPath;

                string[] arrFiles = Directory.GetFiles(strSourcePath); //Storing list of files
from sourcepath in an array

                foreach (string sourceFiles in arrFiles)
                {

                    string strFileName = Path.GetFileName(sourceFiles); //Checking filenames
                    string strDesFilePath = string.Format(@"{0}\{1}", strDesPath, strFileName);

                    if (!File.Exists(strDesFilePath)){ //Checking that the destination path
contatins the same files
```

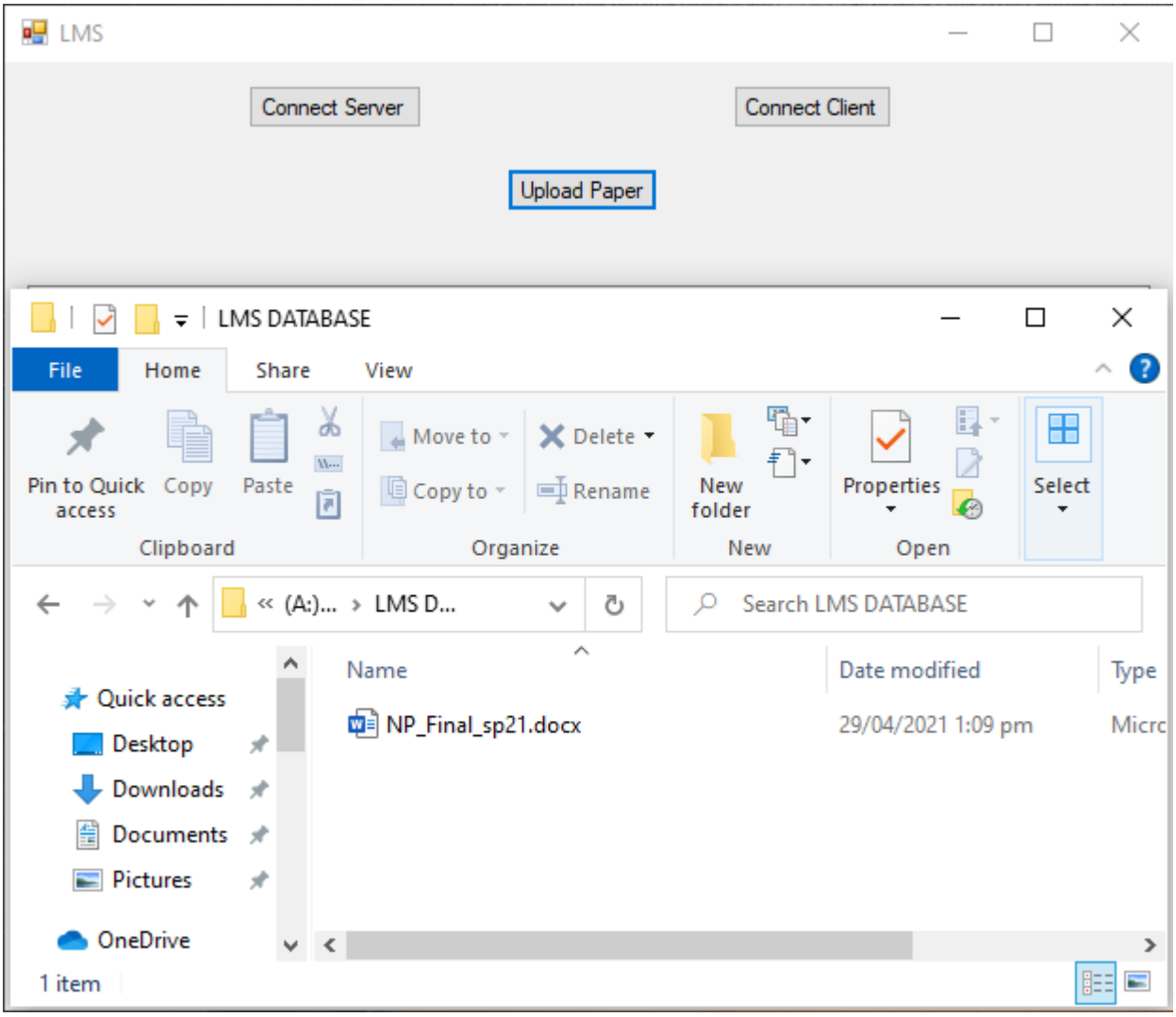
```
                    File.Copy(sourceFiles, strDesFilePath, true);
                }
            }
        }
    }
}
```

OUTPUT:




LMS
Connect Server
Connect Client
Upload Paper
Browse For Folder
This PC
3D Objects
Desktop
Demo
New folder
NP Final Paper
Documents
Downloads
Music
Pictures
Videos
New Volume (A:)
Make New Folder
OK
Cancel


LMS
Connect Server
Connect Client
Upload Paper
Paper Uploaded Successfully ...
OK



ii. Instructor can Explain the paper and Students can ask queries via chat and voice call.

Answer for Question# 4a(ii):

CODE:

SERVER:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Net.Sockets;
using System.Net;

namespace ServerAsync
{
```

```
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            CheckForIllegalCrossThreadCalls = false;

            listBox1.Items.Add("Group Chat OR Group Call");

            listBox1.SetSelected(0, true);

            TcpListener listner = new TcpListener(new IPEndPoint(IPAddress.Loopback, 8001));
            listner.Start(10);

            listner.BeginAcceptTcpClient(new AsyncCallback(ClientConnected), listner);
        }

        Dictionary<string, TcpClient> clientlist = new Dictionary<string, TcpClient>();

        private void ClientConnected(IAsyncResult ar)
        {

            TcpListener listner = (TcpListener)ar.AsyncState;

            TcpClient client = listner.EndAcceptTcpClient(ar);

            NetworkStream ns = client.GetStream();

            byte[] arr = new byte[1024];

            int count = ns.Read(arr, 0, arr.Length);
            string clientName = ASCIIEncoding.ASCII.GetString(arr, 0, count);

            clientlist.Add(clientName, client);

            listBox1.Items.Add(clientName);
            arr = new byte[1024];

            ns.BeginRead(arr, 0, arr.Length, new AsyncCallback(ReadData), new object[] { ns, arr
});

            listner.BeginAcceptTcpClient(new AsyncCallback(ClientConnected), listner);


        }

        private void ReadData(IAsyncResult ar)
        {

            object[] obj = (object[])ar.AsyncState;

            NetworkStream ns = (NetworkStream)obj[0];

            byte[] arr = (byte[])obj[1];

            int size = ns.EndRead(ar);

            richTextBox1.Text += ASCIIEncoding.ASCII.GetString(arr, 0, size) + "\n";
```

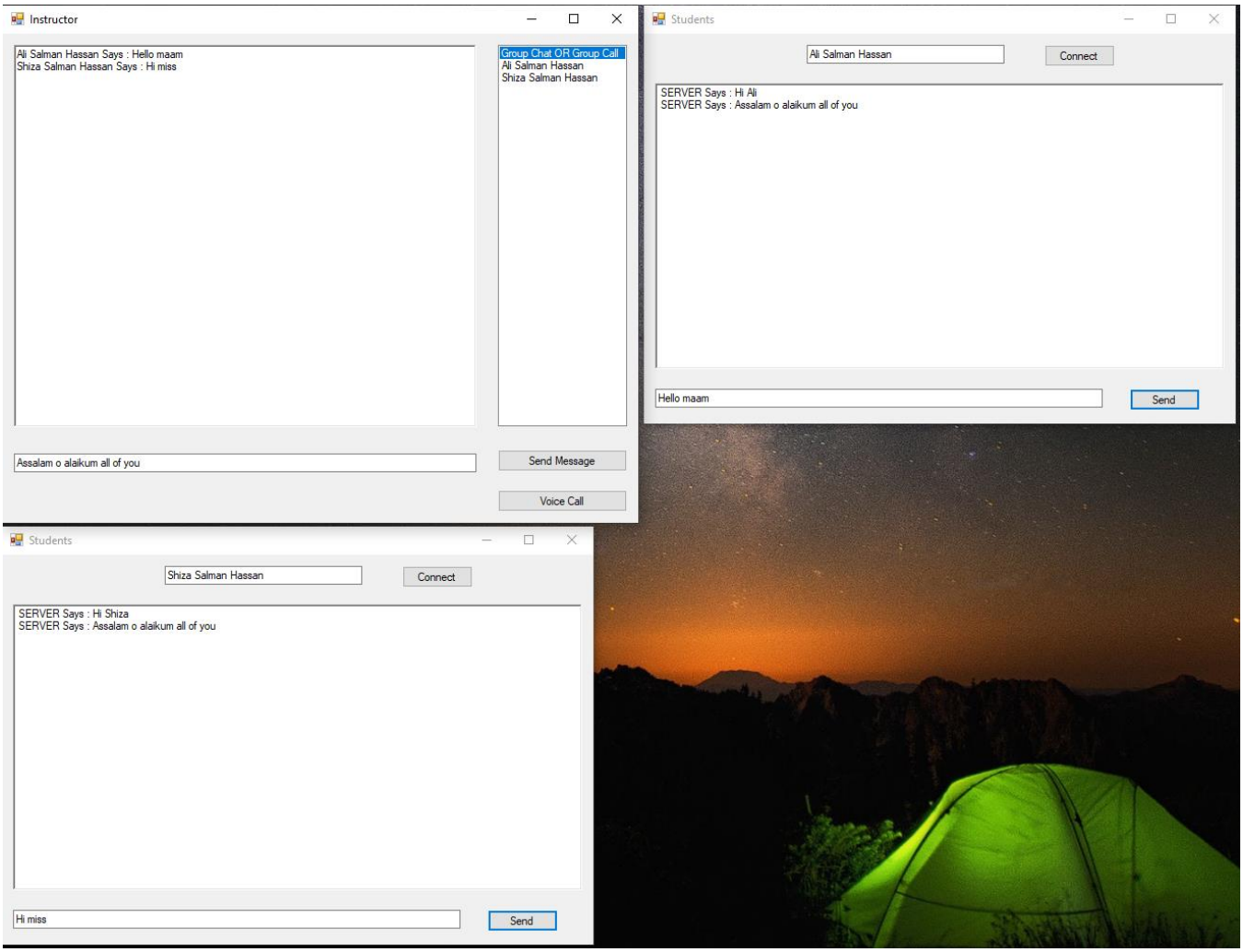
```
            ns.BeginRead(arr, 0, arr.Length, new AsyncCallback(ReadData), new object[] { ns, arr
});

        }

        private void GroupCommunication(String msg)
        {
            foreach (string str in clientlist.Keys)
            {
                NetworkStream ns = clientlist[str].GetStream();
                byte[] arr = ASCIIEncoding.ASCII.GetBytes(" SERVER Says : " + msg);
                ns.Write(arr, 0, arr.Length);
                ns.Flush();
            }
        }

        private void SendMsg(String msg, string str)
        {
            NetworkStream ns = clientlist[str].GetStream();
            byte[] arr = ASCIIEncoding.ASCII.GetBytes(" SERVER Says : " + textBox1.Text);
            ns.Write(arr, 0, arr.Length);
            ns.Flush();

        }

        private void button1_Click_1(object sender, EventArgs e) //CHAT
        {
            if (listBox1.SelectedIndex.Equals(0))
            {
                GroupCommunication(textBox1.Text);
            }
            else
            {
                SendMsg(textBox1.Text, listBox1.SelectedItem.ToString());
            }
        }

        private void button2_Click(object sender, EventArgs e) //VOICE CALL
        {
            if (listBox1.SelectedIndex.Equals(0))
            {
                MessageBox.Show("CALLING ALL STUDENTS ...", "GROUP VOICE CALLING",
MessageBoxButtons.OK);
            }
            else
            {
                MessageBox.Show("CALLING " + listBox1.SelectedItem.ToString()+ "...", "VOICE
CALL", MessageBoxButtons.OK);
            }
        }
    }
}
```

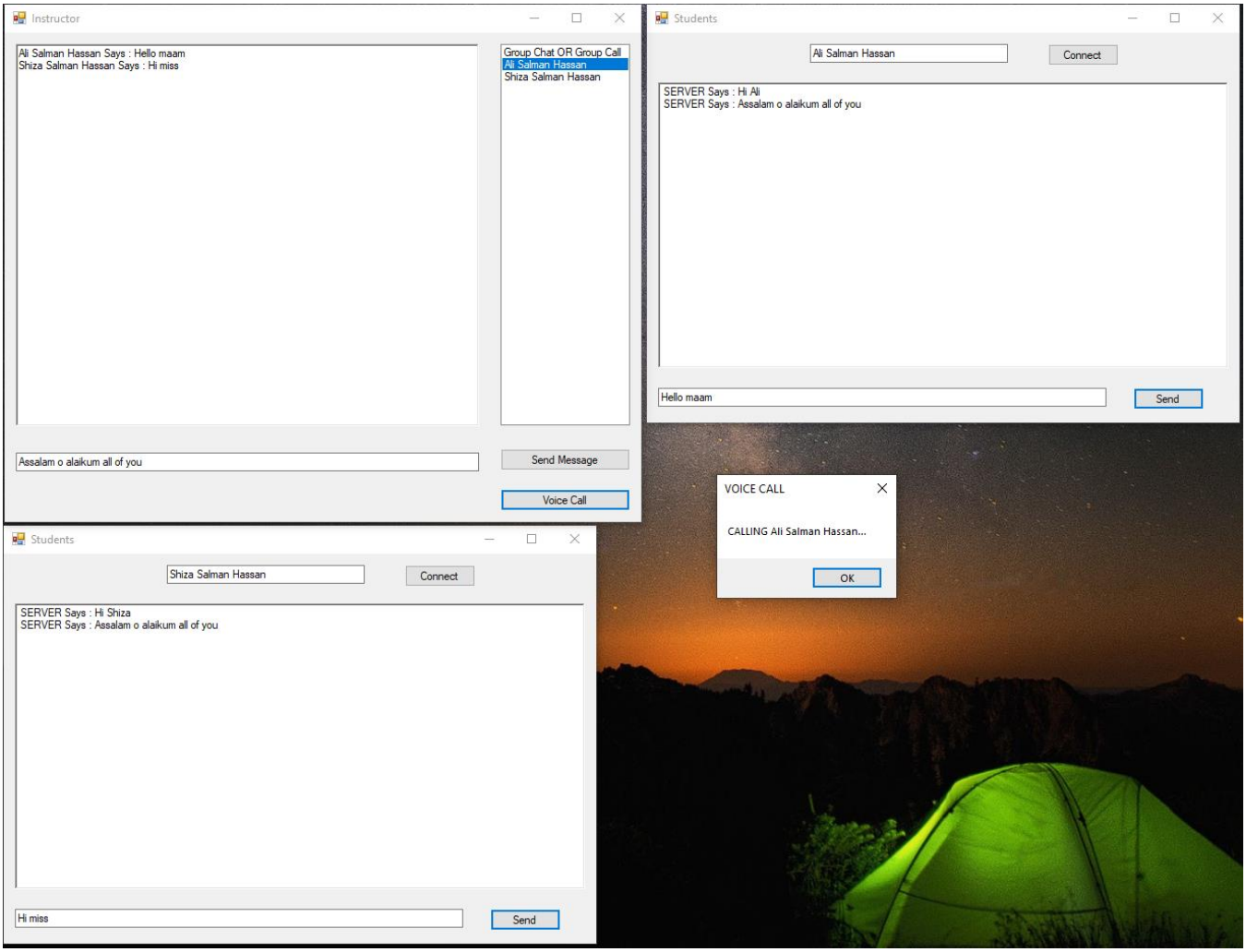
CLIENT:

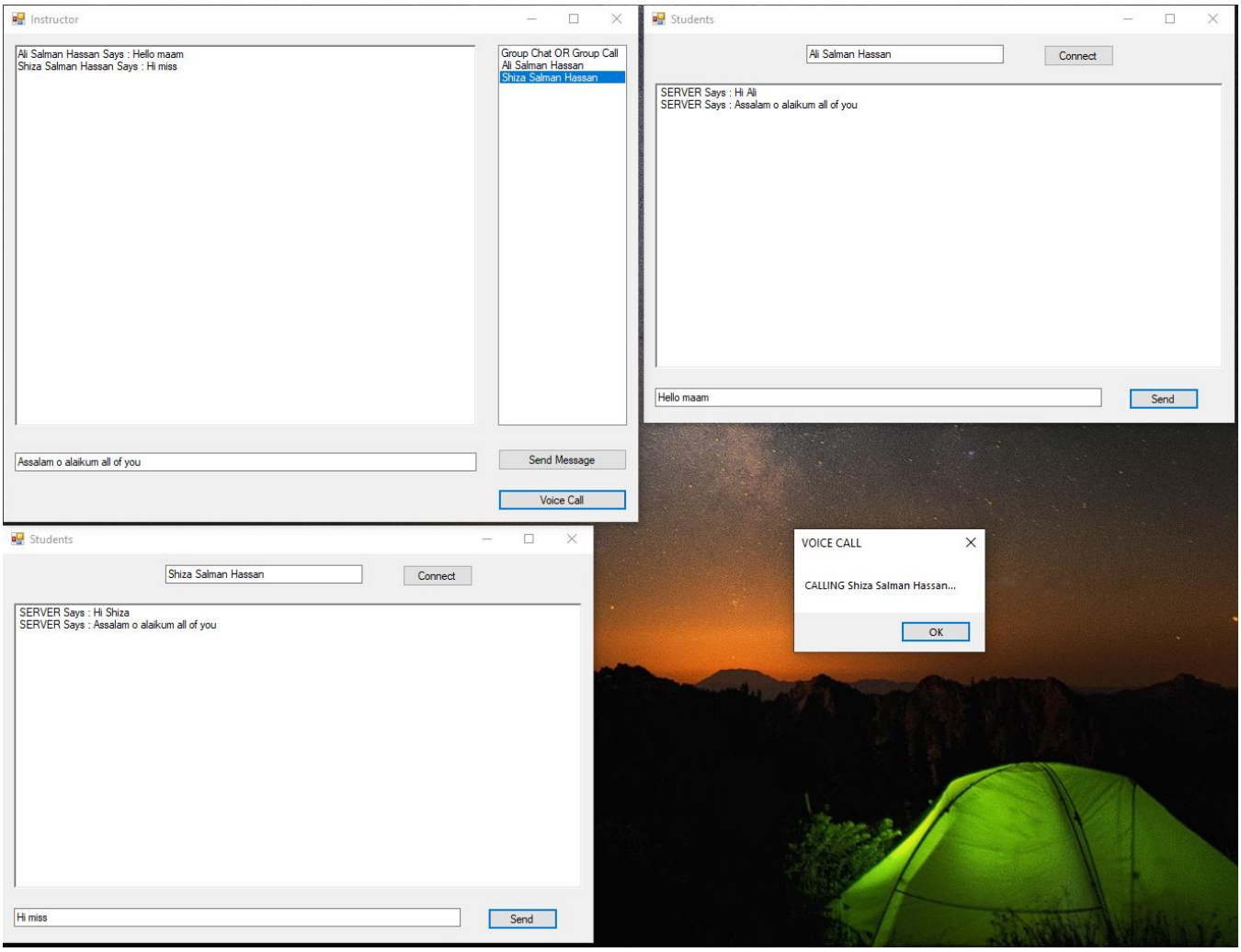
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
```

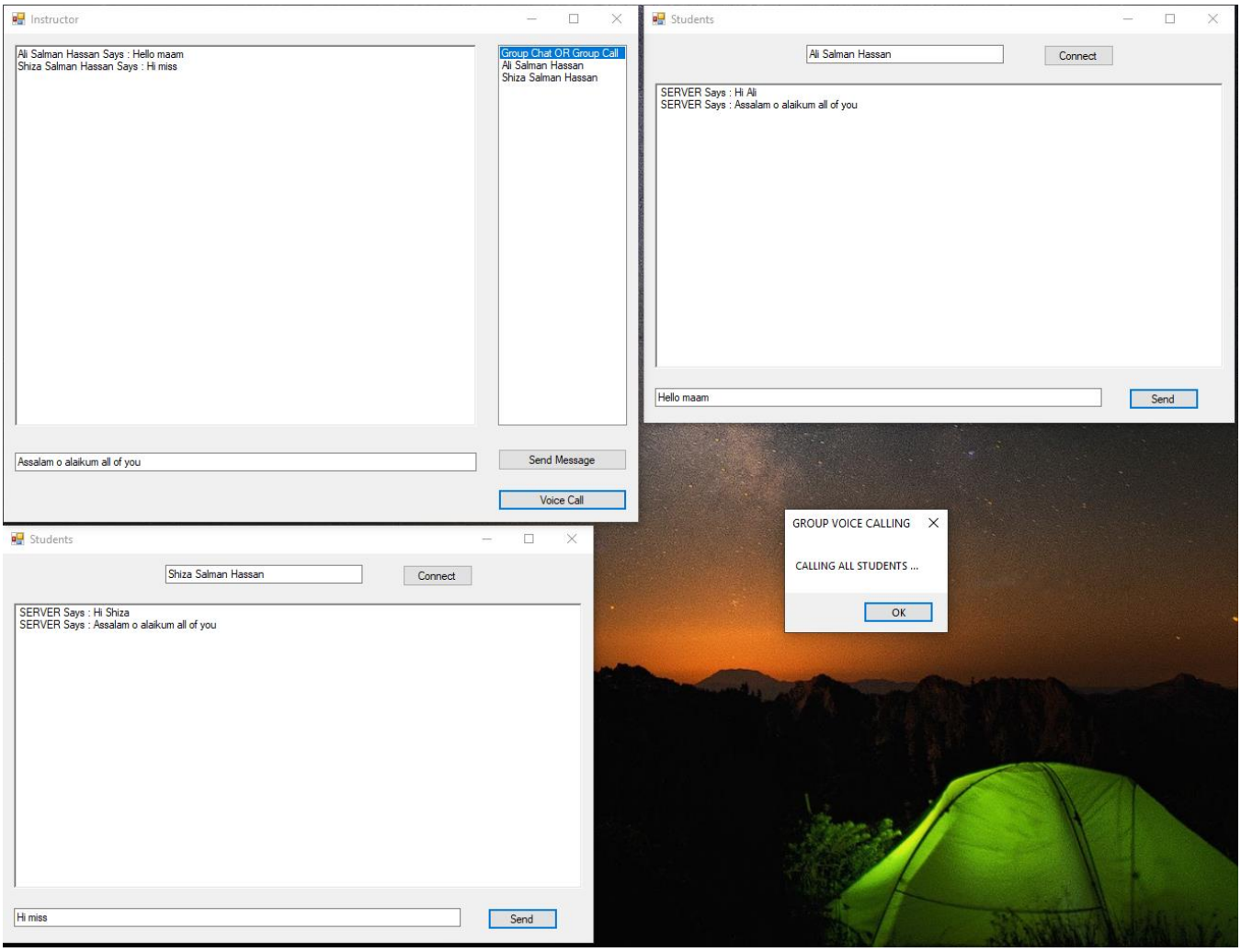
```
using System.Net.Sockets;
using System.Net;

namespace ClientAsync
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
            CheckForIllegalCrossThreadCalls = false;
        }

        TcpClient client = new TcpClient();
        private void Form1_Load(object sender, EventArgs e)
        {
            CheckForIllegalCrossThreadCalls = false;
        }

        private void ReadData(IAsyncResult ar)
        {
            object[] obj = (object[])ar.AsyncState;
            NetworkStream ns = (NetworkStream)obj[0];
            byte[] arr = (byte[])obj[1];

            int size = ns.EndRead(ar);
            richTextBox1.Text += ASCIIEncoding.ASCII.GetString(arr, 0, size) + "\n";
            ns.BeginRead(arr, 0, arr.Length, new AsyncCallback(ReadData), new object[] { ns, arr
});

        }

        private void button1_Click_1(object sender, EventArgs e)
        {
            client.Connect(new IPEndPoint(IPAddress.Loopback, 8001));
            NetworkStream ns = client.GetStream();
            byte[] arr = ASCIIEncoding.ASCII.GetBytes(textBox1.Text);

            ns.Write(arr, 0, arr.Length);
            arr = new byte[1024];

            ns.BeginRead(arr, 0, arr.Length, new AsyncCallback(ReadData), new object[] { ns, arr
});
        }

        private void button2_Click_1(object sender, EventArgs e)
        {
            NetworkStream ns = client.GetStream();
            byte[] arr = ASCIIEncoding.ASCII.GetBytes(textBox1.Text + " Says : " +
textBox2.Text);
            ns.Write(arr, 0, arr.Length);
            ns.Flush();
        }
    }
}
```

OUTPUT:


Instructor
Ali Salman Hassan Says : Hello maam
Shiza Salman Hassan Says : Hi miss
Group Chat OR Group Call
Ali Salman Hassan
Shiza Salman Hassan
Assalam o alaikum all of you
Send Message
Voice Call
Students
Ali Salman Hassan
Connect
SERVER Says : Hi Ali
SERVER Says : Assalam o alaikum all of you
Hello maam
Send
Students
Shiza Salman Hassan
Connect
SERVER Says : Hi Shiza
SERVER Says : Assalam o alaikum all of you
Hi miss
Send


Instructor
Ali Salman Hassan Says : Hello maam
Shiza Salman Hassan Says : Hi miss
Group Chat OR Group Call
Ali Salman Hassan
Shiza Salman Hassan
Assalam o alaikum all of you
Send Message
Voice Call
Students
Ali Salman Hassan
Connect
SERVER Says : Hi Ali
SERVER Says : Assalam o alaikum all of you
Hello maam
Send
VOICE CALL
CALLING Ali Salman Hassan...
OK
Students
Shiza Salman Hassan
Connect
SERVER Says : Hi Shiza
SERVER Says : Assalam o alaikum all of you
Hi miss
Send


Instructor
Ali Salman Hassan Says : Hello maam
Shiza Salman Hassan Says : Hi miss
Group Chat OR Group Call
Ali Salman Hassan
Shiza Salman Hassan
Assalam o alaikum all of you
Send Message
Voice Call
Students
Ali Salman Hassan
Connect
SERVER Says : Hi Ali
SERVER Says : Assalam o alaikum all of you
Hello maam
Send
Students
Shiza Salman Hassan
Connect
SERVER Says : Hi Shiza
SERVER Says : Assalam o alaikum all of you
Hi miss
Send
VOICE CALL
CALLING Shiza Salman Hassan...
OK



b. PAF-KIET university north campus wants to transmit message to all the student about on campus examination and other updates using network functions via smtp, pop3 and IMAP. Create an application for sending the messages using mentioned protocols

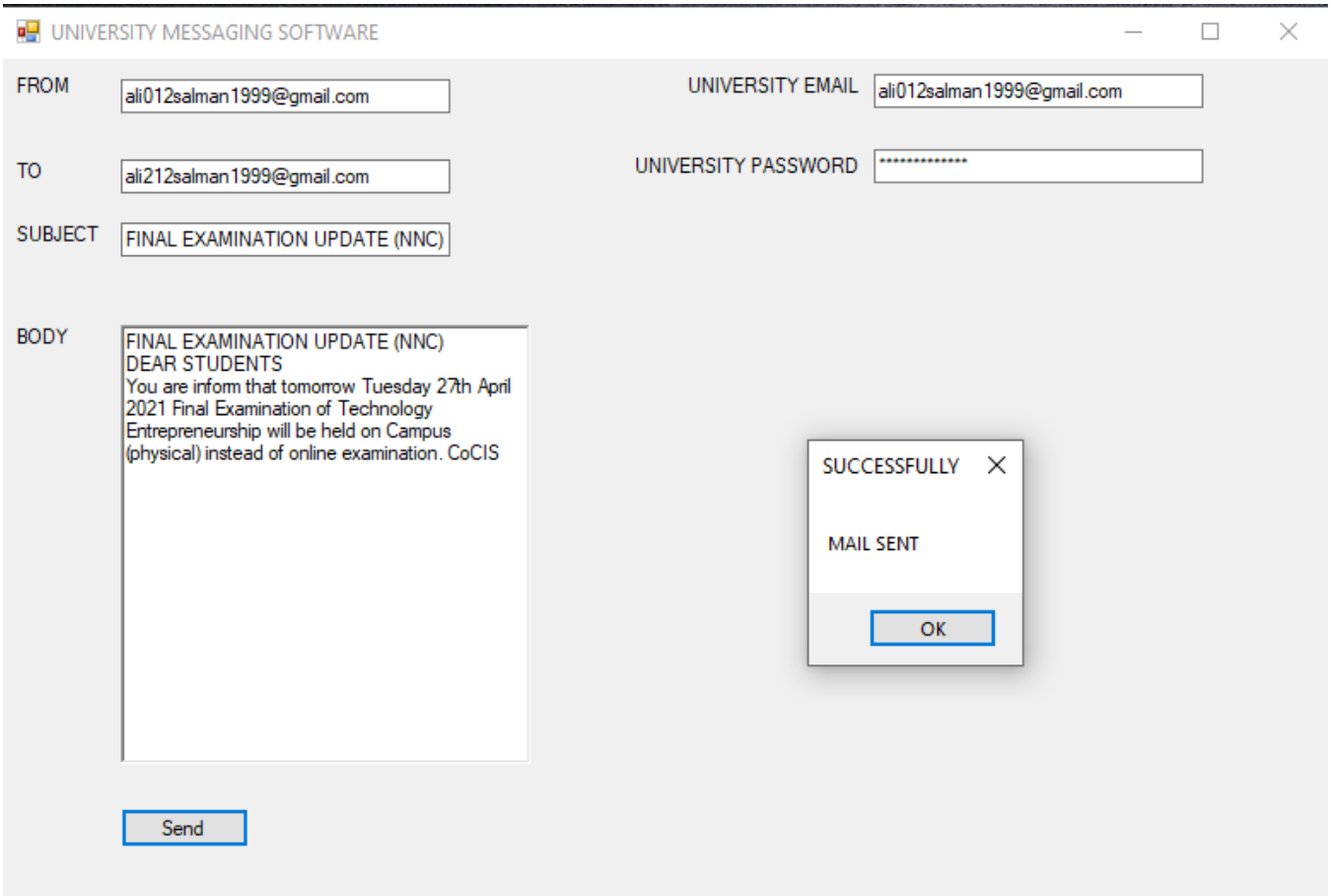
Answer for Question# 4b:

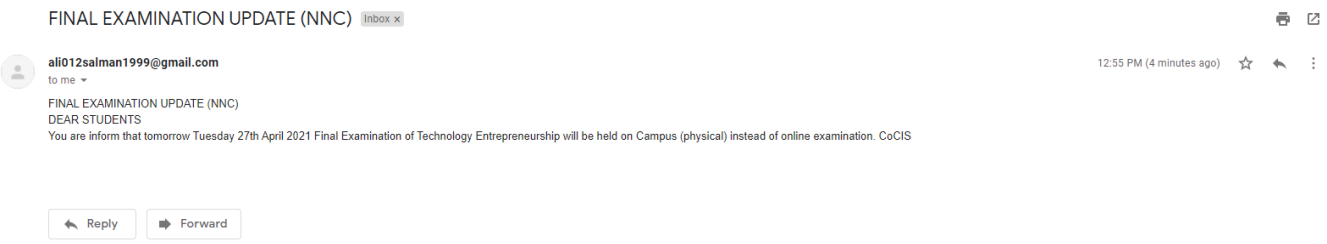
CODE:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Web;
using System.Net.Mail;

namespace emailSMTp
{
```

```
    public partial class Form1 : Form
    {

        public Form1()
        {

            InitializeComponent();
            textBox6.PasswordChar = '*';

        }

        private void button1_Click(object sender, EventArgs e)
        {

            MailMessage mail = new
MailMessage(textBox1.Text,textBox2.Text,textBox3.Text,richTextBox1.Text);
            SmtpClient client = new SmtpClient("smtp.gmail.com");
            client.Port = 587;
            client.UseDefaultCredentials = false;

            client.Credentials = new System.Net.NetworkCredential(textBox5.Text,textBox6.Text);
            client.EnableSsl = true;
            client.Send(mail);
            MessageBox.Show("MAIL SENT","SUCCESSFULLY",MessageBoxButtons.OK);

        }

    }

}
```

OUTPUT:





OVERVIEW OF POP3 AND IMAP LOGIC (Optional):

POP3:

```
Pop3Client pop = new Pop3Client();
pop.Host = "gmail.com";
pop.Username = "ali012salman1999@gmail.com";
pop.Password = "**************";
pop.Port = 969;
pop.EnableSsl = true;
pop.Connect();
MailMessage message = pop.GetMessage(1);
Console.WriteLine("------------------Messsge Headers ---------------");
Console.WriteLine("From : " + message.From.ToString());
Console.WriteLine("To : " + message.To.ToString());
```

```
Console.WriteLine("Date : " + message.Date.ToString(CultureInfo.InvariantCulture));
Console.WriteLine("Subject: " + message.Subject);
Console.WriteLine("------------------- Message Body -----------------");
Console.WriteLine(message.BodyText);
Console.WriteLine("-------------------  Message End ------------------");
message.Save(message.Subject + ".eml", MailMessageFormat.Eml);
Console.WriteLine("Message Saved.");
```

IMAP:

```
ImapClient imap = new ImapClient();

imap.Host = "gmail.com";

imap.Port = 143;

imap.Username = "ali012salman1999@gmail.com";
imap.Password = "*************";
imap.ConnectionProtocols = ConnectionProtocols.Ssl;
imap.Connect();
imap.Select("Inbox");
MailMessage message = imap.GetFullMessage(1);
Console.WriteLine("------------------Message Header ---------------");
Console.WriteLine("From : " + message.From.ToString());
Console.WriteLine("To : " + message.To.ToString());
Console.WriteLine("Date : " + message.Date.ToString(CultureInfo.InvariantCulture));
Console.WriteLine("Subject: " + message.Subject);
Console.WriteLine("------------------- Message Body -----------------");
Console.WriteLine(message.BodyText);
Console.WriteLine("------------------- Message End ------------------");
message.Save(message.Subject + ".eml", MailMessageFormat.Eml);
Console.WriteLine("Message Saved.");
```