RAJSHAHI UNIVERSITY OF ENGINEERING AND TECHNOLOGY

Course No: CSE 3206
Lab report: 01

Date of Submission: 21.01.2020

Submitted to:
Sarower Sattar
Professor,
Department of Computer
Science and Engineering
Rajshahi University of
Engineering and Technology

Submitted by:
Riyad Morshed Shoeb
Roll No: 1603013
Section: A
Department of Computer
Science and Engineering
Rajshahi University of
Engineering and Technology

Objectives:

1. Learn the fundamentals of network programming.
2. Understand the end-to-end communication.

Tools:

1. Java
2. Socket programming

Procedure:

Appendix:

1. Server-side Code:

```
import java.net.*;
import java.io.*;
import java.util.Scanner;

public class server{
        public static void main(String[] args){
                try{
            ServerSocket SerSock = new ServerSocket(1234);
            Socket Sock = SerSock.accept();
            System.out.println("Client Connected. Enter 'Exit/exit'
to disconnect.");

            InputStreamReader RecieveFromClient = new
InputStreamReader(Sock.getInputStream());
            BufferedReader ReadBuffer = new
BufferedReader(RecieveFromClient);
            String ClientInput;
            String SendToClient;
            Scanner UserInput = new Scanner(System.in);
            PrintWriter WriteToClient = new
PrintWriter(Sock.getOutputStream(), true);

            ClientInput = ReadBuffer.readLine();
            while(!ClientInput.equalsIgnoreCase("Exit")){
                    System.out.println("Client: " + ClientInput);

                    System.out.print("Server: ");
                    SendToClient = UserInput.nextLine();
                    WriteToClient.println(SendToClient);
                    if(SendToClient.equalsIgnoreCase("Exit"))
                        break;
                    ClientInput = ReadBuffer.readLine();
            }
            System.out.println("Connection Terminated");

            RecieveFromClient.close();
            WriteToClient.close();
            Sock.close();
        }
        catch(IOException e){
            System.out.println(e.toString());
        }
    }
}
```

2. Client-side Code:

```
import java.net.*;
import java.io.*;
import java.util.Scanner;


public class client{
     public static void main(String[] args){
          try{
               Socket CliSock = new Socket("localhost", 1234);
               System.out.println("Connected   to   Server.   Enter
'Exit/exit' to disconnect.");

               InputStreamReader    RecieveFromServer    =    new
InputStreamReader(CliSock.getInputStream());
               BufferedReader       ReadBuffer        =       new
BufferedReader(RecieveFromServer);
               String ServerInput;
               String SendToServer;
               Scanner UserInput = new Scanner(System.in);
               PrintWriter       WriteToServer        =       new
PrintWriter(CliSock.getOutputStream(), true);

               System.out.print("Client: ");
               SendToServer = UserInput.nextLine();
               WriteToServer.println(SendToServer);
               while(!SendToServer.equalsIgnoreCase("Exit")){
                    ServerInput = ReadBuffer.readLine();
                    System.out.println("Server: " + ServerInput);
                    if(ServerInput.equalsIgnoreCase("Exit"))
                         break;

                    System.out.print("Client: ");
                    SendToServer = UserInput.nextLine();
                    WriteToServer.println(SendToServer);
               }
               System.out.println("Connection Terminated");

               RecieveFromServer.close();
               WriteToServer.close();
               CliSock.close();
          }
          catch(IOException e){
               System.out.println(e.toString());
          }
     }
}
```

Conclusion: