**Name – Tanishq Thuse**

**Branch – CSAI**

**Div – B**

**Roll no. – 52**

**Subject – Computer Network**

Assignment 4 : Write the client server programs using TCP Berkeley socket primitives for wired /wireless network for following a. to say Hello to Each other b. File transfer

**a)To say hello to each other**

**Code :**

**Server.java**

import *java.io.BufferedReader*;

import *java.io.IOException*;

import *java.io.InputStreamReader*;

import *java.io.PrintWriter*;

import *java.net.ServerSocket*;

import *java.net.Socket*;

*public* *class* LineLinkServer {

*// keep port same on both sides*

*private* *static* *final* int PORT\_GATE = 5000;

*public* *static* void main(String[] argv) {

        ServerSocket gate = null;

        Socket peer = null;

        BufferedReader netIn = null;

        PrintWriter netOut = null;

        BufferedReader userIn = null;

        try {

            gate = new ServerSocket(PORT\_GATE);

            System.out.println("[server] gate open on " + PORT\_GATE + ", waiting...");

            peer = gate.accept();

            System.out.println("[server] peer online: " + peer.getRemoteSocketAddress());

            netIn = new BufferedReader(new InputStreamReader(peer.getInputStream()));

            netOut = new PrintWriter(peer.getOutputStream(), true);

            userIn = new BufferedReader(new InputStreamReader(System.in));

*final* Socket peerRef = peer;

*final* ServerSocket gateRef = gate;

*final* BufferedReader netInRef = netIn;

            Thread inboundPump = new Thread(() -> {

                try {

                    String line;

                    while ((line = netInRef.readLine()) != null) {

                        if ("bye".equalsIgnoreCase(line.trim())) {

                            System.out.println("[server] peer said bye. closing.");

                            safeClose(peerRef);

                            safeClose(gateRef);

                            System.exit(0);

                        }

                        System.out.println("[client] " + line);

                    }

                } catch (IOException ex) {

                    System.out.println("[server] link down.");

                } finally {

                    safeClose(peerRef);

                    safeClose(gateRef);

                }

            }, "inbound-pump");

            inboundPump.setDaemon(true);

            inboundPump.start();

            String outbound;

            while ((outbound = userIn.readLine()) != null) {

                netOut.println(outbound);

                if ("bye".equalsIgnoreCase(outbound.trim())) {

                    System.out.println("[server] session ended.");

                    break;

                }

            }

        } catch (IOException ioe) {

            System.out.println("[server] error: " + ioe.getMessage());

        } finally {

            safeClose(netIn);

            if (netOut != null) netOut.close();

            safeClose(userIn);

            safeClose(peer);

            safeClose(gate);

        }

    }

*private* *static* void safeClose(ServerSocket s) {

        if (s != null) {

            try { s.close(); } catch (IOException ignored) {}

        }

    }

*private* *static* void safeClose(Socket s) {

        if (s != null) {

            try { s.close(); } catch (IOException ignored) {}

        }

    }

*private* *static* void safeClose(BufferedReader r) {

        if (r != null) {

            try { r.close(); } catch (IOException ignored) {}

        }

    }

}

**Client.java**

import *java.io.BufferedReader*;

import *java.io.IOException*;

import *java.io.InputStreamReader*;

import *java.io.PrintWriter*;

import *java.net.Socket*;

*public* *class* LineLinkClient {

*private* *static* *final* String HOST\_ALIAS = "localhost";

*private* *static* *final* int PORT\_GATE = 5000;

*public* *static* void main(String[] args) {

        Socket wire = null;

        BufferedReader netIn = null;

        PrintWriter netOut = null;

        BufferedReader userIn = null;

        try {

            wire = new Socket(HOST\_ALIAS, PORT\_GATE);

            System.out.println("[client] connected to " + HOST\_ALIAS + ":" + PORT\_GATE);

            netIn = new BufferedReader(new InputStreamReader(wire.getInputStream()));

            netOut = new PrintWriter(wire.getOutputStream(), true);

            userIn = new BufferedReader(new InputStreamReader(System.in));

*final* Socket wireRef = wire;

*final* BufferedReader netInRef = netIn;

            Thread rxLoop = new Thread(() -> {

                try {

                    String incoming;

                    while ((incoming = netInRef.readLine()) != null) {

                        if ("bye".equalsIgnoreCase(incoming.trim())) {

                            System.out.println("[client] server said bye. closing.");

                            safeClose(wireRef);

                            System.exit(0);

                        }

                        System.out.println("[server] " + incoming);

                    }

                } catch (IOException e) {

                    System.out.println("[client] link closed.");

                } finally {

                    safeClose(wireRef);

                }

            }, "client-receiver");

            rxLoop.setDaemon(true);

            rxLoop.start();

            String outLine;

            while ((outLine = userIn.readLine()) != null) {

                netOut.println(outLine);

                if ("bye".equalsIgnoreCase(outLine.trim())) {

                    System.out.println("[client] session ended.");

                    break;

                }

            }

        } catch (IOException ioe) {

            System.out.println("[client] error: " + ioe.getMessage());

        } finally {

            safeClose(netIn);

            if (netOut != null) netOut.close();

            safeClose(userIn);

            safeClose(wire);

        }

    }

*private* *static* void safeClose(Socket s) {

        if (s != null) {

            try { s.close(); } catch (IOException ignored) {}

        }

    }

*private* *static* void safeClose(BufferedReader r) {

        if (r != null) {

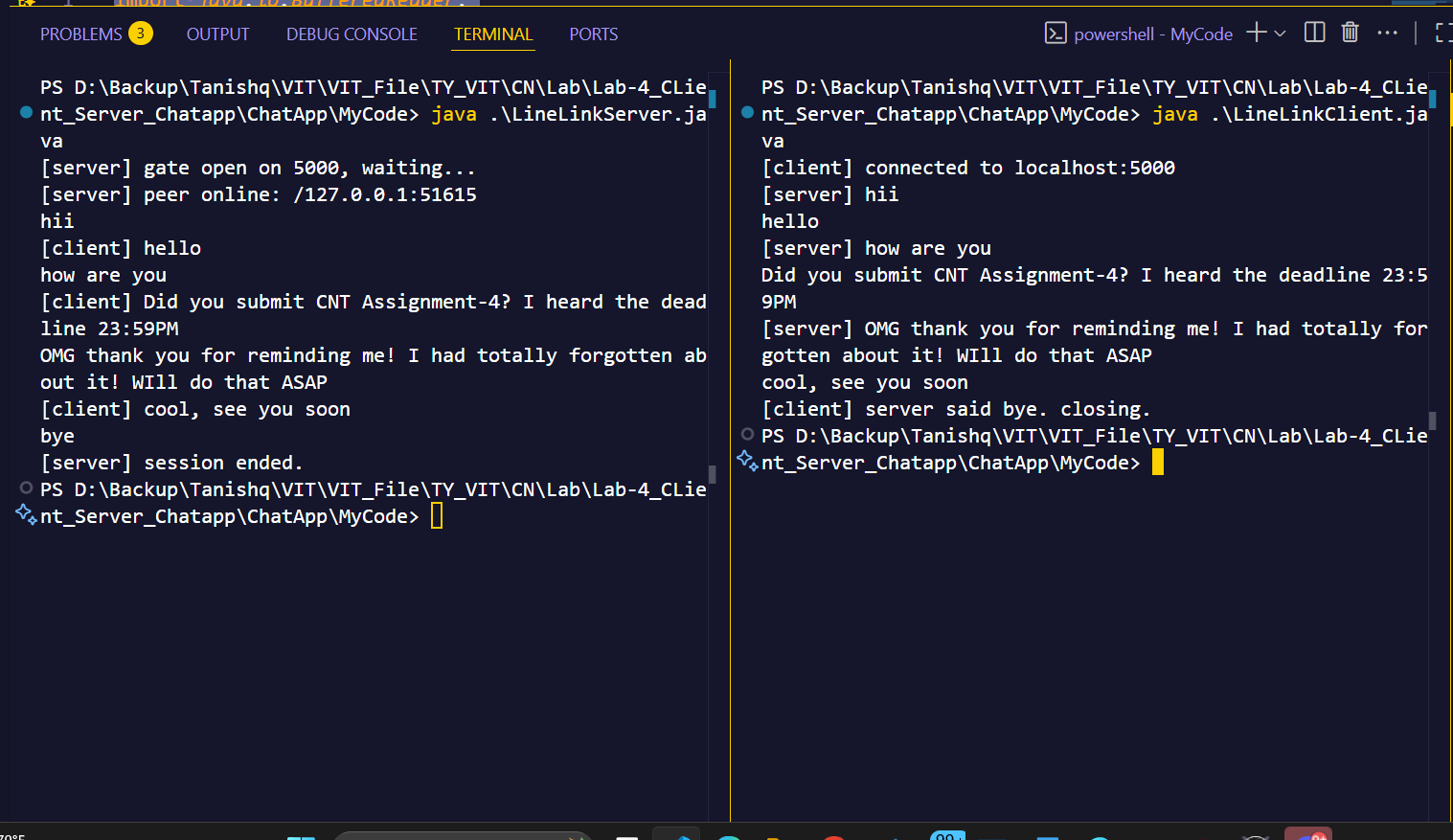
            try { r.close(); } catch (IOException ignored) {}

        }

    }

}

Output Screenshot (very imp) :



**b)File Transfer**

**Code :**

**Server.java**

import *java.io.BufferedInputStream*;

import *java.io.File*;

import *java.io.FileInputStream*;

import *java.io.IOException*;

import *java.io.OutputStream*;

import *java.net.ServerSocket*;

import *java.net.Socket*;

*public* *class* DocCourierServer {

*private* *static* *final* int DOORWAY = 5000;

*public* *static* void main(String[] args) {

        ServerSocket hub = null;

        Socket clientLink = null;

        BufferedInputStream fileFeed = null;

        OutputStream netPipe = null;

        try {

            hub = new ServerSocket(DOORWAY);

            System.out.println("[srv] Ready on port " + DOORWAY + ", waiting for pickup...");

            clientLink = hub.accept();

            System.out.println("[srv] Client connected: " + clientLink.getRemoteSocketAddress());

            File parcel = new File("D:/Backup/Tanishq/VIT/VIT\_File/TY\_VIT/CN/Lab/Lab-4\_CLient\_Server\_Chatapp/fileToShare.txt");

            if (!parcel.exists() || !parcel.isFile()) {

                System.out.println("[srv] Missing parcel: " + parcel.getAbsolutePath());

                return;

            }

            fileFeed = new BufferedInputStream(new FileInputStream(parcel));

            netPipe = clientLink.getOutputStream();

            byte[] packet = new byte[4096];

            int n;

            while ((n = fileFeed.read(packet)) != -1) {

                netPipe.write(packet, 0, n);

            }

            netPipe.flush();

            System.out.println("[srv] Parcel dispatched.");

        } catch (IOException e) {

            System.out.println("[srv] Problem: " + e.getMessage());

        } finally {

            closeQuietly(fileFeed);

            closeQuietly(netPipe);

            closeQuietly(clientLink);

            closeQuietly(hub);

        }

    }

*private* *static* void closeQuietly(AutoCloseable c) {

        if (c != null) {

            try { c.close(); } catch (Exception ignored) {}

        }

    }

}

**Client.java**

import *java.io.BufferedOutputStream*;

import *java.io.FileOutputStream*;

import *java.io.IOException*;

import *java.io.InputStream*;

import *java.net.Socket*;

*public* *class* DocCourierClient {

*private* *static* *final* String HOST\_ADDR = "localhost";

*private* *static* *final* int DOORWAY = 5000;

*public* *static* void main(String[] args) {

        Socket link = null;

        InputStream netFeed = null;

        BufferedOutputStream savePipe = null;

        try {

            link = new Socket(HOST\_ADDR, DOORWAY);

            System.out.println("[cli] Connected to " + HOST\_ADDR + ":" + DOORWAY);

            netFeed = link.getInputStream();

            savePipe = new BufferedOutputStream(new FileOutputStream("receivedfile.txt"));

            byte[] block = new byte[4096];

            int n;

            while ((n = netFeed.read(block)) != -1) {

                savePipe.write(block, 0, n);

            }

            savePipe.flush();

            System.out.println("[cli] Parcel received & stored.");

        } catch (IOException e) {

            System.out.println("[cli] Problem: " + e.getMessage());

        } finally {

            closeQuietly(savePipe);

            closeQuietly(netFeed);

            closeQuietly(link);

        }

    }

*private* *static* void closeQuietly(AutoCloseable c) {

        if (c != null) {

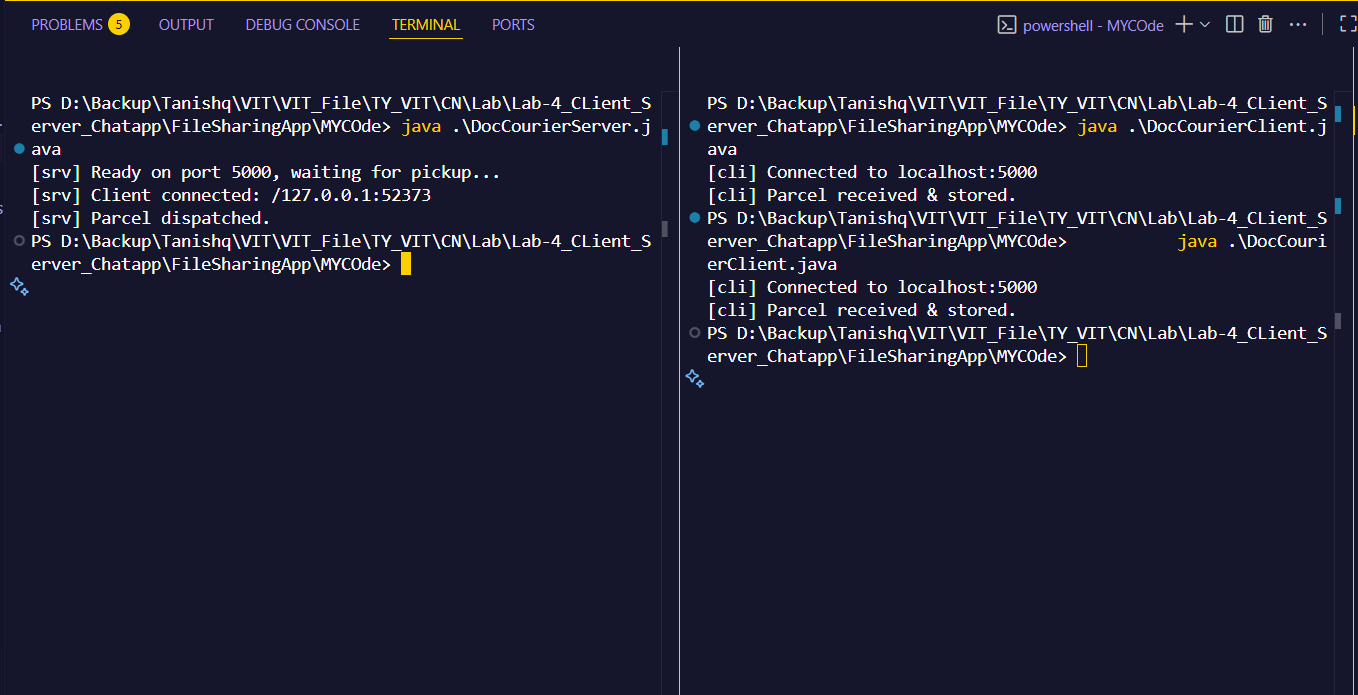
            try { c.close(); } catch (Exception ignored) {}

        }

    }

}

**Output :**

****