Code

Client.java

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

public class Client {
    private String hostname;
    private int port;
    private String userName;

    public Client(String hostname, int port) {
        this.hostname = hostname;
        this.port = port;
    }

    public void execute() {
        try {
            Socket socket = new Socket(hostname, port);

            System.out.println("Connected to the chat server");

            new ReadThread(socket, this).start();
            new WriteThread(socket, this).start();

        } catch (UnknownHostException ex) {
            System.out.println("Server not found: " + ex.getMessage());
        } catch (IOException ex) {
            System.out.println("I/O Error: " + ex.getMessage());
        }

    }

    void setUserName(String userName) {
        this.userName = userName;
    }

    String getUserName() {
        return this.userName;
    }


    public static void main(String[] args) {
        if (args.length < 2) return;

        String hostname = args[0];
        int port = Integer.parseInt(args[1]);

        Client client = new Client(hostname, port);
        client.execute();
```

```
    }
}

class ReadThread extends Thread {
    private BufferedReader reader;
    private Socket socket;
    private Client client;

    public ReadThread(Socket socket, Client client) {
        this.socket = socket;
        this.client = client;

        try {
            InputStream input = socket.getInputStream();
            reader = new BufferedReader(new InputStreamReader(input));
        } catch (IOException ex) {
            System.out.println("Error getting input stream: " + ex.getMessage());
            ex.printStackTrace();
        }
    }

    public void run() {
        while (true) {
            try {
                String response = reader.readLine();
                System.out.println("\n" + response);

                // prints the username after displaying the server's message
                if (client.getUserName() != null) {
                    System.out.print("[" + client.getUserName() + "]: ");
                }
            } catch (IOException ex) {
                System.out.println("Error reading from server: " + ex.getMessage());
                ex.printStackTrace();
                break;
            }
        }
    }
}

class WriteThread extends Thread {
    private PrintWriter writer;
    private Socket socket;
    private Client client;

    public WriteThread(Socket socket, Client client) {
        this.socket = socket;
        this.client = client;

        try {
```

```
            OutputStream output = socket.getOutputStream();
            writer = new PrintWriter(output, true);
        } catch (IOException ex) {
            System.out.println("Error getting output stream: " + ex.getMessage());
            ex.printStackTrace();
        }
    }

    public void run() {

        Console console = System.console();

        String userName = console.readLine("\nEnter your name: ");
        client.setUserName(userName);
        writer.println(userName);

        String text;

        do {
            text = console.readLine("[" + userName + "]: ");
            writer.println(text);

        } while (!text.equals("bye"));

        try {
            socket.close();
        } catch (IOException ex) {

            System.out.println("Error writing to server: " + ex.getMessage());
        }
    }
}
```

Server.java

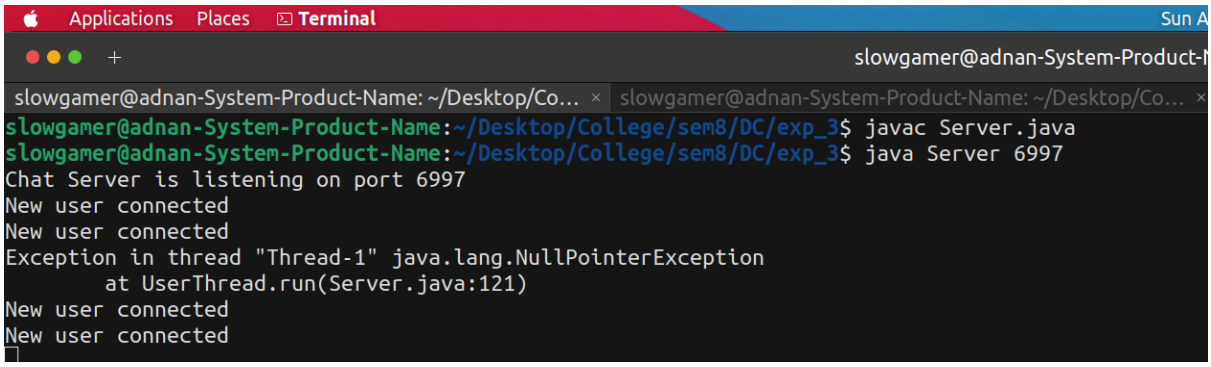
```
import java.io.*;
import java.net.*;
import java.util.*;


public class Server {
    private int port;
    private Set<String> userNames = new HashSet<>();
    private Set<UserThread> userThreads = new HashSet<>();

    public Server(int port) {
        this.port = port;
    }

    public void execute() {
        try (ServerSocket serverSocket = new ServerSocket(port)) {
```

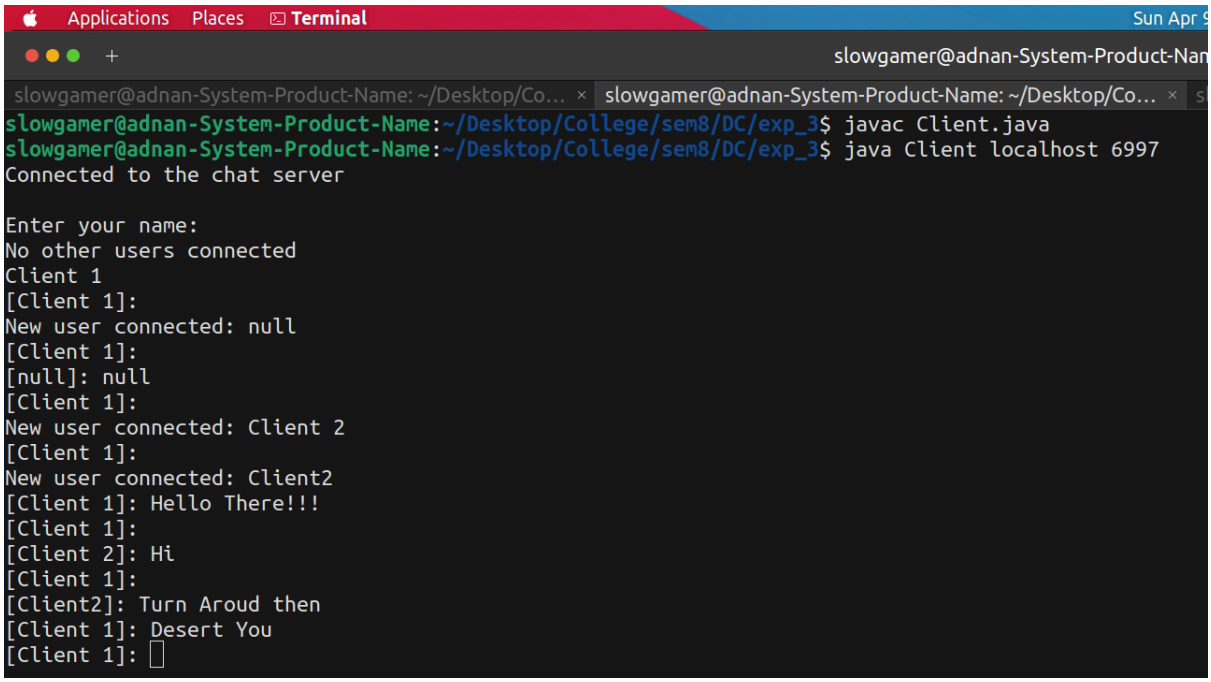
```
            System.out.println("Chat Server is listening on port " + port);

            while (true) {
                Socket socket = serverSocket.accept();
                System.out.println("New user connected");

                UserThread newUser = new UserThread(socket, this);
                userThreads.add(newUser);
                newUser.start();

            }

        } catch (IOException ex) {
            System.out.println("Error in the server: " + ex.getMessage());
            ex.printStackTrace();
        }
    }

    public static void main(String[] args) {
        if (args.length < 1) {
            System.out.println("Syntax: java Server <port-number>");
            System.exit(0);
        }

        int port = Integer.parseInt(args[0]);

        Server server = new Server(port);
        server.execute();
    }

    /**
     * Delivers a message from one user to others (broadcasting)
     */
    void broadcast(String message, UserThread excludeUser) {
        for (UserThread aUser : userThreads) {
            if (aUser != excludeUser) {
                aUser.sendMessage(message);
            }
        }
    }

    /**
     * Stores username of the newly connected client.
     */
    void addUserName(String userName) {
        userNames.add(userName);
    }

    /**
```

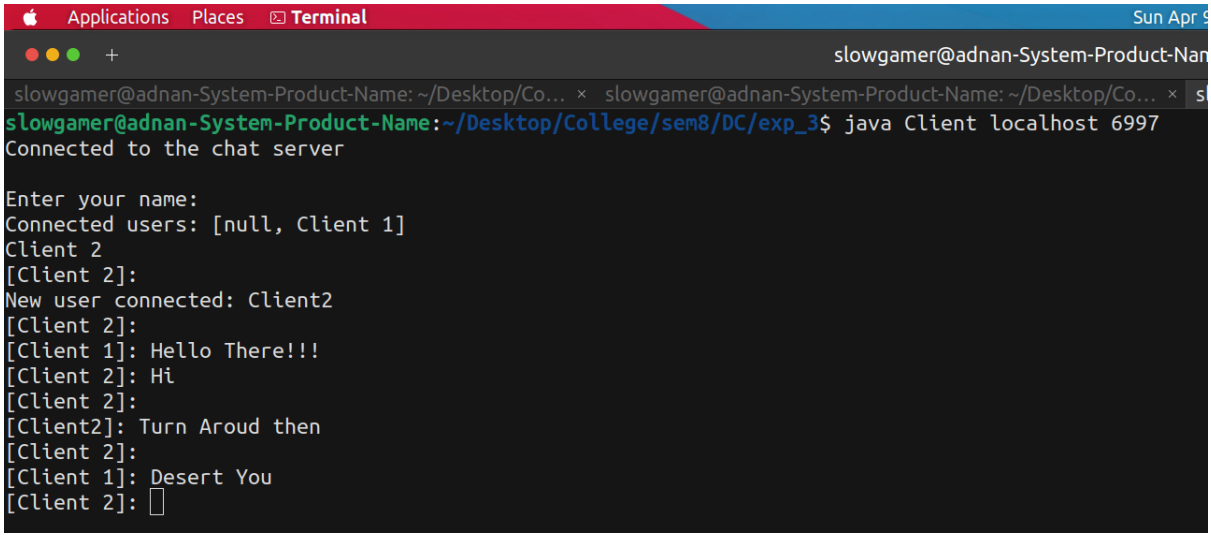
```
     * When a client is disconneted, removes the associated username and UserThread
     */
    void removeUser(String userName, UserThread aUser) {
        boolean removed = userNames.remove(userName);
        if (removed) {
            userThreads.remove(aUser);
            System.out.println("The user " + userName + " quitted");
        }
    }

    Set<String> getUserNames() {
        return this.userNames;
    }

    /**
     * Returns true if there are other users connected (not count the currently connected user)
     */
    boolean hasUsers() {
        return !this.userNames.isEmpty();
    }
}
class UserThread extends Thread {
    private Socket socket;
    private Server server;
    private PrintWriter writer;

    public UserThread(Socket socket, Server server) {
        this.socket = socket;
        this.server = server;
    }

    public void run() {
        try {
            InputStream input = socket.getInputStream();
            BufferedReader reader = new BufferedReader(new InputStreamReader(input));

            OutputStream output = socket.getOutputStream();
            writer = new PrintWriter(output, true);

            printUsers();

            String userName = reader.readLine();
            server.addUserName(userName);

            String serverMessage = "New user connected: " + userName;
            server.broadcast(serverMessage, this);

            String clientMessage;

            do {
```

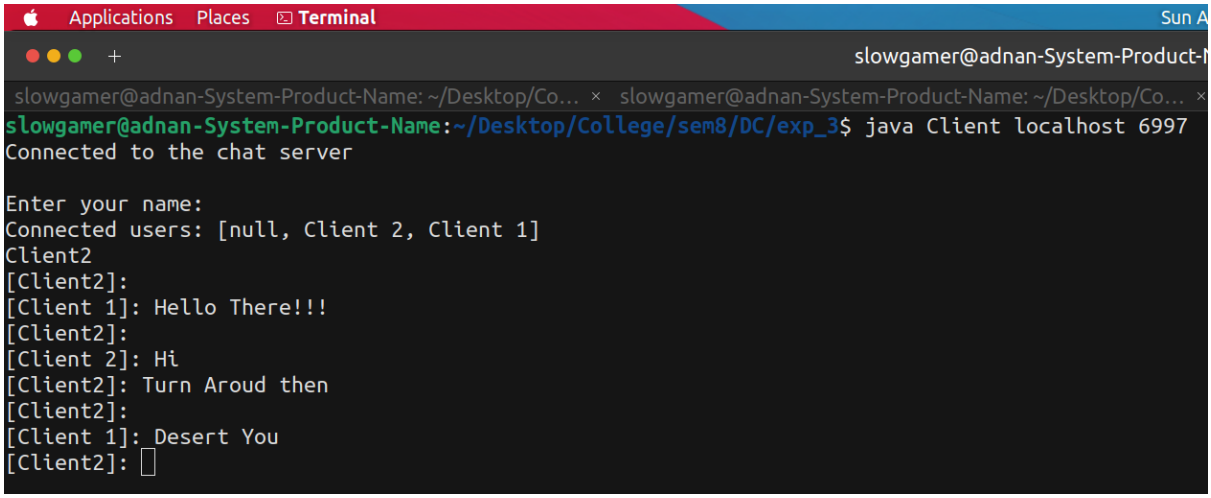
```
            clientMessage = reader.readLine();
            serverMessage = "[" + userName + "]: " + clientMessage;
            server.broadcast(serverMessage, this);

        } while (!clientMessage.equals("bye"));

        server.removeUser(userName, this);
        socket.close();

        serverMessage = userName + " has quitted.";
        server.broadcast(serverMessage, this);

      } catch (IOException ex) {
        System.out.println("Error in UserThread: " + ex.getMessage());
        ex.printStackTrace();
      }
    }

    /**
     * Sends a list of online users to the newly connected user.
     */
    void printUsers() {
      if (server.hasUsers()) {
        writer.println("Connected users: " + server.getUserNames());
      } else {
        writer.println("No other users connected");
      }
    }

    /**
     * Sends a message to the client.
     */
    void sendMessage(String message) {
      writer.println(message);
    }
}
```

Output


Applications Places Terminal
slowgamer@adnan-System-Product-Name:~/Desktop/College/sem8/DC/exp_3$ javac Server.java
slowgamer@adnan-System-Product-Name:~/Desktop/College/sem8/DC/exp_3$ java Server 6997
Chat Server is listening on port 6997
New user connected
New user connected
Exception in thread "Thread-1" java.lang.NullPointerException
at UserThread.run(Server.java:121)
New user connected
New user connected


Applications Places Terminal
slowgamer@adnan-System-Product-Name:~/Desktop/College/sem8/DC/exp_3$ javac Client.java
slowgamer@adnan-System-Product-Name:~/Desktop/College/sem8/DC/exp_3$ java Client localhost 6997
Connected to the chat server
Enter your name:
No other users connected
Client 1
[Client 1]:
New user connected: null
[Client 1]:
[null]: null
[Client 1]:
New user connected: Client 2
[Client 1]:
New user connected: Client2
[Client 1]: Hello There!!!
[Client 1]:
[Client 2]: Hi
[Client 1]:
[Client2]: Turn Aroud then
[Client 1]: Desert You
[Client 1]:


Applications Places Terminal
slowgamer@adnan-System-Product-Name:~/Desktop/College/sem8/DC/exp_3$ java Client localhost 6997
Connected to the chat server
Enter your name:
Connected users: [null, Client 1]
Client 2
[Client 2]:
New user connected: Client2
[Client 2]:
[Client 1]: Hello There!!!
[Client 2]: Hi
[Client 2]:
[Client2]: Turn Aroud then
[Client 2]:
[Client 1]: Desert You
[Client 2]:


Applications Places Terminal
Sun A
slowgamer@adnan-System-Product-N
slowgamer@adnan-System-Product-Name: ~/Desktop/Co...
slowgamer@adnan-System-Product-Name: ~/Desktop/Co...
slowgamer@adnan-System-Product-Name:~/Desktop/College/sem8/DC/exp_3$ java Client localhost 6997
Connected to the chat server
Enter your name:
Connected users: [null, Client 2, Client 1]
Client2
[Client2]:
[Client 1]: Hello There!!!
[Client2]:
[Client 2]: Hi
[Client2]: Turn Aroud then
[Client2]:
[Client 1]: Desert You
[Client2]: