

Programming Assignment Unit 7

Godfrey Ouma

University of the People

CS 1103: Programming 2

Vikas Thada

March 21, 2024

```
package cs;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.HashMap;
import java.util.Map;

//Server class to manage connections from multiple clients
public class ChatServer {
    //Map to store connected clients and their output streams
    private static Map<Integer, PrintWriter> clients = new HashMap<>();

    //Unique ID counter for clients
    private static int userId = 1;

    public static void main(String[] args) {

        //Create server socket

        try {
            ServerSocket ss = new ServerSocket(43257);
            System.out.print("Server has started and its waiting for the connection from
the client");

            while(true) {
                //Accept connection from client
                Socket clientSocket = ss.accept();
                System.out.println("Client connected: " +clientSocket);

                //Create a new thread to handle client communication
                new ClientHandler(clientSocket).start();
            }
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
            System.err.println("Error in the server: " +e.getMessage());
        }
    }

    //Inner class to handle communication with individual clients
    private static class ClientHandler extends Thread{
        private Socket clientSocket;
        private int id;
        private PrintWriter out;
        private BufferedReader in;

        public ClientHandler(Socket socket) {
            this.clientSocket = socket;
            this.id = userId++;
        }

        public void run() {
            try {
                out = new PrintWriter(clientSocket.getOutputStream(), true);
```

```

        in = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));

        //Add client to the map of connected clients
        clients.put(id, out);

        //Broadcast message to all clients that a new user has joined
        broadcast("User " + id + " has joined the chat");

        //Read message from the client and broadcast to all other clients
        String message;
        while((message = in.readLine()) != null) {
            broadcast("User " + id + ": " + message);
        }
    } catch (IOException e) {
        System.err.println("Error handling client: " + e.getMessage());
        e.printStackTrace();
    } finally {
        //Remove client from the map and close streams
        clients.remove(id);
        try {
            clientSocket.close();
        } catch (IOException e) {
            System.err.println("Error closing socket: " + e.getMessage());
        }
        //Broadcast message to all clients that a user has left
        broadcast("User " + id + " has left the chat.");
    }

}

//Method to broadcast a message to all connected clients
private void broadcast(String message) {
    for (PrintWriter client : clients.values()) {
        client.println(message);
    }
}
}

```

Console

```
Server has started and its waiting for the connection from the client
```