

## 1. WELL-COMMENTED JAVA SOURCE CODE

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

// ChatServer class
class ChatServer {
    private static final int PORT = 12345;
    private static int clientIdCounter = 1;
    private static final Map<Integer, ClientHandler> clients = new
ConcurrentHashMap<>();

    public static void main(String[] args) {
        System.out.println("Chat server started...");
        try (ServerSocket serverSocket = new ServerSocket(PORT)) {
            while (true) {
                Socket clientSocket = serverSocket.accept();
                int clientId = clientIdCounter++;
                ClientHandler handler = new ClientHandler(clientSocket, clientId);
                clients.put(clientId, handler);
                new Thread(handler).start();
                broadcast("User " + clientId + " has joined the chat.");
            }
        } catch (IOException e) {
            System.out.println("Server error: " + e.getMessage());
        }
    }

    // Broadcast message to all connected clients
    public static void broadcast(String message) {
        for (ClientHandler client : clients.values()) {
            client.sendMessage(message);
        }
    }
}
```

```

}

// Remove client when they disconnect
public static void removeClient(int clientId) {
    clients.remove(clientId);
    broadcast("User " + clientId + " has left the chat.");
}

// Inner class for handling each client
static class ClientHandler implements Runnable {
    private final Socket socket;
    private final int clientId;
    private PrintWriter out;

    public ClientHandler(Socket socket, int clientId) {
        this.socket = socket;
        this.clientId = clientId;
    }

    @Override
    public void run() {
        try (
            BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
        ) {
            out = new PrintWriter(socket.getOutputStream(), true);
            out.println("Welcome! Your user ID is " + clientId);

            String message;
            while ((message = in.readLine()) != null) {
                ChatServer.broadcast("User " + clientId + ": " + message);
            }
        } catch (IOException e) {
            System.out.println("User " + clientId + " disconnected.");
        }
    }
}

```

```

    } finally {
        try {
            socket.close();
        } catch (IOException ignored) {}
        ChatServer.removeClient(clientId);
    }
}

public void sendMessage(String message) {
    if (out != null) {
        out.println(message);
    }
}
}
}

// ChatClient class
class ChatClient {
    private static final String SERVER_ADDRESS = "localhost";
    private static final int PORT = 12345;

    public static void main(String[] args) {
        try (
            Socket socket = new Socket(SERVER_ADDRESS, PORT);
            BufferedReader serverIn = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
            PrintWriter serverOut = new PrintWriter(socket.getOutputStream(), true);
            BufferedReader userIn = new BufferedReader(new
InputStreamReader(System.in));
        ) {
            // Thread to read messages from server
            new Thread() -> {
                String response;
                try {

```

```

        while ((response = serverIn.readLine()) != null) {
            System.out.println(response);
        }
    } catch (IOException e) {
        System.out.println("Disconnected from server.");
    }
}).start();

// Main thread reads user input and sends to server
String input;
while ((input = userIn.readLine()) != null) {
    serverOut.println(input);
}
} catch (IOException e) {
    System.out.println("Client error: " + e.getMessage());
}
}
}

```

## 2. SCREENSHOT OF TEXT-BASED USER INTERFACE

→ Screenshot of running server (ChatServer Class):



The screenshot shows the Eclipse IDE interface with a terminal window. The terminal displays the output of a Java application. The first line is a command prompt: `PS C:\Users\Fitzgerald Challar> & 'C:\Users\Fitzgerald Challar\AppData\Local\Programs\Eclipse Adoptium\jdk-21.0.6.7-hotspot\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Fitzgerald Challar\AppData\Local\Temp\vscodesws_7ace2\jdt_ws\jdt.ls-java-project\bin' 'ChatServer'`. The subsequent lines show the program's output: `Welcome! Your user ID is 1`, `This is User 1 and it's working fine`, and `User 1: This is User 1 and it's working fine`. The IDE's status bar at the bottom indicates `Ln 4, Col 31`, `Spaces: 4`, `UTF-8`, `Java`, and an `Activate Windows` watermark.

```
PS C:\Users\Fitzgerald Challar> & 'C:\Users\Fitzgerald Challar\AppData\Local\Programs\Eclipse Adoptium\jdk-21.0.6.7-hotspot\bin\java.exe' '-XX
:~ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Fitzgerald Challar\AppData\Local\Temp\vscodesws_7ace2\jdt_ws\jdt.ls-java-project\bin' 'Ch
atServer'
Welcome! Your user ID is 1
This is User 1 and it's working fine
User 1: This is User 1 and it's working fine
```

→ Screenshot of Client 2 (User 2 from ChatClient class)

This screenshot is similar to the first one, showing the Eclipse IDE terminal. The command prompt is the same: `PS C:\Users\Fitzgerald Challar> & 'c:\Users\Fitzgerald Challar\AppData\Local\Programs\Eclipse Adoptium\jdk-21.0.6.7-hotspot\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Fitzgerald Challar\AppData\Local\Temp\vscodesws_7ace2\jdt_ws\jdt.ls-java-project\bin' 'ChatClient'`. The output now shows: `Welcome! Your user ID is 2`, `this is User 2 and it's working fine`, and `User 2: this is User 2 and it's working fine`. The status bar at the bottom shows `Ln 4, Col 31`, `Spaces: 4`, `UTF-8`, `CRLF`, `Java`, and the `Activate Windows` watermark.

```
PS C:\Users\Fitzgerald Challar> & 'c:\Users\Fitzgerald Challar\AppData\Local\Programs\Eclipse Adoptium\jdk-21.0.6.7-hotspot\bin\java.exe' '-XX
:~ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Fitzgerald Challar\AppData\Local\Temp\vscodesws_7ace2\jdt_ws\jdt.ls-java-project\bin' 'ch
atClient'
Welcome! Your user ID is 2
this is User 2 and it's working fine
User 2: this is User 2 and it's working fine
```

### **3. README FILE**

#### **README Instructions**

#### **How to Compile and Run**

##### **1. Compile**

**javac ChatApplication.java**

##### **2. Run the Server**

**java ChatServer**

##### **3. Run Clients in Separate Terminals**

**java ChatClient**

#### **Usage:**

- **Each client gets a unique user ID.**
- **Type messages in the terminal and press Enter to send.**
- **Messages are broadcasted to all connected clients.**