**TASK-3**

**MULTITHREADED CHAT APPLICATION**

**STEP 1:** Create Project Folder named ChatAPP

**STEP 2:** Create Java files

Open Notepad and store the java code in it and name the file as given below

i)Write the server code and store it in a file named ‘ChatServer.java’

import java.io.\*;

import java.net.\*;

import java.util.\*;

public class ChatServer {

private static final int PORT = 12345;

// Store all client output streams to broadcast messages

private static Set<PrintWriter> clientWriters = Collections.synchronizedSet(new HashSet<>());

public static void main(String[] args) {

System.out.println("Chat server started...");

try (ServerSocket serverSocket = new ServerSocket(PORT)) {

while (true) {

Socket clientSocket = serverSocket.accept();

System.out.println("New client connected");

ClientHandler handler = new ClientHandler(clientSocket);

new Thread(handler).start();

}

} catch (IOException e) {

System.err.println("Error in server: " + e.getMessage());

}

}

private static class ClientHandler implements Runnable {

private Socket socket;

private PrintWriter out;

private BufferedReader in;

public ClientHandler(Socket socket) {

this.socket = socket;

}

public void run() {

try {

// Setup input and output streams

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

out = new PrintWriter(socket.getOutputStream(), true);

// Add this client's output stream to the set

clientWriters.add(out);

String message;

while ((message = in.readLine()) != null) {

System.out.println("Received: " + message);

broadcastMessage(message);

}

} catch (IOException e) {

System.err.println("Client connection error: " + e.getMessage());

} finally {

// Client disconnected, clean up

if (out != null) {

clientWriters.remove(out);

}

try {

socket.close();

} catch (IOException e) { /\* ignore \*/ }

}

}

private void broadcastMessage(String message) {

synchronized (clientWriters) {

for (PrintWriter writer : clientWriters) {

writer.println(message);

}

}

}

}

}

ii) Write the client code and store it in a file named ‘ChatClient.java’

import java.io.\*;

import java.net.\*;

public class ChatClient {

private static final String SERVER\_IP = "localhost"; // Change if server on different machine

private static final int SERVER\_PORT = 12345;

public static void main(String[] args) {

try (

Socket socket = new Socket(SERVER\_IP, SERVER\_PORT);

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

BufferedReader serverReader = new BufferedReader(new InputStreamReader(socket.getInputStream()));

PrintWriter serverWriter = new PrintWriter(socket.getOutputStream(), true);

) {

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

// Thread to read messages from server

Thread readThread = new Thread(() -> {

try {

String serverMessage;

while ((serverMessage = serverReader.readLine()) != null) {

System.out.println("Server: " + serverMessage);

}

} catch (IOException e) {

System.out.println("Connection closed.");

}

});

readThread.start();

// Main thread reads from console and sends to server

String userInput;

while ((userInput = consoleReader.readLine()) != null) {

serverWriter.println(userInput);

}

} catch (IOException e) {

System.err.println("Client error: " + e.getMessage());

}

}

}

**STEP 3:** Compile the Code

i. Open Command Prompt

cd C:\Users\YourName\Documents\ChatApp

ii. Compile both Java files:

javac ChatServer.java ChatClient.java

This will create `.class` files if there are no errors.

**STEP 4:** Run the Chat App

i. Start the Server

In the same Command Prompt:

java ChatServer

It should say:

Server started on port 12345

Leave this window open.

ii. Open a new Command Prompt for each client

In each new terminal:

cd C:\Users\YourName\Documents\ChatApp

java ChatClient

* Enter your name when prompted
* Type messages
* Use `/quit` to leave