# **Project 2: Peer-to-Peer Chatroom Application**

Team Members:

Suyog Swami - 1001119101

Vishal Dholakia - 1001106182

Yash Shah - 1001111714

**System Architecture: Peer-to-Peer Architecture**

There is single server which is used to store the authentication information of every client that logs into the system. It also has the information of the first client machine that is connected to and it passes this address to every new machine that is connected to this network.

**Software Architecture: Ring Topology**

Peer-to-Peer architecture is implemented using the ring topology. Here each new client connects with the two other client. And the message is passed between all the clients and this message is not sent to centralized server which is used to do authentication.

**How the code is implemented?**

The code is implemented keeping in mind the ring architecture. As the execution starts, first we need to start the execution of the server. This server is used to authenticate the client and it gives the IP address of the first connected client. Now, as new clients are connected they send their authentication request to the main server. When new client evokes it sends a socket connection request to the main server, which in the response of this request sends a reply that the connection is open and also sends the IP address of the last connected client to the new client. Now this newly connected client replaces the previous IP address on the main server with the IP address of the newly connected client. This is done so that every new incoming client will get address of the lastly connected client in the link.

Now each client will have two IP addresses, one is the IP address of the client who joined in the network previously to that client which will become the local server to that client and the other will be the main server which is used for authentication. The new client sends a ‘join’ message in the network to other client which is ahead of it in the ring, which passes this message similarly till the first client who joined. Now the first client finds that the IP address currently it is connected to not the last client in the ring, and that a new client has joined the network so it updates the IP address in its record and in this way the ring is completed when a new user connects to the network.

When the client is authenticated and the ring is formed the client is redirected to page where the chatrooms are displayed. Here client can join any chatrooms or create one. Now when a new client joins the chatrooms, the local server of that client sends a message to the local server of the client that is ahead of present client in the ring. The local server of each client checks whether the current client is in the chatroom or not. If the client is in the chatroom the message is shown and if the client is not in chatroom then the local server just forwards the message till when the message is received back by the same client.

When a user sign off/leaves the network, the client sends a ‘leave’ message along with the IP address and that message is sent forward to all the other nodes until client having the leaving client as its superior node. The message consist of the sender and the receiver IP address, so when this message is received by the client who’s receiver IP address is equal to sender IP address, that receiver changes its receiver IP address to the receiver IP address of the ‘leave’ message. In this way the ring is maintained when a client leaves the network.

**Prerequisites:**

* NetBeans IDE 8.0.1 OR any IDE supporting java
* JDK 1.8
* JDBC connector (version 5.1.33) is used for the database connectivity.
* MySQL 5.6

**How to Compile and Run the program:**

1. Copy the server file and database same on a machine.
2. Import the client file on any other machine that is on the same network. (currently this code works on wireless network on a simple router or a hotspot network, it does not work on the college network or myresnet).
3. Change the IP address of the server in the client java file.
4. Run the server
5. Run the client
6. And voila the program starts execution.

**Limitations**

We need to start the first client and it should always all be connect first to the main server and then and only then other clients should connect.

**References:**

<http://www.aisectuniversity.ac.in/Anusandhan3/16.pdf>

<http://www2.cs.uregina.ca/~hamilton/courses/330/notes/distributed/distributed.html>

<http://www.youtube.com/watch?v=qWYn1omeqqs>

<http://www.diva-portal.org/smash/get/diva2:524732/FULLTEXT01.pdf>

<http://www.codeproject.com/Articles/1297/Peer-to-Peer-Communicator-and-File-Transfer>

<http://listoffreeware.com/list-of-best-free-p2p-chat-software/>