CSE 5306 Distributed Systems

Lab#1 Document

By

Likitha Seeram

1001363714

likitha.seeram@mavs.uta.edu

Contents

[1. System Requirements 3](#_Toc476858037)

[2. Steps to run the application 3](#_Toc476858038)

[2.1 Scenario 1 3](#_Toc476858039)

[2.1.1 Scenario 1 Assumptions 8](#_Toc476858040)

[2.2 Scenario 2 9](#_Toc476858041)

[2.2.1 Scenario 2 Limitations 9](#_Toc476858042)

[3. Extra Functionality 9](#_Toc476858043)

[4. References 9](#_Toc476858044)

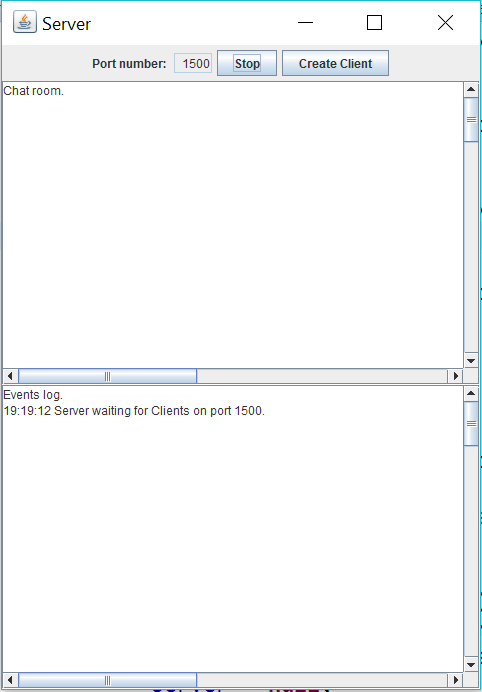
# System Requirements

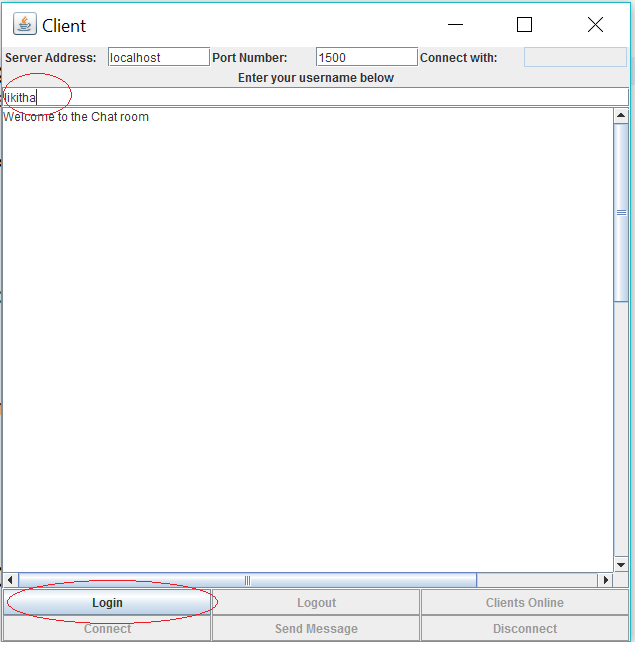
Following are the system requirements of the project:

* Language: Java
* IDE: Eclipse
* Environment: Windows

# Steps to run the application

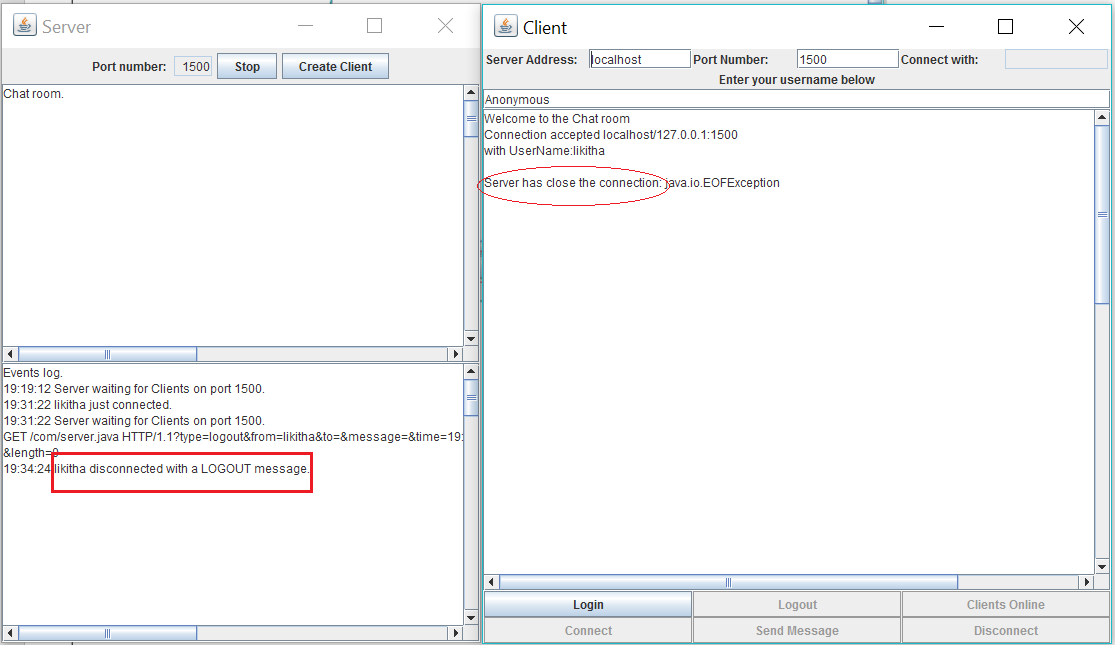
## Scenario 1

1. Start server by running the file **‘ServerGUI.java’**. After the server appears, start the server by providing a port number. I have given the default port as 1500 as port 80 is not available for me.
2. Start client by running **‘ClientGUI.java’**. After the window appears, you can see 3 text fields enabled. Provide the server address and port number (they are set to their default values initially). Enter the user name of the client, which is set to Anonymous by default, and select login option provided below. This lets the client to connect with the server. Giving a port number on which a server is not available gives an exception message. After logging, a connection accepted message is displayed.

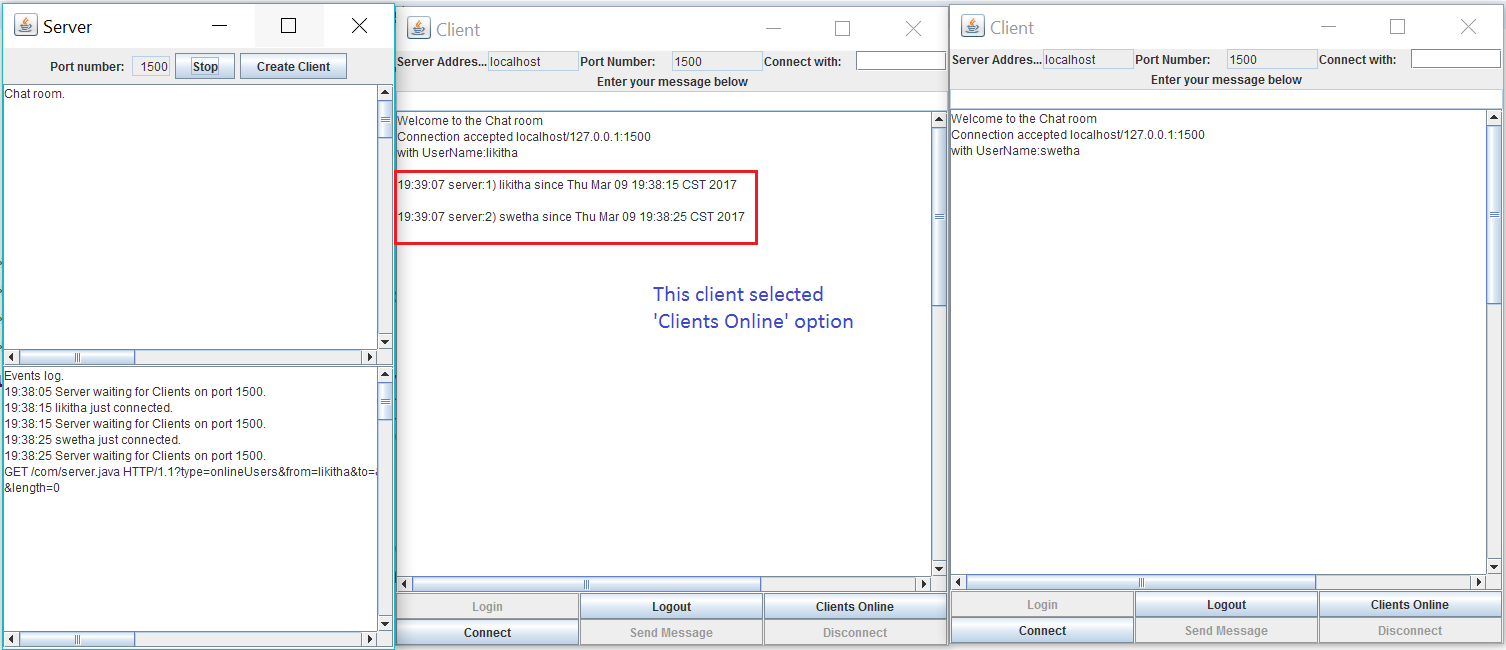


1. When a client is logged in, it can perform 3 actions.

* Logout



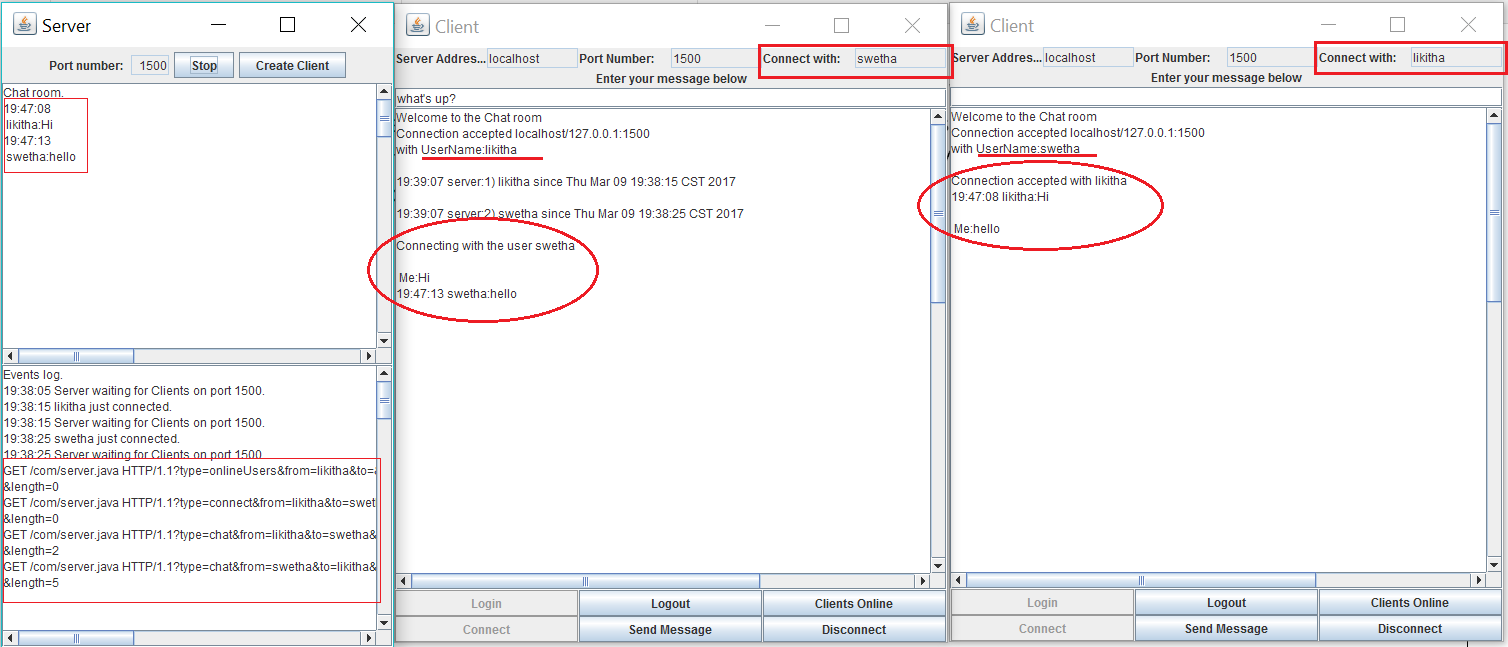
* Check users who are online by selecting ‘Clients Online option’. Create other clients by running the **‘ClientGUI.java’** again.



* Connect with available user by specifying the username in ‘Connect with’ text field. After clicking on connect, a connection is established by server and the other client gets notifies. Now the text field to enter message is enabled and connected users can send messages by clicking ‘Send Message’ button.

Along with this, clients will still have options to logout, check online clients and disconnect from conversation.

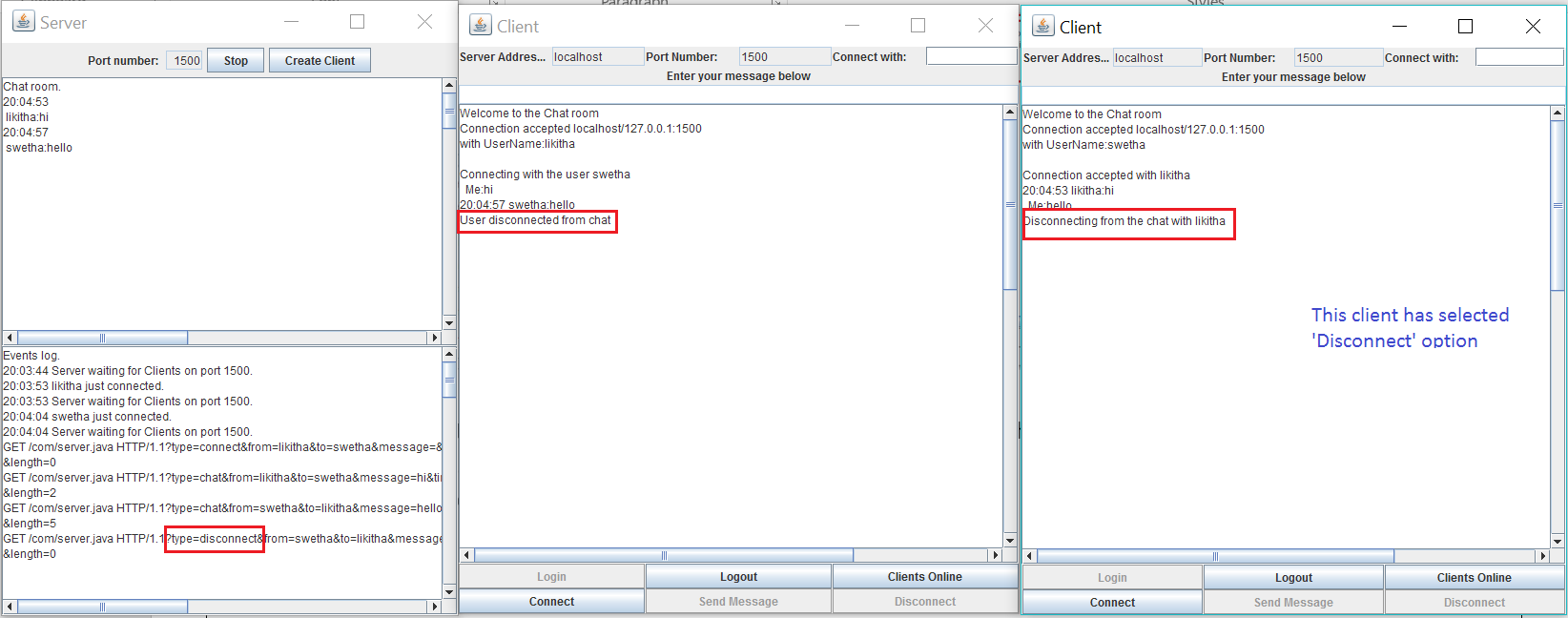
Messages are also logged at server side, along with events.

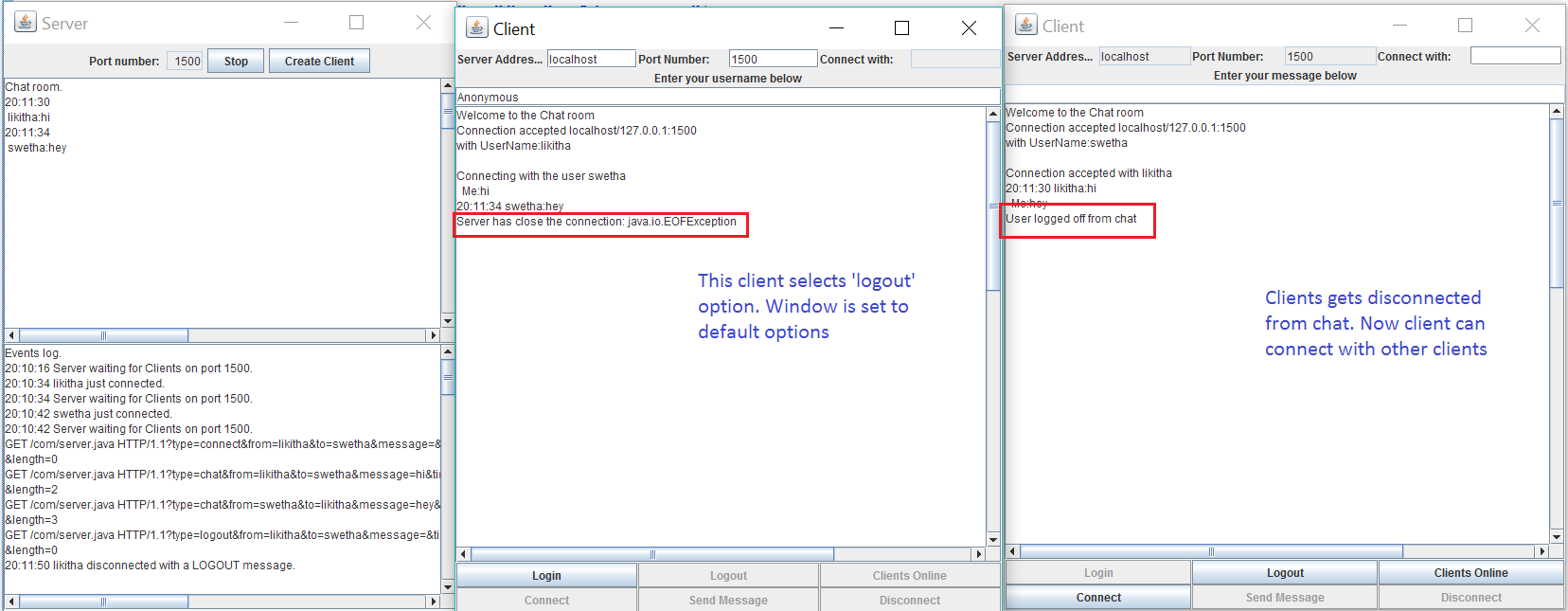


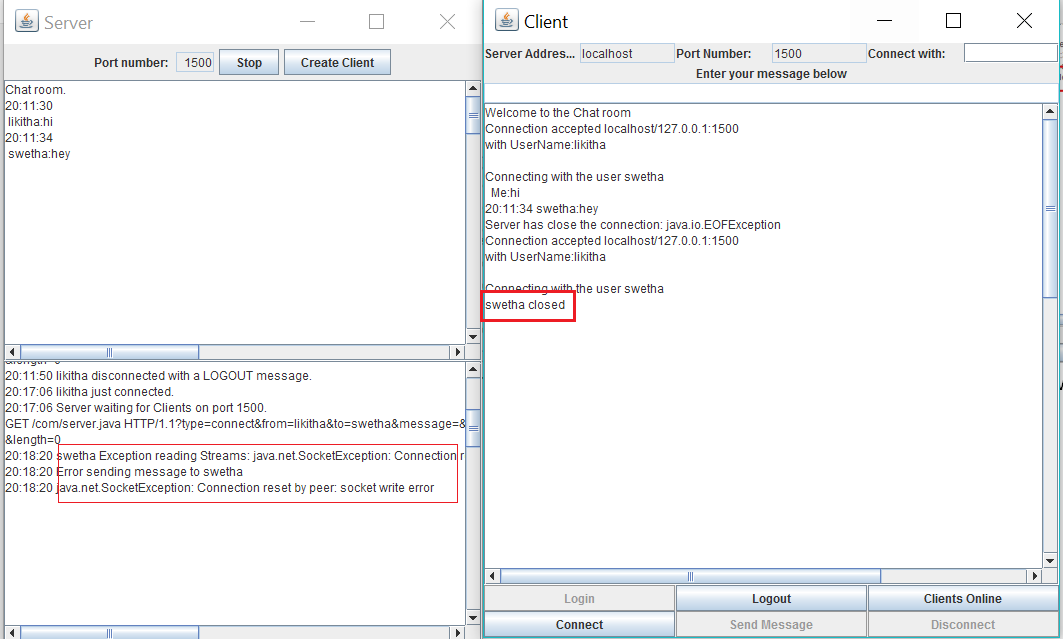
1. When 2 clients are in connection, there are 3 possible ways in which a client gets disconnected from the conversation. The other client in connection will be notified accordingly.

While testing, you can login/connect gain with the same client to test another way.

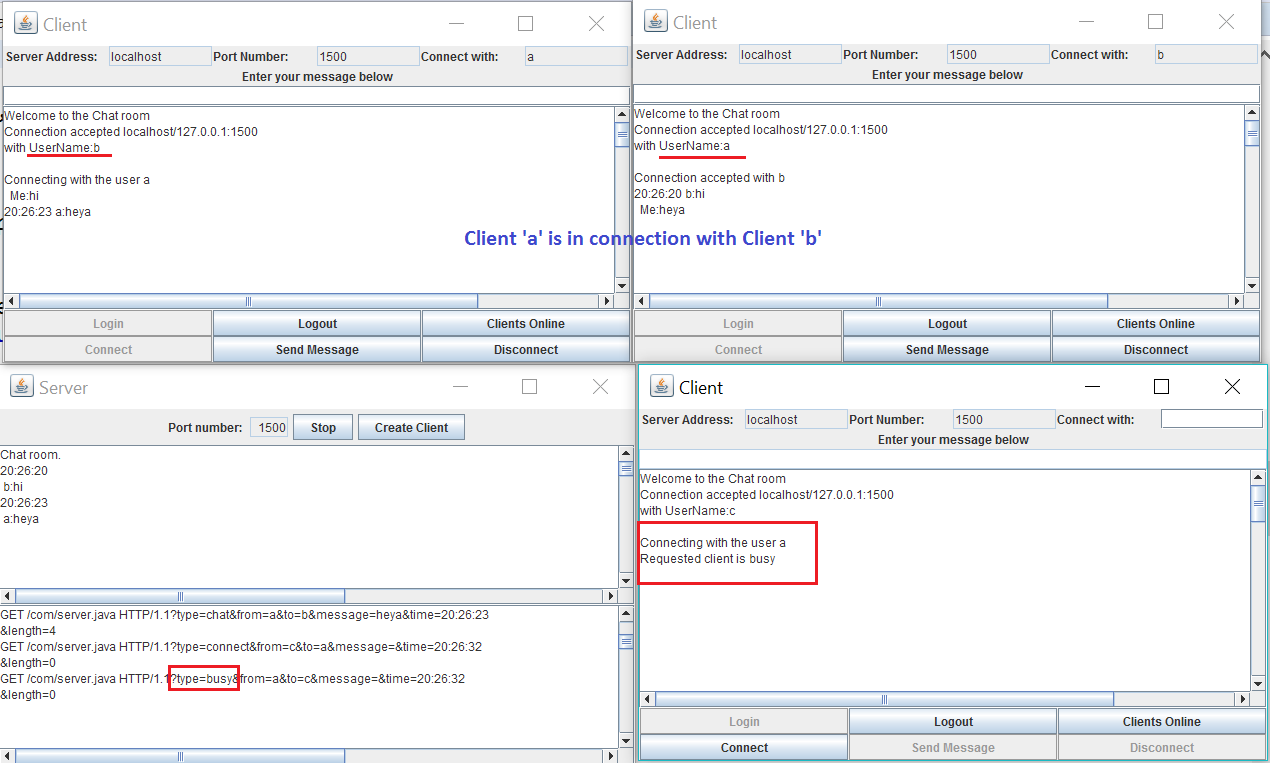
* + By clicking ‘Disconnect’ button



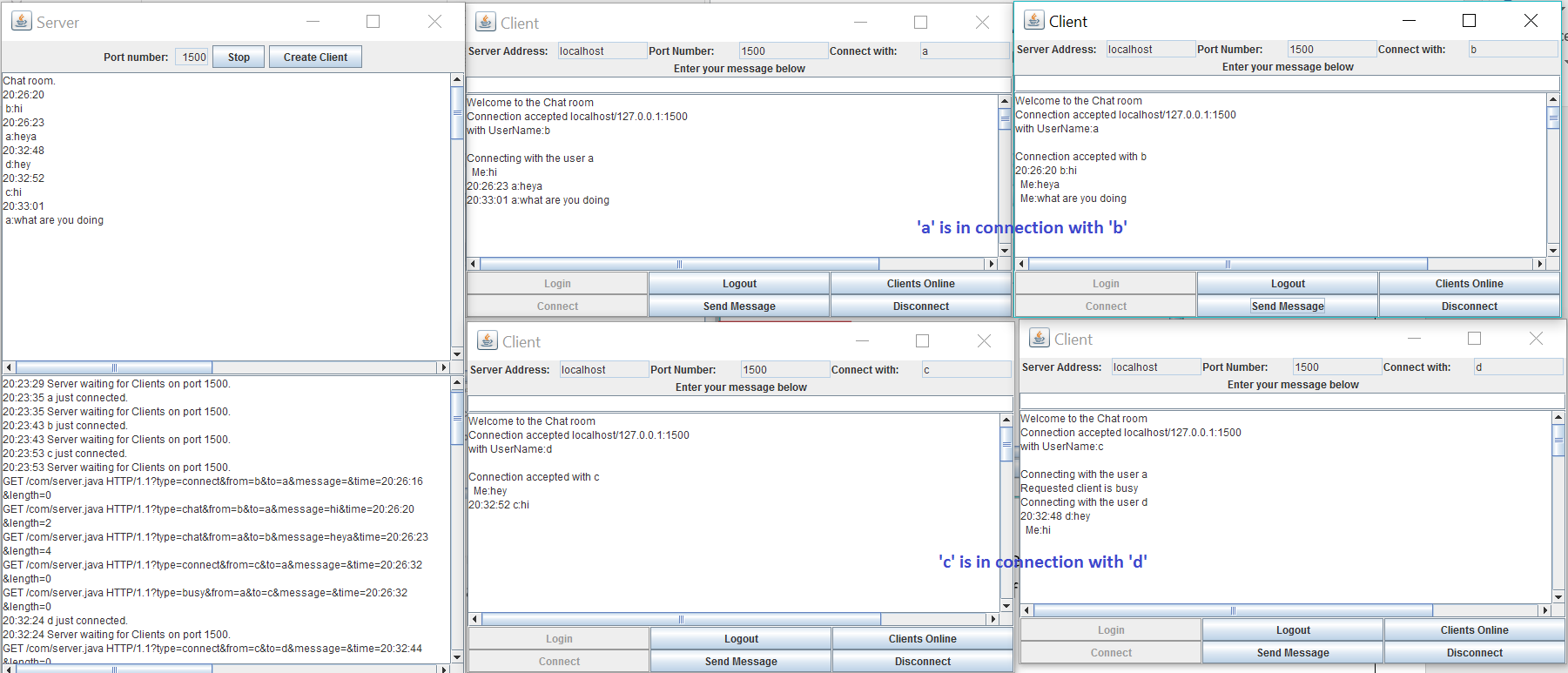
* + When one among the clients logs off
  + When any client closes by selecting **X** button of the window



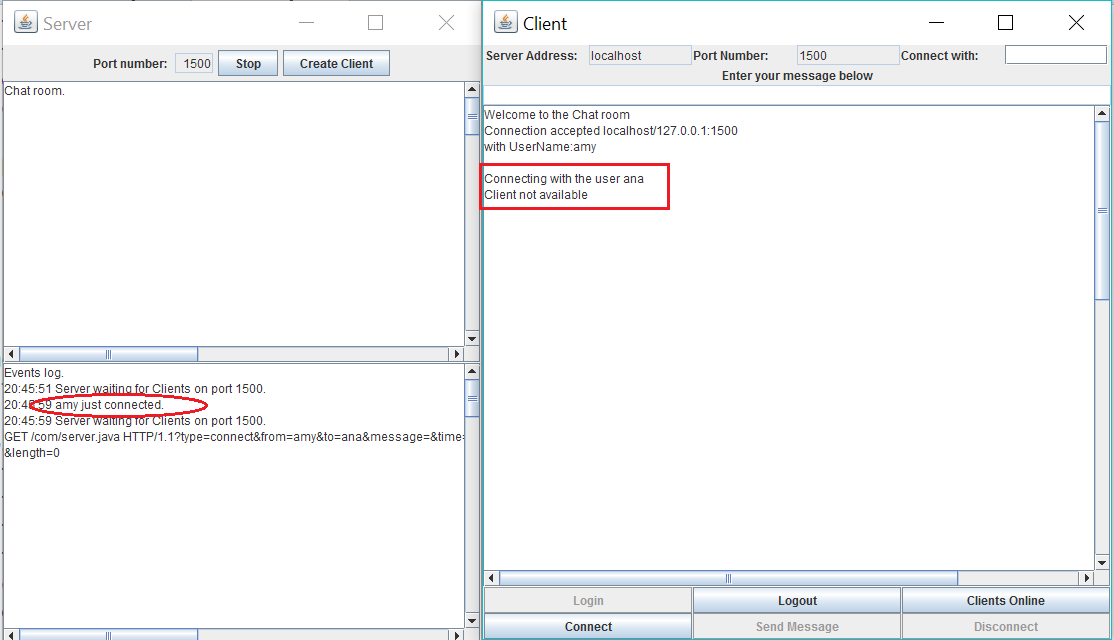
1. When a client requests connection for a client which is already in another connection, then the request is denied. And appropriate message is sent to the requesting client.



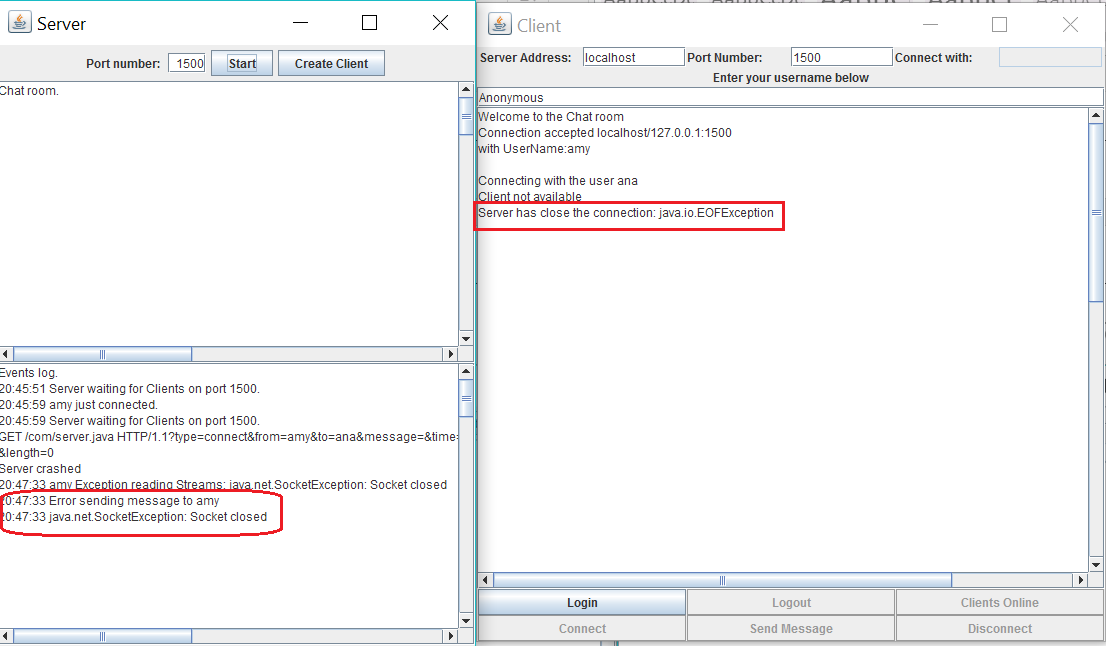
1. Create 4 clients by running ‘ClientGUI.java’ 4 times. Make connections between two of them. Now we can see 2 sets of connections among 2 clients who are conversing with each other. Traffic is maintained separately.



1. When a client gives a bad name for connection i.e., requesting for a client who is not registered with the system currently, then server sends ‘Client not available’ message.



1. Clients are notified whenever server crashes by clicking on ‘Stop’ button or by closing the window X Button. Clients are also logged out of the system



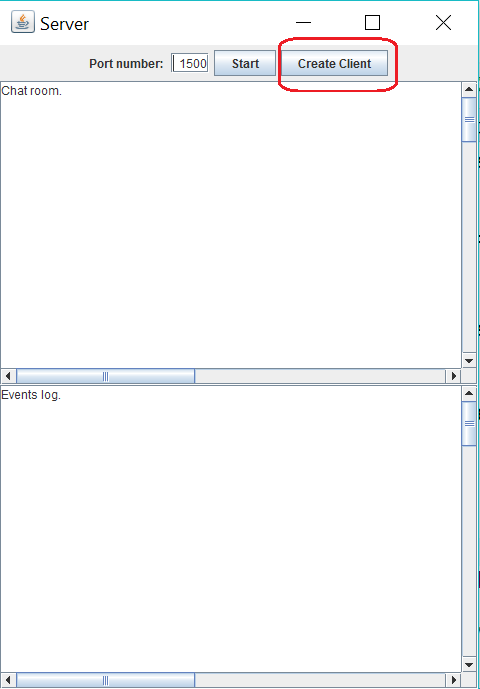
### Scenario 1 Assumptions

* No special characters while sending message or giving a username.

As the HTTP message format have characters like ?, /, &, messages with such special characters cannot be parsed correctly at server side. Other special characters like @, -, . are working when I checked but it would be better to avoid.

## Scenario 2

Steps to be followed are same like the above scenario but instead of running ‘ClientGUI.java’ to create clients, we can choose ‘Create Client’ option present on server window.



### Scenario 2 Limitations

* If any of the client/server window is closed by selecting X button of window, then all the windows get closed.
* Special characters like ?, /, & in user name or message cannot be parsed correctly.

# Extra Functionality

* Server is **Multi-threaded**.

This functionality can be verified by running the ‘ServerGUI.java’ twice and specifying different ports, say 1500 and 1501. Now there are 2 servers that runs on these ports. Clients can login by specifying either of the ports and the communication traffic will be maintained separately.

* A client can request for list of **online clients** at that time.

This can be verified by selecting the ‘Clients Online’ option on the client window.

# References

* <http://www.dreamincode.net/forums/topic/259777-a-simple-chat-program-with-clientserver-gui-optional/>
* <http://www.tcpipguide.com/free/t_HTTPResponseMessageFormat.htm>
* <http://www.tcpipguide.com/free/t_HTTPRequestMessageFormat.htm>
* <https://www.youtube.com/watch?v=vCDrGJWqR8w>
* <http://www.jmarshall.com/easy/http/>