Live Chatroom Project Report

Project Statement of work:

Language: Java

Library: No third-party library.

OS: Windows10/Windows11

JAVA: Azul Zulu Community 16.0.2

Topics: Multi-thread and Sockets, Custom classes
Function: Implement all the function in proposal.

1. **Client**
   1) **Log in with username and password**
      a) If valid, log in.
      b) If invalid, pop-up window tips, re-enter username and password
   2) **Functions served by the server**
      a) **Display online user list.**
      b) **broadcast message to all users**
         i. Validate receiver when Client sends a message to another Client
      c) **Send a message to one user**
         i. Validate the receiver's ID when Client sends a message to another Client
         ii. If the receiver is online, display a message on its windows
         iii. If the receiver is offline, it will fetch the offline file from server when next logging in
      d) **Send files to one user**
         i. Validate receiver's ID when Client sends a message to another Client
         ii. If the receiver is online, display a message on its windows.
         iii. If the receiver is offline, it will fetch the offline file from the server when next logging in
      e) **Display registered users.**
   3) **Log out**
      a) **Close the window to log out.**
         i. Send the disconnection message to the server.

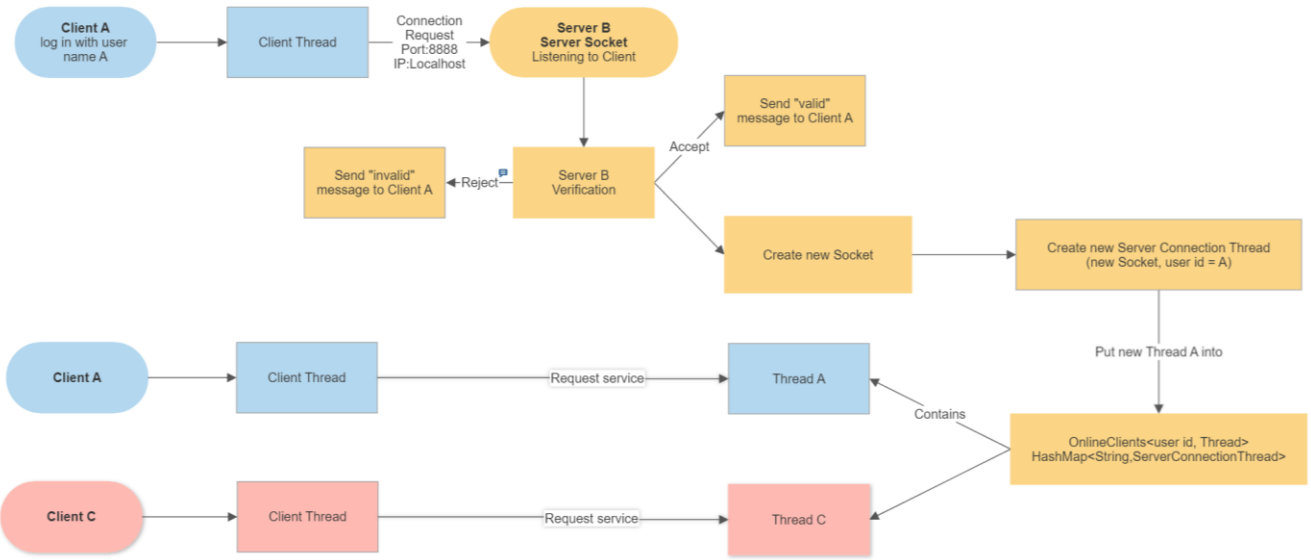
   4) **Display the chat history on client's GUI window.**

2. **Server**
   a) **Maintain username and password data, and validate username and password when user logs in.**
   b) **Maintain the thread pool, and create a new thread when new client connects.**
   c) **Single-server Multi-client.**
   d) **If the receiver is online, forward the message to the receiver.**
   e) **Maintain the offline message. If the receiver is offline, store the message and wait for receiver's next logging in.**
   f) **Display the journal file on server's GUI window.**

Video description:

File name	Corresponding Functions
1.mp4	1.1).
2.mp4	1.2).a).
3.mp4	1.2).b).
4.mp4	1.2).c).
5.mp4	1.2).d).
6.mp4	1.2).e).
7.mp4	1.3).a) & 1.4)
8.mp4	2.a).-f).

Process diagram (New client A connect to server)



Custom Classes in Client

Class	Description
Message	1) Share the class with server. 2) Serves the communication between client and server. 3)include sender, receiver, content, sentTime, type, fileBytes, fileLen = 0, dest, source
User (implements Serializable)	1) Share the class with server. 2) Serves the verification function of server. 3) Identify the thread and socket created in server.
ChatView (extends JFrame)	Display the client GUI window. Users can operate from this interface
Service	Main function of Client (The same as above)

Custom Classes in Server

Class	Description
Message	Share the class with the client.
User (implements Serializable)	Share the class with the client.
ServerConnectionThread(extends Thread)	When a new client is connected to the server, the server will create a new thread containing a new socket to serve the new client.
ServerView (extends JFrame)	Display the server GUI window. The Server manager can operate from this interface
Service	Main function of Server (The same as above)

Implementation:

Function	Client	Server
Client log in with username and password	Establish TCP connection at server's IP address(localhost) and port (8888), send a message to Server	1) Stored username and password data in a HashMap named **usersDatabase<uid, password>** 2) Verification. 3) If correct, create a new socket and new thread, store in a HashMap named **OnlineClient<uid, thread>**
Client A Fetch the offline message. **(Automatically happens when client A logging in)**	log in	1) Stored the offline message data in a HashMap<String, ArrayList<Message>> named **offlineMessages<uid, message>** 2) Query the **offlineMessages<uid, message>**, send to A, and delete the sent message in offlineMessages.
Client queries online user list	Send query message to server	Send back the key of **OnlineClient<uid, thread>**
Client queries valid user list	Send query message to server	Send back the key of **usersDatabase<uid, password>**
Client A send message to B	Send message to server.	1) Query the **usersDatabase<uid, password>.** If B does not exist, send back an error message. 2) Query the **OnlineClient<uid, thread>**, i) If B is online, forward the message to Client B ii)If B is offline, store the message in **offlineMessages<uid, message>**
Client A send message to all	Send message to server	1) Query the **usersDatabase<uid, password>,** and get all registered users. 2) Forward message to all the registered users (except for A itself)
Client A send file to B	Send file message to server	1) Query the **usersDatabase<uid, password>**, If B does not exist, send back an error message. 2) Query the **OnlineClient<uid, thread>.** i) B is online, forward the file message to B directly ii)B is offline, store the file message in **offlineMessages<uid, message>**

Client A logs out (Click the close-window bottom)	Send disconnection message to server	1) Close the Socket, 2) Remove the thread with key = uid **usersDatabase<uid, password>**, If B does not exist send back an error message.

Custom Data Structure

Name	Data Type	Variables	Description	Generated by
Message.type	String	“LogIn”	User log in successfully	Server
		"wrong password"	Correct username, wrong password.	Server
		"InvalidReceiver"	Username does not exist	Server
		"returnList"	The content is online user list.	Server
		"returnUserList"	The content is valid user list.	Server
		"getList"	Get online user list.	Client
		“getUserList”	Get registered user list.	Client
		“userToOne”	Forward message to another user.	Client
		“userToAll”	Broadcast message to all other users.	Client
		"file"	Send file to another client	Client
		"logOut"	Client want to log out.	Client
Message.fileBytes	byte[]	file	File transfer	Client
Server.usersDatabase <uid, password>	HashMap <String, String>	uid	Client’s username	Server
		password	user’s password	
Server.offlineMessages <uid, message>	HashMap<String, ArrayList<Message>>	uid	Client’s username	Server
		message	All the offline message s sent to the uid	
Server.OnlineClient <uid, thread>	HashMap<String, ServerConnectionThread>	uid	Client’s username	Server
		thread	The thread server created for the new client	