Technical Overview of Multi-Client Server Application Built on Java

Introduction

This report provides a technical overview of a multi-client server application built on Java, which allows users to communicate with each other through the application. The report will provide detailed instructions on how to run the application, its features, and limitations.

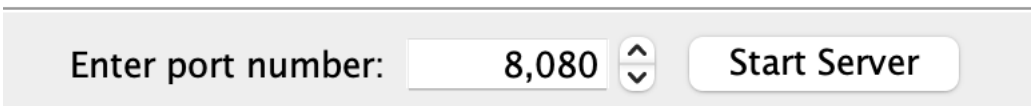
Application Overview

The Java-based multi-client server application consists of two main components: the server and the clients. The server component is responsible for connecting multiple clients, managing user requests, and distributing messages among the clients. The client component is responsible for connecting to the server, sending and receiving messages, and displaying the user interface.

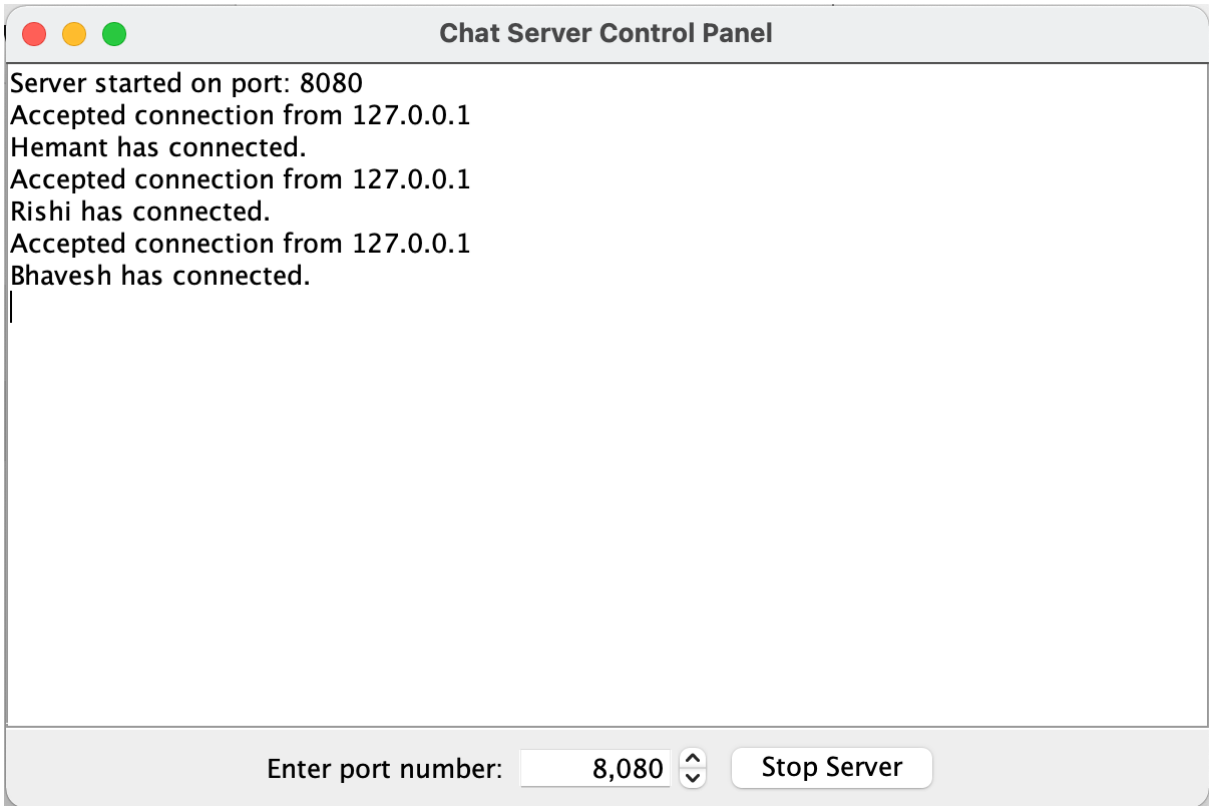
Running the Application

To run the application, the user needs to follow the below steps:

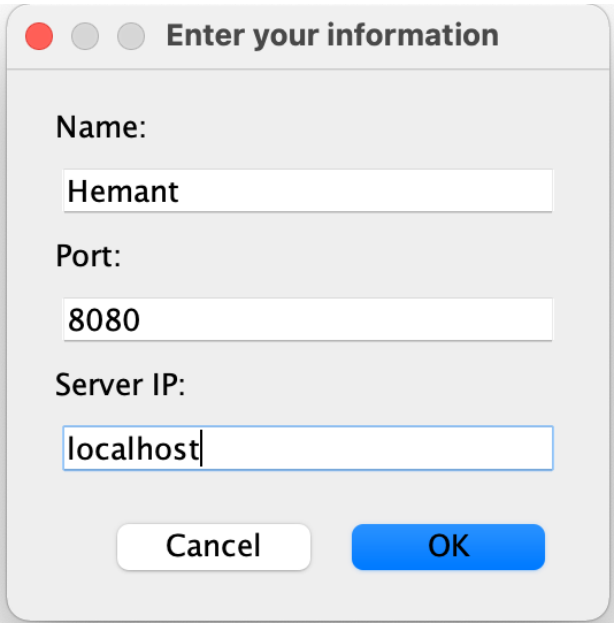
- Launch the ServerGUI.java file to start the server interface which prompts the user to enter a port number. The port number is limited between 0 to 65535. By default the port number is set to 8080.



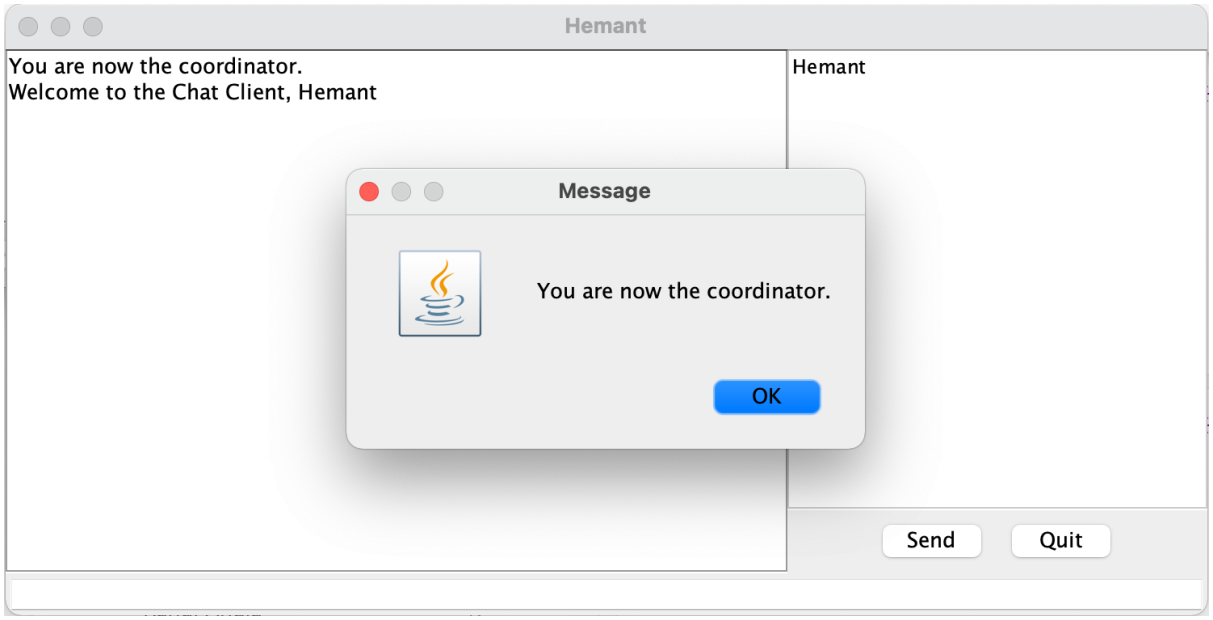
- After entering the port number, the server will get launched, and a GUI will be displayed, which allows the user to stop the server and display if any new client joins the server.



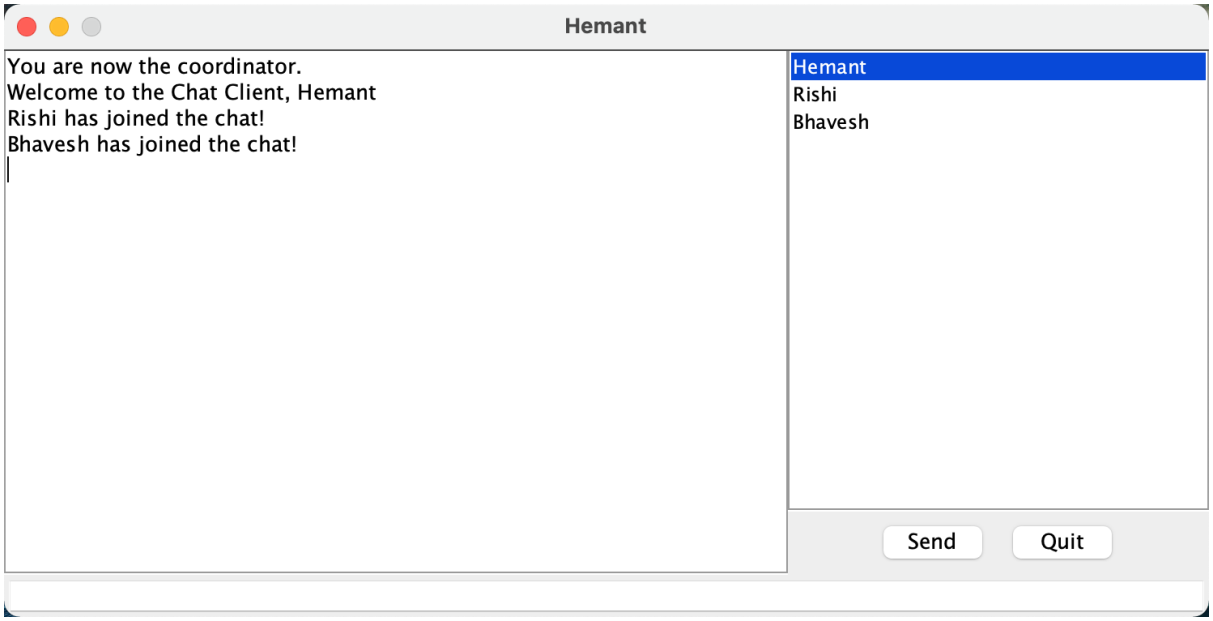
- To launch new clients, the user needs to run the ClientMain.java file, which allows creating multiple clients. The new client will have to enter a unique username, port, and IP address to connect to the server.



- When the first client joins the server, they automatically become the coordinator, if the current coordinator quits then the second person in the user list becomes the new coordinator.

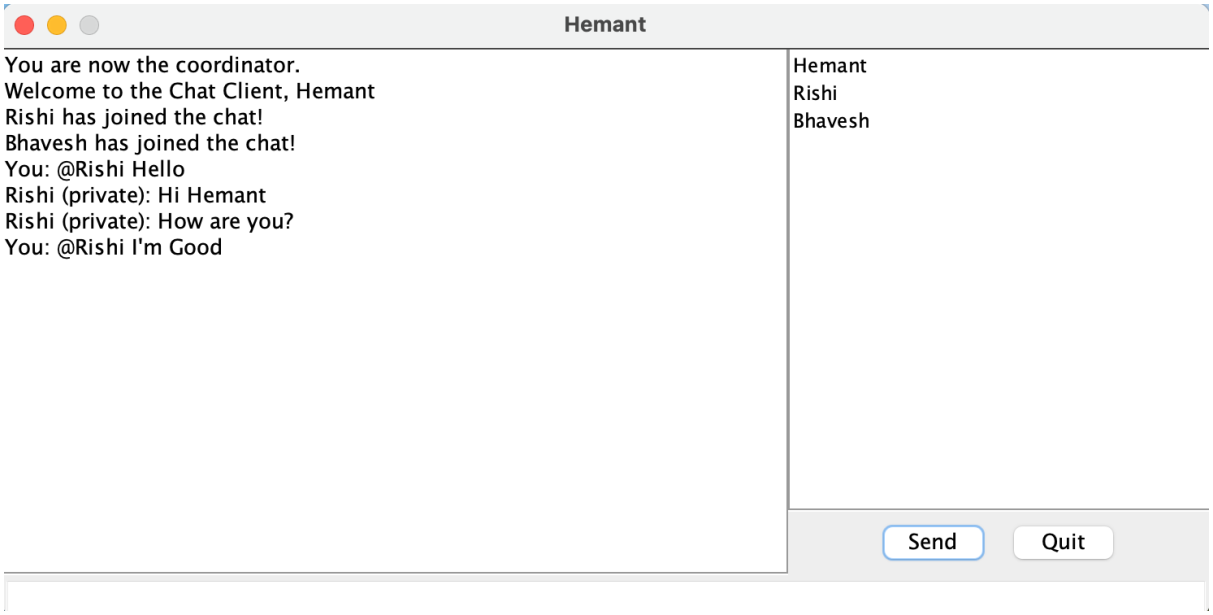


- Once the client is connected, the client will have the ability to send broadcast messages, private messages, view the list of active users and their information, and if the user is an admin, they also have the ability to kick users.

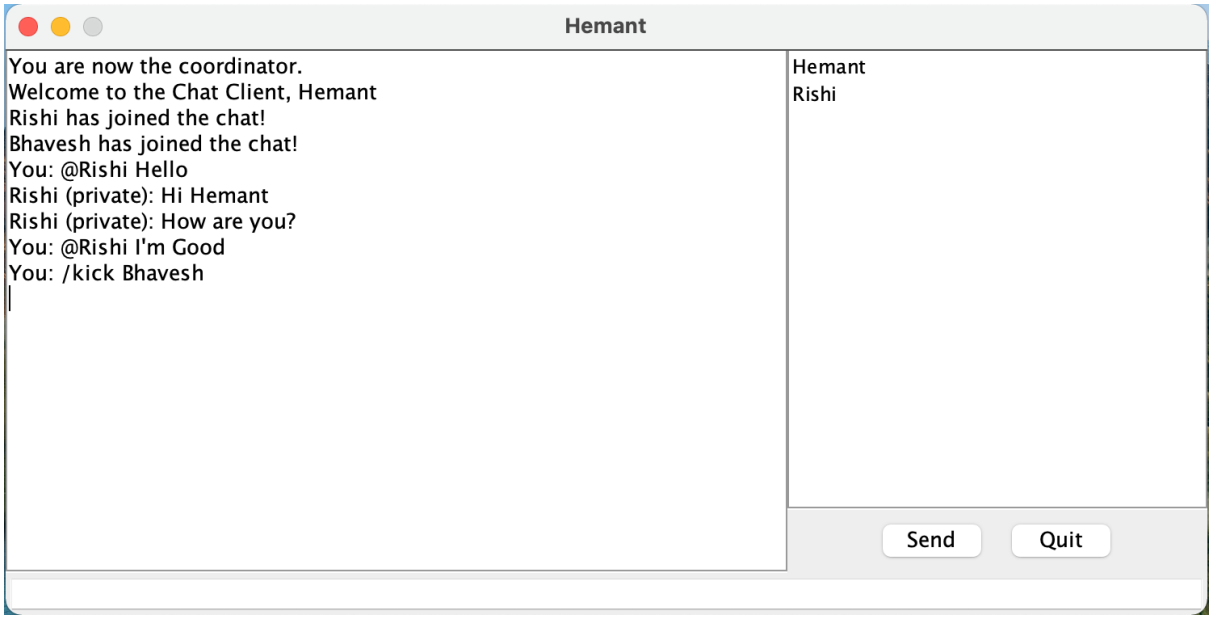


- If the username is duplicate, the application would not allow the client to connect to the server and display an error message asking the user to enter a unique username.

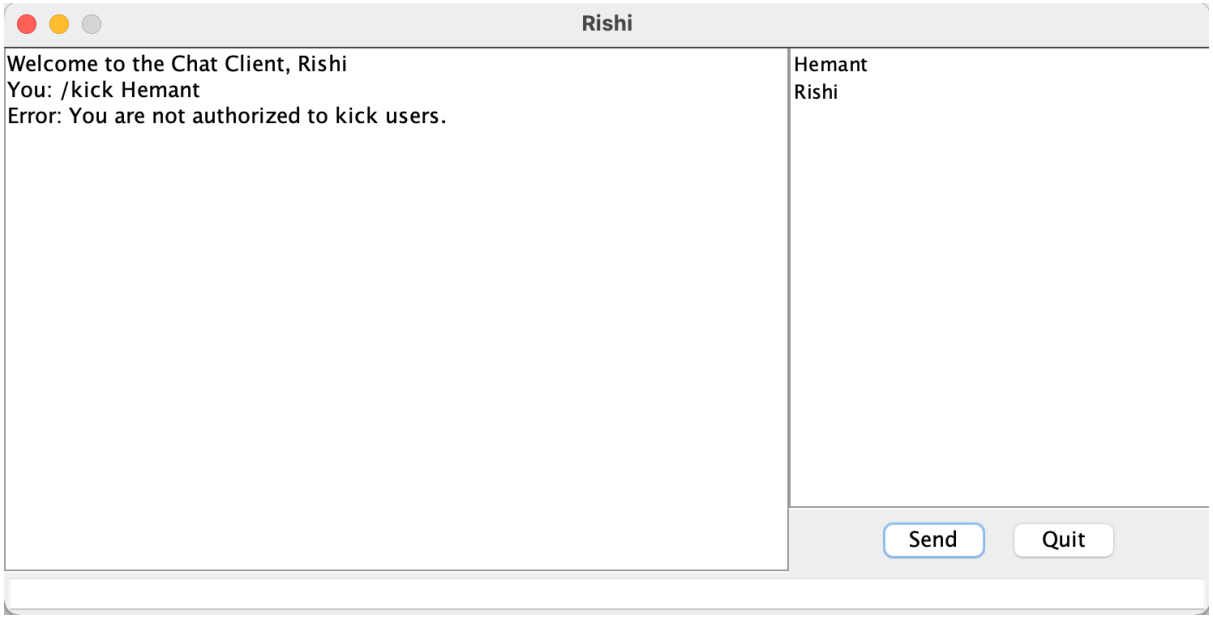
- To send a private message, the client needs to follow the format of "@receiver message".



- To view or kick a user, type in the message box /kick “username” and this will kick the user.

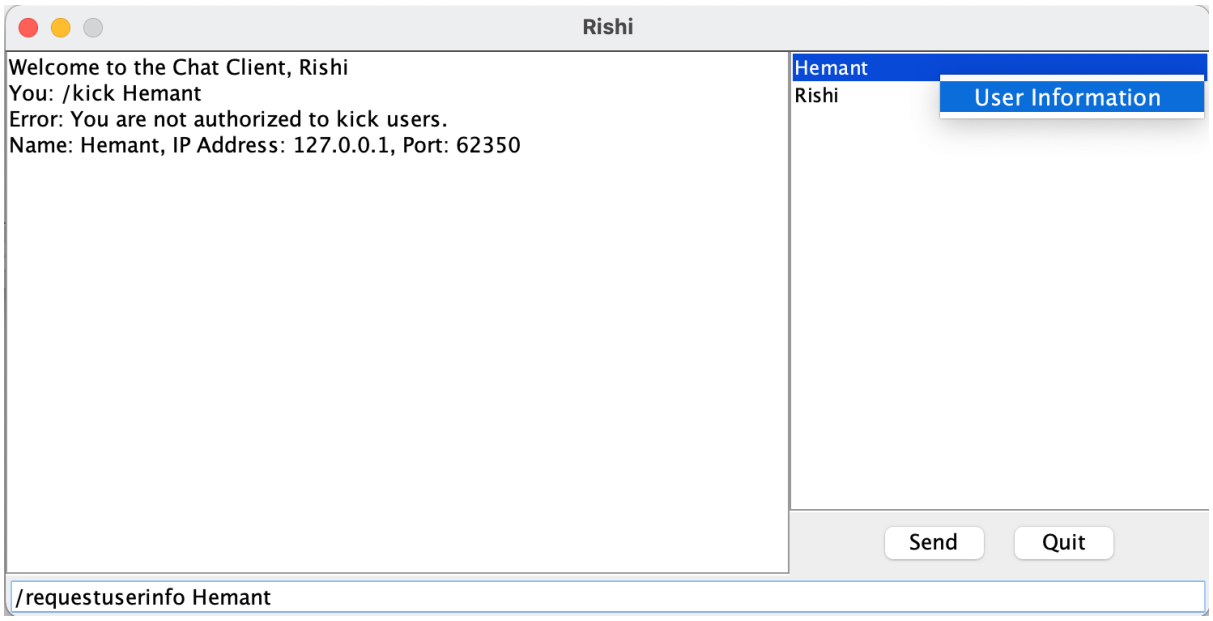


- If a non-admin user attempts to kick another user, an error message will inform them that they do not have the authority to do so.

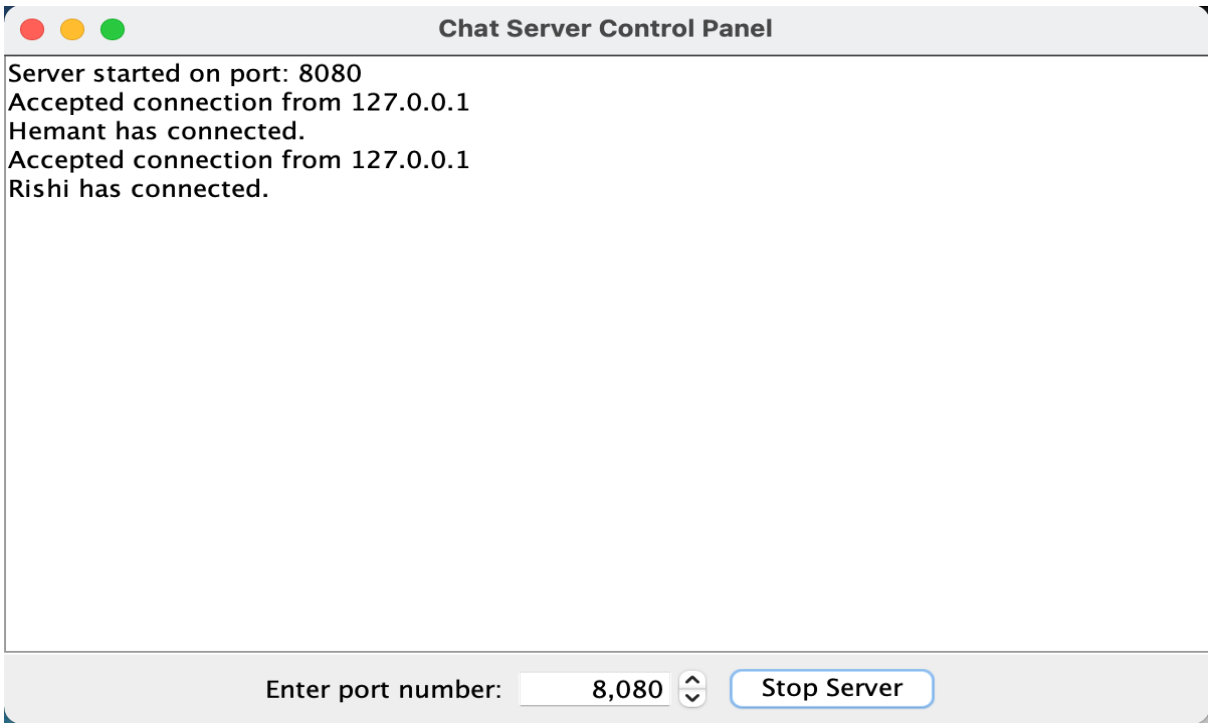


- If the admin kicks a user, the client will disconnect from the server and will be unable to communicate and this will remove them from the user list on the right.

- To view user information, right-click on their name in the side panel and select User Information or simply type the command “/requestuserinfo Username”.



- To close the server, click the "stop server" button in the ServerGUI . This will disconnect all connected clients.



Conclusion

This report has provided an overview of a multi-client server application built on Java, which allows users to communicate with each other through the application. The report has detailed the steps to run the application and its features, which include sending and receiving messages, viewing active users, and kicking users. The application has some limitations, which include limited port numbers and the need for unique usernames. However, the application provides an efficient way for users to communicate with each other in a secure environment.