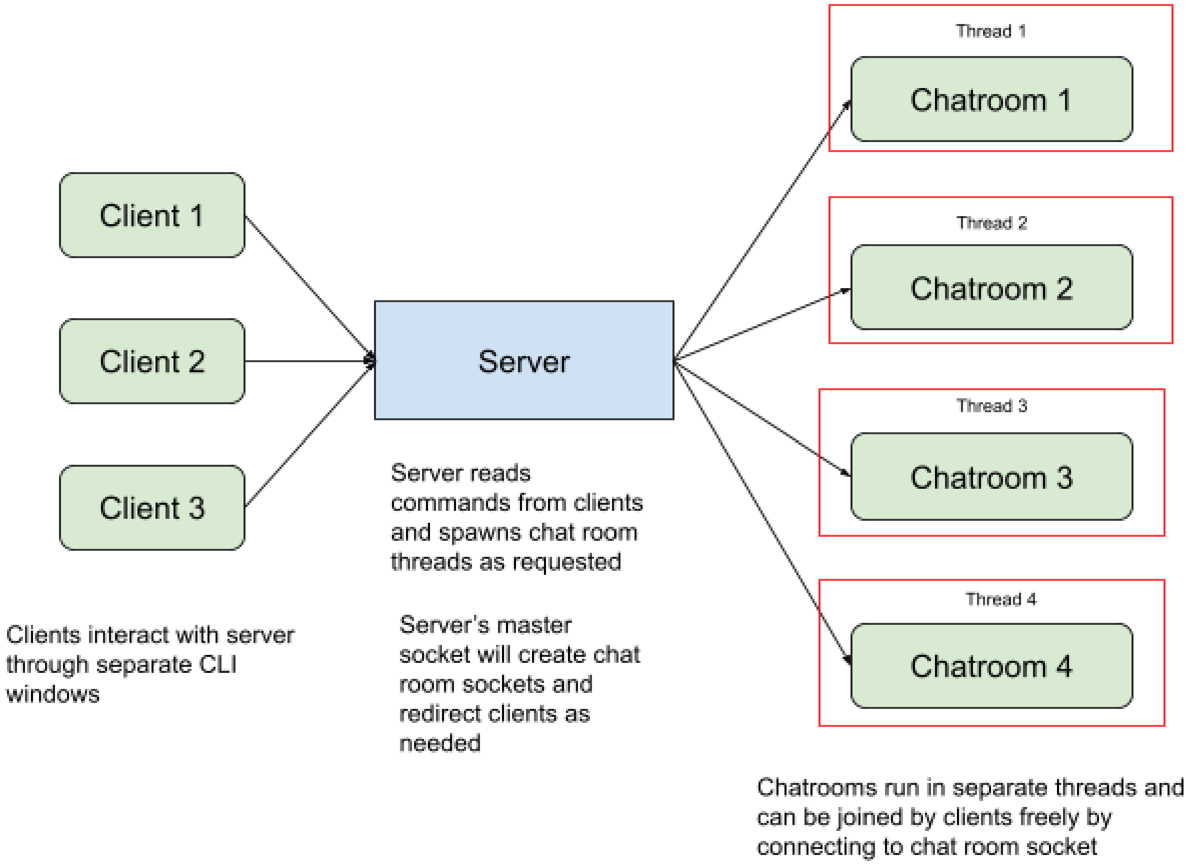
Ethan Cherian
830002546
CSCE 438-500

Machine Problem 1 Design Document

System Diagram

The system can (broadly) be modeled as follows



Server reads commands from clients and spawns chat room threads as requested

Server's master socket will create chat room sockets and redirect clients as needed

Clients interact with server through separate CLI windows

Chatrooms run in separate threads and can be joined by clients freely by connecting to chat room socket

Client

The client can connect to the server by providing the well-known port number of the server's main socket, allowing them to enter commands to the server with regard to the chat room service being provided. The user may create, delete, or list available chat rooms, or join a particular room and chat with associated members. The only ways for a user to leave a room upon joining are either the room being deleted by another user, or by unceremoniously terminating the program.

Server

The server was implemented in a *multi-threaded* fashion through the use of both *threads and the Linux `select()` command*. The server runs in a single infinite loop and uses one master socket, to which the user may write commands. When the server receives a `CREATE` request from a client, it spawns (and detaches) a new thread to run and provides it with the port number it is to run on. This thread is responsible for creating the master socket for the chat room, as well as for managing membership and sending messages among members within the room. When a client sends a `JOIN` request to the server, the reply includes the port number of the desired chat room, allowing the client to communicate on that socket (thus with the chat room thread) freely. The server keeps a global map representing all chat rooms that is updated when a new room is created or an existing one gains a new member; this data structure is importantly shared between threads, allowing the main server to access the data written by chat room threads, though there is a slight delay.

`select()` is used within the chat room threads to determine when a client has attempted to write to the chat room's socket, so the chat room can either add the user to the system or broadcast the message to all other users, depending on if the user is new or existing. Once a chat room is deleted, it notifies all users before terminating their connections and is then set as inactive by the main server.