Assginemt 4

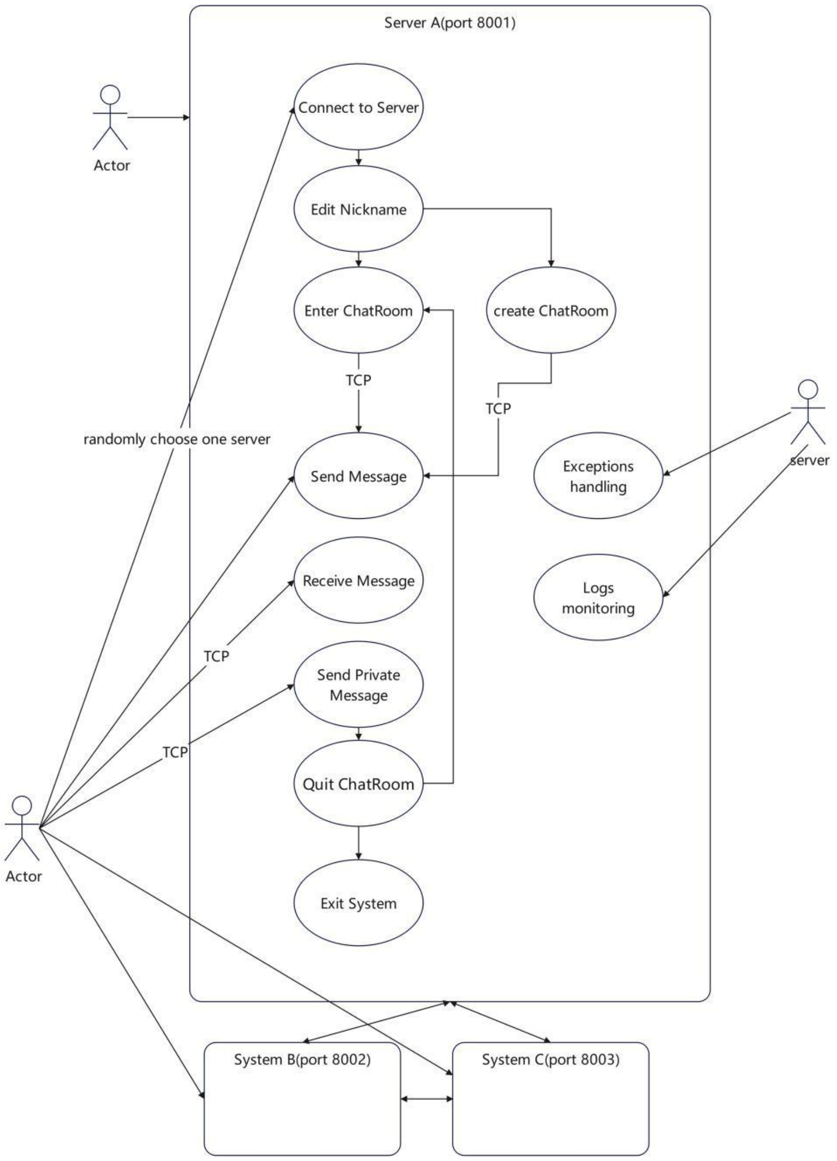
Zijing Li

000213415

Zijing.Li@student.lut.fi

UML Diagram

Use Case:



Functional Analysis:

I have completed the distributed system supporting multi-user communication as per Moodle requirements in this project and fulfilled all the required functionalities. Currently there are three server instances listening on ports 8001, 8002 and 8003, which reflects the scalability and higher availability of the distributed system. Users connect to the server using the TCP protocol and are authorised to perform the following functions:

1. connect to the server and display the IP address and port of that connection.
2. edit nicknames, which cannot be empty, or they will be prompted to re-enter them.
3. the user can select an already existing chat room or create a new one.
4. After entering a chat room, users can chat with other members in the group, send and receive messages from the current chat room.
5. Users can also chat privately with other members of the chat room, the content of the private chat can only be seen by the target person.
6. Users can log out of the chat room and rejoin at any time.
7. Users can disconnect from the server by typing "#exit".

The server side has the following functions:

1. transfer messages between clients.
2. Supports multi-threaded client connections.
3. performs exception handling
4. understands client operations, such as receiving information about user disconnections.

Distributed features:

1. Due to physical constraints, I can only simulate servers in different physical locations by using different ports, which also demonstrates its easy scalability. Users randomly connect to available ports after starting a client, thus increasing the load capacity of the system and ensuring, to some extent, that the load on each port is relatively balanced.
2. Clients connected to different ports can still communicate and share information across ports.
3. The system provides high concurrency so that multiple users can start connections, edit nicknames, and send and receive messages at the same time without being affected.
4. the system can track logs to understand the client's communication situation, will not cause the whole system to crash due to abnormal situations, and can also feedback the corresponding error messages.

Transparency:

1. 1. The client is primarily able to demonstrate transparency of access and location distribution because the user does not need to know how the chat room server was deployed and developed, or in which physical location the local functionality is distributed, and whether it is running locally or on a remote server; the user only

needs to interact with the functionality at the user interface in order to use the system. Users are randomly assigned to different ports without being affected. This is of no concern to the user and therefore hides the distributed nature of the system.
2. when a client has an exception or error, such as a user forcibly terminating a connection to the server or not complying with the communication rules, the error will not affect the normal operation of the server or other clients, and the server can determine what problems have occurred based on the type of error message, so that it can be maintained and updated.
3. You can create chat rooms, nicknames and other information on the client without affecting the normal use of other users.

Expandability:

1. the system can easily expand more available ports to support the use of more clients, enhance the load capacity of the server, and prevent overloading of individual ports. New nodes can be added simply by updating the available ports on both the client and server side, improving the performance of the entire system.
2. 2. Chat rooms can be added and created at will, supporting multiple chat rooms to communicate at the same time. As long as the server is not shut down, new chat rooms created by users will be kept and can be re-entered at any time.

Exception Handling:

1. I used a lot of try-except on the server side, which can solve most of the client-side exceptions, and print out the corresponding reasons in the terminal.
2. Any exception will not affect the normal operation of the entire server, even if the client is forced to terminate the connection, but also to ensure that it does not affect the use of other users, and release the resources of the port.
In addition, users entering or leaving the chat room can receive information about their entry and exit, reflecting the consistency of the distributed system, and real-time monitoring of the system process.