Use Case UC3.3: Chat

Primary Actor: Player

Stakeholders and Interests:

Player: wants to communicate with other players before and during the game.

Developers: want to make sure that players can successfully receive and send text message before and during the game.

Preconditions: Player has launched the game, selected network game mode and entered the lobby.

Success Guarantee: Player has either send or received a message(-s) and is sure that he/she can communicate with other players.

Main Success Scenario:

1. Player types his/her message in the input box.
2. Player presses "send".
3. Player sees his message in the chat window.
4. Another player reacts to the message by sending a response message.
Step 4 can be done multiple times.
5. Player sees the response(-s).

Extensions:

*a. Application crashes.

1. Player restarts the application.
2. Player selects network game mode
3. Player joins another lobby.

3a. Player can join the same lobby.

3a. Player could not connect to the chat server due to a socket exception

1. Player does not see his/her message in the chat window.
2. Player does not see any other messages.
3. Player leaves the lobby and joins it once again.

Step 2 can be done multiple times.

4a. Nobody reacted to the player's message.

1. Player sends another message in order to get a response.

Step 1 can be done multiple times.

1a. Player does not send any more messages.

5a. Some player could not connect to the chat server due to a socket exception so the player cannot see their messages.

Special requirements:

Each message should have the following format:
<username>: message

Each process of sending/receiving of a message should take less then a second.

Technology and Variations:

2a. Player can send messages either by pressing “send” button on the or by pressing Enter key.