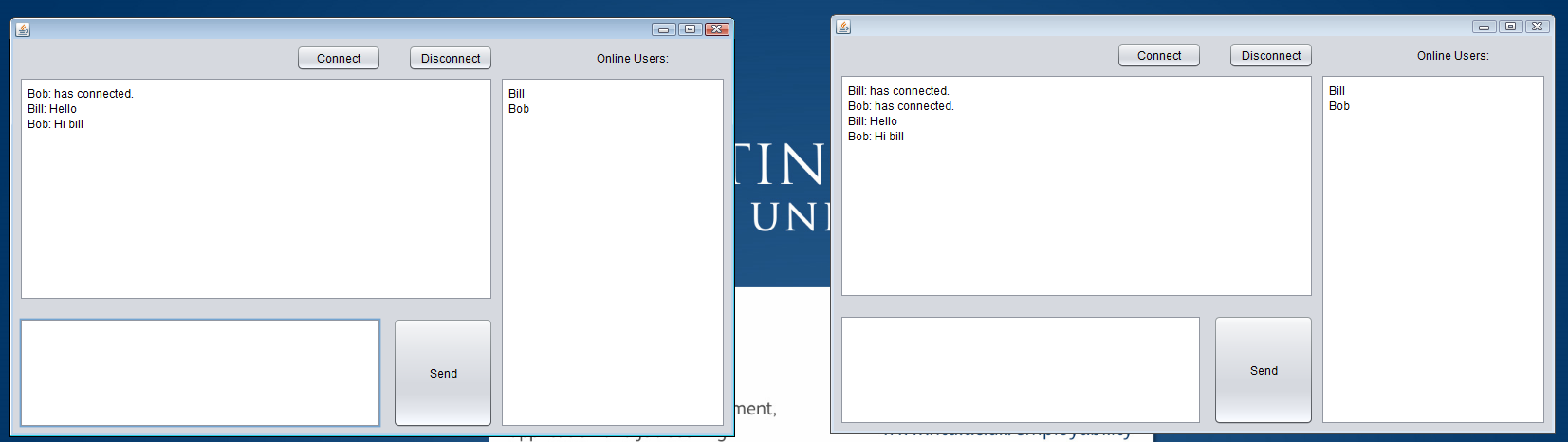
Systems Software Report

By Ahmed Ahmed(N0570451) and Lewis Pressley (N0557446)

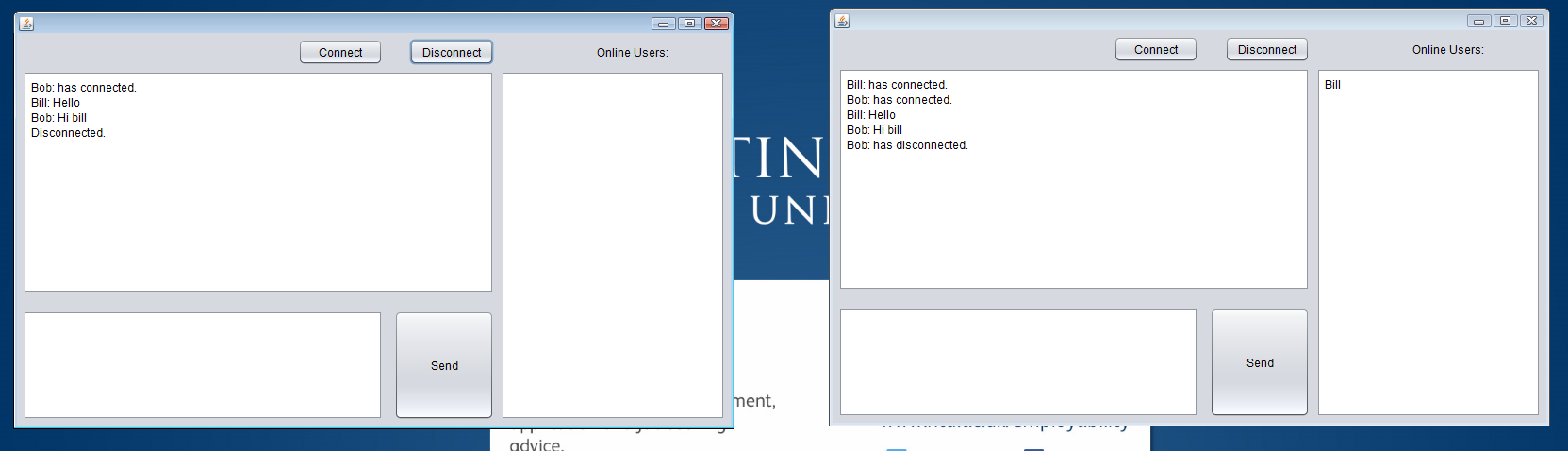
The following report is about the graphical social network system, which is a Spotify/Facebook-service using java in NetBeans. We have built a chat server.

Firstly the chat server works by having a server and client/s. The chat works by having a host and join button. If someone wants to host then they click the host button. This then activates a server with their own private port number and opens a chat window. In the chat window they then click connect to connect to the chat and there username will be displayed in online users. The user can then send messages to the chat text area and any online users can see it and communicate. The host can then leave by disconnecting. If a user wants to join this chat they will need to select the friend from the friends list and then click join. This will open a chat window, which is exactly the same as the hosts chat window.

This shows the chat when they connect and communicate:



This shows when a user disconnects:



Secondly, now an in-depth look into the client side code. So first of all we have a void function run(). This function has a string array called data and strings with different keywords to differentiate what type of message it is. These are connect, disconnect, done and chat. Data is the split up by ‘:’ and this puts it into three different parts, if the final part (data[2]) equals one of these commands e.g. chat, then it follows that if statement. So chat appends the chat text area. Connect, empties the chat area from any previous chats and calls function userAdd, which adds the data[0] (the username) to the userList text area. This allows all users online to be seen. The disconnect calls userRemove function and this takes the username and removes it from userList. Finally done sets user list to clear and calls the writeUsers function. We have two more functions called sendDisconnect and Disconnect and these ensure the user is disconnected and write it into chat so users no that user has disconnected.

Thirdly, the client side has three different buttons. Firstly the connect button. This checks to see if the person is connected, if they are it sends message to say already connected, otherwise it goes through these stages: Using a try catch it creates a socket using the serverIP and the hosts port. Then we have created a streamreader, which is a input stream reader, that gets the input stream from the socket. Then we have a reader which equals a new BufferedReader function taking the streamreader as its parameter and then we create a writer, which equals a new PrintWriter function that takes the output stream from the sock. Now the user is connected to the chat and we send a message to the chat saying that user has connected. We then flush the writer and set the connection to true, so that users don’t do this more than once.

Another button is the send button. This first of creates a string called nothing, which equals “”. We can then do a check to see if the inputTextArea equals nothing. If it does then no message is sent. If there are words then it will use the writer and print line the message username, inputTextArea and keyword Chat, so that it can identify what command is needed in run. The writer is then flushed. If this didn’t work, then the chatTextArea says message was not sent. The final button is disconnect, which simply sendDisconnect and Disconnect functions.