PA 2
Scott Clay
CSCI 3920

For my implementation of the asynchronous messaging app, I chose to have the server listening at a port and its adjacent port (port+1) on separate threads. When a client connects, its outgoing (command sending) socket connects to the main port, and it proceeds with logging in or creating a user, which logs the user in automatically after successful creation. This connection is processed in a new thread. Once logged in, the Server sets the user in it's system to online. If successfully logged in on the server, the client's incoming (message receiving) socket attempts to connect to the Server's outgoing port (port+1) with C|\<phone>|\<pwd>. The Server checks whether the online flag is set, and, if the user is morked online, the Server allows the secondary connection. This connection is saved in the User object for that user on the server, which allows the server to access it to relay messages. When a client sends a message to another client, the Server creates a new thread to send the message to the target client asynchronously.

A series of QUIT messages sent back and forth are used to disconnect the client and server. When disconnected, the client resets the sockets for reuse if the user chooses to reconnect.

On start of the client app, the user can choose to login, create a new user, or exit. Login and create new user will ask for a phone number and password. Create new user will log in automatically after creation. Logged in user will be able to send messages in the form M|\<target user>|\<msg>. Messages will be printed on the console when they are received. User will type QUIT to exit and disconnect.

A user is not allowed to login if that user is logged in elsewhere.