**README**

/\*CSci4061 F2018 Assignment 2

\*section: 10\*

date: 11/10/18

\*name: Jake Johnson, Ren Jeik,Ju-Yu Chou Fu

\*id: 5160192,5343975,5411243

\*/

Purpose

The purpose of our program is to communicate using a client, server, and users. By using pipes we can send messages from user to client, client to user, server to client, and client to server. Each user and server process can submit commands, such as list users and private messaging (p2p). We also implemented commands that are exclusive the the server, such as kick and exit, that dictate which user to kick and when to end the whole chatroom. Overall the purpose was to create a program that emulates a basic chat room in a way that uses our knowledge of pipes and communication between processes.

What Role each member had in the project

Jake Johnson(joh12533):

Worked on the warmup on both server and client side with Ju-Yu and Ren, worked on the methods for the server and client such as broadcast and p2p, and worked with Ren on Kill, Kick, and Cleanup.

Ju-Yu, Chou Fu (chouf001):

I designed how our program should flow. I built the server, client, and child process and let them be able to communicate with each other through pipes. I coded how server should handle commands from admin or users. I contributed effort to the methods in server such as add\_user, kick, kill, cleanup, mass cleanup, and exit. In the end, I also wrote comments and codes to handle errors, and polish our codes.

Ren Jeik Ong (ongxx107):

I implemented the kick method along with kill and cleanup functions. Also, I managed to do extra credit section which \seg works on user process.

**Instructions:**

Compiling the code:

To compile the code, use the makefile. From the terminal, use make clean, followed by make.

Using the Program from the shell:

Open two terminal shells in the directory of the code. On the first, run ./server to create a server process. This has the ability to use all the commands, such as \kick <user> or \exit, which kills the entire chat room.

Next, run ./client on the other terminal shell. This is the user process that has the ability to communicate with the server process. It can use commands such as \p2p <user> <message> to private message another user and \list to list the other users. Otherwise, by typing anything into chat it will send everyone that chat.

Assumptions:

We have assumed that it is okay to send the username when using P2P to identify who is sending the message. It was not told whether or not we had to do this, but decided to do it for usability and testing.

Error Handling:

On a basic level, anytime we write or read we catch errors for if the write or read process fails. We handle this by outputting a error message to the terminal using PERROR. For p2p function, we let server sent error message through pipe so that the senders can see whether their commands have been successfully executed.

Also to address all broken pipe errors, we made sure to close all the pipes, and to cleanup accordingly when a user is kicked.