CAB403 Assignment Report

Eugene Martens

N10318313

Tasks Completed
Able to implement all three 3 tasks

Notable bugs
- ^C within input buffer causes next command sent by the client to not be read properly
- Unsubbing during live feed has been blocked to avoid client crashing due to null pointers.

Group
Worked Alone

Data structures
Created a shared memory block that all process wrote to and from and can be found in shared.h/shared.c, created at the start of a new server init and cleared during server closing.

Client data was handled through the use of a localized linked list that stored channel substrictions. When functions NEXT and LIVEFEED are called, client subbed linked list is matched to retrieve relevant data and update posts and read index of the subscription.

Work is handled with a localized linked list found in worker.c/worker.h, where the jobs are appended to the list from a command or output from CHANNELS/LIVE FEED and NEXT, where poster_thread, outputs to the client and pops the job off the list. This way jobs are handled through one output stream and read on a read_thread on the client side and is akin to FCFS.

Forking
Upen receiving a connection from a client, server will invoke a fork process, that includes a server chat handler that will communicate with the client and accept commands and post responses via read and write.

Critical Section resolution
Shared memory data is locked and unlocked using pthread_mutex's, any threads or processes working on a shared memory information will first lock off the section before altering any data.

Threading
Three threads are initiated at the beginning of each server process fork, one for live feed, one for next and a job poster thread.

The job poster thread looks at the current worker link list and posts any jobs that have been attached to the list and pops them off when done. Client has a dedicated read_thread that catches any posts done by the server thread and is closed on a sig catch.

The server threads live/next and poster, are kept open using a while(no signet and threads_flag), and closed in the event of a Sig catch or when a client closes the session.

Instructions
A make file is included in the project file that you can call within the terminal via “make”
Both server and client as well as any dependencies are compiled and ready to use.