**Project**

**Introduction:**

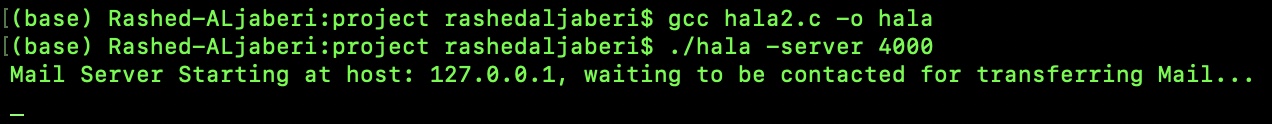
In this course, we learned a lot about networks layers; like Application, Transport and Network. In this assignment, we explore the SMTP application layer protocol and the TCP transport layer protocol, and of course, the IP network layer protocol. The assignment requires programming in c/c++ to implement a SMTP server that receives/sends messages and a client that does the same. Additionally, the server will connect to a number of other clients whom the first client is trying to communicate with and the server will store the messages and then eventuality deliver it to its rightful client when they connect. The program would also demonstrate how sockets work.

**Scenario:**

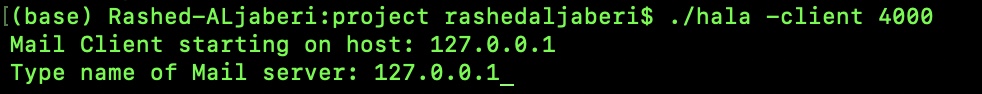
In this demonstration, we will see a scenario where a client sends an email to another client through a server that would pass the message.

First, we will run the code on server

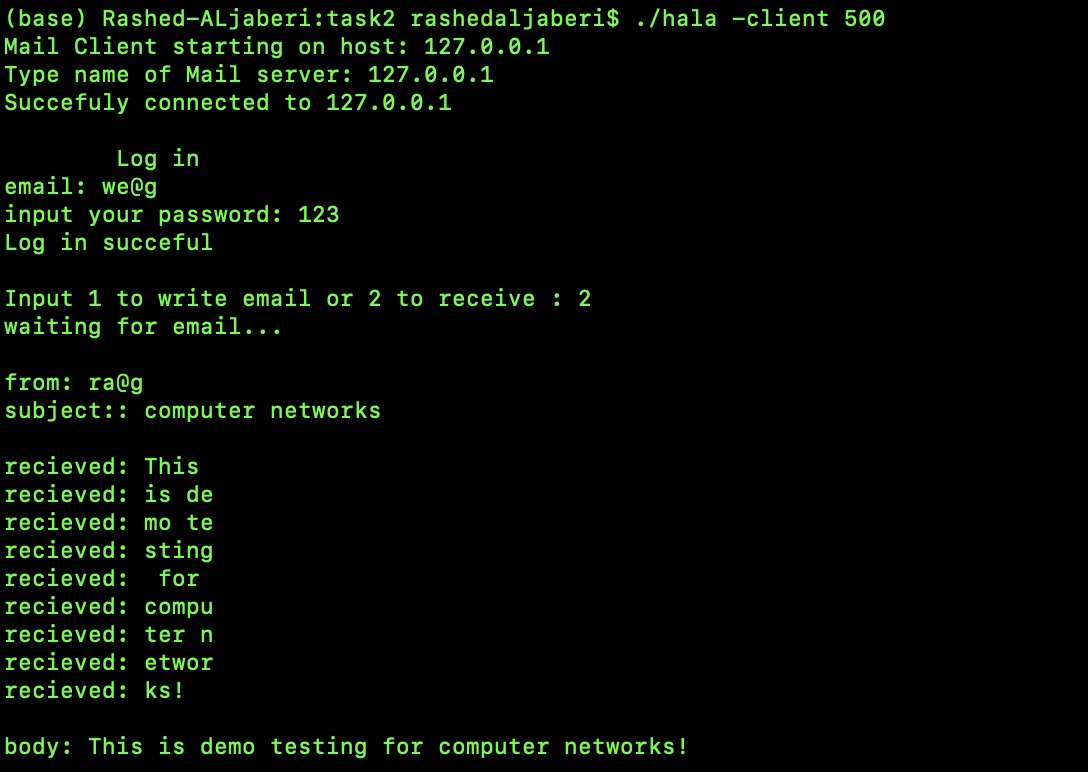
**Server**



Then we run it on the first client (the sender) and we would be prompted to enter the mail server

**Client 0**

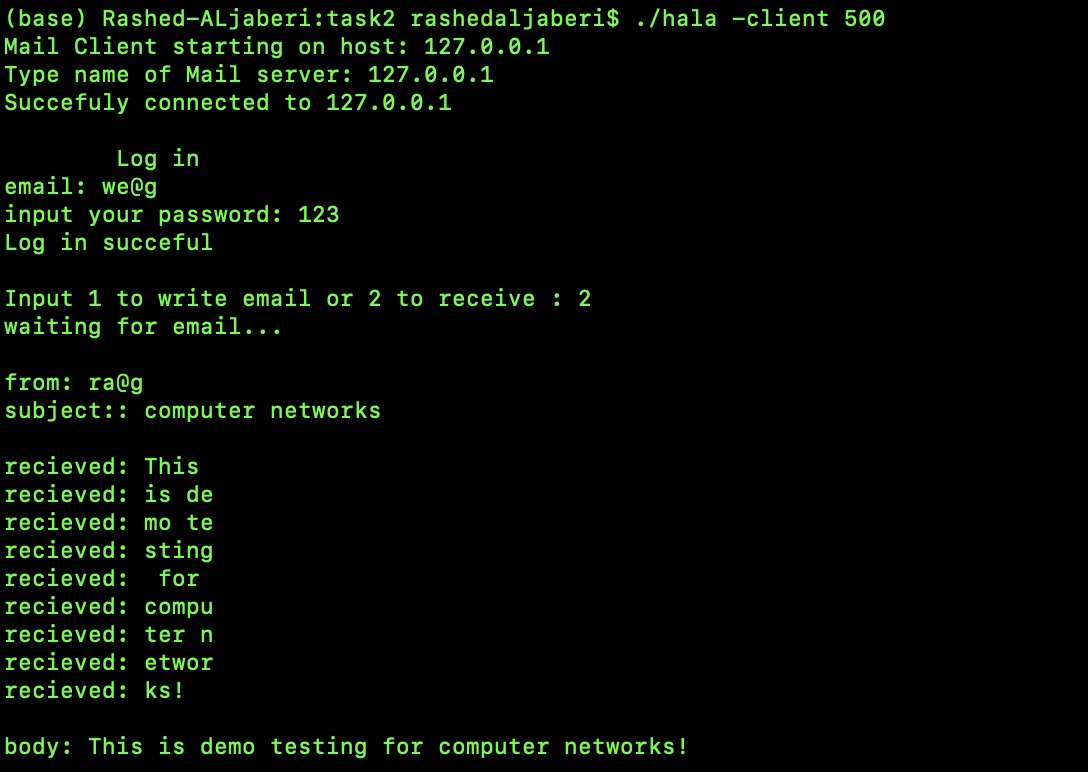
If connected, then we get

**Client 0**

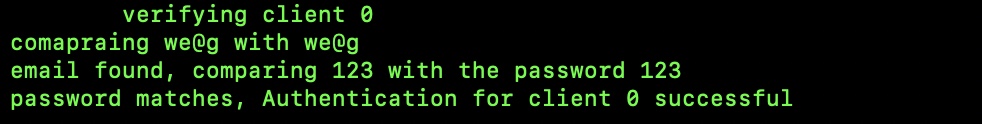
**Server**



Then we are prompted to log in. The server will check if the Credentials are correct

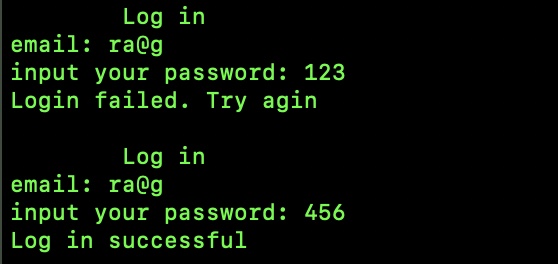
**Client 0**

**Server**

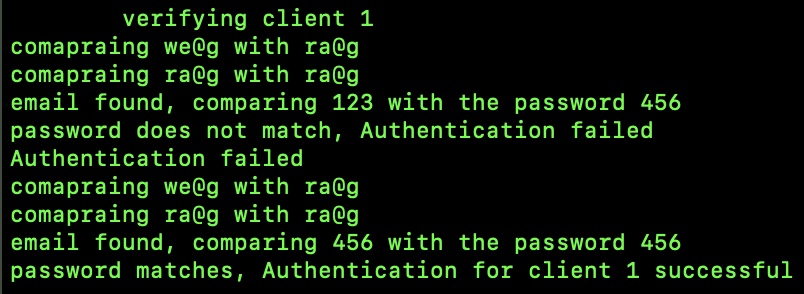


Now we can either send the email or connect from the other computer that will receive the message. It doesn’t matter here because in my code, the sever creates a different process using fork for each client. So each process is independent, and as such, the server is not sequential when it comes to handling clients. Multiple clients can connect concurrently and there is no strict flow that must be followed. In other words, if there is a new email in the server for a client, they will get it whenever they connect without any preconditions that depends on the state of other clients. This allows multiple clients to send any number of emails to any number of other clients at once. Whenever each of the recipient clients connect, they will get their new emails.

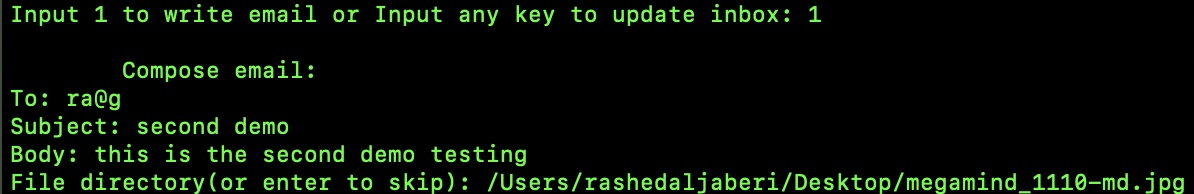
**Client 1**



**Server**

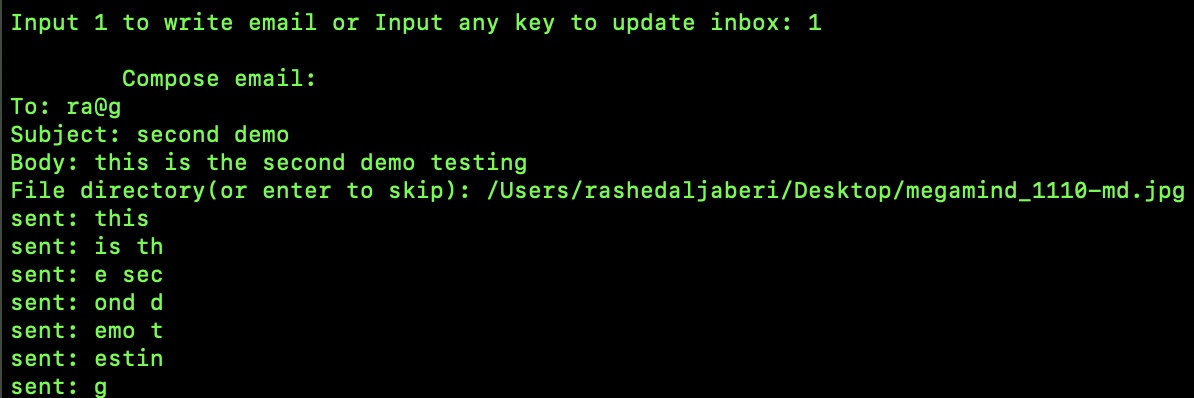


Now both clients are prompted to either send or update inbox

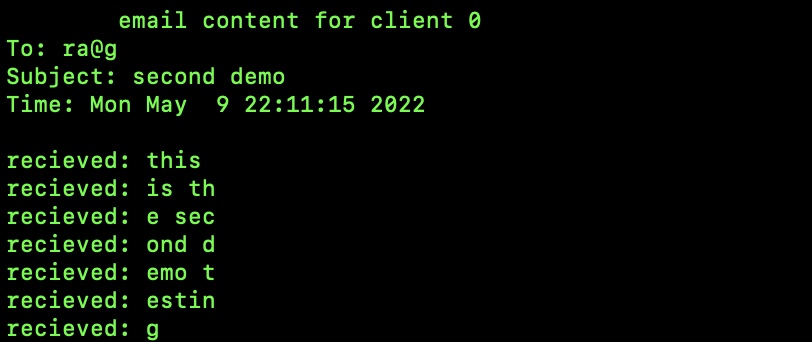
**Client 0**

After the client chooses to write an email and fills the email, it will be sent as chunks

**Client 0**

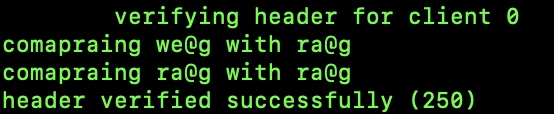


**Server**



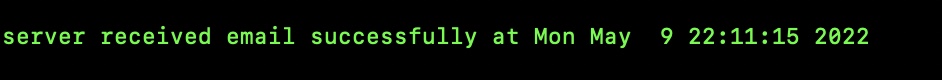
Then the server verifies the ‘to’ header and replies to the client as such (either 250,550) and save the file locally

**Server**

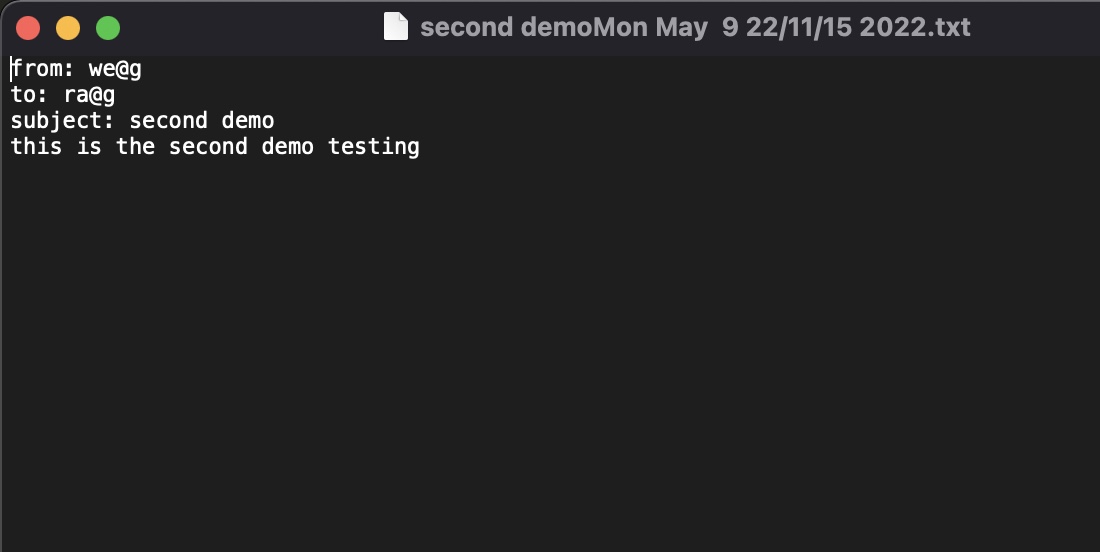


And the client gets the conformation and terminates

**Client 0**

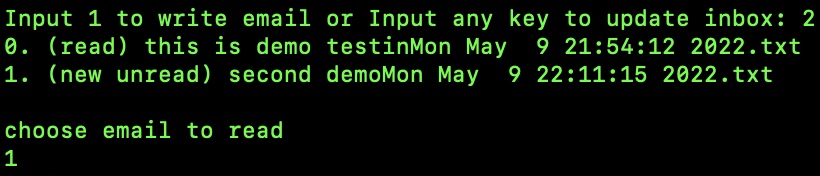


The txt file with its attachments are saved in the sender’s sent folder, server’s sent folder for the email sending it, and the server’s inbox for the email receiving it.

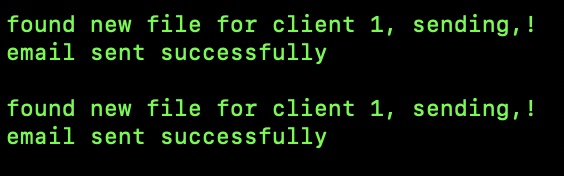


Now the receiver client will request to update their inbox. The server will send all new files to the client.

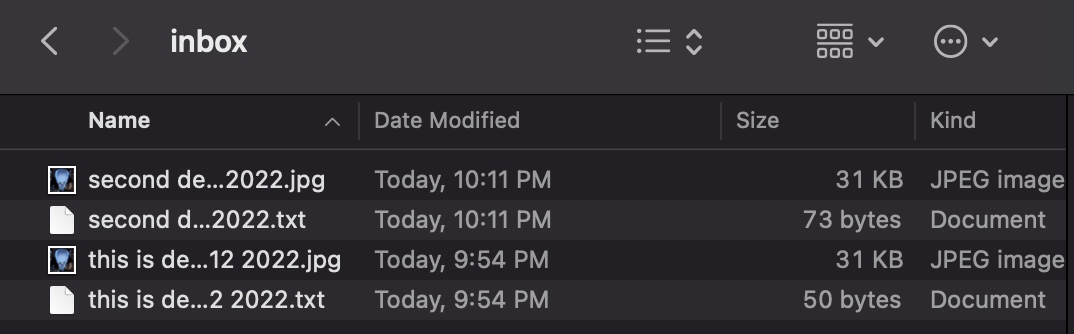
**Client 1**



**Server**

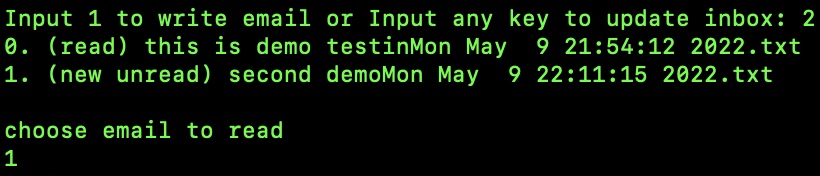
****

All available new files will be sent and the files will be saved in the client’s inbox



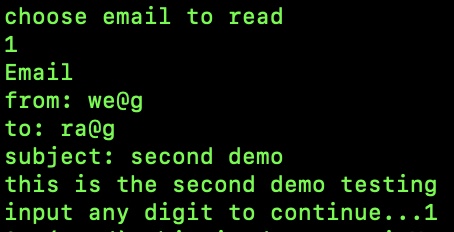
The new files will be marked as unread until the client decides to read them

**Client 1**

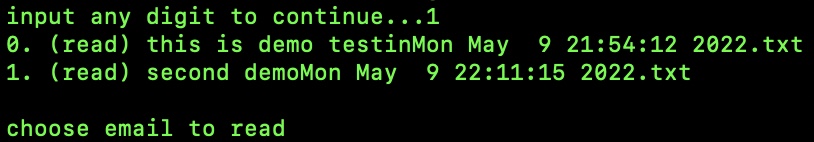


After choosing what email to read, the email is displayed and is marked read so that the next time it shows it is read

**Client1**

****

**Client1**



**Conclusion:**

This assignment requires the use of c/c++ to be able to send SMTP emails through different clients and a server using sockets that utilizes a TCP transport protocol.