Implemented : 1 server and N clients.
Say N clients may connect to the server. Client 1 wants to talk with client m. How do you manage multiple clients and select the one you want to talk to?

Features:
1. Client will be shown a list of Active Idle Clients in the application, after successful login.
2. Client may choose one of the other clients to establish a one-to-one chat session, without his/her permission.
3. After establishing connection, both the users may chat with one another on a one-to-one basis (via server).
4. While leaving, if there exists connection with another Client, it will be safely closed and the other client stays inside the application.
5. Server maintains a Database that record's new client's and deletes outgoing client's. (Mapping done w.r.t. sockfd of client's in Server).
6. A client has the following commands:
   Request for the List of Active Idle Clients. = "REQUEST"
   Connect to a Specific Client = "CONNECT:<remoteUsername>"
   Leaving the Application = "bye" or "Ctrl-C"

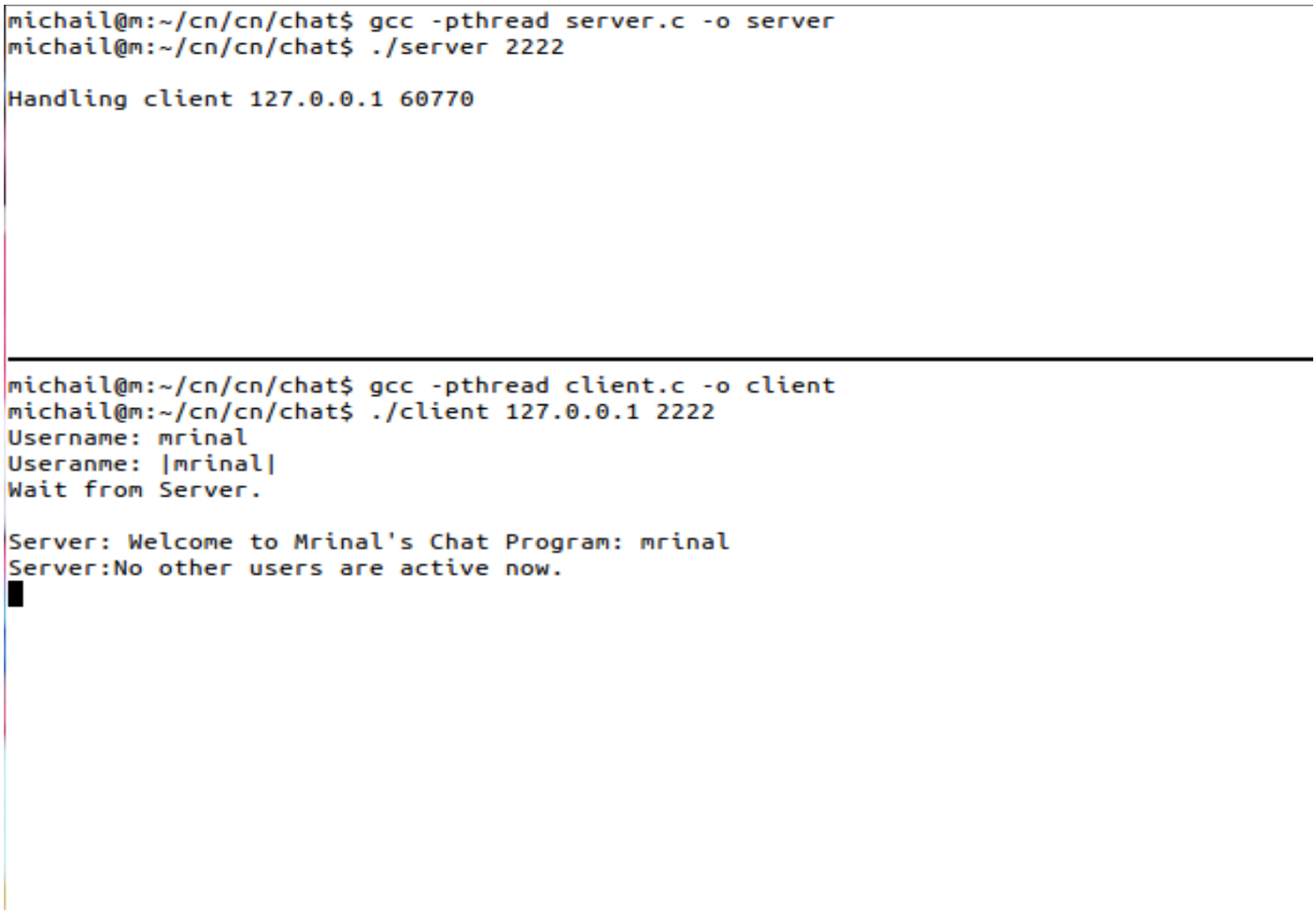
Assumptions:
1. Client on connect, should mention its 'unique' Username.
2. If client-1 chooses to connect to idle client-2, the client-2 will automatically connect. No permission will be requested.
3. User input on Server indicates 'Stop Application'.
4. Client may uses the following commands:
   Request for the List of Active Idle Clients. = "REQUEST"
   Connect to a Specific Client = "CONNECT:<remoteUsername>"
   Leaving the Application = "bye" or "Ctrl-C"

Technical:
1. Client maintains two threads, one to receive from Socket, and another to read from StdIn.
2. Server maintain sockfd's in FD_SET data structure, and uses select function.

Constraints:
1. At max, 30 clients can be supported at once.
2. Username cannot exceed 16 characters.
3. Only Unique Usernames allowed.

Screenshots:

1. Client on connecting to the server.



2. 1st Client requests for the list of active users. “REQUEST” command.
. Client-1 connects and then Client-2 connects.
. Now, client-1 requests the list of active users.

```
michail@m:~/cn/cn/chat$ gcc -pthread server.c -o server
michail@m:~/cn/cn/chat$ ./server 2222

Handling client 127.0.0.1 60799

Handling client 127.0.0.1 60800
```

```
michail@m:~/cn/cn/chat$ ./client 127.0.0.1 2222
Username: mickey
Wait from Server.

Server: Welcome to Mrinal's Chat Program: mickey
Server:No other users are active now.
REQUEST

Server: Active Users are:
minnie
```

```
michail@m:~/cn/cn/chat$ ./client 127.0.0.1 2222
Username: minnie
Wait from Server.

Server: Welcome to Mrinal's Chat Program: minnie
Server: Active Users are:
mickey
```

```
[0] 0:./client*                                                    "m" 21:57 31-Aug-16
```

3. Two clients are connected and their chat session.
Use of CONNECT:<client_name>

1. Client-1 & 2 are logged in.
2. Client-2 connects to Client-1 using “CONNECT:mickey”
3. Connection is established and both users are notified.
4. Both clients may communicate on a one-to-one basis.

```
michail@m:~/cn/cn/chat$ gcc -pthread server.c -o server
michail@m:~/cn/cn/chat$ ./server 2222

Handling client 127.0.0.1 60810

Handling client 127.0.0.1 60811
Connection established between mickey and minnie...
```

```
michail@m:~/cn/cn/chat$ ./client 127.0.0.1 2222
Username: mickey
Wait from Server.

Server: Welcome to Mrinal's Chat Program: mickey
Server:No other users are active now.
Server: You are connected to minnie
minnie:How are you?
I am good.

You: I am good.
How is the whether?

You: How is the whether?
minnie:It's fine.
```

```
michail@m:~/cn/cn/chat$ ./client 127.0.0.1 2222
Username: minnie
Wait from Server.

Server: Welcome to Mrinal's Chat Program: minnie
Server: Active Users are:
mickey

CONNECT:mickey
Server: You are connected to mickey
How are you?

You: How are you?
mickey:I am good.
mickey:How is the whether?
It's fine.

You: It's fine.
```

```
[0] 0:./client*                                                    "m" 22:18 31-Aug-16
```

4. Existing connection closed by Client-1 and the other client stays in the application.

1. Client-1 and Client-2 are communicating with eachother.
2. Client-2 says 'bye' and exits the application.
3. Client-1 stays connected to the server.

```
michail@m:~/cn/cn/chat$ gcc -pthread server.c -o server
michail@m:~/cn/cn/chat$ ./server 2222

Handling client 127.0.0.1 60792

Handling client 127.0.0.1 60793
Connection established between mickey and minnie...
Connection Closed by |minnie|
Connection Closed by ||
```

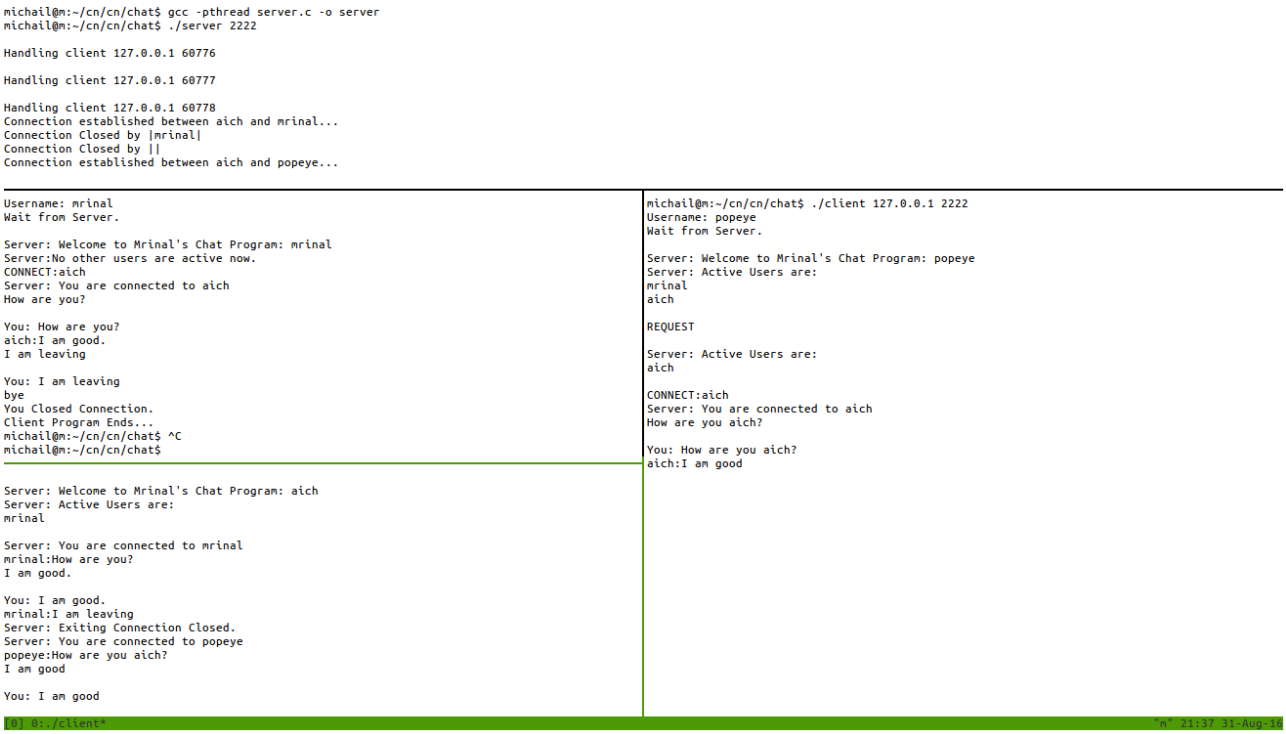
```
michail@m:~/cn/cn/chat$ ./client 127.0.0.1 2222
Username: mickey
Wait from Server.

Server: Welcome to Mrinal's Chat Program: mickey
Server:No other users are active now.
Server: You are connected to minnie
minnie:how are you?
Server: Exiting Connection Closed.
```

```
michail@m:~/cn/cn/chat$ ./client 127.0.0.1 2222
Username: minnie
Wait from Server.

Server: Welcome to Mrinal's Chat Program: minnie
Server: Active Users are:
mickey

CONNECT:mickey
Server: You are connected to mickey
how are you?

You: how are you?
bye
You Closed Connection.
Client Program Ends...
michail@m:~/cn/cn/chat$
```

```
[0] 0:bash*                                                                    "m" 21:54 31-Aug-16
```

5. Exiting connection closed by Client1 and the other client stays in the application and a third client connects to Client2.

1. Client-1 and Client-2 are communicating.
2. Client-1 exits the application.
3. Client-2 is still connected to the server.
4. Client-3 connects to Client-2.



6. Multiple Clients logging.

1. Four clients are logged into the application.
2. Each are shown the list of available clients at the moment of their login (depending on the clients active at that time).

```
michail@m:~/cn/cn/chat$ gcc -pthread server.c -o server
michail@m:~/cn/cn/chat$ ./server 2222

Handling client 127.0.0.1 60780

Handling client 127.0.0.1 60781

Handling client 127.0.0.1 60782

Handling client 127.0.0.1 60783
```

```
michail@m:~/cn/cn/chat$ gcc -pthread client.c -o client
michail@m:~/cn/cn/chat$ ./client 127.0.0.1 2222
Username: mrinal
Wait from Server.

Server: Welcome to Mrinal's Chat Program: mrinal
Server:No other users are active now.
```

```
michail@m:~/cn/cn/chat$ ./client 127.0.0.1 2222
Username: aich
Wait from Server.

Server: Welcome to Mrinal's Chat Program: aich
Server: Active Users are:
mrinal
```

```
michail@m:~/cn/cn/chat$ ./client 127.0.0.1 2222
Username: minnie
Wait from Server.

Server: Welcome to Mrinal's Chat Program: minnie
Server: Active Users are:
mrinal
aich
mickey
```

```
michail@m:~/cn/cn/chat$ ./client 127.0.0.1 2222
Username: mickey
Wait from Server.

Server: Welcome to Mrinal's Chat Program: mickey
Server: Active Users are:
mrinal
aich
```

```
[0] 0:./client*                                                "m" 21:38 31-Aug-16
```

7. Multiple Clients logging and chatting in a one-to-one basis.

1. Four clients are logged into the application.
2. Two one-to-one chat sessions are created.
3. Client 'mickey' is connected to Client 'minnie'. Same with clients 'mrinal' and 'aich'.