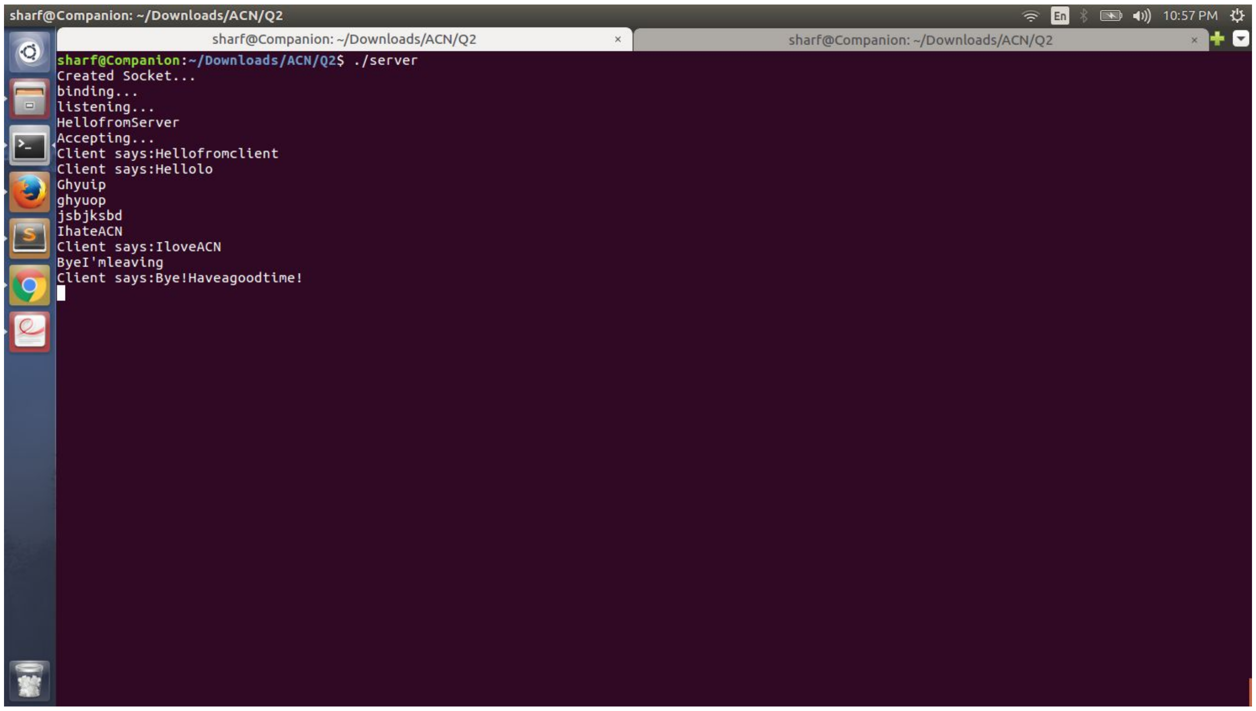
(Kausik Ponnapalli(201401236),Sharfuddin Mohammed(201401205))

Assignment-1

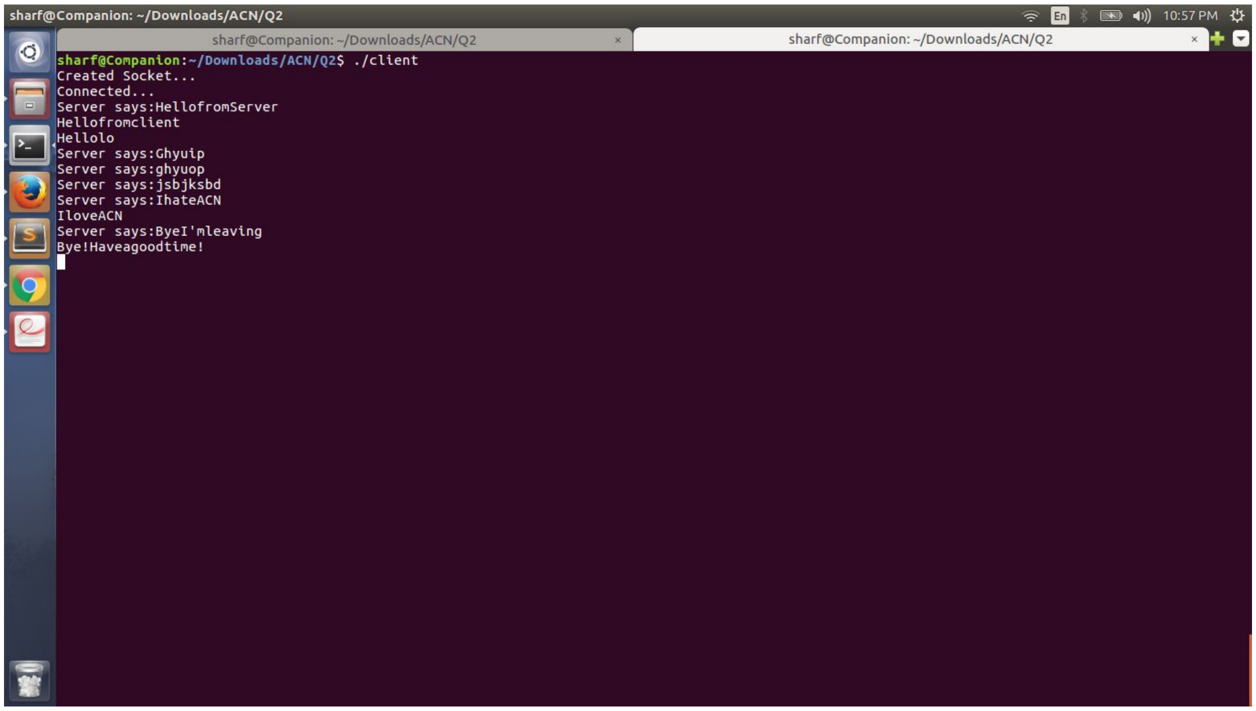
Q1

- We used the broadcast IP to broadcast the messages across all systems with the same broadcast IP address and in our case we tested it on IP broadcast address 10.4.1.25.
- **setsocketopt()** attaches a port number to the socket forcefully.
- **inet_pton()** attaches the socket with IP address.

Q2


sharf@Companion: ~/Downloads/ACN/Q2
sharf@Companion: ~/Downloads/ACN/Q2
sharf@Companion: ~/Downloads/ACN/Q2
10:57 PM

```
sharf@Companion:~/Downloads/ACN/Q2$ ./server
Created Socket...
binding...
listening...
HellofromServer
Accepting...
Client says:Hellofromclient
Client says:Hellolo
Ghyuip
ghyuop
jsbjksbd
IhateACN
Client says:IloveACN
ByeI'mleaving
Client says:Bye!Haveagoodtime!
```


sharf@Companion: ~/Downloads/ACN/Q2
10:57 PM
sharf@Companion: ~/Downloads/ACN/Q2
sharf@Companion: ~/Downloads/ACN/Q2

```
sharf@Companion:~/Downloads/ACN/Q2$ ./client
Created Socket...
Connected...
Server says:HellofromServer
Hellofromclient
Hellolo
Server says:Ghyuip
Server says:ghyuop
Server says:jsbjksbd
Server says:IhateACN
IloveACN
Server says:ByeI'mleaving
Bye!Haveagoodtime!
```

- We established a TCP connection between server and client using system calls.
- We use two threads that run parallely .One for writing and another for reading from the socket.
- We created a new socket file descriptor using **socket()** and it returns 0 when it creates a new socket file descriptor successfully.
- **Bind()** system call associates a socket file descriptor and with the IP address and the Port number of the socket.
- **listen()** waits for the request from a client in which multiple clients can connect to the server simultaneously.
- **accept()** establishes a connection between client and server and assigns a new file descriptor to the client thread.
- We implemented a loop for the reader and writer threads which are running concurrently.

Q3

- **listen()** waits for the request from a client in which multiple clients can connect   to the server simultaneously.
- The number of clients which can connect to the server is limited by the backlog parameter given in the listen() function.
- The client processes were run parallely and we used a thread that loops continuously to accommodate multiple clients.
- we used a new thread to accept a new client every time.