# COMPUTER NETWORKS

**ASSIGNMENT - 2 (Report)**

**Name: Suraj Kiran Mate**
**Entry No. : 2021JCS2387**

In this assignment I have created two python files **client.py** and **server.py**. These files represent the client side code and the server side code respectively. I will go into details of each code in this report below.

The rough idea about the way in which communication takes place in between client and server is as follows:

1. Server turned on and open the port for required connection.
2. Client turned on and sent the connection request at the same port and IP address.
3. Server got the connection request, server accepts the connection request and binds the port with client.
4. After successful connection, client sends the registration request to the server.
5. Server registers user for both sending and receiving.
6. All the clients follow the same procedure form 1 to 5 for successful registration.
7. After registration one user creates the message and sends it to other user.
8. The message first arrive at server, the server will check the correctness of the message format and forwards this message to the receiving user.

In this way message is transferred from sender to receiver in this communication setting.

So at the start of the communication first server creates the port for connection. Then both sender and receiver will connect to the port opened by the server. There can be multiple users other than sender and receiver. Our code also facilitate message broadcasting where the sender will use ALL keyword to broadcast the message to all the users who have registered with that server.

This communication is going to be stateful where once the user gets registered with the server then server is going to remember that user for further communication. Means the user don't need to register again and again before every communication.

We will first go into details of the client side code.

On client side once the client forms successful connection with server then the client will ask to input the message along with some specified format.
The format is generalised as : **@[recipient name]: [content of the message]**

This user put message is then sent for format checking where the message format is checked and several parameters are taken out of the message.

From this user input the message is created by the client side code in certain mentioned format. For creating this message the recipient name is extracted from this input message. Also the content of the message is extracted from this input message. The the server side code will create the message in below format:

**SEND [recipient username] Content-length: [length of message] [content of the message]**

Once after converting the message in this format the message is sent to server. Then before processing on this message server will check for the message format. If the message format is correct then this message is forwarded to the recipient. If any wrong format occurs then the server side will return an error to the client.
There are two functions in client code.
The **sendMsg** function is responsible for sending the messages to the server and the **recvMsg** function is responsible for receiving the messages from the server. From sender side the sendMsg function is active and sends the message in above format to the server. And on the receiver side the recvMsg function is active and receive the message forwarded from the server.

The server forwards the message to the receiver in the following format:

**FORWARD [username] Content-length: [length of message] [content of the message]**

In the recvMsg function the receiver will first check the format of the message is similar to the above standard format or not. If the format is same then the receiver code will extract the content out of this message and display it on the screen of the receiver.

There can be many error formats which can be invoked if the shared message format in between server and the client is not as expected.

These error formats are discussed below:

**ERROR 100 Malformed username : I**f the User wants to gets registered with the server then the user will snd the registration request to the server. If username in the registration request is not well formed then this error occurred.

**ERROR 101 No user registered :** If the client tries to communicate with the server without any prior registration then the server will invoke this error indicating that the client is not yet registered.

**ERROR 102 Unable to send** : If the server is unable to forward the message to the receiver then this error is invoked by the server.

**ERROR 103 Header incomplete :** If the message header is incomplete then this error is returned by the server.

These are all the error invoked by the server. All these error are invoked in response to some of the some of the unexpected input from the client.

**How to run this code:**

1. First run the server.py file.
2. Run client.py file and give input in format python3 client.py [username] [IP address].
3. Then the server will automatically register the client for both sending and receiving port.
4. After registering multiple users one of the user which will behave like sender will put the name of the receiver to send the message. The format is like:

   **@[username]: [content of the message]**

5. Once the message is created it will be sent by pressing the enter key. The message will reach to receivers terminal.

**Thank You** !