CS - 632

Assignment 1

Socket Server with Multiple Clients

- Create a Server script first so that the client communicates with it.
    - Create a socket connection using the **socket()** of the socket library and declare the host and port on which we need to communicate with clients.
    - **Bind** the host and port to the socket server we created above. So, if it binds successfully then it starts waiting for the client otherwise it just returns the error that occurred while establishing a connection.
    - We need to support handling multiple clients or threads at the same time simultaneously. So, for that, we need to create a function that handles requests from the individual client by a thread.
    - Define a new function named **threaded_client** which connects to each client at a different address given by the server.
    - Use the **recv()** function to get data from each client independently and then simply return the reply to the particular client with the same message with the string concatenate **"Server Says"** in the beginning.

- We want to run our Server all the time, which means we do not want to make our Server get stopped. Therefore, we need to use a while loop to make it run the Server endlessly until we manually stop the Server.
- Accept connections from the client using accept the () function of the socket server.
- It returns the type of client which has connected along with the unique thread number or address provided to it.
- After the **start_new_thread()** function of the thread class which creates or assigns a new thread to each client to handle them individually.

- Create a Client script:
  - We need the same socket library to establish a connection with the Server-Side and assign the same host and port number to the client as we defined in the Server otherwise it will not make the connection between them.
  - Set up a connection using **connect()** of the socket library which establishes a connection with the server using the host and port we provided.
  - We want is to make sure that the Client keeps running as the Server is Running.

- And we also going to provide an input option to the client so that it can send data back to the Server and along with this we also use the **recv()** function to receive data from Server Side.

- **Sample Output**: