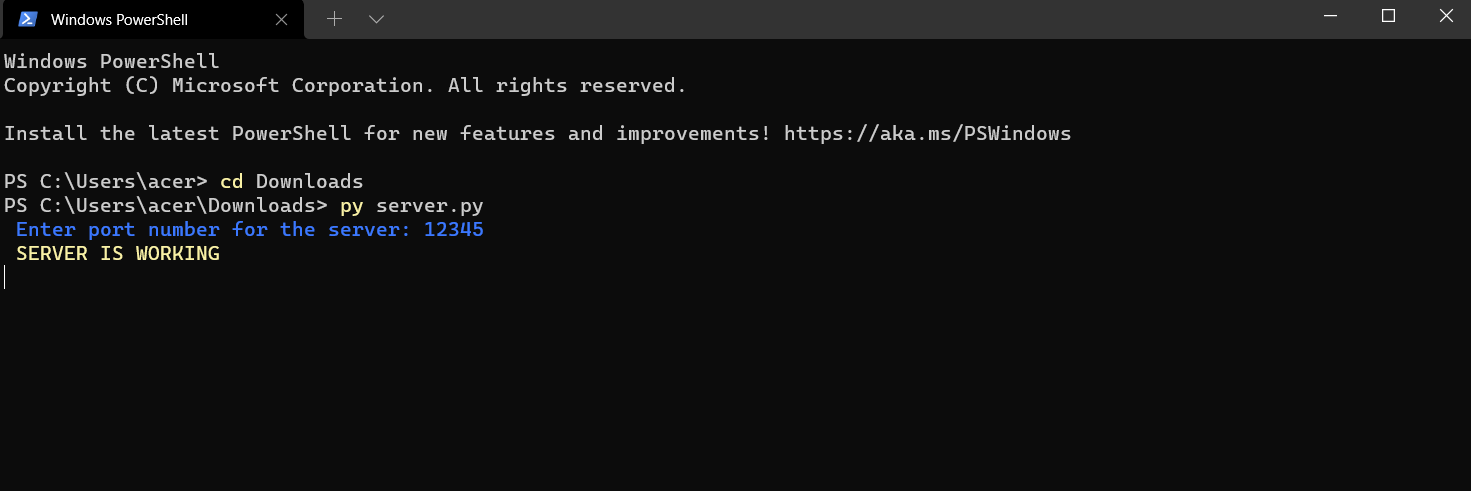
COL334 Assignment 2 Report

By Khyateeswar naidu Nalla

2019CS10376

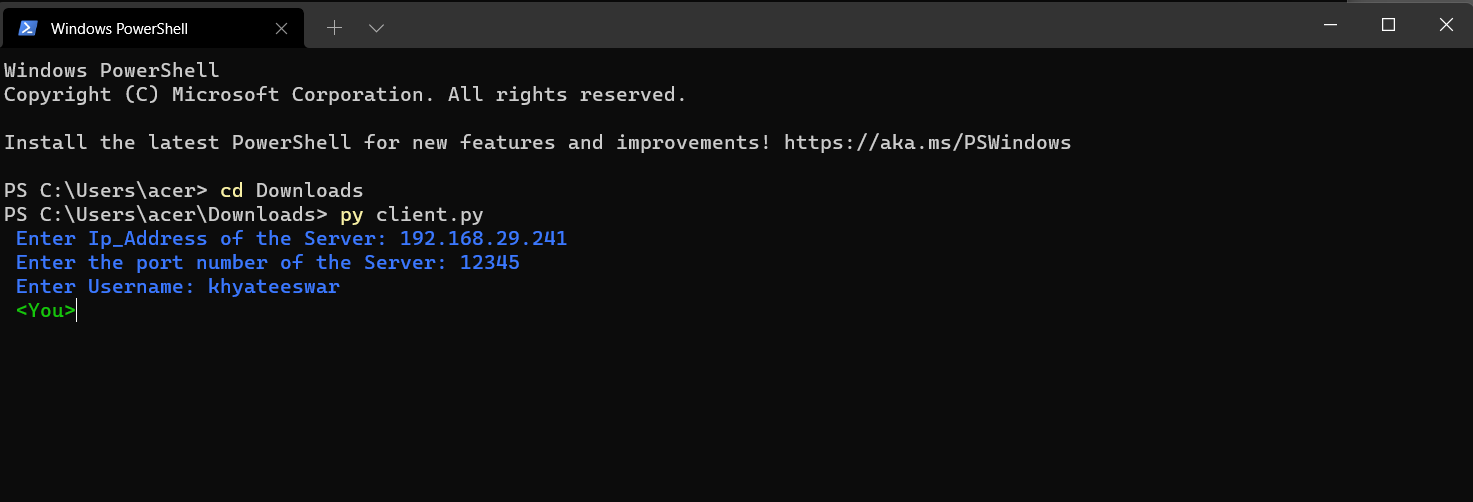
Initiation of the Server:

When we run the server program it asks for the port number to bind a socket to it which is the main socket used for initial communication with the clients. So input a port number > 2000 so that it doesn’t interfere with some default port numbers used by the operating system. Then the server starts listening to the new connections



First connection of client with the server:

When we run the client program it asks for the ip address and port number of the server so input them and then it forms a connection with the server. Then the server allocates two port numbers for the client , sends the two port numbers and then closes the connection. Also the server starts two threads one for receiving and the other for sending each with each socket created for that particular client and waits for clients to connect.



Client Registration:

Then the client application also starts two threads similar to the server and then takes a name as input and requests for registration to send and receive. If the username is unused and if it is a proper name the registration process gets completed and the client is ready to send or receive messages otherwise it throws an error and then asks us to input a proper username again and again till we enter one.

Text

Description automatically generated

Text

Description automatically generated

Sending and Receiving messages:

The sender thread in the client program waits for out input and then when we input it sends it to the server using the protocol given below and then the server checks if the message received has the protocol and if the recipient is available by checking if the recipient is present in the connections dictionary, it forwards the message to the recipient client receiving thread where it prints the message and sends back received acknowledgement to the server and the server also sends an acknowledgement that the message is delivered successfully. The client which sends the message waits till it receives the delivered acknowledgement.

**My Design modifications:**

I have slightly altered the protocol so that it suites my program

|  |  |
| --- | --- |
| Client to server to send a message | SEND <recipient name> <Content Length>  <message body> |
| Server to client for forwarding a message | FORWARD <recipient name> <Content Length> <message body> |
| Server to client if broadcasting | BROADCAST <recipient name> <Content Length>  message |
|  |  |

The remaining protocol is similar to what is mentioned in the assignment question. The first two rows if the above table is modified just to reduce the length of the http request. But the third row, when the message is to be broadcasted I thought I wouldn’t need a acknowledgement so that I changed it to the above so that I don’t need to wait for the client sender thread to wait till the acknowledgement arrives.

My server also doesn’t close the connection if the header is incomplete and returns the client with a error message and the client displays the error and asks the client type the message in format again. But actually it never happens in my code because my client accepts properly formatted messages and always follows the protocol.

I also add the feature to close the client connection by just typing “close” as input instead if the message. This closes the threads and the sockets corresponding to that client.

Text

Description automatically generated

I also added a feature which notifies each client when a new client is joined or if someone is left

Text

Description automatically generated

Text

Description automatically generated