

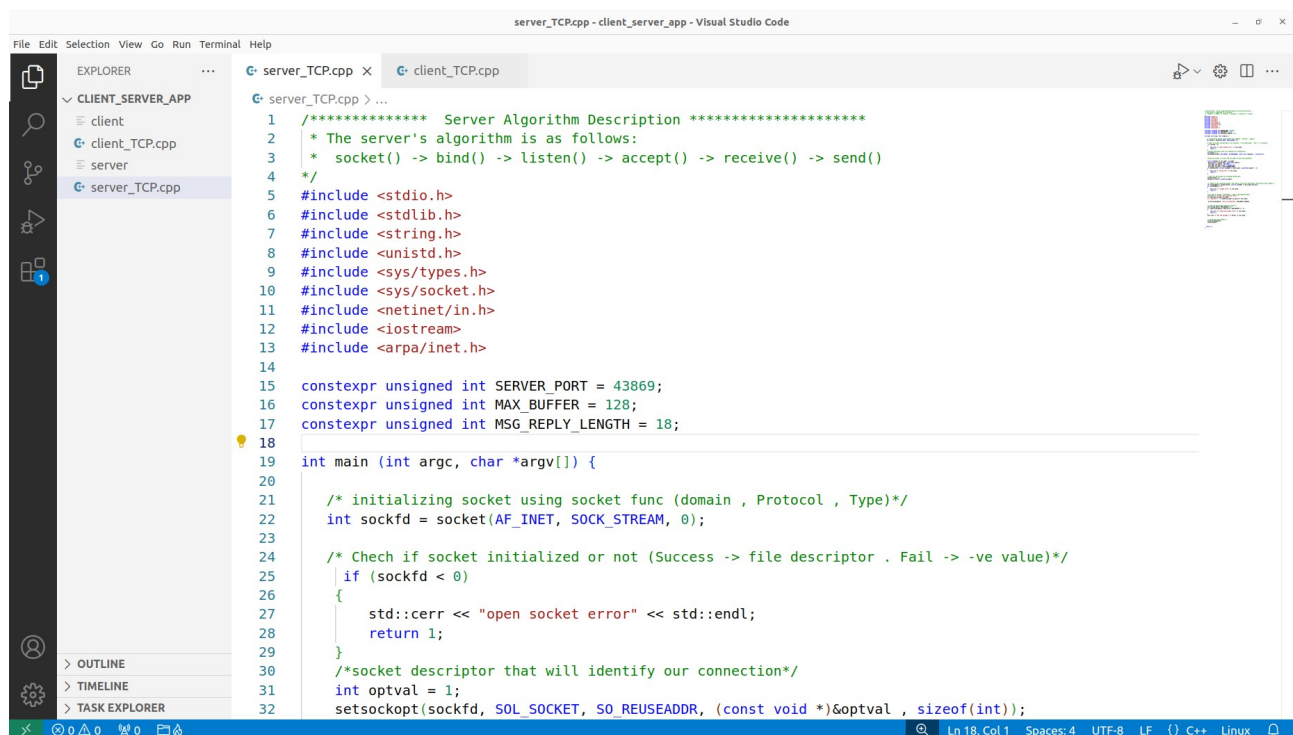
Client-Server Application

Server Test cases :

1- Verify that the server starts correctly and begins listening on the specified port

Steps:

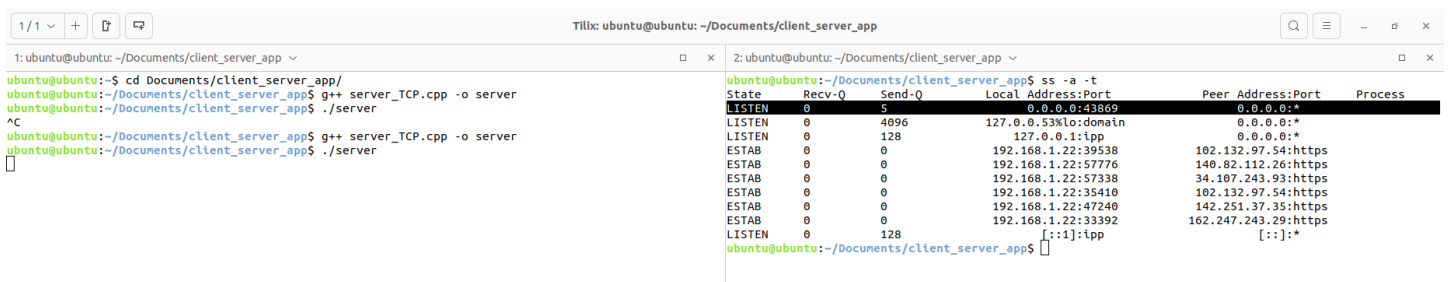
1. Configure the server with a specific port number.(43869)



```
server_TCP.cpp - client_server_app - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
CLIENT_SERVER_APP
  client
  client_TCP.cpp
  server
  server_TCP.cpp
server_TCP.cpp
1  /***** Server Algorithm Description *****/
2  * The server's algorithm is as follows:
3  * socket() -> bind() -> listen() -> accept() -> receive() -> send()
4  */
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <string.h>
8  #include <unistd.h>
9  #include <sys/types.h>
10 #include <sys/socket.h>
11 #include <netinet/in.h>
12 #include <iostream>
13 #include <arpa/inet.h>
14
15 constexpr unsigned int SERVER_PORT = 43869;
16 constexpr unsigned int MAX_BUFFER = 128;
17 constexpr unsigned int MSG_REPLY_LENGTH = 18;
18
19 int main (int argc, char *argv[]) {
20
21     /* initializing socket using socket func (domain , Protocol , Type)*/
22     int sockfd = socket(AF_INET, SOCK_STREAM, 0);
23
24     /* Check if socket initialized or not (Success -> file descriptor . Fail -> -ve value)*/
25     if (sockfd < 0)
26     {
27         std::cerr << "open socket error" << std::endl;
28         return 1;
29     }
30     /*socket descriptor that will identify our connection*/
31     int optval = 1;
32     setsockopt(sockfd, SOL_SOCKET, SO_REUSEADDR, (const void *)&optval, sizeof(int));
33 }
```

2. Start the server.

3. Check if the server is listening on the specified port (ss -a -t)



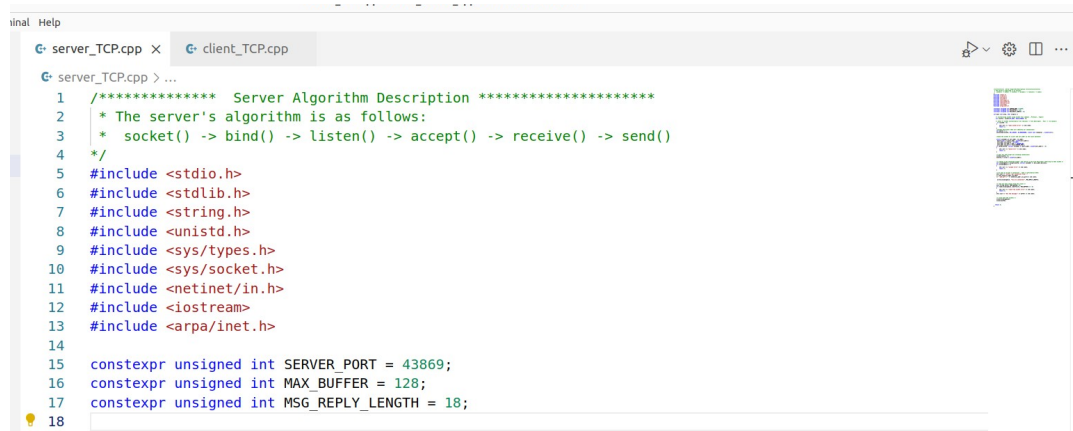
```
1: ubuntu@ubuntu:~/Documents/client_server_app
ubuntu@ubuntu:~/Documents/client_server_app$ cd Documents/client_server_app/
ubuntu@ubuntu:~/Documents/client_server_app$ g++ server_TCP.cpp -o server
ubuntu@ubuntu:~/Documents/client_server_app$ ./server
^C
ubuntu@ubuntu:~/Documents/client_server_app$ g++ server_TCP.cpp -o server
ubuntu@ubuntu:~/Documents/client_server_app$ ./server

2: ubuntu@ubuntu:~/Documents/client_server_app$ ss -a -t
State      Recv-Q    Send-Q    Local Address:Port    Peer Address:Port    Process
LISTEN     0          128      0.0.0.0:43869          0.0.0.0:*
LISTEN     0          4096     127.0.0.1:domain      0.0.0.0:*
LISTEN     0          128      127.0.0.1:ipp         0.0.0.0:*
ESTAB      0          0        192.168.1.22:39538     102.132.97.54:https
ESTAB      0          0        192.168.1.22:57776     140.82.112.26:https
ESTAB      0          0        192.168.1.22:57338     34.107.243.93:https
ESTAB      0          0        192.168.1.22:35410     102.132.97.54:https
ESTAB      0          0        192.168.1.22:47240     142.251.37.35:https
ESTAB      0          0        192.168.1.22:33392     162.247.243.29:https
LISTEN     0          128      :::1:ipp              ::::*
```

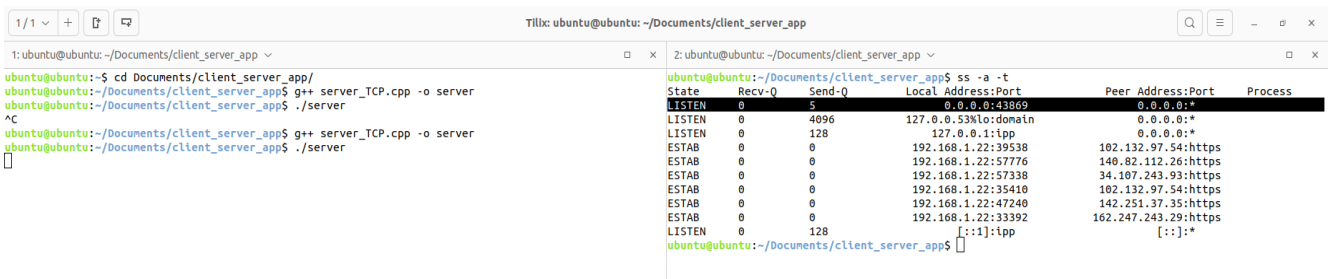
2- Ensure that the server correctly listens on different specified ports.

Steps :

1. Start the server with port (43869).
2. Verify the server is listening on port (43869).



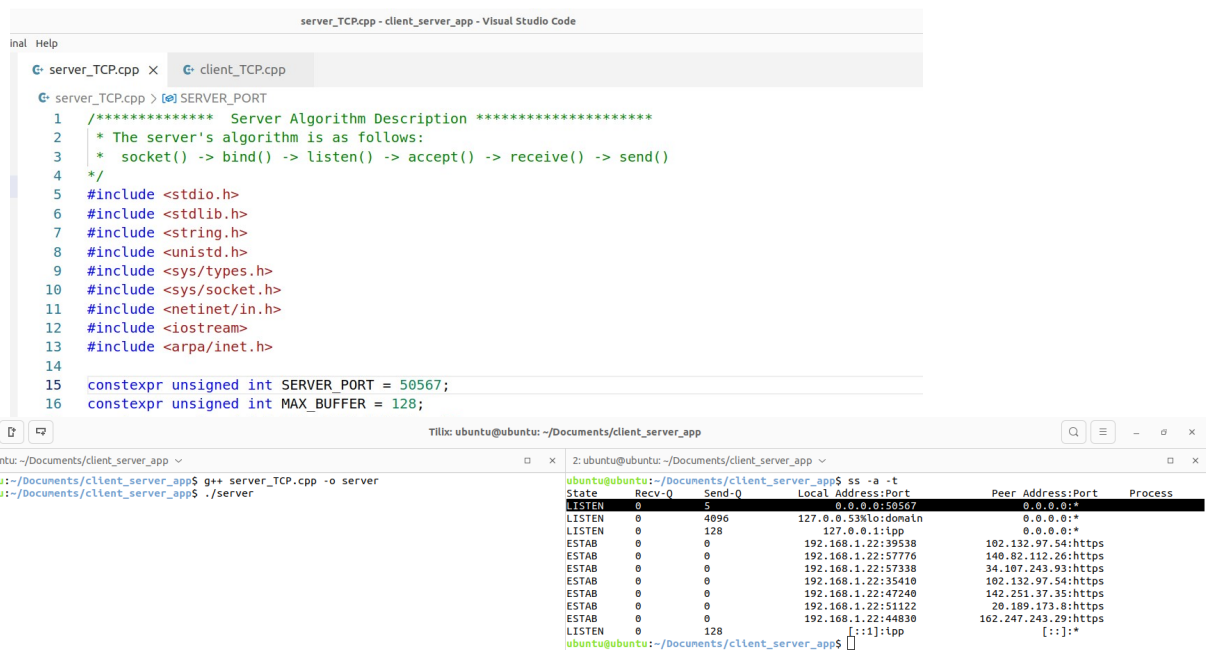
```
1  /***** Server Algorithm Description *****/
2  * The server's algorithm is as follows:
3  * socket() -> bind() -> listen() -> accept() -> receive() -> send()
4  */
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <string.h>
8  #include <unistd.h>
9  #include <sys/types.h>
10 #include <sys/socket.h>
11 #include <netinet/in.h>
12 #include <iostream>
13 #include <arpa/inet.h>
14
15 constexpr unsigned int SERVER_PORT = 43869;
16 constexpr unsigned int MAX_BUFFER = 128;
17 constexpr unsigned int MSG_REPLY_LENGTH = 18;
18
```



```
1: ubuntu@ubuntu:~/Documents/client_server_app
ubuntu@ubuntu:~/Documents/client_server_app$ cd Documents/client_server_app/
ubuntu@ubuntu:~/Documents/client_server_app$ g++ server_TCP.cpp -o server
ubuntu@ubuntu:~/Documents/client_server_app$ ./server
^C
ubuntu@ubuntu:~/Documents/client_server_app$ g++ server_TCP.cpp -o server
ubuntu@ubuntu:~/Documents/client_server_app$ ./server

2: ubuntu@ubuntu:~/Documents/client_server_app$ ss -a -t
State      Recv-Q    Send-Q    Local Address:Port    Peer Address:Port    Process
LISTEN     0         128      127.0.0.1:43869       0.0.0.0:*
LISTEN     0         128      127.0.0.1:tcp        0.0.0.0:*
ESTAB      0         0        192.168.1.22:39538    102.132.97.54:https
ESTAB      0         0        192.168.1.22:57776    140.82.112.26:https
ESTAB      0         0        192.168.1.22:57338    34.107.243.93:https
ESTAB      0         0        192.168.1.22:35410    102.132.97.54:https
ESTAB      0         0        192.168.1.22:47240    142.251.37.35:https
ESTAB      0         0        192.168.1.22:33392    162.247.243.29:https
LISTEN     0         128      [::]:tcp             [::]:*
```

3. Stop the server and restart it with port (50567)
4. Verify the server is listening on port (50567)



```
server_TCP.cpp - client_server_app - Visual Studio Code

1  /***** Server Algorithm Description *****/
2  * The server's algorithm is as follows:
3  * socket() -> bind() -> listen() -> accept() -> receive() -> send()
4  */
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <string.h>
8  #include <unistd.h>
9  #include <sys/types.h>
10 #include <sys/socket.h>
11 #include <netinet/in.h>
12 #include <iostream>
13 #include <arpa/inet.h>
14
15 constexpr unsigned int SERVER_PORT = 50567;
16 constexpr unsigned int MAX_BUFFER = 128;

1: ubuntu@ubuntu:~/Documents/client_server_app
ubuntu@ubuntu:~/Documents/client_server_app$ g++ server_TCP.cpp -o server
ubuntu@ubuntu:~/Documents/client_server_app$ ./server

2: ubuntu@ubuntu:~/Documents/client_server_app$ ss -a -t
State      Recv-Q    Send-Q    Local Address:Port    Peer Address:Port    Process
LISTEN     0         128      127.0.0.1:50567       0.0.0.0:*
LISTEN     0         128      127.0.0.1:tcp        0.0.0.0:*
ESTAB      0         0        192.168.1.22:39538    102.132.97.54:https
ESTAB      0         0        192.168.1.22:57776    140.82.112.26:https
ESTAB      0         0        192.168.1.22:57338    34.107.243.93:https
ESTAB      0         0        192.168.1.22:35410    102.132.97.54:https
ESTAB      0         0        192.168.1.22:47240    142.251.37.35:https
ESTAB      0         0        192.168.1.22:51122    20.189.173.8:https
ESTAB      0         0        192.168.1.22:44830    162.247.243.29:https
LISTEN     0         128      [::]:tcp             [::]:*
```

3- Confirm that the server accepts incoming connections.

Steps :

1. Start the server.
2. Use a client to attempt a connection to the server.
3. Check the server logs or use a monitoring tool to verify the connection was accepted. (netstat -ntp)

```
1/1 v + [ ] [ ]
Tilix: ubuntu@ubuntu: ~
1: ubuntu@ubuntu: ~/Documents/client_server_app v
2: ubuntu@ubuntu: ~/Documents/client_server_app v
3: ubuntu@ubuntu: ~ v
```

```
ubuntu@ubuntu:~$ cd Documents/client_server_app/
ubuntu@ubuntu:~/Documents/client_server_app$ g++ server_TCP.cpp -o server
ubuntu@ubuntu:~/Documents/client_server_app$ ./server
server: got connection from = 127.0.0.1 and port = 55418

```

```
ubuntu@ubuntu:~/Documents/client_server_app$ g++ client_TCP.cpp -o client
ubuntu@ubuntu:~/Documents/client_server_app$ ./client 127.0.0.1
You are connected!
What message for the server? : 
```

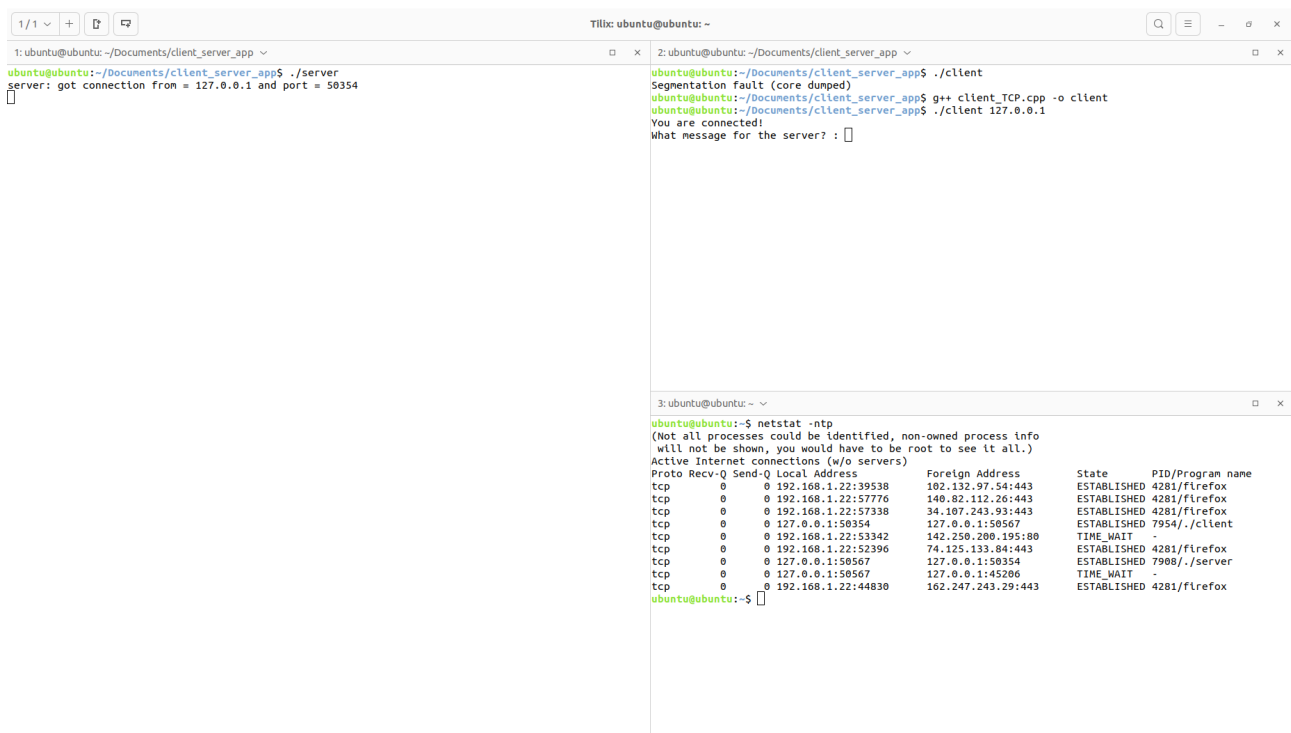
```
ubuntu@ubuntu:~$ netstat -ntp
(Not all processes could be identified, non-owned process info
will not be shown, you would have to be root to see it all.)
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 192.168.1.22:39538     102.132.97.54:443      ESTABLISHED 4281/firefox
tcp        0      0 127.0.0.1:50507       127.0.0.1:55418        ESTABLISHED 8782/./server
tcp        0      0 127.0.0.1:55418       127.0.0.1:50507        ESTABLISHED 8796/./client
tcp        0      0 192.168.1.22:57338    34.107.243.93:443      ESTABLISHED 4281/firefox
tcp        0      0 192.168.1.22:36912    102.247.241.14:443     ESTABLISHED 4281/firefox
tcp        0      0 192.168.1.22:49268    152.199.19.160:443     ESTABLISHED 6672/Code --standar
tcp        0      0 192.168.1.22:41274    142.251.37.35:443      ESTABLISHED 4281/firefox
tcp        0      0 192.168.1.22:35140    140.82.112.26:443      ESTABLISHED 4281/firefox
ubuntu@ubuntu:~$
```

Client Test cases :

1- Ensure the client starts correctly and attempts to connect to the server.

Steps:

1. Start the client with a valid server IP and port number.
2. Check if the client sends a connection request to the server.
(netstat -ntp)



```
1: ubuntu@ubuntu:~/Documents/client_server_app$ ./server
server: got connection from = 127.0.0.1 and port = 50354

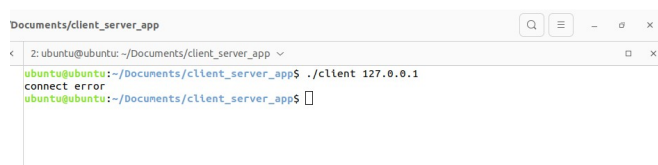
2: ubuntu@ubuntu:~/Documents/client_server_app$ ./client
Segmentation Fault (core dumped)
ubuntu@ubuntu:~/Documents/client_server_app$ g++ client_TCP.cpp -o client
ubuntu@ubuntu:~/Documents/client_server_app$ ./client 127.0.0.1
You are connected!
What message for the server? :

3: ubuntu@ubuntu:~/Documents/client_server_app$ netstat -ntp
(Not all processes could be identified, non-owned process info
will not be shown, you would have to be root to see it all.)
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 192.168.1.22:39538      102.132.97.54:443      ESTABLISHED 4281/firefox
tcp        0      0 192.168.1.22:57776     140.82.112.26:443      ESTABLISHED 4281/firefox
tcp        0      0 192.168.1.22:57338     34.107.243.93:443      ESTABLISHED 4281/firefox
tcp        0      0 127.0.0.1:50354        127.0.0.1:50567        ESTABLISHED 7954/./client
tcp        0      0 192.168.1.22:53342     142.250.200.195:80      TIME_WAIT   -
tcp        0      0 192.168.1.22:52396     74.125.133.84:443      ESTABLISHED 4281/firefox
tcp        0      0 127.0.0.1:50567        127.0.0.1:50354        ESTABLISHED 7908/./server
tcp        0      0 127.0.0.1:50567        127.0.0.1:45206        TIME_WAIT   -
tcp        0      0 192.168.1.22:44830     162.247.243.29:443     ESTABLISHED 4281/firefox
ubuntu@ubuntu:~/Documents/client_server_app$
```

2- Ensure the client correctly handles cases where the server is unavailable or refuses the connection.

Steps:

1. Start the client with a server IP and port where no server is running.
2. Observe how the client reacts to the failed connection attempt.



```
Documents/client_server_app
2: ubuntu@ubuntu:~/Documents/client_server_app$ ./client 127.0.0.1
connect error
ubuntu@ubuntu:~/Documents/client_server_app$
```

Running State of Application :

```
Tillx: ubuntu@ubuntu: ~/Documents/client_server_app
1: ubuntu@ubuntu: ~/Documents/client_server_app
2: ubuntu@ubuntu: ~/Documents/client_server_app
3: ubuntu@ubuntu: ~

ubuntu@ubuntu:~/Documents/client_server_app$ ./server
Server: got connection from = 127.0.0.1 and port = 48602
Got the message:HI
ubuntu@ubuntu:~/Documents/client_server_app$

ubuntu@ubuntu:~/Documents/client_server_app$ telnet 127.0.0.1 50567
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
You are connected!HI
Connection closed by foreign host.
ubuntu@ubuntu:~/Documents/client_server_app$

ubuntu@ubuntu:~$ netstat -ntp
(Not all processes could be identified, non-owned process info
will not be shown, you would have to be root to see it all.)
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 192.168.1.22:39538     102.132.97.54:443      ESTABLISHED 4281/firefox
tcp        0      0 127.0.0.1:55418       127.0.0.1:50567        TIME_WAIT   -
tcp        0      0 192.168.1.22:57338     34.107.243.93:443      ESTABLISHED 4281/firefox
tcp        0      0 192.168.1.22:50924     140.82.114.25:443      ESTABLISHED 4281/firefox
tcp        0      0 192.168.1.22:36912     162.247.241.14:443     ESTABLISHED 4281/firefox
tcp        0      0 127.0.0.1:50567        127.0.0.1:40602        ESTABLISHED 8871/./server
tcp        0      0 192.168.1.22:59638     74.125.133.84:443      TIME_WAIT   -
tcp        0      0 192.168.1.22:48080     140.82.121.3:443       ESTABLISHED 4281/firefox
tcp        0      0 127.0.0.1:40602        127.0.0.1:50567        ESTABLISHED 8872/telnet
tcp        0      0 192.168.1.22:41274     142.251.37.35:443      ESTABLISHED 4281/firefox
ubuntu@ubuntu:~$
```