Using TCP socket, implement HTTP server and client

Creating HTTP SERVER

1. Importing Socket Library

```
import socket
from socket import AF_INET, SOCK_STREAM, SO_REUSEADDR, SOL_SOCKET
#SO_REUSEADDR is to reuse the socket
```

2. Create HTTP Server with response message in http format

```
HOST, PORT = 'localhost', 8081
response = b"HTTP/1.1 200 OK \n\nhello buddy welcome to server"
```

3. Create TCP Server Socket

```
server_socket = socket.socket(AF_INET, SOCK_STREAM)
#setsockopt makes socket to reuse  address to readd user
server_socket.setsockopt(SOL_SOCKET, SO_REUSEADDR, 1)
# Binding the TCP server
server_socket.bind((HOST, PORT))
# Serever can listen to max of 1 Client
server_socket.listen(1)
```

4. Server acts like a HTTP Server

HTTP is a stateless protocol

Tcp is a connection oriented protocol

```
#here we have connect many times to the server
#because server closes connection with client after responding to client
while True:
    try:
        client_socket, addr = server_socket.accept()
        print(client_socket.recv(1024).decode('utf-8'))
        client_socket.sendall(response)
        client_socket.close()
    except Exception as e:
```

```
        print(e)
socketserver.close()
```

Creating HTTP client

1. Creating client socket to connect to the server

```
client_socket = socket.socket(AF_INET, SOCK_STREAM)
HOST, PORT = 'localhost', 8081 #connecting to host on port
```

2. Requesting server to connect using Http request

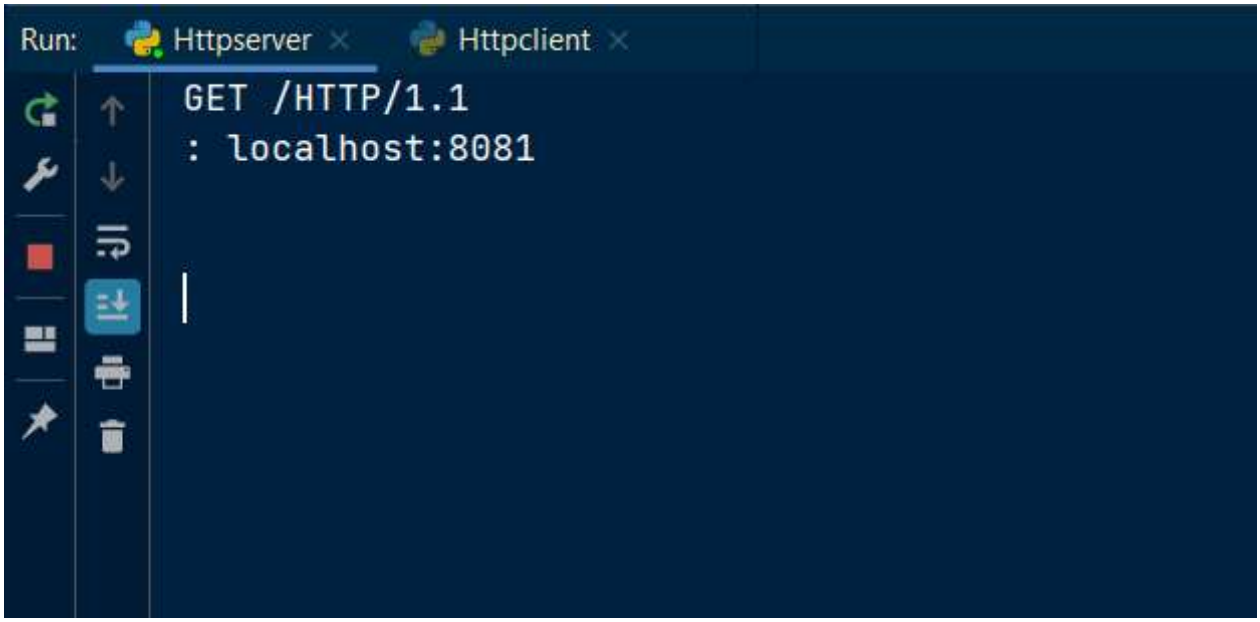
```
request = f"GET /HTTP/1.1\r\n: {HOST}:{PORT}\r\n\r\n".encode('utf8')
client_socket.setsockopt(SOL_SOCKET, SO_REUSEADDR, 1)
client_socket.connect((HOST, PORT))
client_socket.sendall(request)
```

3. After getting the server response printing

```
response = ""
while True:
    data = client_socket.recv(1024)
    if data == b'':
        break
    print(data.decode())
client_socket.close()
```

Output(Screen shots)

1. Server output



2. Client output