Assignment 2 – Socket Programming Assignment - Webserver

- Assignment Includes both the optional exercises
- Following server provides a multithreaded environment for TCP requests and provides responses without blocking the port.
- It listens for requests in port 81
- It also serves requests from a client program and a web based client program(browser)
- Client program can resolve host address by host name
- IP of the web server is 127.0.0.1 and host name is "localhost"

1. sock_server.py

```
import socket
from threading import Thread

notFoundResponse = """HTTP/1.1 404 Not Found\r\n Content-type: text/html\r\n \n
<html>\r\n
<body>\r\n
 <h1>404 Not Found</h1>\r\n
 <p>The requested URL was not found on this server.</p>\r\n
</body>\r\n
</html>\r\n"""

foundResponse = """HTTP/1.1 200 OK\r\n Content-type: text/html\r\n
"""

# Once Server accepts connection it post's to worker thread to
# provide response of HTML File.

def worker(connectionSocket,addr):
   print('Received'+str(addr))
   try:
      message = connectionSocket.recv(1024)
      filename = str(message,"utf-8").split()[1]
      f = open(filename[1:])
      content = f.read()
      outputdata = bytearray()
      outputdata.extend(map(ord, content))

      #Send one HTTP header line into socket
      connectionSocket.send(foundResponse.encode('utf-8'))

      #Send the content of the requested file to the client
      connectionSocket.send(outputdata)

      #Close client socket
      connectionSocket.close()
      f.close()

   except (IOError,IndexError):
      #Send response message for file not found
```

```
        connectionSocket.send(notFoundResponse.encode('utf-8'))
        #Close client socket
        connectionSocket.close()


def serverMain():
    serverSocket = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
    serverSocket.bind(('localhost',81))
    print('Main Server Started !')
    while True:
        serverSocket.listen(5)
        print('Ready to serve ..')
        connectionSocket, addr = serverSocket.accept()

        #Post it to worker thread to unblock this thread
        w = Thread(target=worker, args=(connectionSocket,addr))
        w.start()
    serverSocket.close()

if __name__ == '__main__':
    serverMain()
```

Following client can send a HTTP request to the web server at port 81 and print response header and content

2. sock_client.py

```
import sys
import socket
from sys import argv

#Supports only IPv4 for now
def getHost(hostAddr):
    return socket.gethostbyname(hostAddr)

def clientMain(argv):
    print('Entered Client process !')
    if(len(argv) == 4):
        print('server host : '+argv[1] + ' server port : '+argv[2] + ' filename = '+argv[3])
        hostAddr = getHost(argv[1])
        hostPort = argv[2]
        file = argv[3]
        print('Host is = '+hostAddr)
        clientSock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        clientSock.connect((hostAddr, int(hostPort)))
        reqString = 'GET '+file +' HTTP/1.1\r\nHost:' + hostAddr +":"+hostPort+'\r\nConnection: keep-
alive\r\n \r\n\r\n'
        clientSock.sendall(str.encode(reqString))
        data = clientSock.recv(1024)
        print(str(data))
        clientSock.close()
```

```
    else:
        print('Wrong Usage! , please input format sock_client.py [server_host] [server_port] [filename]')

if __name__ == '__main__':
    clientMain(argv)
```

3. **HelloWorld.html**

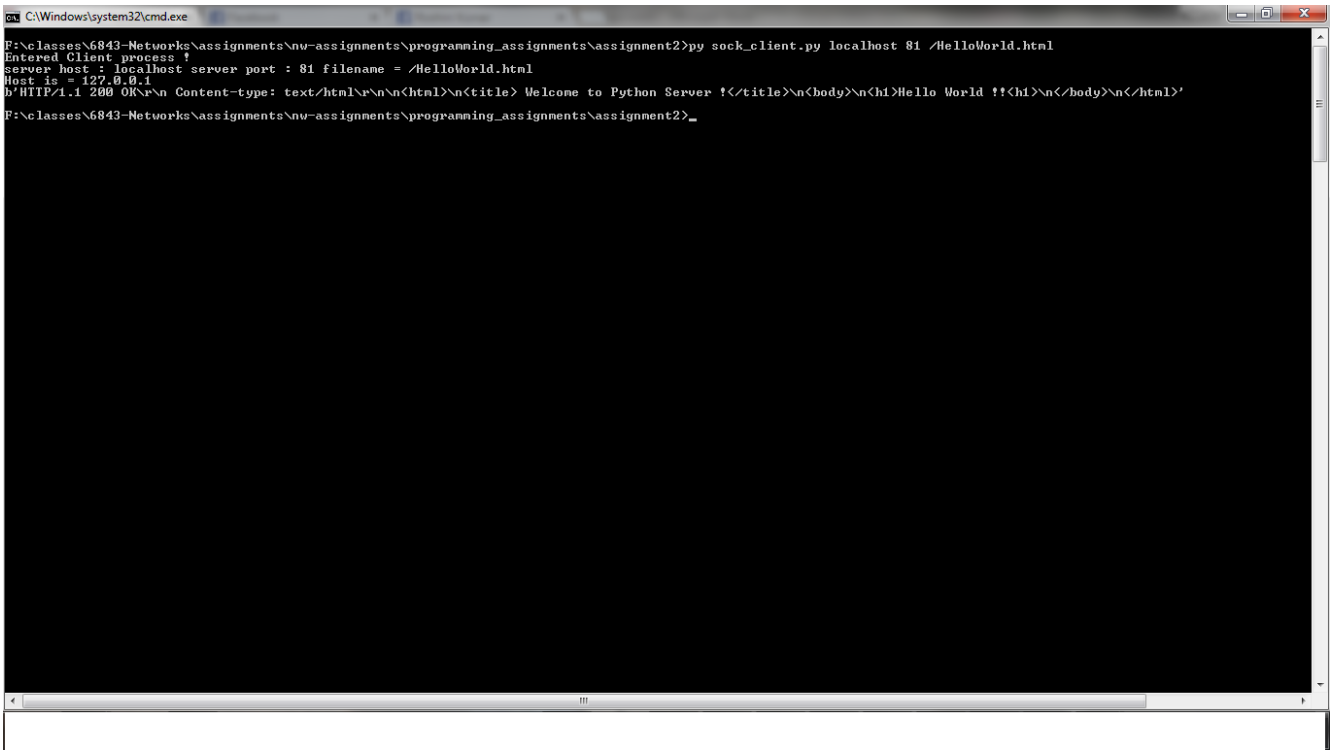
```
<html>
<title> Welcome to Python Server !</title>
<body>
<h1>Hello World !!<h1>
</body>
</html>
```

Screen Shots:

1. Welcome page shown on successful HTTP Request for HelloWorld.html

2. 404 Not Found shown for Invalid page HelloWorld1.html

3. Command Line prints successful HTTP Response field and Content



4. Command Line prints 404 Not found response message and Content