Gebze Technical University

Computer Engineering

Mobile Communications Networks

CSE 476

Term Project Report

Değer MANDAL

161044096

"Coded in PyCharm IDE"

Assignment 1: Web Server

Your Web server will =>

(i) create a connection socket when contacted by a client (browser);

```
#Fill in start
#Connect the TCP socket to the specified port ---
serverSocket.bind(('', 6789))
#Maximum number of connections is 1 ---
serverSocket.listen(1)
#Fill in end
```

(ii) receive the HTTP request from this connection;

```
#After the client receives the connection request, create a
new TCP connection socket ---
connectionSocket, addr = serverSocket.accept()#Fill in start
#Fill in end
```

(iii) parse the request to determine the specific file being requested;
(iv) get the requested file from the server's file system;

```
#Receive the message sent by the client ---
message = connectionSocket.recv(1024)#Fill in start #Fill in
end
print("Message:", message)
filename = message.split()[1]
print("FileName:", filename)
f = open(filename[1:])
outputdata = f.read()#Fill in start #Fill in end
```

(v) create an HTTP response message consisting of the requested file preceded by header lines;
(vi) send the response over the TCP connection to the requesting browser.

```
# Fill in start
header = ' HTTP/1.1 200 OK\nConnection: close\nContent-Type:
text/html\nContent-Length: %d\n\n' % (len(outputdata))
connectionSocket.send(header.encode())
# Fill in end
```

If a browser requests a file that is not present in your server, your server should return a "404 Not Found" error message.

```
# Fill in start
connectionSocket.send(('404 Not Found').encode())
# Fill in end
```

Close client socket;

```
# Close client socket
# Fill in start
connectionSocket.close()
# Fill in end
```

I taken an error so I changed the code;

```
# Send the content of the requested file to the client
for i in range(0, len(outputdata)):
    connectionSocket.send(outputdata[i].encode())
    #I changed the below code with the above code because it
gived to me error
    #connectionSocket.send(outputdata[i])
```

ERROR:

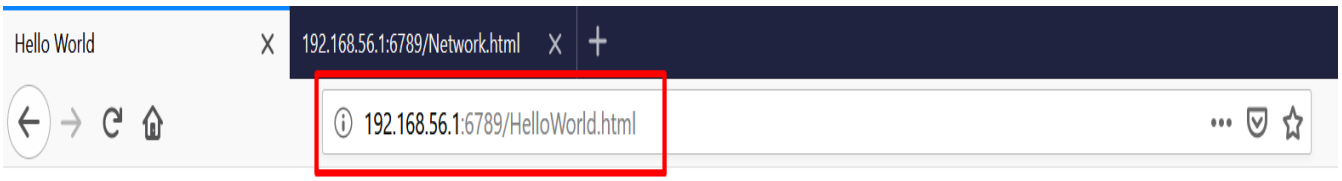
```
C:\Users\Deger\AppData\Local\Programs\Python\Python38-32
Ready to serve...
Traceback (most recent call last):
  File "C:\Users\Deger\Desktop\161044096\Web Server\Webs
Message: b'GET /HelloWorld.html HTTP/1.1\r\nHost: localh
    connectionSocket.send(outputdata[i])
TypeError: a bytes-like object is required, not 'str'
FileName: b'/HelloWorld.html'

Process finished with exit code 1
```

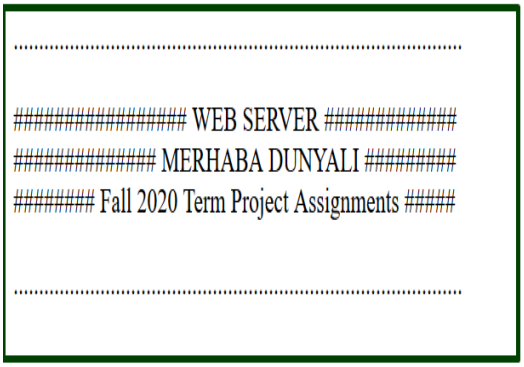
Outputs:

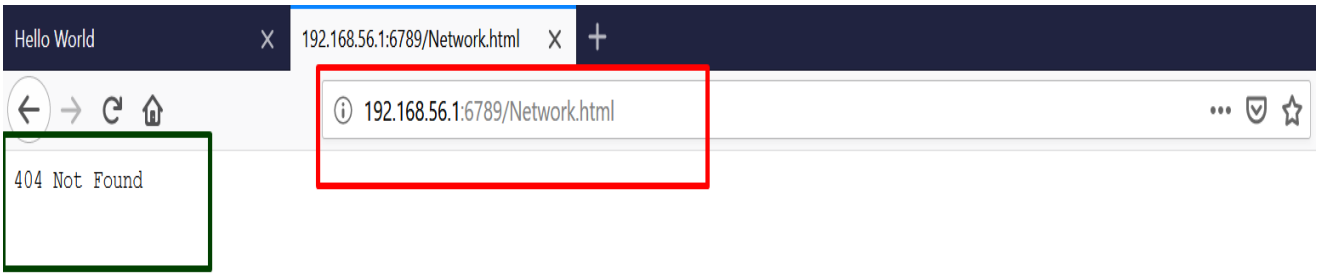
I tried two error code because of Google Chrome and Firefox giving different results.

I used the error code for FIREFOX:

```
#Fill in start
#connectionSocket.send(('HTTP/1.1 404 Not Found ').encode())
connectionSocket.send(('404 Not Found').encode())
# Fill in end
```

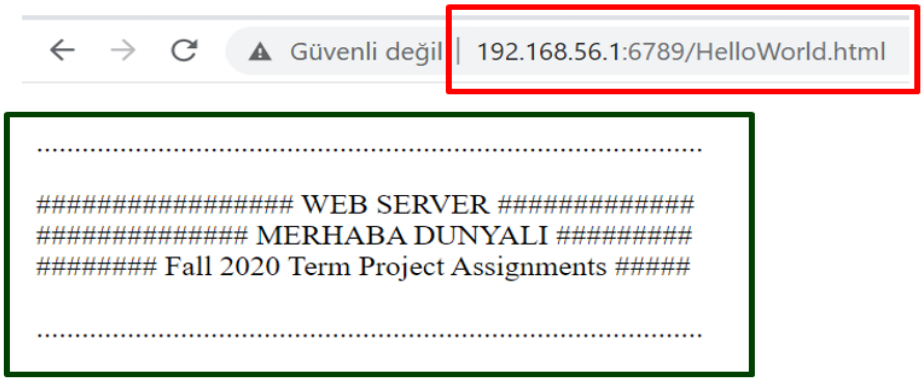


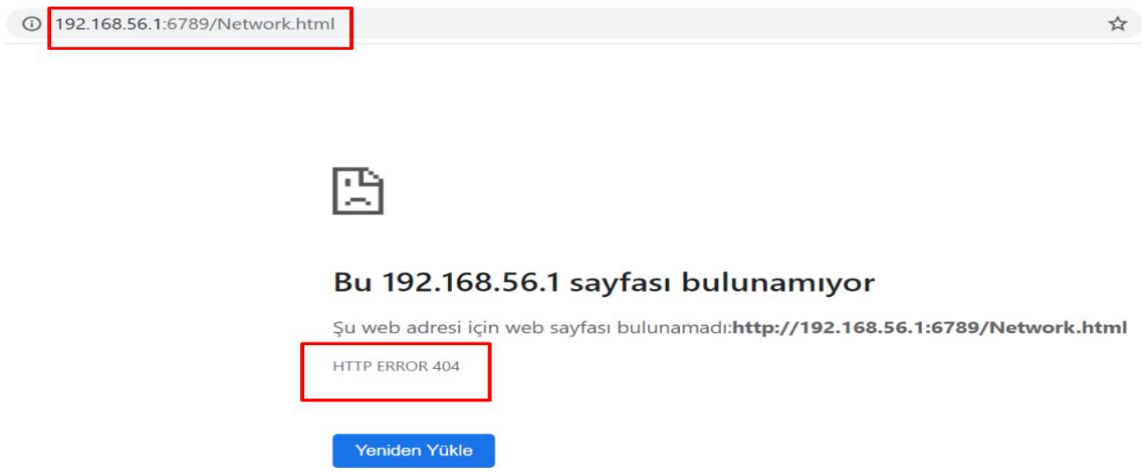




I used the error code for GOOGLE CHROME:

```
# Fill in start
connectionSocket.send(('HTTP/1.1 404 Not Found ').encode())
#connectionSocket.send(('404 Not Found').encode())
# Fill in end
```





Assignment 2: UDP Pinger

```
from socket import *
import time
```

During development, you should run the UDPPingerServer.py on your machine, and test your client by sending packets to localhost (or, 127.0.0.1);

```
#Server address, localhost is used.
serverName = '127.0.0.1'
#Port specified by the server
serverPort = 12000
```

(1) send the ping message using UDP (Note: Unlike TCP, you do not need to establish a connection first, since UDP is a connectionless protocol.)

```
#Create UDP socket, use IPv4 protocol
clientSocket = socket(AF_INET, SOCK_DGRAM)
#Set the socket timeout value to 1 second
clientSocket.settimeout(1)
```

The client message is one line, consisting of ASCII characters in the following format:
Ping sequence_number time
where sequence_number starts at 1 and progresses to 10 for each successive ping message sent by the client, and time is the time when the client sends the message;

```
for i in range(0, 10):
    sendTime = time.time()
```

(2) print the response message from server, if any;

```
    #Generate datagrams, encode bytes to send
    message = ('Ping %d %s' % (i + 1, sendTime)).encode()
    try:
        #Send information to the server
        clientSocket.sendto(message, (serverName, serverPort))
        #Get information from the server, also get the server
address
        modifiedMessage, serverAddress = clientSocket.recvfrom(1024)
```

(3) calculate and print the round trip time (RTT), in seconds, of each packet, if server responses;

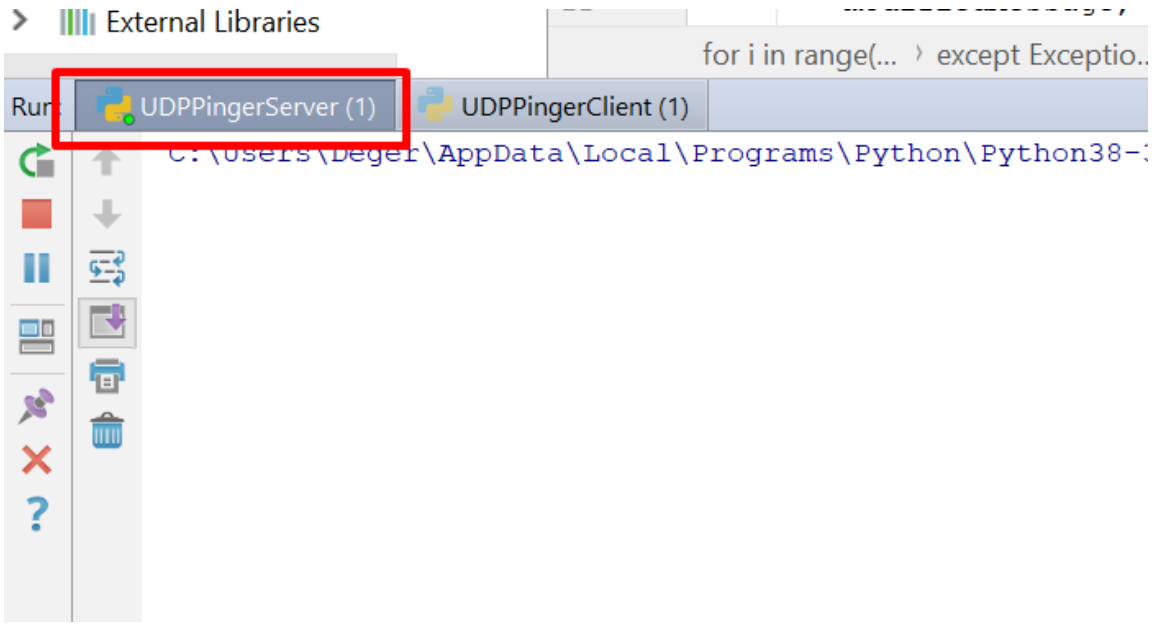
```
        #Calculate round trip time
        rtt = time.time() - sendTime

        print('Sequence %d: Reply from %s RTT = %.4fs' % (i + 1,
serverName, rtt))
    except Exception as e:
        #print(e)
```

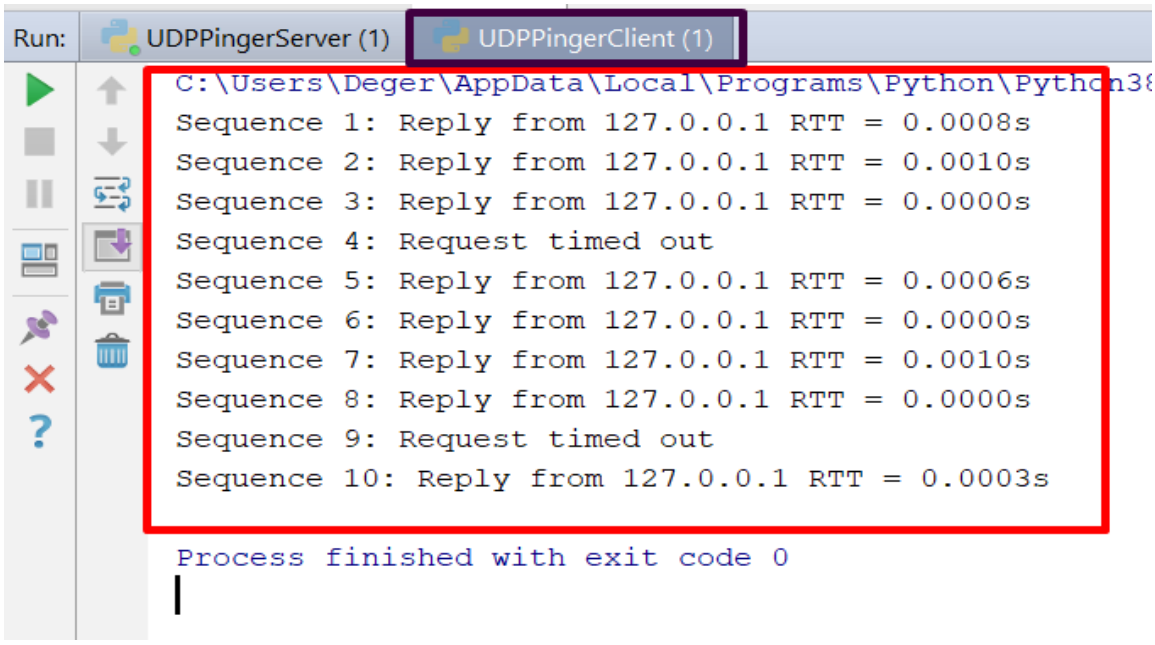
(4) otherwise, print “Request timed out”;

```
        print('Sequence %d: Request timed out' % (i + 1))
```

```
#close the socket
clientSocket.close()
```

Outputs:





for i in range(...

Run: UDPPingerServer (1) UDPPingerClient (1)

```
C:\Users\Deger\AppData\Local\Programs\Python\Python38
Sequence 1: Request timed out
Sequence 2: Reply from 127.0.0.1 RTT = 0.0008s
Sequence 3: Request timed out
Sequence 4: Request timed out
Sequence 5: Request timed out
Sequence 6: Request timed out
Sequence 7: Reply from 127.0.0.1 RTT = 0.0000s
Sequence 8: Reply from 127.0.0.1 RTT = 0.0012s
Sequence 9: Request timed out
Sequence 10: Request timed out

Process finished with exit code 0
```

Run: UDPPingerServer (1) UDPPingerClient (1)

```
C:\Users\Deger\AppData\Local\Programs\Python\Pyt
Sequence 1: Reply from 127.0.0.1 RTT = 0.0010s
Sequence 2: Reply from 127.0.0.1 RTT = 0.0000s
Sequence 3: Reply from 127.0.0.1 RTT = 0.0010s
Sequence 4: Reply from 127.0.0.1 RTT = 0.0006s
Sequence 5: Reply from 127.0.0.1 RTT = 0.0004s
Sequence 6: Reply from 127.0.0.1 RTT = 0.0000s
Sequence 7: Reply from 127.0.0.1 RTT = 0.0000s
Sequence 8: Request timed out
Sequence 9: Request timed out
Sequence 10: Request timed out
```

Run: UDPPingerServer (1) UDPPingerClient (1)

```
C:\Users\Deger\AppData\Local\Programs\Python\Python38-3
Sequence 1: Request timed out
Sequence 2: Reply from 192.168.80.1 RTT = 0.0007s
Sequence 3: Reply from 192.168.80.1 RTT = 0.0000s
Sequence 4: Request timed out
Sequence 5: Reply from 192.168.80.1 RTT = 0.0010s
Sequence 6: Reply from 192.168.80.1 RTT = 0.0004s
Sequence 7: Request timed out
Sequence 8: Request timed out
Sequence 9: Reply from 192.168.80.1 RTT = 0.0008s
Sequence 10: Request timed out

Process finished with exit code 0
```

Assignment 3: Mail Client

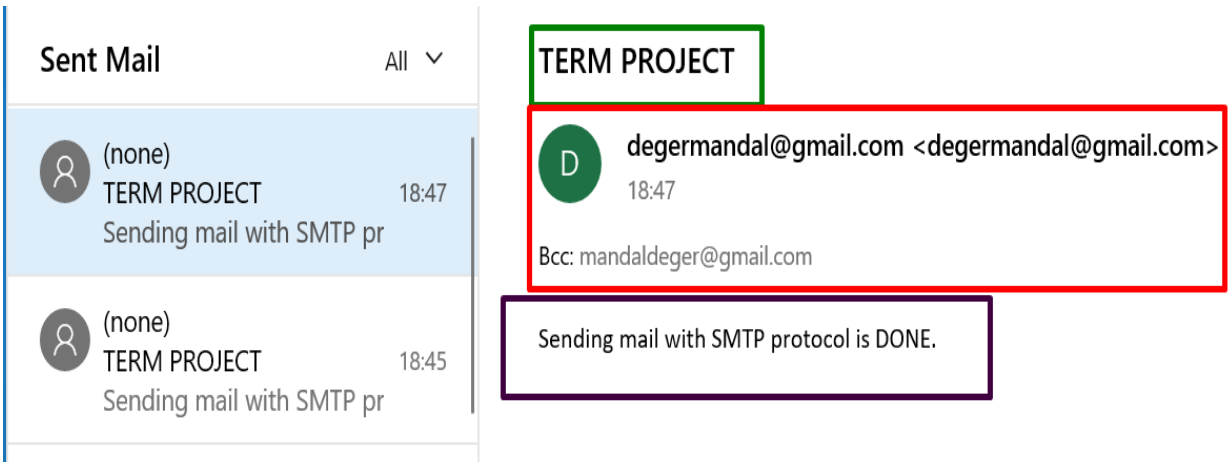
```
import ssl
from socket import *
import base64

#Mail Content
subject = "TERM PROJECT"
textT = "text/plain"
msg = "Sending mail with SMTP protocol is DONE."
endmsg = "\r\n.\r\n"

# Choose gmail mail server
mailserver = ("smtp.gmail.com", 587)

#Sender and receiver
#mailFrom = "mailFrom@gmail.com"
mailFrom = input("Mail From:")
rcptTo = input("Mail To:")
#rcptTo = "rcptTo@gmail.com"

passw = input("Password:")

#Authentication information
username = base64.b64encode(mailFrom.encode()).decode()
password = base64.b64encode(passw.encode()).decode()

# Create socket called clientSocket and establish a TCP connection
with mail server
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect(mailserver)

#Get response and print
recv = clientSocket.recv(1024).decode()
print(recv)
if recv[:3] != '220':
    print('220 reply not received from server.')

# Send HELO command and print server response.
heloCommand = 'HELO Alice\r\n'
clientSocket.send(heloCommand.encode())
recv1 = clientSocket.recv(1024).decode()
print(recv1)
if recv1[:3] != '250':
    print('250 reply not received from server.')

#Send STARTTLS command
command = 'STARTTLS\r\n'
clientSocket.send(command.encode())
#Get response and print
recv = clientSocket.recv(1024).decode()
print(recv)
if recv[:3] != '220':
    print('220 reply not received from server.')
```

```
#Wrap socket for security
Socket = ssl.wrap_socket(clientSocket)

#Send Auth Login command
Socket.sendall('AUTH LOGIN\r\n'.encode())
recv = Socket.recv(1024).decode()
print(recv)
if (recv[:3] != '334'):
    print('334 reply not received from server')

#Send username
Socket.sendall((username + '\r\n').encode())
recv = Socket.recv(1024).decode()
print(recv)
if (recv[:3] != '334'):
    print('334 reply not received from server')

#Send password
Socket.sendall((password + '\r\n').encode())
recv = Socket.recv(1024).decode()
print(recv)
if (recv[:3] != '235'):
    print('235 reply not received from server')

# Send Mail From command and print server response.
Socket.sendall(('MAIL FROM: <' + mailFrom + '>\r\n').encode())
recv = Socket.recv(1024).decode()
print(recv)
if (recv[:3] != '250'):
    print('250 reply not received from server')

# Send Rcpt To command and print server response.
Socket.sendall(('RCPT TO: <' + rcptTo + '>\r\n').encode())
recv = Socket.recv(1024).decode()
print(recv)
if (recv[:3] != '250'):
    print('250 reply not received from server')

# Send DATA command and print server response.
Socket.send('DATA\r\n'.encode())
recv = Socket.recv(1024).decode()
print(recv)
if (recv[:3] != '354'):
    print('354 reply not received from server')

# Send message data.
message = 'Mail from:' + mailFrom + '\r\n'
message += 'Recipient To:' + rcptTo + '\r\n'
message += 'Subject:' + subject + '\r\n'
message += 'Text type:' + textT + '\t\n'
message += '\r\n' + msg
Socket.sendall(message.encode())
```

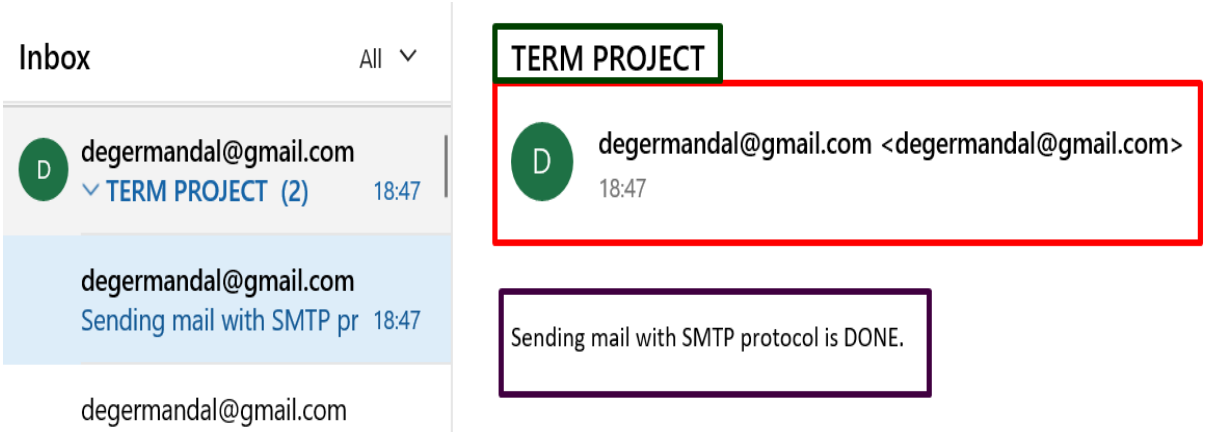
```
# Message ends with a single period.
Socket.sendall(endmsg.encode())
recv = Socket.recv(1024).decode()
print(recv)
if (recv[:3] != '250'):
    print('250 reply not received from server')

# Send QUIT command and get server response.
Socket.sendall('QUIT\r\n'.encode())

# Close connection
Socket.close()
```

Outputs:

Sender:



Receiver:



Terminal Output:

```
Run SMTPClient
C:\Users\Deger\AppData\Local\Programs\Python\Python38-32\python.exe
Mail From:degermandal@gmail.com
Mail To:mandaldeger@gmail.com
Password:[redacted]
220 smtp.gmail.com ESMTP ef11sm22720392ejb.15 - gsmtp

250 smtp.gmail.com at your service

220 2.0.0 Ready to start TLS

334 VXNlcm5hbWU6

334 UGFzc3dvcmQ6

235 2.7.0 Accepted

250 2.1.0 OK ef11sm22720392ejb.15 - gsmtp

250 2.1.5 OK ef11sm22720392ejb.15 - gsmtp

354  Go ahead ef11sm22720392ejb.15 - gsmtp

250 2.0.0 OK  1609698272 ef11sm22720392ejb.15 - gsmtp

Process finished with exit code 0
```