



**Birzeit University-Faculty of Engineering and Technology**  
**Electrical and Computer Engineering Department**

**COMPUTER NETWORKS ENCS3320**

**Network Project Report**  
**Complete Web Server**

**Partners:**

Mayar Abuzahra  
Shahed Jamhour  
Yasmeena Assi

**Student ID:** 1181239  
**Student ID:** 1180654  
**Student ID:** 1180899

**Supervised by:** Dr. Abdalkarim Awad

**Section:** 1

**Submission Date:** 14-8-2021

## ❖ Abstract

---

This project aims to implement HTTP-Web server using socket programming. Using python language, we have implemented a web server that does many types of requests [html file requests, pictures with different extensions, csv files requests]. If the request is not found, html file with specific properties will appear to tell the client that his/her request is not found.

## ❖ Contents

---

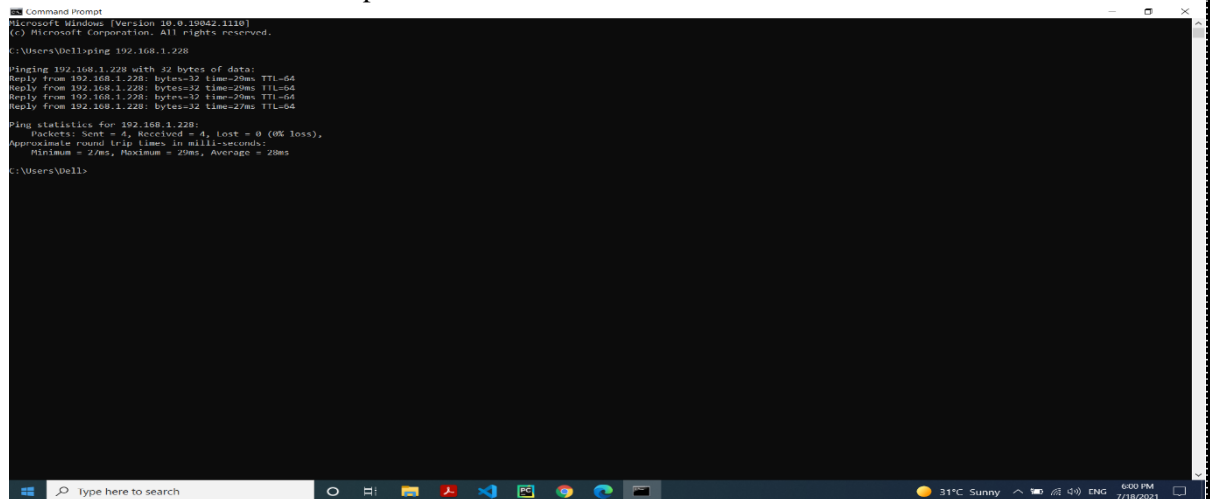
<b>Abstract .....</b>	<b>II</b>
<b>Procedure &amp; Calculations .....</b>	<b>1</b>
Part1 .....	1
Part2 .....	4
<b>Conclusion .....</b>	<b>9</b>

## ❖ Procedure & Calculations

### A. Part1:

#### 1. ping 192.168.1.228

We use this ping to test if the computer is reachable at specific Internet protocol, also to measure the round trip for a message that has been sent from a host to a destination computer, As we see in the picture, we can see there is 4 responses with a small response time. when we put enter after the command the request sent in 4 times, there will be a reply with a round trip static as we see or a timeout failure. As we see there is four lines which we can see from it the size of the data that we are going to see it ,and TTL time which means the time that packet has being inside the network before being discard, also , we can see the time of response, we can get effect from the ping instruction to see the time that we need to send and receive data so we know the speed ,Also as shown in the figure we can see how much packet we sent and how much we received , from this we can check if there is a lost packets.



```
Microsoft Windows [Version 10.0.19042.1110]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Dell>ping 192.168.1.228

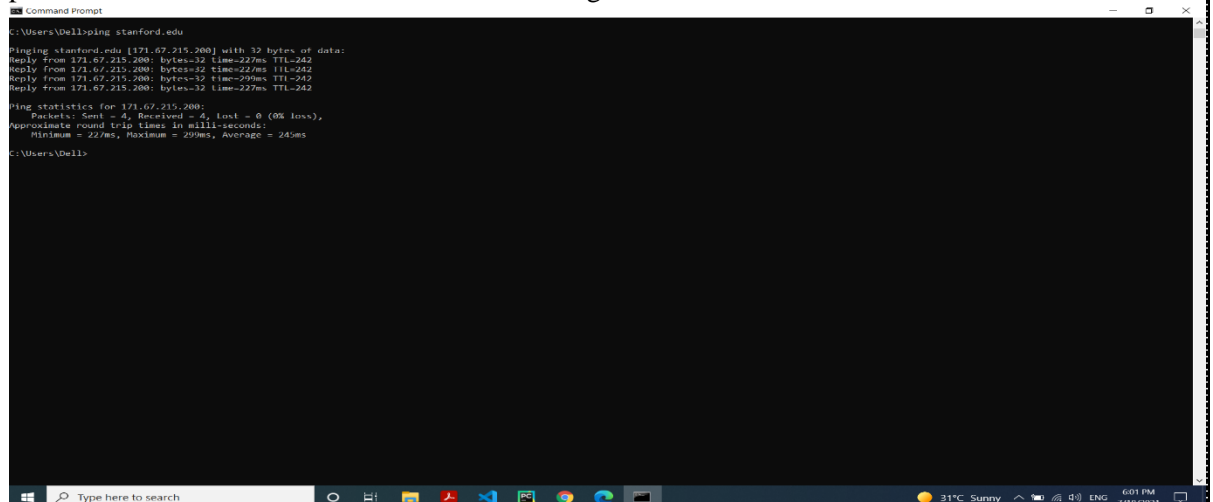
Pinging 192.168.1.228 with 32 bytes of data:
Reply from 192.168.1.228: bytes=32 time=29ms TTL=64
Reply from 192.168.1.228: bytes=32 time=29ms TTL=64
Reply from 192.168.1.228: bytes=32 time=29ms TTL=64
Reply from 192.168.1.228: bytes=32 time=27ms TTL=64

Ping statistics for 192.168.1.228:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 27ms, Maximum = 29ms, Average = 28ms

C:\Users\Dell>
```

#### 2. ping stanford.edu

As we said before we use a ping to check and test many things, from it we can transmits packet to a special Ip Address, and the time of response and the time to live. when we use this command when ping a host IP Address ,so the packet sent to the destination ip Address that we chose it , when it reached the destination Address send a response for the request As shown in the figure, we can see the number of packets that sent or received its equal so there is no loss , as we see from the figure lost=0, so the connection is successful due to these things , also we can notice the TTL time ,which means time to live (TTL) refers to the amount of time or that a packet is set to exist inside a network before being discarded.



```
Microsoft Windows [Version 10.0.19042.1110]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Dell>ping stanford.edu

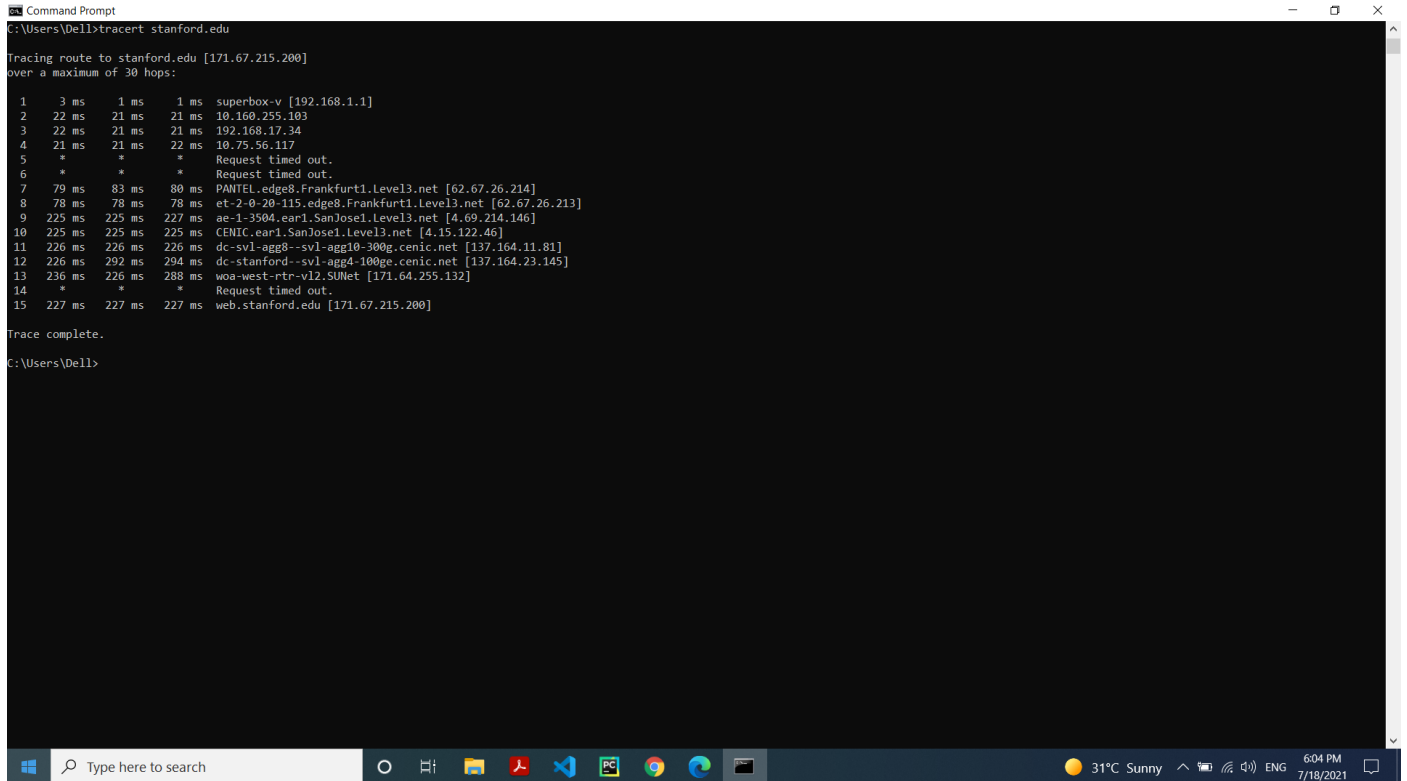
Pinging stanford.edu [171.67.215.200] with 32 bytes of data:
Reply from 171.67.215.200: bytes=32 time=22ms TTL=242
Reply from 171.67.215.200: bytes=32 time=22ms TTL=242
Reply from 171.67.215.200: bytes=32 time=29ms TTL=242
Reply from 171.67.215.200: bytes=32 time=22ms TTL=242

Ping statistics for 171.67.215.200:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 22ms, Maximum = 29ms, Average = 24ms

C:\Users\Dell>
```

### 3. tracer Stanford.edu

tracert is a command line included with windows and other operating system, We can use it to see the internet connection problem like the loses and latency, so this command tells us where is the problem exactly. We use it to get the path from our networks to the destination, also it gives me information about it like the routes in the path number of hops, also if there is a mistake, the response times it shows to us the path traffic takes to reach a specific website, also we can see the delays in each stop.



```
Command Prompt
C:\Users\Dell>tracert stanford.edu

Tracing route to stanford.edu [171.67.215.200]
over a maximum of 30 hops:
 0  3 ms  1 ms  1 ms  superbox-v [192.168.1.1]
 1  22 ms  21 ms  21 ms  10.160.255.103
 2  22 ms  21 ms  21 ms  192.168.17.34
 3  21 ms  21 ms  22 ms  10.75.56.117
 4  *      *      *      Request timed out.
 5  *      *      *      Request timed out.
 6  79 ms  83 ms  80 ms  PANTEL.edge8.Frankfurt1.Level3.net [62.67.26.214]
 7  78 ms  78 ms  78 ms  et-2-0-20-115.edge8.Frankfurt1.Level3.net [62.67.26.213]
 8  225 ms  225 ms  227 ms  ae-1-3504.ear1.SanJose1.Level3.net [4.69.214.146]
 9  225 ms  225 ms  225 ms  CBULC.ear1.SanJose1.Level3.net [4.15.122.46]
10  226 ms  226 ms  226 ms  dc-svl-agg8-svl-agg10-300g.cenic.net [137.164.11.81]
11  226 ms  292 ms  294 ms  dc-stanford-svl-agg4-100g.cenic.net [137.164.23.145]
12  236 ms  226 ms  288 ms  woa-west-rtr-v12.SUNet [171.64.255.132]
13  *      *      *      Request timed out.
14  227 ms  227 ms  227 ms  web.stanford.edu [171.67.215.200]

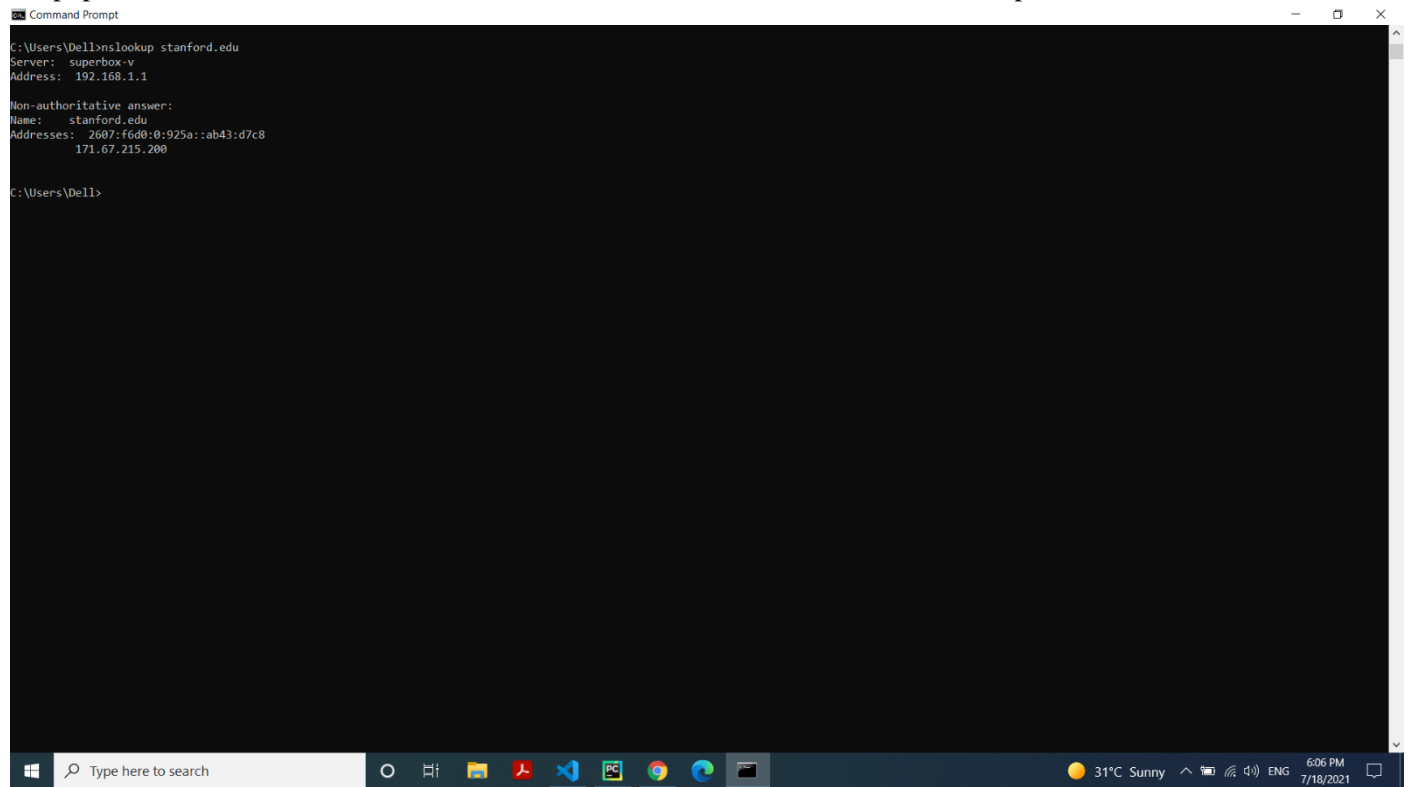
Trace complete.

C:\Users\Dell>
```

### 4. nslookup stanford.edu

nslookup command is available in many operating systems as windows, Linux, macOS, we can use it to get the DNS queries and receive the IP addresses and domain names. This command has two modes: interactive mode the command from itself give a default name server when there is no arguments are given, The non-interactive mode when there is a name or internet Address to looked up for it. So, this command. Here as we see from the figure there is a local DNS server which used to reach the IP address of the server from the closer DNS. Also here in the figure shows the IP Address for a DNS server, and

the popular name for the ,Also the IP address for the server standford, also a name of paths to reach to the server



```
Command Prompt
C:\Users\Dell>nslookup stanford.edu
Server: superbox-v
Address: 192.168.1.1

Non-authoritative answer:
Name: stanford.edu
Addresses: 2607:f6d0:0:925a::ab43:d7c8
          171.67.215.200

C:\Users\Dell>
```

The screenshot shows a Windows Command Prompt window with a black background and white text. The title bar at the top reads "Command Prompt". The command prompt shows the execution of the 'nslookup stanford.edu' command. The output displays the server information for 'stanford.edu', including the IP address 192.168.1.1 and a non-authoritative answer with multiple IP addresses: 2607:f6d0:0:925a::ab43:d7c8 and 171.67.215.200. The Windows taskbar is visible at the bottom, showing the search bar, taskbar icons, and system tray with the date and time (6:06 PM, 7/18/2021).

### B. Part2:

In this part, many types of requests will be applied and different kinds of responds will be done.

All files are attached in project submission:

The screenshot shows the PyCharm IDE with the following details:

- Project:** project1 network
- File:** part2.py
- Code:**

```

1 import socket
2 import csv
3
4 # open connection
5 PORT_NUM = 5000
6 size = 1024
7 try:
8     socket_def = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
9     socket_def.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
10    socket_def.bind(('', PORT_NUM))
11    socket_def.listen(1)
12    print('Connect to server with port number=', PORT_NUM)
13 except socket.error as e:
14    print("error")
15
16 # create two files (sortByName, sortByPrice) in which they aims to sort 'smartphones.csv' according to name and price respectively
17 file = 'smartphones.csv'
18 sortByName = open('sortByName', 'w')
19 sortByPrice = open('sortByPrice', 'w')
20 reader = csv.reader(open(file, delimiter=","))

```
- Run:** part2.py
  - Command: "C:/Users/Dell/PycharmProjects/project1 network/venv/Scripts/python.exe" "C:/Users/Dell/PycharmProjects/project1 network/part2.py"
  - Output: Connect to server with port number= 5000
  - Status: Process finished with exit code -1
- Terminal:** Empty
- Python Console:** Empty
- Event Log:** Empty

After importing socket and csv libraries, this piece of code aims to create a UDP socket for server and if there is not any error in socket creation, the socket will be bind to local port number '5000'.

The screenshot displays the PyCharm IDE interface. The main editor window shows a Python file named `part2.py` containing a simple HTTP server implementation. The code defines a `socket_def` object, listens for incoming connections, and processes requests by splitting them into headers and body parts.

```
response = ('HTTP/1.0 200 OK' + '\n\n' + 'Content-type: text/html').encode()

while (1):
    connection_address = socket_def.accept() #Accept connection with client side
    ip = address[0]
    port = address[1]
    addr=(ip,port)
    print("New connection {addr} connected.")
    FileNameRequest = connection.recv(size).decode('utf-8')
    if not FileNameRequest:
        # if data is not received break
        break
    print(f"[Client request]:")
    requestAfterSplitting = FileNameRequest.split() # Split request from spaces
    request_file = requestAfterSplitting[1]
    sf = request_file.split('?')
    extFile=sf[0]
    extFile = extFile.strip('/')

    fextFile=f'static/{extFile}'
    while (1)
```

The left sidebar shows the project structure, including files like `venv`, `Include`, `Lib`, `Scrips`, `pywemcfg`, `Capture.PNG`, `file.html`, `local.html`, `main.html`, `part2.py`, `Picture.jpg`, `smartphones.csv`, `sortByPrice`, `sortByName`, `style.css`, and `External Libraries`.

The bottom panel shows the Run console output:

```
"C:\Users\Dell\PycharmProjects\project1 network\venv\Scripts/python.exe" "C:/Users/Dell/PycharmProjects/project1 network/part2.py"
Connect to server with port number= 5000

Process finished with exit code -1
```

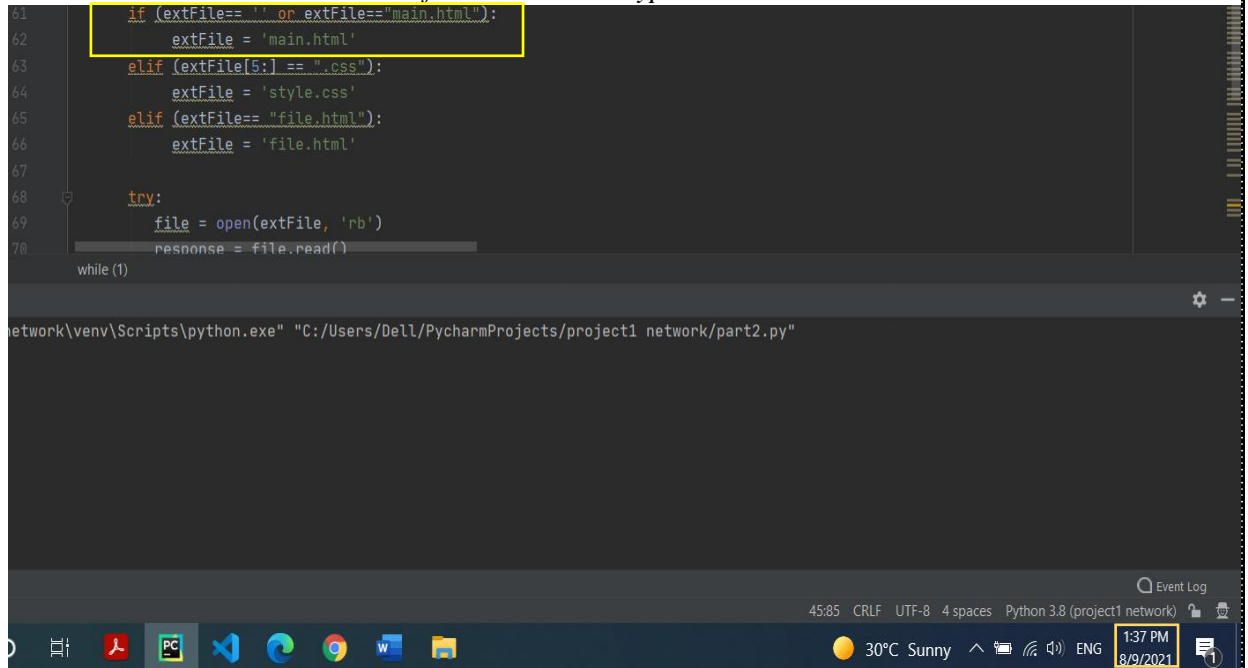
The status bar at the bottom indicates the current file is `part2.py`, the encoding is UTF-8, and the interpreter is Python 3.8 (project1 network).

At first, line 45 determines which file represents the client request [html file, pictures, text files]. To get the name of the request file: 1- Split the 'FileNameRequest' by spaces and take the second string and save it in 'request\_file'.

2- Split 'request\_file' by '?' and take the first value after splitting and delete '/' from it because our request will be in this format: [http://localhost:5000/file\\_name](http://localhost:5000/file_name).

After getting the file name, we need to check which file request is applied:

1- if the request is / or /index.html (for example localhost:9000/ or localhost:9000/index.html) then the server should send main.html file with ContentType: text/html.



```
61 if (extFile == '' or extFile == "main.html"):
62     extFile = 'main.html'
63 elif (extFile[5:] == ".css"):
64     extFile = 'style.css'
65 elif (extFile == "file.html"):
66     extFile = 'file.html'
67
68 try:
69     file = open(extFile, 'rb')
70     response = file.read()
71
72 while (1)
```

network\venv\Scripts\python.exe "C:/Users/Dell/PycharmProjects/project1 network/part2.py"

45:85 CRLF UTF-8 4 spaces Python 3.8 (project1 network) 1:37 PM 8/9/2021




If the request is 'main.html' or if the request is empty request, the response is:

ENC53320 My Simple Webserver

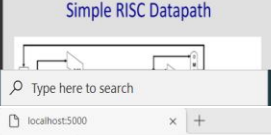
welcome to our course **Computer Networks**

Mayar Abuzahra 1181239

Hi, my name is Mayar Abuzahra. I am 21 years old. Here is some of my projects. To access them, click on the pictures:




Shell Script Project




Simple RISC Datapath

Shahed Jamhour 1180654

I'm ShahEd :) I'm 20. I do Computer Engineering. Here are 2 Projects I've completed through my third year of college. Click on the pictures if you'd like to check them out.




Shell Script Project  
(Text Summarization using Sentence Centrality)



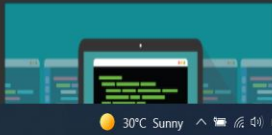
Data Structure Project  
(Equation Manager)

Yasmeena Assi 1180899

My name is Yasmeena Assi. I am 21 years old. Here is some of my projects I have worked in the university.



Data Base project

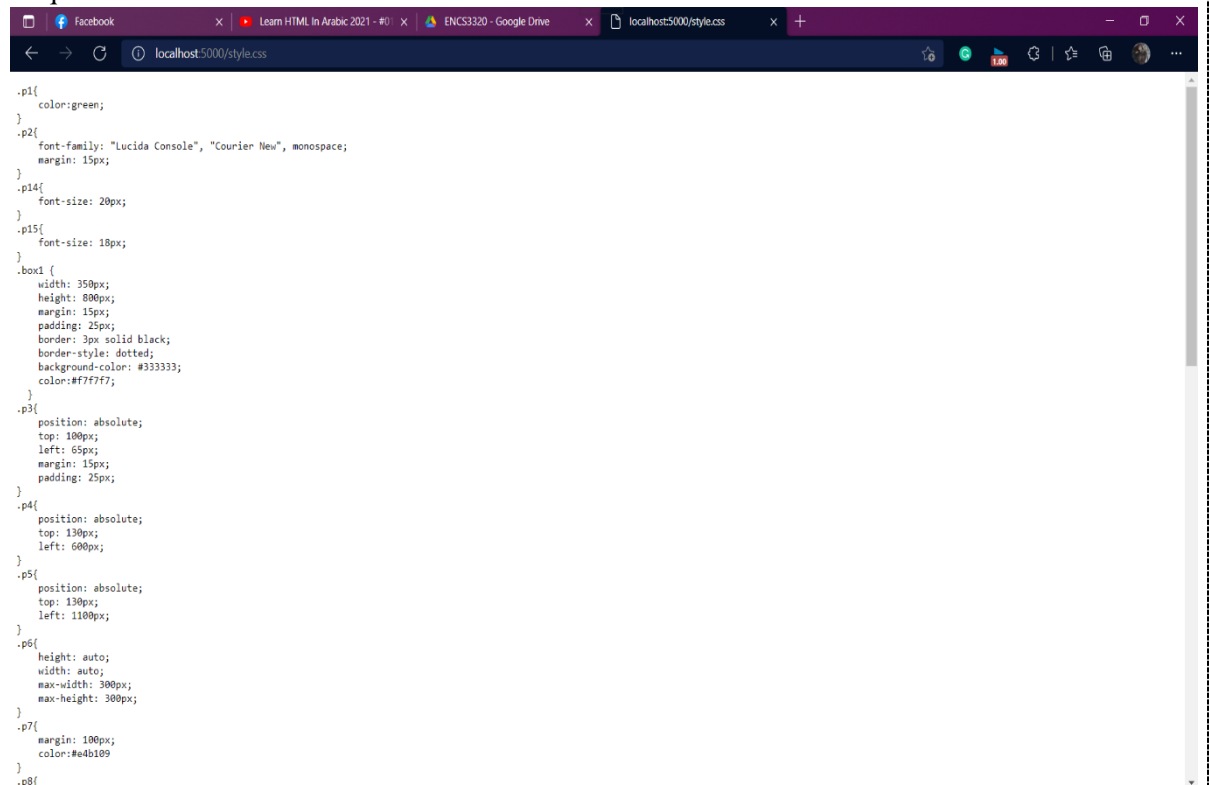


linux project on shell

To access w3schools website [click here](#)

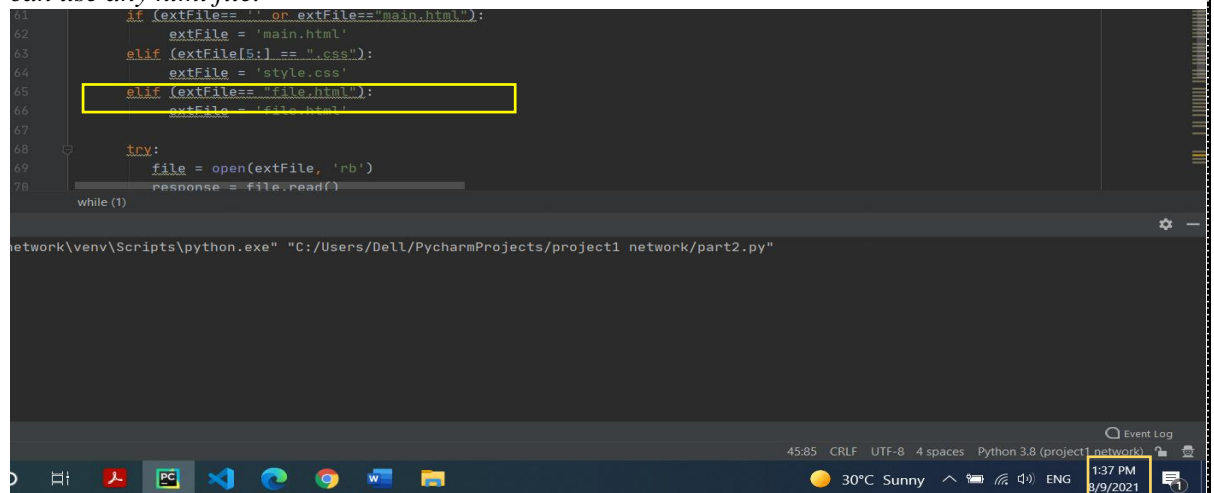
To access local.html file [click here](#)

Request test from another device:



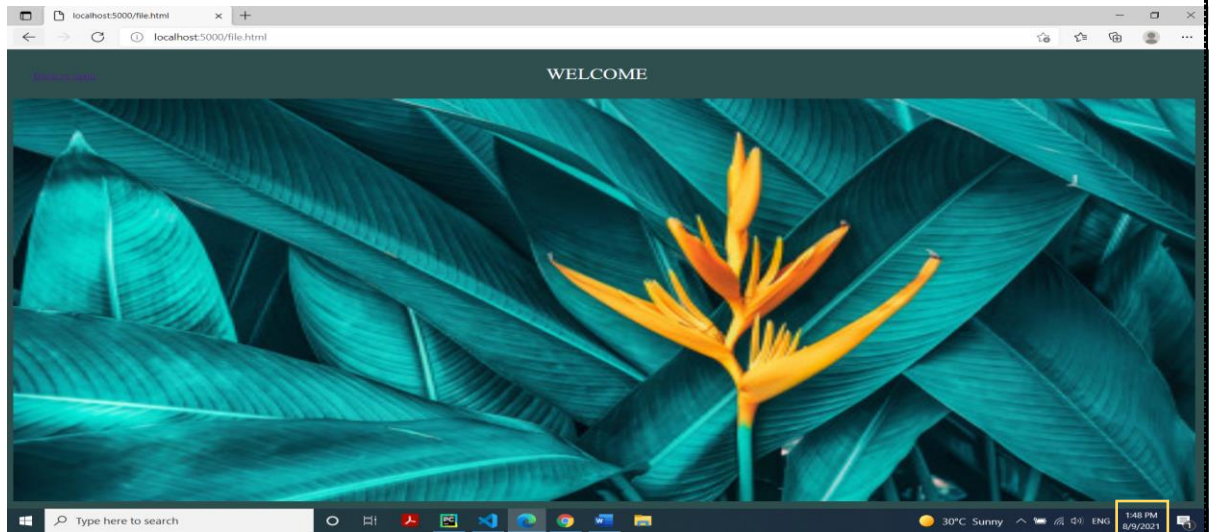
```
.p1{
  color:green;
}
.p2{
  font-family: "Lucida Console", "Courier New", monospace;
  margin: 15px;
}
.p14{
  font-size: 20px;
}
.p15{
  font-size: 18px;
}
.box1 {
  width: 350px;
  height: 800px;
  margin: 15px;
  padding: 25px;
  border: 3px solid black;
  border-style: dotted;
  background-color: #333333;
  color:#f7f7f7;
}
.p3{
  position: absolute;
  top: 100px;
  left: 65px;
  margin: 15px;
  padding: 25px;
}
.p4{
  position: absolute;
  top: 130px;
  left: 600px;
}
.p5{
  position: absolute;
  top: 130px;
  left: 1100px;
}
.p6{
  height: auto;
  width: auto;
  max-width: 300px;
  max-height: 300px;
}
.p7{
  margin: 100px;
  color:#e4b109
}
.p8{
```

2- if the request is /file.html then the server should send html file with Content-Type: text/html. You can use any html file.

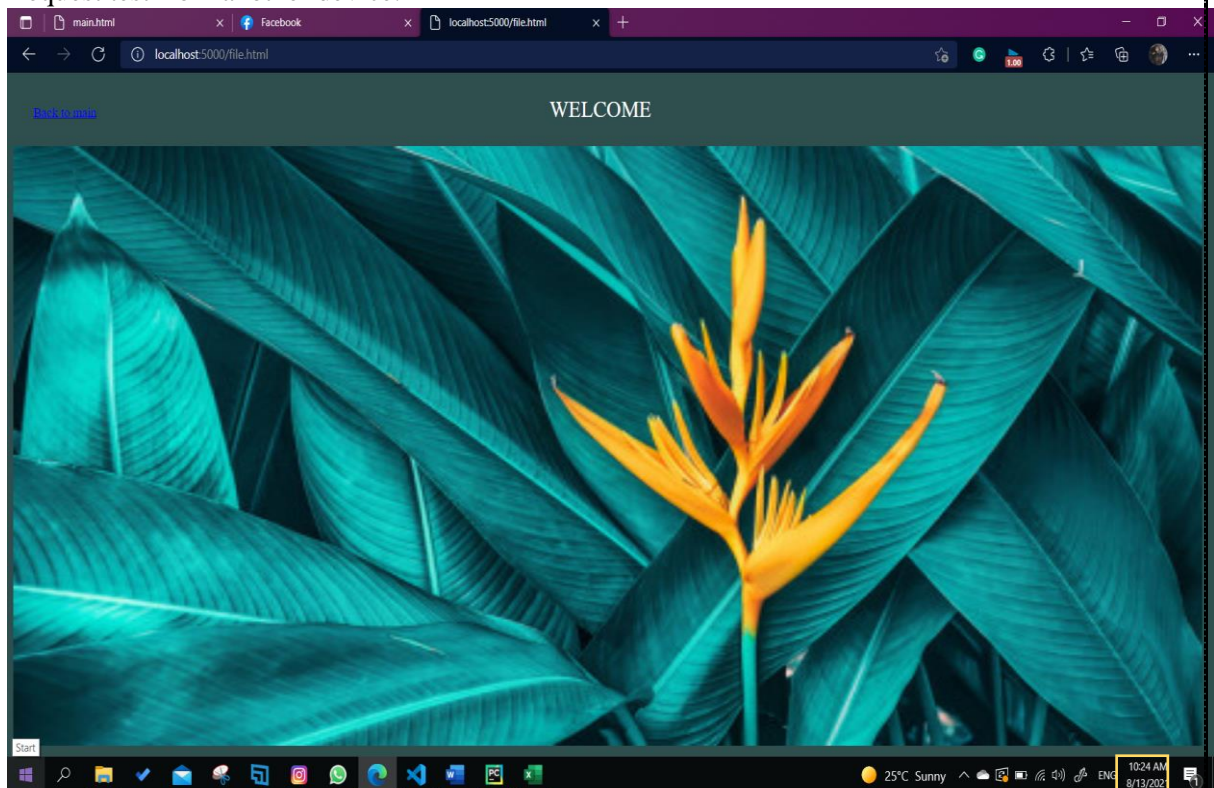


```
61 if (extFile == "" or extFile == "main.html"):
62     extFile = 'main.html'
63 elif (extFile[5:] == ".css"):
64     extFile = 'style.css'
65 elif (extFile == "file.html"):
66     extFile = 'file.html'
67
68 try:
69     file = open(extFile, 'rb')
70     response = file.read()
71
72 while (1)
```

If the request is 'file.html', the response is:



Request test from another device:



3- if the request is /file.css then the server should send html file with Content-Type: text/html. You can use any image.

```
61 if (extFile== "" or extFile=="main.html"):
62     extFile = 'main.html'
63     elif (extFile[5:] == ".css"):
64         extFile = 'style.css'
65     elif (extFile=="file.html"):
66         extFile = 'file.html'
67
68     try:
69         file = open(extFile, 'rb')
70         response = file.read()
71     except:
72         response = "File not found"
73
74 while (1)
```

network\venv\Scripts\python.exe "C:/Users/Dell/PycharmProjects/project1 network/part2.py"

If the request is 'style.css', the response is:

```
localhost:5000/style.css
localhost:5000/style.css

.p1{
  color:green;
}
.p2{
  font-family: "Lucida Console", "Courier New", monospace;
  margin: 15px;
}
.p14{
  font-size: 20px;
}
.p15{
  font-size: 18px;
}
.box1 {
  width: 350px;
  height: 800px;
  margin: 15px;
  padding: 25px;
  border: 3px solid black;
  border-style: dotted;
  background-color: #933333;
  color:#f7f7f7;
}
.p3{
  position: absolute;
  top: 100px;
  left: 65px;
  margin: 15px;
  padding: 25px;
}
.p4{
  position: absolute;
  top: 130px;
  left: 600px;
}
.p5{
  position: absolute;
  top: 130px;
  left: 1100px;
}
.p6{
  height: auto;
  width: auto;
  max-width: 300px;
  max-height: 300px;
}
.p7{
  margin: 100px;
  color:red;
}
```

4- if the request is /imagenam.png then the server should send the png image with Content-Type: image/png. You can use any image.

At first, the file will be read in try-catch format:

```
part2.py
project1 network - part2.py - PyCharm

project1 network
part2.py
file.html
sortByName
sortByPrice
smartphones.csv
sortByName

try:
    file = open(extFile, 'rb')
    response = file.read()
    file.close()
except:
    response = "File not found"

if (extFile[2:]=="png"):
    ft = "image/png"
elif (extFile[2:]=="css"):
    ft = "text/css"
elif (extFile.endswith(".csv")):
    ft = "text/csv"
else:
    ft = ""

header_line = "HTTP/1.1 200 OK\r\n" + "Content-Type: " + str(ft) + "\r\n"
print(header_line)

while (1)
```

Run: part2

NEW Connection ('127.0.0.1', 4757) connected.

[Client request]:

HTTP/1.1 200 OK

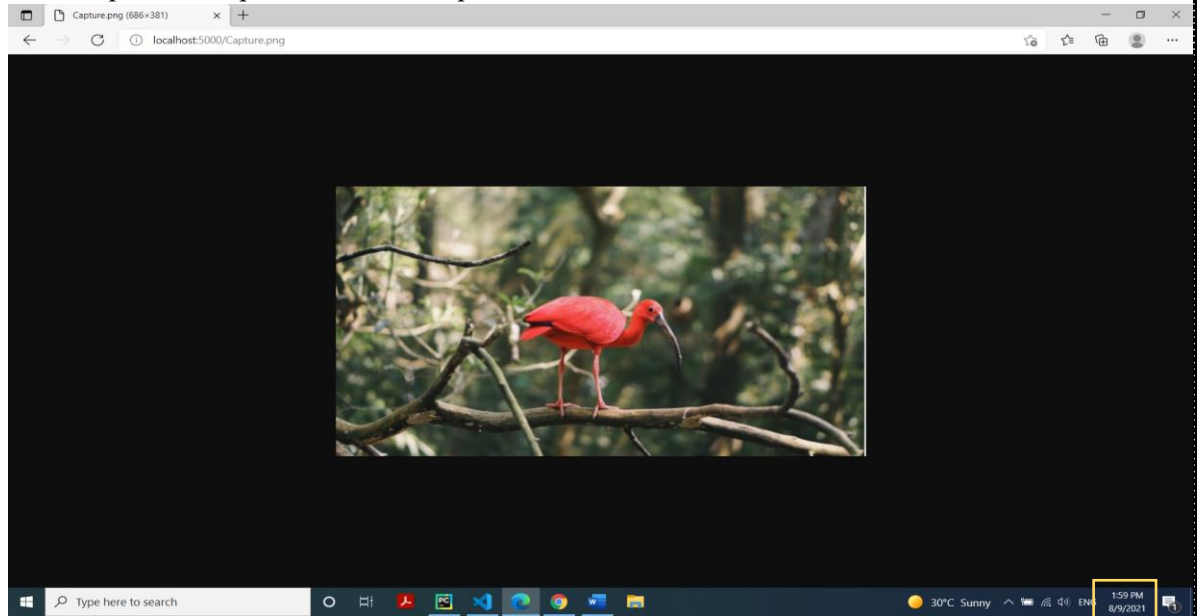
Content-Type:

New connection ('127.0.0.1', 62829) connected.

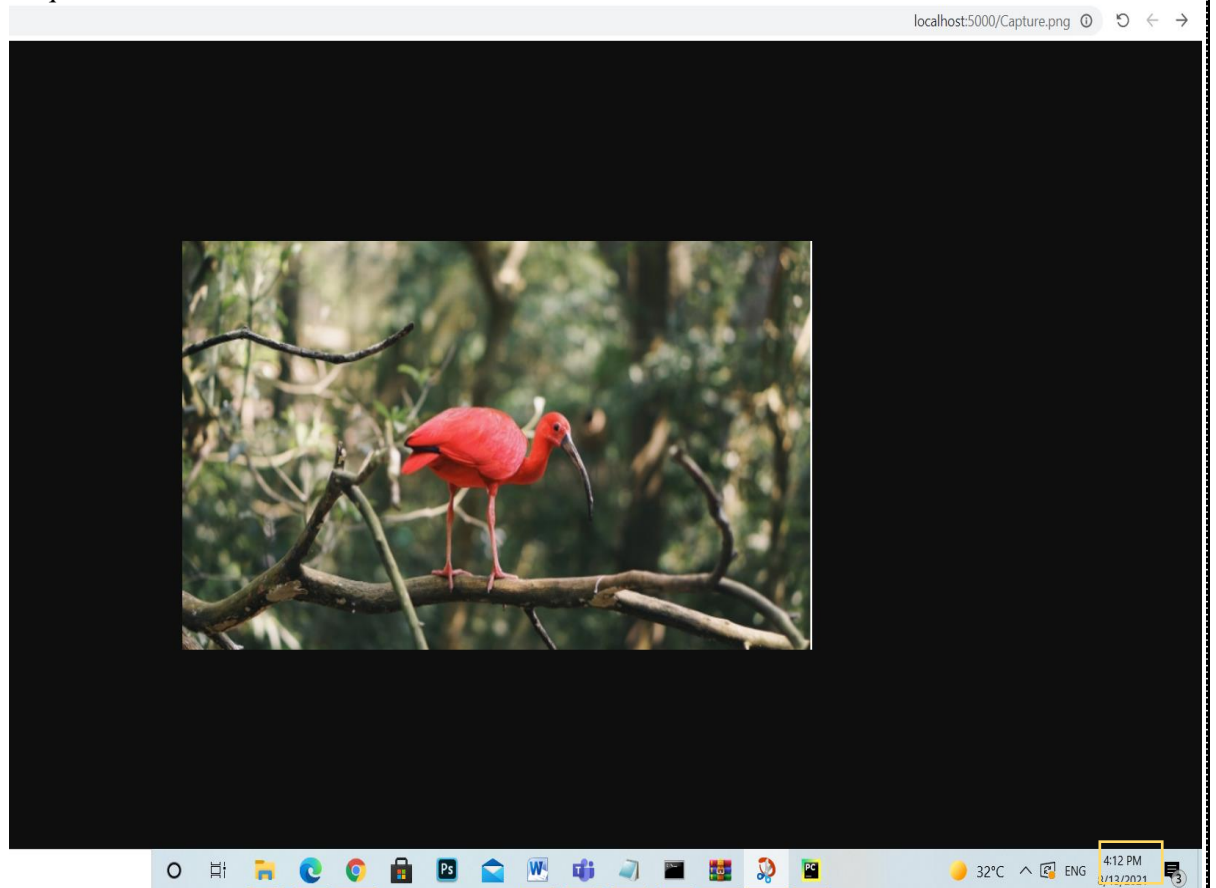
Process finished with exit code -1



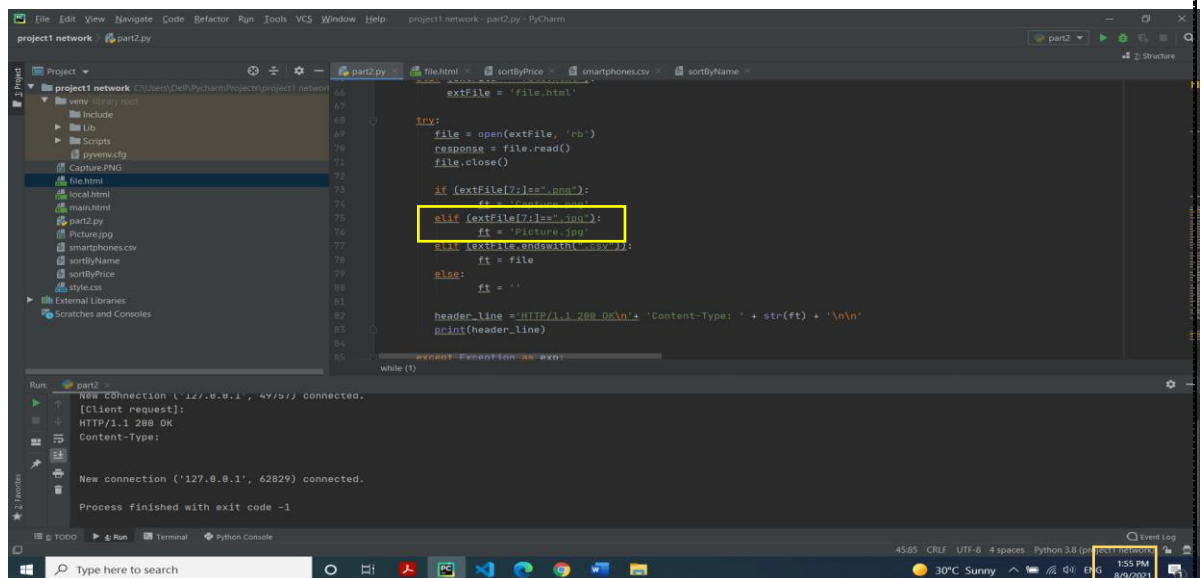
If the request is 'Capture.PNG', the response is:



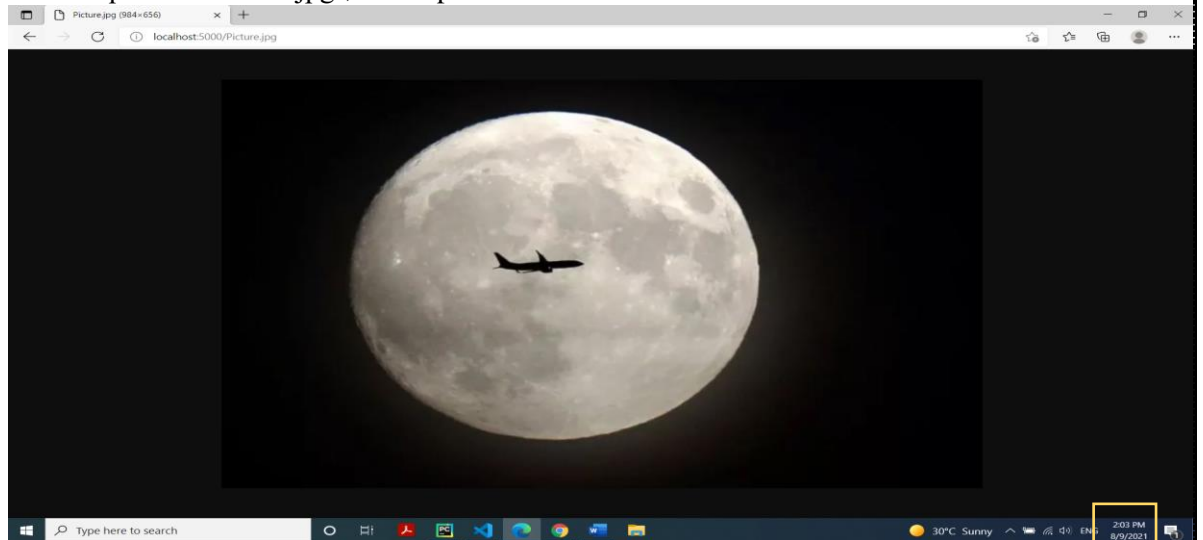
Request test from another device:



5- if the request is /imagenname.jpg then the server should send the jpg image with Content-Type: image/jpeg. You can use any image.

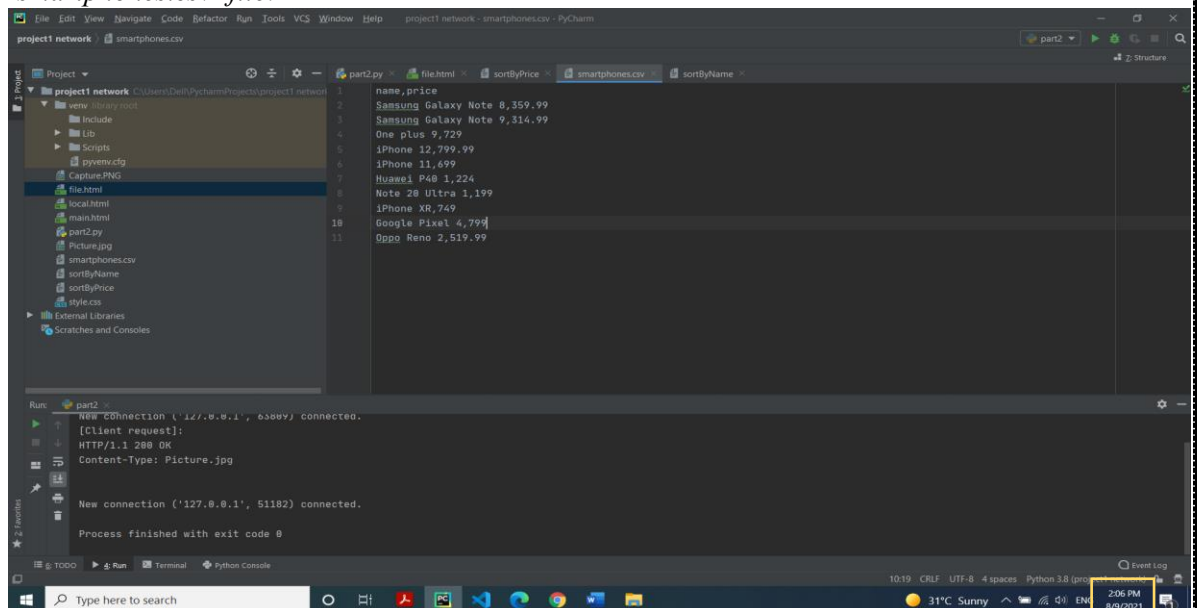


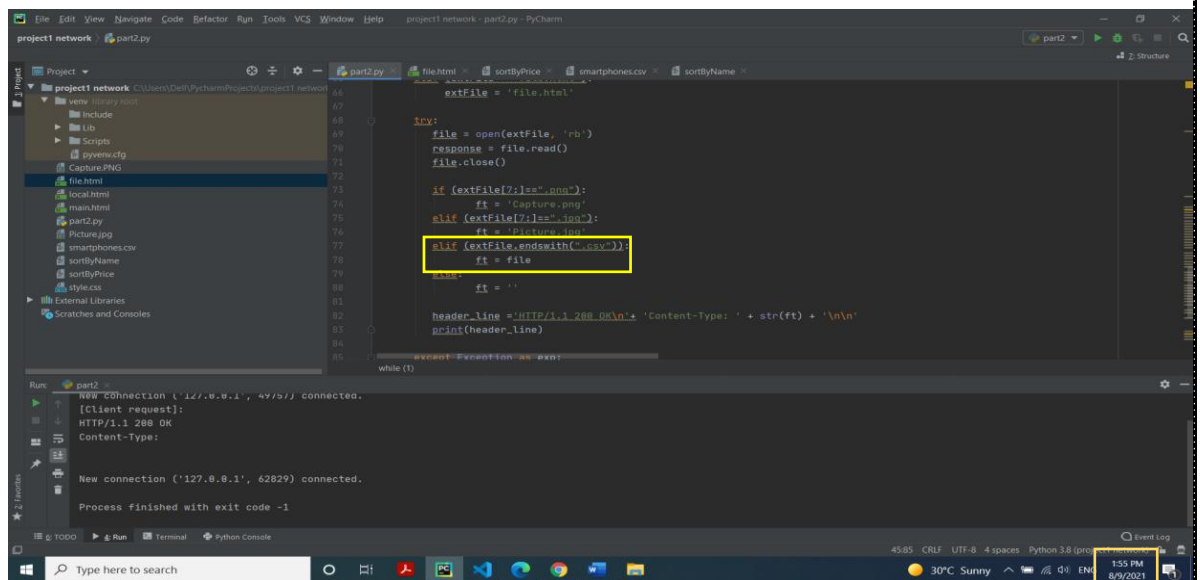
If the request is 'Picture.jpg', the response is:



6- Include a text file (or you can use csv file) that contains names and prices of at least 10 smartphones

'smartphones.csv' file:





If the request is '*smartphones.csv*', the response is:



7- if the request is */SortByName* then the output on the browser should be the names and prices of the smartphones sorted by the name. The server should send text page with Content-Type: *text/plain*. If you wish, you can use *text/html* to display the output in a more convenient way. This piece of code aims to sort file according to name and save it in '*sortByName*' file.

```

15
16 # create two files [sortByName, sortByPrice] in which they aim to sort 'smartphones.csv' according to name and price respectively
17 file = 'smartphones.csv'
18 sortByName = open('sortByName', 'w')
19 sortByPrice = open('sortByPrice', 'w')
20 reader = csv.reader(open(file), delimiter=',')
21 next(reader)
22 names = sorted(reader, key=lambda row: row[0], reverse=False)
23 reader = csv.reader(open(file), delimiter=',')
24 next(reader)
25 price = sorted(reader, key=lambda row: row[1], reverse=False)
26 with open('sortByName', 'w') as f:
27     for item in names:
28         f.write("%s\n" % item)
29 with open('sortByPrice', 'w') as f:
30     for item in price:
31         f.write("%s\n" % item)
32 sortByName.close()
33 sortByPrice.close()
34
with open('sortByPrice', 'w') a... for item in price

```

Run: part2 ×  
 New connection ('127.0.0.1', 63889) connected.  
 [Client request]:  
 HTTP/1.1 200 OK  
 Content-Type: Picture.jpg  
 New connection ('127.0.0.1', 51182) connected.  
 Process finished with exit code 0

If the request is 'sortByName', the response is:

```

["Google Pixel 4", "799"]
["Huawei P40 1", "224"]
["Note 20 Ultra 1", "199"]
["One plus 9", "725"]
["Oppo Reno 2", "519.99"]
["Samsung Galaxy Note 8", "359.99"]
["Samsung Galaxy Note 9", "314.99"]
["iPhone 11", "699"]
["iPhone 12", "799.99"]
["iPhone XR", "749"]

```

8- if the request is /SortByPrice then the output on the browser should be name and price of the smartphones sorted by its price. The server should send text page with ContentType: text/plain. If you wish, you can use text/html to display the output in a more convenient way. This piece of code aims to sort file according to price and save it in 'sortByPrice' file.



The screenshot shows the PyCharm IDE with a project named 'project1 network'. The file explorer on the left shows a directory structure with files like 'part2.py', 'smartphones.csv', 'sortByPrice', and 'sortByName'. The main editor displays the code in 'part2.py', which reads a CSV file and sorts it by name and price. The Run window at the bottom shows the execution output, including connection messages and the final exit code 0.

```

15
16 # create two files [sortByName, sortByPrice] in which they aim to sort 'smartphones.csv' according to name and price respectively
17 file = 'smartphones.csv'
18 sortByName = open('sortByName', 'w')
19 sortByPrice = open('sortByPrice', 'w')
20 reader = csv.reader(open(file), delimiter=",")
21 next(reader)
22 names = sorted(reader, key=lambda row: row[0], reverse=False)
23 reader = csv.reader(open(file), delimiter=",")
24 next(reader)
25 price = sorted(reader, key=lambda row: row[1], reverse=False)
26 with open('sortByName', 'w') as f:
27     for item in names:
28         f.write("%s\n" % item)
29 with open('sortByPrice', 'w') as f:
30     for item in price:
31         f.write("%s\n" % item)
32 sortByName.close()
33 sortByPrice.close()
34
with open('sortByPrice', 'w') as f:
    for item in price:

```

Run: part2.py  
 New connection ('127.0.0.1', 63889) connected.  
 [Client request]:  
 HTTP/1.1 200 OK  
 Content-Type: Picture.jpg  
 New connection ('127.0.0.1', 51182) connected.  
 Process finished with exit code 0

If the request is 'sortByPrice', the response is:

The screenshot shows a web browser window with the address bar set to 'localhost:5000/sortByPrice'. The page content displays a list of smartphones sorted by price, each with its name and price in a list format.

```

["Note 20 Ultra 1", "199"]
["Huawei P40 1", "224"]
["Samsung Galaxy Note 3", "314.99"]
["Samsung Galaxy Note 8", "359.99"]
["Oppo Reno 2", "539.99"]
["iPhone 11", "699"]
["One plus 9", "729"]
["iPhone XR", "749"]
["Google Pixel 4", "799"]
["iPhone 12", "799.99"]

```

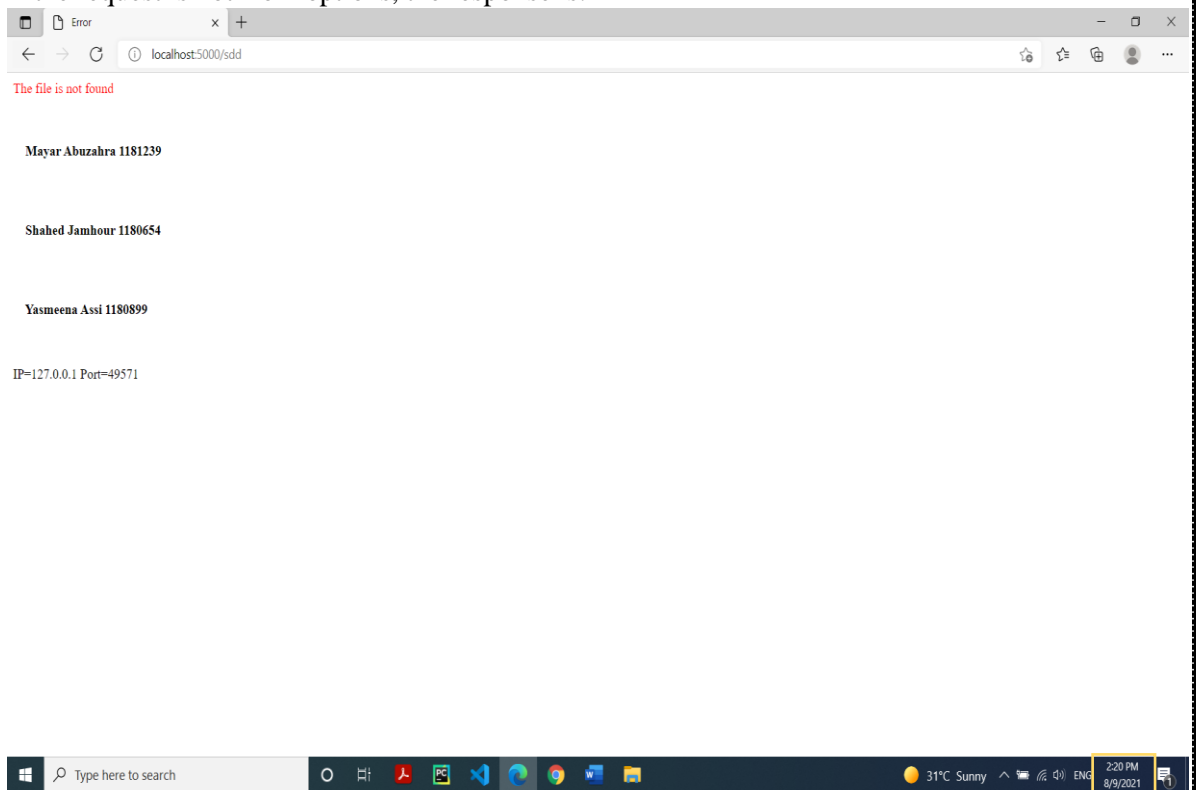
9- If the request is wrong or the file doesn't exist the server should return a simple HTML webpage that contains (Content-Type: text/html)

```
18 for item in price:
19     f.write("%s\n" % item)
20 # print(name)
21 # print(price)
22 sortByPrice.close()
23 sortByPrice.close()
24
25 def request_wrong():
26     global response
27     response = <html><head><title>Error</title></head><body><p style="color:red">The file is not found</p><br><p><b>IP: <ip> Port: <port></b></p></body></html>
28     response = ("IP=" + str(ip) + " Port=" + str(port)).encode()
29
30 while (1):
31     connection,address = socket_def.accept() #Accept connection with client side
32     ip = address[0]
33     port = address[1]
34     addr=(ip,port)
35     print("New connection "+addr+" connected.")
36     with open(sortByName, 'w') as f:
37         for item in names:
38             f.write("%s\n" % item)
```

Run console output:

```
New connection ('127.0.0.1', 49496) connected.
[Client request]:
HTTP/1.1 200 OK
Content-Type:
New connection ('127.0.0.1', 49496) connected.
Process finished with exit code -1
```

If the request is not from options, the response is:



And this a request using command prompt:

The screenshot displays a web browser window and a Windows Command Prompt. The browser window shows a web page titled "ENC53320 My Simple Webserver" with the text "welcome to our course Computer Networks" and the name "Mayar Abuzahra 1181239". The Command Prompt shows the execution of the command `C:\Users\Dell>cd C:\Users\Dell\PycharmProjects\project1 network` and the output `C:\Users\Dell\PycharmProjects\project1 network>main.html`.

main.html

ENC53320 My Simple Webserver

welcome to our course Computer Networks

Mayar Abuzahra 1181239

Hi, my name is Mayar Abuzahra. I am 21 years old. Here is some of my projects. To access them, click on the pictures:

Shell Script Project

Simple RISC Datapath

I'm ShahEd :) Here are 2 Projects of my college. check them out.

Shell Script Project (Text Summarization using Sentence Centrality)

Data Base project

31°C Sunny

## ❖ Conclusion

---

In summary, after doing this project, we have noticed how the HTTP-Web server works and how it responds to various requests by viewing the appropriate webpages and files.

On the whole, it was a beautiful project that let us realize the importance of using HTTP-Web server, and all of its objectives were obtained.