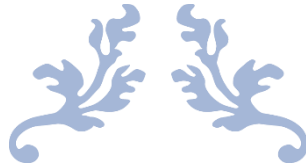


ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

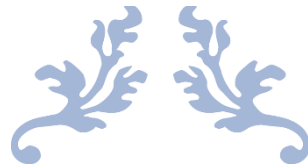


ĐỀ TÀI NGHIÊN CỨU
LẬP TRÌNH SOCKET



Trần Đông Ba 19127334
Kiều Hải Đăng 19127347

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN



ĐỀ TÀI NGHIÊN CỨU
LẬP TRÌNH SOCKET



Trần Đông Ba 19127334
Kiều Hải Đăng 19127347

|Giáo viên hướng dẫn|

ThS. Lê Hà Minh

Ths. Nguyễn Thanh Quân

Thành phố Hồ Chí Minh - 2020

MỤC LỤC

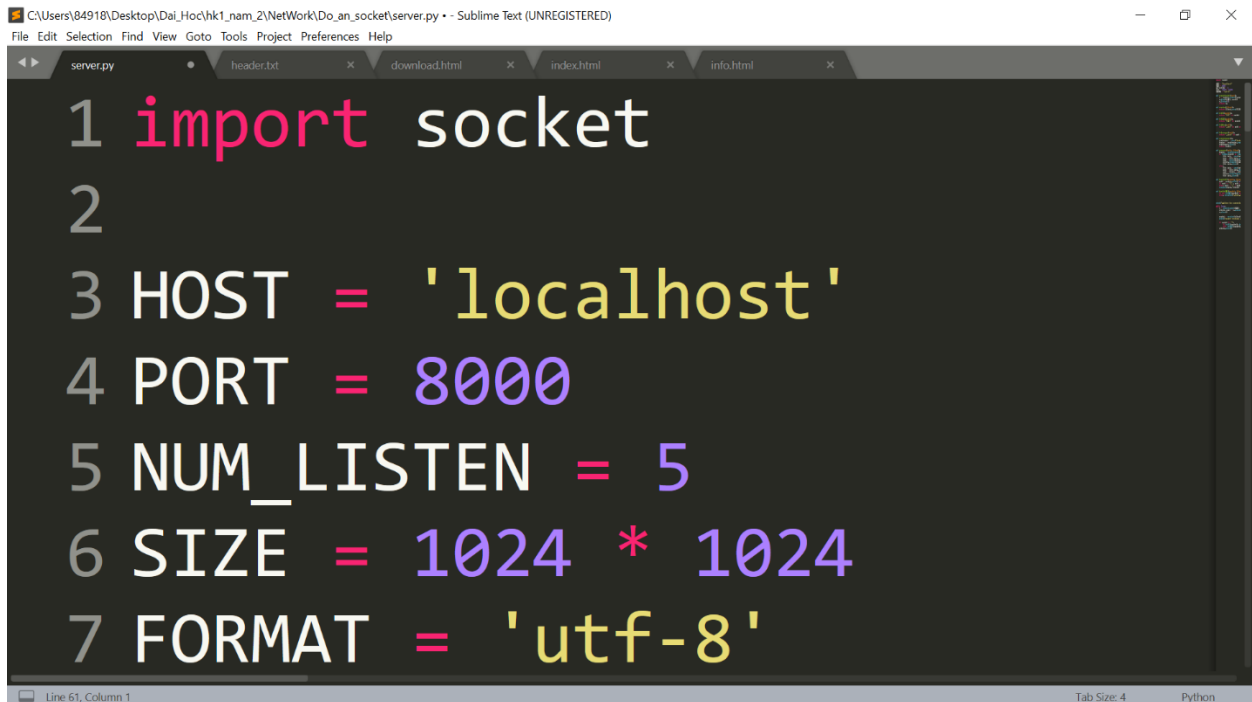
MỤC LỤC	1
PHÂN CÔNG CÔNG VIỆC.....	2
MÔ TẢ THUẬT TOÁN.....	3
DEMO CÁC BƯỚC THỰC HIỆN	12

PHÂN CÔNG CÔNG VIỆC

1. Trần Đông Ba – 19127334:
 - Tìm hiểu thuật toán và code.
 - Tìm hiểu về Python, HTML.
 - Xây dựng code mẫu.
2. Kiều Hải Đăng – 19127347:
 - Tìm hiểu thuật toán và code.
 - Tối ưu code.
 - Viết báo cáo, mô tả thuật toán và demo Web Server.
3. Tiến độ:
 - Hoàn thành ngày 4/12/2020.
 - Mọi thành viên đều làm việc tích cực và nộp sản phẩm đúng thời hạn.

MÔ TẢ THUẬT TOÁN

1. Nạp thư viện socket và khai báo các hằng số cần thiết:



```
1 import socket
2
3 HOST = 'localhost'
4 PORT = 8000
5 NUM_LISTEN = 5
6 SIZE = 1024 * 1024
7 FORMAT = 'utf-8'
```

2. Ta sẽ tạo ra một socket bằng hàm createSocket(port) do ta viết, truyền vào cổng ta muốn mở để server sử dụng cho việc truyền/nhận các gói tin.

Trong đó:

- Hàm socket được tích hợp sẵn trong thư viện socket để tạo ra một socket.
- AF_INET là IPV4.
- SOCK_STREAM là giao thức TCP.
- Tiếp đó ta tiến hành bind để xác nhận socket và mở port trên máy tính. Với lưu lượng truy cập tối đa là 5 client. Sau đó ta trả về một socket để sử dụng.

```
C:\Users\B4918\Desktop\Dat_Hoc\hk1_nam_2\NetWork\Do_an_socket\server.py • Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

server.py • header.txt • download.html • index.html • info.html •
3 HOST = 'localhost'
4 PORT = 8000
5 NUM_LISTEN = 5
6 SIZE = 1024 * 1024
7 FORMAT = 'utf-8'
8
9 def createSocket(port):
10     s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
11     s.bind((HOST, port))
12     s.listen(5)
13     return s;
14
15 def receive(Client):
16     return Client.recv(SIZE).decode(FORMAT)
17

Line 13, Column 14 Tab Size: 4 Python
```

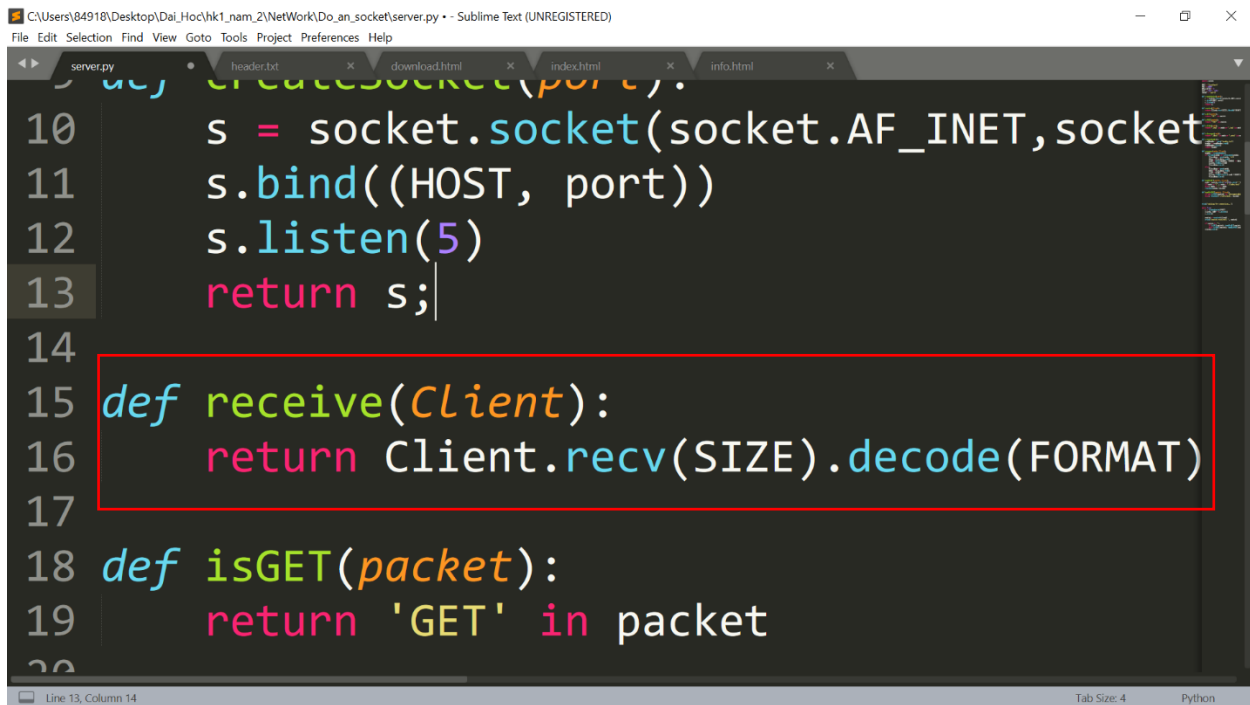
3. Ta khai báo hai biến là client tức là client hiện tại ta đang kết nối và biến addr lưu trữ địa chỉ của client. Sau đó ta đóng socket lại.

```
C:\Users\B4918\Desktop\Dat_Hoc\hk1_nam_2\NetWork\Do_an_socket\server.py • Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

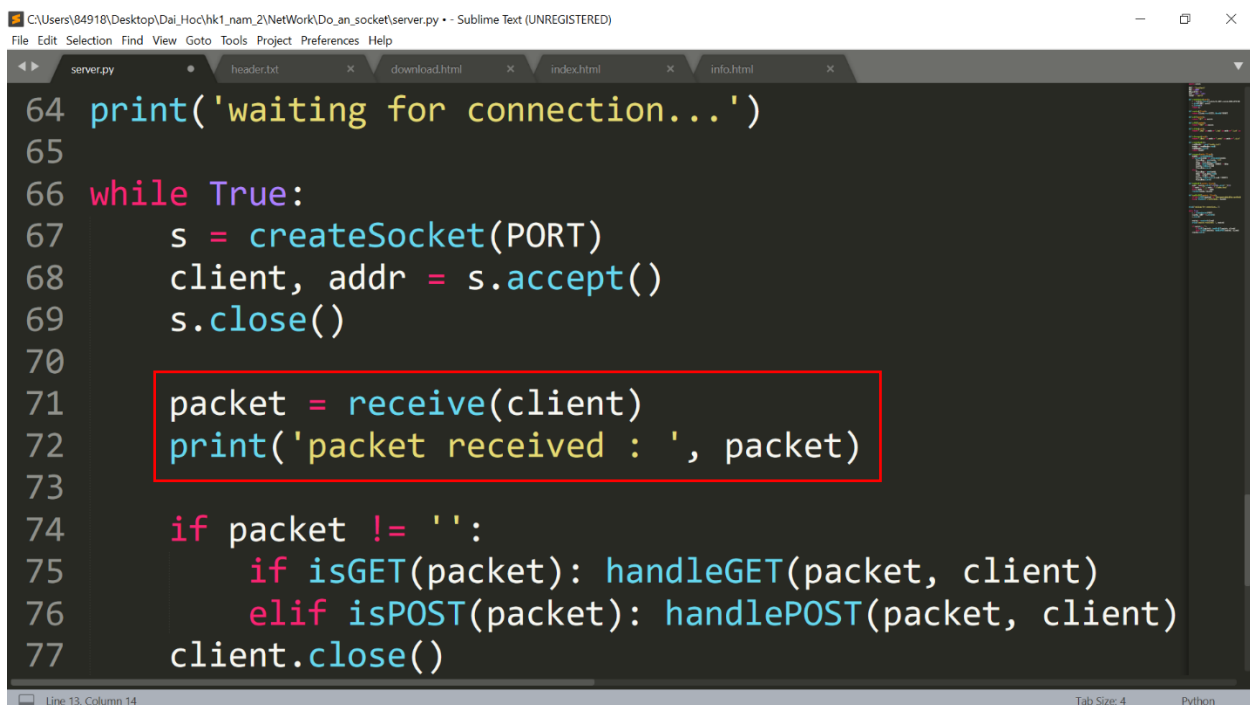
server.py • header.txt • download.html • index.html • info.html •
64 print('waiting for connection...')
65
66 while True:
67     s = createSocket(PORT)
68     client, addr = s.accept()
69     s.close()
70
71     packet = receive(client)
72     print('packet received : ', packet)
73
74     if packet != '':
75         if isGET(packet): handleGET(packet, client)
76         elif isPOST(packet): handlePOST(packet, client)
77     client.close()

Line 13, Column 14 Tab Size: 4 Python
```

- Sau khi chấp nhận kết nối thì Client và Server đã có thể truyền và nhận dữ liệu qua lại. Server sẽ nhận yêu cầu từ Client và lưu vào trong biến packet thông qua hàm `receive(Client)`. Sau đó hiển thị chúng ra màn hình để có thể theo dõi và kiểm tra quá trình trao đổi dữ liệu giữa Client và Server.

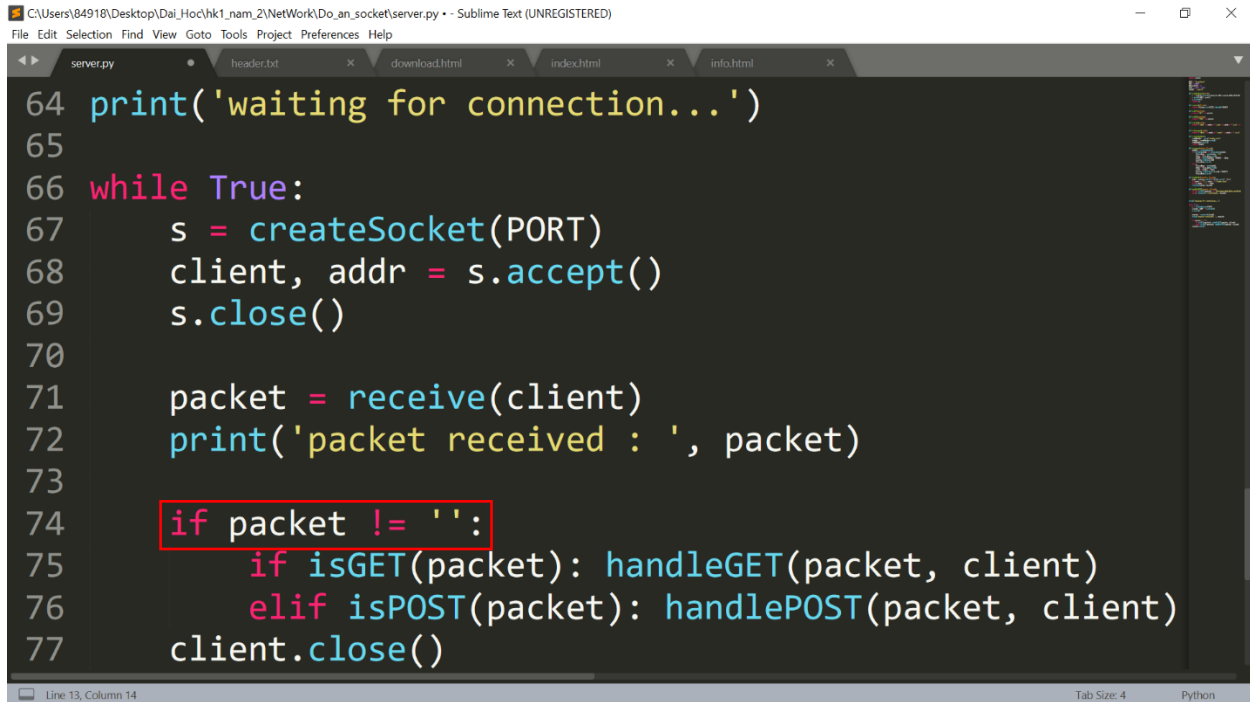


```
server.py
10 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
11 s.bind((HOST, port))
12 s.listen(5)
13 return s;
14
15 def receive(Client):
16     return Client.recv(SIZE).decode(FORMAT)
17
18 def isGET(packet):
19     return 'GET' in packet
```



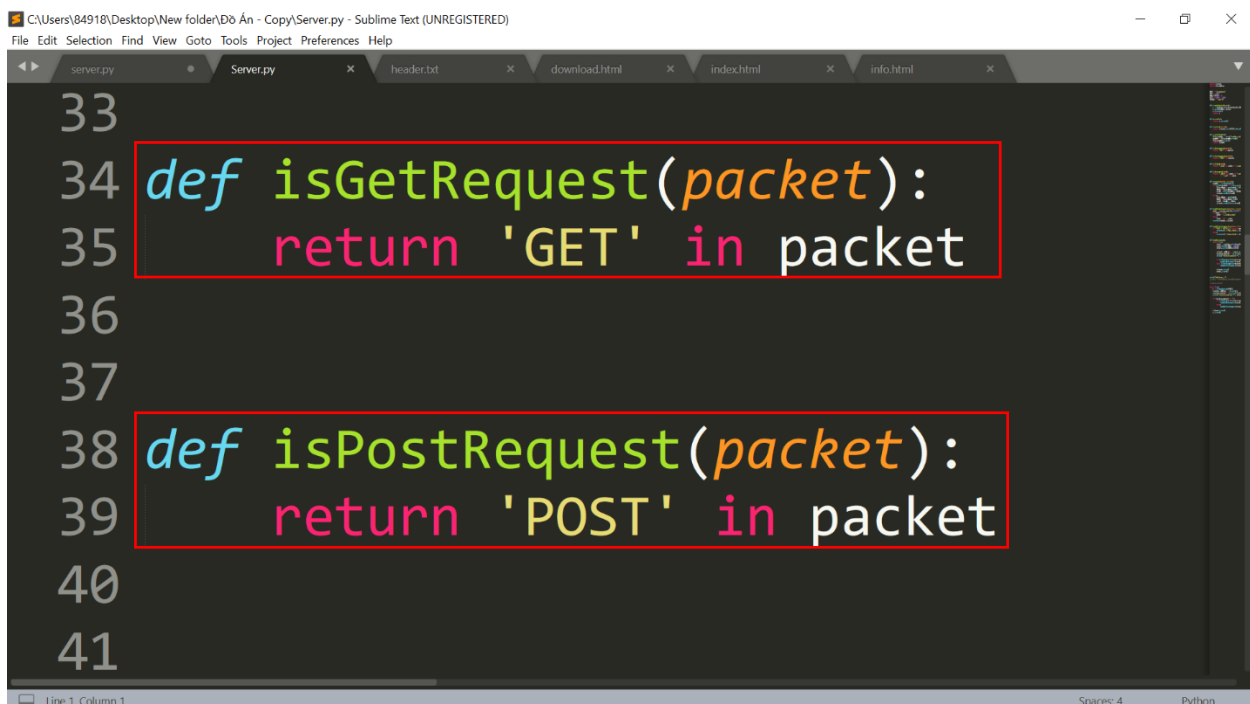
```
server.py
64 print('waiting for connection...')
65
66 while True:
67     s = createSocket(PORT)
68     client, addr = s.accept()
69     s.close()
70
71     packet = receive(client)
72     print('packet received : ', packet)
73
74     if packet != '':
75         if isGET(packet): handleGET(packet, client)
76         elif isPOST(packet): handlePOST(packet, client)
77     client.close()
```

6. Để chắc chắn ta nên kiểm tra packet (tức là dữ liệu nhận được từ client) là không rỗng.



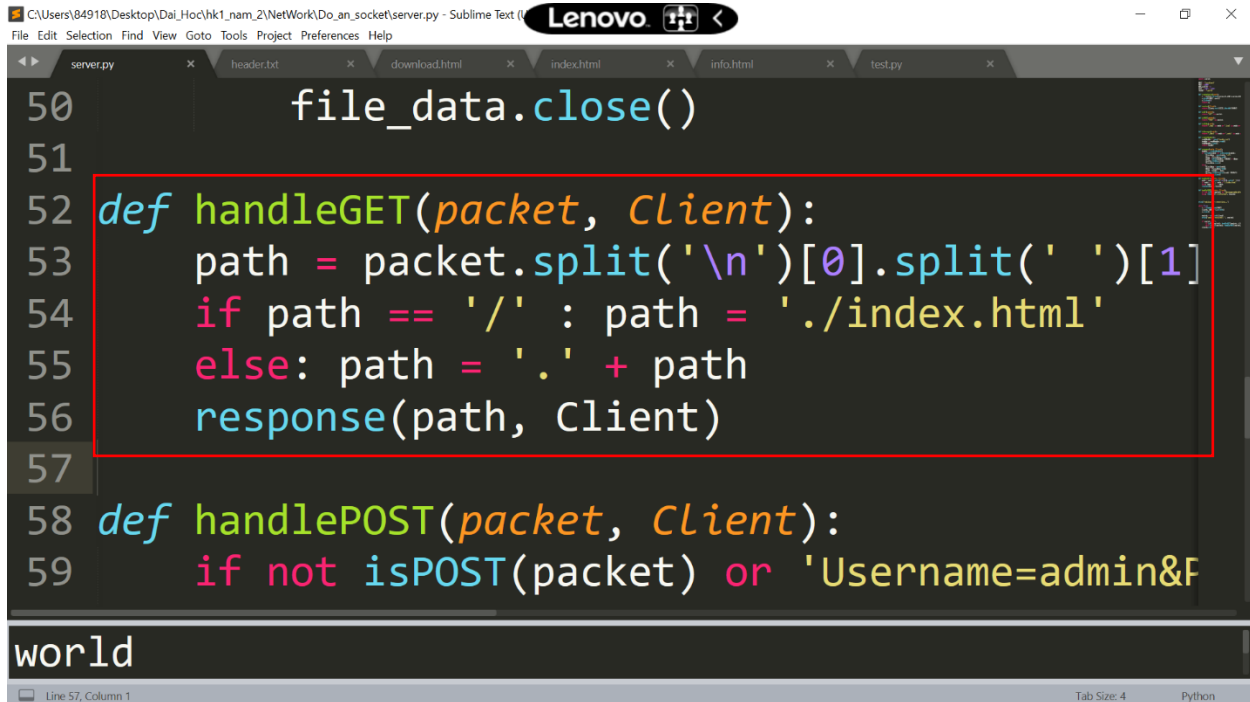
```
64 print('waiting for connection...')
65
66 while True:
67     s = createSocket(PORT)
68     client, addr = s.accept()
69     s.close()
70
71     packet = receive(client)
72     print('packet received : ', packet)
73
74     if packet != '':
75         if isGET(packet): handleGET(packet, client)
76         elif isPOST(packet): handlePOST(packet, client)
77     client.close()
```

7. Ta có 2 phương thức chính là GET và POST. Ta tiến hành kiểm tra xem Client đang yêu cầu theo phương thức nào để ta xử lý theo đúng phương thức đó. Ta viết 2 hàm để kiểm tra xem đó là phương thức GET hay POST.



```
33
34 def isGetRequest(packet):
35     return 'GET' in packet
36
37
38 def isPostRequest(packet):
39     return 'POST' in packet
40
41
```


8. Sau khi biết được Client cần xử được xử lý theo phương thức nào rồi thì ta chỉ cần gọi các hàm để tương ứng là `handleGET()` và `handlePOST()` tương ứng để xử lý yêu cầu.



```
50     file_data.close()
51
52     def handleGET(packet, Client):
53         path = packet.split('\n')[0].split(' ')[1]
54         if path == '/' : path = './index.html'
55         else: path = '.' + path
56         response(path, Client)
57
58     def handlePOST(packet, Client):
59         if not isPOST(packet) or 'Username=admin&F
```

world

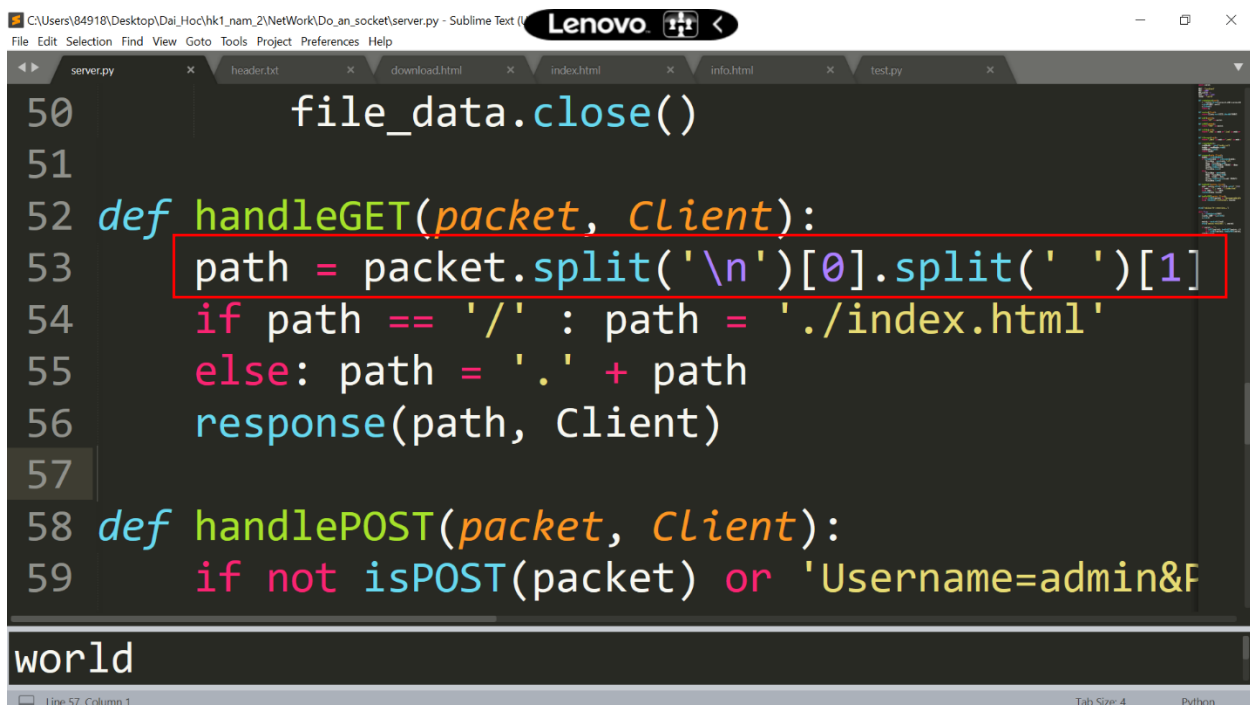


```
53     path = packet.split('\n')[0].split(' ')[1]
54     if path == '/' : path = './index.html'
55     else: path = '.' + path
56     response(path, Client)
57
58     def handlePOST(packet, Client):
59         if not isPOST(packet) or 'Username=admin&Password=admin' not in packet:
60             response('./404.html', Client)
61         else: response('./info.html', Client)
62
63
64
65     print('waiting for connection...')
66
67     while True:
68         s = createSocket(PORT)
69         client, addr = s.accept()
```

world
[Finished in 0.1s]

9. Ta cần biết được Client đang yêu cầu gì để có thể trả về. Do đó ta sẽ bóc tách yêu cầu của Client trong phần Header bằng hàm `split()` có sẵn của Python.

Trong đó: `path = packet.split('\n')[0].split(' ')[1]` ở dòng code này tức là khi ta có data là 1 dòng header với dòng đầu là HTTP / 1.1 200 OK v.v..thì đầu tiên ta chia chuỗi này thành các chuỗi nhỏ sau mỗi dấu '\n' thì khi cắt xong ta được chuỗi như sau : HTTP / 1.1 200 OK tiếp đó từ chuỗi này ta tiếp tục cắt ra thành các chuỗi con ngăn cách nhau bởi dấu ' ' (khoảng trắng). ta được từng phần tử như sau {HTTP, /, 1.1, 200, OK} sau đó lấy phần tử tại vị trí số 1 tức là dấu '/' vậy là ta được phần đường dẫn tới thư mục. ta lưu vào biến `path` để tiện sử dụng. Sau đó tiến hành kiểm tra nếu `path` là '/' tức là khi vừa với vào và kết nối tới server thì ta sẽ thay đường dẫn để client đi đến trang `index.html` còn ngược lại nếu phần ta `split` được không phải '/' mà là `/info.html` thì ta cộng vào đó dấu '.' để biến thành đường dẫn không cố định đến thư mục (vì không phải máy nào cũng có đường dẫn như nhau).



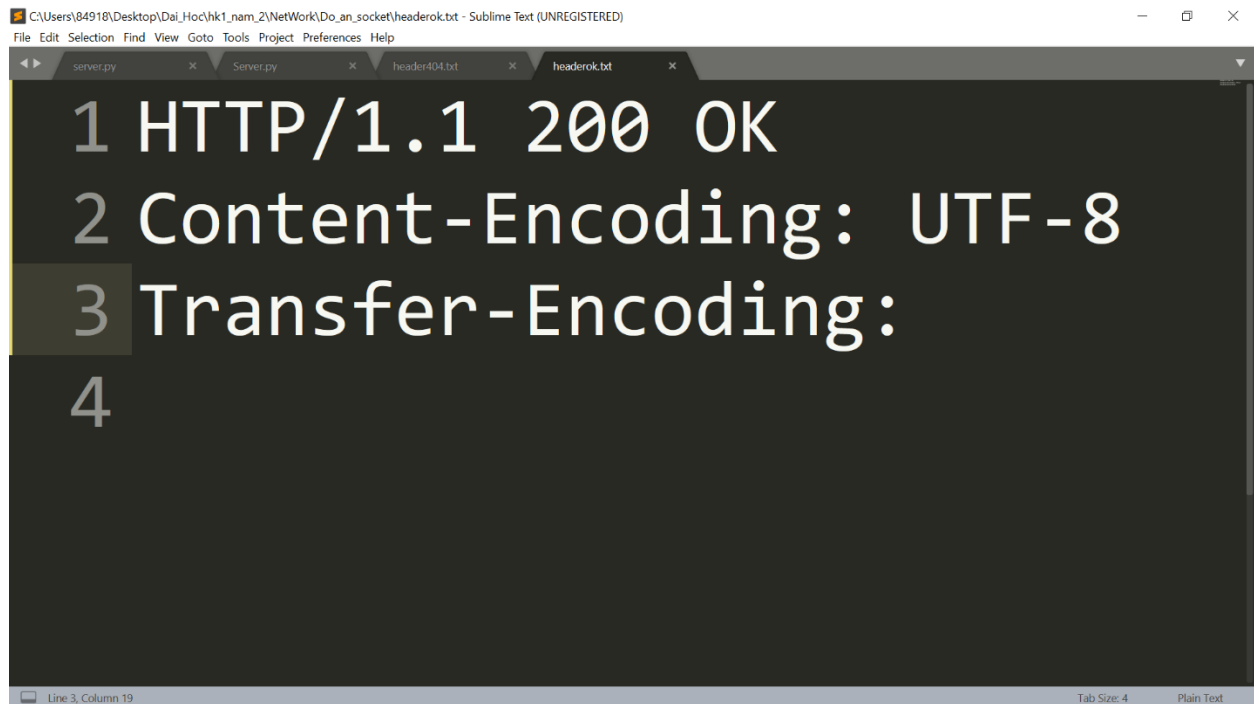
```
50 file_data.close()
51
52 def handleGET(packet, Client):
53     path = packet.split('\n')[0].split(' ')[1]
54     if path == '/': path = './index.html'
55     else: path = '.' + path
56     response(path, Client)
57
58 def handlePOST(packet, Client):
59     if not isPOST(packet) or 'Username=admin&F
```

world

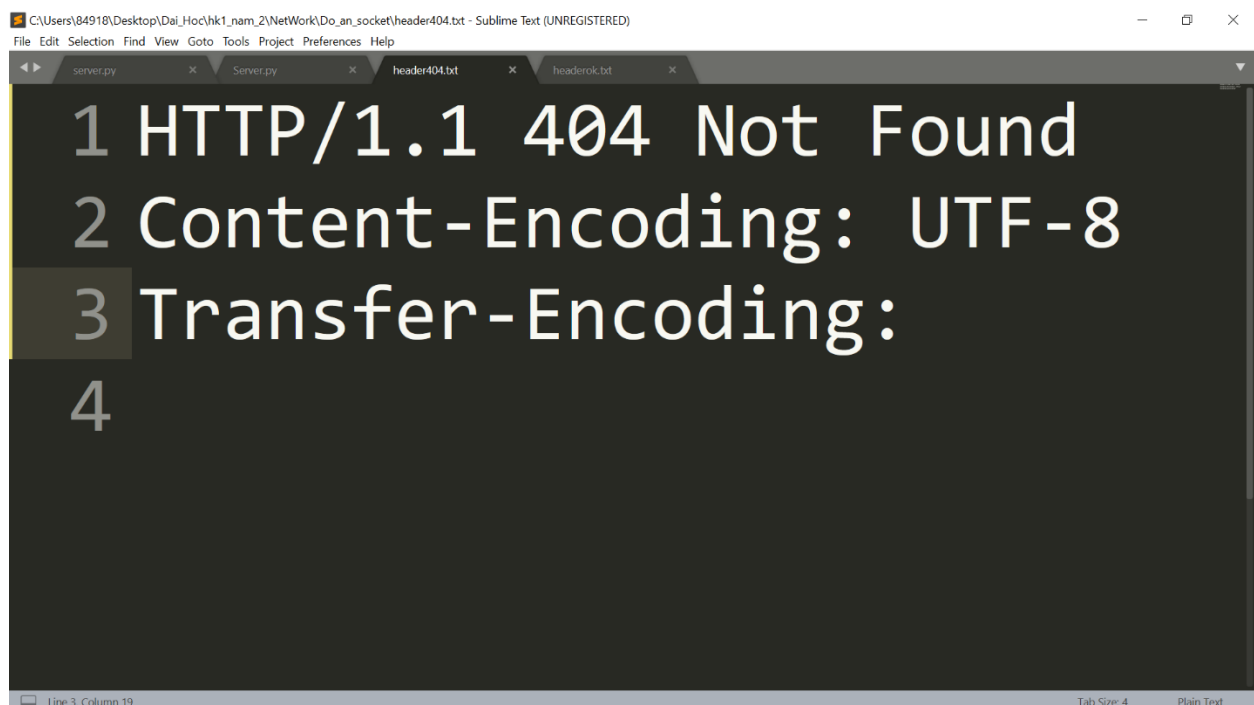
Line 57, Column 1 Python

10. Sau khi xử lý yêu cầu của Client theo đúng phương thức thì ta chỉ việc trả về cho Client bằng hàm `response()`. Trong đó ta cần phải tạo Header bằng hàm `createHeader()`.

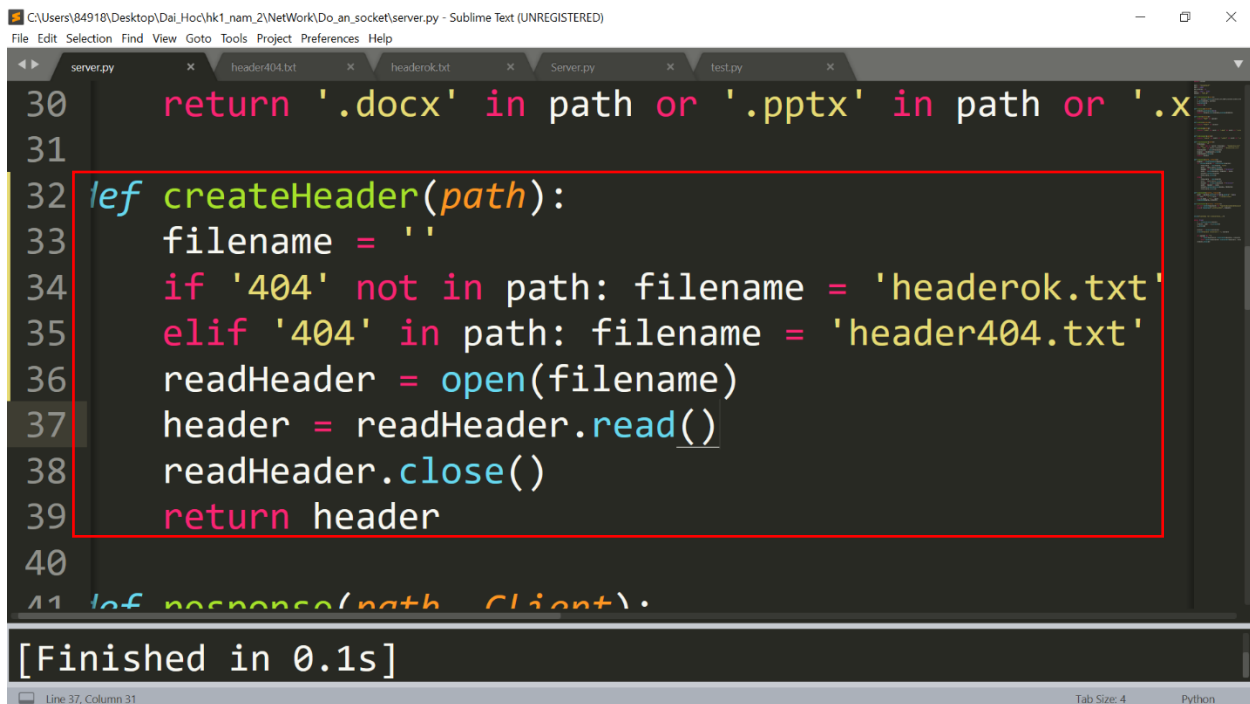
Trong đó: Nếu bị lỗi 404 thì ta trả về header của 404 ngược lại ta trả về header 200 OK (tức là thành công). Nội dung của 2 file như sau:



```
1 HTTP/1.1 200 OK
2 Content-Encoding: UTF-8
3 Transfer-Encoding:
4
```



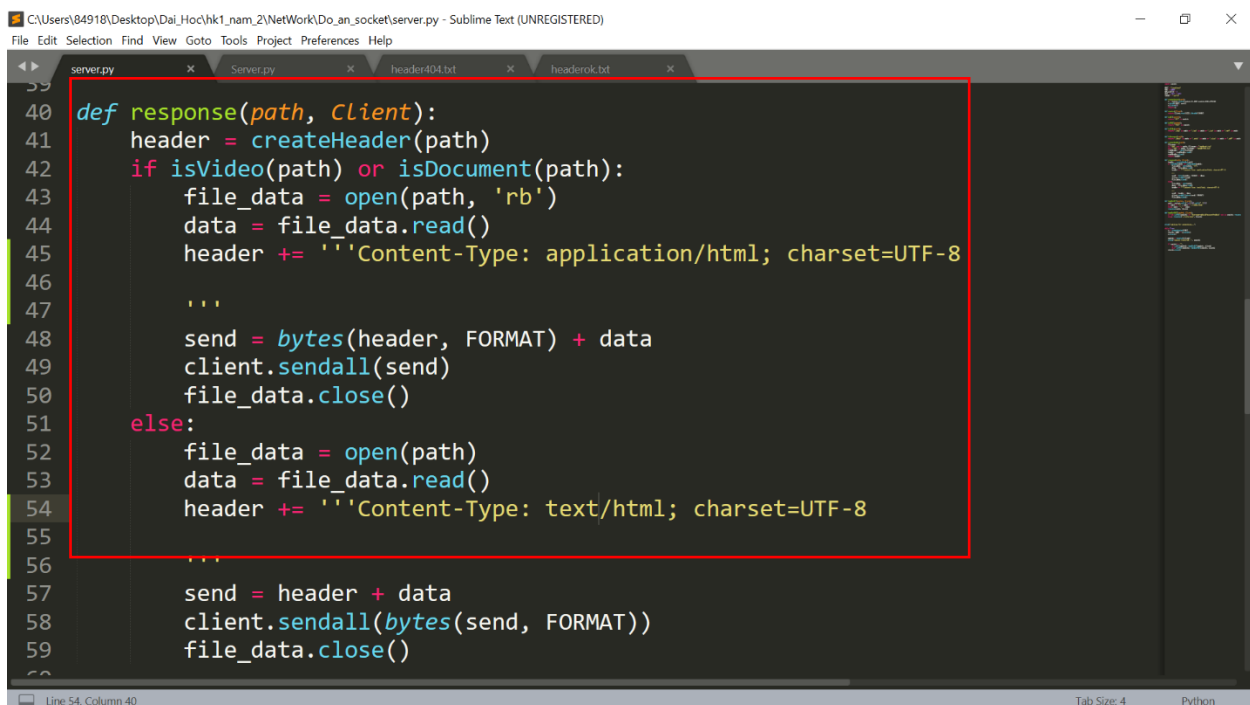
```
1 HTTP/1.1 404 Not Found
2 Content-Encoding: UTF-8
3 Transfer-Encoding:
4
```



```
30     return '.docx' in path or '.pptx' in path or '.x
31
32 def createHeader(path):
33     filename = ''
34     if '404' not in path: filename = 'headerok.txt'
35     elif '404' in path: filename = 'header404.txt'
36     readHeader = open(filename)
37     header = readHeader.read()
38     readHeader.close()
39     return header
40
41 def response(path, Client):
```

[Finished in 0.1s]

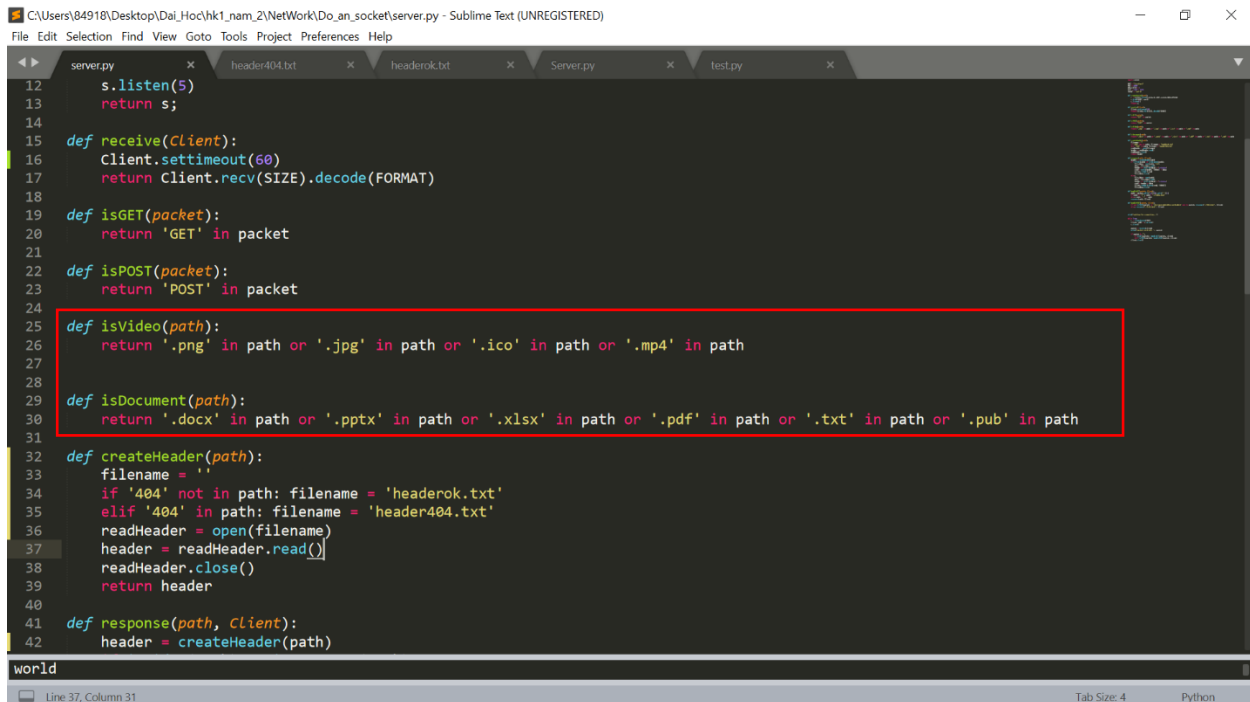
Line 37, Column 31



```
40 def response(path, Client):
41     header = createHeader(path)
42     if isVideo(path) or isDocument(path):
43         file_data = open(path, 'rb')
44         data = file_data.read()
45         header += '''Content-Type: application/html; charset=UTF-8
46
47         '''
48         send = bytes(header, FORMAT) + data
49         client.sendall(send)
50         file_data.close()
51     else:
52         file_data = open(path)
53         data = file_data.read()
54         header += '''Content-Type: text/html; charset=UTF-8
55
56         '''
57         send = header + data
58         client.sendall(bytes(send, FORMAT))
59         file_data.close()
```

Line 54, Column 40

- Khi gửi dữ liệu về Client thì ta cũng cần để ý loại file sẽ được gửi để có thể đọc file đó 1 cách thích hợp. Ta thiết kế 2 hàm để đảm nhận việc nhận biết loại file nào cần đọc theo kiểu Binary hay kiểu text bình thường.



```
server.py
header404.txt
headerok.txt
Server.py
test.py

12 s.listen(5)
13 return s;
14
15 def receive(Client):
16     Client.settimeout(60)
17     return Client.recv(SIZE).decode(FORMAT)
18
19 def isGET(packet):
20     return 'GET' in packet
21
22 def isPOST(packet):
23     return 'POST' in packet
24
25 def isVideo(path):
26     return '.png' in path or '.jpg' in path or '.ico' in path or '.mp4' in path
27
28
29 def isDocument(path):
30     return '.docx' in path or '.pptx' in path or '.xlsx' in path or '.pdf' in path or '.txt' in path or '.pub' in path
31
32 def createHeader(path):
33     filename = ''
34     if '404' not in path: filename = 'headerok.txt'
35     elif '404' in path: filename = 'header404.txt'
36     readHeader = open(filename)
37     header = readHeader.read()
38     readHeader.close()
39     return header
40
41 def response(path, Client):
42     header = createHeader(path)
```

world

Line 37, Column 31

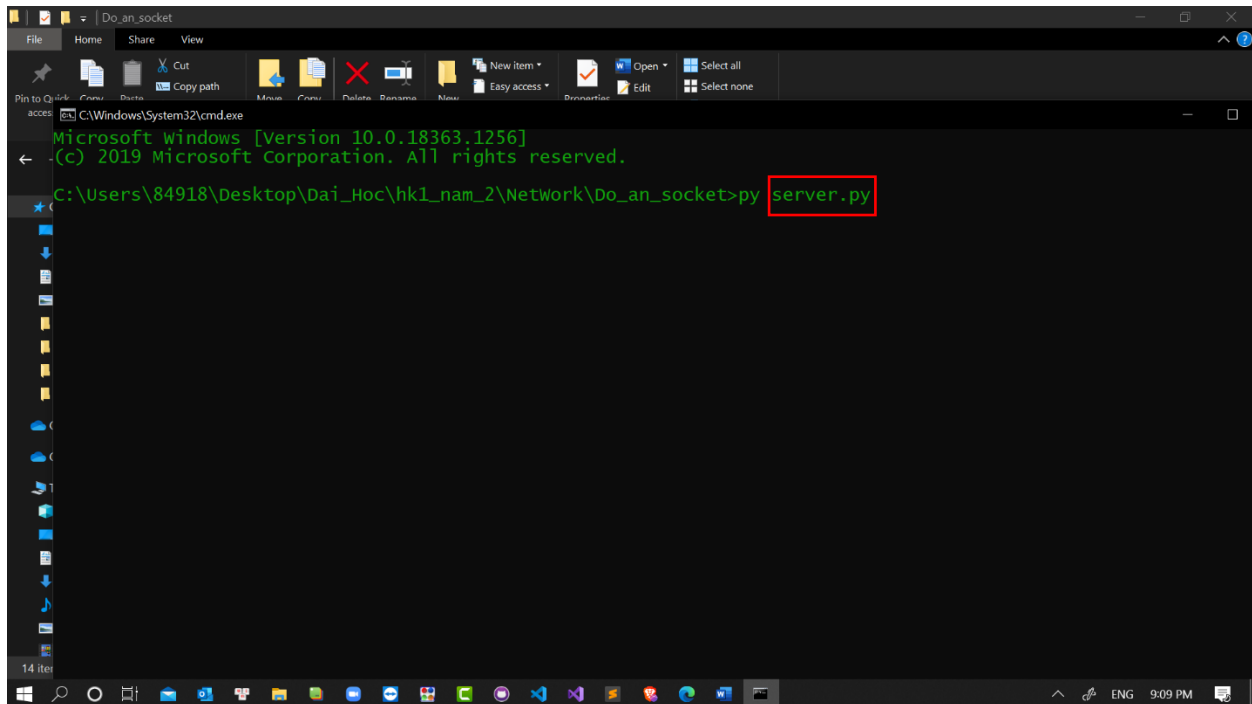
Tab Size: 4 Python

11. Sau khi đã có đầy đủ Header và phần Data thì ta chỉ cần gửi về lại Client là xong.

DEMO CÁC BƯỚC THỰC HIỆN

Bước 1: Chạy Server.

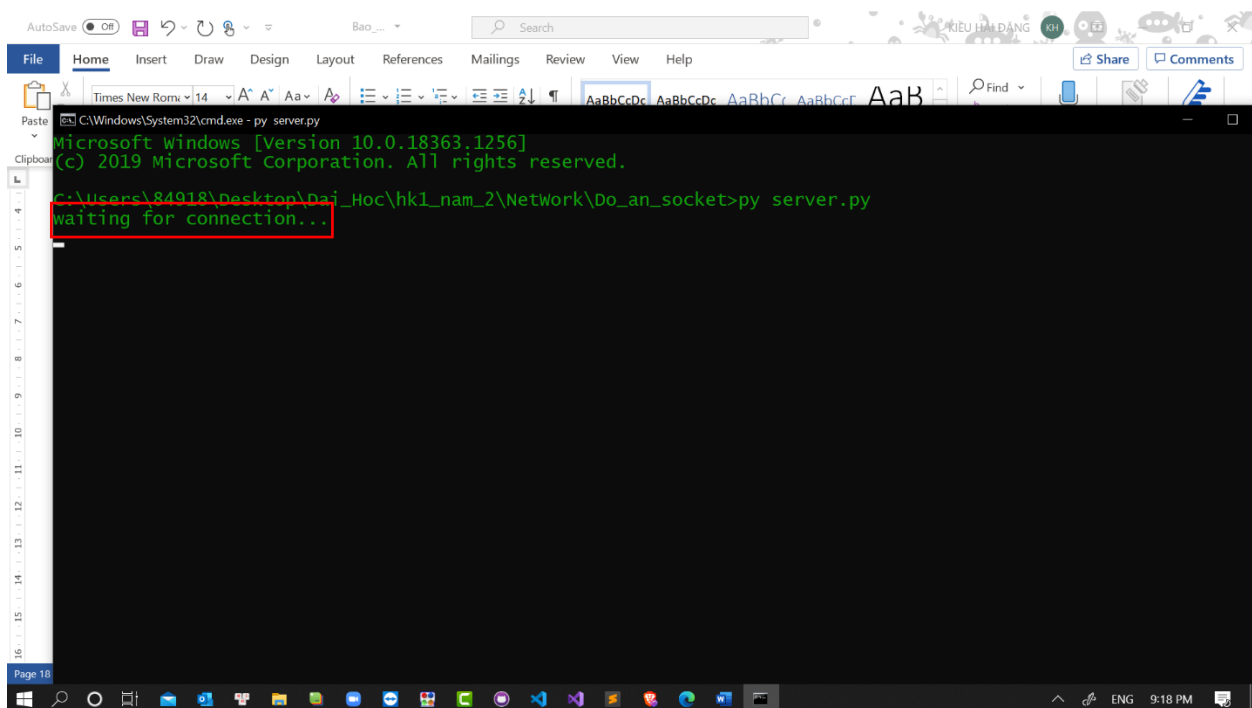
- Đầu tiên ta truy cập vào thư mục chứa file server.py bật cmd lên và gõ lệnh py server.py.



```
Microsoft Windows [Version 10.0.18363.1256]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\84918\Desktop\Dai_Hoc\hk1_nam_2\Network\Do_an_socket>py server.py
```

- Thấy hiển thị dòng “waiting for connection...” như hình bên dưới tức là Server đã chạy thành công và đang chờ Client kết nối tới.

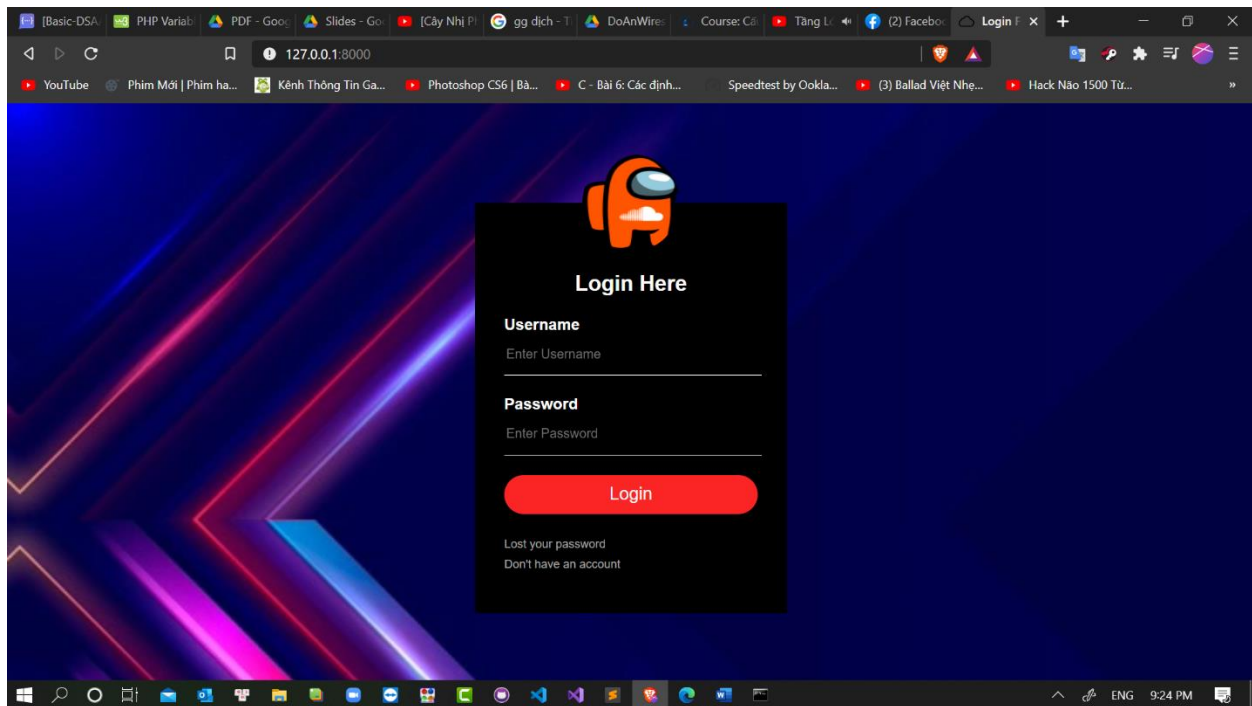


```
Microsoft Windows [Version 10.0.18363.1256]
(c) 2019 Microsoft Corporation. All rights reserved.

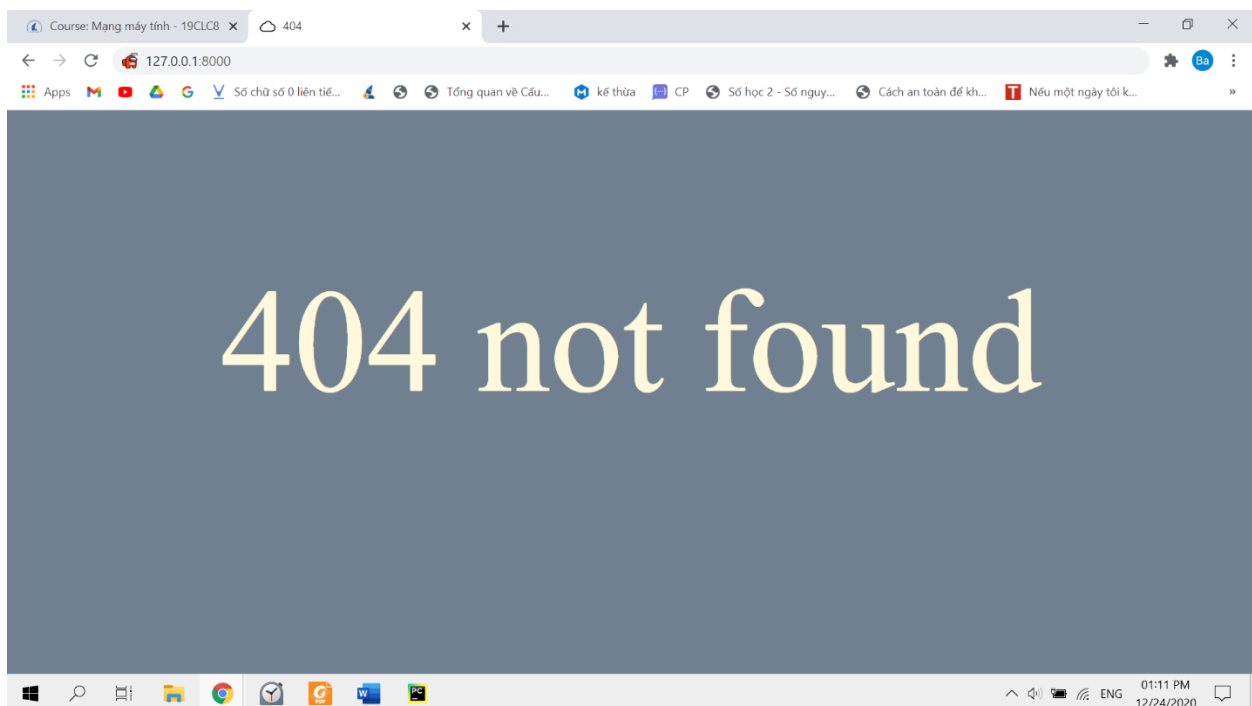
C:\Users\84918\Desktop\Dai_Hoc\hk1_nam_2\Network\Do_an_socket>py server.py
waiting for connection...
```

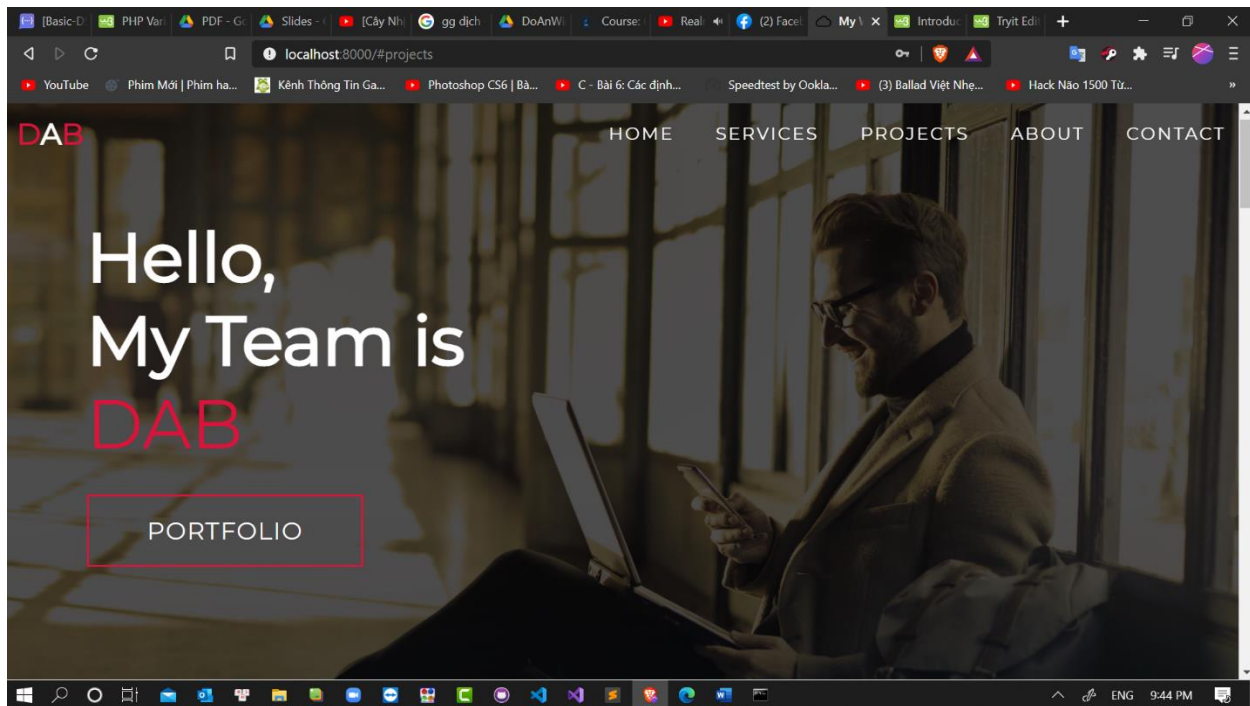
Bước 2: Ở phía Client, ta bật trình duyệt lên và truy cập đường link:

127.0.0.1:8000 để truy cập và tạo kết nối tới Server. Sau đó trình duyệt sẽ hiển thị trang web như hình bên dưới tức là 2 bên đã trao đổi dữ liệu thành công.

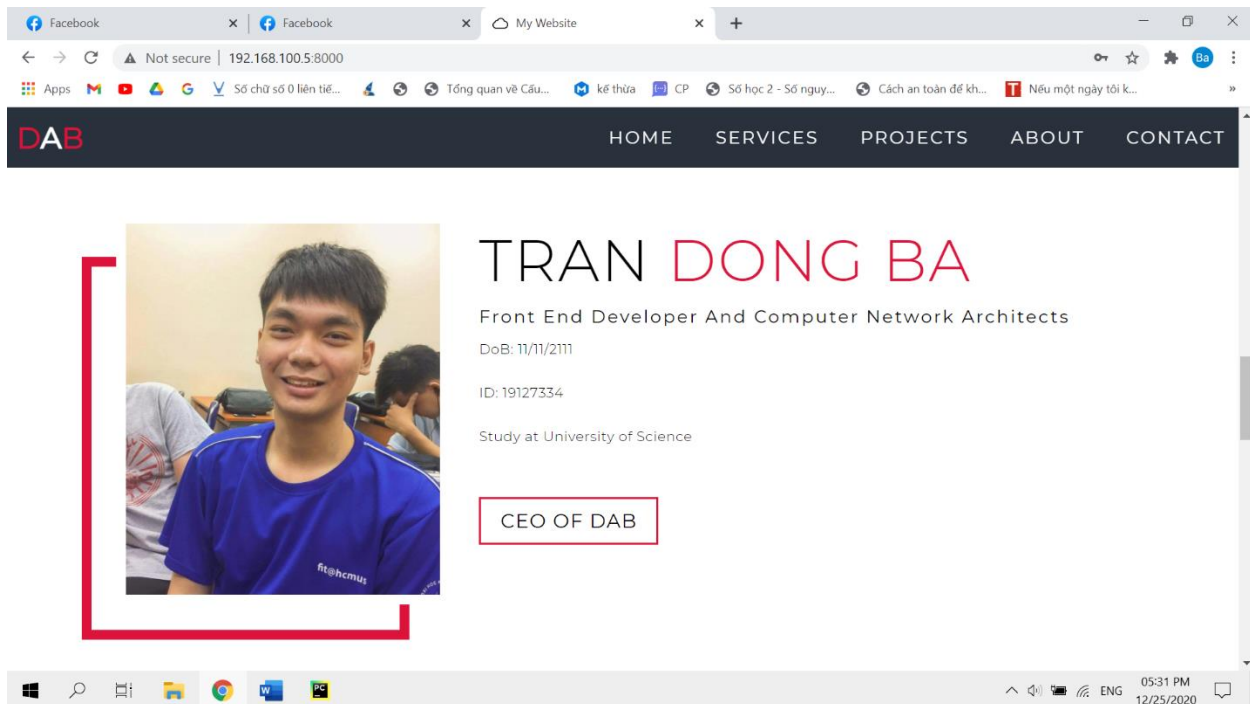


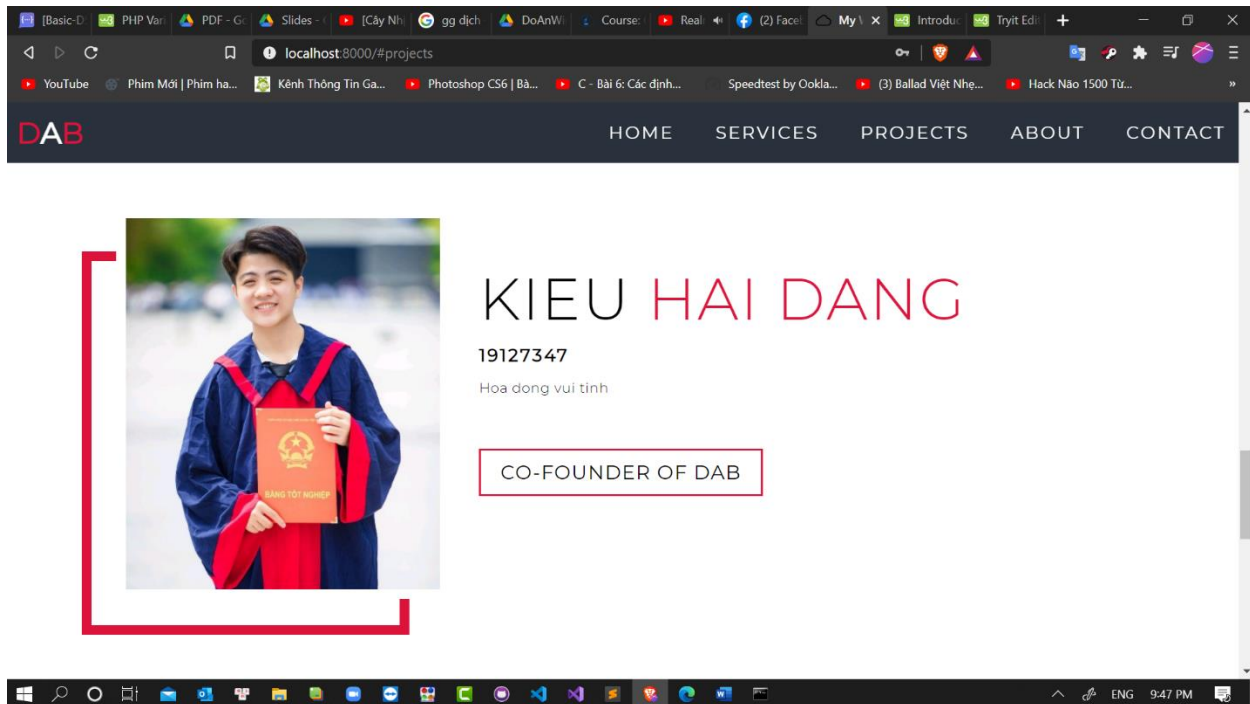
Bước 3: Tiến hành đăng nhập vào trang web. Username: admin, Password: admin. Nếu nhập sai 1 trong 2 thì sẽ hiện thị ra trang thông báo lỗi 404, ngược lại sẽ hiện thị ra trang web của nhóm.



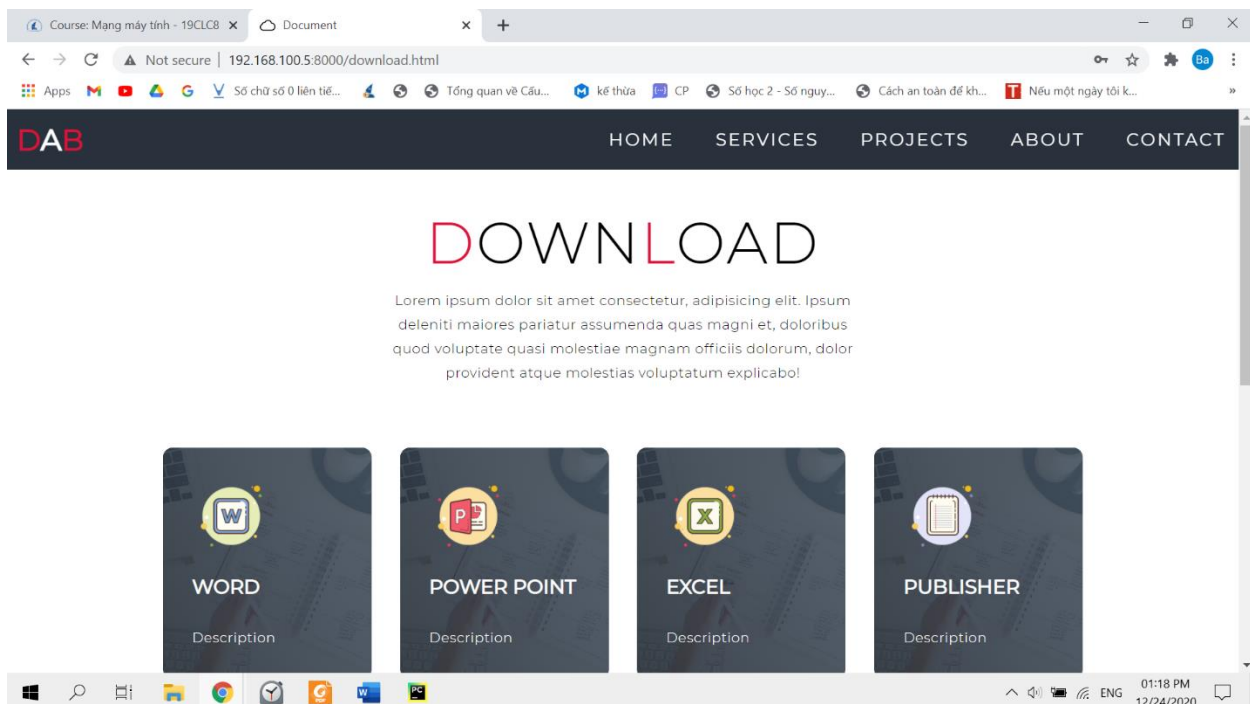


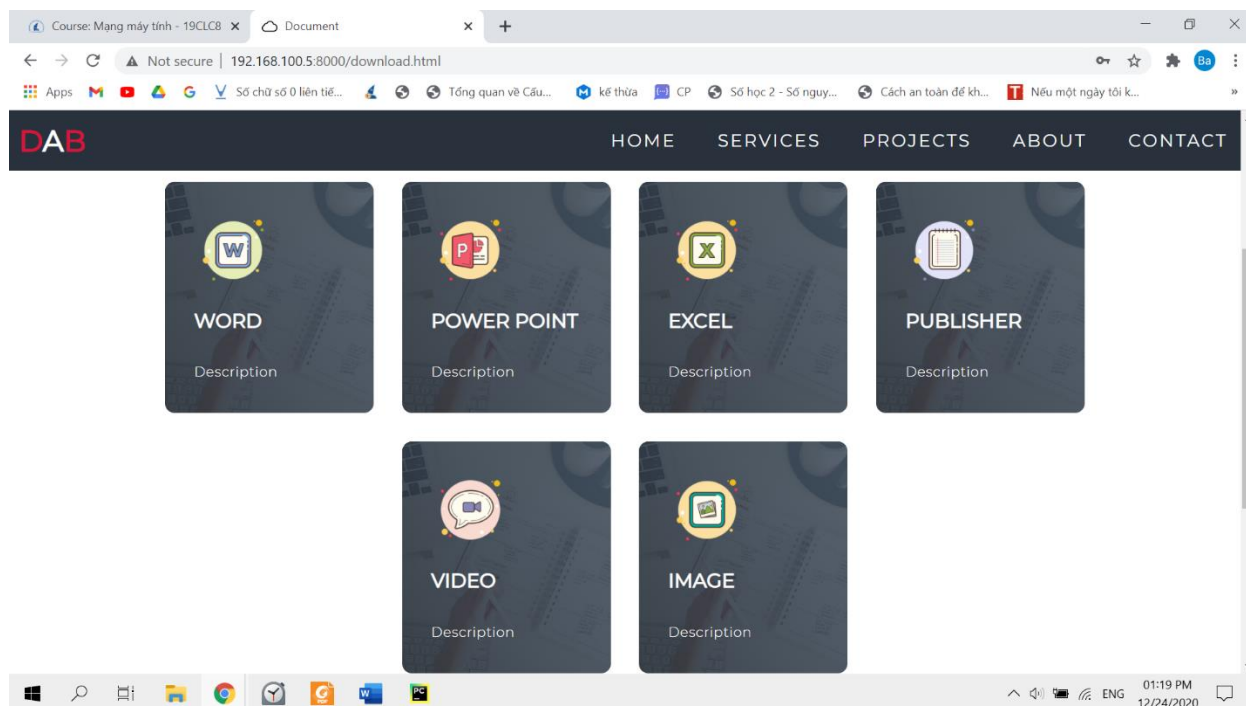
Bước 4: Click vào phần ABOUT trình duyệt sẽ cuộn xuống phần thông tin nhóm.





Bước 5: Ta Click vào phần Services thì sẽ hiện ra trang Download và ở đó chứa các file có thể download được.





Bước 6: Ta chỉ cần Click vào tên file cần download thì trình duyệt sẽ download về.

