

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN

LẬP TRÌNH SOCKET: XÂY DỰNG WEB SERVER ĐƠN GIẢN

Môn học: Mạng máy tính

GVLT: Lê Giang Thanh

GVTH: Lê Hà Minh

Sinh viên thực hiện: Nguyễn Thị Thu Hằng – 18120027

Nguyễn Xuân Mai – 18120056

Thành phố Hồ Chí Minh – Tháng 7/2020

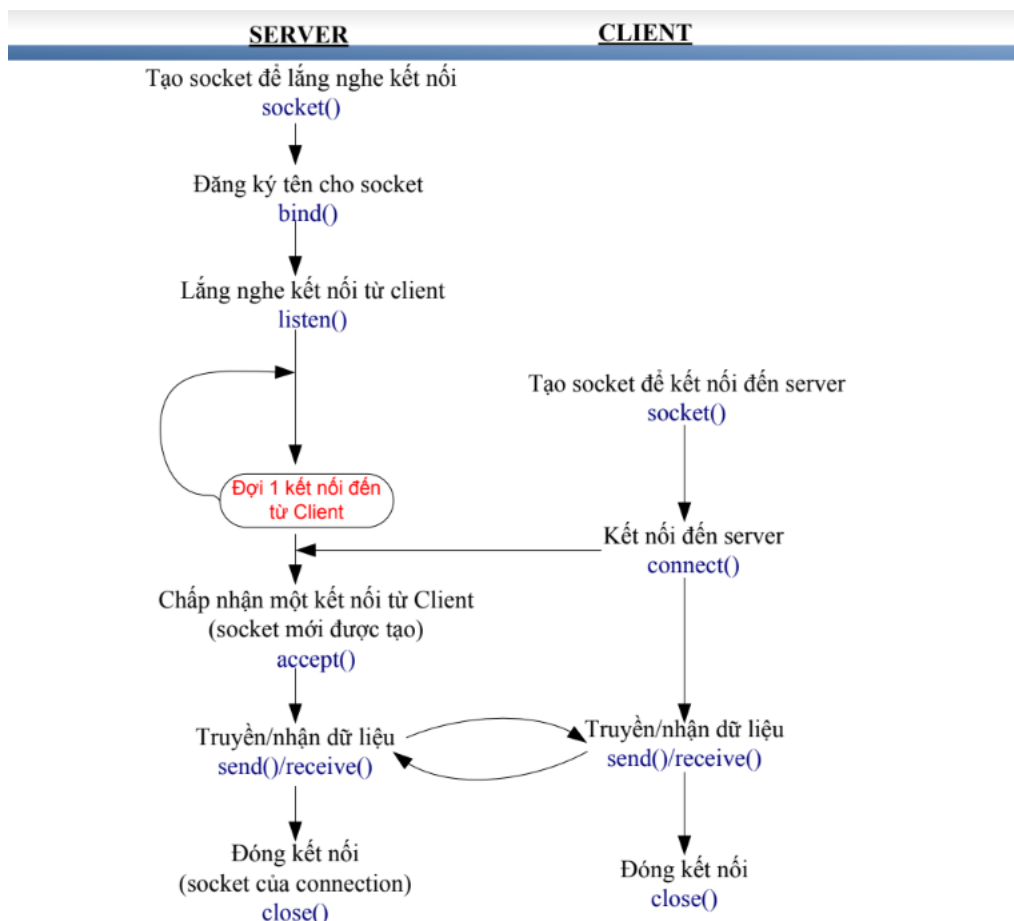
MỤC LỤC

1. Thông tin thành viên	3
2. Giải thích cách thực hiện ứng dụng.....	3
2.1. Tổng quan.....	3
2.2. Chi tiết hàm.....	4
3. Test ứng dụng	9
4. Tham khảo	11

1. Thông tin thành viên

Họ tên	MSSV	Ghi chú
Nguyễn Thị Thu Hằng	18120027	
Nguyễn Xuân Mai	18120056	

2. Giải thích cách thực hiện ứng dụng



Sơ đồ miêu tả mô hình TCP socket (Nguồn: Slide tăng ứng dụng CDIO – FIT)

2.1. Tổng quan

- Socket: “Cánh cửa” giữa ứng dụng và giao thức tầng transport (TCP, UDP) – đồ án này sử dụng TCP.
- Tìm hiểu 2 method của HTTP Protocol trong gói Request message sử dụng trong đồ án này:

- GET method: được sử dụng để lấy thông tin từ server theo URI đã cung cấp.
 - POST method: gửi thông tin tới server (ở đề án này là login form trong *index.html* file bao gồm username và password)
- Tìm hiểu về status code của HTTP Protocol trong gói Response message sử dụng trong đề án này:
- Mã trạng thái
 - Một số nguyên 3 ký tự, trong đó ký tự đầu tiên định nghĩa loại response và 2 ký tự cuối không có bất cứ vai trò nào.
 - Đề án này sử dụng:
 - 200 – Thành công: request được server tiếp nhận, hiểu và xử lý thành công
 - 404 – Lỗi Client: request chứa cú pháp không chính xác hoặc không thực hiện được.

2.2. Chi tiết hàm

- Tạo một socket với 2 đối số
- Socket.AF_INET: một socket Internet Protocol (IP), cụ thể ở đây là IPv4.
 - Socket.SOCK_STREAM: đây là TCP socket.

```
#create a TCP/IP socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

- Đăng ký tên cho socket, ràng buộc địa chỉ vào socket

```
#bind the socket to the port
s.bind(('', PORT))
```

- Cho socket đang lắng nghe tới tối đa ‘_connection’ kết nối.

```
#listening for maximum number of queue connection
s.listen(_connection)
```

- Câu lệnh này lấy địa chỉ của folder hiện tại chứa file .py đang chạy. Trước khi sử dụng phải import thư viện ‘os’

```
#get the directory of __file__ (where __file__ is this .py file - main.py)
cur_path = os.path.dirname(__file__)
print('Server is listening on PORT: ', PORT)
```

- Tạo một vòng lặp mãi mãi cho đến khi bị gián đoạn bởi 1 tác động hay có lỗi xảy ra.

- Khi một client gõ cửa, server chấp nhận kết nối và 1 socket mới được tạo ra. Client và server bây giờ hoàn thành bắt tay, tạo ra 1 TCP connection, client và server bây giờ đã có thể truyền và nhận dữ liệu với nhau.

```
#establish connection with client
client, adr = s.accept()
print('Client connected:', client)
```

- Nhận gói dữ liệu.

```
#received data
data = client.recv(_bytes)
```

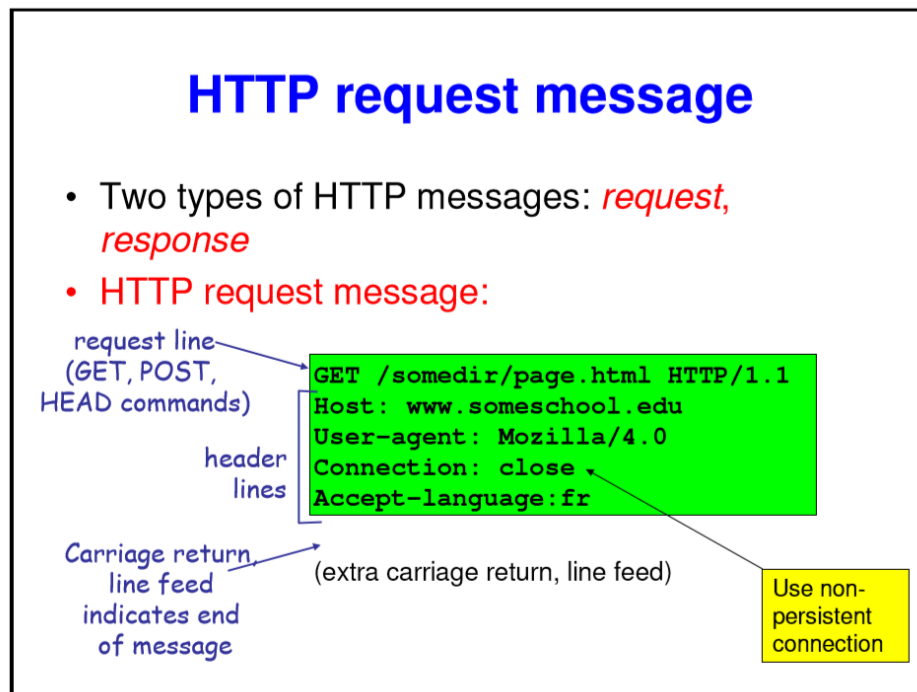
- Phân tích gói requests data vừa nhận

```
#analys request package
requests = data.decode()
print(requests)
```

- Lấy ra method của data.

```
#requests method (at this project we use only GET and POST method)
req_method = requests.split(' ')[0]
```

- Do cấu trúc của gói request như sau:



Nguồn: Slides_12_http.

Sau khi decode, dùng hàm *split* để tách thành 1 mảng dựa trên khoảng trắng và sau đó lấy phần tử đầu tiên. Ta được request method.

- Tiếp theo đọc file index.html (nơi chứa login form của chương trình được viết bằng HTML và CSS). Câu lệnh bên dưới nối tên file vào trong đường dẫn lấy được lúc đầu và đọc file

```
#response path package
res_path = os.path.join(cur_path, 'index.html')    #link to index.html file
status_code, data = read_file(res_path)           #data: data in index.html file
```

- Hàm đọc file:

```
#read file function
def read_file(filename):
    try:
        file_open = open(filename, 'rb')
        data = file_open.read()
        file_open.close()
        response_code = 200

    except FileNotFoundError:
        print('File Not Found')
        response_code = 404
        data = ""

    return response_code, data
```

- Phân tích gói dữ liệu để tìm ra username và password nếu là method “POST”

```
#check method
if req_method == 'GET' or req_method == 'POST':
    if req_method == 'POST':
        #analys requests package to find username and password to check
        usname, psword = analysis_data(requests)
```

- Chi tiết hàm *analysis_data*: hàm dùng để lấy ra username và password được lưu trong data của gói request.

Request Message for POST

```
POST cgi-bin/create.p1 HTTP 1.1
Host: xpto.dei.uc.pt
Accept: image/gif, image/x-xbit, image-jpeg, image/pjpeg, */
Content-type: application/x-www-form-urlencoded
Content-length: 37
```

```
user=joe&pass1=1234&pass2=1234
```

Data is sent to the server inside the HTML message.

Nguồn: Slides_12_http

Thử nhập *username* và *password* là *admin* thì data trả về như sau. Sau đó ta dễ dàng tìm ra được *username* và *password* trong hàm *analysis_data*

```
referer: http://localhost:4000/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9

username=admin&password=admin
```

- Sau khi tìm được, so sánh với yêu cầu của đề, nếu đúng chạy vào đọc file *info.html*

```
if usname == 'admin' and psword == 'admin':
    #if true then path now is info.html
    res_path = os.path.join(cur_path, 'info.html')
```

- Nếu không đúng thì chạy vào đọc file *error 404.html*

```
else:
    #if not true
    status_code = 404
    res_path = os.path.join(cur_path, '404.html')
```

- Nếu là GET, đọc từng file của *info.html*

```

elif req_method == 'GET':
    stop += 1

    #take file_path from requests package
    file_path = get_filepath(requests)

    if file_path.find('.html') == -1:
        res_path = os.path.join(cur_path, file_path)

dr=('127.0.0.1', 5192)>
GET /background.jpg HTTP/1.1
Host: localhost:4000

```

- Hàm *get_filepath* lấy tên file ghép với đường dẫn của thư mục ban đầu *cur_path*

```

def get_filepath(req_pack):
    filepath = req_pack.split(' ')[1]      #file name is index 1 near get/post method in http request package
    filepath = filepath[1:]                #erase '/'

    if filepath == '':
        filepath = 'index.html'

    return os.path.join(os.path.dirname(__file__), filepath)

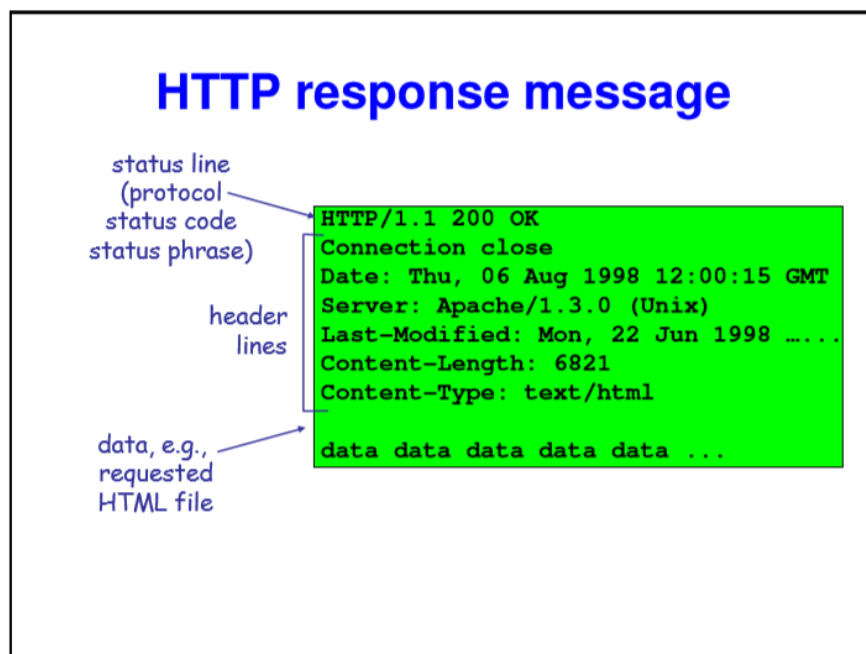
```

- Tạo gói response để trả về:

```

#create responses package to send back
res_header, res_body = create_response(status_code, data)

```



- Chi tiết hàm *create_response*:

```
#create a response package to send
def create_response(status, data):
    if status == 200:
        status_code = 'HTTP/1.1 200 OK\r\n'
    elif status == 404:
        status_code = 'HTTP/1.1 404 NOT FOUND\r\n'

    header = 'Connection: close\r\n'
    header += 'Accept: text/html\r\n'
    header += 'Accept-Language: en_US\r\n'
    header += 'Content-Type: text/html\r\n\r\n'

    res_header = status_code + header
    return res_header, data
```

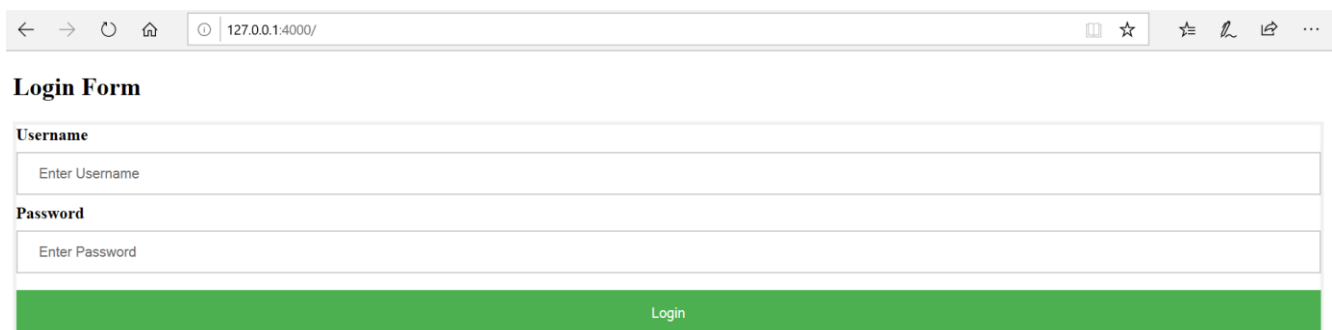
- Gửi lại gói response.

```
#sended
client.send(res_header.encode())
client.send(res_body)
client.close()
```

- Đóng kết nối.

```
except KeyboardInterrupt:
    s.close()
    sys.exit(0)
```

3. Test ứng dụng



The screenshot shows a web browser window with the address bar displaying "127.0.0.1:4000/". The page title is "Login Form". Below the title, there are two input fields: "Username" and "Password". The "Username" field has a placeholder text "Enter Username". The "Password" field has a placeholder text "Enter Password". At the bottom of the form, there is a green button labeled "Login".

Màn hình sau khi kết nối

← → ↻ 🏠 ⓘ 127.0.0.1:4000/ ⓘ ☆ ⚙️ 🔍 📄 ...

Login Form

Username

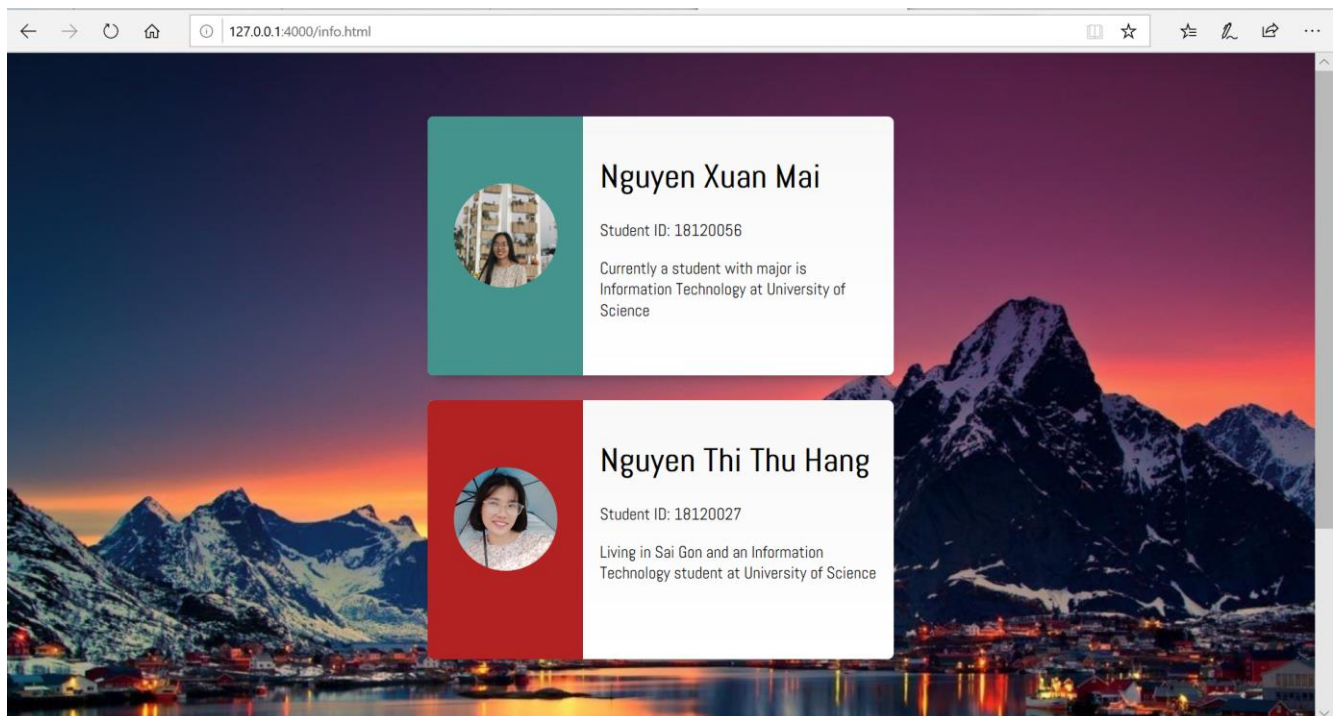
admin

Password

•••••

Login

Nhập username và password đúng



Đây là màn hình trả về nếu nhập đúng

← → ↻ 🏠 ⓘ 127.0.0.1:4000/ ⓘ ☆ ⚙️ 🔍 📄 ...

Login Form

Username

18120012

Password

••••

Login

Thử nhập sai một username và password khác

404

Page not found

Đây là màn hình trả về nếu nhập sai

4. Tham khảo

- Slide Chương 3 – tầng ứng dụng – CDIO, trường đại học Khoa học tự nhiên
- Slides_12_http – references.txt – folder đề bài
- Sách Computer Networking – A top down approach – 6th edition
- Github