ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN 02 MÔN MẠNG MÁY TÍNH

Đề tài:

SOCKET – XÂY DỰNG WEB SERVER ĐƠN GIẢN

Người thực hiện:

Lê Hạnh Linh (18120052)

Trương Trọng Lộc (18120197)

Giảng viên hướng dẫn:

Thầy Lê Hà Minh

Cô Chung Thùy Linh

Thành phố Hồ Chí Minh – Tháng 07, 2020

MỤC LỤC

1. MỤC TIÊU ĐỒ ÁN:

Khi thực hiện đồ án này, nhóm chúng em đề ra các mục tiêu sau đây:

- Hiểu được tổng quan về HyperText Transfer Protocol (HTTP). Cụ thể về nguyên lý hoạt động, các dạng thức thông điệp HTTP (thông điệp yêu cầu, thông điệp phản hồi); một số mã trạng thái và cụm từ thông thường trong thông điệp phản hồi (200 OK, 301 Move Permanently, 404 Not Found); các phương thức HTTP: GET, POST.
- Xây dựng được Web Server đơn giản bằng lập trình Socket qua giao thức HTTP:
    - Trả về được nội dung các trang HTML.
    - Điều hướng được trang web đến nơi mong muốn khi gửi dữ liệu phù hợp.
    - Cho phép tải (Download) các files thông qua trình duyệt Web.
    - Cho phép hiển thị hình ảnh, một số file thông dụng trên trình duyệt Web.
- Có kiến thức về lập trình Socket bằng ngôn ngữ lập trình Java.
- Biết cách sử dụng, tạo các file HTML đơn giản.

2. MÔI TRƯỜNG LẬP TRÌNH:

- Chúng em sử dụng ngôn ngữ lập trình Java để thực hiện đồ án này.
- IDE lập trình: IntelliJ IDEA, Edition: 2020.1.2 cho Windows.
- Java Development Kit (64-bit) 8.0.2510.8.
- Lí do chọn ngôn ngữ lập trình Java:
    - Là ngôn ngữ mà chúng em chưa được tiếp cận. Đây là cơ hội để chúng em trải nghiệm và học hỏi thêm ngôn ngữ lập trình.
    - Giữa C++ và Python, Java là ngôn ngữ lập trình ở mức giữa (theo cảm nhận của chúng em). Vì Python hỗ trợ người dùng cách tối đa, nghĩa là các thư viện có sẵn phong phú, còn C++ thiên về tư duy (ít có thư viện hỗ trợ hơn). Theo chúng em, Java là sự hòa trộn giữa C++ và Python, cung cấp một số thư viện cần thiết để phục vụ cho đồ án, đồng thời vận dụng được tư duy của người lập trình trong một số công việc xử lý trong đồ án (nếu Python thì có thư viện hỗ trợ các công việc này).

3. NHẮC LẠI VỀ GIAO THỨC HTTP:

3.1. Tổng quan về HTTP:

- Là giao thức tầng Ứng dụng của Web.

- Sử dụng mô hình khách – chủ (client – server), trong đó:

  - Client: Web Browser, ví dụ Google Chrome, Cốc Cốc, Firefox…: gửi yêu cầu cho Server và nhận phản hồi từ Server.
  - Server: Web Server: gửi phản hồi cho Client khi nhận được yêu cầu (Request).

- Sử dụng TCP như là một giao thức truyền thông nền.

  - Khởi tạo một kết nối TCP đến Server (Socket), thông qua port 80 (có thể sử dụng port thay thế là 8080).
  - HTTP như là "cánh cửa" cho quá trình gửi – nhận dữ liệu giữa Client và Server, ở bước thứ hai này, Client gửi yêu cầu đến Server.
  - Server gửi phản hồi đến Client.
  - Server đóng kết nối.

- Tiến trình chủ gửi các tập tin được yêu cầu đến tiến trình khách mà không lưu lại bất kỳ thông tin trạng thái nào của tiến trình khách.

- Có 2 phiên bản HTTP thông dụng là HTTP/1.0 và HTTP/1.1. Trong đồ án này, chúng em sử dụng HTTP/1.1, với các lý do:

  - Tất cả các yêu cầu và phản hồi sử dụng chung một kết nối TCP (Kết nối thường trực).
  - Các bước số 2 và 3 có thể được lặp lại nhiều lần trong bước 1 và 4.
  - Chính vì thế, phiên bản HTTP/1.1 có tính mở rộng hơn.

3.2. Cấu trúc thông điệp yêu cầu:

- Sau đây là ví dụ về một thông điệp yêu cầu mà Client gửi lên Server:

```
GET /index.html HTTP/1.1

Host: 192.168.95.1:80

Connection: keep-alive
```

```
Upgrade-Insecure-Requests: 1

User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) coc_coc_browser/87.0.152 Chrome/81.0.4044.152
Safari/537.36

Accept: text/html, application/xhtml+xml, application/xml; q=0.9,
image/webp, image/apng, */*; q=0.8, application/signed-exchange;
v=b3; q=0.9

Accept-Encoding: gzip, deflate

Accept-Language: vi-VN, vi; q=0.9, fr-FR; q=0.8, fr; q=0.7, en-US;
q=0.6, en; q=0.5
```

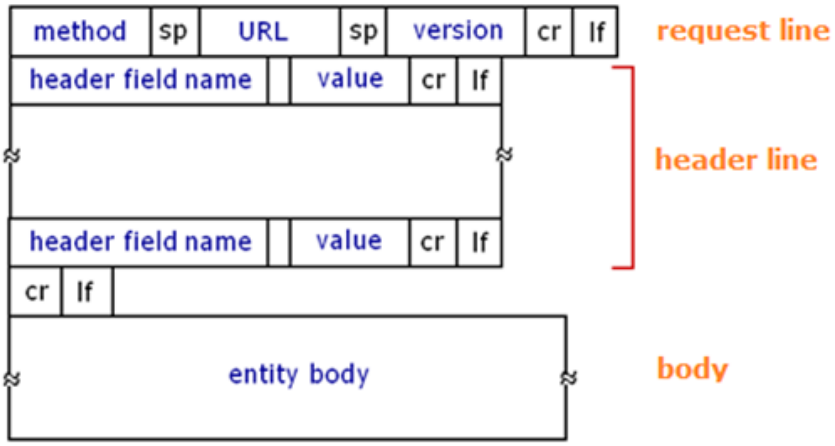
- Các dòng trong thông điệp trên được kết thúc bằng dấu xuống dòng và về đầu dòng.

- Đây là một cấu trúc ví dụ, về mặt thực tế, yêu cầu có thể có ít hoặc nhiều dòng hơn. Nhưng vẫn đảm bảo 2 phần, đó là dòng yêu cầu (request line) và dòng tham số (header line).

- Dòng yêu cầu có 3 trường: phương thức, URL (đường dẫn đến đối tượng) và phiên bản HTTP. Trường phương thức có thể nhận GET, POST, PUT, HEAD và DELETE; nhưng trong phạm vi đồ án này, trường phương thức chỉ nhận GET hoặc POST.

- Các dòng tham số cho biết tên máy chủ chứa đối tượng được yêu cầu, phương thức kết nối (thường trực (Keep-Alive) hay không thường trực (Close)), loại trình duyệt đang gửi yêu cầu đến máy chủ, cho biết thông tin chấp nhận những loại file nào, phương thức nén nào hay ngôn ngữ nào.

- Ngoài ra, theo sau các dòng tham số là một dòng trống và phần thân thông điệp (entity body). Với phương thức GET, phần này thường để trống, nhưng với phương thức POST, phần này được sử dụng để chứa các tham số mà Client gửi lên Server. Sau đây là cấu trúc tổng quát của thông điệp yêu cầu:



- Ví dụ về thông điệp yêu cầu của phương thức POST (thường được sử dụng trong các form hoặc người dùng muốn đưa dữ liệu lên Server):

```
POST /login HTTP/1.1

User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) coc_coc_browser/87.0.152 Chrome/81.0.4044.152
Safari/537.36

Content-Type: application/x-www-form-urlencoded

Content-Length: 32

// lưu ý quan trọng: Đây là dòng trống

username=admin&password=admin
```

3.3. Cấu trúc thông điệp phản hồi:

```
HTTP/1.1 200 OK

Server: WebServer of Linh and Loc

Date: Fri Jul 03 14:36:57 ICT 2020

Content-type: text/html

Content-length: 2663
```

- Các dòng trong thông điệp trên được kết thúc bằng dấu xuống dòng và về đầu dòng.

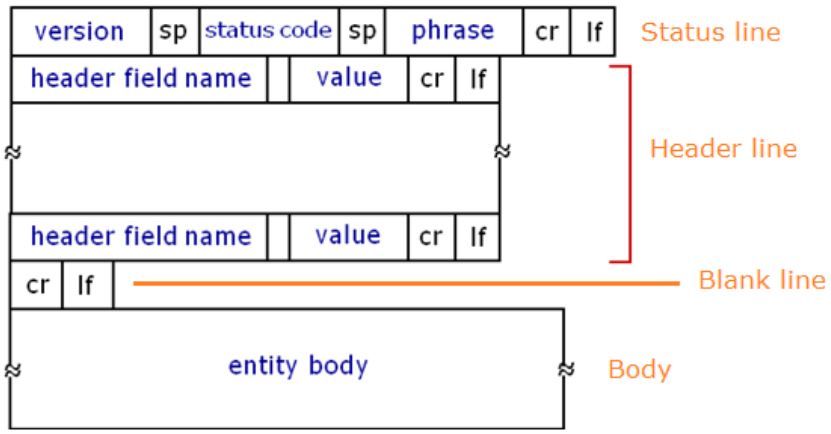
- Đây là một cấu trúc ví dụ, về mặt thực tế, yêu cầu có thể có ít hoặc nhiều dòng hơn. Nhưng vẫn đảm bảo các phần: dòng trạng thái (status line), dòng mô tả (header line) và phân thân thông điệp (entity body).

- Dòng trạng thái cho biết tiến trình chủ đang dùng phiên bản HTTP nào, mã trạng thái và tình trạng phản hồi.

- Các dòng mô tả cho biết thông điệp phát sinh bởi trình chủ nào; thời điểm tiến trình chủ tạo thông điệp phản hồi và gửi đi; loại đối tượng trong phần thân văn bản và kích thước của đối tượng đó (theo byte) và một số thông tin khác.

- Theo sau các dòng mô tả là một dòng trống và phần thân thông điệp (entity body) chứa nội dung đối tượng được yêu cầu.

- Sau đây là cấu trúc tổng quát của thông điệp phản hồi:



- Sau đây là một số mã trạng thái và tình trạng phản hồi thông dụng từ phía tiến trình chủ, những dòng được tô màu biểu thị những mã trạng thái và tình trạng phản hồi được sử dụng trong đồ án này:

Mã trạng thái	Tình trạng phản hồi	Ý nghĩa
200	OK	Thông tin yêu cầu hợp lệ và được trả về
301	Moved Permanently	Đối tượng được yêu cầu chuyển đi nơi khác; URL mới được chỉ định trong dòng *Location:* của thông điệp phản hồi.

400	Bad Request	Tiến trình chủ không hiểu được thông điệp yêu cầu
404	Not Found	Tài liệu yêu cầu không tìm thấy trên máy chủ
505	HTTP Version Not Supported	Phiên bản giao thức HTTP trong thông điệp yêu cầu không được máy chủ hỗ trợ.

4. MÔ TẢ CÁCH THỨC THỰC HIỆN ỨNG DỤNG:

4.1. Sử dụng ServerSocket:

- Java cung cấp lớp thư viện ServerSocket để đại diện cho Server Sockets. Server Socket chạy trên Server và lắng nghe từ kết nối TCP. Mỗi ServerSocket lắng nghe trên một port, thông thường Web là port 80. Khi Client kết nối đến port này, Server Socket được đánh thức và thực hiện quá trình kết nối với Client. Có thể hiểu, Server Socket đợi kết nối trong khi Client khởi tạo kết nối. Khi Server Socket được thiết lập, quá trình truyền nhận dữ liệu giữa Client và Server được thực hiện cho đến khi đóng kết nối.

- Đây là các bước trong chu trình của một chương trình Server:

- Server khởi tạo một đối tượng ServerSocket, biểu thị số hiệu cổng (port) nào để xuất hiện giao tiếp. Ngoài ra ta có thể truyền vào các tham số về địa chỉ IP (nếu cần).
- Server gọi phương thức accept() của lớp ServerSocket. Phương thức này đợi tới khi một Client kết nối với Server trên cổng đã cho và trả về một Socket kết nối giữa Client và Server.
- Tùy thuộc vào loại Server, mà phương thức getInputStream() và getOutputStream() của Socket được gọi để giao tiếp với Client.
- Server và Client giao tiếp với nhau theo Protocol (ở đồ án này là HTTP) cho đến khi đến thời gian đóng kết nối.
- Server và Client đóng kết nối.
- Server quay trở về bước 2 và chờ kết nối kế tiếp.

4.2. Sử dụng đa luồng (Multithreaded):

- Thread (luồng) về cơ bản là một tiến trình con (sub-process). Một đơn vị xử lý nhỏ nhất của máy tính có thể thực hiện một công việc riêng biệt. Trong Java, các luồng được quản lý bởi máy ảo Java (JVM).

- Multi-thread (đa luồng) là một tiến trình thực hiện nhiều luồng đồng thời. Một ứng dụng Java ngoài luồng chính có thể có các luồng khác thực thi đồng thời làm ứng dụng chạy nhanh và hiệu quả hơn.

- Trong khuôn khổ đồ án này, đa luồng được thể hiện rõ ràng ở chỗ: Trong các file HTML có nhiều đối tượng về hình ảnh, CSS,… Khi ta duyệt Web, các đối tượng này sẽ được tải đồng thời bởi các luồng khác nhau.

- Ưu điểm đa luồng:

  - Các luồng là độc lập, vì thế có thể thực hiện nhiều công việc cùng một lúc.
  - Mỗi luồng có thể dùng chung và chia sẻ nguồn tài nguyên trong quá trình chạy, nhưng thực hiện một cách độc lập.
  - Luồng là độc lập vì vậy nó không ảnh hưởng đến luồng khác nếu ngoại lệ xảy ra trong một luồng duy nhất.
  - Có thể thực hiện nhiều hoạt động với nhau để tiết kiệm thời gian, tăng trải nghiệm cho người sử dụng.

4.3. Quy trình xử lý của Web Server khi nhận được yêu cầu từ Client:

- Tiếp nhận yêu cầu từ Client thông qua lớp BufferReader của Java, khởi tạo các biến cần thiết chuẩn bị cho việc phản hồi về lại Client.

- Tiến hành lấy phương thức HTTP và URL từ yêu cầu của Client để bắt đầu xử lý.

- Kiểm tra Client gửi lên phương thức HTTP nào (GET / POST) và gọi hàm xử lý tương ứng (trong khuôn khổ đồ án này, chỉ chấp nhận 2 phương thức GET và POST).

- Sau khi xử lý và gửi phản hồi cho Client, các biến lưu trữ dữ liệu gửi lên từ Client cần được giải phóng và Server đóng kết nối với Client.

4.3.1. Xử lý phương thức GET:

- Xác định loại đối tượng (văn bản HTML / hình ảnh / âm thanh,…).

- Kiểm tra URL ở bước 2 mục 4.3 (trong báo cáo này, chúng em gọi đây là fileName) có phải là file hay không?

  - Nếu phải là file, tiến hành lấy kích thước file và lấy nội dung file.
  - Nếu không phải là định dạng File, tiến hành gọi hàm toHTML(fileName), hàm này có tác dụng như là viết HTML dưới Server thay vì là 1 file HTML riêng. Sở dĩ chúng em làm như vậy là do có thể lấy được đường dẫn qua lại khi thực hiện chức năng số 3 trong yêu cầu đồ án (sẽ được trình bày sau). Sau đó, tiến hành lấy kích thước và nội dung chuỗi mà hàm toHTML trả về.

- Ghi lại các thông tin mà thông điệp phản hồi trả về, cụ thể:

  - Status line: HTTP/1.1 200 OK (phương thức GET nên trả về mã trạng thái là 200).
  - Header line: các thông tin về thông điệp phát sinh bởi trình chủ nào (Server: WebServer of Linh and Loc); thời điểm tiến trình chủ tạo thông điệp phản hồi và gửi đi (Date); loại đối tượng trong phần thân văn bản (Content-type) và kích thước của đối tượng đó (Content-length).
  - Một dòng trống (\r\n), lệnh println trong lớp PrintWriter với hệ điều hành Windows là \r\n.
  - Entity body: chứa nội dung của đối tượng được yêu cầu.

4.3.2. Xử lý phương thức POST:

- Đọc các dòng trong thông điệp yêu cầu gửi lên, lấy ra phần thân thông điệp để xác định dữ liệu mà Client gửi lên.

- Kiểm tra URL ở bước 2 mục 4.3 (trong báo cáo này, chúng em gọi đây là fileName) có phải là “/login” (đăng nhập) hay không?

  - Nếu đúng, tiến hành kiểm tra phần thông điệp chứa userName và passWord có phải là admin và admin hay không. Nếu đúng thì thực hiện Redirect đến trang info.html và sử dụng phương thức GET để lấy trang info.html chứa thông tin thành viên trong nhóm.

Ngược lại, redirect đến trang 404.html với mã trạng thái 404 và tình trạng phản hồi là Not Found.

- Nếu là phương thức POST nhưng fileName != "/login", khi đó nội dung phần data sẽ là 1 đường dẫn, ta thực hiện lấy đường dẫn này để thực hiện chức năng số 3 trong yêu cầu đồ án (sẽ được trình bày sau). Sau đó redirect đến trang theo thông tin của đường dẫn.

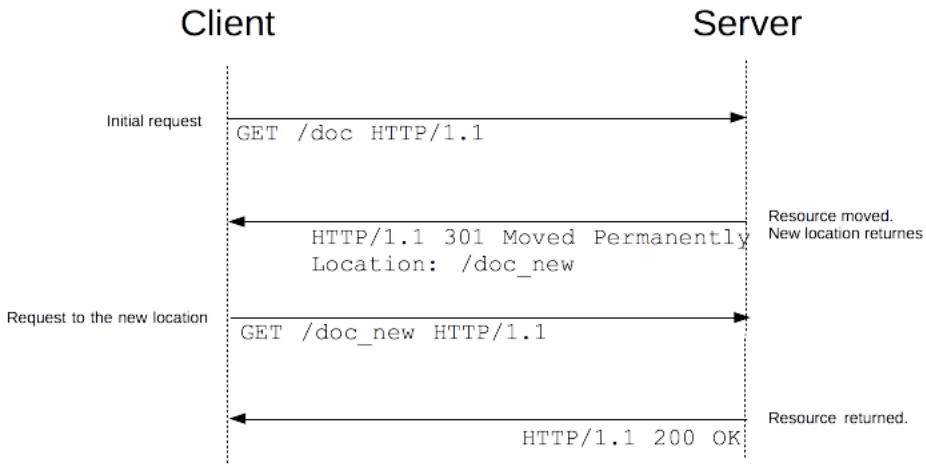
4.3.3. Phản hồi với Redirect:

- Status line: HTTP/1.1 301 Moved Permanently (mã trạng thái 3xx phản ánh chức năng Redirect).

- Header line: các thông tin về thông điệp phát sinh bởi trình chủ nào (Server: WebServer of Linh and Loc); thời điểm tiến trình chủ tạo thông điệp phản hồi và gửi đi (Date); loại đối tượng trong phần thân văn bản (Content-type) và kích thước của đối tượng đó (Content-length). Đặc biệt trong phần này có thêm một dòng là **Location**, lưu đường dẫn mới cần redirect đến.

- Một dòng trống (\r\n), lệnh println trong lớp PrintWriter với hệ điều hành Windows là \r\n.

- Khi Client nhận được phản hồi, Client gửi tiếp yêu cầu GET trang cần chuyển đến, Server thực hiện phản hồi yêu cầu này theo quy trình như đã trình bày ở 4.3.1. Quy trình của Redirect được mô tả như hình sau:

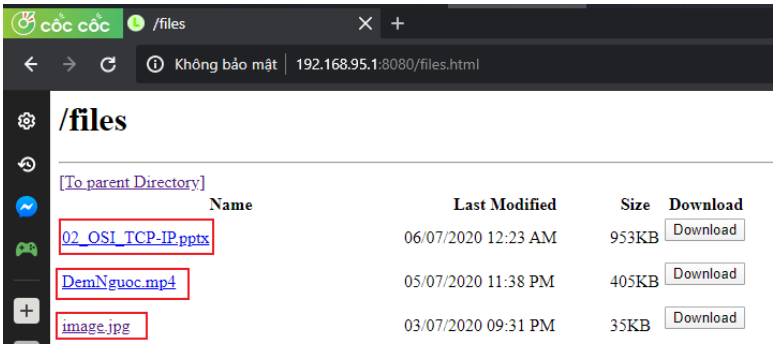


4.3.4. Download Files và cho phép tải hình ảnh trang Web:

- Người quản trị server copy các file hoặc folder muốn chia sẻ vào ./template/files.

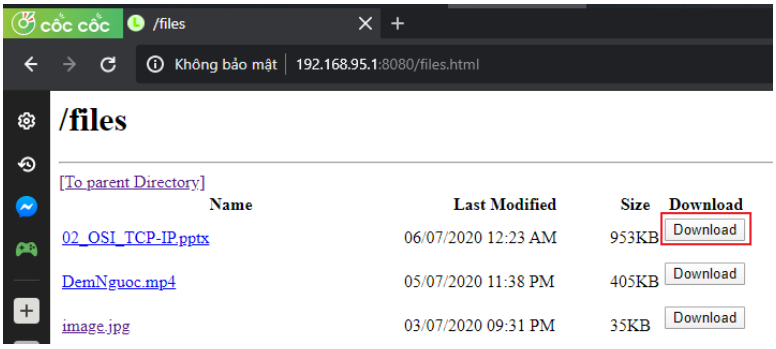
- Dùng phương thức list() của lớp File để lấy tên các file trong 1 directory. Từ tên file dùng các phương thức lastModified() và length() để lấy thông tin về ngày sửa đổi và độ lớn. Tuy nhiên, phương thức lastModified() trả về dữ liệu kiểu long nên dùng phương thức format() của lớp SimpleDateFormat để định dạng về dạng thân thiện với người dùng. Hàm toHtml() dùng để gắn các thông tin này cùng với các thẻ html để trả lời cho client.

- Download theo phương thức GET: thuộc tính download được hỗ trợ bởi thẻ <a> của html. Khi click vào đường dẫn, client tự động gửi request GET đến href đã được đặt rồi tải về máy client. Các thẻ <a> chứa directory không có thuộc tính download này.



- Download theo phương thức POST: Nhờ các nút download. Các nút này thực chất mỗi nút là thành phần của một form với method post, action được đặt là directory cao nhất. Form này có 1 thẻ input loại hidden, có value là tên file và một thẻ input loại submit chính là nút download nhìn thấy. Ví dụ khi bấm nút down load bên dưới, một request với dòng đầu header POST /files (directory cao nhất) HTTP/1.1 với body fileName=02_OSI_TCP-IP.pptx được gửi cho server. Server nhận request và đọc phần body rồi điều hướng (Redirect trình bày ở 4.3.3) người dùng đến đường dẫn của file cần tải. Với kiểu này, người dùng có thể xem trước các file có định dạng hình ảnh, pdf, audio, video trước khi quyết định tải xuống.

- Để trình duyệt hiểu và hiện thị các file xem trước, thực hiện đặt MIME type trong response header phù hợp. Ví dụ .mp4 có MIME type là video/mp4. Trình duyệt sẽ tự hiểu và hiển thị video cho người dùng. Các định dạng file không được hỗ trợ xem trước như docx, pptx, xlsx,.. sẽ tự động tải về mà không xem trước.



5. CÁCH TEST ỨNG DỤNG:

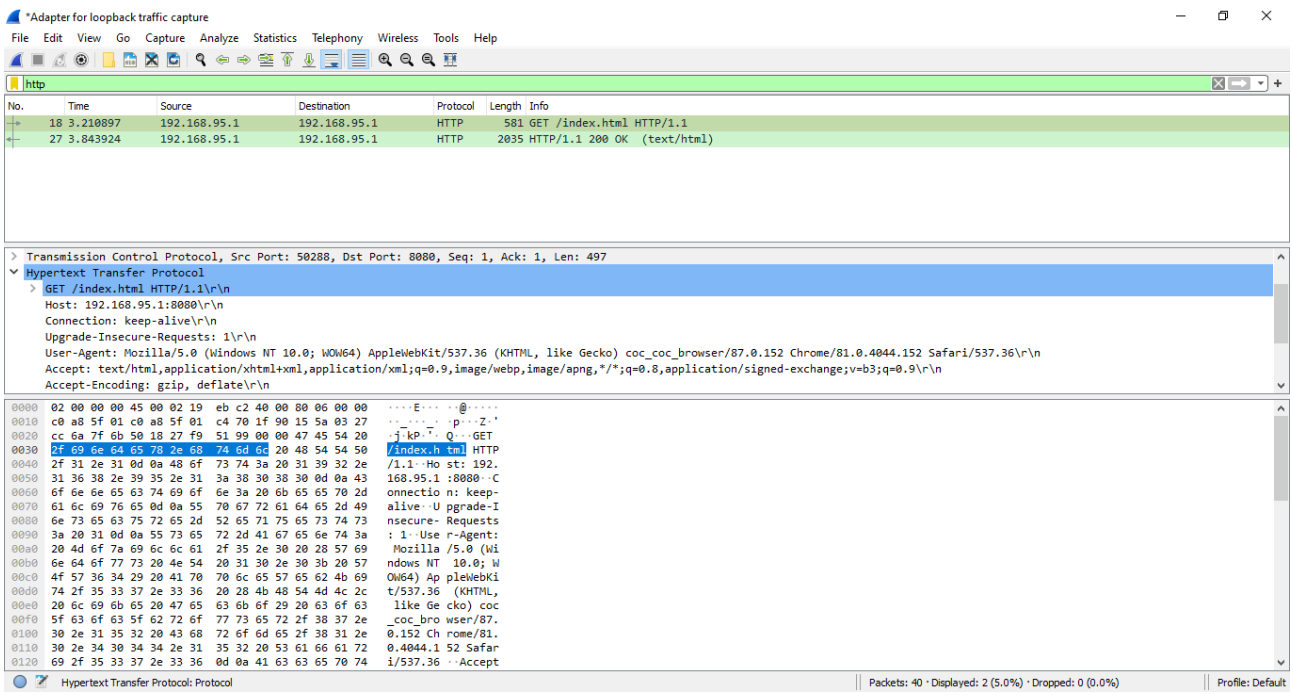
5.1. Sử dụng Wireshark để bắt gói tin (cách test chính của nhóm):

- Chọn card mạng Adapter for loopback traffic capture và tiến hành bắt gói tin.

- Sử dụng bộ lọc của Wireshark để lọc ra các gói tin sử dụng protocol HTTP để dễ quan sát.

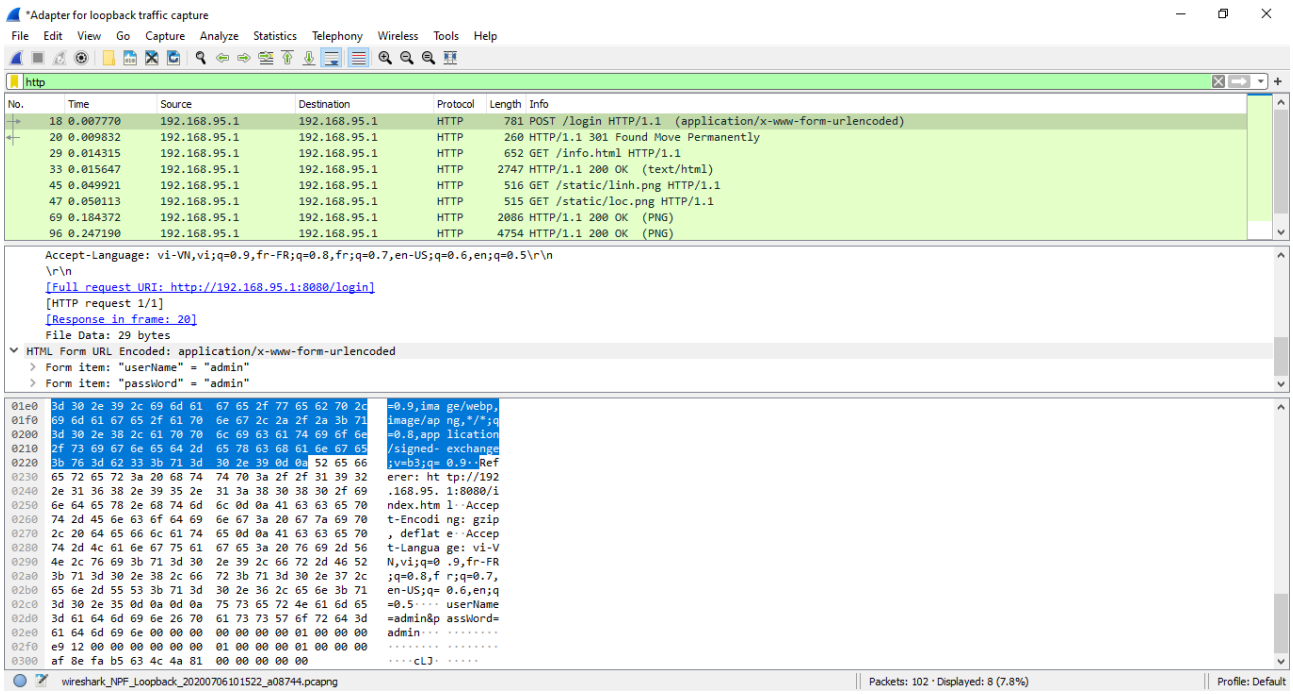
- Ta có thể xem các gói tin TCP còn lại để quan sát quá trình truyền và nhận gói tin.

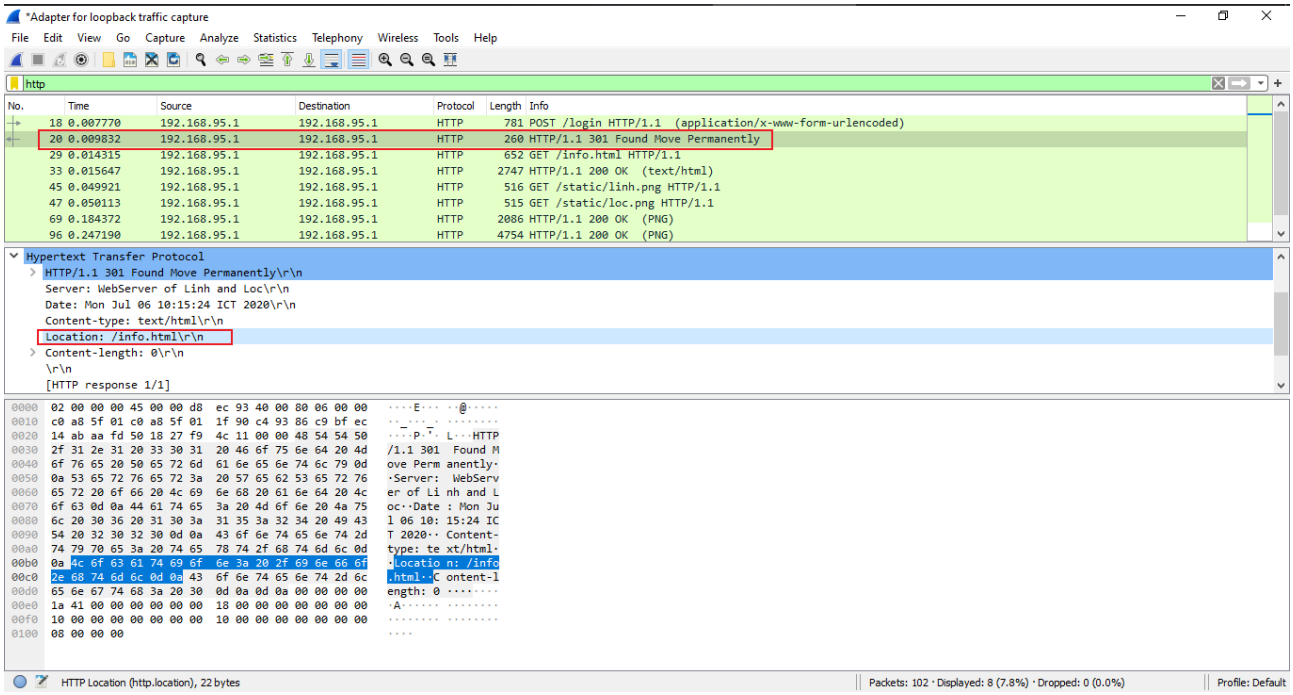
- Hình ảnh minh họa cho phương thức GET trang index.html, ta thấy có thông điệp yêu cầu và thông điệp phản hồi được Wireshark bắt:

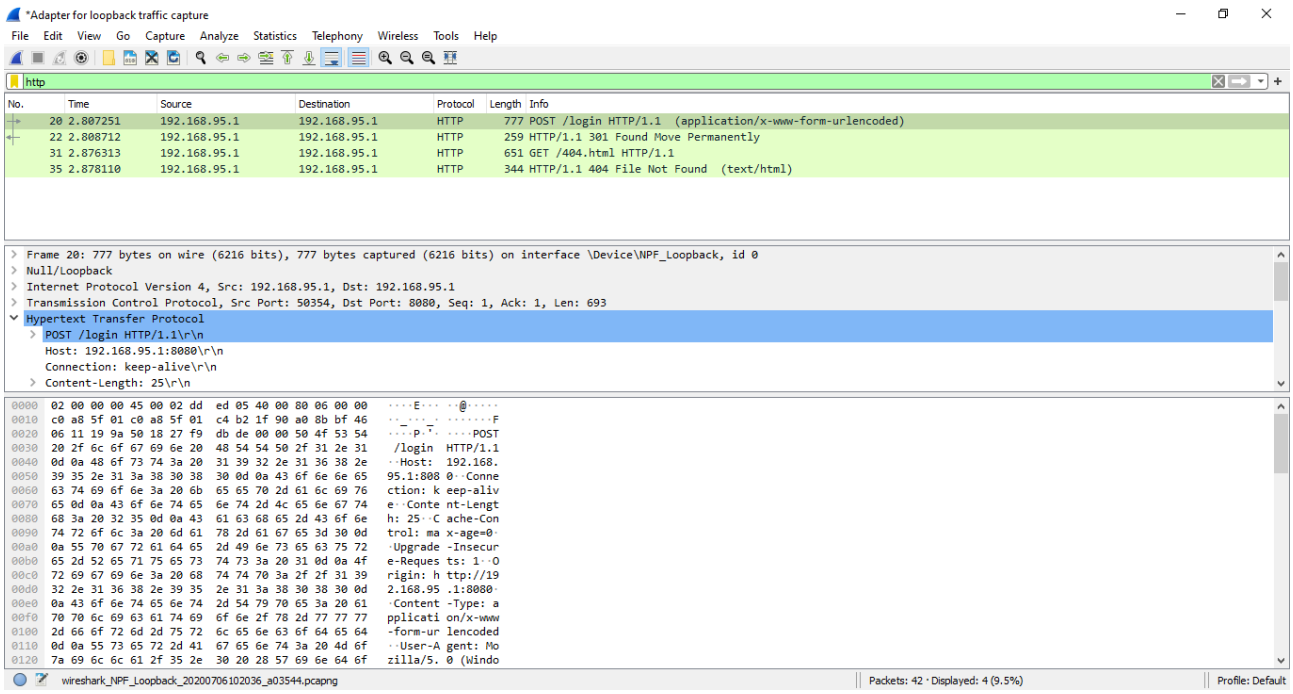


- Hình ảnh minh họa cho phương thức POST khi đăng nhập và redirect đến các trang theo yêu cầu đề bài. Ta thấy phương thức POST đã gửi dữ liệu là username và password lên Server, sau đó Server gửi phản hồi với mã trạng thái 301 báo hiệu redirect đến Location mới, sau đó

là quá tình yêu cầu – phản hồi để tải đến trang trong Location mới (trang info.html nếu đăng nhập thành công, trả về mã trạng thái 404 với trang 404.html nếu đăng nhập không thành công):

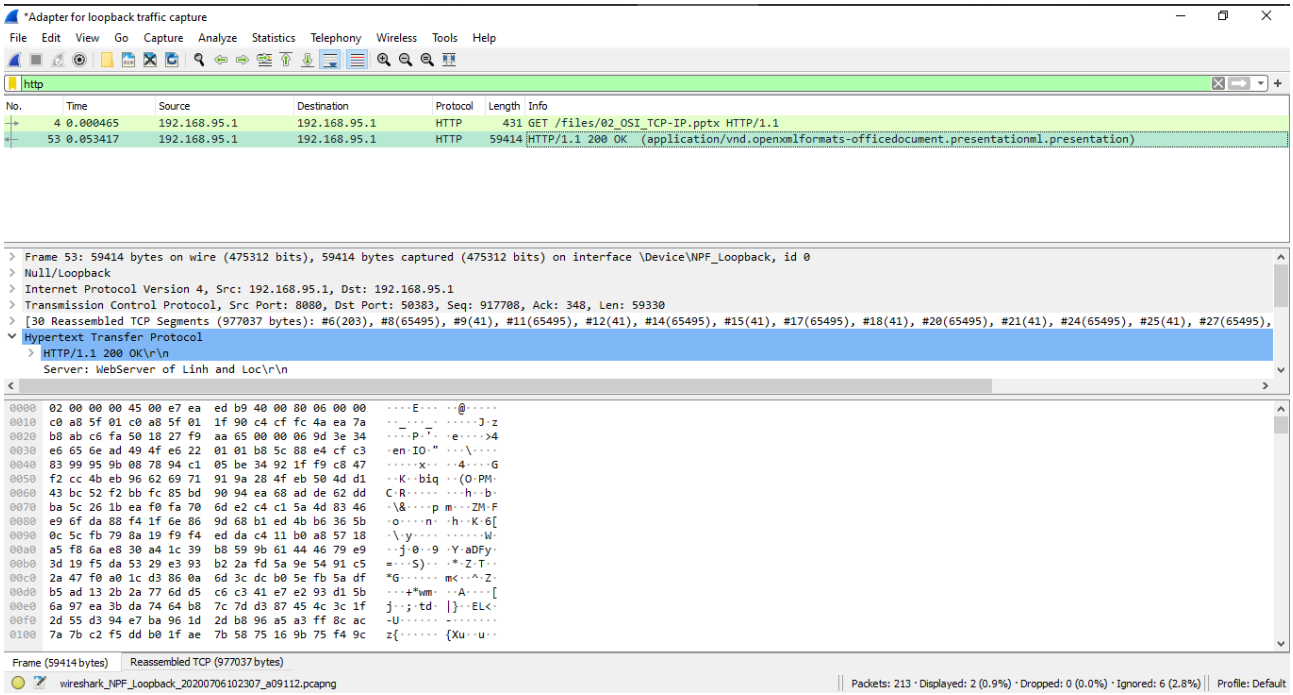






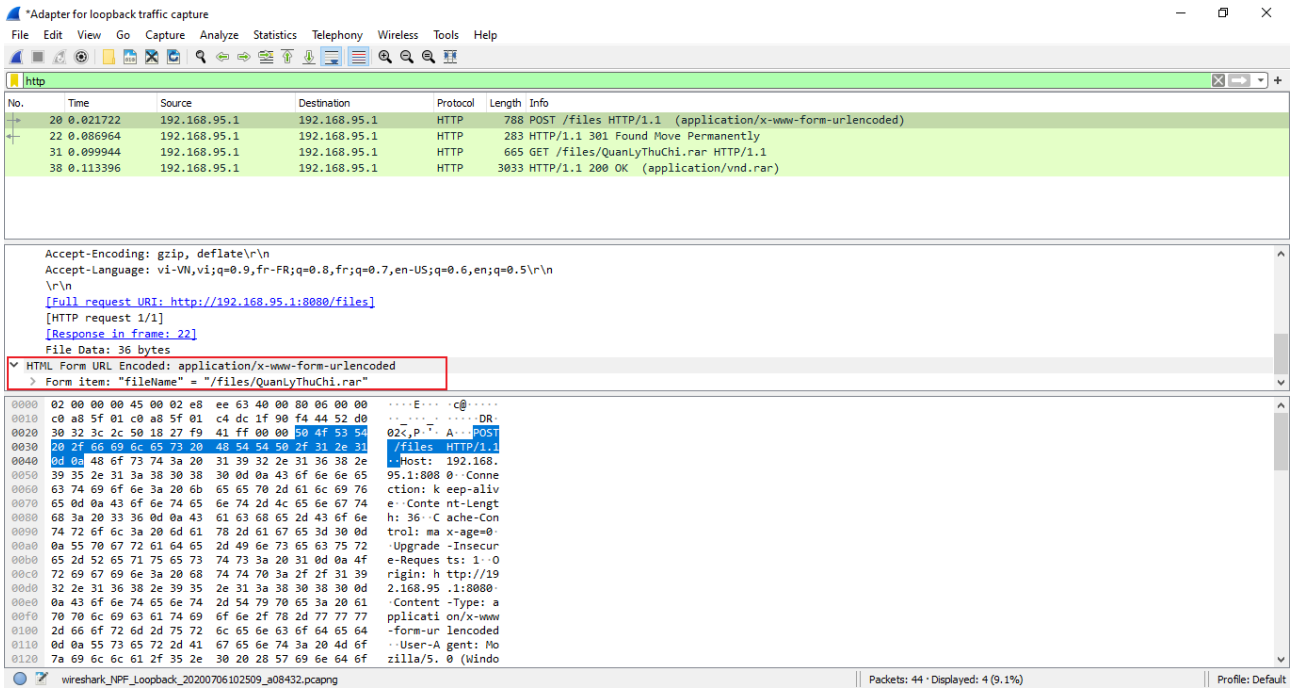
- Với download file, ta sử dụng 2 phương thức GET và POST như sau:

  - Phương thức GET: ta thực hiện bấm click vào tên file, khi đó Wireshark bắt được các gói tin như sau:

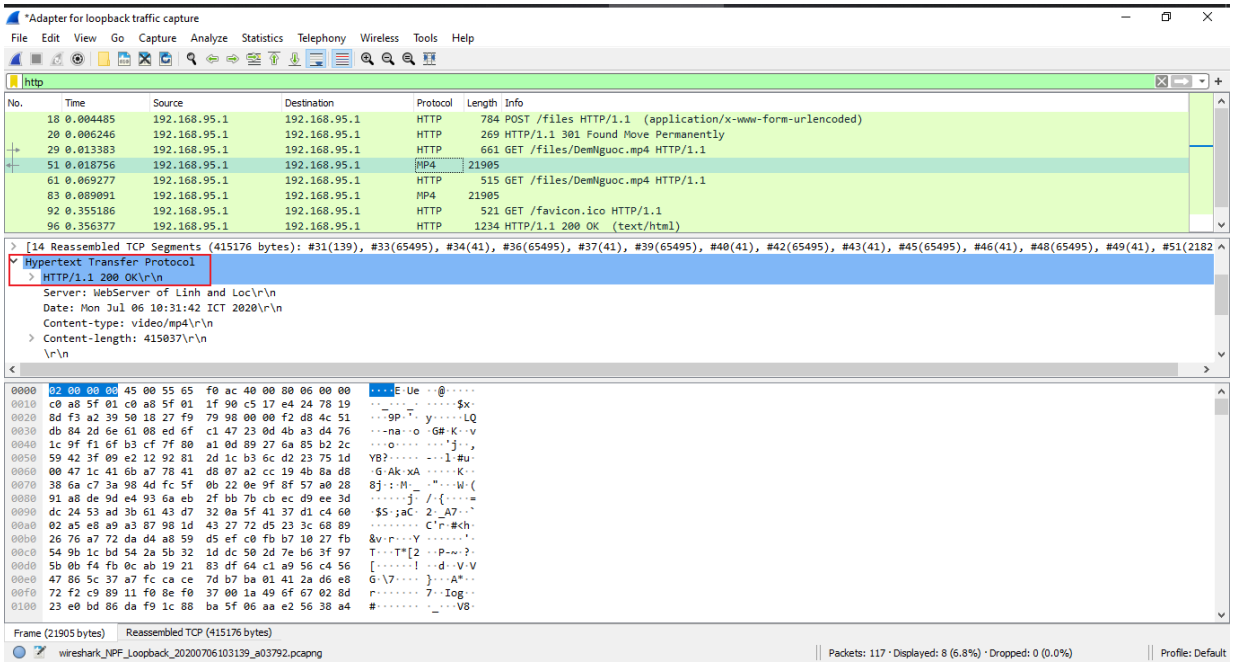


  - Phương thức POST: ta thực hiện bấm click vào nút Download, khi đó Wireshark bắt được các gói tin như hình sau. Ta thấy khi click vào nút Download, giống như khi đăng

nhập, sẽ gửi dữ liệu tên file lên Server, Server tiếp nhận và redirect đến file đó và tiến hành cho phép download:

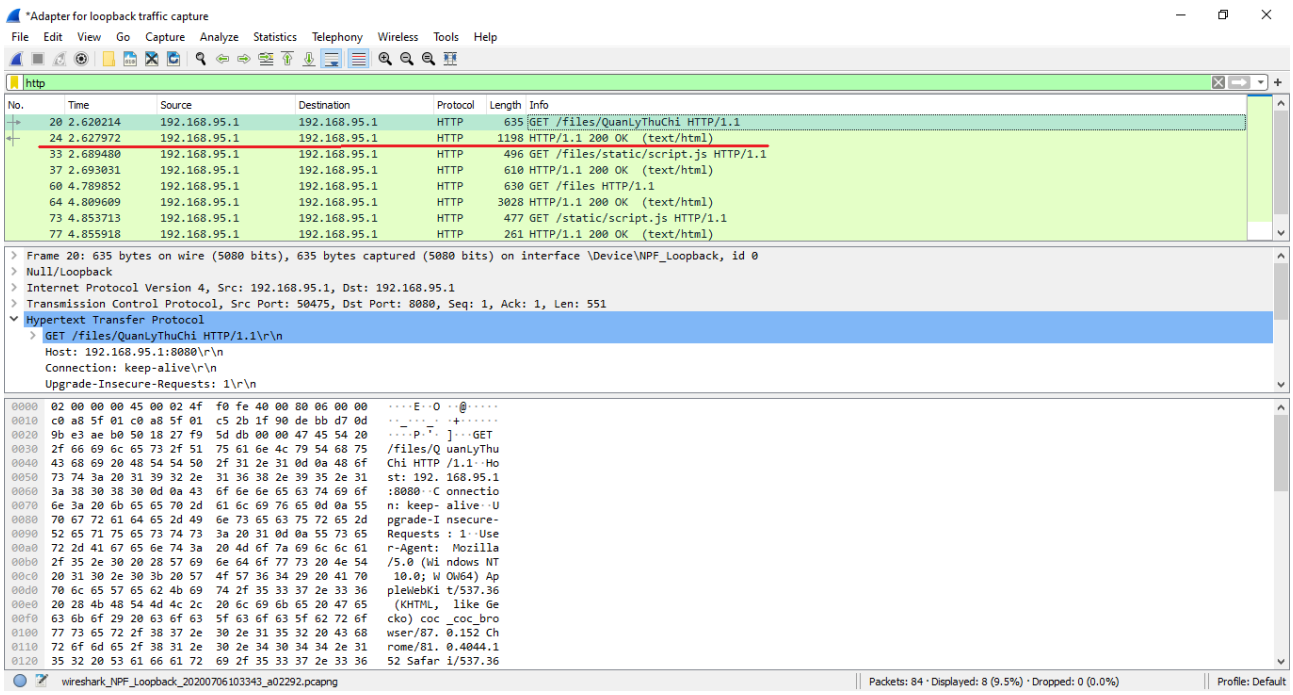


- Ngoài ra, với các file có định dạng .pdf, hình ảnh, .mp3, .mp4 ta có thể thực xem trước khi tải, khi thực hiện xem trước, Wireshark bắt được các gói tin sau (giả sử test với file .mp4):



- Ngoài ra ta có thể vào folder con và quay lại folder mẹ. Khi đó Wireshark để bắt được gói tin thực hiện việc chuyển vào folder con, khi nhấn "To parent Directory", Wireshark bắt được gói tin thực hiện việc quay lại folder trước (folder mẹ). Gia sử ta đang ở files.html, ta đến thư

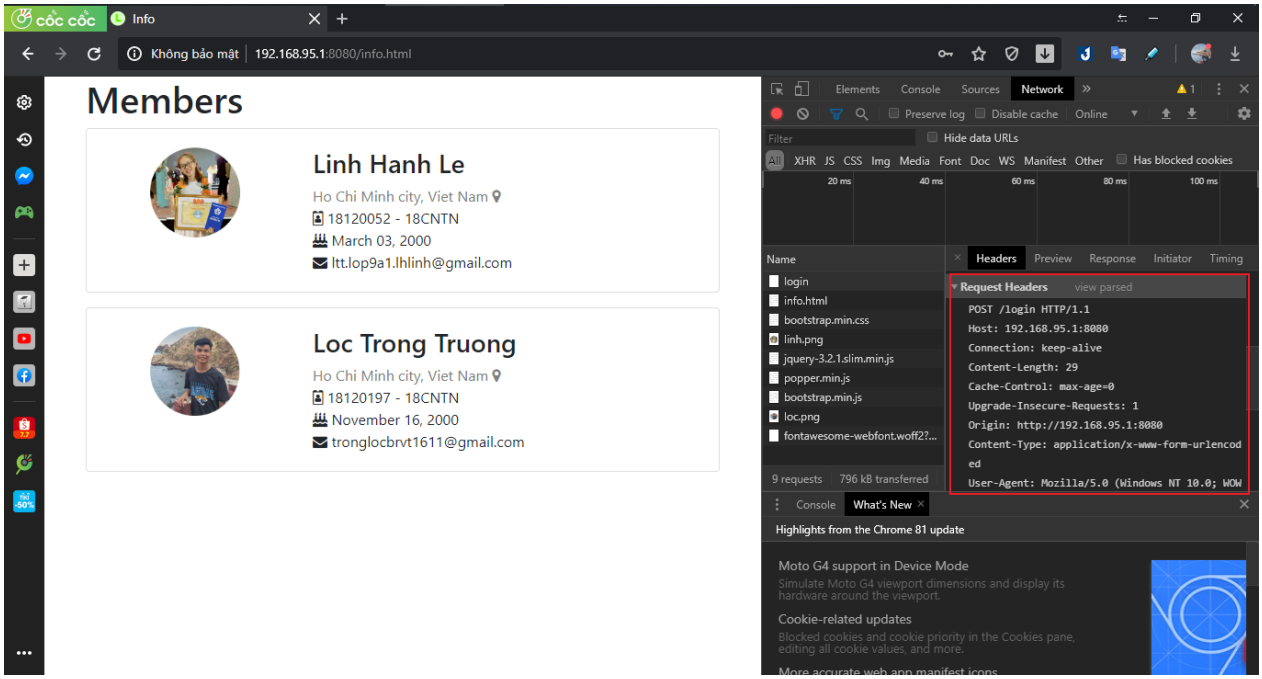
mục QuanLyThuChi và quay trở về thư mục trước (files.html), ta ngăn cách để thể hiện 2 quá trình:



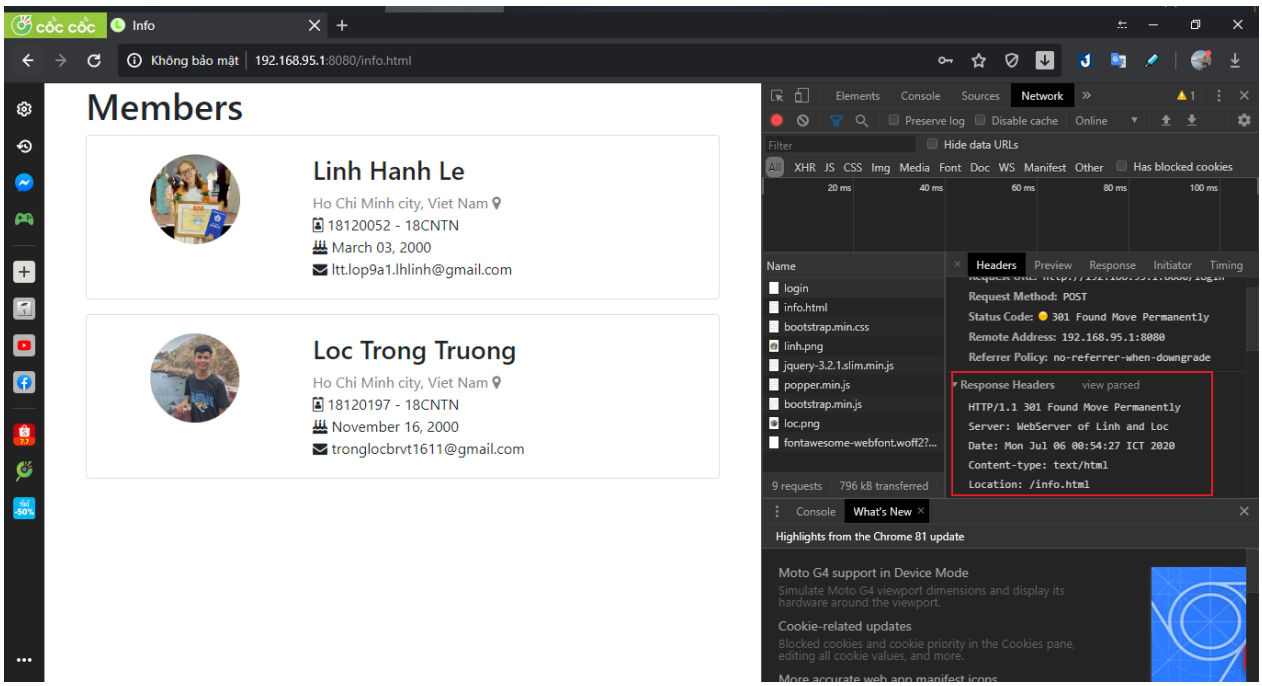
5.2. Sử dụng Panel Network của Browser (F12):

Công cụ hữu ích để phân tích các sự kiện Network. Network log liên tục ghi nhận thông tin khi có network request. Panel này cung cấp thông tin các sự kiện network và cho phép xem chi tiết của từng sự kiện nên có thể theo dõi thông tin của các gói tin như mã trạng thái, tên file, content-type, body data,…

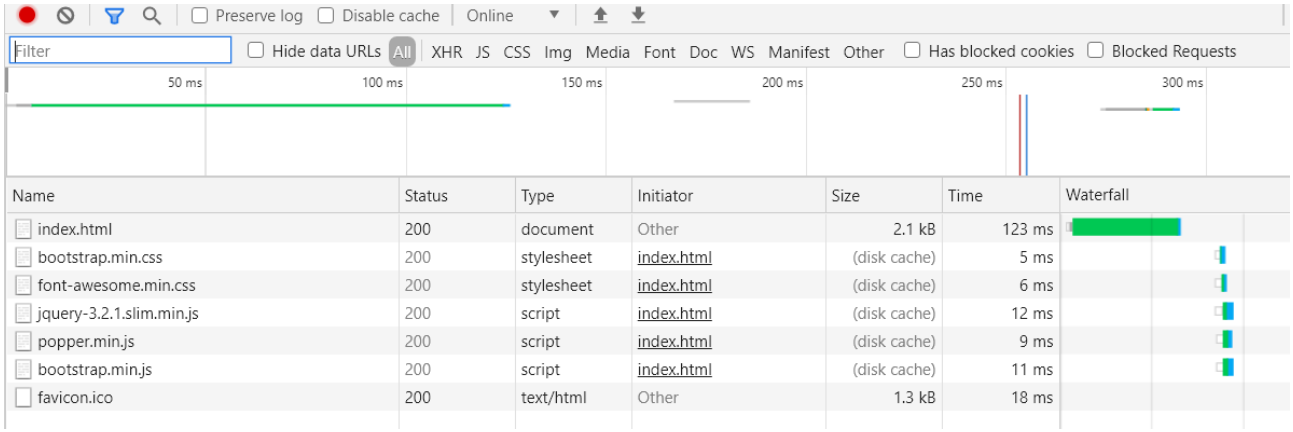
- Thông điệp yêu cầu:



- Thông điệp phản hồi:

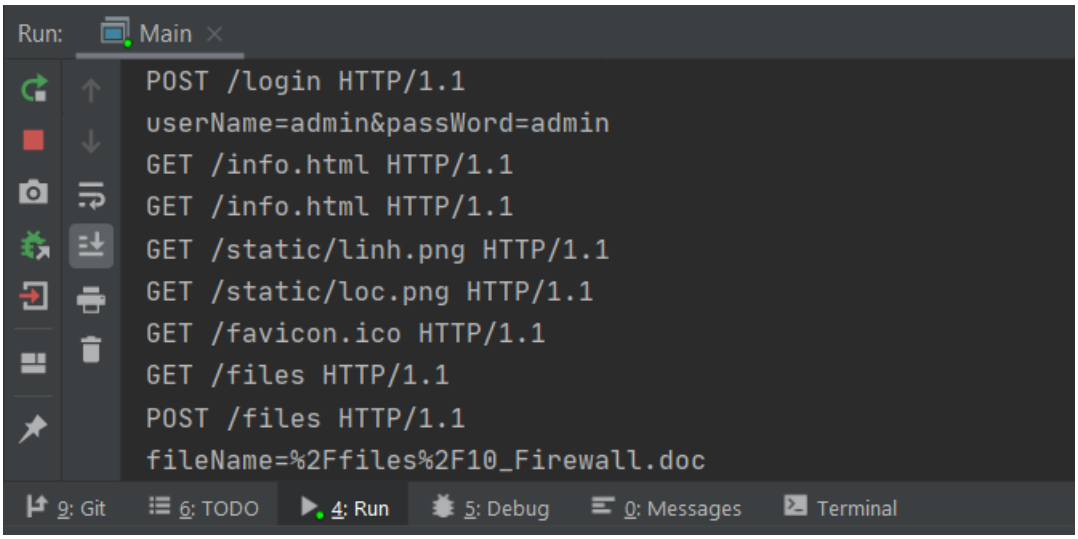


- Các tiến trình khi truy cập vào index.html:



5.3. Sử dụng phương pháp in ra màn hình từ IDE:

Đối với ngôn ngữ Java, IDE IntelliJ IDEA có thể in các giá trị cần kiểm tra ra màn hình bằng lệnh System.out.println(<variable/value>). Ví dụ có thể in ra các dòng đầu của request nhận được và phần body (nếu có) ra màn hình để tiện theo dõi.



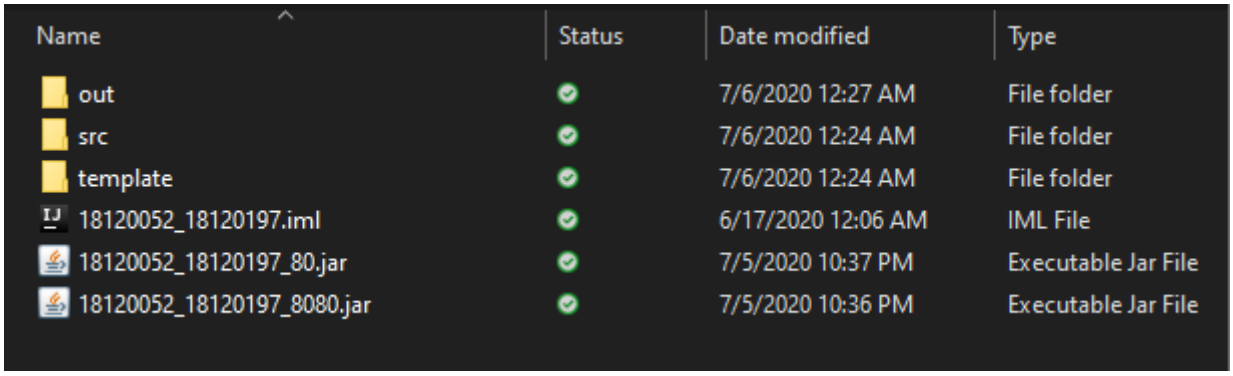
Ngoài ra, ta có thể in ra màn hình thông điệp phản hồi để có thể kiểm tra (thực hiện tương tự như in ra thông điệp yêu cầu).

6. HƯỚNG DẪN SỬ DỤNG ỨNG DỤNG:

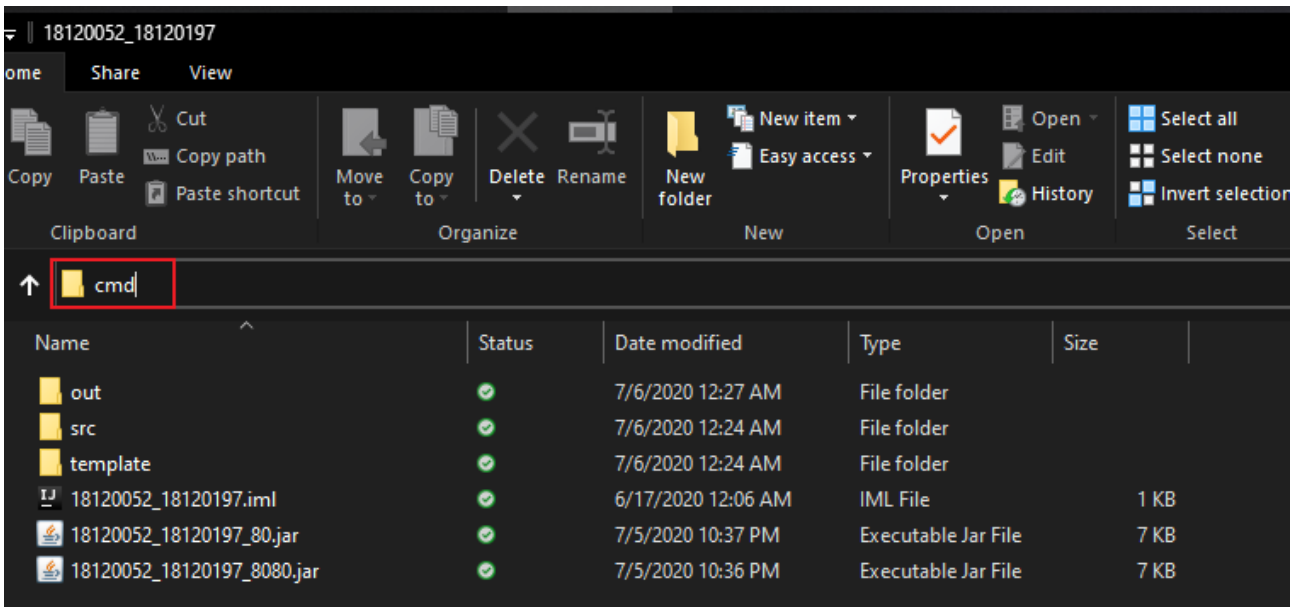
- Để chạy được chương trình này, cần cài đặt JAVA trước đó (vì đây là file .jar được build từ source code JAVA).

- Giải nén file nén, khi đó ta thấy có 1 folder 18120052_18120197 và 1 file pdf (chứa báo cáo), ta vào folder và đảm bảo rằng có 2 file .jar (tương ứng với port 80 và port 8080, để

phòng trường hợp 1 trong 2 port đã được sử dụng cho tiến trình khác) cùng cấp với các folder con. Hình ảnh minh họa các folder con và file trong folder 18120052_18120197 như sau:



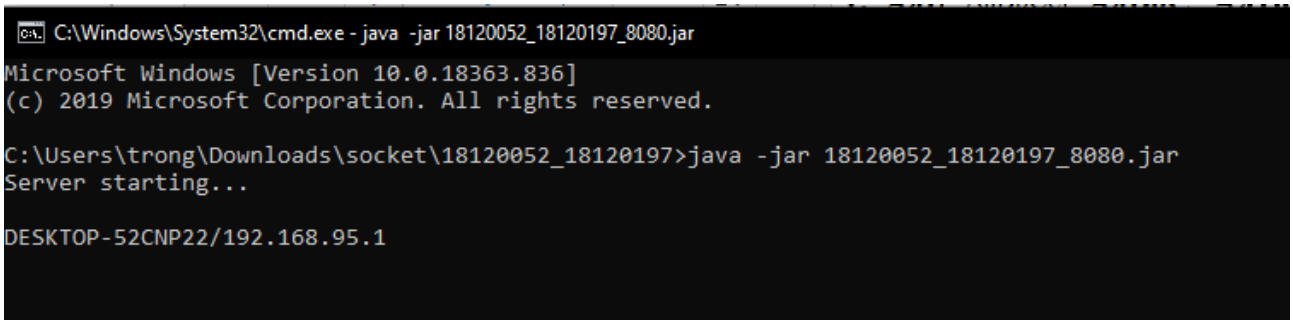
- Sau đó, ta sử dụng Command Line để chạy chương trình (cách vào nhanh cmd: sau khi xuất hiện như hình ảnh trên, tại mục đường dẫn, ta xóa và điền cmd, nhấn Enter):



- Khi đó, trên Command Line đã xuất hiện sẵn đường dẫn, ta nhập theo cú pháp và nhấn Enter:

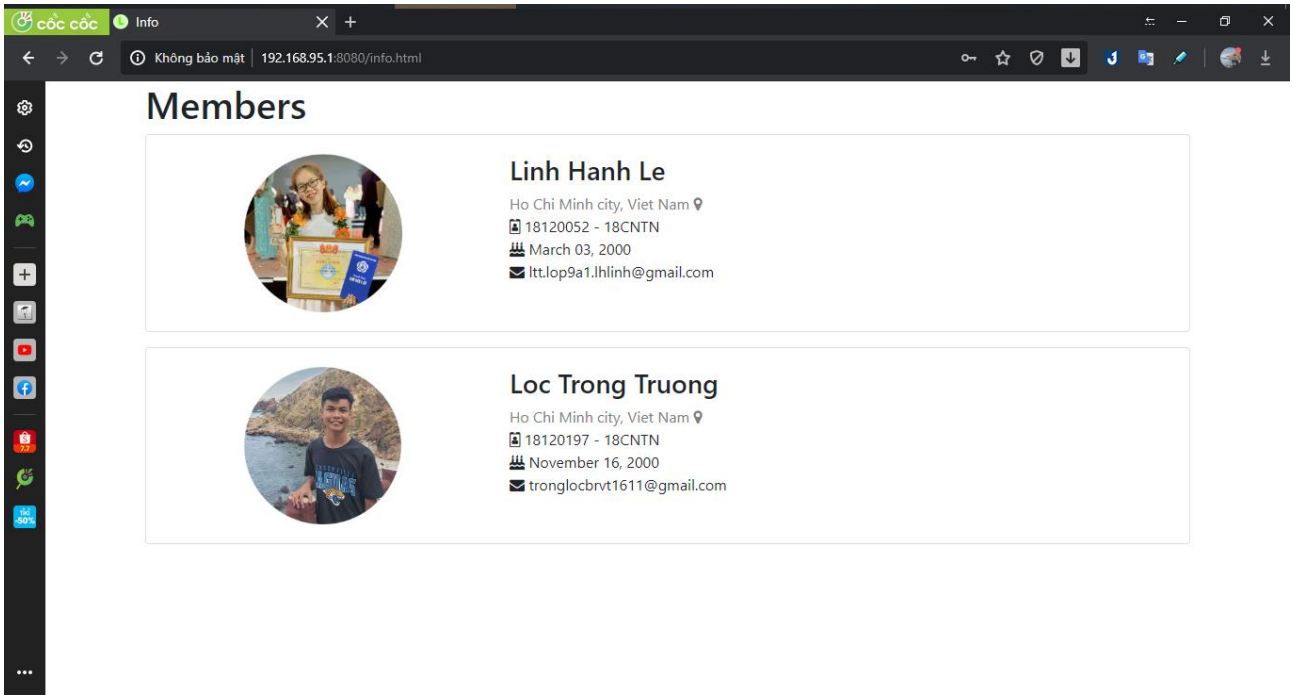
  - Với port 80: **java -jar 18120052_18120197_80.jar**
  - Với port 8080: **java -jar 18120052_18120197_8080.jar**

- Giao diện hiện ra như sau cho thấy Server đã được thiết lập và xuất hiện địa chỉ IP, các bạn lưu lại địa chỉ IP này để tiện truy cập trên Web Browser.

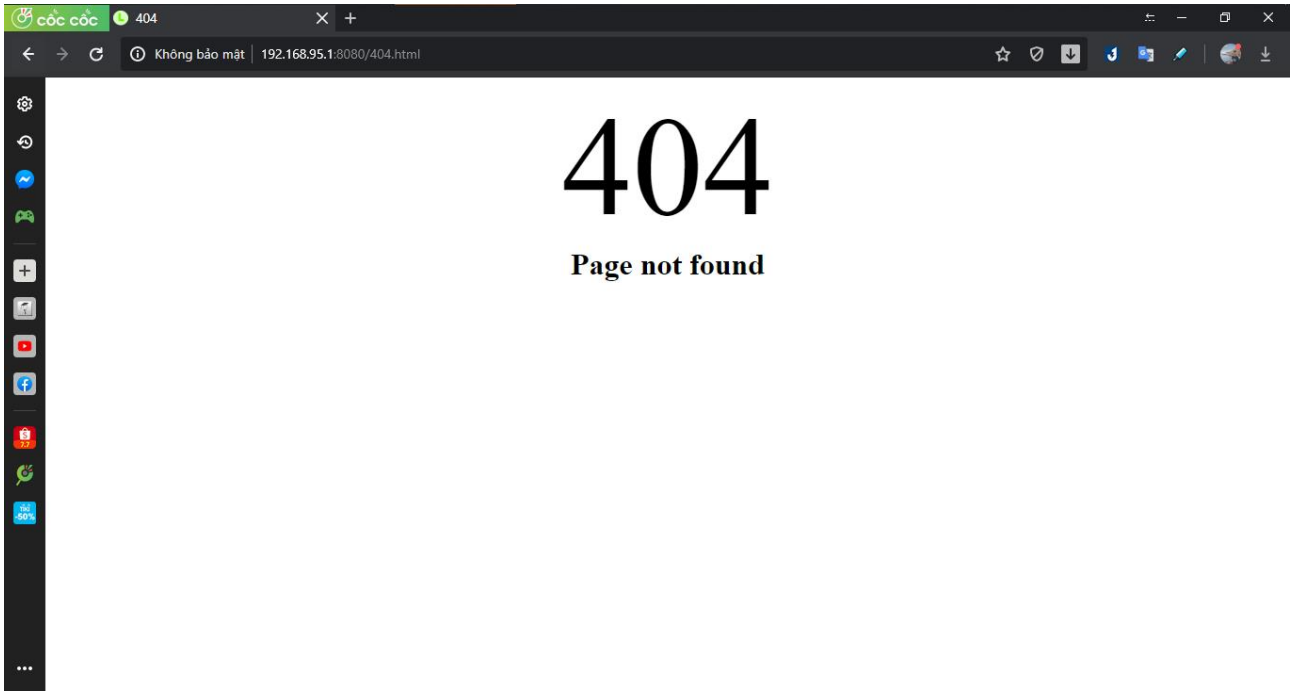


- Các bạn mở trình duyệt và nhập theo cú pháp: <địa chỉ IP>:<port>/index.html, ví dụ http://192.168.95.1:8080/index.html để truy cập vào trang Web.

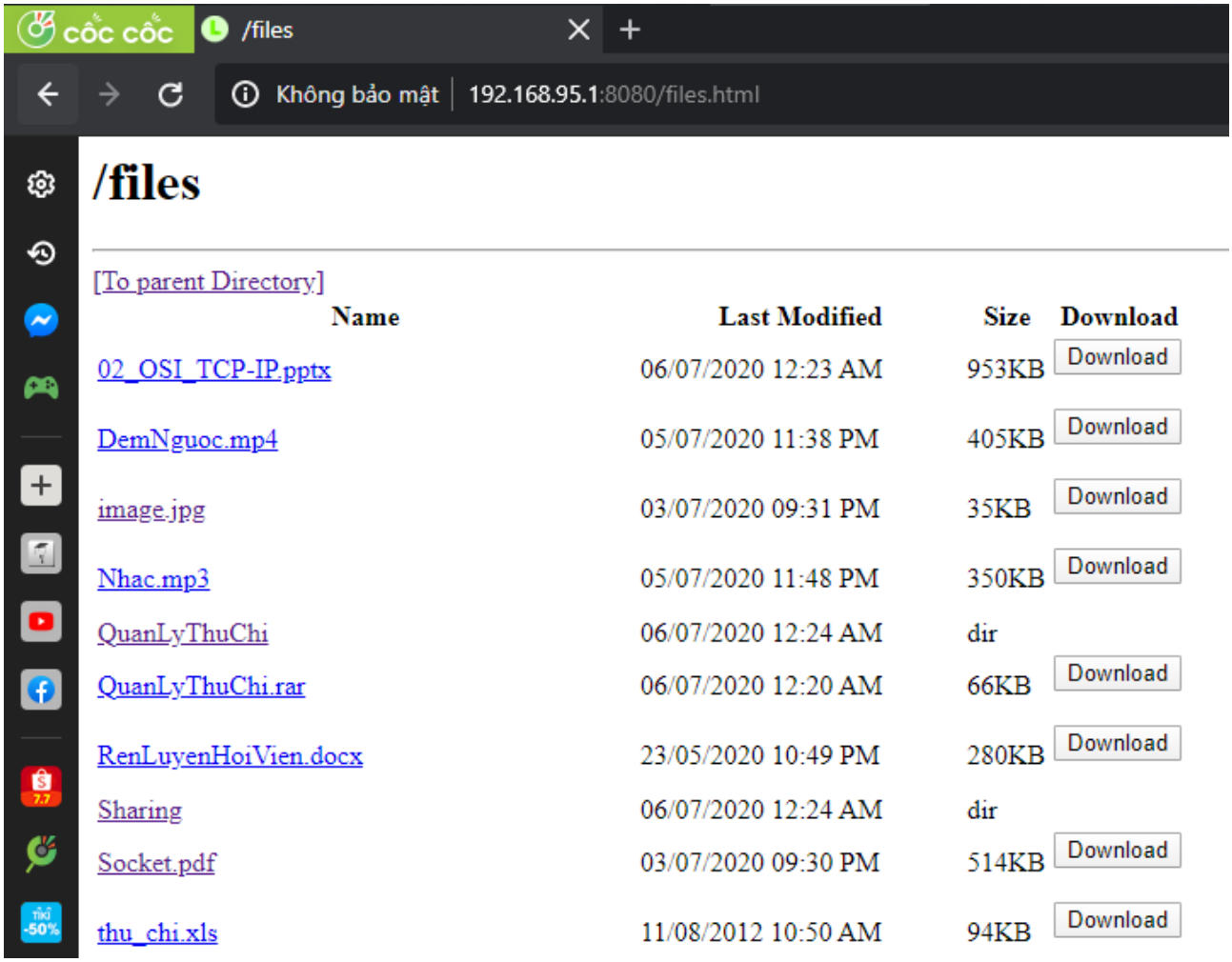
- Khi vào index.html, yêu cầu người dùng login, nếu người dùng login đúng username và password là admin và admin thì chuyển đến trang info.html



- Nếu đăng nhập sai, sẽ chuyển đến 404.html:



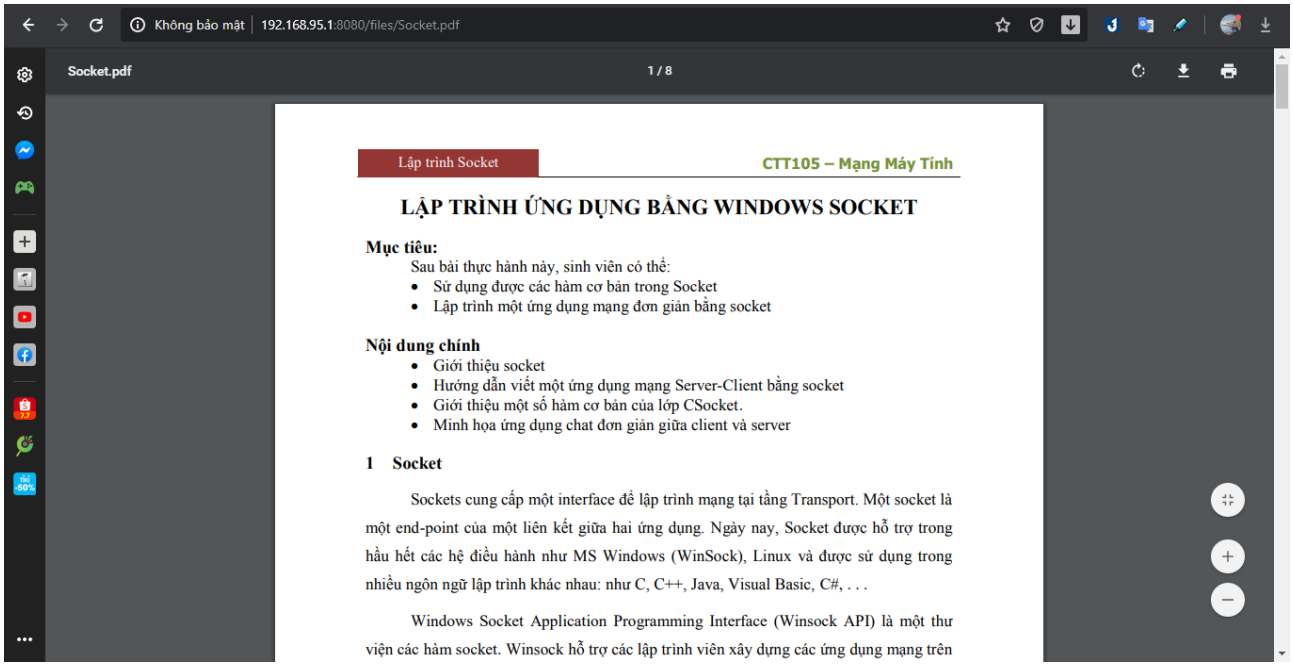
- Truy cập vào files.html để xem danh sách các files và có thể tiến hành download và hiển thị hình ảnh trang web:



- Có thể download file có sẵn hoặc di chuyển vào các folder (size là dir) để lấy các file của folder đó và download.

- Click vào tên file để tiến hành download theo phương thức GET.

- Click và nút download để có thể thực hiện phương thức POST và download (các file pdf, mp3, mp4, hình ảnh cho phép xem trực tuyến trước khi tải). Minh họa file .pdf:



- Để đóng chương trình, ta vào lại cmd khi nãy và nhấn Ctrl + C.

7. ĐÁNH GIÁ MỨC ĐỘ HOÀN THÀNH:

7.1. Về yêu cầu đồ án:

- Mức độ hoàn thành yêu cầu đồ án: **100%**

- Mức độ hoàn thành mục tiêu của nhóm: **100%**

- Cụ thể:

Công việc	Mức độ hoàn thành
Hiển thị trang index.html và cho phép người dùng đăng nhập	100%
Sử dụng phương thức POST gửi dữ liệu đăng nhập lên Web Server	100%
Chuyển hướng đến trang info.html khi đăng nhập thành công	100%
Chuyển hướng đến trang 404.html khi đăng nhập thất bại	100%
Cho phép download files từ trình duyệt bằng phương thức GET	100%
Cho phép download files từ trình duyệt bằng phương thức POST	100%
Cho phép tải và hiển thị hình ảnh trang Web	100%

7.2. Điểm phát triển đồ án từ nhóm:

- Vận dụng và kết hợp đồ án trước (Wireshark) với đồ án hiện tại.

- Ngoài việc cho phép người dùng download bằng phương thức GET và POST, chương trình còn cho phép người dùng xem trực tuyến các file hình ảnh, pdf, âm thanh mp3, video mp4 trực tiếp trên Web.

- Điều hướng được folder cũng như các file giữa các thư mục và các file với nhau (To Parent Directory).

- Lấy thông tin và hiện danh sách file tự động (lập trình động) điều này giúp dễ bảo trì cũng như phát triển sau này.

TÀI LIỆU THAM KHẢO

[1] Slides bài giảng *Protocolo HTTP*, Sistemas Informáticos I.

[2] Giáo trình *Mạng máy tính*, Trường ĐH Khoa học Tự nhiên, ĐHQG-HCM.

[3] Sách *Java Network Programming*, 4th Edition, Elliote Rusty Harold.

[4] Các chủ đề liên quan đến *Socket, Java, HTML* trên https://stackoverflow.com/

[5] Blog *Create a simple HTTP Web Server in Java*, Sylvain Saurel.