

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



# BÁO CÁO ĐỒ ÁN SOCKET HTTP WEB PROXY SERVER

MÔN HỌC: MẠNG MÁY TÍNH  
GIẢNG VIÊN LÝ THUYẾT: LÊ HÀ MINH

HCM, 25/08/2023

# Mục lục

<b>LỜI NÓI ĐẦU</b>	<b>2</b>
<b>1 THÔNG TIN THÀNH VIÊN</b>	<b>2</b>
<b>2 MỨC ĐỘ HOÀN THÀNH</b>	<b>2</b>
<b>3 KỊCH BẢN GIAO TIẾP CỦA CHƯƠNG TRÌNH</b>	<b>3</b>
3.1 Giao thức trao đổi . . . . .	3
3.2 Cấu trúc thông điệp . . . . .	3
3.2.1 HTTP Request Message . . . . .	3
3.2.2 HTTP Response Message . . . . .	4
3.3 Kiểu dữ liệu thông điệp . . . . .	6
3.4 Tổ chức cơ sở dữ liệu . . . . .	6
3.5 Kịch bản . . . . .	7
<b>4 MÔI TRƯỜNG LẬP TRÌNH</b>	<b>8</b>
<b>5 HƯỚNG DẪN SỬ DỤNG CHƯƠNG TRÌNH</b>	<b>8</b>
<b>6 BẢNG PHÂN CÔNG CÔNG VIỆC</b>	<b>14</b>
<b>7 TÀI LIỆU THAM KHẢO</b>	<b>14</b>

## LỜI NÓI ĐẦU

Cảm ơn thầy Minh đã truyền đạt lại cho chúng em những kiến thức cần thiết của môn Mạng máy tính và để nhóm em có thể hoàn thành bài báo cáo này.

Tuy đã cố gắng thực hiện báo cáo một cách tỉ mỉ nhưng không tránh khỏi việc bỏ sót một số chi tiết hoặc gặp phải một số lỗi không đáng có. Vì vậy, chúng em rất mong nhận được sự xem xét và đóng góp từ thầy, cô để tiếp tục cải thiện và phát triển để hoàn thiện hơn trong tương lai.

## 1 THÔNG TIN THÀNH VIÊN

MSSV	HỌ VÀ TÊN
22127317	TRẦN KHÁNH NHƯ
22127384	DƯƠNG QUANG THẮNG
22127398	NGUYỄN VĂN MINH THIÊN

## 2 MỨC ĐỘ HOÀN THÀNH

YÊU CẦU	MỨC ĐỘ HOÀN THÀNH
Cho phép xử lý các phương thức: GET, POST, HEAD	100%
Không cho phép: PUT, PATCH, DELETE, OPTIONS ...	100%
Phản hồi mã 403 và nội dung trang 403	100%
Cache images	100%
Whitelisting	100%
Giới hạn truy cập theo thời gian	100%
Xử lý yêu cầu đồng thời	100%
Config file	100%
Báo cáo	100%
Sử dụng Connection là: Keep-alive để xử lý trường hợp Content-Length và "Transfer-Encoding: chunked"	100%

### 3 KỊCH BẢN GIAO TIẾP CỦA CHƯƠNG TRÌNH

#### 3.1 Giao thức trao đổi

- Giao thức trao đổi giữa Client và Server của tầng Transport được sử dụng trong đồ án là HTTP/1.1

#### 3.2 Cấu trúc thông điệp

##### 3.2.1 HTTP Request Message

### HTTP request message

- two types of HTTP messages: *request, response*
- **HTTP request message:**
  - ASCII (human-readable format)

```

request line (GET, POST, HEAD commands) → GET /index.html HTTP/1.1\r\n
                                         Host: www-net.cs.umass.edu\r\n
                                         User-Agent: Firefox/3.6.10\r\n
                                         Accept: text/html,application/xhtml+xml\r\n
                                         Accept-Language: en-us,en;q=0.5\r\n
                                         Accept-Encoding: gzip,deflate\r\n
                                         Accept-Charset: ISO-8859-1,utf-8;q=0.7\r\n
                                         Keep-Alive: 115\r\n
                                         Connection: keep-alive\r\n
header lines → \r\n
carriage return, line feed at start of line indicates end of header lines

```

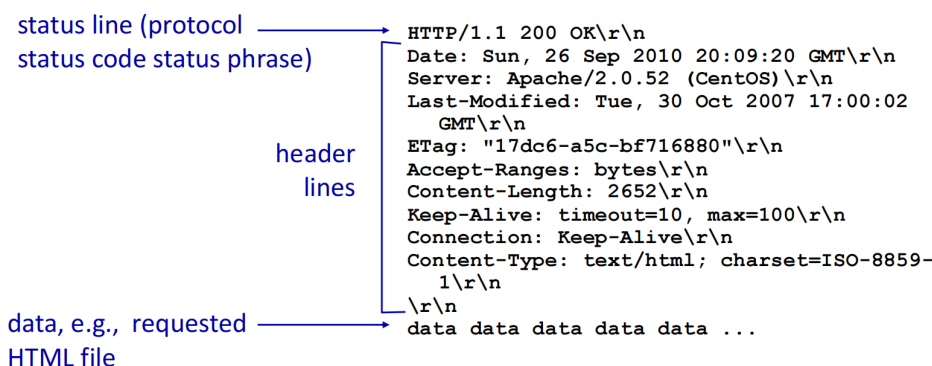
\* Check out the online interactive exercises for more examples: [http://gaia.cs.umass.edu/kurose\\_ross/interactive/](http://gaia.cs.umass.edu/kurose_ross/interactive/)

- **Request line:** chứa thông tin để gửi tới server, giúp server hiểu yêu cầu của client và thực hiện các hành động liên quan. Gồm ba phần cách nhau bởi khoảng trắng:
  - **HTTP Method** (Phương thức HTTP): chỉ định loại hoạt động mà client muốn thực hiện trên tài nguyên được xác định bởi URI, cụ thể như các phương thức:
    - + **GET** cho biết Request chỉ cần lấy dữ liệu (tài nguyên).
    - + **POST** được sử dụng để cho biết Request là gửi dữ liệu lên server.
    - + **HEAD** tương tự như GET, nhưng chỉ yêu cầu máy chủ gửi thông tin tiêu đề của tài liệu, không bao gồm nội dung thực sự.
  - **URI:** là địa chỉ định danh của tài nguyên.
  - **HTTP Version:** là phiên bản HTTP mà client đang sử dụng.
- **Header lines:** bổ sung thêm thông tin liên quan đến yêu cầu và về chính client.

- **Host:** chỉ ra host (domain, IP) và cổng của server mà request gửi đến.
  - **User-Agent:** cho phép server xác định được thông tin về trình duyệt hoặc ứng dụng gửi yêu cầu.
  - **Accept:** định dạng nội dung mà client có thể xử lý như văn bản, hình ảnh hoặc video.
  - **Accept-Language:** xác định ngôn ngữ mà client ưa thích cho phản hồi để server cung cấp nội dung phù hợp với ngôn ngữ của client.
  - **Accept-Encoding:** các kiểu nén được client chấp nhận, cho phép server nén nội dung phản hồi trước khi gửi để tối ưu hóa tốc độ truyền tải.
  - **Accept-Charset:** xác định bộ ký tự mà client muốn nhận được để server có thể cung cấp nội dung được mã hóa bằng bộ ký tự phù hợp.
  - **Keep-Alive:** chỉ ra thời gian duy trì kết nối và số lượng tối đa của kết nối, giúp giảm thiểu việc thiết lập lại kết nối cho các yêu cầu tiếp theo.
  - **Connection:** cách client muốn xử lý kết nối sau khi nhận phản hồi. Nếu giá trị là "Keep-Alive", client muốn duy trì kết nối để sử dụng cho các yêu cầu tiếp theo.
- **End of header lines:** một dòng trống (CR+LF hoặc LF) được sử dụng để đánh dấu sự kết thúc của phần tiêu đề và bắt đầu phần nội dung của yêu cầu.
  - **Body:** nội dung của yêu cầu.

### 3.2.2 HTTP Response Message

## HTTP response message



- **Status line:** dòng trạng thái, gồm có ba phần:
  - **HTTP-version:** phiên bản HTTP cao nhất mà server hỗ trợ.

- **Status-Code:** mã trạng thái của phản hồi, là một số nguyên ba chữ số mô tả kết quả của yêu cầu.
- **Reason-Phrase:** mô tả về Status-Code.

- **Header lines:**

- **Date:** chứa thông tin ngày tháng thông tin mà response được phát sinh.
- **Server:** chứa thông tin về máy chủ web đang xử lý yêu cầu và gửi phản hồi, thường chứa tên và phiên bản của phần mềm máy chủ web.
- **Last-Modified:** thời gian lần cuối cùng mà tài nguyên đã được sửa đổi.
- **ETag:** chuỗi định danh duy nhất cho phiên bản hiện tại của tài nguyên. Nó thường được sử dụng để kiểm tra xem tài nguyên đã thay đổi hay chưa.
- **Accept-Ranges:** cho biết máy chủ có hỗ trợ việc tải tài nguyên theo từng phần (range) hay không. Nếu giá trị là "bytes", nghĩa là máy chủ hỗ trợ việc chỉ định khoảng byte cụ thể khi tải tài nguyên.
- **Content-Length:** chỉ ra kích thước của phần nội dung (body) của phản hồi, thường được tính bằng số byte, giúp bên client biết được kích thước cụ thể của phản hồi để có thể xử lý nó một cách chính xác.
- **Keep-Alive:** xác định cấu hình kết nối duy trì với máy chủ. Nó có thể chứa thời gian duy trì kết nối và số lượng tối đa của kết nối duy trì.
- **Connection:** chỉ ra cách máy khách muốn xử lý kết nối với máy chủ sau khi nhận phản hồi. Nếu giá trị là "Keep-Alive", máy khách muốn duy trì kết nối để sử dụng cho các yêu cầu tiếp theo.
- **Content-Type:** xác định loại nội dung của phần body của phản hồi, cho biết mãng nội dung cụ thể của phần nội dung và thường sẽ chứa các thông số như kiểu nội dung (text, hình ảnh, video...) và bộ mã hóa (charset).

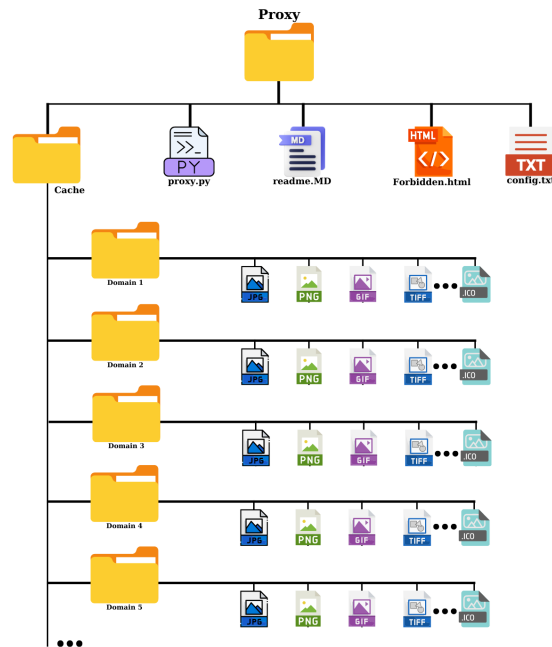
- **End of header lines:** một dòng trống (CR+LF hoặc LF) được sử dụng để đánh dấu sự kết thúc của phần tiêu đề và bắt đầu phần nội dung của phản hồi.
- **Data:** nội dung của phản hồi.

Tóm lại, các thông điệp đều tuân theo chuẩn RFC và thông điệp HTTP. Mỗi thông điệp đều bao gồm phần Header chứa các trường cung cấp thông tin nhằm hỗ trợ việc giao tiếp giữa client và server. Ngoài các trường thông tin nêu trên, còn có các trường thông tin khác có thể xuất hiện tùy thuộc vào mục đích của yêu cầu hoặc phản hồi, phiên bản giao thức, khả năng hỗ trợ của client và server, ... Nếu thông điệp có phần thân (body), nó sẽ được tách riêng với phần Tiêu đề bằng một dòng trống.

### 3.3 Kiểu dữ liệu thông điệp

- Thông điệp được mã hoá dưới dạng ASCII.

### 3.4 Tổ chức cơ sở dữ liệu



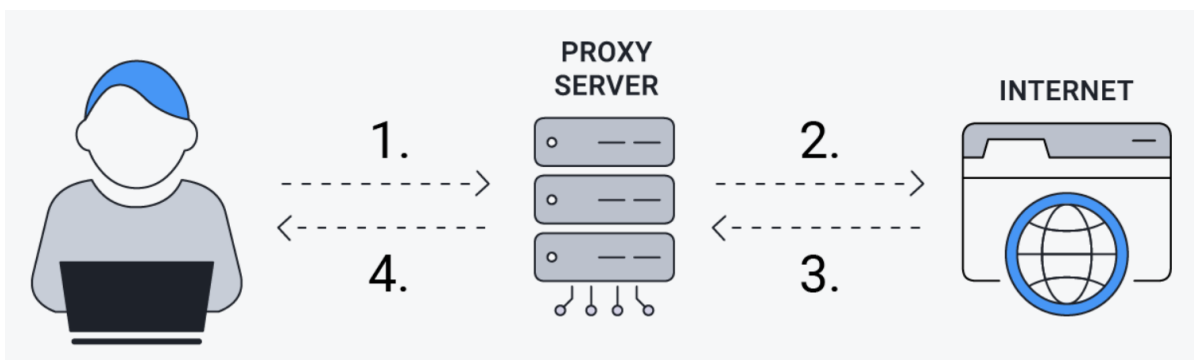
Trong thư mục Proxy chứa:

- proxy.py - lưu trữ source code của chương trình.
- Forbidden.html - Trang web trả về mã lỗi 403.
- config.md - chứa thông tin về time, whitelisting, cache-time.
- thư mục Cache - gồm các thư mục con "domain" lưu trữ hình ảnh nhận được từ các trang web.

### 3.5 Kịch bản

Kịch bản của chương trình xảy ra như sau:

1. Khởi chạy chương trình. Chương trình sẽ lắng nghe các kết nối (mặc định là 127.0.0.1:8888).
2. Tiến hành truy cập 1 trang website. Lúc này, browser sẽ gửi HTTP Request đến chương trình Proxy thay vì gửi cho server.
3. Chương trình Proxy nhận được Request, đọc và phân tích gói tin. Theo thiết lập, chương trình chỉ chấp nhận các gói tin có HTTP Method là GET, POST, HEAD. Nếu phát hiện thấy các Method khác thì gói tin sẽ không được xử lý. Nếu Method hợp lệ thì các trường hợp xảy ra:
  - Nếu đó là gói tin GET truy vấn 1 dữ liệu hình ảnh. Lúc này Proxy sẽ kiểm tra trong cache ảnh này được lưu trong cache hay chưa. Nếu ảnh đã được lưu trong cache. Proxy sẽ lấy ảnh từ cache và gửi về cho browser. Nếu ảnh chưa được lưu, Proxy sẽ chuyển tiếp gói tin cho server.
  - Nếu không phải dữ liệu ảnh hoặc gói tin thuộc Method khác, gói tin được chuyển tiếp cho server.
4. Proxy nhận lại HTTP Response từ server. Nếu Response là dữ liệu ảnh sẽ được lưu vào cache của Proxy. Response sau đó được chuyển về lại browser. Việc này giúp giảm tải cho server và tăng tốc độ truy cập cho người dùng, vì ảnh đã được lưu trữ tại Proxy.
5. Trình duyệt của người dùng nhận Response từ Proxy và hiển thị nội dung tương ứng trên trang web. Nếu Response là dữ liệu ảnh, trình duyệt sẽ hiển thị ảnh. Nếu là nội dung văn bản, trình duyệt sẽ hiển thị văn bản và các phần khác của trang web.
6. Sau khi trình duyệt hiển thị nội dung, quá trình giao tiếp giữa các thành phần kết thúc. Proxy vẫn tiếp tục lắng nghe các kết nối đến từ trình duyệt, sẵn sàng xử lý các yêu cầu mới. Trình duyệt vẫn có thể tiếp tục truy cập các trang web khác hoặc thực hiện các hoạt động khác.



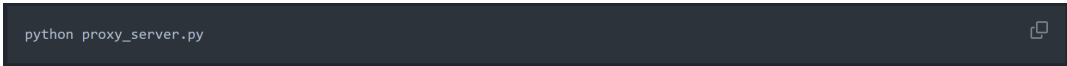


## 4 MÔI TRƯỜNG LẬP TRÌNH

- Ứng dụng được viết trên hệ điều hành Windows 10.
- Ngôn ngữ lập trình: Python 3.
- Trình biên tập mã nguồn: VSCode.
- Trình quản lý mã nguồn: Git.
- Các dữ liệu được lưu trữ dưới dạng file txt.
- Thư viện hỗ trợ socket Python: socket, sys, time, threading, os, timeout

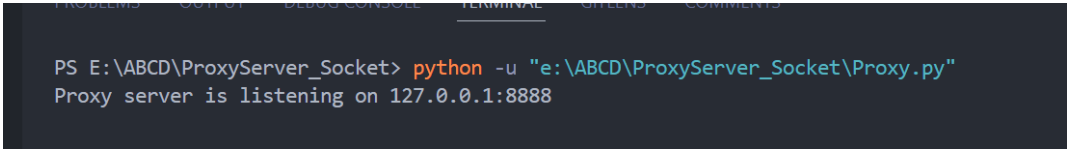
## 5 HƯỚNG DẪN SỬ DỤNG CHƯƠNG TRÌNH

- **Cài đặt và chuẩn bị môi trường:** đảm bảo bạn đã cài đặt Python trên máy tính.
- **Phía Proxy Server:**
  - Khởi động Proxy Server trên file Python để các Client có thể tìm đến mà kết nối. Nếu không làm bước này, tất cả các Client sẽ không hoạt động được.



```
python proxy_server.py
```

- Máy chủ Proxy sẽ bắt đầu lắng nghe trên máy chủ và cổng được chỉ định (mặc định là 127.0.0.1:8888).



```
PS E:\\ABCD\\ProxyServer_Socket> python -u "e:\\ABCD\\ProxyServer_Socket\\Proxy.py"
Proxy server is listening on 127.0.0.1:8888
```

- Phần console của Proxy Server sẽ hiển thị các Host của URL, cập nhật các URL được gửi đến.
- **Phía Client:**
  - Thiết lập proxy của trình duyệt: IP và cổng của proxy server (mặc định là 127.0.0.1:8888).

Connection Settings ✕

---

**Configure Proxy Access to the Internet**

☐ No proxy  
☐ Auto-detect proxy settings for this network  
☐ Use system proxy settings  
☒ **Manual proxy configuration**

HTTP Proxy  Port   
☒ Also use this proxy for HTTPS

HTTPS Proxy  Port   
 SOCKS Host  Port   
☐ SOCKS v4 ☒ SOCKS v5

☐ Automatic proxy configuration URL

No proxy for

Example: .mozilla.org, .net.nz, 192.168.1.0/24  
 Connections to localhost, 127.0.0.1/8, and ::1 are never proxied.

– Thiết lập file config:

```
cache_time=30
whitelisting=oosc.online, example.com
time=8-24
```

- + **cache\_time**: thời gian cho phép dữ liệu được lưu trữ tại cache, nếu dữ liệu được lưu có thời gian tồn tại vượt quá phạm vi cho phép, dữ liệu sẽ được xóa trong lần chạy tiếp theo.
- + **whitelisting**: nơi lưu trữ các tên miền URL cho phép truy cập, nếu tên miền nằm ngoài phạm vi của whitelisting client sẽ không thể truy cập và trả về lỗi 403.
- + **time**: thời gian cho phép hoạt động theo thời gian thực, nếu quá thời gian cho phép hoạt động. Client sẽ bị từ chối truy cập và trả về lỗi 403.

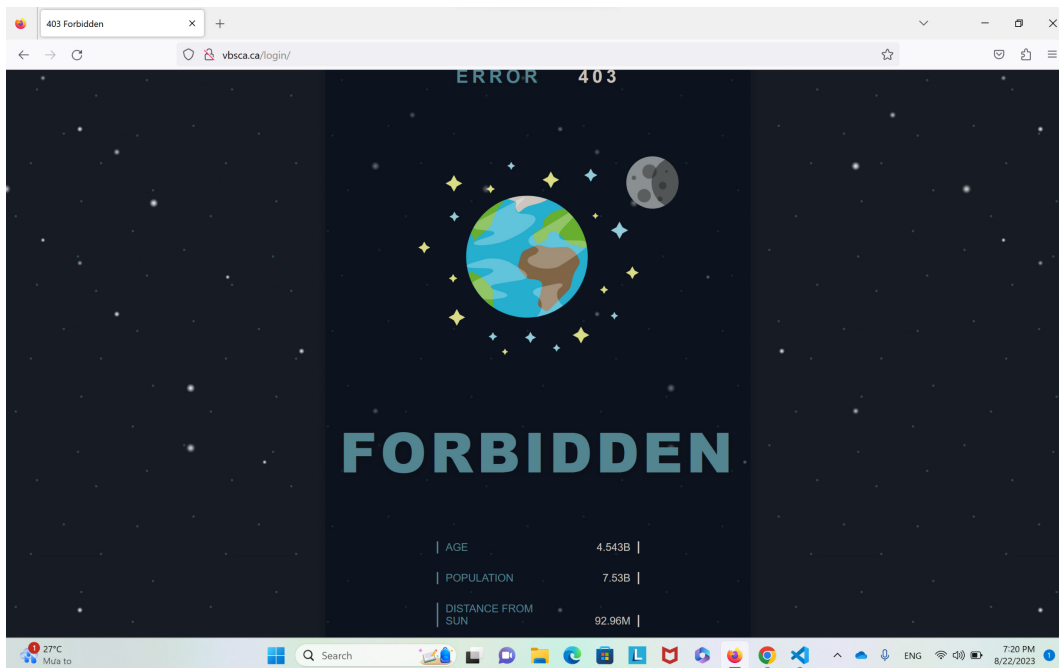
=> Các phạm vi tùy người sử dụng thiết lập.

- Hướng dẫn sử dụng 'whitelisting':

- Client có thể tùy chỉnh tên miền, thời gian và cache\_time trong whitelisting.
- Giả sử whitelisting:

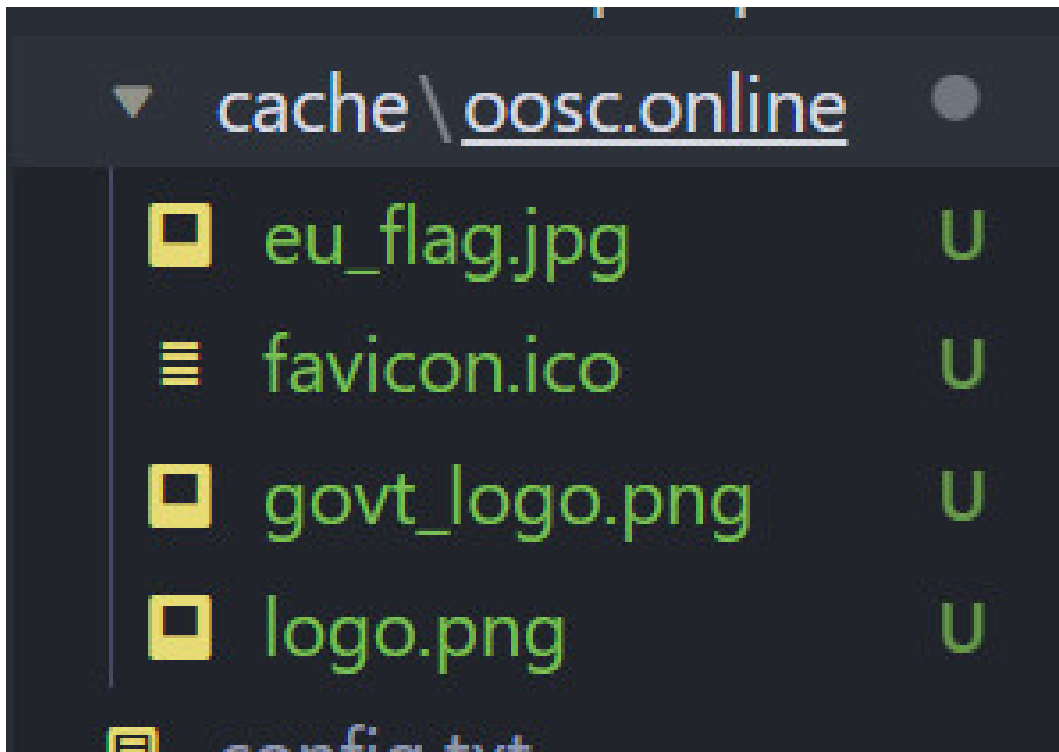
```
cache_time=30  
whitelisting=oosc.online, example.com  
time=8-24
```

- Máy chủ proxy chỉ cho phép client truy cập các tên miền có trong whitelisting. Nếu truy cập vào tên miền không tồn tại trong whitelisting, client sẽ bị từ chối truy cập và trả về lỗi 403.
- Ví dụ khi truy cập vào trang <http://vbsca.ca/login/> không tồn tại trong whitelisting, client sẽ bị từ chối truy cập và trả về lỗi 403.



- Hướng dẫn sử dụng 'cache\_time':

- Người dùng có thể tùy chỉnh thời gian dữ liệu được lưu trong cache (đơn vị tính bằng giây).
- Nếu tùy chọn cache\_time được đặt thành 900, nghĩa là dữ liệu sẽ được lưu trong cache trong 900 giây (15 phút) trước khi bị xóa.
- Giả sử cache đang lưu trữ dữ liệu như dưới đây:

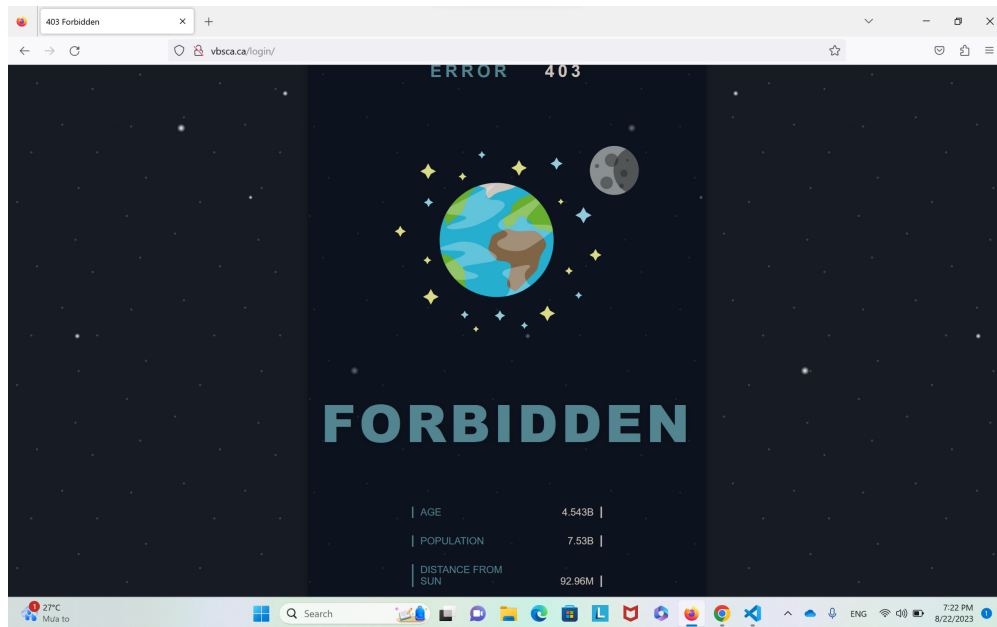


Sau *time* giây dữ liệu sẽ bị xóa khỏi cache:

```
Removed expired cache file: cache\oosc.online\eu_flag.jpg
Removed expired cache file: cache\oosc.online\favicon.ico
Removed expired cache file: cache\oosc.online\govt_logo.png
Removed expired cache file: cache\oosc.online\illustration.png
Removed expired cache file: cache\oosc.online\logo.png
```

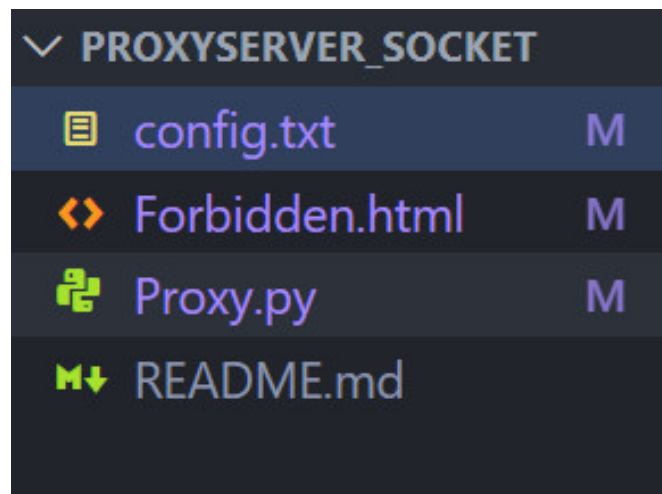
- **Hướng dẫn sử dụng 'time':**

- Người sử dụng có thể tùy chỉnh thời gian truy cập, proxy server sẽ cho phép truy cập trong thời gian đã định. Nếu thời gian truy cập nằm ngoài phạm vi cho phép của proxy server. Client sẽ trả về lỗi 403.



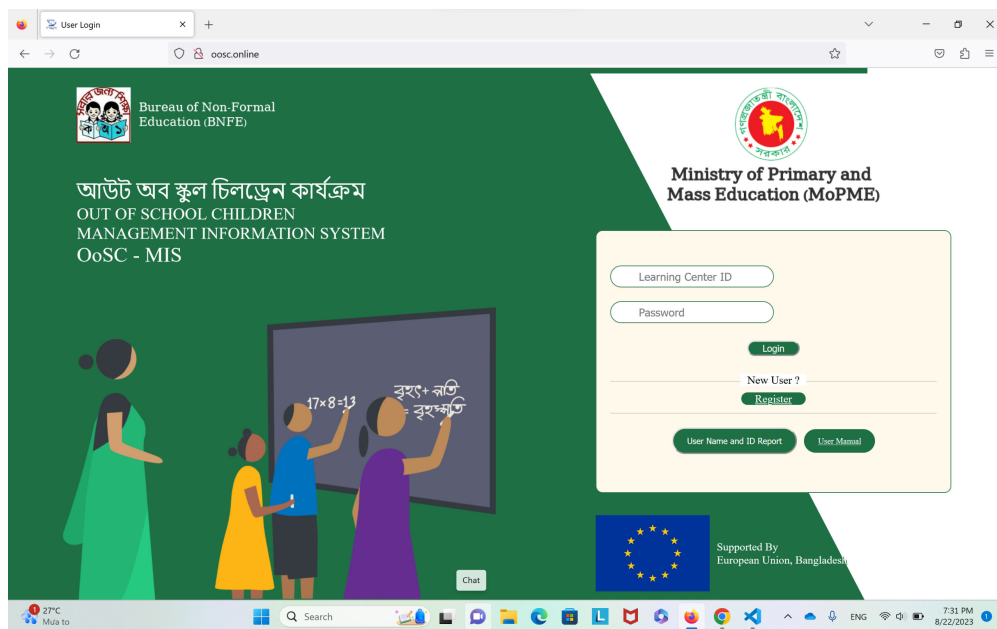
- **Hướng dẫn cách sử dụng cache:**

- Ban đầu sẽ không tồn tại thư mục cache.

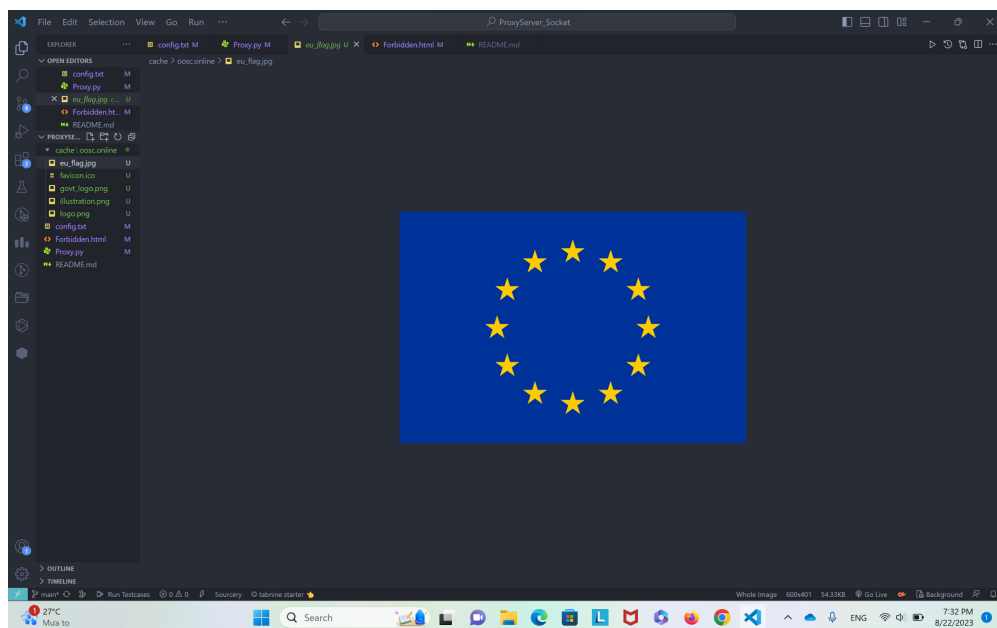


- Khi truy cập vào URL có tên miền nằm trong whistlisting. Thư mục cache sẽ tự động được tạo. Sau đó thư mục con của thư mục cache sẽ được tạo với tên thư mục là tên miền URL để lưu trữ dữ liệu của URL đã truy cập gửi về.

– Client:



– Proxy server:



## 6 BẢNG PHÂN CÔNG CÔNG VIỆC

YÊU CẦU	NGƯỜI THỰC HIỆN
Cho phép xử lý các phương thức: GET, POST, HEAD	Như, Thiện
Không cho phép: PUT, PATCH, DELETE, OPTIONS ...	Thiện
Phản hồi mã 403 và nội dung trang 403	Như
Cache images	Thắng
Whitelisting	Thiện
Giới hạn truy cập theo thời gian	Thiện
Xử lý yêu cầu đồng thời	Như
Config file	Thắng
Báo cáo	Thắng
Sử dụng Connection là: Keep-alive để xử lý trường hợp Content-Length và "Transfer-Encoding: chunked"	Thắng

## 7 TÀI LIỆU THAM KHẢO

- Computer Networking A Top-Down Approach, 7th Edition by James Kurose, Keith Ross
- <https://docs.python.org/3/howto/sockets.html>
- <https://docs.python.org/3/library/threading.html#threading.Thread.join>
- <https://www.rfc-editor.org/rfc/rfc9110GET>
- <https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods/POST>
- <https://www.nginx.com/blog/nginx-caching-guide/>