Bài giảng

PHÁT TRIỂN ỨNG DỤNG WEB

Lê Đình Thanh
Khoa Công nghệ Thông tin
Trường Đại học Công nghệ, ĐHQGHN
E-mail: thanhld@vnu.edu.vn Mobile: 0987.257.504

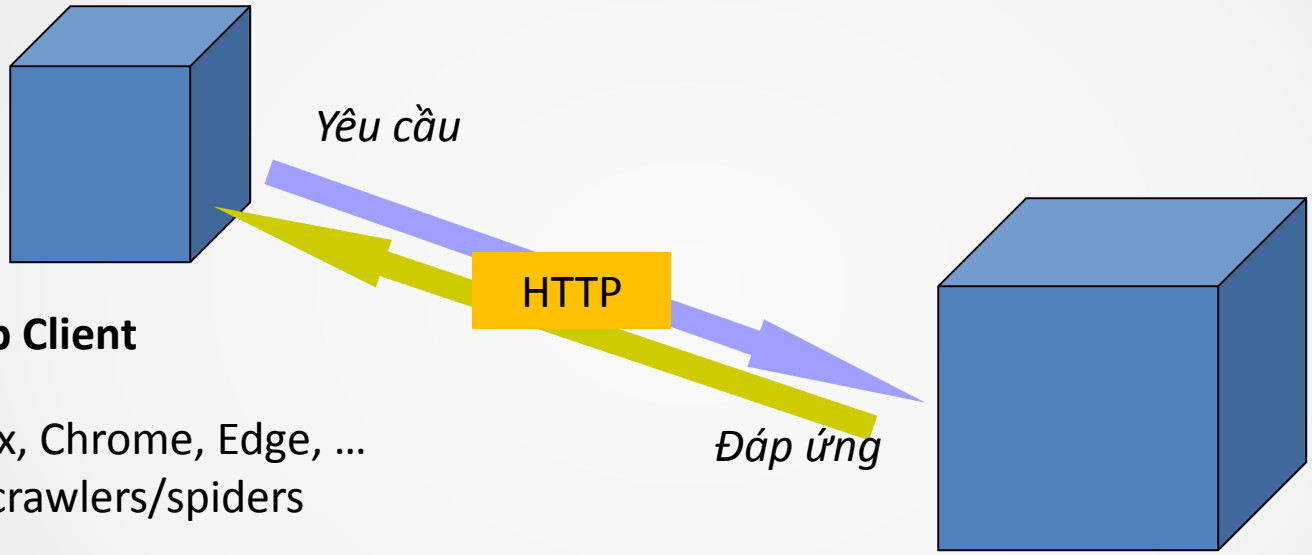
Chương 1

Kiến trúc ứng dụng web

Nội dung

- Kiến trúc của ứng dụng Web
- HTTP
- URL
- HTTP Request
- HTTP Response

Kiến trúc Web



Web Client

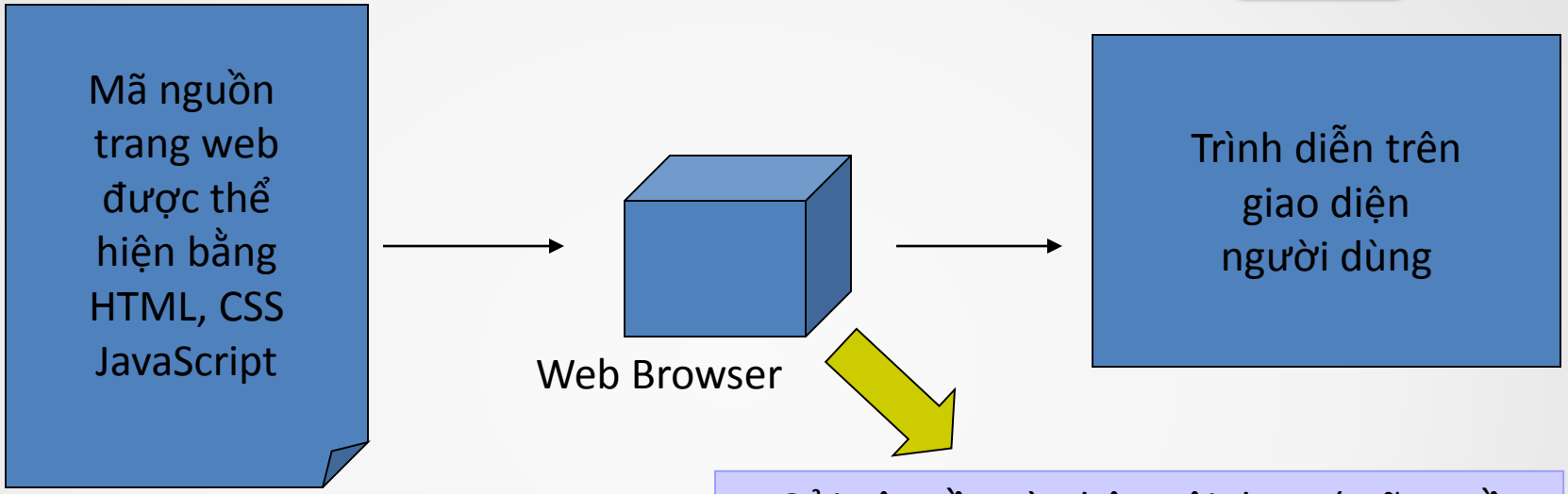
Firefox, Chrome, Edge, …
Web crawlers/spiders

Web Server

Nginx, IIS,
Apache, …

Web Browsers

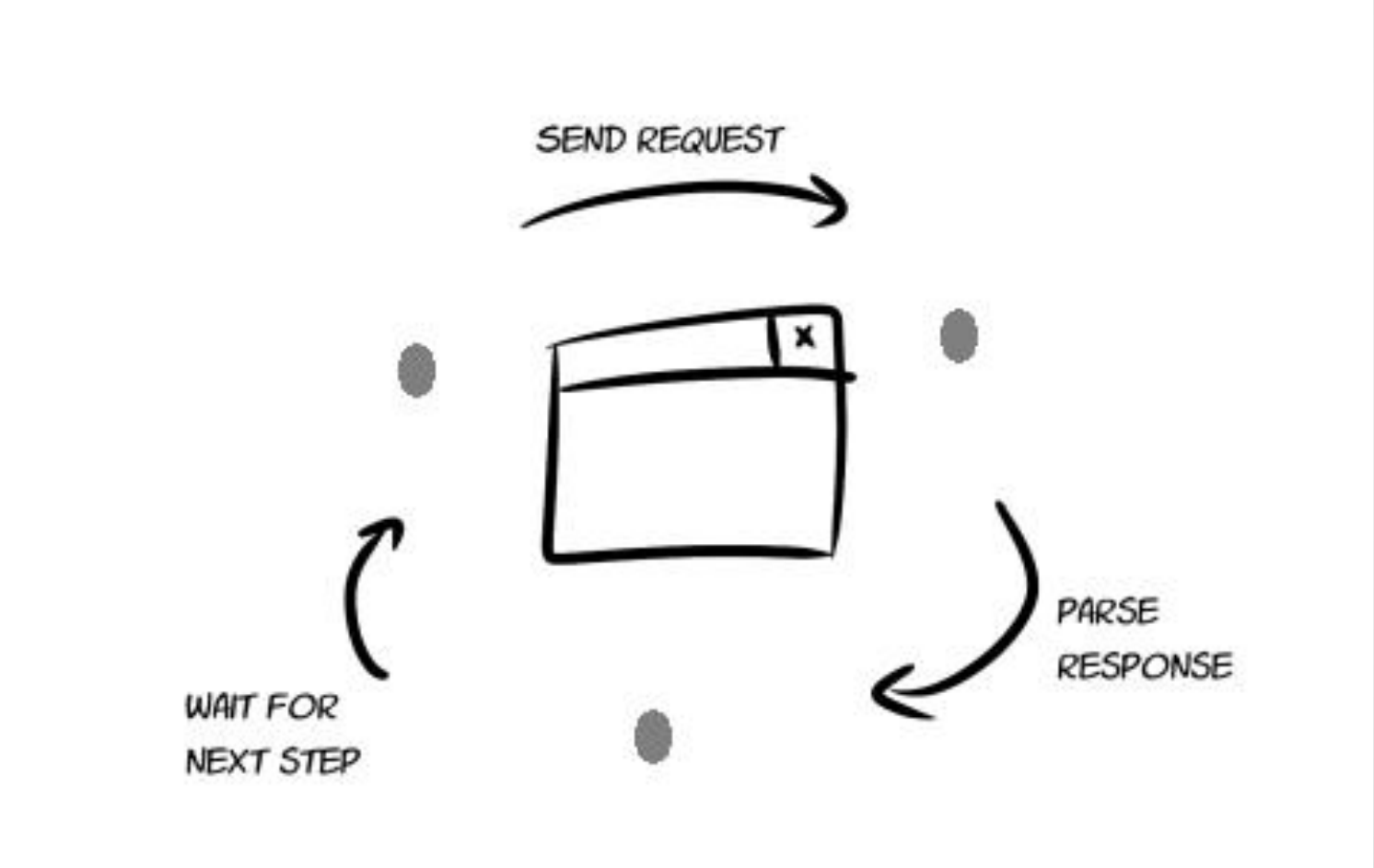
Web Browser



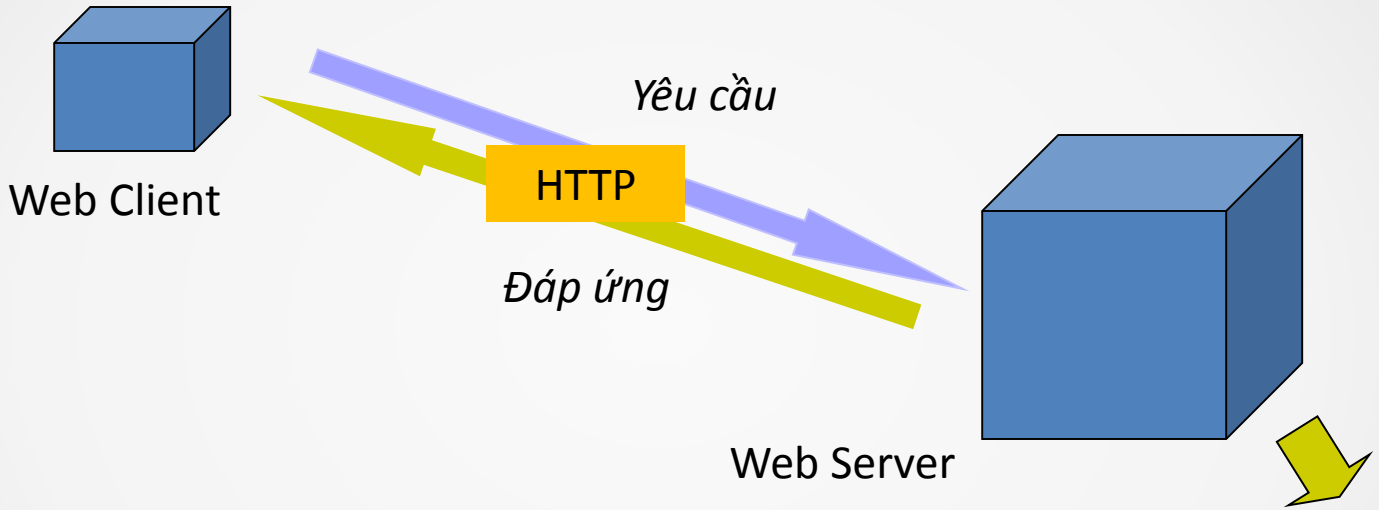
Do Web Server gửi tới.

- Gửi yêu cầu và nhận nội dung (mã nguồn HTML, CSS, JavaScript) trang web từ Web Server.
- Thông dịch mã nguồn trang web và trình diễn kết quả trên giao diện người dùng.

Web Browsers: Chu trình RPW (Request-Parse-Wait)

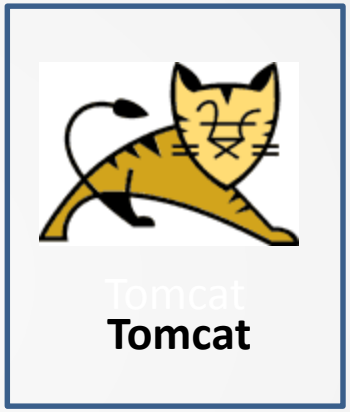


Web Server



- Nhận yêu cầu của Web Client, chuyển cho App Server
- Nhận trang web từ App Server rồi gửi cho Web client.

Web Servers



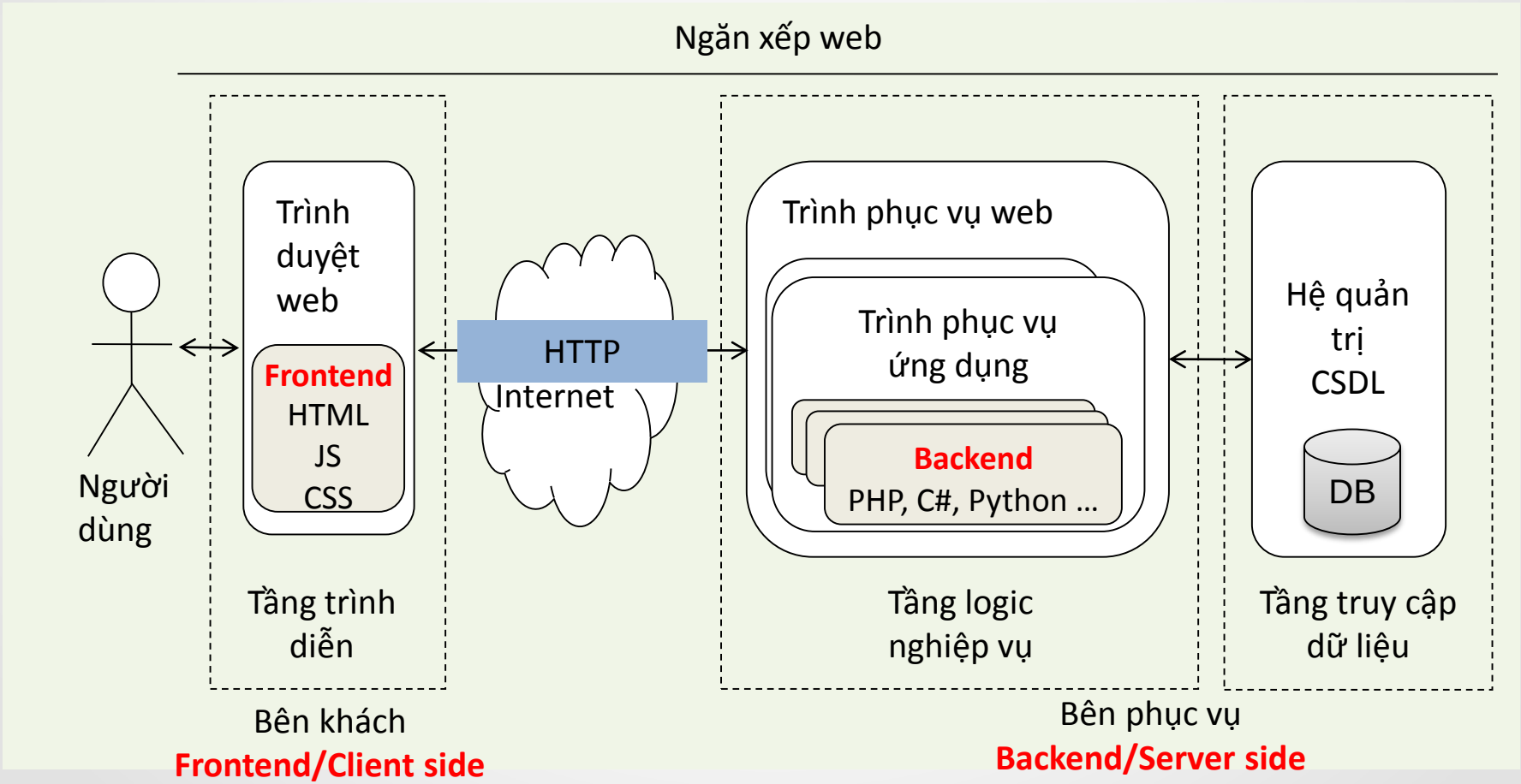
ASP, ASP.NET

PHP

JSP

NGINX

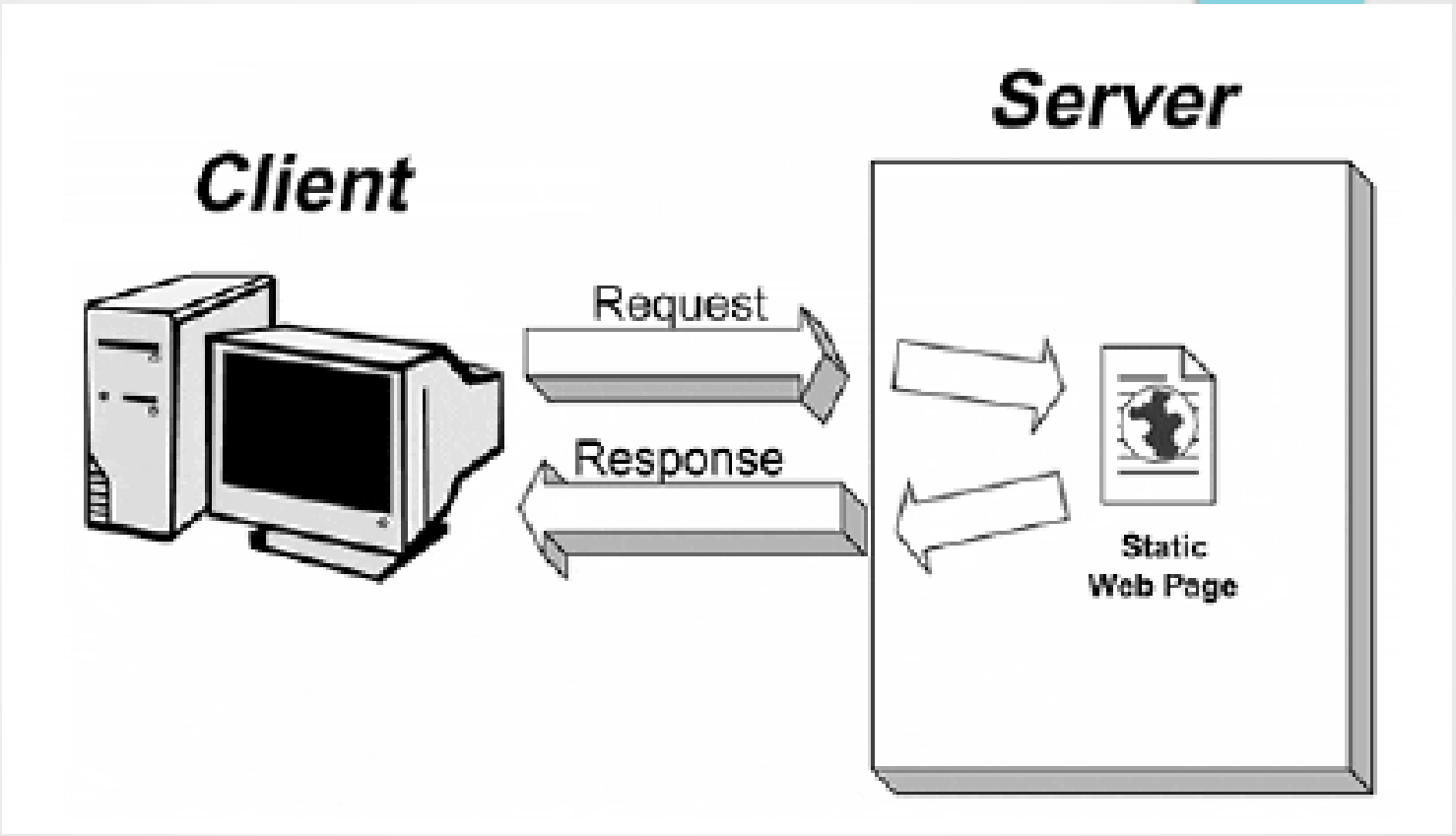
Khung nhìn bao quát



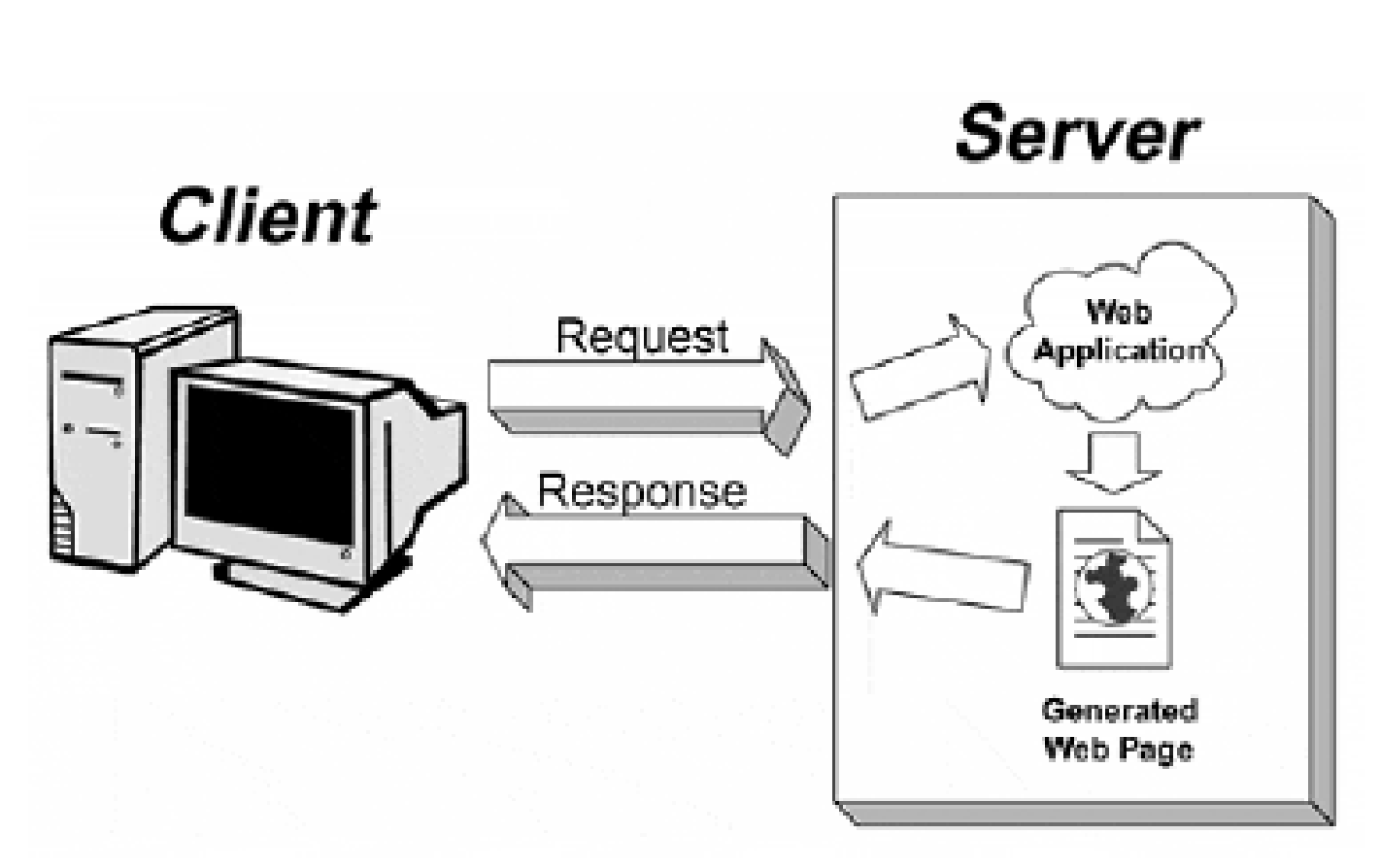
Phân loại web

- Static content: Nội dung web (HTML, JS, CSS, media, JSON, XML, …) được lưu sẵn trên server.
- Dynamic content: Nội dung web được tạo ra khi có yêu cầu.
- Có nhiều công nghệ web động như PHP, ASP.NET, JSP, ...

Web tĩnh



Web động


Server
Client
Request
Response
Web Application
Generated Web Page

Hosting ứng dụng web

- Web Server quản lý tất cả các ứng dụng web
- Các ứng dụng web có thể được tổ chức theo nhóm ứng dụng (Application Pool)
- Mỗi ứng dụng web sử dụng một hoặc nhiều socket (địa chỉ IP:cổng) để giao tiếp với client (IP-based hosting).
- Nhiều ứng dụng web có thể sử dụng cùng một socket với điều kiện mỗi ứng dụng được gắn một tên miền khác nhau (Name-based hosting).
- Tổng quát: Mỗi ứng dụng web có thể được buộc (binding) nhiều định danh sử dụng lược đồ (scheme, là http hoặc https), địa chỉ IP, số hiệu cổng (TCP port) và tên máy (hostname)

Ví dụ hosting ứng dụng web

- IP-based
  - http://danhgiananangluc.vnu.edu.vn:86
  - http://danhgiananangluc.vnu.edu.vn:2015
- Name-based
  - http://danhgiananangluc.vnu.edu.vn
  - http://vitest.vnu.edu.vn

Tài nguyên của ứng dụng web

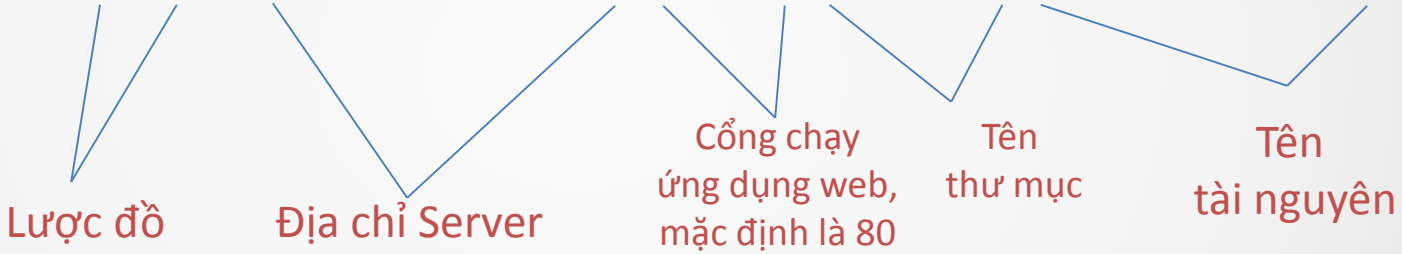
- Một ứng dụng web có các tài nguyên (tệp tài liệu, kịch bản, chương trình, …)
- Web Server
  - Application Pool 1
    - Application 1
      - Resource 1
      - Resource 2
    - Application 2
    - Application 3
  - Application Pool 2
    - Application 4
- Địa chỉ của tài nguyên được xác định bằng URL.

URL: Uniform Resource Locator

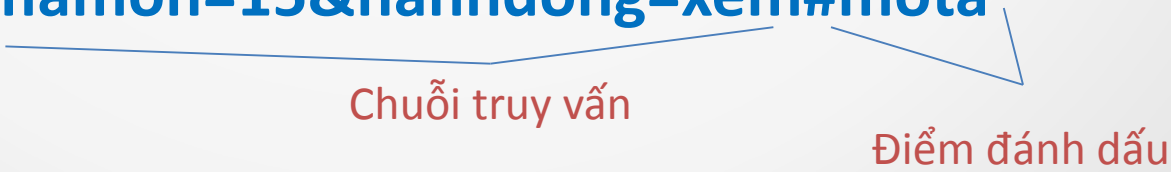
- URL hay Định vị tài nguyên đồng nhất dùng để xác định vị trí (toàn cầu) của một tài nguyên trên Internet

scheme://host:port/path?query_string#bookmark

- Ví dụ **http://uet.vnu.edu.vn:8080/daotao/monhoc.py**



?mamon=15&hanhdong=xem#mota



Biểu diễn mã phần trăm

- URL chỉ được bao gồm các ký tự ASCII in được, tức có mã từ 0x20 đến 0x7e
- Muốn đưa các ký tự dành riêng, các ký tự có mã nhỏ hơn 0x20 hoặc lớn hơn 0x70 vào URL (ví dụ giá trị tham số trong chuỗi truy vấn), cần biểu diễn theo mã phần trăm

!	#	$	&	'	(	)	*	+	,	/	:
%21	%23	%24	%26	%27	%28	%29	%2A	%2B	%2C	%2F	%3A
;	=	?	@	[	]	"	%	-	.	<	>
%3B	%3D	%3F	%40	%5B	%5D	%22	%25% 2D	%2E	%3C	%3E	

RESTful URL / URL ngữ nghĩa

URL phi ngữ nghĩa

http://example.com/index.php?page=marketing

http://example.com/products?cat=12&id=25

http://example.com/services.py?cat=legal

URL ngữ nghĩa

http://example.com/marketing

http://example.com/products/12/25

http://example.com/services/legal

HTTP (HyperText Transfer Protocol)

- HTTP là giao thức dùng để truyền siêu văn bản
- Không trạng thái: server không giữ thông tin về các lần yêu cầu trước của client.
- HTTP Request và HTTP Response…

HTTP Request

- Client gửi yêu cầu đến server bằng một thông báo yêu cầu (http request)
- Một thông báo yêu cầu bao gồm một số dòng văn bản
  - Dòng đầu tiên là dòng yêu cầu(request line) chứa ba thông số:
    - Phương thức yêu cầu (request method): GET/ HEAD/POST/PUT/DELETE/…
    - URI
    - Phiên bản HTTP được sử dụng
  - Các dòng tiêu đề chứa thông tin về các kiểu MINE, tập ký tự được chấp nhận, phiên bản trình duyệt, hệ điều hành sử dụng trên client, …
  - Dòng trắng (chỉ có dấu kết thúc dòng)

HTTP Request – Ví dụ

```
POST /sum.py HTTP/1.1
host: calculator.io
User-Agent: Mozilla/5.0 (Ubuntu; Linux x86_64;
   rv:130.0) Gecko/20100101 Firefox/130.0
Accept: text/html,application/json
Accept-Language: en-US,en;q=0.5
Content-Length: 7
Content-Type: application/x-www-form-urlencoded

x=2&y=3
```

HTTP method: GET

- GET là phương thức hay được sử dụng nhất để gửi yêu cầu từ client đến server nhằm tải tài nguyên từ server về.
- Khi không chỉ rõ yêu cầu theo phương thức nào thì GET được sử dụng
- Với GET,
  - Các cặp tham số/giá trị (dữ liệu của người dùng) được đưa vào chuỗi truy vấn của URL
  - Không có body

  Ví dụ: *http://uet.vnu.edu.vn/daotao/monhoc.py?ma=24&action=1*
  - Chiều dài URL có giới hạn nên số lượng tham số là hạn chế

HTTP method: POST

- POST là phương thức khác được sử dụng để gửi yêu cầu có chứa dữ liệu từ client đến server
- Với POST,
  - Dữ liệu được đưa vào thân HTTP request.
  - Không giới hạn số lượng tham số

HTTP method: Khác

- HEAD: Tương tự GET, chỉ khác là đáp ứng từ server trả về không có thân
- TRACE: Đáp ứng trả về của server cần chứa nguyên nội dung yêu cầu mà server đã nhận
- OPTIONS: Hỏi các phương thức HTTP được server hỗ trợ cho một tài nguyên cụ thể
- PUT: Tương tự POST nhưng với mục đích cập nhật tài nguyên đã có

HTTP Request Headers

Accept cho server biết loại nội dung được chấp nhận, như loại ảnh, định dạng tài liệu văn phòng, ...

Accept-Encoding cho server biết loại mã hóa nội dung được chấp nhận.

Authorization đệ trình thông tin xác thực đối với các kiểu xác thực HTTP.

HTTP Request Headers

Cookie đệ trình cookies lên server.

Host xác định hostname xuất hiện trong URL.

If-Modified-Since cho biết thời điểm cuối trình duyệt nhaạn đươợ tài nguyên. Nếu tài nguyên không thay đổi từ thời điểm đó, server có thể báo cho client sử dụng bản đệm

HTTP Request Headers

If-None-Match xác định một thẻ thực thể, server có thể sử dụng thẻ thực thể để xác định liệu trình duyệt có thể dùng bản đệm của tài nguyên hay không.

Origin được sử dụng trong các Ajax requests nhằm cho biết domain của request

Referer cho biết URL đã chuyển đến yêu cầu này

User-Agent cung cấp thông tin phía client (HĐH, trình duyệt, phần mềm khác, ...)

HTTP Response

- Dữ liệu do server gửi về cho client được định dạng bởi HTTP Response
- Một HTTP Response bao gồm
  - *Dòng trạng thái (status line):* Giao thức được dùng, mã trạng thái và giá trị trạng thái
  - *Đầu đáp ứng (response header):* Chứa chuỗi các cặp tên/giá trị,
  - *Dòng trắng*
  - *Dữ liệu thực sự:* HTML, JavaScript, CSS, JSON, XML, …

HTTP Response – Ví dụ

```
HTTP/1.1 200 OK
Date: Sun, 15 Sep 2024 14:38:41 GMT
Server: Apache/2.4.41 (Ubuntu)
Content-Length: 25
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: application/json

{"result":"OK","value":5}
```

Mã trạng thái

- 1xx — Thông tin
- 2xx — Thành công
- 3xx — Client được chuyển hướng
- 4xx — Yêu cầu có lỗi
- 5xx – Server có lỗi

HTTP Response Headers

Access-Control-Allow-Origin cho biết có thể truy cập tài nguyên bằng Ajax request từ domain khác hay không

Cache-Control thông báo cho trình duyệt có lưu đệm hay không

ETag xác định một thẻ thực thể. Clients có thể đệ trình định danh này trong các request sau tới cùng tài nguyên trong tiêu đề If-None-Match để báo cho server biết phiên bản của tài nguyên trình duyệt đang lưu đệm

HTTP Response Headers

Expires báo cho trình duyệt biết khi nào nội dung không còn hợp lệ

Location cho biết địa chỉ chuyển hướng trình duyệt (mã trạng thái 3xx)

Pragma chứa các chỉ dẫn lưu đệm tại trình duyệt (vd. no-cache).

HTTP Response Headers

Server cung cấp thông tin về phần mềm phía server (HĐH, webserver, các modules, trình thông dịch, ...)

Set-Cookie yêu cầu trình duyệt lưu cookies và gửi lại tại các requests sau

WWW-Authenticate được sử dụng trong đáp ứng với mã trạng thái 401 để cung cấp chi tiết về kiểu xác thực

X-Frame-Options cho biết frame có được tải nội dung response hiện tại không và tải như thế nào

Tiêu đề cho cả request và response

Connection cho biết bên kia kết nối có nên duy trì kết nối TCP không sau khi truyền HTTP đã hoàn thành

Content-Encoding xác định kiểu biểu diễn mã nội dung

Content-Length cho biết độ dày theo byte của thân

Content-Type xác định kiểu nội dung chứa trong thân

Transfer-Encoding xác định biểu diễn mã được thực hiện trên thân khi truyền.

Kết nối liên tục và dẫn ống

- HTTP 1/1
- Kết nối liên tục (HTTP persistent connection)
  - Sử dụng một kết nối TCP để gửi và nhận nhiều yêu cầu/đáp ứng HTTP
  - Tất cả trình duyệt hiện đại hỗ trợ
- Dẫn ống (pipelining)
  - Gửi liên tiếp nhiều yêu cầu mà không phải đợi đáp ứng của các yêu cầu trước đó
  - Không được các trình duyệt ngày nay hỗ trợ

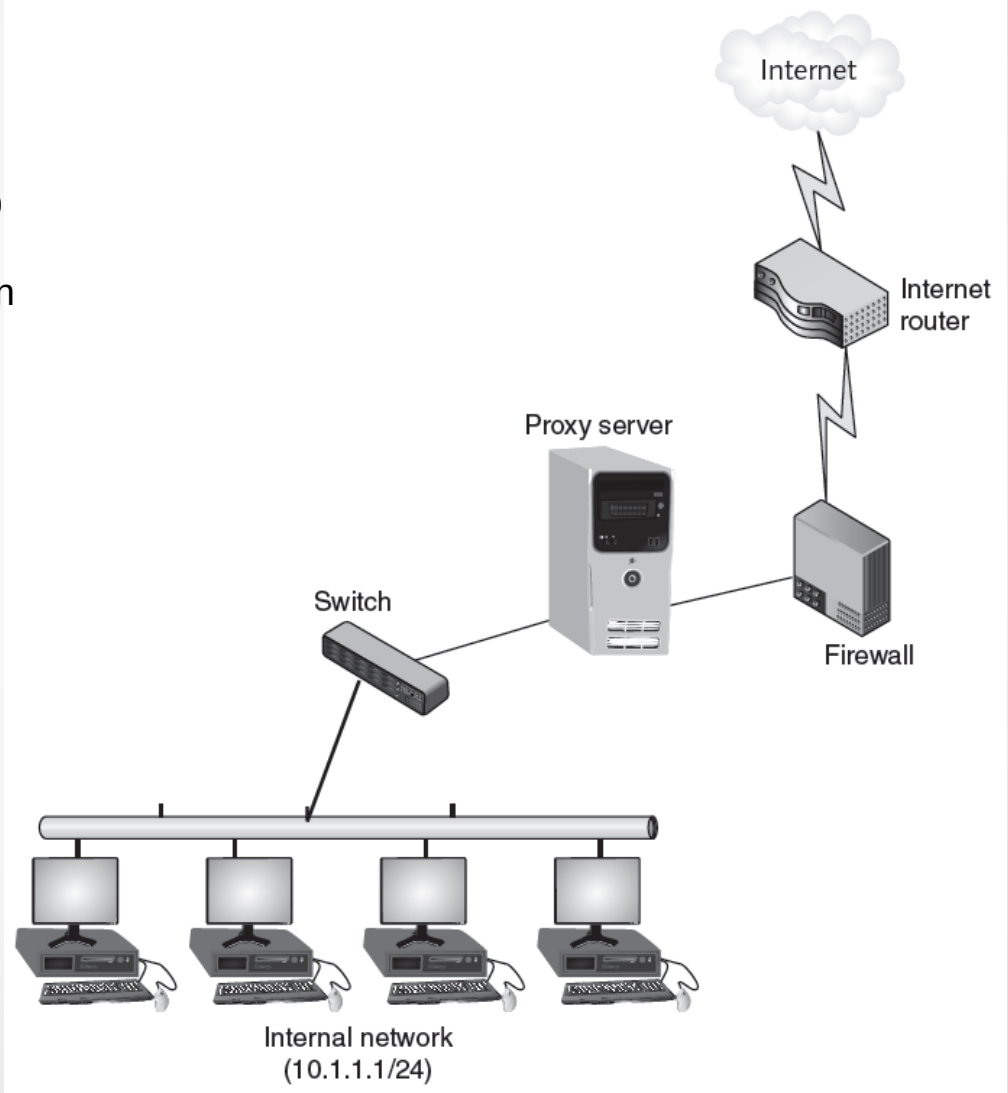
MIME Type

- Mỗi tài nguyên web có một MIME type khác nhau
  - text/html, text/javascript, text/css, image/gif, image/jpeg, image/png, audio/mp3, video/mp4, text/xml, text/json, text/plain, ...

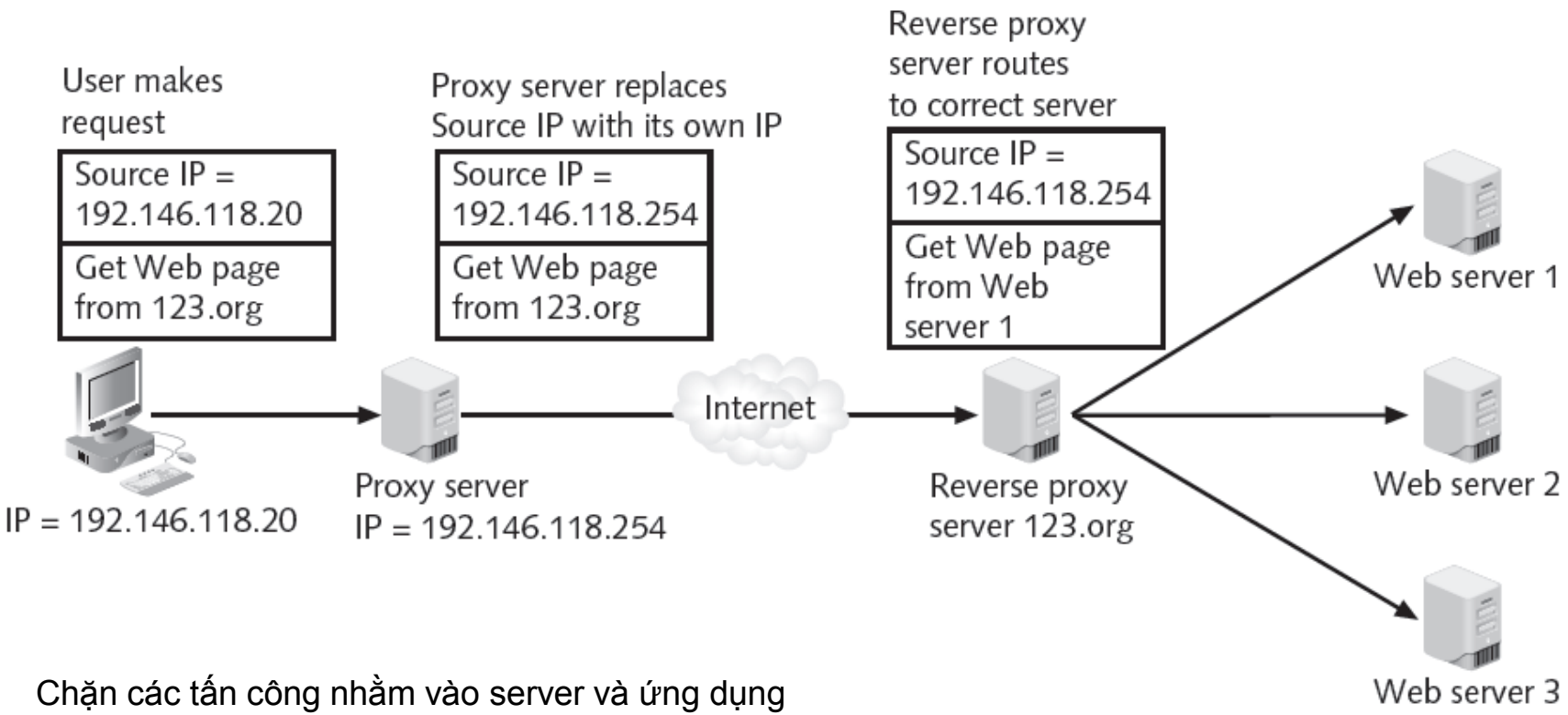
Proxy Server / Forward Proxy

Tăng tốc độ truy cập
Giảm chi phí thuê mạng
Chặn các trang web không được phép
Chặn phần mềm độc hại
Che dấu IP của các máy trong cơ quan

Ví dụ
Squid
WinGate
Ziproxy
Polipo
Apache, IIS



Reverse Proxy



Chặn các tấn công nhằm vào server và ứng dụng
Phát hiện và ngăn chặn tấn công DOS
Che giấu các lỗ hổng an ninh của server và ứng dụng
Các tiêu đề mang thông tin về server
Trang web lỗi
Công cụ

Nginx, F5 Big-IP, Varnish

API và CGI

- Webserver không biên/phiên dịch và thực thi chương trình tạo web động mà giao trách nhiệm đó cho các trình biên/phiên dịch trợ giúp
- Nếu trình biên/phiên dịch là một module của web server thì giao tiếp giữa web server và trình biên/phiên dịch được thực hiện thông qua API
  - Ví dụ Apache và mod_php, mod_python, mod_perl
- Nếu trình biên/phiên dịch không được chứa trong web server thì giao tiếp giữa web server và trình biên/phiên dịch được thực hiện qua Giao diện cổng chung (CGI – Common Gateway Interface)
  - Ví dụ Apache và ActivePerl

Các khái niệm

- Nội dung web (web content): Tài liệu HTML, JavaScript, CSS, media, JSON, PDF, …
- Tài nguyên web (web resource): Tệp, tiến trình, … có thể tạo nội dung web
- Trang web (webpage): Một tài liệu HTML cùng các tài nguyên khác như hình ảnh, âm thanh, … được nó tham chiếu
- Ứng dụng web (web application): Ứng dụng tạo ra các trang web
- Website: Nhóm các trang web được kết nối với nhau và cùng mang thông tin về một chủ đề

Các khái niệm

- Trình khách web (web client): Ứng dụng tiêu thụ/sử dụng/khai thác nội dung web
- Trình duyệt web (web browser): Web client hiển thị nội dung web trên giao diện người dùng và cho phép người dùng tương tác trên giao diện
- Add-on/plugin/extension: Thành phần phần mềm làm tăng các khả năng của trình duyệt như chạy video, applet, quét virus, ngăn chặn quảng cáo, …

Các khái niệm

- Trình phục vụ web (web server): Ứng dụng cung cấp các trang web cho web client
- Trình phục vụ ứng dụng (application server): Ứng dụng phía sau web server có nhiệm vụ cung cấp môi trường thực thi cho ứng dụng web
- Mặt trước (frontend), bên khách (client side): Là phần người dùng nhìn thấy và tương tác trên trình duyệt. Được viết bằng HTML, JavaScript và CSS, chạy trên trình duyệt.
- Mặt sau (backend), bên phục vụ (server side): Là phần người dùng không nhìn thấy, được viết bằng PHP, C#, Python, …, chạy trên app server, cung cấp nội dung cho frontend.

Các khái niệm

- Siêu văn bản (hypertext): Tập đối tượng (văn bản, hình ảnh, âm thanh, chương trình, …) có thể được tạo liên kết logic (còn gọi là siêu liên kết (hyperlink)) đến nhau.

Đọc thêm

RFC 9110: HTTP Semantics

RFC 9112: HTTP/1.1

RFC 9114: HTTP/3

Tiếp theo

Quản trị ứng dụng web