12/27 HTTP

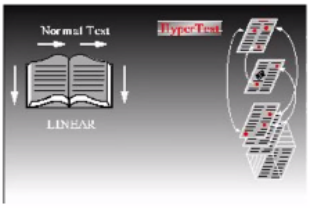
제목

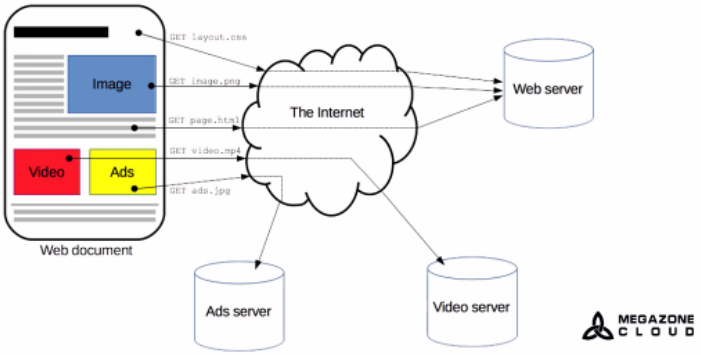
HTTP란

HTTP(HyperText Transfer Protocol)

하이퍼텍스트는 컴퓨팅과 관련된 개념으로, 텍스트 조작을 서로 연결할 수 있는 시스템을 말하며, 이를 통해 사용자는 순차적인 아닌 관련 항목을 통해 정보에 액세스 할 수 있습니다.

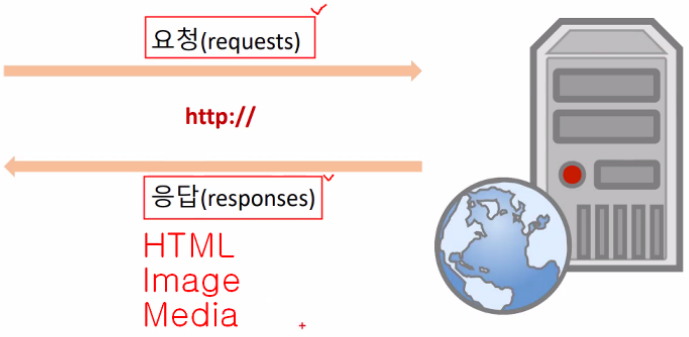
HTML 문서와 같은 리소스들을 가져올 수 있도록 해 주는 프로토콜입니다. HTTP는 웹에서 이루어 지는 모든 데이터 교환의 기초이며, 클라이언트-서버 프로토콜이기도 합니다. 하나의 완전한 문서는 텍스트, 레이아웃 설명, 이미지, 비디오, 스크립트 등 불러온(fetched) 하위 문서들로 재구성 됩니다.





HTTP란

클라이언트와 서버들은 (데이터 스트림과 대조적으로) 개별적인 메시지 교환에 의해 통신합니다. 보통 브라우저인 클라이언트에 의해 전송되는 메시지를 요청(requests)이라고 부르며, 그에 대해 서버에서 응답으로 전송되는 메시지를 응답(responses)이라고 부릅니다.



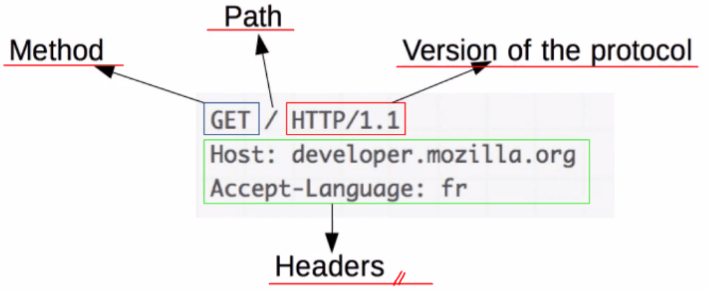
MEGAZONE CLOUD

HTTP 요청

- Method(메서드) – 보통 클라이언트가 수행하고자 하는 동작을 정의한 GET, POST 같은 동사나 OPTIONS 나 HEAD와 같은 명사입니다.
- Path – 가져오려는 리소스의 경로; 예를 들면 프로토콜(http://), 도메인, 또는 TCP 포트인 요소들을 제거한 리소스의 URL 입니다.
- Version of the protocol – HTTP 프로토콜의 버전.
- Headers – 서버에 대한 추가 정보를 전달하는 선태적 헤더들

조회 추가 / 수정 / 삭제

ls /home/ubuntu/.ssh

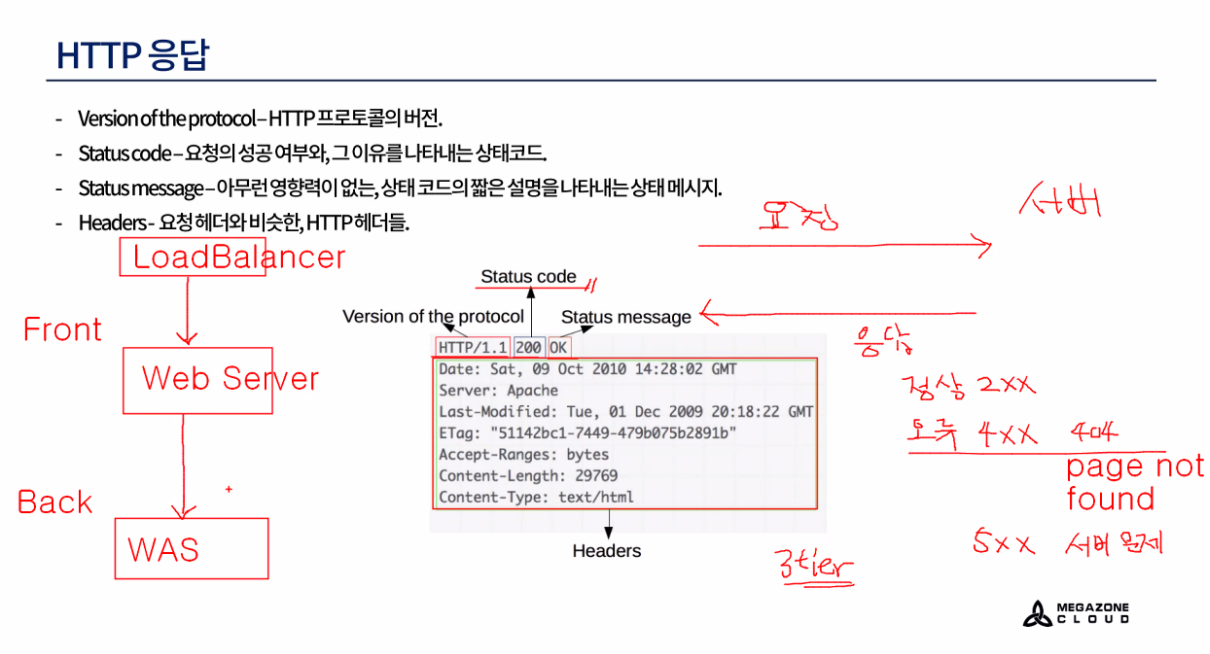


MEGAZONE CLOUD

HTTP 메서드만 봐도 사용자가 어떤 요청을 했는지 구분할 수 있다

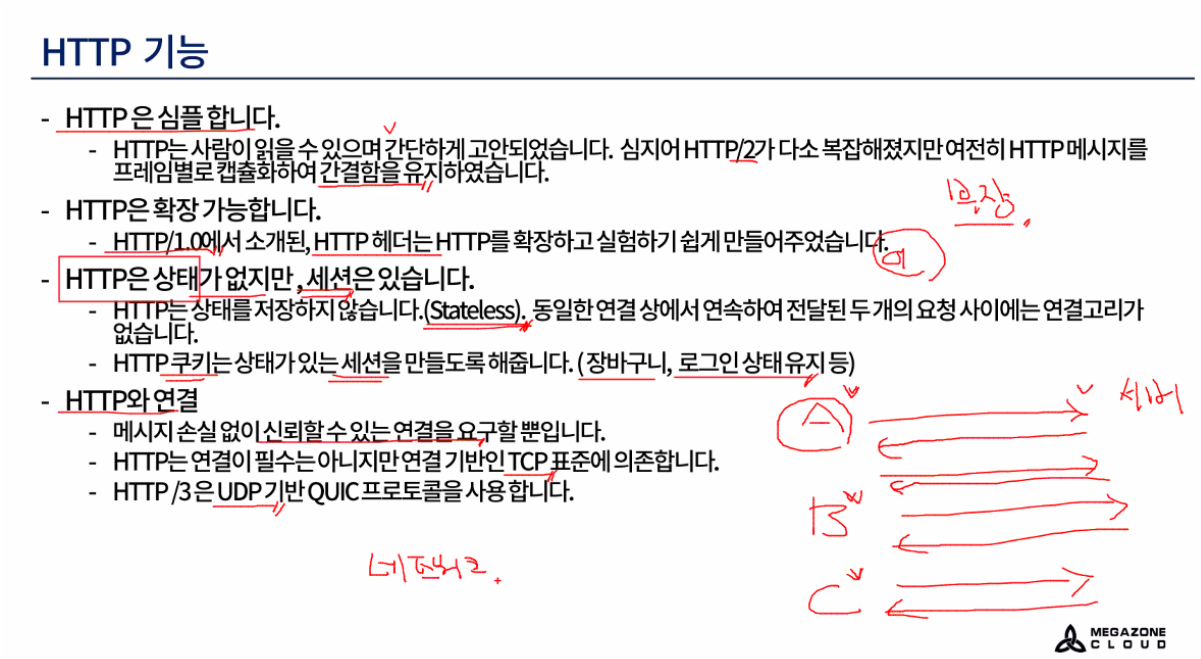
조회하고자 하는 카탈로그 식별 가능 → 추가 path : 서버의 파일이 있는 위치 알려주고자 하는 거 그때 사용하는 것

headers : 상세한 메타 정보, 요청 도메인 주소, 요청자 사용 언어



응답에 대한 상태 - status : 사용자가 서버에 요청을 했을 때, 서버가 정상적으로 응답을 할 수 있는 상황이냐 or 오류가 있냐 를 구분할 수 있다.
200 번대 : 정상
400번대 : 오류 ex) 404 page not found
500번대 : 서버에 대한 문제가 있을 때

응답 헤더 - header 요청 헤더와 비슷한 http



상태정보가 없기 때문에

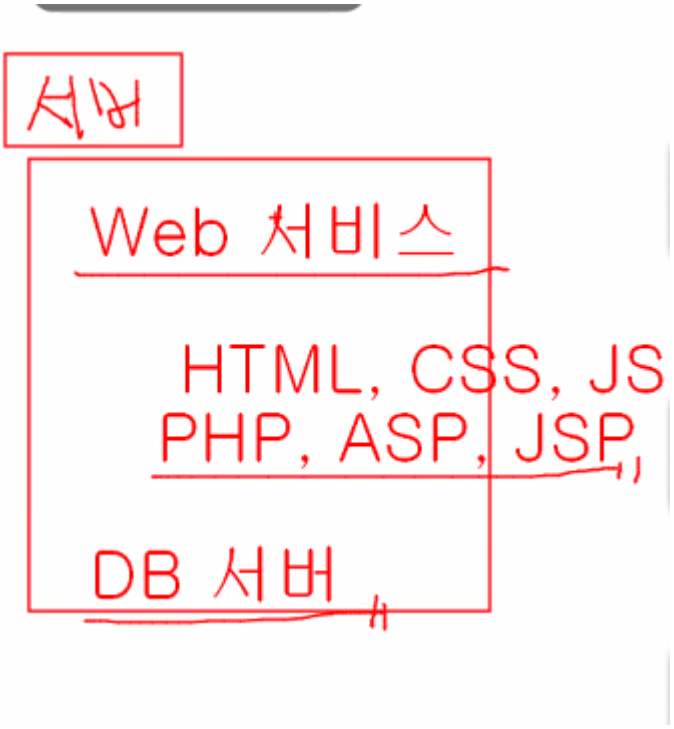
예를 들어 회사에서 부장님이 어제 했었던 작업 이짜나!!! 라고 했을 때 이전 상태 기억하고 있지 않음..어제 머해찌..

그러니까 전부 다 명시해줘야해

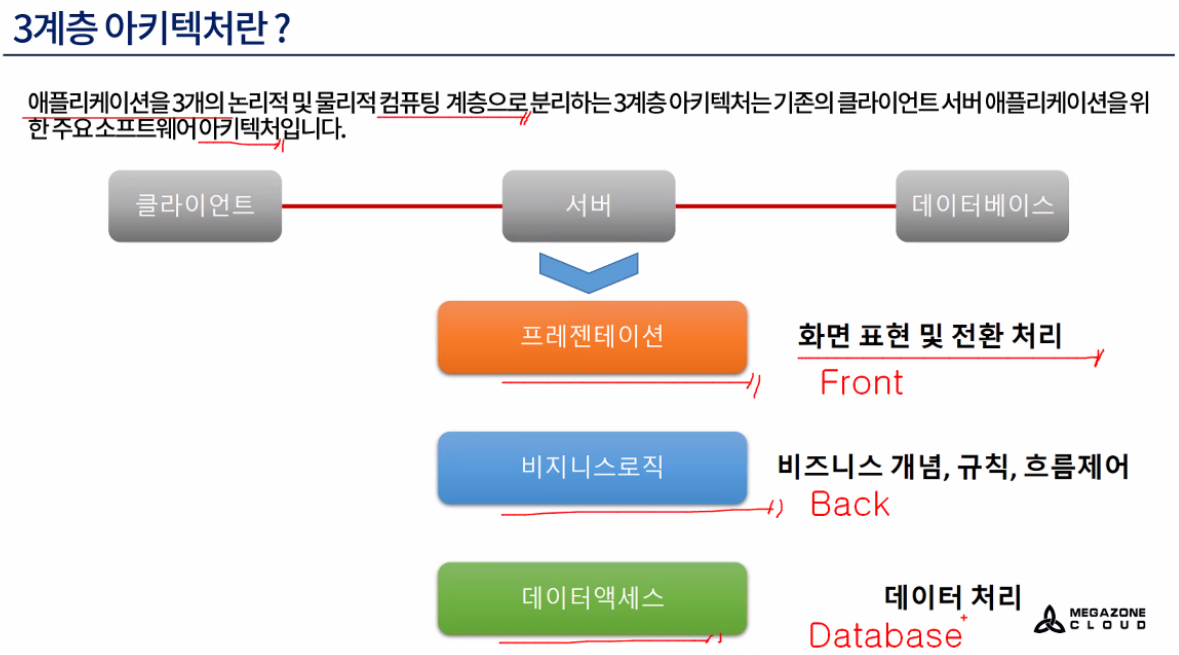
상태가 없다는 건 너무 힘든 상황.. 상태가 있는 것처럼 만드는 거 session

3tier

옛날에

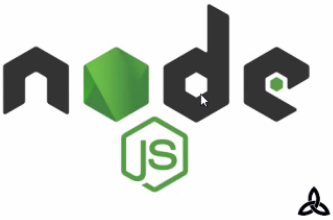

서버
Web 서비스
HTML, CSS, JS
PHP, ASP, JSP,
DB 서버

요즘


3계층 아키텍처란 ?
애플리케이션을 3개의 논리적 및 물리적 컴퓨팅 계층으로 분리하는 3계층 아키텍처는 기존의 클라이언트 서버 애플리케이션을 위한 주요 소프트웨어 아키텍처입니다.
클라이언트
서버
데이터베이스
프레젠테이션
화면 표현 및 전환 처리
Front
비지니스로직
비즈니스 개념, 규칙, 흐름제어
Back
데이터액세스
데이터 처리
Database
MEGAZONE CLOUD

3계층 아키텍처란 - 애플리케이션 계층

- 논리 계층 또는 중간 계층이라고도 하는 애플리케이션 계층은 애플리케이션의 핵심 입니다. 이 계층에서는 프리젠테이션 계층에서 수집된 정보가 처리 됩니다.
- 때때로 이는 데이터 계층의 다른 정보에 대해 처리되며, 비즈니스 규칙의 특정 세트인 비즈니스 로직을 사용합니다.
- 또한 애프리케이션 계층은 데이터 계층의 데이터를 추가, 삭제 또는 수정할 수도 있습니다.
- 애프리케이션 계층은 일반적으로 Python, Java, Perl, PHP 또는 Ruby를 사용하여 개발되며, API 호출을 사용하여 데이터 계층과 통신 합니다.



MEGAZONE CLOUD

3계층 아키텍처란 - 프리젠테이션 계층

- 프리젠테이션 계층은 일반 사용자가 애플리케이션과 상호작용하는 애플리케이션의 사용자 인터페이스 및 통신 계층입니다.
- 주요 목적은 정보를 표시하고 사용자로부터 정보를 수집하는 것입니다. 이 최상회 레벨 계층은 예를 들어 웹브라우저, 데스크탑 애플리케이션 또는 그래픽 사용자 인터페이스(GUI)에서 실행될 수 있습니다.
- 웹 프리젠테이션 계층은 일반적으로 HTML, CSS 및 JavaScript 를 사용하여 개발됩니다.

MEGAZONE CLOUD

3계층 아키텍처란 - 데이터 계층

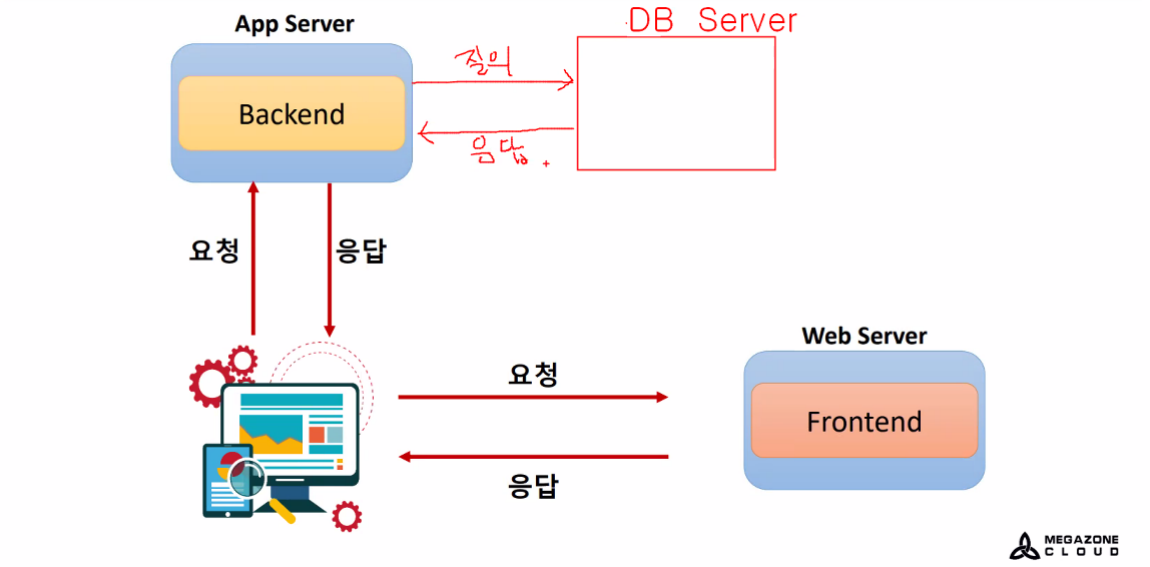
- 종종 데이터베이스 계층, 데이터 액세스 계층 또는 백엔드라고도 불리는 데이터 계층은 애플리케이션이 처리하는 정보가 저장 및 관리되는 곳입니다. 이는 관계형 데이터베이스 관리 시스템(예: PostgreSQL, MySQL, MariaDB, Oracle, DB2, Informix, MS-SQL)또는 no-SQL 데이터베이스 서버(예: Cassansdra, CouchDB, MongoDB)일 수 있습니다.
- 3계층 애플리케이션에서는 모든 통신이 애플리케이션 계층을 통과합니다. 프리젠테이션 계층과 데이터 계층은 서로 간에 직접 통신할 수 없습니다.

3tier 장점

3계층 아키텍처란 - 장점

- 3계층 아키텍처의 주요 장점은 기능의 논리적 및 물리적 분리입니다. 각 계층은 별도의 운영체제 및 서버 플랫폼에서 실행될 수 있습니다. 이는 기능적 요구사항에 가장 적합합니다.
- 각 계층이 하나 이상의 전용 서버 하드웨어 또는 가상 서버에서 실행되므로, 다른 티어에 영향을 주지 않고도 각 계층의 서비스를 사용자 정의하고 최적화 할 수 있습니다.
  - 보다 신속한 개발 : 각 계층이 서로 다른 팀에서 동시에 개발될 수 있으므로, 기업은 애플리케이션을 보다 빠르게 시장에 출시할 수 있으며 프로그래머는 각 계층에 대해 최신 및 최상의 언어와 툴을 사용할 수 있습니다.
  - 확장성 개선 : 필요에 따라 임의의 계층을 다른 계층과 독립적으로 확장 할 수 있습니다.
  - 안정성 개선 : 한 계층의 가동 중단은 다른 계층의 가용성 또는 성능에 별로 영향을 미치지 않습니다.
  - 보안성 강화 : 프리젠테이션 계층과 데이터 계층이 직접 통신할 수 없으므로, 잘 설계된 애플리케이션 계층은 내부 방호벽의 일종으로 작동하여 SQL 인젝션 및 기타 악의적 해위를 방지 할 수 있습니다.

웹 사이트 호출 방식



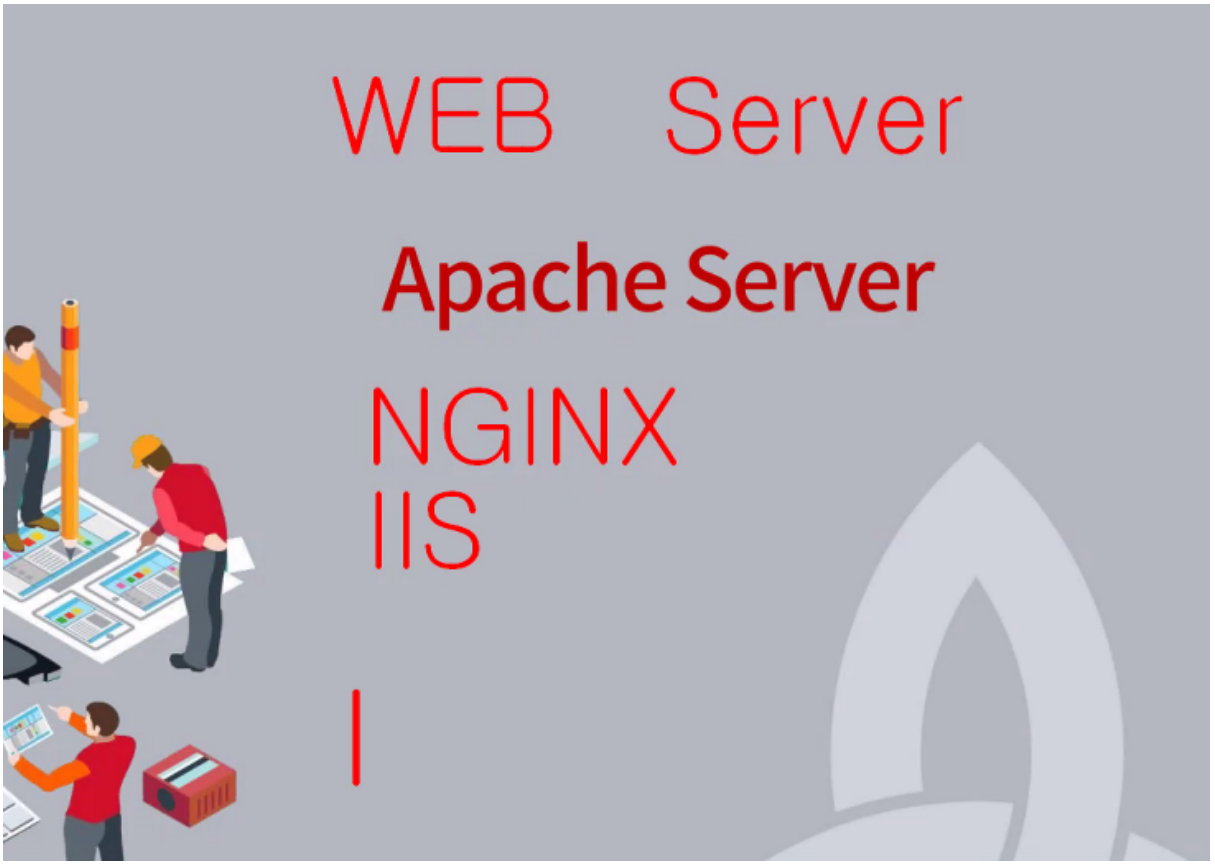
CSR : Client Side Rendering

SSR : Server Side Rendering

SPA : Single Page Appliction

MPA : Multi Page Application

apache server - web server



Web Server란

웹 서버(Web Server)는 다음의 두 가지 의미.

- 소프트웨어 (Software): 웹 브라우저와 같은 클라이언트로부터 HTTP 요청을 받아들이고, HTML 문서와 같은 웹 페이지를 반환하는 컴퓨터 프로그램.
- 하드웨어(Hardware): 위에 언급한 기능을 제공하는 컴퓨터 프로그래을 실행하는 컴퓨터.
- https://webtechstats.thomasorlita.com/category/web-servers/

Web Server와 WAS 역할 차이

- 웹 서버(Web Server)
  "웹 브라우저 클라이너트로부터 HTTP 요청을 받아들이고 HTML 문서와 같은 웹 페이지를 반환하는 컴퓨터 프로그램"
  - 정적 컨텐츠(HTML, 이미지, 파일등..)을 제공하는 서버
  - 동적 컨텐츠를 요청 받으면 WAS에게 해달 요청을 넘겨주게 된다.
- WAS(Web Application Server)
  "인터넷 상에서 HTTP프로토콜을 통해 사용자 컴퓨터나 장치에 애플리케이션을 수행해주는 미들웨어, 주로 동적 서버 컨텐츠를 수행하는 것으로 일반적인 웹 서버와 구별이 되며, 주로 데이터베이스 서버와 같이 수행된다."
  - WAS는 웹 서버로는 처리 할 수 없는 데이터베이스 조회나 다양한 로직 처리가 필요한 동적 컨텐츠를 제공한다.
  - WAS의 존재로 유저의 다양한 요구에 맞춰서 웹 서비스를 제공할 수 있다.

웹 서버를 반드시 구축해야하는 것은 아니다. 하지만 WAS는 동적 컨텐츠 처리를 위해 존재하므로 정적 커텐츠까지 처리해야할 경우 속도가 느려질 수 밖에 없다. 그렇기 때문에 웹 서버를 두고 플러그인 형태로 WAS 를 두어 데이터를 효율적으로 처리 할 수 있도록 구성하는 것이다.

HTML, CSS, JS, Image, Media

Tomcat, WSGI

Servlet JSP Spring Python(Django Flask) ...

페이지 생성

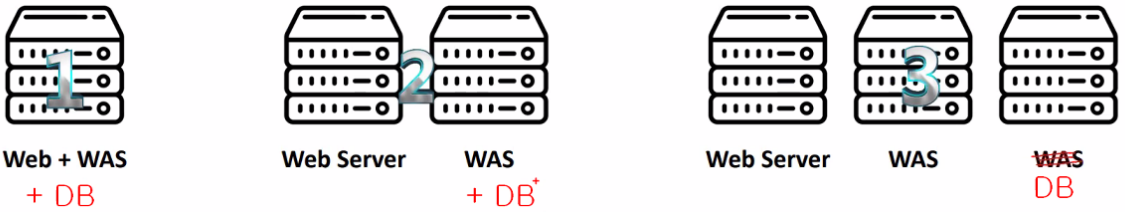
데이터 생성

MEGAZONE CLOUD

web → 정적

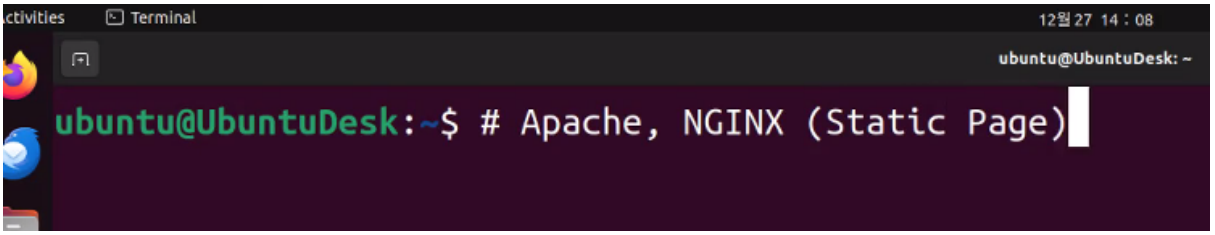
was → 동적, tomcat

서버 구성 방식



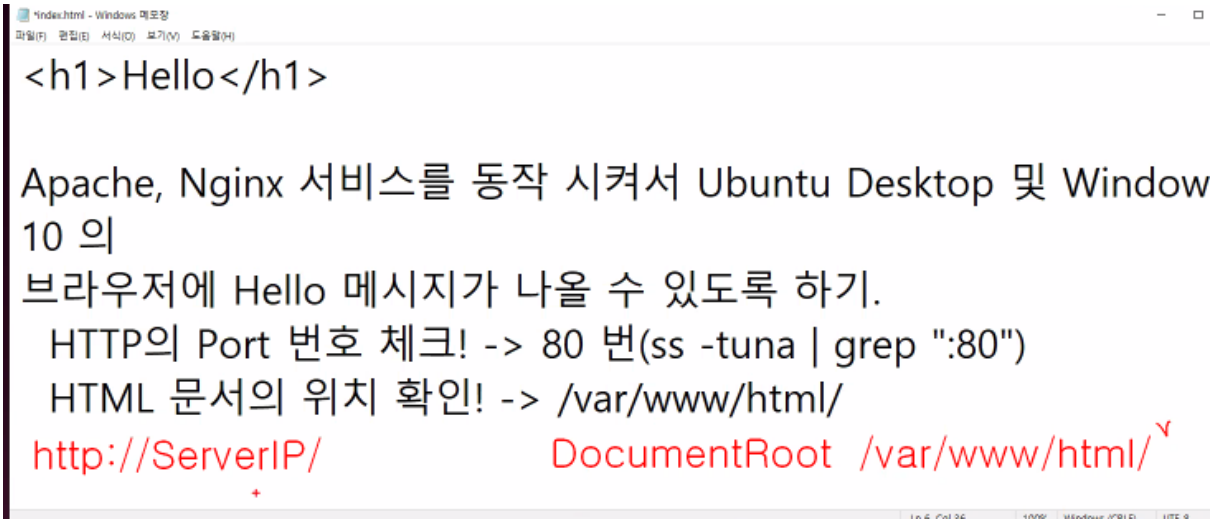
MEGAZONE CLOUD

web server 설치



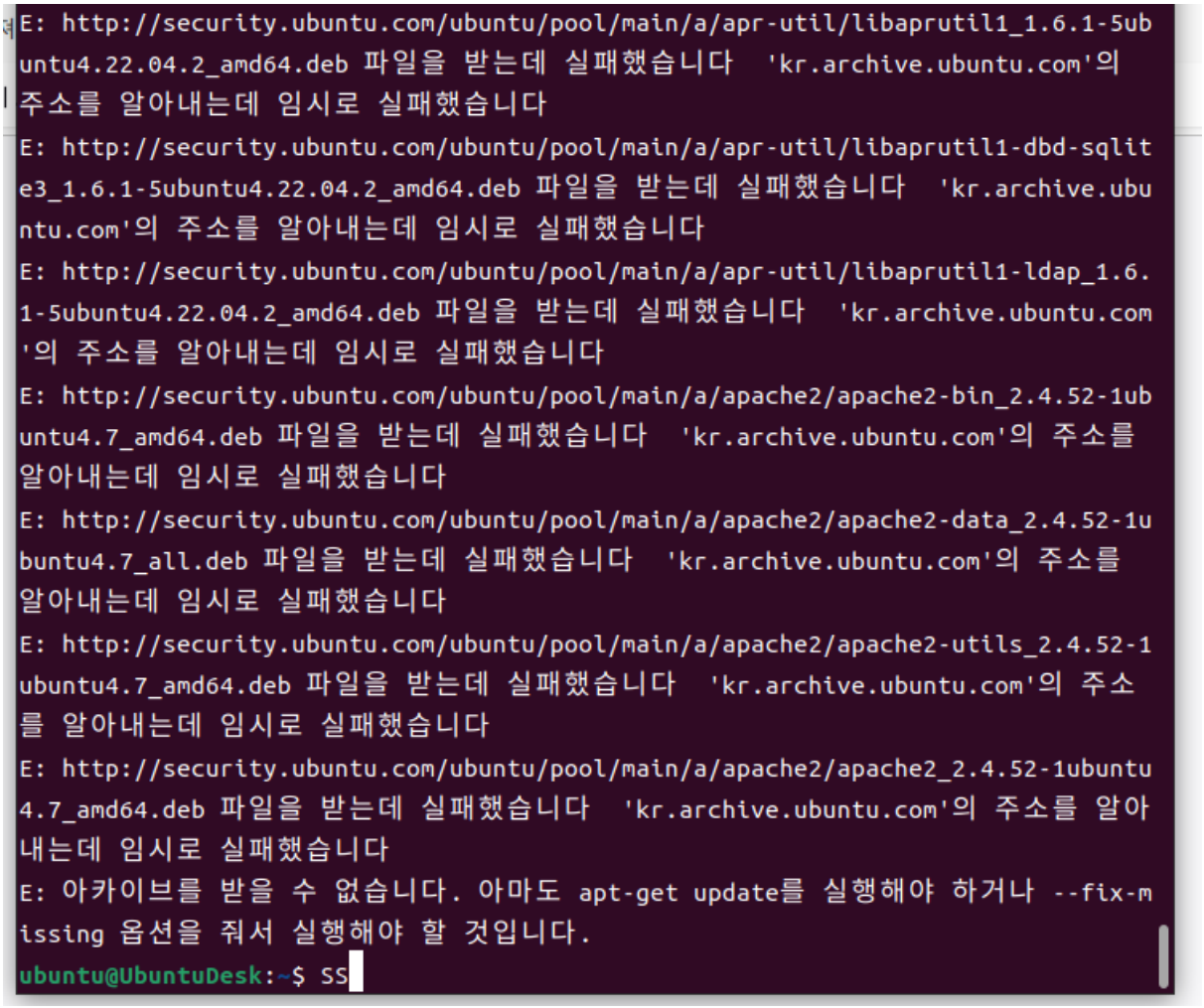
간단히 아파치 톰캣 설치해보는 거

이런 식으로



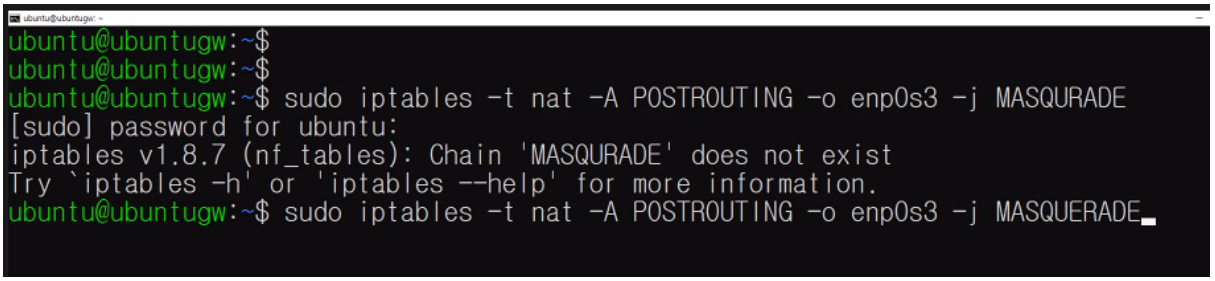
우분투에 아파치

```
sudo apt update
sudo apt install apache2
```



계속 실패

이유가 뭐냐?



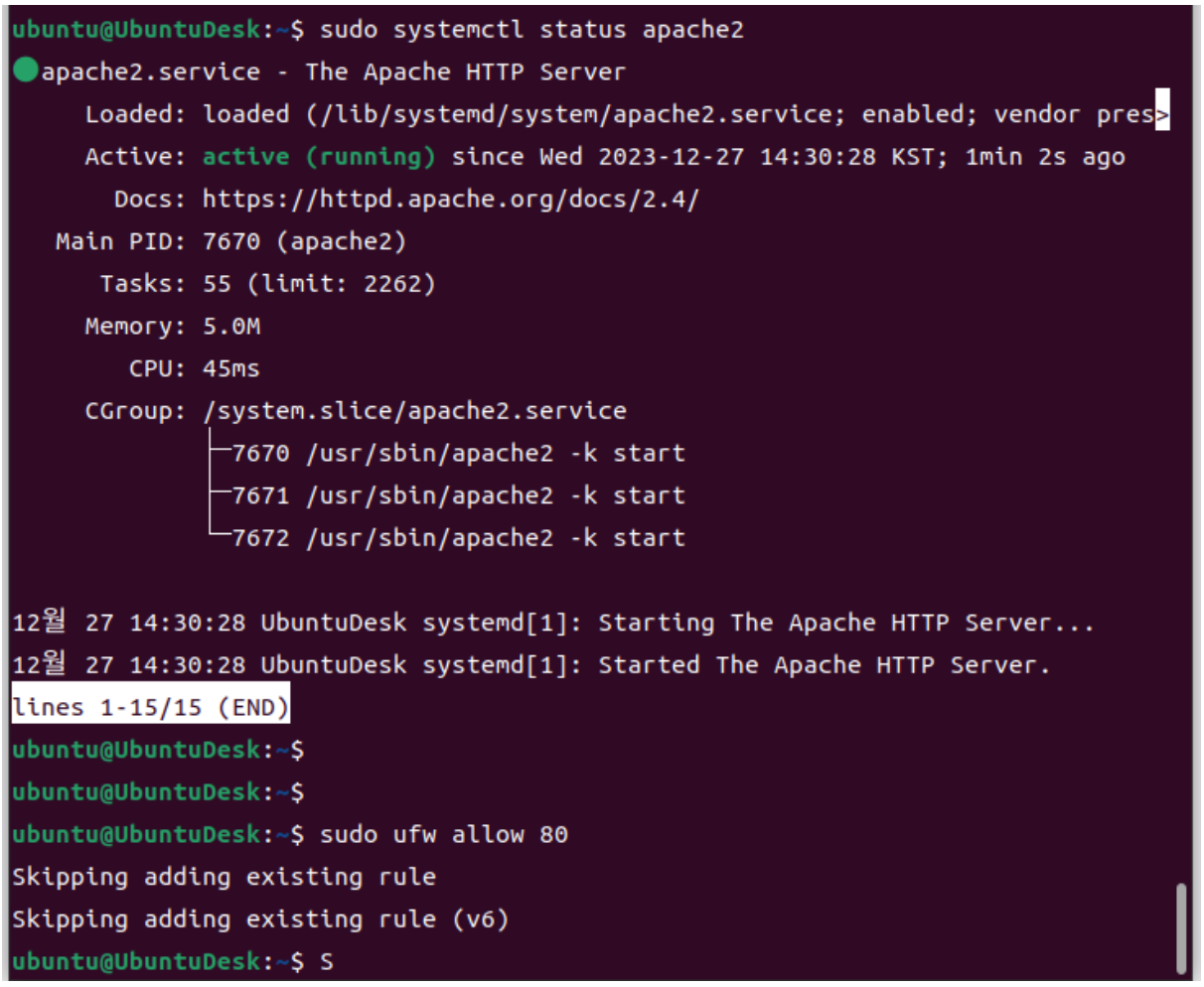
이 iptables 명령은 네트워크 주소 변환(NAT)을 수행하는 데 사용되는 명령입니다. NAT는 네트워크에서 IP 주소를 변환하여 패킷을 라우팅하는 기술 중 하나입니다.

명령은 네트워크 주소 변환을 통해 내부 네트워크에서 외부로 나가는 패킷의 출발지 주소를 외부 인터페이스의 주소로 변경하여 인터넷에 연결할 수 있도록 하는데 사용됩니다.

```
network:
  version: 2
  renderer: NetworkManager
  ethernets:
    enp0s8:
      dhcp4: no
      addresses: [192.168.56.101/24]
      gateway4: 192.168.56.103
      nameservers:
        addresses: [8.8.8.8, 8.8.4.4]
```

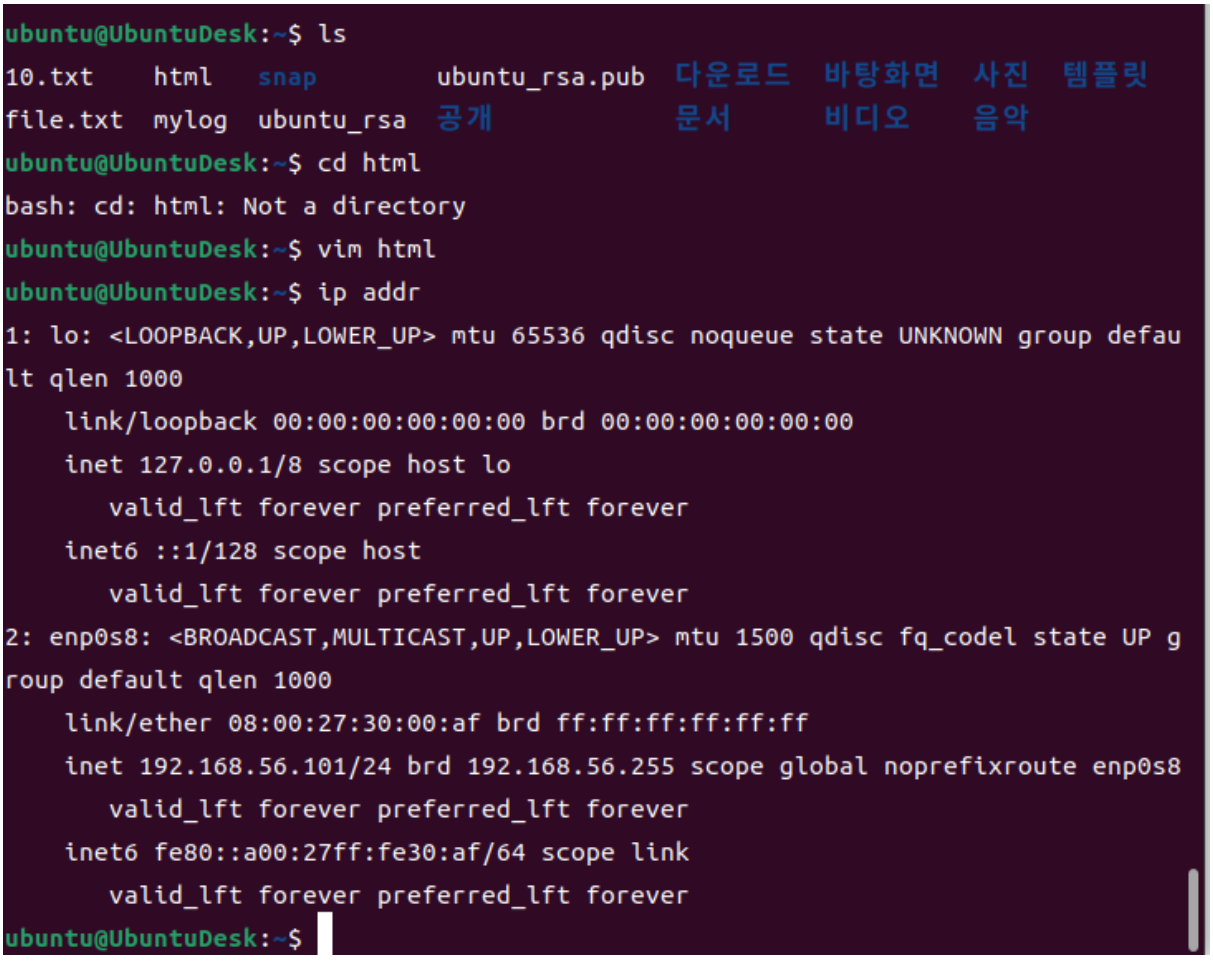
1. `network:`: 이는 YAML 파일의 최상위 키로, 네트워크 설정이 시작되는 곳을 나타냅니다.
2. `version: 2`: 이 설정 파일이 Netplan의 버전 2 형식을 따른다는 것을 나타냅니다.
3. `renderer: NetworkManager`: NetworkManager가 시스템의 모든 장치를 관리하도록 하는 옵션입니다. NetworkManager는 네트워크 연결을 관리하는 유틸리티입니다.
4. `ethernets:`: 이 키는 이 설정 파일에서 구성할 이더넷 인터페이스 목록을 나타냅니다.
5. `enp0s8:`: 이는 구성할 이더넷 인터페이스의 이름을 나타냅니다. 이 설정에서는 `enp0s8` 인터페이스를 구성하고 있습니다.
6. `dhcp4: no`: DHCP를 통해 IPv4 주소를 동적으로 얻지 않고 수동으로 설정할 것임을 나타냅니다.
7. `addresses: [192.168.56.101/24]`: 이 인터페이스에 할당할 정적 IPv4 주소와 서브넷 마스크를 지정합니다. 여기서는 `192.168.56.101` IP 주소를 사용하고 있습니다.
8. `gateway4: 192.168.56.103`: 이 인터페이스에서 사용할 IPv4 게이트웨이의 주소를 지정합니다. 여기서는 `192.168.56.103`을 사용하고 있습니다.
9. `nameservers:`: DNS 이름 해결을 위한 DNS 서버 주소를 설정합니다.
10. `addresses: [8.8.8.8, 8.8.4.4]`: Google Public DNS 서버의 주소인 `8.8.8.8` 및 `8.8.4.4`를 사용하여 DNS를 구성합니다.

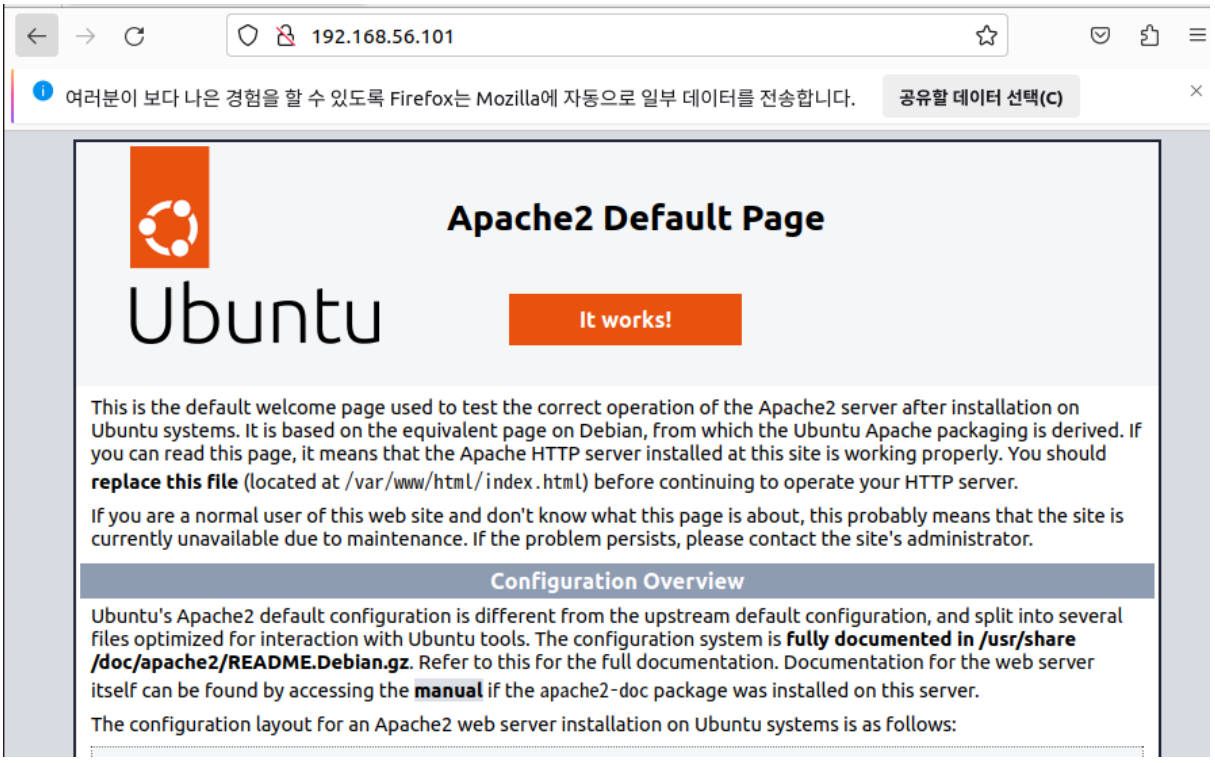
이러한 설정을 통해 `enp0s8` 인터페이스에 고정된 IPv4 주소를 할당하고, 수동으로 게이트웨이 및 DNS 서버를 구성하여 네트워크 연결을 설정하고 있습니다. 이 설정을 적용하려면 `sudo netplan apply` 명령을 사용할 수 있습니다.




```
sudo systemctl start apache2

sudo systemctl restart apache2

sudo systemctl enable apache2

sudo systemctl status apache2

sudo ufw allow 80
```

html 문서 만들기 - 홈 디렉토리 index.html

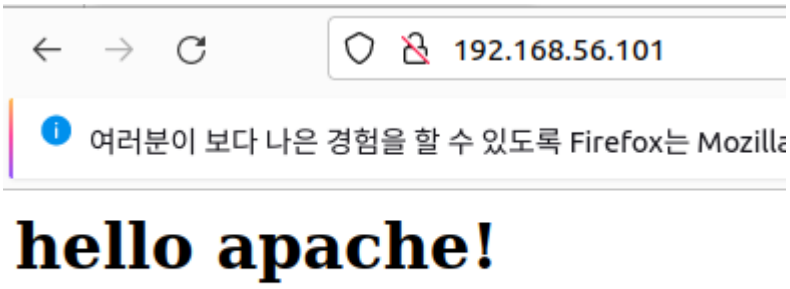

ubuntu@UbuntuDesk:~$ ls
10.txt html snap ubuntu_rsa.pub 다운로드 바탕화면 사진 템플릿
file.txt mylog ubuntu_rsa 공개 문서 비디오 음악
ubuntu@UbuntuDesk:~$ cd html
bash: cd: html: Not a directory
ubuntu@UbuntuDesk:~$ vim html
ubuntu@UbuntuDesk:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defau
lt qlen 1000
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
inet6 ::1/128 scope host
valid_lft forever preferred_lft forever
2: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP g
roup default qlen 1000
link/ether 08:00:27:30:00:af brd ff:ff:ff:ff:ff:ff
inet 192.168.56.101/24 brd 192.168.56.255 scope global noprefixroute enp0s8
valid_lft forever preferred_lft forever
inet6 fe80::a00:27ff:fe30:af/64 scope link
valid_lft forever preferred_lft forever
ubuntu@UbuntuDesk:~$



내가 실수한거

index.html 파일 만드는 위치를 잘못 잡았다.

나는 home 에다가 만들려고 했지만 192.168.56.101 에 접속하자마자 바로 보기 위해서는 /var/www/html 여기에 만들어야 했다.



엔진엑스

```
ubuntu@UbuntuDesk:/var/www/html$ ls
index.html  index.nginx-debian.html  nginx.html
ubuntu@UbuntuDesk:/var/www/html$
```

파일 만들고

엔진엑스는

```
sudo vim /etc/nginx/sites-available/default
```

여기서 수정해줘야 함

이런식으로 추가해줘야해

이 순서대로 찾는 거라서

```
    # Add index.php to the list if you are using PHP
    index nginx.html index.htm index.nginx-debian.html;

    server_name 192.168.56.101;

    location / {
            # First attempt to serve request as file, then
            # as directory, then fall back to displaying a 404.
            try_files $uri $uri/ =404;
    }
```


```
ubuntu@UbuntuDesk:~$ ss -tuna | grep ":80"
tcp    LISTEN 0      511          0.0.0.0:80        0.0.0.0:*
tcp    LISTEN 0      511             [::]:80           [::]:*
ubuntu@UbuntuDesk:~$
```