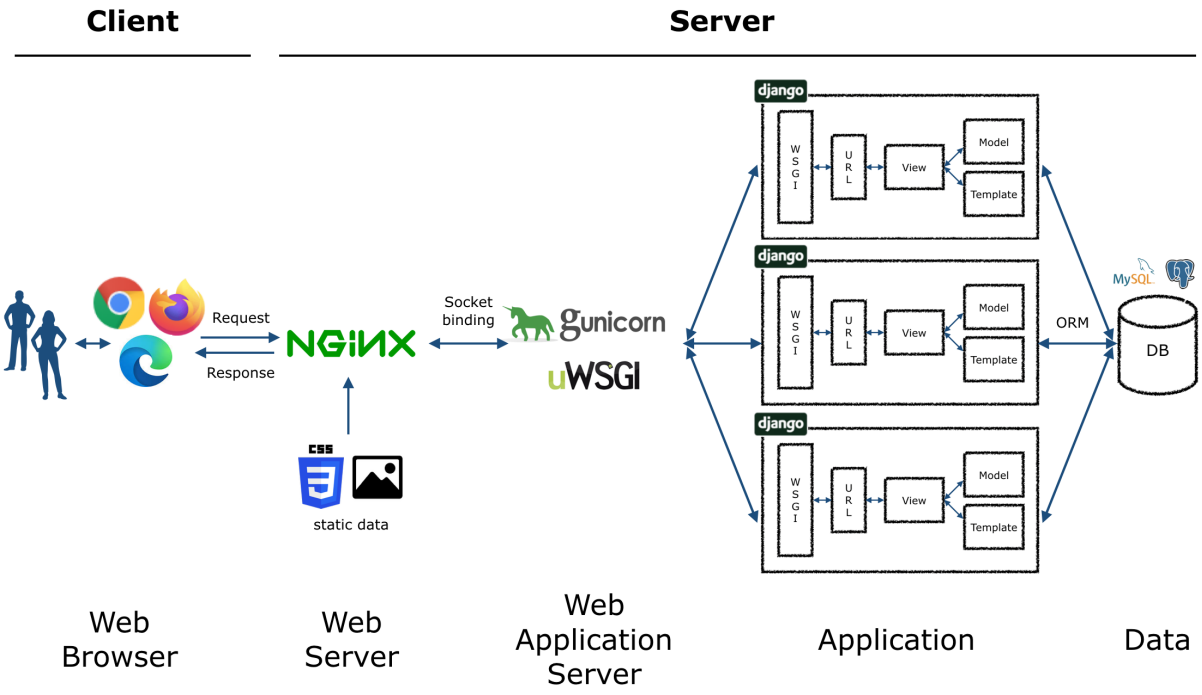
Web Server 와 WAS

웹 애플리케이션(Web Application)

1. 인터넷을 통해 접근할 수 있는 애플리케이션 소프트웨어
2. 웹 브라우저를 통해 사용자가 접근하고, 웹 서버와 상호작용하여 동적 또는 정적 콘텐츠를 제공

→ 우리가 만드는 서비스 (푸바오, NowDoBoss 등...)



장고 기준 흐름 설명

1. 클라이언트 요청: 사용자가 블라우저를 통해 웹 애플리케이션에 접근해 요청을 보낸다.
2. Nginx(웹서버) 에서 요청 처리: HTTP 요청을 받아들이고, 요청된 리소스가 정적 파일(HTML,CSS,JS) 인 경우 이를 직접 클라이언트에 반환한다.
3. 동적 요청 전달: 요청된 리소스가 동적 콘텐츠를 필요로 하는 경우, NGINX는 요청을 Gunicorn 또는 uWSGI에 전달한다.
4. WAS(Gunicorn, uWSGI) 에서 요청 처리 : Gunicorn 또는 uWSGI는 요청을 받아 Django 애플리케이션에 전달한다.

5. Django는 urlconf를 통해 요청을 적절한 View로 매핑하고, View는 필요한 데이터를 Model에서 조회하여 템플릿에 전달한다.
6. 데이터베이스 상호작용: Django의 Model은 데이터베이스(MySQL, PostgreSQL)와 상호작용하여 데이터를 조회하거나 저장합니다.
7. 응답 생성: Django의 Template은 View에서 전달받은 데이터를 HTML로 렌더링하여 동적 웹 페이지를 생성합니다.
8. 응답 반환: 생성된 동적 콘텐츠가 Gunicorn 또는 uWSGI를 통해 NGINX로 반환되고, NGINX는 이를 클라이언트(웹 브라우저)에게 응답으로 전달합니다.
9.

웹 서버(Web Server)

하드웨어:

1. WEB 서버가 설치되어 있는 컴퓨터

소프트웨어:

1. 정적 컨텐츠 제공: HTML, CSS , JavaScript, 이미지 파일과 같은 정적 컨텐츠를 클라이언트에게 제공
2. HTTP 요청 처리: HTTP 프로토콜을 사용하여 클라이언트의 요청을 받고 응답을 보낸다.
3. 로드 밸런싱: 여러 서버에 걸쳐 부하를 분산시키는 역할을 할 수 있다.

ex)

- Apache HTTP Server
- Nginx

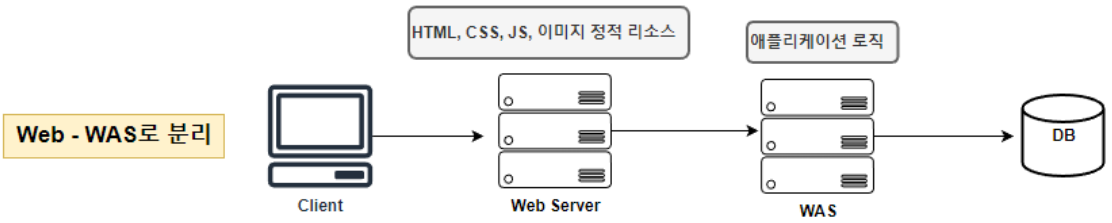
웹 애플리케이션 서버 (WAS)

역할:

1. 동적 컨텐츠 제공 : DB 조회, 비즈니스 로직 처리 등 동적 컨텐츠를 생성하여 웹 서버에 제공한다.
2. 트랜잭션 관리: 데이터베이스 트랜젝션을 관리하고 보장한다.
3. 비즈니스 로직 실행: 복잡한 비즈니스 로직을 처리한다.

웹 서버는 클라이언트의 요청을 받고, 정적 파일을 제공하거나 동적 처리가 필요할 때 웹 애플리케이션 서버로 요청을 전달한다.

웹 애플리케이션 서버는 동적 요청을 처리하고 생성된 콘텐츠를 웹 서비스를 통해 클라이언트에게 반환한다.



헷갈리는 점

- **프론트엔드**: 사용자가 직접 상호작용하는 부분으로, 웹 브라우저와 브라우저에서 해석되는 HTML, CSS, JavaScript 파일들로 구성된다.
- **클라이언트 영역**: 사용자가 웹 브라우저를 통해 요청을 보내고, 브라우저는 서버로부터 받은 정적 파일을 해석하여 화면에 표시한다.
- **웹 서버에서 제공하는 정적 파일**: NGINX와 같은 웹 서버가 제공하는 HTML, CSS, JS 파일이 프론트엔드의 일부이다.