프로젝트 배포

기술 스택

NGINX

팔방 미인인데 성능도 좋아!

High performance load balancer & web server & API gateway & reverse proxy

비동기 방식 → 높은 성능

정적인 파일을 서비스할 때 톰캣보다 성능 좋음

DDOS 공격 방어 가능

프론트와 백엔드의 분기

/ 요청은 frontend로

/api 요청은 backend로!

webserver로서의 역할

API gateway로서의 역할

CORS

Corss-Origin Resource Sharing

domain, port, protocol이 다를 때 발생

nginx의 설정을 기억하자

https://도메인-a.com의 frontend의 JS코드가 XMLHttpRequest를 이용해 https://도매인-b.com/datajson을 요청하는 경우

http → https or https→http

즉, 다른 주소로 보내는 경우!

배포구조

FE와 BE각각 도커위에 돌려~

FE를 돌릴 때 npm run serve는 하지말자!
build를 제대로 거치지 않고 돌아간다 → 성능 저하

왜 도커를 쓰는가?

빠르게 서버 증설 가능
(VM은 OS를 부팅하는 방식)VM 부팅 1분 → 서비스 전체가 중지됨

배포의 편의성
이미지를 만들어두면 찍어내기만 하면 됨 (w/ k8s)

어디까지 도커화해야할까?

FE/BE는 필수적

배포의 효율성/편의성을 생각해보자

DB/Jenkins/Nginx는 선택적

DB를 이미지화해서 새로 배포할 일이 많이 있을까?NO! → 옮긴다면 데이터는??

build 서버를 병렬적으로 추가 증설하는 경우는? 많지 않다!

임의의 포트를 쓰면 안되는 경우

ISP(SKT,KT,LGU)에 따라 닫혀 있는 포트가 존재

어느 곳에서는 되고, 어느 곳에서는 안되는 서비스라면, 고객은 포트가 막혔다고 생각하지 않고 이탈함

[배포] Gitlab → Jenkins

개발자가 gitlab의 특정 브랜치에 머지하면 이벤트가 트리거 되어 Jenkins에서 빌드 시작

빌드가 완료되면 도커 이미지가 제작되어 배포된다.

[보안]

SSL(→TLS)

암호화 필요!

매번 암호화하긴 힘드니까 TLS(Transport Layor Security)사용

WebRTC를 위해서는 SSL 인증서 설치 필요

Cert Bot

https 확산을 위해 시작된 비영리 프로젝트

상용 program 제작할 때는 신뢰할 수 있는 발급자로부터 SSL 인증서를 구매 사용

그 외의 경우 Cert Bot을 이용하여 무료 인증서 발급

cert Bot은 nginx에 자동으로 설정 추가

사용자 계정 만들기

각 프로그램을 실행할 때는 프로그램에 맞는 권한을 가진 사용자 계정을 만들어서 실행한다

ubuntu계정이나 심지어 root 계정으로 실행하는 경우 매우 위험!

사용자 계정으로 실행하자!