

A708

포팅메뉴얼

EV-meeting | SSAFY 7기 | 공통 프로젝트

# 목차

1. 프로젝트 기술 스택
2. 백엔드 빌드 방법
3. 프론트엔드 빌드 방법
4. 배포 명령어 정리
5. MySQL 사용방법
6. Nginx default
7. Ec2 세팅
8. AWS S3세팅

# 프로젝트 기술 스택

툴, 버전 명시

Vue : 2.6.14 버전

Spring boot

MySQL : 5.7.x 버전

Openvidu : LTS 버전

Node.js : LTS 버전 설치 (16.16.x 버전)

# 백엔드 빌드 방법

cd 프로젝트 디렉토리로 이동

gradlew build

/build/libs/프로젝트이름.jar 파일 생성

# 프론트엔드 빌드 방법

cd 프로젝트 디렉토리로 이동

npm run bulid

/dist 디렉토리 생성

# 배포 명령어 정리

**FE 빌드파일 수정 (dist 폴더)**

Sudo systemctl restart nginx

Or

Sudo service nginx restart

Nignx 재시작

**BE 빌드파일 수정 (.jar 파일)**

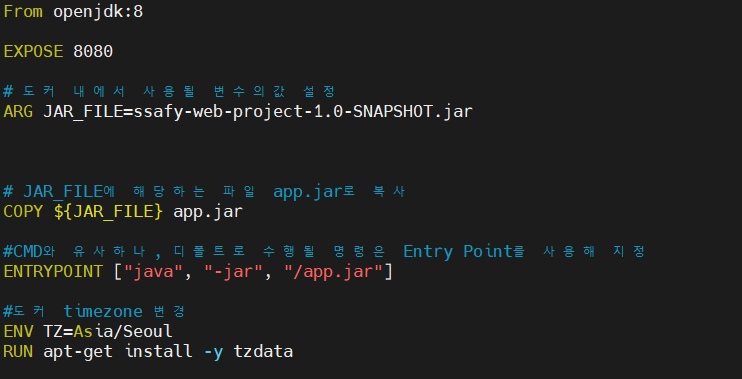
* **소스코드 확인**

서버에서 동작시킬 빌드파일(.jar)을 mobaXterm을 활용하여 서버로 이동합니다.

* **Dockerfile 작성 및 실행**

COMMAND : vim Dockerfile

를 사용하여 서버로 이동한 소스코드와 같은 경로에 도커파일을 작성합니다.

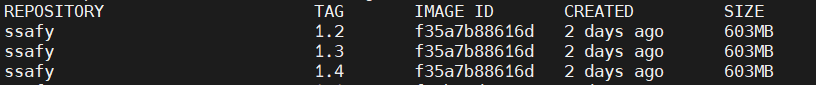


* **Docker image 생성**

COMMAND : sudo docker build –t ssafy:1.4 .

명령어를 통해 docker image 생성합니다.

`docker images` 명령어를 통해 생성된 도커 이미지를 확인합니다.



* **Docker container 실행**

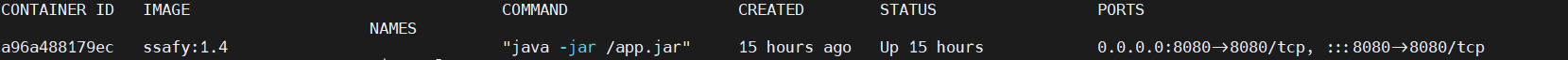
COMMAND : sudo docker run –d –p 8080:8080 ssafy

생성된 도커 이미지를 –d 옵션을 통해 백그라운드에서 실행시키고, -p 옵션을 사용하여 포트를 지정해줍니다.

* **실행중인 컨테이너 확인**

COMMAND: docker ps

실행한 컨테이너가 정상적으로 작동되는지 확인합니다.



Certbot 인증서 발급

Sudo nginx –t : nginx 설정 검사

Sudo certbot certonly –webroot –w /home/ubuntu/dist –d 도메인주소 (i7a708.p.ssafy.io)

발급 완료시 /etc/letsencrypt/live 디렉토리에 인증서 발급

Default 파일 수정 -> HTTPS 연결을 위한 443포트 연결 + 인증서 연결

Openvidu 설정

Sudo su : 관리자 권한

Root로 이동

Cd /opt : openvidu 설치되는 경로

curl https://s3-eu-west-1.amazonaws.com/aws.openvidu.io/install\_openvidu\_latest.sh | bash

openvidu 설치

exit

cd openvidu : 설치된 디렉토리로 이동

sudo vi .env : 설정 파일 수정

도메인, 비밀번호, 인증서 발급방법, 이메일, 포트번호 설정

Sudo rm –rf certificates : 기존 인증파일 삭제

docker run -p 4443:4443 *--rm -e OPENVIDU\_SECRET=MY\_SECRET openvidu/openvidu-server-kms:2.19.0*

openvidu 컨테이너 실행

./openvidu start : 서버에 openvidu 실행

./openvidu restart : 설정값 변경 후 openvidu 재시작

./openvidu stop : openvidu 종료

# MySQL 사용방법

Workbench 실행

Connections 추가

Hostname : ec2 주소 입력 (i7a708.p.ssafy.io)

Username : mysql username 입력 (i7a708)

Password : mysql pw 입력

# Nginx default

/etc/nginx/sites-available/default 파일

server {

# listen 80 default\_server;

# listen [::]:80 default\_server;

listen 80;

listen [::]:80;

root /home/ubuntu/dist;

index index.html;

server\_name i7a708.p.ssafy.io;

#server\_name \_;

**return** 301 https:*//i7a708.p.ssafy.io$request\_uri;*

}

server {

listen 443 ssl;

listen [::]:443;

server\_name i7a708.p.ssafy.io;

ssl\_certificate /etc/letsencrypt/live/i7a708.p.ssafy.io/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/i7a708.p.ssafy.io/privkey.pem;

location / {

root /home/ubuntu/dist;

# root /var/www/html;

# index index.html index. nginx-debian.html;

index index.html;

try\_files $uri $uri/ /index.html = 404;

}

# location /cam {

# allow all;

# deny all;

# proxy\_pass http:*//i7a708.p.ssafy.io;*

# }

location /demo {

proxy\_pass http:*//localhost:8081;*

proxy\_redirect off;

charset utf-8;

proxy\_set\_header X-Readl-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For @proxy\_add\_x\_forwarded\_for;

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-NginX-Proxy true;

}

}

# EC2 세팅

Application.prpoerties

#it will be set build date by gradle. if this value is @build.date@, front-end is development mode  
build.date=@build.date@  
server.port=8080  
server.address=0.0.0.0  
server.servlet.contextPath=/  
# Charset of HTTP requests and responses. Added to the "Content-Type" header if not set explicitly.  
server.servlet.encoding.charset=UTF-8  
# Enable http encoding support.  
server.servlet.encoding.enabled=true  
# Force the encoding to the configured charset on HTTP requests and responses.  
server.servlet.encoding.force=true  
  
# for SPA  
spring.resources.static-locations=classpath:/dist/  
spa.default-file=/dist/index.html  
spring.mvc.throw-exception-if-no-handler-found=true  
spring.resources.add-mappings=false  
  
# Swagger  
springfox.documentation.swagger.use-model-v3=false  
  
#database  
spring.jpa.hibernate.naming.implicit-strategy=org.springframework.boot.orm.jpa.hibernate.SpringImplicitNamingStrategy  
spring.jpa.hibernate.naming.physical-strategy=org.springframework.boot.orm.jpa.hibernate.SpringPhysicalNamingStrategy  
spring.jpa.hibernate.ddl-auto=update  
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL57Dialect  
spring.data.web.pageable.one-indexed-parameters=true  
  
#spring.datasource.url=jdbc:mysql://localhost:3306/ssafy\_web\_db?useUnicode=true&characterEncoding=utf8&serverTimezone=Asia/Seoul&zeroDateTimeBehavior=convertToNull&rewriteBatchedStatements=true  
#spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver  
#spring.datasource.hikari.username=root  
#spring.datasource.hikari.password=1234  
  
spring.datasource.url=jdbc:mysql://${EC2.DOMAIN}:3306/ssafy\_web\_db?useUnicode=true&characterEncoding=utf8&serverTimezone=Asia/Seoul&zeroDateTimeBehavior=convertToNull&rewriteBatchedStatements=true  
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver  
spring.datasource.hikari.username=${MYSQL.USERNAME}  
spring.datasource.hikari.password=${MYSQL.PASSWORD}  
  
# jwt  
jwt.secret=dyAeHubOOc8KaOfYB6XEQoEj1QzRlVgtjNL8PYs1A1tymZvvqkcEU7L1imkKHeDa  
# unit is ms. 15 \* 24 \* 60 \* 60 \* 1000 = 15days  
jwt.expiration=1296000000  
  
#logging  
#logging.file.name=./ssafy-web.log  
#logging.level.root=INFO  
#logging.level.com.samsung.security=DEBUG  
#logging.level.org.springframework.web=DEBUG  
#logging.level.org.apache.tiles=INFO  
#logging.level.org.sringframework.boot=DEBUG  
#logging.level.org.sringframework.security=DEBUG  
  
spring.devtools.livereload.enabled=true  
  
#gzip compression  
#server.compression.enabled=true  
#server.compression.mime-types=application/json,application/xml,text/html,text/xml,text/plain,application/javascript,text/css  
  
#for health check  
management.servlet.context-path=/manage  
management.health.db.enabled=true  
management.health.default.enabled=true  
management.health.diskspace.enabled=true  
  
# S3 Bucket  
cloud.aws.credentials.accessKey=${S3.ACCESSKEY}  
cloud.aws.credentials.secretKey=${S3.SECRETKEY}  
cloud.aws.stack.auto=false  
  
# AWS S3 Service bucket  
cloud.aws.s3.bucket=${S3.BUCKETNAME}  
cloud.aws.region.static=${S3.REGION}  
  
# AWS S3 Bucket URL  
cloud.aws.s3.bucket.url=${S3.URL}  
  
# multipart  
spring.http.multipart.max-file-size=20MB  
spring.http.multipart.max-request-size=20MB  
  
spring.servlet.multipart.max-file-size=20MB  
spring.servlet.multipart.max-request-size=20MB  
  
# SSL  
server.ssl.key-store = classpath:keystore.p12  
server.ssl.key-store-type = PKCS12  
server.ssl.key-store-password = a708  
  
# google email  
spring.mail.host=smtp.gmail.com  
spring.mail.port=465  
spring.mail.username=${EMAIL.ID}@gmail.com  
spring.mail.password=${EMAIL.PW}  
spring.mail.properties.mail.smtp.auth=true  
spring.mail.properties.mail.smtp.ssl.enable=true  
spring.mail.transport.protocol=smtp  
spring.mail.debug=true  
spring.mail.default.encoding=UTF-8

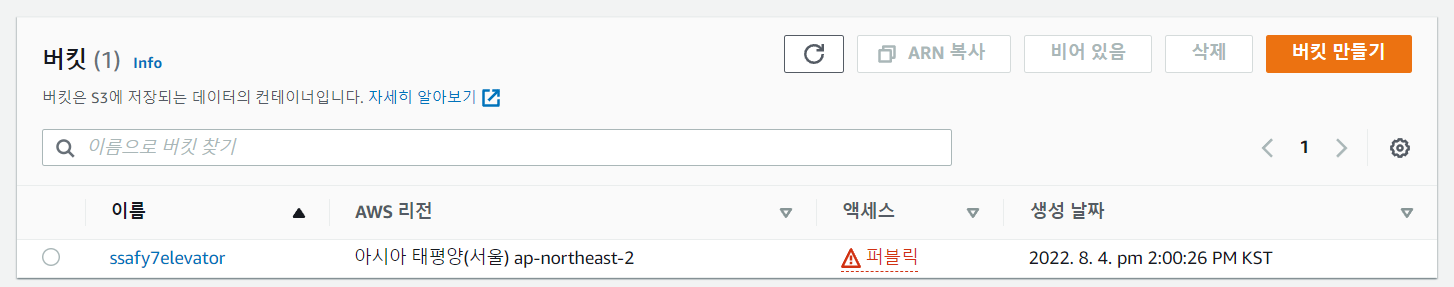
env.properties

# AWS EC2  
EC2.DOMAIN = i7a708.p.ssafy.io  
#EC2.DOMAIN = localhost  
  
# MYSQL  
MYSQL.USERNAME = i7a708  
MYSQL.PASSWORD = a708evmeeting!  
  
# AWS S3  
S3.URL = https://ssafy7elevator.s3.ap-northeast-2.amazonaws.com/  
S3.BUCKETNAME = ssafy7elevator  
S3.REGION = ap-northeast-2  
S3.ACCESSKEY = AKIAWMUOWC4X6NPA7PTS  
S3.SECRETKEY = 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  
  
# GOOGLE EMAIL  
EMAIL.ID = i7a708  
EMAIL.PW = mpxahhtrkbncjico

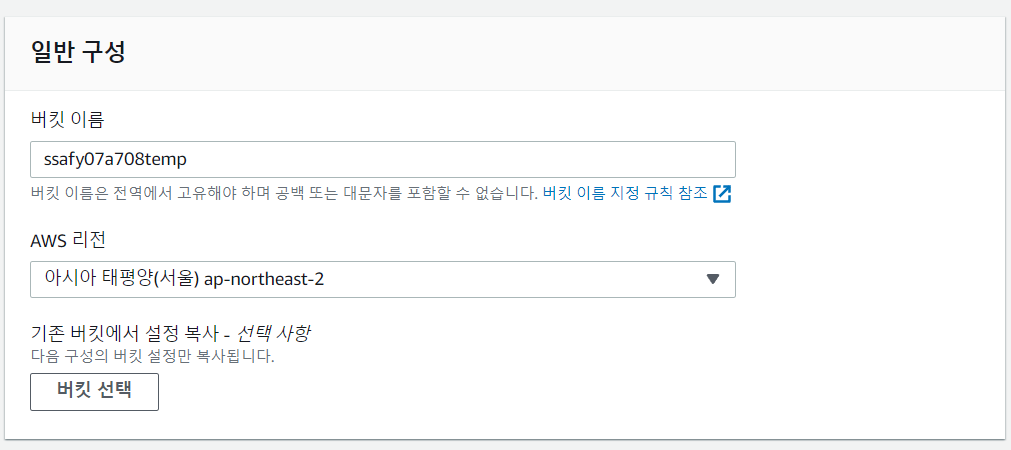
# AWS S3 버킷

사용자의 프로필 이미지를 저장하고, url을 통해 화면에 출력하기 위해 사용하는 클라우드입니다.

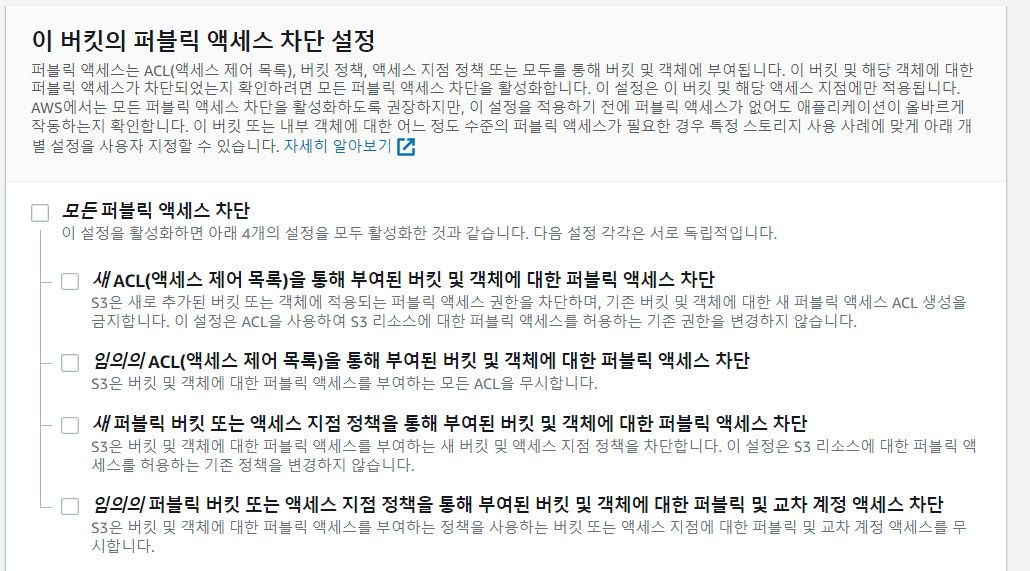
1. 버킷 만들기



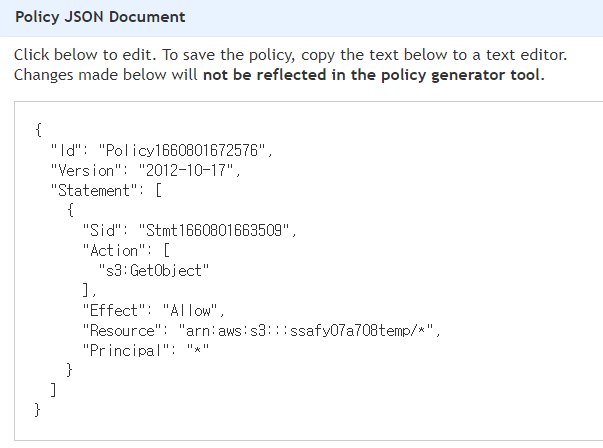
1. 버킷 이름 및 사용 지역 설정



1. 퍼블릭 액세스 차단 해제



1. 버킷 생성 후 권한 > 버킷 정책 > 정책 생성기
   * + Select Type of Policy : S3 Bucket Policy
     + Effect : Allow
     + Principal : \*
     + Actions : getObject() 선택
     + ARN : 버킷 ARN + /\*
       - Ex) arn:aws:s3:::ssafy07a708temp/\*
2. 버킷 정책 편집



1. IAM > 액세스 키 들어가서 키 발급(csv파일)

