DoYou Know?

포팅 매뉴얼

B208

김우빈, 송선아, 이기진, 박호현, 배건길, 최준혁

목차

😉

포팅메뉴얼

1. 개요

프로젝트 사용 도구

- 이슈관리 : Jira
- 형상관리 : GitLab
- 커뮤니케이션 : Notion, Mattermost
- UI/UX : Pigma

프로젝트 개발 환경

Frontend

- Visual Studio Code : 1.70.0
- Vue.js : 3.0
- Node.js : 16.16.0

Backend

- IntelliJ : 11.0.15+10-b2043.56 amd64
- Java : 1.8
- SpringBoot : 2.7.2

DB

- MySQL : 8.0.29

Server

- Ubuntu : 20.04

2. 프로젝트 빌드

프로젝트 빌드방법

Frontend

```
npm i
npm run build
```

Backend

```
Gradle -> build
```

프로젝트 환경변수

Frontend

```
VUE_APP_NEWS_API_KEY=?
```

Backend

```
server:
  port: 포트번호
  error:
    whitelabel:
      enabled: false

  servlet:
    context-path: /api

spring:
  # TODO : PUSH 하기 전에 다시 한번 확인할 것.
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: ec2 mysql 주소
    username: 유저이름
    password: 비밀번호
  mvc:
    pathmatch:
      matching-strategy: ant_path_matcher

  jpa:
    database-platform: org.hibernate.dialect.MySQL5InnoDBDialect
    open-in-view: false
    hibernate:
      format_sql: true
      ddl-auto: update
      show-sql: true

logging:
  level:
    org:
      hibernate:
        SQL: DEBUG
        type:
          descriptor:
            sql:
              BasicBinder: TRACE
    com:
      amazonaws:
        util:
          EC2MetadataUtils: error

cloud:
  aws:
    credentials:
      accessKey: aws accesskey
      secretKey: aws secretkey
    s3:
      bucket: dyk
      path: s3 path
    region:
      static: ap-northeast-2
      auto: false
    stack:
      auto: false
```

GIT Ignore

Frontend

```
.env.local
```

Backend

```
application.yml
```

3. 프로젝트 배포

EC2 [원격 설정]

저장소 세팅 [ubuntu 20.04(LTS)]

```
sudo apt-get update
sudo apt-get install software-properties-common
sudo add-apt-repository universe
sudo apt-get update


[java version 8 설치]
sudo apt-get install openjdk-8-jdk

[nodejs 16.16.0 설치]
sudo curl -sL https://deb.nodesource.com/setup_16.x | sudo -E bash -
sudo apt-get install -y nodejs
node -v
```

certbot 설치

```
sudo apt-get update
sudo apt-get install certbot python3-certbot-nginx
```

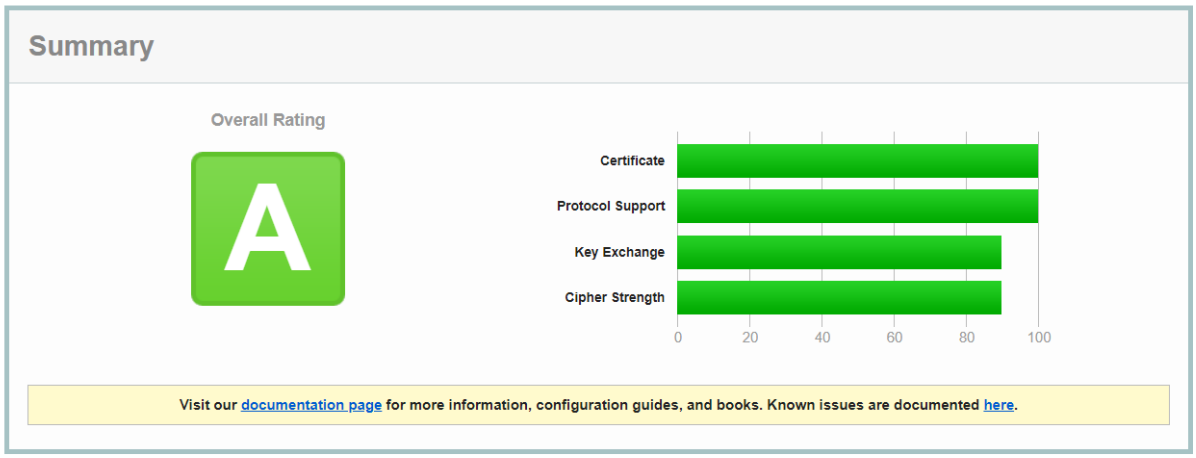
SSL 설정 - certbot 이용 자동화 (유효기간 90일)

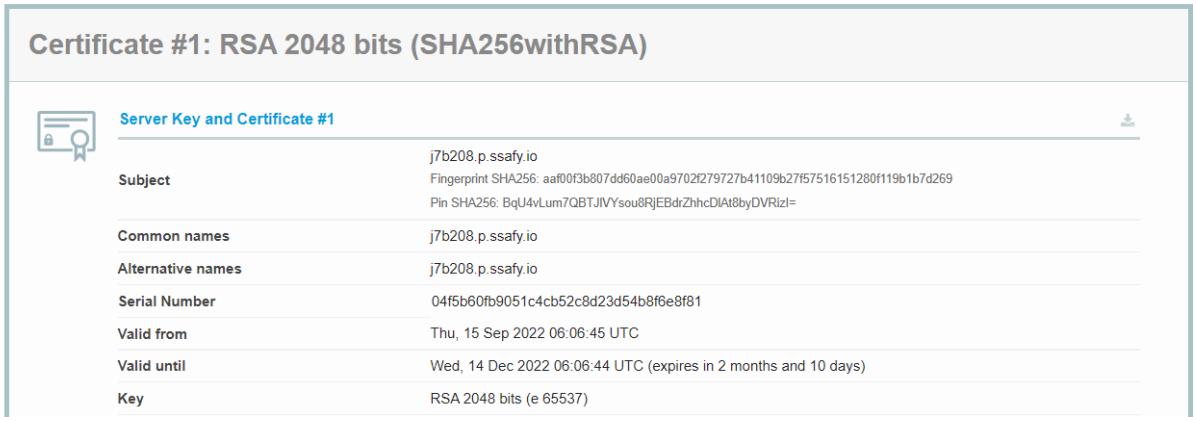
```
[SSL 설정]
sudo certbot --nginx -d j7b208.p.ssafy.io

[갱신 테스트]
sudo certbot renew --dry-run

[인증서 만료일 확인]
certbot ceritificates
```

인증서 파일 위치

/etc/letsencrypt/live/j7b208.p.ssafy.io

https://www.ssllabs.com/ssltest/ - SSL 적용 확인 및 평가






```
[Crontab 보기]
sudo crontab -l

[Crontab 편집]
sudo crontab -e
sudo crontab -l
[Crontab 실행 로그]
view /var/log/syslog
```

```
# ┌──────────── min (0 - 59)
# │ ┌────────────── hour (0 - 23)
# │ │ ┌──────────────── day of month (1 - 31)
# │ │ │ ┌────────────────── month (1 - 12)
# │ │ │ │ ┌──────────────────── day of week (0 - 6) (0 to 6 are Sunday to
# │ │ │ │ │                    Saturday, or use names; 7 is also Sunday)
# │ │ │ │ │
# │ │ │ │ │
# * * * * *  command to execute
```

```
1시간 마다 twitter 데이터 및 spark를 활용하여 감정분석
0 */1 * * * /home/hadoop/twitter/./test


test.sh 내용


#!/bin/bash
python3 /home/hadoop/twitter/getHourlyTwitter.py
sudo rm /home/hadoop/twitter/data/twittersentiment.txt
sudo touch /home/hadoop/twitter/data/twittersentiment.txt
sudo chmod ugo+rwx /home/hadoop/twitter/data/twittersentiment.txt
python3 /home/hadoop/twitter/exporttxt.py
sudo rm -r /home/hadoop/twitter/output
python3 /home/hadoop/twitter/wordcount.py
python3 /home/hadoop/twitter/exportmysql.py
```

j7b208.conf

```
upstream backend {
   server j7b208.p.ssafy.io:8080;
 }
 upstream frontend {
   server j7b208.p.ssafy.io:3000;
 }
 upstream backend2 {
   server j7b208.p.ssafy.io:8000;
 }
 server {
   listen 80;
   server_name j7b208.p.ssafy.io;
   location / {
     return 301 https://$host$request_uri;
   }
 }
 server {
   listen 443 ssl;
   server_name j7b208.p.ssafy.io;
   access_log /var/log/nginx/access.log;
   ssl_certificate /etc/letsencrypt/live/j7b208.p.ssafy.io/fullchain.pem;
   ssl_certificate_key /etc/letsencrypt/live/j7b208.p.ssafy.io/privkey.pem;
   ssl_protocols TLSv1 TLSv1.1 TLSv1.2 SSLv3;
   ssl_ciphers ALL;
   location /api {
     proxy_pass http://backend;
     proxy_redirect off;
     charset utf-8;
     proxy_http_version 1.1;
     proxy_set_header Upgrade $http_upgrade;
     proxy_set_header Host $http_host;
     proxy_set_header X-Real-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;
   }
   location /api2 {
     proxy_pass http://backend2;
     proxy_redirect off;
     charset utf-8;
     proxy_http_version 1.1;
     proxy_set_header Upgrade $http_upgrade;
     proxy_set_header Host $http_host;
     proxy_set_header X-Real-IP $remote_addr;
     proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
     proxy_set_header X-Forwarded-Proto $scheme;
     proxy_set_header X-Nginx-Proxy true;
   }
   location / {
     proxy_pass http://frontend;
     rewrite ^/(.*)$ /$1 break;
     proxy_redirect off;
     charset utf-8;
     proxy_http_version 1.1;
     proxy_set_header Upgrade $http_upgrade;
     proxy_set_header Host $http_host;
     proxy_set_header X-Real-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;
   }
 }openvidu.conf
```

Nginx 명령어

```
//nginx 서버 상태
sudo systemctl status nginx

//nginx 서버 켜기
sudo systemctl start nginx

//nginx 서버 중지
sudo systemctl stop nginx
```

```
//nginx 서버 재시작
sudo systemctl restart nginx
```

CI/CD 자동배포

Docker & Jenkins

Docker 설치

```
sudo apt update
sudo apt install docker-ce docker-ce-cli containerd.io docker-compose
```

docker-compose.yml 파일

```
vim docker-compose.yml

version: '3'

services:
    jenkins:
        image: jenkins/jenkins:lts
        container_name: jenkins
        volumes:
            - /var/run/docker.sock:/var/run/docker.sock
            - /jenkins:/var/jenkins_home
        ports:
            - "9090:8080"
        privileged: true
        user: root



위 내용을 복사하여 붙여넣기합니다. 각 단어에 대한 설명은 간단하게만 하겠습니다.

services : 컨테이너 서비스
jenkins : 서비스 이름
image : 컨테이너 생성시 사용할 image, 여기서는 jenkins/jenkins:lts 이미지를 사용(jenkins의 lts버전을 가져온다는 뜻)
container_name : 컨테이너 이름
volumes : 공유 폴더 느낌, aws의 /var/run/docker.sock와 컨테이너 내부의 /var/run/docker.sock를 연결, /jenkins 폴더와 /var/jenkins_home 폴더를 연결.
ports : 포트 매핑, aws의 9090 포트와 컨테이너의 8080 포트를 연결한다.
privileged : 컨테이너 시스템의 주요 자원에 연결할 수 있게 하는 것 기본적으로 False로 한다고 한다.
user : 젠킨스에 접속할 유저 계정 (root로 할 경우 관리자)
```

Docker 명령어

```
컨테이너를 생성
sudo docker-compose up -d

컨테이너가 올라가 있는 것을 확인
sudo docker ps


여기서 말하는 Administrator password는

sudo docker logs jenkins 명령어를 통해 password 확인
```

port 사용 확인을 위한 tool 설치

```
sudo apt-get install net-tools
netstat -ano [전체 포트 사용 조회]
```

Jenkins 접속(http://j7b208.p.ssafy.io:9090/)

```
[시크릿키 조회]
docker logs jenkins-docker
[또는]
docker exec <CONTAINER_NAME> cat /var/jenkins_home/secrets/initialAdminPassword

[ Admin User ]
ID : b208
PASSWORD : b208
```

jenkins 구성

```
docker image prune -a --force
mkdir -p /var/jenkins_home/images_tar

cp /var/jenkins_home/envforder/.env /var/jenkins_home/workspace/deploytest/Frontend/
cd /var/jenkins_home/workspace/deploytest/Frontend/
docker build -t dykfront .
docker save dykfront > /var/jenkins_home/images_tar/dykfront.tar

cd /var/jenkins_home/workspace/deploytest/Backend/DoYouKnow
docker build -t dykback .
docker save dykback > /var/jenkins_home/images_tar/dykback.tar

cp /var/jenkins_home/envforder/project_setting.py /var/jenkins_home/workspace/deploytest/Backend/Django/
cd /var/jenkins_home/workspace/deploytest/Backend/Django
docker build -t djangoapi .
docker save djangoapi > /var/jenkins_home/images_tar/djangoapi.tar
```

Dockerfile - Springboot

```
FROM openjdk:8 AS builder
COPY gradlew .
COPY gradle gradle
COPY build.gradle .
COPY settings.gradle .
COPY src src
RUN chmod =x ./gradlew
RUN ./gradlew bootJar
#RUN ./gradlew clean build --exclude-task test

FROM openjdk:8
COPY --from=builder build/libs/DoYouKnow-0.0.1-SNAPSHOT.jar DoYouKnow.jar

EXPOSE 8080
CMD ["java","-jar","/DoYouKnow.jar"]
```

Dockerfile - Vue

```
FROM node:16.15.0 as build-stage
WORKDIR /var/jenkins_home/workspace/deploytest/Frontend/frontend
COPY package*.json ./
RUN npm install
COPY . .
RUN npm run build
FROM nginx:stable-alpine as production-stage

COPY --from=build-stage /var/jenkins_home/workspace/deploytest/Frontend/frontend/dist /usr/share/nginx/html

#COPY --from=build-stage /var/jenkins_home/workspace/deploytest/Frontend/frontend/deploy_conf/nginx.conf /etc/nginx/sites-available/de

EXPOSE 80
CMD ["nginx", "-g","daemon off;"]
```

Dockerfile - Django

```
From python:3.9

ENV PYTHONUNBUFFERED 1

RUN apt-get -y update

RUN apt-get -y install vim

COPY ./ /var/jenkins_home/workspace/deploytest/Backend/Django

WORKDIR /var/jenkins_home/workspace/deploytest/Backend/Django


RUN pip3 install --upgrade pip

RUN pip3 install -r requirements.txt

EXPOSE 8000

CMD ["python", "manage.py", "makemigrations"]

CMD ["python", "manage.py", "migrate"]

CMD ["python", "manage.py","runserver","0.0.0.0:8000"]
```

jenkins 빌드후 조치

```
sudo docker load < /jenkins/images_tar/dykfront.tar
sudo docker load < /jenkins/images_tar/dykback.tar
sudo docker load < /jenkins/images_tar/djangoapi.tar

if (sudo docker ps | grep "dykfront"); then sudo docker stop dykfront; fi
if (sudo docker ps | grep "dykback"); then sudo docker stop dykback; fi
if (sudo docker ps | grep "djangoapi"); then sudo docker stop djangoapi; fi

sudo docker run -it -d --rm -p 3000:80 --name dykfront dykfront
echo "Run dykfront"
sudo docker run -it -d --rm -p 8080:8080  --name dykback dykback
echo "Run dykback"
sudo docker run -it -d --rm -p 8000:8000  --name djangoapi djangoapi
echo "Run djangoapi"
```

4. DB설치 및 사용방법

1. ubuntu 패키지 정보 업데이트

sudo apt update

2. mysql 설치

sudo apt install mysql-server

3. mysql 설치 확인

dpkg -l | grep mysql-server

```
root@ip-172-26-3-4:/etc/mysql/mysql.conf.d# dpkg -l | grep mysql-server
ii  mysql-server                 8.0.30-0ubuntu0.20.04.2          all          MySQL database server (metapackage depending on the latest version)
ii  mysql-server-8.0             8.0.30-0ubuntu0.20.04.2          amd64        MySQL database server binaries and system database setup
ii  mysql-server-core-8.0        8.0.30-0ubuntu0.20.04.2          amd64        MySQL database server binaries
```

4. mysql 실행여부 확인

sudo netstat -tap | grep mysql

```
root@ip-172-26-3-4:/etc/mysql/mysql.conf.d# sudo netstat -tap | grep mysql
tcp        0      0 localhost:33060         0.0.0.0:*               LISTEN      340014/mysqld
tcp        0      0 0.0.0.0:mysql           0.0.0.0:*               LISTEN      340014/mysqld
tcp        0      0 ip-172-26-3-4.ap-:mysql 118.42.123.215:63311    ESTABLISHED 340014/mysqld
tcp        0      0 ip-172-26-3-4.ap-:mysql 118.42.123.215:63310    ESTABLISHED 340014/mysqld
```

5. mysql 계정설정

접속

mysql -u root -p

처음 비밀번호는 없으니 엔터

use mysql

select host,user,authentication_string from user;

현재 만든 계정 확인

```
mysql> use mysql
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select Host, User, authentication_string from user;
+-----------+------------------+-------------------------------------------------------------------------+
| Host      | User             | authentication_string                                                   |
+-----------+------------------+-------------------------------------------------------------------------+
| %         | b208             | $A$005$vjh> |'B%UlzQ|
6qGoaKeruLWl/D/8K41hrTPtiml/09mkQYzOhxdR8LgqK1 |
| %         | root             | *C227B91DF2FFA1D588C82BACFEC17C0014BA675A                               |
| localhost | debian-sys-maint | $A$005$a {bP\|::OND
                                                mCd/NHIZzxPvqbXeDfn4wdGqUUdG4Kj5mrWjxkiJLEPCD |
| localhost | mysql.infoschema | $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBRBEUSED |
| localhost | mysql.session    | $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBRBEUSED |
| localhost | mysql.sys        | $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBRBEUSED |
+-----------+------------------+-------------------------------------------------------------------------+
6 rows in set (0.00 sec)
```

alter user 'root'@'localhost' identified with mysql_native_password by '비밀번호 설정';

root 비밀번호 설정

FLUSH PRIVILEGES;

exit

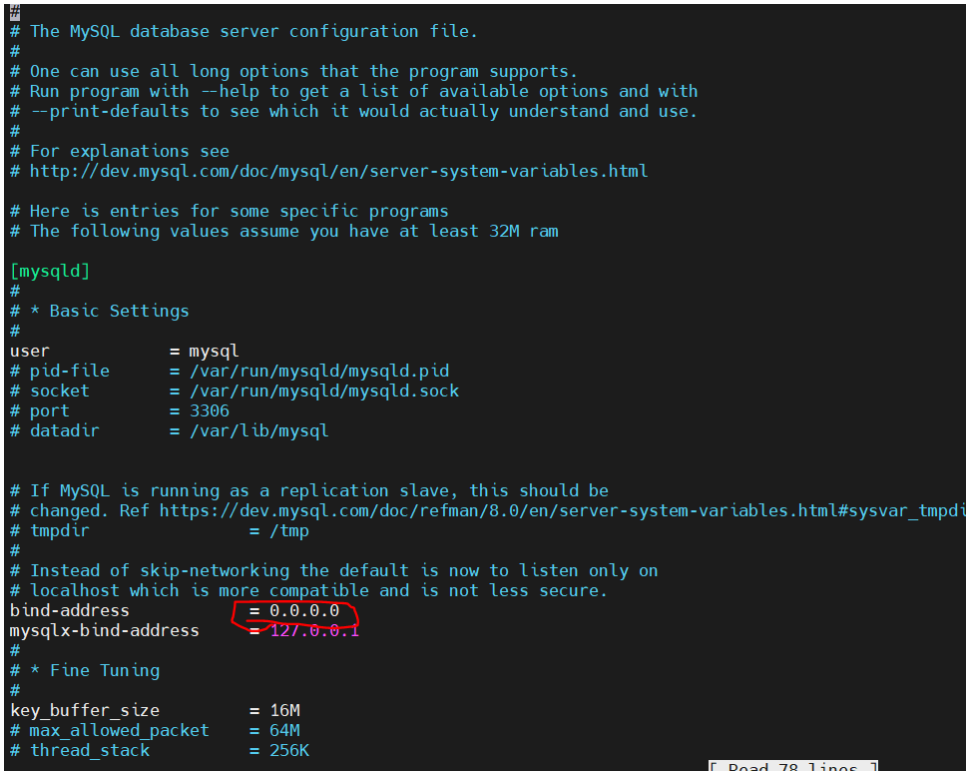
mysql -u root -p 시 이제 비밀번호 치고 들어가야함

6. mysql 외부 접속 설정

cd /etc/mysql/mysql.conf.d

sudo nano mysqld.cnf

bind-address = 0.0.0.0 으로 수정



수행

service mysql restart

mysql -u root -p

원하는 username , password 계정 설정

mysql> create user 'username'@'%' identified by 'password';

만든계정에 모든 권한 주기 설정

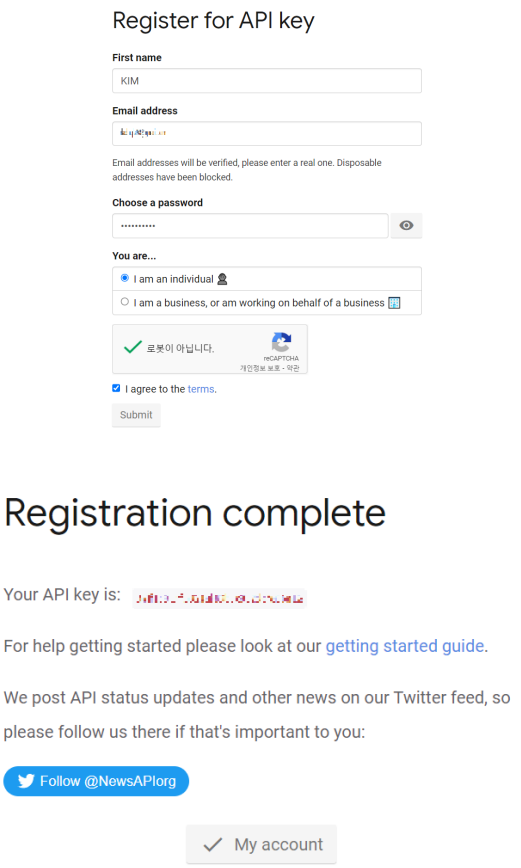
mysql> grant all privileges on . to username@'%';

5. 외부 서비스

NewsAPI

- https://newsapi.org/
- 키워드에 맞는 연관 뉴스를 리턴하는 API

어플리케이션 추가



- Request URL

```
//keyword에 맞는 뉴스리턴
GET https://newsapi.org/v2/everything?apiKey={apikey}&q=${keyword}&sortBy=relevancy
```

Pytrends

- https://pypi.org/project/pytrends/
- 키워드에 맞는 연관검색어를 리턴해주는 API

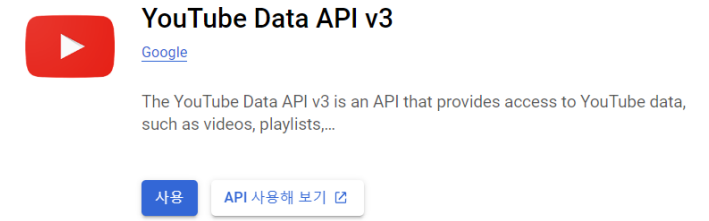
Request

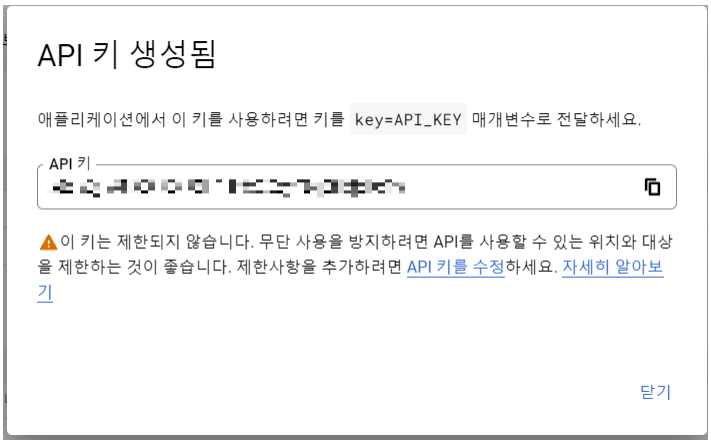
```
//keyword에 맞는 연관검색어 리턴
GET : https://pytrend/relativerisingkeyword/{keyword}
```

유튜브

- https://console.cloud.google.com/apis
- 키워드에 맞는 유튜브 영상 랜더링을 위한 오픈API

어플리케이션 추가



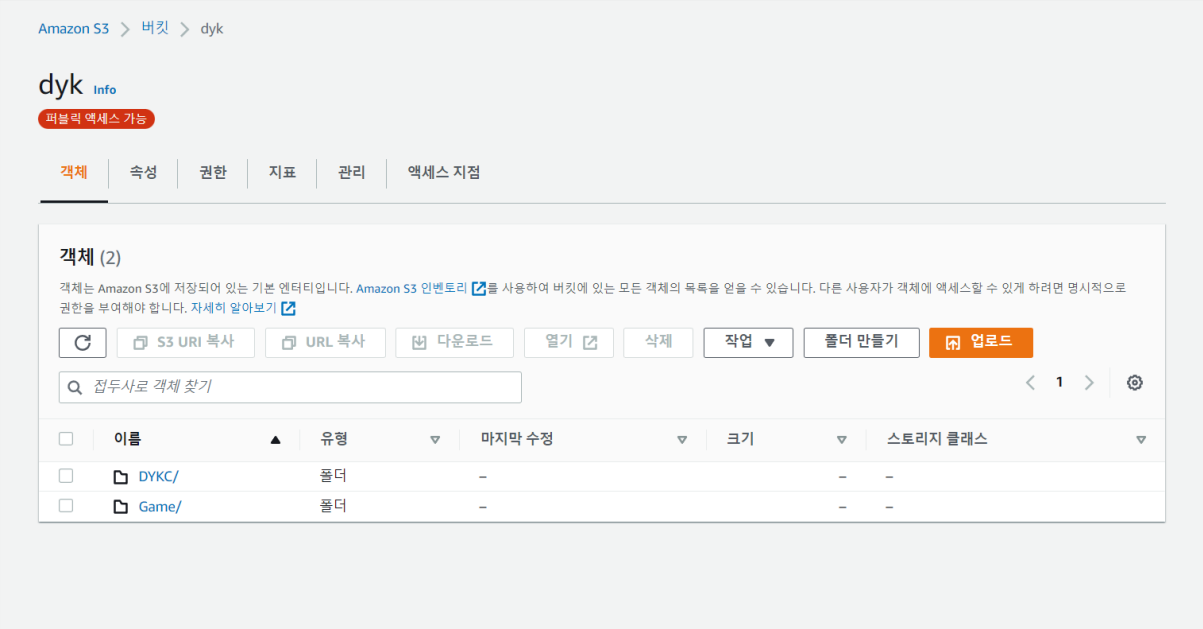


Request

```
//키워드를 통한 유튜브 검색리스트 GET
GET : https://www.googleapis.com/youtube/v3/search
```

Amazon S3

이미지저장



build.gradle

```
//S3
implementation 'org.springframework.cloud:spring-cloud-starter-aws:2.2.6.RELEASE'
```

AwsS3Config.class

```
@Configuration
public class AwsS3Config {

    @Value("${cloud.aws.credentials.access-key}")
    private String accessKey;

    @Value("${cloud.aws.credentials.secret-key}")
    private String secretKey;

    @Value("${cloud.aws.region.static}")
    private String region;

    @Bean
    public AmazonS3Client amazonS3Client() {
        BasicAWSCredentials awsCreds = new BasicAWSCredentials(accessKey, secretKey);
        return (AmazonS3Client) AmazonS3ClientBuilder.standard()
                .withRegion(region)
                .withCredentials(new AWSStaticCredentialsProvider(awsCreds))
                .build();
    }
}
```

S3Uploader .class

```
@Slf4j
@Component
@RequiredArgsConstructor
public class S3Uploader {

    private final AmazonS3Client amazonS3Client;

    @Value("${cloud.aws.s3.bucket}")
    public String bucket;

    public String getPath(String path){
        return amazonS3Client.getUrl(bucket, path).toString();
    }
    public String upload(MultipartFile multipartFile, String dirName) throws IOException{
        File uploadFile = convert(multipartFile).orElseThrow(() -> new IllegalArgumentException("파일 전환 실패"));

        return upload(uploadFile, dirName);
    }
    // S3로 파일 업로드하기
    private String upload(File uploadFile, String dirName) {
        String fileName = dirName + "/" + UUID.randomUUID() + uploadFile.getName();   // S3에 저장된 파일 이름
        putS3(uploadFile, fileName); // s3로 업로드
        removeNewFile(uploadFile);
        return fileName;
    }

    // S3로 업로드
    private String putS3(File uploadFile, String fileName) {
        amazonS3Client.putObject(new PutObjectRequest(bucket, fileName, uploadFile).withCannedAcl(CannedAccessControlList.PublicRead))
        return amazonS3Client.getUrl(bucket, fileName).toString();
    }

    // 로컬에 저장된 이미지 지우기
    private void removeNewFile(File targetFile) {
        if (targetFile.delete()) {
            log.info("File delete success");
            return;
        }
        log.info("File delete fail");
    }

    private Optional<File> convert(MultipartFile multipartFile) throws IOException{
        File convertFile = new File(System.getProperty("user.dir") + "/" + multipartFile.getOriginalFilename());
        // 바로 위에서 지정한 경로에 File이 생성됨 (경로가 잘못되었다면 생성 불가능)
        if (convertFile.createNewFile()) {
            try (FileOutputStream fos = new FileOutputStream(convertFile)) { // FileOutputStream 데이터를 파일에 바이트 스트림으로 저장하기 위함
                fos.write(multipartFile.getBytes());
            }
            return Optional.of(convertFile);
        }
```

```
        return Optional.empty();

    }
}
```