포팅 매뉴얼

A303

1. 포팅 매뉴얼

개발 환경

AWS EC2

- ubuntu : 20.04
- Docker : 23.0.4
- Nginx : 1.18.0

AWS S3

Spring boot

- Java : JDK 11.0.16
- Spring boot 2.7.10
- Intellij : 2022.3.2

Flask

- Python 3.9.10
- Flask : 2.3.2
- SQLAlchemy : 2.0.12
- uWSGI : 2.0.21

Database

- MySQL : 8.0.33
- Redis : 7.0.11 - 6379 port

Android

- Android Studio : Flamingo 2022.2.1
- gradle JDK : jbr-17
- minSDK : 28
- target SDK : 33
- OS : Android 9 +

Web Server 설정

Certbot SSL Certification

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

$ certbot certonly --nginx -d k8a303.p.ssafy.io
```

Nginx : 1.18.0

- 특이사항
  - flask 배포에 사용되는 uWSGI는 unix socket을 사용
  - socket dir `/tmp/flask/flask.sock`
- nginx configuration file : `/etc/nginx/sites-available/groot.conf`

```
server {
  listen 80;
  server_name k8a303.p.ssafy.io;
  return 301 https://k8a303.p.ssafy.io$request_uri;

}
server {
  listen 443 ssl http2;
  server_name k8a303.p.ssafy.io;

  ssl_certificate /etc/letsencrypt/live/k8a303.p.ssafy.io/fullchain.pem;
  ssl_certificate_key /etc/letsencrypt/live/k8a303.p.ssafy.io/privkey.pem;

  client_max_body_size 50M;

  location / {
    proxy_pass http://localhost:3000;
  }

  location /api/recommendations {
    rewrite ^/api(.*)$ $1?$args break;
    include uwsgi_params;
    uwsgi_pass unix:/tmp/flask/flask.sock;
  }

  location /api {
    proxy_pass http://localhost:8080;
    proxy_redirect off;
    charset utf-8;

    proxy_http_version 1.1;
    proxy_set_header Connection "upgrade";
    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 /test {
    alias /home/ubuntu/img-test;
  }
}
```

- create symbolic link

```
$ sudo ln -s /etc/nginx/sites-available/groot.conf /etc/nginx/sites-enabled

$ sudo nginx -s reload
```

프로젝트 빌드 환경 변수

Spring boot

- `/src/main/resources/application-keys.properties`

```
# mysql
mysql.url=
mysql.dbname=
mysql.username=
mysql.password=

# jwt
jwt.secretKey=

# S3
s3.accessKey=
s3.secretKey=
s3.bucket=

# PlantNet
#plantnet.url=
plantnet.apiKey=

# Firebase
firebase.admin-key=

# temp
plant.temp.dir=

# Redis
redis.port=
redis.host=
```

  - MySQL
    - host URL, db name, username, password
  - JWT
    - Secret key
  - S3
    - bucket name, accesskey, secret key
  - PlantNet
    - PlantNet API key
  - temp
    - temp file directory
  - Firebase
    - Firebase Admin SDK private key file class path
  - Redis

        - port, host name
- firebase secret key
    - Class path : `/src/main/SECRET_KEY_PATH/FILE_NAME.json`

Flask

- `/domain/config.py`

```
mysql = {
    'user'     : ,
    'passwd'   : ,
    'host'     : ,
    'database' :
}
```

    - MySQL
        - username, password, host URL, db name

Android

- local.properties

```
## This file must *NOT* be checked into Version Control Systems,
# as it contains information specific to your local configuration.
#
# Location of the SDK. This is only used by Gradle.
# For customization when using a Version Control System, please read the
# header note.
#Thu Apr 20 13:50:49 KST 2023
sdk.dir=

# Naver
naver_client_id =
naver_client_secret =

# KAKAO
kakao_native_app_key =
kakao_oauth_host =

# OpenWeather
weather_api_key =

# Youtube
youtube_api_key =
```

    - SDK - sdk directory
    - Naver, KAKAO
        - naver, kakao social 로그인 key
    - OpenWeather
        - OpenWeather api key
    - Youtube
        - yotube api key

프로젝트 빌드 및 배포

Spring boot

- Dockerfile

```
FROM openjdk:11
ARG JAR_FILE=build/libs/*.jar
COPY ${JAR_FILE} app.jar
ENTRYPOINT ["java","-jar","/app.jar"]
```

- build jar

```
$ cd ./back-end/backend/
$ chmod +x ./gradlew
$ ./gradlew clean build
```

- deploy via docker

```
$ cd ./back-end/backend/
$ docker build -t springboot
$ docker run --name spring -d -p 8080:8080 springboot
```

- deploy

```
$ java -jar {FILE_NAME}.jar
```

Flask

- 특이사항
  - uWSGI 통한 unix socket 사용 : socket directory : `/tmp/flask/`
- Dockerfile

```
FROM python:3.9.10

COPY requirements.txt requirements.txt
RUN pip3 install -r requirements.txt
RUN pip3 install uwsgi

COPY . /app

WORKDIR /app

CMD ["uwsgi", "uwsgi.ini"]
```

- deploy via docker

```
$ cd ./domain/
$ docker built -t flask .
$ docker run -d -v {SOCKET_DIR}:/tmp --name flask flask
```

- socket directory에 맞게 docker volume 내 SOCKET_DIR 수정

Android build

- Android Studio > Build APK

기타 특이사항

AR Asset

- AR 및 사진에 사용되는 캐릭터 Asset은 업로드 후 `glb` 파일과 `png` 파일의 URL을 MySQL 의 `characters` table에 insert 해야합니다

---

2. 외부 서비스 문서

Social Login

- Kakao, Naver social 인증 관련 key 발급
- `front-end/groot/local.properties` 내 관련 정보 등록

OpenWeather

- `front-end/groot/local.properties` 내 관련 정보 등록

Youtube

- `front-end/groot/local.properties` 내 관련 정보 등록

PlantNet API

- Pl@ntNet API https://my.plantnet.org/ 접속 후 계정 생성
- 계정 생성 후 API key 발급
  - 발급받은 API Key를 `back-end/backend/src/main/resources` 의 `application-keys.properties`파일 plantnet.apiKey에 등록
- 무료 계정의 경우 일별 500회 제한

Firebase

- Google Firebase https://console.firebase.google.com/u/0/ 접속 후 계정 생성
- 계정 생성 후 프로젝트 생성
- Android app에 firebase 추가 > 구성파일 다운로드 후 `front-end/groot/app` 에 `google-services.json` 파일 추가
- Messaging, Realtime Database, Firestore database 추가
- 프로젝트 설정 > 서비스 계정 > Firebase Admin SDK > 배공개 키 생성 후 `back-end/backend/src/main/resources` 하위 directory에 key 파일 저장 후 `application-keys.properties` 파일 firebase.admin-key에 classpath 작성