포팅 매뉴얼

1. 배포 요구사항

프로젝트 Application을 배포하기 위해선 아래와 같은 사양을 요구한다.

1) Ubuntu Linux 20.04 LTS
2) Node.js
3) Java 11
4) Docker
5) Nginx
6) SSL Certificate Key (Let’s Encrypt)
7) MySQL - Server 8.0.34
8) ✨Elasticsearch 8.7.1 - Kibana

✨표시는 Docker Container로 설치 및 가동한 Application이다.

2. 프로젝트 구조

GitLab에 저장된 프로젝트 코드의 전체 구조는 다음과 같다.

Name	Last commit	Last update
backend	feat:백엔드프로젝트 생성	1 month ago
frontend	fix: 전체버그수정	6 hours ago
README.md	docs: README 구조 작성	8 hours ago

Frontend 전체 구조는 다음과 같다.

Name	Last commit	Last update
..		
public	fix: 회원가입 확인	2 days ago
src	fix: 전체버그수정	7 hours ago
.gitignore	feat: 프론트엔드프로젝트 생성	1 month ago
README.md	feat: api 유저 및 복지사업 추가	3 weeks ago
package-lock.json	feat : Drag & Drop	1 day ago
package.json	feat : Drag & Drop	1 day ago

Backend 전체 구조는 다음과 같다.

Name	Last commit	Last update
..		
gradle/wrapper	style: 폴더 구조 변경	1 month ago
src	Merge branch 'S09P22C209-246-last-read-time' into BE	19 hours ago
.gitignore	style: 폴더 구조 변경	1 month ago
Dockerfile	refactor: dockerfile 복구	2 weeks ago
build.gradle	feat: writed Elasticsearch Rest API for Spring	3 weeks ago
default.conf	feat: nginx.conf, default.conf file added and Dockerfile fixed	2 weeks ago
gradlew	style: 폴더 구조 변경	1 month ago
gradlew.bat	style: 폴더 구조 변경	1 month ago
nginx.conf	feat: nginx.conf, default.conf file added and Dockerfile fixed	2 weeks ago
settings.gradle	style: 폴더 구조 변경	1 month ago

3. 프로젝트 배포

Application 배포를 위해 사전 작업을 거쳐야 한다. 모든 작업은 요구사항에서 제시한 Ubuntu Linux에서 진행함을 전제한다.

또한 GitLab Project 파일을 참조해야 할 경우 Root Directory는 `S09P22C209` 이다.

1) Docker Install → 최신 버전으로 설치한다.

a) Elasticsearch Container Pull and Run

```
docker run -d -p 9200:9200 -p 9300:9300 -e "discovery.type=single-node" -e ES_JAVA_OPTS="-Xms1g -Xmx1g" -e TZ=Asia/Seoul --name elk-e docker.elastic.co/elasticsearch/elasticsearch:8.7.1
```

b) Kibana Container Pull and Run

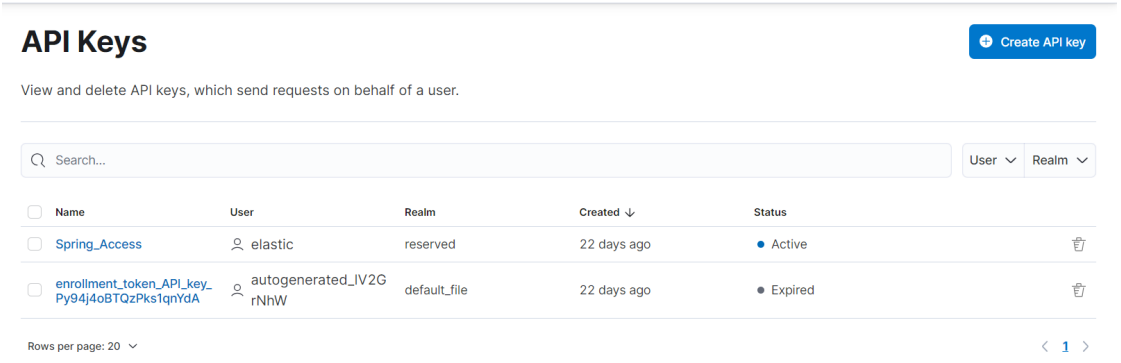
```
docker run -d -p 5601:5601 -e TZ=Asia/Seoul --name elk-k docker.elastic.co/kibana/kibana:8.7.1
```

c) 두 컨테이너를 모두 설치 및 실행했으면 `docker exec -it elk-e /bin/bash` 및 `bin/elasticsearch-reset-password --username elastic -i` 명령어를 순서대로 실행하여 Elasticsearch에 접근할 비밀번호를 설정한다.

d) Kibana 접속에 필요한 Elasticsearch 접근 Token을 발급 받는 명령어 `bin/elasticsearch-create-enrollment-token -s kibana` 을 실행하고, Token을 기억해 둔다.

e) `exit` 명령어를 실행하여 Elasticsearch 컨테이너로부터 접속을 종료하고, 브라우저로 `http://localhost:5601` 주소로 접속하여 인증 절차를 실시한다.

f) 최종적으로 `http://localhost:5601/app/management/security/api_keys` 으로 이동하여 아래와 같이 API Key를 발급 받아야 Spring으로부터 Elasticsearch로의 접근이 가능해진다.



2) MySQL-Server Install → 제시한 버전으로 설치해야 문제가 발생하지 않는다.

설치가 끝났으면 프로젝트 파일에서 `exec` Directory 내의 `dream_231006.sql` 파일을 `mysql -u root -p [password] dream < dream_231006.sql` 명령어를 실행한다.

(당연히 명령어를 실행할 현재 위치에서 sql 파일이 존재해야 한다)

3) Project Frontend Build to Docker Container

a) 루트 디렉토리에서 `frontend` 디렉토리로 이동한다.

b) `npm install` 명령어를 실행하여 의존성을 모두 확인 및 설치한다.

c) `npm run build` 명령어를 실행하여 React App을 배포 가능하도록 빌드한다.

d) 현재 위치 그대로, `docker build -t ${docker_repo}:front-0.1 .` 명령어를 실행한다. `${docker_repo}` 는 docker 사용자 계정 및 repository를 의미한다.

e) 마지막으로 `docker run -d --name front-app --restart=always -p 80:80 -p 443:443 -p 8090:8090 --add-host host.docker.internal:host-gateway --volume /etc/letsencrypt/:/etc/letsencrypt ${docker_repo}:front-0.1` 명령어를 실행한다. 이를 위해서 요구사항에서 설명한 SSL 인증서를 미리 발급 받을 필요가 있다.

4) Project Backend Build to Docker Container

a) 루트 디렉토리에서 `backend` 디렉토리로 이동한다.

b) `./gradlew clean bootJar` 명령어를 실행하여 Spring Project를 빌드한다.

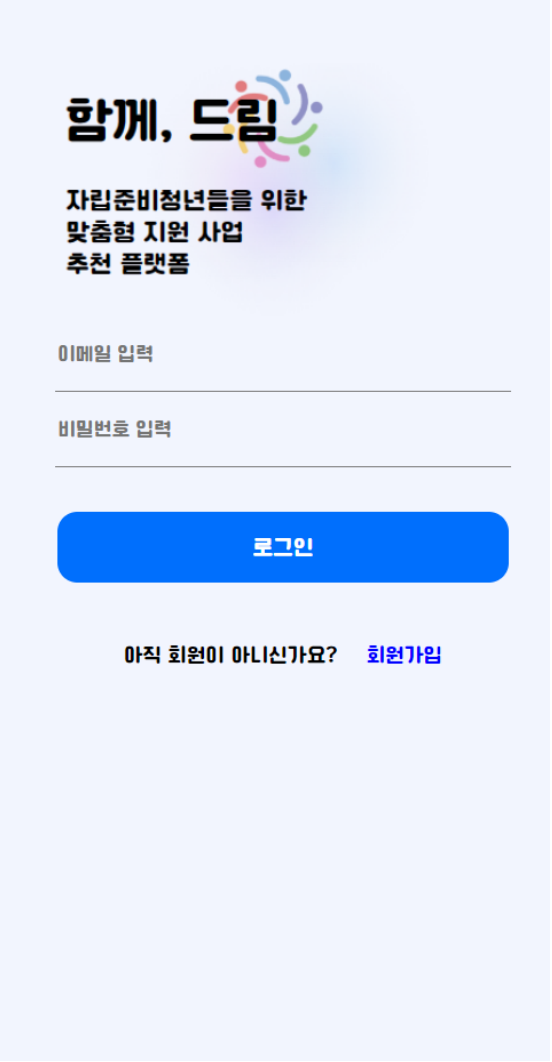
만약 권한 문제가 발생하였다면 `sudo` 권한 등으로 실행할 필요가 있다.

c) `docker build -t ${docker_repo}:back-0.1 .` 명령어를 실행한다.

d) 마지막으로 `docker run -d --name back-server --restart=always -p 8085:8080 --add-host host.docker.internal:host-gateway --volume /etc/letsencrypt/:/etc/letsencrypt ${docker_repo}:back-0.1` 명령어를 실행한다.

5) Web Application 접속

`https://localhost` 홈페이지로 이동하여 배포 상태를 확인한다.



6) Elasticsearch Data Synchro

`http://localhost:9200/korean_analyzer_stopwords` 주소로 Post 요청을 한다.

주의) 위 요청을 보내기 전, `exec` 디렉토리 내의 `user_dic.txt` `synonyms.txt` `stopwords.txt` 파일을 모두 Elasticsearch Container로 접속하여 `/usr/share/elasticsearch/config` 디렉토리에 모두 복사해야 한다. 절대 Host Server가 아닌 Elasticsearch Container임에 주의한다.

이때, Request Body에 다음과 같이 JSON 형식의 내용을 입력한다.

```
{
    "settings": {
        "analysis": {
            "analyzer": {
                "nori_analyzer": {
                    "type": "custom",
                    "tokenizer": "korean_nori_tokenizer",
                    "filter": [
                        "nori_filter",
                        "custom_stop",
                        "custom_synonym"
                    ]
                }
            },
            "tokenizer": {
                "korean_nori_tokenizer": {
                    "type": "nori_tokenizer",
                    "decompound_mode": "mixed",
                    "user_dictionary": "user_dic.txt"
                }
            },
            "filter": {
                "nori_filter": {
                    "type": "nori_part_of_speech",
                    "stoptags": ["E", "IC", "J", "NNB", "NP", "NR", "MAJ", "MAG",
"VV", "VA", "VX", "VCP", "VCN", "MM", "XSV"]
                },
                "custom_stop": {
                    "type": "stop",
                    "stopwords_path": "stopwords.txt"
                },
                "custom_synonym": {
                    "type": "synonym",
                    "synonyms_path": "synonyms.txt"
                }
            }
        }
    }
}
```

- 또한 Request Header에 아래와 같이 항목을 추가한다. ApiKey 뒤에 들어갈 항목은 Kibana 접속 후 발급 받은 API Key 이다.

마지막으로 `https://localhost:8090/welfare_info/synchro` 주소로 접속하여 MySQL DB와 Elasticsearch 간 데이터 동기화를 진행한다.

4. 보안 인증 키

1. Frontend로부터 Backend로의 접속이 불가하다면, `\frontend\src\api` 파일에 접근한 뒤, `baseURL` 변수를 Backend Container가 가동되는 주소를 입력한다.
2. Backend로부터 MySQL로의 접속이 불가하다면, `\backend\src\main\resources\applicaion.yml` 파일에 접근하여 MySQL 접속 정보를 수정한다.
3. Backend로부터 Elasticsearch으로의 접속이 불가하다면,

   `\backend\src\main\java\com\dream\backend\elastic\service\ElasticConnectionService.java` 파일에 접근하여 `serverUrl` 및 `apiKey` 변수를 위 단계에서 진행한 결과에 따라 서버 주소 및 Kibana API Key를 입력한다.