

포팅 매뉴얼

1) 사용한 JVM, 웹서버, WAS 제품 등의 종류와 설정값, 버전(IDE버전 포함) 기재

- IDE

IntelliJ IDEA Ultimate 2022.2.2

- Node.js

[node-v14.17.0](#)

- Vue.js

v2.7.10

- JDK

openjdk 11.0.16 2022-07-19

OpenJDK Runtime Environment (build 11.0.16+8-post-Ubuntu-0ubuntu120.04)

OpenJDK 64-Bit Server VM (build 11.0.16+8-post-Ubuntu-0ubuntu120.04, mixed mode, sharing)

- Spring Boot

v2.5.2

- Apache Tomcat (내장 아파치 톰캣 WAS)

[org.apache.tomcat.embed](#)

» [tomcat-embed-core](#)

v9.0.48

- 웹서버 및 리버스 프록시 서버

nginx version: nginx/1.22.0 (Ubuntu)

docker pull nginx:stable-alpine

port: 80(HTTP), 443(HTTPS)

- Docker

Docker version 20.10.18, build b40c2f6

- MySQL

v8.0.30-1.el8

docker pull mysql:8.0

port: 3306

- Jenkins CI/CD

v2.361.1

docker pull jenkins

port: 8090

2) 빌드 시 사용되는 환경변수 등 주요 내용 상세 기재

▼ build.gradle 파일

```
buildscript {
    ext {
        querydslVersion = "5.0.0"
    }
}

plugins {
    id 'org.springframework.boot' version '2.7.5'
    id 'io.spring.dependency-management' version '1.0.15.RELEASE'
    id "com.ewerk.gradle.plugins.querydsl" version "1.0.10"
    id 'java'
}
```

```

}

group = 'com.togedocs'
version = '0.0.1-SNAPSHOT'
sourceCompatibility = '11'

configurations {
    compileOnly {
        extendsFrom annotationProcessor
    }
}

repositories {
    mavenCentral()
}

dependencies {
    implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
    implementation 'org.springframework.boot:spring-boot-starter-data-mongodb'
    implementation 'org.springframework.boot:spring-boot-starter-jdbc'
    implementation 'org.springframework.boot:spring-boot-starter-oauth2-client'
    implementation 'org.springframework.boot:spring-boot-starter-security'
    implementation 'org.springframework.boot:spring-boot-starter-validation'
    implementation 'org.springframework.boot:spring-boot-starter-web'
    implementation 'org.springframework.boot:spring-boot-starter-web-services'
    implementation 'org.springframework.boot:spring-boot-starter-websocket'
    compileOnly 'org.projectlombok:lombok'
    developmentOnly 'org.springframework.boot:spring-boot-devtools'
    runtimeOnly 'com.mysql:mysql-connector-j'
    annotationProcessor 'org.projectlombok:lombok'
    testImplementation 'org.springframework.boot:spring-boot-starter-test'
    testImplementation 'org.springframework.security:spring-security-test'
    implementation 'io.jsonwebtoken:jjwt:0.9.1'
    implementation "com.querydsl:querydsl-jpa:${queryDslVersion}"
    annotationProcessor "com.querydsl:querydsl-apt:${queryDslVersion}"
}

def queryDslDir = "$buildDir/generated/querydsl"

querydsl {
    jpa = true
    querydslSourcesDir = queryDslDir
}

sourceSets {
    main.java.srcDir queryDslDir
}

compileQuerydsl {
    options.annotationProcessorPath = configurations.querydsl
}

configurations {
    querydsl.extendsFrom compileClasspath
}

tasks.named('test') {
    useJUnitPlatform()
}

```

▼ Spring Boot Application 빌드 Dockerfile

```

FROM gradle:7.4-jdk-alpine

WORKDIR /backend

COPY . .

RUN gradle clean build --no-daemon

EXPOSE 8081

CMD ["java", "-jar", "build/libs/backend-0.0.1-SNAPSHOT.jar"]

```

- `-jar` : jar파일 실행

▼ Nginx sites-available/default 설정 파일

```

server {
    listen 80 default_server;
    listen [::]:80 default_server;

```

```

server_name k7a404.p.ssafy.io;

return 301 https://$server_name$request_uri;
}

server {
    listen 443 ssl;
    listen [::]:443 ssl;

    root /usr/share/nginx/html;
    index index.html index.htm;

    server_name k7a404.p.ssafy.io;

    ssl_certificate /etc/letsencrypt/live/k7a404.p.ssafy.io/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/k7a404.p.ssafy.io/privkey.pem;

    location / {
        try_files $uri $uri/ /index.html;
    }

    location /api {
        proxy_pass http://backend:8081;
        proxy_http_version 1.1;
        proxy_set_header Connection "";
        proxy_set_header Host $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-Forwarded-Host $host;
        proxy_set_header X-Forwarded-Port $server_port;
    }

    location /docs {
        proxy_pass http://backend:8081;
    }
}

```

- 443(HTTPS), 80(HTTP) port
 - `/` 경로: 정적인 리소스
 - `/api` 경로: `http://localhost:8081` 스프링 부트 애플리케이션으로 리버스 프록시 됨
 - `/docs` 경로: REST API 문서

3) 배포 시 특이사항 기재

A. Docker + Jenkins CI/CD 설정 과정

| S | W | Name | 최근 성공 | 최근 실패 | 최근 소요 시간 |
|---|---|------------------|------------------|-------------------|--------------|
| 🔄 | 🚀 | backendTest | 1 day 20 hr #207 | 2 days 21 hr #106 | 3.5 sec |
| ✅ | 🚀 | Docker Test | 17 days #5 | 17 days #1 | 28 ms |
| ✅ | 🚀 | gitlab_test | — | — | — |
| ✅ | 🚀 | jenkins_test | 2 hr 5 min #317 | 2 days 19 hr #340 | 2 sec |
| ✅ | 🚀 | TopoDocs_backend | 1 hr 48 min #260 | 1 day 20 hr #173 | 1 min 28 sec |

Jenkins 설정

- Jenkins port: 8090
- 아래와 같이 Front-end, Back-end CI/CD 구축을 위한 아이템 2개 생성
- Jenkins Item 구성은 다음과 같이 설정

Git ?

Repositories ?


Repository URL ? ✕

https://lab.ssfy.com/s07-final/S07P31A404.git

Credentials ?

dlsk1233@naver.com/***** ▼

+ Add

 고급...

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ? ✕

*/develop

Add Branch

빌드 유발

- ☐ 빌드를 원격으로 유발 (예: 스크립트 사용) ?
- ☐ Build after other projects are built ?
- ☐ Build periodically ?
- ☒ Build when a change is pushed to GitLab. GitLab webhook URL: http://k7a404.p.ssfy.io:8090/project/TogeDocs_backend ?
- Enabled GitLab triggers
- ☒ Push Events
- ☐ Push Events in case of branch delete
- ☒ Opened Merge Request Events
- ☐ Build only if new commits were pushed to Merge Request ?
- ☐ Accepted Merge Request Events
- ☐ Closed Merge Request Events
- Rebuild open Merge Requests
- Never ▼
- ☒ Approved Merge Requests (EE-only)
- ☒ Comments
- Comment (regex) for triggering a build ?
- Jenkins please retry a build

☒ Enable [ci-skip]

☒ Ignore WIP Merge Requests

Labels that forces builds if they are added (comma-separated)

☒ Set build description to build cause (eg. Merge request or Git Push)

☐ Build on successful pipeline events

Pending build name for pipeline ?

☐ Cancel pending merge request builds on update

Allowed branches

☒ Allow all branches to trigger this job ?

☐ Filter branches by name ?

☐ Filter branches by regex ?

☐ Filter merge request by label

Secret token ?

Generate

Build Steps

Execute shell ?

Command

See [the list of available environment variables](#)

```

# JenkinsTest/backend 도커 이미지 생성, JenkinsTest/backend는
docker build -t backend:latest ./backend
# JenkinsTest/backend 도커 컨테이너 실행
docker run -d --name backend -p 8081:8081 --network devops backend:latest
  
```

고급...

- 다음의 명령어 수행: `sudo docker network create devops` : docker 컨테이너간의 네트워크 생성
- Jenkins 컨테이너 안에서 Docker를 실행하면 Host의 도커와 연결됨
- Dockerfile을 /backend 와 /frontend에 각각 하나씩 넣고 nginx.conf는 /frontend에 넣음

Spring Boot_application.yml 수정

- `server.address=localhost`로 되어있어 스프링부트로 HTTP 요청을 보낼 수 없었음. 따라서 해당 설정 삭제함.

```

server:
  address: localhost <= 이거 없어야함
servlet:
  encoding:
    force: 'true'
    charset: UTF-8
    enabled: 'true'
  contextPath: /
  port: '8081'
  
```

Docker_Frontend 명령어

- `-v /etc/letsencrypt/:/etc/letsencrypt/` 로 경로를 잡은 이유는 pem키가 symlink로 저장되었기 때문에 letsencrypt경로로부터 volume mount 해야함!!!

Docker_Frontend Dockerfile

```
FROM node:current-slim as build-stage

WORKDIR /frontend
# root 다음 Test을 지정 즉 호스트 디렉토리

COPY . .
# 앞의 점은 app파일의 모든 디렉토리 뒤의 점은 도커의 root directory

RUN npm install
RUN npm run build

CMD ["npm", "start"]

FROM nginx:stable-alpine as production-stage

RUN rm /etc/nginx/conf.d/default.conf
COPY ./nginx.conf /etc/nginx/conf.d/nginx.conf
COPY --from=build-stage ./frontend/dist /usr/share/nginx/html

EXPOSE 80 443

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

Docker_Backend 명령어

```
# 도커 시작 전, 기존에 실행중인 도커를 멈추고 제거하는 작업.
docker ps -f name=backend -q | xargs --no-run-if-empty docker container stop

# 컨테이너 제거
docker container ls -a -f name=backend -q | xargs -r docker container rm

# backend 도커 이미지 생성
docker build -t backend:latest ./backend

# backend 도커 컨테이너 실행
docker run -d --name backend -p 8081:8081 -v /home/ubuntu/properties:/home/ubuntu/properties/ --network idontknownetwork backend:late
```

Docker_Backend Dockerfile

```
FROM gradle:7.4-jdk-alpine

WORKDIR /backend

COPY . .

RUN gradle clean build --no-daemon

EXPOSE 8081

CMD ["java", "-jar", "build/libs/backend-0.0.1-SNAPSHOT.jar"]
```

nginx.conf

```
server {
    listen 80 default_server;
    listen [::]:80 default_server;

    server_name k7a404.p.ssafy.io;

    return 301 https://$server_name$request_uri;
}

server {
    listen 443 ssl;
    listen [::]:443 ssl;

    root /usr/share/nginx/html;
    index index.html index.htm;
```

```

server_name k7a404.p.ssafy.io;

ssl_certificate /etc/letsencrypt/live/k7a404.p.ssafy.io/fullchain.pem;
ssl_certificate_key /etc/letsencrypt/live/k7a404.p.ssafy.io/privkey.pem;

location / {
    try_files $uri $uri/ /index.html;
}

location /api {
    proxy_pass http://backend:8081;
    proxy_http_version 1.1;
    proxy_set_header Connection "";
    proxy_set_header Host $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-Forwarded-Host $host;
    proxy_set_header X-Forwarded-Port $server_port;
}

location /docs {
    proxy_pass http://backend:8081;
}
}

```

4) API 명세서



API 명세서

표 +

필터

| # No | 분류 | Aa 이름 | URI Pattern | METHOD | 비고 |
|------|------|----------------------------|--------------------------------------|-----------|---|
| 1 | 공통변집 | ■ 행 추가 | /docs/{projectId}/rows | POST | |
| 2 | 공통변집 | ■ 열 추가 | /docs/{projectId}/cols | POST | |
| 3 | 공통변집 | ■ 행 옮기기 | /docs/{projectId}/rows | PATCH | |
| 4 | 공통변집 | ■ 열 옮기기 | /docs/{projectId}/cols | PATCH | |
| 5 | 공통변집 | ■ 행 삭제 | /docs/{projectId}/rows/{rowId} | DELETE | |
| 6 | 공통변집 | ■ 열 삭제 | /docs/{projectId}/cols/{colId} | DELETE | |
| 7 | 공통변집 | ■ 셀 수정 | /docs/{projectId}/cell | PATCH | |
| 8 | 공통변집 | ■ 문서 가져오기 | /docs/{projectId} | GET | |
| 9 | 공통변집 | ■ 프로젝트 정보 수정 | /docs/{projectId} | PATCH | |
| 10 | 공통변집 | baseURL 수정 | /docs/{projectId}/url | PATCH | |
| 11 | 공통변집 | ■ 열 수정 | /docs/{projectId}/cols/{colId} | PATCH | |
| 12 | 공통변집 | REFRESH 요청 | /pub/{projectId}/refresh | WebSocket | |
| 13 | 공통변집 | ■ 포커스 변경 | /pub/{projectId}/focus | WebSocket | |
| 14 | 프로젝트 | ■ 프로젝트 생성 | /project | POST | |
| 15 | 프로젝트 | ■ 프로젝트 삭제 | /project/{projectId} | DELETE | 매니저만 권한이 있음 |
| | 프로젝트 | ■ 초대코드로 프로젝트 들어가기 | /project/join | POST | + 이미 있다면 예외 처리 |
| 16 | 프로젝트 | ■ 팀원 관리 조회 (팀원 목록 & 초대 코드) | /project/{projectId}/members | GET | |
| 19 | 프로젝트 | ■ 팀원 추방 | /project/{projectId}/member/{userId} | DELETE | 매니저만 권한이 있음. 매니저는 1명 이상 남아있어야 함. + 이미 없다면 예외 처리 |
| 20 | 프로젝트 | ■ 팀원 권한 수정 | /project/{projectId}/member | PATCH | 매니저만 권한이 있음. 매니저는 1명 이상 남아있어야 함. |

| | | | | | |
|----|---------|----------------|--------------------------|--------|---|
| | 프로젝트 | 초대코드로 프로젝트 찾기 | /project/code/{code} | GET | |
| | | 프로젝트에서 나가기 | ??? | DELETE | |
| 21 | 회원 | 소셜 로그인 | | POST | projectId로 find함 이건 API 호출 없음 |
| 22 | 회원 | 회원 정보 수정 | /user/info | PATCH | 이름, 프로필 사진 수정 |
| 23 | 회원 | 회원 탈퇴 | /user/withdraw | DELETE | project_user 테이블에서 삭제 만약 탈퇴하려는 사람이 혼자 매니저인 프로젝트라면, 권한을 넘기고 탈퇴하라고 알림. 혼자인 프로젝트라면, 프로젝트까지 삭제한다고 알림. |
| 24 | 회원 | ■ 프로젝트 목록 조회 | /user/project | GET | |
| 25 | API 테스트 | 로그 전체조회 | /log/{projectId}/{rowId} | GET | |
| 26 | API 테스트 | ■ 로그 남기기(등록) | /log/{projectId}/{rowId} | POST | |
| 27 | | 프로젝트 api 목록 조회 | | | |

▼ application.yml

- local: 개발 환경
- prod: 배포 환경

```
spring:
  profiles:
    group:
      local:
        - local
        - secret
      prod :
        - secret
        - db
  jpa:
    hibernate:
      naming:
        implicit-strategy: org.springframework.boot.orm.jpa.hibernate.SpringImplicitNamingStrategy
        physical-strategy: org.springframework.boot.orm.jpa.hibernate.SpringPhysicalNamingStrategy
      ddl-auto: update
      dialect: org.hibernate.dialect.MySQL8Dialect
      format_sql: true
      default_batch_fetch_size: 100
      generate-ddl: true
  data:
    web:
      pageable:
        one-indexed-parameters: true
  devtools:
    livereload:
      enabled: 'true'

# server
server:
  servlet:
    encoding:
      force: 'true'
      charset: UTF-8
      enabled: 'true'
    contextPath: /
  port: '8081'
build:
  date: '@build.date@'

# log
logging:
  level:
    org:
      springframework:
        security: DEBUG
        web: DEBUG
      apache:
        tiles: INFO
      hibernate:
        SQL: DEBUG
      root: INFO
      com:
        samsung:
          security: DEBUG
  file:
    name: './ssafy-web.log'

# jwt token
jwt:
  access-token-props:
    secret: adsjkQWFRaeiasjodfiwAWeeifjaSDOFJaiewAEWgIREAJORaerjAOESJOGDASIKFJIAJqiojuerfiAE
```



```

    expiration-time-milli-sec: '10800000'
  refresh-token-props:
    expiration-time-milli-sec: '864000000'
    secret: ZqefBscadfsGjaiADFSGaoGHdsjjsDFksjgDskdjgDjgHJmdSlzLfjbhikjaqWSogvjdjsjzUEvLkvvhndFGVnkfa==
  ---
# local only
#spring.config.activate.on-profile: local

# database
spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    username: a404
    password: k7a4@4r@@t
    url: jdbc:mysql://k7a404.p.ssafy.io:3306/togedocs?useUnicode=true&characterEncoding=utf8&serverTimezone=Asia/Seoul&zeroDateT
imeBehavior=convertToNull&rewriteBatchedStatements=true
  data:
    mongodb:
      uri: mongodb://root:k7a4%404r%40%40t@k7a404.p.ssafy.io:27017/togedocs
#security
security:
  oauth2:
    client:
      registration:
        google:
          client-id: 693779976943-29nocg5vr041lva4a0colr1mm1b813j8.apps.googleusercontent.com
          client-secret: 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
          scope:
            - email
            - profile

```

구글 API Console