동감 포팅 매뉴얼

프로젝트 기간 : 2023.10.10 ~ 2023.11.17 (6 주)

E107 감자튀김

팀장 : 최동우

팀원 : 공정민, 박진희, 양불회, 정현우

목차

1. 사용한 JVM, 웹서버, WAS 제품 등의 종류와 설정 값, 버전 기재

1. Java

   1.8
2. 웹서버

   NginX
3. WAS

   Tomcat
4. Gradle

   6.8.3
5. Spring Boot

   2.7.17
6. Node

   18.18.2
7. React

   18.2.0
8. Yarn

   1.22.19
9. Recoil

   0.7.7
10. Vite

    4.4.5
11. Intellij

    2023.1.5
12. Visual Studio Code

    1.18.1
13. MariaDB

    Docker image : mariadb:latest
14. PostgreSQL

    Docker image : mdillon/postgis:latest
15. Redis

    Docker image : redis:lastest

2. 빌드 시 사용되는 환경 변수 등의 내용

Backend

application.properties

```
# Mariadb
spring.datasource.url=jdbc:mariadb://127.0.0.1:3306/ditto?characterEncoding=UTF-8
spring.datasource.username=${username}
spring.datasource.password=${password}
spring.jpa.show-sql=true
spring.jpa.open-in-view=true
spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.format_sql=true


# JWT
jwt.secret-key=${jwtSecret}
jwt.expiration=1000*60*60*24

# nginx
spring.profiles.active=set1

# PostgreSQL
spring.second-datasource.url=jdbc:postgresql://localhost:5432/ditto
spring.second-datasource.username=postgres
spring.second-datasource.password=${password}
spring.second-
datasource.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
spring.second-datasource.jpa.hibernate.ddl-auto=update

# redis
spring.redis.host=127.0.0.1
spring.redis.port=6379

# Swagger
springdoc.use-fqn=true

# Mattermost
notification.mattermost.enabled=true
notification.mattermost.webhook-url=${webhook}
notification.mattermost.color=red
notification.mattermost.footer=Footer Text

# S3
cloud.aws.s3.bucket=french-fries
cloud.aws.stack.auto=false
cloud.aws.region.static=ap-northeast-2
cloud.aws.credentials.accessKey=${S3access}
cloud.aws.credentials.secretKey=${S3secret}

# Image Upload size limit
spring.servlet.multipart.max-file-size=10MB
spring.servlet.multipart.max-request-size=10MB

# Time zone setting
spring.jackson.time-zone=Asia/Seoul
spring.jpa.properties.hibernate.jdbc.time_zone=Asia/Seoul

#---
spring.config.activate.on-profile=set1
server.port=8080

#---
```

```
spring.config.activate.on-profile=set2
server.port=8081
```

build.gradle

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

plugins {
    id 'java'
    id 'org.springframework.boot' version '2.7.17'
    id 'io.spring.dependency-management' version '1.0.15.RELEASE'
}

group = 'com.example'
version = '0.0.1-SNAPSHOT'

java {
    sourceCompatibility = '1.8'
}

configurations {
    compileOnly {
        extendsFrom annotationProcessor
    }
}

repositories {
    mavenCentral()
}

dependencies {
    // mariadb
    implementation 'org.hibernate:hibernate-core:5.6.15.Final'
    implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
    implementation 'org.springframework.boot:spring-boot-starter-jdbc'
    implementation group: 'org.mariadb.jdbc', name: 'mariadb-java-client', version:
'2.4.1'

    // PostgreSQL
    runtimeOnly 'org.postgresql:postgresql'

    // hibernate-spatial - jts Point 쓰기 위해서 추가
  implementation 'org.hibernate:hibernate-spatial:5.6.11.Final'

    // query dsl
    implementation "com.querydsl:querydsl-jpa"
    implementation "com.querydsl:querydsl-core"
    implementation "com.querydsl:querydsl-collections"
    annotationProcessor "com.querydsl:querydsl-
apt:${dependencyManagement.importedProperties['querydsl.version']}:jpa" // querydsl
JPAAnnotationProcessor 사용 지정
  annotationProcessor "jakarta.annotation:jakarta.annotation-api" //
java.lang.NoClassDefFoundError (javax.annotation.Generated) 대응 코드
  annotationProcessor "jakarta.persistence:jakarta.persistence-api" //
java.lang.NoClassDefFoundError (javax.annotation.Entity) 대응 코드
```

```
  // swagger
    implementation 'org.springdoc:springdoc-openapi-ui:1.6.6'

    // mattermost
    implementation 'com.google.code.gson:gson:2.8.5'

    // health check
    implementation 'org.springframework.boot:spring-boot-starter-actuator'

    // security
    implementation 'org.springframework.boot:spring-boot-starter-security'

    // json
    implementation 'org.json:json:20210307'
    implementation 'io.jsonwebtoken:jjwt-api:0.11.2'
    runtimeOnly 'io.jsonwebtoken:jjwt-jackson:0.11.2'
    runtimeOnly 'io.jsonwebtoken:jjwt-impl:0.11.2'
//  implementation 'io.jsonwebtoken:jjwt:0.9.1'

    // s3
    implementation 'org.springframework.cloud:spring-cloud-starter-aws:2.2.6.RELEASE'

    // web socket
    implementation 'org.springframework.boot:spring-boot-starter-websocket'

    // redis
    implementation 'org.springframework.boot:spring-boot-starter-data-redis'

    // firebase
    implementation 'com.google.firebase:firebase-admin:9.1.1'

    implementation 'org.springframework.boot:spring-boot-starter-web'
    compileOnly 'org.projectlombok:lombok'
    developmentOnly 'org.springframework.boot:spring-boot-devtools'
    annotationProcessor 'org.projectlombok:lombok'
    testImplementation 'org.springframework.boot:spring-boot-starter-test'
}


// Querydsl 설정부
def generated = 'src/main/generated'

// querydsl QClass 파일 생성 위치를 지정
tasks.withType(JavaCompile) {
    options.getGeneratedSourceOutputDirectory().set(file(generated))
}

// java source set 에 querydsl QClass 위치 추가
sourceSets {
    main.java.srcDirs += [ generated ]
}

// gradle clean 시에 QClass 디렉토리 삭제
clean {
    delete file(generated)
}

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

3. 배포 시 특이사항

NginX 설정

```
server {
	listen 443 ssl default_server;
	listen [::]:443 ssl default_server;
	ssl on;

	ssl_certificate /etc/letsencrypt/live/k9e107.p.ssafy.io/fullchain.pem;
	ssl_certificate_key /etc/letsencrypt/live/k9e107.p.ssafy.io/privkey.pem;

	root /var/www/html;

	# Add index.php to the list if you are using PHP
	index index.html index.htm index.nginx-debian.html;

	server_name _;

	include /etc/nginx/conf.d/service-url.inc;

	location / {
		# First attempt to serve request as file, then
		# as directory, then fall back to displaying a 404.
		# try_files $uri $uri/ =404;
		proxy_pass $service_url;
	}
}
```

myapp.conf

환경 변수

client_secret=Am9dwTw7bvgdeN94b5J65zABZ0GLeroF password=ssafy
client_id=3e96fcc37dbab390f429c7e1bd803fdc username=root
webhook=https://meeting.ssafy.com/hooks/s3rypxdoz3n838pgfzddecqw3r
S3access=AKIASTQWHCXK2ZUHUP4S S3secret=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
jwtSecret=-
wyErY1L7MRCox5UxXgT786d8sfjh2dd4tkllIsSEKQOOIkdWdkdsfsdlfkasjfpoutpoaisdjflkfjfdhowkrdmsepw
haejrlfrgpgodigkskwlsWksjanrlsepdlrsjsm

myapp.service

[Unit]

```
Description=myapp start

[Service]
EnvironmentFile=/home/ubuntu/myapp.conf
ExecStart=/bin/bash -c "exec java -jar /home/ubuntu/*.jar"

[Install]
WantedBy=multi-user.target
```

sudo systemctl restart myapp.service

4. DB 접속 정보

----MariaDB---

Host : k9e107.p.ssafy.io

Port : 3306

User : root

Password : ssafy

Database : ditto

설치

```
docker pull mariadb

docker run --name mariadb -e MYSQL_ROOT_PASSWORD=ssafy -e MYSQL_DATABASE=ditto -e
MYSQL_USER=root -e MYSQL_PASSWORD=ssafy -p 3306:3306 mariadb
```

----PostGIS---

Host : k9e107.p.ssafy.io

Port : 5432

User : postgres

Password : ssafy

Database : ditto

설치

```
docker pull mdillon/postgis

docker run --name ditto-postgres -e POSTGRES_PASSWORD=ssafy -e POSTGRES_DB=ditto -e TZ=Asia/Seoul -p 5432:5432 -d mdillon/postgis
```

----Redis----

Host : k9e107.p.ssafy.io

Port : 6379

설치

```
docker pull redis

docker run --name redis -p 6379:6379 redis
```