빌드 및 배포 문서

목차

1. 개발 환경

- node.js : 18.16.1
- vscode : 1.82.3
- react native : 0.72.5
- jdk : 17.0.8
- springboot : 3.1.3
- intellij : 2023.1.4
- docker : 24.0.6
- mysql : 8.0.33
- nginx : 1.18.0 (Ubuntu)
- jenkins : 2.422

2. 설정파일

springboot

- build.gradle

```
plugins {
  id 'java'
  id 'org.springframework.boot' version '3.1.3'
  id 'io.spring.dependency-management' version '1.1.3'
}

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

java {
  sourceCompatibility = '17'
}

configurations {
  compileOnly {
    extendsFrom annotationProcessor
  }
}

repositories {
  mavenCentral()
}
```

```
dependencies {

  // spring boot
  implementation 'org.springframework:spring-context-support'
  developmentOnly 'org.springframework.boot:spring-boot-devtools'
  implementation 'org.springframework.boot:spring-boot-starter-web'
  implementation 'org.springframework.boot:spring-boot-starter-mail'
  implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
  implementation 'org.springframework.boot:spring-boot-starter-security'
  implementation 'org.springframework.boot:spring-boot-starter-data-redis'
  implementation 'org.springframework.boot:spring-boot-starter-oauth2-client'
//  implementation 'org.springframework.cloud:spring-cloud-starter-config'
//  implementation 'org.springframework.cloud:spring-cloud-dependencies:2021.0.0'
//  implementation 'org.springframework.boot:spring-boot-starter-actuator'
  // validation
  implementation 'org.springframework.boot:spring-boot-starter-validation'

  // test
  testImplementation 'org.springframework.boot:spring-boot-starter-test'
  testImplementation 'org.springframework.security:spring-security-test'

  // lombok
  compileOnly 'org.projectlombok:lombok'
  annotationProcessor 'org.projectlombok:lombok'

  // mysql
  runtimeOnly 'com.mysql:mysql-connector-j'

  // querydsl
  implementation 'com.querydsl:querydsl-jpa:5.0.0:jakarta'
  annotationProcessor "com.querydsl:querydsl-apt:${dependencyManagement.importedProperties['querydsl.version']}:jakarta"
  annotationProcessor "jakarta.annotation:jakarta.annotation-api"
  annotationProcessor "jakarta.persistence:jakarta.persistence-api"

  // queryparameter log
  // implementation 'com.github.gavlyukovskiy:p6spy-spring-boot-starter:1.9.0'

  // swagger
  implementation 'org.springdoc:springdoc-openapi-starter-webmvc-ui:2.0.2'

  // jwt
  implementation 'io.jsonwebtoken:jjwt-api:0.11.5'
  implementation 'io.jsonwebtoken:jjwt-impl:0.11.5'
  implementation 'io.jsonwebtoken:jjwt-jackson:0.11.5'
}

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

//def querydslSrcDir = 'build/generated'
//clean {
//  delete file(querydslSrcDir)
//}
//tasks.withType(JavaCompile) {
//  options.generatedSourceOutputDirectory = file(querydslSrcDir)
//}
```

application.yml

```
spring:
  profiles:
    include: dev, env  #dev
#    include: server, env  #server

  ## 이미지 파일 크기 제한
  servlet:
    multipart:
      max-file-size: 10MB
      max-request-size: 30MB
      #enabled: true  # multipart 에 대한 일을 처리하게끔 해준다(기본값)

  ## swagger 에러 해결
  mvc:
    pathmatch:
      matching-strategy: ant_path_matcher

  ## spring boot 2.5 이상 버전의 경우 자동으로 실행되지 않아
  ## spring.sql.init.mode=always 프로퍼티 값을 입력해주어야 한다.
  sql:
```

```
    init:
      mode: never
#      mode: always


server:
  ## UTF-8 인코딩 설정
  servlet:
    encoding:
      charset: UTF-8
      force: true

# Swagger springdoc-ui Configuration
springdoc:
  packages-to-scan: com.ssafy.donworry.api.controller
  default-consumes-media-type: application/json;charset=UTF-8
  default-produces-media-type: application/json;charset=UTF-8
  swagger-ui:
    path: demo-ui.html            # Swagger UI 경로 => localhost:8001/demo-ui.html
    tags-sorter: alpha            # alpha: 알파벳 순 태그 정렬, method: HTTP Method 순 정렬
    operations-sorter: alpha      # alpha: 알파벳 순 태그 정렬, method: HTTP Method 순 정렬
  api-docs:
    path: /api-docs/json
    groups:
      enabled: true
  cache:
    disabled: true


## AWS S3를 연동한 프로젝트에서 발생하는 오류 방지
## 프로젝트 배포시 기본으로 CloudFormation 구성을 시작하기 때문에
## 설정한 CloudFormation이 없으면 프로젝트 실행이 되지 않음.
## 해당 기능을 사용하지 않도록 false로 설정.
cloud:
  aws:
    stack:
      auto: false
```

application-dev.yml, application-server.yml

```
server:
  port: 8001

spring:
  datasource:
    url: ${MYSQL_URL}
    username: ${MYSQL_USER}
    password: ${MYSQL_PASSWORD}
    driver-class-name: com.mysql.cj.jdbc.Driver
  redis:
    host: j9c210.p.ssafy.io
    port: 6379
  jpa:
    hibernate:
      ddl-auto: update
    properties:
      hibernate:
        dialect: org.hibernate.dialect.MySQL8Dialect
        format_sql: true
        highlight_sql: true
        default_batch_fetch_size: 100
        ## reserved words error config (add backticks)
        auto_quote_keyword: true
        globally_quoted_identifiers: true
    ## ddl-auto 먼저 진행 후 script 진행
    defer-datasource-initialization: true
  mail:
    host: smtp.gmail.com
    port: 587
    username: ${EMAIL_ID}
    password: ${EMAIL_PASSWORD}
    properties:
      mail:`
        smtp:
          auth: true
          starttls:
            enable: true
            required: true
          connectiontimeout: 5000
          timeout: 5000
          writetimeout: 5000
```

```
    auth-code-expiration-millis: 1800000  # 30 * 60 * 1000 == 30분

## log 설정 trace -> debug -> info -> warn -> error -> fatal
logging:
  level:
    org:
      hibernate:
        SQL: trace
    com:
      ssafy:
        donworry: debug
  file:
    name: /log.log

## image file 설정
file:
  dir: /var/images/
```

application-env.yml

```
##db
MYSQL_USER: <secret>
MYSQL_PASSWORD: <secret>
MYSQL_URL: jdbc:mysql://<도메인 url>/donworry?serverTimezone=UTC&useUniCode=yes&characterEncoding=UTF-8

##jwt
JWT_SECRET: <secret>


##email
EMAIL_ID: <email>
EMAIL_PASSWORD: <password>

##social-google
OAUTH2_GOOGLE_ID: <secret>
OAUTH2_GOOGLE_PASSWORD: <secret>

##social-kakao
OAUTH2_KAKAO_REDIRECT_URL: http://localhost:3000/oauth/kakao/callback
OAUTH2_KAKAO_REST_API: <secret>


---

##dev
spring:
  config:
    activate:
      on-profile: dev

##db
MYSQL_USER: <secret>
MYSQL_PASSWORD: <secret>
MYSQL_URL: jdbc:mysql://<도메인 url>/donworry?serverTimezone=UTC&useUniCode=yes&characterEncoding=UTF-8

---

##server
spring:
  config:
    activate:
      on-profile: server

##db
MYSQL_USER: <secret>
MYSQL_PASSWORD: <secret>
MYSQL_URL: jdbc:mysql://<도메인 url>/donworry?serverTimezone=UTC&useUniCode=yes&characterEncoding=UTF-8
```

front env

```
APP_ENV_KAKAO_API_KEY = <secret>
APP_ENV_REDIRECT_URI = <secret>
```

3. 배포관련

nginx default conf파일

```
##
# You should look at the following URL's in order to grasp a solid understanding
# of Nginx configuration files in order to fully unleash the power of Nginx.
# https://www.nginx.com/resources/wiki/start/
# https://www.nginx.com/resources/wiki/start/topics/tutorials/config_pitfalls/
# https://wiki.debian.org/Nginx/DirectoryStructure
#
# In most cases, administrators will remove this file from sites-enabled/ and
# leave it as reference inside of sites-available where it will continue to be
# updated by the nginx packaging team.
#
# This file will automatically load configuration files provided by other
# applications, such as Drupal or Wordpress. These applications will be made
# available underneath a path with that package name, such as /drupal8.
#
# Please see /usr/share/doc/nginx-doc/examples/ for more detailed examples.
##

# Default server configuration
#
server {
  listen 80 default_server;
  listen [::]:80 default_server;

  # SSL configuration
  #
  # listen 443 ssl default_server;
  # listen [::]:443 ssl default_server;
  #
  # Note: You should disable gzip for SSL traffic.
  # See: https://bugs.debian.org/773332
  #
  # Read up on ssl_ciphers to ensure a secure configuration.
  # See: https://bugs.debian.org/765782
  #
  # Self signed certs generated by the ssl-cert package
  # Don't use them in a production server!
  #
  # include snippets/snakeoil.conf;

  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 _;

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

  # pass PHP scripts to FastCGI server
  #
  #location ~ \.php$ {
  # include snippets/fastcgi-php.conf;
  #
  # # With php-fpm (or other unix sockets):
  # fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
  # # With php-cgi (or other tcp sockets):
  # fastcgi_pass 127.0.0.1:9000;
  #}

  # deny access to .htaccess files, if Apache's document root
  # concurs with nginx's one
  #
  #location ~ /\.ht {
  # deny all;
  #}
}


# Virtual Host configuration for example.com
#
# You can move that to a different file under sites-available/ and symlink that
# to sites-enabled/ to enable it.
#
#server {
# listen 80;
```

```
# listen [::]:80;
#
# server_name example.com;
#
# root /var/www/example.com;
# index index.html;
#
# location / {
#   try_files $uri $uri/ =404;
# }
#}

server {

  # SSL configuration
  #
  # listen 443 ssl default_server;
  # listen [::]:443 ssl default_server;
  #
  # Note: You should disable gzip for SSL traffic.
  # See: https://bugs.debian.org/773332
  #
  # Read up on ssl_ciphers to ensure a secure configuration.
  # See: https://bugs.debian.org/765782
  #
  # Self signed certs generated by the ssl-cert package
  # Don't use them in a production server!
  #
  # include snippets/snakeoil.conf;

  root /var/www/html;

  client_max_body_size 30M;

  # Add index.php to the list if you are using PHP
  index index.html index.htm index.nginx-debian.html;
      server_name j9c210.p.ssafy.io; # managed by Certbot


  #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;
    proxy_pass http://172.22.1.3:3000;

  }

  location /api {
                proxy_pass http://172.27.1.3:8001;
        }

  location /api/notifications/subscribe {
                proxy_pass http://172.27.1.3:8001;
                proxy_set_header Connection '';
                proxy_set_header Cache-Control 'no-cache';
                proxy_set_header X-Accel-Buffering 'no';
                proxy_set_header Content-Type 'text/event-stream';
                proxy_buffering off;
                chunked_transfer_encoding on;
                proxy_read_timeout 86400s;
        }

        location /api1{
                proxy_pass http://172.28.1.3:8002;
        }

      location /api1/alarm/subscribe {
          proxy_pass http://172.28.1.3:8002;
          proxy_set_header Connection '';
          proxy_set_header Cache-Control 'no-cache';
          proxy_set_header X-Accel-Buffering 'no';
          proxy_set_header Content-Type 'text/event-stream';
          proxy_buffering off;
          chunked_transfer_encoding on;
          proxy_read_timeout 86400s;
      }

        location ~ ^/(swagger|webjars|configuration|swagger-resources|v2|csrf|demo-ui.html) {
               proxy_pass http://172.27.1.3:8001;
               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;
        }

  # pass PHP scripts to FastCGI server
  #
  #location ~ \.php$ {
  # include snippets/fastcgi-php.conf;
  #
  # # With php-fpm (or other unix sockets):
  # fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
  # # With php-cgi (or other tcp sockets):
  # fastcgi_pass 127.0.0.1:9000;
  #}

  # deny access to .htaccess files, if Apache's document root
  # concurs with nginx's one
  #
  #location ~ /\.ht {
  # deny all;
  #}


    listen [::]:443 ssl ipv6only=on; # managed by Certbot
    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/j9c210.p.ssafy.io/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/j9c210.p.ssafy.io/privkey.pem; # managed by Certbot
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

}
server {
    if ($host = j9c210.p.ssafy.io) {
        return 301 https://$host$request_uri;
    } # managed by Certbot


  listen 80 ;
  listen [::]:80 ;
    server_name j9c210.p.ssafy.io;
    return 404; # managed by Certbot


}
```

springboot.dockerfile

```
FROM openjdk:17-jdk-slim

ARG JAR_FILE=build/*.jar

COPY ${JAR_FILE} app.jar

EXPOSE 8001

ENTRYPOINT [ "java", "-jar", "/app.jar" ]
```

springboot.docker-compose

```
version: "3.3"

services:

  springboot:
    container_name: "springboot"

    build:
      context: "."
      dockerfile: "springboot.dockerfile"

    volumes:
      - "./images:/var/images"
      - "./logs:/var/log"

    expose:
      - "8001"

    networks:
      default_bridge:
```

```
        ipv4_address: 172.27.1.3


networks:
  default_bridge:
    ipam:
      driver: default
      config:
        - subnet: 172.27.1.0/24
```

mysql.docker-compose

```
version: "3.3"

services:

  mysql:
    container_name: "mysql"

    image: "mysql:8.0.33"

    restart: "always"

    ports:
      - "3306:3306"

    volumes:
      - "/home/ubuntu/donworry/default/mysql_home:/var/lib/mysql"

    command:
      - "--character-set-server=utf8mb4"
      - "--collation-server=utf8mb4_unicode_ci"
      - "--skip-character-set-client-handshake"

    env_file:
      - "./env/mysql.env"
```

redis.docker-compose

```
version: "3.3"

services:
  redis:
    container_name: "redis"
    image: "redis"
    ports:
      - "6379:6379"
```

jenkins CI/CD Pipeline Script

```
node {
    stage('Initial'){
        sh "pwd"
        sh "rm -r ./**"
    }
    stage('Pull') {
        cleanWs()
       checkout scmGit(branches: [[name: '*/develop']], extensions: [], userRemoteConfigs: [[credentialsId: 'donworry', url: 'https://l
    }

    stage('Change application.yml'){
        sh "pwd"
        sh "cp /var/jenkins_home/application.yml /var/jenkins_home/workspace/donworry/backend/src/main/resources/"
        sh "cp /var/jenkins_home/application-env.yml /var/jenkins_home/workspace/donworry/backend/src/main/resources/"
        sh "cp /var/jenkins_home/application-server.yml /var/jenkins_home/workspace/donworry/backend/src/main/resources/"
    }

    stage('Build'){
        dir('backend') {
            // some block
            sh "pwd"
            sh "chmod +x gradlew"
            withGradle {
                // some block
                sh './gradlew clean'
```

```
                sh './gradlew bootJar'
            }
        }
    }
    stage('ssh back') {
        dir('backend') {
        sshPublisher(publishers: [sshPublisherDesc(configName: 'c210ec2',
        transfers: [sshTransfer(cleanRemote: false, excludes: '', execCommand: '''
        cd /home/ubuntu/donworry/be
        sudo docker-compose down
        sudo docker image rm be-springboot
        sudo docker-compose up -d
        cd /home/ubuntu/donworry/redis
        sudo docker-compose down
        sudo docker-compose up -d''',
        execTimeout: 120000, flatten: false, makeEmptyDirs: false, noDefaultExcludes: false, patternSeparator: '[, ]+', remoteDirectory
        }

    }
}
```