포팅 매뉴얼

목차

1. 개발환경

- Server: Ubuntu 20.04.6 LTS
- JDK: 11.0.16
- Node.js: 18

- Nginx: 1.27.4
- MySQL: 8.0.41
- Redis: 7.4.1
- Jenkins: 2.492.1
- Vscode: 1.96.4
- Intellij: 2023.3.8

2. 설정파일 및 환경 변수 정보

Spring

▼ application-prod.properties

```
#it will be set build date by gradle. if this value is @build.date@, front-end
build.date=@build.date@
server.port=8076
server.address=0.0.0.0
server.ssl.enabled=true
server.ssl.key-store=<key-store 위치>
server.ssl.key-store-password=<ssl 비밀번호>
server.ssl.key-store-type=PKCS12
server.ssl.key-alias=tomcat
server.ssl.protocol=TLS
server.servlet.contextPath=/
# Charset of HTTP requests and responses. Added to the "Content-Type"
server.servlet.encoding.charset=UTF-8
# Enable http encoding support.
server.servlet.encoding.enabled=true
# Force the encoding to the configured charset on HTTP requests and res
server.servlet.encoding.force=true

spring.jackson.time-zone=Asia/Seoul

# for SPA
spring.web.resources.static-locations=classpath:/dist/
spa.default-file=/dist/index.html
spring.mvc.throw-exception-if-no-handler-found=true
```

```
spring.web.resources.add-mappings=false

# Swagger
springfox.documentation.swagger.use-model-v3=false

#database
spring.jpa.hibernate.naming.implicit-strategy=org.springframework.boot.c
spring.jpa.hibernate.naming.physical-strategy=org.springframework.boot.
spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL57Dia
spring.data.web.pageable.one-indexed-parameters=true
spring.datasource.url=jdbc:mysql://<mysql 주소>/E103_DB?useUnicode=t
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.username=<db user 이름>
spring.datasource.password=<db user 비밀번호>

# jwt
jwt.secret=dyAeHubOOc8KaOfYB6XEQoEj1QzRlVgtjNL8PYs1A1tymZvvqkc
# unit is ms. 15 * 24 * 60 * 60 * 1000 = 15days
jwt.expiration=1296000000

#logging
logging.file.name=./ssafy-web.log
logging.level.root=INFO
logging.level.com.samsung.security=DEBUG
logging.level.org.springframework.web=DEBUG
logging.level.org.apache.tiles=INFO
logging.level.org.sringframework.boot=DEBUG
logging.level.org.sringframework.security=DEBUG



spring.devtools.livereload.enabled=true

#gzip compression
server.compression.enabled=true
server.compression.mime-types=application/json,application/xml,text/htm

#for health check
```

```
management.servlet.context-path=/manage
management.health.db.enabled=true
management.health.default.enabled=true
management.health.diskspace.enabled=true

# Redis Configuration
spring.redis.host=<redis 주소>
spring.redis.port=<redis 포트 번호>
spring.redis.timeout=60000
spring.redis.database=0
spring.redis.lettuce.pool.max-active=8
spring.redis.lettuce.pool.max-idle=8
spring.redis.lettuce.pool.min-idle=0
spring.redis.lettuce.pool.max-wait=-1ms
# Keyspace Notification (TTL Listener)
spring.redis.notify-keyspace-events=Ex

# SMTP server setting (Gmail)
spring.mail.host=smtp.gmail.com
spring.mail.port=587
spring.mail.username=<smtp 주소>
spring.mail.password=<smtp 비밀번호>

# SMTP authentication and security setting
spring.mail.properties.mail.smtp.auth=true
spring.mail.properties.mail.smtp.starttls.enable=true
spring.mail.properties.mail.smtp.starttls.required=true
spring.mail.properties.mail.smtp.connectiontimeout=5000
spring.mail.properties.mail.smtp.timeout=5000
spring.mail.properties.mail.smtp.writetimeout=5000
```

Nginx

▼ default.conf

```
server {
    listen 80;
    server_name <도메인 명>;
```

```
    location /.well-known/acme-challenge/ {
        alias /var/www/certbot/.well-known/acme-challenge/;
        allow all;
    }

    location / {
        return 301 https://$host$request_uri;
    }
}

server {
    listen 443 ssl;
    server_name <도메인 명>;

    ssl_certificate /etc/letsencrypt/live/<도메인 명>/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/<도메인 명>/privkey.pem;

    location / {
        root /usr/share/nginx/html;
        index index.html;
        try_files $uri $uri/ /index.html;
    }

    location /.well-known/acme-challenge/ {
        root /var/www/certbot;
        allow all;
    }

    location /api/ {
        proxy_pass https://<도메인 명>:<spring-boot 포트>;
        proxy_http_version 1.1;
        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 Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
```

```
        proxy_cache_bypass $http_upgrade;
    }
}
```

3. 인증서 발급

cerbot 설치

```
sudo apt update
sudo apt install certbot
```

인증서 발급 요청

```
sudo certbot certonly --standalone -d <도메인 이름>
```

발급 확인

```
cat /etc/letsencrypt/live/<도메인 이름>/fullchain.pem
```

권한 설정

```
sudo chmod 644 /etc/letsencrypt/archive/i12e103.p.ssafy.io/cert1.pem
sudo chmod 644 /etc/letsencrypt/archive/i12e103.p.ssafy.io/chain1.pem
sudo chmod 644 /etc/letsencrypt/archive/i12e103.p.ssafy.io/fullchain1.pem
sudo chmod 644 /etc/letsencrypt/archive/i12e103.p.ssafy.io/privkey1.pem
```

4. Turn Server 구축

coturn 설치

```
sudo apt-get update
sudo apt-get install coturn
```

시스템 시작 시 coturn 자동 시작 살정

```
sudo vi /etc/default/coturn

TURNSERVER_ENABLED=1
```

/etc/turnserver.conf 설정

```
listening-port=3478
tls-listening-port=5349
listening-ip=<사설IP>
external-ip=<공인IP>/<사설IP>
relay-ip=<사설IP>

fingerprint
lt-cred-mech
user=계정아이디:계정비밀번호
realm=도메인 주소
server-name=도메인명

cert=/etc/letsencrypt/live/도메인 주소/fullchain.pem
pkey=/etc/letsencrypt/live/도메인 주소/privkey.pem

verbose
```

turnadmin 사용자 계정 추가

```
turnadmin -a -u 계정이름 -p 계정패스워드 -r 릴름명
```

3478 포트 허용

```
sudo ufw allow 3478/tcp
sudo ufw allow 3478/udp
```

turn 서버 실행

```
sudo service coturn start
```

5. Docker 설치

```
sudo apt-get update
sudo apt-get upgrade -y
sudo apt-get install -y ca-certificates curl gnupg lsb-release
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearm
echo   "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/d
$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/
sudo apt-get update
sudo apt-get install -y docker-ce docker-ce-cli containerd.io docker-buildx-pl
docker --version
sudo usermod -aG docker $USER
```

6. Jenkins 설치

jenkins container 생성 및 구동

```
cd /home/ubuntu && mkdir jenkins-data

sudo ufw allow 8080/tcp
sudo ufw reload
sudo ufw status

sudo docker run -d -p 8080:8080 -v /home/ubuntu/jenkins-data:/var/jenkins_

sudo docker logs jenkins

sudo docker stop jenkins
sudo docker ps -a
```

환경 설정 변경

```
cd /home/ubuntu/jenkins-data

mkdir update-center-rootCAs
```

```
wget https://cdn.jsdelivr.net/gh/lework/jenkins-update-center/rootCA/update-

sudo sed -i 's#https://updates.jenkins.io/update-center.json#https://raw.githu

sudo docker restart jenkins
```

config 보안 설정 확인

```
vi /home/ubuntu/jenkins-data/config.xml



 <useSecurity>true</useSecurity>
  ...(중략)
 <securityRealm class="hudson.security.HudsonPrivateSecurityRealm">
    <disableSignup>true</disableSignup>
```

젠킨스 내 도커 볼륨 연결

```
docker run -d -p 8080:8080 -v /home/ubuntu/jenkins-data:/var/jenkins_ho
me -v /var/run/docker.sock:/var/run/docker.sock --name jenkins jenkins/je
nkins:lts
```

젠킨스 컨테이너 내 docker-cli 설치

```
docker exec -it --user root jenkins bash
```

```
apt-get update
apt-get install -y curl
curl -fsSL https://download.docker.com/linux/debian/gpg | apt-key add -
echo "deb [arch=amd64] https://download.docker.com/linux/debian bookw
orm stable" > /etc/apt/sources.list.d/docker.list
apt-get update
apt-get install -y docker-ce-cli
exit
```

```
docker restart jenkins
```

docker.sock 파일 권한 부여

```
sudo chmod 666 /var/run/docker.sock
```

7. Docker compose 빌드

```
docker compose up --build -d
```