포팅메뉴얼

개발 환경

종류	기술스택	버전
Front	Kotlin	1.8
	Jetpack Compose	
	Compose UI	1.5.4
	Compose Material3	1.1.2
	Retrofit2	2.9.0
Back	Java	11.0.15
	Spring Boot	2.7.17
	JUnit	4.13.1

배포 환경

종류	기술스택	버전
OS	Ubuntu	20.04
DB	MariaDB	11.1.2
	Redis	7.2.3
	MongoDB	7.0.2
CI/CD	Jenkins	2.428

💫 배포 환경 설정

환경 세팅

- **Nginx 설치**

```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get install nginx

cd /etc/nginx/conf.d
sudo vim default.conf
```

`default.conf`

```
upstream backend {
 server k9a502.p.ssafy.io:8080;
}

server {
 listen 80;
 server_name k9a502.p.ssafy.io;
 location /api {
  rewrite ^/api(/.*)$ $1 break;
  proxy_pass http://backend;
  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;
 }

    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/k9a502.p.ssafy.io/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/k9a502.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

}
```

- **Docker 설치**

```
sudo apt-get -y install apt-transport-https ca-certificates curl gnupg-agent software-properties-common
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
sudo apt-get update && sudo apt-get install docker-ce docker-ce-cli containerd.io

sudo usermod -aG docker ubuntu
sudo service docker restart
```

- **Docker-compose 설치**

```
sudo apt install jq
DCVERSION=$(curl --silent https://api.github.com/repos/docker/compose/releases/latest | jq .name -r)
DCDESTINATION=/usr/bin/docker-compose
sudo curl -L https://github.com/docker/compose/releases/download/${DCVERSION}/docker-compose-$(uname -s)-$(uname -m) -o $DCDESTINATION
sudo chmod 755 $DCDESTINATION
```

- **MariaDB 설치 및 세팅**

```
docker run -d \
--name db \
-p 3306:3306 \
-e MYSQL_ROOT_PASSWORD=root \
-e MYSQL_USER=fin \
-e MYSQL_PASSWORD=fin \
-e MYSQL_DATABASE=finale \
-v /app/mariadb:/var/lib/mysql \
-e MYSQL_CHARACTER_SET_SERVER=utf8mb4 \
-e MYSQL_COLLATION_SERVER=utf8mb4_unicode_ci \
-e lower_case_table_names=1 \
-e default-time-zone='+9:00' \
-e max_allowed_packet=128M
mariadb:latest

docker exec -it db mariadb -u fin -p
fin

create database finale;
```

- **MongoDB 설치 및 세팅**

```
sudo docker run -d --name mongodb -p 27017:27017 \
-v ~/dockerdata:/data/db \
-e MONGO_INITDB_ROOT_USERNAME=fin \
-e MONGO_INITDB_ROOT_PASSWORD=fin \
mongo

docker exec -it mongodb mongosh -u fin -p
fin

use finale;

db.createCollection("member_stat");
```

- **Java 11 설치**

```
sudo apt-get install openjdk-11-jdk
```

💫 CI/CD 세팅

Jenkins 설치

- **가상 메모리 설정**

```
df -h # 용량 할당
sudo fallocate -l 8G /swapfile # Swap 영역 할당 (일반적으로 서버 메모리의 2배)
sudo chmod 600 /swapfile # Swapfile 권한 수정
sudo mkswap /swapfile # Swapfile 생성
sudo swapon /swapfile # Swapfile 활성화
free -h # swap 영역이 할당 되었는지 확인
```

- **Jenkins 설치**

```
wget -q -O - https://pkg.jenkins.io/debian/jenkins-ci.org.key | sudo apt-key add -
echo deb http://pkg.jenkins.io/debian-stable binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys FCEF32E745F2C3D5  // 키 등록
sudo apt-get update
sudo apt-get install jenkins
sudo cat /var/lib/jenkins/secrets/initialAdminPassword  // 초기 비밀번호
```

- **초기 설정 플러그인 및 Jenkins Pipeline, Gitlab 관련 플러그인 설치**
- **Jenkins 관리 → Credentials → new credentials**
- **Kind에 Username with password 선택**
- **GitLab 계정을 바탕으로 Username, Password 작성, ID는 임의 ex) mintae로 설정**

Jenkins 구성

1. 자동 빌드 설정하기

1. **프로젝트 아이템 생성 (빈 프로젝트)**
2. **Pipeline 선택 후 생성**
3. **Build Triggers - Build when a change is pushed to GitLab webhook 선택**
4. **Enable GItLab triggers의 push Events, Approved Merge Requests 선택**
5. **Pipeline의 Pipeline script 설정**

```
pipeline {
    agent any
    environment {
        CREDENTIAL_ID = 'mintae'
        SOURCE_CODE_URL = 'https://lab.ssafy.com/s09-final/S09P31A502.git'
        RELEASE_BRANCH = 'develop'
    }
    stages {
        stage('clone repo') {
            steps {
                git url: "$SOURCE_CODE_URL",
                branch: "$RELEASE_BRANCH",
                credentialsId: "$CREDENTIAL_ID"
                sh "ls -al"
            }
        }
        stage('set backend environment'){
            steps{
                dir("./server"){
                    sh '''
                    cp /app/config/.env /var/lib/jenkins/workspace/backend/server/.env
                    cp /app/config/data.sql /var/lib/jenkins/workspace/backend/server/data.sql
                    cp /app/config/Dockerfile /var/lib/jenkins/workspace/backend/server/Dockerfile
```

```
                    cp /app/config/docker-compose.yml /var/lib/jenkins/workspace/backend/docker-compose.yml'''
                    sh "chmod +x ./gradlew"
                    sh "./gradlew clean"
                    sh "./gradlew build -x test"
                }
            }
        }
        stage('down container') {
            steps {
                dir("./"){
                    sh "docker-compose -f docker-compose.yml down --rmi all"
                }
            }
        }
        stage('build docker') {
            steps {
                dir("./"){
                    sh "docker-compose -f docker-compose.yml build --no-cache"
                }
            }
        }
        stage('up container') {
            steps {
                dir("./"){
                    sh "docker-compose -f docker-compose.yml up -d"
                }
            }
        }
    }
}
```

6. ⚠️ **git clone 할 시 docker-compose, Dockerfile, .env, data.sql 등이 없기 때문에, 배포 서버 `/app/config`경로에 세팅 필요**

환경 변수 및 프로퍼티 파일

`docker-compose.yml`

```
version: "3.8"
services:
  backend:
    env_file:
      - "./server/.env"
    environment:
      - "SPRING_PROFILES_ACTIVE=prod"
      - "TZ=Asia/Seoul"
    build:
      context: ./server
      dockerfile: Dockerfile
    restart: always
    ports:
      - 8080:8080
    container_name: backend
    networks:
      - deploy

  redis:
    image: redis:latest
    container_name: redis
    hostname: redis
    command: redis-server --requirepass fin --port 6379
    volumes:
      - redis-volume:/data
    environment:
      - REDIS_PASSWORD=fin
      - "TZ=Asia/Seoul"
    ports:
      - 6379:6379
    restart: always
    networks:
      - deploy

networks:
  deploy:
    external: true

volumes:
  redis-volume:
```

`Dockerfile`

```
FROM gradle:8.3-jdk11
ENV USE_PROFILE prod
COPY ./build/libs/neulhaerang-0.0.1-SNAPSHOT.jar neulhaerang.jar
ENTRYPOINT ["java","-Dspring.profiles.active=${USE_PROFILE}", "-jar", "neulhaerang.jar"]
```

`.env`

```
DB_HOST=k9a502.p.ssafy.io
DB_PORT=3306
DB_NAME=finale
DB_USER=fin
DB_PASSWORD=fin
REDIS_HOST=k9a502.p.ssafy.io
REDIS_PORT=6379
REDIS_PASSWORD=fin
MONGO_HOST=k9a502.p.ssafy.io
MONGO_USER=fin
MONGO_PASSWORD=fin
MONGO_DATABASE=finale
JWT_KEY=${설정할 jwt key}
GPT_KEY=${발급받은 gpt key}
FCM_SECRET_KEY=${발급받은 firebase key의 경로}
```

💫 서비스 실행

!! 위 단계에서 환경 설정을 이행 후 진행

- ☑️ ~~Docker~~
- ☑️ ~~docker compose~~
- ☑️ ~~MariaDB~~
- ☑️ ~~MongoDB~~

Local에서 백엔드(Neulhaerang) & Redis 실행

1. `git clone https://lab.ssafy.com/s09-final/S09P31A502.git` 이후 루트 경로에 `docker-compose.yml` 생성
2. `server` 디렉토리 내에 `Dockerfile` 생성
3. `server` 디렉토리 내에 `.env` 생성
4. `server/src/main/resources/application.yml` 수정

   ```
   spring:
     jpa:
       hibernate:
         ddl-auto: create    // 로 변경
     sql:
       init:
         mode: always  // 로 변경
   ```

5. 실행 빌드 파일 생성
6. 루트 경로에서 `docker-compose up`

PlayStore에서 필수 앱 다운로드

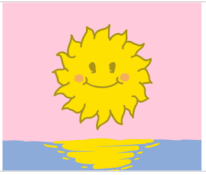
늘해랑 앱 다운로드

https://play.google.com/store/apps/details?id=com.finale.neulhaerang&hl=ko-KR

늘해랑 - Google Play 앱

스탯 성장을 통해 한눈에 관리할 수 있는 투두리스트

https://play.google.com/store/apps/details?id=com.finale.neulhaerang&hl=ko-KR



삼성 헬스 앱 다운로드

https://play.google.com/store/apps/details?id=com.sec.android.app.shealth&hl=ko-KR

Samsung Health(삼성 헬스) - Google Play 앱

각종 운동과 식단관리, 수면습관까지! 매일의 목표와 다양한 조언을 통해 더 건강한 내일에 도전하세요!

https://play.google.com/store/apps/details?id=com.sec.android.app.shealth&hl=ko-KR

헬스 커넥트 앱 다운로드

https://play.google.com/store/apps/details?id=com.google.android.apps.healthdata&hl=ko-KR

헬스 커넥트(베타) - Google Play 앱

건강, 피트니스, 웰빙 앱 간에 간편하게 데이터를 공유하세요.

https://play.google.com/store/apps/details?id=com.google.android.apps.healthdata&hl=ko-KR

헬스 커넥트 & 삼성 헬스 연동

1. 삼성 헬스 앱 접속 후 삼성 계정 로그인
2. 헬스 커넥트를 다운 받은 상태로 `설정 > 헬스 커넥트 > 권한 허용`
3. 늘해랑 앱 접속 후 권한 허용