Porting Menual

개발환경

Deploy Server

- Ubuntu 20:04:LTS

Front-end

- Vue
  - vue : 2.6.11
  - vue/cli : 4.5.0

IDE

- IntelliJ : 2019.3.5
- Visual Studio Code : 1.59.0
- MySqlWorkbench : 8.0.21

DB

- mysql : 8.0.25

Back-end

- java : openjdk version "1.8.0_262"
- Spring
  - spring boot 2.5.6
  - Gradle 7.2
- npm : 6.14.13
- docker
  - docker : 20.10.9
  - docker-compose version 1.28.0-rc2

배포방법

> 먼저 배포할 서버에 접속해 있다고 가정하여 진행하겠습니다.
> 설명할 서버 환경은 Ubuntu 20.04:LTS 기준입니다.

Java 8 버전 설치

1. Azul의 public key 추가

```
sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys 0x219BD9C9
```

2. Azul Repository 추가

```
sudo apt-add-repository 'deb http://repos.azulsystems.com/ubuntu stable main'
```

3. apt-get 업데이트

```
apt-get update
```

4. zulu-8 설치

```
apt-get install zulu-8
```

5. 환경변수 설정

```
# /etc/profile

export JAVA_HOME=/usr/lib/jvm/zulu-8-amd64
```

6. 자바 설치확인

```
java -version
```

Docker-CE설치

1. apt 라이브러리 update 하기

```
sudo apt update
```

2. 다운로드를 위한 util 설치

```
sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    gnupg-agent \
    software-properties-common
```

3. Docker 공식 GPG 키 추가하기

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

4. apt 리스트에 도커 다운경로 추가하기

```
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable"
```

5. 도커 다운로드

```
apt-cache policy docker-ce
```

6. 도커 설치하기

```
sudo apt install docker-ce

# apt 패키지 인덱스 업데이트하기
sudo apt-get update
```

7. sudo 없이 도커를 사용하기 위해 사용자를 docker 그룹에 등록

```
sudo usermod -aG docker [현재 사용자]

# 이를 반영하기 위한 시스템 재부팅
sudo reboot
```

8. docker-ce를 설치했다면 이제 docker-compose도 설치해주자

```
sudo curl -L https://github.com/docker/compose/releases/download/1.28.0-rc2/docker-compose-`uname -s`-`uname -m` -o /usr/local/bin/dock
```

9. docker-compose도 실행권한을 주자 (다른 방식 연습!)

```
sudo chmod +x /usr/local/bin/docker-compose
```

Mysql 설치

1. Mysql 설치

```
sudo apt-get update
sudo apt-get install mysql-server
```

2. ubunut mysql은 대소문자 구별이 default로 설정되어 있음.
   - 이를 해결하려면 아래의 과정을 수행한다.
   1. Mysql 서비스 중지

      ```
      sudo service mysql stop
      ```

   2. Mysql 데이터 디렉토리 삭제

      ```
      sudo rm -rf /var/lib/mysql
      ```

   3. Mysql 데이터 디렉토리 재생성

      ```
      sudo mkdir /var/lib/mysql
      sudo chown mysql:mysql /var/lib/mysql
      sudo chmod 700 /var/lib/mysql
      ```

   4. lower_case_table_names = 1를 `[mysqld]` 섹션에 추가

      ```
      # /etc/mysql/mysql.conf.d/mysqld.conf
      [mysqld]
      ...

      lower_case_table_names = 1
      ```

   5. Mysql 초기화

      ```
      sudo mysqld --defaults-file=/etc/mysql/my.cnf --initialize --lower_case_table_names=1 --user=mysql --console
      ```

   6. Mysql 서비스 시작

      ```
      sudo service mysql start
      ```

   7. 새로 생성된 root의 비밀번호 검색

      ```
      sudo grep 'temporary password' /var/log/mysql/error.log
      ```

   8. root로 접속

```
sudo mysql -u root -p
```

9. 비밀번호 변경

```
ALTER USER 'root'@'localhost' IDENTIFIED BY '1234';
```

10. lower_case_table_names을 해서 1 출력 확인

```
SHOW VARIABLES LIKE 'lower_case_%';
```

3. AWS 설정에서 보안그룹 설정을 통해 mysql 기본 포트(3306)을 열어주어야 한다.
4. 외부접속 허용하기 ⇒ mysqld.cnf 수정

```
# /etc/mysql/mysql.conf.d/mysqld.cnf
sudo vi /etc/mysql/mysql.conf.d/mysqld.cnf
```

```
# bind-address를 찾아서 0.0.0.0으로 수정
bind-address = 0.0.0.0
```

5. mysql 재시작 및 접속

```
sudo service mysql restart

sudo mysql -u root -p
```

6. 외부접속 권한을 가진 유저생성

```
CREATE USER 'root'@'%' IDENTIFIED BY '1234';

GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' WITH GRANT OPTION;

FLUSH PRIVILEGES;
```

- MYSQL 8.0부터 Grant 명령어를 사용하여 묵시적으로 사용자를 생성할 수 없다.
  따라서 '%' : 원격접속을 할 수 있는 사용자(여기서는 root)를 생성해 주어야한다.

소스코드 Git clone 받기

```
git clone https://lab.ssafy.com/s05-final/S05P31F005.git
```

Script 실행

```
cd S05P31F005
bash start.sh
```

- Docker를 사용하여 build및 실행시키는 단계입니다.
- start.sh 쉘 스크립트안에 명령어가 내장되어 있으며, 총 2개의 컨테이너를 데몬으로 실행시키게 됩니다.

- front: reverse proxy를 담당하는 nginx + vue코드
- server : Spring Boot Application

Docker

```
# start.sh

# gradlew 실행권한 부여 및 빌드
cd backend && chmod 777 ./gradlew && ./gradlew bootJar && cd ..
# 도커이미지 pull
docker-compose pull

# 도커컴포즈 빌드 및 데몬 실행
COMPOSE_DOCKER_CLI_BUILD=1 DOCKER_BUILDKIT=1 docker-compose up --build -d

# remove unused docker images
unused=$(docker images -f "dangling=true" -q)
if [ ${#unused} != 0 ]
then
    docker rmi $unused -f

fi
exit 0
```

```
version: '3'
services:
  server:
    container_name: server
    build:
      context: ./backend
    ports:
      - "8080:8080"

  front:
    container_name: front
    build:
      context: ./frontend
    restart: always
    depends_on:
      - server
    ports:
      - "80:80"
```

- nginx 설정파일

  경로 : S05P31F005/frontend/.nginx/nginx.conf

```
server {
  listen 80;
  client_max_body_size 5M;
  location /api {
      proxy_set_header Host $host;
      proxy_set_header X-Real-IP $remote_addr;
      proxy_pass http://server:8080;
  }

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

배포시 주의사항

1. 현재 모든 경로가 하드코딩되어 있으므로 도메인을 변경 혹은 localhost로 실행할 경우, 하드코딩된 경로수정필요
   - 수정할 문자열
     - k5f001.p.ssafy.io → localhost
   - applicaion.yaml 수정

```
# 디비수정
spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://k5f001.p.ssafy.io:3306/ssurbar?serverTimezone=UTC&characterEncoding=UTF-8
    username: root
    password: 1234
```

   - frontend/src/utils/axios.js 수정

```
import axios from "axios";
import store from "@/store/index.js";

// axios 객체 생성
const _axios = axios.create({
  baseURL: "http://k5f001.p.ssafy.io:8080/api/v1",
  headers: {
    "Content-type": "application/json",
  },
  withCredentials: true,
});
```

데이터베이스 접속정보

- 경로 : S05P31F005/backend/src/main/resources/application.yaml

Mysql

- host : k5f001.p.ssafy.io
- port : 3306
- user : root
- password : 1234