포팅메뉴얼

1. 빌드 및 배포
    설정값과 버전
    환경변수
    application.yml
    빌드 & 배포 순서, 특이사항
우분투에 docker 설치
    1. 우분투 시스템 패키지 업데이트
    2. 필요한 패키지 설치
    3. Docker의 공식 GPG키를 추가
    4. Docker의 공식 apt 저장소를 추가
    5. 시스템 패키지 업데이트
    6. Docker 설치
MYSQL 설치
    1. MySQL 이미지를 pull 하기
    2. Docker 컨테이너 볼륨 설정
    volume 확인
    mysql 컨테이너 실행 (볼륨마운트)
    3. MYSQL 컨테이너 bash 접속하기
    4. MYSQL 서버에 접속하기
    5. 새로운 계정 만들기
    권한 설정
    6. 생성한 user로 MYSQL 서버에 접속한다.
    7. 데이터 베이스 생성
    8. 데이터 베이스 보기
    9. IP 접속가능 범위 보기 (중요💥 db한번털림)
    10. root 사용자 접속권한 삭제
    11. MYSQL 연결 후 데이터 연결하기

Git Clone 하기
    1. git clone 하기
BackEnd 빌드
    1. 자바설치
    2. gradle 설치
    3. gradle 실행
    4. Spring 빌드하기
    5. Dockerfile로 빌드하기
    6. BackEnd 도커 이미지 실행
    1. 컨테이너 존재 시 컨테이너 종료 후 삭제I
    2. clone한 폴더에서 Data 폴더로 이동
    4. 이미지 확인(선택사항)
    5. 컨테이너 생성
    1. 컨테이너 존재 시 컨테이너 종료 후 삭제
    2. 이미지 존재 시 삭제
    3. clone한 폴더에서 Data 폴더로 이동
    4. 필요한 패키지 import 한 뒤, build
    5. 이미지 생성
    6. 컨테이너 생성
Nginx 설치 및 설정
    1. Nginx 설치
    2. Nginx 실행
    3. SSL 설정
    4. certbot을 사용해서 ssl 설정하기
    5. NGINX 설정하기
    Nginx 파일설정
    2. DB 덤프 파일

1. 빌드 및 배포

설정값과 버전

1. 프로젝트 사용 도구

- 이슈 관리 : JIRA
- 형상 관리 : Gitlab
- 커뮤니케이션 : Notion, Mattermost, Discode
- 디자인 : Figma
- UCC : 모바비, 애프터이펙트, 프리미어
- CI/CD : Jenkins

2. 개발 환경

- Vue.js
- TypeScript
- Java 11
- SpringBoot 2.7.14
- MySQL
- Docker
- intelliJ
- VScode
- Nginx
- Jenkins
- Node.js 18.16.1
- Bootstrap

환경변수

application.yml

```
jwt:
  secret: 47b7172f9748fa42759cbaa4efbcdfb4d7b5edd67a4d10b36fb95a7dca962591597a2ef42cb4ea07950b404748c0b3707c4754981214ba3c471bf3cb18b8

spring:
  security:
    user:
      name: user@gmail.com
      password: 1234

  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://i9B202.p.ssafy.io:2231/test?serverTimezone=UTC&characterEncoding=UTF-8
    username: npdp
    password: ffc704cc-3c03-4b47-b239-2ffd84e593ff

  jpa:
    database: mysql
    database-platform: org.hibernate.dialect.MySQL8Dialect
    show-sql: true
    hibernate:
      ddl-auto: update
    properties:
      hibernate:
        format_sql: true
        default_batch_fetch_size: 1000

  mail:
    default-encoding: UTF-8
    host: smtp.gmail.com    # SMTP 서버 호스트
    port: 587               # SMTP 서버 포트
    username: ssafyidle@gmail.com    # SMTP 서버 아이디
    password: daktgkytdyaxvtna       # SMTP 서버 앱 패스워드
#    protocol: smtps
    properties:
      mail:
        smtp:
```

```
          auth: true      # 사용자 인증 시도 여부 (기본값: false)
          timeout: 5000   # Socket Read Timeout 시간 (기본값: 무한대)
          starttls:
            enable: true  # StartTLS 활성화 여부 (기본값: false)
  auth-code-expiration-millis: 600000  # 10 * 60 * 1000 == 10분

  redis:
    host: i9b202.p.ssafy.io
    port: 6379

  profiles:
    active: oauth

logging:
  level:
    org.hibernate: INFO
    org.hibernate.type: INFO
    com.project.npdp.snslogin: DEBUG
```

빌드 & 배포 순서, 특이사항

우분투에 docker 설치

1. 우분투 시스템 패키지 업데이트

```
sudo apt-get update
```

2. 필요한 패키지 설치

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

3. Docker의 공식 GPG키를 추가

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

4. Docker의 공식 apt 저장소를 추가

```
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
```

5. 시스템 패키지 업데이트

```
sudo apt-get update
```

6. Docker 설치

```
sudo apt-get install docker-ce docker-ce-cli containerd.io
```

MYSQL 설치

1. MySQL 이미지를 pull 하기

```
sudo docker mysql:latest
```

2. Docker 컨테이너 볼륨 설정

```
sudo docker volume create mysql-volume
```

volume 확인

```
sudo docker volume ls
```

mysql 컨테이너 실행 (볼륨마운트)

```
sudo docker run -d --name mysql-container -p 2231:3306 -v mysql-volume:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=1234 mysql:latest
```

3. MYSQL 컨테이너 bash 접속하기

```
sudo docker exec -it mysql-container bash
```

4. MYSQL 서버에 접속하기

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

5. 새로운 계정 만들기

```
mysql> CREATE USER npdp@'%' identified by '1234';
```

npdp는 계정id
%는 모든 IP에서 접속가능
1234는 비밀번호

권한 설정

```
mysql> GRANT ALL PRIVILEGES ON *.* to npdp@'%';
mysql> FLUSH PRIVILEGES;
mysql> exit;
```

6. 생성한 user로 MYSQL 서버에 접속한다.

```
bash# mysql -u npdp -p
1234
```

7. 데이터 베이스 생성

```
mysql> CREATE DATABASE test;mysql> SHOW DATABASES;
```

8. 데이터 베이스 보기

```
mysql> SHOW DATABASES;
```

9. IP 접속가능 범위 보기 (중요 db한번털림)

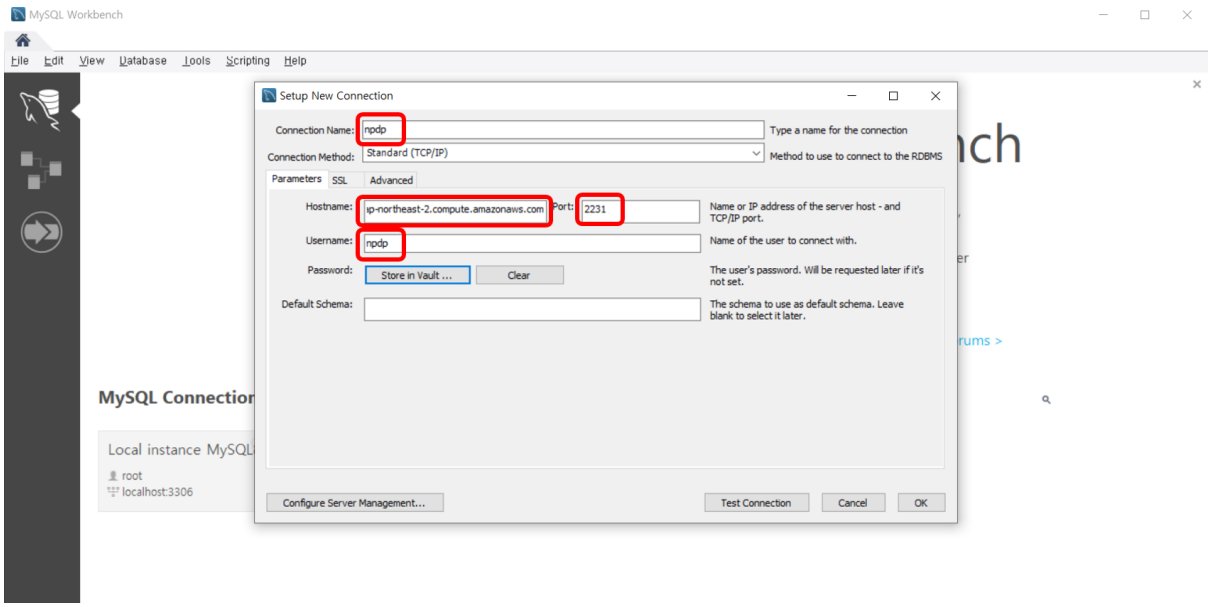
```
mysql> SELECT user, host FROM mysql.user;
```

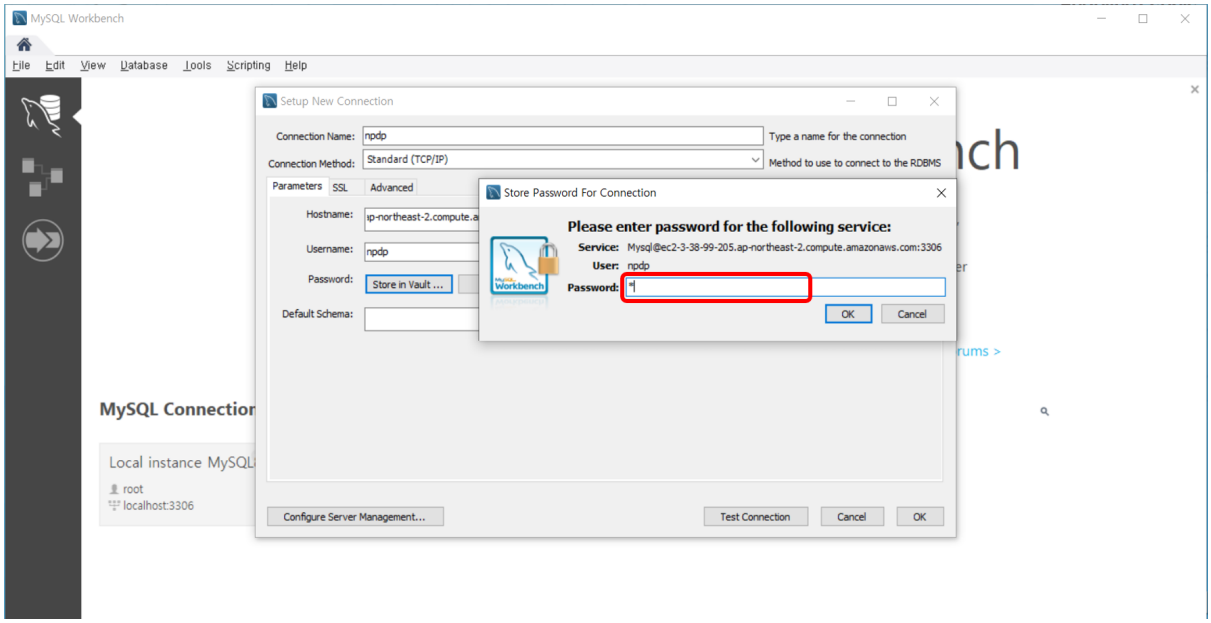
10. root 사용자 접속권한 삭제

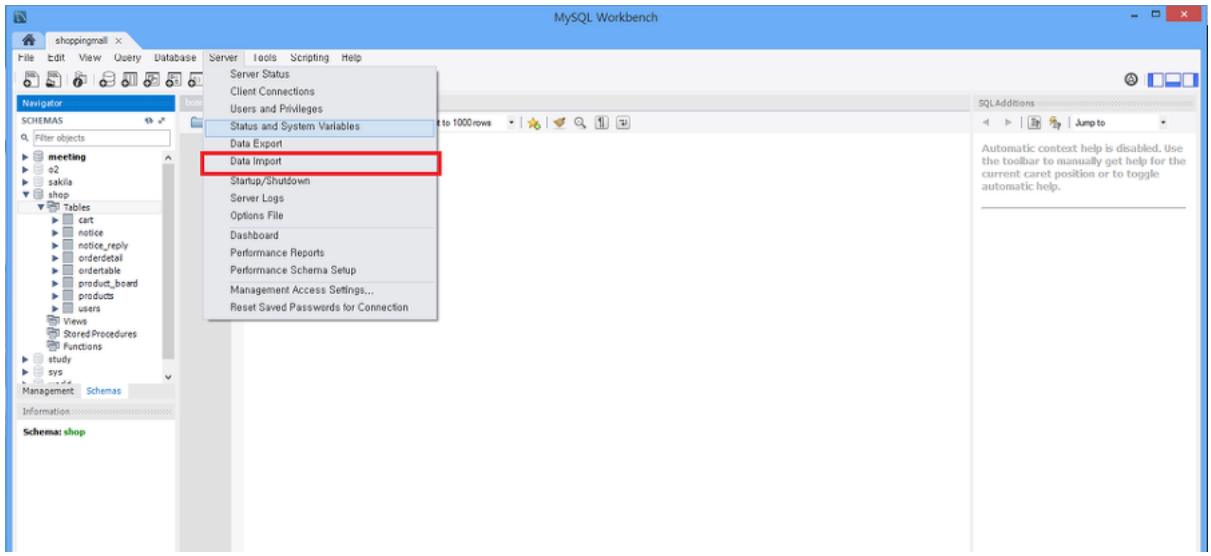
```
mysql> DELETE FROM mysql.user WHERE User='root' AND Host='%';
mysql> FLUSH PRIVILEGES;
```

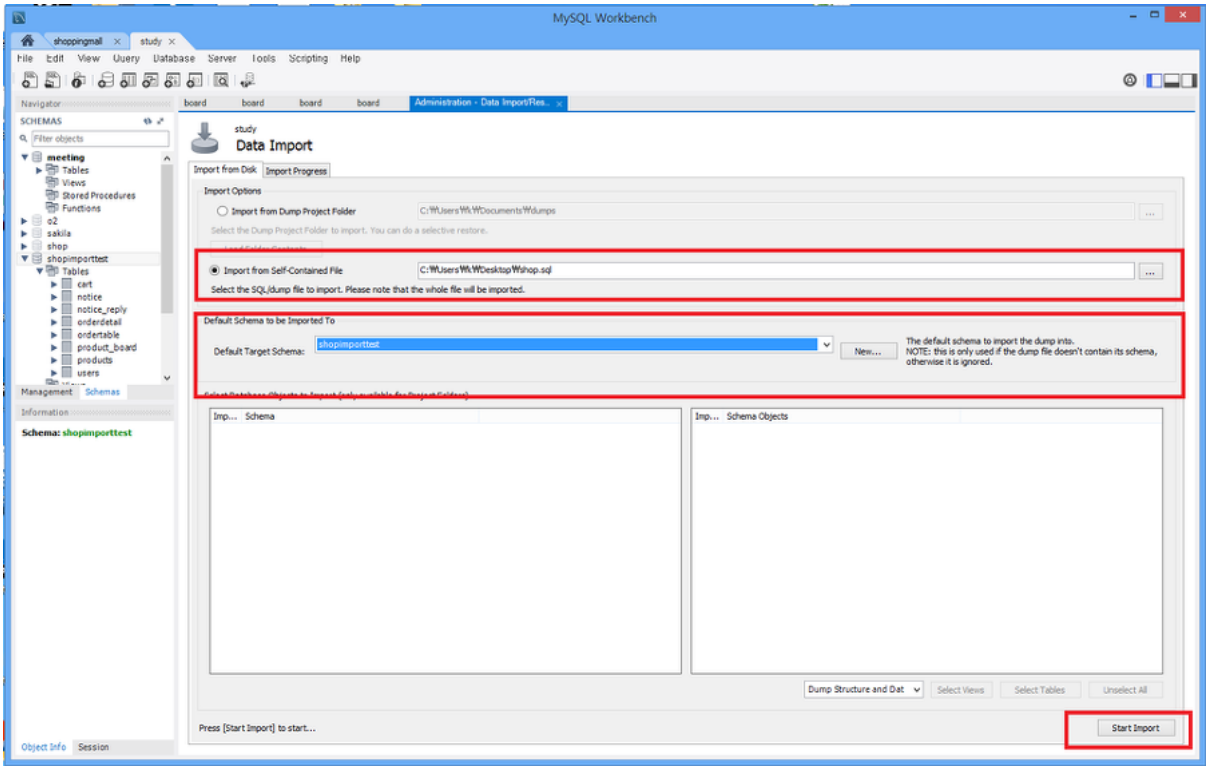
exit 두번해서 ubuntu로 돌아가기

11. MYSQL 연결 후 데이터 연결하기




MySQL Workbench
File Edit View Database Tools Scripting Help
Setup New Connection
Connection Name: npdp
Type a name for the connection
Connection Method: Standard (TCP/IP)
Method to use to connect to the RDBMS
Parameters SSL Advanced
Hostname:
Username: npdp
Password: Store in Vault ...
Default Schema:
Store Password For Connection
Please enter password for the following service:
Service: Mysql@ec2-3-38-99-205.ap-northeast-2.compute.amazonaws.com:3306
User: npdp
Password:
OK
Cancel
Configure Server Management...
Test Connection
Cancel
OK
MySQL Connectior
Local instance MySQL
root
localhost:3306


MySQL Workbench
shoppingmall
File Edit View Query Database Server Tools Scripting Help
Server Status
Client Connections
Users and Privileges
Status and System Variables
Data Export
Data Import
Startup/Shutdown
Server Logs
Options File
Dashboard
Performance Reports
Performance Schema Setup
Management Access Settings...
Reset Saved Passwords for Connection
Navigator
SCHEMAS
meeting
o2
sakila
shop
Tables
cart
notice
notice_reply
orderdetail
ordertable
product_board
products
users
Views
Stored Procedures
Functions
study
sys
Management Schemas
Information
Schema: shop
SQLAdditions
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.



Git Clone 하기

1. git clone 하기

```
git clone git주소
```

아이디, 비밀번호 입력하면 클론 완료

BackEnd 빌드

1. 자바설치

```
sudo apt update
sudo apt install openjdk-11-jdk
java -version
```

2. gradle 설치

```
/S09P12B202/Backend/NangPaDaePa 으로 이동 후

sh gradlew
```

3. gradle 실행

```
sh gradlew build
```

이렇게 하면 jar 파일이 생김

4. Spring 빌드하기

```
cd build
cd libs

후
java -jar NangPaDaePa-0.0.1-SNAPSHOT.jar

빌드가 정상적으로 되면
Ctrl + C 로 빌드 취소하기
```

5. Dockerfile로 빌드하기

```
cd ..
cd ..
S09P12B202/Backend/NangPaDaePa$ 에서

sudo docker build -t nangpadaepa .
(마지막에 . 찍고 띄어쓰기 필수)
```

6. BackEnd 도커 이미지 실행

```
sudo docker run -d --name nangpadaepa-back -p 8080:8080 nangpadaepa
```

`Back (Django)`

1. 컨테이너 존재 시 컨테이너 종료 후 삭제I

```
docker stop my-django-app
docker rm -f my-django-app
```

2. clone한 폴더에서 Data 폴더로 이동

```
cd Data
```

3. docker file로 이미지 생성

```
sudo docker build -t my-django-image -f Dockerfile .
```

4. 이미지 확인(선택사항)

```
sudo docker images
```

5. 컨테이너 생성

```
sudo docker run -d --name nangpadaepa-data -p 8000:8000 my-django-image
```

`Front (Vue)`

1. 컨테이너 존재 시 컨테이너 종료 후 삭제

```
sudo docker stop my_vue
sudo docker rm -f my_vue
```

2. 이미지 존재 시 삭제

```
sudo docker image rm vue_app
```

3. clone한 폴더에서 Data 폴더로 이동

```
cd my-vue-project
cd Front
cd front_cli
```

4. 필요한 패키지 import 한 뒤, build

```
npm install
npm run build
```

5. 이미지 생성

```
sudo docker build -t vue_app
```

6. 컨테이너 생성

```
sudo docker run -d --name nangpadaepa-front -p 3030:80 vue_app
```

Nginx 설치 및 설정

1. Nginx 설치

```
sudo apt install nginx
```

2. Nginx 실행

```
sudo service nginx start
```

3. SSL 설정

```
# snap을 이용하여 core 설치 -> snap을 최신 버전으로 유지하기 위해 설치
$ sudo snap install core

# core를 refresh 해준다.
$ sudo snap refresh core
```

```
# 기존에 잘못된 certbot이 설치되어있을 수도 있으니 삭제 해준다.
$ sudo apt remove certbot

# certbot 설치
$ sudo snap install --classic certbot

# certbot 명령을 로컬에서 실행할 수 있도록 snap의 certbot 파일을 로컬의 cerbot과 링크(연결) 시켜준다. -s 옵션은 심볼릭링크를 하겠다는 것.
$ sudo ln -s /snap/bin/certbot /usr/bin/certbot
```

4. certbot을 사용해서 ssl 설정하기

```
sudo certbot --nginx
```

5. NGINX 설정하기

```
sudo vi /etc/nginx/sites-enabled/default


설정 끝나면
sudo nginx -t
sudo service nginx restart
```

Nginx 파일설정

```
server {
    listen 80;
    server_name i9b202.p.ssafy.io;
    return 301 https://$host$request_uri;
}

# HTTPS 설정
server {
    listen 443 ssl;
    server_name i9b202.p.ssafy.io;

    ssl_certificate /etc/letsencrypt/live/i9b202.p.ssafy.io/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/i9b202.p.ssafy.io/privkey.pem;

    # Vue 앱에 대한 프록시 설정
    location / {
        proxy_pass http://127.0.0.1:3030;
        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;
    }

    # API에 대한 프록시 설정
    location /api {
        proxy_pass http://127.0.0.1:8080;
        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;
    }

    # django에 대한 프록시 설정
    location /data{
            proxy_pass http://127.0.0.1:8000;
            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;
    }
}
                                                                                290,9         98%
```

계정

2. DB 덤프 파일

Dump data.zip