

# Porting Manual

## 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
- redis-server : 5.0.7

### Back-end

- java : openjdk version "1.8.0\_262"
- Spring
  - spring boot 2.5.2
  - gradle-6.8.3
- npm : 6.14.13
- docker
  - docker : 20.10.7
  - docker-compose version 1.28.0-rc

## 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. ubuntu 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.ssfy.com/s05-webmobile2-sub3/S05P13A602.git
```

## SSL 인증서 발급

### SSL For Free

#### 1. SSL 발급받을 도메인 입력하기

<https://www.sslforfree.com/> ← 접속

# SSL 무료

몇 분 만에 무료 SSL 인증서 및 와일드카드 SSL 인증서

 보안 | https:// naver.com

무료 SSL 인증서 생성



## 영원히 100% 무료

다시는 SSL 비용을 지불하지 마십시오. 무료 90일 인증서와 함께 ZeroSSL로 구동됩니다.



## 널리 신뢰받는

당사의 무료 SSL 인증서는 전 세계 모든 주요 브라우저의 99.9%에서 신뢰됩니다.



## SSL 혜택 즐기기

사용자 정보를 보호하고 신뢰를 구축하며 검색 엔진 순위를 향상시킵니다.

## 2. 회원가입

# 가입하기

SSL 인증서를 만들고 관리하려면 무료 계정에 가입하세요.

도메인:

이메일 주소

xxxxxxx@gmail.com

비밀번호

.....

등록하다

중요한 서비스 업데이트를 수신하는 데 동의합니다.

로그인

### 3. 인증서 신청

#### a. 도메인 입력

# New Certificate

Cancel

## SSL Certificate Setup

You're on your way to issuing a brand-new SSL certificate for one or multiple domains. Before you can install your new certificate, please complete the steps below.

✓

Domains

☐

I need a wildcard certificate

PRO

Please enter at least one domain to secure. For single-domain certificates the WWW-version of your domain will always be included at no extra charge.

Enter Domains

🔒

✓

example.com

+

Add Domain

PRO

Next Step →

b. 인증 날짜 90일 설정

# New Certificate

Cancel

## SSL Certificate Setup

You're on your way to issuing a brand-new SSL certificate for one or multiple domains. Before you can install your new certificate, please complete the steps below.

✓

Domains

✓

Validity

You can now choose between generating 90-day or one-year certificate validity. To keep manual work at a minimum, we recommend 1-year certificates.

🔒

90-Day Certificate

☐

1-Year Certificate

PRO

Next Step →

c. Free 선택 후 Next Step

Porting Manual

7

## New Certificate

Cancel

### SSL Certificate Setup

You're on your way to issuing a brand-new SSL certificate for one or multiple domains. Before you can install your new certificate, please complete the steps below.





✓ Domains

✓ Validity

✓ CSR & Contact

#### Finalize Your Order

Based on your selection of a 90-Day SSL Certificate you are fine staying on the Free Plan. To create and validate your SSL Certificate, please click "Next Step" below.

 <b>Free</b> \$0 / month	 <b>Basic</b> \$10 / month or \$8 if billed yearly	 <b>Premium</b> \$50 / month or \$40 if billed yearly	 <b>Business</b> \$100 / month or \$80 if billed yearly
<b>Selected</b>	Select	Select	Select
3 90-Day Certificates	∞ 90-Day Certificates	∞ 90-Day Certificates	∞ 90-Day Certificates
✕ 1-Year Certificates	3 1-Year Certificates	10 1-Year Certificates	25 1-Year Certificates
✕ Multi-Domain Certs	✓ Multi-Domain Certs	✓ Multi-Domain Certs	✓ Multi-Domain Certs
✕ 90-Day Wildcards	✕ 90-Day Wildcards	∞ 90-Day Wildcards	∞ 90-Day Wildcards
✕ 1-Year Wildcards	✕ 1-Year Wildcards	1 1-Year Wildcards	3 1-Year Wildcards
✕ REST API Access	✓ REST API Access	✓ REST API Access	✓ REST API Access
✕ Technical Support	✓ Technical Support	✓ Technical Support	✓ Technical Support

← →

Next Step →

#### 4. 도메인 소유권 인증

이제 해당 도메인이 내가 소유하고 있다는 것을 증명해야한다.

그 이유는 우리가 막 naver같은 도메인에 SSL의 인증서를 발급받는 것을 막기 위함

##### a. 인증 txt 다운로드



## Verify Domain

Verify Later

✓ Your certificate has been created and is ready for domain verification.

i5a602.p.ssafy.io

Congratulations, your SSL certificate is en route! However, you need to verify ownership of your domain before installing your certificate. Please follow the steps below.

### Verification Method for i5a602.p.ssafy.io

We need you to verify ownership of each domain in your certificate. Please select your preferred verification method and click "Next Step".

☐ Email Verification

☐ DNS (CNAME)

☒ HTTP File Upload

[1] 클릭

✓ Follow the steps below

To verify your domain using HTTP File Upload, please follow the steps below:

- 1 Download your Auth File using the following link: [Download Auth File](#)
- 2 Upload the Auth File to your HTTP server under: `/.well-known/pki-validation/`
- 3 Make sure your file is available under the following link: `https://i5a602.p.ssafy.io/.well-known/pki-validation/047018FBD9362BD5496D8D3E205D3C2A.txt`
- 4 Click "Next Step" to continue.

[2] 파일 다운로드 클릭

Next Step →

> Finalize

b. .txt 파일 복사하기

S05P13A602/.nginx/zerossI 폴더 밑에 .txt파일을 복사

c. nginx 실행

```
docker-compose up
```

d. 인증확인

**HTTP File Upload**

✓ Follow the steps below

To verify your domain using **HTTP File Upload**, please follow the steps below:

- 1 Download your Auth File using the following link: [Download Auth File](#)
- 2 Upload the Auth File to your HTTP server under: `/.well-known/pki-validation/`
- 3 Make sure your file is available under the following link: [http://\[redacted\]/.well-known/pki-validation/047018FBD9362BD5496D8D3E205D3C2A.txt](http://[redacted]/.well-known/pki-validation/047018FBD9362BD5496D8D3E205D3C2A.txt)
- 4 Click "Next Step" to continue.

[1] 링크 클릭

[2] 인증서 발급받기

Next Step →

- 링크를 클릭하여 정상적으로 .txt 파일의 내용이 보이는지 확인을 합니다.
- 그리고 인증서 발급창으로 이동합니다.

## 5. 인증서 발급받기

- Verify Domain 버튼을 눌러 도메인을 인증
- 성공한다면, Server Type을 Nginx로 선택한후, 인증서를 다운로드 받습니다.

## 6. 인증서 복사하기

S05P13A602/nginx/cert 폴더에 저장

## Script 실행

```
cd S05P13A602
bash start.sh
```

- Docker를 사용하여 build및 실행시키는 단계입니다.
- start.sh 쉘 스크립트안에 명령어가 내장되어 있으며, 총 3개의 컨테이너를 데몬으로 실행시키게 됩니다.
  - nginx : reverse proxy를 담당하는 nginx
  - client : Vue Application
  - server : Spring Boot Application

## Docker

```
# start.sh

# Spring gradle build
cd commbServer && ./gradlew bootJar && cd ..
# 도커이미지 최신화
docker-compose pull

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

# 더미이미지 삭제
docker rmi $(docker images -f "dangling=true" -q) -f
```

```
#docker-compose.yml

version: '3'
services:
  # 스프링 컨테이너
  server:
    container_name: server
    build:
      context: ./commbServer
      args:
        DOMAIN: i5a602.p.ssafy.io
    ports:
      - "8080:8080"

  # vue 컨테이너
  client:
    container_name: client
    build: ./client
    ports:
      - "8000:8000"
    depends_on:
      - server

  # nginx 컨테이너
  nginx:
    container_name: nginx
    # image: proxy
    build: ./nginx
    depends_on:
      - server
      - client
    volumes:
      - .nginx/conf.d:/etc/nginx/conf.d # 설정정보
      - .nginx/zeross:/var/www/zeross/.well-known/pki-validation # SSL 도메인 인증
      - .nginx/cert:/cert # SSL 인증서 보관장소
    ports:
      - 80:80 # for Http
      - 443:443 # for Https
```

- nginx 설정파일

경로 : S05P13A602/nginx/conf.d/default.conf

```
server {
    listen      80; # 80포트 허용
    server_name i5a602.p.ssafy.io; # 서버이름

    # reverse proxy
    location /api { # /api 요청시 spring으로
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_pass http://server:8080;
    }

    # 이경로로 요청이오면 SSL 도메인 확인 txt 반환
    location /.well-known/pki-validation/ {
        allow all;
        root /var/www/zeross;
    }

    location / { # 나머지는 vue로
        proxy_pass http://client:8000;
    }
}

server {
    listen 443 ssl; # 443 포트허용
    server_name i5a602.p.ssafy.io;

    proxy_connect_timeout 1d;
    proxy_send_timeout 1d;
    proxy_read_timeout 1d;
```

```
# 위와동일
location /api {
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_pass http://server:8080;
}

location / {
    proxy_pass http://client:8080;
}

# SSL 인증설정
ssl_certificate /cert/certificate.crt;
ssl_certificate_key /cert/private.key;
}
```

## 배포시 주의사항

1. 현재 모든 경로가 하드코딩되어 있으므로 도메인을 변경 혹은 localhost로 실행할 경우, 하드코딩된 경로수정필요

- 수정할 문자열
  - i5a602.p.ssafy.io → localhost
  - https → http
- applicaion.yaml 수정

```
# 디비수정
spring:
  datasource:
    url : jdbc:mysql://i5a602.p.ssafy.io/Commb?serverTimezone=Asia/Seoul
    username : root
    password : 1234

# redis 수정
redis:
  host : i5a602.p.ssafy.io
  password : 1234
  port : 6379

# dynamic경로수정
dynamic:
  path: https://i5a602.p.ssafy.io/
  front:
    path: https://i5a602.p.ssafy.io/
```

- .env.production 수정

```
VUE_APP_SERVER_URL=https://i5a602.p.ssafy.io/api
```

2. Https 주의

- FCM(Firebase Cloud Messaging) 서비스를 사용중인데,  
localhost 이외의 환경에서 https가 아닐경우 제대로 실행되지 않는다!
- 따라서, 특정 도메인에 배포할경우 꼭 **SSL 인증서가 필요함**.

## 데이터베이스 접속정보

- 경로 : S05P13A602/commbServer/src/main/resources/application.yaml

## Mysql

- host : [i5a602.p.ssafy.io](https://i5a602.p.ssafy.io)
- port : 3306
- user : root
- password : 1234

## Redis

- host : [i5a602.p.ssafy.io](https://i5a602.p.ssafy.io)
- port : 6379
- password : 1234

---

## 외부서비스 필요정보

- 경로 : S05P13A602/commbServer/src/main/resources/application.yaml

## Kakao API

```
spring:
  security:
    oauth2:
      client:
        registration:
          kakao:
            authorization-grant-type: authorization_code
            client-id: xxxxxxxx
            client-secret: xxxxxx
            redirect-uri: "{baseUrl}/api/login/oauth2/code/{registrationId}"
            scope:
            client-authentication-method: POST
            client-name: Kakao

        provider:
          kakao:
            authorization-uri: https://kauth.kakao.com/oauth/authorize
            token-uri: https://kauth.kakao.com/oauth/token
            user-info-uri: https://kapi.kakao.com/v2/user/me
            user-name-attribute: id
```

## SMTP

```
spring
mail:
  host: smtp.gmail.com
  port: 587
  username: xxxx
  password: xxxxx
  properties:
    mail:
      smtp:
        auth: true
        starttls:
          enable: true
          required: true
        connectiontimeout: 5000
        timeout: 5000
        writetimeout: 5000
```

## S3 Buket

```
cloud:
  aws:
    credentials:
      accessKey : xxxxx
      secretKey : xxxxxx
    s3:
      bucket : ${buket_id}
    region:
      static: ap-northeast-2
    stack:
      auto: false
```

## Firebase Cloud Messaging

- config.js

```
{
  "type": "xxxx",
  "project_id": "xxxx",
  "private_key_id": "xxxxxx",
  "private_key": "-----BEGIN PRIVATE KEY-----\xxxxxxx\n-----END PRIVATE KEY-----\n",
  "client_email": "xxxxx",
  "client_id": "xxxx",
  "auth_uri": "xxxxx",
  "token_uri": "xxxxxx",
  "auth_provider_x509_cert_url": "xxxx",
  "client_x509_cert_url": "xxxxxxxxxx"
}
```