



# 포팅 매뉴얼

## 도메인 생성 & SSL 인증서 발급

1. `duckdns.org` 에서 도메인 주소 생성 후 ec2 접속
2. `duck.sh` 를 다음과 같이 작성

```
mkdir duckdns
cd duckdns
vi duck.sh
```

```
#!/bin/bash
current=""
while true; do
    latest=`ec2-metadata --public-ipv4`
    echo "public-ipv4=$latest"
    if [ "$current" == "$latest" ]
    then
        echo "ip not changed"
    else
        echo "ip has changed - updating"
        current=$latest
        echo url="https://www.duckdns.org/update?domains=exampledomain&token=a7c4d0ad-114e-40ef-ba1d-d217904a50f2&ip=" | curl -k -o ~/
    fi
    sleep 5m
done
```

3. `duck_daemon.sh` 에 다음과 같이 추가

```
chmod 700 duck.sh
vi duck_daemon.sh
```

```
#!/bin/bash
su - ubuntu -c "nohup ~/duckdns/duck.sh > ~/duckdns/duck.log 2>&1&"
```

```
chmod +x duck_daemon.sh
sudo chown root duck_daemon.sh
sudo chmod 744 duck_daemon.sh
```

4. 환경설정 이후 SSL 인증서 발급

```
apt-get install python3-certbot-nginx
certbot certonly --nginx -d day6scrooge.duckdns.org
```

## Nginx 설치 및 설정

```
sudo apt install nginx

sudo mkdir /etc/nginx/site-available
sudo mkdir /etc/nginx/site-enabled
```

default.conf를 다음과 같이 설정

```
server {
    listen 80;

    server_name day6scrooge.duckdns.org;
    return 301 https://day6scrooge.duckdns.org$request_uri;
    client_max_body_size 20M;
}

server {
    listen 443 ssl;
    server_name day6scrooge.duckdns.org;
    # Certificate
    ssl_certificate /etc/letsencrypt/live/day6scrooge.duckdns.org/fullchain.pem;

    # Private Key
    ssl_certificate_key /etc/letsencrypt/live/day6scrooge.duckdns.org/privkey.pem;

    # 리액트 애플리케이션으로의 프록시 설정
    location / {
        proxy_pass http://localhost:3000; # 리액트 애플리케이션 실행 포트
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_set_header Host $host;
    }

    # 스프링 애플리케이션으로의 프록시 설정
    location /api {
        proxy_pass http://localhost:8081;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_set_header Host $host;
    }
    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
        root /usr/share/nginx/html;
    }
    client_max_body_size 20M;
}
```

## Jenkins

### 1. java 설치

```
(sudo su)

apt-get install openjdk-8-jdk
```

### 2. Jenkins 설치 및 실행

```
wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -
sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > \
/etc/apt/sources.list.d/jenkins.list'
sudo apt-get update
sudo apt-get install jenkins
sudo systemctl start jenkins
```

### 3. 빌드 설정

- 소스 코드 관리

Git 체크

Repository URL : GitLab 레포지토리 URL 입력

Credentials : Add 드롭박스에서 Jenkins 선택 후 GitLab에서 발급받은 GitLab API token 추가

Branches to build : \*/develop

- 빌드 유발

Build when a change is pushed to GitLab. GitLab webhook URL:{publicIP:8080/project/gitlab} 선택

Opened Merge Request Events 체크

Approved Merge Requests 체크

#### 4. GitLab Webhook 설정

GitLab 레포지토리 - Settings - Webhooks

Secret token 입력

Merge request events 및 Enable SSL verification 체크

## Spring

```
(sudo su)
cd "/var/lib/jenkins/workspace/gitlab/Server/scrooge/"
chmod +x ./gradlew
./gradlew wrapper --gradle-version 8.1.1
./gradlew bootRun
```

## React

```
(sudo su)
cd "/var/lib/jenkins/workspace/gitlab/scrooge-react/"
npm install -g serve
serve -s build
```

## FastAPI

### 1. python 설치

```
sudo su
apt-get upgrade
apt install python3-pip
```

### 2. fastapi 디렉토리로 이동 후 가상환경 설정

```
cd /var/lib/jenkins/workspace/gitlab/image-analysis/
python -m venv venv
source venv/bin/activate
pip install -r requirements.txt
```

### 3. fastapi 서버 실행

```
uvicorn main:app --host 0.0.0.0 --port 8000 --workers 4 --no-access-log &

백그라운드 실행 시
nohup uvicorn main:app --host 0.0.0.0 --port 8000 --workers 4 --no-access-log > fastapi.log 2>&1 &
```

## GCP 버킷 연결

### 1. 서비스 계정 생성

GCP 네비게이션 - API 및 서비스 - 사용자 인증 정보 - +사용자 인증 정보 만들기 - 서비스 계정 - 이름 설정  
main/resources 디렉터리에 json 키 파일 저장 후 application.properties 에서 적용  
spring.cloud.gcp.storage.credentials.location=classpath:{json파일명}

### 2. 서비스 계정 권한 부여

GCP 버킷에 접속 - 권한 탭에서 해당 서비스 계정에 대해 역할 지정  
역할 : 저장소 개체 관리자 저장소 개체 생성자 저장소 관리자

### 3. 공개 액세스 허용

권한 탭에서 액세스 권한 부여  
새 주 구성원 으로 allUsers 설정 후 역할을 저장소 개체 뷰어 로 설정

## Loaddata - MySQL 서버 생성 후 데이터 load

```
(sudo su)

apt-get update
apt-get install mysql-server
ufw allow mysql

(mysql 서버 시작)
systemctl start mysql

(ubuntu 재실행시 mysql 자동실행)
systemctl enable mysql

mysql -u root -p
# Enter password: day6s

create database scrooge;
mysql -u root -p scrooge < mysqldumpdata.sql
```

## 기술 버전

### BE

- IntelliJ IDEA 2023.01
- Java - openjdk 1.8
- Spring Boot 2.7.14
- JPA
- Spring Web
- MySQL 8.0.33.0
- Swagger 3.0.0
- GCP 1.2.5
- Spring Security - 5.7.10
- JWT - 0.11.2

### FE

- Gitignore 처리한 핵심 킷들  
.env.local(환경 변수)

/node\_modules(라이브러리)

- **개발 환경**

Node.js: 18.16.0

VSCode: 1.18.1

- **라이브러리**

React: 18.2.0

React-dom: 18.2.0

React-Redux: 8.1.2

@reduxjs/toolkit: 1.9.5

React-router-dom: 6.14.2

js-cookie: 3.0.5

recharts: 2.7.2

serve: 14.2.0

stompjs: 2.3.3

net: 1.0.2

axios: 1.4.0

## **Mobile**

- Android Studio 2022.2.11
- Android SDK 33
- Kotlin 1.8.20
- okhttp3 4.9.1
- retrofit2 2.9.0

## **AI**

- Python - 3.8.10
- OpenCV-python - 4.8.0.76
- FastAPI - 0.101.1

## **Infra**

- AWS EC2 Ubuntu 20.04 LTS(GNU/Linux 5.4.0-1018-aws x86\_64)
- Jenkins 2.401.3
- NGINX 1.18.0