



포팅메뉴얼

자바 설치

```
sudo apt-get update && sudo apt-get upgrade  
  
sudo apt-get install openjdk-11-jdk  
  
java --version
```

```
ubuntu@ip-172-26-13-164:~$ java --version  
openjdk 11.0.20 2023-07-18  
OpenJDK Runtime Environment (build 11.0.20+8-post-Ubuntu-1ubuntu120.04)  
OpenJDK 64-Bit Server VM (build 11.0.20+8-post-Ubuntu-1ubuntu120.04, mixed mode, sharing)
```

도커 설치

```
sudo apt-get update  
  
sudo apt-get install ca-certificates curl gnupg  
  
sudo install -m 0755 -d /etc/apt/keyrings  
  
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg  
  
sudo chmod a+r /etc/apt/keyrings/docker.gpg  
  
echo "deb [arch=$(dpkg --print-architecture)] signed-by=/etc/apt/keyrings/docker.gpg https://download.docker.com/linux/ubuntu $(. /  
  
sudo apt-get update  
  
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

```
ubuntu@ip-172-26-13-164:~$ sudo docker -v  
Docker version 24.0.5, build ced0996  
ubuntu@ip-172-26-13-164:~$ sudo docker compose version  
Docker Compose version v2.20.2
```

Vim 설치

```
sudo apt update  
  
sudo apt install vim
```

MYSQL 설치

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

```
sudo systemctl status mysql

sudo apt install -y mysql-client
```

MySQL 새로운 사용자 권한 설정

```
sudo mysql -u root -p
비밀번호는 없음 강 엔터 고

Chu 데이터베이스 생성하기
CREATE SCHEMA chu;

CREATE USER 'chufamily11'@'%' IDENTIFIED WITH mysql_native_password BY '!';

#이렇게 필요한 권한만 줘야해
GRANT SELECT, INSERT, UPDATE, DELETE, ALTER, CREATE, DROP ON Chu.* TO 'chufamily11'@'%';

FLUSH PRIVILEGES;

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

#bind-address          = 127.0.0.1
bind-address           = 0.0.0.0

sudo service mysql restart
```

젠킨스 도커 이미지로 띄우기

/home/ubuntu/docker-compose.yml

```
version: '3'

services:
  jenkins:
    image: jenkins/jenkins:lts
    container_name: jenkins
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
      - /jenkins:/var/jenkins_home
    ports:
      - "8080:8080"
    user: root
```





```
sudo docker compose up -d
```

젠킨스 인증정보 설정

Global credentials (unrestricted)

+ Add Credentials

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
 jenkins_token	tjslws8062@skuniv.ac.kr/*****	Username with password	
 ssh_key	ssh_key	SSH Username with private key	ssh_key 

- 비밀번호는 GIT 계정에서 ACCESS-TOKEN 발급 후 지정

젠킨스 TOOLS 설정

- 프론트엔드 빌드를 위한 노드 버전 설정

NodeJS installations ^ Edited

NodeJS installations

List of NodeJS installations on this system

Add NodeJS

NodeJS Name

NodeJS 18.17.0

☒ Install automatically ?

Install from nodejs.org

Version

NodeJS 18.17.0

For the underlying architecture, if available, force the installation of the 32bit package. Otherwise the build will fail

☐ Force 32bit architecture

젠킨스 환경 변수 설정

젠킨스 깃랩 설정

GitLab

☒ Enable authentication for '/project' end-point ?

GitLab connections

Connection name ?

A name for the connection

deploy-test

GitLab host URL ?

The complete URL to the GitLab server (e.g. http://gitlab.mydomain.com)

https://lab.ssafy.com/

Credentials ?

API Token for accessing GitLab

- none -

Add

API Token for GitLab access required

깃랩 웹훅 설정

Project Hooks (2)		
http://3.35.134.99:8080/project/deployment-pipeline-back/ Push events SSL Verification: enabled	Test ▾	Edit Delete
http://3.35.134.99:8080/project/deployment-pipeline-front/ Push events SSL Verification: enabled	Test ▾	Edit Delete

젠킨스 플러그인 설치

- gitlab 관련
- docker 관련
- generic webhook trigger
- ssh agent plugin
- ssh server
- node 관련 플러그인 깔기 (NodeJS Plugin)

깃 웹훅 설정은 해봤으니 파이프라인 구성만 잘 하면 됨

백엔드 파이프라인 deployment-pipeline

프론트 파이프라인 deployment-pipeline-front

서버 폴더 구조 생성

- chu-back, chu-front → home/ubuntu 위치에 생성
- deploy_be.sh, deploy_fe.sh → home/ubuntu 위치에 생성
- deploy 파일들은 권한 부여 미리 해야함

```
deploy_be.sh

cd chu-back
sudo docker compose down
sudo docker compose up -d --build
yes | sudo docker system prune
```

```
deploy_fe.sh

cd chu-front
# yes | sudo docker system prune
tar -xvf Frontend_0.1.0.tar
rm -rf Frontend_0.1.0.tar
sudo docker compose down
sudo docker compose up -d --build
# sudo service nginx restart
```

chu-back 폴더 내의 파일들

- 빌드된 jar 파일 (젠킨스가 만들어서 복사할꺼임)
- Dockerfile
- docker-compose.yml

Dockerfile

```
FROM adoptopenjdk/openjdk11
## 환경변수 설정
COPY ./Chu-0.0.1-SNAPSHOT.jar /Chu-0.0.1-SNAPSHOT.jar
CMD ["java", "-jar", "Chu-0.0.1-SNAPSHOT.jar"]
```

```
version: '3'
services:
  spring-boot-app:
    build:
      context: .
      dockerfile: Dockerfile
    image: chu-back
    container_name: chu-back
    ports:
      - "9090:9090"
```

chu-front 폴더 내의 파일들

- 빌드된 tar 파일 (젠킨스가 만들어서 복사할꺼임)
- Dockerfile
- docker-compose.yml
- fullchain.pem
- privkey.pem

Dockerfile

```
# Node.js image get
FROM node:16 as build

# for tar file directory
RUN mkdir /Frontend

WORKDIR /Frontend

COPY ./package.json ./package-lock.json ./

# install dependency for application
RUN npm install

COPY . .

# build React application
RUN npm run build

# set Runtime image
FROM nginx:latest

# copy build react app file to nginx static file directory
COPY --from=build /Frontend/build /usr/share/nginx/html

# 도커 상의 엔진엑스에 컨피그 파일 입히기
COPY front.conf /etc/nginx/conf.d/default.conf

# RUN React Application
CMD ["nginx", "-g", "daemon off;"]
```

docker-compose.yml

```
version: '3'
services:
  react-app:
    build:
      context: .
      dockerfile: Dockerfile
```

```
image: chu-front
container_name: chu-front
ports:
  - "3002:3002"
```

브라우저 새로고침 문제 해결

```
front.conf

server {
    listen 80;
    server_name _;

    location / {
        root    /usr/share/nginx/html;
        index   index.html index.htm;
        try_files $uri /index.html;
    }

    error_page   500 502 503 504  /50x.html;
    location = /50x.html {
        root    /usr/share/nginx/html;
    }
}
```

Nginx 설치

```
sudo apt update
sudo apt install nginx
```

인증서 발급받기

```
#인증서 발급받기 이전에

/etc/nginx/site-available/default
server-name 수정

sudo certbot --nginx -d your_domain.com

sudo certbot --nginx -d _

#이렇게 하면
/etc/nginx/site-available/default가 알아서 바뀔거야
```

/etc/nginx/site-available/default 에 코드 추가

```
server {
    if ($host = _) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    listen 80 default_server;
    listen [::]:80 default_server;

    server_name i9b111.p.ssafy.io;
    return 404; #managed by Certbot
}
```

openvidu

- 관련 포트 싹 다 열기
- .env 파일에서 http, https 주석 해제 후 포트 변경, 8442, 8445 포트 열기
- docker exec -it openviduNginx컨테이너번호 cat /etc/nginx/conf.d/default.conf
- docker exec -it 39bc765c9cd1 cat /etc/nginx/conf.d/default.conf

openvidu 관련 포트 열기

```
sudo ufw allow 22/tcp

sudo ufw allow 80/tcp

sudo ufw allow 443/tcp

sudo ufw allow 3478/tcp
sudo ufw allow 3478/udp

sudo ufw allow 40000:57000/tcp
sudo ufw allow 40000:57000/udp

sudo ufw allow 57001:65535/tcp
sudo ufw allow 57001:65535/udp

sudo ufw allow 8442

sudo ufw allow 8445
```

openvidu 설치

```
sudo su

cd /opt

curl https://s3-eu-west-1.amazonaws.com/aws.openvidu.io/install_openvidu_2.22.0.sh | bash
```

openvidu 관련 설정

```
vi /opt/openvidu/.env

./openvidu start
```