



## 같이 집관

삼성 SW 청년 아카데미 대전 캠퍼스 7기  
공동 프로젝트 (6 주 2022/07/11 ~ 2022/08/19)

### 포팅 매뉴얼

담당 컨설턴트 : 이승윤 컨설턴트  
유일권(팀장), 김보연, 최종수, 최주희, 최준혁, 홍성덕

## <<목차>>

1. 프로젝트 소개 .....	1
2. 기술 스택 .....	2
3. 빌드 상세 내용 .....	3
4. DB 설치 및 설정 .....	5
5. 프로퍼티 정의.....	6

## 1. 프로젝트 소개

스포츠 경기를 현장에서 직접 관람을 하면, 사람들과 같은 팀을 응원하면서 소속감/유대감을 느낄 수 있어 재밌게 응원할 수 있다. 그러나 최근 코로나의 여파 또는 티켓을 구하는 데에 실패하거나 시간/거리 상의 이유로 직관을 하기가 어려운 경우가 많다. 이러한 이유로 '같이 집관'을 서비스하게 되었다.

'같이 집관'은 직관을 가지 못하더라도 화상에서 실시간으로 사람들과 경기 화면을 보면서 응원할 수 있다. 또한 현장감과 소속감을 느낄 수 있도록 응원가, 전광판, 실시간 경기 정보, 승부예측 시스템 등도 구현되어 있다.

## 2. 기술 스택

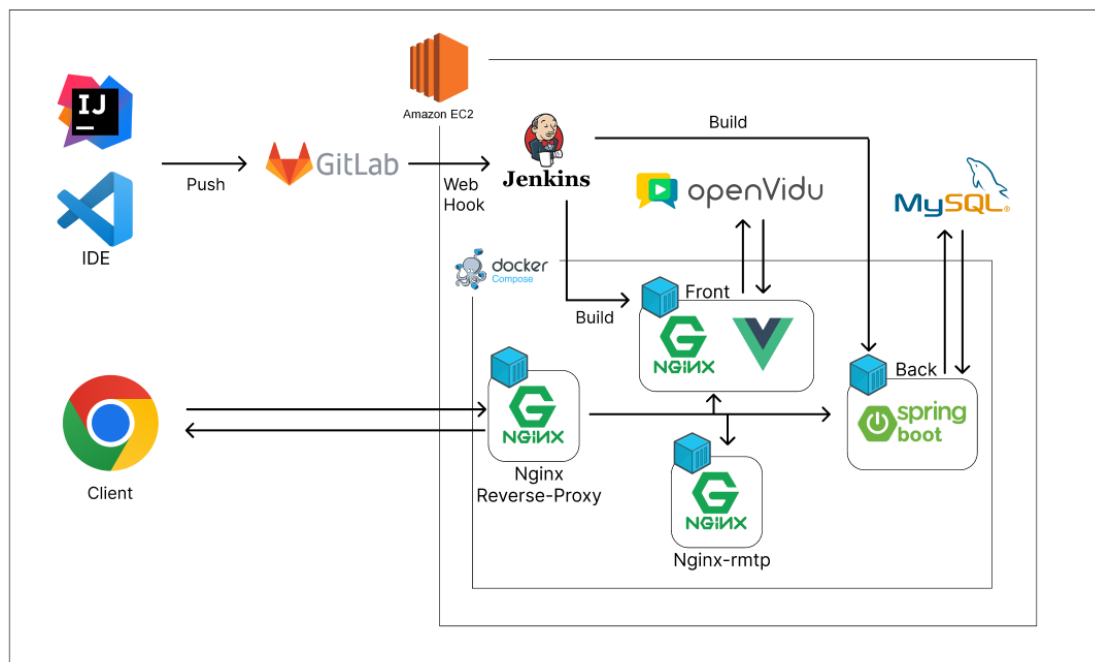
구분	기술 스택	상세 내용	버전
공통	형상 및 이슈 관리	GitLab, Jira	-
	커뮤니케이션	Mattermost, Notion	-
	화상 회의	Webex	-
Backend	DB	MySql	8.0.29
		JPA	2.7.2
	JDK	Zulu	8.33.0.1
	Spring	Spring	5.3.2
		Spring Boot	2.7.2
	IDE	IntelliJ	2022.1.2
	Build	Gradle	7.5
	WebRTC	OpenVidu	2.22
	API Docs	Swagger2	3.0.0
Frontend	HTML5		-
	CSS3		
	JavaScript(ES6)		
	Vue	Vue	3.2.13
		Vue-router	4.0.13
		Pinia	2.0.14
		Vuetify	3.0.0-beta
	IDE	Visual Studio Code	1.70
Server	Server	AWS EC2	
	OS	Ubuntu	20.04.3 LTS
	배포	Docker	20.10.17
		Jenkins	2.346.2
	웹 서버	Nginx	1.23.1
		Nginx-rmtp	1.18.0

### 3. 빌드 상세 내용

#### ▶ 개요

저희 “같이집관” 서비스의 배포환경 및 CI/CD 배포 자동화 흐름도 입니다.

배포 환경 및 서비스 흐름도



팀원들이 GitLab 에 코드를 작성하여 push 하게 되면 EC2 인스턴스의 Jenkins 가 GitLab 의 WebHook 을 통해 FrontEnd, BackEnd 프로젝트를 빌드하게 됩니다.

빌드가 완료되고, 빌드 된 2 개의 컨테이너와 Nginx-rtmp, Nginx 컨테이너를 docker-compose 를 통해 실행 시킵니다.

Nginx 컨테이너는 서비스화를 위해 리버스 프록시 서버로 설정하였습니다. Frontend 는 443 포트로, Backend 서버는 8081 포트로 Nginx-rtmp 서버는 8080 포트로 설정해 Load Balancing 이 가능하도록 구축하였습니다.

## ▶ Docker

- Docker 및 Docker-Compose 설치

아래 명령어를 순서대로 입력하여 Docker 를 설치한다.

```
$ sudo apt-get update
$ sudo apt-get install ca-certificates curl gnupg lsb-release
$ echo ₩
    "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg]
    https://download.docker.com/linux/ubuntu ₩
    $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
$ sudo apt-get update
$ sudo apt-get install docker-ce docker-ce-cli containerd.io
```

명령어를 순서대로 입력하고 설치가 되었다면 아래 명령어로 확인한다.

```
$ sudo docker --version
```

Docker 를 설치했다면, Docker-compose 를 설치한다.

```
$ sudo curl -L
    "https://github.com/docker/compose/releases/download/1.29.2/docker-
    compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose$ sudo
    apt-get install ca-certificates curl gnupg lsb-release

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

Docker-compose 설치가 다되었다면, 아래 명령어로 확인한다.

```
$ docker-compose -version
```

## ▶ FrontEnd

프로젝트를 GitLab 에서 Clone 받은 후, 아래 명령어를 통해 Docker Container Image 를 생성한다.

빌드를 위한 Dockerfile 은 프로젝트 내에 작성되어 있다.

```
$ docker build -t cheertogether_fe ./frontend
```

컨테이너 이미지 이름은 “cheertogether\_fe”로 설정하였다.

## ▶ BackEnd

프로젝트의 Backend 폴더에서 Gradle 을 통해 빌드를 진행한다.

```
$ ./gradlew clean build
```

이후 아래 명령어를 통해 Docker Container Image 를 생성한다.

빌드를 위한 Dockerfile 은 프로젝트 내에 작성되어 있다.

```
$ docker build -t cheertogether_be ./backend
```

## ▶ 방화벽 설정

같이 집관을 EC2 에서 실행하기 위해서 리눅스 방화벽을 열어주어야한다.

만약 개인 EC2 를 사용한다면, EC2 보안그룹에서도 제외해주어야 한다.

```
$ sudo ufw allow <포트 번호>
```

방화벽을 해제해야하는 포트들

```
3306 : MySql  
80 : HTTP  
443 : HTTPS  
3478, 40000~57000, 57000~65535 TCP+UDP : Openvidu  
1935 : Nginx-rtmp 서버
```

## ▶ Docker-Compose

프로젝트의 최상단에서 Docker-Compose 를 통해서 4 개의 컨테이너를 한번에 실행 시킨다.

```
$ docker-compose up -d    #-d 옵션으로 백그라운드 실행
```

Docker-Compose 의 여러 명령어를 통해 컨테이너를 관리 한다.

```
$ docker-compose down    #docker-compose 종료
```

```
$ docker-compose logs    #docker-compose 로그 확인
```

## ▶ Openvidu

WebRTC 사용을 위해 서버에서 OpenVidu 를 설치한다.

```
$ cd /opt                #설치 권장 위치
```

```
$ sudo curl https://s3-eu-west-1.amazonaws.com/aws.openvidu.io/install_openvidu_latest.sh | sudo bash
```

OpenVidu 폴더로 이동해서 .env 파일을 수정한다.

```
$ cd /openvidu          #설치 권장 위치
```

```
$ sudo vim .env
```

```
$ sudo apt-get update    #vim 이 설치 안되어있는 경우
```

```
$ sudo apt-get install -y vim
```

```

DOMAIN_OR_PUBLIC_IP=<Linux 서버의 public ip 주소 또는 도메인>

OPENVIDU_SECRET=<사용할 비밀번호 입력>

CERTIFICATE_TYPE=letsencrypt

# default 값은 selfsigned 지만 selfsigned 방식 사용시 보안 문제를 야기합니다.
# SSL 키가 있다면 owncert 방식으로 하되, /owncert 디렉토리 안에 키가 있어야함!

LETSencrypt_EMAIL=<이메일>

HTTP_PORT=80

HTTPS_PORT=443

# HTTP_PORT 와 HTTPS_PORT 는 letsencrypt 방식의 키를 발급 받기 전까진 기본
포트인 80, 443 을 사용해야 합니다! 키를 발급받고 난 후부터는 포트 변경 가능!

```

.env 파일 수정 방법

모든 설치가 끝나면 서버를 실행한다.

```

$ sudo ./openvidu start    #Openvidu 실행

$ sudo docker ps -a        #실행중인 컨테이너 확인

```

아래처럼 컨테이너가 올라와있다면 정상 실행된 것이다.

CONTAINER ID	IMAGE	STATUS	PORTS	COMMAND	CR
491739a415f4	square4us_frontend	Up 8 minutes	0.0.0.0:3000->80/tcp, :::3000->80/tcp	"/docker-entrypoint.s..."	8
1c44263f7e94	square4us_backend	Up 8 minutes	0.0.0.0:8080->8080/tcp, :::8080->8080/tcp	"/bin/sh -c 'java -j..."	8
c2730f1d7278	openvidu/openvidu-coturn:5.0.0	Up 4 hours		"docker-entrypoint.s..."	4
80301ac70daf	kurento/kurento-media-server:6.16.0	Up 4 hours (healthy)		"/entrypoint.sh"	4
2cbaab3ff763	openvidu/openvidu-server:2.19.0	Up 4 hours		openvidu_kms_1 "/usr/local/bin/entr..."	4
674f9a1810b7	openvidu/openvidu-redis:3.0.0	Up 4 hours		openvidu_openvidu-server_1 "docker-entrypoint.s..."	4
aab6dbbf5c5c	openvidu/openvidu-proxy:7.0.0	Up 4 hours		openvidu_redis_1 "/docker-entrypoint.s..."	4
d4290338ad29	openvidu/openvidu-call:2.19.0	Up 4 hours		openvidu_nginx_1 "docker-entrypoint.s..."	4



https:// <Linux 서버의 public ip 주소 또는 도메인>:<HTTPS\_PORT>에 접속해서 아래의 화면이 등장하면 성공이다.



## 4. DB 설치 및 설정

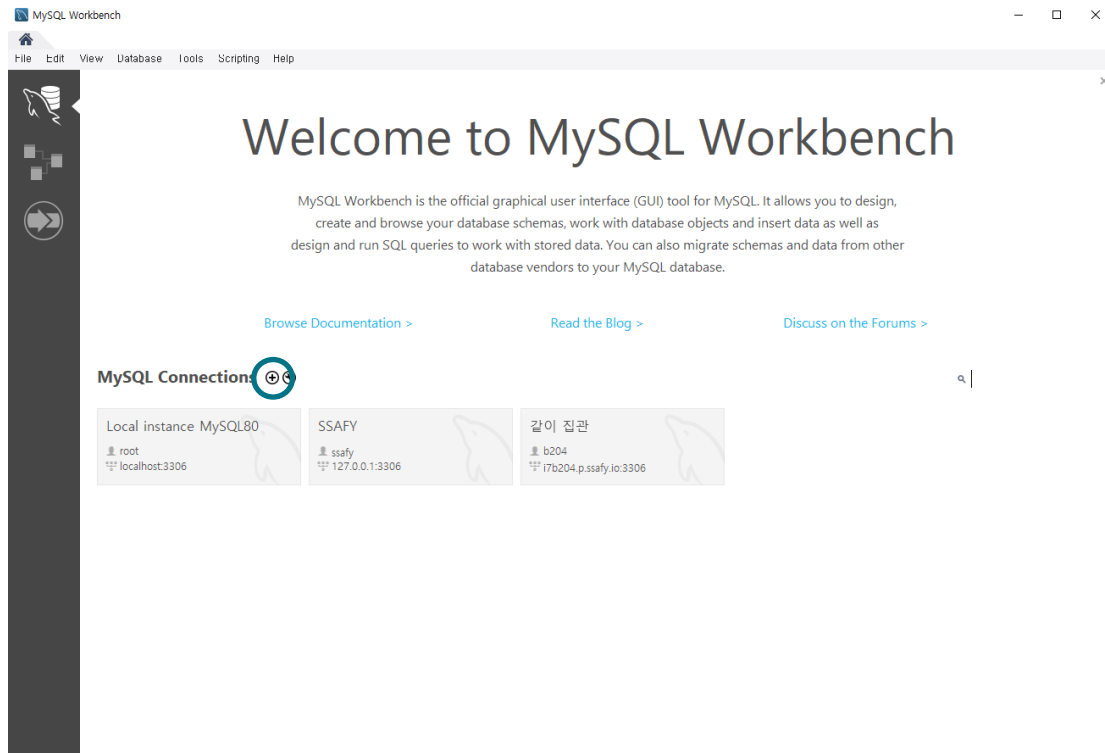
같이집관 서비스를 사용하기 위해서 DB 를 설치한다.

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

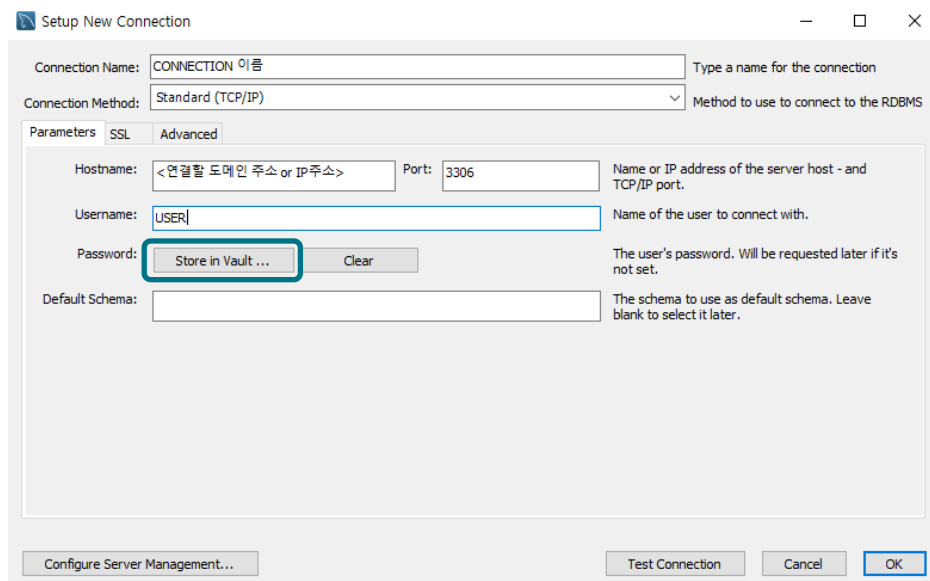
설치가 완료되면 MySql 서비스에 접속하고, 새 유저를 생성한다.

```
$ sudo /usr/bin/mysql -u root -p #root 유저로 비밀번호를 입력하고 접속  
$ CREATE USER 'USER@%' IDENTIFIED BY 'PASSWORD';  
$ GRANT ALL PRIVILEGES ON '.' TO 'USER '@%' IDENTIFIED BY 'PASSWORD';
```

이후 MySql WorkBench 에서 서버의 MySql 과 연결한다.



+ 버튼을 눌러 새로운 연결을 생성한다.



빈칸을 채우고, 비밀번호를 'Store In Vault'를 눌러 입력하고 OK 를 누른다.

이후 접속을 할 수 있습니다.