Coding with me

PORTING MAUNAL

SSAFY 7기 a304

2022

내용

[1. 기술 스택 2](#_Toc111594912)

[2. 서버 아키텍쳐 2](#_Toc111594913)

[3. Openvidu 서버 설치 4](#_Toc111594914)

[3.1 Openvidu 설치 4](#_Toc111594915)

[3.1.1 도커 설치(설치된 경우 생략) 4](#_Toc111594916)

[3.1.2 기존 파일 삭제(docker, openvidu 데이터가 이미 있는 경우 5](#_Toc111594917)

[3.1.3 openvidu On premises 설치 **(공식문서)** 6](#_Toc111594918)

[3.1.4 openvidu 설정 변경 6](#_Toc111594919)

[3.1.5 포트 개방 8](#_Toc111594920)

[3.1.6 Openvidu on promises 실행 8](#_Toc111594921)

[3.2 Nginx 설치 9](#_Toc111594922)

[3.3 Let’s encrypt & Nginx 세팅 9](#_Toc111594923)

[4. 컴파일러 서버 설치 10](#_Toc111594924)

[4.1 컴파일러 설치 10](#_Toc111594925)

[5. Spring/MySql/Quasar 서버 설치 11](#_Toc111594926)

[5.1 Mysql 설치 11](#_Toc111594927)

[5.2 (Optional) Mysql 외부 접속 허용 11](#_Toc111594928)

[5.3 Mysql 스키마 생성, 데이터 추가 13](#_Toc111594929)

[5.4 Spring 빌드 13](#_Toc111594930)

[5.5 Spring 배포 14](#_Toc111594931)

[5.6 Quasar 설치 14](#_Toc111594932)

[5.7 Quasar 실행 15](#_Toc111594933)

# 기술 스택

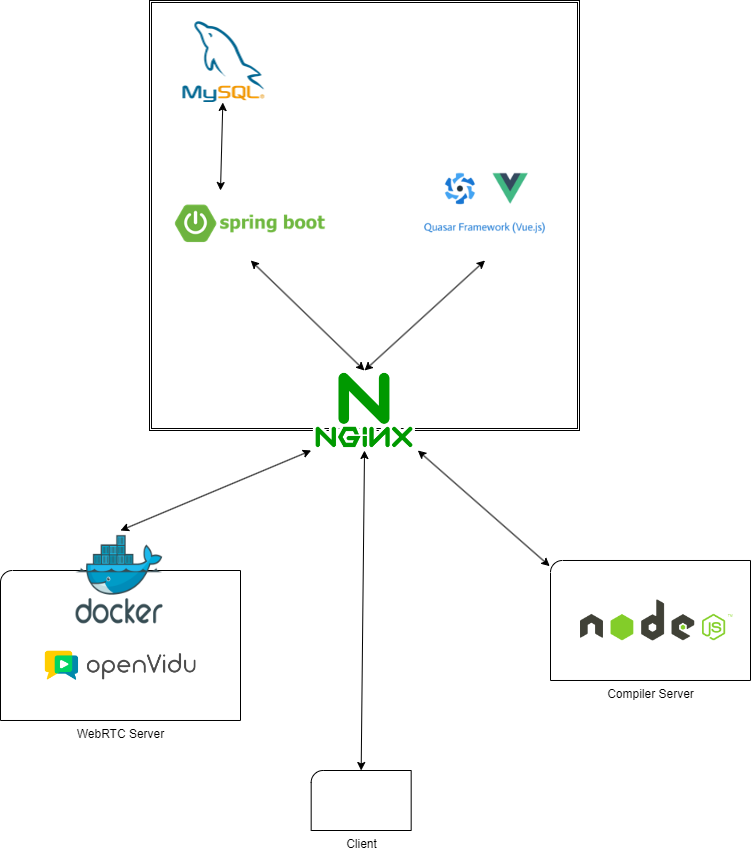
* 1. Spring/Quasar (Vue) Server: AWS EC2 Ubuntu 20.04 LTS
  2. Openvidu Server: Oracle Cloud Ubuntu 20.04 LTS

(Openvidu Server의 경우 ARM CPU 사용불가)

* 1. Compiler Server: Oracle Cloud Ubuntu 20.04 LTS
  2. Mysql: 20.04.2
  3. Spring Boot: 2.4.5
  4. Quasar: 1.0.5
  5. Openvidu: 2.22.0
  6. Nginx: 1.18.0
  7. Node js: 10.19.0
  8. java: 1.8.0
  9. gcc: 9.4.0
  10. g++: 9.4.0
  11. Python: 3.8.10

# 서버 아키텍쳐

* 성능을 위해, 컴파일러를 담당하는 서버와 WebRTC 서버를 별도로 분리해 총 3가지 서버로 이루어져 있음.
* Client에서 Nginx를 거쳐 Quasar로 접근해 웹페이지를 받아오고, Spring을 통해 API를 호출한다.
* WebRTC는 기본적으로 API를 통해 인증과 토큰 발급을 진행하지만, 그 이후는 client와 Openvidu 서버끼리만 통신한다.
* Spring에서 컴파일러 서버에 코드 실행을 요청하고, 그 결과를 사용자에게 반환한다.



# Openvidu 서버 설치

### Openvidu 설치

### 도커 설치(설치된 경우 생략)

|  |
| --- |
| # 도커 설치  sudo apt-get update  sudo apt-get install \  ca-certificates \  curl \  gnupg \  lsb-release  sudo mkdir -p /etc/apt/keyrings  curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg  echo \  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \  $(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null  sudo apt-get update  sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin  # **sudo apt-get update 실행할 때**  **GPG error: https://download.docker.com/linux/ubuntu jammy**  **InRelease: The foll owing signatures couldn't be verified because the public key is not available: N O\_PUBKEY 7EA0A9C3F273FCD8 에러 발생시**  # 이 명령어 실행 후 재도전  sudo chmod a+r /etc/apt/keyrings/docker.gpg  #docker compose 설치  sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose  sudo chmod +x /usr/local/bin/docker-compose |

### 기존 파일 삭제(docker, openvidu 데이터가 이미 있는 경우

|  |
| --- |
| **#기존 openvidu 관련 docker image가 존재한다면 모두 삭제해 주고 시작해야한다.**  **# docker, openvidu 설치를 한적이 없으면 생략**  sudo docker ps -a  #openvidu, kurento media server등의 컨테이너가 존재한다면 삭제한다.  $ sudo docker rm <ID or Name>  #컨테이너 모두 삭제를 원할 경우  $ sudo docker rm $(docker ps -a)  $ sudo docker images  # 이미지도 삭제  $ sudo docker rmi <ID or IMAGE>  # 이미지 전체 삭제를 원할 경우  $ sudo docker rmi $(sudo docker images) |

### **openvidu On premises 설치**[**(공식문서)**](https://docs.openvidu.io/en/stable/deployment/ce/on-premises/)

|  |
| --- |
| # 관리자 권한  $ sudo su  # openvidu가 설치되는 경로  $ cd /opt  # openvidu on promises 설치  $ curl <https://s3-eu-west-1.amazonaws.com/aws.openvidu.io/install\_openvidu\_latest.sh> | bash  $ exit |

### openvidu 설정 변경

|  |
| --- |
| $ cd /opt/openvidu  openvidu 설정 변경  인증서 타입의 기본값은 selfsigned이다.  만약 certbot을 이용해 인증서를 발급받았다면 설정을 변경해 주어야 한다.  참고: vi에서 ‘:/검색할 키워드 ‘를 통해 빠르게 원하는 내용을 찾을 수 있다.  # /opt/openvidu  $ vi .env  # OpenVidu configuration  # ----------------------  #Documentation: <https://docs.openvidu.io/en/stable/reference-docs/openvidu-config/>  # NOTE: This file doesn't need to quote assignment values, like most shells do.  # All values are stored as-is, even if they contain spaces, so don't quote them.  # Domain name. If you do not have one, the public IP of the machine.  # For example: 198.51.100.1, or openvidu.example.com  # 본인 서버의 도메인(example.com) 또는  # $ curl ifconfig.me 커맨드를 실행한 결과로 얻는 ip를 기입한다.  # openvidu 서버가 여기 작성한 도메인으로 실행된다.  DOMAIN\_OR\_PUBLIC\_IP=<도메인 또는 public IP>  # OpenVidu SECRET used for apps to connect to OpenVidu server and users to access to OpenVidu Dashboard  OPENVIDU\_SECRET=MY\_SECRET  # HTTPS 인증서 별도로 등록안했으면 selfsigned  # 인증서를 certbot으로 발급받았다면 letsencrypt  CERTIFICATE\_TYPE=letsencrypt  # If CERTIFICATE\_TYPE=letsencrypt, you need to configure a valid email for notifications  # 만약 인증서 타입이 letsencrypt라면 이메일 설정.  LETSENCRYPT\_EMAIL=user@example.com  # Proxy configuration  # If you want to change the ports on which openvidu listens, uncomment the following lines  # Allows any request to http://DOMAIN\_OR\_PUBLIC\_IP:HTTP\_PORT/ to be automatically  # redirected to https://DOMAIN\_OR\_PUBLIC\_IP:HTTPS\_PORT/.  # WARNING: the default port 80 cannot be changed during the first boot  # if you have chosen to deploy with the option CERTIFICATE\_TYPE=letsencrypt  # NGINX의 포트를 변경한다.  HTTP\_PORT=8081  # Changes the port of all services exposed by OpenVidu.  # SDKs, REST clients and browsers will have to connect to this port  HTTPS\_PORT=8443  ...  **.env 파일 저장 후, 기존 openvidu 인증 파일 삭제**  만약 /opt/openvidu경로 안에 certificates 폴더가 존재한다면 설정한 환경에 인증서가 제대로 적용되지 않을 때가 있다.  이 경우 폴더 자체를 삭제해 주면 된다.  # /opt/openvidu  $ sudo rm -rf certificates  **openvidu On Promises 실행** |

### 포트 개방

|  |
| --- |
| 포트 개방  aws도 동일하게 처리 필요    sudo apt update  sudo apt install netfilter-persistent  sudo iptables -A INPUT -p udp --match multiport --dports 40000:65535 -j ACCEPT  sudo iptables -A INPUT -p tcp --match multiport --dports 40000:65535 -j ACCEPT  sudo iptables -I INPUT 1 -p tcp --dport 22 -j ACCEPT  sudo iptables -I INPUT 1 -p tcp --dport 80 -j ACCEPT  sudo iptables -I INPUT 1 -p tcp --dport 443 -j ACCEPT  sudo iptables -I INPUT 1 -p tcp --dport 8081 -j ACCEPT  sudo iptables -I INPUT 1 -p udp --dport 8081 -j ACCEPT  sudo iptables -I INPUT 1 -p tcp --dport 8443 -j ACCEPT  sudo iptables -I INPUT 1 -p tcp --dport 8443 -j ACCEPT  service iptables save  service iptables restart  sudo netfilter-persistent save  sudo netfilter-persistent start |

### Openvidu on promises 실행

|  |
| --- |
| # /opt/openvidu  $ sudo ./openvidu start  # 종료할 때는 같은 경로에서  $ ./openvidu stop을 하면 된다. |

### Nginx 설치

|  |
| --- |
| #운영체제에서 사용 가능한 패키지들과 그 버전에 대한 정보(리스트) 업데이트  sudo apt update  #nginx 설치하기  sudo apt install nginx -y  #설치한 nginx 버전 확인  nginx -v  #nginx running 상태 확인  sudo service nginx status |

### Let’s encrypt & Nginx 세팅

|  |
| --- |
| sudo apt install software-properties-common  add-apt-repository ppa:certbot/certbot  apt update  apt install python3-certbot-nginx  certbot certonly --nginx -d (도메인)  # 경로 이동  cd /etc/nginx/sites-available  # 파일 생성  sudo vi 도메인명.conf  # 파일에 아래와 같은 내용을 입력하면 된다.(도메인은 서버에 맞는걸로)  upstream openvidu {  server 127.0.0.1:8443;  }  server {  server\_name (openvidu 도메인)  location / {  proxy\_pass https://openvidu/;  proxy\_set\_header X-Real-IP $remote\_addr;  #proxy\_set\_header X-Forward-For $proxy\_add\_x\_forwarded\_for;  proxy\_set\_header Host $host;  }  listen [::]:443 ssl ipv6only=on; # managed by Certbot  listen 443 ssl; # managed by Certbot  ssl\_certificate /etc/letsencrypt/live/(openvidu 도메인)/fullchain.pem; # managed by Certbot  ssl\_certificate\_key /etc/letsencrypt/live/(openvidu 도메인)/privkey.pem; # managed by Certbot  include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot  ssl\_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot  }  # 생성한 파일의 심볼릭 링크를 생성하고, nginx를 재시작한다.  sudo ln -s /etc/nginx/sites-available/도메인명.conf /etc/nginx/sites-enabled  # nginx 문법 검사  sudo nginx -t  #nginx 재시작  sudo service nginx reload |

# 컴파일러 서버 설치

### 컴파일러 설치

|  |
| --- |
| # 컴파일러를 설치하려는 서버에 접속 선행  # 업데이트  sudo apt update  # 서버 실행 & 자바스크립트 컴파일을 위한 node js 설치  sudo apt install nodejs  # 파이썬 설치  Sudo apt install python3  # c언어 설치  Sudo apt install gcc  # c++ 설치  Sudo apt install g++  # open jdk 1.8 설치  sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys 0x219BD9C9  sudo apt-add-repository 'deb http://repos.azulsystems.com/ubuntu stable main'  sudo apt update  sudo apt install zulu-8 |

|  |
| --- |
| # 프로젝트를 받아온다.  Git clone <https://lab.ssafy.com/s07-webmobile1-sub2/S07P12A304.git>  # 프로젝트 경로도 이동  cd S07P12A304/compiler  # 의존성 설치  npm i  # 서버 시작  node index.js |

* 1. 서버 배포

# Spring/MySql/Quasar 서버 설치

* Spring/Mysql/Quasar은 같은 서버에서 작동하는 것으로 설계되어있다.
* 같은 서버에 3개 전부 설치한 뒤에 nginx로 처리한다.

### Mysql 설치

|  |
| --- |
| sudo apt update  sudo apt install mysql-server |

### (Optional) Mysql 외부 접속 허용

* Spring과 mysql이 같은 서버에 있어 생략해도 실행에 문제가 없습니다.
* 개발 목적으로 외부에서 접속할 때 필요한 설정입니다.

|  |
| --- |
| cd /etc/mysql/mysql.conf.d  sudo vi mysqld.cnf  # mysqld.cnf 내용 변경. => bind-address를 0.0.0.0으로 변경한다.  #bind-address = 127.0.0.1  bind-address = 0.0.0.0  # mysql 재시작  sudo systemctl restart mysql  # mysql 접속  sudo mysql --defaults-file=/etc/mysql/debian.cnf  # mysql에서 계정만들기  ('admin'과 'vn23d@jve'는 각각 아이디, 비밀번호로 원하는 대로 설정가능.)  create USER 'admin'@'%' IDENTIFIED BY 'vn23d@jve';  grant all privileges on \*.\* to 'admin'@'%';  # mysql 종료 후, port 개방  Exit  sudo iptables -I INPUT 1 -p tcp --dport 22 -j ACCEPT  sudo iptables -A INPUT -p tcp --dport 8080 -j ACCEPT  sudo iptables -L  sudo apt install netfilter-persistent  sudo apt install iptables-persistent  sudo netfilter-persistent save  sudo netfilter-persistent start  service iptables save  service iptables restart |

### Mysql 스키마 생성, 데이터 추가

|  |
| --- |
| # Mysql 접속 후, 스키마 생성  mysql -u admin -p  (비번 입력)  CREATE SCHEMA ‘ssafy\_web\_db’ DEFAULT CHARACTER SET utf8;  # 종료  Exit  # 더미 데이터 추가  (exec에 있는 backup.sql 파일 서버로 업로드 하여 사용)  mysql -u admin -p ssafy\_web\_db < backup.sql |

### Spring 빌드

|  |
| --- |
| # 프로젝트 받아오기  Git clone <https://lab.ssafy.com/s07-webmobile1-sub2/S07P12A304.git>  # Application.properties 설정 변경  Backend/src/main/resources/application.properties   * Mysql에서 설정한 스키마명(ssafy\_web\_db), username/password를 맞춰서 넣는다.      * Openvidu가 있는 서버 도메인, .env 파일에 지정한 OPENVIDU\_SECRET를 넣는다.      * Jar 파일을 생성해준다.(사진은 Intellij 기준) |

### Spring 배포

|  |
| --- |
| # open jdk 1.8 설치  sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys 0x219BD9C9  sudo apt-add-repository 'deb http://repos.azulsystems.com/ubuntu stable main'  sudo apt update  sudo apt install zulu-8  # 이전에 생성한 java 파일 실행  Java -jar ssafy-web-project-1.0-SNAPSHOT.jar |

### Quasar 설치

|  |
| --- |
| # 서버 실행 & 자바스크립트 컴파일을 위한 node js 설치  sudo apt install nodejs  # quasar 설치  sudo Npm install vue  sudo npm install -g @vue/cli  sudo npm install -g @quasar/cli  # quasar 설치 확인  quasar -v  # yarn 설치  sudo npm install yarn |

### Quasar 실행

|  |
| --- |
| # 프로젝트를 받아온다.  git clone <https://lab.ssafy.com/s07-webmobile1-sub2/S07P12A304.git>  cd S07P12A304/FrontEnd  # 의존성 설치하고, 프로젝트 실행  yarn  sudo quasar dev |

* 1. Let’s encrypt, Nginx 설치
* [3.2](#_Nginx_설치), [3.3](#_Let’s_encrypt_&)과 동일하게 하되, nginx설정만 아래로 한다.

|  |
| --- |
| upstream spring {  server 127.0.0.1:8443;  }  server {  server\_name (도메인);  location /api {  proxy\_pass http://spring;  proxy\_set\_header X-Real-IP $remote\_addr;  #proxy\_set\_header X-Forward-For $proxy\_add\_x\_forwarded\_for;  proxy\_set\_header Host $host;  }  location /{  proxy\_pass https://localhost:9000;  proxy\_set\_header X-Real-IP $remote\_addr;  #proxy\_set\_header X-Forward-For $proxy\_add\_x\_forwarded\_for;  proxy\_set\_header Host $host;  proxy\_http\_version 1.1;  proxy\_set\_header Upgrade $http\_upgrade;  proxy\_set\_header Connection "Upgrade";  }  listen [::]:443 ssl ipv6only=on; # managed by Certbot  listen 443 ssl; # managed by Certbot  ssl\_certificate /etc/letsencrypt/live/(도메인)/fullchain.pem; # managed by Certbot  ssl\_certificate\_key /etc/letsencrypt/live/(도메인)/privkey.pem; # managed by Certbot  include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot  ssl\_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot |