포팅 메뉴얼

팀 C204	문서희(팀장), 김대현, 김윤민, 노은영, 서은지, 천지석
담당 컨설턴트	김성재 컨설턴트
프로젝트 기간	2022.07.11 ~ 2022.08.19 [공통, 6 주]

삼성청년 SW 아카데미
광주캠퍼스 7 기

목차

1. 프로젝트 기술 스택

1) 이슈 관리

- Jira

2) 형상 관리

- Gitlab

3) 커뮤니케이션

- Mattermost
- Notion
- Webex

4) 개발 환경

가. OS

- Window 10

나. IDE

- IntelliJ 7.4.1
- Visual Studio Code 1.70.0
- UI/UX : Figma, Procreate

다. Database

- MySQL 8.0.29
- MongoDB 5.0.10
- Redis 7.0.4

라. Server

- AWS EC2(Ubuntu 20.04.4 LTS)

마. Backend

- Java 1.8
- Spring Boot 2.7.1
- Spring Data JPA 2.7.1
- Spring Security 5.7.1
- OpenVidu 2.22.0

바. Frontend

- HTML5, CSS3, JS(ES6)
- React 17.0.1
- Redux 4.2
- Node js 16.9.0
- Face-api.js 0.22.2
- Openvidu-browser 2.22.0
- Web Speech API

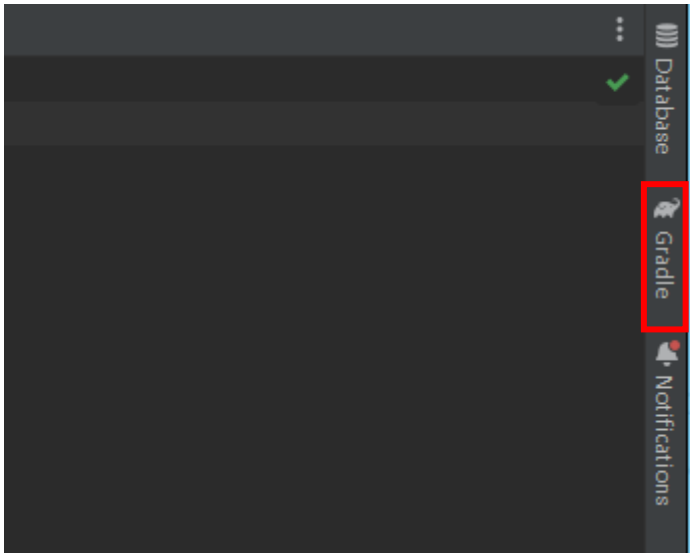
사. Deployment

- Docker 20.10.12
- Jenkins 2.346.2
- Nginx 1.21.6

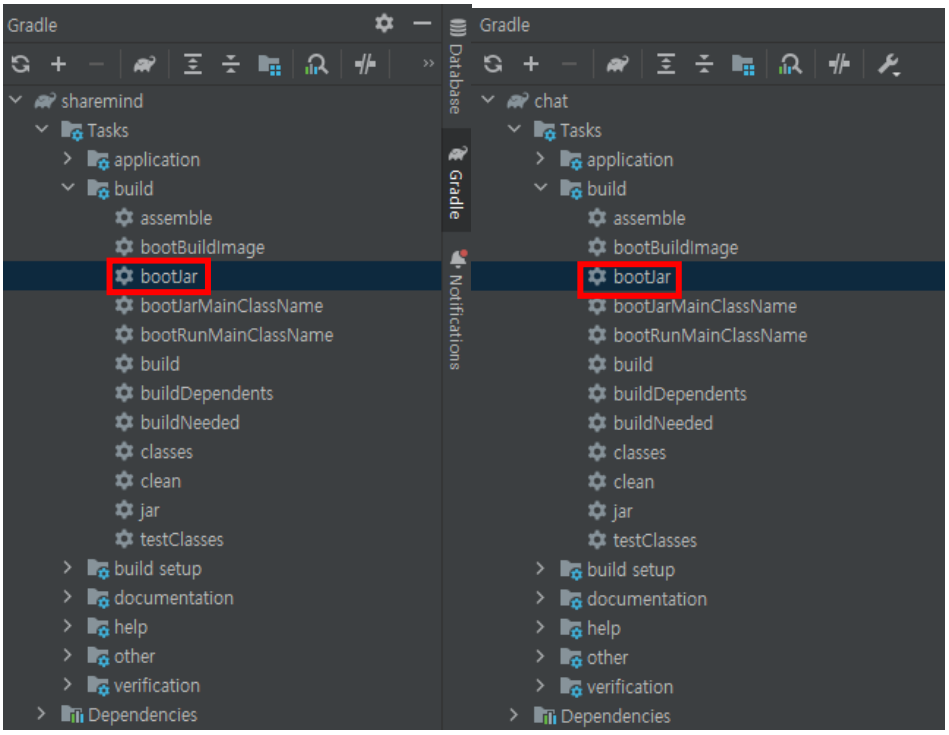
2. 빌드 상세

1) Backend

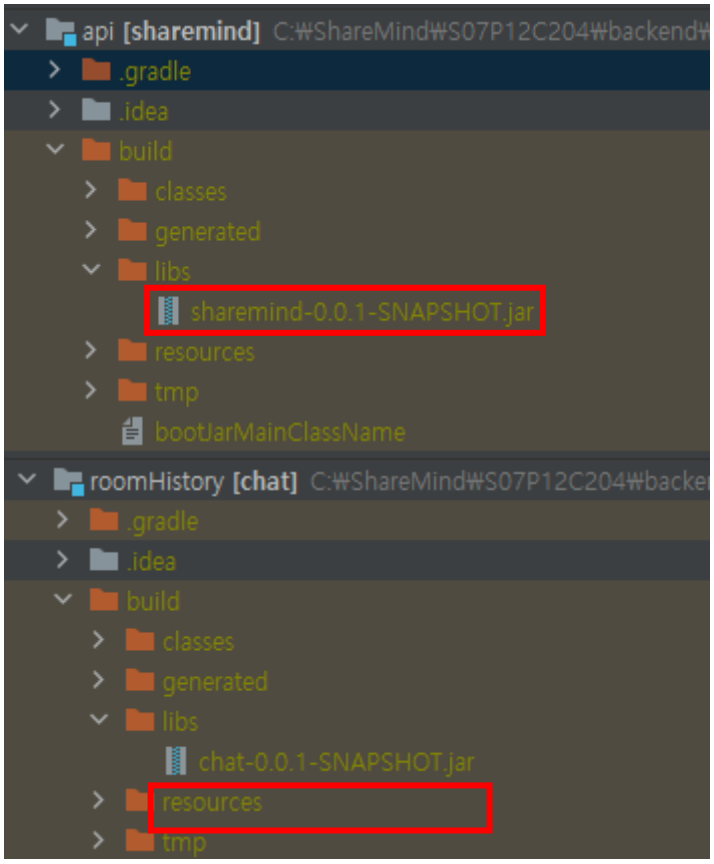
- 화면 우측 Gradle 클릭



- 프로젝트 하위 Tasks > build > Bootjar 실행



- build > libs 에 생성된 jar 확인



2) Frontend

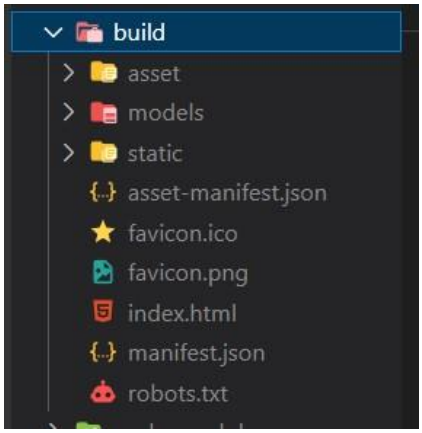
- node_modules 를 위한 install

```
npm install
```

- 빌드하기

```
npm run build
```

- root>build 에 빌드 산출물 존재



3. 배포 특이사항

젠킨스에 프론트엔드와 백엔드 아이템을 각각 등록하고, GitLab webhook 을 이용해 각 브랜치에 변동사항이 push 될 때마다 자동으로 배포되도록 구성합니다.

1) Frontend

- frontend 디렉토리로 이동합니다.

```
cd frontend
```

- 기존에 실행 중이던 frontend 컨테이너를 중지합니다.

```
docker stop frontend
```

- Dockerfile 을 통해 만들어진 최근 frontend-image 를 빌드합니다.

```
docker build -t frontend-image:latest .
```

- 백그라운드 모드에서 호스트의 3000 번 포트와 컨테이너의 3000 번 포트를 연결하고, frontend-image 로 frontend 라는 이름의 컨테이너를 실행합니다.

```
docker run -d --rm --name frontend -p 3000:3000 frontend-image
```

2) Backend

가. 백엔드 api(8080 번 포트)

- backend/api 디렉토리로 이동합니다.

```
cd backend/api
```

- bootjar 를 실행합니다.

```
./gradlew bootjar
```

- 기존에 실행 중이던 backend-api 컨테이너를 중지합니다.

```
docker stop backend-api
```

- Dockerfile 을 통해 만들어진 최근 backend-image 를 빌드합니다.

```
docker stop backend-api
```

- 백그라운드 모드에서 호스트의 8080 번 포트와 컨테이너의 8080 번 포트를 연결하고, youniverse-net 네트워크에서 backend-image 로 backend-api 라는 이름의 컨테이너를 실행합니다.

```
docker run -d --rm -p 8080:8080 --name backend-api --network youniverse-net backend-image
```

나. 로그 api(8000 번 포트)

- ../roomHistory 디렉토리로 이동합니다.

```
cd ../roomHistory
```

- bootjar 를 실행합니다.

```
./gradlew bootjar
```

- 기존에 실행 중이던 log-api 컨테이너를 중지합니다.

```
docker stop log-api
```

- Dockerfile 을 통해 만들어진 최근 log-image 를 빌드합니다.

```
docker build -t log-image:latest .
```

- 백그라운드 모드에서 호스트의 8000 번 포트와 컨테이너의 8000 번 포트를 연결하고, youniverse-net 네트워크에서 log-image 로 log-api 라는 이름의 컨테이너를 실행합니다.

```
docker run -d --rm -p 8000:8000 --name log-api --network youniverse-net chat-image
```

4. 외부 서비스

카카오

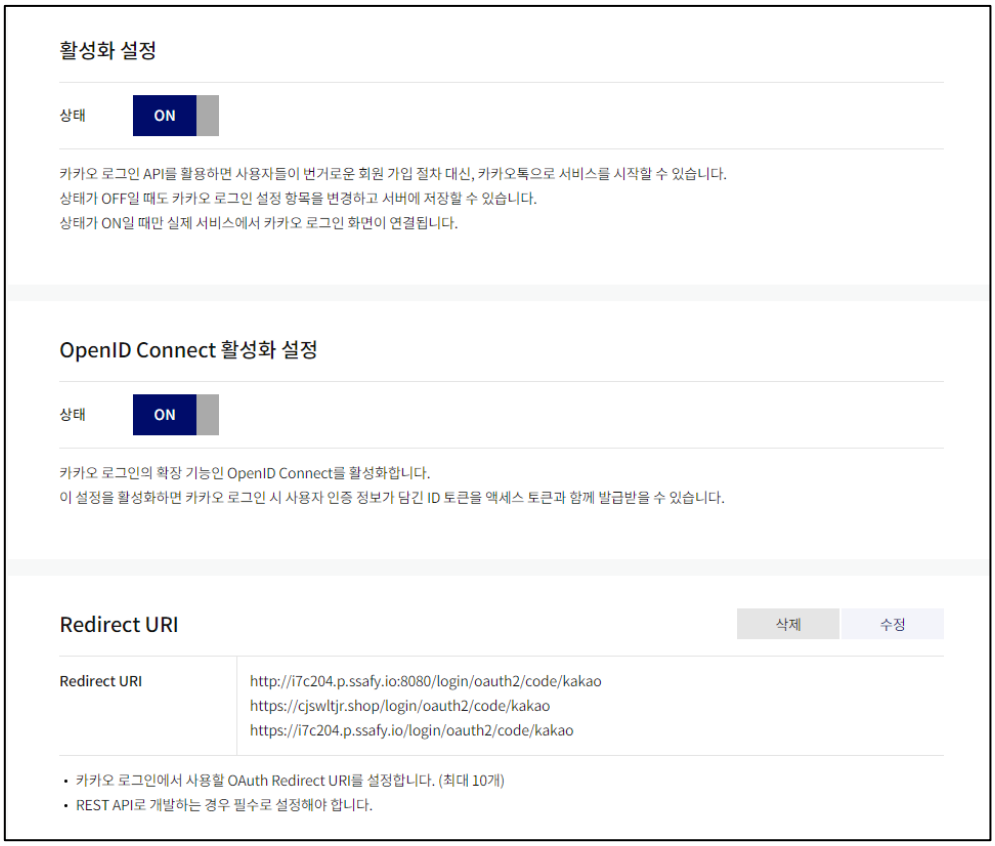
시각장애인이 스크린 리더를 이용하여 회원가입을 하려면 아이디, 비밀번호 및 기타 정보들을 모두 입력해야하는 번거로움이 있습니다. 저희는 카카오 로그인으로 회원 가입

절차를 없애 서비스의 접근성을 높였습니다. 또한 초대 링크 전송 시에도 링크를 복사하고 메신저로 공유해야하는 과정을 카카오톡 링크 공유로 대체하여 사용 시 번거로움을 최소화했습니다.

1) 애플리케이션 추가

2) 도메인 등록

3) 카카오 로그인, OpenID Connect 활성화 및 Redirect URI 설정



i. 개인정보 및 접근권한 동의 항목 설정

닉네임	profile_nickname	● 필수 동의	설정
카카오계정(이메일)	account_email	● 선택 동의	설정
카카오 서비스 내 친구목록(프로필사진, 닉네임, 즐겨찾기 포함)	friends	● 이용 중 동의	설정
카카오톡 메시지 전송	talk_message	● 이용 중 동의	설정

5. EC2 세팅

1) Docker 설치

- 세팅을 위해 최신 상태로 업데이트

```
sudo apt-get update
sudo apt-get upgrade
```

- docker 설치

```
curl https://get.docker.com | sudo sh
```

2) MySQL 도커 컨테이너 실행

- 이미지 가져오기

```
docker pull mysql
```

- 도커 컨테이너 실행

```
docker run --name mysql-container -e MYSQL_ROOT_PASSWORD=<password> -d -p 3306:3306 mysql
```

3) MongoDB 도커 컨테이너 실행

- 이미지 가져오기

```
docker pull mongo
```

- 도커 컨테이너 실행

```
docker run --name mongodb-container -d -p 27017:27017 mongo
```

4) Redis 도커 컨테이너 실행

- 이미지 가져오기

```
docker pull redis
```

- 도커 컨테이너 실행

```
docker run --name redis-container -p 6379:6379 -d redis
```

5) Openvidu On-premises 를 이용해 배포

- 권한 설정

```
sudo su
```

- /opt 디렉토리로 변경

```
cd /opt
```

- 오픈비두 설치

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

- /opt/openvidu 디렉토리로 변경

```
cd /opt/openvidu
```

- .env 파일 변경

```
nano .env
```

- 변경 후 .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
DOMAIN_OR_PUBLIC_IP=cjswltjr.shop

# OpenVidu SECRET used for apps to connect to OpenVidu server and users to access to OpenVidu Dashboard
OPENVIDU_SECRET=MY_SECRET

# Certificate type:
# - selfsigned:  Self signed certificate. Not recommended for production use.
#                Users will see an ERROR when connected to web page.
# - owncert:     Valid certificate purchased in a Internet services company.
#                Please put the certificates files inside folder ./owncert
#                with names certificate.key and certificate.cert
# - letsencrypt: Generate a new certificate using letsencrypt. Please set the
#                required contact email for Let's Encrypt in LETSENCRYPT_EMAIL
#                variable.
CERTIFICATE_TYPE=letsencrypt

# If CERTIFICATE_TYPE=letsencrypt, you need to configure a valid email for notifications
LETSENCRYPT_EMAIL=cjswltjr159@naver.com
```

- 오픈비두 실행

```
./openvidu start
```

6. 프로퍼티

1) Nginx 설정 변경

Openvidu Troubleshooting and FAQ 에서 16.2 Modify Openvidu Nginx configuration 을 참고하여 Nginx 설정을 변경하였습니다.

- 권한 설정

```
sudo su
```

- /opt/openvidu 디렉토리로 변경

```
cd /opt/openvidu
```

- custom-nginx.conf 파일 생성
  - custom-nginx.conf : 오픈비두의 nginx 환경 설정 정보를 가짐
  - nginx.conf : nginx 의 주요 설정 파일을 가짐

```
docker-compose exec nginx cat /etc/nginx/conf.d/default.conf > custom-nginx.conf
docker-compose exec nginx cat /etc/nginx/nginx.conf > nginx.conf
```

- /opt/openvidu/docker-compose.yml 파일에 설정 추가

```
nginx:
    ...
    volumes:
        ...
        - ./custom-nginx.conf:/custom-nginx/custom-nginx.conf
        - ./nginx.conf:/etc/nginx/nginx.conf
```

- custom-nginx.conf 환경 설정

```
nano custom-nginx.conf
```

- 변경 된 custom-nginx.conf 프로퍼티

```
server {
    listen 80;
    listen [::]:80;
    listen 443 ssl;
    listen [::]:443 ssl;
    server_name cjswltjr.shop;

    # SSL Config
   ssl_certificate         /etc/letsencrypt/live/cjswltjr.shop/fullchain.pem;
   ssl_certificate_key     /etc/letsencrypt/live/cjswltjr.shop/privkey.pem;
   ssl_trusted_certificate /etc/letsencrypt/live/cjswltjr.shop/fullchain.pem;

    # Your App
    location / {
        proxy_pass http://cjswltjr.shop:3000;
    }

    location /login {
        proxy_pass http://cjswltjr.shop:8080/login;
    }

    location /logout {
        proxy_pass http://cjswltjr.shop:8080/logout;
    }

    location /token {
        proxy_pass http://cjswltjr.shop:8080/token;
    }

    location /api/back {
        proxy_pass http://cjswltjr.shop:8080$request_uri;
    }

    location /logs {
        proxy_pass http://3.39.255.188:8000$request_uri;
    }
```

- 오픈비두 재시작

```
./openvidu restart
```