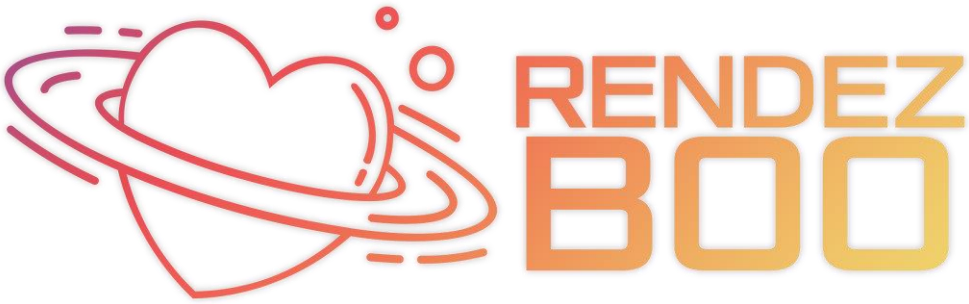

빌드 및 배포

공통 프로젝트

2023.01.03 ~ 2023.02.17

SSAFY 서울캠퍼스 8기

이홍주 김명준 안예나 유선준 이재원 정훈

1. 기술 스택

구분		상세내용	버전
공통	형상관리	GitLab	-
	이슈관리	Jira	-
	커뮤니케이션	Mattermost	-
		Notion	-
	IDE	IntelliJ	2022.3.1
		Visual Studio Code	1.75.1
BackEnd	DB	MySQL	8.0.31
		MongoDB	1.34.2
		Redis	7.0.8
	Java	Zulu	11.0.18
		JPA	-
	Spring	Spring Boot	2.7.7
	클라우드 스토리지	AWS S3	-
	Build	Gradle	7.6
	API Docs	Swagger2	3.0.0
	Security	JWT	0.11.5
FrontEnd	React	React	17.0.2
		Redux	4.2.1
		Redux Toolkit	1.9.2
	WebRTC	openvidu-browser	2.25.0
		Kurento	6.18.0
	WebSocket	StompJS	6.1.2
	Axios		0.21.1
	tensorflow-models/facemesh		0.0.3
	tensorflow/tfjs		1.7.4
	face-api.js		0.22.2
	Moment.js		2.29.4
	Three.js		0.149.0
Server	서버	AWS EC2	-
	플랫폼	Ubuntu	20.04 LTS
	배포	Jenkins	2.375.2
		Docker	23.0.0
		Docker Compose	2.15.1
		Nginx	1.23.3

2. 빌드 및 프로퍼티 상세내용

외부로 공개되면 안되는 모든 정보는 반출 신청 시 다른 비공개 파일로 뺄 예정입니다.

Configuration Settings

- 환경변수에 AWS_ACCESS_KEY_ID, AWS_SECRET_KEY 추가 필요

application.yml

- MySQL url, username, password
- 네이버, 카카오 CLIENT_SECRET
- JWT secret
- Openvidu url, secret
- AWS S3 cloudfront

OAuthServiceImpl.java

- 네이버, 카카오 CLIENT_ID

SmsServiceImpl.java

- CoolSMS apiKey, apiSecretKey
- 인증번호 문자 발신번호

3. 배포 특이사항

Jenkins를 사용하여 GitLab에서 Merge 성공 시 자동으로 빌드 및 배포되게 CI/CD를 구성하였습니다.

Jenkins 플러그인 설치

- Gitlab 검색 후 상단 4개 설치
- Docker 검색 후 상단 4개 설치

backend 브랜치 Merge 성공 시 자동으로 실행되는 shell script

```
cd a107
docker-compose build
docker-compose up -d
```

BackEnd Dockerfile

```
FROM nginx
RUN rm /etc/nginx/conf.d/default.conf
COPY ./conf/nginx.conf /etc/nginx/nginx.conf
COPY ./certificates /etc/letsencrypt
EXPOSE 80
EXPOSE 443
CMD ["nginx", "-g", "daemon off;"]
```

BackEnd docker-compose.yml

```
version: "3.7"

services:
  proxy-nginx:
    container_name: proxy-nginx
    build:
      context: .
      dockerfile: ./Dockerfile
    ports:
      - 80:80
      - 443:443
    networks:
```

```
      - a107_network
      - front
      - openvidu_default

  redis:
    container_name: redis
    image: redis:latest
    ports:
      - "6379:6379"
    restart: always
    networks:
      - a107_network

  mongodb:
    container_name: mongodb
    image: mongo
    restart: always
    ports:
      - "27017:27017"
    environment:
      MONGO_INITDB_ROOT_USERNAME: ssafy
      MONGO_INITDB_ROOT_PASSWORD: ssafy
      MONGO_INITDB_DATABASE: a107_mongodb
    volumes:
      - /data/mongodb:/data/db
    networks:
      - a107_network

  mysql:
    container_name: mysql
    image: mysql:8.0.31
    volumes:
      - a107-mysql-data:/var/lib/mysql
    restart: always
    ports:
      - "3306:3306"
    environment:
      MYSQL_ROOT_PASSWORD: ssafy
      MYSQL_DB: a107_mysql
      MYSQL_USER: ssafy
      MYSQL_PASSWORD: ssafy
    networks:
      - a107_network

volumes:
  a107-mysql-data:

networks:
  a107_network:
  front:
    external:
      name: a107-app_default
  openvidu_default:
    external:
      name: openvidu_default
```

frontend 브랜치 Merge 성공 시 실행되는 shell script

```
cd A107-front/a107-app
docker-compose build
docker-compose up -d
```

FrontEnd Dockerfile

```
FROM node:16.18.1-alpine as builder
WORKDIR "/app"
COPY package.json .
RUN npm install --legacy-peer-deps
COPY . .
ENV NODE_ENV production
RUN npm run build

FROM nginx
RUN mkdir /app
WORKDIR /app
RUN rm /etc/nginx/conf.d/default.conf
COPY ./nginx.conf /etc/nginx/conf.d
COPY --from=builder /app/build /app/build
EXPOSE 3000
CMD ["nginx", "-g", "daemon off;"]
```

FrontEnd docker-compose.yml

```
version: "3.7"

services:
  a107_frontend:
    container_name: react
    build:
      context: .
      dockerfile: ./Dockerfile
    ports:
      - "3000:3000"
    restart: always
```