

**A201 : ONSIKGO**

삼성SW청년아카데미 부울경캠퍼스 7기

공통프로젝트(6주; 2022.07.05 ~ 2022.08.19)

**포팅 매뉴얼**

담당 컨설턴트 : 최호근

장창완(팀장), 공지훈, 김가흔, 손효재, 이동근, 최지은

목차

[1. 프로젝트 개요 2](#_Toc111626188)

[2. 프로젝트 기술 스택 2](#_Toc111626189)

[3. 주요 환경 변수 3](#_Toc111626190)

[4. 도커 이미지 빌드 및 실행 4](#_Toc111626191)

[5. 배포 특이사항 5](#_Toc111626205)

[6. 외부 서비스 7](#_Toc111626257)

## 

## 프로젝트 개요

식품은 유통기한 내에 팔지 못하면 버려야 한다. 국내에서 하루 배출되는 식품 관련 쓰레기는 2만t이 넘는다. 음식 쓰레기를 수거·재활용할 때 배출되는 온실가스가 엄청나기 때문에 기후 위기와도 직결된다. 식품의 경우, 유통기한이라는 조건이 붙기 떄문에 이를 해결하기 위해서는 판매자와 소비자의 정확한 매칭이 필요하다.  
 온식고는 남는 음식 소개로 업자 매출은 증가, 고객 지출은 감소, 환경은 보호하는 것을 목표로 한다. 공급자는 당일 판매하지 못한 상품을 폐기하지 않고 판매하여 매출 손실을 줄인다. 소비자는 같은 식품을 저렴하게 구매할 수 있다. 이를 통해 공급자와 소비자 모두 가계 부담을 줄일 수 있다. 무엇보다 음식 쓰레기를 줄여 환경보호에 기여할 수 있다.

## 프로젝트 기술 스택

가. 이슈 관리 : Jira

나. 형상 관리 : Gitlab

다. 커뮤니케이션 : Notion, Mattermost

라. 개발 환경

1) OS : Windows 10

2) IDE

가) IntelliJ 2021.3.2

나) Visual Studio Code 1.70.1

다) UI/UX : Figma

3) Database :

가) MySQL 8.0.30

나) Redis 7.0.4

4) Server : AWS EC2 Ubuntu 20.04 LTS

5) Dev-Ops

가) Docker 20.10.17

나) Jenkins 2.60.3

마. 상세 사용

1) Frontend

가) HTML5, CSS3, JavaScript(ES6)

나) Vue 2.7.7, Vuex 3.6.2

다) Node.js 16.15.1

라) vuetify 2.6.0, bootstrap-vue 2.22.0

2) Backend

가) Spring boot 2.7.2

나) Open JDK 8

다) Gradle 7.5

## 주요 환경 변수

1. Frontend
2. Backend

# database

spring.datasource.url=DB주소

spring.datasource.username=DB 호스트 유저 이름

spring.datasource.password=DB 호스트 유저 비밀번호

# jwt

jwt.secret=jwt 시크릿 키

# s3

cloud.aws.credentials.access-key=발급받은 엑세스 키

cloud.aws.credentials.secret-key=발급받은 시크릿 키

cloud.aws.s3.bucket=s3 버킷 이름

# email

spring.mail.username=네이버 아이디

spring.mail.password=네이버 비밀번호

# redis

spring.redis.host=레디스 호스트 주소

spring.redis.port=레디스 포트 번호

spring.redis.password=레디스 비밀번호

# ssl

server.ssl.key-store-password=ssl 인증서 비밀번호

## 도커 이미지 빌드 및 실행

## Docker 설치

## sudo apt-get update

## sudo apt-get install \

## ca-certificates \

## curl \

## gnupg \

## lsb-release

## curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

## 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

1. Vue  
   docker build -t frontend .  
     
   docker run -d -p 80:80 -p 443:443   
   -v /home/ubuntu/certbot/conf:/etc/letsencrypt/   
   -v /home/ubuntu/certbot/www:/var/www/certbot   
   -v /home/ubuntu/nginx/conf/default.conf:/etc/nginx/conf.d/default.conf   
   --name=frontend frontend
2. Spring Boot  
   docker build -t backend .  
     
   docker run -p 8081:8080 -d -e TZ=Asia/Seoul --name=backend backend
3. Redis  
   docker pull redis  
     
   docker run -d --name redis -p 7000:6379 -v /etc/redis/redis.conf:/usr/local/etc/redis/redis.conf redis:latest --requirepass <password>
4. Mysql  
   docker pull mysql  
     
   docker run -d --name mysql -e MYSQL\_ROOT\_PASSWORD=<password> -p 3306:3306 mysql:latest

## 배포 특이사항

## Spring boot에 SSL 적용 1) Certbot container 생성 및 인증서 발급

## sudo mkdir certbot

## cd certbot

## sudo mkdir conf www logs

## ​

## sudo docker pull certbot/certbot

## sudo docker run -it --rm --name certbot -p 80:80 \

## -v "/home/ubuntu/certbot/conf:/etc/letsencrypt" \

## -v "/home/ubuntu/certbot/log:/var/log/letsencrypt" \

## -v "/home/ubuntu/certbot/www:/var/www/certbot" \

## certbot/certbot certonly

## 2) SSL인증서를 spring boot에서 필요한 형식(PKCS12)로 변환

## openssl pkcs12 -export -in fullchain.pem -inkey privkey.pem-out keystore.p12 -name tomcat -CAfile chain.pem -caname root

## 3) keystore.p12 파일을 /src/main/resources에 이동

## nginx SSL 설정 1) /home/ubuntu/nginx/conf/default.conf

## map $http\_upgrade $connection\_upgrade {

## default upgrade;

## '' close;

## }

## server {

## listen 80;

## server\_name i7e201.p.ssafy.io;

## location / {

## return 301 https://$host$request\_uri;

## }

## }

## server {

## listen 443 ssl;

## server\_name i7e201.p.ssafy.io;

## access\_log /var/log/nginx/access.log;

## error\_log /var/log/nginx/error.log;

## ssl\_certificate /etc/letsencrypt/live/i7e201.p.ssafy.io/fullchain.pem;

## ssl\_certificate\_key /etc/letsencrypt/live/i7e201.p.ssafy.io/privkey.pem;

## ssl\_protocols TLSv1 TLSv1.1 TLSv1.2 SSLv3;

## ssl\_ciphers ALL;

## location / {

## root /usr/share/nginx/html;

## index index.html index.htm

## proxy\_redirect off;

## charset utf-8;

## try\_files $uri $uri/ /index.html;

## proxy\_http\_version 1.1;

## proxy\_set\_header Upgrade $http\_upgrade;

## proxy\_set\_header Connection "upgrade";

## proxy\_set\_header Host $host;

## proxy\_set\_header X-Real-IP $remote\_addr;

## proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

## proxy\_set\_header X-Forwarded-Proto $scheme;

## proxy\_set\_header X-Nginx-Proxy true;

## }

## }

## 외부 서비스

### [카카오 로그인 기능](https://developers.kakao.com/docs/latest/ko/kakaologin/common)

### [네이버 로그인 기능](https://developers.naver.com/products/login/api/api.md)

### [AWS S3](https://aws.amazon.com/ko/s3/)

라) [채널 톡](https://channel.io/ko)  
마) [카카오 맵](https://apis.map.kakao.com/)