|  |
| --- |
|  |

|  |
| --- |
|  |
| **포팅 매뉴얼** |
|  |
|  |
|  |
|  |
|  |



목차

[I. 개요 2](#_Toc96072459)

[1. 프로젝트 개요 2](#_Toc96072460)

[2. 프로젝트 사용 도구 2](#_Toc96072461)

[3. 개발환경 2](#_Toc96072462)

[4. 외부 서비스 2](#_Toc96072463)

[5. Gitgnore 처리한 핵심 키들 3](#_Toc96072464)

[II. 빌드 3](#_Toc96072465)

[1. 환경변수 형태 3](#_Toc96072466)

[2. 빌드하기 5](#_Toc96072467)

[3. 배포하기 5](#_Toc96072468)

[4. 서비스 이용 방법 7](#_Toc96072469)

[가) 카카오 페이 7](#_Toc96072470)

[나) Firebase 실시간 알림 8](#_Toc96072471)

[다) Google Cloud Platform 8](#_Toc96072472)

# 개요

## 프로젝트 개요

## 프로젝트 사용 도구

이슈 관리 : JIRA

형상 관리 : Gitlab

커뮤니케이션 : Notion, Slack, Mattermost

디자인 : Figma

UCC : 모바비, 애프터이펙트, 프리미어

CI/CD : Jenkins

## 개발환경

VS Code : 1.75.1,

IntelliJ : 11.0.13+8-b1504.49 amd64

JVM : jdk1.8.0\_202

Node.js : 16.16.0

SERVER : AWS EC2 Ubuntu 20.04.5 LTS

DB : MySQL

## 외부 서비스

Firebase : fcm message foreground, background 시 알림 기능

ImPort : 코인 결제, pages/cash 에서 구현 내용 확인 가능

KakaoMap API : script 태그 index.html에서 확인가능, pages/map.js에서 구현 내용 확인 가능

## Gitgnore 처리한 핵심 키들

React : .env (최상단 위치)

Spring : application.yml,

(\src\main\resources 위치)

# 빌드

## 환경변수 형태

.env

REACT\_APP\_BASEURL= https://i8b107.p.ssafy.io/api

PUBLIC\_URL= https://i8b107.p.ssafy.io/api

// FIREBASE RTDB 프론트 키

REACT\_APP\_FIREBASE\_APIKEY= AIzaSyA12FoQ6AFA8jRk-1PrG9Vn7DnWHI44lmY

.application.yml

socket-server:

port: socket포트번호

server:

servlet:

context-path: /api

port: 8081

tomcat:

basedir: .

accesslog:

enabled: true

pattern: '%{yyyy-MM-dd HH:mm:ss}t\t%s\t%r\t%{User-Agent}i\t%{Referer}i\t%a\t%b'

spring:

datasource:

driver-class-name: com.mysql.cj.jdbc.Driver

url: jdbc:mysql://${datasource}/${schema}?serverTimezone=Asia/Seoul&characterEncoding=UTF-8

username: ${dbUser}

password: ${dbPwd}

hikari:

connection-timeout: 10000000

validation-timeout: 10000000

max-lifetime: 580000000

jpa:

properties:

hibernate:

globally\_quoted\_identifiers: 'true'

hibernate:

ddl-auto: update

use-new-id-generator-mappings: true

naming:

physical-strategy: org.hibernate.boot.model.naming.PhysicalNamingStrategyStandardImpl

show-sql: true

generate-ddl: true

devtools:

restart:

enabled: false

jackson:

property-naming-strategy: LOWER\_CAMEL\_CASE

main:

allow-circular-references: true

web:

resources:

add-mappings: false

jwt:

header:

access: Authorization

refresh: RefreshToken

type:

access: "Bearer "

refresh: "refresh-"

time:

access: 18000000

refresh: 6048000000

secret:

key: ${jwt\_secret\_key}

user:

url:

client: ${client\_url}

logout: /logout

admin: /api/admin

member: /api/member

marker: /api/marker

call: /api/call

room: /api/room

coin: /api/coin

role:

admin: ROLE\_ADMIN

member: ROLE\_MEMBER

marker: ROLE\_MARKER

call: ROLE\_CALL

room: ROLE\_ROOM

coin: ROLE\_COIN

permit:

all: '/\*\*'

session:

id: 'JSESSIONID'

cors:

method: 'POST, GET, PUT, PATCH, DELETE'

header: '\*'

pattern: '\*'

source:

pattern: '/\*\*'

cookie:

credential: true

logging:

level:

B107.server.meerkat: debug

org:

hibernate:

SQL: DEBUG

type:

descriptor:

sql:

BasicBinder: TRACE

## 빌드하기

1) Front-react

npm i

npm run build

2) Back-spring

Maven - Compile 실행

Maven – Package 실행

## 배포하기

## Docker-compose

## version: "3"

## services:

## jenkins:

## image: jenkins/jenkins:lts

## user: root

## volumes:

## - /jenkins:/var/jenkins\_home

## - /var/run/docker.sock:/var/run/docker.sock ubuntu/jenkins\_home"

## ports:

## - 8082:8080

## environment:

## - TZ=Asia/Seoul

## privileged: true

## restart: "unless-stopped"

## mysql:

## image: mysql:8.0

## container\_name: mysql

## ports:

## - 3306:3306 # HOST:CONTAINER

## enviroment:

## - MYSQL\_ROOT\_PASSWORD='!rhdxhdB107'

## - TZ=Asia/Seoul

## command:

## - --character-set-server=utf8mb4

## - --collation-server=utf8mb4\_unicode\_ci

## volumes:

## - ./data:/var/lib/mysql

## api:

## build:

## context: ./server

## dockerfile: Dockerfile

## volumes:

## - /jenkins/workspace/common-B107-meerkat-develop:/var/jenkins\_home/workspace/common-B107-meerkat-develop

## - /var/run/docker.sock:/var/run/docker.sock

## - /etc/localtime:/etc/localtime:ro

## - /usr/share/zoneinfo/Asia/Seoul:/etc/timezone:ro

## - /logs:/logs

## environment:

## datasource: "i8b107.p.ssafy.io"

## dbUser: "admin"

## dbPwd: "!ssafyB107"

## jwt\_secret\_key: "ssafy8thCommonB107"

## schema: "devdb"

## ports:

## - 8081:8081

## - 8085:8085

## networks:

## - web

## client:

## build:

## context: ./client

## dockerfile: Dockerfile

## volumes:

## - /jenkins/workspace/common-B107-meerkat-develop:/var/jenkins\_home/workspace/common-B107-meerkat-develop

## - /var/run/docker.sock:/var/run/docker.sock

## environment:

## TZ: "Asia/Seoul"

## ports:

## - 3000:3000

## networks:

## - web

## networks:

## web:

## driver: bridge

## 

Nginx 설정

server {

server\_name i8b107.p.ssafy.io;

location / {

proxy\_pass http://127.0.0.1:3000;

proxy\_set\_header Host $host;

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

}

location /api {

proxy\_pass http://127.0.0.1:8081;

proxy\_set\_header Host $host;

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

}

location /socket.io/ {

proxy\_pass http://localhost:8085;

proxy\_redirect off;

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\_buffer\_size 128k;

proxy\_buffers 4 256k;

proxy\_busy\_buffers\_size 256k;

}

listen 443 ssl; # managed by Certbot

ssl\_certificate /etc/letsencrypt/live/i8b107.p.ssafy.io/fullchain.pem; # managed by Certbot

ssl\_certificate\_key /etc/letsencrypt/live/i8b107.p.ssafy.io/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

access\_log /logs/access.log;

error\_log /logs/error.log;

}

server {

if ($host = i8b107.p.ssafy.io) {

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

} # managed by Certbot

listen 80;

server\_name i8b107.p.ssafy.io;

return 404; # managed by Certbot

}

이후 sudo service nginx start

## 서비스 이용 방법

### 카카오 맵

준비 : 카카오페이 API 등록

- 내 애플리케이션 → 애플리케이션 추가하기

- 들어갈 경우 Admin키가 있음

- 플랫폼 등록

### Firebase 실시간 알림

1. Firebase 콘솔 프로젝트 만들기 (GCP에 프로젝트 있을 경우 그거 선택)

2. 설정 → 프로젝트 설정 → 서비스 계정 → 새 비공개 키 생성

3. 프로젝트 설정 -> 클라우드 메시징 -> 구글 클라우드 콘솔에서 API 관리

-> 사용 -> 서버키 확인

3. 프로젝트 생성 후 -> 앱 추가 -> 플랫폼 선택(Web) -> 앱 등록

4. 내 앱 -> SDK 설정 및 구성 ( npm or CDN )