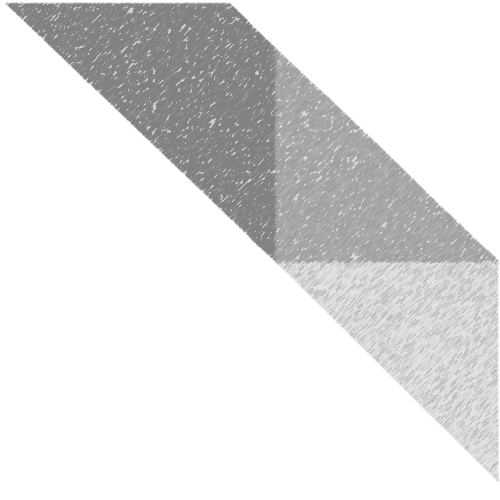
|  |
| --- |
|  |



|  |
| --- |
| **우리들의 추억 카드 추카** |
| **포팅 메뉴얼** |
|  |
|  |
|  |
|  |
|  |

ㅊㅋ포팅매뉴얼

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. Gitignore 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 로그인 7](#_Toc96072470)

[나) Firebase 실시간 알림 8](#_Toc96072471)

[다) S3 8](#_Toc96072472)

# 개요

## 프로젝트 개요

## 목표: 졸업, 생일, 스승의 날 등 특별한 날에 지난 시간을 추억하는 서비스를 제공하고, 펀딩을 통해 마음을 전달합니다.

졸업식, 생일 등 단체로 축하해줄 특별한 날이 다가오길 기대하며 서로의 마음을 롤링페이퍼로 전달합니다.

곧 싸피 수료가 다가오는데요, 이런 특별한 날에 각 반이나 캠퍼스 단위로 이벤트를 생성하고 그 안에서 서로에게 응원의 메시지와 선물을 할 수 있습니다.

## 프로젝트 사용 도구

이슈 관리 : JIRA

형상 관리 : Gitlab

커뮤니케이션 : Notion, Mattermost

BE : SpringBoot3, Java17, JPA

FE : React(18.2.0)

SERVER : AWS EC2 Ubuntu 20.04.3 LTS

DB : MySQL (8.0.36), MongoDB

## 외부 서비스

Kakao OAuth : ConfigRepository 관리(주소: <https://github.com/lhh9799/C107_config> 권한 필요)

Firebase FCM (BE) : chuka-fcm.json에 설정 내용 있음

Redis cloud : ConfigRepository 에 설정 내용 있음

Amazon S3 (BE) : ConfigRepository 에 설정 내용 있음

## Gitgnore 처리한 핵심 키들

React : .env (최상단 위치)

SpringBoot3 : ConfigRepository 에 설정 내용 있음

(<https://github.com/lhh9799/C107_config> 권한 필요)

# 빌드

## 환경변수 형태

ConfigRepository 에 설정

# Gateway

# user

spring.cloud.gateway.routes[0].id=user

spring.cloud.gateway.routes[0].predicates[0].name=Path

spring.cloud.gateway.routes[0].predicates[0].args.pattern=/api/v1/users/\*\*

spring.cloud.gateway.routes[0].uri=http://ec2-43-203-200-59.ap-northeast-2.compute.amazonaws.com:8081

spring.cloud.gateway.routes[0].filters[0]=AuthenticationFilter

# auth

spring.cloud.gateway.routes[1].id=auth

spring.cloud.gateway.routes[1].predicates[0].name=Path

spring.cloud.gateway.routes[1].predicates[0].args.pattern=/api/v1/auth/\*\*

spring.cloud.gateway.routes[1].uri=http://ec2-43-203-200-59.ap-northeast-2.compute.amazonaws.com:8081

# notification

spring.cloud.gateway.routes[2].id=notification

spring.cloud.gateway.routes[2].predicates[0].name=Path

spring.cloud.gateway.routes[2].predicates[0].args.pattern=/api/v1/notifications/\*\*

spring.cloud.gateway.routes[2].filters[0]=AuthenticationFilter

spring.cloud.gateway.routes[2].uri=http://ec2-43-203-200-59.ap-northeast-2.compute.amazonaws.com:8082

# funding

spring.cloud.gateway.routes[3].id=funding

spring.cloud.gateway.routes[3].predicates[0].name=Path

spring.cloud.gateway.routes[3].predicates[0].args.pattern=/api/v1/fundings/\*\*

spring.cloud.gateway.routes[3].uri=http://k10c107.p.ssafy.io:8083

spring.cloud.gateway.routes[3].filters[0]=AuthenticationUserOrNotFilter

# crawling

spring.cloud.gateway.routes[4].id=crawling

spring.cloud.gateway.routes[4].predicates[0].name=Path

spring.cloud.gateway.routes[4].predicates[0].args.pattern=/api/v1/crawling/\*\*

spring.cloud.gateway.routes[4].uri=http://k10c107.p.ssafy.io:8085

# review

spring.cloud.gateway.routes[5].id=review

spring.cloud.gateway.routes[5].predicates[0].name=Path

spring.cloud.gateway.routes[5].predicates[0].args.pattern=/api/v1/reviews/\*\*

spring.cloud.gateway.routes[5].uri=http://k10c107.p.ssafy.io:8084

# event

spring.cloud.gateway.routes[6].id=event

spring.cloud.gateway.routes[6].predicates[0].name=Path

spring.cloud.gateway.routes[6].predicates[0].args.pattern=/api/v1/events/\*\*

spring.cloud.gateway.routes[6].uri=http://k10c107.p.ssafy.io:8084

spring.cloud.gateway.routes[6].filters[0]=AuthenticationUserOrNotFilter

# 욕설 필터링

bad-words-url=https://s3.ap-northeast-2.amazonaws.com/my.first.mela.sss.bucket/badwords.txt

# RabbitMQ

spring.rabbitmq.host=k10c107.p.ssafy.io

spring.rabbitmq.port=5672

spring.rabbitmq.username=guest

spring.rabbitmq.password=guest

## RabbitMQ properties for message queue

rabbitmq.user\_to\_notification.queue=user\_to\_notification.queue

rabbitmq.user\_to\_notification.exchange=user\_to\_notification.topic

rabbitmq.event\_to\_notification.exchange=event\_to\_notification.topic

rabbitmq.event\_to\_notification.queue=event\_to\_notification.queue

rabbitmq.product.queue=product.queue

rabbitmq.product.exchange=product.topic

rabbitmq.crawling.queue=crawling.queue

rabbitmq.crawling.exchange=crawling.topic

rabbitmq.funding\_to\_notification.queue=funding\_to\_notification.queue

rabbitmq.funding\_to\_notification.exchange=funding\_to\_notification.topic

# 결제 PG키

portone.rest.api.key=3517761515155163

portone.rest.api.secert.key=qrMi0GMdbknZ4iYPeCvUaDwfzc66BZEDt0TW9SMLtDzFtAwmm9VrV8l8VEglbkv7p6e5ryk81Zb4sIms

# REDIS

spring.data.redis.host=redis-10500.c74.us-east-1-4.ec2.cloud.redislabs.com

spring.data.redis.port=10500

spring.data.redis.password=qUvoSAD6uFSmsHxmWKPNorjIsO6x4ExM

# Prometheus

management.endpoint.metrics.enabled=true

management.endpoints.web.exposure.include=prometheus

management.endpoint.prometheus.enabled=true

management.metrics.export.prometheus.enabled=true

management.endpoints.web.base-path=/actuator/user

management.endpoints.web.base-path=/actuator/notification

management.endpoints.web.base-path=/actuator/event

management.endpoints.web.base-path=/actuator/funding

# eureka

eureka.client.register-with-eureka=true

eureka.client.fetch-registry=true

eureka.client.service-url.defaultZone=http://luckyseven:cnzkcnzk7!@k10c107.p.ssafy.io:8761/eureka

# UserDB

spring.datasource.url=jdbc:mysql://ec2-43-203-200-59.ap-northeast-2.compute.amazonaws.com:3306/user

spring.datasource.username=root

spring.datasource.password=luckyseven777!

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# NotificationDB

spring.data.mongodb.host=k10c107.p.ssafy.io

spring.data.mongodb.port=27017

spring.data.mongodb.authentication-database=admin

spring.data.mongodb.username=likpark

spring.data.mongodb.password=luckyseven777!

spring.data.mongodb.database=notification

# EventDB

## Event mongoDB

spring.data.mongodb.host=k10c107.p.ssafy.io

spring.data.mongodb.port=27018

spring.data.mongodb.authentication-database=admin

spring.data.mongodb.username=luckyseven

spring.data.mongodb.password=cnzkcnzk7!

spring.data.mongodb.database=rollingpaper

mongodb.connection-string=mongodb://luckyseven:cnzkcnzk7!@k10c107.p.ssafy.io:27018/

## Event MySQL

spring.datasource.url=jdbc:mysql://k10c107.p.ssafy.io:3308/event

spring.datasource.username=root

spring.datasource.password=luckyseven777!

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# FundingDB

spring.datasource.url=jdbc:mysql://k10c107.p.ssafy.io:3307/funding

spring.datasource.username=root

spring.datasource.password=luckyseven777!

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# JWT 시크릿 키 / DB 암호화 복호화 키

secret-key=0b3fa07a7a9064b5153cf7390f3d4a568c481ff33bee2641b921473545b2b812

# kakao api

kakao.api.admin.key=18db6d2706f5a8a7147f7ba69bc228f3

kakao.api.rest.key=7d732bca71d1f9e25d1cb971f592a663

kakao.api.oauth.token=https://kauth.kakao.com/oauth/token

kakao.api.user.unlink=https://kapi.kakao.com/v1/user/unlink

kakao.api.user.me=https://kapi.kakao.com/v2/user/me

kakao.api.redirect=https://chuka.kr/login/oauth2

# Amazon S3

cloud.aws.credentials.access-key=AKIAVRUVTBJNLKYQSAH2

cloud.aws.credentials.secret-key=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

cloud.aws.region.static=ap-northeast-2

cloud.aws.s3.bucket=chu.ka

## 빌드하기

1) Front

npm i

npm run dev

2) Back-spring

Clean 실행

Build실행

Bootjar 실행

## 배포하기

## # 기본 ec2서버

server {

client\_max\_body\_size 10M;

root /var/www/html;

index index.html index.htm index.nginx-debian.html;

server\_name chuka.kr;

location / {

proxy\_pass http://localhost:5000;

}

location /api {

proxy\_pass http://localhost:8080;

proxy\_http\_version 1.1;

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-Forwarded-Host $host;

proxy\_set\_header X-Forwarded-Port $server\_port;

}

listen [::]:443 ssl ipv6only=on; # managed by Certbot

listen 443 ssl; # managed by Certbot

ssl\_certificate /etc/letsencrypt/live/chuka.kr/fullchain.pem; # managed by Certbot

ssl\_certificate\_key /etc/letsencrypt/live/chuka.kr/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

}

server {

if ($host = chuka.kr) {

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

} # managed by Certbot

listen 80 default\_server;

listen [::]:80 default\_server;

server\_name chuka.kr;

return 404; # managed by Certbot

}

이후 sudo systemctl restart nginx

# 추가 ec2서버

server {

root /var/www/html;

index index.html index.htm index.nginx-debian.html;

server\_name mynameis.site;

location / {

try\_files $uri $uri/ =404;

}

location /api/v1/auth {

# limit\_req zone=ddos\_req;

proxy\_pass http://localhost:8081;

proxy\_http\_version 1.1;

proxy\_set\_header Connection "";

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-Forwarded-Host $host;

proxy\_set\_header X-Forwarded-Port $server\_port;

}

location /api/v1/users {

# limit\_req zone=ddos\_req;

proxy\_pass http://localhost:8081;

proxy\_http\_version 1.1;

proxy\_set\_header Connection "";

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-Forwarded-Host $host;

proxy\_set\_header X-Forwarded-Port $server\_port;

}

location /api/v1/notifications {

# limit\_req zone=ddos\_req;

proxy\_pass http://localhost:8082;

proxy\_http\_version 1.1;

proxy\_set\_header Connection "";

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-Forwarded-Host $host;

proxy\_set\_header X-Forwarded-Port $server\_port;

}

listen [::]:443 ssl ipv6only=on; # managed by Certbot

listen 443 ssl; # managed by Certbot

ssl\_certificate /etc/letsencrypt/live/mynameis.site/fullchain.pem; # managed by Certbot

ssl\_certificate\_key /etc/letsencrypt/live/mynameis.site/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

}

server {

if ($host = mynameis.site) {

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

} # managed by Certbot

listen 80 default\_server;

listen [::]:80 default\_server;

server\_name mynameis.site;

return 404; # managed by Certbot

}

이후 sudo systemctl restart nginx

## 서비스 이용 방법

### 카카오 로그인

준비 : 카카오로그인 API 등록

- 내 애플리케이션 → 애플리케이션 추가하기

- 테스트 앱 등록 → 필수항목 추가를 위해

- 플랫폼 등록 → 앱

- 카카오 로그인 설정

- Redirect URI 설정 → kakao.api.redirect 환경변수와 동일한 값 등록

### Firebase → FCM

1. Firebase 프로젝트 생성

2. 설정 → 프로젝트 설정 → 서비스 계정 → 새 비공개 키 생성

3. realtimedatabase 생성 후 주소 확인

### Amazon S3 이미지 저장

1. Amazon S3 관리 콘솔 접속
2. 버킷 만들기
3. AWS 보안 자격 증명 (IAM)
   1. 사용자 추가
   2. 권한 정책 (AmazonS3FullAccess) 추가
4. Gradle Dependency 추가
5. S3Config 클래스 작성

### Portone 결제

1. Portone 관리 콘솔 접속
2. 결제 연동
3. 채널 추가
   1. 연동 모드 “테스트”로 선택
   2. 결제대행사 “카카오페이” 선택
   3. 결제 모듈 “간편결제 일반/정기결제 v1 v2”선택
   4. PG상점아이디 “카카오페이 일반결제(TC0ONETIME)” 선택
4. FRONTEN의 .ENV 파일에 VITE\_PAYMENT\_KEY에 고객사 식별코드 입력
5. Gradle Dependency 추가