|  |
| --- |
|  |





|  |
| --- |
| **I DON CARE ~아이 돈 케어** |
| **포팅 메뉴얼** |
|  |
|  |
|  |
|  |
|  |

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 핵 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)

# 개요

## 프로젝트 개요

금융은 우리 생활의 필수 요소입니다. 실제로 알파세대는 금융과 경제 교육이 주요 과목만큼 중요하다고 인식하고 있습니다. 그러나 현실에서는 연령 제한, 용돈 부족, 금융용어의 난이도 등으로 이런 인식이 실천으로 이어지지 못하고 있습니다. 그래서 저희는 '아이돈케어'라는 서비스를 만들게 되었습니다. '아이돈케어'는 알파세대와 부모가 함께 사용하는 서비스로, 용돈 관리와 경제 교육을 동시에 할 수 있게 해줍니다. 자녀와 부모 모두 휴대폰에 앱을 설치하면, 주고 받는 용돈을 기반으로 자녀가 스스로 돈을 모으고 쓰며 불리고 나누기까지 가능합니다. 거기에 활동 보고서와 거래 내역 공유 기능까지! 이제 자녀의 경제 교육도 '아이돈케어'가 함께 합니다.

길게 말하지 않아도 압니다, 긴 설명보다 직접 체험이 중요하니까요! 아이의 돈을 신경쓰지 않아도 되는! '아이 돈 케어', 지금 바로 시작해보세요!

## 프로젝트 사용 도구

이슈 관리 : JIRA

형상 관리 : Gitlab

커뮤니케이션 : Notion, Mattermost

디자인 : Figma

UCC : 모바비, 애프터이펙트, 프리미어

CI/CD : Jenkins

## 개발환경

VS Code : 1.79.2

IntelliJ : ideaIC-2023.1.3.win

JVM : 16.0.1 (스프링은 11로 빌드)

Node.js : 18.0.1

SERVER : AWS EC2 Ubuntu 20.04.3 LTS

DB : MySQL

## 외부 서비스

Kakao Login: application.yml에 해당 내용 있음

## Gitgnore 처리한 핵심 키들

React : .env

Spring : application.yml,

# 빌드

## 환경변수 형태

.env

VUE\_APP\_KAKAO\_API\_KEY={카카오 로그인 api key}

.application.properties

# Server Setting #

server.port={포트번호: Number}

# DB Setting #

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

spring.datasource.url=jdbc:mysql://{호스트: String}:{포트번호: Number}/{데이터베이스 이름: String}?serverTimezone=Asia/Seoul&characterEncoding=UTF-8

spring.datasource.username={DB 아이디: String}

spring.datasource.password={DB 패스워드: String}

spring.main.allow-circular-references=true

# DB Logging #

# Development Level #

decorator.datasource.p6spy.enable-logging=true

# JPQ Setting #

# create -> create-drop -> update -> none

spring.jpa.hibernate.ddl-auto=update

spring.main.allow-bean-definition-overriding=true

# Batch Loading #

spring.jpa.properties.hibernate.default\_batch\_fetch\_size=100

# Swagger #

springdoc.default-consumes-media-type=application/json;charset=UTF-8

springdoc.default-produces-media-type=application/json;charset=UTF-8

springdoc.swagger-ui.path=/

springdoc.swagger-ui.disable-swagger-default-url=true

springdoc.swagger-ui.display-request-duration=true

springdoc.swagger-ui.operations-sorter=alpha

# Redis Setting #

redis.host={레디스 호스트: String}

redis.port={레디스 포트: Number}

# JWT Setting #

jwt.secret={시크릿 키: String}

jwt.access-expiration-time=43200000

jwt.refresh-expiration-time=604800000

# Kakao Login Setting #

kakao.rest-api-key={카카오 Rest Api Key : String}

kakao.client-secret={카카오 로그인 Secret key : String}

# externnal

openbank.url=https://port-0-openbankapi-iciy2almk8xusg.sel5.cloudtype.app

springsecurity.secret.key={시크릿 키 : String}

springsecurity.secret.salt={salt 값 : String}

# Open Banking Request Setting

idontcare.account=000000000000

idontcare.bankcode=41

## 빌드하기

1) Front

npm i

npm run build

2) Back-spring

Gradle 실행

Bootjar 실행

## 배포하기

Nginx 설정

server{

server\_name j9d209.p.ssafy.io;

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

listen 80;

return 404;

}

server{

server\_name j9d209.p.ssafy.io;

location / {

proxy\_pass http://127.0.0.1:8081$request\_uri;

}

listen 9081 ssl;

ssl\_certificate /etc/letsencrypt/live/j9d209.p.ssafy.io/fullchain.pem; # managed by Certbot

ssl\_certificate\_key /etc/letsencrypt/live/j9d209.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

}

server{

server\_name j9d209.p.ssafy.io;

location / {

proxy\_pass http://127.0.0.1:8082$request\_uri;

proxy\_set\_header HOST $http\_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;

proxy\_redirect off;

charset utf-8;

proxy\_http\_version 1.1;

proxy\_set\_header Upgrade $http\_upgrade;

proxy\_set\_header Connection "upgrade";

}

listen 9082 ssl;

ssl\_certificate /etc/letsencrypt/live/j9d209.p.ssafy.io/fullchain.pem; # managed by Certbot

ssl\_certificate\_key /etc/letsencrypt/live/j9d209.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

}

server{

server\_name j9d209.p.ssafy.io;

location / {

proxy\_pass http://127.0.0.1:10000$request\_uri;

proxy\_set\_header HOST $http\_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;

proxy\_redirect off;

charset utf-8;

proxy\_http\_version 1.1;

proxy\_set\_header Upgrade $http\_upgrade;

proxy\_set\_header Connection "upgrade";

}

listen 5173 ssl;

ssl\_certificate /etc/letsencrypt/live/j9d209.p.ssafy.io/fullchain.pem; # managed by Certbot

ssl\_certificate\_key /etc/letsencrypt/live/j9d209.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

}

server{

server\_name j9d209.p.ssafy.io;

location / {

proxy\_pass http://127.0.0.1:3000;

}

listen 443 ssl;

ssl\_certificate /etc/letsencrypt/live/j9d209.p.ssafy.io/fullchain.pem; # managed by Certbot

ssl\_certificate\_key /etc/letsencrypt/live/j9d209.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

}

이후 sudo service nginx start

## 서비스 이용 방법

### 카카오

준비 ; 카카오 로그인 API 등록

- 내 애플리케이션 → 애플리케이션 추가하기

- 들어갈 경우 Admin키가 있음 → 보안을 위해 yml 파일에 추가(gitgnore 했는지 확인!)

- 플랫폼 등록 ⇒ 웹 (적는것은 프론트!)