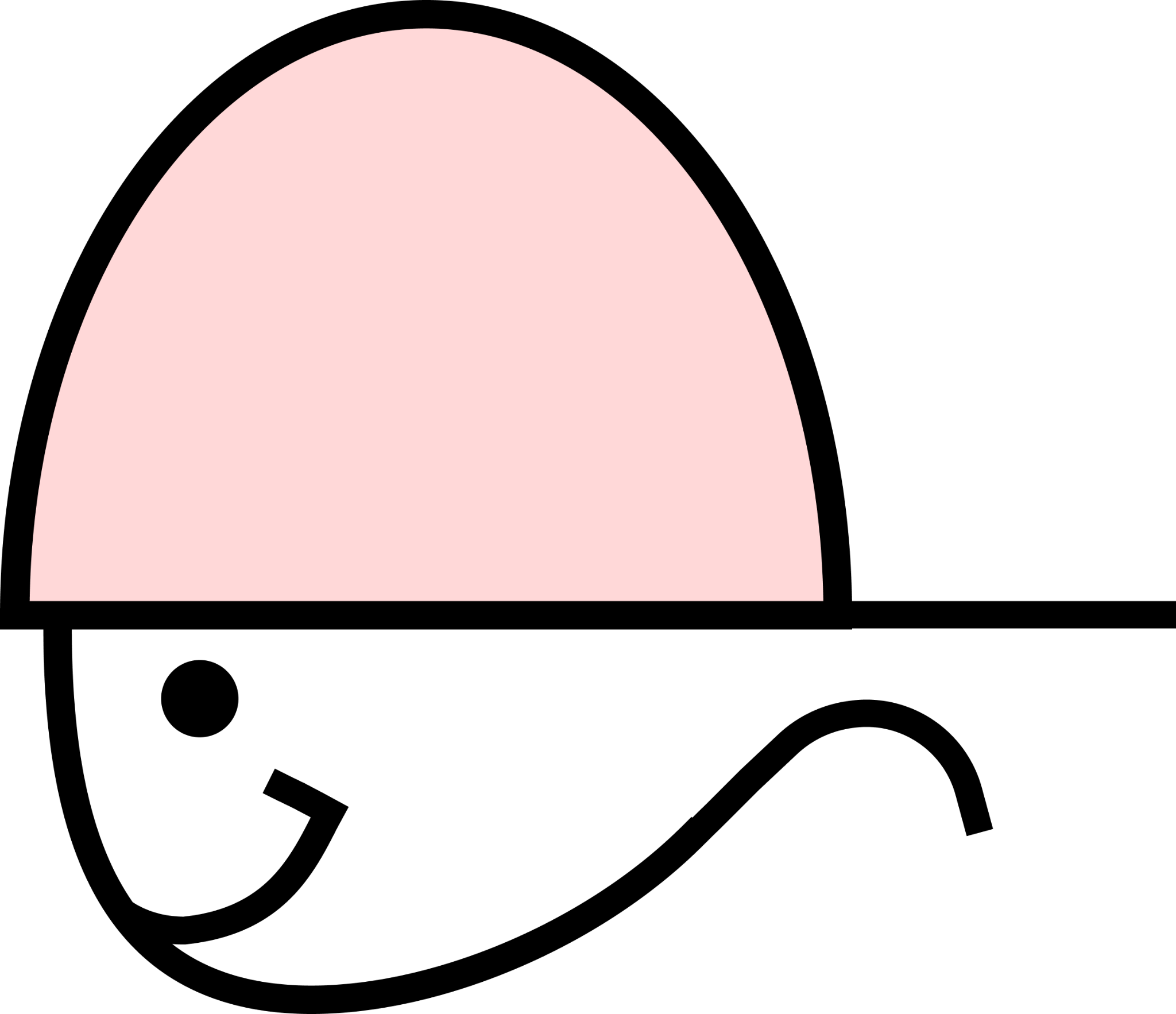
|  |
| --- |



| **아이와의 추억을 캡슐에** |
| --- |
| **포팅 메뉴얼** |
|  |
|  |
|  |
|  |
|  |



목차

[**I. 개요**](#_heading=h.1fob9te) **2**

[**1. 프로젝트 개요**](#_heading=h.3znysh7) **2**

[**2. 프로젝트 사용 도구**](#_heading=h.2et92p0) **2**

[**3. 개발환경**](#_heading=h.tyjcwt) **2**

[**4. 외부 서비스**](#_heading=h.3dy6vkm) **3**

[**5. Gitgnore 처리한 핵심 키들**](#_heading=h.1t3h5sf) **3**

[**II. 빌드**](#_heading=h.4d34og8) **3**

[**1. 환경변수 형태**](#_heading=h.2s8eyo1) **3**

[**2. 빌드하기**](#_heading=h.17dp8vu) **5**

[**3. 배포하기**](#_heading=h.3rdcrjn) **6**

[**4. 서비스 이용 방법**](#_heading=h.26in1rg)**8**

**가**[**) S3 버킷**](#_heading=h.1ksv4uv) **8**

# 개요

## 프로젝트 개요

아이들과 함께할 문화생활을 찾기 귀찮고, 찾을 여유가 없는 부모들에게 아이들과 함께 할 수 있는 문화생활을 추천해준다. 추천해주는 문화생활로 코스를 만들 수 있고, 아이들과의 추억을 저장해서 조부모님, 친구들과 공유를 할 수 있게 한다.

## 프로젝트 사용 도구

이슈 관리 : JIRA

형상 관리 : Gitlab

커뮤니케이션 : Notion, Mattermost

디자인 : Figma

UCC : 모바비

CI/CD : Jenkins

가상 머신 : Docker

## 개발환경

VS Code : 1.80.0

IntelliJ : 2023.1.3

PyCharm 2023.2

JVM : 11

Flutter : 3.13.2

SERVER : AWS EC2 Ubuntu 20.04.3 LTS, Nginx 1.25.1

SQL : MySql 8.0.34

NoSQL : MongoDB 6.0.8

## 외부 서비스

GoogleLoginAPI : application.yml에 해당 내용 있음

DynamoDB : 협업기반 필터링을 사용한 추천을 사용하기 위해 사용하는 클라우드 저장소

amazon S3 : 이미지 저장을 위한 클라우드 저장소

## Gitgnore 처리한 핵심 키들

Spring : .idea (인텔리제이 사용자 설정 폴더), application-dev.yml (키들을 저장한 파일)

FastApi : .env (키들을 저장한 파일)

Flutter : .env (키들을 저장한 파일)

# 빌드

빌드는 jenkins를 통해서 진행함

## 환경변수 형태

.application.yml

# MongoDB

mongodb:

host: [호스트 도메인]

port: [포트 번호]

database: [사용할 DB]

username: [사용할 UserName]

password: [UserName에 대한 비밀 번호]

authentication-database: [인증 DB]

# MySQL

datasource:

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

url: jdbc:mysql://[도메인이름]:3306/[db스키마이름]?serverTimezone=UTC&characterEncoding=UTF-8

username: [UserName]

password: [UserName에 대한 비밀번호]

# Security OAuth

security:

oauth2:

client:

registration:

kakao:

client-id: [클라이언트 아이디]

client-secret: [클라이언트 secret 키]

redirect-uri: [카카오 로그인 시 redirect-uri]

authorization-grant-type:

client-authentication-method:

client-name:

scope: [사용자 동의 항목]

provider:

kakao:

authorization-uri:

token-uri:

user-info-uri:

user-name-attribute:

# JWT 시크릿 키

jwt:

issuer: CHAEUM

secret\_key: [시크릿\_키]

# S3 버킷

cloud:

aws:

s3:

bucket: [버킷 이름]

credentials:

access-key: [액세스를 위해 만든 사용자 액세스 키]

secret-key: [액세스를 위해 만든 사용자 시크릿 키]

region:

static: ap-northeast-2

auto: false

stack:

auto: false

## 빌드하기

1) Back Spring - pipeline

stage('BE-Build') {

steps {

dir("./backend") {

sh '''

cp /var/jenkins\_home/workspace/docker/application-dev.yml ./src/main/resources

chmod +x ./gradlew

./gradlew clean build

mv ./build/libs/youkids-0.0.1-SNAPSHOT.jar /var/jenkins\_home/workspace/Youkids

cp /var/jenkins\_home/workspace/docker/Dockerfile /var/jenkins\_home/workspace/Youkids

'''

}

}

2) Back FastAPI - pipeline

stage('Docker Image Build') {

steps {

sh '''

docker stop youkidsfast || true

docker rm youkidsfast || true

docker build -t youkidsfast:v1 .

'''

}

}

## 배포하기

1. Nginx 설정

events {

worker\_connections 4096;

}

http {

upstream backend {

server j9a604.p.ssafy.io:8080;

}

upstream fastapi {

server j9a604.p.ssafy.io:8081;

}

server {

listen 80;

listen [::]:80;

server\_name j9a604.p.ssafy.io; # 등록한 도메인으로 변경

location /.well-known/acme-challenge/ {

root /var/www/certbot;

}

location / {

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

}

}

server {

listen 443 ssl;

ssl\_certificate /etc/letsencrypt/live/j9a604.p.ssafy.io/fullchain.pem; # managed by Certbot

ssl\_certificate\_key /etc/letsencrypt/live/j9a604.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\_name j9a604.p.ssafy.io;

location / {

root html;

index index.html index.htm;

}

location /api {

rewrite ^/api(/.\*)$ $1 break;

proxy\_pass http://backend;

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;

}

location /fastapi {

rewrite ^/fastapi(/.\*)$ $1 break;

proxy\_pass http://fastapi;

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;

}

}

}

1. Back Spring - jenkins pipeline

stage('Docker Image Build') {

steps {

sh '''

docker stop youkids || true

docker rm youkids || true

docker build -t youkids:v1 .

'''

}

}

stage('Docker Container') {

steps {

sh '''

docker run -d -p 8080:8080 --name youkids youkids:v1

'''

}

}

1. Back FastAPI- jenkins pipeline

stage('Docker Container') {

steps {

sh '''

docker run -d -p 8081:8000 --name youkidsfast youkidsfast:v1

'''

}

}

## 서비스 이용 방법

### S3 버킷

1. aws S3 접속 후 새 버킷 만들기

→ 버킷만들기

2. 브라우저가 이동될텐데 버킷 이름 지정하고 만들기

→ 버킷 이름은 aws 클라우드 상에서 유일해야함

→ 객체 소유권 : ACL 비활성화됨

→ 버킷의 퍼블릭 액세스 차단 모두 해제 후 경고 체크

→ 버전 관리 비활성화

3. 사용자 생성

→ 오른쪽 아이디 클릭 후 보안 자격 증명 클릭

→ 사용자 클릭 후 사용자 생성, 이름 생성

→ 직접 정책 연결 : AmazonS3FullAccess 클릭 및 사용자 생성 완료

4. Springboot 연결을 위한 AccessKey 생성

→ 액세스 키 활성화

→ → 액세스 키는 항상 볼 수 있지만, 시크릿 키는 한 번 넘어가면 사라지므로 csv 다운로드 혹은 따로 저장해두기

5. Application.yml

Application.yml에 적힌대로 cloud:s3 … 설정