|  |
| --- |
|  |

|  |
| --- |
| **너의 일상을 채워** |
| **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) BARD API**](#_heading=h.lnxbz9) **8**

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

# 개요

## 프로젝트 개요

요즘에는 인스타나 트위터를 통해서 활동을 기록하고 공유하는 문화가 자연스럽다. 또한 다른 사람이 공유한 게시글, 피드를 보며 자극을 받는 경우도 있다. 이러한 사람들을 위해서 동기부여 앱을 만들려고 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

JVM : 17

Node.js : 18.16.1

SERVER : AWS EC2 Ubuntu 20.04.3 LTS, Nginx 1.25.1

SQL : MySql 8.0.34

NoSQL : MongoDB 6.0.8

## 외부 서비스

KakaoLoginAPI : application.yml에 해당 내용 있음

GoogleBardAPI : 구글에서 제공하는 Bard API를 사용함

amazon S3 : 이미지 저장을 위한 클라우드 저장소

## Gitgnore 처리한 핵심 키들

Spring : .idea (인텔리제이 사용자 설정 폴더)

# 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) Front

젠킨스에 적히는 pipeline

stage('FE-build') {

steps {

dir("./FE/chaeum-frontend") {

nodejs(nodeJSInstallationName: 'NodeJS18.16.1') {

sh 'npm install && npm run build'

}

}

}

}

2) Back

젠킨스에 적히는 pipeline

stage('BE-Build') {

steps {

dir("./BE/chaeum-backend") {

sh '''

chmod +x ./gradlew

./gradlew clean build

'''

}

}

}

## 배포하기

1. Nginx 설정

server {

listen 80;

server\_name i9a810.p.ssafy.io;

return 301 https://i9a810.p.ssafy.io$request\_uri;

}

server {

listen 443 ssl;

server\_name i9a810.p.ssafy.io;

location / {

root /etc/nginx/build/build;

index index.html;

}

location /api/ {

proxy\_pass http://i9a810.p.ssafy.io:8080;

}

ssl\_certificate /etc/letsencrypt/live/i9a810.p.ssafy.io/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/i9a810.p.ssafy.io/privkey.pem;

}

1. Front - jenkins pipeline

stage('Deploy') {

steps {

sshagent(credentials: ['aws\_key']) {

sh '''

pwd

chmod +x ./frontend/deploy.sh

ssh -t -t ubuntu@i9a810.p.ssafy.io /home/ubuntu/jenkins/jenkins1/workspace/euinyunTestFE/frontend/deploy.sh

'''

}

}

}

1. Front - deploy.sh

도커위에 빌드가 되길 원하는 폴더랑 바인드 마운트를 한 nginx를 올린 후, 빌드가 될 주소에 build.tar 넣기

mv ./jenkins/jenkins1/workspace/euinyunTestFE/FE/chaeum-frontend/build.tar /home/ubuntu/nginx/build tar -xvf /home/ubuntu/nginx/build/build.tar -C /home/ubuntu/nginx/build rm -rf /home/ubuntu/nginx/build/build.tar sudo docker restart webserver

1. Back - jenkins pipeline

stage('Deploy') {

steps {

sshagent(credentials: ['aws\_key']) {

sh '''

ssh -o StrictHostKeyChecking=no ubuntu@i9a810.p.ssafy.io uptime

chmod +x ./backend/deploy.sh

ssh -t -t ubuntu@i9a810.p.ssafy.io /home/ubuntu/jenkins/jenkins1/workspace/CocoChachaBE/backend/deploy.sh

'''

}

}

}

1. Back - deploy.sh

도커위에 빌드가 되길 원하는 폴더랑 바인드 마운트를 해서 빌드가 될 주소에 .jar 넣기

#!/bin/bash

pid=$(pgrep -f chaeum-backend)

echo "###########first check point###########"

if [ -n "${pid}" ]

then

kill -15 ${pid}

echo kill process ${pid}

else

echo no process

fi

echo "###########second check point###########"

chmod +x /home/ubuntu/jenkins/jenkins1/workspace/CocoChachaBE/BE/chaeum-backend/build/libs/chaeum-backend-0.0.1-SNAPSHOT.jar

nohup java -jar /home/ubuntu/jenkins/jenkins1/workspace/CocoChachaBE/BE/chaeum-backend/build/libs/chaeum-backend-0.0.1-SNAPSHOT.jar &

sudo disown -h

echo "###########프끝!!!!###########”

위 과정을 모두 마무리 하면 develop branch에 commit 혹은 merge할 때마다 자동으로 배포가 진행됨

## 서비스 이용 방법

### Bard API

- Spring Boot에 아래 있는 의존성 추가(Gradle 기준)

implementation group: 'com.pkslow', name: 'google-bard', version: '0.3.4'

- google bard에 접속 한 후 쿠키(\_\_Secure-1PSID) 확인

- 토큰에 해당 쿠키의 값을 넣어서 사용하면 됨

### 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