|  |
| --- |
|  |

|  |
| --- |
| **BookNEST** |
| **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 3](#_Toc96072463)

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

[II. 빌드 4](#_Toc96072465)

[1. 환경변수 형태 4](#_Toc96072466)

[2. 빌드하기 5](#_Toc96072467)

[3. 배포하기 6](#_Toc96072468)

# 개요

## 프로젝트 개요

북네스트는 LLM 기반 태깅과 사용자 로그 분석을 통해 작동하는 개인화 도서 추천 서비스입니다. 도서의 의미와 사용자의 선호를 태그 벡터로 정밀하게 매칭해 최적의 책을 추천합니다. *“나의 취향이 가득한 책 둥지를 틀다 BookNEST”*

## 프로젝트 사용 도구

이슈 관리 : JIRA

형상 관리 : Gitlab

커뮤니케이션 : Notion, Mattermost

디자인 : Figma

UCC : ~~모바비, 애프터이펙트, 프리미어~~

CI/CD : Jenkins, Docker

## 개발환경

|  |  |  |
| --- | --- | --- |
|  | Skill | Version |
| Back | Java | 17 |
| ****SpringBoot**** | 3.4.3 |
| IntelliJ | 21.0.5+8-b631.28 amd64 |
| ****Front**** | VS Code | 1.98.2 |
| Node.js | 22.14.0 |
| NPM | 11.0.0 |
| Yarn | 1.22.22 |
| ****Data**** | MariaDB | 10.6.21 |
| ****Redis**** | 6.0.16 |
| ****Elasticsearch**** | 8.17.4 |
| ****Logstash**** | 8.17.4 |
| ****Kibana**** | 8.17.4 |
| ****Airflow**** | 2.8.1 |
| ****Server**** | AWS EC2 | Ubuntu 22.04.5 LTS |

EC2 포트 번호

|  |  |
| --- | --- |
| skill | Port |
| **SpringBoot** | 8081/8082 |
| **MariaDB** | 3306 |
| **Redis** | 6379 |
| **Nginx** | 80/443 |
| **Jenkins** | 8080 |
| **Elasticsearch** | 5601 |
| **Logstash** | 5044 |
| **Kibana** | 9200 |

|  |  |
| --- | --- |
| **FastAPI** | 8000 |

## 외부 서비스

Google OAuth : application-secret.yml 에 해당 내용 있음

KakaoAPI : application-secret.yml에 해당 내용 있음

Naver OAuth : application-secret.yml에 해당 내용 있음

## Gitignore 처리한 핵심 키들

Vite : .env (최상단 위치)

Spring : application-secret.yml

(\src\main\resources, 또는 classPath에 위치)

FastAPI : .env (app/ 내에 위치)

ELK : .env (최상단 위치)

# 빌드

## 환경변수 형태

**.env :**

// API 요청 BASE URL

VITE\_API\_BASE\_URL=

// 카카오 로그인 클라이언트 ID

VITE\_KAKAO\_CLIENT\_ID=

// 카카오 로그인 redirectURI

VITE\_KAKAO\_REDIRECT\_URI=

// 구글 로그인 클라이언트 ID

VITE\_GOOGLE\_ CLIENT\_ID=

// 구글 로그인 클라이언트 SECRET

VITE\_GOOGLE\_CLIENT\_SECRET=

// 구글 로그인 redirectURI

VITE\_GOOGLE\_REDIRECT\_URI=

// 네이버 로그인 클라이언트 ID

VITE\_NAVER\_CLIENT\_ID=

// 네이버 로그인 클라이언트 SECRET

VITE\_NAVER\_CLIENT\_SECRET=

// 네이버 로그인 redirectURI

VITE\_NAVER\_REDIRECT\_URI=

// 이미지 업로드 클라우드 설정

VITE\_CLOUDINARY\_CLOUD\_NAME=

VITE\_CLOUDINARY\_API\_KEY=

VITE\_CLOUDINARY\_API\_SECRET=

VITE\_CLOUDINARY\_UPLOAD\_PRESET=

**application-secret.yml :**

# 도메인

app:

domain: 도메인

# DB

spring:

datasource:

url: ${DB\_URL}

username: ${DB\_USERNAME}

password: ${DB\_PASSWORD}

driver-class-name: org.mariadb.jdbc.Driver

data:

redis:

password: 비밀번호

port: 포트번호

fastapi:

url: <서버ip>:포트번호

# Security OAuth

oauth:

kakao:

client-id: 클라이언트 아이디

google:

client-id: 클라이언트 아이디

client-secret: 비밀번호

naver:

client-id: 클라이언트 아이디

client-secret: 비밀번호

# redirect-uri는 각 profile에 설정

# JWT 시크릿 키

jwt:

secret: 시크릿 키

## 빌드하기

1) Front

yarn install

yarn build

2) Back-spring

Gradle 실행

Bootjar 실행

**dockerfile :**

# Java 17 베이스 이미지 사용  
FROM openjdk:17-jdk-slim  
  
# Dockerfile 내에서 사용할 변수 JAR\_FILE을 정의  
ARG *JAR\_FILE*=\*.jar  
  
# JAR\_FILE 경로에 해당하는 파일을 Docker 이미지 내부로 복사  
COPY ${*JAR\_FILE*} spring-prod.jar  
  
# Docker 컨테이너가 시작될 때 실행할 명령을 지정  
ENTRYPOINT ["java", "-jar", "/spring-prod.jar"]

**Docker-compose.bule.yml :**

version: '3'  
services:  
 backend:build: .ports:  
 - "8111:80"container\_name: spring-booknest-blue  
 volumes:  
 - /home/ubuntu/app-logs-blue:/app/logs  
 environment:  
 - TZ=Asia/Seoul  
 networks:  
 - servernetwork  
  
networks:  
 servernetwork:  
 external: true  
 driver: bridge

**Docker-compose.green.yml :**

version: '3'  
services:  
 backend:  
 build: .  
 ports:  
 - "8112:80"  
 container\_name: spring-booknest-green  
 volumes:  
 - /home/ubuntu/app-logs-green:/app/logs  
 environment:  
 - TZ=Asia/Seoul  
 networks:  
 - servernetwork  
  
networks:  
 servernetwork:  
 external: true  
 driver: bridge

## 배포하기

**Nginx 설정**

- 해당 위치에 설정 파일 생성

sudo vi /etc/nginx/conf.d/project.conf

**project.conf**

listen 80;

server\_name www.gomin.my gomin.my;

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

}

# 80 포트에서 HTTPS로 리디렉션

server {

listen 80;

server\_name www.mybooknest.site mybooknest.site;

if ($host = mybooknest.site) {

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

} # managed by Certbot

if ($host = www.mybooknest.site) {

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

} # managed by Certbot

return 301 https://$host$request\_uri; # HTTP 트래픽을 HTTPS로 리디렉션

}

# 443 포트에서 HTTPS 처리

server {

listen 443 ssl;

server\_name www.mybooknest.site mybooknest.site;

ssl\_certificate /etc/letsencrypt/live/www.mybooknest.site/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/www.mybooknest.site/privkey.pem;

ssl\_protocols TLSv1.2 TLSv1.3;

ssl\_ciphers HIGH:!aNULL:!MD5;

# 정적 파일 서비스

location / {

root /home/ubuntu/dist;

index index.html;

try\_files $uri /index.html;

}

location = /index.html {

add\_header Cache-Control "no-cache, no-store, must-revalidate";

add\_header Pragma "no-cache";

add\_header Expires 0;

}

add\_header Cache-Control "public, max-age=31536000, immutable";

}

# API 요청을 Backend(Spring Boot)로 프록시

location /api {

add\_header X-Served-By $host;

proxy\_set\_header Host $host;

proxy\_set\_header X-Forwarded-Scheme $scheme;

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

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

proxy\_pass http://spring-booknest-server;

proxy\_redirect off;

}

}

이후 sudo systemctl restart nginx