포팅 메뉴얼

1. 시스템 환경 및 소프트웨어 정보

1.1 백엔드 (Spring Boot)

- **프레임워크**: Spring Boot
- **버전**: 3.3.1
- **JVM 버전:** Azul Zulu 17.0.11
- **빌드 도구**: Gradle
- **Docker 이미지**: Jenkins에서 빌드 후 Docker Hub에 푸시
- **IDE**: IntelliJ IDEA
- **IDE 버전**: 2024.1.4

1.2 프론트엔드 (Next.js)

- **프레임워크**: Next.js
- **빌드 도구**: npm
- **Docker 이미지**: Jenkins에서 빌드 후 Docker Hub에 푸시
- **IDE**: Visual Studio Code
- **IDE 버전**: 1.90.2

1.3 웹 서버

- **종류: Nginx**
- **역할:** Reverse Proxy로서 요청을 앤드포인트에 따라 백엔드, 프론트엔드로 구분하여 처리하고, 로드벨런싱을 통해 실행중인 도커 컨테이너(Blue or Green 컨테이너)로 라우팅

1.4 WAS

- **설정 파일:** `application.properties`, `info.properties`
    - info.properties 내용

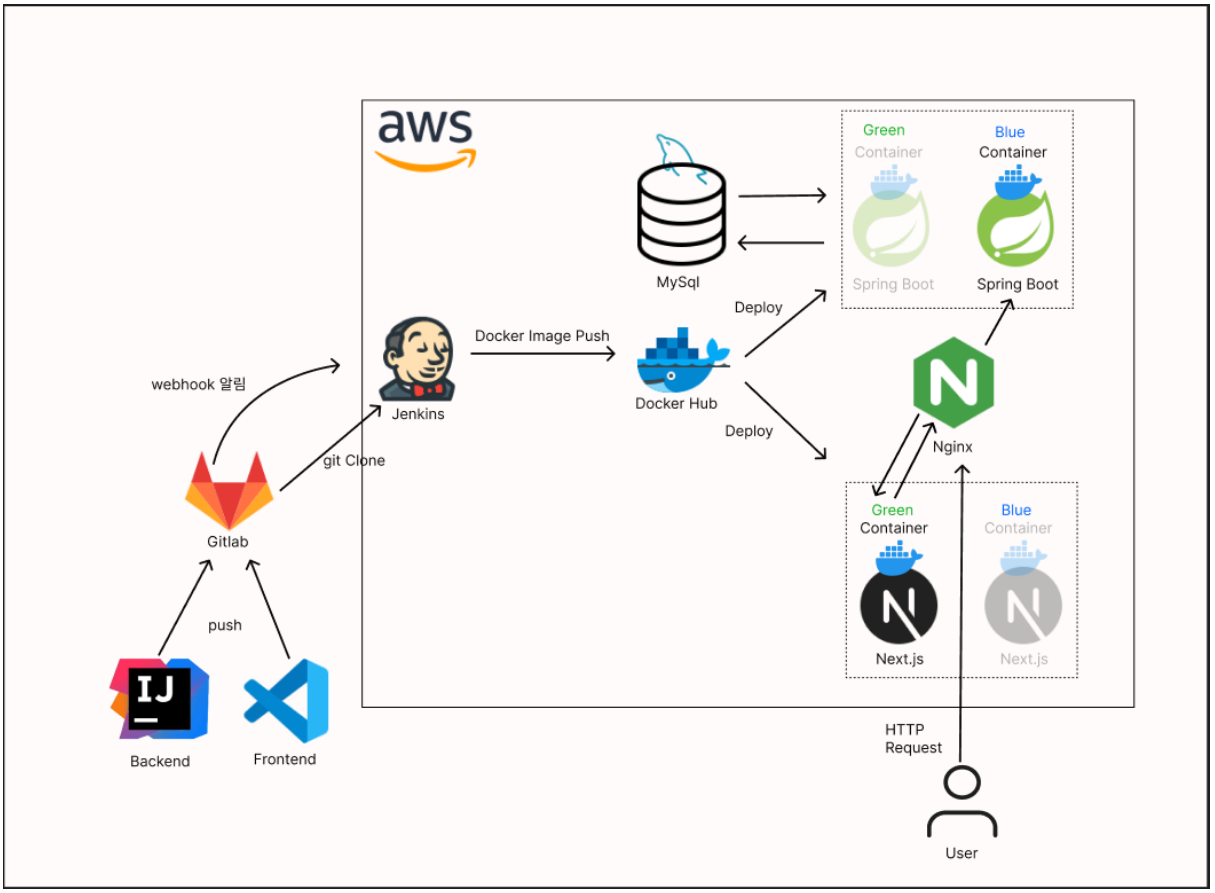
```
spring.datasource.url=jdbc:mysql://a508.cby8ig0ech43.ap
spring.datasource.username=admin
spring.datasource.password=ssafyA508!
spring.datasource.driver-class-name=com.mysql.cj.jdbc.D

spring.security.oauth2.client.registration.kakao.client
spring.security.oauth2.client.registration.kakao.client

spring.security.oauth2.client.registration.naver.client
spring.security.oauth2.client.registration.naver.client
```

종류: Spring Boot 3.0.0 내장 Tomcat

- **포트:** 8080 (내장 Tomcat 기본 포트)

1.5 인프라 구성

1.5.1 구조



1.5.2 EC2 인스턴스

- **역할:** Nginx와 여러 Docker 컨테이너(Spring Boot,Next.js)의 실행
- **설치된 소프트웨어:** Docker, Jenkins, JDK17, Certbot, Nginx, Node

1.5.3 RDS (Relational Database Service)

- **DB 종류:** Mysql
- **버전:** 8.0.37
- **접속정보**
  - **URL:** `a508.cby8ig0ech43.ap-northeast-2.rds.amazonaws.com` `/wms`
  - **Username:** `admin`
  - **Password:** `ssafyA508!`

1.5.4 Jenkins

- **접속정보**
  - Username: `a508`
  - Password: `ssafyA508!`

2. 빌드 및 배포

2.1 소스 클론

- GitLab에서 소스를 클론합니다

```
git clone https://lab.ssafy.com/s11-webmobile2-sub2/S11P
12A508.git
```

- 프론트엔드와 백엔드의 root 폴더로 이동합니다

```
#frontend root folder
cd wms-front
```

```
#backend root folder
cd wms_backend
```

2.2 빌드 과정

- (Backend) Gradle을 사용하여 빌드를 수행합니다:

```
./gradlew clean build
```

- (Frontend) npm을 사용하여 빌드를 수행합니다.

```
./npm run build
```

2.3 배포 과정

- 변경된 프로젝트를 master branch에 push 합니다.
- code push시 GitLab이 Jenkins에 웹훅을 통해 알림을 보냅니다.
- Jenkins는 GitLab에서 소스 코드를 클론한 후 CI/CD 파이프라인을 실행합니다.
  - 백엔드(Spring Boot)와 프론트엔드(Next.js)를 빌드합니다.
  - 각각의 애플리케이션을 Docker 이미지로 빌드한 후, Docker Hub에 이미지를 푸시합니다.
- **Docker Hub**: Jenkins가 푸시한 이미지를 저장하고, AWS의 인프라로 이미지를 배포합니다.

Blue-Green Deployment

- **Nginx**는 사용자 요청을 프록시하여 Green 컨테이너 또는 Blue 컨테이너로 라우팅합니다.
- Blue-Green Deployment 전략을 사용하여 애플리케이션의 무중단 배포를 구현합니다.