

연금술사 포팅 매뉴얼

목차

1. 프로젝트 개요

1-1. 기술 스택 및 버전

1-2. 환경변수 정보

1-3. 네트워크 구성 및 인프라 아키텍처

1-4. 프로젝트 구조

1-5. 외부 서비스

2. 환경 설정

2-1. EC2 초기 셋업

2-2. Jenkins 설치 및 설정



2-3. MySQL 설치 및 설정

2-4. Redis 설치

2-5. TLS 인증서 설정

2-6. Jenkins-GitLab 연결

2-7. Jenkins Kubeconfig 설정

3. 데이터베이스 설정

3-1. MySQL DB 생성 및 데이터 주입

3-2. Redis 데이터 관리

4. 접속 정보 및 배포 특이사항

4-1. 접속 정보

4-2. 배포 특이사항

5. APK 파일 설치 방법

1. 프로젝트 개요

1-1. 기술 스택 및 버전

- **OS:** Ubuntu 22.04 LTS
- **JVM:** OpenJDK 17.0.16
- **웹서버:** Traefik (k3s 기본)
- **WAS:** Spring Boot 내장 Tomcat, FastAPI + Uvicorn
- **컨테이너:** Docker 28.4.0, k3s v1.33.4+k3s1
- **CI/CD:** Jenkins 2.516.2
- **데이터베이스:** MySQL 8.0.43, Redis (Docker)
- **서버 스펙:** EC2 xlarge (4vCPUs, 16GB RAM, 320GB SSD)

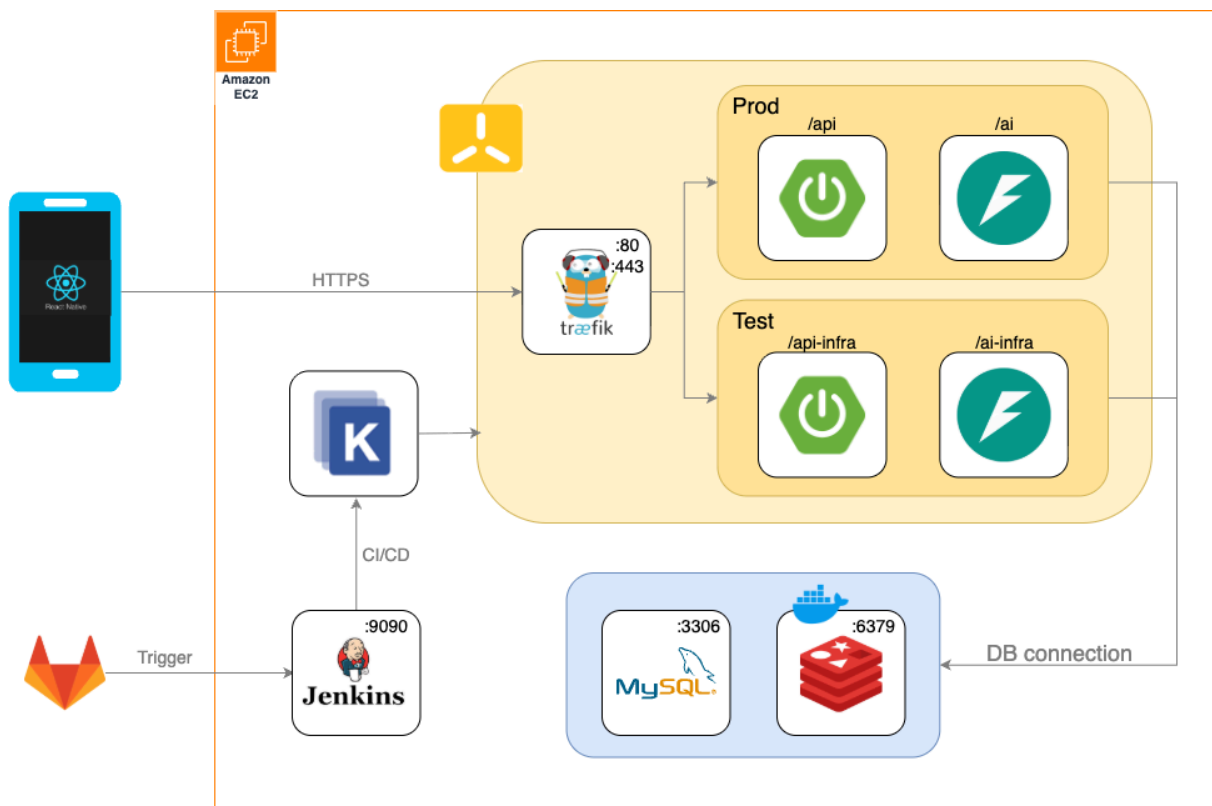
1-2. 환경변수 정보

- **백엔드:** `k8s/base/backend/configmap.yml`
- **AI 서비스:** `k8s/base/ai/configmap.yml`
- **민감정보:** Kubernetes Secret `backend-secrets`
- **Jenkins:** `Jenkinsfile.prod` , `Jenkinsfile.test`
- **GPU env**

```
# AI 모델 학습 환경 설정 (GPU 서버)
# GPU 서버에서 모델 학습 및 결과 전송을 위한 환경변수 설정:
AI_TOKEN=[jwt-token]
MODEL_BASE_PATH=/app/saved_models
BASE_API_PATH=https://j13a103.p.ssafy.io/api
JUPYTER_BASE_PATH=/home/j-j13a103/ai
FASTAPI_BASE_PATH=https://j13a103.p.ssafy.io/ai
```

1-3. 네트워크 구성 및 인프라 아키텍처

- 도메인: j13a103.p.ssafy.io
- 프로덕션: `/api` (백엔드), `/ai` (AI) → master 브랜치
- 테스트: `/api-infra`, `/ai-infra` → infra/swandkim/internal/infra-refactor 브랜치
- 포트: 22(SSH), 80(HTTP), 443(HTTPS), 9090(Jenkins)



1-4. 프로젝트 구조

S13P11A506/

— ai/	# FastAPI AI 서비스
— Dockerfile	# AI 서비스 컨테이너 이미지
— Dockerfile.base	# AI 베이스 이미지 (의존성 분리)
— .dockerignore	# Docker 빌드 제외 파일 목록
— requirements.txt	# Python 패키지 의존성

```

├── backend/                # Spring Boot 백엔드
│   ├── Dockerfile         # 백엔드 서비스 컨테이너 이미지
│   ├── .dockerignore      # Docker 빌드 제외 파일 목록
│   └── k8s/               # Kubernetes 매니페스트 파일
│       ├── base/          # 공통 기본 리소스
│       │   ├── ai/        # AI 서비스 k8s 리소스
│       │   │   ├── configmap.yml # AI 환경변수 설정
│       │   │   ├── deployment.yml # AI Pod 배포 설정
│       │   │   └── service.yml   # AI 서비스 네트워크 설정
│       │   ├── backend/     # 백엔드 서비스 k8s 리소스
│       │   │   ├── configmap.yml # 백엔드 환경변수 설정
│       │   │   ├── deployment.yml # 백엔드 Pod 배포 설정
│       │   │   └── service.yml   # 백엔드 서비스 네트워크 설정
│       │   ├── shared/      # 공유 리소스
│       │   │   └── ingress.yml  # 외부 트래픽 라우팅 설정
│       │   └── kustomization.yml # base 환경 통합 설정
│       ├── overlays/       # 환경별 커스터마이징
│       │   ├── prod/       # 프로덕션 환경
│       │   │   ├── kustomization.yml # prod 환경 설정
│       │   │   ├── patch-ai.yml   # AI prod 환경 패치
│       │   │   ├── patch-backend.yml # 백엔드 prod 환경 패치
│       │   │   └── patch-ingress.yml # 인그레스 prod 환경 패치
│       │   └── infra-refactor/    # 인프라 테스트 환경
│       │       ├── kustomization.yml # 테스트 환경 설정
│       │       ├── patch-backend.yml # 백엔드 테스트 환경 패치
│       │       └── patch-ingress.yml # 인그레스 테스트 환경 패치
│       └── Jenkinsfile.prod # 프로덕션 CI/CD 파이프라인

```

1-5. 외부 서비스

- **GMS API:** <https://gms.ssafy.io/gmsapi> (Secret Key 필요)

2. 환경 설정

2-1. EC2 초기 셋업

```
# 방화벽 설정
sudo ufw --force enable
sudo ufw allow 22/tcp 80/tcp 443/tcp 9090/tcp

# 기본 업데이트 및 도구 설치
sudo apt update && sudo apt upgrade -y
sudo apt install -y git vim htop

# Docker 설치
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt update && sudo apt install -y docker-ce docker-ce-cli containerd.io
sudo systemctl start docker && sudo systemctl enable docker

# k3s 설치
curl -sfL https://get.k3s.io | sh -s - --write-kubeconfig-mode 644

# kubectl 설정
mkdir -p ~/.kube
sudo cp /etc/rancher/k3s/k3s.yaml ~/.kube/config
sudo chown $USER:$USER ~/.kube/config

# 사용자 권한 설정
sudo usermod -aG docker $USER
```

2-2. Jenkins 설치 및 설정

0) 사전: Java 17+ 보장

```
sudo apt install -y openjdk-17-jdk
```

1) Jenkins 저장소/키 추가 + 설치

```
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key \
| sudo tee /usr/share/keyrings/jenkins-keyring.asc >/dev/null
echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://pk
g.jenkins.io/debian-stable binary/" \
| sudo tee /etc/apt/sources.list.d/jenkins.list >/dev/null
sudo apt update
sudo apt install -y jenkins
```

2) 포트 변경 (8080 -> 9090)

```
sudo sed -i 's/^HTTP_PORT=.* HTTP_PORT=9090/' /etc/default/jenkins
```

3) JVM 옵션에 타임존 설정 (기존 JAVA_ARGS에 추가)

파일 내 JAVA_ARGS 라인에 -Duser.timezone=Asia/Seoul 추가 (없으면 새로 추가)

```
if grep -q '^JAVA_ARGS=' /etc/default/jenkins; then
    sudo sed -i 's/^JAVA_ARGS="#JAVA_ARGS="-Duser.timezone=Asia/Seoul
#/' /etc/default/jenkins
else
    echo 'JAVA_ARGS="-Djava.awt.headless=true -Duser.timezone=Asia/Seo
ul"' | sudo tee -a /etc/default/jenkins
fi
```

4) 서비스 적용

```
sudo systemctl daemon-reload
sudo systemctl enable --now jenkins
```

사용자 권한 설정


```
sudo usermod -aG docker jenkins
```

Jenkins 상태 확인

```
sudo systemctl status jenkins
```

```
# Jenkins 초기 비밀번호 확인
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

Jenkins 설정:

1. <http://j13a103.p.ssafy.io:9090> 접속
2. 초기 비밀번호 입력
3. "Install suggested plugins" 선택
4. 관리자 계정 생성 (admin)
5. 추가 플러그인 설치: Docker, Docker Pipeline, NodeJS, GitLab, GitLab Branch Source, Kubernetes, Kubernetes CLI, Pyenv Pipeline, ShiningPanda
6. 도구 설정 (Jenkins 관리 > Tools)
 - a. NodeJS
 - Name: Node22
 - Install automatically: 
 - Version: NodeJS 22.19.0 (목록에서 선택) ← LTS
 - b. Docker
 - Name: Docker
 - Installation root : 공란
 - Install automatically: 체크 해제
 - c. Python
 - Name: Python3
 - Home or executable : /usr/bin/python3
 - Install automatically: 체크 해제

2-3. MySQL 설치 및 설정

```
# MySQL 설치
sudo apt install mysql-server

# 보안 설정 (root 비밀번호 등)
```



```
sudo mysql_secure_installation
```

```
# admin 계정 생성
```

```
sudo mysql
```

```
CREATE USER 'admin'@'%' IDENTIFIED BY 'my_password';  
GRANT ALL PRIVILEGES ON *.* TO 'admin'@'%' WITH GRANT OPTION;  
FLUSH PRIVILEGES;  
EXIT;
```

```
# 네트워크 설정 (bind-address를 EC2 프라이빗 IP로 변경)
```

```
sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf
```

```
# bind-address = 172.26.7.217
```

```
sudo systemctl restart mysql
```

2-4. Redis 설치

```
docker run -d \  
  --name redis-stack \  
  --restart unless-stopped \  
  -p 172.26.7.217:6379:6379 \  
  -v redis-stack-data:/data \  
  -e TZ=Asia/Seoul \  
  redis/redis-stack-server:latest
```

```
# 비밀번호 설정
```

```
docker exec -it redis-stack redis-cli CONFIG SET requirepass "암호"
```

2-5. TLS 인증서 설정

```
# cert-manager 설치
```

```
kubectl apply -f https://github.com/cert-manager/cert-manager/releases/d
```

ownload/v1.13.0/cert-manager.yaml

```
# ClusterIssuer 생성
cat > cert-issuer.yml << 'EOF'
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: letsencrypt-prod
spec:
  acme:
    server: https://acme-v02.api.letsencrypt.org/directory
    email: admin@j13a103.p.ssafy.io
    privateKeySecretRef:
      name: letsencrypt-prod
    solvers:
      - http01:
          ingress:
            class: traefik
EOF

kubectl apply -f cert-issuer.yml
```

2-6. Jenkins-GitLab 연결

1. GitLab Personal Access Token 발급

- Scopes: `api`, `read_repository`

2. Jenkins Credentials 등록

T	P	Store ↓	Domain	ID	Name
		System	(global)	gitlab-ygss-token	GitLab API token (GitLab YGSS Project Access Token)
		System	(global)	gitlab-git-token	swandkim@gmail.com/***** (GitLab Username & Password)
		System	(global)	mysql-password	MySQL admin password for YGSS backend
		System	(global)	jwt-secret	JWT secret key for YGSS backend authentication
		System	(global)	gms-secret-key	GMS Secret Key
		System	(global)	redis-password	redis 컨테이너 password

- gitlab-ygss-token (GitLab API Token)
- gitlab-git-token (Username with password, GitLab 계정)
- mysql-password (Secret text)
- jwt-secret (Secret text)
- gms-secret-key (Secret text)
- redis-password (Secret text)

3. 프로덕션용 Pipeline 생성

- Definition: Pipeline script from SCM
- SCM : Git
- Repository URL: GitLab 프로젝트 URL
- Credentials: GitLab Username & Password
- Branches to build - Branch Sepcifier : `*/master`
- Script Path: `Jenkinsfile.prod`
- Lightweight checkout : 체크

4. 테스트용 Pipeline 생성 (선택사항)

- Definition: Pipeline script from SCM
- SCM : Git
- Repository URL: GitLab 프로젝트 URL
- Credentials: GitLab Username & Password
- Branches to build - Branch Sepcifier : `*/infra/swandkim/internal/infra-refactor`

- Script Path: `Jenkinsfile.test`
- Lightweight checkout : 체크

2-7. Jenkins kubeconfig 설정

```
# Jenkins용 kubeconfig 설정
sudo install -d -o jenkins -g jenkins -m 700 /var/lib/jenkins/.kube
sudo cp /etc/rancher/k3s/k3s.yaml /var/lib/jenkins/.kube/config
sudo chown jenkins:jenkins /var/lib/jenkins/.kube/config
```

3. 데이터베이스 설정

3-1. MySQL DB 생성 및 데이터 주입

```
# 파일 EC2 내 복사
scp -i J13A103T.pem *.sql ubuntu@j13a103.p.ssafy.io:/home/ubuntu/ygss-project/sql
```

```
# 데이터베이스 생성
mysql -u admin -p
# 비밀번호 입력
```

```
CREATE DATABASE ygss;
USE ygss;
```

```
source /home/ubuntu/ygss-project/sql/schema.sql;
source /home/ubuntu/ygss-project/sql/basic_insert.sql;
```

3-2. Redis 데이터 관리

- 데이터 업데이트: `POST /redis/update` (토큰 인증 필요)
- Redis CLI 접속: `redis-cli -h 172.26.7.217 -p 6379 -a [password]`

4. 접속 정보 및 배포 특이사항

4-1. 접속 정보

- Jenkins: `http://j13a103.p.ssafy.io:9090`
- 애플리케이션: `https://j13a103.p.ssafy.io`
- MySQL: `172.26.7.217:3306 (admin/[password])`
- Redis: `172.26.7.217:6379 (password 설정됨)`
- 헬스체크 URL
 - 프로덕션 백엔드 : `https://j13a103.p.ssafy.io/api/infra`
 - 프로덕션 AI 서비스 : `https://j13a103.p.ssafy.io/ai/health/`
 - 테스트용 백엔드 : `https://j13a103.p.ssafy.io/api-infra/infra`
 - 테스트용 AI 서비스 : `https://j13a103.p.ssafy.io/ai-infra/health/`

4-2. 배포 특이사항

- 배포 방식: Rolling Update (maxUnavailable: 0, maxSurge: 1)
- 네임스페이스: `ygss-prod` (프로덕션), `ygss-infra` (테스트)
- 볼륨: AI 모델 저장소 `/opt/ygss/ai-models` → `/app/saved_models`

5. APK 파일 설치 방법

→ 프로젝트 README의 QR 코드를 스캔하여 APK 직접 다운로드 및 설치

※ 다운로드 시 유의사항

1. 경고문구 확인 시 '무시하고 다운로드' 선택
2. '출처를 알 수 없는 앱 차단됨' 허용 방법
 - a. 설정 > 보안 및 개인 정보 보호 > 보안 위험 자동 차단 끄기
 - b. 설정 > 보안 및 개인 정보 보호 > 기타 보안 설정 > 출처를 알 수 없는 앱 설치 → Drive, Chrome 허용