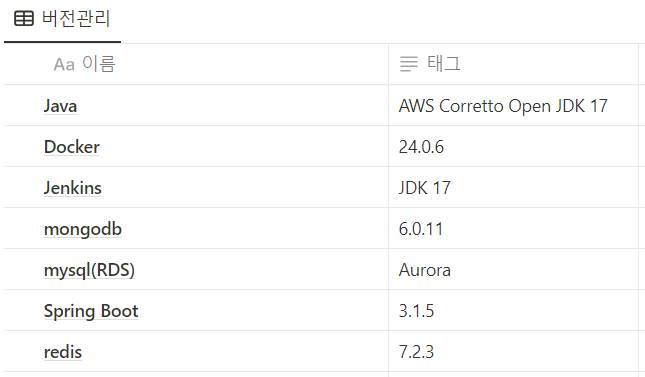
1. GitLab 소스 클론 이후 빌드 및 배포할 수 있도록 정리한 문서



1) 사용한 JVM, 웹 서버, WAS 제품 등의 종류와 설정 값, 버전(IDE버전 포함) 기재

- JDK: Corretto:17

- WAS: Tomcat, Netty(API Gateway)

-웹 서버: Nginxx

IDE:

- Front: VSCode(1.18.1)

- Back: InteliiJ(11.0.19)

2) 빌드 시 사용되는 환경 변수 등의 주요 내용 상세

- Docker 환경 Dockerfile

[APIGateway Server]

#환경

FROM amazoncorretto:17

#변수설정

ARG JAR\_FILE=build/libs/\*.jar

#복사

COPY ${JAR\_FILE} gateway.jar

#Port번호

EXPOSE 8080

#서버 명령어

ENTRYPOINT ["java", "-jar", "/gateway.jar"]

#Testing Line

#인식

[Config Server]

#환경

FROM amazoncorretto:17

#변수설정

ARG JAR\_FILE=build/libs/\*.jar

#복사

COPY ${JAR\_FILE} config.jar

#Port번호

EXPOSE 8888

#서버 명령어

ENTRYPOINT ["java", "-jar", "/config.jar"]

#Testing Line

#인식

[Discovery Server]

#환경

FROM amazoncorretto:17

#변수설정

ARG JAR\_FILE=build/libs/\*.jar

#복사

COPY ${JAR\_FILE} discovery.jar

#Port번호

EXPOSE 8761

#서버 명령어

ENTRYPOINT ["java", "-jar", "/discovery.jar"]

#Testing Line

#인식

[Auth Server]

#환경

FROM amazoncorretto:17

#변수설정

ARG JAR\_FILE=build/libs/\*.jar

#복사

COPY ${JAR\_FILE} auth.jar

#Port번호

EXPOSE 8082

#서버 명령어

ENTRYPOINT ["java", "-jar", "/auth.jar"]

#Testing Line

#인식

[GPT Server]

#환경

FROM amazoncorretto:17

#변수설정

ARG JAR\_FILE=build/libs/\*.jar

#복사

COPY ${JAR\_FILE} gpt.jar

#Port번호

EXPOSE 8084

#서버 명령어

ENTRYPOINT ["java", "-jar", "/gpt.jar"]

#Testing Line

#인식

[Editor Server]

#환경

FROM amazoncorretto:17

#변수설정

ARG JAR\_FILE=build/libs/\*.jar

#복사

COPY ${JAR\_FILE} editor.jar

#Port번호

EXPOSE 8085

#서버 명령어

ENTRYPOINT ["java", "-jar", "/editor.jar"]

#Testing Line

#인식

[Diagram Server]

#환경

FROM amazoncorretto:17

#변수설정

ARG JAR\_FILE=build/libs/\*.jar

#복사

COPY ${JAR\_FILE} diagram.jar

#Port번호

EXPOSE 8087

#서버 명령어

ENTRYPOINT ["java", "-jar", "/diagram.jar"]

#Testing Line

#인식

[Quiz Server]

#환경

FROM amazoncorretto:17

#변수설정

ARG JAR\_FILE=build/libs/\*.jar

#복사

COPY ${JAR\_FILE} quiz.jar

#Port번호

EXPOSE 8088

#서버 명령어

ENTRYPOINT ["java", "-jar", "/quiz.jar"]

#Testing Line

#인식

[Search Server]

#환경

FROM amazoncorretto:17

#변수설정

ARG JAR\_FILE=build/libs/\*.jar

#복사

COPY ${JAR\_FILE} search.jar

#Port번호

EXPOSE 8086

#서버 명령어

ENTRYPOINT ["java", "-jar", "/search.jar"]

#Testing Line

#인식

[Front Server(Next.JS)]

# 베이스 이미지 선택 (Node.js 18)

FROM node:18

# 앱 디렉토리 생성 및 작업 디렉토리로 설정

WORKDIR /usr/src/app

# 앱 종속성 설치

COPY package\*.json ./

RUN npm install

# 앱 소스 복사

COPY . .

# 빌드

RUN yarn build

# 포트 3000으로 노출

EXPOSE 3000

# 앱 실행

CMD [ "yarn", "start" ]

- Jenkins (Execute Shell)

[API Gateway]

cd gateway

chmod +x ./gradlew

./gradlew clean build -x test

docker ps -f name=gateway -q | xargs --no-run-if-empty docker container stop

docker container ls -a -f name=gateway -q | xargs -r docker container rm || true

docker build -t gateway .

docker run -it -d -p 8080:8080 --network msa-server -e TZ=Asia/Seoul --name=gateway gateway

docker rmi -f $(docker images -f "dangling=true" -q) || true

[Config Server]

cd config

chmod +x ./gradlew

./gradlew clean build -x test

docker ps -f name=config -q | xargs --no-run-if-empty docker container stop

docker container ls -a -f name=config -q | xargs -r docker container rm || true

docker build -t config .

docker run -it -d -p 8888:8888 --network msa-server -e TZ=Asia/Seoul --name=config config

docker rmi -f $(docker images -f "dangling=true" -q) || true

[Discovery Server]

cd discovery

chmod +x ./gradlew

./gradlew clean build -x test

docker ps -f name=discovery -q | xargs --no-run-if-empty docker container stop

docker container ls -a -f name=discovery -q | xargs -r docker container rm || true

docker build -t discovery .

docker run -it -d -p 8761:8761 --network msa-server -e TZ=Asia/Seoul --name=discovery discovery

docker rmi -f $(docker images -f "dangling=true" -q) || true

[GPT Server]

cd gpt

chmod +x ./gradlew

./gradlew clean build -x test

docker ps -f name=gpt -q | xargs --no-run-if-empty docker container stop

docker container ls -a -f name=gpt -q | xargs -r docker container rm || true

docker build -t gpt .

docker run -it -d -p 8084:8084 --network msa-server -e TZ=Asia/Seoul --name=gpt gpt

docker rmi -f $(docker images -f "dangling=true" -q) || true

[Editor Server]

cd editor

chmod +x ./gradlew

./gradlew clean build -x test

docker ps -f name=editor -q | xargs --no-run-if-empty docker container stop

docker container ls -a -f name=editor -q | xargs -r docker container rm || true

docker build -t editor .

docker run -it -d -p 8085:8085 --network msa-server -e TZ=Asia/Seoul --name=editor editor

docker rmi -f $(docker images -f "dangling=true" -q) || true

[Diagram Server]

cd diagram

cp /var/jenkins\_home/diagram/natural-402603-1827cceef8e7.json ./src/main/resources/static

chmod +x ./gradlew

./gradlew clean build -x test

docker ps -f name=diagram -q | xargs --no-run-if-empty docker container stop

docker container ls -a -f name=diagram -q | xargs -r docker container rm || true

docker build -t diagram .

docker run -it -d -p 8087:8087 --network msa-server -e TZ=Asia/Seoul --name=diagram diagram

docker rmi -f $(docker images -f "dangling=true" -q) || true

[Quiz Server]

cd quiz

chmod +x ./gradlew

./gradlew clean build -x test

docker ps -f name=quiz -q | xargs --no-run-if-empty docker container stop

docker container ls -a -f name=quiz -q | xargs -r docker container rm || true

docker build -t quiz .

docker run -it -d -p 8088:8088 --network msa-server -e TZ=Asia/Seoul --name=quiz quiz

docker rmi -f $(docker images -f "dangling=true" -q) || true

[Search Server]

cd search

chmod +x ./gradlew

./gradlew clean build -x test

docker ps -f name=search -q | xargs --no-run-if-empty docker container stop

docker container ls -a -f name=search -q | xargs -r docker container rm || true

docker build -t search .

docker run -it -d -p 8086:8086 --network msa-server -e TZ=Asia/Seoul --name=search search

docker rmi -f $(docker images -f "dangling=true" -q) || true

[Front Server]

cd front

cp /var/jenkins\_home/front/.env .

docker ps -f name=front -q | xargs --no-run-if-empty docker container stop

docker container ls -a -f name=front -q | xargs -r docker container rm || true

docker build -t front .

docker run -it -d --rm -p 3000:3000 -e TZ=Asia/Seoul --name=front front

docker rmi -f $(docker images -f "dangling=true" -q) || true

[Auth Server]

cd auth

chmod +x ./gradlew

./gradlew clean build -x test

docker ps -f name=auth -q | xargs --no-run-if-empty docker container stop

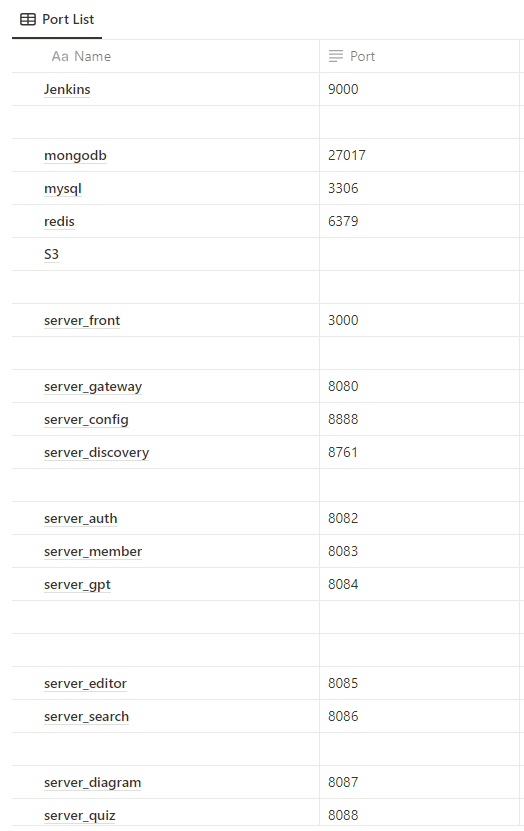
docker container ls -a -f name=auth -q | xargs -r docker container rm || true

docker build -t auth .

docker run -it -d -p 8082:8082 --network msa-server -e TZ=Asia/Seoul --name=auth auth

docker rmi -f $(docker images -f "dangling=true" -q) || true

포트



Jenkins Build Deploy 상태



\*시연 시나리오는 README.md에 첨부