

# 환경 세팅 메뉴얼

## Front End

Dockerfile

Build

.env

## Back End

### 1. DreamGream Spring Server

Dockerfile

Docker Image Build

환경 변수 세팅

### 2. Image Request Spring Server

Dockerfile

Docker Image Build

환경 변수

참고 사이트

### 3. Prompt Api Server

Dockerfile

.env

Docker Image Build

## DB

### 1. MySQL

참고 사이트

MySQL Container 실행 및 접속하기

### 2. Redis

docker-compose.yml

redis.conf

## Message Queue

### RabbitMQ

참고 사이트

RabbitMQ Container 실행하기

Admin User 추가하기

Spring Server application.yml 세팅

## Front End

---

- node 18.15.0
- IDE : vscode

## Dockerfile

Dockerfile은 FrontEnd/dream-gream 위치에 있다.

```
FROM node:18 AS build

WORKDIR /app

COPY package*.json ./
RUN npm install
COPY . ./
COPY .env.prod .env
RUN npm run build

FROM nginx:stable-alpine
COPY --from=build /app/build /usr/share/nginx/html
COPY nginx.conf /etc/nginx/conf.d/default.conf
EXPOSE 3000
CMD ["nginx", "-g", "daemon off;"]
```

## Build

```
# docker hub private repository에 fe라는 tag로 docker 이미지를 build
docker build -t jlal1226/secret-repo:fe --platform linux/amd64 .
docker push jlal1226/secret-repo:fe
```

## .env

```
REACT_APP_KAKAO_KEY="카카오에서 발급받은 KEY"
REACT_APP_PUBLIC_URL=http://localhost:3000
REACT_APP_API_URL="요청보낼 서버 주소"
REACT_APP_ORIGIN_URL=localhost:3000
```

## Back End

- Java 11
- Spring Boot 2.7.13
- Spring Data JPA

- Spring Security
- Query DSL
- Gradle
- Python 3.10
- FastAPI 0.100.0
- IDE : IntelliJ, vscode

## 1. DreamGream Spring Server

### Dockerfile

```
FROM openjdk:11-jdk

ARG JAR_FILE=build/libs/*.jar

COPY ${JAR_FILE} app.jar

ENTRYPOINT ["java","-jar", "/app.jar"]
```

### Docker Image Build

```
docker build -t jlal1226/secret-repo:DreamGream --platform linux/amd64 .
docker push jlal1226/secret-repo:DreamGream
```

## 환경 변수 세팅

### 1. S3

```
cloud:
  aws:
    s3:
      bucket: ${S3_BUCKET_NAME}
    region:
      static: ${S3_REGION}
    auto: false
```


```
stack:
  auto: false
credentials:
  access-key: ${S3_ACCESS_KEY}
  secret-key: ${S3_SECRET_KEY}
```

프로젝트 실행 전, S3 관련 환경 변수를 직접 추가해야 한다.

## 참고 사이트

### [Spring Boot] AWS S3를 이용한 파일 업로드

AWS S3란? AWS Simple Storage Service의 줄임말로 파일 서버의 역할을 하는 서비스 프로젝트 개발 중 파일을 저장하고 불러오는 작업이 필요한 경우에 프로젝트 내부 폴더에 저장할 수 있

 <https://chb2005.tistory.com/200>



## 2. Kakao Login

kakao developers에서 발급받은 api key를 사용해야한다.

```
spring:
  security:
    oauth2:
      client:
        registration:
          kakao:
            client-name: Kakao
            client-id: <클라이언트 ID>
            client-secret: <클라이언트 비밀키>
            scope: profile_nickname, account_email
            redirect-uri: <카카오 redirect url>
            client-authentication-method: client_secret_post
            authorization-grant-type: authorization_code
        provider:
          kakao:
            authorization-uri: https://kauth.kakao.com/oauth/authorize
            token-uri: https://kauth.kakao.com/oauth/token
            user-info-uri: https://kapi.kakao.com/v2/user/me
            user-name-attribute: id
```

## 2. Image Request Spring Server

## Dockerfile

```
FROM openjdk:11-jdk

ARG JAR_FILE=build/libs/*.jar

COPY ${JAR_FILE} app.jar

ENTRYPOINT ["java","-jar", "/app.jar"]
```

## Docker Image Build

```
docker build -t jlal1226/secret-repo:ImageRequestServer --platform linux/amd64 .
docker push jlal1226/secret-repo:ImageRequestServer
```

## 환경 변수


```
cloud:
  aws:
    s3:
      bucket: ${S3_BUCKET_NAME}
      region:
        static: ${S3_REGION}
        auto: false
      stack:
        auto: false
      credentials:
        access-key: ${S3_ACCESS_KEY}
        secret-key: ${S3_SECRET_KEY}
```

프로젝트 실행 전, S3 관련 환경 변수를 직접 추가해야 한다.

## 참고 사이트

### [Spring Boot] AWS S3를 이용한 파일 업로드

AWS S3 란? AWS Simple Storage Service의 줄임말로 파일 서버의 역할을 하는 서비스 프로젝트 개발 중 파일을 저장하고 불러오는 작업이 필요한 경우에 프로젝트 내부 폴더에 저장할 수 있지만, AWS S3

 <https://chb2005.tistory.com/200>



## 3. Prompt Api Server

### Dockerfile

```
FROM python:3.10

# 작업 디렉토리 설정
WORKDIR /app

# 필요한 파일들을 컨테이너에 복사
COPY main.py /app/main.py
COPY api/ /app/api/
COPY models/ /app/models/
COPY langchain.py /app/langchain.py
COPY .env /app/.env
COPY requirements.txt /app/requirements.txt

# 필요한 라이브러리 설치
RUN pip install -r requirements.txt

# FastAPI 앱 실행
CMD ["uvicorn", "main:app", "--host", "0.0.0.0", "--port", "8002"]
```

.env 파일에는 GPT API 요청에 필요한 KEY가 필요하다.

### .env

```
API_KEY = <api key>
```

### Docker Image Build

```
docker build -t jlal1226/secret-repo:PromptApi --platform linux/amd64 .
docker push jlal1226/secret-repo:PromptApi
```

## DB

---

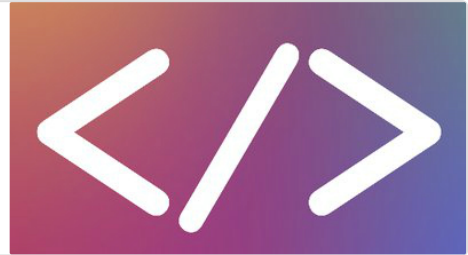
### 1. MySQL

## 참고 사이트

PoimaWeb

웹 프로그래밍 튜토리얼

 <https://poimaweb.com/docker-mysql>



## MySQL Container 실행 및 접속하기

```
docker run --name ssafy-db -e MYSQL_ROOT_PASSWORD=ssafya609 -d -p 3306:3306 mysql

# ssafy-db container 접속
docker exec -it ssafy-db bash

# mysql 접속
>> mysql -u root -p
>> ssafya609

# dreamgreem database 생성
>> create database dreamgreem;
```

## 2. Redis

### docker-compose.yml

```
version: '3'
services:
  redis:
    image: redis:7.0.12
    container_name: ssafy-redis
    environment:
      - TZ=Asia/Seoul
    networks:
      - ssafy-net
    volumes:
      - /home/ubuntu/etc/redis/redis.conf:/usr/local/etc/redis/redis.conf
      - redis_dev_data:/data
    command: ["redis-server", "/usr/local/etc/redis/redis.conf"]
  volumes:
    redis_dev_data:
      external: true
  networks:
```

```
ssafy-net:
  external: true
```

## redis.conf

```
# 바인딩 될 ip 설정
bind 127.0.0.1

# 포트 설정
port 16379

# 비밀번호 설정
requirepass "ssafya609"

# 백업 주기 설정
save 60 100 600 50 30 1000

# 최대 메모리 설정
maxmemory 2GB

# 최대로 접속 가능한 클라이언트값
maxclients 1000
```

# Message Queue

## RabbitMQ

### 참고 사이트

[docker] 도커로 RabbitMQ 설치부터 관리페이지 접속까지

docker pull rabbitmq 도커 명령어로 rabbitmq를 다운로드 한다. (위처럼 입력하면 latest 버전을 받는다.) docker run -d -p 15672:15672 -p 5672:5672 --name rabbitmq rabbitmq 포트는 기

 <https://onethejay.tistory.com/32>

  
페이지가 작동하지 않습니다.  
localhost에서 방문할 페이지가 없습니다.  
ERR\_HTTPS\_RESPONSE

## RabbitMQ Container 실행하기



```
docker run -d -p 15672:15672 -p 5672:5672 --name rabbitmq rabbitmq
```

## Admin User 추가하기

```
# admin user 추가
rabbitmqctl add_user admin ssafya609

# admin user를 administrator로 설정
rabbitmqctl set_user_tags admin administrator

# 권한 추가
rabbitmqctl set_permissions -p / admin ".*" ".*" ".*"
```

## Spring Server application.yml 세팅

```
spring:
  config:
    activate:
      on-profile: "rabbitmq-prod"
  # RabbitMQ 연결 정보
  rabbitmq:
    host: rabbitmq
    port: 5672
    username: admin
    password: ssafya609
  rabbitmq:
    exchange:
      name: image-creation-exchange
      type: direct
    queue:
      request:
        name: image-creation-request-queue
        routing-key: image.creation.request
      response:
        name: image-creation-response-queue
        routing-key: image.creation.response
```