개발 환경

IDE	VS_CODE	1.85.1
	INTELIJ	2023.3.2
FRONT	NODE	V20.10.0
	REACT	18.2.0
	REACT-ROUTER-DOM	6.21.3
	RECOIL	0.7.7
	AXIOS	1.6.5
	WEBSOCKET	1.0.34
	MUI/MATERIAL	5.15.6
	JS-FILE-DOWNLOAD	0.4.12
	STOMP/STOMPJ	7.0.0

개발 환경

BACKEND	JAVA	17.0.9 2023-10-17 LTS
	SPRINGBOOT	3.2.1
DATABASE	MARIA_DB	5.6.47.0
DEV-OPS	DOCKER	25.0.0
	DOCKER_COMPOSE	V2.24.1
ANALYZE	NODE_EXPORTER	1.7.0
	PROMETHEUS	2.49
	GRAFANA	10.3.1
	NGRINDER	3.5.8

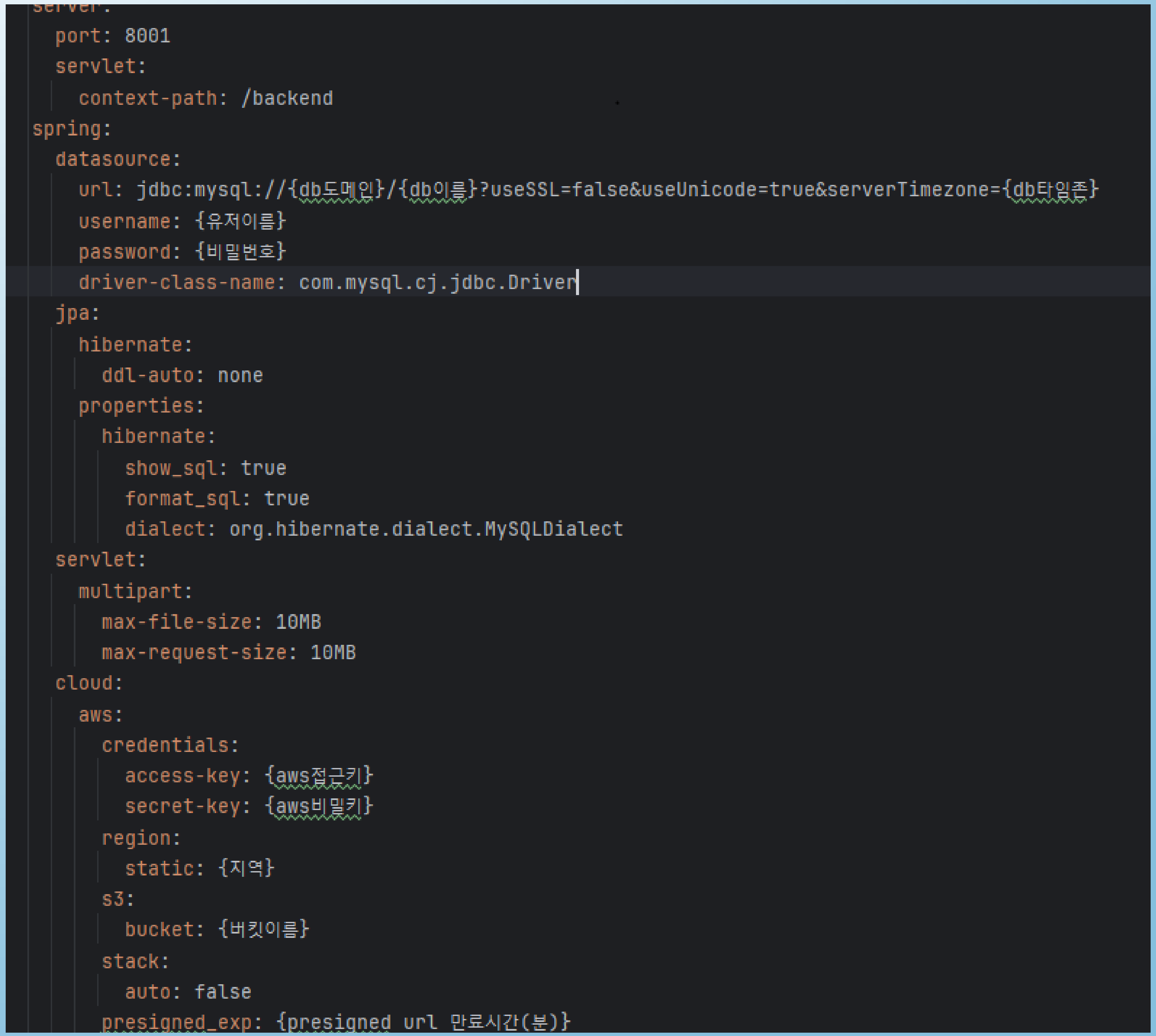
환경 변수-FRONTEND

PATH-/project/front/src/.env.production

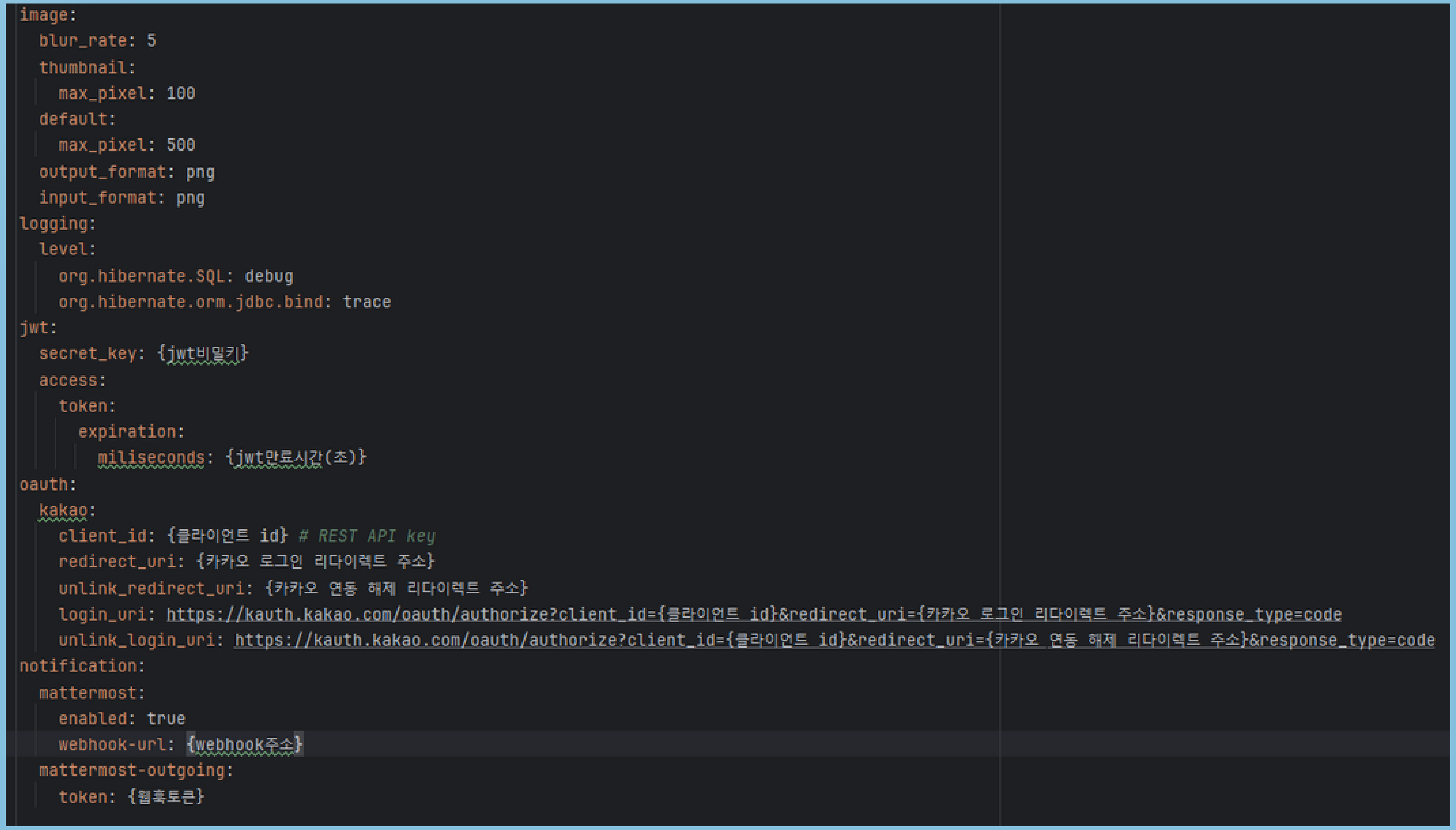
```
REACT_APP_BACKEND_API_URL="백엔드 도메인"
REACT_APP_KAKAO_NATIVE_APP_KEY="카카오 앱 키"
REACT_APP_KAKAO_RESTAPI_KEY="카카오 REST API키"
REACT_APP_KAKAO_JAVASCRIPT_KEY="카카오 JAVASCRIPT키"
REACT_APP_KAKAO_ADMIN_KEY="카카오 ADMIN키"
REACT_APP_BACKEND_SOCKET_URL="백엔드 소켓 URL"
```

환경 변수-BACKEND

PATH-/project/backend/src/main/resources/application.yml



```yaml
server:
  port: 8001
  servlet:
    context-path: /backend
spring:
  datasource:
    url: jdbc:mysql://{db도메인}/{db이름}?useSSL=false&useUnicode=true&serverTimezone={db타임존}
    username: {유저이름}
    password: {비밀번호}
    driver-class-name: com.mysql.cj.jdbc.Driver
  jpa:
    hibernate:
      ddl-auto: none
    properties:
      hibernate:
        show_sql: true
        format_sql: true
        dialect: org.hibernate.dialect.MySQLDialect
  servlet:
    multipart:
      max-file-size: 10MB
      max-request-size: 10MB
  cloud:
    aws:
      credentials:
        access-key: {aws접근키}
        secret-key: {aws비밀키}
      region:
        static: {지역}
      s3:
        bucket: {버킷이름}
      stack:
        auto: false
      presigned_exp: {presigned url 만료시간(분)}
```



```yaml
image:
  blur_rate: 5
  thumbnail:
    max_pixel: 100
  default:
    max_pixel: 500
  output_format: png
  input_format: png
logging:
  level:
    org.hibernate.SQL: debug
    org.hibernate.orm.jdbc.bind: trace
jwt:
  secret_key: {jwt비밀키}
  access:
    token:
      expiration:
        miliseconds: {jwt만료시간(초)}
oauth:
  kakao:
    client_id: {클라이언트 id} # REST API key
    redirect_uri: {카카오 로그인 리다이렉트 주소}
    unlink_redirect_uri: {카카오 연동 해제 리다이렉트 주소}
    login_uri: https://kauth.kakao.com/oauth/authorize?client_id={클라이언트 id}&redirect_uri={카카오 로그인 리다이렉트 주소}&response_type=code
    unlink_login_uri: https://kauth.kakao.com/oauth/authorize?client_id={클라이언트 id}&redirect_uri={카카오 연동 해제 리다이렉트 주소}&response_type=code
notification:
  mattermost:
    enabled: true
    webhook-url: {webhook주소}
  mattermost-outgoing:
    token: {웹훅토큰}
```

구동 환경 설치

1.도커 설치

운영체제: UBUNTU 20.04.6 LTS

1-1 HTTP 설치

```
$ sudo apt update
$ sudo apt-get install -y ca-certificates ₩
    curl ₩
    software-properties-common ₩
    apt-transport-https ₩
    gnupg ₩
    lsb-release
```

1-2 GPG 키 및 저장소 추가

```
$ sudo mkdir -p /etc/apt/keyrings
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/etc/apt/keyrings/docker.gpg

$ echo ₩
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu ₩
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

구동 환경 설치

1-3 도커 엔진 설치(특정 버전 명시)

```
$ sudo apt update
$ sudo apt-get install docker-ce=<VERSION_STRING> docker-ce-cli=<VERSION_STRING> containerd.io
```

1-4 도커 조회

```
$ sudo docker version
$ sudo docker compose version
```

2.도커 컴포즈 설치

2-1 도커 컴포즈 설치(특정 버전 명시 가능)

```
$ sudo curl -L "https://github.com/docker/compose/releases/download/1.24.1/docker-compose-$(uname
-s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

구동 환경 설치

2-2 도커 컴포즈 권한 설정

```
$ sudo chmod +x /usr/local/bin/docker-compose
```

2-3 심볼릭 링크 설정

```
$ sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
```

2-4 도커 컴포즈 설치 확인

```
$ docker-compose -version
```

구동 환경 설치

3.도커 컴포즈 파일 세팅

3-1 back-docker-compose.yml 생성

```
$ sudo vi back-docker-compose.yml
```

```
version: "3.7"
services:
  backend:
    image: codakcodak.site:5000/backend:0.1
    container_name: backend
    restart: always
    ports:
      - "0.0.0.0:8001:8001"
```

3-2 back-run-deploy.sh 생성

```
$ sudo vi back-run-deploy.sh
```

```
echo killing old docker processes
docker compose -f ./back-docker-compose.yml rm -fs

echo removing all volumes
yes | docker volume prune -a
yes | docker image prune -a

echo building docker containers
docker compose -f ./back-docker-compose.yml up --build -d
```

구동 환경 설치

3-3 front-docker-compose.yml 생성

```
$ sudo vi front-docker-compose.yml
```

```
version: "3.7"
services:
  nginx:
    image: codakcodak.site:5000/front:0.1
    container_name: nginx
    restart: always
    ports:
      - "0.0.0.0:80:80"
      - "0.0.0.0:443:443"
    volumes:
      - ./data/certbot/conf:/etc/letsencrypt
      - ./data/certbot/www:/var/www/certbot
    platform: linux/amd64

  certbot:
    container_name: certbot
    image: certbot/certbot
    restart: unless-stopped
    volumes:
      - ./data/certbot/conf:/etc/letsencrypt
      - ./data/certbot/www:/var/www/certbot
    entrypoint: "/bin/sh -c 'trap exit TERM; while :; do certbot renew; sleep 12h & wait $${!}; done;'"
```

구동 환경 설치

3-4 front-run-deploy.sh 생성

```
$ sudo vi front-run-deploy.sh
```

```
echo killing old docker processes
docker compose -f ./front-docker-compose.yml rm -fs

echo removing all volumes
yes | docker volume prune -a
yes | docker image prune -a

echo building docker containers
docker compose -f ./front-docker-compose.yml up --build -d
```

구동 환경 설치

3-5 https인증서 코드 생성

```
sudo vi init-letsencrypt.sh
```

https 적용 Let's Encrypt

지금까지 서버를 생성해서 dns까지 적용해서 사용하다가 위치 기반을 사용할 일이 생겼다. 자동으로 현재 위치를 불러올 수 있도록 하려면 https를 사용해야만 가능하더라. 회원가입때...

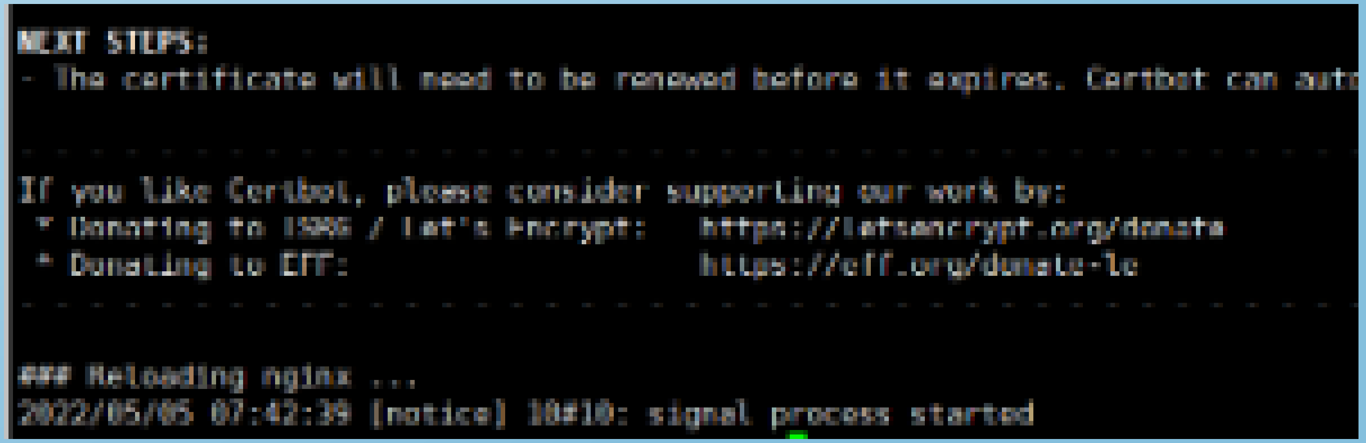
velog.io

https://velog.io/@ililil9482/https-%EC%A0%81%EC%9A%A9-Lets-Encrypt

해당 블로그를 참고하여 init-letsencrypt.sh의 내용을 기입

3-6 https인증서 발급

```
sudo bash init-letsencrypt.sh
```



```
NEXT STEPS:
- The certificate will need to be renewed before it expires. Certbot can auto

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
If you like Certbot, please consider supporting our work by:
 * Donating to ISRG / Let's Encrypt:   https://letsencrypt.org/donate
 * Donating to EFF:                    https://eff.org/donate-le
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

### Reloading nginx ...
2022/05/05 07:42:39 [notice] 10#10: signal process started
```

발급 성공시 로그화면

프로젝트 구동

1. 프론트엔드 구동

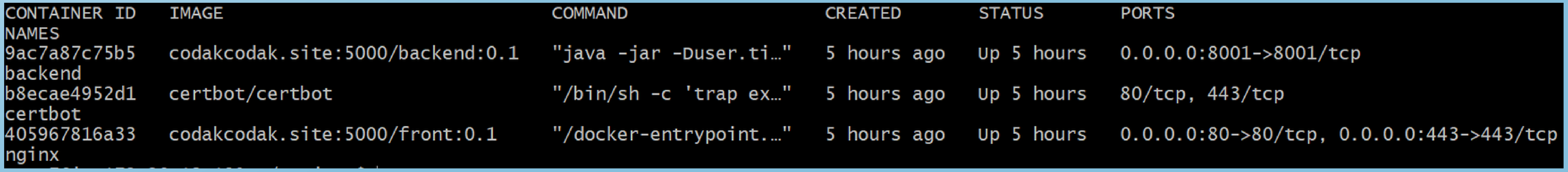
```
sudo bash front-run-deploy.sh
```

2. 백엔드 구동

```
sudo bash back-run-deploy.sh
```

3. 도커 프로세스 확인

```
sudo docker ps
```



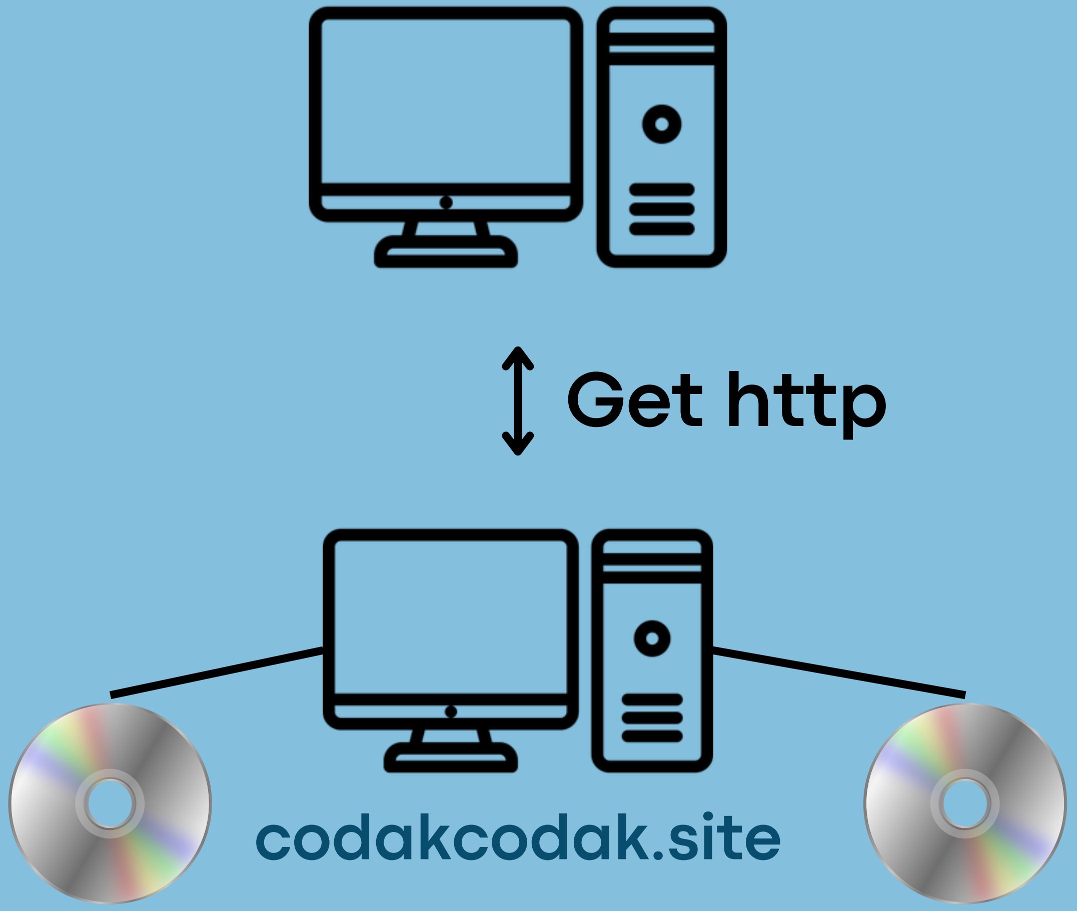
```
CONTAINER ID   IMAGE                              COMMAND                  CREATED         STATUS         PORTS
NAMES
9ac7a87c75b5   codakcodak.site:5000/backend:0.1   "java -jar -Duser.ti…"   5 hours ago     Up 5 hours     0.0.0.0:8001->8001/tcp
backend
b8ecae4952d1   certbot/certbot                    "/bin/sh -c 'trap ex…"   5 hours ago     Up 5 hours     80/tcp, 443/tcp
certbot
405967816a33   codakcodak.site:5000/front:0.1     "/docker-entrypoint.…"   5 hours ago     Up 5 hours     0.0.0.0:80->80/tcp, 0.0.0.0:443->443/tcp
nginx
```

참고사항

- docker image

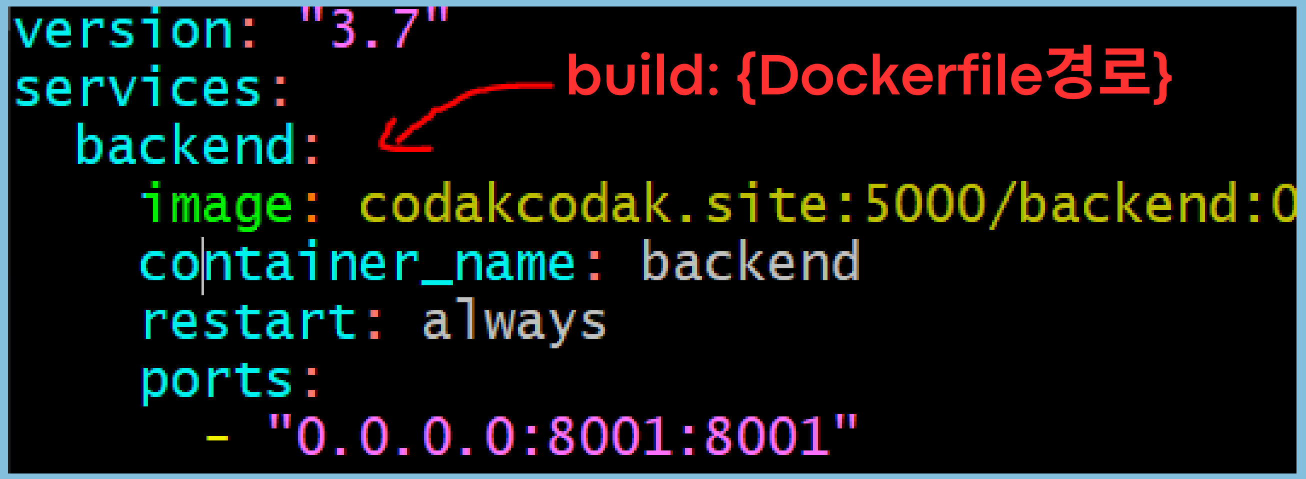
현재 빌드된 도커 이미지들은 codakcodak.site에서 관리

codakcodak.site에서 private image hub인 registry를 운영하여 프로젝트를 구동하는 환경에서 원격으로 이미지를 받아와 실행



- docker image build

codakcodak.site에서 image들을 관리하고 있지만 직접 빌드하여 프로젝트를 구동할 경우 back-docker-compose.yml, front-docker-compose.yml의 image항목 수정 필요

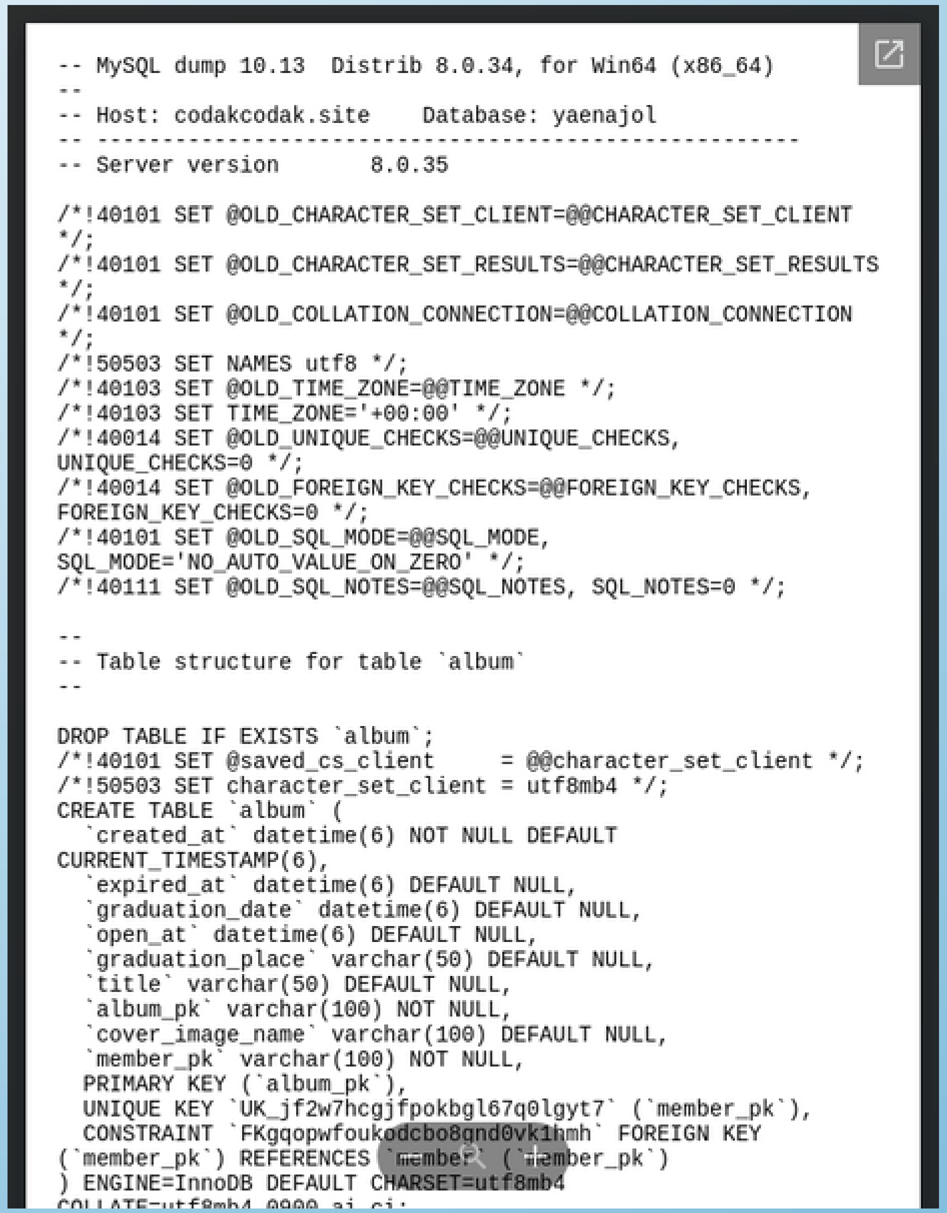


```
version: "3.7"
services:
  backend:
    image: codakcodak.site:5000/backend:0
    container_name: backend
    restart: always
    ports:
      - "0.0.0.0:8001:8001"
```

build: {Dockerfile경로}

DATABASE

sql 파일

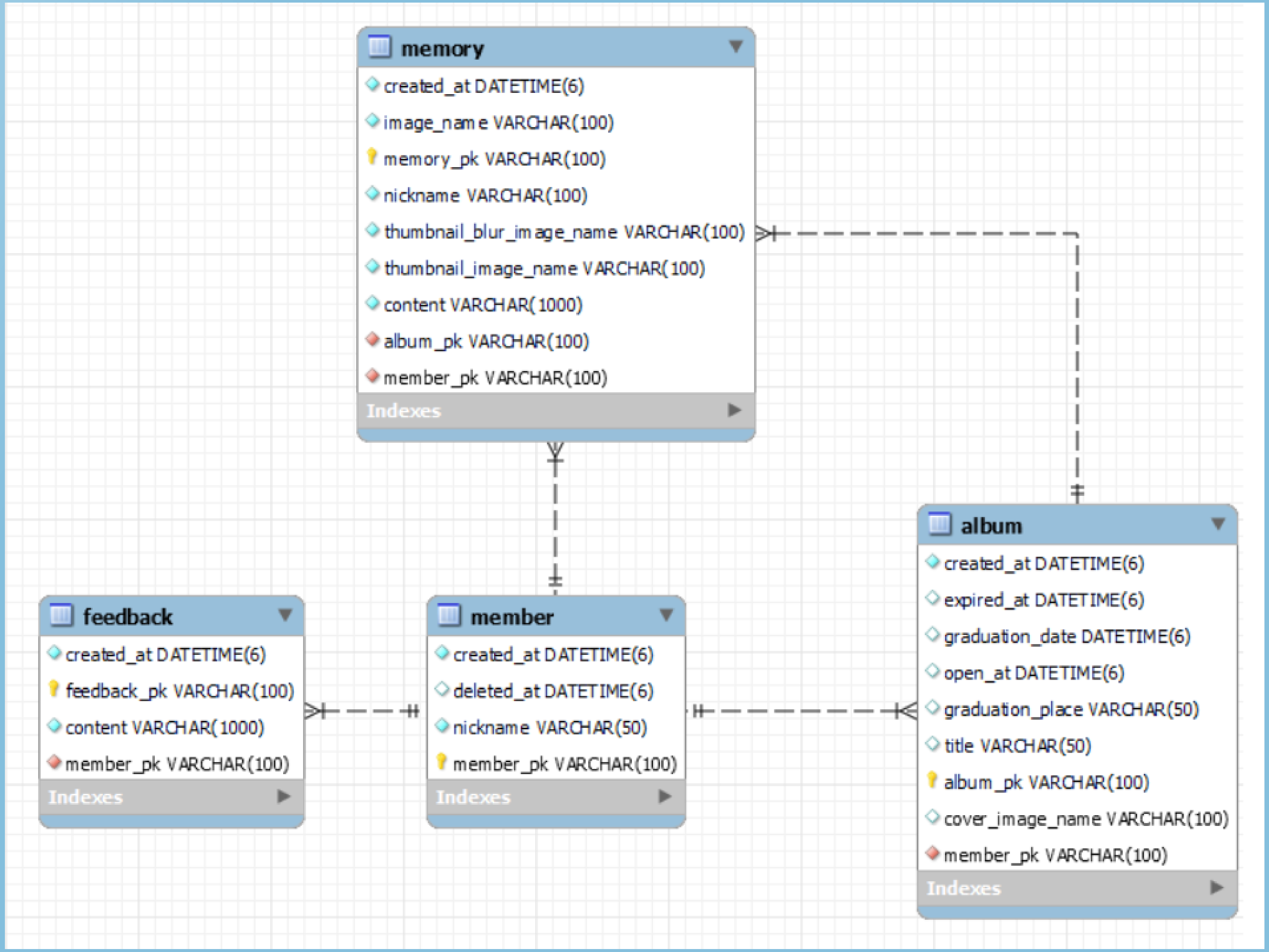


```
-- MySQL dump 10.13  Distrib 8.0.34, for Win64 (x86_64)
--
-- Host: codakcodak.site    Database: yaenajol
-- ------------------------------------------------------
-- Server version       8.0.35

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT
*/;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS
*/;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION
*/;
/*!50503 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS,
UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `album`
--

DROP TABLE IF EXISTS `album`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `album` (
  `created_at` datetime(6) NOT NULL DEFAULT
CURRENT_TIMESTAMP(6),
  `expired_at` datetime(6) DEFAULT NULL,
  `graduation_date` datetime(6) DEFAULT NULL,
  `open_at` datetime(6) DEFAULT NULL,
  `graduation_place` varchar(50) DEFAULT NULL,
  `title` varchar(50) DEFAULT NULL,
  `album_pk` varchar(100) NOT NULL,
  `cover_image_name` varchar(100) DEFAULT NULL,
  `member_pk` varchar(100) NOT NULL,
  PRIMARY KEY (`album_pk`),
  UNIQUE KEY `UK_jf2w7hcgjfpokbgl67q0lgyt7` (`member_pk`),
  CONSTRAINT `FKgqopwfoukodcbo8gnd0vk1hmh` FOREIGN KEY
(`member_pk`) REFERENCES `member` (`member_pk`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;
```

https://drive.google.com/file/d/1UCi73EHmDZt7C6bF2fqy9SNxtXEdlpZk/view?usp=sharing

클릭 하면 다운로드 페이지로 이동

erd


memory
created_at DATETIME(6)
image_name VARCHAR(100)
memory_pk VARCHAR(100)
nickname VARCHAR(100)
thumbnail_blur_image_name VARCHAR(100)
thumbnail_image_name VARCHAR(100)
content VARCHAR(1000)
album_pk VARCHAR(100)
member_pk VARCHAR(100)
Indexes
album
created_at DATETIME(6)
expired_at DATETIME(6)
graduation_date DATETIME(6)
open_at DATETIME(6)
graduation_place VARCHAR(50)
title VARCHAR(50)
album_pk VARCHAR(100)
cover_image_name VARCHAR(100)
member_pk VARCHAR(100)
Indexes
feedback
created_at DATETIME(6)
feedback_pk VARCHAR(100)
content VARCHAR(1000)
member_pk VARCHAR(100)
Indexes
member
created_at DATETIME(6)
deleted_at DATETIME(6)
nickname VARCHAR(50)
member_pk VARCHAR(100)
Indexes