20\_08\_07 학습내용 – 드디어 Docker 개발환경 구성이 완료되었다.!

docker-compose.yml에 대한 설명 p2 ~ p4

1. mysql  
    table 받을 폴더 마운팅 / 계정 설정 p3
2. php   
    depends on mysql컨테이너 p3  
    .php 파일 동작을 위한 폴더 마운팅 p3
3. nginx   
    build : dockerfile p3  
    depends on / links 설정 p4  
    html, php를 실행시키기 위한 폴더 마운팅 p4  
    default.conf 작성 후 폴더 마운팅하기 p4

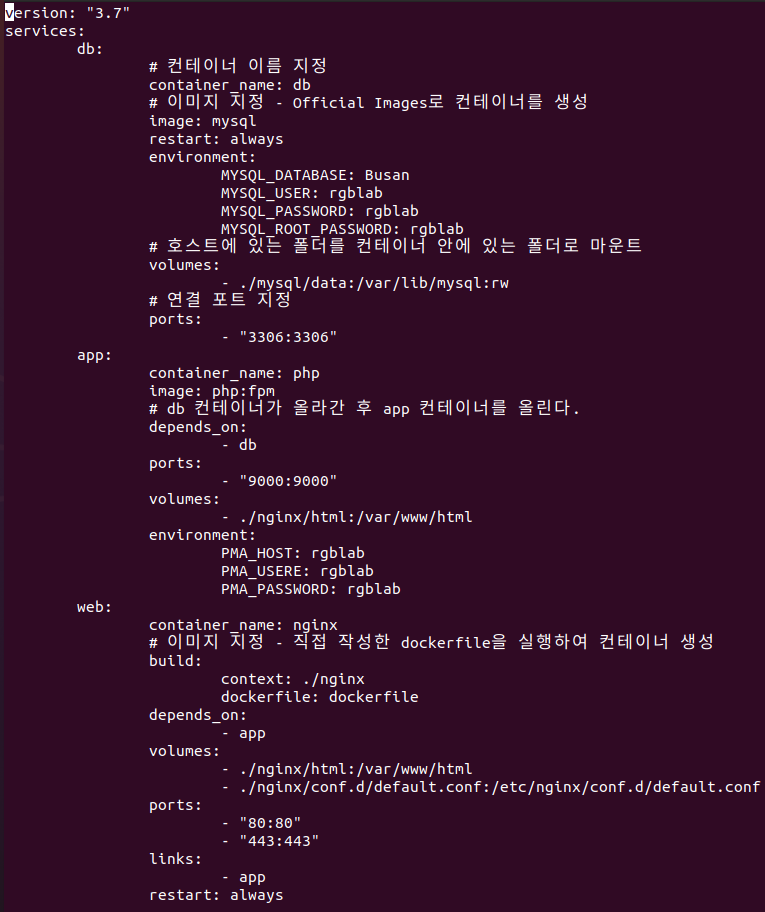
mysql 사용하기

1. mysql 컨테이너 접근하기 p5
2. 컨테이너 접근 후 mysql 접속하기 p5
3. 테이블 생성 p5 ~ p6

생성한 테이블을 index.php에서 보여주기

1. php 문법 중 mysqli 클래스 에러 문제 p7
2. docker 수정  
    php디렉토리 생성 및 해당 디렉토리 안에 php용 dockerfile 생성 p8  
    (dockerfile 내부에 mysqli 설치 코드 포함하였음)  
    docker-compose.yml 수정 ( image -> build로 수정) p8
3. mysqli 설치 완료 확인 p9
4. index.php를 앞서 생성한 mysql table들을 출력하도록 변경 후 확인 p9 ~ 10
5. VM 외부 실제 PC에서도 접속 되는지 확인 p11

docker-compose.yml



docker-compose.yml에서

1. Mysql은
2. db 데이터를 받아올 폴더를 마운팅해야함

volumes:

- ./mysql/data (내 폴더 위치임) :/var/lib/mysql:rw

1. 계정 설정

environment:

MYSQL\_DATABASE: Busan

MYSQL\_USER: rgblab

MYSQL\_PASSWORD: rgblab

MYSQL\_ROOT\_PASSWORD: rgblab

1. PHP는
2. Mysql에 depends on임을 밝혀야함

depends\_on: *mysql컨테이너이름*

1. .php 가 있는 폴더를 마운팅 해야함. (#php작동을 위해 마운팅 필요한듯)

volumes:

- ./nginx/html:/var/www/html

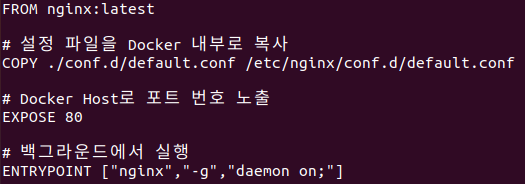
1. Nginx는 (얘가 어려움)
2. 여기서 이미지 지정 대신 build 명령어 사용해서 컨테이너 생성함

build:

context: ./nginx # dockerfile이 있는 폴더

dockerfile: dockerfile

========================



ENTRYPOINT : 컨테이너 생성시 명령어, CMD랑 결과 차이 거의 없음

다만 docker run시 추가 명령어는 CMD로 들어간다 그정도??

================================

1. Php에 depends on 임을 밝히고, link 걸어줘야함

depends\_on :

* *php컨테이너이름*

links :

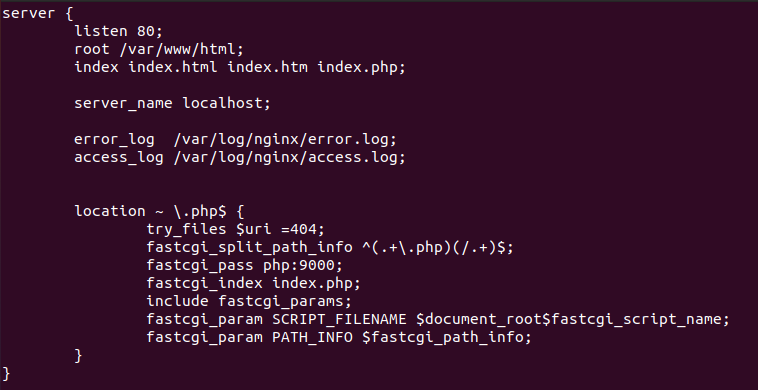
* *php컨테이너 이름*

1. nginx 폴더내에 있는 .php나 .html을 호스팅 하기 위한 마운팅 필요

Default.conf에 index 있는데?? 지우고 함 돌려봐야겠다. 🡪 해봣는데 필요하더라  
지우면 404 notfound 뜹니다.

1. nginx 폴더내에 default.conf를 작성하고 마운팅 해줘야함.  
   ========================

Default.conf



Server 블록 부모는 http블록이지만 생략 가능한 듯 하다.  
 Server 블록은 하나의 웹 사이트를 선언하는데 사용됨

Location 블록은 server 블록 안에 등장하면서 특정 URL을 처리하는 방법을 정의한다 를들어 http://hello-world.com/c/1 과 http:// hello-world.com /m/1 로 접근하는 요청을 다르게 처리하고 싶을 때 사용한다.



Mysql 컨테이너 접근

docker exec -it db(컨테이너 이름입니다.) /bin/bash

Mysql 서버 로그인

mysql –u root –p

예제 DB 생성

use Busan;

drop table if exists Img\_Record;

drop table if exists Flight\_Record;

create table Flight\_Record(

Flight\_Num varchar(10) primary key,

Flight\_Day varchar(8),

Drone\_Model varchar(20),

Flight\_Start\_t varchar(5),

Flight\_End\_t varchar(5),

Flight\_Length float

);

create table Img\_Record(

ID int Auto\_increment Primary Key,

Flight\_Num varchar(10),

capture\_time varchar(20),

Latitude float,

Longitude float,

Altitude float,

ImgPath varchar(50),

Foreign Key (Flight\_Num) REFERENCES Flight\_Record(Flight\_Num)

);

insert into Flight\_Record Values ('2020080701','20200807','dev/test3','09:00','09:07',0.42);

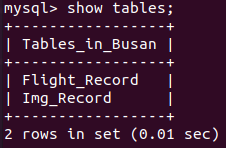
insert into Img\_Record Values (1,'2020080701','09:00',35.888232,128.605075,28.3,'http://');

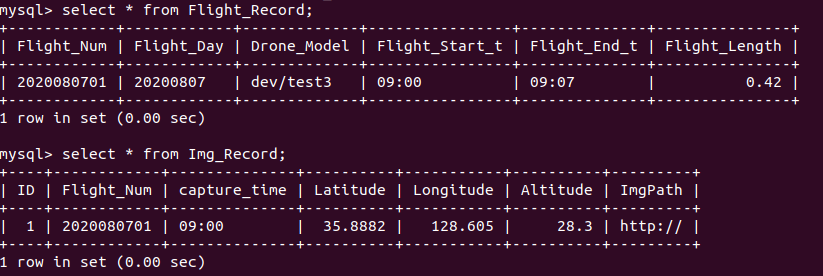
show tables;

select \* from img\_record;

select \* from flight\_record

결과



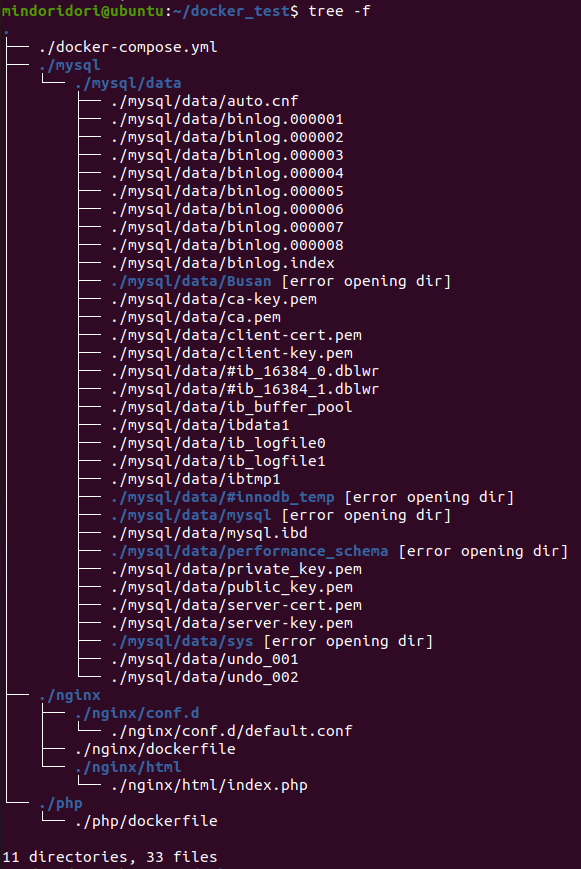


해당 결과를 php에서 보여주기

위에까지 진행한 후 php 파일을 돌렸을 때 mysql 접속 class인 mysqli가 없다고 나왔다. 그로 인해 php 컨테이너 생성 시 mysqli 클래스 설치를 해 주기 위해서

php 컨테이너 생성을 docker-compse.yml에서 image를 사용하여 생성하던 방식에서 dockerfile을 사용하는 방식으로 변경해 주었다.

수정은 다음과 같이 이루어졌다.



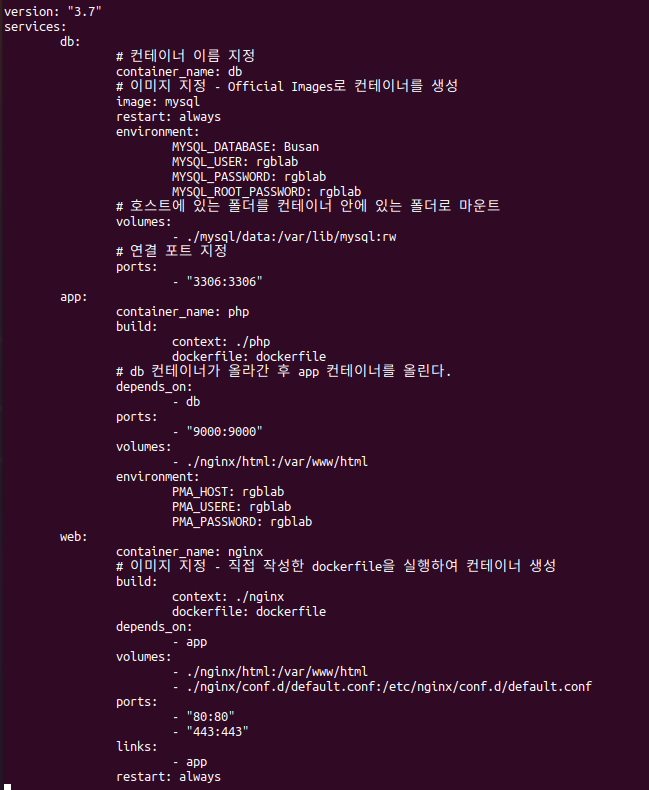
트리를 보면 알 수 있듯이 php라는 폴더와  
 그 안에 php 컨테이너 생성을 위한 dockerfile를 새로 생성하였다.

Php 폴더 안 dockerfile 안의 내용은



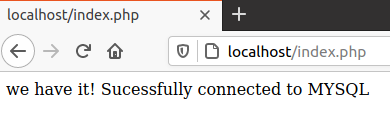
위와 같이 매우 간단하며,

Docker-compose.yml은



상자로 표시한 부분이 변경되었다.  
(원래는 build가 아닌 image : php:fpm 이었다..)

이렇게 변경하고 docker-compose up 후 filefox를 새로고침 해 보니



제대로 연결 되었음을 알 수 있다.

(위에 해당하는 php 코드는 아래와 같다.)



이제 db안의 내용을 불러다가 출력해 보자!

Php 코드를 변경해서 실행시켜보면

|  |  |
| --- | --- |
|  |  |

결과가 잘 나오는 것을 알 수 있다!!

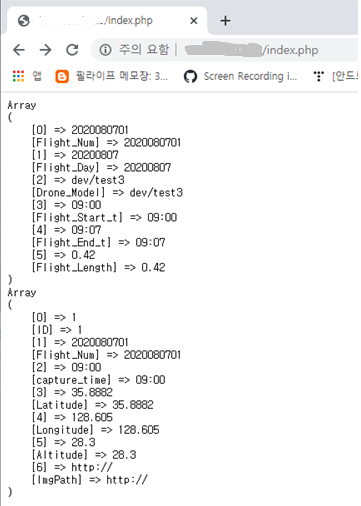
앞서 만든 테이블 2개를 다 출력해 보면

|  |  |
| --- | --- |
|  |  |

또한 VM 내부에서 뿐만 아니라

VM 바깥 원래 PC에서도 접속이 가능하다.

(윈도우 10 환경에서 접속 가능)



단, 현재 VM 네트워크 설정이 NAT으로 되어있기에 내 PC에서만 접속이 가능함. (다른 pc불가)

nginx>html에 .html이 있다면 .php뿐만아니라 html도 가능함  
