Lab. Docker Compose Lab

1. Flask App을 Docker Compose로 실행하기

    1)Flask Container

        -Connection Port : 5000

        -Redis Host Name : redis

    2)Redis Container

        -Image : redis

    3)app.py

```
import time

import redis
from flask import Flask

app = Flask(__name__)
cache = redis.Redis(host='redis', port=6379)


def get_hit_count():
    retries = 5
    while True:
        try:
            return cache.incr('hits')
        except redis.exceptions.ConnectionError as exc:
            if retries == 0:
                raise exc
            retries -= 1
            time.sleep(0.5)


@app.route('/')
def hello():
    count = get_hit_count()
    return 'Hello World! I have been seen {} times.\n'.format(count)
```

    4)requirements.txt

```
flask
redis
```

    5)Dockerfile

```
FROM            python:3.7-alpine
WORKDIR         /code
ENV             FLASK_APP app.py
ENV             FLASK_RUN_HOST 0.0.0.0
RUN             apk add --no-cache gcc musl-dev linux-headers
COPY            requirements.txt requirements.txt
RUN             pip install -r requirements.txt
COPY            . .
CMD             ["flask", "run"]
```

    6)확인 순서

        -flask Application을 Build하여 Image를 생성

        -50000 Port로 접속할 수 있게 docker-compose.yml 작성

        -Docker Compose를 실행

    7)Code

```
$ mkdir demo
$ cd demo
$ vim app.py
$ vim requirements.txt
$ vim Dockerfile

$ docker build -t flask-redis .

$ vim docker-compose.yml
    version: '3'

    services:
      flask:
        image: flask-redis
        ports:
          - 50000:5000
      redis:
        image: redis

$ docker-compose up
```

        -Web Browser에서 확인

            -http:{IP}:50000

2. Front-end, Back-end, Database로 구성된 방명록 서비스 실행하기

1)Front-end
- Image : subicura/guestbook-frontend:latest
- Port : 60000
- PORT 환경변수 : Service를 실행할 Port
- GUESTBOOK_API_ADDR 환경변수 : Back-end Server 주소 ex)backend:8000

2)Back-end
- Image : subicura/guestbook-backend:latest
- PORT 환경변수 : Service를 실행할 Port
- GUESTBOOK_DB_ADDR 환경변수 : Database Server 주소 ex)mongodb:27017

3)Database
- Image : mongo:4
- 연결되는 Port : 27017
- Volume 설정 : /data/db

4)Code

```
$ mkdir demo
$ cd demo
$ vim docker-compose.yml

    version: '3'

    services:
      frontend:
        image: subicura/guestbook-frontend:latest
        ports:
          - 60000:3000
        environment:
          - PORT=3000
          - GUESTBOOK_API_ADDR=backend:5000
        depends_on:
          - backend
      backend:
        image: subicura/guestbook-backend:latest
        environment:
          - PORT=5000
          - GUESTBOOK_DB_ADDR=mongodb:27017
        depends_on:
          - mongodb

      mongodb:
        image: mongo:4
        volumes:
          - db_data:/data/db   <---띄우지 말것

    volumes:
      db_data: {}

$ docker-compose up
```

-Web Browser에서
    -http://{IP}:60000