Lab6. Django Project Automatic Deployment with Jenkins II

1. Django Project Dockerizing Version 1

1)demo 디렉토리 생성

```
$ mkdir demo
$ cd demo
```

2)dockerfile 작성하기

```
$ nano dockerfile
FROM        python:latest
RUN         apt-get update
RUN         apt-get -y upgrade
RUN         apt-get -y dist-upgrade
RUN         apt install -y nginx
RUN         git clone  https://github.com/gitinstructor/DjangoHome.git
WORKDIR     /DjangoHome/myproject/
RUN         pip install --upgrade pip
RUN         pip install -r requirements.txt
RUN         pip install gunicorn  uwsgi

EXPOSE      80

CMD         ["bash", "-c", "python manage.py runserver 0:80"]
```

3)build

```
$ docker build -t myweb:v1 .
```

4)이미지 확인

```
$ docker images
REPOSITORY      TAG         IMAGE ID        CREATED         SIZE
myweb           v1          dkfdfkddjd      33 seconds ago  1.03GB
```

5)이미지 Container로 실행하기

```
$ docker run -dp 80:80 myweb:v1
```

6)Container 확인하기

```
$ docker ps -a
```

7)Web Browser에서 확인하기

2. Django Project Dockerizing Version 2

1)uwsgi 이용하기

2)dockerfile 수정하기

```
FROM        python:latest

RUN         apt-get update
RUN         apt-get -y upgrade
RUN         apt-get -y dist-upgrade
RUN         apt install -y nginx
RUN         git clone  https://github.com/gitinstructor/DjangoHome.git

WORKDIR     /DjangoHome/myproject/
RUN         pip install --upgrade pip
RUN         pip install -r requirements.txt
RUN         pip install gunicorn  uwsgi
RUN         mkdir -p /var/log/uwsgi/myproject/
COPY        ./myproject.ini    /DjangoHome/myproject/.config/uwsgi/myproject.ini
CMD         ["uwsgi", "--ini", "./.config/uwsgi/myproject.ini"]

EXPOSE      80
```

3)이미지에서 사용할 myproject.ini 새로 생성하기

```
[uwsgi]
chdir = /DjangoHome/myproject/

module = myproject.wsgi:application

uid = root
gid = root

http = :80

enable-threads = true
master = true
vacuum = true
pidfile = /tmp/myproject.pid
logto = /var/log/uwsgi/myproject/@(exec://date +%%Y-%%m-%%d).log
log-reopen = true
```

4)Container, Image 모두 지우기

```
$ docker rm -f `docker ps -a -q`
$ docker rmi -f `docker images -q`
```

```
    $ docker volume prune
    $ docker system prune -a

  5)build
    $ docker build -t myweb:v2 .

  6)이미지 확인
    $ docker images
    REPOSITORY      TAG       IMAGE ID       CREATED         SIZE
    myweb           v2        dkfdfkddjd     33 seconds ago  1.03GB

  7)이미지 Container로 실행하기
    $ docker run -dp 80:80 myweb:v2

  8)Container 확인하기
    $ docker ps -a

  9)Web Browser에서 확인하기


3. Django Project Dockerizing Version 3
  1)Nginx와 연동하기
  2)nginx.conf 파일 생성하기
    $ nano nginx.conf
      user root;          <---여기가 중요
      worker_processes auto;
      pid /run/nginx.pid;
      include /etc/nginx/modules-enabled/*.conf;

      events {
            worker_connections 768;
            # multi_accept on;
      }

      http {

            ##
            # Basic Settings
            ##

            sendfile on;
            tcp_nopush on;
            types_hash_max_size 2048;
            # server_tokens off;

            # server_names_hash_bucket_size 64;
            # server_name_in_redirect off;

            include /etc/nginx/mime.types;
            default_type application/octet-stream;

            ##
            # SSL Settings
            ##

            ssl_protocols TLSv1 TLSv1.1 TLSv1.2 TLSv1.3; # Dropping SSLv3, ref: POODLE
            ssl_prefer_server_ciphers on;

            ##
            # Logging Settings
            ##

            access_log /var/log/nginx/access.log;
            error_log /var/log/nginx/error.log;

            ##
            # Gzip Settings
            ##

            gzip on;

            # gzip_vary on;
            # gzip_proxied any;
            # gzip_comp_level 6;
            # gzip_buffers 16 8k;
            # gzip_http_version 1.1;
            # gzip_types text/plain text/css application/json application/javascript text/xml application/xml application/xml+rss
            text/javascript;

            ##
            # Virtual Host Configs
            ##

            include /etc/nginx/conf.d/*.conf;
```

```
        include /etc/nginx/sites-enabled/*;
    }
```

3)dockerfile 수정하기

```
FROM        python:latest

RUN         apt-get update
RUN         apt-get -y upgrade
RUN         apt-get -y dist-upgrade
RUN         apt install -y nginx
COPY        ./nginx.conf    /etc/nginx/nginx.conf

RUN         git clone  https://github.com/gitinstructor/DjangoHome.git
WORKDIR     /DjangoHome/myproject/
RUN         cp .config/nginx/myproject.conf /etc/nginx/sites-enabled/
CMD         ["nginx", "-g", "daemon on;"]

RUN         pip install --upgrade pip
RUN         pip install -r requirements.txt
RUN         pip install gunicorn  uwsgi
RUN         mkdir -p /var/log/uwsgi/myproject
COPY        ./myproject.ini    /DjangoHome/myproject/.config/uwsgi/myproject.ini

RUN         service nginx restart
CMD         ["uwsgi", "--ini", "./.config/uwsgi/myproject.ini"]

EXPOSE      80
```

4)myproject.ini 수정하기

```
[uwsgi]
chdir = /DjangoHome/myproject/

module = myproject.wsgi:application

uid = root
gid = root

http = :80                          <---- 삭제할 것
socket = /tmp/myproject.sock        <--- 새로 추가
chmod-socket = 666                  <--- 새로 추가
chown-socket = root:root            <--- 새로 추가

enable-threads = true
master = true
vacuum = true
pidfile = /tmp/myproject.pid
logto = /var/log/uwsgi/myproject/@(exec://date +%%Y-%%m-%%d).log
log-reopen = true
```

5)Container, Image 모두 지우기

```
$ docker rm -f `docker ps -a -q`
$ docker rmi -f `docker images -q`
$ docker volume prune
$ docker system prune -a
```

6)build

```
$ docker build -t myweb:v3 .
```

7)이미지 확인

```
$ docker images
REPOSITORY      TAG         IMAGE ID        CREATED         SIZE
myweb           v3          dkfdfkddjd      33 seconds ago  1.03GB
```

8)이미지 Container로 실행하기

```
$ docker run -dp 80:80 myweb:v3
```

9)Container 확인하기

```
$ docker ps -a
```

10)Web Browser에서 확인하기

11)하지만, 정상적으로 서비스가 진행되고 있지 않다. 그 이유는 nginx 서비스가 restart가 되지 않았기 때문이다.

12)그래서, Container ID로 들어가서 다음과 같이 명령을 실행하면 정상적으로 서비스가 진행된다.

```
$ docker exec -it dk(Container ID) bash
root@...:/DjangoHome/myproject# service nginx restart
Restarting nginx: nginx.
root@...:/DjangoHome/myproject# exit
```

13)Web Browser에서 확인하기

4. Django Project Dockerizing Version 4

1)dockerfile 수정하기

```
FROM        python:latest
```

```
RUN             apt-get update
RUN             apt-get -y upgrade
RUN             apt-get -y dist-upgrade

RUN             git clone  https://github.com/gitinstructor/DjangoHome.git
WORKDIR         /DjangoHome/myproject/

RUN             pip install --upgrade pip
RUN             pip install -r requirements.txt
RUN             pip install gunicorn  uwsgi
RUN             mkdir -p /var/log/uwsgi/myproject
COPY            ./myproject.ini     /DjangoHome/myproject/.config/uwsgi/myproject.ini

CMD             ["gunicorn", "--bind", "0:8000", "myproject.wsgi:application"]

EXPOSE          8000
```

2)docker-compose.yml 생성하기

```
$ nano docker-compose.yml
    version: '3'
    services:

      django:
        container_name: django
        build:
          context: .
          dockerfile: ./dockerfile
         restart: always
         ports:
           - 80:8000
```

3)docker-compose로 build하기

```
$ docker-compose up -d
```

4)이미지 확인

```
$ docker images
REPOSITORY        TAG         IMAGE ID         CREATED                SIZE
demo-django       latest      ac378c01d36b   About a minute age    1.02GB
```

5)Web Browser에서 확인하기