Using Docker for Django Web Development

Kevin Supendi 13514094
School of Electrical Engineering and Informatics
Informatics / Computer Science
Bandung Institute of Technology
Bandung, Indonesia
13514094@std.stei.itb.ac.id

***Abstract*—Docker is new and growing software. It brought new concept of virtualization and various uses in IT. This paper focuses on Docker use case for Django framework, especially for "environment" problem in application development stage.**

***Keywords*—Docker, Django, environment, development**

I. INTRODUCTION

Developers often have a hard time setting up their development environment. This is especially true if the project they are working on has many dependencies, so every member has to install all the dependencies, even though each person has their own pace when they are working. Here is when Docker comes to play.

Docker is a new way of virtualization, it is more lightweight than the other virtual machine such as Oracle VirtualBox and VMWare. It can boot up in seconds, and consumes only dozens of MB of memory. Docker provides isolated environment. Just make the configurations in Docker, and it could be shared to others, and everyone will have the same environment through Docker.

Django has several dependencies, although not much, it is enough to show the advantage of using Docker for development.

II. THEORY

Docker brought a concept called containers. Unlike Virtual Machines, containers do not bundle the whole operating system, only the required settings and the bare minimum. Using containers, developers could choose which environment they want to use and they can switch easily between containers because containers are light.

There are other features of Docker, which permits cooperation between containers. The feature is called Docker Compose. With Docker Compose, one could run phpmyadmin container and MySQL container at the same time, then make them communicate to each other, so phpmyadmin application could use the database from MySQL container.

Docker opens many possibilities of uses, and this paper will try to setup the environment for Django Web Development using Docker Compose. The configurations can be uploaded and shared among developers, and it is reusable and customizable.

The purpose of this project is to configure phpmyadmin, Django Web framework and MySQL using Docker Compose.

III. METHOD

Docker and Docker Compose must be installed on the machine. Finish setup until Docker command is recognized by Command Prompt (or Terminal in Linux system).

A. *Define the project components*

The components for this project are Dockerfile and docker-compose.yml. Place them inside an empty project directory.

```
FROM python:2.7
RUN mkdir /code
WORKDIR /code
RUN pip install mysql-python
RUN pip install django
```

Image 1 : Dockerfile

```
version: '2'
 services:
   db:
     image: mysql
     volumes:
       - db_data:/var/lib/mysql
     environment:
        MYSQL_ROOT_PASSWORD: docker
        MYSQL_DATABASE: root
        MYSQL_USER: root
        MYSQL_PASSWORD: docker
   web:
     build: .
     command: python manage.py runserver 0.0.0.0:8000
     volumes:
       - .:/code
     ports:
       - "8000:8000"
     depends_on:
       - db
   phpmyadmin:
     image: phpmyadmin/phpmyadmin
     restart: always
     links:
       - db:mysql
     ports:
       - "8181:80"
     environment:
        MYSQL_USERNAME: root
        MYSQL_ROOT_PASSWORD: docker
 volumes:
     db_data:
```

Image 2 : docker-compose.yml

B. Create Django project

Change directory to project directory, then run this command :
docker-compose run web django-admin.py startproject composeexample .

This command will instruct Docker Compose to run web image declared in docker-compose.yml, then execute “django-admin.py startproject composeexample .” inside the web image. This will generate Django default project files inside a folder named composeexample.

C. Connect the Database

Setup database connection by modifying Django settings file, in composeexample/settings.py

```
…

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'postgres',
        'USER': 'postgres',
        'HOST': 'db',
        'PORT': 5432,
    }
 }

…
```

Image 3 : a portion of settings.py

D. Running Docker Compose

After all the setup is done, run Docker Compose with the command:

```
docker-compose up
```

The shell will show server log, and now Django is accessible in port 8000 and phpmyadmin in port 8181. Find the ip of Docker Machine with command:

```
docker-machine ip default
```

IV. RESULTS

If the project has been configured correctly, Django default page and phpmyadmin will be accessible. Setup for Web Development is faster and reusable for other developers.

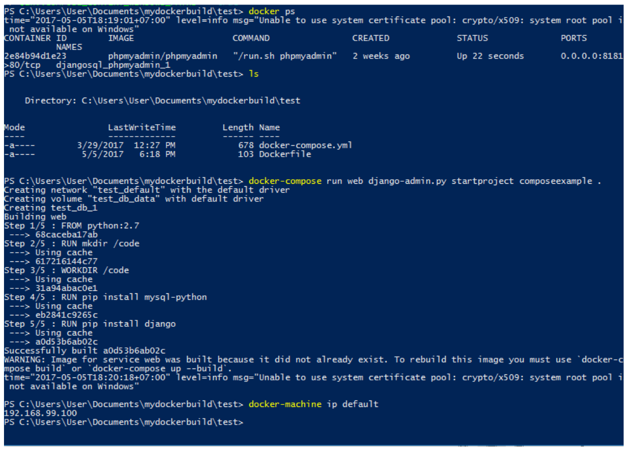
If the project need additional dependencies, just add them in Dockerfile, then push the Dockerfile to git. Any project member that has pulled the Dockerfile automatically installed the dependencies too. This could speed up the software development process and takes less time for setting up the environment.

The downside of Docker is the project members have to install and learn Docker first before they can use it. Once they do, they work and develop faster without worrying about configurations and settings.

V. CONCLUSION

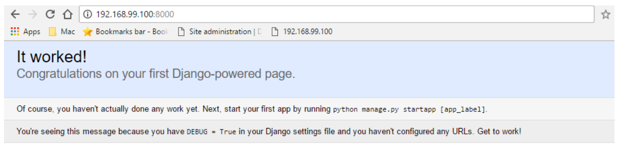
Docker is useful for projects with many dependencies. However, one must consider that not all member want to learn about Docker. Therefore, Docker maybe not so useful for little projects and it could slow down the development process instead.

VI. APPENDIX



```
PS C:\Users\User\Documents\mydockerbuild\test> docker ps
time="2017-05-05T18:19:01+07:00" level=info msg="Unable to use system certificate pool: crypto/x509: system root pool i
 not available on Windows"
CONTAINER ID        IMAGE                   COMMAND                  CREATED             STATUS              PORTS
          NAMES
2e84b94d1e23        phpmyadmin/phpmyadmin   "/run.sh phpmyadmin"   2 weeks ago         Up 22 seconds       0.0.0.0:8181
>80/tcp   djangosql_phpmyadmin_1
PS C:\Users\User\Documents\mydockerbuild\test> ls


    Directory: C:\Users\User\Documents\mydockerbuild\test


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----        3/29/2017  12:27 PM            678 docker-compose.yml
-a----         5/5/2017   6:18 PM            103 Dockerfile


PS C:\Users\User\Documents\mydockerbuild\test> docker-compose run web django-admin.py startproject composeexample .
Creating network "test_default" with the default driver
Creating volume "test_db_data" with default driver
Creating test_db_1
Building web
Step 1/5 : FROM python:2.7
 ---> 68caceba17ab
Step 2/5 : RUN mkdir /code
 ---> Using cache
 ---> 617216144c77
Step 3/5 : WORKDIR /code
 ---> Using cache
 ---> 31a94abac0e1
Step 4/5 : RUN pip install mysql-python
 ---> Using cache
 ---> eb2841c9265c
Step 5/5 : RUN pip install django
 ---> Using cache
 ---> a0d53b6ab02c
Successfully built a0d53b6ab02c
WARNING: Image for service web was built because it did not already exist. To rebuild this image you must use `docker-c
mpose build` or `docker-compose up --build`.
time="2017-05-05T18:20:18+07:00" level=info msg="Unable to use system certificate pool: crypto/x509: system root pool i
 not available on Windows"

PS C:\Users\User\Documents\mydockerbuild\test> docker-machine ip default
192.168.99.100
PS C:\Users\User\Documents\mydockerbuild\test>
```

Screenshot 1



192.168.99.100:8000

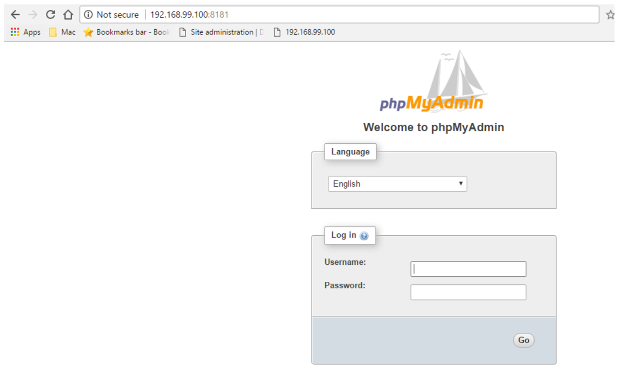
It worked!

Congratulations on your first Django-powered page.

Of course, you haven't actually done any work yet. Next, start your first app by running `python manage.py startapp [app_label]`.

You're seeing this message because you have `DEBUG = True` in your Django settings file and you haven't configured any URLs. Get to work!

Screenshot 2



Not secure | 192.168.99.100:8181

Screenshot 3

VII. REFERENCES

[1] https://www.docker.com/ accessed 5th May 2017 04.20 PM
[2] https://docs.docker.com/ accessed 5th May 2017 04.23 PM

VIII. STATEMENT

I hereby declare that this paper is my own work and not a copy, translation, nor plagiarism of somebody else's work.

Bandung, 5th May 2017

[illegible]

Kevin Supendi 13514094