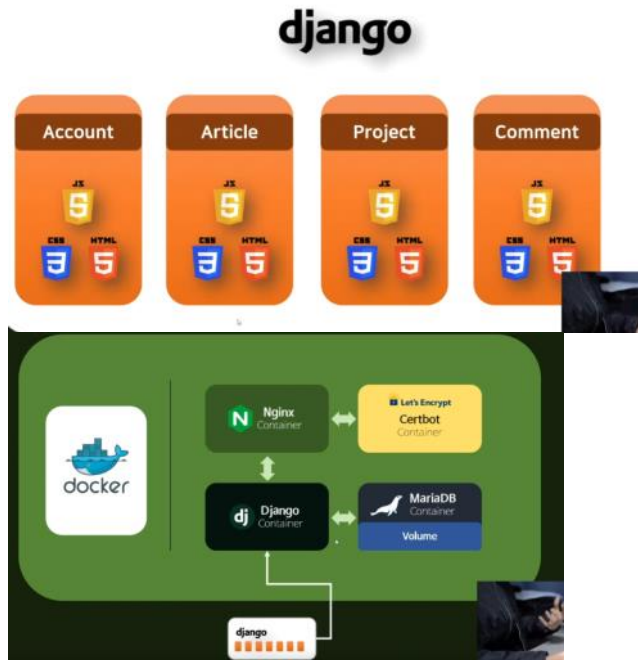


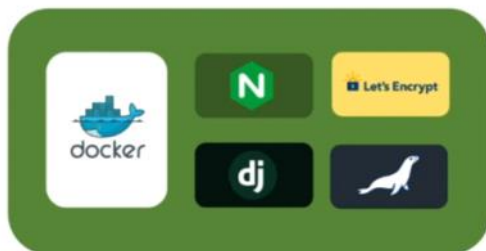
# What is DOCKER?: Service Deployment

2021년 10월 14일 목요일    오후 9:59

- Why Docker? 서비스 배포로 들어가며



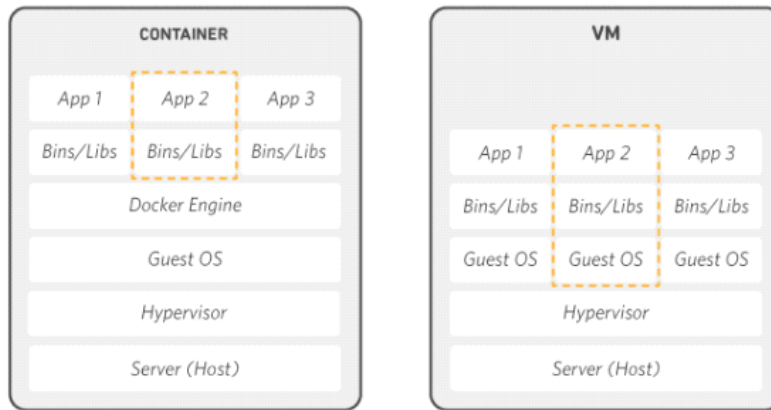
- 4개의 애플리케이션을 django container로 구성



- vultr 가상서버에 올려 서비스하는 것이 목표

- What is DOCKER?

- Virtualization(가상화) 기반 컨테이너라는 개념을 사용함. 실제 OS사용하는 것과 거의 동일한 성능



- 어느 곳에서든 동일한 환경에서 구동 가능.
- Develop Fast, Deploy Fast, Build Fast, Ship Fast, Maintain Fast
- Docker 없이 배포하는 경우, Deploy Environment Setting  
OS version match, Install required programs, Install python, Install library, Set env PATH, Deploy Test....
- Docker 활용 시 초기 셋팅하여 Image형태로 만들어 여러개의 Container로 구축하면 추가 설정이 불필요함.

#### • VPS 대여

- Vultr를 통해 VPS, 갓아 사설 서버 대여 및 접속
- VPS: Virtual Private Server
- Vultr 가입 후 Instance 생성

Choose Server



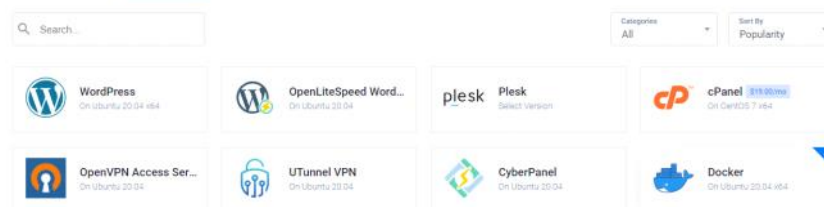
#### Server Location

All Locations America Europe Australia Asia

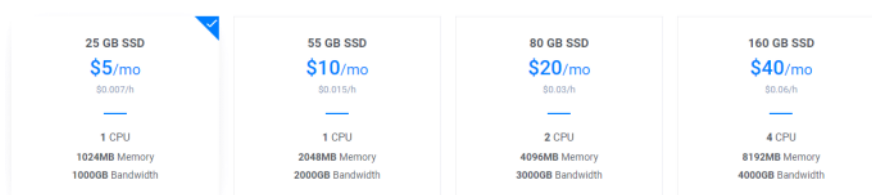


#### Server Type

64 bit OS 32 bit OS Marketplace Apps Upload ISO ISO Library Backup Snapshot



#### Server Size



## Products

Instances Snapshots ISOs Scripts DNS Block Storage Objects Firewall Network Load Balancers Kubernetes

Server added successfully!

Search...

Sort Location

<input type="checkbox"/> Server	OS	Location	Charges	Status
<input type="checkbox"/> Cloud Instance 1024 MB Cloud Compute		Seoul	—	Installing

Bandwidth Usage  
0GB / 1000GB

CPU Usage  
0%

Current Charges  
\$0.01

Location:	Seoul	CPU:	1 vCore	Label:	<a href="#">[Click here to set]</a>
IP Address:	158.247.206.129	RAM:	1024 MB	Tag:	<a href="#">[Click here to set]</a>
Username:	root	Storage:	25 GB SSD	OS:	Ubuntu 20.04 x64
Password:	.....	Bandwidth:	0 GB of 1000 GB	Application:	Docker on Ubuntu 20.04 x64

## 서버 접근 테스트

```
C:\Users\WDDubi>ssh root@158.247.206.129
The authenticity of host '158.247.206.129 (158.247.206.129)' can't be established.
ECDSA key fingerprint is SHA256:tk15U0INePPAheqD3k8BrIQMsSc60a60JhL8Lxx2s0I.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '158.247.206.129' (ECDSA) to the list of known hosts.
```

```
root@158.247.206.129's password:
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.4.0-84-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Thu 14 Oct 2021 02:25:00 PM UTC

System load:  0.05          Processes:            147
Usage of /:   13.6% of 23.41GB Users logged in:     0
Memory usage: 22%          IPv4 address for docker0: 172.17.0.1
Swap usage:   0%           IPv4 address for enp1s0: 158.247.206.129

59 updates can be applied immediately.
32 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

root@vultr:~#
```

## docker 설치 확인

```
root@vultr:~# docker

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Options:
  -c, --config string      Location of client config files (default "/root/.docker")
  -H, --host list          Daemon socket(s) to connect to
  -l, --log-level string   Set the logging level ("debug"|"info"|"warn"|"error"|"fatal") (default "info")
  --tls                    Use TLS; implied by --tlsverify
  --tlscacert string       Trust certs signed only by this CA (default "/root/.docker/ca.pem")
  --tlscert string         Path to TLS certificate file (default "/root/.docker/cert.pem")
  --tlskey string          Path to TLS key file (default "/root/.docker/key.pem")
  --tlsverify              Use TLS and verify the remote
  -v, --version            Print version information and quit

Management Commands:
  app*      Docker App (Docker Inc., v0.9.1-beta3)
  builder   Manage builds
  buildx*   Build with BuildKit (Docker Inc., v0.6.1-1-docker)
```

```
Management Commands:
  app*      Docker App (Docker Inc., v0.9.1-beta3)
  builder   Manage builds
  buildx*   Build with BuildKit (Docker Inc., v0.6.1-docker)
  config    Manage Docker configs
```

```
root@vultr:~# docker container ls
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
root@vultr:~#
```

# Docker Container, Image

2021년 10월 15일 금요일    오후 6:49

## • Docker GUI Portainer 컨테이너 생성

- Docker를 GUI 환경에서 실행할 수 있게 해주는 Portainer를 Docker hub 으로부터 이미지를 내려받아 구동
- <https://hub.docker.com/r/portainer/portainer-ce>
- <https://docs.portainer.io/v/ce-2.9/start/install/server/docker/linux>

## Deployment

First, create the volume that Portainer Server will use to store its database:

```
1 docker volume create portainer_data
```

Then, download and install the Portainer Server container:

```
1 docker run -d -p 8000:8000 -p 9443:9443 --name portainer \
2 --restart=always \
3 -v /var/run/docker.sock:/var/run/docker.sock \
4 -v portainer_data:/data \
5 portainer/portainer-ce:latest
```

```
docker run -d -p 9000:9000 --name portainer --restart=always -v /var/run/docker.sock:/var/run/docker.sock -v
portainer_data:/data portainer/portainer-ce:latest
```

```
root@vultr:~# docker volume create portainer_data
portainer_data
root@vultr:~# docker run -d -p 9000:9000 --name portainer --restart=always -v /var/run/docker.sock:/var/run/docker.sock
-v portainer_data:/data portainer/portainer-ce:latest
Unable to find image 'portainer/portainer-ce:latest' locally
latest: Pulling from portainer/portainer-ce
7721cab3d696: Pull complete
0645e7e2a110: Pull complete
6329543ecfca: Pull complete
Digest: sha256:76ff22486bcd3713631b5f317efcb69e707a122fe96ffcc0589cf2d3e8e6b890
Status: Downloaded newer image for portainer/portainer-ce:latest
606aeaaf2eaa85f77381e60a3a94f6921e22019e5af7549863f884bbf4eef86f
```

## - 서버에 설치된 portainer 접근

주요 요약 | 158.247.206.129:9000/#/init/admin

id the game an... | GitHub - jaewonle... | 파스프링리스 온라인... | Colab Notebooks ~... | ddub6796/hjm: d... | 2021 Django Project | 직장학교 광고! Dja... | SAMSUNG CIC

portainer.io

▼ New Portainer installation

Please create the initial administrator user.

Username

Password

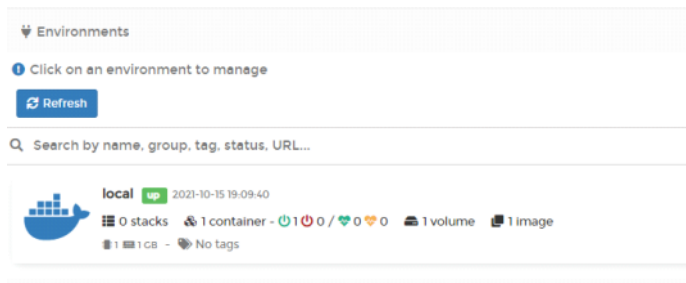
Confirm password

✖ The password must be at least 8 characters long

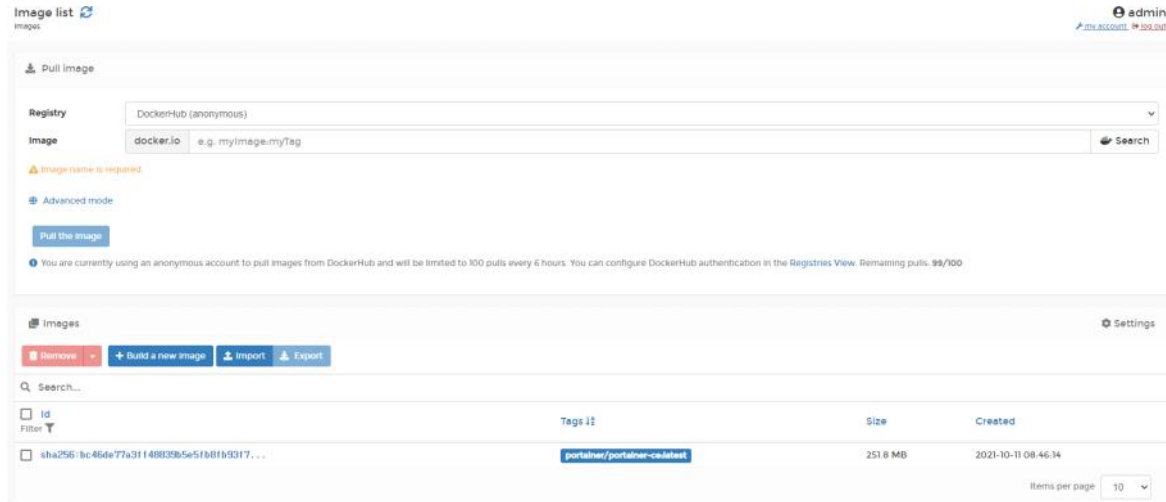
☒ Allow collection of anonymous statistics. You can find more information about this in our privacy policy.

➤ Restore Portainer from backup

## - 컨테이너 구동 확인



## - 서버 내 설치된 이미지 리스트 확인

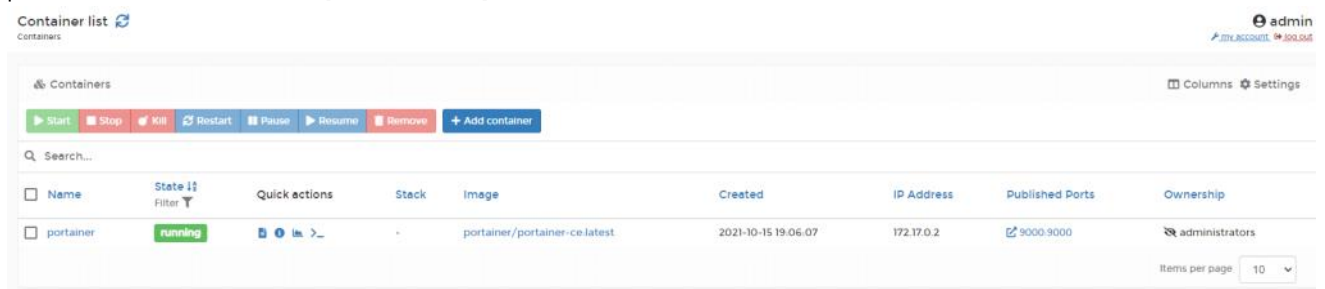


## • Port의 이해 그리고 Nginx 컨테이너 생성

- Port: 소프트웨어에서는 네트워크 서비스나 특정 프로세스를 식별하는 논리 단위. 각 포트는 번호로 구별되며 이 번호를 포트 번호라고 한다. 포트 번호는 IP 주소와 함께 쓰여 해당하는 프로토콜에 의해 사용
- 본 프로젝트에서 vultr 서버(158.247.206.129)의 9000번 포트와 portainer를 9000번 포트와 연결해주었음.

## - Nginx 컨테이너 추가하기

portainer -> container list -> [Add container] 버튼 클릭하여 컨테이너 추가



## - 아래와 같이 설정 후 [Deploy the container]

Create container

Containers > Add container

Name: nginx

Image configuration

Registry: DockerHub (anonymous)

Image: docker.io/nginx

Advanced mode

Always pull the image: ☒

You are currently using an anonymous account to pull images from DockerHub and will be limited to 100 pulls every 6 hours. You can configure DockerHub authentication in the [Registries View](#). Remaining pulls: 99/100

Network ports configuration

Publish all exposed network ports to random host ports: ☐

Manual network port publishing: [publish a new network port](#)

host: 80 → container: 80 TCP UDP

Access control

Enable access control: ☒

Administrators: I want to restrict the management of this resource to administrators only

Restricted: I want to restrict the management of this resource to a set of users and/or teams

Actions

Auto remove: ☐

Deploy the container

- container list에 80번 포트에 nginx이미지로 컨테이너가 생성됨

Container list

Containers

Start Stop K8 Restart Pause Resume Remove Add container

Search...

Name	State	Quick actions	Stack	Image	Created	IP Address	Published Ports	Ownership
nginx	running	<a href="#">Stop</a> <a href="#">Restart</a> <a href="#">Pause</a> <a href="#">Resume</a> <a href="#">Remove</a>	-	nginx:latest	2021-10-15 19:26:22	172.17.0.3	80:80	administrators
portainer	running	<a href="#">Stop</a> <a href="#">Restart</a> <a href="#">Pause</a> <a href="#">Resume</a> <a href="#">Remove</a>	-	portainer/portainer-ce:latest	2021-10-15 19:06:07	172.17.0.2	9000:9000	administrators

Items per page: 10

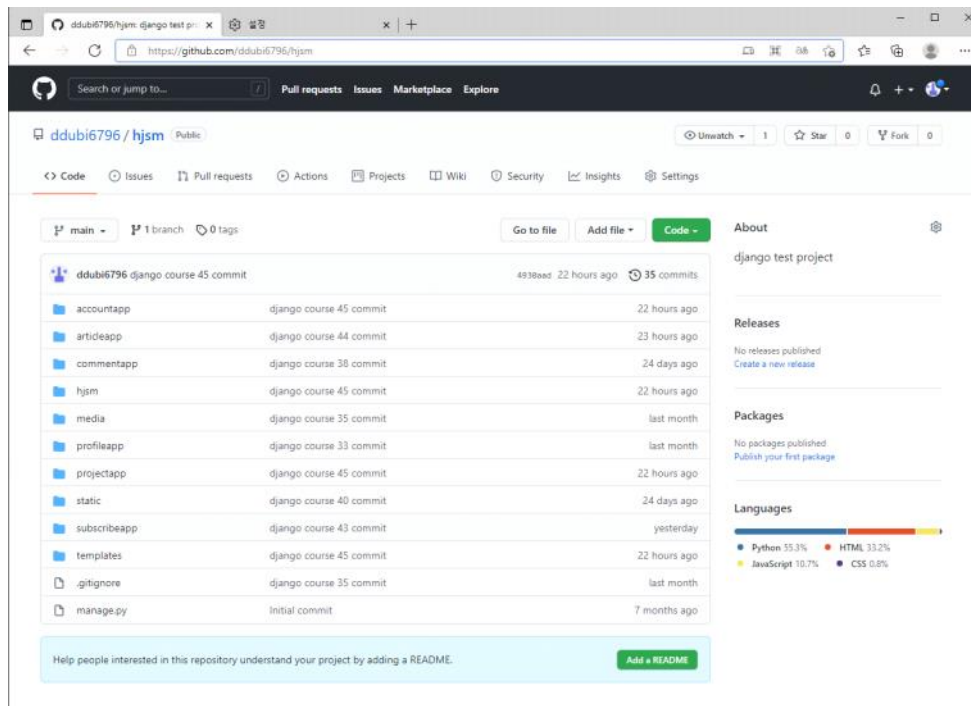
## Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to [nginx.org](https://nginx.org).  
Commercial support is available at [nginx.com](https://nginx.com).

Thank you for using nginx.

- django 소스코드 Github 업로드
  - Django container를 Vultr서버에 업로드 하기 위한 과정
    - a. Github에 django 소스 업로드
    - b. Dockerfile 작성
    - c. Image 빌드
    - d. Container 실행
  - github에 소스 업로드



- 프로젝트에 필요한 라이브러리 정보 가져오기(동일한 가상 환경을 구축하기 위함)

```
(venv) C:\Users\DDubi\PycharmProjects\hjasm>pip freeze > requirements.txt
```

manage.py  
requirements.txt

```
1 asgiref==3.3.1
2 beautifulsoup4==4.9.3
3 Django==3.1.7
4 django-bootstrap4==3.0.1
5 django-enviro==0.4.5
6 importlib-metadata==2.1.1
7 Pillow==8.3.1
8 pytz==2021.1
9 soupsieve==2.2.1
10 sqlparse==0.4.1
11 zipp==3.5.0
```

#### ○ Dockerfile 작성

Dockerfile: 컨테이너에 설치해야하는 패키지, 소스코드, 명령어, 환경변수설정 등을 기록한 하나의 파일

- FROM : 생성할 이미지의 베이스가 될 이미지를 뜻합니다. 반드시 한번 이상 입력해야 합니다.
- LABEL : 이미지에 메타데이터를 추가합니다. (나중에 원하는 조건의 컨테이너, 이미지 등을 쉽게 찾을 수 있도록 도와주기 때문에 기억해두는게 좋습니다)
- RUN : 이미지를 만들기 위해 컨테이너 내부에서 command명령어를 실행합니다. (여기서 주의할 점은 설치과정에서 별도의 입력이 불가능하기 때문에 apache2를 설치할 때 뒤에 -y를 붙여줘야 합니다.
- ADD : 파일을 이미지에 추가합니다. 여기서는 Dockerfile이 위치한 폴더에 test.html 파일을 가져와서 이미지의 /var/www/html 디렉토리에 추가합니다.
- WORKDIR : 명령어를 실행할 디렉토리. 배시 셸에서의 cd 명령어와 동일한 기능을 합니다.
- EXPOSE: 이미지에서 노출할 포트를 설정합니다.
- CMD : 컨테이너가 시작될 때마다 실행할 명령어. Dockerfile에서 한번만 사용할 수 있습니다.



```
FROM python:3.9.0

WORKDIR /home/

RUN git clone https://github.com/ddubi6796/hjasm.git

WORKDIR /home/hjasm/

RUN pip install -r requirements.txt

RUN echo "SECRET_KEY=!3+u3gg^^)gh*^(!*pdz1w@15pw*9tom)icq(e=k$eth$w*io7" > .env

RUN python manage.py migrate

EXPOSE 8000

CMD ["python", "manage.py", "runserver", "0.0.0.0:8000"]
```

- env파일의 SECRET\_KEY가 없으면 django가 정상 동작할 수 없으므로 임시로 프로젝트 내 env파일에서 SECRET\_KEY를 복사하여 준다. (추후 변경 필수!)

#### ○ TEST IMAGE 빌드

Build image

images > Build image

admin

my account

log out

Builder

Output

Naming

You can specify multiple names to your image.

Names

add additional name

A name must be specified in one of the following formats: `name:tag`, `repository/name:tag` or `registryfish:port/repository/name:tag` format. If you omit the tag the default `latest` value is assumed.

name

django\_test\_image:1

✓

✗

Build method

Web editor

Use our Web editor

Upload

Upload a tarball or a Dockerfile from your computer

URL

Specify a URL to a file

Uploaded

You can upload a Dockerfile or a tar archive containing a Dockerfile from your computer. When using a tarball, the root folder will be used as the build context.

Select file

Dockerfile

Actions

Build this image

Images

Settings

Remove

+ Build a new image

Import

Export

Search...

id

Filter

Tags

Size

Created

sha256:6514e8b0e167229746611bf1ebf1be81...

Unused

django\_test\_image:1

953.7 MB

2021-10-15 20:49:48

sha256:87a942281133e2da99cb16d653cd13...

nginx:latest

133.3 MB

2021-10-12 11:03:40

sha256:bc46de77a3f148839b5e5fb8fb9317...

portainer/portainer-ce:latest

251.8 MB

2021-10-11 08:46:34

sha256:0a1fb4652f1c063a01bb4e9e480b20b...

Unused

python:3.9.0

886 MB

2020-12-04 07:38:38

Items per page

10

#### ○ TEST Container 추가

Create container

Containers > Add container

admin

my account log out

Name

Image configuration

Registry

Image

Advanced mode

Always pull the image ☒

You are currently using an anonymous account to pull images from DockerHub and will be limited to 100 pulls every 6 hours. You can configure DockerHub authentication in the [Registries View](#). Remaining pulls: 97/100

Network ports configuration

Publish all exposed network ports to random host ports ☐

Manual network port publishing ☒ publish a new network port

host  → container

Access control

Enable access control ☒

☒ Administrators I want to restrict the management of this resource to administrators only

☐ Restricted I want to restrict the management of this resource to a set of users and/or teams

Container list

Containers

admin

my account log out

Columns Settings

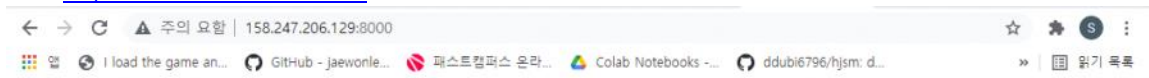
Start Stop Kill Restart Pause Resume Remove Add container

Search...

<input type="checkbox"/>	Name	State <input checked="" type="checkbox"/> Filter	Quick actions	Stack	Image	Created	IP Address	Published Ports	Ownership
<input type="checkbox"/>	django_container	running	<input type="button" value="Stop"/> <input type="button" value="Kill"/> <input type="button" value="Restart"/> <input type="button" value="Pause"/> <input type="button" value="Resume"/> <input type="button" value="Remove"/>	-	django_test_image:1	2021-10-15 20:57:58	172.17.0.4	8000-8000	administrators
<input type="checkbox"/>	nginx	running	<input type="button" value="Stop"/> <input type="button" value="Kill"/> <input type="button" value="Restart"/> <input type="button" value="Pause"/> <input type="button" value="Resume"/> <input type="button" value="Remove"/>	-	nginx:latest	2021-10-15 19:26:22	172.17.0.3	80-80	administrators
<input type="checkbox"/>	portainer	running	<input type="button" value="Stop"/> <input type="button" value="Kill"/> <input type="button" value="Restart"/> <input type="button" value="Pause"/> <input type="button" value="Resume"/> <input type="button" value="Remove"/>	-	portainer/portainer-ce:latest	2021-10-15 19:06:07	172.17.0.2	9000-9000	administrators

Items per page 10

▪ <http://158.247.206.129:8000/> 접속 확인



공지사항 | 제류문의 | 서비스 소개

HJSM

- Unicorn 을 사용해 지금까지 사용해왔던 runserver 명령어 대체 (python manage.py runserver는 배포 환경에서 사용 불가)

## runserver

django-admin **runserver** [addipport]

Starts a lightweight development Web server on the local machine. By default, the server runs on port 8000 on the IP address **127.0.0.1**. You can pass in an IP address and port number explicitly.

If you run this script as a user with normal privileges (recommended), you might not have access to start a port on a low port number. Low port numbers are reserved for the superuser (root).

This server uses the WSGI application object specified by the **WSGI\_APPLICATION** setting.

DO NOT USE THIS SERVER IN A PRODUCTION SETTING. It has not gone through security audits or performance tests. (And that's how it's gonna stay. We're in the business of making Web frameworks, not Web servers, so improving this server to be able to handle a production environment is outside the scope of Django.)

The development server automatically reloads Python code for each request, as needed. You don't need to restart the server for code

## runserver

django-admin **runserver** [addrport]

Starts a lightweight development Web server on the local machine. By default, the server runs on port 8000 on the IP address **127.0.0.1**. You can pass in an IP address and port number explicitly.

If you run this script as a user with normal privileges (recommended), you might not have access to start a port on a low port number. Low port numbers are reserved for the superuser (root).

This server uses the WSGI application object specified by the `WSGI_APPLICATION` setting.

**DO NOT USE THIS SERVER IN A PRODUCTION SETTING.** It has not gone through security audits or performance tests. (And that's how it's gonna stay. We're in the business of making Web frameworks, not Web servers, so improving this server to be able to handle a production environment is outside the scope of Django.)

The development server automatically reloads Python code for each request, as needed. You don't need to restart the server for code changes to take effect. However, some actions like adding files don't trigger a restart, so you'll have to restart the server in these cases.

-> django 컨테이너 내부에 gunicorn 라이브러리 설치하여 gunicorn command로 대체

gunicorn? Gunicorn 'Green Unicorn' is a Python WSGI HTTP Server for UNIX

(Nginx 웹서버와 django 컨테이너를 연결해주는 interface)

WSGI? python으로 작성된 웹 어플리케이션과 python으로 작성된 서버 사이의 약속된 인터페이스 또는 규칙

### - Gunicorn 설치

```
(venv) C:\Users\DDubi\PycharmProjects\hjsm>pip install gunicorn
Collecting gunicorn
  Downloading gunicorn-20.1.0-py3-none-any.whl (79 kB)
    |#####| 79 kB 5.5 MB/s
Requirement already satisfied: setuptools>=3.0 in c:\users\ddubi\pycharmprojects\hjsm\venv\lib\site-packages (from gunicorn) (54.2.0)
Installing collected packages: gunicorn
Successfully installed gunicorn-20.1.0
WARNING: You are using pip version 21.0.1; however, version 21.3 is available.
You should consider upgrading via the 'c:\users\ddubi\pycharmprojects\hjsm\venv\scripts\python.exe -m pip install --upgrade pip' command.
```

requirements.txt 에 반영 후 git push

```
(venv) C:\Users\DDubi\PycharmProjects\hjsm>pip freeze > requirements.txt
```

### - Gunicorn command로 변경

```
FROM python:3.9.0

WORKDIR /home/

RUN git clone https://github.com/ddubi6796/hjsm.git

WORKDIR /home/hjsm/

RUN pip install -r requirements.txt

RUN pip install gunicorn

RUN echo "SECRET_KEY=!3+u3gg^*)gh*^(!+pdzlw@15pw*9tom)lcq(e=k$eth$w*1o7" > .env

RUN python manage.py migrate

EXPOSE 8000

CMD ["gunicorn", "hjsm.wsgi", "--bind", "0.0.0.0:8000"]
```

### - image 빌드

Build image

images > Build image

Builder Output

Naming

You can specify multiple names to your image.

Names [add additional name](#)

A name must be specified in one of the following formats: `name:tag`, `repository/name:tag` or `registry/fqn:port/repository/name:tag` format. If you omit the tag the default `latest` value is assumed.

name

Build method

☒ Web editor  
Use our Web editor

☐ Upload  
Upload a tarball or a Dockerfile from your computer

☐ URL  
Specify a URL to a file

Upload

You can upload a Dockerfile or a tar archive containing a Dockerfile from your computer. When using a tarball, the root folder will be used as the build context.

Select file

Actions

## - container 추가

The screenshot shows the 'Create container' interface in Docker Desktop. The 'Name' field is set to 'django\_container\_gunicorn'. The 'Image' is 'django\_test\_image:2'. The 'Always pull the image' toggle is turned on. The 'Network ports configuration' section shows 'Publish all exposed network ports to random host ports' is turned on. The 'Access control' section has 'Enable access control' turned on. The 'Actions' section shows 'Deploy to container' button. Below this is the 'Containers' list showing four containers: 'django\_container\_gunicorn', 'django\_container', 'nginx', and 'portainer', all in 'running' state.

Name	State	Quick actions	Stack	Image	Created	IP Address	Published Ports
django_container_gunicorn	running	[Stop] [Restart] [Pause] [Kill]	-	django_test_image:2	2021-10-15 21:27:59	172.17.0.5	8080:8000
django_container	running	[Stop] [Restart] [Pause] [Kill]	-	django_test_image:1	2021-10-15 20:57:58	172.17.0.4	8000:8000
nginx	running	[Stop] [Restart] [Pause] [Kill]	-	nginx:latest	2021-10-15 19:26:22	172.17.0.3	80:80
portainer	running	[Stop] [Restart] [Pause] [Kill]	-	portainer/portainer-ce:latest	2021-10-15 19:06:07	172.17.0.2	9000:9000

- 이슈: static 파일 not supported 에러 발생 -> django 컨테이너와 nginx컨테이너 연결을 위한 설정이 필요함.

The screenshot shows a web browser displaying a Django project page titled '2021 Django Project'. The page has a 'No Articles YET!' message and a 'Create Article' button. The browser's DevTools console is open, showing several errors related to static file handling. The errors indicate that the browser refused to apply styles from a specific URL because its MIME type ('text/html') is not a supported stylesheet MIME type, and strict MIME checking is enabled. The errors also mention a failed load resource with a status of 404 (Not Found).

2021 Django Project  
Articles Projects Subscription login SignUp

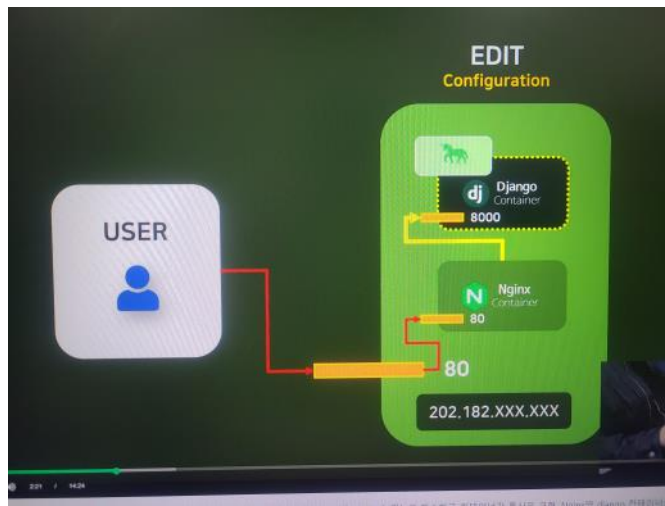
No Articles YET!

Create Article

공지사항 | 제휴문의 | 서비스 소개  
HJSM

DevTools console errors:

- Refused to apply style from 'http://158.247.206.129:8080/static/be 158.247.206.129/1 se.css' because its MIME type ('text/html') is not a supported stylesheet MIME type, and strict MIME checking is enabled.
- Failed to load resource: the server responded with a status of 404 (Not Found)
- Refused to apply style from 'http://158.247.206.129:8080/static/be 158.247.206.129/1 se.css' because its MIME type ('text/html') is not a supported stylesheet MIME type, and strict MIME checking is enabled.

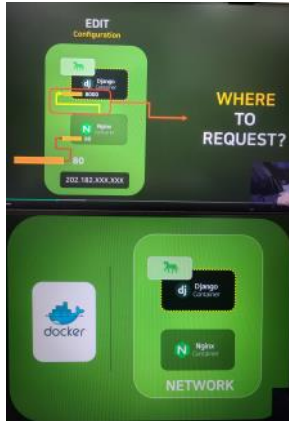


# Docker Network, Volume

2021년 10월 15일 금요일 오후 9:32

- Docker Network의 이해 및 구현

- 네트워크: 여러개의 컨테이너를 하나의 묶어주는 것



- 네트워크 안에서는 컨테이너 name을 도메인처럼 사용할 수 있음



- portainer를 제외한 컨테이너 delete 후, nginx-django네트워크 추가하기

- django\_container\_gunicorn 컨테이너 추가(nginx-django네트워크 설정)

Create container

Name:

Image configuration

Registry: DockerHub (anonymous)

Image:

Advanced mode

Always pull the image: ☒

Network ports configuration

Manual network port publishing: ☐

Access control

Enable access control: ☒

Administrators: ☒ I want to restrict the management of this resource to administrators only

Restricted: ☐ I want to restrict the management of this resource to a set of users and/or teams

Actions

Auto remove: ☐

Deploy the container

Advanced container settings

Command & logging

Volumes

Network:

Env

Labels

Restart policy

Runtime & Resources

Capabilities

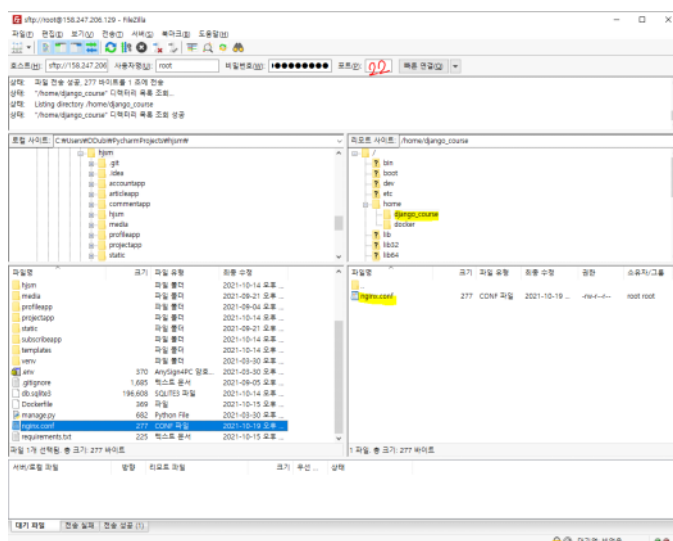
- 
- nginx 설정파일 생성(nginx.conf) 후 filezilla로 가상서버에 설정파일 업로드
- <https://gunicorn.org/#deployment>

```
worker_processes auto;

events {
}

http {
    server {
        listen 80;

        location / {
            proxy_pass http://django_container_gunicorn:8000;
            proxy_set_header Host $host;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        }
    }
}
```



- nginx 컨테이너 생성
- network와 volume설정: django컨테이너와 동일한 네트워크로 설정, 가상서버와 컨테이너 내부를 연결



Containers Columns Settings

Start
Stop
Kill
Restart
Pause
Resume
Remove
+ Add container

<input type="checkbox"/>	Name	State <small>Filter</small>	Quick actions	Stack	Image	Created	IP Address	Published Ports	Ownership
<input type="checkbox"/>	nginx	running		-	nginx:latest	2021-10-19 21:26:57	172.18.0.3	<a href="#">80:80</a>	administrators
<input type="checkbox"/>	django_container_gunicorn	running		-	django_test_image-2	2021-10-19 21:06:29	172.18.0.2	-	administrators
<input type="checkbox"/>	portainer	running		-	portainer/portainer-ce:latest	2021-10-15 19:06:07	172.17.0.2	<a href="#">9000:9000</a>	administrators

Items per page 10

- [illegible]

- Docker 페이로 16



- 해결방안: 1) Collect static content from django container  
2) Synchronize static contents with nginx container

- Docker 컨테이너 생성 시, Collectstatic 명령을 통한 Static 파일 취합

- python manage.py collectstatic
- C:\Users\DDubi\PycharmProjects\hjsm\staticfiles

```
(venv) C:\Users\DDubi\PycharmProjects\hjsm>python manage.py collectstatic

138 static files copied to 'C:\Users\DDubi\PycharmProjects\hjsm\staticfiles'.
```

- 초기 settings.py 내부에 설정해주었던 STATIC\_ROOT 경로에 파일이 취합됨

```
STATIC_URL = '/static/'

STATIC_ROOT = os.path.join(BASE_DIR, 'staticfiles')

STATICFILES_DIRS = [
    BASE_DIR / "static"
]
```

- Dockerfile 내부에 collectstatic 명령어 추가(도커 컨테이너 생성 시 static파일 취합)

```
RUN echo "SECRET_KEY=!3+u3gg^^)gh*^(!*pdz

RUN python manage.py migrate

RUN python manage.py collectstatic

EXPOSE 8000
```

- 이미지 생성

name  ✓

Build method

☒ Web editor  
Use our Web editor

☒ Upload  
Upload a tarball or a Dockerfile from your computer

☐ URL  
Specify a URL to a file

Upload

You can upload a Dockerfile or a tar archive containing a Dockerfile from your computer. When using a tarball, the root folder will be used as the build context.

Step 9/11 : RUN python manage.py collectstatic

---> Running in 4210e4dd34f

138 static files copied to '/home/hjsm/staticfiles'.

Removing intermediate container 4210e4dd34f

---> 81a1flab2052

- 생성한 docker 이미지 내 django의 static content를 nginx로 동기화 필요 -> Docker Volume

- Docker Volume

- Bind Volume: Host server(Vultr)의 Nginx.conf와 Nginx Container의 Nginx.conf 연동 시 사용
- Named Volume: docker 내부에 새로운 Volume을 생성하여 컨테이너와 연동 후 연동한 컨테이너가 삭제되어도 데이터 유지



- Docker Volume 생성 및 Container 적용

- static, media Volume 생성

Create volume  
Volumes > Add volume

Name

Driver configuration

Driver

Driver options [?](#) [+ add driver option](#)

Use NFS volume ☐

Use CIFS volume ☐

Access control

Enable access control [?](#) ☒

☒ Administrators  
I want to restrict the management of this resource to administrators only

☐ Restricted  
I want to restrict the management of this resource to

Actions

[Create the volume](#)

Create volume  
Volumes > Add volume

Name

Driver configuration

Driver

Driver options [?](#) [+ add driver option](#)

Use NFS volume ☐

Use CIFS volume ☐

Access control

Enable access control [?](#) ☒

☒ Administrators  
I want to restrict the management of this resource to administrators only

☐ Restricted  
I want to restrict the management of this resource to

Actions

[Create the volume](#)

- django\_container\_gunicorn 컨테이너 재생성

network: nginx-django

volumes: static, media 볼륨 연결.

container의 경로는 dockerfile의 `WORKDIR/home/hjism/` + settings.py 내부에서 정의한 static, media root 경로

Create container  
Containers > Add container

admin  
my account log out

Name:

Image configuration

Registry: DockerHub (anonymous)

Image:  [Search](#)

Advanced mode

Always pull the image: ☒

You are currently using an anonymous account to pull images from DockerHub and will be limited to 100 pulls every 6 hours. You can configure DockerHub authentication in the Registries View. Remaining pulls: 100/100

Network ports configuration

Publish all exposed network ports to random host ports: ☐

Manual network port publishing: [publish a new network port](#)

Access control

Enable access control: ☐

**Administrators**  
I want to restrict the management of this resource to administrators only

**Restricted**  
I want to restrict the management of this resource to a set of users and/or teams

Actions

Auto remove: ☐

[Deploy the container](#)

Advanced container settings

Command & logging Volumes **Network** Env Labels Restart policy Runtime & Resources Capabilities

Network:

Advanced container settings

Command & logging Volumes Network Env Labels Restart policy Runtime & Resources Capabilities

Volume mapping [map additional volume](#)

container	/home/hjgm/staticfiles/	Volume	Bind	<input type="checkbox"/>
→ volume	static-local	Writable	Read-only	<input type="checkbox"/>
container	/home/hjgm/media/	Volume	Bind	<input type="checkbox"/>
→ volume	media-local	Writable	Read-only	<input type="checkbox"/>

- nginx.conf 수정 후 filezilla 를 통해 서버에 업로드  
static, dynamic content 모두 전송  
alias에는 nginx 컨테이너 내 생성할 volume 경로
- (참고) MIME(Multipurpose Internet Mail Extensions) 이란?  
MIME은 다목적 인터넷 메일 확장이란 뜻으로 전자우편을 위하여 데이터 형식을 정의한 인터넷 표준 포맷

```
http {
    server {
        listen 80;

        include mime.types;

        location /static/ {
            alias /data/static/;
        }

        location /media/ {
            alias /data/media/;
        }

        location / {
            proxy_pass http://django_container_gunicorn:8000;
            proxy_set_header Host $host;
        }
    }
}
```

- nginx 컨테이너 재생성  
network: nginx-django

volumes: static, media 볼륨 연결 + nginx.conf bind 연결

nginx 컨테이너 내 static, media 볼륨 경로는 임의로 설정(mkdir)

Create container  
Container > Add container

Name:

Image configuration

Registry: DockerHub (anonymous)

Image: docker.io/nginx:latest

Advanced mode

Always pull the image: ☒

Network ports configuration

Publish all exposed network ports to random host ports: ☐

Manual network port publishing: ☒ publish a new network port

host: 80 → container: 80

Access control

Enable access control: ☒

Administrators: I want to restrict the management of this resource to administrators only

Restricted: I want to restrict the management of this resource to a set of users and/or teams

Actions

Auto remove: ☐

Deploy the container

Advanced container settings

Command & logging | Volumes | **Network** | Env | Labels | Restart policy | Runtime & Resources | Capabilities

Network: nginx-django

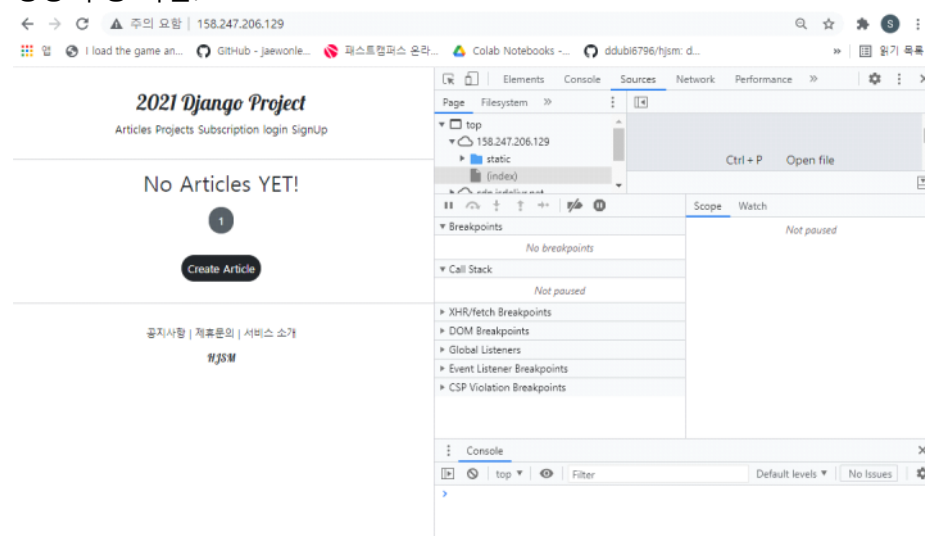
Volume mapping

container: /data/static/ → volume: static-local

container: /data/media/ → volume: media-local

container: /etc/nginx/nginx.conf → host: /home/django-course/nginx.conf

- 정상 구동 확인!

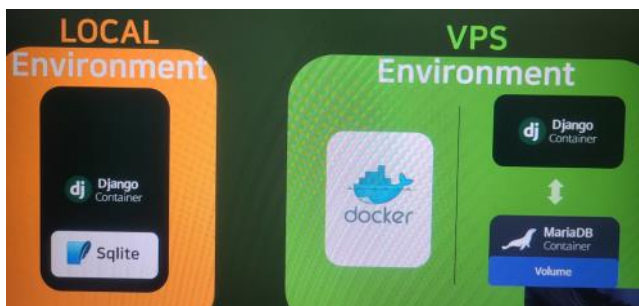


# Local, Remote environment detachment

2021년 10월 19일 화요일    오후 10:27

- MariaDB 컨테이너를 이용한 DB 분리

- django컨테이너 내부에 DB가 있는 구조 -> 컨테이너 장애 시 전체 시스템에 이슈가 됨
- DB를 별도의 컨테이너로 생성 후, volume을 활용하여 데이터가 지속적으로 유지되도록 함



- MariaDB 컨테이너 생성(구동 테스트)

[https://hub.docker.com/\\_/mariadb](https://hub.docker.com/_/mariadb)

Starting a MariaDB instance is simple:

```
$ docker run -p 127.0.0.1:3306:3306 --name some-mariadb -e MARIADB_ROOT_PASSWORD=my-secret-pw -d mariadb:tag
```

환경변수(Env) MARIADB\_ROOT\_PASSWORD 설정

Name: mariadb

Image configuration

Registry: DockerHub (anonymous)

Image: docker.io/mariadb:10.5

Advanced mode

Always pull the image: ☒

You are currently using an anonymous account to pull images from DockerHub and will be limited to 100 pulls every 6 hours. You can configure DockerHub authentication in the Registries View. Remaining pulls: 100/100

Network ports configuration

Publish all exposed network ports to random host ports: ☐

Manual network port publishing: [publish a new network port](#)

Access control

Enable access control: ☒

**Administrators**  
I want to restrict the management of this resource to administrators only

**Restricted**  
I want to restrict the management of this resource to a set of users and/or teams

Actions

Auto remove: ☐

[Deploy the container](#)

Advanced container settings

Command & logging | Volumes | Network | **Env** | Labels | Restart policy | Runtime & Resources | Capabilities

Environment variables

These values will be applied to the container when deployed

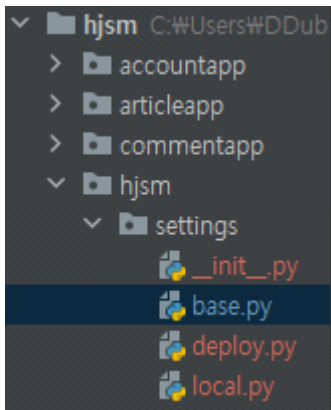
Advanced mode: ☒ Switch to advanced mode to copy & paste multiple variables

[Add an environment variable](#) [Load variables from .env file](#)

name	value	
MARIADB_ROOT_PASSWORD	password1234	<a href="#">Remove value</a>

## • 개발/배포 설정 분리

- settings 파이썬 패키지 추가 후, 하위로 settings.py 이동 후 base.py로 이름 변경
- 로컬용 환경변수 설정을 위한 local.py 파일 생성
- 서버용 환경변수 설정을 위한 deploy.py 파일 생성



- base.py로 부터 일부 설정을 local과 deploy로 분리
- local.py

```

from .base import *

env = environ.Env(
    # set casting, default value
    DEBUG=(bool, False)
)

# reading .env file
environ.Env.read_env(
    env_file=os.path.join(BASE_DIR, '.env')
)

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/3.1/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = env('SECRET_KEY')

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = ['*']

# Database
# https://docs.djangoproject.com/en/3.1/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}

```

#### - deploy.py

```

from .base import *

env = environ.Env(
    # set casting, default value
    DEBUG=(bool, False)
)

# reading .env file
environ.Env.read_env(
    env_file=os.path.join(BASE_DIR, '.env')
)

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/3.1/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = env('SECRET_KEY')

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = False

ALLOWED_HOSTS = ['*']

# Database
# https://docs.djangoproject.com/en/3.1/ref/settings/#databases
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'django',
        'USER': 'django',
        'PASSWORD': 'password1234',
        'HOST': 'mariadb', #mariadb의 컨테이너명
        'PORT': '3306', #mariadb가 사용하는 port
    }
}

```

#### - base.py 내 BASE\_DIR 변경: hjsm/settings 하위로 이동하였으므로

```

# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent

```

#### - manage.py 내 settings 경로 변경(local 구동 시 hjsm/settings/local 설정파일 사용)

```

def main():
    """Run administrative tasks."""
    os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'hjsm.settings.local')
    try:

```

- MariaDB 컨테이너 설정 및 Django 연동

- database 볼륨 생성

Create volume  
Volumes > Add volume

- mariadb 컨테이너 추가

- [https://hub.docker.com/\\_/mariadb](https://hub.docker.com/_/mariadb)

Creating database dumps

Most of the normal tools will work, although their usage might be a little convoluted in some cases to ensure they have access to the `mysqld` server. A simple way to ensure this is to use `docker exec` and run the tool from the same container, similar to the following:

```
$ docker exec some-mariadb sh -c 'exec mysqldump --all-databases -uroot -p"$MARIADB_ROOT_PASSWORD" > /some/path/on/your/host/all-databases'
```

- Env: settings/deploy.py에서 설정한 값과 동일하게 설정



Command & logging Volumes Network **Env** Labels Restart policy Runtime & Resources Capabilities

Environment variables

These values will be applied to the container when deployed

Advanced mode  
Switch to advanced mode to copy & paste multiple variables

Add an environment variable Load variables from .env file

name	value	Remove value
MYSQL_ROOT_PASSWORD	password1234	Remove value
MYSQL_DATABASE	django	Remove value
MYSQL_USER	django	Remove value
MYSQL_PASSWORD	password1234	Remove value

- Git Push
- Dockerfile 수정
- RUN python manage.py migrate 삭제
- <https://docs.gunicorn.org/en/latest/run.html#django> Mariadb

```

root@python:3.9.0
WORKDIR /home/
RUN echo "testing"
RUN git clone https://github.com/doulo796/hjss.git
WORKDIR /home/hjss/
RUN pip install -r requirements.txt
RUN pip install gunicorn
RUN pip install mysqlclient
RUN echo "SECRET_KEY=$(cat /dev/urandom | tr -dc 'a-z0-9' | fold -n 50 | xargs | sha256sum | cut -d ' ' -f 1)" > .env
RUN python manage.py collectstatic
EXPOSE 8000
CMD ["python", "-c", "python manage.py migrate --settings=hjss.settings.deploy && gunicorn hjss.wsgi --env DJANGO_SETTINGS_MODULE=hjss.settings.deploy --bind 0.0.0.0:8000"]

```

- django 이미지 생성

Names **add additional name**

A name must be specified in one of the following formats: `name:tag`, `repository/name:tag` Or `registryfqdn:port/repository/name:tag` format. If `latest` value is assumed.

name **django\_test\_image-4** ✓

Build method

Web editor Use our Web editor

Upload Upload a tarball or a Dockerfile from your computer

Spec

Upload

You can upload a Dockerfile or a tar archive containing a Dockerfile from your computer. When using a tarball, the root folder will be used as 1

Select file Dockerfile

Actions

Build the image

- django 컨테이너 생성

Name

Image configuration

Registry

Image

☒ Advanced mode

Always pull the image ☒

You are currently using an anonymous account to pull images from DockerHub and will be limited to 100 pulls every 6 hours. You can configure Docker authentication in the [Registries View](#). Remaining pulls: 99/100

Network ports configuration

Publish all exposed network ports to random host ports ☐

Manual network port publishing

Access control

Enable access control ☒

☒ Administrators  
 I want to restrict the management of this resource to administrators only

☐ Restricted  
 I want to restrict the management of this resource to a set of teams

Actions

Auto remove ☐

Advanced container settings

Command & logging Volumes **Network** Env Labels Restart policy Runtime & Resources

Network

Command & logging **Volumes** Network Env Labels Restart policy Runtime & Resources

Volume mapping

container	/home/njam/staticfiles/	Volume	Bind	<input type="button" value="X"/>
→ volume	static - local	Writable	Read-only	
container	/home/njam/media/	Volume	Bind	<input type="button" value="X"/>
→ volume	media - local	Writable	Read-only	

- 해당 컨테이너 삭제 시에도 mariadb의 데이터는 유지됨.
- (별도의 컨테이너에 named Volume을 사용했기 때문)

# Docker Swarm, Stack, Secret

2021년 10월 20일 수요일    오후 11:02

- Container의 한계, Docker Stack의 이해
  - container만으로 해결할 수 없는 한계
    - 1) Repetitive configuration: 반복적으로 환경설정이 필요
    - 2) Container shutdown: 컨테이너가 꺼졌을때 발생하는 이슈
  - TOTAL STACK Settings
    - > Docker Stack: 다수의 컨테이너를 한번에 배포 - 같은 환경설정 공유(Docker Compose와 비슷),
    - > Service: scale out containers - 문제 발생 시, 설정 파일을 불러와 Automatic Reboot
- Docker Swarm의 이해
  - 컨테이너 오케스트레이션(Container Orchestration) 툴
  - 오케스트레이션? 여러 대의 서버와 여러 개의 서비스를 편리하게 관리해주는 작업. 스케줄링scheduling, 클러스터링clustering, 서비스 디스커버리 service discovery, 로깅logging, 모니터링monitoring 등의 작업을 수행

- Stack을 위한 yml 파일 작성
  - 현재 node를 manager node로 설정(swarm 모드 켜기)
    - o ssh root@158.247.206.129
    - o Password: 4Gc\*9g5G+C2f4Y7m
    - o cd home/django\_course
    - o docker swarm init

```
root@vultr:~# cd ...
root@vultr:~# cd home/django_course
root@vultr:~/home/django_course# docker swarm init
Swarm initialized: current node (hpayjfx24712dcccghwrbmza) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-3ium6t8021o0xy7pbu795rqnndz7cbghq201guk8qo06xtzgg-ccxiizrpctc7dfgbryo9umpc4k 158.247.206.129:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.
```

- services, swarm 메뉴 활성화됨



- docker-compose.yml 파일 작성

```
version: '3.7'
services:
  django:
    image: django_test_image:3
    ports:
      - 8000:8000
```

- django\_stack 스택 생성
- stack과 service, container 생성 확인
- <http://158.247.206.129:8000/>

**Create stack**  
Stacks > Add stack

Name **django\_stack**

This stack will be deployed using the equivalent of the `docker stack deploy` command.

Build method

☒ Web editor  
Use our Web editor

☒ Upload  
Upload from your computer

Upload

You must upload a Compose file from your computer.

**Upload file** **docker-compose.yml**

Environment variables

These values will be used as substitutions in the stack file

☒ Advanced mode  
☒ Switch to advanced mode to copy & paste multiple variables

☒ Add an environment variable ☐ Load variables from env file

Access control

Enable access control ☒

**Administrators**  
I want to restrict the management of this resource to administrators only

Actions

**Deploy the stack**

**Stacks list** [Refresh](#)  
Stacks

**Remove** **+ Add stack**

Q Search...

Name	Type
django_stack	Swarm

**Service list** [Refresh](#)  
Services

**Update** **Remove** **+ Add service**

Q Search...

Name	Stack	Image
django_stack_django	django_stack	django_test_image:3

**Container list** [Refresh](#)  
Containers

**Start** **Stop** **Restart** **Inspect** **Logs** **Exec** **Attach** **Detach** **Remove** **+ Add container**

Q Search...

Name	Stack ID	Task actions	Stack	Image
django_stack_django_1atnmg...	swa122	<b>Restart</b> <b>Stop</b> <b>Start</b> <b>Inspect</b> <b>Logs</b> <b>Exec</b> <b>Attach</b> <b>Detach</b> <b>Remove</b>	django_stack	django_test_image:3
django_container_genserv...	swa122	<b>Restart</b> <b>Stop</b> <b>Start</b> <b>Inspect</b> <b>Logs</b> <b>Exec</b> <b>Attach</b> <b>Detach</b> <b>Remove</b>	-	django_test_image:4
nginx	swa122	<b>Restart</b> <b>Stop</b> <b>Start</b> <b>Inspect</b> <b>Logs</b> <b>Exec</b> <b>Attach</b> <b>Detach</b> <b>Remove</b>	-	nginx:alpine
postgres	swa122	<b>Restart</b> <b>Stop</b> <b>Start</b> <b>Inspect</b> <b>Logs</b> <b>Exec</b> <b>Attach</b> <b>Detach</b> <b>Remove</b>	-	postgres:postgres-alpine

vultr.guest  
manager  
CPU: 1  
Memory: 1.03 GB  
**ready**

django\_stack\_django  
Image: django\_test\_image:3  
Status: running  
Update: 2021-10-21 22:23:22

django\_stack\_django  
Image: django\_test\_image:3  
Status: running  
Update: 2021-10-21 22:25:17

← → 🔄 주의 사항 | 158.247.206.129:8000

📁 1 load the game an... 📁 GitHub - jasonle... 📁 퍼스노트북스 운영... 📁 Colab Notebooks ~...

## 2021 Django Project

[rticles Projects Subscriptions login SignUp](#)

No Articles YET!

1

Create Article

1 기사장 | 계통문의 | 서비스 소개  
JSM

-> static 파일 필요; nginx 연동 필요

- 통합 yml파일 작성
  - nginx 추가: bind volume 설정 및 named volume 설정

- django 서비스 추가 시 "django\_container\_gunicorn"으로 설정 - 설정해준 도메인명으로 생성 필요
- mariadb 서비스 추가 시 deploy settings에서 설정한 HOST 명으로 작성 필요

```
version: "3.7"
services:
  nginx:
    image: nginx:1.19.5
    networks:
      - network
    volumes:
      - home/django_course/nginx.conf:/etc/nginx/nginx.conf
      - static-volume:/data/static
      - media-volume:/data/media
    ports:
      - 80:80
  django_container_gunicorn:
    image: django_test_image:4
    networks:
      - network
    volumes:
      - static-volume:/home/hjasm/staticfiles
      - media-volume:/home/hjasm/media
  mariadb:
    image: mariadb:10.5
    networks:
      - network
    volumes:
      - maria-database:/var/lib/mysql
    environment:
      MYSQL_ROOT_PASSWORD: password1234
      MYSQL_DATABASE: django
      MYSQL_USER: django
      MYSQL_PASSWORD: password1234

networks:
  network:

volumes:
  static-volume:
  media-volume:
  maria-database:
```

- stack 추가 및 확인

Create stack

Name

This field must consist of lower case alphanumeric characters, '-', or '/' (e.g. 'my-stack' or 'abc/123').

This check will be deployed using the equivalent of the 'docker stack deploy' command.

Build method

☒ Web editor  
Use our Web editor

☐ Upload  
Upload from your computer

Upload

You must upload a Compose file from your computer.

Environment variables

These values will be used as substitutions in the stack file

☒ Advanced mode  
Switch to advanced mode to copy & paste multiple variables

Access control

☒ Enable access control

☒ Administrators  
I want to restrict the management of this resource to administrators only

Actions

---

Service list

Services

Search...

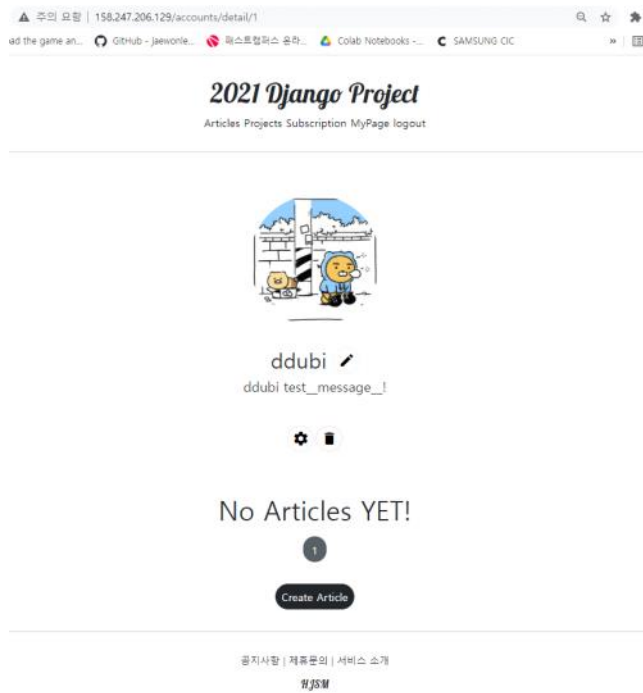
<input type="checkbox"/>	Name	Stack	Image	Scheduling Mode
<input type="checkbox"/>	django_stack_django_container_gunicorn	django_stack	django_test_image:4	replicated 1 / 1 Scale
<input type="checkbox"/>	django_stack_mariadb	django_stack	mariadb:10.5	replicated 1 / 1 Scale
<input type="checkbox"/>	django_stack_nginx	django_stack	nginx:1.19.5	replicated 1 / 1 Scale

---

Containers

Search...

<input type="checkbox"/>	Name	State	Quick actions	Stack	Image
<input type="checkbox"/>	django_stack_nginx.1.x6l5hp54...	running	<input type="button" value="Stop"/> <input type="button" value="Kill"/> <input type="button" value="Restart"/>	django_stack	nginx:1.19.5
<input type="checkbox"/>	django_stack_django_container...	running	<input type="button" value="Stop"/> <input type="button" value="Kill"/> <input type="button" value="Restart"/>	django_stack	django_test_image:4
<input type="checkbox"/>	django_stack_mariadb.1.icczkz...	running	<input type="button" value="Stop"/> <input type="button" value="Kill"/> <input type="button" value="Restart"/>	django_stack	mariadb:10.5



- Docker Secret을 이용한 보안

- 보안 관련된 정보들을 소스코드 혹은 파일에 저장하는 방식이 아닌, Docker 내에서 별도로 관리하기 위해 사용함
- dockerfile 내 보안 정보를 secret으로 생성 후 해당 항목 삭제

```

RUN pip install mysqlclient
RUN echo "SECRET_KEY=13+u3gg^^(!*&pdziw@15pw*9tom)icq(e=k$eth$w*io?" > .env
RUN python manage.py collectstatic
  
```

Create secret

Secrets > Add secret

Name: **13+u3gg^^(!\*&pdziw@15pw\*9tom)icq(e=k\$eth\$w\*io?**

Secret: **13+u3gg^^(!\*&pdziw@15pw\*9tom)icq(e=k\$eth\$w\*io?**

Encode secret: ☒

Labels: **add label**

Access control

Enable access control: ☒

**Administrators**

I want to restrict the management of this resource to administrators only

Actions

**Create the secret**

Create secret

Secrets > Add secret

Name: **MYSQL\_PASSWORD**

Secret: **password1234**

Encode secret: ☒

Labels: **add label**

Access control

Enable access control: ☒

**Administrators**

I want to restrict the management of this resource to administrators

Actions

**Create the secret**

Create secret

Secrets > Add secret

Name

Secret

Encode secret ☒

Labels

Access control

Enable access control ☒

☒ Administrators  
I want to restrict the management of this resource to administrators

Actions

- docker-compose.yml 파일 변경 (secret 제공을 위함)

- 참고)

## Docker Secrets

As an alternative to passing sensitive information via environment variables, `_FILE` may be appended to the previously listed environment variables, causing the initialization script to load the values for those variables from files present in the container. In particular, this can be used to load passwords from Docker secrets stored in `/run/secrets/<secret_name>` files. For example:

```
$ docker run --name some-mysql --env MYSQL_ROOT_PASSWORD_FILE=/run/secrets/mysql-root -d mariadb:tag
```

```
django_container_gunicorn:
  image: django_test_image:4
  networks:
    - network
  volumes:
    - static-volume:/home/hjsm/staticfiles
    - media-volume:/home/hjsm/media
  secrets:
    - MYSQL_PASSWORD
    - DJANGO_SECRET_KEY
mariadb:
  image: mariadb:10.5
  networks:
    - network
  volumes:
    - maria-database:/var/lib/mysql
  secrets:
    - MYSQL_PASSWORD
    - MYSQL_ROOT_PASSWORD
  environment:
    MYSQL_DATABASE: django
    MYSQL_USER: django
    MYSQL_PASSWORD_FILE: /run/secrets/MYSQL_PASSWORD
    MYSQL_ROOT_PASSWORD_FILE: /run/secrets/MYSQL_ROOT_PASSWORD

networks:
  network:

volumes:
  static-volume:
  media-volume:
  maria-database:

secrets:
  DJANGO_SECRET_KEY:
    external: true
  MYSQL_PASSWORD:
    external: true
  MYSQL_ROOT_PASSWORD:
    external: true
```

- deploy.py 수정 및 commit&push

```
from .base import *

def read_secret(secret_name):
    file = open('/run/secrets/' + secret_name)
    secret = file.read()
    secret = secret.rstrip().lstrip()
    file.close()
    return secret

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = os.environ.get('SECRET_KEY')
SECRET_KEY = read_secret('SECRET_KEY')

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = False

ALLOWED_HOSTS = ['*']

# Database
# https://docs.djangoproject.com/en/3.1/ref/settings/#databases
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'django',
        'USER': 'django',
        'PASSWORD': read_secret('MYSQL_PASSWORD'),
        'HOST': 'mariadb',
        'PORT': 3306,
    }
}
```

- dockerfile 수정
- collectstatic을 마지막으로 순서 변경
- (secret 사용 시 collectstatic수행 시 필요한 secret\_key를 불러올 수 없기 때문

```
FROM python:3.7.4
WORKDIR /home/
RUN echo "testing"
RUN git clone https://github.com/ddubi796/hjse.git
WORKDIR /home/hjse/
RUN pip install -r requirements.txt
RUN pip install gunicorn
RUN pip install gysgclient
EXPOSE 8000
CMD ["bash", "python manage.py collectstatic --noinput --settings=hjse.settings.deploy && python manage.py migrate --settings=hjse.settings.deploy &&
```

- django\_test\_image:5 이미지 생성

**Build image**  
Images > Build image

**Naming**

You can specify multiple names to your image.

**Names** [add additional name](#)

A name must be specified in one of the following formats: `name:tag`, `repository/name:tag` or `registry/fqn:port/repository/name:tag` format. If you omit the tag the default `latest` value is assumed.

name  ✓

**Build method**

☒ Web editor  
Use our Web editor

☒ Upload  
Upload a tarball or a Dockerfile from your computer

**Upload**

You can upload a Dockerfile or a tar archive containing a Dockerfile from your computer. When using a tarball, the root folder will be used as the build context.

[Select file](#) [Dockerfile](#)

**Actions**

[Build the image](#)

- django\_stack 재생성

**Name**

This stack will be deployed using the equivalent of the `docker stack deploy` command.

**Build method**

☒ Web editor  
Use our Web editor

☒ Upload  
Upload from your computer

**Upload**

You can upload a Compose file from your computer.

[Select file](#) [docker-compose.yml](#)

**Environment variables**

- 정상 구동 확인!

