목차

[Django VM 3tier 2](#_Toc101252345)

[RDS 생성 2](#_Toc101252346)

[Django 서버 3](#_Toc101252347)

[Nginx 서버 6](#_Toc101252348)

[Django container 3tier 8](#_Toc101252349)

[Mysql 이미지 생성 8](#_Toc101252350)

[Django 이미지 생성 9](#_Toc101252351)

[Nginx 서버 10](#_Toc101252352)

[Docker-compose 12](#_Toc101252353)

[Docker compose 설치 12](#_Toc101252354)

[Docker 설치 12](#_Toc101252355)

[DB 생성 13](#_Toc101252356)

[Nginx 생성 14](#_Toc101252357)

[Docker-compose.yaml 16](#_Toc101252358)

# Django VM 3tier

## RDS 생성

* Mysql 5.7.37 설치
* 일단 모든 접속을 허용한다

**rds 사용하기**

|  |
| --- |
| # maria DB client 설치  sudo apt-get install mariadb-client -y  # 접속 확인  mysql -h mysql.cersrmaeghas.us-west-1.rds.amazonaws.com -u root -p  # DB 생성  CREATE DATABASE django\_db;  CREATE USER 'python'@'%' IDENTIFIED BY 'python';  GRANT ALL PRIVILEGES ON django\_db.\* TO 'python'@'%';  flush privileges;  # DB dump  mysql -h mysql.cersrmaeghas.us-west-1.rds.amazonaws.com -u python -p django\_db < ./Dump.sql |

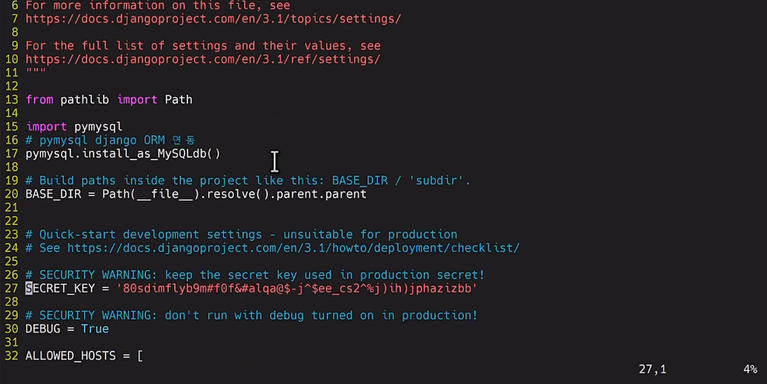
## Django 서버

EC2 인스턴스 만들기

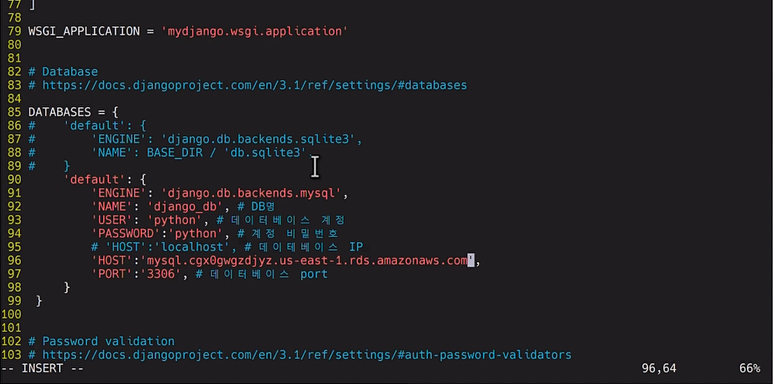
* Ubuntu 로 생성
* 일단 모든 접속을 허용한다

|  |
| --- |
| # update  sudo apt-get update  # 의존성 검사  sudo apt-get dist-upgrade -y  # 파이썬 확인 (3.8)  python3 --version  # pip 설치  sudo apt-get install python3-pip  #장고 설치  sudo pip install django==3.1.13  # git clone  git clone <https://github.com/gymin97/django-web.git>    # 접속 확인  python3 manage.py runserver 0:8000  # settings.py 수정  vi ./django-web/mydjango/setting.py |

1. 15번째, 17째 줄 주석 해제



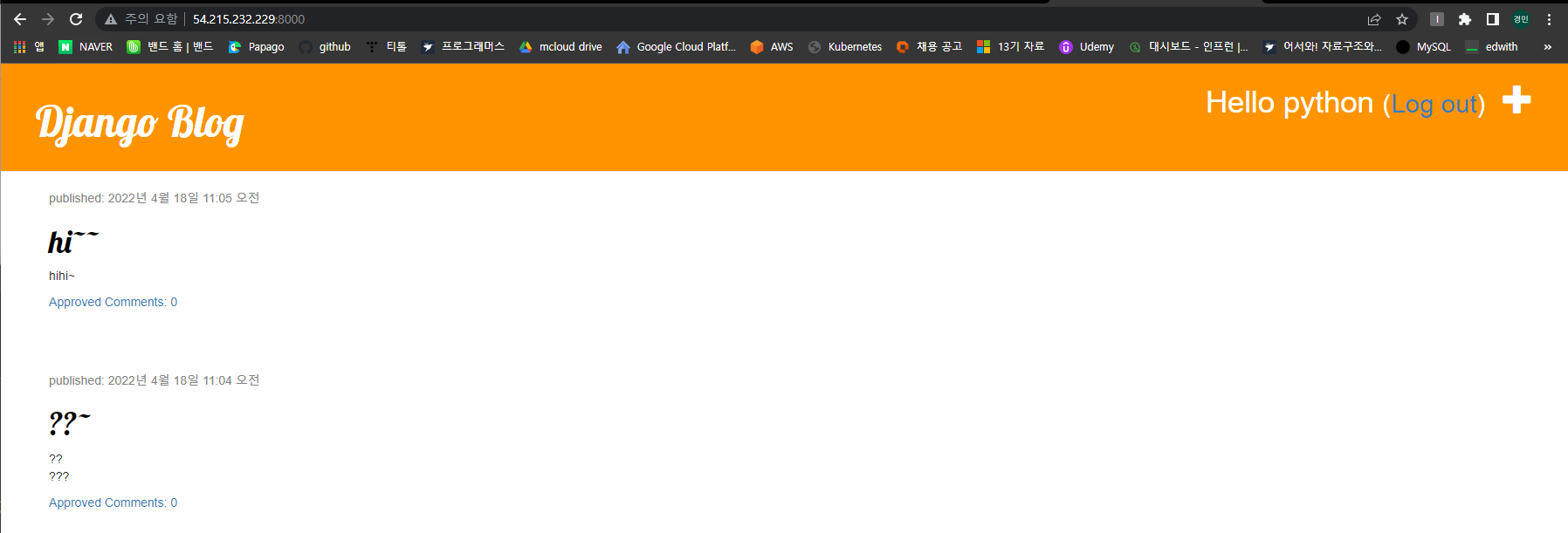
2. DB 사용부분 수정



## manage.py에서

|  |
| --- |
| #!/usr/bin/env python  """Django's command-line utility for administrative tasks."""  import os  import sys  def main():      """Run administrative tasks."""      os.environ.setdefault('DJANGO\_SETTINGS\_MODULE', **'mydjango.settings')**      try:          from django.core.management import execute\_from\_command\_line      except ImportError as exc:          raise ImportError(              "Couldn't import Django. Are you sure it's installed and "              "available on your PYTHONPATH environment variable? Did you "              "forget to activate a virtual environment?"          ) from exc      execute\_from\_command\_line(sys.argv)  if \_\_name\_\_ == '\_\_main\_\_':      main() |

|  |
| --- |
| # migrate  python3 manage.py migrate  # 실행하기  python3 manage.py runserver 0:8000 |



## Nginx 서버

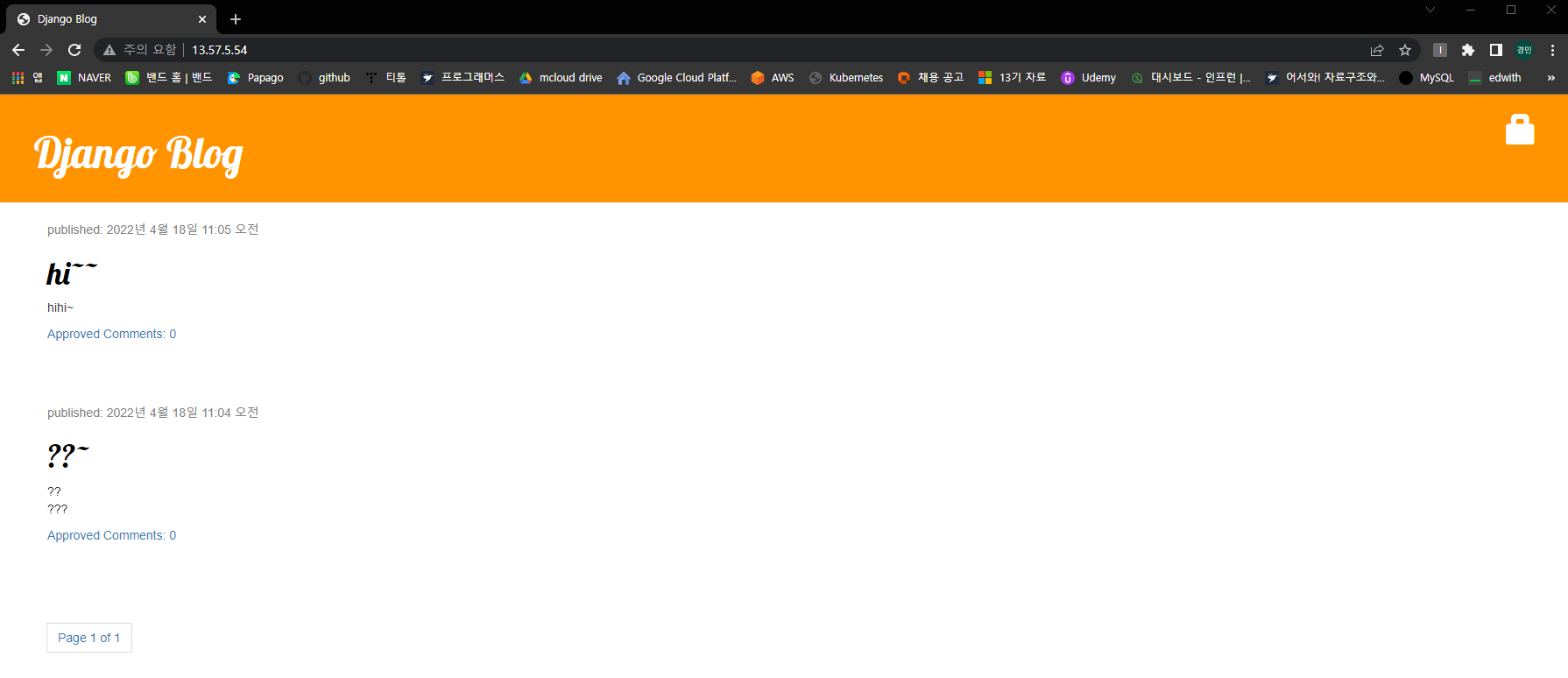
* Ubuntu 로 생성
* 일단 모든 접속을 허용한다

|  |
| --- |
| # centos ec2 생성  sudo amazon-linux-extras install nginx1  vi /etc/nginx/nginx.conf |

Nginx.conf

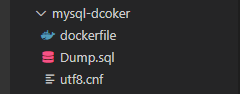
|  |
| --- |
| user nginx;  worker\_processes auto;  error\_log /var/log/nginx/error.log;  pid /run/nginx.pid;  # Load dynamic modules. See /usr/share/nginx/README.dynamic.  include /usr/share/nginx/modules/\*.conf;  events {  worker\_connections 1024;  }  http {  log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '  '$status $body\_bytes\_sent "$http\_referer" '  '"$http\_user\_agent" "$http\_x\_forwarded\_for"';  upstream backend {  **server django-ip:8000;**  }  access\_log /var/log/nginx/access.log main;  sendfile on;  tcp\_nopush on;  tcp\_nodelay on;  keepalive\_timeout 65;  types\_hash\_max\_size 2048;  default\_type application/octet-stream;  # Load modular configuration files from the /etc/nginx/conf.d directory.  # See http://nginx.org/en/docs/ngx\_core\_module.html#include  # for more information.  include /etc/nginx/conf.d/\*.conf;  server {  listen 80 default\_server;  listen [::]:80 default\_server;  server\_name \_;  root /usr/share/nginx/html/;  # Load configuration files for the default server block.  include /etc/nginx/default.d/\*.conf;  **location / {**  **proxy\_pass http://backend;**  **}**  error\_page 404 /404.html;  location = /40x.html {  }  error\_page 500 502 503 504 /50x.html;  location = /50x.html {  }  }  } |

|  |
| --- |
| # nginx 시작  systemctl start nginx |



# Django container 3tier

## Mysql 이미지 생성



Dockerfile

|  |
| --- |
| FROM mysql:5.7  COPY ./utf8.cnf /etc/mysql/conf.d/  COPY ./Dump.sql /docker-entrypoint-initdb.d/ |

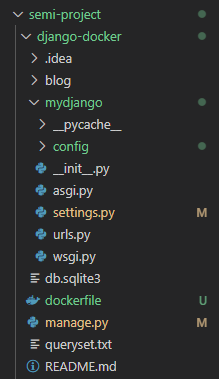
Uft8.cnf

|  |
| --- |
| [client]  default-character-set = utf8  [mysqld]  init\_connect = SET collation\_connection = utf8\_general\_ci  init\_connect = SET NAMES utf8  character-set-server = utf8  collation-server = utf8\_general\_ci  skip-character-set-client-handshake  [mysqldump]  default-character-set = utf8  [mysql]  default-character-set = utf8 |

도커 실행

|  |
| --- |
| docker network create --driver bridge msanet  docker run --name mysql-svc -d --net msanet -e MYSQL\_ROOT\_PASSWORD='rlarudals' -e MYSQL\_DATABASE='django\_db' -e MYSQL\_USER='python' -e MYSQL\_PASSWORD='python' -e MYSQL\_ROOT\_HOST='%' gymin97/django:mysql\_v1 |

## Django 이미지 생성



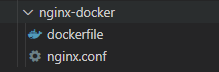
Dockerfile

|  |
| --- |
| FROM python:3.8  RUN python -m pip install --upgrade pip  WORKDIR /usr/src/app  RUN pip install django==3.1.13  RUN pip install pymysql==1.0.2  COPY . .  EXPOSE 8000  CMD ["python", "manage.py", "runserver", "0:8000"] |

도커 실행

|  |
| --- |
| docker run --name django-svc -d -p 8000:8000 --net msanet gymin97/django:django\_v2  # 잘 떠있는지 확인  curl localhost:8000 |

## Nginx 서버



Dockerfile

|  |
| --- |
| FROM nginx  COPY ./nginx.conf /etc/nginx/ |

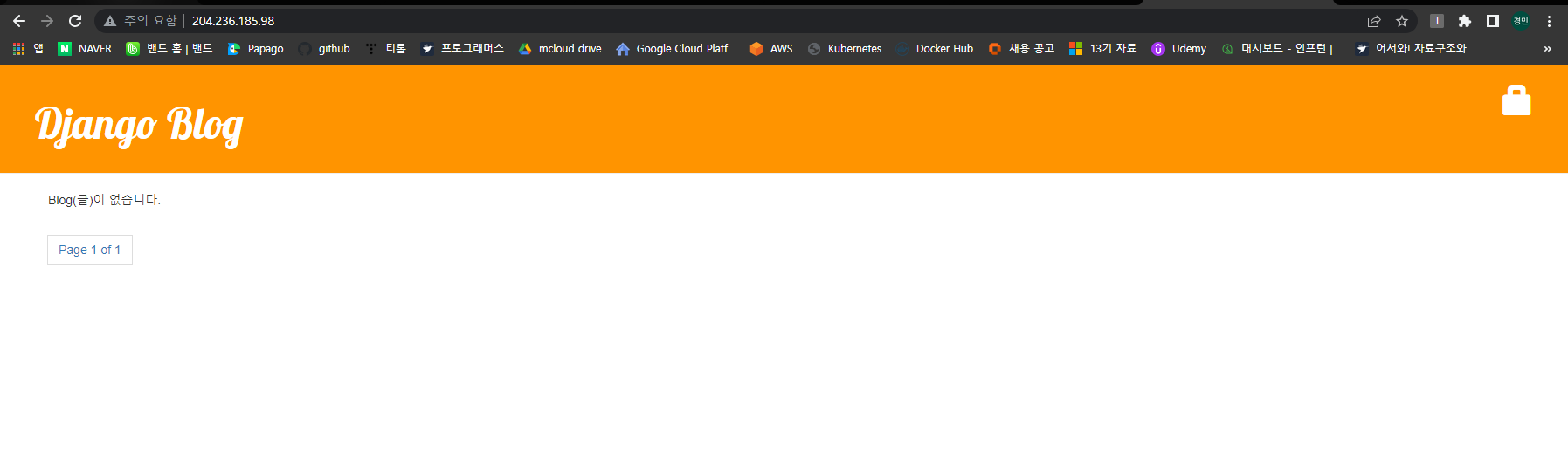
Nginx,conf

|  |
| --- |
| user nginx;  worker\_processes auto;  error\_log /var/log/nginx/error.log;  pid /run/nginx.pid;  # Load dynamic modules. See /usr/share/nginx/README.dynamic.  include /usr/share/nginx/modules/\*.conf;  events {  worker\_connections 1024;  }  http {  log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '  '$status $body\_bytes\_sent "$http\_referer" '  '"$http\_user\_agent" "$http\_x\_forwarded\_for"';  upstream backend {  **server django-svc:8000;**  **}**  access\_log /var/log/nginx/access.log main;  sendfile on;  tcp\_nopush on;  tcp\_nodelay on;  keepalive\_timeout 65;  types\_hash\_max\_size 2048;  default\_type application/octet-stream;  # Load modular configuration files from the /etc/nginx/conf.d directory.  # See http://nginx.org/en/docs/ngx\_core\_module.html#include  # for more information.  include /etc/nginx/conf.d/\*.conf;    server {  listen 80 default\_server;  listen [::]:80 default\_server;  server\_name \_;  root /usr/share/nginx/html/;  # Load configuration files for the default server block.  include /etc/nginx/default.d/\*.conf;  **location / {**  **proxy\_pass http://backend;**  **}**    error\_page 404 /404.html;  location = /40x.html {  }  error\_page 500 502 503 504 /50x.html;  location = /50x.html {  }  }  } |

도커 실행

|  |
| --- |
| docker run --name nginx-svc -d -p 80:80 --net msanet gymin97/django:nginx\_v1 |

실행 확인



# Docker-compose

## Docker compose 설치

|  |
| --- |
| DOCKER\_COMPOSE\_VERSION=v2.2.3  sudo curl -L "https://github.com/docker/compose/releases/download/${DOCKER\_COMPOSE\_VERSION}/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose  sudo chmod +x /usr/local/bin/docker-compose  sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose |

## Docker 설치

|  |
| --- |
| # amazon linux docker 설치  sudo yum update -y  sudo amazon-linux-extras install docker -y  systemctl start docker  systemctl enable docker |

## DB 생성

|  |
| --- |
| # conf 파일 생성  mkdir -p /work/test3/db/conf  cat <<EOF | sudo tee /work/test3/db/conf/my.cnf  [client]  default-character-set = utf8mb4  [mysql]  default-character-set = utf8mb4  [mysqld]  character-set-client-handshake = FALSE  character-set-server = utf8mb4  collation-server = utf8mb4\_unicode\_ci  EOF  # 환경변수파일 생성  mkdir -p /work/test3/db/env  cat <<EOF | sudo tee /work/test3/db/env/.env  MYSQL\_HOST=mysql-svc  MYSQL\_PORT=3306  MYSQL\_ROOT\_PASSWORD=frodo5020!!  MYSQL\_DATABASE=django\_db  MYSQL\_USER=python  MYSQL\_PASSWORD=python  MYSQL\_ROOT\_HOST='%'  EOF  # 데이터 마운트 경로  mkdir -p /work/test3/db/data |

## Nginx 생성

|  |
| --- |
| # conf 파일 생성  mkdir -p /work/test3/nginx/conf  cat <<EOF | sudo tee /work/test3/nginx/conf/nginx.conf  user nginx;  worker\_processes auto;  error\_log /var/log/nginx/error.log;  pid /run/nginx.pid;    # Load dynamic modules. See /usr/share/nginx/README.dynamic.  include /usr/share/nginx/modules/\*.conf;  events {  worker\_connections 1024;  }  http {  log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '  '$status $body\_bytes\_sent "$http\_referer" '  '"$http\_user\_agent" "$http\_x\_forwarded\_for"';    upstream backend {  server django-svc:8000;  }  access\_log /var/log/nginx/access.log main;  sendfile on;  tcp\_nopush on;  tcp\_nodelay on;  keepalive\_timeout 65;  types\_hash\_max\_size 2048;  default\_type application/octet-stream;  # Load modular configuration files from the /etc/nginx/conf.d directory.  # See http://nginx.org/en/docs/ngx\_core\_module.html#include  # for more information.  include /etc/nginx/conf.d/\*.conf;  server {  listen 80 default\_server;  listen [::]:80 default\_server;  server\_name \_;  root /usr/share/nginx/html/;  # Load configuration files for the default server block.  include /etc/nginx/default.d/\*.conf;  location / {  proxy\_pass http://backend;  proxy\_set\_header X-Forwarded-Port \$server\_port;  proxy\_set\_header X-Forwarded-Host \$host;  }  error\_page 404 /404.html;  location = /40x.html {  }  error\_page 500 502 503 504 /50x.html;  location = /50x.html {  }  }  }  EOF  # 데이터 마운트 경로  mkdir -p /work/test3/nginx/log |

## Docker-compose.yaml

|  |
| --- |
| mkdir -p /work/myproject  cat <<EOF| sudo tee /work/myproject/docker-compose.yml  version: "3"  services:  mysql-svc:  image: gymin97/django:mysql\_v1  volumes:  - /work/test3/db/conf:/etc/mysql/conf.d  - /work/test3/db/data:/var/lib/mysql  env\_file: /work/test3/db/env/.env  django-svc:  image: gymin97/django:django\_v2  restart: always  nginx-svc:  image: gymin97/django:nginx\_v1  volumes:  - /work/test3/nginx/conf:/etc/nginx  - /work/test3/nginx/log:/var/log/nginx  ports:  - "80:80"  EOF |

도커 컴포즈 실행

|  |
| --- |
| cd /work/myproject  docker-compose up |