

서버(django) 배포

비고

AWS

구분

배포

준비사항

Django project

AWS (<https://aws.amazon.com/ko/>)

참고

Deploy

cloud9

EC2

서버 설정

project clone

gunicorn

nginx

배포확인

DNS

Route53

HTTPS

Let's Encrypt

certbot

준비사항**Django project**

- 완성된 프로젝트
- 의존성 저장 - `pip freeze > requirements.txt`
- 원격저장소 업로드

[AWS \(<https://aws.amazon.com/ko/>\)](https://aws.amazon.com/ko/)

- AWS 계정 생성
- 기본정보입력
- 카드정보입력 (해외결제가 가능한 청크카드 or 신용카드)
- 휴대폰인증
- 완료후 로그인

참고

- vim 명령어
 - `i` 버튼으로 수정모드로 전환
 - 방향키를 이용하여 이동
 - 수정
 - `esc`로 수정모드 빠져나오기
 - `:wq` 명령어로 저장 후 종료

Deploy

cloud9

- AWS Management Console에서 Cloud9 검색 후 Create environment 클릭



- 이름입력 후 Next step

Name environment

Environment name and description

Name
The name needs to be unique per user. You can update it at any time in your environment settings.
movie-pit-computed
Limit: 60 characters

Description - Optional
This will appear on your environment's card in your dashboard. You can update it at any time in your environment settings.
Write a short description for your environment
Limit: 200 characters

Cancel **Next step**

- 설정
 - Platform
 - Ubuntu Server 18.04 LTS
 - Cost-saving setting
 - 일정시간 후 깨지도록 설정가능 (Never 설정시 과금주의)

Step 1
Name environment

Step 2
Configure settings

Step 3
Review

Configure settings

Environment settings

Environment type: **Info**
Run your environment in a new EC2 instance or an existing server. With EC2 instances, you can connect directly through Secure Shell (SSH) or connect to AWS Systems Manager (without opening inbound ports).

Create a new EC2 instance for environment (access via SSH)
Launch a new instance in this region that your environment can access directly via SSH.

Create a new no-ingress EC2 instance for environment (access via Systems Manager)
Launch a new instance in this region that your environment can access through Systems Manager.

Create and run in remote server (SSH connection)
Configure the secure connection to the remote server for your environment.

Instance type
 t2.micro (1 GB RAM + 1 vCPU)
Recommended for small-scale web projects and exploration.

 t2.small (2 GB RAM + 2 vCPU)
Recommended for small-scale web projects.

 m5.large (8 GB RAM + 2 vCPU)
Recommended for production and general-purpose development.

 Other instance type
Select an instance type.

Platform
 Amazon Linux 2 (recommended)

 Amazon Linux AMI

 Ubuntu Server 18.04 LTS

Cost-saving setting
Choose a predetermined amount of time to auto-hibernate your environment and prevent unnecessary charges. We recommend a hibernation setting of half an hour of no activity to maximize savings.
After 30 minutes (default)

AWS role
AWS Cloud9 creates a service-linked role for you. This allows AWS Cloud9 to call other AWS services on your behalf. You can delete the role from the AWS IAM console once you no longer have any AWS Cloud9 environments.[Learn more](#)

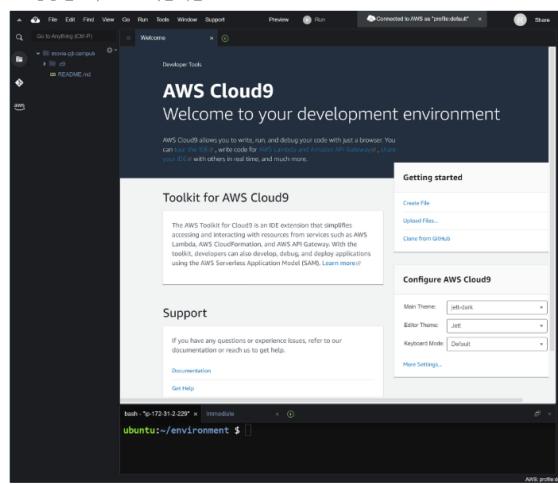
AWSServiceRoleForAWSCloud9

Network settings (advanced)

No tags associated with the resource.
Add new tag
You can add 50 more tags.

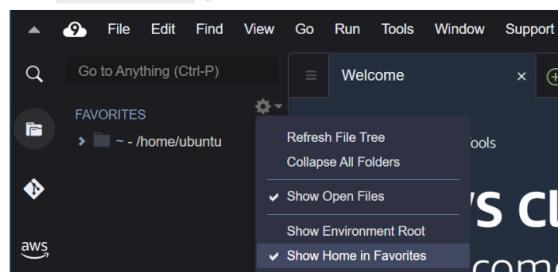
Cancel **Previous step** **Next step**

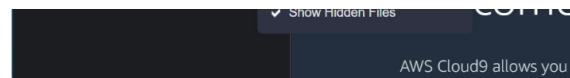
- 생성 완료 후 cloud9 화면 확인



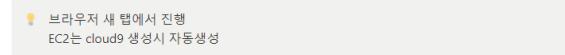
- 파일트리 설정 (home directory 기준으로 진행)

- Show Environment Root 체크해제
- Show Home in Favorites 체크





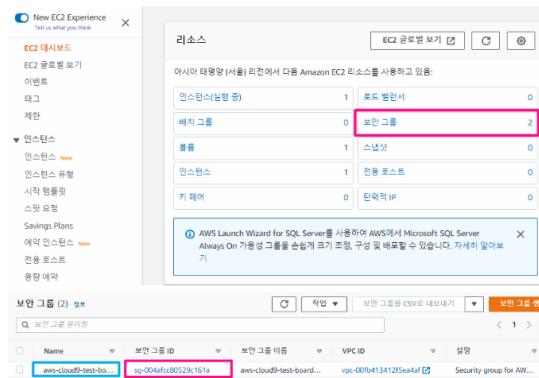
EC2



- 서비스 검색



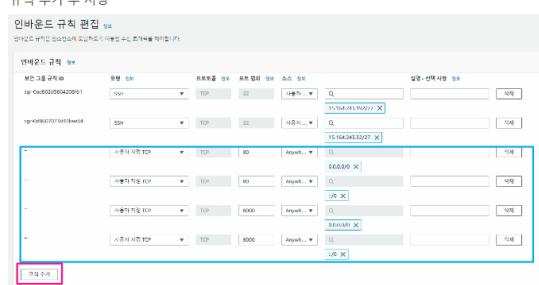
- 보안그룹 탭 이동 후 생성된 보안 그룹 ID 클릭



- 하단 화면의 인바운드 규칙 편집



- 규칙 추가 후 저장



- 포트 범위 - 80, 8000(테스트용)

- 소스 - 0.0.0.0/0, ::/0

서버 설정



- pyenv 설치 후 터미널 재시작

- <https://github.com/pyenv/pyenv>

```
git clone https://github.com/pyenv/pyenv.git ~/.pyenv
sed -Ei -e '/^(["#"]$)/ {a \
export PYENV_ROOT="$HOME/.pyenv"
a \
export PATH="$PYENV_ROOT/bin:$PATH"
a \
'-e 'a' -e '$!{n;ba};' ~./profile
echo 'eval "$(pyenv init --path)"' > ~./profile
echo 'eval "$(pyenv init -)"' > ~./bashrc
source ~./profile
source ~./bashrc
```

- pyenv 설치 확인

```
pyenv -v  
# 출력 확인 => pyenv VERSION_INFO
```

- python 설치 (프로젝트에서 사용한 버전설치)
 - global 설정 후 버전확인

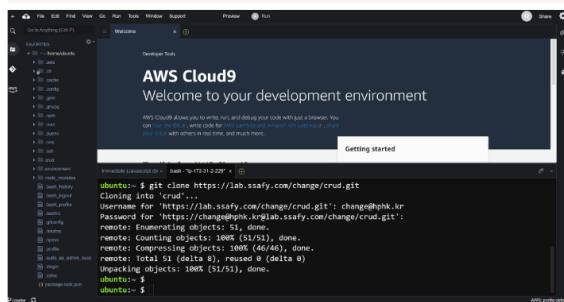
```
pyenv install 3.9.X  
pyenv global 3.9.X  
python -V  
=> Python 3.9.X
```

project clone

💡 프로젝트 폴더와 마스터 앱, 두 이름에 주의하며 진행해주세요.
두 폴더의 이름을 통일하면 조금 더 편하게 설정할 수 있습니다.

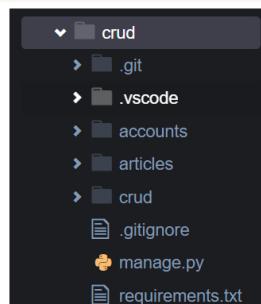
- clone
 - home을 기준으로 진행

```
cd ~  
git clone {project_remote_url}
```



- 폴더구조
 - 프로젝트 이름은 변수처럼 사용예정 이름을 기억해주세요!

```
home/  
ubuntu/  
  (프로젝트 폴더)  
    (마스터 앱)  
      settings.py  
      ...  
      (앱1)  
      (앱2)  
      ...  
    manage.py  
    requirements.txt
```



- 프로젝트 폴더로 이동

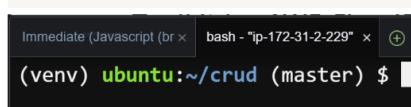
```
cd ~/({프로젝트 폴더})
```

- 가상환경생성 (가상환경이름 기억)

```
python -m venv venv
```

- 가상환경 activate (window와 명령어가 다름)

```
source venv/bin/activate
```



- 라이브러리 설치

```
pip install -r requirements.txt
```

- 마이그레이션

```
python manage.py migrate
```

- createsuperuser

```
python manage.py createsuperuser
```

- loaddata (fixture가 있는 경우)

```
python manage.py loaddata {데이터 dump 파일}
```

- collectstatic

- o `settings.py` 수정

```
# settings.py  
  
STATIC_ROOT = BASE_DIR / 'staticfiles'
```

- collectstatic

```
python manage.py collectstatic
```

gunicorn

- ## • 설치

8. <https://docs.unicorn.org/en/stable/install.html>

```
pip install gunicorn
```

- 서버실행

```
gunicorn --bind 0.0.0.0:8000 {마스터 앱}.wsgi:application
```

- django 페이지 확인

DisallowHost at /

```
Invalid HTTP_HOST header: '13.209.9.14:8000'. You may need to add '13.209.9.14' to ALLOWED_HOSTS.  
Request Method: GET  
Request URL: http://13.209.9.14:8000/  
Django Version: 3.2.12  
Exception Type: DisallowedHost  
Exception Value: Invalid HTTP_HOST header: '13.209.9.14:8000'. You may need to add '13.209.9.14' to ALLOWED_HOSTS.  
Exception Location: /home/ubuntu/pyenv/versions/3.9/lib/python3.9/site-packages/django/http/request.py, line 149, in get_host  
Python Executable: /home/ubuntu/pyenv/versions/3.9/bin/python3  
Python Version: 3.9.10  
Python Path:  
['/home/ubuntu/boardack'],  
'/usr/local/lib/python3.9',  
'/usr/local/lib/python3.9/lib-dynload',  
'/usr/local/lib/python3.9/lib/python3.9/lib-dynload.so',  
'/home/ubuntu/pyenv/versions/3.9/lib/python3.9',  
'/home/ubuntu/pyenv/versions/3.9/lib/python3.9/lib-dynload.so',  
'/home/ubuntu/pyenv/versions/3.9/lib/python3.9/site-packages'  
Environment: The full environment is shown below.
```

- #### **• settings.py** 수정 후 서버 재시작

```
# settings.py

ALLOWED_HOSTS = [
    # 할당된 EC2 인스턴스의 IP주소 입력. 현재 예시의 경우 아래와 같이 입력
    '13.209.9.14',
]
```

- 아래의 코드를 각자 프로젝트 이름에 맞게 수정 후 메모장에 입력(복사)

```
[Unit]
Description=gunicorn daemon
After=network.target

[Service]
User=ubuntu
Group=www-data
WorkingDirectory=/home/ubuntu/{프로젝트 폴더}
ExecStart=/home/ubuntu/{프로젝트 폴더}/venv/bin/gunicorn \
          -w 3 \
          -b 127.0.0.1:8000 \
          {미스터 앱}.wsgi:application

[Install]
WantedBy=multi-user.target
```

- 위에 작성한 내용으로 아래와 같이 파일수정

```
sudo vi /etc/systemd/system/gunicorn.service
```

- 시스템 데몬 재시작

```
sudo systemctl daemon-reload
```

- 서비스 실행 및 등록

```
sudo systemctl start gunicorn
sudo systemctl enable gunicorn
sudo systemctl status gunicorn.service

# 중지
# sudo systemctl stop gunicorn
# 재시작
# sudo systemctl restart gunicorn
```

nginx

- 💡 vim을 사용하여 터미널에서 파일을 수정합니다.
사용법을 숙지하고 진행해주세요.

- 설치

```
sudo apt-get update
sudo apt-get install -y nginx
```

- 복사할 코드 작성

- 아래의 코드에서 각자의 프로젝트이름에 맞게 수정 후 메모장에 입력
- staticfiles 의 경우 다른 폴더를 썼다면 이름수정

```
server {
    listen 80;
    server_name {서버IP주소};

    location /static/ {
        root /home/ubuntu/{프로젝트 폴더}/staticfiles/;
    }

    location / {
        include proxy_params;
        proxy_pass http://127.0.0.1:8000;
    }
}
```

- 파일수정

```
sudo vi /etc/nginx/sites-available/django_test
```

- 사이트 추가

```
sudo ln -s /etc/nginx/sites-available/django_test /etc/nginx/sites-enabled
```

- 80번 포트의 프로세서 종료

```
sudo lsof -t -i tcp:80 -s tcp:listen | sudo xargs kill
```

- nginx restart ⇒ status 확인

```
sudo systemctl restart nginx
systemctl status nginx.service
```

```
(venv) ubuntu:~/crud (master) $ systemctl status nginx.service
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
   Active: failed (Result: exit-code) since Thu 2021-05-13 14:27:19 UTC; 1min 1s ago
     Docs: man:nginx(8)
   Process: 28177 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=1/F
   Process: 28186 ExecStartPre=/usr/sbin/nginx -t -q daemon on; master_process on; (code=exited, s

May 13 14:27:19 ip-172-31-2-229 nginx[28177]: nginx: [emerg] bind() to :::80 failed (98: Address
May 13 14:27:19 ip-172-31-2-229 nginx[28177]: nginx: [emerg] bind() to ::1:80 failed (98: Address
May 13 14:27:19 ip-172-31-2-229 nginx[28177]: nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address
May 13 14:27:19 ip-172-31-2-229 nginx[28177]: nginx: [emerg] bind() to ::1:80 failed (98: Address a
May 13 14:27:19 ip-172-31-2-229 nginx[28177]: nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address
May 13 14:27:19 ip-172-31-2-229 nginx[28177]: nginx: [emerg] still could not bind()
May 13 14:27:19 ip-172-31-2-229 systemd[1]: nginx.service: Control process exited, code=exited statu
May 13 14:27:19 ip-172-31-2-229 systemd[1]: nginx.service: Failed with result 'exit-code'.
May 13 14:27:19 ip-172-31-2-229 systemd[1]: Failed to start A high performance web server and a reve
```

배포확인

- EC2 대시보드에서 퍼블릭 IP로 접속

DNS

 도메인 결제 후 진행합니다.

Route53

- 호스팅 영역 ⇒ 도메인 선택 ⇒ 레코드 생성
 - 레코드 유형 - A
 - 값 - { 서버 IP 주소 }

Route 53 > 호스팅 영역 > changeo.click > 레코드 생성

빠른 레코드 생성 Info

▼ 레코드 1

레코드 유형 Info	레코드 유형 Info	선택
blog	changes.click	
설정 항목: a-e, 0-8 까지 (# \$ % & ! ^ * - . / ; < = > + @ { } _ -)	A-IPv4 주소 및 일부 AWS 리소스 트래픽 ...	▼
3.35.230.223		▶ Info
별도로 줄여서 여러 값을 입력합니다.		

TTL(초) Info

300	1분	1시간	1일
-----	----	-----	----

경고: 60~172,800(초)

라우팅 정책 Info

단순 라우팅	▼
--------	---

워크

레코드 생성

- nginx 설정 수정

```
sudo vi /etc/nginx/sites-available/django_test
```

```
server {
    listen 80;
    server_name {서버IP주소} {도메인주소};

    location /static/ {
        root /home/ubuntu/{프로젝트 폴더}/staticfiles/;
    }

    location / {
        include proxy_params;
        proxy_pass http://127.0.0.1:8000;
    }
}
```

- `settings.py` 수정

```
ALLOWED_HOSTS = [  
    '{서버IP주소}',  
    '{도메인주소}'  
]
```

- 수정 후 `gunicorn`, `nginx` 재시작

```
sudo systemctl restart gunicorn  
sudo systemctl restart nginx
```

HTTPS

Domain 연결이 안되어 있을 경우, HTTPS 적용 불가

📌 <https://howhttps.works/ko/>

Let's Encrypt

 <https://letsencrypt.org/ko/getting-started/>
certbot 사용을 권장

certbot

<https://certbot.eff.org/>

- Software(nqinx), System(Ubuntu) 선택 후 가이드진행



- core 설치 (EC2에 설치되어있음)

```
sudo snap install core; sudo snap refresh core
```

- certbot 설치

```
sudo snap install --classic certbot
```

- 심볼릭 링크

```
sudo ln -s /snap/bin/certbot /usr/bin/certbot
```

- 자동 설정

```
sudo certbot --nginx
```

- 이메일 입력

```
(venv) ubuntu:~/crud (master) $ sudo certbot --nginx
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Plugins selected: Authenticator nginx, Installer nginx
Enter email address (used for urgent renewal and security notices)
(Enter 'c' to cancel): █
```

- 동의(y 입력)

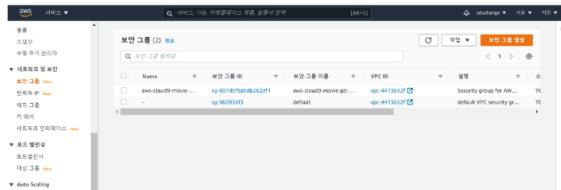
```
Please read the Terms of Service at
https://letsencrypt.org/documents/LE-SA-v1.2-November-15-2017.pdf. You must
agree in order to register with the ACME server. Do you agree?
-----
(Y)es/(N)o: █

Would you be willing, once your first certificate is successfully issued, to
share your email address with the Electronic Frontier Foundation, a founding
partner of the Let's Encrypt project and the non-profit organization that
develops Certbot? We'd like to send you email about our work encrypting the web,
EFF news, campaigns, and ways to support digital freedom.
-----
(Y)es/(N)o: █
```

- 도메인 선택

Which names would you like to activate HTTPS for?

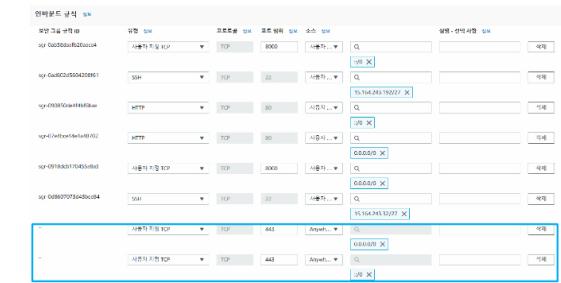
- EC2 보안그룹 탭 이동 후 생성된 보안 그룹 ID 클릭



- 하단 화면의 인바운드 규칙 편집



- 규칙 추가 후 저장



- 포트 범위 - 443

- 소스 - `0.0.0.0/0`, `::/0`

- <https://> 주소로 요청 후 확인

