EC2 장고 웹서버 연동

2020.05.07

참고 블로그 : <https://dailyheumsi.tistory.com/19>

장고로 서비스를 하면 기본적으로 8000포트로 열립니다. 따라서 웹서버를 통해서 80포트인 웹으로 접근했을 때 8000포트 앱서버로 접근하도록 만들어 줘야합니다. 이때 웹서버와 앱서버 사이를 통신하는 인터페이스가 wsgi 입니다.

이전에 사용한 manage.py runserver 를 대신할 wsgi 서버를 설치합니다. Wsgi 서버에는 uwsgi, gunicorn 등 다양하게 있지만 상대적으로 쉬운gunicorn을 사용하도록 하겠습니다.

1. 가상환경 활성화

source myvenv/bin/activate

1. 기존에 사용하던 nohup 말고 screen 을 이용해서 서버를 유지하는 것으로 바꾸겠습니다. 이유는 nohup은 실행을 시키는 것외에 다시 들어가서 설정을 바꾼다거나 재실행을 할 수 없습니다. 그래서 사용하는 것이 screen 입니다. 지속 유지되는 터미널을 만든다라고 생각하시면 쉽습니다.

* screen 설치

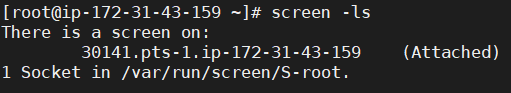
yum install screen

* screen 생성 (이름 : funiv)

screen –S funiv

* screen 생성 확인

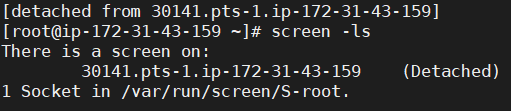
screen –ls



30141 이라는 아이디로 스크린이 생성되었습니다.

Attached 는 현재 해당 스크린으로 들어와 있다는 상태를 나타냅니다.

Ctrl + a + d 를 누르면 스크린을 벗어날 수 있습니다.



screen 밖으로 나왔기 때문에 Detached로 변경된 것을 볼 수 있습니다.

screen을 삭제하려면 screen –X –S 삭제할스크린 kill 입니다.

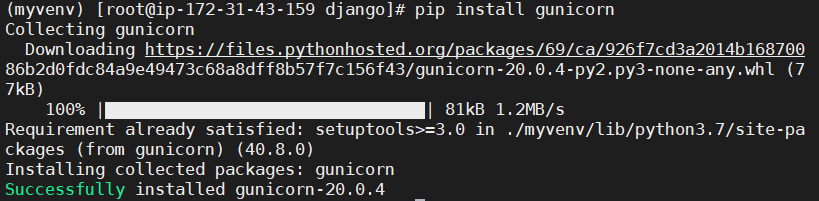
gunicorn 실행 종료는 ps –ef | grep python 으로 gunicorn으로 돌아가는 프로세스 아이디를 알아 낸 다음 kill 프로세스아이디 를 이용해서 서버를 종료시킬 수 있습니다.

* 다시 screen에 들어가서 gunicorn을 이용해 장고 서버를 실행

screen –r 30141

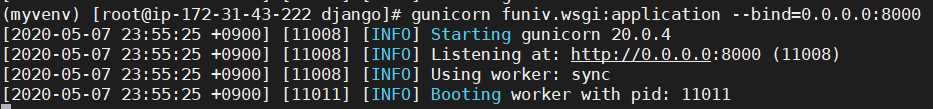
* gunicorn 설치

pip install gunicorn



* gunicorn 실행 (여기서 프로젝트 명은 funiv 입니다)

gunicorn 프로젝트명.wsgi:application --bind=0.0.0.0:8000



* 스크린 빠져나오기 및 세션 종료 후 서버 유지되는지 확인하기

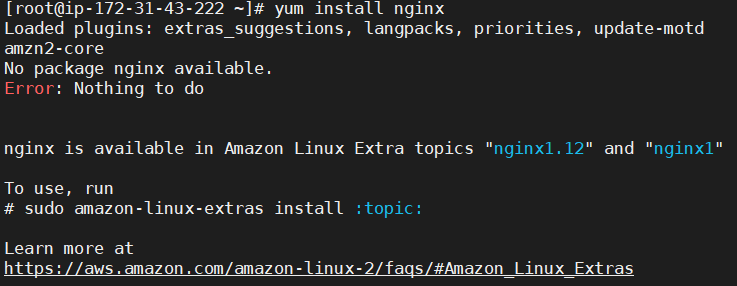
Nginx는 웹 서버로서 비슷한 소프트웨어로 apache가 있습니다. Nginx를 통해 80포트로 접속을 해도 내가 설정한 웹 사이트를 볼 수 있게 연결 시켜줄 수 있습니다.

사용자 요청 – nginx – wsgi – Django 흐름입니다.

<https://aws.amazon.com/ko/premiumsupport/knowledge-center/ec2-install-extras-library-software/>

1. Nginx 설치

yum install nginx 를 치면 다른 명령어로 설치하라고 알려줍니다.



sudo amazon-linux-extras install nginx1.12

1. Nginx 폴더로 이동하여 설정 확인

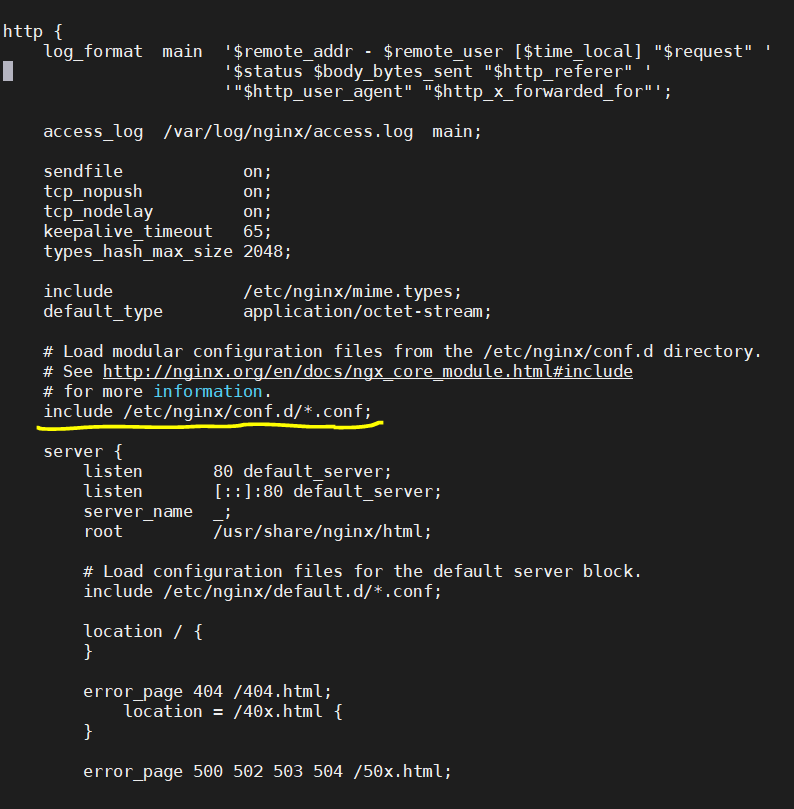
* Nginx 폴더로 이동

cd /etc/nginx

* 설정파일 확인

vi nginx.conf

* http 블록 안에 있는 include 부분 확인.



여기서는 /etc/nginx/conf.d 폴더안에 어떠한 이름이든 확장자가 conf로 끝나는 설정파일을 포함시킨다고 되어있습니다. 따라서 conf.d 폴더로 이동해 설정파일을 만들어 저장하도록 하겠습니다.

* conf.d 로 이동

cd /etc/nginx/conf.d

* 프로젝트 설정파일 생성

vi funiv.conf

* 내용 작성 및 저장

server {

listen 80;

server\_name funiv.net;

location / {

proxy\_pass http://127.0.0.1:8000/;

}

}

funiv.net 서버이름에 80포트로 들어온 요청사항은 내부서버의 8000포트 앱서버로 요청을 보내준다는 뜻입니다.

* nginx 설정 저장

nginx –t

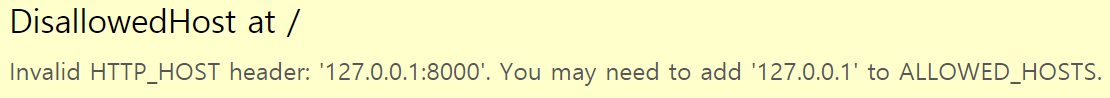
* nginx 재시작

service nginx restart

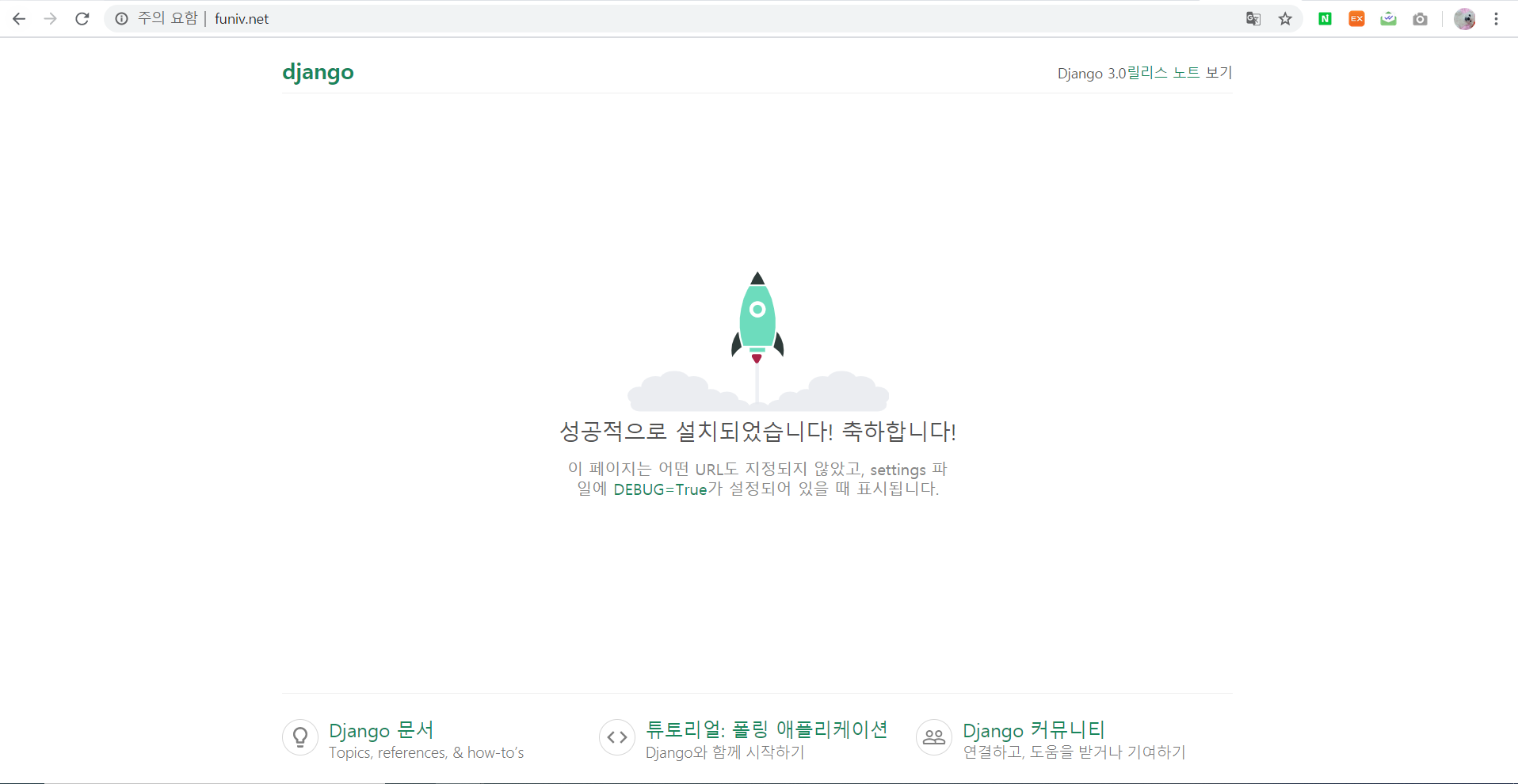
* 확인

설정해 두었던 도메인으로 접속해 봅니다.

이때 Django settings.py에서 허용된 호스트에 127.0.0.1을 추가하지 않았다면 아래와 같은 에러가 뜰 것입니다.



settings.py로 가서 추가해주고 서버를 재시작하면 적용된 것을 확인할 수 있습니다. Ctrl + C



요즘 웹사이트는 https가 거의 필수입니다. 유료정책인 것도 있지만 무료로 사용가능한 인증서도 있기에 해당 인증서를 이용하여 https를 이용해 보도록 하겠습니다. 단점이라면 90일마다 갱신하는 것이지만 추가 작업을 통해 자동으로 발급받게 설정해두면 됩니다.

1. EPEL repository 활성화

amazon-linux-extras install epel

1. Let’s Encrypt 를 사용하기 위한 패키지 설치

yum install -y certbot-nginx

1. certbot이 80포트를 사용할지도 모르기에 443 포트를 열어줘야합니다.

* 방화벽 패키지 설치

yum install firewalld

* 시스템에 등록하고 작동되도록 실행

systemctl unmask firewalld

systemctl enable firewalld

systemctl start firewalld

* 80포트, 8000포트 열어주기

firewall-cmd --permanent --zone=public --add-port=80/tcp

firewall-cmd --reload

firewall-cmd --list-ports

* 443 포트 열어주기

firewall-cmd --permanent --add-service=https

firewall-cmd --reload

1. 인증받기

sudo certbot certonly --standalone -n --agree-tos --email 인증할이메일 -d 설정한도메인

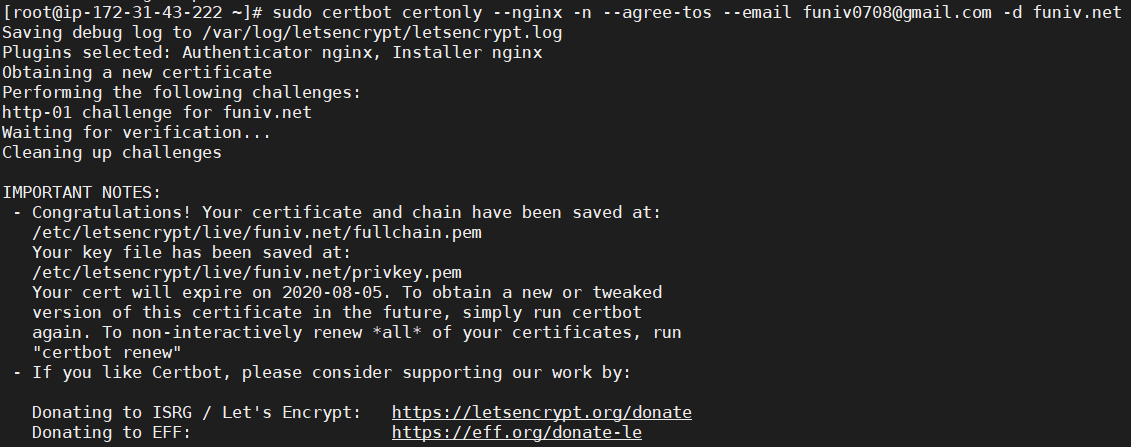
포트를 열어 줬음에도 오류가 납니다.



해당 80포트가 nginx가 사용중이라 뜨는 오류입니다

* 다른 방법으로 인증시도

certbot certonly --nginx -n --agree-tos --email 이메일 -d 도메인



1. 제대로 생성 됐는지 확인

cd /etc/letsencrypt/live/도메인

해당 폴더 안에 4개의 파일이 생성 되어있다면 제대로 인증서 발급이 된 것입니다.



1. nginx 설정 바꾸기

server {

listen 80;

server\_name funiv.net.;

return 301 https://$host$request\_uri;

# 301 리다이렉트는 영구적으로 옮겼을 때 사용합니다.

}

server {;

listen 443;

server\_name funiv.net.

ssl on;

ssl\_certificate /etc/letsencrypt/live/funiv.net/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/funiv.net/privkey.pem;

ssl\_session\_cache shared:SSL:1m;

ssl\_session\_timeout 10m;

ssl\_protocols TLSv1 TLSv1.1 TLSv1.2;

ssl\_ciphers HIGH:SEED:!aNULL:!eNULL:!EXPORT:!DES:!RC4:!MD5:!PSK:!RSAPSK:!aDH:!aECDH:!EDH-DSS-DES-CBC3-SHA:!KRB5-DES-CBC3-SHA:!SRP;

ssl\_prefer\_server\_ciphers on;

location / {

proxy\_pass http://127.0.0.1:8000/;

}

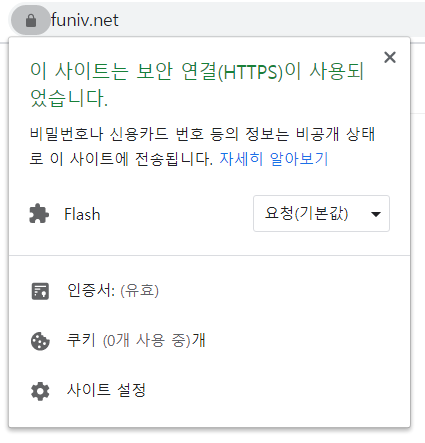
}

1. nginx 설정 변경 저장 및 리로드

nginx –t

service nginx reload

1. https 적용 되는지 확인



1. crontab을 이용해 인증서 자동 갱신

* crontab 편집기 실행

sudo crontab –e

* 내용 작성 및 저장

0 3 1 \* \* letsencrypt renew >> /var/log/le-renew.log

3 3 1 \* \* systemctl reload gunicorn

4 3 1 \* \* systemctl reload nginx