Django

> Django 서버 배포

django 추가코드

0. settings.py

- ALLOWED_HOSTS
  - EC2 서버주소를 등록
  - 편하게 배포하기 위하여 `*` 로 등록 후 추후 수정가능

```
# settings.py
ALLOWED_HOSTS = [
    '.compute.amazonaws.com',
    '*',
]
```

1. 의존성 저장

- freeze

```
pip freeze > requirements.txt
```

2. git push

- 원격저장소에 업로드 (add, commit, push)

aws

0. 준비

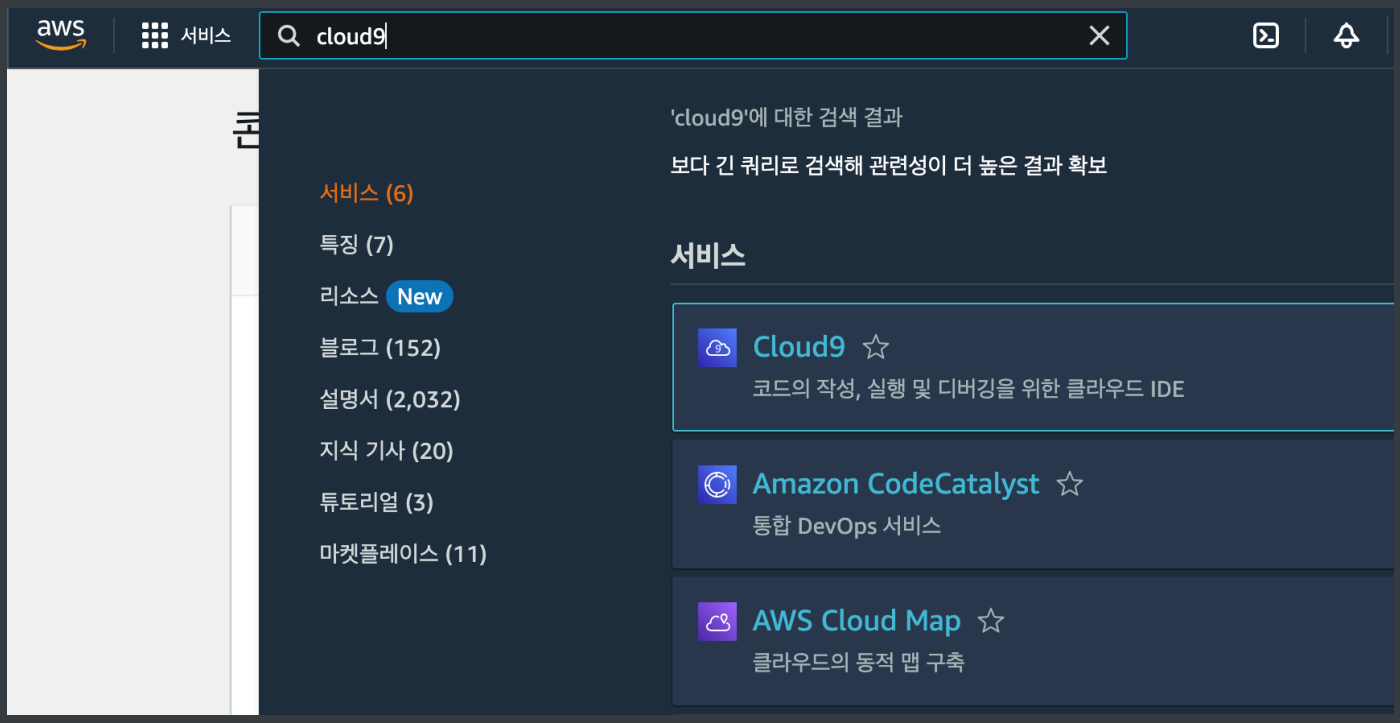
- 완성된 django프로젝트
- 해외결제가 가능한 체크카드 or 신용카드
- 여유로운 마음

1. https://aws.amazon.com/ko/

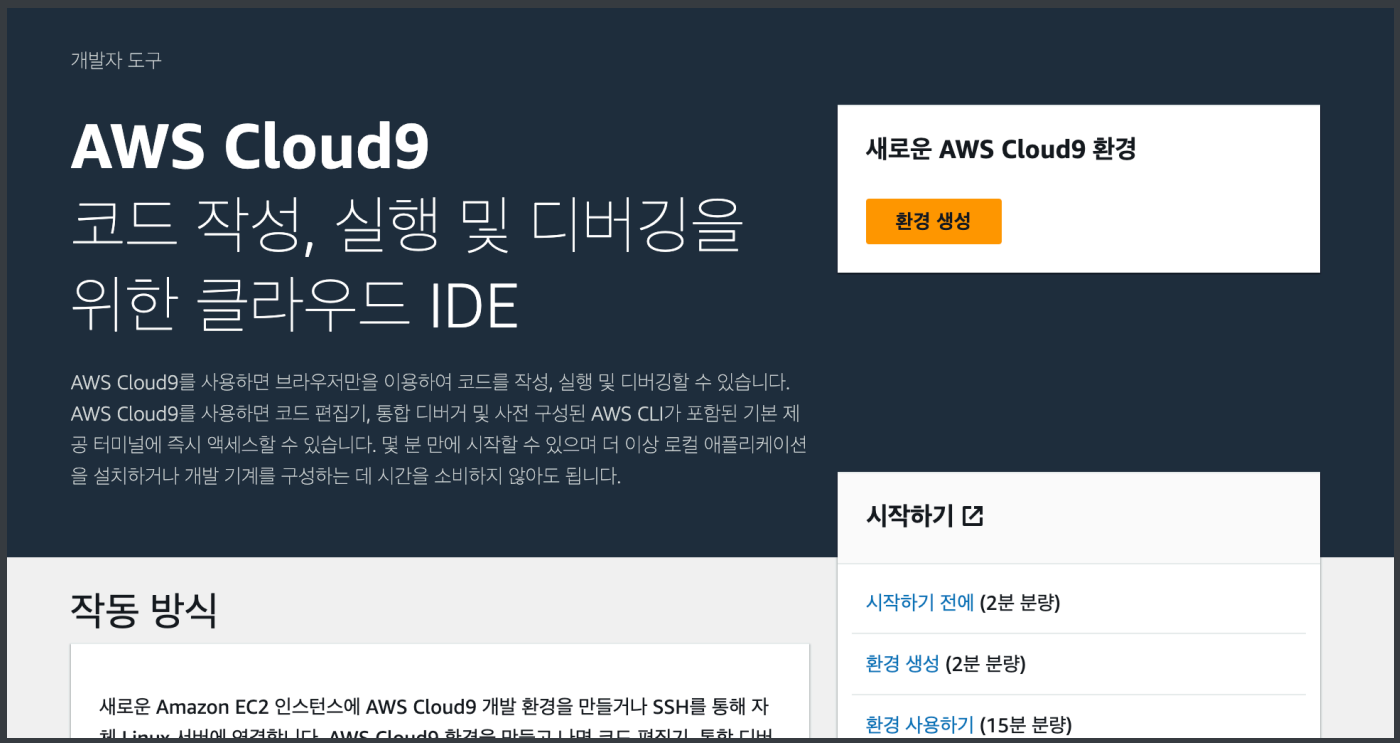
- AWS 계정 생성
- 기본정보입력
- 카드정보입력
- 휴대폰인증
- 완료후 로그인

2. aws cloud9

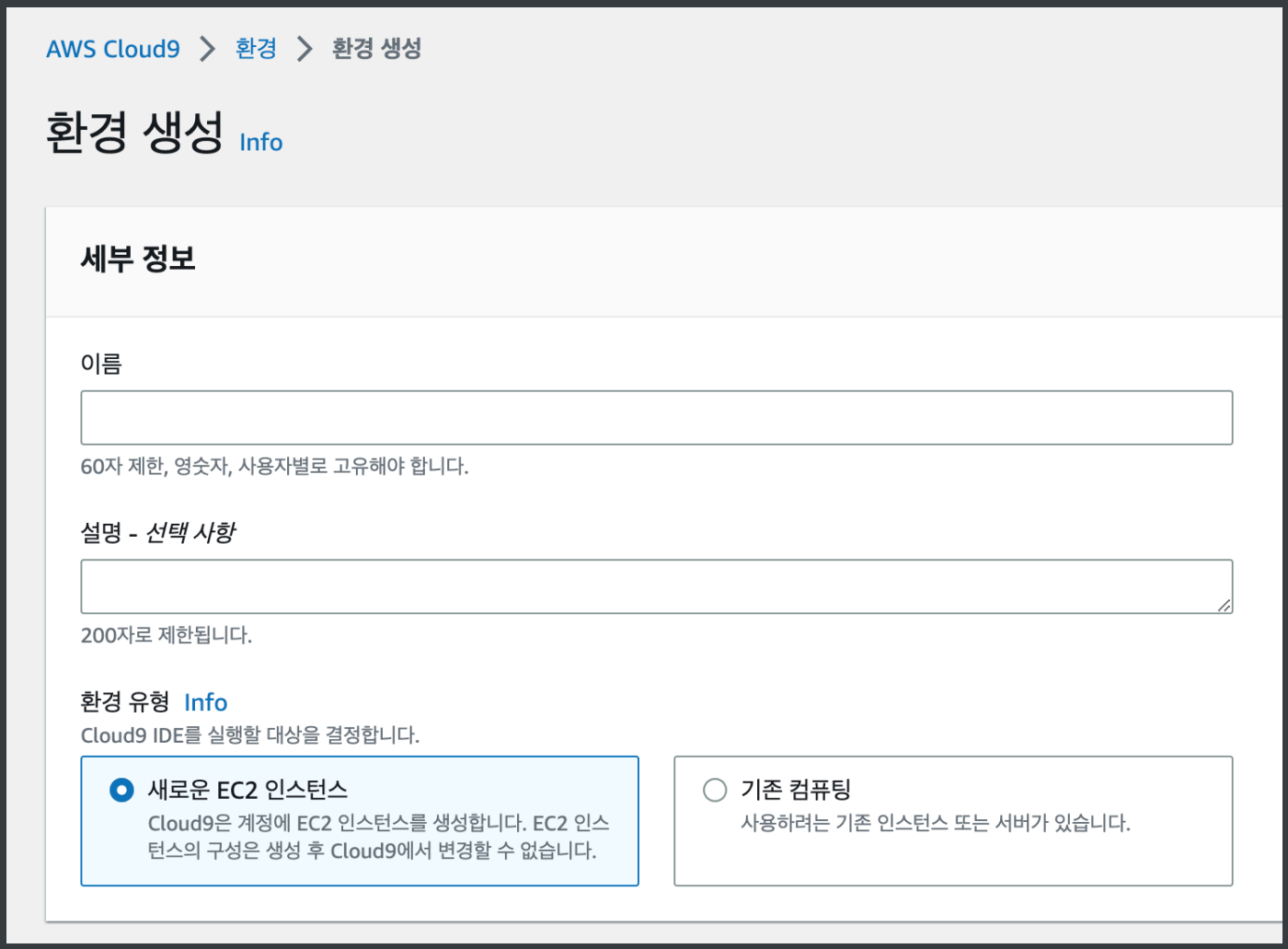
- AWS Management Console 에서 Cloud9 검색


aws
서비스
cloud9
'cloud9'에 대한 검색 결과
보다 긴 쿼리로 검색해 관련성이 더 높은 결과 확보
서비스 (6)
특징 (7)
리소스 New
블로그 (152)
설명서 (2,032)
지식 기사 (20)
튜토리얼 (3)
마켓플레이스 (11)
서비스
Cloud9
코드의 작성, 실행 및 디버깅을 위한 클라우드 IDE
Amazon CodeCatalyst
통합 DevOps 서비스
AWS Cloud Map
클라우드의 동적 맵 구축

- 환경 생성

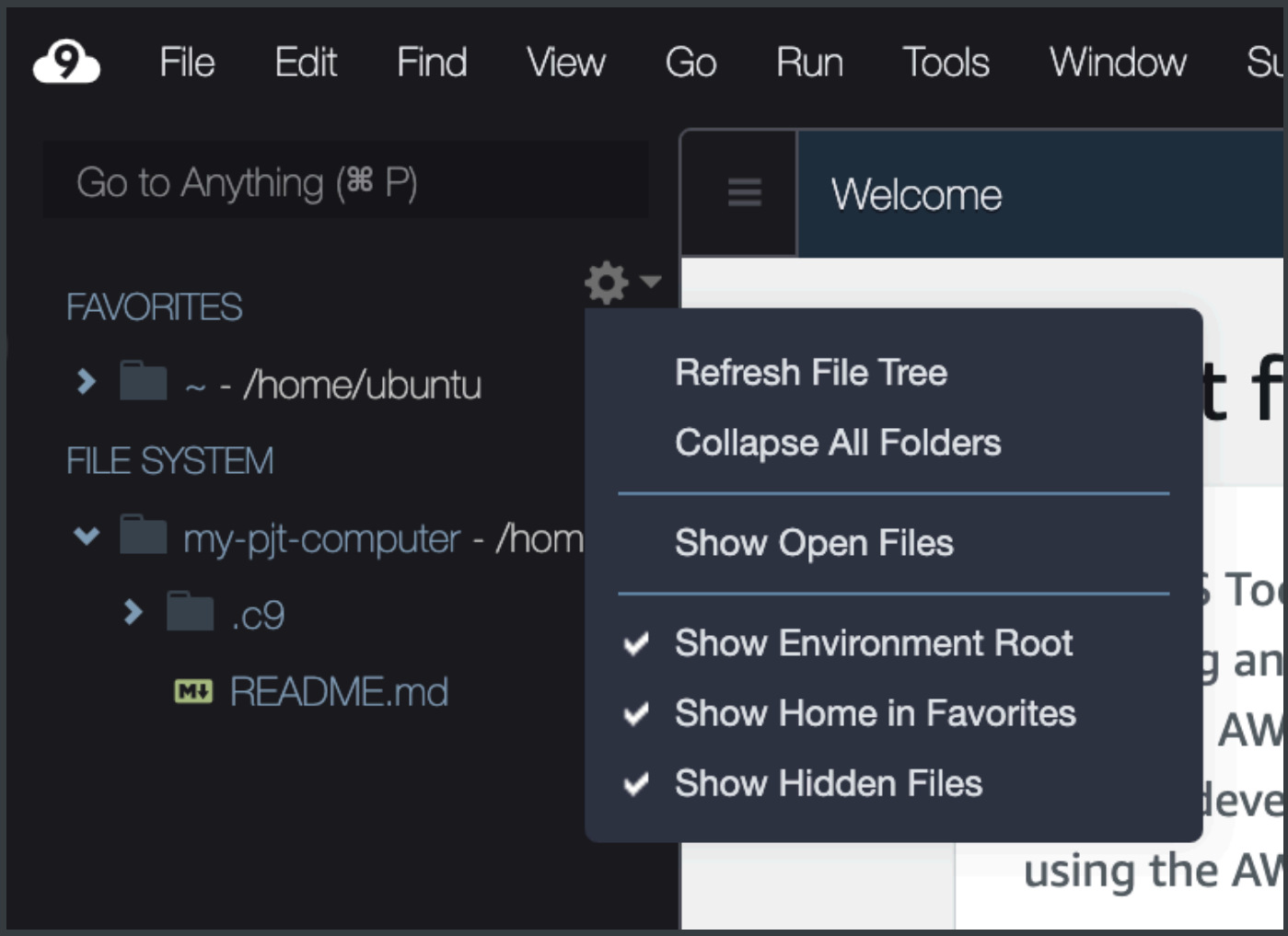

개발자 도구
AWS Cloud9
코드 작성, 실행 및 디버깅을 위한 클라우드 IDE
AWS Cloud9를 사용하면 브라우저만을 이용하여 코드를 작성, 실행 및 디버깅할 수 있습니다. AWS Cloud9를 사용하면 코드 편집기, 통합 디버거 및 사전 구성된 AWS CLI가 포함된 기본 제공 터미널에 즉시 액세스할 수 있습니다. 몇 분 만에 시작할 수 있으며 더 이상 로컬 애플리케이션을 설치하거나 개발 기계를 구성하는 데 시간을 소비하지 않아도 됩니다.
새로운 AWS Cloud9 환경
환경 생성
시작하기
시작하기 전에 (2분 분량)
환경 생성 (2분 분량)
환경 사용하기 (15분 분량)
작동 방식
새로운 Amazon EC2 인스턴스에 AWS Cloud9 개발 환경을 만들거나 SSH를 통해 자

- 세부 정보



- 인스턴스
  - 서버를 끄지 않으려면 `안함` 으로 설정

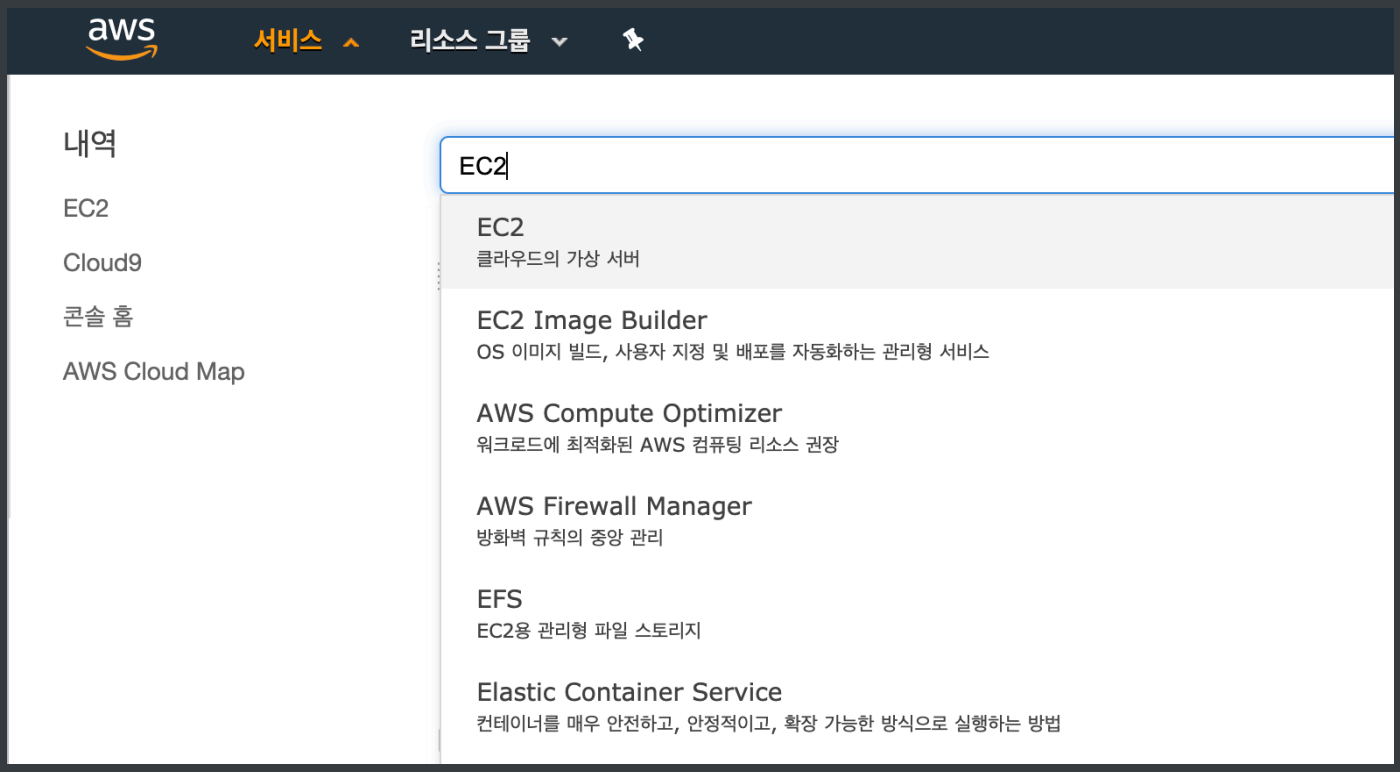
- 파일트리설정
  - [v] Show Home in Favorites
  - [v] Show Hidden Files



EC2

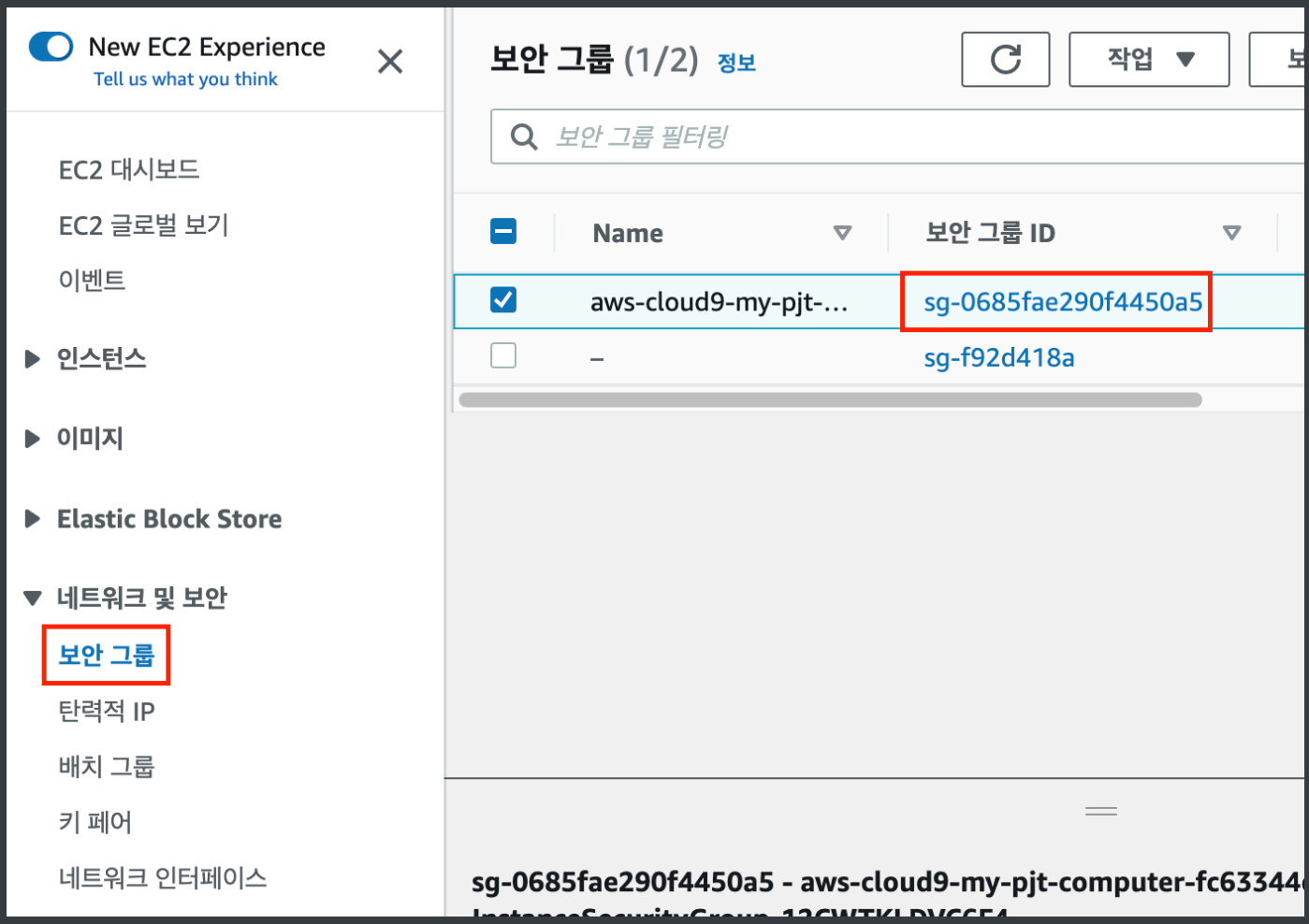
> 새로운 탭에서 진행
>
> EC2는 cloud9 생성시 자동생성

0. 서비스검색



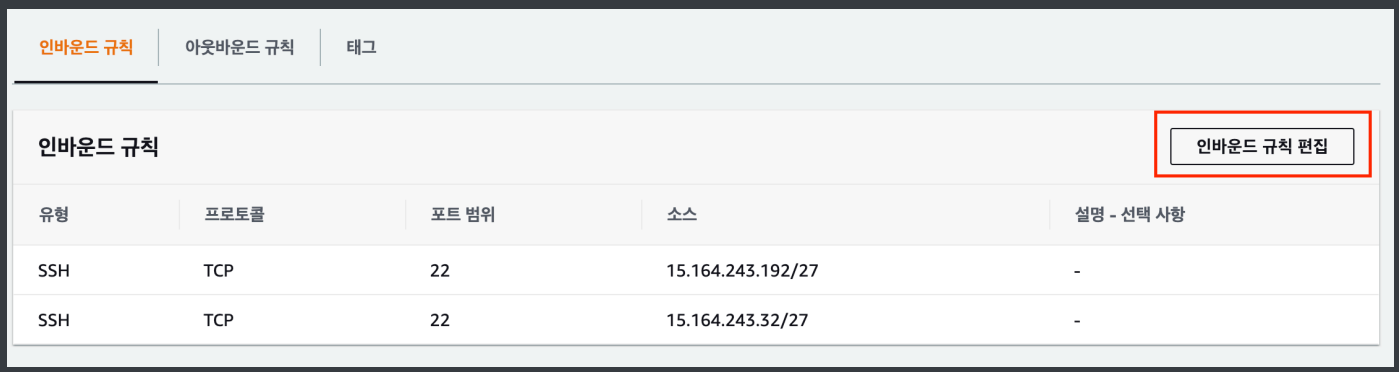
1. 보안그룹

- 생성된 ID 클릭

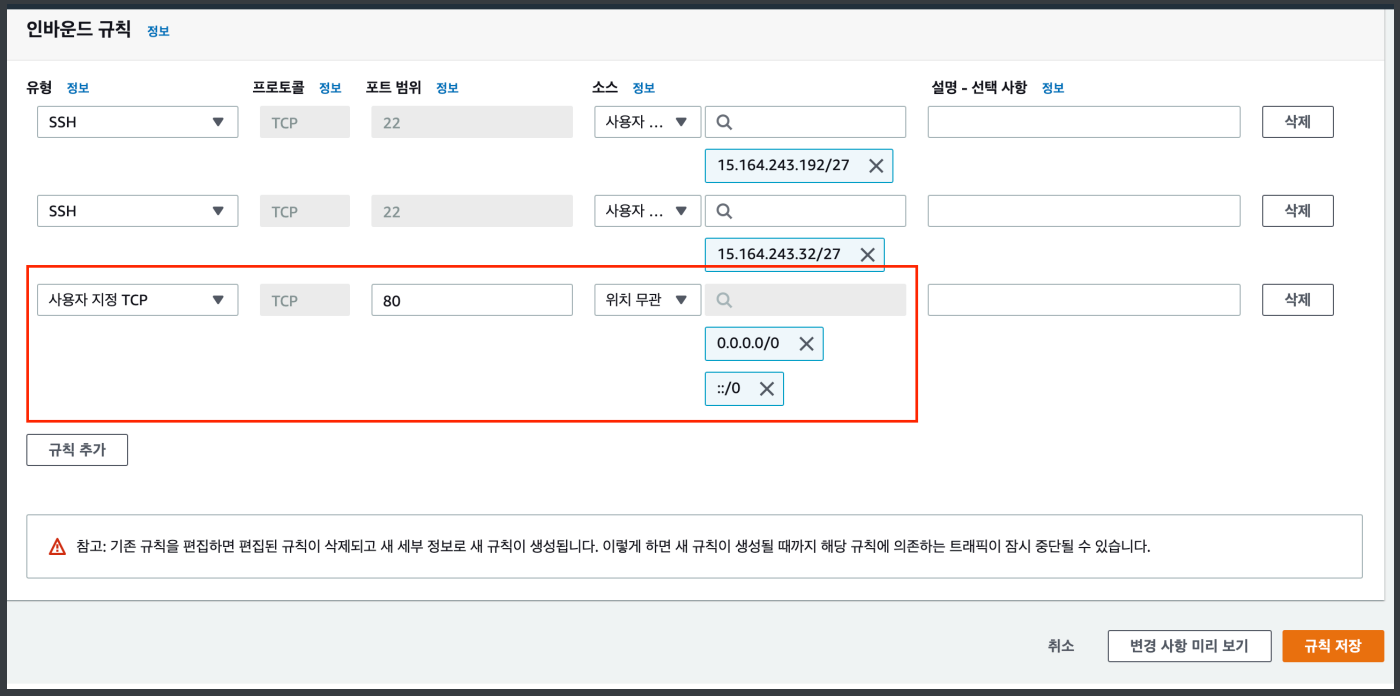

New EC2 Experience
Tell us what you think
EC2 대시보드
EC2 글로벌 보기
이벤트
인스턴스
이미지
Elastic Block Store
네트워크 및 보안
보안 그룹
탄력적 IP
배치 그룹
키 페어
네트워크 인터페이스
보안 그룹 (1/2) 정보
작업
보안 그룹 필터링
Name
보안 그룹 ID
aws-cloud9-my-pjt-...
sg-0685fae290f4450a5
–
sg-f92d418a
sg-0685fae290f4450a5 - aws-cloud9-my-pjt-computer-fc63344

2. 인바운드 설정

- 편집


인바운드 규칙
아웃바운드 규칙
태그
인바운드 규칙
인바운드 규칙 편집
유형
프로토콜
포트 범위
소스
설명 - 선택 사항
SSH
TCP
22
15.164.243.192/27
-
SSH
TCP
22
15.164.243.32/27
-

- 규칙 추가 후 저장



python

0. pyenv

- 설치 & 설정
  - 전체 복사 후 터미널에서 실행

```
git clone https://github.com/pyenv/pyenv.git ~/.pyenv

echo 'export PYENV_ROOT="$HOME/.pyenv"' >> ~/.bashrc
echo 'export PATH="$PYENV_ROOT/bin:$PATH"' >> ~/.bashrc
echo -e 'if command -v pyenv 1>/dev/null 2>&1; then\n  eval "$(pyenv
init -)"\nfi' >> ~/.bashrc

source ~/.bashrc
```

1. python 설치&전역등록

> 프로젝트 진행한 버전에 맞게 설치

```
pyenv install 3.11.4
pyenv global 3.11.4
python -V
#=> Python 3.11.4
```

- module not found

```
sudo apt-get install liblzma-dev
sudo apt-get install libbz2-dev
```

project clone

> 루트폴더와 프로젝트, 두개의 폴더 이름에 주의하며 진행해주세요.
> 두 폴더의 이름을 통일하면 조금더 편하게 설정할 수 있습니다.

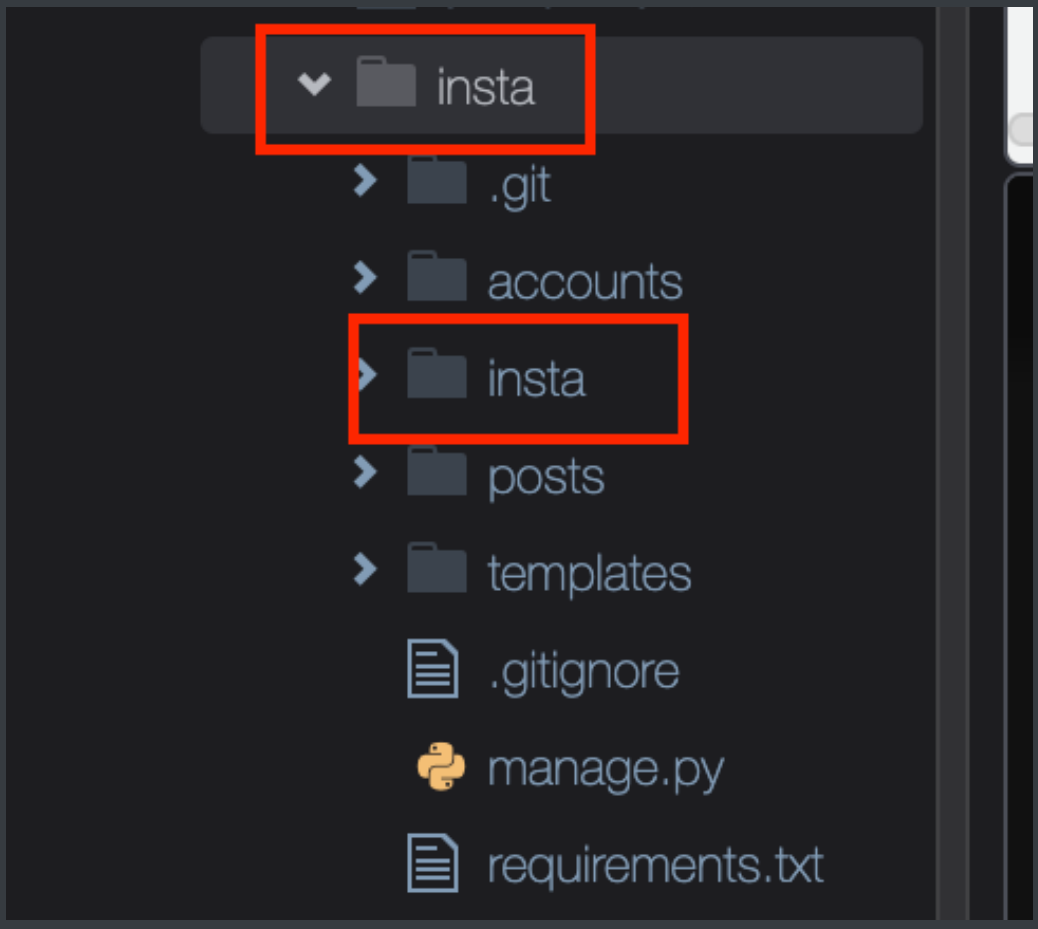
0. 준비

- `~` 로 이동 => `cd ~`
  - 명령어를 작성하는 위치 주의!

- clone
  - need github auth token

```
git clone {project_remote_url}
```

- 편의를 위해 폴더명 변경
  - 프로젝트 전체 폴더의 이름을 프로젝트이름과 동일하게 변경



- 폴더구조
  - 프로젝트이름은 변수처럼 사용예정 이름 기억!

```
home
  ubuntu
    {프로젝트이름}
      {프로젝트이름}
      {앱}
      manage.py
```

- 클론한 폴더로 이동

```
cd {프로젝트이름}
```

- 가상환경

```
python -m venv venv
source venv/bin/activate
```

- 라이브러리 설치

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

- 마이그레이션

```
python manage.py migrate
```

- createsuperuser

```
python manage.py createsuperuser
```

nginx

0. 설치

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

1. 설정

> vi를 사용하여 터미널에서 파일을 수정합니다.
> 사용법을 숙지하고 진행해주세요.

- 복사할 코드 미리 작성하기
  - 아래의 코드에서 각자의 프로젝트이름에 맞게 수정 후 아래에 붙여넣기
  - staticfiles의 경우 다른 폴더를 썼다면 이름수정

```
    server_name *.compute.amazonaws.com;

    location / {
        uwsgi_pass unix:///home/ubuntu/{프로젝트이름}/tmp/{프로젝트이름}.sock;
        include uwsgi_params;
    }

    location /static/ {
      alias /home/ubuntu/{프로젝트이름}/staticfiles/;
    }
```

- 결과 붙여넣기 위한 빈칸

```

```

- 파일 수정

```
sudo vi /etc/nginx/sites-enabled/default
```

- 아래의 표시된 부분 수정
  - `i` 버튼으로 수정모드로 전환
  - 아래의 부분으로 방향키를 이용하여 이동
  - 수정
  - `esc` 로 수정모드 빠져나오기
  - `:wq` 명령어로 저장 후 종료

```
#
server {
        listen 80 default_server;
        listen [::]:80 default_server;

        # SSL configuration
        #
        # listen 443 ssl default_server;
        # listen [::]:443 ssl default_server;
        #
        # Note: You should disable gzip for SSL traffic.
        # See: https://bugs.debian.org/773332
        #
        # Read up on ssl_ciphers to ensure a secure configuration.
        # See: https://bugs.debian.org/765782
        #
        # Self signed certs generated by the ssl-cert package
        # Don't use them in a production server!
        #
        # include snippets/snakeoil.conf;

        root /var/www/html;

        # Add index.php to the list if you are using PHP
        index index.html index.htm index.nginx-debian.html;

        server_name _;

        location / {
                # First attempt to serve request as file, then
                # as directory, then fall back to displaying a 404.
                try_files $uri $uri/ =404;
        }

        # pass PHP scripts to FastCGI server
        #
```

uWSGI

0. 설치

```
pip install uwsgi
```

1. 폴더&파일 생성

- 프로젝트 폴더 이동 (기존의 위치와 동일)

```
cd ~/{프로젝트이름}
```

- uwsgi 설정, 로그 저장할 폴더 생성 (파일트리에서 생성해도 무방)

```
mkdir tmp
mkdir -p log/uwsgi
mkdir -p .config/uwsgi/
```

- uwsgi 설정파일 생성 (파일트리에서 생성해도 무방)

```
touch .config/uwsgi/{프로젝트이름}.ini
```

3. 수정

- `.config/uwsgi/{프로젝트이름}.ini` 설정파일 수정

```
# {프로젝트이름}/.config/uwsgi/{프로젝트이름}.ini

[uwsgi]
chdir = /home/ubuntu/{프로젝트이름}
module = {프로젝트이름}.wsgi:application
home = /home/ubuntu/{프로젝트이름}/venv
```

```
uid = ubuntu
gid = ubuntu

socket = /home/ubuntu/{프로젝트이름}/tmp/{프로젝트이름}.sock
chmod-socket = 666
chown-socket = ubuntu:ubuntu

enable-threads = true
master = true
vacuum = true
pidfile = /home/ubuntu/{프로젝트이름}/tmp/{프로젝트이름}.pid
logto = /home/ubuntu/{프로젝트이름}/log/uwsgi/@(exec://date +%%Y-%%m-
%%d).log
log-reopen = true
```

4. daemon

- 설정파일 생성 (파일트리에서 생성해도 무방)

```
touch .config/uwsgi/uwsgi.service
```

- `.config/uwsgi/uwsgi.service` 설정파일 수정

```
[Unit]
Description=uWSGI Service
After=syslog.target

[Service]
User=ubuntu
```

```
ExecStart=/home/ubuntu/{프로젝트이름}/venv/bin/uwsgi -i /home/ubuntu/{프로젝트이름}/.config/uwsgi/insta.ini


Restart=always
KillSignal=SIGQUIT
Type=notify
StandardError=syslog
NotifyAccess=all

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

- 심볼릭링크 생성

```
sudo ln -s ~/{프로젝트이름}/.config/uwsgi/uwsgi.service /etc/systemd/system/uwsgi.service
```

- 등록

```
# daemon reload
sudo systemctl daemon-reload

# uswgi daemon enable and restart
sudo systemctl enable uwsgi
sudo systemctl restart uwsgi.service

# check daemon
sudo systemctl | grep nginx
```

```
sudo systemctl | grep uwsgi

# nginx restart
sudo systemctl restart nginx
sudo systemctl restart uwsgi
```

- 아래의 에러 상황에서 80번 포트 프로세스 종료

```
(venv) ubuntu:~/insta (master) $ sudo 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 Sun 2023-07-09 15:00:35 UTC; 16s ago
     Docs: man:nginx(8)
  Process: 11265 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=1/FAILURE)
  Process: 11254 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUC

Jul 09 15:00:34 ip-172-31-86-28 nginx[11265]: nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address already i
Jul 09 15:00:34 ip-172-31-86-28 nginx[11265]: nginx: [emerg] bind() to [::]:80 failed (98: Address already in u
Jul 09 15:00:34 ip-172-31-86-28 nginx[11265]: nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address already i
Jul 09 15:00:34 ip-172-31-86-28 nginx[11265]: nginx: [emerg] bind() to [::]:80 failed (98: Address already in u
Jul 09 15:00:35 ip-172-31-86-28 nginx[11265]: nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address already i
Jul 09 15:00:35 ip-172-31-86-28 nginx[11265]: nginx: [emerg] bind() to [::]:80 failed (98: Address already in u
Jul 09 15:00:35 ip-172-31-86-28 nginx[11265]: nginx: [emerg] still could not bind()
Jul 09 15:00:35 ip-172-31-86-28 systemd[1]: nginx.service: Control process exited, code=exited status=1
Jul 09 15:00:35 ip-172-31-86-28 systemd[1]: nginx.service: Failed with result 'exit-code'.
Jul 09 15:00:35 ip-172-31-86-28 systemd[1]: Failed to start A high performance web server and a reverse proxy s
```

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

- 최종확인
  - EC2대시보드에서 DNS혹은 IP확인