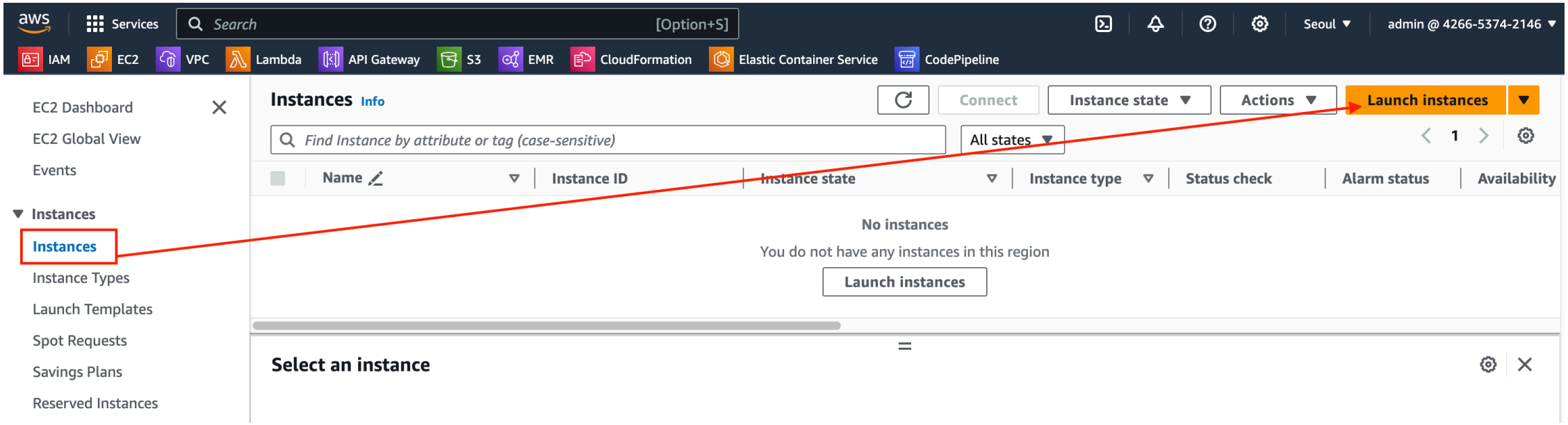
aws ec2 생성

단계1: Launch instances



단계2: Instances name

단계3: Application and OS Images

Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Recents | My AMIs | Quick Start

Amazon Linux

macOS

Ubuntu

Windows

Red Hat

SUSE Li

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type — Free tier eligible

ami-062cf18d655c0b1e8 (64-bit (x86)) / ami-09cb0f54fe24c54a6 (64-bit (Arm))

Virtualization: hvm ENA enabled: true Root device type: ebs

Description

Ubuntu Server 24.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).

Architecture

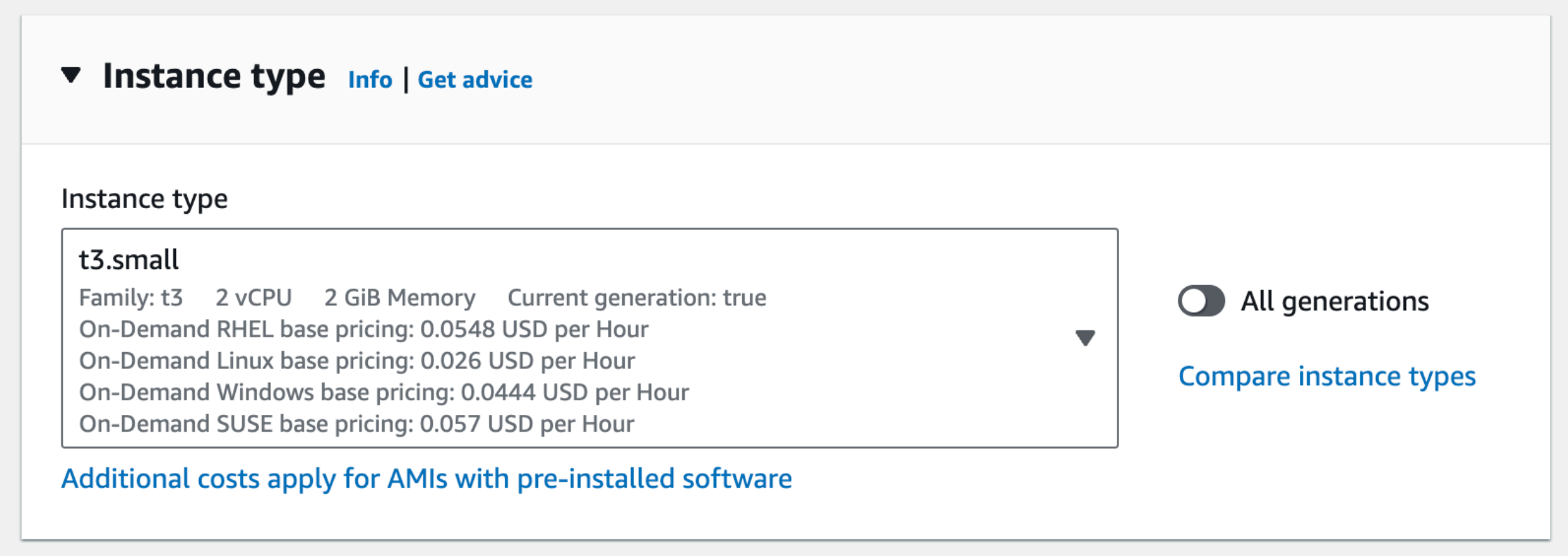
64-bit (x86)

AMI ID

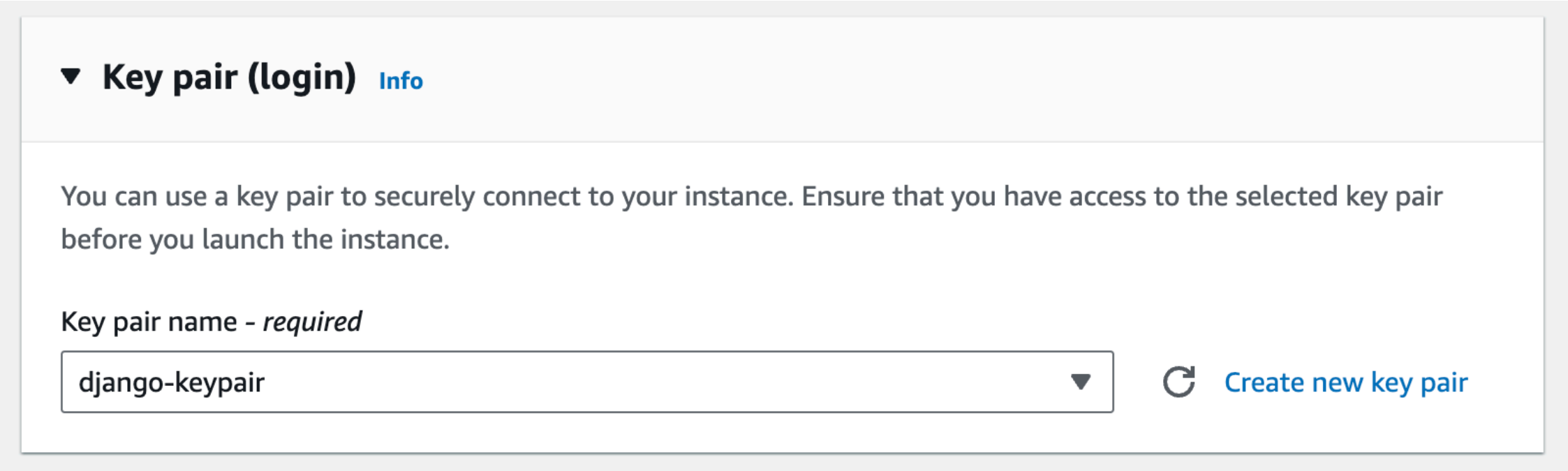
ami-062cf18d655c0b1e8

Verified provider

단계4: Instance type



단계5: Key pair


Key pair (login)
Info
You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.
Key pair name - required
django-keypair
Create new key pair

단계6: Network settings

Network settings Info

VPC - *required* Info

vpc-0e092393ffbd671b9 (default)
172.31.0.0/16

Subnet Info

No preference

Create new subnet

Auto-assign public IP Info

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

- (●) Create security group
- ( ) Select existing security group

Security group name - *required*

```
django-sg
```

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and ._-:/()#,@[]+=&;{}!$*

Description - *required* Info

```
django-sg created 2024-07-26T23:10:29.398Z
```

- ssh protocol

Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

(●) Create security group

( ) Select existing security group

Security group name - *required*

```
django-sg
```

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and ._-:/()#,@[]+=&;{}!$*

Description - *required* Info

```
django-sg created 2024-07-26T23:10:29.398Z
```

- nginx server, django server

Security group rule 2 (TCP, 80, 0.0.0.0/0, nginx server, django server)

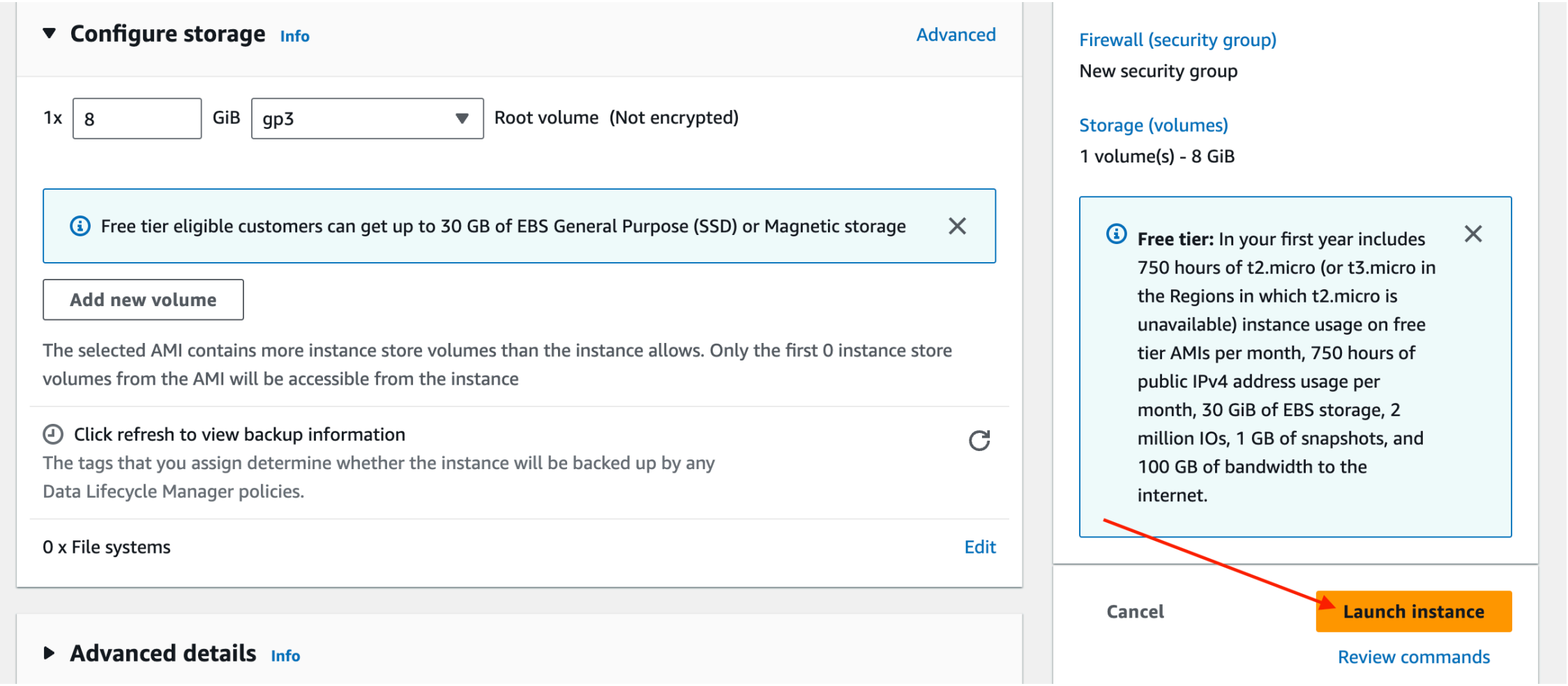
Remove

Type Info	Protocol Info	Port range Info
HTTP	TCP	80

Source type Info	Source Info	Description - *optional* Info
Anywhere	*Add CIDR, prefix list or security*	nginx server, django server
	0.0.0.0/0	

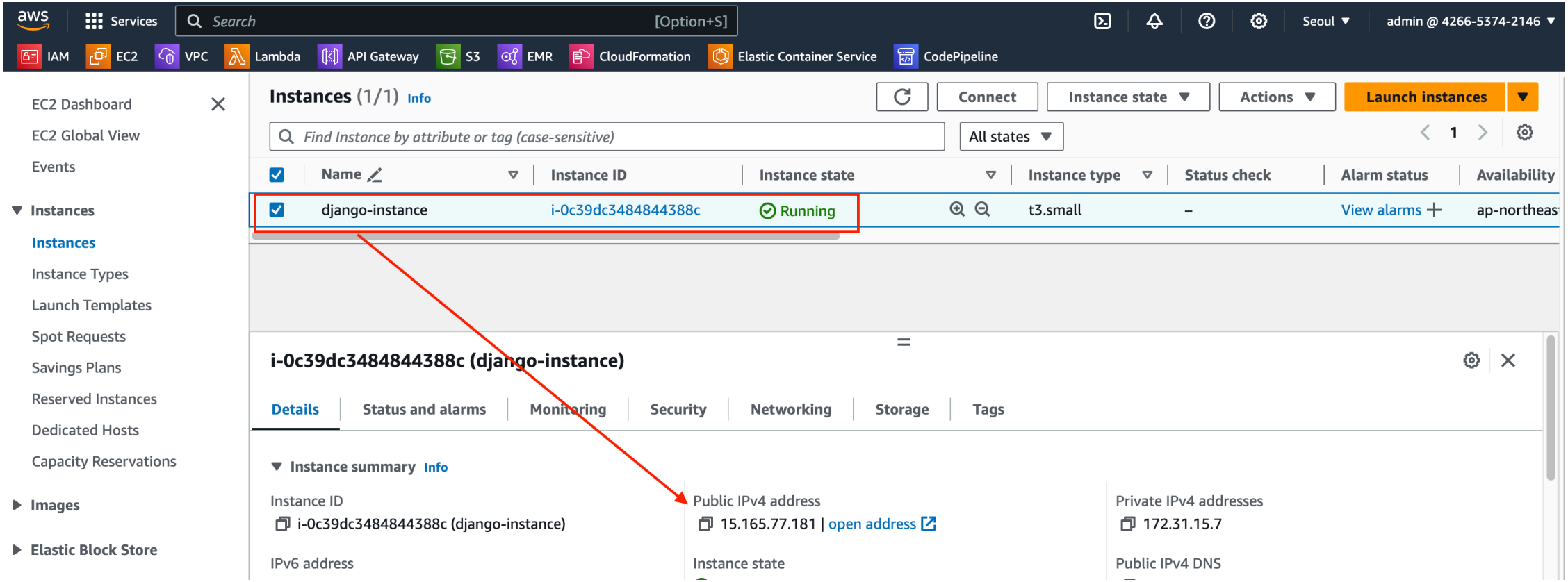
Add security group rule

단계7: configure storage & Launch instance



ec2 접속

단계1: Public IPv4 address



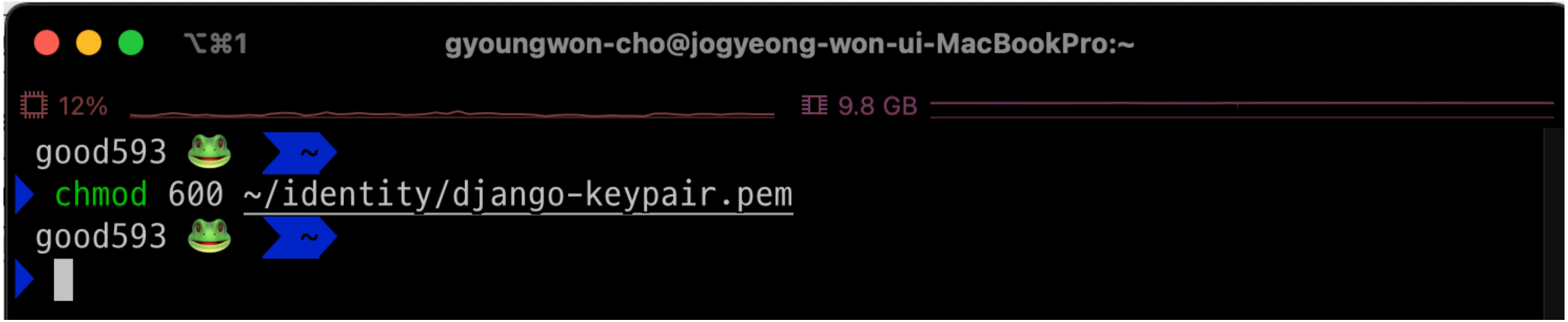
단계2: keypair 이동

```
⌥⌘1          gyoungwon-cho@jogyeong-won-ui-MacBookPro:~
13%                                          10.0 GB
good593 🐸  ~
 mv ~/Downloads/django-keypair.pem ~/identity
good593 🐸  ~

good593 🐸  ~
 ls ~/identity
django-keypair.pem
good593 🐸  ~
```

단계3: (윈도우 생략) 권한 변경

```
# pem 키 권한 변경 -> 600이 아닐 경우 보안 취약으로 판단
chmod 600 ~/identity/django-keypair.pem
```



단계4: config 수정

```
vim ~/.ssh/config
```

단계5: ec2 접속 by ssh

```
⌥⌘1                    ubuntu@ip-172-31-15-7: ~
10%                                          9.5 GB
clear
good593 🐸  ~
ssh django-instance
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1009-aws x86_64)

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

 System information as of Fri Jul 26 23:45:18 UTC 2024
```

docker in ec2 설치

단계1: 우분투 시스템 패키지 업데이트

```
sudo apt-get update
```

```
⌥⌘1                    ubuntu@ip-172-31-15-7: ~
10%                                              9.5 GB
ubuntu@ip-172-31-15-7:~$ sudo apt-get update
Hit:1 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates InReleas
e [126 kB]
Get:3 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports InRele
ase [126 kB]
Get:4 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 P
ackages [15.0 MB]
```

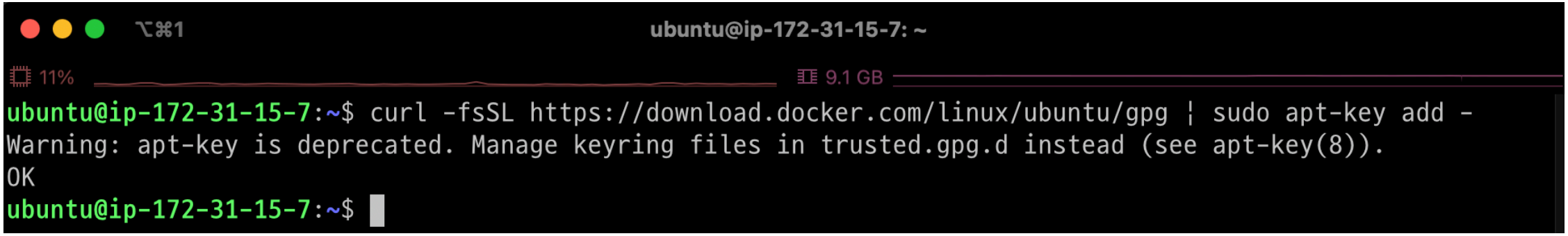
단계2: 우분투 시스템 패키지 업데이트

```
sudo apt-get install apt-transport-https ca-certificates curl gnupg-agent software-properties-common -y
```

```
12%                                                                    9.6 GB
ubuntu@ip-172-31-15-7:~$ sudo apt-get install apt-transport-https ca-certificate
s curl gnupg-agent software-properties-common -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
apt-transport-https is already the newest version (2.7.14build2).
ca-certificates is already the newest version (20240203).
curl is already the newest version (8.5.0-2ubuntu10.1).
gnupg-agent is already the newest version (2.4.4-2ubuntu17).
software-properties-common is already the newest version (0.99.48).
0 upgraded, 0 newly installed, 0 to remove and 26 not upgraded.
ubuntu@ip-172-31-15-7:~$
```

단계3: Docker의 공식 GPG키를 추가

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```



단계4: Docker의 공식 apt 저장소를 추가

```
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
```

```
⌥⌘1                                ubuntu@ip-172-31-15-7: ~
12%                                                          9.2 GB
ubuntu@ip-172-31-15-7:~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubunt
u $(lsb_release -cs) stable"
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubunt
```

단계5: 시스템 패키지 업데이트

```
sudo apt-get update
```

```
12%                                                                    9.1 GB
ubuntu@ip-172-31-15-7:~$ sudo apt-get update
Hit:1 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg key
ring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
ubuntu@ip-172-31-15-7:~$
```

단계6: Docker 설치

```
sudo apt-get install docker-ce docker-ce-cli containerd.io -y
```

```
ubuntu@ip-172-31-15-7: ~
22%                                                    9.1 GB
ubuntu@ip-172-31-15-7:~$ sudo apt-get install docker-ce docker-ce-cli containerd.io -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
docker-ce is already the newest version (5:27.1.1-1~ubuntu.24.04~noble).
docker-ce-cli is already the newest version (5:27.1.1-1~ubuntu.24.04~noble).
containerd.io is already the newest version (1.7.19-1).
0 upgraded, 0 newly installed, 0 to remove and 26 not upgraded.
ubuntu@ip-172-31-15-7:~$
```

단계7: Docker 설치 확인

```
sudo systemctl status docker
```

```
ubuntu@ip-172-31-15-7: ~
12%                                                    9.1 GB
ubuntu@ip-172-31-15-7:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: enabled)
     Active: active (running) since Sat 2024-07-27 00:07:34 UTC; 1min 8s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 3610 (dockerd)
      Tasks: 9
     Memory: 20.8M (peak: 21.5M)
        CPU: 518ms
     CGroup: /system.slice/docker.service
             └─3610 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Jul 27 00:07:33 ip-172-31-15-7 systemd[1]: Starting docker.service - Docker Application Container Engine.>
Jul 27 00:07:33 ip-172-31-15-7 dockerd[3610]: time="2024-07-27T00:07:33.855390893Z" level=info msg="Start>
```

단계8: 권한 변경

```
# 파일의 권한을 666으로 변경하여 그룹 내 다른 사용자도 접근 가능하게 변경
sudo chmod 666 /var/run/docker.sock
docker ps
```

```
⌥⌘1                                ubuntu@ip-172-31-15-7: ~
10%                                                        9.2 GB
ubuntu@ip-172-31-15-7:~$ sudo chmod 666 /var/run/docker.sock
ubuntu@ip-172-31-15-7:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
ubuntu@ip-172-31-15-7:~$
```

Django

단계1: search django image

```
docker search goodwon593
```

```
13%                                                                    9.1 GB
ubuntu@ip-172-31-15-7:~$ docker search goodwon593
NAME                                      DESCRIPTION   STARS     OFFICIAL
goodwon593/kubeflow_jupyter                             0
goodwon593/kubeflow_jupyter_tensorflow                  0
goodwon593/kubeflow_codeserver                          0
goodwon593/kubeflow_transformer                         0
goodwon593/helloapache                                  0
goodwon593/kubeflow_tensorflow                          0
goodwon593/hellojs-image                                0
goodwon593/django-image                                 0
ubuntu@ip-172-31-15-7:~$
```

단계2: pull django image

```
docker pull goodwon593/django-image
```

```
⌥⌘1                                ubuntu@ip-172-31-15-7: ~
11%                                                    9.1 GB
ubuntu@ip-172-31-15-7:~$ docker pull goodwon593/django-image
Using default tag: latest
latest: Pulling from goodwon593/django-image
262a5f25eec7: Pull complete
fe1e9dc73ada: Pull complete
4499f9c982c7: Pull complete
3ccf698a8df7: Pull complete
```

단계3: django image 확인

```
docker images
```

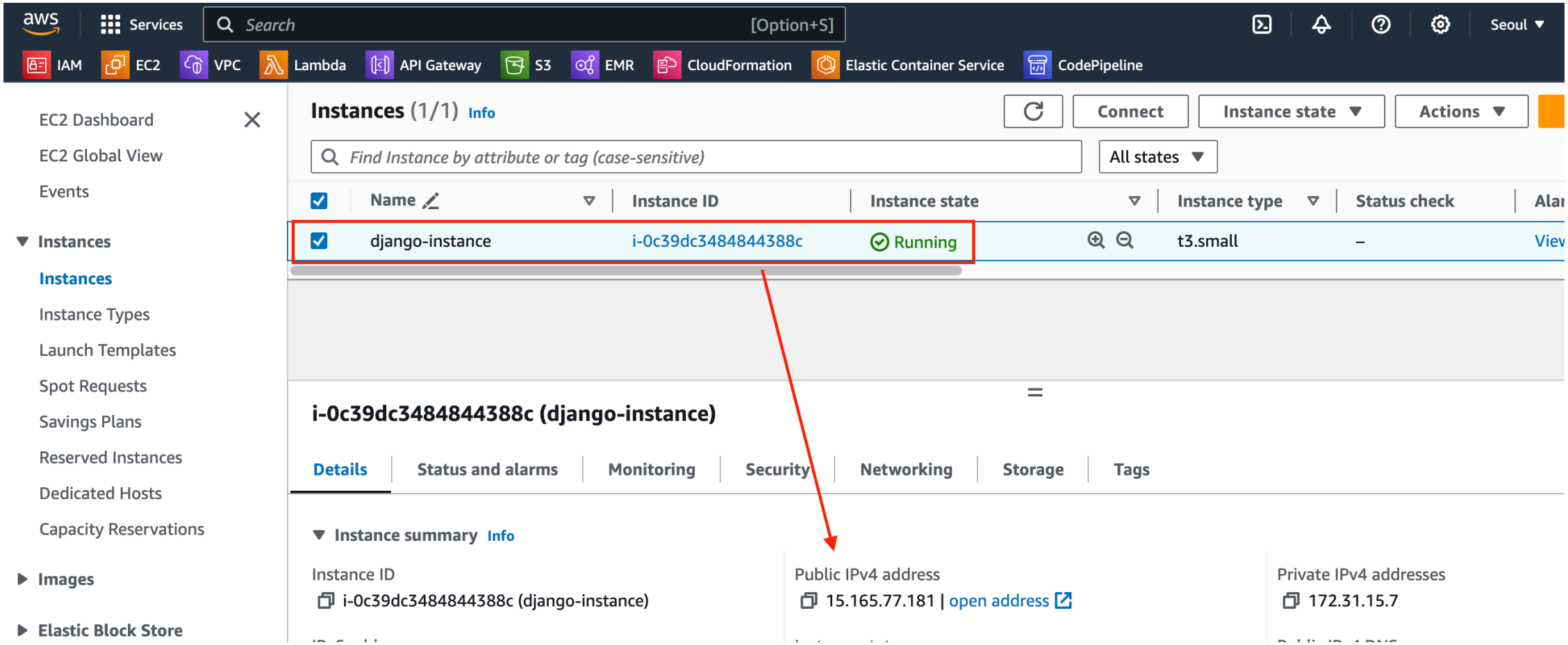
```
ubuntu@ip-172-31-15-7: ~
13%                                                    9.1 GB
ubuntu@ip-172-31-15-7:~$ docker images
REPOSITORY                TAG       IMAGE ID       CREATED          SIZE
goodwon593/django-image   latest    01fab73f3a86   37 hours ago     234MB
hello-world               latest    d2c94e258dcb   15 months ago    13.3kB
ubuntu@ip-172-31-15-7:~$
```

단계4: Create container

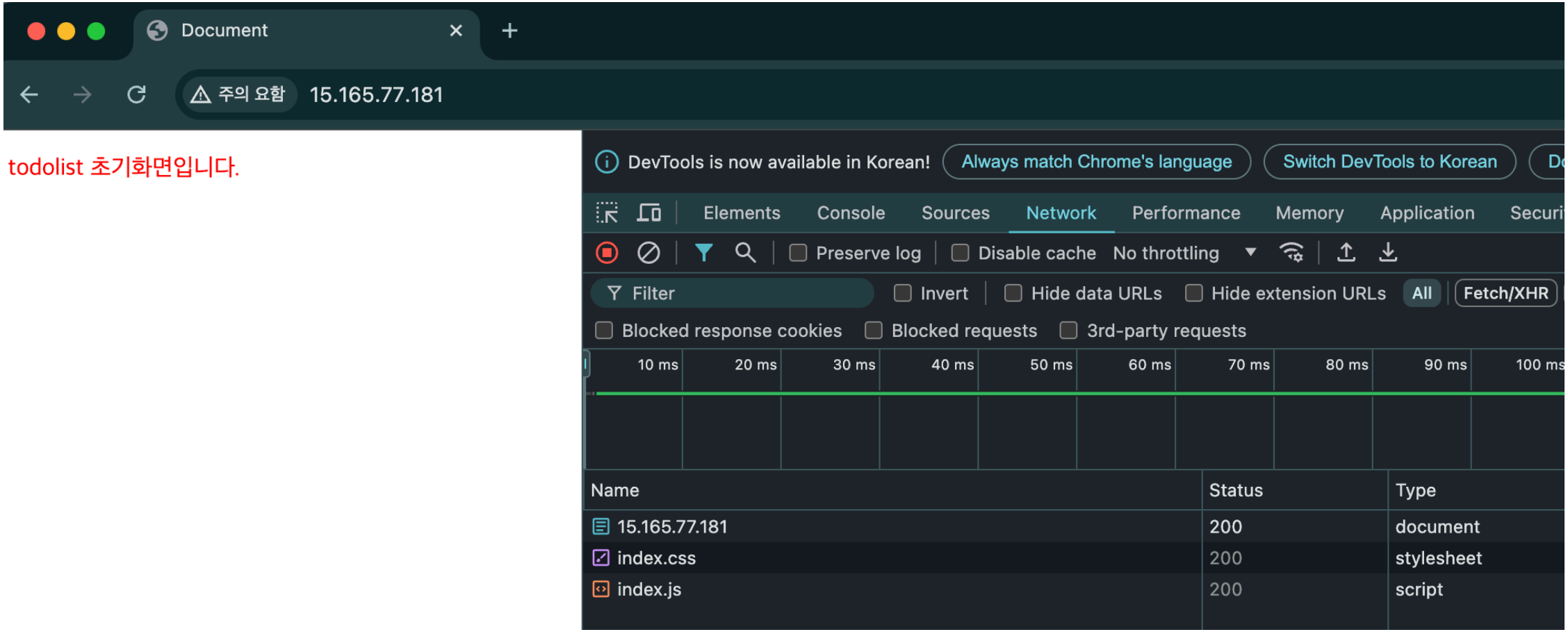
```
# Create container
docker run --name django-container -d -p 80:80 goodwon593/django-image
# 결과 확인
docker ps
```

```
ubuntu@ip-172-31-15-7: ~
10%                                                    8.6 GB
ubuntu@ip-172-31-15-7:~$ docker run --name django-container -d -p 80:80 goodwon593/django-image
0d905743405e4c7fcba2493fcb399cddb654521e03eac5d6d08481fb3e2b2351
ubuntu@ip-172-31-15-7:~$ docker ps
CONTAINER ID   IMAGE                    COMMAND      CREATED         STATUS         PORTS
               NAMES
0d905743405e   goodwon593/django-image   "./run.sh"   5 seconds ago   Up 4 seconds   0.0.0.0:80->80/tcp, :
::80->80/tcp   django-container
ubuntu@ip-172-31-15-7:~$
```

단계5: Public IPv4 address



단계6: Djagon 접속 on aws ec2



참고 문서

- https://haengsin.tistory.com/128
- https://velog.io/@osk3856/Docker-Ubuntu-22.04-Docker-Installation