Lesson 10 Demo 01

Running a Docker Container on an AWS EC2 Instance

Objective: To demonstrate the process of running a Hello World Docker container on an AWS EC2 instance for efficient container deployment

Tools required: AWS Management Console

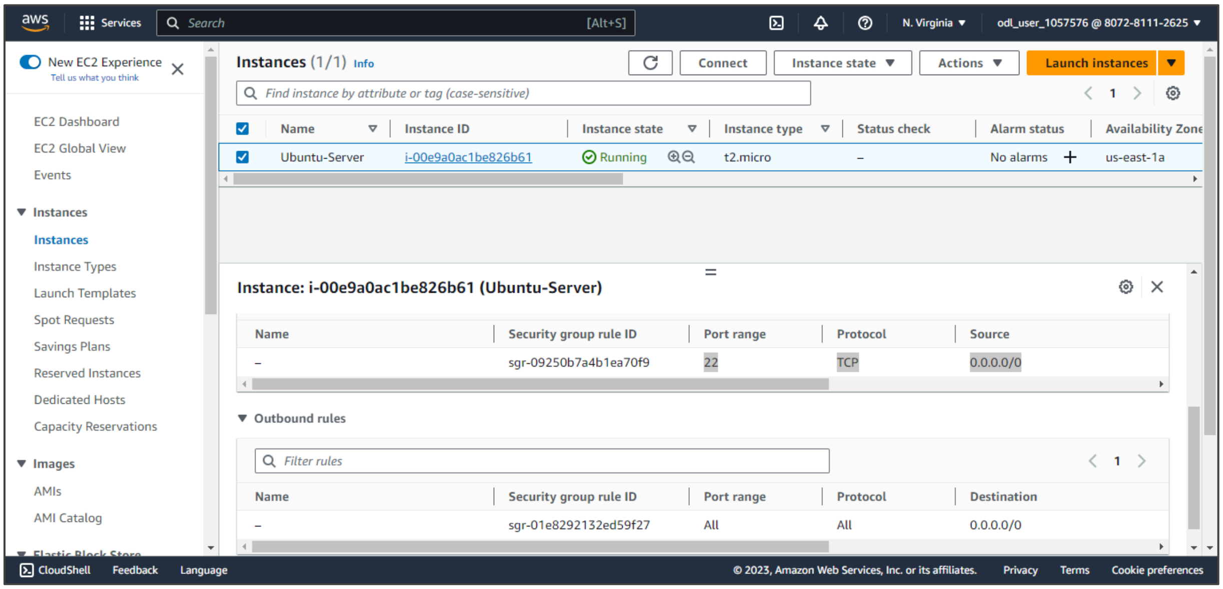
Prerequisites: An EC2 instance with Ubuntu installed

Steps to be followed:

1. Set up an EC2 instance
2. Install the Docker on Ubuntu
3. Run the Hello World container

Step 1: Set up an EC2 instance

1.1 Launch a new EC2 instance with Ubuntu as the operating system



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
Ubuntu-Server	i-00e9a0ac1be826b61	Running	t2.micro	–	No alarms	us-east-1a

Instance: i-00e9a0ac1be826b61 (Ubuntu-Server)

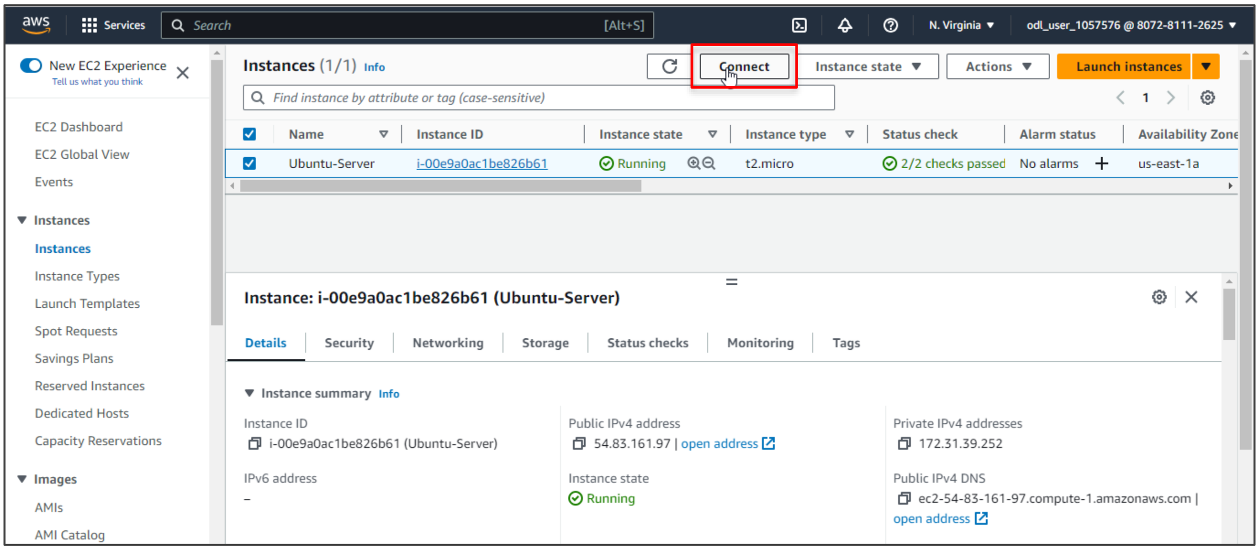
Name	Security group rule ID	Port range	Protocol	Source
–	sgr-09250b7a4b1ea70f9	22	TCP	0.0.0.0/0

Outbound rules

Name	Security group rule ID	Port range	Protocol	Destination
–	sgr-01e8292132ed59f27	All	All	0.0.0.0/0

Note: Ensure that you have the necessary security group rules to allow SSH access. Refer to previous lesson demos on how to launch an EC2 instance.

1.2 Select the instance and click **Connect**



☑	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
☑	Ubuntu-Server	i-00e9a0ac1be826b61	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a

Instance: i-00e9a0ac1be826b61 (Ubuntu-Server)

Instance ID
i-00e9a0ac1be826b61 (Ubuntu-Server)

Public IPv4 address
54.83.161.97 | open address

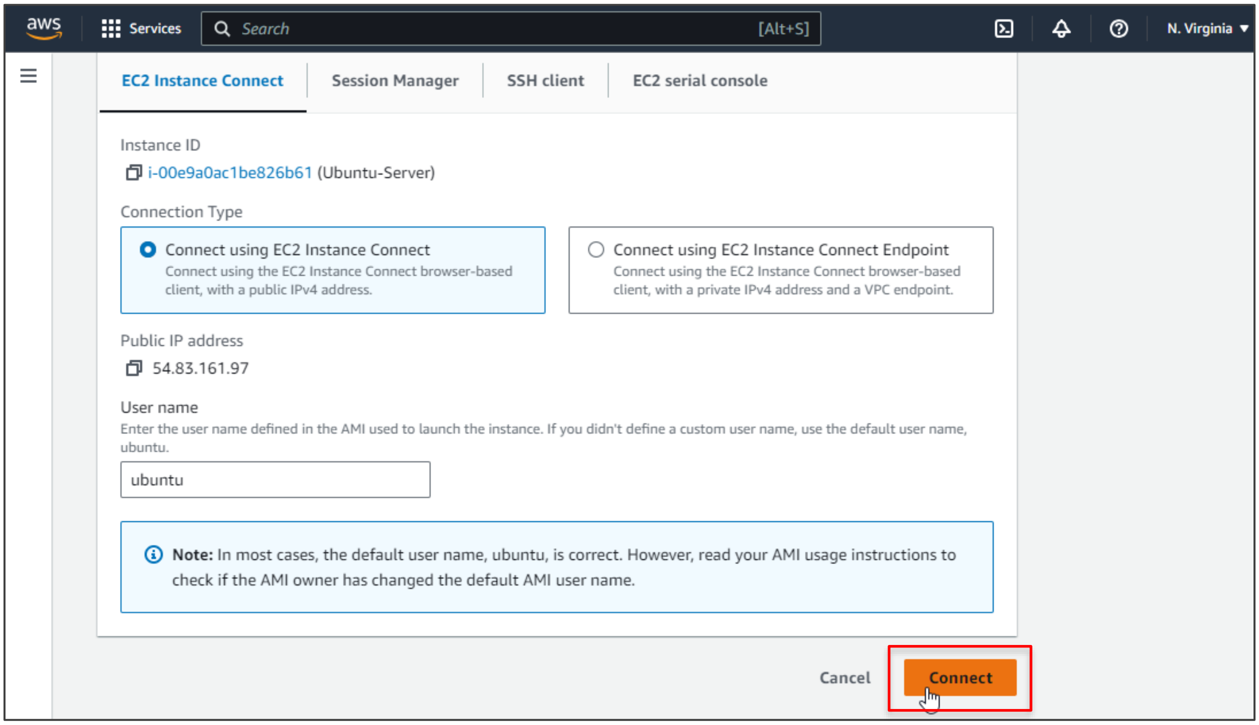
Private IPv4 addresses
172.31.39.252

IPv6 address
–

Instance state
Running

Public IPv4 DNS
ec2-54-83-161-97.compute-1.amazonaws.com | open address

1.3 Click **Connect**



Instance ID
i-00e9a0ac1be826b61 (Ubuntu-Server)

Connection Type

● Connect using EC2 Instance Connect
Connect using the EC2 Instance Connect browser-based client, with a public IPv4 address.

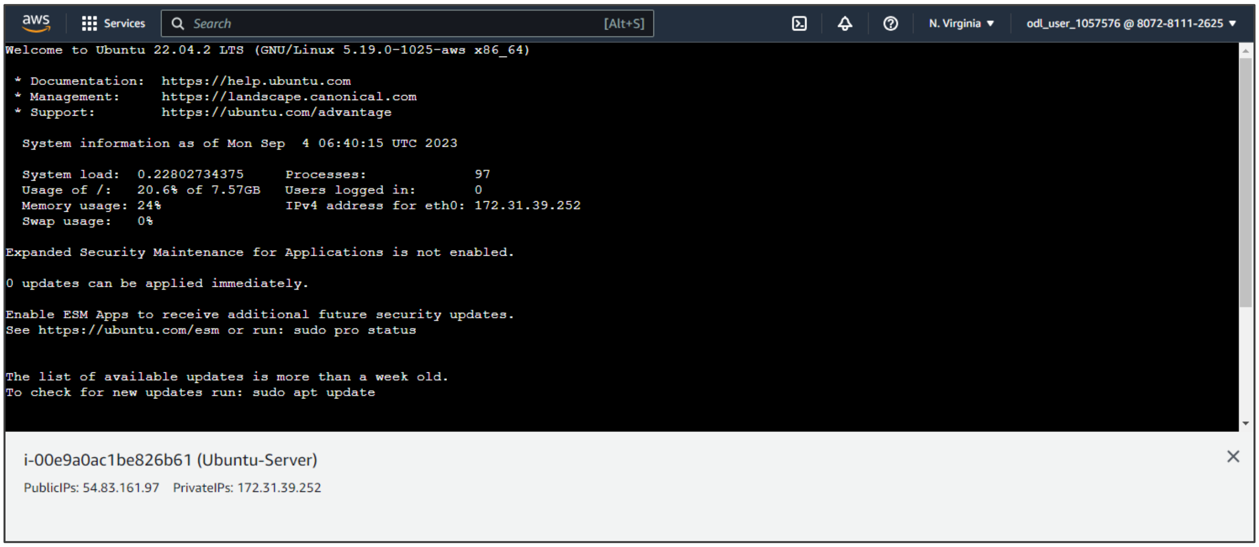
○ Connect using EC2 Instance Connect Endpoint
Connect using the EC2 Instance Connect browser-based client, with a private IPv4 address and a VPC endpoint.

Public IP address
54.83.161.97

User name
Enter the user name defined in the AMI used to launch the instance. If you didn't define a custom user name, use the default user name, ubuntu.

ubuntu

Note: In most cases, the default user name, ubuntu, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.



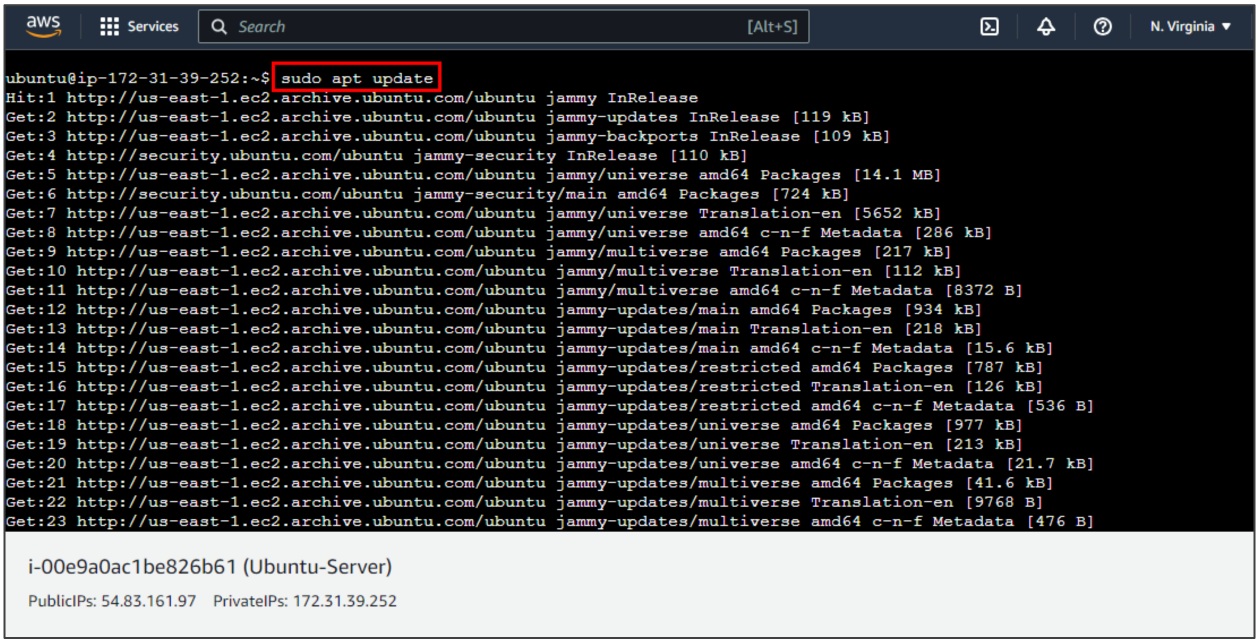
```
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.19.0-1025-aws x86_64)

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

  System information as of Mon Sep  4 06:40:15 UTC 2023

  System load:  0.22802734375     Processes:             97
  Usage of /:   20.6% of 7.57GB   Users logged in:       0
  Memory usage: 24%               IPv4 address for eth0: 172.31.39.252
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
```

i-00e9a0ac1be826b61 (Ubuntu-Server)

PublicIPs: 54.83.161.97   PrivateIPs: 172.31.39.252

Step 2: Install the Docker on Ubuntu

2.1 Run the following command to update the package list:
sudo apt update



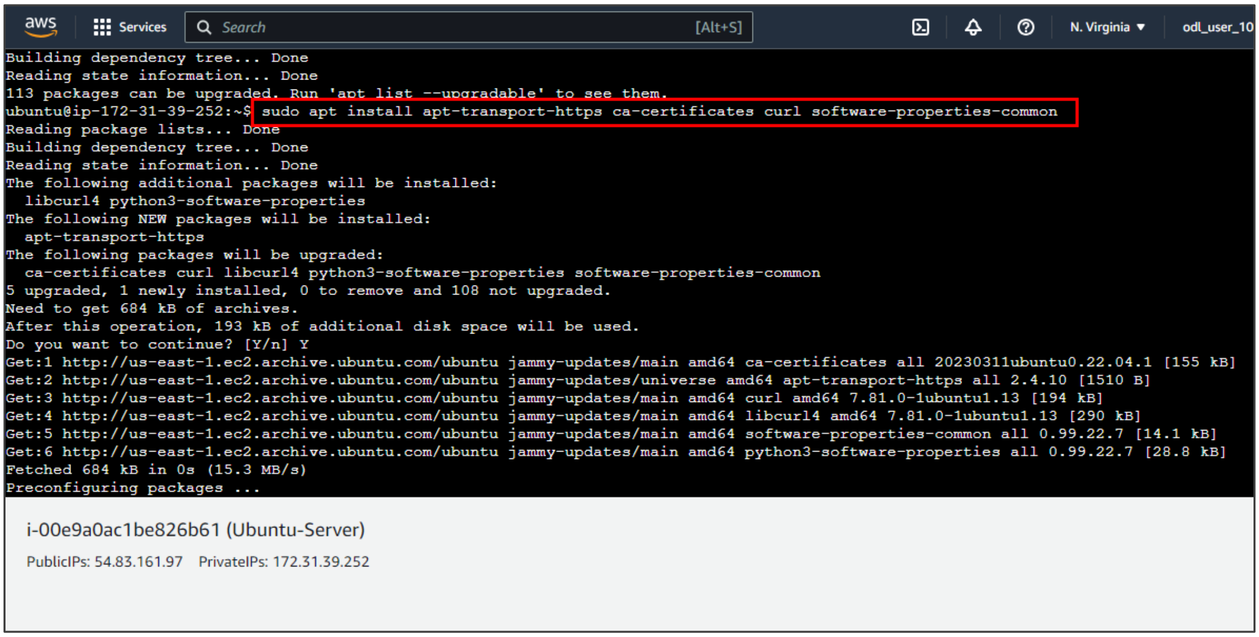
```
ubuntu@ip-172-31-39-252:~$ sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
Get:6 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [724 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-en [5652 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 c-n-f Metadata [286 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [217 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse Translation-en [112 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 c-n-f Metadata [8372 B]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [934 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [218 kB]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [15.6 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [787 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted Translation-en [126 kB]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 c-n-f Metadata [536 B]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [977 kB]
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe Translation-en [213 kB]
Get:20 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 c-n-f Metadata [21.7 kB]
Get:21 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [41.6 kB]
Get:22 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse Translation-en [9768 B]
Get:23 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 c-n-f Metadata [476 B]
```

i-00e9a0ac1be826b61 (Ubuntu-Server)

PublicIPs: 54.83.161.97   PrivateIPs: 172.31.39.252

2.2 Run the following command to install the necessary dependencies:

sudo apt install apt-transport-https ca-certificates curl software-properties-common



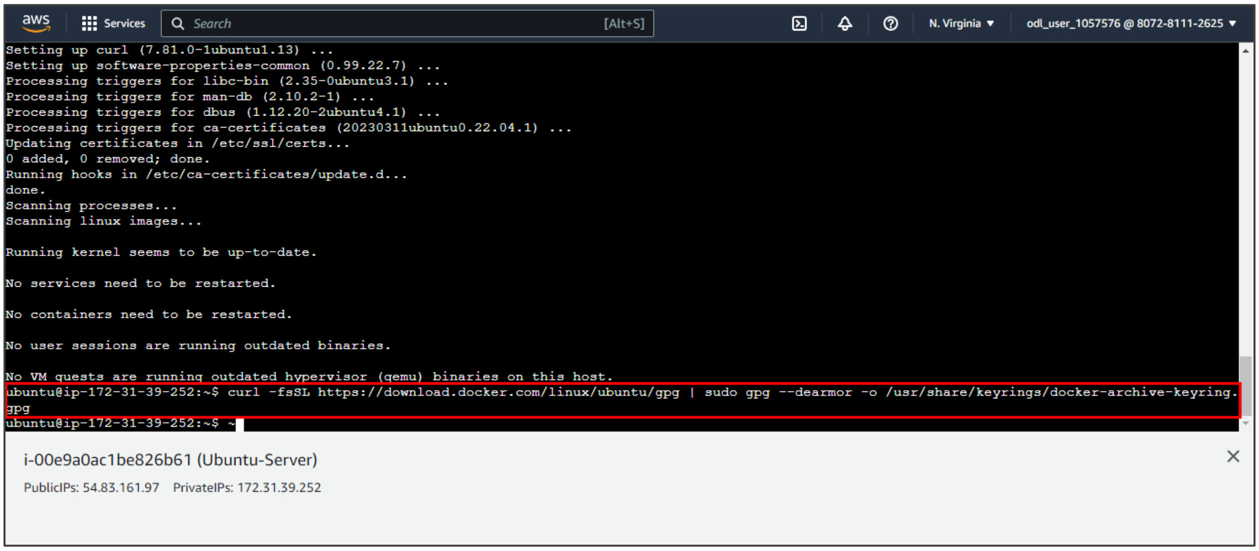
```
Building dependency tree... Done
Reading state information... Done
113 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-172-31-39-252:~$ sudo apt install apt-transport-https ca-certificates curl software-properties-common
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libcurl4 python3-software-properties
The following NEW packages will be installed:
  apt-transport-https
The following packages will be upgraded:
  ca-certificates curl libcurl4 python3-software-properties software-properties-common
5 upgraded, 1 newly installed, 0 to remove and 108 not upgraded.
Need to get 684 kB of archives.
After this operation, 193 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 ca-certificates all 20230311ubuntu0.22.04.1 [155 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 apt-transport-https all 2.4.10 [1510 B]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 curl amd64 7.81.0-1ubuntu1.13 [194 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libcurl4 amd64 7.81.0-1ubuntu1.13 [290 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 software-properties-common all 0.99.22.7 [14.1 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 python3-software-properties all 0.99.22.7 [28.8 kB]
Fetched 684 kB in 0s (15.3 MB/s)
Preconfiguring packages ...
```

i-00e9a0ac1be826b61 (Ubuntu-Server)

PublicIPs: 54.83.161.97   PrivateIPs: 172.31.39.252

2.3 Run the following command to add the Docker GPG key:

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg



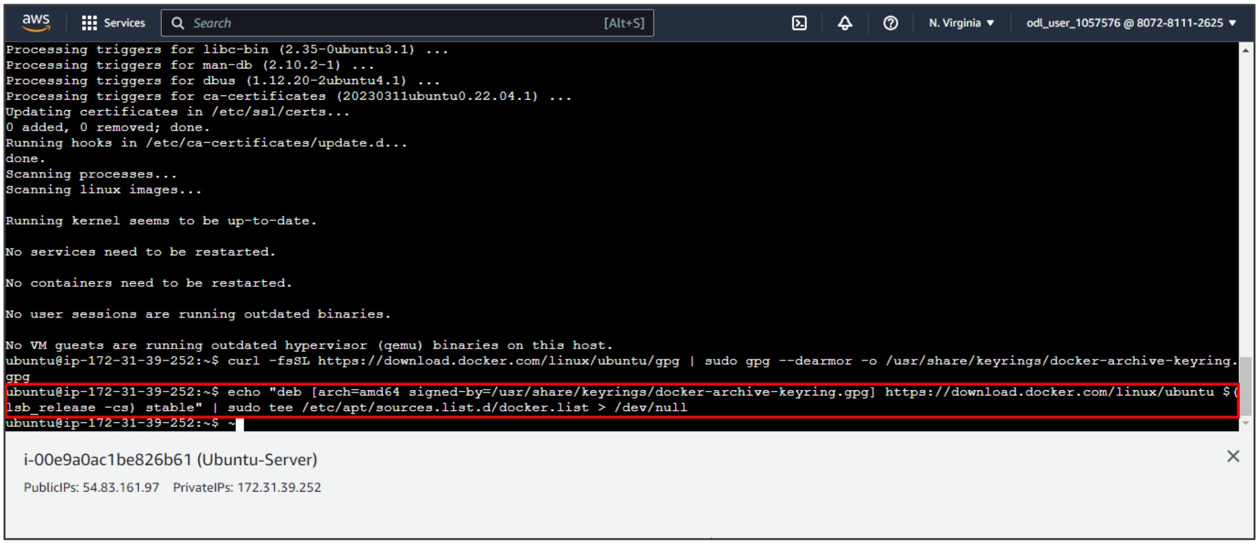
```
Setting up curl (7.81.0-1ubuntu1.13) ...
Setting up software-properties-common (0.99.22.7) ...
Processing triggers for libc-bin (2.35-0ubuntu3.1) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for dbus (1.12.20-2ubuntu4.1) ...
Processing triggers for ca-certificates (20230311ubuntu0.22.04.1) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-39-252:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.
gpg
ubuntu@ip-172-31-39-252:~$ ~
```

i-00e9a0ac1be826b61 (Ubuntu-Server)

PublicIPs: 54.83.161.97   PrivateIPs: 172.31.39.252

2.4 Run the following command to add the Docker repository:

echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null



```
Processing triggers for libc-bin (2.35-0ubuntu3.1) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for dbus (1.12.20-2ubuntu4.1) ...
Processing triggers for ca-certificates (20230311ubuntu0.22.04.1) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-39-252:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.
gpg
ubuntu@ip-172-31-39-252:~$ echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(
lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
ubuntu@ip-172-31-39-252:~$ ~
```

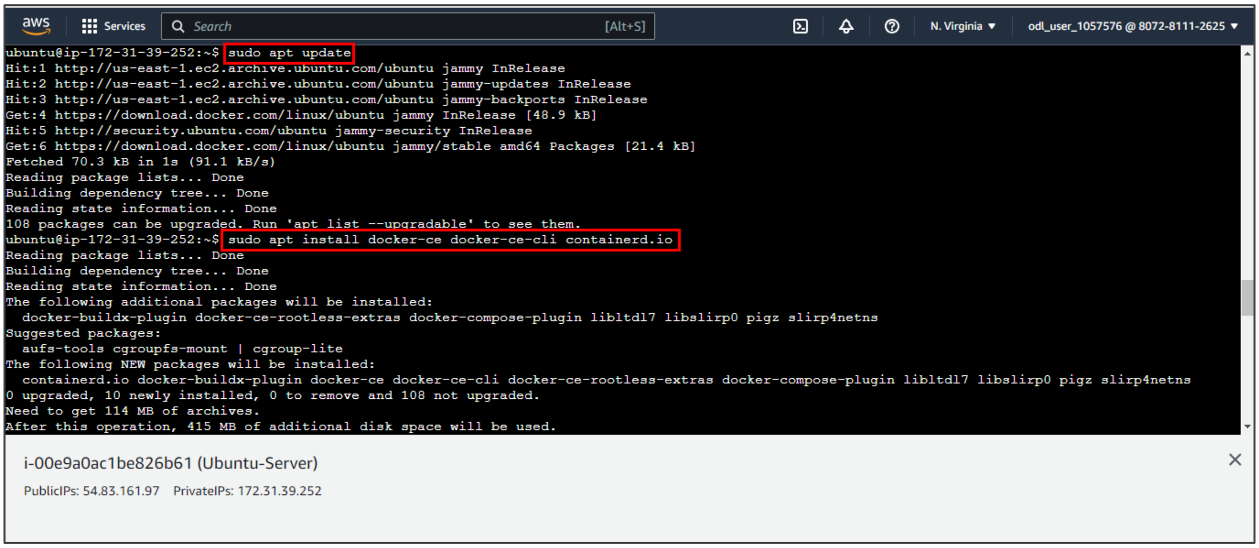
i-00e9a0ac1be826b61 (Ubuntu-Server)

PublicIPs: 54.83.161.97   PrivateIPs: 172.31.39.252

2.5 Run the following commands to install Docker:

sudo apt update

sudo apt install docker-ce docker-ce-cli containerd.io



```
ubuntu@ip-172-31-39-252:~$ sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:4 https://download.docker.com/linux/ubuntu jammy InRelease [48.9 kB]
Hit:5 http://security.ubuntu.com/ubuntu jammy-security InRelease
Get:6 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages [21.4 kB]
Fetched 70.3 kB in 1s (91.1 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
108 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-172-31-39-252:~$ sudo apt install docker-ce docker-ce-cli containerd.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  docker-buildx-plugin docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
0 upgraded, 10 newly installed, 0 to remove and 108 not upgraded.
Need to get 114 MB of archives.
After this operation, 415 MB of additional disk space will be used.
```

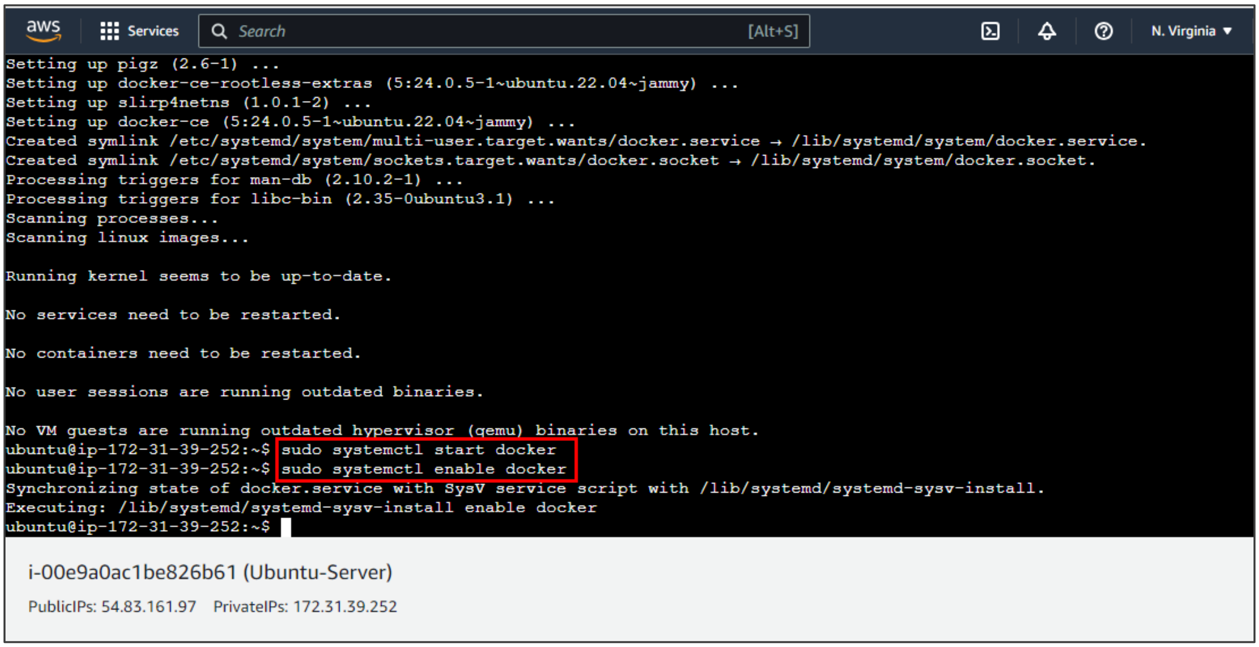
i-00e9a0ac1be826b61 (Ubuntu-Server)

PublicIPs: 54.83.161.97   PrivateIPs: 172.31.39.252

2.6 Run the following commands to start and enable the Docker service:

sudo systemctl start docker

sudo systemctl enable docker



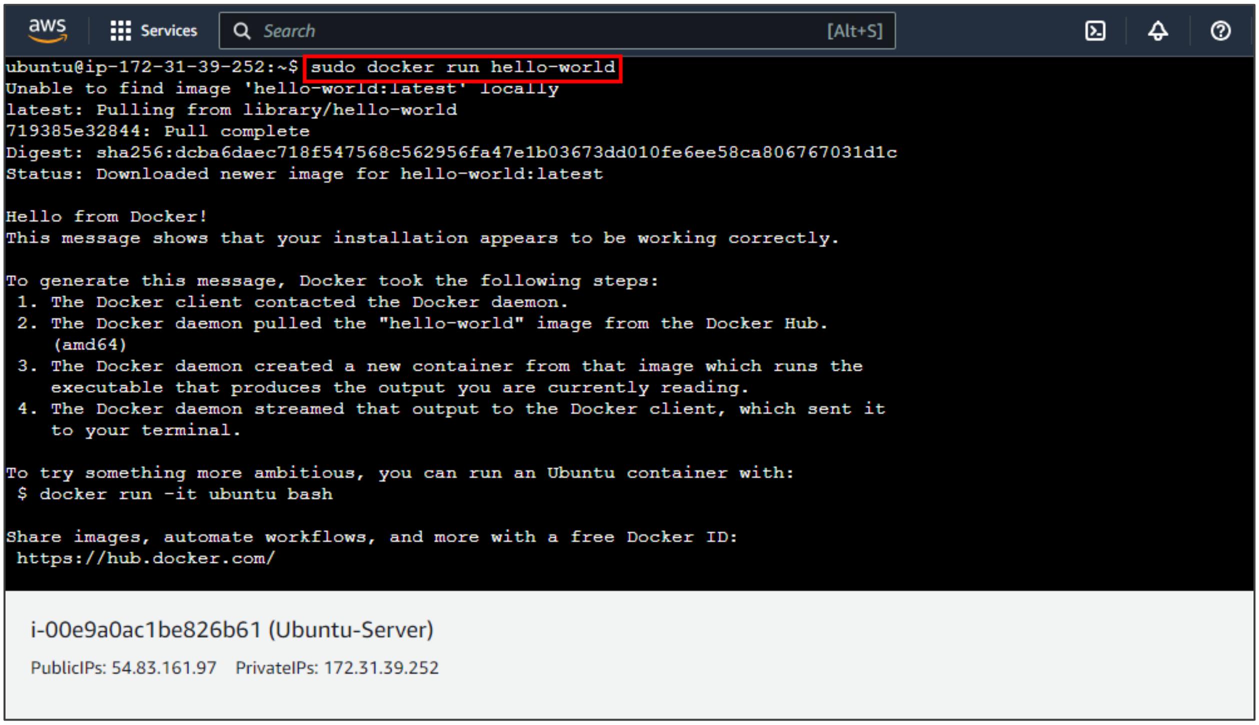
```
Setting up pigz (2.6-1) ...
Setting up docker-ce-rootless-extras (5:24.0.5-1~ubuntu.22.04~jammy) ...
Setting up slirp4netns (1.0.1-2) ...
Setting up docker-ce (5:24.0.5-1~ubuntu.22.04~jammy) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-39-252:~$ sudo systemctl start docker
ubuntu@ip-172-31-39-252:~$ sudo systemctl enable docker
Synchronizing state of docker.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable docker
ubuntu@ip-172-31-39-252:~$
```

i-00e9a0ac1be826b61 (Ubuntu-Server)

PublicIPs: 54.83.161.97 PrivateIPs: 172.31.39.252

Step 3: Run the Hello World container

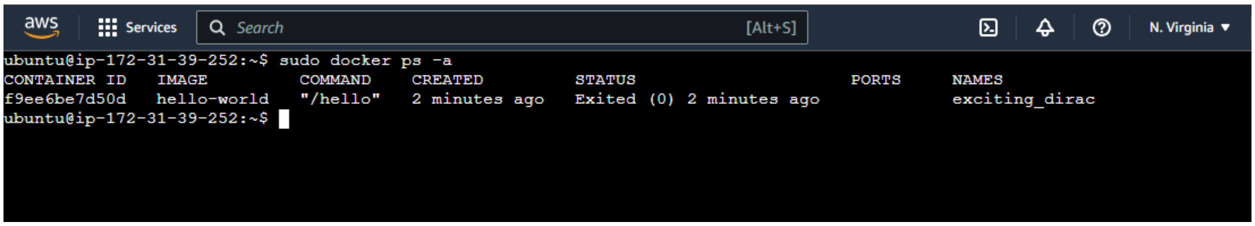
3.1 Run the **Hello World** container using the below command:
sudo docker run hello-world



```
ubuntu@ip-172-31-39-252:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
719385e32844: Pull complete
Digest: sha256:dcba6daec718f547568c562956fa47e1b03673dd010fe6ee58ca806767031d1c
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/
```

i-00e9a0ac1be826b61 (Ubuntu-Server)

PublicIPs: 54.83.161.97 PrivateIPs: 172.31.39.252

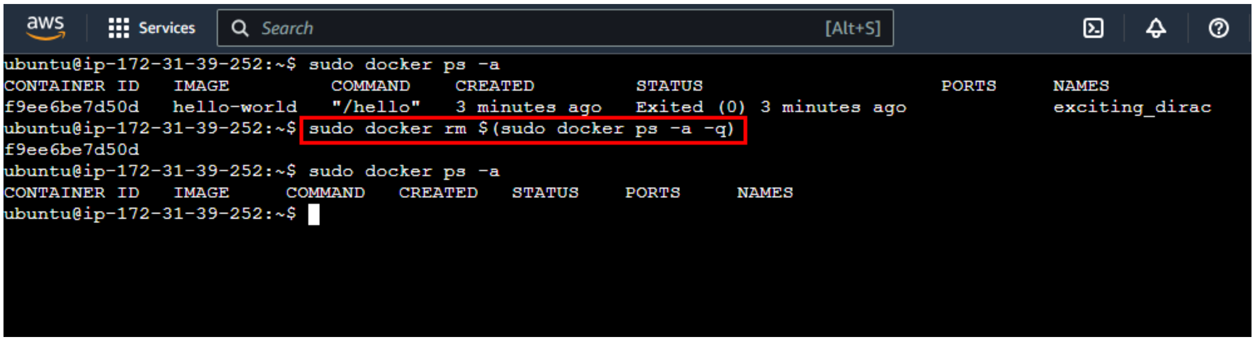


```
ubuntu@ip-172-31-39-252:~$ sudo docker ps -a
CONTAINER ID   IMAGE         COMMAND    CREATED         STATUS                     PORTS     NAMES
f9ee6be7d50d   hello-world   "/hello"   2 minutes ago   Exited (0) 2 minutes ago             exciting_dirac
ubuntu@ip-172-31-39-252:~$
```

You will see a message that indicates your installation is working correctly.

3.2 Execute the following command to remove the container:

sudo docker rm $(docker ps -a -q)



```
ubuntu@ip-172-31-39-252:~$ sudo docker ps -a
CONTAINER ID   IMAGE         COMMAND    CREATED         STATUS                     PORTS     NAMES
f9ee6be7d50d   hello-world   "/hello"   3 minutes ago   Exited (0) 3 minutes ago             exciting_dirac
ubuntu@ip-172-31-39-252:~$ sudo docker rm $(sudo docker ps -a -q)
f9ee6be7d50d
ubuntu@ip-172-31-39-252:~$ sudo docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
ubuntu@ip-172-31-39-252:~$
```

By following these steps, you have successfully run a Hello World container on an EC2 instance using Ubuntu and Docker to enhance container deployment efficiency.