Lab. Installation of Docker Engine on Ubuntu at AWS EC2

※Refer to https://docs.docker.com/engine/install/ubuntu/

1. Install using the repository

   1)Set up the repository

```
$ sudo apt-get update
$ sudo apt-get install ₩
    ca-certificates ₩
    curl ₩
    gnupg
```

   2)Add Docker's official GPG key

```
$ sudo install -m 0755 -d /etc/apt/keyrings
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
$ sudo chmod a+r /etc/apt/keyrings/docker.gpg
```

   3)Set up the stable repository

```
$ echo ₩
    "deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu ₩ "$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | ₩
    sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

2. Install Docker Engine

   1)Repository update

```
$ sudo apt update
```

   2)Install latest version of Docker engine and containerd

```
$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

   3)Docker Service 상태 확인

```
$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2024-01-09 11:15:35 UTC; 5min ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 2469 (dockerd)
      Tasks: 8
     Memory: 34.1M
        CPU: 321ms
     CGroup: /system.slice/docker.service
             └─2469 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Jan 09 11:15:35 ip-172-31-4-124 systemd[1]: Starting Docker Application Container Engine...
Jan 09 11:15:35 ip-172-31-4-124 dockerd[2469]: time="2024-01-09T11:15:35.148902052Z" level=info msg="Starting up"
Jan 09 11:15:35 ip-172-31-4-124 dockerd[2469]: time="2024-01-09T11:15:35.151154217Z" level=info msg="detected 127.0.0.53 nameserver, assuming systemd-resolved, so using resolv.conf: /run/systemd/resolve/res>Jan 09 11:15:35 ip-172-31-4-124 dockerd[2469]: time="2024-01-09T11:15:35.306600798Z" level=info msg="Loading containers: start."
Jan 09 11:15:35 ip-172-31-4-124 dockerd[2469]: time="2024-01-09T11:15:35.626878778Z" level=info msg="Loading containers: done."
Jan 09 11:15:35 ip-172-31-4-124 dockerd[2469]: time="2024-01-09T11:15:35.694477632Z" level=info msg="Docker daemon" commit=311b9ff graphdriver=overlay2 version=24.0.7
Jan 09 11:15:35 ip-172-31-4-124 dockerd[2469]: time="2024-01-09T11:15:35.694911232Z" level=info msg="Daemon has completed initialization"
Jan 09 11:15:35 ip-172-31-4-124 dockerd[2469]: time="2024-01-09T11:15:35.747364896Z" level=info
```

```
    msg="API listen on /run/docker.sock"
    Jan 09 11:15:35 ip-172-31-4-124 systemd[1]: Started Docker Application Container Engine.
    lines 1-21/21 (END)
```

4)docker version 확인

```
    $ docker -v
    Docker version 23.0.4, build f480fb1
```

3. Post-installation steps for Linux

1)Manage Docker as a non-root user

-Docker는 TCP 포트 대신 Unix socket과 bind하기 때문에 반드시 root 또는 sudo 권한으로 실행해야 한다.

-일반 유저권한으로 실행하기 위한 작업 필요

2)docker 그룹 생성

```
    $ sudo groupadd docker
```

3)docker 그룹에 유저 추가

```
    $ sudo usermod -aG docker $USER
```

4)log out and log in

```
    $ logout
    $ id
    uid=1000(ubuntu) gid=1000(ubuntu)
    groups=1000(ubuntu),4(adm),20(dialout),24(cdrom),25(floppy),27(sudo),29(audio),30(dip),44(video),46(plugdev),
    119(netdev),120(lxd),999(docker)
```

5)설치 버전 확인하기

```
    $ docker version
    Client: Docker Engine - Community
     Version:           24.0.7
     API version:       1.43
     Go version:        go1.20.10
     Git commit:        afdd53b
     Built:             Thu Oct 26 09:07:41 2023
     OS/Arch:           linux/amd64
     Context:           default

    Server: Docker Engine - Community
     Engine:
      Version:          24.0.7
      API version:      1.43 (minimum version 1.12)
      Go version:       go1.20.10
      Git commit:       311b9ff
      Built:            Thu Oct 26 09:07:41 2023
      OS/Arch:          linux/amd64
      Experimental:     false
     containerd:
      Version:          1.6.26
      GitCommit:        3dd1e886e55dd695541fdcd67420c2888645a495
     runc:
      Version:          1.1.10
      GitCommit:        v1.1.10-0-g18a0cb0
     docker-init:
      Version:          0.19.0
      GitCommit:        de40ad0
```

6)Verify that you can run docker commands without sudo

```
    $ docker run hello-world
    Unable to find image 'hello-world:latest' locally
    latest: Pulling from library/hello-world
```

```
2db29710123e: Pull complete
Digest: sha256:4e83453afed1b4fa1a3500525091dbfca6ce1e66903fd4c01ff015dbcb1ba33e
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/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

4. Uninstall Docker Engine

1)Uninstall the Docker Engine, CLI, and Containerd packages:

```
$ sudo apt-get purge docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin docker-ce-rootless-extras
```

2)Images, containers, volumes, or customized configuration files on your host are not automatically removed. To delete all images, containers, and volumes:

```
$ sudo rm -rf /var/lib/docker
$ sudo rm -rf /var/lib/containerd
```