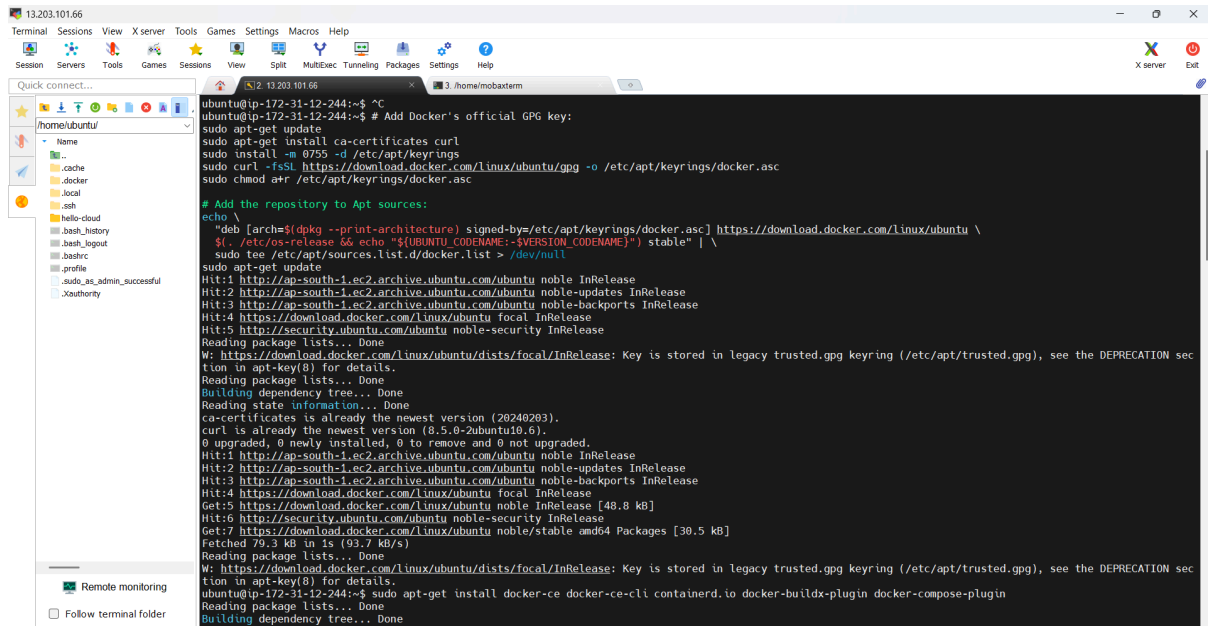


Documentations

- Below the Screenshots is

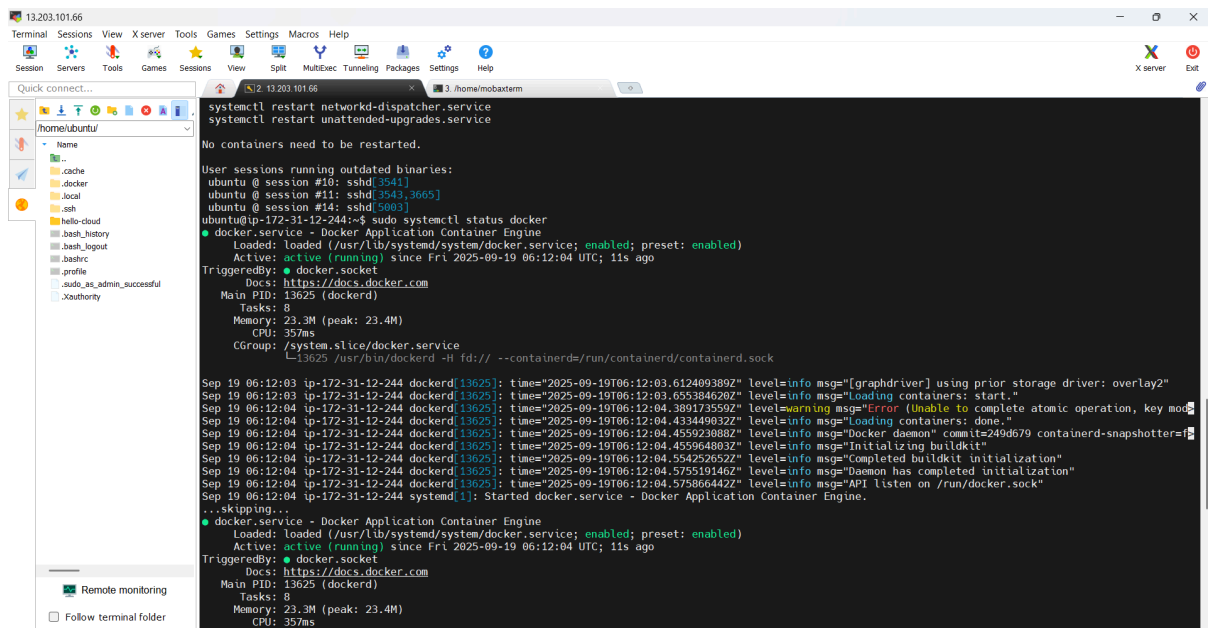
1.Docker installation & version check.

- Docker was successfully installed and verified using the “docker –version” command, confirming that the system is ready for containerization.



```
ubuntu@ip-172-31-12-244:~$ ^C
ubuntu@ip-172-31-12-244:~$ # Add Docker's official GPG key:
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \
  $(. /etc/os-release && echo "${UBUNTU_CODENAME:-$VERSION_CODENAME}") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu focal InRelease
Hit:5 https://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/focal/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20240203).
curl is already the newest version (8.5.0-2ubuntu10.6).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu focal InRelease
Get:5 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Hit:6 https://security.ubuntu.com/ubuntu noble-security InRelease
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [30.5 kB]
Fetched 79.3 kB in 1s (93.7 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/focal/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
ubuntu@ip-172-31-12-244:~$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
Reading package lists... Done
Building dependency tree... Done
```



```
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

User sessions running outdated binaries:
ubuntu @ session #10: sshd[3541]
ubuntu @ session #11: sshd[3543,3665]
ubuntu @ session #14: sshd[5002]
ubuntu@ip-172-31-12-244:~$ 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 Fri 2025-09-19 06:12:04 UTC; 11s ago
   TriggeredBy: ● docker.socket
   Docs: https://docs.docker.com
   Main PID: 13625 (dockerd)
   Tasks: 8
   Memory: 23.3M (peak: 23.4M)
   CPU: 357ms
   CGroup: /system.slice/docker.service
           └─13625 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Sep 19 06:12:03 ip-172-31-12-244 dockerd[13625]: time="2025-09-19T06:12:03.612499389Z" level=info msg="[graphdriver] using prior storage driver: overlay2"
Sep 19 06:12:03 ip-172-31-12-244 dockerd[13625]: time="2025-09-19T06:12:03.656384620Z" level=info msg="Loading containers: start."
Sep 19 06:12:04 ip-172-31-12-244 dockerd[13625]: time="2025-09-19T06:12:04.389173559Z" level=warning msg="Error (Unable to complete atomic operation, key mod"
Sep 19 06:12:04 ip-172-31-12-244 dockerd[13625]: time="2025-09-19T06:12:04.433449032Z" level=info msg="Loading containers: done."
Sep 19 06:12:04 ip-172-31-12-244 dockerd[13625]: time="2025-09-19T06:12:04.455923888Z" level=info msg="Docker daemon commit=249d679 container-snapshots="
Sep 19 06:12:04 ip-172-31-12-244 dockerd[13625]: time="2025-09-19T06:12:04.455964803Z" level=info msg="Initializing buildkit"
Sep 19 06:12:04 ip-172-31-12-244 dockerd[13625]: time="2025-09-19T06:12:04.554252652Z" level=info msg="Completed buildkit initialization"
Sep 19 06:12:04 ip-172-31-12-244 dockerd[13625]: time="2025-09-19T06:12:04.575519146Z" level=info msg="Daemon has completed initialization"
Sep 19 06:12:04 ip-172-31-12-244 dockerd[13625]: time="2025-09-19T06:12:04.575866442Z" level=info msg="API listen on /run/docker.sock"
Sep 19 06:12:04 ip-172-31-12-244 systemd[1]: Started docker.service - Docker Application Container Engine.

--skipping--
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: enabled)
   Active: active (running) since Fri 2025-09-19 06:12:04 UTC; 11s ago
   TriggeredBy: ● docker.socket
   Docs: https://docs.docker.com
   Main PID: 13625 (dockerd)
   Tasks: 8
   Memory: 23.3M (peak: 23.4M)
   CPU: 357ms
```



2.Docker file creation & Container build

- A Dockerfile was created and used to build a custom image(hello-cloud:1.0).
- This define the environment, dependencies, and instructions needed to run the application inside a container.

```
ubuntu@ip-172-31-12-244:~$ cd hello-cloud
ubuntu@ip-172-31-12-244:~/hello-cloud$ nano requirements.txt
ubuntu@ip-172-31-12-244:~/hello-cloud$ nano app.py
ubuntu@ip-172-31-12-244:~/hello-cloud$ nano Dockerfile
ubuntu@ip-172-31-12-244:~/hello-cloud$ ls
Dockerfile app.py requirements.txt
ubuntu@ip-172-31-12-244:~/hello-cloud$ docker build -t hello-cloud:1.0 .
[+] Building 2.2s (10/10) FINISHED
=> [internal] load build definition from Dockerfile
=> transferring dockerfile: 421B
=> [internal] load metadata for docker.io/library/python:3.11-slim
=> [internal] load .dockerignore
=> transferring context: 2B
=> [1/5] FROM docker.io/library/python:3.11-slim@sha256:a0939570b38cddeb861b8e75d20b1c8218b21562b18f301171904b544e8cf228
=> [internal] load build context
=> transferring context: 738B
=> CACHED [2/5] WORKDIR /app
=> CACHED [3/5] COPY requirements.txt .
=> CACHED [4/5] RUN pip install --no-cache-dir -r requirements.txt
=> CACHED [5/5] COPY . .
=> exporting to image
=> exporting layers
=> writing image sha256:02c513ac735592e96bf1d8f79348ca03f30336878723ceda3e651b0adaf9c24a
=> naming to docker.io/library/hello-cloud:1.0
```

3.Container running and webpage accessible in browser

□ Container Status Check:

This is important for ensuring that the container is actually running. You can verify this with a command like **“docker ps”** ,which helps confirm that your container is active and serving the webpage.

□ Accessing the Webpage:

Providing the exact URL format for accessing the webpage(e.g.,**http://localhost:5000**)would be important because it guides users on how to open the browser and connect to the containerized service

```

ubuntu@ip-172-31-12-244:~/hello-cloud$ docker run -d -p 5000:5000 --name hello-cloud hello-cloud:1.0
ff7d58ffe3c4f5b115cd0bd9d304f8658b66a8a79e1db8267c628c30d38f628
ubuntu@ip-172-31-12-244:~/hello-cloud$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
ff7d58ffe3c4   hello-cloud:1.0  "python app.py"         10 seconds ago Up 10 seconds  0.0.0.0:5000->5000/tcp, [::]:5000->5000/tcp  hello-cloud
ubuntu@ip-172-31-12-244:~/hello-cloud$ curl http://localhost:5000
Hello Cloud!ubuntu@ip-172-31-12-244:~/hello-cloud$

```

- Include the public URL/IP:port link to the web app:
 - To access the deployed web application, include the public URL or IP address with the port number in your documentation. For example: <http://13.203.75.97.5000>.
 - When visited, the app should display a confirmation message like **“Hello Cloud!”** indicating successful deployment.

