**Internship Task Documentation**

**Task1:** Ansible for Configuration Management — Install Docker, Pull Image, Run Container Automatically  
**Name:** Ujjawal Rawat  
**Date:** 03-Oct-2025

**1. Objective**

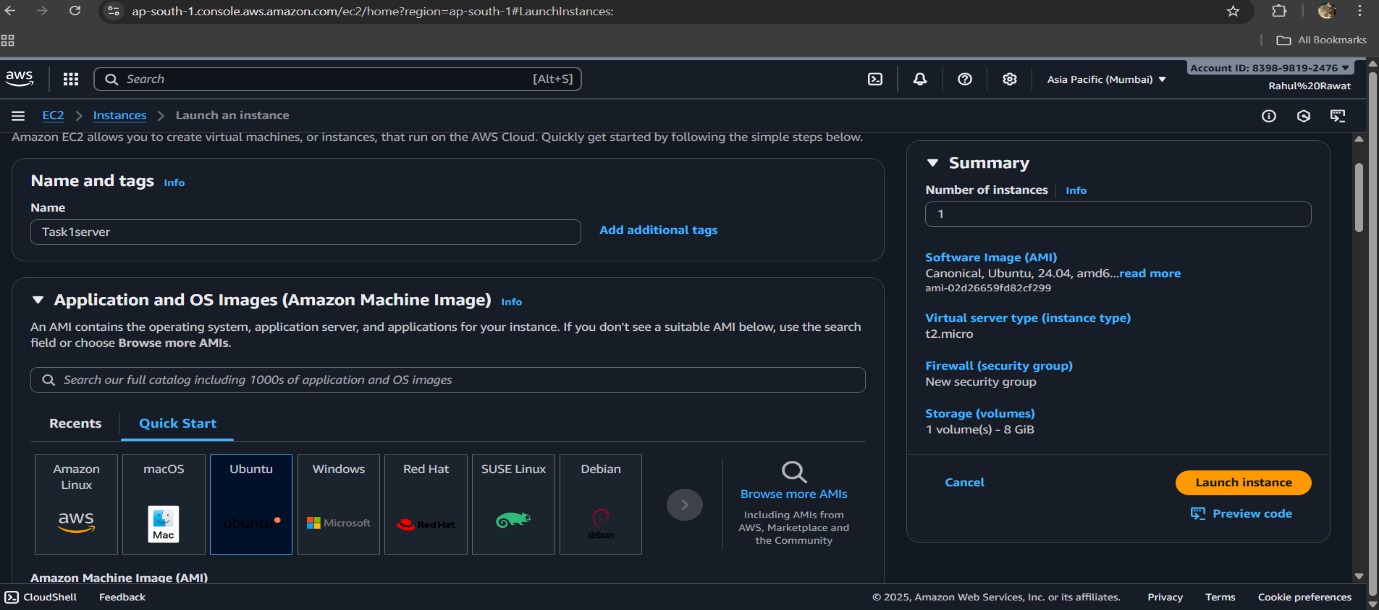
The objective of this task is to automate the deployment of Docker on an Ubuntu EC2 instance using Ansible. The playbook should:

1. Install Docker automatically on the server.
2. Pull a Docker image from Docker Hub.
3. Run the Docker container automatically with restart policy, so it starts on server boot.

This demonstrates **configuration management** and **automation skills** using Ansible.

**2. Prerequisites**

* **AWS EC2 Ubuntu instance** running.
* **Private PEM key** to access the EC2 via SSH.
* **Docker Hub account** with a prepared Docker image.
* **Ansible installed** on your local machine or EC2.



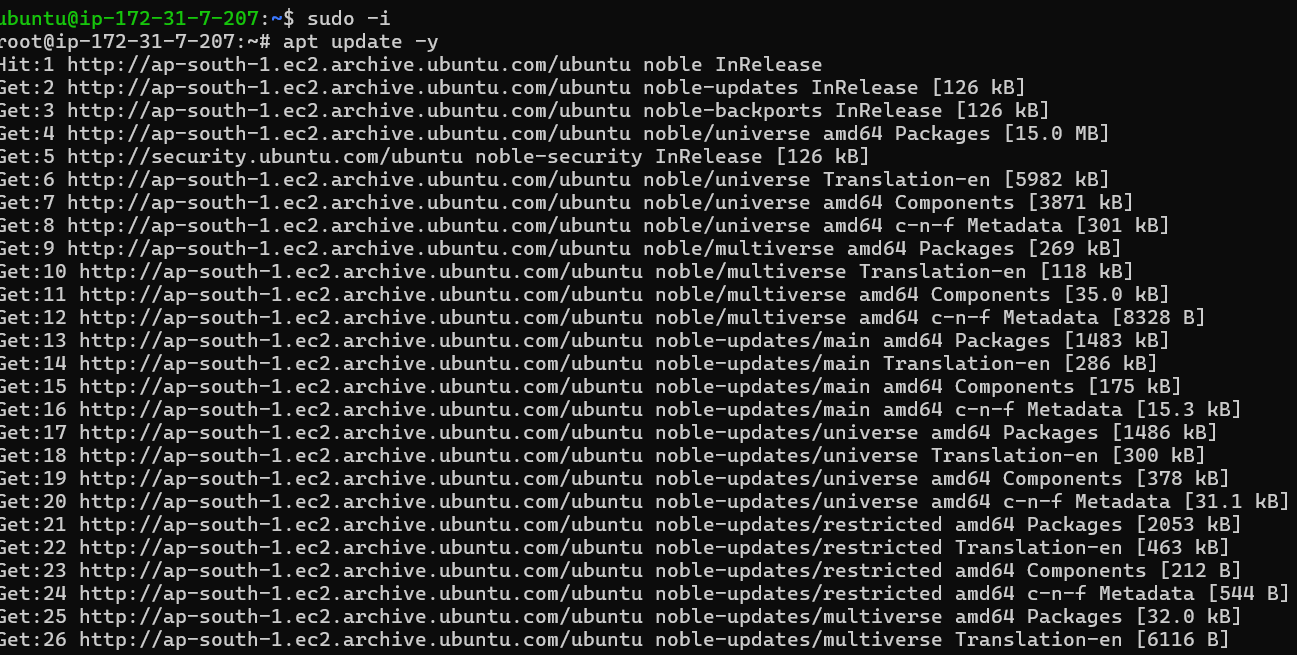
**Step 1: Test SSH Connection**

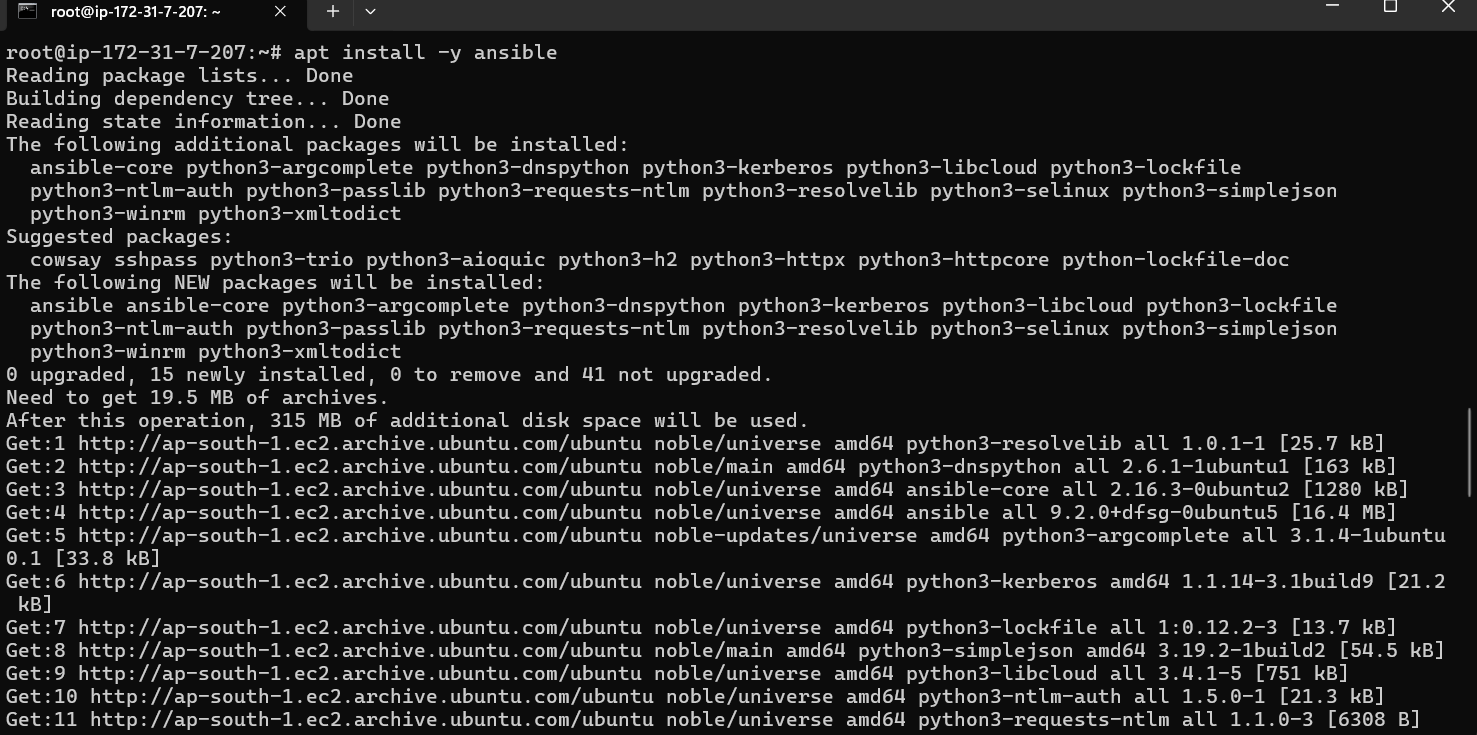
* SSH into the EC2 to verify connectivity:

ssh -i /root/.ssh/devkey.pem ubuntu@<EC2\_PUBLIC\_IP>

**Expected:** You should see the terminal prompt of your EC2 instance.

**Explanation:** Verifying SSH ensures that Ansible can later connect and execute tasks remotely.





**Step 2: Create Ansible Inventory**

* Ansible uses an **inventory file** to know which hosts to manage.

**inventory.ini**

[servers]

myec2 ansible\_host=<EC2\_PUBLIC\_IP> ansible\_user=ubuntu ansible\_ssh\_private\_key\_file=/root/.ssh/devkey.pem

**Explanation:**

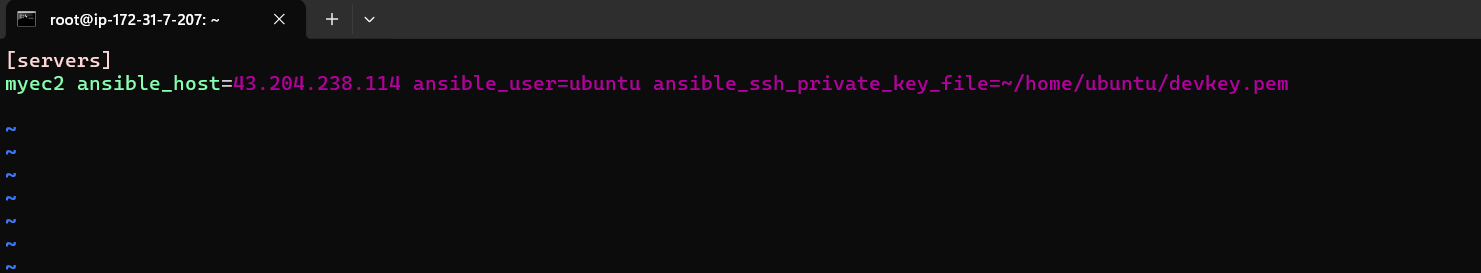
* [servers] is a group name.
* ansible\_host → public IP of EC2.
* ansible\_user → user to connect as (ubuntu on Ubuntu EC2).
* ansible\_ssh\_private\_key\_file → full path to PEM key.
* Test the connection:

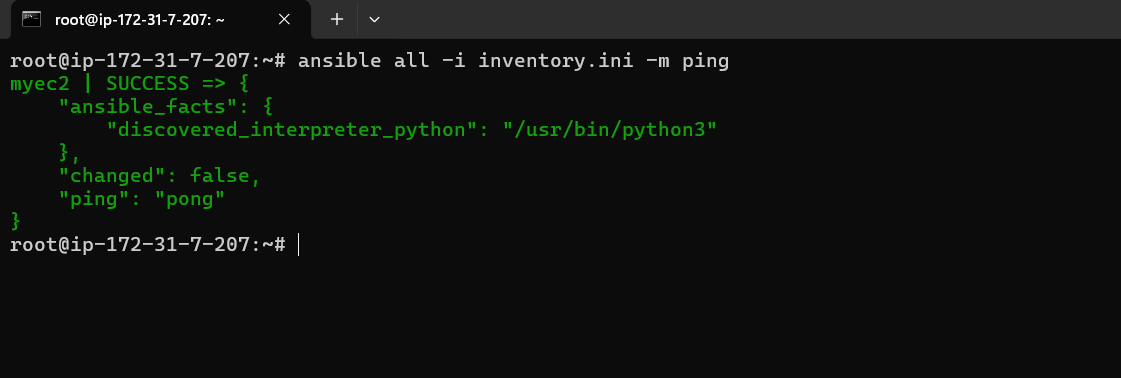
ansible all -i inventory.ini -m ping

**Expected Output:**

myec2 | SUCCESS => {"ping": "pong"}

**Explanation:** A successful ping confirms that Ansible can communicate with the EC2 host.





**Step 3: Prepare Docker Image**

1. Create a folder for the application:

mkdir -p ~/myapp

cd ~/myapp

1. Create **Dockerfile**:

FROM nginx:latest

COPY index.html /usr/share/nginx/html/index.html

EXPOSE 80

**Explanation:**

* FROM nginx:latest → base image is Nginx web server.
* COPY index.html → copies the HTML file to be served.
* EXPOSE 80 → exposes port 80 for HTTP traffic.

1. Create **index.html**:

<!DOCTYPE html>

<html>

<head>

<title>My App</title>

</head>

<body>

<h1>Hello from my Docker app!</h1>

</body>

</html>

1. Build Docker image:

docker build -t myapp .

1. Tag the image for Docker Hub:

docker tag myapp <DockerHubUsername>/myapp:latest

1. Push to Docker Hub:

docker login -u <DockerHubUsername>

docker push <DockerHubUsername>/myapp:latest

****

**Step 4: Create Ansible Playbook**

**docker-playbook.yml**

---

- name: Setup Docker and run container

hosts: all

become: yes

tasks:

- name: Update apt and install required packages

apt:

name: "{{ item }}"

state: present

update\_cache: yes

loop:

- apt-transport-https

- ca-certificates

- curl

- gnupg-agent

- software-properties-common

- name: Add Docker GPG key

apt\_key:

url: https://download.docker.com/linux/ubuntu/gpg

state: present

- name: Add Docker repository

apt\_repository:

repo: deb [arch=amd64] https://download.docker.com/linux/ubuntu {{ ansible\_distribution\_release }} stable

state: present

- name: Install Docker

apt:

name: docker-ce

state: latest

update\_cache: yes

- name: Start and enable Docker service

systemd:

name: docker

state: started

enabled: yes

- name: Pull Docker image from Docker Hub

docker\_image:

name: <DockerHubUsername>/myapp

source: pull

- name: Run Docker container

docker\_container:

name: my\_container

image: <DockerHubUsername>/myapp

state: started

restart\_policy: always

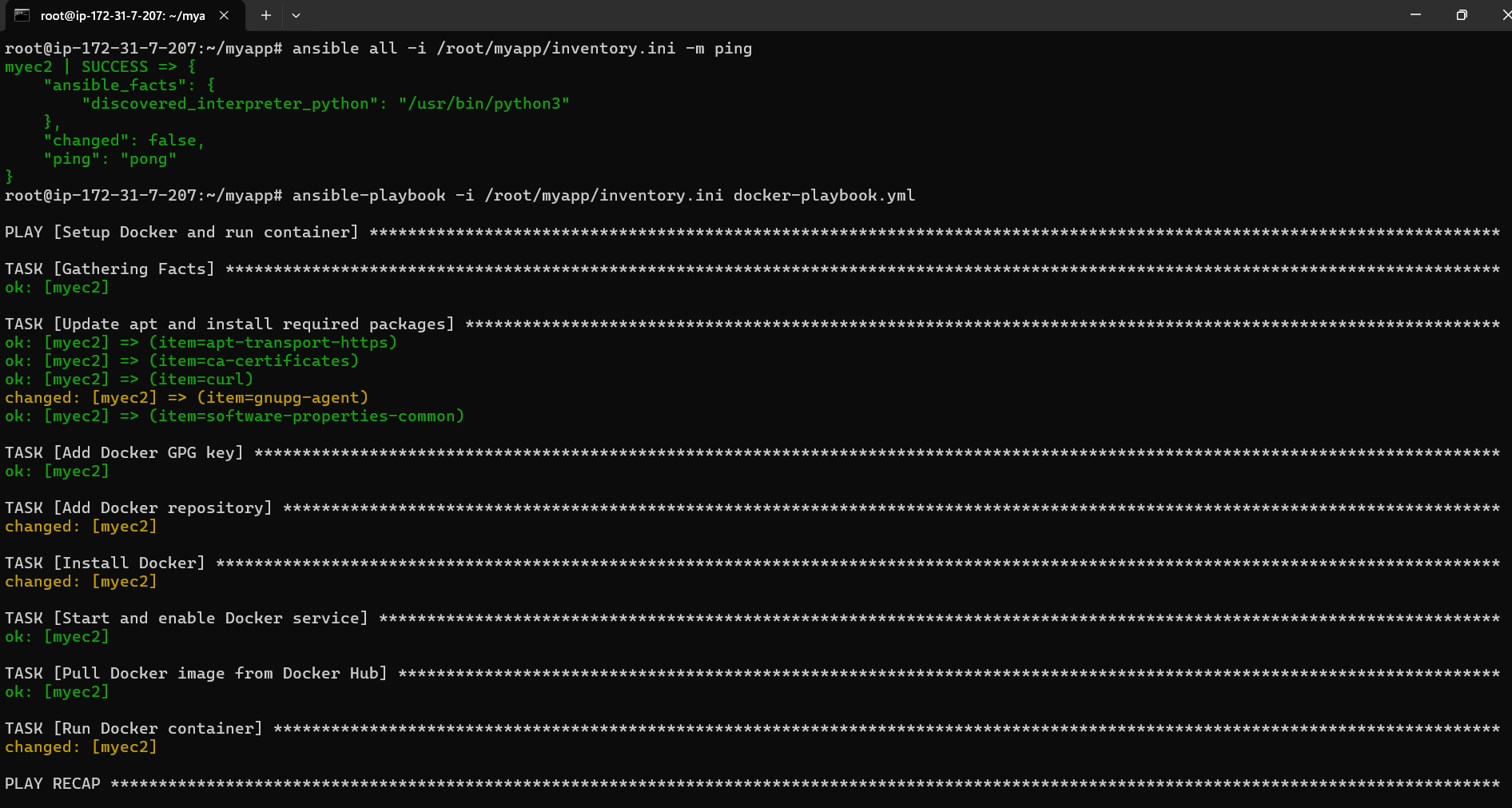
published\_ports:

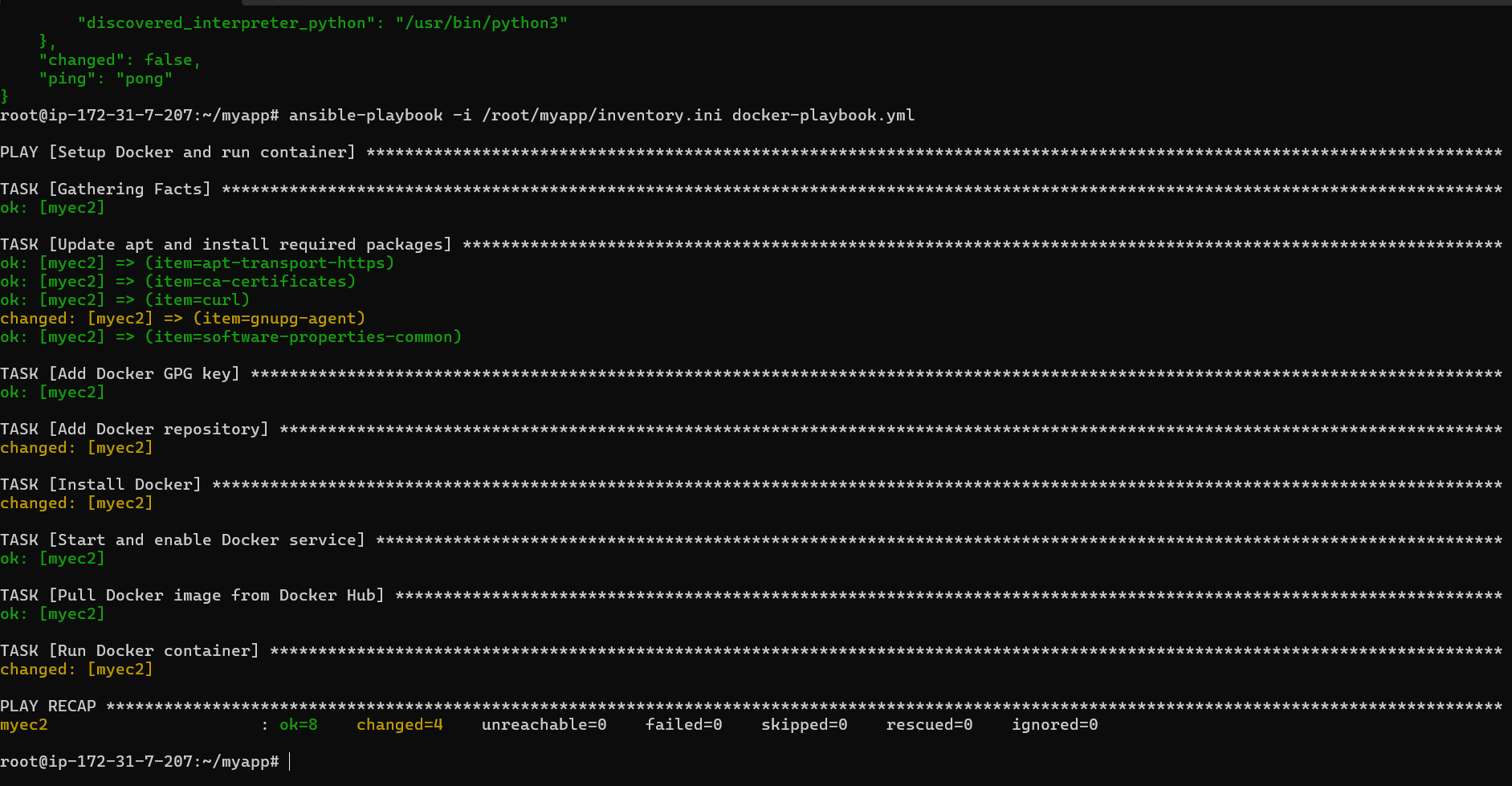
- "8080:80"

**Step 6: Run the Playbook**

ansible-playbook -i inventory.ini docker-playbook.yml

**Outcome:**

* Docker installed ✅
* Docker image pulled ✅
* Container running on port 8080 ✅



**Step 6: Verify Docker Container**

docker ps

* You should see my\_container running.
* Access via browser:

http://<EC2\_PUBLIC\_IP>:8080

**4. Conclusion**

* Successfully automated **Docker deployment using Ansible**.
* Container is running and accessible over HTTP.
* Playbook is **reusable**, which demonstrates good DevOps practices.
* This approach reduces manual server setup time and ensures consistency.