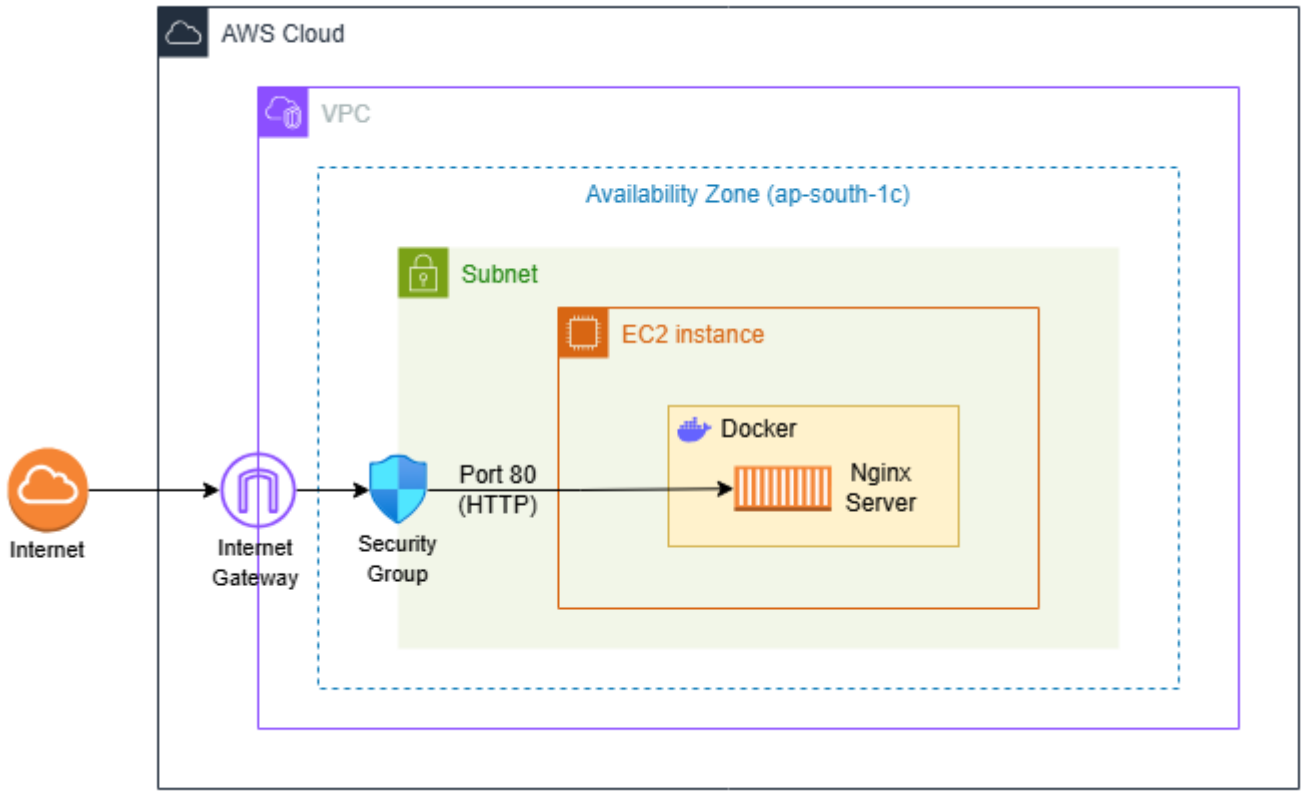
TASK 3 - AWS EC2 Deployment

Requirements: Launch EC2 (cost-optimized: t2.micro / t3.micro), Install Docker and Run containers on EC2.

Prerequisites: An AWS account with permission to create EC2 instances and Security Groups.

Architecture:



An EC2 instance is launched inside a VPC/subnet with an Internet Gateway attached, and a Security Group allows inbound HTTP (80) so users on the internet can access a NGINX server running inside a Docker container on the instance.

Components and roles:

VPC + Subnet (network boundary): The VPC is your isolated network in AWS, and the EC2 instance is placed inside a subnet within that VPC (diagram shows a single Availability Zone and subnet). Because this instance is reachable from the internet, the subnet effectively behaves as a public subnet (i.e., it has a route to the internet via an Internet Gateway and the instance has a public IP).

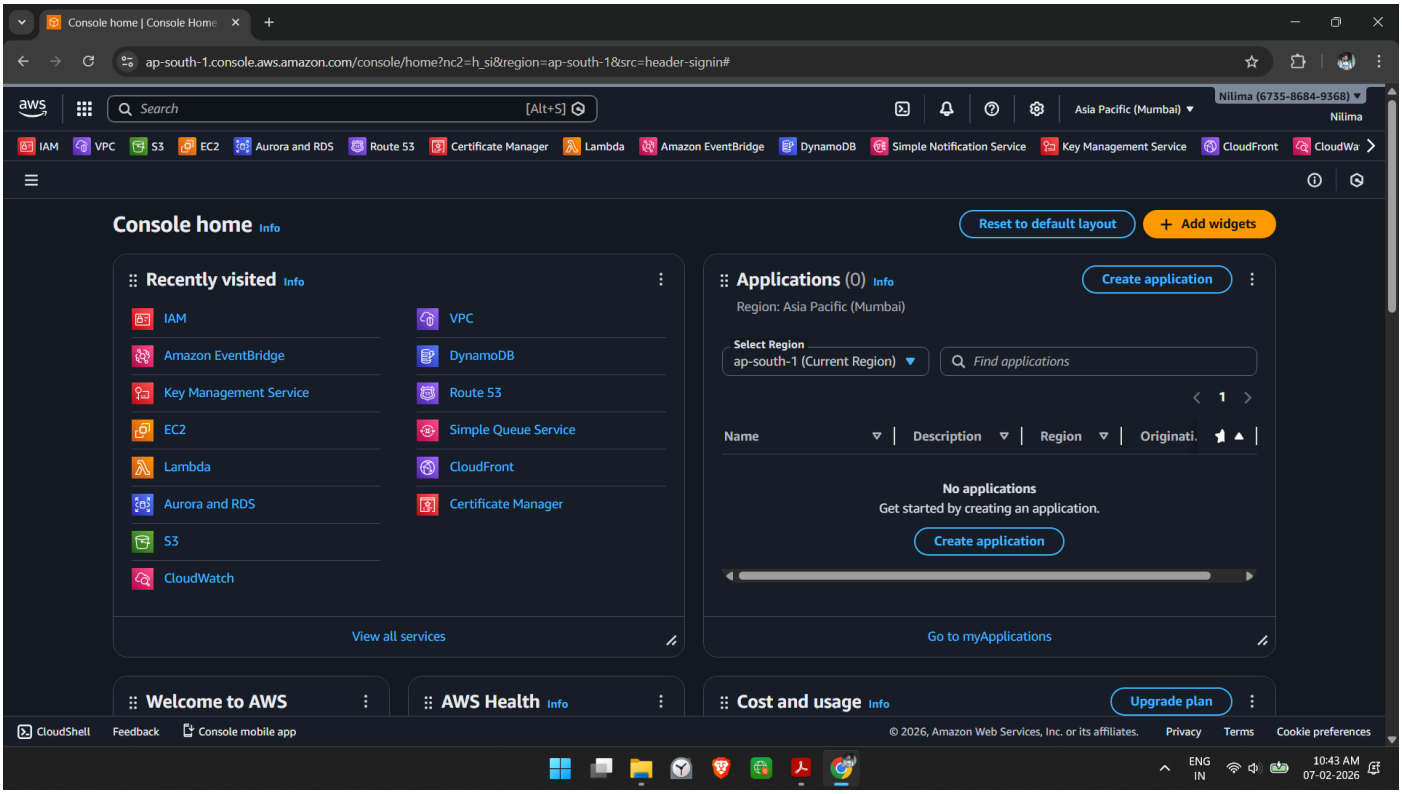
Internet Gateway (IGW) (internet connectivity): An Internet Gateway is the VPC component that enables communication between resources in your VPC and the public internet. For internet access to work, the subnet's route table typically includes a default route 0.0.0.0/0 pointing to the IGW, which sends non-local traffic out to the internet.

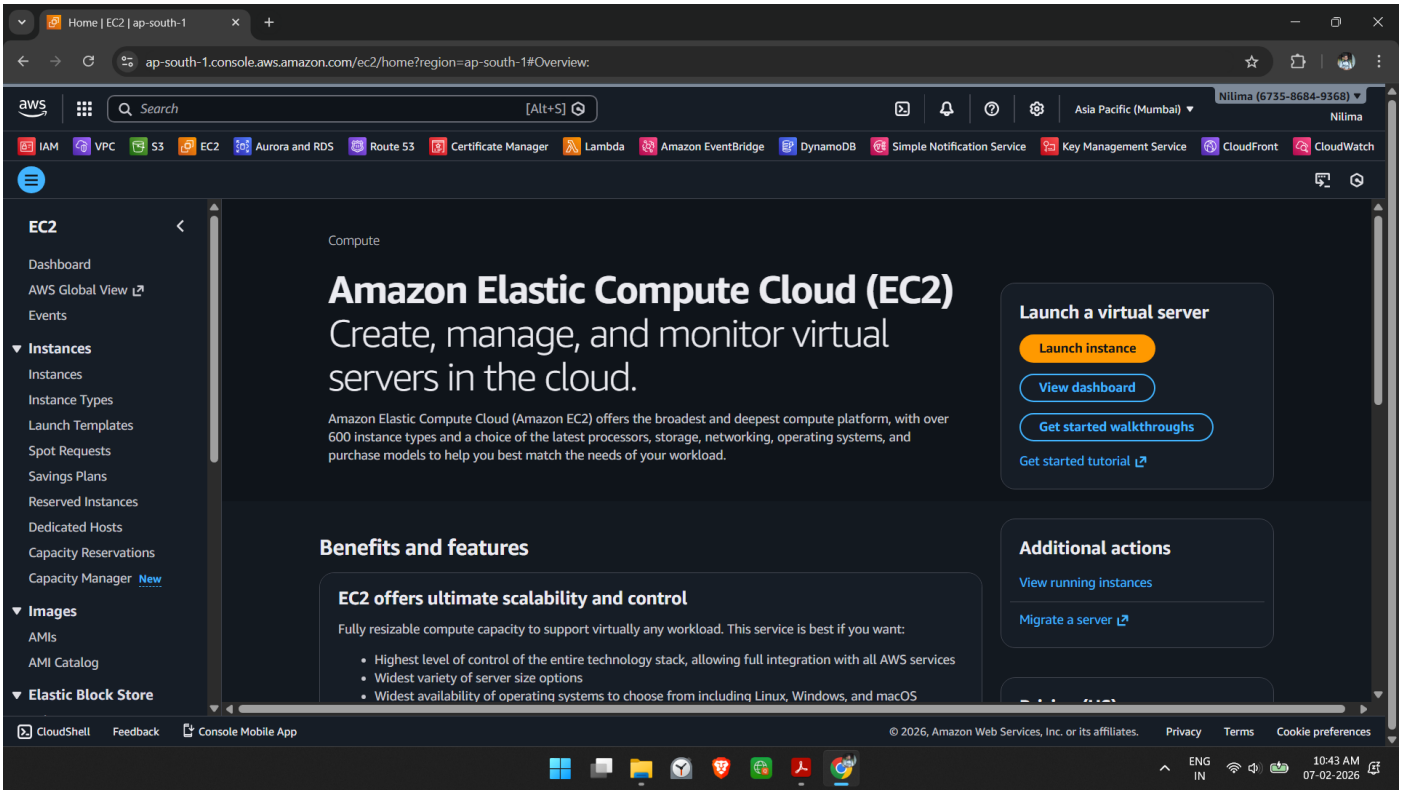
EC2 instance (compute host): The EC2 instance is the virtual machine that runs Docker and hosts your application workload. It has a public IPv4 address (shown in screenshots), which is what you use in the browser to reach the web server.

Step-by-step procedure:

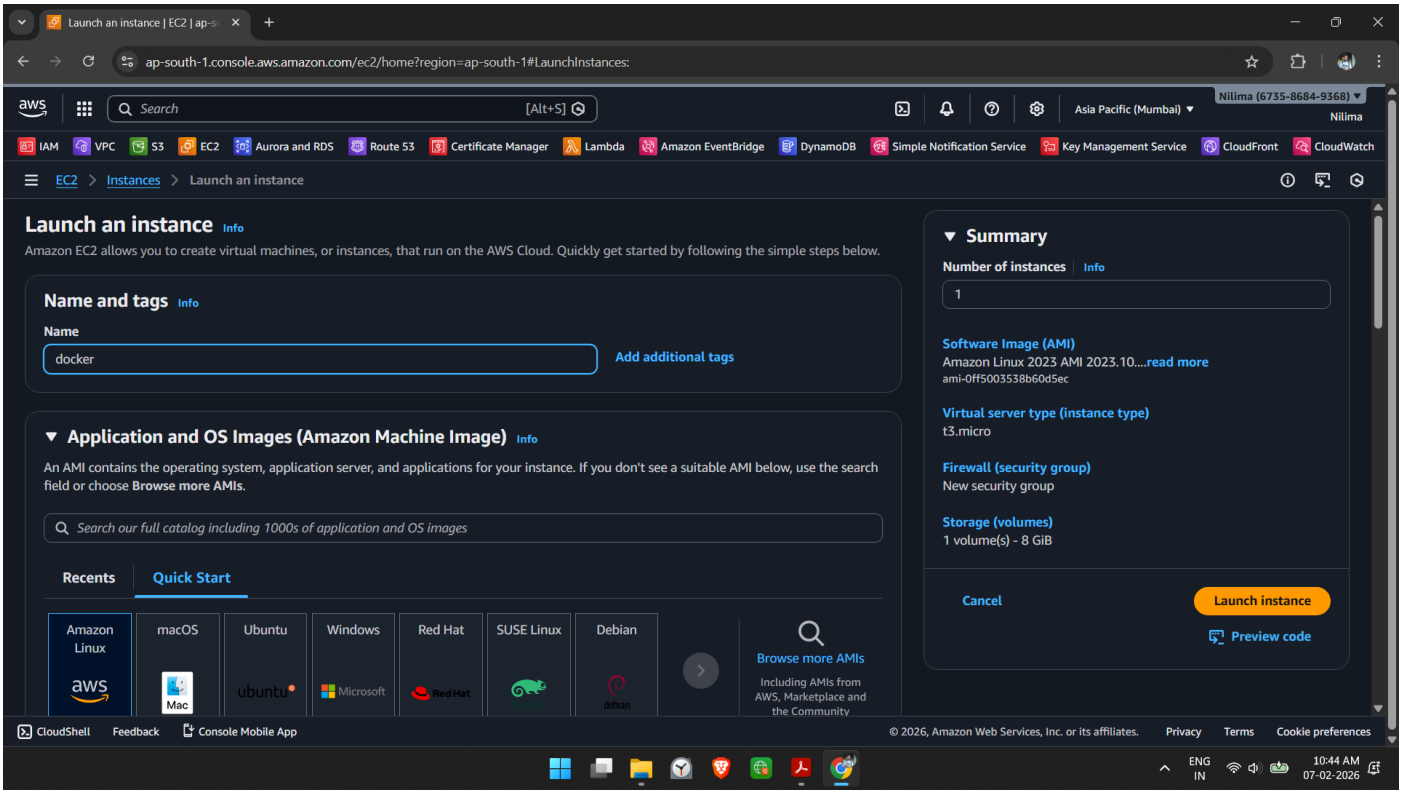
I) Launch an EC2 instance

1. Open the AWS Console and go to EC2 → Launch instance.

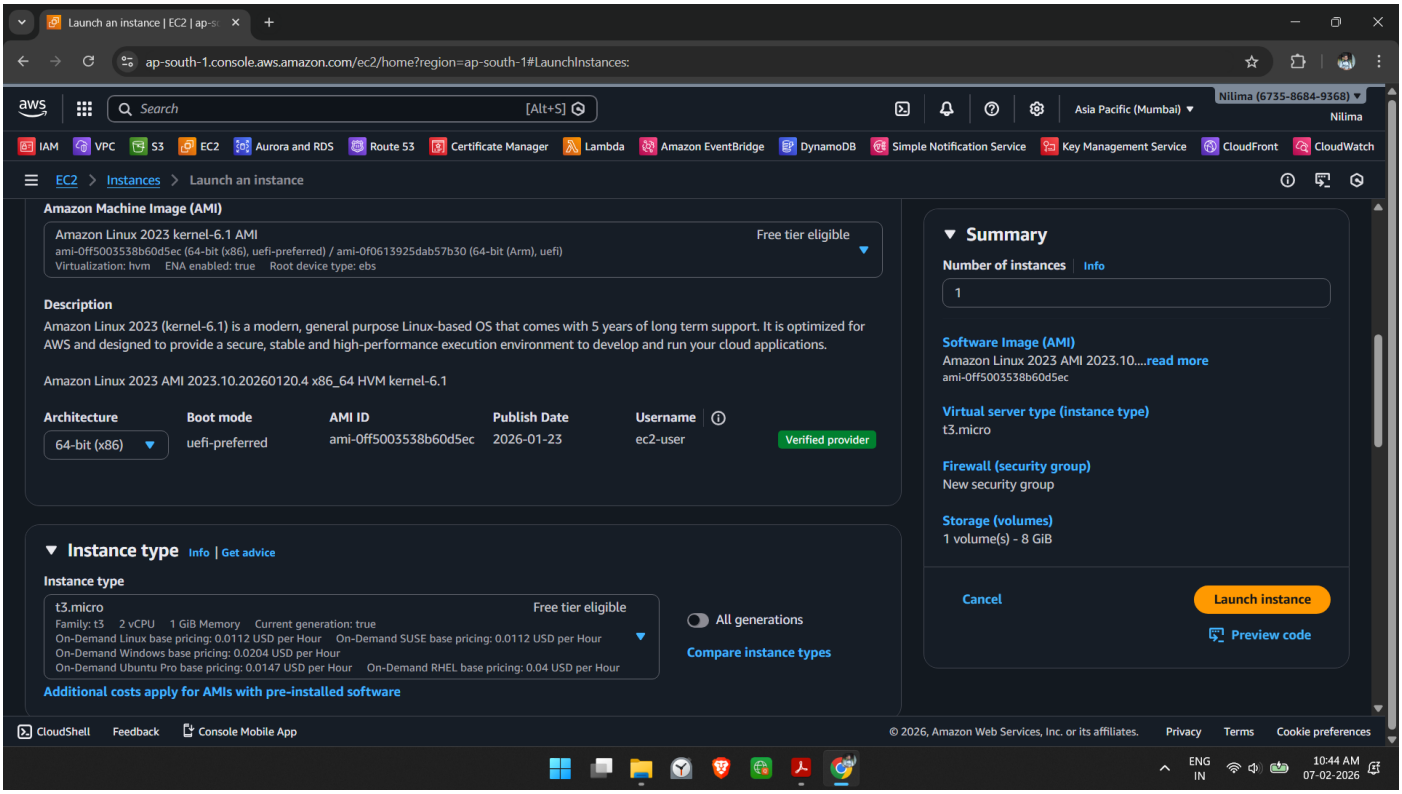




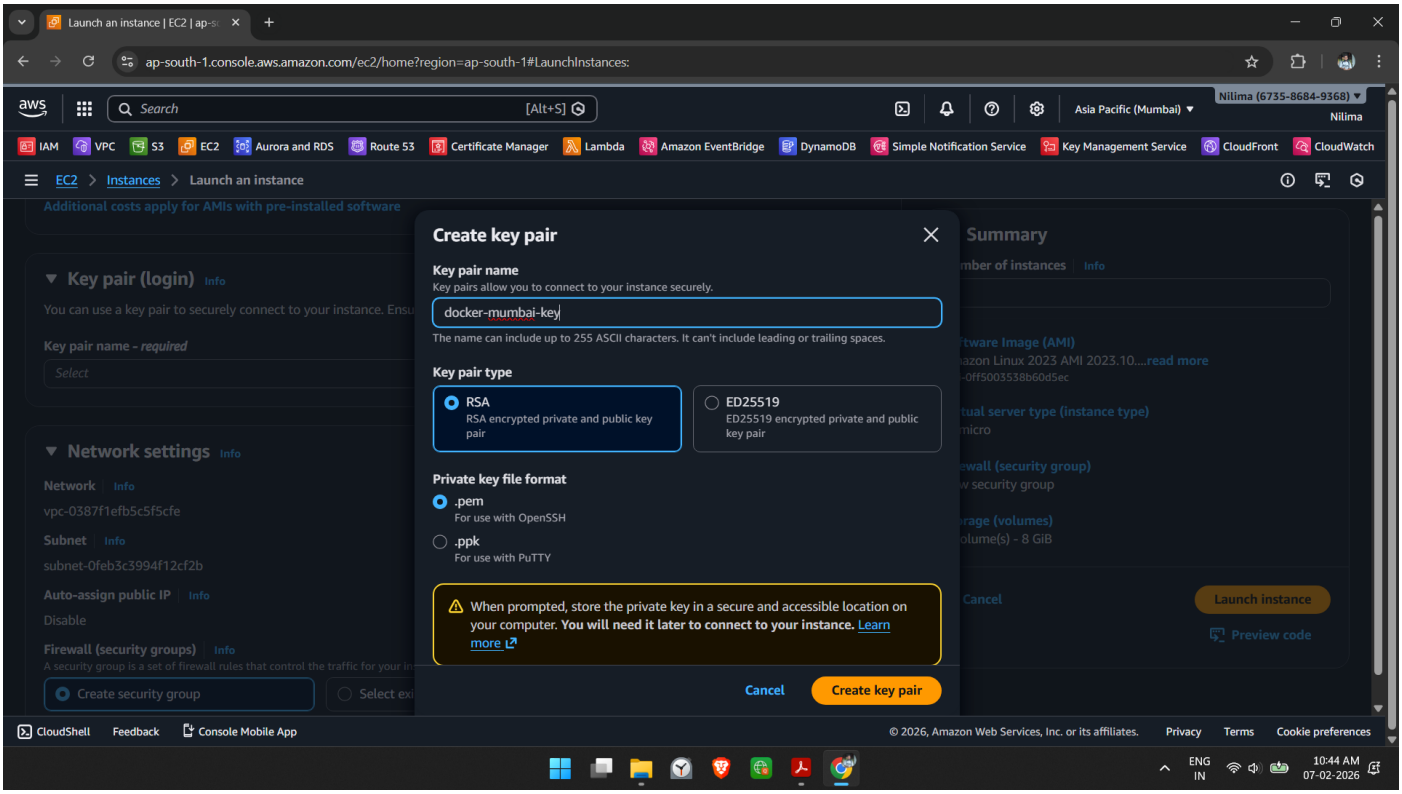
2. Provide an instance name (shown as docker).



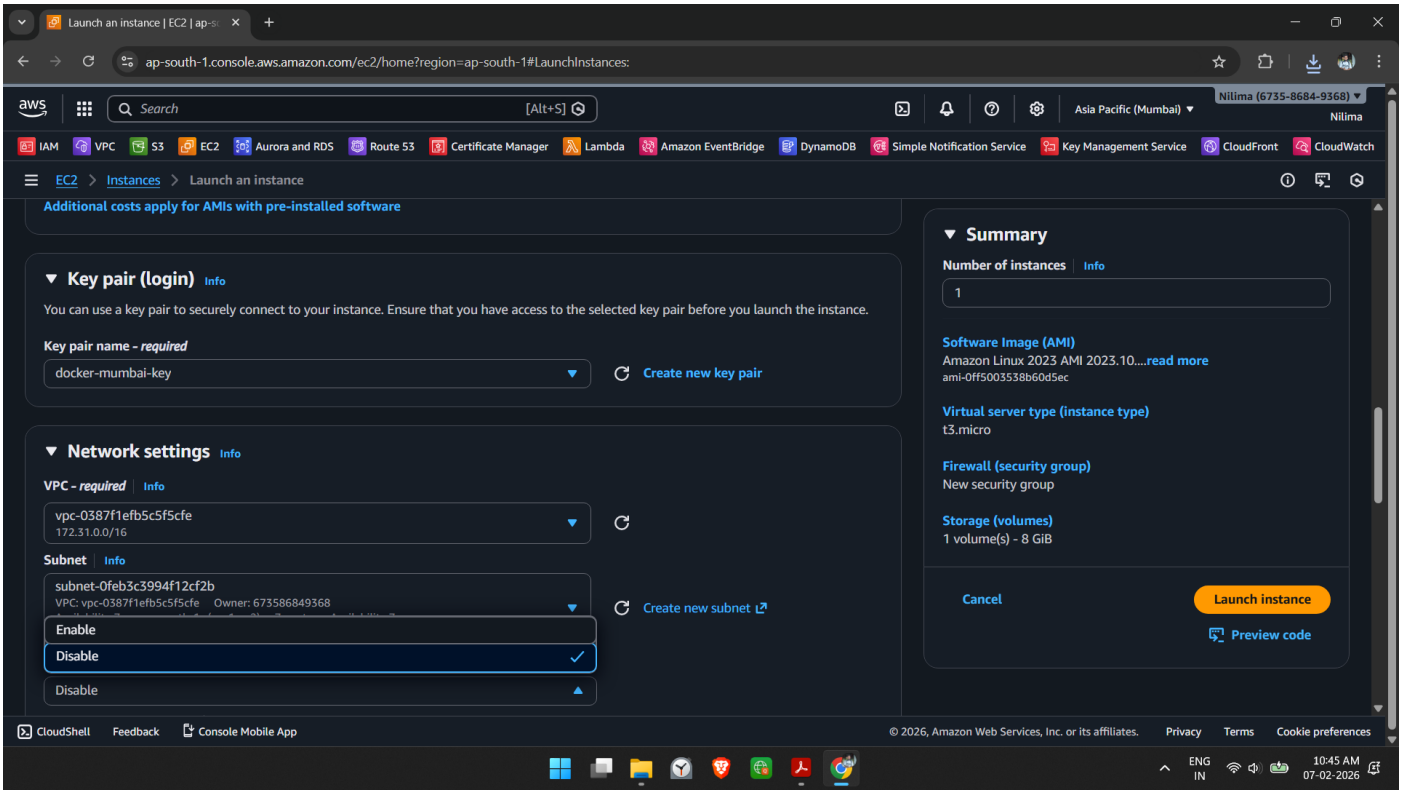
3. Select Amazon Linux 2023 AMI and choose an instance type (shown as t3.micro).



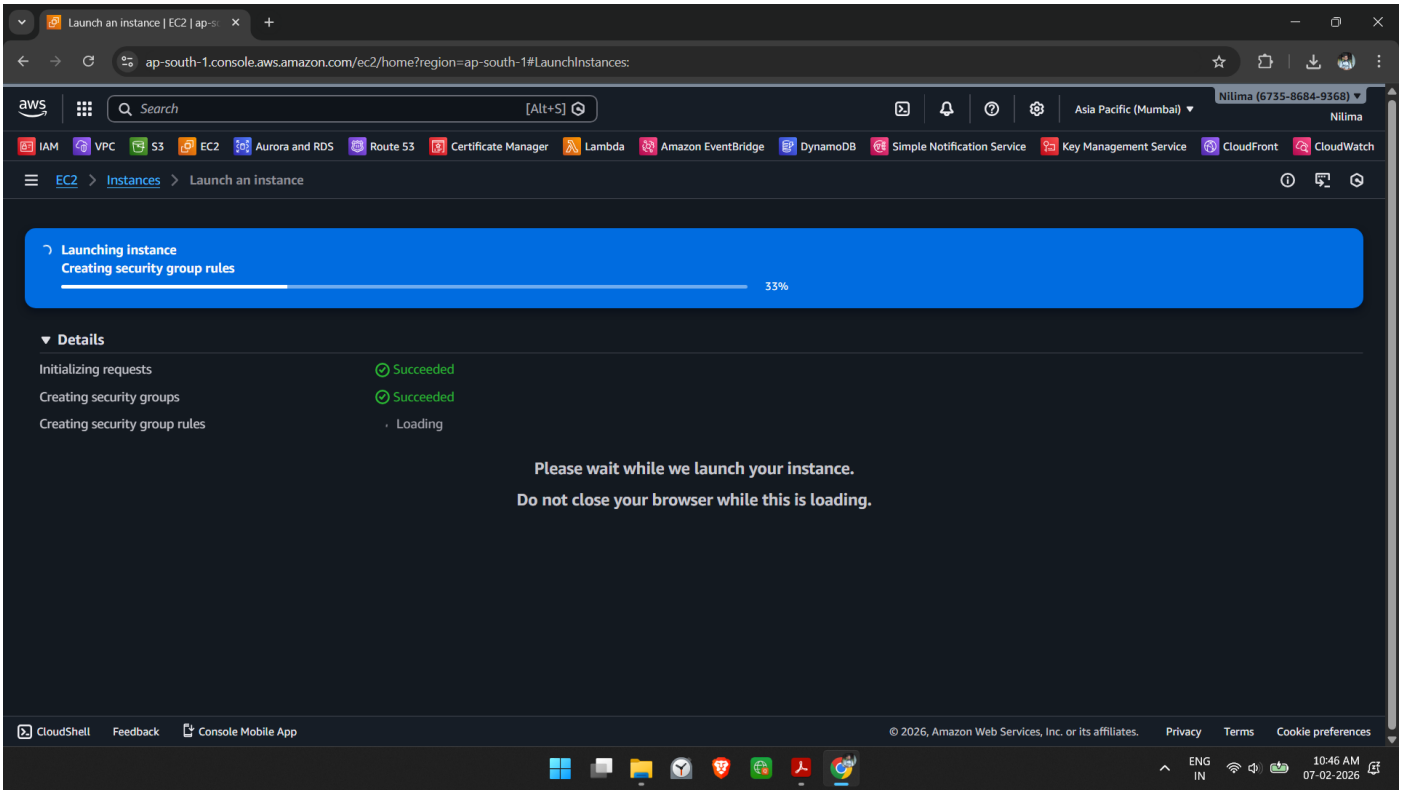
4. Create/select a key pair (shown as docker-mumbai-key) and download the .pem file.

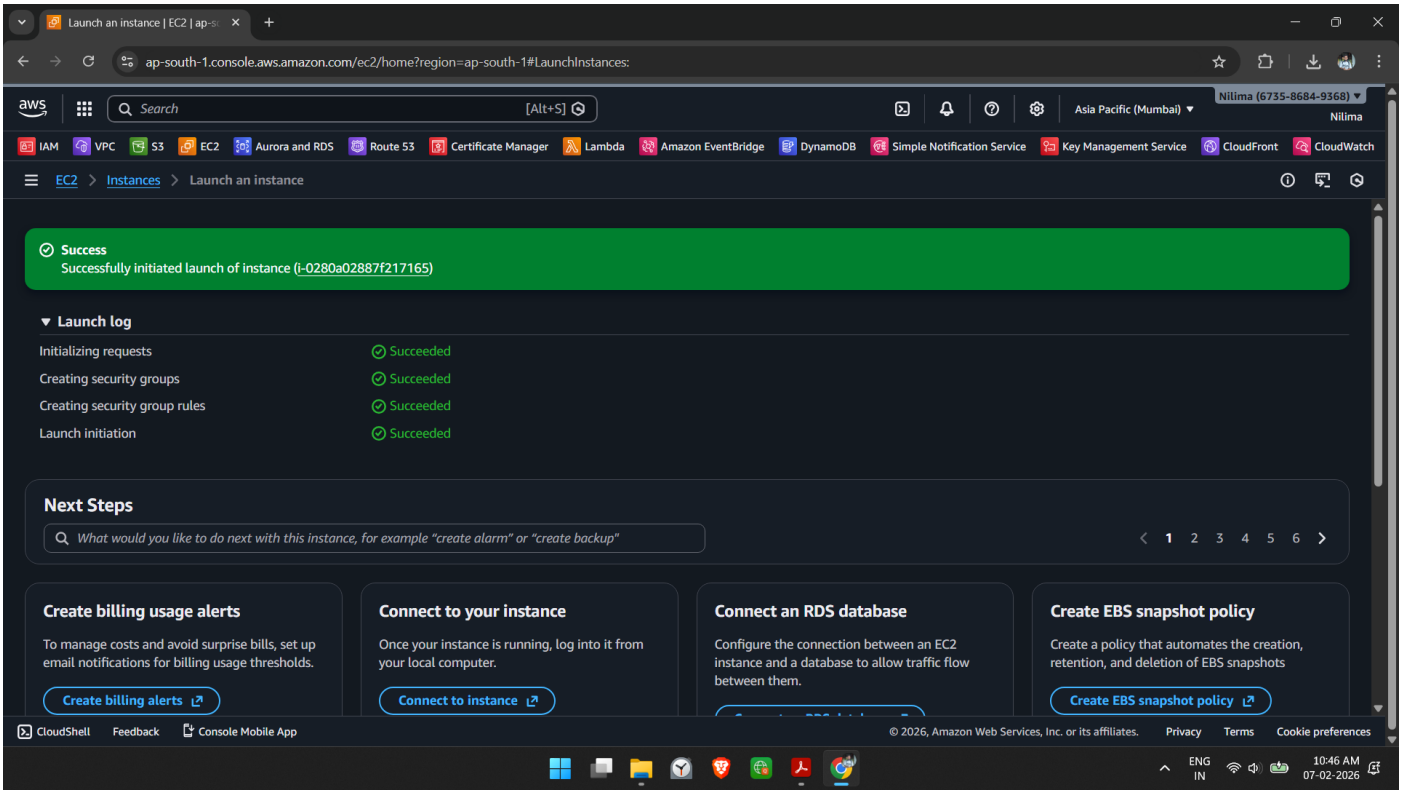


5. In Network settings, ensure the instance will receive a public IPv4 address (Auto-assign public IP must be enabled so the website can be reached from the internet).

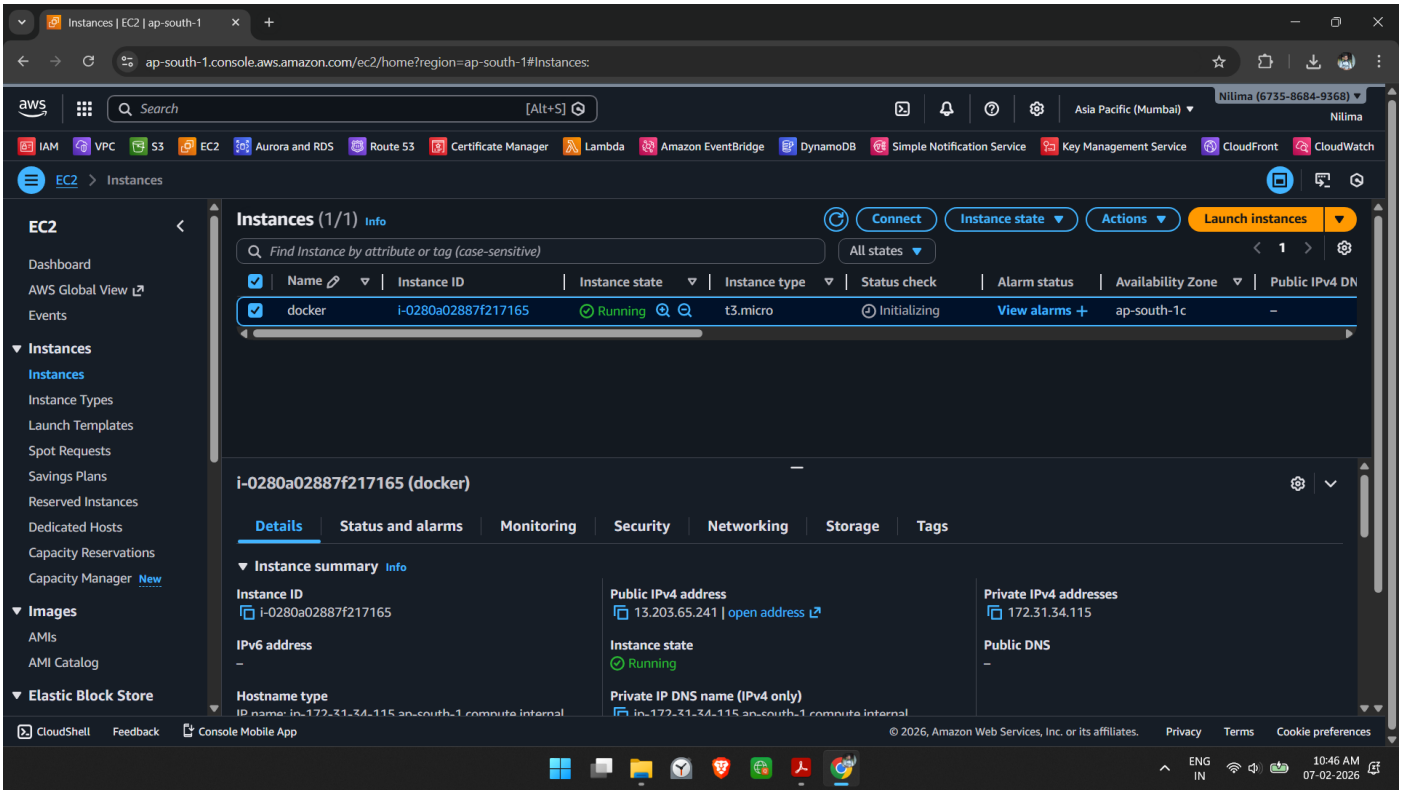


6. Click Launch instance and confirm the success message.



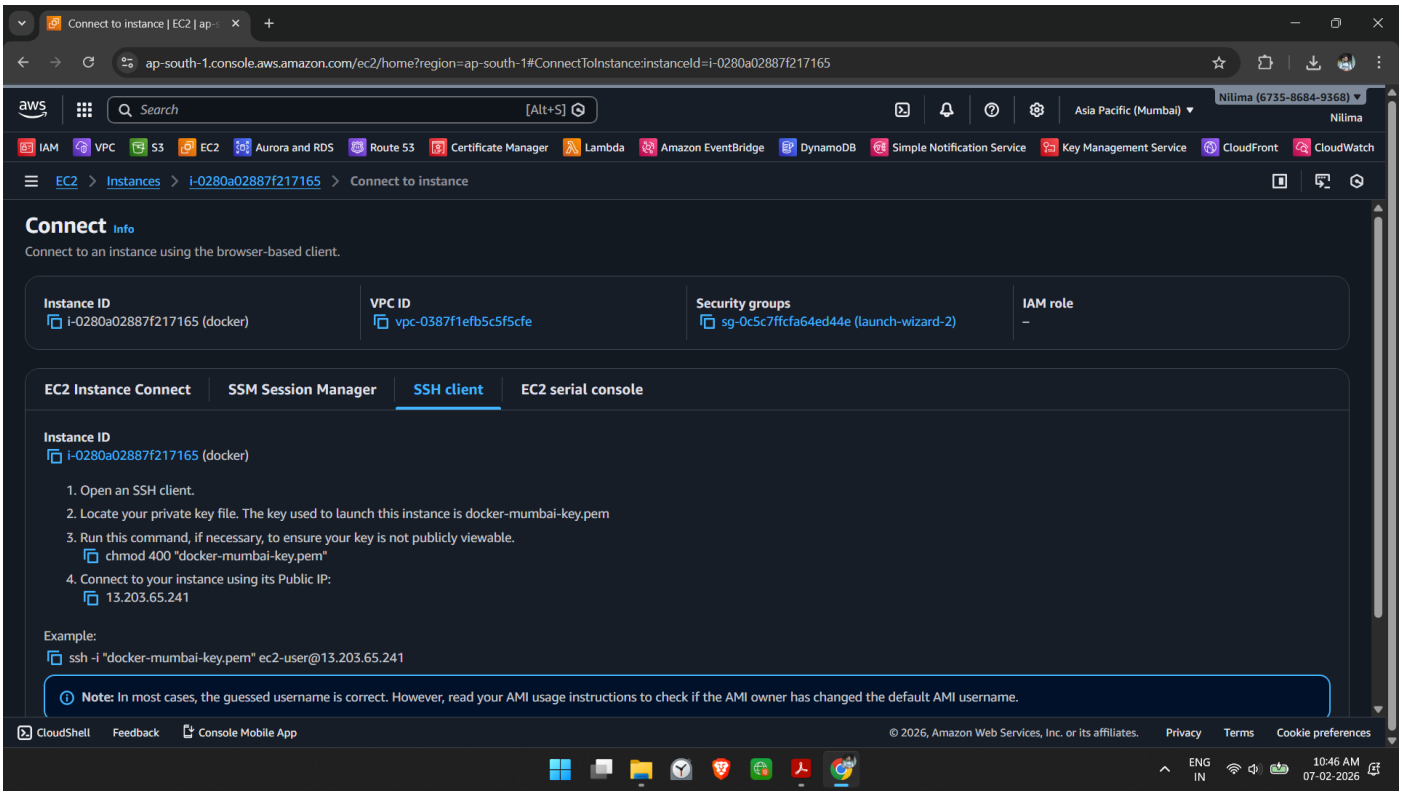


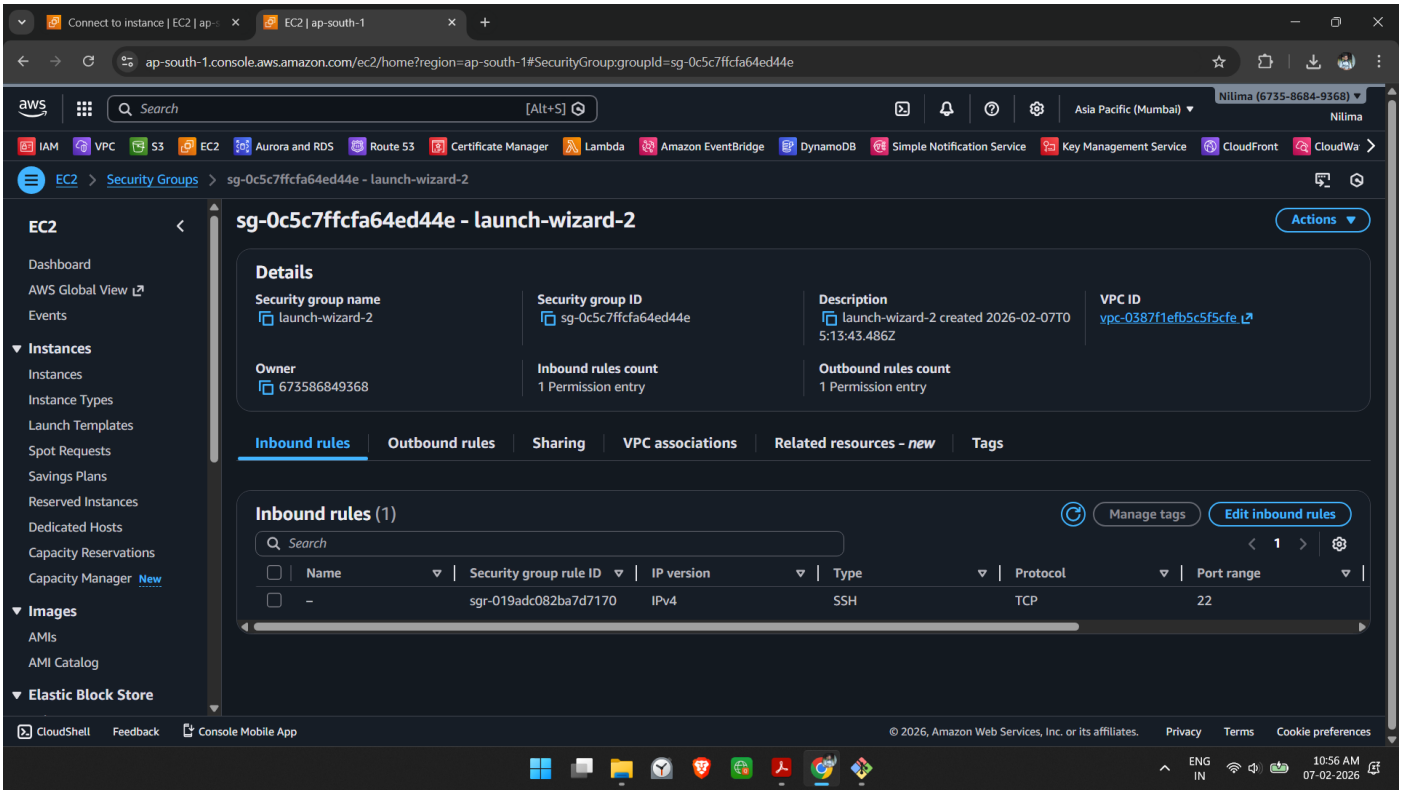
7. Open EC2 → Instances, select the instance, and copy the Public IPv4 address (shown as 13.203.65.241).



This public IP will be used later to test NGINX in the browser.

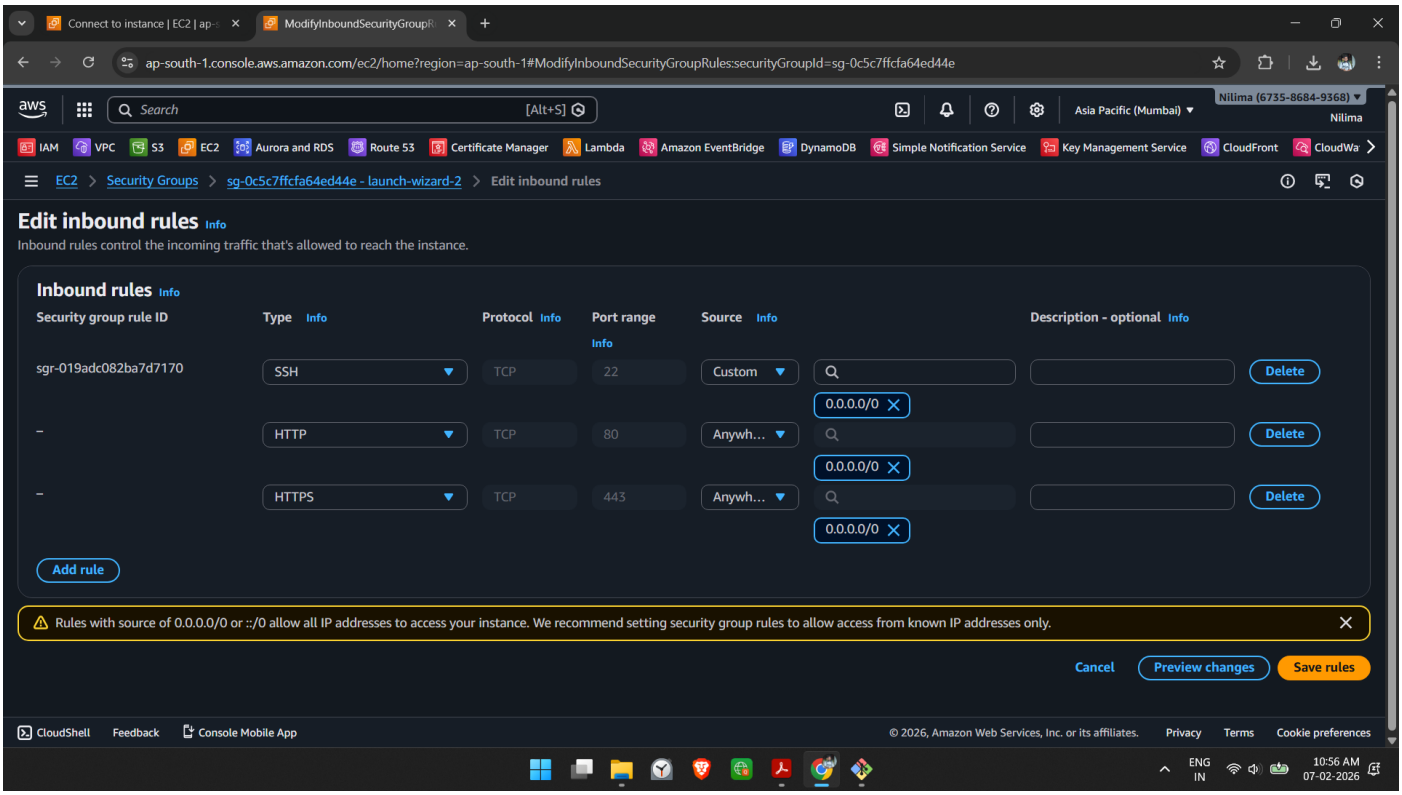
8. Open the instance's Security Group (shown as launch-wizard-2).





9. Edit inbound rules to allow:

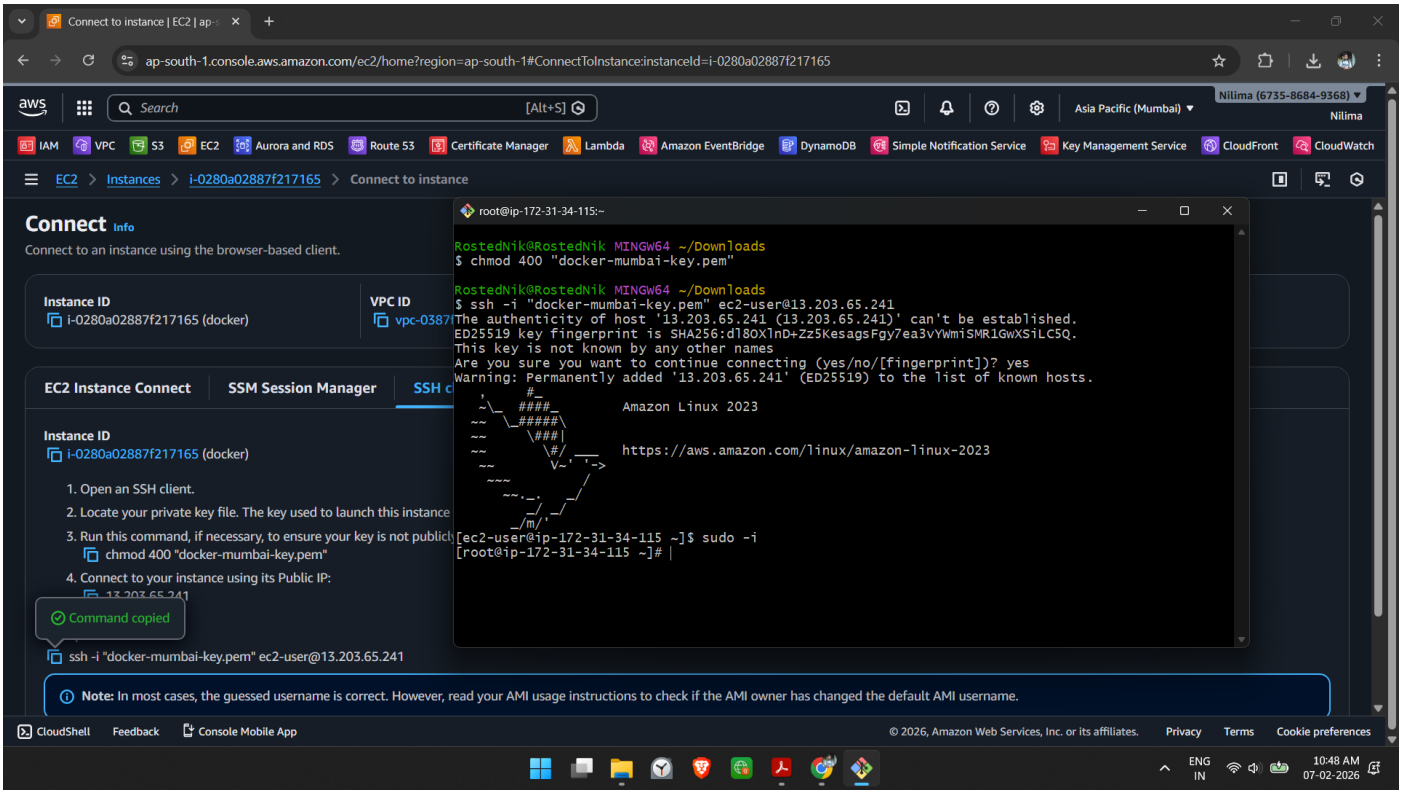
- SSH (22) for administration,
- HTTP (80) for the NGINX website,
- HTTPS (443) optional (as shown in rule edit screen).



Save the rules (your screenshots show sources set to 0.0.0.0/0, meaning open to the internet).

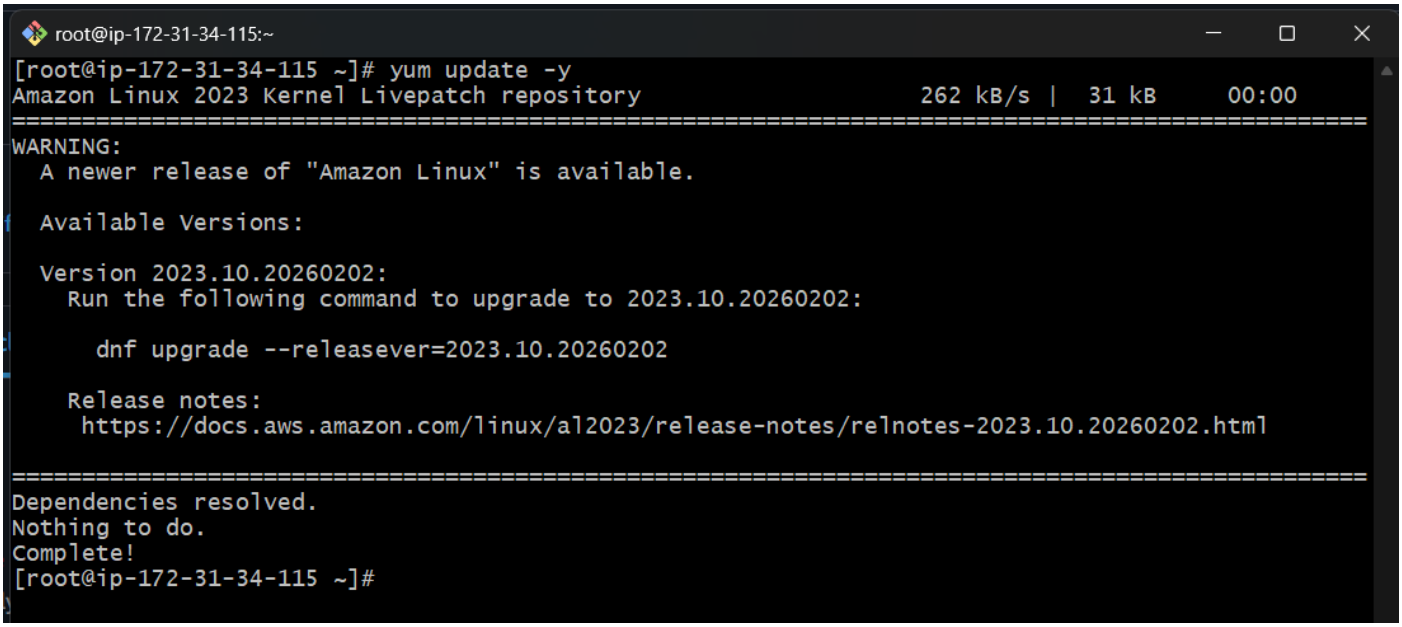
Note: Security groups are virtual firewalls that control inbound/outbound traffic to EC2, and allowing HTTP/HTTPS enables public web access.

10. From your local machine, set correct permissions on the key file:
    `chmod 400 docker-mumbai-key.pem`
11. Connect using the SSH command provided by the EC2 console (pattern shown in your screenshot):
    `ssh -i "docker-mumbai-key.pem" ec2-user@13.203.65.241`
12. Accept the host fingerprint prompt on first connection (the “yes” confirmation is shown).

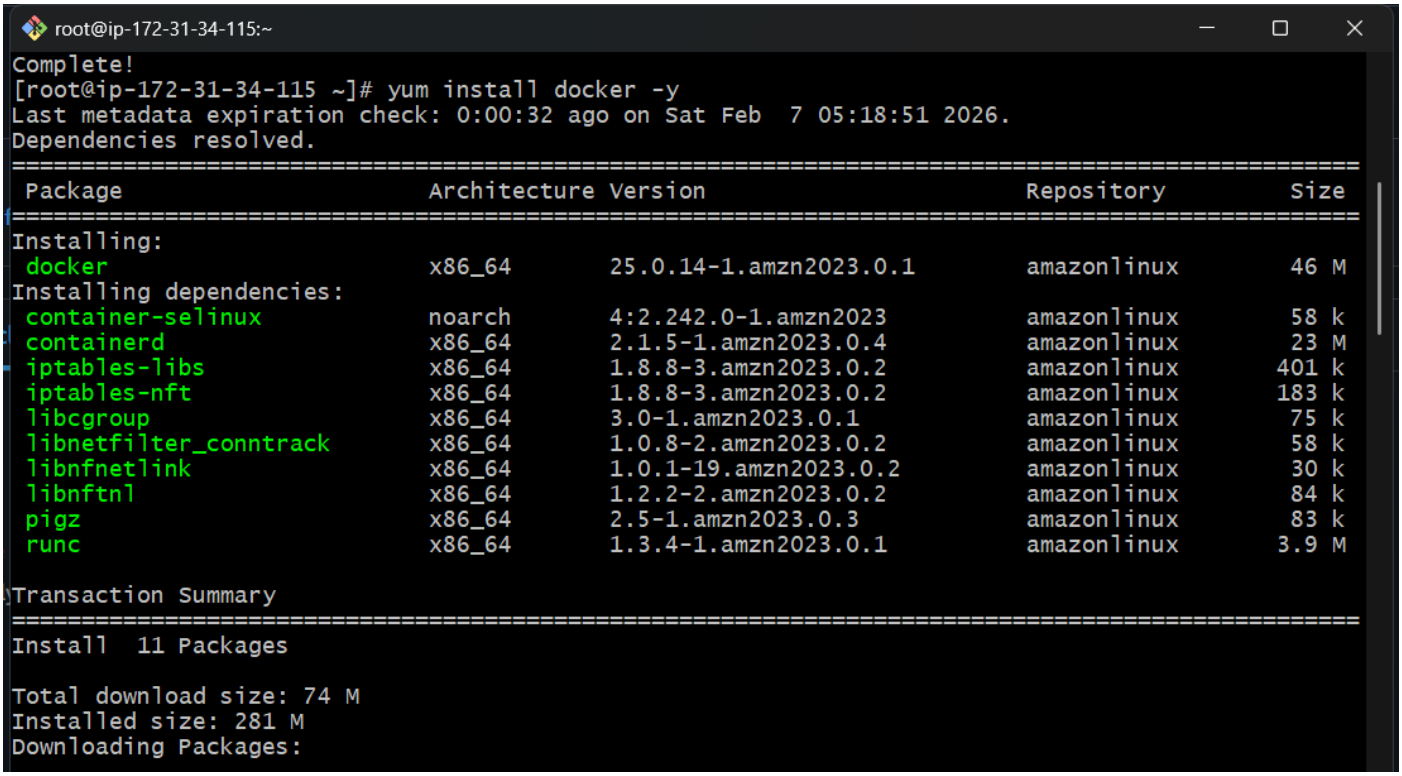


II) Install and start Docker

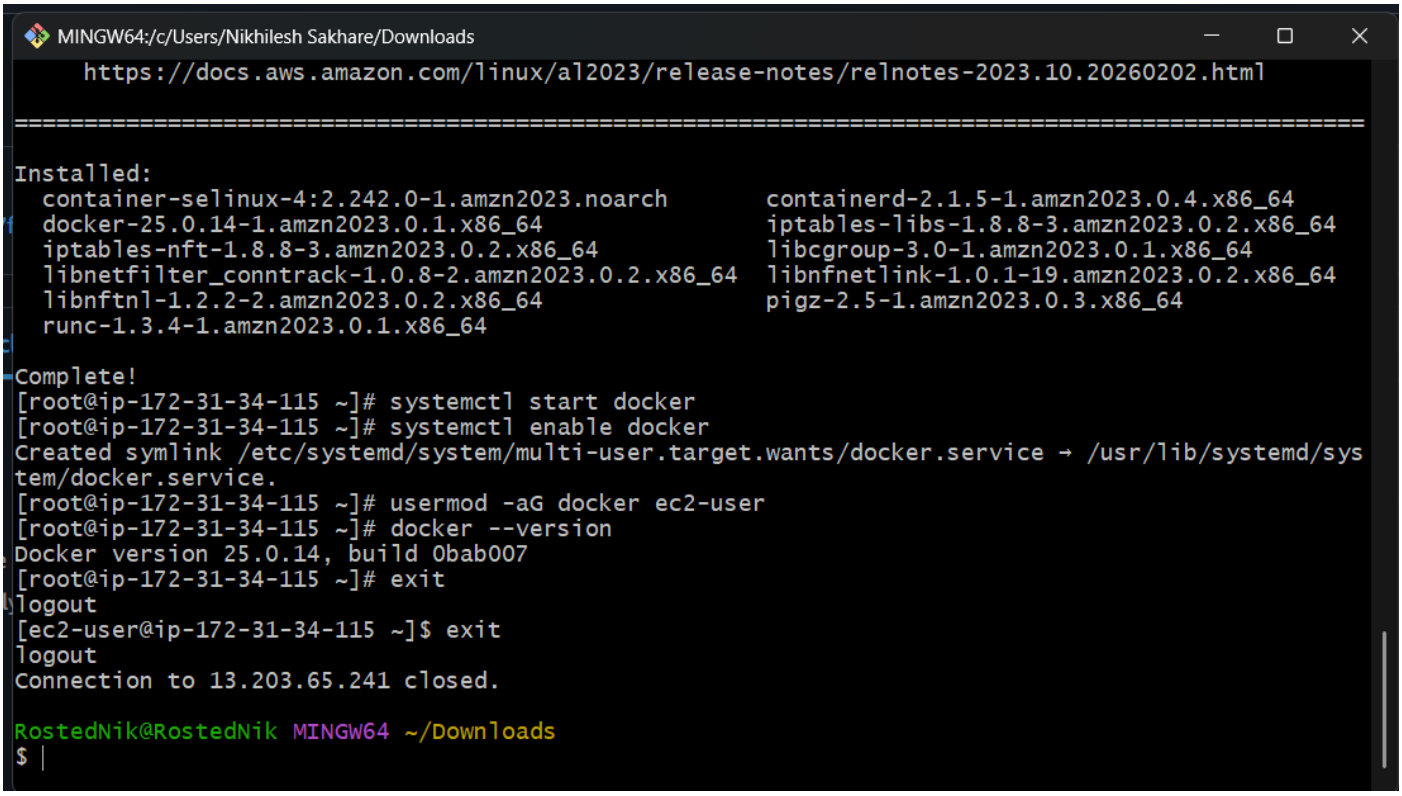
13. Switch to root (shown as `sudo -i`).
14. Update packages (shown as `yum update -y`).

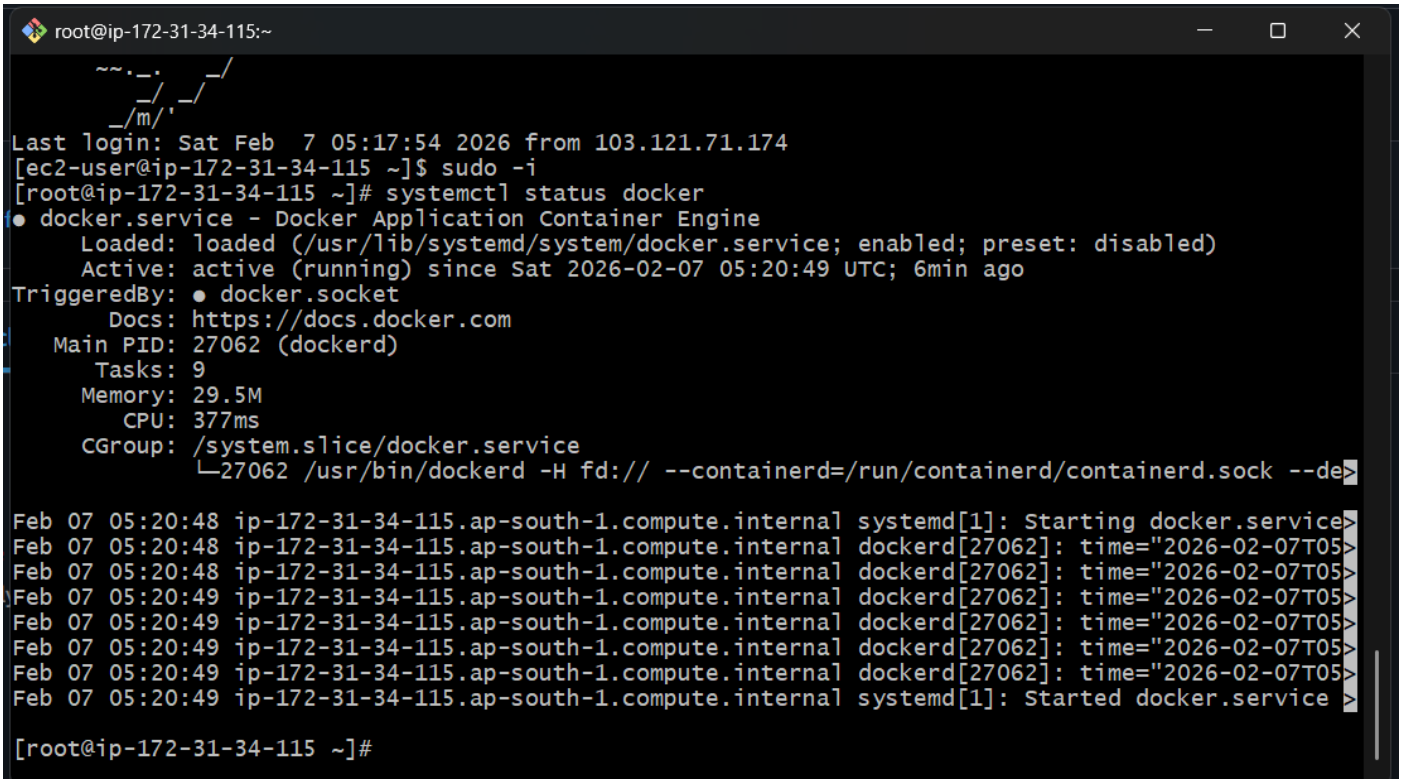


15. Install Docker (shown as `yum install docker -y`).



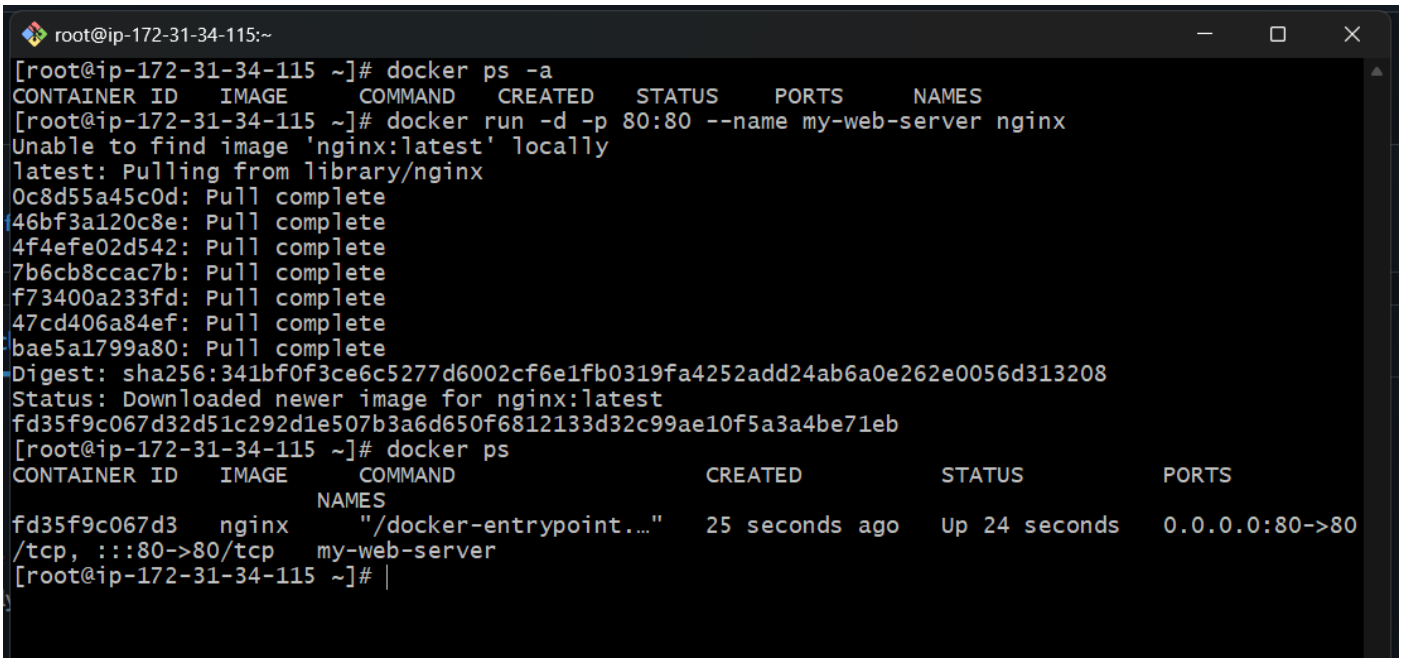
16. Start Docker and enable it on boot (shown as `systemctl start docker` and `systemctl enable docker`).
17. (Optional) Add user to the docker group (shown as `usermod -aG docker ec2-user`).
18. Verify Docker service and/or version (shown via `systemctl status docker` and `docker --version`).



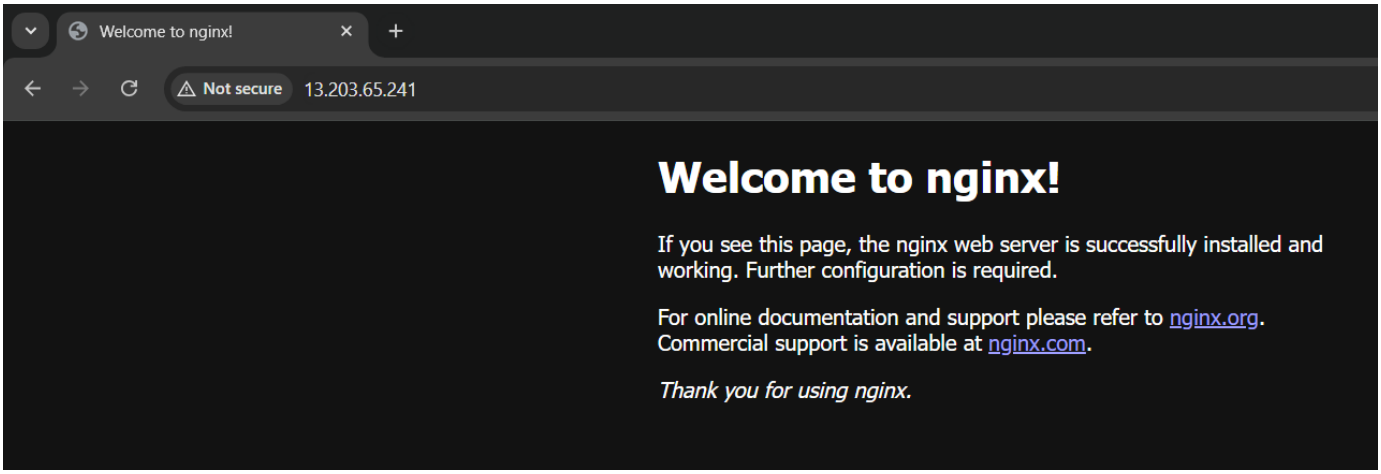


III) Run NGINX as a Docker container

19. List containers (shown as `docker ps -a`) to confirm current state.
20. Run the official NGINX image in detached mode and map host port 80 to container port 80:
    `docker run -d -p 80:80 --name my-web-server nginx`
21. Confirm the container is running and ports are mapped (shown by `docker ps` with `0.0.0.0:80->80/tcp`).



22. Open a browser and access the site using the instance public IP:
    `http://13.203.65.241`



The “Welcome to nginx!” default page confirms the NGINX container is serving traffic publicly.