**Docker challenge-1**

1. **Create a customized Docker image using a Dockerfile.**
2. **Push the image to Docker Hub.**
3. **Push the same image to Amazon ECR.**
4. **Provision one EC2 instance using Terraform and install Jenkins.**
5. **Create one Jenkins job to build and push the Docker image to Docker Hub.**
   * **Source:**[**https://github.com/betawins/Python-app.git**](https://github.com/betawins/Python-app.git)

**Source Codes:**[**https://github.com/betawins/docker-tasks.git**](https://github.com/betawins/docker-tasks.git)

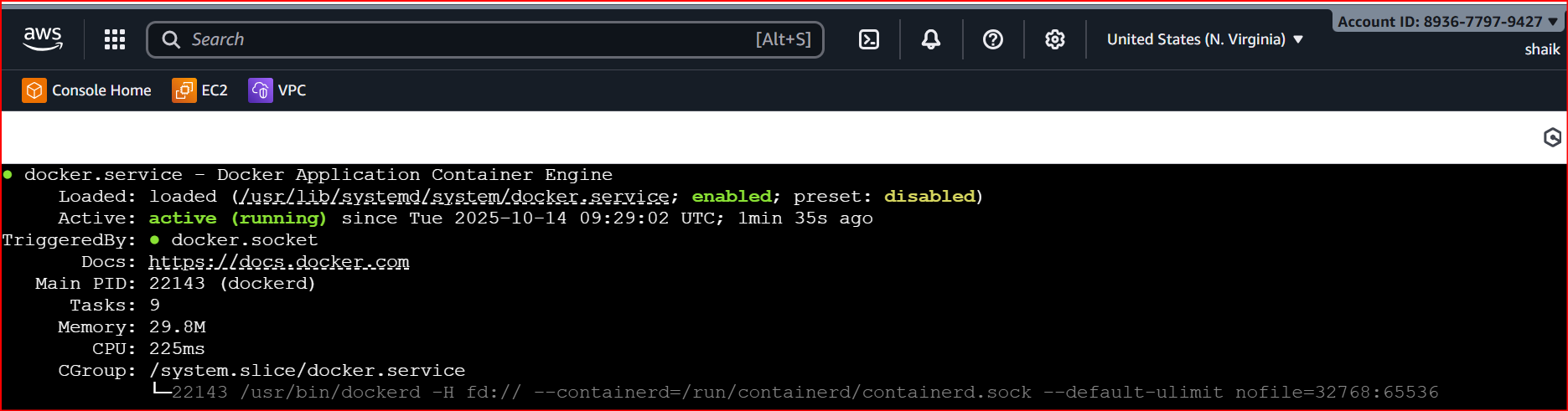
* **From the frontend source code, write a Dockerfile, build a Docker image, run it, and push that image to your Docker registry.**
* **From the Java-based source code, write a Dockerfile, build, run, and push the image to the Docker registry.**
* **From the Node.js-based source code, write a Dockerfile, build with tag v1, run, and push it to the Docker registry.**
* **Write a docker-compose file to set up WordPress with a MySQL database**

**1.Install docker**

yum install docker -y

sudo systemctl start docker

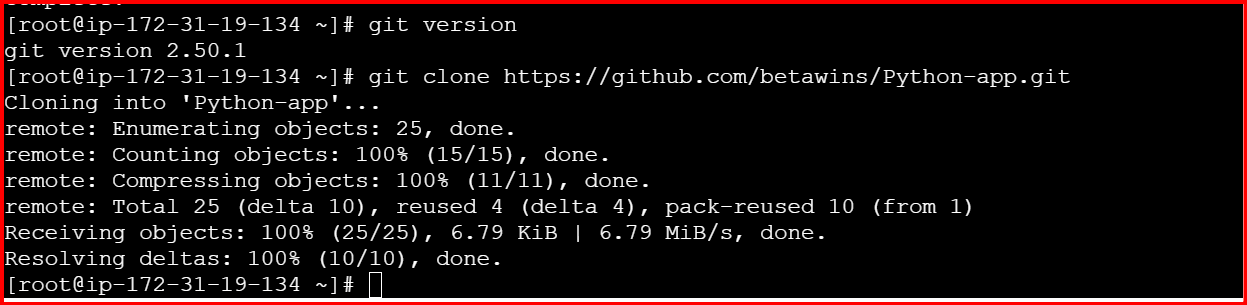
sudo systemctl enable docker



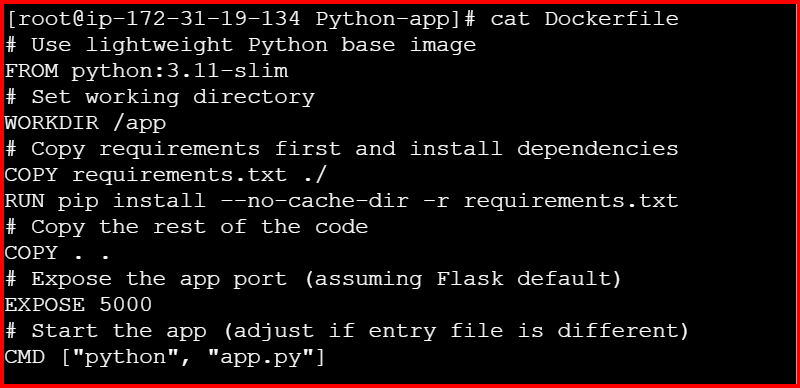
. sudo usermod -aG docker $USER

Git clone

git clone https://github.com/betawins/Python-app.git



Write a docker file



**Build the Docker image**

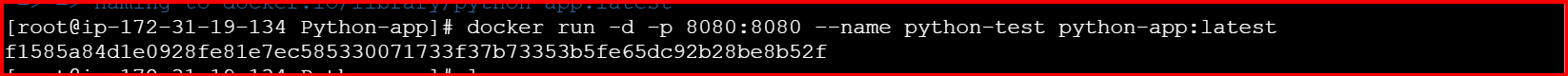
docker build -t python-app:latest .

**Run the container to test**

docker run -d -p 8080:8080 --name python-test python-app:latest

**Open browser → http://localhost:8080**





<http://54.175.209.118:8080/>



**2 Push the image to Docker Hub.**

1. Create a Docker Hub account (if not already)

https://hub.docker.com

Create a repo there, e.g. python-app.

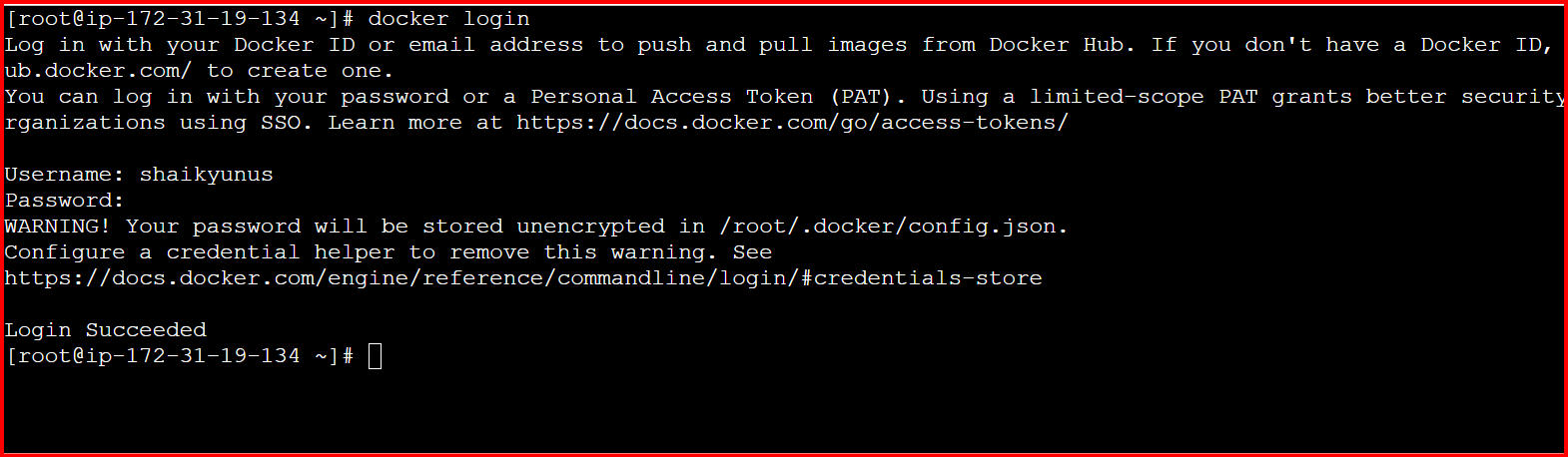
2. Log in to Docker Hub from EC2

docker login

It will ask for:

● Username → your Docker Hub username

● Password → your Docker Hub password or access token



3. Tag your image with Docker Hub repo name

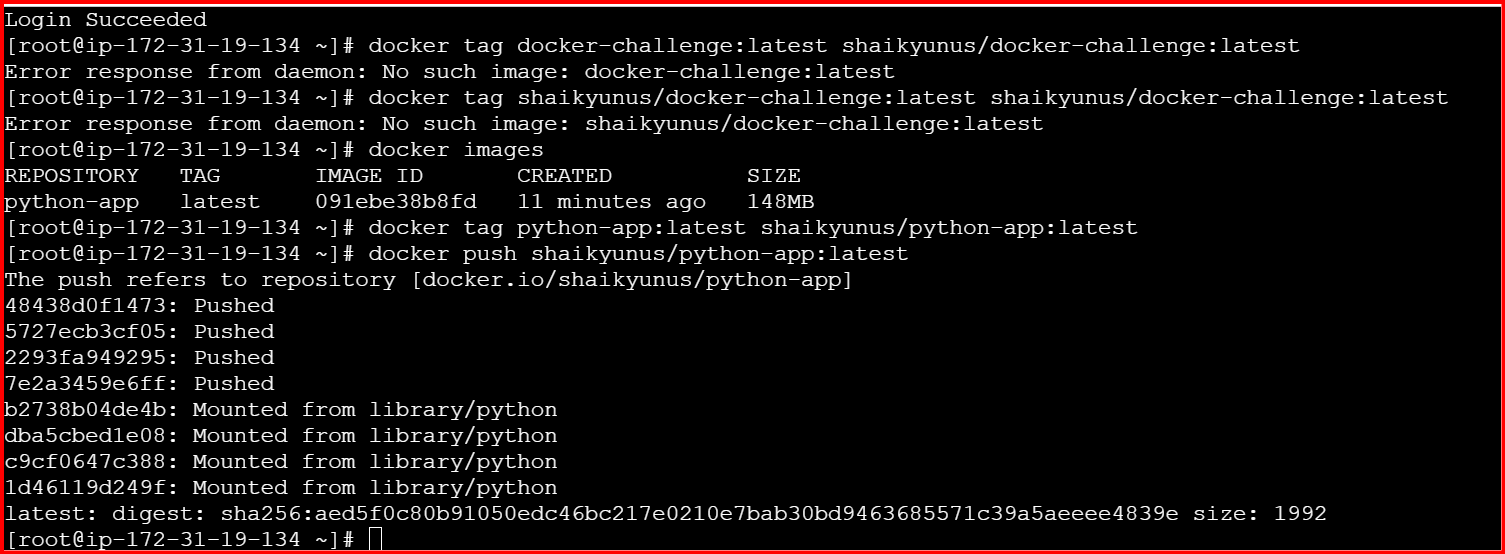
docker tag python-app:latest your-username/python-app:latest

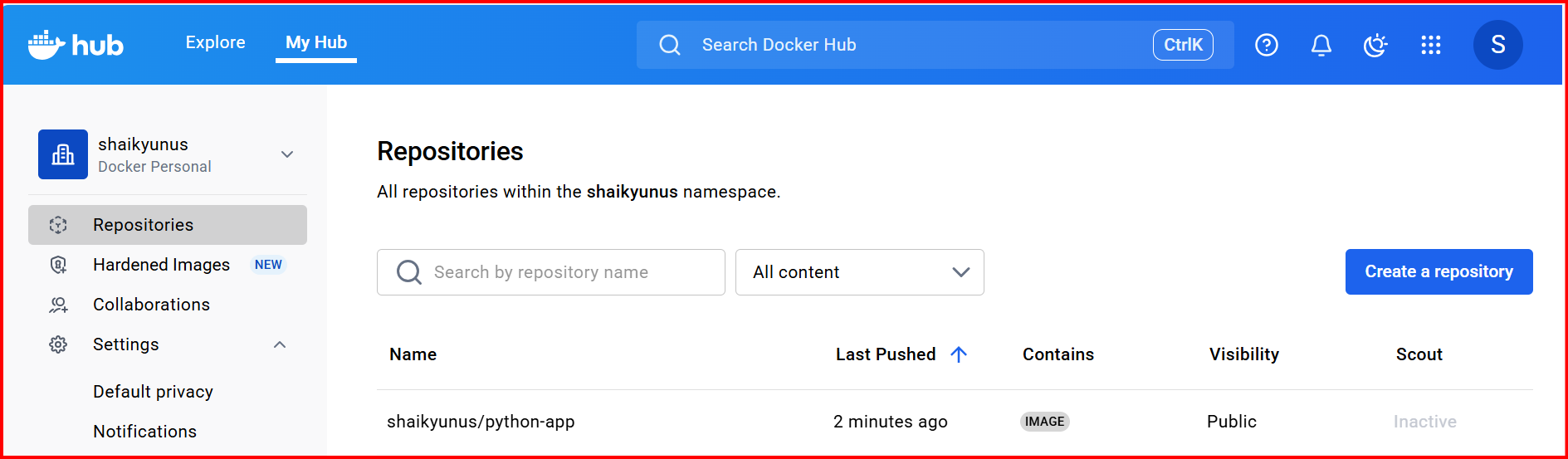
4. Push to Docker Hub

docker push your-username/python-app:latest

5. Verify

Go to your Docker Hub repo in browser → you should see python-app:latest image uploaded.



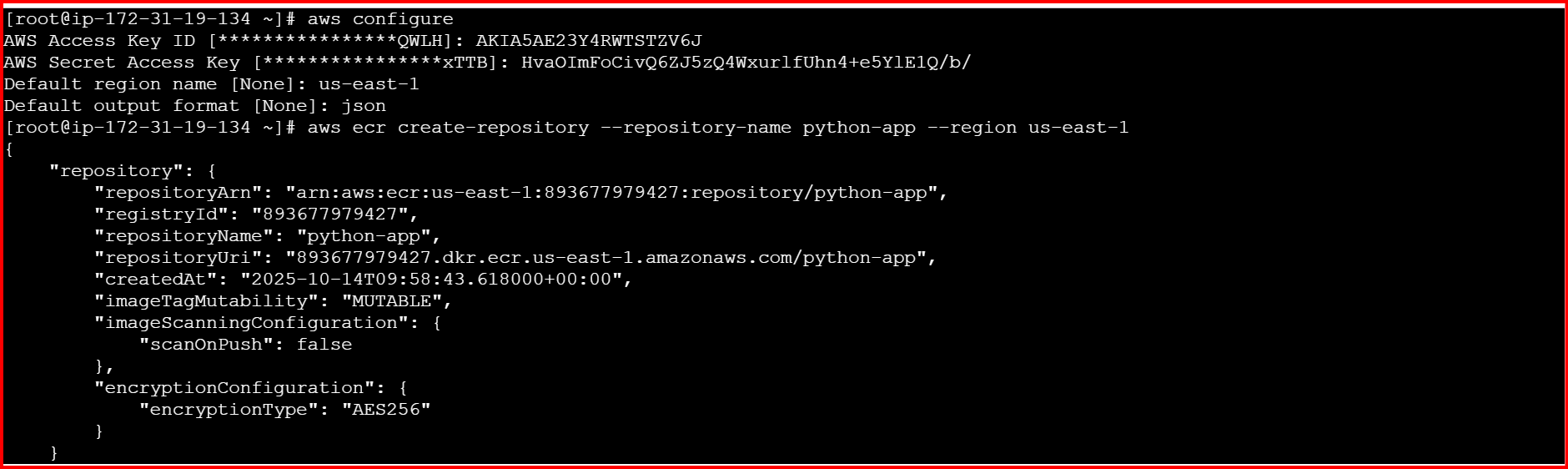
****

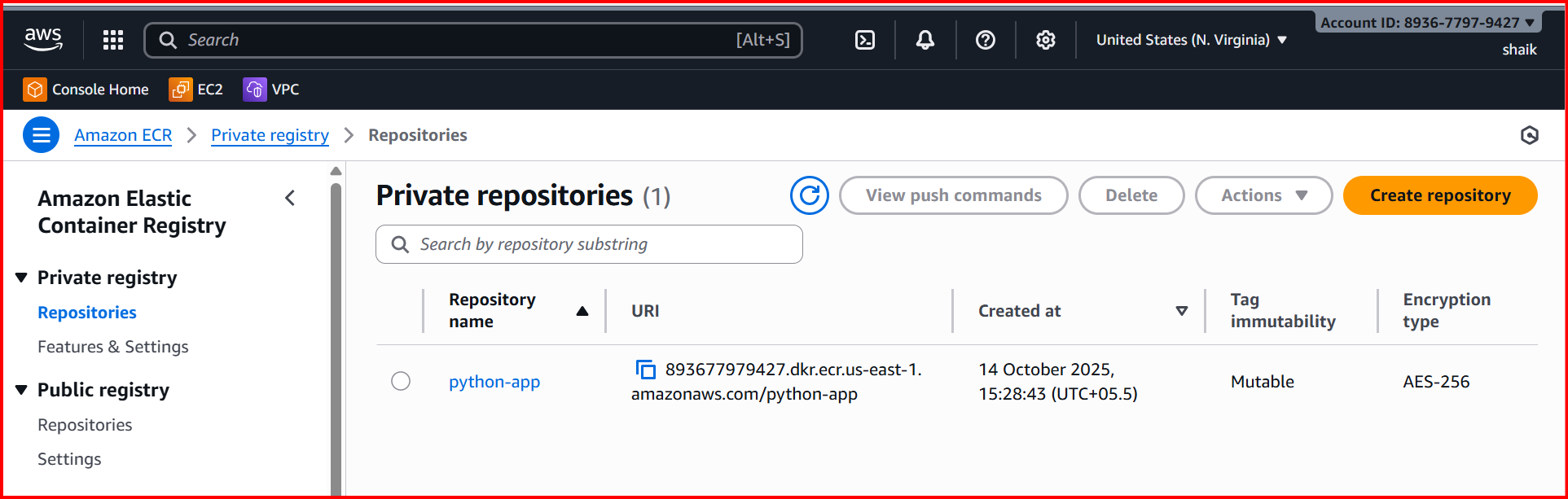
**3.Push the same image to Amazon ECR.**

Configure the aws

**Create the repository in AWS**

aws ecr create-repository --repository-name python-app --region us-east-1

****

****

**Authenticate Docker to ECR**

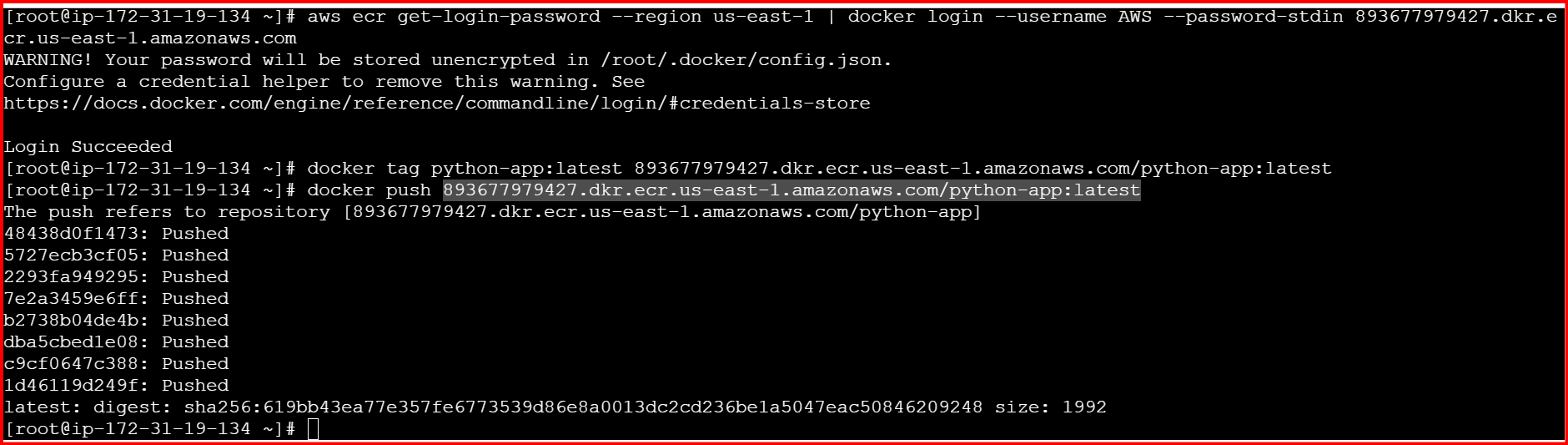
aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin <AWS\_ACCOUNT\_ID>.dkr.ecr.us-east-1.amazonaws.com4. **Tag your Docker image for ECR**

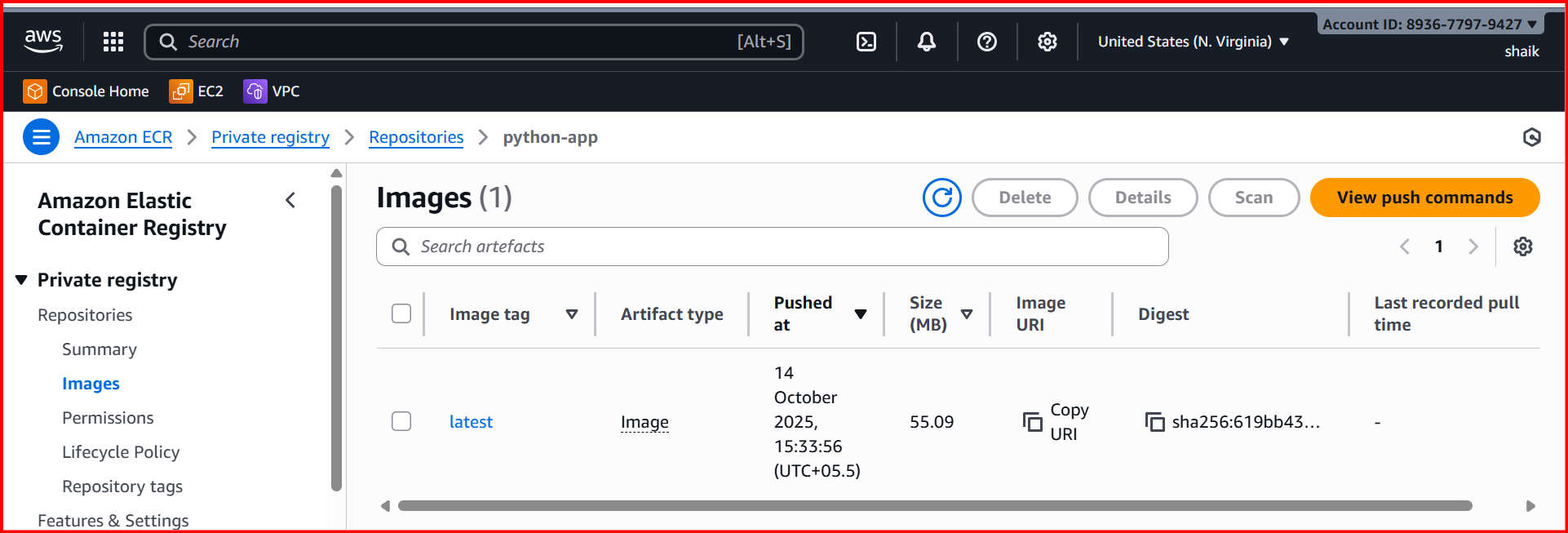
docker tag python-app:latest <AWS\_ACCOUNT\_ID>.dkr.ecr.us-east1.amazonaws.com/python-app:latest

**Push image to ECR**

docker push <AWS\_ACCOUNT\_ID>.dkr.ecr.us-east-1.amazonaws.com/python-app:latest

**Go to AWS Console → ECR → Repositories → python-app**

****

****

**4.Provision one EC2 instance using Terraform and install Jenkins.**

**sudo yum install -y yum-utils**

**sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo**

**sudo yum -y install terraform**

**variables.tf**

**variable "ami\_id" {**

**description = "AMI ID for the EC2 instance"**

**type = string**

**}**

**variable "key\_name" {**

**description = "Key pair name for SSH access"**

**type = string**

**}**

**variable "my\_ip" {**

**description = "Your public IP address for security group ingress"**

**type = string**

**}**

**variable "region" {**

**description = "AWS region to deploy resources"**

**type = string**

**default = "us-east-1"**

**}**

**main.tf**

**provider "aws" {**

**region = var.region**

**}**

**# VPC**

**resource "aws\_vpc" "main" {**

**cidr\_block = "10.0.0.0/16"**

**enable\_dns\_hostnames = true**

**enable\_dns\_support = true**

**tags = { Name = "jenkins-vpc" }**

**}**

**# Subnet**

**resource "aws\_subnet" "public" {**

**vpc\_id = aws\_vpc.main.id**

**cidr\_block = "10.0.1.0/24"**

**availability\_zone = "us-east-1a"**

**map\_public\_ip\_on\_launch = true**

**tags = { Name = "jenkins-subnet" }**

**}**

**# Internet Gateway**

**resource "aws\_internet\_gateway" "igw" {**

**vpc\_id = aws\_vpc.main.id**

**tags = { Name = "jenkins-igw" }**

**}**

**# Route Table**

**resource "aws\_route\_table" "rt" {**

**vpc\_id = aws\_vpc.main.id**

**route {**

**cidr\_block = "0.0.0.0/0"**

**gateway\_id = aws\_internet\_gateway.igw.id**

**}**

**tags = { Name = "jenkins-rt" }**

**}**

**resource "aws\_route\_table\_association" "rta" {**

**subnet\_id = aws\_subnet.public.id**

**route\_table\_id = aws\_route\_table.rt.id**

**}**

**# Security Group**

**resource "aws\_security\_group" "jenkins\_sg" {**

**name = "jenkins-sg"**

**description = "Allow SSH, HTTP, and Jenkins port"**

**vpc\_id = aws\_vpc.main.id**

**ingress {**

**description = "SSH"**

**from\_port = 22**

**to\_port = 22**

**protocol = "tcp"**

**cidr\_blocks = [var.my\_ip]**

**}**

**ingress {**

**description = "Jenkins"**

**from\_port = 8080**

**to\_port = 8080**

**protocol = "tcp"**

**cidr\_blocks = ["0.0.0.0/0"]**

**}**

**ingress {**

**description = "HTTP"**

**from\_port = 80**

**to\_port = 80**

**protocol = "tcp"**

**cidr\_blocks = ["0.0.0.0/0"]**

**}**

**egress {**

**description = "All traffic out"**

**from\_port = 0**

**to\_port = 0**

**protocol = "-1"**

**cidr\_blocks = ["0.0.0.0/0"]**

**}**

**}**

**# EC2 Instance with Jenkins**

**resource "aws\_instance" "jenkins" {**

**ami = var.ami\_id**

**instance\_type = "t3.micro"**

**subnet\_id = aws\_subnet.public.id**

**vpc\_security\_group\_ids = [aws\_security\_group.jenkins\_sg.id]**

**key\_name = var.key\_name**

**associate\_public\_ip\_address = true**

**user\_data = <<-EOF**

**#!/bin/bash**

**set -e**

**yum update -y**

**yum install -y wget git**

**dnf install -y java-17-amazon-corretto**

**wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat/jenkins.repo**

**rpm --import https://pkg.jenkins.io/redhat/jenkins.io-2023.key**

**yum upgrade -y**

**yum install -y jenkins**

**systemctl enable jenkins**

**systemctl start jenkins**

**EOF**

**tags = { Name = "jenkins-server" }**

**}**

**# Output public IP**

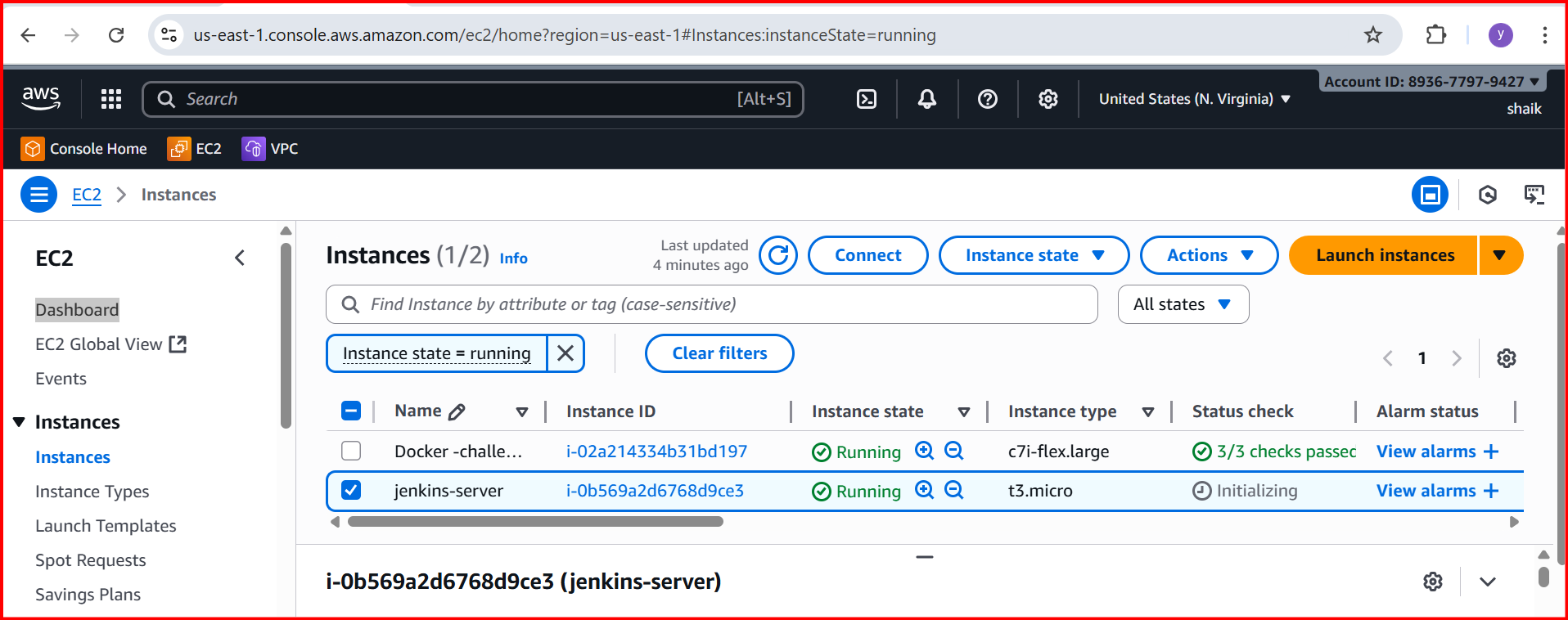
**output "jenkins\_public\_ip" {**

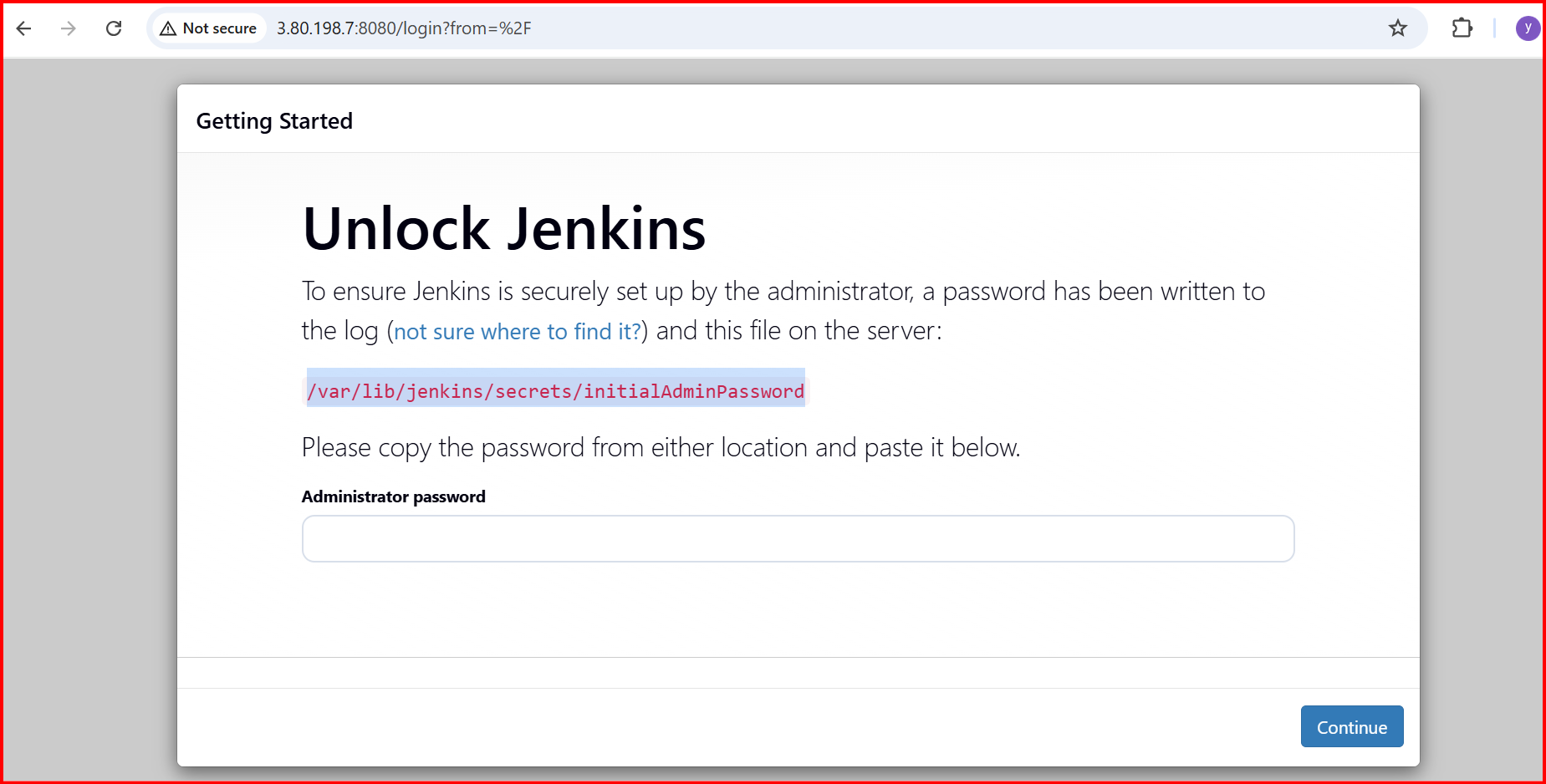
**value = aws\_instance.jenkins.public\_ip**

**}**

**terraform init**

**terraform apply**

****

****

**5.Create one Jenkins job to build and push the Docker image to Docker Hub.**

* + **Source:**[**https://github.com/betawins/Python-app.git**](https://github.com/betawins/Python-app.git)

**Source Codes:**[**https://github.com/betawins/docker-tasks.git**](https://github.com/betawins/docker-tasks.git)

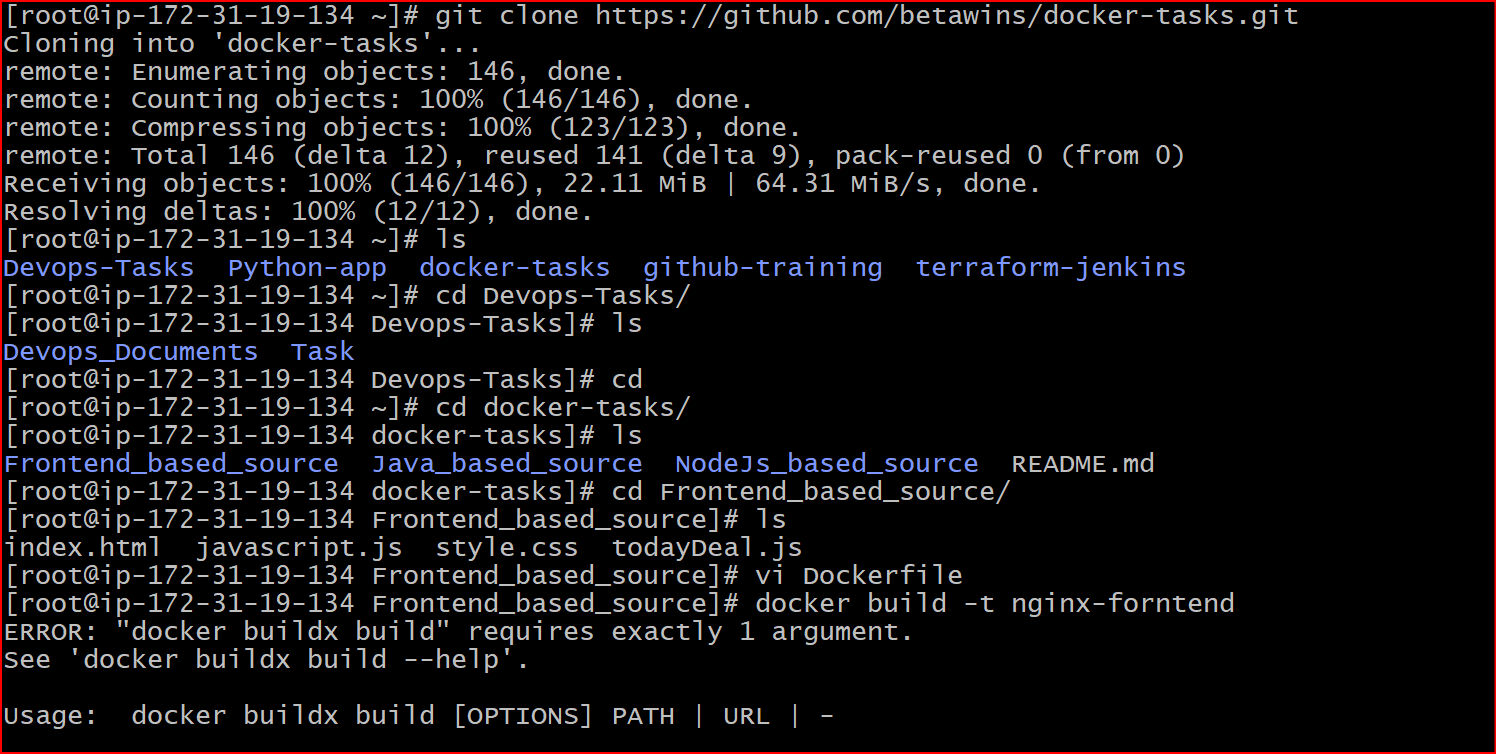
**Install Jenkins**

**Install plugins: Docker pipeline & pipeline stage**

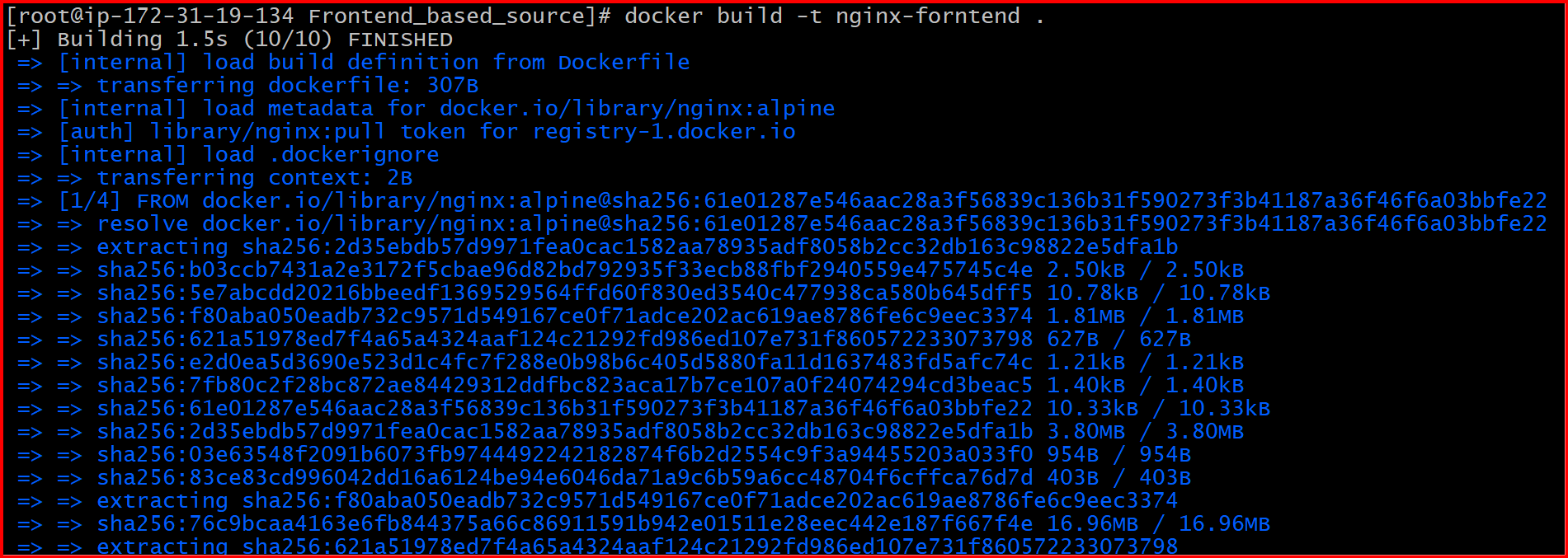
**Git clone :** [**https://github.com/betawins/docker-tasks.git**](https://github.com/betawins/docker-tasks.git)

**Cd Devops-Task**

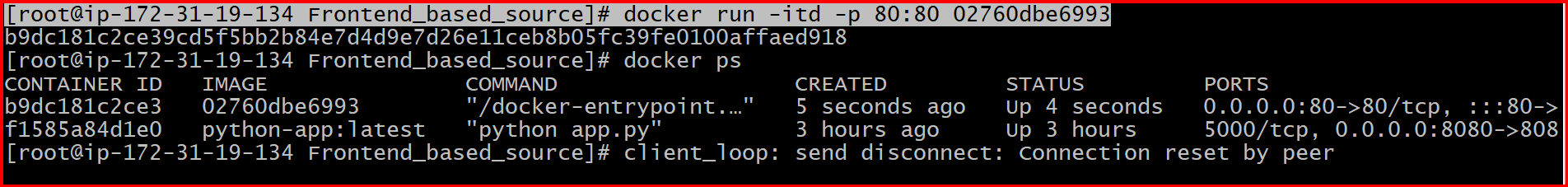
**Cd Fronted-based-source**

****

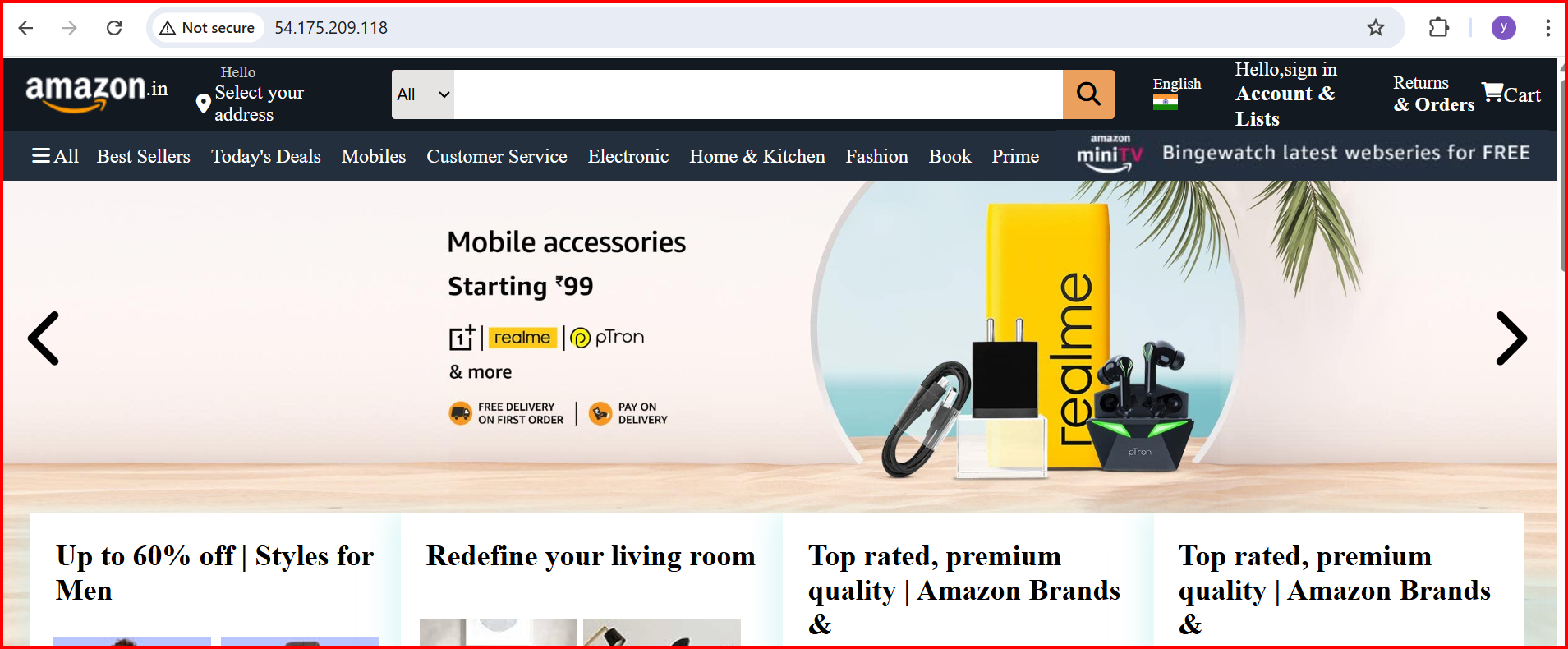
**Docker build -t nginx-forntend .**

****

**Docker run -itd -p 80:80 imageid**

****

[**http://54.175.209.118:80**](http://54.175.209.118:80)

****

* **From the Java-based source code, write a Dockerfile, build, run, and push the image to the Docker registry.**