**Author:-Dinesh.C.Patel**

**Infra Support Engineer (RHCSA,RHCE,AWS,DO188 & HashiCorp Terraform Certified)[8766924468]**

**Email:-dineshcpatel2727@gmail.com**

**\*\*Step-by-step guide for installing Terraform on both Windows and Linux (RHEL/Ubuntu/Debian based distros).\*\***

Terraform is an **open-source Infrastructure as Code (IaC) tool** created by **HashiCorp**.  
It allows you to **define, provision, and manage infrastructure** (servers, networks, databases, cloud services, etc.) using simple configuration files written in **HCL (HashiCorp Configuration Language)**.

* **Key Points about Terraform**
* **Infrastructure as Code (IaC):** You write .tf files to describe infrastructure instead of manually creating it in the cloud.
* **Cloud-agnostic:** Works with multiple providers (AWS, Azure, GCP, VMware, Kubernetes, etc.).
* **Declarative:** You declare *what* you want, and Terraform figures out *how* to create it.
* **Idempotent:** Running the same code multiple times will not create duplicates—it ensures infrastructure matches your code.
* **Execution Plan:** Terraform shows what changes it will make (terraform plan) before actually applying them (terraform apply).
* **Terraform Workflow**

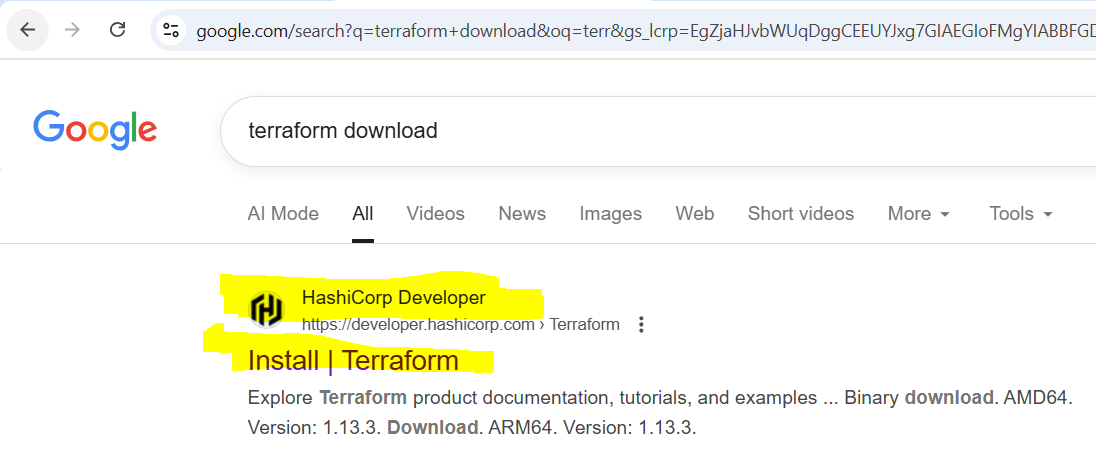
1. **Write** → Define infrastructure in .tf files.
2. **Init** → Initialize providers & plugins (terraform init).
3. **Plan** → Preview what Terraform will do (terraform plan).
4. **Apply** → Deploy the infrastructure (terraform apply).
5. **Destroy** → Tear down infrastructure when no longer needed (terraform destroy).

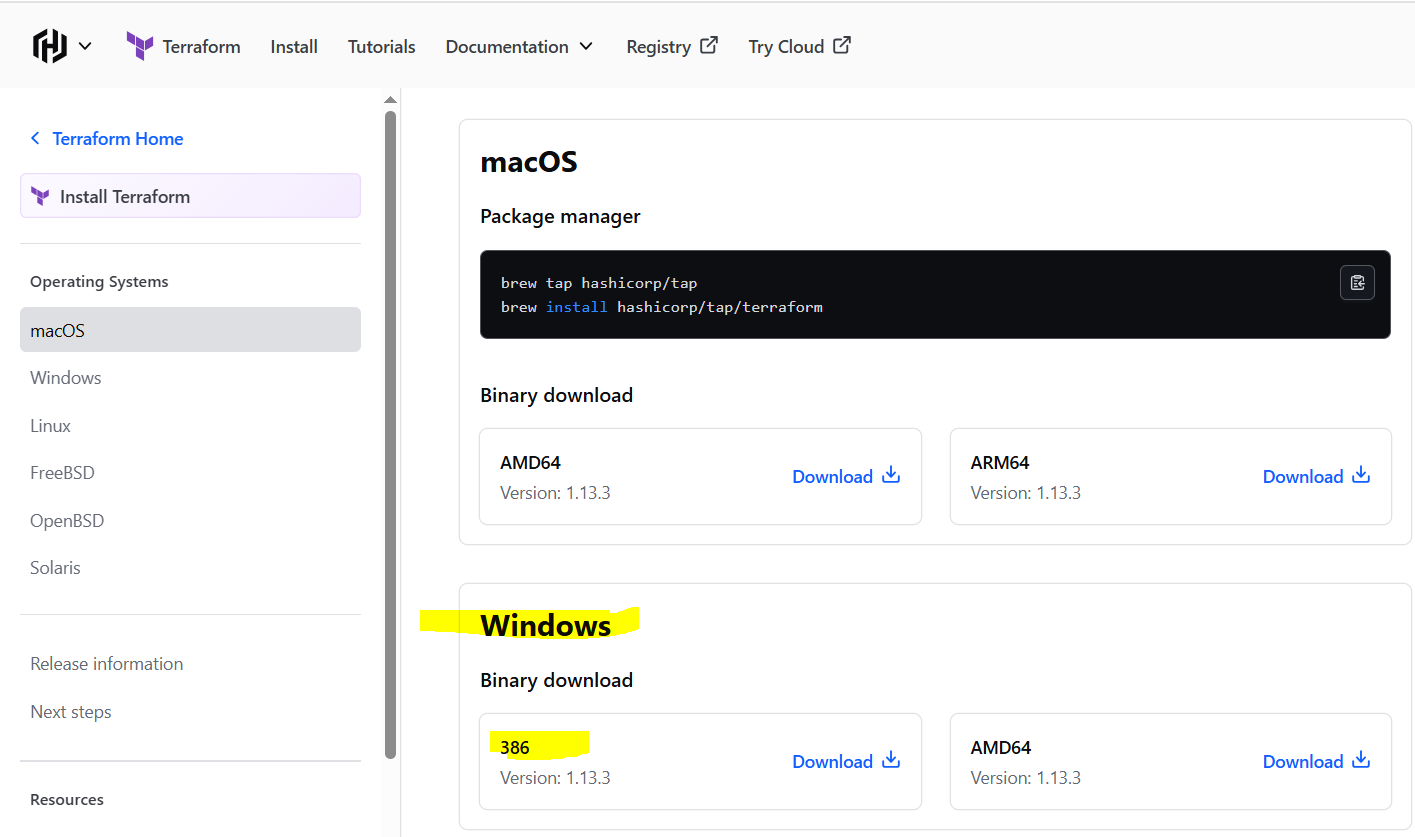
* **Difference Between Terraform vs Ansible:-**

| Feature | Terraform | Ansible |
| --- | --- | --- |
| Type | Infrastructure Provisioning Tool (IaC) | Configuration Management & Automation Tool |
| Main Purpose | To create, change, and destroy infrastructure (servers, networks, storage, cloud services, etc.) | To configure, deploy applications, and manage servers |
| Approach | Declarative – You define what you want (desired state), Terraform figures out how to get there. | Procedural (mostly) – You define step by step tasks to reach the state. |
| Execution Model | Builds an execution plan (terraform plan → terraform apply) | Executes tasks sequentially in playbooks |
| State Management | Maintains a state file to track resources | Stateless – does not track past states |
| Best For | Provisioning cloud infrastructure (AWS, Azure, GCP, VMware, etc.) | Configuring software, OS, and apps on provisioned servers |
| Idempotency | Ensures resources always match the desired code (no duplicates) | Can be idempotent, but depends on playbook design |

* **Now install Terraform on Windows OS:-**

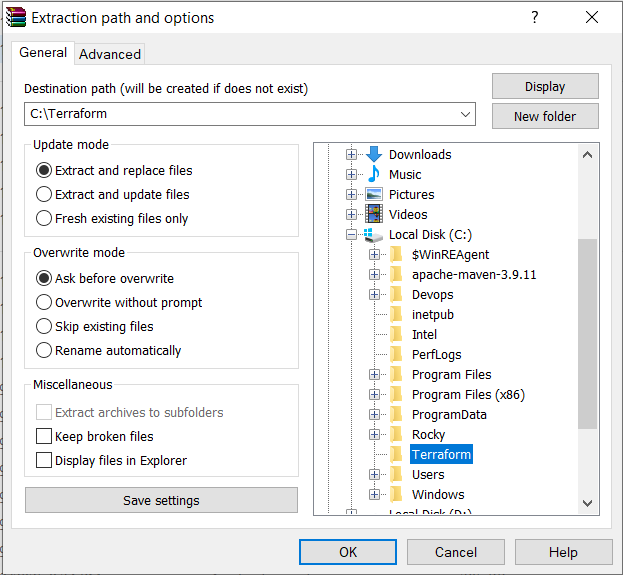
Go to any browser type terraform download and choose first site and select windows 386



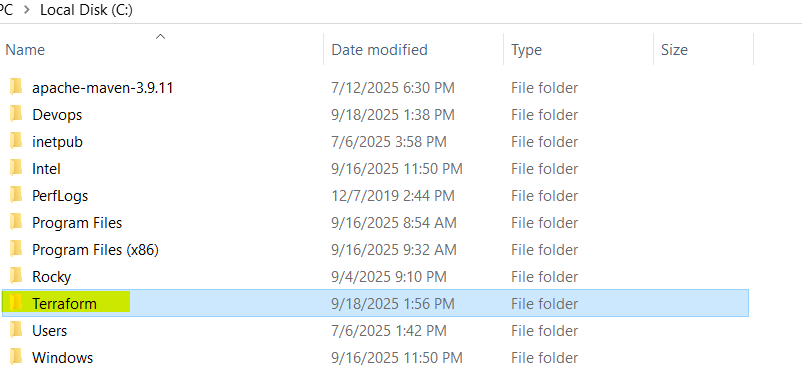


Then download and zip file is downloaded

Then extract the zip file in windows in C drive and then create a folder with name is terraform

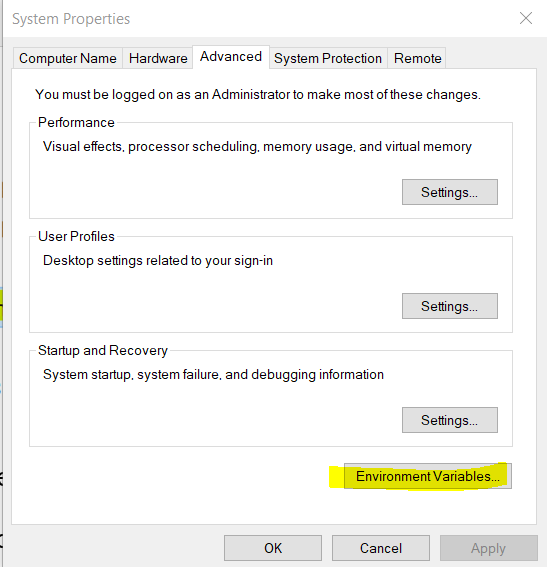


Then click ok and go to C drive inside terraform folder



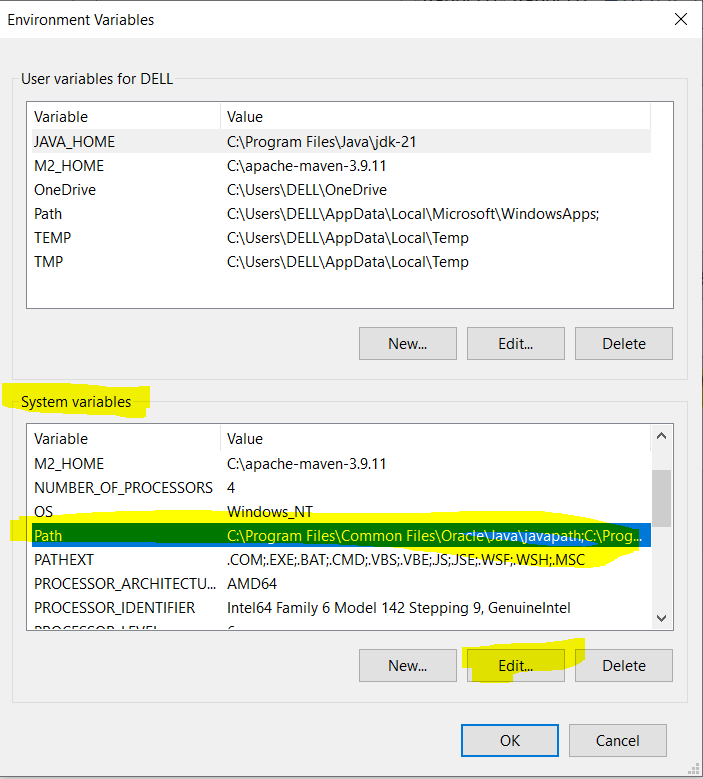
Go to inside terraform folder then copy the path

Once the copy the path go windows search bar type environment variable  
then click on environment variable

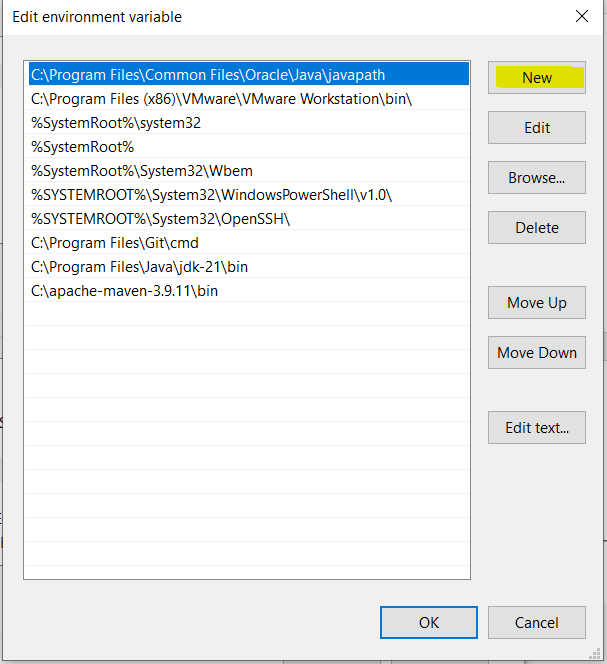


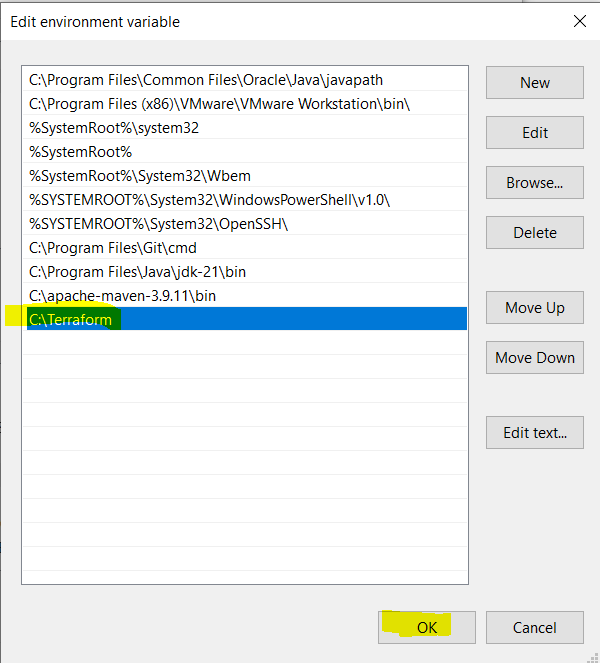
Then new windows is open

Select system veriable and click on path option and make it edit



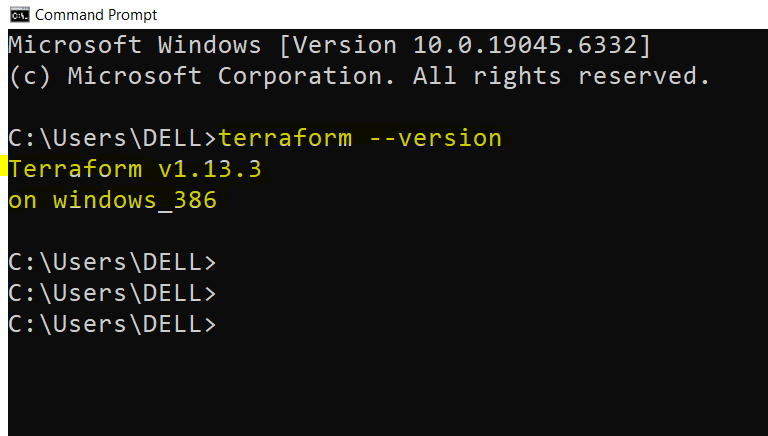
Then click on new & then the terraform location path and then ok





Now bo to windows serach bar type cmd  
then command prompt is open

Run the command  
terraform --version



If this output is not visible then restart the windows

Now Successfully installed Terraform on windows OS

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* Now Install Terraform on Redhat Linux  
  Login to AWS Console then Go to EC2 Service  
  Launch 1 EC2 instance

Name:- Terraform\_Server

AMI:- Redhat-Linux

Instance Type:- t3.micro

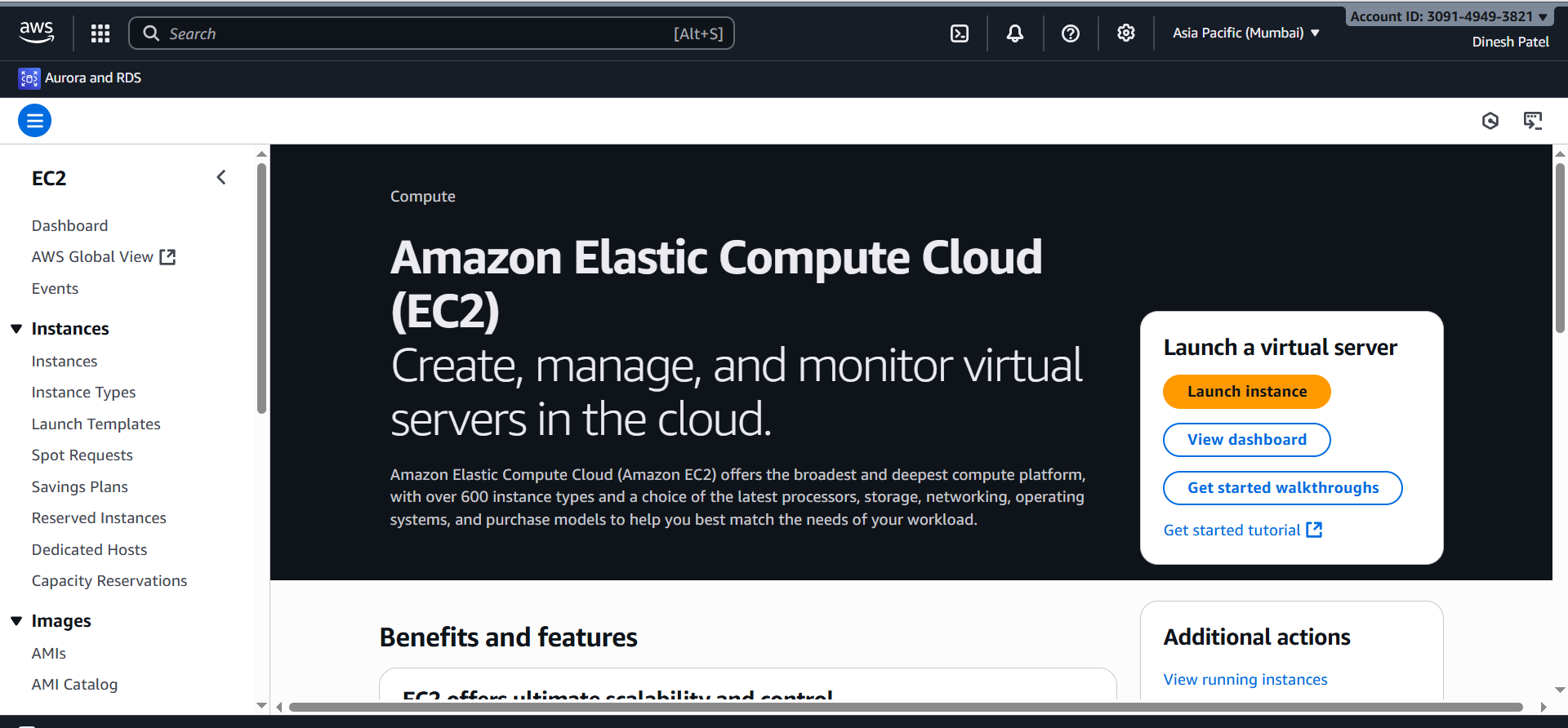
Key:- Terraform\_Key

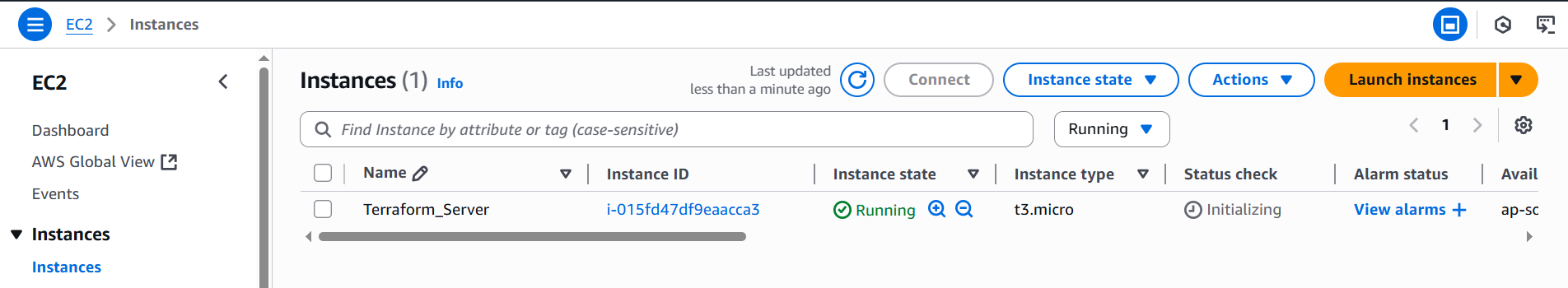
VPC:- Default

AZ:- 1a  
SG:- Terraform\_SG

Open port  
SSH and HTTP

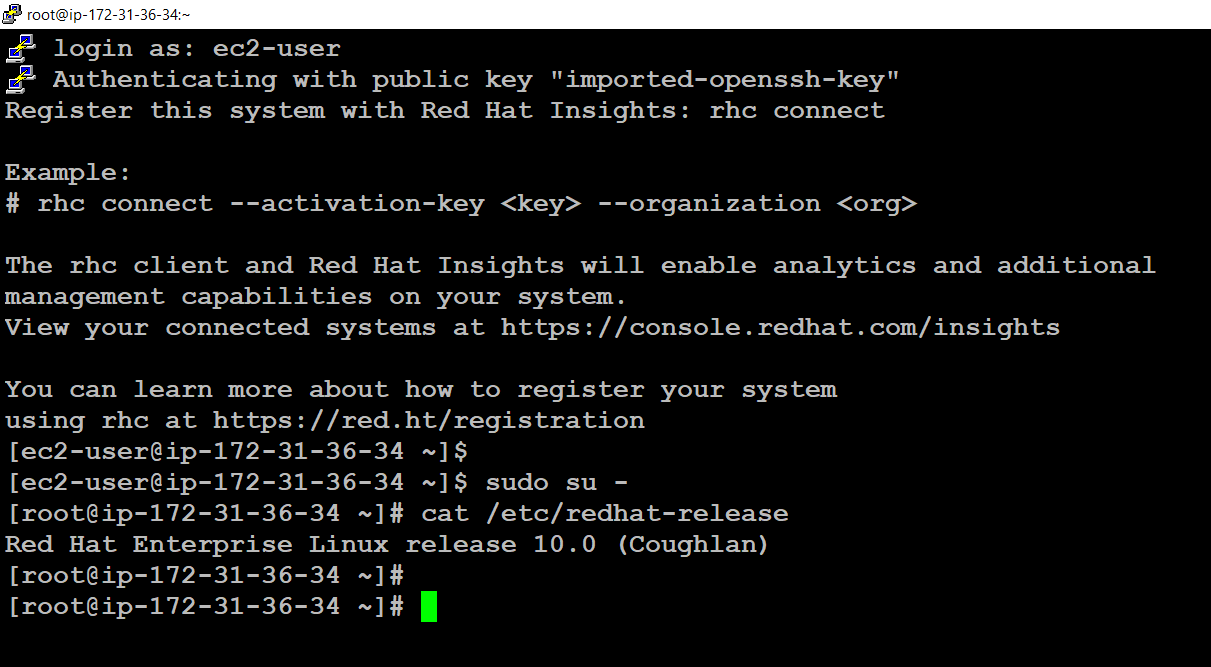
Storage:- 10GB



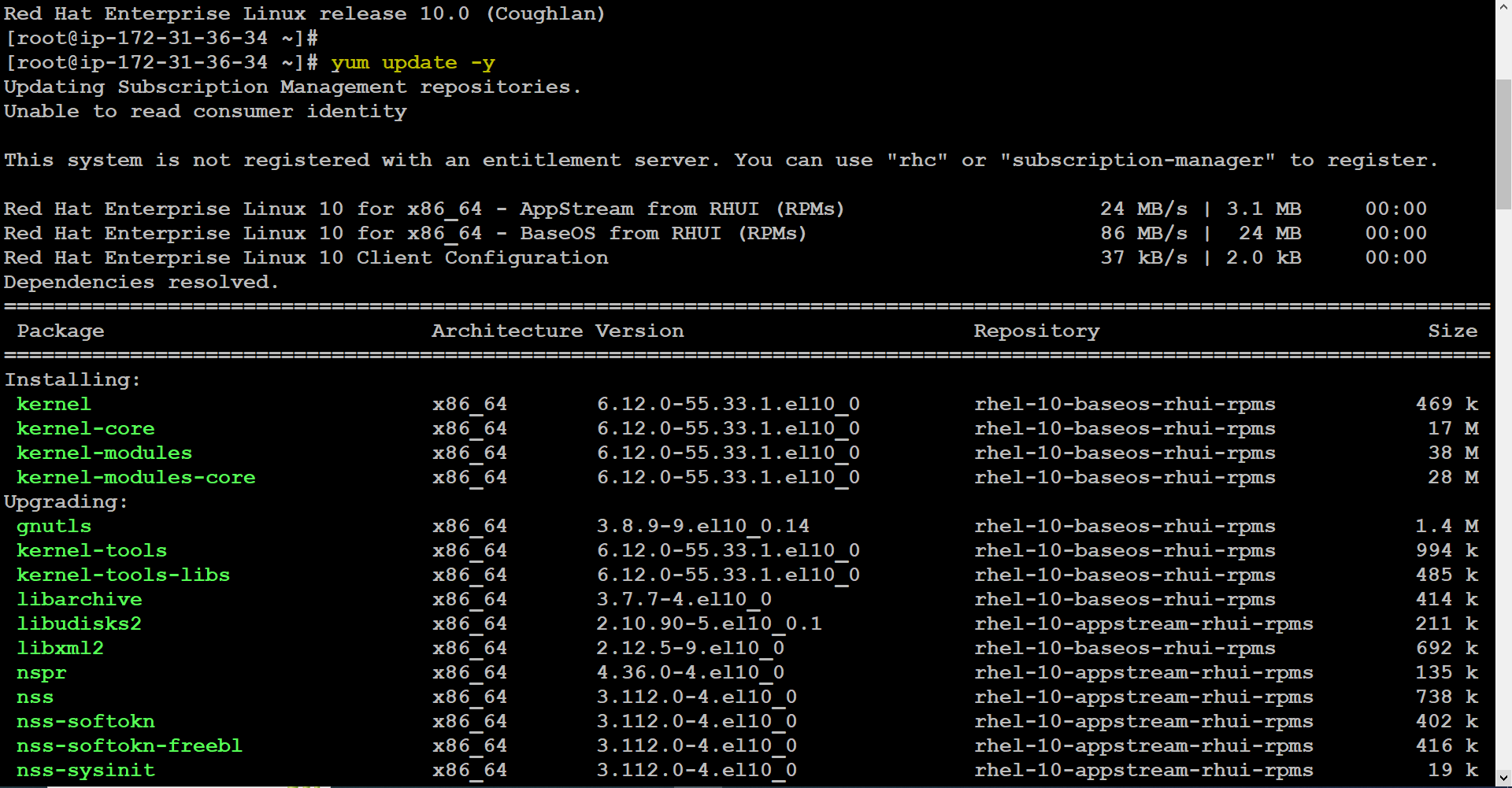


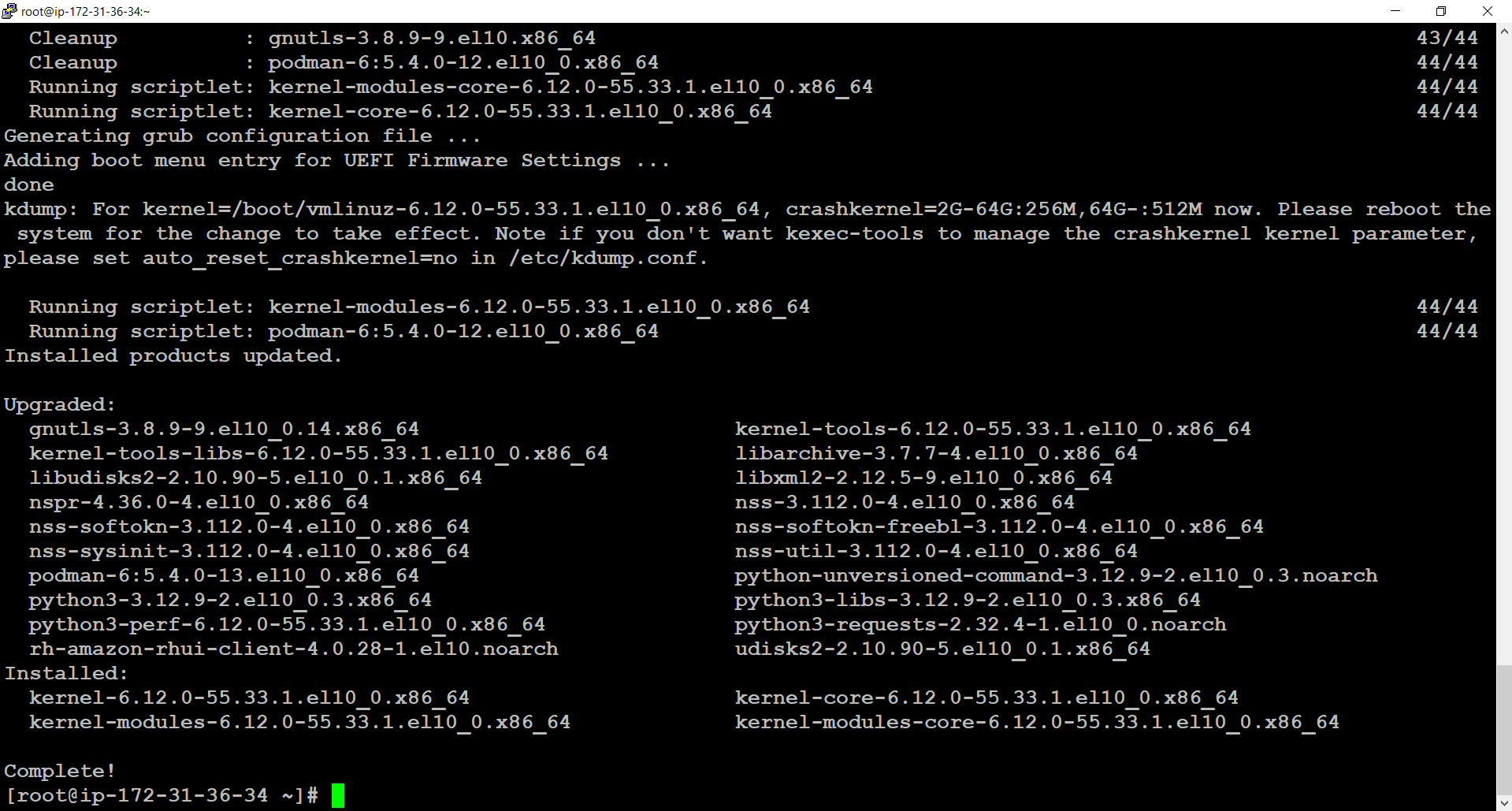
Now to instance via putty

Login User:- ec2-user



#yum update -y

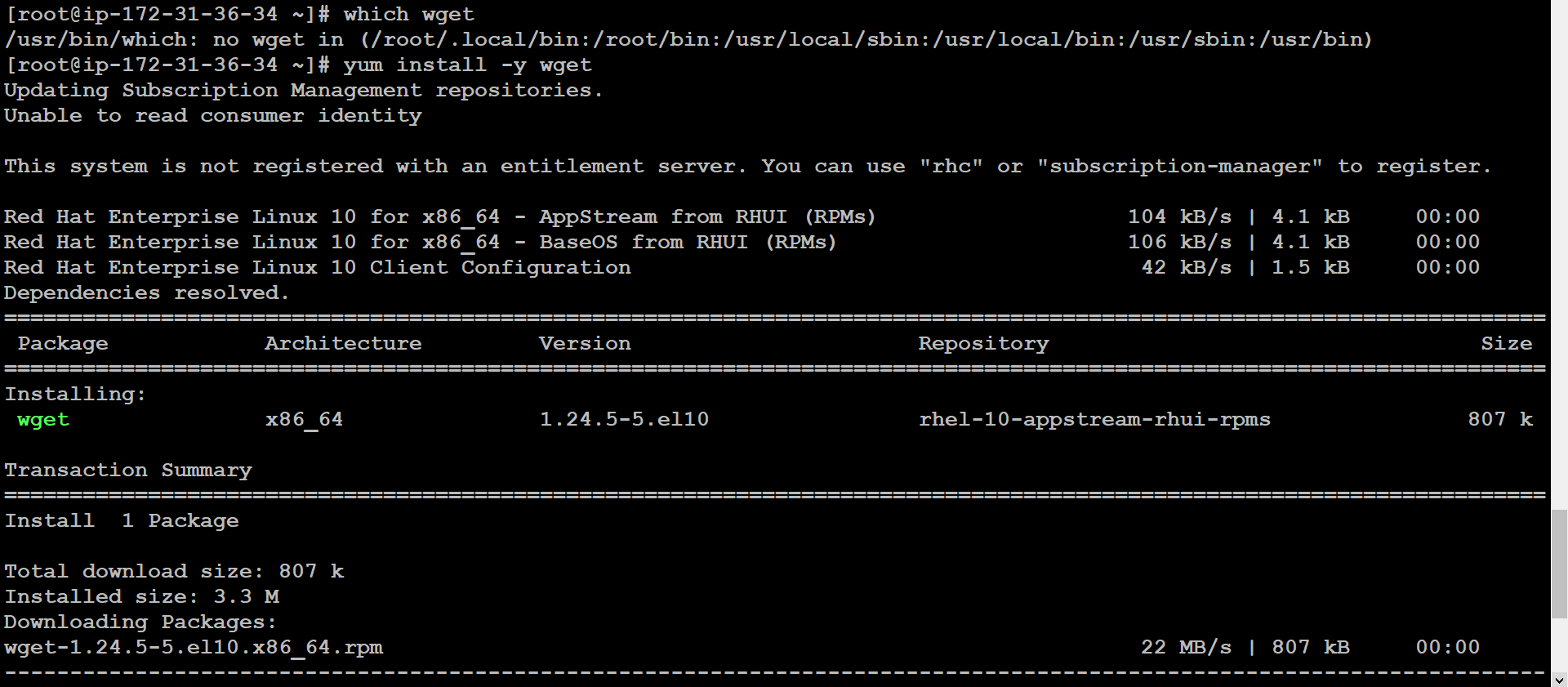


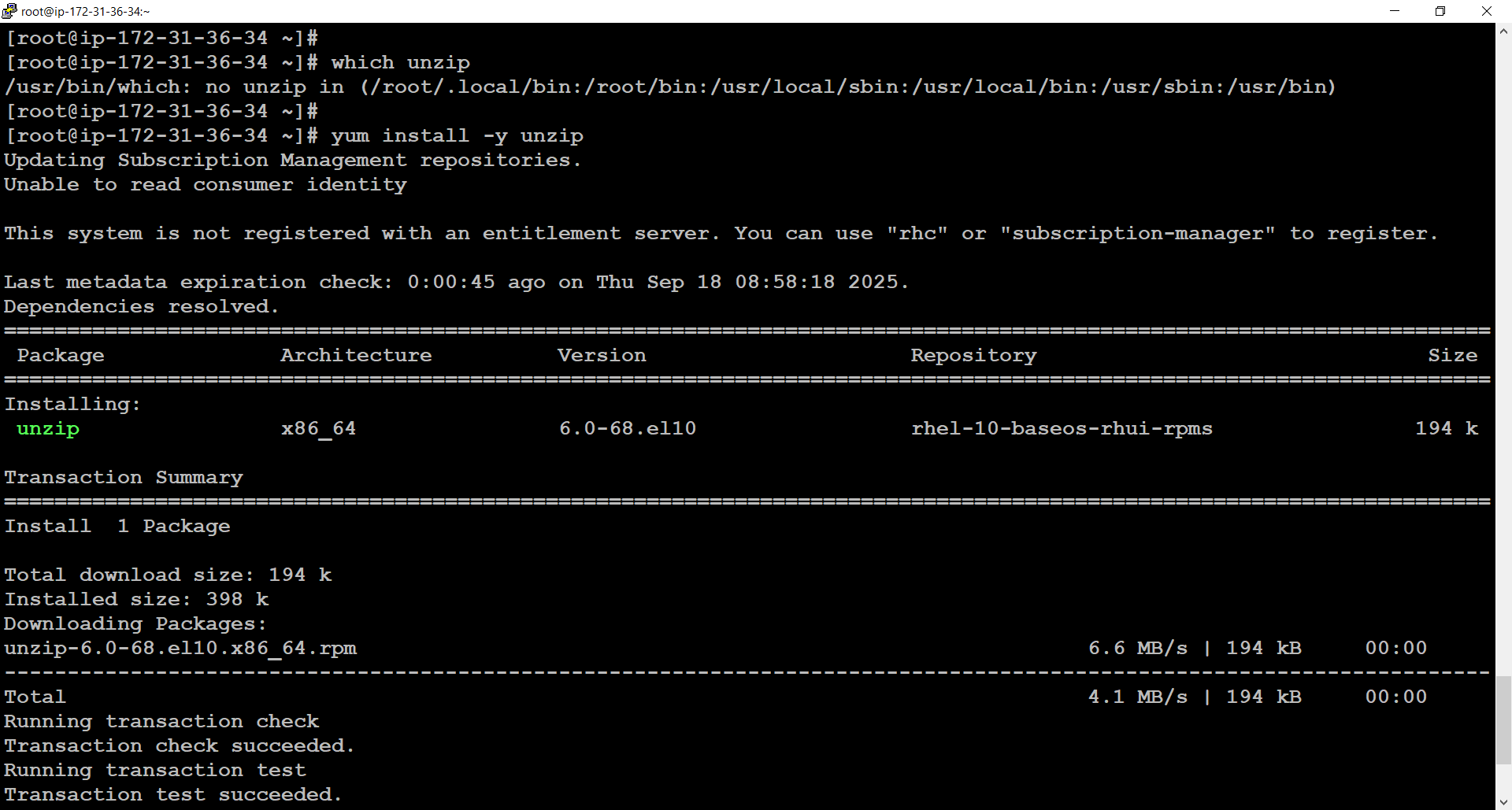


Now install wget and unzip command

#yum install -y wget

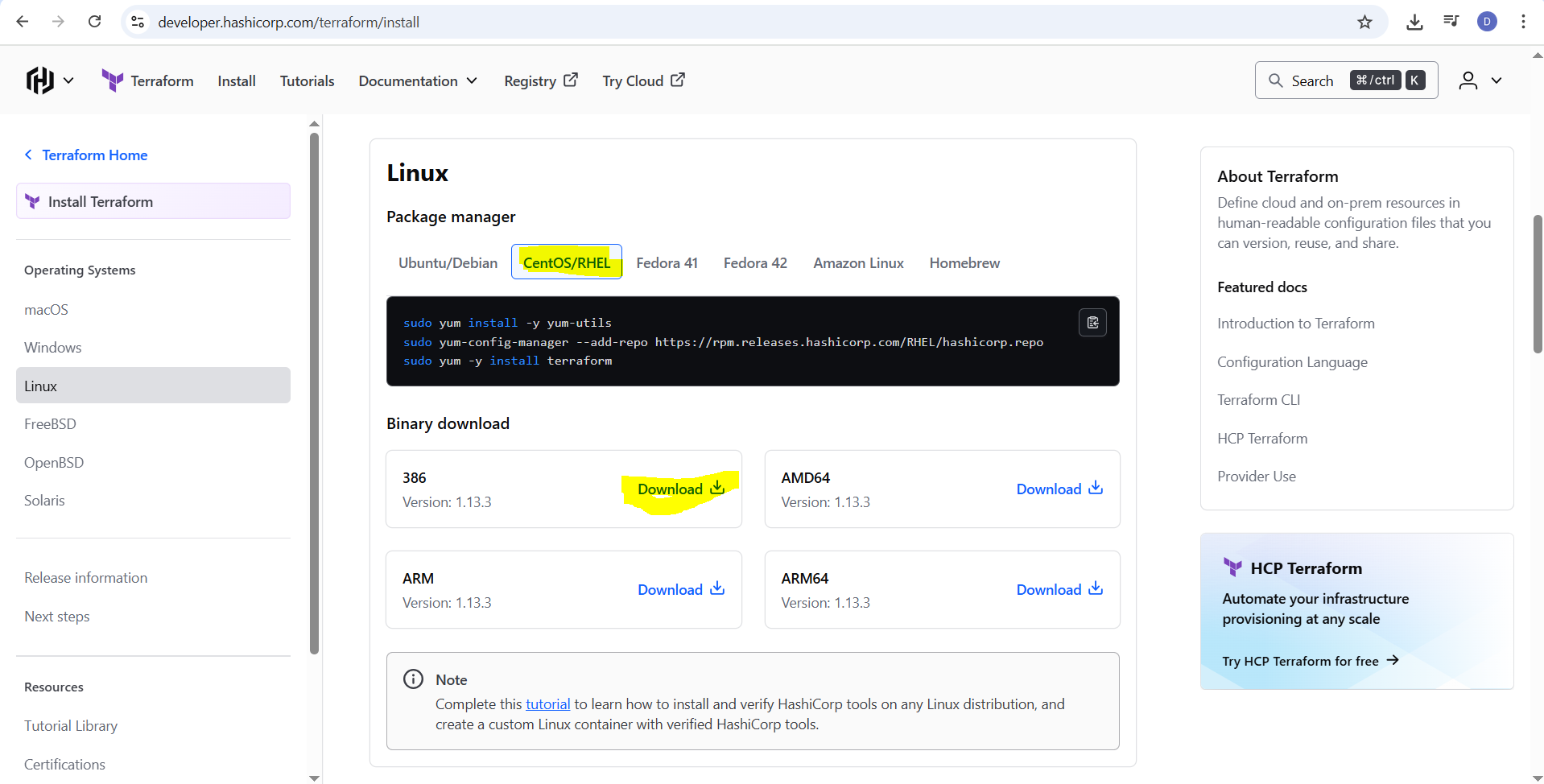
#yum install -y unzip





Now install Terraform on Redhat-Linux

First download the in browser

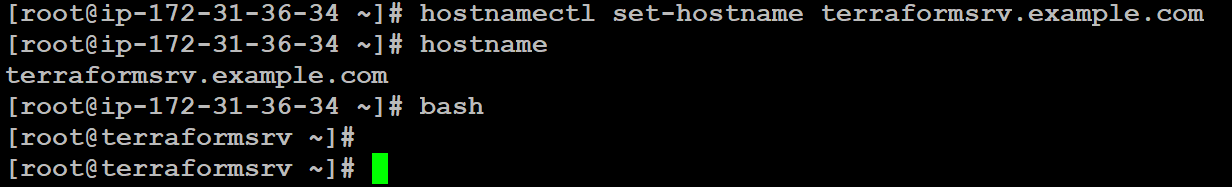


Now set the hostname

#hostnamectl set-hostname terraformsrv.example.com

#hostname

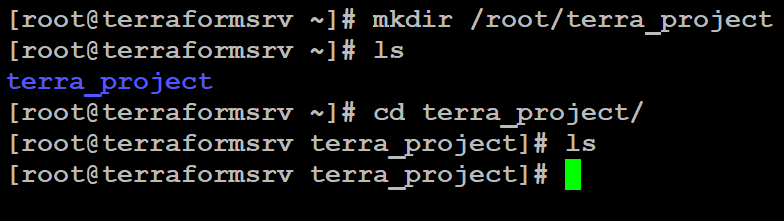
#bash



Now crate one directory inside /root

# mkdir /root/terra\_project

#ls  
# cd terra\_project



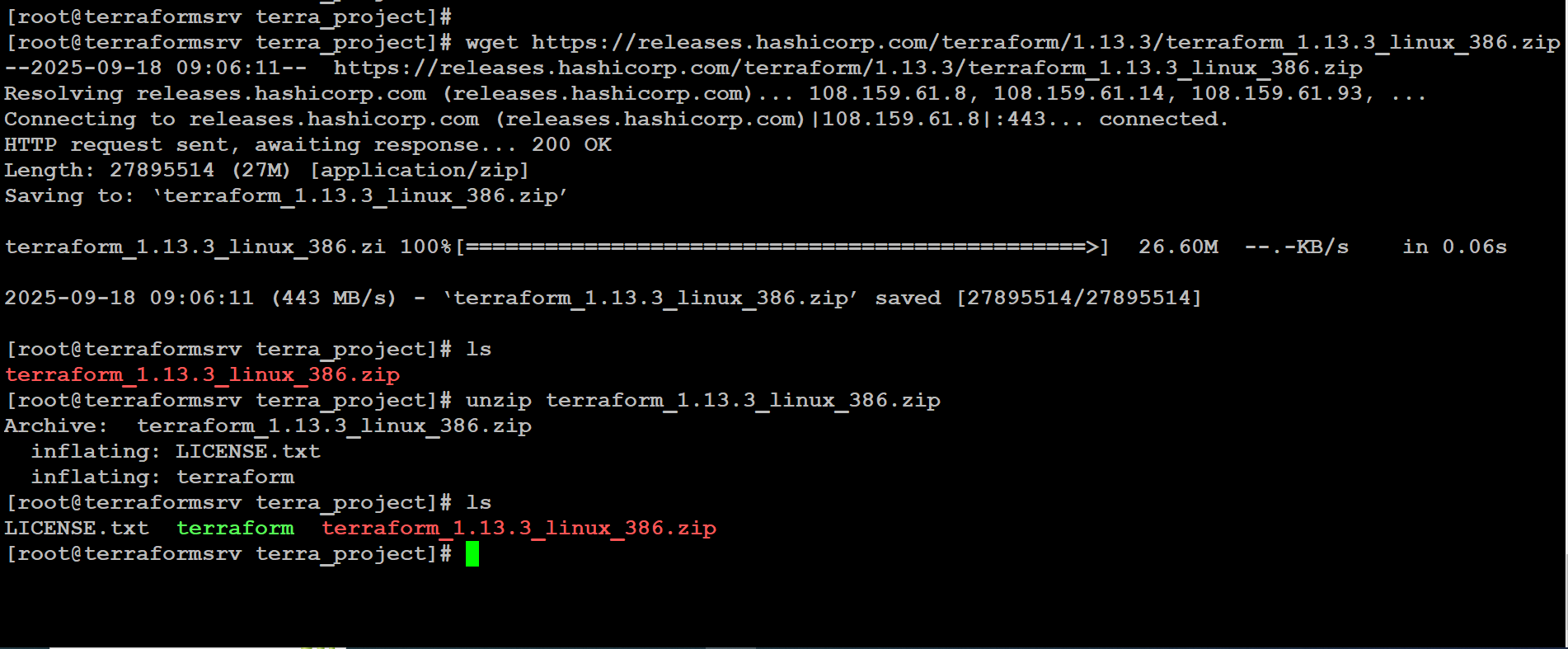
# wget <https://releases.hashicorp.com/terraform/1.13.3/terraform_1.13.3_linux_386.zip>

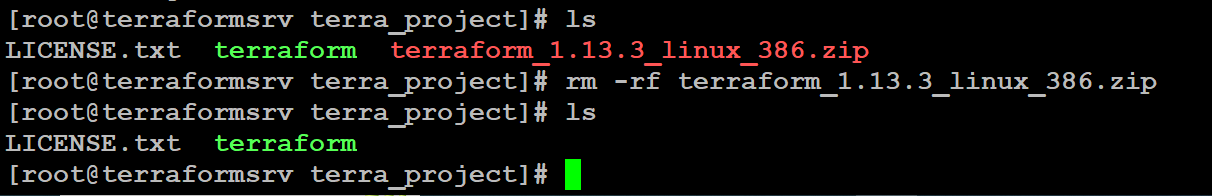
#ls

# unzip terraform\_1.13.3\_linux\_386.zip

#ls

#rm -rf terraform\_1.13.3\_linux\_386.zip

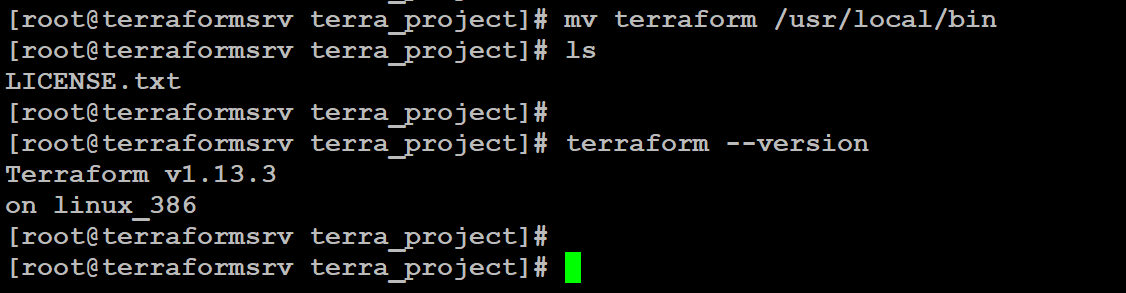




Now move the terraform directory to   
# mv terraform /usr/local/bin

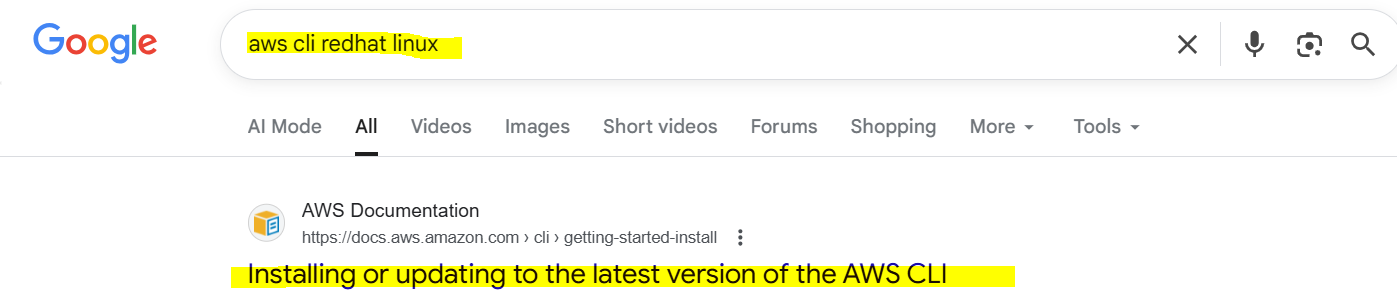
#ls

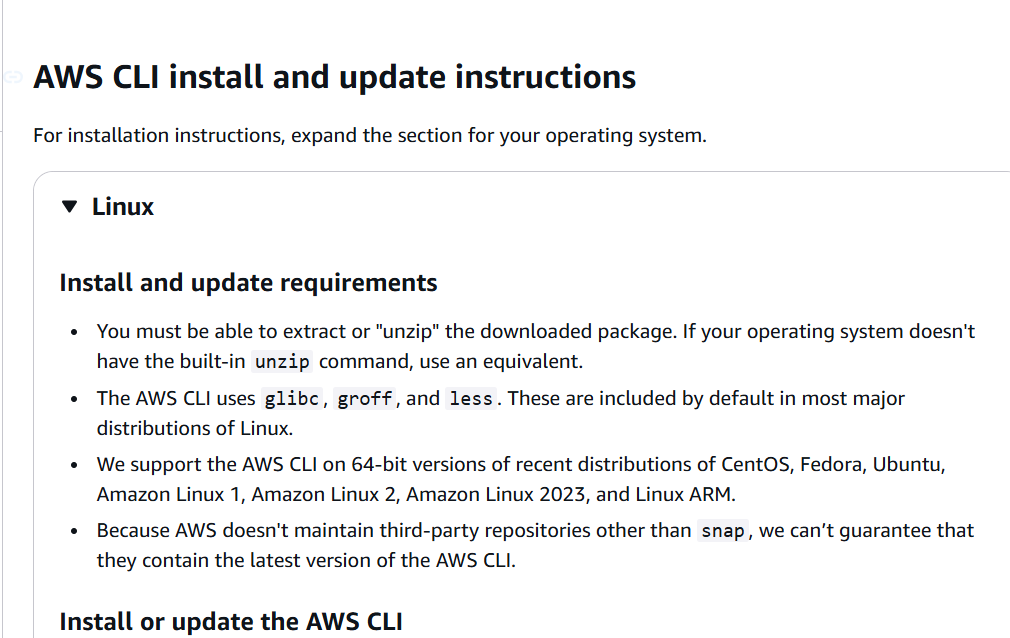
# terraform –version

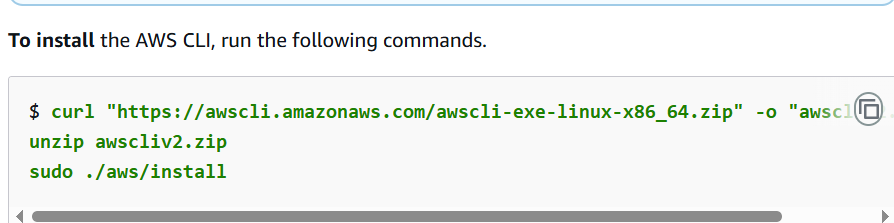


Now install AWS CLI on terraform server

Go to any browser type aws cli installed on redhat-linux

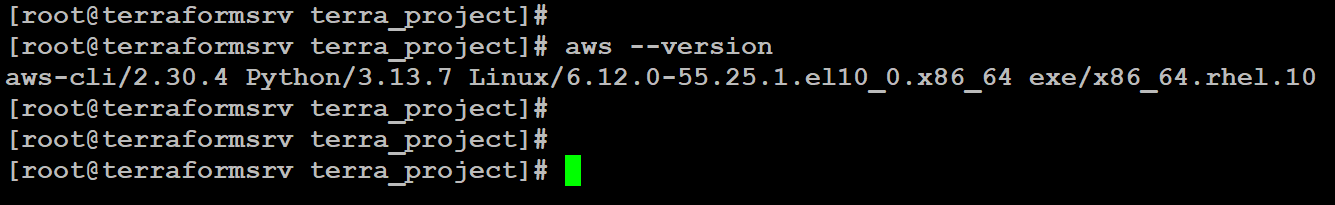






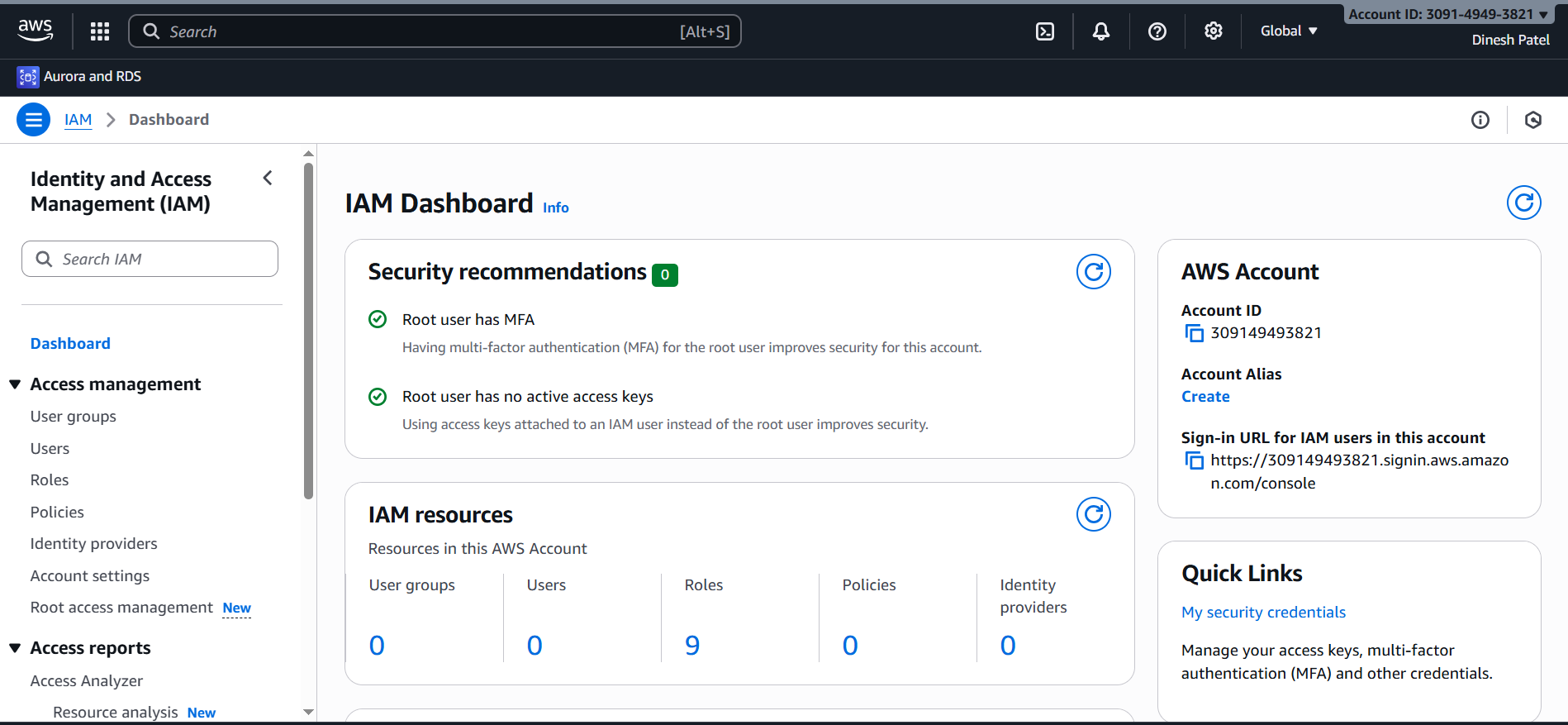
Now to check aws cli version  
run the below command

#aws --version

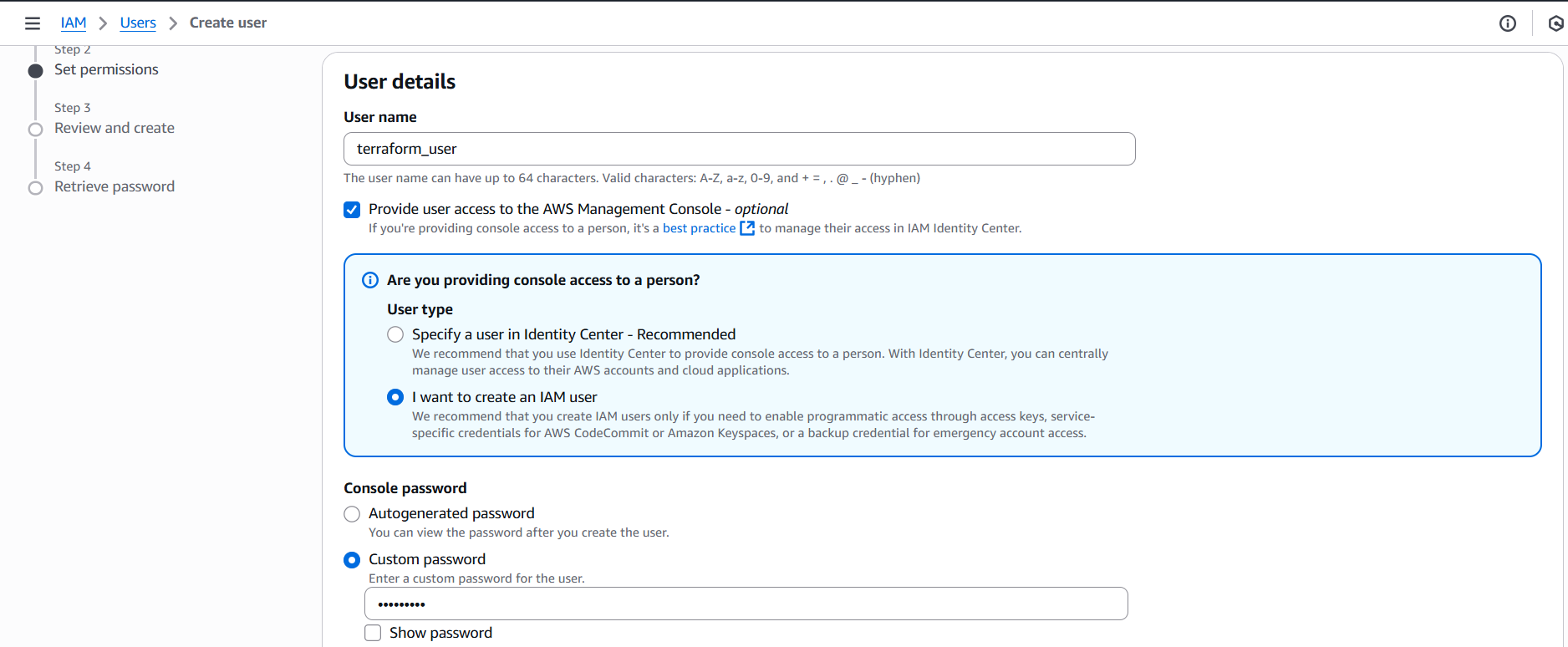


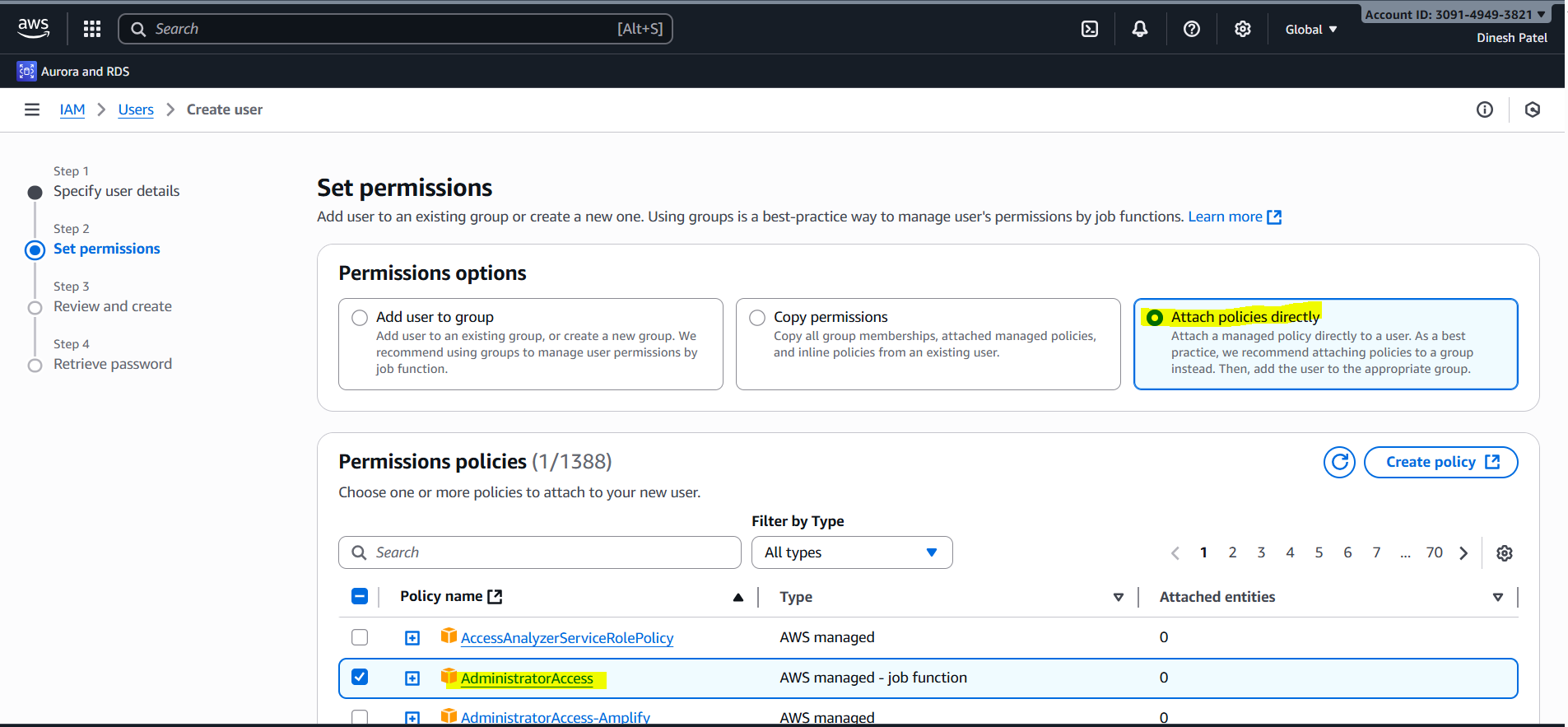
Now to run terraform script to automate to aws infra we need one IAM user and roles

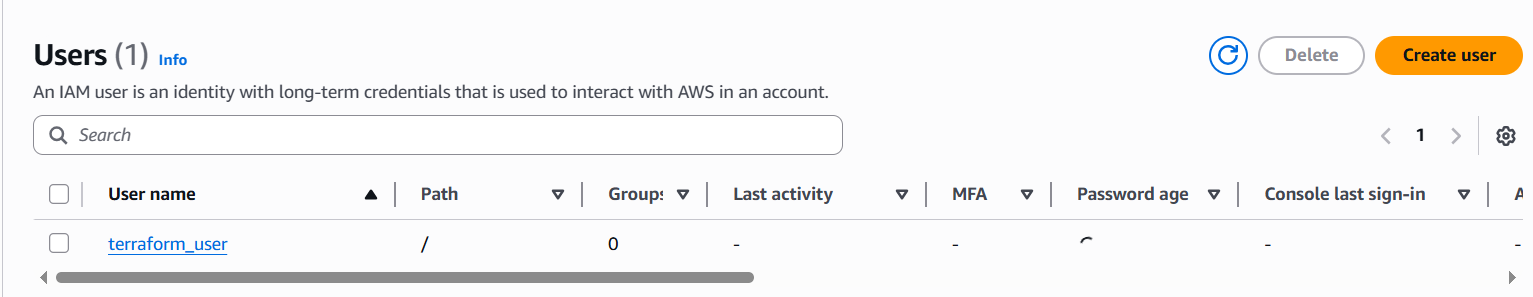
Go to aws console search IAM Service create one IAM user



User Name:- terraform\_user







Create an IAM user in AWS Console (e.g., terraform\_user) with programmatic access (Access Key and Secret Key).

Now go to terraform\_server

And run one command

# aws configure --profile terraform\_user

Terraform setup is now complete. You can use it to manage infrastructure as code (IaC) and automate cloud resource provisioning efficiently.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_