**Capstone Project – 1**

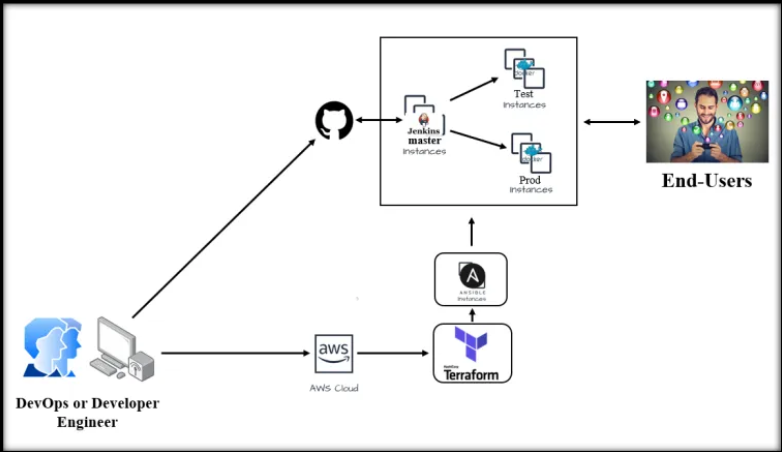
Task to be Performed:

1. No of instances:

* Terraform master
* Ansible master
* Jenkins master
* Jenkins test server
* Jenkins prod server

1. Operating-System: Ubuntu 20.04

**SOLUTION:**



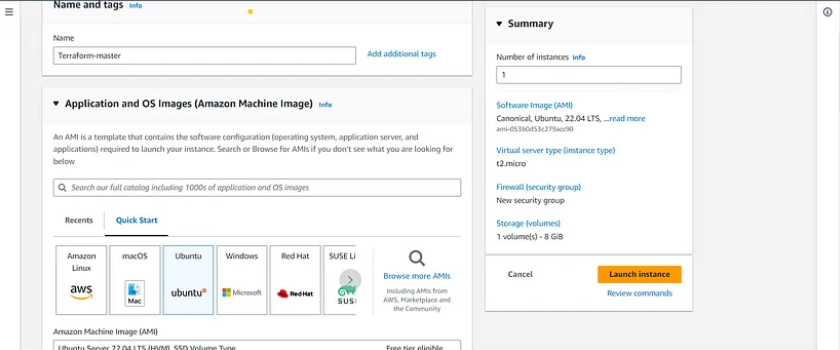
**Pre-requisites:**

AWS account

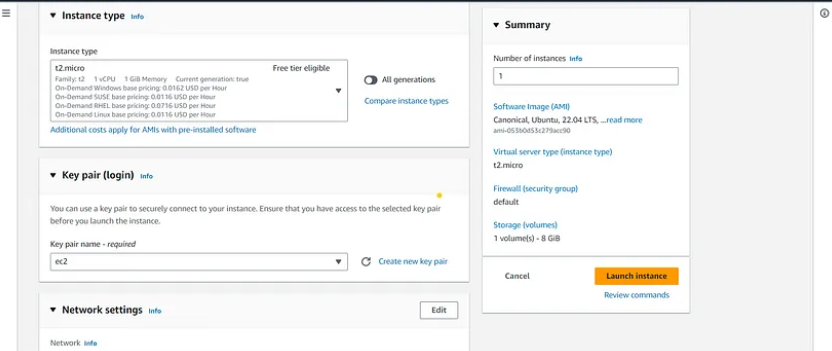
Knowledge about jenkins(Master-slave), terraform ,ansible

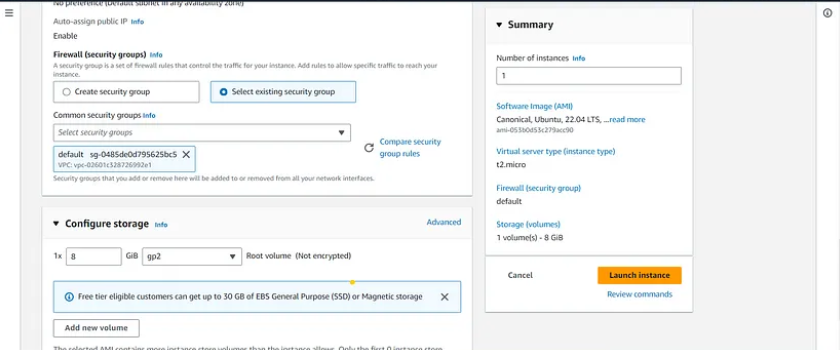
**Step-1: Creating a terraform master instance:**

Login into AWS management console and going EC2 management console for launching terraform master instance.



Here the instance specifications according to our spec needs. I am selecting os=Ubuntu 22.04 LTS, instance type= t2.micro.





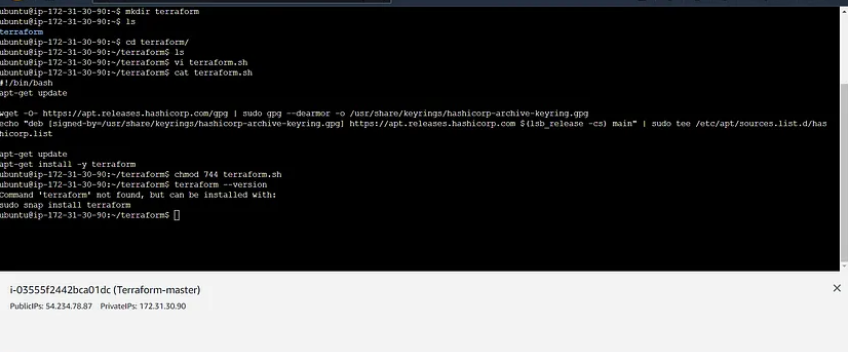
The instance has been created, now we need to connect the instance and download terraform package on it in order terraform to provision our resources.

For installing terraform I am creating a script file, in that script file that will contain the script to download and install the terraform on our instance and changing the file permission of the file for execution purpose.

Before the execution of the file making the directory as terraform for our working purpose.

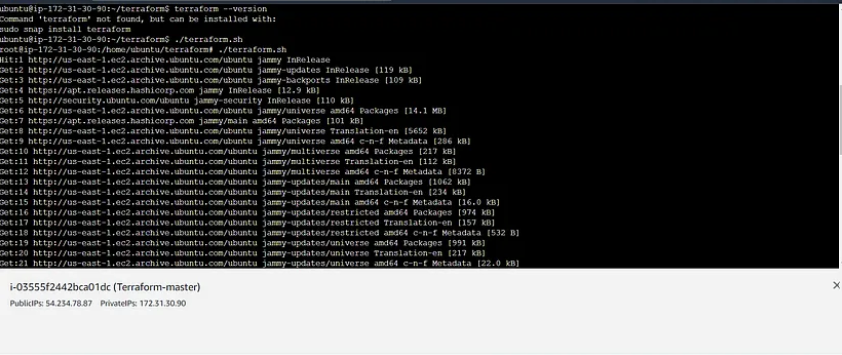
**vi terraform.sh**

#!/bin/bash  
 apt-get update  
  
wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg  
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(lsb\_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list  
  
apt-get update   
apt-get install -y terraform

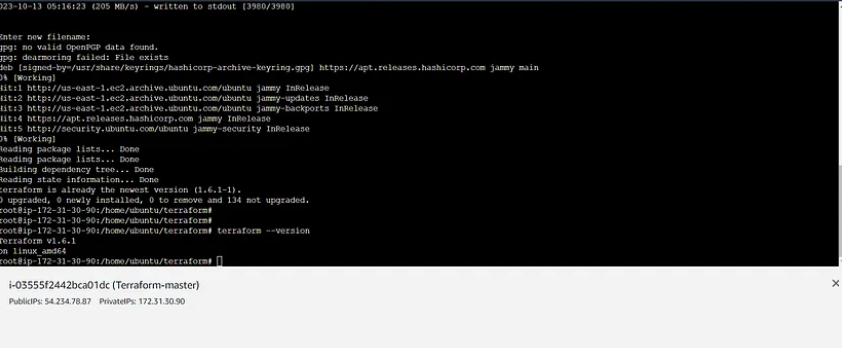


To run the vi terraform.sh file using the command

./terraform.sh

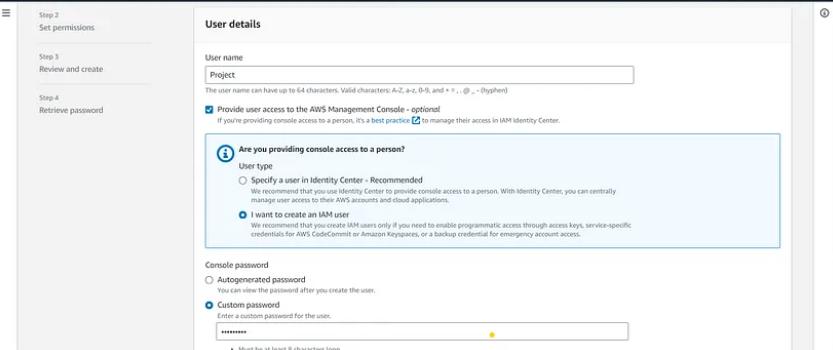


After executing the file. Checking the terraform version.

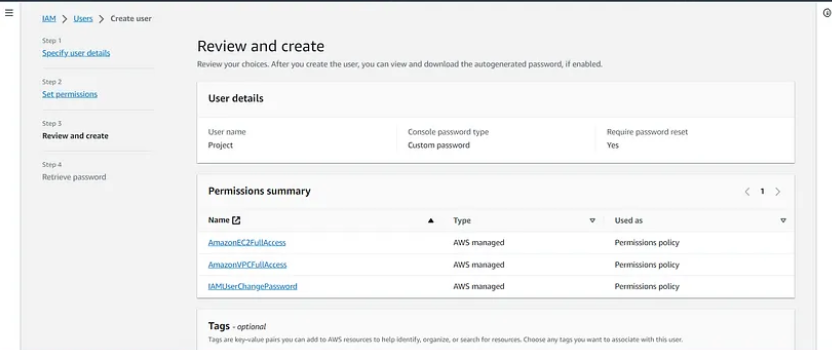


Creating a IAM user for terraform to access our cloud. Here I am giving 2 permissions — EC2 full access and VPC full access.

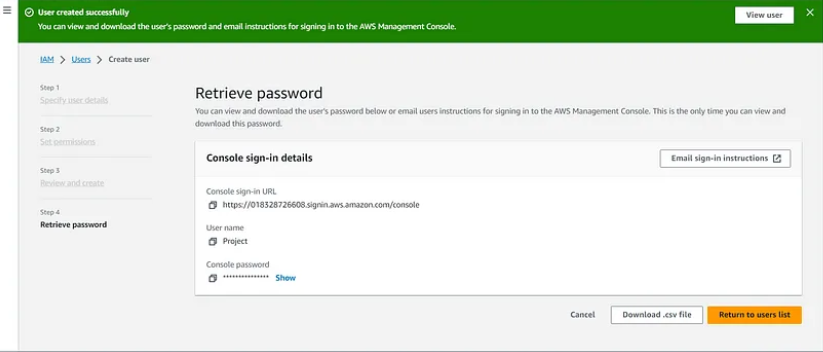
Click->Iam user->create user ->user details



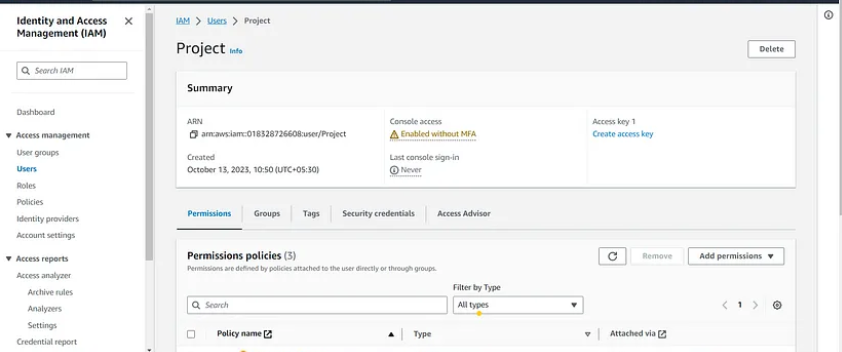
Attach policies

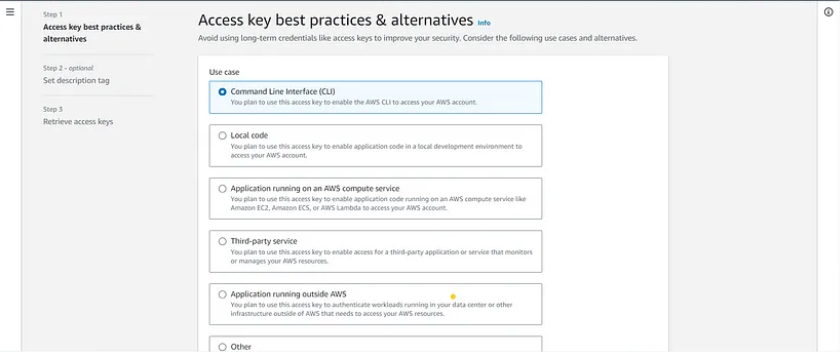


Note the username & password

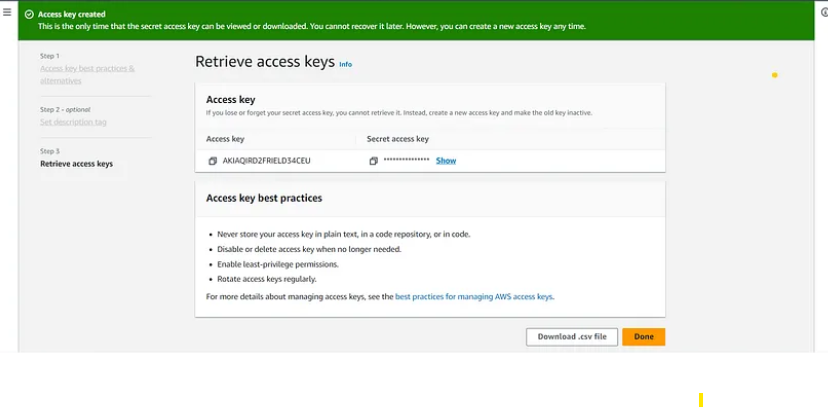


Create access key





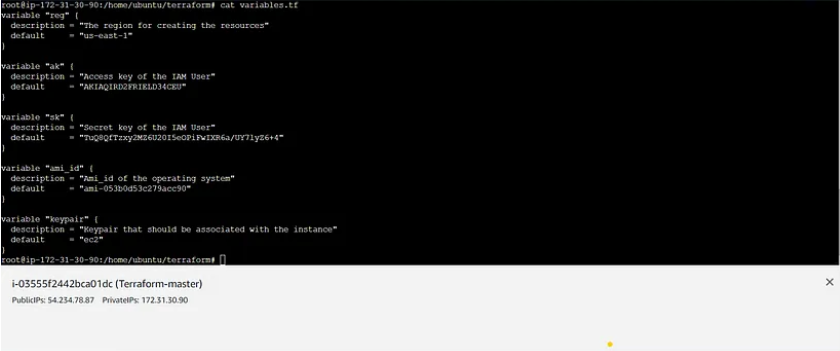
Note the access key and secret keys



**Creating the variables.tf file. Which contains all the variables details**

**variables.tf**

variable "reg" {  
 description = "The region for creating the resources"  
 default = "ap-south-1"  
}  
variable "ak" {  
 description = "Access key of the IAM User"  
 default = "AKIAZANDQIRABD4RV2WB"  
}  
  
variable "sk" {  
 description = "Secret key of the IAM User"  
 default = "n5Kn6O9CV3YtaG68CiUaEZk+WmMVAKJQH2SBYAYm"  
}  
  
variable "ami\_id" {  
 description = "Ami\_id of the operating system"  
 default = "ami-08e5424edfe926b43"  
}  
  
variable "keypair" {  
 description = "Keypair that should be associated with the instance"  
 default = "console\_server"  
}



Now creating the main terraform file. Which contains all the details for provisioning the infrastructure.

Which will create the VPC and required components for vpc to assist and 4 instances will be created.

1for ansible master and another one Jenkins master and another two instances for test and prod.

**vi main.tf**

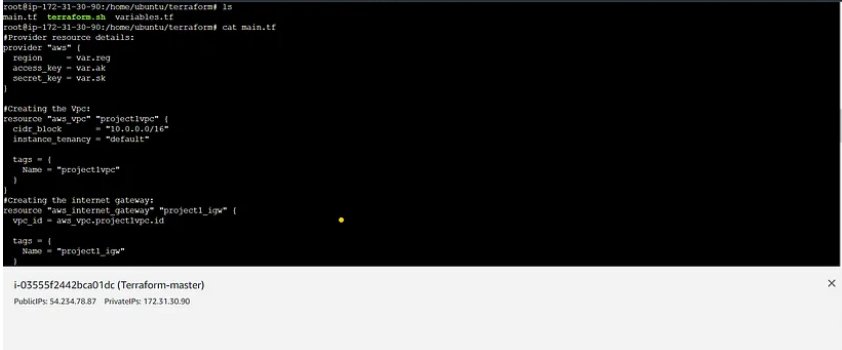
#Provider resource details:  
provider "aws" {  
 region = var.reg  
 access\_key = var.ak  
 secret\_key = var.sk  
}  
  
#Creating the Vpc:  
resource "aws\_vpc" "project1vpc" {  
 cidr\_block = "10.0.0.0/16"  
 instance\_tenancy = "default"  
  
 tags = {  
 Name = "project1vpc"  
 }  
}  
#Creating the internet gateway:  
resource "aws\_internet\_gateway" "project1\_igw" {  
 vpc\_id = aws\_vpc.project1vpc.id  
 tags = {  
 Name = "project1\_igw"  
 }  
}

#creating the subnet1:  
resource "aws\_subnet" "project1\_subnet" {  
 vpc\_id = aws\_vpc.project1vpc.id  
 cidr\_block = "10.0.1.0/24"  
  
 tags = {  
 Name = "project1\_subnet"  
 }  
}  
#creating the route table:  
resource "aws\_route\_table" "project1\_route\_table" {  
 vpc\_id = aws\_vpc.project1vpc.id  
 tags = {  
 Name = "project1\_route\_table"  
 }  
}

#associating route internet gateway in route table:  
resource "aws\_route" "project1\_routing" {  
 route\_table\_id = aws\_route\_table.project1\_route\_table.id  
 destination\_cidr\_block = "0.0.0.0/0" # Route all traffic to the Internet Gateway  
 gateway\_id = aws\_internet\_gateway.project1\_igw.id  
}  
#creating the subnet association with the route table:  
resource "aws\_route\_table\_association" "subnet\_association" {  
 subnet\_id = aws\_subnet.project1\_subnet.id  
 route\_table\_id = aws\_route\_table.project1\_route\_table.id  
}  
#creating the security group:  
resource "aws\_security\_group" "project1\_sc" {  
 name = "project1\_sc"  
 description = "security group for AWS EC2 instances"  
 vpc\_id = aws\_vpc.project1vpc.id  
 # Ingress rules (inbound traffic)  
 # Allow SSH (port 22) from anywhere  
 ingress {  
 from\_port = 22  
 to\_port = 22  
 protocol = "tcp"  
 cidr\_blocks = ["0.0.0.0/0"]  
 }

# Allow HTTP (port 80) from anywhere  
 ingress {  
 from\_port = 80  
 to\_port = 80  
 protocol = "tcp"  
 cidr\_blocks = ["0.0.0.0/0"]  
 }  
 # Allow HTTPS (port 443) from anywhere  
 ingress {  
 from\_port = 443  
 to\_port = 443  
 protocol = "tcp"  
 cidr\_blocks = ["0.0.0.0/0"]  
 }  
 # Allow CUSTOM TCP (port 8080) from anywhere  
 ingress {  
 from\_port = 8080  
 to\_port = 8080  
 protocol = "tcp"  
 cidr\_blocks = ["0.0.0.0/0"]  
 }  
 # Egress rules (outbound traffic)  
 egress {  
 from\_port = 0  
 to\_port = 0  
 protocol = "-1"  
 cidr\_blocks = ["0.0.0.0/0"]  
 }

tags = {  
 Name = "project1\_sc"  
 }  
}  
#creating the instance:  
resource "aws\_instance" "server" {  
 count = "4"  
 ami = var.ami\_id  
 instance\_type = "t2.micro"  
 subnet\_id = aws\_subnet.project1\_subnet.id  
 key\_name = var.keypair  
 vpc\_security\_group\_ids = [aws\_security\_group.project1\_sc.id]  
 associate\_public\_ip\_address = true  
 tags = {  
 Name = "Server - ${count.index}"  
 }  
}



**Now we need to perform terraform init command for terraform to download the dependencies required to support the provision for our requirement.**

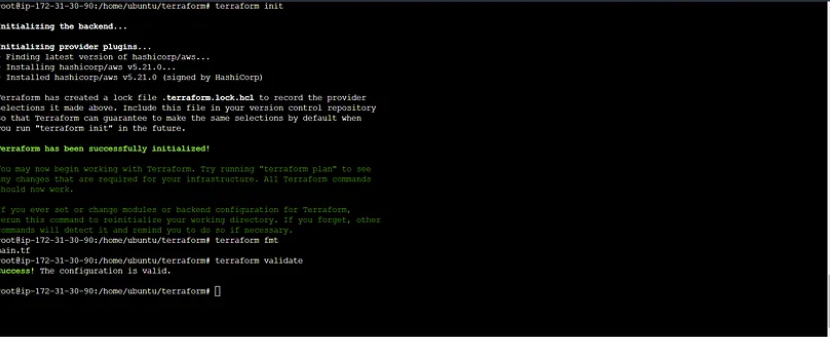
terraform init

**After that we need to format the terraform files with the help of the command**

terraform fmt

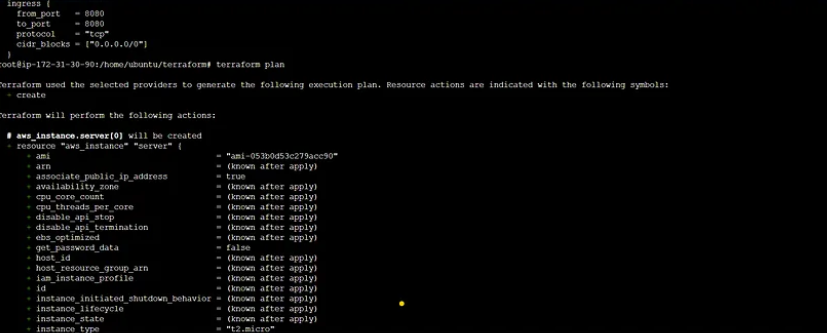
**Then we need to validate the credentials by using the command**

terraform validate

****

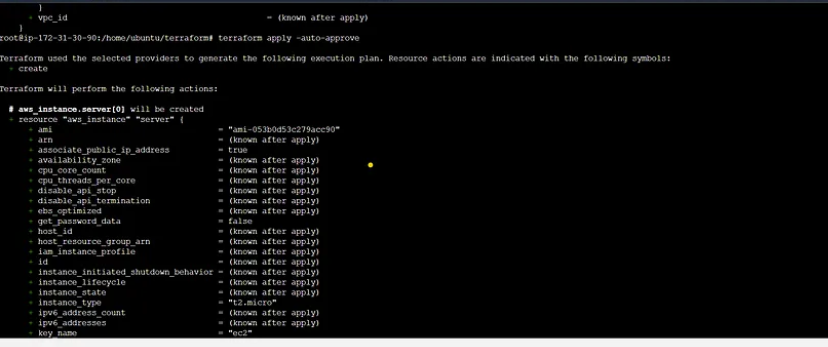
**The credentials have been valid now we need to initiate terraform to plan how to provision the infrastructure requirements.**

terraform plan

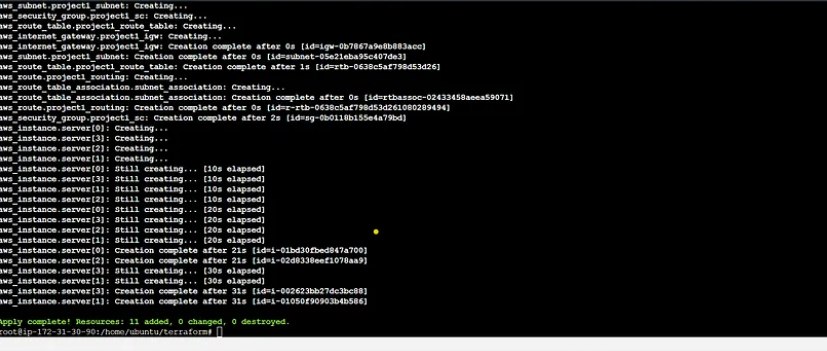


**Finally, we need to perform below command for terraform to provision the infrastructure.**

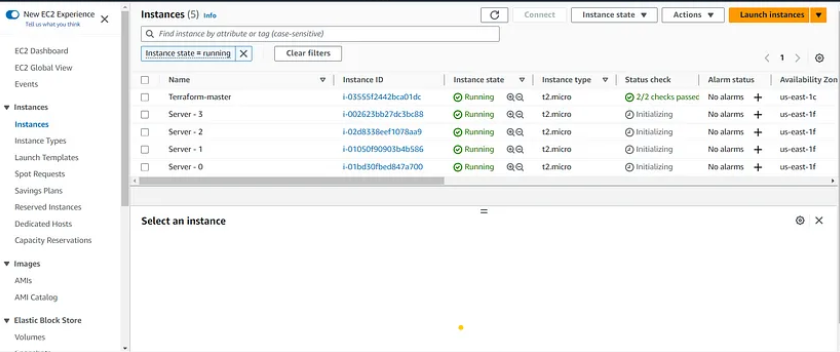
terraform apply -auto-approve



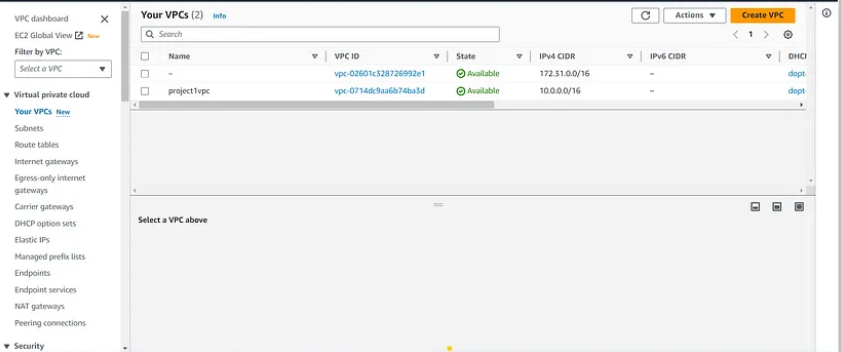
created infrastructure.



Resources had been provisioned by terraform now we need to check it on the console.

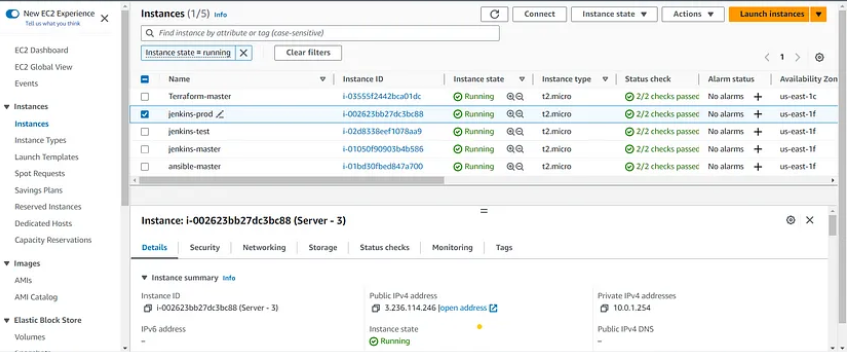


VPC also created



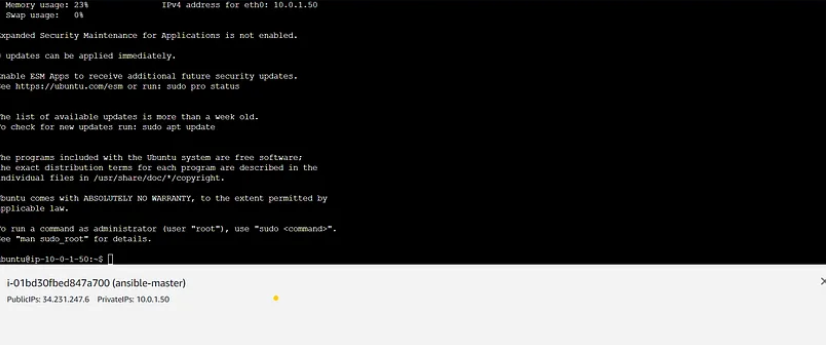
Resources has been provisioned successfully

Now we need to rename the resources the instances according to our specifications like ansible master, Jenkins master, Jenkins test, Jenkins prod.



* Connect the all instances by EC2 instance connect for ansible cluster formation to install the necessary software on each instances.
* Other than ansible master nodes we need to make a small change on other servers for ssh connection in order ansible to install the packages.
* For that we need to set up the password for the root user, modify the slight changes on vi /etc/ssh/sshd\_config
* Once making the changes restart the sshd service with systemctl restart sshd.
* We need to install ansible on the ansible master instance,For now we connect the Ansible master instance using mobaxterm or Ec2 instance connect
* For that I am creating the script file with the script to install ansible.

**Using ec2 instance connect**

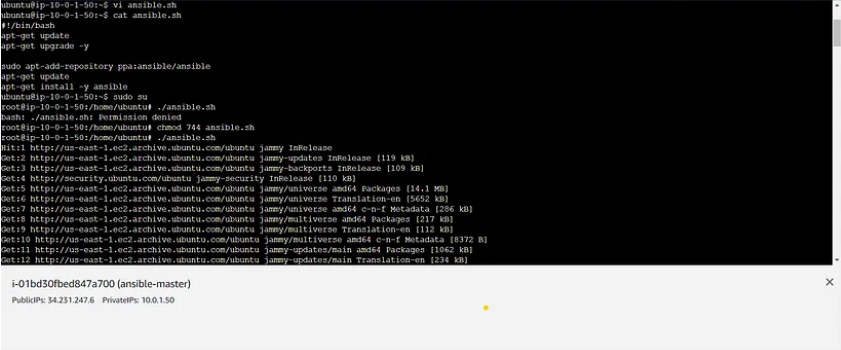
****

**vi ansible.sh**

#!/bin/bash  
apt-get update  
apt-get upgrade -y  
  
sudo apt-add-repository ppa:ansible/ansible  
apt-get update  
apt-get install -y ansible

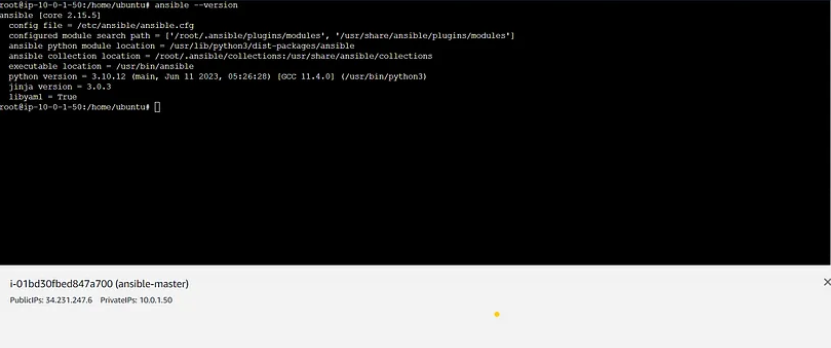
**TO give the right permission**

chmod 744 ansible.sh  
  
# Run the Script  
./ansible.sh



**Now we need to check whether ansible is installed or not by checking the version**

ansible –version



Ansible has been successfully installed. Now we need configure the host list on the ansible host file, for installing the packages on node servers like Jenkins master, Jenkins test, Jenkins prod.

