CREATING A EKS CLUSTER USING TERRAFORM

**Introduction**

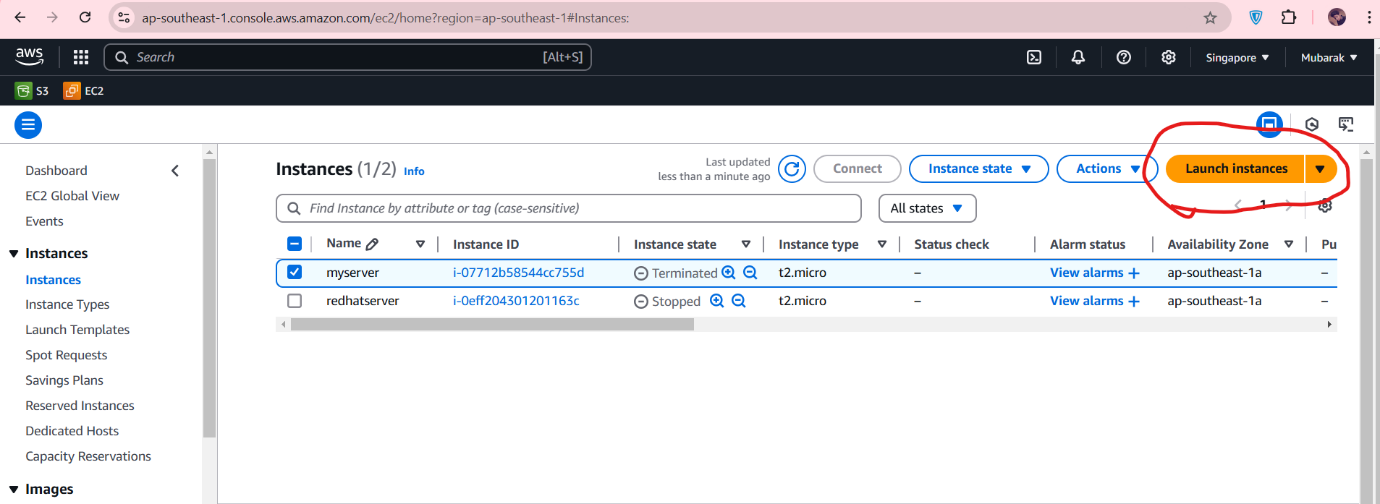
we are going to create a EKS cluster using terraform scripts. Basically Terraform

is IAAC tool where we can able to create our infrastructure using the code manner. the procedure we are following the entire EKS cluster with ACM and VPC module and EKS modules.

**Prerequisite**

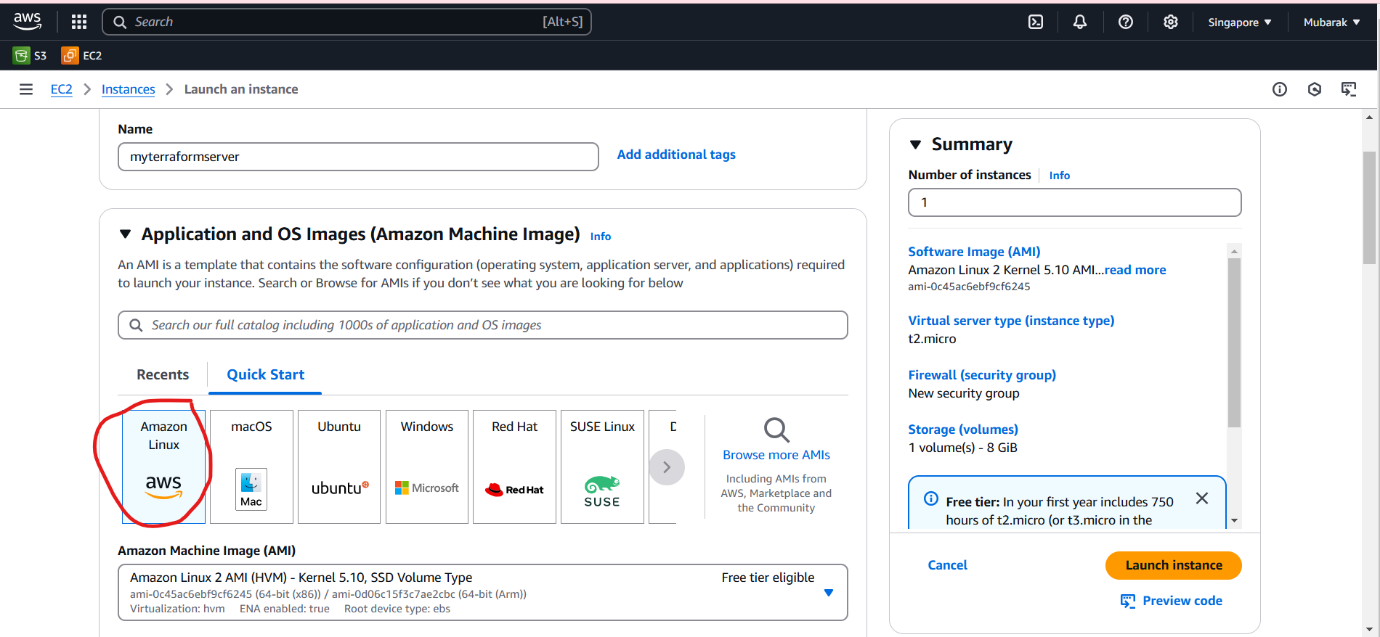
1. Need AWS Console.
2. Need to create an ec2 server for terraform to create an infra.
3. Create a IAM role of Terraform with admin Permission.
4. Install terraform in ec2 server and attach the roles as well.
5. Create a script for EKS cluster using terraform and need to run.

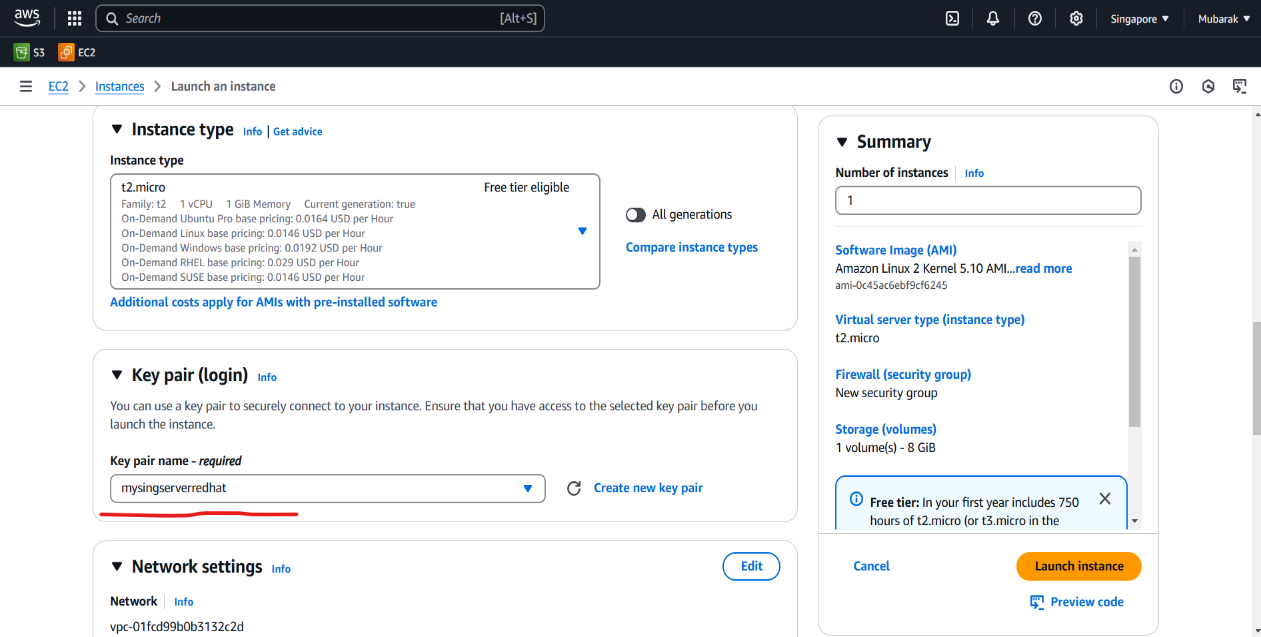
STEP1 - Log into the AWS console and create a Ec2 instance for Terraform

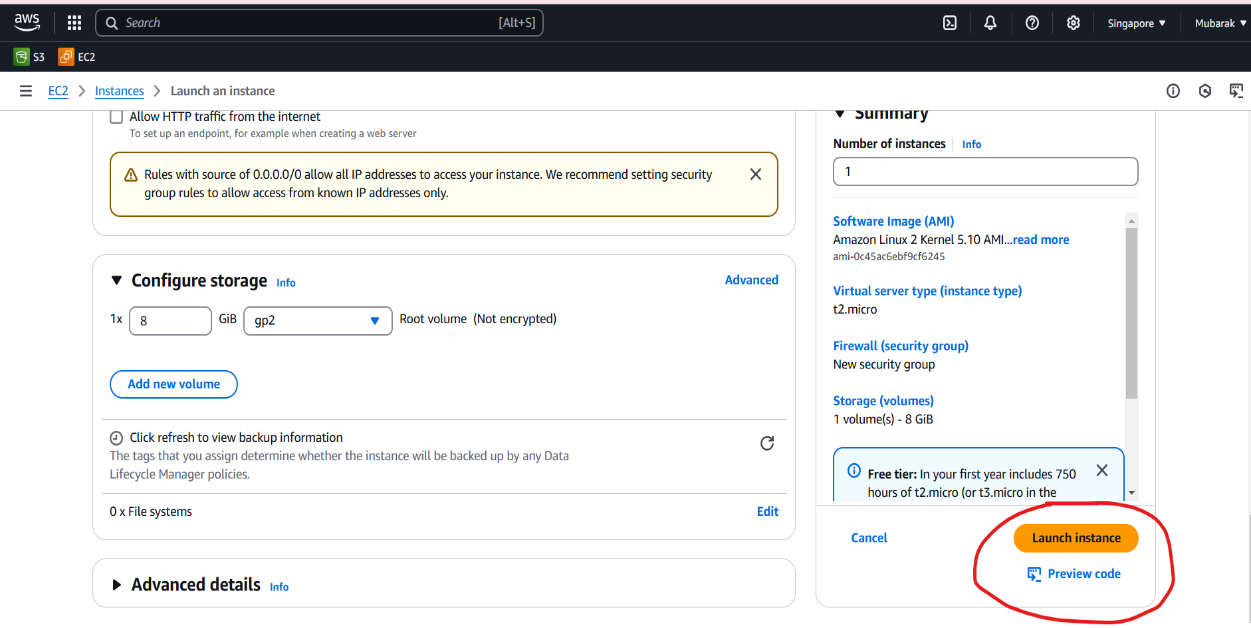


STEP 2 – create a on demand server with basic details . like tag name

Ami , instance type, key pair ,default network, security groups and storage .

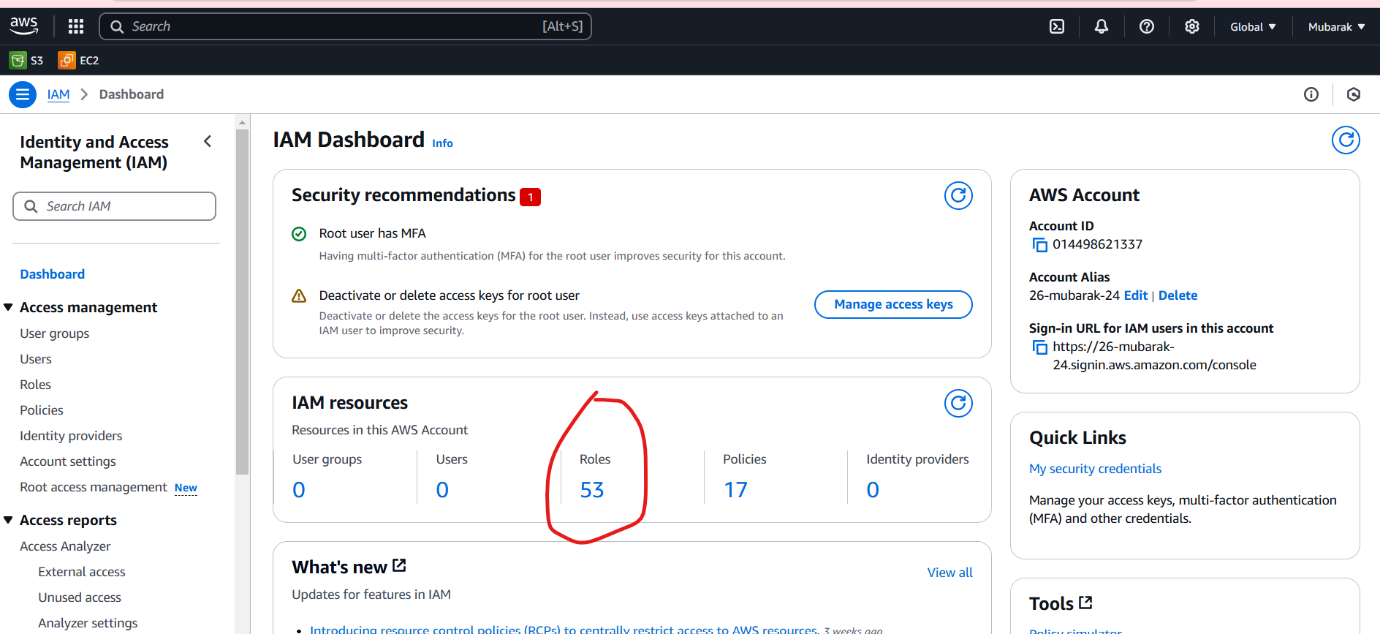


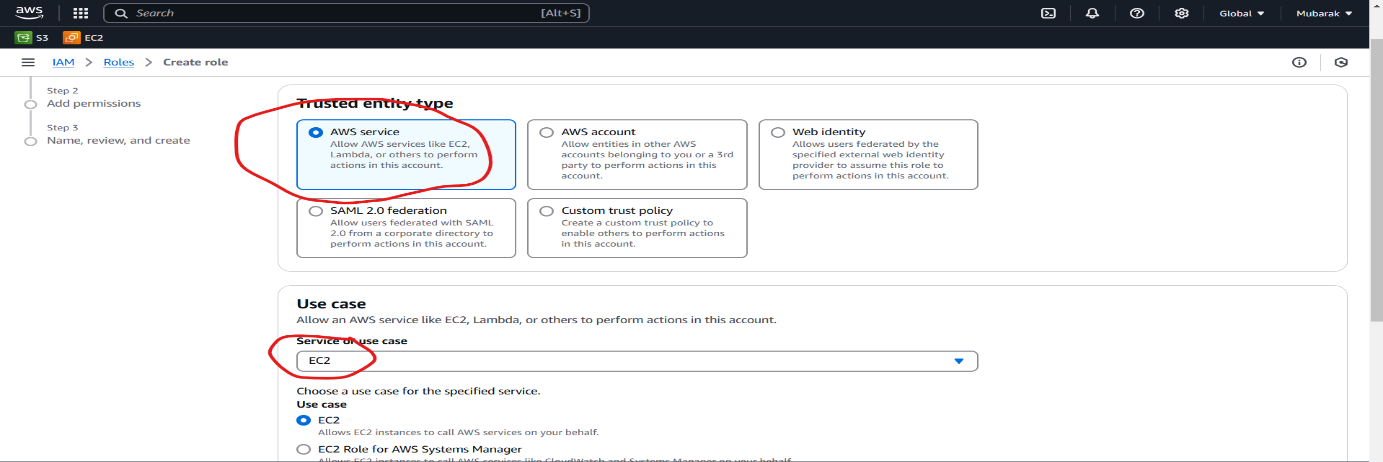


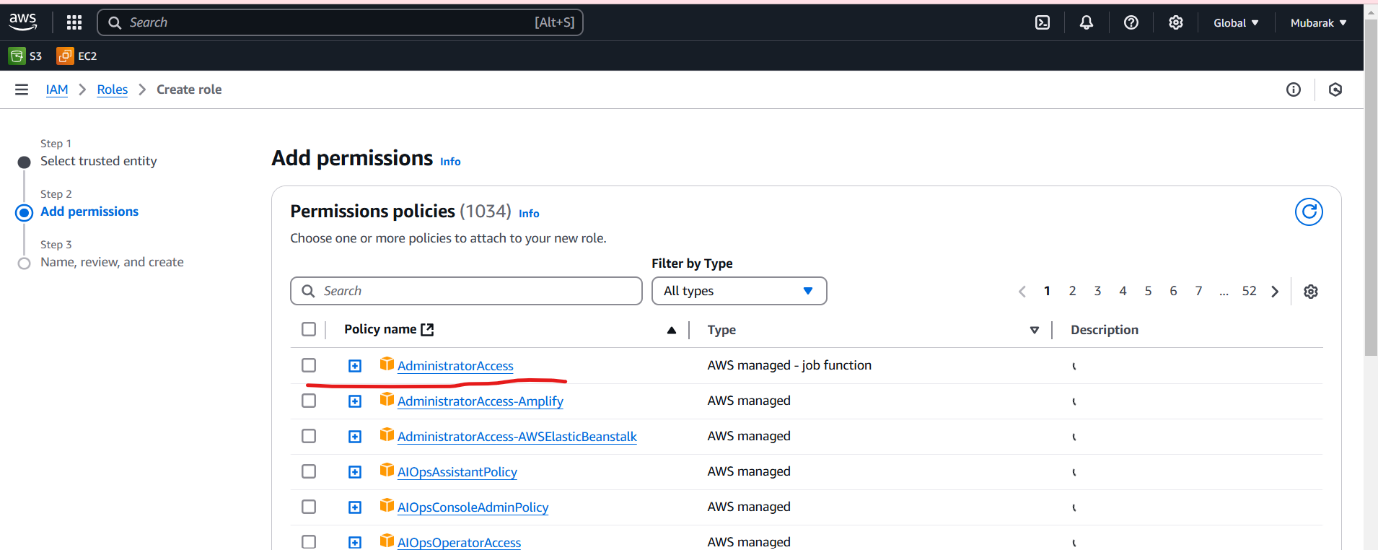


**before going to create ec2 machine first create a IAM Role**

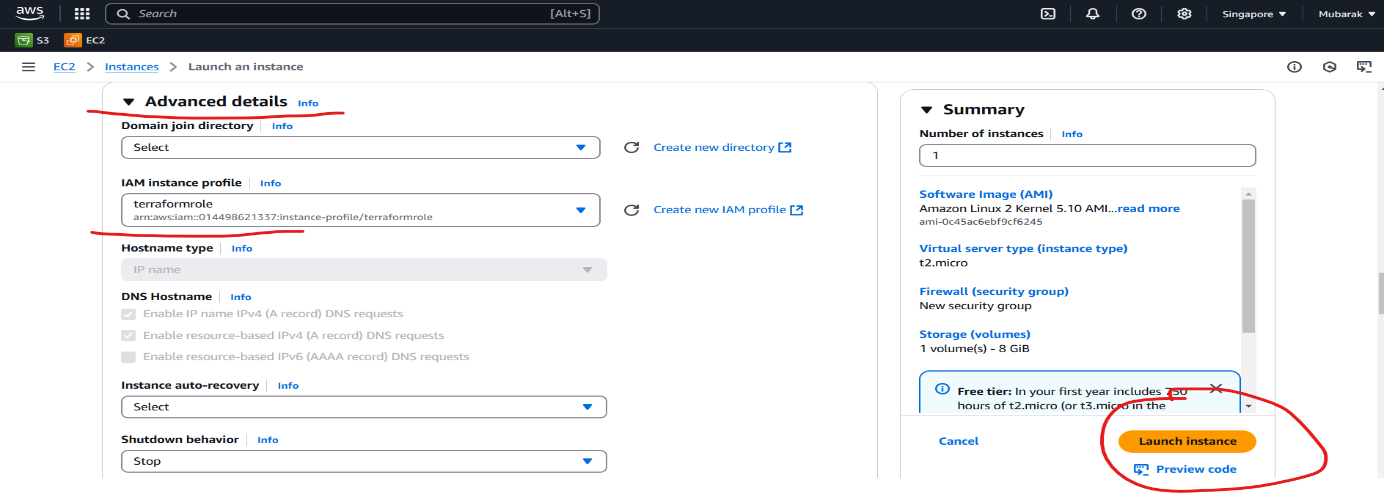
STEP 3 – before going to create ec2 machine first create a IAM Role into that server for giving a permission to terraform for access the AWS console. Go IAM – select Role – aws service - ec2- adminaccess.







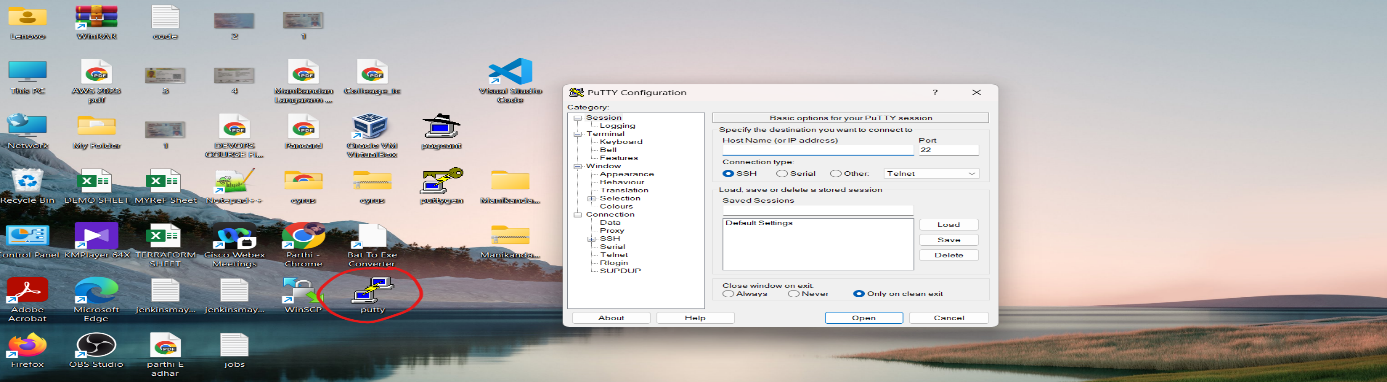
\*\*before going to create a ec2 server first create the IAM role first and attach the role to server where you can see the advance details – instance profile – add the role over there\*\*

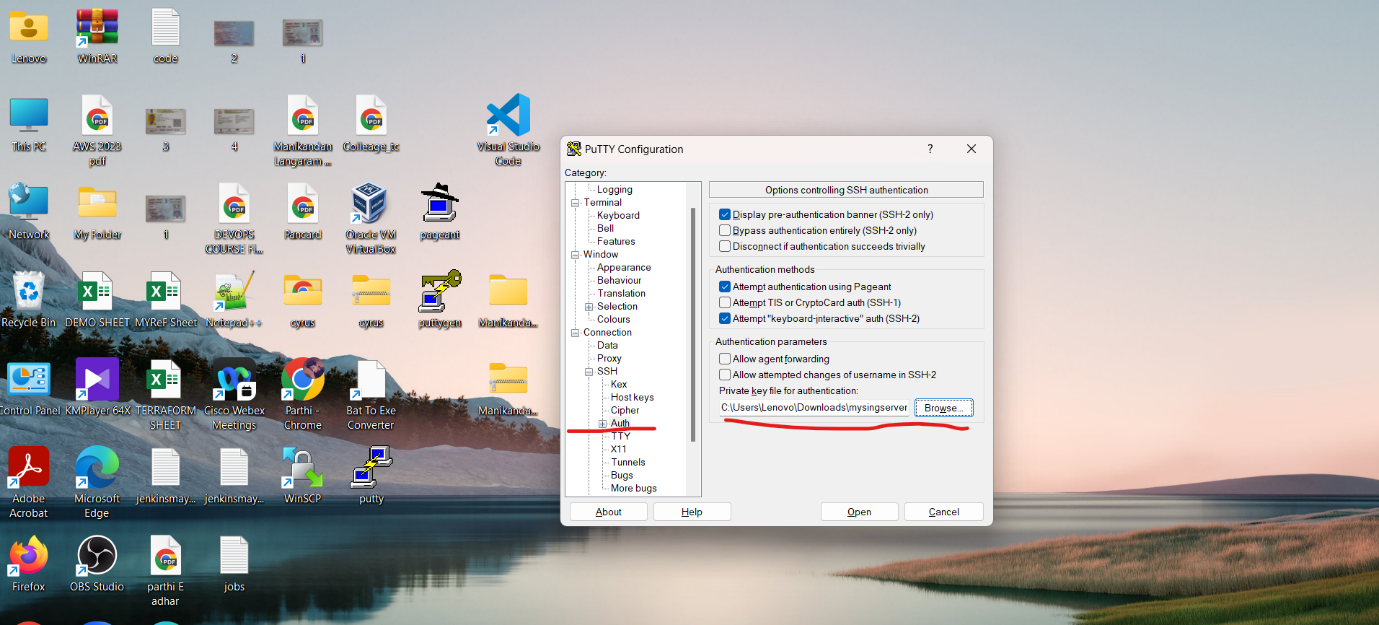


and finally create the server for terraform.

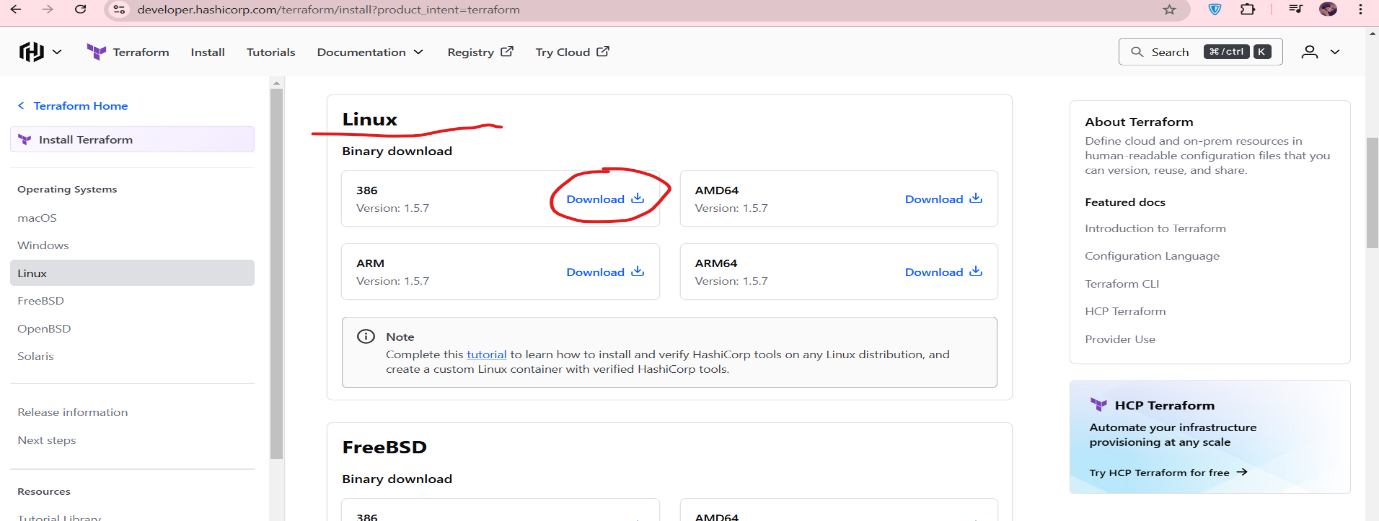
Now we login into the server using Putty and going to install terraform into that server.

STEP4 – login into putty, if you don’t have putty download the putty – in Brower search putty download then u can download that.



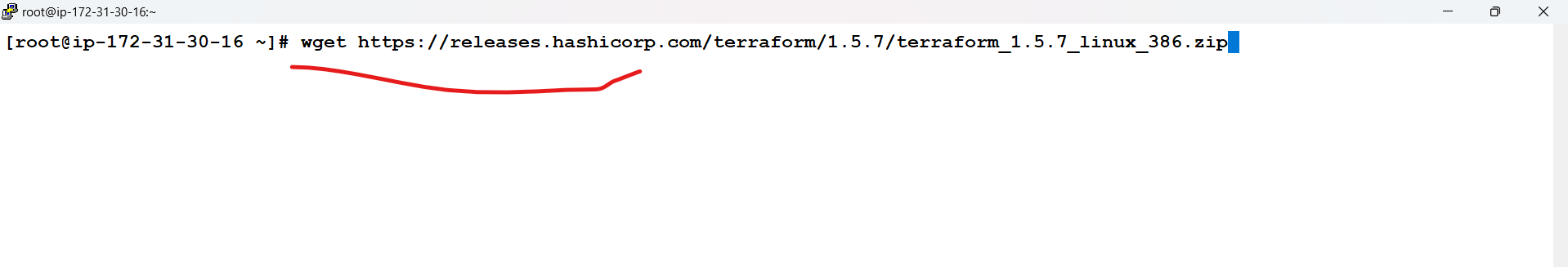


STEP5 – install the terraform in ec2 server using the terraform official page

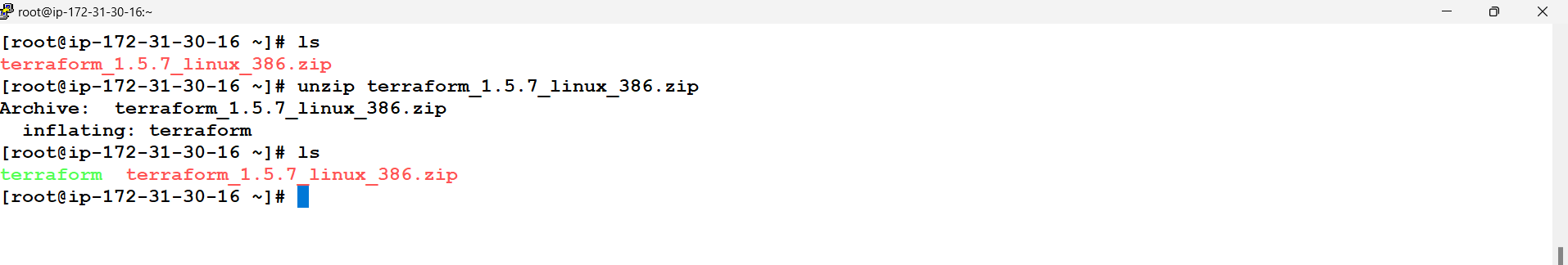
We can able to download the terraform version of 1.57 for stable version.

STEP6 – download the terraform for Linux server using command

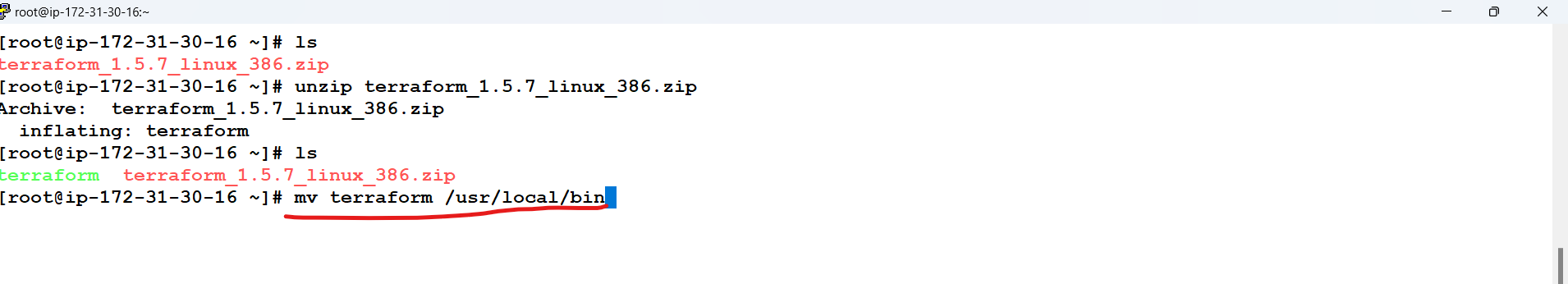
Wget <https://releases.hashicorp.com/terraform/1.5.7/terraform_1.5.7_linux_386.zip>

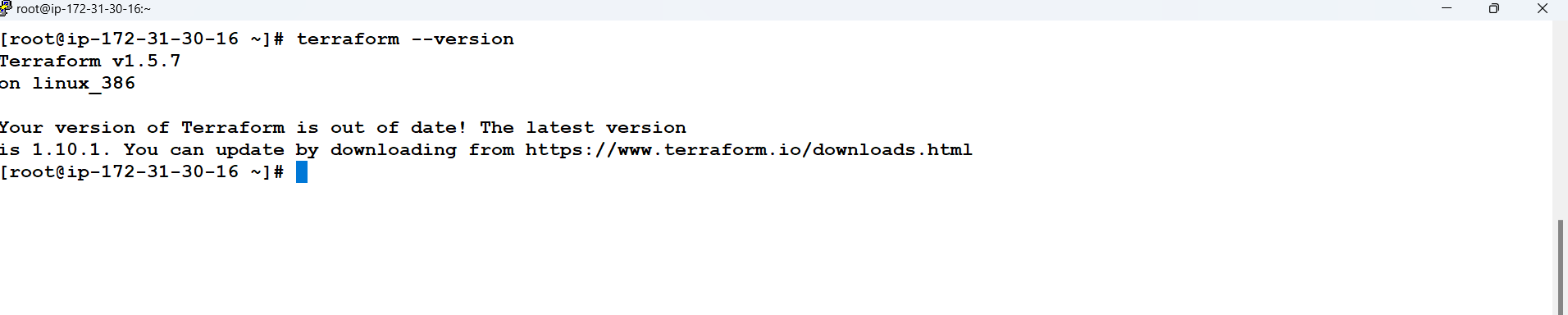


Unzip the terraform – using the command to unzip the file for terraform.



Move the file terraform to the bin path as well





Once installed the terraform check the version as well. after all, done we can write code for EKS cluster creation.

Download the Vs code editor tool for writing the terraform code for editing here the which we are going to use for creating the infra.

.

├── main.tf

├── variables.tf

├── outputs.tf

├── modules

│ ├── eks

│ │ ├── main.tf

│ │ ├── variables.tf

│ │ ├── outputs.tf

│ ├── vpc

│ ├── main.tf

│ ├── variables.tf

│ ├── outputs.tf

Main.tf – code

provider "aws" {

region = "us-west-2"

}

# VPC Module

module "vpc" {

source = "./modules/vpc"

cidr\_block = "10.0.0.0/16"

}

# EKS Module

module "eks" {

source = "./modules/eks"

cluster\_name = "my-eks-cluster"

vpc\_id = module.vpc.vpc\_id

subnet\_ids = module.vpc.subnet\_ids

desired\_capacity = 2

max\_size = 3

min\_size = 1

}

# Kubernetes Provider

provider "kubernetes" {

host = module.eks.cluster\_endpoint

cluster\_ca\_certificate = base64decode(module.eks.cluster\_ca\_certificate)

token = module.eks.cluster\_token

}

# Web Application Deployment

resource "kubernetes\_namespace" "web\_app" {

metadata {

name = "web-app"

}

}

VPC module code

modules/vpc/main.tf

resource "aws\_vpc" "main" {

cidr\_block = var.cidr\_block

}

resource "aws\_subnet" "subnet" {

count = 2

vpc\_id = aws\_vpc.main.id

cidr\_block = cidrsubnet(aws\_vpc.main.cidr\_block, 8, count.index)

availability\_zone = data.aws\_availability\_zones.available.names[count.index]

}

data "aws\_availability\_zones" "available" {}

output "vpc\_id" {

value = aws\_vpc.main.id

}

output "subnet\_ids" {

value = aws\_subnet.subnet[\*].id

}

modules/vpc/variables.tf

variable "cidr\_block" {

type = string

}

modules/vpc/outputs.tf

output "vpc\_id" {

value = aws\_vpc.main.id

}

output "subnet\_ids" {

value = aws\_subnet.subnet[\*].id

}

CODE FOR EKS MODULE

modules/eks/main.tf

resource "aws\_eks\_cluster" "eks" {

name = var.cluster\_name

role\_arn = aws\_iam\_role.eks\_role.arn

vpc\_config {

subnet\_ids = var.subnet\_ids

}

enabled\_cluster\_log\_types = ["api", "audit", "authenticator"]

}

resource "aws\_iam\_role" "eks\_role" {

name = "eks-role"

assume\_role\_policy = jsonencode({

Version = "2012-10-17",

Statement = [

{

Action = "sts:AssumeRole",

Effect = "Allow",

Principal = { Service = "eks.amazonaws.com" }

},

],

})

}

resource "aws\_iam\_role\_policy\_attachment" "eks\_policy" {

role = aws\_iam\_role.eks\_role.name

policy\_arn = "arn:aws:iam::aws:policy/AmazonEKSClusterPolicy"

}

output "cluster\_endpoint" {

value = aws\_eks\_cluster.eks.endpoint

}

output "cluster\_ca\_certificate" {

value = aws\_eks\_cluster.eks.certificate\_authority[0].data

}

output "cluster\_token" {

value = data.aws\_eks\_cluster\_auth.eks.token

}

data "aws\_eks\_cluster\_auth" "eks" {

name = aws\_eks\_cluster.eks.name

}

modules/eks/variables.tf

variable "cluster\_name" {

type = string

}

variable "vpc\_id" {

type = string

}

variable "subnet\_ids" {

type = list(string)

}

variable "desired\_capacity" {

type = number

}

variable "max\_size" {

type = number

}

variable "min\_size" {

type = number

}

modules/eks/outputs.tf

output "cluster\_endpoint" {

value = aws\_eks\_cluster.eks.endpoint

}

output "cluster\_ca\_certificate" {

value = aws\_eks\_cluster.eks.certificate\_authority[0].data

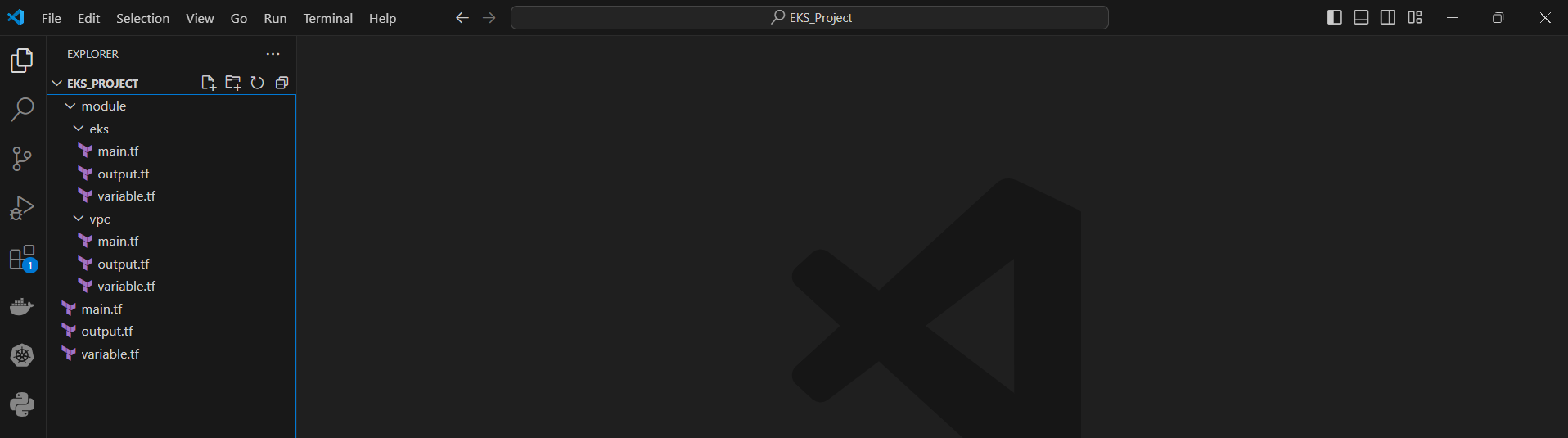
}

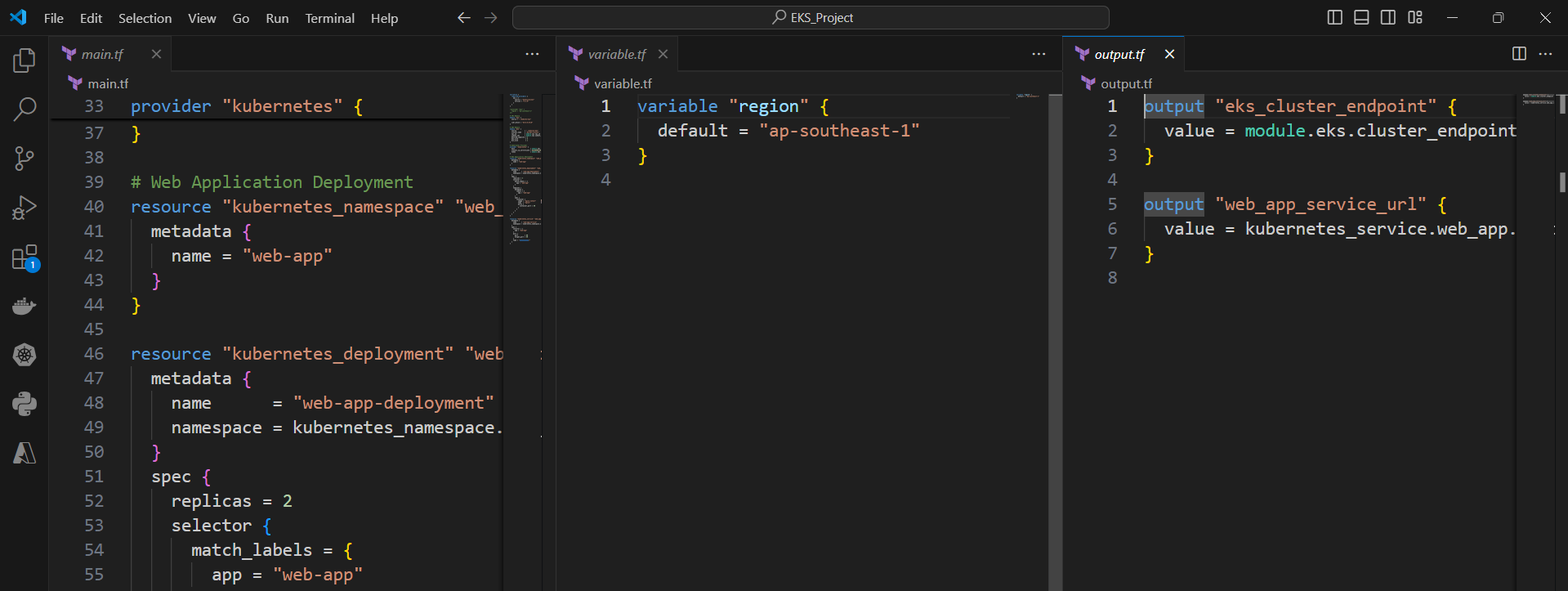
output "cluster\_token" {

value = data.aws\_eks\_cluster\_auth.eks.token

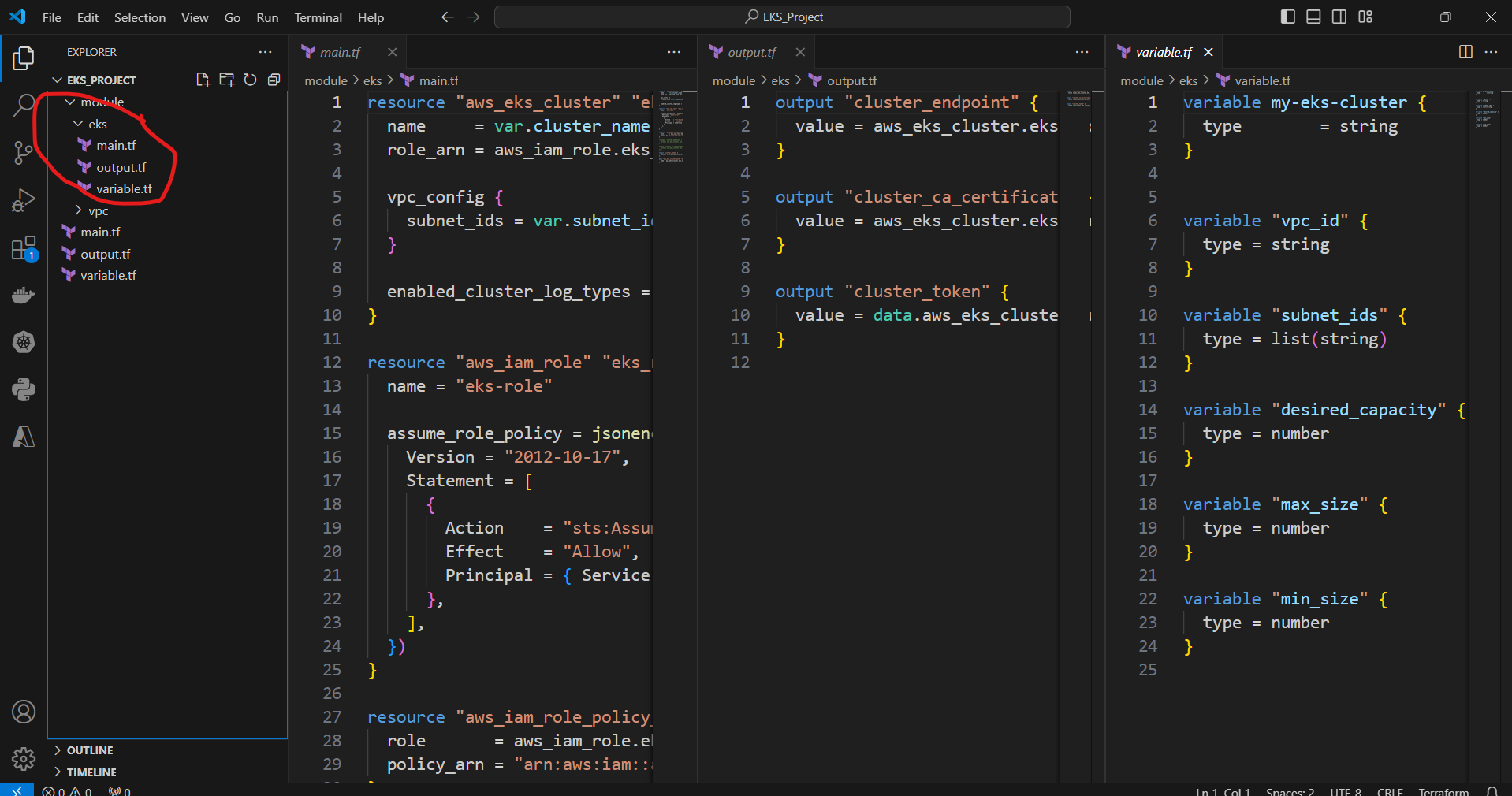
}

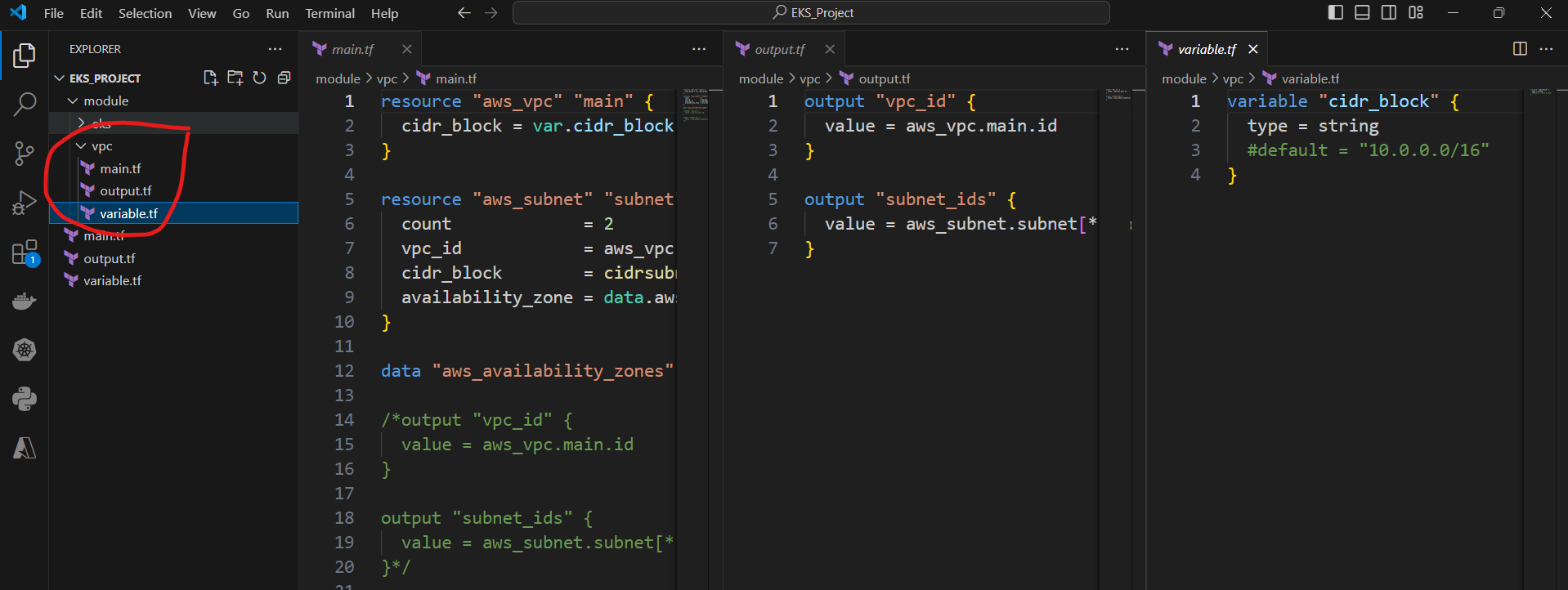
Create a EKS folder first and create separate module for eks and vpc and write codes as well as per the module setup like this.





This is main files main.tf and variable.tf and output.tf. after that we create the separate scripts for eks and vpc like this.





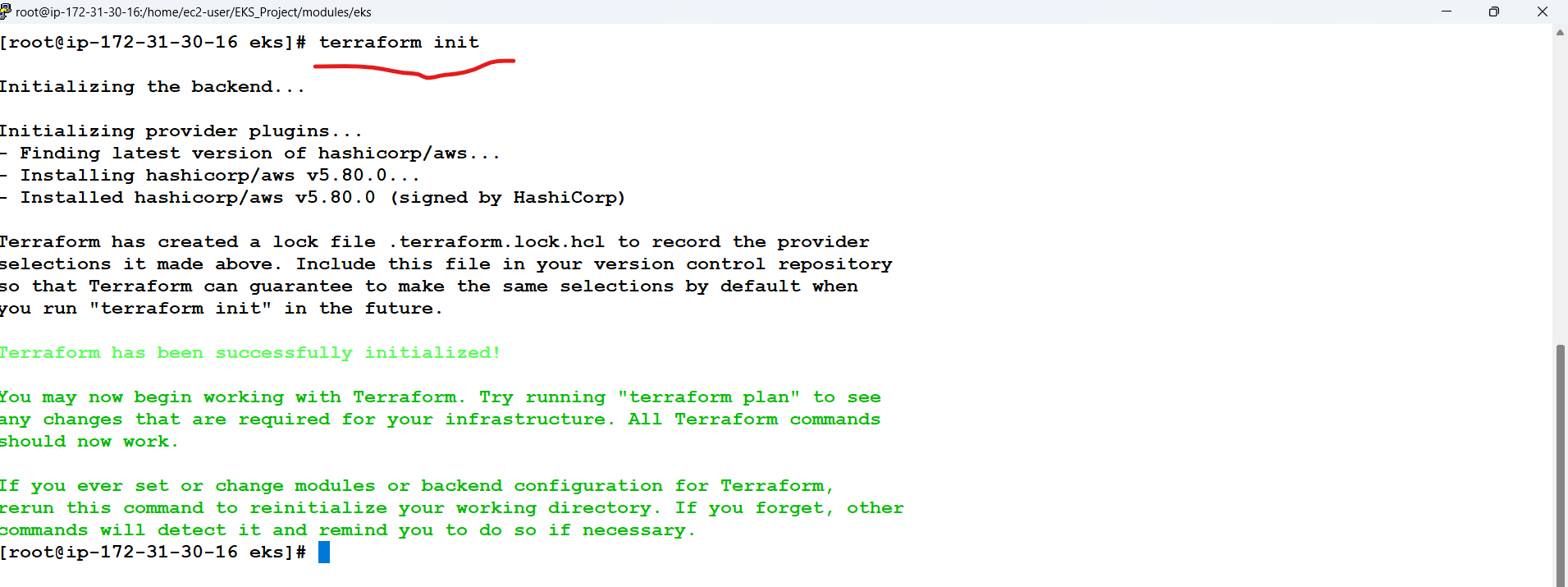
Once it all done copy the code to ec2 server and execute the code using the terraform commands

Login into the serve and create a separate folder of the code like

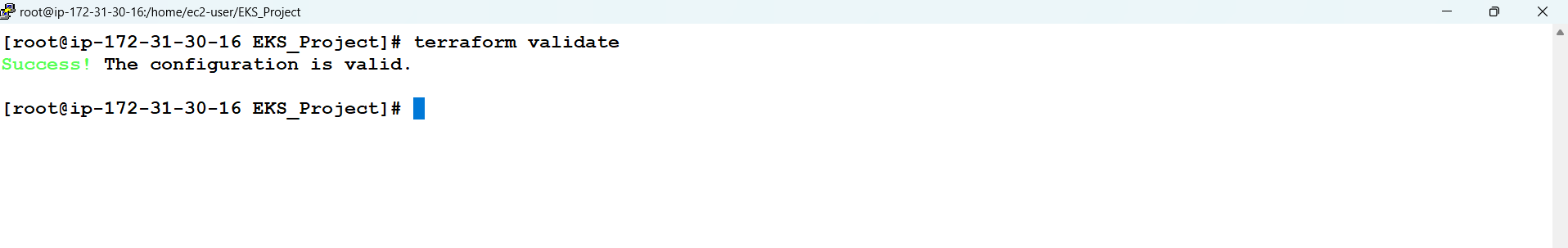
EKS\_Project which I have create and I copy paste all the code into the particular file as well

Once it all done run the terraform commands to execute the codes

First command – terraform init

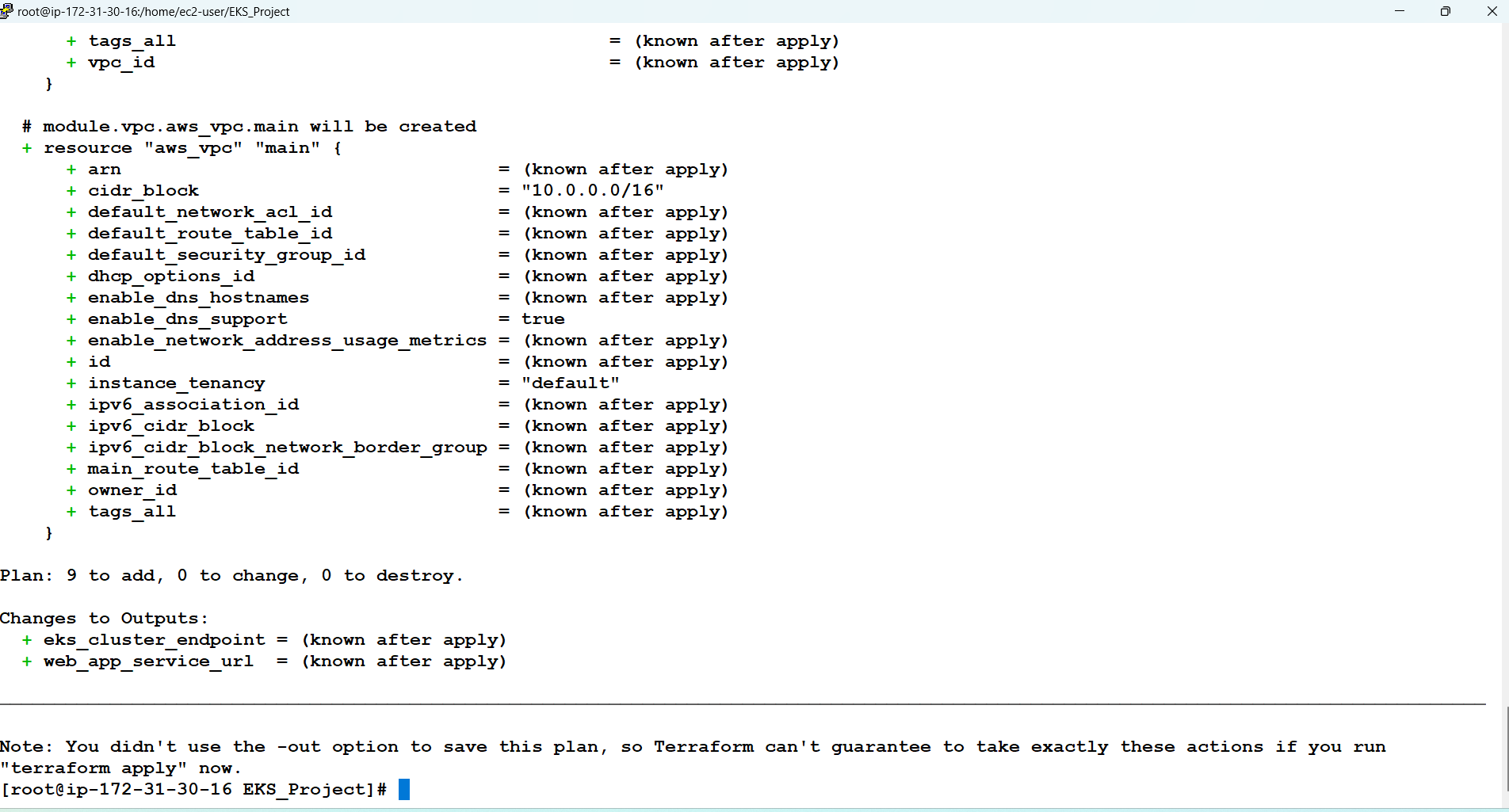


Second command – terraform validate

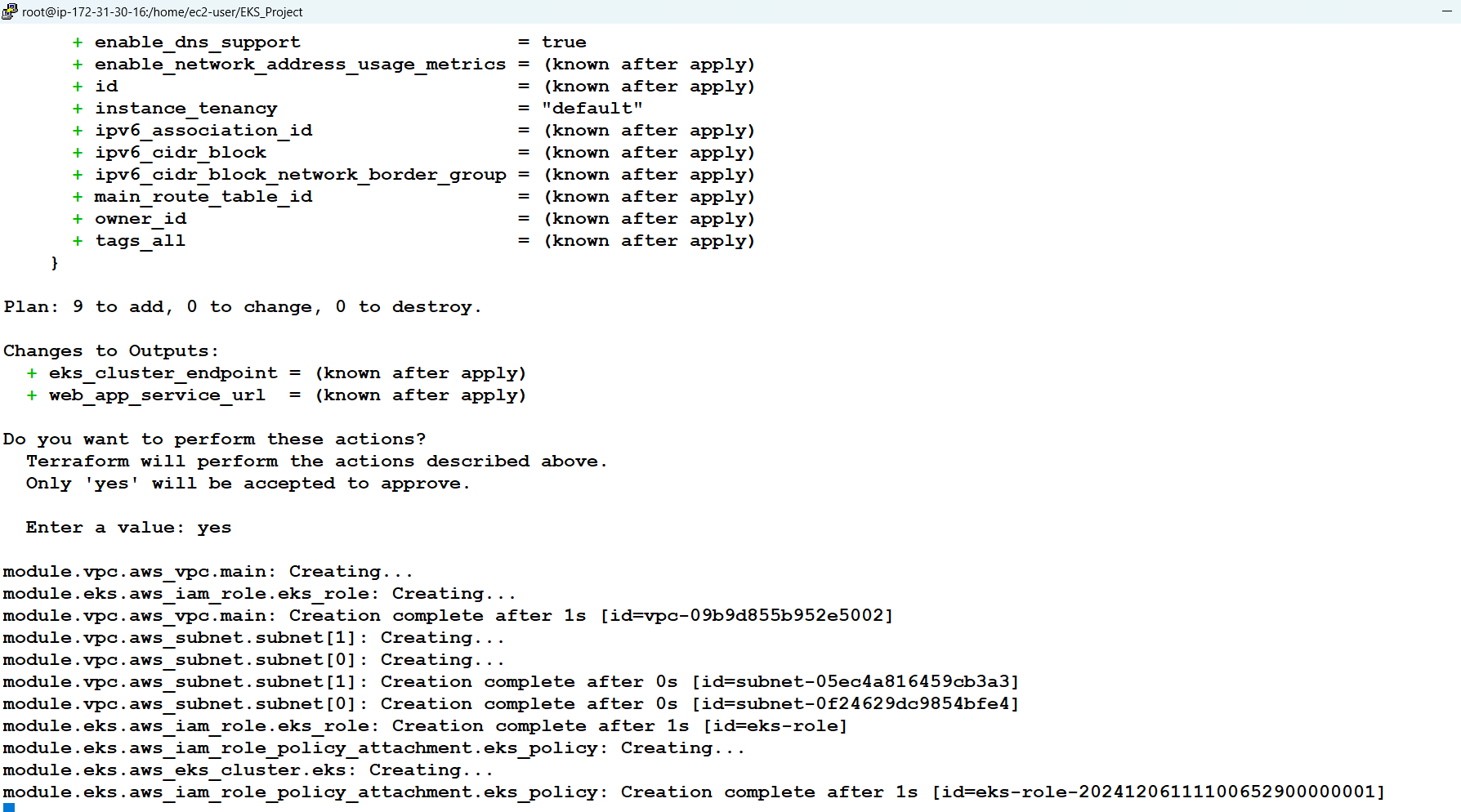


Once the code validated run the terraform plan command to check which

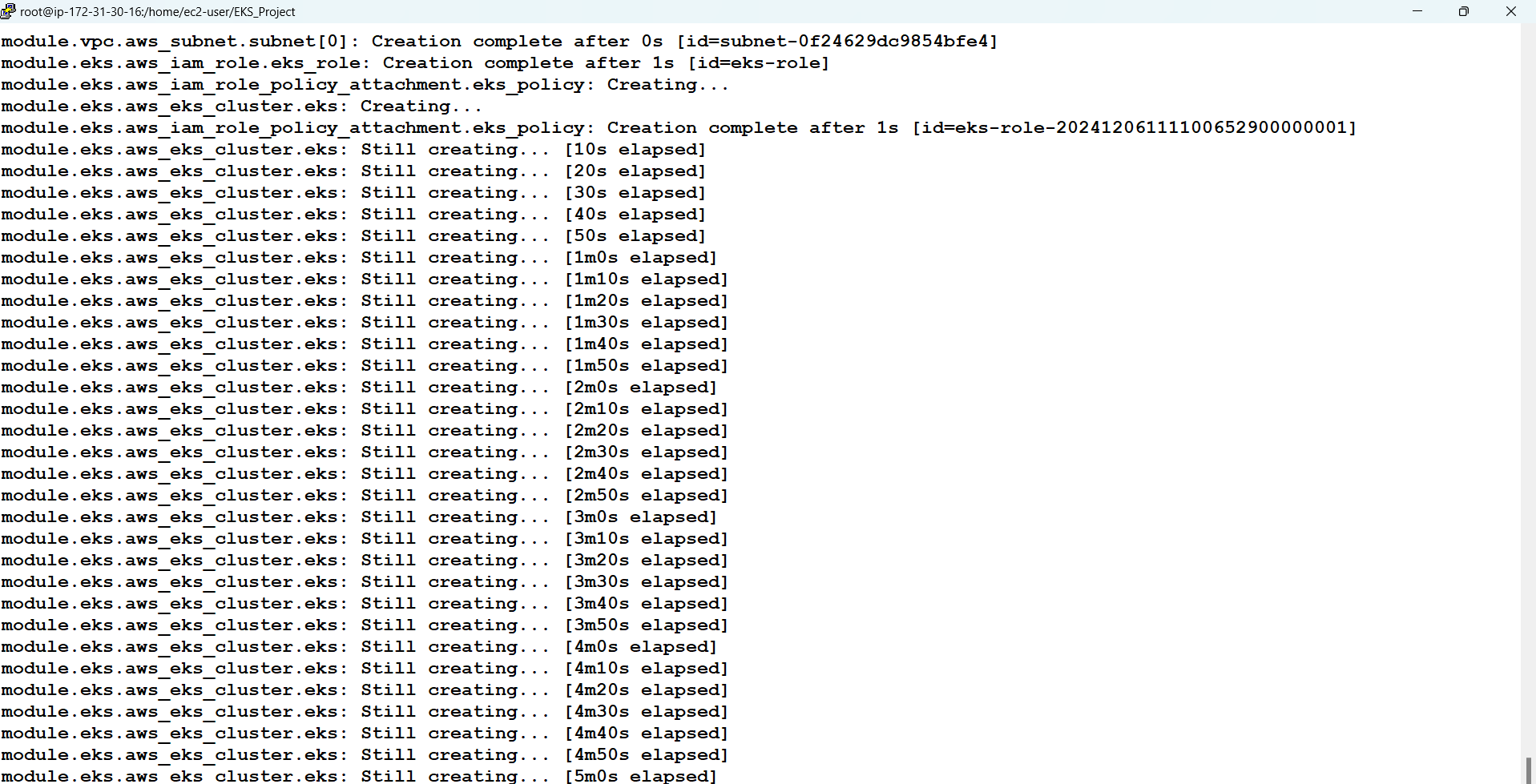
Resources were going to create.



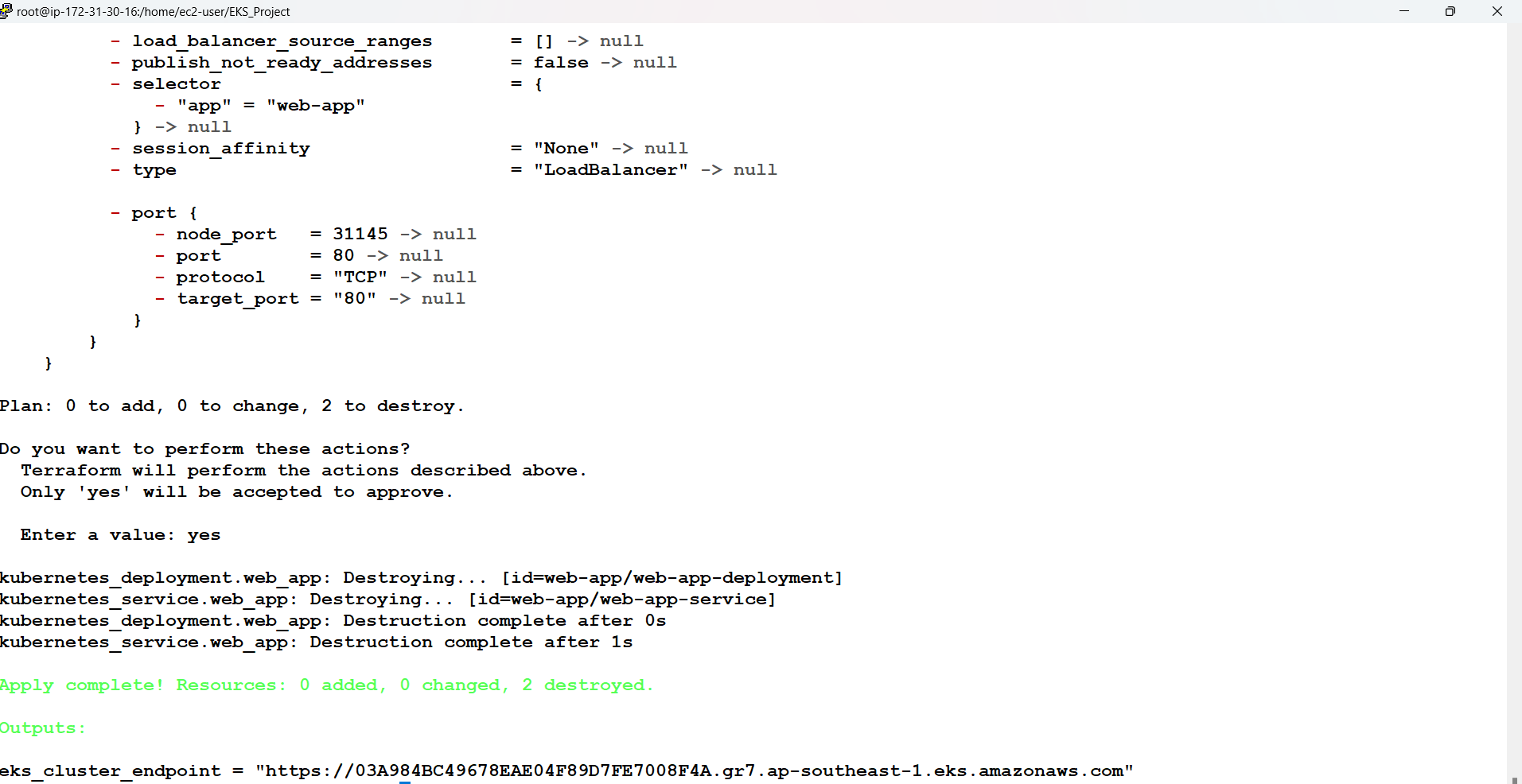
Final command terraform apply to execute the operation to create the infra setup for eks cluster.



It will take some to time to create will wait for some times

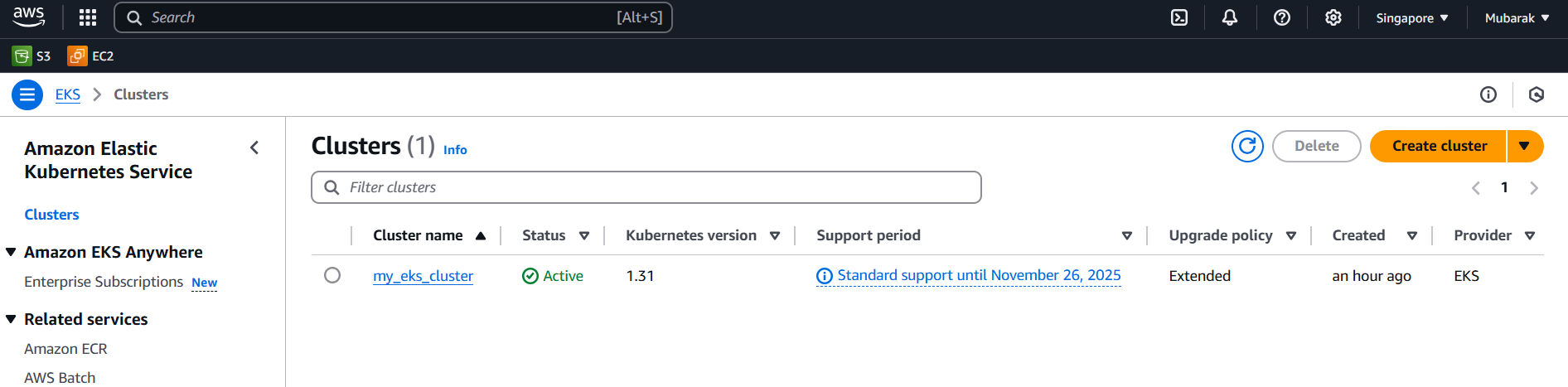


Once it completed we get output like this



We will eks endpoint for eks cluster.

Finally our EKS cluster and VPC are created in the console



Our final output

Once all finished destroy the eks cluster using terraform command

Command – terraform destroy.

---------------------------------------the end-----------------------------------------------------