LAB-1: Deploying a VM in AWS Using the Terraform Workflow :

In this hands-on lab, we will be following the Terraform workflow — Write > Plan > Apply — to deploy a virtual machine (VM) in AWS. After a successful deployment, we will then clean up our infrastructure and destroy the resource we created.

Steps:

Create a Directory and Write your Terraform code(write)

Step-1: create a new directory in the home directory called terraform\_code

mkdir terraform\_code

step-2: switch to the new directory

cd terraform\_code

step-3: using vim editor , create a file called main.tf where you will write your code:

vim main.tf

step-4: in the file, write the provided code that will be used to create the required VM(EC2 instance) in AWS:

provider "aws" {

region = "us-east-1"

}

resource "aws\_instance" "vm" {

ami = "DUMMY\_VALUE\_AMI\_ID"

subnet\_id = "DUMMY\_VALUE\_SUBNET\_ID"

instance\_type = "t3.micro"

tags = {

Name = "my-first-tf-node"

}

}

Step-5: press Escape and enter :wq to save and exit the file.

Paste the ami value into your code for the ami parameter, replacing the DUMMY\_VALUE\_AMI\_ID placeholder text

Paste the subnet\_id value into your code for the subnet\_id parameter, replacing the DUMMY\_VALUE\_SUBNET\_ID placeholder text.

Initialize and Review your code(plan)

1.Initialize the Terraform configuration and download the required providers:

terraform init

2.Review the actions that will be performed when you deploy your code:

terraform plan

In this case, it will create 1 resource: the EC2 instance you configured in your code.

Deploy your Terraform Code(Apply), Verify your resources , and clean up

1. Deploy the code:

terraform apply

2. When prompted, type *yes* and press **Enter**.

3.Once the code has executed successfully, note in the output that 1 resource has been created.

4. Back in the CLI, remove the infrastructure you just created:

terraform destroy

5.in the plan output , notice that it will destroy 1 resource: the EC2 instance you just created.