

EXPERIMENT – 9

Name: - Pulkit Singh Kathayat

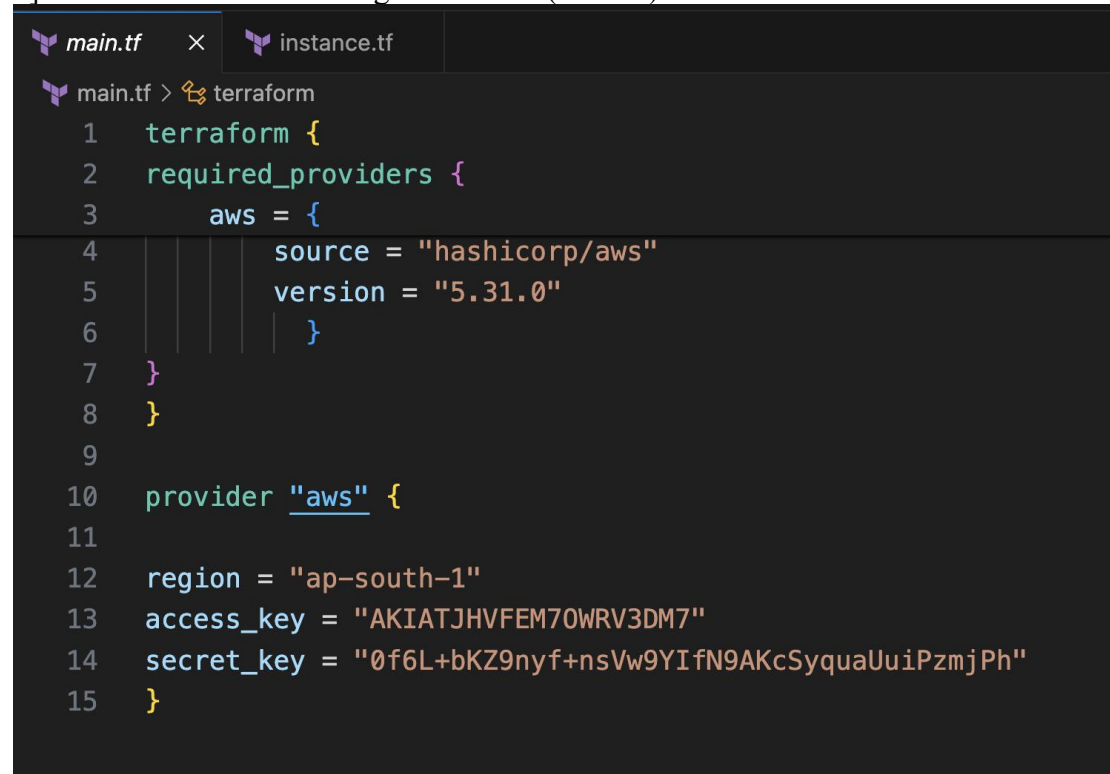
Batch – 3 [DevOps Non-Hons]

SAP ID- 500094778

Subject – System Provisioning and Configuration Management Lab

Aim: Creating Multiple EC2 Instances with for each in Terraform.

1] Create a Terraform Configuration File (main.tf)

A screenshot of a code editor with two tabs at the top: 'main.tf' (active) and 'instance.tf'. The 'main.tf' tab shows a Terraform configuration file. The code is as follows:

```
1 terraform {
2   required_providers {
3     aws = {
4       source = "hashicorp/aws"
5       version = "5.31.0"
6     }
7   }
8 }
9
10 provider "aws" {
11
12   region = "ap-south-1"
13   access_key = "AKIATJHVFEM70WRV3DM7"
14   secret_key = "0f6L+bKZ9nyf+nsVw9YIfN9AKcSyquaUuiPzmjPh"
15 }
```

2] Create new file name as “instance.tf”

```
main.tf instance.tf ×
instance.tf > variable "instances" > default > instance3 > instance_type
1 resource "aws_instance" "ec2_instances" {
9 }
10
11 variable "instances" {
12     description = "Map of EC2 instances with settings"
13     default = {
14         "instance1" = {
15             ami = "ami-03f4878755434977f"
16             instance_type="t2.micro"
17         },
18         "instance2"={
19             ami = "ami-03f4878755434977f"
20             instance_type="t2.micro"
21         },
22         "instance3"={
23             ami = "ami-03f4878755434977f"
24             instance_type="t2.micro"
25         }
26     }
27 }
28 }
```

3] Initialize Terraform using command “terraform init”

```
Terraform-9 — -zsh — 85x25
pulkitkathayat@192 Documents % cd Terraform-9
pulkitkathayat@192 Terraform-9 % terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.31.0"...
- Installing hashicorp/aws v5.31.0...
- Installed hashicorp/aws v5.31.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
pulkitkathayat@192 Terraform-9 %
```

4] Validate it using command “terraform validate

```
Terrafrom-9 --zsh --85x25
pulkitkathayat@192 Terrafrom-9 % terraform validate
Success! The configuration is valid.

pulkitkathayat@192 Terrafrom-9 %
```

5] Check the Plan using command “terraform plan”

```
pulkitkathayat@192 Terrafrom-9 % terraform plan

Terraform used the selected providers to generate the following execution plan.
Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# aws_instance.ec2_instances["instance1"] will be created
+ resource "aws_instance" "ec2_instances" {
  + ami                        = "ami-03f4878755434977f"
  + arn                      = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone         = (known after apply)
  + cpu_core_count            = (known after apply)
  + cpu_threads_per_core      = (known after apply)
  + disable_api_stop          = (known after apply)
  + disable_api_termination   = (known after apply)
  + ebs_optimized              = (known after apply)
```

```
+ ebs_optimized              = (known after apply)
+ get_password_data          = false
+ host_id                    = (known after apply)
+ host_resource_group_arn    = (known after apply)
+ iam_instance_profile       = (known after apply)
+ id                         = (known after apply)
+ instance_initiated_shutdown_behavior = (known after apply)
+ instance_lifecycle         = (known after apply)
+ instance_state              = (known after apply)
+ instance_type               = "t2.micro"
+ ipv6_address_count          = (known after apply)
+ ipv6_addresses              = (known after apply)
+ key_name                    = (known after apply)
+ monitoring                  = (known after apply)
+ outpost_arn                 = (known after apply)
+ password_data               = (known after apply)
+ placement_group             = (known after apply)
+ placement_partition_number = (known after apply)
+ primary_network_interface_id = (known after apply)
+ private_dns                 = (known after apply)
+ private_ip                  = (known after apply)
+ public_dns                  = (known after apply)
+ public_ip                   = (known after apply)
+ secondary_private_ips       = (known after apply)
+ security_groups              = (known after apply)
+ source_dest_check           = true
+ spot_instance_request_id    = (known after apply)
+ subnet_id                   = (known after apply)
+ tags                        = {
  + "Name" = "EC2-Instance-instance3"
}
+ tags_all                    = {
  + "Name" = "EC2-Instance-instance3"
}
+ tenancy                     = (known after apply)
+ user_data                   = (known after apply)
+ user_data_base64            = (known after apply)
+ user_data_replace_on_change = false
+ vpc_security_group_ids      = (known after apply)
}

Plan: 3 to add, 0 to change, 0 to destroy.

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee
to take exactly these actions if you run "terraform apply" now.
pulkitkathayat@192 Terrafrom-9 %
```