ANSHIKA SRIVASTAVA
ROLL NUMBER – R2142220907
SAP ID – 500107049

LAB EXERCISE 6

Lab Exercise 6– Terraform Variables

Objective:

Learn how to define and use variables in Terraform configuration.

Prerequisites:

- Install Terraform on your machine.

Steps:

1. Create a Terraform Directory:

- Create a new directory for your Terraform project.

mkdir terraform-variables

cd terraform-variables

```
anshi@HP MINGW64 /d/Academics/SPCM Lab
$ mkdir terraform-variables

anshi@HP MINGW64 /d/Academics/SPCM Lab
$ cd terraform-variables
```

2. Create a Terraform Configuration File:

- Create a file named main.tf within your project directory.

 # main.tf

```
resource "aws_instance" "myinstance-1" {
  ami = var.myami
  instance_type = var.my_instance_type
  count = var.mycount
```

```
  tags = {
    Name= "My Instance"
  }
}
```

```
resource "aws_instance" "Anshikainstance-1" {
    ami = var.myami
    instance_type = var.my_instance_type
    count = var.mycount
    tags = {
      Name= "Anshika Instance"
    }
}

```

3. Define Variables:

- Open a new file named variables.tf. Define variables for region, ami, and instance_type.

 # variables.tf

```
variable "myami" {
  type = string
  default = "ami-08718895af4dfa033"
}

variable "mycount" {

  type = number
  default = 5
}

variable "my_instance_type" {
   type = string
   default = "t2.micro"
```

```
}
```

variables.tf > variable "my_instance_type"

```
variable "myami" {
  type = string
  default = "ami-09423ec3aa48e9438"
}

variable "mycount" {

  type = number
  default = 5
}

variable "my_instance_type" {
    type = string
    default = "t3.micro"
}
```

4. Initialize and Apply:

- Run the following Terraform commands to initialize and apply the configuration.

terraform init

```
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.83.0"...
- Installing hashicorp/aws v5.83.0...
- Installed hashicorp/aws v5.83.0 (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.
```

terraform plan

```
PS D:\Academics\SPCM Lab\terraform-variables> 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.Anshikainstance-1[0] will be created
  + resource "aws_instance" "Anshikainstance-1" {
      + ami                                  = "ami-08718895af4dfa033"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
```

```
      + maintenance_options (known after apply)

      + metadata_options (known after apply)

      + network_interface (known after apply)

      + private_dns_name_options (known after apply)

      + root_block_device (known after apply)
    }

  Plan: 5 to add, 0 to change, 0 to destroy.
```

terraform apply -auto-approve

```
PS D:\Academics\SPCM Lab\terraform-variables> terraform apply -auto-approve

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.Anshikainstance-1[0] will be created
  + resource "aws_instance" "Anshikainstance-1" {
      + ami                                  = "ami-09423ec3aa48e9438"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
```

```
aws_instance.Anshikainstance-1[2]: Creating...
aws_instance.Anshikainstance-1[1]: Still creating... [10s elapsed]
aws_instance.Anshikainstance-1[0]: Still creating... [10s elapsed]
aws_instance.Anshikainstance-1[3]: Still creating... [10s elapsed]
aws_instance.Anshikainstance-1[4]: Still creating... [10s elapsed]
aws_instance.Anshikainstance-1[2]: Still creating... [10s elapsed]
aws_instance.Anshikainstance-1[0]: Creation complete after 15s [id=i-09b7512831521e78d]
aws_instance.Anshikainstance-1[2]: Creation complete after 15s [id=i-0efd01b20f1418103]
aws_instance.Anshikainstance-1[3]: Creation complete after 15s [id=i-0a9358d51a39145fb]
aws_instance.Anshikainstance-1[1]: Creation complete after 15s [id=i-04ab31ee89cb0ba45]
aws_instance.Anshikainstance-1[4]: Creation complete after 15s [id=i-0775df993c75e9472]

Apply complete! Resources: 5 added, 0 changed, 0 destroyed.
PS D:\Academics\SPCM Lab\terraform-variables>
```

Observe how the region changes based on the variable override.

Anshika Instance	i-04ab31ee89cb0ba45	Running	t3.micro	3/3 checks passec	View alarms +	eu-north-1b	ec2-13-60-7
Anshika Instance	i-0efd01b20f1418103	Running	t3.micro	3/3 checks passec	View alarms +	eu-north-1b	ec2-13-53-7
Anshika Instance	i-09b7512831521e78d	Running	t3.micro	3/3 checks passec	View alarms +	eu-north-1b	ec2-13-48-5
Anshika Instance	i-0775df993c75e9472	Running	t3.micro	3/3 checks passec	View alarms +	eu-north-1b	ec2-16-171
Anshika Instance	i-0a9358d51a39145fb	Running	t3.micro	3/3 checks passec	View alarms +	eu-north-1b	ec2-13-51-7

5. Clean Up:

After testing, you can clean up resources.

terraform destroy

```
PS D:\Academics\SPCM Lab\terraform-variables> terraform destroy
aws_instance.Anshikainstance-1[0]: Refreshing state... [id=i-09b7512831521e78d]
aws_instance.Anshikainstance-1[1]: Refreshing state... [id=i-04ab31ee89cb0ba45]
aws_instance.Anshikainstance-1[3]: Refreshing state... [id=i-0a9358d51a39145fb]
aws_instance.Anshikainstance-1[2]: Refreshing state... [id=i-0efd01b20f1418103]
aws_instance.Anshikainstance-1[4]: Refreshing state... [id=i-0775df993c75e9472]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated
with the following symbols:
  - destroy
```

```
aws_instance.Anshikainstance-1[4]: Still destroying... [id=i-0775df993c75e9472, 1m0s elapsed]
aws_instance.Anshikainstance-1[0]: Still destroying... [id=i-09b7512831521e78d, 1m0s elapsed]
aws_instance.Anshikainstance-1[1]: Still destroying... [id=i-04ab31ee89cb0ba45, 1m0s elapsed]
aws_instance.Anshikainstance-1[3]: Still destroying... [id=i-0a9358d51a39145fb, 1m0s elapsed]
aws_instance.Anshikainstance-1[1]: Destruction complete after 1m2s
aws_instance.Anshikainstance-1[0]: Destruction complete after 1m2s
aws_instance.Anshikainstance-1[4]: Still destroying... [id=i-0775df993c75e9472, 1m10s elapsed]
aws_instance.Anshikainstance-1[3]: Still destroying... [id=i-0a9358d51a39145fb, 1m10s elapsed]
aws_instance.Anshikainstance-1[4]: Still destroying... [id=i-0775df993c75e9472, 1m20s elapsed]
aws_instance.Anshikainstance-1[3]: Still destroying... [id=i-0a9358d51a39145fb, 1m20s elapsed]
aws_instance.Anshikainstance-1[4]: Destruction complete after 1m23s
aws_instance.Anshikainstance-1[3]: Destruction complete after 1m23s

Destroy complete! Resources: 5 destroyed.
```

Confirm the destruction by typing yes.

6. Conclusion:

This lab exercise introduces you to Terraform variables and demonstrates how to use them in your configurations. Experiment with different variable values and overrides to understand their impact on the infrastructure provisioning process.