**Pratik Agrawal**

**500123601**

**Devops B2**

# 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
```

## 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"
  }
}
```

## 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"
}
```

## 4. Initialize and Apply:

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

```
terraform init
```

**terraform plan**

**terraform apply -auto-approve**

Observe how the region changes based on the variable override.

```
PS C:\Users\prati\OneDrive\Documents\Devsecops lab\terraform-variables> terraform init
Initializing the backend...
Initializing provider plugins...

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.
PS C:\Users\prati\OneDrive\Documents\Devsecops lab\terraform-variables> terraform plan

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration
and found no differences, so no changes are needed.
PS C:\Users\prati\OneDrive\Documents\Devsecops lab\terraform-variables> terraform apply -auto-approve

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration
and found no differences, so no changes are needed.

Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
```

# 5. Clean Up:

After testing, you can clean up resources.

**terraform destroy**

Confirm the destruction by typing yes.

```
PS C:\Users\prati\OneDrive\Documents\Devsecops lab\terraform-variables> terraform destroy

No changes. No objects need to be destroyed.

Either you have not created any objects yet or the existing objects were
already deleted outside of Terraform.

Destroy complete! Resources: 0 destroyed.
```

# 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.