Lab Exercise 6– Terraform Variables

Objective:

Learn how to define and use variables in Terraform configuration.

Prerequisites:

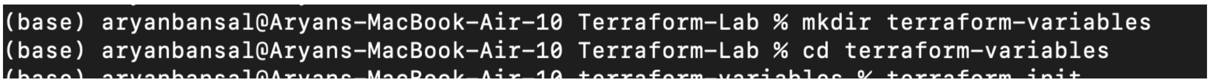
- Install Terraform on your machine.

Steps:

1. Create a Terraform Directory:

- Create a new directory for your Terraform project.

```
mkdir terraform-variables

cd terraform-variables
```



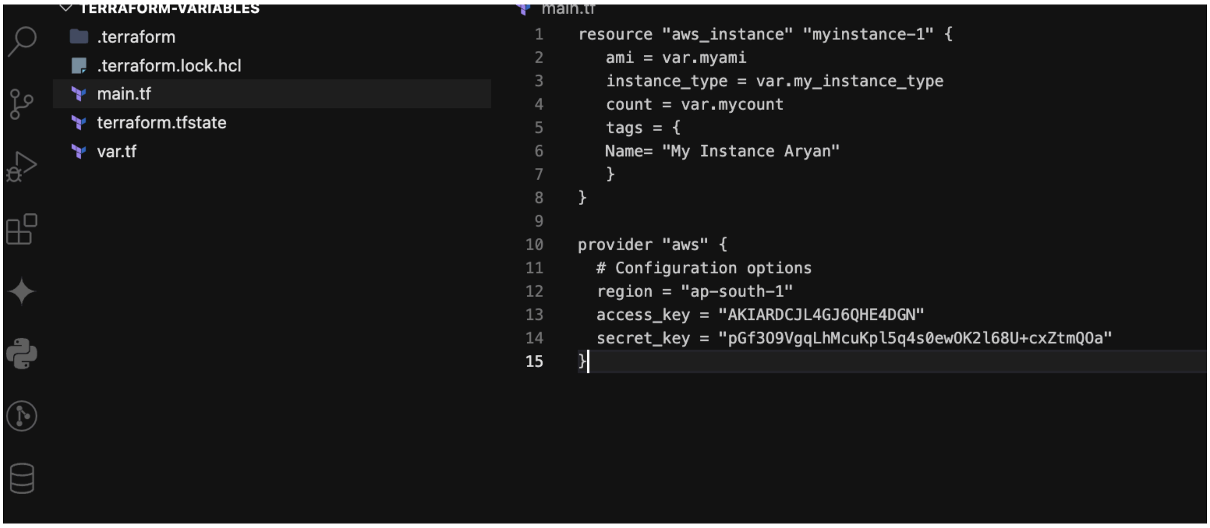
```
(base) aryanbansal@Aryans-MacBook-Air-10 Terraform-Lab % mkdir terraform-variables
(base) aryanbansal@Aryans-MacBook-Air-10 Terraform-Lab % cd terraform-variables
(base) aryanbansal@Aryans-MacBook-Air-10 terraform-variables % terraform init
```

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
```



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

provider "aws" {
  # Configuration options
  region = "ap-south-1"
  access_key = "AKIARDCJL4GJ6QHE4DGN"
  secret_key = "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"
}
```

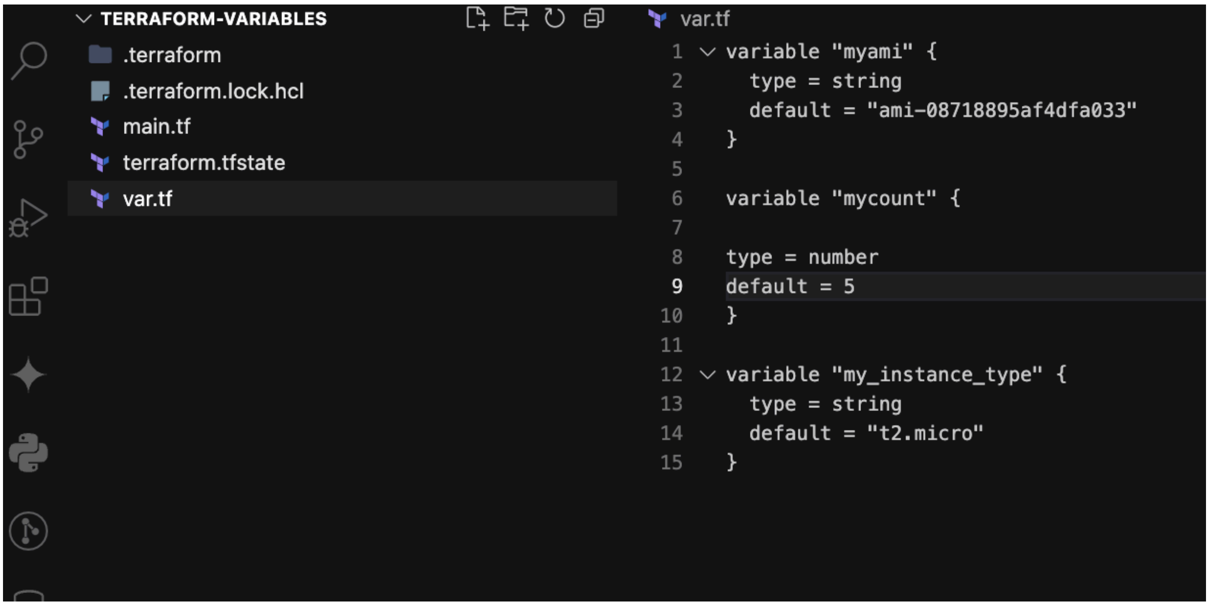
```
  tags = {
    Name= "My Instance"
  }
}
```

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"
}
```

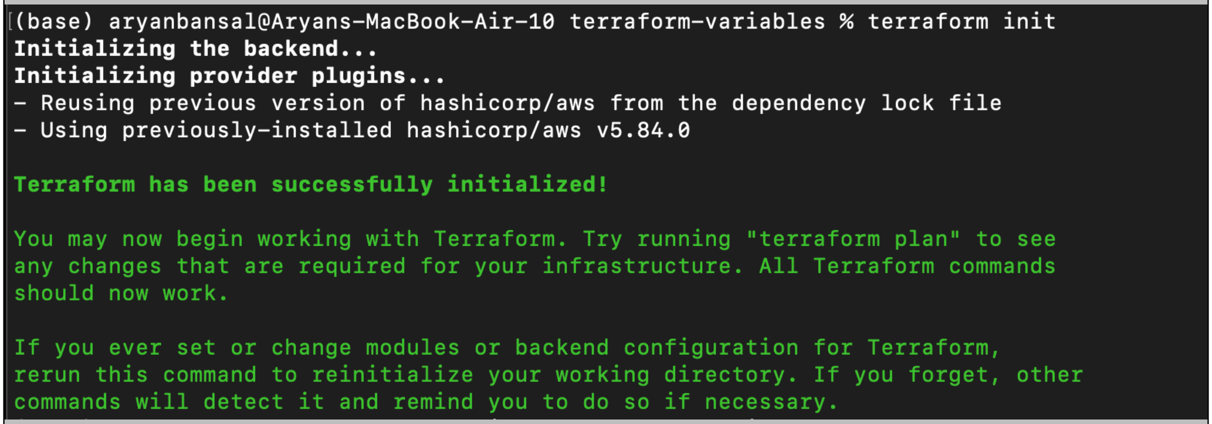


```
variable "myami" {
  type = string
  default = "ami-08718895af4dfa033"
}

variable "mycount" {

type = number
default = 5
}

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

4. Initialize and Apply:

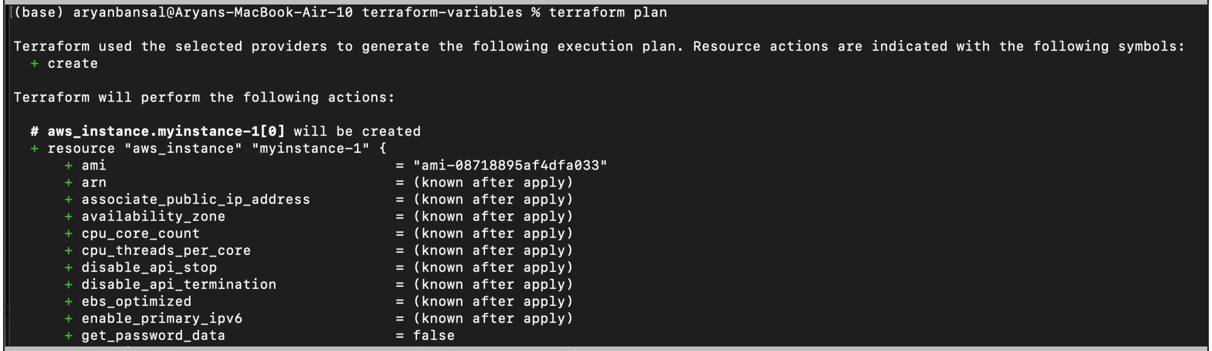
- Run the following Terraform commands to initialize and apply the configuration.

```
terraform init
```



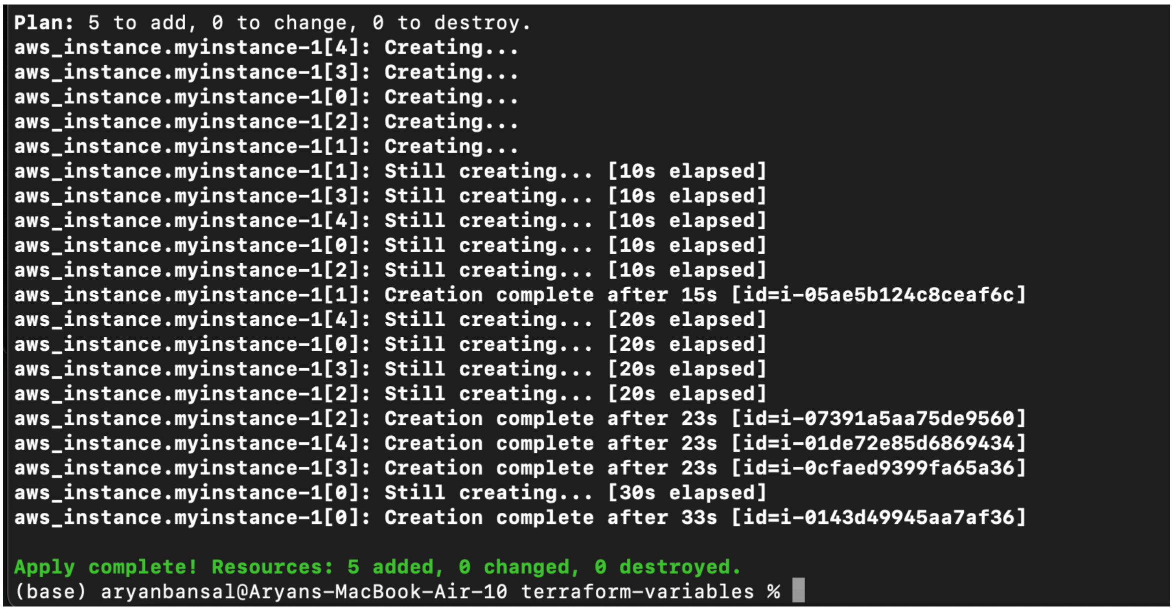
```
[(base) aryanbansal@Aryans-MacBook-Air-10 terraform-variables % terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.84.0

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
```

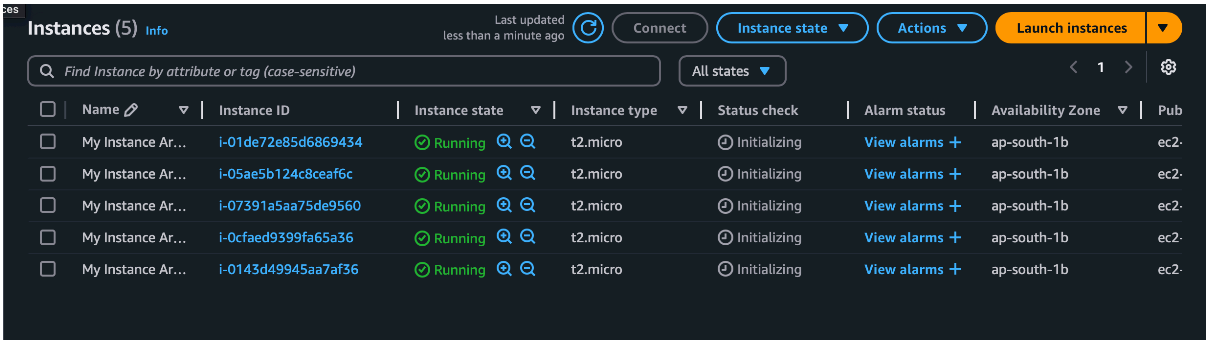


```
[(base) aryanbansal@Aryans-MacBook-Air-10 terraform-variables % terraform plan

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.myinstance-1[0] will be created
  + resource "aws_instance" "myinstance-1" {
      + ami                                  = "ami-08718895af4dfa033"
      + 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)
      + enable_primary_ipv6                  = (known after apply)
      + get_password_data                    = false
```

```
terraform apply -auto-approve
```



```
Plan: 5 to add, 0 to change, 0 to destroy.
aws_instance.myinstance-1[4]: Creating...
aws_instance.myinstance-1[3]: Creating...
aws_instance.myinstance-1[0]: Creating...
aws_instance.myinstance-1[2]: Creating...
aws_instance.myinstance-1[1]: Creating...
aws_instance.myinstance-1[1]: Still creating... [10s elapsed]
aws_instance.myinstance-1[3]: Still creating... [10s elapsed]
aws_instance.myinstance-1[4]: Still creating... [10s elapsed]
aws_instance.myinstance-1[0]: Still creating... [10s elapsed]
aws_instance.myinstance-1[2]: Still creating... [10s elapsed]
aws_instance.myinstance-1[1]: Creation complete after 15s [id=i-05ae5b124c8ceaf6c]
aws_instance.myinstance-1[4]: Still creating... [20s elapsed]
aws_instance.myinstance-1[0]: Still creating... [20s elapsed]
aws_instance.myinstance-1[3]: Still creating... [20s elapsed]
aws_instance.myinstance-1[2]: Still creating... [20s elapsed]
aws_instance.myinstance-1[2]: Creation complete after 23s [id=i-07391a5aa75de9560]
aws_instance.myinstance-1[4]: Creation complete after 23s [id=i-01de72e85d6869434]
aws_instance.myinstance-1[3]: Creation complete after 23s [id=i-0cfaed9399fa65a36]
aws_instance.myinstance-1[0]: Still creating... [30s elapsed]
aws_instance.myinstance-1[0]: Creation complete after 33s [id=i-0143d49945aa7af36]

Apply complete! Resources: 5 added, 0 changed, 0 destroyed.
(base) aryanbansal@Aryans-MacBook-Air-10 terraform-variables %
```



Instances (5) Info
Last updated less than a minute ago
Connect
Instance state ▼
Actions ▼
Launch instances ▼
Find Instance by attribute or tag (case-sensitive)
All states ▼
1

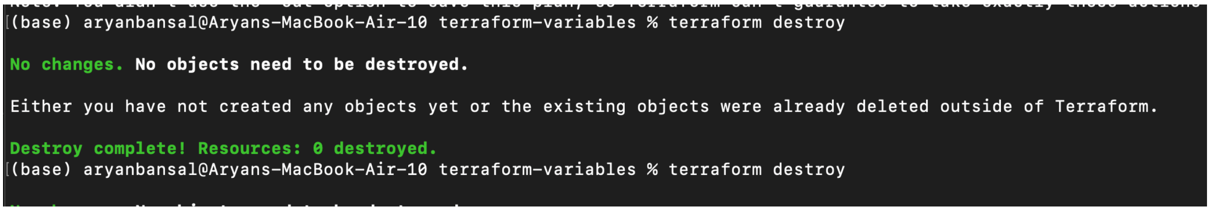
☐	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Pub
☐	My Instance Ar...	i-01de72e85d6869434	Running	t2.micro	Initializing	View alarms +	ap-south-1b	ec2-
☐	My Instance Ar...	i-05ae5b124c8ceaf6c	Running	t2.micro	Initializing	View alarms +	ap-south-1b	ec2-
☐	My Instance Ar...	i-07391a5aa75de9560	Running	t2.micro	Initializing	View alarms +	ap-south-1b	ec2-
☐	My Instance Ar...	i-0cfaed9399fa65a36	Running	t2.micro	Initializing	View alarms +	ap-south-1b	ec2-
☐	My Instance Ar...	i-0143d49945aa7af36	Running	t2.micro	Initializing	View alarms +	ap-south-1b	ec2-

Observe how the region changes based on the variable override.

5. Clean Up:

After testing, you can clean up resources.

terraform destroy



```
(base) aryanbansal@Aryans-MacBook-Air-10 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.
(base) aryanbansal@Aryans-MacBook-Air-10 terraform-variables % terraform destroy
```

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.