School of Computer Science

UNIVERSITY OF PETROLEUM AND ENERGY STUDIES
DEHRADUN, UTTARAKHAND

System Provisioning and Configuration Management

Submitted To:

Dr. Hitesh Kumar Sharma

Submitted By:

Mridul Vasudeva

Batch – 1(DevOps)

500091321

R2142210502

Lab Exercise 4– Terraform Variables

Objective:

Learn how to define and use variables in Terraform configuration.

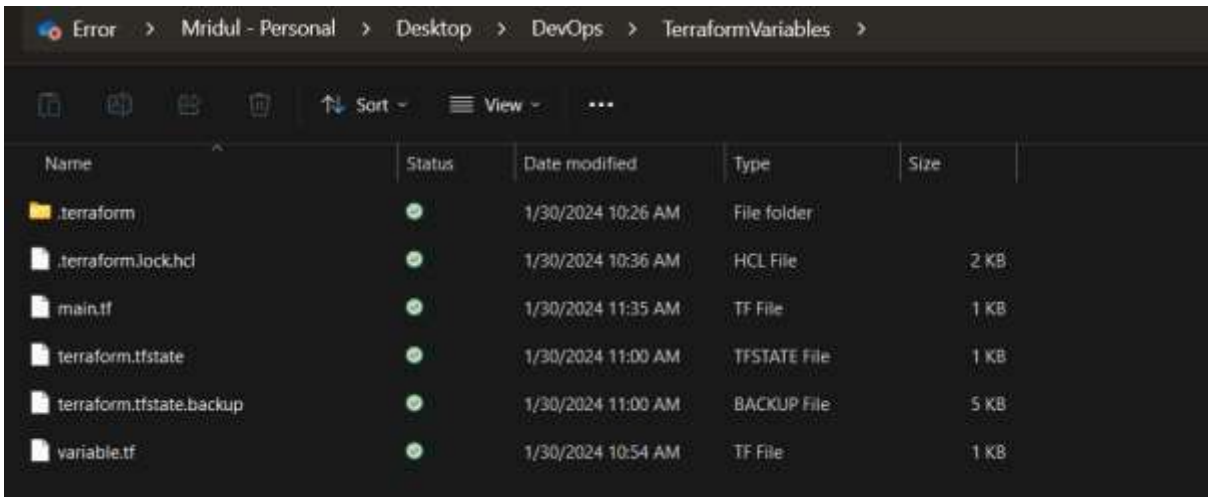
Prerequisites:

- Install Terraform on your machine.

Steps:

1. Create a Terraform Directory:

- Create a new directory for your Terraform project.



2. Create a Terraform Configuration File:

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

```
main.tf > resource "aws_instance" "example"
resource "aws_instance" "example" {
 ami = var.ami
 instance_type = var.instance_ty
}

terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.31.0"
    }
  }
}

provider "aws" {
    region = "ap-south-1"
    access_key = "AKIAZI2LIAJGSHGMMMHP"
    secret_key = "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"
}
```

3. Define Variables:

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

 # variables.tf

```
variable.tf > variable "instance_ty"
variable "ami" {
 description = "AMI ID"
 default = "ami-03f4878755434977f"
}

variable "instance_ty" {
  description = "ec2-instance"
  default = "t2.micro"
}
```

Use Variables in main.tf:

- Modify main.tf to use the variables. # **main.tf**

```
main.tf > resource "aws_instance" "example"
resource "aws_instance" "example" {
 ami = var.ami
 instance_type = var.instance_ty
}

terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.31.0"
    }
  }
}

provider "aws" {
    region = "ap-south-1"
    access_key = "AKIAZI2LIAJGSHGMMMHP"
    secret_key = "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"
}
```

4. Initialize and Apply:

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

```
PS C:\Users\Dell\OneDrive\Desktop\DevOps\TerraformVariables> 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.
```

```
PS C:\Users\Dell\OneDrive\Desktop\DevOps\TerraformVariables> terraform apply

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.example will be created
  + resource "aws_instance" "example" {
      + 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)
      + 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)
```

```
Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_instance.example: Creating...
aws_instance.example: Still creating... [10s elapsed]
aws_instance.example: Still creating... [20s elapsed]
aws_instance.example: Still creating... [30s elapsed]
aws_instance.example: Creation complete after 33s [id=i-01fafce2aefe1e3c2]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

5. Clean Up:

```
PS C:\Users\Dell\OneDrive\Desktop\DevOps\TerraformVariables> terraform destroy
aws_instance.example: Refreshing state... [id=i-01fafce2aefe1e3c2]

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

Terraform will perform the following actions:

  # aws_instance.example will be destroyed
  - resource "aws_instance" "example" {
      - ami                                  = "ami-03f4878755434977f" -> null
      - arn                                  = "arn:aws:ec2:ap-south-1:637423583821:instance/i-01fafce2aefe1e3c2" -> null
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "ap-south-1a" -> null
      - cpu_core_count                       = 1 -> null
      - cpu_threads_per_core                 = 1 -> null
      - disable_api_stop                     = false -> null
      - disable_api_termination              = false -> null
      - ebs_optimized                        = false -> null
      - get_password_data                    = false -> null
      - hibernation                          = false -> null
      - id                                   = "i-01fafce2aefe1e3c2" -> null
      - instance_initiated_shutdown_behavior = "stop" -> null
      - instance_state                       = "running" -> null
      - instance_type                        = "t2.micro" -> null
      - ipv6_address_count                   = 0 -> null
      - ipv6_addresses                       = [] -> null
      - monitoring                           = false -> null
      - placement_partition_number           = 0 -> null
      - primary_network_interface_id         = "eni-03c4c13fe075584fa" -> null
      - private_dns                          = "ip-172-31-41-191.ap-south-1.compute.internal" -> null
      - private_ip                           = "172.31.41.191" -> null
      - public_dns                           = "ec2-13-126-53-242.ap-south-1.compute.amazonaws.com" -> null
```

```
Plan: 0 to add, 0 to change, 1 to destroy.

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_instance.example: Destroying... [id=i-01fafce2aefe1e3c2]
aws_instance.example: Still destroying... [id=i-01fafce2aefe1e3c2, 10s elapsed]
aws_instance.example: Still destroying... [id=i-01fafce2aefe1e3c2, 20s elapsed]
aws_instance.example: Still destroying... [id=i-01fafce2aefe1e3c2, 30s elapsed]
aws_instance.example: Destruction complete after 34s

Destroy complete! Resources: 1 destroyed.
```


Services
Search
[Alt+S]
Mumbai
EC2 Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Instances info
Connect
Instance state
Actions
Launch instances
Find instance by attribute or tag (case-sensitive)
Any state
Instance state = running
Clear filters
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
No matching instances found