



**SYSTEM PROVISIONING AND CONFIGURATION
MANAGEMENT LAB**

**Lab File
(2023-2024)**

for

6th Semester

Submitted To

Dr. Hitesh Kumar Sharma
Cluster Head (Cybernetics)
School of Computer Science

Submitted By:

Arpit Goyal
B. Tech. CSE DevOps [6th
Semester]
500094790
R2142210148
B-3

Exercise 6– Terraform Multiple tfvars Files

Objective:

Learn how to use multiple tfvars files in Terraform for different environments.


Prerequisites:

- Terraform installed on your machine.
- Basic knowledge of Terraform configuration and variables.

Steps:

1. Create a Terraform Directory:

- Create Terraform Configuration Files:
- Create a file named main.tf and Create a file named variables.tf:



```
main.tf
1 terraform {
2   required_providers {
3     aws = {
4       source = "hashicorp/aws"
5       version = "5.31.0"
6     }
7   }
8 }
9
10 provider "aws" {
11   region = var.region
12   access_key = var.access_key
13   secret_key = var.secret_key
14 }
15

variables.tf
1 variable region {
2   type = string
3   default = "ap-south-1"
4   description = "AWS Region"
5 }
6
7 variable "ami" {
8   type = string
9   default = "ami-03f4878755434977f"
10  description = "AMI ID"
11 }
12
13 variable "instance_type" {
14   type = string
15   default = "t2.micro"
16   description = "Instance Type"
17 }
18
```

2. Create Multiple tfvars Files:

- Create a file named dev.tfvars:

```
dev.tfvars
1 region= "ap-south-1"
2 ami= "ami-03f4878755434977f"
3 instance_type = "t2.micro"
```

Create a file named prod.tfvars:

```
prod.tfvars
1 region= "us-east-1"
2 ami= "ami-0c7217cdde317cfec"
3 instance_type = "t2.micro"
```

3. Initialize and Apply for Dev Environment:

```
Exp2 terraform apply -var-file=dev.tfvars

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.Ayroid-ec2[0] will be created
+ resource "aws_instance" "Ayroid-ec2" {
```

4. Initialize and Apply for Prod Environment:

```
Exp2 terraform apply -var-file=prod.tfvars
aws_instance.Ayroid-ec2[0]: Refreshing state... [id=i-0c845335fb832ccf9]

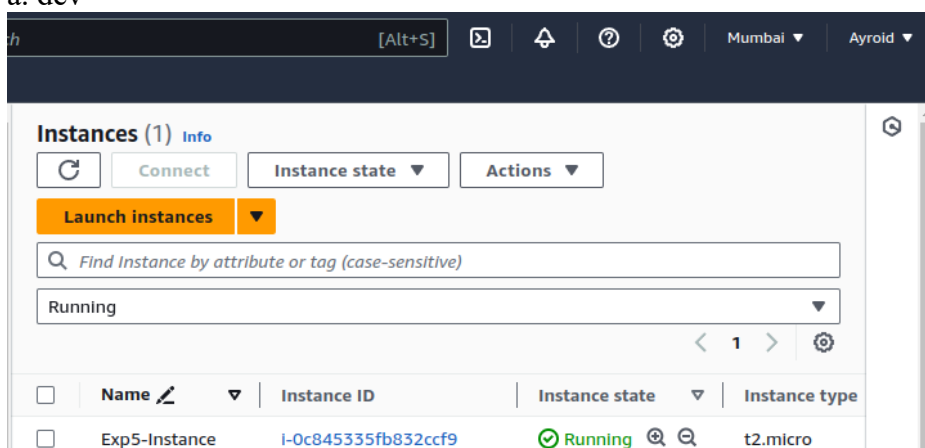
Terraform used the selected providers to generate the following execution plan. Resource actions
+ create

Terraform will perform the following actions:

# aws_instance.Ayroid-ec2[0] will be created
+ resource "aws_instance" "Ayroid-ec2" {
```

5. Test and Verify:

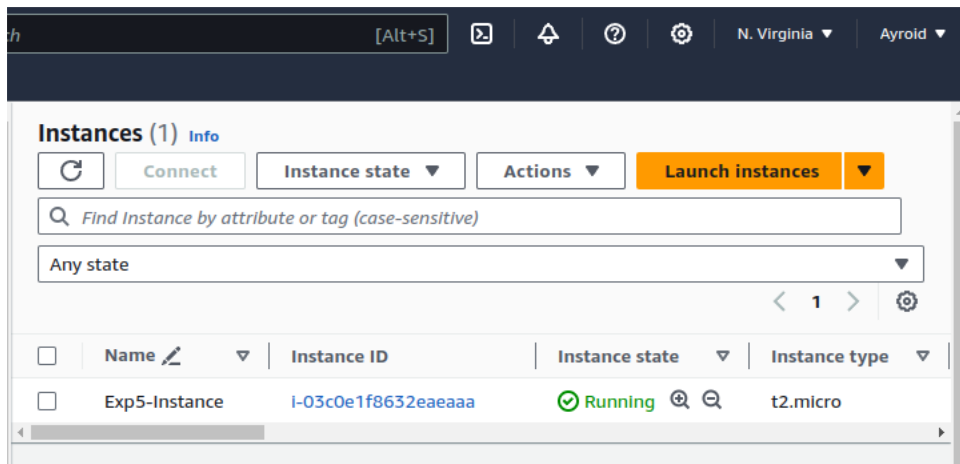
a. dev



The screenshot shows the AWS Management Console interface for the Mumbai region and the Ayroid account. The 'Instances (1)' section is active, displaying a table with one instance. The instance is named 'Exp5-Instance', has an ID of 'i-0c845335fb832ccf9', and is in a 'Running' state. The instance type is 't2.micro'. The console also shows buttons for 'Launch instances', 'Connect', 'Instance state', and 'Actions'.

	Name	Instance ID	Instance state	Instance type
<input type="checkbox"/>	Exp5-Instance	i-0c845335fb832ccf9	Running	t2.micro

b. prod



6. Clean up resources in both environments

a. dev

```
→ Exp2 terraform destroy -var-file=dev.tfvars
aws_instance.Ayroid-ec2[0]: Refreshing state... [id=i-0ea2577c5a7fa799a]

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.Ayroid-ec2[0] will be destroyed
```

b. prod

```
→ Exp2 terraform destroy -var-file=prod.tfvars
aws_instance.Ayroid-ec2[0]: Refreshing state... [id=i-03c0e1f8632eaeaaa]

Terraform used the selected providers to generate the following execution plan. Resource
- destroy

Terraform will perform the following actions:

# aws_instance.Ayroid-ec2[0] will be destroyed
- resource "aws_instance" "Ayroid-ec2" {
```