



**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 4– Terraform Variables

Objective:

Learn how to define and use variables in Terraform configuration.

Prerequisites:

- Install Terraform on your machine.

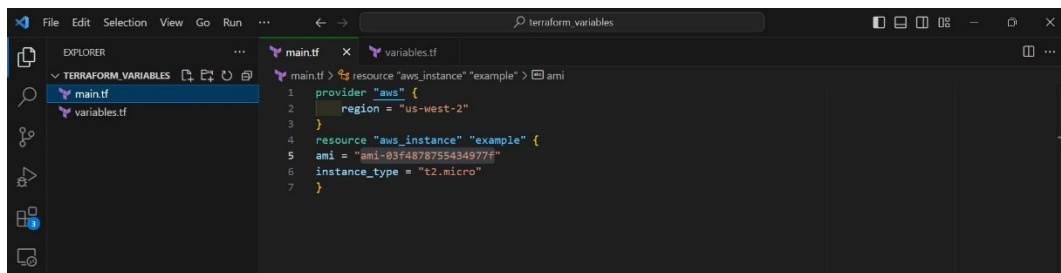
Steps:

1. Create a Terraform Directory:

- Create a new directory for your Terraform project.

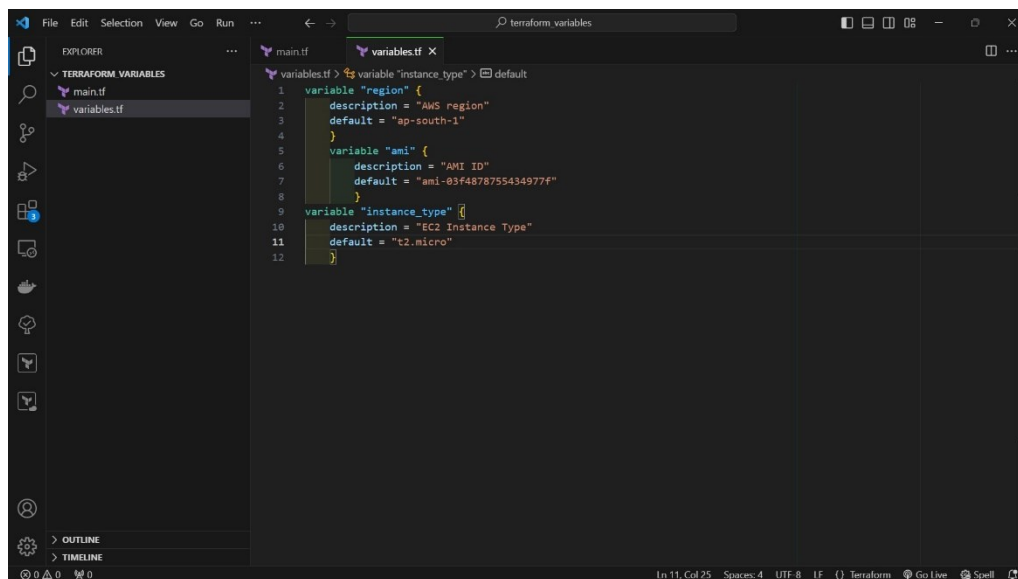
2. Create a Terraform Configuration File:

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



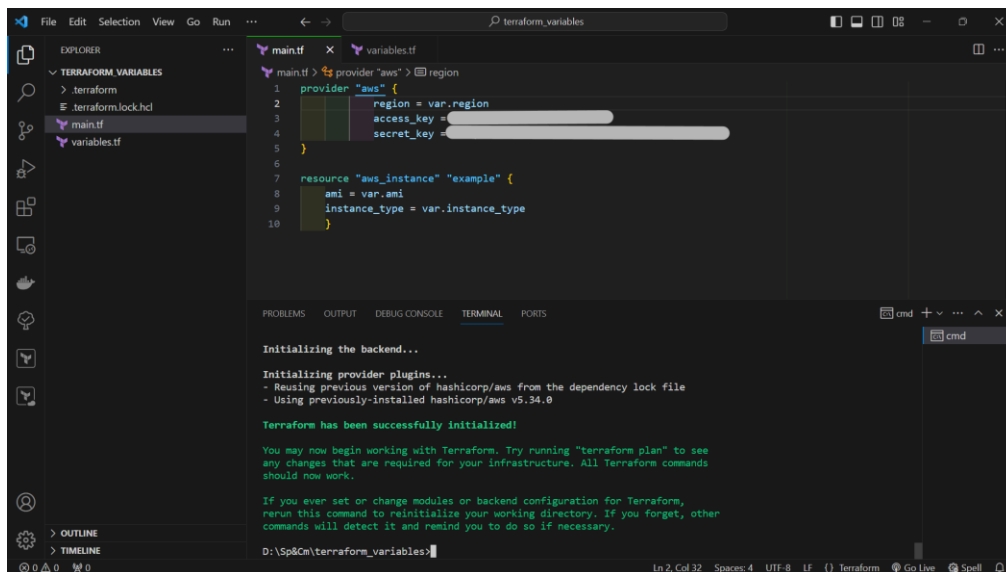
3. Define Variables:

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



4. Use Variables in main.tf:

- Modify main.tf to use the variables.



The screenshot shows the Visual Studio Code editor with two files open: `main.tf` and `variables.tf`. The `main.tf` file contains the following Terraform configuration:

```
1 provider "aws" {
2   region = var.region
3   access_key = [REDACTED]
4   secret_key = [REDACTED]
5 }
6
7 resource "aws_instance" "example" {
8   ami = var.ami
9   instance_type = var.instance_type
10 }
```

The `variables.tf` file is empty. The terminal at the bottom shows the output of the `terraform init` command:

```
Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.34.0

Terraform has been successfully initialized!

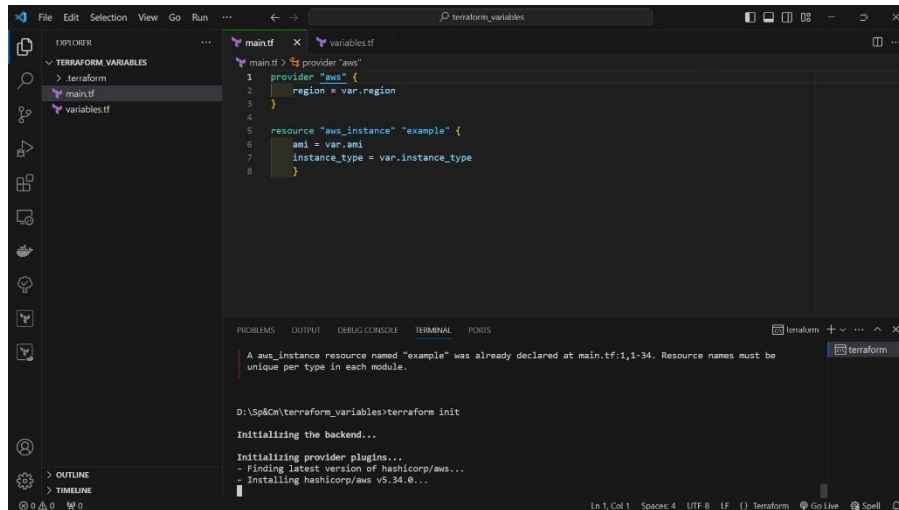
You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

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.

D:\Sp8Cn\terraform_variables>
```

5. Initialize and Apply:

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



The screenshot shows the Visual Studio Code editor with two files open: `main.tf` and `variables.tf`. The `main.tf` file contains the following Terraform configuration:

```
1 provider "aws" {
2   region = var.region
3 }
4
5 resource "aws_instance" "example" {
6   ami = var.ami
7   instance_type = var.instance_type
8 }
```

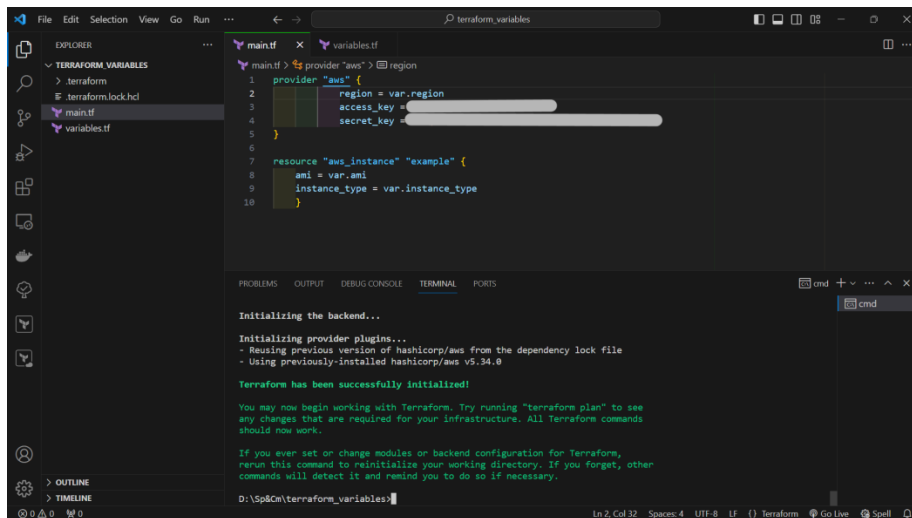
The `variables.tf` file is empty. The terminal at the bottom shows the output of the `terraform init` command:

```
A aws_instance resource named "example" was already declared at main.tf:1,1-34. Resource names must be
unique per type in each module.

D:\Sp8Cn\terraform_variables>terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.34.0...
```



```
main.tf x variables.tf
main.tf > provider "aws" { region
1 provider "aws" {
2   region = var.region
3   access_key =
4   secret_key =
5 }
6
7 resource "aws_instance" "example" {
8   ami = var.ami
9   instance_type = var.instance_type
10 }
```

Initializing the backend...

Initializing provider plugins...

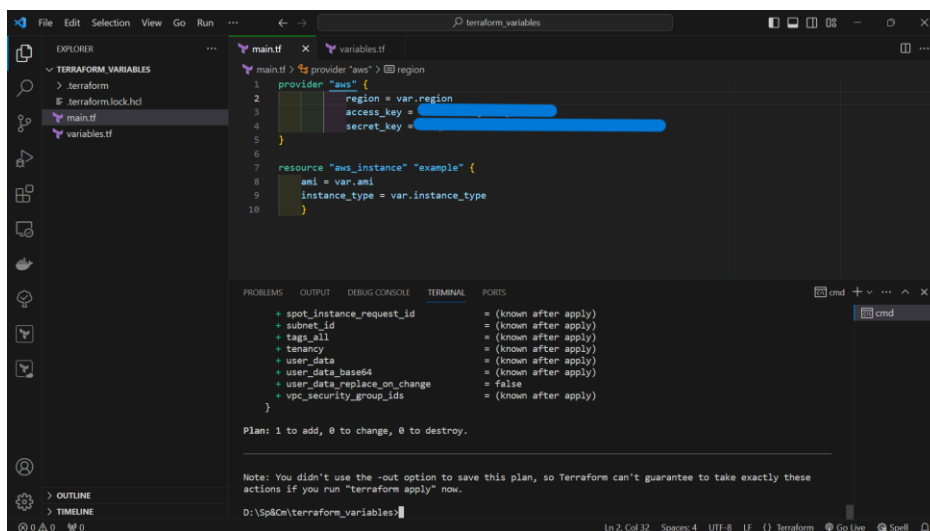
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.34.0

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.

D:\Sp8Cn\terraform_variables



```
main.tf x variables.tf
main.tf > provider "aws" { region
1 provider "aws" {
2   region = var.region
3   access_key =
4   secret_key =
5 }
6
7 resource "aws_instance" "example" {
8   ami = var.ami
9   instance_type = var.instance_type
10 }
```

spot_instance_request_id = (known after apply)

subnet_id = (known after apply)

tag_all = (known after apply)

tenancy = (known after apply)

user_data = (known after apply)

user_data_base64 = (known after apply)

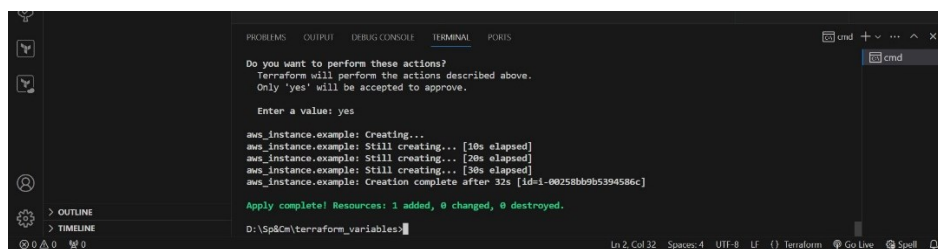
user_data_replace_on_change = false

vpc_security_group_ids = (known after apply)

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

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.

D:\Sp8Cn\terraform_variables



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]

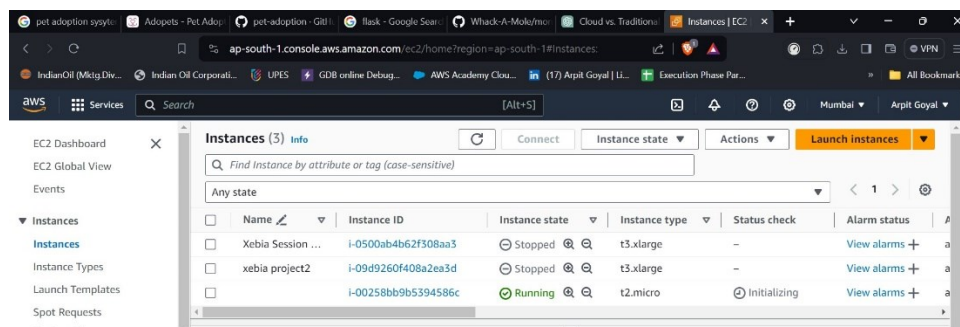
aws_instance.example: Still creating... [30s elapsed]

aws_instance.example: Creation complete after 32s [id=1-00258bb9b5394586c]

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

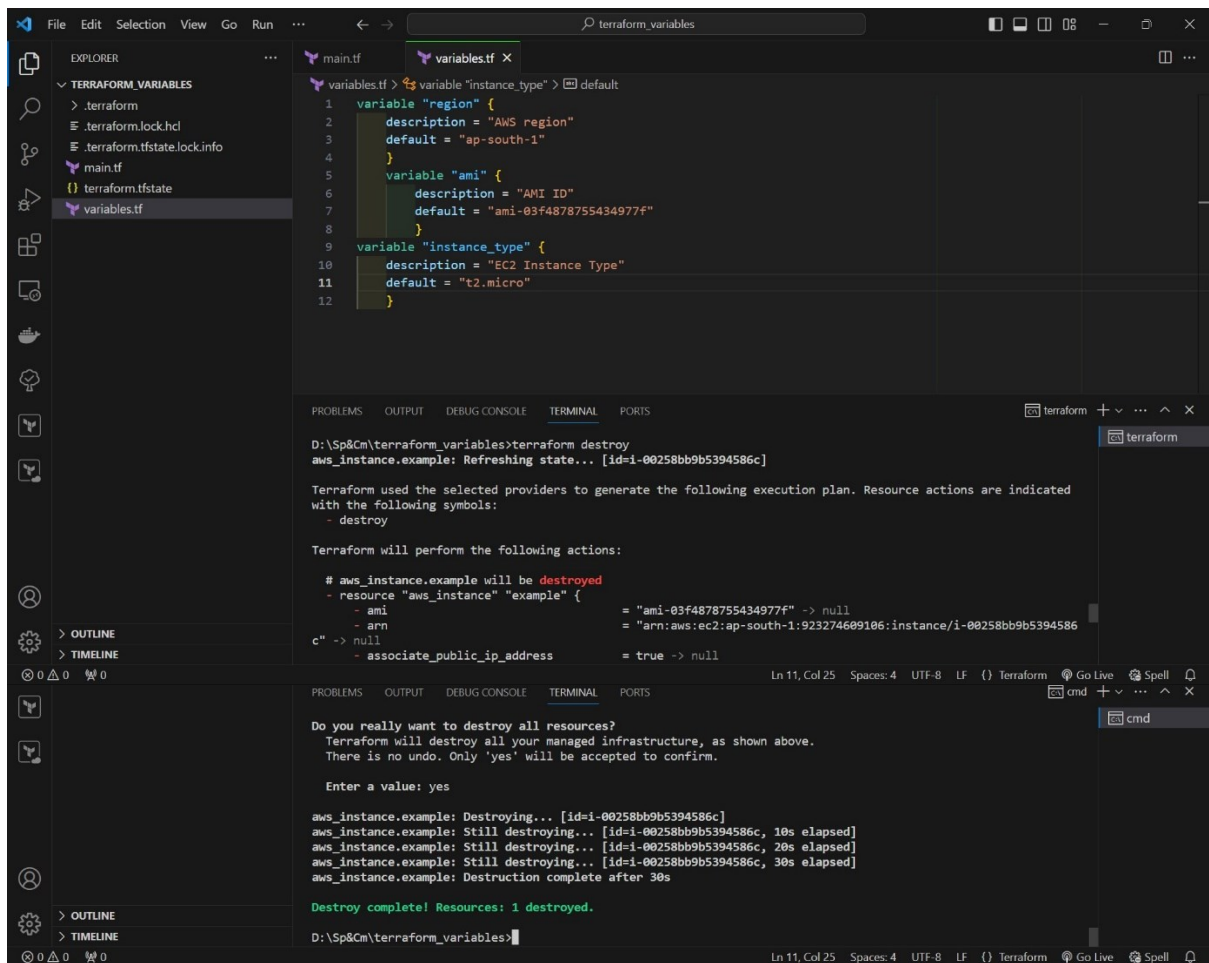
D:\Sp8Cn\terraform_variables

Verify Resource :



	Name	Instance ID	Instance state	Instance type	Status check	Alarm status
<input type="checkbox"/>	Xebia Session ...	i-0500ab4b62f308aa3	Stopped	t3.xlarge	-	View alarms
<input type="checkbox"/>	xebia project2	i-09d9260f408a2ea3d	Stopped	t3.xlarge	-	View alarms
<input type="checkbox"/>		i-00258bb9b5394586c	Running	t2.micro	Initializing	View alarms

6. Clean Up: After testing, you can clean up resources. Confirm the destruction by typing yes.



The screenshot shows the Visual Studio Code interface with a Terraform configuration file named `variables.tf` open. The configuration defines variables for `region`, `ami`, and `instance_type`. The `main.tf` file is also visible in the Explorer. The terminal window shows the command `terraform destroy` being executed. The output indicates that Terraform will destroy the `aws_instance.example` resource. The user is prompted to confirm the destruction by typing 'yes'. The terminal output shows the destruction process completing successfully.

```
variables.tf > variable "instance_type" > default
1 variable "region" {
2   description = "AWS region"
3   default = "ap-south-1"
4 }
5 variable "ami" {
6   description = "AMI ID"
7   default = "ami-03f4878755434977f"
8 }
9 variable "instance_type" {
10  description = "EC2 Instance Type"
11  default = "t2.micro"
12 }
```

```
D:\Sp8Cm\terraform_variables>terraform destroy
aws_instance.example: Refreshing state... [id=i-00258bb9b5394586c]

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-1:923274609106:instance/i-00258bb9b5394586c" -> null
  c             -> null
  associate_public_ip_address = true -> null

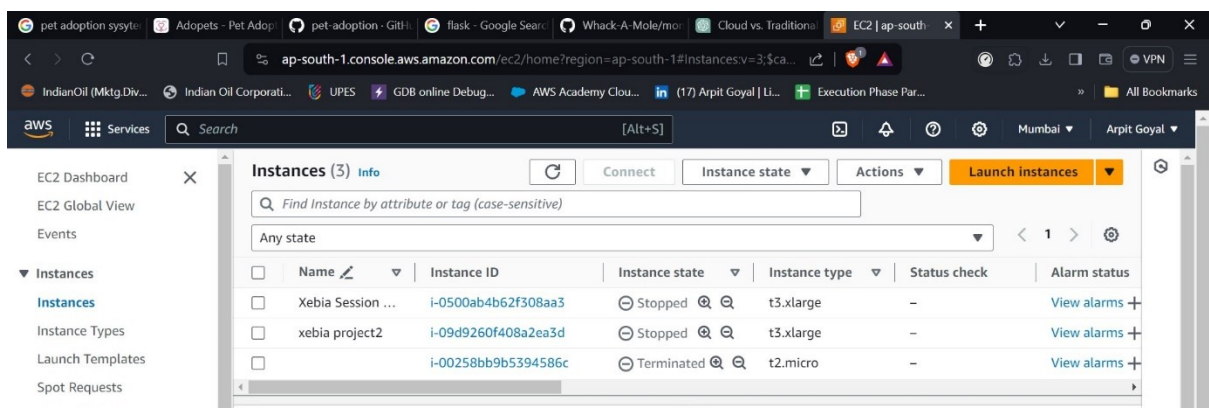
Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

aws_instance.example: Destroying... [id=i-00258bb9b5394586c]
aws_instance.example: Still destroying... [id=i-00258bb9b5394586c, 10s elapsed]
aws_instance.example: Still destroying... [id=i-00258bb9b5394586c, 20s elapsed]
aws_instance.example: Still destroying... [id=i-00258bb9b5394586c, 30s elapsed]
aws_instance.example: Destruction complete after 30s

Destroy complete! Resources: 1 destroyed.

D:\Sp8Cm\terraform_variables>
```



The screenshot shows the AWS Management Console for the `ap-south-1` region. The `Instances (3)` page is displayed, showing a list of EC2 instances. The instances are `Xebia Session ...`, `xebia project2`, and `i-00258bb9b5394586c`. The instance `i-00258bb9b5394586c` is in the `Terminated` state, which is highlighted in red. The other two instances are in the `Stopped` state.

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status
<input type="checkbox"/>	Xebia Session ...	i-0500ab4b62f308aa3	Stopped	t3.xlarge	-	View alarms
<input type="checkbox"/>	xebia project2	i-09d9260f408a2ea3d	Stopped	t3.xlarge	-	View alarms
<input type="checkbox"/>		i-00258bb9b5394586c	Terminated	t2.micro	-	View alarms