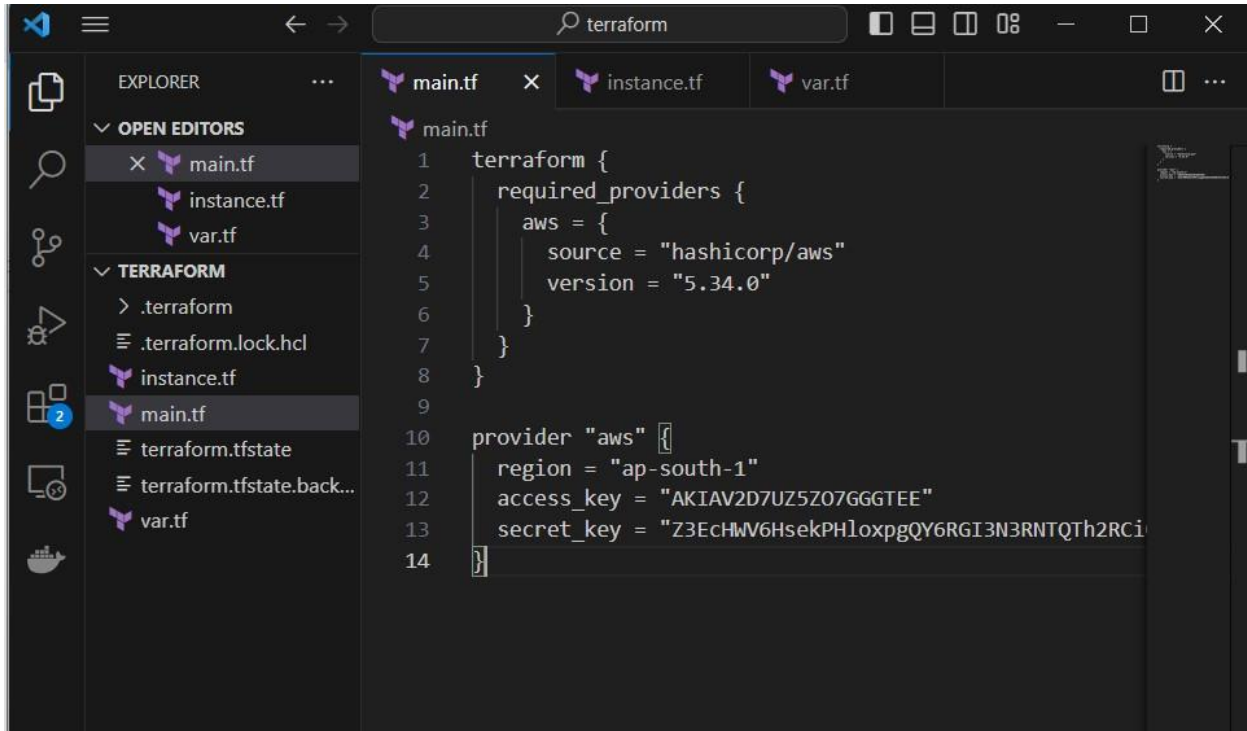


LAB-5

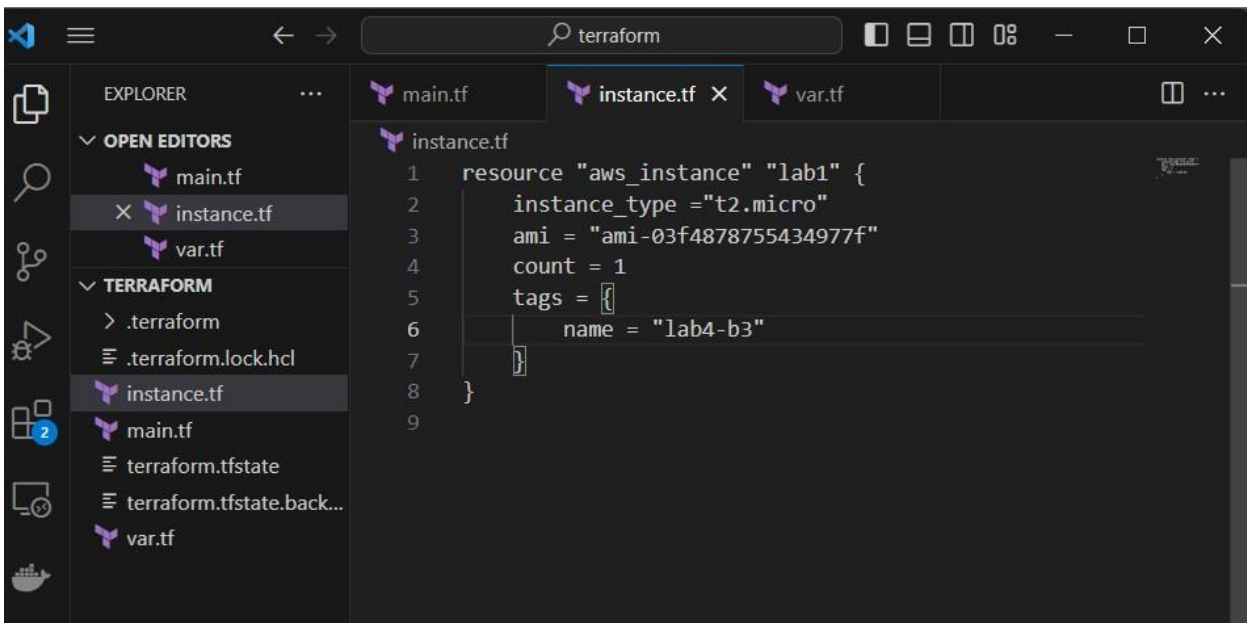
Terraform Variable with Command Line Argument

Step 1: Make changes in var.tf file



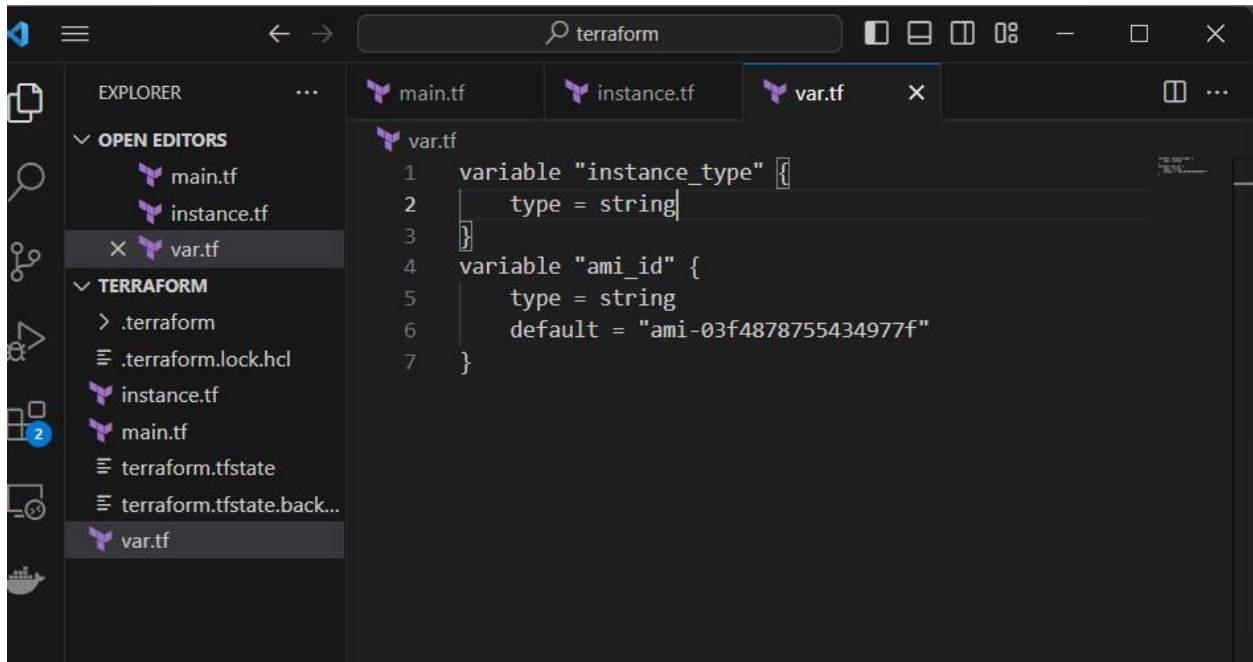
The screenshot shows the Visual Studio Code interface with a Terraform project. The Explorer panel on the left shows the file structure with 'main.tf' selected. The main editor displays the content of 'main.tf'.

```
1 terraform {
2   required_providers {
3     aws = {
4       source = "hashicorp/aws"
5       version = "5.34.0"
6     }
7   }
8 }
9
10 provider "aws" {
11   region = "ap-south-1"
12   access_key = "AKIAV2D7UZ5Z07GGGTEE"
13   secret_key = "Z3EchWV6HsekPH1oxpgQY6RGI3N3RNTQTh2RCi"
14 }
```



The screenshot shows the Visual Studio Code interface with a Terraform project. The Explorer panel on the left shows the file structure with 'instance.tf' selected. The main editor displays the content of 'instance.tf'.

```
1 resource "aws_instance" "lab1" {
2   instance_type = "t2.micro"
3   ami = "ami-03f4878755434977f"
4   count = 1
5   tags = {
6     name = "lab4-b3"
7   }
8 }
9
```



Step 2: Now we need to run terraform cycle

A screenshot of a Windows Command Prompt window. The title bar reads 'Command Prompt'. The command 'C:\Users\hp\terraform>terraform init' has been entered. The output shows the initialization process, including reusing previous versions of providers from a lock file. The message 'Terraform has been successfully initialized!' is displayed in green. Instructions for running 'terraform plan' and reinitializing the directory are also shown in green.

```
C:\Users\hp\terraform>terraform init  
  
Initializing the backend...  
  
Initializing provider plugins...  
- Reusing previous version of hashicorp/aws from the dependency lock file  
- Using previously-installed hashicorp/aws v5.34.0  
  
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.  
  
C:\Users\hp\terraform>
```

Now we have to ways to declare variable in CLI

First: We can give value after running terraform plan

```
Command Prompt

C:\Users\hp\terraform>terraform plan
var.instance_type
  Enter a value: t2.micro

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.lab1[0] will be created
+ resource "aws_instance" "lab1" {
  + 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)
  + instance_state         = (known after apply)
  + instance_type          = "t2.micro"
  + ipv6_address_count     = (known after apply)
  + ipv6_addresses        = (known after apply)
  + key_name               = (known after apply)
  + monitoring             = (known after apply)
  + outpost_arn            = (known after apply)
  + password_data          = (known after apply)
  + placement_group        = (known after apply)
```

Second: By declaring variable during running command

```
Command Prompt - terraform x + v
C:\Users\hp\terraform>terraform apply
var.instance_type
  Enter a value: t2.micro

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.lab1[0] will be created
+ resource "aws_instance" "lab1" {
  + 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)
  + instance_state        = (known after apply)
  + instance_type         = "t2.micro"
  + ipv6_address_count     = (known after apply)
  + ipv6_addresses        = (known after apply)
  + key_name               = (known after apply)
  + monitoring             = (known after apply)
  + outpost_arn            = (known after apply)
  + password_data          = (known after apply)
  + placement_group        = (known after apply)
```

```
Command Prompt

+ private_dns           = (known after apply)
+ private_ip            = (known after apply)
+ public_dns            = (known after apply)
+ public_ip             = (known after apply)
+ secondary_private_ips = (known after apply)
+ security_groups       = (known after apply)
+ source_dest_check     = true
+ spot_instance_request_id = (known after apply)
+ subnet_id             = (known after apply)
+ tags                  = {
  + "name" = "lab4-b3"
}
+ tags_all              = {
  + "name" = "lab4-b3"
}
+ 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.

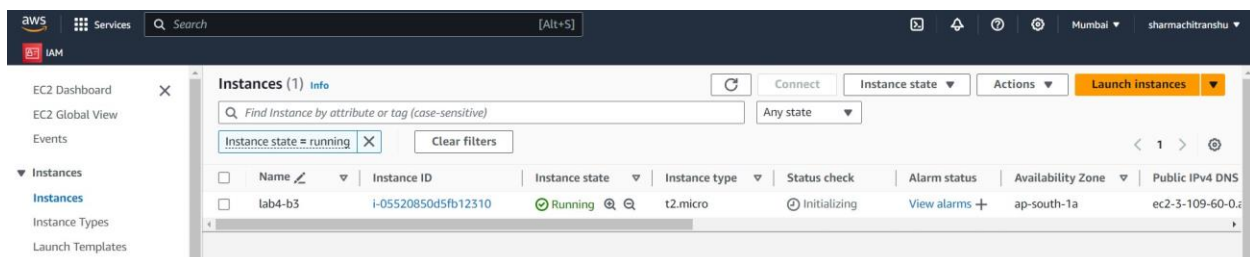
Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

Enter a value: yes

aws_instance.lab1[0]: Creating...
aws_instance.lab1[0]: Still creating... [10s elapsed]
aws_instance.lab1[0]: Still creating... [20s elapsed]
aws_instance.lab1[0]: Still creating... [30s elapsed]
aws_instance.lab1[0]: Creation complete after 33s [id=i-05520850d5fb12310]

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

C:\Users\hp\terraform>
```




```

Command Prompt
C:\Users\hp\terraform>terraform destroy
var.instance_type
  Enter a value: t2.micro

aws_instance.lab1[0]: Refreshing state... [id=i-05520850d5fb12310]

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.lab1[0] will be destroyed
  - resource "aws_instance" "lab1" {
    - ami                                = "ami-03f4878755434977f" -> null
  }
  - arn                                = "arn:aws:ec2:ap-south-1:39969
9660658:instance/i-05520850d5fb12310" -> 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-05520850d5fb12310" -> 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-0660e5364bfd78d52" -> null
  - private_dns                        = "ip-172-31-46-149.ap-south-1.
compute.internal" -> null

```

```
Command Prompt

- enable_resource_name_dns_aaaa_record = false -> null
- hostname_type                         = "ip-name" -> null
}

- root_block_device {
  - delete_on_termination = true -> null
  - device_name           = "/dev/sda1" -> null
  - encrypted             = false -> null
  - iops                  = 100 -> null
  - tags                  = {} -> null
  - throughput            = 0 -> null
  - volume_id             = "vol-0529c4704b8bb77d8" -> null
  - volume_size           = 8 -> null
  - volume_type           = "gp2" -> 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.lab1[0]: Destroying... [id=i-05520850d5fb12310]
aws_instance.lab1[0]: Still destroying... [id=i-05520850d5fb12310, 10s elapsed]
aws_instance.lab1[0]: Still destroying... [id=i-05520850d5fb12310, 20s elapsed]
aws_instance.lab1[0]: Still destroying... [id=i-05520850d5fb12310, 30s elapsed]
aws_instance.lab1[0]: Still destroying... [id=i-05520850d5fb12310, 40s elapsed]
aws_instance.lab1[0]: Destruction complete after 41s

Destroy complete! Resources: 1 destroyed.

C:\Users\hp\terraform>
```

The screenshot shows the AWS Management Console interface. The left sidebar contains navigation links for IAM, EC2 Dashboard, EC2 Global View, and Events. The main content area is titled 'Instances (2)' and includes a search bar and a table of instances. The table has columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, and Public IPv4 DNS. One instance, 'lab4-b3' with ID 'i-05520850d5fb12310', is listed with a state of 'Terminated'.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
lab4-b3	i-05520850d5fb12310	Terminated	t2.micro	-	View alarms	ap-south-1a	-