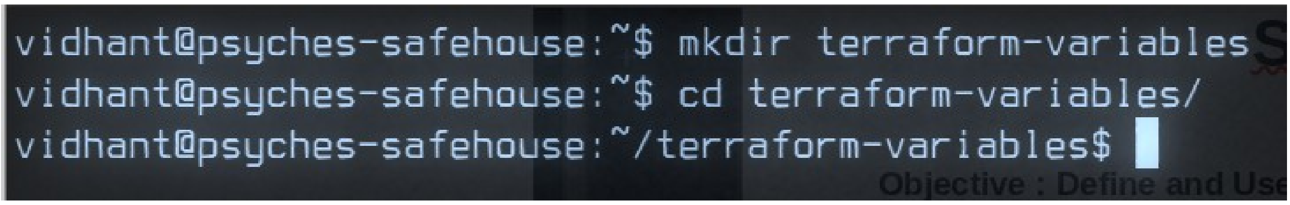
SPCM Lab-6

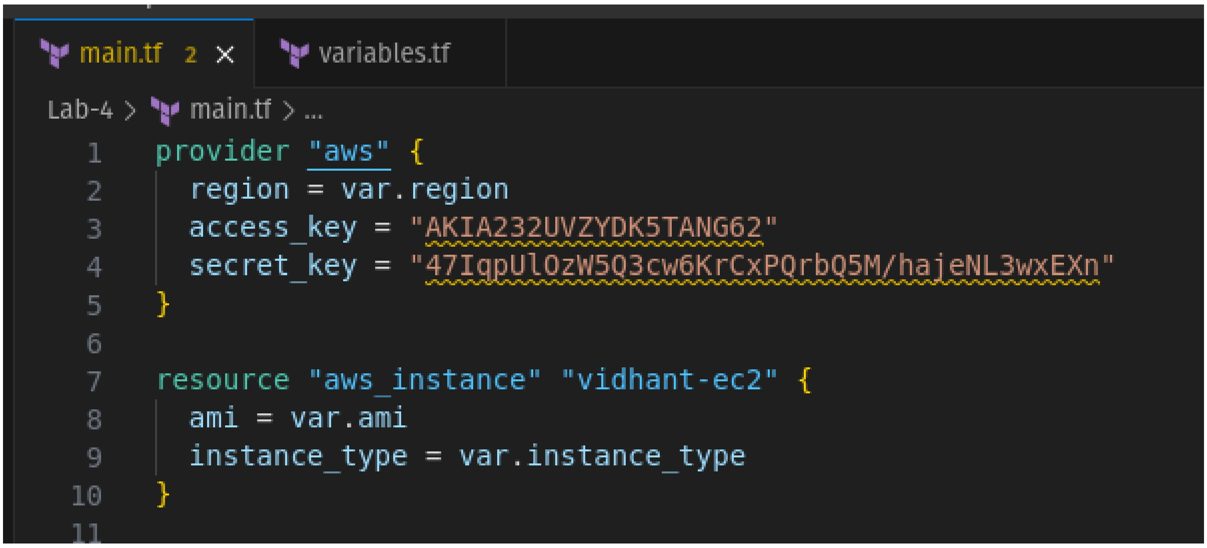
Objective : understanding how to use multiple tfvars files in terraform for different evironments.

1. Create Terraform directory.



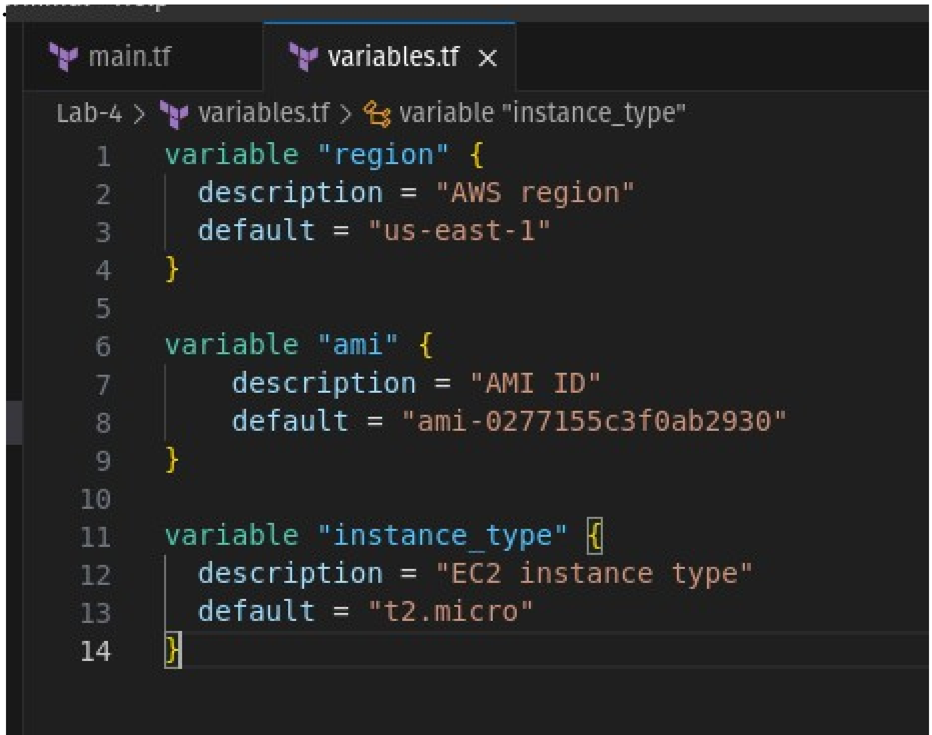
```
vidhant@psyches-safehouse:~$ mkdir terraform-variables
vidhant@psyches-safehouse:~$ cd terraform-variables/
vidhant@psyches-safehouse:~/terraform-variables$
```

2. Create terraform configuration file (main.tf) :



```
provider "aws" {
  region = var.region
  access_key = "AKIA232UVZYDK5TANG62"
  secret_key = "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"
}

resource "aws_instance" "vidhant-ec2" {
  ami = var.ami
  instance_type = var.instance_type
}
```

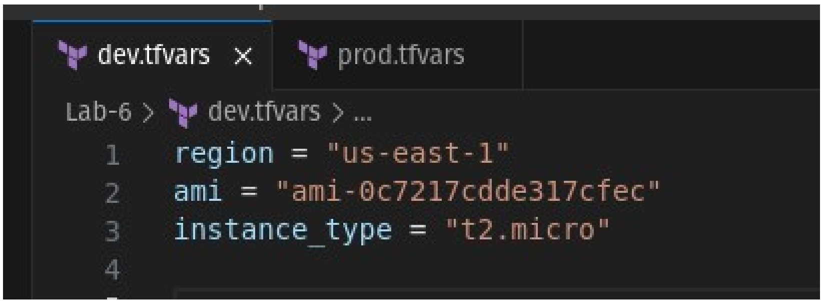
3. Define Variables : create a new file variables.tf and define variables for region, ami and instance_type.



```
variable "region" {
  description = "AWS region"
  default = "us-east-1"
}

variable "ami" {
    description = "AMI ID"
    default = "ami-0277155c3f0ab2930"
}

variable "instance_type" {
  description = "EC2 instance type"
  default = "t2.micro"
}
```

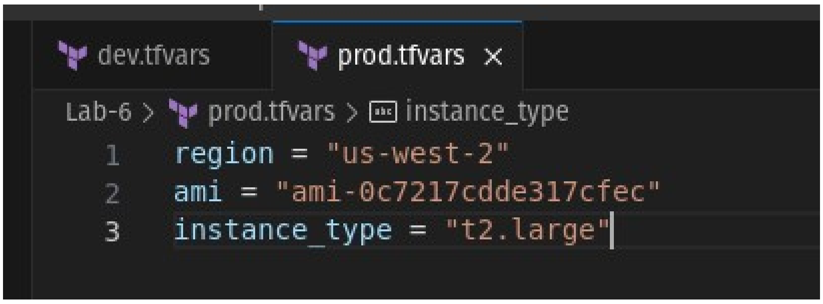
4. Create multiple tfvars Files :

(a) Create a file named dev.tfvars (configuration for development environment)



```
region = "us-east-1"
ami = "ami-0c7217cdde317cfec"
instance_type = "t2.micro"

```

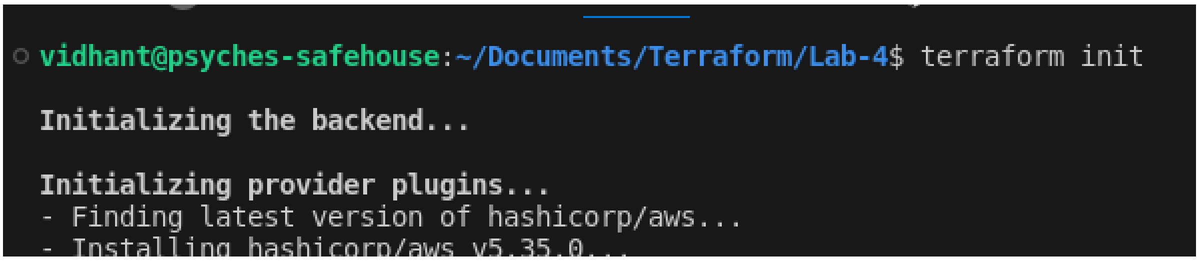
(b) Create a file name dev.tfvars (configuration for production environment)



```
region = "us-west-2"
ami = "ami-0c7217cdde317cfec"
instance_type = "t2.large"
```

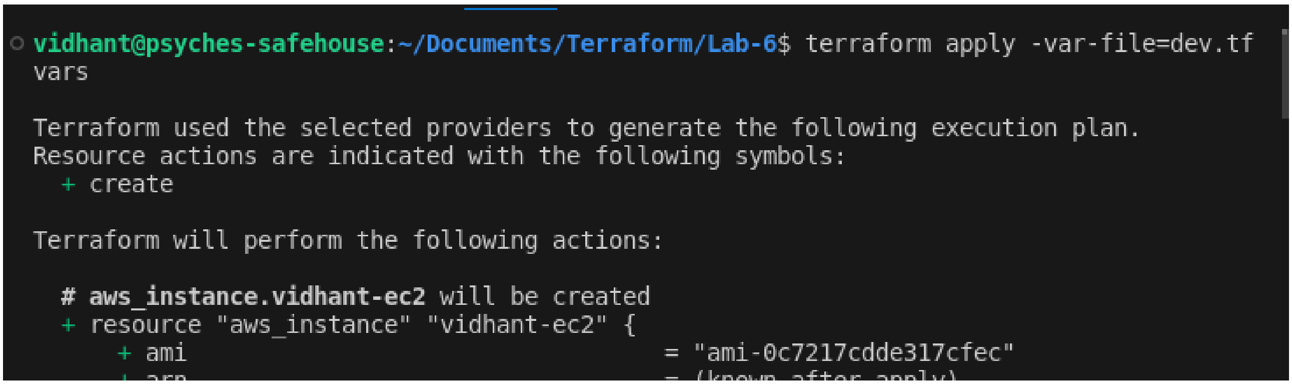
5. Initialize and Apply for Development Environment.

terraform init :



```
○ vidhant@psyches-safehouse:~/Documents/Terraform/Lab-4$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.35.0...
```

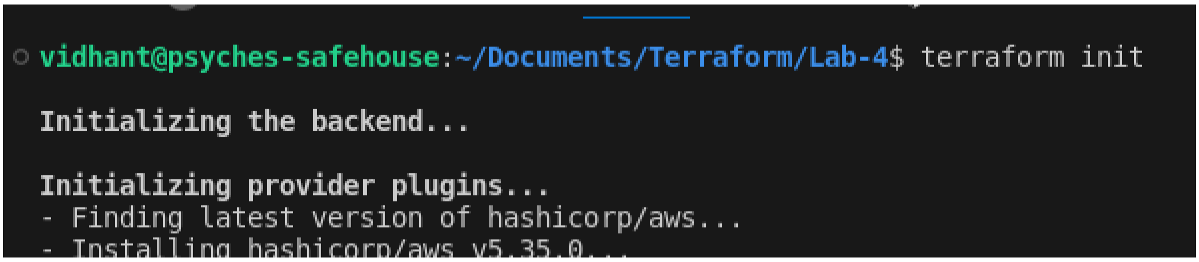
terraform apply -var-file=dev.tfvars



```
○ vidhant@psyches-safehouse:~/Documents/Terraform/Lab-6$ terraform apply -var-file=dev.tf
  vars

  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.vidhant-ec2 will be created
    + resource "aws_instance" "vidhant-ec2" {
        + ami                                  = "ami-0c7217cdde317cfec"
        + arn                                  = (known after apply)
```

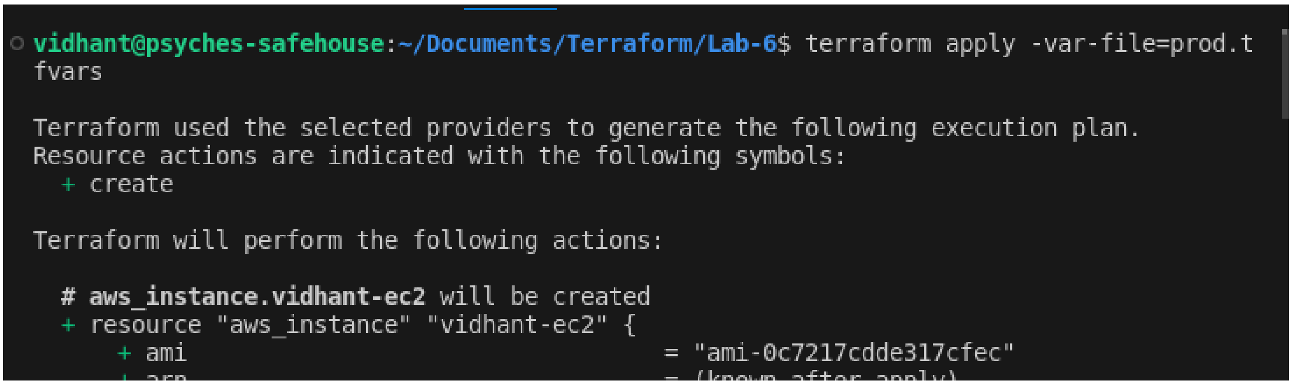
6. Initialize and Apply for Prod Environment.

terraform init :



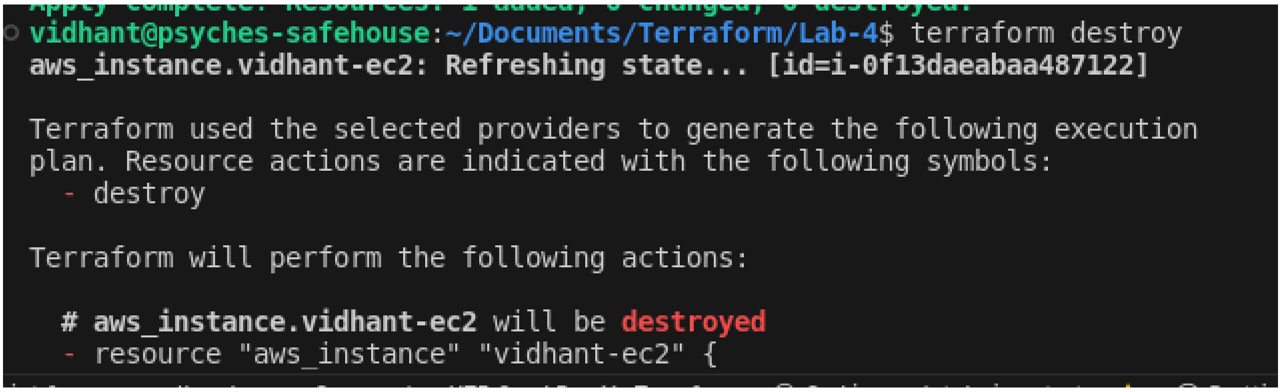
```
○ vidhant@psyches-safehouse:~/Documents/Terraform/Lab-4$ terraform init

  Initializing the backend...

  Initializing provider plugins...
  - Finding latest version of hashicorp/aws...
  - Installing hashicorp/aws v5.35.0...
```

terraform apply -var-file=prod.tfvars



```
○ vidhant@psyches-safehouse:~/Documents/Terraform/Lab-6$ terraform apply -var-file=prod.t
  fvars

  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.vidhant-ec2 will be created
    + resource "aws_instance" "vidhant-ec2" {
        + ami                                  = "ami-0c7217cdde317cfec"
        + arn                                  = (known after apply)
```

7. Cleaning up resources (terraform destroy)



```
  Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
○ vidhant@psyches-safehouse:~/Documents/Terraform/Lab-4$ terraform destroy
  aws_instance.vidhant-ec2: Refreshing state... [id=i-0f13daeabaa487122]

  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.vidhant-ec2 will be destroyed
    - resource "aws_instance" "vidhant-ec2" {
```