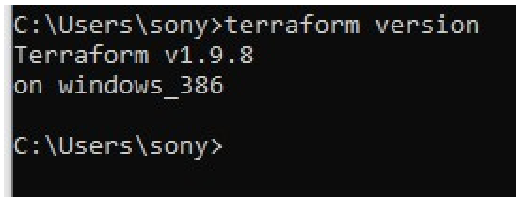
Download and do the path set for Terraform. After path set, check Terraform Version.

Command is: terraform version

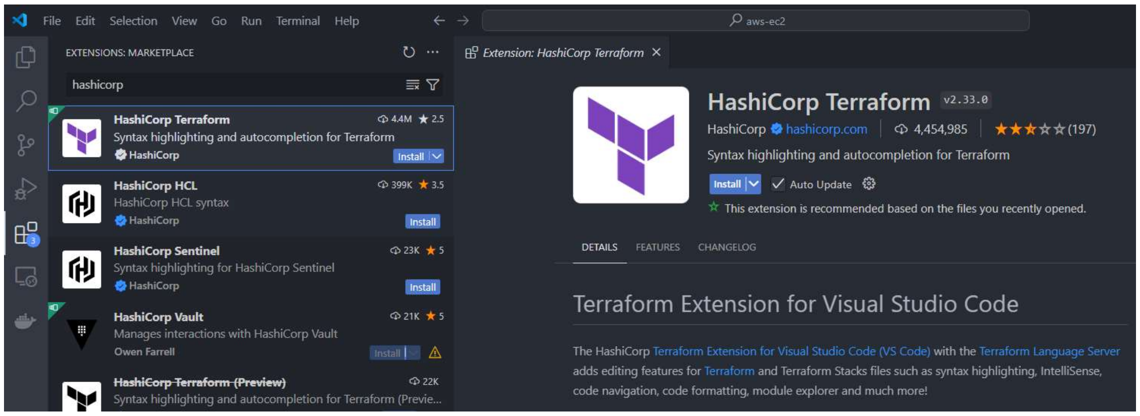


```
C:\Users\sony>terraform version
Terraform v1.9.8
on windows_386

C:\Users\sony>
```

Launch AWS EC2 instance with Terraform.

1. Install Terraform extension on VS Code.


File Edit Selection View Go Run Terminal Help
aws-ec2
EXTENSIONS: MARKETPLACE
hashicorp
HashiCorp Terraform 4.4M 2.5
Syntax highlighting and autocompletion for Terraform
HashiCorp Install
HashiCorp HCL 399K 3.5
HashiCorp HCL syntax
HashiCorp Install
HashiCorp Sentinel 23K 5
Syntax highlighting for HashiCorp Sentinel
HashiCorp Install
HashiCorp Vault 21K 5
Manages interactions with HashiCorp Vault
Owen Farrell Install
~~HashiCorp Terraform (Preview)~~ 22K
Syntax highlighting and autocompletion for Terraform (Previe...
Extension: HashiCorp Terraform
HashiCorp Terraform v2.33.0
HashiCorp hashicorp.com 4,454,985 (197)
Syntax highlighting and autocompletion for Terraform
Install Auto Update
This extension is recommended based on the files you recently opened.
DETAILS FEATURES CHANGELOG

Terraform Extension for Visual Studio Code

The HashiCorp Terraform Extension for Visual Studio Code (VS Code) with the Terraform Language Server adds editing features for Terraform and Terraform Stacks files such as syntax highlighting, IntelliSense, code navigation, code formatting, module explorer and much more!

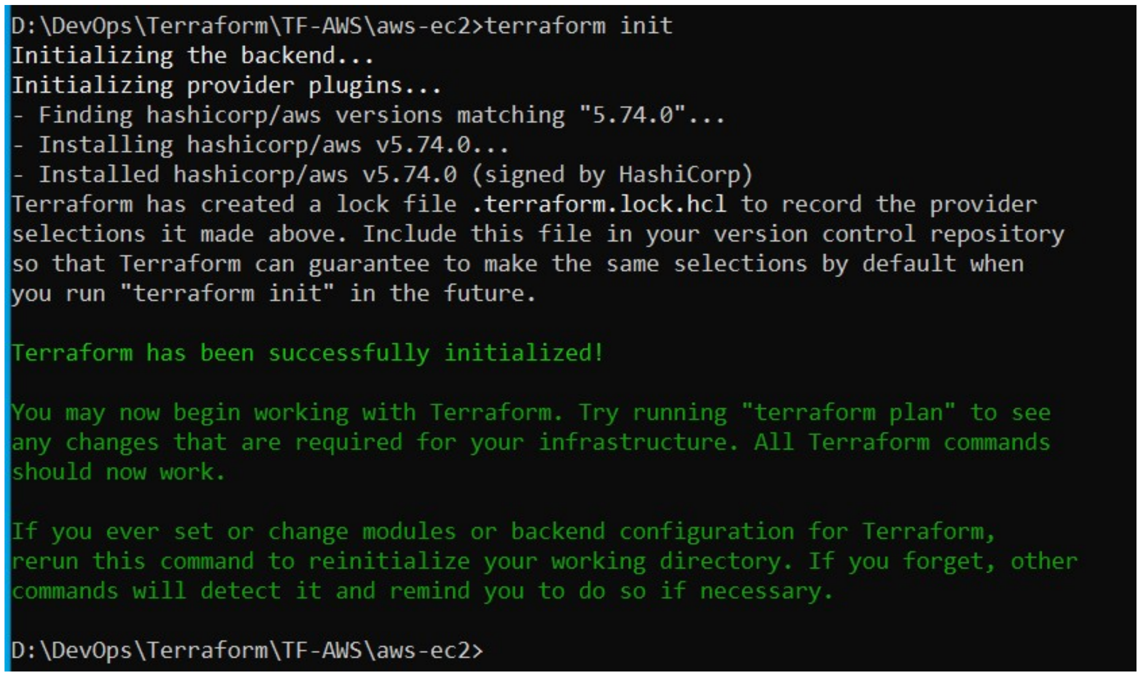
2. Create a main.tf file.
3. Open google.com and search for terraform.
4. Open the link https://www.terraform.io/.
5. Click on **Registry**.
6. Click on **Browse Providers**.
7. Click on **AWS**.
8. Click on **Use Provider**.
9. Copy the code and paste in main.tf file.

Sample main.tf file:

```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.74.0"
    }
  }
}

provider "aws" {
    region = "ap-south-1"
}

resource "aws_instance" "ec2-server" {
    ami = "ami-04a37924ffe27da53"
    instance_type = "t2.micro"
    tags = {
        Name = "vm-1"
    }
}
```



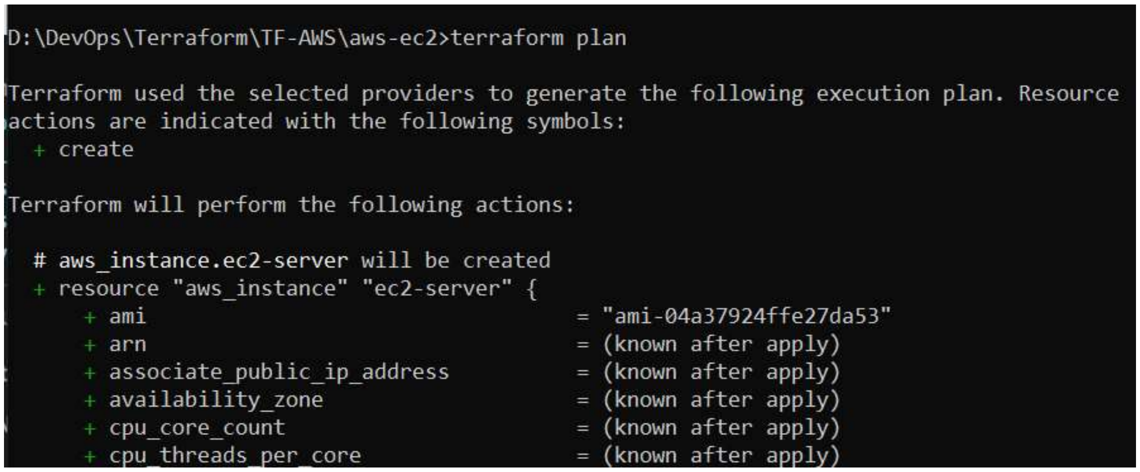
```
D:\DevOps\Terraform\TF-AWS\aws-ec2>terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.74.0"...
- Installing hashicorp/aws v5.74.0...
- Installed hashicorp/aws v5.74.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl 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.

D:\DevOps\Terraform\TF-AWS\aws-ec2>
```

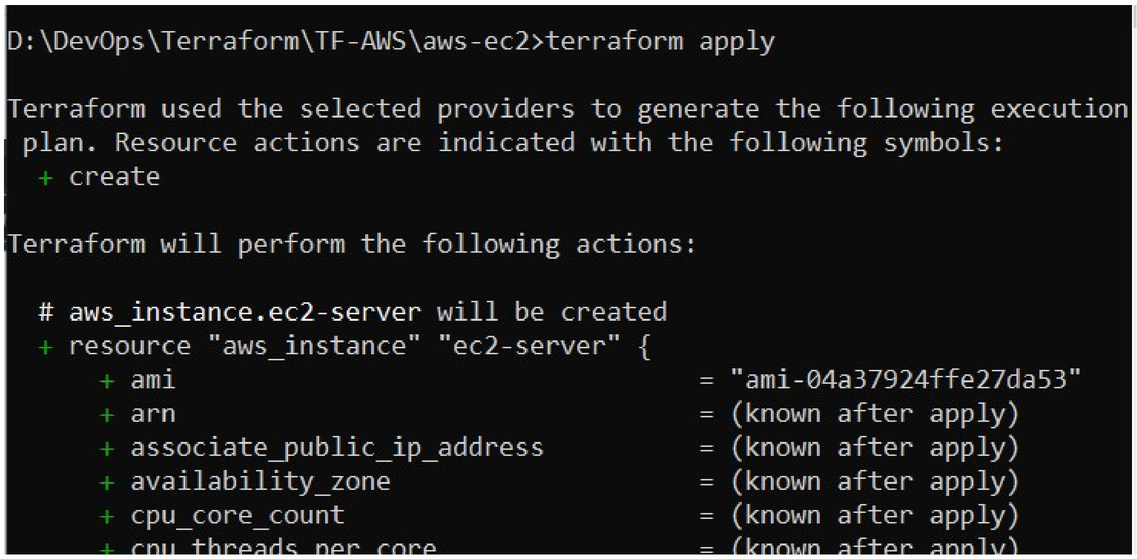


```
D:\DevOps\Terraform\TF-AWS\aws-ec2>terraform plan

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.ec2-server will be created
  + resource "aws_instance" "ec2-server" {
      + ami                                  = "ami-04a37924ffe27da53"
      + 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)
```



```
D:\DevOps\Terraform\TF-AWS\aws-ec2>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.ec2-server will be created
  + resource "aws_instance" "ec2-server" {
      + ami                                  = "ami-04a37924ffe27da53"
      + 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)
```