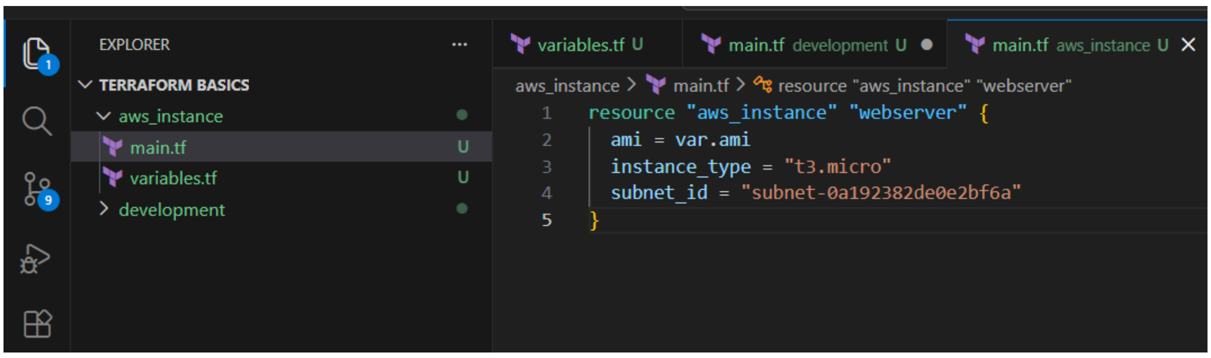
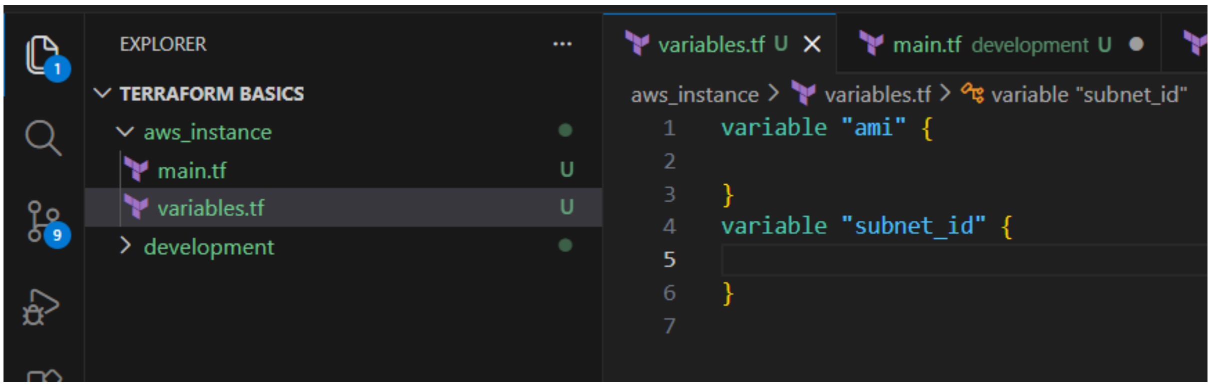
1. Watch the Terraform-06 video.

2. Execute the Script Shown in the Video.

Create a directory named as aws_instance in your terraform folder in that directory I have 2 files one is main.tf and variables.tf

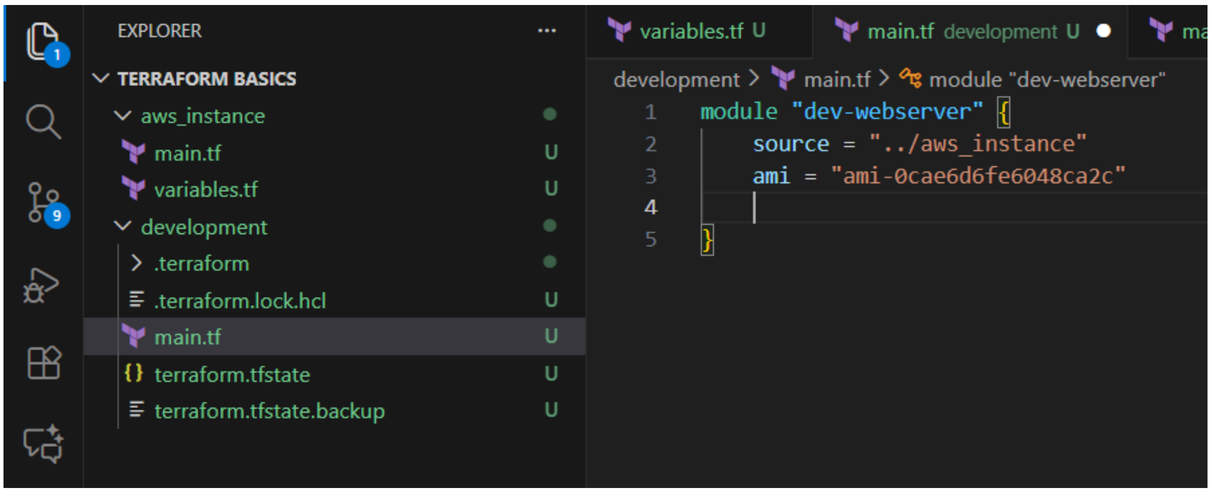


```
resource "aws_instance" "webserver" {
  ami = var.ami
  instance_type = "t3.micro"
  subnet_id = "subnet-0a192382de0e2bf6a"
}
```



```
variable "ami" {

}
variable "subnet_id" {

}

```

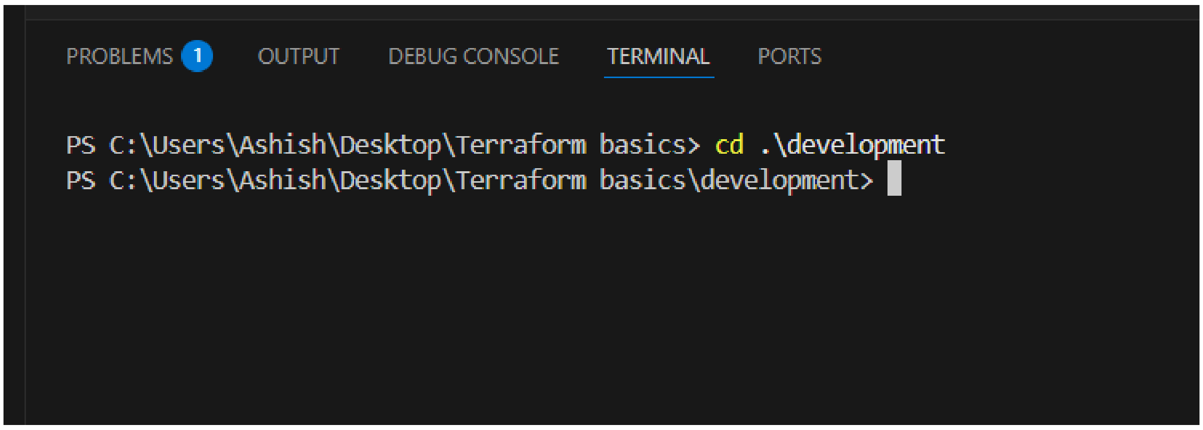
Create another folder named as development in the terraform folder in that I have a file main.tf



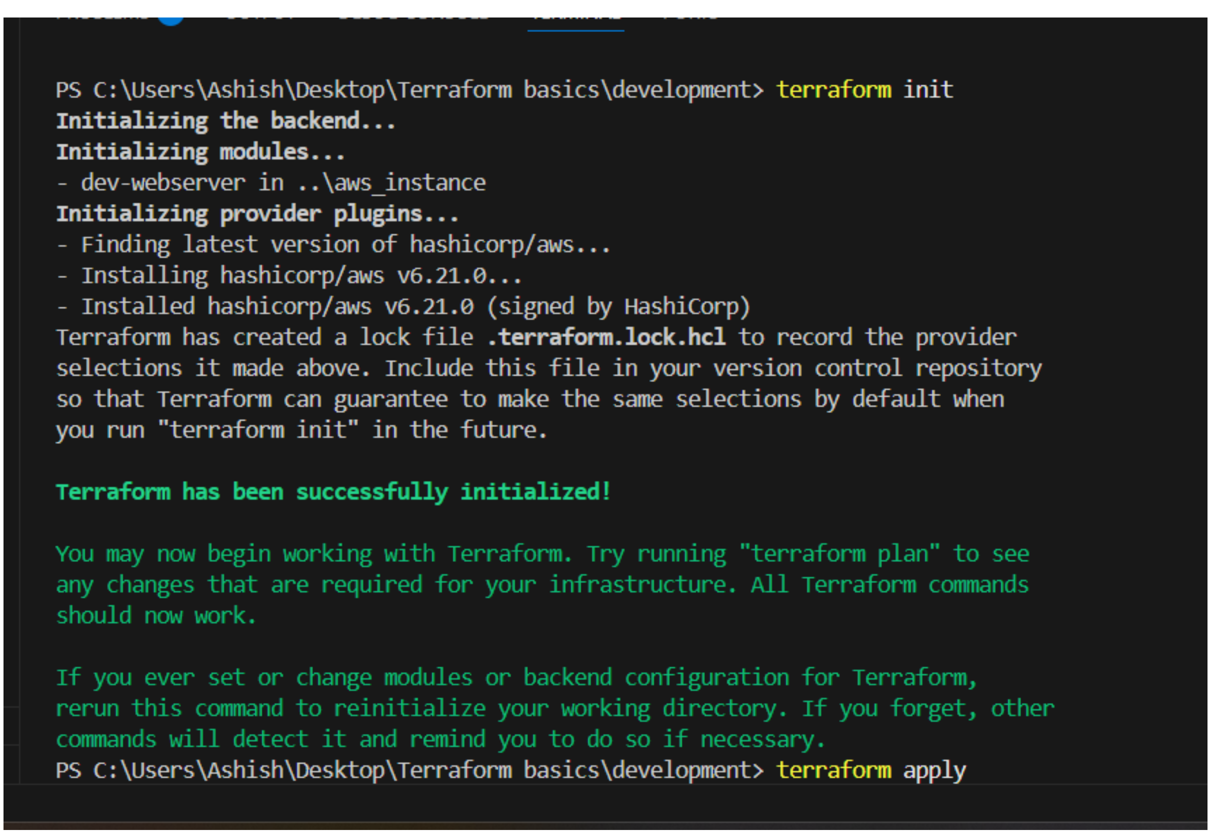
```
module "dev-webserver" {
    source = "../aws_instance"
    ami = "ami-0cae6d6fe6048ca2c"

}
```

Then change your directory to development by using

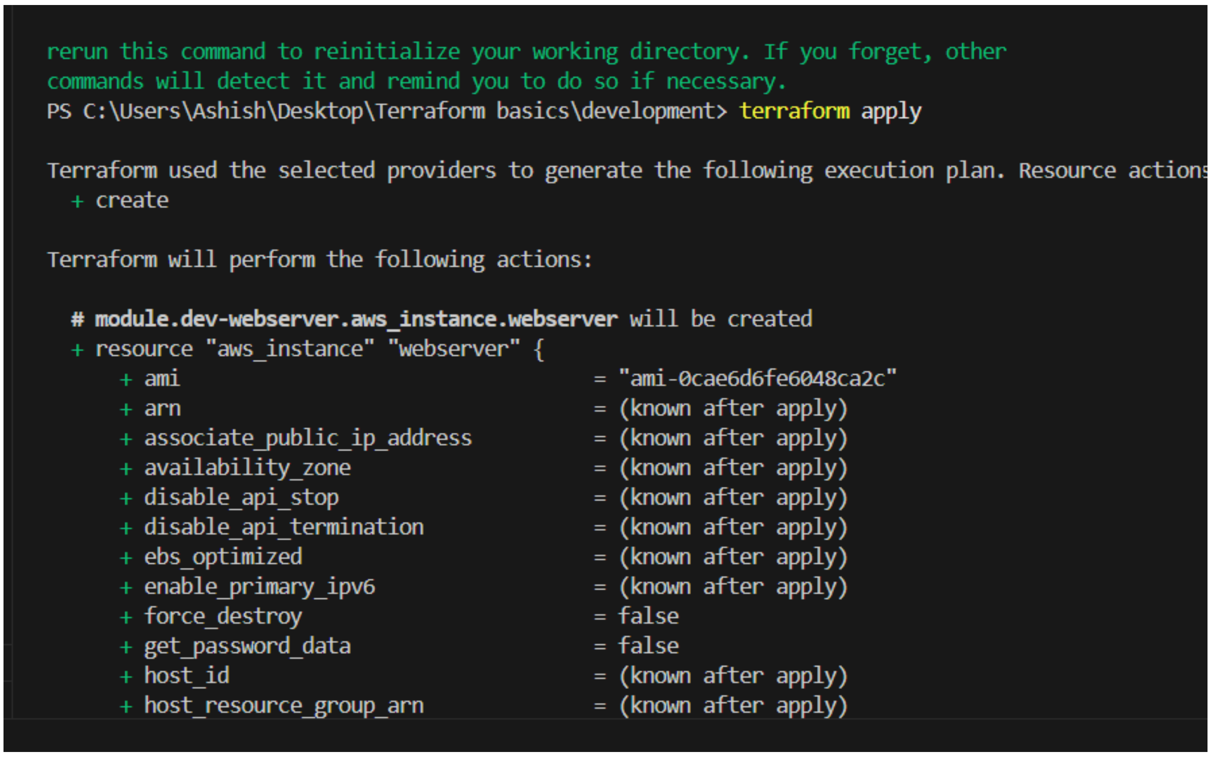
cd .\development



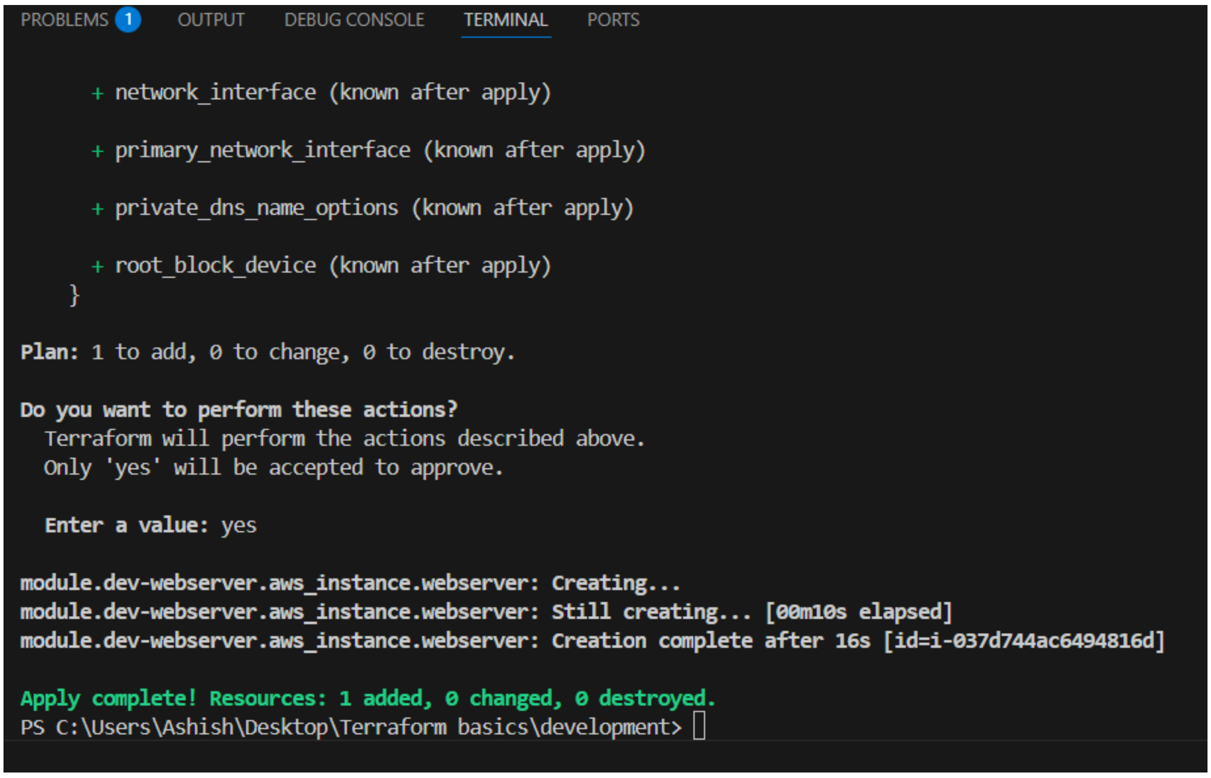
```
PS C:\Users\Ashish\Desktop\Terraform basics> cd .\development
PS C:\Users\Ashish\Desktop\Terraform basics\development>
```



```
PS C:\Users\Ashish\Desktop\Terraform basics\development> terraform init
Initializing the backend...
Initializing modules...
- dev-webserver in ..\aws_instance
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.21.0...
- Installed hashicorp/aws v6.21.0 (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.
PS C:\Users\Ashish\Desktop\Terraform basics\development> terraform apply
```

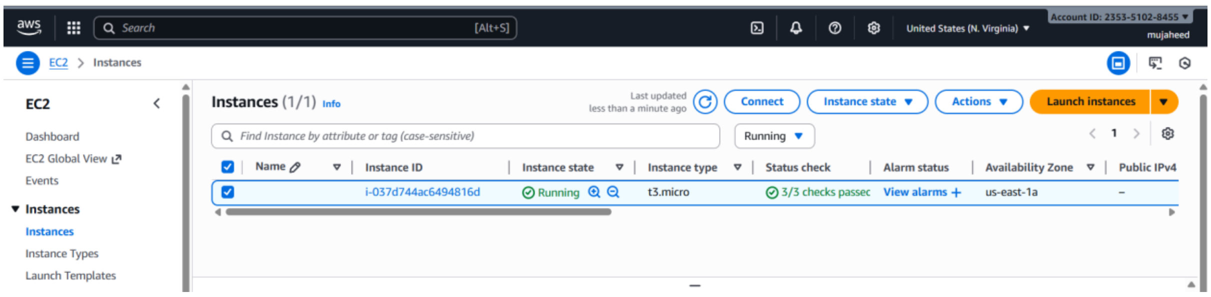


```
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS C:\Users\Ashish\Desktop\Terraform basics\development> terraform apply

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

Terraform will perform the following actions:

  # module.dev-webserver.aws_instance.webserver will be created
  + resource "aws_instance" "webserver" {
      + ami                                  = "ami-0cae6d6fe6048ca2c"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + enable_primary_ipv6                  = (known after apply)
      + force_destroy                        = false
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
```



```
      + network_interface (known after apply)

      + primary_network_interface (known after apply)

      + private_dns_name_options (known after apply)

      + root_block_device (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

module.dev-webserver.aws_instance.webserver: Creating...
module.dev-webserver.aws_instance.webserver: Still creating... [00m10s elapsed]
module.dev-webserver.aws_instance.webserver: Creation complete after 16s [id=i-037d744ac6494816d]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\Ashish\Desktop\Terraform basics\development> 
```

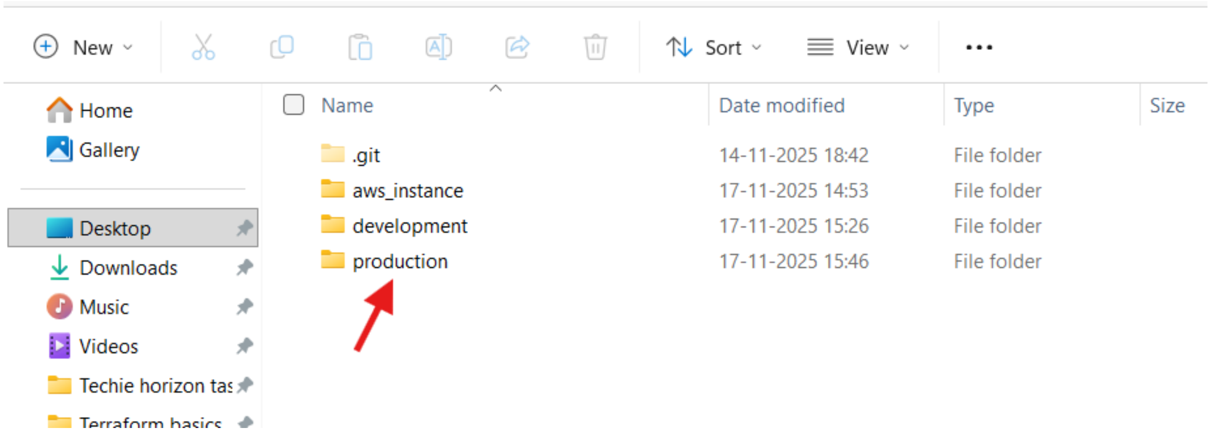
An instance will be created.



Instances (1/1)

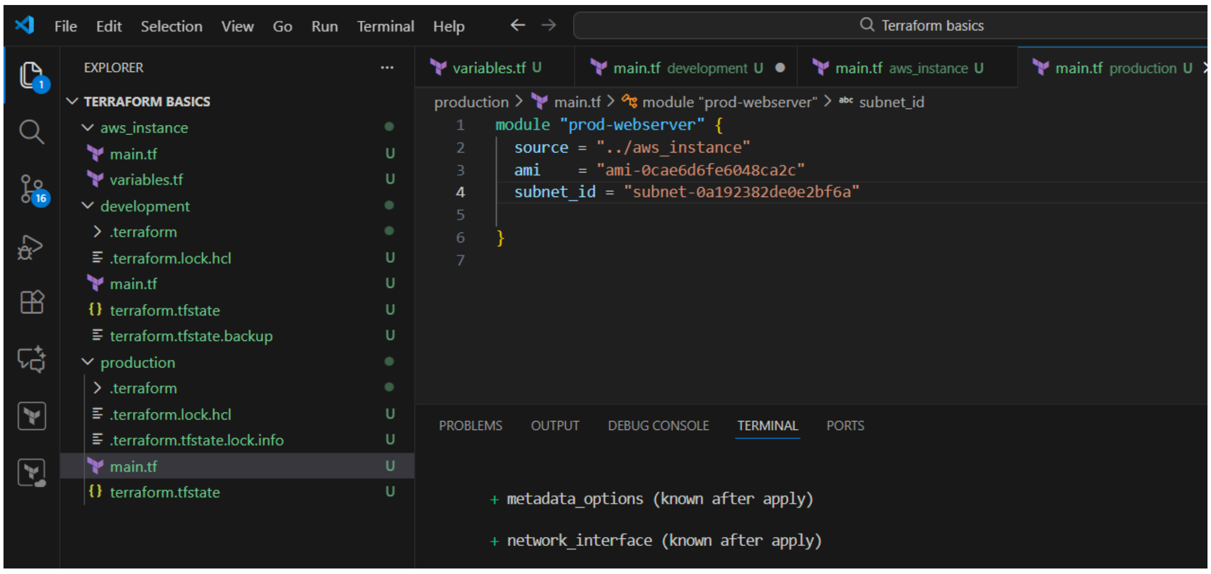
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
	i-037d744ac6494816d	Running	t3.micro	3/3 checks passec	View alarms +	us-east-1a	–

If you want to create one more instance for production environment you create a directory in terraform folder.



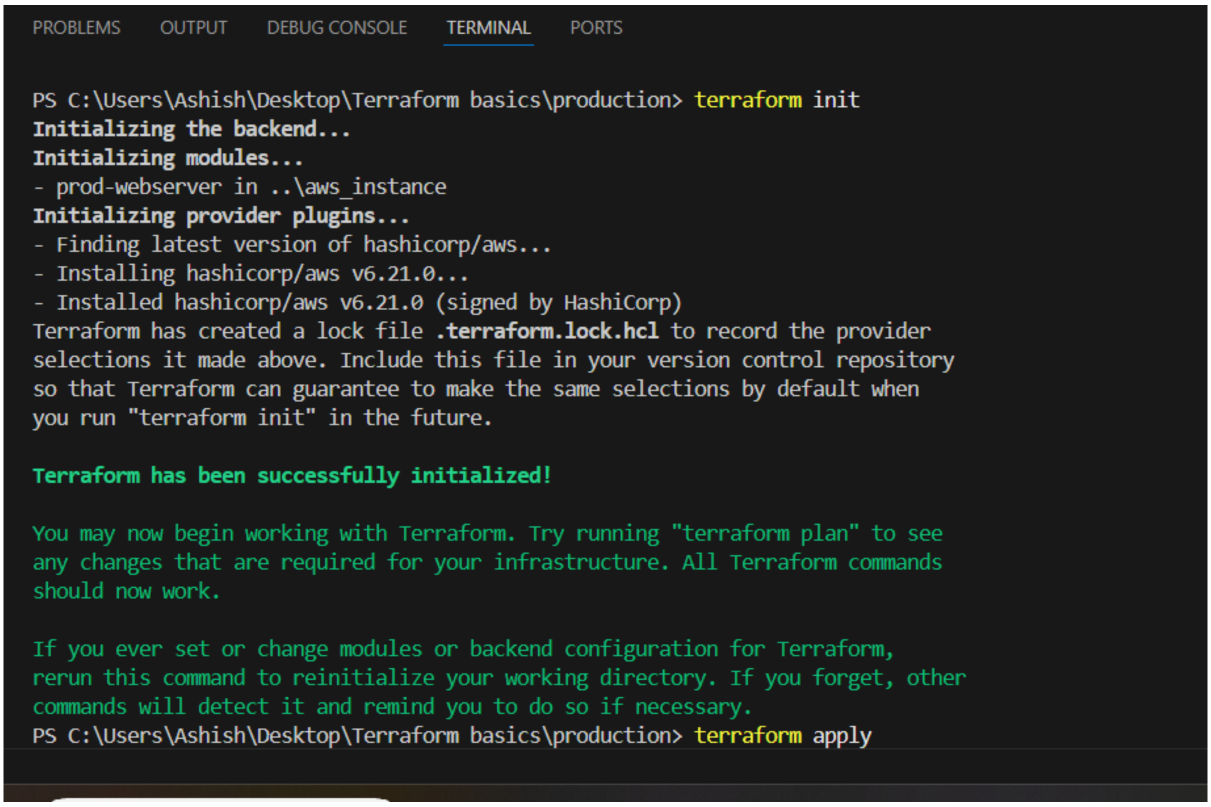
Name	Date modified	Type	Size
.git	14-11-2025 18:42	File folder	
aws_instance	17-11-2025 14:53	File folder	
development	17-11-2025 15:26	File folder	
production	17-11-2025 15:46	File folder	

In that create a file.

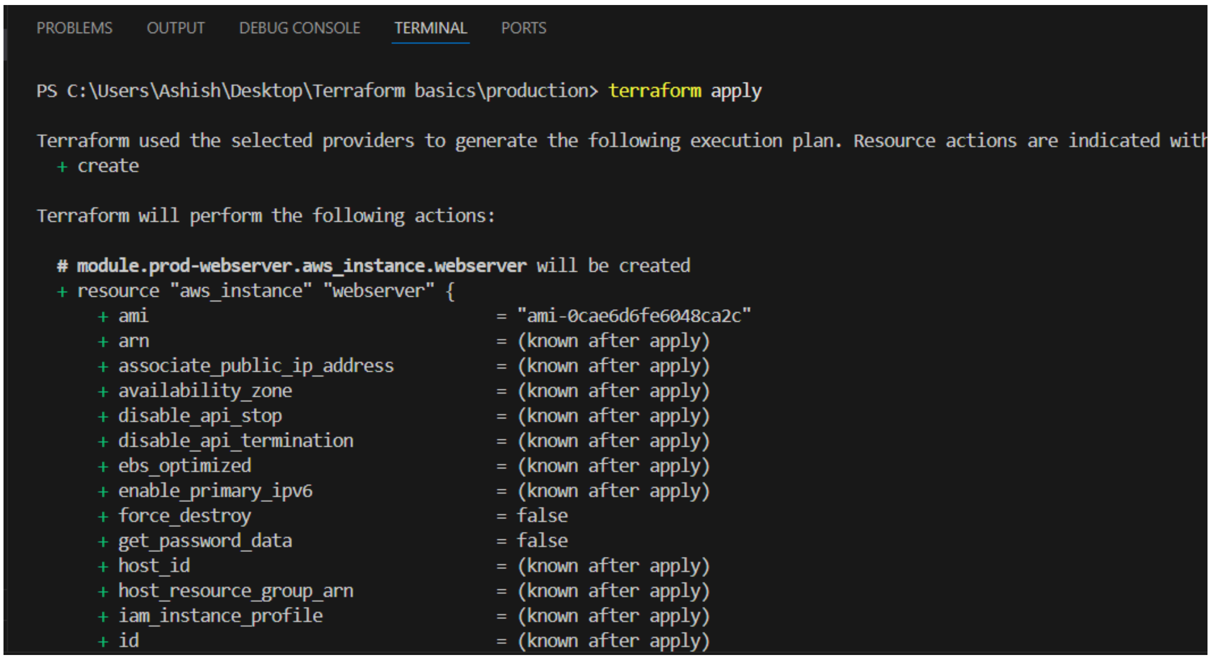


```
module "prod-webserver" {
  source = "../aws_instance"
  ami    = "ami-0cae6d6fe6048ca2c"
  subnet_id = "subnet-0a192382de0e2bf6a"

}
```

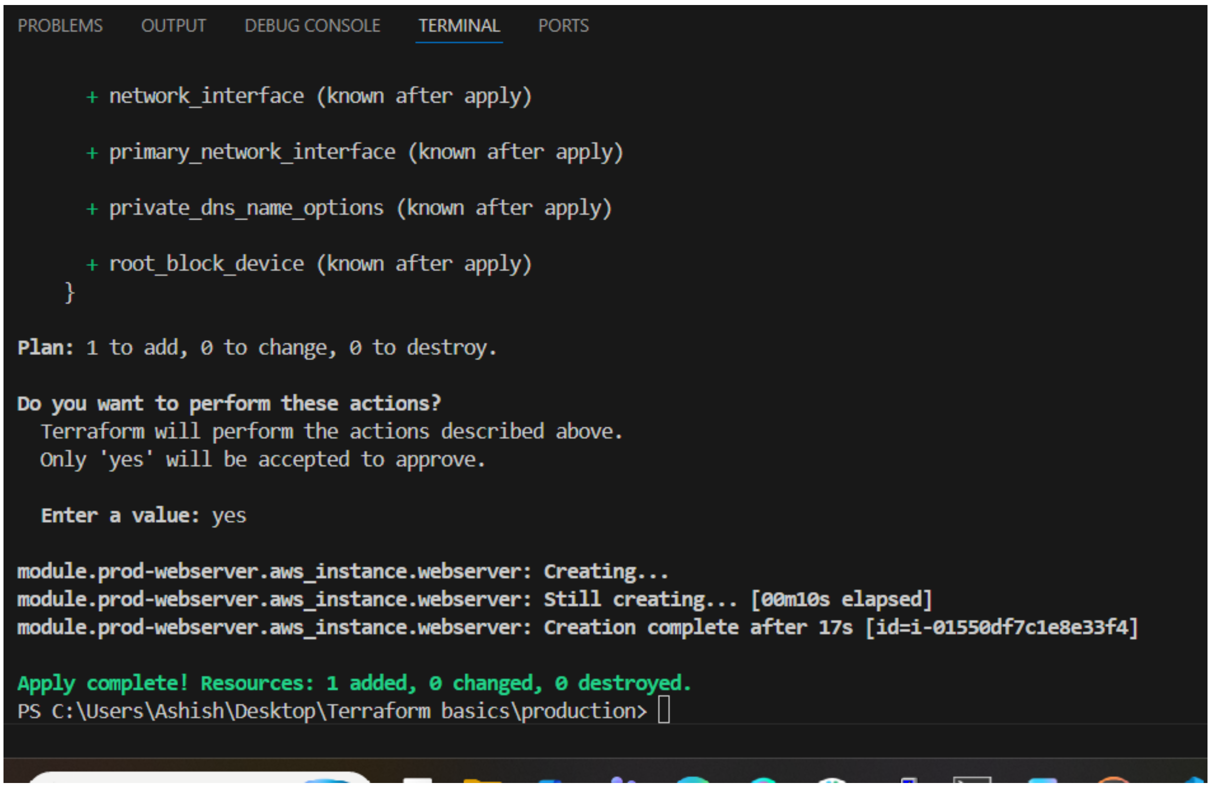
```
+ metadata_options (known after apply)

+ network_interface (known after apply)
```



```
PS C:\Users\Ashish\Desktop\Terraform basics\production> terraform init
Initializing the backend...
Initializing modules...
- prod-webserver in ..\aws_instance
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.21.0...
- Installed hashicorp/aws v6.21.0 (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.
PS C:\Users\Ashish\Desktop\Terraform basics\production> terraform apply
```

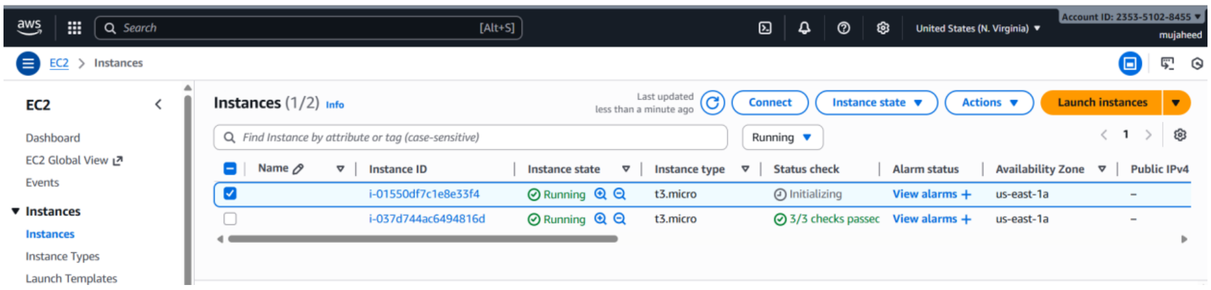


```
PS C:\Users\Ashish\Desktop\Terraform basics\production> terraform apply

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

Terraform will perform the following actions:

  # module.prod-webserver.aws_instance.webserver will be created
  + resource "aws_instance" "webserver" {
      + ami                                  = "ami-0cae6d6fe6048ca2c"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + enable_primary_ipv6                  = (known after apply)
      + force_destroy                        = false
      + 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)
```



```
      + network_interface (known after apply)

      + primary_network_interface (known after apply)

      + private_dns_name_options (known after apply)

      + root_block_device (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

module.prod-webserver.aws_instance.webserver: Creating...
module.prod-webserver.aws_instance.webserver: Still creating... [00m10s elapsed]
module.prod-webserver.aws_instance.webserver: Creation complete after 17s [id=i-01550df7c1e8e33f4]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\Ashish\Desktop\Terraform basics\production>
```

Here an instance has been created for production purpose.

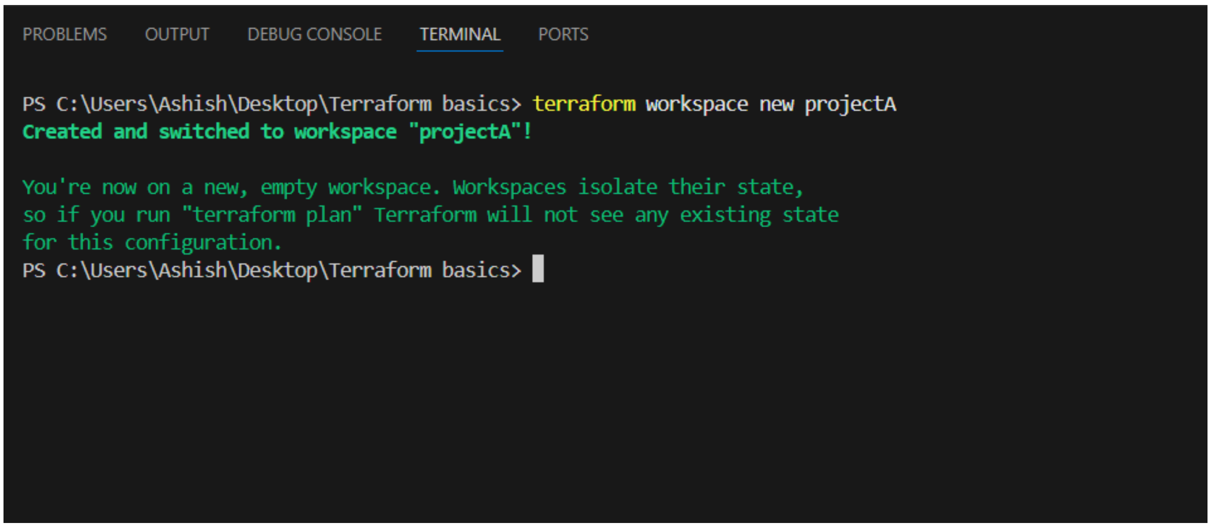


Instances (1/2)

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
✓		i-01550df7c1e8e33f4	Running	t3.micro	Initializing	View alarms +	us-east-1a	–
		i-037d744ac6494816d	Running	t3.micro	3/3 checks passec	View alarms +	us-east-1a	–

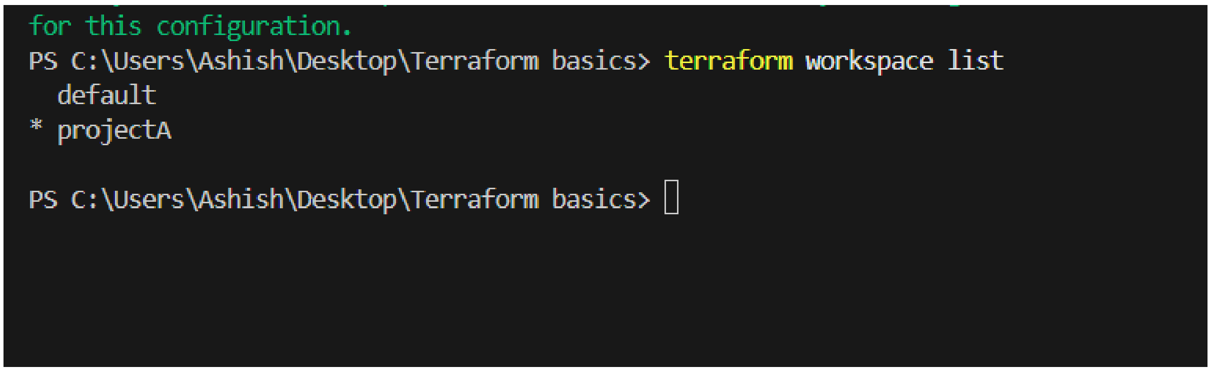
Terraform workspace:

- terraform workspace new projectA



```
PS C:\Users\Ashish\Desktop\Terraform basics> terraform workspace new projectA
Created and switched to workspace "projectA"!

You're now on a new, empty workspace. Workspaces isolate their state,
so if you run "terraform plan" Terraform will not see any existing state
for this configuration.
PS C:\Users\Ashish\Desktop\Terraform basics>
```

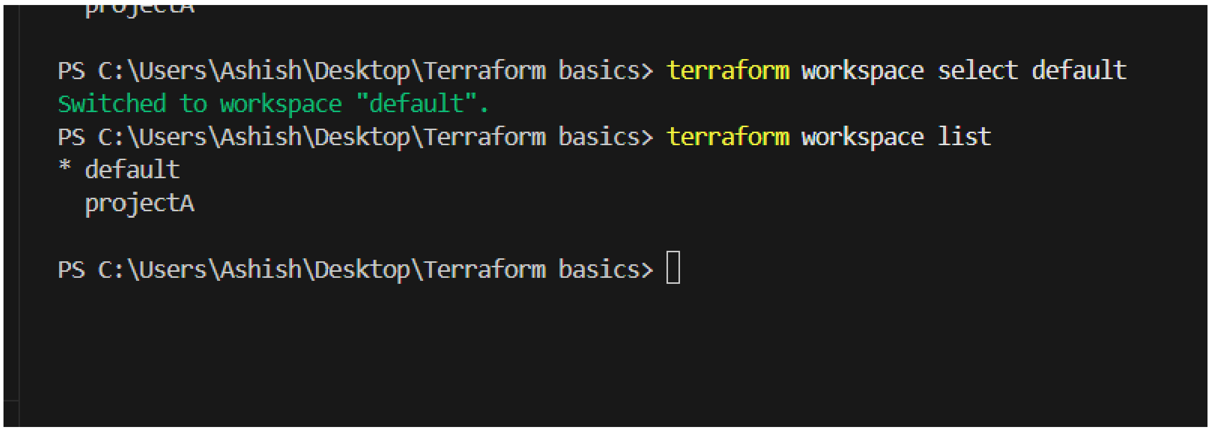
- terraform workspace list



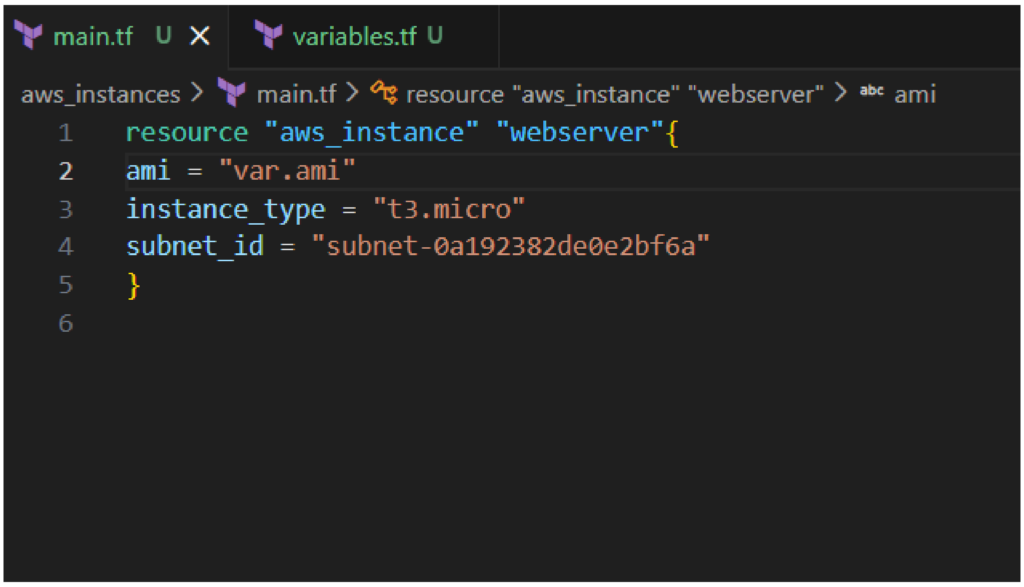
```
for this configuration.
PS C:\Users\Ashish\Desktop\Terraform basics> terraform workspace list
  default
* projectA

PS C:\Users\Ashish\Desktop\Terraform basics>
```

To select different workspace

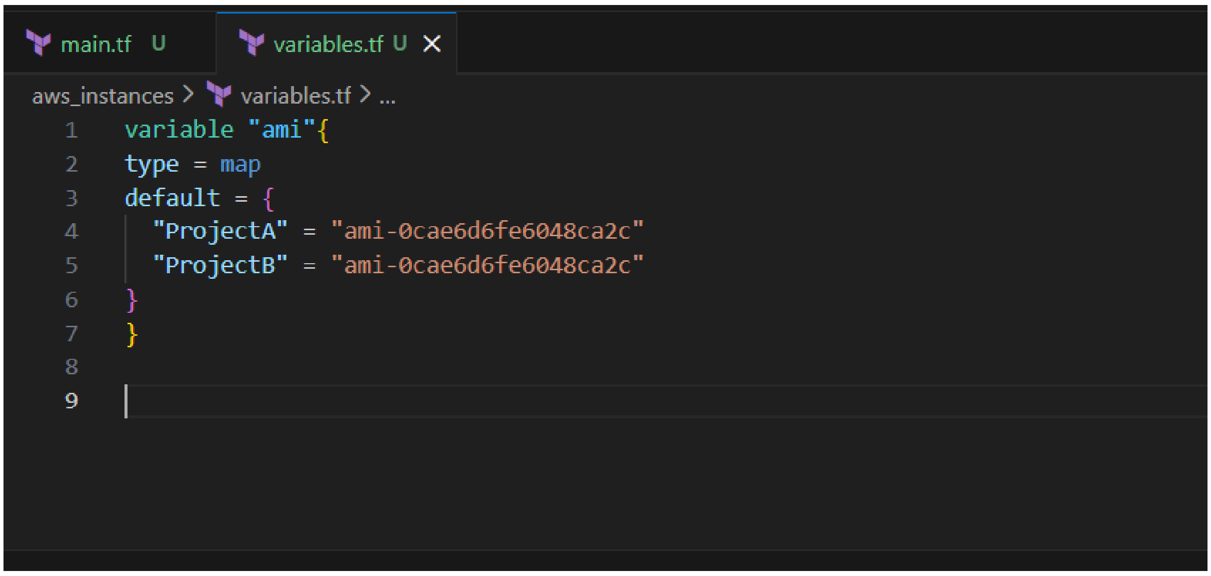
- terraform workspace select default



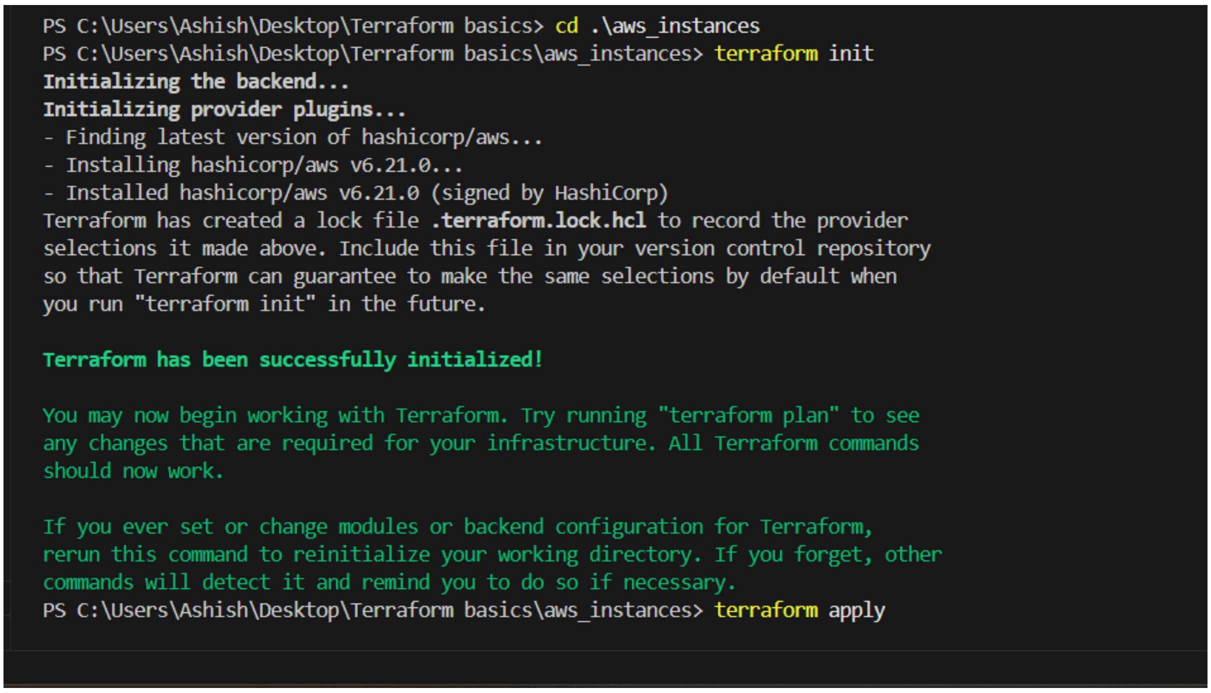
```
  projectA

PS C:\Users\Ashish\Desktop\Terraform basics> terraform workspace select default
Switched to workspace "default".
PS C:\Users\Ashish\Desktop\Terraform basics> terraform workspace list
* default
  projectA

PS C:\Users\Ashish\Desktop\Terraform basics>
```


main.tf U
variables.tf U
aws_instances > main.tf > resource "aws_instance" "webserver" > ami

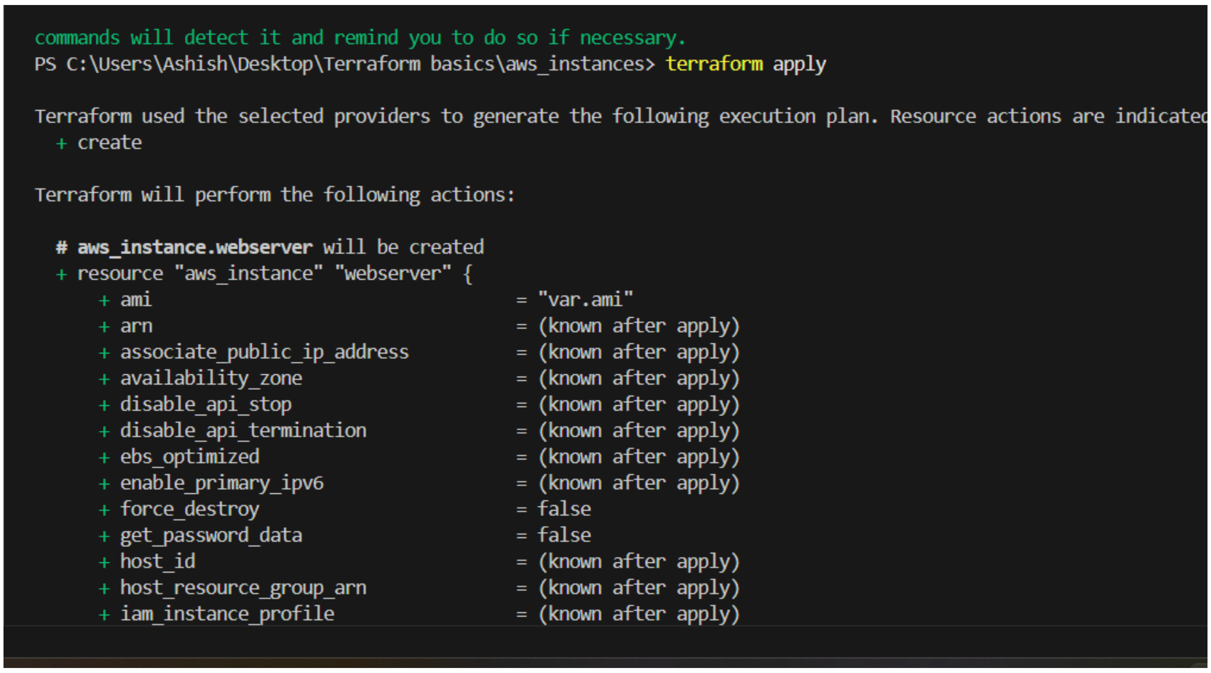
```
resource "aws_instance" "webserver"{
ami = "var.ami"
instance_type = "t3.micro"
subnet_id = "subnet-0a192382de0e2bf6a"
}

```



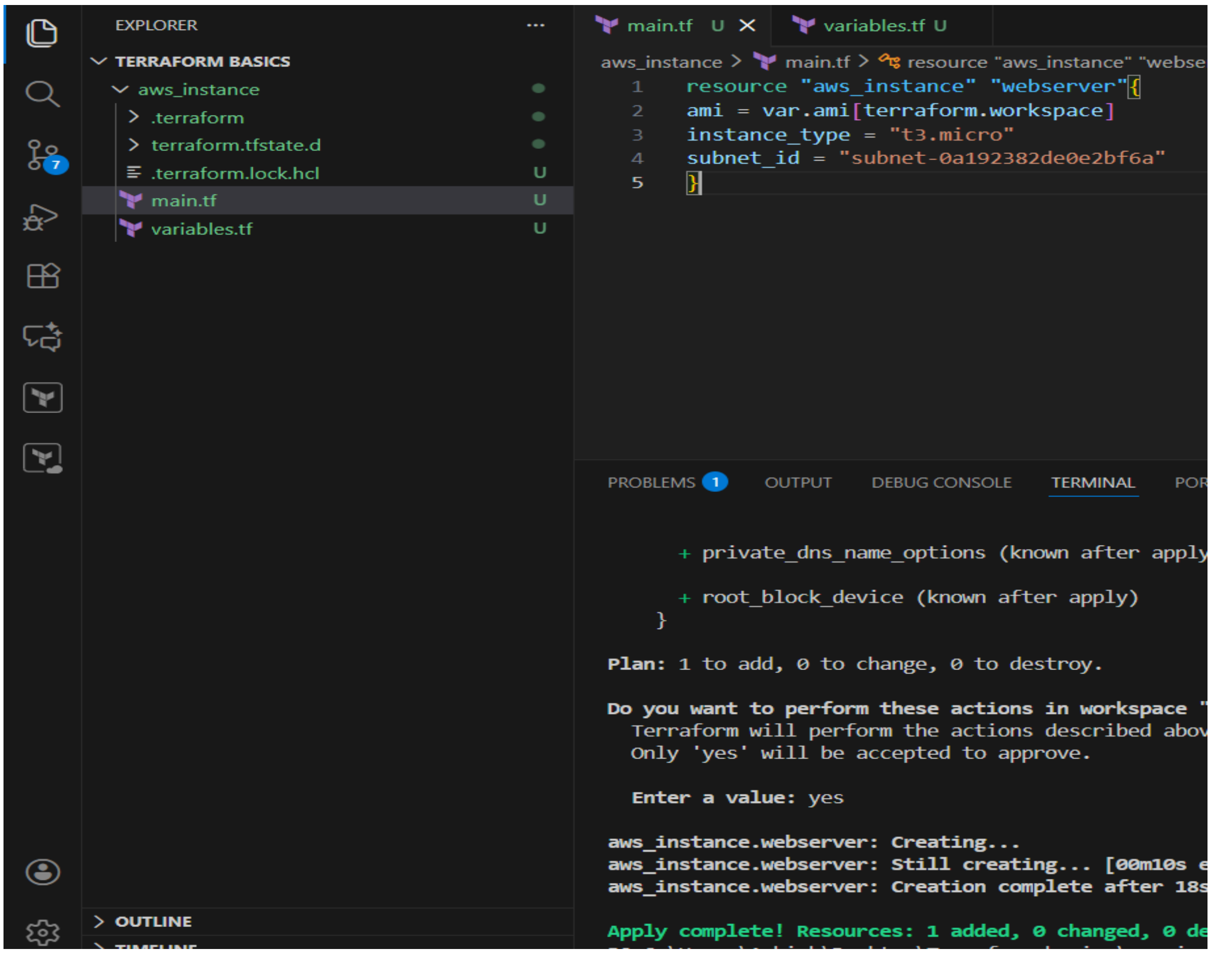
```
variable "ami"{
type = map
default = {
  "ProjectA" = "ami-0cae6d6fe6048ca2c"
  "ProjectB" = "ami-0cae6d6fe6048ca2c"
}
}

```



```
PS C:\Users\Ashish\Desktop\Terraform basics> cd .\aws_instances
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instances> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.21.0...
- Installed hashicorp/aws v6.21.0 (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.
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instances> terraform apply
```

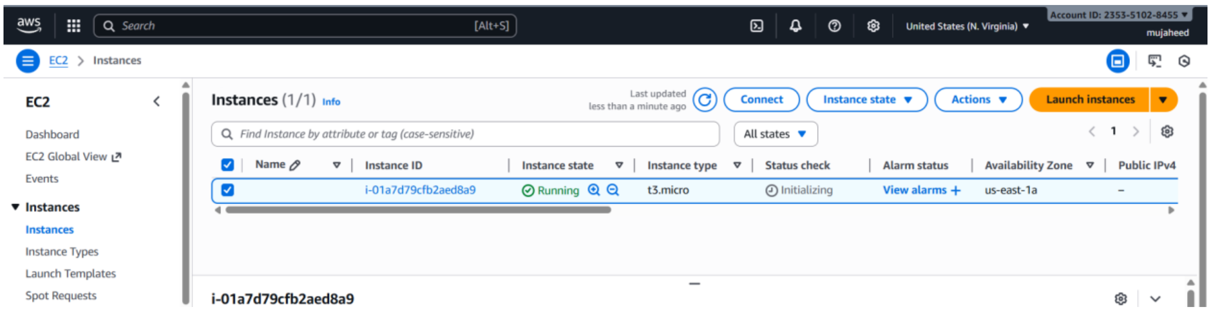


```
commands will detect it and remind you to do so if necessary.
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instances> terraform apply

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

Terraform will perform the following actions:

  # aws_instance.webserver will be created
  + resource "aws_instance" "webserver" {
      + ami                                  = "var.ami"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + enable_primary_ipv6                  = (known after apply)
      + force_destroy                        = false
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
```



```
resource "aws_instance" "webserver"{
ami = var.ami[terraform.workspace]
instance_type = "t3.micro"
subnet_id = "subnet-0a192382de0e2bf6a"
}
```

```
      + private_dns_name_options (known after apply

      + root_block_device (known after apply)
    }

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

Do you want to perform these actions in workspace "
  Terraform will perform the actions described abov
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_instance.webserver: Creating...
aws_instance.webserver: Still creating... [00m10s e
aws_instance.webserver: Creation complete after 18s

Apply complete! Resources: 1 added, 0 changed, 0 de
```



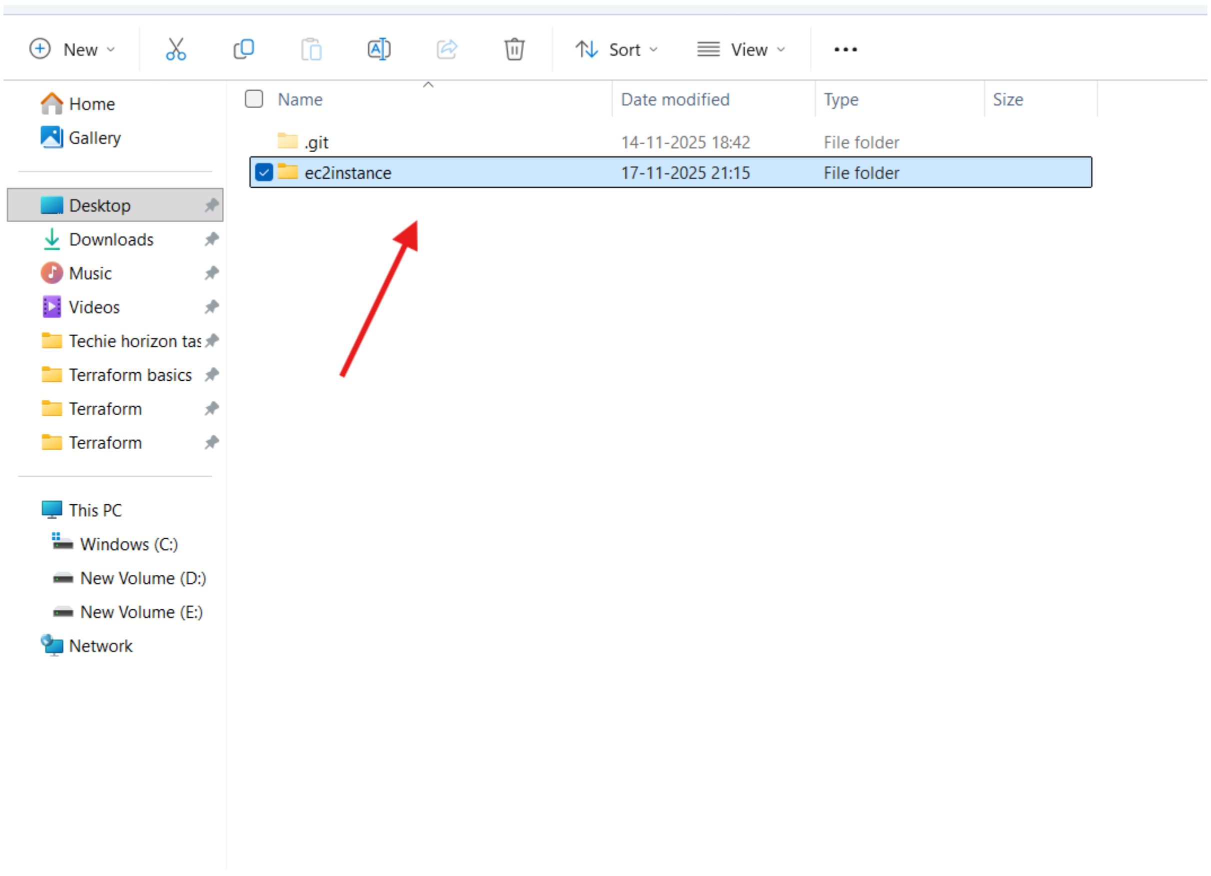
Instances (1/1) Info

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
✓		i-01a7d79cfb2aed8a9	Running	t3.micro	Initializing	View alarms +	us-east-1a	–

i-01a7d79cfb2aed8a9

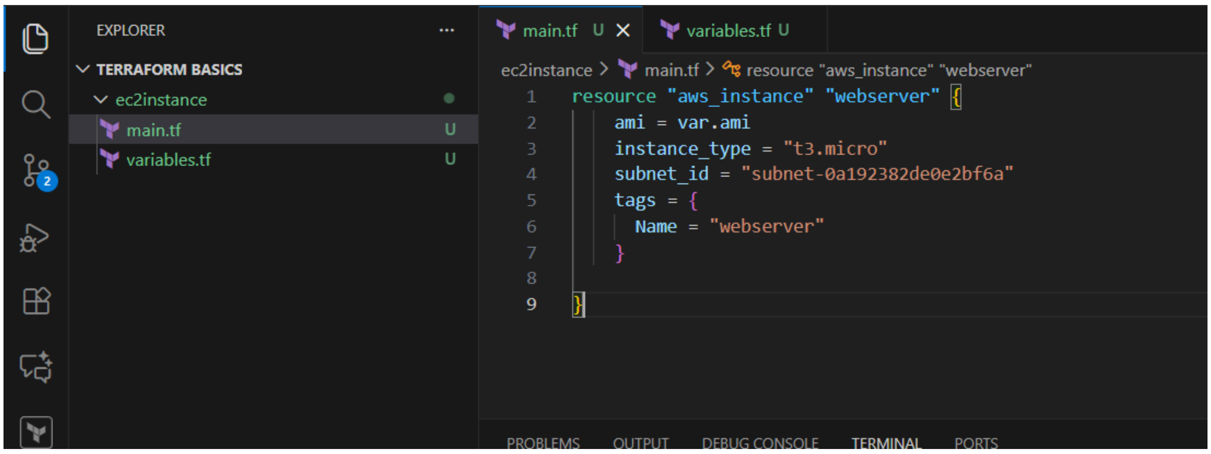
3 .Provision EC2, S3, and VPC using Terraform modules.

Create a directory named as ec2instance.

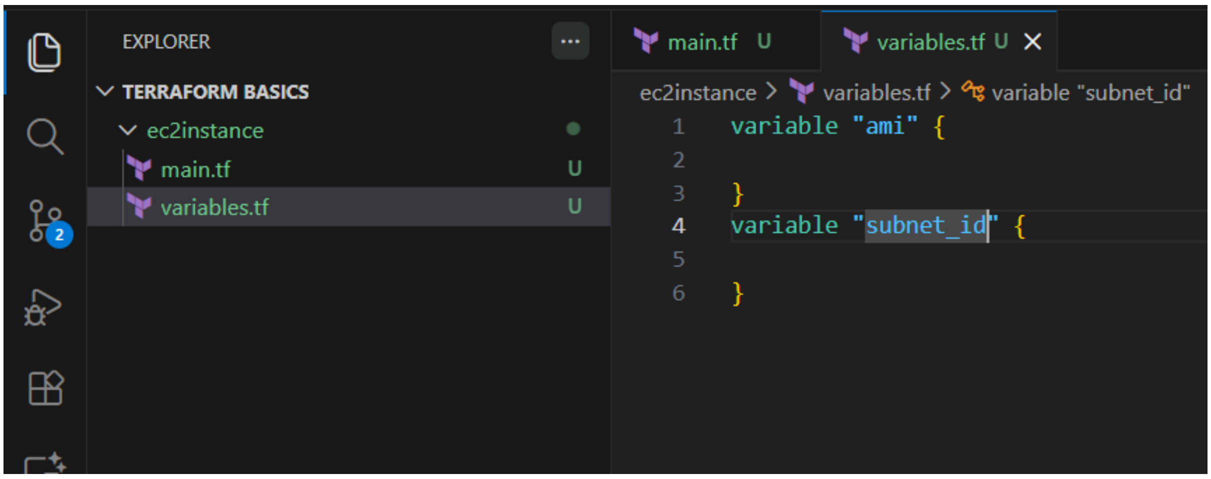


Name	Date modified	Type	Size
.git	14-11-2025 18:42	File folder	
ec2instance	17-11-2025 21:15	File folder	

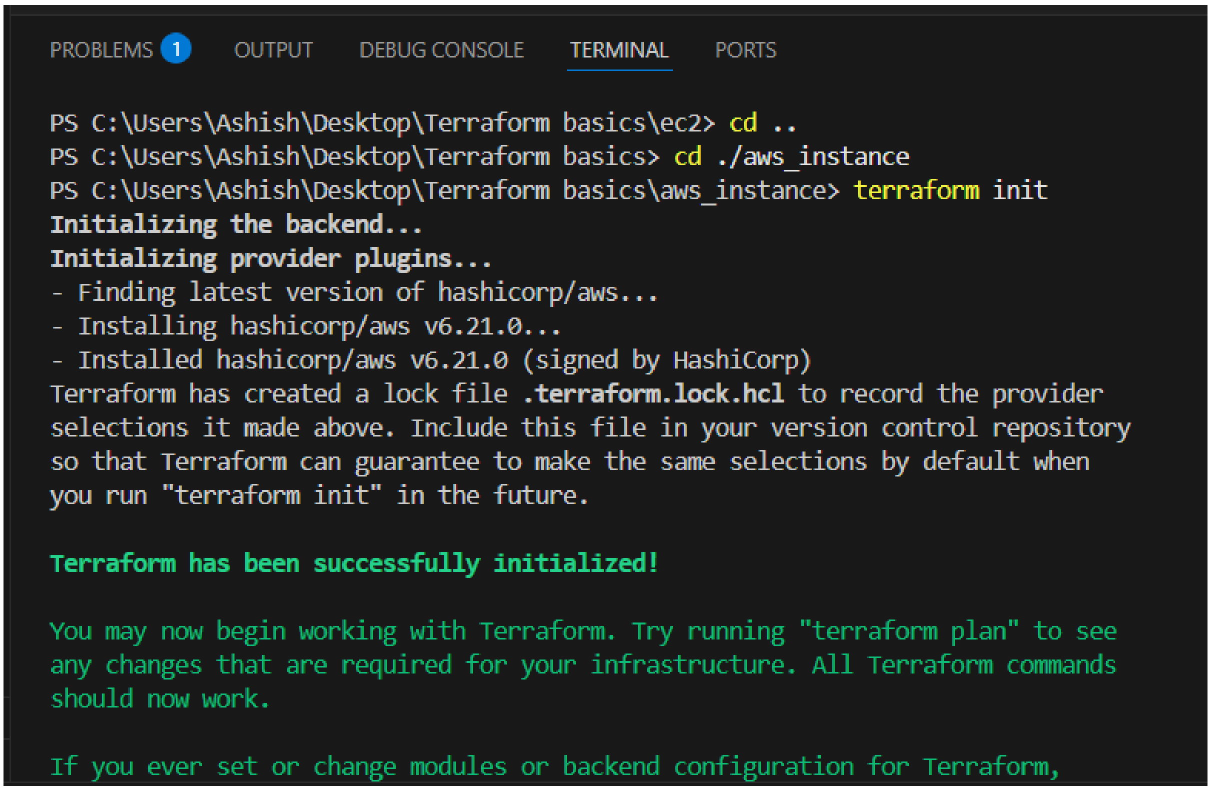
In that directory I have main.tf, variable.tf



```
resource "aws_instance" "webserver" {
    ami = var.ami
    instance_type = "t3.micro"
    subnet_id = "subnet-0a192382de0e2bf6a"
    tags = {
      Name = "webserver"
    }

}
```

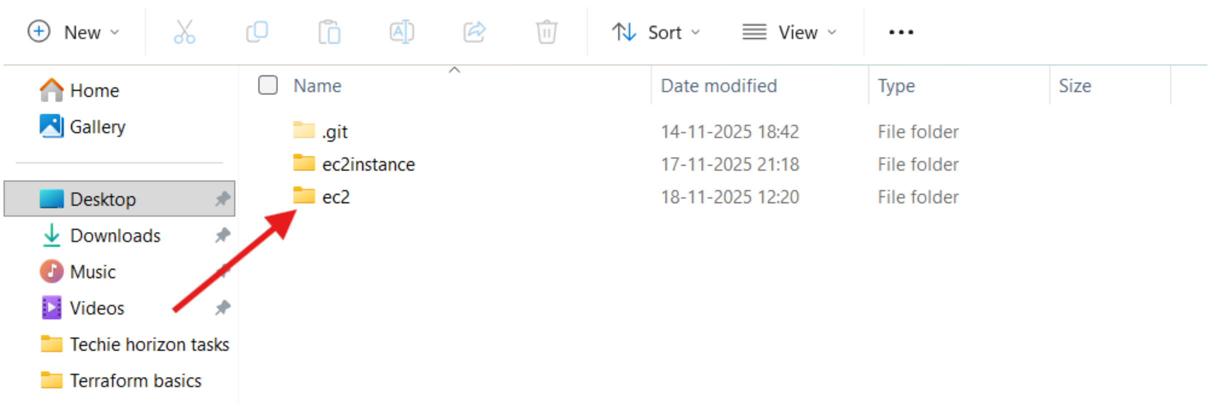


```
variable "ami" {

}
variable "subnet_id" {

}
```



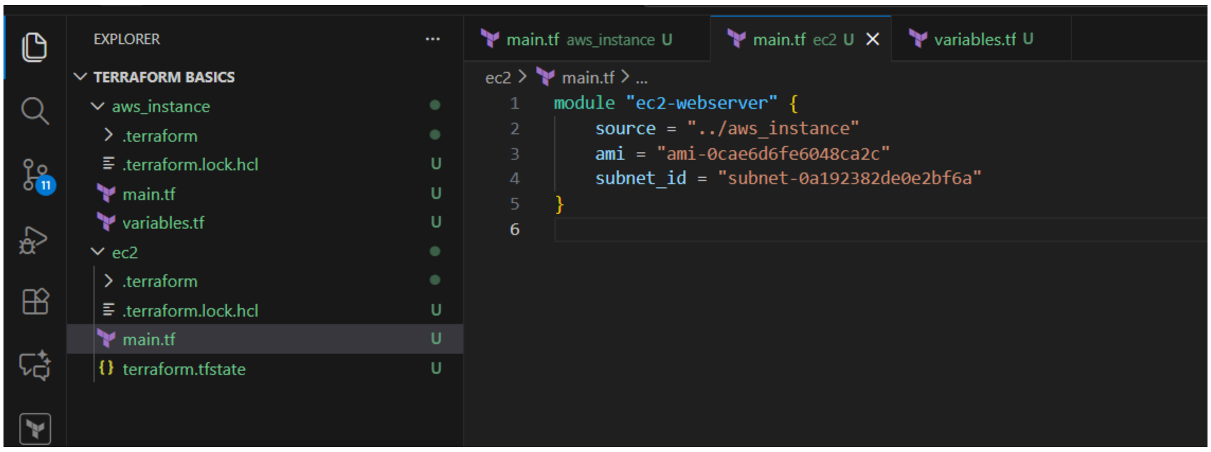
```
PS C:\Users\Ashish\Desktop\Terraform basics\ec2> cd ..
PS C:\Users\Ashish\Desktop\Terraform basics> cd ./aws_instance
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.21.0...
- Installed hashicorp/aws v6.21.0 (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,
```

Create a directory in your terraform folder named as ec2.

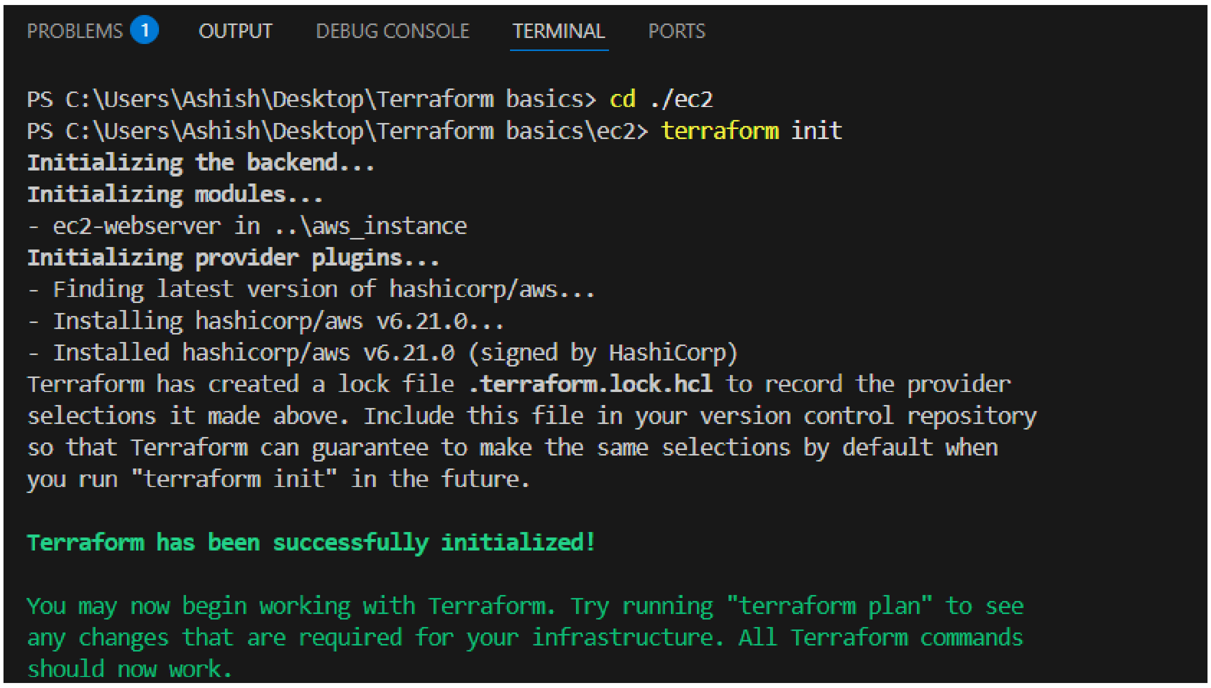


Name	Date modified	Type	Size
.git	14-11-2025 18:42	File folder	
ec2instance	17-11-2025 21:18	File folder	
ec2	18-11-2025 12:20	File folder	

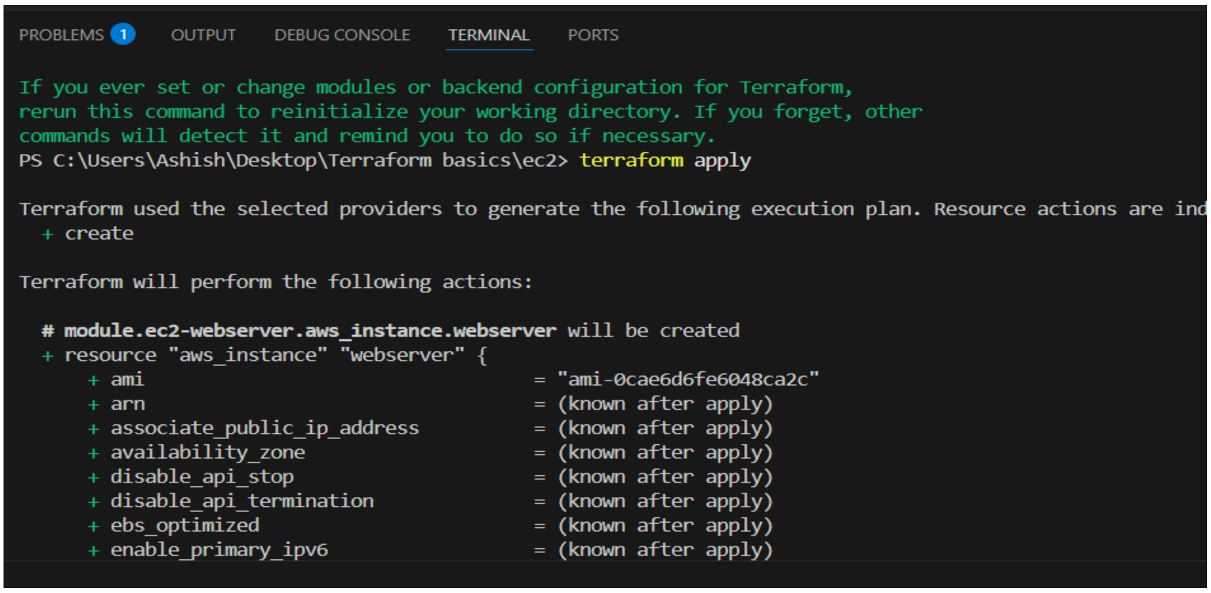
Create a file in ec2 as main.



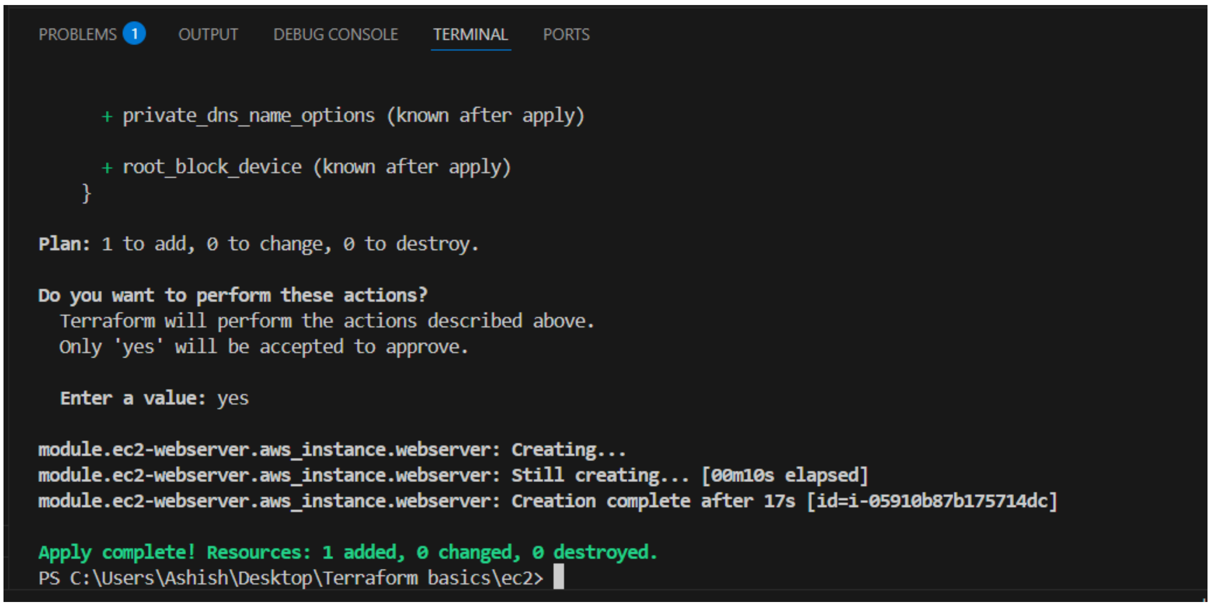
```
module "ec2-webserver" {
    source = "../aws_instance"
    ami = "ami-0cae6d6fe6048ca2c"
    subnet_id = "subnet-0a192382de0e2bf6a"
}
```



```
PS C:\Users\Ashish\Desktop\Terraform basics> cd ./ec2
PS C:\Users\Ashish\Desktop\Terraform basics\ec2> terraform init
Initializing the backend...
Initializing modules...
- ec2-webserver in ..\aws_instance
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.21.0...
- Installed hashicorp/aws v6.21.0 (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.
```

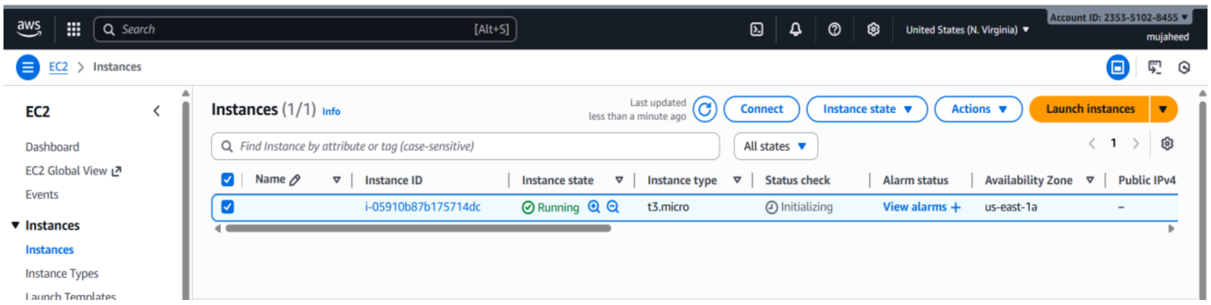


```
If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS C:\Users\Ashish\Desktop\Terraform basics\ec2> terraform apply

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

Terraform will perform the following actions:

  # module.ec2-webserver.aws_instance.webserver will be created
  + resource "aws_instance" "webserver" {
      + ami                                  = "ami-0cae6d6fe6048ca2c"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + enable_primary_ipv6                  = (known after apply)
```



```
      + private_dns_name_options (known after apply)

      + root_block_device (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

module.ec2-webserver.aws_instance.webserver: Creating...
module.ec2-webserver.aws_instance.webserver: Still creating... [00m10s elapsed]
module.ec2-webserver.aws_instance.webserver: Creation complete after 17s [id=i-05910b87b175714dc]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\Ashish\Desktop\Terraform basics\ec2> 
```

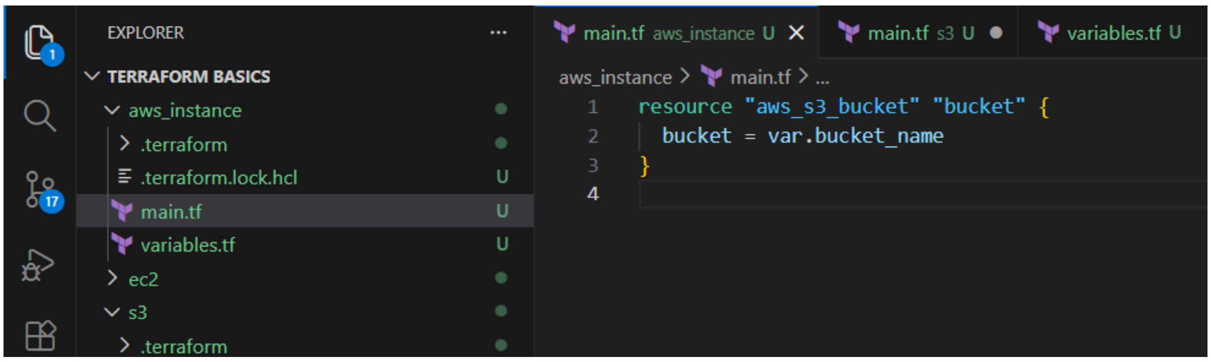
An ec2 instance has been created using terraform module.



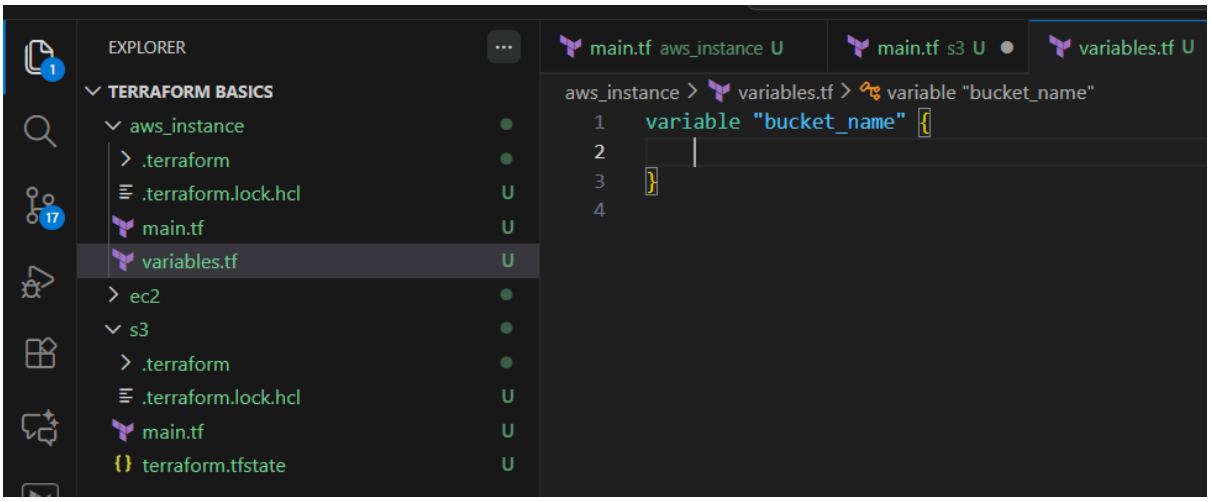
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
	i-05910b87b175714dc	Running	t3.micro	Initializing	View alarms +	us-east-1a	–

S3:

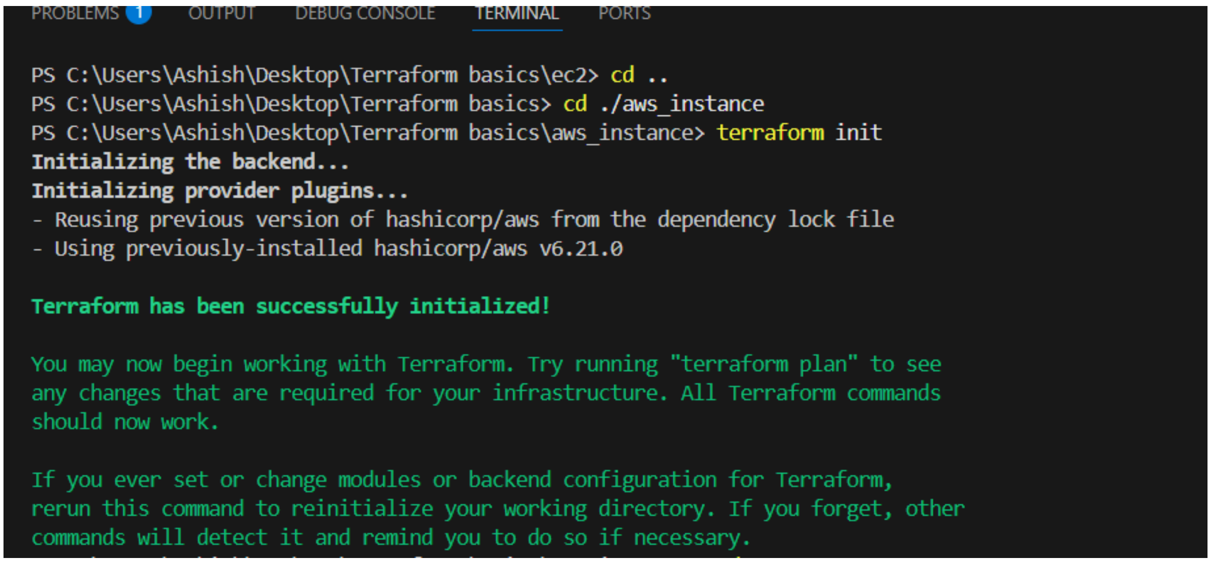
In the aws_instance folder change the main.tf,variable.tf files.



```
resource "aws_s3_bucket" "bucket" {
  bucket = var.bucket_name
}
```

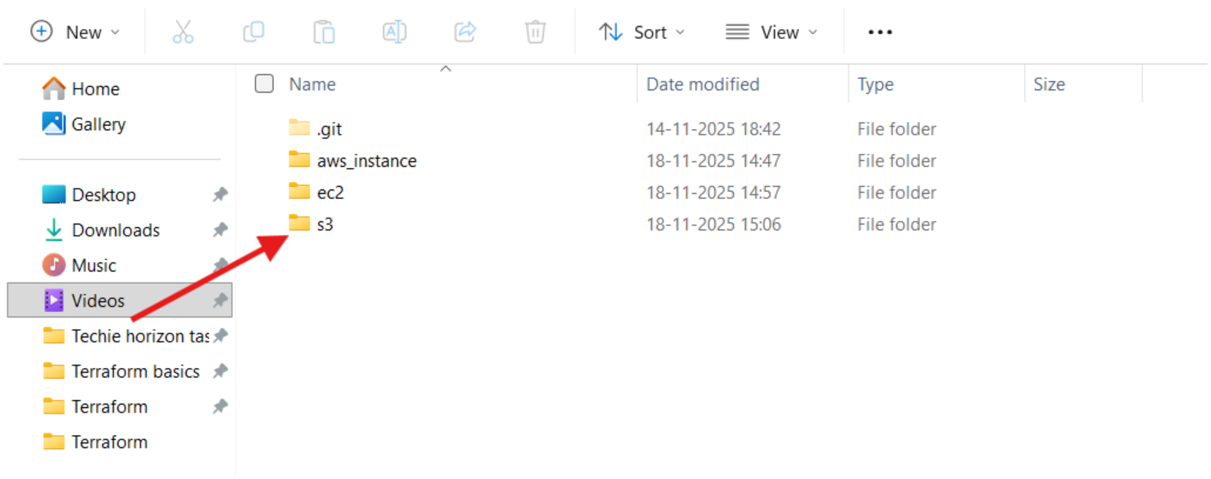


```
variable "bucket_name" {

}
```



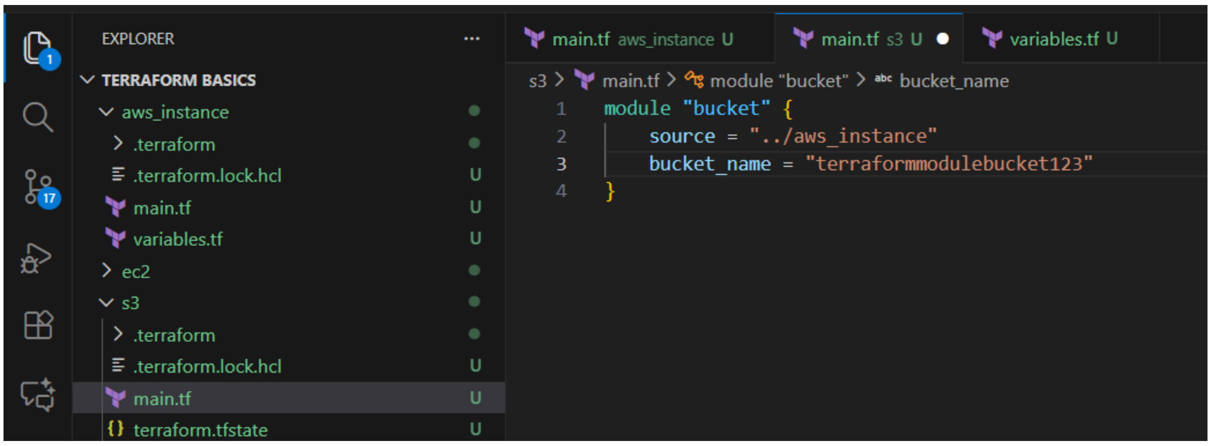
```
PS C:\Users\Ashish\Desktop\Terraform basics\ec2> cd ..
PS C:\Users\Ashish\Desktop\Terraform basics> cd ./aws_instance
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance> terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v6.21.0

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.
```

Create a directory in the terraform folder named as s3.

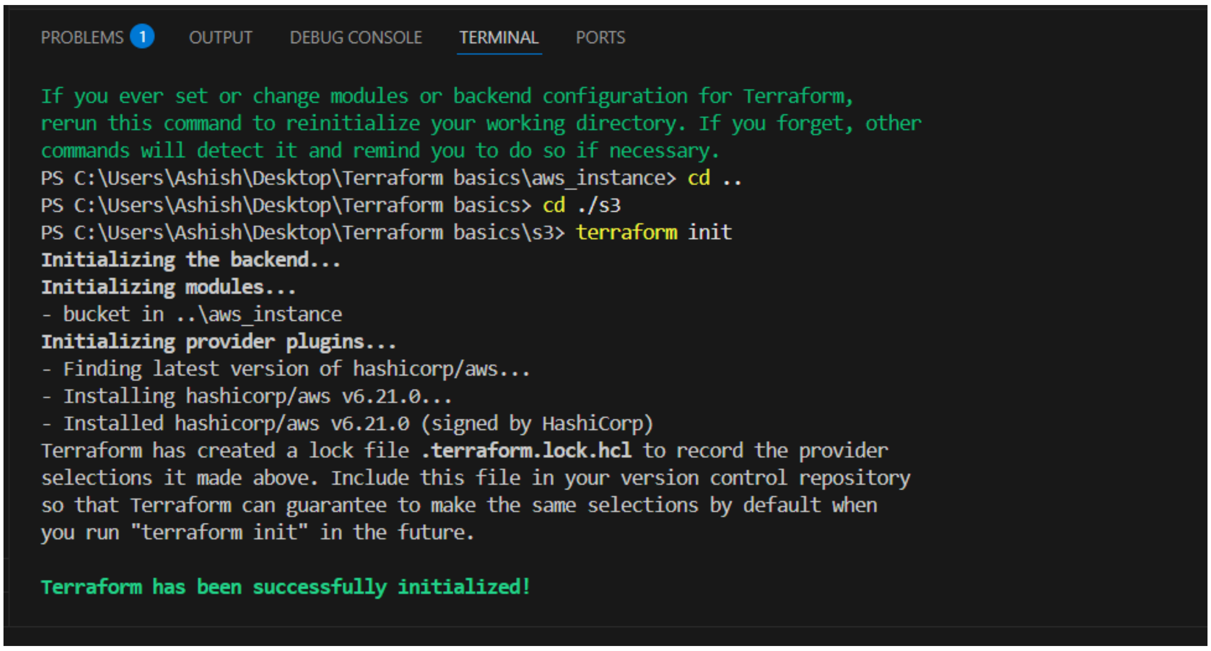


Name	Date modified	Type	Size
.git	14-11-2025 18:42	File folder	
aws_instance	18-11-2025 14:47	File folder	
ec2	18-11-2025 14:57	File folder	
s3	18-11-2025 15:06	File folder	

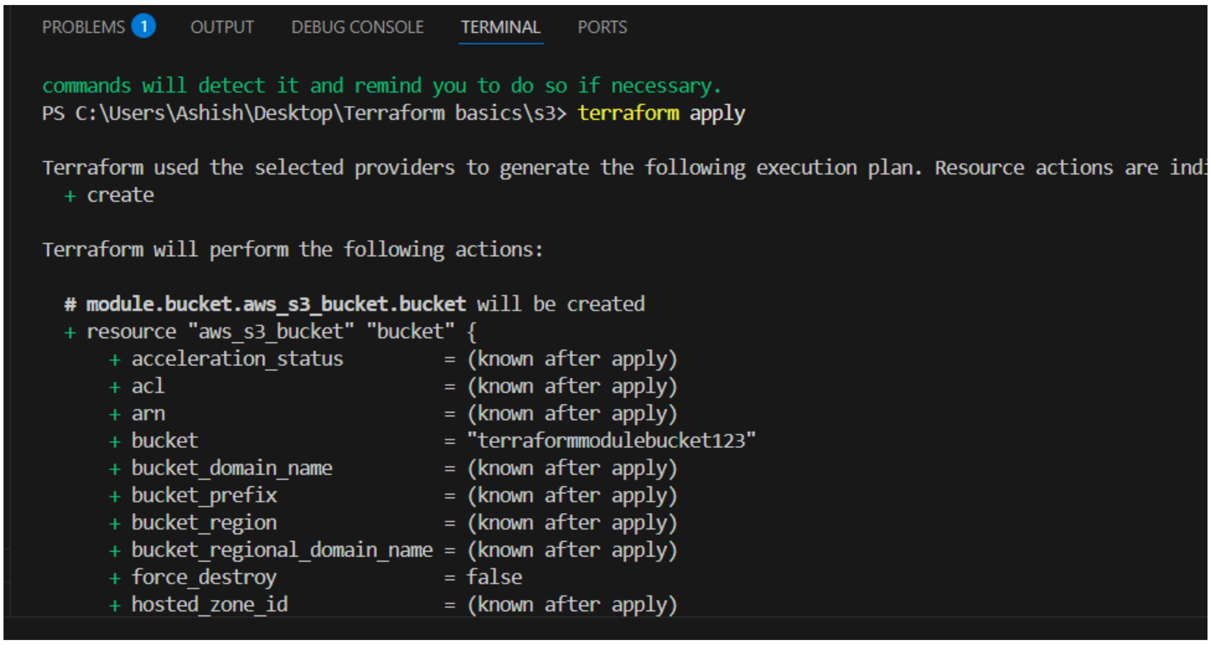
Create a file in the s3 folder as main.tf



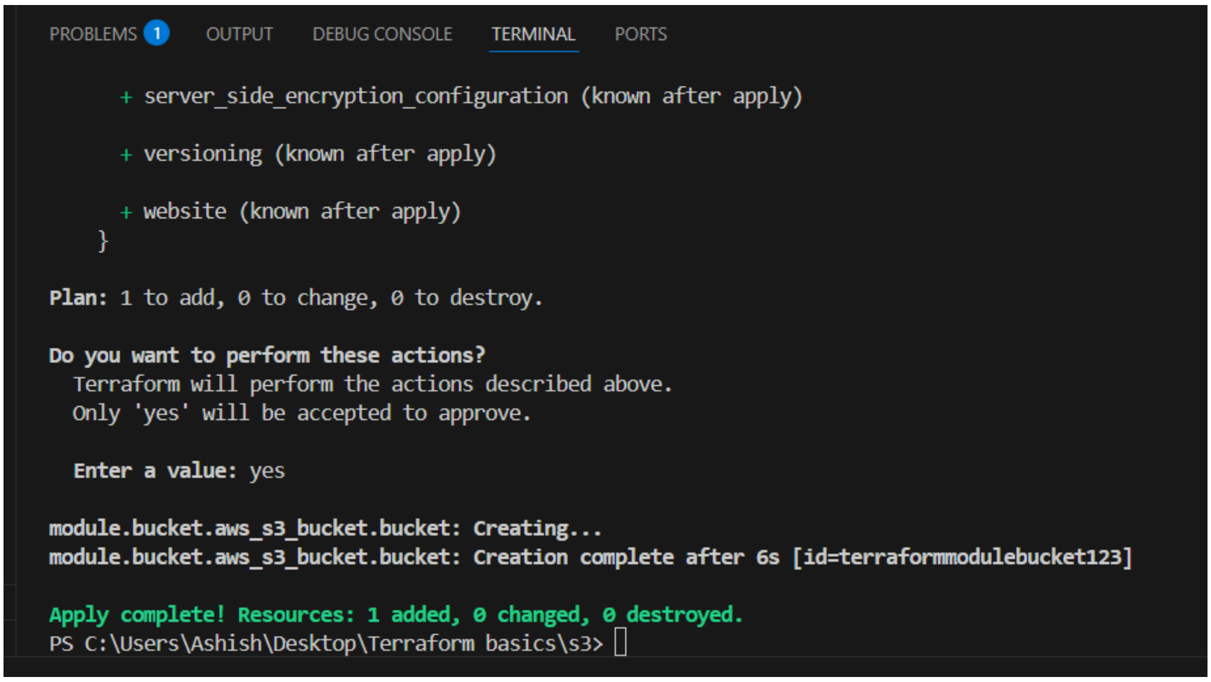
```
module "bucket" {
    source = "../aws_instance"
    bucket_name = "terraformmodulebucket123"
}
```



```
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.
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance> cd ..
PS C:\Users\Ashish\Desktop\Terraform basics> cd ./s3
PS C:\Users\Ashish\Desktop\Terraform basics\s3> terraform init
Initializing the backend...
Initializing modules...
- bucket in ..\aws_instance
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.21.0...
- Installed hashicorp/aws v6.21.0 (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!
```

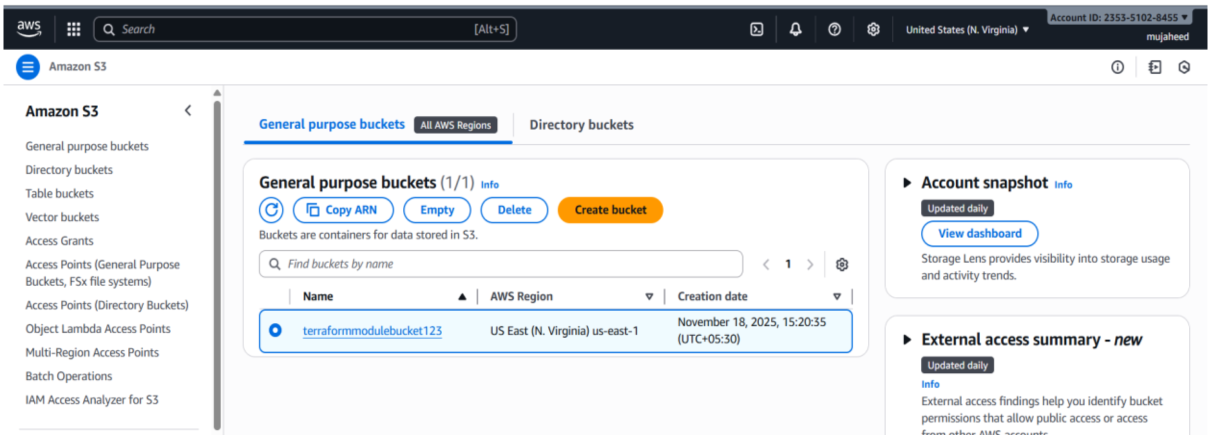


```
commands will detect it and remind you to do so if necessary.
PS C:\Users\Ashish\Desktop\Terraform basics\s3> terraform apply

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

Terraform will perform the following actions:

  # module.bucket.aws_s3_bucket.bucket will be created
  + resource "aws_s3_bucket" "bucket" {
      + acceleration_status         = (known after apply)
      + acl                         = (known after apply)
      + arn                         = (known after apply)
      + bucket                      = "terraformmodulebucket123"
      + bucket_domain_name          = (known after apply)
      + bucket_prefix               = (known after apply)
      + bucket_region               = (known after apply)
      + bucket_regional_domain_name = (known after apply)
      + force_destroy               = false
      + hosted_zone_id              = (known after apply)
```



```
      + server_side_encryption_configuration (known after apply)

      + versioning (known after apply)

      + website (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

module.bucket.aws_s3_bucket.bucket: Creating...
module.bucket.aws_s3_bucket.bucket: Creation complete after 6s [id=terraformmodulebucket123]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\Ashish\Desktop\Terraform basics\s3>
```

An s3 bucket has been created by using terraform modules.



General purpose buckets (1/1) Info

Buckets are containers for data stored in S3.

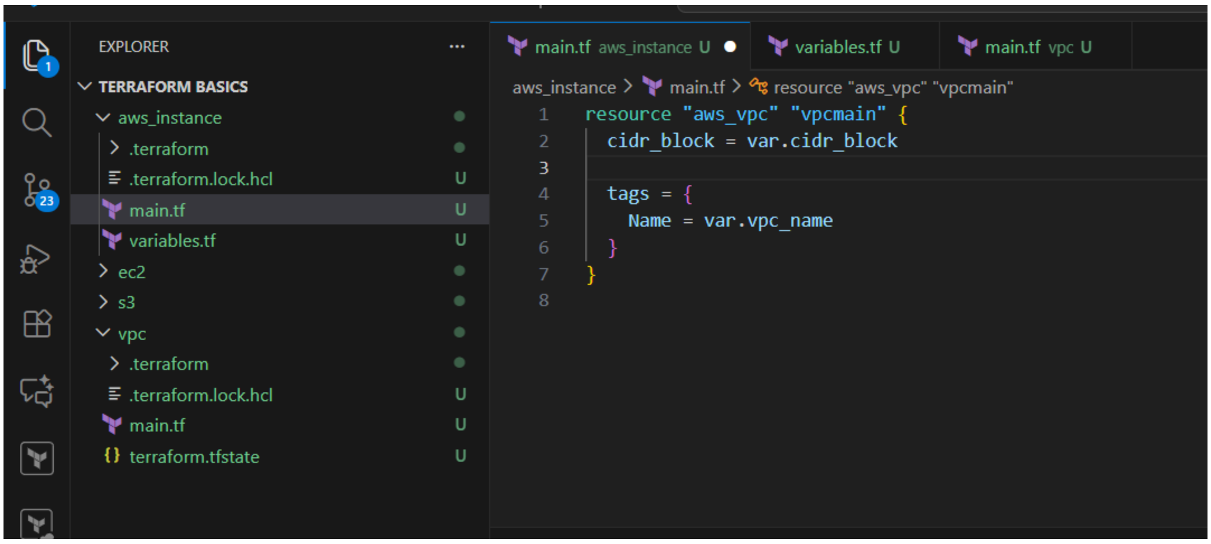
Name	AWS Region	Creation date
terraformmodulebucket123	US East (N. Virginia) us-east-1	November 18, 2025, 15:20:35 (UTC+05:30)

Storage Lens provides visibility into storage usage and activity trends.

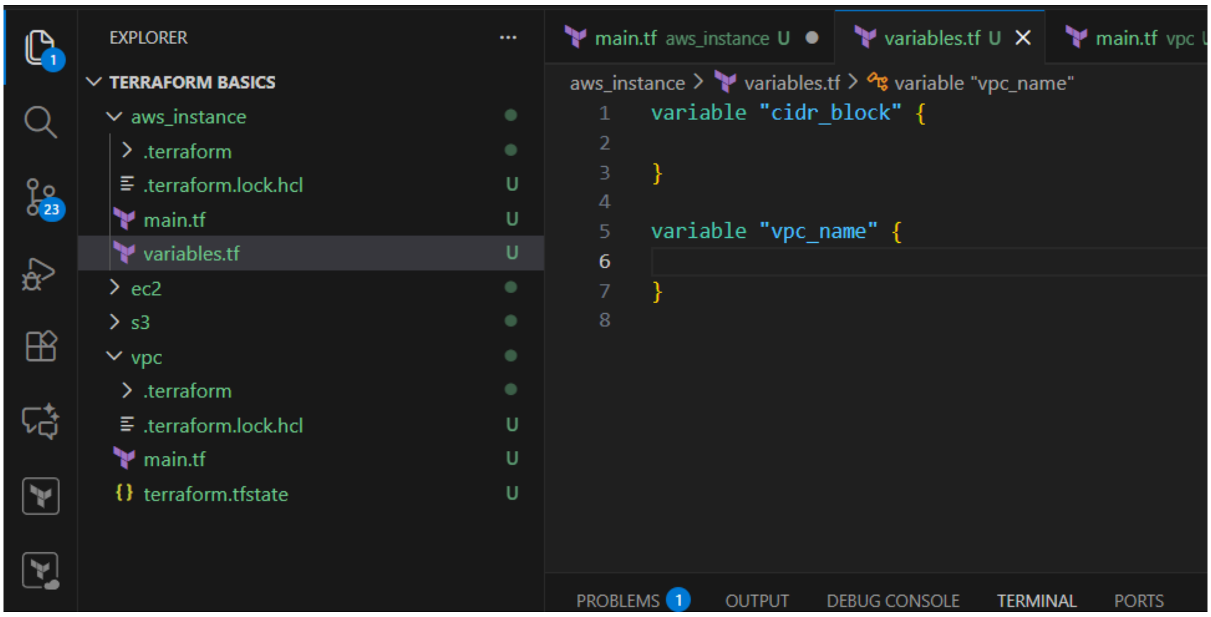
External access findings help you identify bucket permissions that allow public access or access

Vpc:

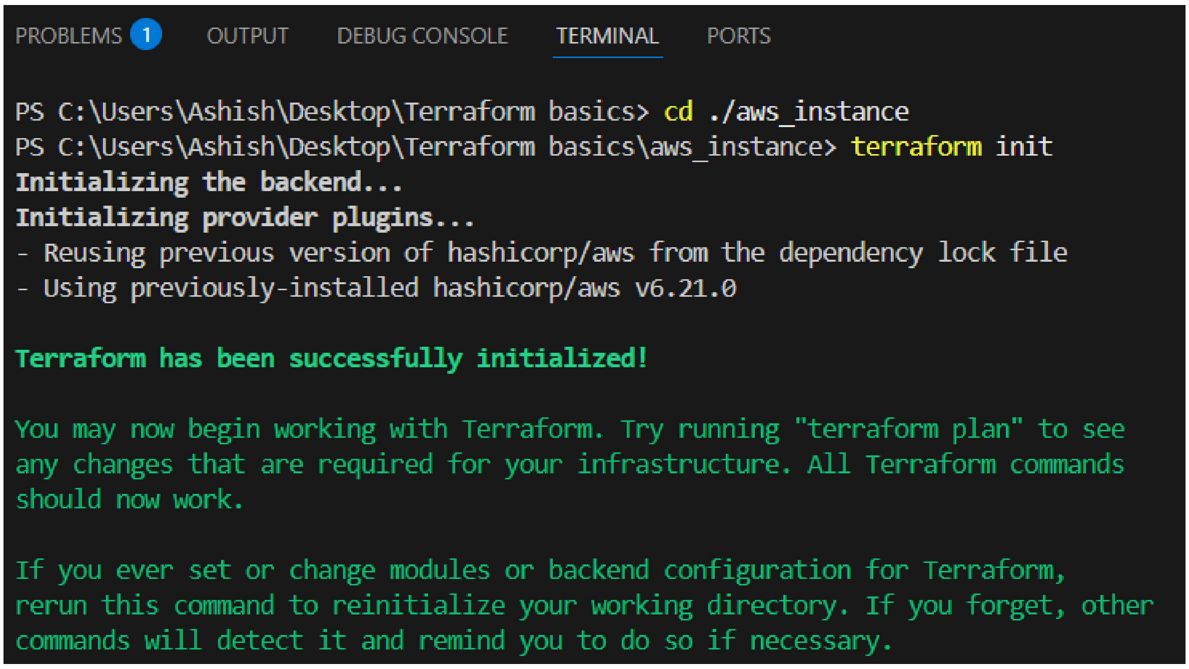
Change the data In main.tf,variables.tf in aws_instance directory.



```
resource "aws_vpc" "vpcmain" {
  cidr_block = var.cidr_block

  tags = {
    Name = var.vpc_name
  }
}
```

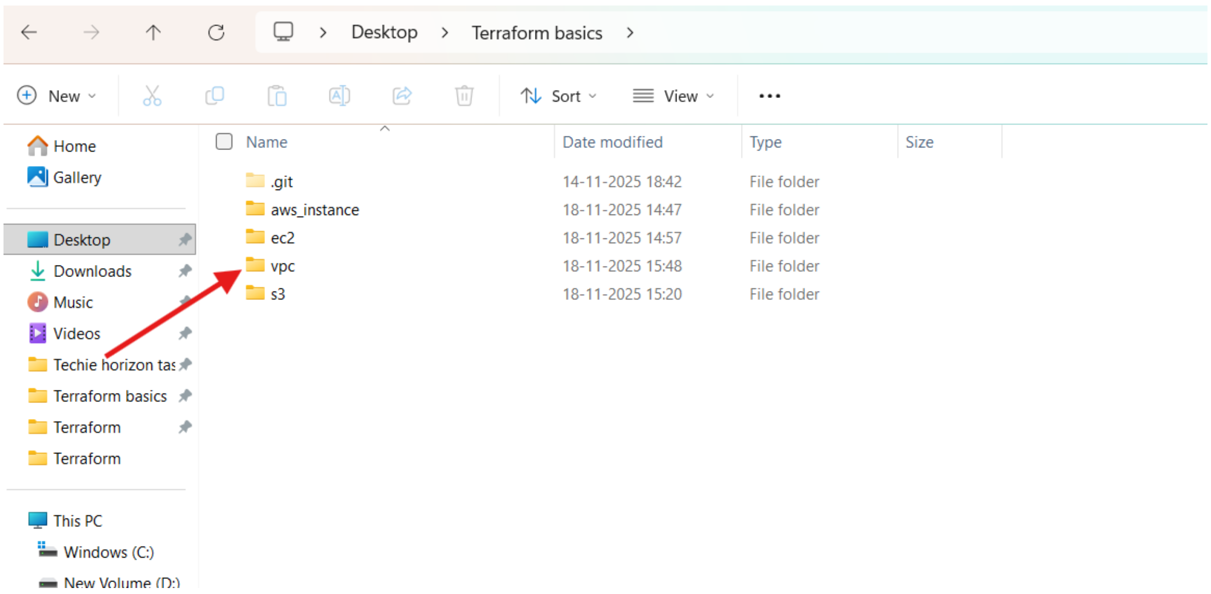


```
variable "cidr_block" {

}

variable "vpc_name" {

}
```



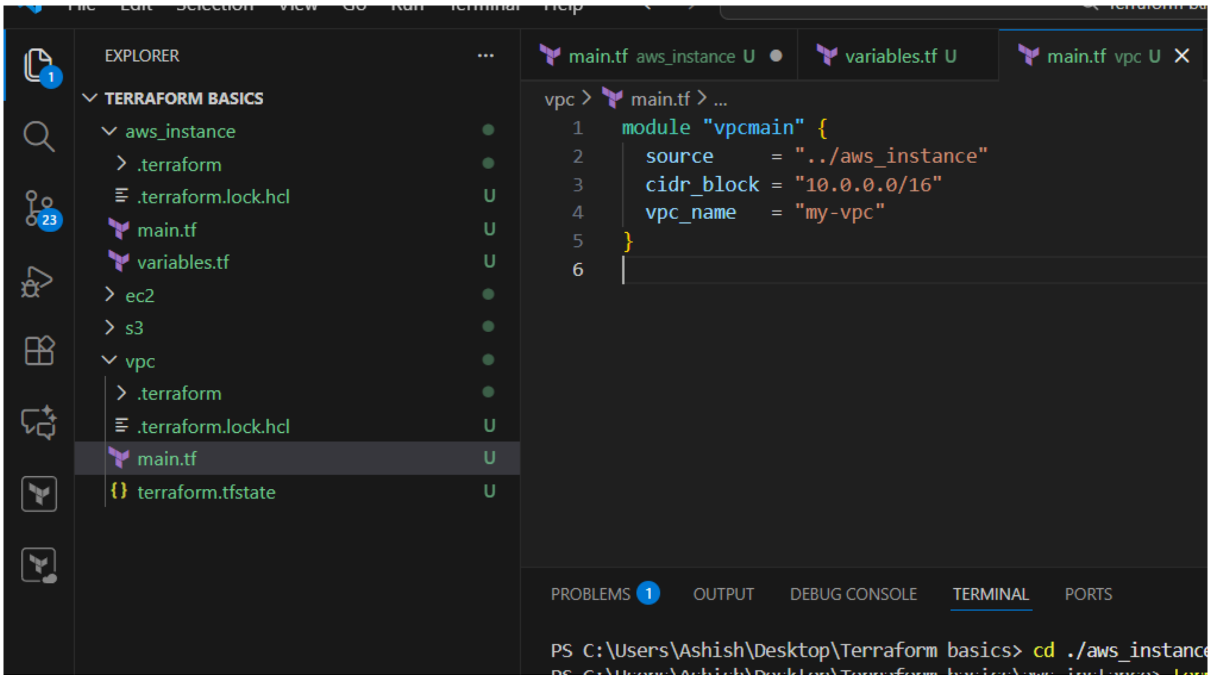
```
PS C:\Users\Ashish\Desktop\Terraform basics> cd ./aws_instance
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance> terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v6.21.0

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.
```

Create a directory named as vpc in the terraform folder.



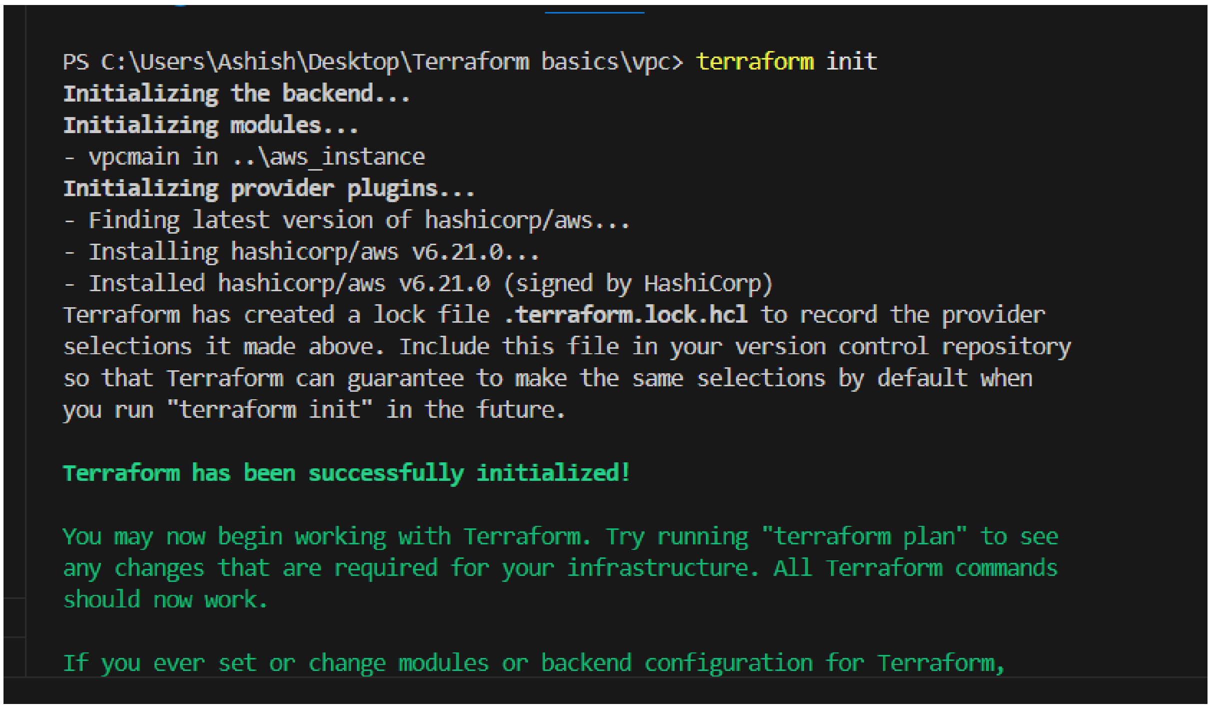
Name	Date modified	Type	Size
.git	14-11-2025 18:42	File folder	
aws_instance	18-11-2025 14:47	File folder	
ec2	18-11-2025 14:57	File folder	
vpc	18-11-2025 15:48	File folder	
s3	18-11-2025 15:20	File folder	

Create a file main.tf in vpc.

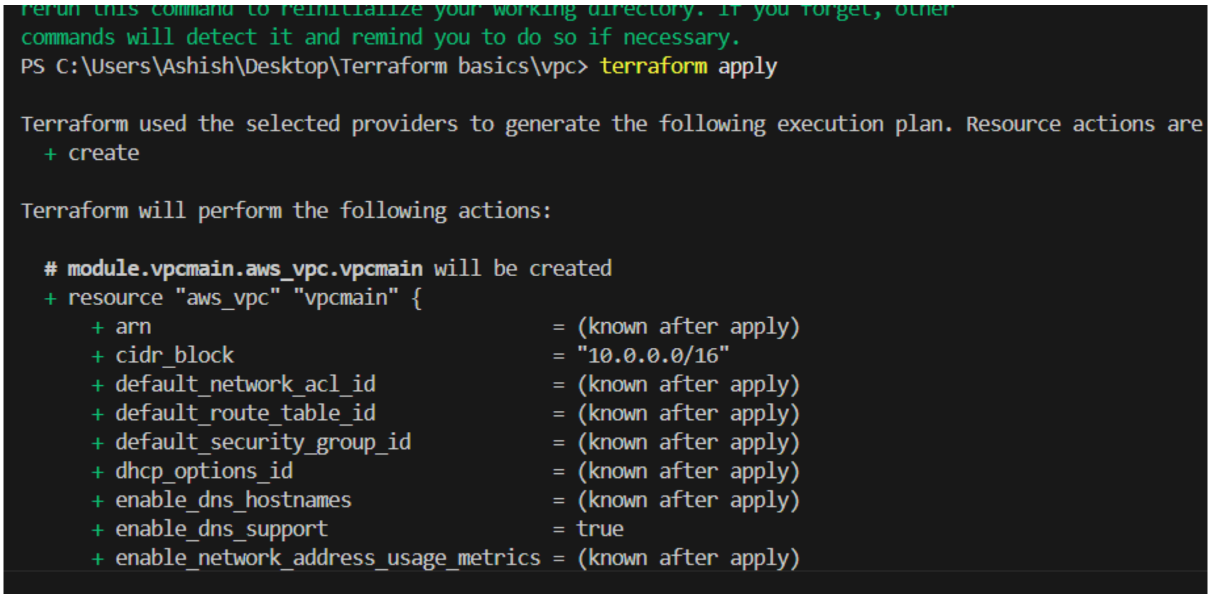


```
module "vpcmain" {
  source     = "../aws_instance"
  cidr_block = "10.0.0.0/16"
  vpc_name   = "my-vpc"
}

```

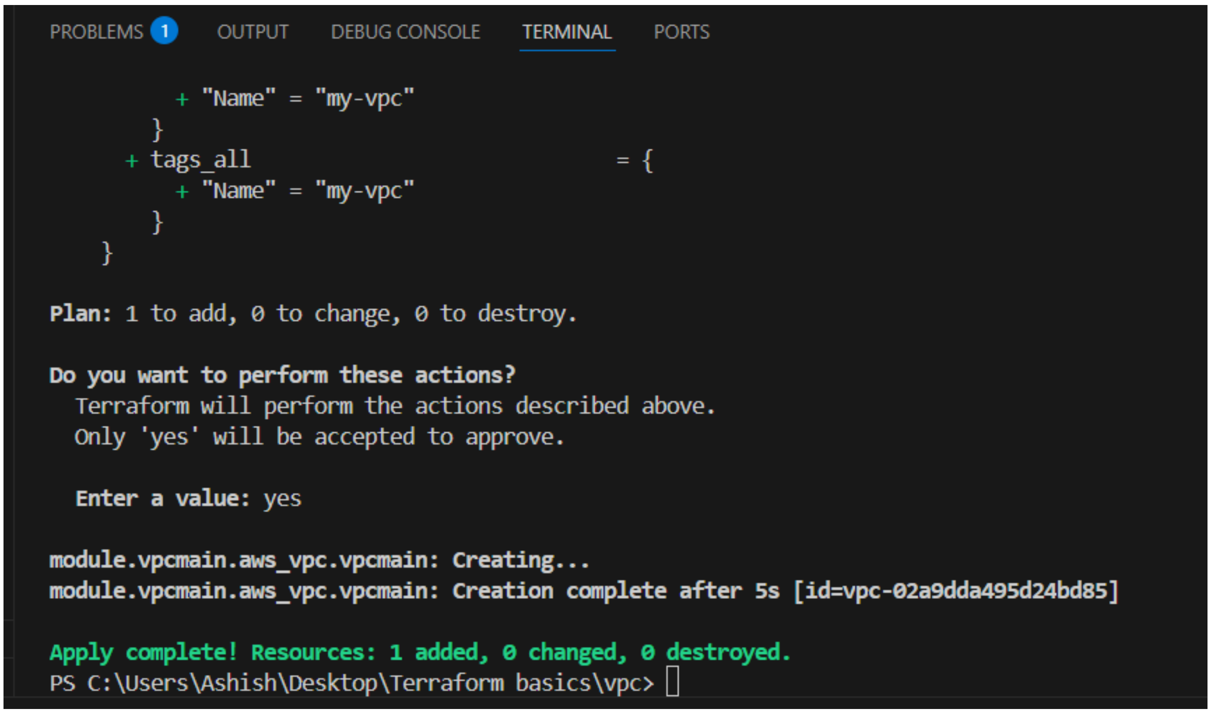
```
PS C:\Users\Ashish\Desktop\Terraform basics> cd ./aws_instance
```



```
PS C:\Users\Ashish\Desktop\Terraform basics\vpc> terraform init
Initializing the backend...
Initializing modules...
- vpcmain in ..\aws_instance
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.21.0...
- Installed hashicorp/aws v6.21.0 (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,
```

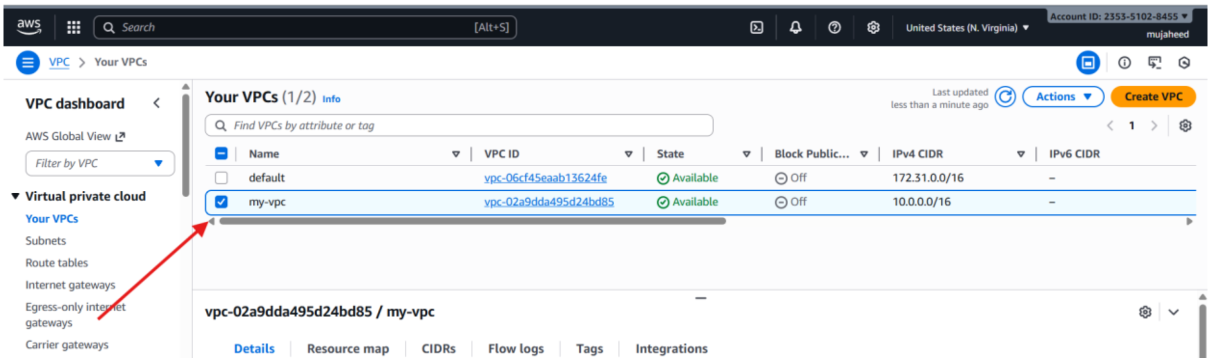


```
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS C:\Users\Ashish\Desktop\Terraform basics\vpc> terraform apply

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

Terraform will perform the following actions:

  # module.vpcmain.aws_vpc.vpcmain will be created
  + resource "aws_vpc" "vpcmain" {
      + arn                                  = (known after apply)
      + cidr_block                           = "10.0.0.0/16"
      + default_network_acl_id               = (known after apply)
      + default_route_table_id               = (known after apply)
      + default_security_group_id            = (known after apply)
      + dhcp_options_id                      = (known after apply)
      + enable_dns_hostnames                 = (known after apply)
      + enable_dns_support                   = true
      + enable_network_address_usage_metrics = (known after apply)
```



```
            + "Name" = "my-vpc"
          }
        + tags_all                         = {
            + "Name" = "my-vpc"
          }
      }

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

module.vpcmain.aws_vpc.vpcmain: Creating...
module.vpcmain.aws_vpc.vpcmain: Creation complete after 5s [id=vpc-02a9dda495d24bd85]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\Ashish\Desktop\Terraform basics\vpc>
```

A vpc has been created by using terraform modules.



Your VPCs (1/2) Info

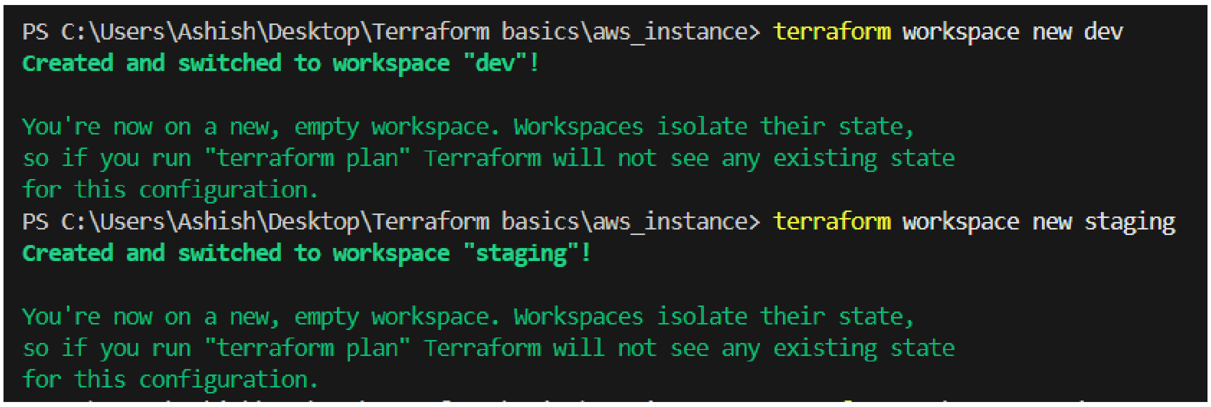
Name	VPC ID	State	Block Public...	IPv4 CIDR	IPv6 CIDR
default	vpc-06cf45eaab13624fe	Available	Off	172.31.0.0/16	–
my-vpc	vpc-02a9dda495d24bd85	Available	Off	10.0.0.0/16	–

vpc-02a9dda495d24bd85 / my-vpc

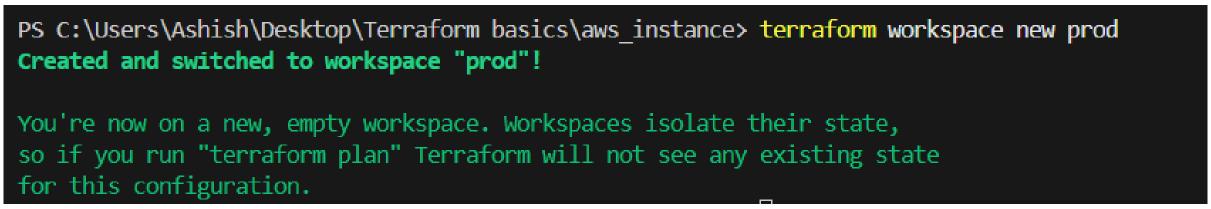
4. Provision EC2 for 3 different environments (Dev, Staging, and Prod) using Terraform workspaces.

Create 3 workspace

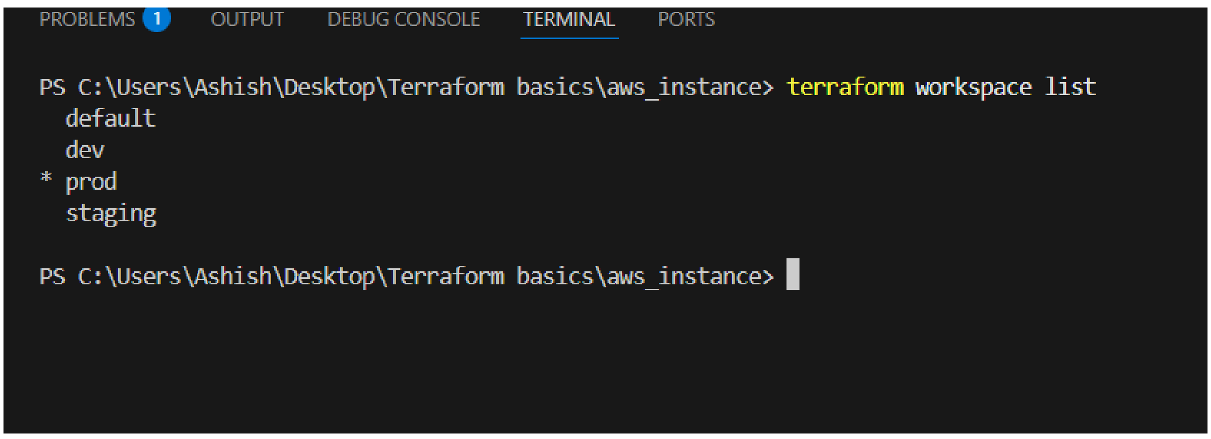
- terraform workspace new dev
- terraform workspace new staging
- terraform workspace new prod



```
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance> terraform workspace new dev
Created and switched to workspace "dev"!

You're now on a new, empty workspace. Workspaces isolate their state,
so if you run "terraform plan" Terraform will not see any existing state
for this configuration.
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance> terraform workspace new staging
Created and switched to workspace "staging"!

You're now on a new, empty workspace. Workspaces isolate their state,
so if you run "terraform plan" Terraform will not see any existing state
for this configuration.
```

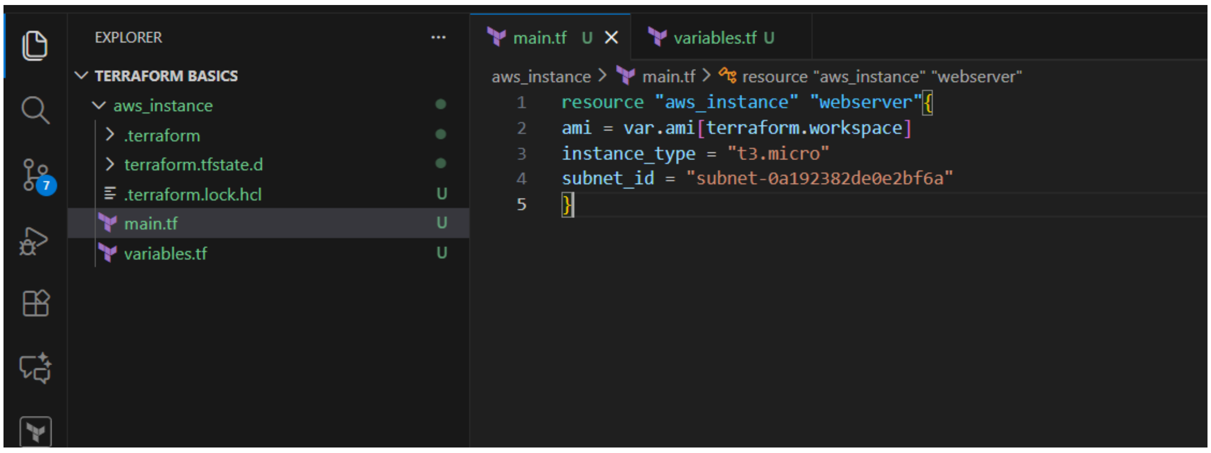


```
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance> terraform workspace new prod
Created and switched to workspace "prod"!

You're now on a new, empty workspace. Workspaces isolate their state,
so if you run "terraform plan" Terraform will not see any existing state
for this configuration.
```

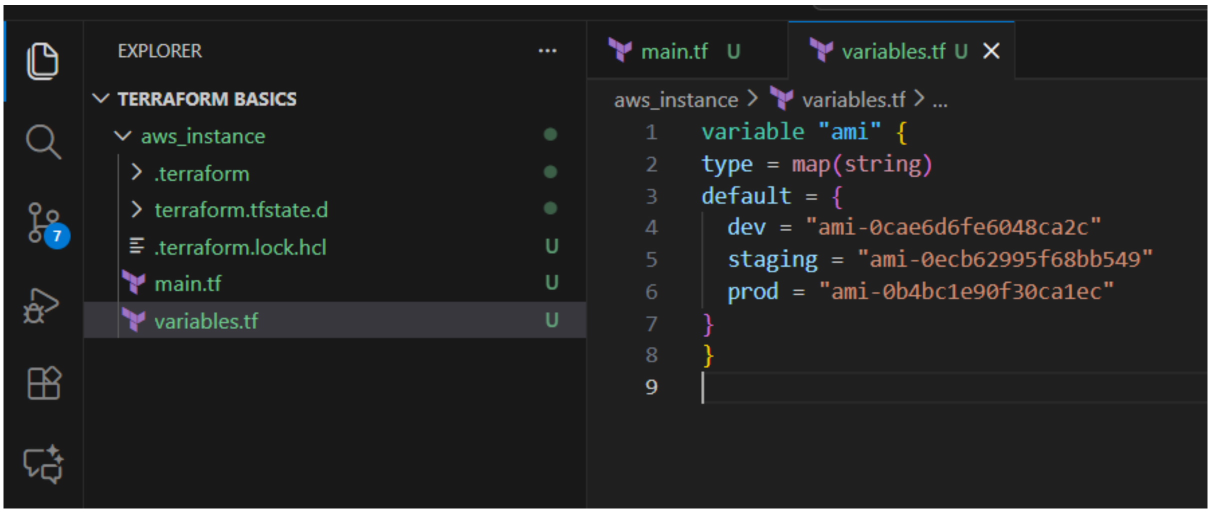


```
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance> terraform workspace list
  default
  dev
* prod
  staging

PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance>
```

Give main.tf, variables.tf as



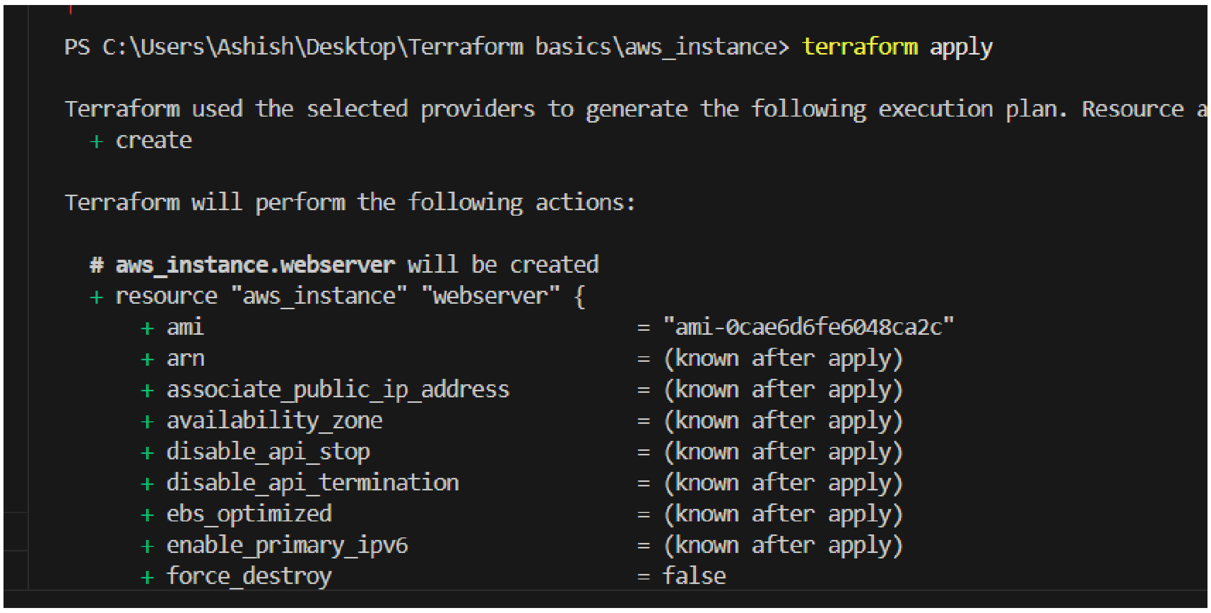
```
resource "aws_instance" "webserver"{
ami = var.ami[terraform.workspace]
instance_type = "t3.micro"
subnet_id = "subnet-0a192382de0e2bf6a"
}
```



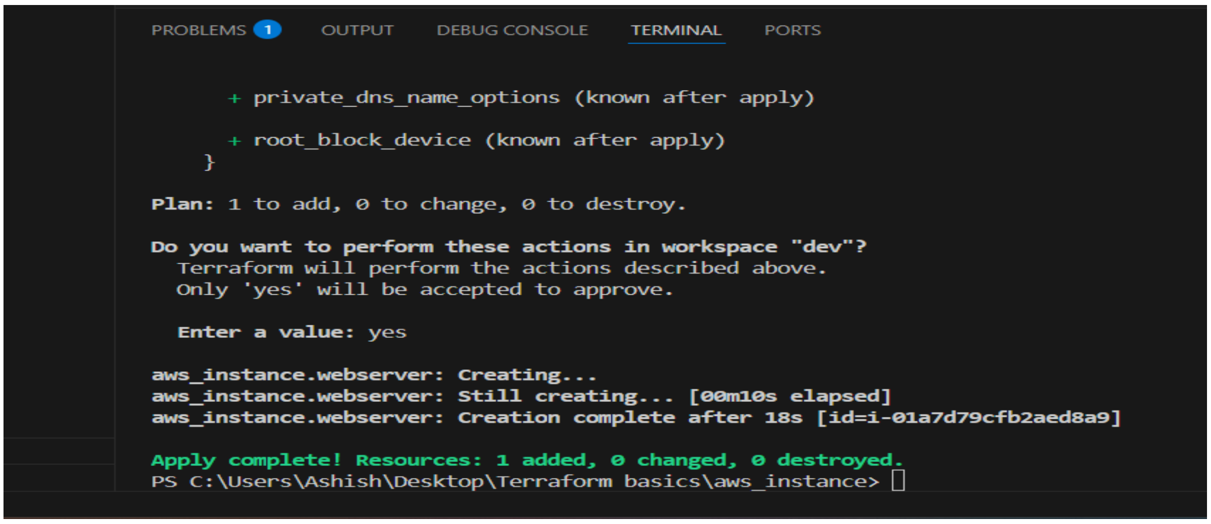
```
variable "ami" {
type = map(string)
default = {
  dev = "ami-0cae6d6fe6048ca2c"
  staging = "ami-0ecb62995f68bb549"
  prod = "ami-0b4bc1e90f30ca1ec"
}
}
```

Select dev as workspace.

- terraform workspace select dev

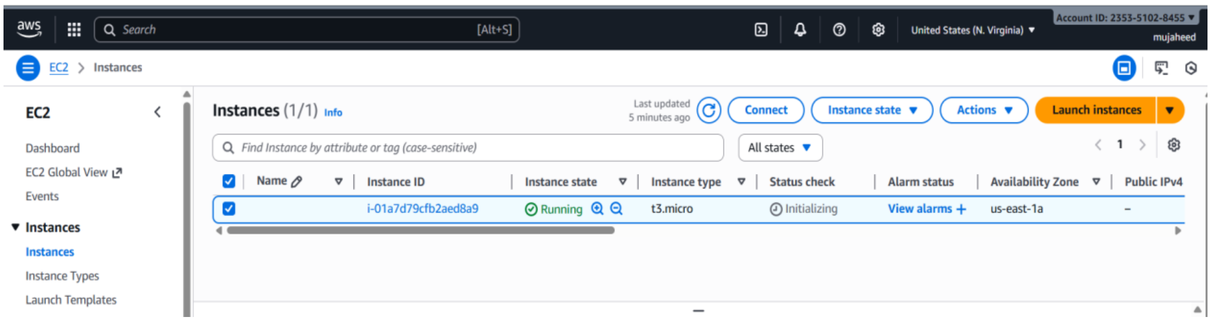


```
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance> terraform apply

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

Terraform will perform the following actions:

  # aws_instance.webserver will be created
  + resource "aws_instance" "webserver" {
      + ami                                  = "ami-0cae6d6fe6048ca2c"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + enable_primary_ipv6                  = (known after apply)
      + force_destroy                        = false
```



```
      + private_dns_name_options (known after apply)

      + root_block_device (known after apply)
    }

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

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

  Enter a value: yes

aws_instance.webserver: Creating...
aws_instance.webserver: Still creating... [00m10s elapsed]
aws_instance.webserver: Creation complete after 18s [id=i-01a7d79cfb2aed8a9]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance> 
```

An instance has been created with dev workspace.

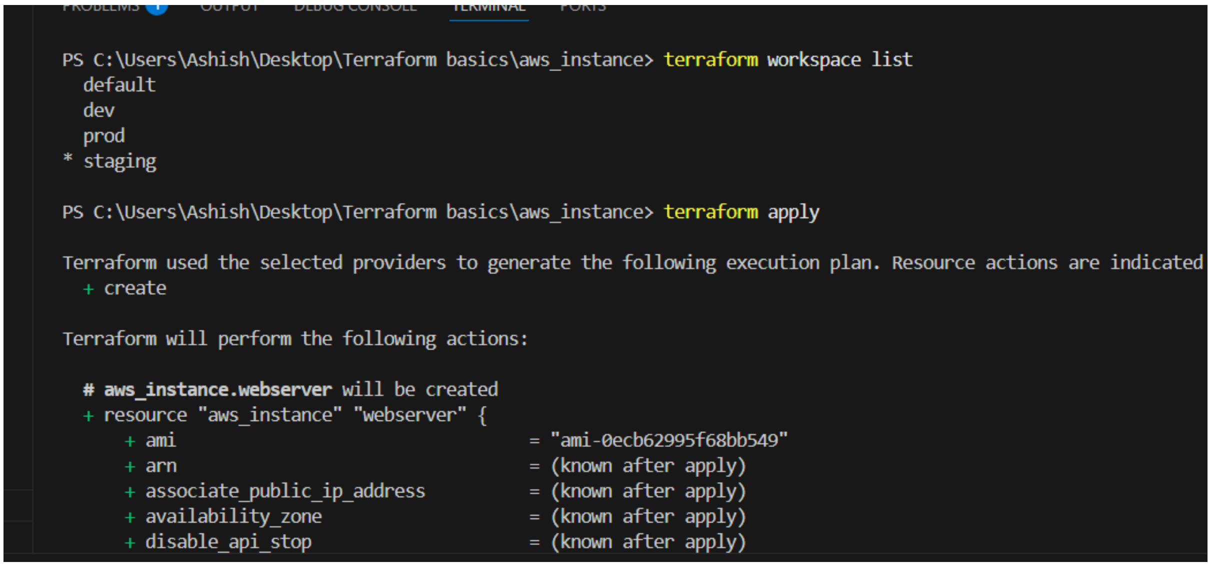


Instances (1/1) Info

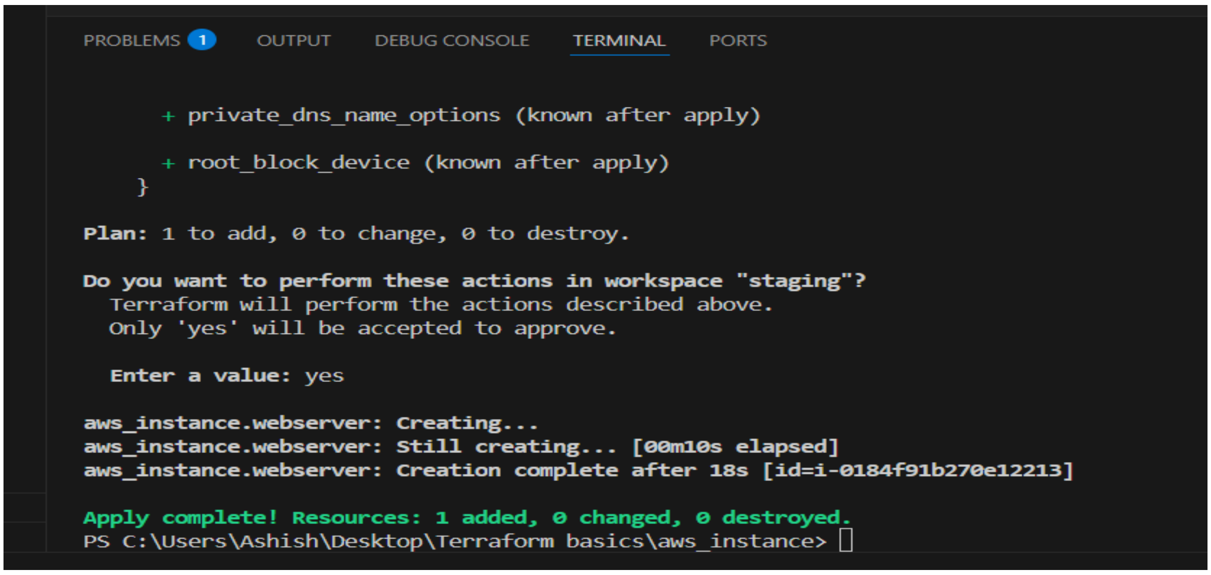
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
	i-01a7d79cfb2aed8a9	Running	t3.micro	Initializing	View alarms +	us-east-1a	–

Shift to staging branch

- terraform workspace select staging
- terraform apply

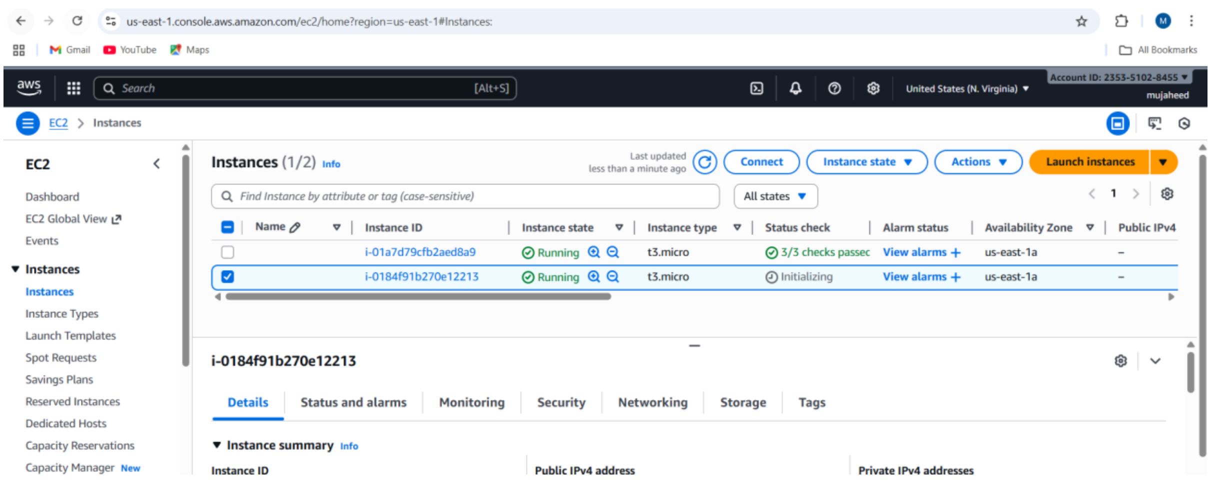


```
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance> terraform workspace list
  default
  dev
  prod
* staging

PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance> terraform apply

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

Terraform will perform the following actions:

  # aws_instance.webserver will be created
  + resource "aws_instance" "webserver" {
      + ami                                  = "ami-0ecb62995f68bb549"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + disable_api_stop                     = (known after apply)
```



```
      + private_dns_name_options (known after apply)

      + root_block_device (known after apply)
    }

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

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

  Enter a value: yes

aws_instance.webserver: Creating...
aws_instance.webserver: Still creating... [00m10s elapsed]
aws_instance.webserver: Creation complete after 18s [id=i-0184f91b270e12213]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance> 
```

An instance has been created for staging workspace.



us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances:

EC2 > Instances

Instances (1/2) Info

Last updated less than a minute ago

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
☐		i-01a7d79cfb2aed8a9	Running	t3.micro	3/3 checks passec	View alarms +	us-east-1a	–
☑		i-0184f91b270e12213	Running	t3.micro	Initializing	View alarms +	us-east-1a	–

i-0184f91b270e12213

Instance summary Info

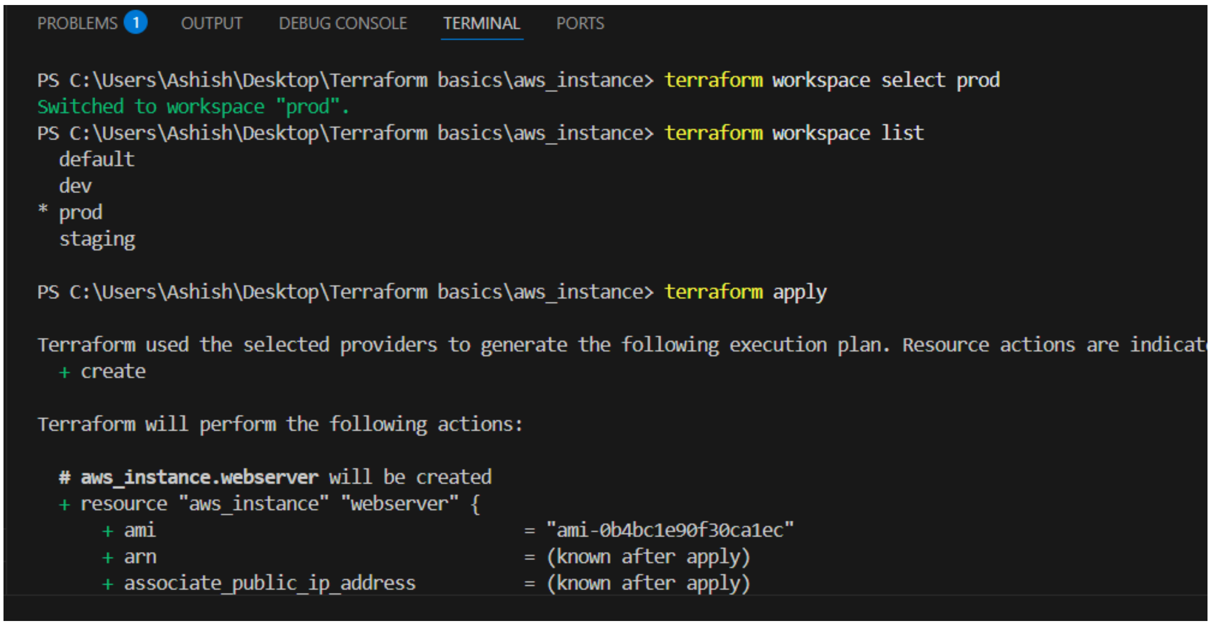
Instance ID

Public IPv4 address

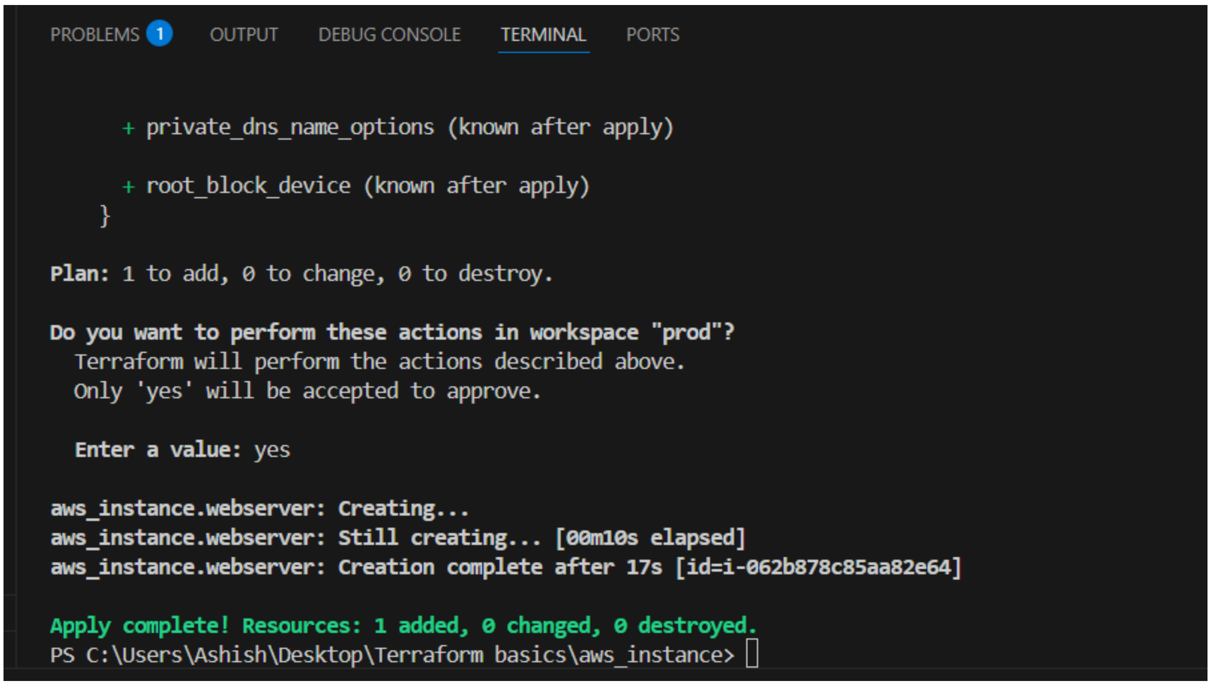
Private IPv4 addresses

Shift to production workspace

- terraform workspace select staging
- terraform apply

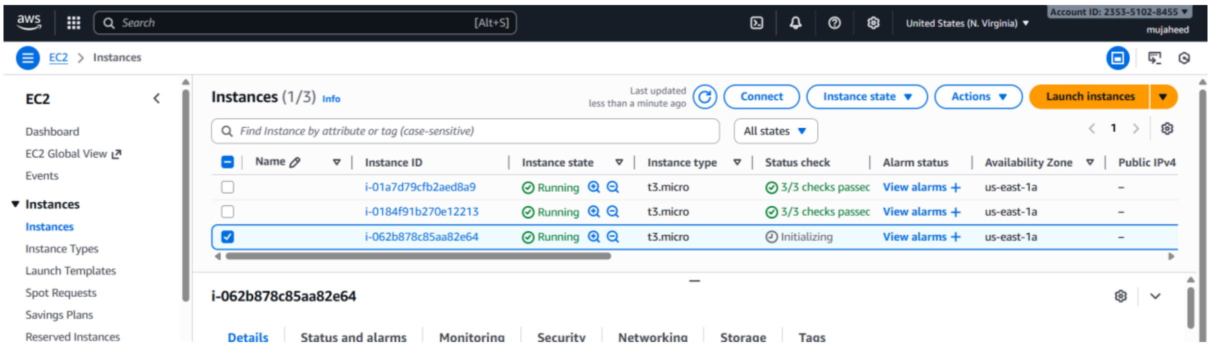


```
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance> terraform workspace select prod
Switched to workspace "prod".
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance> terraform workspace list
  default
  dev
* prod
  staging

PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance> terraform apply

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

Terraform will perform the following actions:

  # aws_instance.webserver will be created
  + resource "aws_instance" "webserver" {
      + ami                                  = "ami-0b4bc1e90f30ca1ec"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
```



```
      + private_dns_name_options (known after apply)

      + root_block_device (known after apply)
    }

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

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

  Enter a value: yes

aws_instance.webserver: Creating...
aws_instance.webserver: Still creating... [00m10s elapsed]
aws_instance.webserver: Creation complete after 17s [id=i-062b878c85aa82e64]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\Ashish\Desktop\Terraform basics\aws_instance>
```

An Instance has been created with prod workspace.



Instances (1/3) Info

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
☐		i-01a7d79cfb2aed8a9	Running	t3.micro	3/3 checks passec	View alarms +	us-east-1a	–
☐		i-0184f91b270e12213	Running	t3.micro	3/3 checks passec	View alarms +	us-east-1a	–
☑		i-062b878c85aa82e64	Running	t3.micro	Initializing	View alarms +	us-east-1a	–

i-062b878c85aa82e64