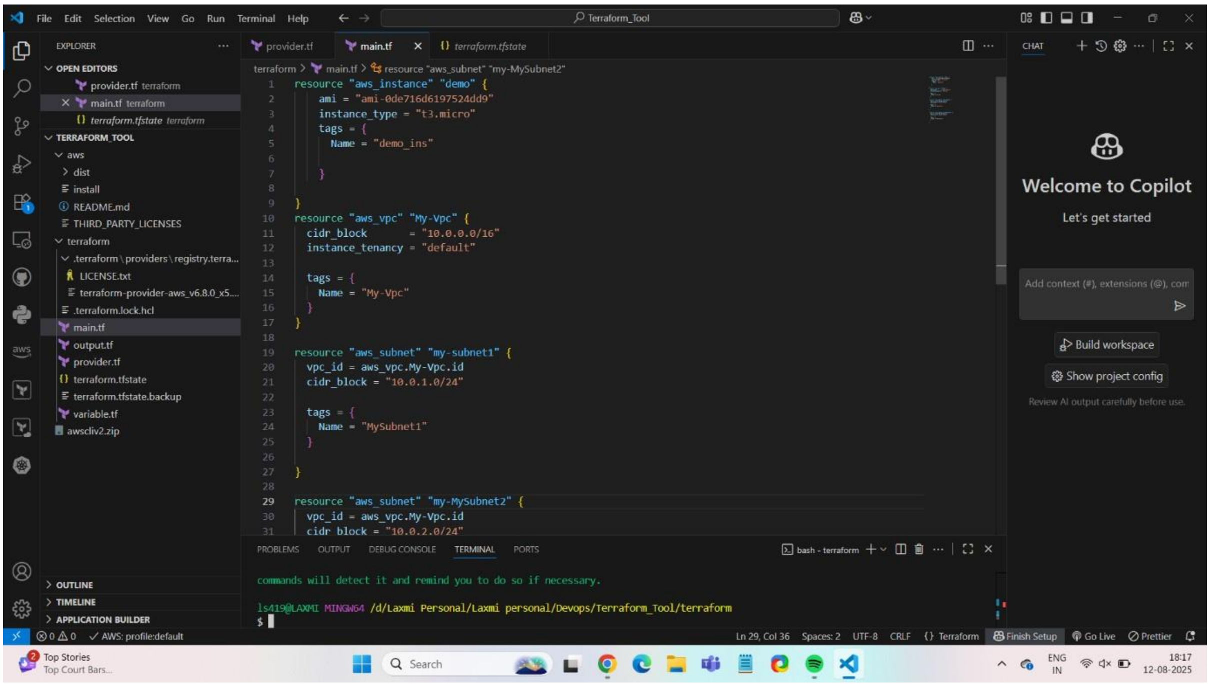
Name :- Laxmi Swami

Topic :- Multiple Resources Created

TASK :- Create one VPC and 2 subnet & one instance in the same main.tf file

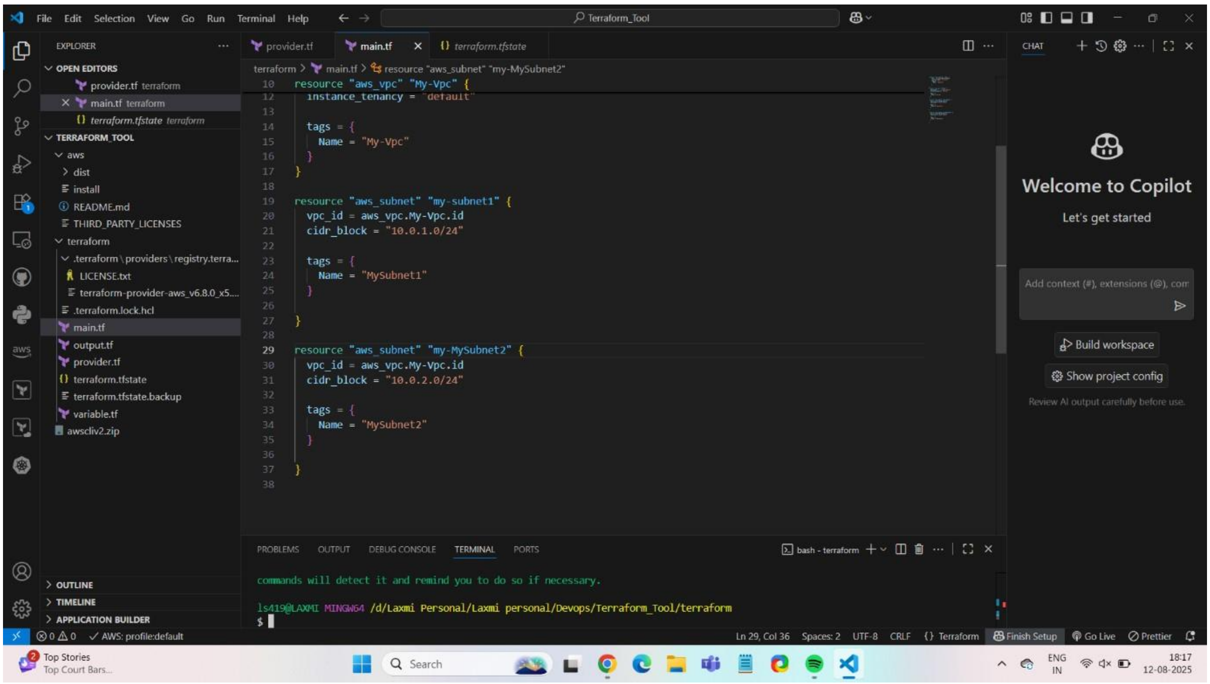
STEP 1 :- Write below script to create vpc and subnet

```
resource “aws_vpc”
“My-Vpc” { cidr_block = “10.0.0.0/16” instance_tenancy = “default” tags
= {
Name = “My-Vpc”
        }
}


resource “aws_vpc” “my-subnet1” {
vpc_id = aws_vpc. My-Vpc.id
cidr_block = “10.0.1.0/24”       tags
= {
      Name = “my-subnet1”
                }
}
resource “aws_vpc” “my-subnet2” {
vpc_id = aws_vpc. My-Vpc.id
cidr_block = “10.0.2.0/24”       tags
= {

      Name = “my-subnet2”
                }
}
```



```
resource "aws_instance" "demo" {
    ami = "ami-0de716d6197524dd9"
    instance_type = "t3.micro"
    tags = {
      Name = "demo_ins"

    }

}
resource "aws_vpc" "My-Vpc" {
  cidr_block       = "10.0.0.0/16"
  instance_tenancy = "default"

  tags = {
    Name = "My-Vpc"
  }
}

resource "aws_subnet" "my-subnet1" {
  vpc_id = aws_vpc.My-Vpc.id
  cidr_block = "10.0.1.0/24"

  tags = {
    Name = "MySubnet1"
  }

}

resource "aws_subnet" "my-MySubnet2" {
  vpc_id = aws_vpc.My-Vpc.id
  cidr_block = "10.0.2.0/24"
```

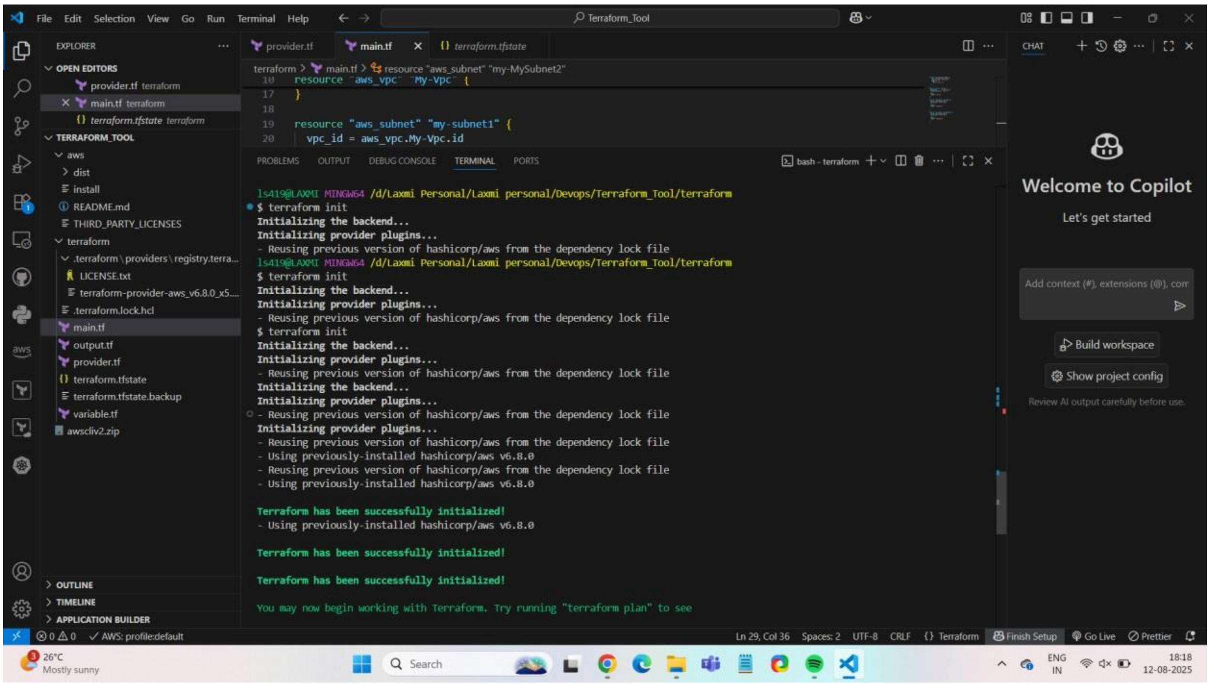
```
commands will detect it and remind you to do so if necessary.

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$
```



```
resource "aws_vpc" "My-Vpc" {
  instance_tenancy = "default"

  tags = {
    Name = "My-Vpc"
  }
}

resource "aws_subnet" "my-subnet1" {
  vpc_id = aws_vpc.My-Vpc.id
  cidr_block = "10.0.1.0/24"

  tags = {
    Name = "MySubnet1"
  }

}

resource "aws_subnet" "my-MySubnet2" {
  vpc_id = aws_vpc.My-Vpc.id
  cidr_block = "10.0.2.0/24"

  tags = {
    Name = "MySubnet2"
  }

}

```

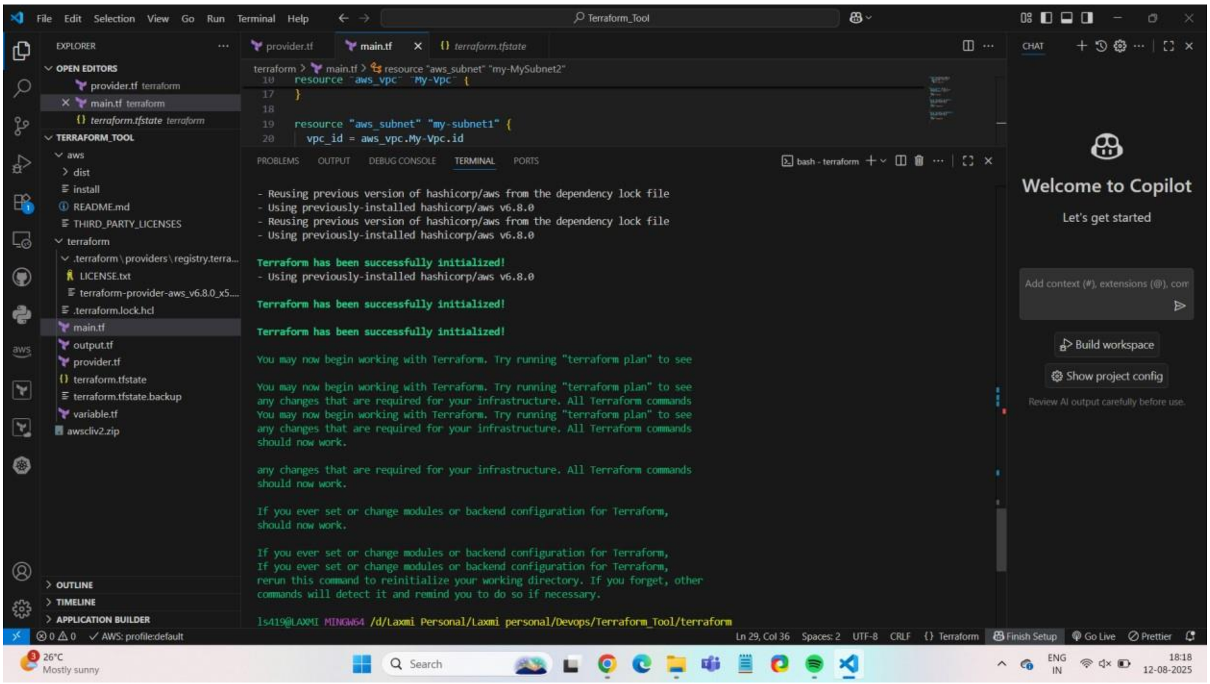
```
commands will detect it and remind you to do so if necessary.

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$
```

STEP 2 :- Now fetch **terraform init** command.



```
resource "aws_vpc" "My-Vpc" {
}

resource "aws_subnet" "my-subnet1" {
  vpc_id = aws_vpc.My-Vpc.id
```

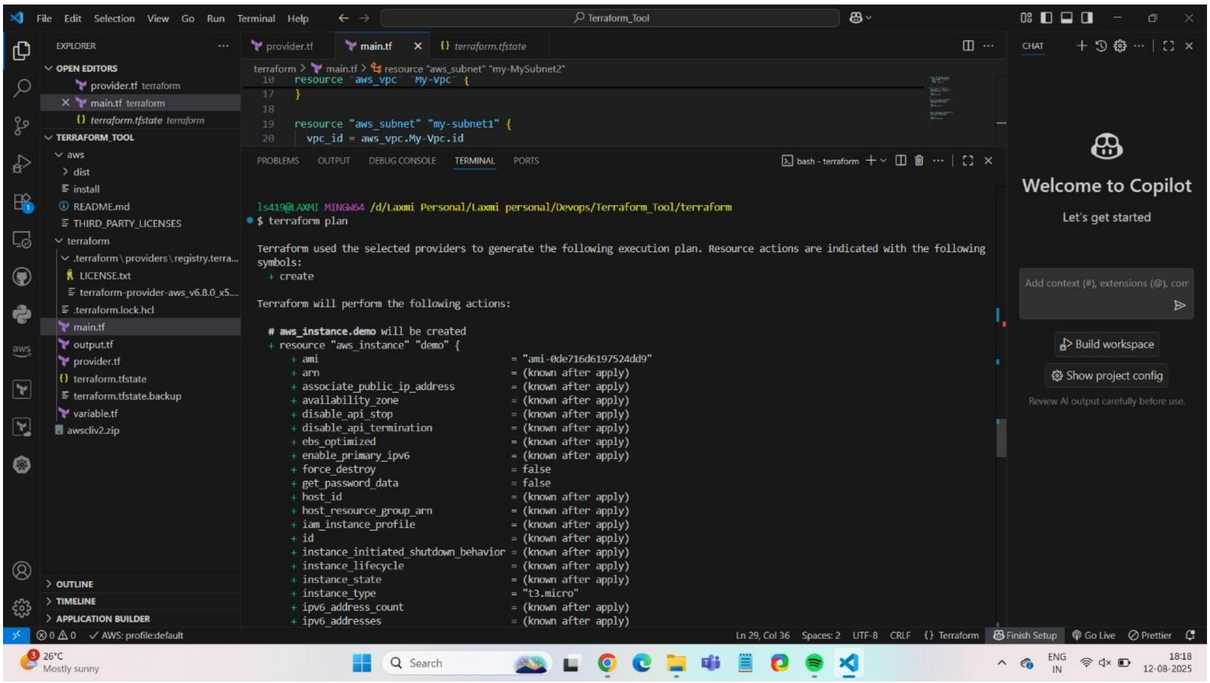
```
ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$ terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$ terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
$ terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v6.8.0
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v6.8.0

Terraform has been successfully initialized!
- Using previously-installed hashicorp/aws v6.8.0

Terraform has been successfully initialized!

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
```



```
resource "aws_vpc" "My-Vpc" {
}

resource "aws_subnet" "my-subnet1" {
  vpc_id = aws_vpc.My-Vpc.id
```

```
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v6.8.0
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v6.8.0

Terraform has been successfully initialized!
- Using previously-installed hashicorp/aws v6.8.0

Terraform has been successfully initialized!

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see

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

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,
should now work.

If you ever set or change modules or backend configuration for Terraform,
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.

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
```

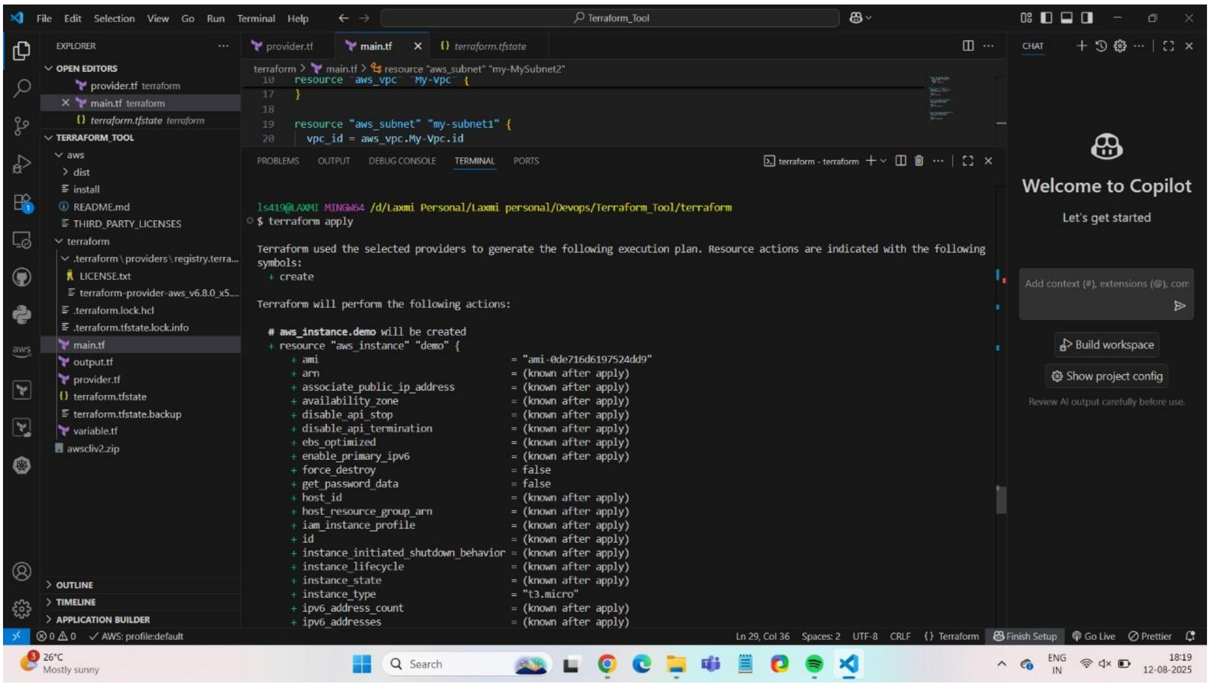
STEP 3 :- Now fetch command **terraform plan.**



```
resource "aws_vpc" "My-Vpc" {
}

resource "aws_subnet" "my-subnet1" {
  vpc_id = aws_vpc.My-Vpc.id
```

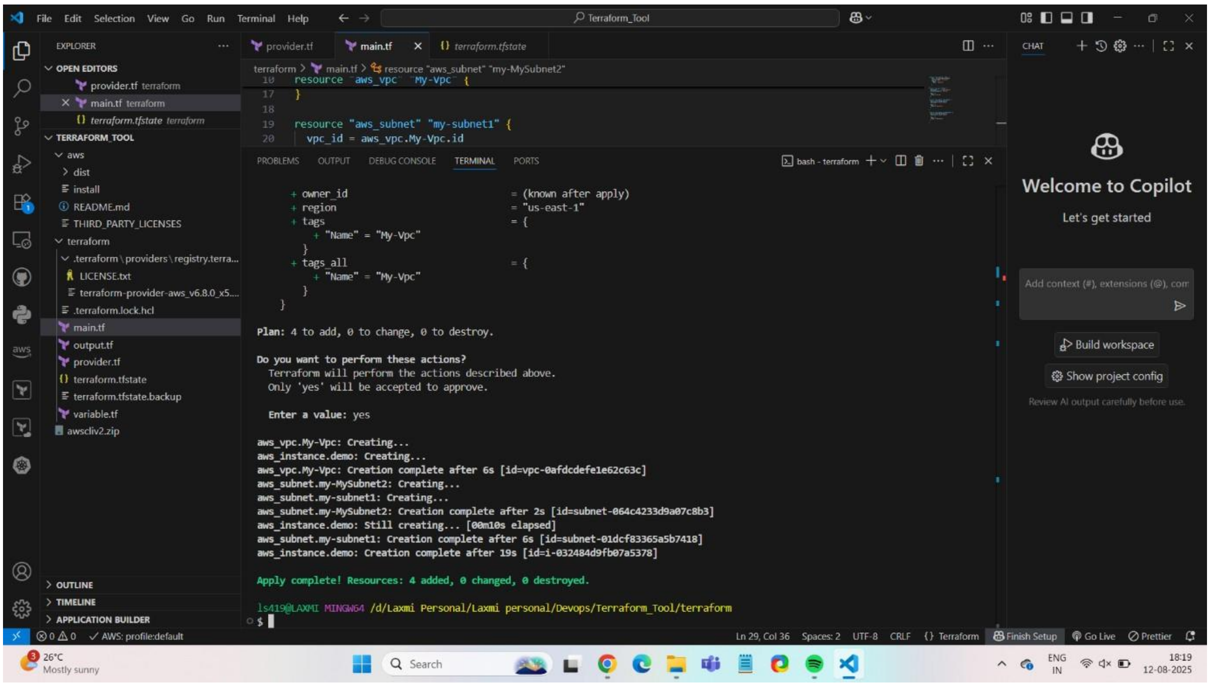
```
ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$ terraform plan

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.demo will be created
  + resource "aws_instance" "demo" {
      + ami                                  = "ami-0de716d6197524dd9"
      + 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)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t3.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
```

STEP 4 :- Now fetch command **terraform apply** and chech vpc instance and 2 subnet created or not.



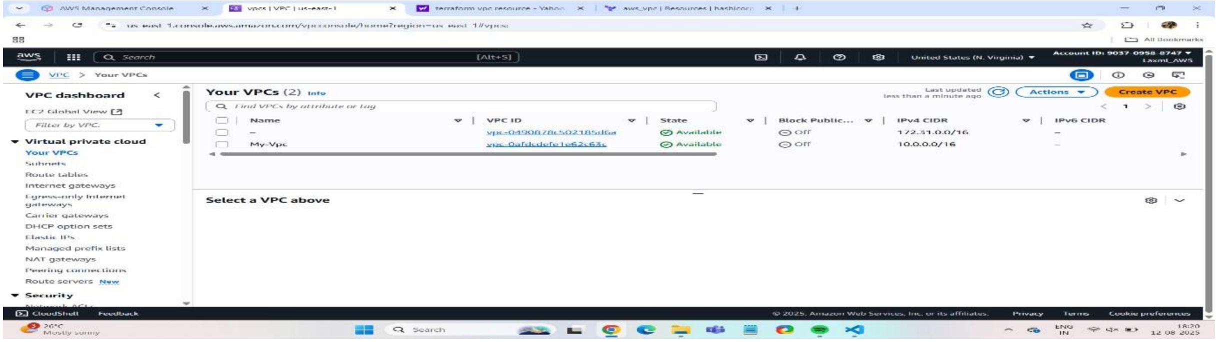
```
resource "aws_vpc" "My-Vpc" {
}

resource "aws_subnet" "my-subnet1" {
  vpc_id = aws_vpc.My-Vpc.id
```

```
ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$ 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.demo will be created
  + resource "aws_instance" "demo" {
      + ami                                  = "ami-0de716d6197524dd9"
      + 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)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t3.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
```



```
resource "aws_vpc" "My-Vpc" {
}

resource "aws_subnet" "my-subnet1" {
  vpc_id = aws_vpc.My-Vpc.id
```

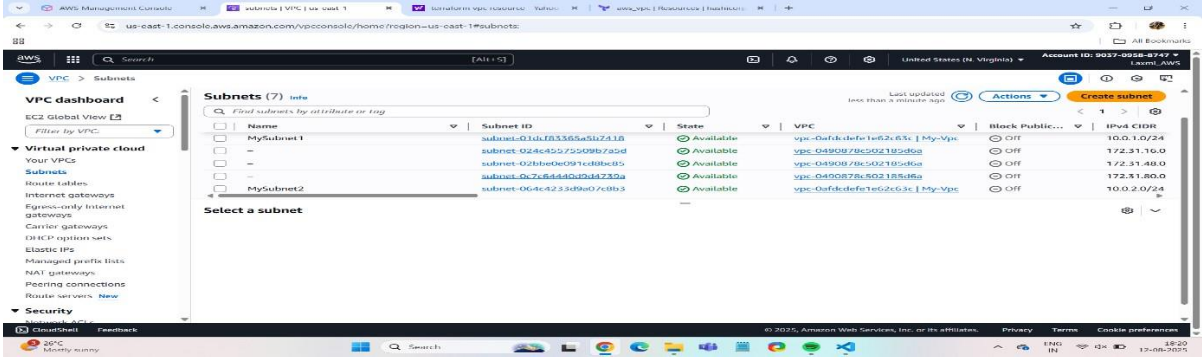
```
      + owner_id                             = (known after apply)
      + region                               = "us-east-1"
      + tags                                 = {
          + "Name" = "My-Vpc"
        }
      + tags_all                             = {
          + "Name" = "My-Vpc"
        }
    }

Plan: 4 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_vpc.My-Vpc: Creating...
aws_instance.demo: Creating...
aws_vpc.My-Vpc: Creation complete after 6s [id=vpc-0afdcdefe1e62c63c]
aws_subnet.my-MySubnet2: Creating...
aws_subnet.my-subnet1: Creating...
aws_subnet.my-MySubnet2: Creation complete after 2s [id=subnet-064c4233d9a07c8b3]
aws_instance.demo: Still creating... [00m10s elapsed]
aws_subnet.my-subnet1: Creation complete after 6s [id=subnet-01dcf83365a5b7418]
aws_instance.demo: Creation complete after 19s [id=i-032484d9fb07a5378]

Apply complete! Resources: 4 added, 0 changed, 0 destroyed.

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$
```


vpcs | VPC | us-east-1
terraform vpc resource - Yahoo
aws_vpc | Resources | hashicorp
us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#vpcs:
United States (N. Virginia)
Account ID: 9037-0958-8747
Laxmi_AWS
VPC > Your VPCs
VPC dashboard
EC2 Global View
Filter by VPC
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
Peering connections
Route servers New
Security
Actions
Create VPC
Last updated less than a minute ago
CloudShell Feedback
© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Your VPCs (2) Info

Name	VPC ID	State	Block Public...	IPv4 CIDR	IPv6 CIDR
–	vpc-0490878c502185d6a	Available	Off	172.31.0.0/16	–
My-Vpc	vpc-0afdcdefe1e62c63c	Available	Off	10.0.0.0/16	–

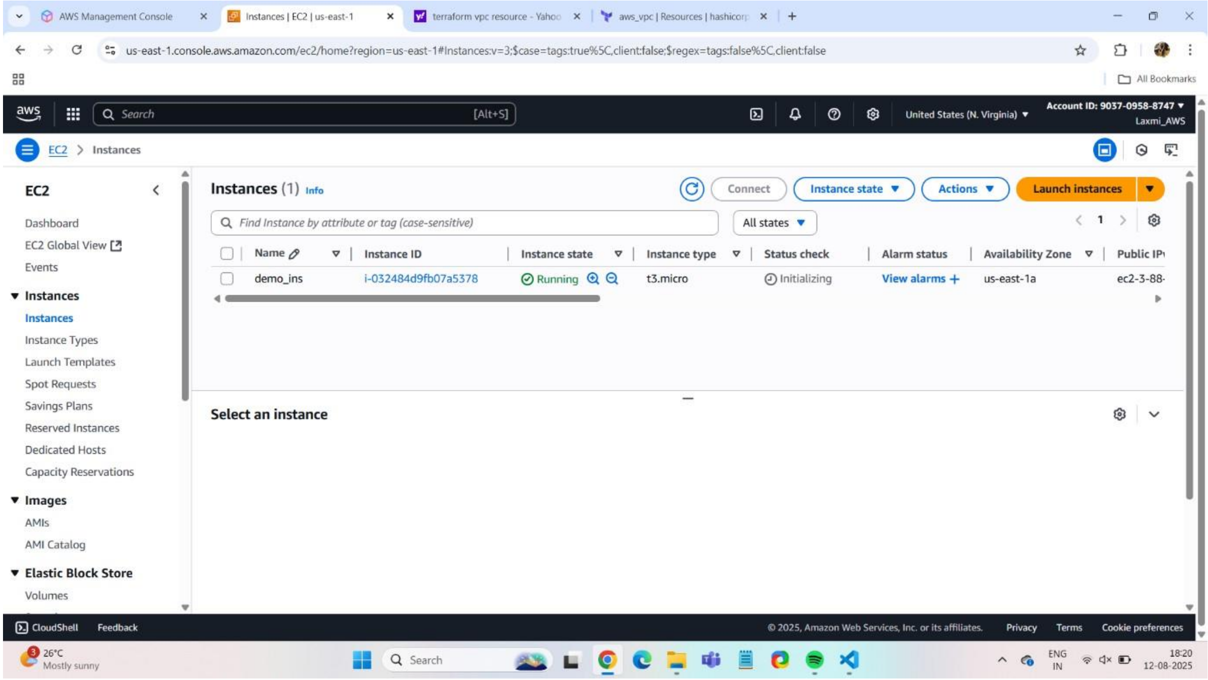
Select a VPC above


subnets | VPC | us-east-1
terraform vpc resource - Yahoo
aws_vpc | Resources | hashicorp
us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#subnets:
United States (N. Virginia)
Account ID: 9037-0958-8747
Laxmi_AWS
VPC > Subnets
VPC dashboard
EC2 Global View
Filter by VPC
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
Peering connections
Route servers New
Security
Actions
Create subnet
Last updated less than a minute ago
CloudShell Feedback
© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Subnets (7) Info

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
MySubnet1	subnet-01dcf83365a5b7418	Available	vpc-0afdcdefe1e62c63c | My-Vpc	Off	10.0.1.0/24
–	subnet-024c45575509b7a5d	Available	vpc-0490878c502185d6a	Off	172.31.16.0
–	subnet-02bbe0e091cd8bc85	Available	vpc-0490878c502185d6a	Off	172.31.48.0
–	subnet-0c7c64440d9d4739a	Available	vpc-0490878c502185d6a	Off	172.31.80.0
MySubnet2	subnet-064c4233d9a07c8b3	Available	vpc-0afdcdefe1e62c63c | My-Vpc	Off	10.0.2.0/24

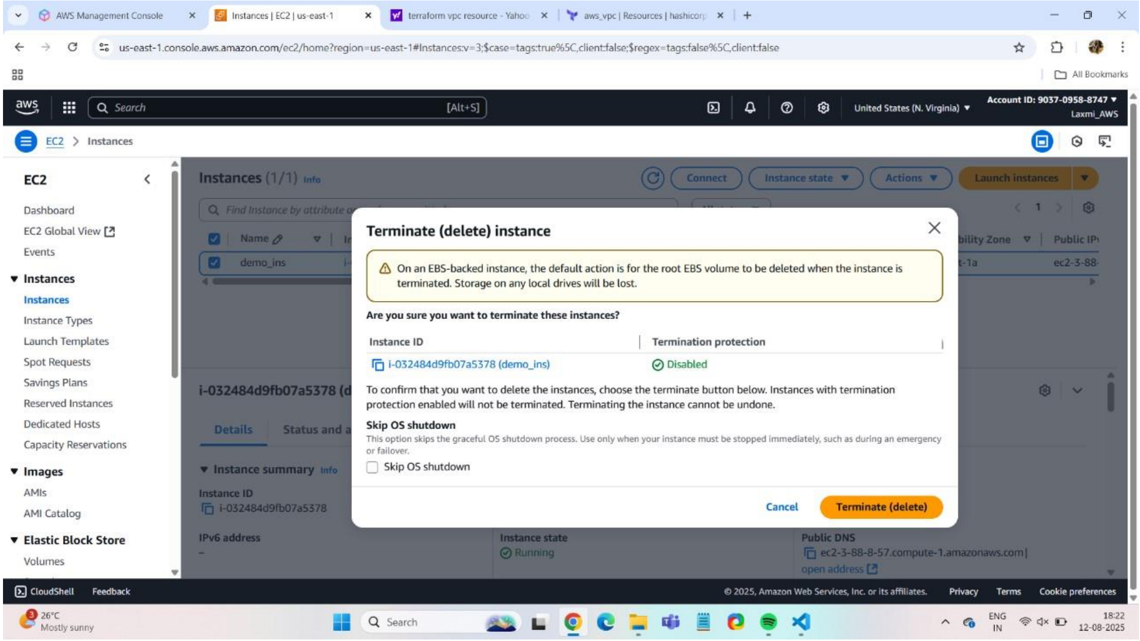
Select a subnet



Instances (1) Info

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
□	demo_ins	i-032484d9fb07a5378	Running	t3.micro	Initializing	View alarms +	us-east-1a	ec2-3-88-

NOTE : If I terminate the instance manually and again fetch command terraform apply then what will happen ? => It will create a new instance and 1 added message will be display.



Terminate (delete) instance

On an EBS-backed instance, the default action is for the root EBS volume to be deleted when the instance is terminated. Storage on any local drives will be lost.

Are you sure you want to terminate these instances?

Instance ID	Termination protection
i-032484d9fb07a5378 (demo_ins)	Disabled

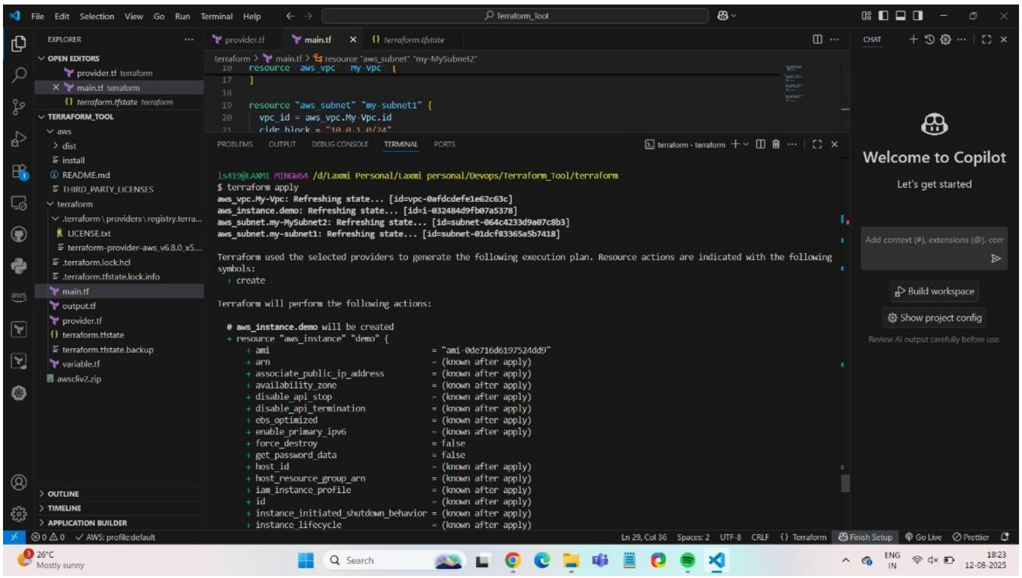
To confirm that you want to delete the instances, choose the terminate button below. Instances with termination protection enabled will not be terminated. Terminating the instance cannot be undone.

Skip OS shutdown

This option skips the graceful OS shutdown process. Use only when your instance must be stopped immediately, such as during an emergency or failover.

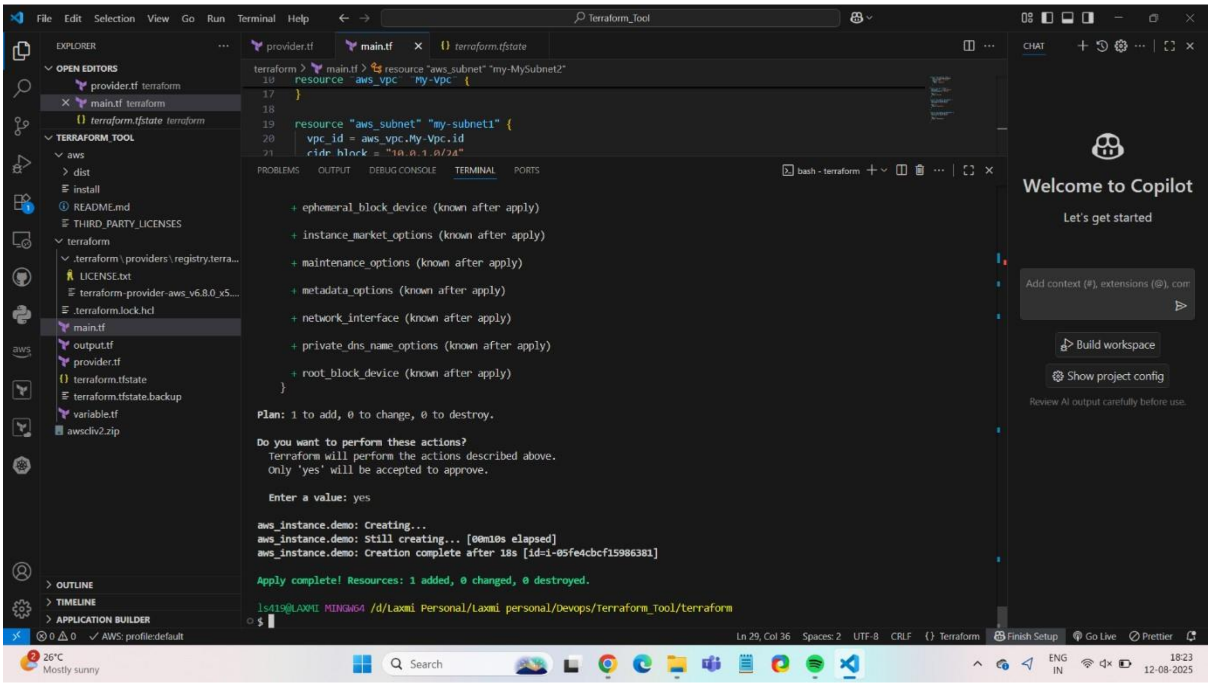
□ Skip OS shutdown

Cancel | Terminate (delete)



```
resource "aws_vpc" "My-vpc" {
}

resource "aws_subnet" "my-subnet1" {
  vpc_id = aws_vpc.My-Vpc.id
  cidr_block = "10.0.1.0/24"
```

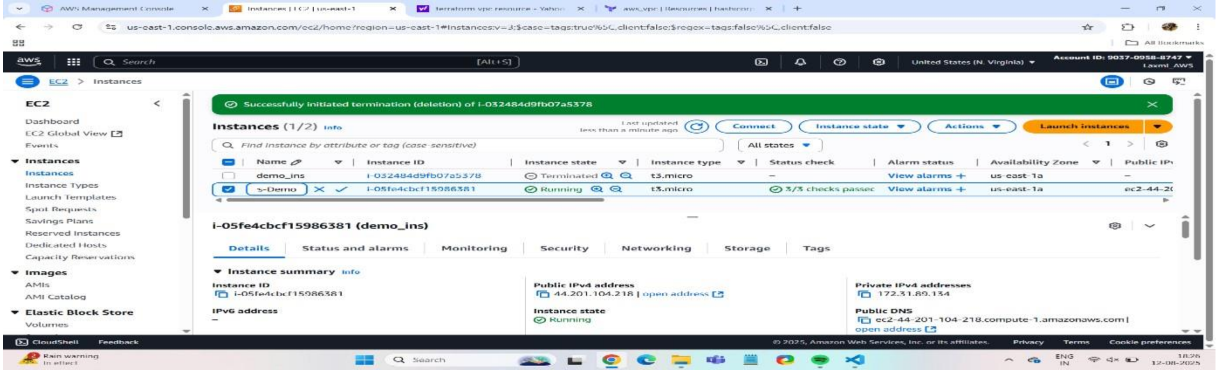
```
ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$ terraform apply
aws_vpc.My-Vpc: Refreshing state... [id=vpc-0afdcdefe1e62c63c]
aws_instance.demo: Refreshing state... [id=i-032484d9fb07a5378]
aws_subnet.my-MySubnet2: Refreshing state... [id=subnet-064c4233d9a07c8b3]
aws_subnet.my-subnet1: Refreshing state... [id=subnet-01dcf83365a5b7418]

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.demo will be created
  + resource "aws_instance" "demo" {
      + ami                                  = "ami-0de716d6197524dd9"
      + 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)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
```



```
resource "aws_vpc" "My-vpc" {
}

resource "aws_subnet" "my-subnet1" {
  vpc_id = aws_vpc.My-Vpc.id
  cidr_block = "10.0.1.0/24"
```

```
      + ephemeral_block_device (known after apply)

      + instance_market_options (known after apply)

      + maintenance_options (known after apply)

      + metadata_options (known after apply)

      + 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

aws_instance.demo: Creating...
aws_instance.demo: Still creating... [00m10s elapsed]
aws_instance.demo: Creation complete after 18s [id=i-05fe4cbcf15986381]

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

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$
```

NOTE : If I change the instance name and fetch the command terraform apply then what will happen? => Here the instance name will be changed again and will replace the new name as name present in state.tf file.



us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances:v=3;$case=tags:true%5C,client:false;$regex=tags:false%5C,client:false

EC2 > Instances

Successfully initiated termination (deletion) of i-032484d9fb07a5378

Instances (1/2) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
demo_ins	i-032484d9fb07a5378	Terminated	t3.micro	–	View alarms +	us-east-1a	–
s-Demo	i-05fe4cbcf15986381	Running	t3.micro	3/3 checks passec	View alarms +	us-east-1a	ec2-44-2(

i-05fe4cbcf15986381 (demo_ins)

Instance summary Info

Instance ID: i-05fe4cbcf15986381

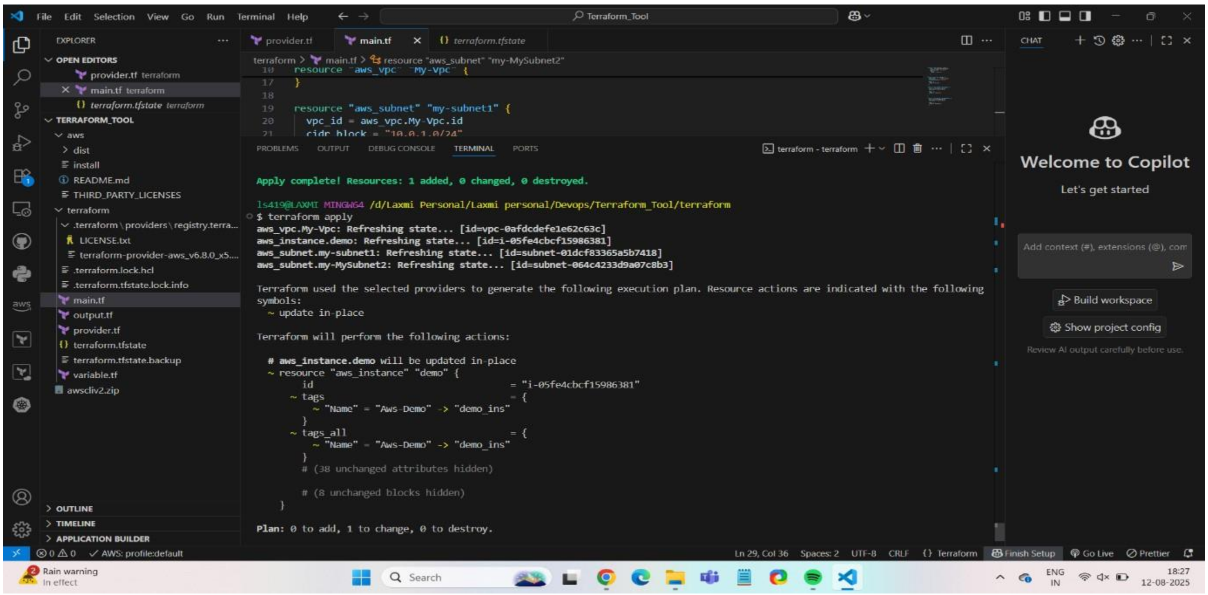
IPv6 address: –

Public IPv4 address: 44.201.104.218 | open address

Instance state: Running

Private IPv4 addresses: 172.31.89.134

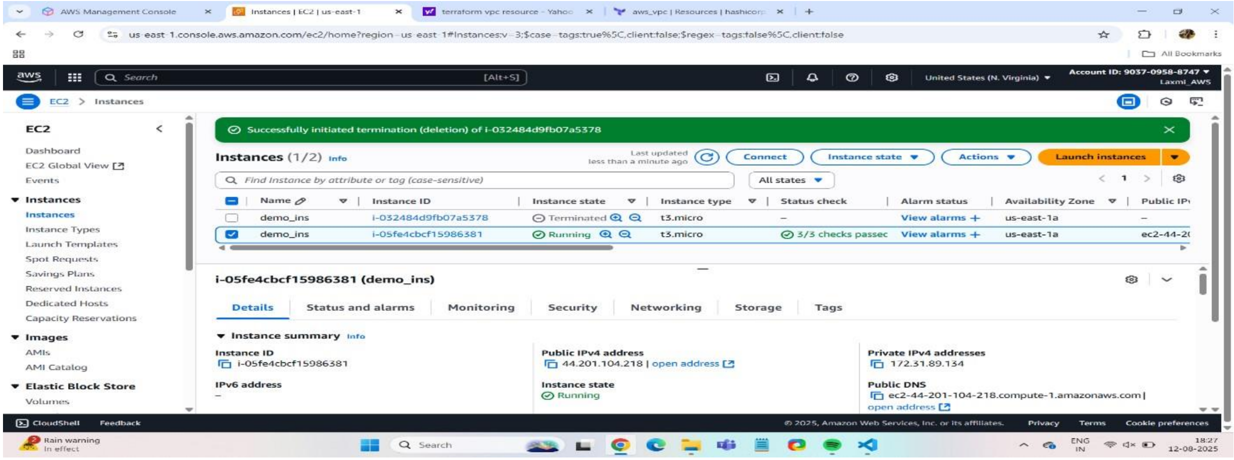
Public DNS: ec2-44-201-104-218.compute-1.amazonaws.com | open address



```
resource "aws_vpc" "My-Vpc" {
}

resource "aws_subnet" "my-subnet1" {
  vpc_id = aws_vpc.My-Vpc.id
  cidr_block = "10.0.1.0/24"
```

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

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$ terraform apply
aws_vpc.My-Vpc: Refreshing state... [id=vpc-0afdcdefe1e62c63c]
aws_instance.demo: Refreshing state... [id=i-05fe4cbcf15986381]
aws_subnet.my-subnet1: Refreshing state... [id=subnet-01dcf83365a5b7418]
aws_subnet.my-MySubnet2: Refreshing state... [id=subnet-064c4233d9a07c8b3]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following
symbols:
  ~ update in-place

Terraform will perform the following actions:

  # aws_instance.demo will be updated in-place
  ~ resource "aws_instance" "demo" {
        id                                   = "i-05fe4cbcf15986381"
      ~ tags                                 = {
          ~ "Name" = "Aws-Demo" -> "demo_ins"
        }
      ~ tags_all                             = {
          ~ "Name" = "Aws-Demo" -> "demo_ins"
        }
        # (38 unchanged attributes hidden)

        # (8 unchanged blocks hidden)
    }

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

File Edit Selection View Go Run Terminal Help
Terraform_Tool
provider.tf
main.tf
terraform.tfstate
terraform > main.tf > resource "aws_subnet" "my-MySubnet2"
EXPLORER
OPEN EDITORS
TERRAFORM_TOOL
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Welcome to Copilot
Let's get started
Build workspace
Show project config
Ln 29, Col 36 Spaces: 2 UTF-8 CRLF Terraform
18:27
12-08-2025



Successfully initiated termination (deletion) of i-032484d9fb07a5378

Instances (1/2) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
demo_ins	i-032484d9fb07a5378	Terminated	t3.micro	–	View alarms +	us-east-1a	–
demo_ins	i-05fe4cbcf15986381	Running	t3.micro	3/3 checks passec	View alarms +	us-east-1a	ec2-44-2(

i-05fe4cbcf15986381 (demo_ins)

Instance summary

Instance ID: i-05fe4cbcf15986381

IPv6 address: –

Public IPv4 address: 44.201.104.218 | open address

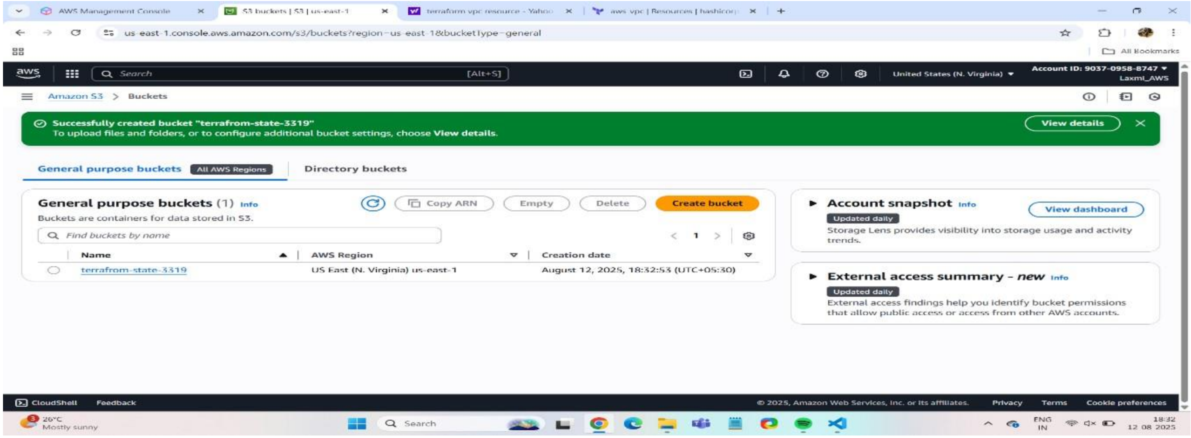
Instance state: Running

Private IPv4 addresses: 172.31.89.134

Public DNS: ec2-44-201-104-218.compute-1.amazonaws.com | open address

us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances:v=3;$case=tags:true%5C,client:false;$regex=tags:false%5C,client:false
United States (N. Virginia)
Account ID: 9037-0958-8747
Laxmi_AWS
EC2 > Instances
Connect
Instance state
Actions
Launch instances
All states
Details
Status and alarms
Monitoring
Security
Networking
Storage
Tags
18:27
12-08-2025

STEP 5 :- Now create one s3 bucket manually.



Successfully created bucket "terrafrom-state-3319"
To upload files and folders, or to configure additional bucket settings, choose **View details**.

General purpose buckets (1) Info

Buckets are containers for data stored in S3.

Name	AWS Region	Creation date
terrafrom-state-3319	US East (N. Virginia) us-east-1	August 12, 2025, 18:32:53 (UTC+05:30)

Account snapshot Info
Updated daily
Storage Lens provides visibility into storage usage and activity trends.

External access summary - *new* Info
Updated daily
External access findings help you identify bucket permissions that allow public access or access from other AWS accounts.

STEP 6 :- Now create one backend.tf file and write below script in it.

terraform { backend

= “s3” {   bucket

= “terraform-

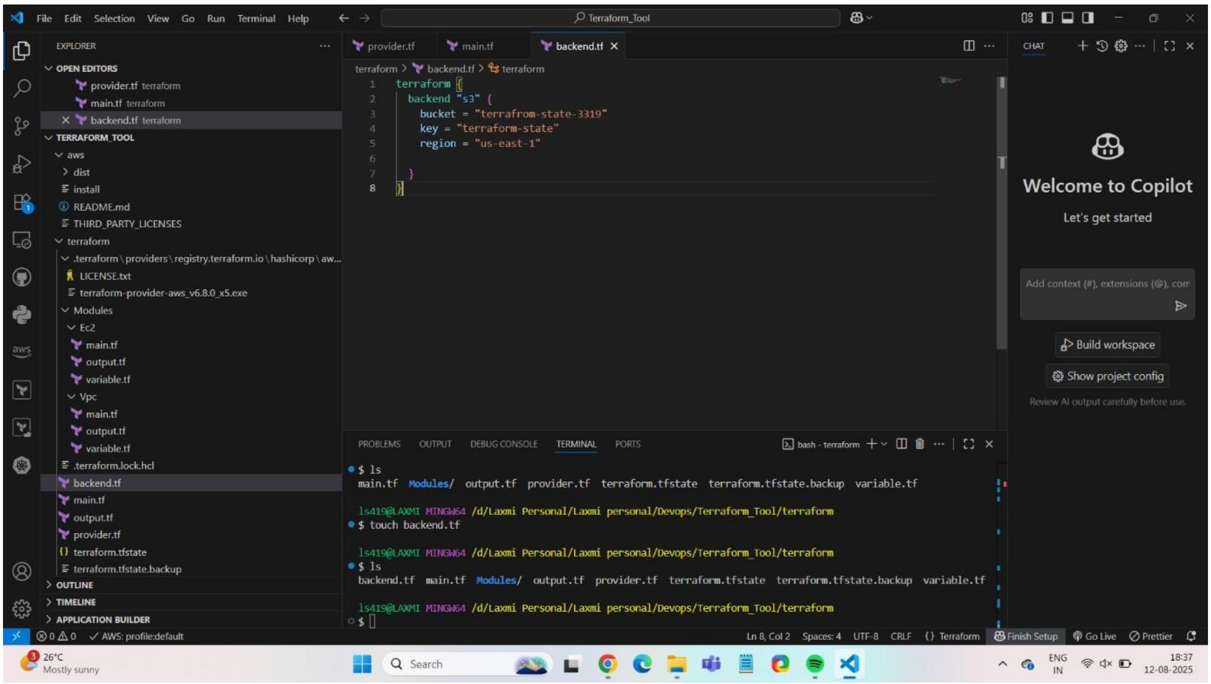
state3319”   key

= “terraform-state”

region = “us-east-1”

}

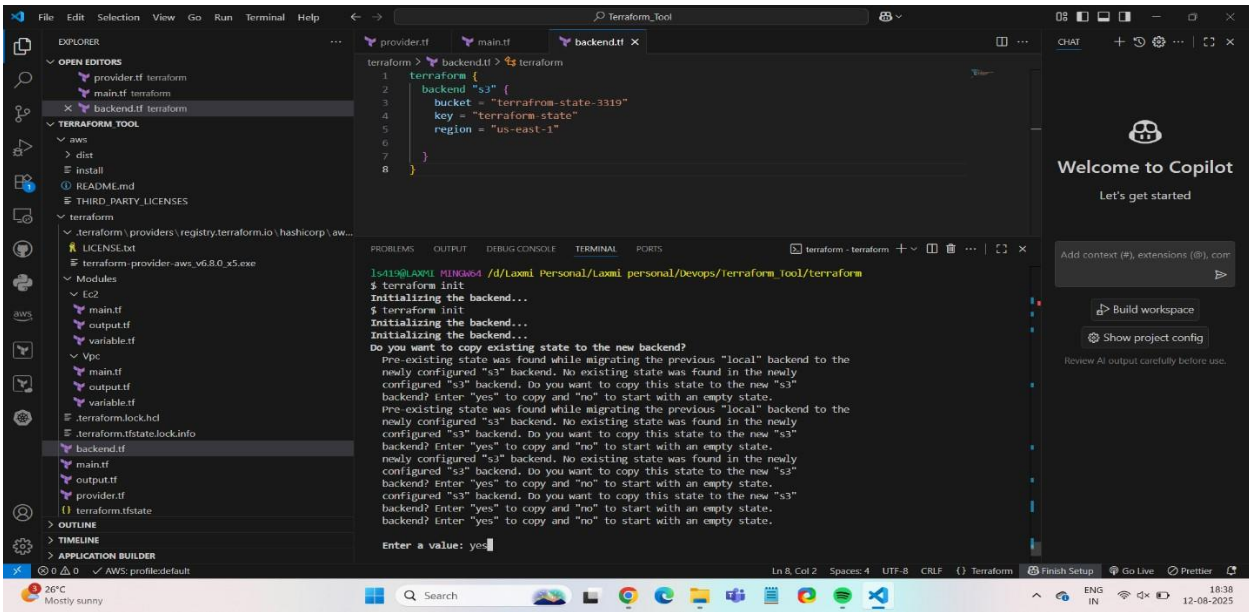
}



```
terraform {
  backend "s3" {
    bucket = "terrafrom-state-3319"
    key = "terraform-state"
    region = "us-east-1"

  }
}
```

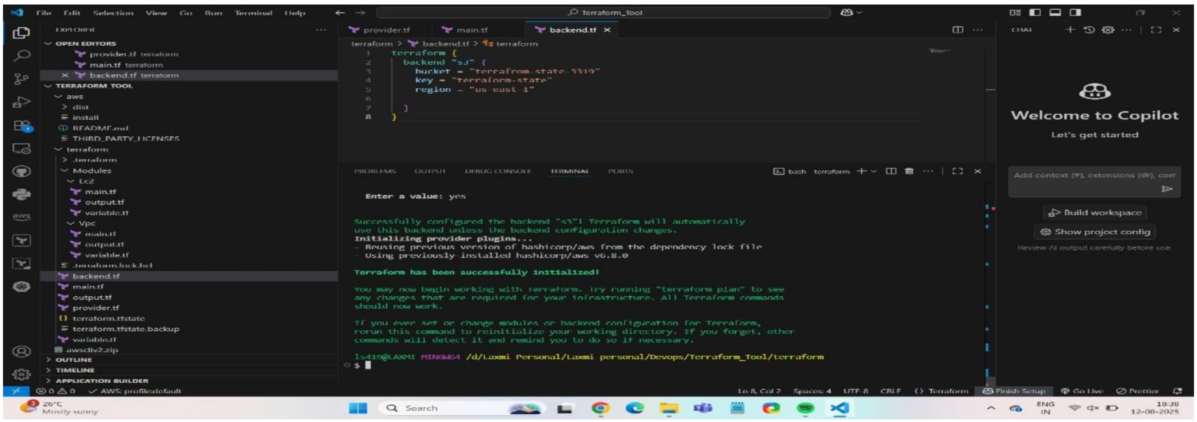
```
$ ls
main.tf  Modules/  output.tf  provider.tf  terraform.tfstate  terraform.tfstate.backup  variable.tf

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$ touch backend.tf

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$ ls
backend.tf  main.tf  Modules/  output.tf  provider.tf  terraform.tfstate  terraform.tfstate.backup  variable.tf

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$ 
```

STEP 7 :- Now again fetch the command **terraform init** as we have made some changes by creating s3 bucket and backend.tf



```
terraform {
  backend "s3" {
    bucket = "terrafrom-state-3319"
    key = "terraform-state"
    region = "us-east-1"

  }
}
```

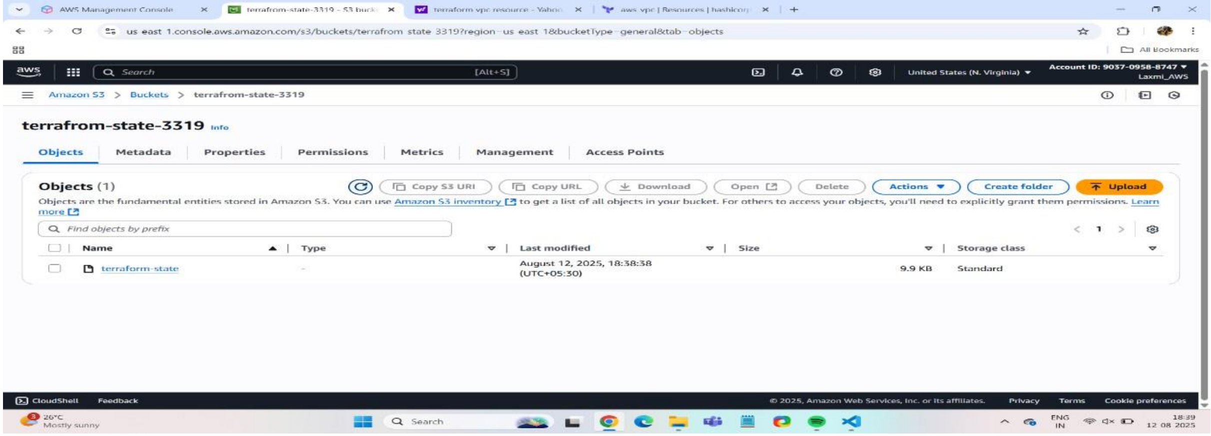
```
ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$ terraform init
Initializing the backend...
$ terraform init
Initializing the backend...
Initializing the backend...
Do you want to copy existing state to the new backend?
  Pre-existing state was found while migrating the previous "local" backend to the
  newly configured "s3" backend. No existing state was found in the newly
  configured "s3" backend. Do you want to copy this state to the new "s3"
  backend? Enter "yes" to copy and "no" to start with an empty state.
  Pre-existing state was found while migrating the previous "local" backend to the
  newly configured "s3" backend. No existing state was found in the newly
  configured "s3" backend. Do you want to copy this state to the new "s3"
  backend? Enter "yes" to copy and "no" to start with an empty state.
  newly configured "s3" backend. No existing state was found in the newly
  configured "s3" backend. Do you want to copy this state to the new "s3"
  backend? Enter "yes" to copy and "no" to start with an empty state.
  configured "s3" backend. Do you want to copy this state to the new "s3"
  backend? Enter "yes" to copy and "no" to start with an empty state.
  backend? Enter "yes" to copy and "no" to start with an empty state.

  Enter a value: yes
```



```
terraform {
  backend "s3" {
    bucket = "terrafrom-state-3319"
    key = "terraform-state"
    region = "us-east-1"

  }
}
```

```
  Enter a value: yes


Successfully configured the backend "s3"! Terraform will automatically
use this backend unless the backend configuration changes.
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously installed hashicorp/aws v6.8.0

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.

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$ 
```

STEP 8 :- Now terraform-state folder will be automatically gets created inside s3 bucket and all the data inside state file will be moved to s3 bucket terraformstate file.

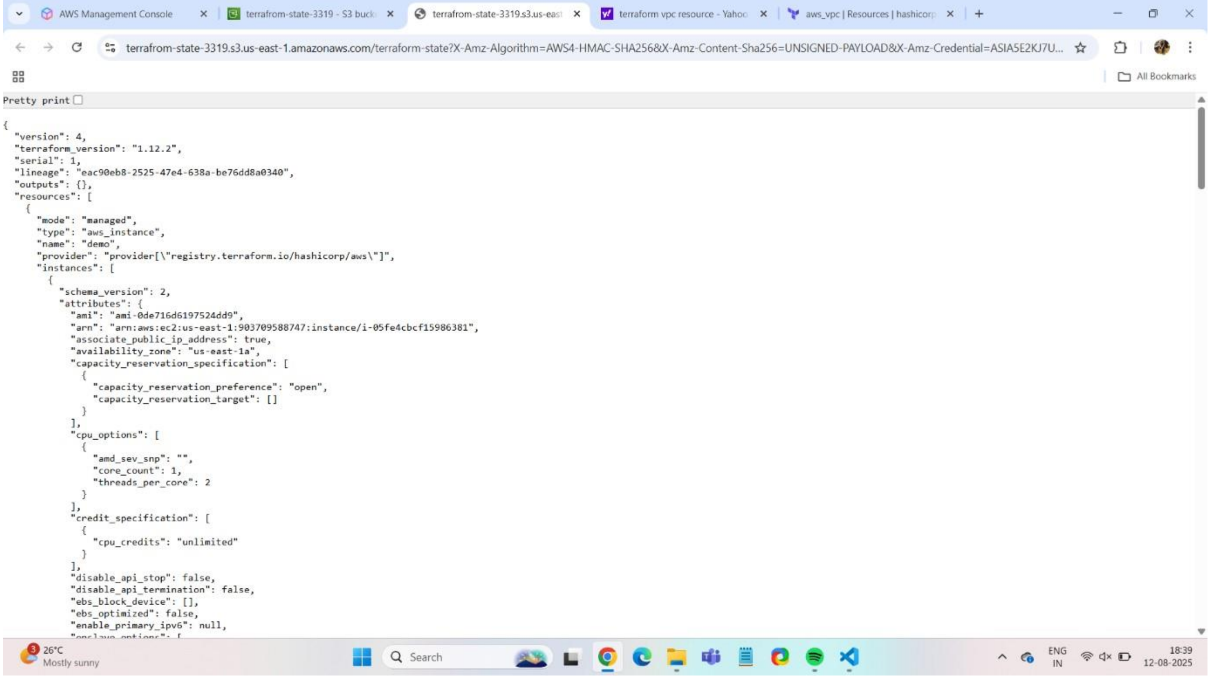


terrafrom-state-3319

Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

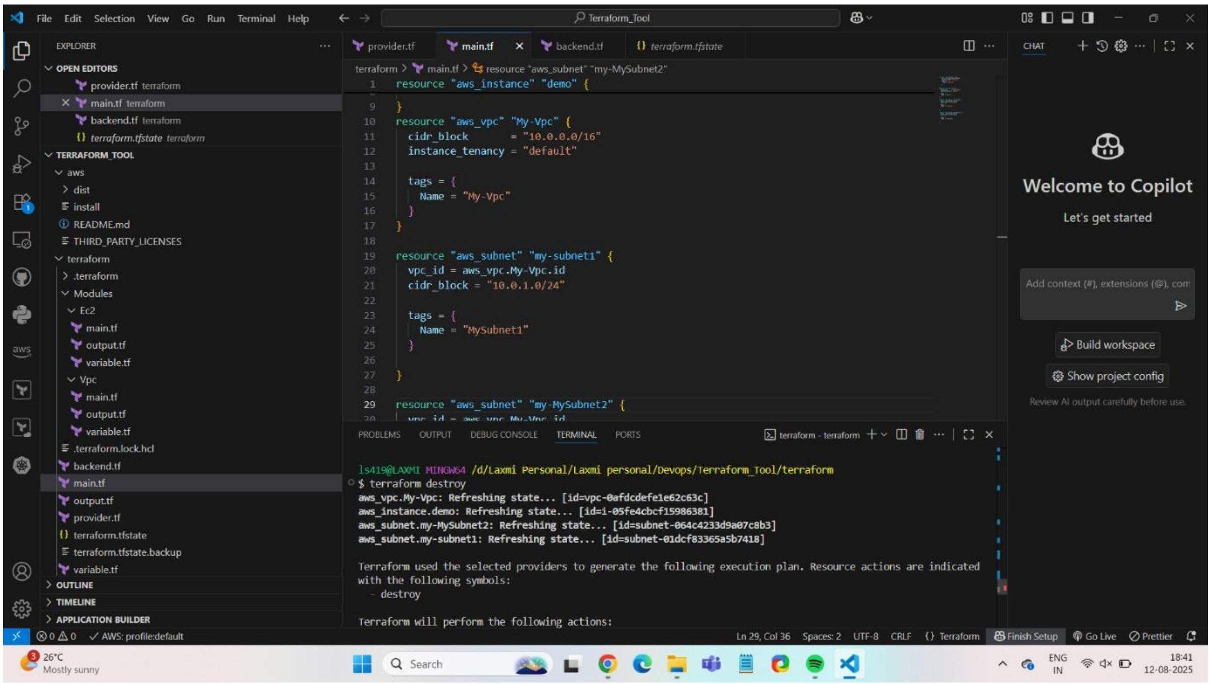
Name	Type	Last modified	Size	Storage class
terraform-state	-	August 12, 2025, 18:38:38 (UTC+05:30)	9.9 KB	Standard



terrafrom-state-3319.s3.us-east-1.amazonaws.com/terraform-state?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Content-Sha256=UNSIGNED-PAYLOAD&X-Amz-Credential=ASIA5E2KJ7U...

```
{
  "version": 4,
  "terraform_version": "1.12.2",
  "serial": 1,
  "lineage": "eac90eb8-2525-47e4-638a-be76dd8a0340",
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "aws_instance",
      "name": "demo",
      "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 2,
          "attributes": {
            "ami": "ami-0de716d6197524dd9",
            "arn": "arn:aws:ec2:us-east-1:903709588747:instance/i-05fe4cbcf15986381",
            "associate_public_ip_address": true,
            "availability_zone": "us-east-1a",
            "capacity_reservation_specification": [
              {
                "capacity_reservation_preference": "open",
                "capacity_reservation_target": []
              }
            ],
            "cpu_options": [
              {
                "amd_sev_snp": "",
                "core_count": 1,
                "threads_per_core": 2
              }
            ],
            "credit_specification": [
              {
                "cpu_credits": "unlimited"
              }
            ],
            "disable_api_stop": false,
            "disable_api_termination": false,
            "ebs_block_device": [],
            "ebs_optimized": false,
            "enable_primary_ipv6": null,
```

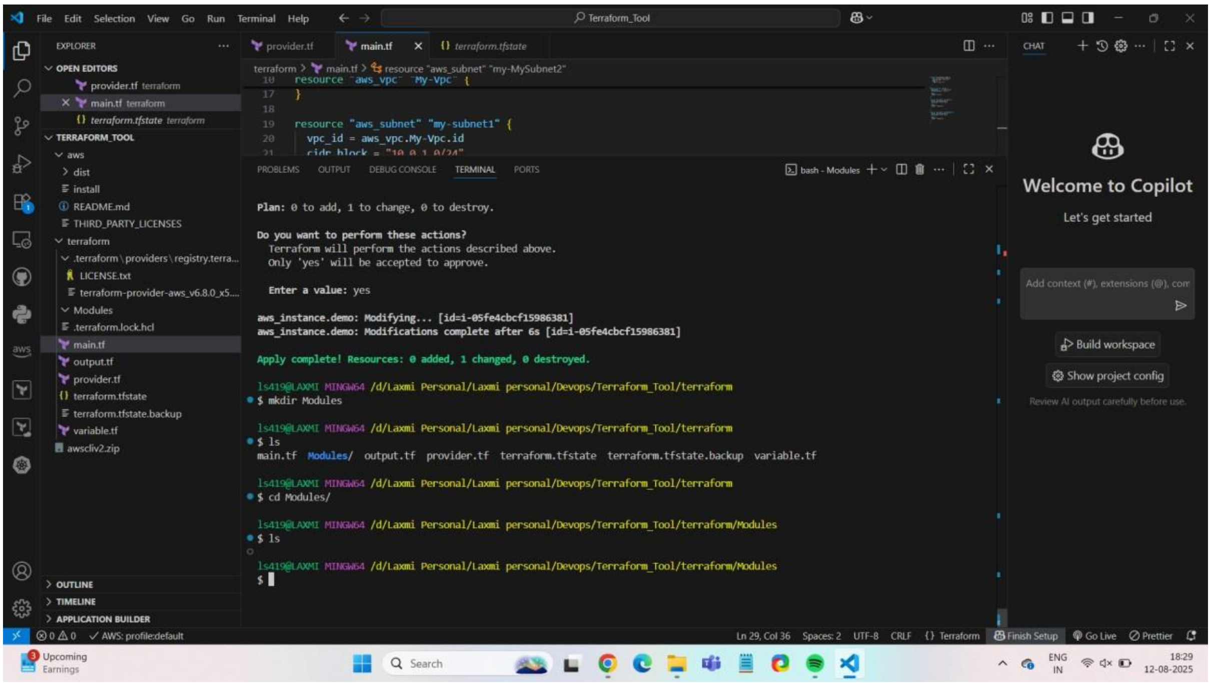
STEP 9 :- Now fetch terraform destroy command to delete the entire work.



```
resource "aws_instance" "demo" {
}
resource "aws_vpc" "My-Vpc" {
  cidr_block       = "10.0.0.0/16"
  instance_tenancy = "default"

  tags = {
    Name = "My-Vpc"
  }
}

resource "aws_subnet" "my-subnet1" {
  vpc_id = aws_vpc.My-Vpc.id
  cidr_block = "10.0.1.0/24"

  tags = {
    Name = "MySubnet1"
  }

}

resource "aws_subnet" "my-MySubnet2" {
  vpc_id = aws_vpc.My-Vpc.id
```

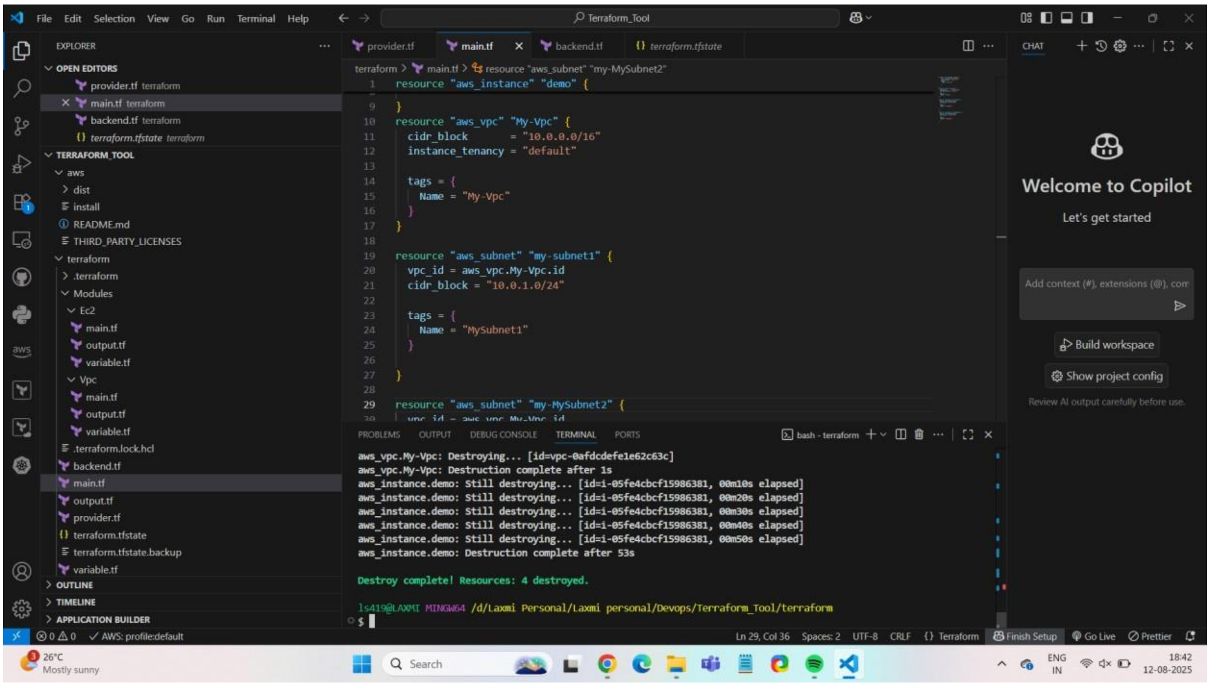
```
ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$ terraform destroy
aws_vpc.My-Vpc: Refreshing state... [id=vpc-0afdcdefe1e62c63c]
aws_instance.demo: Refreshing state... [id=i-05fe4cbcf15986381]
aws_subnet.my-MySubnet2: Refreshing state... [id=subnet-064c4233d9a07c8b3]
aws_subnet.my-subnet1: Refreshing state... [id=subnet-01dcf83365a5b7418]

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:
```

STEP 10 :- Now create module folder inside that create two more folder name ec2 and vpc. Also create 3 files name main.tf variable.tf output.tf inside each ec2 and vpc folder. Copy paste all the data from main.tf instance data to ec2 main.tf folder and vpc data from main.tf to vpc main.tf folder.



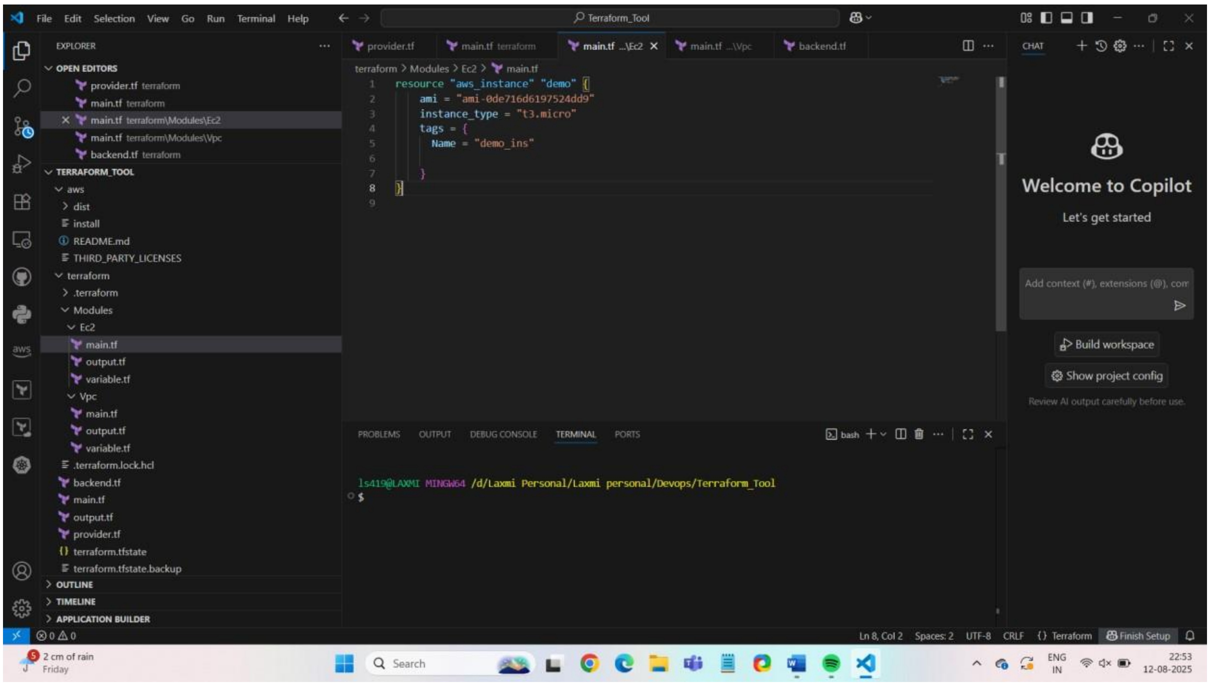
```
resource "aws_vpc" "My-Vpc" {
}

resource "aws_subnet" "my-subnet1" {
  vpc_id = aws_vpc.My-Vpc.id
  cidr_block = "10.0.1.0/24"
```

```
Plan: 0 to add, 1 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.demo: Modifying... [id=i-05fe4cbcf15986381]
aws_instance.demo: Modifications complete after 6s [id=i-05fe4cbcf15986381]

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

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$ mkdir Modules

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$ ls
main.tf  Modules/  output.tf  provider.tf  terraform.tfstate  terraform.tfstate.backup  variable.tf

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$ cd Modules/

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform/Modules
$ ls

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform/Modules
$ 
```



```
resource "aws_instance" "demo" {
}
resource "aws_vpc" "My-Vpc" {
  cidr_block       = "10.0.0.0/16"
  instance_tenancy = "default"

  tags = {
    Name = "My-Vpc"
  }
}

resource "aws_subnet" "my-subnet1" {
  vpc_id = aws_vpc.My-Vpc.id
  cidr_block = "10.0.1.0/24"

  tags = {
    Name = "MySubnet1"
  }

}

resource "aws_subnet" "my-MySubnet2" {
  vpc_id = aws_vpc.My-Vpc.id
```

```
aws_vpc.My-Vpc: Destroying... [id=vpc-0afdcdefe1e62c63c]
aws_vpc.My-Vpc: Destruction complete after 1s
aws_instance.demo: Still destroying... [id=i-05fe4cbcf15986381, 00m10s elapsed]
aws_instance.demo: Still destroying... [id=i-05fe4cbcf15986381, 00m20s elapsed]
aws_instance.demo: Still destroying... [id=i-05fe4cbcf15986381, 00m30s elapsed]
aws_instance.demo: Still destroying... [id=i-05fe4cbcf15986381, 00m40s elapsed]
aws_instance.demo: Still destroying... [id=i-05fe4cbcf15986381, 00m50s elapsed]
aws_instance.demo: Destruction complete after 53s

Destroy complete! Resources: 4 destroyed.

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$
```



```
resource "aws_instance" "demo" {
    ami = "ami-0de716d6197524dd9"
    instance_type = "t3.micro"
    tags = {
      Name = "demo_ins"

    }
}
```

```
ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool
$
```

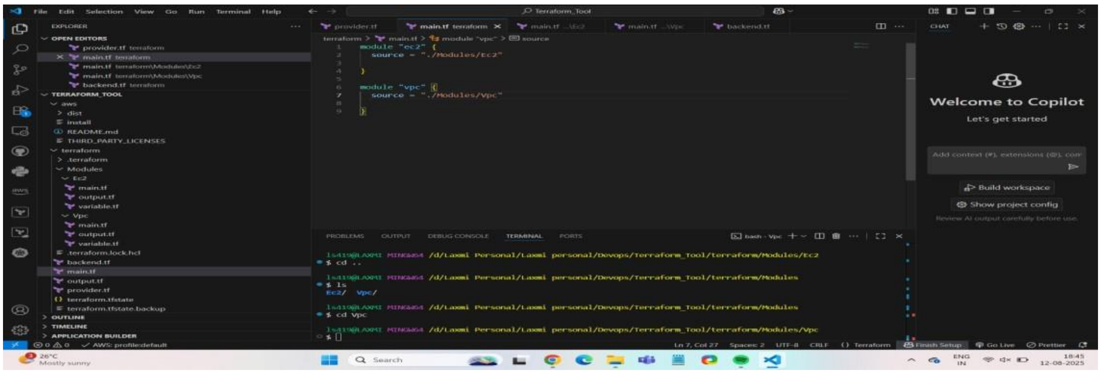
STEP 11 :- Now in main.tf write below script

module "ec2" { source = "./module/ec2"

}

module "vpc" {

source = "./modue/vpc }



```
module "ec2" {
  source = "./Modules/Ec2"

}

module "vpc" {
  source = "./Modules/Vpc"

}
```

```
ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform/Modules/Ec2
$ cd ..

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform/Modules
$ ls
Ec2/  Vpc/

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform/Modules
$ cd Vpc

ls419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform/Modules/Vpc
$ 
```