Software Provisioning and Configuration Management

LAB FILE SUBMITTED BY:

NAME: Palak Gupta

SAP ID: 500109569

BATCH: 1 Devops

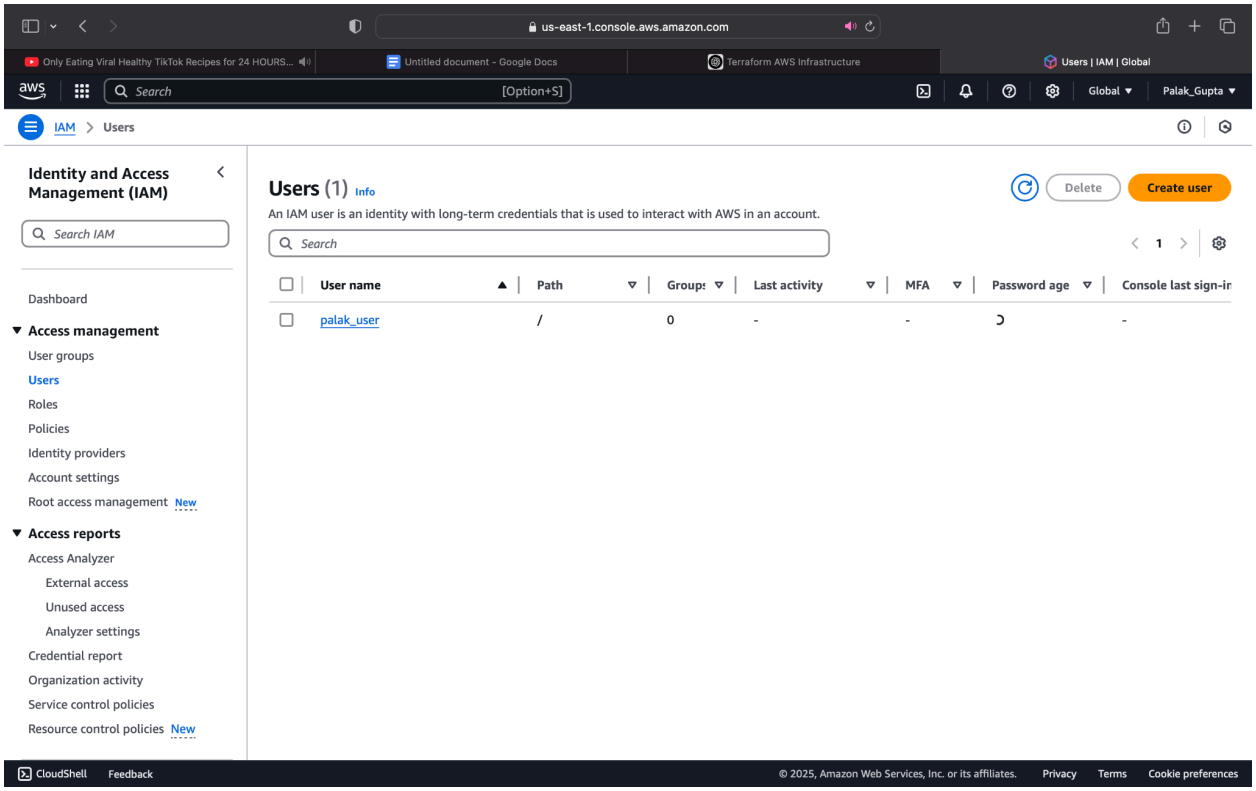
ASSIGNMENT 1

Write Terraform script to do perform following tasks on AWS cloud Platform

First create a IAM user and attach AdministratorAccess policy to it also create access key for this user and download it

palak-user_accessKeys-3

Access key ID	Secret access key
AKIAQMEY6IA6ULZVXAVS	PcKZJQhcPeFoCJMSQ/gxm/zp7Ctr+ZeF0eTj1fOR

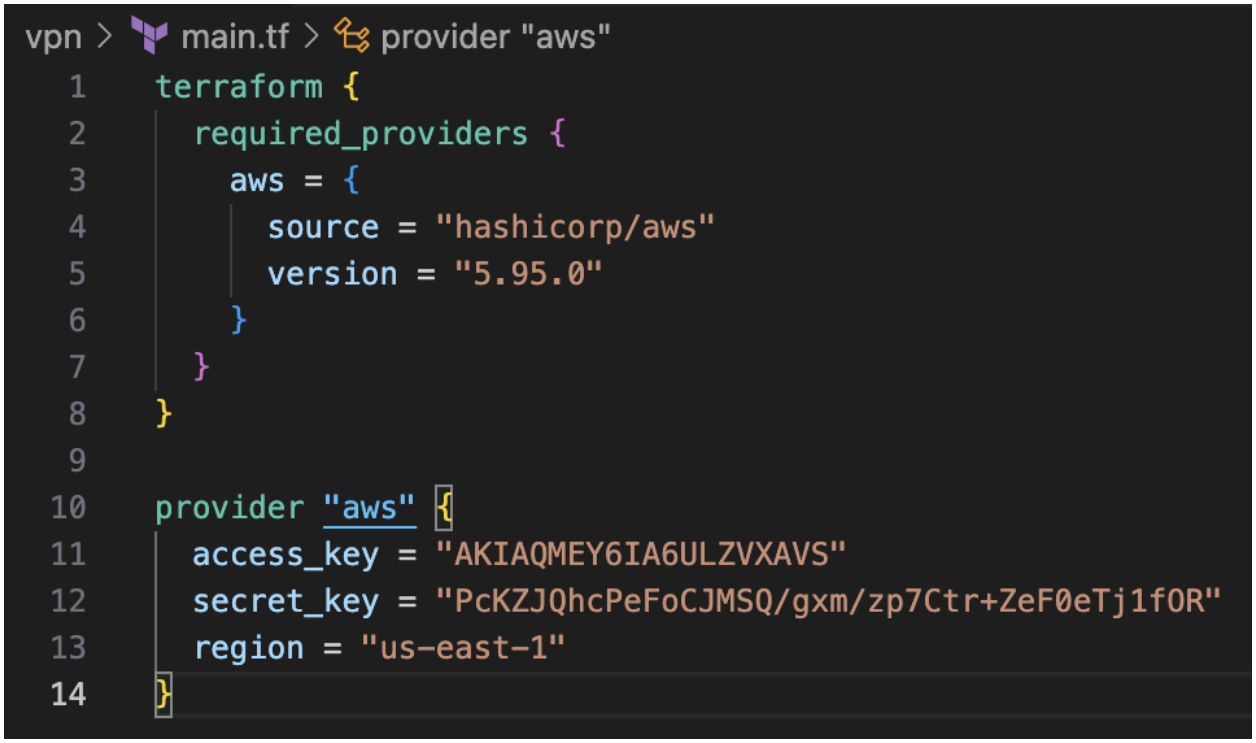


Step 1: Create two T2 Micro EC2 Instances.

Now in any text editor create a folder and add two files named:

main.tf

This file contain the provider information and access key and secret key to connect to your IAM user



variable.tf

This file contains the information about the resource you want to create. When creating a ec2 we need to provide the ami and type of the instance we want to create. I have also provided the count of instances as 2.

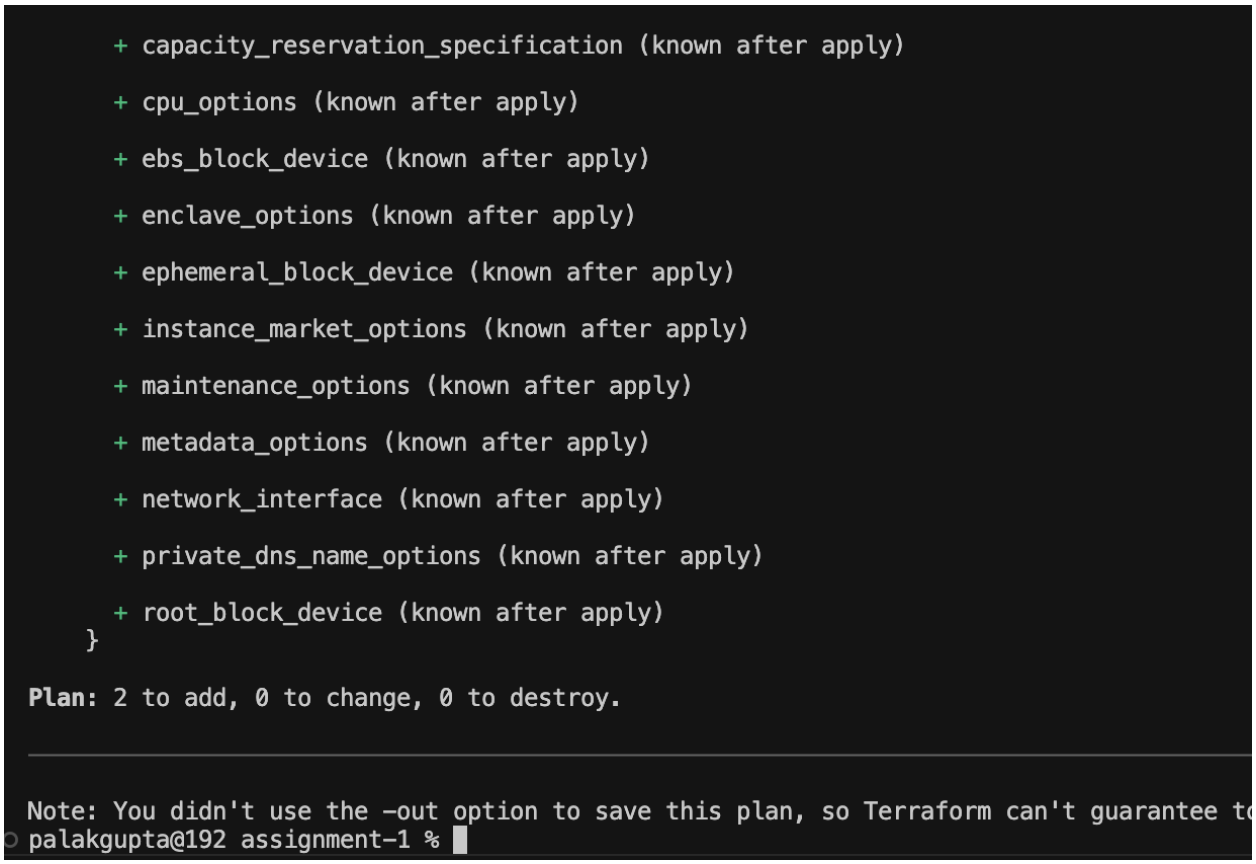
variable.tf > resource "aws_instance" "ec2"

```
1   resource "aws_instance" "ec2" {
2     ami = "ami-0e449927258d45bc4"
3     instance_type = "t2.micro"
4     count = 2
5
6     tags = {
7       Name = "palak-ec2"
8     }
9   }
```

In order to use terraform it is essential to first use the command : terraform init

```
palakgupta@Palaks-MacBook-Air assignment-1 % cd vpn
palakgupta@Palaks-MacBook-Air vpn % terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.95.0"...
- Installing hashicorp/aws v5.95.0...
- Installed hashicorp/aws v5.95.0 (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.
```

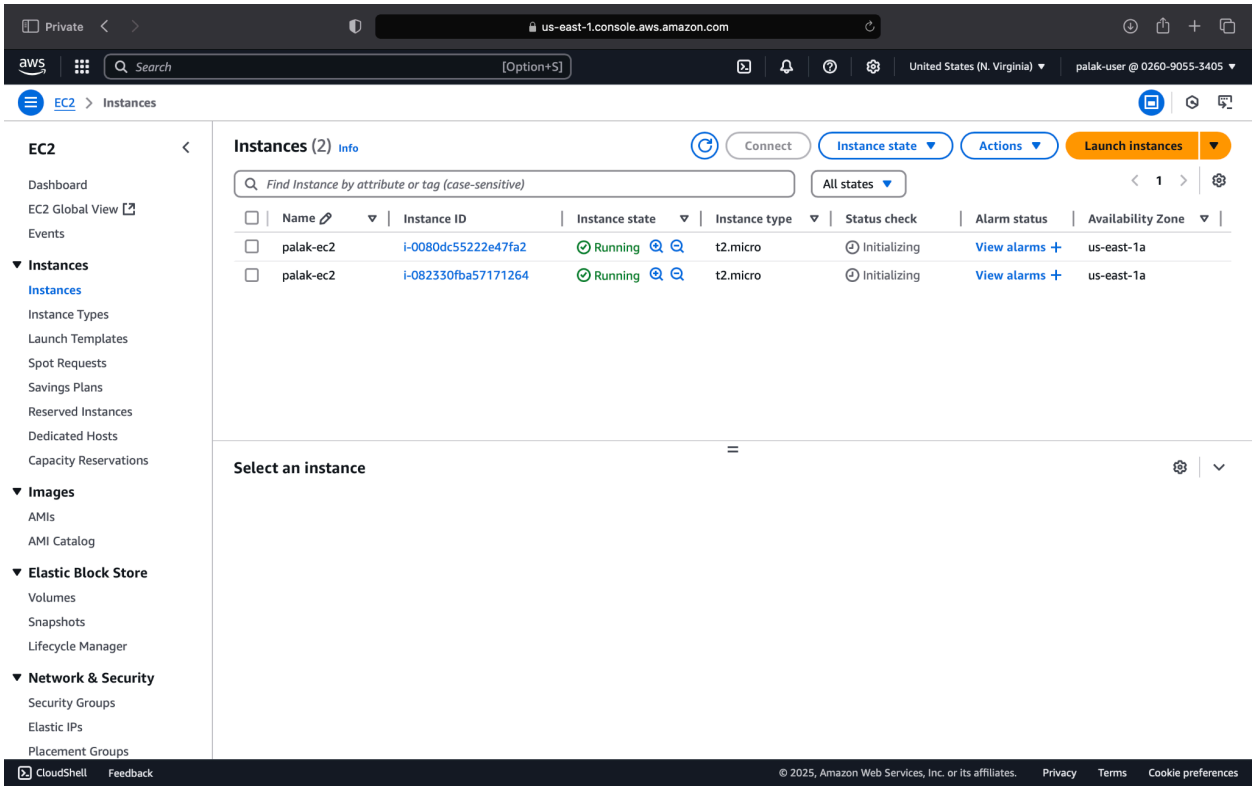
Then to confirm your changes you can use terraform plan to see what all changes will be made, here we can see 2 ec2 instance of type t2.micro will be created



To apply these changes terraform apply or terraform apply -auto-approve(does not ask for verification) is used

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

      + 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: 2 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.ec2[0]: Creating...
aws_instance.ec2[1]: Creating...
aws_instance.ec2[1]: Still creating... [10s elapsed]
aws_instance.ec2[0]: Still creating... [10s elapsed]
aws_instance.ec2[0]: Creation complete after 18s [id=i-082330fba57171264]
aws_instance.ec2[1]: Creation complete after 18s [id=i-0080dc55222e47fa2]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
palakgupta@Palaks-MacBook-Air assignment-1 %
```

We can view our changes via logining into our console

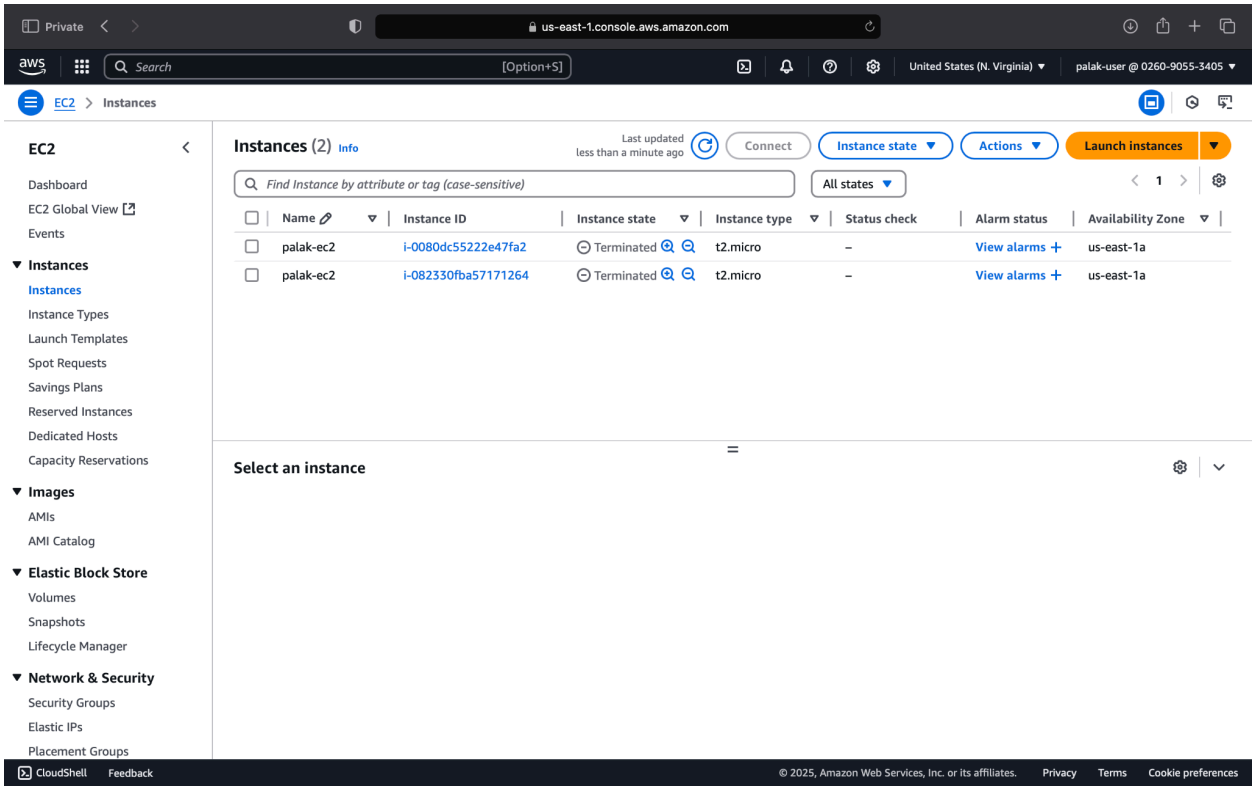


To discard the changes made use terraform destroy or terraform destroy -auto-approve. This will delete the changes made earlier

```
palakgupta@Palaks-MacBook-Air assignment-1 % terraform destroy -auto-approve
aws_instance.ec2[1]: Refreshing state... [id=i-0080dc55222e47fa2]
aws_instance.ec2[0]: Refreshing state... [id=i-082330fba57171264]

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.ec2[0] will be destroyed
  - resource "aws_instance" "ec2" {
      - ami                                  = "ami-0e449927258d45bc4" -> null
      - arn                                  = "arn:aws:ec2:us-east-1:026090553405:instance/i-082330fba57171264" -> null
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "us-east-1a" -> null
      - cpu_core_count                       = 1 -> null
      - cpu_threads_per_core                 = 1 -> null
      - disable_api_stop                     = false -> null
      - disable_api_termination              = false -> null
      - ebs_optimized                        = false -> null
      - get_password_data                    = false -> null
      - hibernation                          = false -> null
      - id                                   = "i-082330fba57171264" -> null
      - instance_initiated_shutdown_behavior = "stop" -> null
      - instance_state                       = "running" -> null
      - instance_type                        = "t2.micro" -> null
      - ipv6_address_count                   = 0 -> null
      - ipv6_addresses                       = [] -> null
      - monitoring                           = false -> null
      - placement_partition_number           = 0 -> null
      - primary_network_interface_id         = "eni-090cf519afedffe0b" -> null
      - private_dns                          = "ip-172-31-22-14.ec2.internal" -> null
      - private_ip                           = "172.31.22.14" -> null
      - public_dns                           = "ec2-54-175-42-74.compute-1.amazonaws.com" -> null
      - public_ip                            = "54.175.42.74" -> null
      - secondary_private_ips                = [] -> null
      - security_groups                      = [
          - "default",
        ] -> null
      - source_dest_check                    = true -> null
      - subnet_id                            = "subnet-01bb354a11d3835b9" -> null
      - tags                                 = {
          - "Name" = "palak-ec2"
        } -> null
      - tags_all                             = {
          - "Name" = "palak-ec2"
        } -> null
      - tenancy                              = "default" -> null
      - user_data_replace_on_change          = false -> null
      - vpc_security_group_ids               = [
          - "sg-0baf5641f098abf84",
        ] -> null
```

Ln 9, Col 2 Spaces: 2 UTF-8 LF {} Terraform

```
Plan: 0 to add, 0 to change, 2 to destroy.
aws_instance.ec2[0]: Destroying... [id=i-082330fba57171264]
aws_instance.ec2[1]: Destroying... [id=i-0080dc55222e47fa2]
aws_instance.ec2[0]: Still destroying... [id=i-082330fba57171264, 10s elapsed]
aws_instance.ec2[1]: Still destroying... [id=i-0080dc55222e47fa2, 10s elapsed]
aws_instance.ec2[0]: Still destroying... [id=i-082330fba57171264, 20s elapsed]
aws_instance.ec2[1]: Still destroying... [id=i-0080dc55222e47fa2, 20s elapsed]
aws_instance.ec2[0]: Still destroying... [id=i-082330fba57171264, 30s elapsed]
aws_instance.ec2[1]: Still destroying... [id=i-0080dc55222e47fa2, 30s elapsed]
aws_instance.ec2[0]: Destruction complete after 33s
aws_instance.ec2[1]: Still destroying... [id=i-0080dc55222e47fa2, 40s elapsed]
aws_instance.ec2[1]: Destruction complete after 44s

Destroy complete! Resources: 2 destroyed.
palakgupta@Palaks-MacBook-Air assignment-1 %
```



Step2: Create a VPN on AWS

In the same project folder, create new main.tf and vpc.tf .

In main.tf, keep the same provider block with access and secret keys from your IAM user.

vpn > main.tf > provider "aws"

```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.95.0"
    }
  }
}

provider "aws" {
  access_key = "AKIAQMEY6IA6ULZVXAVS"
  secret_key = "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"
  region = "us-east-1"
}
```

Use Terraform resources to create:

- A Virtual Private Gateway using aws_vpn_gateway
- A Customer Gateway using aws_customer_gateway (requires a static IP)
- A VPN Connection using aws_vpn_connection
- Optionally, use aws_vpn_gateway_attachment to attach the VPN to your existing VPC

vpn > vpc.tf > resource "aws_route_table" "public_rt"

```hcl
resource "aws_vpc" "my_vpc" {
  cidr_block = "10.0.0.0/16"

  tags = {
    Name = "my-vpc"
  }
}
resource "aws_subnet" "public_subnet" {
  vpc_id            = aws_vpc.my_vpc.id
  cidr_block        = "10.0.1.0/24"
  map_public_ip_on_launch = true
  availability_zone = "us-east-1a"

  tags = {
    Name = "my-public-subnet"
  }
}

resource "aws_internet_gateway" "igw" {
  vpc_id = aws_vpc.my_vpc.id

  tags = {
    Name = "my-igw"
  }
}

resource "aws_route_table" "public_rt" {
  vpc_id = aws_vpc.my_vpc.id

  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.igw.id
  }

  tags = {
    Name = "public-rt"
  }
}
resource "aws_route_table_association" "public_association" {
  subnet_id      = aws_subnet.public_subnet.id
  route_table_id = aws_route_table.public_rt.id
}
```

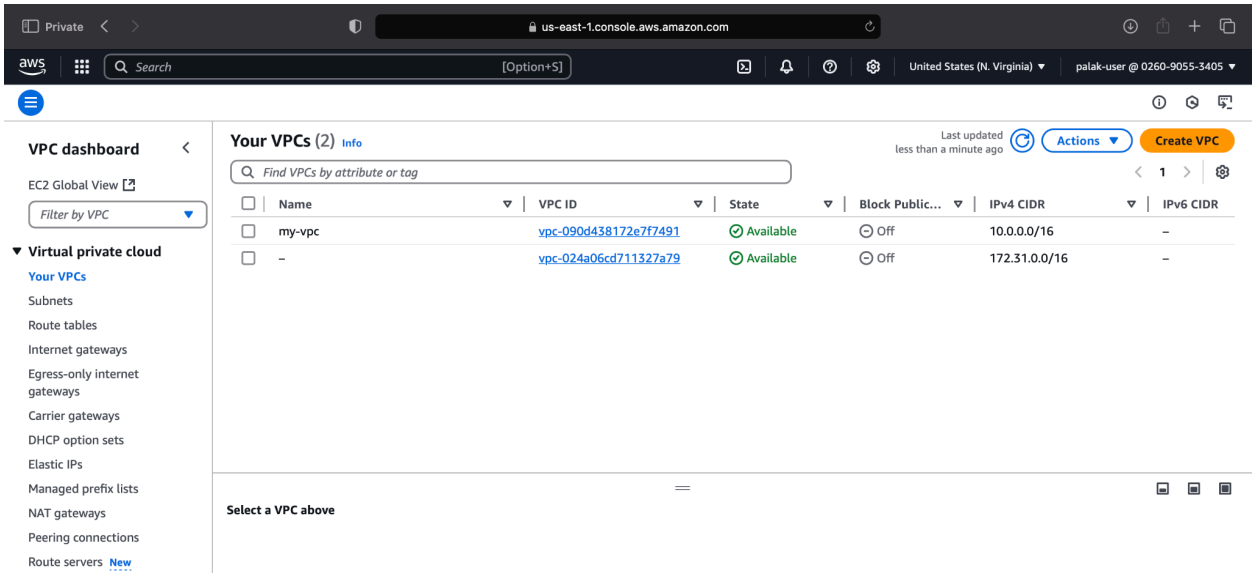
```
palakgupta@Palaks-MacBook-Air assignment-1 % cd vpn
palakgupta@Palaks-MacBook-Air vpn % terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.95.0"...
- Installing hashicorp/aws v5.95.0...
- Installed hashicorp/aws v5.95.0 (signed 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.
```

```
  # aws_vpc.my_vpc will be created
  + resource "aws_vpc" "my_vpc" {
      + 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)
      + id                                   = (known after apply)
      + instance_tenancy                     = "default"
      + ipv6_association_id                  = (known after apply)
      + ipv6_cidr_block                      = (known after apply)
      + ipv6_cidr_block_network_border_group = (known after apply)
      + main_route_table_id                  = (known after apply)
      + owner_id                             = (known after apply)
      + tags                                 = {
          + "Name" = "my-vpc"
        }
      + tags_all                             = {
          + "Name" = "my-vpc"
        }
    }

Plan: 5 to add, 0 to change, 0 to destroy.
```

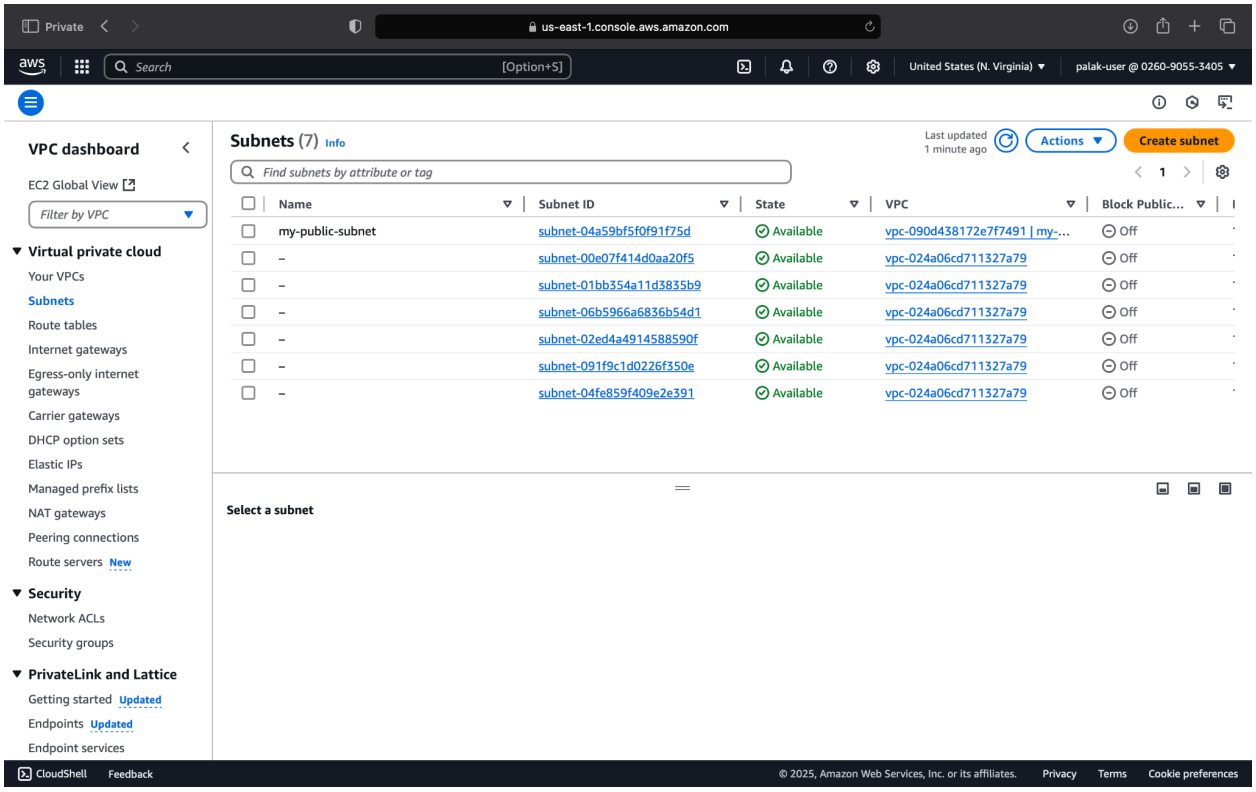
```
  # aws_vpc.my_vpc will be created
  + resource "aws_vpc" "my_vpc" {
      + arn                                  = (known after apply)
      + cidr_block                           = "10.0.0.0/16"
      + default_network_acl_id               = (known after apply)
      + default_route_table_id               = (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)
      + id                                   = (known after apply)
      + instance_tenancy                     = "default"
      + ipv6_association_id                  = (known after apply)
      + ipv6_cidr_block                      = (known after apply)
      + ipv6_cidr_block_network_border_group = (known after apply)
      + main_route_table_id                  = (known after apply)
      + owner_id                             = (known after apply)
      + tags                                 = {
          + "Name" = "my-vpc"
        }
      + tags_all                             = {
          + "Name" = "my-vpc"
        }
    }

Plan: 5 to add, 0 to change, 0 to destroy.
aws_vpc.my_vpc: Creating...
aws_vpc.my_vpc: Creation complete after 4s [id=vpc-090d438172e7f7491]
aws_internet_gateway.igw: Creating...
aws_subnet.public_subnet: Creating...
aws_internet_gateway.igw: Creation complete after 2s [id=igw-0c09a197f26820600]
aws_route_table.public_rt: Creating...
aws_route_table.public_rt: Creation complete after 3s [id=rtb-056f93e8fdd0ce30a]
aws_subnet.public_subnet: Still creating... [10s elapsed]
aws_subnet.public_subnet: Creation complete after 13s [id=subnet-04a59bf5f0f91f75d]
aws_route_table_association.public_association: Creating...
aws_route_table_association.public_association: Creation complete after 2s [id=rtbassoc-02b149137501ef980]

Apply complete! Resources: 5 added, 0 changed, 0 destroyed.
palakgupta@Palaks-MacBook-Air vpn %
```

VPC

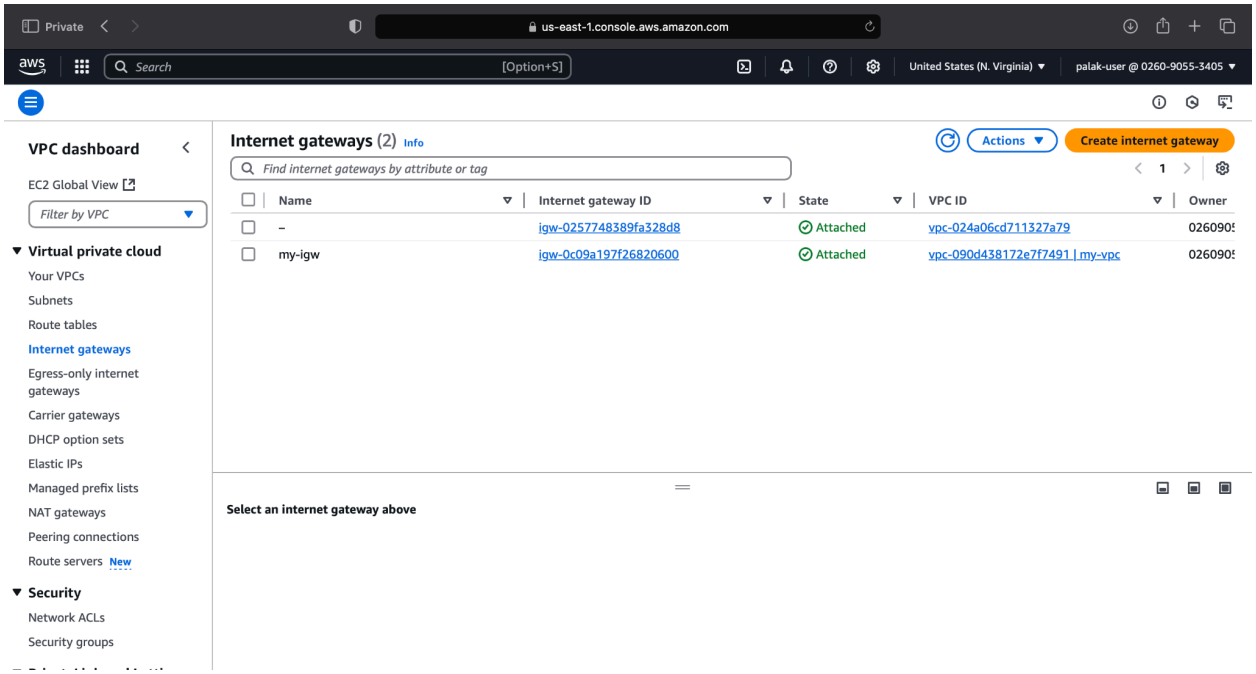
You can verify the setup in the AWS Console by navigating to VPC



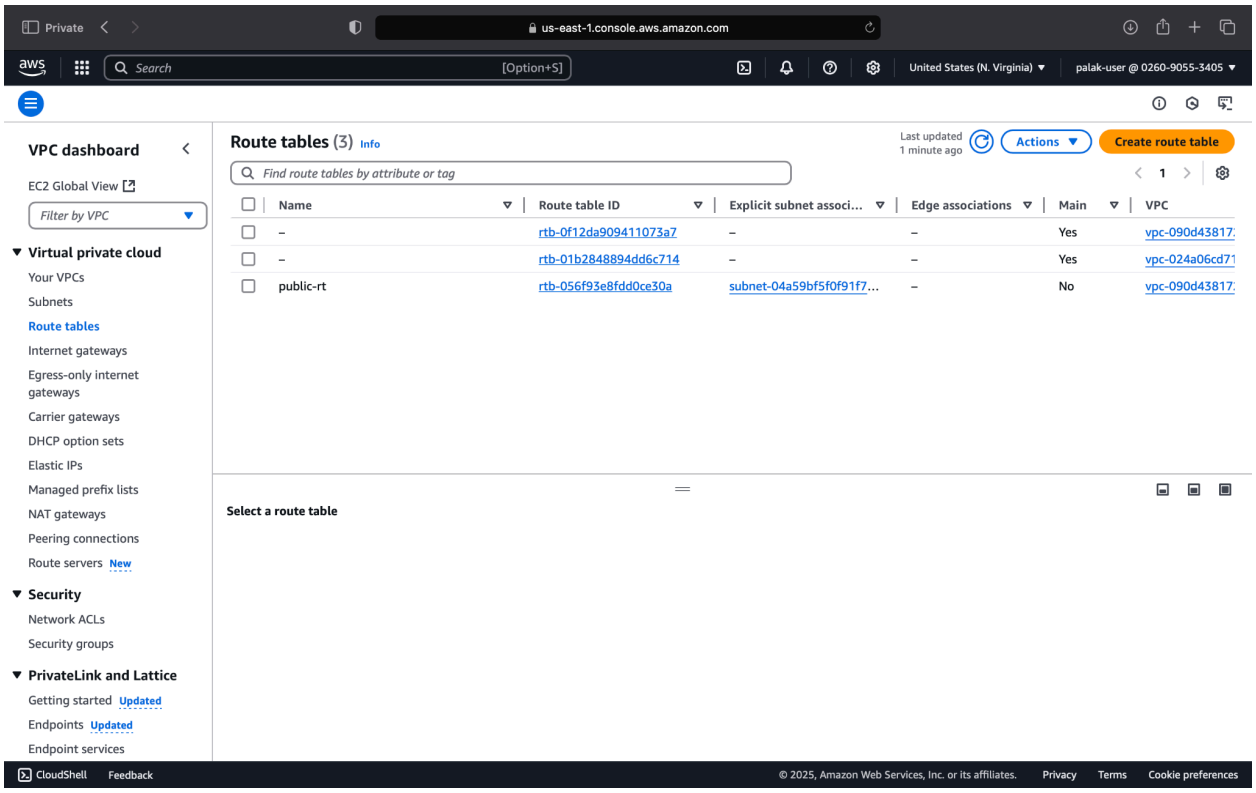
Subnet



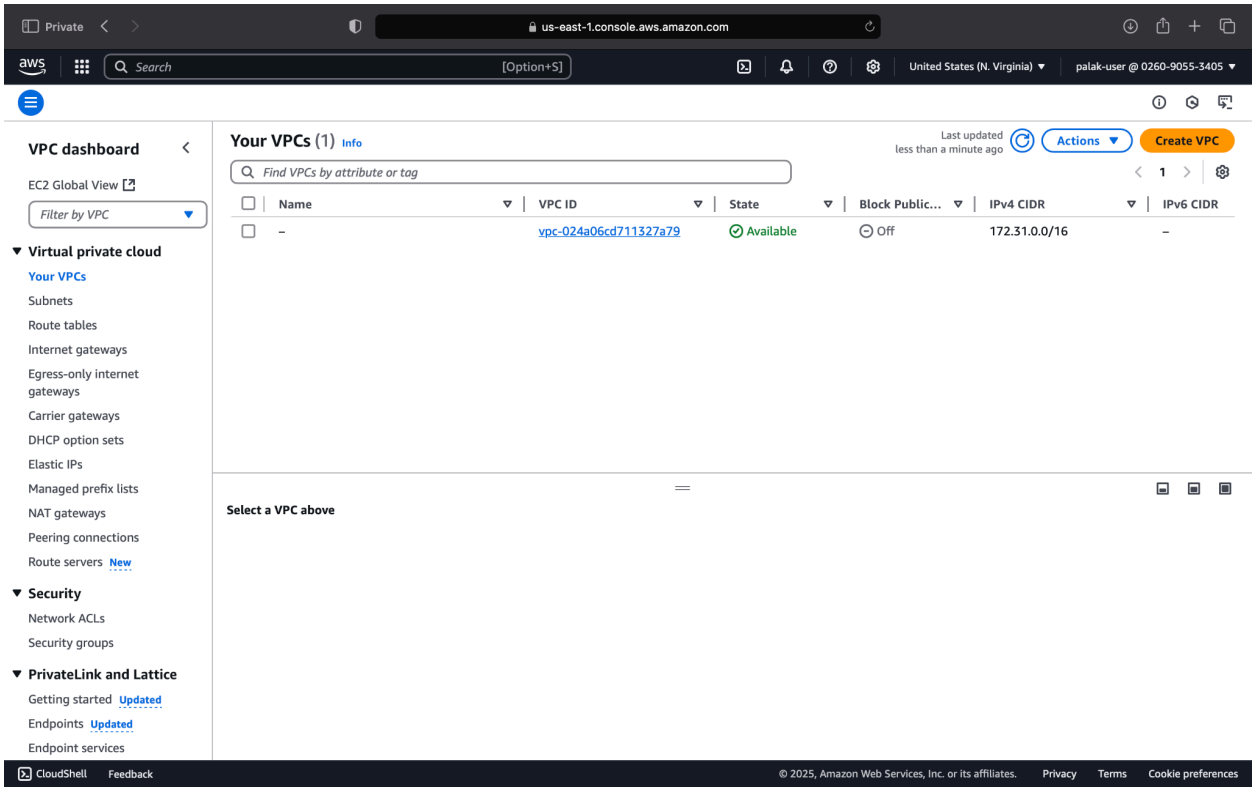
Internet gateway


Private
us-east-1.console.aws.amazon.com
Search
[Option+S]
United States (N. Virginia)
palak-user @ 0260-9055-3405
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
Network ACLs
Security groups
Internet gateways (2) Info
Find internet gateways by attribute or tag
Actions
Create internet gateway
Name
Internet gateway ID
State
VPC ID
Owner
–
igw-0257748389fa328d8
Attached
vpc-024a06cd711327a79
my-igw
igw-0c09a197f26820600
Attached
vpc-090d438172e7f7491 | my-vpc
Select an internet gateway above

Route table


Private
us-east-1.console.aws.amazon.com
Search
[Option+S]
United States (N. Virginia)
palak-user @ 0260-9055-3405
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
Network ACLs
Security groups
PrivateLink and Lattice
Getting started Updated
Endpoints Updated
Endpoint services
Route tables (3) Info
Last updated 1 minute ago
Actions
Create route table
Find route tables by attribute or tag
Name
Route table ID
Explicit subnet associ...
Edge associations
Main
VPC
–
rtb-0f12da909411073a7
Yes
–
rtb-01b2848894dd6c714
Yes
public-rt
rtb-056f93e8fdd0ce30a
subnet-04a59bf5f0f91f7...
No
Select a route table
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

```
  # aws_vpc.my_vpc will be destroyed
  - resource "aws_vpc" "my_vpc" {
      - arn                                  = "arn:aws:ec2:us-east-1:026090553405:vpc/vpc-090d438172e7f7491" -> null
      - assign_generated_ipv6_cidr_block     = false -> null
      - cidr_block                           = "10.0.0.0/16" -> null
      - default_network_acl_id               = "acl-03e468a0b88498025" -> null
      - default_route_table_id               = "rtb-0f12da909411073a7" -> null
      - default_security_group_id            = "sg-077df9ade9f65a5f2" -> null
      - dhcp_options_id                      = "dopt-0918aed1e65e814cc" -> null
      - enable_dns_hostnames                 = false -> null
      - enable_dns_support                   = true -> null
      - enable_network_address_usage_metrics = false -> null
      - id                                   = "vpc-090d438172e7f7491" -> null
      - instance_tenancy                     = "default" -> null
      - ipv6_netmask_length                  = 0 -> null
      - main_route_table_id                  = "rtb-0f12da909411073a7" -> null
      - owner_id                             = "026090553405" -> null
      - tags                                 = {
          - "Name" = "my-vpc"
        } -> null
      - tags_all                             = {
          - "Name" = "my-vpc"
        } -> null
        # (4 unchanged attributes hidden)
    }

Plan: 0 to add, 0 to change, 5 to destroy.
aws_route_table_association.public_association: Destroying... [id=rtbassoc-02b149137501ef980]
aws_route_table_association.public_association: Destruction complete after 2s
aws_route_table.public_rt: Destroying... [id=rtb-056f93e8fdd0ce30a]
aws_subnet.public_subnet: Destroying... [id=subnet-04a59bf5f0f91f75d]
aws_subnet.public_subnet: Destruction complete after 2s
aws_route_table.public_rt: Destruction complete after 2s
aws_internet_gateway.igw: Destroying... [id=igw-0c09a197f26820600]
aws_internet_gateway.igw: Destruction complete after 1s
aws_vpc.my_vpc: Destroying... [id=vpc-090d438172e7f7491]
aws_vpc.my_vpc: Destruction complete after 1s

Destroy complete! Resources: 5 destroyed.
palakgupta@Palaks-MacBook-Air vpn %
```



Step 3: Create a S3 Bucket

In the same project folder, create new main.tf and resource.tf .

In main.tf, keep the same provider block with access and secret keys from your IAM user.

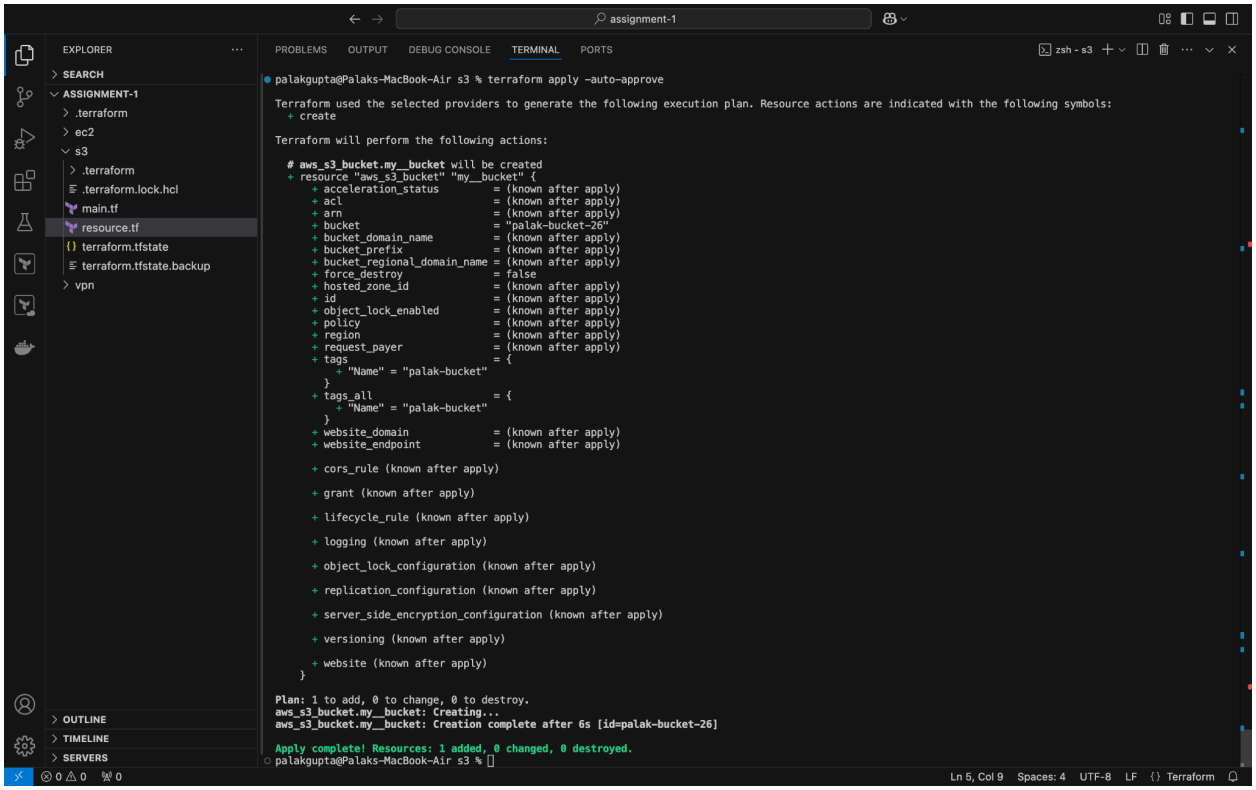
```
vpn > main.tf > provider "aws"
1   terraform {
2     required_providers {
3       aws = {
4         source = "hashicorp/aws"
5         version = "5.95.0"
6       }
7     }
8   }
9
10  provider "aws" {
11    access_key = "AKIAQMEY6IA6ULZVXAVS"
12    secret_key = "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"
13    region = "us-east-1"
14  }
```

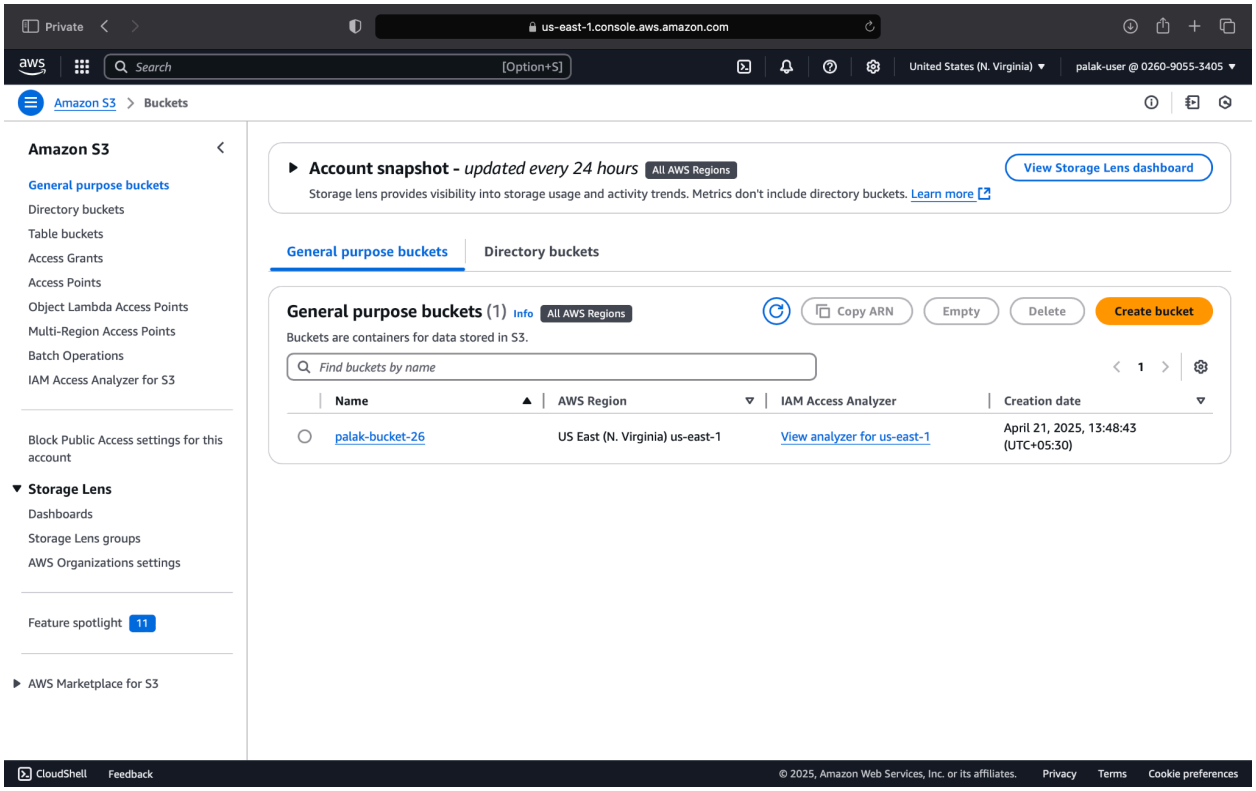
Define the aws_s3_bucket resource with a unique bucket name.

```
s3 > resource.tf > resource "aws_s3_bucket" "my__
1   resource "aws_s3_bucket" "my__bucket" {
2     bucket = "palak-bucket-26"
3
4     tags = {
5       Name = "palak-bucket"
6     }
7   }
```

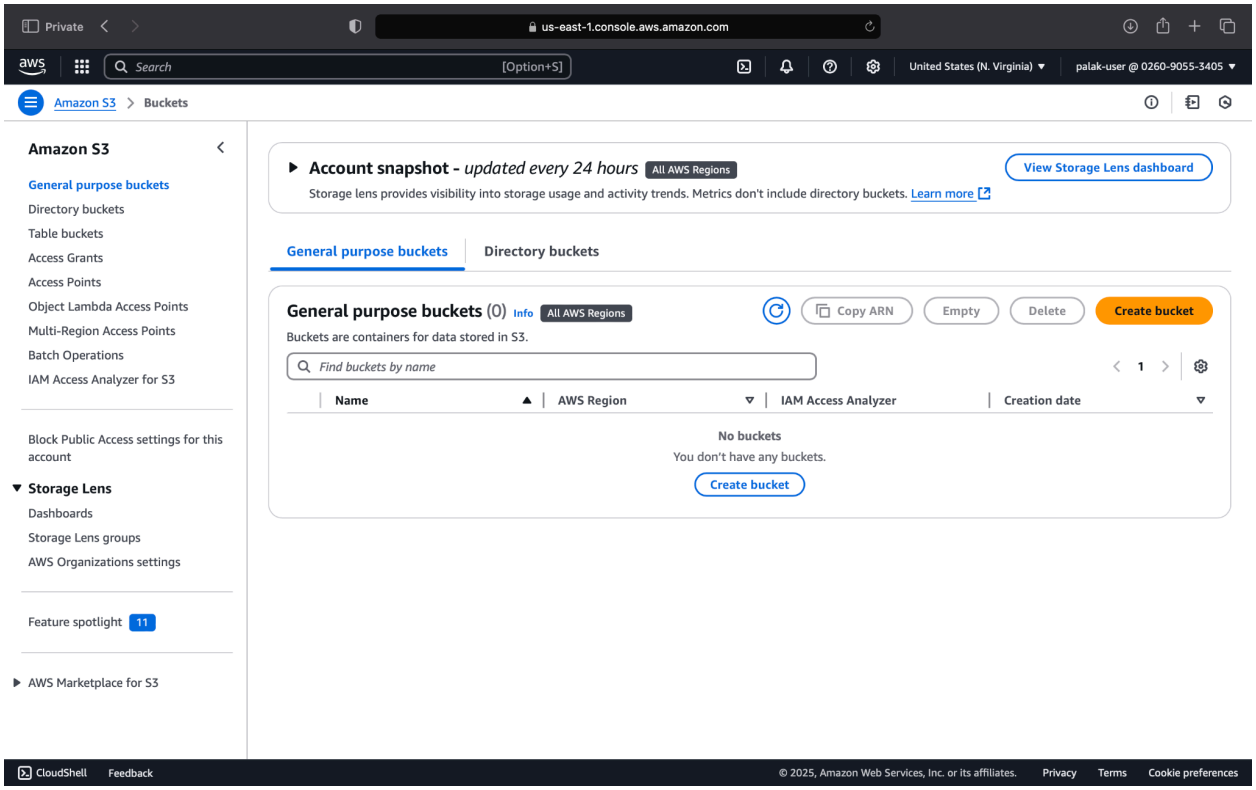
```
palakgupta@Palaks-MacBook-Air vpn % cd ..
palakgupta@Palaks-MacBook-Air assignment-1 % cd s3
palakgupta@Palaks-MacBook-Air s3 % terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.95.0"...
- Installing hashicorp/aws v5.95.0...
- Installed hashicorp/aws v5.95.0 (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.
palakgupta@Palaks-MacBook-Air s3 %
```

```
      + tags_all                    = {
          + "Name" = "palak-bucket"
        }
      + website_domain              = (known after apply)
      + website_endpoint            = (known after apply)

      + cors_rule (known after apply)

      + grant (known after apply)

      + lifecycle_rule (known after apply)

      + logging (known after apply)

      + object_lock_configuration (known after apply)

      + replication_configuration (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.

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these
palakgupta@Palaks-MacBook-Air s3 %
```


assignment-1
EXPLORER
SEARCH
ASSIGNMENT-1
.terraform
ec2
s3
.terraform
.terraform.lock.hcl
main.tf
resource.tf
terraform.tfstate
terraform.tfstate.backup
vpn
OUTLINE
TIMELINE
SERVERS
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
zsh - s3
palakgupta@Palaks-MacBook-Air s3 % terraform apply -auto-approve
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_s3_bucket.my__bucket will be created
+ resource "aws_s3_bucket" "my__bucket" {
+ acceleration_status = (known after apply)
+ acl = (known after apply)
+ arn = (known after apply)
+ bucket = "palak-bucket-26"
+ bucket_domain_name = (known after apply)
+ bucket_prefix = (known after apply)
+ bucket_regional_domain_name = (known after apply)
+ force_destroy = false
+ hosted_zone_id = (known after apply)
+ id = (known after apply)
+ object_lock_enabled = (known after apply)
+ policy = (known after apply)
+ region = (known after apply)
+ request_payer = (known after apply)
+ tags = {
+ "Name" = "palak-bucket"
}
+ tags_all = {
+ "Name" = "palak-bucket"
}
+ website_domain = (known after apply)
+ website_endpoint = (known after apply)
+ cors_rule (known after apply)
+ grant (known after apply)
+ lifecycle_rule (known after apply)
+ logging (known after apply)
+ object_lock_configuration (known after apply)
+ replication_configuration (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.
aws_s3_bucket.my__bucket: Creating...
aws_s3_bucket.my__bucket: Creation complete after 6s [id=palak-bucket-26]
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
palakgupta@Palaks-MacBook-Air s3 %
Ln 5, Col 9
Spaces: 4
UTF-8
LF
Terraform


Private
us-east-1.console.aws.amazon.com
Search
[Option+S]
United States (N. Virginia)
palak-user @ 0260-9055-3405
Amazon S3 > Buckets
Amazon S3
General purpose buckets
Directory buckets
Table buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings
Feature spotlight 11
AWS Marketplace for S3
Account snapshot - updated every 24 hours
All AWS Regions
Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. Learn more
View Storage Lens dashboard
General purpose buckets
Directory buckets
General purpose buckets (1) Info
All AWS Regions
Buckets are containers for data stored in S3.
Copy ARN
Empty
Delete
Create bucket
Find buckets by name
Name
AWS Region
IAM Access Analyzer
Creation date
palak-bucket-26
US East (N. Virginia) us-east-1
View analyzer for us-east-1
April 21, 2025, 13:48:43 (UTC+05:30)
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

```
palakgupta@Palaks-MacBook-Air s3 % terraform destroy -auto-approve
aws_s3_bucket.my__bucket: Refreshing state... [id=palak-bucket-26]

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_s3_bucket.my__bucket will be destroyed
  - resource "aws_s3_bucket" "my__bucket" {
      - arn                         = "arn:aws:s3:::palak-bucket-26" -> null
      - bucket                      = "palak-bucket-26" -> null
      - bucket_domain_name          = "palak-bucket-26.s3.amazonaws.com" -> null
      - bucket_regional_domain_name = "palak-bucket-26.s3.us-east-1.amazonaws.com" -> null
      - force_destroy               = false -> null
      - hosted_zone_id              = "Z3AQBSTGFYJSTF" -> null
      - id                          = "palak-bucket-26" -> null
      - object_lock_enabled         = false -> null
      - region                      = "us-east-1" -> null
      - request_payer               = "BucketOwner" -> null
      - tags                        = {
          - "Name" = "palak-bucket"
        } -> null
      - tags_all                    = {
          - "Name" = "palak-bucket"
        } -> null
        # (3 unchanged attributes hidden)

      - grant {
          - id          = "6105aafe8facbce97c5480b1d597043abce631c0930d1ff236ff2ce9b3a5f661" -> null
          - permissions = [
              - "FULL_CONTROL",
            ] -> null
          - type        = "CanonicalUser" -> null
            # (1 unchanged attribute hidden)
        }

      - server_side_encryption_configuration {
          - rule {
              - bucket_key_enabled = false -> null

              - apply_server_side_encryption_by_default {
                  - sse_algorithm     = "AES256" -> null
                    # (1 unchanged attribute hidden)
                }
            }
        }

      - versioning {
          - enabled    = false -> null
          - mfa_delete = false -> null
        }
    }

Plan: 0 to add, 0 to change, 1 to destroy.
aws_s3_bucket.my__bucket: Destroying... [id=palak-bucket-26]
aws_s3_bucket.my__bucket: Destruction complete after 1s

Destroy complete! Resources: 1 destroyed.
palakgupta@Palaks-MacBook-Air s3 %
```


Private
us-east-1.console.aws.amazon.com
aws
Search
[Option+S]
United States (N. Virginia)
palak-user @ 0260-9055-3405
Amazon S3
Buckets
Amazon S3
General purpose buckets
Directory buckets
Table buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings
Feature spotlight
11
AWS Marketplace for S3
Account snapshot - updated every 24 hours
All AWS Regions
Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. Learn more
View Storage Lens dashboard
General purpose buckets
Directory buckets
General purpose buckets (0)
Info
All AWS Regions
Buckets are containers for data stored in S3.
Copy ARN
Empty
Delete
Create bucket
Find buckets by name
1
Name
AWS Region
IAM Access Analyzer
Creation date
No buckets
You don't have any buckets.
Create bucket
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences