Pimpri Chinchwad Education Trust’s
Pimpri Chinchwad College of Engineering
(PCCoE)
(An Autonomous Institute)
Affiliated to Savitribai Phule Pune
University(SPPU) ISO 21001:2018 Certified by
TUV

Course: DevOps Laboratory	**Code:** BIT26VS01
Name: Amar Vaijinath Chavan	**PRN:** 124B2F001
Assignment 14: Discover Infrastructure as Code using Terraform and write a Terraform script to create a virtual machine (EC2).	

Aim: To explore the principles of Infrastructure as Code (IaC) and automate the provisioning of an AWS EC2 instance using Terraform.

Objectives:

- To understand the core concepts of Infrastructure as Code (IaC) and the Terraform lifecycle.
- To develop a Terraform configuration for provisioning cloud infrastructure.
- To execute the workflow of init, plan, apply, and destroy to manage cloud resources.

Theory

1. Infrastructure as Code (IaC)

Infrastructure as Code is a key DevOps practice that involves managing and provisioning computing infrastructure through machine-readable definition files rather than physical hardware configuration or interactive configuration tools. IaC allows for consistency, repeatability, and version control of infrastructure, significantly reducing manual errors and deployment time.

2. Terraform Overview

Terraform is an open-source IaC tool created by HashiCorp. It allows users to define both cloud and on-premises resources in human-readable configuration files that can be versioned, shared, and reused. Terraform uses HashiCorp Configuration Language (HCL) to describe the desired "End State" of the infrastructure.

3. Terraform Architecture and Lifecycle

Terraform operates on a plugin-based architecture, utilizing **Providers** (like AWS, Azure, or GCP) to interact with remote APIs. The standard lifecycle consists of:

- **terraform init**: Initializes the working directory and downloads necessary provider plugins.

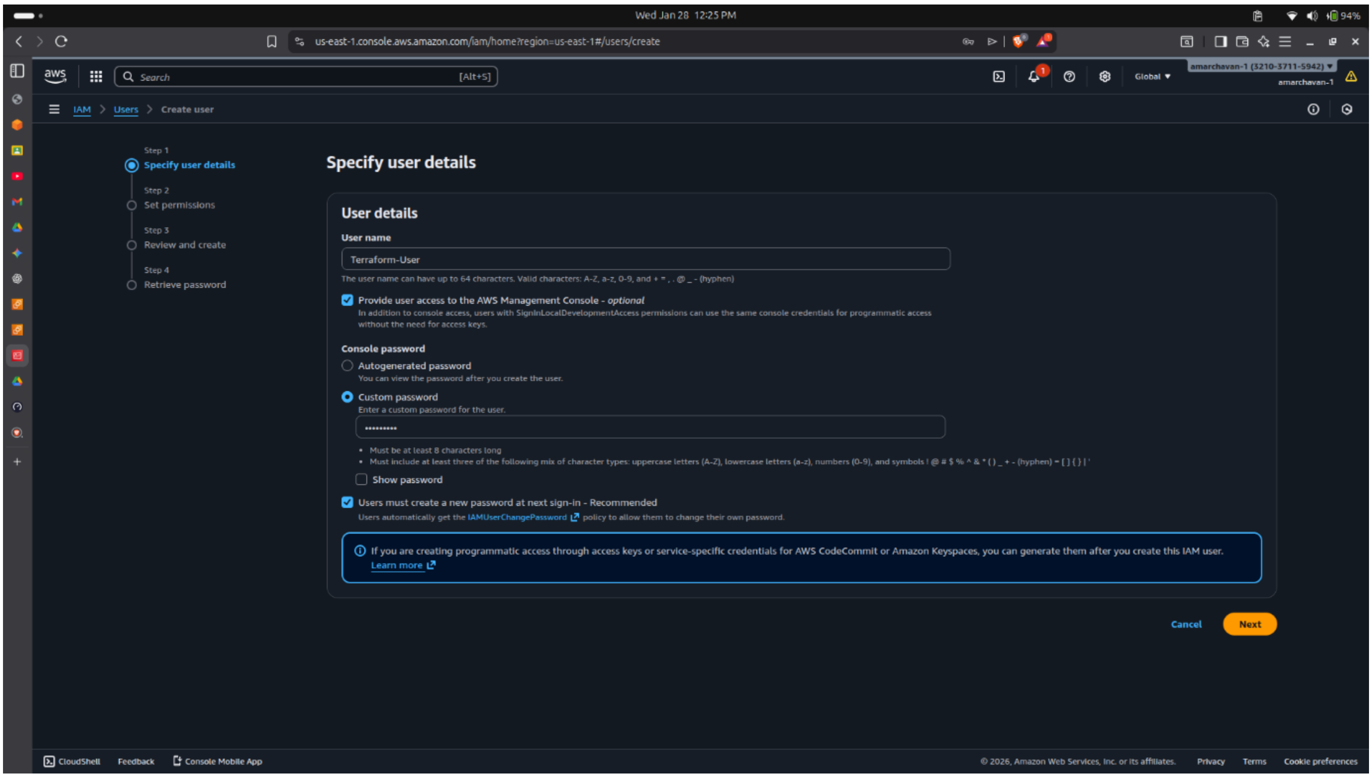
- **terraform plan**: Creates an execution plan, showing what actions Terraform will take to reach the desired state.
- **terraform apply**: Executes the proposed plan to create or modify infrastructure.
- **terraform destroy**: Safely removes all managed infrastructure defined in the configuration.

4. **State Management**
   Terraform maintains a **State File** (terraform.tfstate) that acts as a source of truth, mapping the configuration to the real-world resources. This file allows Terraform to determine which changes are needed when the configuration is updated.

Practical Procedure / Steps

Step 1: Configure AWS CLI & Programmatic Access Before executing Terraform, the local environment must be authenticated with AWS.

- **Create IAM User**: A new user named Terraform-User was created in the AWS Console.
- **Attach Permissions**: The AdministratorAccess policy was attached directly to ensure the user has rights to manage EC2 and Security Groups.
- **Generate Access Keys**: An Access Key and Secret Access Key were generated for Command Line Interface (CLI) use.
- **AWS Configure**: On the local Ubuntu machine, the command aws configure was used to input the Access Key ID, Secret Access Key, and set the default region to us-east-1.



Specify user details

User details

User name

Terraform-User

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)

☑ Provide user access to the AWS Management Console - *optional*
In addition to console access, users with SignInLocalDevelopmentAccess permissions can use the same console credentials for programmatic access without the need for access keys.

Console password

○ Autogenerated password
You can view the password after you create the user.

● Custom password
Enter a custom password for the user.

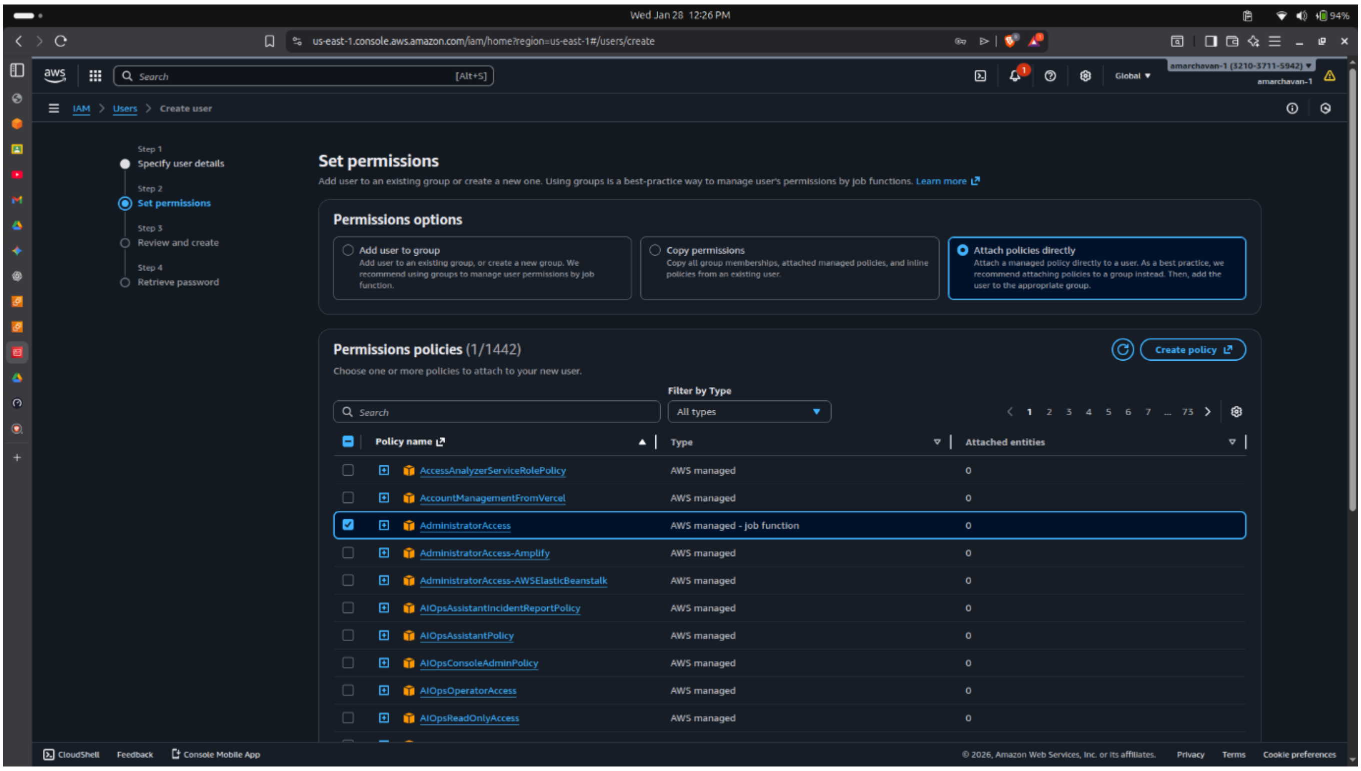
•••••••••

- Must be at least 8 characters long
- Must include at least three of the following mix of character types: uppercase letters (A-Z), lowercase letters (a-z), numbers (0-9), and symbols ! @ # $ % ^ & * ( ) _ + - (hyphen) = [ ] { } | '

□ Show password

☑ Users must create a new password at next sign-in - Recommended
Users automatically get the IAMUserChangePassword policy to allow them to change their own password.

If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user.
Learn more


Wed Jan 28 12:26 PM
94%
us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/users/create
aws
Search [Alt+S]
Global ▾
amarchavan-1 (3210-3711-5942) ▾
amarchavan-1
IAM > Users > Create user
CloudShell
Feedback
Console Mobile App
© 2026, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Step 1
● Specify user details

Step 2
◉ Set permissions

Step 3
○ Review and create

Step 4
○ Retrieve password

Set permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. Learn more

Permissions options

○ Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.

○ Copy permissions
Copy all group memberships, attached managed policies, and inline policies from an existing user.

● Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

Permissions policies (1/1442)

Create policy

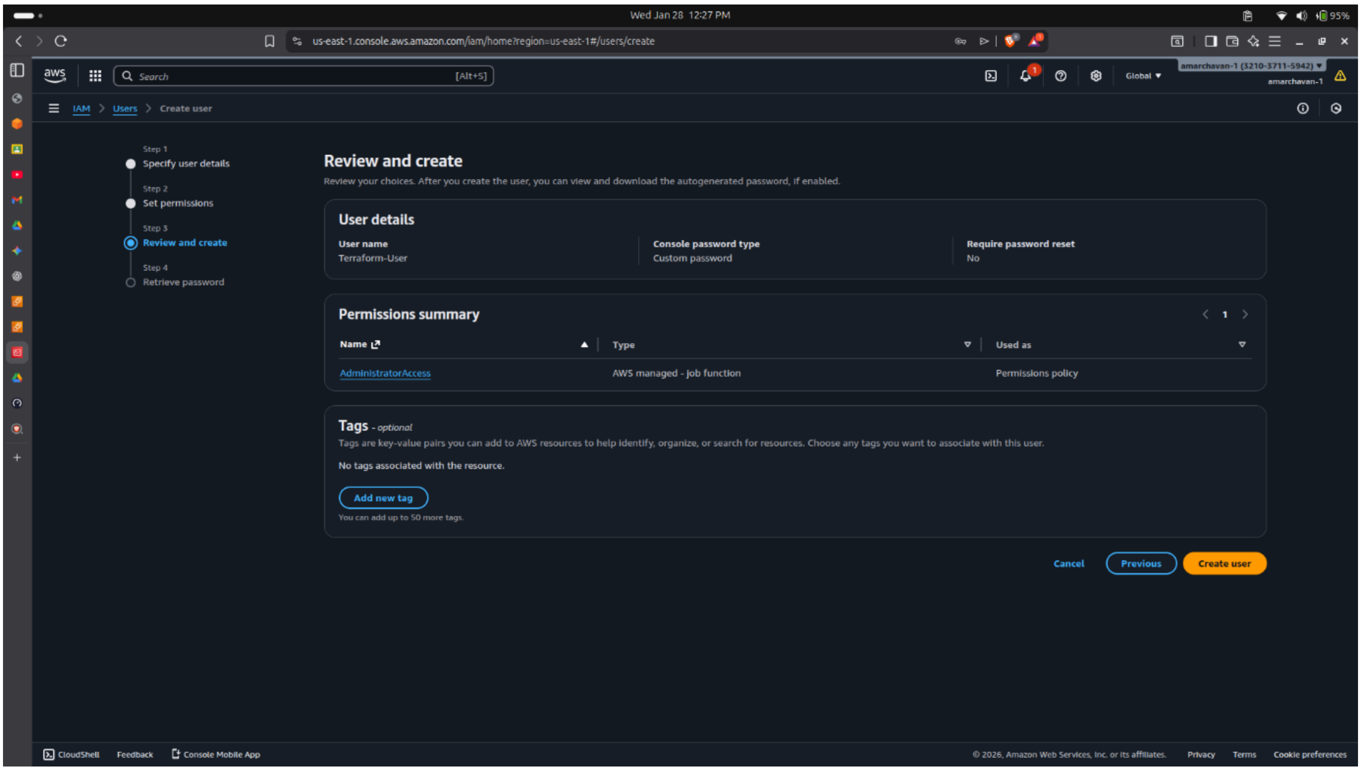
Choose one or more policies to attach to your new user.

Search

Filter by Type: All types ▾

< 1 2 3 4 5 6 7 ... 73 >

☐	Policy name	Type	Attached entities
☐	AccessAnalyzerServiceRolePolicy	AWS managed	0
☐	AccountManagementFromVercel	AWS managed	0
☑	AdministratorAccess	AWS managed - job function	0
☐	AdministratorAccess-Amplify	AWS managed	0
☐	AdministratorAccess-AWSElasticBeanstalk	AWS managed	0
☐	AIOpsAssistantIncidentReportPolicy	AWS managed	0
☐	AIOpsAssistantPolicy	AWS managed	0
☐	AIOpsConsoleAdminPolicy	AWS managed	0
☐	AIOpsOperatorAccess	AWS managed	0
☐	AIOpsReadOnlyAccess	AWS managed	0


Wed Jan 28 12:27 PM
95%
us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/users/create
aws
Search [Alt+S]
Global ▾
amarchavan-1 (3210-3711-5942) ▾
amarchavan-1
IAM > Users > Create user
CloudShell
Feedback
Console Mobile App
© 2026, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Step 1
● Specify user details

Step 2
● Set permissions

Step 3
◉ Review and create

Step 4
○ Retrieve password

Review and create

Review your choices. After you create the user, you can view and download the autogenerated password, if enabled.

User details

User name	Console password type	Require password reset
Terraform-User	Custom password	No

Permissions summary

< 1 >

Name	Type	Used as
AdministratorAccess	AWS managed - job function	Permissions policy

Tags - *optional*

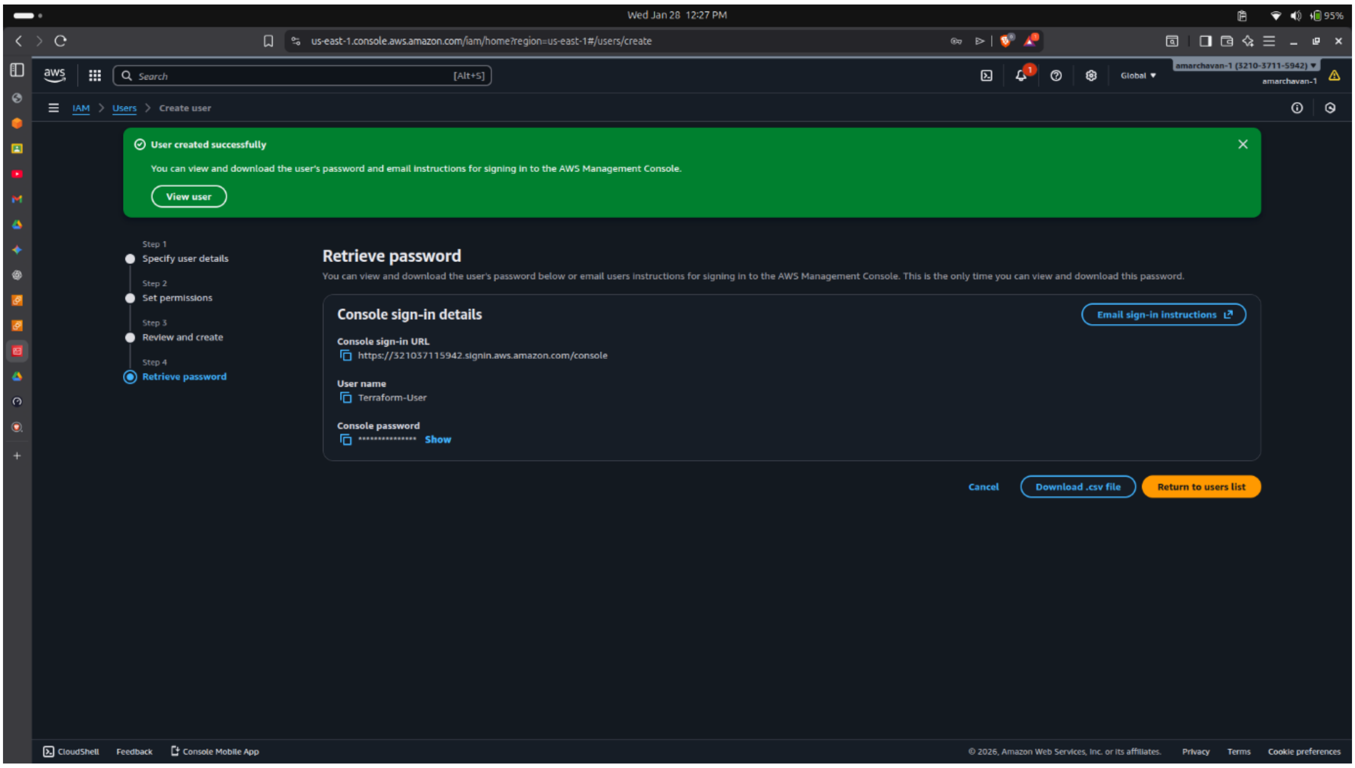
Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this user.

No tags associated with the resource.

Add new tag

You can add up to 50 more tags.

Cancel Previous Create user



User created successfully

You can view and download the user's password and email instructions for signing in to the AWS Management Console.

Retrieve password

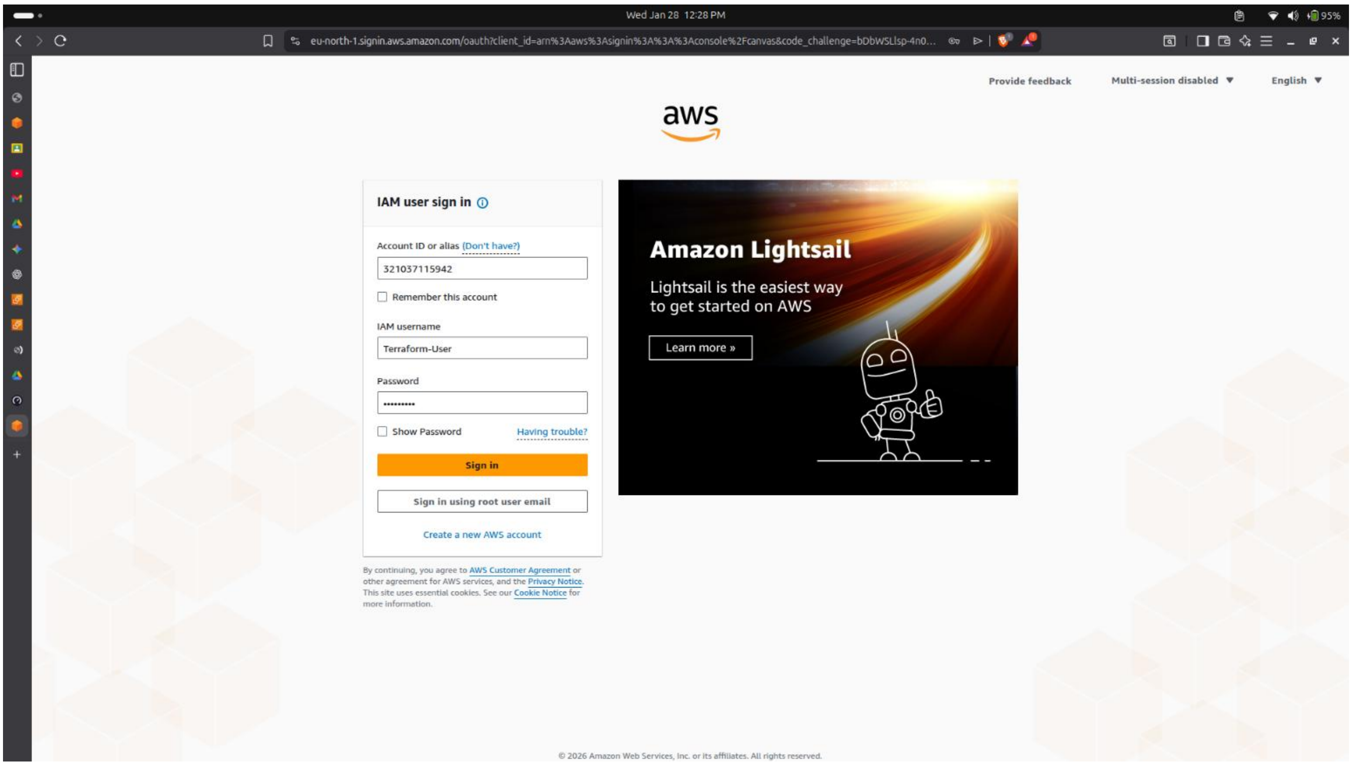
You can view and download the user's password below or email users instructions for signing in to the AWS Management Console. This is the only time you can view and download this password.

Console sign-in details

Console sign-in URL
https://321037115942.signin.aws.amazon.com/console

User name
Terraform-User

Console password
****



Wed Jan 28 12:28 PM
eu-north-1.signin.aws.amazon.com/oauth?client_id=arn%3Aaws%3Asignin%3A%3A%3Aconsole%2Fcanvas&code_challenge=bDbWSLlsp-4n0...
Provide feedback
Multi-session disabled ▾
English ▾
aws
Having trouble?
Sign in
Sign in using root user email
Create a new AWS account
Amazon Lightsail
Lightsail is the easiest way to get started on AWS
Learn more »
© 2026 Amazon Web Services, Inc. or its affiliates. All rights reserved.

IAM user sign in

Account ID or alias (Don't have?)
321037115942

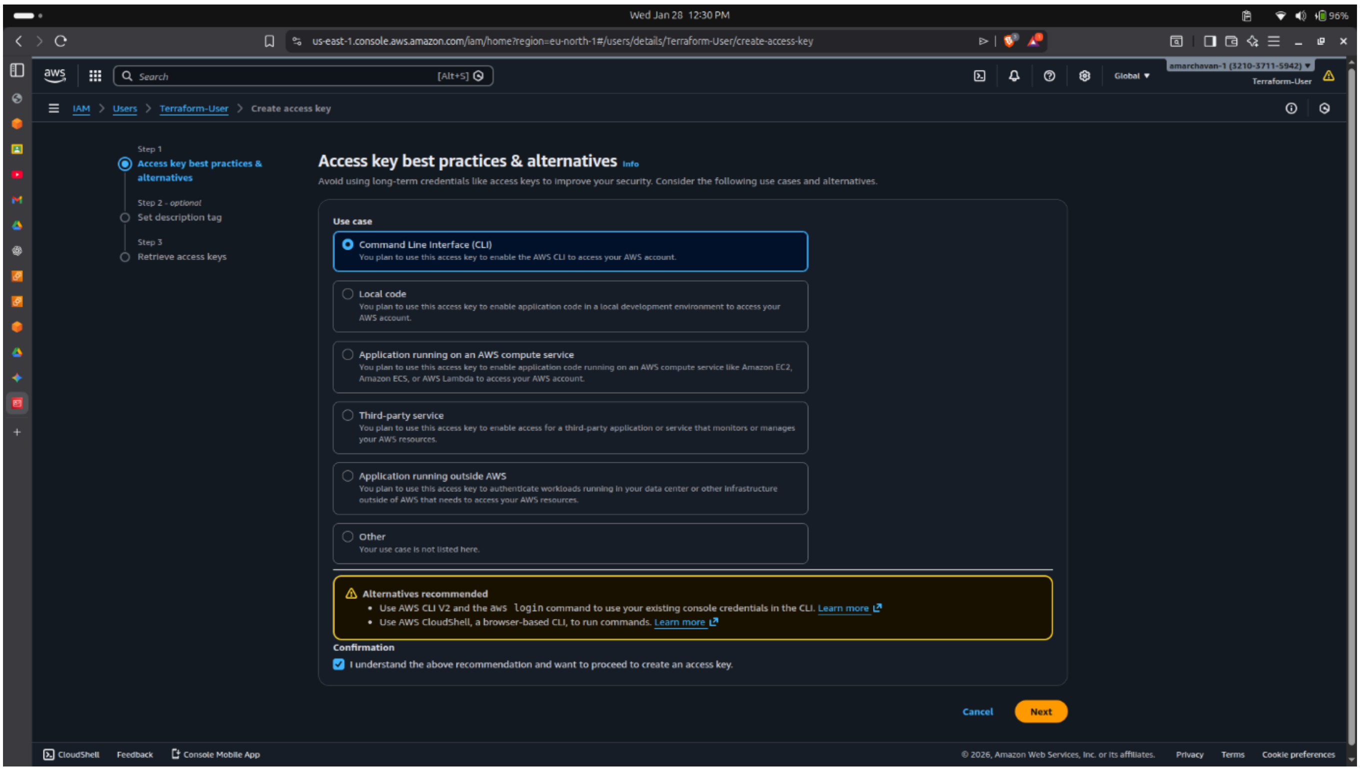
□ Remember this account

IAM username
Terraform-User

Password
••••••••••

□ Show Password

By continuing, you agree to AWS Customer Agreement or other agreement for AWS services, and the Privacy Notice. This site uses essential cookies. See our Cookie Notice for more information.



us-east-1.console.aws.amazon.com/iam/home?region=eu-north-1#/users/details/Terraform-User/create-access-key

IAM > Users > Terraform-User > Create access key

Step 1
● Access key best practices & alternatives

Step 2 - *optional*
○ Set description tag

Step 3
○ Retrieve access keys

Access key best practices & alternatives Info

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

Use case

● Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.

○ Local code
You plan to use this access key to enable application code in a local development environment to access your AWS account.

○ Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

○ Third-party service
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.

○ Application running outside AWS
You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.

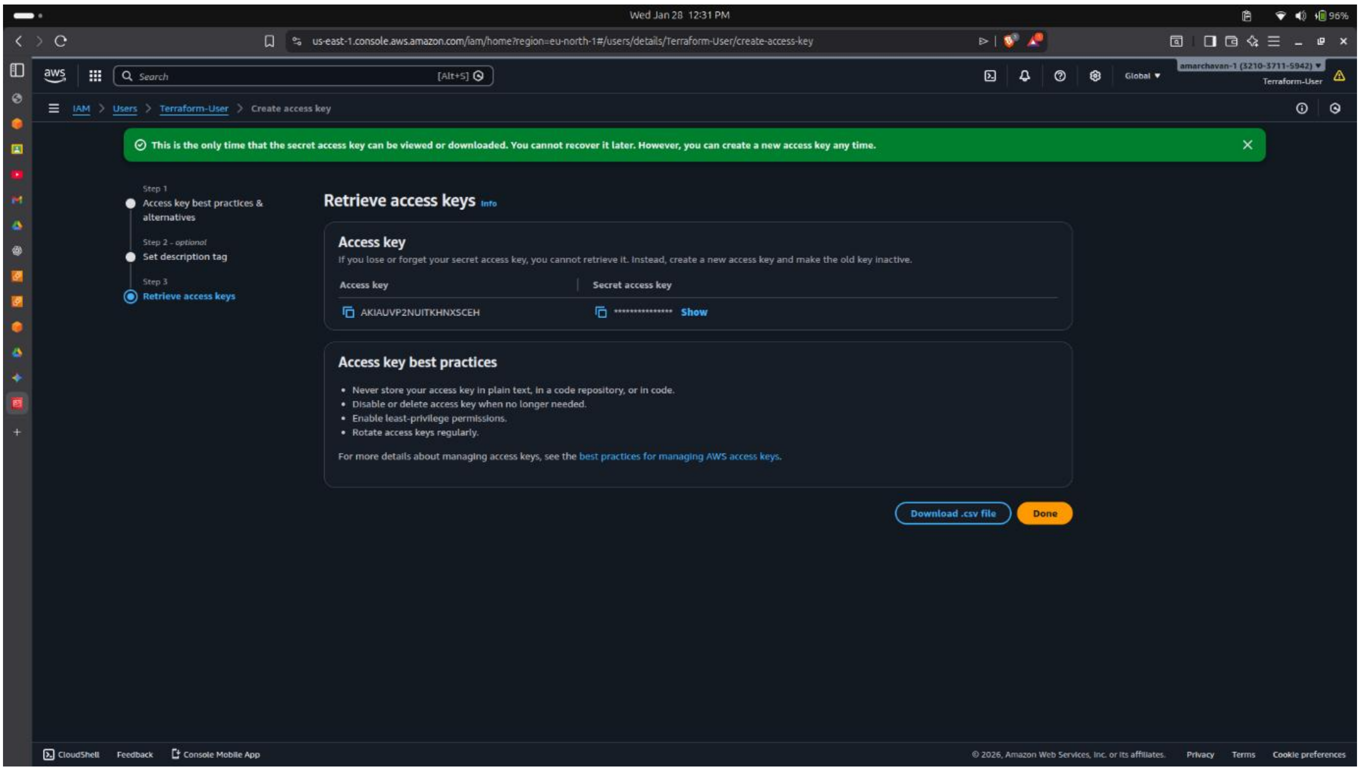
○ Other
Your use case is not listed here.

Alternatives recommended
- Use AWS CLI V2 and the `aws login` command to use your existing console credentials in the CLI. Learn more
- Use AWS CloudShell, a browser-based CLI, to run commands. Learn more

Confirmation

☑ I understand the above recommendation and want to proceed to create an access key.

Cancel Next



us-east-1.console.aws.amazon.com/iam/home?region=eu-north-1#/users/details/Terraform-User/create-access-key

IAM > Users > Terraform-User > Create access key

This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.

Step 1
● Access key best practices & alternatives

Step 2 - *optional*
● Set description tag

Step 3
● Retrieve access keys

Retrieve access keys Info

Access key

If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

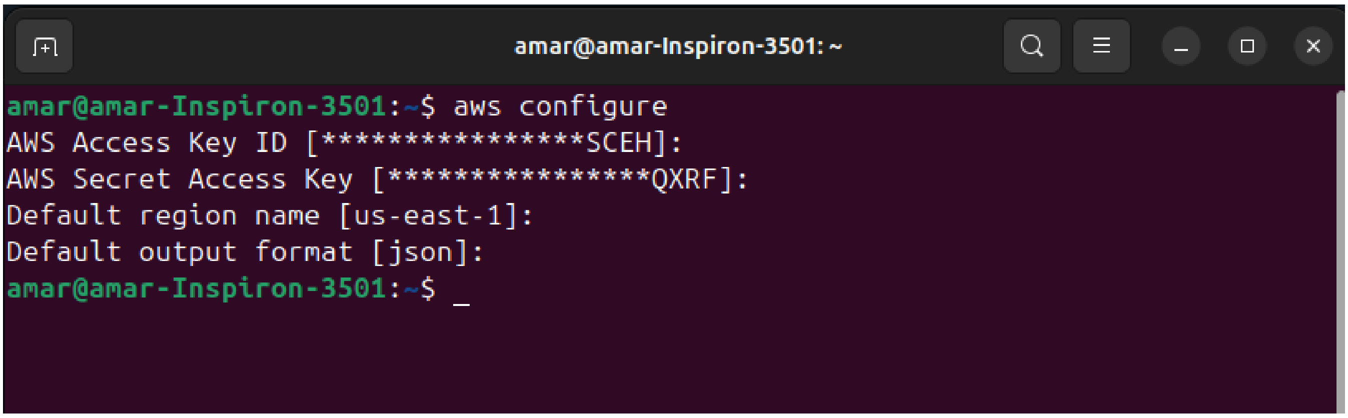
Access key	Secret access key
AKIAUVP2NUITKHNXSCEH	*************** Show

Access key best practices

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the best practices for managing AWS access keys.

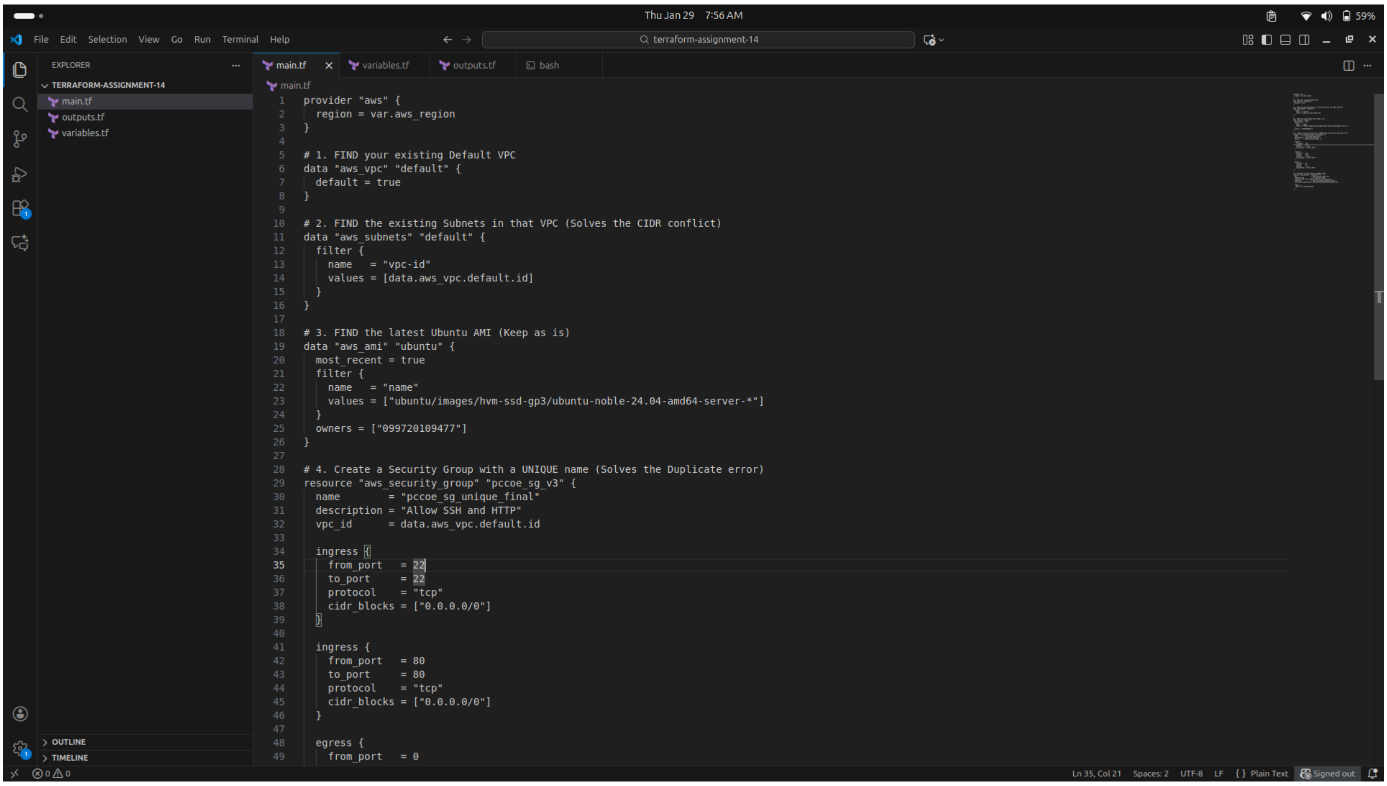
Download .csv file Done



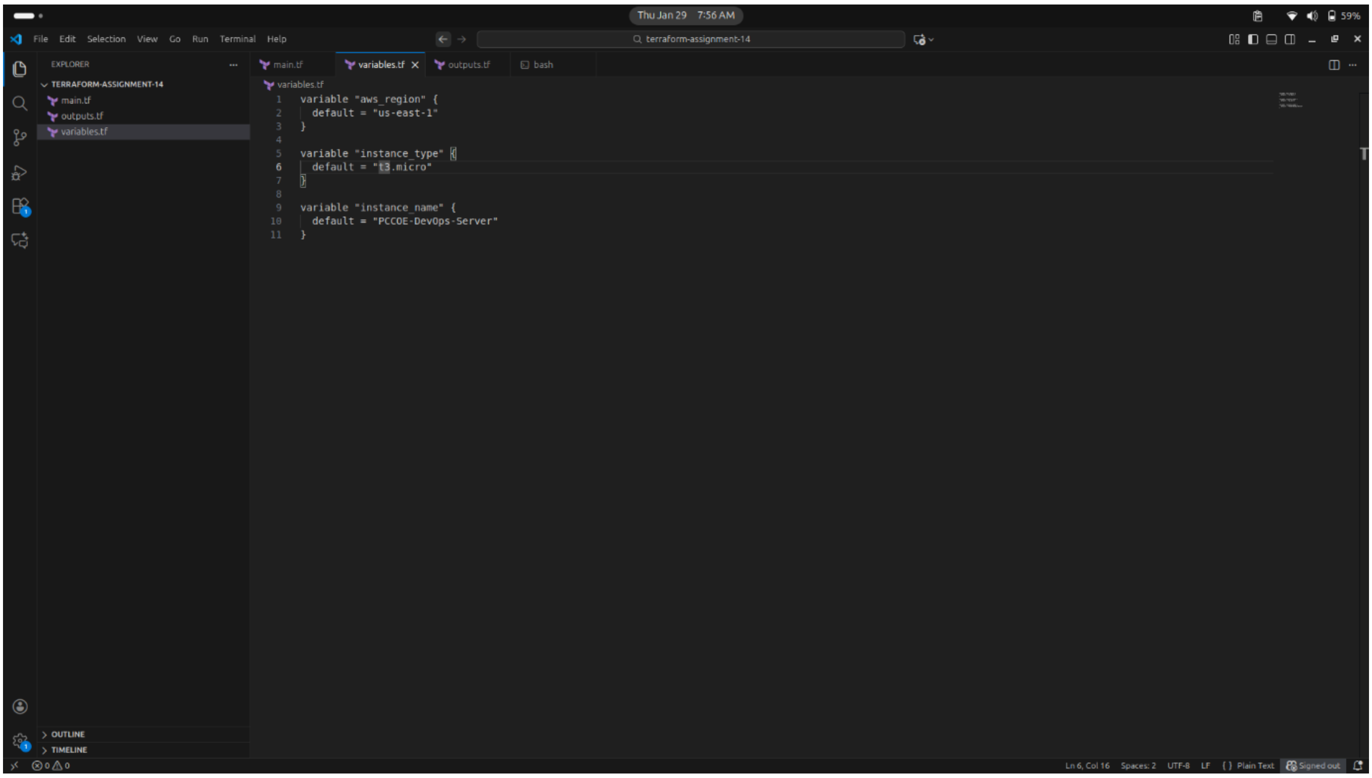
```
amar@amar-Inspiron-3501:~$ aws configure
AWS Access Key ID [****************SCEH]:
AWS Secret Access Key [****************QXRF]:
Default region name [us-east-1]:
Default output format [json]:
amar@amar-Inspiron-3501:~$ _
```

Step 2: Terraform Initialization

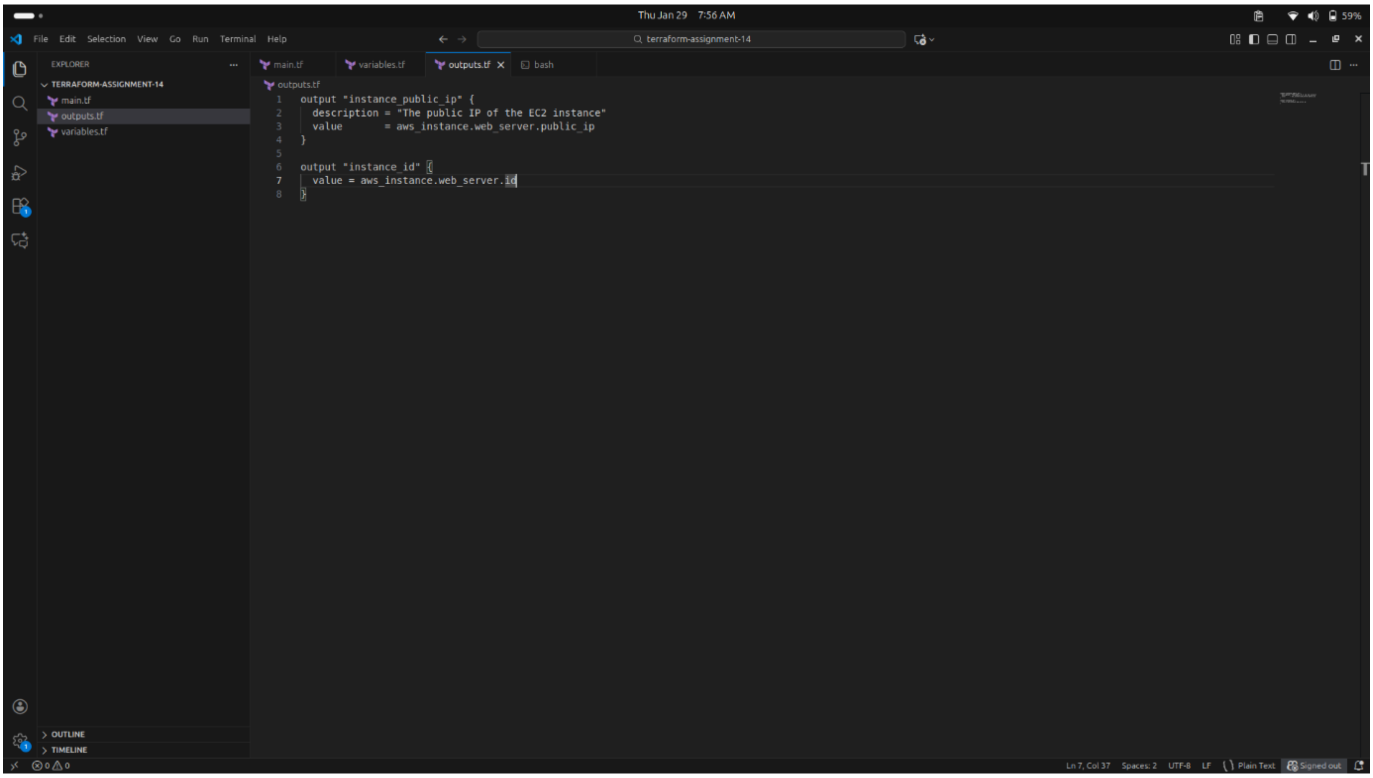
- Navigate to the project directory terraform-assignment-14.
- Execute terraform init to initialize the backend and download the HashiCorp AWS provider plugin (version v6.29.0).



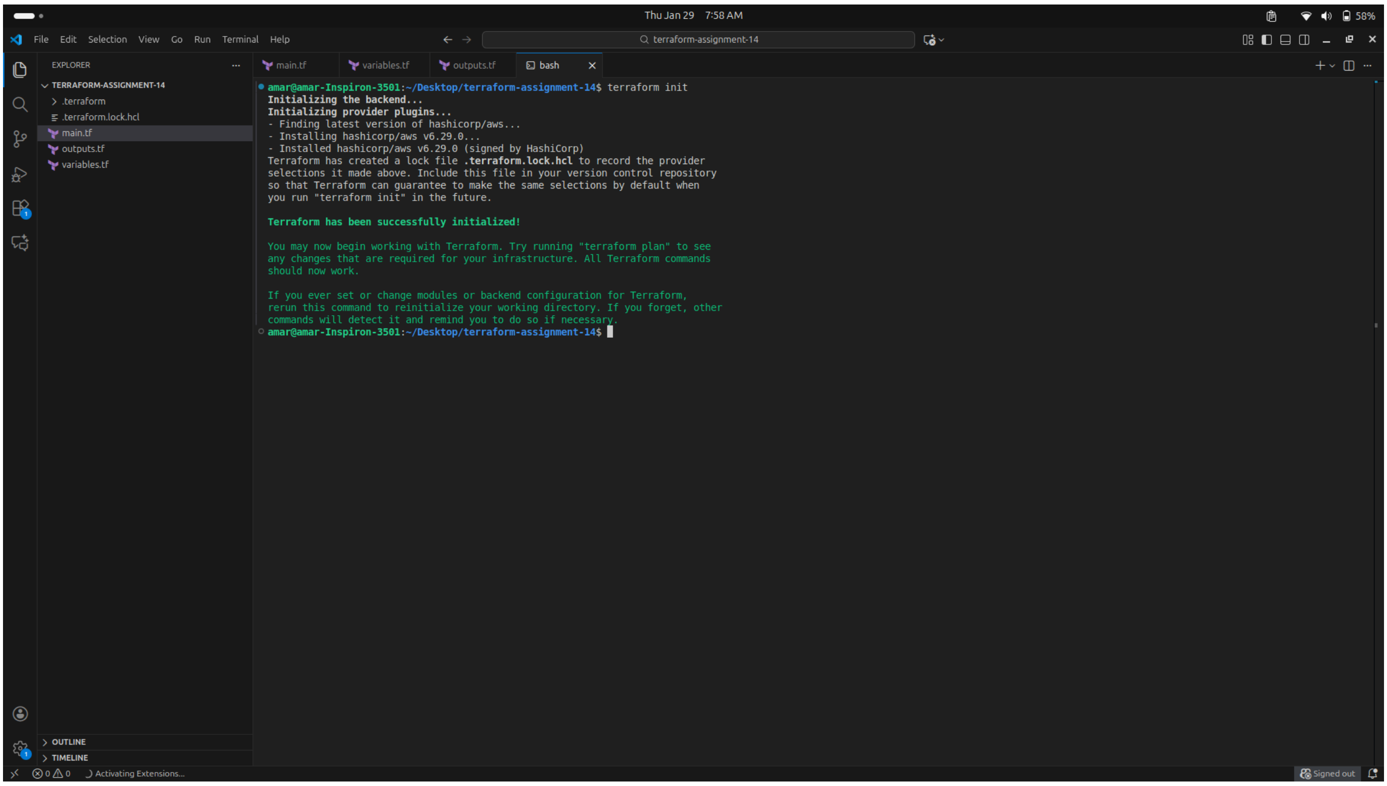
```
provider "aws" {
  region = var.aws_region
}

# 1. FIND your existing Default VPC
data "aws_vpc" "default" {
  default = true
}

# 2. FIND the existing Subnets in that VPC (Solves the CIDR conflict)
data "aws_subnets" "default" {
  filter {
    name   = "vpc-id"
    values = [data.aws_vpc.default.id]
  }
}

# 3. FIND the latest Ubuntu AMI (Keep as is)
data "aws_ami" "ubuntu" {
  most_recent = true
  filter {
    name   = "name"
    values = ["ubuntu/images/hvm-ssd-gp3/ubuntu-noble-24.04-amd64-server-*"]
  }
  owners = ["099720109477"]
}

# 4. Create a Security Group with a UNIQUE name (Solves the Duplicate error)
resource "aws_security_group" "pccoe_sg_v3" {
  name        = "pccoe_sg_unique_final"
  description = "Allow SSH and HTTP"
  vpc_id      = data.aws_vpc.default.id

  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  egress {
    from_port   = 0
```



```
variable "aws_region" {
  default = "us-east-1"
}

variable "instance_type" {
  default = "t3.micro"
}

variable "instance_name" {
  default = "PCCOE-DevOps-Server"
}
```



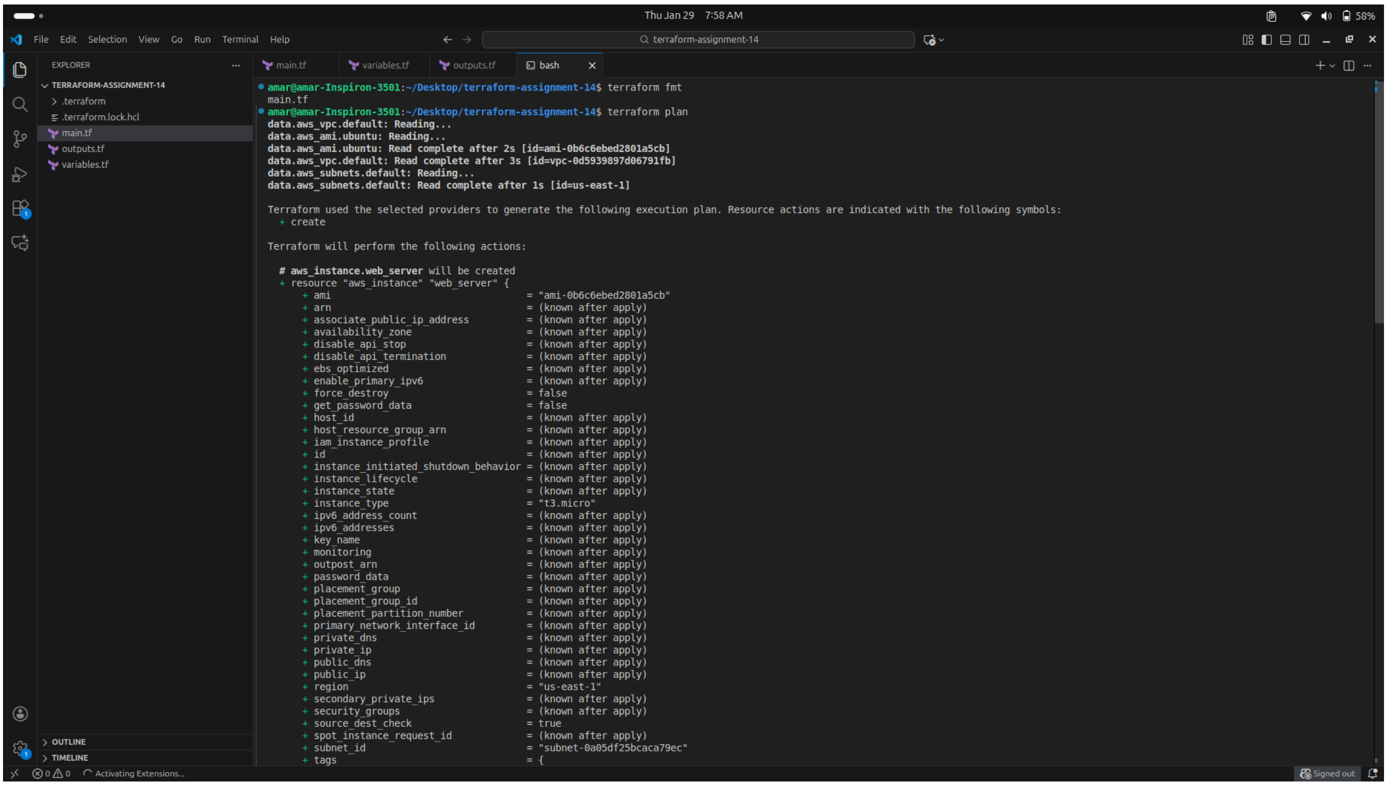
```
output "instance_public_ip" {
  description = "The public IP of the EC2 instance"
  value       = aws_instance.web_server.public_ip
}

output "instance_id" {
  value = aws_instance.web_server.id
}
```



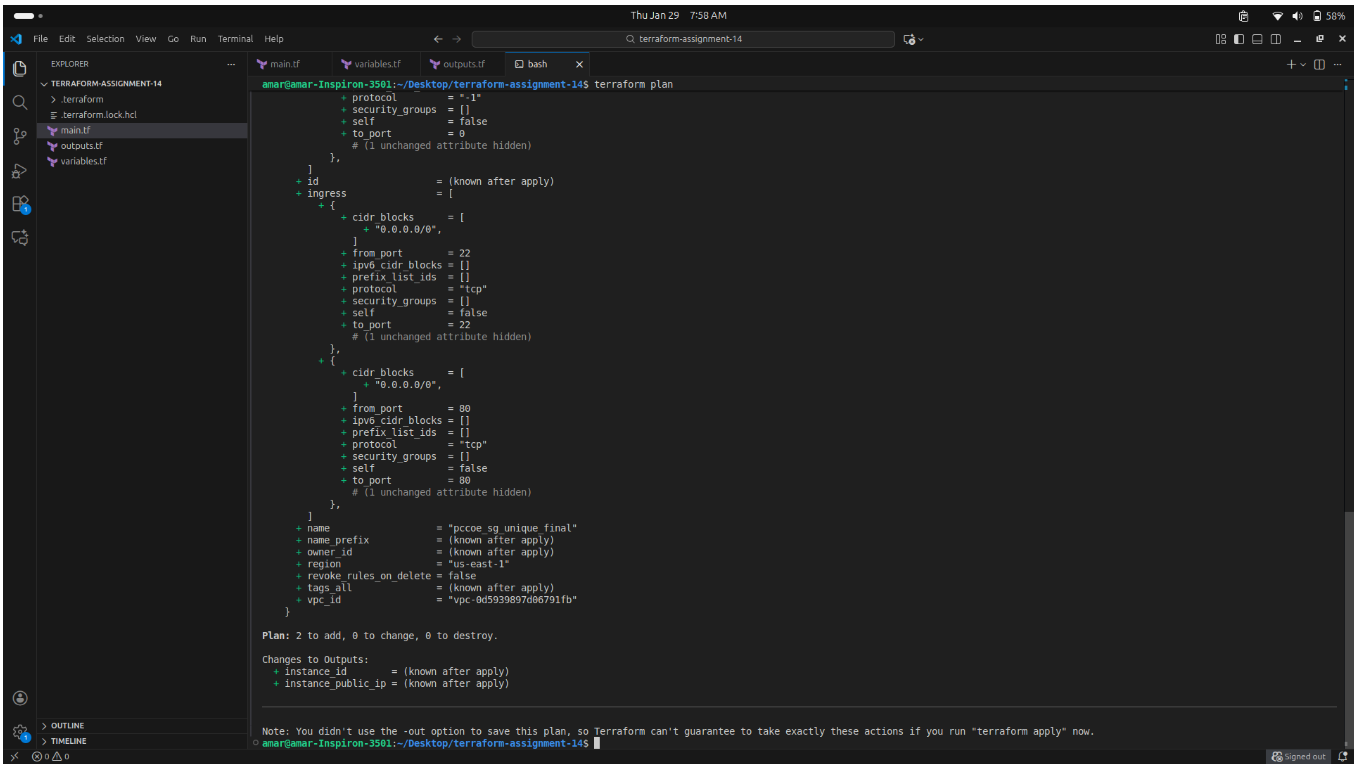
```
amar@amar-Inspiron-3501:~/Desktop/terraform-assignment-14$ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.29.0...
- Installed hashicorp/aws v6.29.0 (signed by HashiCorp)
Terraform has created a 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.
amar@amar-Inspiron-3501:~/Desktop/terraform-assignment-14$
```

Step 3: Planning the Infrastructure

- Run terraform plan to generate an execution plan.
- The plan confirmed that **2 resources** (an AWS Instance and a Security Group) would be added.



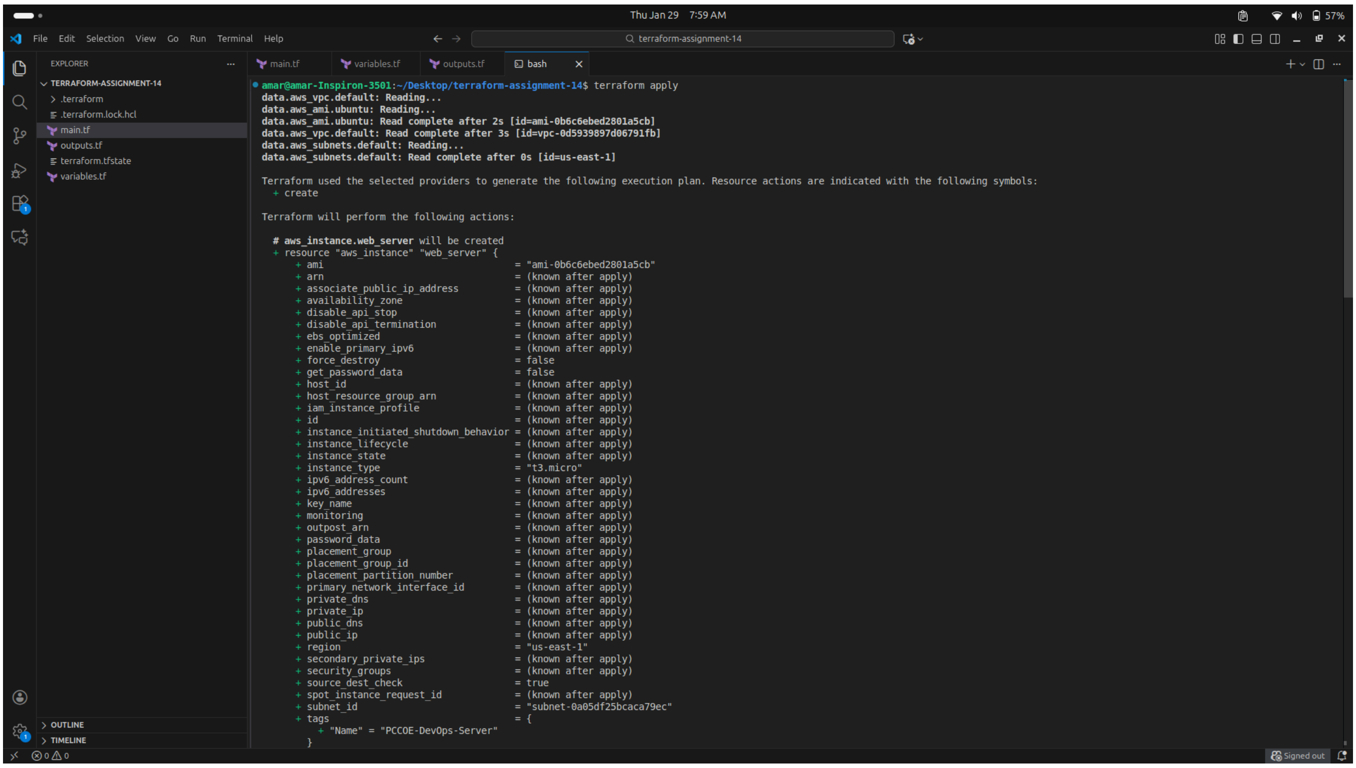
```
amar@amar-Inspiron-3501:~/Desktop/terraform-assignment-14$ terraform fmt
main.tf
amar@amar-Inspiron-3501:~/Desktop/terraform-assignment-14$ terraform plan
data.aws_vpc.default: Reading...
data.aws_ami.ubuntu: Reading...
data.aws_ami.ubuntu: Read complete after 2s [id=ami-0b6c6ebed2801a5cb]
data.aws_vpc.default: Read complete after 3s [id=vpc-0d5939897d06791fb]
data.aws_subnets.default: Reading...
data.aws_subnets.default: Read complete after 1s [id=us-east-1]

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.web_server will be created
  + resource "aws_instance" "web_server" {
      + ami                                  = "ami-0b6c6ebed2801a5cb"
      + 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)
      + key_name                             = (known after apply)
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
      + placement_group                      = (known after apply)
      + placement_group_id                   = (known after apply)
      + placement_partition_number           = (known after apply)
      + primary_network_interface_id         = (known after apply)
      + private_dns                          = (known after apply)
      + private_ip                           = (known after apply)
      + public_dns                           = (known after apply)
      + public_ip                            = (known after apply)
      + region                               = "us-east-1"
      + secondary_private_ips                = (known after apply)
      + security_groups                      = (known after apply)
      + source_dest_check                    = true
      + spot_instance_request_id             = (known after apply)
      + subnet_id                            = "subnet-0a05df25bcaca79ec"
      + tags                                 = {
```



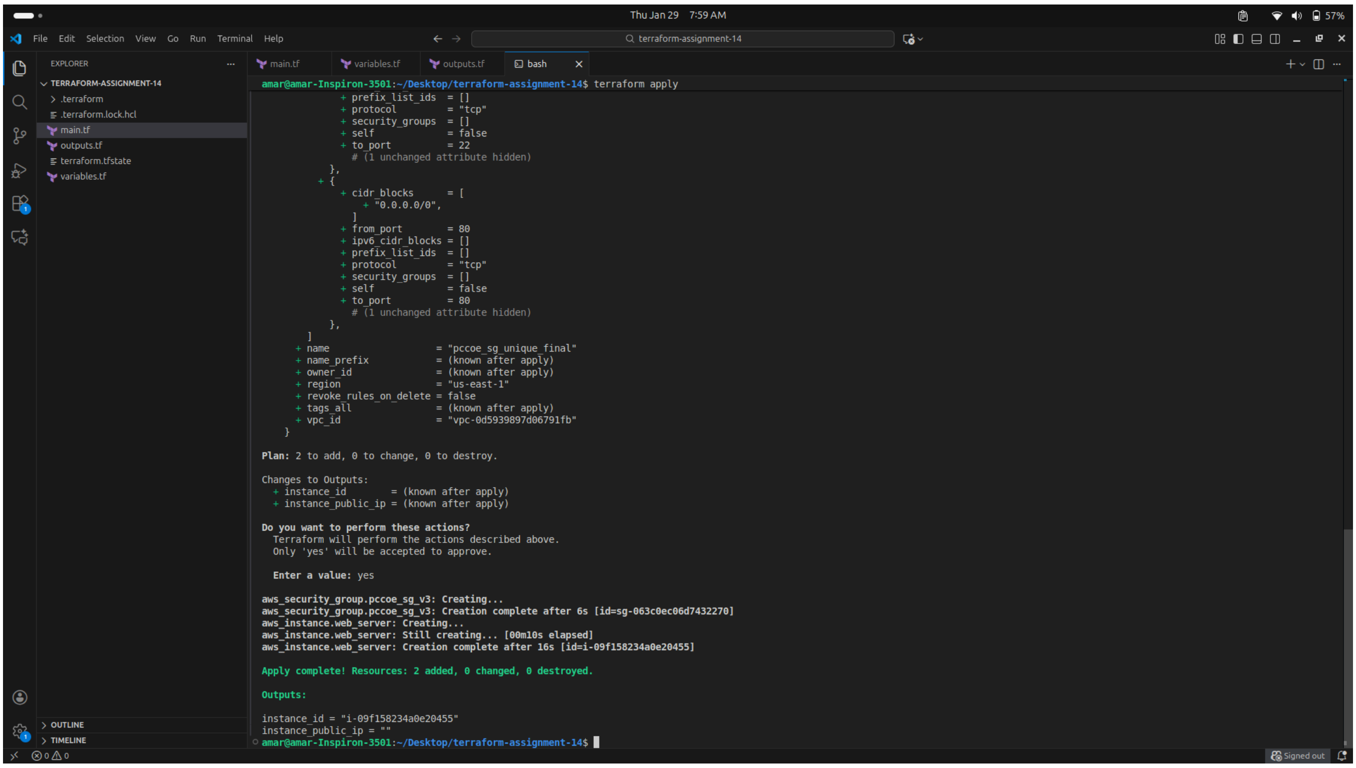
```
amar@amar-Inspiron-3501:~/Desktop/terraform-assignment-14$ terraform plan
              + protocol         = "-1"
              + security_groups  = []
              + self             = false
              + to_port          = 0
                # (1 unchanged attribute hidden)
            },
        ]
      + id                     = (known after apply)
      + ingress                = [
          + {
              + cidr_blocks      = [
                  + "0.0.0.0/0",
                ]
              + from_port        = 22
              + ipv6_cidr_blocks = []
              + prefix_list_ids  = []
              + protocol         = "tcp"
              + security_groups  = []
              + self             = false
              + to_port          = 22
                # (1 unchanged attribute hidden)
            },
          + {
              + cidr_blocks      = [
                  + "0.0.0.0/0",
                ]
              + from_port        = 80
              + ipv6_cidr_blocks = []
              + prefix_list_ids  = []
              + protocol         = "tcp"
              + security_groups  = []
              + self             = false
              + to_port          = 80
                # (1 unchanged attribute hidden)
            },
        ]
      + name                   = "pccoe_sg_unique_final"
      + name_prefix            = (known after apply)
      + owner_id               = (known after apply)
      + region                 = "us-east-1"
      + revoke_rules_on_delete = false
      + tags_all               = (known after apply)
      + vpc_id                 = "vpc-0d5939897d06791fb"
    }

Plan: 2 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + instance_id        = (known after apply)
  + instance_public_ip = (known after apply)

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
amar@amar-Inspiron-3501:~/Desktop/terraform-assignment-14$
```

Step 4: Applying and Provisioning

- Execute terraform apply and type yes to confirm.
- Terraform provisioned the aws_security_group.pccoe_sg_v3 followed by the aws_instance.web_server named PCCOE-DevOps-Server.



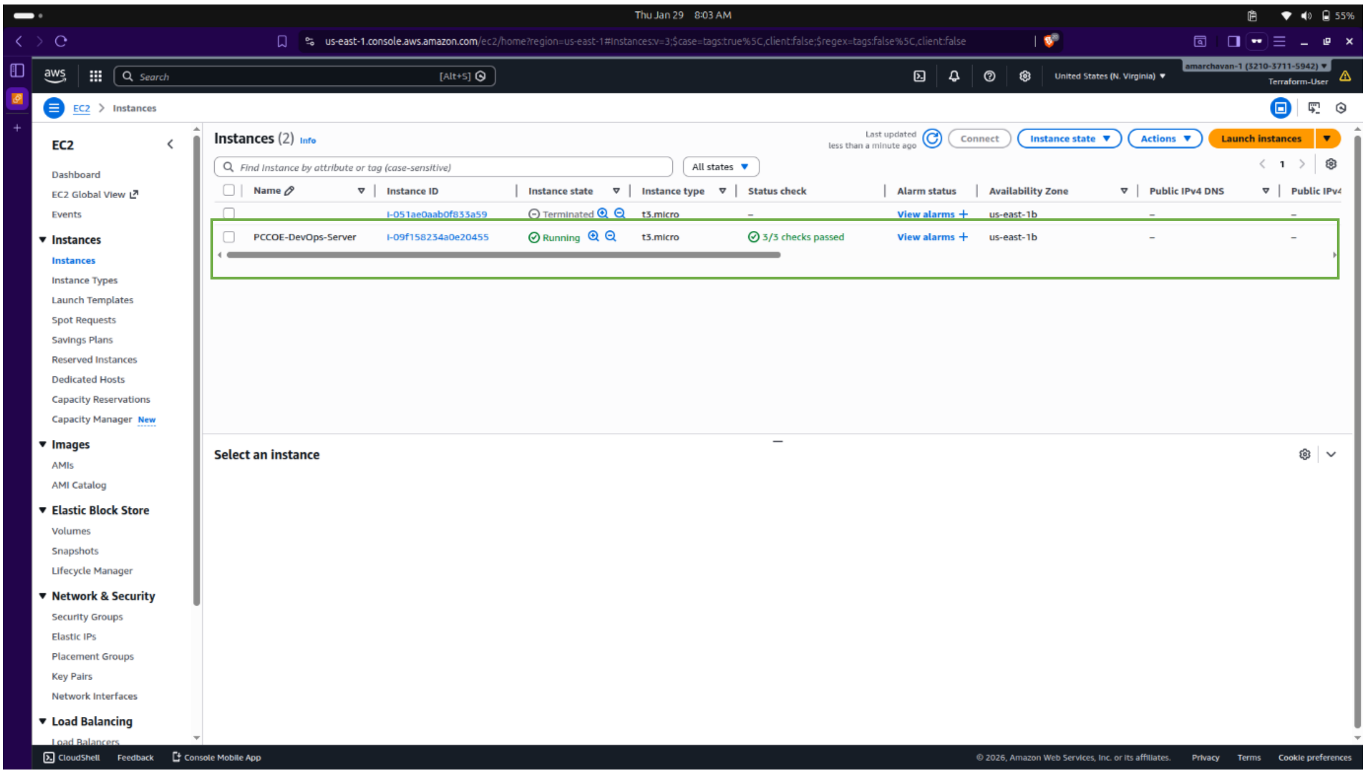
```
amar@amar-Inspiron-3501:~/Desktop/terraform-assignment-14$ terraform apply
data.aws_vpc.default: Reading...
data.aws_ami.ubuntu: Reading...
data.aws_ami.ubuntu: Read complete after 2s [id=ami-0b6c6ebed2801a5cb]
data.aws_vpc.default: Read complete after 3s [id=vpc-0d5939897d06791fb]
data.aws_subnets.default: Reading...
data.aws_subnets.default: Read complete after 0s [id=us-east-1]

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.web_server will be created
  + resource "aws_instance" "web_server" {
      + ami                                  = "ami-0b6c6ebed2801a5cb"
      + 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)
      + key_name                             = (known after apply)
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
      + placement_group                      = (known after apply)
      + placement_group_id                   = (known after apply)
      + placement_partition_number           = (known after apply)
      + primary_network_interface_id         = (known after apply)
      + private_dns                          = (known after apply)
      + private_ip                           = (known after apply)
      + public_dns                           = (known after apply)
      + public_ip                            = (known after apply)
      + region                               = "us-east-1"
      + secondary_private_ips                = (known after apply)
      + security_groups                      = (known after apply)
      + source_dest_check                    = true
      + spot_instance_request_id             = (known after apply)
      + subnet_id                            = "subnet-0a05df25bcaca79ec"
      + tags                                 = {
          + "Name" = "PCCOE-DevOps-Server"
        }
```



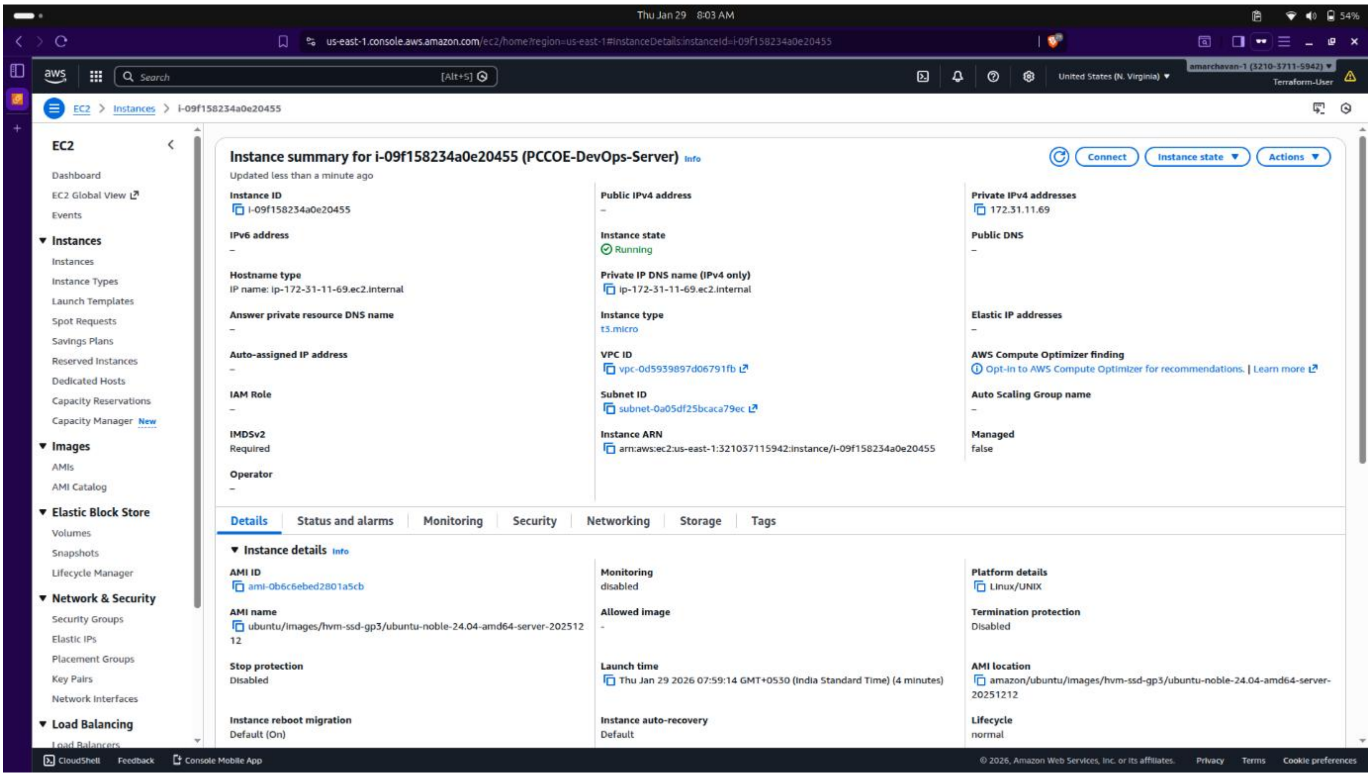
```
amar@amar-Inspiron-3501:~/Desktop/terraform-assignment-14$ terraform apply
              + prefix_list_ids  = []
              + protocol         = "tcp"
              + security_groups  = []
              + self             = false
              + to_port          = 22
                # (1 unchanged attribute hidden)
            },
          + {
              + cidr_blocks      = [
                  + "0.0.0.0/0",
                ]
              + from_port        = 80
              + ipv6_cidr_blocks = []
              + prefix_list_ids  = []
              + protocol         = "tcp"
              + security_groups  = []
              + self             = false
              + to_port          = 80
                # (1 unchanged attribute hidden)
            },
        ]
      + name                   = "pccoe_sg_unique_final"
      + name_prefix            = (known after apply)
      + owner_id               = (known after apply)
      + region                 = "us-east-1"
      + revoke_rules_on_delete = false
      + tags_all               = (known after apply)
      + vpc_id                 = "vpc-0d5939897d06791fb"
    }

Plan: 2 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + instance_id        = (known after apply)
  + instance_public_ip = (known after apply)

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_security_group.pccoe_sg_v3: Creating...
aws_security_group.pccoe_sg_v3: Creation complete after 6s [id=sg-063c0ec06d7432270]
aws_instance.web_server: Creating...
aws_instance.web_server: Still creating... [00m10s elapsed]
aws_instance.web_server: Creation complete after 16s [id=i-09f158234a0e20455]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.

Outputs:

instance_id = "i-09f158234a0e20455"
instance_public_ip = ""
amar@amar-Inspiron-3501:~/Desktop/terraform-assignment-14$ 
```

Step 5: Verification in AWS Console

- The EC2 Dashboard confirmed the instance i-09f158234a0e20455 was in a **"Running"** state with **3/3 status checks passed**.



□	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4
□		i-051ae0aab0f833a59	Terminated	t3.micro	–	View alarms +	us-east-1b	–	–
□	PCCOE-DevOps-Server	i-09f158234a0e20455	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1b	–	–



Instance summary for i-09f158234a0e20455 (PCCOE-DevOps-Server) Info

Updated less than a minute ago

Instance ID: i-09f158234a0e20455

Public IPv4 address: –

Private IPv4 addresses: 172.31.11.69

IPv6 address: –

Instance state: Running

Public DNS: –

Hostname type: IP name: ip-172-31-11-69.ec2.internal

Private IP DNS name (IPv4 only): ip-172-31-11-69.ec2.internal

Answer private resource DNS name: –

Instance type: t3.micro

Elastic IP addresses: –

Auto-assigned IP address: –

VPC ID: vpc-0d5939897d06791fb

AWS Compute Optimizer finding: Opt-in to AWS Compute Optimizer for recommendations. | Learn more

IAM Role: –

Subnet ID: subnet-0a05df25bcaca79ec

Auto Scaling Group name: –

IMDSv2: Required

Instance ARN: arn:aws:ec2:us-east-1:321037115942:instance/i-09f158234a0e20455

Managed: false

Operator: –

▼ Instance details Info

AMI ID: ami-0b6c6ebed2801a5cb

Monitoring: disabled

Platform details: Linux/UNIX

AMI name: ubuntu/images/hvm-ssd-gp3/ubuntu-noble-24.04-amd64-server-20251212

Allowed image: -

Termination protection: Disabled

Stop protection: Disabled

Launch time: Thu Jan 29 2026 07:59:14 GMT+0530 (India Standard Time) (4 minutes)

AMI location: amazon/ubuntu/images/hvm-ssd-gp3/ubuntu-noble-24.04-amd64-server-20251212

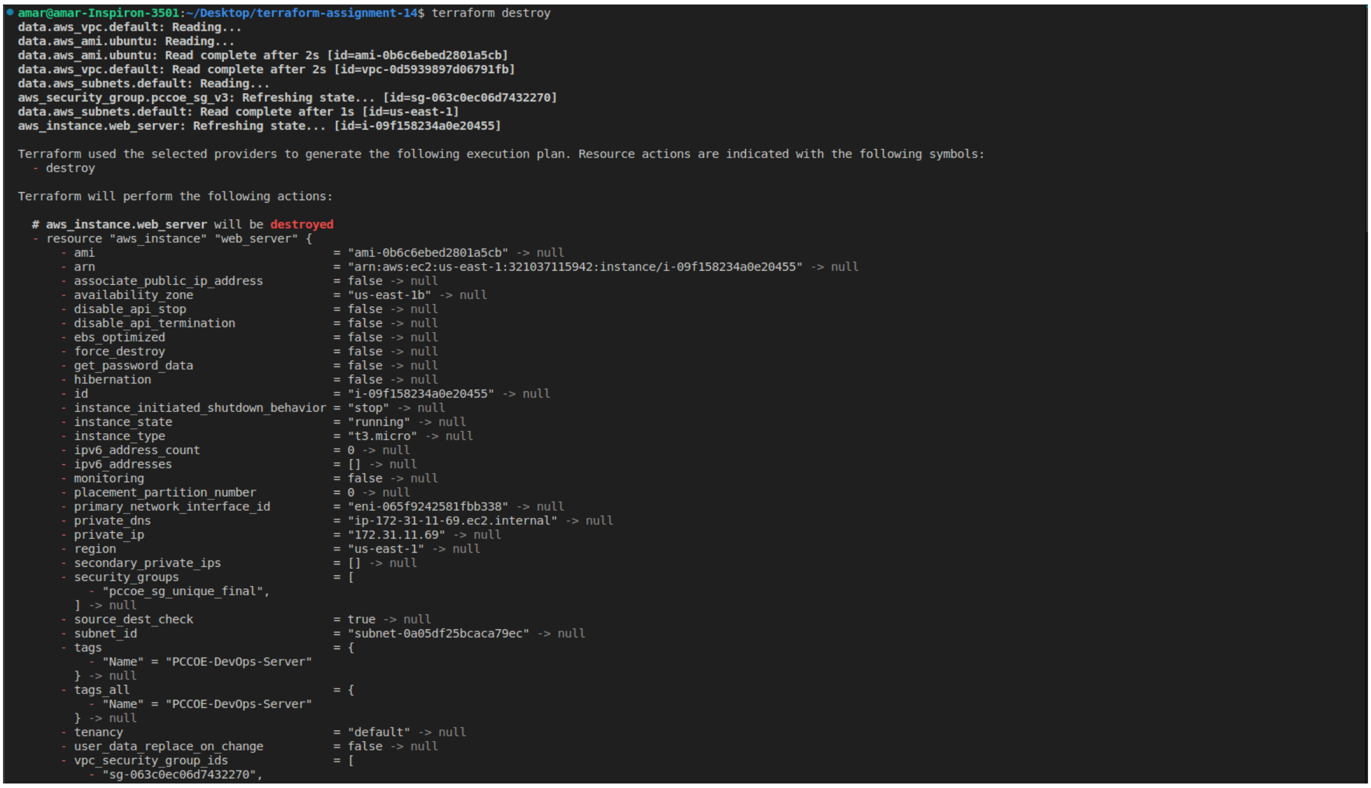
Instance reboot migration: Default (On)

Instance auto-recovery: Default

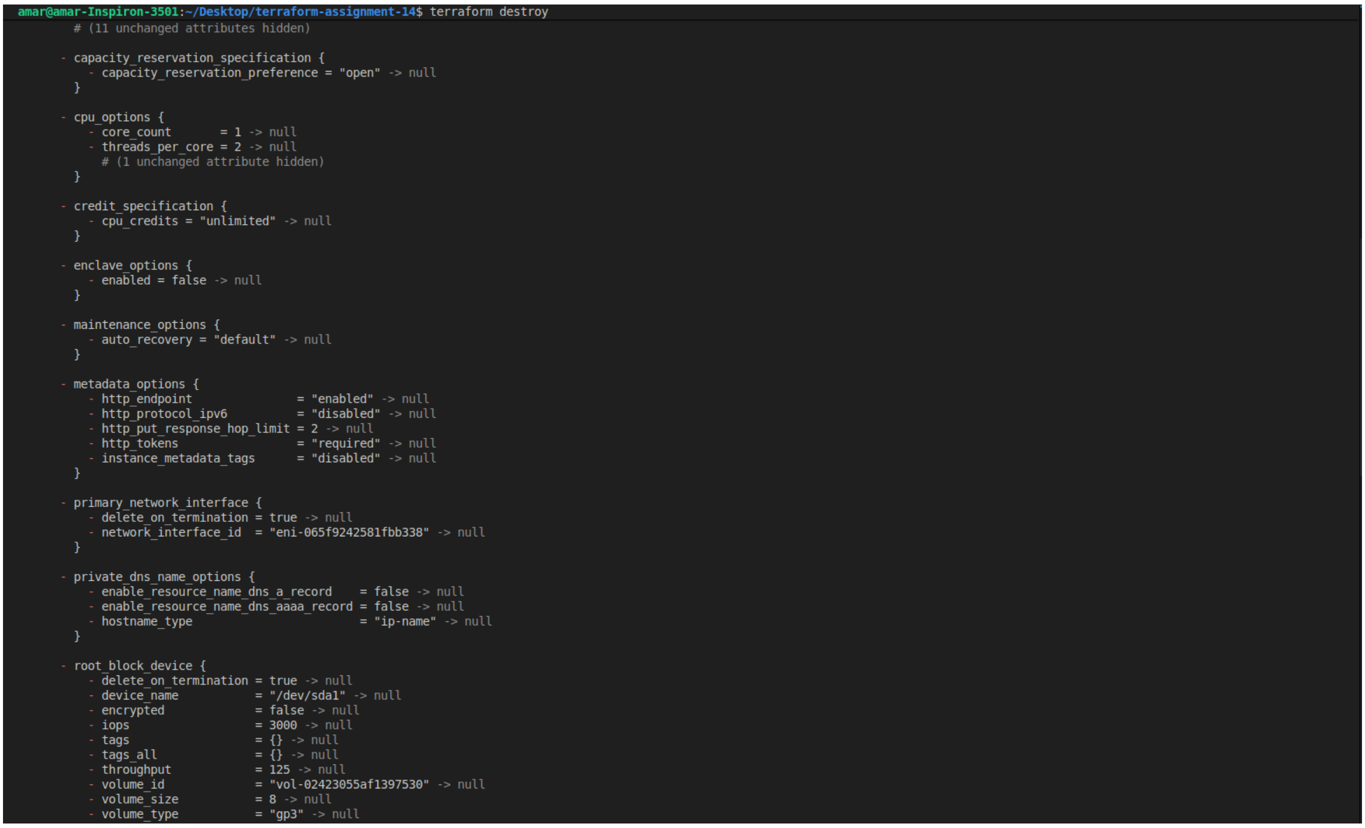
Lifecycle: normal

Step 6: Resource Destruction

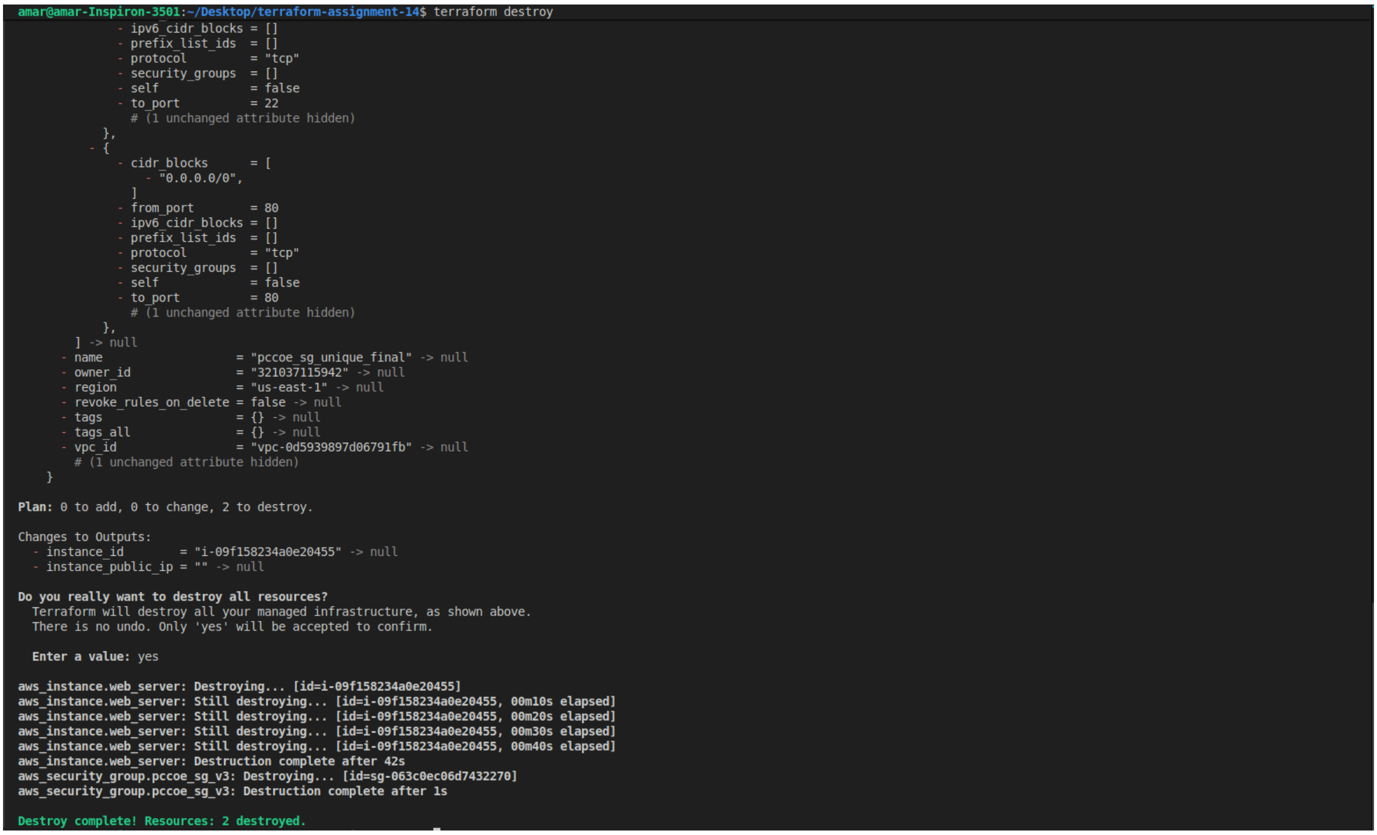
- To clean up the cloud environment, run terraform destroy.
- This safely terminated the instance and deleted the security group from the AWS region.



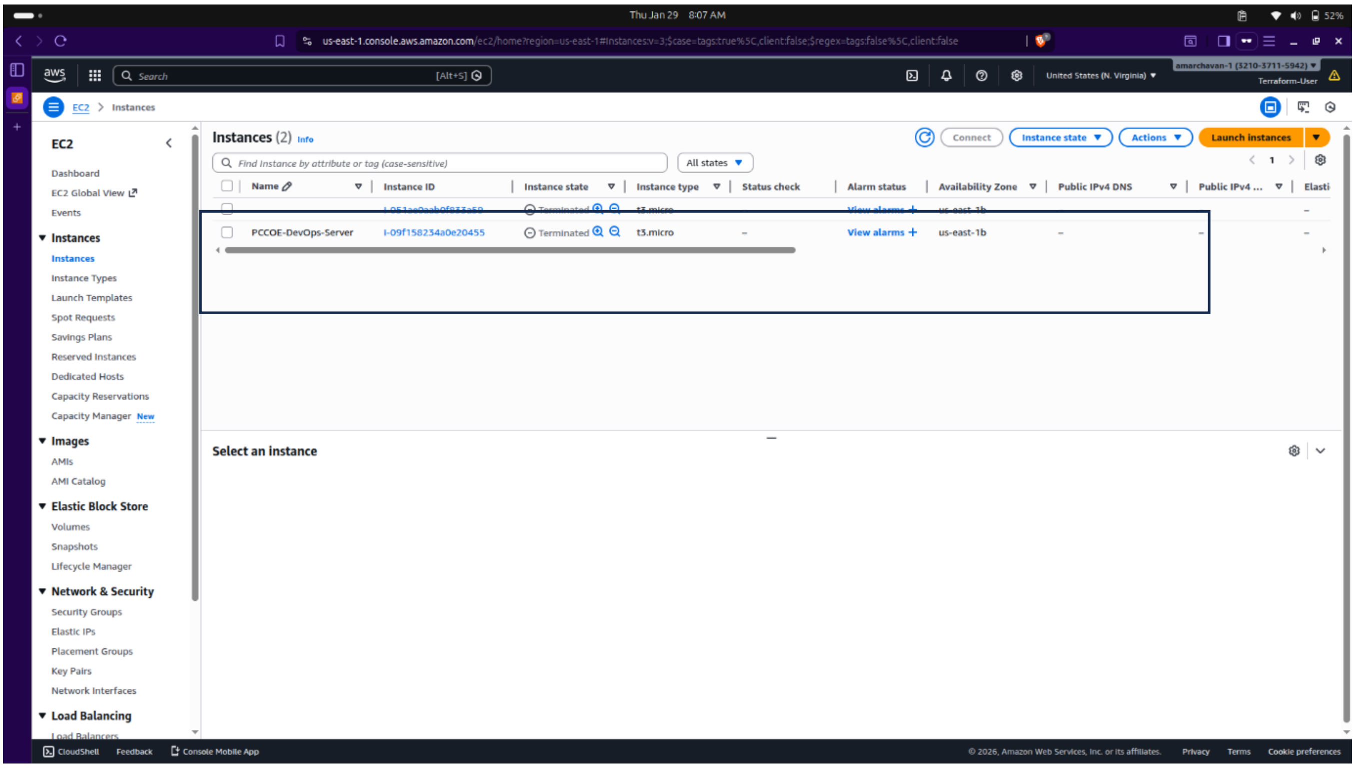
```
amar@amar-Inspiron-3501:~/Desktop/terraform-assignment-14$ terraform destroy
data.aws_vpc.default: Reading...
data.aws_ami.ubuntu: Reading...
data.aws_ami.ubuntu: Read complete after 2s [id=ami-0b6c6ebed2801a5cb]
data.aws_vpc.default: Read complete after 2s [id=vpc-0d5939897d06791fb]
data.aws_subnets.default: Reading...
aws_security_group.pccoe_sg_v3: Refreshing state... [id=sg-063c0ec06d7432270]
data.aws_subnets.default: Read complete after 1s [id=us-east-1]
aws_instance.web_server: Refreshing state... [id=i-09f158234a0e20455]

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.web_server will be destroyed
  - resource "aws_instance" "web_server" {
      - ami                                  = "ami-0b6c6ebed2801a5cb" -> null
      - arn                                  = "arn:aws:ec2:us-east-1:321037115942:instance/i-09f158234a0e20455" -> null
      - associate_public_ip_address          = false -> null
      - availability_zone                    = "us-east-1b" -> null
      - disable_api_stop                     = false -> null
      - disable_api_termination              = false -> null
      - ebs_optimized                        = false -> null
      - force_destroy                        = false -> null
      - get_password_data                    = false -> null
      - hibernation                          = false -> null
      - id                                   = "i-09f158234a0e20455" -> null
      - instance_initiated_shutdown_behavior = "stop" -> null
      - instance_state                       = "running" -> null
      - instance_type                        = "t3.micro" -> null
      - ipv6_address_count                   = 0 -> null
      - ipv6_addresses                       = [] -> null
      - monitoring                           = false -> null
      - placement_partition_number           = 0 -> null
      - primary_network_interface_id         = "eni-065f9242581fbb338" -> null
      - private_dns                          = "ip-172-31-11-69.ec2.internal" -> null
      - private_ip                           = "172.31.11.69" -> null
      - region                               = "us-east-1" -> null
      - secondary_private_ips                = [] -> null
      - security_groups                      = [
          - "pccoe_sg_unique_final",
        ] -> null
      - source_dest_check                    = true -> null
      - subnet_id                            = "subnet-0a05df25bcaca79ec" -> null
      - tags                                 = {
          - "Name" = "PCCOE-DevOps-Server"
        } -> null
      - tags_all                             = {
          - "Name" = "PCCOE-DevOps-Server"
        } -> null
      - tenancy                              = "default" -> null
      - user_data_replace_on_change          = false -> null
      - vpc_security_group_ids               = [
          - "sg-063c0ec06d7432270",
```



```
amar@amar-Inspiron-3501:~/Desktop/terraform-assignment-14$ terraform destroy
        # (11 unchanged attributes hidden)

      - capacity_reservation_specification {
          - capacity_reservation_preference = "open" -> null
        }

      - cpu_options {
          - core_count       = 1 -> null
          - threads_per_core = 2 -> null
            # (1 unchanged attribute hidden)
        }

      - credit_specification {
          - cpu_credits = "unlimited" -> null
        }

      - enclave_options {
          - enabled = false -> null
        }

      - maintenance_options {
          - auto_recovery = "default" -> null
        }

      - metadata_options {
          - http_endpoint               = "enabled" -> null
          - http_protocol_ipv6          = "disabled" -> null
          - http_put_response_hop_limit = 2 -> null
          - http_tokens                 = "required" -> null
          - instance_metadata_tags      = "disabled" -> null
        }

      - primary_network_interface {
          - delete_on_termination = true -> null
          - network_interface_id  = "eni-065f9242581fbb338" -> null
        }

      - private_dns_name_options {
          - enable_resource_name_dns_a_record    = false -> null
          - enable_resource_name_dns_aaaa_record = false -> null
          - hostname_type                        = "ip-name" -> null
        }

      - root_block_device {
          - delete_on_termination = true -> null
          - device_name           = "/dev/sda1" -> null
          - encrypted             = false -> null
          - iops                  = 3000 -> null
          - tags                  = {} -> null
          - tags_all              = {} -> null
          - throughput            = 125 -> null
          - volume_id             = "vol-02423055af1397530" -> null
          - volume_size           = 8 -> null
          - volume_type           = "gp3" -> null
```



```
amar@amar-Inspiron-3501:~/Desktop/terraform-assignment-14$ terraform destroy
              - ipv6_cidr_blocks = []
              - prefix_list_ids  = []
              - protocol         = "tcp"
              - security_groups  = []
              - self             = false
              - to_port          = 22
                # (1 unchanged attribute hidden)
            },
          - {
              - cidr_blocks      = [
                  - "0.0.0.0/0",
                ]
              - from_port        = 80
              - ipv6_cidr_blocks = []
              - prefix_list_ids  = []
              - protocol         = "tcp"
              - security_groups  = []
              - self             = false
              - to_port          = 80
                # (1 unchanged attribute hidden)
            },
        ] -> null
      - name                   = "pccoe_sg_unique_final" -> null
      - owner_id               = "321037115942" -> null
      - region                 = "us-east-1" -> null
      - revoke_rules_on_delete = false -> null
      - tags                   = {} -> null
      - tags_all               = {} -> null
      - vpc_id                 = "vpc-0d5939897d06791fb" -> null
        # (1 unchanged attribute hidden)
    }

Plan: 0 to add, 0 to change, 2 to destroy.

Changes to Outputs:
  - instance_id        = "i-09f158234a0e20455" -> null
  - instance_public_ip = "" -> null

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_instance.web_server: Destroying... [id=i-09f158234a0e20455]
aws_instance.web_server: Still destroying... [id=i-09f158234a0e20455, 00m10s elapsed]
aws_instance.web_server: Still destroying... [id=i-09f158234a0e20455, 00m20s elapsed]
aws_instance.web_server: Still destroying... [id=i-09f158234a0e20455, 00m30s elapsed]
aws_instance.web_server: Still destroying... [id=i-09f158234a0e20455, 00m40s elapsed]
aws_instance.web_server: Destruction complete after 42s
aws_security_group.pccoe_sg_v3: Destroying... [id=sg-063c0ec06d7432270]
aws_security_group.pccoe_sg_v3: Destruction complete after 1s

Destroy complete! Resources: 2 destroyed.
```



Instances (2) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elasti
	i-051ae0aab0f833a59	Terminated	t3.micro	–	View alarms +	us-east-1b	–		–
PCCOE-DevOps-Server	i-09f158234a0e20455	Terminated	t3.micro	–	View alarms +	us-east-1b	–	–	–

Conclusion

The successful execution of Assignment 14 effectively demonstrates the core principles of Infrastructure as Code (IaC) by automating cloud resource management. By transitioning from manual console-based provisioning to programmatic configuration via the AWS CLI, I established a secure and authenticated environment necessary for automated orchestration. The implementation of the Terraform lifecycle—encompassing init, plan, apply, and destroy—provided a structured and predictable workflow for managing the PCCOE-DevOps-Server instance.

A key takeaway was the importance of State Management, where the terraform.tfstate file ensured that the local configuration remained synchronized with the real-world cloud resources. Furthermore, the ability to safely terminate and delete the infrastructure using the destroy command highlighted the efficiency and cost-control benefits of using Terraform in a DevOps pipeline. Ultimately, this assignment validates that using HCL-based scripts ensures consistency, repeatability, and significantly reduces the manual overhead typically associated with managing enterprise-scale cloud infrastructure.