Name: Abdurrahman Qureshi

Roll No: 242466

Practical No: 14

Date Of Performance: 10/10/2025

Aim: To understand Infrastructure as Code (IaC) concepts and implement AWS EC2 instance provisioning using Terraform's lifecycle management.

1. What is Terraform?

2. What is Infrastructure as a Code (IaC) ?

3. Perform an experiment, to understand Terraform lifecycle, core concepts/terminologies and install it on a Linux Machine.

4. Using Terraform , create an EC2 instance on AWS cloud

5. Explain following Terraform commands in one line

- terraform init:
- terraform validate:
- terraform plan:
- terraform apply:
- terraform destroy.

Ans.1:

An open-source Infrastructure as Code (IaC) tool by HashiCorp that uses declarative configuration files to define, provision, and manage cloud infrastructure safely and efficiently.

Ans.2:

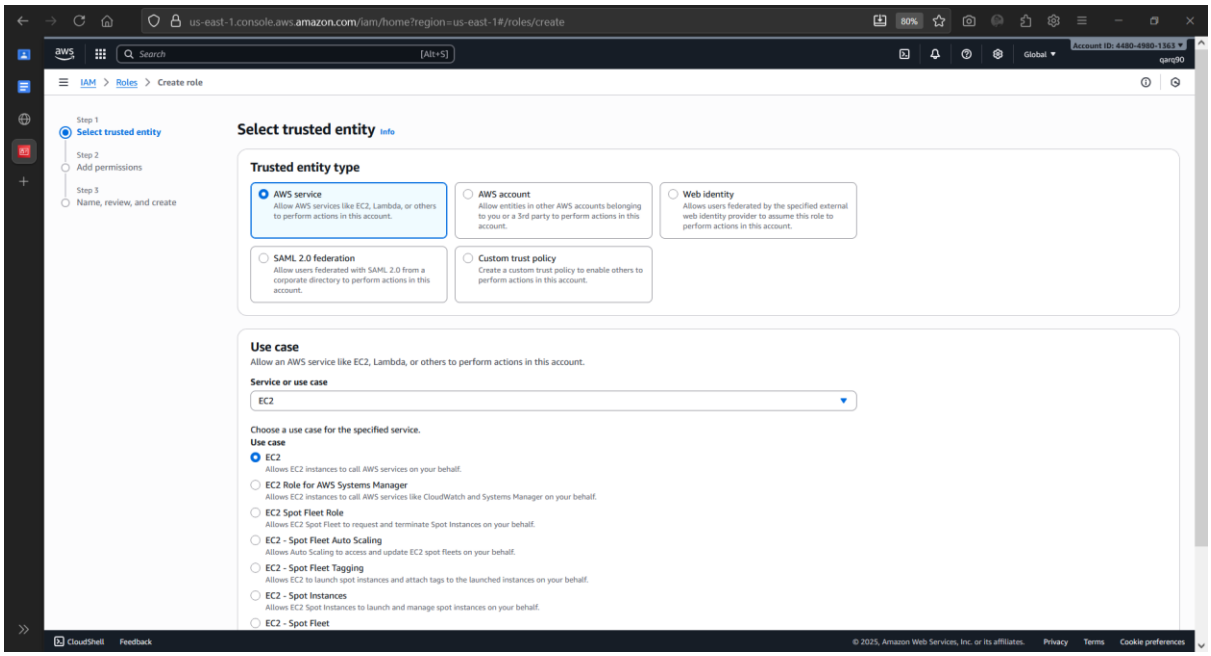
Infrastructure as Code (IaC) is a key DevOps practice that manages and provisions computing infrastructure through machine-readable definition files, instead of manual processes. It treats servers, networks, and databases as software, defined in structured code.

This enables automated, consistent, and repeatable environment setups, version control for infrastructure changes, and significantly reduces human error, leading to faster, more reliable deployments across development, staging, and production.

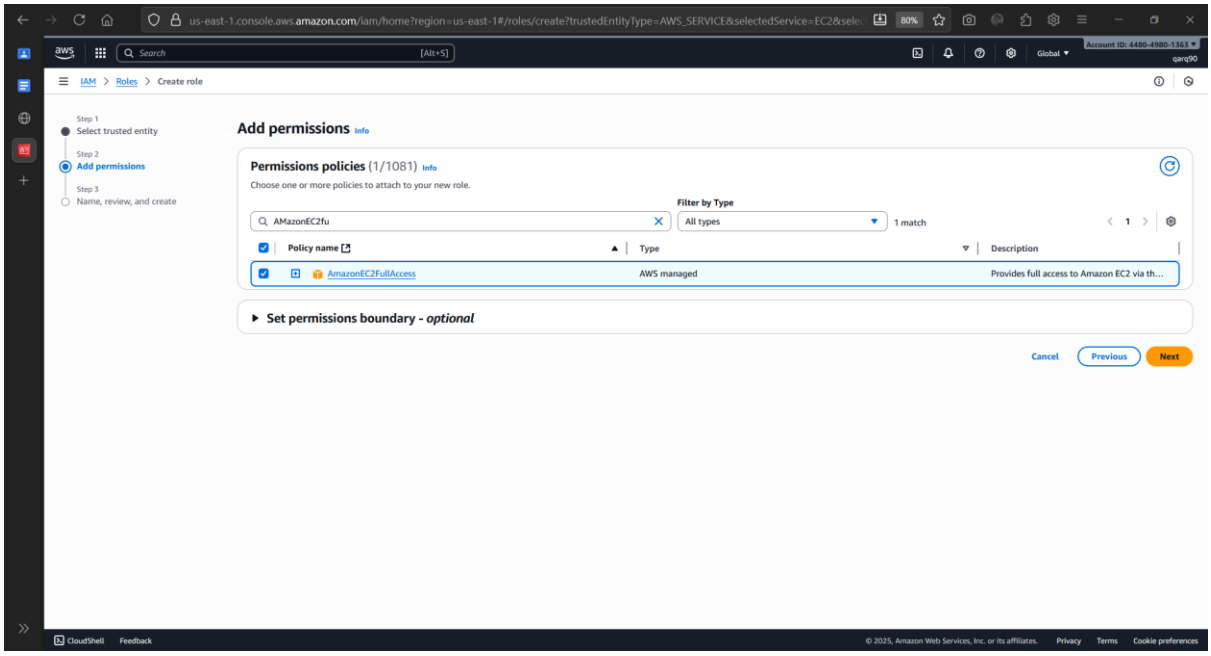
Ans.5:

- terraform init: Initializes a working directory, downloading provider plugins.
- terraform validate: Checks the configuration files for syntax errors.
- terraform plan: Creates an execution plan showing what infrastructure will be created.
- terraform apply: Builds or changes the infrastructure as defined in the plan.
- terraform destroy: Destroys all Terraform-managed infrastructure.

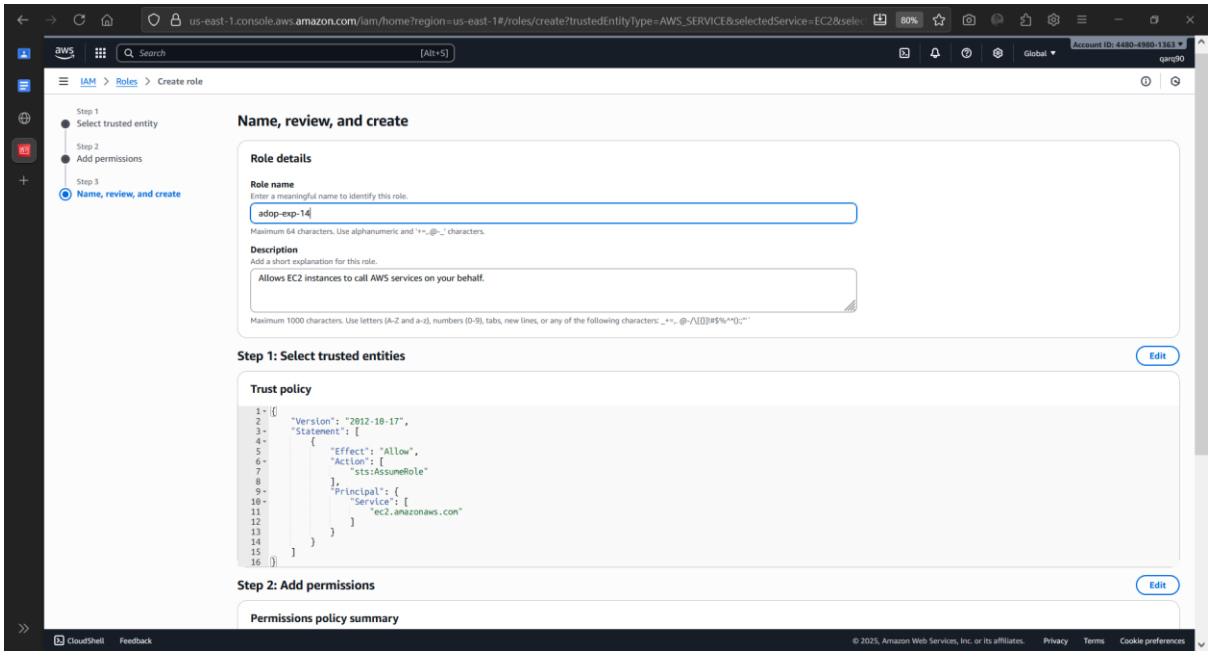
Ans.3 & 4:



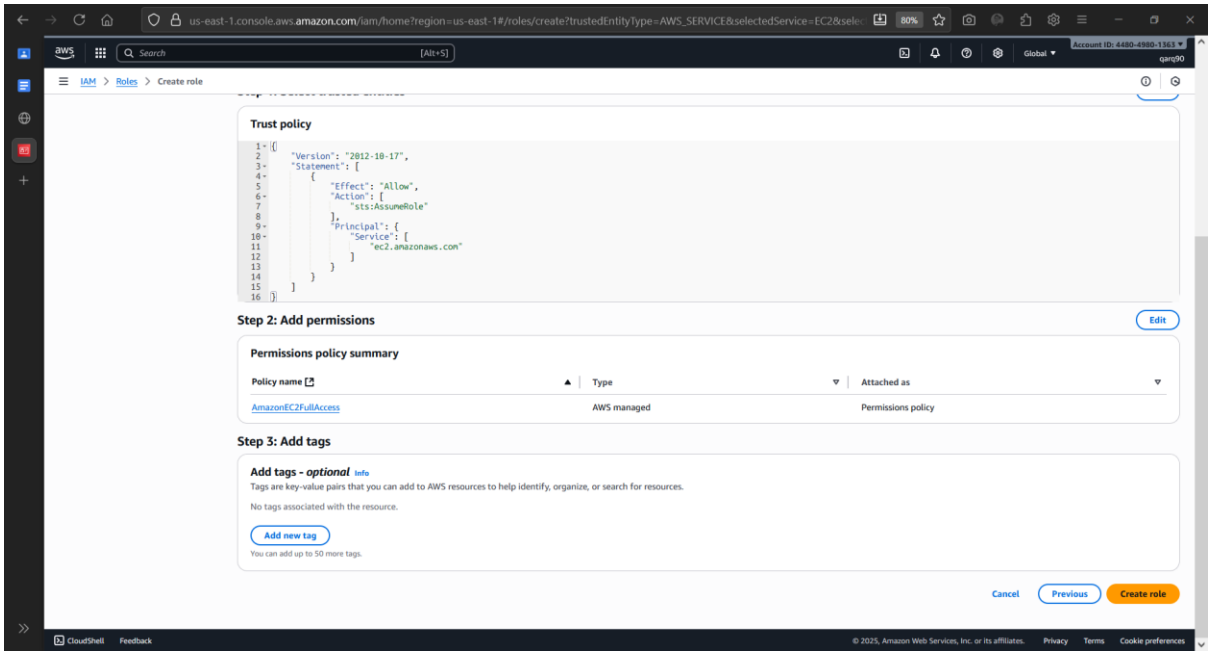
Creating IAM Role



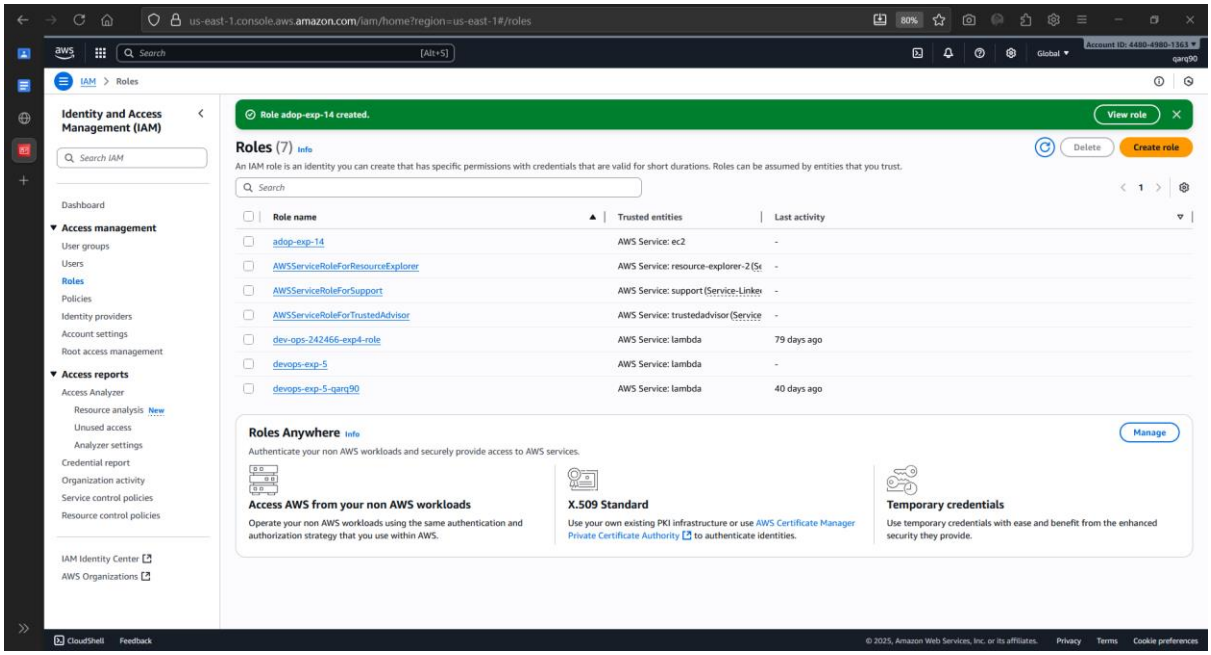
Adding Permission



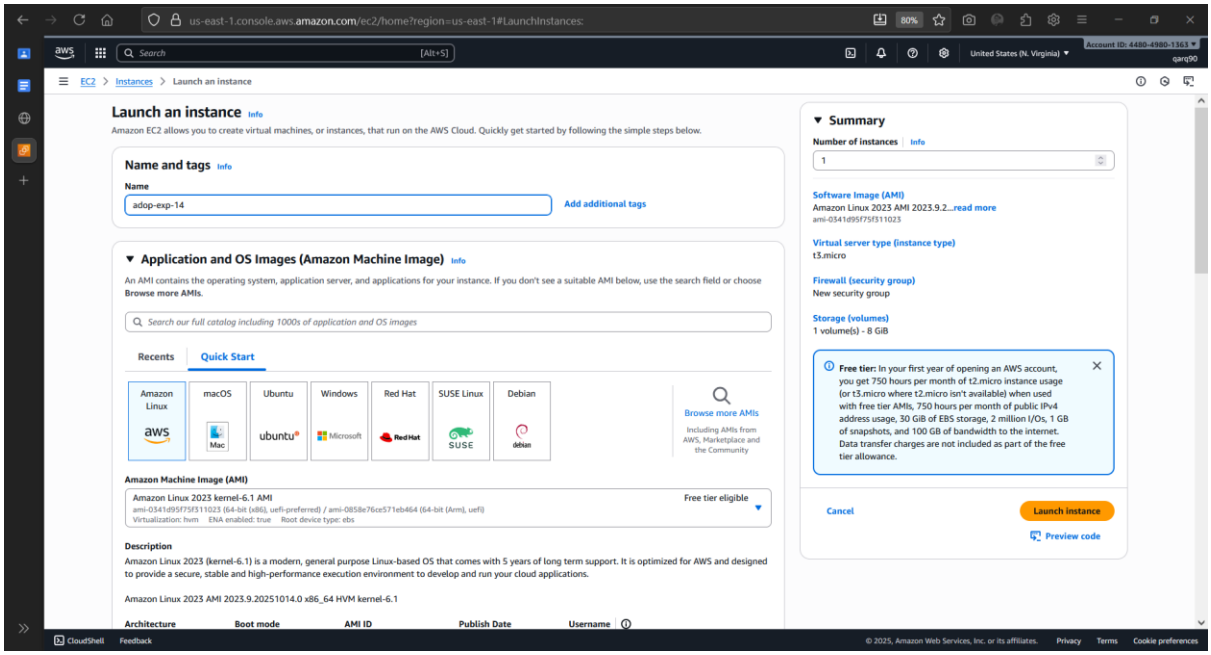
Setting Role Name



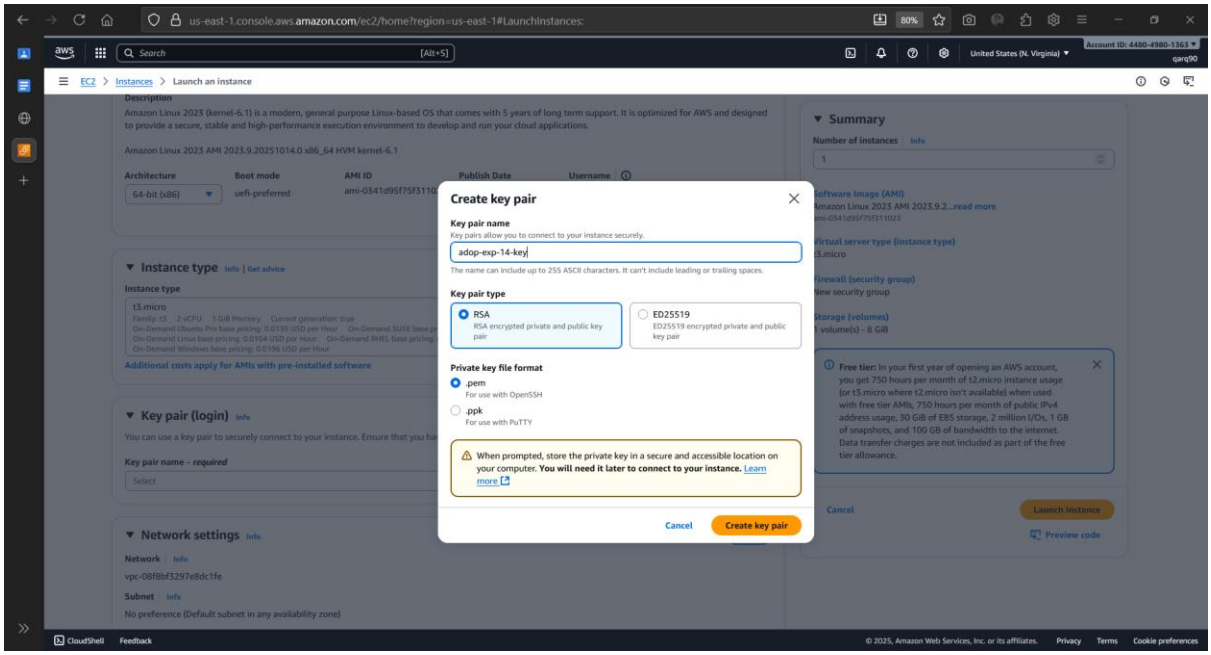
Role Configuration Complete



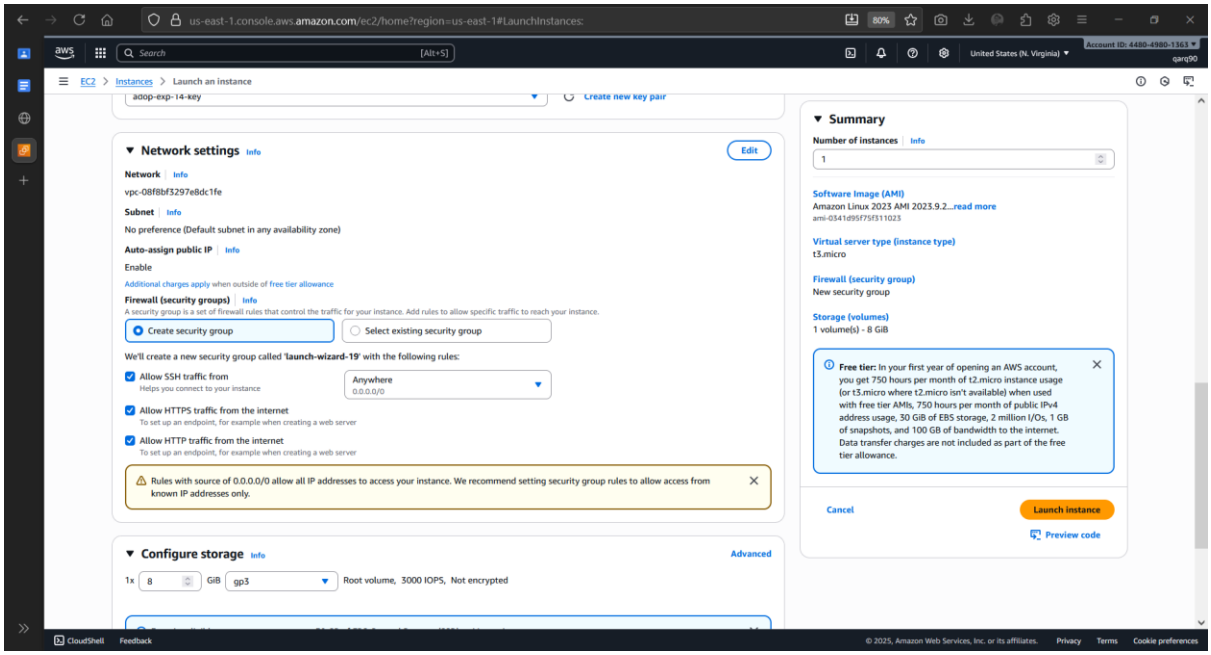
Role Created Successfully



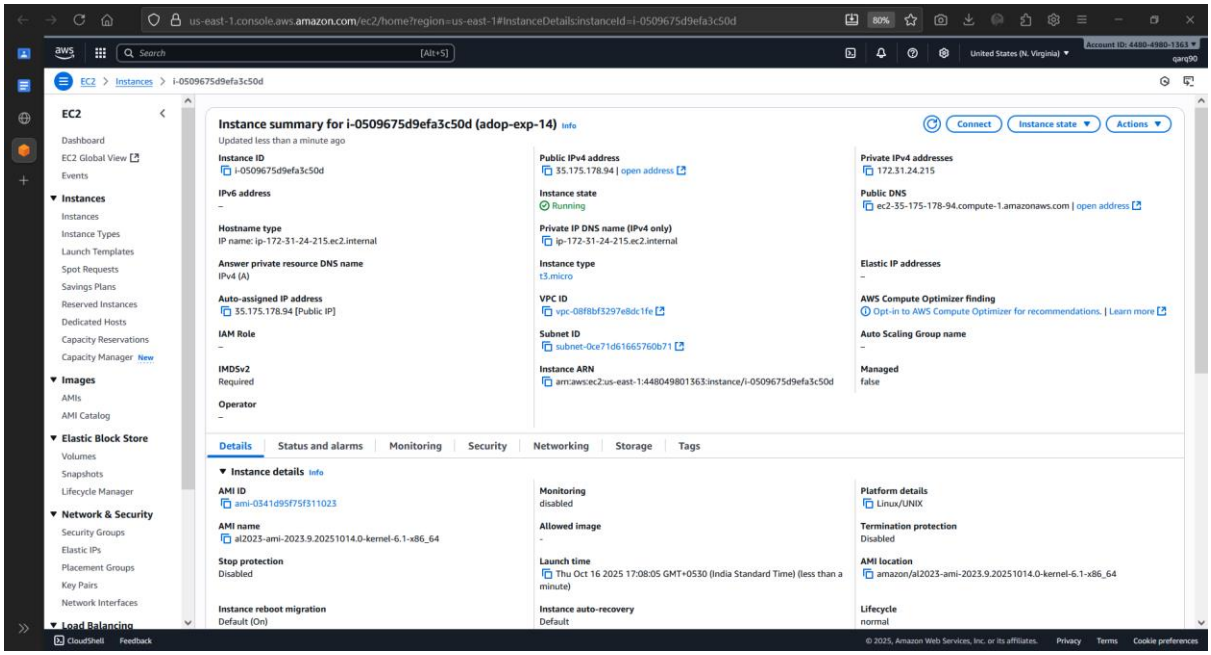
Creating an EC2 Instance



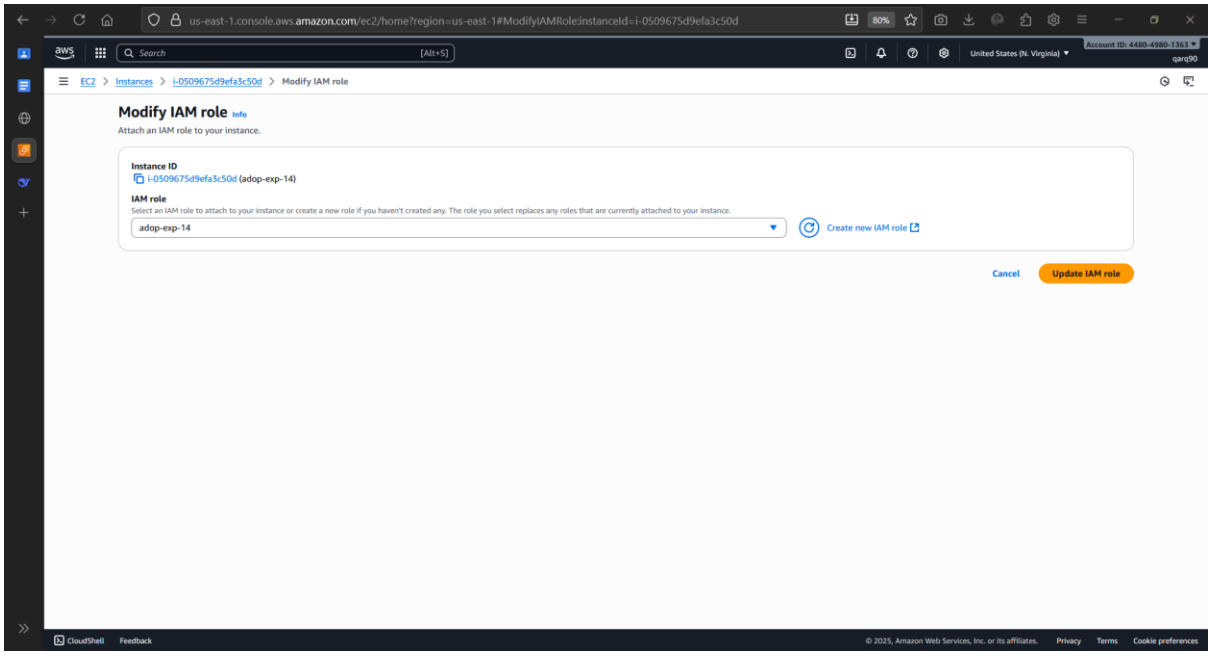
Creating Instance Key



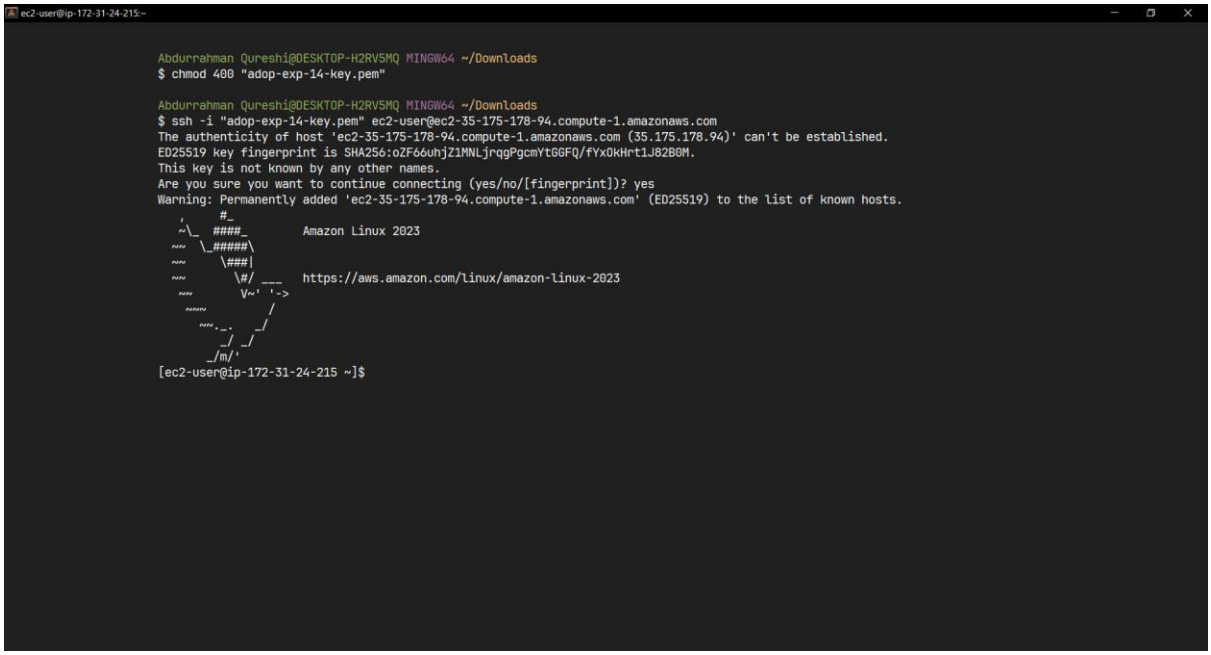
Configuring Instance Network Settings



Instance Details



Adding the IAM Role to the Instance



Connected to the Instance Successfully

variables.tf:

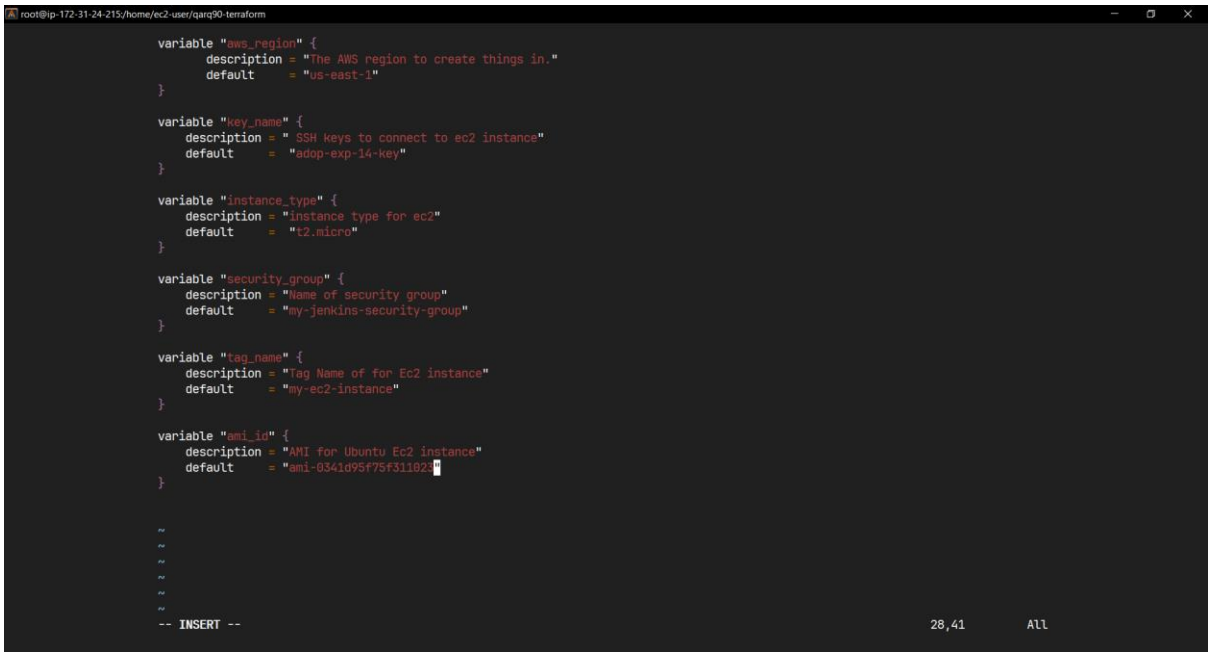
```
sudo vi variables.tf

variable "aws_region" {
       description = "The AWS region to create things in."
       default     = "us-east-1”
}

variable "key_name" {
    description = " SSH keys to connect to ec2 instance"
    default     =  "[ Your PEM File Name without extension ]"
}

variable "instance_type" {
    description = "instance type for ec2"
    default     =  "t2.micro"
}

variable "security_group" {
    description = "Name of security group"
    default     = "my-jenkins-security-group"
}
```

```
variable "tag_name" {
    description = "Tag Name of for Ec2 instance"
    default     = "my-ec2-instance"
}

variable "ami_id" {
    description = "AMI for Ubuntu Ec2 instance"
    default     = "[ Select instance from portal ] "
}
```

main.tf:

```
sudo vi main.tf

provider "aws" {
  region = var.aws_region
}

#Create security group with firewall rules
resource "aws_security_group" "security_jenkins_grp" {
  name        = var.security_group
  description = "security group for jenkins"

  ingress {
    from_port   = 8080
    to_port     = 8080
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

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

 # outbound from jenkis server
  egress {
    from_port   = 0
    to_port     = 65535
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  tags= {
```

```
    Name = var.security_group
  }
}

resource "aws_instance" "myFirstInstance" {
  ami           = var.ami_id
  key_name = var.key_name
  instance_type = var.instance_type
  vpc_security_group_ids = [aws_security_group.security_jenkins_grp.id]
  tags= {
    Name = var.tag_name
  }
}

# Create Elastic IP address
resource "aws_eip" "myFirstInstance" {
  domain      = "vpc"
  instance = aws_instance.myFirstInstance.id
tags= {
    Name = "jenkins_elastic_ip"
  }
}
```



Variable.tf File

```
root@ip-172-31-24-215:/home/ec2-user/qarq90-terraform

provider "aws" {
  region = var.aws_region
}

#Create security group with firewall rules

resource "aws_security_group" "security_jenkins_grp" {
  name        = var.security_group
  description = "security group for jenkins"

ingress {
    from_port   = 8080
    to_port     = 8080
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

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

# outbound from jenkis server

  egress {
    from_port   = 0
    to_port     = 65535
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  tags= {
    Name = var.security_group
  }
}
-- INSERT --                                                                    1,1           Top
```

Main.tf

```
root@ip-172-31-24-215:/home/ec2-user/qarq90-terraform

[ec2-user@ip-172-31-24-215 ~]$ sudo su
[root@ip-172-31-24-215 ec2-user]# mkdir qarq90-terraform
[root@ip-172-31-24-215 ec2-user]# cd qarq90-terraform
[root@ip-172-31-24-215 qarq90-terraform]# sudo vi variables.tf
[root@ip-172-31-24-215 qarq90-terraform]# sudo vi main.tf
[root@ip-172-31-24-215 qarq90-terraform]# ls
main.tf  variables.tf
[root@ip-172-31-24-215 qarq90-terraform]# sudo yum update -y
Amazon Linux 2023 Kernel Livepatch repository                                    220 kB/s |  26 kB     00:00
Last metadata expiration check: 0:00:01 ago on Thu Oct 16 12:04:27 2025.
Dependencies resolved.
Nothing to do.
Complete!
[root@ip-172-31-24-215 qarq90-terraform]# sudo yum install -y wget unzip
Last metadata expiration check: 0:00:15 ago on Thu Oct 16 12:04:27 2025.
Package wget-1.21.3-1.amzn2023.0.4.x86_64 is already installed.
Package unzip-6.0-57.amzn2023.0.2.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[root@ip-172-31-24-215 qarq90-terraform]# wget https://releases.hashicorp.com/terraform/1.0.9/terraform_1.0.9_linux_amd64.zip
--2025-10-16 12:05:17--  https://releases.hashicorp.com/terraform/1.0.9/terraform_1.0.9_linux_amd64.zip
Resolving releases.hashicorp.com (releases.hashicorp.com)... 3.171.85.128, 3.171.85.65, 3.171.85.80, ...
Connecting to releases.hashicorp.com (releases.hashicorp.com)|3.171.85.128|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 32674820 (31M) [application/zip]
Saving to: 'terraform_1.0.9_linux_amd64.zip'

terraform_1.0.9_linux_amd64.zip  100%[===================================================>]  31.16M  --.-KB/s    in 0.08s

2025-10-16 12:05:17 (379 MB/s) - 'terraform_1.0.9_linux_amd64.zip' saved [32674820/32674820]

[root@ip-172-31-24-215 qarq90-terraform]#
```

Installing Terrform

```
root@ip-172-31-24-215:/home/ec2-user/qarq90-terraform

[root@ip-172-31-24-215 qarq90-terraform]# unzip terraform_1.0.9_linux_amd64.zip
Archive:  terraform_1.0.9_linux_amd64.zip
  inflating: terraform
[root@ip-172-31-24-215 qarq90-terraform]# sudo cp terraform /bin
[root@ip-172-31-24-215 qarq90-terraform]# sudo cp terraform /bin/
[root@ip-172-31-24-215 qarq90-terraform]# terraform --version
Terraform v1.0.9
on linux_amd64

Your version of Terraform is out of date! The latest version
is 1.13.4. You can update by downloading from https://www.terraform.io/downloads.html
[root@ip-172-31-24-215 qarq90-terraform]#
```

Installing Dependencies

```
root@ip-172-31-24-215:/home/ec2-user/qarq90-terraform

[root@ip-172-31-24-215 qarq90-terraform]# terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.16.0...
- Installed hashicorp/aws v6.16.0 (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.
[root@ip-172-31-24-215 qarq90-terraform]#
```

"`terraform init`" Ouput

```
root@ip-172-31-24-215:/home/ec2-user/qarq90-terraform

variable "aws_region" {
provider "aws" {
[root@ip-172-31-24-215 qarq90-terraform]# terraform validate
Success! The configuration is valid.

[root@ip-172-31-24-215 qarq90-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_eip.myFirstInstance will be created
  + resource "aws_eip" "myFirstInstance" {
      + allocation_id        = (known after apply)
      + arn                  = (known after apply)
      + association_id       = (known after apply)
      + carrier_ip           = (known after apply)
      + customer_owned_ip    = (known after apply)
      + domain               = "vpc"
      + id                   = (known after apply)
      + instance             = (known after apply)
      + ipam_pool_id         = (known after apply)
      + network_border_group = (known after apply)
      + network_interface    = (known after apply)
      + private_dns          = (known after apply)
      + private_ip           = (known after apply)
      + ptr_record           = (known after apply)
      + public_dns           = (known after apply)
      + public_ip            = (known after apply)
      + public_ipv4_pool     = (known after apply)
      + region               = "us-east-1"
      + tags                 = {
          + "Name" = "jenkins_elastic_ip"
        }
      + tags_all             = {
```

```
root@ip-172-31-24-215:/home/ec2-user/qarq90-terraform

          + self             = false
          + to_port          = 22
        },
      + {
          + cidr_blocks      = [
              + "0.0.0.0/0",
            ]
          + description      = ""
          + from_port        = 8080
          + ipv6_cidr_blocks = []
          + prefix_list_ids  = []
          + protocol         = "tcp"
          + security_groups  = []
          + self             = false
          + to_port          = 8080
        },
    ]
    + name                   = "my-jenkins-security-group"
    + name_prefix            = (known after apply)
    + owner_id               = (known after apply)
    + region                 = "us-east-1"
    + revoke_rules_on_delete = false
    + tags                   = {
        + "Name" = "my-jenkins-security-group"
      }
    + tags_all               = {
        + "Name" = "my-jenkins-security-group"
      }
    + vpc_id                 = (known after apply)
  }

Plan: 3 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 actions if you run
"terraform apply" now.
[root@ip-172-31-24-215 qarq90-terraform]#
```

```
root@ip-172-31-24-215:/home/ec2-user/qarq90-terraform

                + self            = false
                + to_port         = 22
              },
              + {
                + cidr_blocks      = [
                    + "0.0.0.0/0",
                  ]
                + description      = ""
                + from_port        = 8080
                + ipv6_cidr_blocks = []
                + prefix_list_ids  = []
                + protocol         = "tcp"
                + security_groups  = []
                + self             = false
                + to_port          = 8080
              },
            ]
        + name                   = "my-jenkins-security-group"
        + name_prefix            = (known after apply)
        + owner_id               = (known after apply)
        + region                 = "us-east-1"
        + revoke_rules_on_delete = false
        + tags                   = {
            + "Name" = "my-jenkins-security-group"
          }
        + tags_all               = {
            + "Name" = "my-jenkins-security-group"
          }
        + vpc_id                 = (known after apply)
      }

Plan: 3 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
```

"terraform validate" and "terraform plan" Output

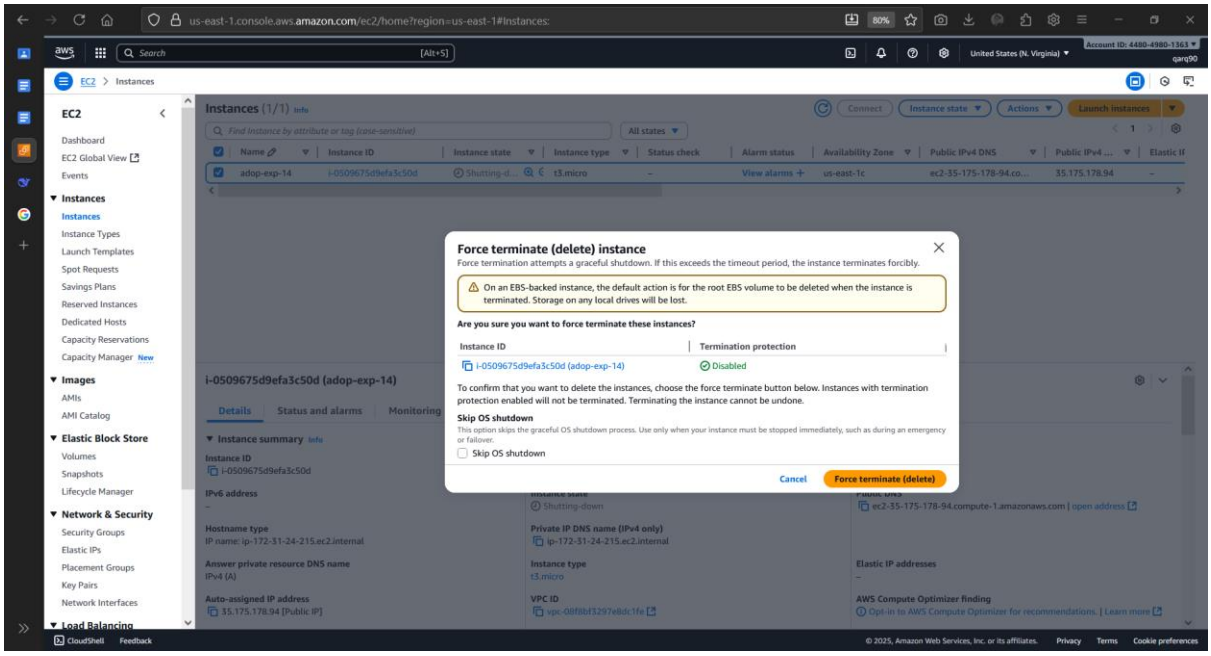
```
root@ip-172-31-24-215:/home/ec2-user/qarq90-terraform

        + private_dns_name_options {
            + enable_resource_name_dns_a_record    = (known after apply)
            + enable_resource_name_dns_aaaa_record = (known after apply)
            + hostname_type                        = (known after apply)
          }

        + root_block_device {
            + delete_on_termination = (known after apply)
            + device_name           = (known after apply)
            + encrypted             = (known after apply)
            + iops                  = (known after apply)
            + kms_key_id            = (known after apply)
            + tags                  = (known after apply)
            + tags_all              = (known after apply)
            + throughput            = (known after apply)
            + volume_id             = (known after apply)
            + volume_size           = (known after apply)
            + volume_type           = (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.myFirstInstance: Creating...
aws_instance.myFirstInstance: Still creating... [10s elapsed]
aws_instance.myFirstInstance: Creation complete after 13s [id=i-0988a23b28aabd4d5]
aws_eip.myFirstInstance: Creating...
aws_eip.myFirstInstance: Creation complete after 2s [id=eipalloc-0ec844d5a591abeee]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
[root@ip-172-31-24-215 qarq90-terraform]#
```

"terraform apply" Output

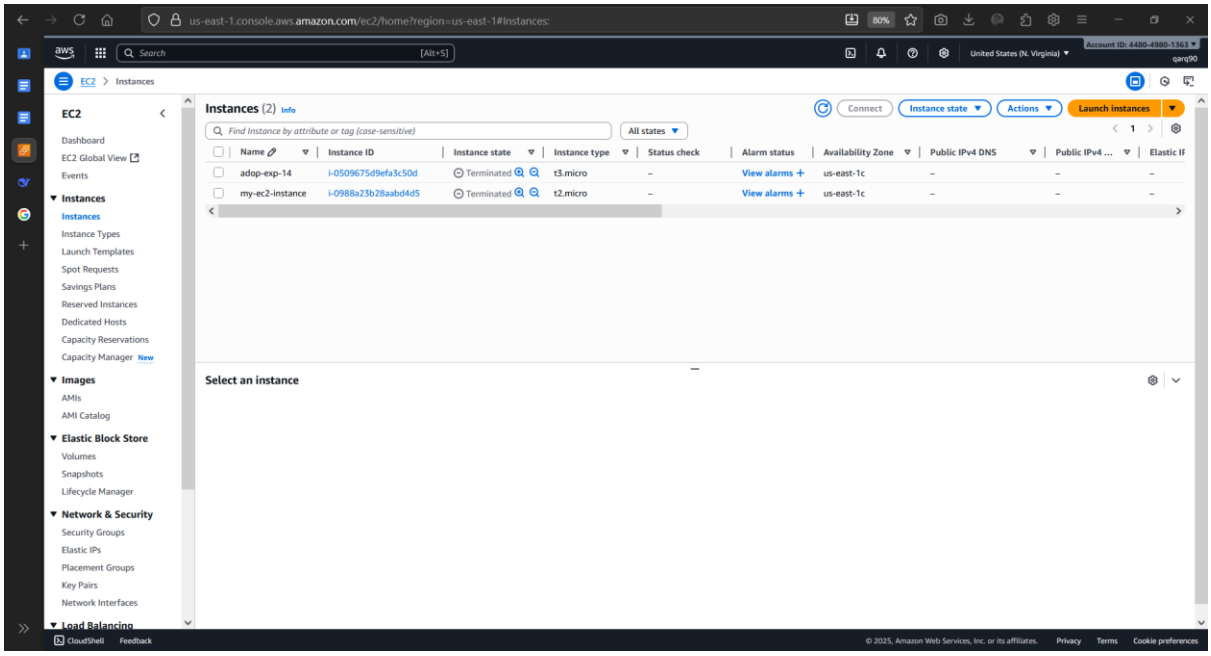
```
root@ip-172-31-24-215:/home/ec2-user/qarq90-terraform

                - security_groups  = []
                - self             = false
                - to_port          = 22
              },
              - {
                - cidr_blocks      = [
                    - "0.0.0.0/0",
                  ]
                - description      = ""
                - from_port        = 8080
                - ipv6_cidr_blocks = []
                - prefix_list_ids  = []
                - protocol         = "tcp"
                - security_groups  = []
                - self             = false
                - to_port          = 8080
              },
          ] -> null
          - name                   = "my-jenkins-security-group" -> null
          - owner_id               = "448049801363" -> null
          - region                 = "us-east-1" -> null
          - revoke_rules_on_delete = false -> null
          - tags                   = {
              - "Name" = "my-jenkins-security-group"
          } -> null
          - tags_all               = {
              - "Name" = "my-jenkins-security-group"
          } -> null
          - vpc_id                 = "vpc-08f8bf3297e8dc1fe" -> null
      }

Plan: 0 to add, 0 to change, 3 to destroy.

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
```

```
root@ip-172-31-24-215:/home/ec2-user/qarq90-terraform

                - self             = false
                - to_port          = 8080
              },
          ] -> null
          - name                   = "my-jenkins-security-group" -> null
          - owner_id               = "448049801363" -> null
          - region                 = "us-east-1" -> null
          - revoke_rules_on_delete = false -> null
          - tags                   = {
              - "Name" = "my-jenkins-security-group"
          } -> null
          - tags_all               = {
              - "Name" = "my-jenkins-security-group"
          } -> null
          - vpc_id                 = "vpc-08f8bf3297e8dc1fe" -> null
      }

Plan: 0 to add, 0 to change, 3 to destroy.

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_eip.myFirstInstance: Destroying... [id=eipalloc-0ec844d5a591abeee]
aws_eip.myFirstInstance: Destruction complete after 2s
aws_instance.myFirstInstance: Destroying... [id=i-0988a23b28aabd4d5]
aws_instance.myFirstInstance: Still destroying... [id=i-0988a23b28aabd4d5, 10s elapsed]
aws_instance.myFirstInstance: Still destroying... [id=i-0988a23b28aabd4d5, 20s elapsed]
aws_instance.myFirstInstance: Still destroying... [id=i-0988a23b28aabd4d5, 30s elapsed]
aws_instance.myFirstInstance: Still destroying... [id=i-0988a23b28aabd4d5, 40s elapsed]
aws_instance.myFirstInstance: Destruction complete after 40s
aws_security_group.security_jenkins_grp: Destroying... [id=sg-082d88a92ac0c12c4]
aws_security_group.security_jenkins_grp: Destruction complete after 1s

Destroy complete! Resources: 3 destroyed.
[root@ip-172-31-24-215 qarq90-terraform]#
```

"`terraform destroy`" Output



Deleting Instance



Terminated the Instance