Case Study 13

Basic Infrastructure Management with Terraform

Name: Aryan Deepak Saraf
Class: D15B
Roll no.: 53

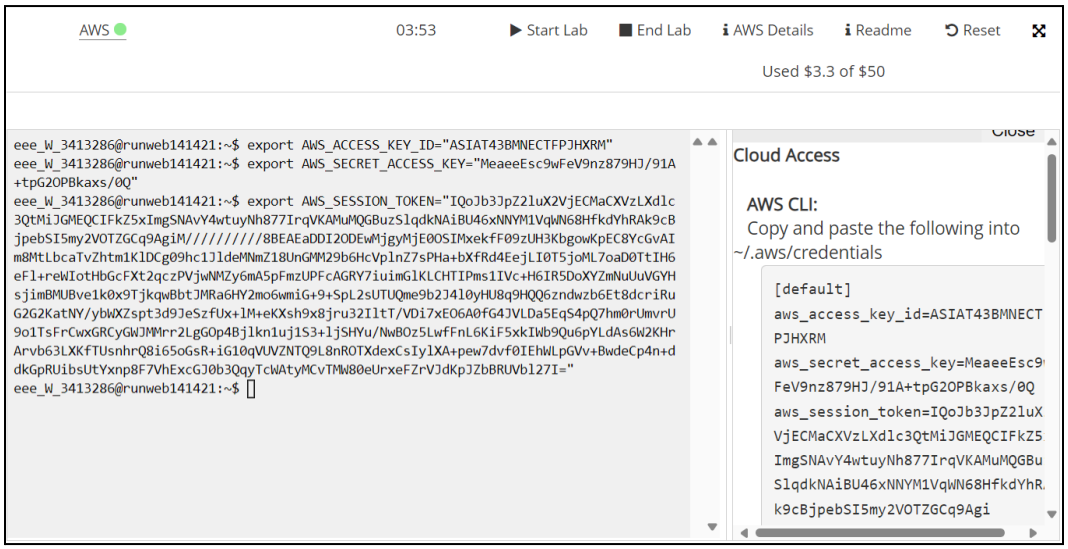
- **Concepts Used**: Terraform, AWS EC2, and S3.
- **Problem Statement**: "Use Terraform to create an EC2 instance and an S3 bucket. Store the EC2 instance's IP address in the S3 bucket."
- **Tasks**:
    - Write a simple Terraform script to provision an EC2 instance and an S3 bucket.
    - Use Terraform outputs to extract the EC2 instance's IP address.
    - Store the IP address in a text file in the S3 bucket.

SOLUTION

Set up AWS Learner Lab for Terraform

Log into your **AWS Learner Lab**.
Go to **Details** of your session to view **Access Key**, **Secret Key**, and **Session Token**. These credentials will be used for authenticating with AWS services via Terraform.
For temporary credentials, such as those from AWS Learner Lab, use the following environment variables after configuring your profile:

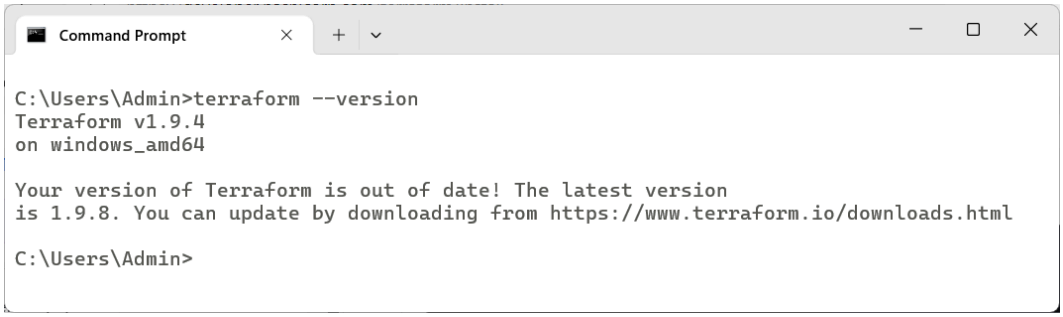
```
export AWS_ACCESS_KEY_ID="your_access_key"
export AWS_SECRET_ACCESS_KEY="your_secret_key"
export AWS_SESSION_TOKEN="your_session_token"
```



Install Terraform on Your Local System

If you haven't installed Terraform, follow the instructions below:

- Download Terraform: Terraform Download.
- Install Terraform and verify installation by running terraform --version.

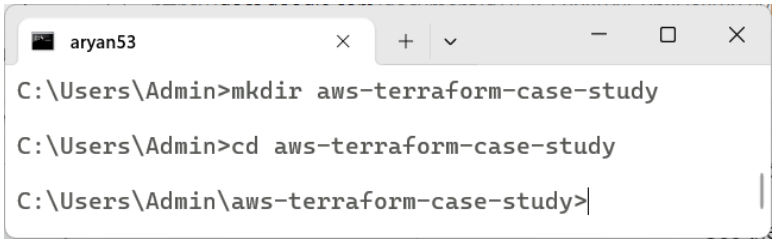


Set up Terraform Project Folder

Create a project folder and navigate to it:

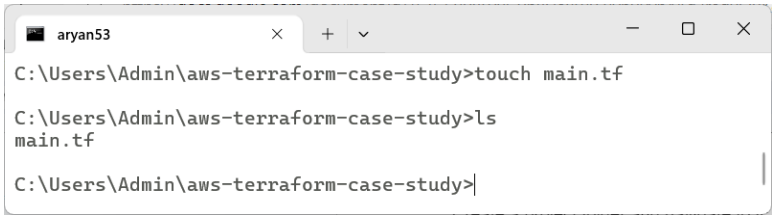
mkdir aws-terraform-case-study

cd aws-terraform-case-study



Create a Terraform configuration file (main.tf):

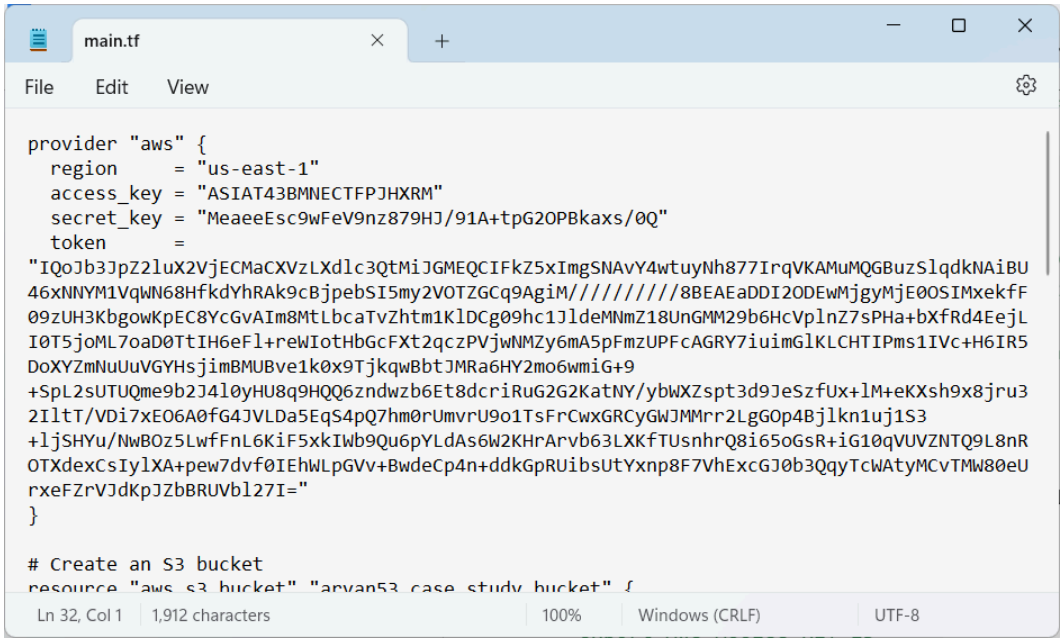
touch main.tf



Write the Terraform Configuration (main.tf)

```
provider "aws" {
  region     = "us-east-1"
  access_key = "ASIAT43BMNECTFPJHXRM"
  secret_key = "MeaeeEsc9wFeV9nz879HJ/91A+tpG2OPBkaxs/0Q"
  token      =
"IQoJb3JpZ2luX2VjECMaCXVzLXdlc3QtMiJGMEQCIFkZ5xImgSNAvY4wtuyNh877IrqVKAMuMQ
GBuzSlqdkNAiBU46xNNYM1VqWN68HfkdYhRAk9cBjpebSI5my2VOTZGCq9AgiM//////////8BEAEa
DDI2ODEwMjgyMjE0OSIMxekfF09zUH3KbgowKpEC8YcGvAIm8MtLbcaTvZhtm1KlDCg09hc1Jlde
MNmZ18UnGMM29b6HcVplnZ7sPHa+bXfRd4EejLI0T5joML7oaD0TtIH6eFl+reWIotHbGcFXt2qczP
VjwNMZy6mA5pFmzUPFcAGRY7iuimGlKLCHTIPms1IVc+H6IR5DoXYZmNuUuVGYHsjimBMUB
ve1k0x9TjkqwBbtJMRa6HY2mo6wmiG+9+SpL2sUTUQme9b2J4l0yHU8q9HQQ6zndwzb6Et8dcriRuG
2G2KatNY/ybWXZspt3d9JeSzfUx+lM+eKXsh9x8jru32IltT/VDi7xEO6A0fG4JVLDa5EqS4pQ7hm0rU
mvrU9o1TsFrCwxGRCyGWJMMrr2LgGOp4Bjlkn1uj1S3+ljSHYu/NwBOz5LwfFnL6KiF5xkIWb9Qu6
pYLdAs6W2KHrArvb63LXKfTUsnhrQ8i65oGsR+iG10qVUVZNTQ9L8nROTXdexCsIylXA+pew7dvf
```

```
0IEhWLpGVv+BwdeCp4n+ddkGpRUibsUtYxnp8F7VhExcGJ0b3QqyTcWAtyMCvTMW80eUrxeFZrV
JdKpJZbBRUVbl27I="
}

# Create an S3 bucket
resource "aws_s3_bucket" "aryan53_case_study_bucket" {
  bucket = "my-ec2-ip-bucket-${random_id.bucket_id.hex}"
}

resource "random_id" "bucket_id" {
  byte_length = 4
}

# Create a local file with the EC2 instance's IP address
resource "local_file" "ec2_ip_file" {
  content  = aws_instance.aryan_instance.public_ip
  filename = "ec2_instance_ip.txt"
}

# Upload the EC2 instance IP to the S3 bucket using aws_s3_object
resource "aws_s3_object" "ec2_ip_object" {
  bucket = aws_s3_bucket.aryan53_case_study_bucket.bucket
  key    = "ec2_instance_ip.txt"
  source = local_file.ec2_ip_file.filename
  acl    = "private"
}

# Create an EC2 instance
resource "aws_instance" "aryan_instance" {
  ami           = "ami-06b21ccaeff8cd686"  # Amazon Linux 2 AMI
  instance_type = "t2.micro"

  tags = {
    Name = "Terraform-EC2"
  }
}

# Output the public IP of the EC2 instance
output "instance_public_ip" {
  value = aws_instance.aryan_instance.public_ip
}
```



Key parts of the configuration:

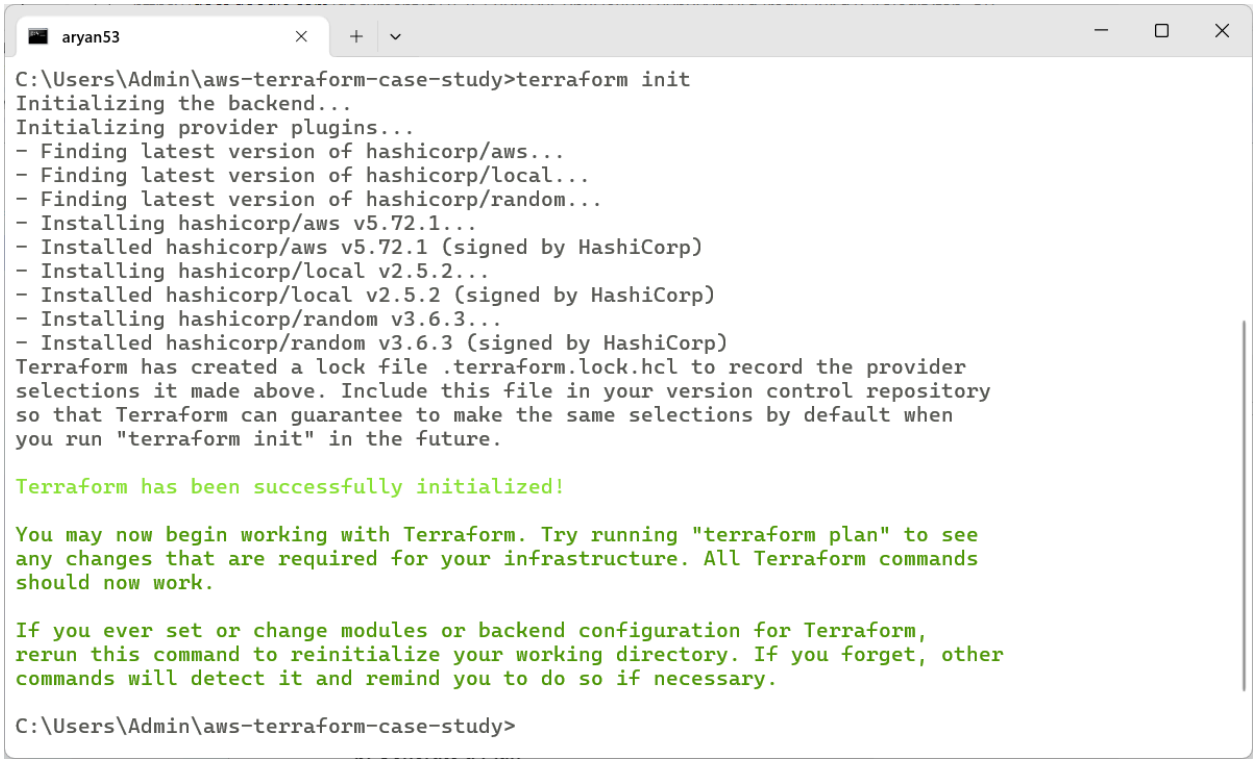
- **Provider Block**: Connects Terraform to AWS.
- **EC2 Instance**: Provisions a basic t2.micro instance using the specified AMI (Amazon Linux 2).
- **S3 Bucket**: Creates an S3 bucket with a unique name.
- **Local File**: Saves the public IP of the EC2 instance into a text file (ec2_instance_ip.txt).
- **S3 Object**: Uploads the text file containing the EC2 instance's IP address to the S3 bucket.

Initialize and Deploy with Terraform

a. Initialize the Terraform Working Directory

Before applying the configuration, initialize the directory:

terraform init

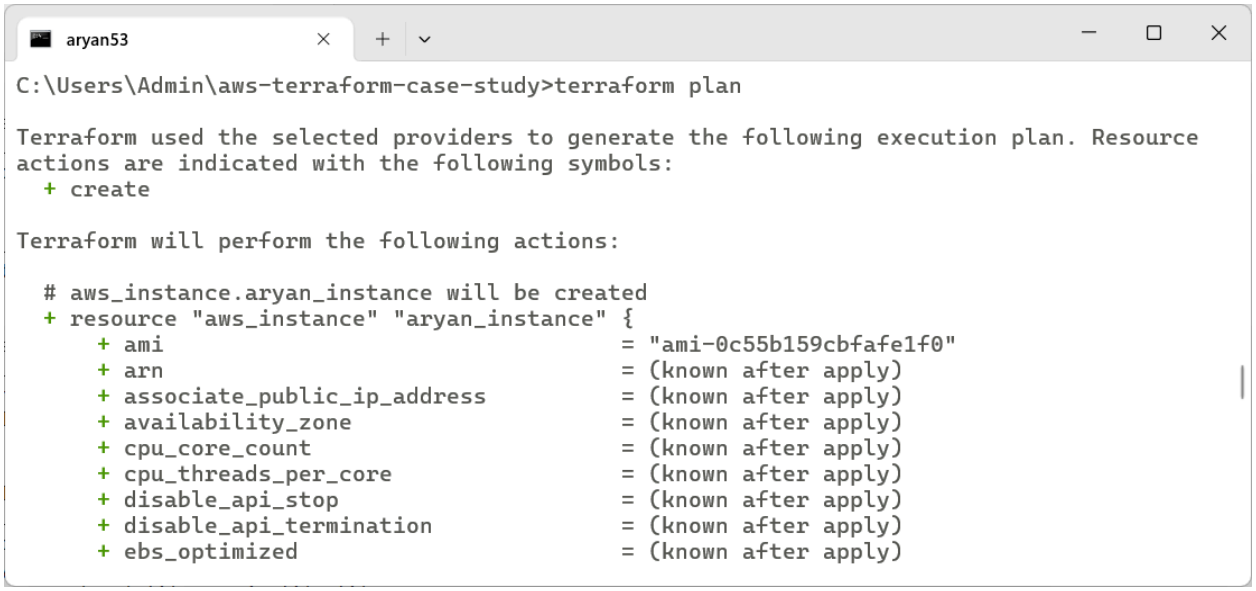


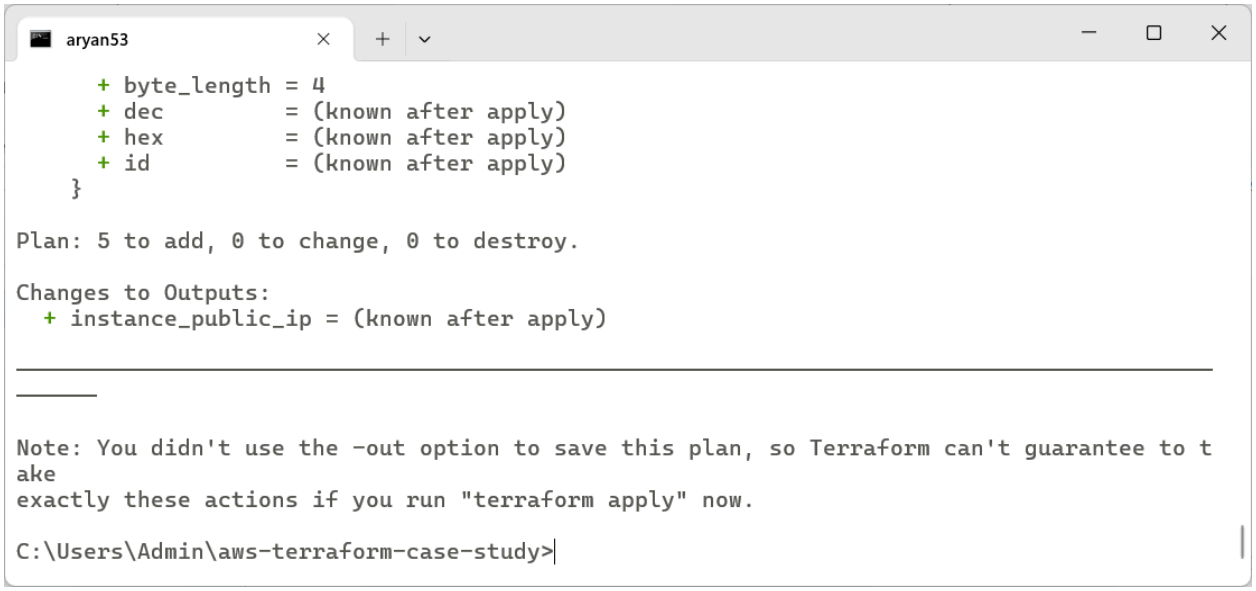
This command downloads the necessary provider plugins (AWS in this case).

b. Generate a Plan

To preview the actions that Terraform will take:

terraform plan





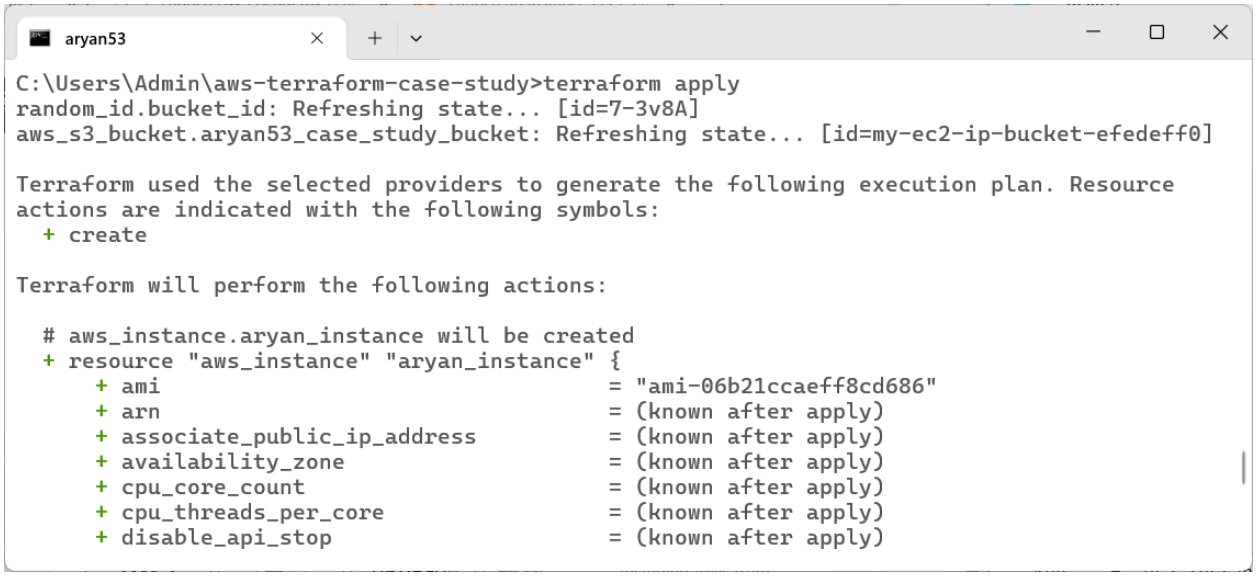
It will show you the resources Terraform plans to create.

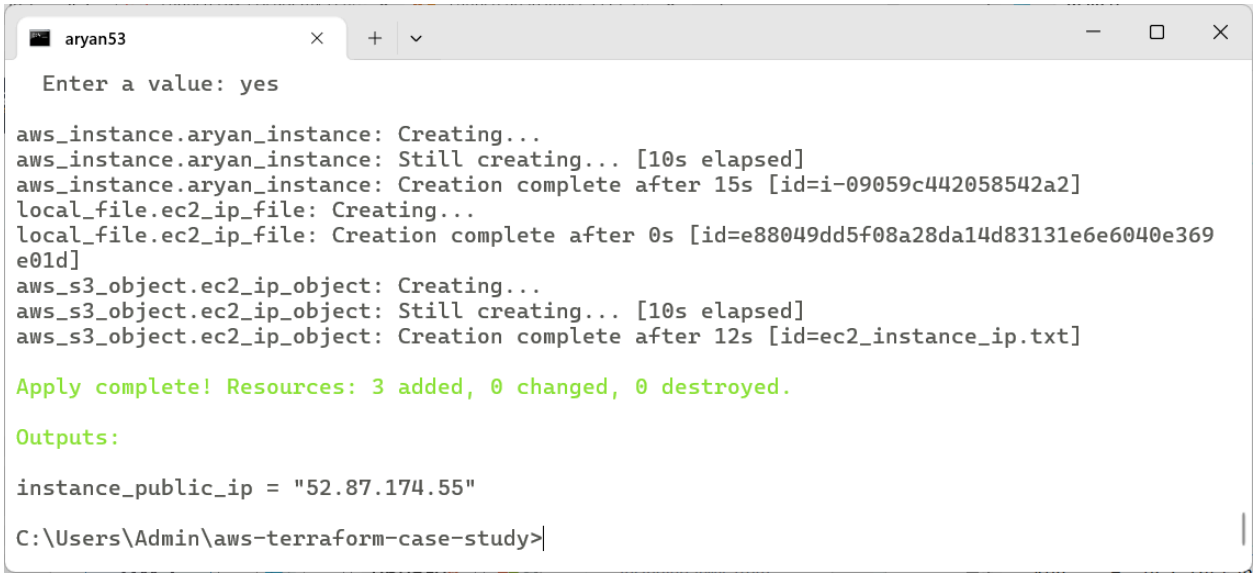
c. Apply the Terraform Configuration

Run the following command to provision the EC2 instance and S3 bucket:

terraform apply

Type yes when prompted to confirm.





d. Verify the IP Address in S3

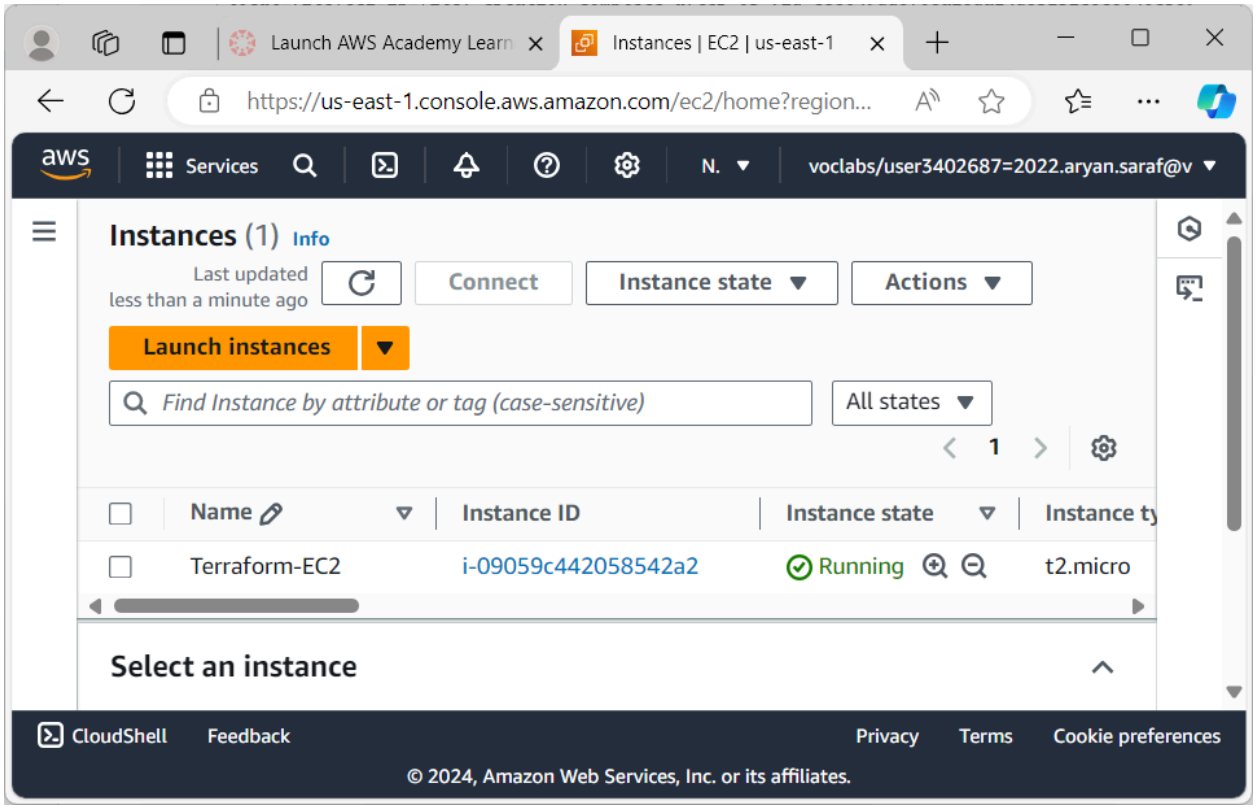
- Once the infrastructure is created, the EC2 instance's IP address will be saved in the file ec2_instance_ip.txt and uploaded to the S3 bucket.

Verify the Results

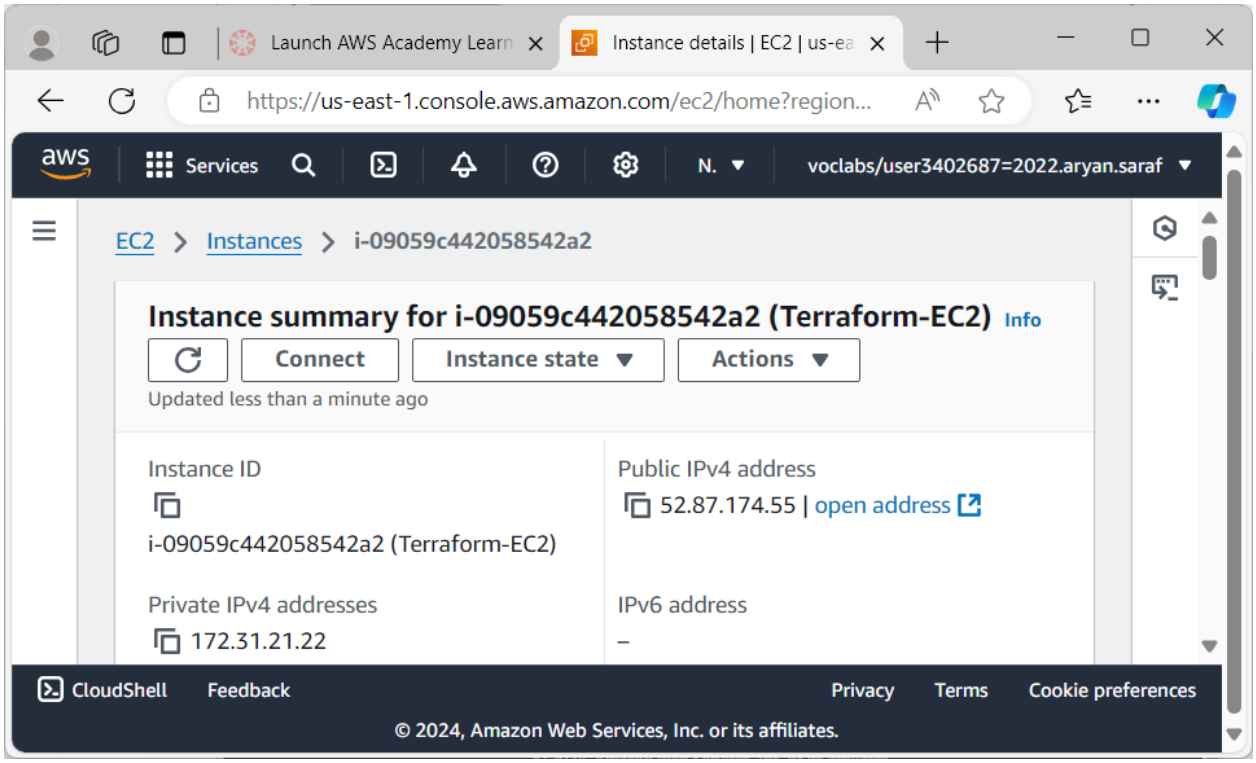
After the Terraform script has completed, verify the following:

Check the S3 bucket: Open the AWS Console and navigate to the S3 service. Verify that the EC2 instance's public IP is stored in the ec2_instance_ip.txt file.
(You can also use the output shown in the Terraform terminal to verify the public IP.)

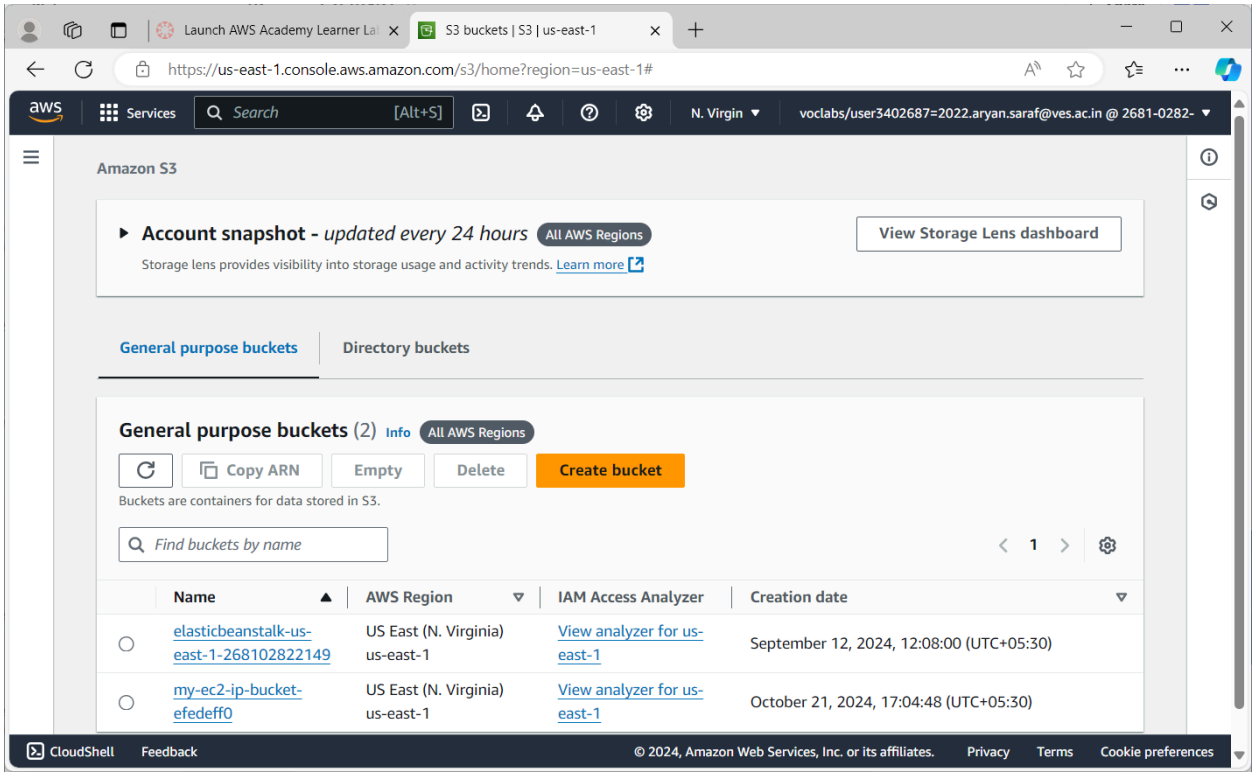
Go to the EC2 instances and check whether our specified instance through terraform is same or not.



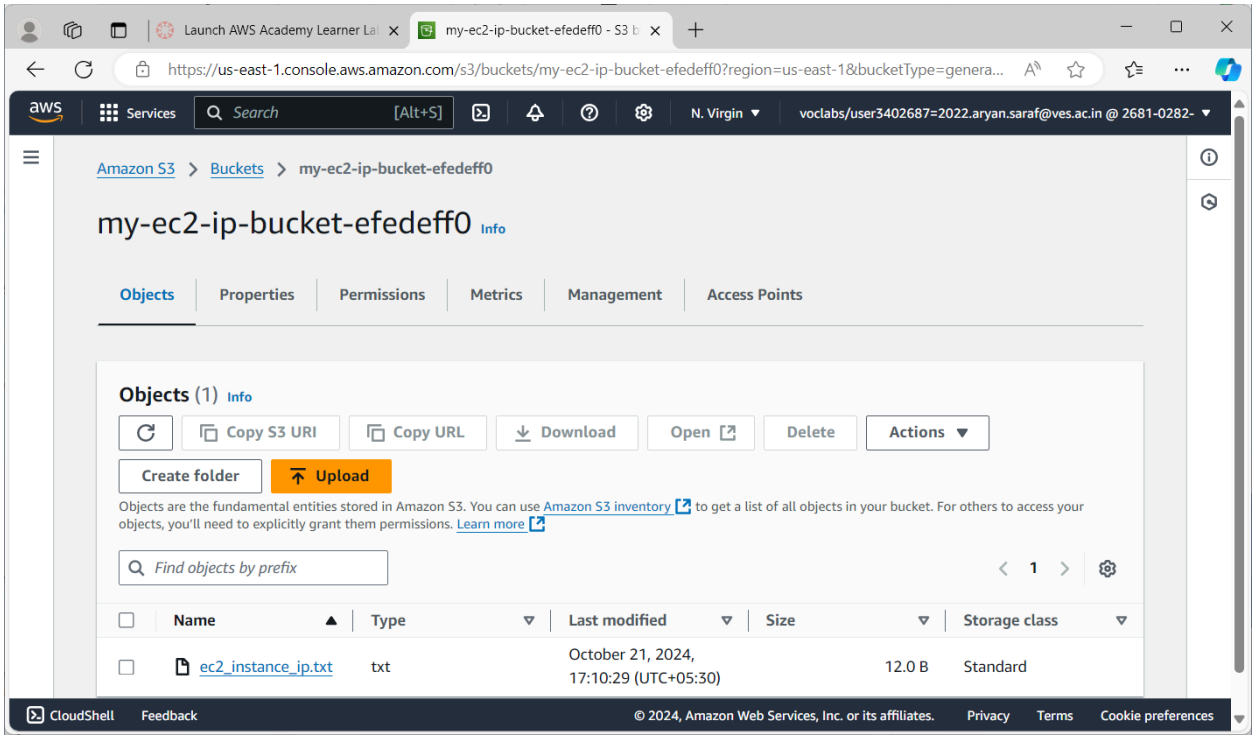
Check the public key of the EC2 instance.



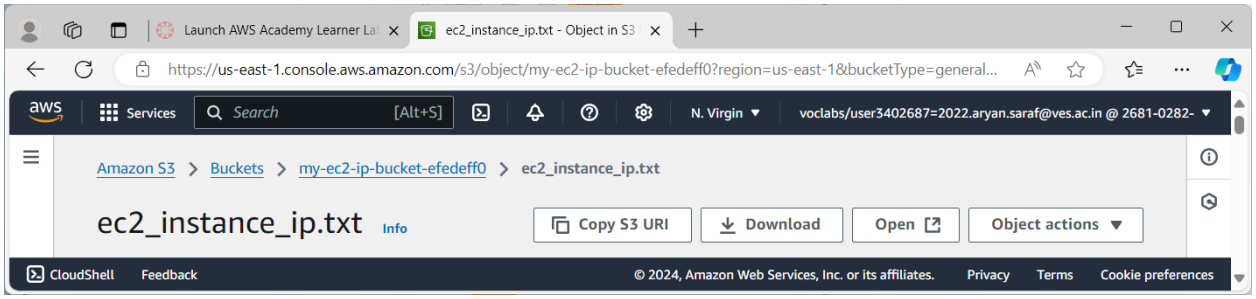
Go to the s3 bucket and check whether our specified s3 bucket through terraform is the same or not.



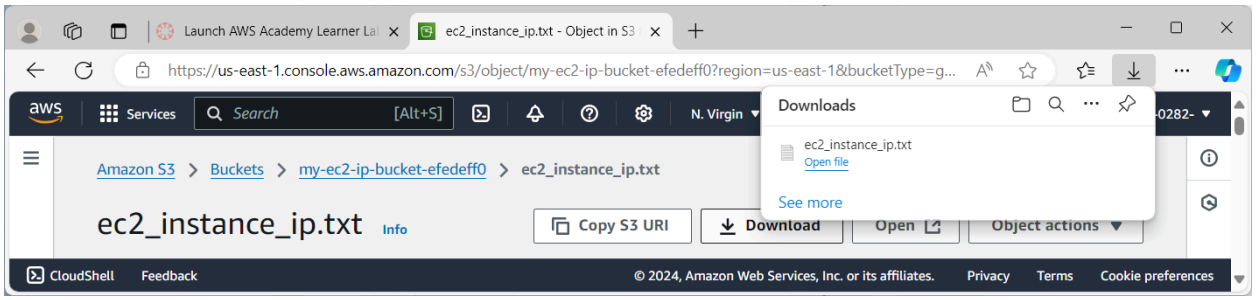
Navigate inside the s3 bucket in which there will be one txt file.



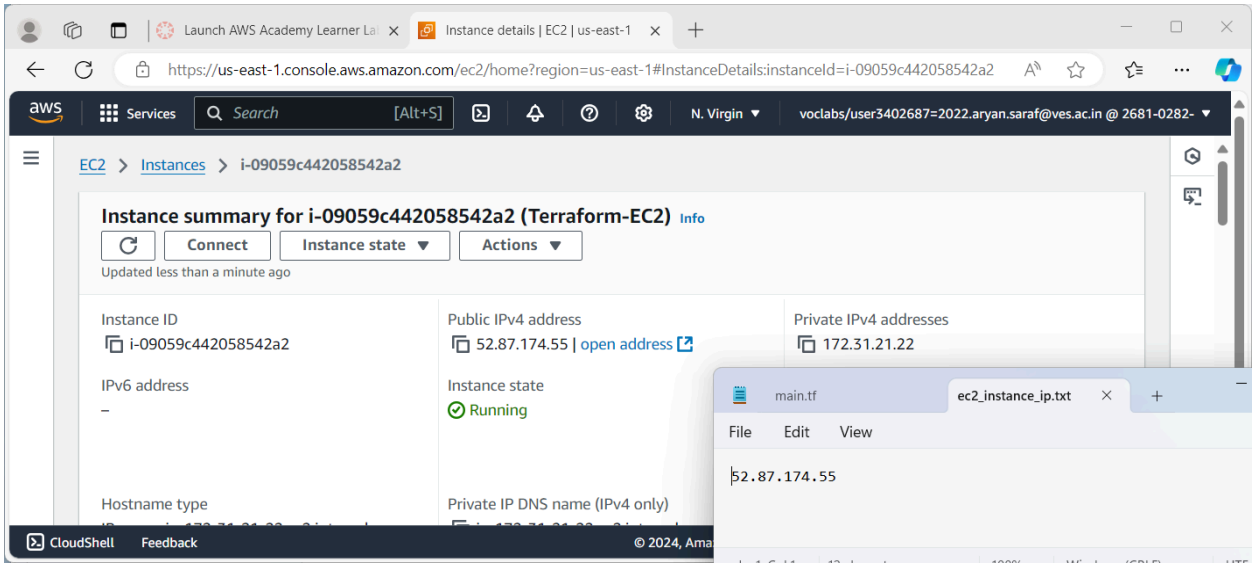
Open txt file.



Download the txt file.



Match the public ip address of the ec2 instance with the ip address in the txt file.



Clean Up Resources

Once you're done with the lab, ensure to delete the resources to avoid any potential charges:

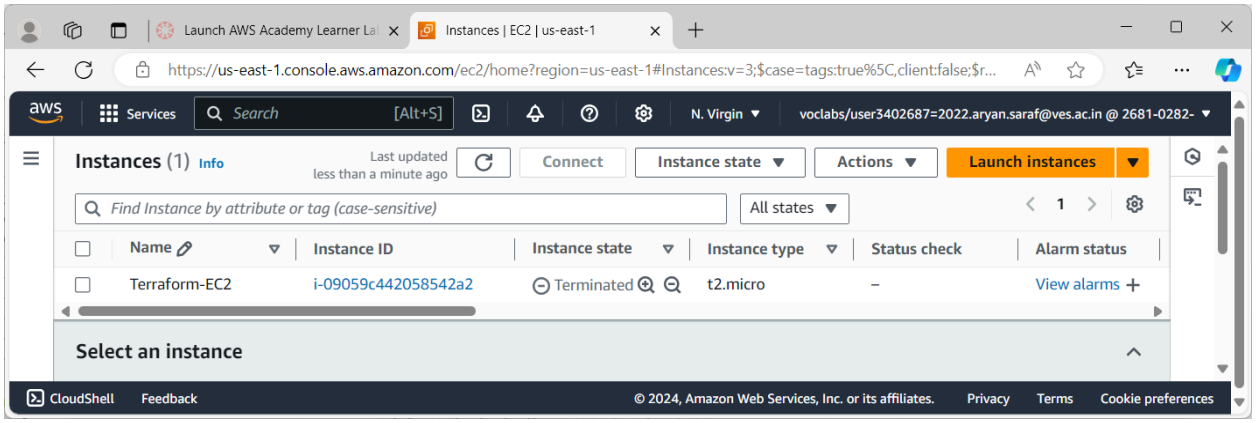
terraform destroy

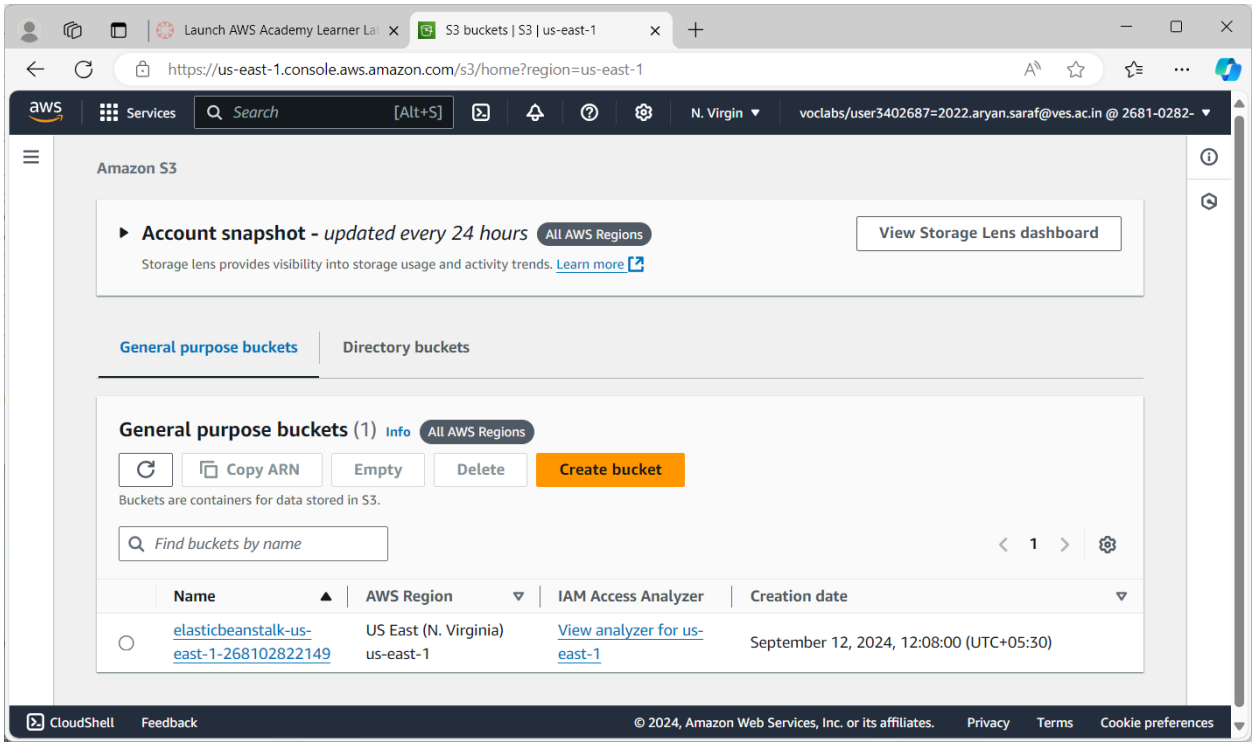
Type yes to confirm.

```
aryan53
C:\Users\Admin\aws-terraform-case-study>terraform destroy
random_id.bucket_id: Refreshing state... [id=7-3v8A]
aws_s3_bucket.aryan53_case_study_bucket: Refreshing state... [id=my-ec2-ip-bucket-efedeff0]
aws_instance.aryan_instance: Refreshing state... [id=i-09059c442058542a2]
local_file.ec2_ip_file: Refreshing state... [id=e88049dd5f08a28da14d83131e6e6040e369e01d]
aws_s3_object.ec2_ip_object: Refreshing state... [id=ec2_instance_ip.txt]

Terraform used the selected providers to generate the following execution plan. Resource actions are indic
ated with the
following symbols:
  - destroy

Terraform will perform the following actions:

  # aws_instance.aryan_instance will be destroyed
  - resource "aws_instance" "aryan_instance" {
      - ami                                  = "ami-06b21ccaeff8cd686" -> null
      - arn                                  = "arn:aws:ec2:us-east-1:268102822149:instance/i-09059c442058
542a2" -> null
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "us-east-1a" -> null
      - cpu_core_count                       = 1 -> null
      - cpu_threads_per_core                 = 1 -> null
```

```
aryan53
  Enter a value: yes

aws_s3_object.ec2_ip_object: Destroying... [id=ec2_instance_ip.txt]
aws_s3_object.ec2_ip_object: Destruction complete after 1s
local_file.ec2_ip_file: Destroying... [id=e88049dd5f08a28da14d83131e6e6040e369e01d]
aws_s3_bucket.aryan53_case_study_bucket: Destroying... [id=my-ec2-ip-bucket-efedeff0]
local_file.ec2_ip_file: Destruction complete after 0s
aws_instance.aryan_instance: Destroying... [id=i-09059c442058542a2]
aws_s3_bucket.aryan53_case_study_bucket: Destruction complete after 1s
random_id.bucket_id: Destroying... [id=7-3v8A]
random_id.bucket_id: Destruction complete after 0s
aws_instance.aryan_instance: Still destroying... [id=i-09059c442058542a2, 10s elapsed]
aws_instance.aryan_instance: Still destroying... [id=i-09059c442058542a2, 20s elapsed]
aws_instance.aryan_instance: Still destroying... [id=i-09059c442058542a2, 30s elapsed]
aws_instance.aryan_instance: Still destroying... [id=i-09059c442058542a2, 40s elapsed]
aws_instance.aryan_instance: Still destroying... [id=i-09059c442058542a2, 50s elapsed]
aws_instance.aryan_instance: Destruction complete after 52s

Destroy complete! Resources: 5 destroyed.

C:\Users\Admin\aws-terraform-case-study>
```

Check whether the resources are cleared or not.





Conclusion

This experiment covers important topics related to cloud infrastructure management, automation, and provisioning using Terraform. It emphasizes key AWS services like EC2 and S3, showing how infrastructure can be efficiently automated and deployed at scale.