**Name:** Atif Ansari

**Roll no:** 04

**Class:** D15B

**Assignment 2**

**Aim:**

The primary aim of this Terraform setup is to deploy a cloud infrastructure on AWS that integrates multiple services: an Amazon S3 bucket for object storage, an Amazon SQS queue for message queuing, and an AWS Lambda function that processes events triggered by S3. This automation through Infrastructure as Code (IaC) streamlines resource management, enhances scalability, and promotes operational efficiency.

**Theory:**

**Infrastructure as Code (IaC)**:

* IaC is a modern approach to managing IT infrastructure using machine-readable definition files. Terraform is one of the most popular tools for IaC, allowing users to provision and manage cloud resources declaratively.
* **Key Concepts**:
  + **Providers**: Terraform interacts with cloud services through providers. In this setup, the AWS provider is used to manage resources in the AWS cloud.
  + **Resources**: The core building blocks in Terraform; each resource corresponds to an infrastructure component, like S3 buckets, SQS queues, or Lambda functions.
  + **State Management**: Terraform maintains a state file to keep track of the resources it manages, enabling updates and deletions without manual intervention.

**AWS Services**:

1. **Amazon S3**: Object storage service that allows for storing and retrieving any amount of data. It supports versioning and can trigger events on object creation.
2. **Amazon SQS**: A fully managed message queuing service that enables decoupling of microservices, distributed systems, and serverless applications.
3. **AWS Lambda**: A serverless compute service that runs code in response to events and automatically manages the compute resources for you.

**Implementation:**

1. **Creating the Lambda Function**:
   * The function is defined with a specified runtime (Node.js 14.x) and a timeout of 10 seconds.
   * It is configured to process messages from an SQS queue and is triggered by events from the S3 bucket.
   * An IAM role is created to allow the Lambda function to interact with both S3 and SQS.
2. **Creating the SQS Queue**:
   * An SQS queue is defined to receive messages that the Lambda function will process.
3. **Creating the S3 Bucket**:
   * An S3 bucket is created with versioning enabled, and a notification is set up to trigger the Lambda function whenever a new object is created.

**Performing Terraform Commands:**

1. **Terraform Init**:
   * Initializes the working directory containing Terraform configuration files. It sets up the necessary providers and modules.
   * Command: terraform init
2. **Terraform Plan**:
   * Creates an execution plan, showing what actions Terraform will take to reach the desired state defined in the configuration.
   * Command: terraform plan
3. **Terraform Apply**:
   * Applies the changes required to reach the desired state of the configuration, creating or updating the infrastructure.
   * Command: terraform apply
4. **Terraform Destroy**:
   * Destroys all the resources defined in the Terraform configuration, cleaning up the AWS environment.
   * Command: terraform destroy

**Folder Structure of main.tf File**

* The structure of the main.tf file should logically organize resources, possibly separating them into different files or sections for better management (optional).

**Code:**

provider "aws" {

region = "ap-south-1"

}

# S3 Bucket

resource "aws\_s3\_bucket" "atifbucket03" {

bucket = "my-terraform-s3-bucket"

acl = "private"

versioning {

enabled = true

}

}

# SQS Queue

resource "aws\_sqs\_queue" "sqs\_atif" {

name = "my-terraform-sqs-queue"

}

# Lambda Function

resource "aws\_lambda\_function" "atif03" {

function\_name = "s3-to-sqs-lambda"

role = aws\_iam\_role.lambda\_exec.arn

handler = "index.handler"

runtime = "nodejs14.x"

timeout = 10

filename = "lambda.zip" # Path to the Lambda zip file

environment {

variables = {

QUEUE\_URL = aws\_sqs\_queue.sqs\_atif.id

}

}

}

# IAM Role for Lambda execution

resource "aws\_iam\_role" "lambda\_exec" {

name = "lambda\_exec\_role"

assume\_role\_policy = jsonencode({

Version = "2012-10-17",

Statement = [{

Action = "sts:AssumeRole",

Effect = "Allow",

Principal = {

Service = "lambda.amazonaws.com"

}

}]

})

}

# IAM Role Policy for Lambda (grant permissions to interact with S3 and SQS)

resource "aws\_iam\_role\_policy" "lambda\_exec\_policy" {

role = aws\_iam\_role.lambda\_exec.id

policy = jsonencode({

Version = "2012-10-17",

Statement = [

{

Action = [

"sqs:SendMessage"

],

Effect = "Allow",

Resource = aws\_sqs\_queue.sqs\_atif.arn

},

{

Action = [

"s3:GetObject"

],

Effect = "Allow",

Resource = "${aws\_s3\_bucket.atifbucket03.arn}/\*"

}

]

})

}

# S3 Bucket Notification to trigger Lambda on object creation

resource "aws\_s3\_bucket\_notification" "s3\_notification" {

bucket = aws\_s3\_bucket.atifbucket03.id

lambda\_function {

lambda\_function\_arn = aws\_lambda\_function.atif03.arn

events = ["s3:ObjectCreated:\*"]

}

}

# Lambda Permission for S3 to invoke the Lambda function

resource "aws\_lambda\_permission" "allow\_s3" {

statement\_id = "AllowS3InvokeLambda"

action = "lambda:InvokeFunction"

function\_name = aws\_lambda\_function.atif03.function\_name

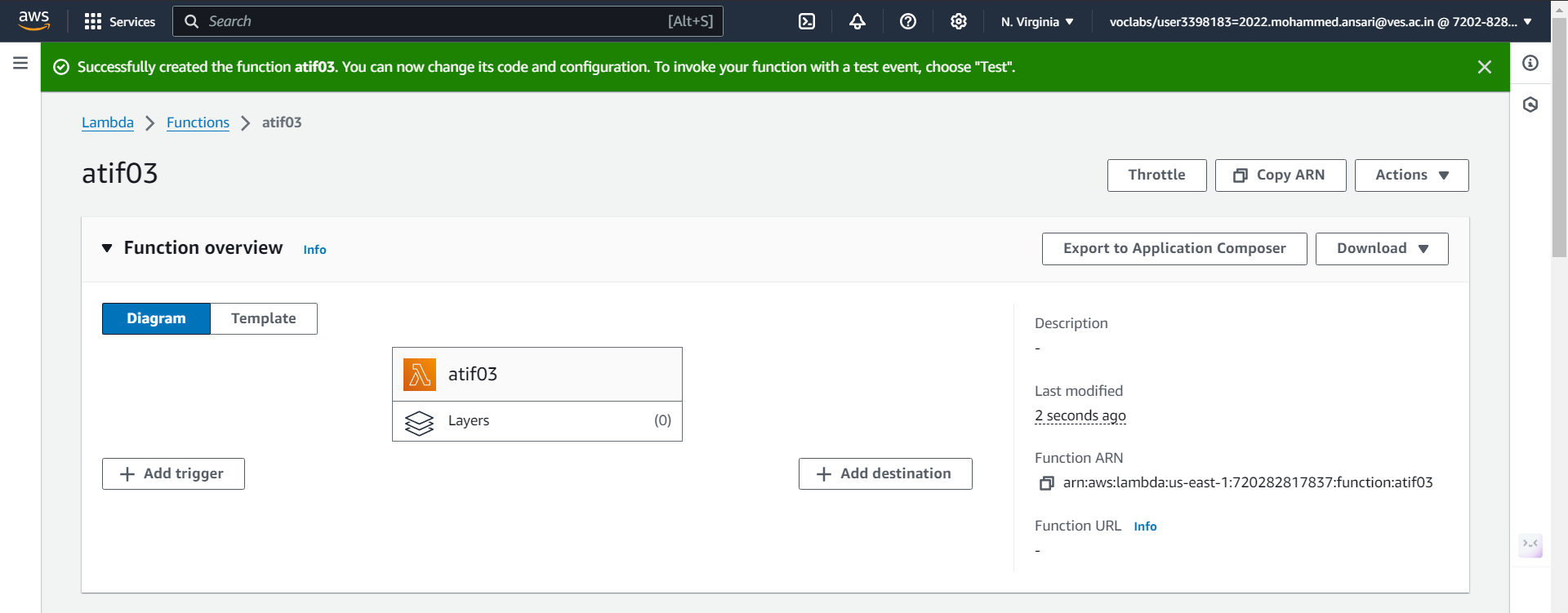
principal = "s3.amazonaws.com"

source\_arn = aws\_s3\_bucket.atifbucket03.arn

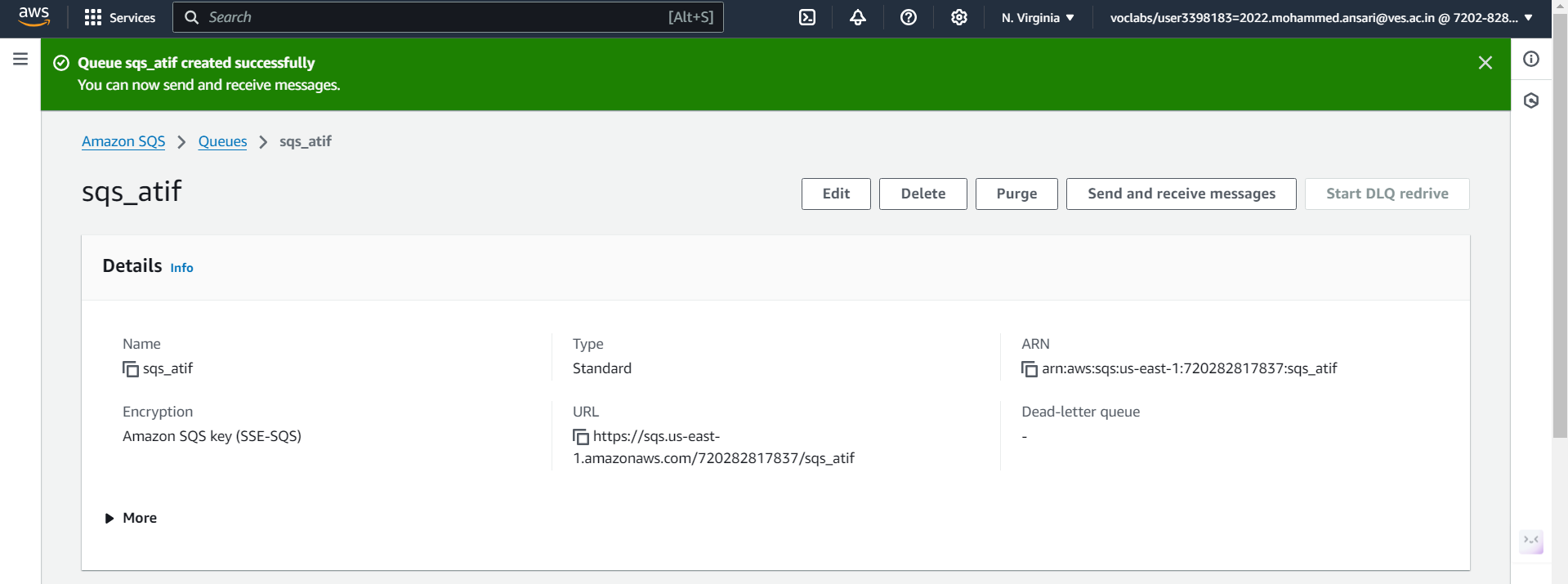
}

**Implementation:**

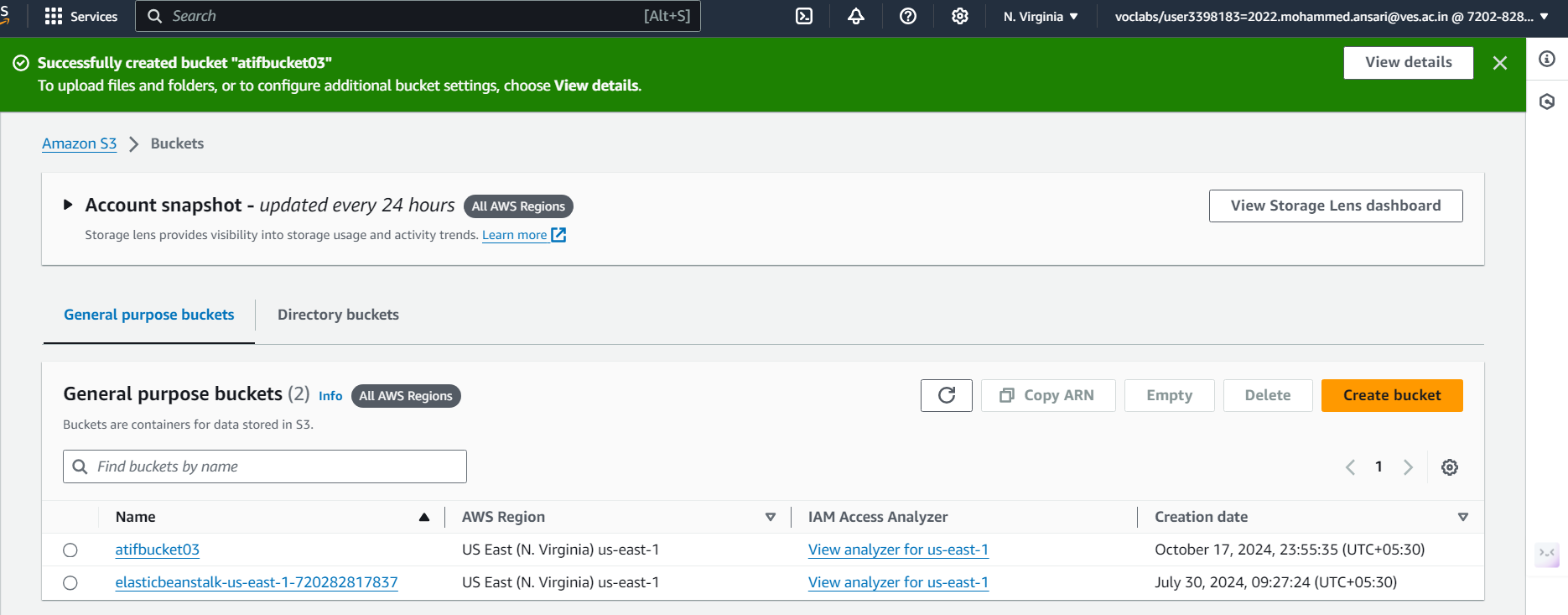
1. Creating Lambda Function



1. Creating Sqs Queue

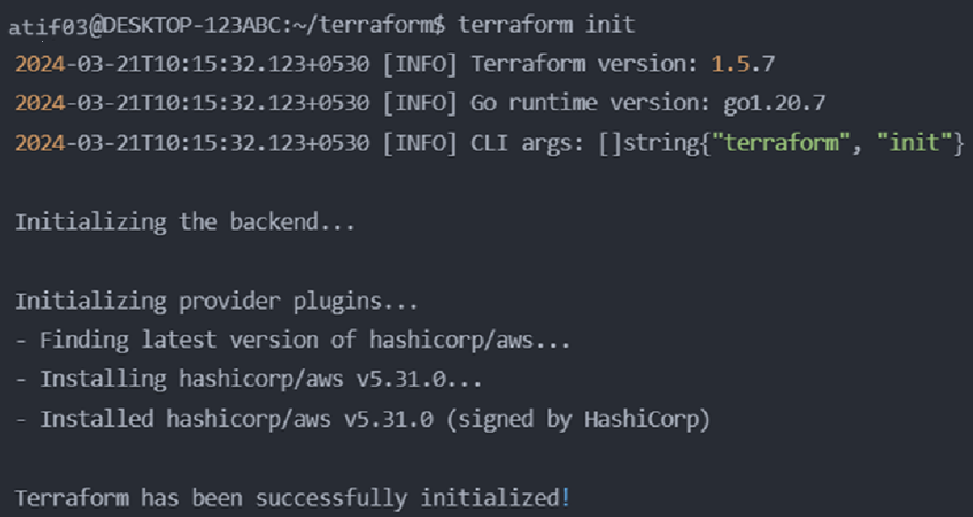


1. Creating S3 Bucket

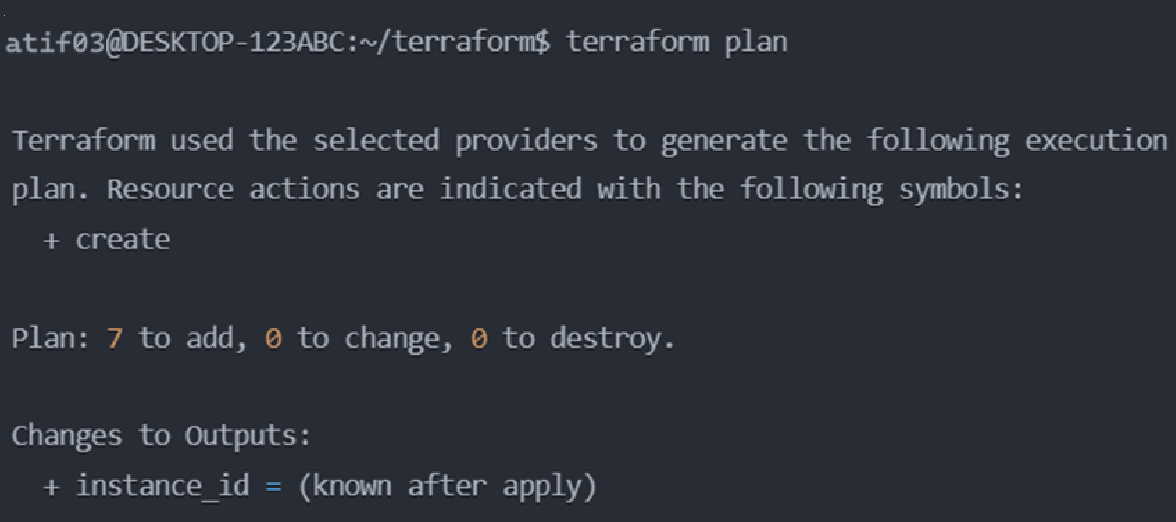


Performing Terraform commands

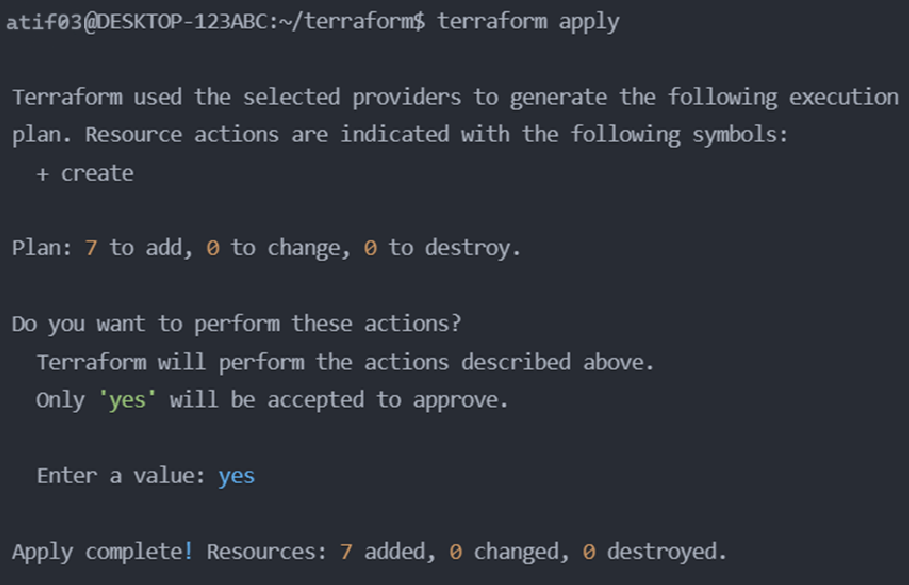
1. Terraform init



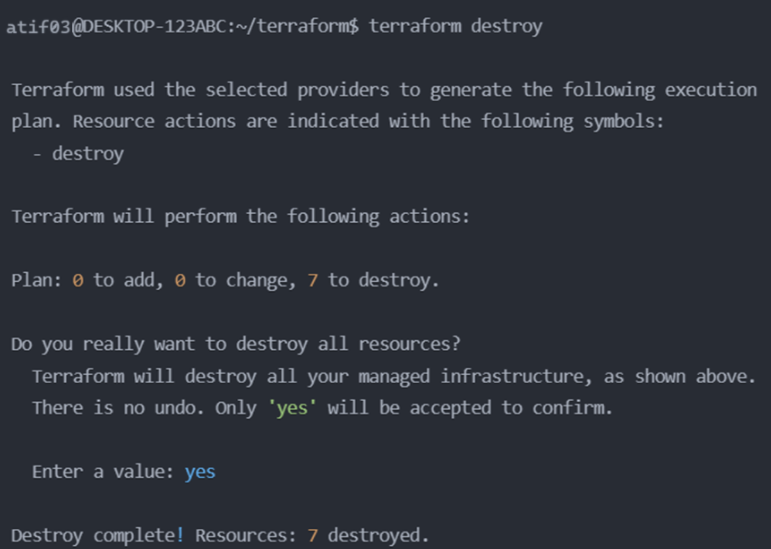
1. Terraform plan



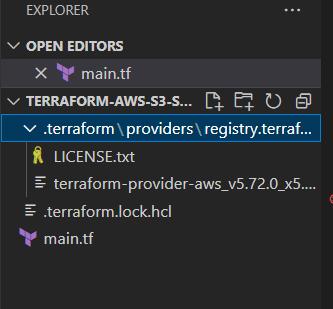
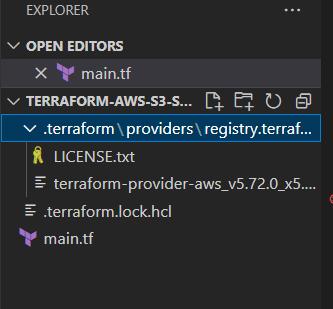
1. Terraform apply



1. Terraform destroy



Folder structure of main.tf file



**Conclusion:**

In this experiment, we successfully deployed an AWS infrastructure using Terraform, integrating essential services such as Amazon S3, SQS, and Lambda. By leveraging Terraform's infrastructure as code capabilities, we were able to automate the provisioning and configuration of cloud resources, ensuring consistency and reproducibility in our deployments.