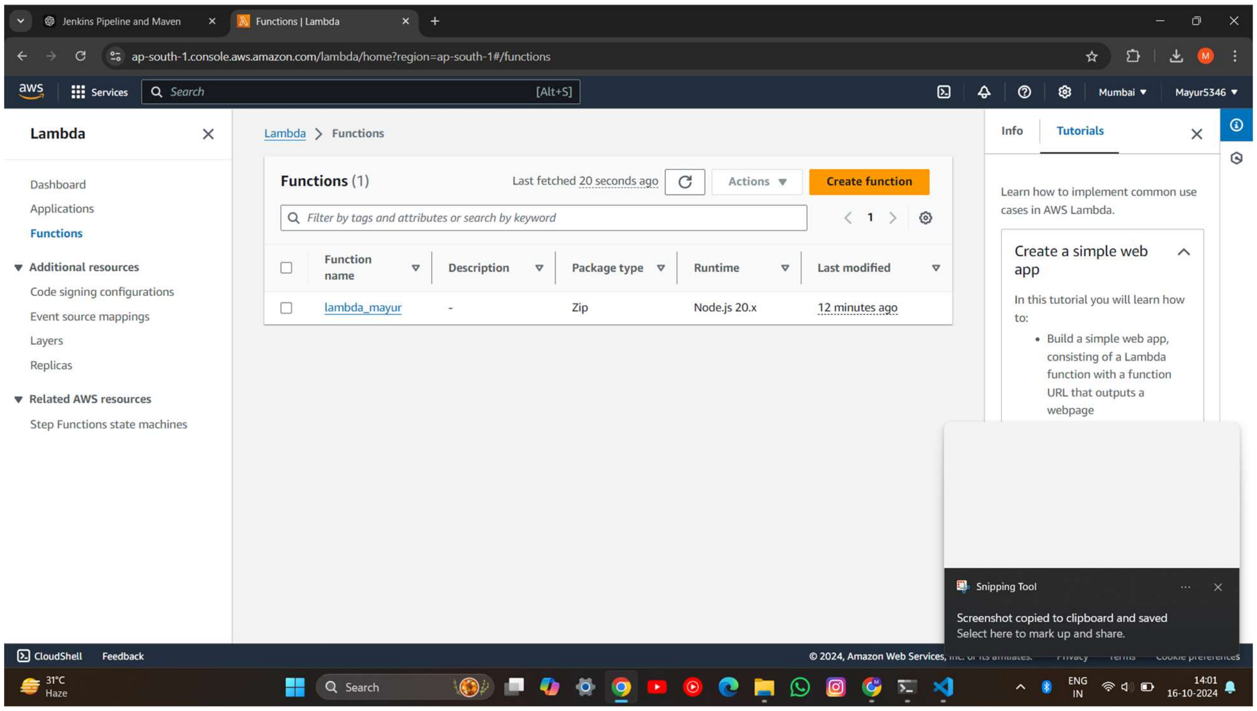
Mayur Jaiswal
D15B
26
Adv Devops Assignment 2

Code:
```
provider "aws" {
  region = "ap-south-1"
}

# S3 Bucket
resource "aws_s3_bucket" "s3mayur" {
  bucket = "my-terraform-s3-bucket"
  acl    = "private"

  versioning {
    enabled = true
  }
}

# SQS Queue
resource "aws_sqs_queue" "sqsmayur" {
  name = "my-terraform-sqs-queue"
}

# Lambda Function
resource "aws_lambda_function" "lambda_mayur" {
  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.sqsmayur.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.sqsmayur.arn
      },
      {
        Action = [
          "s3:GetObject"
        ],
        Effect   = "Allow",
        Resource = "${aws_s3_bucket.s3mayur.arn}/*"
      }
    ]
  })
}

# S3 Bucket Notification to trigger Lambda on object creation
resource "aws_s3_bucket_notification" "s3_notification" {
  bucket = aws_s3_bucket.s3mayur.id

  lambda_function {
    lambda_function_arn = aws_lambda_function.lambda_mayur.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.lambda_mayur.function_name
  principal     = "s3.amazonaws.com"

  source_arn = aws_s3_bucket.s3mayur.arn
}
```

Implementation:

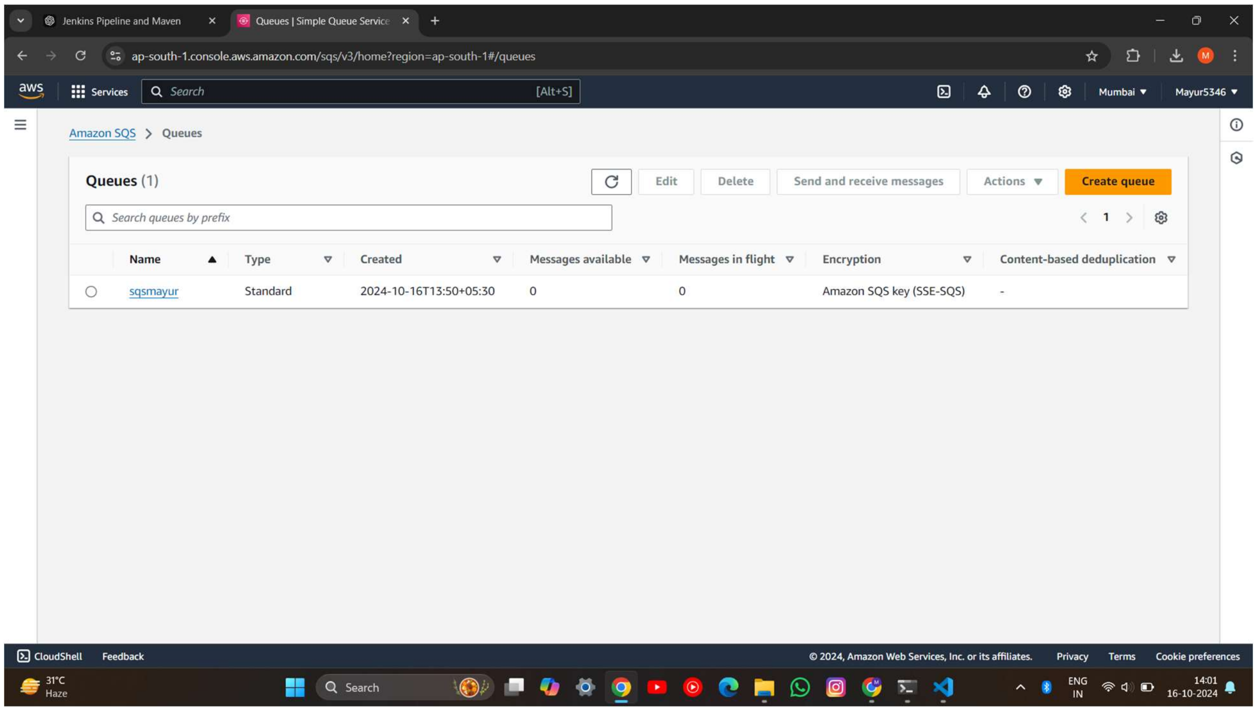
1. Creating Lambda Function



Functions (1)

	Function name	Description	Package type	Runtime	Last modified
☐	lambda_mayur	-	Zip	Node.js 20.x	12 minutes ago

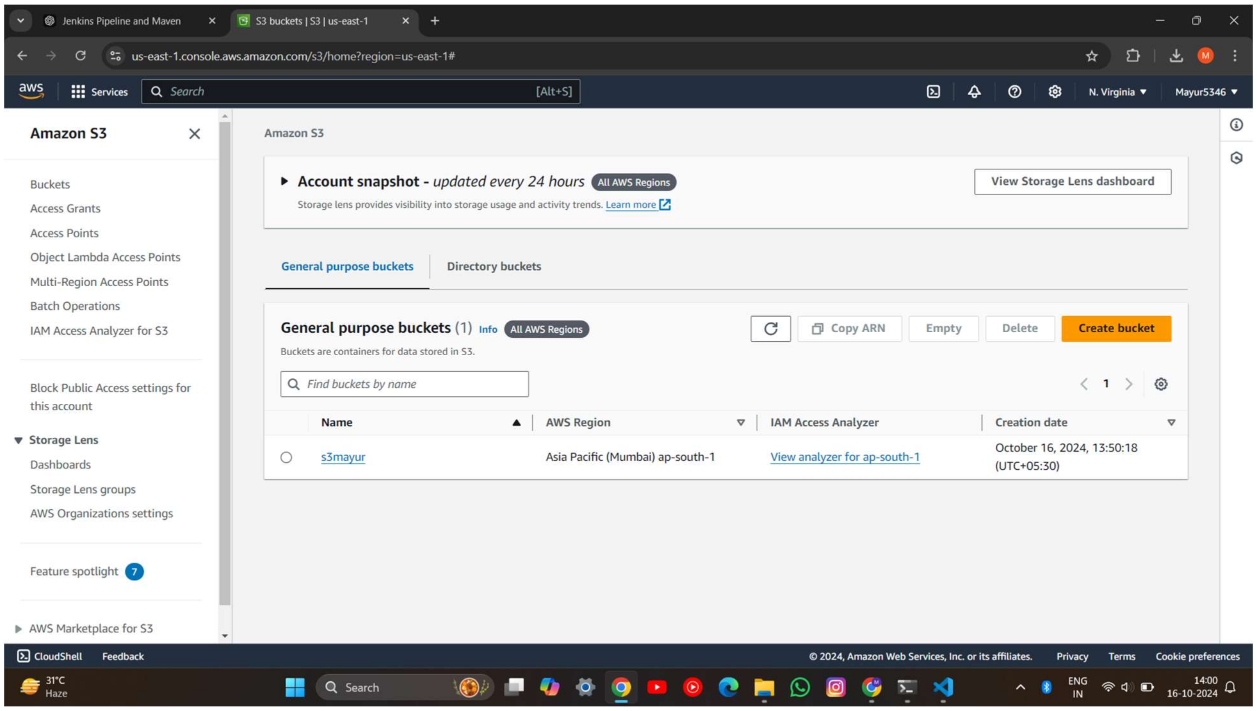
2. Creating Sqs Queue


Jenkins Pipeline and Maven
Queues | Simple Queue Service
ap-south-1.console.aws.amazon.com/sqs/v3/home?region=ap-south-1#/queues
Services
Search
[Alt+S]
Mumbai
Mayur5346
Amazon SQS > Queues
Edit
Delete
Send and receive messages
Actions
Create queue
Search queues by prefix
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
31°C Haze
Search
ENG IN
14:01 16-10-2024

Queues (1)

Name	Type	Created	Messages available	Messages in flight	Encryption	Content-based deduplication
sqsmayur	Standard	2024-10-16T13:50+05:30	0	0	Amazon SQS key (SSE-SQS)	-

3. Creating S3 Bucket



Account snapshot - *updated every 24 hours* All AWS Regions

Storage lens provides visibility into storage usage and activity trends. Learn more

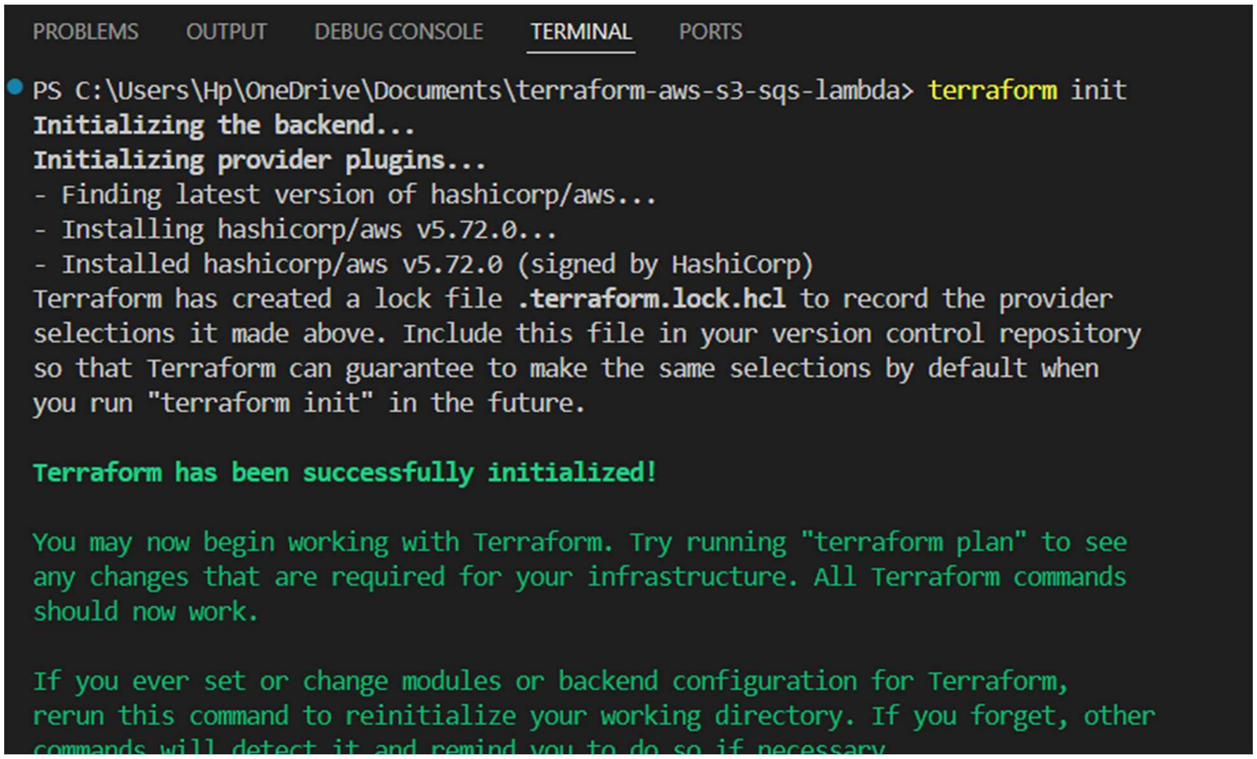
General purpose buckets (1) Info All AWS Regions

Buckets are containers for data stored in S3.

Name	AWS Region	IAM Access Analyzer	Creation date
s3mayur	Asia Pacific (Mumbai) ap-south-1	View analyzer for ap-south-1	October 16, 2024, 13:50:18 (UTC+05:30)

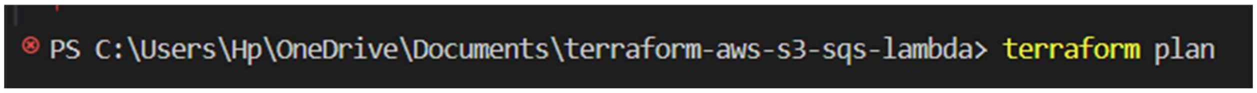
Performing Terraform commands

1. Terraform init

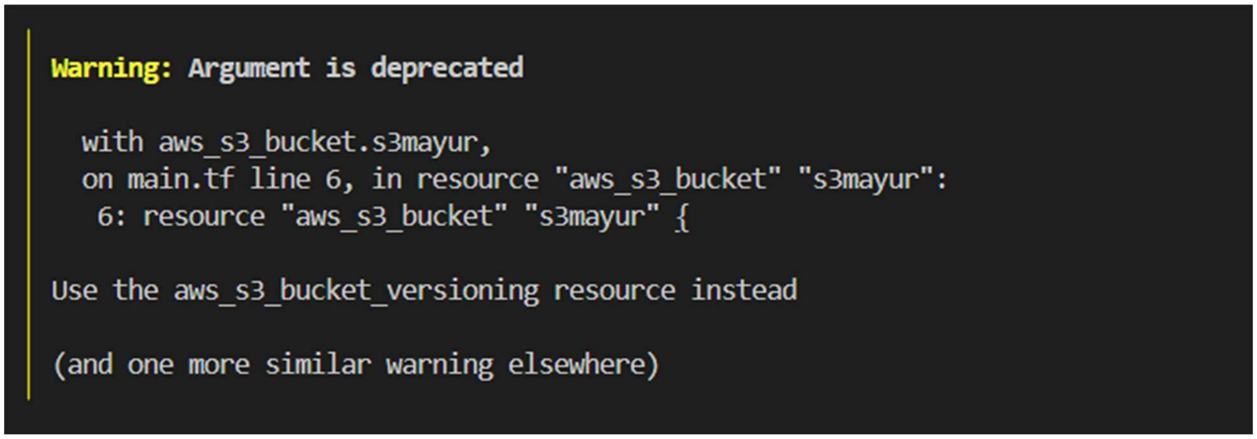


```
PS C:\Users\Hp\OneDrive\Documents\terraform-aws-s3-sqs-lambda> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.72.0...
- Installed hashicorp/aws v5.72.0 (signed by HashiCorp)
Terraform has created 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.
```

2. Terraform plan

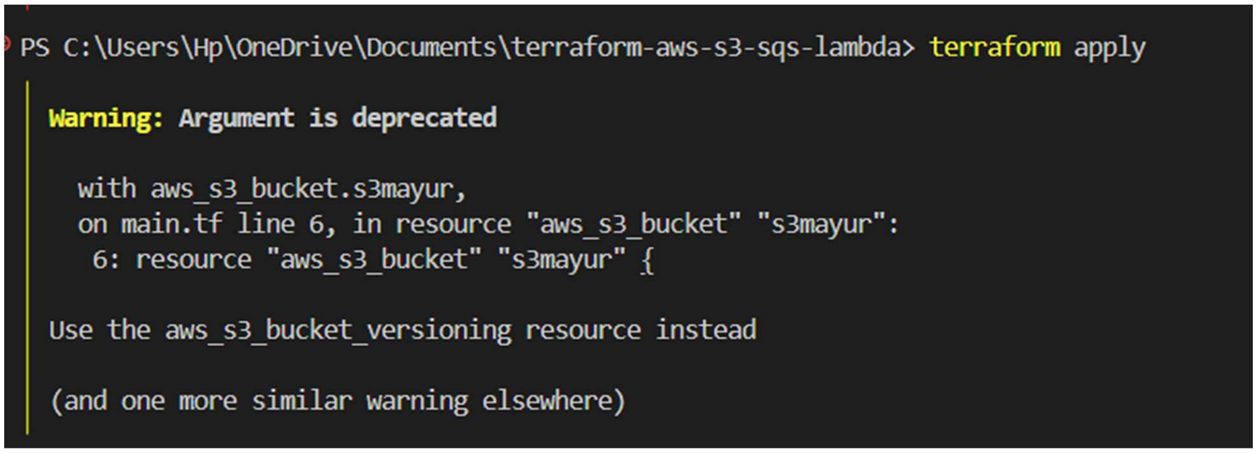


```
PS C:\Users\Hp\OneDrive\Documents\terraform-aws-s3-sqs-lambda> terraform plan
```



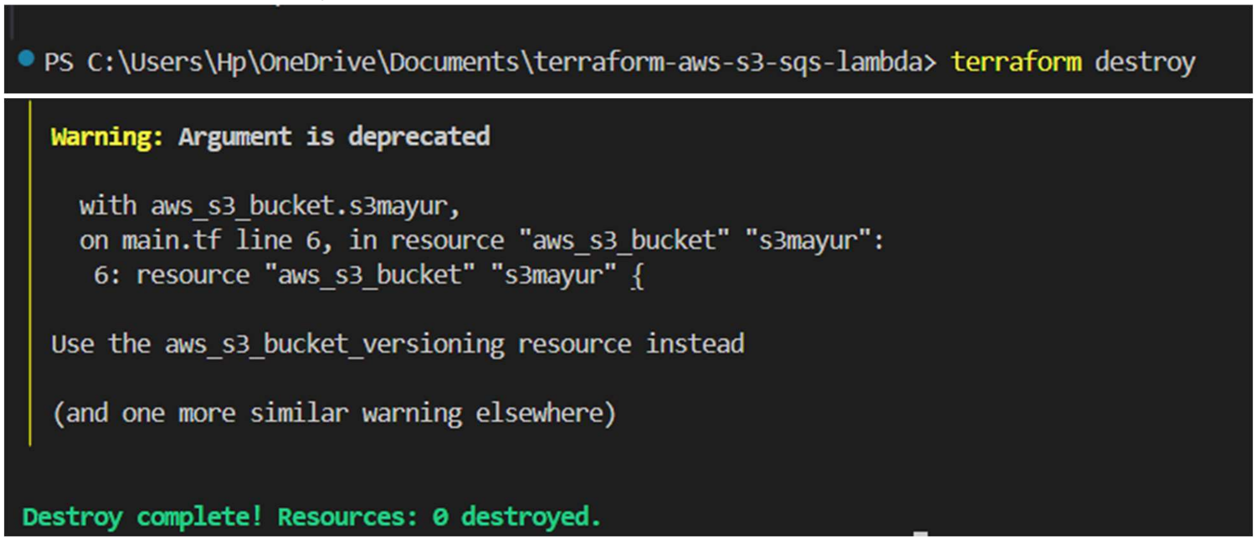
```
Warning: Argument is deprecated

  with aws_s3_bucket.s3mayur,
  on main.tf line 6, in resource "aws_s3_bucket" "s3mayur":
   6: resource "aws_s3_bucket" "s3mayur" {

Use the aws_s3_bucket_versioning resource instead

(and one more similar warning elsewhere)
```

3. Terraform apply



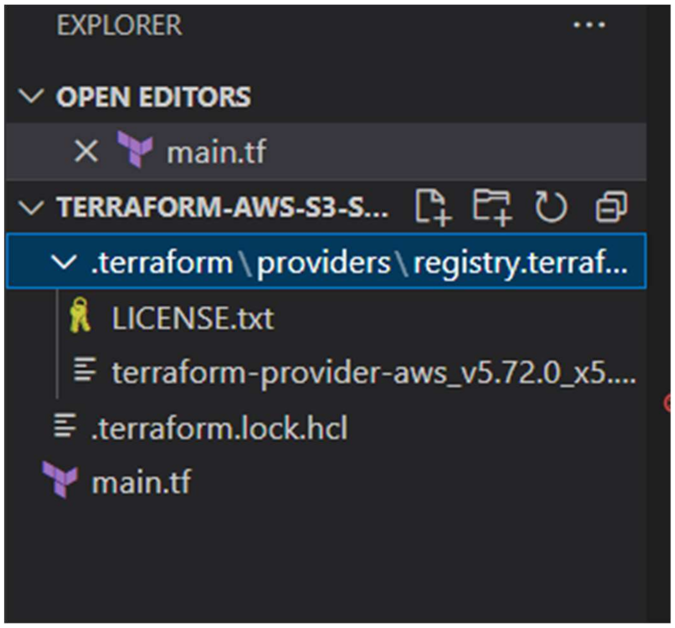
```
PS C:\Users\Hp\OneDrive\Documents\terraform-aws-s3-sqs-lambda> terraform apply

  Warning: Argument is deprecated

    with aws_s3_bucket.s3mayur,
    on main.tf line 6, in resource "aws_s3_bucket" "s3mayur":
     6: resource "aws_s3_bucket" "s3mayur" {

  Use the aws_s3_bucket_versioning resource instead

  (and one more similar warning elsewhere)
```

4. Terraform destroy



```
PS C:\Users\Hp\OneDrive\Documents\terraform-aws-s3-sqs-lambda> terraform destroy

  Warning: Argument is deprecated

    with aws_s3_bucket.s3mayur,
    on main.tf line 6, in resource "aws_s3_bucket" "s3mayur":
     6: resource "aws_s3_bucket" "s3mayur" {

  Use the aws_s3_bucket_versioning resource instead

  (and one more similar warning elsewhere)


Destroy complete! Resources: 0 destroyed.
```

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.