Lab Exercise 14 –Provisioning an S3 Bucket on AWS

Exercise Steps:

Step 1: Create a New Directory:

Create a new directory to store your Terraform configuration:

```
mkdir Terraform-S3-Demo
cd Terraform-S3-Demo
```

Step 2: Create the Terraform Configuration File (main.tf):

Create a file named main.tf with the following content:

```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "5.31.0"
    }
  }
}

provider "aws" {
  region     = "us-east-1"  # Replace with your preferred region
  access_key = "your IAM access key"  # Replace with your Access Key
  secret_key = "your secret access key"  # Replace with your Secret Key
}
```

This file sets up the Terraform AWS provider.

Step 3: Create a Terraform Configuration File for the S3 Bucket (s3.tf):

Create another file named s3.tf with the following content:

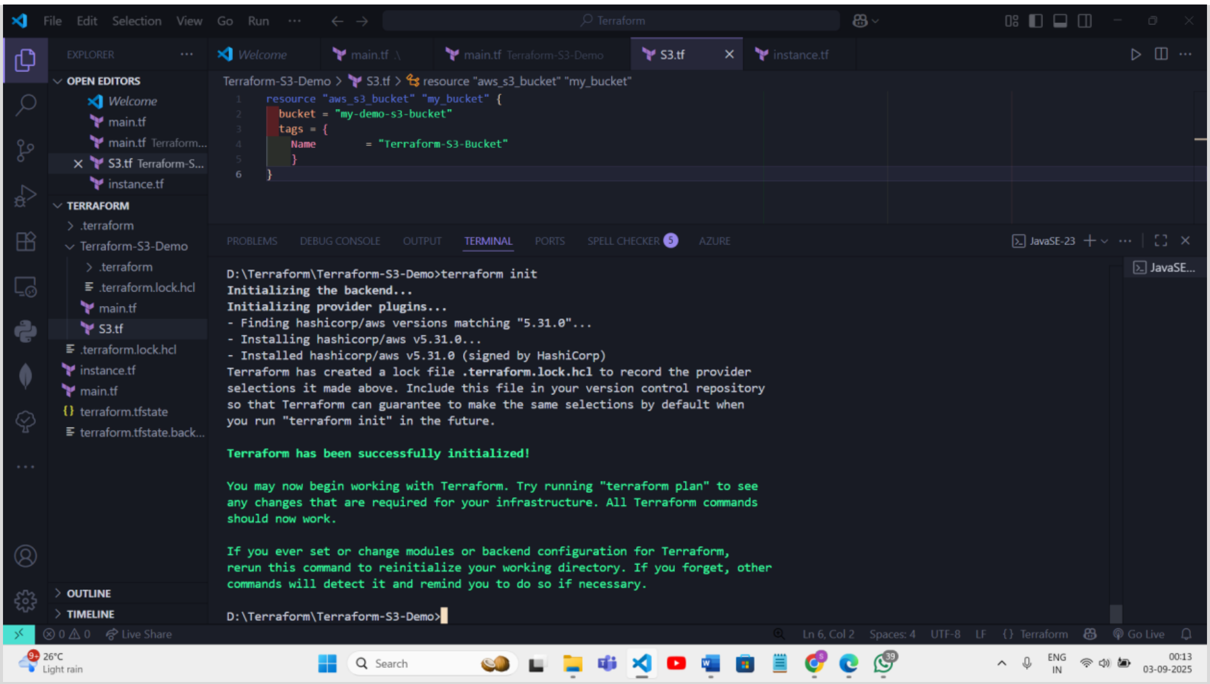
```
resource "aws_s3_bucket" "my_bucket" {
  bucket = "my-demo-s3-bucket"
  tags = {
    Name        = "Terraform-S3-Bucket"
    }
}
```

This file provisions an S3 bucket with a unique name using a random string suffix.

Step 4: Initialize Terraform:

Run the following command to initialize your Terraform working directory:

```
terraform init
```



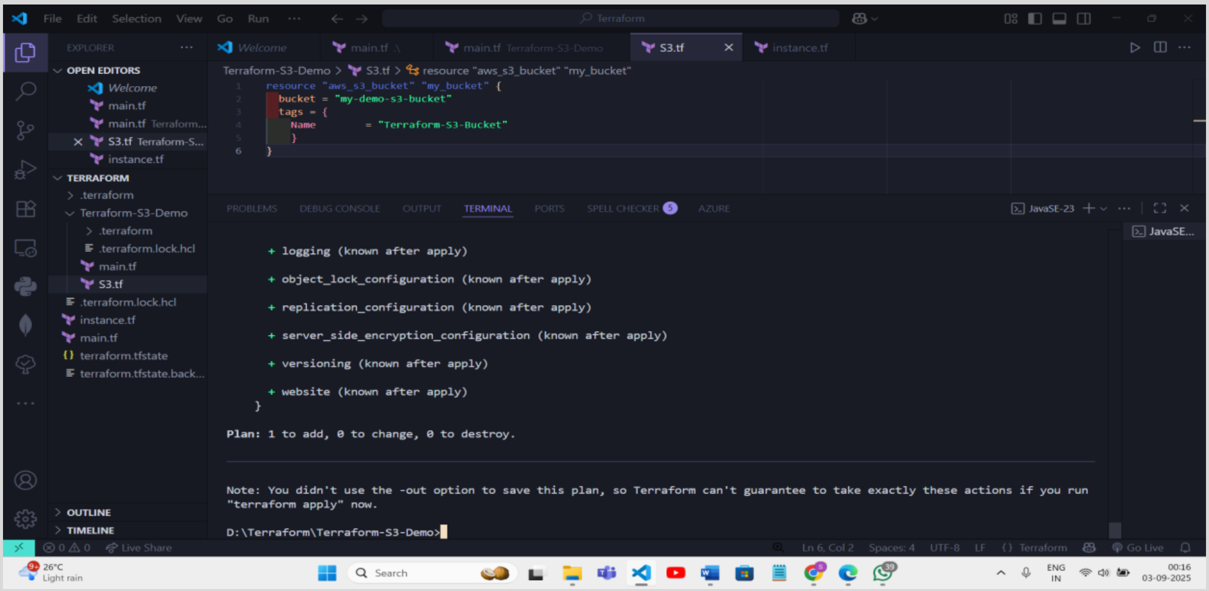
```
resource "aws_s3_bucket" "my_bucket" {
  bucket = "my-demo-s3-bucket"
  tags = {
    Name        = "Terraform-S3-Bucket"
  }
}
```

```
D:\Terraform\Terraform-S3-Demo>terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.31.0"...
- Installing hashicorp/aws v5.31.0...
- Installed hashicorp/aws v5.31.0 (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.

D:\Terraform\Terraform-S3-Demo>
```

Step 5: Review the Plan:

Preview the changes Terraform will make:

terraform plan



```
resource "aws_s3_bucket" "my_bucket" {
  bucket = "my-demo-s3-bucket"
  tags = {
    Name        = "Terraform-S3-Bucket"
  }
}
```

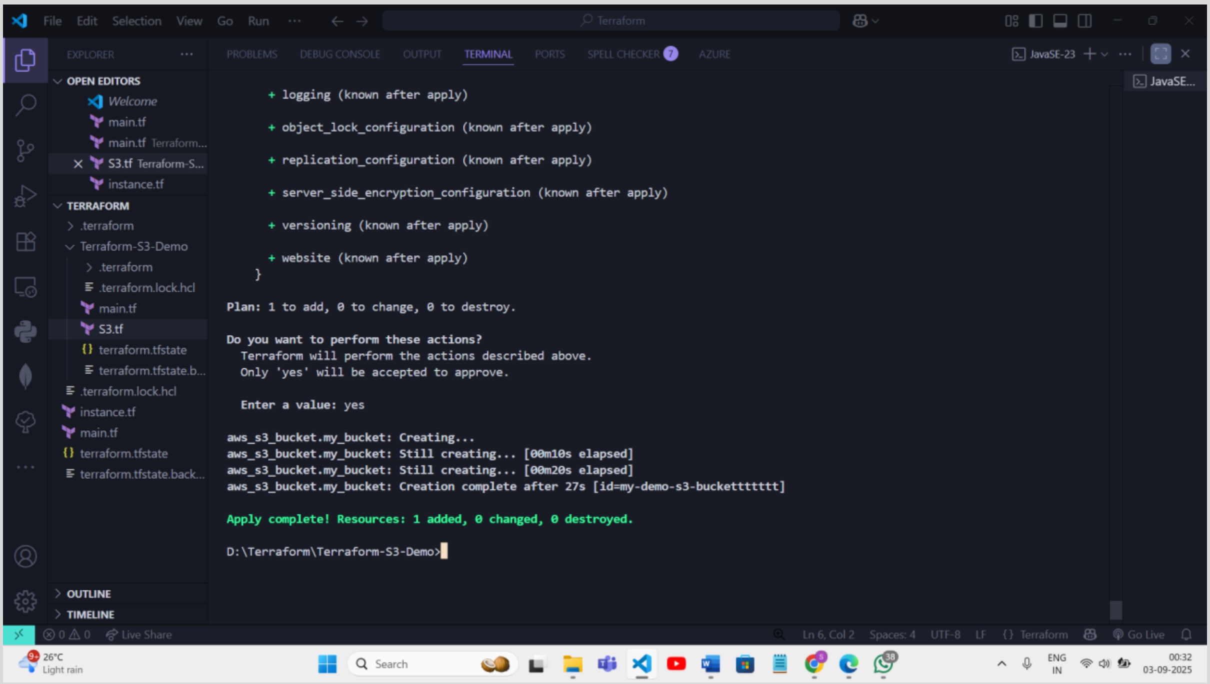
```
      + logging (known after apply)

      + object_lock_configuration (known after apply)

      + replication_configuration (known after apply)

      + server_side_encryption_configuration (known after apply)

      + versioning (known after apply)

      + website (known after apply)
    }

Plan: 1 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.

D:\Terraform\Terraform-S3-Demo>
```

Review the output to ensure it meets your expectations.

Step 6: Apply the Changes:

Create the resources:

```
terraform apply
```



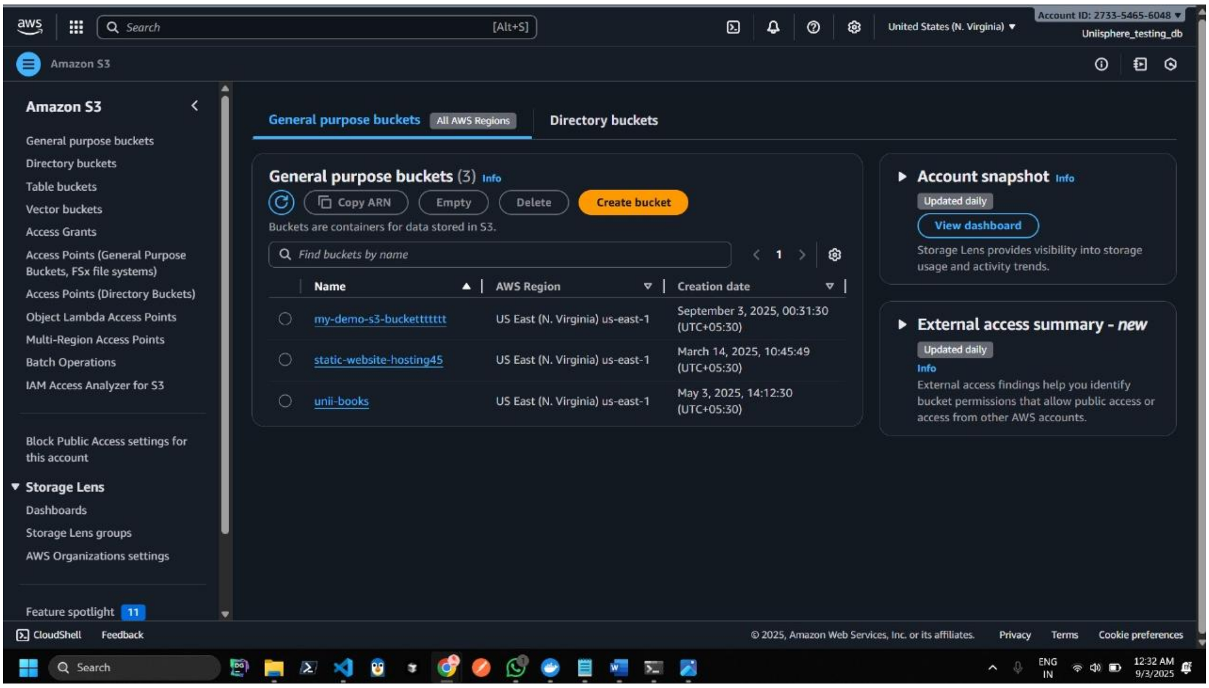
```
        + logging (known after apply)

        + object_lock_configuration (known after apply)

        + replication_configuration (known after apply)

        + server_side_encryption_configuration (known after apply)

        + versioning (known after apply)

        + website (known after apply)
    }

Plan: 1 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_s3_bucket.my_bucket: Creating...
aws_s3_bucket.my_bucket: Still creating... [00m10s elapsed]
aws_s3_bucket.my_bucket: Still creating... [00m20s elapsed]
aws_s3_bucket.my_bucket: Creation complete after 27s [id=my-demo-s3-buckettttttt]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

D:\Terraform\Terraform-S3-Demo>
```

When prompted, type yes to confirm.

Step 7: Verify Resources:

1. Log in to your AWS Management Console.
2. Navigate to the **S3** dashboard.

3. Verify that the S3 bucket has been created with the specified configuration.



Amazon S3
General purpose buckets
Directory buckets
Table buckets
Vector buckets
Access Grants
Access Points (General Purpose Buckets, FSx file systems)
Access Points (Directory Buckets)
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings
Feature spotlight 11
General purpose buckets All AWS Regions
Directory buckets
Copy ARN
Empty
Delete
Create bucket
Find buckets by name
View dashboard
CloudShell
Feedback

General purpose buckets (3) Info

Buckets are containers for data stored in S3.

Name	AWS Region	Creation date
my-demo-s3-buckettttttt	US East (N. Virginia) us-east-1	September 3, 2025, 00:31:30 (UTC+05:30)
static-website-hosting45	US East (N. Virginia) us-east-1	March 14, 2025, 10:45:49 (UTC+05:30)
unii-books	US East (N. Virginia) us-east-1	May 3, 2025, 14:12:30 (UTC+05:30)

Account snapshot Info

Updated daily

Storage Lens provides visibility into storage usage and activity trends.

External access summary - *new*

Updated daily

Info

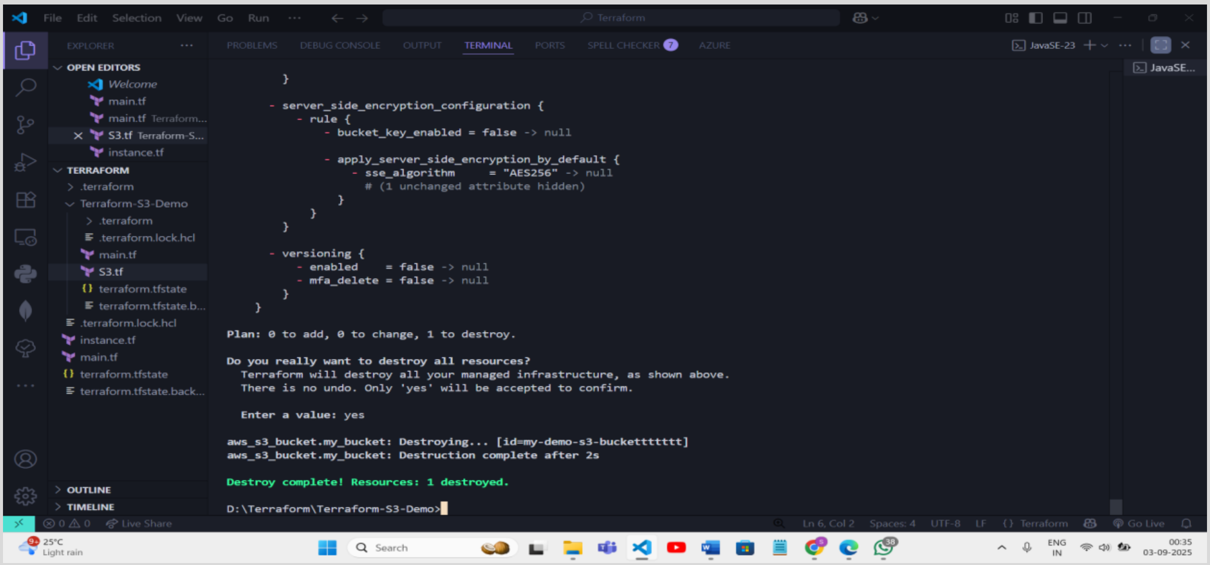
External access findings help you identify bucket permissions that allow public access or access from other AWS accounts.

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step 8: Cleanup Resources:

To remove the resources created, run the following command:

```
terraform destroy
```



```
          }

        - server_side_encryption_configuration {
          - rule {
              - bucket_key_enabled = false -> null

              - apply_server_side_encryption_by_default {
                  - sse_algorithm     = "AES256" -> null
                    # (1 unchanged attribute hidden)
                }
            }
        }

        - versioning {
          - enabled    = false -> null
          - mfa_delete = false -> null
        }
    }

Plan: 0 to add, 0 to change, 1 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_s3_bucket.my_bucket: Destroying... [id=my-demo-s3-buckettttttt]
aws_s3_bucket.my_bucket: Destruction complete after 2s

Destroy complete! Resources: 1 destroyed.

D:\Terraform\Terraform-S3-Demo>
```

When prompted, type yes to confirm.