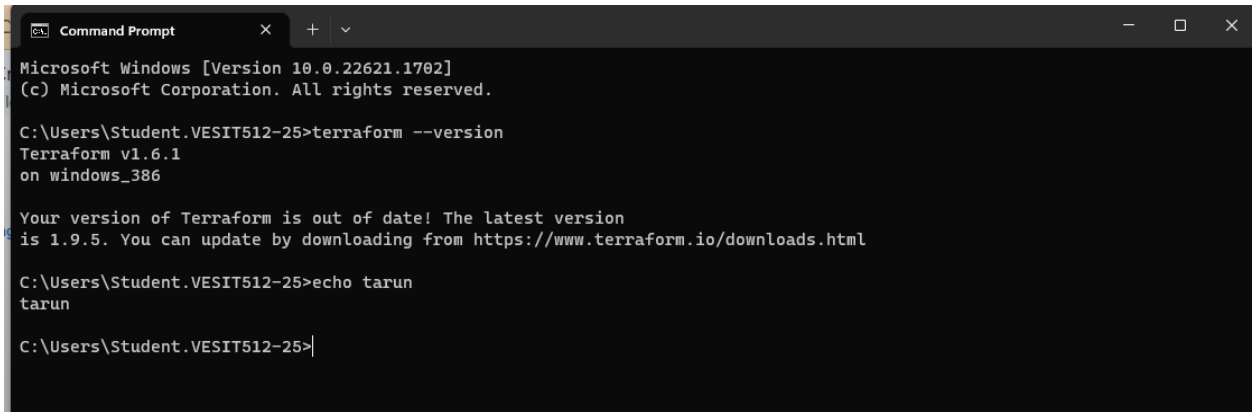
Experiment no:6

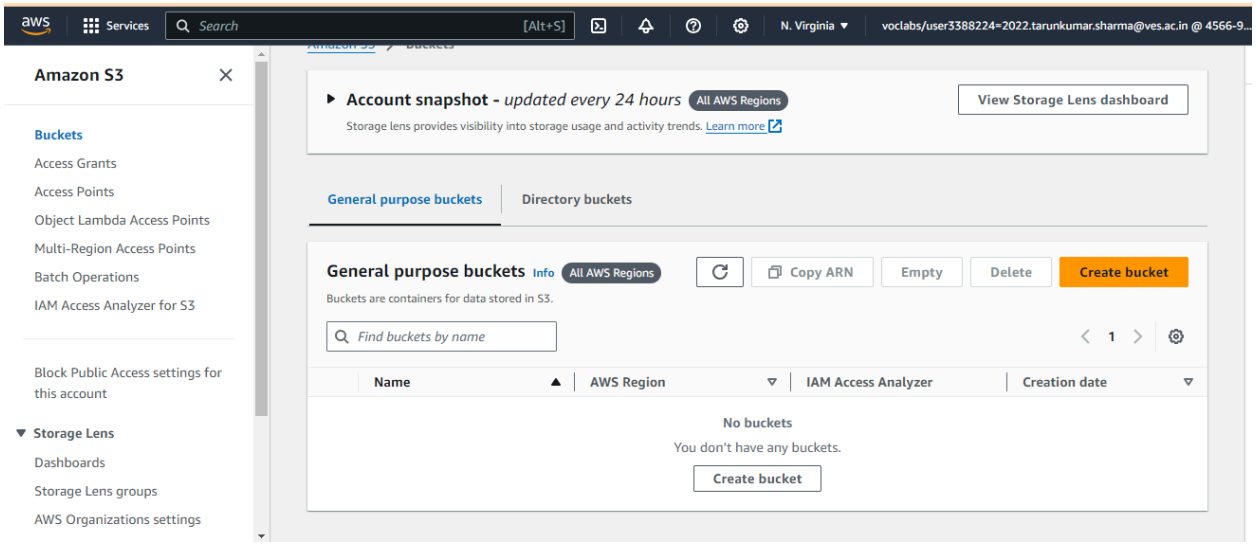
Prerequisite:

1) Install Atom Editor for Writing the Scripts from https://atom.io/

2) Must have an AWS Access Key ID and Secret Access Key

Step 1: Write a Terraform Script in Atom for creating S3 Bucket on Amazon AWS





3.Create a new provider.tf file and write the following contents into it.

s3.tf

File Edit View

```
resource "aws_s3_bucket" "Tarun" {
    bucket = "balti"

    tags = {
        Name        = "My Bucket"
        Environment = "Dev"
    }
}
```

```
provider "aws" {
access_key= "ASIAWUVJXREHDEFQUC2K"
secret_key= "S9+s4EBVDR6q8nC1jub8pt5mnwESPp8xnHdA0jaz"
token=
"IQoJb3JpZ2luX2VjECEaCXVzLXdlc3QtMiJIMEYCIQCsdQtJ3HJAQpMMlROMKyNzUEycmeefYnPy2mXuluuICgIhAIZUpPNWtF4M0tw3oe0U2Z430ZWqK3mvd3r
ZR9TggGUCKroCCFoQARoMNDU2Njk2MjM2MzAyIgyheDaNkkWDrxaqADEqlwIg9lucyiXzaR3RduShXnRKZIfpkpOnJyd/FA/BRCxMQBZacarsUYZKkSYq83dMQM8
T7tEQCm1JqoQ6Hh3LL+ulUBqroPKcn5hUxc+cYg+aL2VmQ2POz8XGBzm/h+TVYRcEk1UUoVLSbsFrx02J7lsCiV+Kqschq2MHHhBJnN/K/Y8q5tkNIcacRFF7w26
r31EHiFqMhDR0P3mO7W7v2C1dLBqQKjna1YkAnZPEnou2tQbfAL3PGy5+FuIlnVqmdQITa0S5Czz7IbQZpUznokJ5rhgesci/u1rPElHzIzjbaBMtClpjqIM7
+AuuZIlS1Oq2sjCGJ74XMFazciW+UhHXOFJjEwTnp5zZblyC8Zl/PUZgGjkkp/YwksSvtwY6nAGR/Rbd4Qpi8FkMm0naFMKVfybNcy4SBOFzb4NGmWs88ehjURZw
CN+eQWNy4rh6eWm8DAY0
+X70438XyCkLf9hTACX9zbMVTXsP7w0nOvRcVWvPX+aLpp0pDBmmQnaf1WdPPtNUEU5CvpyNutmG+Fmk4smZpIJI+QNRFwSyvFnRP7EcVvLSuyUQ+R/l+EPJtk7b
VrgqzMQKTbivDrY="

region= "us-east-1"

}
```

```
C:\Windows\System32\cmd.e

Microsoft Windows [Version 10.0.22621.1702]
(c) Microsoft Corporation. All rights reserved.

C:\tarun>dir
 Volume in drive C has no label.
 Volume Serial Number is F49B-BB37

 Directory of C:\tarun

19/09/2024  15:11    <DIR>          .
19/09/2024  15:11               923 provider.tf
19/09/2024  15:02               146 s3.tf
               2 File(s)          1,069 bytes
               1 Dir(s)  114,530,758,656 bytes free

C:\tarun>
```

Save both the files in same directory Terraform_Scripts/S3

Step 2: Open Command Prompt and go to Terraform_Script\S3 directory where our .tf files are stored

```
C:\tarun>set aws_access_key_id=ASIAWUVJXREHDEFQUC2K

C:\tarun>set aws_secret_access_key=S9+s4EBVDR6q8nC1jub8pt5mnwESPp8xnHdA0jaz

C:\tarun>set aws_session_token=IQoJb3JpZ2luX2VjECEaCXVzLXdlc3QtMiJIMEYCIQCsdQtJ3HJAQpMMlROMKyNzUEycmeefYnPy2mXuluuICgIhA
IZUpPNWtF4M0tw3oe0U2Z430ZWqK3mvd3rZR9TggGUCkroCCFoQARoMNDU2Njk2MjM2MzAyIgyheDaNkkWDrxaqADEqlwIg9lucyiXzaR3RduShXnRKZIfpk
pOnJyd/FA/BRCxMQBZacarsUYZkkSYq83dMQM8T7tEQCm1JqoQ6Hh3LL+ulUBqroPKcn5hUxc+cYg+aL2VmQ2POz8XGBzm/h+TVYRcEk1UUoVLSbsFrx02J7
lsCiV+Kqschq2MHHhBJnN/K/Y8q5tkNIcacRFF7w26r3lEHiFqMhDR0P3mO7W7v2C1dLBqQKjna1YkAnZPEnou2tQbfAL3PGy5+FuIlnVqmdQITa0S5Czz7I
bQZpUznokJ5rhgesci/u1rPElHzIzjbaBMtClpjqIM7+AuuZIlS1Oq2sjCGJ74XMFazciW+UhHXOFJjEwTnp5zZblyC8Zl/PUZgGjkkp/YwksSvtwY6nAGR/
Rbd4Qpi8FkMm0naFMKVfybNcy4SBOFzb4NGmWs88ehjURZwCN+eQWNy4rh6eWm8DAY0+X70438XyCkLf9hTACX9zbMVTXsP7w0nOvRcVWvPX+aLpp0pDBmmQ
naf1WdPPtNUEU5CvpyNutmG+Fmk4smZpIJI+QNRFwSyvFnRP7EcVvLSuyUQ+R/l+EPJtk7bVrgqzMQKTbivDrY=

C:\tarun>
```

Step 3: Execute Terraform Init command to initialize the resources

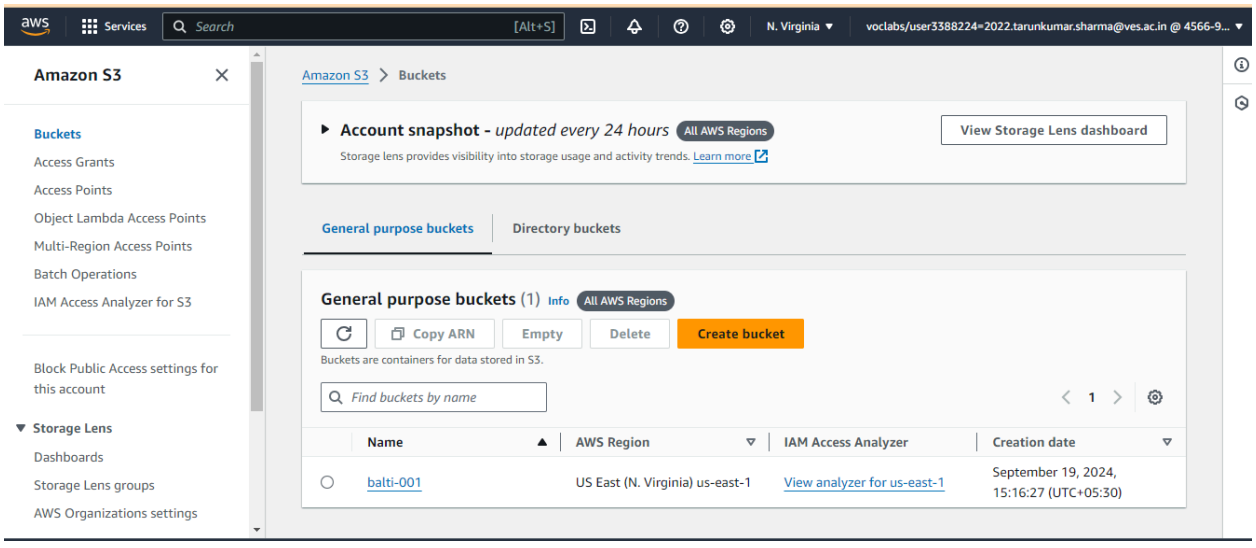
```
C:\tarun>terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.67.0...
- Installed hashicorp/aws v5.67.0 (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.
```

Step 4: Execute Terraform plan to see the available resources

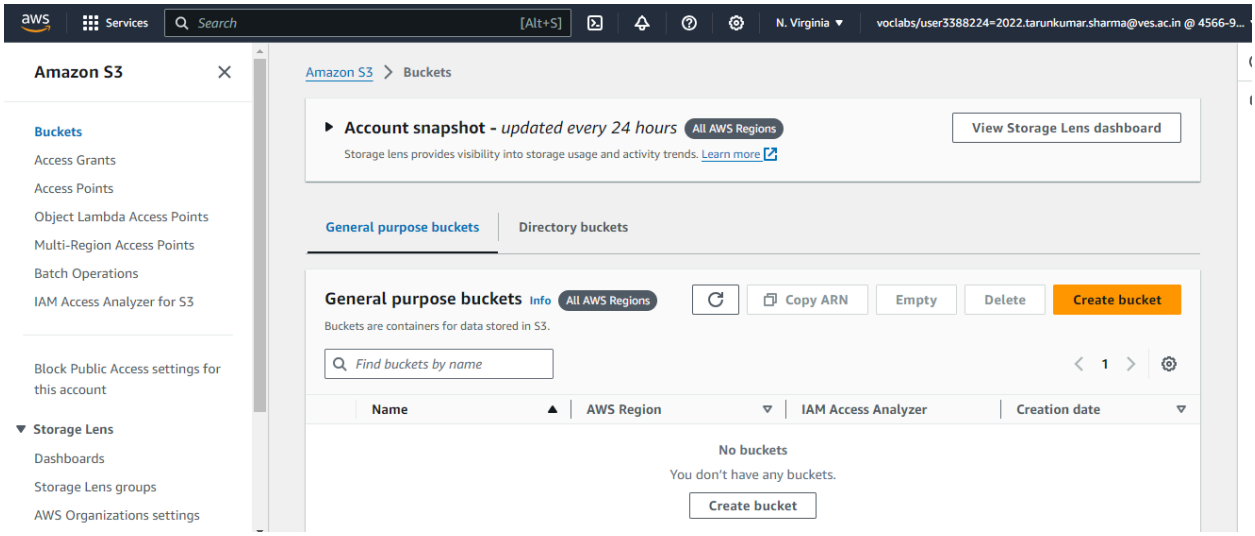
```
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
  + create

Terraform will perform the following actions:

  # aws_s3_bucket.Tarun will be created
  + resource "aws_s3_bucket" "Tarun" {
      + acceleration_status         = (known after apply)
      + acl                         = (known after apply)
      + arn                         = (known after apply)
      + bucket                      = "balti"
      + bucket_domain_name          = (known after apply)
      + bucket_prefix               = (known after apply)
      + bucket_regional_domain_name = (known after apply)
      + force_destroy               = false
      + hosted_zone_id              = (known after apply)
      + id                          = (known after apply)
      + object_lock_enabled         = (known after apply)
      + policy                      = (known after apply)
      + region                      = (known after apply)
      + request_payer               = (known after apply)
      + tags                        = {
          + "Environment" = "Dev"
          + "Name"        = "My Bucket"
        }
      + tags_all                    = {
          + "Environment" = "Dev"
          + "Name"        = "My Bucket"
        }
      + website_domain              = (known after apply)
      + website_endpoint            = (known after apply)
    }

Plan: 1 to add, 0 to change, 0 to destroy.
```

```
C:\tarun>terraform apply

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_s3_bucket.Tarun will be created
  + resource "aws_s3_bucket" "Tarun" {
      + acceleration_status         = (known after apply)
      + acl                         = (known after apply)
      + arn                         = (known after apply)
      + bucket                      = "balti"
      + bucket_domain_name          = (known after apply)
      + bucket_prefix               = (known after apply)
      + bucket_regional_domain_name = (known after apply)
      + force_destroy               = false
      + hosted_zone_id              = (known after apply)
      + id                          = (known after apply)
      + object_lock_enabled         = (known after apply)
      + policy                      = (known after apply)
      + region                      = (known after apply)
      + request_payer               = (known after apply)
      + tags                        = {
          + "Environment" = "Dev"
          + "Name"        = "My Bucket"
        }
      + tags_all                    = {
          + "Environment" = "Dev"
          + "Name"        = "My Bucket"
        }
      + website_domain              = (known after apply)
      + website_endpoint            = (known after apply)
```

```
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.tarun: Creating...
aws_s3_bucket.tarun: Creation complete after 5s [id=balti-001]

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



```
C:\tarun>terraform destroy
aws_s3_bucket.tarun: Refreshing state... [id=balti-001]

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

Terraform will perform the following actions:

  # aws_s3_bucket.tarun will be destroyed
  - resource "aws_s3_bucket" "tarun" {
      - arn                         = "arn:aws:s3:::balti-001" -> null
      - bucket                      = "balti-001" -> null
      - bucket_domain_name          = "balti-001.s3.amazonaws.com" -> null
      - bucket_regional_domain_name = "balti-001.s3.us-east-1.amazonaws.com" -> null
      - force_destroy               = false -> null
      - hosted_zone_id              = "Z3AQBSTGFYJSTF" -> null
      - id                          = "balti-001" -> null
      - object_lock_enabled         = false -> null
      - region                      = "us-east-1" -> null
      - request_payer               = "BucketOwner" -> null
      - tags                        = {
          - "Environment" = "Dev"
          - "Name"        = "My Bucket"
        } -> null
      - tags_all                    = {
          - "Environment" = "Dev"
          - "Name"        = "My Bucket"
        } -> null

      - grant {
          - id          = "2eebc183fba74a8482ad10907f869f29440b71300ef41029fc4dbc8a9f9a720e" -> null
          - permissions = [
              - "FULL_CONTROL",
            ] -> null
          - type        = "CanonicalUser" -> null
        }

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

              - apply_server_side_encryption_by_default {
                  - sse_algorithm = "AES256" -> null
                }
            }
        }

      - 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.tarun: Destroying... [id=balti-001]
aws_s3_bucket.tarun: Destruction complete after 1s

Destroy complete! Resources: 1 destroyed.

C:\tarun>
```


aws
Services
Search
[Alt+S]
N. Virginia
voclabs/user3388224=2022.tarunkumar.sharma@ves.ac.in @ 4566-9...
Amazon S3
Buckets
Access Grants
Access Points
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
Amazon S3 > Buckets
Account snapshot - updated every 24 hours
All AWS Regions
Storage lens provides visibility into storage usage and activity trends. Learn more
View Storage Lens dashboard
General purpose buckets
Directory buckets
General purpose buckets Info
All AWS Regions
Buckets are containers for data stored in S3.
Copy ARN
Empty
Delete
Create bucket
Find buckets by name
1
Name
AWS Region
IAM Access Analyzer
Creation date
No buckets
You don't have any buckets.
Create bucket