Name: Abhishek Chaurasiya
Class:D15B / 09
ADV DEVOPS PRAC 6 S3 BUCKET

AIM: Creating S3 Bucket using terraform

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

s3.tf provider.tf

s3.tf

```
resource ="aws_s3_bucket" "Abhi"{
  bucket = "Abhibucket12"
  acl = "public-read"

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

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

provider.tf

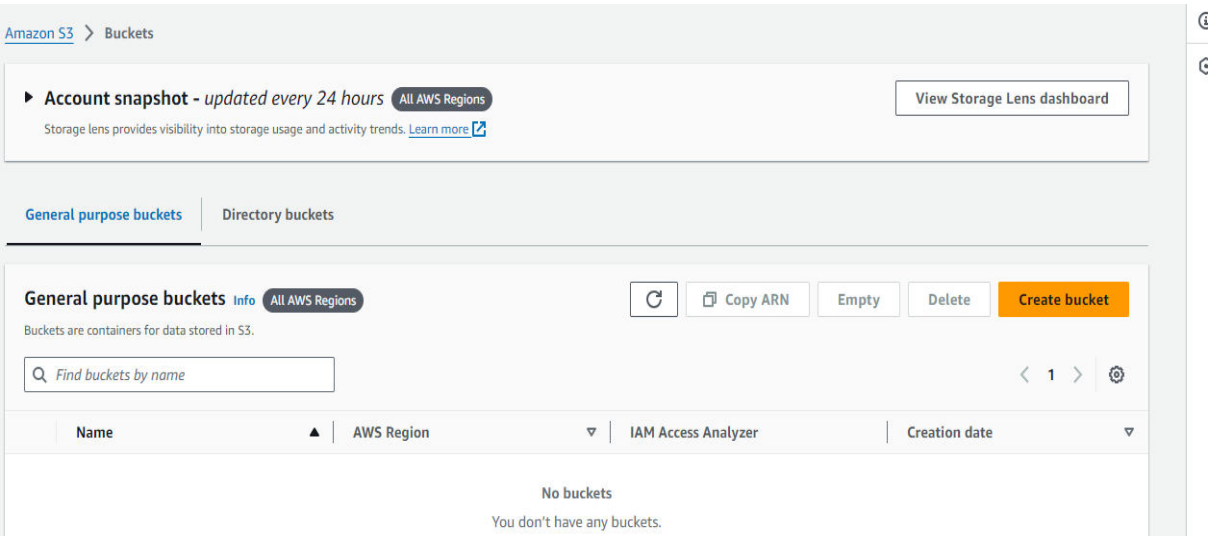
```
provider "aws" {
  access_key = "ASIAZS4OFYUJU2C3DE7L"
  secret_key ="0VE/jdclc9HmleCVTZ4s5aU2bxrGchVe3dir02mu"
  region ="us-east-1"
}
```

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:\Users\Student>terraform --version
Terraform v1.9.3
on windows_amd64
```

```
C:\Users\Student>cd Terraform_script
The system cannot find the path specified.

C:\Users\Student>cd C:\Terraform_script

C:\Terraform_script>cd s3

C:\Terraform_script\s3>dir
 Volume in drive C has no label.
 Volume Serial Number is 5C9A-38B0

 Directory of C:\Terraform_script\s3

08-08-2024  14:25    <DIR>          .
08-08-2024  14:25    <DIR>          ..
08-08-2024  14:24               147 provider.tf
08-08-2024  14:18               193 s3.tf
               2 File(s)            340 bytes
               2 Dir(s)  160,110,776,320 bytes free
```



Step 3: Execute Terraform Init command to initialize the resources

```
PS C:\Terraform_script\s3> terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.61.0

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

```
C:\Terraform_script\s3> terraform plan

raform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symb
+ create

raform will perform the following actions:

# aws_s3_bucket.Abhi will be created
+ resource "aws_s3_bucket" "Abhi" {
    + acceleration_status         = (known after apply)
```



```
# aws_s3_bucket.nidhi will be created
+ resource "aws_s3_bucket" "Abhi" {
    + acceleration_status         = (known after apply)
    + acl                         = (known after apply)
    + arn                         = (known after apply)
    + bucket                      = "Abhi09"
    + 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)
```

```
      + 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 "terra
now.

\Terraform_script\s3>
```


Start

Ln 6, Col 2 Spaces: 4 UTF-8 CRLF Plain Tex

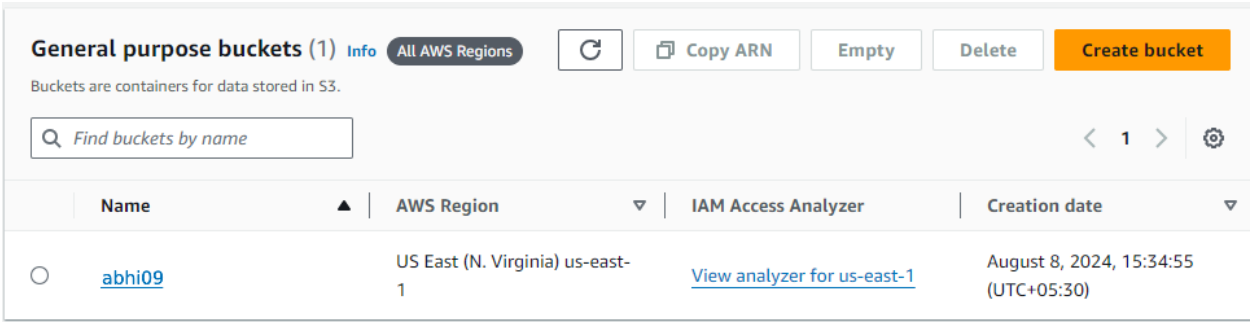
Step 5: Execute Terraform apply to apply the configuration, which will automatically create an

S3 bucket based on our configuration.

```
# aws_s3_bucket.nidhi will be created
+ resource "aws_s3_bucket" "Abhi" {
    + acceleration_status         = (known after apply)
    + acl                         = (known after apply)
    + arn                         = (known after apply)
    + bucket                      = "Abhi09"
    + 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)
```

```
      }
    + website_domain              = (known after apply)
    + website_endpoint            = (known after apply)

    + cors_rule (known after apply)

    + grant (known after apply)

    + lifecycle_rule (known after 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:
```

```
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.abhi : Creating...
aws_s3_bucket.abhi : Creation complete after 8s [id=abhi09]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Terraform_script\s3>
```



Step 6: Execute Terraform destroy to delete the configuration, which will automatically delete an EC2 instance

```
          - 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.abhi : Destroying... [id=abhi09]
aws_s3_bucket.abhi : Destruction complete after 1s

Destroy complete! Resources: 1 destroyed.
PS C:\Terraform script\s3>
```