Creating S3 Bucket using terraform

Rushabh Jain
D15B 25

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

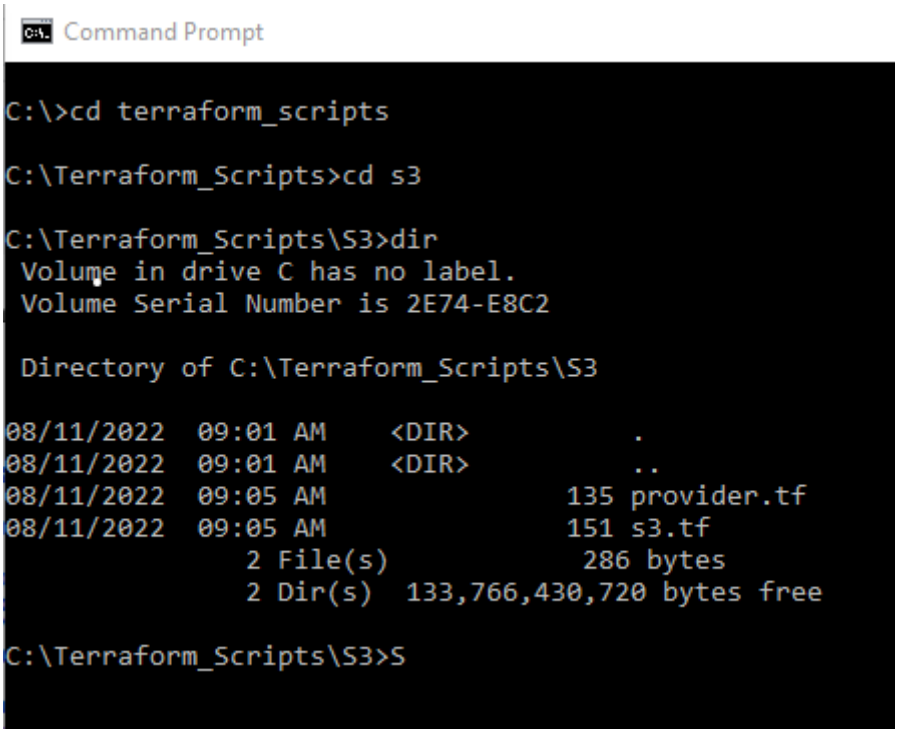
```
resource "aws_s3_bucket" "rushabh" {
    bucket = "rush"

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

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

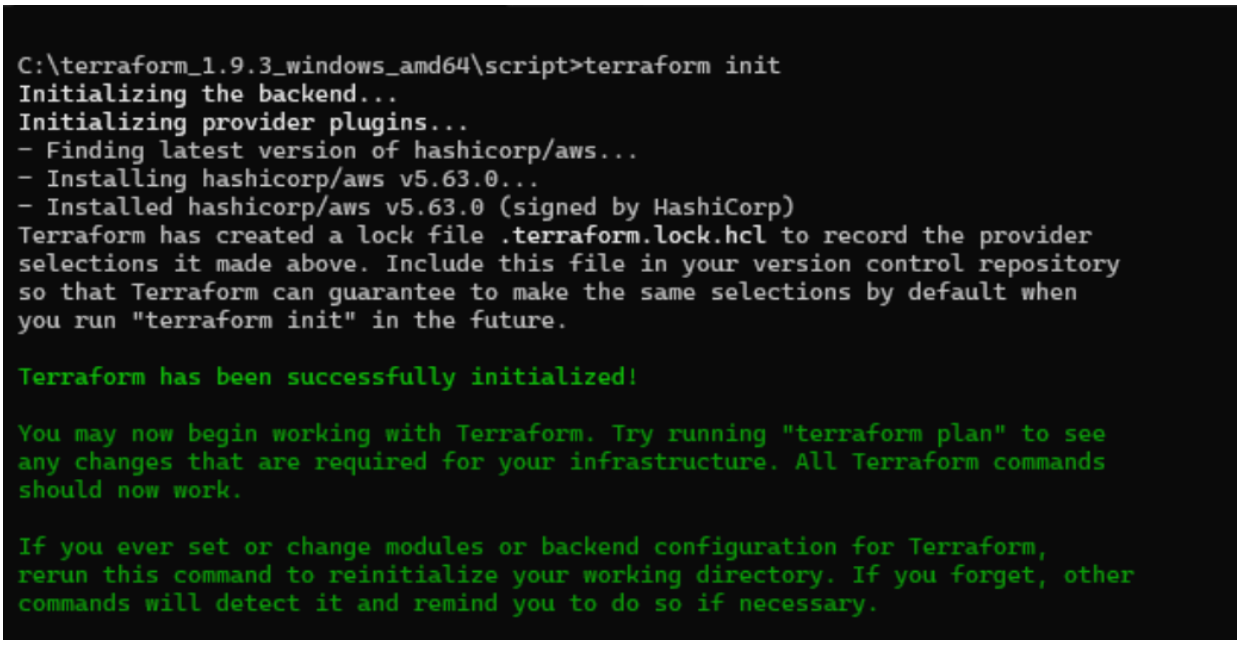
```
provider "aws" {
access_key= "AKIAYIY3PKM7VRJB3TFP"
secret_key="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"
region = "ap-south-1"
}
```

Save both the files in same directory Terraform_Scripts/S3

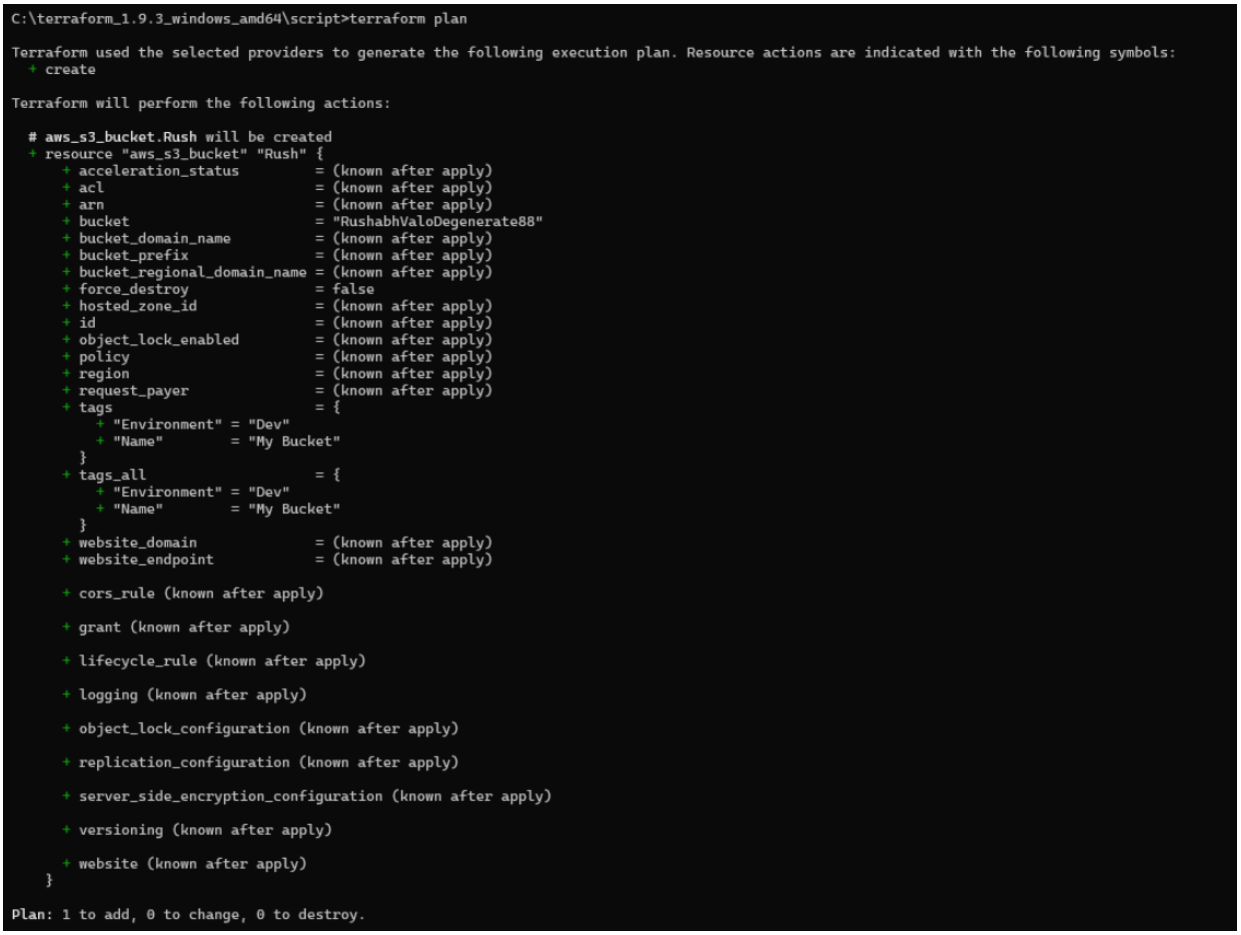
Step 2: Open Command Prompt and go to Terraform_Script\S3 directory where our .tf files are stored



Step 3: Execute Terraform Init command to initialize the resources



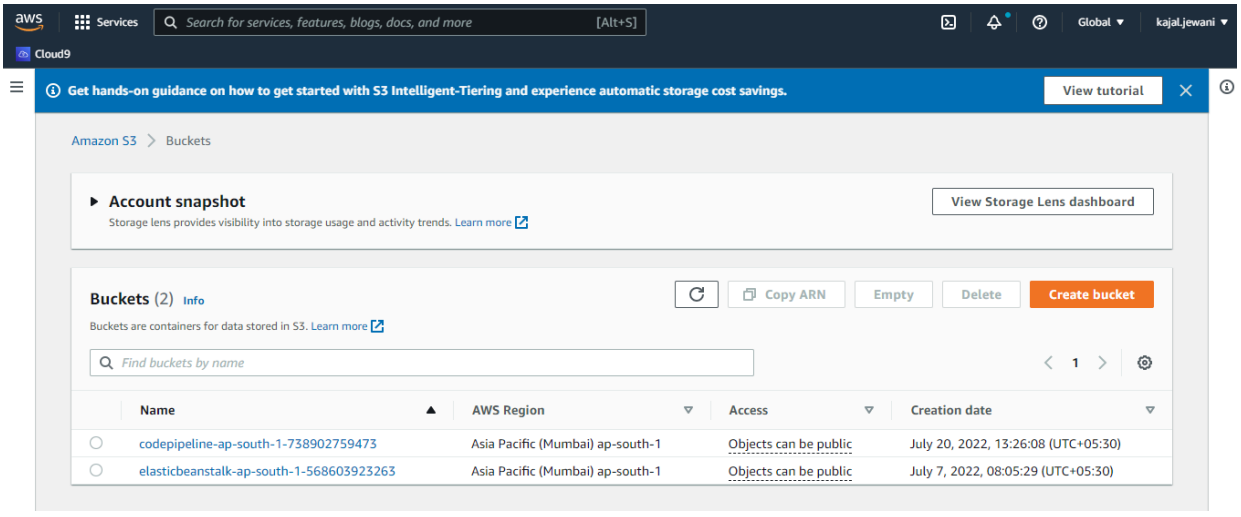
Step 4: Execute Terraform plan to see the available resources



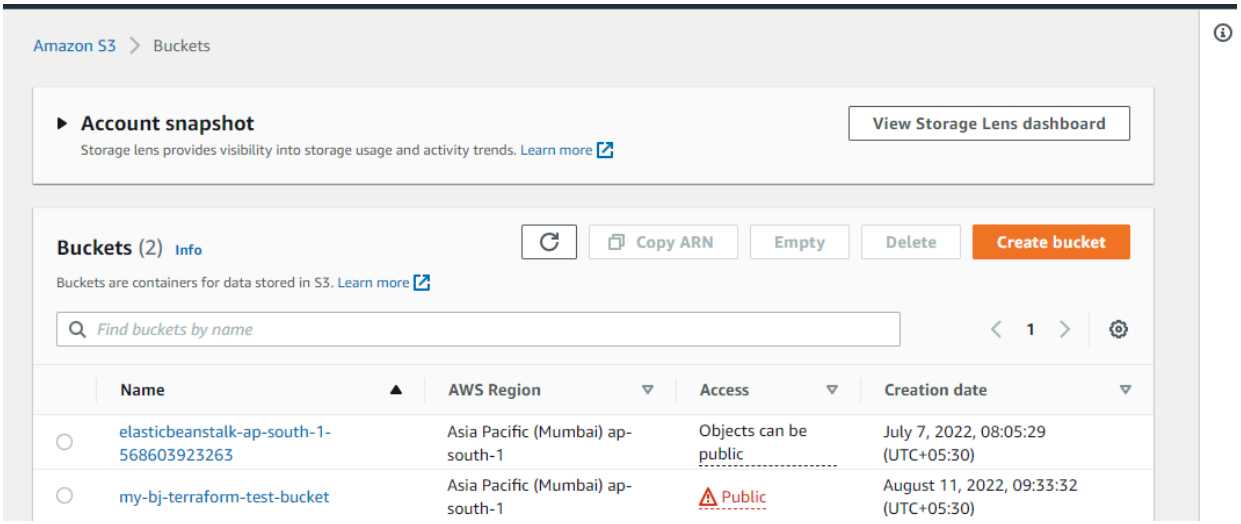
Step 5: Execute Terraform apply to apply the configuration, which will automatically create an S3 bucket based on our configuration.

```
C:\terraform_1.9.3_windows_amd64\script>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.Rush will be created
  + resource "aws_s3_bucket" "Rush" {
      + acceleration_status         = (known after apply)
      + acl                         = (known after apply)
      + arn                         = (known after apply)
      + bucket                      = "rushabhvalodegenerate"
      + 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)

      + 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: yes

aws_s3_bucket.Rush: Creating...
aws_s3_bucket.Rush: Creation complete after 7s [id=rushabhvalodegenerate]

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

AWS S3bucket dashboard, Before Executing Apply command:



AWS S3 Bucket dashboard, After Executing Apply step:



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

```
C:\terraform_1.9.3_windows_amd64\script>terraform destroy
aws_s3_bucket.Rush: Refreshing state... [id=rushabhvalodegenerate]

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.Rush will be destroyed
  - resource "aws_s3_bucket" "Rush" {
      - arn                         = "arn:aws:s3:::rushabhvalodegenerate" -> null
      - bucket                      = "rushabhvalodegenerate" -> null
      - bucket_domain_name          = "rushabhvalodegenerate.s3.amazonaws.com" -> null
      - bucket_regional_domain_name = "rushabhvalodegenerate.s3.us-east-1.amazonaws.com" -> null
      - force_destroy               = false -> null
      - hosted_zone_id              = "Z3AQBSTGFYJSTF" -> null
      - id                          = "rushabhvalodegenerate" -> 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
        # (3 unchanged attributes hidden)

      - grant {
          - id          = "5cf69121f6024296a31f10380ffb5fedeb079c92a2bbe87098a3e6d4389f5ef1" -> null
          - permissions = [
              - "FULL_CONTROL",
            ] -> null
          - type        = "CanonicalUser" -> null
            # (1 unchanged attribute hidden)
        }

      - 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.Rush: Destroying... [id=rushabhvalodegenerate]
aws_s3_bucket.Rush: Destruction complete after 1s

Destroy complete! Resources: 1 destroyed.
```

AWS EC2 dashboard, After Executing Destroy step: