**Name: Roma Shirodkar**

**Div: D15B**

**Roll No: 58**

**Experiment 5**

**Aim:** To Build, change, and destroy AWS / GCP /Microsoft Azure/ DigitalOcean infrastructure Using Terraform. (S3 Bucket)

**Theory:**

Terraform is an open-source Infrastructure as Code (IaC) tool developed by HashiCorp. It enables you to define and provision data center infrastructure using a declarative configuration language known as HashiCorp Configuration Language (HCL) or JSON. Terraform allows you to manage cloud resources in a consistent and reproducible manner, automating the creation, modification, and deletion of infrastructure components.

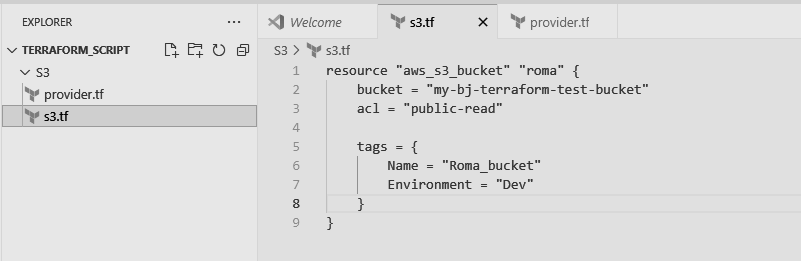
*Build Infrastructure:* Terraform allows you to define your desired infrastructure in code. You can create resources like EC2 instances, S3 buckets, RDS databases, and more in AWS using simple configuration files.

*Destroy Infrastructure:* If you need to remove resources, Terraform can easily delete them by managing the lifecycle. This is particularly useful for cleaning up after testing or decommissioning environments.

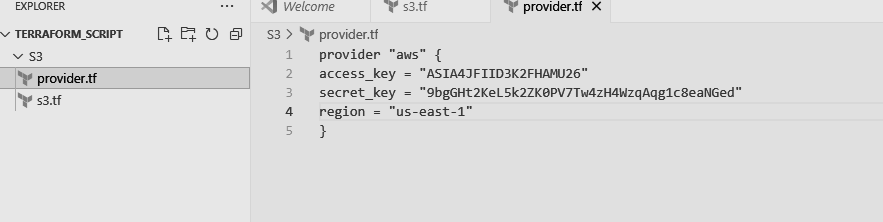
***Change Infrastructure:*** When you need to make changes, you simply modify the configuration files. Terraform calculates the difference between the desired state and the current state, allowing you to preview the changes before applying them.

Steps-

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

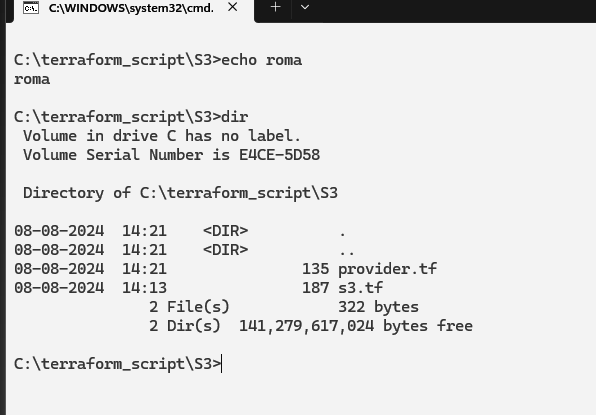


Create a new provider.tf file and write the following contents into it.

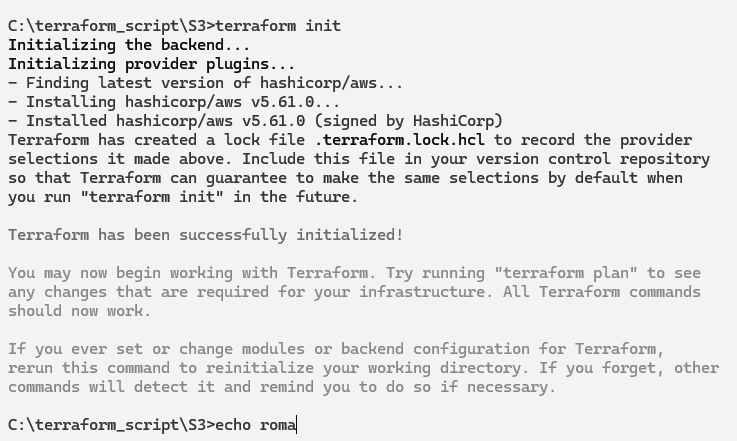


Save both the files in same directory Terraform\_Scripts/S3

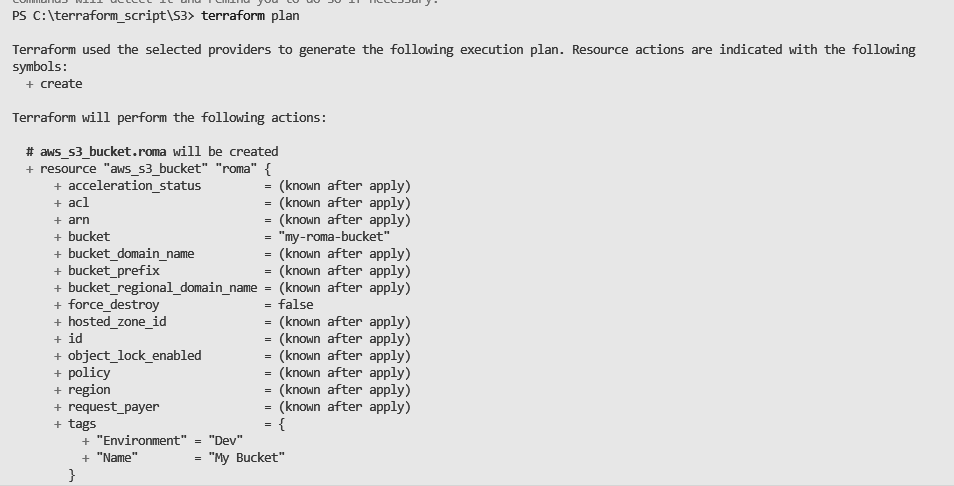
Step 2: Open Command Prompt and go to Terraform\_Script\S3 directory where our .tf files are stored

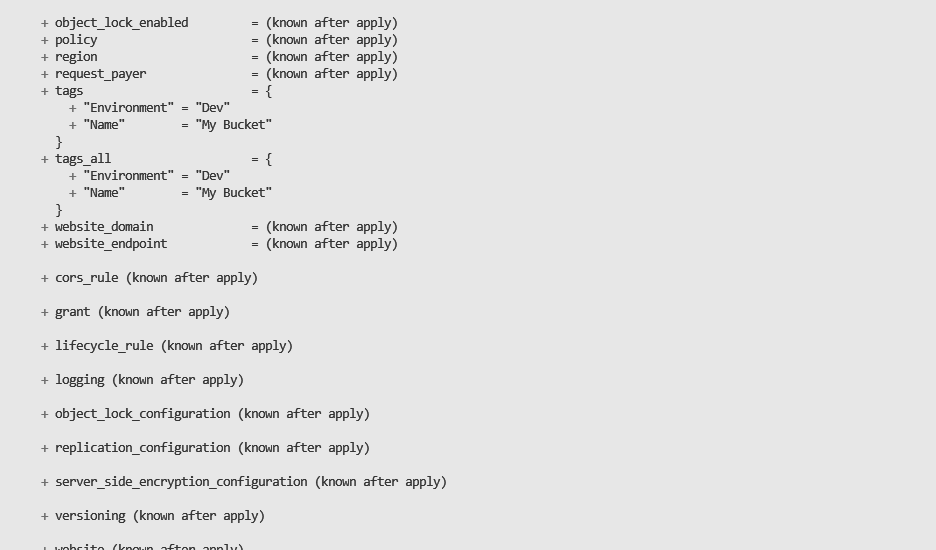


Step 3: Execute Terraform Init command to initialize the resources

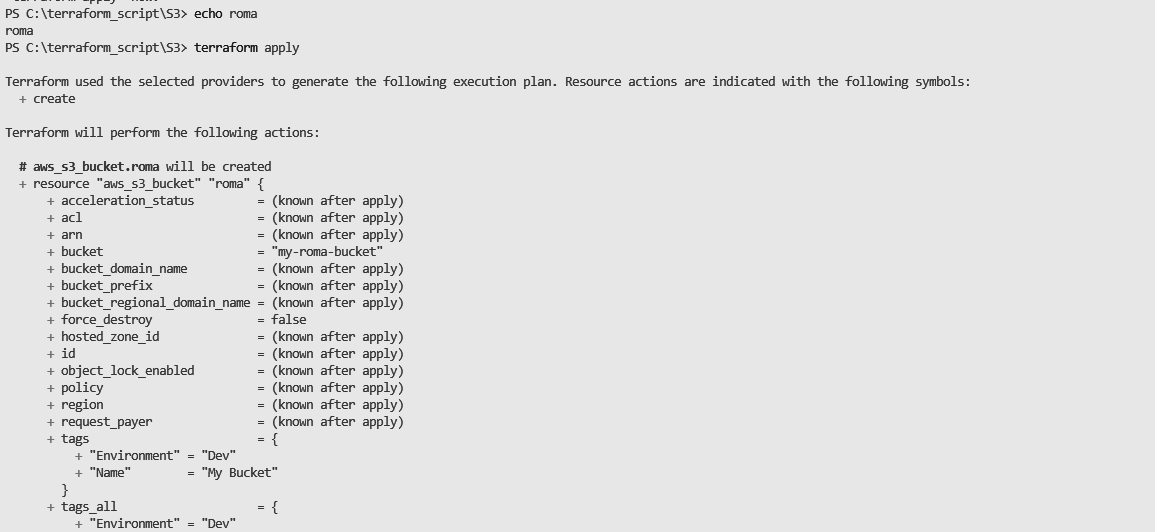


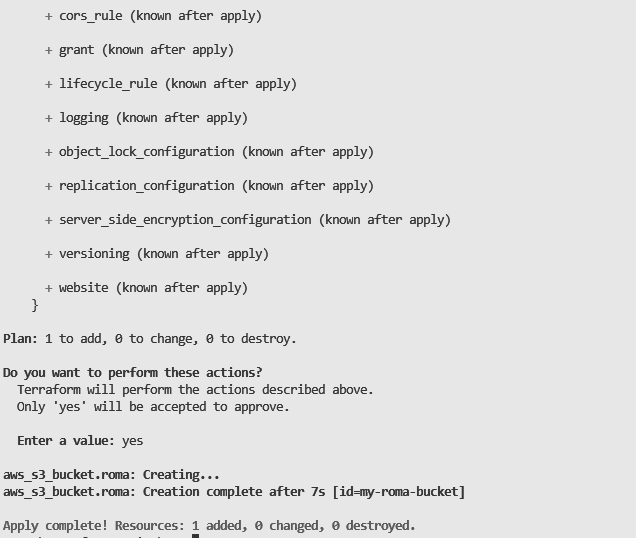
Step 4: Execute Terraform plan to see the available resources



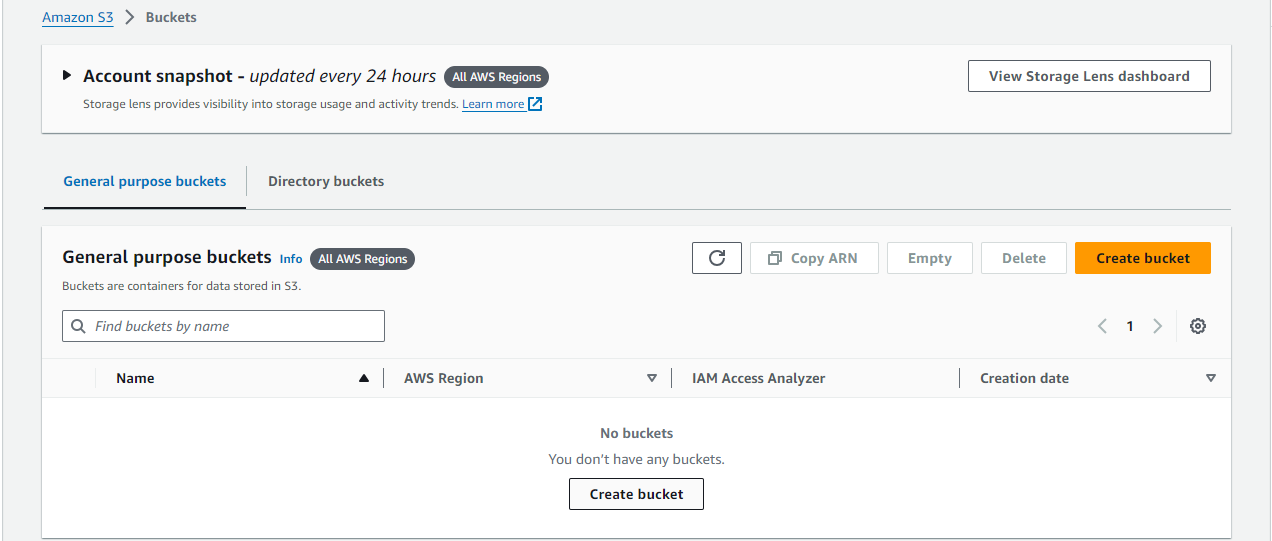


Step 5: Execute Terraform apply to apply the configuration, which will automatically create an S3 bucket based on our configuration.

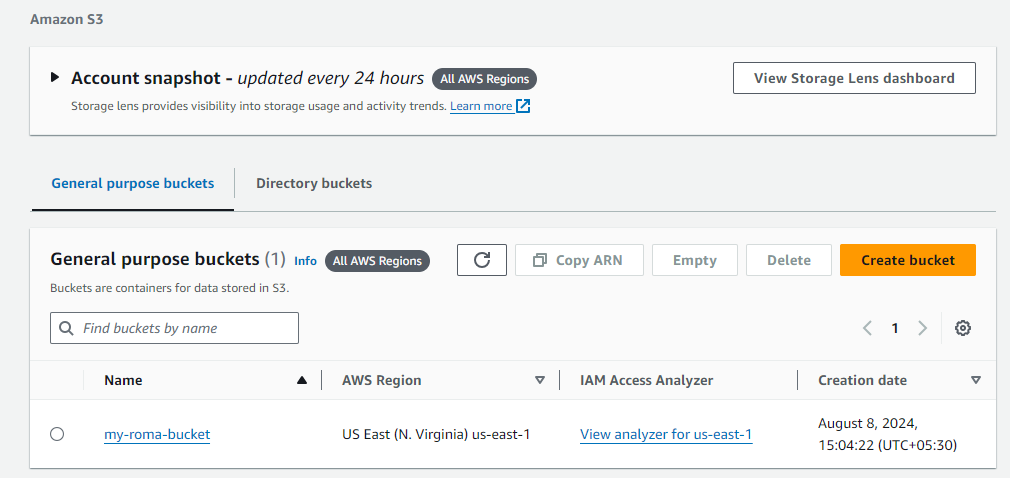




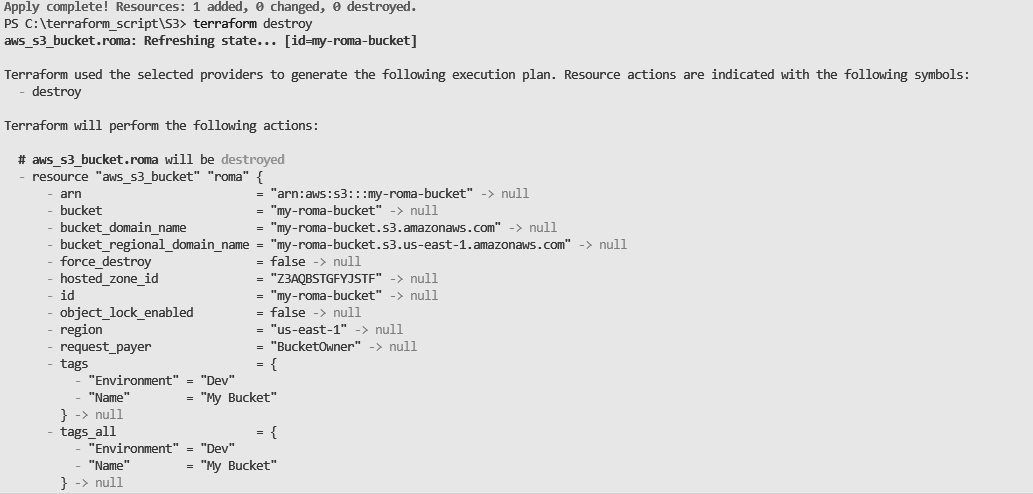
AWS S3bucket dashboard, Before Executing Apply command:



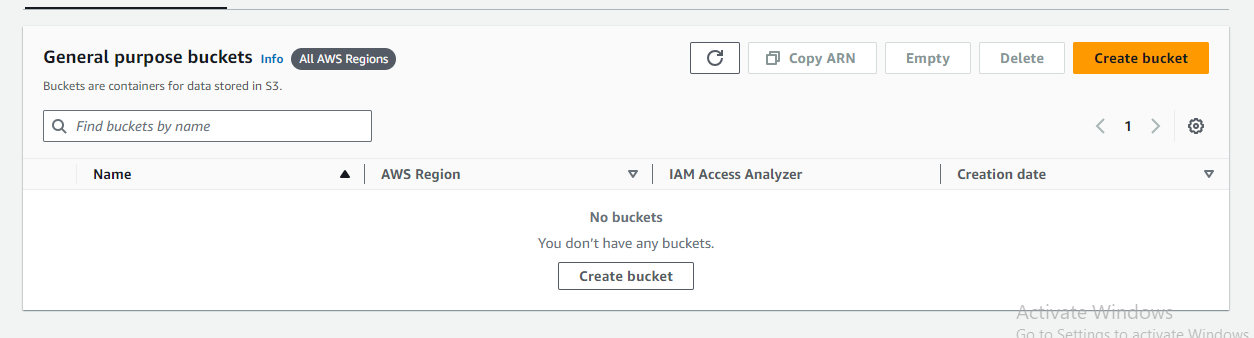
AWS S3 Bucket dashboard, After Executing Apply step:



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



AWS EC2 dashboard, After Executing Destroy step:



Conclusion: Thus, in this experiment, we learned how to effectively use Terraform to build, change, and destroy infrastructure across multiple cloud providers such as AWS, GCP, Microsoft Azure, and DigitalOcean. We explored the importance of Infrastructure as Code (IaC) in automating cloud resource management, enhancing collaboration, and reducing human error.