**HOW TO CREATE A S3 BUCKET USING TERRAFORM SCRIPT FROM CLI?**

In this blog post, we are going to explain how to create S3 Bucket using Terraform Script from CLI. This is one of the first blog posts on the Terraform blog series we have planned to publish in this blog. We are providing consulting services for Terraform and also helping companies to write **Terraform scripts** to provision infrastructure as they need.

If you have any questions related to this article, please write it in the comments section or contact us for more information. We would be happy to help you in learning more about Terraform.

**Note:**The objective of this Terraform script is to teach you the basics of how to write and execute Terraform script and an introduction to Infrastructure as Code (IaC).

**Create S3 Bucket Using Terraform CLI Commands**

**Prerequisites**

Here is the prerequisite to execute the scripts given in this blog post.

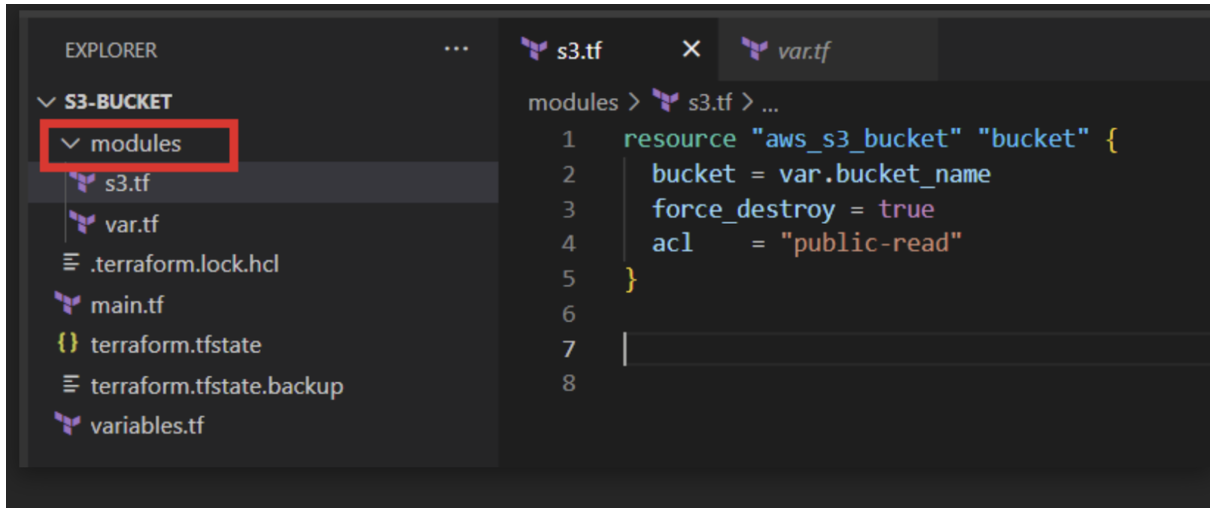
1. Access key ID & Secret Access Key of your AWS IAM User’s.
2. AWS IAM User must have an AWS IAM role with S3 permissions
3. This example is written using the Visual Studio Code

**Steps To Create S3 Bucket Using Terraform**

Here is the list of steps that would help you to write and execute the simple Terraform script that will create a S3 bucket in the backend.

**Step 1: Create a module folder in the local**

If you look at the below screenshot, this is the project that is used for writing the terraform scripts in our local project. The first step is to create a “modules” folder where we store all the terraform scripts that are used for creating S3 buckets. This folder will be referred from outside files.



If you look at the above screenshot, we have created two files under the “modules” folder:

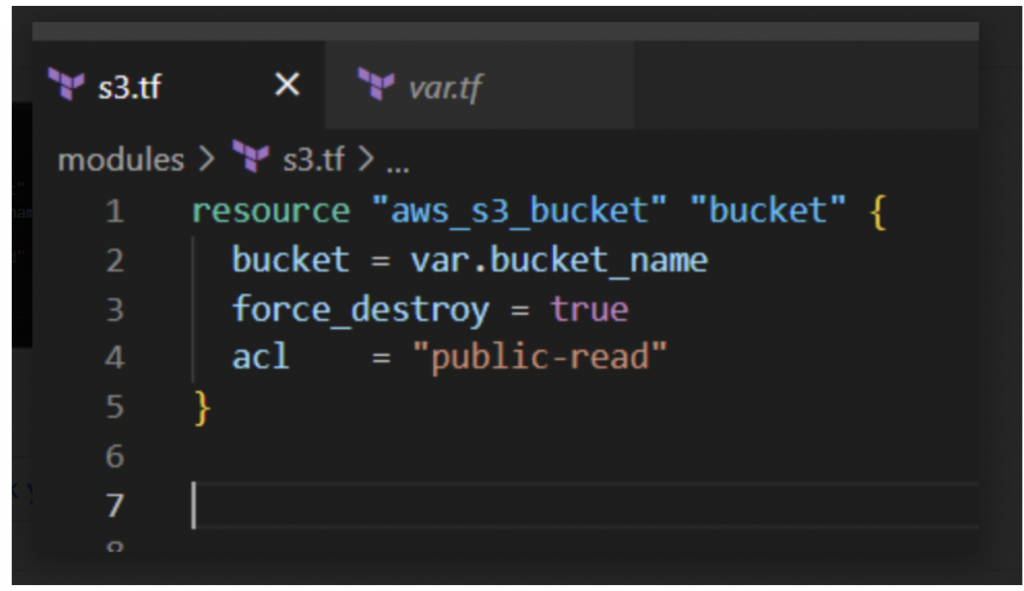
1. s3.tf
2. var.tf

The next step is to write the configuration parameter inside the scripts. Let’s move on to the next step.

**Step 2: Add Bucket Configurations**

As we have explained, this example is to create an S3 bucket. We have to write the required configurations to create S3 buckets in the AWS infrastructure using the code.

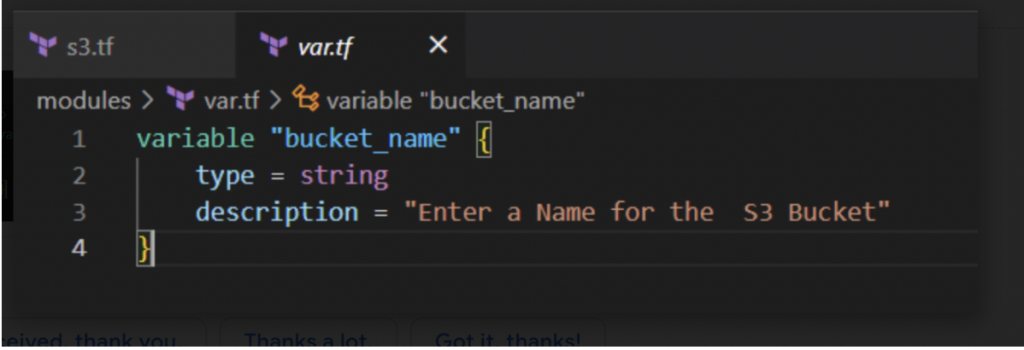
Here is the script written for creating the S3 bucket:



* In the above script, it contains a block named “resource” which has the resource type “*aws\_s3\_bucket*“. This is the fixed resource value for S3 defined in the AWS documentation. “bucket” is a resource name that the user provides.
* bucket and acl are the argument types, which we have provided the details of our resource. Either we can provide value directly or use the *variables.tf* file to declare the value of an argument.

**Step 3: Defining Variables**

The next step would be to declare variables that are used in the project. We have added all the variables in the file *var.tf.*

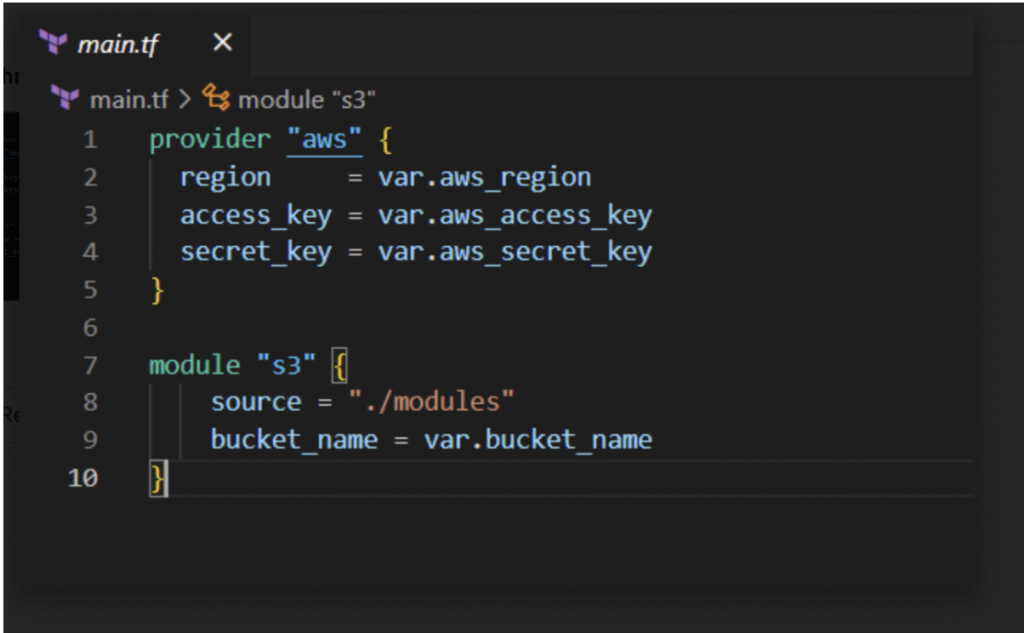


For our bucket we provide an argument “bucket” as *var.bucket\_name*, Which is declared in *var.tf*

**Step 4: Create main.tf and variables.tf in root folder**

The final step would be to create the two files outside the “*modules*” folder which are used for invoking the scripts and declaring the global variables.

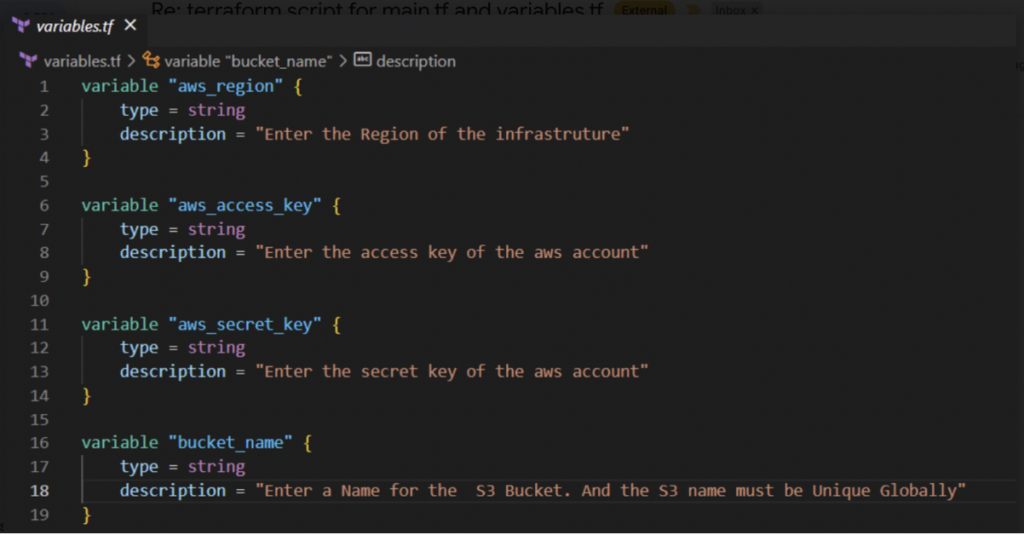
Here is the snapshot of the *main.tf*code.



1. In the “provider” section we provide our aws “access\_key” and “secret\_key” and the region where we need to create our bucket.
2. Since we are creating S3 using terraform modules, we need to add an S3 module to create an S3 bucket.
3. For this, we will use the keyword “module” and the name of the modules (folder) which we have created earlier.
4. In argument “source”, we will provide a source to the S3 module.
5. For “bucket name”, from the *modules/var.tf* we defined var.bucket name in *variables.tf*.

**Step 5: Add all the variables**

 This is the variables, where we enter the details of these variables when we run the script.



* From the above code we give the values for Access key, Secret key, Region and the Bucket name.

So, We done our code. Now we can execute our code from **CLI**.

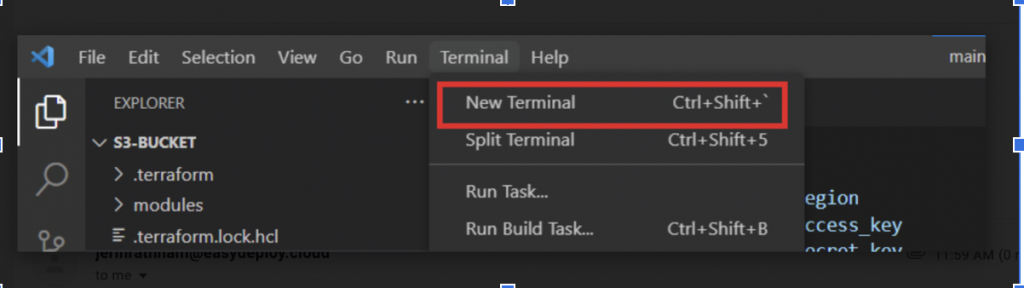
**RUN TERRAFORM CODE FROM OUR SYSTEM**

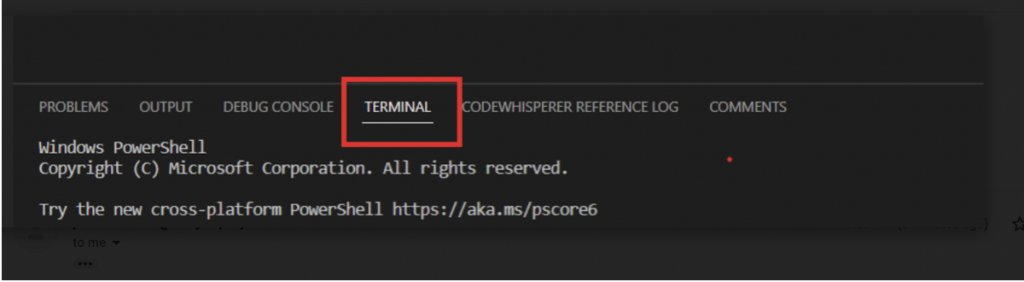
In the above sections, we have explained the code and why it is required. We have completed writing the code that is required to create the S3 bucket, now it’s time to execute the code using the following commands.

We are using Visual Source Code to run this example project. Here is the steps to execute this script:

1. Open Visual Source Code
2. Open menu “*Terminal*”
3. Click on “*New Terminal*” window (This is the place where we will run all the commands to execute the script)

Here are the snapshots of the terminal window for your reference. You can follow the same steps to execute the terraform scripts.

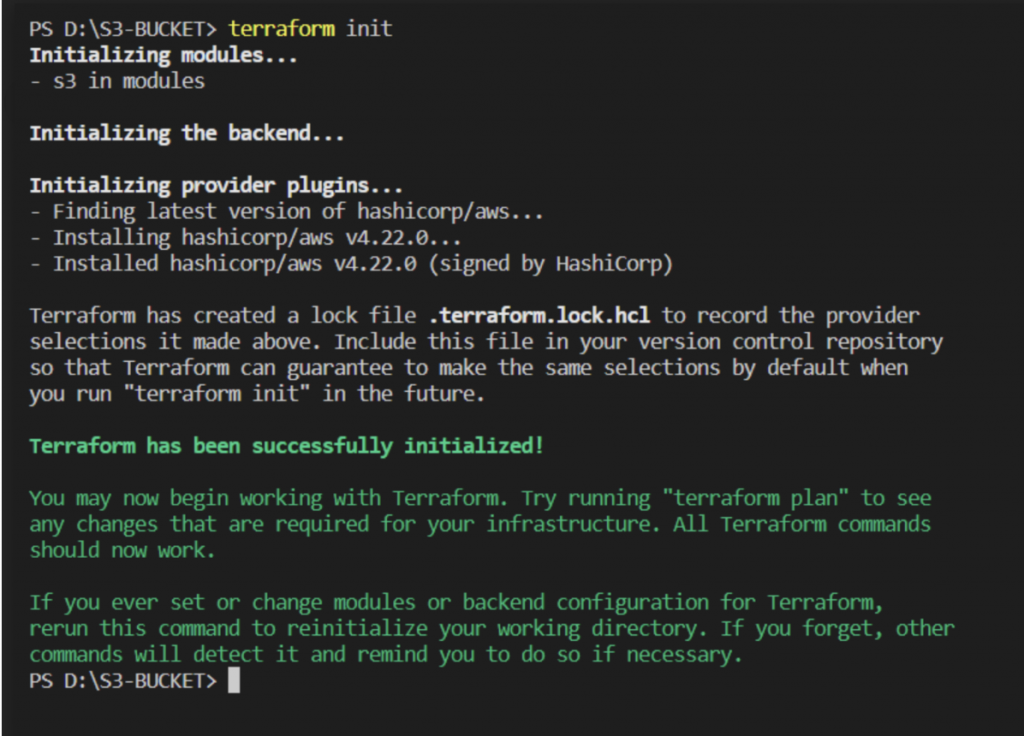


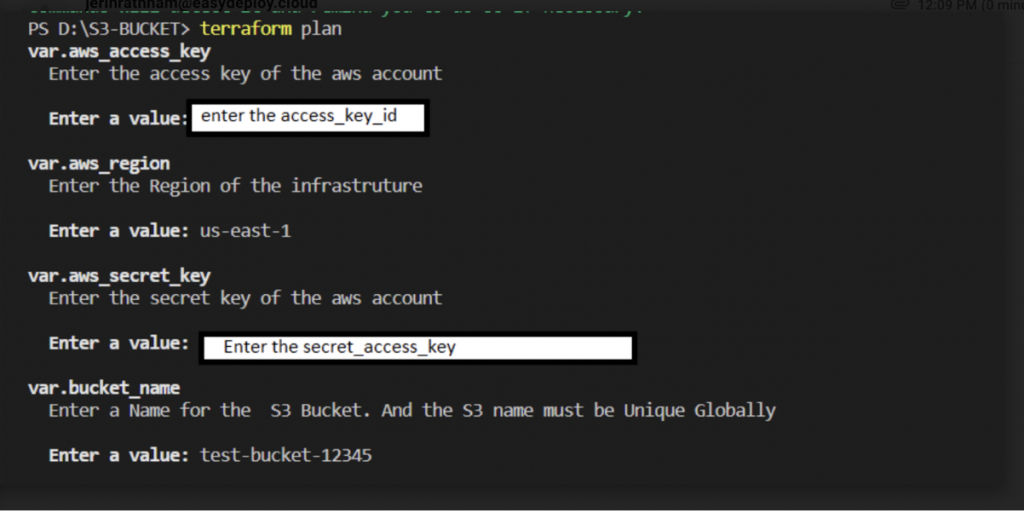


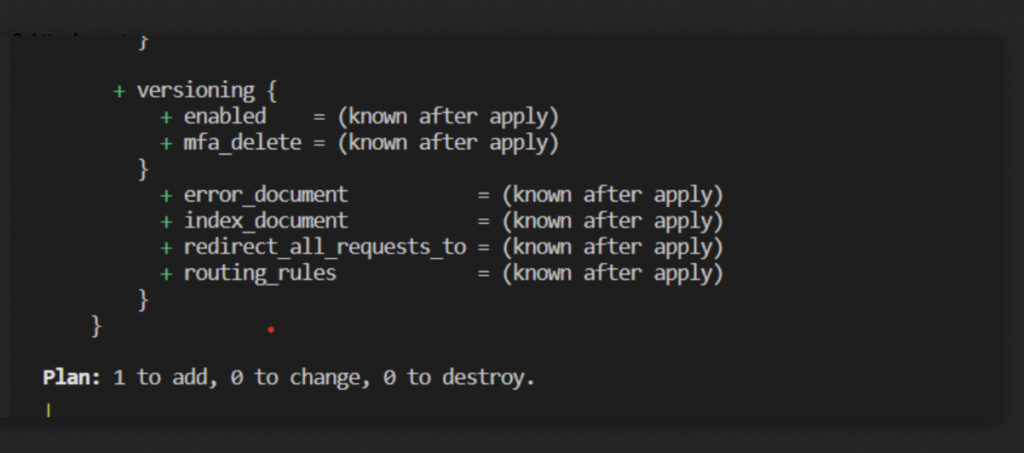
Here is the three [commands](https://www.terraform.io/cli/commands) that is required to execute this script from the CLI:

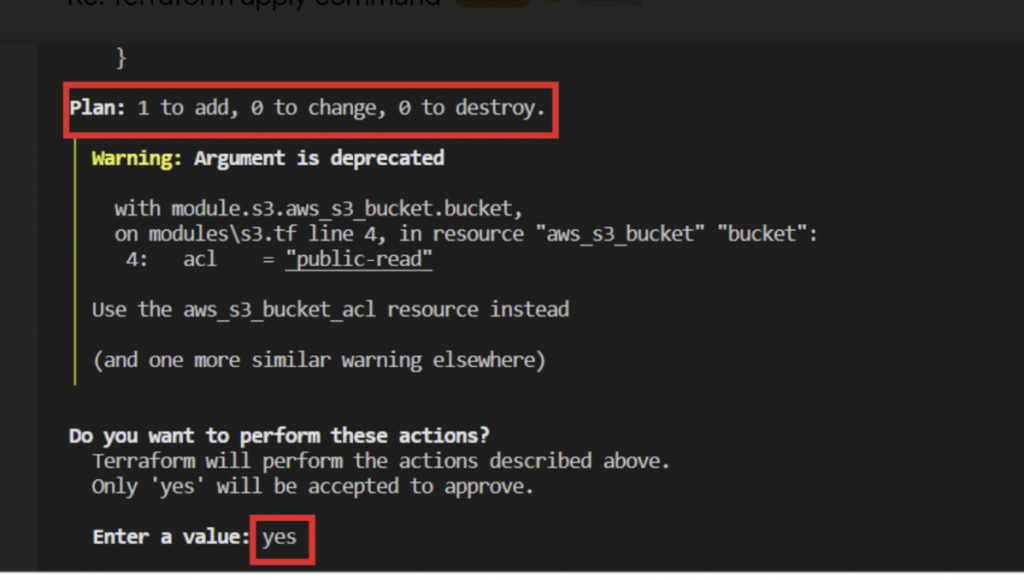
* **init:** This command is used to initialize a working directory containing Terraform configuration files. This is the first command that should be run after writing a new Terraform configuration or cloning an existing one from version control.
* **plan:**The terraform plan command creates an execution plan, which lets you preview the changes that Terraform plans to make to your infrastructure.
* **apply:** This command performs a plan just like terraform plan does, but then actually carries out the planned changes to each resource. It asks for confirmation from the user before making any changes, unless it was explicitly told to skip approval. Use “terraform apply -auto-approve” if you want to execute it without asking for confirmation. So after the apply command execution s3 bucket was created successfully. You can check your bucket in aws console.
* **destroy:**For delete your bucket run terraform destroy command. It shows the destroy plan of our code and asks for confirmation. Give yes to that confirmation and delete the bucket.

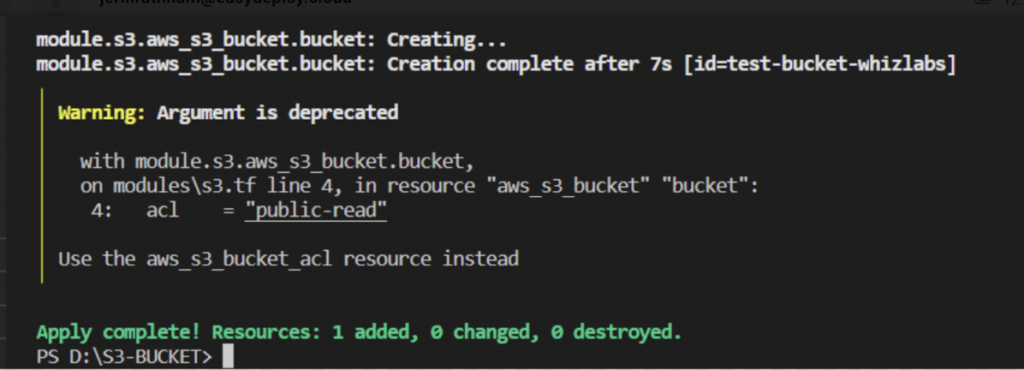
The following screenshots shows that console messages for executing the above commands:



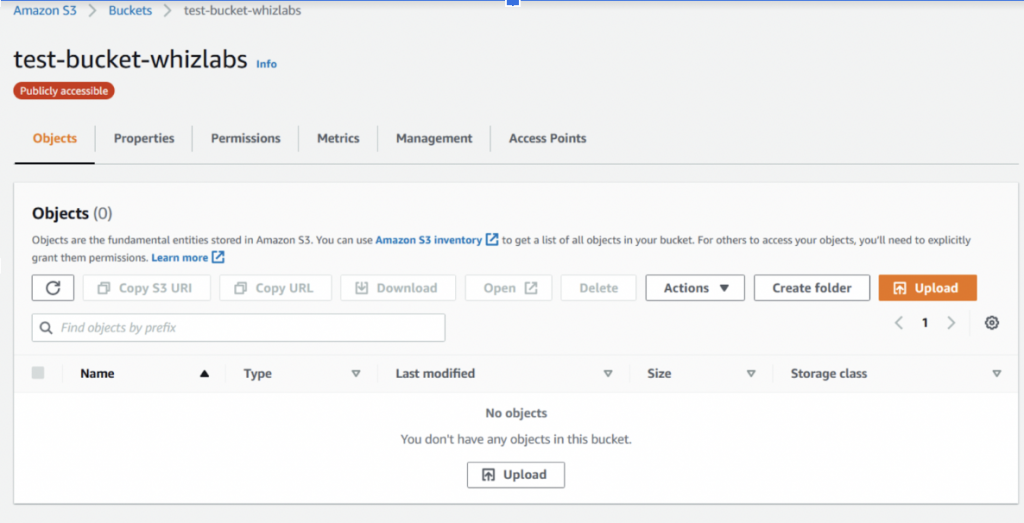
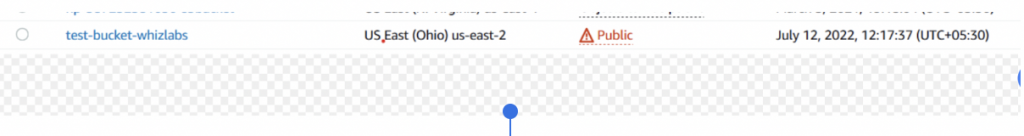








Now you can check the AWS console to see a new S3 bucket is created. Here is how it looks:



With this, we have learned how to create a S3 bucket using terraform scripts through CLI commands. In our next article, we will explain how to achieve the same objective using the **Terraform Cloud**.