**TERRAFORM-05**

**(State locks, creation of s3, dynamolock table, tf provisioners ,and commands)**

**1.Terraform Remote state and state locking:**

**EXPLANATION:**

* **With the help of state file terraform can understand what the actual infrastructure is and what activity should be run by terraform**
* **But If we have multiple people’s then the state file will be on my local, so I have to give the state file to them to get the info of infra**
* **In order to avoid such cases we can store our main.tf and state.tf we can stored in centralized repo but if we stored there and my team-mate are working on same time then terraform will throw errors  
  (We can store terraform configuration files and state file in github or any other repository but it is not good practice.)**
* **In order to avoid that we can use (terraform.lock.file) then it will be locking our state file for the time till our work will get completed**
* **To store our state file in remote we have :**

**S3, TERRAFORM STORAGE, HASHICROP CONSULE**

**With the help of S3 no one have to download the state file he can get easily access by connecting to s3**

**“We can use s3 as remote backend and dynamo db for state locking.”**

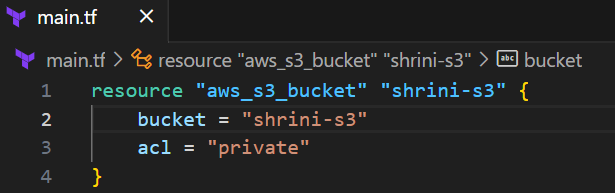
**STATE LOCKING:**

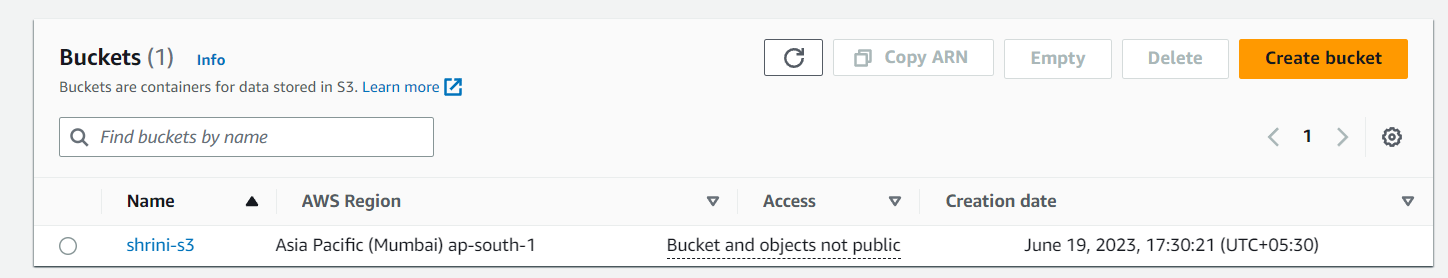
**State locking is used to lock the state file so that no two users can execute the state file at the same point of time.**

****

**Creating S3 as backend:**

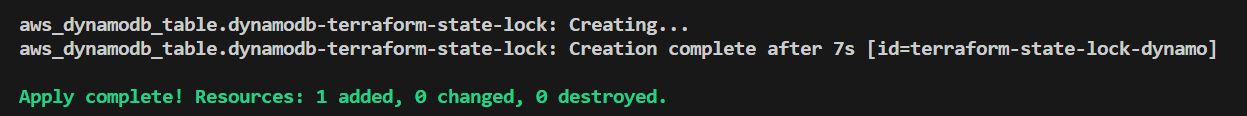
**Bucket will create where we configured our aws region.**

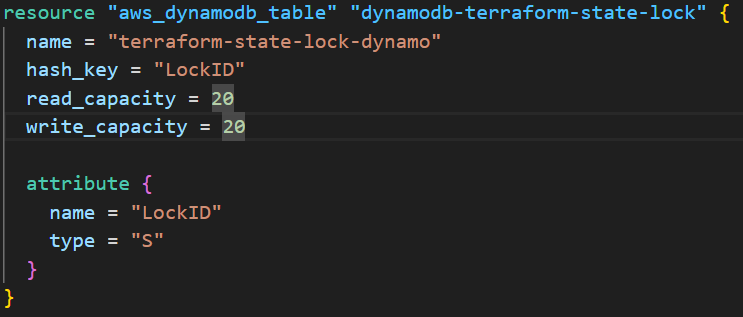
****

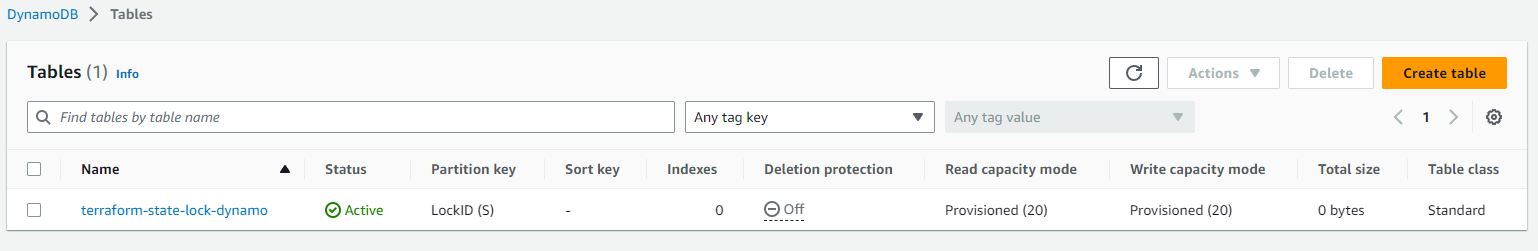
****

**Now we will create dynamoDB using terraform to store our state. Locking file**

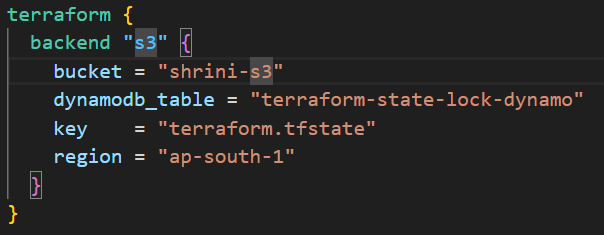
**In the below script we are using {hash\_key = “LockID”}  
for storing the “creation ID” which we will get on state.tf file**

****

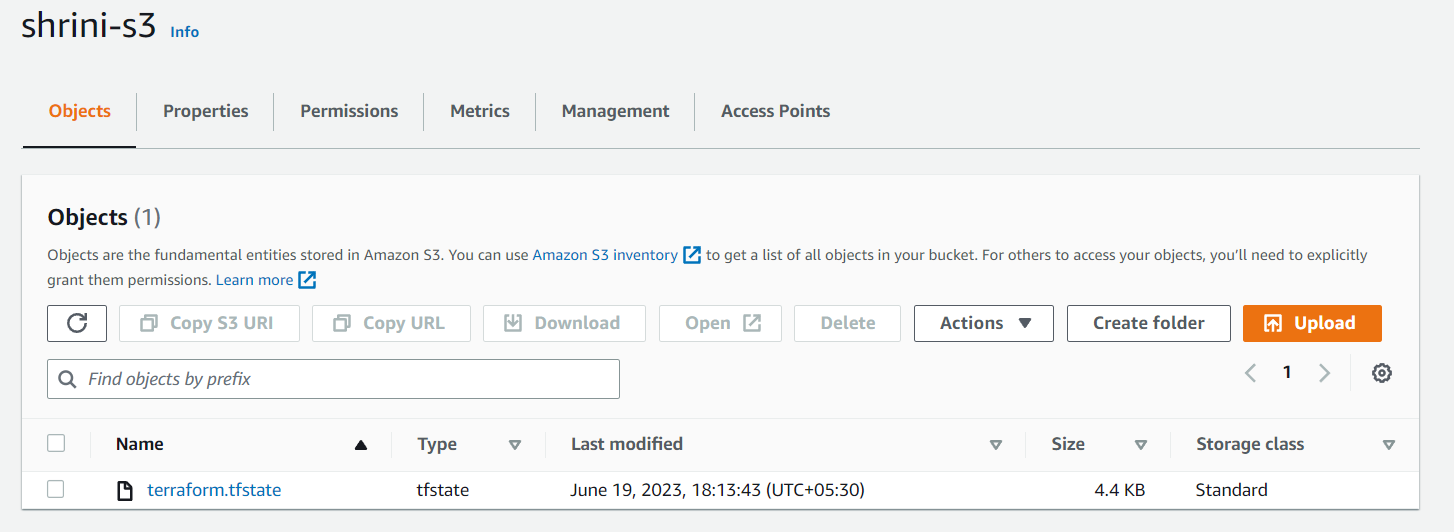
****

****

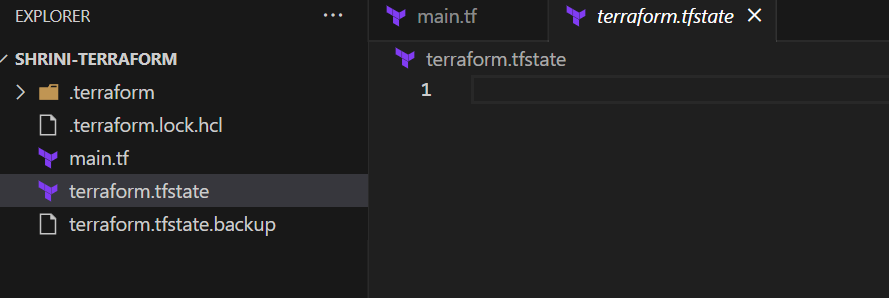
**After we will use S3 as backend for terraform.tfstate file:**

****

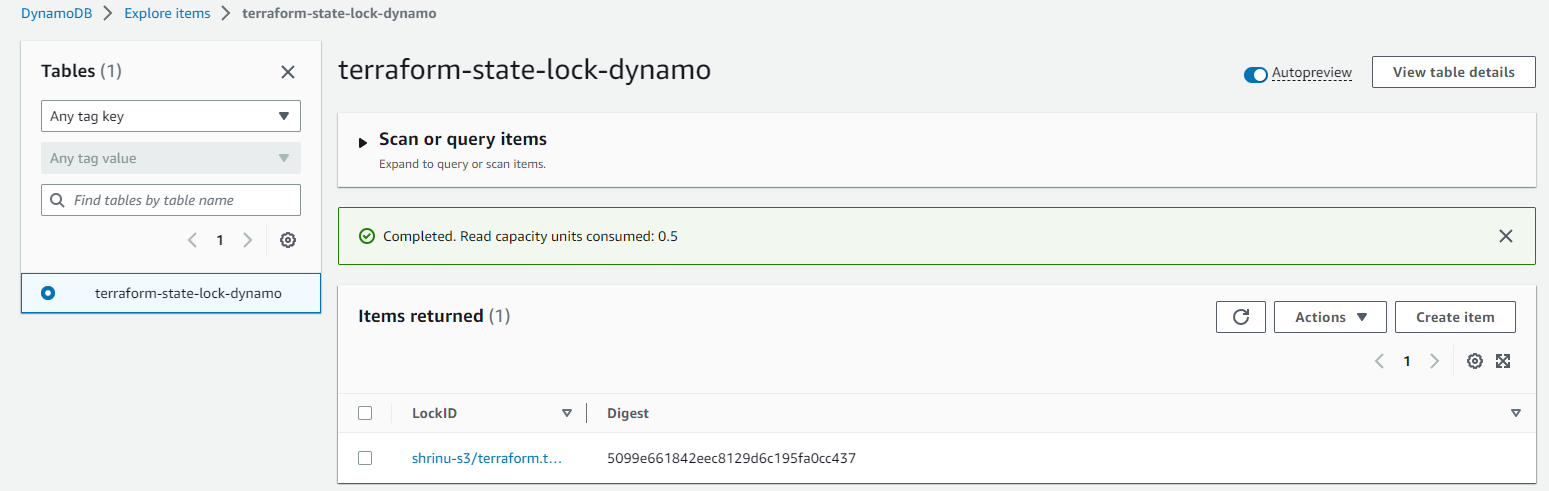
**After running the terraform init command our state file will be goes to s3:**

****

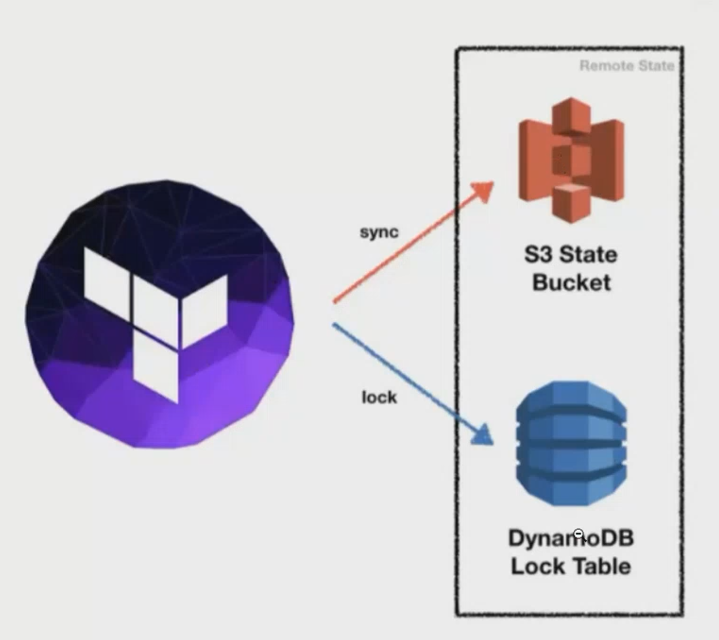
**The state file will be no data in our local state file after initializing s3 as backend:**

****

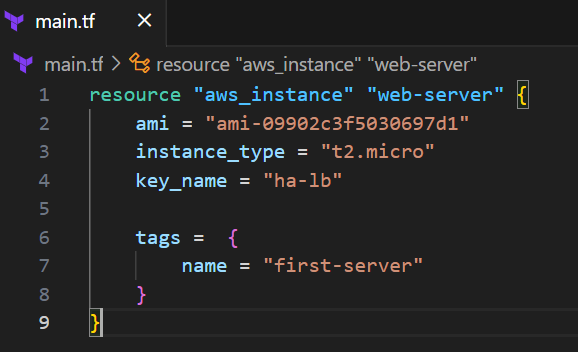
**“DynamoDB supports the state. locking file that’s why we are using it.”**

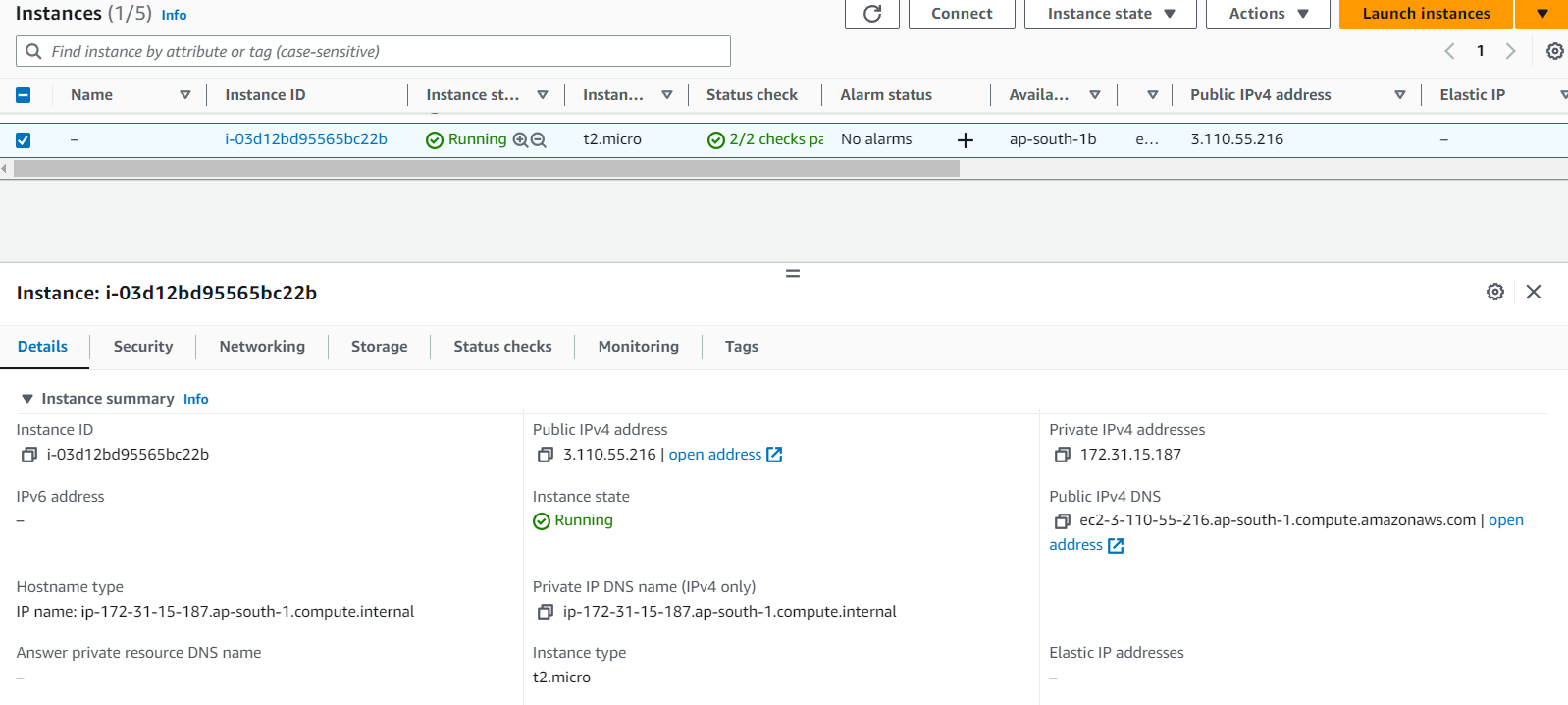
****

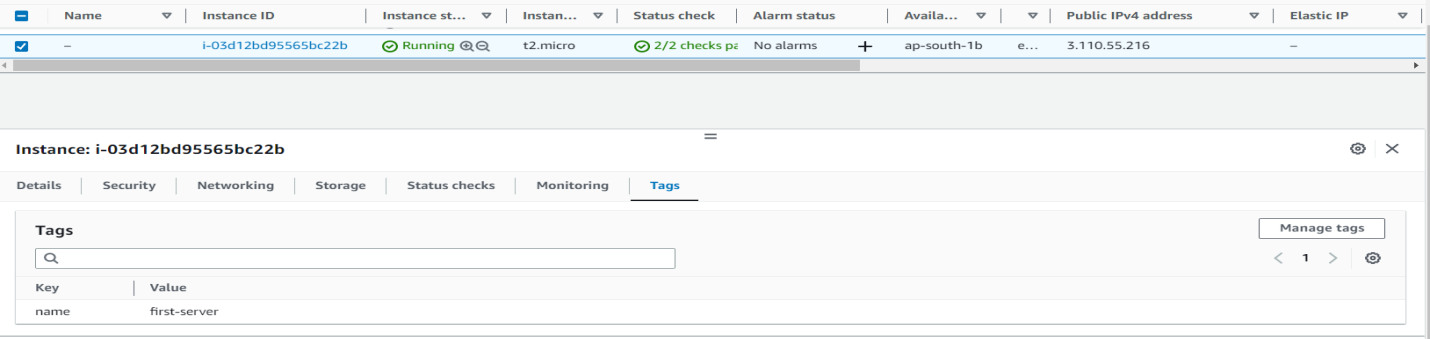
**Structural view of dynamo db:**

****

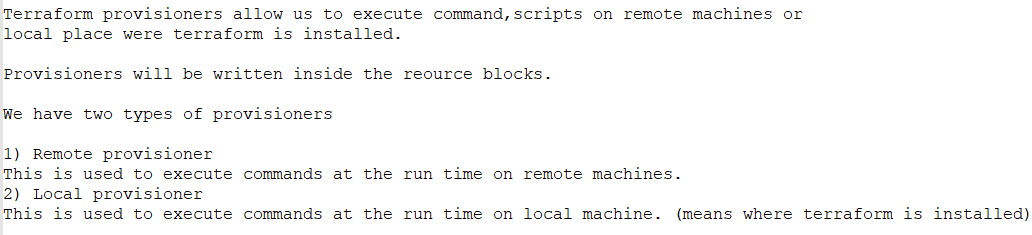
**Creating Ec2-instances and giving tags with the help of terraform:**

****

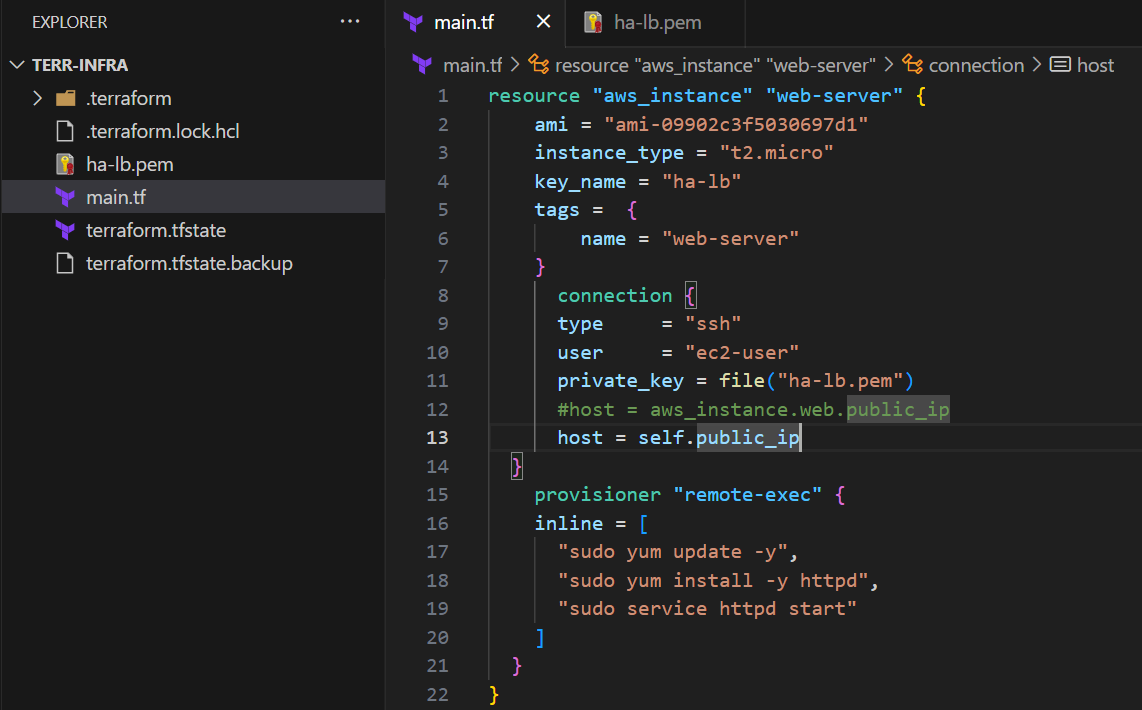
****

****

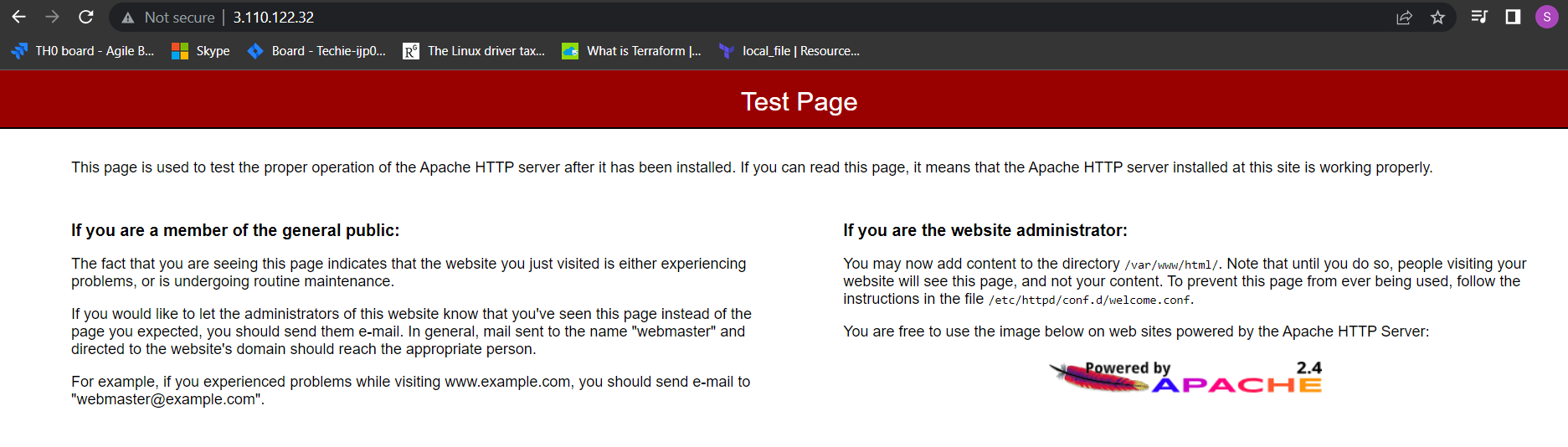
**Terraform provisioners:**



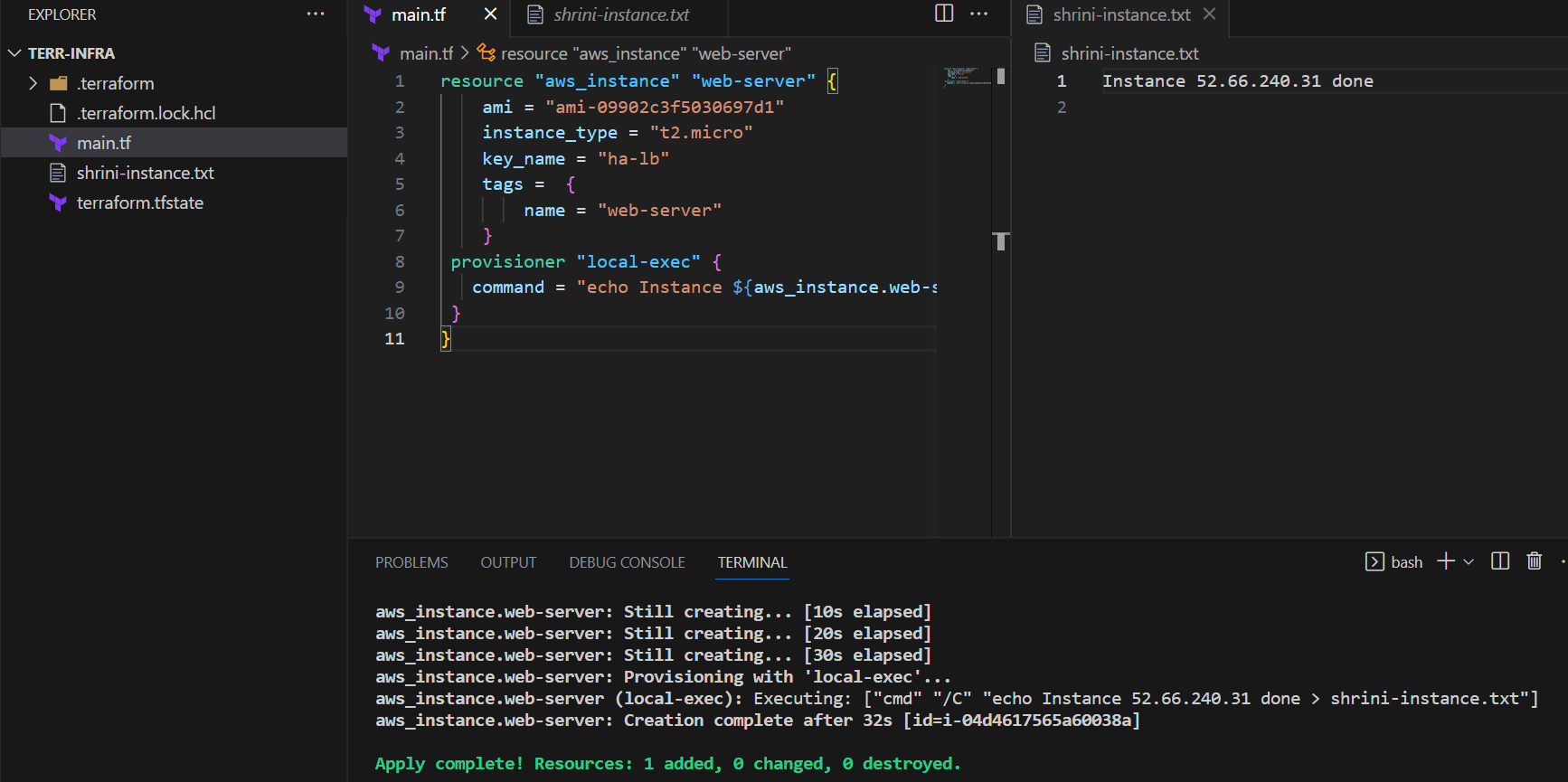
* **Example of remote provisioner:**

****

**“For using remote provisioner we have to give user connection with pem.file”**

****

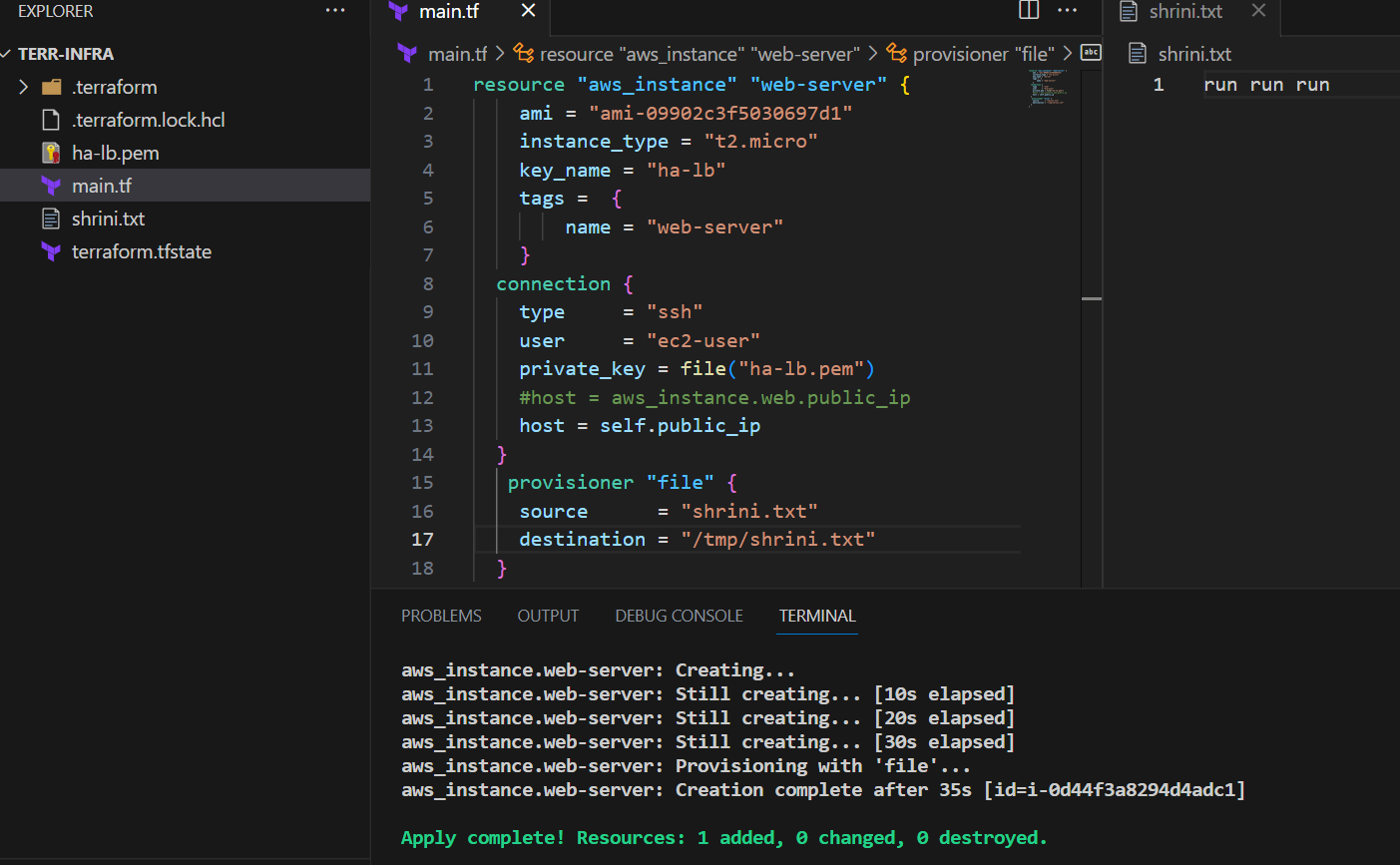
* **Example of local provisioner:**

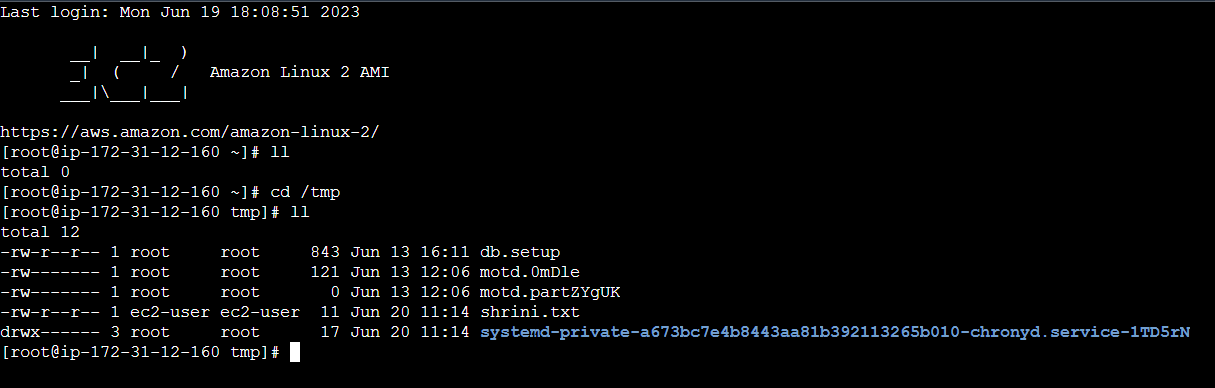
****

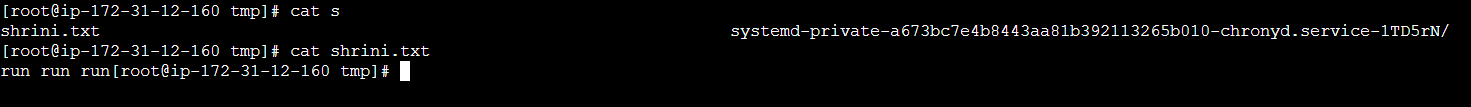
**“Local we don’t have to mention any connection”**

**There is one more type provisioning FILE PROVISIONING which helps to move the file from our local to remote in /tmp/ location on that also we have to give user connection with the help of PEM.KEY file.**

* **Examples of file provisioner:**

****

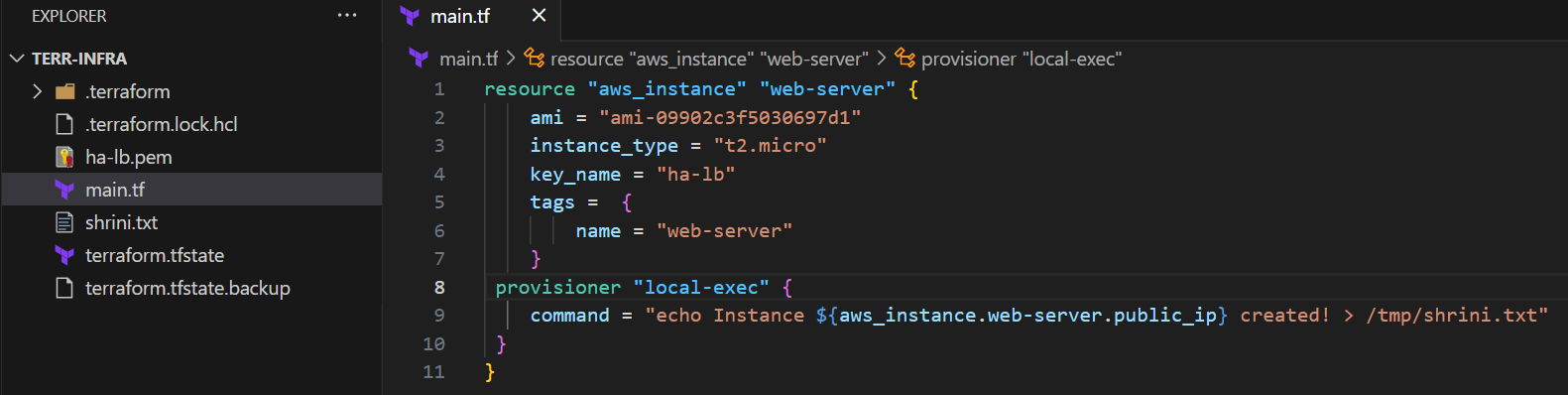
****

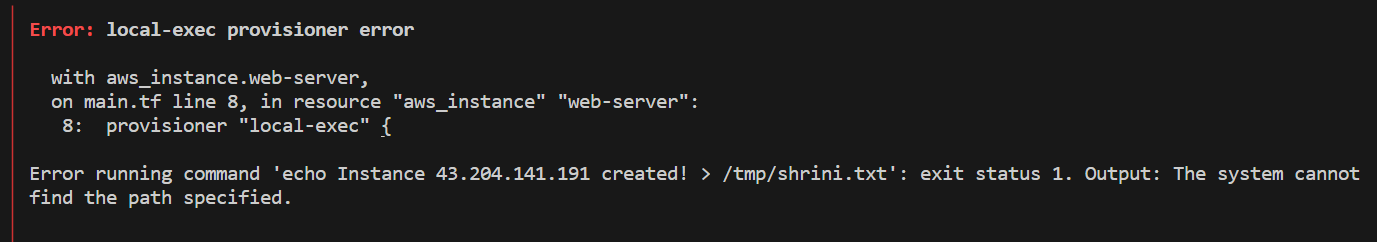
****

**Terraform taint and untaint :**

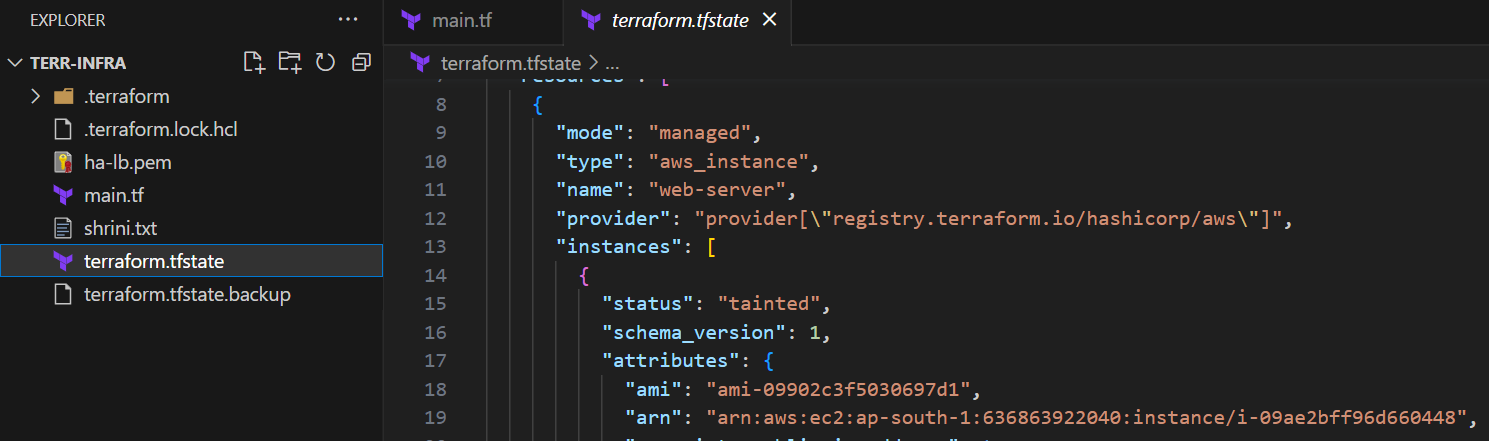
**Terraform taint:**These would be cases when resource creation will get failed, if this happens  
Then, terraform will be marked as "Tainted".  
We can see this when we execute terraform plan command and this will be replaced when we use terraform apply command.

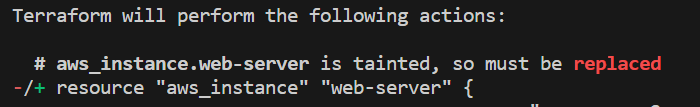
Example:   
Here I am giving wrong location in provisioner block Now this will give error, resources will be created but provisioner not while executing terraform apply

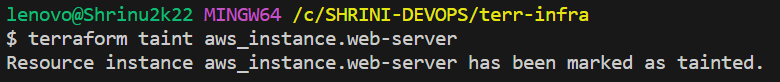
****

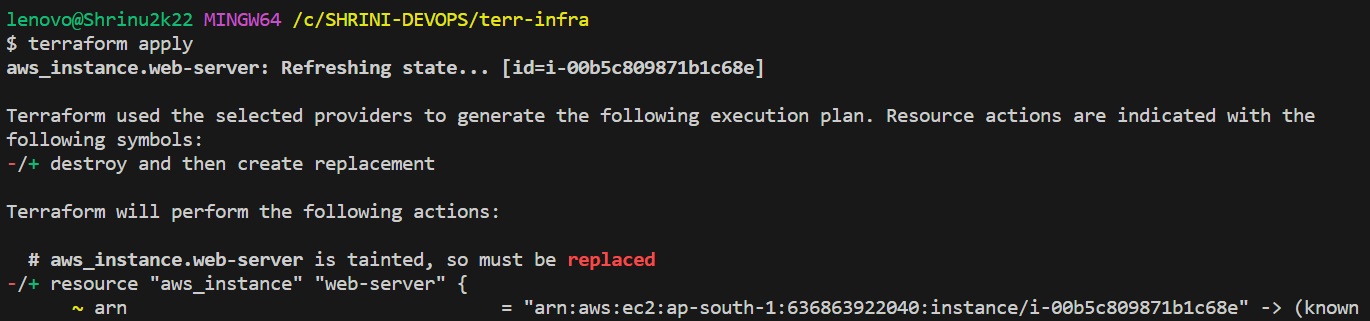
****

And we opened our state file it will show the msg “state.tf” file it will show as “tainted it means  
resource creation will get failed, this will be replaced when we use terraform apply command.

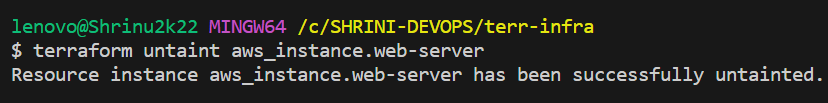


****

We can also do it by command: “terraform taint aws\_instance.web-server” ****

Now I will run the same code it will once again get replaced;  


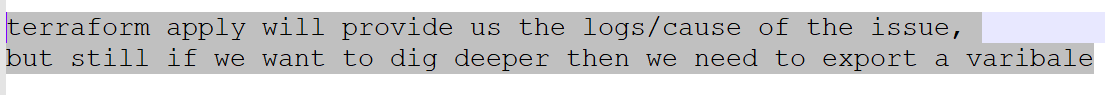
**Terraform untaint:**To undo the changes we can use untaint command.  
 **“terraform untaint aws\_instance.webserver”**

this resource will not be created when using terraform apply.

“With the help of Taint and un taint we can manage our resources should be replaced or not”

**{Terraform taint/un taint providers.logicial name}**

**Debugging:**

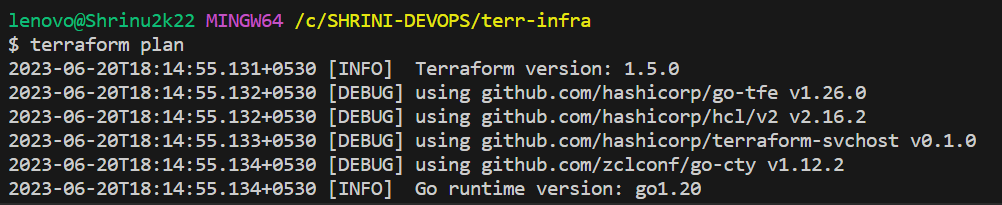
****

We have two types temporary and permanent log files:-

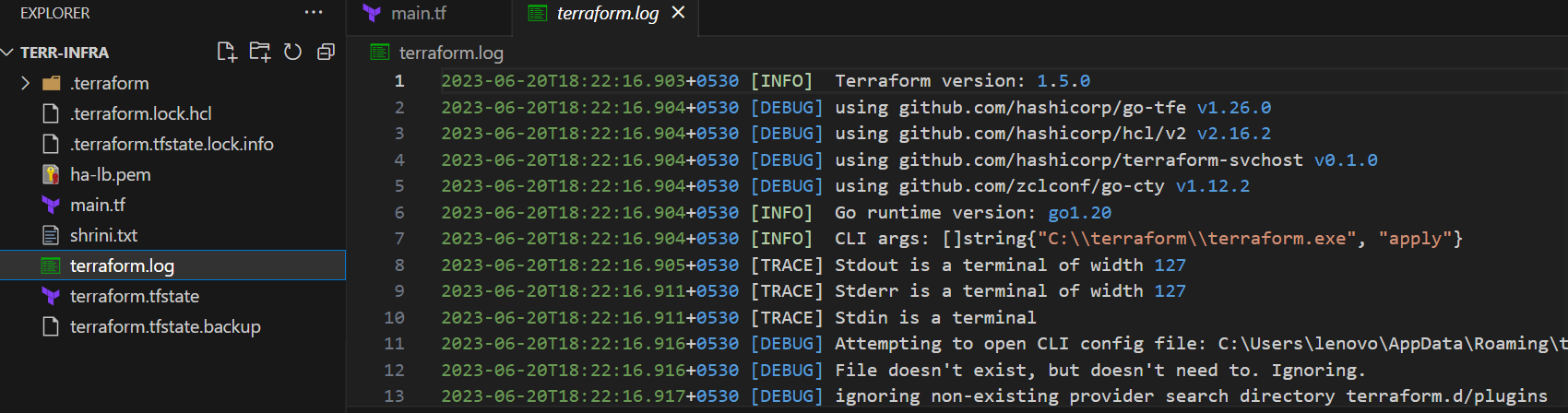
**Terraform provides 5 levels of logs:**

**1) INFO   
2) WARNING  
3) ERROR  
4) DEBUG  
5) TRACE**

According to our requirement we will change the type of logs in below command we are using trace   
export TF\_LOG=TRACE ----linux  
Set-Item -Path env:TF\_LOG -value "TRACE" ----windows  
While executing **“terraform plan”** it will give complete logs in detailed

****

**To store the logs permanently then we can export a path**

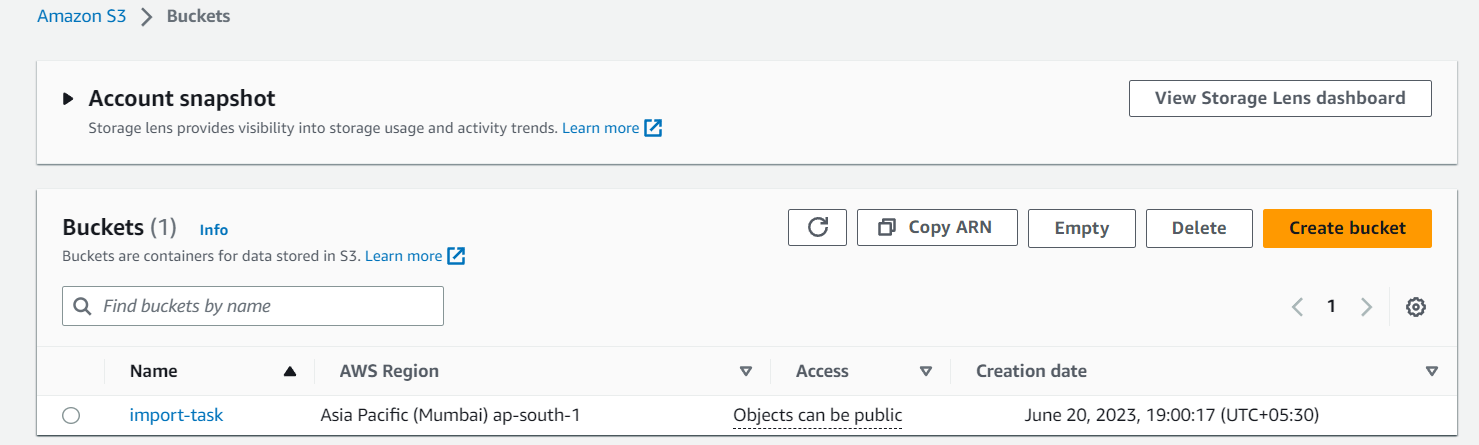
export TF\_LOG\_PATH=/tmp/terraform.log  
Set-Item -Path env:TF\_LOG\_PATH -value "terraform.log" and “terraform apply”  


**Terraform import:**

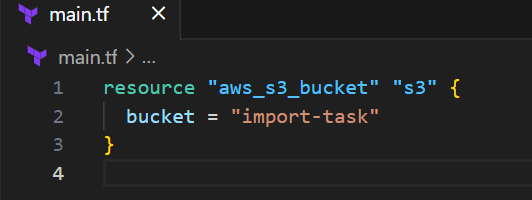
Terraform import is used to import the existing infrastructure in terraform state file.  
Once import is done then we can be able to create/delete and manage the infrastructure.

In order to import any resource we need to write the resource details in configuration file.

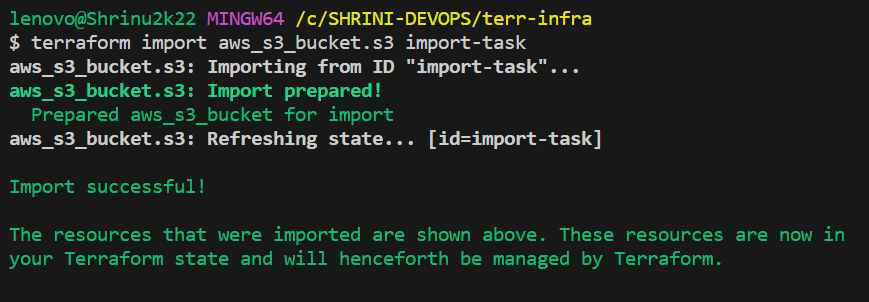
**Importing s3 bucket:**

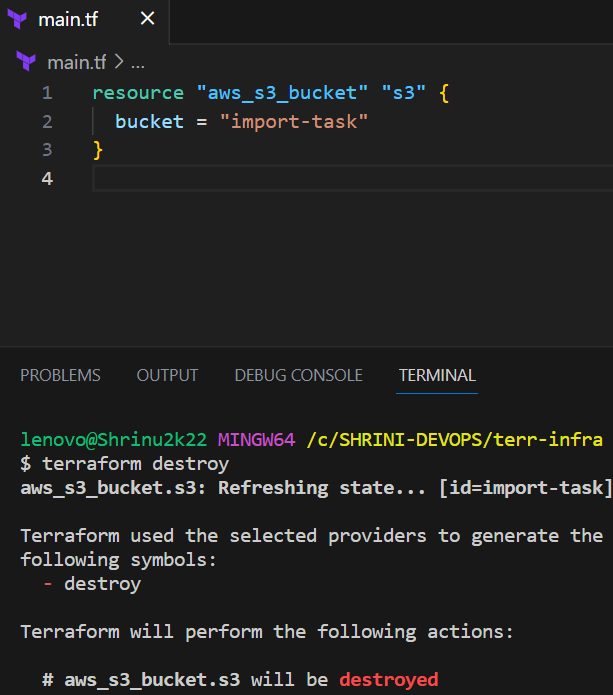
Here we have existing s3-bucket we will try to import 1st and then deleting by terraform:  


Here in the code block I am giving “desired logical name” and using the bucket of existing one:

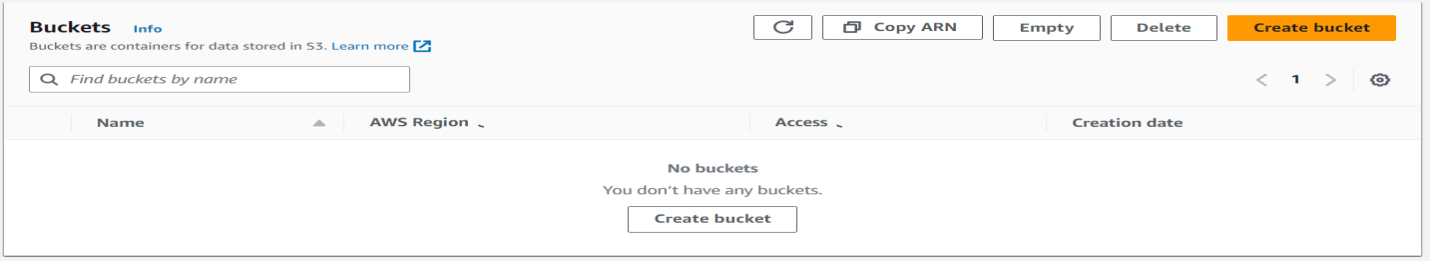


And then executing the command of “terraform import aws\_s3\_bucket.s3 import-task”:-



Import successful, now we can delete the existing bucket also,  


And in aws also the bucket get deleted:



**------------------------------------------------------------------------------------------------------------­**