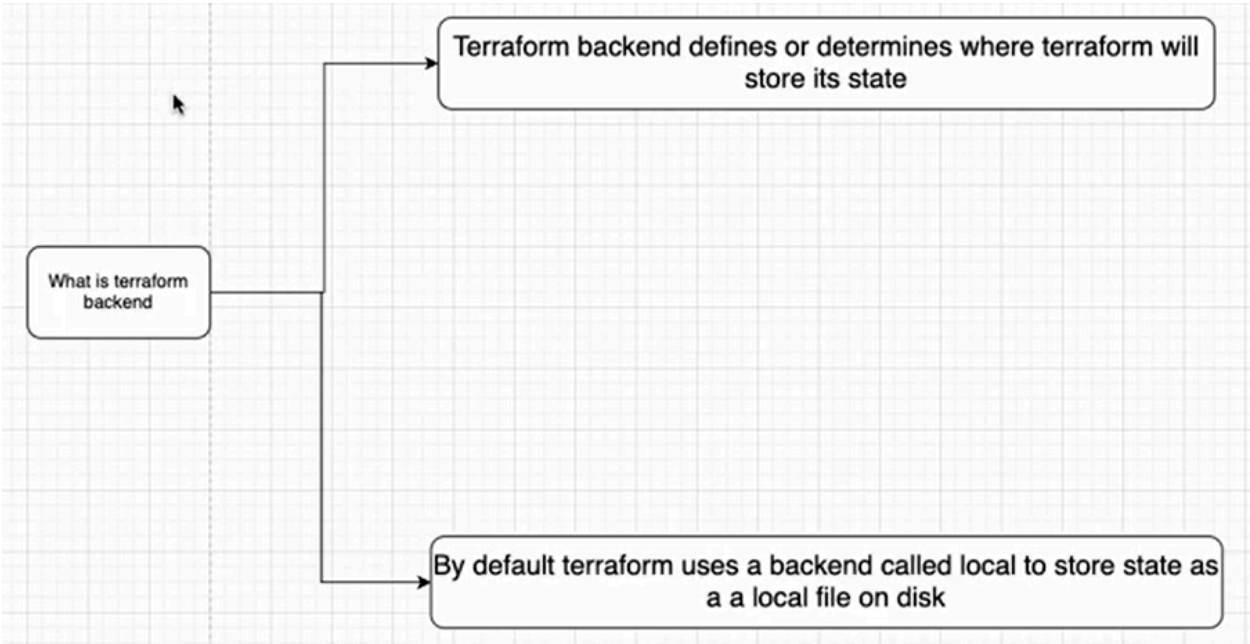
Remote Backend for State File



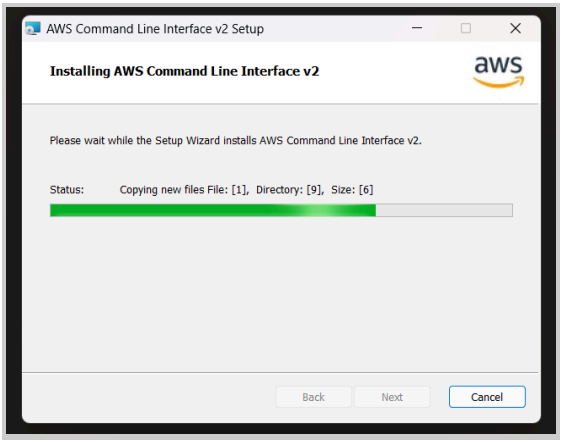
Previously worked on configuring and applying a Terraform configuration.
Terraform created a state file to keep track of the resources it managed

File *terraform.tfstate* stores the current state of my infrastructure: describes the resources to manage, their configurations and current state.

On applying changes, Terraform updates this state file to reflect the new state of the infra

```
root@advika:~# cd terraform
root@advika:~/terraform# ls
 main.tf   terraform.tfstate   terraform.tfstate.backup   variables.tf   '~'
root@advika:~/terraform#
```

Downloaded AWSCLI v2



```
C:\Users\advik>aws --version
'aws' is not recognized as an internal or external command,
operable program or batch file.
```

Added system path to the Path Environment variable

```
C:\Users\advik>aws --version
aws-cli/2.16.12 Python/3.11.8 Windows/10 exe/AMD64
```

Configured AWS:

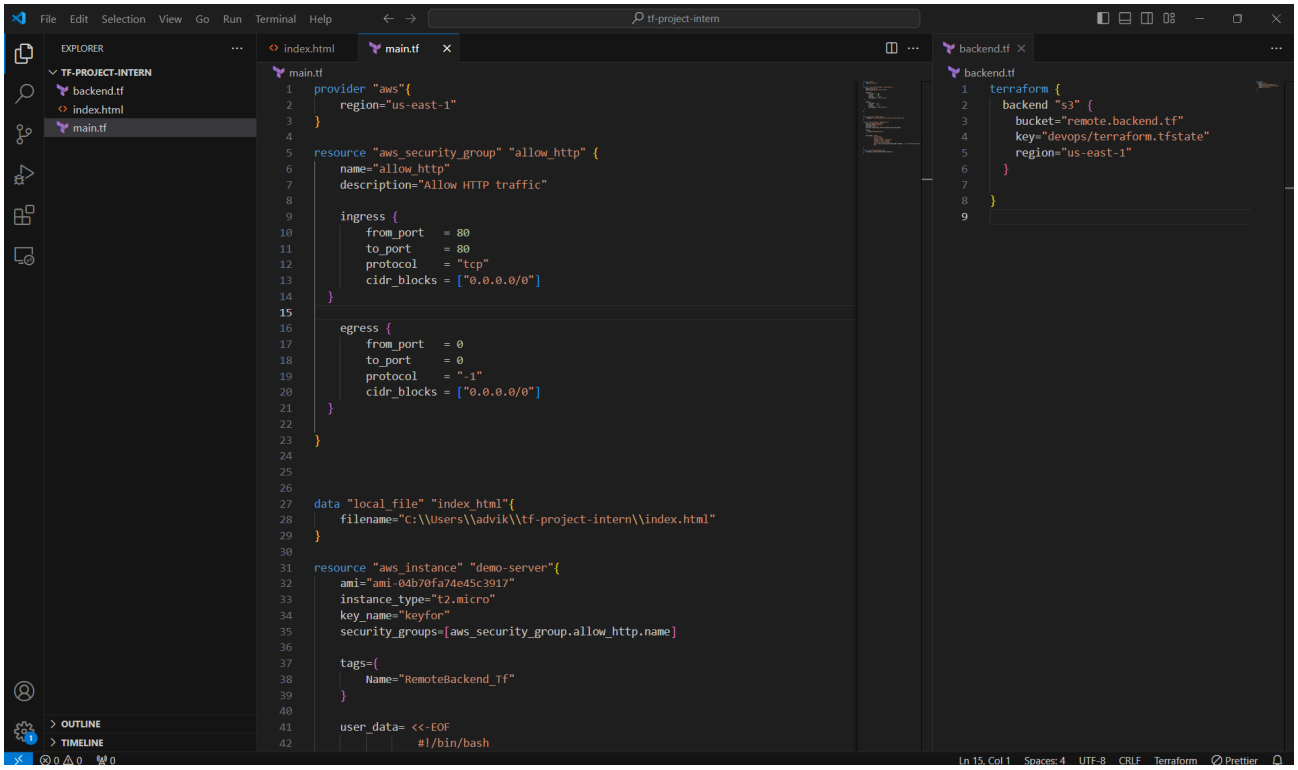
```
C:\Users\advik>aws configure
AWS Access Key ID [****************HPVJ]: AKIA6GBMEYIR
AWS Secret Access Key [****************IayW]: Nx59KPLb
Default region name [us-east-1]: us-east-1
Default output format [None]:
```

Installed and verified Terraform installation:

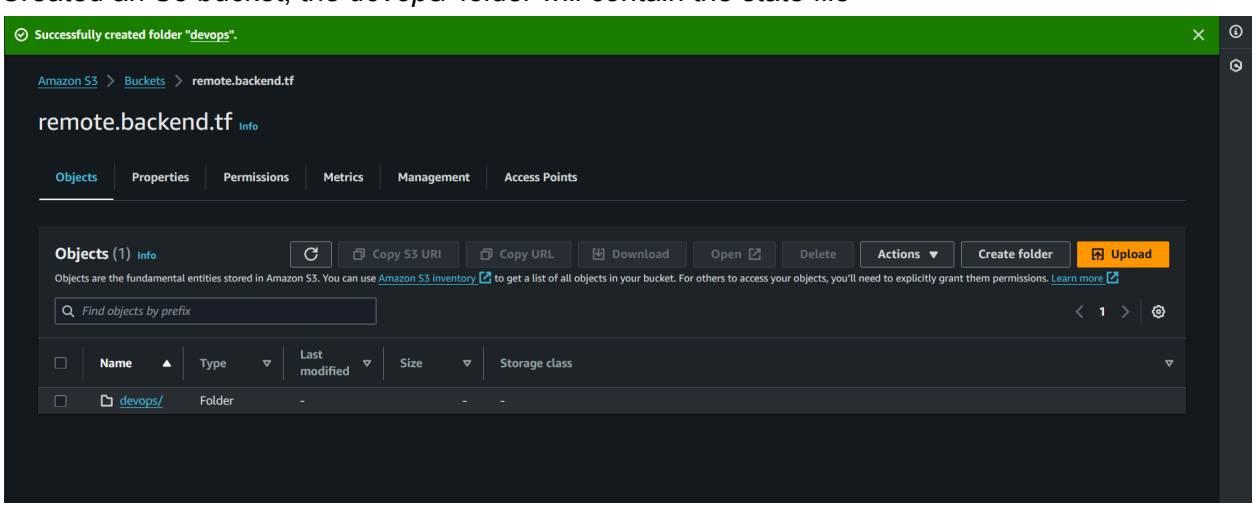
```
C:\Users\advik>terraform --version
Terraform v1.8.5
on windows_amd64
```

CREATION OF RESOURCES

CREATED A DIRECTORY *TF-PROJECT-INTERN*
Created a *main.tf* file, a *backend.tf* file
Added the previously created index.html file to the dir



Created an S3 bucket, the *devops/* folder will contain the state file



UPLOADING STATE FILE TO REMOTE BACKEND i.e., S3 BUCKET

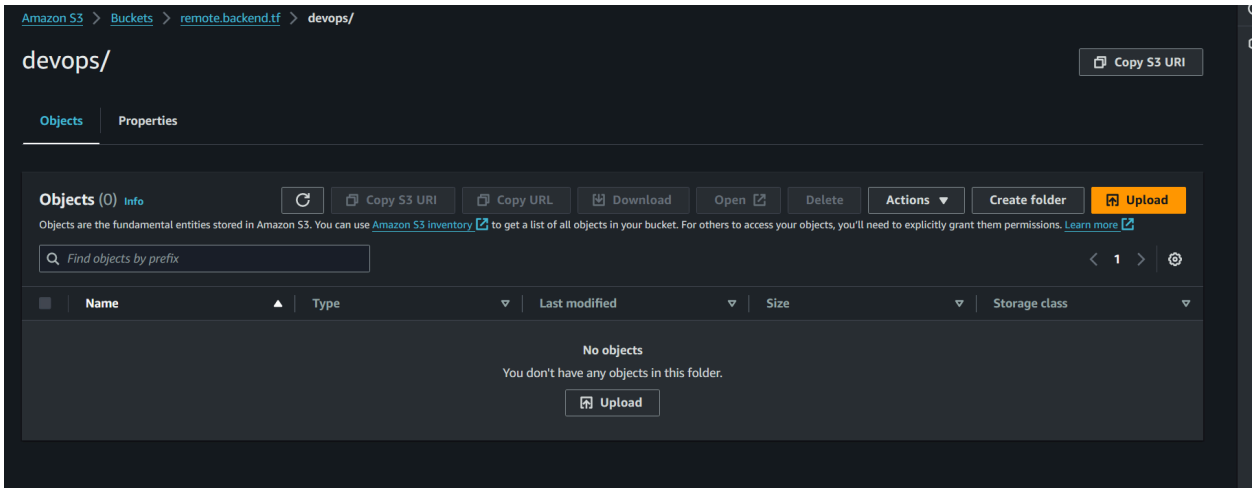
Created a *main.tf* file in the a directory and Performed the necessary terraform *init, validate, plan* and *apply*

```
root@advika: ~/tf-intern-proj

root@advika:~/tf-intern-project# ls
root@advika:~/tf-intern-project# vi main.tf
root@advika:~/tf-intern-project# ls
main.tf
root@advika:~/tf-intern-project# terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.1.0"...
- Finding latest version of hashicorp/local...
- Installing hashicorp/aws v5.1.0...
- Installed hashicorp/aws v5.1.0 (signed by HashiCorp)
- Installing hashicorp/local v2.5.1...
- Installed hashicorp/local v2.5.1 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

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.
root@advika:~/tf-intern-project# ls
main.tf
root@advika:~/tf-intern-project#
```

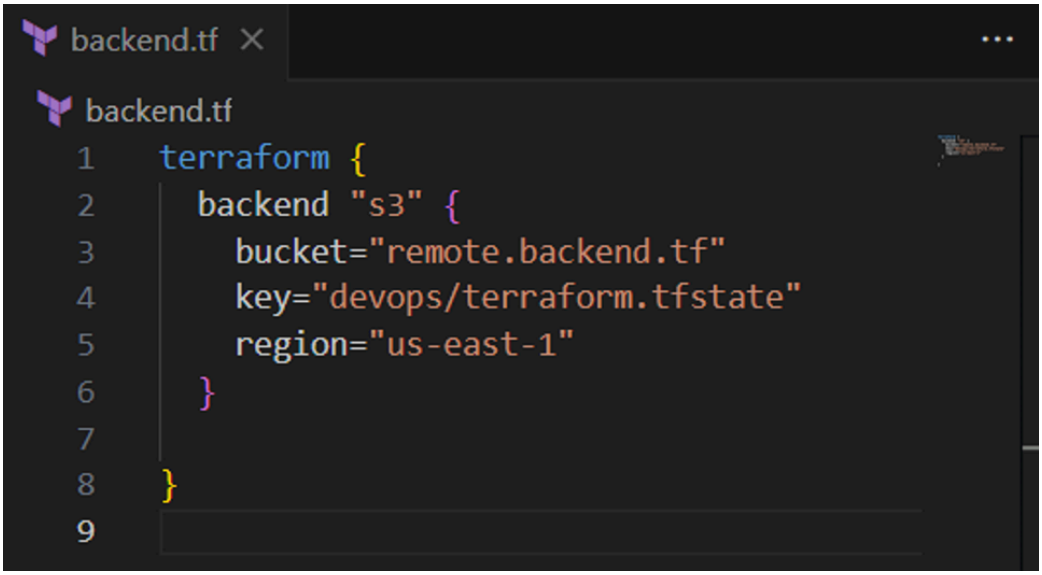
```
root@advika:~/tf-intern-project# terraform validate
Success! The configuration is valid.

root@advika:~/tf-intern-project# terraform plan
data.local_file.index_html: Reading...
data.local_file.index_html: Read complete after 0s [id=4b46429e39ef9148a8c7e6ec57d7182c465ea15e]

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_instance.demo-server will be created
  + resource "aws_instance" "demo-server" {
```

```
ec2_instance_public_ip = "34.228.29.209"
root@advika:~/tf-intern-project# ls
main.tf  terraform.tfstate
root@advika:~/tf-intern-project#
```

The *devops/* folder was empty because the backend hasn't been initialised to S3 yet



Then I created an alt backend.tf file that upload the state file onto an S3 bucket folder



```
root@advika:~/tf-intern-project# ls
main.tf  terraform.tfstate
root@advika:~/tf-intern-project# vi backend.tf
root@advika:~/tf-intern-project# ls
backend.tf  main.tf  terraform.tfstate
root@advika:~/tf-intern-project# terraform init

Initializing the backend...
Do you want to copy existing state to the new backend?
  Pre-existing state was found while migrating the previous "local" backend to the
  newly configured "s3" backend. No existing state was found in the newly
  configured "s3" backend. Do you want to copy this state to the new "s3"
  backend? Enter "yes" to copy and "no" to start with an empty state.

  Enter a value: yes


Successfully configured the backend "s3"! Terraform will automatically
use this backend unless the backend configuration changes.

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Reusing previous version of hashicorp/local from the dependency lock file
- Using previously-installed hashicorp/aws v5.1.0
- Using previously-installed hashicorp/local v2.5.1

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.
root@advika:~/tf-intern-project# ls
backend.tf  main.tf  terraform.tfstate  terraform.tfstate.backup
root@advika:~/tf-intern-project#
```

The state file is uploaded to the key path on my S3 bucket!