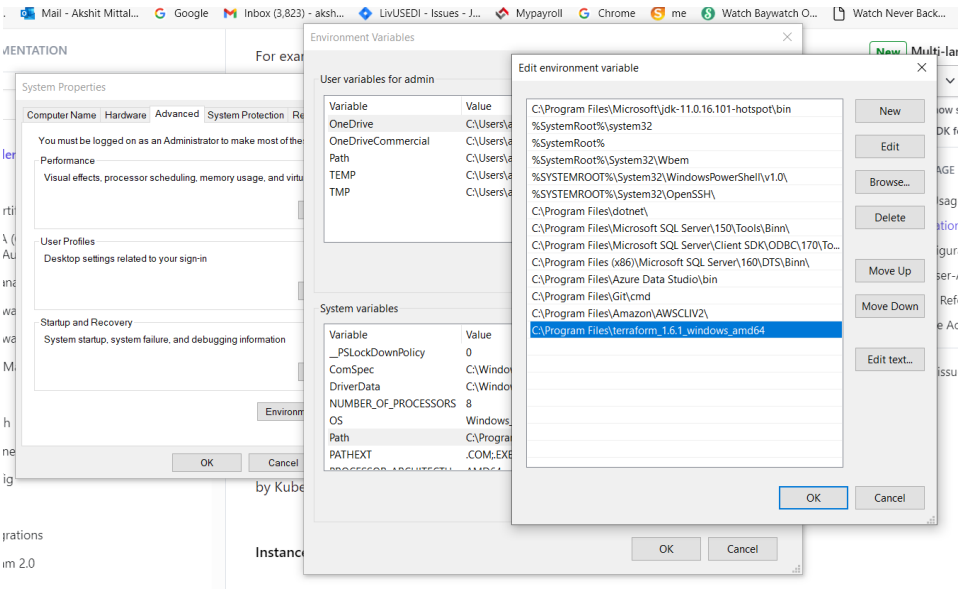
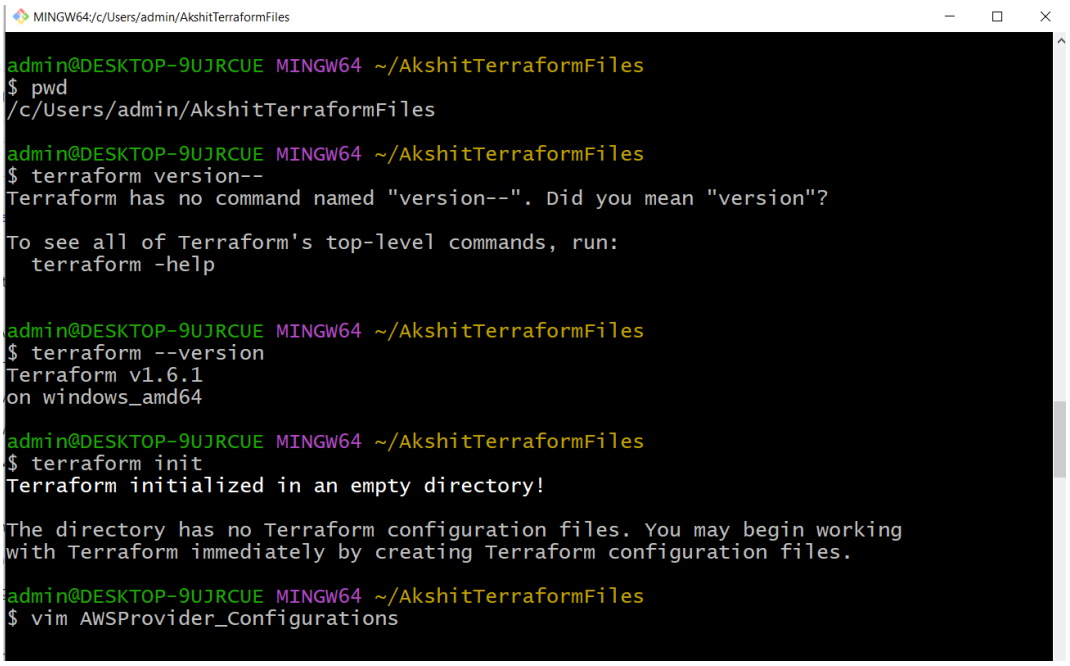
Connect TERRAFORM with AWS

Link:-
- HashiCorp AWS documentation- [Docs overview | hashicorp/aws | Terraform | Terraform Registry]

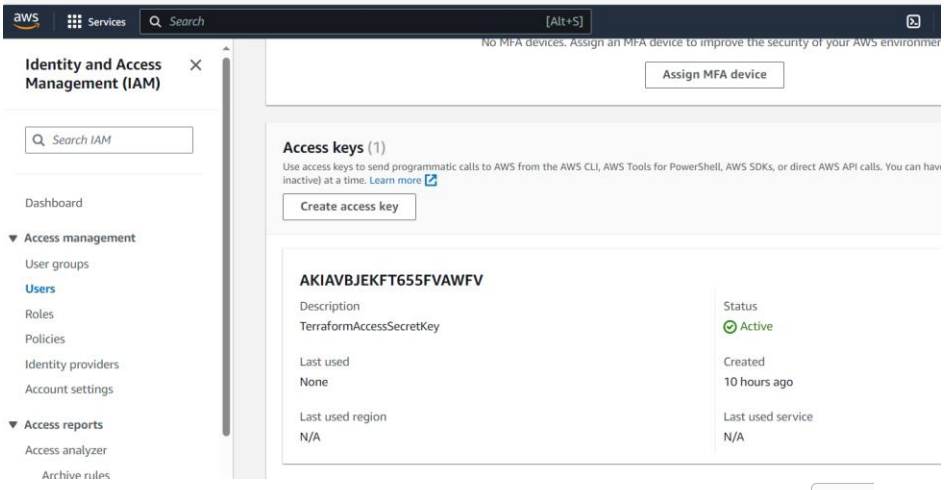
Context:- Terraform is an Infrastructure as Code (IaC) tool used to create and manage infrastructure through code. To do so, it requires machine configurations to be provided on which the infrastructure is to be deployed. Typically, the provider used is a cloud service provider, such as AWS. Terraform initializes itself based on the AWS configurations provided to connect to AWS. We will be doing this scenario from Step1 TO Step5

1. Install Terraform on your system → (if windows) : add the terraform folder directory path to your system Environmental Variable Path
   Install AWS CLI on your system → (if windows) : add the AWS CLI folder directory path to your system Environmental Variable Path (USED IN STEP 4)
   → login into your CLI as the Root user OR open bin Bash as administrator (if windows) and check the pwd (power working directory) which should be the root or admin directory as seen the the screenshot. → make the terraform files folder in which you will save all files related to terraform and go into that directory/folder → check if terraform is successfully installed by checking the terraform version → initialize terraform by init command (it will get failed because we have not given the provider configuration file yet) → make a config file in the same terraform folder/directory with the extension .tf (terraform does not recognize the config file without the .tf extension)





2. AWS Provider Configuration File (format of all files are been taken from documentation of Terraform in AWS section provided by Hashi-corp) (Region, Secret and Access Key is been taken from the AWS by creating User in the IAM role) → save the file



Provider Configuration

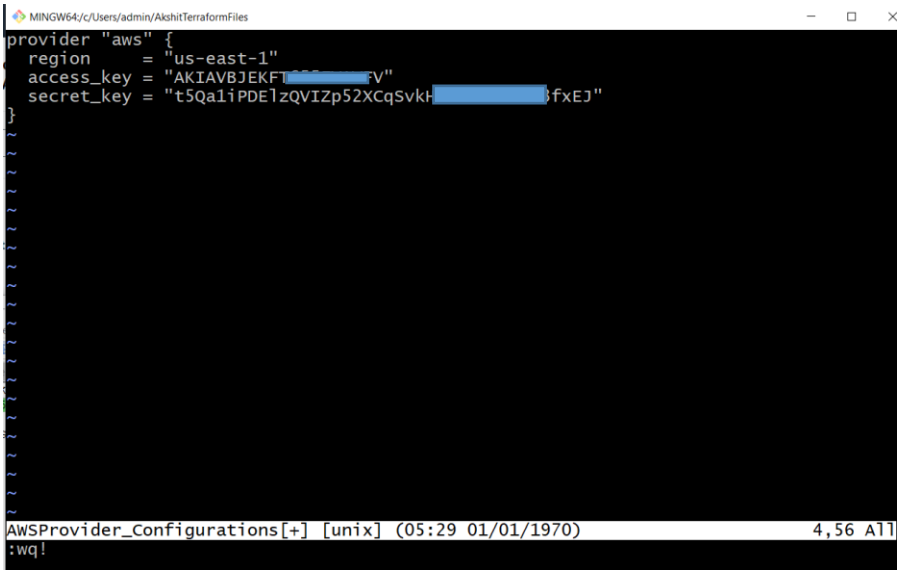
Warning:

Hard-coded credentials are not recommended in any Terraform configuration and risks secret leakage should this file ever be committed to a public version control system.

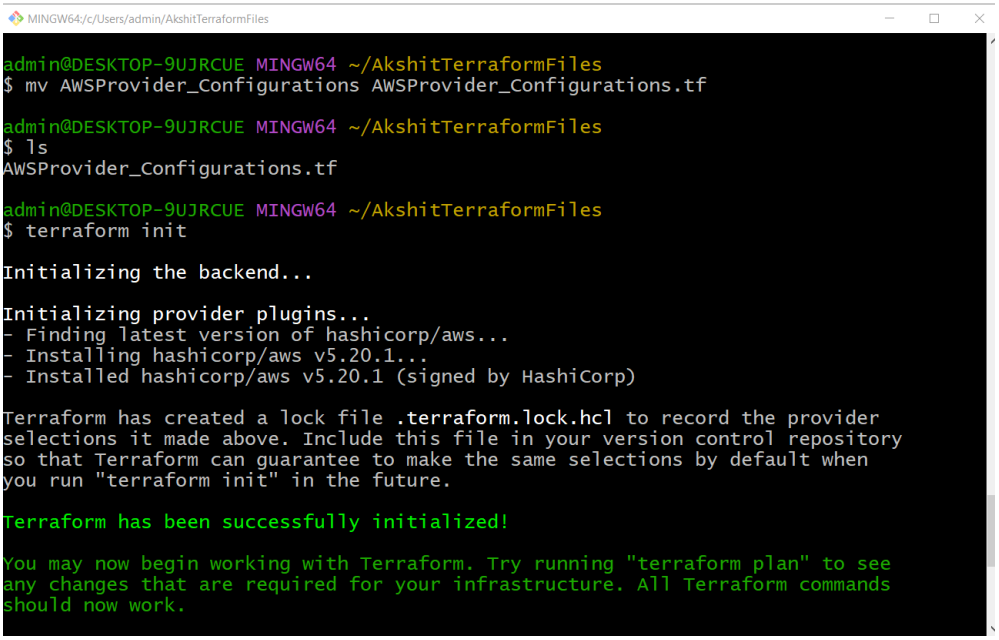
Credentials can be provided by adding an `access_key`, `secret_key`, and optionally `token`, to the `aws` provider block.

Usage:

```
provider "aws" {
  region     = "us-west-2"
  access_key = "my-access-key"
  secret_key = "my-secret-key"
}
```



3. Run the terraform init command on the terraform files directory which contains the config file. It can be seen in the screenshot below that we have to rename the config file with no extension to .tf extension. We did that to check if terraform needs the .tf extension to recognize the config file . And the answer is YES.
   Now the Terraform is successfully connected to AWS



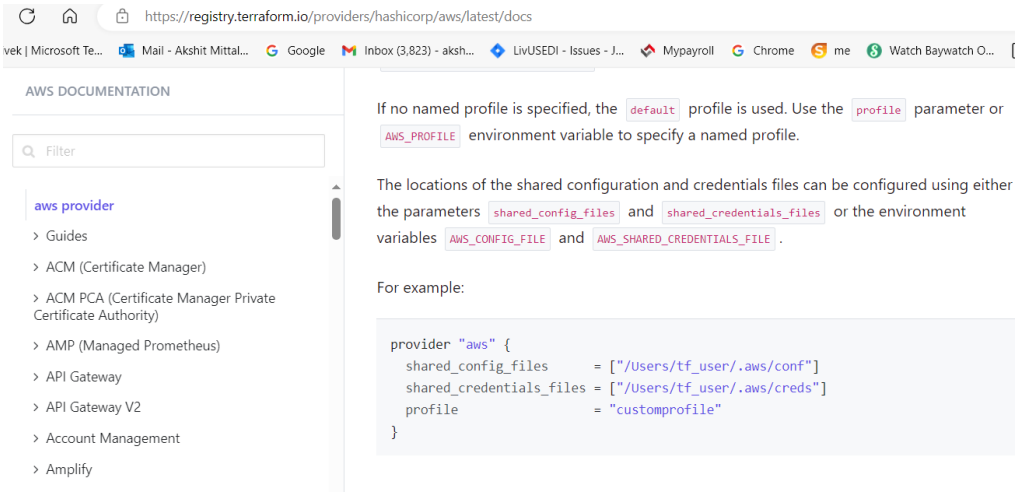
4. Now
   We want to give credentials securely at different location and not HardCode it in the AWSProvider_configuration file.
   Go into the main admin directory where .aws directory/folder is present after installing AWS CLI and go to .aws directory → command aws configure, where you will fill the same secret and access key again. Fill the values and check the credential file in .aws by concatenate command. You will find both keys values there.

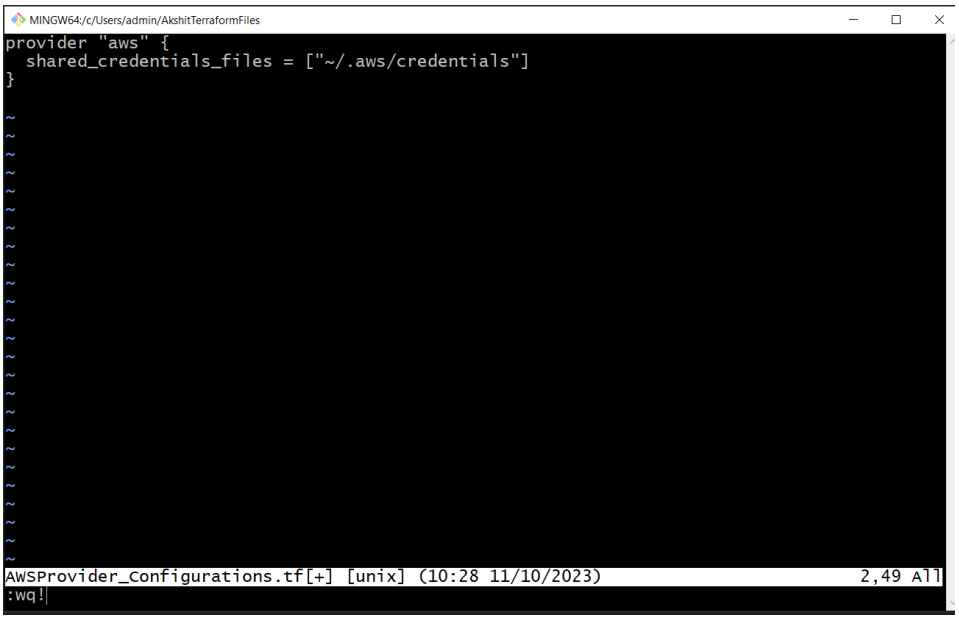
```
admin@DESKTOP-9UJRCUE MINGW64 ~/.aws
$ ls
config  credentials

admin@DESKTOP-9UJRCUE MINGW64 ~/.aws
$ cat credentials
[default]
aws_access_key_id = AKIAVBJE[redacted]FVAWFV
aws_secret_access_key = t5Qa1iPDElzQVIZp52XCqSv[redacted]xd53fxEJ
```

5. Come to the terraform folder and open awsprovider_configuration file again. Refactor the file as per following giving the path of the AWS credentials → Run the terraform init command again and it will be successfully running
   Please note* We are taking all the file formats from the HashiCorp Terraform website → AWS documentations section.

   Now, whenever you need to give new Access/Secret Key, you can just command AWS Configure in .aws directory and fill the new keys there and terraform will automatically connect to it






```
admin@DESKTOP-9UJRCUE MINGW64 ~/AkshitTerraformFiles
$ vim AWSProvider_Configurations.tf

admin@DESKTOP-9UJRCUE MINGW64 ~/AkshitTerraformFiles
$ terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.20.1

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.

admin@DESKTOP-9UJRCUE MINGW64 ~/AkshitTerraformFiles
$
```