Terraform – AWS Lab Work

First, we will create aws profile so that no one can see our credentials of our IAM user

```
C:\Users\HP\Documents\terra>aws configure --profile mylogin
AWS Access Key ID [****************HQOY]:
AWS Secret Access Key [****************hQX2]:
Default region name [ap-south-1]:
Default output format [None]:

C:\Users\HP\Documents\terra>aws configure list-profiles
default
mylogin
```

Now, we will make our file with name my.tf :-

my.tf - Notepad

File Edit Format View Help

```
provider "aws"{
     region = "ap-south-1"
     profile = "mylogin"
}
resource "aws_instance" "web" {
ami           = "ami-052c08d70def0ac62"
  instance_type = "t2.micro"
}
```

Now, in command prompt, terraform init command downloads plugin for aws:-

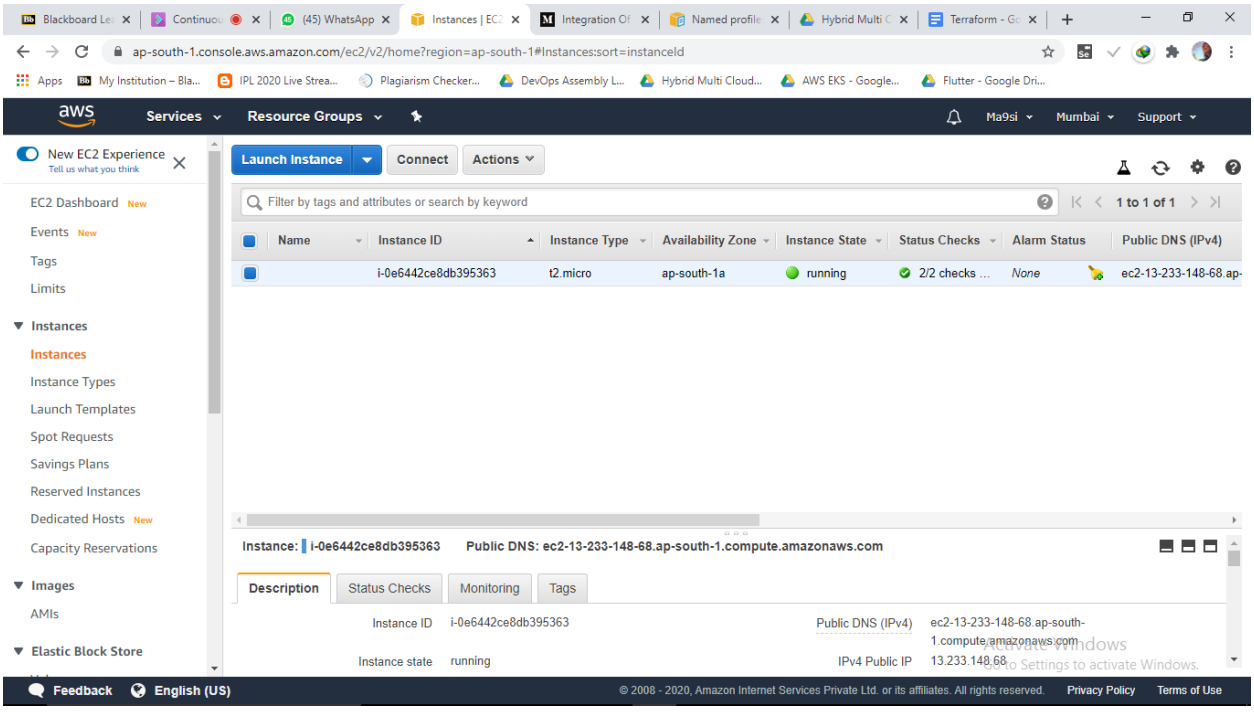
```
C:\Users\HP\Documents\terra>terraform init

Initializing the backend...

Initializing provider plugins...
- Checking for available provider plugins...
- Downloading plugin for provider "aws" (hashicorp/aws) 3.4.0...

The following providers do not have any version constraints in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking
changes, it is recommended to add version = "..." constraints to the
corresponding provider blocks in configuration, with the constraint strings
suggested below.

* provider.aws: version = "~> 3.4"

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.
```

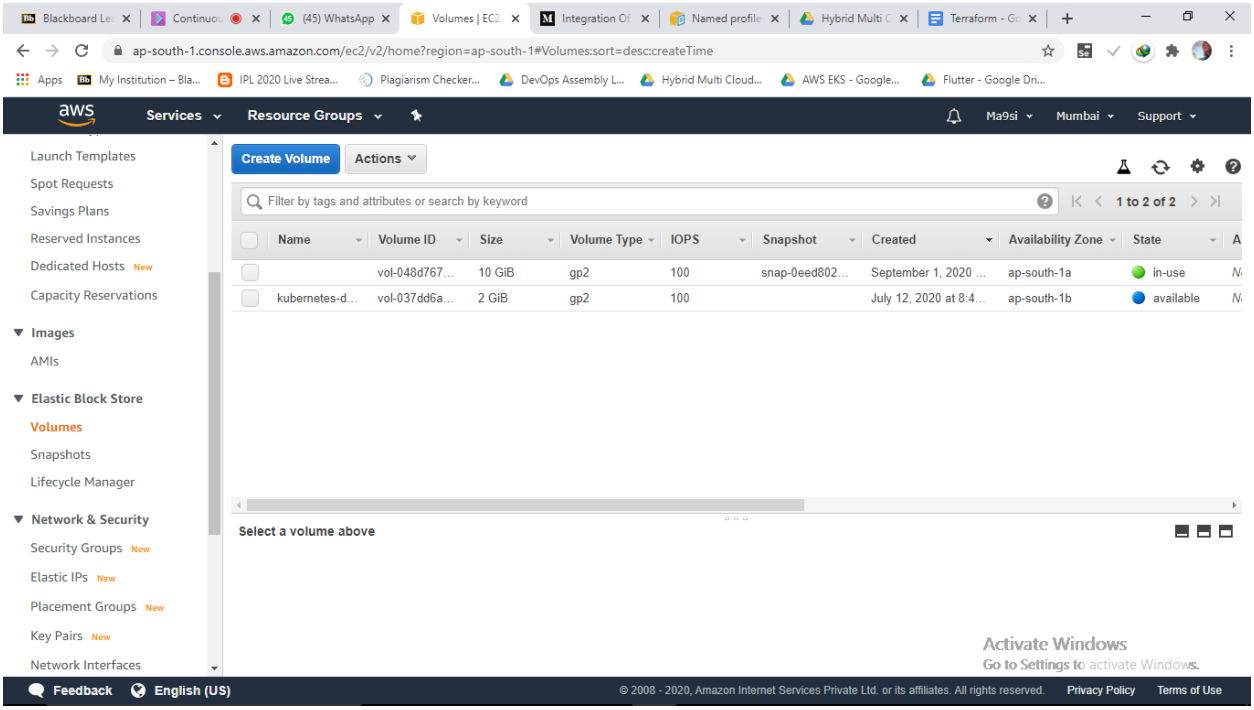
Now, we will run terraform apply command to run our terraform file:-

```
C:\Users\HP\Documents\terra>terraform apply

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.web will be created
  + resource "aws_instance" "web" {
      + ami                          = "ami-052c08d70def0ac62"
      + arn                          = (known after apply)
      + associate_public_ip_address  = (known after apply)
      + availability_zone            = (known after apply)
      + cpu_core_count               = (known after apply)
      + cpu_threads_per_core         = (known after apply)
      + get_password_data            = false
      + host_id                      = (known after apply)
      + id                           = (known after apply)
      + instance_state               = (known after apply)
      + instance_type                = "t2.micro"
      + ipv6_address_count           = (known after apply)
      + ipv6_addresses               = (known after apply)
      + key_name                     = (known after apply)
      + outpost_arn                  = (known after apply)
      + password_data                = (known after apply)
      + placement_group              = (known after apply)
      + primary_network_interface_id = (known after apply)
      + private_dns                  = (known after apply)
      + private_ip                   = (known after apply)
      + public_dns                   = (known after apply)
      + public_ip                    = (known after apply)
      + secondary_private_ips        = (known after apply)
      + security_groups              = (known after apply)
      + source_dest_check            = true
      + subnet_id                    = (known after apply)
      + tenancy                      = (known after apply)
      + volume_tags                  = (known after apply)
      + vpc_security_group_ids       = (known after apply)

      + ebs_block_device {
```

```
          + delete_on_termination = (known after apply)
          + device_name           = (known after apply)
          + encrypted             = (known after apply)
          + iops                  = (known after apply)
          + kms_key_id            = (known after apply)
          + volume_id             = (known after apply)
          + volume_size           = (known after apply)
          + volume_type           = (known after apply)
        }
    }

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_instance.web: Creating...
aws_instance.web: Still creating... [10s elapsed]
aws_instance.web: Still creating... [20s elapsed]
aws_instance.web: Still creating... [30s elapsed]
aws_instance.web: Still creating... [40s elapsed]
aws_instance.web: Still creating... [50s elapsed]
aws_instance.web: Creation complete after 57s [id=i-0e6442ce8db395363]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

Now, in aws our instance has been launched:-





Now, we will terminate our instance with terraform destroy command:-

```
C:\Users\HP\Documents\terra>terraform destroy
aws_instance.web: Refreshing state... [id=i-0e6442ce8db395363]

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # aws_instance.web will be destroyed
  - resource "aws_instance" "web" {
      - ami                          = "ami-052c08d70def0ac62" -> null
      - arn                          = "arn:aws:ec2:ap-south-1:240424121088:instance/i-0e6442ce8db395363" -> null
      - associate_public_ip_address  = true -> null
      - availability_zone            = "ap-south-1a" -> null
      - cpu_core_count               = 1 -> null
      - cpu_threads_per_core         = 1 -> null
      - disable_api_termination      = false -> null
      - ebs_optimized                = false -> null
      - get_password_data            = false -> null
      - hibernation                  = false -> null
      - id                           = "i-0e6442ce8db395363" -> null
      - instance_state               = "running" -> null
      - instance_type                = "t2.micro" -> null
      - ipv6_address_count           = 0 -> null
      - ipv6_addresses               = [] -> null
      - monitoring                   = false -> null
      - primary_network_interface_id = "eni-00e201636d002665f" -> null
      - private_dns                  = "ip-172-31-32-101.ap-south-1.compute.internal" -> null
```

```
        }

      - metadata_options {
          - http_endpoint               = "enabled" -> null
          - http_put_response_hop_limit = 1 -> null
          - http_tokens                 = "optional" -> null
        }

      - root_block_device {
          - delete_on_termination = true -> null
          - device_name           = "/dev/sda1" -> null
          - encrypted             = false -> null
          - iops                  = 100 -> null
          - volume_id             = "vol-048d76718afcd4139" -> null
          - volume_size           = 10 -> null
          - volume_type           = "gp2" -> null
        }
    }

Plan: 0 to add, 0 to change, 1 to destroy.

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_instance.web: Destroying... [id=i-0e6442ce8db395363]
aws_instance.web: Still destroying... [id=i-0e6442ce8db395363, 10s elapsed]
aws_instance.web: Still destroying... [id=i-0e6442ce8db395363, 20s elapsed]
aws_instance.web: Still destroying... [id=i-0e6442ce8db395363, 30s elapsed]
aws_instance.web: Still destroying... [id=i-0e6442ce8db395363, 40s elapsed]
aws_instance.web: Destruction complete after 42s

Destroy complete! Resources: 1 destroyed.
```

You can see that instance has been terminated:-