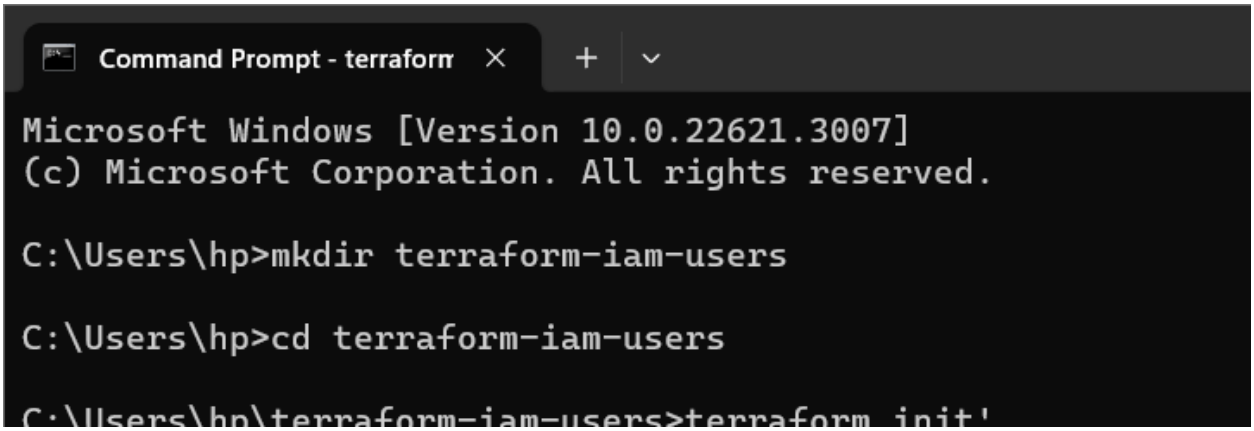
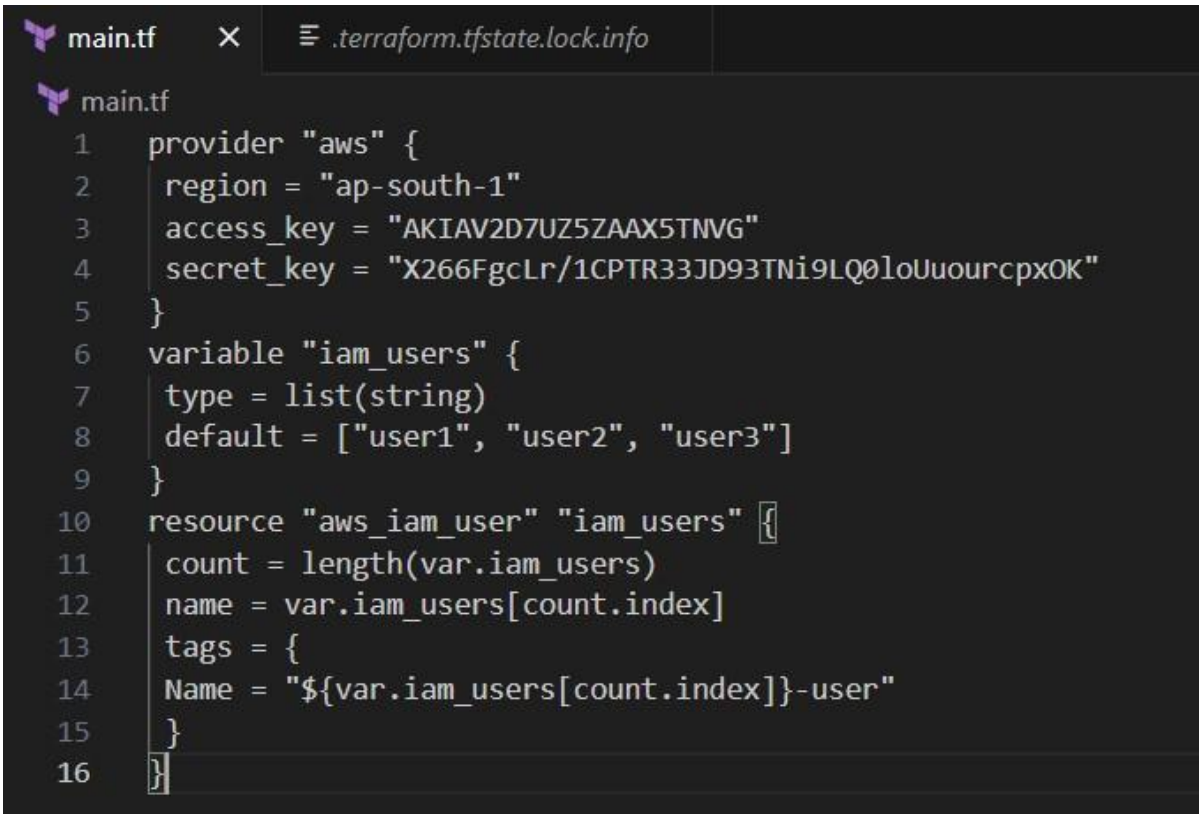
Lab-7

Creating Multiple IAM users in Terraform

Step 1: Create a Terraform Directory



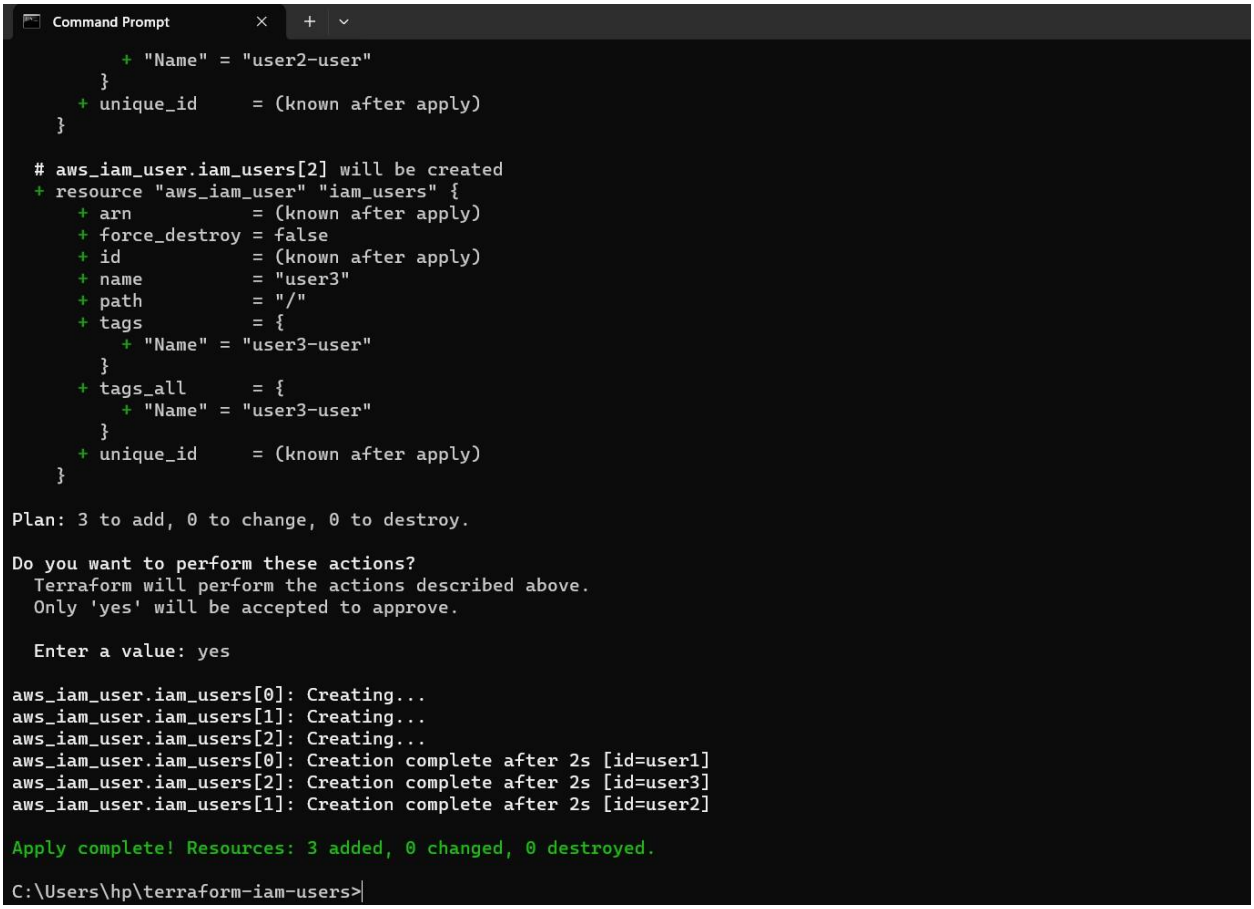
Step 2: Create a file name main.tf



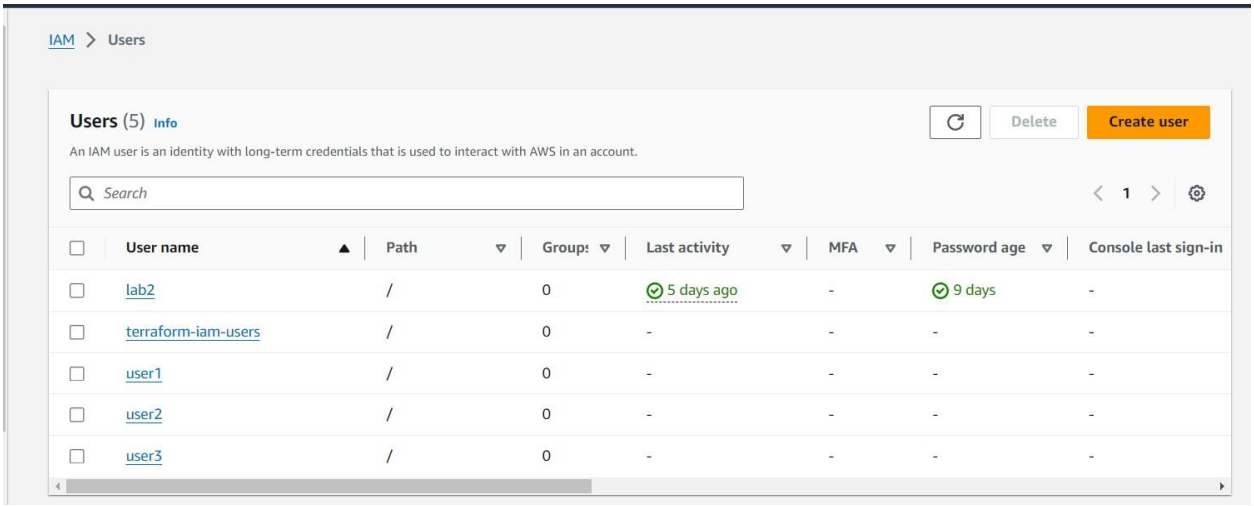
Step 3: Run the following commands in terraform

```
C:\Users\hp\terraform-iam-users>terraform init'
Terraform has no command named "init'". Did you mean "init"?

To see all of Terraform's top-level commands, run:
  terraform -help


C:\Users\hp\terraform-iam-users>terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.36.0...
- Installed hashicorp/aws v5.36.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the providerselections 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.
```

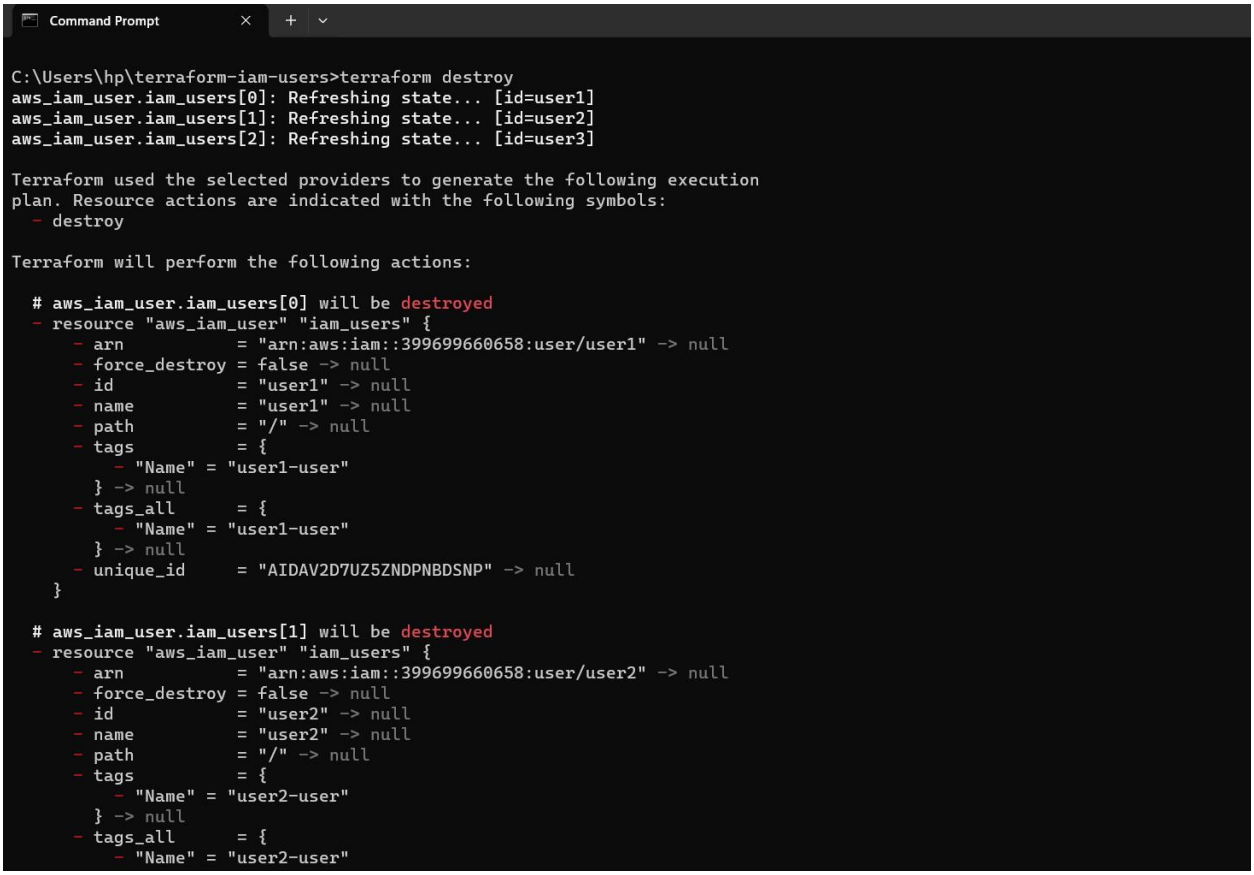
```
Command Prompt

C:\Users\hp\terraform-iam-users>terraform apply

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_iam_user.iam_users[0] will be created
  + resource "aws_iam_user" "iam_users" {
      + arn           = (known after apply)
      + force_destroy = false
      + id            = (known after apply)
      + name          = "user1"
      + path          = "/"
      + tags          = {
          + "Name" = "user1-user"
        }
      + tags_all      = {
          + "Name" = "user1-user"
        }
      + unique_id     = (known after apply)
    }

  # aws_iam_user.iam_users[1] will be created
  + resource "aws_iam_user" "iam_users" {
      + arn           = (known after apply)
      + force_destroy = false
      + id            = (known after apply)
      + name          = "user2"
      + path          = "/"
      + tags          = {
          + "Name" = "user2-user"
        }
      + tags_all      = {
          + "Name" = "user2-user"
        }
      + unique_id     = (known after apply)
    }
```



Step 4: Verify users in Aws Console



Step 5: Cleanup



```
Command Prompt
      - tags_all      = {
          - "Name" = "user2-user"
        } -> null
      - unique_id     = "AIDAV2D7UZ5ZLVG7SJAVJ" -> null
    }

  # aws_iam_user.iam_users[2] will be destroyed
  - resource "aws_iam_user" "iam_users" {
      - arn           = "arn:aws:iam::399699660658:user/user3" -> null
      - force_destroy = false -> null
      - id            = "user3" -> null
      - name          = "user3" -> null
      - path          = "/" -> null
      - tags          = {
          - "Name" = "user3-user"
        } -> null
      - tags_all      = {
          - "Name" = "user3-user"
        } -> null
      - unique_id     = "AIDAV2D7UZ5ZERXOG5HIZ" -> null
    }

Plan: 0 to add, 0 to change, 3 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_iam_user.iam_users[2]: Destroying... [id=user3]
aws_iam_user.iam_users[1]: Destroying... [id=user2]
aws_iam_user.iam_users[0]: Destroying... [id=user1]
aws_iam_user.iam_users[2]: Destruction complete after 1s
aws_iam_user.iam_users[1]: Destruction complete after 1s
aws_iam_user.iam_users[0]: Destruction complete after 1s

Destroy complete! Resources: 3 destroyed.

C:\Users\hp\terraform-iam-users>
```