Lab Exercise 7– Creating Multiple IAM Users in Terraform

Objective:

Learn how to use Terraform to create multiple IAM users with unique settings.

Prerequisites:

- Terraform installed on your machine.
- AWS CLI configured with the necessary credentials.

Steps:

1. Create a Terraform Directory:

```
mkdir terraform-iam-users
cd terraform-iam-users
```

- Create Terraform Configuration Files:
- Create a file named main.tf:

# main.tf

```
provider "aws" {
  region = "us-east-1"
}

variable "iam_users" {
  type    = list(string)
  default = ["user1", "user2", "user3"]
}
```

```
resource "aws_iam_user" "iam_users" {
  count = length(var.iam_users)
  name = var.iam_users[count.index]

  tags = {
    Name = "${var.iam_users[count.index]}-user"
  }
}
```

In this configuration, we define a list variable iam_users containing the names of the IAM users we want to create. The aws_iam_user resource is then used in a loop to create users based on the values in the list.

2. Initialize and Apply:

Run the following Terraform commands to initialize and apply the configuration:

```
terraform init
terraform apply
```

Terraform will prompt you to confirm the creation of IAM users. Type yes and press Enter.

3. Verify Users in AWS Console:

- Log in to the AWS Management Console and navigate to the IAM service.
- Verify that the IAM users with the specified names and tags have been created.

4. Update IAM Users:

- If you want to add or remove IAM users, modify the iam_users list in the main.tf file.
- Rerun the terraform apply command to apply the changes:

```
terraform apply
```

5. Clean Up:

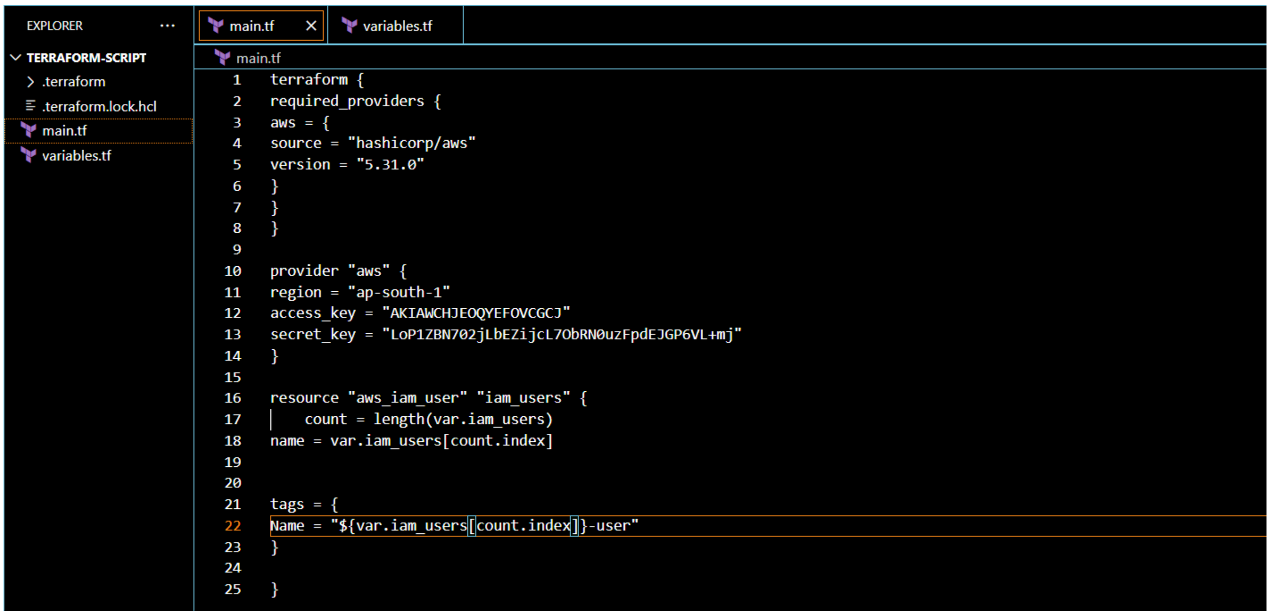
- After testing, you can clean up the IAM users:

```
terraform destroy
```

- Confirm the destruction by typing yes.

6. Conclusion:

This lab exercise demonstrates how to create multiple IAM users in AWS using Terraform. The use of variables and loops allows you to easily manage and scale the creation of IAM users. Experiment with different user names and settings in the main.tf file to understand how Terraform provisions resources based on your configuration.

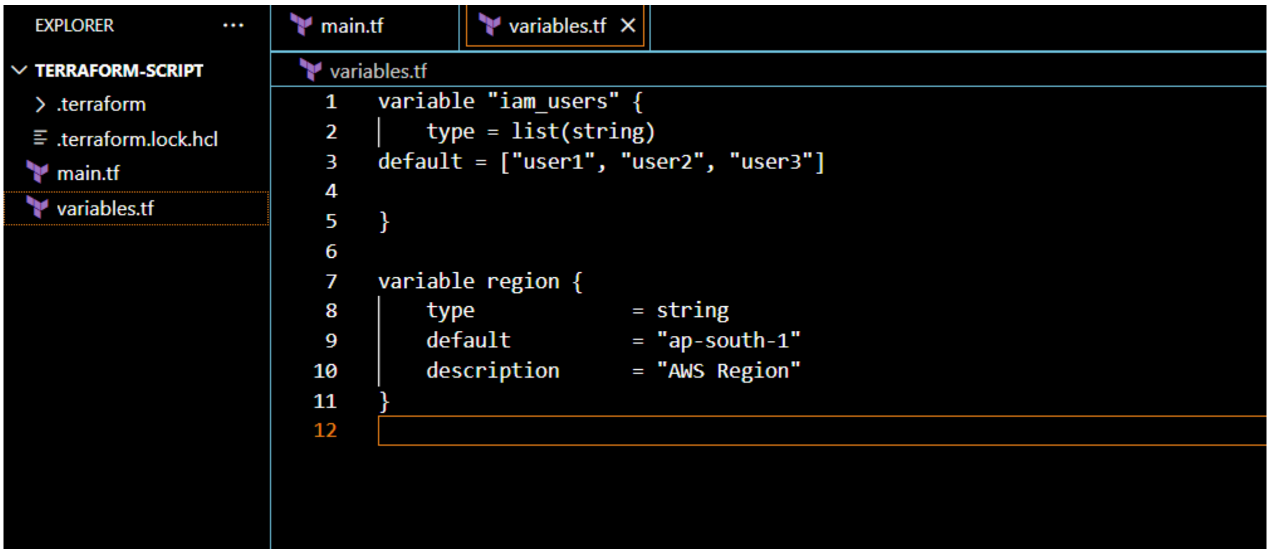


EXPLORER ···
TERRAFORM-SCRIPT
> .terraform
.terraform.lock.hcl
main.tf
variables.tf

main.tf × | variables.tf

main.tf

```
terraform {
required_providers {
aws = {
source = "hashicorp/aws"
version = "5.31.0"
}
}
}

provider "aws" {
region = "ap-south-1"
access_key = "AKIAWCHJEOQYEFOVCGCJ"
secret_key = "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"
}

resource "aws_iam_user" "iam_users" {
    count = length(var.iam_users)
name = var.iam_users[count.index]


tags = {
Name = "${var.iam_users[count.index]}-user"
}

}
```

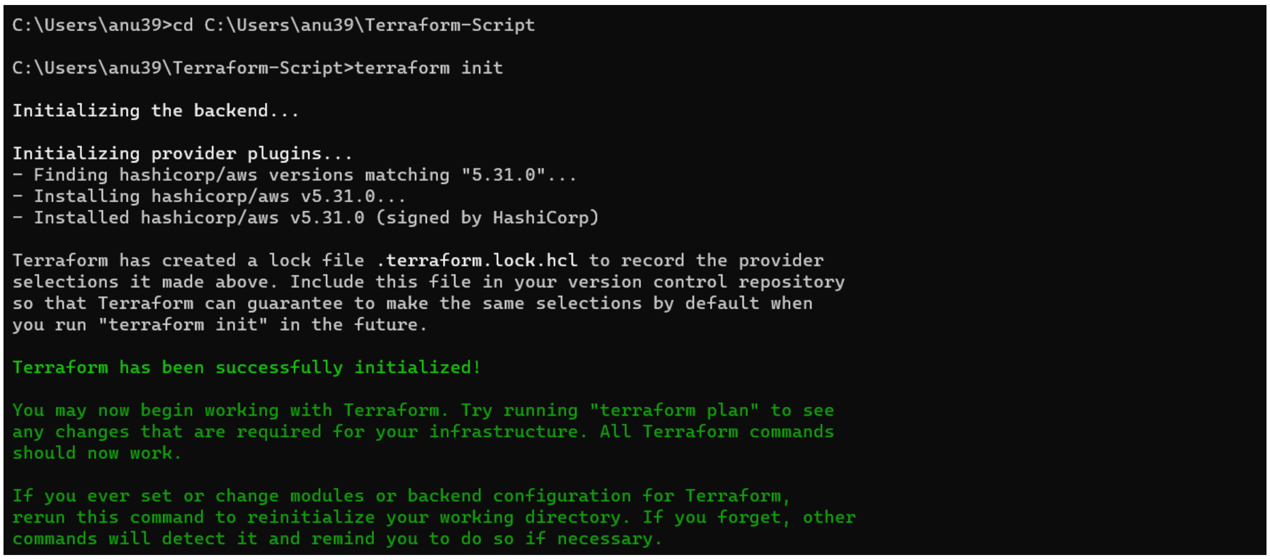


EXPLORER ···
TERRAFORM-SCRIPT
> .terraform
.terraform.lock.hcl
main.tf
variables.tf

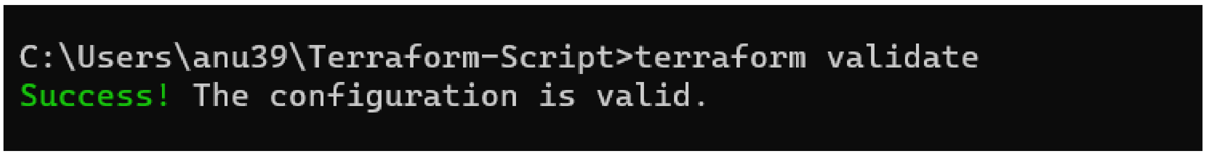
main.tf | variables.tf ×

variables.tf

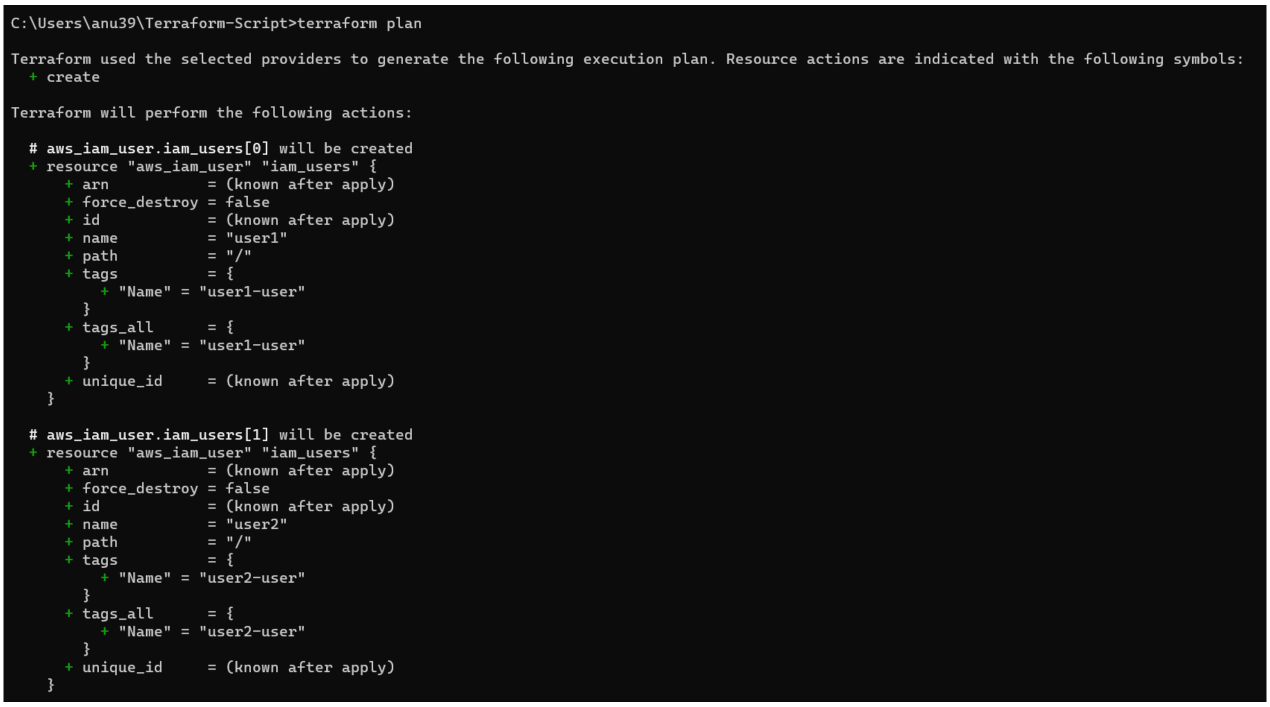
```
variable "iam_users" {
    type = list(string)
default = ["user1", "user2", "user3"]

}

variable "region" {
    type            = string
    default         = "ap-south-1"
    description     = "AWS Region"
}

```



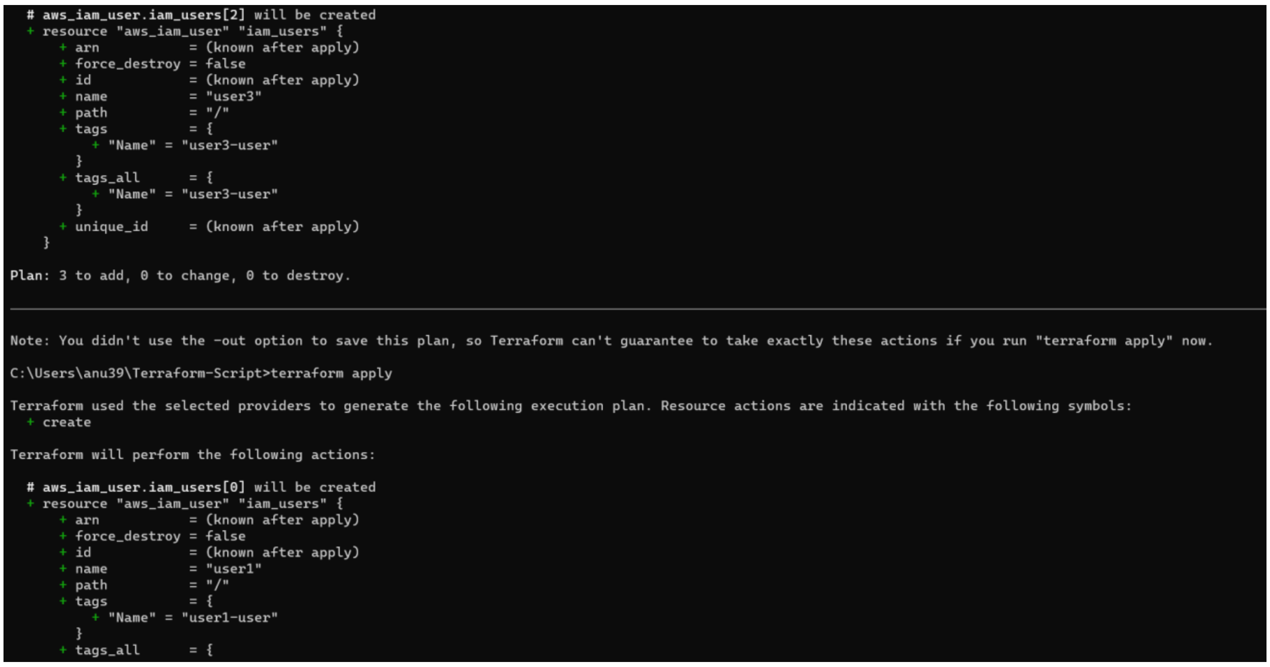
```
C:\Users\anu39>cd C:\Users\anu39\Terraform-Script

C:\Users\anu39\Terraform-Script>terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.31.0"...
- Installing hashicorp/aws v5.31.0...
- Installed hashicorp/aws v5.31.0 (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.
```



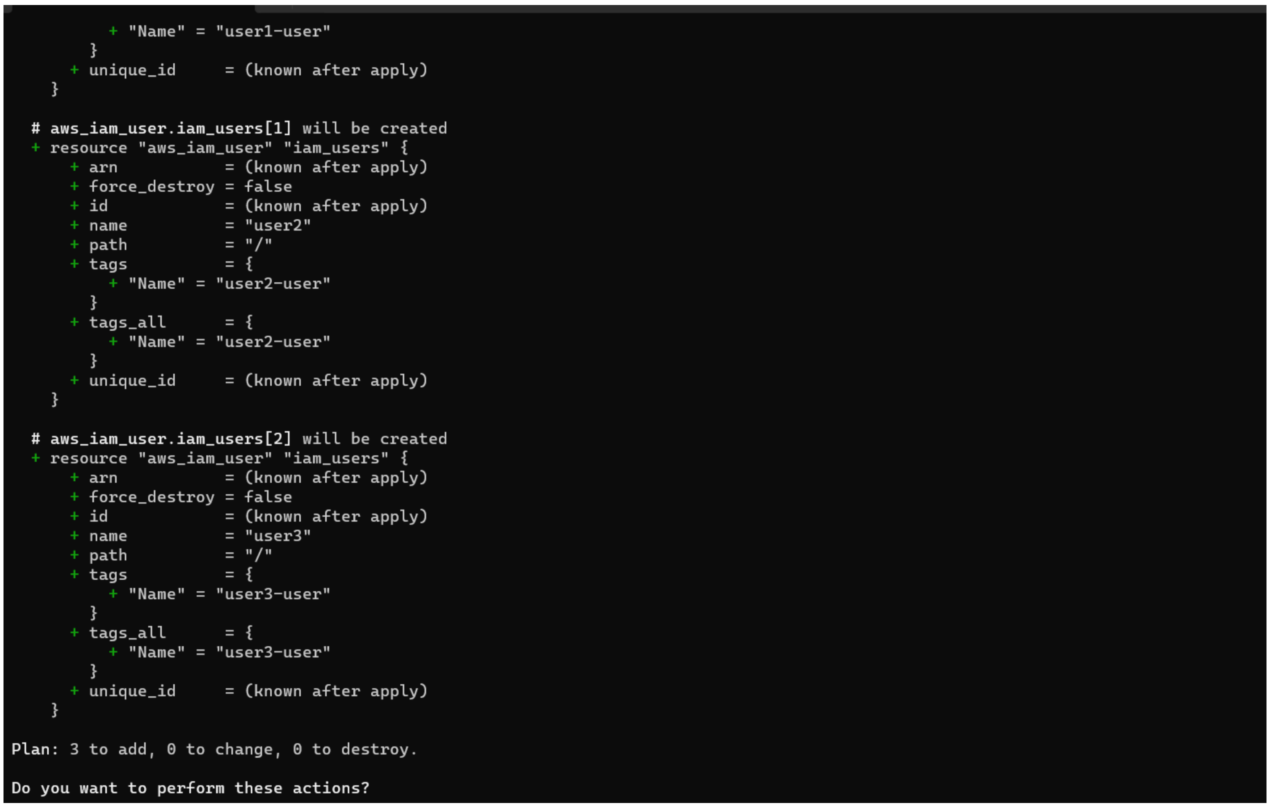
```
C:\Users\anu39\Terraform-Script>terraform validate
Success! The configuration is valid.
```



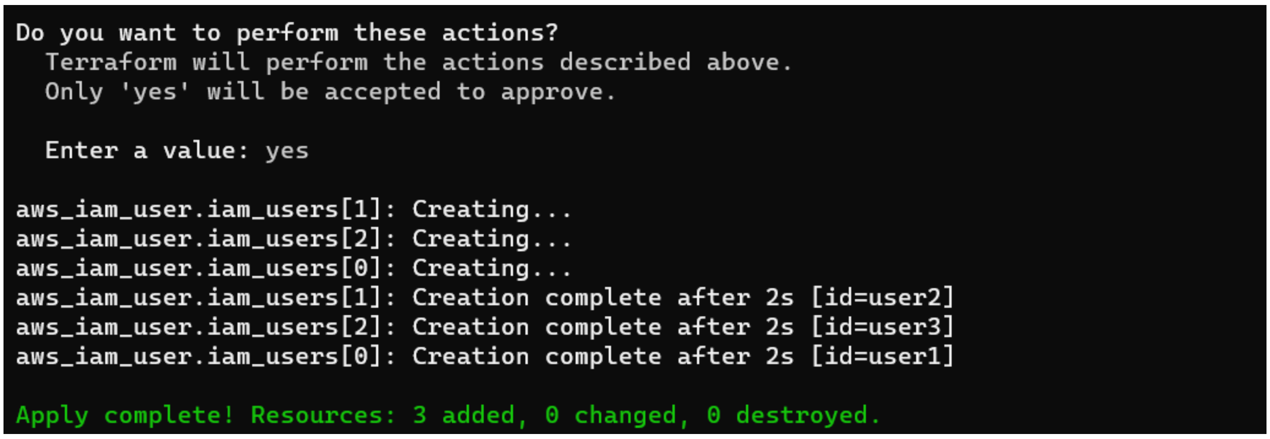
```
C:\Users\anu39\Terraform-Script>terraform plan

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)
    }
```



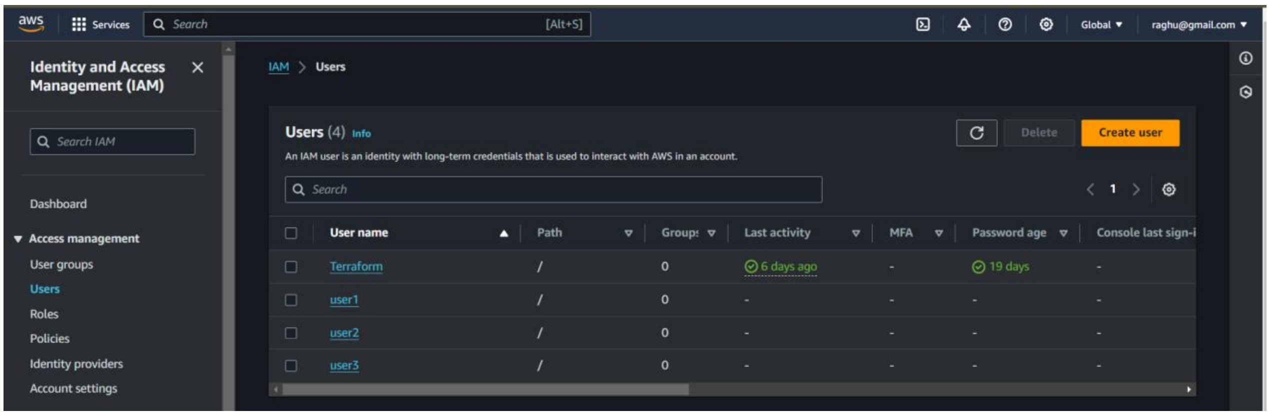
```
  # aws_iam_user.iam_users[2] will be created
  + resource "aws_iam_user" "iam_users" {
      + arn           = (known after apply)
      + force_destroy = false
      + id            = (known after apply)
      + name          = "user3"
      + path          = "/"
      + tags          = {
          + "Name" = "user3-user"
        }
      + tags_all      = {
          + "Name" = "user3-user"
        }
      + unique_id     = (known after apply)
    }

Plan: 3 to add, 0 to change, 0 to destroy.

─────────────────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.

C:\Users\anu39\Terraform-Script>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)
    }

  # aws_iam_user.iam_users[2] will be created
  + resource "aws_iam_user" "iam_users" {
      + arn           = (known after apply)
      + force_destroy = false
      + id            = (known after apply)
      + name          = "user3"
      + path          = "/"
      + tags          = {
          + "Name" = "user3-user"
        }
      + tags_all      = {
          + "Name" = "user3-user"
        }
      + unique_id     = (known after apply)
    }

Plan: 3 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
```



```
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_iam_user.iam_users[1]: Creating...
aws_iam_user.iam_users[2]: Creating...
aws_iam_user.iam_users[0]: Creating...
aws_iam_user.iam_users[1]: Creation complete after 2s [id=user2]
aws_iam_user.iam_users[2]: Creation complete after 2s [id=user3]
aws_iam_user.iam_users[0]: Creation complete after 2s [id=user1]

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.
```

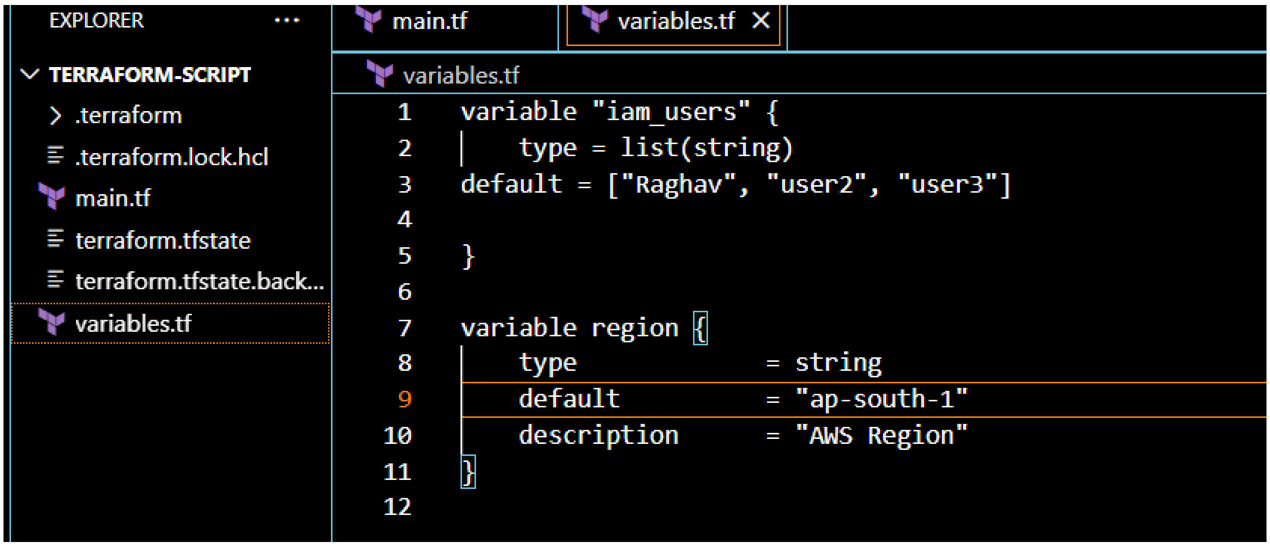


Users (4)

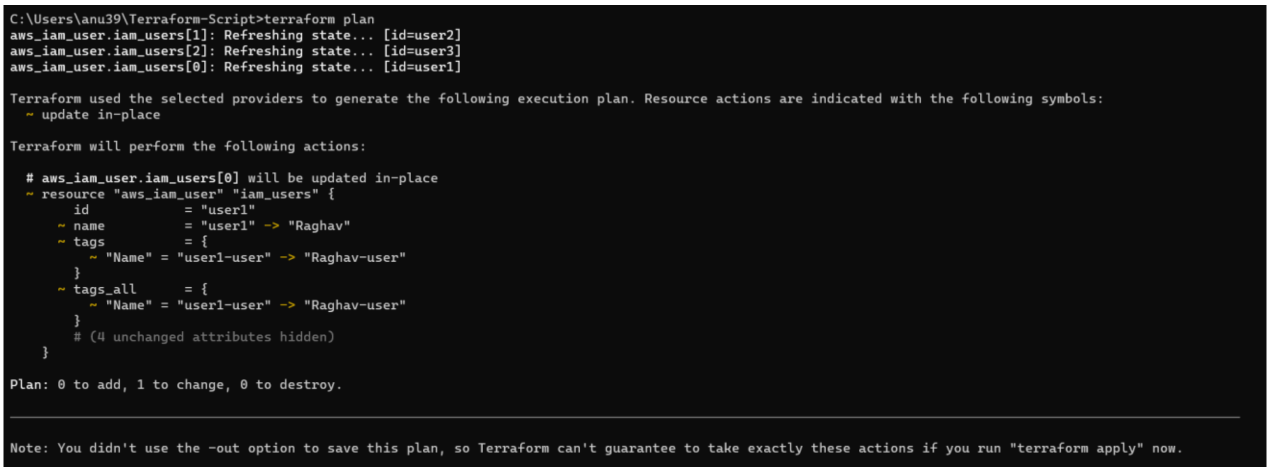
An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

User name	Path	Group:	Last activity	MFA	Password age	Console last sign-i
Terraform	/	0	✓ 6 days ago	-	✓ 19 days	-
user1	/	0	-	-	-	-
user2	/	0	-	-	-	-
user3	/	0	-	-	-	-

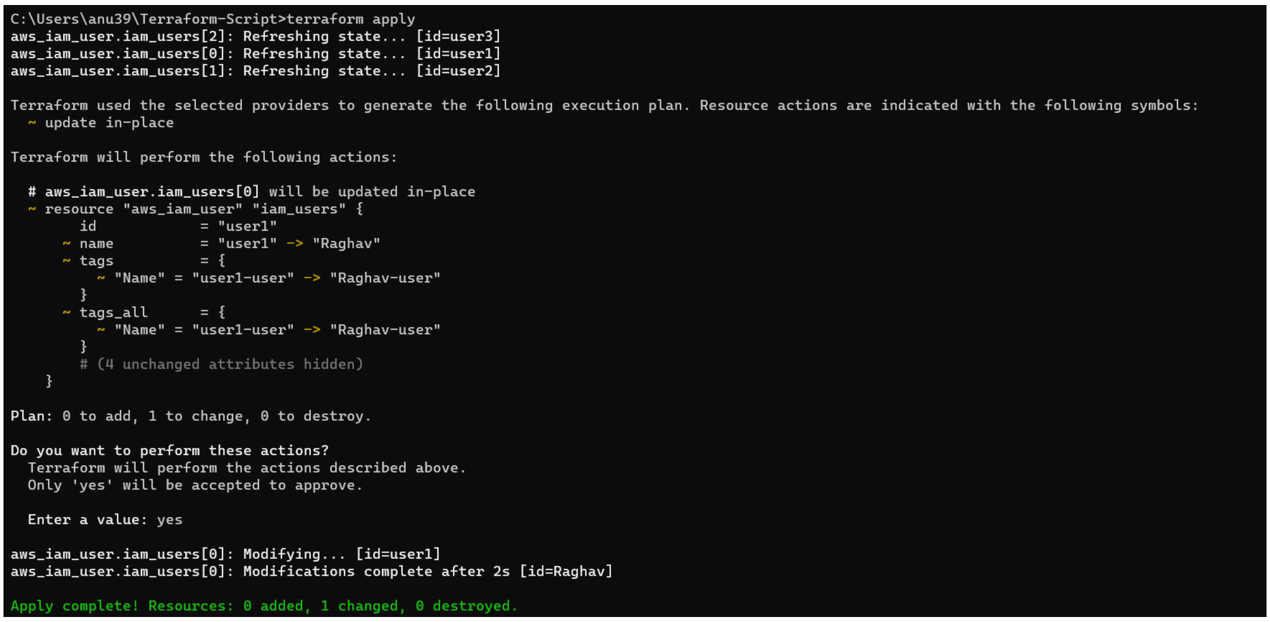
Update the list of users to update the count of IAM Users on AWS: -



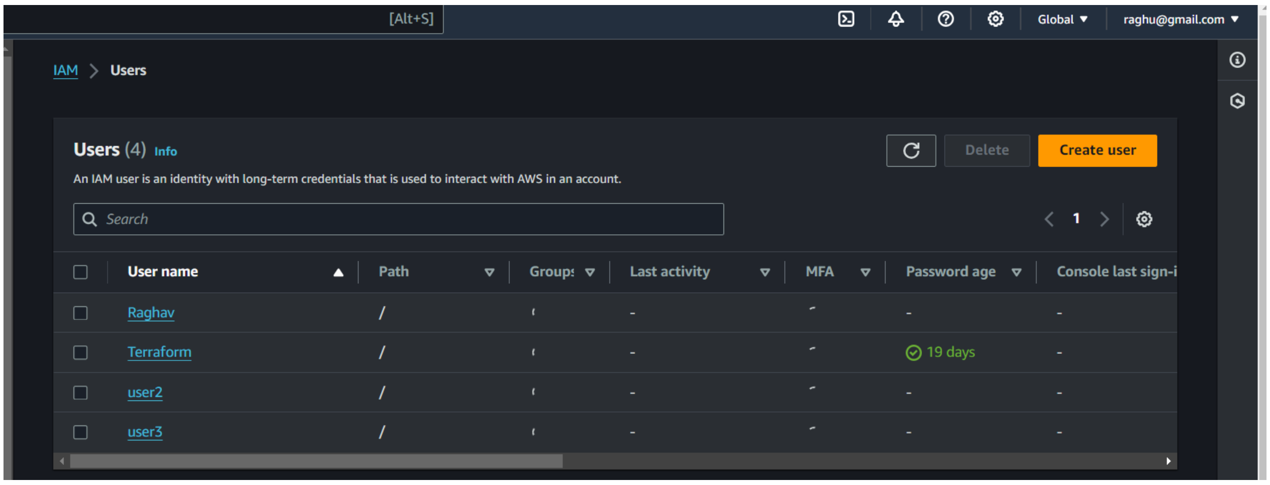
```
variable "iam_users" {
    type = list(string)
default = ["Raghav", "user2", "user3"]

}

variable region {
    type            = string
    default         = "ap-south-1"
    description     = "AWS Region"
}

```



```
C:\Users\anu39\Terraform-Script>terraform plan
aws_iam_user.iam_users[1]: Refreshing state... [id=user2]
aws_iam_user.iam_users[2]: Refreshing state... [id=user3]
aws_iam_user.iam_users[0]: Refreshing state... [id=user1]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  ~ update in-place

Terraform will perform the following actions:

  # aws_iam_user.iam_users[0] will be updated in-place
  ~ resource "aws_iam_user" "iam_users" {
        id            = "user1"
      ~ name          = "user1" -> "Raghav"
      ~ tags          = {
          ~ "Name" = "user1-user" -> "Raghav-user"
        }
      ~ tags_all      = {
          ~ "Name" = "user1-user" -> "Raghav-user"
        }
        # (4 unchanged attributes hidden)
    }

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

─────────────────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
```



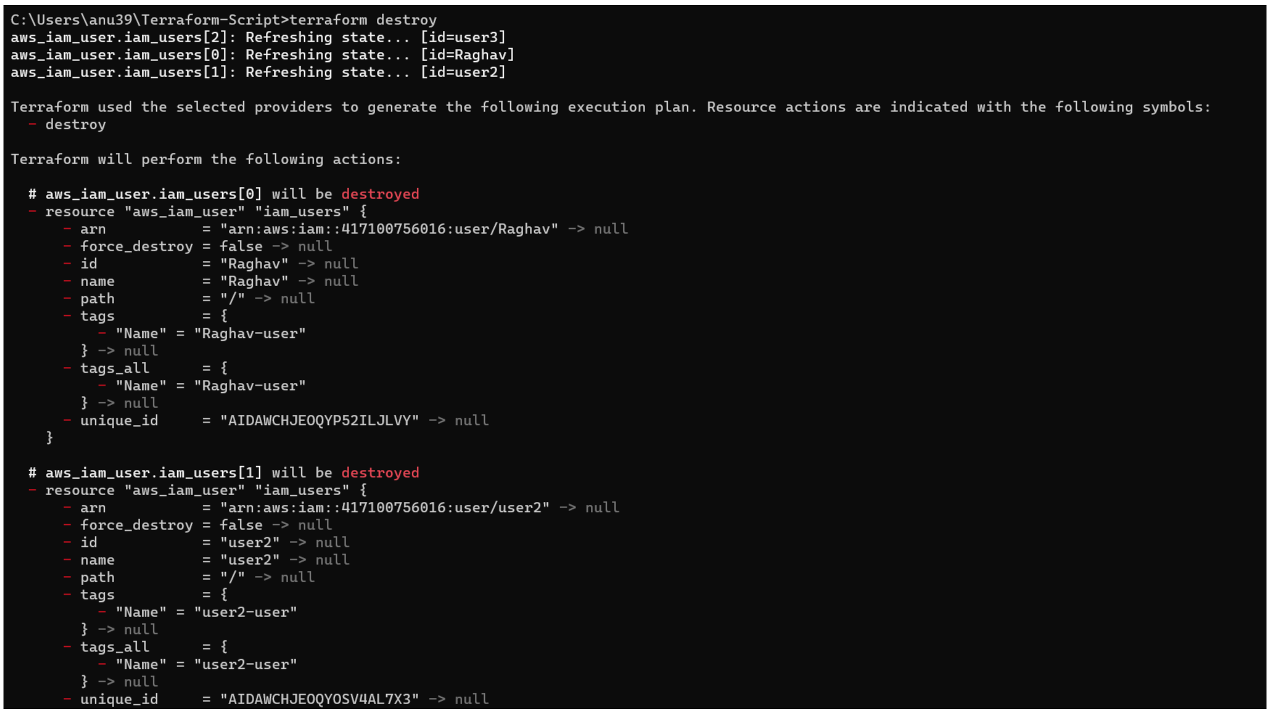
```
C:\Users\anu39\Terraform-Script>terraform apply
aws_iam_user.iam_users[2]: Refreshing state... [id=user3]
aws_iam_user.iam_users[0]: Refreshing state... [id=user1]
aws_iam_user.iam_users[1]: Refreshing state... [id=user2]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  ~ update in-place

Terraform will perform the following actions:

  # aws_iam_user.iam_users[0] will be updated in-place
  ~ resource "aws_iam_user" "iam_users" {
        id            = "user1"
      ~ name          = "user1" -> "Raghav"
      ~ tags          = {
          ~ "Name" = "user1-user" -> "Raghav-user"
        }
      ~ tags_all      = {
          ~ "Name" = "user1-user" -> "Raghav-user"
        }
        # (4 unchanged attributes hidden)
    }

Plan: 0 to add, 1 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_iam_user.iam_users[0]: Modifying... [id=user1]
aws_iam_user.iam_users[0]: Modifications complete after 2s [id=Raghav]

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



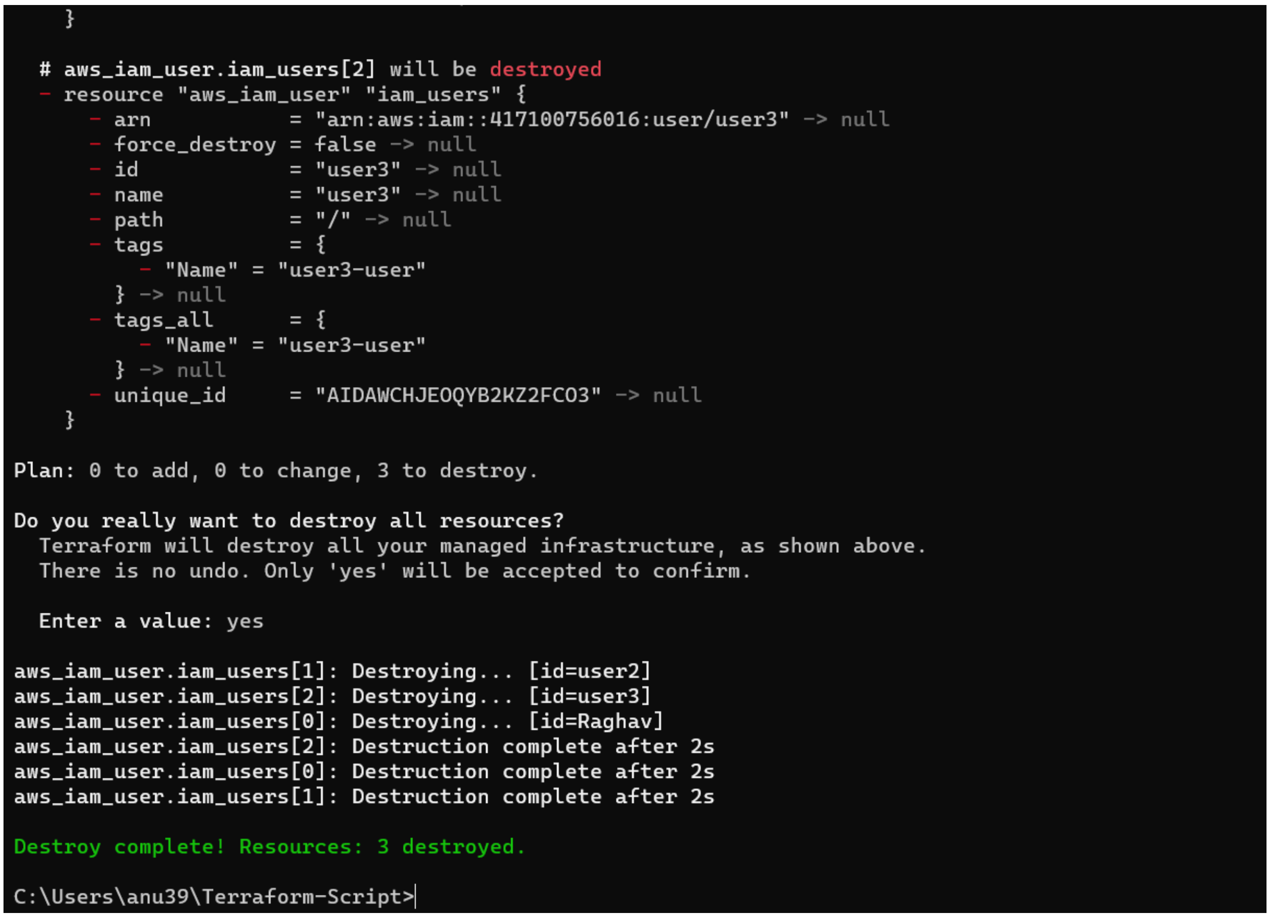
Users (4) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

	User name	Path	Group:	Last activity	MFA	Password age	Console last sign-i
☐	Raghav	/	(	-	-	-	-
☐	Terraform	/	(	-	-	✓ 19 days	-
☐	user2	/	(	-	-	-	-
☐	user3	/	(	-	-	-	-



```
C:\Users\anu39\Terraform-Script>terraform destroy
aws_iam_user.iam_users[2]: Refreshing state... [id=user3]
aws_iam_user.iam_users[0]: Refreshing state... [id=Raghav]
aws_iam_user.iam_users[1]: Refreshing state... [id=user2]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # aws_iam_user.iam_users[0] will be destroyed
  - resource "aws_iam_user" "iam_users" {
      - arn           = "arn:aws:iam::417100756016:user/Raghav" -> null
      - force_destroy = false -> null
      - id            = "Raghav" -> null
      - name          = "Raghav" -> null
      - path          = "/" -> null
      - tags          = {
          - "Name" = "Raghav-user"
        } -> null
      - tags_all      = {
          - "Name" = "Raghav-user"
        } -> null
      - unique_id     = "AIDAWCHJEOQYP52ILJLVY" -> null
    }

  # aws_iam_user.iam_users[1] will be destroyed
  - resource "aws_iam_user" "iam_users" {
      - arn           = "arn:aws:iam::417100756016:user/user2" -> null
      - force_destroy = false -> null
      - id            = "user2" -> null
      - name          = "user2" -> null
      - path          = "/" -> null
      - tags          = {
          - "Name" = "user2-user"
        } -> null
      - tags_all      = {
          - "Name" = "user2-user"
        } -> null
      - unique_id     = "AIDAWCHJEOQYOSV4AL7X3" -> null
```


```
    }

  # aws_iam_user.iam_users[2] will be destroyed
  - resource "aws_iam_user" "iam_users" {
      - arn           = "arn:aws:iam::417100756016: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     = "AIDAWCHJEOQYB2KZ2FCO3" -> 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[1]: Destroying... [id=user2]
aws_iam_user.iam_users[2]: Destroying... [id=user3]
aws_iam_user.iam_users[0]: Destroying... [id=Raghav]
aws_iam_user.iam_users[2]: Destruction complete after 2s
aws_iam_user.iam_users[0]: Destruction complete after 2s
aws_iam_user.iam_users[1]: Destruction complete after 2s

Destroy complete! Resources: 3 destroyed.

C:\Users\anu39\Terraform-Script>
```