LAB-7

Creating Multiple IAM Users in Terraform

Step 1: Create a new Terraform IAM Users directory

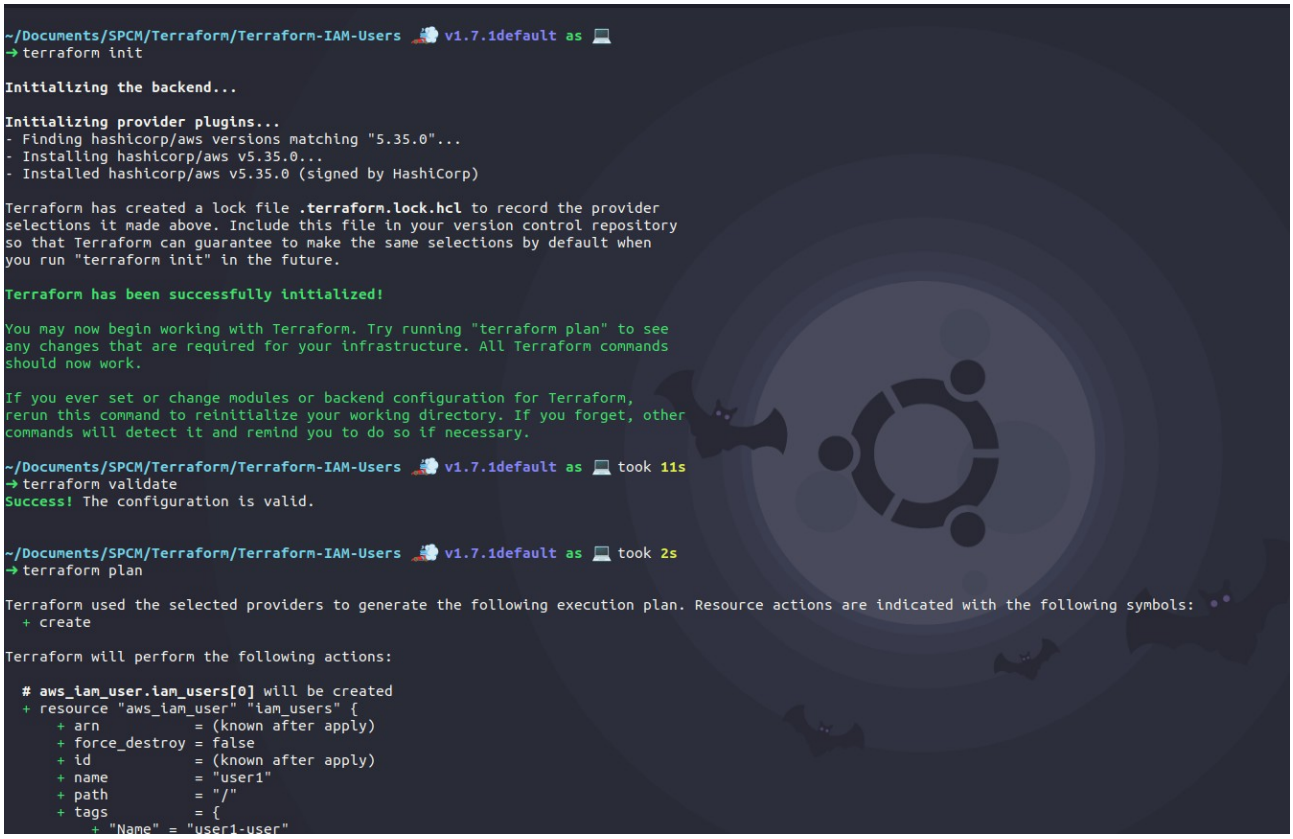
```
~/Documents/SPCM/Terraform  v1.7.1default as
➜ mkdir Terraform-IAM-Users

~/Documents/SPCM/Terraform  v1.7.1default as
➜ cd Terraform-IAM-Users

~/Documents/SPCM/Terraform/Terraform-IAM-Users as
➜
```

Step 2: Create a main.tf file

main.tf

```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.35.0"
    }
  }
}

provider "aws" {
  region = "ap-south-1"
  access_key = [redacted]
  secret_key = [redacted]
}

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"
    }
}
```

Step 3: Initialize and plan

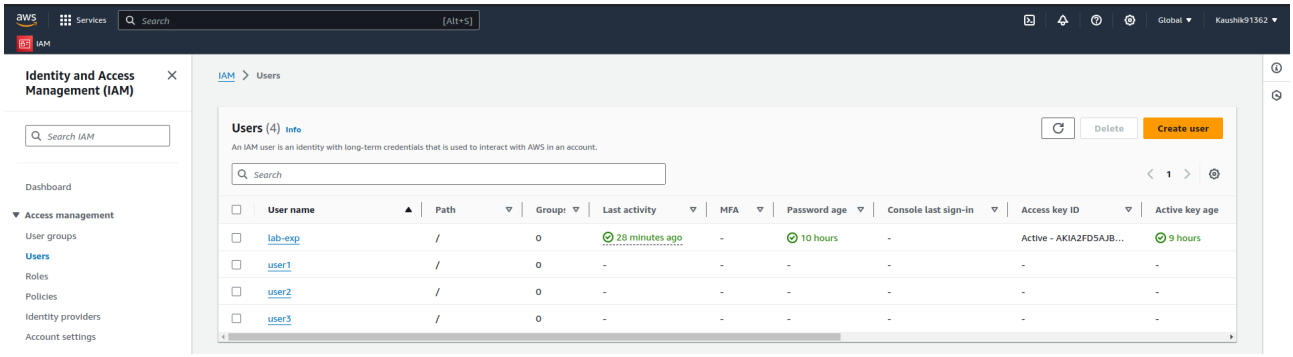


```
~/Documents/SPCM/Terraform/Terraform-IAM-Users v1.7.1default as
➜ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.35.0"...
- Installing hashicorp/aws v5.35.0...
- Installed hashicorp/aws v5.35.0 (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.

~/Documents/SPCM/Terraform/Terraform-IAM-Users v1.7.1default as took 11s
➜ terraform validate
Success! The configuration is valid.


~/Documents/SPCM/Terraform/Terraform-IAM-Users v1.7.1default as took 2s
➜ 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"
```

Step 4: Apply

```
~/Documents/SPCM/Terraform/Terraform-IAM-Users v1.7.1default as took 2s
➜ 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          = "/"
```

Step 5: Verify Users in AWS Console



Step 6: Add or remove IAM user

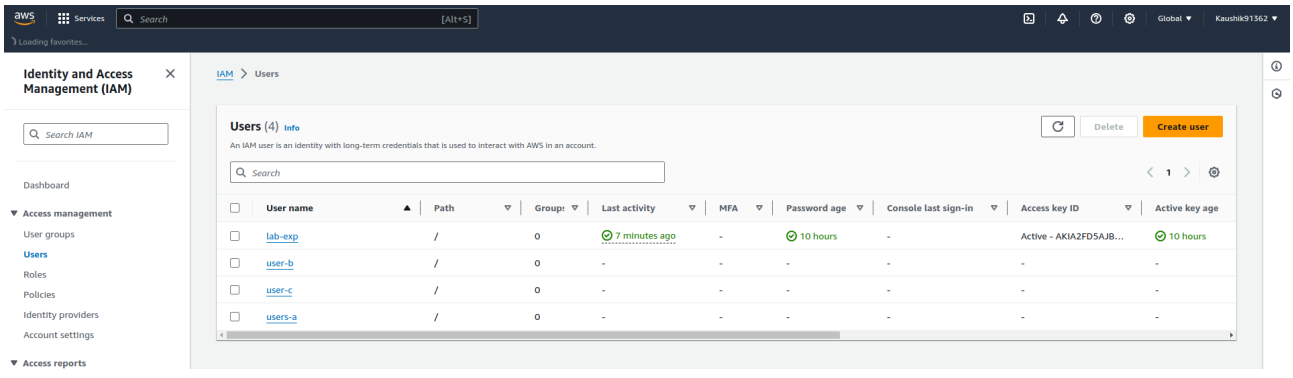
Modify the main.tf file to add or remove users the rerun terraform apply command to apply changes

main.tf

```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.35.0"
    }
  }
}

provider "aws" {
  region = "ap-south-1"
  access_key = "AKIA2FD5AJBR2NUSDMTH"
  secret_key = "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"
}

variable "iam_users"{
    type =list(string)
    default= ["user a","user b","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"
    }
}
```

```
~/Documents/SPCM/Terraform/Terraform-IAM-Users  v1.7.1default as 💻
→ terraform apply
aws_iam_user.iam_users[2]: Refreshing state... [id=user3]
aws_iam_user.iam_users[1]: Refreshing state... [id=user2]
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" -> "users-a"
      ~ tags          = {
          ~ "Name" = "user1-user" -> "users-a-user"
        }
      ~ tags_all      = {
          ~ "Name" = "user1-user" -> "users-a-user"
        }
```

```
Plan: 0 to add, 3 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[2]: Modifying... [id=user3]
aws_iam_user.iam_users[1]: Modifying... [id=user2]
aws_iam_user.iam_users[2]: Modifications complete after 1s [id=user-c]
aws_iam_user.iam_users[0]: Modifications complete after 1s [id=users-a]
aws_iam_user.iam_users[1]: Modifications complete after 1s [id=user-b]

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



Step 7: Clean Up

```
~/Documents/SPCM/Terraform/Terraform-IAM-Users  v1.7.1default as 💻 took 8s
➜ terraform destroy
aws_iam_user.iam_users[0]: Refreshing state... [id=users-a]
aws_iam_user.iam_users[2]: Refreshing state... [id=user-c]
aws_iam_user.iam_users[1]: Refreshing state... [id=user-b]

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::698194348131:user/users-a" -> null
      - force_destroy = false -> null
      - id            = "users-a" -> null
      - name          = "users-a" -> null
      - path          = "/" -> null
      - tags          = {
          - "Name" = "users-a-user"
        } -> null
      - tags_all      = {
          - "Name" = "users-a-user"
        } -> null
      - unique_id     = "AIDA2FD5AJBRT73ZBEZ5P" -> null
    }

  # aws_iam_user.iam_users[1] will be destroyed
  - resource "aws_iam_user" "iam_users" {
      - arn           = "arn:aws:iam::698194348131:user/user-b" -> null
      - force_destroy = false -> null
      - id            = "user-b" -> null
      - name          = "user-b" -> null
      - path          = "/" -> null
      - tags          = {
          - "Name" = "user-b-user"
        } -> null
      - tags_all      = {
```