



## **Lab 13**

**Lab title: Terraform IAM Management with AWS**

**Submitted to: Engr. Mohammad Shoaib**

**Submitted by: Anara Hayat**

**Reg#No: 2023-BSE-008**

## Task 0 Lab Setup (Codespace & GH CLI)

All actions below should be executed inside the Codespace shell.

Create Codespace & connect:

```
~\OneDrive\Desktop\CC Labs\Lab13 git:(master) (2.93/s)
gh repo create Lab13 --public
✓ Created repository Anara-hayat/Lab13 on github.com
https://github.com/Anara-hayat/Lab13

~\OneDrive\Desktop\CC Labs\Lab13 git:(main) (6.994s)
gh codespace create --repo Anara-hayat/Lab13
✓ Codespaces usage for this repository is paid for by Anara-hayat
? Choose Machine Type: 2 cores, 8 GB RAM, 32 GB storage
friendly-space-potato-wrq7x749q56vhr7r

~\OneDrive\Desktop\CC Labs\Lab13 git:(main) (1.047s)
gh codespace list

```

NAME	DISPLAY NAME	REPOSITORY	BRANCH	STATE	CREATED AT
obscure-space...	obscure spa...	Anara-hayat...	main*	Shutdown	about 28 da...
obscure-happi...	obscure hap...	Anara-hayat...	main*	Shutdown	about 14 da...
orange-robot-...	orange robot	Anara-hayat...	main	Shutdown	about 5 day...
animated-capy...	animated ca...	Anara-hayat...	main	Shutdown	about 2 day...
friendly-spac...	friendly sp...	Anara-hayat...	main	Available	less than a...

```

~\OneDrive\Desktop\CC Labs\Lab13 git:(main)
gh codespace ssh -c friendly-space-potato-wrq7x749q56vhr7r
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-1030-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

@Anara-hayat → /workspaces/Lab13 (main) $
```

## Task 1 — Create IAM Group and Output Details

In this task, you will create an IAM group named "developers" and output its details.

1. Create the initial project structure:

```
@Anara-hayat → /workspaces/Lab13 (main) $ mkdir -p ~/Lab13
@Anara-hayat → /workspaces/Lab13 (main) $ cd ~/Lab13
@Anara-hayat → ~/Lab13 $
```

2. Create the main Terraform file:

```
@Anara-hayat → /workspaces/Lab13 (main) $ touch main.tf
@Anara-hayat → /workspaces/Lab13 (main) $
```

3. Create main.tf with AWS provider configuration:

```
provider "aws" {
  shared_config_files      = ["~/.aws/config"]
  shared_credentials_files = ["~/.aws/credentials"]
}

resource "aws_iam_group" "developers" {
  name = "developers"
  path = "/groups/"
}

output "group_details" {
  value = {
    group_name = aws_iam_group.developers.name
    group_arn  = aws_iam_group.developers.arn
    unique_id  = aws_iam_group.developers.unique_id
  }
}
```

4. Initialize Terraform:

```
@Anara-hayat → /workspaces/Lab13 (main) $ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.27.0...
- Installed hashicorp/aws v6.27.0 (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.

## 5. Apply the configuration:

```
@Anara-hayat → /workspaces/Lab13 (main) $ terraform apply -auto-approve

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_group.developers will be created
+ resource "aws_iam_group" "developers" {
  + arn      = (known after apply)
  + id       = (known after apply)
  + name     = "developers"
  + path     = "/groups/"
  + unique_id = (known after apply)
}

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

Changes to Outputs:
+ group_details = {
  + group_arn = (known after apply)
  + group_name = "developers"
  + unique_id = (known after apply)
}
aws_iam_group.developers: Creating...
aws_iam_group.developers: Creation complete after 1s [id=developers]

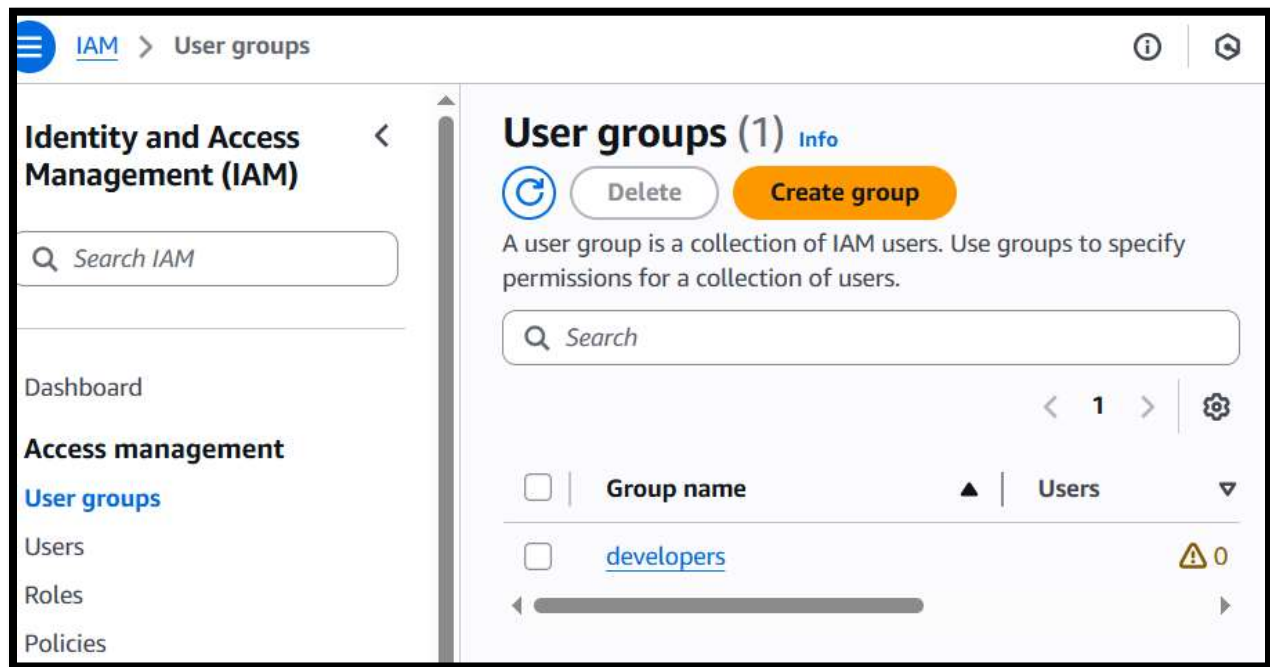
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:
group_details = {
  "group_arn" = "arn:aws:iam::249471344514:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATUFNGT6BISKGTVWD"
}
```

6. Display the output:

```
@Anara-hayat → /workspaces/Lab13 (main) $ terraform output
group_details = {
  "group_arn" = "arn:aws:iam::249471344514:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATUFNGT6BISKGWTVWD"
}
```

7. Verify the group in AWS Console:



## Task 2 — Create IAM User with Group Membership

In this task, you will create an IAM user named "loadbalancer" and add it to the developers group.

1. Update main.tf to add the IAM user resource:

```
provider "aws" {
  shared_config_files      = ["~/.aws/config"]
  shared_credentials_files = ["~/.aws/credentials"]
}

resource "aws_iam_group" "developers" {
  name = "developers"
  path = "/groups/"
}

output "group_details" {
  value = {
    group_name = aws_iam_group.developers.name
    group_arn  = aws_iam_group.developers.arn
    unique_id  = aws_iam_group.developers.unique_id
  }
}

resource "aws_iam_user" "lb" {
  name = "loadbalancer"
  path = "/users/"
  force_destroy = true
  tags = {
    DisplayName = "Load Balancer"
  }
}

resource "aws_iam_user_group_membership" "lb_membership" {
  user = aws_iam_user.lb.name
  groups = [
    aws_iam_group.developers.name
  ]
}

output "user_details" {
  value = {
    user_name = aws_iam_user.lb.name
    user_arn  = aws_iam_user.lb.arn
    unique_id = aws_iam_user.lb.unique_id
  }
}

}:wq
```

## 2. Apply the configuration:

```
aws_iam_user.lb: Creating...
aws_iam_user.lb: Creation complete after 1s [id=loadbalancer]
aws_iam_user_group_membership.lb_membership: Creating...
aws_iam_user_group_membership.lb_membership: Creation complete after 1s [id=terraform-202601022031334287000000001]
```

**Apply complete! Resources: 2 added, 0 changed, 0 destroyed.**

**Outputs:**

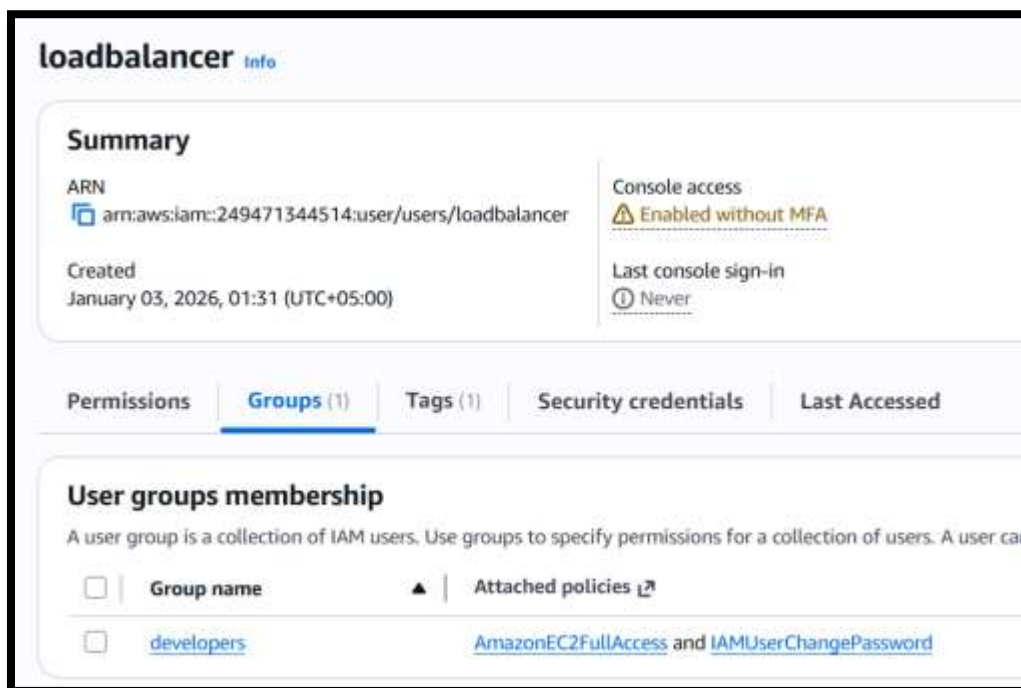
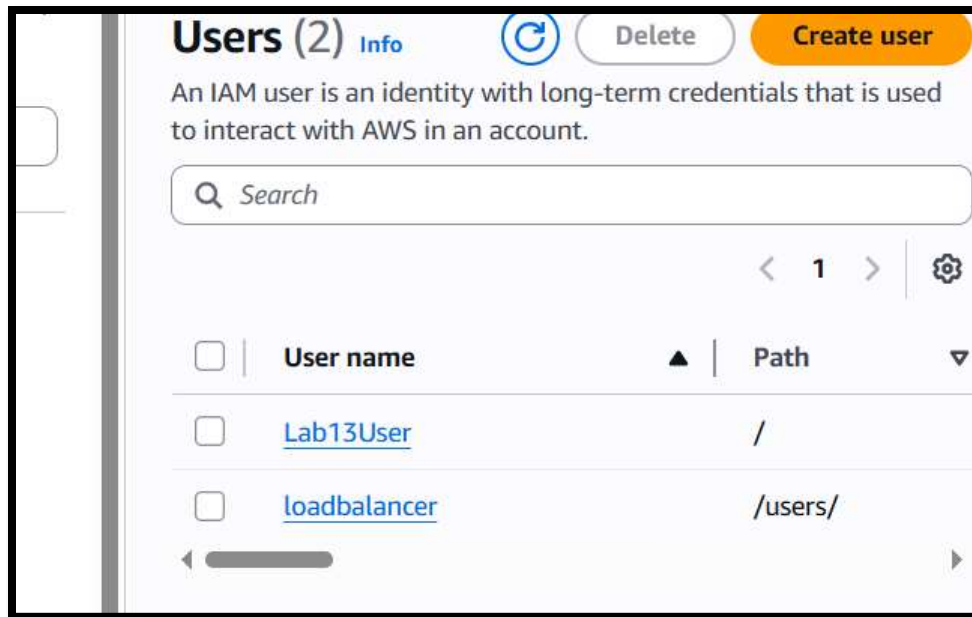
```
group_details = {
  "group_arn" = "arn:aws:iam::249471344514:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATUFNGT6BISKGWTVWD"
}
user_details = {
  "unique_id" = "AIDATUFNGT6BD3M4QT2VS"
  "user_arn" = "arn:aws:iam::249471344514:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@Anara-hayat → /workspaces/Lab13 (main) $
```

**3. Display the outputs:**

```
@Anara-hayat → /workspaces/Lab13 (main) $ terraform output
group_details = {
  "group_arn" = "arn:aws:iam::249471344514:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATUFNGT6BISKGWTVWD"
}
user_details = {
  "unique_id" = "AIDATUFNGT6BD3M4QT2VS"
  "user_arn" = "arn:aws:iam::249471344514:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@Anara-hayat → /workspaces/Lab13 (main) $
```

**4. Verify the user in AWS Console:**

- Navigate to IAM → Users in AWS Console
- Click on "loadbalancer" user
- Check the "Groups" tab



### Task 3 — Attach Policies to IAM Group

In this task, you will attach AWS managed policies (AmazonEC2FullAccess and IAMUserChangePassword) to the developers group.

1. Update main.tf to add policy attachments:

```

    DisplayName = "LoadBalancer"
  }
}

resource "aws_iam_user_group_membership" "lb_membership" {
  user = aws_iam_user.lb.name
  groups = [
    aws_iam_group.developers.name
  ]
}

output "user_details" {
  value = {
    user_name = aws_iam_user.lb.name
    user_arn  = aws_iam_user.lb.arn
    unique_id = aws_iam_user.lb.unique_id
  }
}

resource "aws_iam_group_policy_attachment" "developer_ec2_fullaccess" {
  group = aws_iam_group.developers.name
  policy_arn = "arn:aws:iam::aws:policy/AmazonEC2FullAccess"
}

resource "aws_iam_group_policy_attachment" "change_password" {
  group = aws_iam_group.developers.name
  policy_arn = "arn:aws:iam::aws:policy/IAMUserChangePassword"
}

```

## 2. Apply the configuration:

```

Terraform will perform the following actions:

# aws_iam_group_policy_attachment.change_password will be created
+ resource "aws_iam_group_policy_attachment" "change_password" {
  + group      = "developers"
  + id         = (known after apply)
  + policy_arn = "arn:aws:iam::aws:policy/IAMUserChangePassword"
}

# aws_iam_group_policy_attachment.developer_ec2_fullaccess will be created
+ resource "aws_iam_group_policy_attachment" "developer_ec2_fullaccess" {
  + group      = "developers"
  + id         = (known after apply)
  + policy_arn = "arn:aws:iam::aws:policy/AmazonEC2FullAccess"
}

Plan: 2 to add, 0 to change, 0 to destroy.
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Creating...
aws_iam_group_policy_attachment.change_password: Creating...
aws_iam_group_policy_attachment.change_password: Creation complete after 1s [id=developers-20260102204305454700000001]
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Creation complete after 1s [id=developers-20260102204305464100000002]

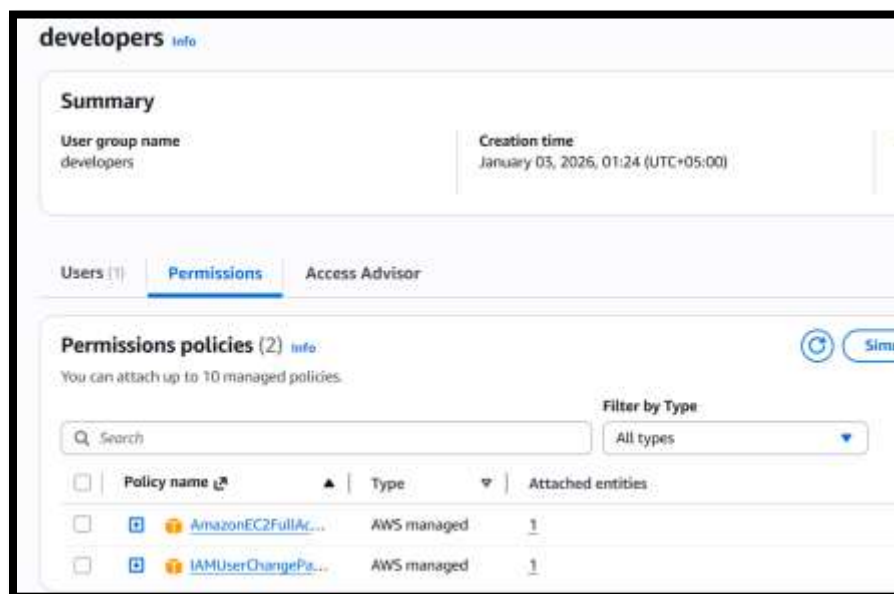
Apply complete! Resources: 2 added, 0 changed, 0 destroyed.

Outputs:
group_details = {
  "group_arn" = "arn:aws:iam::249471344514:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATUFNGT6BISKGWTVWD"
}
user_details = {
  "unique_id" = "AIDATUFNGT6BD3M4QT2VS"
  "user_arn" = "arn:aws:iam::249471344514:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}

```

### 3. Verify policies in AWS Console:

- Navigate to IAM → Groups → developers
- Click on "Permissions" tab



## Task 4 — Create Login Profile for IAM User

In this task, you will create a login profile for the loadbalancer user using a bash script and null\_resource provisioner.

1. Create variables.tf file:

```
variable "iam_password" {  
  description = "Temporary password for the IAM user"  
  type        = string  
  sensitive   = true  
  default     = "1dontKnow"
```

2. Create the bash script create-login-profile.sh:

```
#!/usr/bin/env bash  
set -euo pipefail  
  
USERNAME="$1"  
PASSWORD="$2"  
  
# Check if login profile already exists  
if aws iam get-login-profile --user-name "$USERNAME" >/dev/null 2>&1; then  
  echo "Login profile already exists for $USERNAME. Skipping."  
else  
  echo "Creating login profile for $USERNAME"  
  aws iam create-login-profile \  
    --user-name "$USERNAME" \  
    --password "$PASSWORD" \  
    --password-reset-required  
fi
```

3. Make the script executable:

```
@Anara-hayat → /workspaces/Lab13 (main) $ chmod +x create-login-profile.sh  
@Anara-hayat → /workspaces/Lab13 (main) $
```

4. Update main.tf to add the null\_resource provisioner:

Add this resource after the user creation:

```

}
resource "aws_iam_user" "lb" {
  name = "loadbalancer"
  path = "/users/"
  force_destroy = true
  tags = {
    DisplayName = "Load Balancer"
  }
}

resource "null_resource" "create_login_profile" {
  triggers = {
    password_hash = sha256(var.iam_password)
    user          = aws_iam_user.lb.name
  }

  depends_on = [aws_iam_user.lb]

  provisioner "local-exec" {
    command = "${path.module}/create-login-profile.sh ${aws_iam_user.lb.name} '${var.iam_password}'"
  }
}

resource "aws_iam_user_group_membership" "lb_membership" {
  user = aws_iam_user.lb.name
  groups = [
    aws_iam_group.developers.name
  ]
}

```

##### 5. Apply the configuration with a custom password:

```

@Anara-hayat + /workspaces/Lab13 (main) $ terraform apply -auto-approve -var="iam_password=MySecurePass123!"
aws_iam_user.lb: Refreshing state... [id=loadbalancer]
aws_iam_group.developers: Refreshing state... [id=developers]
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Refreshing state... [id=developers-20260102204305464100000002]
aws_iam_group_policy_attachment.change_password: Refreshing state... [id=developers-20260102204305454700000001]
aws_iam_user_group_membership.lb_membership: Refreshing state... [id=terraform-20260102203133428700000001]

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:

# null_resource.create_login_profile will be created
+ resource "null_resource" "create_login_profile" {
  + id          = (known after apply)
  + triggers = {
    + "password_hash" = (sensitive value)
    + "user"          = "loadbalancer"
  }
}

Plan: 1 to add, 0 to change, 0 to destroy.
null_resource.create_login_profile: Creating...
null_resource.create_login_profile: Provisioning with 'local-exec'...
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)

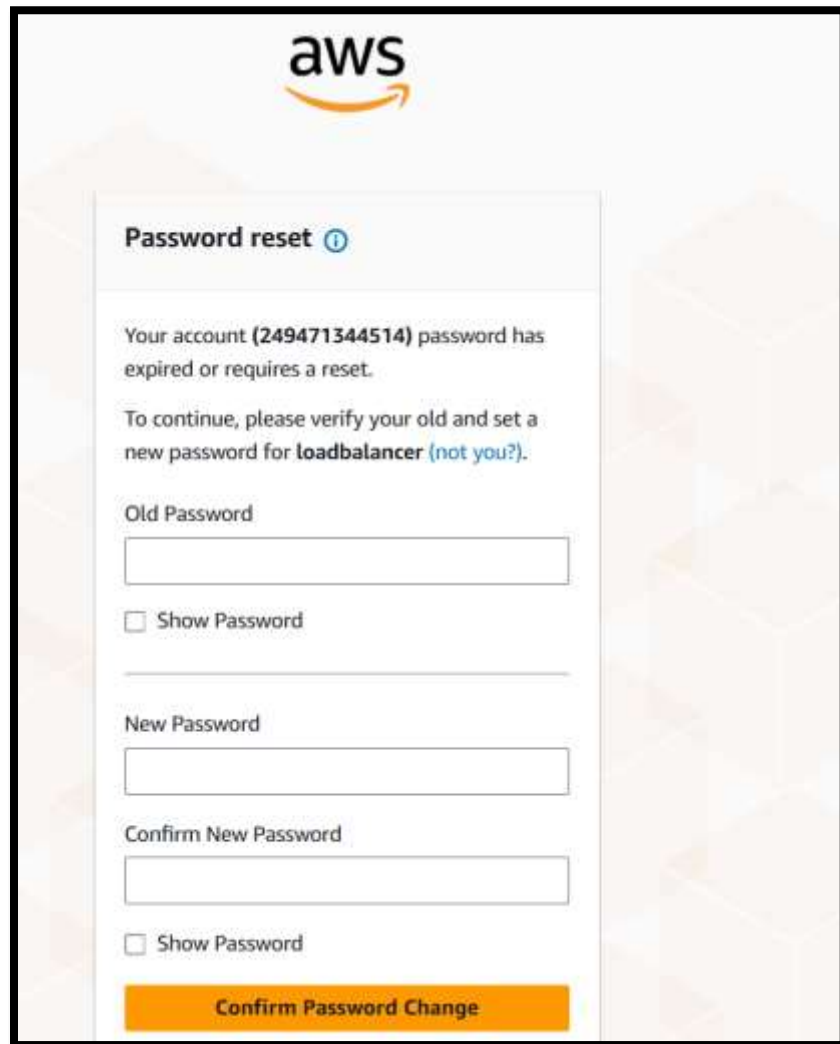
```

## 6. Verify login profile creation:

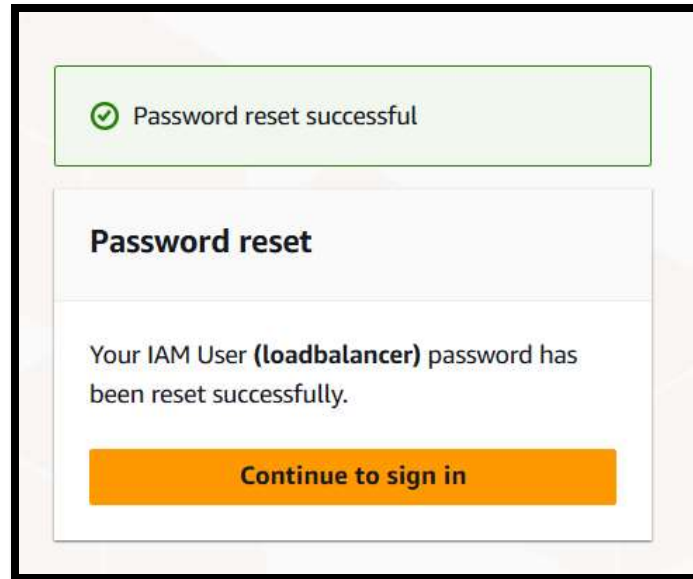
```
@Anara-hayat → /workspaces/Lab13 (main) $ aws iam get-login-profile --user-name loadbalancer
{
  "LoginProfile": {
    "UserName": "loadbalancer",
    "CreateDate": "2026-01-02T20:51:41+00:00",
    "PasswordResetRequired": true
  }
}
```

## 7. Test login in AWS Console:

- Open AWS Console login page
- Sign in as IAM user with username "loadbalancer" and the password you set
- You should be prompted to change password



The screenshot shows the AWS Password reset page. At the top is the AWS logo. Below it, the heading "Password reset" is followed by an information icon. The main text states: "Your account (249471344514) password has expired or requires a reset. To continue, please verify your old and set a new password for loadbalancer (not you?).". There are three input fields: "Old Password", "New Password", and "Confirm New Password". Each field has a "Show Password" checkbox below it. At the bottom is an orange button labeled "Confirm Password Change".



## Task 5 — Generate Access Keys for IAM User

In this task, you will create access keys for the loadbalancer user and view them in terraform state.

### 1. Update main.tf to add access key resource and outputs:

```
value = {
  user_name = aws_iam_user.lb.name
  user_arn  = aws_iam_user.lb.arn
  unique_id = aws_iam_user.lb.unique_id
}

resource "aws_iam_group_policy_attachment" "developer_ec2_fullaccess" {
  group = aws_iam_group.developers.name
  policy_arn = "arn:aws:iam::aws:policy/AmazonEC2FullAccess"
}

resource "aws_iam_group_policy_attachment" "change_password" {
  group = aws_iam_group.developers.name
  policy_arn = "arn:aws:iam::aws:policy/IAMUserChangePassword"
}

resource "aws_iam_access_key" "lb_access_key" {
  user = aws_iam_user.lb.name
}

output "access_key_id" {
  value = aws_iam_access_key.lb_access_key.id
}

output "access_key_secret" {
  value = aws_iam_access_key.lb_access_key.secret
  sensitive = true
}
```

## 2. Apply the configuration:

```
+ access_key_secret = (sensitive value)
null_resource.create_login_profile: Destroying... [id=1741059235772507478]
null_resource.create_login_profile: Destruction complete after 0s
null_resource.create_login_profile: Creating...
null_resource.create_login_profile: Provisioning with 'local-exec'...
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
aws_iam_access_key.lb_access_key: Creating...
aws_iam_access_key.lb_access_key: Creation complete after 1s [id=AKIATUFNGT6BNCB3WLLG]
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
null_resource.create_login_profile: Creation complete after 1s [id=19221029713432849]
```

**Apply complete! Resources: 2 added, 0 changed, 1 destroyed.**

### Outputs:

```
access_key_id = "AKIATUFNGT6BNCB3WLLG"
access_key_secret = <sensitive>
group_details = {
  "group_arn" = "arn:aws:iam::249471344514:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATUFNGT6BISKGWTVWD"
}
user_details = {
  "unique_id" = "AIDATUFNGT6BD3M4QT2VS"
  "user_arn" = "arn:aws:iam::249471344514:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
```

## 3. Display outputs:

```
@Anara-hayat → /workspaces/Lab13 (main) $ terraform output
access_key_id = "AKIATUFNGT6BNCB3WLLG"
access_key_secret = <sensitive>
group_details = {
  "group_arn" = "arn:aws:iam::249471344514:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATUFNGT6BISKGWTVWD"
}
user_details = {
  "unique_id" = "AIDATUFNGT6BD3M4QT2VS"
  "user_arn" = "arn:aws:iam::249471344514:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
```

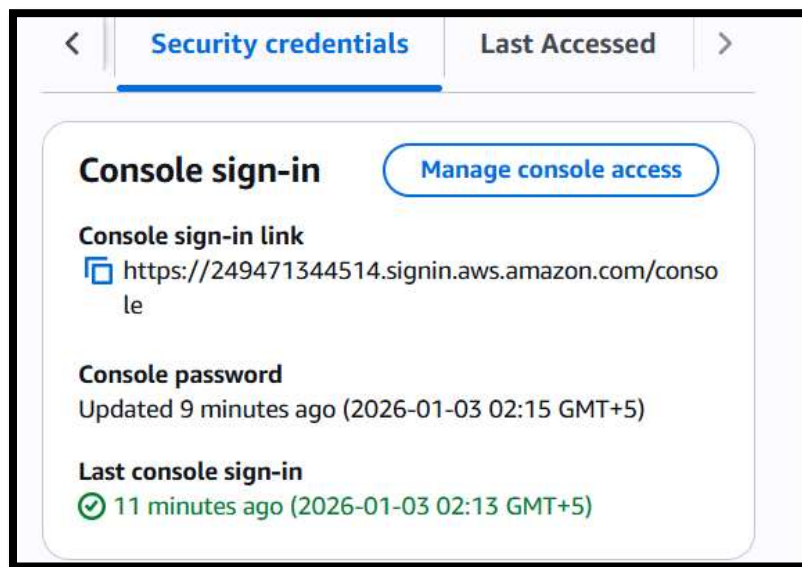
## 4. View the secret in terraform state:

```
@Anara-hayat → /workspaces/Lab13 (main) $ cat terraform.tfstate | grep -A 10 "access_key_secret"
"access_key_secret": {
  "value": "LVR2UtrS1b/5cEpi6TY8FRi4cFpjqEaNfxEJo",
  "type": "string",
  "sensitive": true
},
"group_details": {
  "value": {
    "group_arn": "arn:aws:iam::249471344514:group/groups/developers",
    "group_name": "developers",
    "unique_id": "AGPATUFNGT6BISKGWTVWD"
  },

```

### 5. Verify access key in AWS Console:

Navigate to IAM → Users → loadbalancer → Security credentials



## Task 6 — Implement Terraform Remote State with S3

In this task, you will configure Terraform to use S3 backend for remote state storage.

### 1. Create S3 bucket in AWS Console:

- Navigate to S3 in AWS Console
- Click "Create bucket"
- Bucket name: myapp-s3-bucket-demo (use a unique name if this is taken)
- Enable versioning
- Keep other settings as default
- Click "Create bucket"



### General configuration

**AWS Region**  
South America (São Paulo) sa-east-1

**Bucket name** [Info](#)

myapp-s3-bucket3

Bucket names must be 3 to 63 characters and unique within the global namespace. Bucket names must also begin and end with a letter or number. Valid characters are a-z, 0-9, periods (.), and hyphens (-). [Learn more](#)

**Copy settings from existing bucket - optional**  
Only the bucket settings in the following configuration are copied.

Choose bucket

Format: s3://bucket/prefix

### Bucket Versioning

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. [Learn more](#)

**Bucket Versioning**

☐ Disable

☒ Enable

## 2. Update main.tf to add S3 backend configuration:

Add this at the beginning of main.tf (before the provider block):

```

    user = aws_iam_user.lb.name
  }

  output "access_key_id" {
    value = aws_iam_access_key.lb_access_key.id
  }

  output "access_key_secret" {
    value     = aws_iam_access_key.lb_access_key.secret
    sensitive = true
  }
}

terraform {
  backend "s3" {
    bucket = "myapp-s3-bucket3"
    key    = "myapp/terraform.tfstate"
    region = "sa-east-1"
    encrypt = true
    use_lockfile = true
  }
}

```

### 3. Reinitialize Terraform with the backend:

```

@Anara-hayat → /workspaces/Lab13 (main) $ terraform init -migrate-state
Initializing the backend...
Do you want to copy existing state to the new backend?
Pre-existing state was found while migrating the previous "local" backend to the
newly configured "s3" backend. No existing state was found in the newly
configured "s3" backend. Do you want to copy this state to the new "s3"
backend? Enter "yes" to copy and "no" to start with an empty state.

Enter a value: yes

Releasing state lock. This may take a few moments...

Successfully configured the backend "s3"! Terraform will automatically
use this backend unless the backend configuration changes.
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Reusing previous version of hashicorp/null from the dependency lock file
- Using previously-installed hashicorp/null v3.2.4
- Using previously-installed hashicorp/aws v6.27.0

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.
@Anara-hayat → /workspaces/Lab13 (main) $

```

#### 4. Apply the configuration:

```
@Anara-hayat → /workspaces/Lab13 (main) $ terraform apply -auto-approve -var="iam_password=Loadbalancer@123"
aws_iam_user.lb: Refreshing state... [id=loadbalancer]
aws_iam_group.developers: Refreshing state... [id=developers]
aws_iam_group_policy_attachment.change_password: Refreshing state... [id=developers-20260102204305454700000001]
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Refreshing state... [id=developers-20260102204305464100000002]
aws_iam_user_group_membership.lb_membership: Refreshing state... [id=terraform-20260102203133428700000001]
aws_iam_access_key.lb_access_key: Refreshing state... [id=AKIATUFNGT6BNCB3WLLG]
null_resource.create_login_profile: Refreshing state... [id=1922102971343255849]

No changes. Your infrastructure matches the configuration.

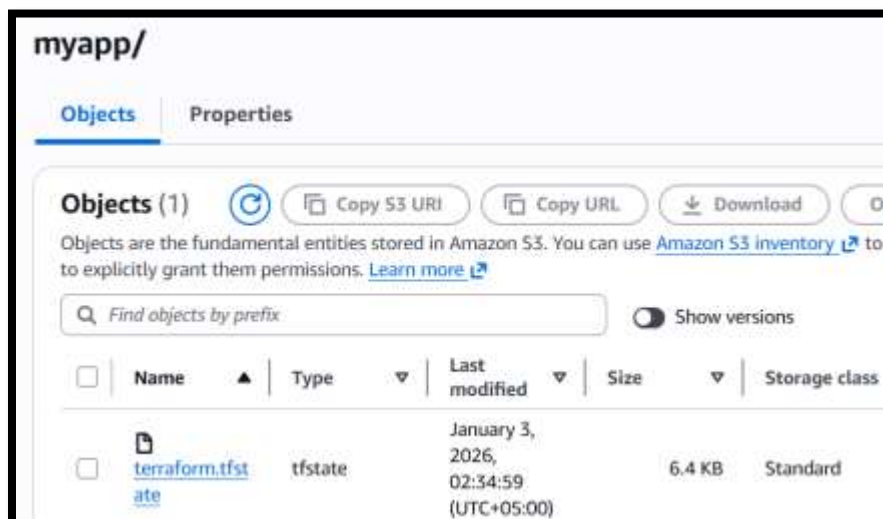
Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.
Releasing state lock. This may take a few moments...

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

Outputs:
access_key_id = "AKIATUFNGT6BNCB3WLLG"
access_key_secret = <sensitive>
group_details = {
  "group_arn" = "arn:aws:iam::249471344514:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATUFNGT6BISKGWTVWD"
}
user_details = {
  "unique_id" = "AIDATUFNGT6BD3M4QT2VS"
  "user_arn" = "arn:aws:iam::249471344514:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
```

#### 5. Verify state file in S3:

- Navigate to S3 → myapp-s3-bucket-demo → myapp/
- You should see terraform.tfstate file



## 6. Check local state file:

```
@Anara-hayat → /workspaces/Lab13 (main) $ ls -la terraform.tfstate*  
-rw-rw-rw- 1 codespace codespace 0 Jan 2 21:33 terraform.tfstate  
-rw-rw-rw- 1 codespace codespace 6602 Jan 2 21:33 terraform.tfstate.backup
```

## 7. Destroy resources and verify state change:

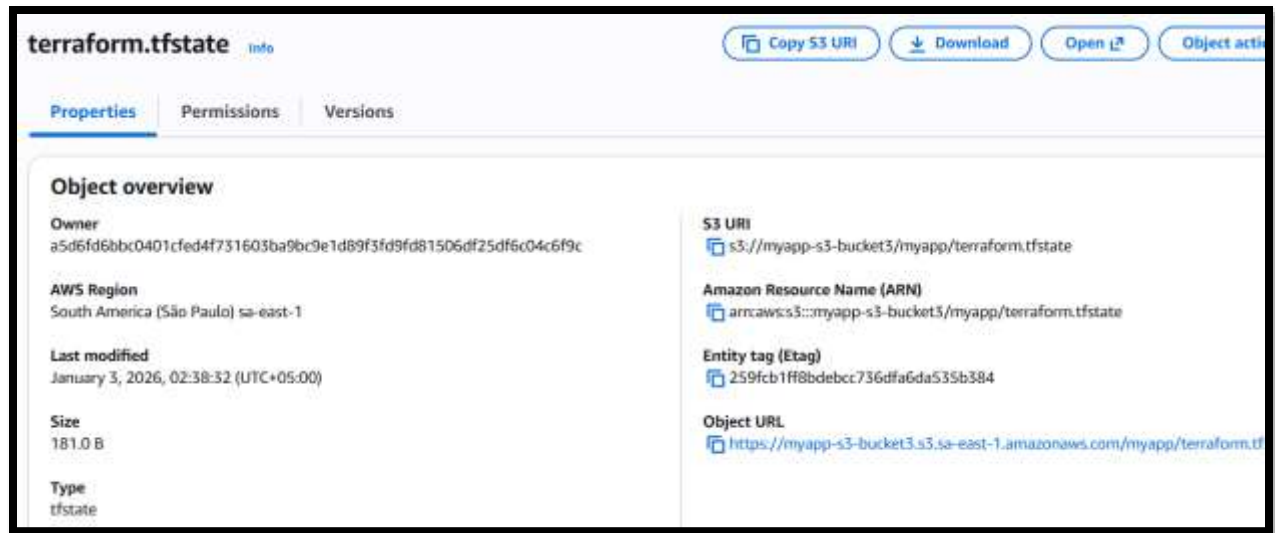
terraform destroy -auto-approve

```
null_resource.create_login_profile: Destroying... [id=1922102971343255849]  
null_resource.create_login_profile: Destruction complete after 0s  
aws_iam_group_policy_attachment.change_password: Destroying... [id=developers-202601022043054547000000001]  
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Destroying... [id=developers-202601022043054641000000002]  
aws_iam_access_key.lb_access_key: Destroying... [id=AKIATUFNGT6BNCB3WLLG]  
aws_iam_user_group_membership.lb_membership: Destroying... [id=terraform-202601022031334287000000001]  
aws_iam_user_group_membership.lb_membership: Destruction complete after 0s  
aws_iam_group_policy_attachment.change_password: Destruction complete after 1s  
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Destruction complete after 1s  
aws_iam_group.developers: Destroying... [id=developers]  
aws_iam_access_key.lb_access_key: Destruction complete after 1s  
aws_iam_user.lb: Destroying... [id=loadbalancer]  
aws_iam_group.developers: Destruction complete after 0s  
aws_iam_user.lb: Destruction complete after 2s  
Releasing state lock. This may take a few moments...  
  
Destroy complete! Resources: 7 destroyed.
```

## 8. Verify updated state in S3:

- Refresh S3 bucket view
- Check the terraform.tfstate file (it should show empty resources)

```
@Anara-hayat → /workspaces/Lab13 (main) $ aws s3 cp s3://myapp-s3-bucket3/myapp/terraform.tfstate - | jq  
{  
  "version": 4,  
  "terraform_version": "1.10.0",  
  "serial": 3,  
  "lineage": "79506c35-b767-7dd5-9ba3-e0fdcd568c9a",  
  "outputs": {},  
  "resources": [],  
  "check_results": null  
}
```



## Task 7 — Create Multiple Users from CSV File

In this task, you will create multiple IAM users dynamically from a CSV file.

1. Create locals.tf file:

```
locals {  
  users = csvdecode(file("users.csv"))  
}
```

2. Create users.csv file:



**3. Update main.tf to create multiple users:**

**Replace the single user resources with:**

```

# Create multiple IAM users from CSV
resource "aws_iam_user" "users" {
  for_each = { for user in local.users : user.user_name => user }

  name           = each.value.user_name
  path           = "/users/"
  force_destroy = true

  tags = {
    DisplayName = each.value.user_name
    CreatedBy   = "Terraform"
  }
}

# Add all users to developers group
resource "aws_iam_user_group_membership" "users_membership" {
  for_each = aws_iam_user.users

  user = each.value.name
  groups = [
    aws_iam_group.developers.name
  ]
}

# Create login profiles for all users
resource "null_resource" "create_login_profiles" {
  for_each = aws_iam_user.users

  triggers = {
    password_hash = sha256(var.iam_password)
    user          = each.value.name
  }

  depends_on = [aws_iam_user.users]

  provisioner "local-exec" {
    command = "${path.module}/create-login-profile.sh ${each.value.name}
    password!"
  }
}

```

#### 4. Reinitialize Terraform (since we changed the configuration significantly):

```

@Anara-hayat + /workspaces/Lab13 (main) $ terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Reusing previous version of hashicorp/null from the dependency lock file
- Using previously-installed hashicorp/aws v6.27.0
- Using previously-installed hashicorp/null v3.2.4

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.

```

#### 5. Apply the configuration to create all users:

**terraform apply -auto-approve -var="iam\_password=MySecurePass123!"**

```

}
"Phyllis" = {
  "access_key_id" = "AKIATUFNGT6BJHBRCZ5R"
  "user_arn" = "arn:aws:iam::249471344514:user/users/Phyllis"
  "user_unique_id" = "AIDATUFNGT6BFV4F30MW0"
}
"Robert" = {
  "access_key_id" = "AKIATUFNGT6BDIJ5NKR"
  "user_arn" = "arn:aws:iam::249471344514:user/users/Robert"
  "user_unique_id" = "AIDATUFNGT6BB5J2G6WUG"
}
"Ryan" = {
  "access_key_id" = "AKIATUFNGT6BCQXGP60"
  "user_arn" = "arn:aws:iam::249471344514:user/users/Ryan"
  "user_unique_id" = "AIDATUFNGT6BIDVUCXGFF"
}
"Stanley" = {
  "access_key_id" = "AKIATUFNGT6BLG0WCF2N"
  "user_arn" = "arn:aws:iam::249471344514:user/users/Stanley"
  "user_unique_id" = "AIDATUFNGT6BGWQPQGFPG"
}
"Toby" = {
  "access_key_id" = "AKIATUFNGT6BHCWLMW7F"
  "user_arn" = "arn:aws:iam::249471344514:user/users/Toby"
  "user_unique_id" = "AIDATUFNGT6BJTUM3ENW5"
}
}
group_details = {
  "group_arn" = "arn:aws:iam::249471344514:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATUFNGT6BJ4YHF52BW"
}
}

```

## 6. Display the outputs:

```

@Anara-hayat ~ /workspaces/Lab13 (main) $ terraform output
all_access_key_secrets = <sensitive>
all_users_details = {
  "Andy" = {
    "access_key_id" = "AKIATUFNGT6BKPRXIN4R"
    "user_arn" = "arn:aws:iam::249471344514:user/users/Andy"
    "user_unique_id" = "AIDATUFNGT6BHMYPWPBHS"
  }
  "Angela" = {
    "access_key_id" = "AKIATUFNGT6BHKFL5XFQ"
    "user_arn" = "arn:aws:iam::249471344514:user/users/Angela"
    "user_unique_id" = "AIDATUFNGT6BLP27TGHEE"
  }
  "Charles" = {
    "access_key_id" = "AKIATUFNGT6BKB6J2YXF"
    "user_arn" = "arn:aws:iam::249471344514:user/users/Charles"
    "user_unique_id" = "AIDATUFNGT6BBQZUY6FIZ"
  }
  "Clark" = {
    "access_key_id" = "AKIATUFNGT6BAJICG2MV"
    "user_arn" = "arn:aws:iam::249471344514:user/users/Clark"
    "user_unique_id" = "AIDATUFNGT6BFM7XZ5QXF"
  }
  "Creed" = {
    "access_key_id" = "AKIATUFNGT6BLKKYUVJ0"
    "user_arn" = "arn:aws:iam::249471344514:user/users/Creed"
    "user_unique_id" = "AIDATUFNGT6BMKB43R654"
  }
  "Darryl" = {
    "access_key_id" = "AKIATUFNGT6BPBU7ZAG6"
    "user_arn" = "arn:aws:iam::249471344514:user/users/Darryl"
    "user_unique_id" = "AIDATUFNGT6BKP3QUYYHW"
  }
  "David" = {
    "access_key_id" = "AKIATUFNGT6BPAGWNPSB"
    "user_arn" = "arn:aws:iam::249471344514:user/users/David"
    "user_unique_id" = "AIDATUFNGT6BGQCKW5UQ2"
  }
}

```

## 7. View secrets in terraform. tfstate:

```
@Anara-hayat → /workspaces/Lab13 (main) $ terraform output all_access_key_secrets
{
  "Andy" = "d4ho0Hv0+N9B5hPxUfz0vCU2iKpHXY6WJiD5RnH"
  "Angela" = "F17GkN2ypr6ybnd06QTvj01lvxaPa94tEi8ITeWG"
  "Charles" = "/1MzCDiiWExWT6rkgn/GGARQZscW8rDCGPLU1PBG"
  "Clark" = "/Yfzr+3D1V1PH9XW5Z90H0ImJmr3fMibJwhj7+MU"
  "Creed" = "C5vemQD/EyMvsiwIPgmEjGaJmNX4gF+VetKb5M7p"
  "Darryl" = "WXI63+F1riPoIGETFZJ+RYGj34cFfsgo/68Jsiw3"
  "David" = "GMQCI1wVA0DzR4Vlnkawa+nhQqwP8nNby/uXgWHy"
  "Dwight" = "A3q1bK7N7p/DHN6aCHVyyvUd3KuHeYWTpcb@aEnQ"
  "Erin" = "wY4lmVzgZeGrrLIKIeuSxz0Q8VvuPKRv8L2mGhZ8"
  "Gabe" = "B/zZrY2kbTob7PLLWppu0HEHi+GDeQNYCQnLHB6N"
  "Holly" = "/thLGUw4A81LX6zJKWXFq6ZmisBAfrJS9JRx/GLW"
  "Jan" = "IFw1MgJw7wa8LrXDFvF3qwmeZtDtF4F2326/2VWg"
  "Jim" = "aA50Sx0ynd0i0JhW9YJ2ncvXFeUrwXjXo//yqWmZ"
  "Jo" = "HhPSczpdz7NnnSKIlbJbU9mZbBTcmwKnBz0pPv4q"
  "Kelly" = "g3kUEXVDNqcjKgFIATyzHNuc4NoZY0AlLuDv3YMq"
  "Kevin" = "5bGK7S1IzNHU079qqw4+Ho7Zyzub9/gpMyLv/zNE"
  "Meredith" = "5jPKbq2EG2b13JA25QoF1sDYzuZIXnQ809+uja1X"
  "Michael" = "yPm8eHp1BqsUpe+YtoiqpSGjL8lmq4xhts1gkg/T"
  "Oscar" = "c84xgsjrR3G9Ja/DhVPw2WpW0SpYX6xnRZGbDEWv"
  "Pam" = "4oWn/NcHzaCTWnB+1THuJ+MwaqdvLb8LL5n01uIg"
  "Peter" = "BVyK4e4DU5uP6bZIkFke/s6++iW20/rn60Mi1LUU"
  "Phyllis" = "@AY0HaBiUNJTD7l15Hv8BC0ypeAlB2cFS0D70jad"
  "Robert" = "bYCzv6KBrVFVbppL9xrhtmcvMglzzMgVwBV62um"
  "Ryan" = "E7pGR9H3bay8gQ9Dd6nY0vnPVGgUIC32zrpfRium"
  "Stanley" = "JLTpvt7/UEFdAfUJ9xbym0VKwoneTQDLT1uEzBZn"
  "Toby" = "+Ef/yt11kpe+GVXxaZM1tHX2wRFR3wwA4SSbqqot"
}
```

## 8. Verify all users in AWS Console:

Navigate to IAM → Users

**Users (27)** [Info](#)

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

Q Search

<input type="checkbox"/>	User name ▲	Path ▼	Group: ▼	Last activity ▼
<input type="checkbox"/>	<a href="#">Andy</a>	/users/	1	-
<input type="checkbox"/>	<a href="#">Angela</a>	/users/	1	-
<input type="checkbox"/>	<a href="#">Charles</a>	/users/	1	-
<input type="checkbox"/>	<a href="#">Clark</a>	/users/	1	-
<input type="checkbox"/>	<a href="#">Creed</a>	/users/	1	-
<input type="checkbox"/>	<a href="#">Darryl</a>	/users/	1	-
<input type="checkbox"/>	<a href="#">David</a>	/users/	1	-
<input type="checkbox"/>	<a href="#">Dwight</a>	/users/	1	-
<input type="checkbox"/>	<a href="#">Erin</a>	/users/	1	-
<input type="checkbox"/>	<a href="#">Gabe</a>	/users/	1	-
<input type="checkbox"/>	<a href="#">Holly</a>	/users/	1	-

## 9. Verify group membership:

Navigate to IAM → Groups → developers → Users tab

**Users (26)** | **Permissions** | **Access Advisor**

**Users in this group (26)** [Refresh](#)

An IAM user is an entity that you create in AWS to represent the person or application that uses it to interact with AWS.

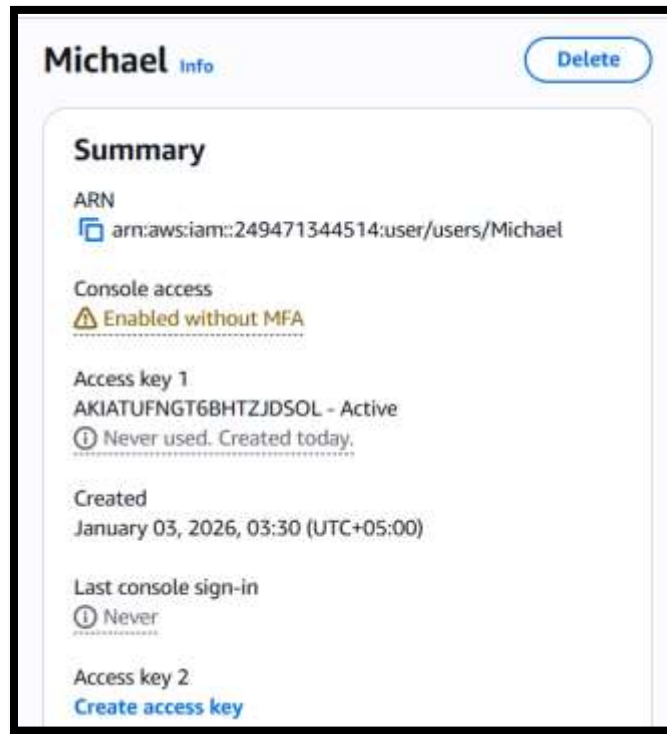
Q Search

<input type="checkbox"/>	User name ↗	Groups	Last activity ▼	Creation time
<input type="checkbox"/>	<a href="#">Andy</a>		None	26 minutes ago
<input type="checkbox"/>	<a href="#">Angela</a>		None	26 minutes ago
<input type="checkbox"/>	<a href="#">Charles</a>		None	26 minutes ago
<input type="checkbox"/>	<a href="#">Clark</a>		None	26 minutes ago
<input type="checkbox"/>	<a href="#">Creed</a>		None	26 minutes ago
<input type="checkbox"/>	<a href="#">Darryl</a>		None	26 minutes ago
<input type="checkbox"/>	<a href="#">David</a>		None	26 minutes ago
<input type="checkbox"/>	<a href="#">Dwight</a>		None	26 minutes ago
<input type="checkbox"/>	<a href="#">Erin</a>		None	26 minutes ago

## 10. Verify one user's access keys:

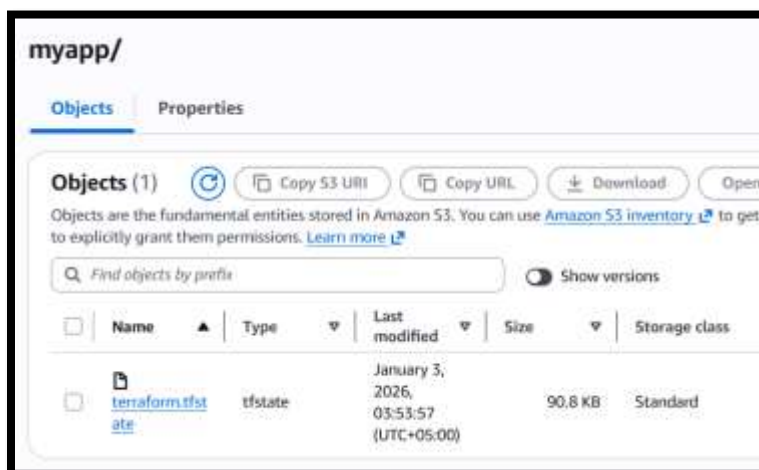
Click on any user (e.g., "Michael")

Go to Security credentials tab



## 11. Check terraform state in S3:

- Navigate to S3 bucket and view the state file



## Cleanup

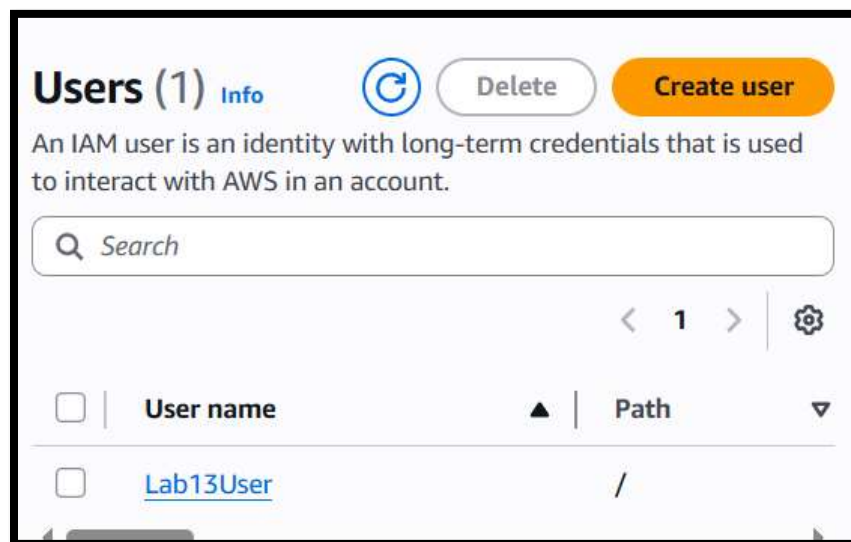
### 1. Destroy all resources:

```
aws_iam_user.users["Jan"]: Destroying... [id=Jan]
aws_iam_user.users["Kevin"]: Destruction complete after 3s
aws_iam_user.users["Ryan"]: Destroying... [id=Ryan]
aws_iam_user.users["Stanley"]: Destruction complete after 4s
aws_iam_user.users["Andy"]: Destroying... [id=Andy]
aws_iam_user.users["Meredith"]: Destruction complete after 2s
aws_iam_user.users["Holly"]: Destroying... [id=Holly]
aws_iam_user.users["Robert"]: Destruction complete after 2s
aws_iam_user.users["Kelly"]: Destruction complete after 8s
aws_iam_user.users["Jo"]: Destruction complete after 3s
aws_iam_user.users["Pam"]: Destruction complete after 3s
aws_iam_user.users["Dwight"]: Destruction complete after 3s
aws_iam_user.users["Charles"]: Destruction complete after 5s
aws_iam_user.users["Jan"]: Destruction complete after 2s
aws_iam_user.users["Ryan"]: Destruction complete after 2s
aws_iam_user.users["Andy"]: Destruction complete after 2s
aws_iam_user.users["Holly"]: Destruction complete after 3s
Releasing state lock. This may take a few moments...
```

**Destroy complete! Resources: 107 destroyed.**

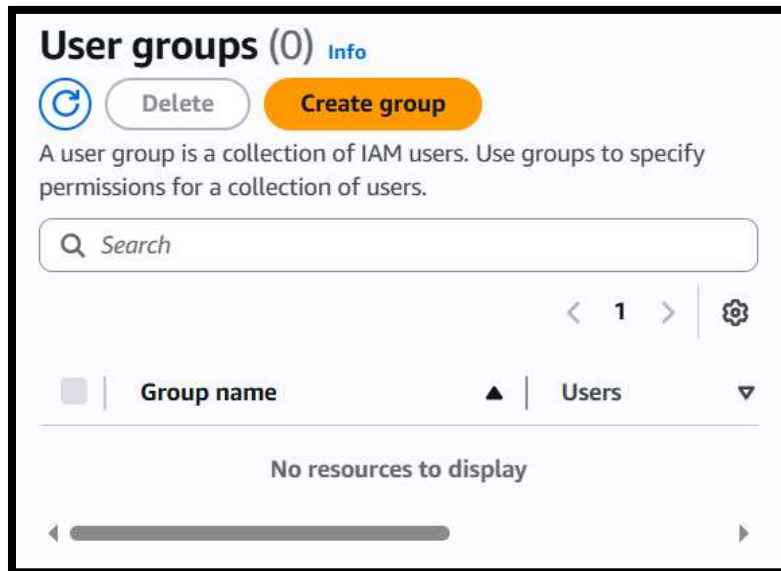
### 2. Verify users deleted in AWS Console:

- Navigate to IAM → Users



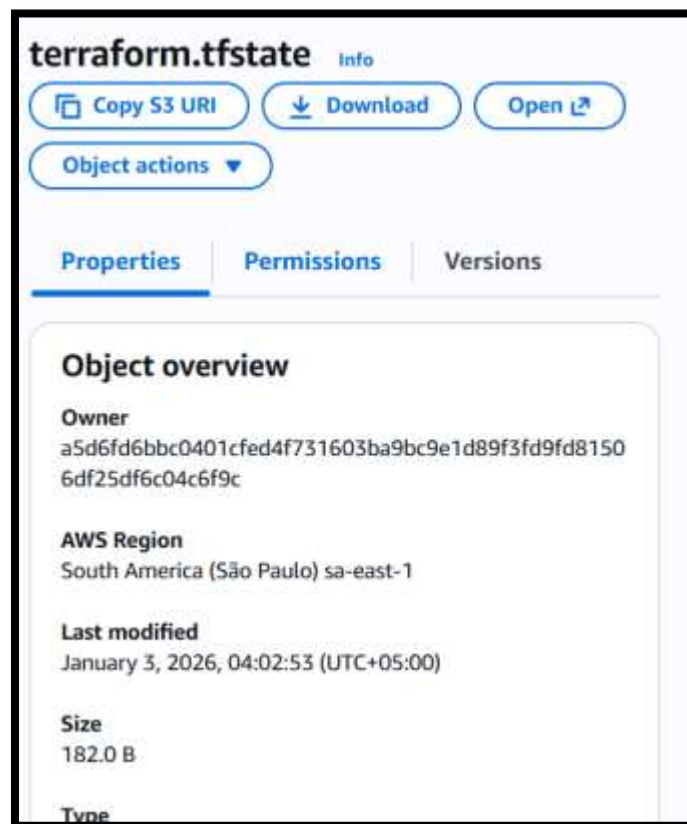
### 3. Verify group deleted in AWS Console:

Navigate to IAM → Groups



#### 4. Check S3 state file:

- Navigate to S3 bucket



5. List all project files:

```
Destroy Complete! Resources: 10/ destroyed.
@Anara-hayat → /workspaces/Lab13 (main) $ ls -la
total 60
drwxrwxrwx+ 4 codespace root      4096 Jan  2 22:52 .
drwxr-xrwx+ 5 codespace root      4096 Jan  2 19:43 ..
drwxrwxrwx+ 8 codespace root      4096 Jan  2 19:43 .git
drwxr-xr-x+ 3 codespace codespace 4096 Jan  2 21:33 .terraform
-rw-r--r--  1 codespace codespace 2422 Jan  2 20:51 .terraform.lock.hcl
-rw-r--r--  1 codespace codespace 4922 Nov 26  2024 LICENSE.txt
-rw-rw-rw-  1 codespace root        18 Jan  2 19:43 README.md
-rwxrwxrwx  1 codespace codespace  380 Jan  2 22:34 create-login-profile.sh
-rw-rw-rw-  1 codespace codespace   50 Jan  2 22:08 locals.tf
-rw-rw-rw-  1 codespace codespace 2498 Jan  2 22:52 main.tf
-rw-rw-rw-  1 codespace codespace    0 Jan  2 21:33 terraform.tfstate
-rw-rw-rw-  1 codespace codespace 6602 Jan  2 21:33 terraform.tfstate.backup
-rw-rw-rw-  1 codespace codespace  167 Jan  2 22:09 users.csv
-rw-rw-rw-  1 codespace codespace  150 Jan  2 20:46 variables.tf
```

6. (Optional) Delete S3 bucket:

- If you want to clean up completely, delete the S3 bucket from AWS Console

## Empty bucket [Info](#)



- Emptying the bucket deletes all objects in the bucket and cannot be undone.
- Objects added to the bucket while the empty bucket action is in progress might be deleted.
- To prevent new objects from being added to this bucket while the empty bucket action is in progress, you can set the bucket to read-only.

[Learn more](#) [↗](#)



If your bucket contains a large number of objects, creating a lifecycle rule to delete all objects in the bucket can help reduce the number of objects and the time it takes to empty the bucket.


### Permanently delete all objects in bucket "myapp-s3-bucket3"?

To confirm deletion, type *permanently delete* in the text input field.

## Delete bucket [Info](#)



- Deleting a bucket cannot be undone.
- Bucket names are unique. If you delete a bucket, another AWS user can use the name.
- If this bucket is used with a Multi-Region Access Point in an external account, initiate failover before deleting the bucket.
- If this bucket is used with an access point in an external account, the requests made through those access points will fail.

[Learn more](#) 

### Delete bucket "myapp-s3-bucket3"?

To confirm deletion, enter the name of the bucket in the text input field.

\*\*\*\*\*