Name: Rughma Malik

Reg no: 2023-BSE-54

Course: Cloud Computing Lab

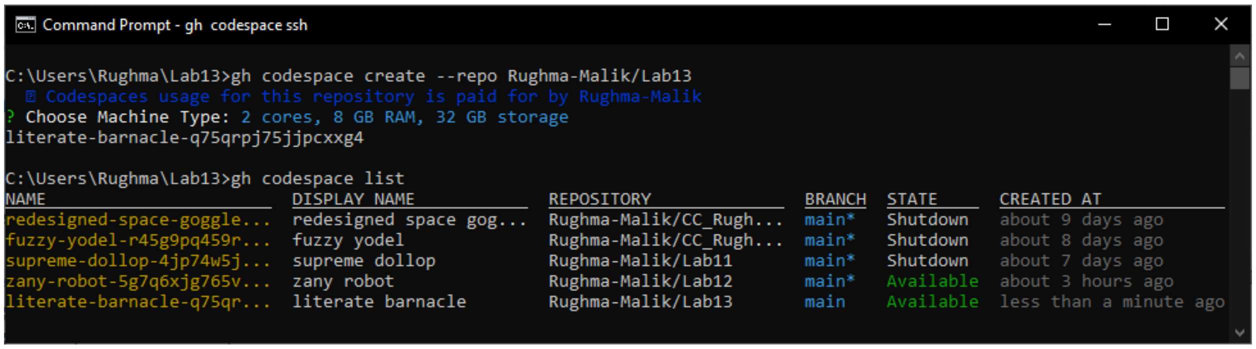
Section: V-B

LAB 12

Terraform Provisioners, Modules & Nginx Reverse Proxy/Load Balancer

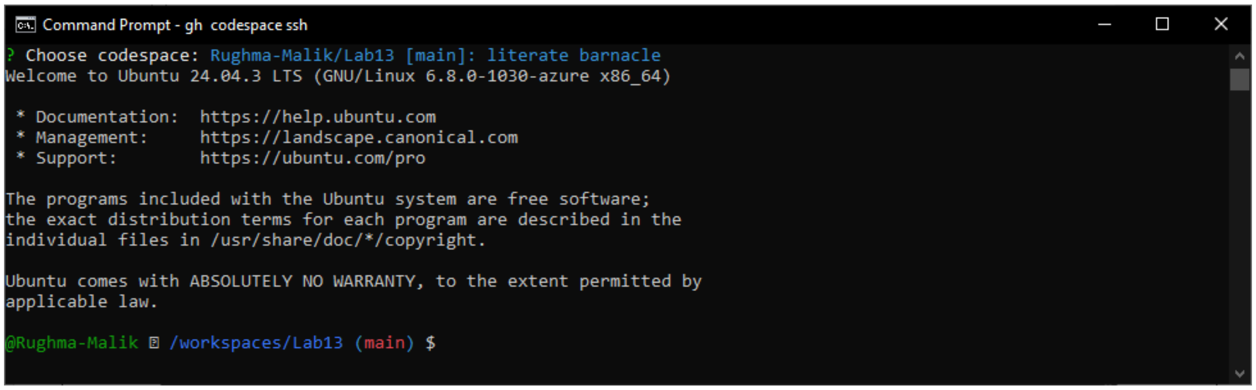
Task 0 Lab Setup (Codespace & GH CLI)

Create Codespace & connect:



Command Prompt - gh codespace ssh

```
C:\Users\Rughma\Lab13>gh codespace create --repo Rughma-Malik/Lab13
  □ Codespaces usage for this repository is paid for by Rughma-Malik
? Choose Machine Type: 2 cores, 8 GB RAM, 32 GB storage
literate-barnacle-q75qrpj75jjpcxxg4

C:\Users\Rughma\Lab13>gh codespace list
NAME                          DISPLAY NAME             REPOSITORY               BRANCH  STATE      CREATED AT
redesigned-space-goggle...    redesigned space gog...  Rughma-Malik/CC_Rugh...  main*   Shutdown   about 9 days ago
fuzzy-yodel-r45g9pq459r...    fuzzy yodel              Rughma-Malik/CC_Rugh...  main*   Shutdown   about 8 days ago
supreme-dollop-4jp74w5j...    supreme dollop           Rughma-Malik/Lab11       main*   Shutdown   about 7 days ago
zany-robot-5g7q6xjg765v...    zany robot               Rughma-Malik/Lab12       main*   Available  about 3 hours ago
literate-barnacle-q75qr...    literate barnacle        Rughma-Malik/Lab13       main    Available  less than a minute ago
```



Command Prompt - gh codespace ssh

```
? Choose codespace: Rughma-Malik/Lab13 [main]: literate barnacle
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.

@Rughma-Malik □ /workspaces/Lab13 (main) $
```

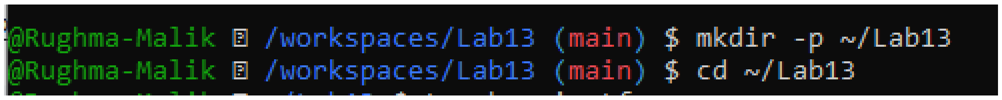
Task 1 — Create IAM Group and Output Details

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

1. Create the initial project structure:

mkdir -p ~/Lab13

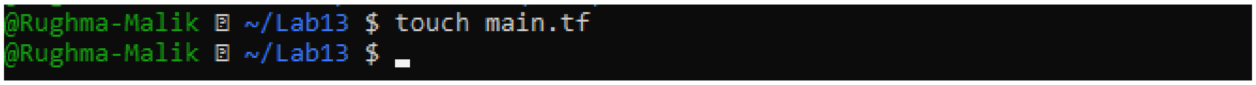
cd ~/Lab13



```
@Rughma-Malik ➜ /workspaces/Lab13 (main) $ mkdir -p ~/Lab13
@Rughma-Malik ➜ /workspaces/Lab13 (main) $ cd ~/Lab13
```

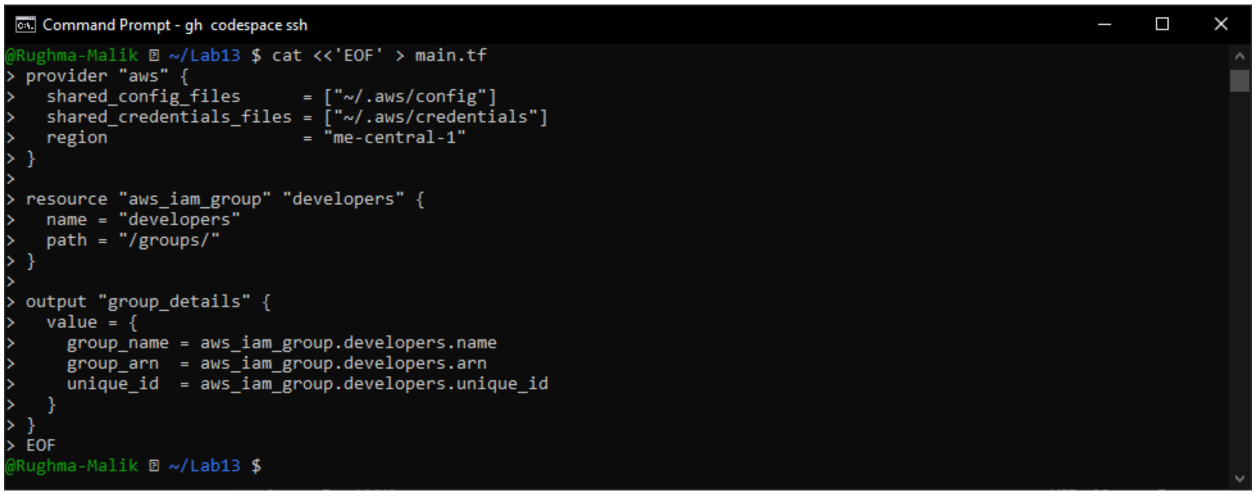
2. Create the main Terraform file:

touch main.tf



```
@Rughma-Malik ➜ ~/Lab13 $ touch main.tf
@Rughma-Malik ➜ ~/Lab13 $
```

3. Create main.tf with AWS provider configuration:

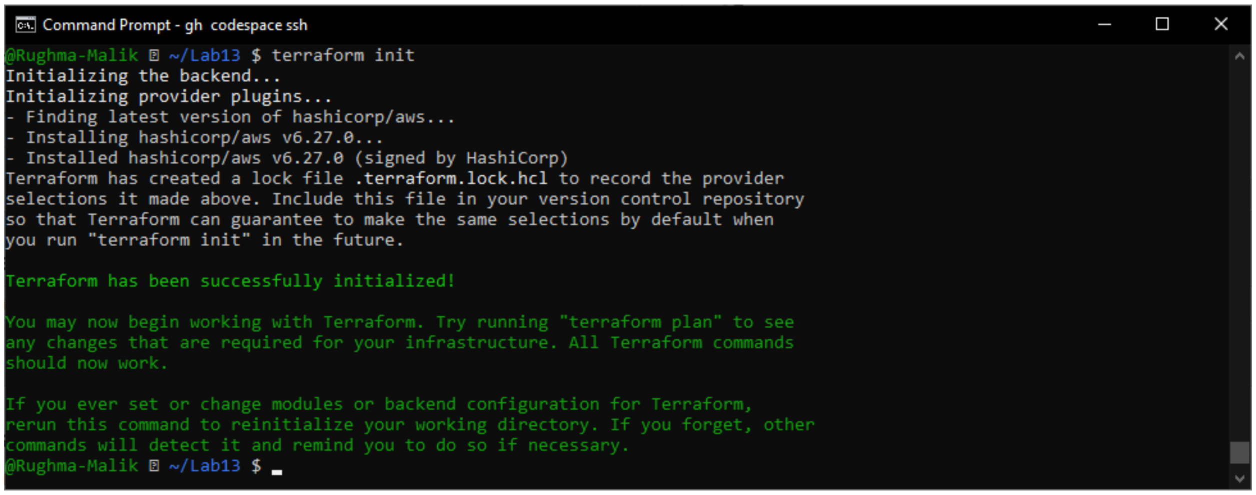


Command Prompt - gh codespace ssh

```
@Rughma-Malik ➜ ~/Lab13 $ cat <<'EOF' > main.tf
> provider "aws" {
>   shared_config_files      = ["~/.aws/config"]
>   shared_credentials_files = ["~/.aws/credentials"]
>   region                   = "me-central-1"
> }
>
> 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
>   }
> }
> EOF
@Rughma-Malik ➜ ~/Lab13 $
```

4. Initialize Terraform:

terraform init

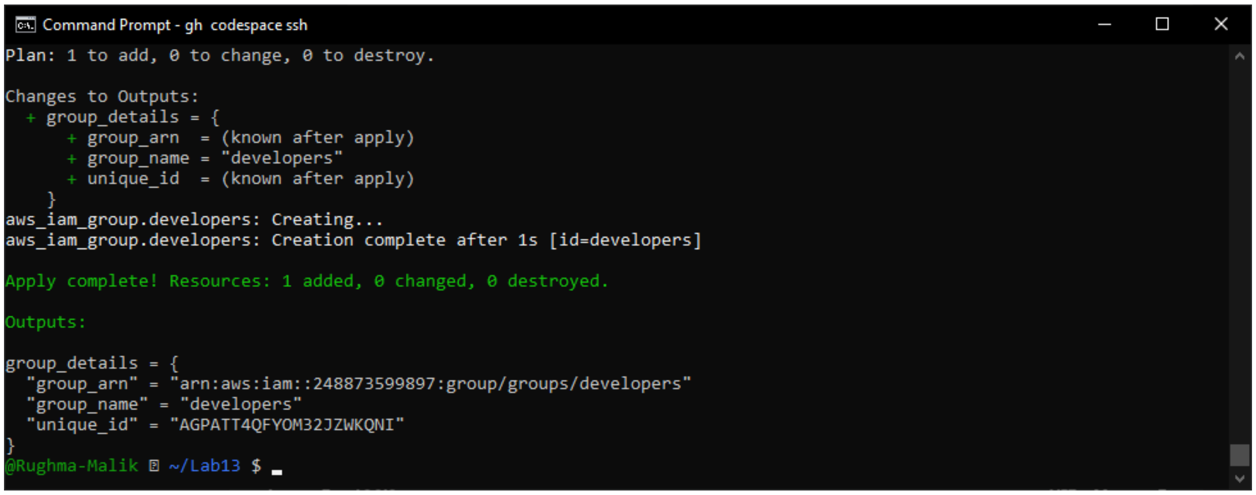


Command Prompt - gh codespace ssh

```
@Rughma-Malik ➜ ~/Lab13 $ 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.
@Rughma-Malik ➜ ~/Lab13 $
```

5. Apply the configuration:

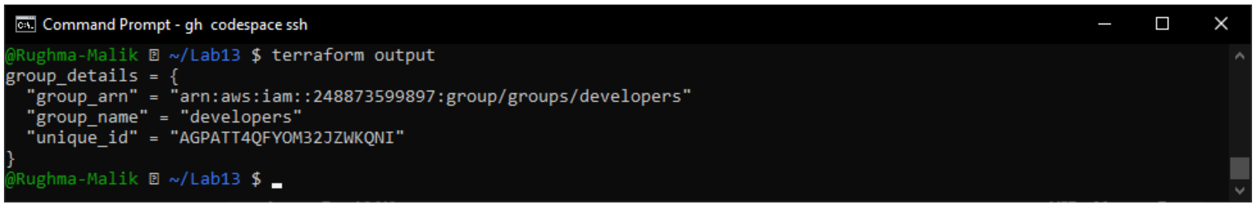
terraform apply -auto-approve



```
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::248873599897:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATT4QFYOM32JZWKQNI"
}
@Rughma-Malik ➜ ~/Lab13 $ _
```

6. Display the output:

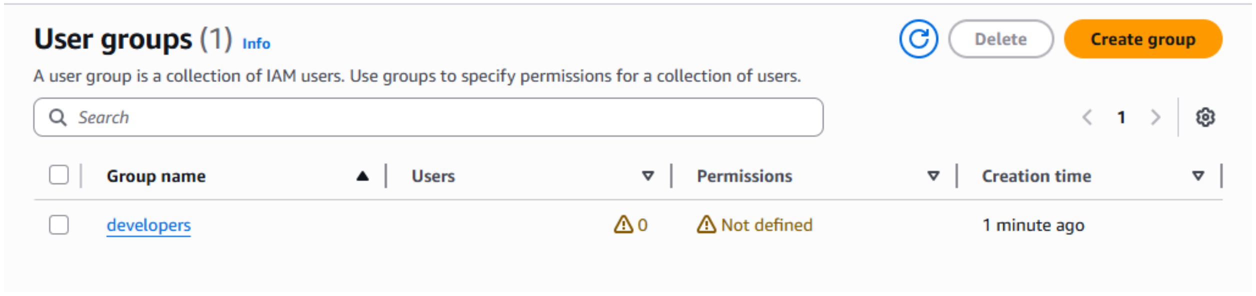
terraform output



```
@Rughma-Malik ➜ ~/Lab13 $ terraform output
group_details = {
  "group_arn" = "arn:aws:iam::248873599897:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATT4QFYOM32JZWKQNI"
}
@Rughma-Malik ➜ ~/Lab13 $ _
```

7. Verify the group in AWS Console:

- Navigate to IAM → Groups in AWS Console



User groups (1)

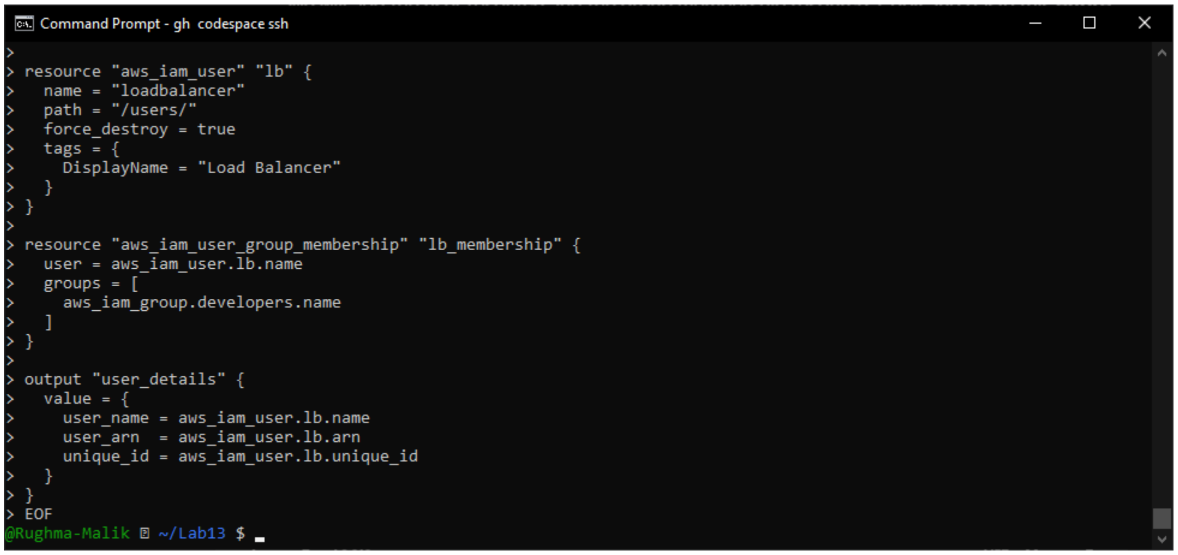
A user group is a collection of IAM users. Use groups to specify permissions for a collection of users.

Group name	Users	Permissions	Creation time
developers	⚠ 0	⚠ Not defined	1 minute ago

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:

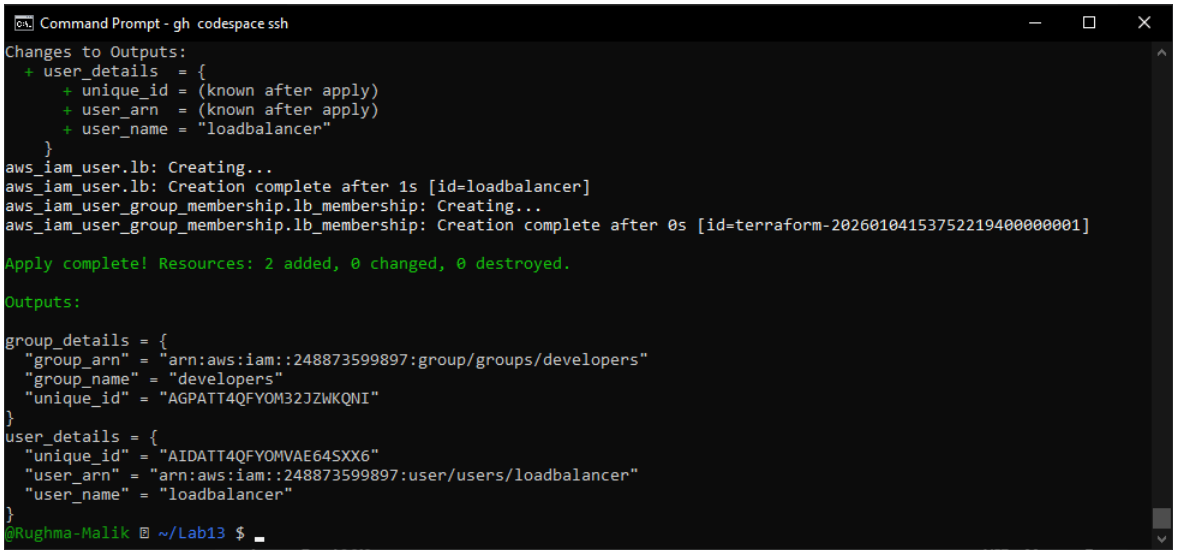


Command Prompt - gh codespace ssh

```
>
> 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
>   }
> }
> EOF
@Rughma-Malik ➜ ~/Lab13 $ 
```

2. Apply the configuration:

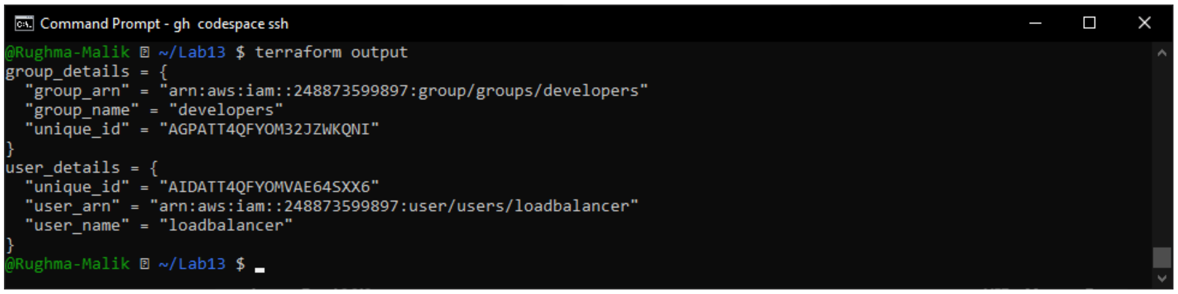
terraform apply -auto-approve



Command Prompt - gh codespace ssh

```
Changes to Outputs:
  + user_details  = {
      + unique_id = (known after apply)
      + user_arn  = (known after apply)
      + user_name = "loadbalancer"
    }
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 0s [id=terraform-20260104153752219400000001]

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

Outputs:

group_details = {
  "group_arn" = "arn:aws:iam::248873599897:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATT4QFYOM32JZWKQNI"
}
user_details = {
  "unique_id" = "AIDATT4QFYOMVAE64SXX6"
  "user_arn" = "arn:aws:iam::248873599897:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@Rughma-Malik ➜ ~/Lab13 $ 
```

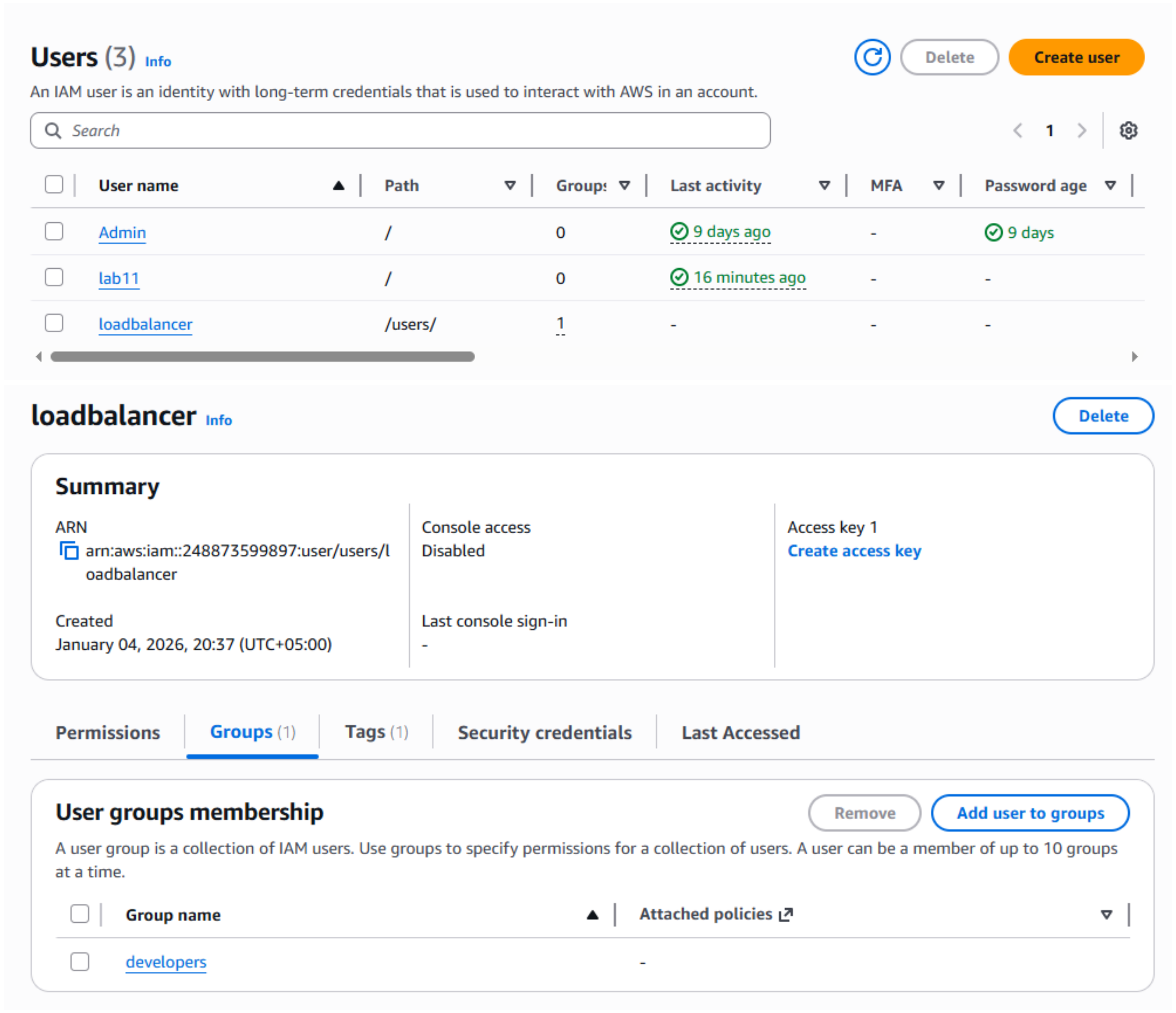
3. Display the outputs:



Command Prompt - gh codespace ssh

```
@Rughma-Malik ➜ ~/Lab13 $ terraform output
group_details = {
  "group_arn" = "arn:aws:iam::248873599897:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATT4QFYOM32JZWKQNI"
}
user_details = {
  "unique_id" = "AIDATT4QFYOMVAE64SXX6"
  "user_arn" = "arn:aws:iam::248873599897:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@Rughma-Malik ➜ ~/Lab13 $ 
```

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



Users (3) Info

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

User name	Path	Groups	Last activity	MFA	Password age
Admin	/	0	9 days ago	-	9 days
lab11	/	0	16 minutes ago	-	-
loadbalancer	/users/	1	-	-	-

loadbalancer Info

Summary

ARN
arn:aws:iam::248873599897:user/users/loadbalancer

Console access
Disabled

Access key 1
Create access key

Created
January 04, 2026, 20:37 (UTC+05:00)

Last console sign-in
-

Permissions | Groups (1) | Tags (1) | Security credentials | Last Accessed

User groups membership

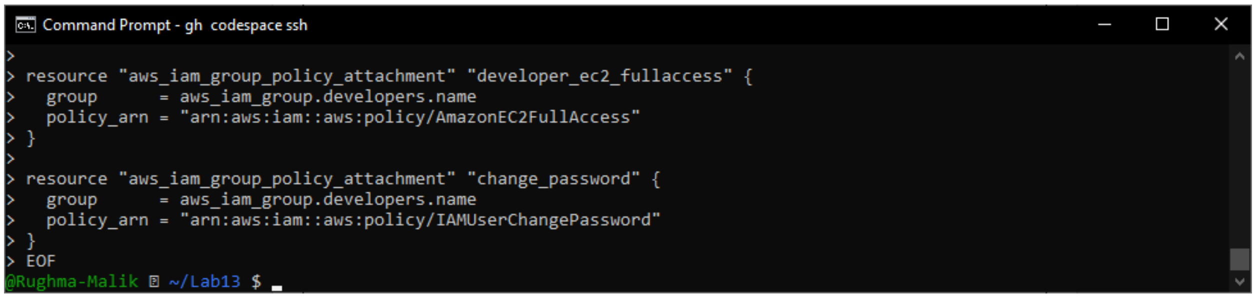
A user group is a collection of IAM users. Use groups to specify permissions for a collection of users. A user can be a member of up to 10 groups at a time.

Group name	Attached policies
developers	-

Task 3 — Attach Policies to IAM Group

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

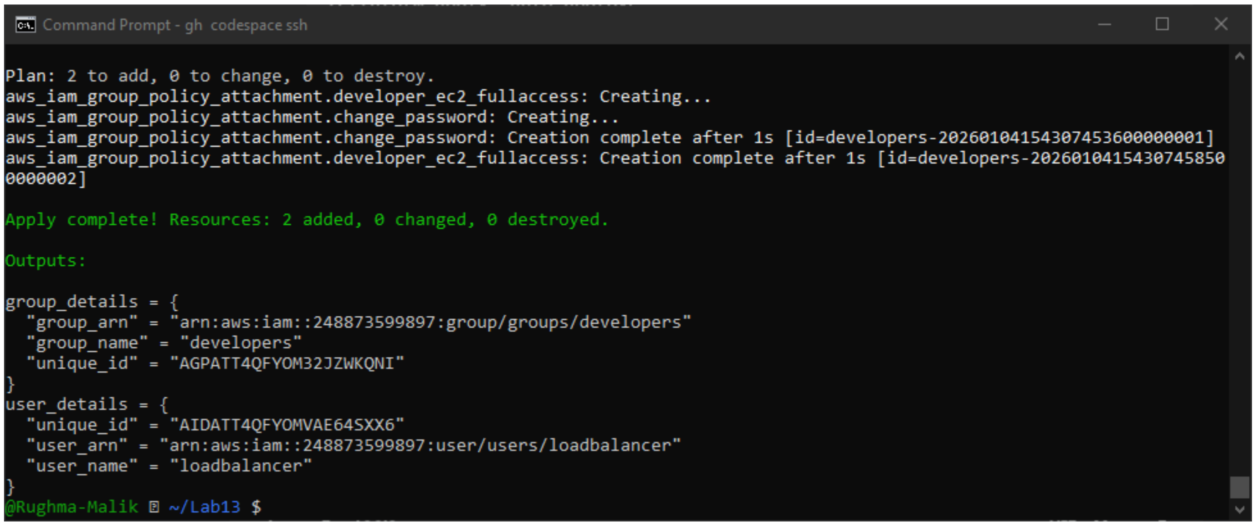
1. Update main.tf to add policy attachments:



```
Command Prompt - gh  codespace ssh
>
> 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"
> }
> EOF
@Rughma-Malik ➜ ~/Lab13 $ _
```

2. Apply the configuration:

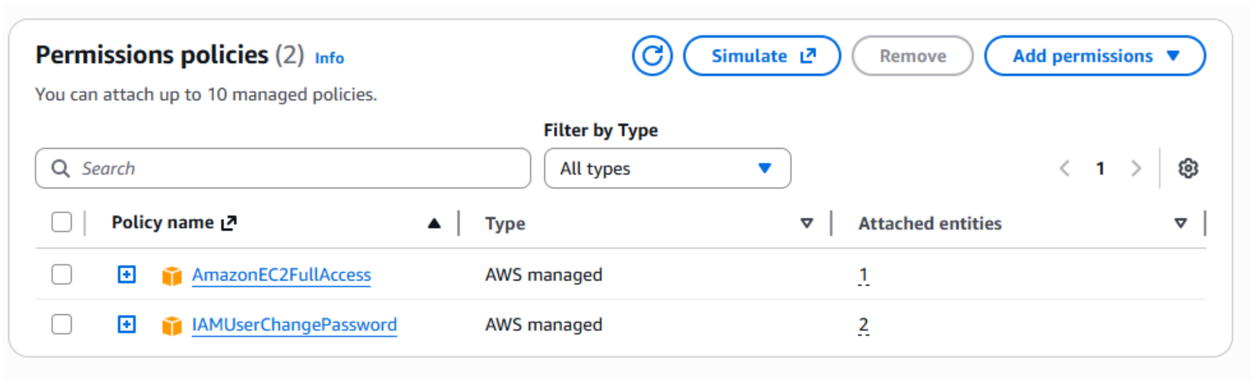
terraform apply -auto-approve



```
Command Prompt - gh  codespace ssh

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-20260104154307453600000001]
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Creation complete after 1s [id=developers-2026010415430745850
0000002]

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

Outputs:

group_details = {
  "group_arn" = "arn:aws:iam::248873599897:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATT4QFYOM32JZWKQNI"
}
user_details = {
  "unique_id" = "AIDATT4QFYOMVAE64SXX6"
  "user_arn" = "arn:aws:iam::248873599897:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@Rughma-Malik ➜ ~/Lab13 $
```

3. Verify policies in AWS Console:

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



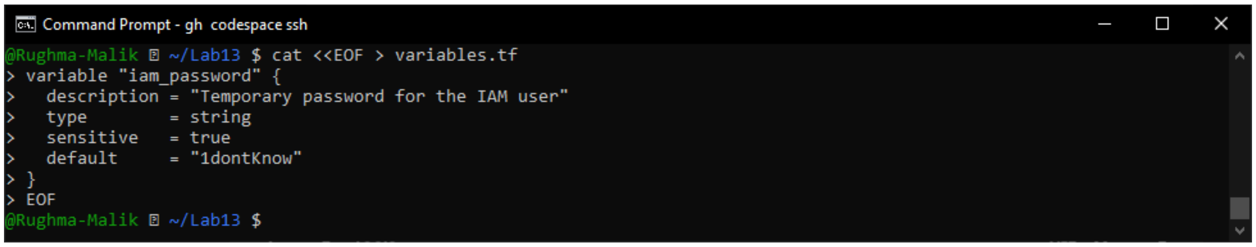
Permissions policies (2) Info
Simulate
Remove
Add permissions
You can attach up to 10 managed policies.
Filter by Type
Search
All types
1

Policy name	Type	Attached entities
AmazonEC2FullAccess	AWS managed	1
IAMUserChangePassword	AWS managed	2

Task 4 — Create Login Profile for IAM User

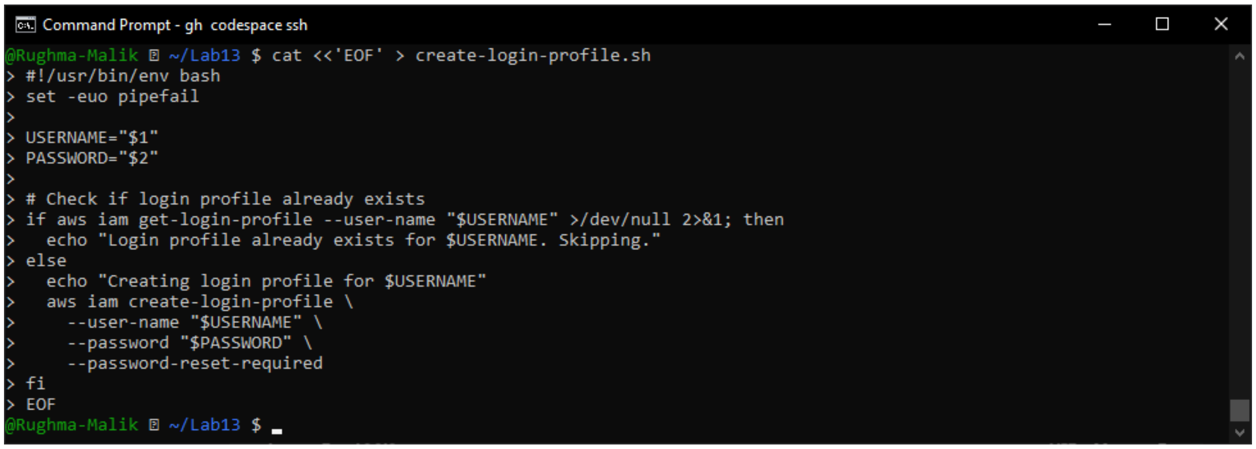
In this task, you will create a login profile for the loadbalancer user using a bash script and null_resource provisioner.

1. Create variables.tf file:



```
Command Prompt - gh  codespace ssh
@Rughma-Malik ➜ ~/Lab13 $ cat <<EOF > variables.tf
> variable "iam_password" {
>   description = "Temporary password for the IAM user"
>   type        = string
>   sensitive   = true
>   default     = "1dontKnow"
> }
> EOF
@Rughma-Malik ➜ ~/Lab13 $
```

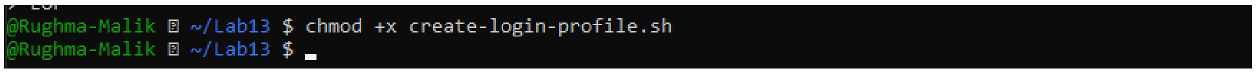
2. Create the bash script create-login-profile.sh:



```
Command Prompt - gh  codespace ssh
@Rughma-Malik ➜ ~/Lab13 $ cat <<'EOF' > 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
> EOF
@Rughma-Malik ➜ ~/Lab13 $ _
```

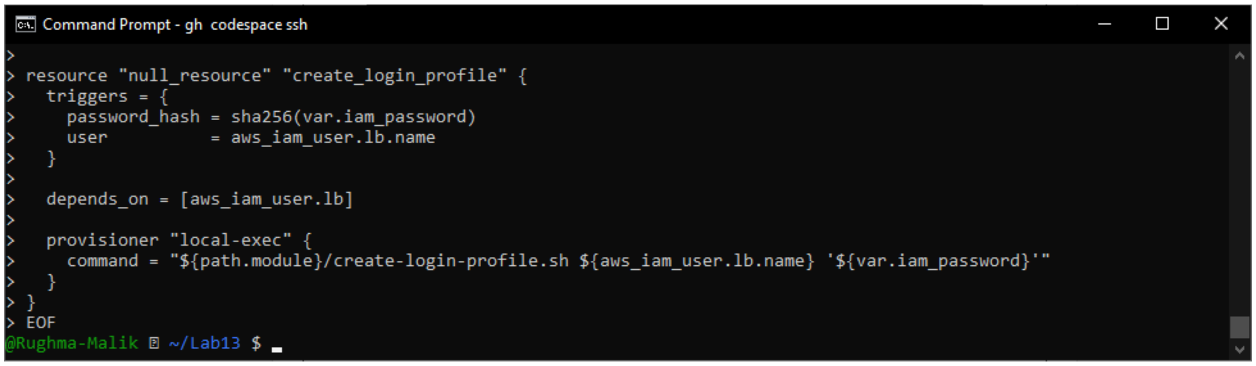
3. Make the script executable:

chmod +x create-login-profile.sh



```
@Rughma-Malik ➜ ~/Lab13 $ chmod +x create-login-profile.sh
@Rughma-Malik ➜ ~/Lab13 $ _
```

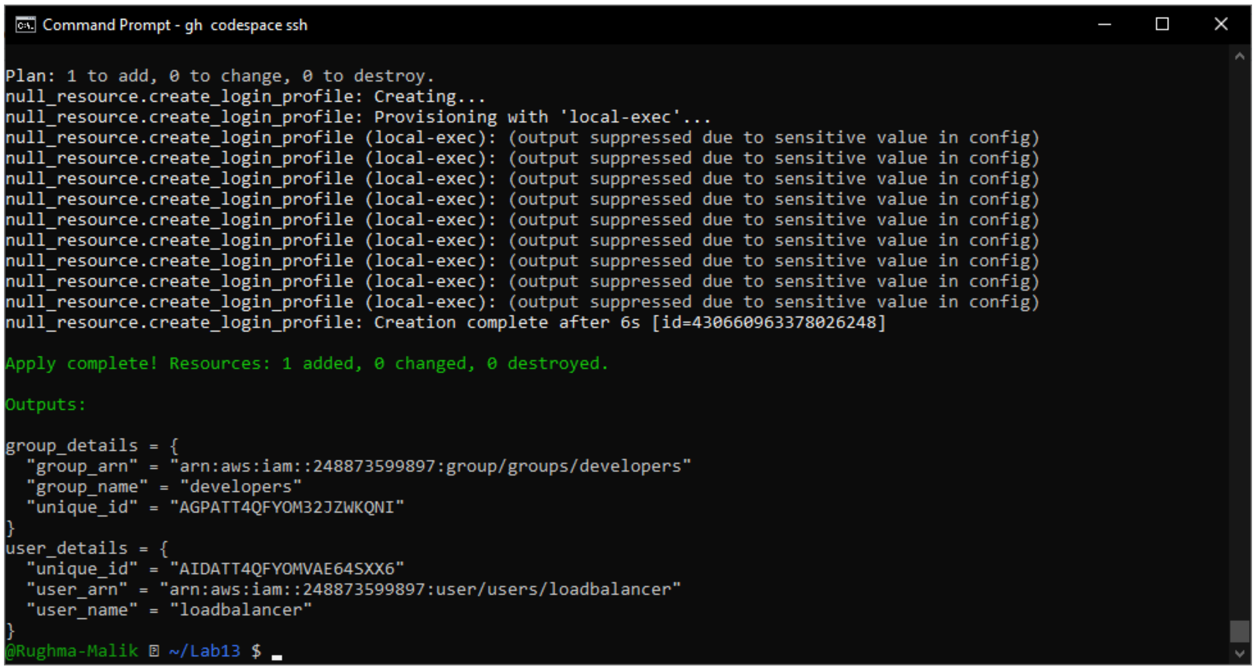
4. Update main.tf to add the null_resource provisioner:



```
Command Prompt - gh  codespace ssh
>
> 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}'"
>   }
> }
> EOF
@Rughma-Malik ➜ ~/Lab13 $ _
```

5. Apply the configuration with a custom password:

terraform apply -auto-approve -var="iam_password=MySecurePass123!"

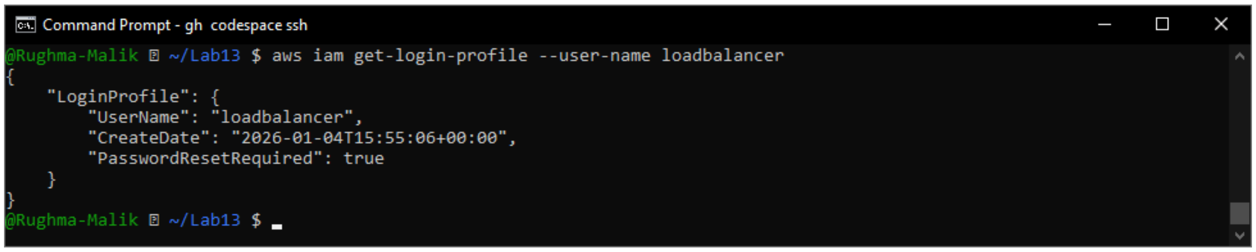


Command Prompt - gh codespace ssh

```
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)
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)
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)
null_resource.create_login_profile: Creation complete after 6s [id=430660963378026248]

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

Outputs:

group_details = {
  "group_arn" = "arn:aws:iam::248873599897:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATT4QFYOM32JZWKQNI"
}
user_details = {
  "unique_id" = "AIDATT4QFYOMVAE64SXX6"
  "user_arn" = "arn:aws:iam::248873599897:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@Rughma-Malik ➜ ~/Lab13 $ _
```

6. Verify login profile creation:

aws iam get-login-profile --user-name loadbalancer

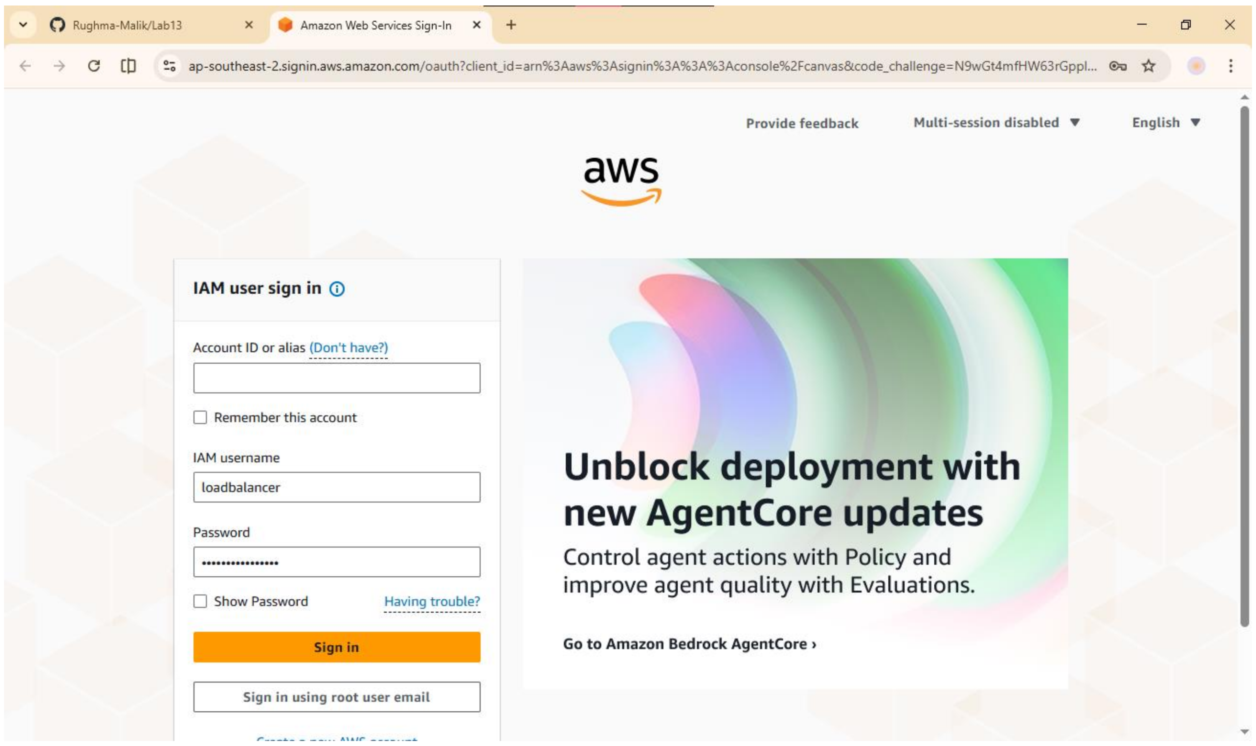


Command Prompt - gh codespace ssh

```
@Rughma-Malik ➜ ~/Lab13 $ aws iam get-login-profile --user-name loadbalancer
{
    "LoginProfile": {
        "UserName": "loadbalancer",
        "CreateDate": "2026-01-04T15:55:06+00:00",
        "PasswordResetRequired": true
    }
}
@Rughma-Malik ➜ ~/Lab13 $ _
```

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


Rughma-Malik/Lab13

Amazon Web Services Sign-In

ap-southeast-2.signin.aws.amazon.com/oauth?client_id=arn%3Aaws%3Asignin%3A%3A%3Aconsole%2Fcanvas&code_challenge=N9wGt4mfHW63rGppl...

Provide feedback | Multi-session disabled | English

IAM user sign in

Account ID or alias (Don't have?)

Remember this account

IAM username: loadbalancer

Password: ••••••••••••••••

Show Password | Having trouble?

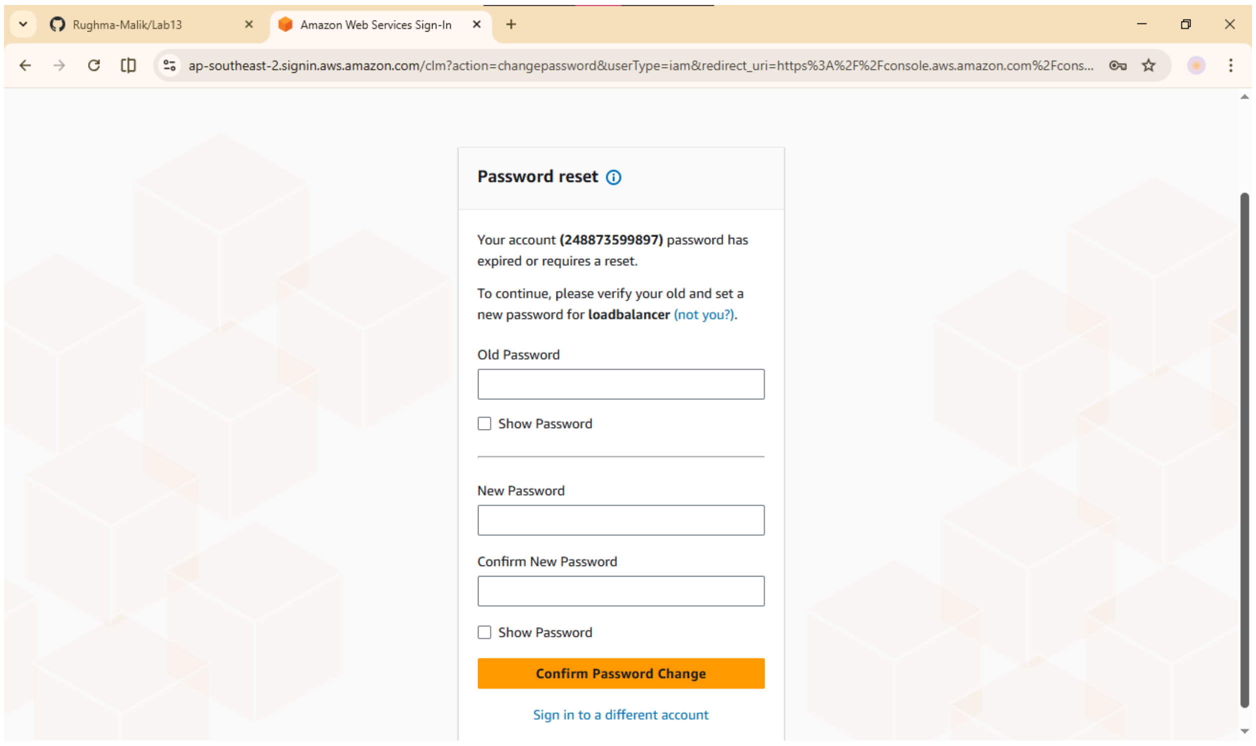
Sign in

Sign in using root user email

Unblock deployment with new AgentCore updates

Control agent actions with Policy and improve agent quality with Evaluations.

Go to Amazon Bedrock AgentCore ›


Rughma-Malik/Lab13

Amazon Web Services Sign-In

ap-southeast-2.signin.aws.amazon.com/clm?action=changepassword&userType=iam&redirect_uri=https%3A%2F%2Fconsole.aws.amazon.com%2Fcons...

Password reset

Your account **(248873599897)** password has expired or requires a reset.

To continue, please verify your old and set a new password for **loadbalancer** (not you?).

Old Password

Show Password

New Password

Confirm New Password

Show Password

Confirm Password Change

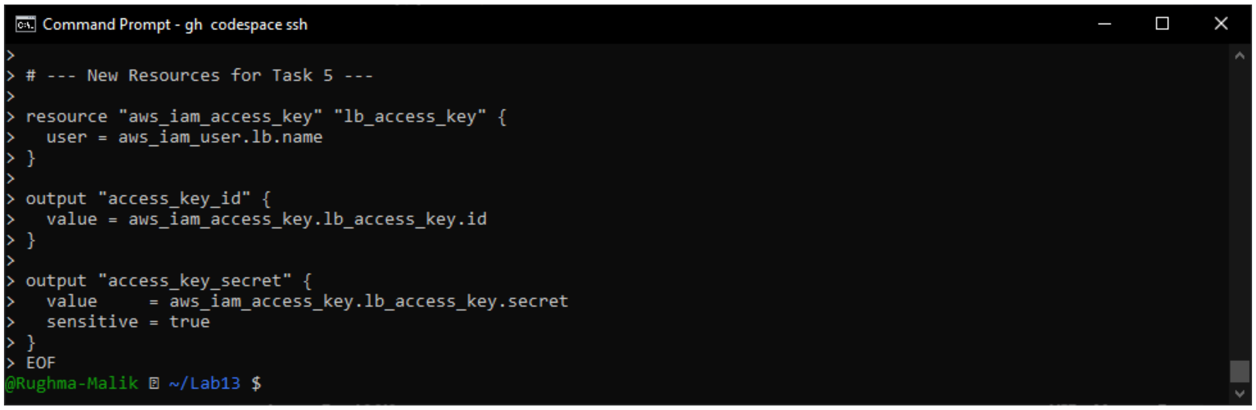
Sign in to a different account

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:

Add these resources:

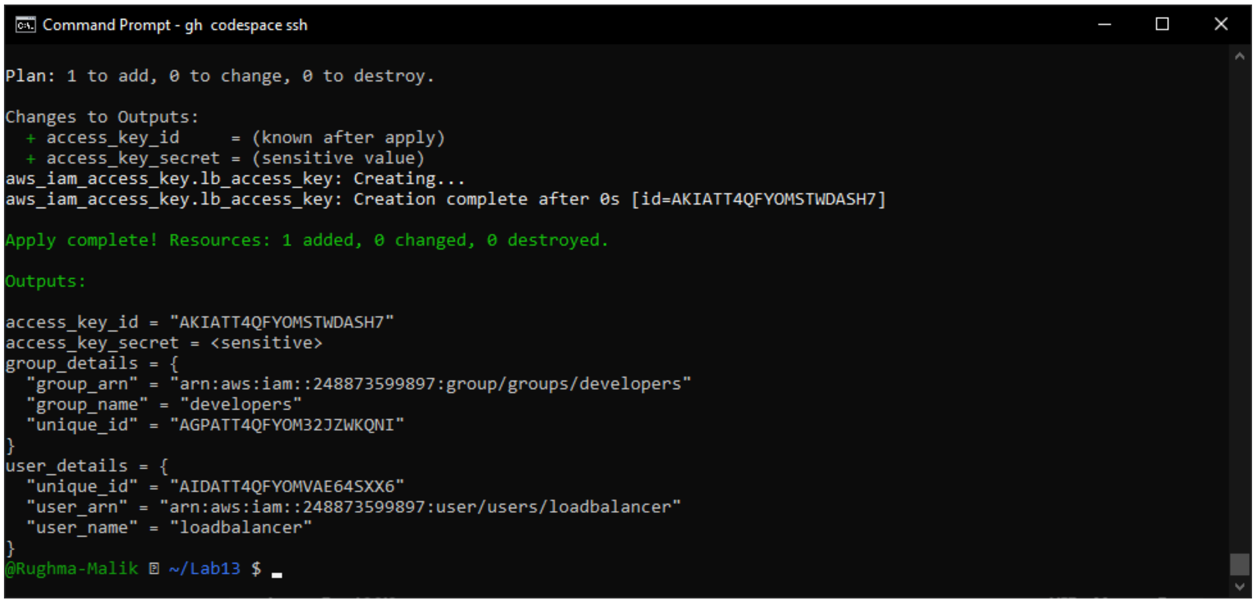


Command Prompt - gh codespace ssh

```
>
> # --- New Resources for Task 5 ---
>
> 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
> }
> EOF
@Rughma-Malik ➜ ~/Lab13 $
```

2. Apply the configuration:

terraform apply -auto-approve -var="iam_password=MySecurePass123!"

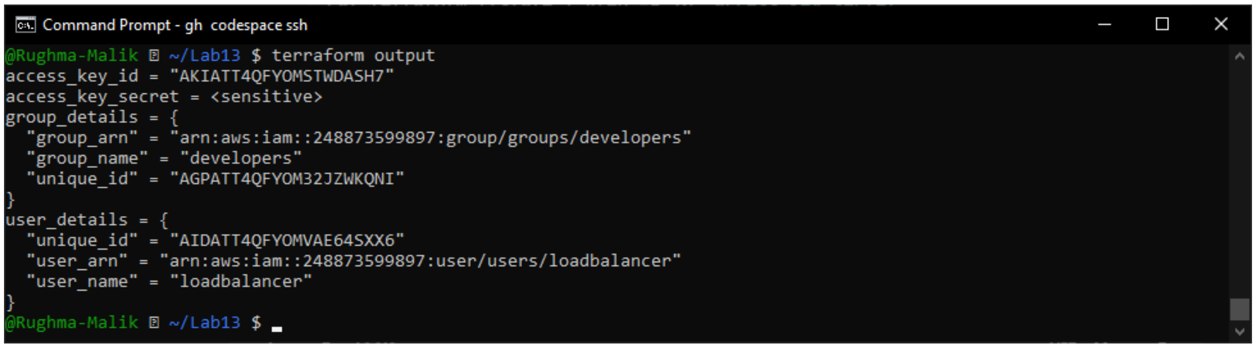


Command Prompt - gh codespace ssh

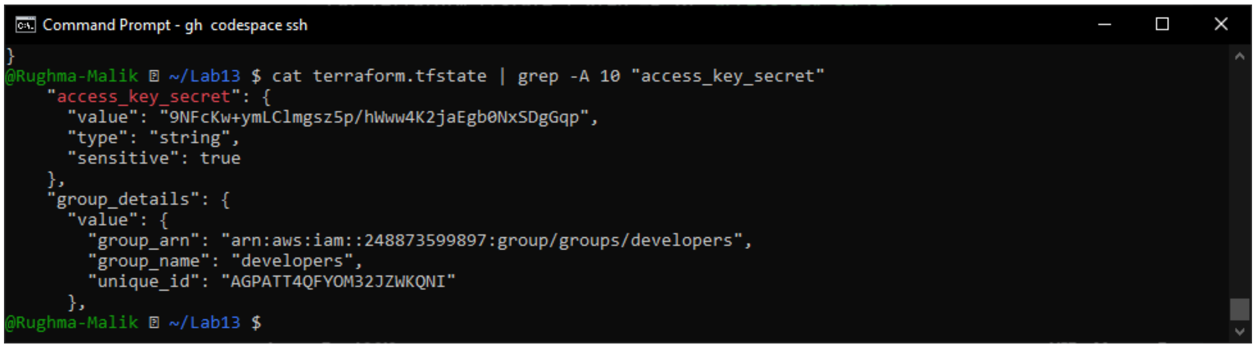
```
Plan: 1 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + access_key_id     = (known after apply)
  + access_key_secret = (sensitive value)
aws_iam_access_key.lb_access_key: Creating...
aws_iam_access_key.lb_access_key: Creation complete after 0s [id=AKIATT4QFYOMSTWDASH7]

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

Outputs:

access_key_id = "AKIATT4QFYOMSTWDASH7"
access_key_secret = <sensitive>
group_details = {
  "group_arn" = "arn:aws:iam::248873599897:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATT4QFYOM32JZWKQNI"
}
user_details = {
  "unique_id" = "AIDATT4QFYOMVAE64SXX6"
  "user_arn" = "arn:aws:iam::248873599897:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@Rughma-Malik ➜ ~/Lab13 $ _
```

3. Display outputs:

terraform output



```
Command Prompt - gh  codespace ssh
@Rughma-Malik ➜ ~/Lab13 $ terraform output
access_key_id = "AKIATT4QFYOMSTWDASH7"
access_key_secret = <sensitive>
group_details = {
  "group_arn" = "arn:aws:iam::248873599897:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATT4QFYOM32JZWKQNI"
}
user_details = {
  "unique_id" = "AIDATT4QFYOMVAE64SXX6"
  "user_arn" = "arn:aws:iam::248873599897:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@Rughma-Malik ➜ ~/Lab13 $ _
```



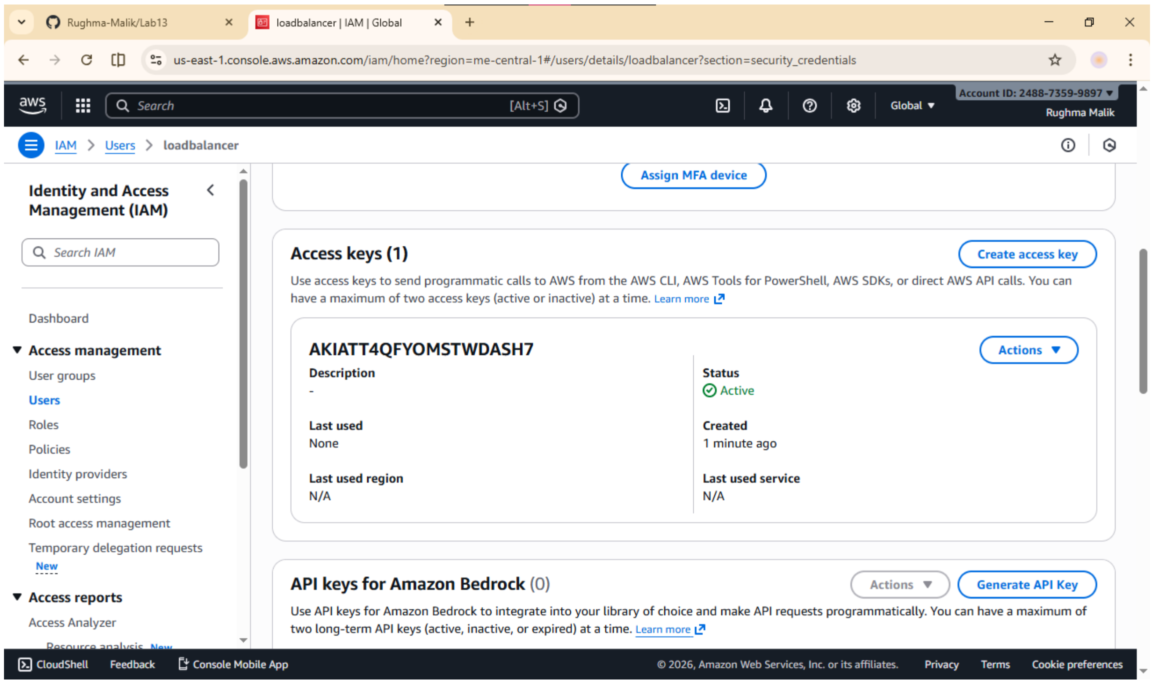
```
Command Prompt - gh  codespace ssh
}
@Rughma-Malik ➜ ~/Lab13 $ cat terraform.tfstate | grep -A 10 "access_key_secret"
    "access_key_secret": {
      "value": "9NFcKw+ymLClmgsz5p/hWww4K2jaEgb0NxSDgGqp",
      "type": "string",
      "sensitive": true
    },
    "group_details": {
      "value": {
        "group_arn": "arn:aws:iam::248873599897:group/groups/developers",
        "group_name": "developers",
        "unique_id": "AGPATT4QFYOM32JZWKQNI"
      },
@Rughma-Malik ➜ ~/Lab13 $
```

4. View the secret in terraform state:

cat terraform.tfstate | grep -A 10 "access_key_secret"

5. Verify access key in AWS Console:

   - Navigate to IAM → Users → loadbalancer → Security credentials



us-east-1.console.aws.amazon.com/iam/home?region=me-central-1#/users/details/loadbalancer?section=security_credentials

IAM > Users > loadbalancer

Access keys (1)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. Learn more

AKIATT4QFYOMSTWDASH7

Description: -	Status: Active
Last used: None	Created: 1 minute ago
Last used region: N/A	Last used service: N/A

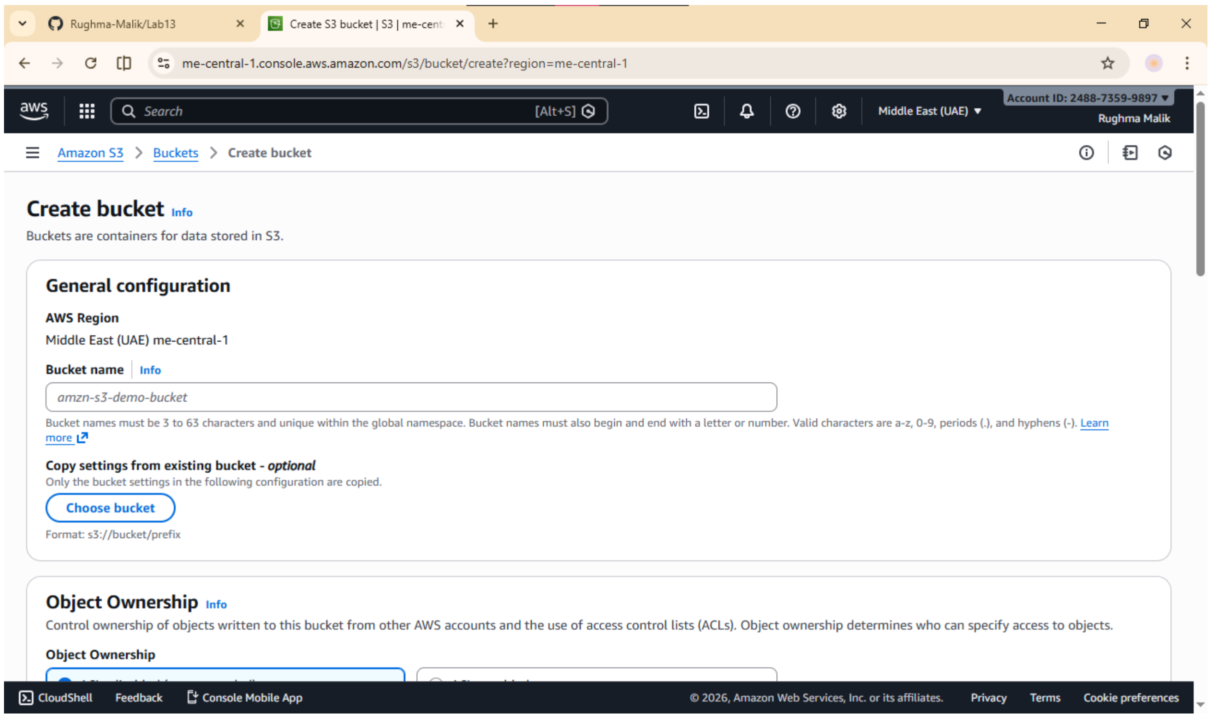
API keys for Amazon Bedrock (0)

Use API keys for Amazon Bedrock to integrate into your library of choice and make API requests programmatically. You can have a maximum of two long-term API keys (active, inactive, or expired) at a time. Learn more

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:



me-central-1.console.aws.amazon.com/s3/bucket/create?region=me-central-1

Account ID: 2488-7359-9897
Rughma Malik

Amazon S3 > Buckets > Create bucket

Create bucket

Buckets are containers for data stored in S3.

General configuration

AWS Region
Middle East (UAE) me-central-1

Bucket name
amzn-s3-demo-bucket

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

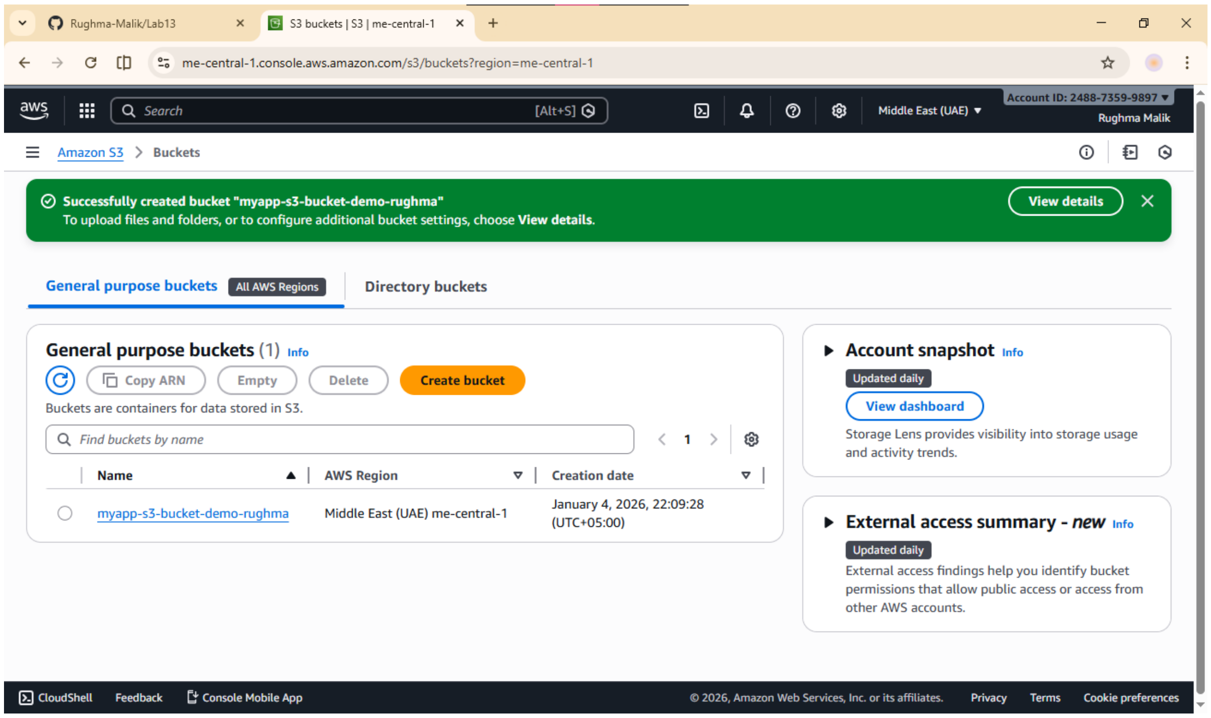
Object Ownership

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

Object Ownership

© 2026, Amazon Web Services, Inc. or its affiliates.

- Click "Create bucket"



me-central-1.console.aws.amazon.com/s3/buckets?region=me-central-1

Amazon S3 > Buckets

Successfully created bucket "myapp-s3-bucket-demo-rughma"
To upload files and folders, or to configure additional bucket settings, choose **View details**.

General purpose buckets (1)

Buckets are containers for data stored in S3.

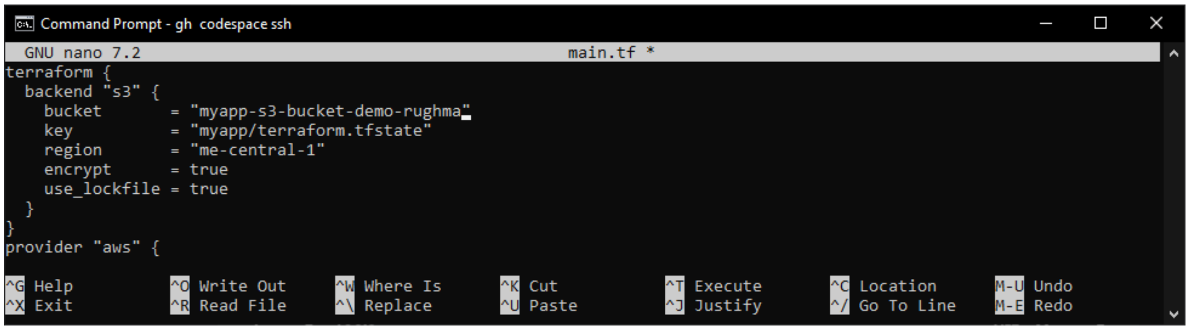
Name	AWS Region	Creation date
myapp-s3-bucket-demo-rughma	Middle East (UAE) me-central-1	January 4, 2026, 22:09:28 (UTC+05:00)

Account snapshot
Updated daily
Storage Lens provides visibility into storage usage and activity trends.

External access summary - *new*
Updated daily
External access findings help you identify bucket permissions that allow public access or access from other AWS accounts.

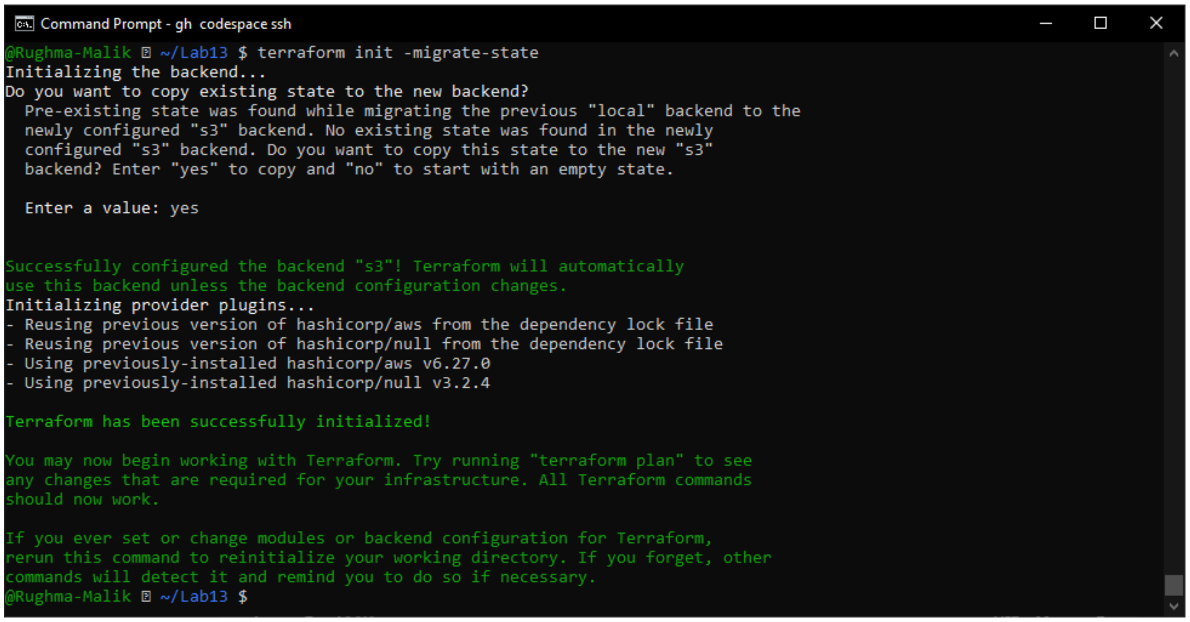
© 2026, Amazon Web Services, Inc. or its affiliates.

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



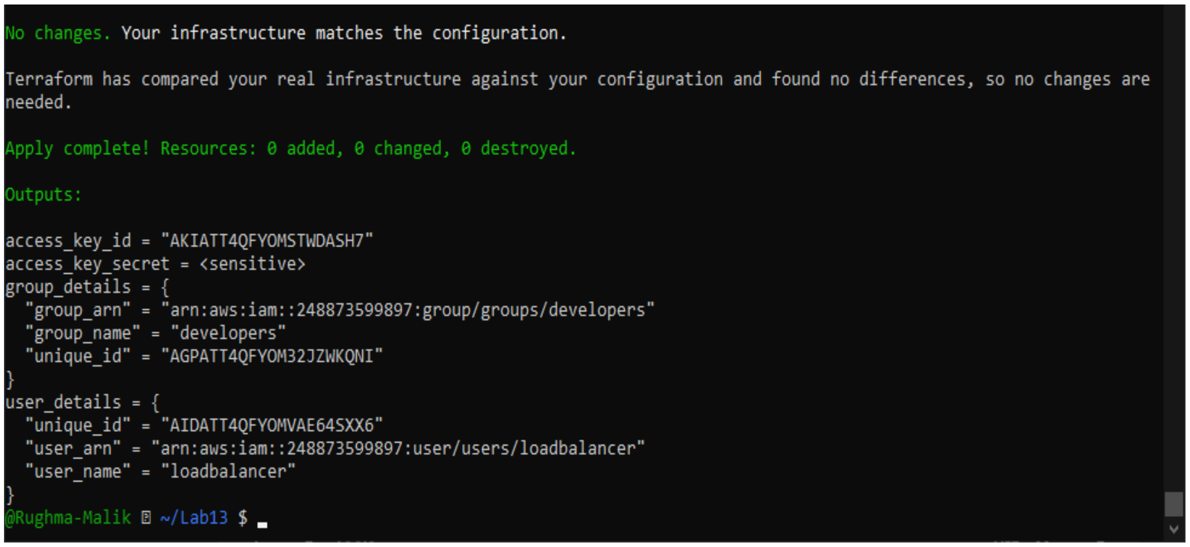
```
Command Prompt - gh  codespace ssh
  GNU nano 7.2                                    main.tf *
terraform {
  backend "s3" {
    bucket       = "myapp-s3-bucket-demo-rughma"
    key          = "myapp/terraform.tfstate"
    region       = "me-central-1"
    encrypt      = true
    use_lockfile = true
  }
}
provider "aws" {

^G Help          ^O Write Out     ^W Where Is      ^K Cut           ^T Execute       ^C Location      M-U Undo
^X Exit          ^R Read File     ^\ Replace       ^U Paste         ^J Justify       ^/ Go To Line    M-E Redo
```

3. Reinitialize Terraform with the backend:



```
Command Prompt - gh  codespace ssh
@Rughma-Malik ➜ ~/Lab13 $ 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


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/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.
@Rughma-Malik ➜ ~/Lab13 $
```

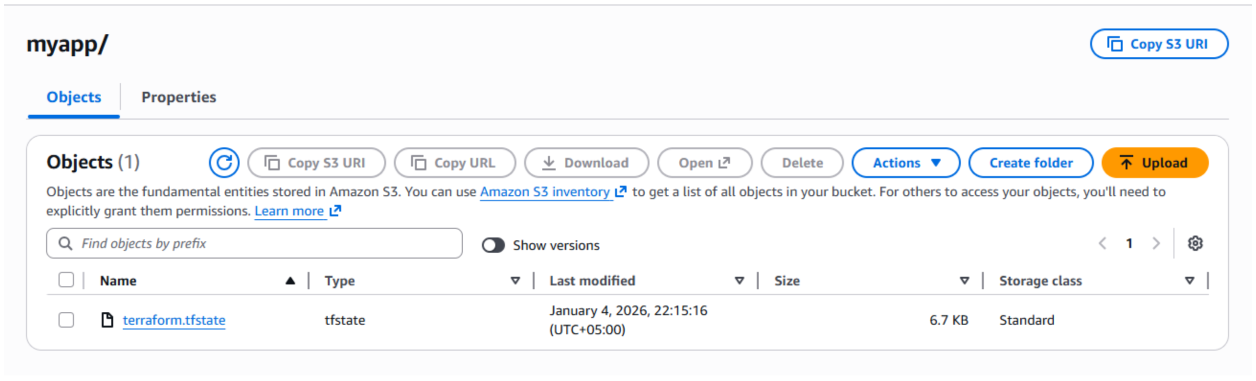
4. Apply the configuration:



```
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.

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

Outputs:

access_key_id = "AKIATT4QFYOMSTWDASH7"
access_key_secret = <sensitive>
group_details = {
  "group_arn" = "arn:aws:iam::248873599897:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPATT4QFYOM32JZWKQNI"
}
user_details = {
  "unique_id" = "AIDATT4QFYOMVAE64SXX6"
  "user_arn" = "arn:aws:iam::248873599897:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@Rughma-Malik ➜ ~/Lab13 $
```

5. Verify state file in S3:

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



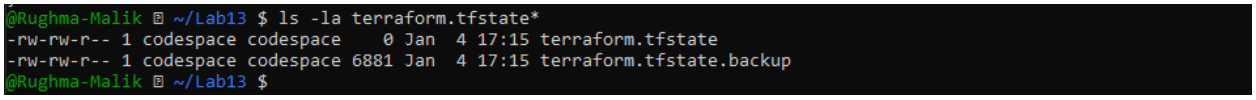
myapp/

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
terraform.tfstate	tfstate	January 4, 2026, 22:15:16 (UTC+05:00)	6.7 KB	Standard

6. Check local state file:

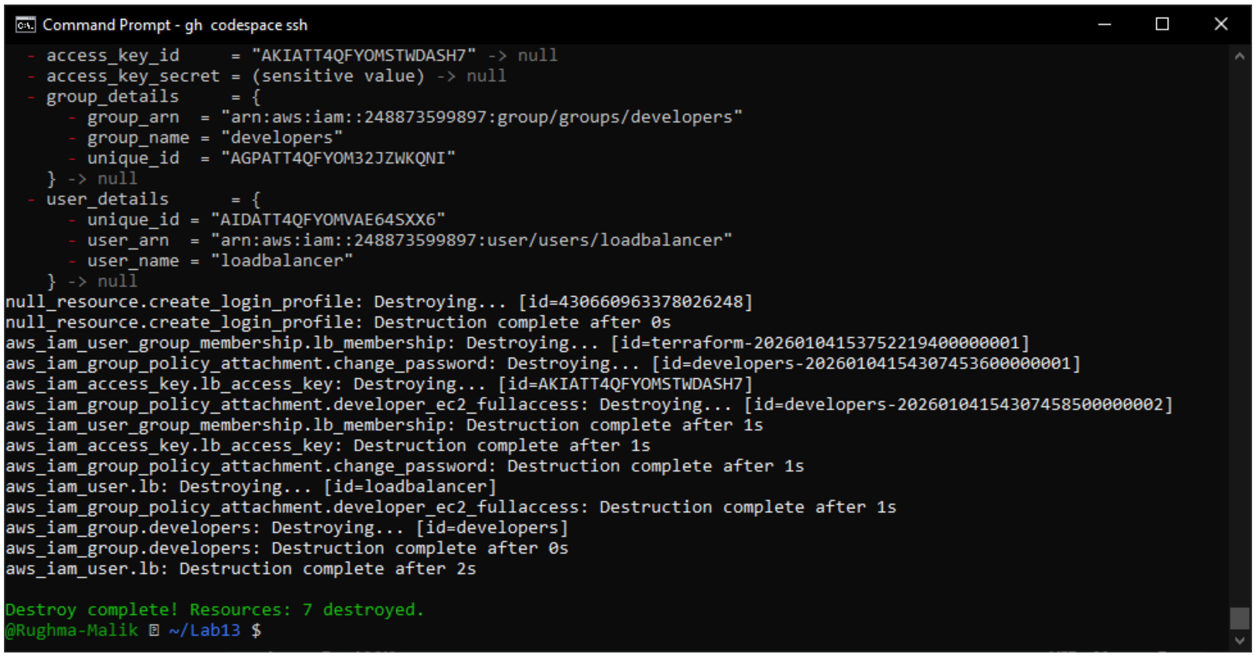
ls -la terraform.tfstate*



```
@Rughma-Malik ➜ ~/Lab13 $ ls -la terraform.tfstate*
-rw-rw-r-- 1 codespace codespace    0 Jan  4 17:15 terraform.tfstate
-rw-rw-r-- 1 codespace codespace 6881 Jan  4 17:15 terraform.tfstate.backup
@Rughma-Malik ➜ ~/Lab13 $
```

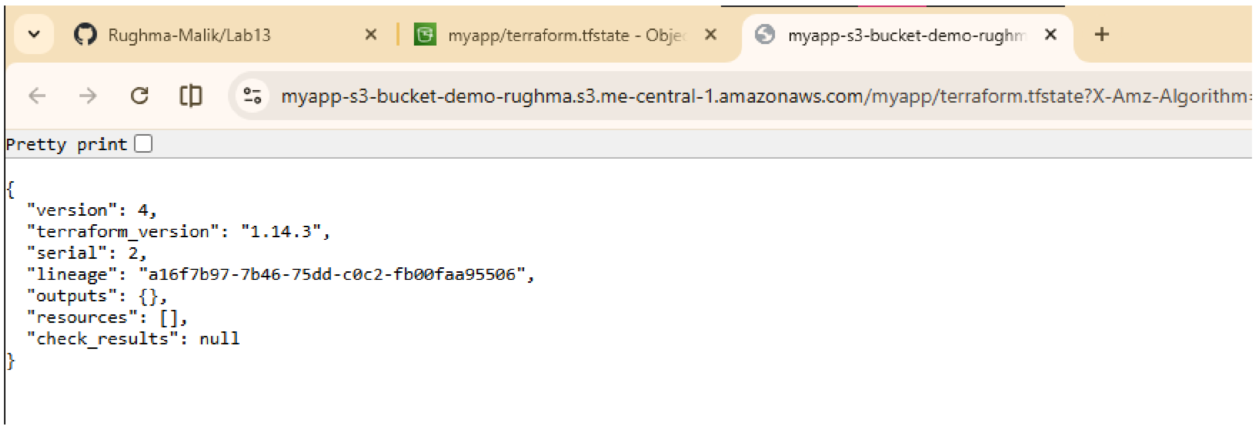
7. Destroy resources and verify state change:

terraform destroy -auto-approve



```
  - access_key_id     = "AKIATT4QFYOMSTWDASH7" -> null
  - access_key_secret = (sensitive value) -> null
  - group_details     = {
      - group_arn  = "arn:aws:iam::248873599897:group/groups/developers"
      - group_name = "developers"
      - unique_id  = "AGPATT4QFYOM32JZWKQNI"
    } -> null
  - user_details      = {
      - unique_id = "AIDATT4QFYOMVAE64SXX6"
      - user_arn  = "arn:aws:iam::248873599897:user/users/loadbalancer"
      - user_name = "loadbalancer"
    } -> null
null_resource.create_login_profile: Destroying... [id=4306609633780262248]
null_resource.create_login_profile: Destruction complete after 0s
aws_iam_user_group_membership.lb_membership: Destroying... [id=terraform-20260104153752219400000001]
aws_iam_group_policy_attachment.change_password: Destroying... [id=developers-20260104154307453600000001]
aws_iam_access_key.lb_access_key: Destroying... [id=AKIATT4QFYOMSTWDASH7]
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Destroying... [id=developers-20260104154307458500000002]
aws_iam_user_group_membership.lb_membership: Destruction complete after 1s
aws_iam_access_key.lb_access_key: Destruction complete after 1s
aws_iam_group_policy_attachment.change_password: Destruction complete after 1s
aws_iam_user.lb: Destroying... [id=loadbalancer]
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Destruction complete after 1s
aws_iam_group.developers: Destroying... [id=developers]
aws_iam_group.developers: Destruction complete after 0s
aws_iam_user.lb: Destruction complete after 2s

Destroy complete! Resources: 7 destroyed.
@Rughma-Malik ➜ ~/Lab13 $
```

8. Verify updated state in S3:



Rughma-Malik/Lab13 | myapp/terraform.tfstate - Obje | myapp-s3-bucket-demo-rughm

myapp-s3-bucket-demo-rughma.s3.me-central-1.amazonaws.com/myapp/terraform.tfstate?X-Amz-Algorithm

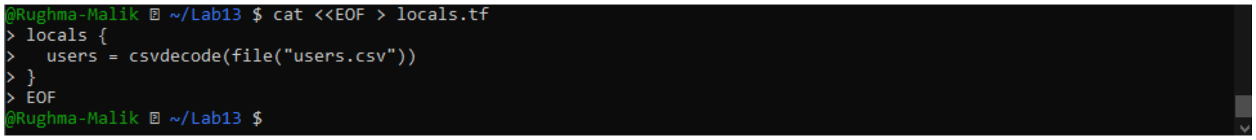
Pretty print

```
{
  "version": 4,
  "terraform_version": "1.14.3",
  "serial": 2,
  "lineage": "a16f7b97-7b46-75dd-c0c2-fb00faa95506",
  "outputs": {},
  "resources": [],
  "check_results": null
}
```

Task 7 — Create Multiple Users from CSV File

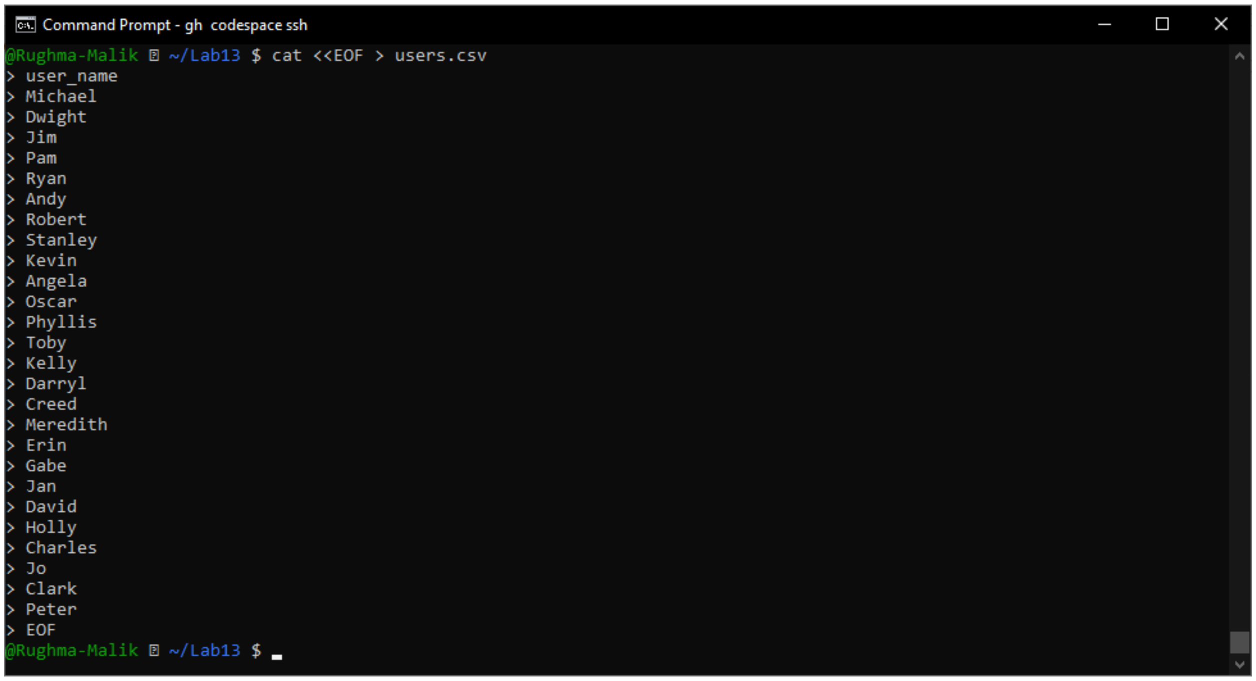
In this task, you will create multiple IAM users dynamically from a CSV file.

1. Create locals.tf file:



```
@Rughma-Malik ➜ ~/Lab13 $ cat <<EOF > locals.tf
> locals {
>   users = csvdecode(file("users.csv"))
> }
> EOF
@Rughma-Malik ➜ ~/Lab13 $
```

2. Create users.csv file:

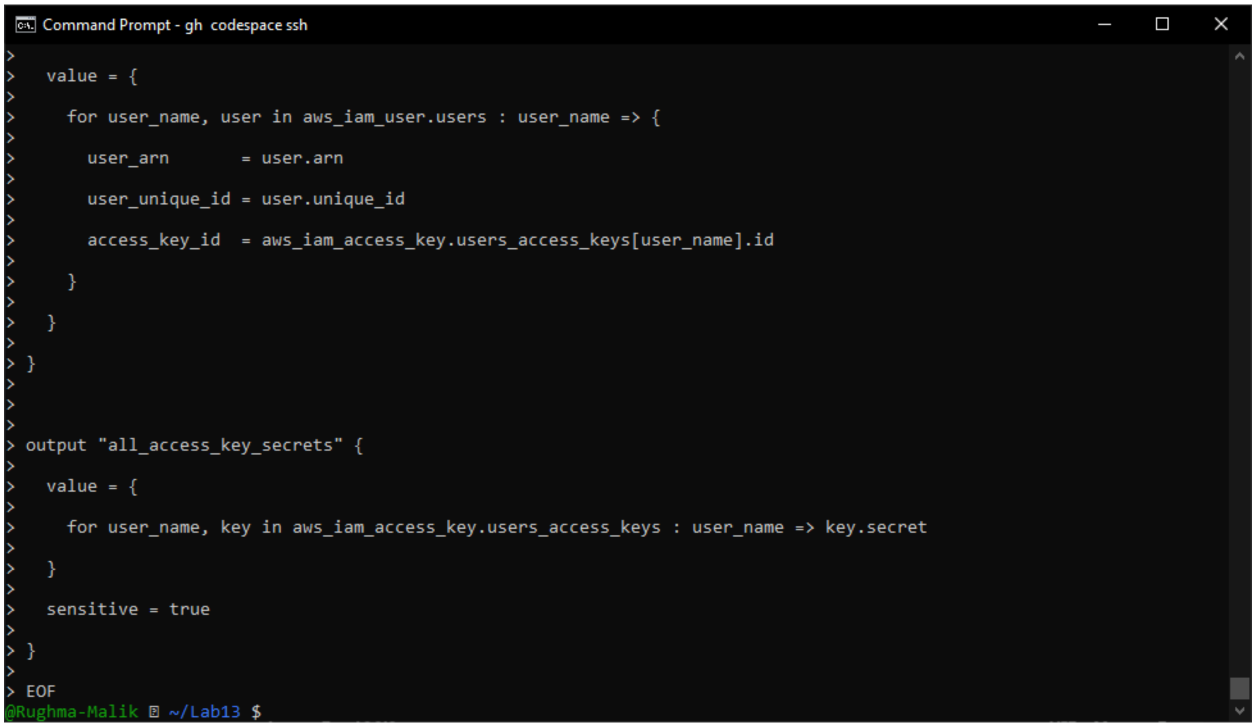


Command Prompt - gh codespace ssh

```
@Rughma-Malik ➜ ~/Lab13 $ cat <<EOF > users.csv
> user_name
> Michael
> Dwight
> Jim
> Pam
> Ryan
> Andy
> Robert
> Stanley
> Kevin
> Angela
> Oscar
> Phyllis
> Toby
> Kelly
> Darryl
> Creed
> Meredith
> Erin
> Gabe
> Jan
> David
> Holly
> Charles
> Jo
> Clark
> Peter
> EOF
@Rughma-Malik ➜ ~/Lab13 $ _
```

3. Update main.tf to create multiple users:

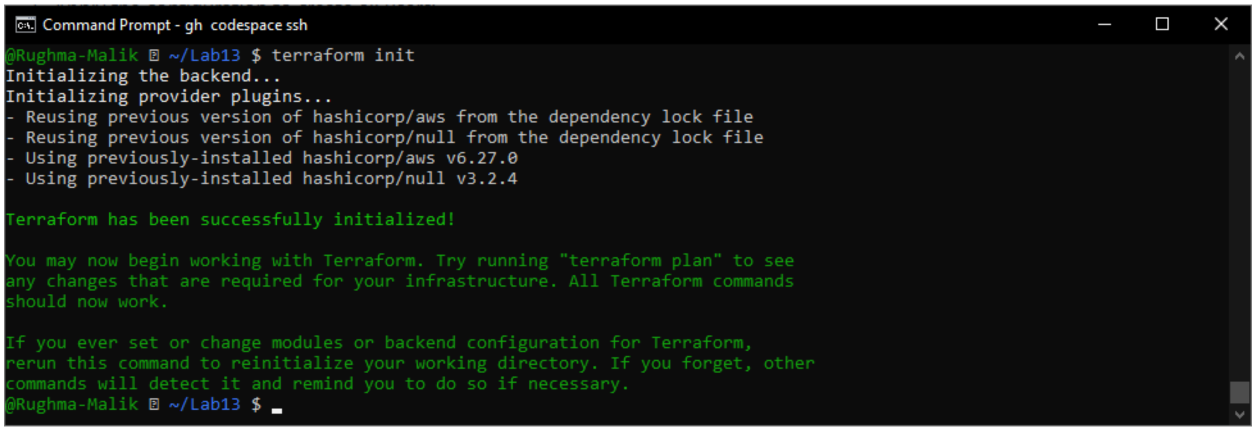
Replace the single user resources with:



```
>
>   value = {
>
>     for user_name, user in aws_iam_user.users : user_name => {
>
>       user_arn       = user.arn
>
>       user_unique_id = user.unique_id
>
>       access_key_id  = aws_iam_access_key.users_access_keys[user_name].id
>
>     }
>
>   }
>
> }
>
>
>
> output "all_access_key_secrets" {
>
>   value = {
>
>     for user_name, key in aws_iam_access_key.users_access_keys : user_name => key.secret
>
>   }
>
>   sensitive = true
>
> }
>
> EOF
@Rughma-Malik ➜ ~/Lab13 $
```

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

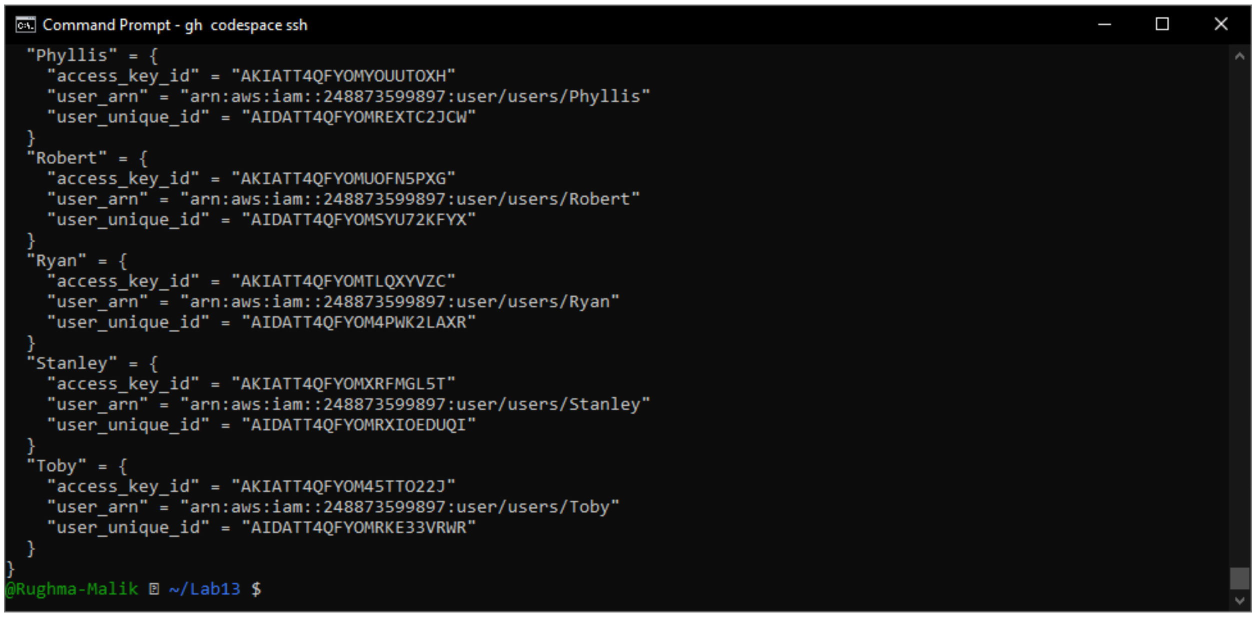
terraform init



```
@Rughma-Malik ➜ ~/Lab13 $ 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.
@Rughma-Malik ➜ ~/Lab13 $ _
```

5. Apply the configuration to create all users:

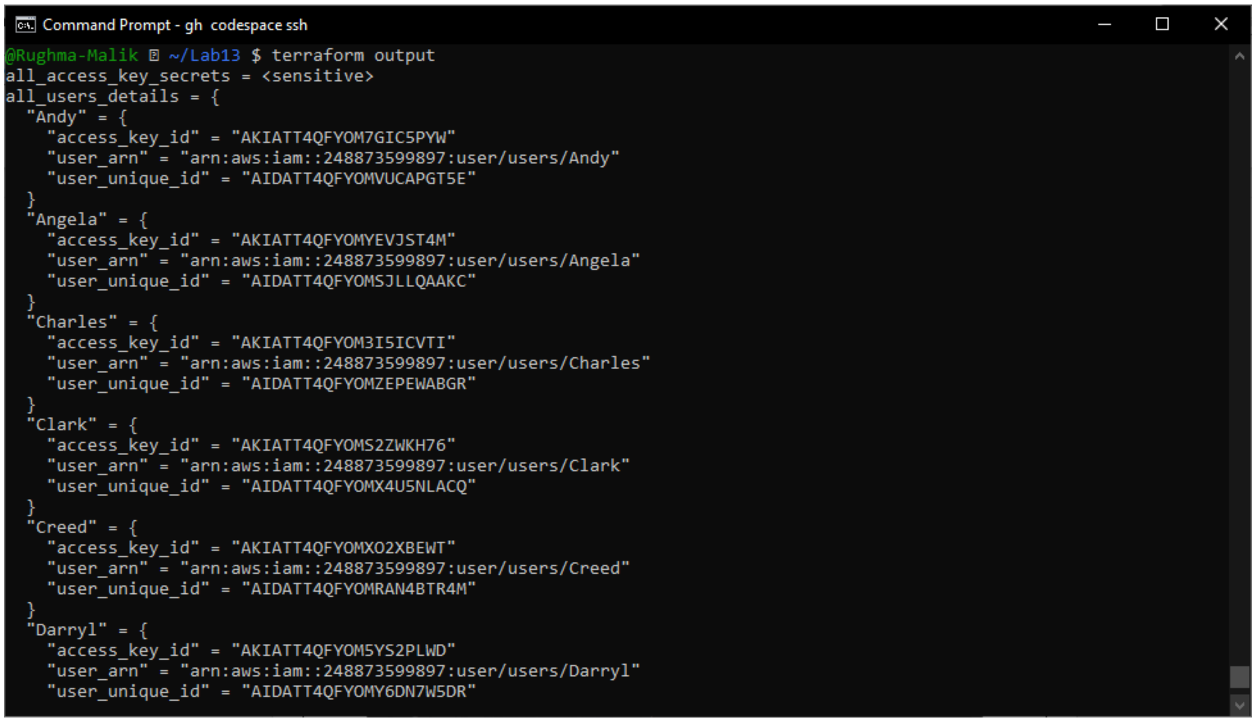
terraform apply -auto-approve -var="iam_password=MySecurePass123!"



```
  "Phyllis" = {
    "access_key_id" = "AKIATT4QFYOMYOUUTOXH"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Phyllis"
    "user_unique_id" = "AIDATT4QFYOMREXTC2JCW"
  }
  "Robert" = {
    "access_key_id" = "AKIATT4QFYOMUOFN5PXG"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Robert"
    "user_unique_id" = "AIDATT4QFYOMSYU72KFYX"
  }
  "Ryan" = {
    "access_key_id" = "AKIATT4QFYOMTLQXYVZC"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Ryan"
    "user_unique_id" = "AIDATT4QFYOM4PWK2LAXR"
  }
  "Stanley" = {
    "access_key_id" = "AKIATT4QFYOMXRFMGL5T"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Stanley"
    "user_unique_id" = "AIDATT4QFYOMRXIOEDUQI"
  }
  "Toby" = {
    "access_key_id" = "AKIATT4QFYOM45TTO22J"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Toby"
    "user_unique_id" = "AIDATT4QFYOMRKE33VRWR"
  }
}
@Rughma-Malik ➜ ~/Lab13 $
```

6. Display the outputs:

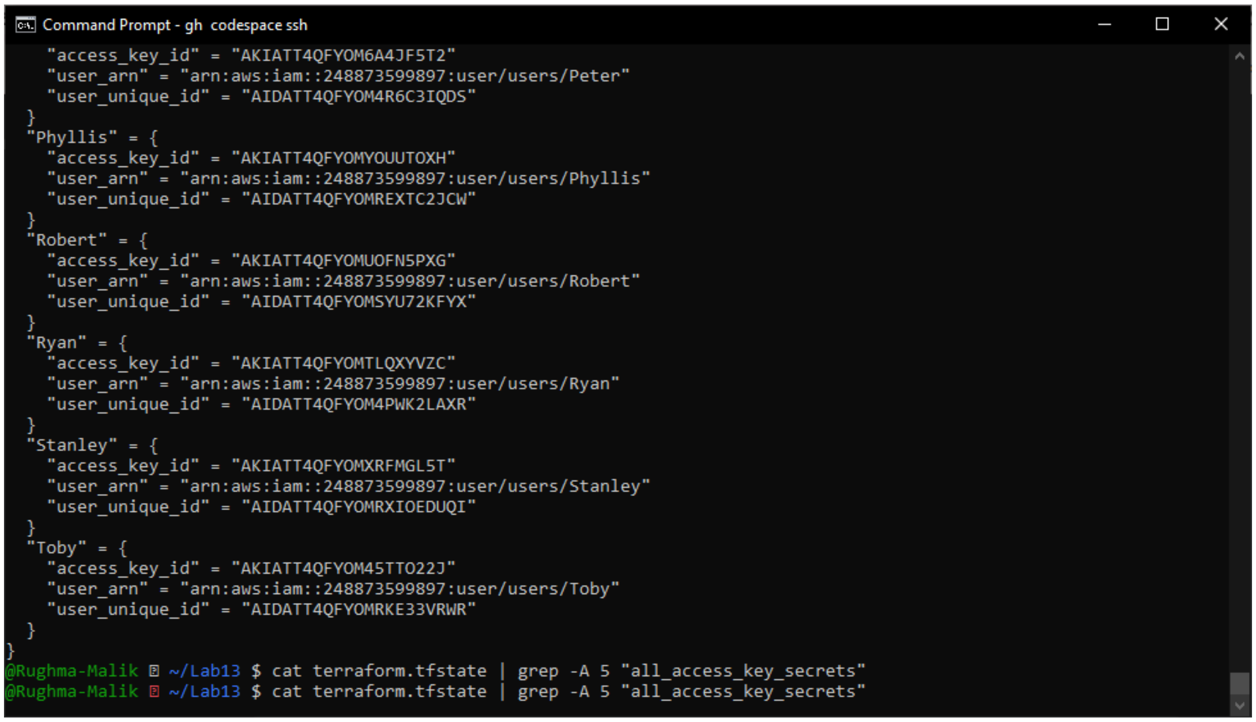
terraform output



```
@Rughma-Malik ➜ ~/Lab13 $ terraform output
all_access_key_secrets = <sensitive>
all_users_details = {
  "Andy" = {
    "access_key_id" = "AKIATT4QFYOM7GIC5PYW"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Andy"
    "user_unique_id" = "AIDATT4QFYOMVUCAPGT5E"
  }
  "Angela" = {
    "access_key_id" = "AKIATT4QFYOMYEVJST4M"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Angela"
    "user_unique_id" = "AIDATT4QFYOMSJLLQAAKC"
  }
  "Charles" = {
    "access_key_id" = "AKIATT4QFYOM3I5ICVTI"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Charles"
    "user_unique_id" = "AIDATT4QFYOMZEPEWABGR"
  }
  "Clark" = {
    "access_key_id" = "AKIATT4QFYOMS2ZWKH76"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Clark"
    "user_unique_id" = "AIDATT4QFYOMX4U5NLACQ"
  }
  "Creed" = {
    "access_key_id" = "AKIATT4QFYOMXO2XBEWT"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Creed"
    "user_unique_id" = "AIDATT4QFYOMRAN4BTR4M"
  }
  "Darryl" = {
    "access_key_id" = "AKIATT4QFYOM5YS2PLWD"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Darryl"
    "user_unique_id" = "AIDATT4QFYOMY6DN7W5DR"
```

7. View secrets in terraform. tfstate:

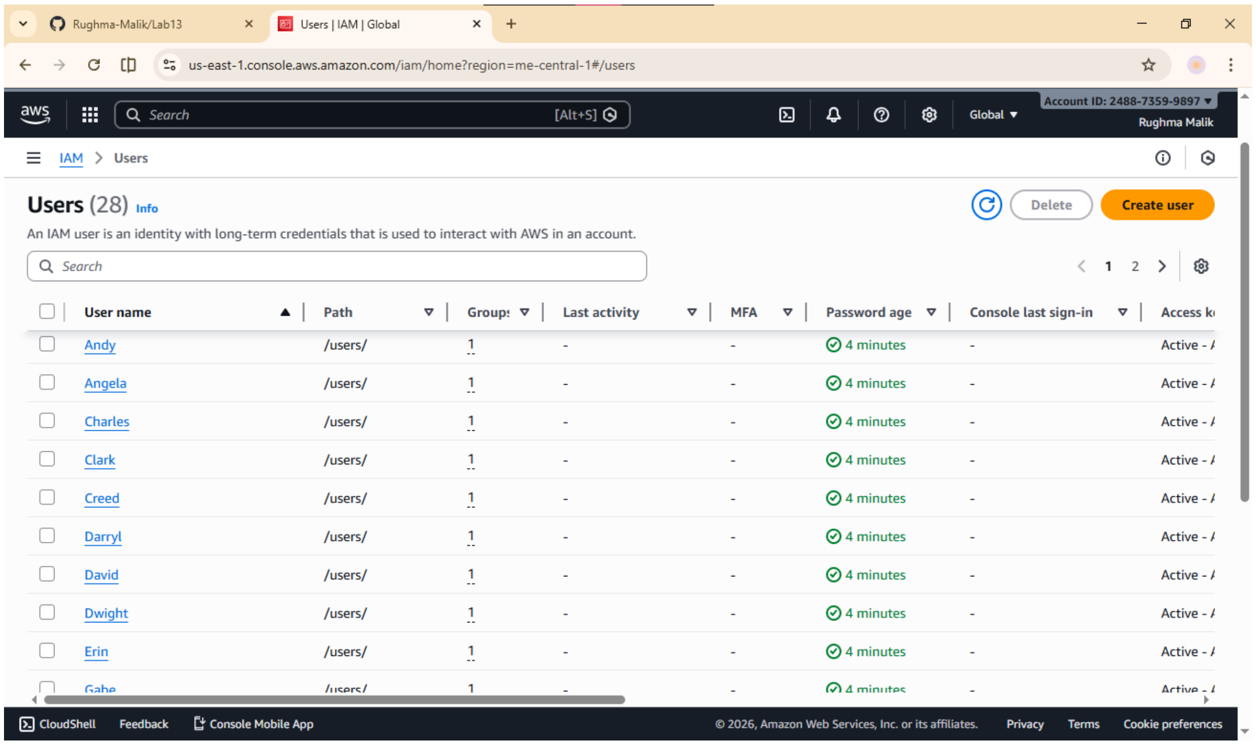
cat terraform.tfstate | grep -A 5 "all_access_key_secrets"



```
    "access_key_id" = "AKIATT4QFYOM6A4JF5T2"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Peter"
    "user_unique_id" = "AIDATT4QFYOM4R6C3IQDS"
  }
  "Phyllis" = {
    "access_key_id" = "AKIATT4QFYOMYOUUTOXH"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Phyllis"
    "user_unique_id" = "AIDATT4QFYOMREXTC2JCW"
  }
  "Robert" = {
    "access_key_id" = "AKIATT4QFYOMUOFN5PXG"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Robert"
    "user_unique_id" = "AIDATT4QFYOMSYU72KFYX"
  }
  "Ryan" = {
    "access_key_id" = "AKIATT4QFYOMTLQXYVZC"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Ryan"
    "user_unique_id" = "AIDATT4QFYOM4PWK2LAXR"
  }
  "Stanley" = {
    "access_key_id" = "AKIATT4QFYOMXRFMGL5T"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Stanley"
    "user_unique_id" = "AIDATT4QFYOMRXIOEDUQI"
  }
  "Toby" = {
    "access_key_id" = "AKIATT4QFYOM45TTO22J"
    "user_arn" = "arn:aws:iam::248873599897:user/users/Toby"
    "user_unique_id" = "AIDATT4QFYOMRKE33VRWR"
  }
}
@Rughma-Malik ➜ ~/Lab13 $ cat terraform.tfstate | grep -A 5 "all_access_key_secrets"
@Rughma-Malik ➜ ~/Lab13 $ cat terraform.tfstate | grep -A 5 "all_access_key_secrets"
```

8. Verify all users in AWS Console:

- Navigate to IAM → Users



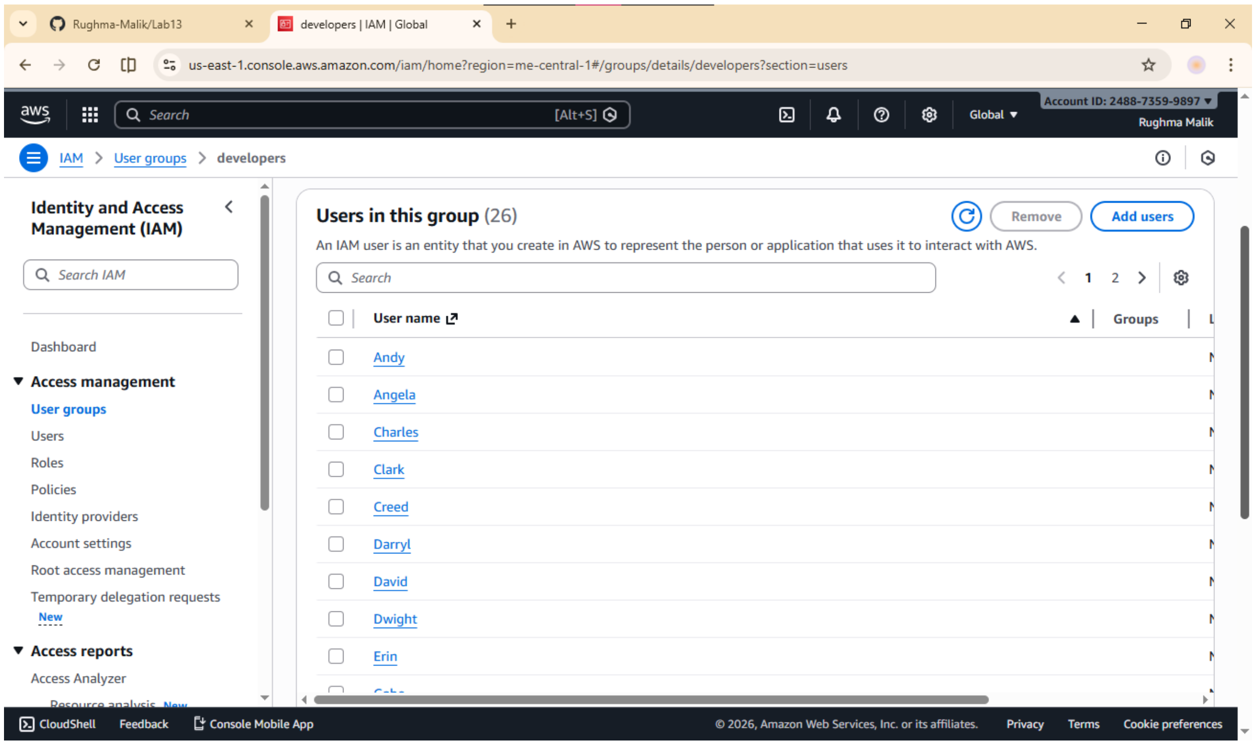
Users (28) Info

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

User name	Path	Groups	Last activity	MFA	Password age	Console last sign-in	Access k
Andy	/users/	1	-	-	✓ 4 minutes	-	Active -
Angela	/users/	1	-	-	✓ 4 minutes	-	Active -
Charles	/users/	1	-	-	✓ 4 minutes	-	Active -
Clark	/users/	1	-	-	✓ 4 minutes	-	Active -
Creed	/users/	1	-	-	✓ 4 minutes	-	Active -
Darryl	/users/	1	-	-	✓ 4 minutes	-	Active -
David	/users/	1	-	-	✓ 4 minutes	-	Active -
Dwight	/users/	1	-	-	✓ 4 minutes	-	Active -
Erin	/users/	1	-	-	✓ 4 minutes	-	Active -
Gabe	/users/	1	-	-	✓ 4 minutes	-	Active -

9. Verify group membership:

- Navigate to IAM → Groups → developers → Users tab



us-east-1.console.aws.amazon.com/iam/home?region=me-central-1#/groups/details/developers?section=users

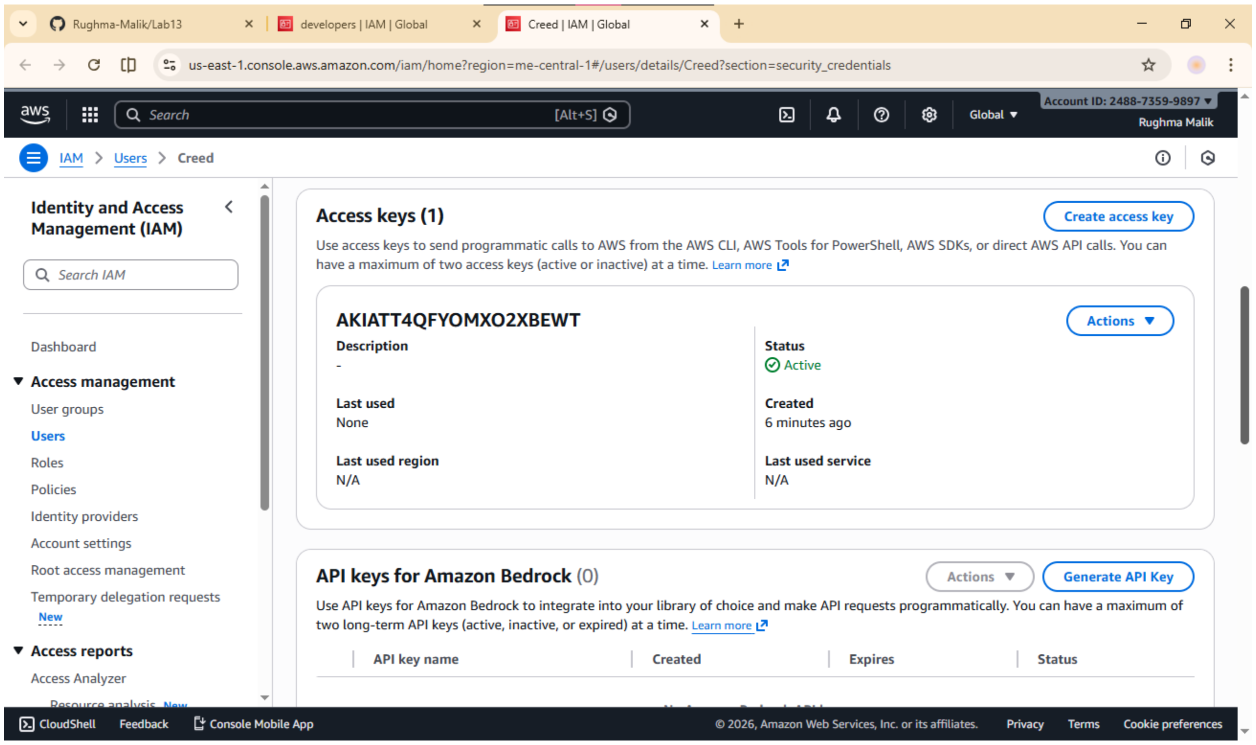
IAM > User groups > developers

Users in this group (26)

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

User name	Groups
Andy	
Angela	
Charles	
Clark	
Creed	
Darryl	
David	
Dwight	
Erin	

10. Verify one user's access keys:



us-east-1.console.aws.amazon.com/iam/home?region=me-central-1#/users/details/Creed?section=security_credentials

IAM > Users > Creed

Access keys (1)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. Learn more

AKIATT4QFYOMXO2XBEWT

Description: -
Status: Active
Last used: None
Created: 6 minutes ago
Last used region: N/A
Last used service: N/A

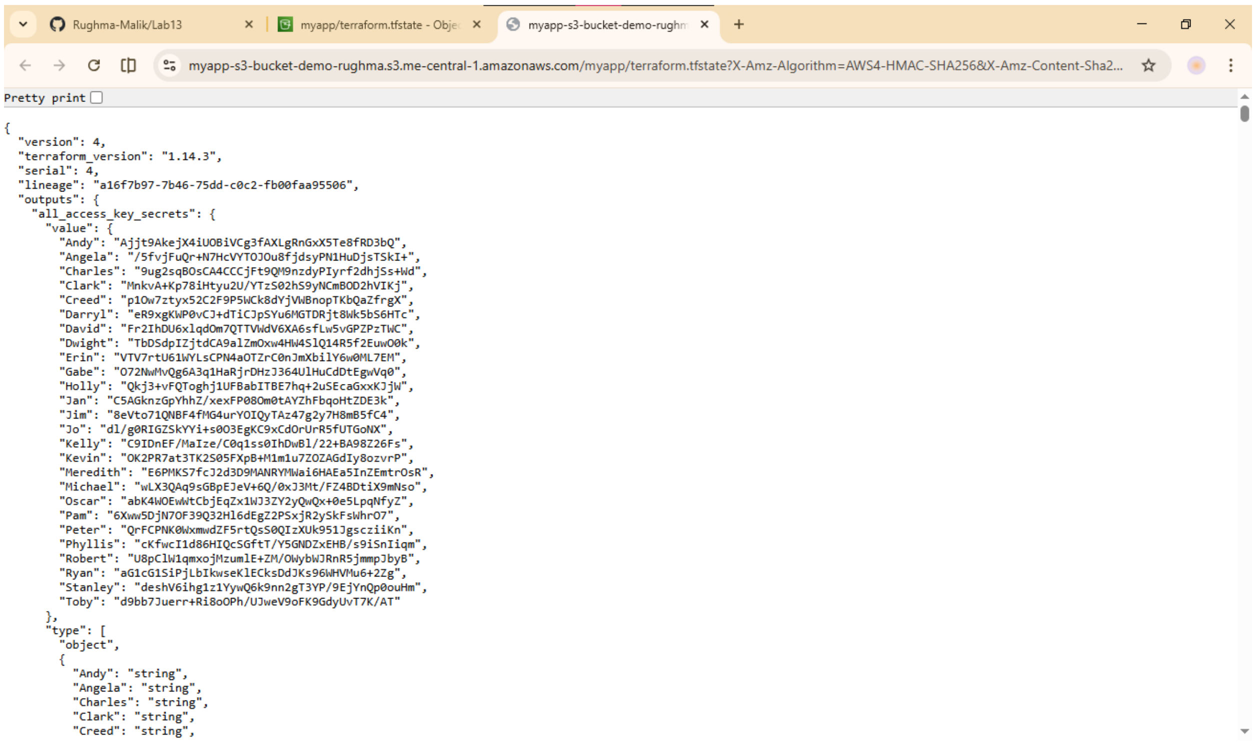
API keys for Amazon Bedrock (0)

Use API keys for Amazon Bedrock to integrate into your library of choice and make API requests programmatically. You can have a maximum of two long-term API keys (active, inactive, or expired) at a time. Learn more

API key name	Created	Expires	Status

11. Check terraform state in S3:

- Navigate to S3 bucket and view the state file



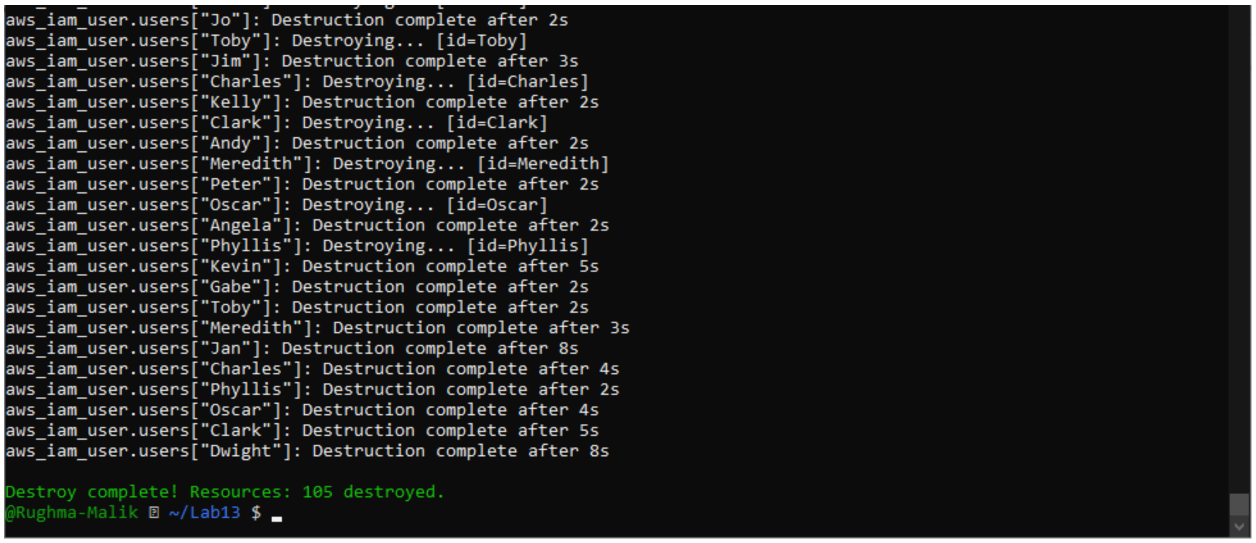
Rughma-Malik/Lab13
myapp/terraform.tfstate - Obje
myapp-s3-bucket-demo-rughm
myapp-s3-bucket-demo-rughma.s3.me-central-1.amazonaws.com/myapp/terraform.tfstate?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Content-Sha2...
Pretty print

```
{
  "version": 4,
  "terraform_version": "1.14.3",
  "serial": 4,
  "lineage": "a16f7b97-7b46-75dd-c0c2-fb00faa95506",
  "outputs": {
    "all_access_key_secrets": {
      "value": {
        "Andy": "Ajjt9AkejX4iUOBiVCg3fAXLgRnGxX5Te8fRD3bQ",
        "Angela": "/5fvjFuQr+N7HcVYTOJOu8fjdsyPN1HuDjsTSkI+",
        "Charles": "9ug2sqBOsCA4CCCjFt9QM9nzdyPIyrf2dhjSs+Wd",
        "Clark": "MnkvA+Kp78iHtyu2U/YTzS02hS9yNCmBOD2hVIKj",
        "Creed": "p1Ow7ztyx52C2F9P5WCk8dYjVWBnopTKbQaZfrgX",
        "Darryl": "eR9xgKWP0vCJ+dTiCJpSYu6MGTDRjt8Wk5bS6HTc",
        "David": "Fr2IhDU6xlqdOm7QTTVWdV6XA6sfLw5vGPZPzTWC",
        "Dwight": "TbDSdpIZjtdCA9alZmOxw4HW4SlQ14R5f2EuwO0k",
        "Erin": "VTV7rtU61WYLsCPN4aOTZrC0nJmXbilY6w0ML7EM",
        "Gabe": "O72NwMvQg6A3q1HaRjrDHzJ364UlHuCdDtEgwVq0",
        "Holly": "Qkj3+vFQToghj1UFBabITBE7hq+2uSEcaGxxKJjW",
        "Jan": "C5AGknzGpYhhZ/xexFP08Om0tAYZhFbqoHtZDE3k",
        "Jim": "8eVto71QNBF4fMG4urYOIQyTAz47g2y7H8mB5fC4",
        "Jo": "dl/g0RIGZSkYYi+s0O3EgKC9xCdOrUrR5fUTGoNX",
        "Kelly": "C9IDnEF/MaIze/C0q1ss0IhDwBl/22+BA98Z26Fs",
        "Kevin": "OK2PR7at3TK2S05FXpB+M1m1u7ZOZAGdIy8ozvrP",
        "Meredith": "E6PMKS7fcJ2d3D9MANRYMWai6HAEa5InZEmtrOsR",
        "Michael": "wLX3QAq9sGBpEJeV+6Q/0xJ3Mt/FZ4BDtiX9mNso",
        "Oscar": "abK4WOEwWtCbjEqZx1WJ3ZY2yQwQx+0e5LpqNfyZ",
        "Pam": "6Xww5DjN7OF39Q32Hl6dEgZ2PSxjR2ySkFsWhrO7",
        "Peter": "QrFCPNK0WxmwdZF5rtQsS0QIzXUk951JgscziiKn",
        "Phyllis": "cKfwcI1d86HIQcSGftT/Y5GNDZxEHB/s9iSnIiqm",
        "Robert": "U8pClW1qmxojMzumlE+ZM/OWybWJRnR5jmmpJbyB",
        "Ryan": "aG1cG1SiPjLbIkwseKlECksDdJKs96WHVMu6+2Zg",
        "Stanley": "deshV6ihg1z1YywQ6k9nn2gT3YP/9EjYnQp0ouHm",
        "Toby": "d9bb7Juerr+Ri8oOPh/UJweV9oFK9GdyUvT7K/AT"
      },
      "type": [
        "object",
        {
          "Andy": "string",
          "Angela": "string",
          "Charles": "string",
          "Clark": "string",
          "Creed": "string",
```

Cleanup

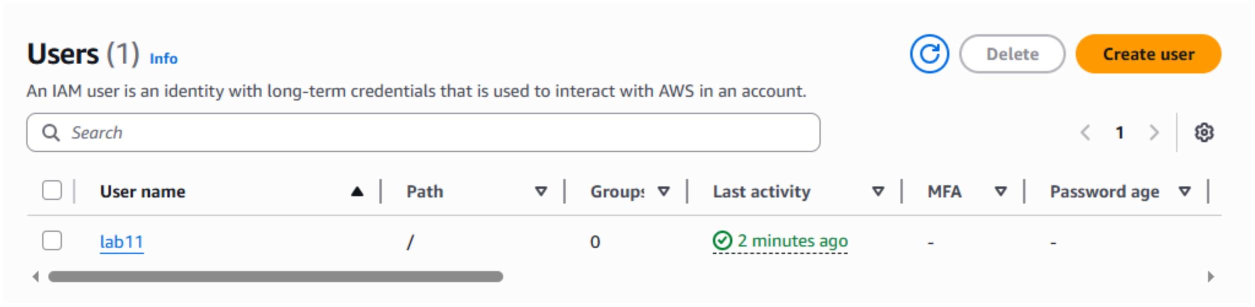
1. Destroy all resources:

terraform destroy -auto-approve



```
aws_iam_user.users["Jo"]: Destruction complete after 2s
aws_iam_user.users["Toby"]: Destroying... [id=Toby]
aws_iam_user.users["Jim"]: Destruction complete after 3s
aws_iam_user.users["Charles"]: Destroying... [id=Charles]
aws_iam_user.users["Kelly"]: Destruction complete after 2s
aws_iam_user.users["Clark"]: Destroying... [id=Clark]
aws_iam_user.users["Andy"]: Destruction complete after 2s
aws_iam_user.users["Meredith"]: Destroying... [id=Meredith]
aws_iam_user.users["Peter"]: Destruction complete after 2s
aws_iam_user.users["Oscar"]: Destroying... [id=Oscar]
aws_iam_user.users["Angela"]: Destruction complete after 2s
aws_iam_user.users["Phyllis"]: Destroying... [id=Phyllis]
aws_iam_user.users["Kevin"]: Destruction complete after 5s
aws_iam_user.users["Gabe"]: Destruction complete after 2s
aws_iam_user.users["Toby"]: Destruction complete after 2s
aws_iam_user.users["Meredith"]: Destruction complete after 3s
aws_iam_user.users["Jan"]: Destruction complete after 8s
aws_iam_user.users["Charles"]: Destruction complete after 4s
aws_iam_user.users["Phyllis"]: Destruction complete after 2s
aws_iam_user.users["Oscar"]: Destruction complete after 4s
aws_iam_user.users["Clark"]: Destruction complete after 5s
aws_iam_user.users["Dwight"]: Destruction complete after 8s

Destroy complete! Resources: 105 destroyed.
@Rughma-Malik ➜ ~/Lab13 $ _
```

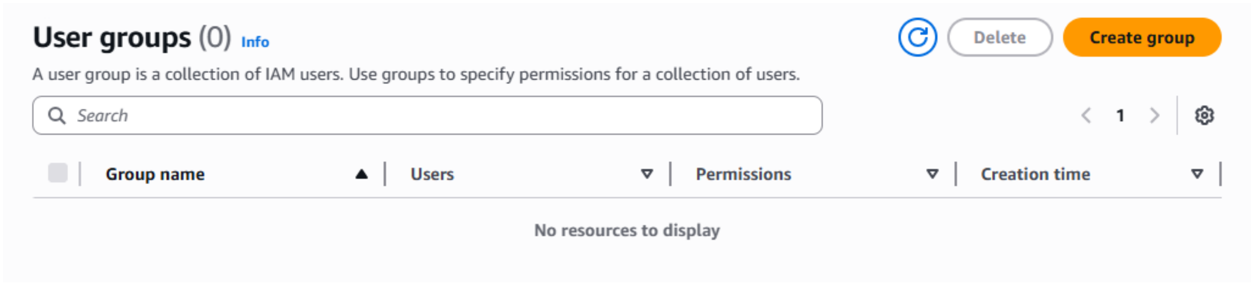
2. Verify users deleted in AWS Console:



Users (1) Info
Delete
Create user
An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.
Search

User name	Path	Groups	Last activity	MFA	Password age
lab11	/	0	2 minutes ago	-	-

3. Verify group deleted in AWS Console:

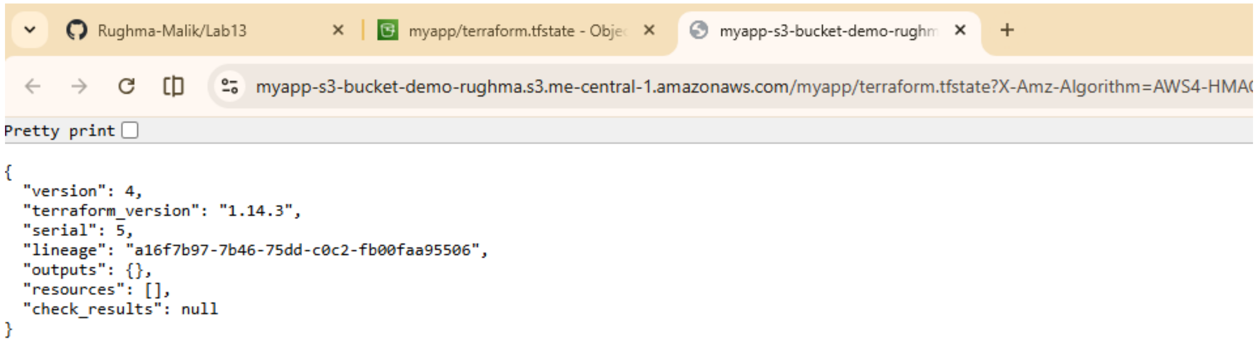


User groups (0) Info
Delete
Create group
A user group is a collection of IAM users. Use groups to specify permissions for a collection of users.
Search

Group name	Users	Permissions	Creation time

No resources to display

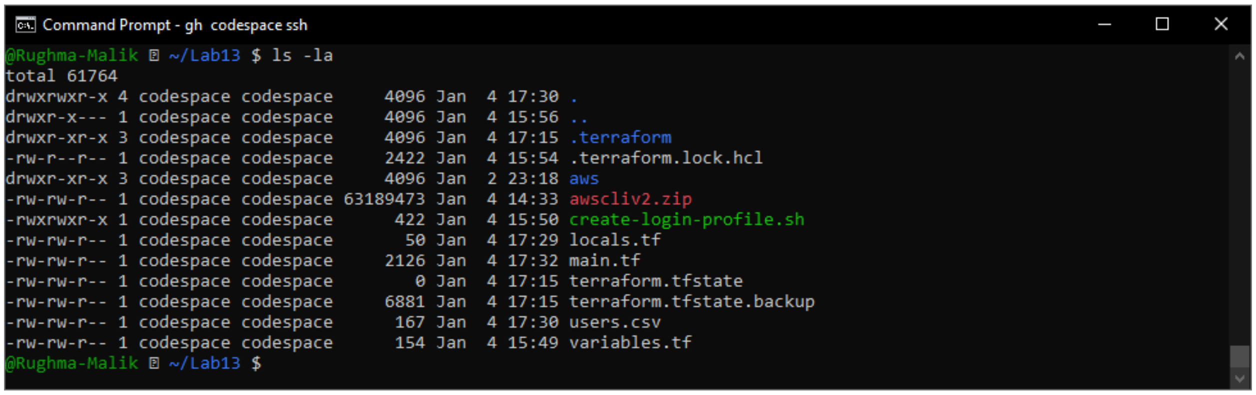
4. Check S3 state file:



Rughma-Malik/Lab13
myapp/terraform.tfstate - Obje
myapp-s3-bucket-demo-rughm
myapp-s3-bucket-demo-rughma.s3.me-central-1.amazonaws.com/myapp/terraform.tfstate?X-Amz-Algorithm=AWS4-HMAC
Pretty print

```
{
  "version": 4,
  "terraform_version": "1.14.3",
  "serial": 5,
  "lineage": "a16f7b97-7b46-75dd-c0c2-fb00faa95506",
  "outputs": {},
  "resources": [],
  "check_results": null
}
```

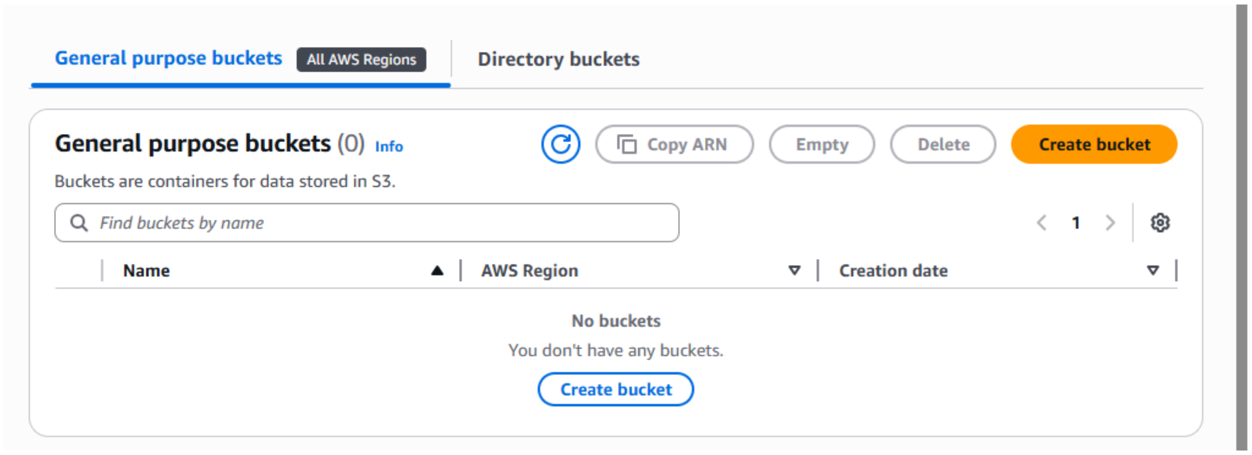
5. List all project files:



```
@Rughma-Malik ➜ ~/Lab13 $ ls -la
total 61764
drwxrwxr-x 4 codespace codespace     4096 Jan  4 17:30 .
drwxr-x--- 1 codespace codespace     4096 Jan  4 15:56 ..
drwxr-xr-x 3 codespace codespace     4096 Jan  4 17:15 .terraform
-rw-r--r-- 1 codespace codespace     2422 Jan  4 15:54 .terraform.lock.hcl
drwxr-xr-x 3 codespace codespace     4096 Jan  2 23:18 aws
-rw-rw-r-- 1 codespace codespace 63189473 Jan  4 14:33 awscliv2.zip
-rwxrwxr-x 1 codespace codespace      422 Jan  4 15:50 create-login-profile.sh
-rw-rw-r-- 1 codespace codespace       50 Jan  4 17:29 locals.tf
-rw-rw-r-- 1 codespace codespace     2126 Jan  4 17:32 main.tf
-rw-rw-r-- 1 codespace codespace        0 Jan  4 17:15 terraform.tfstate
-rw-rw-r-- 1 codespace codespace     6881 Jan  4 17:15 terraform.tfstate.backup
-rw-rw-r-- 1 codespace codespace      167 Jan  4 17:30 users.csv
-rw-rw-r-- 1 codespace codespace      154 Jan  4 15:49 variables.tf
@Rughma-Malik ➜ ~/Lab13 $
```

6. (Optional) Delete S3 bucket:

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



General purpose buckets (0) Info

Buckets are containers for data stored in S3.

No buckets

You don't have any buckets.