Cloud Computing Lab

Name: Arooj Saleem

Roll # 2023-BSE-013

Submitted To: Engr. Muhammad Shoaib

Lab Title: Terraform IAM Management with AWS

Lab # 13

Task 0 Lab Setup (Codespace & GH CLI)

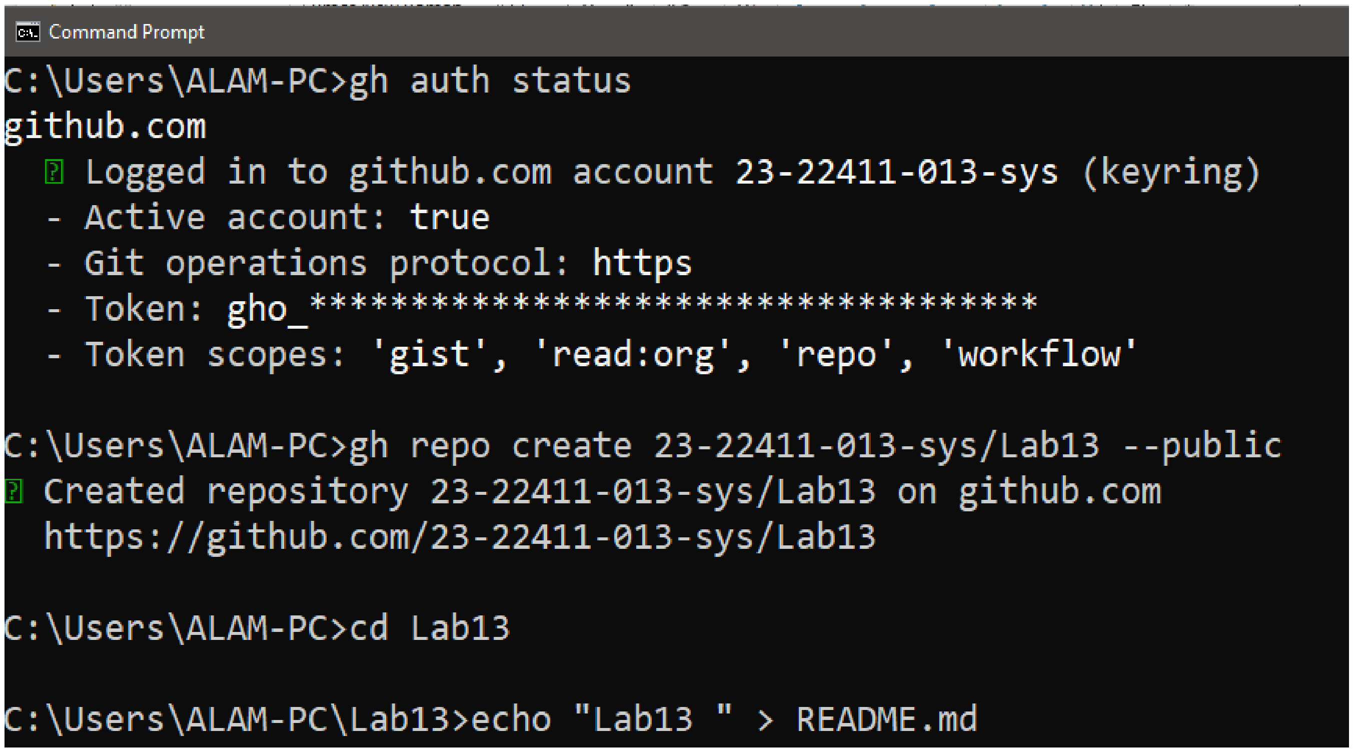
Create Codespace & connect:

create or open codespace via GH CLI (example)

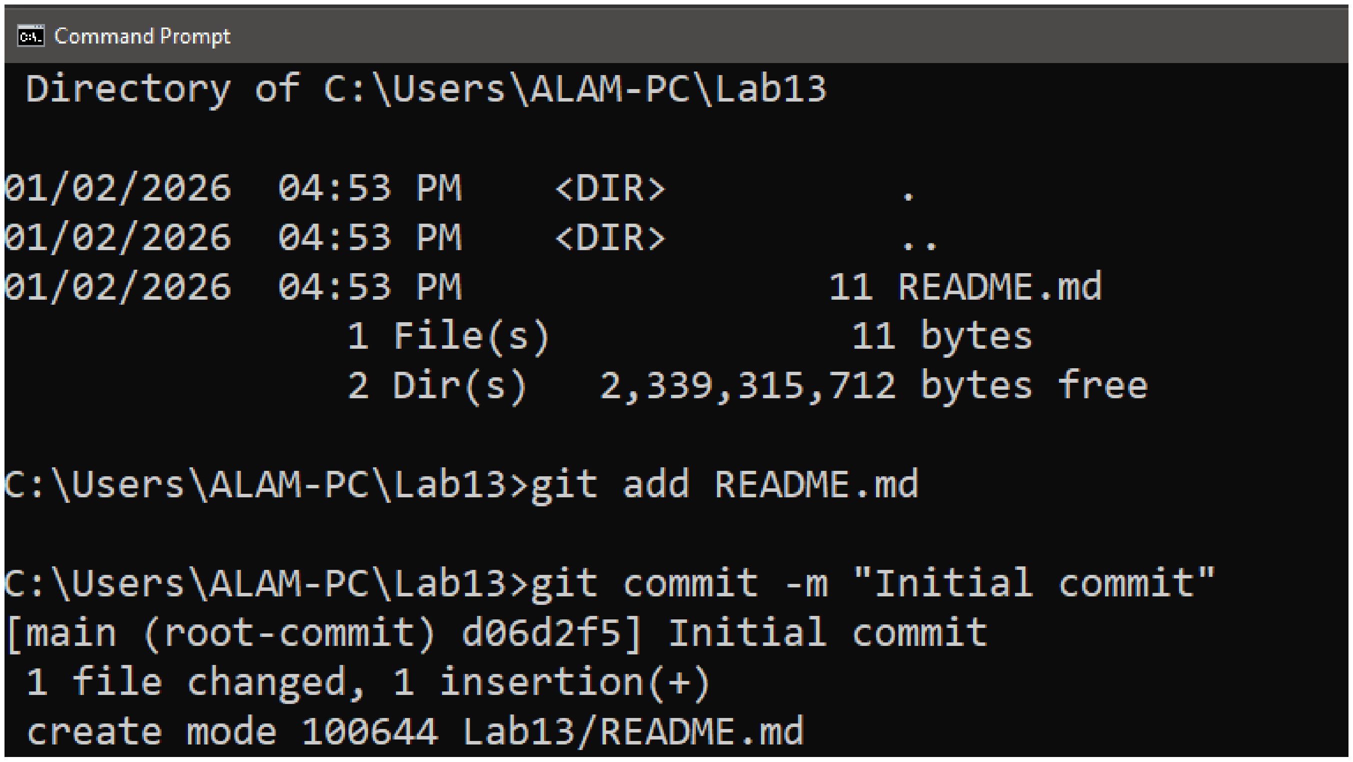
gh repo create CC_<YourName>_<YourRollNumber>/Lab13 --public

gh codespace create --repo <user_name>/Lab13

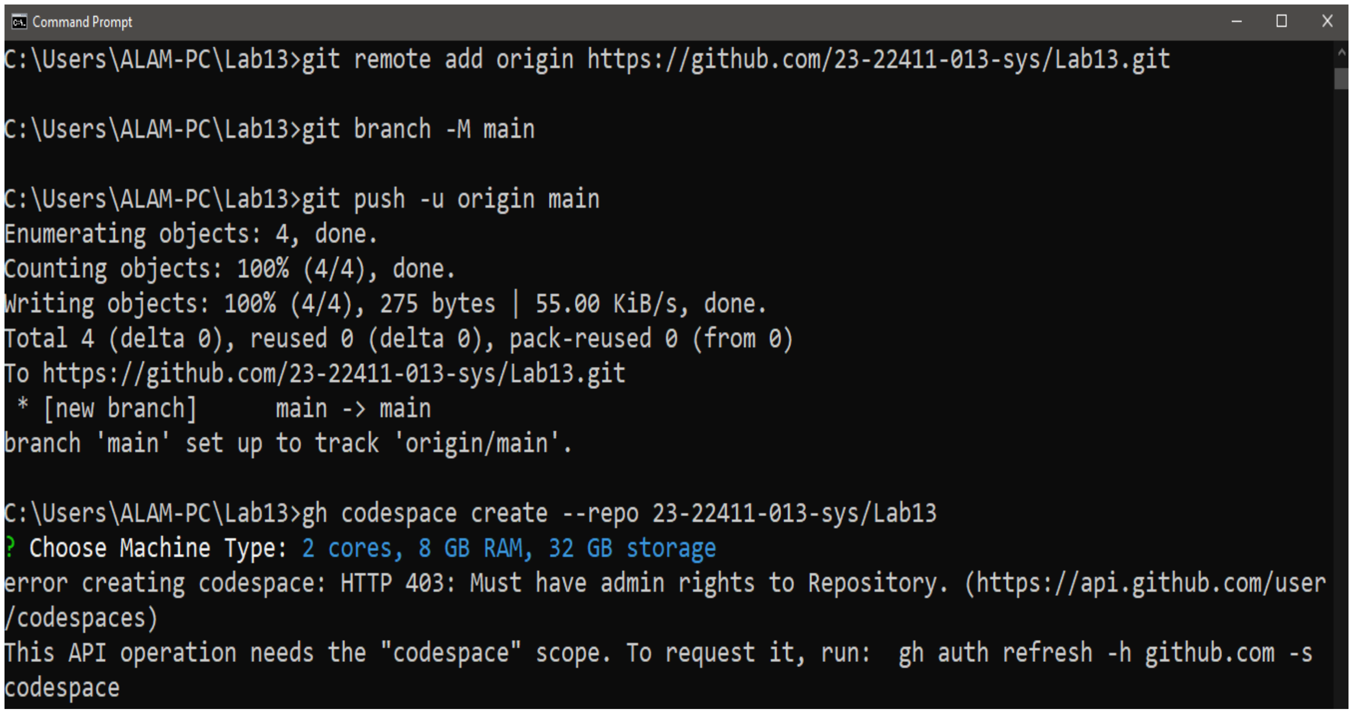
gh codespace list



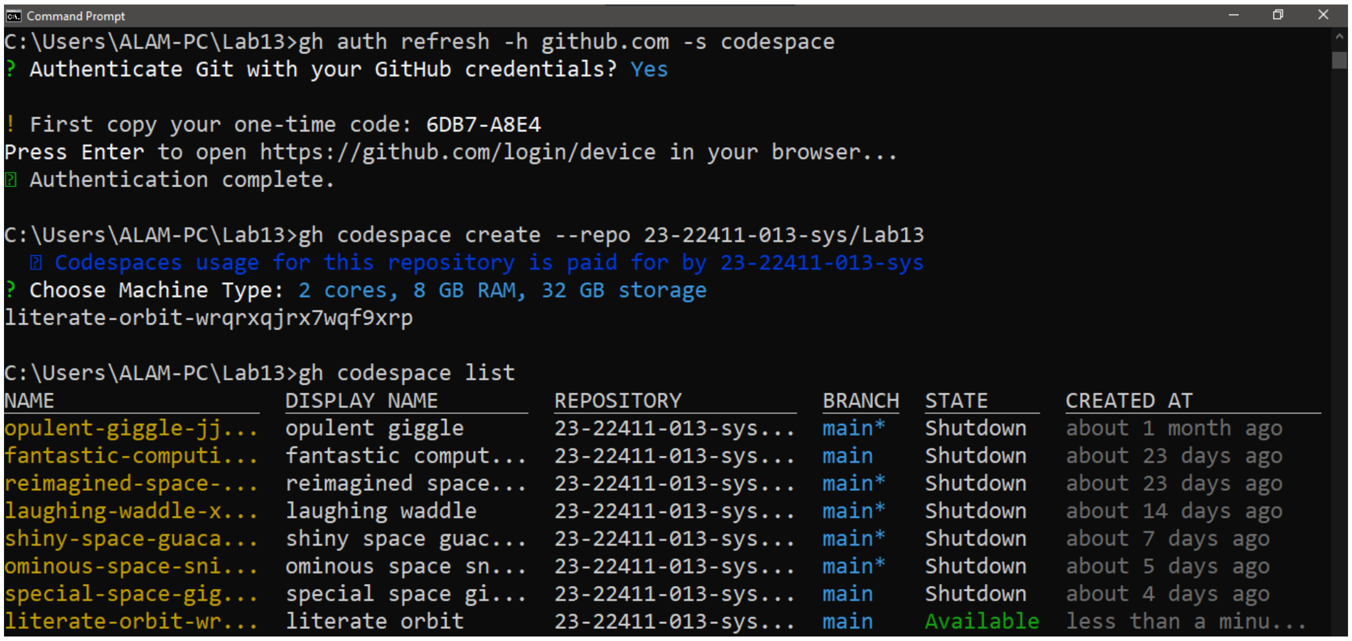
```
C:\Users\ALAM-PC>gh auth status
github.com
  ? Logged in to github.com account 23-22411-013-sys (keyring)
  - Active account: true
  - Git operations protocol: https
  - Token: gho_************************************
  - Token scopes: 'gist', 'read:org', 'repo', 'workflow'

C:\Users\ALAM-PC>gh repo create 23-22411-013-sys/Lab13 --public
? Created repository 23-22411-013-sys/Lab13 on github.com
  https://github.com/23-22411-013-sys/Lab13

C:\Users\ALAM-PC>cd Lab13

C:\Users\ALAM-PC\Lab13>echo "Lab13 " > README.md
```



```
 Directory of C:\Users\ALAM-PC\Lab13

01/02/2026  04:53 PM    <DIR>          .
01/02/2026  04:53 PM    <DIR>          ..
01/02/2026  04:53 PM                11 README.md
               1 File(s)             11 bytes
               2 Dir(s)   2,339,315,712 bytes free

C:\Users\ALAM-PC\Lab13>git add README.md

C:\Users\ALAM-PC\Lab13>git commit -m "Initial commit"
[main (root-commit) d06d2f5] Initial commit
 1 file changed, 1 insertion(+)
 create mode 100644 Lab13/README.md
```

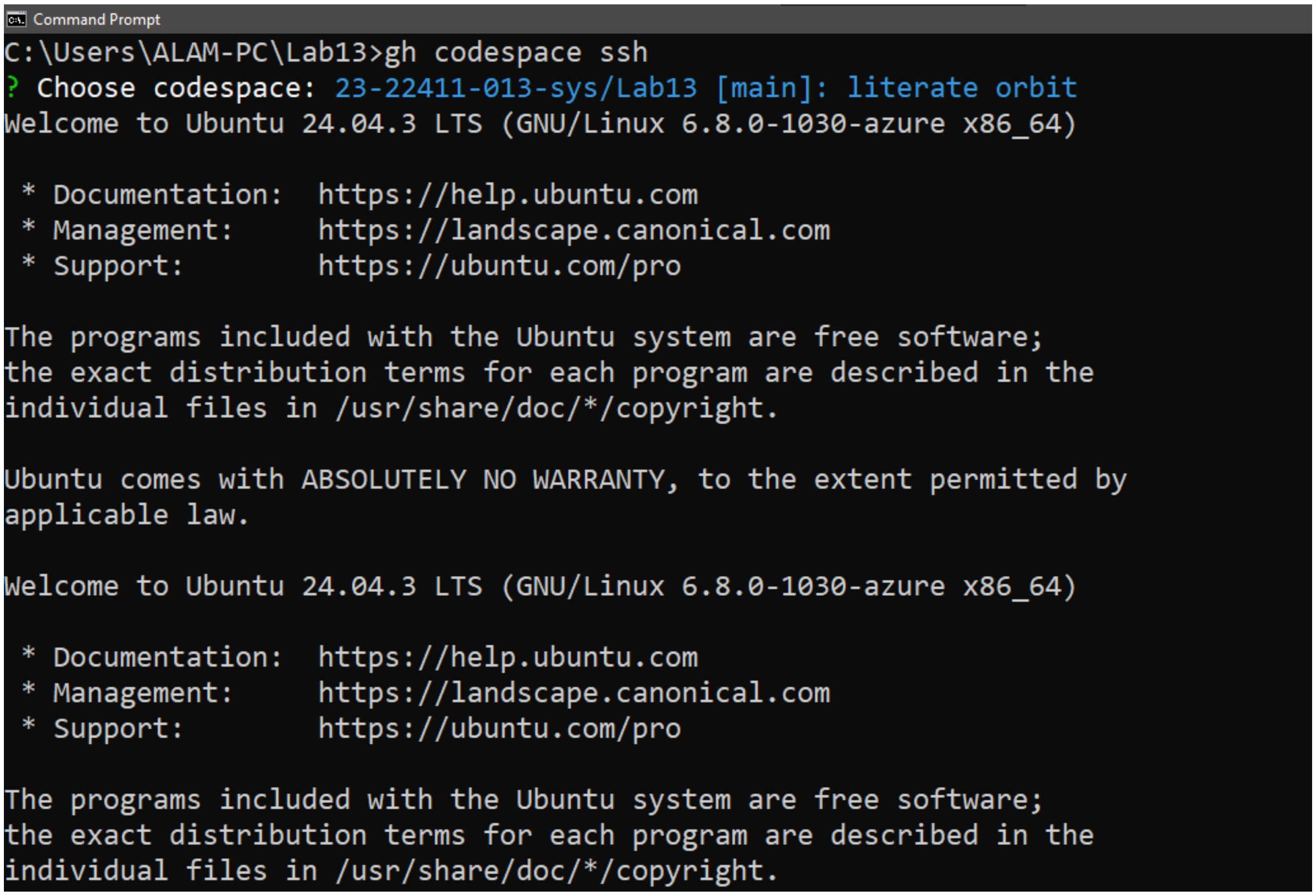


```
C:\Users\ALAM-PC\Lab13>git remote add origin https://github.com/23-22411-013-sys/Lab13.git

C:\Users\ALAM-PC\Lab13>git branch -M main

C:\Users\ALAM-PC\Lab13>git push -u origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Writing objects: 100% (4/4), 275 bytes | 55.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/23-22411-013-sys/Lab13.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

C:\Users\ALAM-PC\Lab13>gh codespace create --repo 23-22411-013-sys/Lab13
? Choose Machine Type: 2 cores, 8 GB RAM, 32 GB storage
error creating codespace: HTTP 403: Must have admin rights to Repository. (https://api.github.com/user
/codespaces)
This API operation needs the "codespace" scope. To request it, run:  gh auth refresh -h github.com -s
codespace
```



```
C:\Users\ALAM-PC\Lab13>gh auth refresh -h github.com -s codespace
? Authenticate Git with your GitHub credentials? Yes

! First copy your one-time code: 6DB7-A8E4
Press Enter to open https://github.com/login/device in your browser...
✓ Authentication complete.

C:\Users\ALAM-PC\Lab13>gh codespace create --repo 23-22411-013-sys/Lab13
  ✓ Codespaces usage for this repository is paid for by 23-22411-013-sys
? Choose Machine Type: 2 cores, 8 GB RAM, 32 GB storage
literate-orbit-wrqrxqjrx7wqf9xrp

C:\Users\ALAM-PC\Lab13>gh codespace list
NAME                  DISPLAY NAME          REPOSITORY           BRANCH   STATE      CREATED AT
opulent-giggle-jj...  opulent giggle        23-22411-013-sys...  main*    Shutdown   about 1 month ago
fantastic-computi...  fantastic comput...   23-22411-013-sys...  main     Shutdown   about 23 days ago
reimagined-space-...  reimagined space...   23-22411-013-sys...  main*    Shutdown   about 23 days ago
laughing-waddle-x...  laughing waddle       23-22411-013-sys...  main*    Shutdown   about 14 days ago
shiny-space-guaca...  shiny space guac...   23-22411-013-sys...  main*    Shutdown   about 7 days ago
ominous-space-sni...  ominous space sn...   23-22411-013-sys...  main*    Shutdown   about 5 days ago
special-space-gig...  special space gi...   23-22411-013-sys...  main     Shutdown   about 4 days ago
literate-orbit-wr...  literate orbit        23-22411-013-sys...  main     Available  less than a minu...
```

gh codespace ssh -c <your_codespace_name>



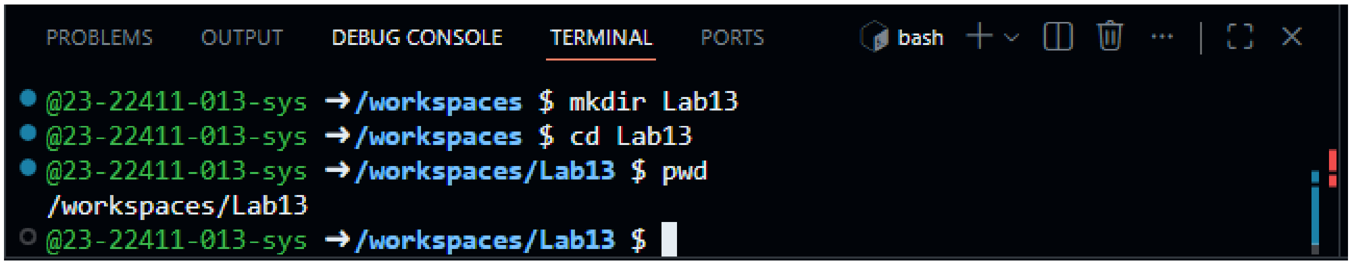
```
C:\Users\ALAM-PC\Lab13>gh codespace ssh
? Choose codespace: 23-22411-013-sys/Lab13 [main]: literate orbit
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-1030-azure x86_64)

 * 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.

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.
```

Task 1 — Create IAM Group and Output Details

Create the initial project structure:

mkdir -p ~/Lab13

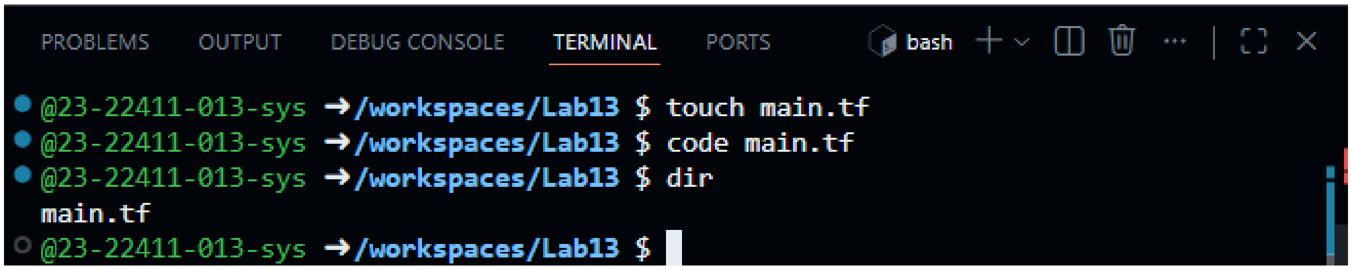
cd ~/Lab13



```
@23-22411-013-sys ➜/workspaces $ mkdir Lab13
@23-22411-013-sys ➜/workspaces $ cd Lab13
@23-22411-013-sys ➜/workspaces/Lab13 $ pwd
/workspaces/Lab13
@23-22411-013-sys ➜/workspaces/Lab13 $
```

2. Create the main Terraform file:

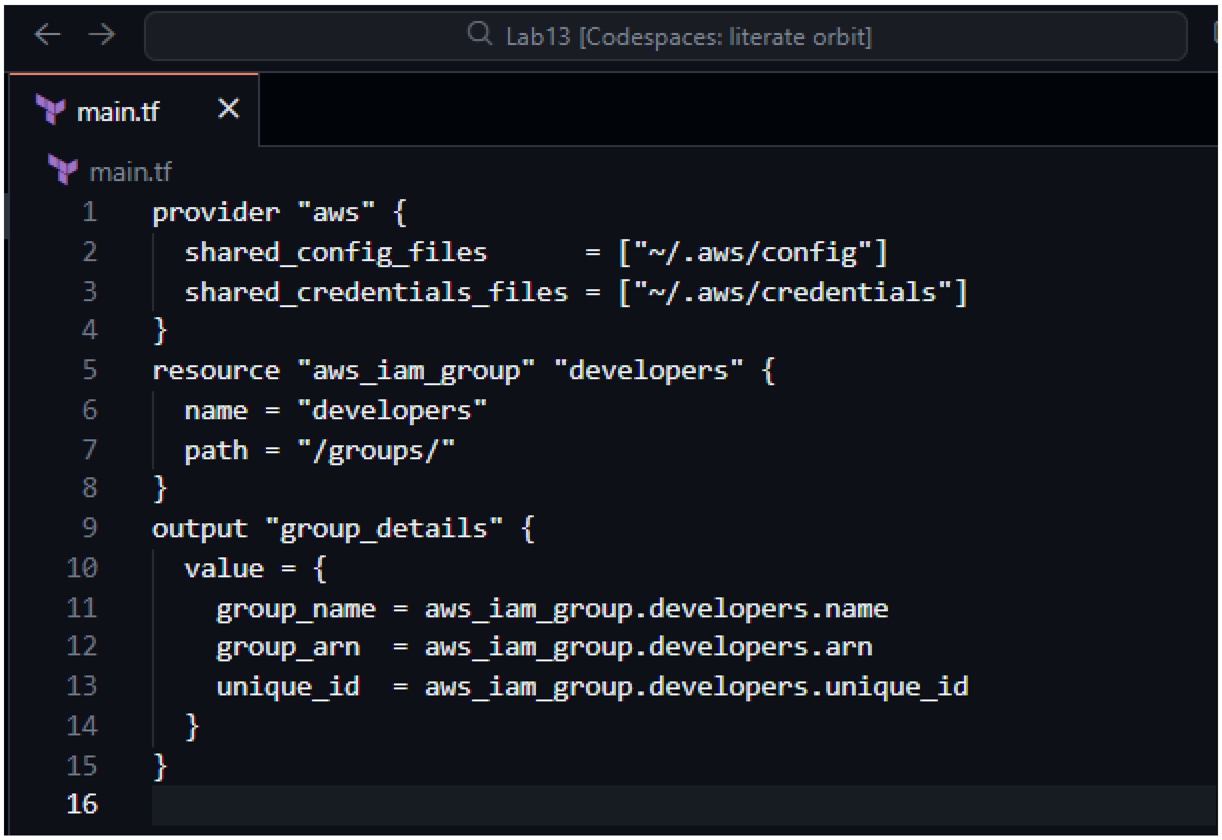
touch main.tf



```
@23-22411-013-sys ➜/workspaces/Lab13 $ touch main.tf
@23-22411-013-sys ➜/workspaces/Lab13 $ code main.tf
@23-22411-013-sys ➜/workspaces/Lab13 $ dir
main.tf
@23-22411-013-sys ➜/workspaces/Lab13 $
```

3. Create main.tf with AWS provider configuration:

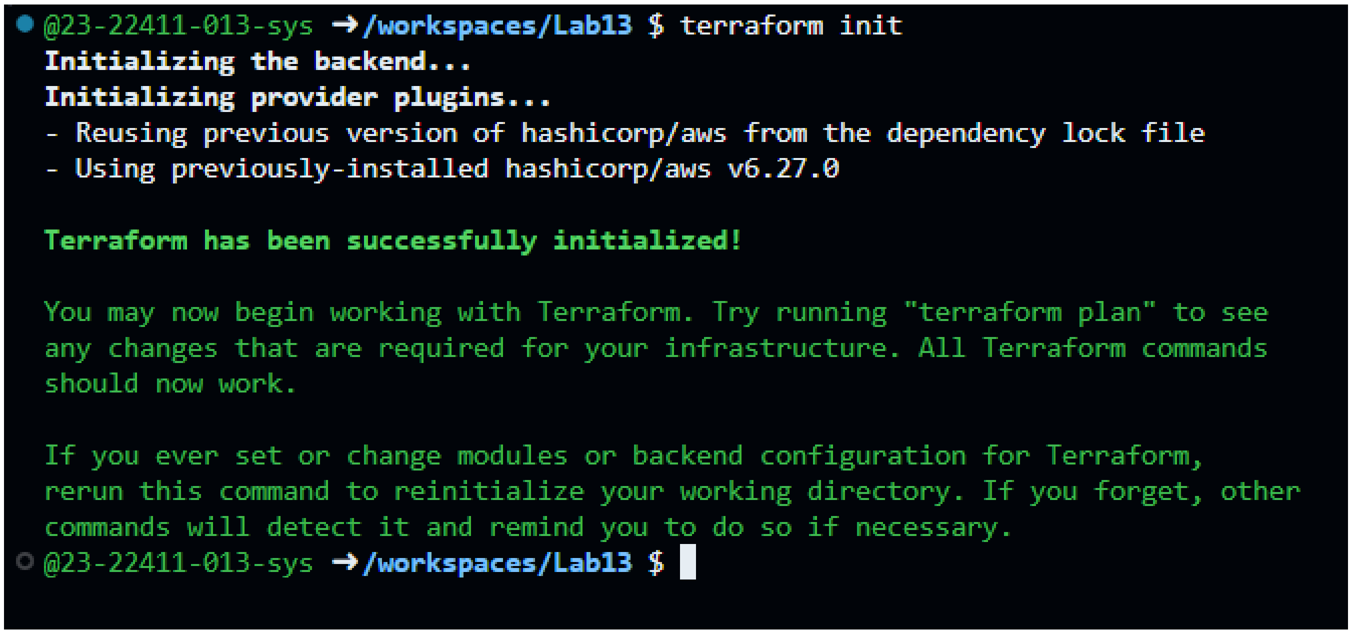
```
provider "aws" {
  shared_config_files      = ["~/.aws/config"]
  shared_credentials_files = ["~/.aws/credentials"]
}
resource "aws_iam_group" "developers" {
  name = "developers"
  path = "/groups/"
}
output "group_details" {
  value = {
    group_name = aws_iam_group.developers.name
    group_arn  = aws_iam_group.developers.arn
    unique_id  = aws_iam_group.developers.unique_id
  }
}
```



```
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:

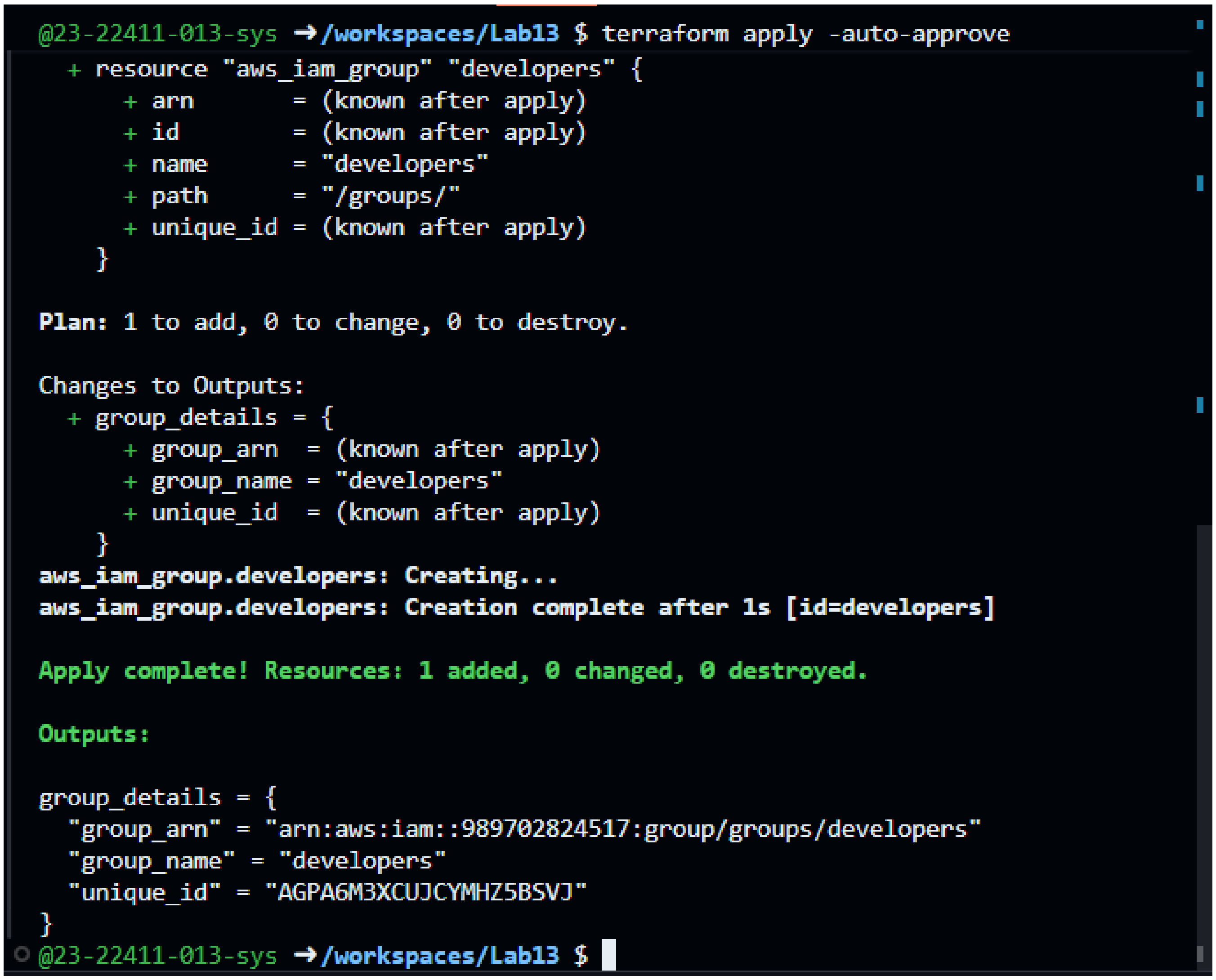
terraform init



```
@23-22411-013-sys ➜ /workspaces/Lab13 $ terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- 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.
@23-22411-013-sys ➜ /workspaces/Lab13 $
```

5. Apply the configuration:

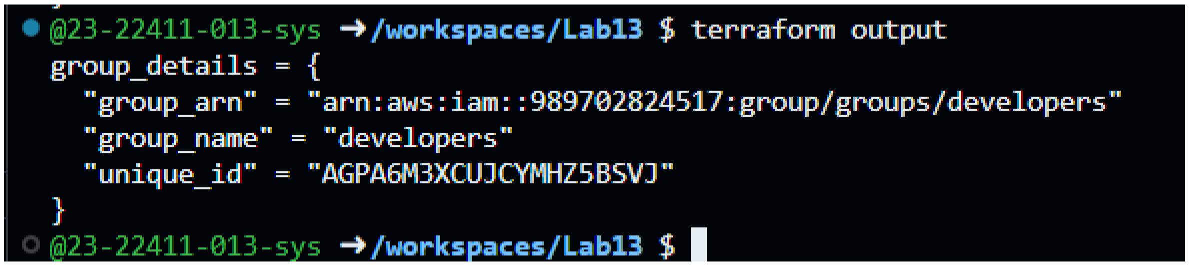
terraform apply -auto-approve



```
@23-22411-013-sys →/workspaces/Lab13 $ terraform apply -auto-approve
  + 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::989702824517:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA6M3XCUJCYMHZ5BSVJ"
}
@23-22411-013-sys →/workspaces/Lab13 $
```

6. Display the output:

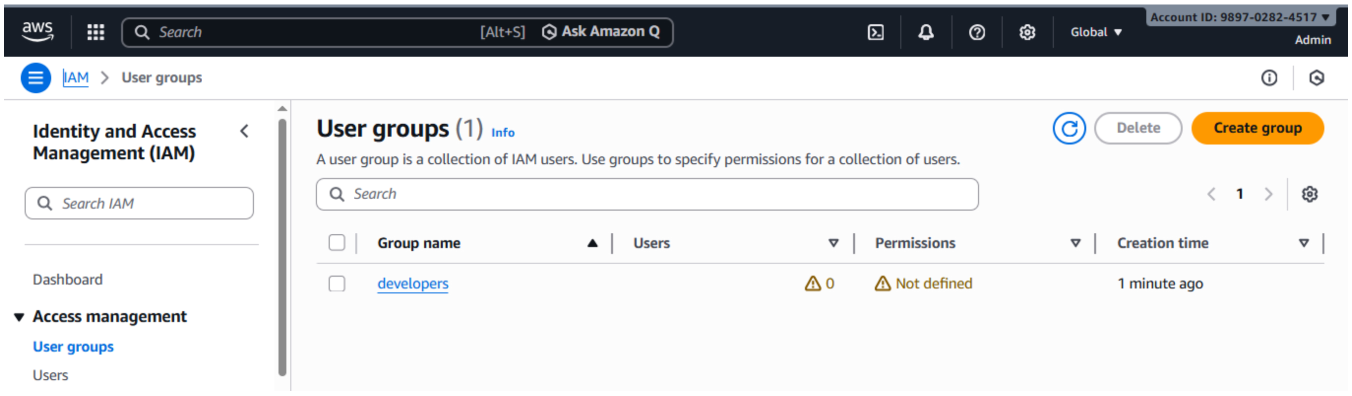
terraform output



```
@23-22411-013-sys →/workspaces/Lab13 $ terraform output
group_details = {
  "group_arn" = "arn:aws:iam::989702824517:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA6M3XCUJCYMHZ5BSVJ"
}
@23-22411-013-sys →/workspaces/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.

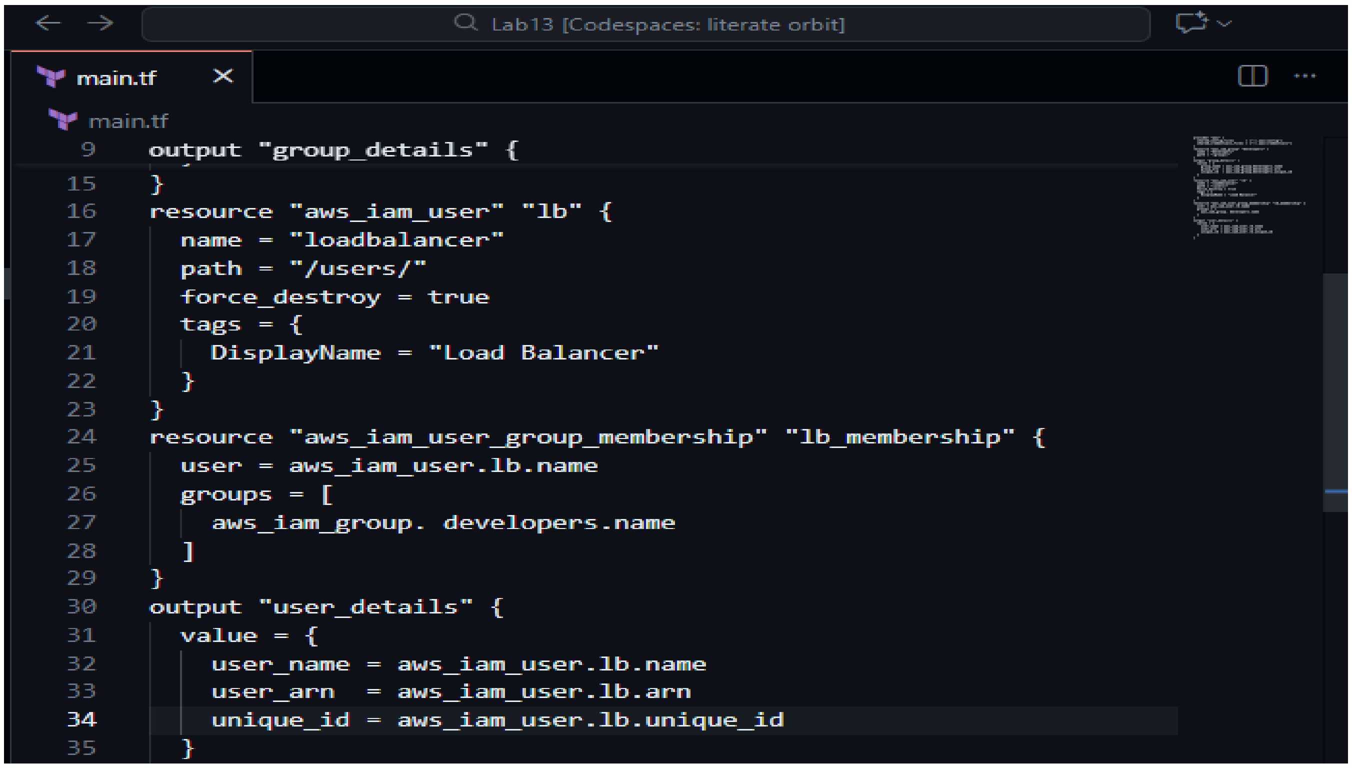
Group name	Users	Permissions	Creation time
developers	⚠ 0	⚠ Not defined	1 minute ago

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:

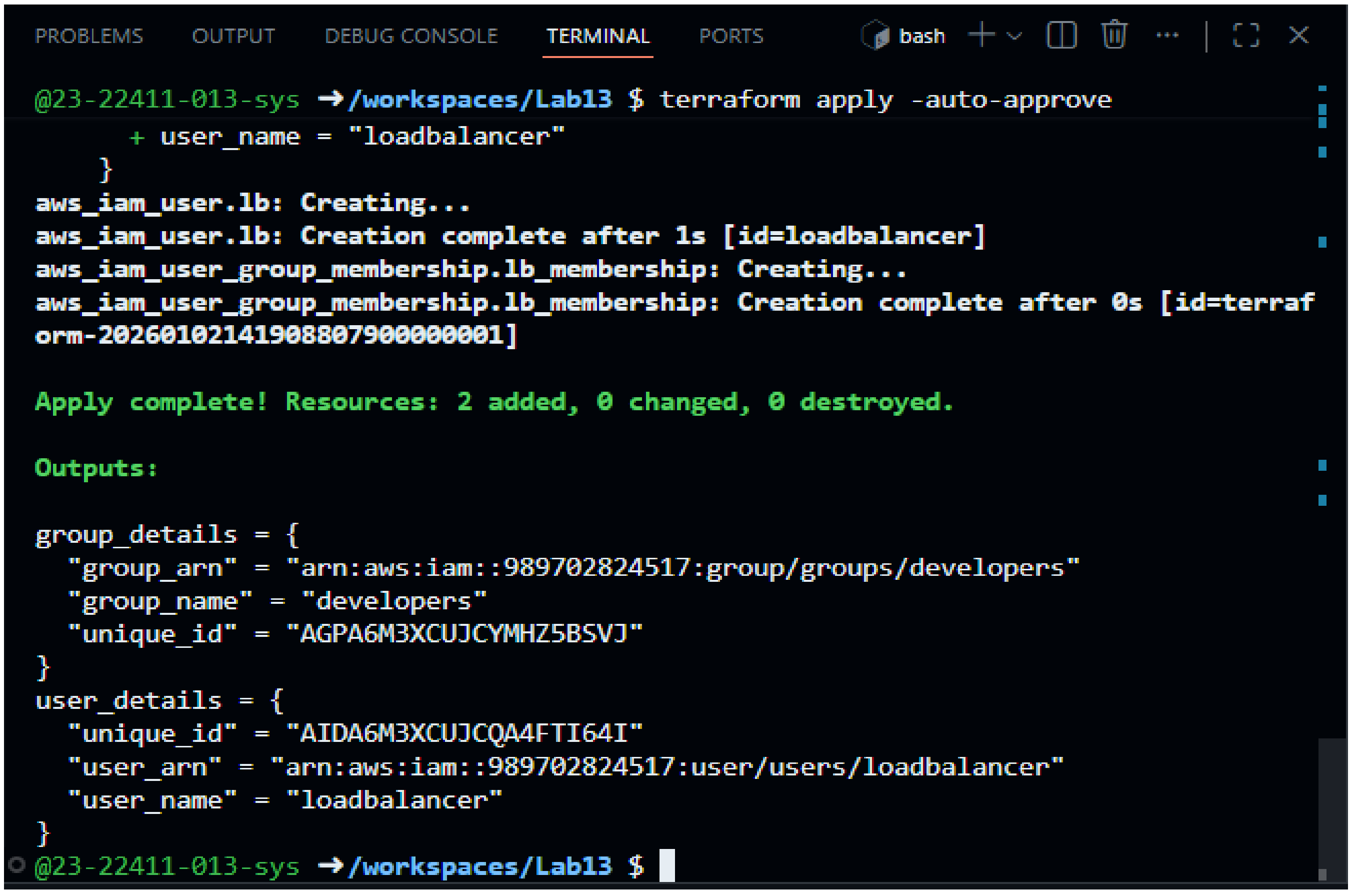
```
provider "aws" {
  shared_config_files      = ["~/.aws/config"]
  shared_credentials_files = ["~/.aws/credentials"]
}
resource "aws_iam_group" "developers" {
  name = "developers"
  path = "/groups/"
}
output "group_details" {
  value = {
    group_name = aws_iam_group.developers. name
    group_arn  = 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
  }
}
```



```
output "group_details" {
}
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
  }
```

2. Apply the configuration:

terraform apply -auto-approve

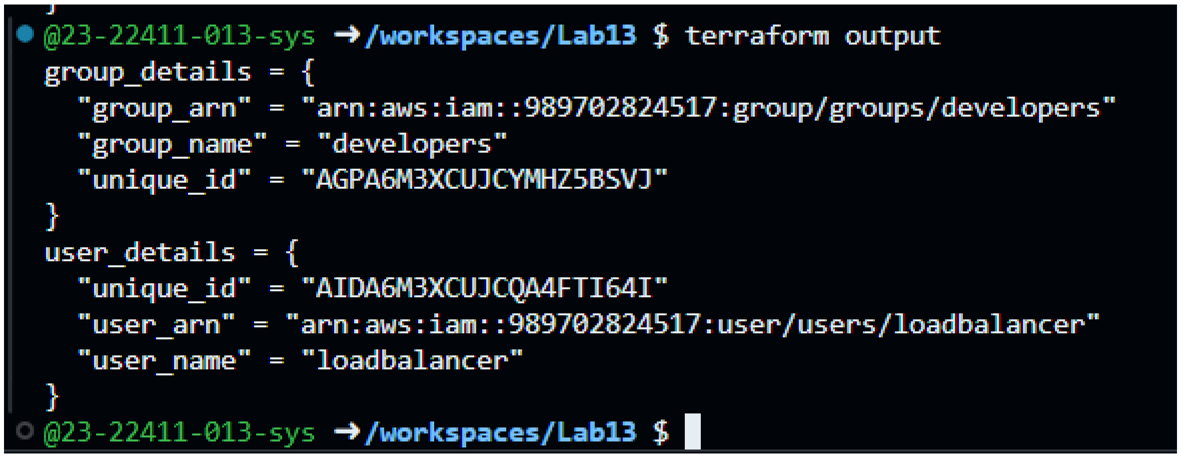


PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS bash

```
@23-22411-013-sys →/workspaces/Lab13 $ terraform apply -auto-approve
      + 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=terraf
orm-20260102141908807900000001]

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

Outputs:

group_details = {
  "group_arn" = "arn:aws:iam::989702824517:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA6M3XCUJCYMHZ5BSVJ"
}
user_details = {
  "unique_id" = "AIDA6M3XCUJCQA4FTI64I"
  "user_arn" = "arn:aws:iam::989702824517:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@23-22411-013-sys →/workspaces/Lab13 $
```

3. Display the outputs:

terraform output

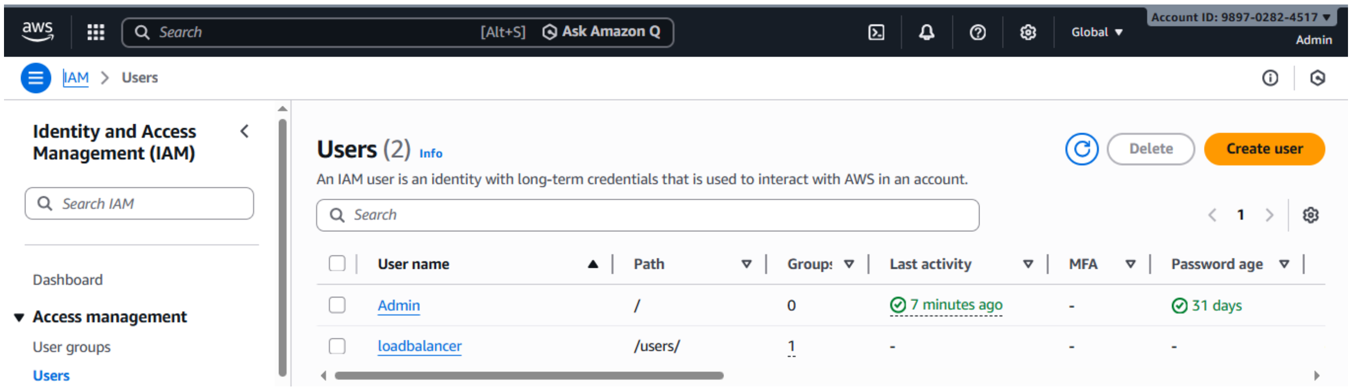


```
@23-22411-013-sys →/workspaces/Lab13 $ terraform output
group_details = {
  "group_arn" = "arn:aws:iam::989702824517:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA6M3XCUJCYMHZ5BSVJ"
}
user_details = {
  "unique_id" = "AIDA6M3XCUJCQA4FTI64I"
  "user_arn" = "arn:aws:iam::989702824517:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@23-22411-013-sys →/workspaces/Lab13 $
```

4. Verify the user in AWS Console:

Navigate to IAM → Users in AWS Console

Click on "loadbalancer" user

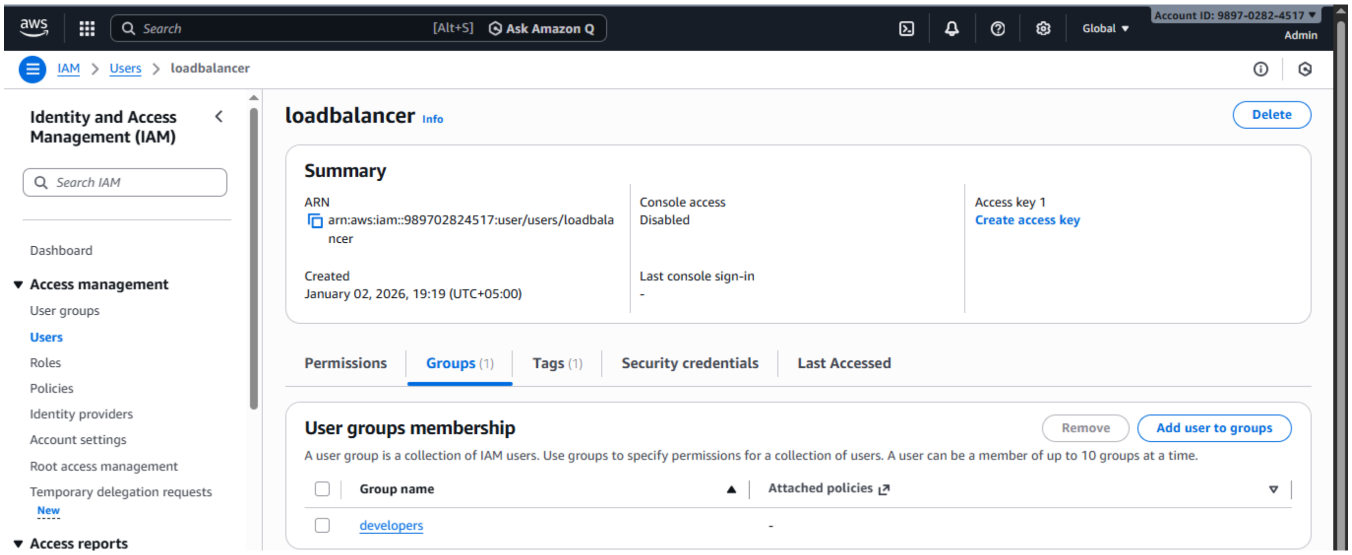

IAM > Users

Users (2) 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	7 minutes ago	-	31 days
loadbalancer	/users/	1	-	-	-

Check the "Groups" tab


IAM > Users > loadbalancer

loadbalancer Info

Summary

ARN
arn:aws:iam::989702824517:user/users/loadbalancer

Console access
Disabled

Access key 1
Create access key

Created
January 02, 2026, 19:19 (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:

```
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
  }
}
resource "aws_iam_group_policy_attachment" "developer_ec2_fullaccess" {
    group = aws_iam_group.developers.name
```

    policy_arn = "arn:aws:iam::aws:policy/AmazonEC2FullAccess"

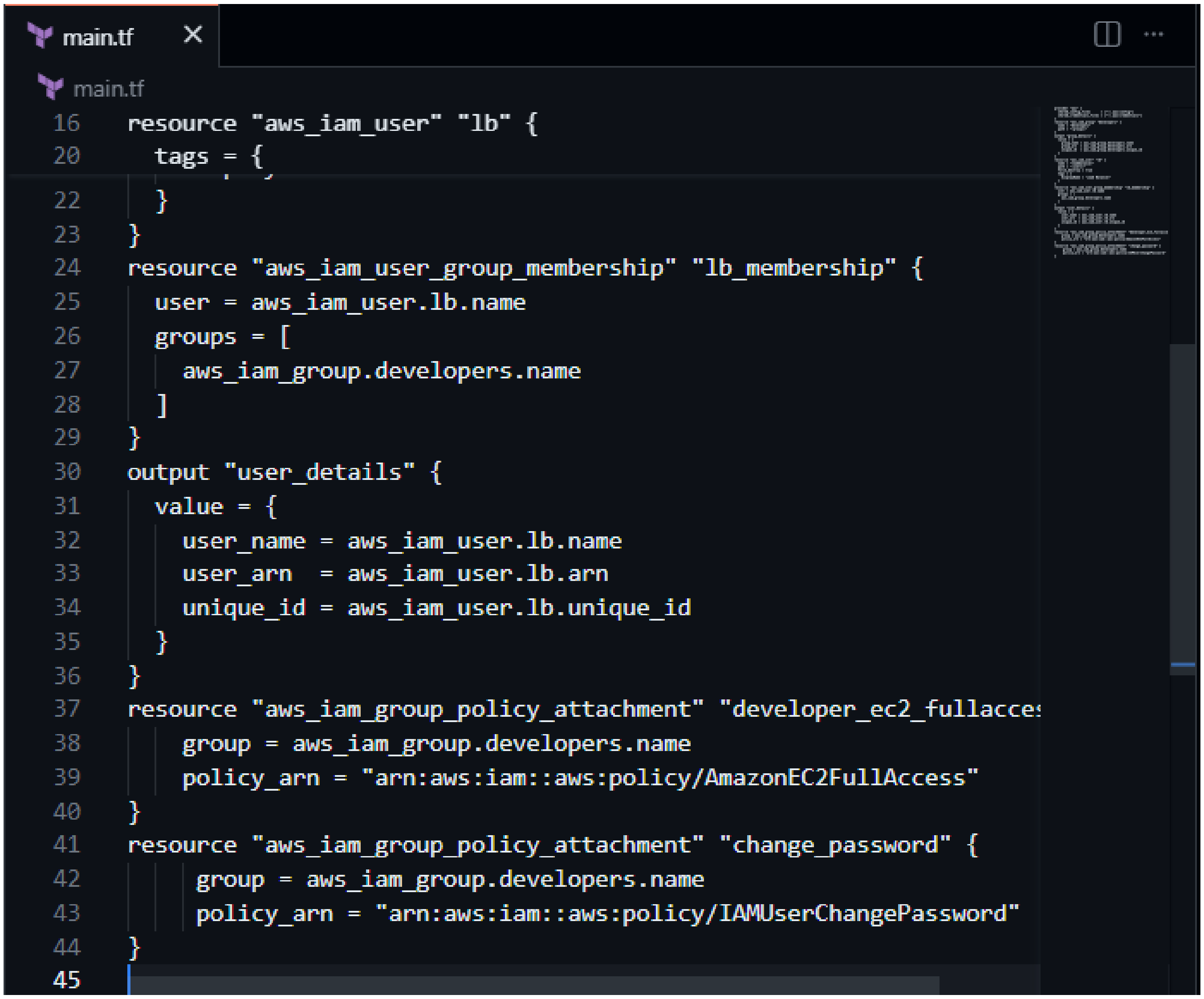
}

resource "aws_iam_group_policy_attachment" "change_password" {

    group = aws_iam_group.developers.name

    policy_arn = "arn:aws:iam::aws:policy/IAMUserChangePassword"

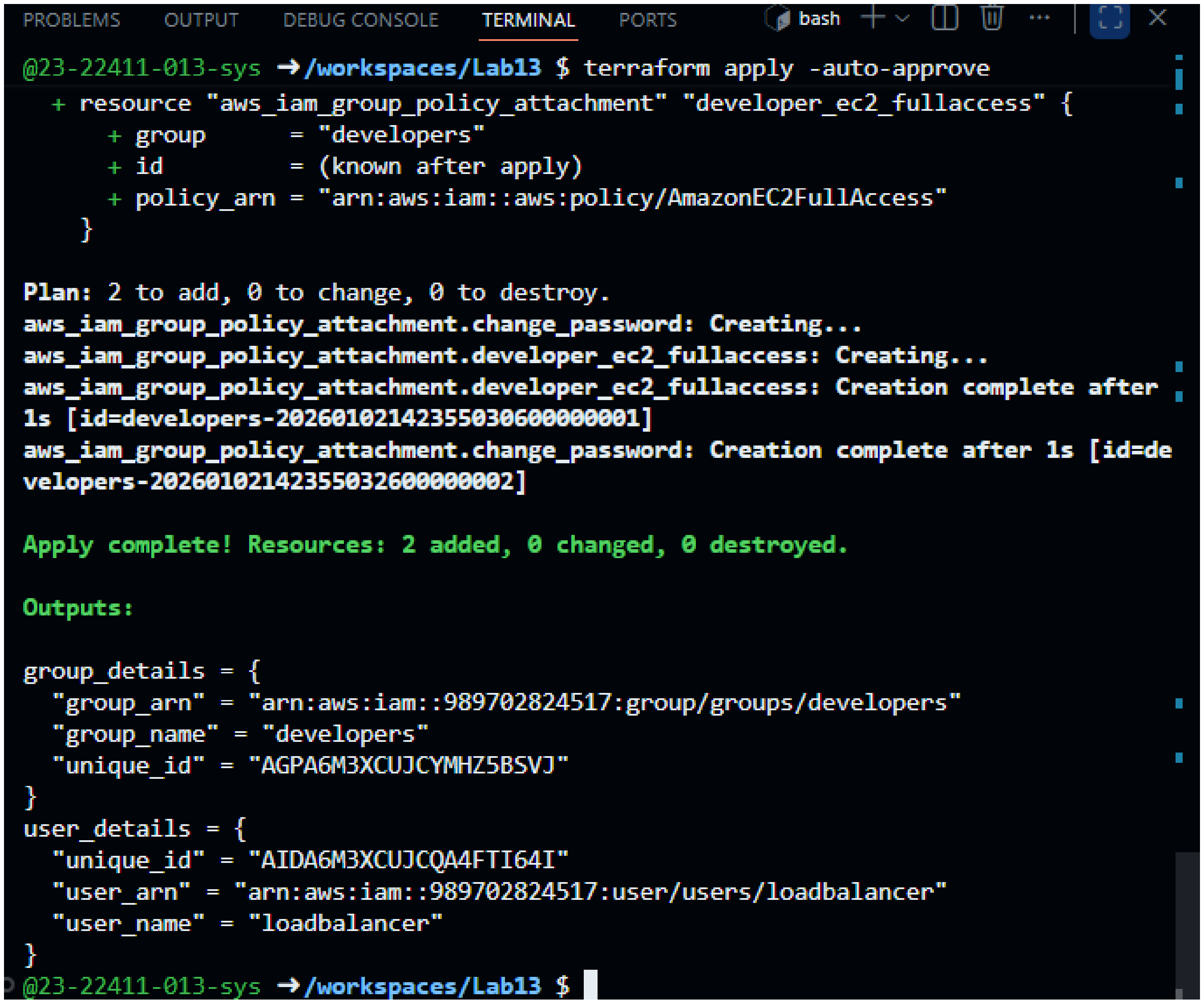
}


main.tf

```
resource "aws_iam_user" "lb" {
  tags = {
  }
}
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_fullacce
    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 apply -auto-approve

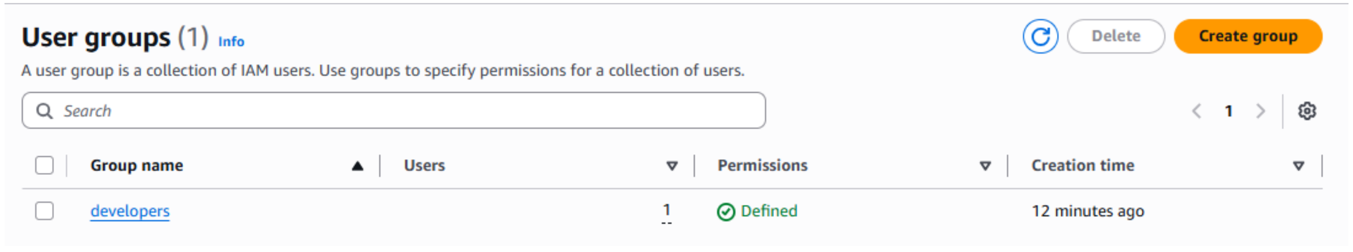


```
@23-22411-013-sys →/workspaces/Lab13 $ terraform apply -auto-approve
  + resource "aws_iam_group_policy_attachment" "developer_ec2_fullaccess" {
      + group      = "developers"
      + id         = (known after apply)
      + policy_arn = "arn:aws:iam::aws:policy/AmazonEC2FullAccess"
    }

Plan: 2 to add, 0 to change, 0 to destroy.
aws_iam_group_policy_attachment.change_password: Creating...
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Creating...
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Creation complete after
1s [id=developers-20260102142355030600000001]
aws_iam_group_policy_attachment.change_password: Creation complete after 1s [id=de
velopers-20260102142355032600000002]

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

Outputs:

group_details = {
  "group_arn" = "arn:aws:iam::989702824517:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA6M3XCUJCYMHZ5BSVJ"
}
user_details = {
  "unique_id" = "AIDA6M3XCUJCQA4FTI64I"
  "user_arn" = "arn:aws:iam::989702824517:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@23-22411-013-sys →/workspaces/Lab13 $
```

3. Verify policies in AWS Console:

Navigate to IAM → Groups → developers

Click on "Permissions" tab



User groups (1) Info

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

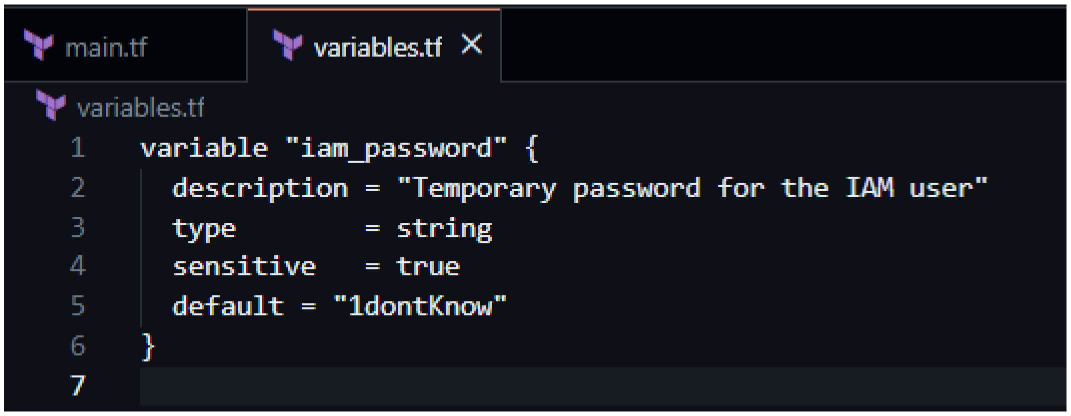
Group name	Users	Permissions	Creation time
developers	1	Defined	12 minutes ago

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:

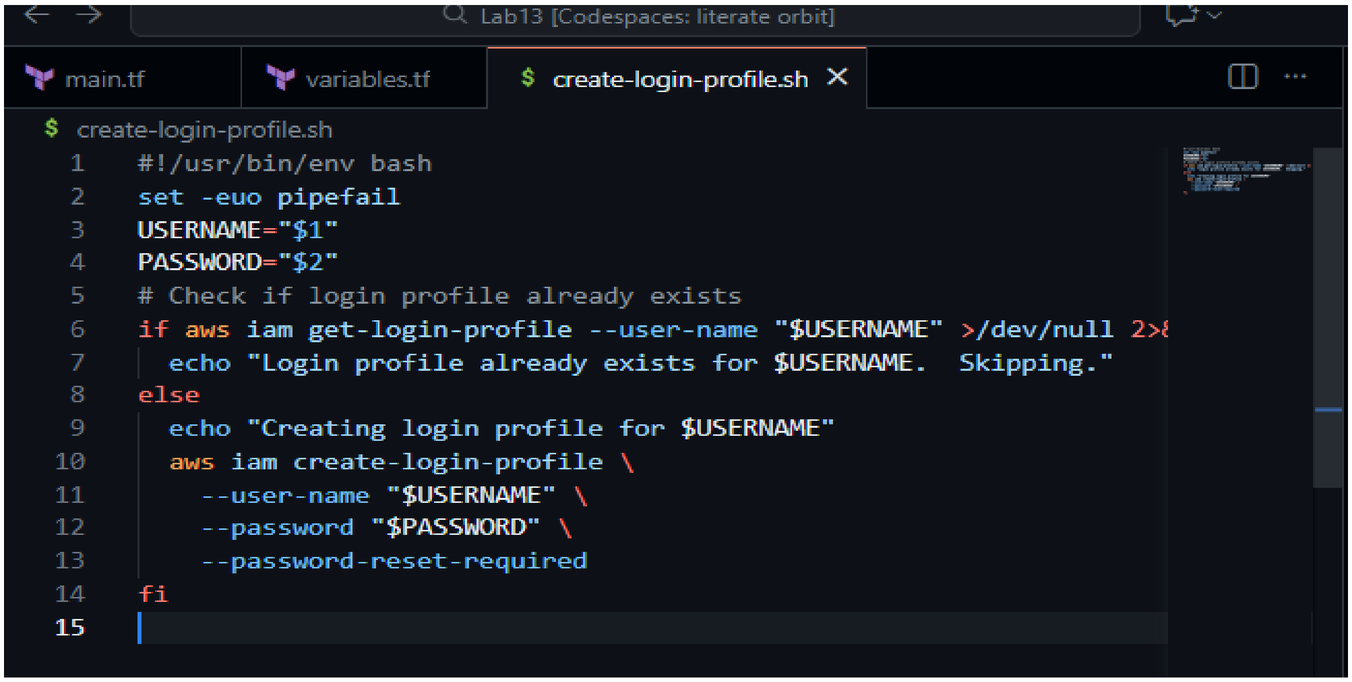
```
variable "iam_password" {
  description = "Temporary password for the IAM user"
  type        = string
  sensitive   = true
  default = "1dontKnow"
}
```



```
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:

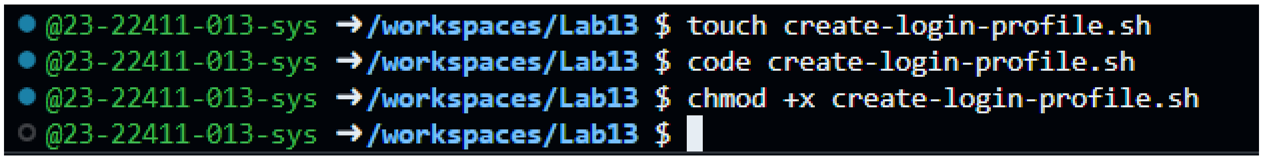
```
#!/usr/bin/env bash
set -euo pipefail
USERNAME="$1"
PASSWORD="$2"
# Check if login profile already exists
if aws iam get-login-profile --user-name "$USERNAME" >/dev/null 2>&1; then
  echo "Login profile already exists for $USERNAME.  Skipping."
else
  echo "Creating login profile for $USERNAME"
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



```
#!/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>&
  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:

chmod +x create-login-profile.sh

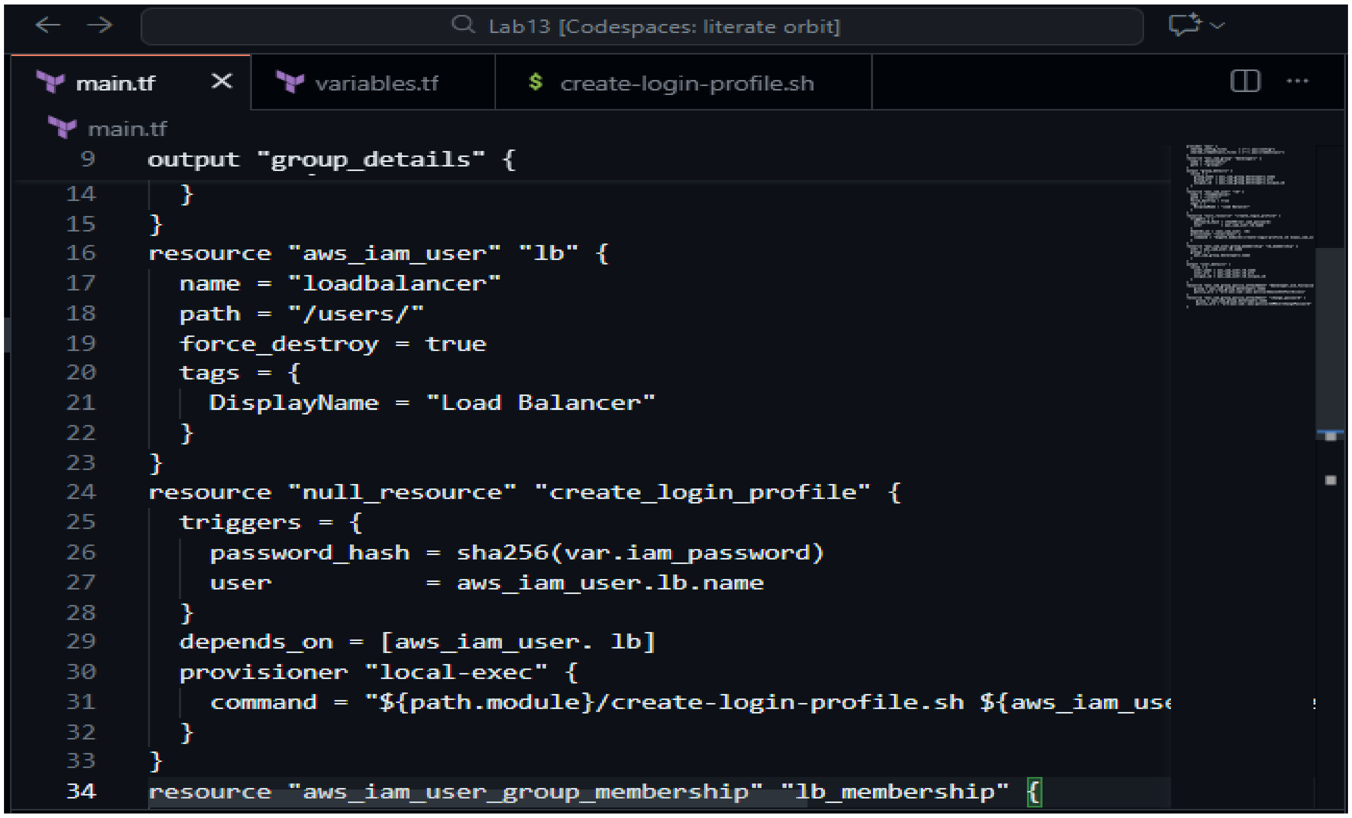


```
@23-22411-013-sys →/workspaces/Lab13 $ touch create-login-profile.sh
@23-22411-013-sys →/workspaces/Lab13 $ code create-login-profile.sh
@23-22411-013-sys →/workspaces/Lab13 $ chmod +x create-login-profile.sh
@23-22411-013-sys →/workspaces/Lab13 $
```

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

Add this resource after the user creation:

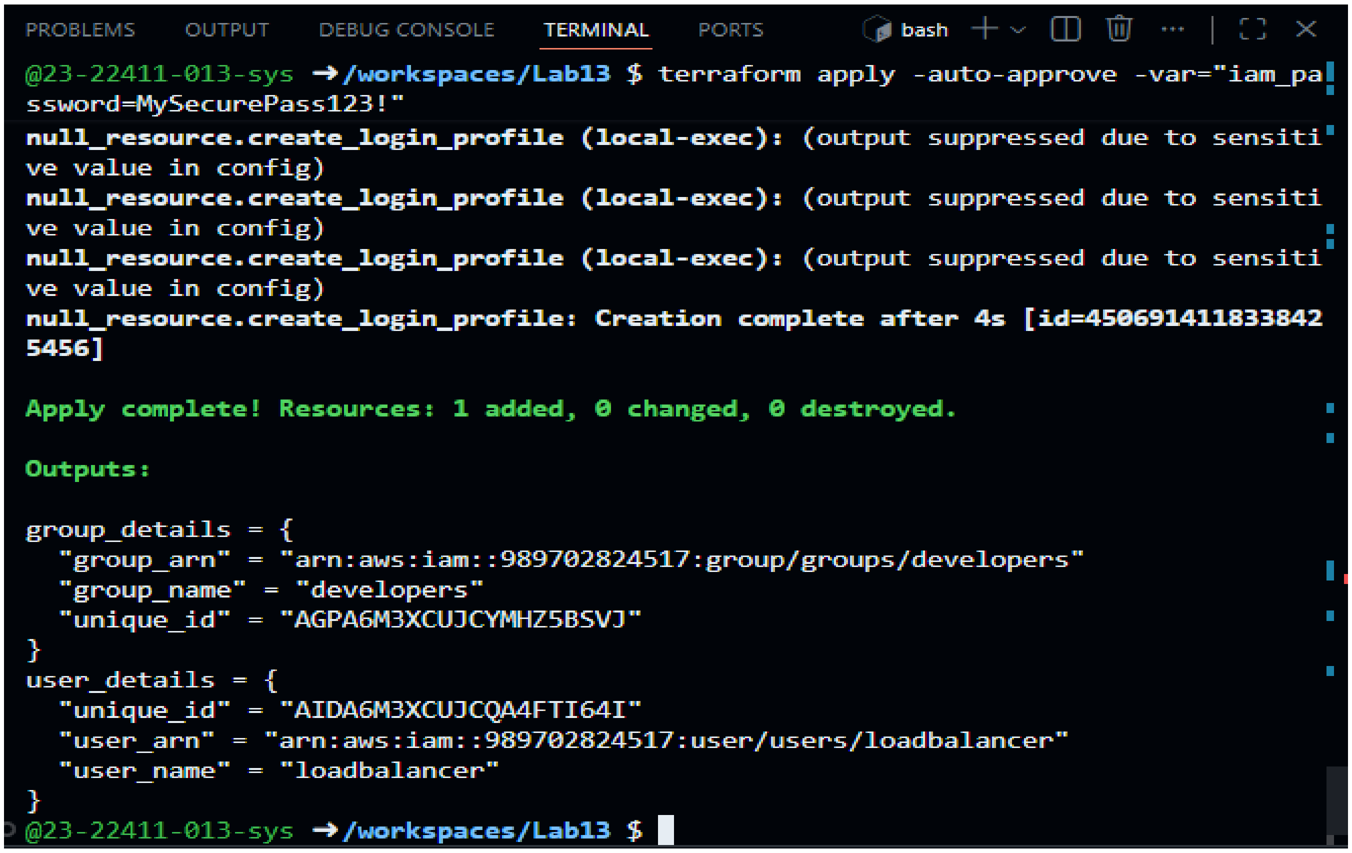
```
resource "null_resource" "create_login_profile" {
  triggers = {
    password_hash = sha256(var.iam_password)
    user          = aws_iam_user.lb.name
  }
  depends_on = [aws_iam_user. lb]
  provisioner "local-exec" {
    command = "${path.module}/create-login-profile.sh ${aws_iam_user.lb.name}
'${var.iam_password}'"
  }
}
```



```
output "group_details" {
  }
}
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_use
  }
}
resource "aws_iam_user_group_membership" "lb_membership" {
```

5. Apply the configuration with a custom password:

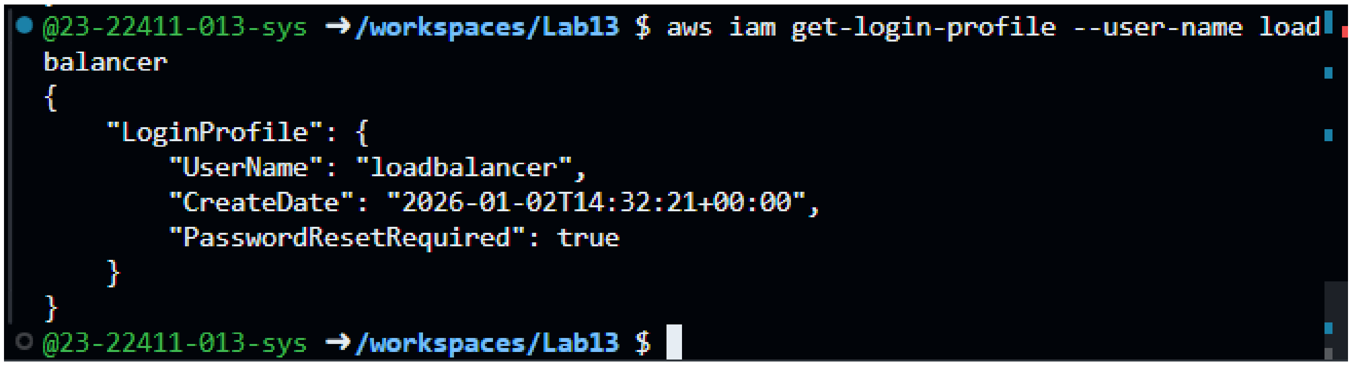
terraform apply -auto-approve -var="iam_password=MySecurePass123!"



```
@23-22411-013-sys →/workspaces/Lab13 $ terraform apply -auto-approve -var="iam_pa
ssword=MySecurePass123!"
null_resource.create_login_profile (local-exec): (output suppressed due to sensiti
ve value in config)
null_resource.create_login_profile (local-exec): (output suppressed due to sensiti
ve value in config)
null_resource.create_login_profile (local-exec): (output suppressed due to sensiti
ve value in config)
null_resource.create_login_profile: Creation complete after 4s [id=450691411833842
5456]

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

Outputs:

group_details = {
  "group_arn" = "arn:aws:iam::989702824517:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA6M3XCUJCYMHZ5BSVJ"
}
user_details = {
  "unique_id" = "AIDA6M3XCUJCQA4FTI64I"
  "user_arn" = "arn:aws:iam::989702824517:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@23-22411-013-sys →/workspaces/Lab13 $
```

6. Verify login profile creation:

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

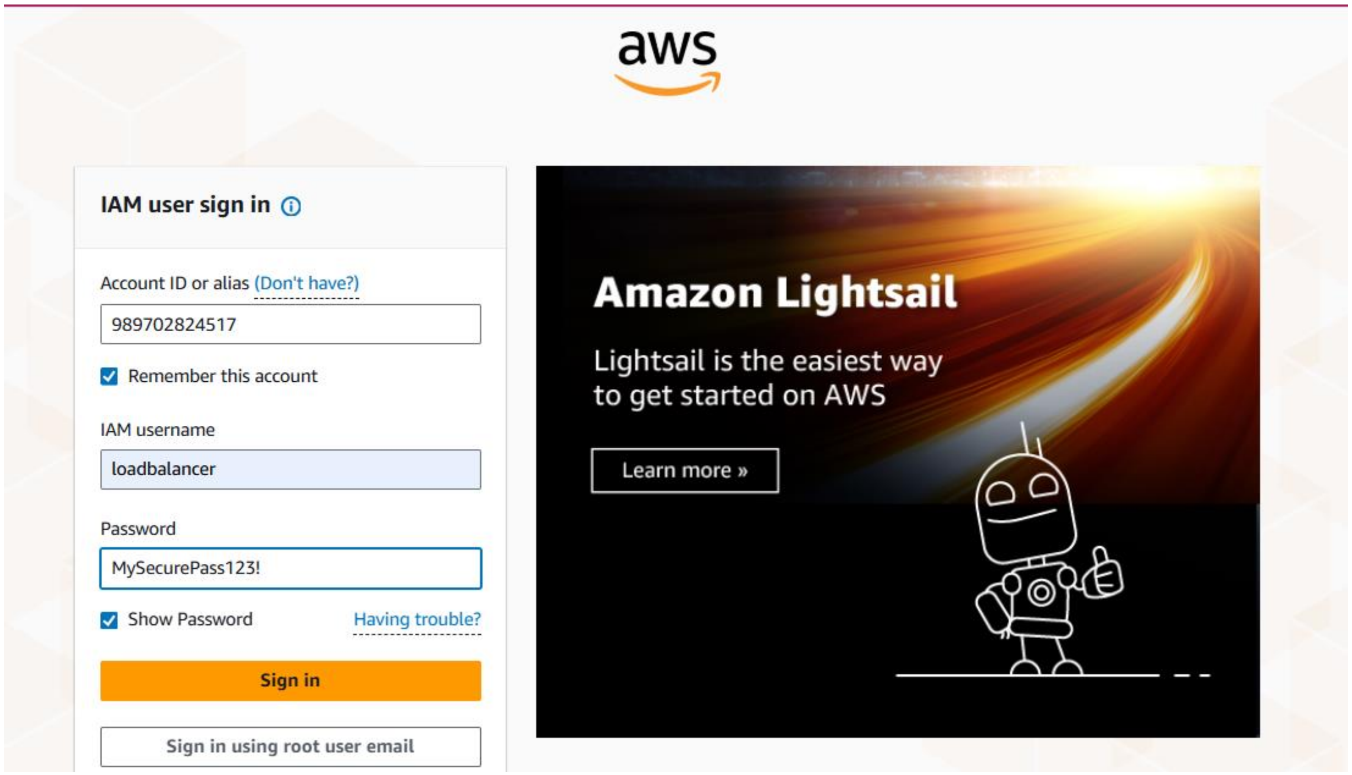


```
@23-22411-013-sys ➜ /workspaces/Lab13 $ aws iam get-login-profile --user-name load
balancer
{
    "LoginProfile": {
        "UserName": "loadbalancer",
        "CreateDate": "2026-01-02T14:32:21+00:00",
        "PasswordResetRequired": true
    }
}
@23-22411-013-sys ➜ /workspaces/Lab13 $
```

7. Test login in AWS Console:

Open AWS Console login page

Sign in as IAM user with username "loadbalancer" and the password you set



IAM user sign in

Account ID or alias (Don't have?)

989702824517

☑ Remember this account

IAM username

loadbalancer

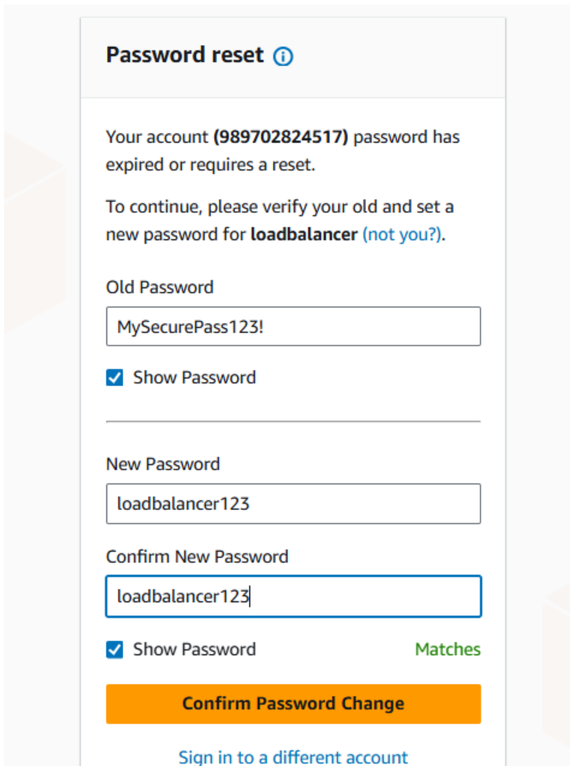
Password

MySecurePass123!

☑ Show Password

Having trouble?
Sign in
Sign in using root user email
Amazon Lightsail
Lightsail is the easiest way to get started on AWS
Learn more »

You should be prompted to change password



Password reset

Your account **(989702824517)** password has expired or requires a reset.

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

Old Password: MySecurePass123!

☑ Show Password

New Password: loadbalancer123

Confirm New Password: loadbalancer123

☑ Show Password — Matches

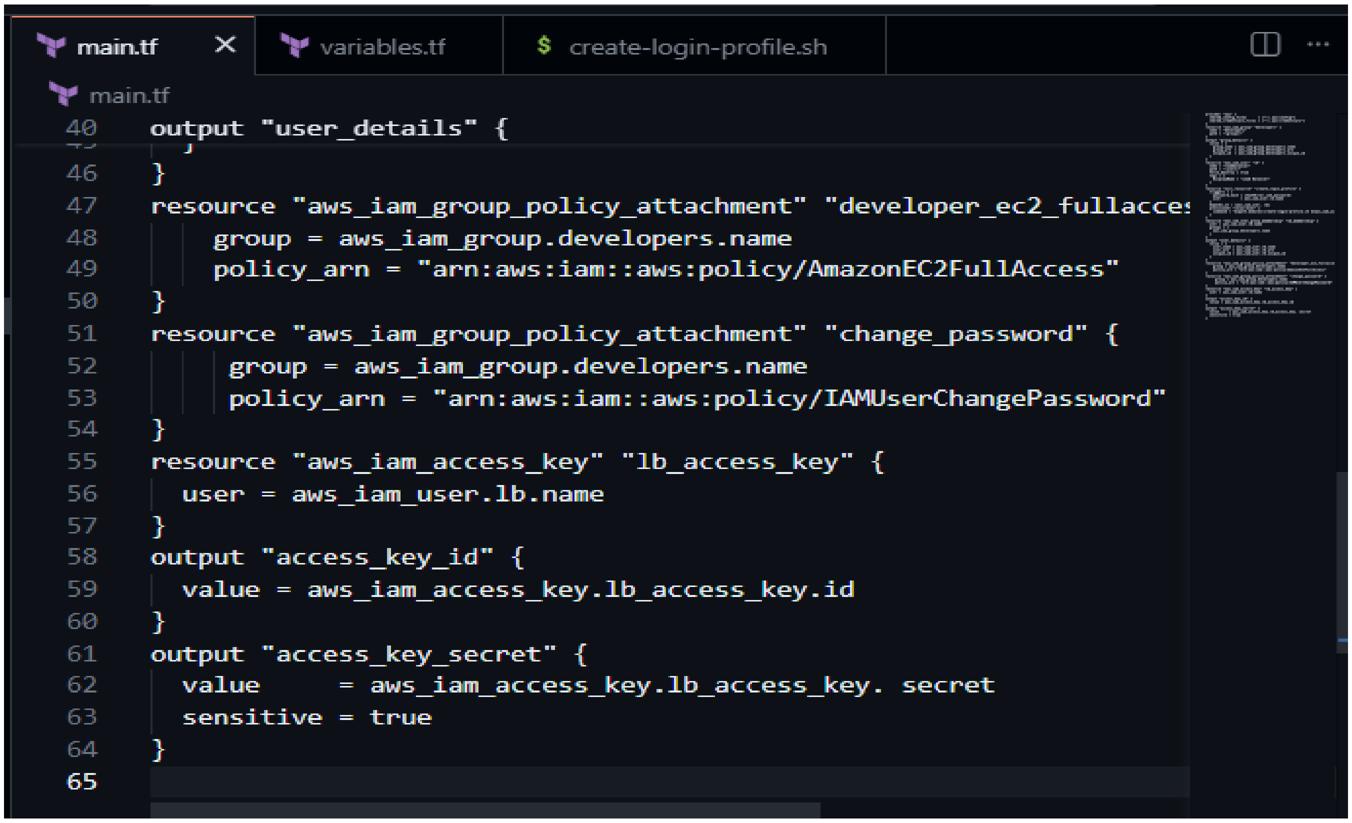
Task 5 — Generate Access Keys for IAM User

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:

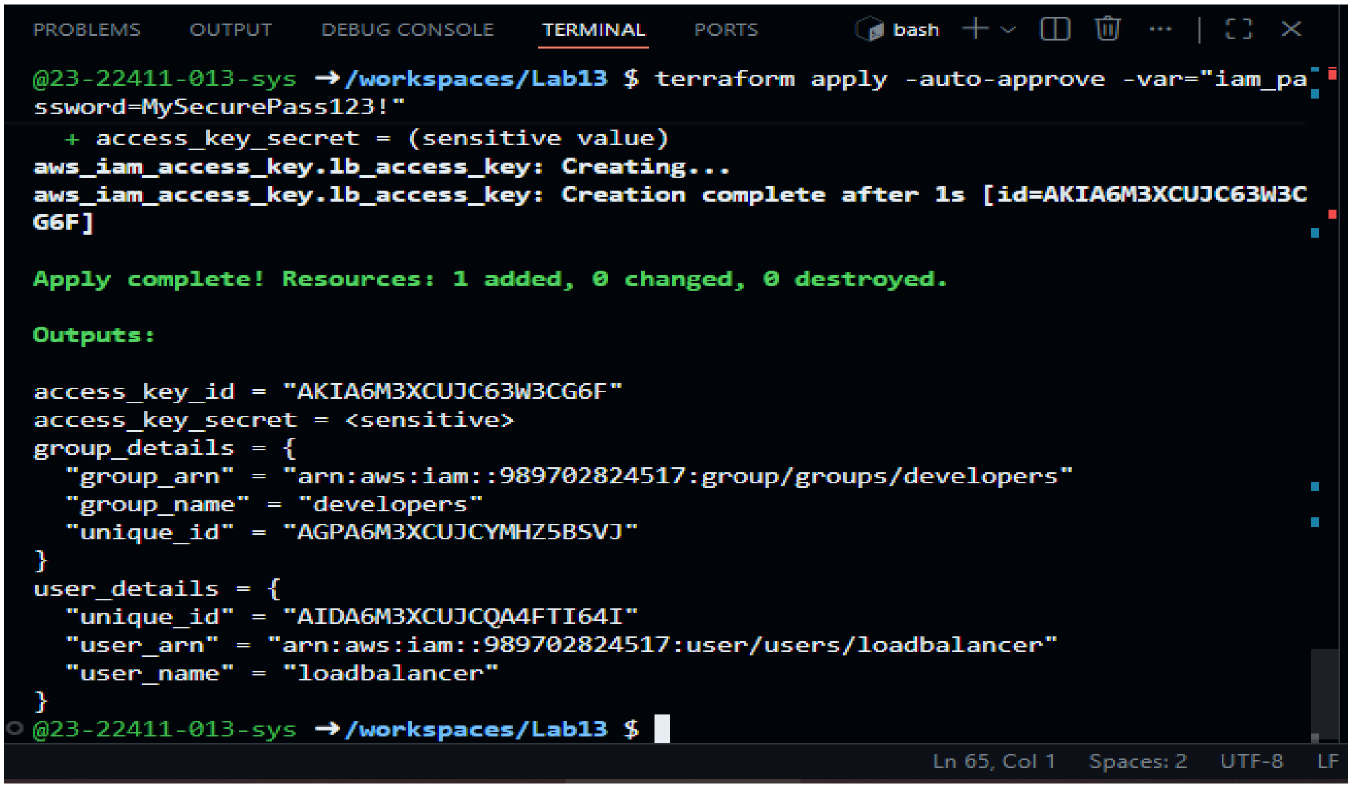
```
resource "aws_iam_access_key" "lb_access_key" {
  user = aws_iam_user.lb.name
}
output "access_key_id" {
  value = aws_iam_access_key.lb_access_key.id
}
output "access_key_secret" {
  value     = aws_iam_access_key.lb_access_key. secret
  sensitive = true
}
```



```
main.tf   variables.tf   create-login-profile.sh
main.tf
output "user_details" {
}
resource "aws_iam_group_policy_attachment" "developer_ec2_fullacces
    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:

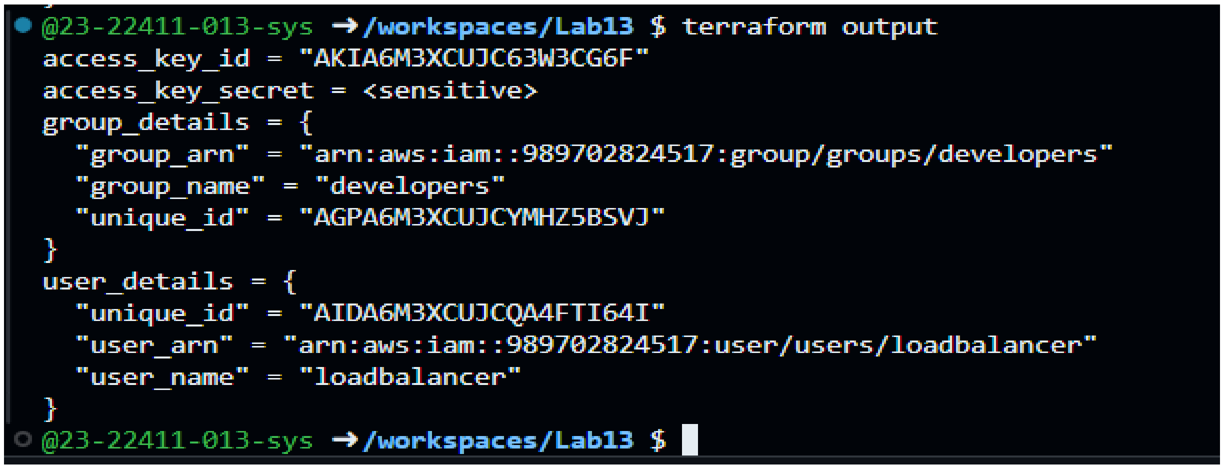
terraform apply -auto-approve -var="iam_password=MySecurePass123!"



```
@23-22411-013-sys →/workspaces/Lab13 $ terraform apply -auto-approve -var="iam_pa
ssword=MySecurePass123!"
  + access_key_secret = (sensitive value)
aws_iam_access_key.lb_access_key: Creating...
aws_iam_access_key.lb_access_key: Creation complete after 1s [id=AKIA6M3XCUJC63W3C
G6F]

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

Outputs:

access_key_id = "AKIA6M3XCUJC63W3CG6F"
access_key_secret = <sensitive>
group_details = {
  "group_arn" = "arn:aws:iam::989702824517:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA6M3XCUJCYMHZ5BSVJ"
}
user_details = {
  "unique_id" = "AIDA6M3XCUJCQA4FTI64I"
  "user_arn" = "arn:aws:iam::989702824517:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@23-22411-013-sys →/workspaces/Lab13 $
```

3. Display outputs:

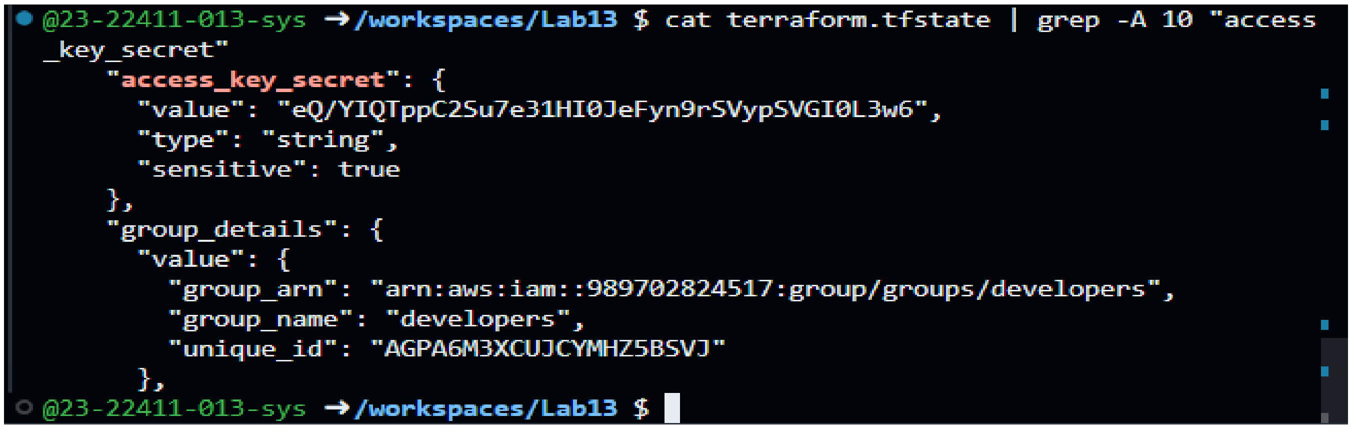
terraform output



```
@23-22411-013-sys →/workspaces/Lab13 $ terraform output
access_key_id = "AKIA6M3XCUJC63W3CG6F"
access_key_secret = <sensitive>
group_details = {
  "group_arn" = "arn:aws:iam::989702824517:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA6M3XCUJCYMHZ5BSVJ"
}
user_details = {
  "unique_id" = "AIDA6M3XCUJCQA4FTI64I"
  "user_arn" = "arn:aws:iam::989702824517:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@23-22411-013-sys →/workspaces/Lab13 $
```

4. View the secret in terraform state:

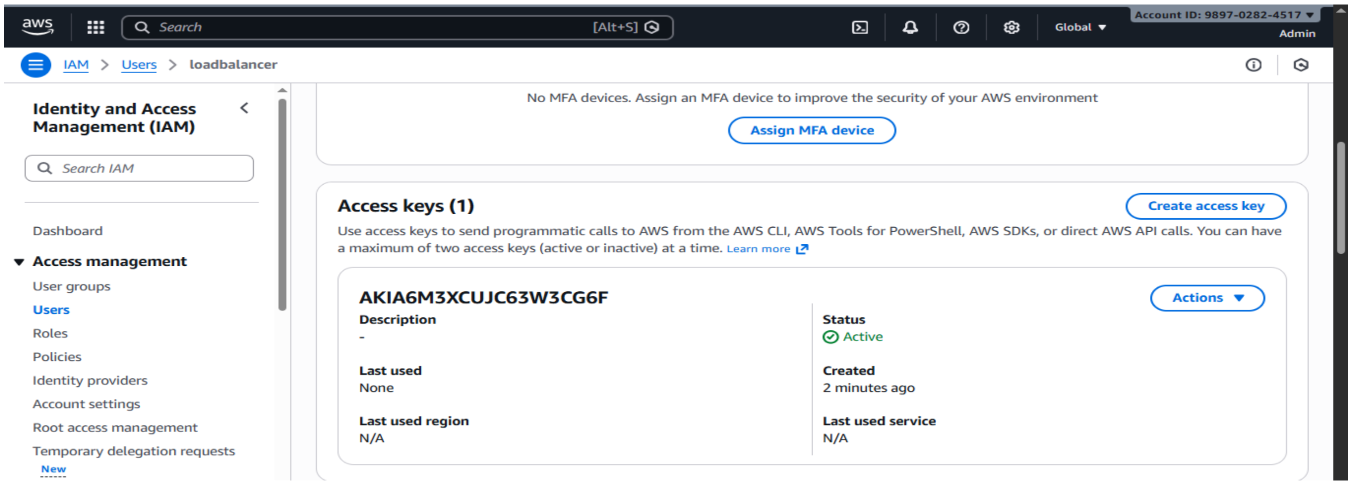
cat terraform.tfstate | grep -A 10 "access_key_secret"



```
@23-22411-013-sys →/workspaces/Lab13 $ cat terraform.tfstate | grep -A 10 "access
_key_secret"
    "access_key_secret": {
      "value": "eQ/YIQTppC2Su7e31HI0JeFyn9rSVypSVGI0L3w6",
      "type": "string",
      "sensitive": true
    },
    "group_details": {
      "value": {
        "group_arn": "arn:aws:iam::989702824517:group/groups/developers",
        "group_name": "developers",
        "unique_id": "AGPA6M3XCUJCYMHZ5BSVJ"
      },
@23-22411-013-sys →/workspaces/Lab13 $
```

5. Verify access key in AWS Console:

Navigate to IAM → Users → loadbalancer → Security credentials



No MFA devices. Assign an MFA device to improve the security of your AWS environment

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

AKIA6M3XCUJC63W3CG6F

Description -	Status Active
Last used None	Created 2 minutes ago
Last used region N/A	Last used service N/A

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

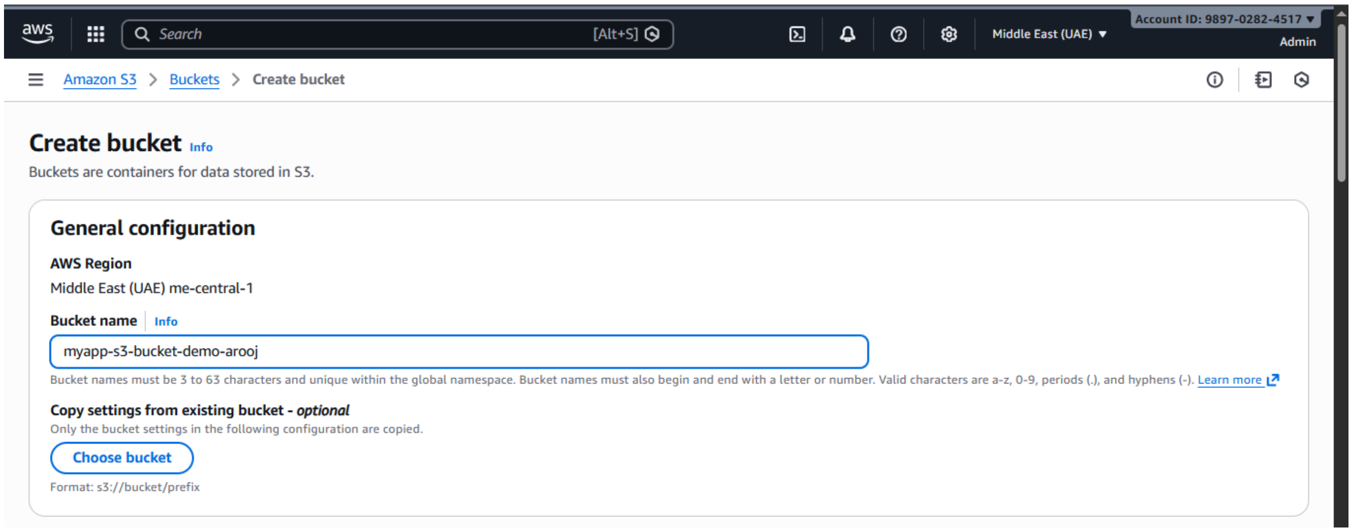
Click "Create bucket"

Bucket name: myapp-s3-bucket-demo (use a unique name if this is taken)

Enable versioning

Keep other settings as default

[Click "Create bucket"


aws
Search
[Alt+S]
Middle East (UAE) ▾
Account ID: 9897-0282-4517 ▾
Admin
Amazon S3 > Buckets > Create bucket
Info
Choose bucket

Create bucket

Buckets are containers for data stored in S3.

General configuration

AWS Region

Middle East (UAE) me-central-1

Bucket name

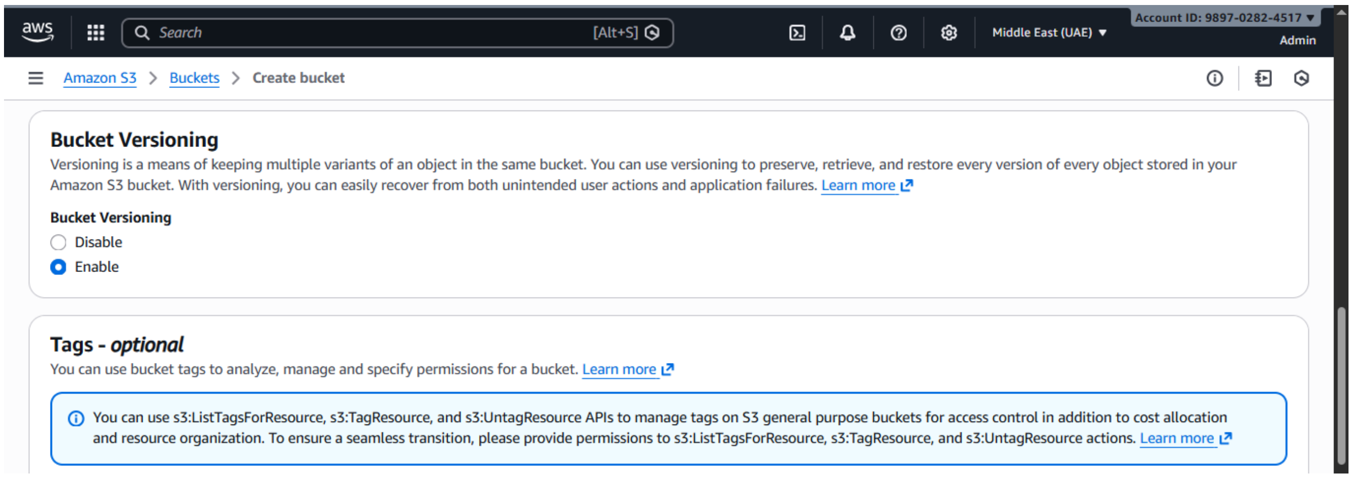
myapp-s3-bucket-demo-arooj

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.

Format: s3://bucket/prefix


aws
Search
[Alt+S]
Middle East (UAE) ▾
Account ID: 9897-0282-4517 ▾
Admin
Amazon S3 > Buckets > Create bucket

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

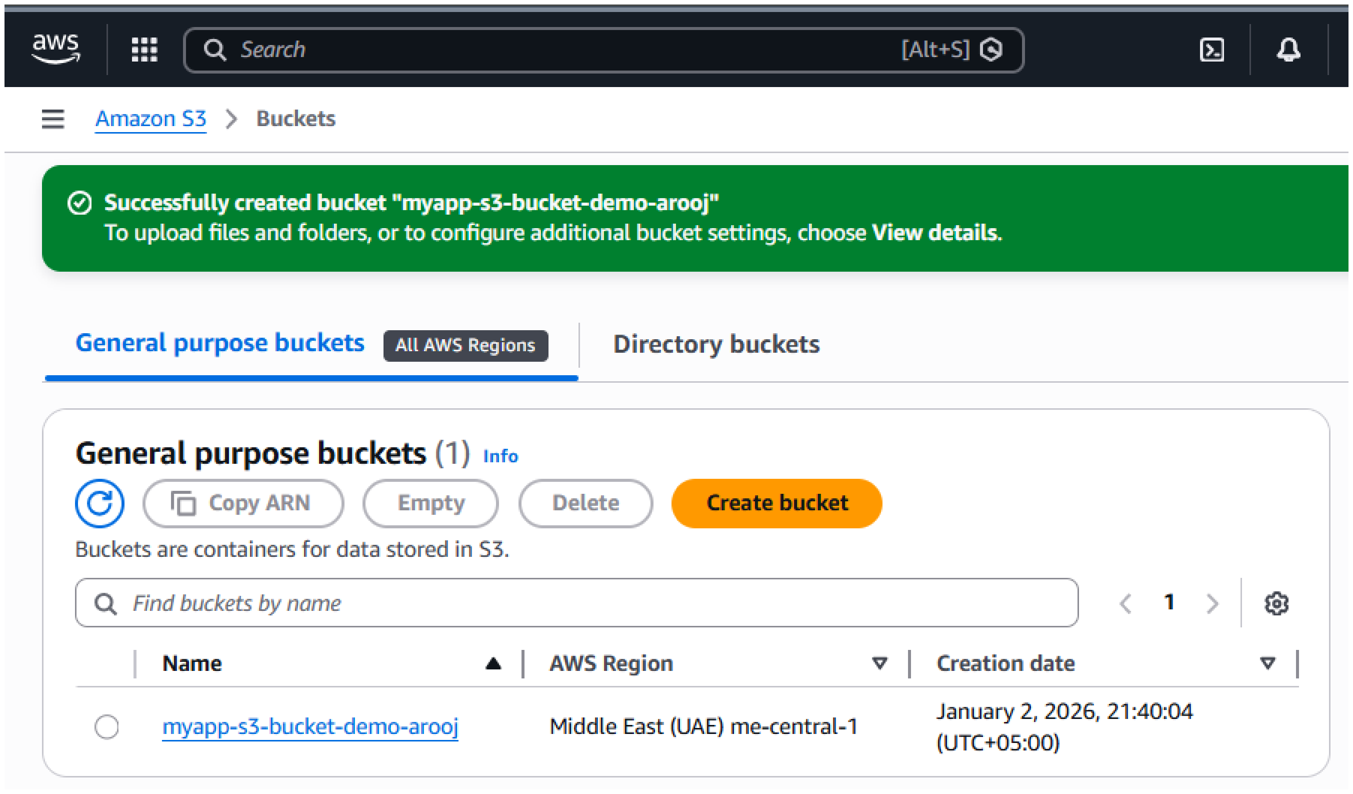
Bucket Versioning

Disable [✓ Enable]

Tags - *optional*

You can use bucket tags to analyze, manage and specify permissions for a bucket. Learn more

You can use s3:ListTagsForResource, s3:TagResource, and s3:UntagResource APIs to manage tags on S3 general purpose buckets for access control in addition to cost allocation and resource organization. To ensure a seamless transition, please provide permissions to s3:ListTagsForResource, s3:TagResource, and s3:UntagResource actions. Learn more


Amazon S3 > Buckets

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

General purpose buckets (1)

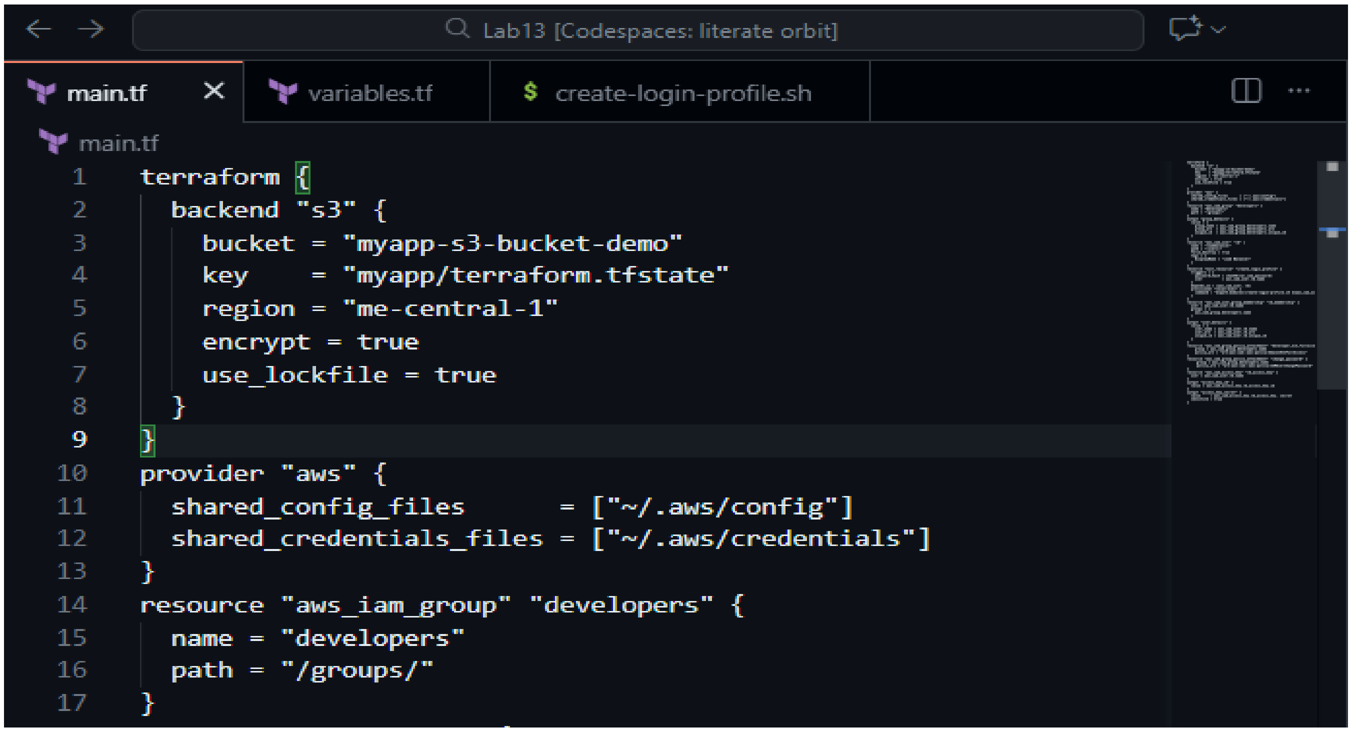
Buckets are containers for data stored in S3.

Name	AWS Region	Creation date
myapp-s3-bucket-demo-arooj	Middle East (UAE) me-central-1	January 2, 2026, 21:40:04 (UTC+05:00)

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

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

```
terraform {
 backend "s3" {
  bucket = "myapp-s3-bucket-demo"
  key    = "myapp/terraform.tfstate"
  region = "me-central-1"
  encrypt = true
  use_lockfile = true
 }
}
```


Lab13 [Codespaces: literate orbit]

main.tf | variables.tf | create-login-profile.sh

main.tf

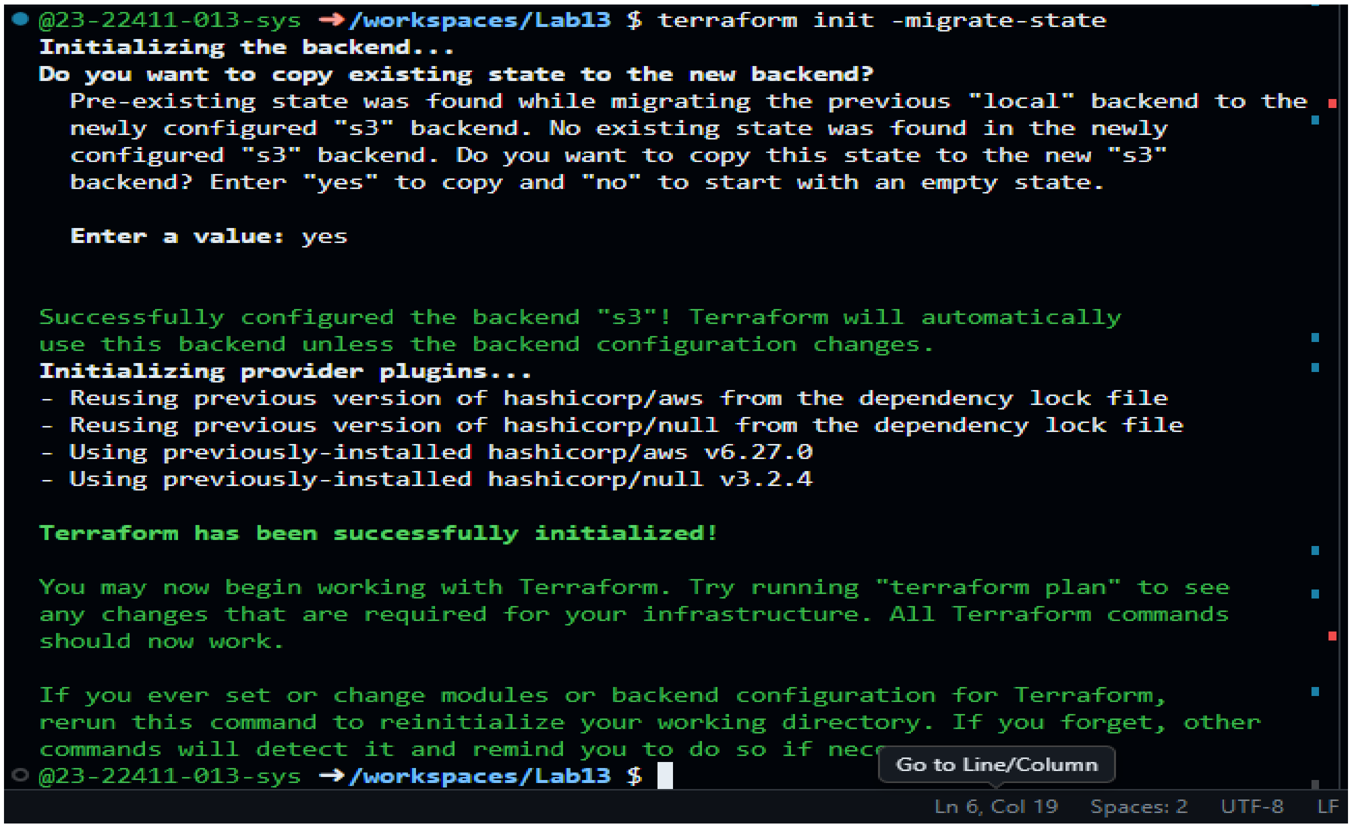
```
terraform {
  backend "s3" {
    bucket = "myapp-s3-bucket-demo"
    key    = "myapp/terraform.tfstate"
    region = "me-central-1"
    encrypt = true
    use_lockfile = true
  }
}
provider "aws" {
  shared_config_files      = ["~/.aws/config"]
  shared_credentials_files = ["~/.aws/credentials"]
}
resource "aws_iam_group" "developers" {
  name = "developers"
  path = "/groups/"
}
```

- **Save screenshot as:** task6_main_tf_backend. png — main.tf showing backend configuration.

3. Reinitialize Terraform with the backend:

terraform init -migrate-state

- Type yes when prompted to migrate state

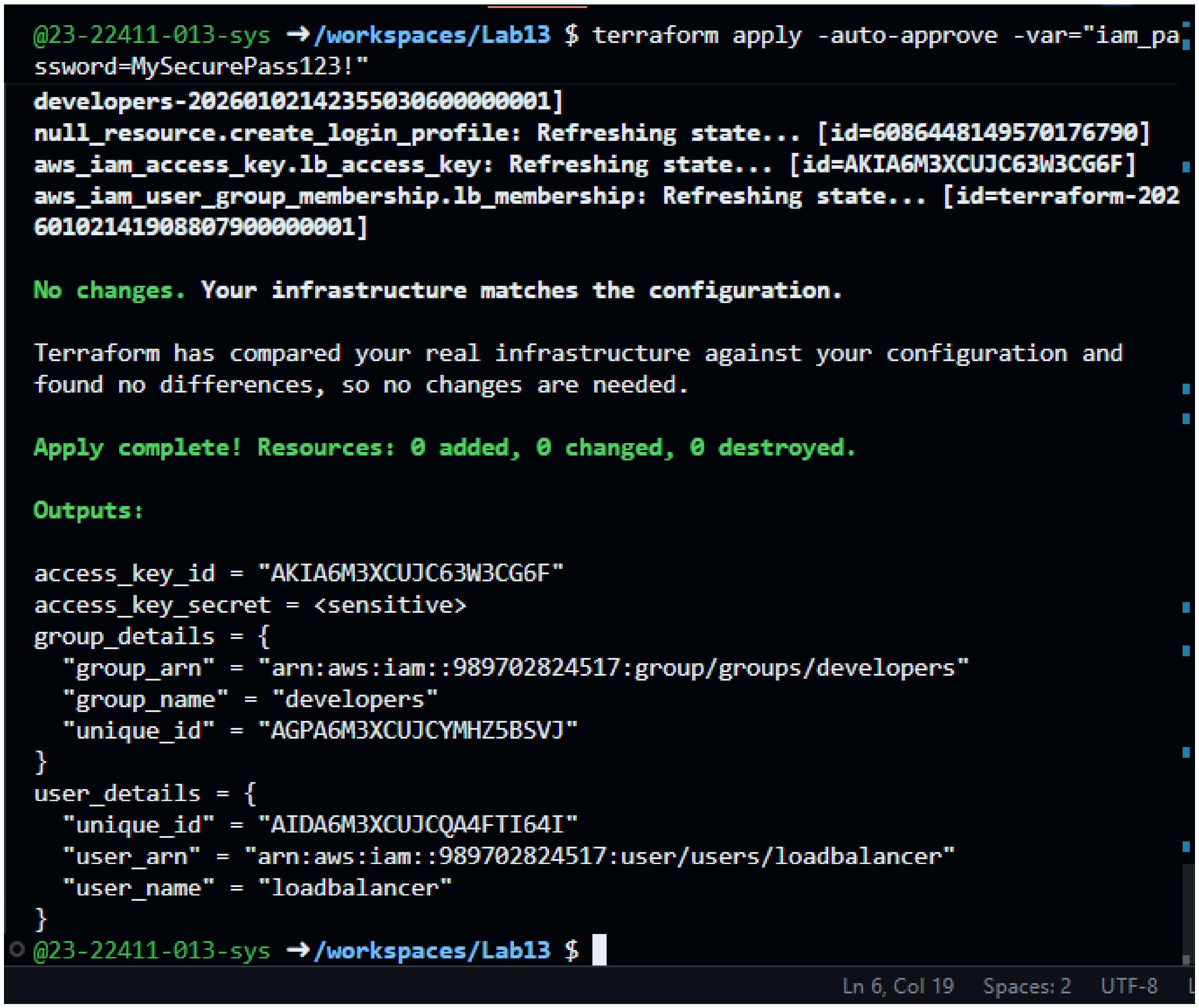


```
@23-22411-013-sys →/workspaces/Lab13 $ terraform init -migrate-state
Initializing the backend...
Do you want to copy existing state to the new backend?
  Pre-existing state was found while migrating the previous "local" backend to the
  newly configured "s3" backend. No existing state was found in 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 nec
@23-22411-013-sys →/workspaces/Lab13 $
```

- **Save screenshot as:** task6_terraform_init_migrate.png — terraform init output showing state migration.

4. Apply the configuration:

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

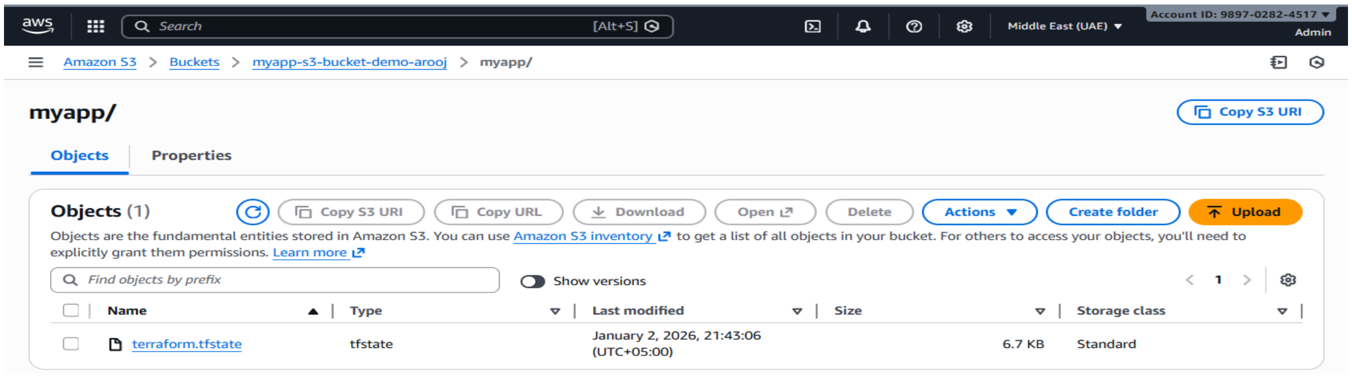


```
@23-22411-013-sys ➜/workspaces/Lab13 $ terraform apply -auto-approve -var="iam_pa
ssword=MySecurePass123!"
developers-20260102142355030600000001]
null_resource.create_login_profile: Refreshing state... [id=6086448149570176790]
aws_iam_access_key.lb_access_key: Refreshing state... [id=AKIA6M3XCUJC63W3CG6F]
aws_iam_user_group_membership.lb_membership: Refreshing state... [id=terraform-202
60102141908807900000001]

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 = "AKIA6M3XCUJC63W3CG6F"
access_key_secret = <sensitive>
group_details = {
  "group_arn" = "arn:aws:iam::989702824517:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA6M3XCUJCYMHZ5BSVJ"
}
user_details = {
  "unique_id" = "AIDA6M3XCUJCQA4FTI64I"
  "user_arn" = "arn:aws:iam::989702824517:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@23-22411-013-sys ➜/workspaces/Lab13 $
```

5. Verify state file in S3:

Navigate to S3 → myapp-s3-bucket-demo → myapp/

You should see terraform.tfstate file



Amazon S3 > Buckets > myapp-s3-bucket-demo-arooj > myapp/

myapp/

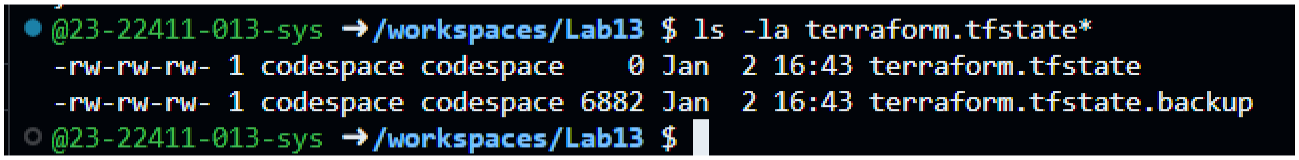
Objects (1)

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 2, 2026, 21:43:06 (UTC+05:00)	6.7 KB	Standard

6. Check local state file:

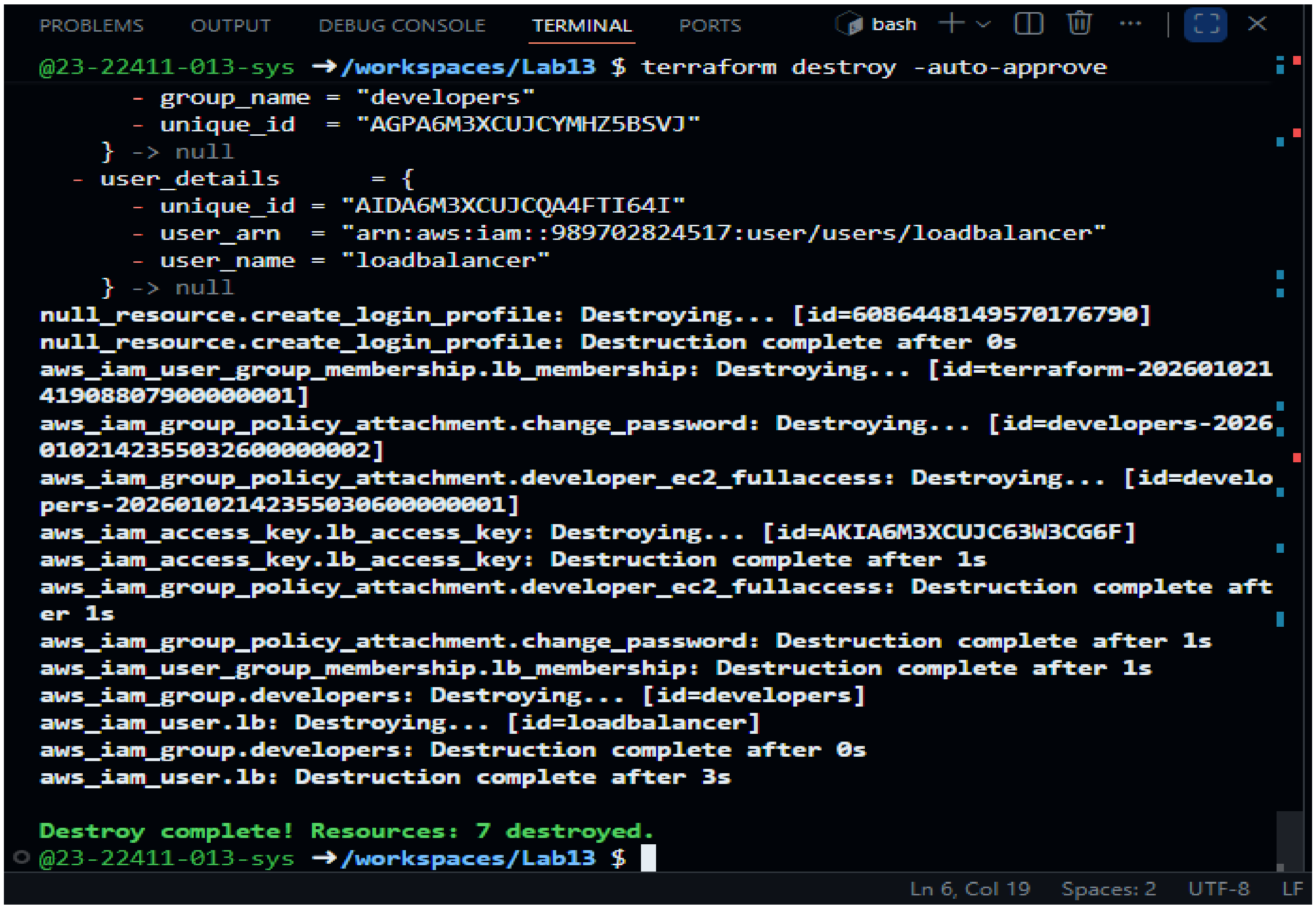
ls -la terraform.tfstate*



```
@23-22411-013-sys ➜ /workspaces/Lab13 $ ls -la terraform.tfstate*
-rw-rw-rw- 1 codespace codespace    0 Jan  2 16:43 terraform.tfstate
-rw-rw-rw- 1 codespace codespace 6882 Jan  2 16:43 terraform.tfstate.backup
@23-22411-013-sys ➜ /workspaces/Lab13 $ 
```

7. Destroy resources and verify state change:

terraform destroy -auto-approve

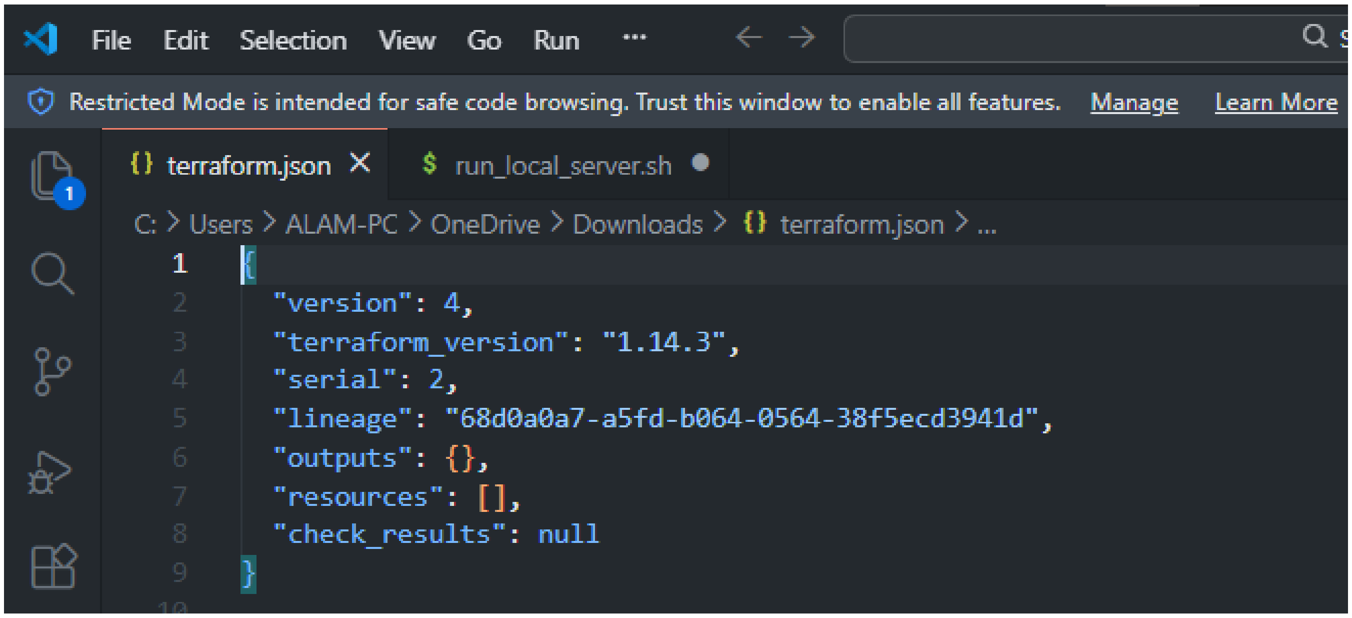


```
@23-22411-013-sys ➜ /workspaces/Lab13 $ terraform destroy -auto-approve
          - group_name = "developers"
          - unique_id  = "AGPA6M3XCUJCYMHZ5BSVJ"
        } -> null
      - user_details       = {
          - unique_id = "AIDA6M3XCUJCQA4FTI64I"
          - user_arn  = "arn:aws:iam::989702824517:user/users/loadbalancer"
          - user_name = "loadbalancer"
        } -> null
null_resource.create_login_profile: Destroying... [id=6086448149570176790]
null_resource.create_login_profile: Destruction complete after 0s
aws_iam_user_group_membership.lb_membership: Destroying... [id=terraform-202601021
41908807900000001]
aws_iam_group_policy_attachment.change_password: Destroying... [id=developers-2026
0102142355032600000002]
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Destroying... [id=develo
pers-20260102142355030600000001]
aws_iam_access_key.lb_access_key: Destroying... [id=AKIA6M3XCUJC63W3CG6F]
aws_iam_access_key.lb_access_key: Destruction complete after 1s
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Destruction complete aft
er 1s
aws_iam_group_policy_attachment.change_password: Destruction complete after 1s
aws_iam_user_group_membership.lb_membership: Destruction complete after 1s
aws_iam_group.developers: Destroying... [id=developers]
aws_iam_user.lb: Destroying... [id=loadbalancer]
aws_iam_group.developers: Destruction complete after 0s
aws_iam_user.lb: Destruction complete after 3s

Destroy complete! Resources: 7 destroyed.
@23-22411-013-sys ➜ /workspaces/Lab13 $ 
```

8. Verify updated state in S3:

Refresh S3 bucket view

Check the terraform.tfstate file (it should show empty resources)


File Edit Selection View Go Run ···
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
terraform.json
run_local_server.sh
C: > Users > ALAM-PC > OneDrive > Downloads > terraform.json > ...

```
{
  "version": 4,
  "terraform_version": "1.14.3",
  "serial": 2,
  "lineage": "68d0a0a7-a5fd-b064-0564-38f5ecd3941d",
  "outputs": {},
  "resources": [],
  "check_results": null
}
```

Task 7 — Create Multiple Users from CSV File

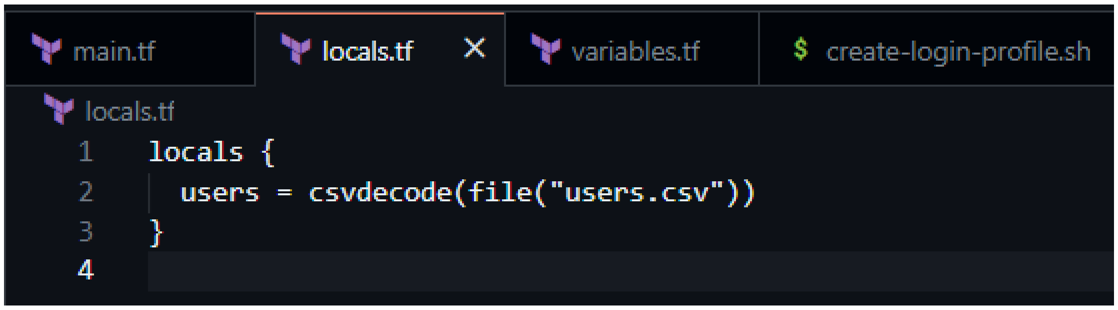
In this task, you will create multiple IAM users dynamically from a CSV file.

1. Create locals.tf file:

locals {

  users = csvdecode(file("users.csv"))

}



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

2. Create users.csv file:

user_name

Michael

Dwight

Jim

Pam

Ryan

Andy

Robert

Stanley

Kevin

Angela

Oscar

Phyllis

Toby

Kelly

Darryl

Creed

Meredith

Erin

Gabe

Jan

David

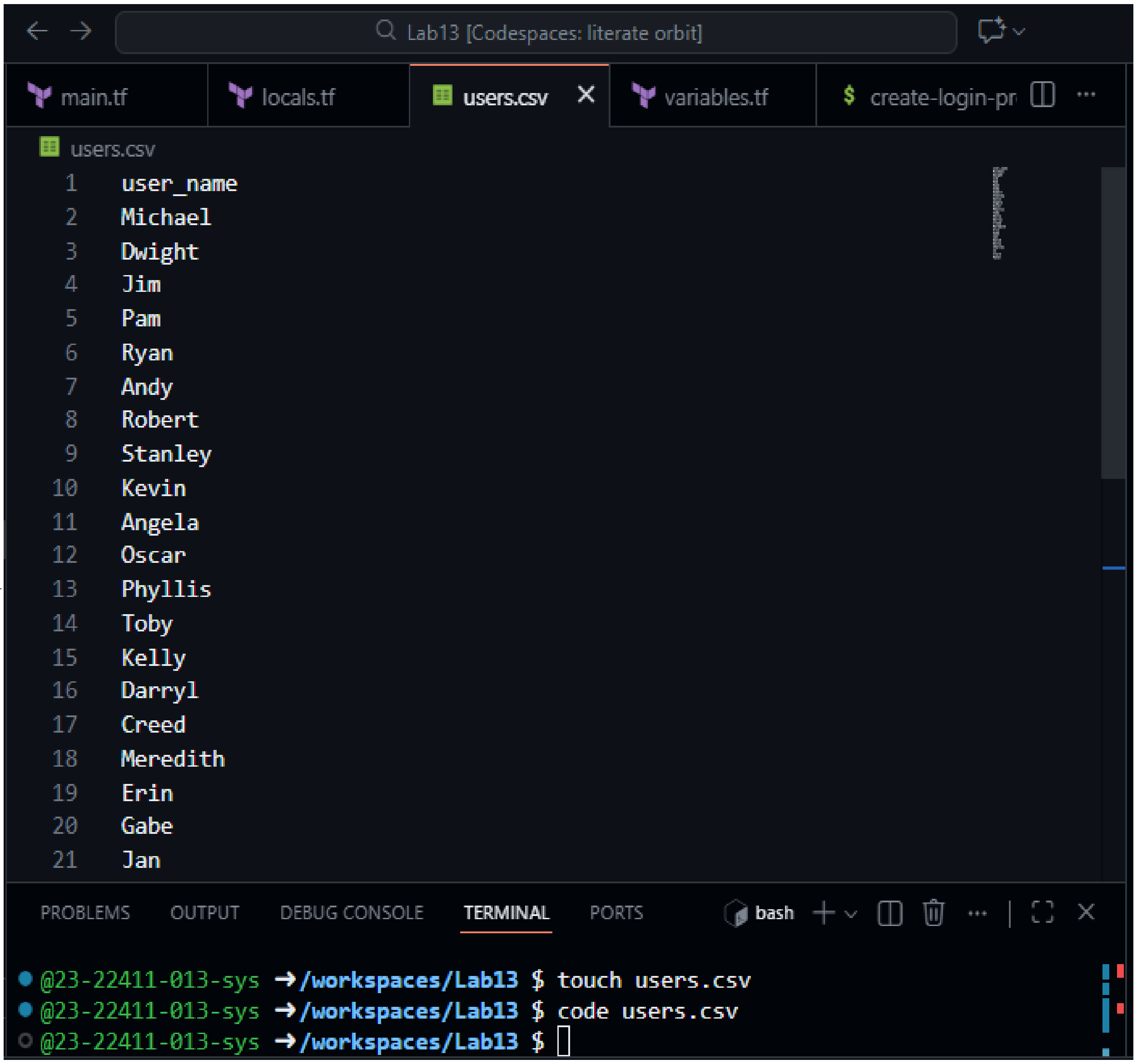
Holly

Charles

Jo

Clark

Peter



```
users.csv
user_name
Michael
Dwight
Jim
Pam
Ryan
Andy
Robert
Stanley
Kevin
Angela
Oscar
Phyllis
Toby
Kelly
Darryl
Creed
Meredith
Erin
Gabe
Jan
```

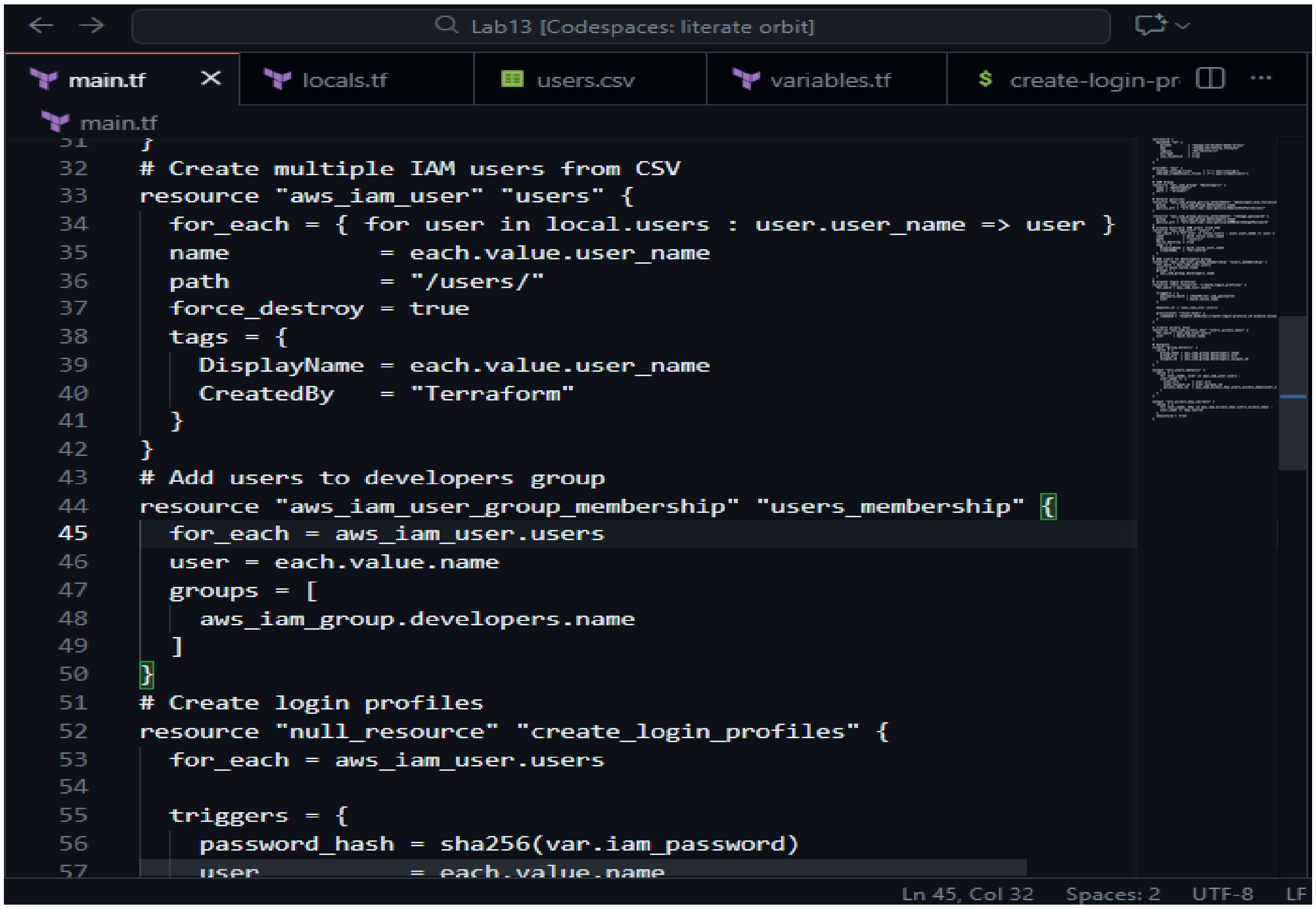
```
@23-22411-013-sys ➜ /workspaces/Lab13 $ touch users.csv
@23-22411-013-sys ➜ /workspaces/Lab13 $ code users.csv
@23-22411-013-sys ➜ /workspaces/Lab13 $
```

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 ${each.value.name} '${var.iam_password}'"
  }}
# Create access keys for all users
resource "aws_iam_access_key" "users_access_keys" {
  for_each = aws_iam_user. users
  user = each.value.name}
# Output all user details
output "all_users_details" {
  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 (sensitive)
output "all_access_key_secrets" {
  value = {
    for user_name, key in aws_iam_access_key.users_access_keys :  user_name => key.secret  }
  sensitive = true}
```

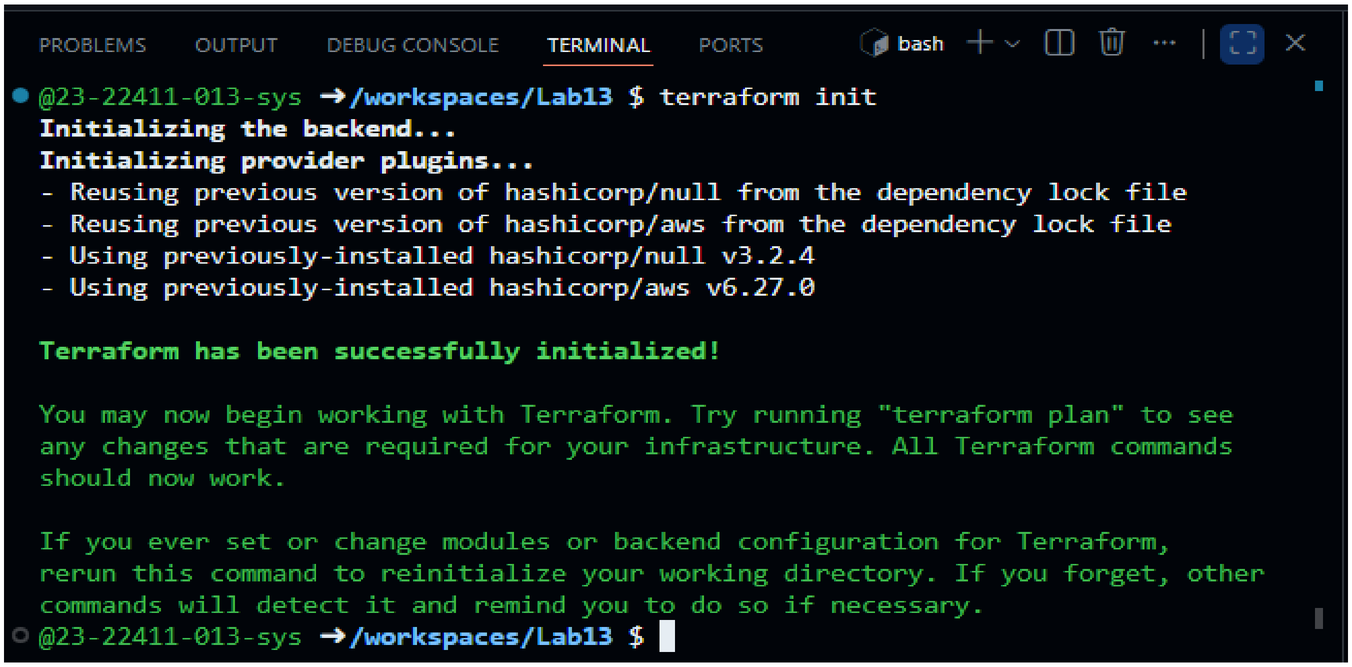


```
}
# Create multiple IAM users from CSV
resource "aws_iam_user" "users" {
  for_each = { for user in local.users : user.user_name => user }
  name          = each.value.user_name
  path          = "/users/"
  force_destroy = true
  tags = {
    DisplayName = each.value.user_name
    CreatedBy   = "Terraform"
  }
}
# Add 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
resource "null_resource" "create_login_profiles" {
  for_each = aws_iam_user.users

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

- **Save screenshot as:** task7_main_tf_multiple_users.png — main.tf showing multiple user resources.

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

```
terraform init
```

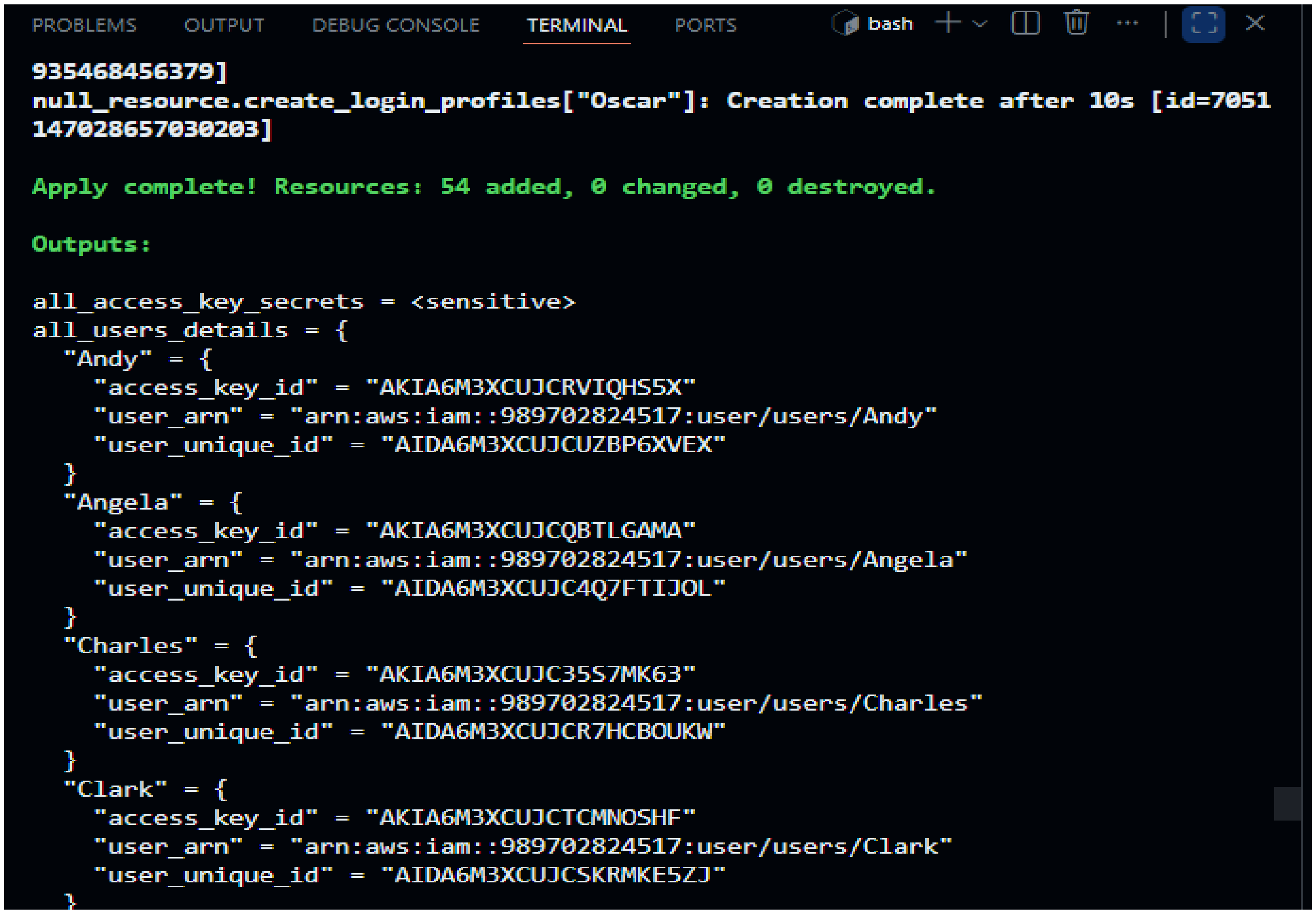


```
@23-22411-013-sys ➜/workspaces/Lab13 $ terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/null from the dependency lock file
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/null 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.
@23-22411-013-sys ➜/workspaces/Lab13 $
```

- **Save screenshot as:** task7_terraform_init.png — terraform init output.

5. Apply the configuration to create all users:

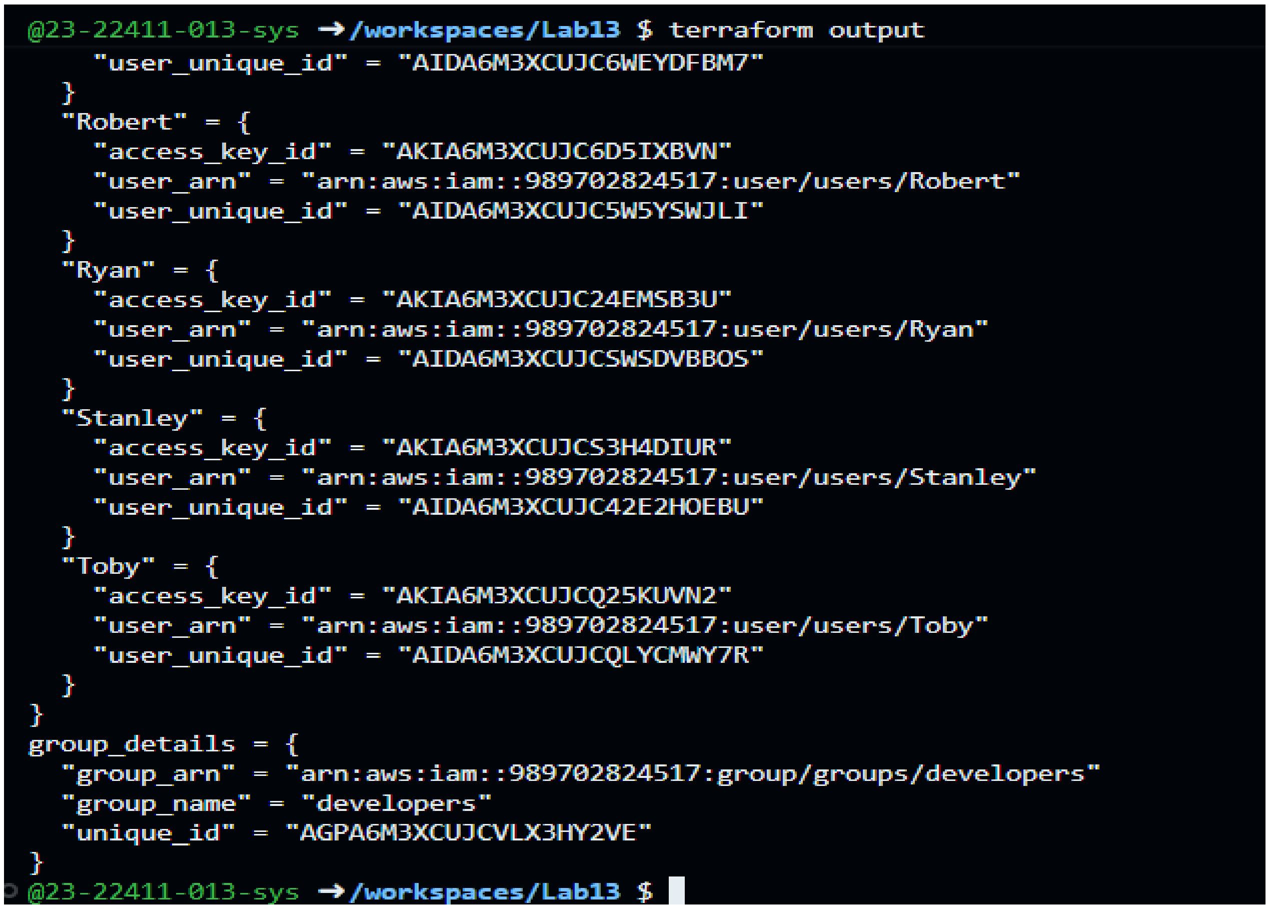
terraform apply -auto-approve -var="iam_password=MySecurePass123!"



```
935468456379]
null_resource.create_login_profiles["Oscar"]: Creation complete after 10s [id=7051
147028657030203]

Apply complete! Resources: 54 added, 0 changed, 0 destroyed.

Outputs:

all_access_key_secrets = <sensitive>
all_users_details = {
  "Andy" = {
    "access_key_id" = "AKIA6M3XCUJCRVIQHS5X"
    "user_arn" = "arn:aws:iam::989702824517:user/users/Andy"
    "user_unique_id" = "AIDA6M3XCUJCUZBP6XVEX"
  }
  "Angela" = {
    "access_key_id" = "AKIA6M3XCUJCQBTLGAMA"
    "user_arn" = "arn:aws:iam::989702824517:user/users/Angela"
    "user_unique_id" = "AIDA6M3XCUJC4Q7FTIJOL"
  }
  "Charles" = {
    "access_key_id" = "AKIA6M3XCUJC35S7MK63"
    "user_arn" = "arn:aws:iam::989702824517:user/users/Charles"
    "user_unique_id" = "AIDA6M3XCUJCR7HCBOUKW"
  }
  "Clark" = {
    "access_key_id" = "AKIA6M3XCUJCTCMNOSHF"
    "user_arn" = "arn:aws:iam::989702824517:user/users/Clark"
    "user_unique_id" = "AIDA6M3XCUJCSKRMKE5ZJ"
  }
```

6. Display the outputs:

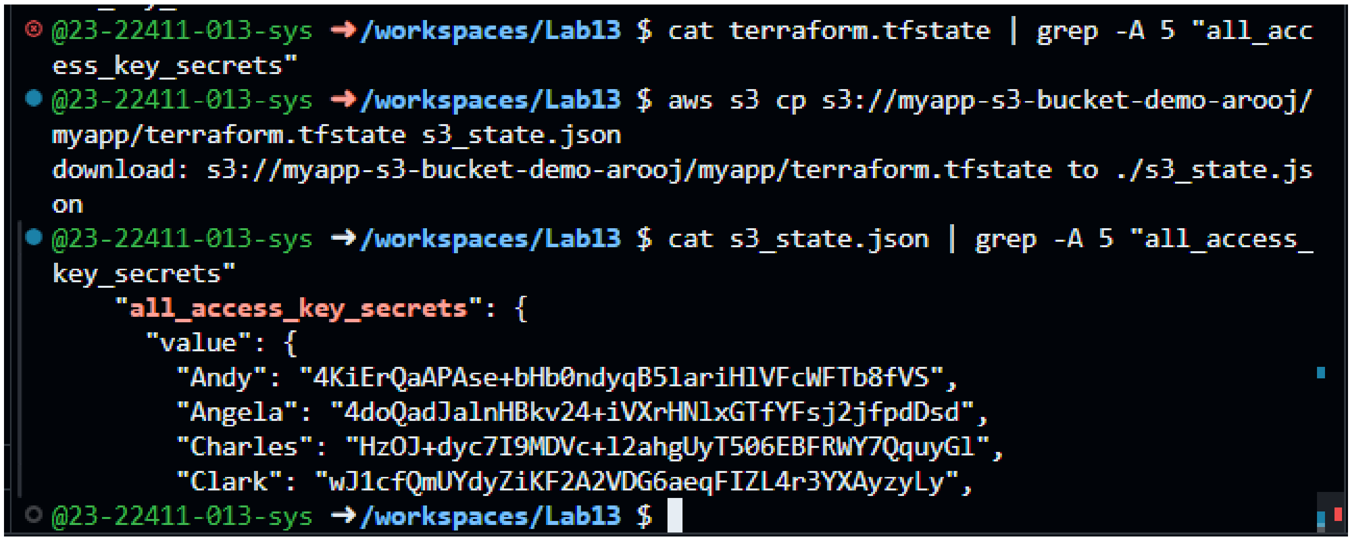
terraform output



```
@23-22411-013-sys ➜/workspaces/Lab13 $ terraform output
    "user_unique_id" = "AIDA6M3XCUJC6WEYDFBM7"
  }
  "Robert" = {
    "access_key_id" = "AKIA6M3XCUJC6D5IXBVN"
    "user_arn" = "arn:aws:iam::989702824517:user/users/Robert"
    "user_unique_id" = "AIDA6M3XCUJC5W5YSWJLI"
  }
  "Ryan" = {
    "access_key_id" = "AKIA6M3XCUJC24EMSB3U"
    "user_arn" = "arn:aws:iam::989702824517:user/users/Ryan"
    "user_unique_id" = "AIDA6M3XCUJCSWSDVBBOS"
  }
  "Stanley" = {
    "access_key_id" = "AKIA6M3XCUJCS3H4DIUR"
    "user_arn" = "arn:aws:iam::989702824517:user/users/Stanley"
    "user_unique_id" = "AIDA6M3XCUJC42E2HOEBU"
  }
  "Toby" = {
    "access_key_id" = "AKIA6M3XCUJCQ25KUVN2"
    "user_arn" = "arn:aws:iam::989702824517:user/users/Toby"
    "user_unique_id" = "AIDA6M3XCUJCQLYCMWY7R"
  }
}
group_details = {
  "group_arn" = "arn:aws:iam::989702824517:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA6M3XCUJCVLX3HY2VE"
}
@23-22411-013-sys ➜/workspaces/Lab13 $
```

7. View secrets in terraform. tfstate:

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

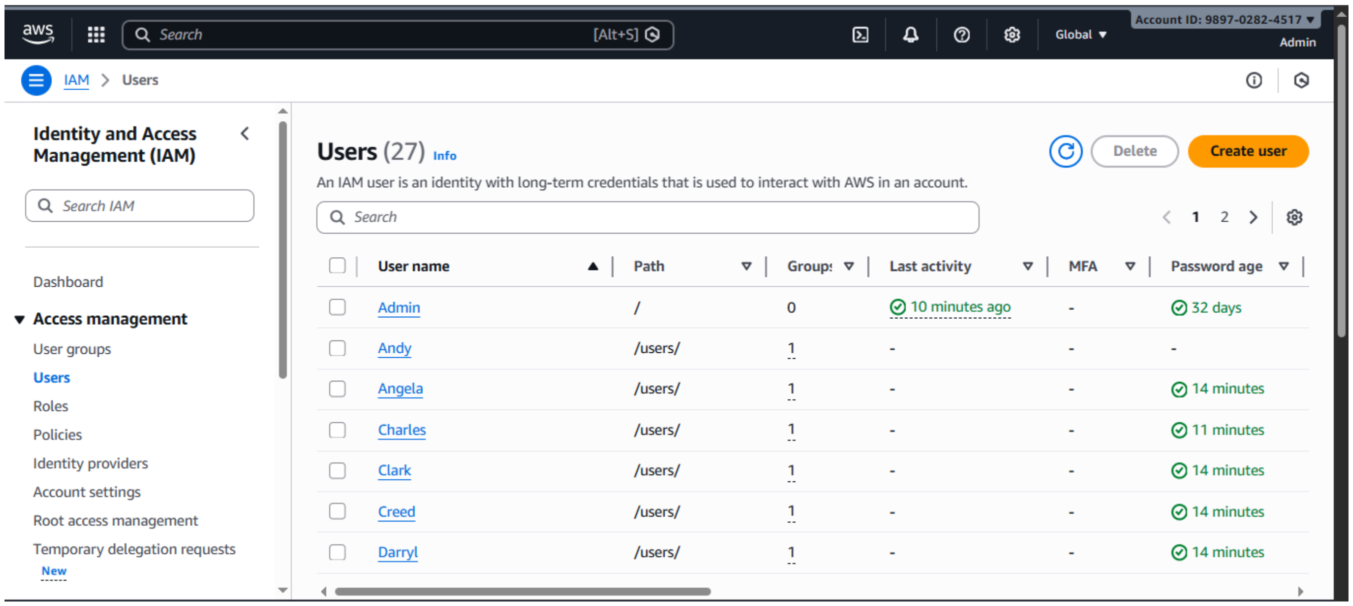


```
@23-22411-013-sys ➜/workspaces/Lab13 $ cat terraform.tfstate | grep -A 5 "all_acc
ess_key_secrets"
@23-22411-013-sys ➜/workspaces/Lab13 $ aws s3 cp s3://myapp-s3-bucket-demo-arooj/
myapp/terraform.tfstate s3_state.json
download: s3://myapp-s3-bucket-demo-arooj/myapp/terraform.tfstate to ./s3_state.js
on
@23-22411-013-sys ➜/workspaces/Lab13 $ cat s3_state.json | grep -A 5 "all_access_
key_secrets"
    "all_access_key_secrets": {
      "value": {
        "Andy": "4KiErQaAPAse+bHb0ndyqB5lariHlVFcWFTb8fVS",
        "Angela": "4doQadJalnHBkv24+iVXrHNlxGTfYFsj2jfpdDsd",
        "Charles": "HzOJ+dyc7I9MDVc+l2ahgUyT506EBFRWY7QquyGl",
        "Clark": "wJ1cfQmUYdyZiKF2A2VDG6aeqFIZL4r3YXAyzyLy",
@23-22411-013-sys ➜/workspaces/Lab13 $
```

The local terraform.tfstate file did not display the all_access_key_secrets output because Terraform was configured to use a **remote S3 backend**. Therefore, the state file was downloaded directly from the S3 bucket using the AWS CLI and saved locally as s3_state.json. This allowed successful verification of the all_access_key_secrets output stored in the remote Terraform state.

8. Verify all users in AWS Console:

Navigate to IAM → Users



Users (27) Info

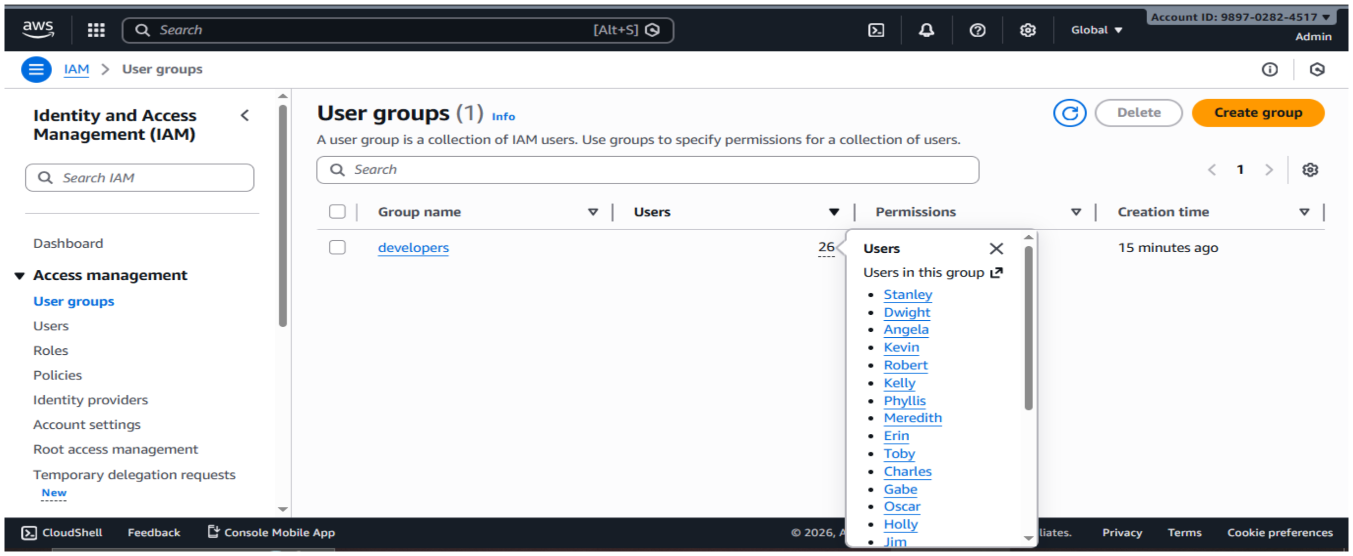
An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

User name	Path	Groups	Last activity	MFA	Password age
Admin	/	0	10 minutes ago	-	32 days
Andy	/users/	1	-	-	-
Angela	/users/	1	-	-	14 minutes
Charles	/users/	1	-	-	11 minutes
Clark	/users/	1	-	-	14 minutes
Creed	/users/	1	-	-	14 minutes
Darryl	/users/	1	-	-	14 minutes

- **Save screenshot as:** task7_aws_console_all_users.png — AWS Console showing all created users.

9. Verify group membership:

Navigate to IAM → Groups → developers → Users tab



User groups (1) Info

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	26		15 minutes ago

Users

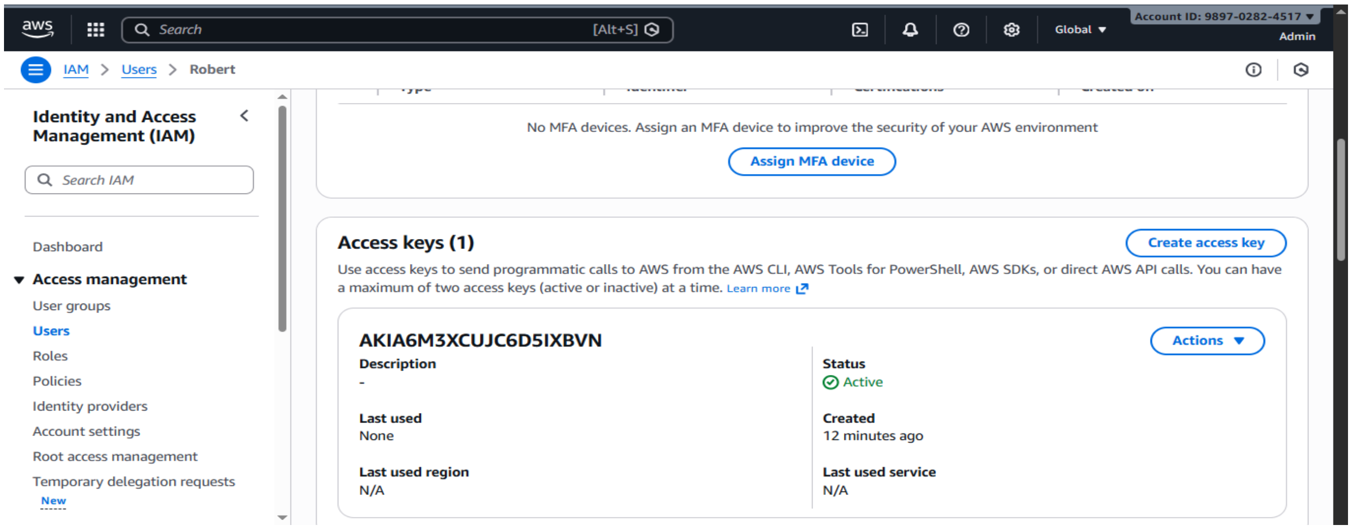
Users in this group

- Stanley
- Dwight
- Angela
- Kevin
- Robert
- Kelly
- Phyllis
- Meredith
- Erin
- Toby
- Charles
- Gabe
- Oscar
- Holly
- Jim

10. Verify one user's access keys:

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

Go to Security credentials tab



No MFA devices. Assign an MFA device to improve the security of your AWS environment

Assign MFA device

Access keys (1)

Create access key

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

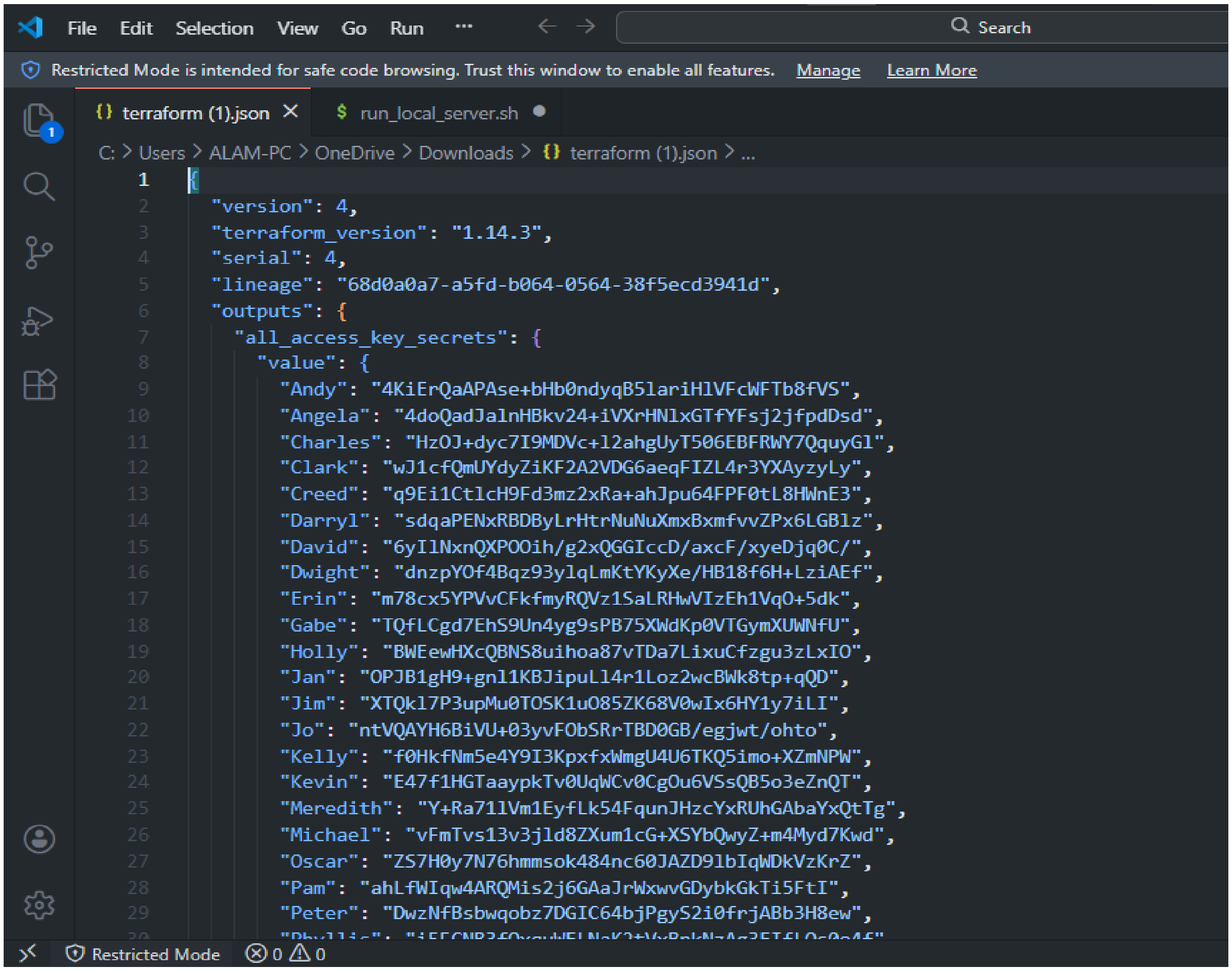
AKIA6M3XCUJC6D5IXBVN

Actions ▾

Description	**Status**
-	✓ Active
Last used	**Created**
None	12 minutes ago
Last used region	**Last used service**
N/A	N/A

11. Check terraform state in S3:

Navigate to S3 bucket and view the state file

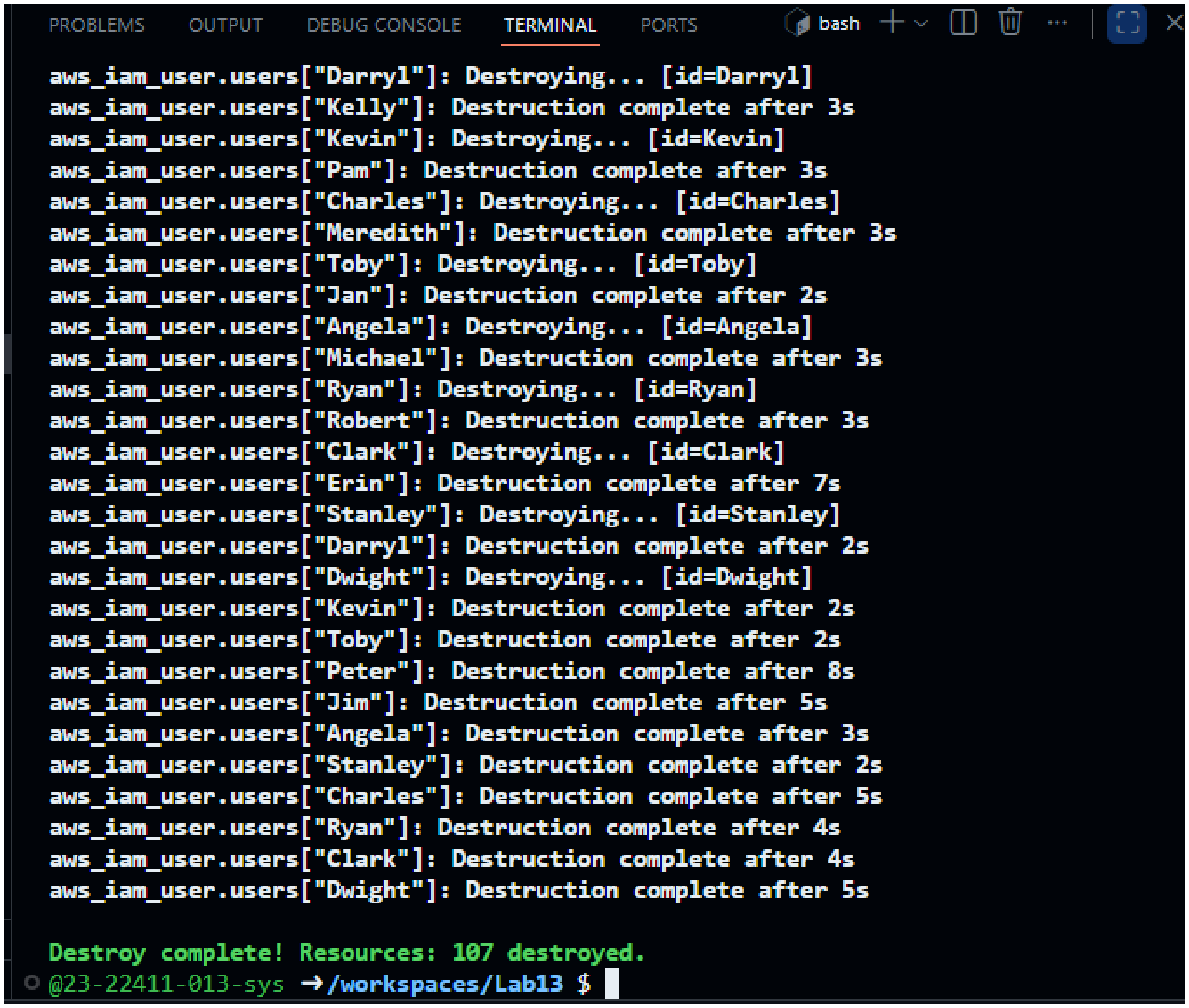

File Edit Selection View Go Run ···
Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
terraform (1).json
run_local_server.sh
C: > Users > ALAM-PC > OneDrive > Downloads > terraform (1).json > ...
Restricted Mode
0 0

```
{
  "version": 4,
  "terraform_version": "1.14.3",
  "serial": 4,
  "lineage": "68d0a0a7-a5fd-b064-0564-38f5ecd3941d",
  "outputs": {
    "all_access_key_secrets": {
      "value": {
        "Andy": "4KiErQaAPAse+bHb0ndyqB5lariHlVFcWFTb8fVS",
        "Angela": "4doQadJalnHBkv24+iVXrHNlxGTfYFsj2jfpdDsd",
        "Charles": "HzOJ+dyc7I9MDVc+l2ahgUyT5O6EBFRWY7QquyGl",
        "Clark": "wJ1cfQmUYdyZiKF2A2VDG6aeqFIZL4r3YXAyzyLy",
        "Creed": "q9Ei1CtlcH9Fd3mz2xRa+ahJpu64FPF0tL8HWnE3",
        "Darryl": "sdqaPENxRBDByLrHtrNuNuXmxBxmfvvZPx6LGBlz",
        "David": "6yIlNxnQXPOOih/g2xQGGIccD/axcF/xyeDjq0C/",
        "Dwight": "dnzpYOf4Bqz93ylqLmKtYKyXe/HB18f6H+LziAEf",
        "Erin": "m78cx5YPVvCFkfmyRQVz1SaLRHwVIzEh1VqO+5dk",
        "Gabe": "TQfLCgd7EhS9Un4yg9sPB75XWdKp0VTGymXUWNfU",
        "Holly": "BWEewHXcQBNS8uihoa87vTDa7LixuCfzgu3zLxIO",
        "Jan": "OPJB1gH9+gnl1KBJipuLl4r1Loz2wcBWk8tp+qQD",
        "Jim": "XTQkl7P3upMu0TOSK1uO85ZK68V0wIx6HY1y7iLI",
        "Jo": "ntVQAYH6BiVU+03yvFObSRrTBD0GB/egjwt/ohto",
        "Kelly": "f0HkfNm5e4Y9I3KpxfxWmgU4U6TKQ5imo+XZmNPW",
        "Kevin": "E47f1HGTaaypkTv0UqWCv0CgOu6VSsQB5o3eZnQT",
        "Meredith": "Y+Ra71lVm1EyfLk54FqunJHzcYxRUhGAbaYxQtTg",
        "Michael": "vFmTvs13v3jld8ZXum1cG+XSYbQwyZ+m4Myd7Kwd",
        "Oscar": "ZS7H0y7N76hmmsok484nc60JAZD9lbIqWDkVzKrZ",
        "Pam": "ahLfWIqw4ARQMis2j6GAaJrWxwvGDybkGkTi5FtI",
        "Peter": "DwzNfBsbwqobz7DGIC64bjPgyS2i0frjABb3H8ew",
```

Cleanup

1. Destroy all resources:

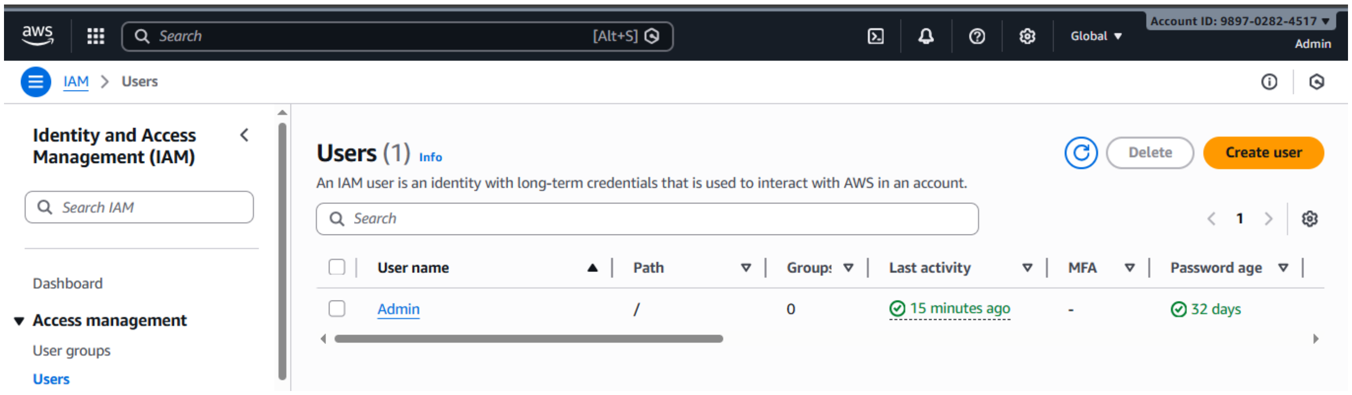
terraform destroy -auto-approve



```
aws_iam_user.users["Darryl"]: Destroying... [id=Darryl]
aws_iam_user.users["Kelly"]: Destruction complete after 3s
aws_iam_user.users["Kevin"]: Destroying... [id=Kevin]
aws_iam_user.users["Pam"]: Destruction complete after 3s
aws_iam_user.users["Charles"]: Destroying... [id=Charles]
aws_iam_user.users["Meredith"]: Destruction complete after 3s
aws_iam_user.users["Toby"]: Destroying... [id=Toby]
aws_iam_user.users["Jan"]: Destruction complete after 2s
aws_iam_user.users["Angela"]: Destroying... [id=Angela]
aws_iam_user.users["Michael"]: Destruction complete after 3s
aws_iam_user.users["Ryan"]: Destroying... [id=Ryan]
aws_iam_user.users["Robert"]: Destruction complete after 3s
aws_iam_user.users["Clark"]: Destroying... [id=Clark]
aws_iam_user.users["Erin"]: Destruction complete after 7s
aws_iam_user.users["Stanley"]: Destroying... [id=Stanley]
aws_iam_user.users["Darryl"]: Destruction complete after 2s
aws_iam_user.users["Dwight"]: Destroying... [id=Dwight]
aws_iam_user.users["Kevin"]: Destruction complete after 2s
aws_iam_user.users["Toby"]: Destruction complete after 2s
aws_iam_user.users["Peter"]: Destruction complete after 8s
aws_iam_user.users["Jim"]: Destruction complete after 5s
aws_iam_user.users["Angela"]: Destruction complete after 3s
aws_iam_user.users["Stanley"]: Destruction complete after 2s
aws_iam_user.users["Charles"]: Destruction complete after 5s
aws_iam_user.users["Ryan"]: Destruction complete after 4s
aws_iam_user.users["Clark"]: Destruction complete after 4s
aws_iam_user.users["Dwight"]: Destruction complete after 5s

Destroy complete! Resources: 107 destroyed.
@23-22411-013-sys →/workspaces/Lab13 $
```

2. Verify users deleted in AWS Console:

Navigate to IAM → Users



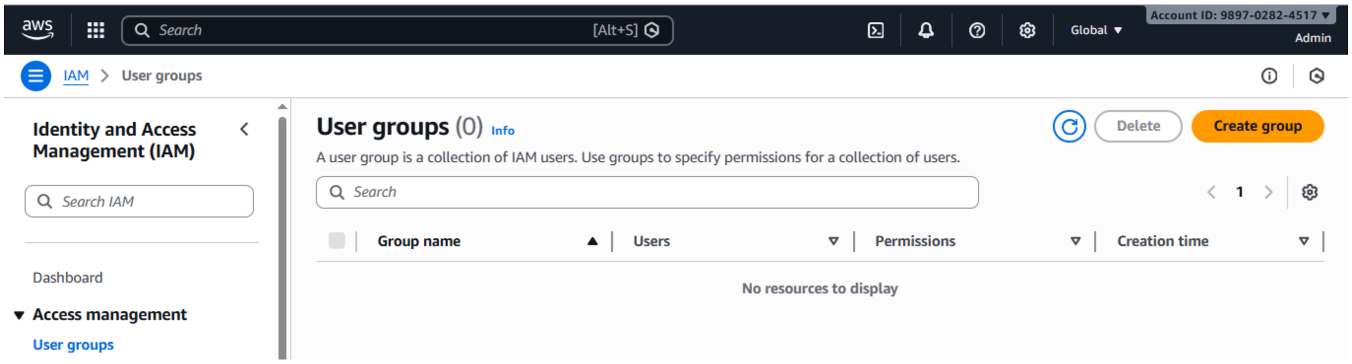
Users (1) Info

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	15 minutes ago	-	32 days

3. Verify group deleted in AWS Console:

Navigate to IAM → Groups



User groups (0) Info

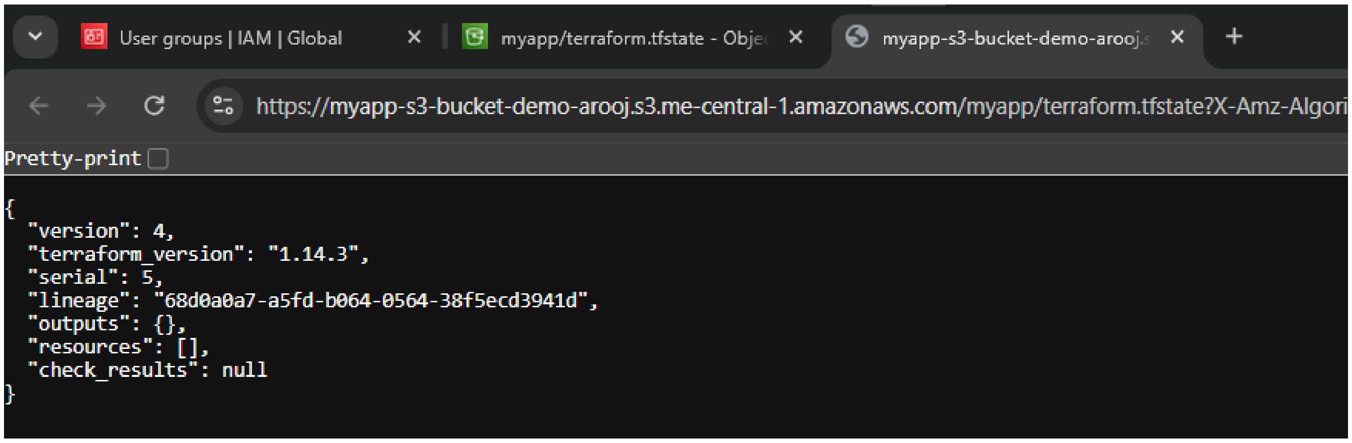
A user group is a collection of IAM users. Use groups to specify permissions for a collection of users.

Group name	Users	Permissions	Creation time

No resources to display

4. Check S3 state file:

Navigate to S3 bucket

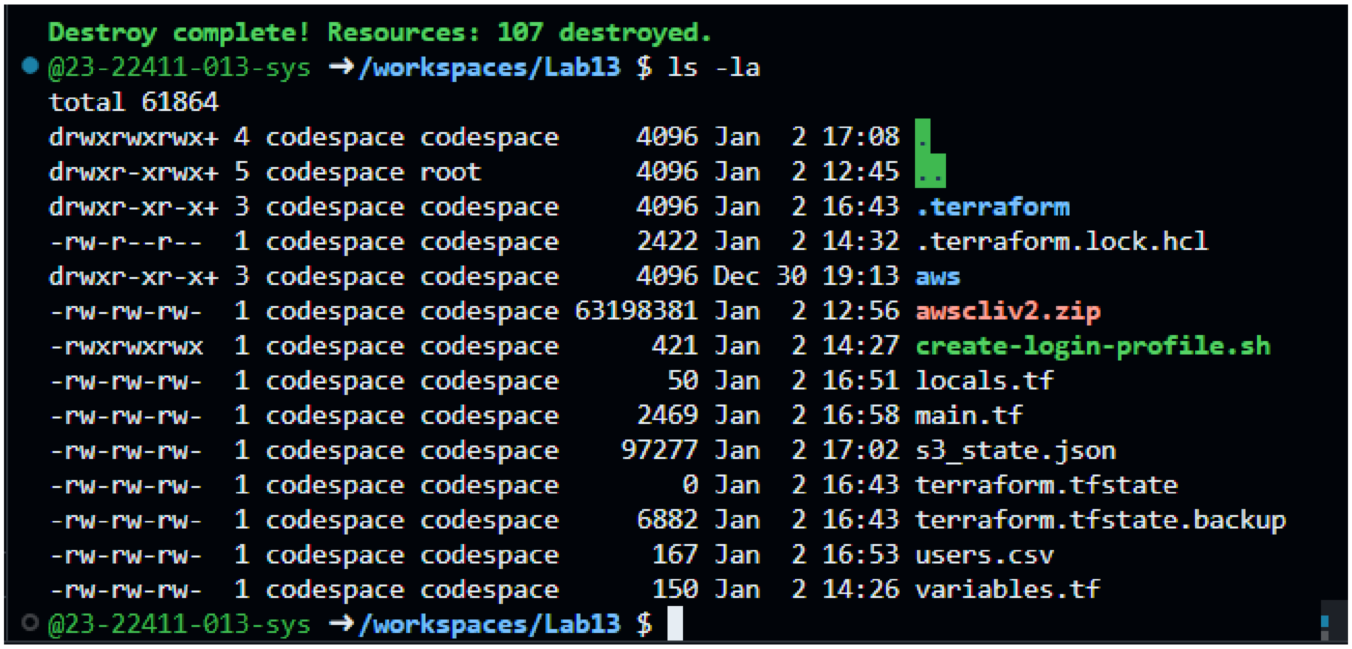

User groups | IAM | Global
myapp/terraform.tfstate - Obje
myapp-s3-bucket-demo-arooj.s
Pretty-print

https://myapp-s3-bucket-demo-arooj.s3.me-central-1.amazonaws.com/myapp/terraform.tfstate?X-Amz-Algori

```
{
  "version": 4,
  "terraform_version": "1.14.3",
  "serial": 5,
  "lineage": "68d0a0a7-a5fd-b064-0564-38f5ecd3941d",
  "outputs": {},
  "resources": [],
  "check_results": null
}
```

5. List all project files:

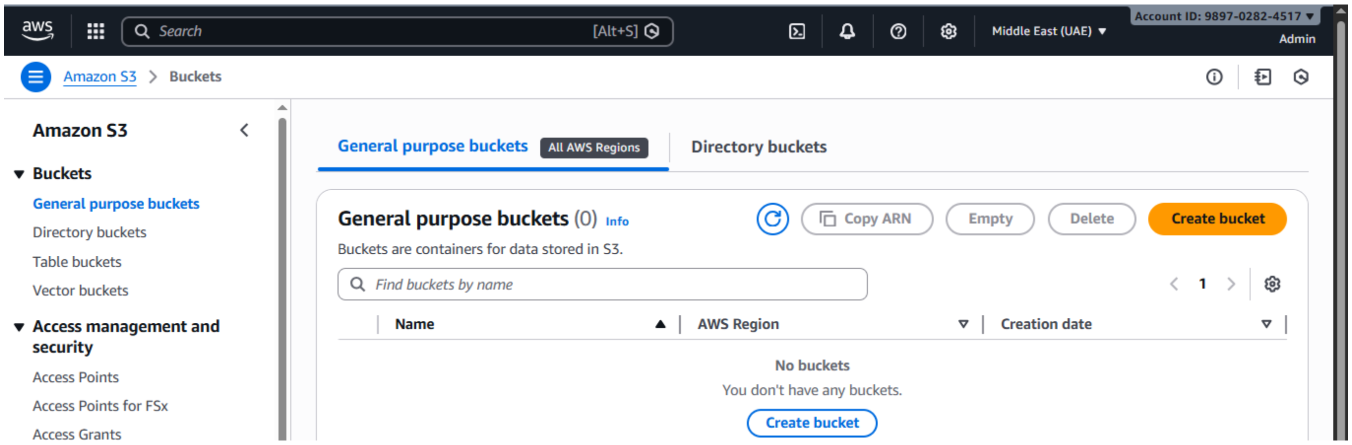
ls -la



```
Destroy complete! Resources: 107 destroyed.
@23-22411-013-sys ➜ /workspaces/Lab13 $ ls -la
total 61864
drwxrwxrwx+ 4 codespace codespace     4096 Jan  2 17:08 .
drwxr-xrwx+ 5 codespace root          4096 Jan  2 12:45 ..
drwxr-xr-x+ 3 codespace codespace     4096 Jan  2 16:43 .terraform
-rw-r--r--  1 codespace codespace     2422 Jan  2 14:32 .terraform.lock.hcl
drwxr-xr-x+ 3 codespace codespace     4096 Dec 30 19:13 aws
-rw-rw-rw-  1 codespace codespace 63198381 Jan  2 12:56 awscliv2.zip
-rwxrwxrwx  1 codespace codespace      421 Jan  2 14:27 create-login-profile.sh
-rw-rw-rw-  1 codespace codespace       50 Jan  2 16:51 locals.tf
-rw-rw-rw-  1 codespace codespace     2469 Jan  2 16:58 main.tf
-rw-rw-rw-  1 codespace codespace    97277 Jan  2 17:02 s3_state.json
-rw-rw-rw-  1 codespace codespace        0 Jan  2 16:43 terraform.tfstate
-rw-rw-rw-  1 codespace codespace     6882 Jan  2 16:43 terraform.tfstate.backup
-rw-rw-rw-  1 codespace codespace      167 Jan  2 16:53 users.csv
-rw-rw-rw-  1 codespace codespace      150 Jan  2 14:26 variables.tf
@23-22411-013-sys ➜ /workspaces/Lab13 $
```

6. 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.

Create bucket