

**Name: Zuha Irfan**

**Reg no: 2023-BSE-073**

**Course: Cloud Computing Lab**

**Section: V-B**

## LAB 13

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

All actions below should be executed inside the Codespace shell.

Create Codespace & connect:

```
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073 (main) $ gh codespace list
NAME          DISPLAY NAME      REPOSITORY           BRANCH STATE    CREATED AT
super-space-computing-machine-wrq76pj0qj535xj super space computing-machine Zuha-Irfan/CC-ZuhaIrfan-073 main Available about 17 minutes ago
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073 (main) $ |
```

```
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073 (main) $ gh codespace ssh
? Choose codespace: Zuha-Irfan/CC-ZuhaIrfan-073 [main]: super space computing-machine
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
Last login: Wed Jan 14 09:29:45 2026 from ::1
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073 (main) $ |
```

### Task 1 — Create IAM Group and Output Details

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

Create the initial project structure:

```
mkdir -p ~/Lab13
```

```
cd ~/Lab13
```

Create the main Terraform file:

```
touch main.tf
```

```
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ touch main.tf
```

Create main.tf with AWS provider configuration:

```

@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ vim main.tf
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ cat main.tf
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
    }
}

```

Initialize Terraform:

[terraform init](#)

```

@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.27.0...
- Installed hashicorp/aws v6.27.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record
the provider
selections it made above. Include this file in your version control
repository
so that Terraform can guarantee to make the same selections by default
when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform
plan" to see
any changes that are required for your infrastructure. All Terra
form 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.
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ |

```

Apply the configuration:

[terraform apply -auto-approve](#)

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

#### Outputs:

```
group_details = {  
  "group_arn" = "arn:aws:iam::538079272540:group/groups/developers"  
  "group_name" = "developers"  
  "unique_id" = "AGPAX2SAGDZODXXWVG57E"  
}
```

```
@Zuha-Irfan ~ /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $
```

Display the output:

#### terraform output

```
@Zuha-Irfan ~ /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ terraform output  
group_details = {  
  "group_arn" = "arn:aws:iam::538079272540:group/groups/developers"  
  "group_name" = "developers"  
  "unique_id" = "AGPAX2SAGDZODXXWVG57E"  
}
```

Verify the group in AWS Console:

#### [Navigate to IAM → Groups in AWS Console](#)

The screenshot shows the AWS IAM Groups page. On the left, there's a navigation sidebar with 'Identity and Access Management (IAM)' selected. Under 'Access management', 'User groups' is also selected. The main pane displays a table titled 'User groups (1)'. The table has one row for a group named 'developers'. The 'Group name' column shows 'developers', the 'Users' column shows '0', the 'Permissions' column shows 'Not defined', and the 'Creation time' column shows '11 minutes ago'. There are buttons for 'Delete' and 'Create group' at the top right of the table area.

## 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.

Update main.tf to add the IAM user resource:

```

@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ vim main.tf
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ cat main.tf
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
  }
}

```

Apply the configuration:

`terraform apply -auto-approve`

```

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

Outputs:

group_details = {
  "group_arn" = "arn:aws:iam::538079272540:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPAX2SAGDZODXXWVG57E"
}
user_details = {
  "unique_id" = "AIDAX2SAGDZOH3YAHYCMA"
  "user_arn" = "arn:aws:iam::538079272540:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ |

```

Display the outputs:

`terraform output`

```

@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ terraform output
group_details = {
  "group_arn" = "arn:aws:iam::538079272540:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPAX2SAGDZODXXWVG57E"
}
user_details = {
  "unique_id" = "AIDAX2SAGDZOH3YAHYCMA"
  "user_arn" = "arn:aws:iam::538079272540:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}

```

Verify the user in AWS Console:

**Navigate to IAM → Users in AWS Console**

**Click on "loadbalancer" user**

**Check the "Groups" tab**

The screenshot shows the AWS IAM User details page for a user named 'loadbalancer'. The 'Groups' tab is selected, displaying a single entry: 'developers'. Other tabs visible include 'Permissions', 'Tags', 'Security credentials', and 'Last Accessed'. The 'Summary' section provides basic information like ARN, creation date, and console access status.

The screenshot shows the AWS IAM User Groups details page for a group named 'developers'. The 'Users' tab is selected, showing one user assigned to the group: 'loadbalancer'. Other tabs visible include 'Permissions' and 'Access Advisor'. The 'Summary' section provides basic information like ARN and creation date.

### Task 3 — Attach Policies to IAM Group

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

Update main.tf to add policy attachments:

```

@Zuha-Irfan → /workspaces/CC-Zuhairfan-073/Lab13 (main) $ vim main.tf
@Zuha-Irfan → /workspaces/CC-Zuhairfan-073/Lab13 (main) $ cat main.tf
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"
}

```

Apply the configuration:

**terraform apply -auto-approve**

```

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

Outputs:

group_details = {
  "group_arn" = "arn:aws:iam::538079272540:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPAX2SAGDZODXXWVG57E"
}
user_details = {
  "unique_id" = "AIDAX2SAGDZOH3YAHYCMA"
  "user_arn" = "arn:aws:iam::538079272540:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@Zuha-Irfan → /workspaces/CC-Zuhairfan-073/Lab13 (main) $ |

```

Verify policies in AWS Console:

**Navigate to IAM → Groups → developers**

**Click on "Permissions" tab**

The screenshot shows the AWS IAM console with the 'User groups' section selected. A user group named 'developers' is viewed. The 'Permissions' tab is active, showing two policies attached to the group: 'AmazonEC2FullAccess' and 'IAMUserChangePassword', both of which are AWS managed policies.

## 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.

Create variables.tf file:

```
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ vim variable.tf
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ cat variable.tf
variable "iam_password" {
  description = "Temporary password for the IAM user"
  type        = string
  sensitive   = true
  default     = "IdontKnow"
}
```

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

```
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ vim create-login-profile.sh
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ cat 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
```

Make the script executable:

**chmod +x create-login-profile.sh**

```
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ chmod +x create-login-profile.sh
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ ls -l create-login-profile.sh
-rwxrwxrwx 1 codespace codespace 423 Jan  1 16:37 create-login-profile.sh
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ |
```

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

```
Zuha-Irfan ~ /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ vim main.tf
Zuha-Irfan ~ /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ cat main.tf
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 "null_resource" "create_login_profile" {
  triggers = {
    password_hash = sha256(var.iam_password)
    user          = aws_iam_user.lb.name
  }

  depends_on = [aws_iam_user.lb]

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

resource "aws_iam_group_policy_attachment" "developer_ec2_fullaccess" {
```

Apply the configuration with a custom password:

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

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

Outputs:

```
group_details = {
  "group_arn" = "arn:aws:iam::538079272540:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPAX2SAGDZODXXWVG57E"
}
user_details = {
  "unique_id" = "AIDAX2SAGDZOH3YAHYCMA"
  "user_arn" = "arn:aws:iam::538079272540:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
```

Verify login profile creation:

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

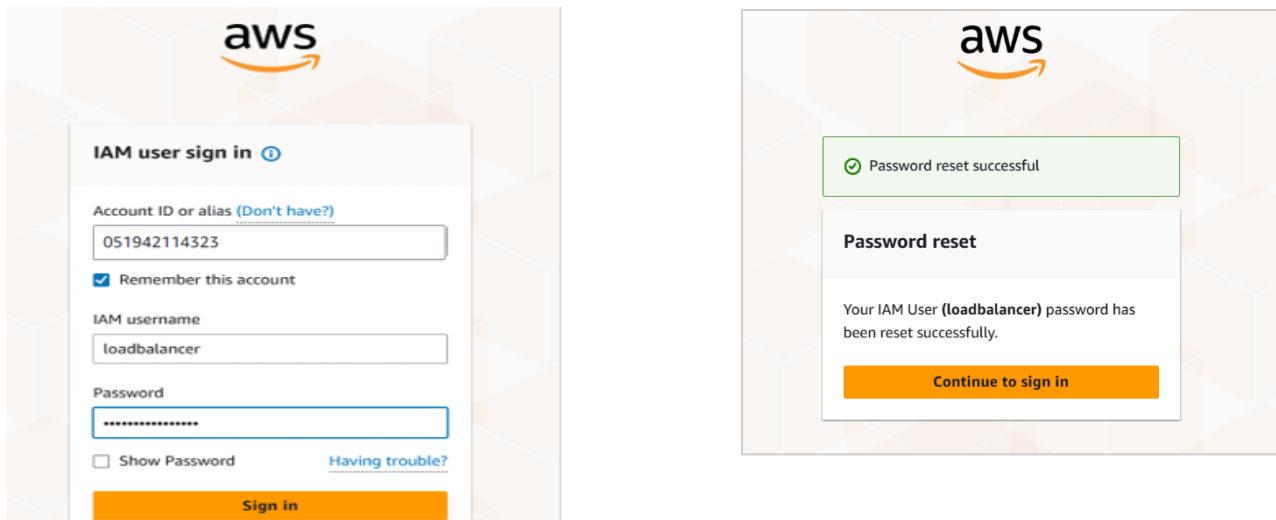
```
Zuhair-Irfan → /workspaces/CC-ZuhairFan-873/Lab13 (main) $ aws iam get-login-profile --user-name loadbalancer
{
    "LoginProfile": {
        "UserName": "loadbalancer",
        "CreateDate": "2026-01-01T16:44:27+00:00",
        "PasswordResetRequired": true
    }
}
```

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



## 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.

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

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

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

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

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

Apply the configuration:

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

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

Outputs:

access_key_id = "AKIA2SAGDZOBASUHMU"
access_key_secret = <sensitive>
group_details = {
  "group_arn" = "arn:aws:iam::538079272540:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPAX2SAGDZODXXWVG57E"
}
user_details = {
  "unique_id" = "AIDAX2SAGDZOH3YAHYCMA"
  "user_arn" = "arn:aws:iam::538079272540:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
```

Display outputs:

**terraform output**

```
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ terraform output
access_key_id = "AKIA2SAGDZOBASUHMU"
access_key_secret = <sensitive>
group_details = {
  "group_arn" = "arn:aws:iam::538079272540:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPAX2SAGDZODXXWVG57E"
}
user_details = {
  "unique_id" = "AIDAX2SAGDZOH3YAHYCMA"
  "user_arn" = "arn:aws:iam::538079272540:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
```

View the secret in terraform state:

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

```
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ cat terraform.tfstate | grep -A 10 "access_key_secret"
"access_key_secret": {
  "value": "757IpPCA4j405x9H/LFUJMvvr2lKTfKJQlhUGLb/",
  "type": "string",
  "sensitive": true
},
"group_details": {
  "value": {
    "group_arn": "arn:aws:iam::538079272540:group/groups/developers",
    "group_name": "developers",
    "unique_id": "AGPAX2SAGDZODXXWVG57E"
  }
},
```

Verify access key in AWS Console:

**Navigate to IAM → Users → loadbalancer → Security credentials**

The screenshot shows the AWS IAM User Details page for a user named 'loadbalancer'. The 'Summary' section includes the ARN (arn:aws:iam::538079272540:user/users/loadbalancer), creation date (January 01, 2026, 21:27 (UTC-05:00)), and access key details. The 'Permissions' tab is selected, showing two attached policies: 'AWSLambdaBasicExecutionRole' and 'AmazonCloudWatchLogsFullAccess'. There are buttons for 'Add permissions' and 'Remove'.

## Task 6 — Implement Terraform Remote State with S3

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

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"](#)

A green success message box displays: 'Successfully created bucket "myapp-s3-bucket-nimra". To upload files and folders, or to configure additional bucket settings, choose View details.' A 'View details' button and a close 'X' button are visible.

Update main.tf to add S3 backend configuration:

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

```
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ vim main.tf
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ cat main.tf
provider "aws" {
  shared_config_files      = ["~/.aws/config"]
  shared_credentials_files = ["~/.aws/credentials"]
}

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

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

Reinitialize Terraform with the backend:

**terraform init -migrate-state**

Type yes when prompted to migrate state

```
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ terraform init -migrate-state
Initializing the backend...
Do you want to copy existing state to the new backend?
  Pre-existing state was found while migrating the previous "local" backend to the
  newly configured "s3" backend. No existing state was found in the newly
  configured "s3" backend. Do you want to copy this state to the new "s3"
  backend? Enter "yes" to copy and "no" to start with an empty state.

Enter a value: yes

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!
```

Apply the configuration:

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

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

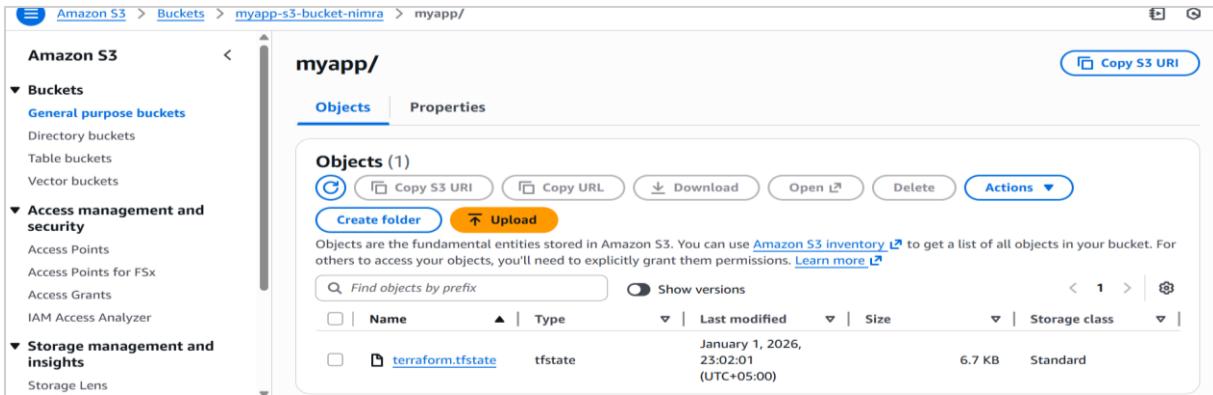
Outputs:

access_key_id = "AKIA2SAGDZOBAZSUHMU"
access_key_secret = <sensitive>
group_details = {
  "group_arn" = "arn:aws:iam::538079272540:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPAX2SAGDZODXXWVG57E"
}
user_details = {
  "unique_id" = "AIDAX2SAGDZOH3YAHYCMA"
  "user_arn" = "arn:aws:iam::538079272540:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
```

Verify state file in S3:

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

You should see terraform.tfstate file



The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with 'Amazon S3' navigation, including 'Buckets', 'Access management and security', and 'Storage management and insights'. The main area is titled 'myapp/' and shows 'Objects (1)'. There's a 'Create folder' button and an 'Upload' button. Below that, a note says: '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](#)'. A table lists the single object: 'terraform.tfstate'.

Name	Type	Last modified	Size	Storage class
terraform.tfstate	tfstate	January 1, 2026, 23:02:01 (UTC+05:00)	6.7 KB	Standard

Check local state file:

ls -la terraform.tfstate\*

```
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ ls -la terraform.tfstate*
-rw-rw-rw- 1 codespace codespace 0 Jan 1 18:02 terraform.tfstate
-rw-rw-rw- 1 codespace codespace 6882 Jan 1 18:02 terraform.tfstate.backup
```

Destroy resources and verify state change:

terraform destroy -auto-approve

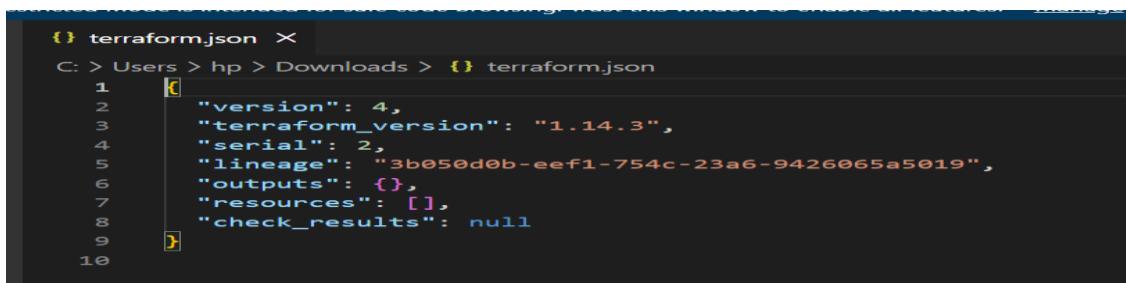
```
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Destroying... [id=developers-202601011633340680000001]
aws_iam_user_group_membership.lb_membership: Destroying... [id=terraform-20260101162718388700000001]
aws_iam_access_key.lb_access_key: Destroying... [id=AKIAZ2AGDZOBASUHNU]
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Destruction complete after 1s
aws_iam_user_group_membership.lb_membership: Destruction complete after 1s
aws_iam_access_key.lb.access_key: Destruction complete after 1s
aws_iam_user.lb: Destroying... [id=loadbalancer]
aws_iam_group_policy_attachment.change_password: Destruction complete after 1s
aws_iam_group.developers: Destruction complete after 0s
aws_iam_group.developers: Destruction complete after 0s
aws_iam_user.lb: Destruction complete after 2s

Destroy complete! Resources: 7 destroyed.
```

Verify updated state in S3:

Refresh S3 bucket view

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



```
❶ terraform.json ✘
C: > Users > hp > Downloads > ❷ terraform.json
1   {
2     "version": 4,
3     "terraform_version": "1.14.3",
4     "serial": 2,
5     "lineage": "3b050d0b-eef1-754c-23a6-9426065a5019",
6     "outputs": {},
7     "resources": [],
8     "check_results": null
9   }
10
```

## Task 7 — Create Multiple Users from CSV File

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

Create locals.tf file:

```
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ vim locals.tf
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ cat locals.tf
locals {
    users = csvdecode(file("users.csv"))
}
```

Create users.csv file:

```
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ vim users.csv
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-073/Lab13 (main) $ cat 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
```

Update main.tf to create multiple users:

```
# Create login profiles for all users
resource "null_resource" "create_login_profiles" {
  triggers = [
    "aws_iam_user.users"
  ]
  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
}
```

Replace the single user resources with:

Reinitialize Terraform (since we changed the configuration significantly):

**terraform init**

```
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-873/Lab13 (main) $ 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.
```

Apply the configuration to create all users:

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

```
azurerm_resource_group: creation complete after 11s [id: rg-lab13]
Apply complete! Resources: 107 added, 0 changed, 0 destroyed.

Outputs:

all_access_key_secrets = <sensitive>
all_users_details = {
  "Andy" = {
    "access_key_id" = "AKIAJX2SAGDZOEWI5VT62"
    "user_arn" = "arn:aws:iam::538079272540:user/users/Andy"
    "user_unique_id" = "AIDAX2SAGDZOPDE3FPKE2"
  }
  "Angela" = {
    "access_key_id" = "AKIAJX2SAGDZOIVAU6GV5"
    "user_arn" = "arn:aws:iam::538079272540:user/users/Angela"
    "user_unique_id" = "AIDAX2SAGDZOJPGRAK4MV"
  }
  "Charles" = {
    "access_key_id" = "AKIAJX2SAGDZOEF6W5RN6"
    "user_arn" = "arn:aws:iam::538079272540:user/users/Charles"
    "user_unique_id" = "AIDAX2SAGDZO0YCFQXMCK"
  }
  "Clark" = {
    "access_key_id" = "AKIAJX2SAGDZOLSABSMHS"
    "user_arn" = "arn:aws:iam::538079272540:user/users/Clark"
    "user_unique_id" = "AIDAX2SAGDZOKERCTXF3J"
  }
  "Creed" = {
    "access_key_id" = "AKIAJX2SAGDZOHO45OJ3XG"
    "user_arn" = "arn:aws:iam::538079272540:user/users/Creed"
  }
}
```

Display the outputs:

**terraform output**

```
@Zuha-Irfan → /workspaces/CC-ZuhaIrfan-873/Lab13 (main) $ terraform output
all_access_key_secrets = <sensitive>
all_users_details = {
  "Andy" = {
    "access_key_id" = "AKIAJX2SAGDZOEWI5VT62"
    "user_arn" = "arn:aws:iam::538079272540:user/users/Andy"
    "user_unique_id" = "AIDAX2SAGDZOPDE3FPKE2"
  }
  "Angela" = {
    "access_key_id" = "AKIAJX2SAGDZOIVAU6GV5"
    "user_arn" = "arn:aws:iam::538079272540:user/users/Angela"
    "user_unique_id" = "AIDAX2SAGDZOJPGRAH4MV"
  }
  "Charles" = {
    "access_key_id" = "AKIAJX2SAGDZOEF6W5RN6"
    "user_arn" = "arn:aws:iam::538079272540:user/users/Charles"
    "user_unique_id" = "AIDAX2SAGDZO0YCFQXMCK"
  }
  "Clark" = {
    "access_key_id" = "AKIAJX2SAGDZOLSABSMHS"
    "user_arn" = "arn:aws:iam::538079272540:user/users/Clark"
    "user_unique_id" = "AIDAX2SAGDZOKERCTXF3J"
  }
  "Creed" = {
    "access_key_id" = "AKIAJX2SAGDZOHO45OJ3XG"
    "user_arn" = "arn:aws:iam::538079272540:user/users/Creed"
    "user_unique_id" = "AIDAX2SAGDZOFEMSNBUO"
  }
  "Darryl" = {
    "access_key_id" = "AKIAJX2SAGDZOHMRADRV5"
    "user_arn" = "arn:aws:iam::538079272540:user/users/Darryl"
    "user_unique_id" = "AIDAX2SAGDZOH3PM5D564"
  }
}
```

View secrets in terraform. tfstate:

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

```
@Zuhai-Irfan ~ /workspaces/CC-ZuhaiIrfan-073/Lab13 (main) $ cat terraform.tfstate | grep -A 5 "all_access_key_secrets"
```

Verify all users in AWS Console:

Navigate to IAM → Users

User name	Path	Groups	Last activity	MFA	Password age	Console last sign-in	Access key ID	Active key age	Access key last use	ARN	Creation time	Cs
Andy	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Angela	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Charles	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Clark	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Creed	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Darryl	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
David	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Dwight	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Eris	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Gabe	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Holly	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Jan	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Jim	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Jo	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Kelly	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Kevin	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Heredith	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Michael	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Oscar	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒
Pam	/users/	1	-	-	⌚ 7 minutes	-	Active - AIAIA25AG02...	⌚ 7 minutes	-	arnaws:iam::538079272540:user/user...	7 minutes ago	🕒

Verify group membership:

Navigate to IAM → Groups → developers → Users tab

User name	Groups	Last activity	Creation time
Andy	1	None	15 minutes ago
Angela	1	None	15 minutes ago
Charles	1	None	15 minutes ago
Clark	1	None	15 minutes ago
Creed	1	None	15 minutes ago
Darryl	1	None	15 minutes ago
David	1	None	15 minutes ago
Dwight	1	None	15 minutes ago

Verify one user's access keys:

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

Go to Security credentials tab

Check terraform state in S3:

## Navigate to S3 bucket and view the state file

```

C: > Users > hp > Downloads > terraform.tfstate ...
1   {
2     "version": 4,
3     "terraform_version": "1.14.3",
4     "serial": 4,
5     "lineage": "bb050d0b-eef1-754c-23a6-9426065a5019",
6     "outputs": {
7       "all_access_key_secrets": {
8         "value": [
9           "Andy": "Cf7NnylCHICjeFJuYJrmrjlcQNuTc9001UvYfsN3",
10          "Angela": "quHPsvsc9cw9tch6atispypEbmuXfvx1o@glia6",
11          "Charles": "Sxocp-NwMJDZcLD3vz2ncgNhCmcih8q1Wpzi",
12          "Clark": "Qwz1jpBIdoA8g+YUYfdth4V57QwL0vz5gEB13Zc",
13          "Creed": "Synhd77ocNSdg19MhjkVnKrtz4dHkj5aeh9n",
14          "Darryl": "KrgzAQvnnp65Mah0Ogmn1Y/NvxeZvD4ZBT0dtk",
15          "David": "zgtz3nT3FKRx5mT31Ftgt12tdH6Iey56j3177Dg",
16          "Dwight": "PmHkhnUyrrpYBgxx+Tu9p0+r+tkTIL9Vb+w+VUDmm9A",
17          "Erin": "V3YCJ17XAFLsXgkBvOnBo65vxGgf3+o7k1BEscrJ",
18          "Gabe": "18ph42wraVtz2bmV83h518R04/Vzdc+c*xA2k15k3o",
19          "Holly": "1W5+f1c0LukvR8VPyPL8k0UJZ53iZ/Y4SEXQZ4",
20          "Jan": "RsKhW1Ytr2/e91OOljae3RN8bu1U1xpQW8MaYwOKA",
21          "Jim": "TyR1QCGB4+F8kwyNplcsyWNUk10saoZc/RN3LxO",
22          "Jo": "6Qf85KM+AqK0S+185sysoMjZFS5HwBZ41guQjilk",
23          "Kelly": "XFBzosCLWnXfijaz2pobUV7Fkw0FqyQdqzwn6N24",
24          "Kevin": "4BUvldrnkD0vOnr/ApcNDixXb953y8PanH2Vak",
25          "Meredith": "aRMP6uTMWlnuy0Wvrcnen45EqghZMFNCNCHd7H",
26          "Michael": "134HpoL9kqCMriqp+OrogjehlwZsec2yUGUytA",
27          "Oscar": "PM40kmtrpJTX1RtmAfqQqo5fOKKaPCwpqdPTB1oE",
28          "Pam": "NISZVm7R4XG7tjKJXJ4tRrKgc91n7990zB4h3Ur",
29          "Peter": "1jqXKM0GudxtjJxN4aPEwihly+p7/zNQ0g1qBr9pt",
        ]
      }
    }
  }
}

```

## Cleanup

Destroy all resources:

```

aws_iam_user.users["Jim"] : Destroying... [id=Jim]
aws_iam_user.users["Meredith"] : Destruction complete after 3s
aws_iam_user.users["Clark"] : Destroying... [id=Clark]
aws_iam_user.users["David"] : Destruction complete after 2s
aws_iam_user.users["Oscar"] : Destruction complete after 2s
aws_iam_user.users["Charles"] : Destruction complete after 2s
aws_iam_user.users["Jim"] : Destruction complete after 2s
aws_iam_user.users["Michael"] : Destruction complete after 5s
aws_iam_user.users["Peter"] : Destruction complete after 4s
aws_iam_user.users["Jo"] : Destruction complete after 5s
aws_iam_user.users["Stanley"] : Destruction complete after 5s
aws_iam_user.users["Gabe"] : Destruction complete after 7s
aws_iam_user.users["Clark"] : Destruction complete after 7s

Destroy complete! Resources: 107 destroyed.
@7uha-Trfan → /workspaces/CC-7uhaTrfan-073/Lab13 (main) $ 

```

**terraform destroy -auto-approve**

Verify users deleted in AWS Console:

**Navigate to IAM → Users**

The screenshot shows the AWS IAM Users console. On the left, there's a sidebar with 'Identity and Access Management (IAM)' and 'Access management' sections, with 'Users' selected. The main area is titled 'Users (0) info' and contains a search bar and a table header with columns for 'User name', 'Path', 'Group', 'Last activity', 'MFA', and 'Password age'. A message at the bottom says 'No resources to display'.

Verify group deleted in AWS Console:

**Navigate to IAM → Groups**

The screenshot shows the AWS IAM Groups console. The sidebar is identical to the Users console. The main area is titled 'User groups (0) info' and contains a search bar and a table header with columns for 'Group name', 'Users', 'Permissions', and 'Creation time'. A message at the bottom says 'No resources to display'.

Check S3 state file:

Navigate to S3 bucket

The screenshot shows a terminal window with the command 'ls -la' entered. The output shows a file named 'terraform (2).json'. The file is opened in a code editor, displaying its JSON content. The file contains information about a Terraform run, including the version (4), Terraform version (1.14.3), serial number (6), lineage (3b050d0b-eef1-754c-23a6-9426065a5019), outputs ({}), resources ([]), and check results (null).

```
{ } terraform (2).json ×
C: > Users > hp > Downloads > { } terraform (2).json > ...
1  [ ]
2
3
4
5
6
7
8
9
10
```

```
{
  "version": 4,
  "terraform_version": "1.14.3",
  "serial": 6,
  "lineage": "3b050d0b-eef1-754c-23a6-9426065a5019",
  "outputs": {},
  "resources": [],
  "check_results": null
}
```

List all project files:

```
ls -la
```

```
@Zuhair-Irfan ~ /workspaces/CC-Zuhair-Irfan-073/Lab13 (main) $ ls -la
total 61776
drwxrwxrwx+ 5 codespace root          4096 Jan  1 18:30 .
drwxr-xrwx+ 5 codespace root          4096 Jan  1 15:21 .
drwxrwxrwx+ 8 codespace root          4096 Jan  1 15:21 .
drwxr-xr-x+ 3 codespace codespace    4096 Jan  1 18:02 .terraform
-rw-r--r--  1 codespace codespace    2422 Jan  1 16:44 .terraform.lock.hcl
-rw-rw-rw-  1 codespace root          10 Jan  1 15:21 README.md
drwxr-xr-x+ 3 codespace codespace    4096 Dec 30 19:13 aws
-rw-rw-rw-  1 codespace codespace 63198381 Jan  1 15:44 awscliv2.zip
-rwxrwxrwx  1 codespace codespace    423 Jan  1 16:37 create-login-profile.sh
-rw-rw-rw-  1 codespace codespace     50 Jan  1 18:17 locals.tf
-rw-rw-rw-  1 codespace codespace   2899 Jan  1 18:30 main.tf
-rw-rw-rw-  1 codespace codespace      0 Jan  1 18:02 terraform.tfstate
-rw-rw-rw-  1 codespace codespace   6882 Jan  1 18:02 terraform.tfstate.backup
-rw-rw-rw-  1 codespace codespace     167 Jan  1 18:18 users.csv
-rw-rw-rw-  1 codespace codespace    150 Jan  1 16:37 variable.tf
```

(Optional) Delete S3 bucket:

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

