

Fatima Jinnah Women University

Subject: Cloud Computing



Lab 13

Name:

Tehreem khan(5-B)

Registration number:

2023-BSE-064

Submitted To:

Sir Shoaib

Task 0:

```
PS C:\Users\tehre> gh codespace create --repo tehreem-0514/CC_TehreemKhan_064_Lab13
? Codespaces usage for this repository is paid for by tehreem-0514
? Choose Machine Type: 2 cores, 8 GB RAM, 32 GB storage
expert-palm-tree-r45g9x7x7vr6h5rx9
PS C:\Users\tehre> gh codespace list
```

NAME	DISPLAY NAME	REPOSITORY	BRANCH	STATE	CREATED AT
symmetrical-cod-9664p55vpjph95gv	symmetrical cod	tehreem-0514/CC_TehreemKhan_064	main*	Shutdown	about 6 days ago
didactic-waddle-9664p55vpvx4c7r56	didactic waddle	tehreem-0514/CC_TehreemKhan_064_Lab11	main*	Shutdown	about 6 days ago
refactored-goggles-wrq764v454vf9vrp	refactored goggles	tehreem-0514/CC_TehreemKhan_064_Lab12	main*	Shutdown	about 5 days ago
expert-palm-tree-r45g9x7x7vr6h5rx9	expert palm-tree	tehreem-0514/CC_TehreemKhan_064_Lab13	main	Available	less than a minute ago

Task 1:

```
@tehreem-0514 ② /workspaces/CC_TehreemKhan_064_Lab13 (main) $ mkdir -p ~/Lab13
@tehreem-0514 ② /workspaces/CC_TehreemKhan_064_Lab13 (main) $ cd ~/Lab13
@tehreem-0514 ② ~/Lab13 $
```

```
@tehreem-0514 ② ~/Lab13 $ touch main.tf
@tehreem-0514 ② ~/Lab13 $
```

```
GNU nano 7.2                                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
  }
}
```

```
@tehreem-0514 ② ~/Lab13 $ @tehreem-0514 ② ~/Lab13 $ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.28.0...
- Installed hashicorp/aws v6.28.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.
```

```
Terraform has been successfully initialized!
```

```
You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.
```

```
If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
@tehreem-0514 ② ~/Lab13 $
```

```

@tehreem-0514 ② ~/Lab13 $ terraform apply -auto-approve

Terraform used the selected providers to generate the following execution plan. Resource
actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

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

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

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

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

Outputs:

group_details = {
  "group_arn" = "arn:aws:iam::754080462526:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA27EVSW27I05KQPCYS"
}
@tehreem-0514 ② ~/Lab13 $ ■

```

```

@tehreem-0514 ② ~/Lab13 $ @tehreem-0514 ② ~/Lab13 $ terraform output
group_details = {
  "group_arn" = "arn:aws:iam::754080462526:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA27EVSW27I05KQPCYS"
}
@tehreem-0514 ② ~/Lab13 $

```

The screenshot shows the AWS IAM User Groups page. The left sidebar has 'Identity and Access Management (IAM)' selected. The main area title is 'User groups (1)'. A sub-header says 'A user group is a collection of IAM users. Use groups to specify permissions for a collection of users.' There is a search bar and a 'Create group' button. A table lists one group: 'developers' (Group name), '0' (Users), 'Not defined' (Permissions), and '2 minutes ago' (Creation time). Navigation icons are at the bottom right.

Group name	Users	Permissions	Creation time
developers	0	Not defined	2 minutes ago

Task 2:

```
GNU nano 7.2                                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
  }
}
```

```

+ name          = "loadbalancer"
+ path          = "/users/"
+ tags          = {
    + "DisplayName" = "Load Balancer"
}
+ tags_all     = {
    + "DisplayName" = "Load Balancer"
}
+ unique_id    = (known after apply)
}

# aws_iam_user_group_membership.lb_membership will be created
+ resource "aws_iam_user_group_membership" "lb_membership" {
    + groups = [
        + "developers",
    ]
    + id      = (known after apply)
    + user    = "loadbalancer"
}

Plan: 2 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ user_details  = {
    + unique_id = (known after apply)
    + user_arn  = (known after apply)
    + user_name = "loadbalancer"
}
aws_iam_user.lb: Creating...
aws_iam_user.lb: Creation complete after 1s [id=loadbalancer]
aws_iam_user_group_membership.lb_membership: Creating...
aws_iam_user_group_membership.lb_membership: Creation complete after 1s [id=terraform-20260108192525387700000001]

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

Outputs:

group_details = {
  "group_arn" = "arn:aws:iam::754080462526:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA27EVSW27I05KQPCYS"
}
user_details = {
  "unique_id" = "AIDA27EVSW27KTWHMHE5V"
  "user_arn" = "arn:aws:iam::754080462526:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@tehreem-0514 ② ~/Lab13 $
```

```

@tehreem-0514 ② ~/Lab13 $ terraform output
group_details = {
  "group_arn" = "arn:aws:iam::754080462526:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA27EVSW27I05KQPCYS"
}
user_details = {
  "unique_id" = "AIDA27EVSW27KTWHMHE5V"
  "user_arn" = "arn:aws:iam::754080462526:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@tehreem-0514 ② ~/Lab13 $
```

The image shows two screenshots of the AWS IAM console.

Screenshot 1: User groups

This screenshot shows the "User groups" page. It displays a table with one item: "developers". The "Users" column shows "1" user associated with the group, which is "loadbalancer". The "Permissions" column indicates "Not defined". The "Creation time" column shows "5 minutes ago".

Group name	Users	Permissions	Creation time
developers	1 loadbalancer	Not defined	5 minutes ago

Screenshot 2: User details

This screenshot shows the "Summary" tab for the user "loadbalancer".

Summary Details:

- ARN:** arn:aws:iam::754080462526:user/users/loadbalancer
- Console access:** Disabled
- Access key 1:** Create access key
- Created:** January 09, 2026, 00:25 (UTC+05:00)
- Last console sign-in:** -

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:

```
GNU nano 7.2                                main.tf *
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"
}
```

```

@tehreem-0514 ~ ~/Lab13 $ @tehreem-0514 ~ ~/Lab13 $ terraform apply -auto-approve
aws_iam_group.developers: Refreshing state... [id=developers]
aws_iam_user.lb: Refreshing state... [id=loadbalancer]
aws_iam_user_group_membership.lb_membership: Refreshing state... [id=terraform-202601081925253
877000000001]

Terraform used the selected providers to generate the following execution plan. Resource
actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

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

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

Plan: 2 to add, 0 to change, 0 to destroy.
aws_iam_group_policy_attachment.change_password: Creating...
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Creating...
aws_iam_group_policy_attachment.change_password: Creation complete after 1s [id=developers-202
60108192846214100000001]
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Creation complete after 1s [id=devel
opers-20260108192846236000000002]

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

Outputs:

group_details = {
  "group_arn" = "arn:aws:iam::754080462526:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA27EVSW27I05KQPCYS"
}
user_details = {
  "unique_id" = "AIDA27EVSW27KTWHMHE5V"
  "user_arn" = "arn:aws:iam::754080462526:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@tehreem-0514 ~ ~/Lab13 $

```

The screenshot shows the AWS IAM User Groups page. At the top, there's a navigation bar with the AWS logo, a search bar, and account information (Account ID: 7540-8046-2526, tehreem0514). Below the navigation, the URL is IAM > User groups > developers.

User group name	Creation time	ARN
developers	January 09, 2026, 00:21 (UTC+05:00)	arn:aws:iam::754080462526:group/groups/developers

Below the table, there are tabs for Users (1), Permissions (selected), and Access Advisor.

Permissions policies (2) Info

You can attach up to 10 managed policies.

Filter by Type

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

Task 4:

```
GNU nano 7.2                               variables.tf *
variable "iam_password" {
  description = "Temporary password for the IAM user"
  type        = string
  sensitive   = true
  default     = "IdontKnow"
}

GNU nano 7.2                               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
```

```
@tehreem-0514 ② ~/Lab13 $ chmod +x create-login-profile.sh
@tehreem-0514 ② ~/Lab13 $

GNU nano 7.2                                main.tf *

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"
}

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}'"
  }
}
```

```
Terraform will perform the following actions:

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

Plan: 1 to add, 0 to change, 0 to destroy.
null_resource.create_login_profile: Creating...
null_resource.create_login_profile: Provisioning with 'local-exec'...
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
null_resource.create_login_profile (local-exec): Creation complete after 3s [id=7283792120899687562]

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

Outputs:

```
group_details = {
  "group_arn" = "arn:aws:iam::754080462526:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA27EVSW27I05KQPCYS"
}
user_details = {
  "unique_id" = "AIDA27EVSW27KTWHMHE5V"
  "user_arn" = "arn:aws:iam::754080462526:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
```

```
@tehreem-0514 ② ~/Lab13 $ aws iam get-login-profile --user-name loadbalancer
{
  "LoginProfile": {
    "UserName": "loadbalancer",
    "CreateDate": "2026-01-08T19:33:25+00:00",
    "PasswordResetRequired": true
  }
}
@tehreem-0514 ② ~/Lab13 $
```



IAM user sign in [?](#)

Account ID or alias (Don't have?)

Remember this account

IAM username

Password

Show Password

[Having trouble?](#)

[Sign in](#)

[Sign in using root user email](#)

[Create a new AWS account](#)



Password reset (i)

Your account (**754080462526**) password has expired or requires a reset.

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

Old Password

Show Password

New Password

Confirm New Password

Show Password

Matches

Confirm Password Change

[Sign in to a different account](#)

Task 5:

```

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

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

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

[02601081928462360000000002]
aws_iam_user_group_membership.lb_membership: Refreshing state... [id=terraform-202601081925253
87700000001]
null_resource.create_login_profile: Refreshing state... [id=3980999670864814577]

Terraform used the selected providers to generate the following execution plan. Resource
actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# aws_iam_access_key.lb_access_key will be created
+ resource "aws_iam_access_key" "lb_access_key" {
    + create_date          = (known after apply)
    + encrypted_secret     = (known after apply)
    + encrypted_ses_smtp_password_v4 = (known after apply)
    + id                   = (known after apply)
    + key_fingerprint      = (known after apply)
    + secret               = (sensitive value)
    + ses_smtp_password_v4 = (sensitive value)
    + status               = "Active"
    + user                 = "loadbalancer"
  }

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

Changes to Outputs:
+ access_key_id      = (known after apply)
+ access_key_secret   = (sensitive value)
aws_iam_access_key.lb_access_key: Creating...
aws_iam_access_key.lb_access_key: Creation complete after 1s [id=AKIA27EVSW27B43GFWJP]

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

Outputs:

access_key_id = "AKIA27EVSW27B43GFWJP"
access_key_secret = <sensitive>
group_details = {
  "group_arn" = "arn:aws:iam::754080462526:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA27EVSW27I05KQPCYS"
}
user_details = {
  "unique_id" = "AIDA27EVSW27KTWHMHE5V"
  "user_arn" = "arn:aws:iam::754080462526:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@tehreem-0514 ② ~/Lab13 $
```

```
@tehreem-0514 ② ~/Lab13 $ terraform output
access_key_id = "AKIA27EVSW27B43GFWJP"
access_key_secret = <sensitive>
group_details = {
  "group_arn" = "arn:aws:iam::754080462526:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA27EVSW27I05KQPCYS"
}
user_details = {
  "unique_id" = "AIDA27EVSW27KTWHMHE5V"
  "user_arn" = "arn:aws:iam::754080462526:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@tehreem-0514 ② ~/Lab13 $
```

```
@tehreem-0514 ② ~/Lab13 $ cat terraform.tfstate | grep -A 10 "access_key_secret"
"access_key_secret": {
  "value": "tzSzGYCkJDItqys0uqlfJqh3sTa3rrlqIUgdTQro",
  "type": "string",
  "sensitive": true
},
"group_details": {
  "value": {
    "group_arn": "arn:aws:iam::754080462526:group/groups/developers",
    "group_name": "developers",
    "unique_id": "AGPA27EVSW27I05KQPCYS"
  }
},
@tehreem-0514 ② ~/Lab13 $
```

The screenshot shows the AWS IAM Access Keys page. At the top, there is a search bar and a navigation bar with the account ID (7540-8046-2526) and user (tehreem0514). Below the navigation, the URL is IAM > Users > loadbalancer. The main section is titled 'Access keys (1)' and contains a table with one row. The table columns are 'Description' (AKIA27EVSW27B43GFWJP), 'Status' (Active), and 'Actions'. The 'Last used' and 'Created' fields are also present.

Description	Status	Actions
AKIA27EVSW27B43GFWJP	Active	Actions

Task 6:

The screenshot shows the AWS S3 Buckets page. At the top, there is a search bar and a navigation bar with the account ID (7540-8046-2526) and region (Middle East (UAE)). Below the navigation, the URL is Amazon S3 > Buckets. The main section is titled 'General purpose buckets (1) [Info](#)' and contains a table with one row. The table columns are 'Name' (myapp-s3-bucket-demo-064), 'AWS Region' (Middle East (UAE) - central-1), and 'Creation date' (January 9, 2026, 01:47:27 (UTC+05:00)).

Name	AWS Region	Creation date
myapp-s3-bucket-demo-064	Middle East (UAE) - central-1	January 9, 2026, 01:47:27 (UTC+05:00)

Amazon S3 > Buckets > Create bucket

SS will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

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

```
terraform {  
  backend "s3" {  
    bucket = "myapp-s3-bucket-demo-064"  
    key    = "myapp/terraform.tfstate"  
    region = "me-central-1"  
    encrypt = true  
    use_lockfile = true  
  }  
}
```

File Operations:

Help: ^G
Exit: ^X

Text Operations:

Write Out: ^O
Read File: ^R
Where Is: ^W
Replace: ^\

Cut/Paste:

Cut: ^K
Paste: ^U

Execute: ^T
Justify: ^J

Location: ^C
Go To Line: ^/

```
@tehreem-0514 ~ ~/Lab13 $ @tehreem-0514 ~ ~/Lab13 $ terraform init -migrate-state  
Initializing the backend...  
Initializing provider plugins...  
- Reusing previous version of hashicorp/aws from the dependency lock file  
- Reusing previous version of hashicorp/null from the dependency lock file  
- Using previously-installed hashicorp/aws v6.28.0  
- Using previously-installed hashicorp/null v3.2.4  
  
Terraform has been successfully initialized!  
  
You may now begin working with Terraform. Try running "terraform plan" to see  
any changes that are required for your infrastructure. All Terraform commands  
should now work.  
  
If you ever set or change modules or backend configuration for Terraform,  
rerun this command to reinitialize your working directory. If you forget, other  
commands will detect it and remind you to do so if necessary.
```

```

# aws_iam_user.lb will be created
+ resource "aws_iam_user" "lb" {
    + arn          = (known after apply)
    + force_destroy = true
    + id           = (known after apply)
    + name         = "loadbalancer"
    + path         = "/users/"
    + tags         = {
        + "DisplayName" = "Load Balancer"
    }
    + tags_all     = {
        + "DisplayName" = "Load Balancer"
    }
    + unique_id    = (known after apply)
}

# aws_iam_user_group_membership.lb_membership will be created
+ resource "aws_iam_user_group_membership" "lb_membership" {
    + groups = [
        + "developers",
    ]
    + id      = (known after apply)
    + user    = "loadbalancer"
}

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

```

Plan: 7 to add, 0 to change, 0 to destroy.

Changes to Outputs:

```

+ access_key_id      = (known after apply)
+ access_key_secret = (sensitive value)
+ group_details     = {
    + group_arn   = (known after apply)
    + group_name  = "developers"
    + unique_id   = (known after apply)
}
+ user_details       = {
    + unique_id = (known after apply)
    + user_arn  = (known after apply)
    + user_name = "loadbalancer"
}

```

The screenshot shows the AWS S3 console interface. At the top, the account ID is 7540-8046-2526, and the region is Middle East (UAE). The path in the navigation bar is Amazon S3 > Buckets > myapp-s3-bucket-demo-064 > myapp/ > terraform.tfstate. The object name is terraform.tfstate. On the left, there are tabs for Properties, Permissions, and Versions. The Properties tab is selected. In the Object overview section, it shows the Owner (tahreem0514), AWS Region (Middle East (UAE) me-central-1), Last modified (January 9, 2026, 01:52:49 (UTC+05:00)), Size (181.0 B), and Type. To the right, there is an Object overview panel with sections for S3 URI (s3://myapp-s3-bucket-demo-064/myapp/terraform.tfstate), Amazon Resource Name (ARN) (arn:aws:s3:::myapp-s3-bucket-demo-064/myapp/terraform.tfstate), Entity tag (Etag) (f010960fb9c768533deb7827295e19d8), and Object URL (https://myapp-s3-bucket-demo-064.s3.me-central-1.amazonaws.com/myapp/terraform.tfstate). Below the object details, there is a terminal window showing the command: ls -la terraform.tfstate*. The output is:

```
@tahreem-0514 ~ ~/Lab13 $ ls -la terraform.tfstate*
-rw-rw-r-- 1 codespace codespace 0 Jan  8 20:52 terraform.tfstate
-rw-rw-r-- 1 codespace codespace 6882 Jan  8 20:52 terraform.tfstate.backup
@tahreem-0514 ~ ~/Lab13 $
```

Pretty print

```
{
  "version": 4,
  "terraform_version": "1.14.3",
  "serial": 1,
  "lineage": "e79d5cd0-ad15-c4b7-e020-78f9ccb7e88f",
  "outputs": {},
  "resources": [],
  "check_results": null
}
```

Task 7:

```
@tehreem-0514 ② /workspaces/CC_TehreemKhan_064_Lab13/Lab13 (main) $ cat locals.tf
locals {
    users = csvdecode(file("users.csv"))
}

@tehreem-0514 ② /workspaces/CC_TehreemKhan_064_Lab13/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
@tehreem-0514 ② /workspaces/CC_TehreemKhan_064_Lab13/Lab13 (main) $
```

```
GNU nano 7.2                                     main.tf *
  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
}
```

```
@tehreem-0514 ② /workspaces/CC_TehreemKhan_064_Lab13/Lab13 (main) $ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Finding latest version of hashicorp/null...
- Installing hashicorp/aws v6.28.0...
- Installed hashicorp/aws v6.28.0 (signed by HashiCorp)
- Installing hashicorp/null v3.2.4...
- Installed hashicorp/null v3.2.4 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
@tehreem-0514 ② /workspaces/CC_TehreemKhan_064_Lab13/Lab13 (main) $
```