Cloud Computing Lab

Lab 13 tasks

Lab 13 – Terraform IAM Management with AWS Balancer

Submitted To: Muhammad Shoaib

Submitted From: Bushra Ashraf Bhatti

Registration Number: 2023-BSE-015

CC LAB 13 – Terraform IAM Management with AWS:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

   inflating: aws/dist/prompt_toolkit-3.0.51.dist-info/licenses/AUTHORS.rst
   inflating: aws/dist/wheel-0.45.1.dist-info/entry_points.txt
   inflating: aws/dist/wheel-0.45.1.dist-info/INSTALLER
   inflating: aws/dist/wheel-0.45.1.dist-info/REQUESTED
   inflating: aws/dist/wheel-0.45.1.dist-info/METADATA
   inflating: aws/dist/wheel-0.45.1.dist-info/LICENSE.txt
   inflating: aws/dist/wheel-0.45.1.dist-info/direct_url.json
   inflating: aws/dist/wheel-0.45.1.dist-info/WHEEL
   inflating: aws/dist/wheel-0.45.1.dist-info/RECORD
@bushraashraf05 →/workspaces/13lab (main) $ sudo ./aws/install
You can now run: /usr/local/bin/aws --version
@bushraashraf05 →/workspaces/13lab (main) $ aws configure
AWS Access Key ID [None]: AKIA4ZT4HDFZEYYXX5OQ
AWS Secret Access Key [None]: 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
Default region name [None]: me-central-1
Default output format [None]: json
@bushraashraf05 →/workspaces/13lab (main) $ aws sts get-caller-identity
{
    "UserId": "AIDA4ZT4HDFZLL2WQERQ4",
    "Account": "879655065970",
    "Arn": "arn:aws:iam::879655065970:user/terraform-admin"
}
@bushraashraf05 →/workspaces/13lab (main) $
```

Main.tf:

```
 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
  }
}
```

```
@bushraashraf05 →/workspaces/13lab (main) $ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.27.0...
- Installed hashicorp/aws v6.27.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

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

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
@bushraashraf05 →/workspaces/13lab (main) $
```

```
@bushraashraf05 ➜ /workspaces/13lab (main) $ terraform validate
Success! The configuration is valid.

@bushraashraf05 ➜ /workspaces/13lab (main) $
```

```
    }

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::879655065970:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA4ZT4HDFZPMZRMNPII"
}
@bushraashraf05 ➜ /workspaces/13lab (main) $
```

Task 2 — Create IAM User with Group Membership:

Update 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" {
```

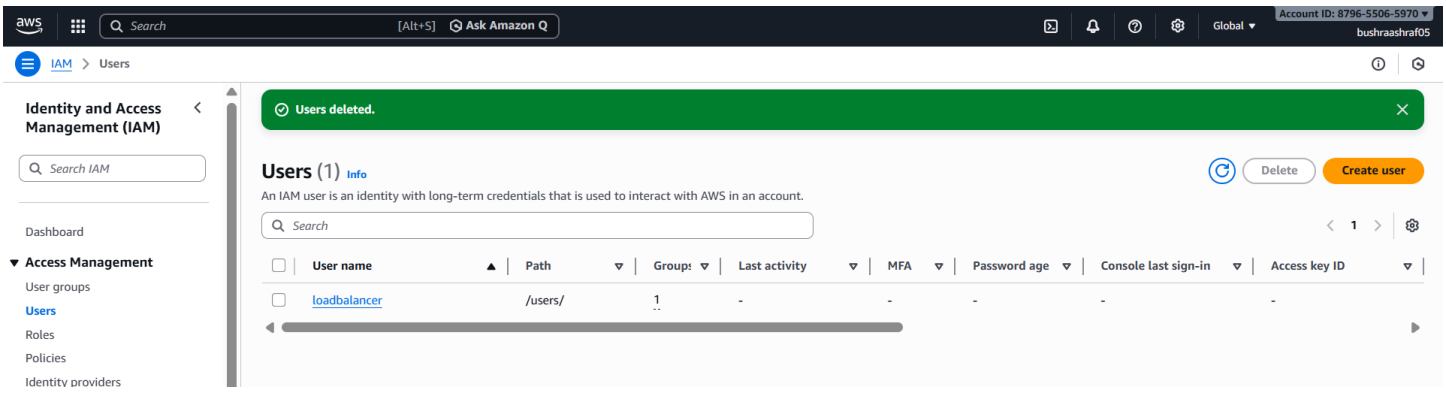
```
  value = {
    group_name = aws_iam_group.developers. name
    group_arn  = aws_iam_group. developers.arn
    unique_id  = aws_iam_group.developers.unique_id
  }
}

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
  }
}
```

Terraform-apply-auto approve:

```
@bushraashraf05 ➜ /workspaces/13lab (main) $ terraform apply -auto-approve
      + user_arn  = (known after apply)
      + user_name = "loadbalancer"
    }
aws_iam_user.lb: Creating...
aws_iam_user.lb: Creation complete after 1s [id=loadbalancer]
aws_iam_user_group_membership.lb_membership: Creating...
aws_iam_user_group_membership.lb_membership: Creation complete after 0s [id=terraform-20260106195536189800

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

Outputs:

group_details = {
  "group_arn" = "arn:aws:iam::879655065970:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA4ZT4HDFZPMZRMNPII"
}
user_details = {
  "unique_id" = "AIDA4ZT4HDFZC3NMB76V3"
  "user_arn" = "arn:aws:iam::879655065970:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@bushraashraf05 ➜ /workspaces/13lab (main) $
```

```
}
@bushraashraf05 ➜ /workspaces/13lab (main) $ terraform output
group_details = {
  "group_arn" = "arn:aws:iam::879655065970:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPA4ZT4HDFZPMZRMNPII"
}
user_details = {
  "unique_id" = "AIDA4ZT4HDFZC3NMB76V3"
  "user_arn" = "arn:aws:iam::879655065970:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
@bushraashraf05 ➜ /workspaces/13lab (main) $
```

Confirm on aws:



Task 3 — Attach Policies to IAM Group

Update 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"
}
```

```
@bushraashraf05 ➜/workspaces/13lab (main) $ nano main.tf
@bushraashraf05 ➜/workspaces/13lab (main) $ 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.
@bushraashraf05 ➜/workspaces/13lab (main) $
```

```
@bushraashraf05 ➜/workspaces/13lab (main) $ terraform validate
Success! The configuration is valid.

@bushraashraf05 ➜/workspaces/13lab (main) $
```

```
@bushraashraf05 ➜/workspaces/13lab (main) $ terraform apply -auto-approve
aws_iam_user.lb: Refreshing state... [id=loadbalancer]
aws_iam_group.developers: Refreshing state... [id=developers]

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

Terraform planned the following actions, but then encountered a problem:

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

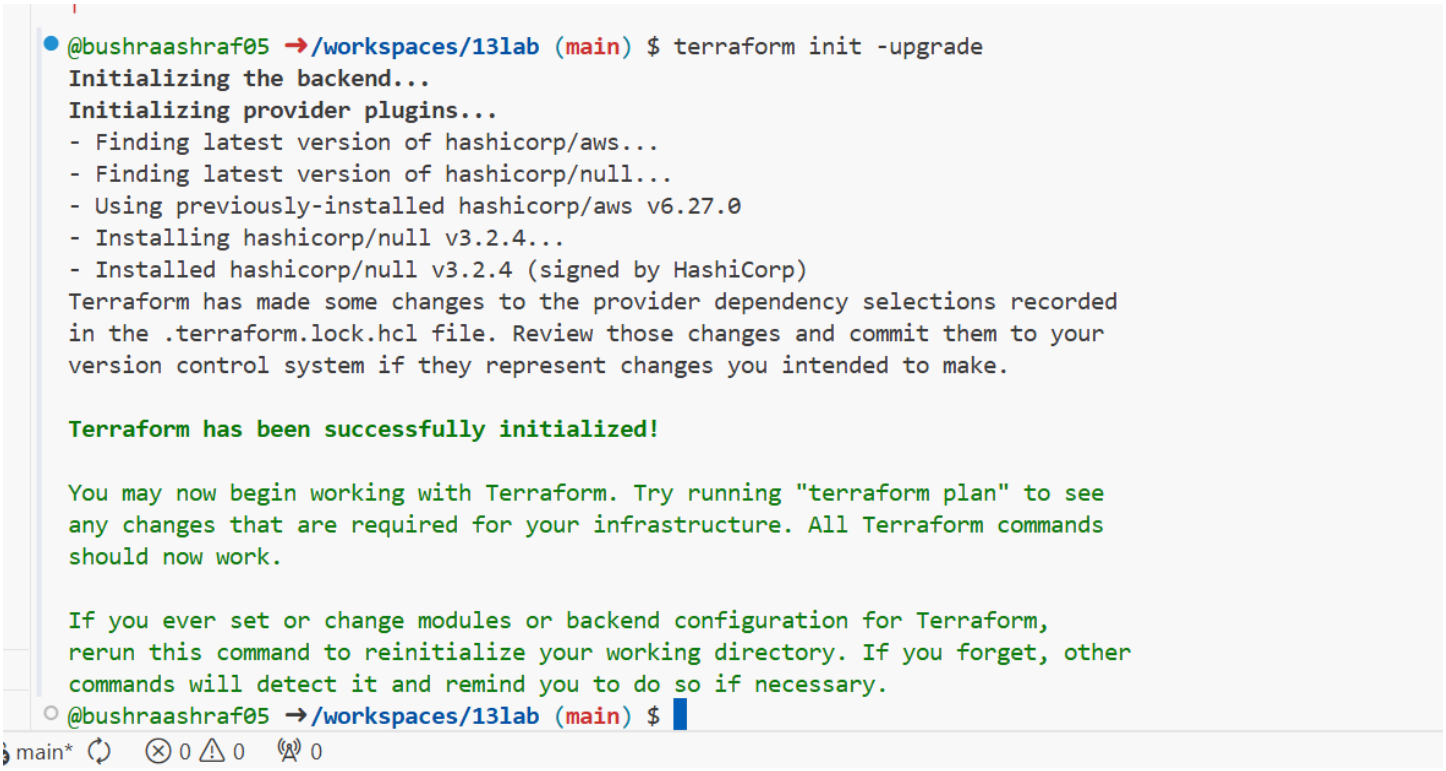
Task 4 — Create Login Profile for IAM User

```
  GNU nano 7.2                                    variables.tf *
variable "iam_password" {
  description = "Temporary password for the IAM user"
  type        = string
  sensitive   = true
  default = "1dontKnow"
}
```

```
  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
```

Update main.tf:

```
  GNU nano 7.2                                                  main.tf *
  value = {
    user_name = aws_iam_user.loadbalancer.name
    user_arn  = aws_iam_user.loadbalancer.arn
    unique_id = aws_iam_user.loadbalancer.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}'"
  }
}
```



Add this resource after 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}'"
  }
}
```

```
}
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 init –upgrade

 # 1. Download the installer

curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"

 # 2. Unzip it (Install unzip first if needed)

sudo apt-get install unzip -y

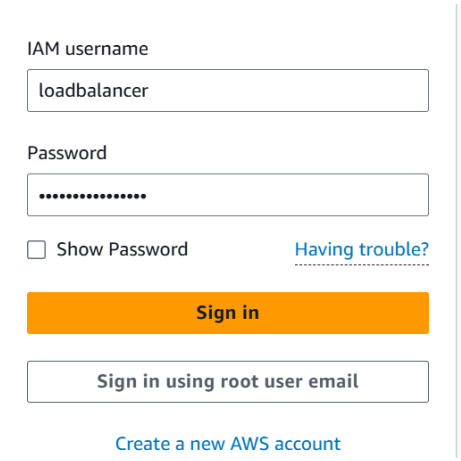
unzip -o awscliv2.zip

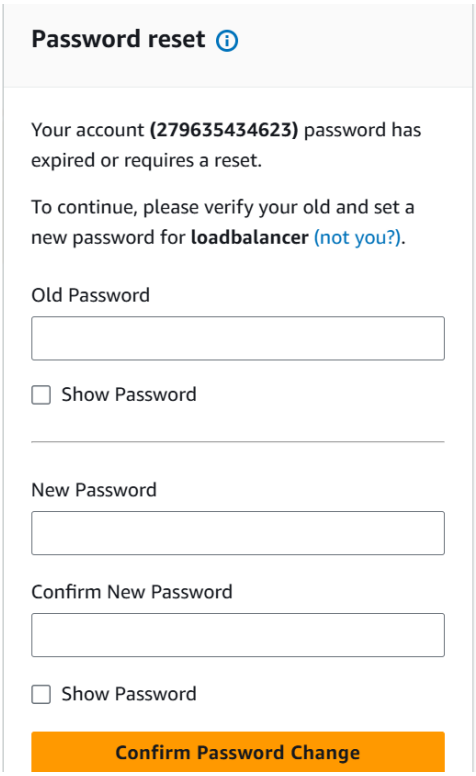
 # 3. Run the installer

sudo ./aws/install --update

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

```
null_resource.create_login_profile (local-exec): (output suppressed due to sensitive value in config)
null_resource.create_login_profile: Creation complete after 4s [id=8755808408375960610]

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

Outputs:

group_details = {
  "group_arn" = "arn:aws:iam::279635434623:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPAUCG4SDR7Z4TE35ZRX"
}
user_details = {
  "unique_id" = "AIDAUCG4SDR7ZJYJP2L4O"
  "user_arn" = "arn:aws:iam::279635434623:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
```





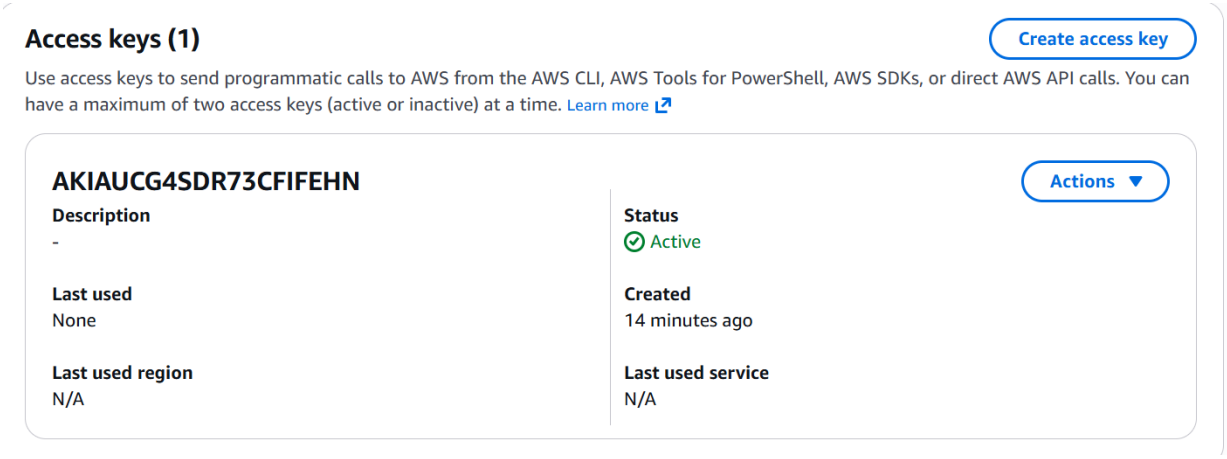
Task 5 — Generate Access Keys for IAM User:

```
  GNU nano 7.2                                                    main.tf *
# IAM User
################################
resource "aws_iam_user" "lb" {
  name          = "loadbalancer"
  path          = "/users/"
  force_destroy = true

  tags = {
    DisplayName = "Load Balancer"
  }
}

################################
# Login Profile (via script)
################################
resource "null_resource" "create_login_profile" {
  triggers = {
    password_hash = sha256(var.iam_password)
    user          = aws_iam_user.lb.name
  }
```

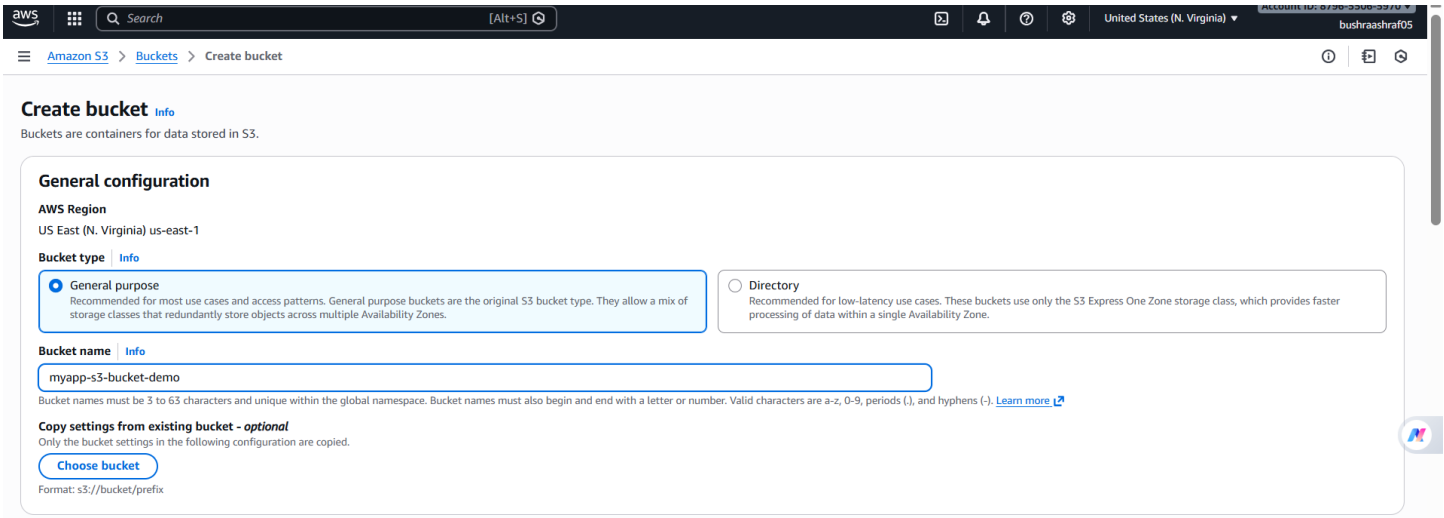
```
  "group_arn" = "arn:aws:iam::279635434623:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPAUCG4SDR7Z4TE35ZRX"
}
user_details = {
  "unique_id" = "AIDAUCG4SDR7ZJYJP2L4O"
  "user_arn" = "arn:aws:iam::279635434623:user/users/loadbalancer"
  "user_name" = "loadbalancer"
```

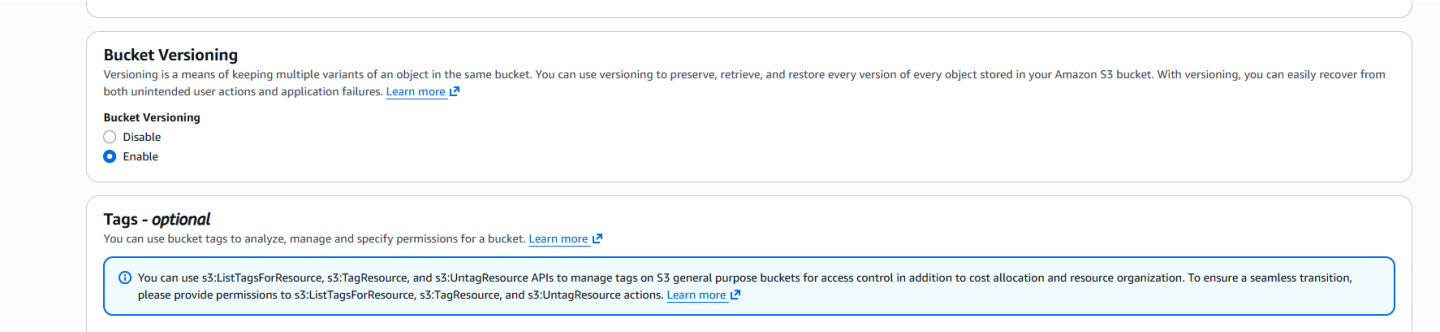
Terraform output:

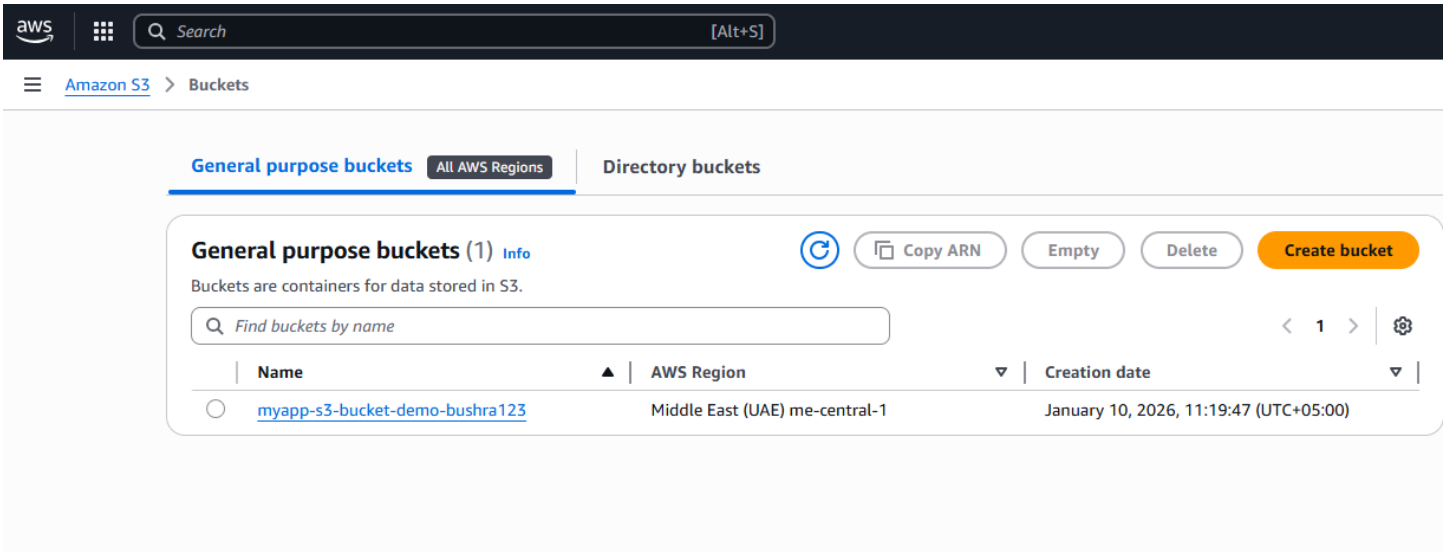
```
  "group_arn" = "arn:aws:iam::279635434623:group/groups/developers"
  "group_name" = "developers"
  "unique_id" = "AGPAUCG4SDR7Z4TE35ZRX"
}
user_details = {
  "unique_id" = "AIDAUCG4SDR7ZJYJP2L4O"
  "user_arn" = "arn:aws:iam::279635434623:user/users/loadbalancer"
  "user_name" = "loadbalancer"
}
```

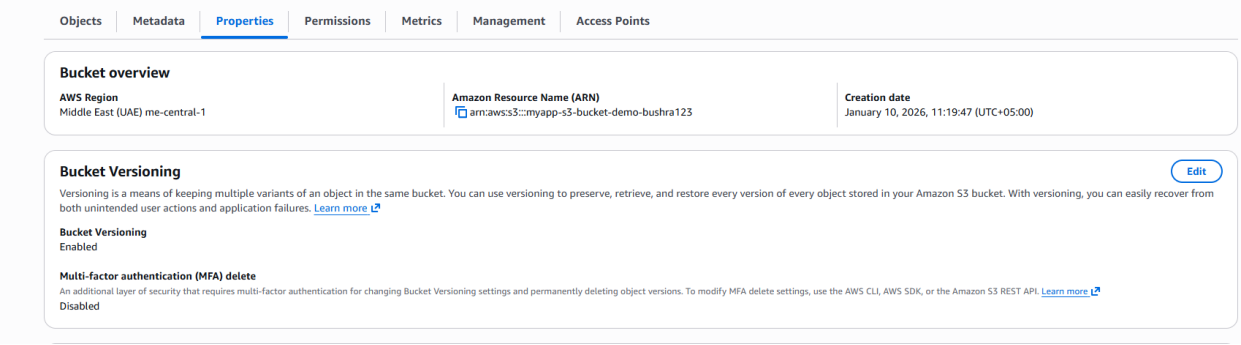


Task 6 — Implement Terraform Remote State with S3

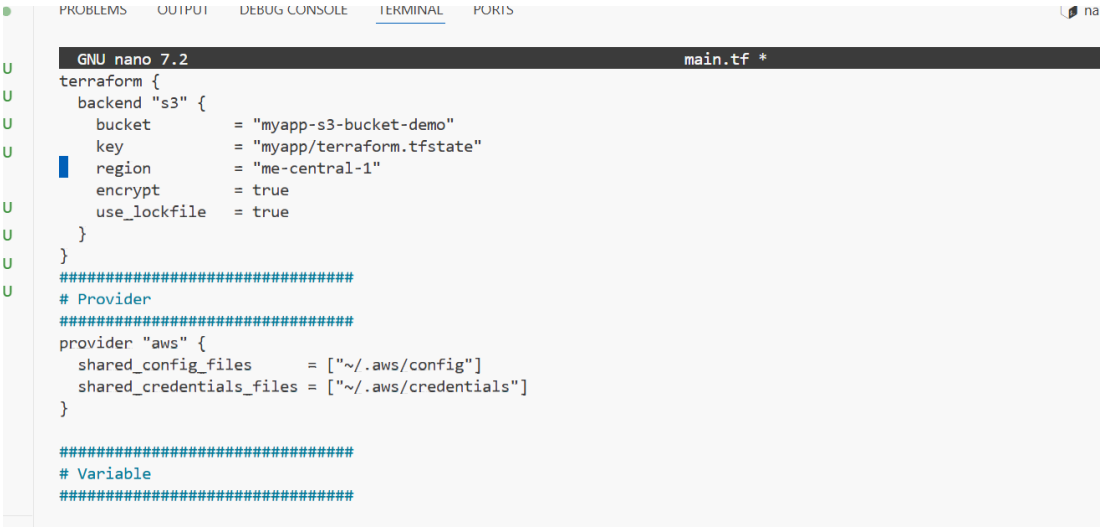






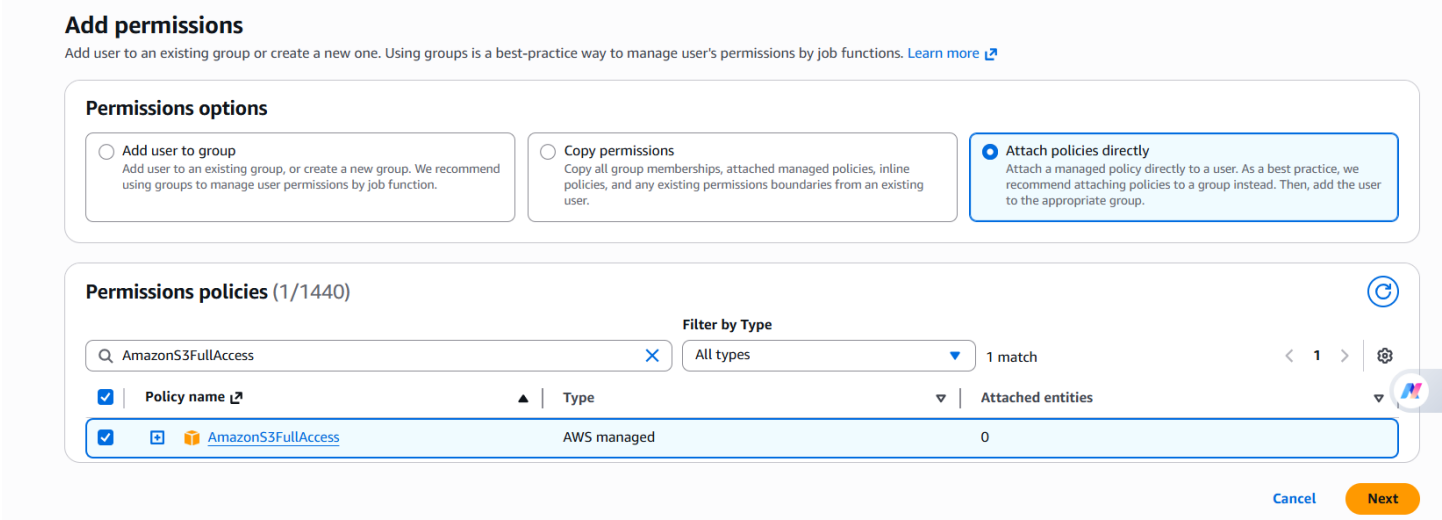


Update main.tf:



```
@bushraashraf05 ➜ /workspaces/13lab (main) $ terraform validate
Success! The configuration is valid.

@bushraashraf05 ➜ /workspaces/13lab (main) $
```

```
@bushraashraf05 ➜ /workspaces/13lab (main) $ aws sts get-caller-identity
{
    "UserId": "AIDA4ZT4HDFZC3NMB76V3",
    "Account": "879655065970",
    "Arn": "arn:aws:iam::879655065970:user/users/loadbalancer"
}
@bushraashraf05 ➜ /workspaces/13lab (main) $
```



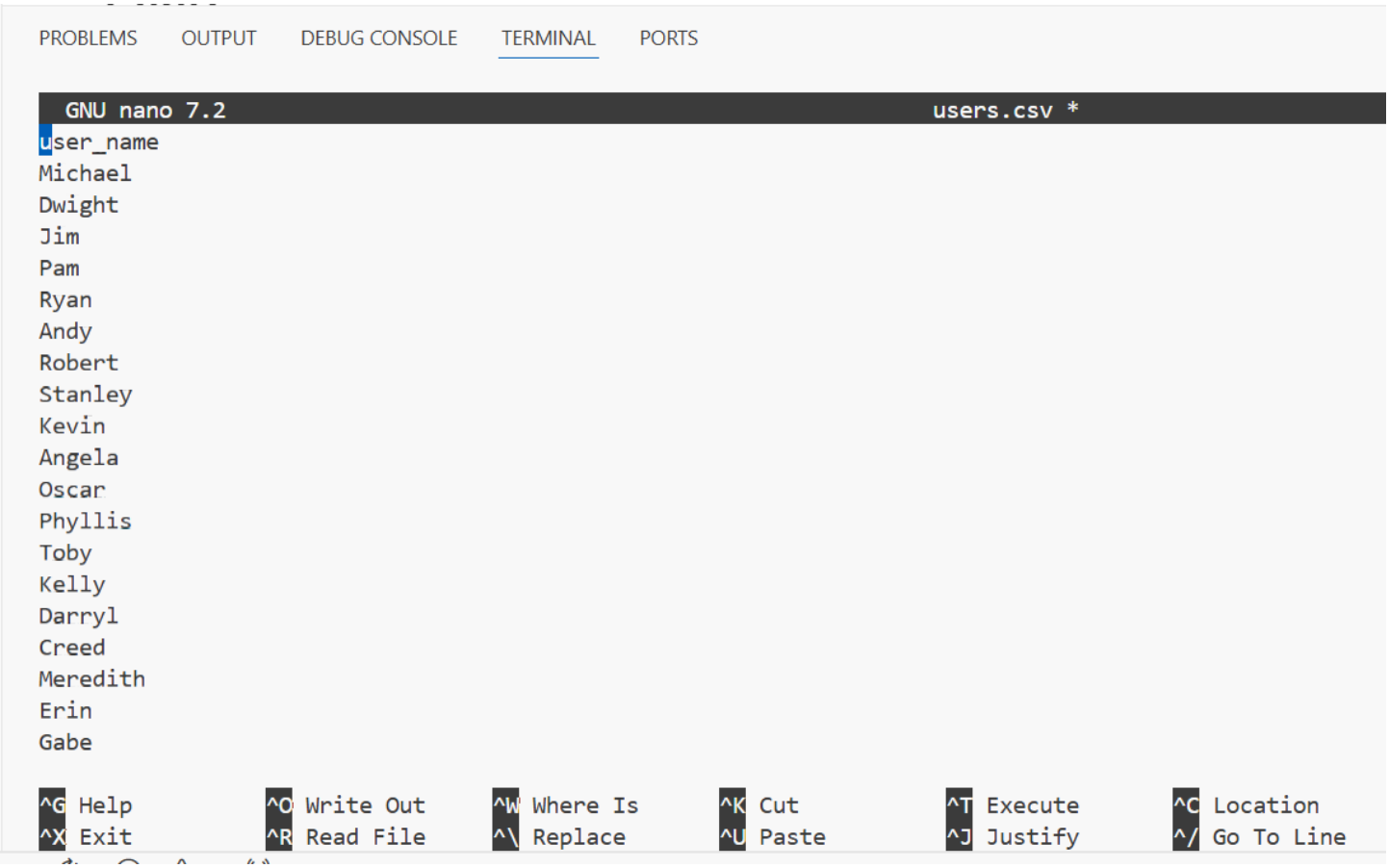

```
@bushraashraf05 ➜ /workspaces/13lab (main) $ rm -rf .terraform
@bushraashraf05 ➜ /workspaces/13lab (main) $ rm -f .terraform.lock.hcl
@bushraashraf05 ➜ /workspaces/13lab (main) $ terraform init -reconfigure -migrate-state
```

```
aws_iam_group_policy_attachment.developer_ec2_fullaccess: Destruction complete after 0s
aws_iam_user_group_membership.lb_membership: Destruction complete after 1s
aws_iam_group.developers: Destroying... [id=developers]
aws_iam_access_key.lb_access_key: Destruction complete after 1s
aws_iam_user.lb: Destroying... [id=loadbalancer]
aws_iam_group.developers: Destruction complete after 0s
aws_iam_user.lb: Destruction complete after 2s

Destroy complete! Resources: 7 destroyed.
```

Task 7 — Create Multiple Users from CSV File

```
  GNU nano 7.2                                                  locals.tf *
locals {
  users = csvdecode(file("users.csv"))
}
```

Create users.csv file:



Update main.tf:

(After IAM group resource):

Add this code:

```
###################################
# 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
    }
  }
}
```

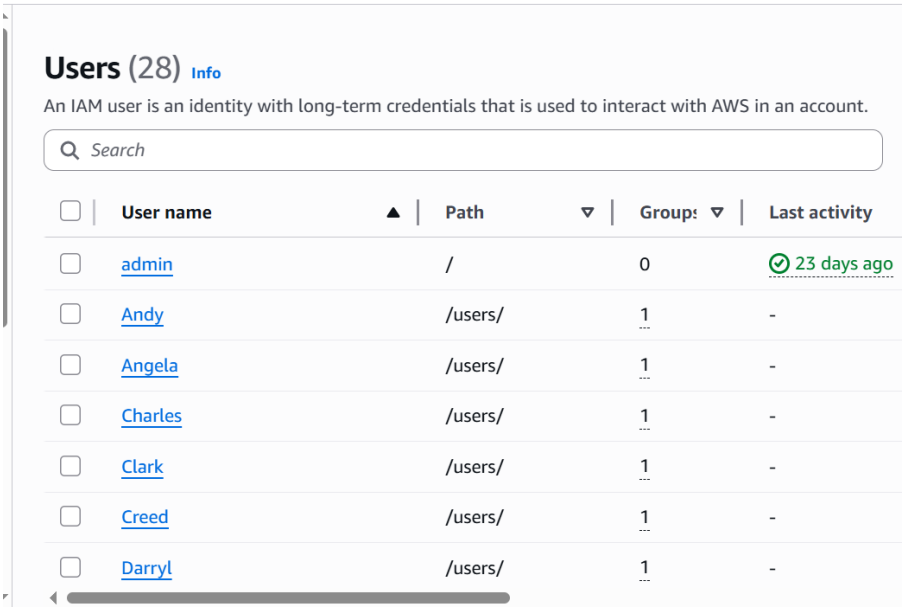
```
##################################
# Output all access key secrets (sensitive)
##################################
output "all_access_key_secrets" {
  value = {
    for user_name, key in aws_iam_access_key.users_access_keys :
    user_name => key.secret
  }
  sensitive = true
}
```

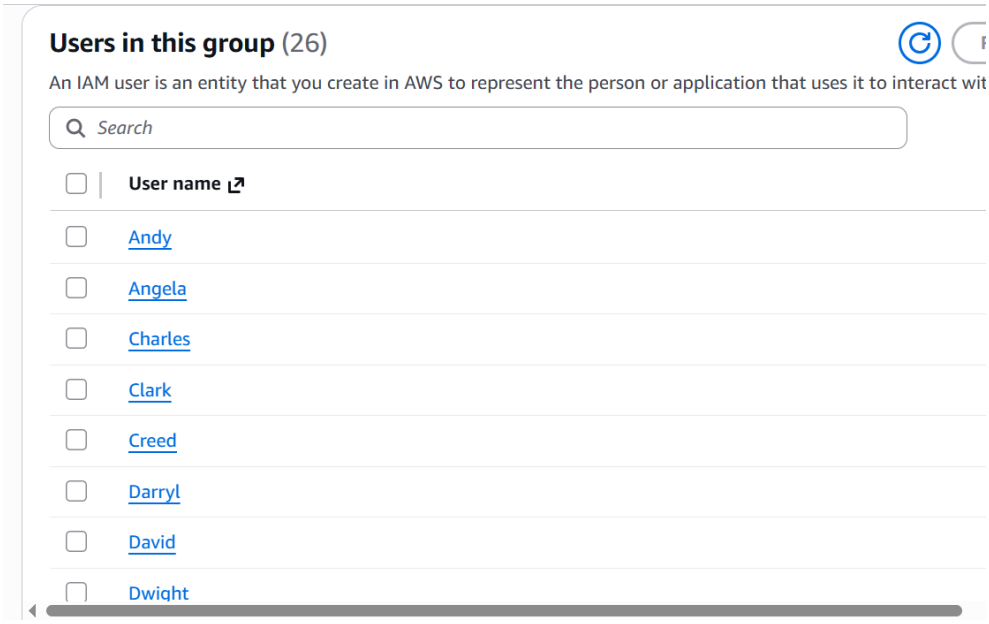
Terraform init:

```
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.
```

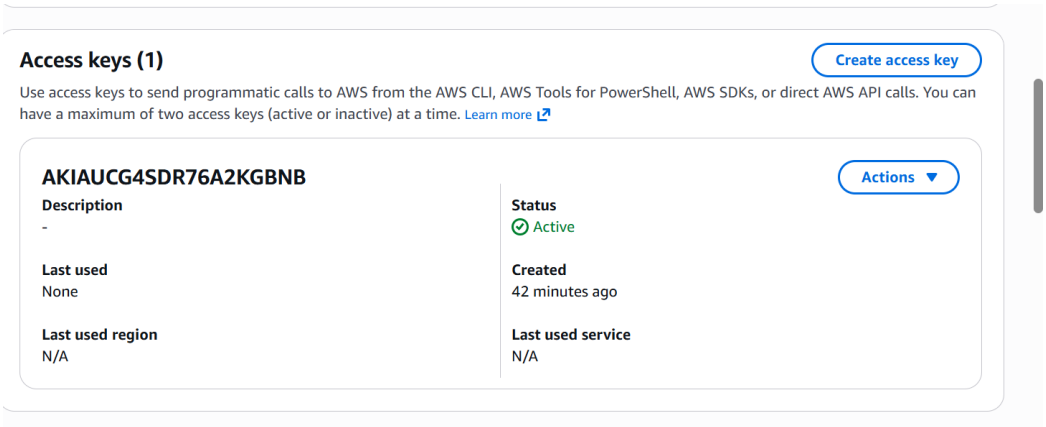
Apply the configuration to create all users:

```
      "user_unique_id" = "AIDAUCG4SDR7T27MJUDY3"
    }
    "Stanley" = {
      "access_key_id" = "AKIAUCG4SDR74XX5LVNR"
      "user_arn" = "arn:aws:iam::279635434623:user/users/Stanley"
      "user_unique_id" = "AIDAUCG4SDR7XYJNOBKI2"
    }
    "Toby" = {
      "access_key_id" = "AKIAUCG4SDR72DDAUOSN"
      "user_arn" = "arn:aws:iam::279635434623:user/users/Toby"
      "user_unique_id" = "AIDAUCG4SDR7UR6JT5D3I"
    }
  }
```

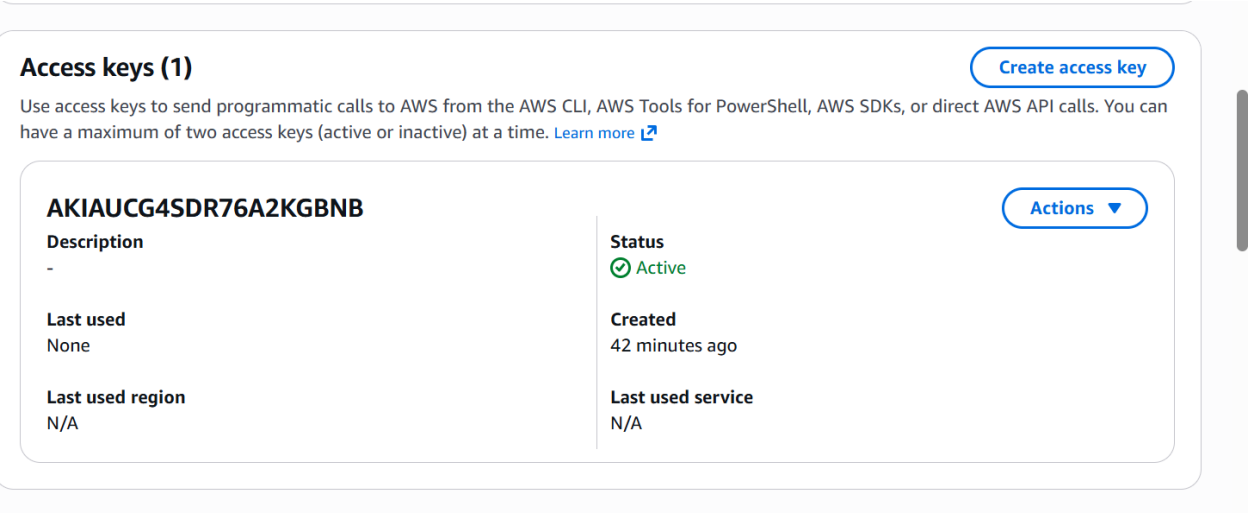




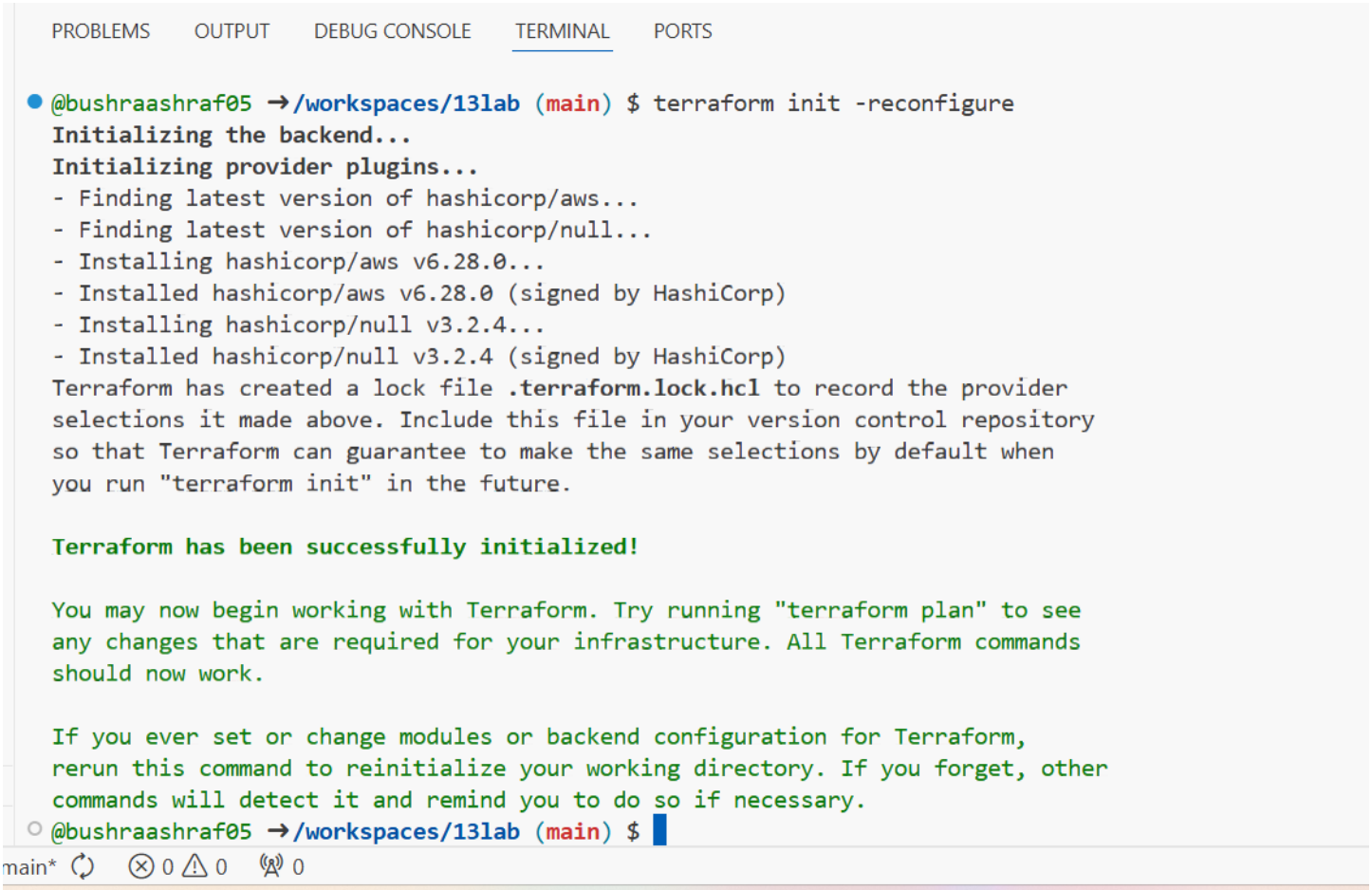
Verify one user's access keys:



Check terraform state in S3:



Destroy all reasources:



```
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.
@bushraashraf05 →/workspaces/13lab (main) $ terraform destroy -auto-approve
aws_iam_user.lb: Refreshing state... [id=loadbalancer]
aws_iam_user_group_membership.lb_membership: Refreshing state... [id=terraform-20260106195536189800000001]
aws_iam_group.developers: Refreshing state... [id=developers]
```

****