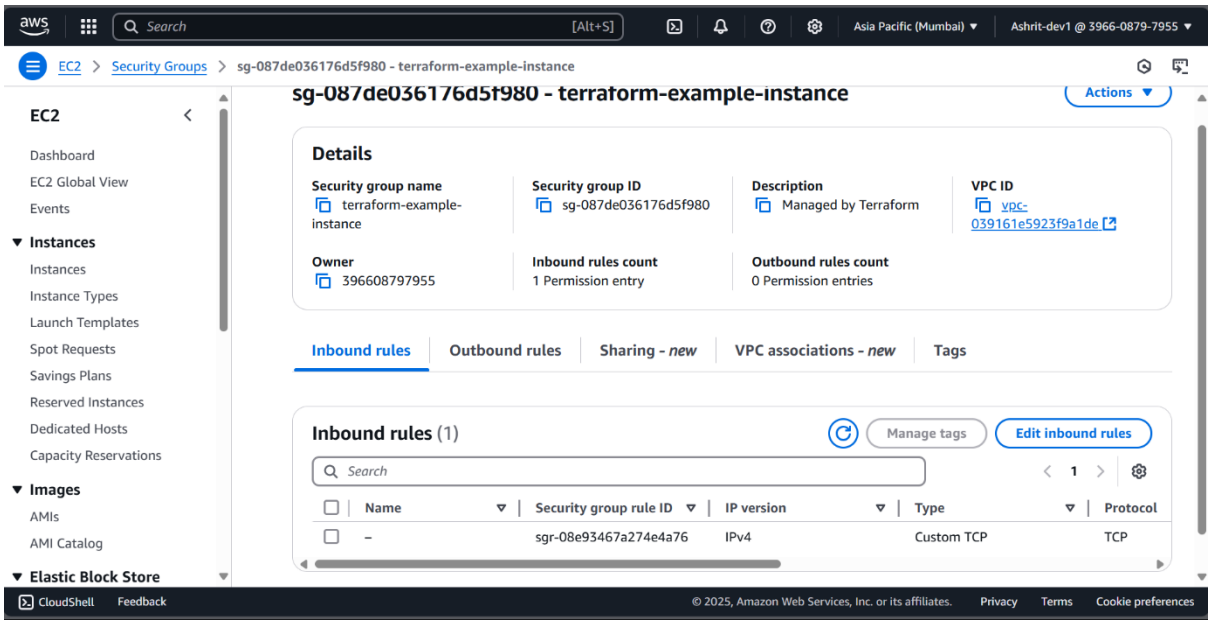
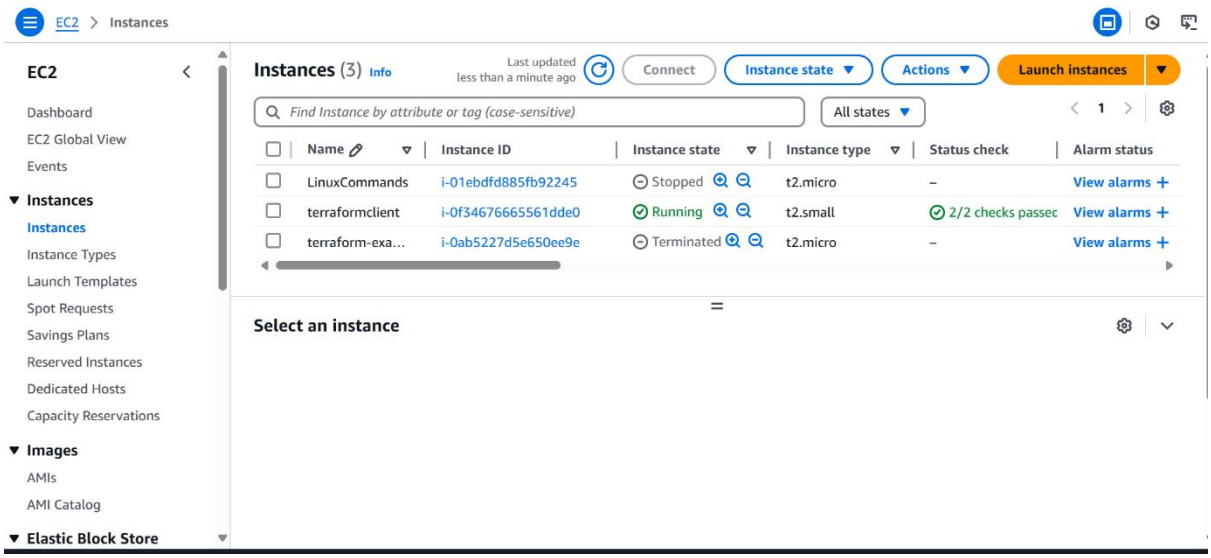
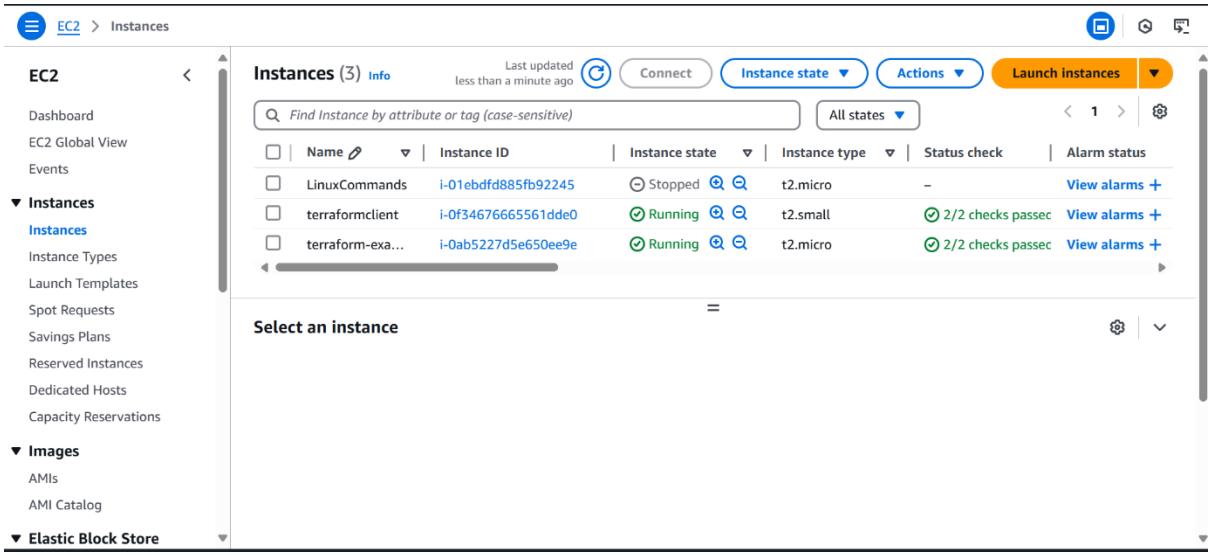
Week 8 Assignment 2








```
ubuntu@ip-172-31-41-104:~/demo1$ terraform destroy
aws_instance.example: Refreshing state... [id=i-0ab5227d5e650ee9e]

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

Terraform will perform the following actions:

  # aws_instance.example will be destroyed
  - resource "aws_instance" "example" {
      - ami                                  = "ami-0f1ba6e558ee844a5" -> null
      - arn                                  = "arn:aws:ec2:ap-south-1:396608797955:instance/i-0ab5227d5e650ee9e" -> null
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "ap-south-1a" -> null
      - cpu_core_count                       = 1 -> null
      - cpu_threads_per_core                 = 1 -> null
      - disable_api_stop                     = false -> null
      - disable_api_termination              = false -> null
      - ebs_optimized                        = false -> null
      - get_password_data                    = false -> null
      - hibernation                          = false -> null
      - id                                   = "i-0ab5227d5e650ee9e" -> null
      - instance_initiated_shutdown_behavior = "stop" -> null
      - instance_state                       = "running" -> null
      - instance_type                        = "t2.micro" -> null
      - ipv6_address_count                   = 0 -> null
      - ipv6_addresses                       = [] -> null
      - monitoring                           = false -> null
      - placement_partition_number           = 0 -> null
      - primary_network_interface_id         = "eni-0351e598918e4512b" -> null
      - private_dns                          = "ip-172-31-33-200.ap-south-1.compute.internal" -> null
      - private_ip                           = "172.31.33.200" -> null
      - public_dns                           = "ec2-3-108-191-44.ap-south-1.compute.amazonaws.com" -> null
```

```
ubuntu@ip-172-31-41-104:~/demo1$ terraform apply

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_instance.example will be created
  + resource "aws_instance" "example" {
      + ami                                  = "ami-0f1ba6e558ee844a5"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + enable_primary_ipv6                  = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = (known after apply)
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
      + placement_group                      = (known after apply)
```

```
ubuntu@ip-172-31-41-104:~/demo1$ terraform plan

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_instance.example will be created
  + resource "aws_instance" "example" {
      + ami                                  = "ami-0f1ba6e558ee844a5"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + enable_primary_ipv6                  = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = (known after apply)
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
```

```
Terraform v1.10.5
on linux_amd64
ubuntu@ip-172-31-41-104:~$ mkdir demo1
ubuntu@ip-172-31-41-104:~$ cd demo1/
ubuntu@ip-172-31-41-104:~/demo1$ vim main.tf
ubuntu@ip-172-31-41-104:~/demo1$ cat main.tf
provider "aws" {
  region = "ap-south-1"
}

resource "aws_instance" "example" {
  ami           = "ami-0f1ba6e558ee844a5"
  instance_type = "t2.micro"

  tags = {
    Name = "terraform-example"
  }
}
ubuntu@ip-172-31-41-104:~/demo1$ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.88.0...
- Installed hashicorp/aws v5.88.0 (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.
```

```
  # aws_iam_user_policy_attachment.attachment[2] will be created
  + resource "aws_iam_user_policy_attachment" "attachment" {
      + id         = (known after apply)
      + policy_arn = (known after apply)
      + user       = "jar"
    }

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

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_iam_policy.policy: Creating...
aws_iam_user.lb[0]: Creating...
aws_iam_user.lb[2]: Creating...
aws_iam_user.lb[1]: Creating...
aws_iam_user.lb[2]: Creation complete after 1s [id=jar]
aws_iam_user.lb[0]: Creation complete after 1s [id=foo]
aws_iam_user.lb[1]: Creation complete after 1s [id=bar]
aws_iam_policy.policy: Creation complete after 1s [id=arn:aws:iam::396608797955:policy/random-policy]
aws_iam_user_policy_attachment.attachment[1]: Creating...
aws_iam_user_policy_attachment.attachment[0]: Creating...
aws_iam_user_policy_attachment.attachment[2]: Creating...
aws_iam_user_policy_attachment.attachment[2]: Creation complete after 0s [id=jar-20250228112300063100000003]
aws_iam_user_policy_attachment.attachment[1]: Creation complete after 0s [id=bar-20250228112300058500000001]
aws_iam_user_policy_attachment.attachment[0]: Creation complete after 0s [id=foo-20250228112300059900000002]

Apply complete! Resources: 7 added, 0 changed, 0 destroyed.
ubuntu@ip-172-31-41-104:~/demo1$
```

Users (4) Info

Delete | Create user

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

Search

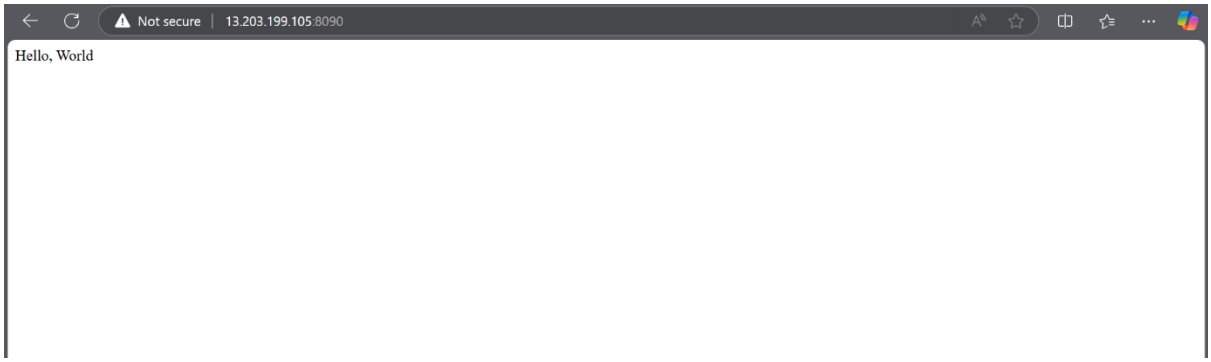
1

User name	Path	Groups	Last activity	MFA	Password age
Ashrit-dev1	/	1	9 minutes ago	-	3 days
bar	/system/	0	-	-	-
foo	/system/	0	-	-	-
jar	/system/	0	-	-	-

```
  # aws_iam_user_policy_attachment.attachment[2] will be created
  + resource "aws_iam_user_policy_attachment" "attachment" {
      + id         = (known after apply)
      + policy_arn = (known after apply)
      + user       = "jar"
    }

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

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_iam_policy.policy: Creating...
aws_iam_user.lb[0]: Creating...
aws_iam_user.lb[2]: Creating...
aws_iam_user.lb[1]: Creating...
aws_iam_user.lb[2]: Creation complete after 1s [id=jar]
aws_iam_user.lb[0]: Creation complete after 1s [id=foo]
aws_iam_user.lb[1]: Creation complete after 1s [id=bar]
aws_iam_policy.policy: Creation complete after 1s [id=arn:aws:iam::396608797955:policy/random-policy]
aws_iam_user_policy_attachment.attachment[1]: Creating...
aws_iam_user_policy_attachment.attachment[0]: Creating...
aws_iam_user_policy_attachment.attachment[2]: Creating...
aws_iam_user_policy_attachment.attachment[2]: Creation complete after 0s [id=jar-20250228112300063100000003]
aws_iam_user_policy_attachment.attachment[1]: Creation complete after 0s [id=bar-20250228112300058500000001]
aws_iam_user_policy_attachment.attachment[0]: Creation complete after 0s [id=foo-20250228112300059900000002]

Apply complete! Resources: 7 added, 0 changed, 0 destroyed.
ubuntu@ip-172-31-41-104:~/demo1$
```

```
Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_security_group.instance: Creating...
aws_security_group.instance: Creation complete after 2s [id=sg-038ef3ef59f49ea30]
aws_instance.example: Creating...
aws_instance.example: Still creating... [10s elapsed]
aws_instance.example: Creation complete after 12s [id=i-0e89bc617d837cf23]

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

Outputs:

public_ip = "15.207.18.70"
ubuntu@ip-172-31-41-104:~/demo1$
```

```
output "public_ip" {
  value       = aws_instance.example.public_ip
  description = "The public IP address of the web server"
}
ubuntu@ip-172-31-41-104:~/demo1$ terraform init
\Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.88.0

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.
ubuntu@ip-172-31-41-104:~/demo1$ terraform apply

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_instance.example will be created
  + resource "aws_instance" "example" {
      + ami                                  = "ami-0f1ba6e558ee844a5"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
```




```
ubuntu@ip-172-31-41-104:~/demo1$ terraform plan -var="server_port=8090"
aws_security_group.instance: Refreshing state... [id=sg-04e021a4be55f2204]
aws_instance.example: Refreshing state... [id=i-0e23d0898b274b2c0]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
  ~ update in-place
-/+ destroy and then create replacement

Terraform will perform the following actions:

  # aws_instance.example must be replaced
-/+ resource "aws_instance" "example" {
      ~ arn                                  = "arn:aws:ec2:ap-south-1:396608797955:instance/i-0e23d0898b274b2c0" -> (known af
ter apply)
      ~ associate_public_ip_address          = true -> (known after apply)
      ~ availability_zone                    = "ap-south-1b" -> (known after apply)
      ~ cpu_core_count                       = 1 -> (known after apply)
      ~ cpu_threads_per_core                 = 1 -> (known after apply)
      ~ disable_api_stop                     = false -> (known after apply)
      ~ disable_api_termination              = false -> (known after apply)
      ~ ebs_optimized                        = false -> (known after apply)
      + enable_primary_ipv6                  = (known after apply)
      - hibernation                          = false -> null
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      ~ id                                   = "i-0e23d0898b274b2c0" -> (known after apply)
      ~ instance_initiated_shutdown_behavior = "stop" -> (known after apply)
      + instance_lifecycle                   = (known after apply)
      ~ instance_state                       = "running" -> (known after apply)
      ~ ipv6_address_count                   = 0 -> (known after apply)
      ~ ipv6_addresses                       = [] -> (known after apply)
      + key_name                             = (known after apply)
```

```
ubuntu@ip-172-31-41-104:~/demo1$ cat main.tf
variable "server_port" {
  description = "The port the server will use for HTTP requests"
  type        = number
  default     = 8080
}

provider "aws" {
  region = "ap-south-1"
}

resource "aws_instance" "example" {
  ami                    = "ami-0f1ba6e558ee844a5"
  instance_type          = "t2.micro"
  vpc_security_group_ids = [aws_security_group.instance.id]
  user_data = <<-EOF
              #!/bin/bash
              echo "Hello, World" > index.html
              nohup busybox httpd -f -p ${var.server_port} &
              EOF

  user_data_replace_on_change = true

  tags = {
    Name = "terraform-example"
  }
}


resource "aws_security_group" "instance" {
  name = "terraform-example-instance"
  ingress {
    from_port   = var.server_port
    to_port     = var.server_port
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
```



```
ubuntu@ip-172-31-41-104:~/demo1$ vim main.tf
ubuntu@ip-172-31-41-104:~/demo1$ cat main.tf
variable "server_port" {
  description = "The port the server will use for HTTP requests"
  type        = number
  default     = 8080
}

provider "aws" {
  region = "ap-south-1"
}

resource "aws_instance" "example" {
  ami                    = "ami-0f1ba6e558ee844a5"
  instance_type          = "t2.micro"
  vpc_security_group_ids = [aws_security_group.instance.id]
  user_data = <<-EOF
              #!/bin/bash
              echo "Hello, World" > index.html
              nohup busybox httpd -f -p ${var.server_port} &
              EOF

  user_data_replace_on_change = true

  tags = {
    Name = "terraform-example"
  }
}
```