



# SPCM LAB

## LAB EXPERIMENTS

NAME: HIMANSH VATSAL

SAP ID: 500096347

BATCH: B-3(DEVOPS)

COURSE: SPCM

ROLL NO.: R2142211058

INSTRUCTOR: DR. Hitesh Sharma

S. No	EXPERIMENTS	Page no.
1	Install Terraform on machine	
2	Terraform AWS Provider and IAM user Settings	
3	Provisioning an EC2 Instance on AWS	
4	Terraform Variables	
5	Terraform Variables with Command Line Arguments	
6	Terraform Multiple vars Files	
7	Creating Multiple IAM Users in Terraform	
8	Creating a VPC in Terraform	
9	Creating Multiple EC2 Instances with for_each in Terraform	
10	Creating an AWS RDS Instance in Terraform	

# **EXPERIMENT-1**

```
shivasatyal@Shivs-MacBook-Air ~ % [echo $0 | awk '{ eval "$1</opt/homebrew/bin/brew shellenv" }' ] >> /Users/shivasatyal/.zprofile
shivasatyal@Shivs-MacBook-Air ~ % /bin/bash -c `curl https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh`
Checking for sudo access (which may request your password)...
This script will install:
  /opt/homebrew/bin/brew
  /opt/homebrew/share/doc/homebrew
  /opt/homebrew/share/man/man1/brew.1
  /opt/homebrew/share/man/man3/completion_.brew
  /opt/homebrew/share/man/man3/completion_d/brew
  /opt/homebrew

Press RETURN/ENTER to continue or any other key to abort:
++ /usr/bin/sudo /usr/bin/shh(shm -B shivasatyal/admin /opt/homebrew
[sudo] password for shivasatyal: 
Installing Homebrew...
HEAD is now at c0dcf5d Merge pull request #16598 from Homebrew/dependabot/bundler/Library/Homebrew/rspec-3.13.0
Installation successful!

Homebrew has enabled anonymous aggregate formulas and cask analytics.
Read the analytics documentation (and how to opt-out) here:
https://docs.brew.sh/Analytics.html

No analytics data has been sent yet (nor will any be during this install run).

Homebrew is run entirely by unpaid volunteers. Please consider donating:
https://github.com/Homebrew/brew#donations

Next steps:
- Run brew help to get started
- Further documentation:
  https://docs.brew.sh

shivasatyal@Shivs-MacBook-Air ~ % brew tap hashicorp/tap
brew install hashicorp/tap/terrafom
Tapping hashicorp/tap
Cloning into '/opt/homebrew/Library/Taps/hashicorp/homebrew-tap'...
remote: Counting objects: 100% (4088/4088), done.
remote: Compressing objects: 100% (3193/3193), done.
remote: Total 4088 (delta 2630), pack-reused 0
Receiving objects: 100% (4088/4088), 743.38 KB/s, done.
Resolving deltas: 100% (2630/2630), done.
2 casts and 29 files in 07 files, 181.7 KB.
-- Fetching hashicorp/tap/terrafom
  Downloading https://github.com/hashicorp/tap/terrafom/1.2/terrafom-1.2.2.d
  % Total    % Received ========= 100 0%
  Installing terrafom from hashicorp/tap
  Terrafom 1.2.2 is already installed.
Update that from Software Update in System Settings.

If that doesn't show you any updates, run:
  sudo rm -rf /Library/Developer/CommandLineTools
  sudo xcode-select --install

Alternatively, manually download them from:
  https://developer.apple.com/download/all/
You should download the Command Line Tools for Xcode 15.1.

  % /opt/homebrew/Cellar/terrafom/1.2.2: 3 files, 88.6MB, built in 4 seconds
  % Total    % Received ========= 100 0%
  Disabling this behaviour by setting HOMEBREW_NO_INSTALL_CLEANUP.
  Disable these hints with HOMEBREW_NO_ENV_HINTS (see 'man brew').
```

```
Last login: Thu Feb 22 21:31:29 on ttys004
shivavatsal@Shivas-Air ~ % terraform --version
Terraform v1.7.2
on darwin_arm64

Your version of Terraform is out of date! The latest version
is 1.7.4. You can update by downloading from https://www.terraform.io/downloads.html
shivavatsal@Shivas-Air ~ %
```

## EXPERIMENT-2

```
shivavatsal@Shivas-MacBook-Air ~ % cd Documents/
shivavatsal@Shivas-MacBook-Air Documents % mkdir terraform
shivavatsal@Shivas-MacBook-Air Documents % cd terraform
shivavatsal@Shivas-MacBook-Air terraform % mkdir dir1
shivavatsal@Shivas-MacBook-Air terraform % cd dir1
shivavatsal@Shivas-MacBook-Air dir1 % vi main.tf
shivavatsal@Shivas-MacBook-Air dir1 % terraform init

Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 5.0"...
- Installing hashicorp/aws v5.35.0...
- Installed hashicorp/aws v5.35.0 (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.
shivavatsal@Shivas-MacBook-Air dir1 % vi main.tf
shivavatsal@Shivas-MacBook-Air dir1 % terraform init

Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.35.0

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.
shivavatsal@Shivas-MacBook-Air dir1 %
```

```
MINGW64:/c/Users/abhis/Documents/terraform/dir1
$ terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = ">= 5.0"
    }
  }
}

provider "aws" {
  region = "us-east-1"
  access_key = "AKIAZQ3DXP3L163GMSTH"
  secret_key = "iWLeP1zWwX3LA+F7pS2J5D0hq4ZAiVqfIC4kpDK1"
}

# ... (remaining code from the image)

[1] el1.tf [unix] (10:14 07/02/2024)
[1] el1.tf [unix] 14L, 240B
10,19 411
23:46 24-02-2024
9°C Partly cloudy
Instances | EC2 | ap-southeast-2 | Relaunch to update Sydney ▾ hivatal ▾
[Option+5]
Find Instance by attribute or tag (case-sensitive)
Any state ▾
Name Instance ID Instance state Instance type Status check Alarm status Availability Zone Publ
UPES-EC2-Inst... i-0c865e6406fce6f23 Running t2.micro Initializing View alarms + ap-southeast-2a ec2-...
Select an instance
CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
```

## EXPERIMENT-3

```
Destroy complete! Resources: 0 destroyed.

(shivavatsal@Shivas-MacBook-Air:~/dir1% ls
main.tf          terraform-vpc           terraform.state
terraform         terraform-provider    terraform.tfstate
(shivavatsal@Shivas-MacBook-Air:~/dir1% cd aws-terraform-demo
(shivavatsal@Shivas-MacBook-Air:~/dir1% cd aws-terraform-demo
(shivavatsal@Shivas-MacBook-Air:~/aws-terraform-demo% vi main.tf
(shivavatsal@Shivas-MacBook-Air:~/aws-terraform-demo% terraform init

Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "0.31.0"...
- Installing hashicorp/aws v0.31.0 (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,
run this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
(shivavatsal@Shivas-MacBook-Air:~/aws-terraform-demo% vi instance.tf
(shivavatsal@Shivas-MacBook-Air:~/aws-terraform-demo% terraform init

Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v0.31.0
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,
run this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
(shivavatsal@Shivas-MacBook-Air:~/aws-terraform-demo% 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.My-Instance(*) will be created
+ resource "aws_instance" "My-instance" {
  ami                               = "ami-0aef097e0d1773b989"
  ami_id                           = "(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)"
```

```

shivavatsal@Shivas-MacBook-Air aws-terraform-demo % vi main.tf
shivavatsal@Shivas-MacBook-Air aws-terraform-demo % ls
LICENSE          main.tf          README.md        terraform.tfstate
shivavatsal@Shivas-MacBook-Air aws-terraform-demo % terraform terraform plan
Terraform has no command named "terrafrom".
To see all of Terraform's top-level commands, run:
  terraform -help

shivavatsal@Shivas-MacBook-Air aws-terraform-demo % terraform plan
+ create

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

Terraform will perform the following actions:

# aws_instance.My-instance(*) will be created
+ resource "aws_instance" "My-instance" {
  ami                               = "ami-04f5997681773b989"
  ami_id                            = "(known after apply)"
  arn                               = "(known after apply)"
  associate_public_ip_address       = "(known after apply)"
  availability_zone                 = "(known after apply)"
  cpu_core_count                    = "(known after apply)"
  cpu_type                          = "(known after apply)"
  disable_api_stop                 = "(known after apply)"
  disable_api_termination          = "(known after apply)"
  disable_eni_stretch               = "(known after apply)"
  get_password_data                = "(known after apply)"
  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)"
  ipv4_address                     = "(known after apply)"
  ipv4_address_count               = "(known after apply)"
  ipv6_addresses                   = "(known after apply)"
  monitoring                        = "(known after apply)"
  outputst_arn                      = "(known after apply)"
  placement_attributes              = "(known after apply)"
  placement_group                  = "(known after apply)"
  placement_partition_number       = "(known after apply)"
  private_ip                        = "(known after apply)"
  private_dns                       = "(known after apply)"
  private_ip_name                  = "(known after apply)"
  public_ip                         = "(known after apply)"
  secondary_private_ips            = "(known after apply)"
  source_dest_check                = "true"
  spot_instance_request_id         = "(known after apply)"
  subnet_id                         = "(known after apply)"
  tags {
    + "Name" = "UPES-EC2-Instmace"
  }
  tags_all                          = {
    + "Name" = "UPES-EC2-Instmace"
  }
  tenancy                           = "(known after apply)"
  user_data                          = "(known after apply)"
}

```

The screenshot shows the AWS Cloud Console interface for managing EC2 instances. On the left, the navigation menu includes 'Instances', 'Images', 'Elastic Block Store', 'Network & Security', and 'CloudShell'. The main content area displays a table of instances with one entry:

Name	Instance ID	Instance state	Type	Status check	Alarm status	Availability Zone	Public IP
UPES-EC2-Inst...	i-0c865e6406fce6f23	Running	t2.micro	Initializing		ap-southeast-2a	ec2-...

A modal window titled 'Select an instance' is overlaid on the page, containing the same information about the running instance.

## EXPERIMENT-4

```

AbhishekPandey@Groot9798 MINGW64 ~/Documents/terraform/dir1
$ mkdir terraform-variables

AbhishekPandey@Groot9798 MINGW64 ~/Documents/terraform/dir1
$ cd terraform-variables

AbhishekPandey@Groot9798 MINGW64 ~/Documents/terraform/dir1/terraform-variables
$ vi main.tf

AbhishekPandey@Groot9798 MINGW64 ~/Documents/terraform/dir1/terraform-variables
$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.31.0"...
- Installing hashicorp/aws v5.31.0...
- Installed hashicorp/aws v5.31.0 (signed by HashiCorp)

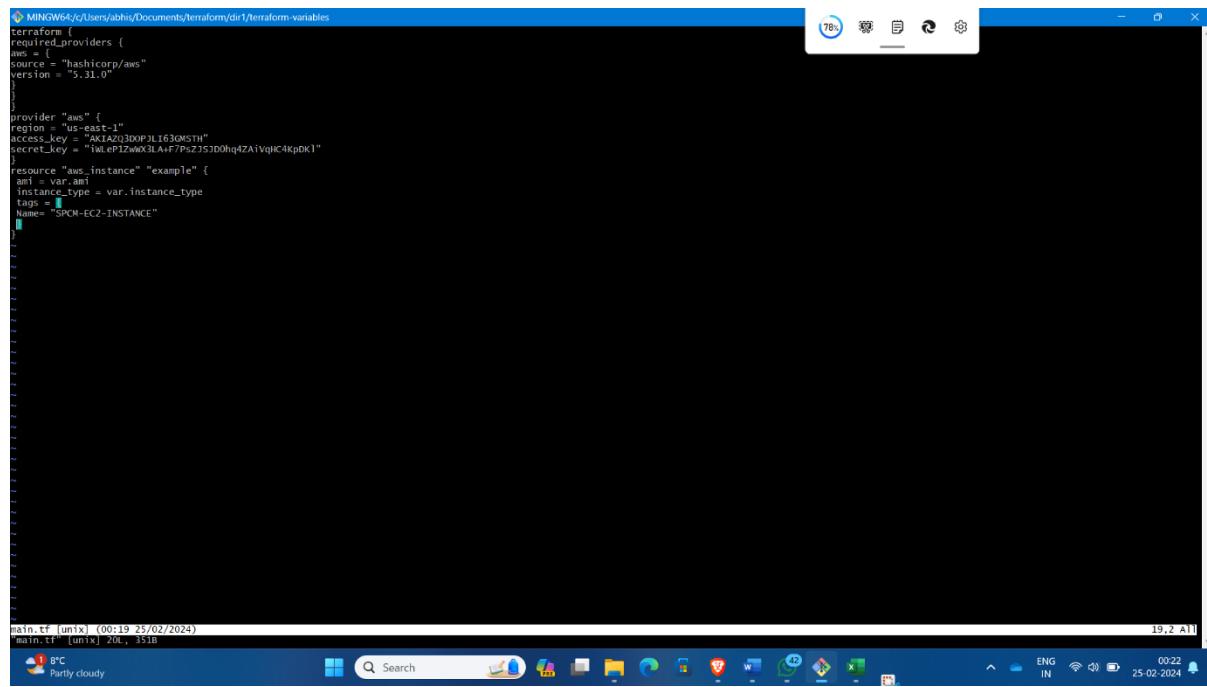
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

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.

```



```

$ MinGW64/c/Users/abhis/Documents/terraform/dir1/terraform-variables
terraform {
  required_providers {
    aws {
      source = "hashicorp/aws"
      version = "5.31.0"
    }
  }
}

provider "aws" {
  region = "us-east-1"
  access_key = "AKIAZQ3D0P3L163GM5H"
  secret_key = "iWLEP3ZwX3LA+F7PsZ35jD0hq4ZaiVqjC4kp0K"
}

resource "aws_instance" "example" {
  ami           = var.ami
  instance_type = var.instance_type
  tags          = [
    {Name="SPCM-EC2-INSTANCE"}
  ]
}

```

main.tf [unix] (00:19 25/02/2024)  
 main.tf [unix] 20L, 353B  
 78% 8°C Partly cloudy 00:22 ENG IN 25-02-2024 19,2 A1

```
MINGW64/c/Users/abhi/Documents/terraform/dir1/terraform-variables
variable "region" {
  description = "AWS region"
  default = "us-west-1"
}
variable "ami" {
  description = "AMI ID"
  default = "ami-0c7217cdde317cfec"
}
variable "instance_type" {
  description = "EC2 instance Type"
  default = "t2.micro"
}

variables.tf [unix] (00:20 25/02/2024)
variables.tf [unix] 12L, 23/B

variables.tf [unix] (00:20 25/02/2024) 12,1 All
variables.tf [unix] 12L, 23/B

MINGW64/c/Users/abhi/Documents/terraform/dir1/terraform-variables
- Using previously-installed hashicorp/aws v5.31.0

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.

abhishekPandey@root9798 MINGW64 ~/Documents/terraform/dir1/terraform-variables
$ vi variables.tf
abhishekPandey@root9798 MINGW64 ~/Documents/terraform/dir1/terraform-variables
$ terraform init

Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.31.0

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.

abhishekPandey@root9798 MINGW64 ~/Documents/terraform/dir1/terraform-variables
$ 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-0c7217cdde317cfec"
    arn          = (Known after apply)
    associate_public_ip_address = (Known after apply)
    availability_zone      = (Known after apply)
    ebs_optimized        = (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)
    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)
```

```
MINGW64:/c/Users/abhis/Documents/terraform/dir1/terraform-variables
+ ami
  = "ami-0c7217cdde317cfec"
+ 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_ami_stop
  = (known after apply)
+ disable_api_termination
  = (known after apply)
+ ebs_optimized
  = (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
+ instance_state
+ instance_type
  = "t2.micro"
+ ipv6_address_count
  = (known after apply)
+ ipv6_addresses
  = (known after apply)
+ key_name
  = (known after apply)
+ monitoring
  = (known after apply)
+ network_interface
  = (known after apply)
+ password_data
  = (known after apply)
+ placement_group
  = (known after apply)
+ placement_partition_number
  = (known after apply)
+ primary_network_interface_id
  = (known after apply)
+ private_dns
  = (known after apply)
+ private_ip
  = (known after apply)
+ public_dns
  = (known after apply)
+ public_ip
  = (known after apply)
+ secondary_private_ips
  = (known after apply)
+ security_groups
  = (known after apply)
+ source_dest_check
  = true
+ spot_instance_request_id
  = (known after apply)
+ subnet_id
  = (known after apply)
+ tags
  = {
    "Name" = "SPCM-EC2-INSTANCE"
  }
+ tags.all
  = {
    "Name" = "SPCM-EC2-INSTANCE"
  }
+ tenancy
  = (known after apply)
+ user_data
  = (known after apply)
+ user_data_base64
  = (known after apply)
+ user_data_replace_on_change
  = false
+ vpc_security_group_ids
  = (known after apply)
}

Plan: 1 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_instance.example: creating...
```

8°C Partly cloudy

```
MINGW64:/c/User/abhis/Documents/terraform/dir1/terraform-variables
+ instance_lifecycle
  = (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)
+ network_interface
  = (known after apply)
+ password_data
  = (known after apply)
+ placement_group
  = (known after apply)
+ placement_partition_number
  = (known after apply)
+ primary_network_interface_id
  = (known after apply)
+ private_dns
  = (known after apply)
+ private_ip
  = (known after apply)
+ public_dns
  = (known after apply)
+ public_ip
  = (known after apply)
+ secondary_private_ips
  = (known after apply)
+ security_groups
  = (known after apply)
+ source_dest_check
  = (known after apply)
+ spot_instance_request_id
  = (known after apply)
+ subnet_id
  = (known after apply)
+ tags
  = {
    "Name" = "SPCM-EC2-INSTANCE"
  }
+ tags.all
  = {
    "Name" = "SPCM-EC2-INSTANCE"
  }
+ tenancy
  = (known after apply)
+ user_data
  = (known after apply)
+ user_data_base64
  = (known after apply)
+ user_data_replace_on_change
  = false
+ vpc_security_group_ids
  = (known after apply)
}

Plan: 1 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_instance.example: Creating...
aws_instance.example: Still creating... [10s elapsed]
aws_instance.example: Still creating... [20s elapsed]
aws_instance.example: Still creating... [30s elapsed]
aws_instance.example: Creation complete after 39s [id=i-04bic1ldef4b2ba62]

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

AbhishekPandey@Groot9798 MINGW64 ~/Documents/terraform/dir1/terraform-variables
$ vi main.tf
AbhishekPandey@Groot9798 MINGW64 ~/Documents/terraform/dir1/terraform-variables
$ vi variables.tf
AbhishekPandey@Groot9798 MINGW64 ~/Documents/terraform/dir1/terraform-variables
$ |
```

8°C Partly cloudy

Screenshot of the AWS CloudShell interface showing the EC2 Instances page and a terminal session.

**AWS CloudShell**

**EC2 Dashboard**

**Instances**

- Instances
- Instance Types
- Launch Templates
- Spot Requests
- Savings Plans
- Reserved Instances
- Dedicated Hosts
- Capacity Reservations

**Images**

- AMIs
- AMI Catalog

**Elastic Block Store**

- Volumes

**CloudShell** Feedback

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

00:25 25-02-2024 ENG IN

**Instances (1) Info**

Find Instance by attribute or tag (case-sensitive)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
SPCM-EC2-INS...	i-04b1c11def4b2ba62	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1d	ec2-100-2...

Select an instance

MINGW64:/c/Users/AbhishekPandey/Documents/terraform/dir1/terraform-variables

```

$ vi main.tf
$ vi variables.tf
$ terraform destroy
aws_instance.example: Refreshing state... [id=i-04b1c11def4b2ba62]
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
    - associate_public_ip_address
    - availability_zone
    - cpu_core_count
    - ebs_optimized
    - disable_api_termination
    - disable_api_stop
    - ebs_optimized
    - ebs_optimized
    - hibernation
    - id
    - instance_initiated_shutdown_behavior
    - ipv6_address_count
    - ipv6_addresses
    - monitoring
    - placement_group
    - primary_network_interface_id
    - private_dns
    - private_ip
    - public_ip
    - secondary_private_ips
    - security_groups
    - source_dest_check
    - subnet_id
    - tags
      - "Name" = "SPCM-EC2-INSTANCE"
    ] -> null
  - tags_all
    - "Name" = "SPCM-EC2-INSTANCE"
  } -> null
  - tenancy
  - user_data_replace_on_change
  - vpc_security_group_ids
    - sg-097ab5b52bc492",
  ] -> null
}

```

8°C Partly cloudy

00:27 25-02-2024 ENG IN

```
MINGW64/c:/User/abhishek/Documents/terraform/dir1/terraform-variables
credit_specification {
  cpu_credits = "standard" -> null
}
enclave_options {
  enabled = false -> null
}
maintenance_options {
  auto_recovery = "default" -> null
}
metadata_options {
  http_endpoint           = "enabled" -> null
  http_endpoint_ip_v6     = "disabled" -> null
  http_put_response_hop_limit = 1 -> null
  http_tokens             = "optional" -> null
  instance_metadata_tags   = "disabled" -> null
}
private_dns_name_options {
  enable_resource_name_dns_a_record = false -> null
  enable_resource_name_dns_aaaa_record = false -> null
  hostname_type                  = "ip-name" -> null
}
root_block_device {
  delete_on_termination = true -> null
  device_name          = "/dev/sda1" -> null
  encrypted            = false -> null
  iops                 = 100 -> null
  tags                 = {} -> null
  throughput            = 0 -> null
  volume_id             = "vol-08a7d2074d5a2505a" -> null
  volume_size           = 8 -> null
  volume_type           = "gp2" -> null
}
}

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

Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

aws_instance.example: Destroying... [id=i-04b1c11def4b2ba62]
aws_instance.example: Still destroying... [id=i-04b1c11def4b2ba62, 10s elapsed]
aws_instance.example: Still destroying... [id=i-04b1c11def4b2ba62, 20s elapsed]
aws_instance.example: Still destroying... [id=i-04b1c11def4b2ba62, 30s elapsed]
aws_instance.example: Still destroying... [id=i-04b1c11def4b2ba62, 40s elapsed]
aws_instance.example: Destruction complete after 44s

Destroy complete! Resources: 1 destroyed.

abhishekPandey@GrootE9798 MINGW64 ~/Documents/terraform/dir1/terraform-variables
```

The screenshot shows a dual-pane interface. The left pane is the AWS CloudShell terminal, displaying the command history and output from the previous Terraform destroy operation. The right pane is the AWS Management Console, specifically the EC2 Dashboard. It shows a table of instances with one entry: 'SPCM-E2-INS...' (Instance ID: i-04b1c11def4b2ba62, Instance State: Terminated, Instance Type: t2.micro, Public IP: us-east-1d). Below the table is a modal dialog titled 'Select an instance' with a single option: 'i-04b1c11def4b2ba62'. The status bar at the bottom indicates the user is in N. Virginia and has a session ID of AbhishekPandey900.

## EXPERIMENT-5

```
AbhishekPandey@Groot9798 MINGW64 ~/Documents/terraform/dir1
$ mkdir terraform-variables

AbhishekPandey@Groot9798 MINGW64 ~/Documents/terraform/dir1
$ cd terraform-variables

AbhishekPandey@Groot9798 MINGW64 ~/Documents/terraform/dir1/terraform-variables
$ vi main.tf

AbhishekPandey@Groot9798 MINGW64 ~/Documents/terraform/dir1/terraform-variables
$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.31.0"...
- Installing hashicorp/aws v5.31.0...
- Installed hashicorp/aws v5.31.0 (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.
```

```
MINGW64:/c/Users/abhis/Documents/Terraform/d1/tf/terraform-variables
$ terraform {
  required_providers {
    aws {
      source = "hashicorp/aws"
      version = "5.31.0"
    }
  }
  provider "aws" {
    region = "us-east-1"
    access_key = "AKIAZQ3DXP3L163GMSTH"
    secret_key = "iWLept2wX3LA+FPs2J5D0hg4ZAiVgjIC4kpdk"
  }
  resource "aws_instance" "example" {
    ami           = var.ami
    instance_type = var.instance_type
    tags = {
      Name = "SPCM-EC2-INSTANCE"
    }
  }
}
variable "region" {
  description = "AWS region"
  default = "us-west-1"
}
variable "ami" {
  description = "AMI ID"
  default = "ami-0c7217cdde317cfec"
}
variable "instance_type" {
  description = "EC2 Instance Type"
  default = "t2.micro"
}

variables.tf [unix] (00:19 25/02/2024)
variables.tf [unix] 20L, 351B
19,2 A[1]

MINGW64:/c/Users/abhis/Documents/Terraform/d1/tf/terraform-variables
$ main.tf [unix] (00:19 25/02/2024)
main.tf [unix] 20L, 351B
19,2 A[1]

MINGW64:/c/Users/abhis/Documents/Terraform/d1/tf/terraform-variables
$ terraform {
  required_providers {
    aws {
      source = "hashicorp/aws"
      version = "5.31.0"
    }
  }
  provider "aws" {
    region = "us-east-1"
    access_key = "AKIAZQ3DXP3L163GMSTH"
    secret_key = "iWLept2wX3LA+FPs2J5D0hg4ZAiVgjIC4kpdk"
  }
  resource "aws_instance" "example" {
    ami           = var.ami
    instance_type = var.instance_type
    tags = {
      Name = "SPCM-EC2-INSTANCE"
    }
  }
}
variable "region" {
  description = "AWS region"
  default = "us-west-1"
}
variable "ami" {
  description = "AMI ID"
  default = "ami-0c7217cdde317cfec"
}
variable "instance_type" {
  description = "EC2 Instance Type"
  default = "t2.micro"
}

variables.tf [unix] (00:20 25/02/2024)
variables.tf [unix] 12L, 237B
12,1 A[1]

MINGW64:/c/Users/abhis/Documents/Terraform/d1/tf/terraform-variables
$ B°C
Partly cloudy
Search ENG IN 00:22 25-02-2024
```

```

AbhishekPandey@Groot9798 MINGW64 ~/Documents/terraform/dirl/terraform-variables
$ terraform apply -var "region=us-east-1" -var "ami=ami-0c7217cdde31cfec" -var
"instance_type=t2.micro"

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-0c7217cdde31cfec"
    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)
    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)
    placement_partition_number       = (known after apply)
    primary_network_interface_id    = (known after apply)
    private_dns                      = (known after apply)
    private_ip                       = (known after apply)
    public_dns                        = (known after apply)
    public_ip                         = (known after apply)
    secondary_private_ips           = (known after apply)
    security_groups                  = (known after apply)
    source_dest_check                = true
}

```

```

MINGW64:/Users/abhi/Documents/terraform/dirl/terraform-variables
$ aws_instance.example = (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)
$ placement_partition_number = (known after apply)
$ primary_network_interface_id = (known after apply)
$ private_dns = (known after apply)
$ private_ip = (known after apply)
$ public_dns = (known after apply)
$ public_ip = (known after apply)
$ secondary_private_ips = (known after apply)
$ security_groups = (known after apply)
$ source_dest_check = true
$ spot_instance_request_id = (known after apply)
$ subnet_id = (known after apply)
$ tags = (
  "Name" = "SPCM-EC2-INSTANCE"
)
$ tags_all = (
  "Name" = "SPCM-EC2-INSTANCE"
)
$ tenancy = (known after apply)
$ user_data = (known after apply)
$ user_data_base64 = (known after apply)
$ user_data_replace_on_change = false
$ vpc_security_group_ids = (known after apply)

Plan: 1 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_instance.example: Creating...
aws_instance.example: Still creating... [10s elapsed]
aws_instance.example: Still creating... [20s elapsed]
aws_instance.example: Still creating... [30s elapsed]
aws_instance.example: Creation complete after 37s [id=i-0852e9418bd508788]

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

AbhishekPandey@Groot9798 MINGW64 ~/Documents/terraform/dirl/terraform-variables
$ vi main.tf
AbhishekPandey@Groot9798 MINGW64 ~/Documents/terraform/dirl/terraform-variables
$ |

```

8°C Partly cloudy ENG IN 00:38 25-02-2024

Screenshot of the AWS CloudShell interface showing the AWS Management Console and a terminal window.

**AWS Management Console:**

- Left sidebar: EC2 Dashboard, EC2 Global View, Events, Console-to-Code, Instances (selected), Instances Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, New, Images (AMIs, AMI Catalog), Elastic Block Store (Volumes).
- Top bar: Search, [Alt+S], Connect, Instance state dropdown, Actions dropdown, Launch instances button.
- Table: Instances (2) Info. Rows: SPCM-EC2-INS... (terminated, t2.micro, us-east-1d, View alarms, ec2-3-82-2) and SPCM-EC2-INS... (running, t2.micro, initializing, View alarms, us-east-1d).
- Bottom bar: CloudShell, Feedback, © 2024, Amazon Web Services, Inc. or its affiliates., Privacy, Terms, Cookie preferences, ENG IN, 00:39, 25-02-2024.

**Terminal Window:**

```
MINGW64:/c/Users/AbhishekPandey/Documents/terraform/dirl/terraform-variables
$ vi main.tf
AbhishekPandey@Groot:798 MINGW64 ~/Documents/terraform/dirl/terraform-variables
$ terraform destroy
aws_instance.example: Refreshing state... [id=i-0852e9418bd508788]

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-0c7217cdde317cfec" -> null
    ami_arn                          = "arn:aws:ec2:us-east-1:654654208598:instance/i-0852e9418bd508788" -> null
    associate_public_ip_address       = true -> null
    availability_zone                = "us-east-1d" -> null
    core_dump                        = false -> null
    cpu_threads_per_core            = 1 -> null
    disable_api_stop                 = false -> null
    disable_api_termination          = false -> null
    disable_eni_dhcp_options         = false -> null
    get_password_data               = false -> null
    hibernation                      = false -> null
    id                               = "i-0852e9418bd508788" -> null
    instance_initiated_shutdown_behavior = "stop" -> null
    instance_state                  = "running" -> null
    instance_type                   = "t2.micro" -> null
    ipv6_address_count              = 0 -> null
    ipv6_addresses                  = false -> null
    monitoring                      = 0 -> null
    placement_partition_number       = "eni-00449d4418d1cc26" -> null
    primary_network_interface_id    = "eni-00449d4418d1cc26" -> null
    private_ip                      = "172.31.85.72" -> null
    public_dns                       = "ec2-3-82-241-240.compute-1.amazonaws.com" -> null
    public_ip                        = "3.82.241.240" -> null
    secondary_private_ips           = [] -> null
    security_group_ids              = [
        "sg-07971ab5b52bc492",
    ] -> null
    source_dest_check                = true -> null
    subnet_id                        = "subnet-08512513cdca48db" -> null
    tag {
        "Name" = "SPCM-EC2-INSTANCE"
    } -> null
    tag {
        "Name" = "SPCM-EC2-INSTANCE"
    } -> null
    tenancy                           = "default" -> null
    use_eni_for_replace_on_change     = false -> null
    vpc_security_group_ids           = [
        "sg-07971ab5b52bc492",
    ] -> null
    capacity_reservation_specification {
        capacity_reservation_preference = "open"
    } -> null
}
```

Bottom bar: CloudShell, Feedback, © 2024, Amazon Web Services, Inc. or its affiliates., Privacy, Terms, Cookie preferences, ENG IN, 00:40, 25-02-2024.

```
MINGW64:/c/Users/able/Documents/terraform/dir1/terraform-variables
- credit_specification {
  - cpu_credits = "standard" -> null
}
- enclave_options {
  - enabled = false -> null
}
- maintenance_options {
  - auto_recovery = "default" -> null
}
- metadata_options {
  - http_endpoint = "enabled" -> null
  - http_protocol_ipv6 = "disabled" -> null
  - http_put_response_hop_limit = 1 -> null
  - http_tokens = "optional" -> null
  - instance_metadata_tags = "disabled" -> null
}
- private_dns_name_options {
  - enable_private_dns_a_record = false -> null
  - enable_resource_name_dns_aaaa_record = false -> null
  - hostname_type = "ip-name" -> null
}
- root_block_device {
  - delete_on_termination = true -> null
  - device_name = "/dev/sda1" -> null
  - encrypted = false -> null
  - iops = 100 -> null
  - tags = {} -> null
  - throughput = 0 -> null
  - volume_id = "vol-0ea93ad2844cc9cf" -> null
  - volume_size = 8 -> null
  - volume_type = "gp2" -> null
}
}

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

Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes
aws_instance.example: Destroying... [id=i-0852e9418bd508788]
aws_instance.example: Still destroying... [id=i-0852e9418bd508788, 10s elapsed]
aws_instance.example: Still destroying... [id=i-0852e9418bd508788, 20s elapsed]
aws_instance.example: Still destroying... [id=i-0852e9418bd508788, 30s elapsed]
aws_instance.example: Destruction complete after 33s
Destroy complete! Resources: 1 destroyed.

AbhishekPandey@Groot9798 MINGW64 ~/Documents/terraform/dir1/terraform-variables
```

The screenshot shows a Windows desktop environment with a taskbar at the bottom. The taskbar includes icons for File Explorer, Edge browser, File Manager, and other system tools. The system tray shows the date as 25-02-2024 and the time as 00:41. The main window is a web browser displaying the AWS Management Console. The URL bar shows the address: us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances. The left sidebar navigation menu is expanded, showing sections like EC2 Dashboard, EC2 Global View, Events, Console-to-Code, Instances, Images, and Elastic Block Store. The Instances section is currently selected, showing a table titled 'Instances (2) Info'. The table lists two terminated instances: 'SPCM-EC2-INS...' (Instance ID: i-04b1c11def4fb2ba62) and 'SPCM-EC2-INS...' (Instance ID: i-0852e9418bd508788), both of which are t2.micro type and terminated. A modal dialog box titled 'Select an instance' is open in the foreground, indicating that an instance needs to be chosen for further action.

## EXPERIMENT-6

```
1 resource "aws_instance" "example" {
2   ami           = var.ami
3   instance_type = var.instance_type
4 }
5
6 terraform {
7   required_providers {
8     aws = {
9       source = "hashicorp/aws"
10      version = "5.31.0"
11    }
12  }
13}
14
15 provider "aws" {
16   region        = "ap-south-1"
17   access_key    = "AKIA5FTY77WSIB44R75Q"
18   secret_key    = "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"
19 }
20
```

```
Terraform-multiple-tfvars > 🏷 variables.tf > 📁 variable "region" > 📄 default
1  variable "region" [
2    description = "AWS region"
3    default = "ap-south-1"
4  ]
5
6
7  variable "ami" {
8    description = "AMI ID"
9    type = string
10   default = "ami-03f4878755434977f"
11 }
12
13 variable "instance_type" {
14   description = "EC2 Instance Type"
15   default     = "t2.micro"
16 }
```

```
Terraform-multiple-tfvars > 🏷 dev.tfvars > 📄 region
1  region = "ap-south-1"
2  ami = "ami-0d63de463e6604d0a"
3  instance_type = "t2.micro"
```

```

Terraform-multiple-tfvars > 🌈 ops.tfvars > 📁 region
  1   region = "ap-south-1"
  2   ami     = "ami-03f4878755434977f"
  3   instance_type = "t2.large"

```

```

C:\Desktop\DevOps\Sem6\SMCP\Lab Files\TERRAFORM LAB SCRIPTS\Terraform-multiple-tfvars> terraform apply -var-file=dev.tfvars
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
    + arn
    + associate_public_ip_address
    + availability_zone
    + cpu_core_count
    + cpu_threads_per_core
    + disable_api_stop
    + disable_api_termination
    + ebs_optimized
    + get_password_data
    + host_id
    + host_resource_group_arn
    + iam_instance_profile
    + id
    = ami-0d63de463e6604d0a"
    = (known after apply)
    = false
    = (known after apply)
    = (known after apply)
}
```

```

C:\Desktop\DevOps\Sem6\SMCP\Lab Files\TERRAFORM LAB SCRIPTS\Terraform-multiple-tfvars> terraform apply -var-file=ops.tfvars
aws_instance.example: Refreshing state... [id=i-03753a6e6fb28a490]

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

Terraform will perform the following actions:

# aws_instance.example must be replaced
-/+ resource "aws_instance" "example" {
    ~ ami
    ~ arn
    ~ associate_public_ip_address
    ~ availability_zone
    ~ cpu_core_count
    ~ cpu_threads_per_core
    ~ disable_api_stop
    ~ disable_api_termination
    ~ ebs_optimized
    ~ hibernation
    ~ host_id
    ~ host_resource_group_arn
    ~ iam_instance_profile
    ~ id
    ~ instance_initiated_shutdown_behavior
    ~ instance.lifecycle
    = "ami-0d63de463e6604d0a" -> "ami-03f4878755434977f" # forces replacement
    = "arn:aws:ec2:ap-south-1:985418112420:instance/i-03753a6e6fb28a490" -> (known after apply)
    = true -> (known after apply)
    = "ap-south-1b" -> (known after apply)
    = 1 -> (known after apply)
    = 1 -> (known after apply)
    = false -> null
    = (known after apply)
    = (known after apply)
    = (known after apply)
    = "i-03753a6e6fb28a490" -> (known after apply)
    = "stop" -> (known after apply)
    = (known after apply)
}
```

Instances (1) <small>Info</small>		<input type="button" value="C"/>	<input type="button" value="Connect"/>	<input type="button" value="Instance state ▾"/>	<input type="button" value="Actions ▾"/>	<input type="button" value="Launch instances ▾"/>
<input type="text"/> Find Instance by attribute or tag (case-sensitive)						
<input type="button" value="Instance state = running"/> <input type="button" value="X"/>		<input type="button" value="Clear filters"/>				
Any state						
<input type="checkbox"/>	Name ↴	Instance ID	Instance state	Instance type	Status check	Alarm status
<input type="checkbox"/>	i-0336a03bf788061ec	<span>Running</span>	<span>Q Q</span>	t2.large	<span>① Initializing</span>	<input type="button" value="View alarms +"/>

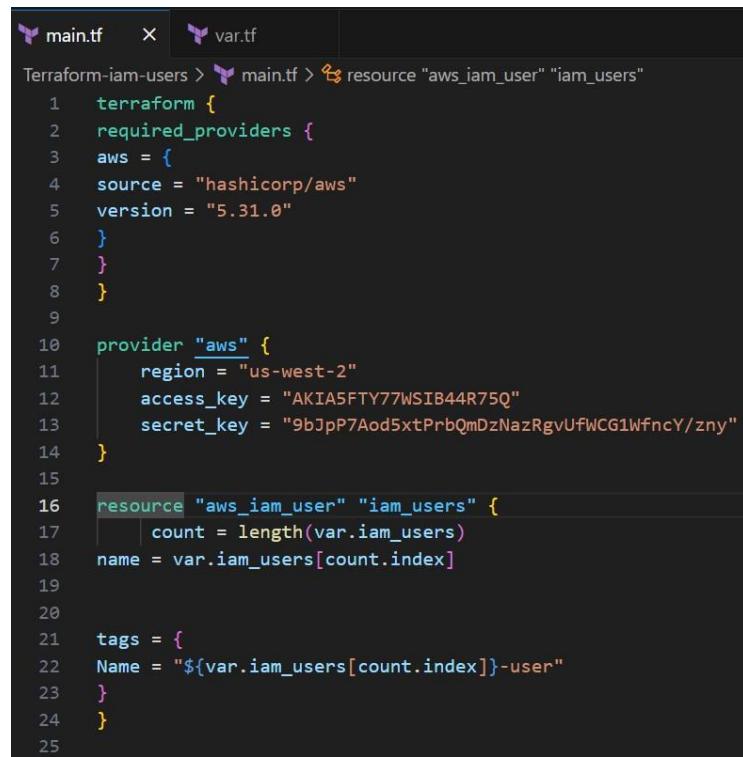
```
C:\Desktop\DevOps\Sem6\SMCP\Lab Files\TERRAFORM LAB SCRIPTS\Terraform-multiple-tfvars> terraform destroy -var-file=ops.tfvars
aws_instance.example: Refreshing state... [id=i-0336a03bf788061ec]

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-03f4878755434977f" -> null
  - arn = "arn:aws:ec2:ap-south-1:1905418112420:instance/i-0336a03bf788061ec" -> null
  - associate_public_ip_address = true -> null
  - availability_zone = "ap-south-1b" -> null
  - cpu_core_count = 2 -> 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-0336a03bf788061ec" -> null
```

## EXPERIMENT-7



The screenshot shows a terminal window with two tabs: "main.tf" and "var.tf". The "main.tf" tab is active and displays the following Terraform code:

```
1  terraform {
2    required_providers {
3      aws = {
4        source  = "hashicorp/aws"
5        version = "5.31.0"
6      }
7    }
8  }
9
10 provider "aws" {
11   region = "us-west-2"
12   access_key = "AKIA5FTY77WSIB44R75Q"
13   secret_key = "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"
14 }
15
16 resource "aws_iam_user" "iam_users" {
17   count = length(var.iam_users)
18   name = var.iam_users[count.index]
19
20
21   tags = {
22     Name = "${var.iam_users[count.index]}-user"
23   }
24 }
25
```

```

main.tf          var.tf
Terraform-iam-users > var.tf > variable "iam_users" > [?] type
1   variable "iam_users" [
2     |   type    = list(string)
3     |   default = ["user3", "user4", "user5"]
4   ]
5
PS C:\Desktop\DevOps\Sem6\SMCP\Lab Files\TERRAFORM LAB SCRIPTS\Terraform-iam-users> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.31.0"...
- Installing hashicorp/aws v5.31.0...
- Installed hashicorp/aws v5.31.0 (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!
PS C:\Desktop\DevOps\Sem6\SMCP\Lab Files\TERRAFORM LAB SCRIPTS\Terraform-iam-users> terraform apply
Terraform used the selected providers to generate the following execution plan. Resource actions are
+ create

Terraform will perform the following actions:

# aws_iam_user.iam_users[0] will be created
+ resource "aws_iam_user" "iam_users" {
    + arn          = (known after apply)
    + force_destroy = false
    + id          = (known after apply)
    + name        = "user3"
    + path        = "/"
    + tags        = {
}

```

IAM > Users

Users (5) <a href="#">Info</a>						
An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.						
<input type="checkbox"/>	User name	Path	Groups	Last activity	MFA	Pass
<input type="checkbox"/>	user1	/	0	26 days ago	-	20
<input type="checkbox"/>	user2	/	0	25 minutes ago	-	3
<input type="checkbox"/>	user3	/	0	-	-	-
<input type="checkbox"/>	user4	/	0	-	-	-
<input type="checkbox"/>	user5	/	0	-	-	-

IAM > Users

**Users (5) Info**

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

	User name	Path	Group	Last activity	MFA	Password last changed
<input type="checkbox"/>	<a href="#">user1</a>	/	0	26 days ago	-	26 days ago
<input type="checkbox"/>	<a href="#">user2</a>	/	0	25 minutes ago	-	3 days ago
<input type="checkbox"/>	<a href="#">user3</a>	/	0	-	-	-
<input type="checkbox"/>	<a href="#">user4</a>	/	0	-	-	-
<input type="checkbox"/>	<a href="#">user5</a>	/	0	-	-	-

```
PS C:\Desktop\DevOps\Sem6\SMCP\Lab Files\TERRAFORM LAB SCRIPTS\Terraform-iam-users> terraform destroy
aws_iam_user.iam_users[2]: Refreshing state... [id=user5]
aws_iam_user.iam_users[1]: Refreshing state... [id=user4]
aws_iam_user.iam_users[0]: Refreshing state... [id=user3]

Terraform used the selected providers to generate the following execution plan. Resource actions are in bold.
- destroy

Terraform will perform the following actions:

# aws_iam_user.iam_users[0] will be destroyed
- resource "aws_iam_user" "iam_users" {
    - arn          = "arn:aws:iam::905418112420:user/user3" -> null
    - force_destroy = false -> null
    - id          = "user3" -> null
    - name        = "user3" -> null
    - path        = "/" -> null
}
```

## EXPERIMENT-8

shivavatsal09hivas-MacBook-Air ~ % cd Documents/Amazon AWS/terraform-demo

shivavatsal09hivas-MacBook-Air ~ % cd Documents/Amazon AWS/terraform-demo

shivavatsal09hivas-MacBook-Air terraform % cd dir1

shivavatsal09hivas-MacBook-Air dir1 % cd terraform-vpc

shivavatsal09hivas-MacBook-Air terraform-vpc % terraform init

shivavatsal09hivas-MacBook-Air terraform-vpc % vi vpc.tf

shivavatsal09hivas-MacBook-Air terraform-vpc % terraform init

```
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.35.0

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,
run this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.

shivavatsal09hivas-MacBook-Air terraform-vpc % terraform apply

[Error: Retrieving AWS account details: validating provider credentials: retrieving caller identity from STS: operation error STS: GetCallerIdentity, https response error StatusCode: 400, RequestID: 4cb2-f6ed-41c1-94d0-0b2c24a85f00, api error IncompleteSignature: 'IAm' not a valid key/value pair (missing equal-sign) in Authorization header: 'AWS4-HMAC-SHA256 Credential/your IAM access key/29244207/gp-southeast-2/sts/aws4_request, SignedHeaders=amz-sdk-invocation-id;amz-sdk-request;content-length;content-type;host;x-amz-date, Signatures=f6ca32272a2c44ddcBf8894a84bb8288dcba8916ee1a09978c2597f3b3f18d03'.

with provider["registry.terraform.io/hashicorp/aws"];
on main.tf line 1, in provider "aws":
  1: provider "aws" {
```

shivavatsal09hivas-MacBook-Air terraform-vpc % vi main.tf

shivavatsal09hivas-MacBook-Air terraform-vpc % terraform init

```
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.35.0

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,
run this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.

shivavatsal09hivas-MacBook-Air terraform-vpc % 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_internet_gateway.gfg-gw will be created
+ resource "aws_internet_gateway" "gfg-gw" {
    + arn          = (known after apply)
    + id          = (known after apply)
    + owner_id    = (known after apply)
    + tags        = [
        + "Name" = "gfg-IO"
      ]
    + tags_all     = [
        + "Name" = "gfg-IO"
      ]
    + vpc_id       = (known after apply)
}

# aws_route_table.gfg-rt will be created
+ resource "aws_route_table" "gfg-rt" {
    + arn          = (known after apply)
    + id          = (known after apply)
    + owner_id    = (known after apply)
    + propagating_vlans = (known after apply)
    + route {
        + carrier_gateway_id      = ""
        + cidr_block              = "0.0.0.0/0"
        + destination_arn          = ""
        + destination_prefix_list_id = ""
        + egress_only_gateway_id   = ""
        + gateway_id               = (known after apply)
        + ipv6_cidr_block          = ""
        + local_gateway_id         = ""
        + nat_gateway_id           = ""
        + network_interface_id    = ""
        + transit_gateway_id       = ""
        + vpc_peering_connection_id = ""
      },
    + tags        = [
        + "Name" = "GFG-Route-Table"
      ]
    + tags_all     = [
        + "Name" = "GFG-Route-Table"
      ]
    + vpc_id       = (known after apply)
}

# aws_route_table_association.gfg-rta will be created
+ resource "aws_route_table_association" "gfg-rta" {
    + id          = (known after apply)
    + route_table_id = (known after apply)
}
```

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, run this command to reinitialize your working directory. If you forget, other commands will detect it and remind you to do so if necessary.

shivavatsal09hivas-MacBook-Air terraform-vpc % 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\_internet\_gateway.gfg-gw will be created
- + resource "aws\_internet\_gateway" "gfg-gw" {
  - + arn = (known after apply)
  - + id = (known after apply)
  - + owner\_id = (known after apply)
  - + tags = [
    - + "Name" = "gfg-IO"
  - + tags\_all = [
    - + "Name" = "gfg-IO"
  - + vpc\_id = (known after apply)

- # aws\_route\_table.gfg-rt will be created
- + resource "aws\_route\_table" "gfg-rt" {
  - + arn = (known after apply)
  - + id = (known after apply)
  - + owner\_id = (known after apply)
  - + propagating\_vlans = (known after apply)
  - + route {
    - + carrier\_gateway\_id = ""
    - + cidr\_block = "0.0.0.0/0"
    - + destination\_arn = ""
    - + destination\_prefix\_list\_id = ""
    - + egress\_only\_gateway\_id = ""
    - + gateway\_id = (known after apply)
    - + ipv6\_cidr\_block = ""
    - + local\_gateway\_id = ""
    - + nat\_gateway\_id = ""
    - + network\_interface\_id = ""
    - + transit\_gateway\_id = ""
    - + vpc\_peering\_connection\_id = ""
  - + tags = [
    - + "Name" = "GFG-Route-Table"
  - + tags\_all = [
    - + "Name" = "GFG-Route-Table"
  - + vpc\_id = (known after apply)

- # aws\_route\_table\_association.gfg-rta will be created
- + resource "aws\_route\_table\_association" "gfg-rta" {
  - + id = (known after apply)
  - + route\_table\_id = (known after apply)

```

}

Plan: 9 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_vpc.vpc: Creating...
aws_vpc.my_vpc: Creating...
aws_subnet.vpc: Creating complete after 6s [id=vpc-06af9331db66e20fe]
aws_internet_gateway.vfg-igw: Creating...
aws_subnet.vfg-subnet: Creating...
aws_route_table.vfg-rtb: Creating...
aws_subnet.vfg-subnet: Creation complete after 3s [id=subnet-08357d5bf2c238459d]
aws_internet_gateway.vfg-igw: Creation complete after 3s [id=igw-05b77f96cd6dd967]
aws_vpc.my_vpc: Still creating... [10s elapsed]
aws_route_table.vfg-rtb: Creation complete after 3s [id=rtb-0d6b1a10ba148ba73]
aws_subnet.vfg-subnet: Refreshing state... [id=subnet-08357d5bf2c238459d]
aws_security_group.vfg-sg: Creation complete after 4s [id=sg-0bbcc3c36d6696973]
aws_route_table_association.vfg-rtas: Creation complete after 4s [id=rtaasoc-042655538df1f38784]
aws_route_table_association.vfg-rtas: Refreshing state after 10s [id=rtaasoc-042655538df1f38784]
aws_subnet.my_subnet[0]: Creating...
aws_subnet.my_subnet[1]: Creating...
aws_subnet.my_subnet[2]: Creating... [10s elapsed]
aws_subnet.my_subnet[1]: Still creating... [10s elapsed]
aws_subnet.my_subnet[1]: Creation complete after 12s [id=subnet-01c44566beaa7f6a]
aws_subnet.my_subnet[2]: Creation complete after 12s [id=subnet-09c6879ab26423987]

Applying changes: 9 to add, 0 to change, 0 to destroy
tshavatsal@Shivas-MacBook-Air: ~ % terraform destroy
aws_vpc.vpc: Refreshing state... [id=vpc-06af9331db66e20fe]
aws_vpc.my_vpc: Refreshing state... [id=vpc-06af9331db66e20fe]
aws_internet_gateway.vfg-igw: Refreshing state... [id=igw-05b77f96cd6dd967]
aws_subnet.vfg-subnet: Refreshing state... [id=subnet-08357d5bf2c238459d]
aws_route_table.vfg-rtb: Refreshing state... [id=rtb-0d6b1a10ba148ba73]
aws_subnet.my_subnet[0]: Refreshing state... [id=subnet-01c44566beaa7f6a]
aws_subnet.my_subnet[1]: Refreshing state... [id=subnet-09c6879ab26423987]
aws_subnet.my_subnet[2]: Refreshing state... [id=subnet-09c6879ab26423987]

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_internet_gateway.vfg-igw will be destroyed
- resource "aws_internet_gateway" "vfg-igw" {
    - arn = "arn:aws:ec2:ap-southeast-2:1891377313468:internet-gateway/igw-05b77f96cd6dd967" -> null
    - id = "igw-05b77f96cd6dd967" -> null
    - owner_id = "891377333448" -> null
    - tags = {
        - "Name" = "vfg-IG"
    } -> null
    - tag_name = "vfg-IG"
    - vpc_id = "vpc-06af9331db66e20fe" -> null
}

# private_dns_hostname_type_on_launch will be destroyed
- resource "aws_vpc" "my_vpc" {
    - arn = "arn:aws:ec2:ap-southeast-2:1891377313468:vpc/vpc-06af9331db66e20fe" -> null
    - tags = {
        - "Name" = "MySubnet-2"
    } -> null
    - tags_all = {
        - "Name" = "MySubnet-2"
    } -> null
    - vpc_id = "vpc-06af9331db66e20fe" -> null
}

# aws_vpc.vfg-vpc will be destroyed
- resource "aws_vpc" "vfg-vpc" {
    - arn = "arn:aws:ec2:ap-southeast-2:1891377313468:vpc/vpc-06af9331db66e20fe" -> null
    - association_generated_ipv4_cidr_block = "10.0.0.0/16" -> null
    - default_network_acl_id = "acl-0a9f272105e3a3e9" -> null
    - default_route_table_id = "rtb-07a098997817874da" -> null
    - dhcp_options_id = "dopt-0a3cd78ffefc98" -> null
    - enable_dns_hostnames = false -> null
    - enable_dhcp = true -> null
    - enable_network_address_usage_metrics = false -> null
    - id = "vpc-06af9331db66e20fe" -> null
    - instance_tenancy = "default" -> null
    - ipv4_nelemas_length = 0 -> null
    - main_route_table_id = "rtb-07a098997817874da" -> null
    - nat_gateway_id = "nat-091377332448" -> null
    - tags = {
        - "Name" = "vfg-VPC"
    } -> null
    - tags_all = {
        - "Name" = "vfg-VPC"
    } -> null
}

# aws_vpc.my_vpc will be destroyed
- resource "aws_vpc" "my_vpc" {
    - arn = "arn:aws:ec2:ap-southeast-2:1891377313468:vpc/vpc-06af9331db66e20fe" -> null
    - association_generated_ipv4_cidr_block = "10.0.0.0/16" -> null
    - default_network_acl_id = "acl-0b0d682c5e5eb9a91" -> null
    - default_route_table_id = "rtb-07a098997817874da" -> null
    - dhcp_options_id = "dopt-0a3cd78ffefc98" -> null
    - enable_dns_hostnames = true -> null
    - enable_dhcp = true -> null
    - enable_network_address_usage_metrics = false -> null
    - id = "vpc-06af9331db66e20fe" -> null
    - instance_tenancy = "default" -> null
    - ipv4_nelemas_length = 0 -> null
    - main_route_table_id = "rtb-07a098997817874da" -> null
    - nat_gateway_id = "nat-091377332448" -> null
    - tags = {
        - "Name" = "MyVPC"
    } -> null
}

```



## Destroy complete! Resources: 9 destroyed.

tshavatsal@Shivas-MacBook-Air terraform-vpc %

# EXPERIMENT-9

```
shivavatsal@Shivas-Air terraform-rds % cd terraform/ec2-for-each
cd: no such file or directory: /Users/shivavatsal/Documents/ec2-for-each/terraform-rds
shivavatsal@Shivas-Air terraform-rds % cd --
shivavatsal@Shivas-Air ~ % cd Documents
shivavatsal@Shivas-Air Documents % cd terraform
shivavatsal@Shivas-Air terraform % cd terraform/ec2-for-each
cd: no such file or directory: /Users/shivavatsal/Documents/ec2-for-each
shivavatsal@Shivas-Air terraform % cd terraform/ec2-for-each
cd: no such file or directory: terraform/ec2-for-each
shivavatsal@Shivas-Air terraform % cd dir1
shivavatsal@Shivas-Air dir1 % terraform/ec2-for-each % vi main.tf
shivavatsal@Shivas-Air terraform/ec2-for-each % vi main.tf
shivavatsal@Shivas-Air terraform/ec2-for-each % terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.31.0

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,
run this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
shivavatsal@Shivas-Air terraform/ec2-for-each % 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.ec2_instances["instance1"] will be created
+ resource "aws_instance" "ec2_instances" {
  + arn = "arn:aws:ec2:ap-south-1:98edb268a3"
  + 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_ami_termination = (known after apply)
  + ebs_optimized = (known after apply)
  + get_password_data = false
}

+ public_ip = (known after apply)
+ secondary_private_ips = (known after apply)
+ security_groups = (known after apply)
+ source_dest_check = true
+ spot_instance_request_id = (known after apply)
+ subnet_id = (known after apply)
+ tags = {
  + "Name" = "EC2-Instance-instance1"
}
+ tags_all = {
  + "Name" = "EC2-Instance-instance1"
}
+ tenancy = (known after apply)
+ user_data = (known after apply)
+ user_data_base64 = (known after apply)
+ user_data_replace_on_change = false
+ vpc_security_group_ids = (known after apply)

# aws_instance.ec2_instances["instance2"] will be created
+ resource "aws_instance" "ec2_instances" {
  + arn = "arn:aws:ec2:ap-south-1:72b989"
  + 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_ami_termination = (known after apply)
  + ebs_optimized = (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)
  + placement_partition_number = (known after apply)
  + primary_network_interface_id = (known after apply)
  + private_dns = (known after apply)
```

```

+ tenancy          = (known after apply)
+ user_data        = (known after apply)
+ user_data_base64 = (known after apply)
+ user_data_replace_on_change = false
+ vpc_security_group_ids = (known after apply)
}

# aws_instance.ec2_instances["instance3"] will be created
resource "aws_instance" "ec2_instances" {
  ami           = "ami-0dd6857b844e855670"
  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_eni_top      = (known after apply)
  disable_api_termination = (known after apply)
  ebs_optimized       = (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)
  placement_partition_number = (known after apply)
  primary_network_interface_id = (known after apply)
  private_dns        = (known after apply)
  private_ip         = (known after apply)
  public_dns         = (known after apply)
  public_ip          = (known after apply)
  secondary_private_ips = (known after apply)
  security_groups   = (known after apply)
  source_dest_check = true
  spot_instance_request_id = (known after apply)
  subnet_id          = (known after apply)
  tags {
    + "Name" = "EC2-Instance-Instance3"
  }
  tags.all          = (
    + "Name" = "EC2-Instance-Instance3"
  )
}

```



```

+ 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)
+ placement_partition_number = (known after apply)
+ primary_network_interface_id = (known after apply)
+ private_dns              = (known after apply)
+ private_ip                = (known after apply)
+ public_dns                = (known after apply)
+ public_ip                  = (known after apply)
+ secondary_private_ips    = (known after apply)
+ security_groups          = (known after apply)
+ source_dest_check        = true
+ spot_instance_request_id = (known after apply)
+ subnet_id                 = (known after apply)
+ tags {
    + "Name" = "EC2-Instance-Instance3"
  }
  tags.all          = (
    + "Name" = "EC2-Instance-Instance3"
  )
}

```

Plan: 3 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_instance.ec2_instances["instance3": Creating...
aws_instance.ec2_instances["instance2": Creating...
aws_instance.ec2_instances["instance1": Creating...
aws_instance.ec2_instances["instance1": Still creating... [1s elapsed]
aws_instance.ec2_instances["instance1": Still creating... [2s elapsed]
aws_instance.ec2_instances["instance1": Still creating... [3s elapsed]
aws_instance.ec2_instances["instance1": Creation complete after 35s [id=i-0e92cb42914fe6aa3]

```

```

terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.31.0"
    }
  }
  provider "aws" {
    region = "ap-southeast-2"
    access_key = "AKIA47CR2m260V4Y5bG3"
    secret_key = "Xjb2IX0VlhDGPS83Ziam1SwM3ggH53xI9Nx6Nwo"
  }
  variable "instances" {
    description = "Map of EC2 instances with settings"
    default = {
      "instance1" = {
        ami = "ami-07eaeb99edeb268a3"
        instance_type = "t2.micro"
      },
      "instance2" = {
        ami = "ami-04f5097681773b989"
        instance_type = "t2.micro"
      },
      "instance3" = {
        ami = "ami-0dd6857b844e855670"
        instance_type = "t2.micro"
      }
    }
  }
  resource "aws_instance" "ec2_instances" {
    for_each = var.instances
    ami      = var.instances[each.key].ami
    instance_type = var.instances[each.key].instance_type
    tags {
      Name = "EC2-Instance-${each.key}"
    }
  }
}

```

"main.tf" 38L, 763B

Screenshot of the AWS Management Console showing the EC2 Instances page. Three t2.micro instances are listed, all running in us-east-1. A modal window titled "Select an instance" is open, prompting the user to choose which instance to interact with.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
EC2-Instance-i...	i-019778f7f8ad49f58	Running	t2.micro	2/2 checks passed	<a href="#">View alarms</a>	us-east-1a	ec2-54-145-46
EC2-Instance-i...	i-07079b45c239b6974	Running	t2.micro	2/2 checks passed	<a href="#">View alarms</a>	us-east-1a	ec2-3-80-40-1
EC2-Instance-i...	i-092528d3a5611baab	Running	t2.micro	2/2 checks passed	<a href="#">View alarms</a>	us-east-1a	ec2-18-208-21

MINGW64 /c/Users/AbhishekPandey/Documents/terraform/dlr/terraform-ec2-for-each

```

- http_tokens          = "optional" -> null
- instance_metadata_tags = "disabled" -> null
}

- private_dns_name_options {
-   enable_resource_name_dns_a_record = false -> null
-   enable_resource_name_dns_aaaa_record = false -> null
-   hostname_type                   = "ip-name" -> null
}

root_block_device {
- delete_on_termination = true -> null
- device_name           = "/dev/sda1" -> null
- encrypted              = false -> null
- iops                  = 100 -> null
- throughput             = 1 -> null
- throughput             = 0 -> null
- volume_id              = "vvol-0125c8caad8137b63" -> null
- volume_size             = 30 -> null
- volume_type             = "gp2" -> null
}
}

Plan: 0 to add, 0 to change, 3 to destroy.

Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

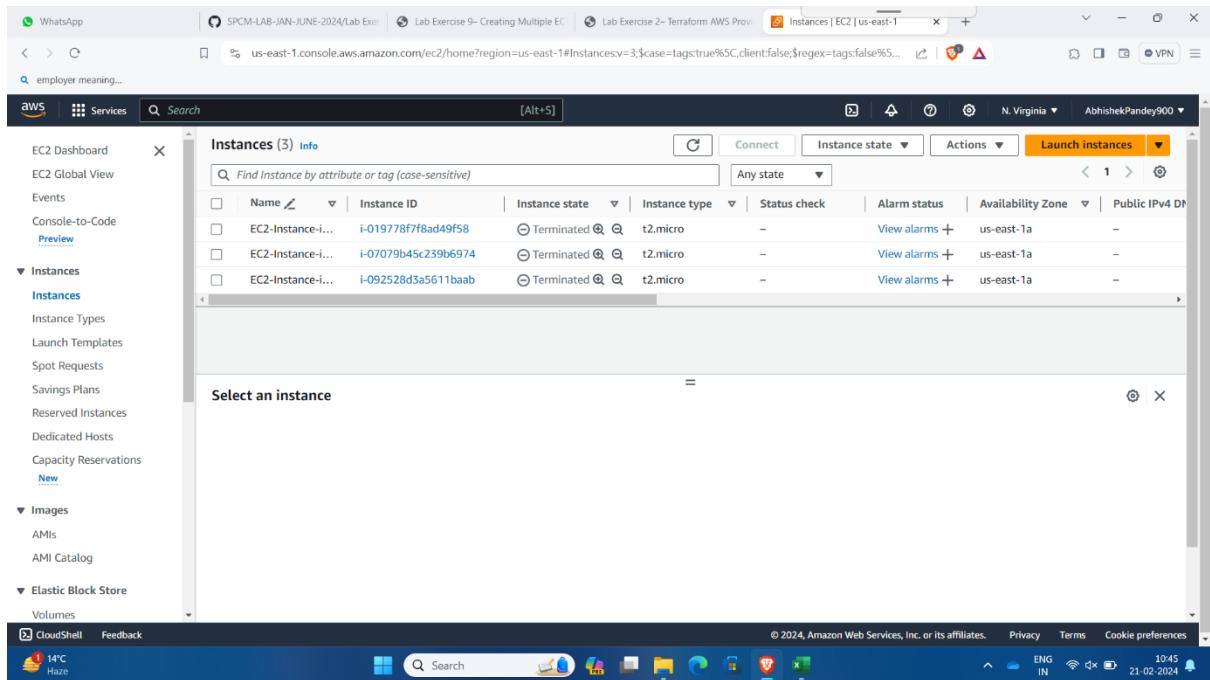
Enter a value: yes

aws_instance.ec2_instances["instance3"] : Destroying... [id=i-07079b45c239b6974]
aws_instance.ec2_instances["instance1"] : Destroying... [id=i-092528d3a5611baab]
aws_instance.ec2_instances["instance2"] : Destroying... [id=i-019778f7f8ad49f58]
aws_instance.ec2_instances["instance3"] : Still destroying... [id=i-07079b45c239b6974, 10s elapsed]
aws_instance.ec2_instances["instance3"] : Still destroying... [id=i-07079b45c239b6974, 10s elapsed]
aws_instance.ec2_instances["instance1"] : Still destroying... [id=i-092528d3a5611baab, 10s elapsed]
aws_instance.ec2_instances["instance1"] : Still destroying... [id=i-092528d3a5611baab, 20s elapsed]
aws_instance.ec2_instances["instance1"] : Still destroying... [id=i-092528d3a5611baab, 20s elapsed]
aws_instance.ec2_instances["instance1"] : Still destroying... [id=i-019778f7f8ad49f58, 20s elapsed]
aws_instance.ec2_instances["instance2"] : Still destroying... [id=i-019778f7f8ad49f58, 30s elapsed]
aws_instance.ec2_instances["instance2"] : Still destroying... [id=i-092528d3a5611baab, 30s elapsed]
aws_instance.ec2_instances["instance2"] : Still destroying... [id=i-07079b45c239b6974, 30s elapsed]
aws_instance.ec2_instances["instance3"] : Still destroying... [id=i-07079b45c239b6974, 30s elapsed]
aws_instance.ec2_instances["instance3"] : Destruction complete after 35s
aws_instance.ec2_instances["instance1"] : Destruction complete after 35s
aws_instance.ec2_instances["instance2"] : Destruction complete after 40s
aws_instance.ec2_instances["instance2"] : Still destroying... [id=i-019778f7f8ad49f58, 40s elapsed]

Destroy complete! Resources: 3 destroyed.

AbhishekPandey@Groot9798 MINGW64 ~/Documents/terraform/dlr/terraform-ec2-for-each
$ |

```



## EXPERIMENT-10

```

shivavatsal@Shivas-Air ~ % cd Documents
shivavatsal@Shivas-Air Documents % cd terraform
shivavatsal@Shivas-Air terraform % mkdir terraform-rds
shivavatsal@Shivas-Air terraform % cd terraform-rds
shivavatsal@Shivas-Air terraform % terraform init
shivavatsal@Shivas-Air terraform-rds % terraform init
Terraform initialized in an empty directory!

The directory has no Terraform configuration files. You may begin working
with Terraform immediately by creating Terraform configuration files.
shivavatsal@Shivas-Air terraform-rds % vi main
shivavatsal@Shivas-Air terraform-rds % vi main.tf
shivavatsal@Shivas-Air terraform-rds % terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws version matching "0.31.0"...
- Installed hashicorp/aws v0.31.0 (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, run
"terraform init" to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
shivavatsal@Shivas-Air terraform-rds % 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_db_instance.My-RDS will be created
+ resource "aws_db_instance" "My-RDS" {
    + address = (known after apply)
    + allocated_storage = 10
    + apply_immediately = false
    + dns = (known after apply)
}

```

```
+ replicas          = (known after apply)
+ resource_id       = (known after apply)
+ skip_final_snapshot = true
+ snapshot_identifier = (known after apply)
+ state             = (known after apply)
+ storage_throughput = (known after apply)
+ storage_type      = (known after apply)
+ tags_all          = (known after apply)
+ timezone          = (known after apply)
+ vcpu              = (known after apply)
+ vpc_security_group_ids = (known after apply)
}
```

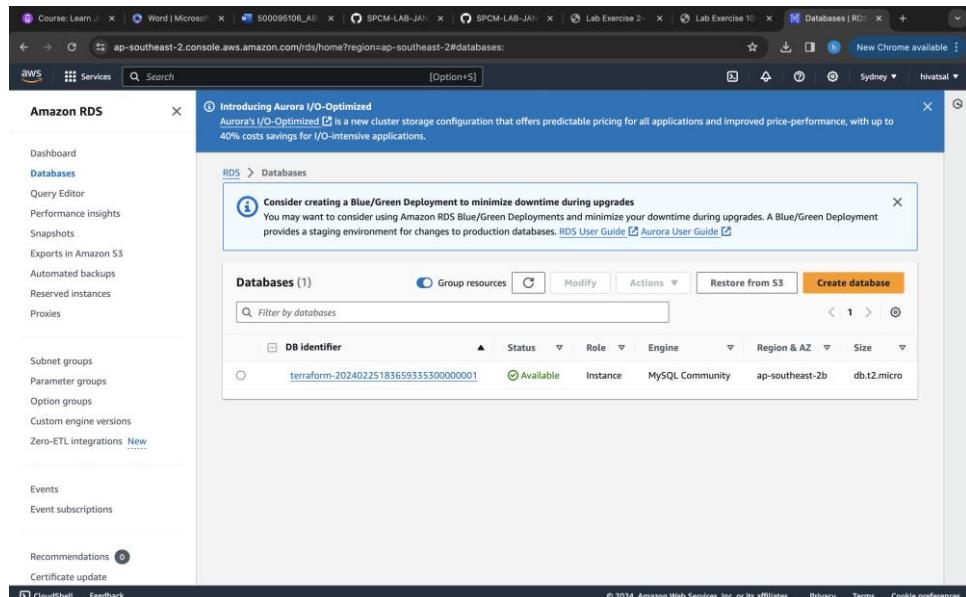
```
Plan: 1 to add, 0 to change, 0 to destroy.  
  
Do you want to perform these actions?  
Terraform will perform the actions described above.  
Only the resources listed above will be changed.
```

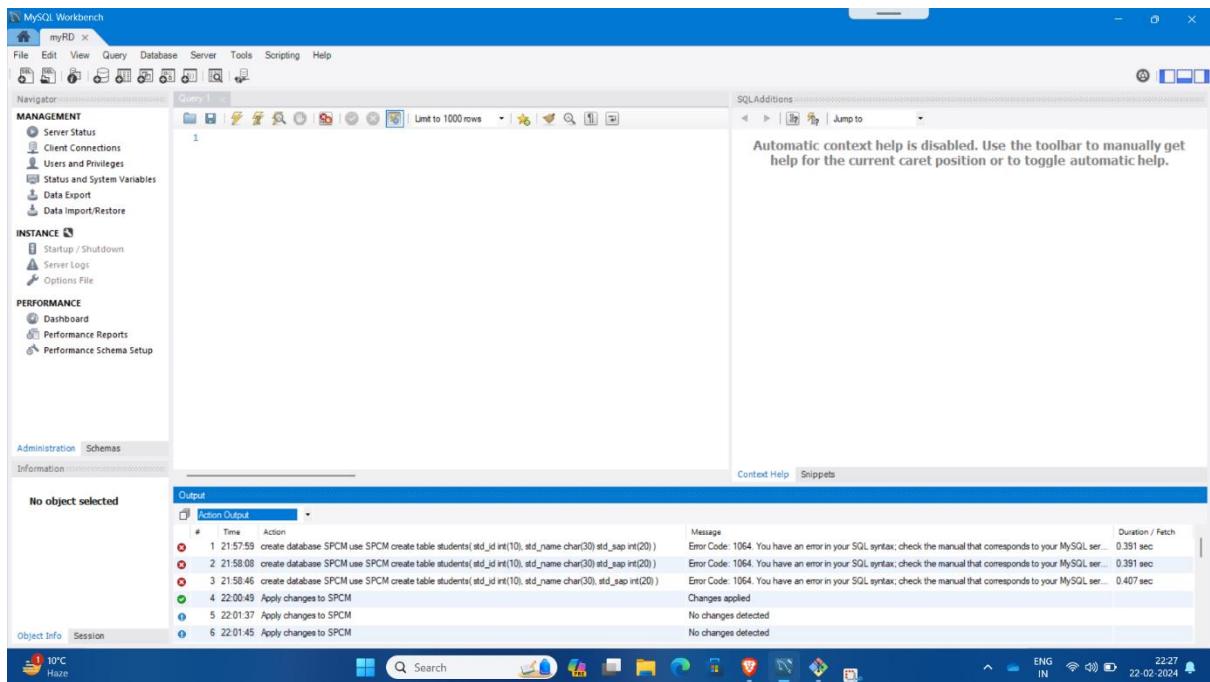
```
Only 'yes' will be accepted to approve.  
Enter a value: yes
```

www.dbsinstances.com

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

```
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.  
shivavatsal@Shivas-Air terraform-rds %
```





The screenshot shows the AWS RDS console. The left sidebar includes options like Dashboard, Databases (selected), Query Editor, Performance insights, Snapshots, Exports in Amazon S3, Automated backups, Reserved instances, Proxies, Subnet groups, Parameter groups, Option groups, Custom engine versions, Zero-ETL integrations, Events, and Event subscriptions. The main area shows the 'Databases' section for the 'myd' database. The 'Summary' tab displays basic information:

DB identifier	Status	Role	Engine	Recommendations
myd	Available	Instance	MySQL Community	
CPU	Class db.t3.micro	Current activity	Region & AZ us-east-1f	

The 'Connectivity & security' tab shows details about the endpoint and networking:

Endpoint & port	Networking	Security
Endpoint myd.cp0gaoac2nl3.us-east-1.rds.amazonaws.com	Availability Zone us-east-1f	VPC security groups default (sg-07971ab5b528bc492)
	VPC	Active