#Name :- Anirudha Kurhade

#Prn:- 22120150

Assignment 5

Aim: - Write IaC using terraform to create EC2 machine on AWS or azure or google cloud.

Theory: -

What is Terraform

Terraform is an open-source infrastructure as code (IaC) tool developed by HashiCorp. It allows you to define, manage, and automate your cloud infrastructure in a declarative way, meaning you describe the desired end state of your infrastructure and Terraform handles the details of making the changes to get you there.

With Terraform, you can provision and manage infrastructure resources across a variety of cloud providers, including AWS, Azure, Google Cloud Platform, and many others. This can include creating and configuring virtual machines, load balancers, databases, and other resources.

One of the key benefits of Terraform is that it allows you to define your infrastructure in a version-controlled, text-based format, which enables you to easily collaborate with others, track changes over time, and reproduce your infrastructure across multiple environments.

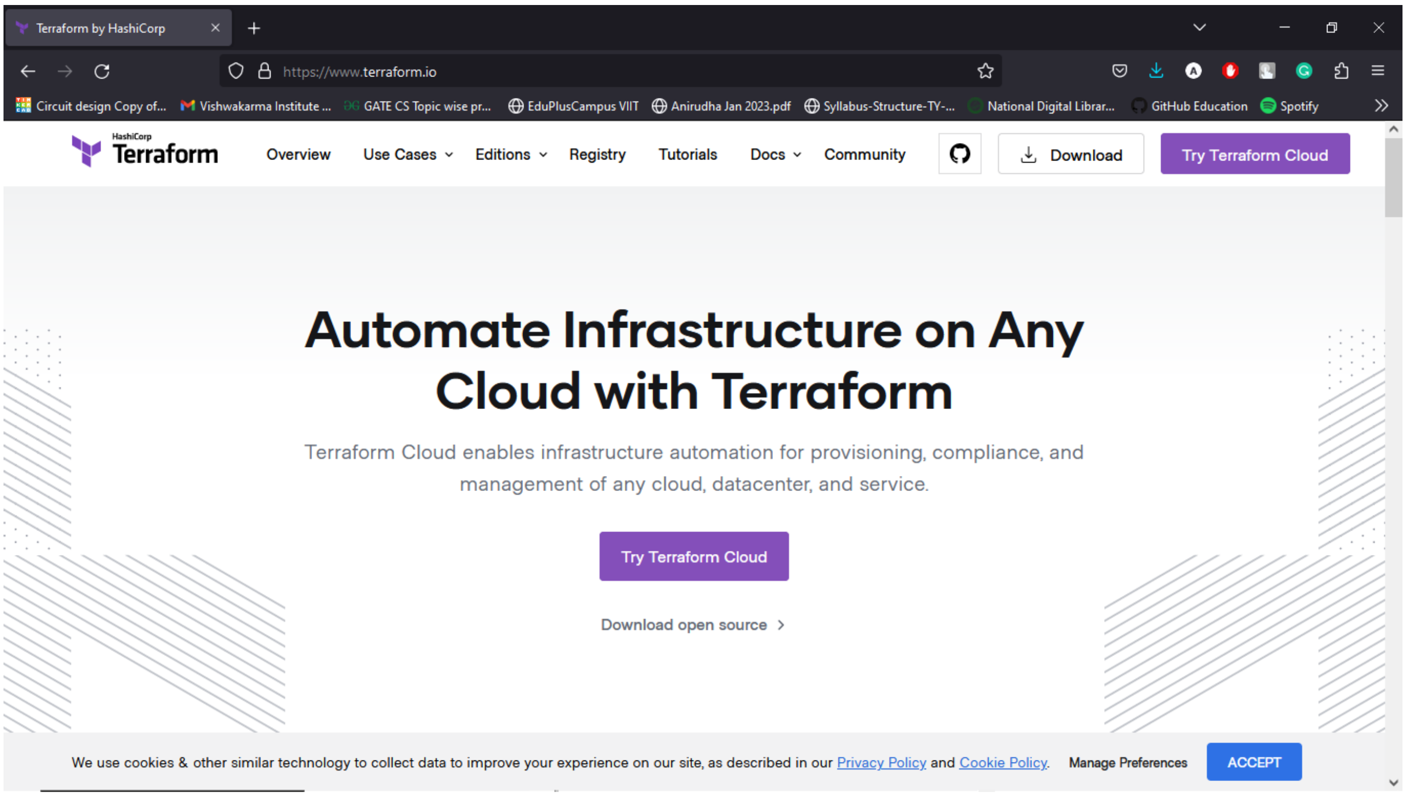
Overall, Terraform provides a powerful and flexible way to manage infrastructure as code, making it a popular choice for organizations that need to manage complex cloud environments at scale.

Steps to install and configure Terraform:-

1. Download the Terraform binary: Go to the Terraform website (https://www.terraform.io/downloads.html) and download the appropriate binary for your operating system.
2. Install the Terraform binary: Once you have downloaded the binary, extract the contents of the archive to a directory on your system. Then, add the directory to your system's PATH environment variable so that you can run the Terraform command from anywhere in your terminal.
3. Verify the installation: To verify that Terraform is installed correctly, open a new terminal window and run the following command: terraform --version. This should display the current version of Terraform that you have installed.
4. Configure the cloud provider credentials: Terraform requires credentials for the cloud provider you are using to manage your infrastructure. For example, if you are using AWS, you will need to provide an access key and secret key. You can set these credentials as environment variables or in a configuration file.

5. Create a Terraform configuration file: Next, create a Terraform configuration file (usually named main.tf) that defines the resources you want to provision. This file will contain the infrastructure code that Terraform will use to create and manage your resources.
6. Initialize the Terraform project: Once you have created your configuration file, navigate to the directory that contains it and run the command terraform init. This will initialize the Terraform project and download any necessary plugins.
7. Apply the Terraform configuration: Finally, run the command terraform apply to apply your configuration and provision your resources. Terraform will analyze your configuration file and make any necessary changes to your cloud infrastructure to bring it into the desired state.

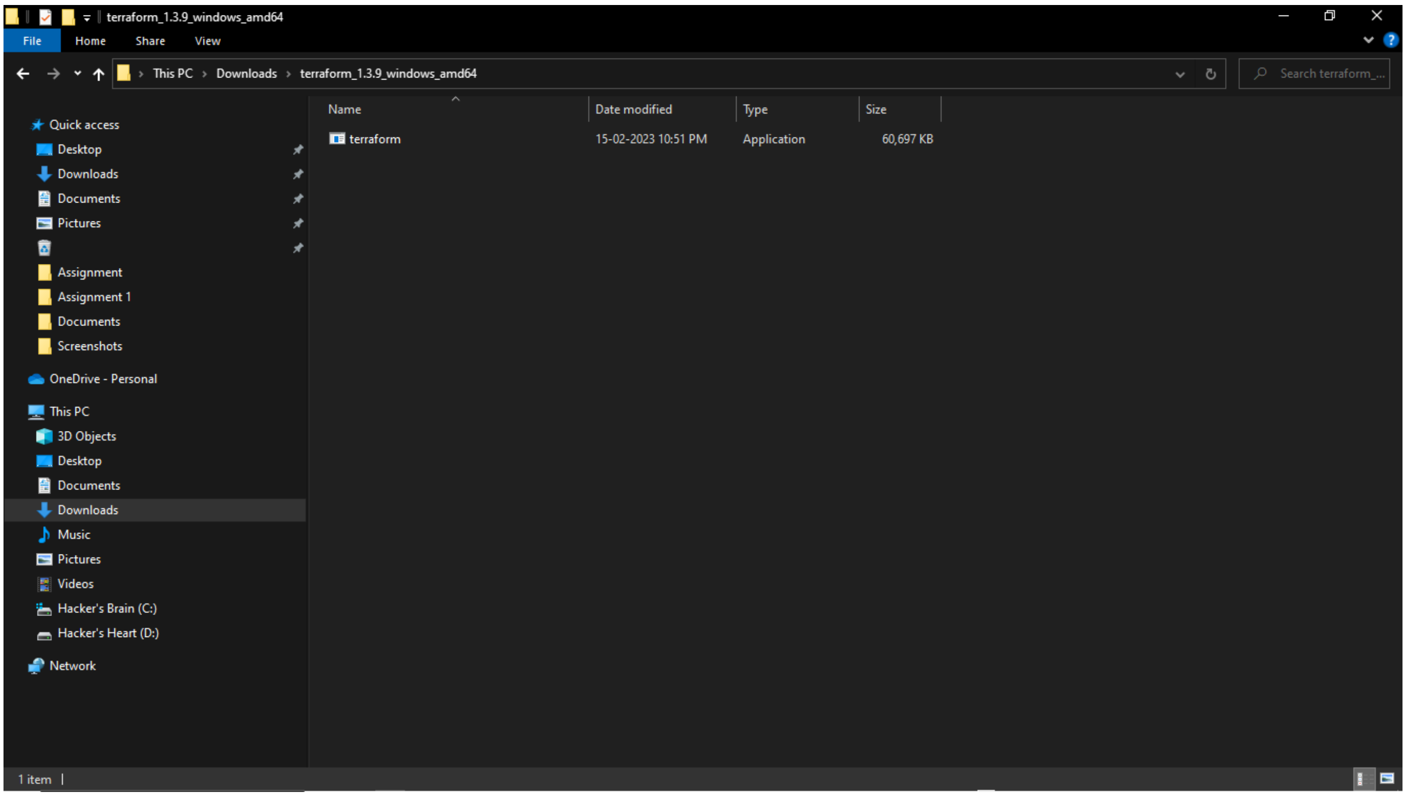
Steps Output: -



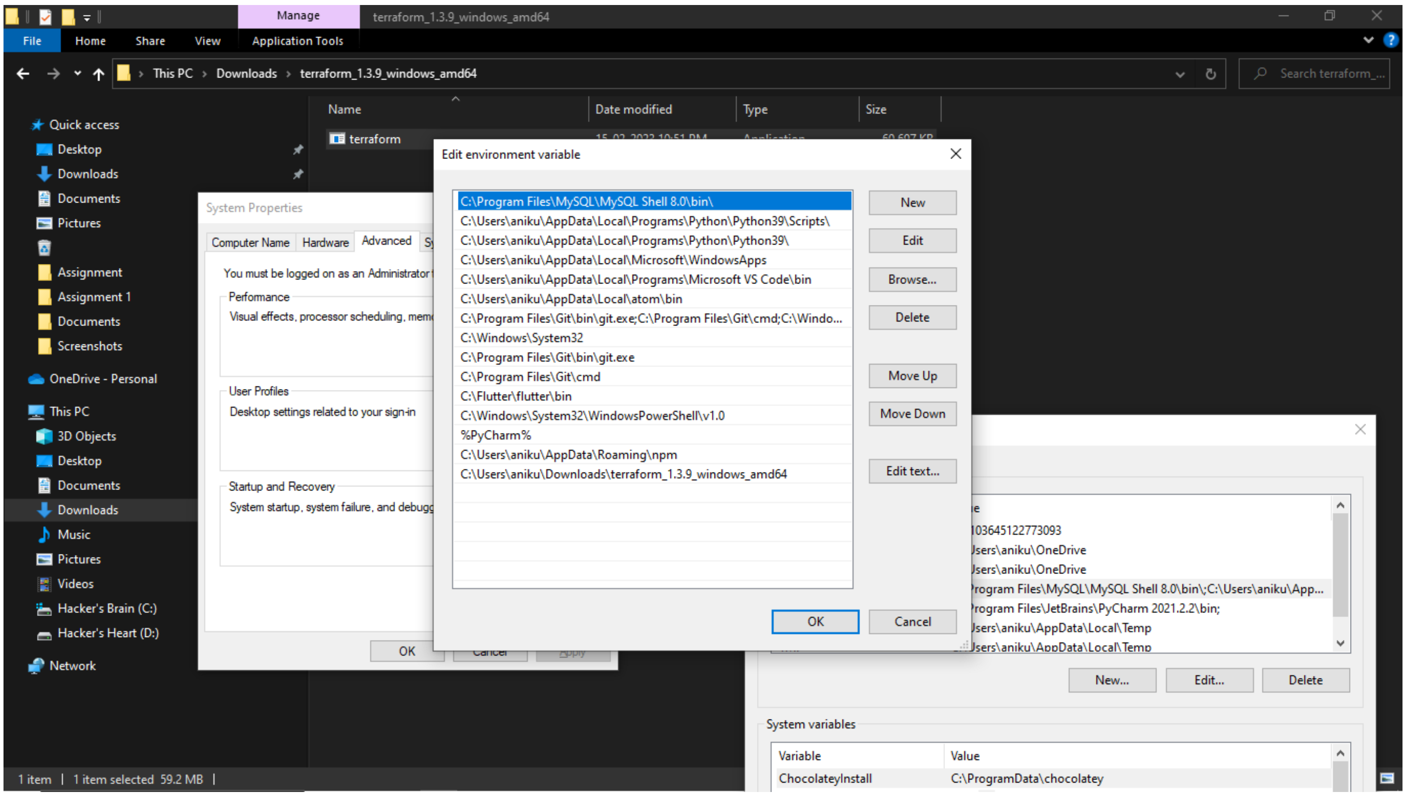
Automate Infrastructure on Any Cloud with Terraform

Terraform Cloud enables infrastructure automation for provisioning, compliance, and management of any cloud, datacenter, and service.

We use cookies & other similar technology to collect data to improve your experience on our site, as described in our Privacy Policy and Cookie Policy.



Name	Date modified	Type	Size
terraform	15-02-2023 10:51 PM	Application	60,697 KB

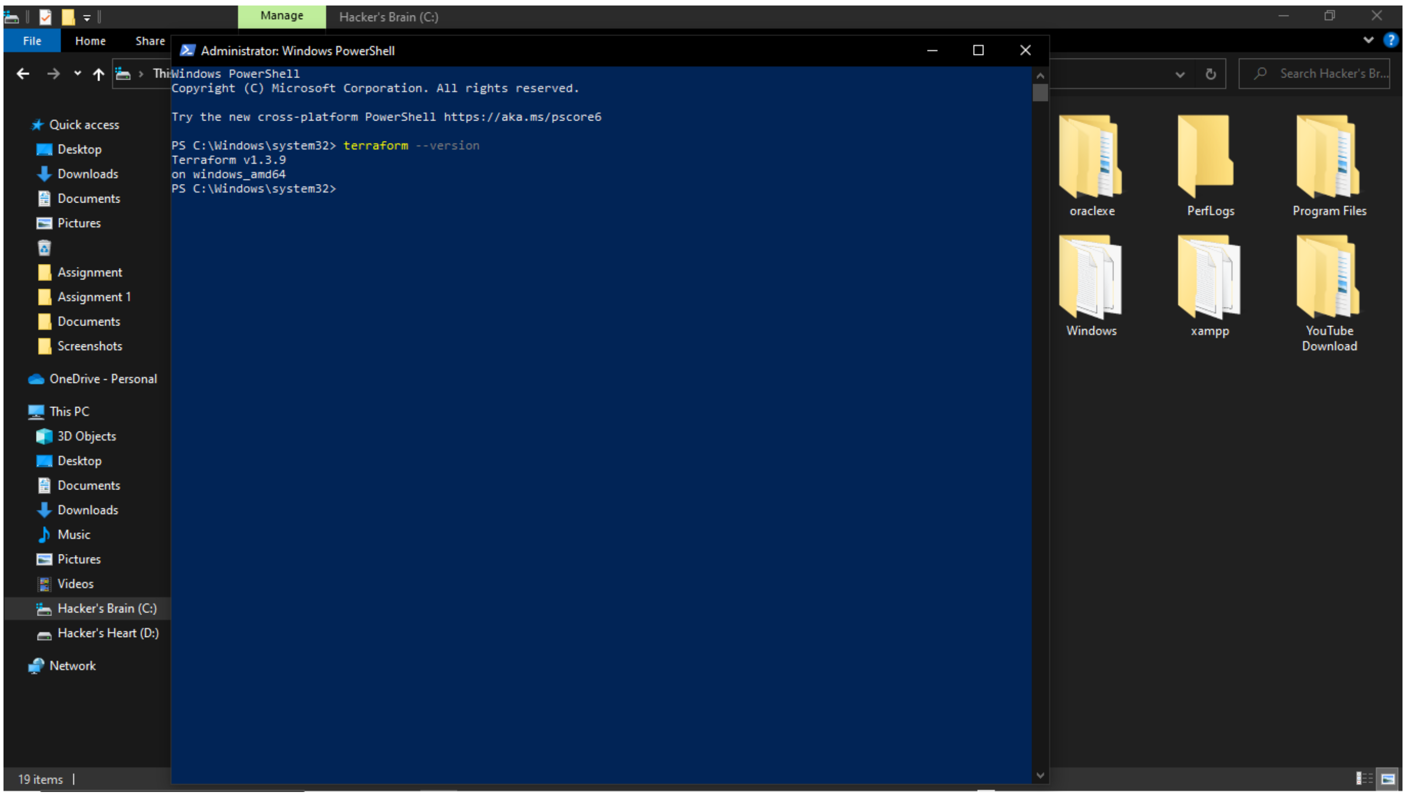


Edit environment variable

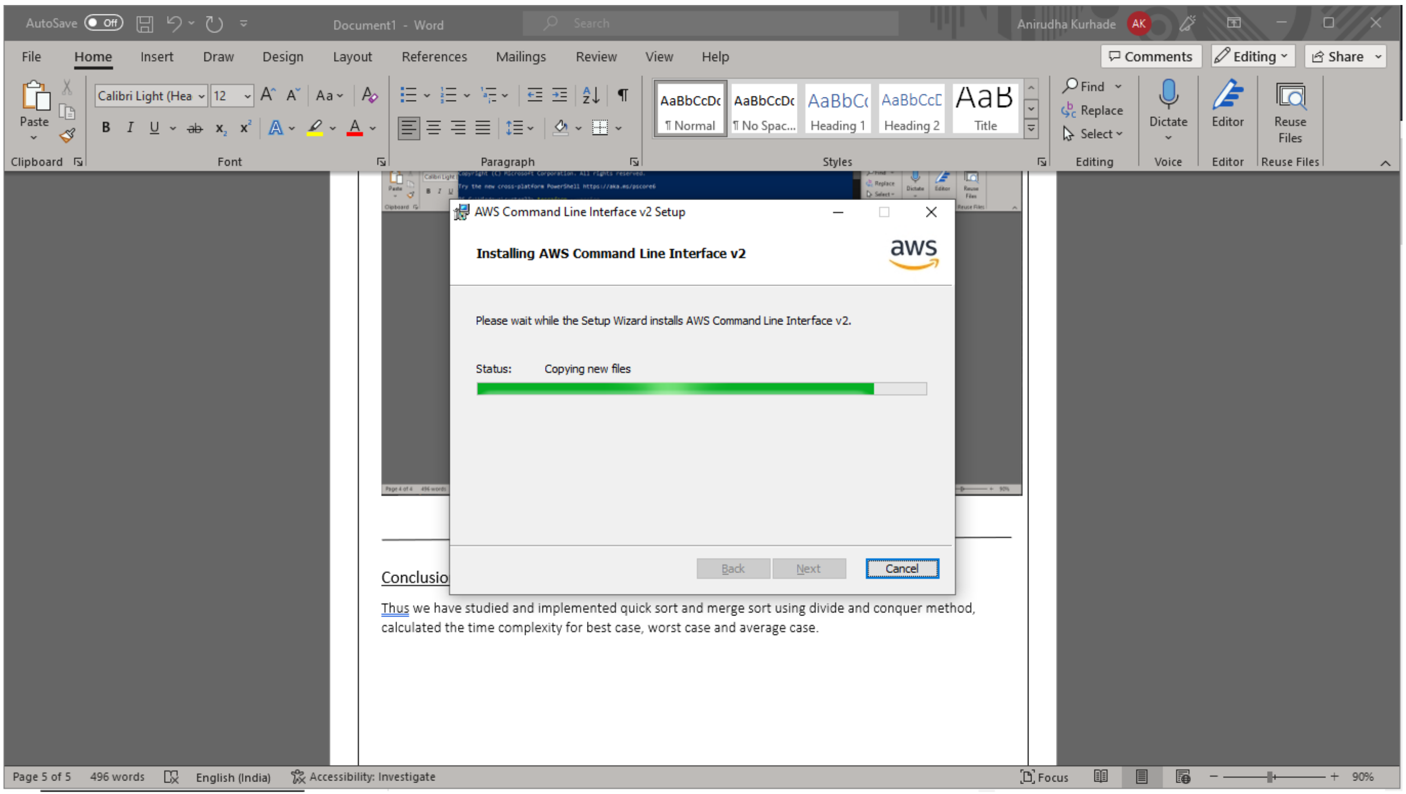
```
C:\Program Files\MySQL\MySQL Shell 8.0\bin\
C:\Users\aniku\AppData\Local\Programs\Python\Python39\Scripts\
C:\Users\aniku\AppData\Local\Programs\Python\Python39\
C:\Users\aniku\AppData\Local\Microsoft\WindowsApps
C:\Users\aniku\AppData\Local\Programs\Microsoft VS Code\bin
C:\Users\aniku\AppData\Local\atom\bin
C:\Program Files\Git\bin\git.exe;C:\Program Files\Git\cmd;C:\Windo...
C:\Windows\System32
C:\Program Files\Git\bin\git.exe
C:\Program Files\Git\cmd
C:\Flutter\flutter\bin
C:\Windows\System32\WindowsPowerShell\v1.0
%PyCharm%
C:\Users\aniku\AppData\Roaming\npm
C:\Users\aniku\Downloads\terraform_1.3.9_windows_amd64
```

Variable	Value
ChocolateyInstall	C:\ProgramData\chocolatey

1 item | 1 item selected 59.2 MB



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Windows\system32> terraform --version
Terraform v1.3.9
on windows_amd64
PS C:\Windows\system32>
```


AWS Command Line Interface v2 Setup

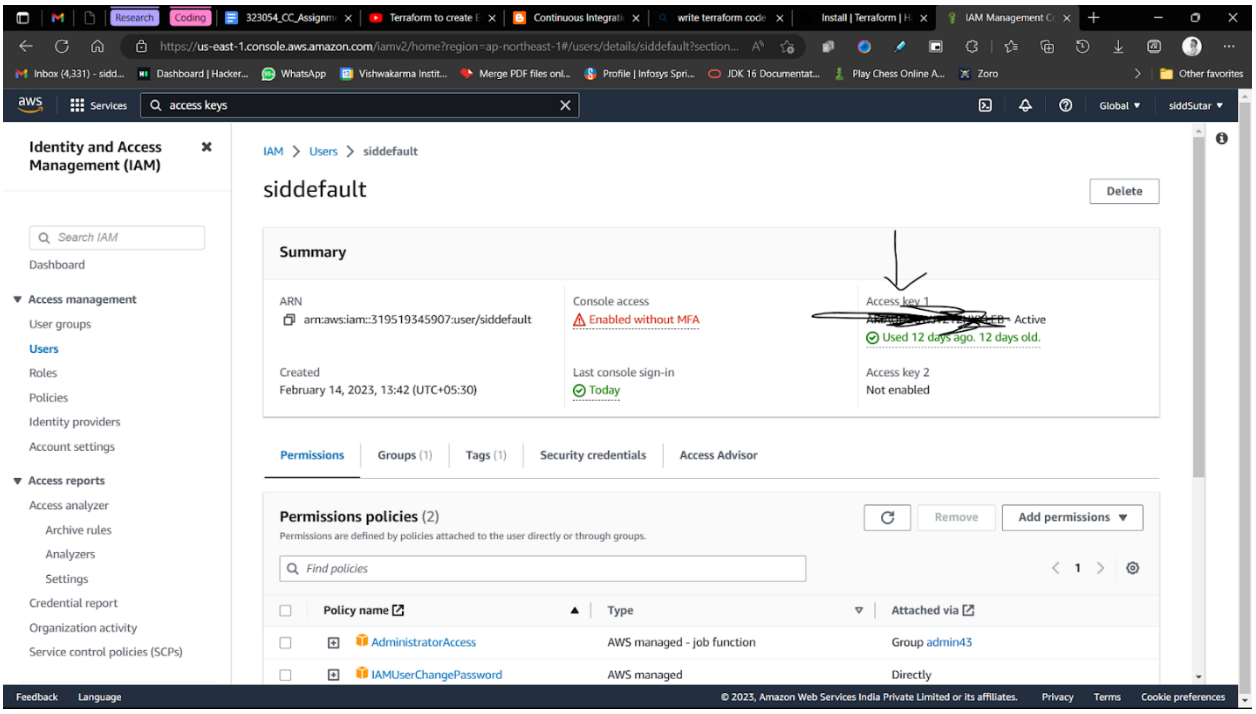
Installing AWS Command Line Interface v2

Please wait while the Setup Wizard installs AWS Command Line Interface v2.

Status: Copying new files

Conclusio

Thus we have studied and implemented quick sort and merge sort using divide and conquer method, calculated the time complexity for best case, worst case and average case.


https://us-east-1.console.aws.amazon.com/iamv2/home?region=ap-northeast-1#/users/details/siddefault?section...

Identity and Access Management (IAM)

IAM > Users > siddefault

siddefault

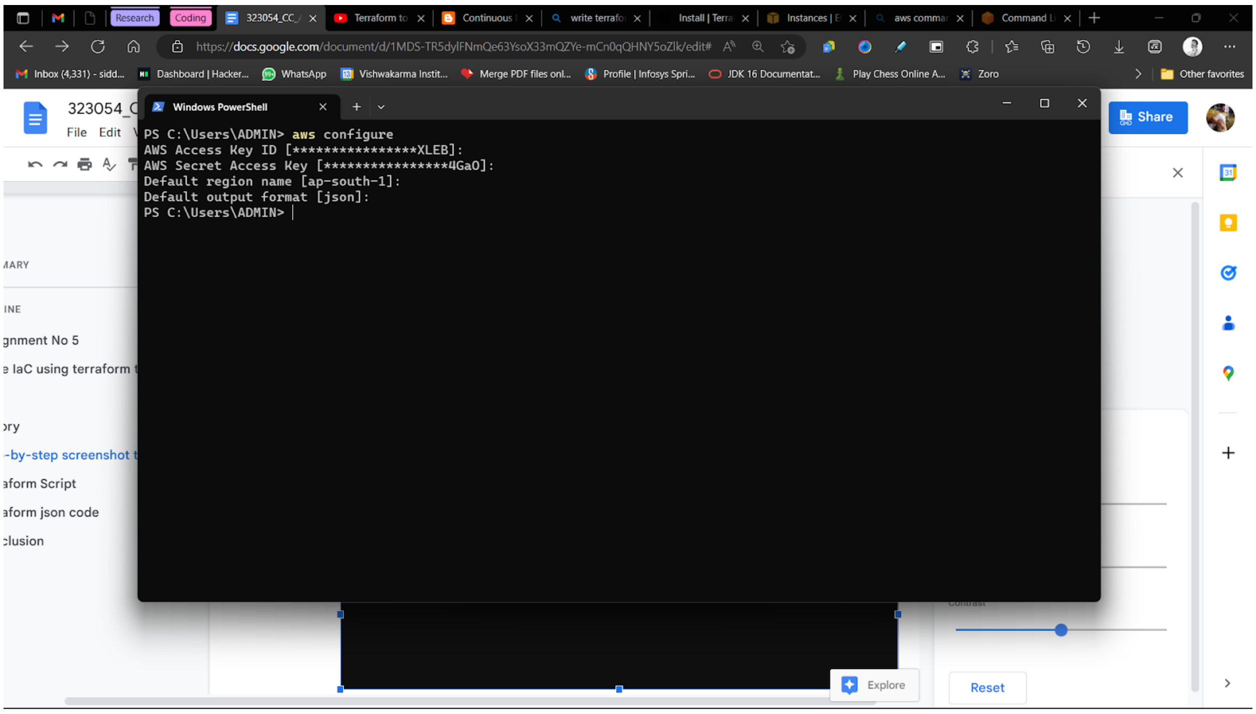
Summary

ARN	Console access	Access key 1
arn:aws:iam::319519345907:user/siddefault	Enabled without MFA	[redacted] - Active
		Used 12 days ago. 12 days old.
Created	**Last console sign-in**	**Access key 2**
February 14, 2023, 13:42 (UTC+05:30)	Today	Not enabled

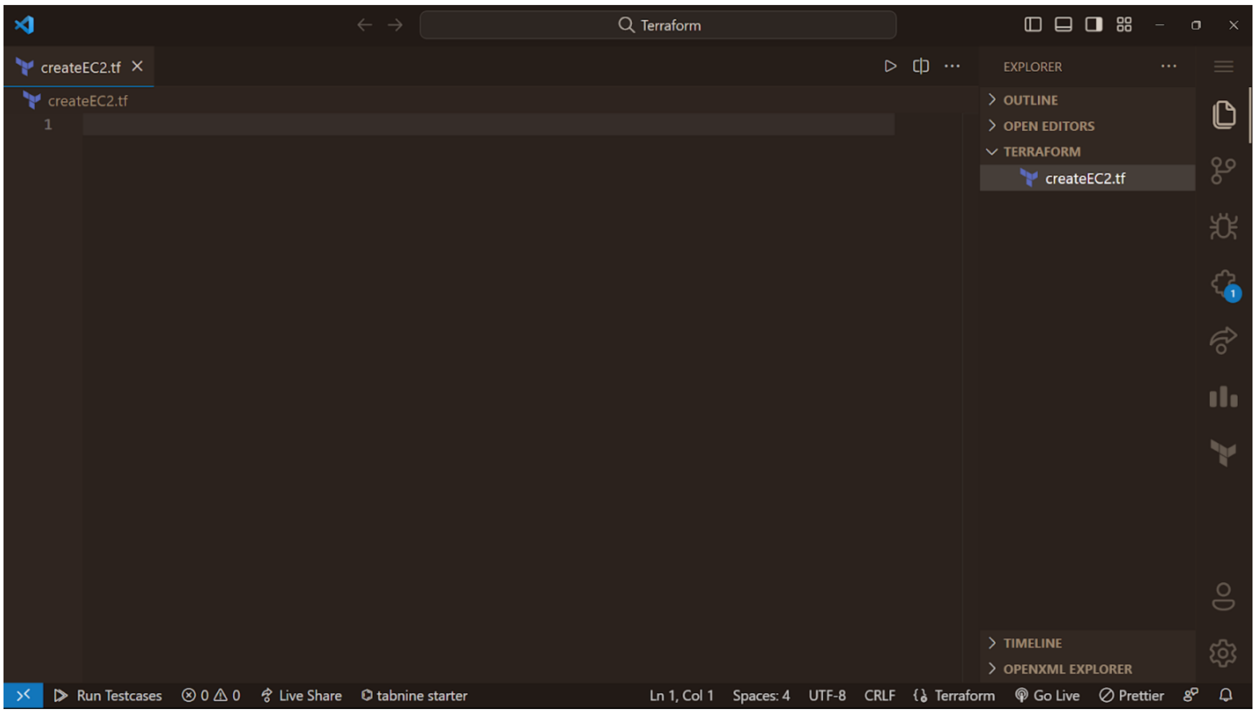
Permissions policies (2)

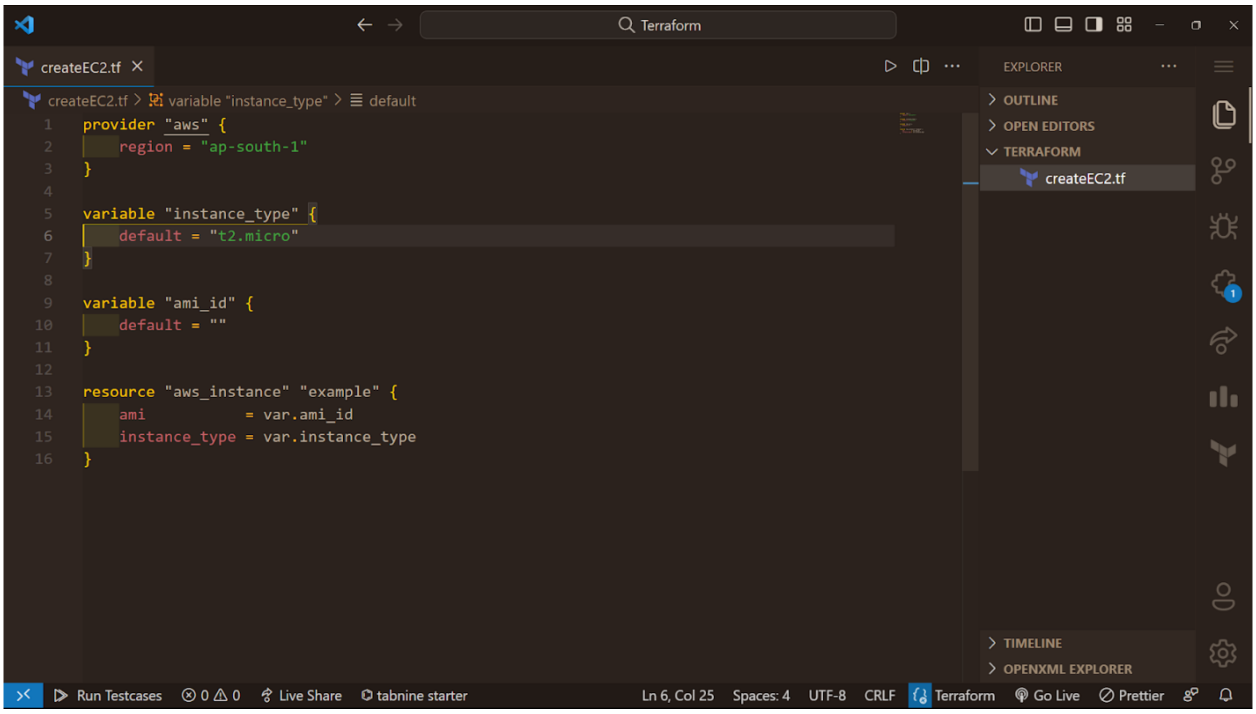
Permissions are defined by policies attached to the user directly or through groups.

Policy name	Type	Attached via
AdministratorAccess	AWS managed - job function	Group admin43
IAMUserChangePassword	AWS managed	Directly

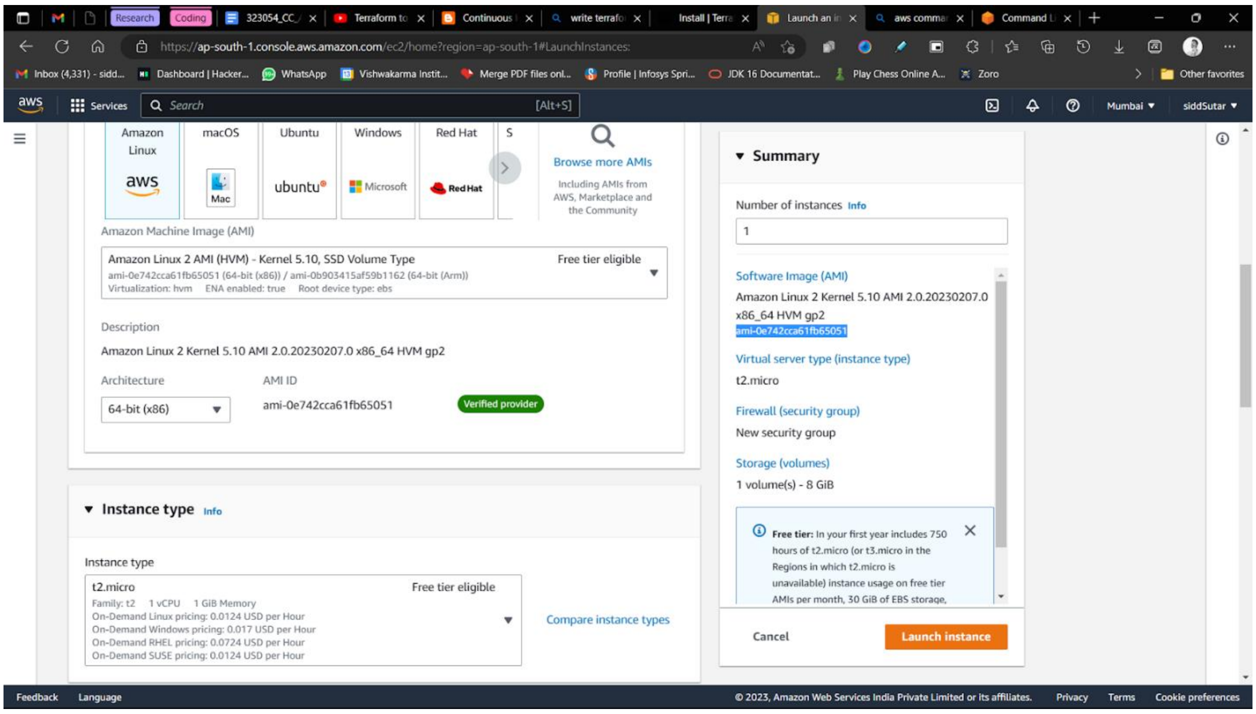


```
PS C:\Users\ADMIN> aws configure
AWS Access Key ID [****************XLEB]:
AWS Secret Access Key [****************4Ga0]:
Default region name [ap-south-1]:
Default output format [json]:
PS C:\Users\ADMIN> |
```





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

variable "instance_type" {
    default = "t2.micro"
}

variable "ami_id" {
    default = ""
}

resource "aws_instance" "example" {
    ami           = var.ami_id
    instance_type = var.instance_type
}
```



https://ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#LaunchInstances:
Services
Search
[Alt+S]
Mumbai
siddSutar
Amazon Linux
macOS
Ubuntu
Windows
Red Hat
Browse more AMIs
Including AMIs from AWS, Marketplace and the Community
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences

Amazon Machine Image (AMI)

Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type — Free tier eligible
ami-0e742cca61fb65051 (64-bit (x86)) / ami-0b903415af59b1162 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Description
Amazon Linux 2 Kernel 5.10 AMI 2.0.20230207.0 x86_64 HVM gp2

Architecture: 64-bit (x86)
AMI ID: ami-0e742cca61fb65051 Verified provider

Instance type Info

Instance type
t2.micro — Free tier eligible
Family: t2 1 vCPU 1 GiB Memory
On-Demand Linux pricing: 0.0124 USD per Hour
On-Demand Windows pricing: 0.017 USD per Hour
On-Demand RHEL pricing: 0.0724 USD per Hour
On-Demand SUSE pricing: 0.0124 USD per Hour

Compare instance types

Summary

Number of instances Info
1

Software Image (AMI)
Amazon Linux 2 Kernel 5.10 AMI 2.0.20230207.0 x86_64 HVM gp2
ami-0e742cca61fb65051

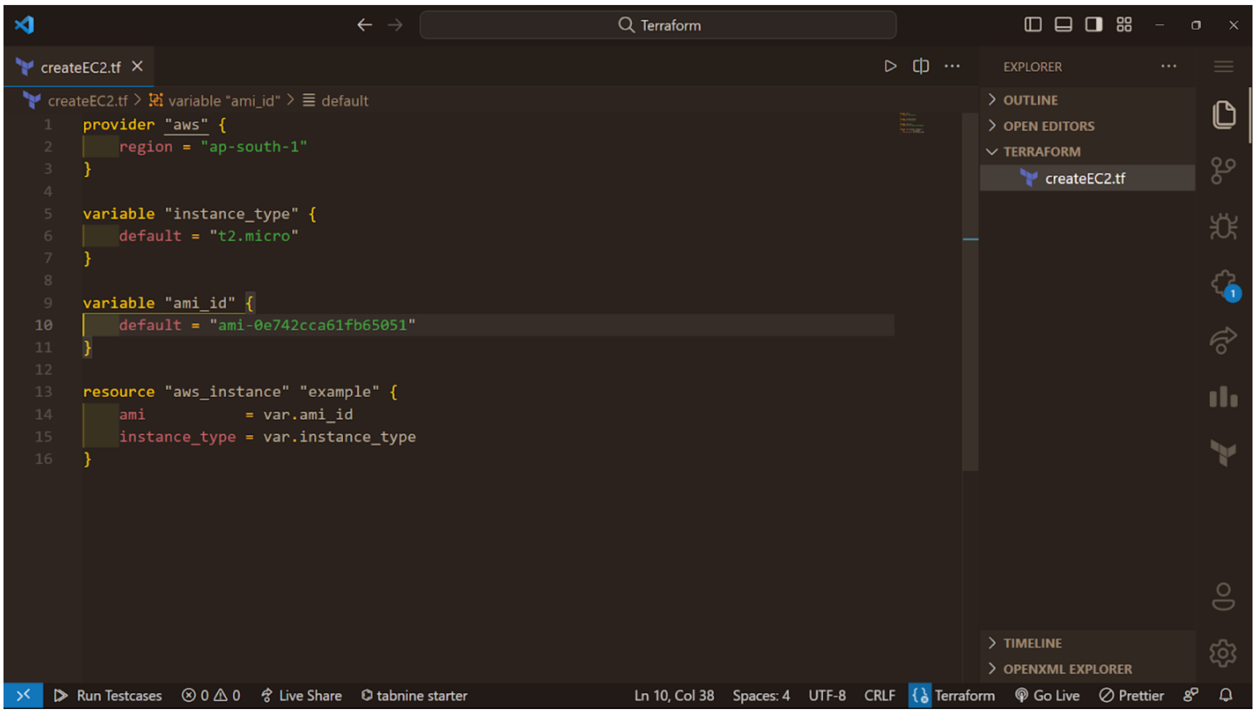
Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

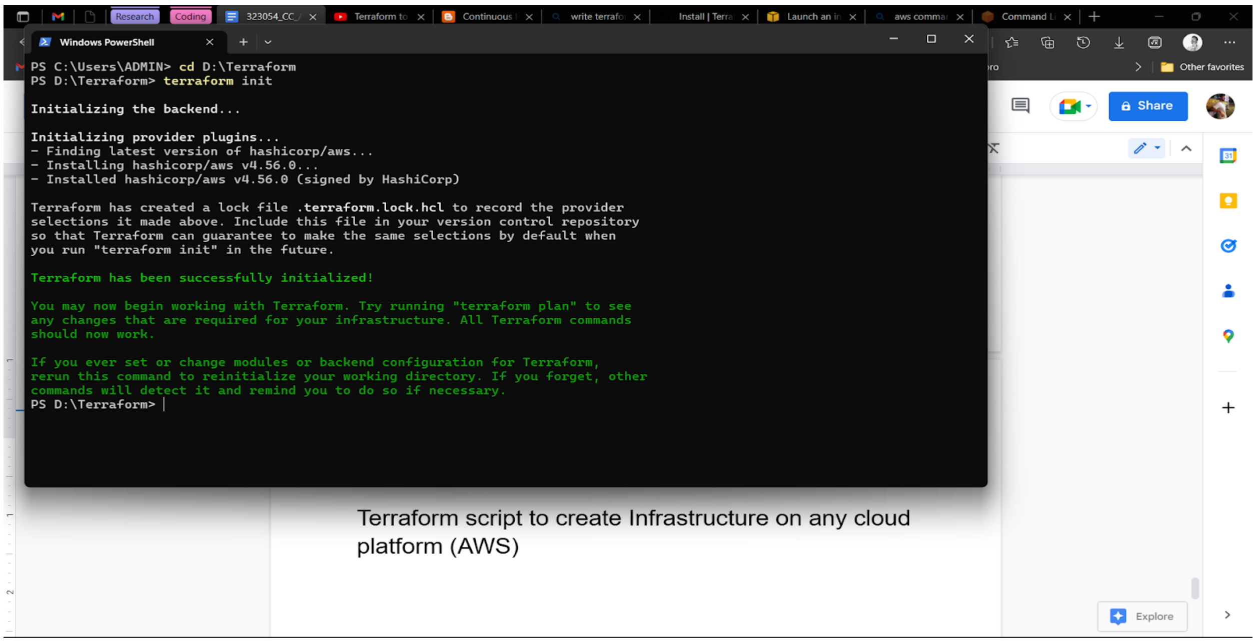
Storage (volumes)
1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage,

Cancel Launch instance

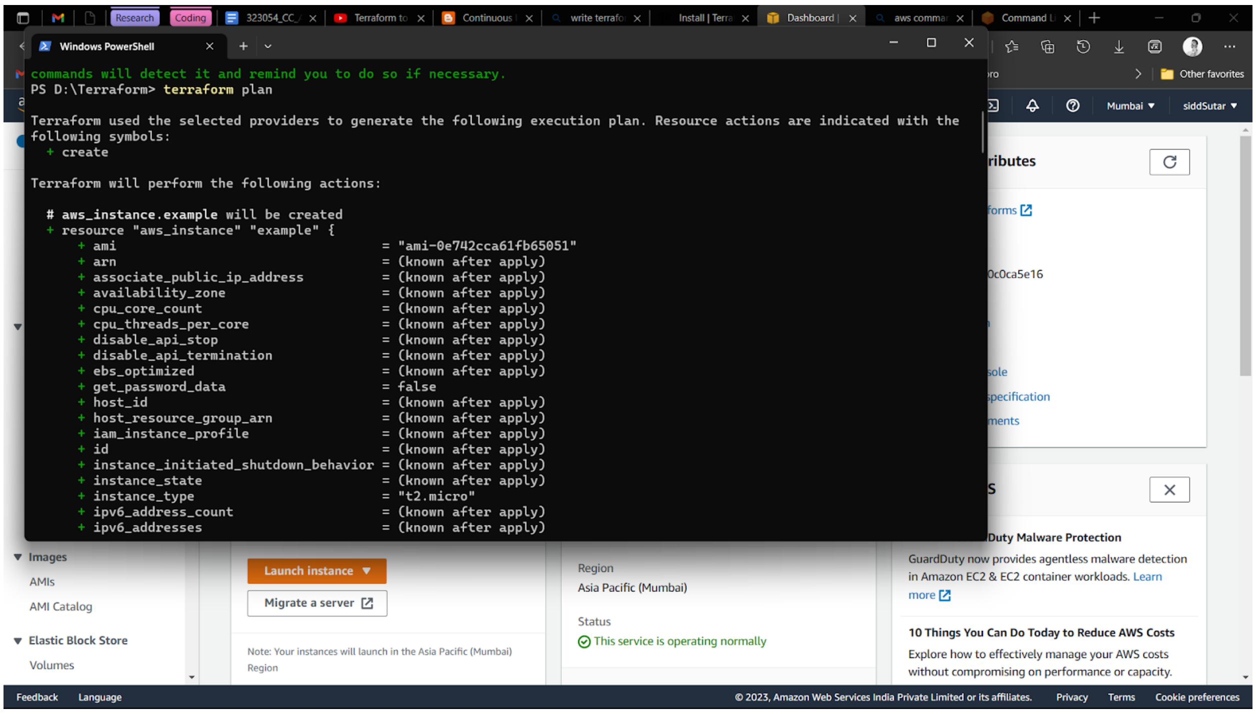


```
provider "aws" {
    region = "ap-south-1"
}

variable "instance_type" {
    default = "t2.micro"
}

variable "ami_id" {
    default = "ami-0e742cca61fb65051"
}

resource "aws_instance" "example" {
    ami           = var.ami_id
    instance_type = var.instance_type
}
```



```
PS C:\Users\ADMIN> cd D:\Terraform
PS D:\Terraform> terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v4.56.0...
- Installed hashicorp/aws v4.56.0 (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.
PS D:\Terraform> |
```

Terraform script to create Infrastructure on any cloud platform (AWS)



```
commands will detect it and remind you to do so if necessary.
PS D:\Terraform> 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-0e742cca61fb65051"
      + 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_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
```

Note: Your instances will launch in the Asia Pacific (Mumbai) Region

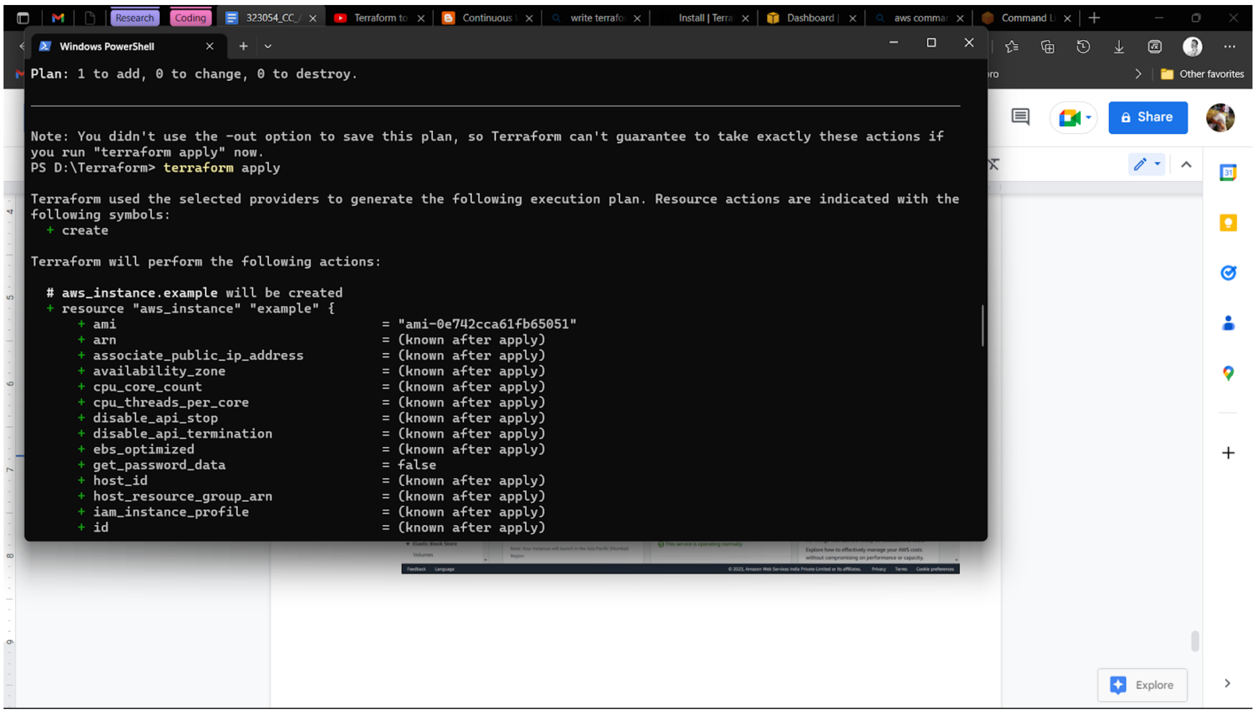
Region
Asia Pacific (Mumbai)

Status
This service is operating normally

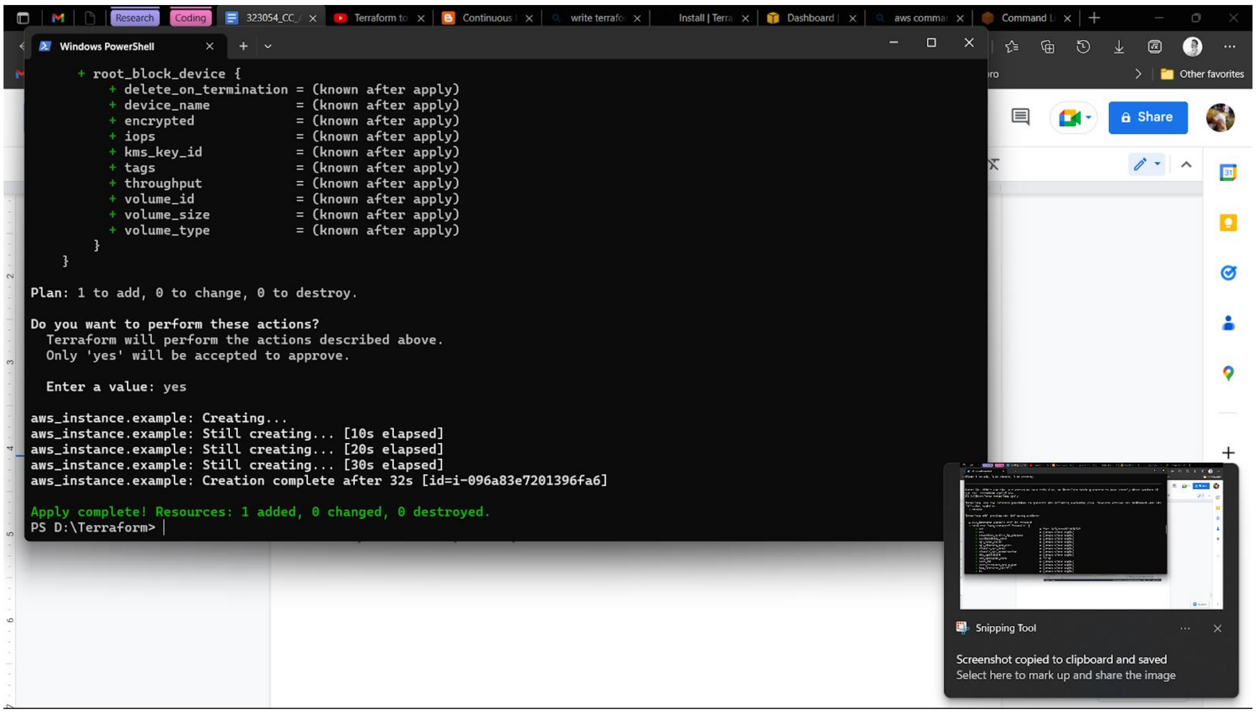
Duty Malware Protection
GuardDuty now provides agentless malware detection in Amazon EC2 & EC2 container workloads. Learn more

10 Things You Can Do Today to Reduce AWS Costs
Explore how to effectively manage your AWS costs without compromising on performance or capacity.

© 2023, Amazon Web Services India Private Limited or its affiliates.

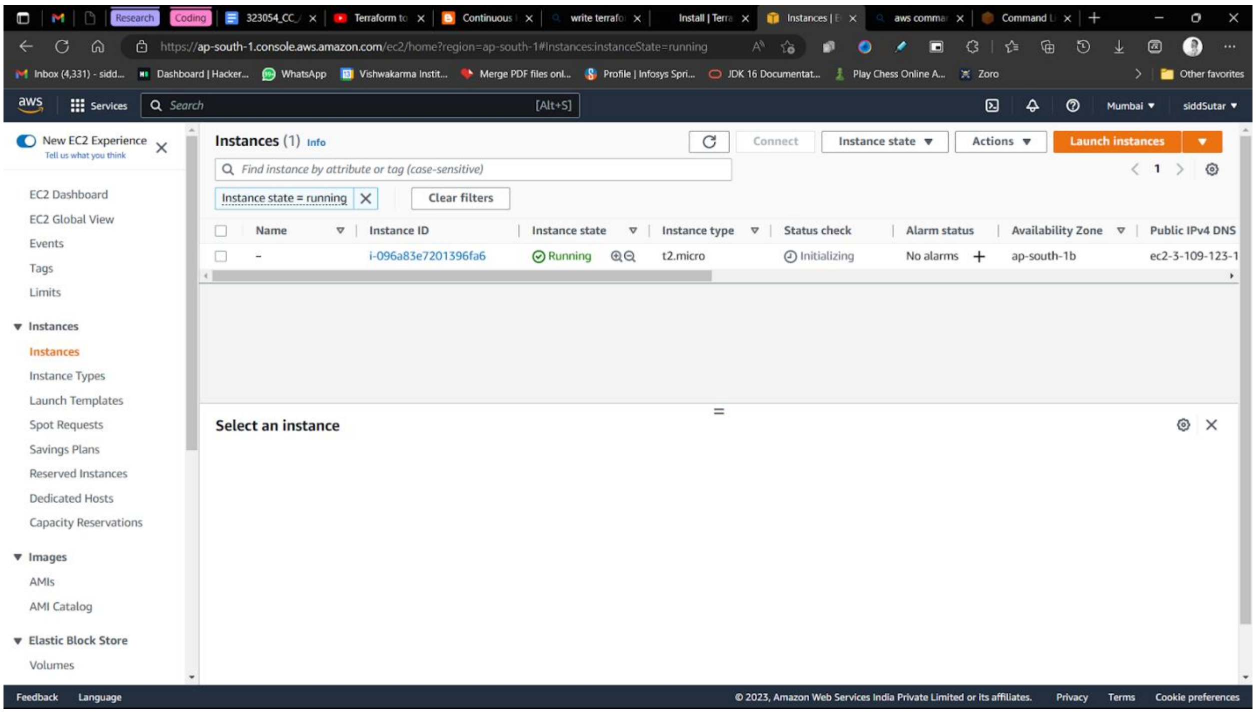


```
Plan: 1 to add, 0 to change, 0 to destroy.

─────────────────────────────────────────────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if
you run "terraform apply" now.
PS D:\Terraform> 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-0e742cca61fb65051"
      + 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)
```



```
      + root_block_device {
          + delete_on_termination = (known after apply)
          + device_name           = (known after apply)
          + encrypted             = (known after apply)
          + iops                  = (known after apply)
          + kms_key_id            = (known after apply)
          + tags                  = (known after apply)
          + throughput            = (known after apply)
          + volume_id             = (known after apply)
          + volume_size           = (known after apply)
          + volume_type           = (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 32s [id=i-096a83e7201396fa6]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS D:\Terraform> |
```

Windows PowerShell
Snipping Tool
Screenshot copied to clipboard and saved
Select here to mark up and share the image

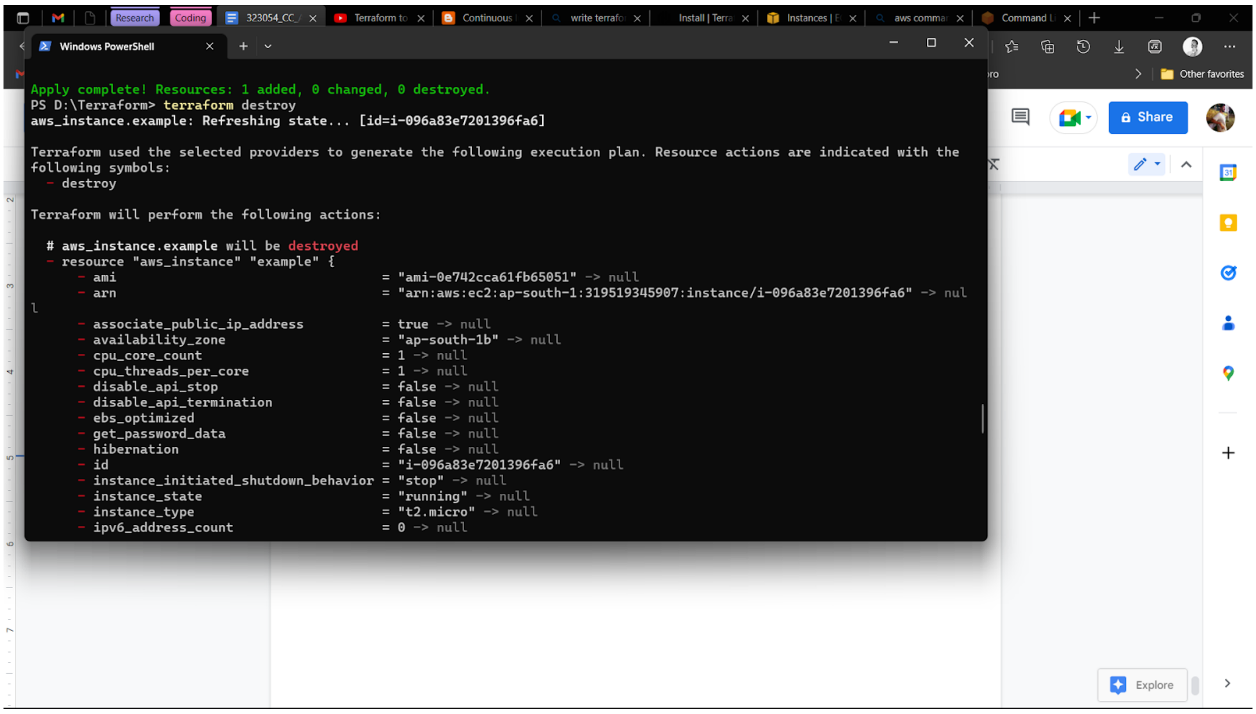


Instances (1) Info

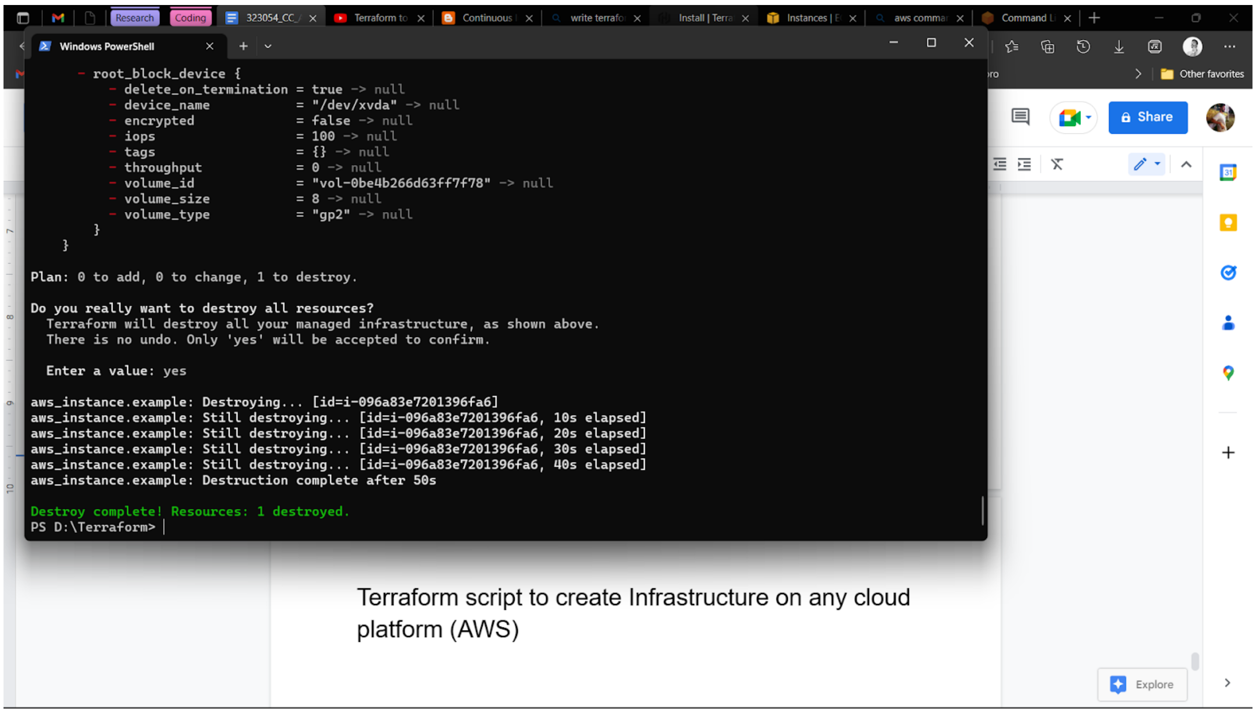
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
–	i-096a83e7201396fa6	Running	t2.micro	Initializing	No alarms	ap-south-1b	ec2-3-109-123-1

Select an instance

Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences

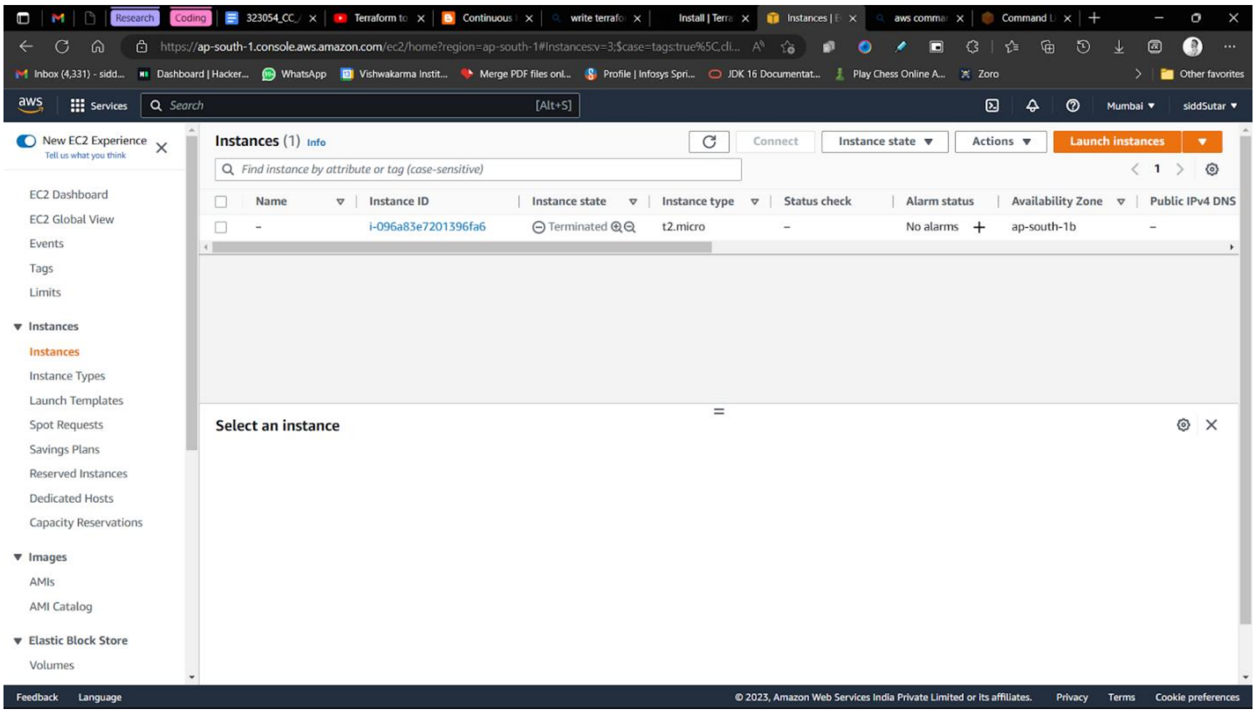


```
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS D:\Terraform> terraform destroy
aws_instance.example: Refreshing state... [id=i-096a83e7201396fa6]

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-0e742cca61fb65051" -> null
      - arn                                  = "arn:aws:ec2:ap-south-1:319519345907:instance/i-096a83e7201396fa6" -> nul
l
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "ap-south-1b" -> 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-096a83e7201396fa6" -> null
      - instance_initiated_shutdown_behavior = "stop" -> null
      - instance_state                       = "running" -> null
      - instance_type                        = "t2.micro" -> null
      - ipv6_address_count                   = 0 -> null
```

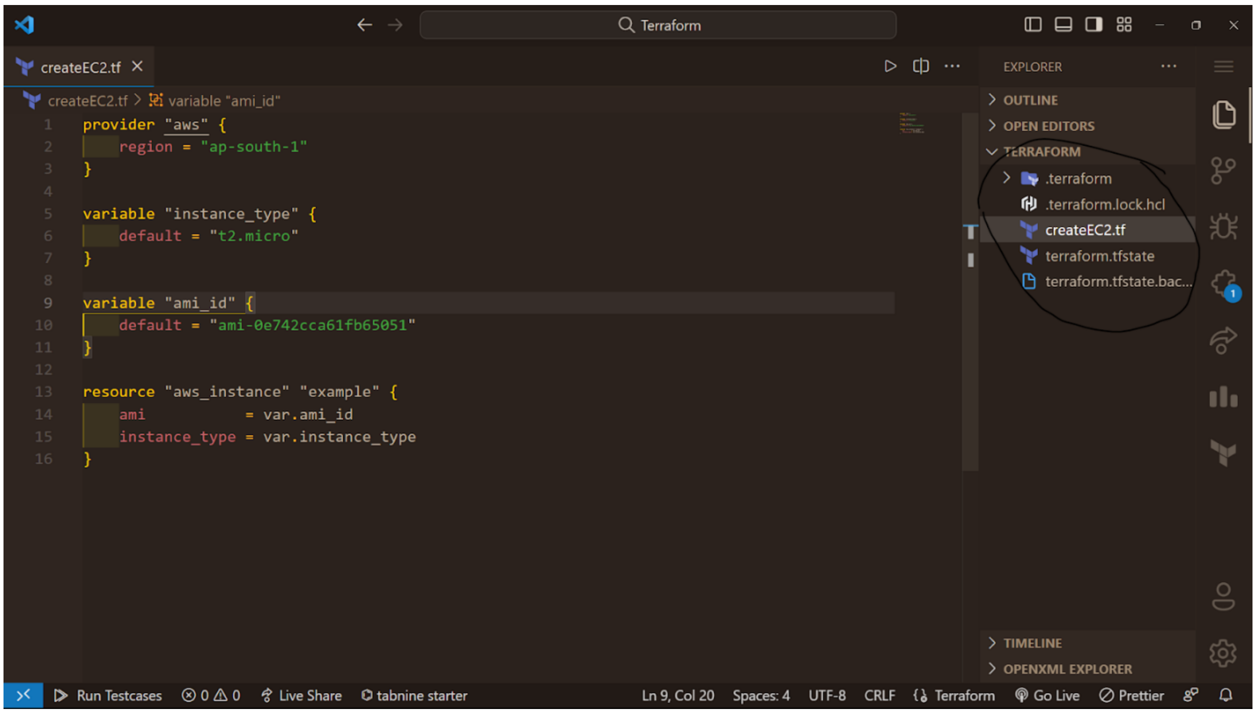


```
      - root_block_device {
          - delete_on_termination = true -> null
          - device_name           = "/dev/xvda" -> null
          - encrypted             = false -> null
          - iops                  = 100 -> null
          - tags                  = {} -> null
          - throughput            = 0 -> null
          - volume_id             = "vol-0be4b266d63ff7f78" -> 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-096a83e7201396fa6]
aws_instance.example: Still destroying... [id=i-096a83e7201396fa6, 10s elapsed]
aws_instance.example: Still destroying... [id=i-096a83e7201396fa6, 20s elapsed]
aws_instance.example: Still destroying... [id=i-096a83e7201396fa6, 30s elapsed]
aws_instance.example: Still destroying... [id=i-096a83e7201396fa6, 40s elapsed]
aws_instance.example: Destruction complete after 50s

Destroy complete! Resources: 1 destroyed.
PS D:\Terraform> |
```

Terraform script to create Infrastructure on any cloud platform (AWS)



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
–	i-096a83e7201396fa6	Terminated	t2.micro	–	No alarms	ap-south-1b	–



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

variable "instance_type" {
    default = "t2.micro"
}

variable "ami_id" {
    default = "ami-0e742cca61fb65051"
}

resource "aws_instance" "example" {
    ami           = var.ami_id
    instance_type = var.instance_type
}
```

Conclusion: -

Thus we have studied and implemented IaC using terraform to create EC2 machine on AWS or azure or google cloud