

# Terraform - Infrastructure as a Code

## Introduction:

**Infrastructure as a code | Automate Infrastructure**

**Define Infrastructure State (EC2, RDS, Load Balancers, Beanstalk)**

**Ansible, puppet and Chef only automated OS related tasks like Installation, what version, and What File we have to configure its Defines Machine State**

**On the Other hand, terraform automates Infrastructure State like any cloud provider GCP, Azure and AWS**

**Terraform Works With automation software like ansible after infra is setup and ready. No Programming Syntax its has own DSL (Domain Specific Language) Similar to JSON**

## Everything Needs Automation:

**Everything we know how to automate Systems in DevOps.**

**With Ansible we can do cloud automation**

**With Ansible we can do automation on CICD**

**with Ansible we can do automation on Jenkins**

---

X

Saiffaizal Panjeshha  
Technical Consultant

**But when it's come to Infrastructure itself, We Find best tool today is Terraform in the market to maintain the state of your cloud Infrastructure and also to automate it. So, you have centralized configuration of your cloud Infrastructure**

## **Installation:**

**Download Terraform Binaries From its Websites.**

- **Linux**
- **MacOS**
- **Windows**

**Stored Binaries in Exported Path:**

**e.g., Linux => /usr/local/bin**

**Task 1:**

**Launch EC2 Instance**

- **AWS Account**
- **IAM Users with Access Keys**
- **Terraform File to Launch Instance**
- **Run Terraform apply**

**Notes:**

**Write instance.tf file**

**Launch Instances**

**Make Some Changes to instance.tf file**

**Apply Changes**

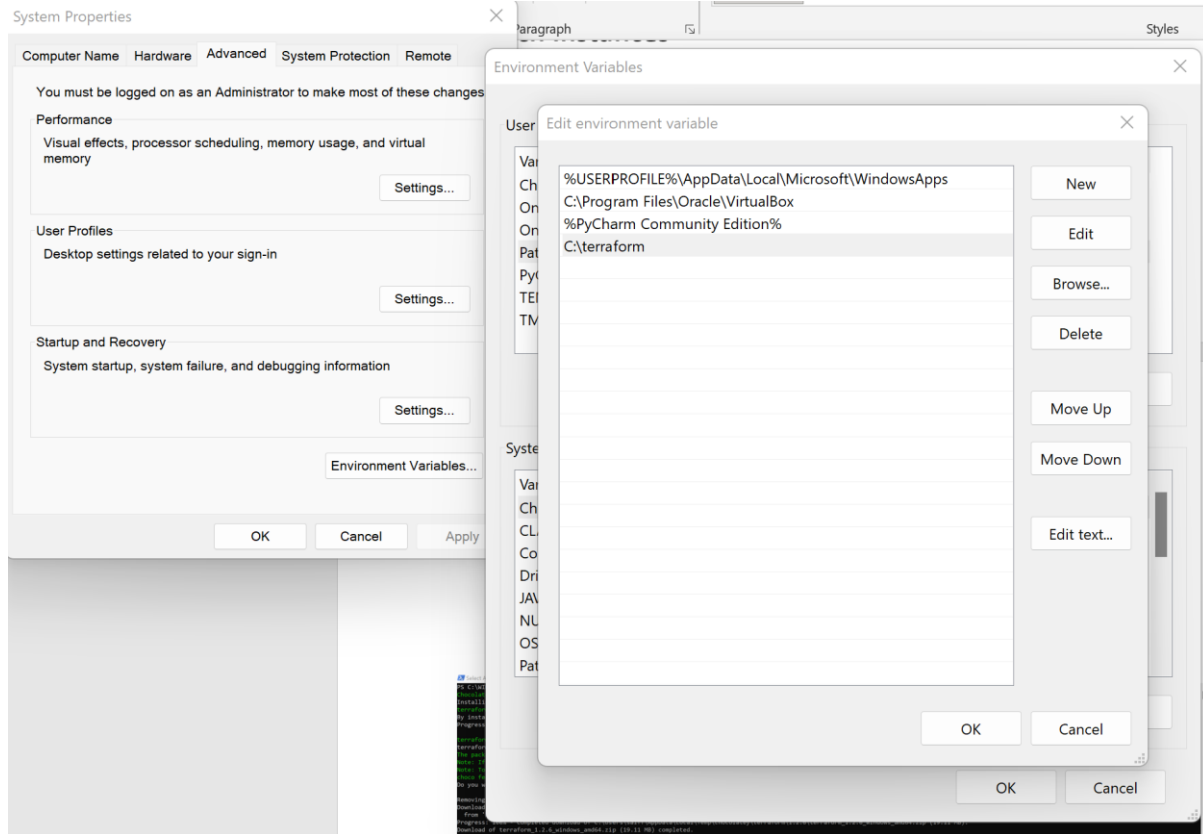
---

X

Saiffaizal Panjsha  
Technical Consultant

# Download Terraform from Official Website

<https://www.terraform.io/downloads>



```
Select Administrator: Windows PowerShell
PS C:\WINDOWS\system32> choco install terraform
Chocolatey v1.2.6
Installing the following packages:
terraform
By installing, you accept licenses for the packages.
Progress: Downloading terraform 1.2.6... 100%
terraform v1.2.6 [Approved]
terraform package files install completed. Performing other installation steps.
The package terraform wants to run 'chocolateyinstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider:
choco feature enable -n allowGlobalConfirmation
Do you want to run the script?([Y]es/[A]ll - yes to all/[N]o/[P]rint): Y

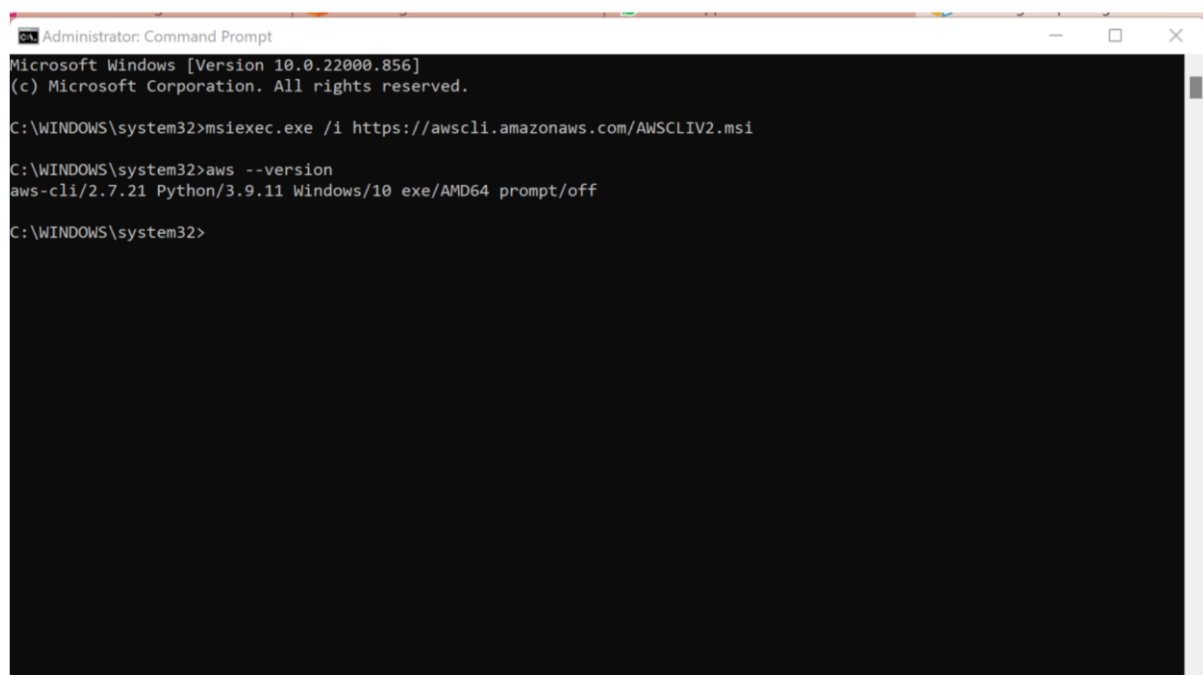
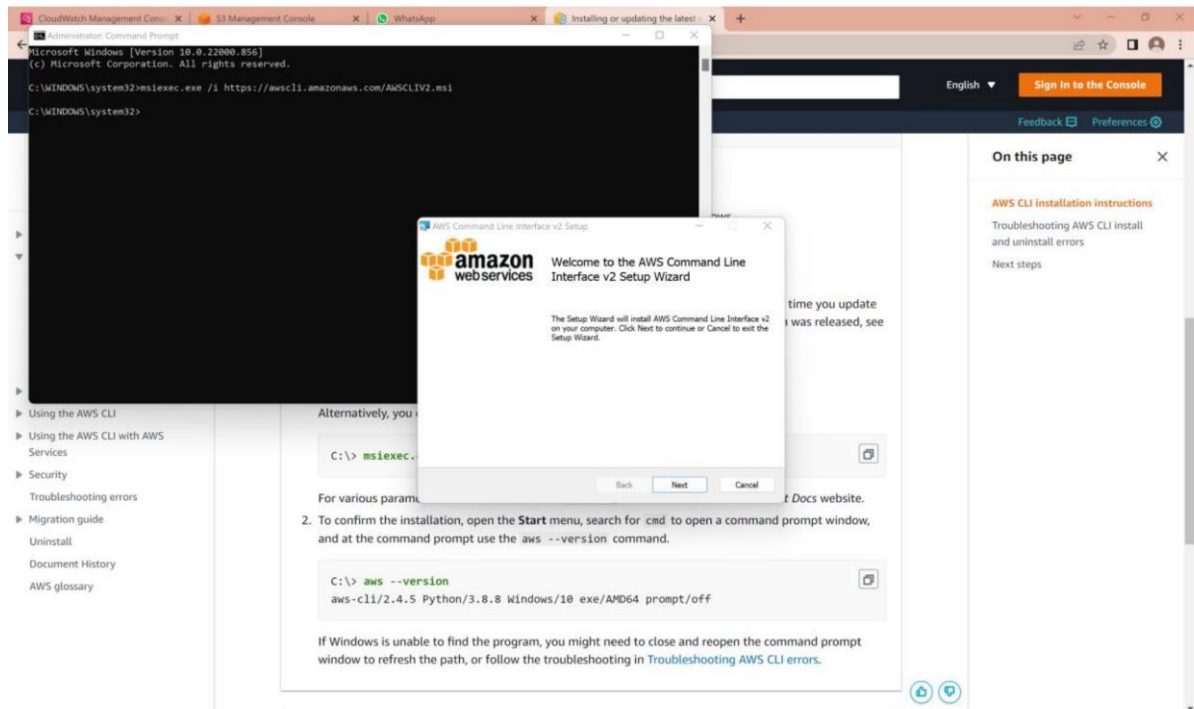
Removing old terraform plugins
Downloading terraform 64 bit
From 'https://releases.hashicorp.com/terraform/1.2.6/terraform_1.2.6_windows_amd64.zip'
Progress: 100% - Completed download of C:\Users\saiiff\AppData\Local\Temp\chocolatey\terraform\1.2.6\terraform_1.2.6_windows_amd64.zip (19.11 MB).
Download of terraform_1.2.6_windows_amd64.zip (19.11 MB) completed.
Hashes match.
Extracting C:\Users\saiiff\AppData\Local\Temp\chocolatey\terraform\1.2.6\terraform_1.2.6_windows_amd64.zip to C:\ProgramData\chocolatey\lib\terraform\tools...
ShimGen has successfully created a shim for terraform.exe
The install of terraform was successful.
Software installed to 'C:\ProgramData\chocolatey\lib\terraform\tools'
Chocolatey installed 1/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\WINDOWS\system32>
```

X

Saiffaizal Panjeshah  
Technical Consultant

# Installing AWSCLI

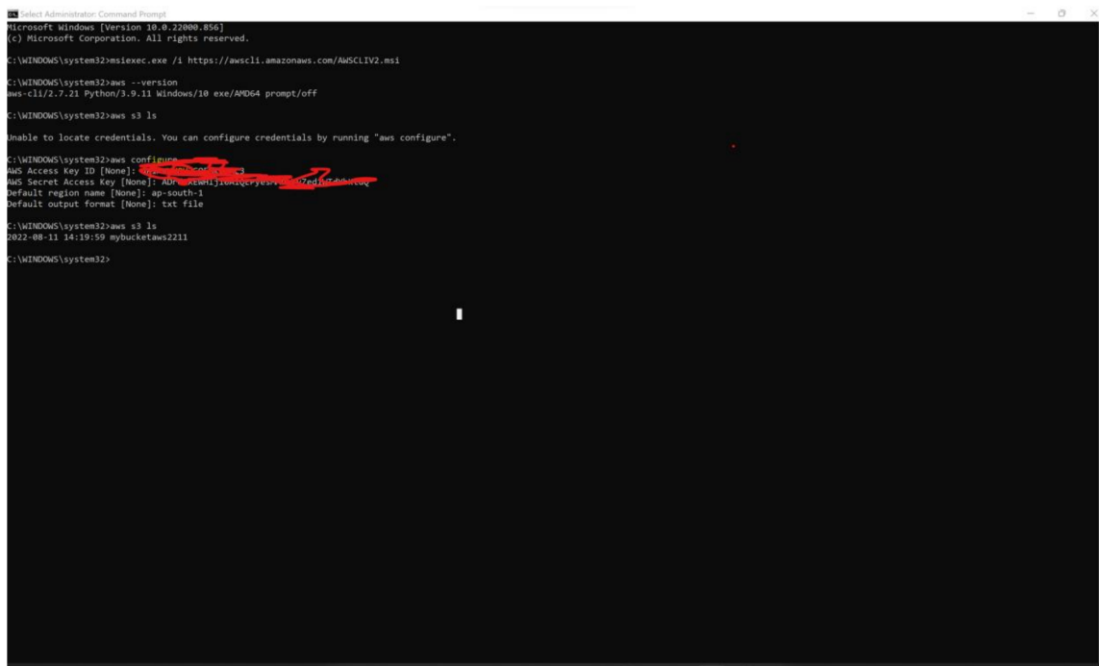
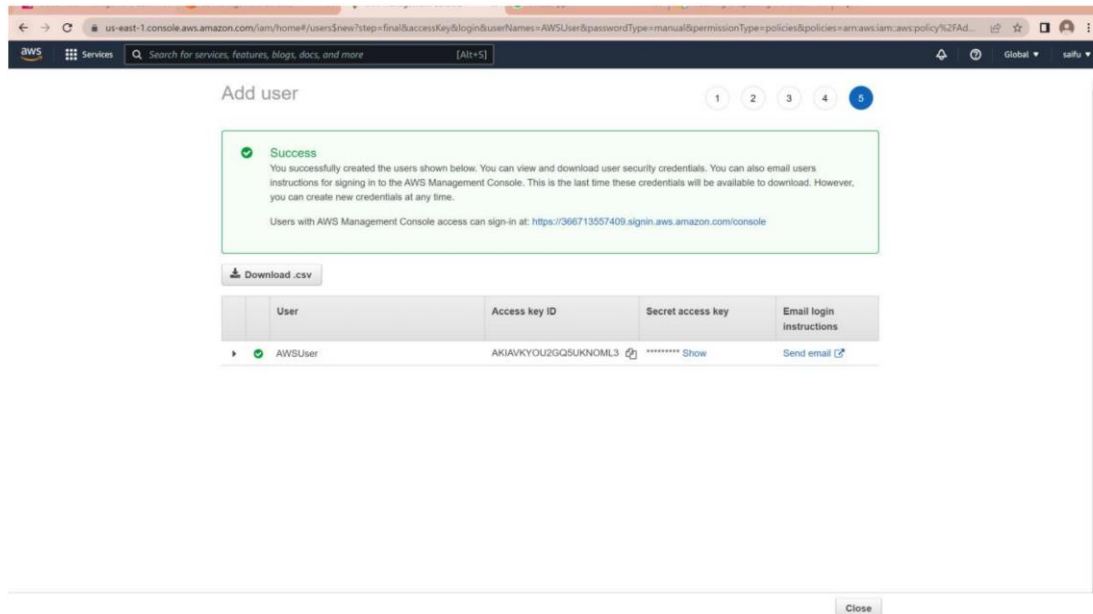
Download and run the AWS CLI MSI installer for Windows (64-bit):



X

Saiffaizal Panjeshia  
Technical Consultant

# AWS Configure:



X

Saiffaizal Panjesha  
Technical Consultant

# Terraform Task:

```
MINGW64/d/terraform-scripts/Task1
saiff@saiffpanjeshah MINGW64 ~
$ terraform --version
Terraform v1.2.6
on windows_amd64

your version of Terraform is out of date! The latest version
is 1.2.7. You can update by downloading from https://www.terraform.io/downloads.html

saiff@saiffpanjeshah MINGW64 ~
$ cd D:
saiff@saiffpanjeshah MINGW64 /d
$ mkdir terraform-scripts
saiff@saiffpanjeshah MINGW64 /d
$ cd ~/terraform-scripts
bash: cd: /c/Users/saiff/terraform-scripts: No such file or directory
saiff@saiffpanjeshah MINGW64 /d
$ cd /terraform-scripts
bash: cd: /terraform-scripts: No such file or directory
saiff@saiffpanjeshah MINGW64 /d
$ cd terraform-scripts
saiff@saiffpanjeshah MINGW64 /d/terraform-scripts
$ mkdir Task1
saiff@saiffpanjeshah MINGW64 /d/terraform-scripts
$ cd Task1
saiff@saiffpanjeshah MINGW64 /d/terraform-scripts/Task1
$
```

## Creating a File With name of firstinstance.tf

```
MINGW64/d/terraform-scripts/Task1
saiff@saiffpanjeshah MINGW64 /d/terraform-scripts/Task1
$
saiff@saiffpanjeshah MINGW64 /d/terraform-scripts/Task1
$ ls
firstinstance.tf
saiff@saiffpanjeshah MINGW64 /d/terraform-scripts/Task1
$ cat firstinstance.tf
provider "aws" {
  region = "ap-south-1"
  access_key = ""
  secret_key = ""
}

resource "aws_instance" "intro" {
  ami = "ami-068257025f72f470d"
  instance_type = "t2.micro"
  availability_zone = "ap-south-1a"
  key_name = "DevopskeyNew"
  vpc_security_groups_ids = ["sg-08e0adca5a4a569f1"]
  tags = {
    Name = "Terraform-Instance1"
  }
}
```

## Start the terraform with init command:

```
MINGW64/d/terraform-scripts/Task1
saiff@saiffpanjeshah MINGW64 /d/terraform-scripts/Task1
$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v4.27.0...
- Installed hashicorp/aws v4.27.0 (signed by HashiCorp)

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

Terraform has been successfully initialized!

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

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.

saiff@saiffpanjeshah MINGW64 /d/terraform-scripts/Task1
$
```

X

Saiffaizal Panjeshah  
Technical Consultant

# Validate the terraform:

```
MINGW64/d/terraform-scripts/Task1
saiff@SaifPanjeshah MINGW64 /d/terraform-scripts/Task1
$ terraform validate

Error: Unsupported argument

   on firstinstance.tf line 12, in resource "aws_instance" "intro":
  12:         vpc_security_groups_ids = ["sg-08e0adca5a4a569f1"]

An argument named "vpc_security_groups_ids" is not expected here. Did you
mean "vpc_security_group_ids"?

saiff@SaifPanjeshah MINGW64 /d/terraform-scripts/Task1
$
```

## Syntax Error:

```
D:\terraform-scripts\Task1\firstinstance.tf (Task1) - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

FOLDERS
  Task1
    .terraform
    .terraform.lock.hcl
    firstinstance.tf

firstinstance.tf
1 provider "aws" {
2   region = "ap-south-1"
3   # access_key = ""
4   # secret_key = ""
5 }
6
7 resource "aws_instance" "intro" {
8   ami = "ami-068257025f72f470d"
9   instance_type = "t2.micro"
10  availability_zone = "ap-south-1a"
11  key_name = "DevopsKeyNew"
12  vpc_security_group_ids = ["sg-08e0adca5a4a569f1"]
13  tags = {
14    Name = "Terraform-Instance1"
15  }
16 }
```

## Validate Successful

```
MINGW64/d/terraform-scripts/Task1
saiff@SaifPanjeshah MINGW64 /d/terraform-scripts/Task1
$ terraform validate

Error: Unsupported argument

   on firstinstance.tf line 12, in resource "aws_instance" "intro":
  12:         vpc_security_groups_ids = ["sg-08e0adca5a4a569f1"]

An argument named "vpc_security_groups_ids" is not expected here. Did you
mean "vpc_security_group_ids"?

saiff@SaifPanjeshah MINGW64 /d/terraform-scripts/Task1
$ terraform validate
Success! The configuration is valid.

saiff@SaifPanjeshah MINGW64 /d/terraform-scripts/Task1
$
```

X

Saiffaizal Panjeshah  
Technical Consultant

# Format Terraform Code in canonical Style

```
MINGW64/d/terraform-scripts/Task1
saiff@saiffpanjeshah MINGW64 /d/terraform-scripts/Task1
$ terraform fmt
firstinstance.tf

saiff@saiffpanjeshah MINGW64 /d/terraform-scripts/Task1
$ cat firstinstance.tf
provider "aws" {
  region = "ap-south-1"
  #      access_key = ""
  #      secret_key = ""
}

resource "aws_instance" "intro" {
  ami           = "ami-068257025f72f470d"
  instance_type = "t2.micro"
  availability_zone = "ap-south-1a"
  key_name       = "DevopsKeyNew"
  vpc_security_group_ids = ["sg-08e0adca5a4a569f1"]
  tags = {
    Name = "Terraform-Instance1"
  }
}
```

## Error:

```
MINGW64/d/terraform-scripts/Task1
saiff@saiffpanjeshah MINGW64 /d/terraform-scripts/Task1
$ terraform fmt
firstinstance.tf

saiff@saiffpanjeshah MINGW64 /d/terraform-scripts/Task1
$ terraform validate
Success! The configuration is valid.

saiff@saiffpanjeshah MINGW64 /d/terraform-scripts/Task1
$ terraform plan

Error: error configuring Terraform AWS Provider: error validating provider credentials: error calling sts:GetCallerIdentity: operation error
STS: GetCallerIdentity, net/http: invalid header field value "AWS4-HMAC-SHA256 Credential=\x16AKIAVKY0U2GQ7KEX3U5K/20220823/ap-south-1/sts/aws4
_request, SignedHeaders=amz-sdk-invocation-id;amz-sdk-request;content-length;content-type;host;x-amz-date, Signature=07727f67a8ee45cf8d9d30338d
06cc409e3ad919a125b7ef51a8b0c82ed32324" for key Authorization

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

## Resolving Error:

```
Microsoft Windows [Version 10.0.22000.856]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>msiexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi

C:\WINDOWS\system32>aws --version
aws-cli/2.7.21 Python/3.9.11 Windows/10 exe/AMD64 prompt/off

C:\WINDOWS\system32>aws s3 ls
Unable to locate credentials. You can configure credentials by running "aws configure".

C:\WINDOWS\system32>aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUzfWh67bq747494MgQy3oP0h8R2VW
Default region name [None]: ap-south-1
Default output format [None]: txt file

C:\WINDOWS\system32>aws s3 ls
2022-08-11 14:19:59 mybucketaws2211

C:\WINDOWS\system32>
```

X

Saiffaizal Panjeshah  
Technical Consultant



# Successful Plan with actions:

```
MINGW64/d/terraform-scripts/Task1
$ 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.intro will be created
+ resource "aws_instance" "intro" {
+   ami                    = "ami-068257025f72f470d"
+   arn                    = (known after apply)
+   associate_public_ip_address = (known after apply)
+   availability_zone       = "ap-south-1a"
+   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)
+   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)
+   key_name                 = "AWSDev"
+   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
+   subnet_id                = (known after apply)
+   tags                     = {
+     "Name" = "Terraform-Instance1"
+   }
+   tags_all                = {
+     "Name" = "Terraform-Instance1"
```

```
    + maintenance_options {
      + auto_recovery = (known after apply)
    }

    + metadata_options {
      + http_endpoint           = (known after apply)
      + http_put_response_hop_limit = (known after apply)
      + http_tokens             = (known after apply)
      + instance_metadata_tags  = (known after apply)
    }

    + network_interface {
      + delete_on_termination = (known after apply)
      + device_index          = (known after apply)
      + network_card_index    = (known after apply)
      + network_interface_id  = (known after apply)
    }

    + private_dns_name_options {
      + enable_resource_name_dns_a_record = (known after apply)
      + enable_resource_name_dns_aaaa_record = (known after apply)
      + hostname_type                     = (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.

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.
```

X

Saiffaizal Panjasha  
Technical Consultant

```
MINGW64/terraform-scripts/Task1
$ 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.intro will be created
+ resource "aws_instance" "intro" {
  + ami                    = "ami-068257025f72f470d"
  + arn                   = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone      = "ap-south-1a"
  + 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)
  + 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)
  + key_name               = "AWSDev"
  + 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
  + subnet_id              = (known after apply)
  + tags                   = {
    + "Name" = "Terraform-Instance1"
  }
  + tags_all              = {
    + "Name" = "Terraform-Instance1"
  }
```

```
    }
  + network_interface {
    + delete_on_termination = (known after apply)
    + device_index          = (known after apply)
    + network_card_index    = (known after apply)
    + network_interface_id  = (known after apply)
  }

  + private_dns_name_options {
    + enable_resource_name_dns_a_record = (known after apply)
    + enable_resource_name_dns_aaaa_record = (known after apply)
    + hostname_type                     = (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)
  }
}

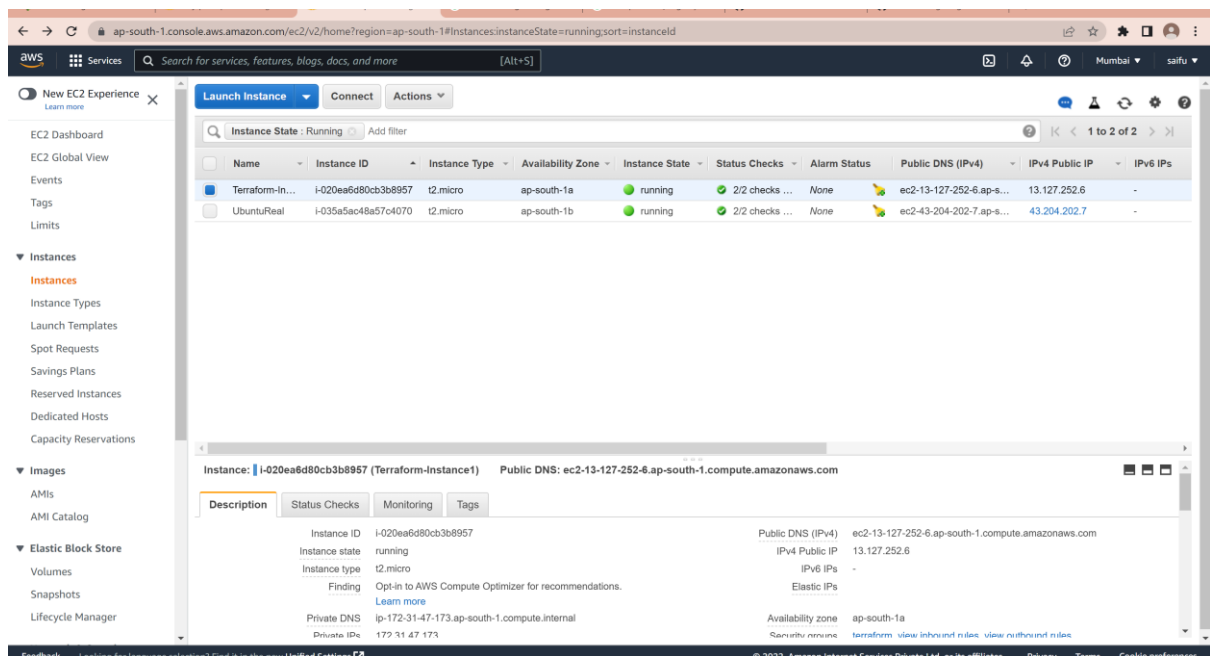
Plan: 1 to add, 0 to change, 0 to destroy.

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

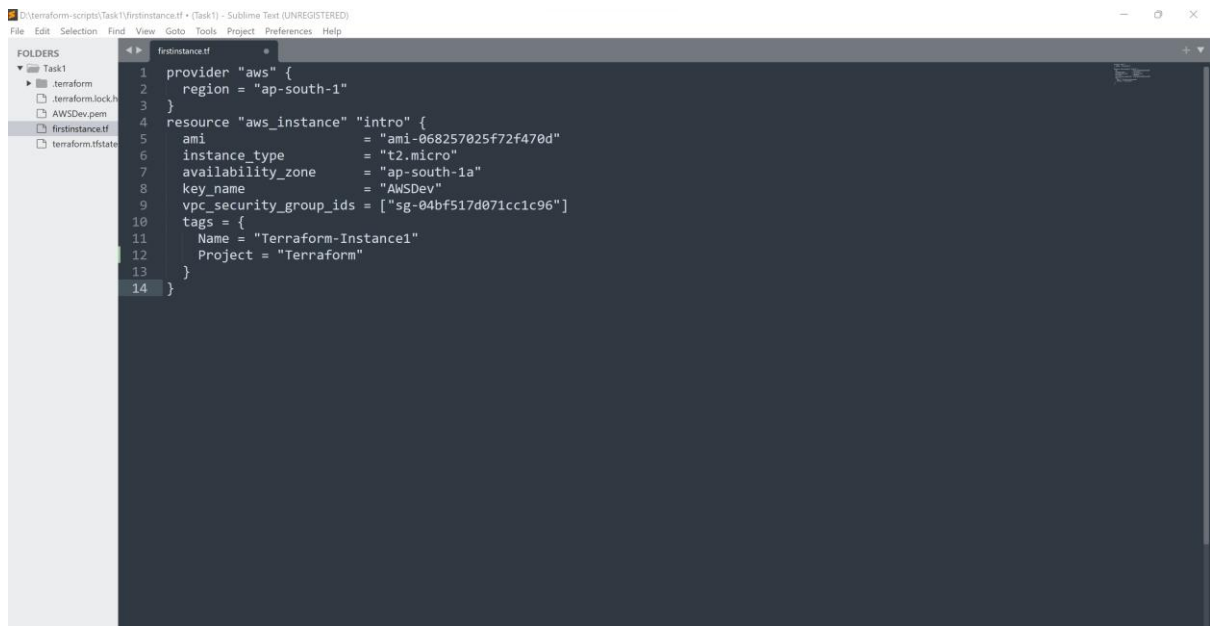
Enter a value: yes

aws_instance.intro: Creating...
aws_instance.intro: Still creating... [10s elapsed]
aws_instance.intro: Still creating... [20s elapsed]
aws_instance.intro: Still creating... [30s elapsed]
aws_instance.intro: Creation complete after 32s [id=i-020ea6d80cb3b8957]

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



## Let's Make Some Changes I have projects tag in File



X

Saiffaizal Panjeshia  
Technical Consultant

```

MINGW64/d/terraform-scripts/Task1
$ ls
AWSDev.pem  firstinstance.tf  terraform.tfstate

saiff@SaiffPanjeshah MINGW64 /d/terraform-scripts/Task1
$ terraform.exe validate
Success! The configuration is valid.

saiff@SaiffPanjeshah MINGW64 /d/terraform-scripts/Task1
$ terraform.exe fmt
firstinstance.tf

saiff@SaiffPanjeshah MINGW64 /d/terraform-scripts/Task1
$ cat firstinstance.tf
provider "aws" {
  region = "ap-south-1"
}

resource "aws_instance" "intro" {
  ami           = "ami-068257025f72f470d"
  instance_type = "t2.micro"
  availability_zone = "ap-south-1a"
  key_name      = "AWSDev"
  vpc_security_group_ids = ["sg-04bf517d071cc1c96"]
  tags = {
    Name     = "Terraform-Instance1"
    Project  = "Terraform"
  }
}

saiff@SaiffPanjeshah MINGW64 /d/terraform-scripts/Task1
$ terraform plan
aws_instance.intro: Refreshing state... [id=i-020ea6d80cb3b8957]

Terraform used the selected providers to generate the following execution
plan. Resource actions are indicated with the following symbols:
~ update in-place

Terraform will perform the following actions:

# aws_instance.intro will be updated in-place
~ resource "aws_instance" "intro" {
  id           = "i-020ea6d80cb3b8957"
  ~ tags       = {
    + "Project" = "Terraform"
    # (1 unchanged element hidden)
  }
  ~ tags_all   = {
    + "Project" = "Terraform"
  }
}

```

```

}

resource "aws_instance" "intro" {
  ami           = "ami-068257025f72f470d"
  instance_type = "t2.micro"
  availability_zone = "ap-south-1a"
  key_name      = "AWSDev"
  vpc_security_group_ids = ["sg-04bf517d071cc1c96"]
  tags = {
    Name     = "Terraform-Instance1"
    Project  = "Terraform"
  }
}

saiff@SaiffPanjeshah MINGW64 /d/terraform-scripts/Task1
$ terraform plan
aws_instance.intro: Refreshing state... [id=i-020ea6d80cb3b8957]

Terraform used the selected providers to generate the following execution
plan. Resource actions are indicated with the following symbols:
~ update in-place

Terraform will perform the following actions:

# aws_instance.intro will be updated in-place
~ resource "aws_instance" "intro" {
  id           = "i-020ea6d80cb3b8957"
  ~ tags       = {
    + "Project" = "Terraform"
    # (1 unchanged element hidden)
  }
  ~ tags_all   = {
    + "Project" = "Terraform"
    # (1 unchanged element hidden)
  }
  # (30 unchanged attributes hidden)
  # (7 unchanged blocks hidden)
}

Plan: 0 to add, 1 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.

X

Saiffaizal Panjeshah  
Technical Consultant

```

saiff@SaifPanjeshah MINGW64 /d/terraform-scripts/Task1
$ terraform apply
aws_instance.intro: Refreshing state... [id=i-020ea6d80cb3b8957]

Terraform used the selected providers to generate the following execution
plan. Resource actions are indicated with the following symbols:
  ~ update in-place

Terraform will perform the following actions:

# aws_instance.intro will be updated in-place
~ resource "aws_instance" "intro" {
  id          = "i-020ea6d80cb3b8957"
  ~ tags      = {
    + "Project" = "Terraform"
    # (1 unchanged element hidden)
  }
  ~ tags_all  = {
    + "Project" = "Terraform"
    # (1 unchanged element hidden)
  }
  # (30 unchanged attributes hidden)
  # (7 unchanged blocks hidden)
}

Plan: 0 to add, 1 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.intro: Modifying... [id=i-020ea6d80cb3b8957]
aws_instance.intro: Modifications complete after 1s [id=i-020ea6d80cb3b8957]

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

saiff@SaifPanjeshah MINGW64 /d/terraform-scripts/Task1
$

```

The screenshot displays the AWS Management Console interface for the 'ap-south-1' region. The 'Launch Instance' button is visible at the top. Below it, a table lists EC2 instances. The instance 'Terraform-In...' (ID: i-020ea6d80cb3b8957) is shown in a 'running' state. The 'Tags' tab is selected for this instance, showing a single tag with the key 'Project' and the value 'Terraform'.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP	IPv6 IPs
Terraform-In...	i-020ea6d80cb3b8957	t2.micro	ap-south-1a	running	2/2 checks ...	None	ec2-13-127-252-6.ap-s...	13.127.252.6	-
UbuntuReal	i-035a5ac48a57c4070	t2.micro	ap-south-1b	running	2/2 checks ...	None	ec2-43-204-202-7.ap-s...	43.204.202.7	-

Below the table, the 'Tags' tab for the selected instance is shown. It contains a table with the following data:

Key	Value
Name	Terraform-Instance1
Project	Terraform

X

Saiffaizal Panjeshah  
Technical Consultant

# Terraform Destroy:

```
MINGW64 ~/terraform-scripts/Task1
$ terraform destroy
aws_instance.intro: Refreshing state... [id=i-020ea6d80cb3b8957]

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.intro will be destroyed
- resource "aws_instance" "intro" {
  - ami                  = "ami-068257025f72f470d" -> null
  - arm                 = "arn:aws-ec2:ap-south-1:366713557409:instance/i-020ea6d80cb3b8957" -> null
  - associate_public_ip_address = true -> null
  - availability_zone     = "ap-south-1a" -> null
  - cpu_core_count        = 1 -> null
  - cpu_threads_per_core   = 1 -> null
  - disable_api_stop       = false -> null
  - disable_api_termination = false -> null
  - ebs_optimized          = false -> null
  - get_password_data      = false -> null
  - hibernation            = false -> null
  - id                    = "i-020ea6d80cb3b8957" -> null
  - instance_initiated_shutdown_behavior = "stop" -> null
  - instance_state         = "running" -> null
  - instance_type          = "t2.micro" -> null
  - ipv6_address_count      = 0 -> null
  - ipv6_addresses         = [] -> null
  - key_name               = "AwsSDev" -> null
  - monitoring             = false -> null
  - primary_network_interface_id = "eni-0f2a9f087b3702f84" -> null
  - private_dns            = "ip-172-31-47-173.ap-south-1.compute.internal" -> null
  - private_ip             = "172.31.47.173" -> null
  - public_dns             = "ec2-13-127-252-6.ap-south-1.compute.amazonaws.com" -> null
  - public_ip              = "13.127.252.6" -> null
  - secondary_private_ips   = [] -> null
  - security_groups        = ["terraform"] -> null
  - source_dest_check       = true -> null
  - subnet_id              = "subnet-0789bb86a3e7f2bb7" -> null
  - tags                   = {
    - "Name" = "Terraform-Instance1"
    - "Project" = "Terraform"
  } -> null
  - tags_all              = {
}

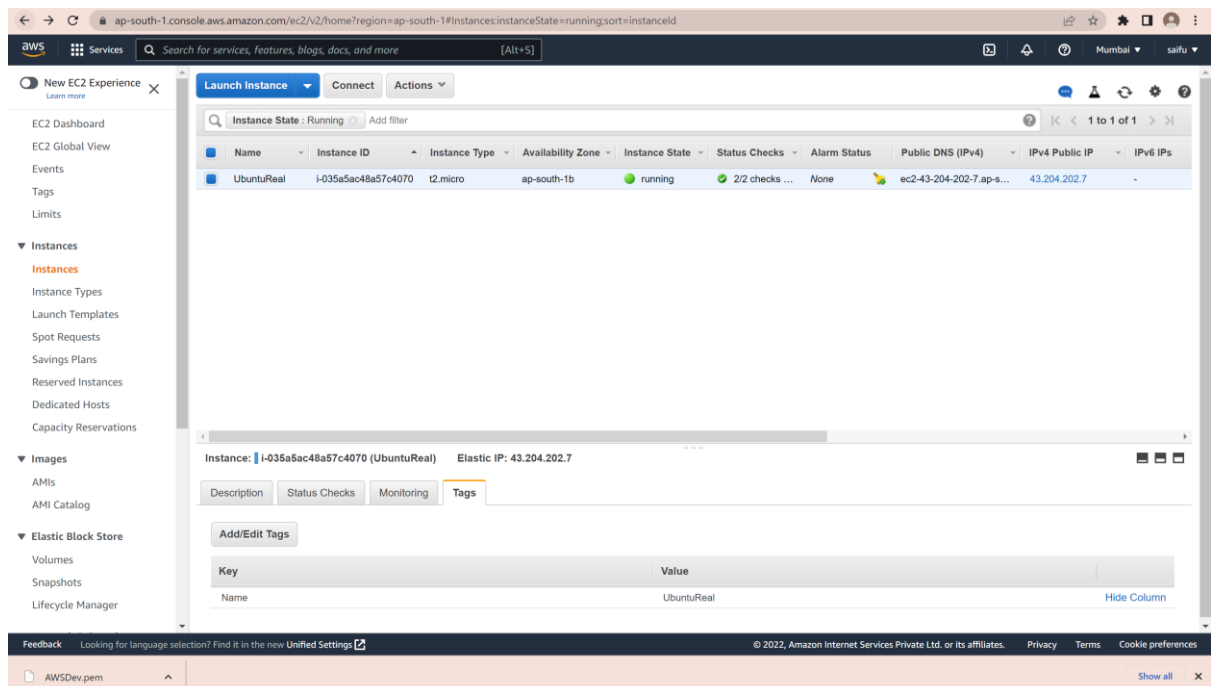
Plan: 0 to add, 0 to change, 1 to destroy.

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.intro: Destroying... [id=i-020ea6d80cb3b8957]
aws_instance.intro: Still destroying... [id=i-020ea6d80cb3b8957, 10s elapsed]
aws_instance.intro: Still destroying... [id=i-020ea6d80cb3b8957, 20s elapsed]
aws_instance.intro: Still destroying... [id=i-020ea6d80cb3b8957, 30s elapsed]
aws_instance.intro: Still destroying... [id=i-020ea6d80cb3b8957, 40s elapsed]
aws_instance.intro: Still destroying... [id=i-020ea6d80cb3b8957, 50s elapsed]
aws_instance.intro: Destruction complete after 50s

Destroy complete! Resources: 1 destroyed.
```



X

Saiffaizal Panjेशa  
Technical Consultant

X

Saiffaizal Panjesha  
Technical Consultant