Terrraform - 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 Panjesha
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

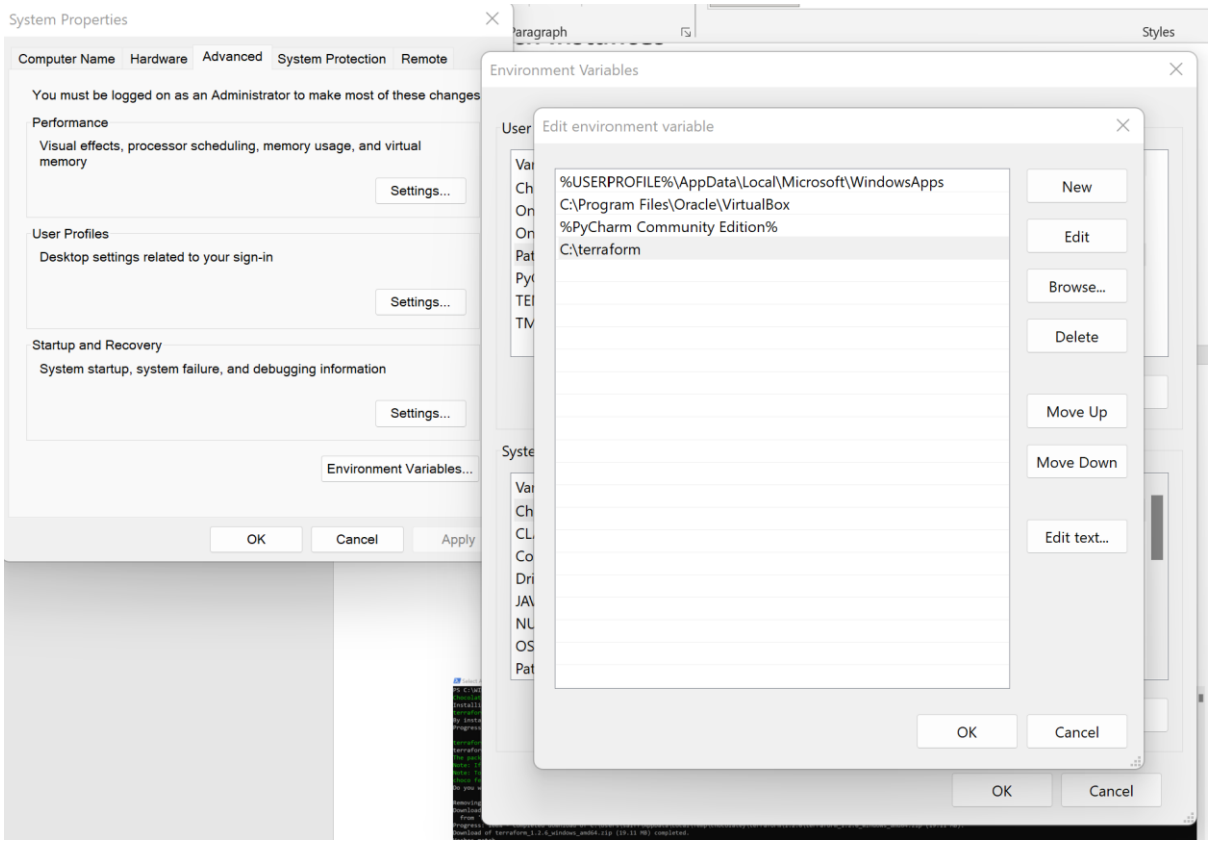
Apply Changes

X

Saiffaizal Panjesha
Technical Consultant

Download Terraform from Official Website

https://www.terraform.io/downloads



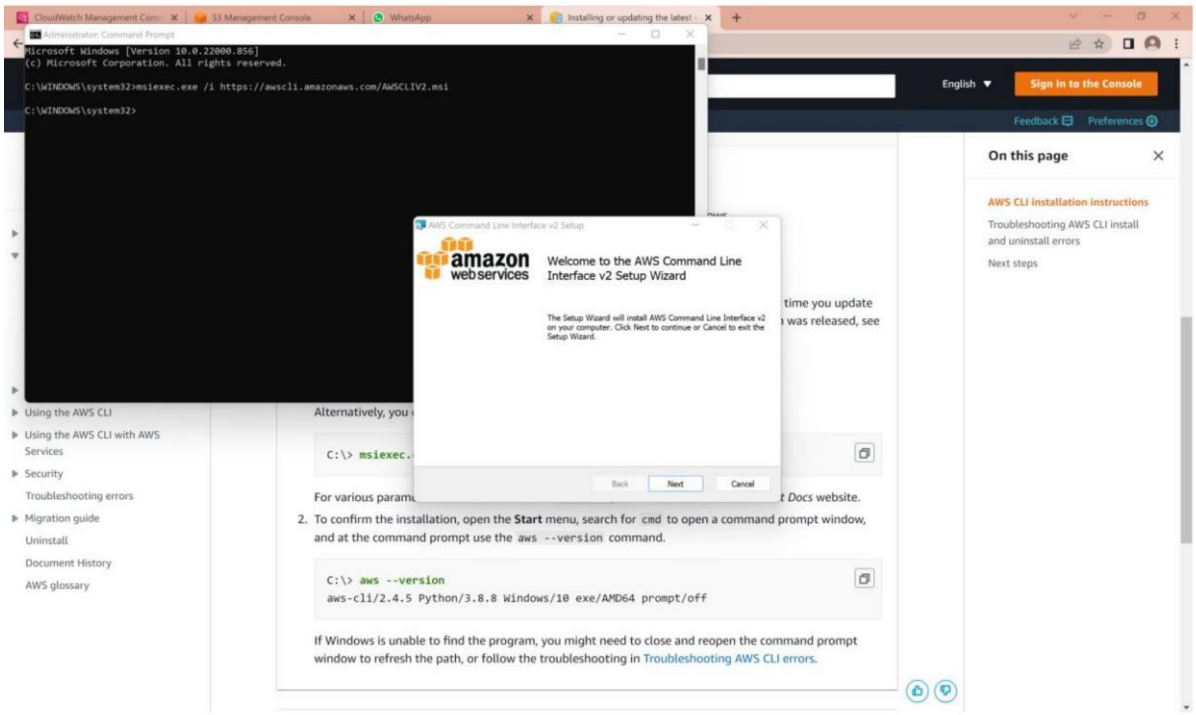
X

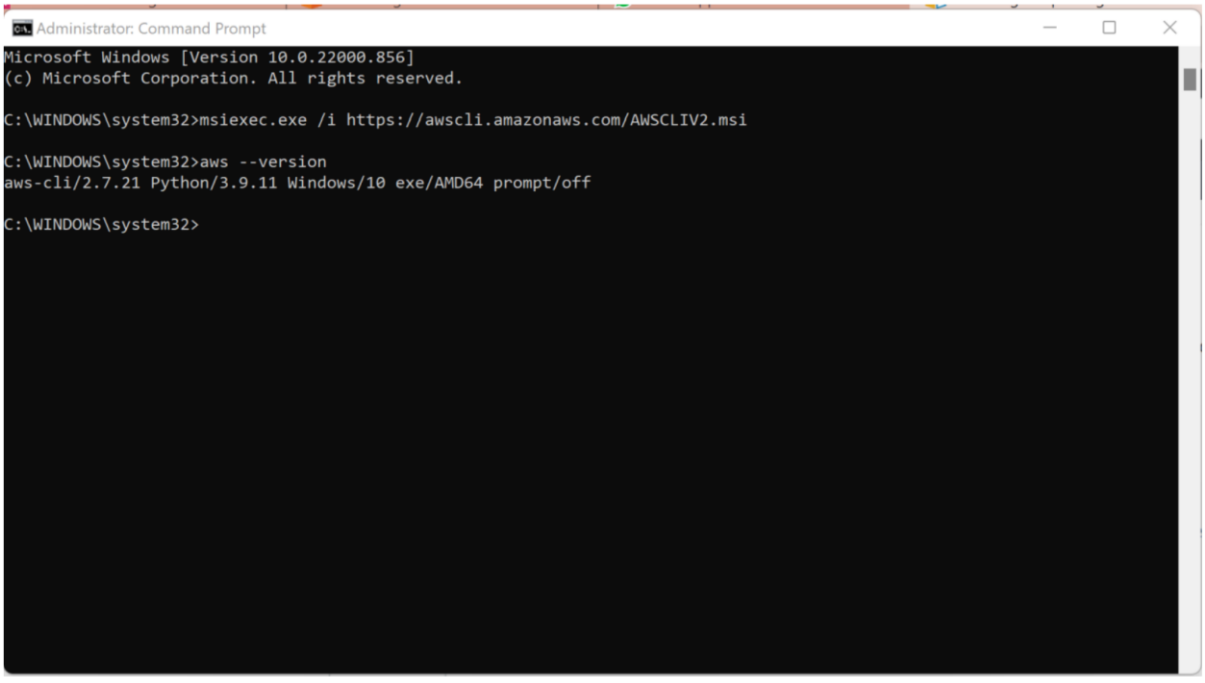
Saiffaizal Panjesha

Technical Consultant

Installing AWSCLI

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



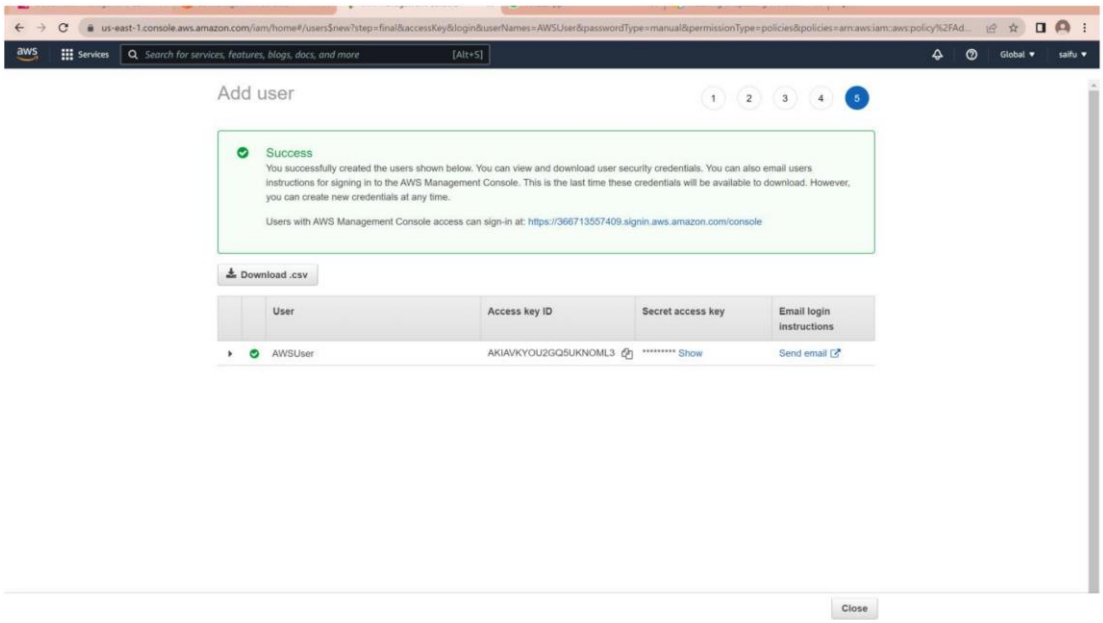


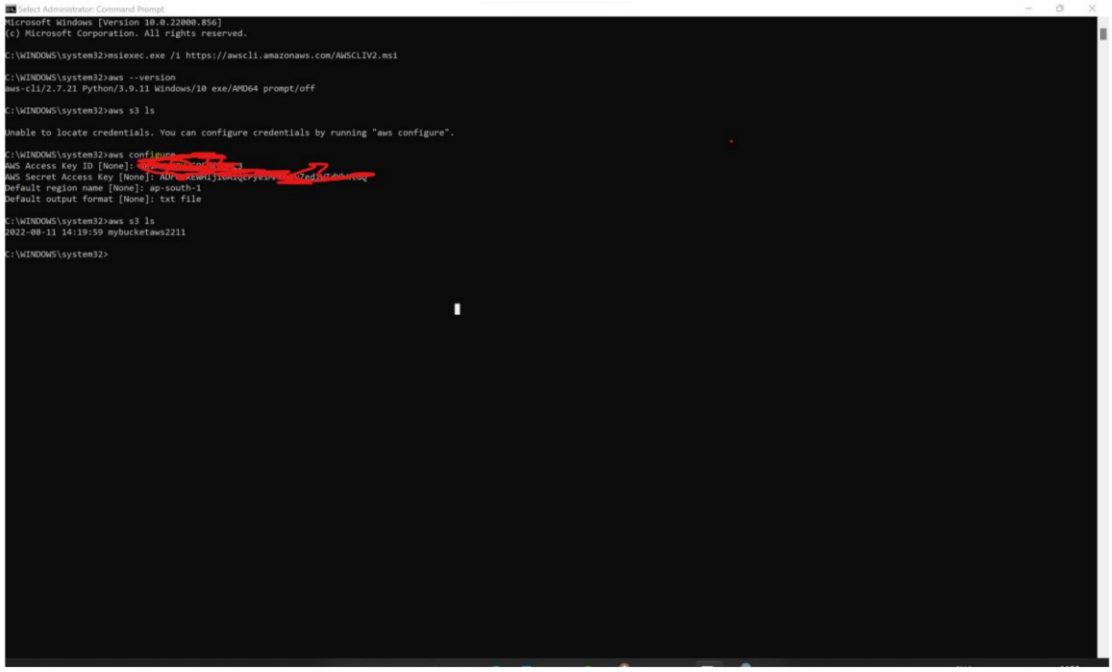
X

Saiffaizal Panjesha

Technical Consultant

AWS Configure:





X

Saiffaizal Panjesha

Technical Consultant

Terraform Task:

```
MINGW64:/d/terraform-scripts/Task1

saiff@SaifPanjeshah 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@SaifPanjeshah MINGW64 ~
$ cd D:

saiff@SaifPanjeshah MINGW64 /d
$ mkdir terraform-scripts

saiff@SaifPanjeshah MINGW64 /d
$ cd ~/terraform-scripts
bash: cd: /c/Users/saiff/terraform-scripts: No such file or directory

saiff@SaifPanjeshah MINGW64 /d
$ cd /terraform-scripts
bash: cd: /terraform-scripts: No such file or directory

saiff@SaifPanjeshah MINGW64 /d
$ cd terraform-scripts

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

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

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

Creating a File With name of firstinstance.tf

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

saiff@SaifPanjeshah MINGW64 /d/terraform-scripts/Task1
$ ls
firstinstance.tf

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

X

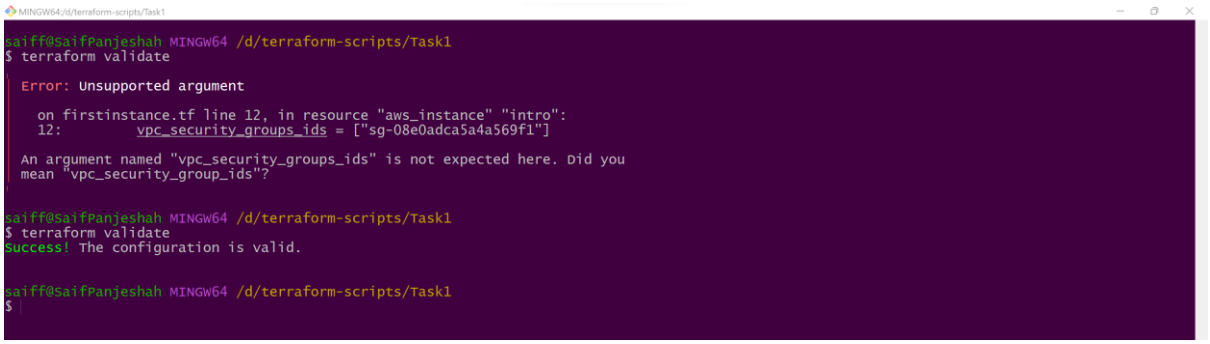
Saiffaizal Panjesha

Technical Consultant

Validate the terraform:

Syntax Error:

Validate Successful

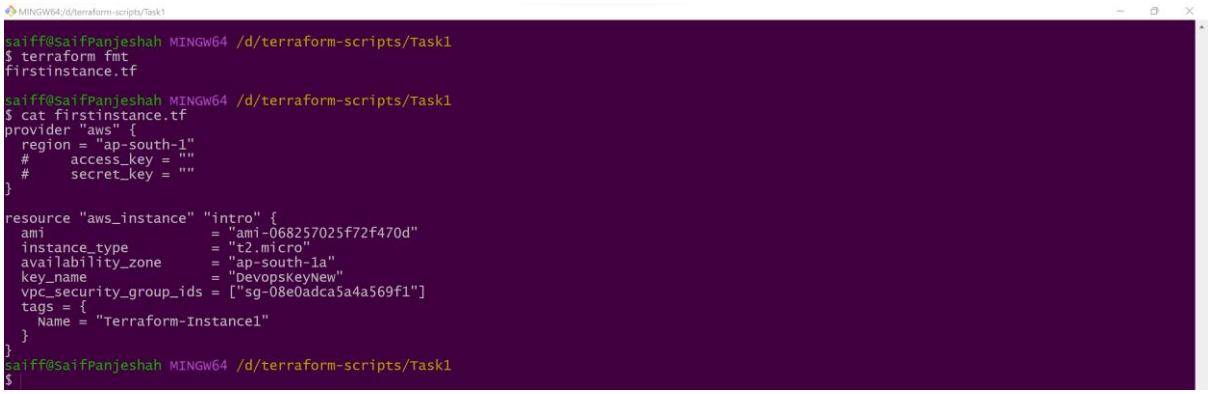


X

Saiffaizal Panjesha

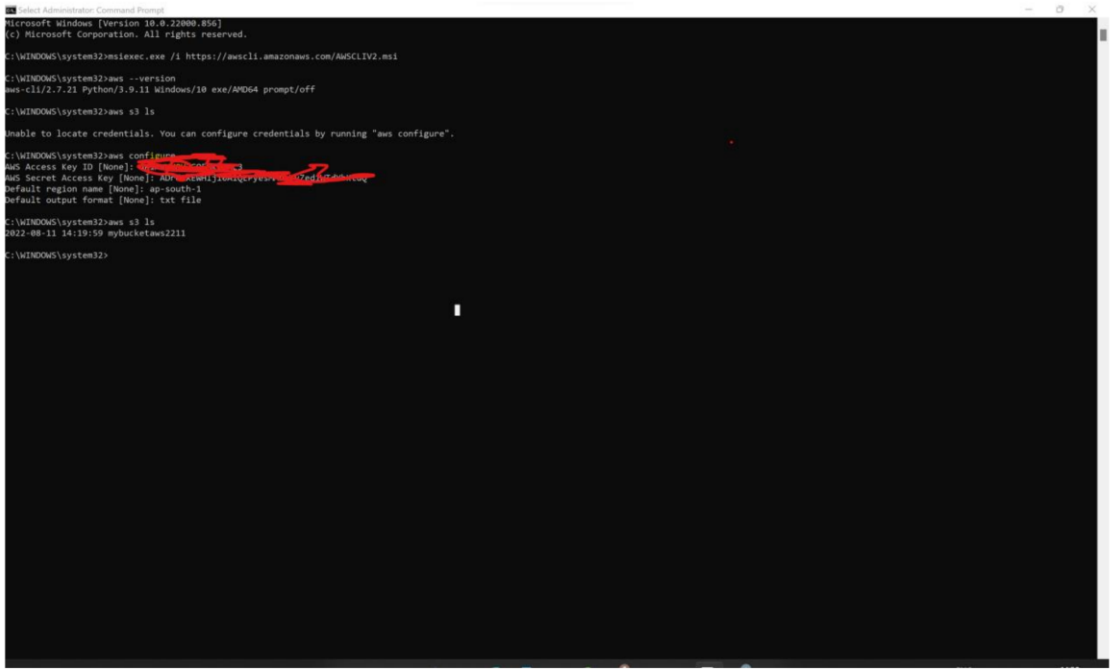
Technical Consultant

Format Terraform Code in canonical Style



Error:

Resolving Error:

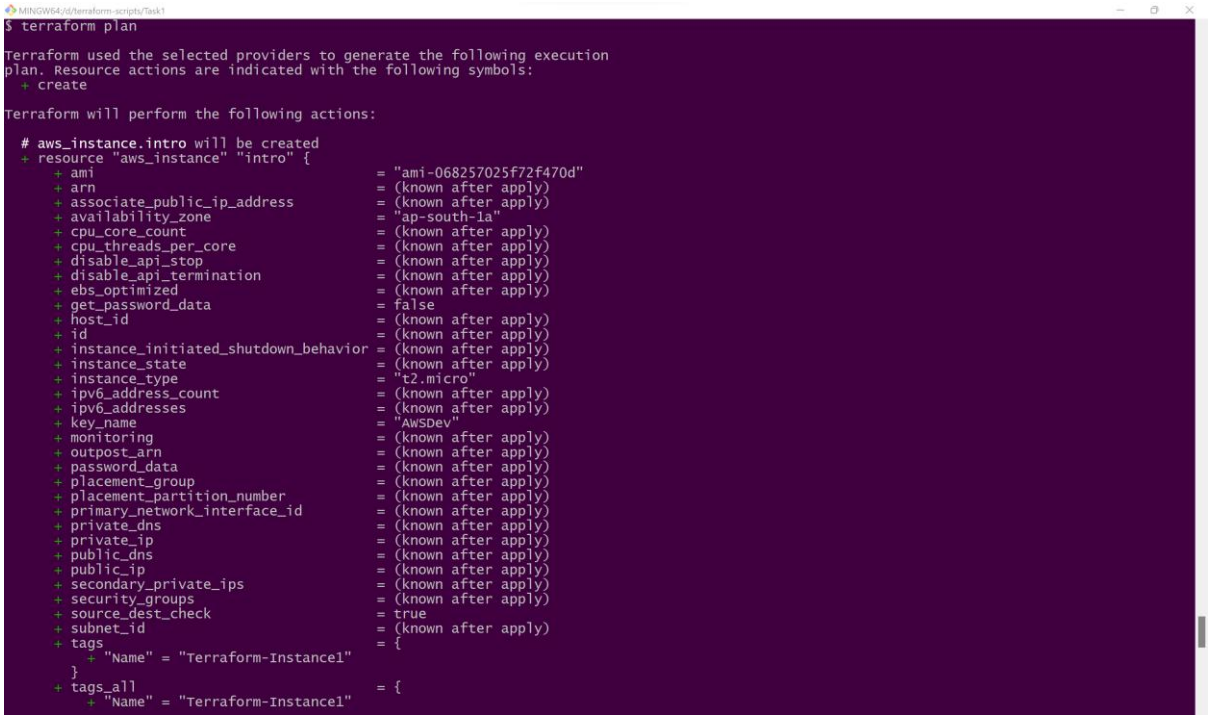


X

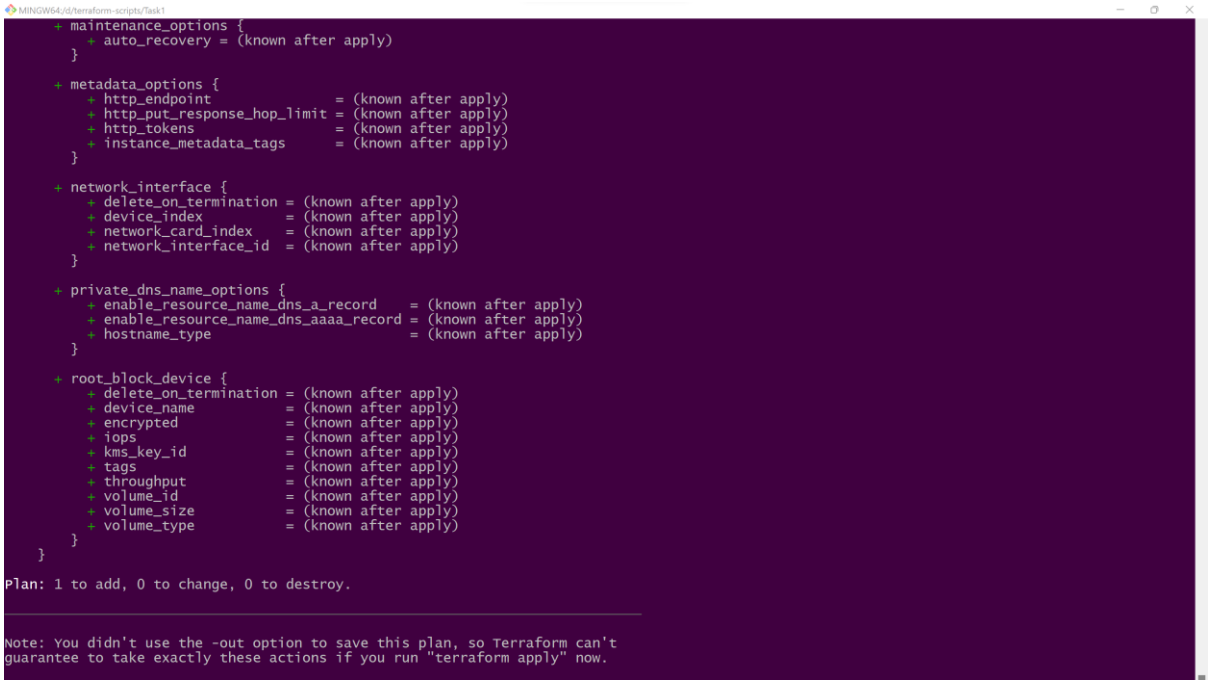
Saiffaizal Panjesha

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"
```

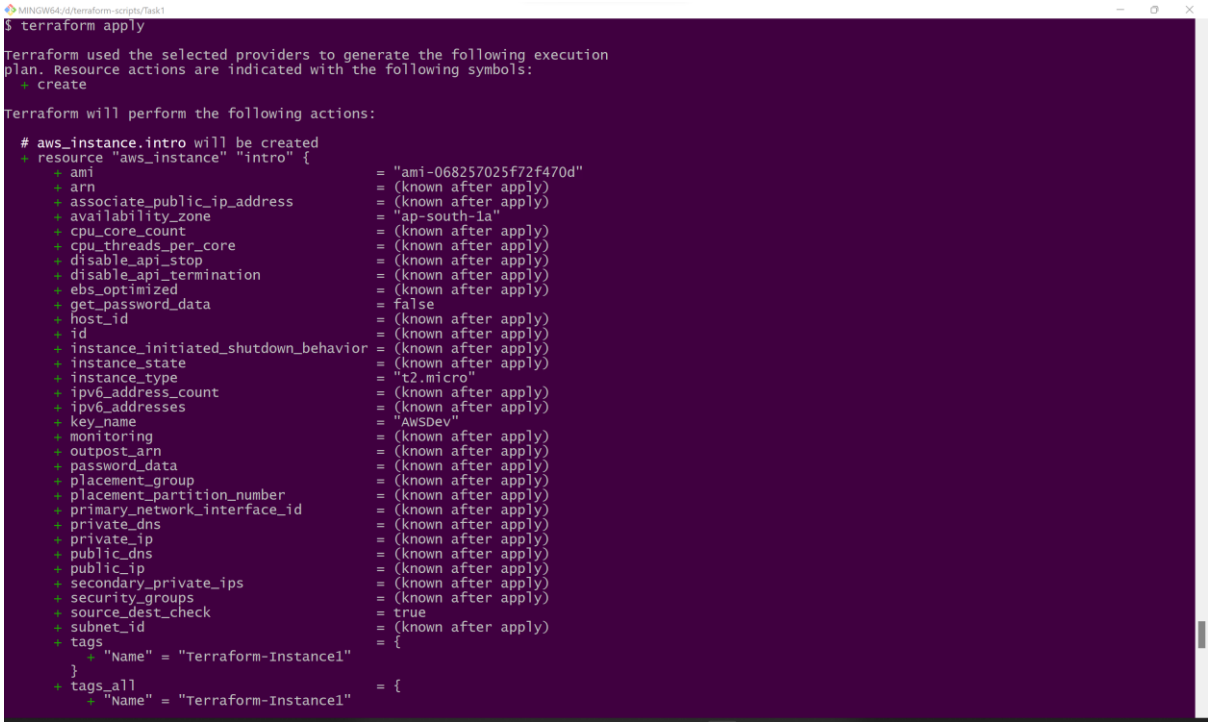


```
MINGW64:/d/terraform-scripts/Task1
      + maintenance_options {
          + auto_recovery = (known after apply)
        }

      + 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.
```

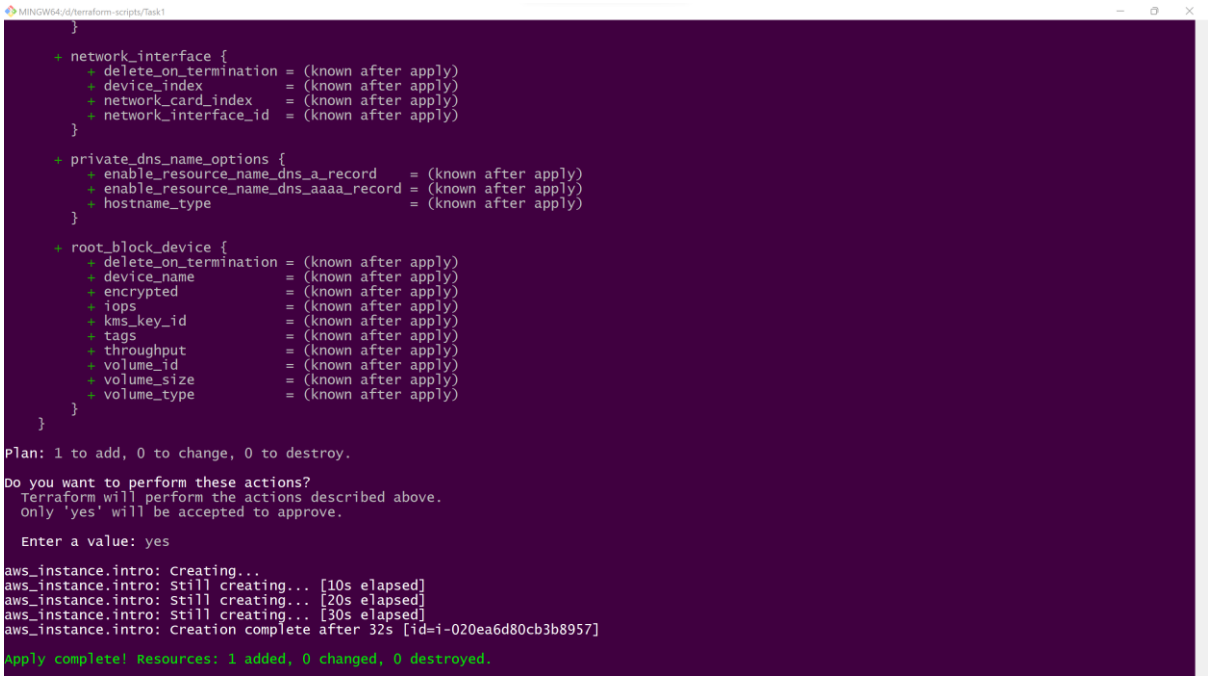
X

Saiffaizal Panjesha

Technical Consultant



```
MINGW64:/d/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"
```

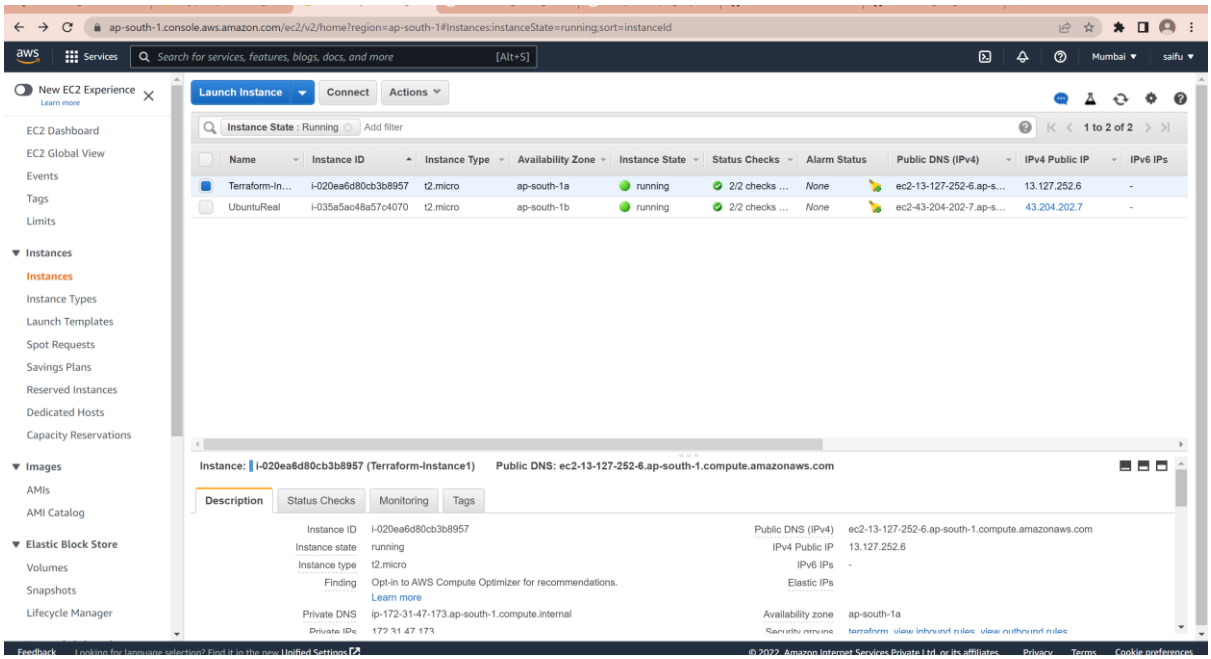


```
MINGW64:/d/terraform-scripts/Task1
        }

      + 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.
```

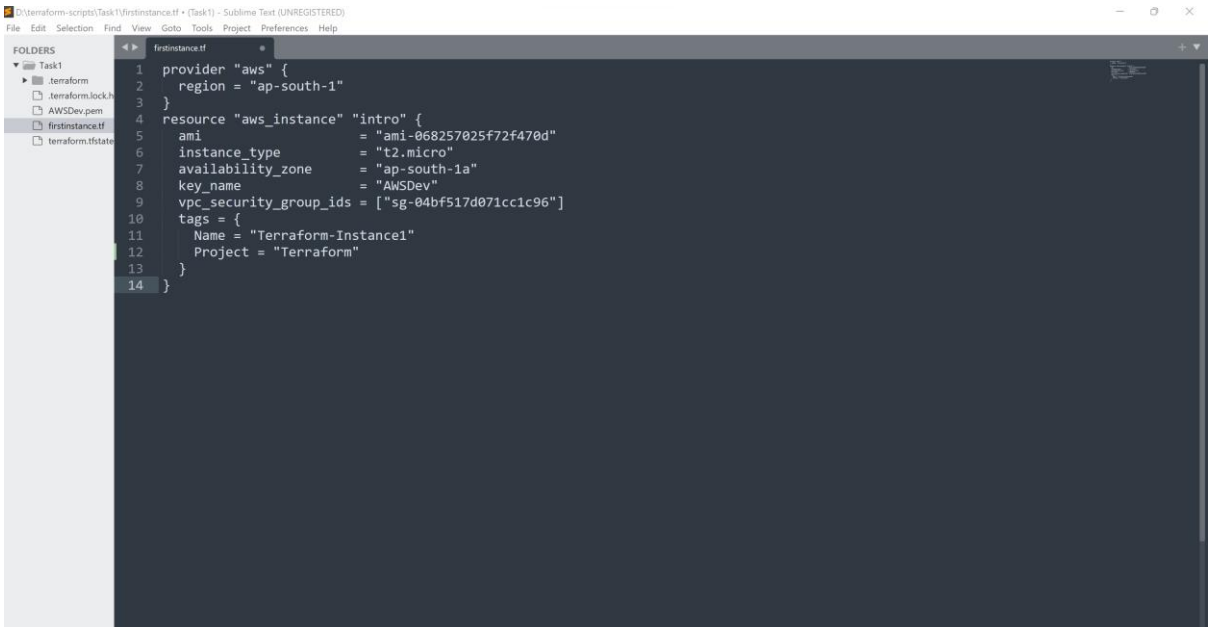
X

Saiffaizal Panjesha

Technical Consultant



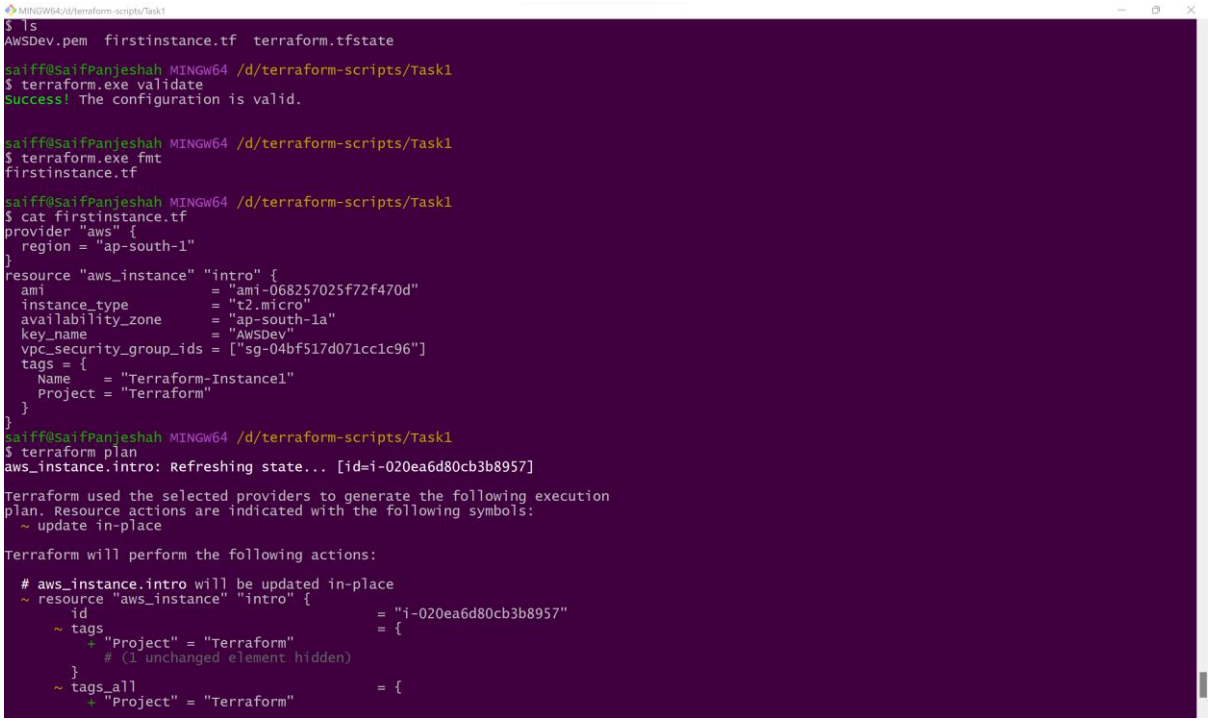
Let’s Make Some Changes I have projects tag in File



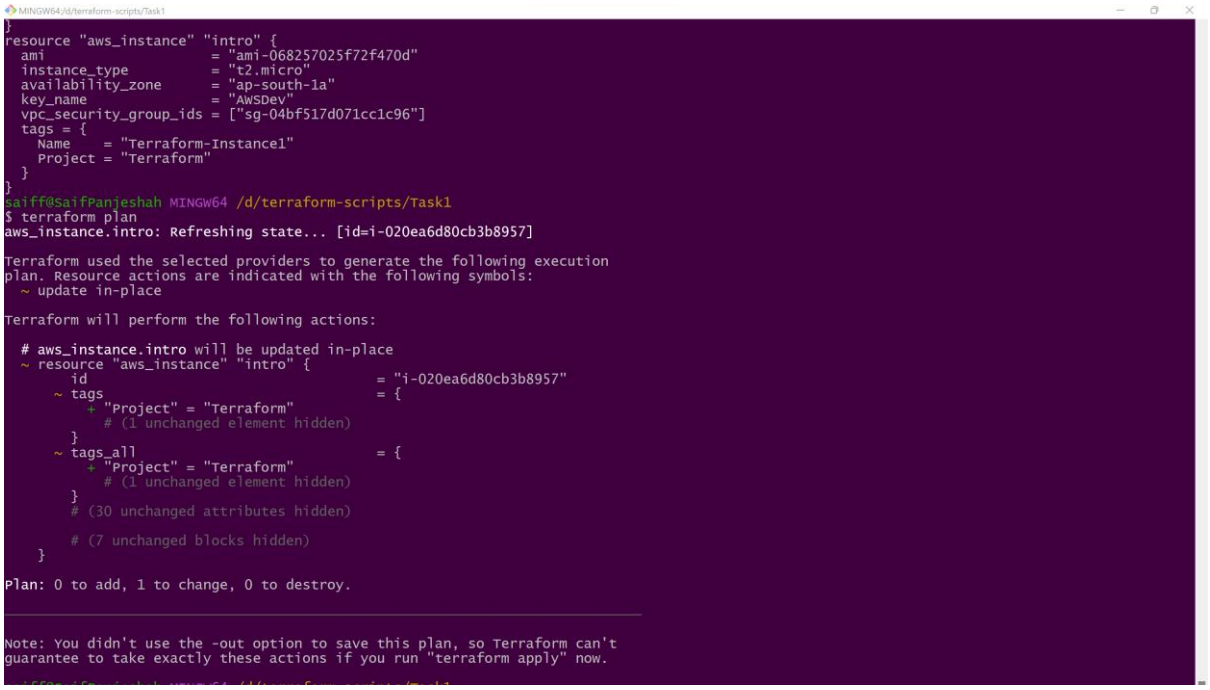
X

Saiffaizal Panjesha

Technical Consultant



```
$ ls
AWSDev.pem  firstinstance.tf  terraform.tfstate

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


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

saiff@SaifPanjeshah 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@SaifPanjeshah 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"
```



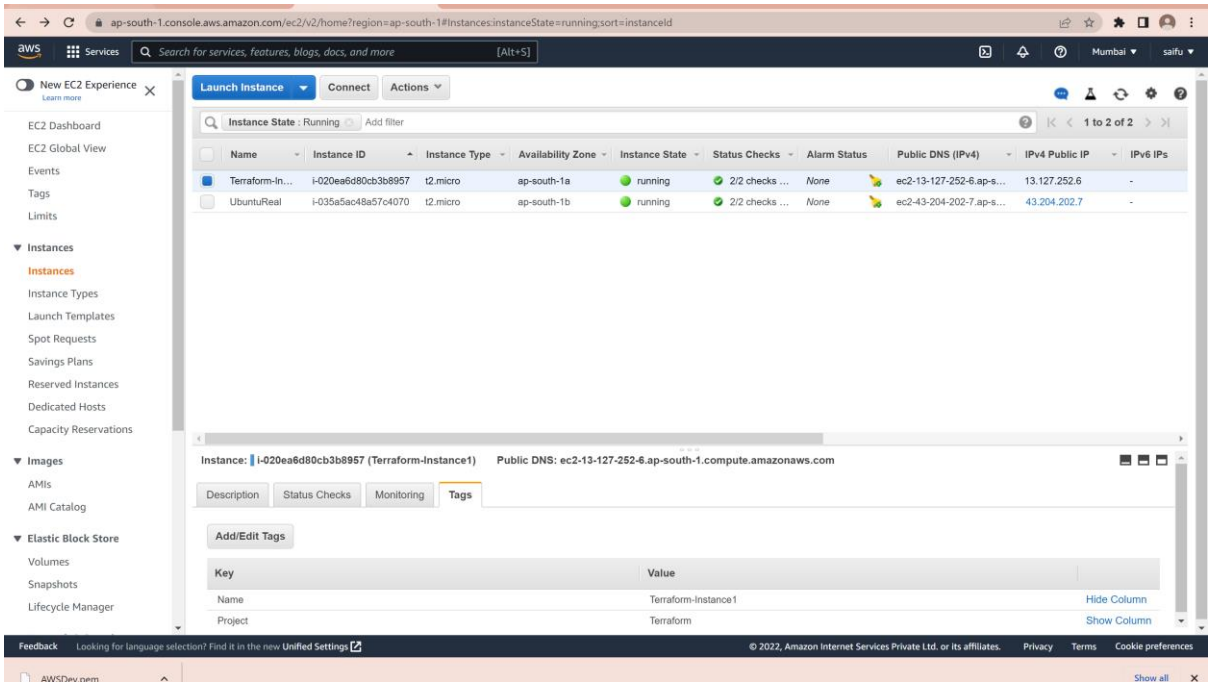
```
}
resource "aws_instance" "intro" {
  ami                    = "ami-068257025f72f470d"
  instance_type          = "t2.micro"
  availability_zone      = "ap-south-1a"
  key_name               = "AWSDev"
  vpc_security_group_ids = ["sg-04bf517d071cc1c96"]
  tags = {
    Name    = "Terraform-Instance1"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

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
$
```



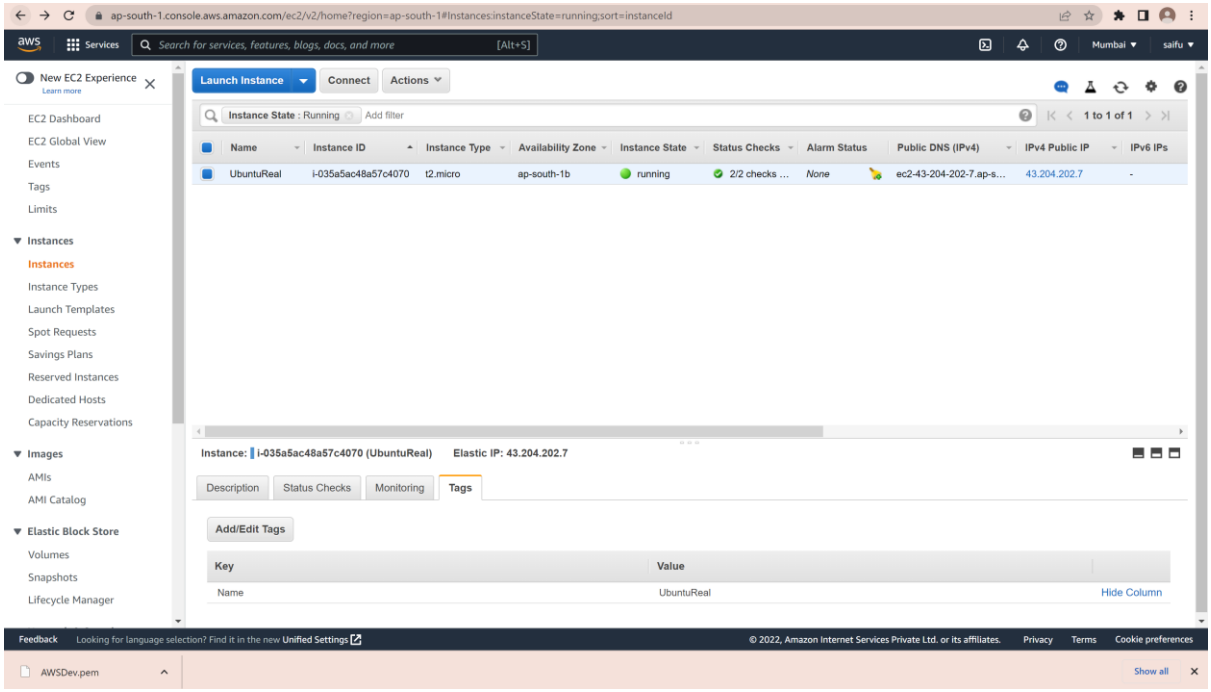
X

Saiffaizal Panjesha

Technical Consultant

Terraform Destroy:

```
saiff@SaifPanjeshah MINGW64 /d/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
      - arn                                  = "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                             = "AWSDev" -> 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.

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



X

Saiffaizal Panjesha

Technical Consultant

X

Saiffaizal Panjesha

Technical Consultant