

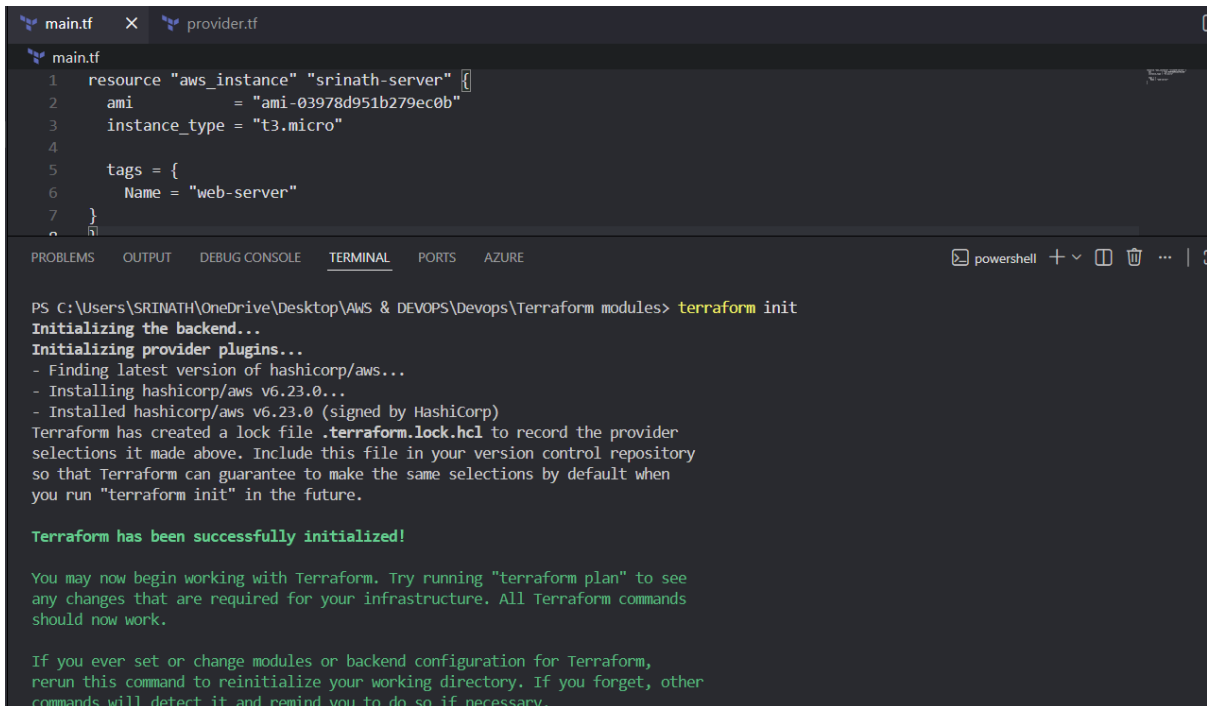
## Task 6

### Terraform:

- Terraform is an open-source infrastructure as code (IaC) tool developed by HashiCorp that allows users to define and provision cloud and on-premises resources using a declarative configuration language.
- It enables users to manage infrastructure throughout its lifecycle by writing human-readable configuration files that can be versioned, reused, and shared.

### Terraform Command:

1) terraform init:



The screenshot shows a VS Code editor with two files open: `main.tf` and `provider.tf`. The `main.tf` file contains the following Terraform configuration:

```
1 resource "aws_instance" "srinath-server" {
2   ami           = "ami-03978d951b279ec0b"
3   instance_type = "t3.micro"
4
5   tags = {
6     Name = "web-server"
7   }
8 }
```

The terminal window at the bottom shows the output of the `terraform init` command:

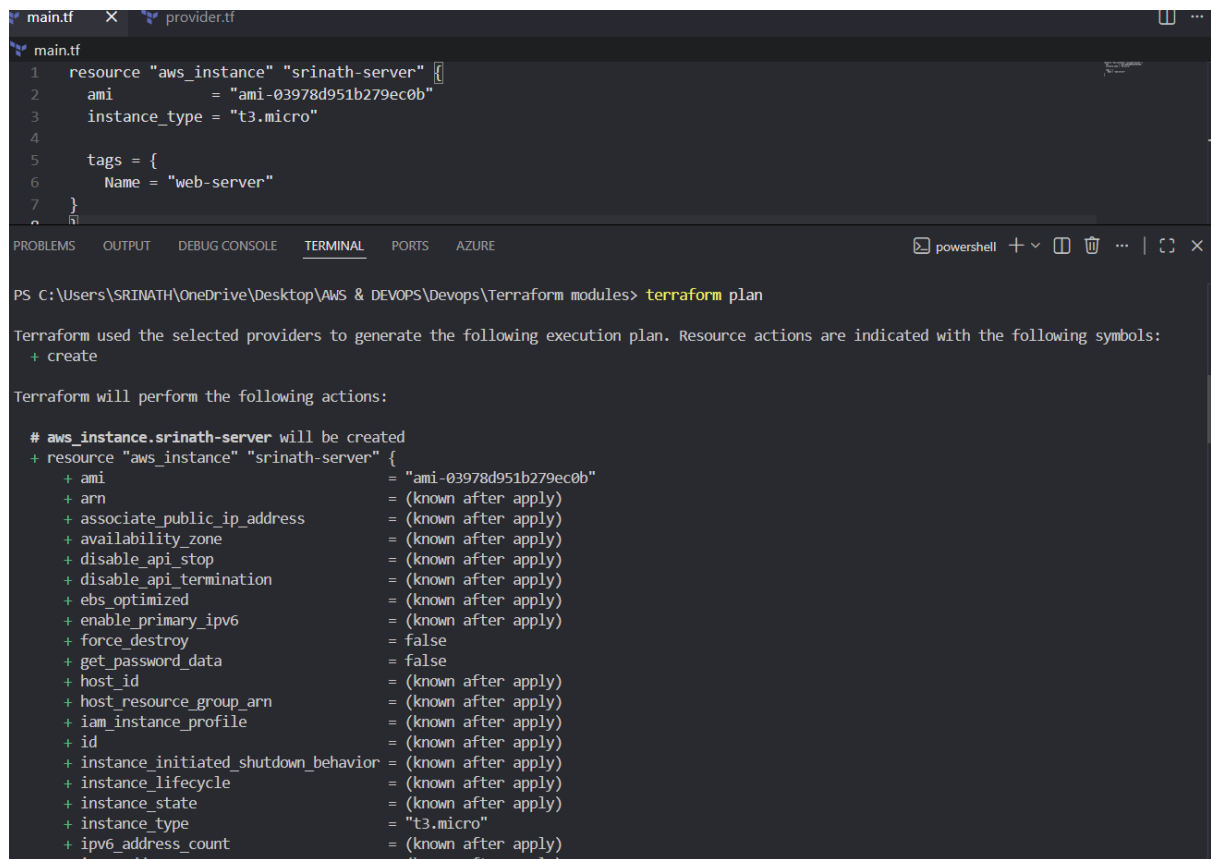
```
PS C:\Users\SRINATH\OneDrive\Desktop\AWS & DEVOPS\Devops\Terraform modules> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.23.0...
- Installed hashicorp/aws v6.23.0 (signed by HashiCorp)
Terraform 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.
```

## 2) terraform plan:



The screenshot shows a VS Code editor with two tabs: `main.tf` and `provider.tf`. The `main.tf` file contains the following Terraform configuration:

```
1 resource "aws_instance" "srinath-server" {
2     ami           = "ami-03978d951b279ec0b"
3     instance_type = "t3.micro"
4
5     tags = {
6         Name = "web-server"
7     }
8 }
```

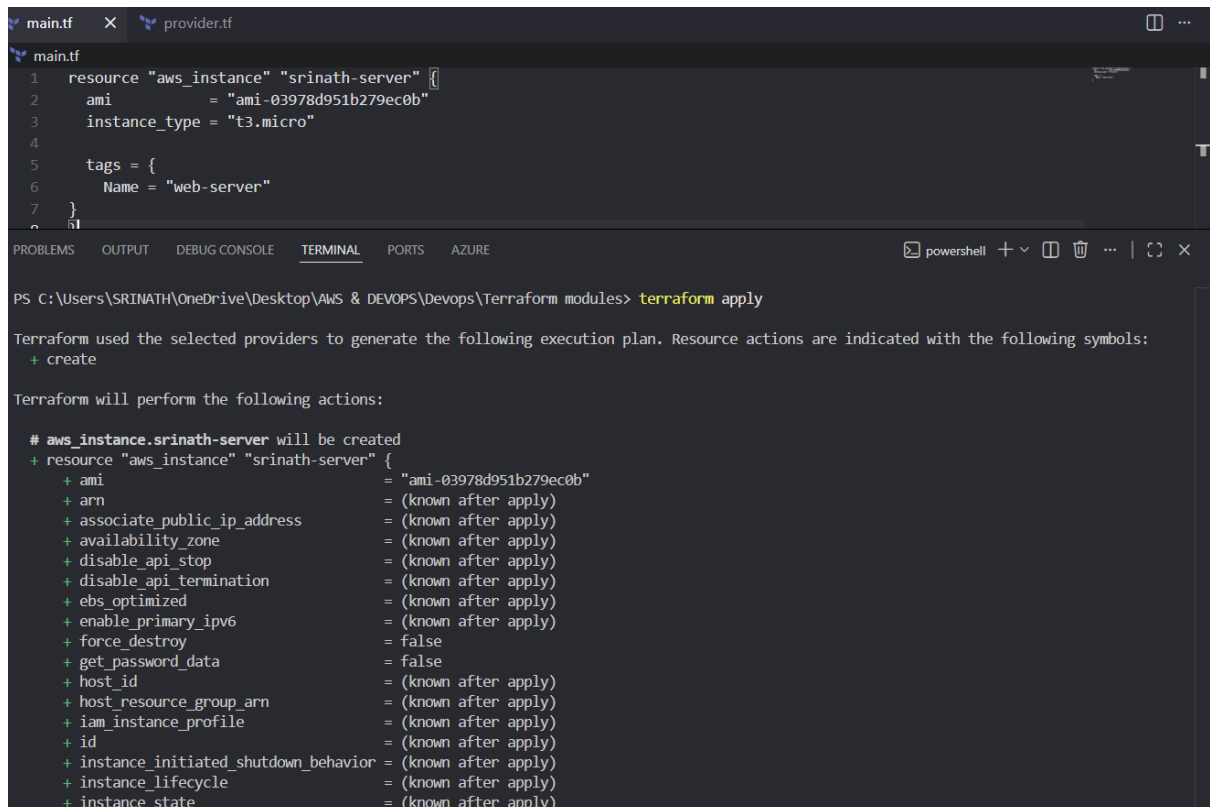
The terminal window shows the command `terraform plan` being executed. The output indicates that Terraform will create the `aws_instance.srinath-server` resource. The plan lists the following actions:

- `+ create`

The plan also lists the following attributes for the resource:

- `ami`: `"ami-03978d951b279ec0b"`
- `arn`: `(known after apply)`
- `associate_public_ip_address`: `(known after apply)`
- `availability_zone`: `(known after apply)`
- `disable_api_stop`: `(known after apply)`
- `disable_api_termination`: `(known after apply)`
- `ebs_optimized`: `(known after apply)`
- `enable_primary_ipv6`: `(known after apply)`
- `force_destroy`: `false`
- `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`: `"t3.micro"`
- `ipv6_address_count`: `(known after apply)`

## 3) terraform apply



The screenshot shows a VS Code editor with two tabs: `main.tf` and `provider.tf`. The `main.tf` file contains the following Terraform configuration:

```
1 resource "aws_instance" "srinath-server" {
2     ami           = "ami-03978d951b279ec0b"
3     instance_type = "t3.micro"
4
5     tags = {
6         Name = "web-server"
7     }
8 }
```

The terminal window shows the command `terraform apply` being executed. The output indicates that Terraform will create the `aws_instance.srinath-server` resource. The plan lists the following actions:

- `+ create`

The plan also lists the following attributes for the resource:

- `ami`: `"ami-03978d951b279ec0b"`
- `arn`: `(known after apply)`
- `associate_public_ip_address`: `(known after apply)`
- `availability_zone`: `(known after apply)`
- `disable_api_stop`: `(known after apply)`
- `disable_api_termination`: `(known after apply)`
- `ebs_optimized`: `(known after apply)`
- `enable_primary_ipv6`: `(known after apply)`
- `force_destroy`: `false`
- `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)`

# Output

