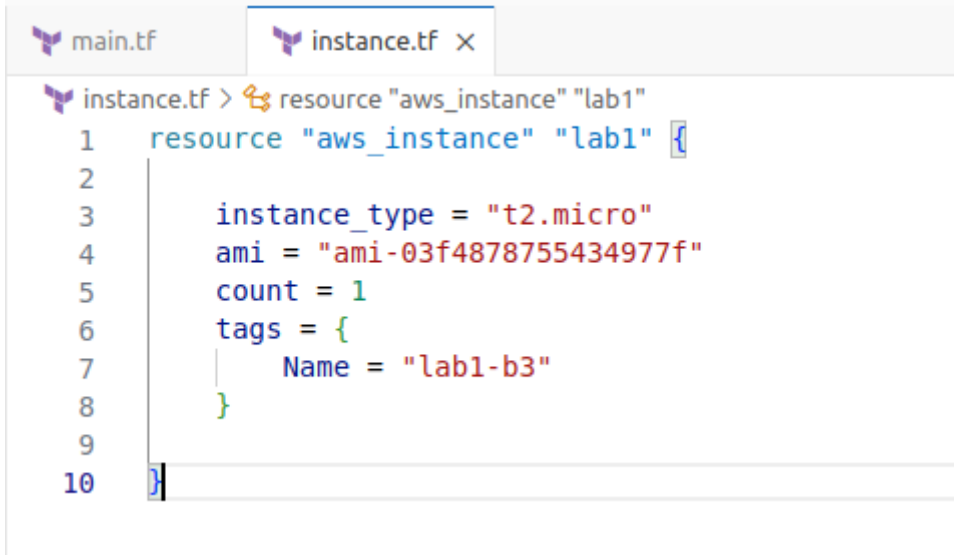


# LAB-3

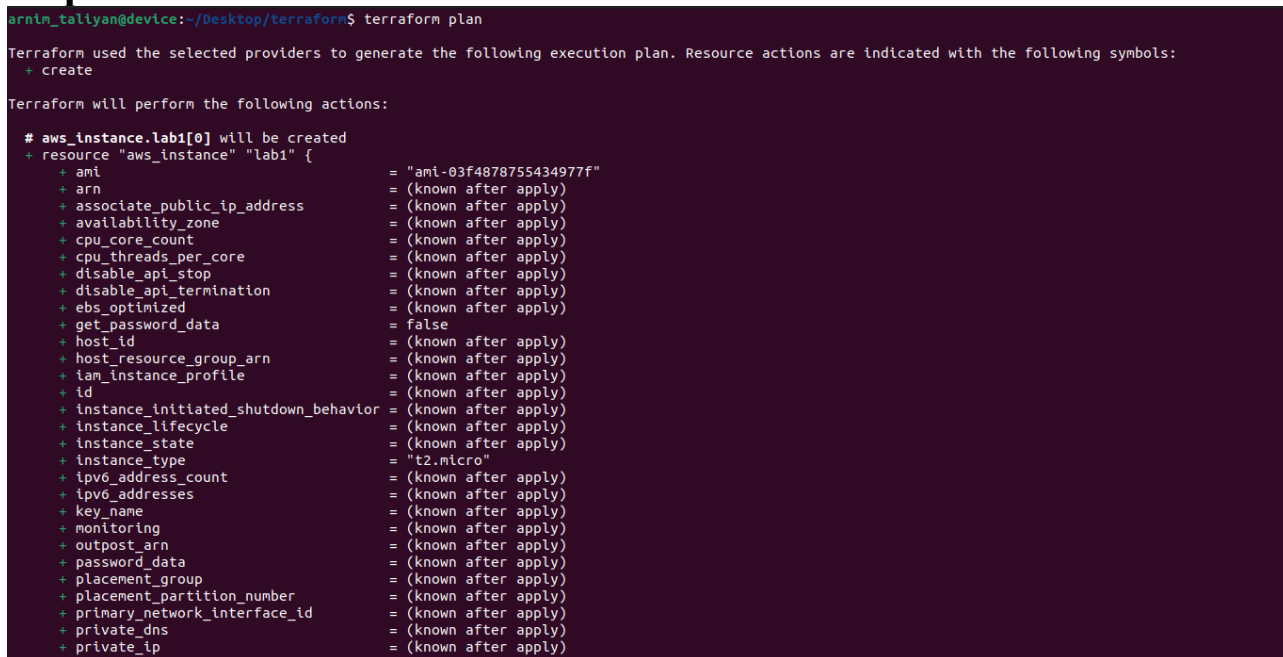
## Provisioning on EC2 Instance on AWS

### Step 1: Create Terraform configuration file for EC2 instance



```
main.tf  instance.tf x
instance.tf > resource "aws_instance" "lab1"
1  resource "aws_instance" "lab1" {
2
3      instance_type = "t2.micro"
4      ami = "ami-03f4878755434977f"
5      count = 1
6      tags = {
7          Name = "lab1-b3"
8      }
9
10 }
```

### Step 2: Review Plan



```
arnin_talayan@device:~/desktop/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.lab1[0] will be created
+ resource "aws_instance" "lab1" {
  + ami                    = "ami-03f4878755434977f"
  + 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_ip             = (known after apply)
  + subnet_id             = (known after apply)
  + tags                  = {
    + Name = "lab1-b3"
  }
  + timeouts              = {}
}
```

## Step 3: Apply Changes

```
arnin_taliyan@device:~/desktop/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.lab1[0] will be created
+ resource "aws_instance" "lab1" {
  + ami                     = "ami-03f4878755434977f"
  + 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)
  + private_ip_prefix_id    = (known after apply)
  + subnet_id               = (known after apply)
  + tags                    = {}
  + vpc_security_group_ids = (known after apply)
}
```

## Step 4: Verify Resources

The screenshot shows the AWS Management Console interface. On the left, the navigation menu includes 'EC2 Dashboard', 'EC2 Global View', 'Events', and 'Instances'. The 'Instances' section is expanded, showing 'Instances', 'Instance Types', and 'Launch Templates'. The main content area is titled 'Instances (1) Info' and includes a search bar and a filter 'Instance state = running'. Below this, a table lists the instance details:

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
<input type="checkbox"/>	lab1-b3	i-06f2bd3067651f62c	Running	t2.micro	Initializing	<a href="#">View alarms</a>	ap-south-1b

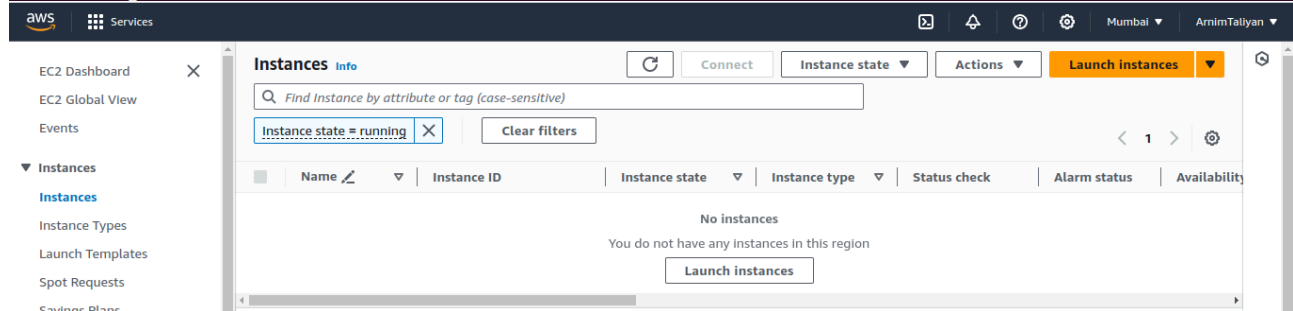
## Step 5: Cleanup Resources

```
arnin_taliyan@device:~/desktop/terraform$ terraform destroy
aws_instance.lab1[0]: Refreshing state... [id=i-06f2bd3067651f62c]

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.lab1[0] will be destroyed
- resource "aws_instance" "lab1" {
  ami                  = "ami-03f4878755434977f" -> null
  arn                  = "arn:aws:ec2:ap-south-1:533266967718:instance/i-06f2bd3067651f62c" -> null
  associate_public_ip_address = false -> 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-06f2bd3067651f62c" -> null
  instance_initiated_shutdown_behavior = "stop" -> null
  instance_state        = "stopped" -> null
  instance_type         = "t2.micro" -> null
  ipv6_address_count     = 0 -> null
  ipv6_addresses        = [] -> null
  monitoring            = false -> null
  placement_partition_number = 0 -> null
  primary_network_interface_id = "eni-02e1ba629a249df88" -> null
  private_dns            = "ip-172-31-1-105.ap-south-1.compute.internal" -> null
  private_ip            = "172.31.1.105" -> null
  secondary_private_ips  = [] -> null
  security_groups        = [
    - "default",
  ] -> null
  source_dest_check      = true -> null
  subnet_id              = "subnet-05d27f498c7f6a158" -> null
}
```



The screenshot shows the AWS Management Console interface. On the left is a navigation sidebar with options like 'EC2 Dashboard', 'EC2 Global View', 'Events', and 'Instances'. The 'Instances' section is expanded, showing 'Instances', 'Instance Types', 'Launch Templates', 'Spot Requests', and 'Savings Plans'. The main content area is titled 'Instances Info' and includes a search bar, a filter dropdown set to 'Instance state = running', and a 'Clear filters' button. Below this is a table with columns: Name, Instance ID, Instance state, Instance type, Status check, Alarm status, and Availability. The table is empty, displaying the message 'No instances' and 'You do not have any instances in this region'. A 'Launch instances' button is located at the bottom of the table. The top of the console shows the AWS logo, a 'Services' dropdown, and the region 'Mumbai' and user 'ArninTaliyan'.