**Name – Nandan Shailesh Kasat**

**Batch – Jan-Feb 2024**

**Terraform Task_2**

Q1. Install Terraform in local machine, configure AWS provider. Initialize Terraform configuration.

Create a VPC with CIDR block 192.168.0.0/16
Create two public subnets in different availability zones within the VPC.
Create an Internet Gateway and attach it to the VPC.
Create a route table and associate it with the public subnets, setting the default route to the Internet Gateway.
Create security groups allowing HTTP (port 80) and SSH (port 22) access.

Note: Provide access_key, secret_key and region, CIDR for vpc and subnets using input variable.

And display vpc-id, sg-id

Ans:

**1. Install Terraform and Configure AWS Provider**:

**2. Create a VPC**:

Define your VPC resource in Terraform. Specify the CIDR block (e.g., 192.168.0.0/16) for your VPC.

```
 GNU nano 7.2                                                    main.tf
terraform {
  required_version = "~> 1.1"
  required_providers {
    aws = {
      version = "~>3.1"
    }
  }
}
provider "aws" {
  access_key = var.access_key
  secret_key = var.secret_key
  region = var.region_name
}

resource "aws_instance" "myec2" {
  ami                    = var.ami_id
  instance_type          = "t2.micro"
  vpc_security_group_ids = [aws_security_group.mysg.id]
  key_name= "tf-key-pair"
  tags = {
    Name = "terraform1"
  }
}
resource "aws_vpc" "my_vpc" {
  cidr_block = "192.168.0.0/16"
}
```

## 3. Create Public Subnets:

Create two public subnets in different availability zones (AZs) within your VPC.

```
 GNU nano 7.2                                                    main.tf
resource "aws_subnet" "public_subnet1" {
  vpc_id     = aws_vpc.my_vpc.id
  cidr_block = "192.168.1.0/24"
  availability_zone = "ap-southeast-1"
}

resource "aws_subnet" "public_subnet2" {
  vpc_id     = aws_vpc.my_vpc.id
  cidr_block = "192.168.2.0/24"
  availability_zone = "ap-southeast-2"
}
```

## 4. Create an Internet Gateway (IGW):

Deploy an IGW and associate it with your VPC to enable internet traffic.

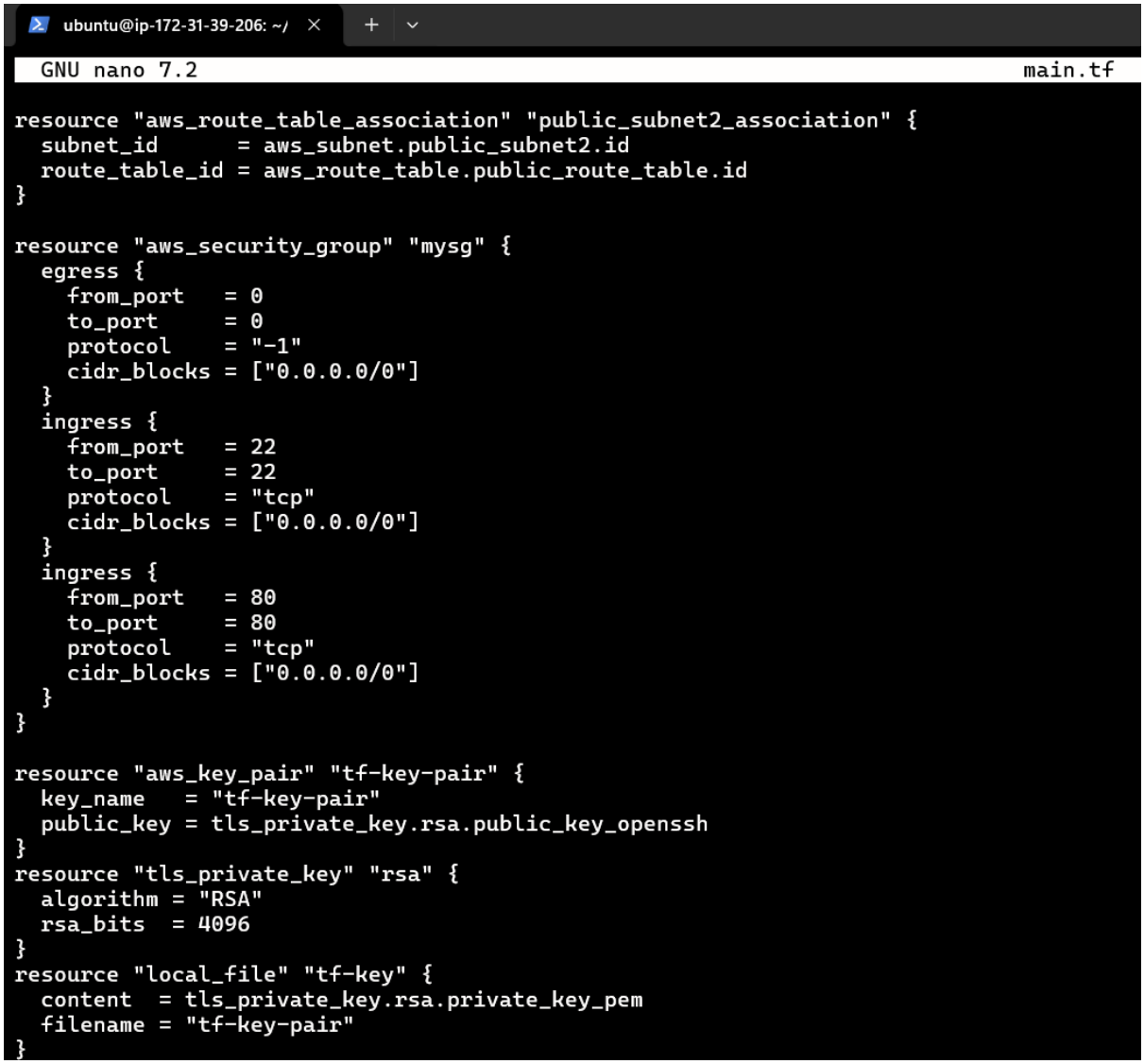
```
 GNU nano 7.2                                                    main.tf
resource "aws_internet_gateway" "my_igw" {
  vpc_id = aws_vpc.my_vpc.id
}
```

## 5. Create a Route Table:

Define a route table for your VPC. Associate the public subnets with this route table. Set the default route to the IGW.


## 6. Create Security Groups:

Define security groups allowing HTTP (port 80) and SSH (port 22) access.

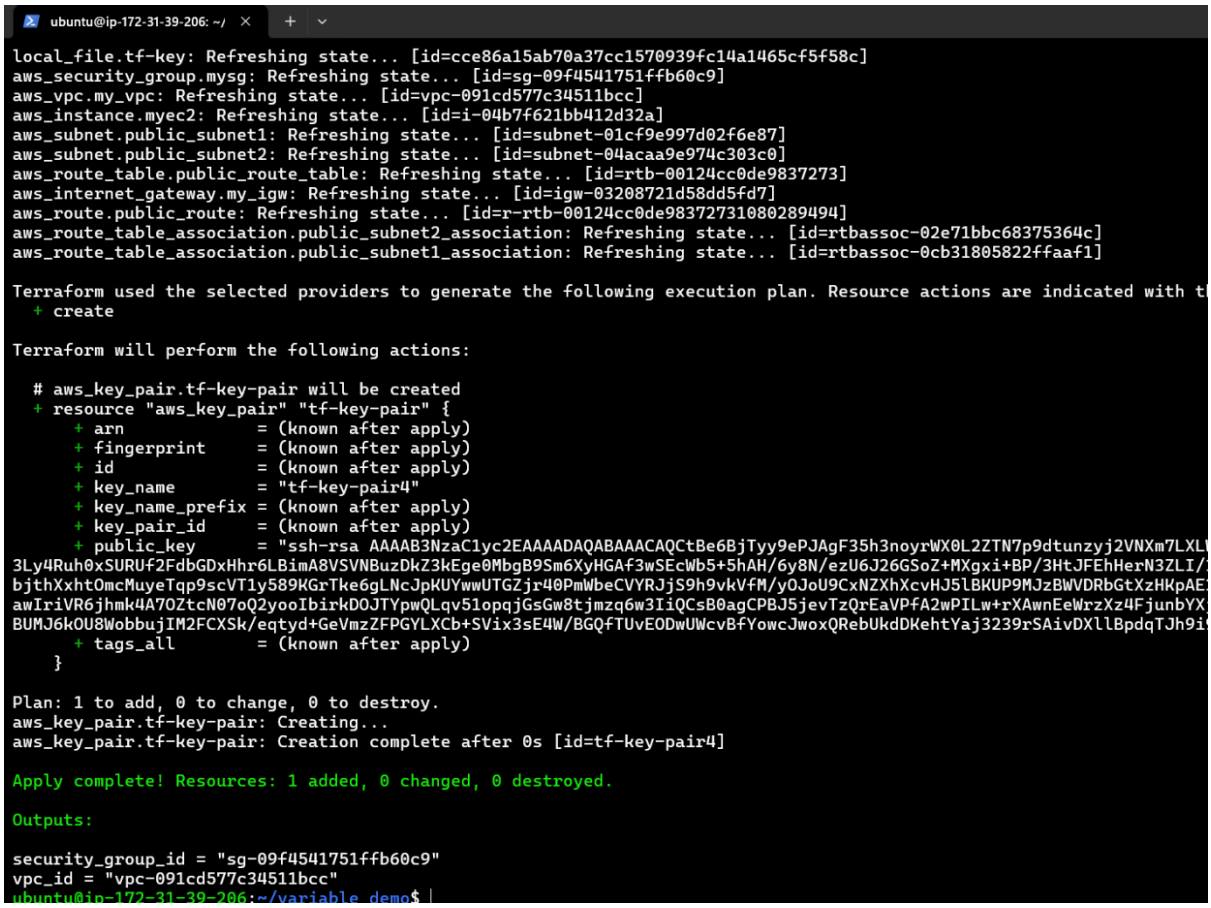



In Terraform.tfvars mention the Access Key, Secret Key and ami_id.


## 7. Display VPC ID and Security Group ID:

After applying your Terraform configuration, you can retrieve the VPC ID and security group ID using Terraform outputs.


## 8. Final Output:

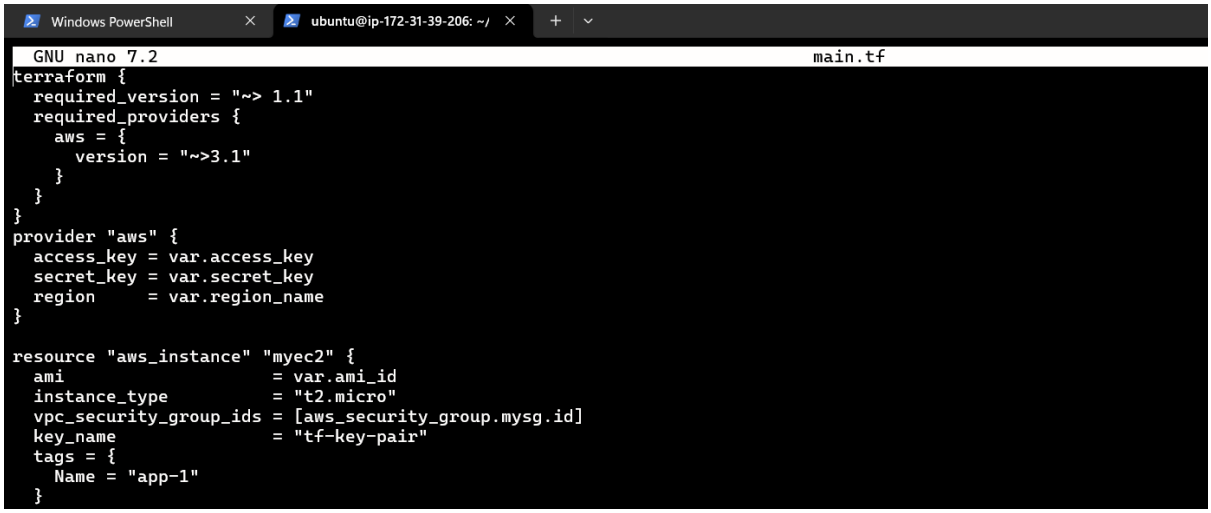


Q2. Launch an EC2 instances with names “app-1” and install apache, create two pages at its default location using provisioner block. Display webpages on browser.

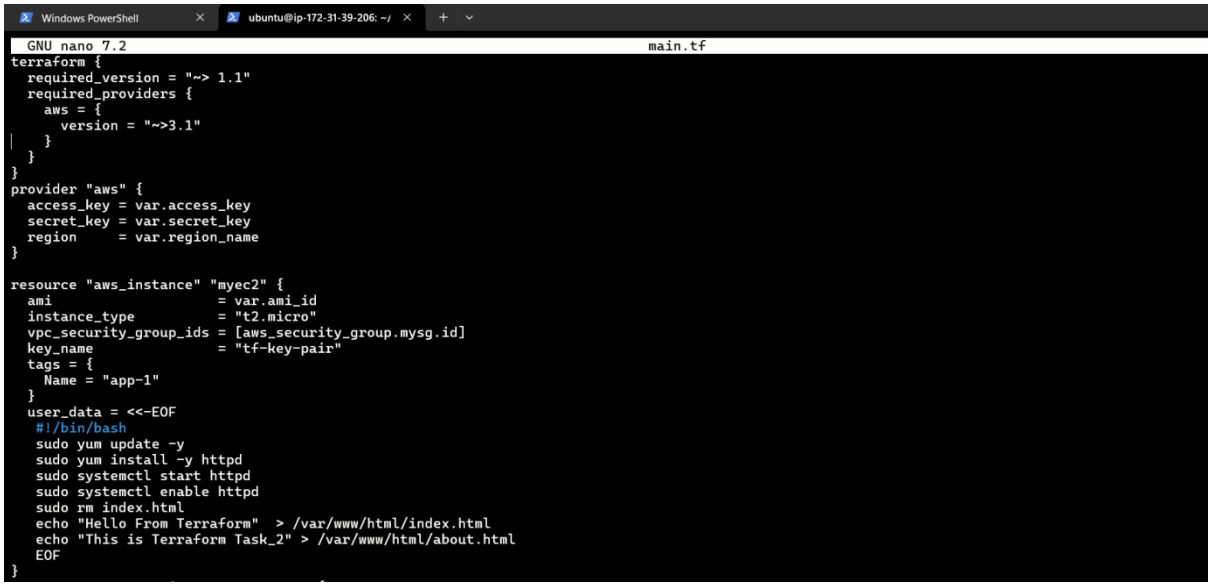
Ans:

## 1. Launch an EC2 instance name “app_1”

Define your EC2 instance resource in Terraform. Set the instance name to “app-1” and choose an appropriate instance type (e.g., t2.micro).




## 2. Install Apache:

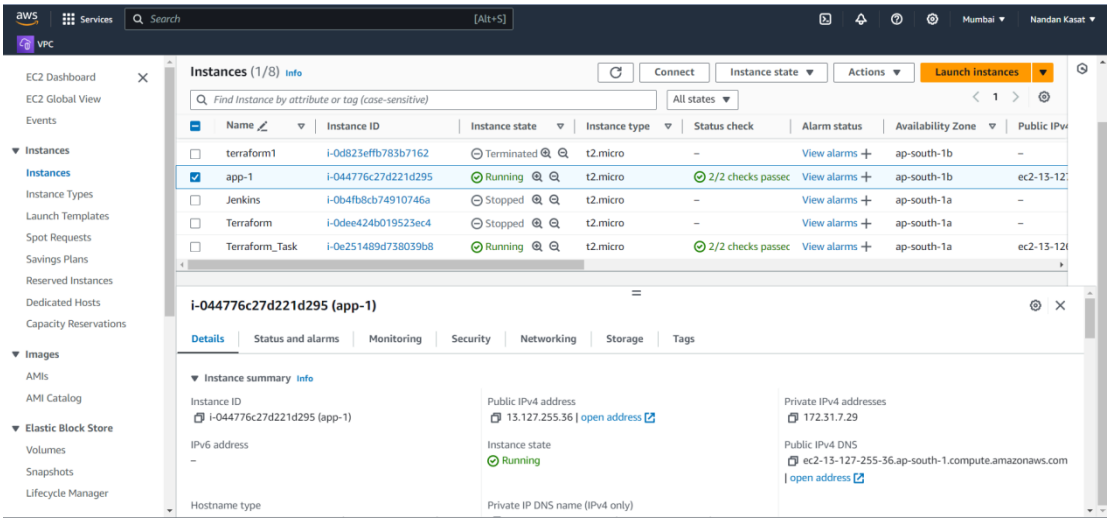


## 3. Create Web Pages:

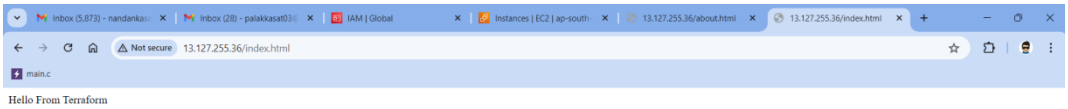
After installing Apache, create two HTML pages (e.g., index.html and about.html) in the default web server location (/var/www/html).

```
    echo "Hello From Terraform"  > /var/www/html/index.html
    echo "This is Terraform Task_2" > /var/www/html/about.html
    EOF
}
```

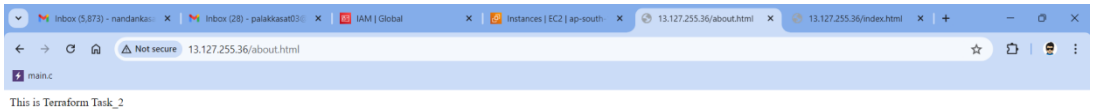
## 4. You can see instance app_1 on dashboard.




## 5. Index.html




## 6. About.html



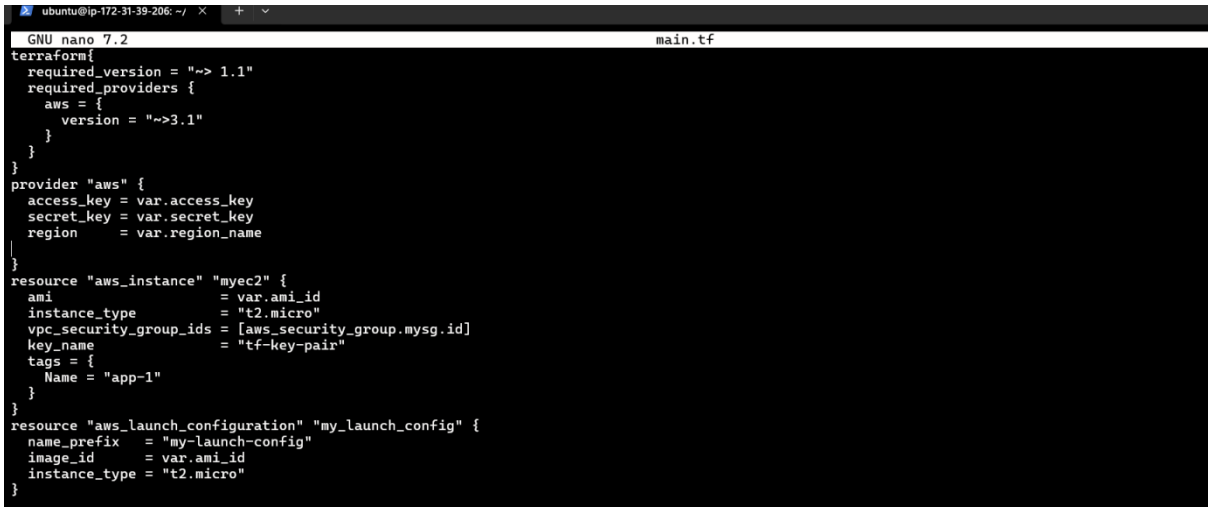
Q3. Create an Auto Scaling Group with a Launch Configuration to manage the EC2 instances, using Teeraform.

Ans:

## 1. Start the Instance:

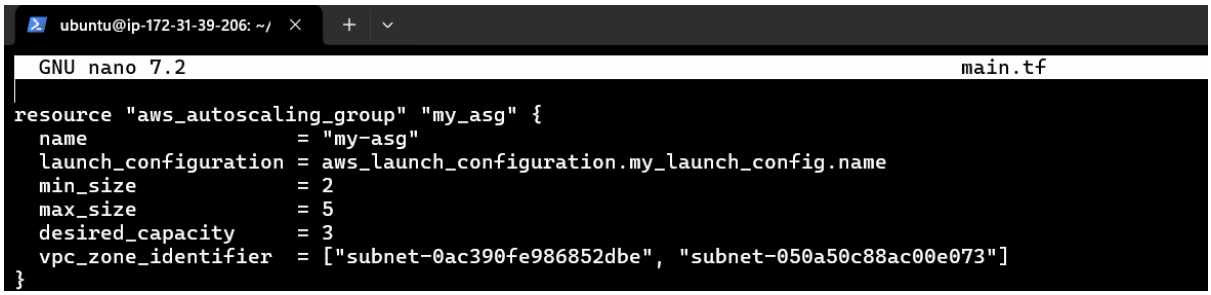
## 2. Define a Launch Configuration:

First, create a launch configuration that defines the instance specifications.




## 3. Create an Auto Scaling Group:

Define your ASG resource, referencing the launch template.



## 4. Scaling Policies:

We can define scaling policies based on metrics like CPU utilization or custom metrics.
Attach these policies to your ASG.

```
resource "aws_autoscaling_policy" "scale_up_policy" {
  name                   = "scale-up-policy"
  autoscaling_group_name = aws_autoscaling_group.my_asg.name
  adjustment_type        = "ChangeInCapacity"
  scaling_adjustment     = 1
}
```

## 5. The Auto Scaling Group is created on the Dashboard.

```
ubuntu@ip-172-31-39-206: ~/
      + health_check_type         = (known after apply)
      + id                        = (known after apply)
      + launch_configuration      = "my-launch-config20240705054307660800000002"
      + max_size                  = 5
      + metrics_granularity       = "1Minute"
      + min_size                  = 2
      + name                      = "my-asg"
      + name_prefix               = (known after apply)
      + protect_from_scale_in     = false
      + service_linked_role_arn   = (known after apply)
      + vpc_zone_identifier       = [
          + "subnet-050a50c88ac00e073",
          + "subnet-0ac390fe986852dbe",
        ]
      + wait_for_capacity_timeout = "10m"
    }

  # aws_autoscaling_policy.scale_up_policy will be created
  + resource "aws_autoscaling_policy" "scale_up_policy" {
      + adjustment_type         = "ChangeInCapacity"
      + arn                     = (known after apply)
      + autoscaling_group_name  = "my-asg"
      + id                      = (known after apply)
      + metric_aggregation_type = (known after apply)
      + name                    = "scale-up-policy"
      + policy_type             = "SimpleScaling"
      + scaling_adjustment      = 1
    }

Plan: 2 to add, 0 to change, 0 to destroy.
aws_autoscaling_group.my_asg: Creating...
aws_autoscaling_group.my_asg: Still creating... [10s elapsed]
aws_autoscaling_group.my_asg: Still creating... [20s elapsed]
aws_autoscaling_group.my_asg: Still creating... [30s elapsed]
aws_autoscaling_group.my_asg: Still creating... [40s elapsed]
aws_autoscaling_group.my_asg: Still creating... [50s elapsed]
aws_autoscaling_group.my_asg: Still creating... [1m0s elapsed]
aws_autoscaling_group.my_asg: Creation complete after 1m7s [id=my-asg]
aws_autoscaling_policy.scale_up_policy: Creating...
aws_autoscaling_policy.scale_up_policy: Creation complete after 0s [id=scale-up-policy]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
```