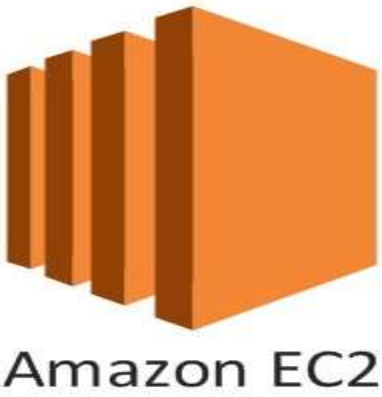
Terraform code Examples



Terraform Module for provisioning a general purpose EC2 host.

Format

```
resource "aws_instance"    "Instance-1" {

  ami                         = "……."
  instance_type               = "……."
  subnet_id                    = "……."
  vpc_security_group_ids = ["……."]
  key_name                     = "……."

tags = {
     Name              =    "……."
     OS                =     "……."
     App               =    "……."
     Environment"      =    "……."
  }
}
```

1) To create EC2 instance

vi test1.tf

```
provider "aws" {
  profile = "default"
  region  = "ap-south-1"
}

resource "aws_instance"  "instance-1" {
    ami                = "ami-04b1ddd35fd71475a"
    instance_type   = "t2.micro"
  }
```

2) To create multiple EC2 instances with different AMI

vi test2.tf

```
provider "aws" {
        profile = "default"
        region  = "ap-south-1"
}

resource "aws_instance"    "instance-1"   {
        ami                  = "ami-04b1ddd35fd71475a"
        instance_type    = "t2.micro"

  }

resource "aws_instance"  "instance-2" {
    ami                = "ami-0a9d27a9f4f5c0efc"
    instance_type    = "t2.micro"
  }
```

```
resource "aws_instance"  "instance-3" {
   ami                  = "ami-0a4a70bd98c6d6441"
   instance_type     = "t2.micro"
 }
```

3) To create multiple EC2 instance with same AMI

```
# vi  test3.tk

provider "aws" {
  profile      = "default"
  region       = "ap-south-1"
}

resource "aws_instance"  "instance-1" {
   ami                = "ami-04b1ddd35fd71475a"
   instance_type  = "t2.micro"
    count    =  3

 }
```

4) Create Security group and allow SSH and HTTP

```
# vi  test4.tk

provider "aws" {

    profile     = "default"

    region    = "ap-south-1"

}

resource "aws_security_group" "SG_1" {

  name        = "linux-sg-ssh-http"

  description = "Allow HTTP and SSH traffic"
```

```
  ingress {
    description = "SSH"
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  ingress {
    description = "HTTP"
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
   egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
}
```

5) Creating Keypair

a) First create private key in your system

vi test5.tk

- Ssh-keygen

  Enter key name: deepakawskey1

  Then press 3 times enter
- Dir or ls

Now create terraform file

- Nano test1.tf

```
provider "aws" {
  profile = "default"
  region  = "ap-south-1"
}
resource "aws_key_pair" "keypair-1" {
  key_name   = "deepakawskey1"
  public_key = file("deepakawskey1.pub")
}
```

6) Create instance with –Instance with default SG and Apache web service installed.

vi test6.tk

```
provider "aws" {
  profile = "default"
  region  = "ap-south-1"
}
resource "aws_instance"  "instance-1" {
    ami                = "ami-04b1ddd35fd71475a"
    instance_type  = "t2.micro"
     key_name       =  "deepakawskey1"
```

```
    count              =  1



  connection {
    type               = "ssh"
    host               = self.public_ip
    private_key        = file("deepakawskey1")
    user               = "ec2-user"
  }


 provisioner "remote-exec" {
    inline  =  [

      "sudo  yum install httpd  -y",

      "sudo  systemctl start httpd",
    ]
  }
}
```

7) Create instance with –Instance with default SG and Apache web service installed.

vi test7.tk

```
provider "aws" {
  profile = "default"
  region  = "ap-south-1"
}
resource "aws_instance"  "instance-1" {
    ami                    = "ami-04b1ddd35fd71475a"
    instance_type          = "t2.micro"
```

```
    key_name        =  "deepkey1"
    user_data       = <<-EOF


                #!/bin/bash
                sudo su
                yum install httpd -y
                echo "<html> <h1> Welcome to India </h1> </html>" >>
                /var/www/html/index.html
                sudo systemctl start httpd
                sudo systemctl enable httpd
                EOF
}
```

8) Create Instance with default SG and run apache web service script in new instance

a) Create scipt.sh file in current directory

```
#!/bin/bash
sudo su
yum install httpd -y
echo "<html> <h1> Welcome to India </h1> </html>" >>
/var/www/html/index.html
sudo systemctl start httpd
sudo systemctl enable httpd
```

b) # vi test7.sh

```
provider "aws" {
  profile = "default"
  region  = "ap-south-1"
}
resource "aws_instance"  "instance-1" {
```

```
    ami                     = "ami-04b1ddd35fd71475a"
    instance_type           = "t2.micro"
    key_name                =  "deepkey1"
    user_data               = "${file("script.sh")}"

  connection {
    type                = "ssh"
    host                = self.public_ip
    private_key         = file("deepkey1")
    user                = "ec2-user"
   }

  }
```

9) Create Load Balancer

```
  provider "aws" {
    profile = "default"
    region  = "ap-south-1"
  }

  resource "aws_elb" "elb1" {
    name               = "deepak-terraform-elb1"
    availability_zones = ["ap-south-1a", "ap-south-1b"]
    security_groups    = [ "sg-06629a21f198dc9f2"]
    instances          = [ "i-0d94c426466f0f6a8", "i-0c15facb3d9260613"]


    listener {
      instance_port     = 80
      instance_protocol = "http"
      lb_port           = 80
```

```
    lb_protocol             = "http"
  }

}
```

10) Create instance –Install docker –pull image and run container.

a) Create scipt.sh file in current directory

```
#!/bin/bash
#!/bin/bash
sudo su
yum install docker -y
systemctl start docker
systemctl enable docker
docker pull deepaksaidockerhub/dec
docker run -it -p 82:80 -d deepaksaidockerhub/dec
```

b) # vi test7.sh

```
provider "aws" {
  profile = "default"
  region  = "ap-south-1"
}
resource "aws_instance"  "instance-1" {
    ami                     = "ami-04b1ddd35fd71475a"
    instance_type           = "t2.micro"
    key_name                =  "deepkey1"
    user_data               = "${file("script.sh")}"
}
```