```
JENKINS FILE FOR DEPLOYMENT
pipeline{
    agent any
    stages{
        stage ("build"){
            steps {
                sh "echo 'building the project' "
                sh 'pip install flask'
     }

        }


    stage ("test"){
            steps {
                sh "echo 'testing the project' "
                sh "python test.py"


            }
        }

     stage ("run"){
            steps {
                sh "echo 'running the project' "
                sh "python  app.py"
            }
        }
    }
}
```

TERRAFORM FILE

Main.tf

```
resource "aws_vpc" "vpc_sre" {
    cidr_block = "10.0.0.0/16"
    enable_dns_hostnames = true
    instance_tenancy = "default"
    tags = {
      "Name" = "vpc_sre"
    }
  }


resource "aws_internet_gateway" "vpc_sre_igw" {
    vpc_id =  aws_vpc.vpc_sre.id

    tags = {
      Name = "internet-gate"
    }

}


resource "aws_subnet" "subnet_1" {                              #create a Subnet
resource
    vpc_id = aws_vpc.vpc_sre.id
    cidr_block = "10.0.0.0/24"
    map_public_ip_on_launch = true
```

```
    availability_zone = "us-west-1a"
    tags = {
      "key" = "Public Subnet",
      "Name" = "subnet_1"
    }
}


resource "aws_subnet" "subnet_2" {                              #create a Subnet
resource
    vpc_id = aws_vpc.vpc_sre.id
    cidr_block = "10.0.1.0/24"
    map_public_ip_on_launch = true
    availability_zone = "us-west-1c"
    tags = {
      "key" = "Public Subnet",
      "Name" = "subnet_2"
    }

}


#resource "aws_route_table" "my_vpc_us_west1_public"{
route table already defined below
 #   vpc_id = aws}



/*create our instance to test everything by declaring aws_instance
resource "aws_instance" "instance1" {
# ami = "ami-0d9858aa3c6322f73"
```

```
instance_type = "t2.micro"

tags = {

Name = "city-serve"

}

}*/


#declaring aws_route_table and aws_route_table_association resources

resource "aws_route_table" "vpc_sre_public_tbl" {
#give the resource(aws_route_table) and add a name(any) to the resource

vpc_id = aws_vpc.vpc_sre.id


route {

    cidr_block = "0.0.0.0/0"

    gateway_id = aws_internet_gateway.vpc_sre_igw.id

}

tags = {

  Name = "Public Subnet Route Table."

}


}



resource "aws_route_table_association" "vpc_sre_rt_tbl_assoc_1" {              #
creating aws_route_table_association resource & gave it the name
"my_vpc_us_west1_public"

subnet_id = aws_subnet.subnet_1.id

route_table_id = aws_route_table.vpc_sre_public_tbl.id


}
```

```
resource "aws_route_table_association" "vpc_sre_rt_tbl_assoc_2" {              #
creating aws_route_table_association resource & gave it the name
"my_vpc_us_west1_public"

subnet_id = aws_subnet.subnet_2.id

route_table_id = aws_route_table.vpc_sre_public_tbl.id


}

#add Security Group by adding aws_security_group resource

/* we're allowing incoming SSH connections (22/tcp) from any addresses
(0.0.0.0/0) inside the Security Group,

and also we're allowing any connection initiation to the outside world from the
Security Group.

So, we'll be able to SSH to the instance protected by this Security Group and
make any connections from it.*/

resource "aws_security_group" "allow_ssh" {

  name        = "allow_ssh_sg"

  description = "Allow SSH inbound connections"

  vpc_id = aws_vpc.vpc_sre.id


  ingress {

    from_port   = 22

    to_port     = 22

    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"]
  }


  tags = {
    Name = "allow_ssh_sg"
  }
}


resource "aws_security_group" "allow_http" {
  name        = "allow_http_sg"
  description = "Allow HTTP inbound connections"
  vpc_id = aws_vpc.vpc_sre.id


  ingress {
    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"]
  }


  tags = {
    Name = "allow_Http_sg"
  }
}
```