Main.tf

provider "aws" {

region = "ap-south-1"

}

Vpc.tf

# Creating VPC

resource "aws\_vpc" "demovpc" {

cidr\_block = "${var.vpc\_cidr}"

instance\_tenancy = "default"

tags = {

Name = "Demo VPC"

}

}

Subnet.tf

# Creating 1st web subnet

resource "aws\_subnet" "public-subnet-1" {

vpc\_id = "${aws\_vpc.demovpc.id}"

cidr\_block = "${var.subnet1\_cidr}"

map\_public\_ip\_on\_launch = true

availability\_zone = "ap-south-1a"

tags = {

Name = "Web Subnet 1"

}

}

# Creating 2nd web subnet

resource "aws\_subnet" "public-subnet-2" {

vpc\_id = "${aws\_vpc.demovpc.id}"

cidr\_block = "${var.subnet2\_cidr}"

map\_public\_ip\_on\_launch = true

availability\_zone = "ap-south-1b"

tags = {

Name = "Web Subnet 2"

}

}

# Creating 1st application subnet

resource "aws\_subnet" "application-subnet-1" {

vpc\_id = "${aws\_vpc.demovpc.id}"

cidr\_block = "${var.subnet3\_cidr}"

map\_public\_ip\_on\_launch = false

availability\_zone = "ap-south-1a"

tags = {

Name = "Application Subnet 1"

}

}

# Creating 2nd application subnet

resource "aws\_subnet" "application-subnet-2" {

vpc\_id = "${aws\_vpc.demovpc.id}"

cidr\_block = "${var.subnet4\_cidr}"

map\_public\_ip\_on\_launch = false

availability\_zone = "ap-south-1b"

tags = {

Name = "Application Subnet 2"

}

}

# Create Database Private Subnet

resource "aws\_subnet" "private-subnet-5" {

vpc\_id = "${aws\_vpc.demovpc.id}"

cidr\_block = "${var.subnet5\_cidr}"

availability\_zone = "ap-south-1a"

tags = {

Name = "Database Subnet 1"

}

}

# Create Database Private Subnet

resource "aws\_subnet" "private-subnet-6" {

vpc\_id = "${aws\_vpc.demovpc.id}"

cidr\_block = "${var.subnet6\_cidr}"

availability\_zone = "ap-south-1b"

tags = {

Name = "Database Subnet 2"

}

}

Vars.tf

# Defining CIDR Block for VPC

variable "vpc\_cidr" {

default = "10.0.0.0/16"

}

# Defining CIDR Block for 1st Subnet

variable "subnet1\_cidr" {

default = "10.0.1.0/24"

}

# Defining CIDR Block for 2nd Subnet

variable "subnet2\_cidr" {

default = "10.0.2.0/24"

}

# Defining CIDR Block for 3rd Subnet

variable "subnet3\_cidr" {

default = "10.0.3.0/24"

}

# Defining CIDR Block for 4th Subnet

variable "subnet4\_cidr" {

default = "10.0.4.0/24"

}

# Defining CIDR Block for 5th Subnet

variable "subnet5\_cidr" {

default = "10.0.5.0/24"

}

# Defining CIDR Block for 6th Subnet

variable "subnet6\_cidr" {

default = "10.0.6.0/24"

}

Route.tf

# Creating Route Table

resource "aws\_route\_table" "route" {

vpc\_id = "${aws\_vpc.demovpc.id}"

route {

cidr\_block = "0.0.0.0/0"

gateway\_id = "${aws\_internet\_gateway.demogateway.id}"

}

tags = {

Name = "Route to internet"

}

}

# Associating Route Table

resource "aws\_route\_table\_association" "rt1" {

subnet\_id = "${aws\_subnet.public-subnet-1.id}"

route\_table\_id = "${aws\_route\_table.route.id}"

}

# Associating Route Table

resource "aws\_route\_table\_association" "rt2" {

subnet\_id = "${aws\_subnet.public-subnet-2.id}"

route\_table\_id = "${aws\_route\_table.route.id}"

}

Igw.tf

# Creating Internet Gateway

resource "aws\_internet\_gateway" "demogateway" {

vpc\_id = "${aws\_vpc.demovpc.id}"

}

Ec2.tf

resource "aws\_instance" "demoinstance" {

ami = "ami-0e742cca61fb65051"

instance\_type = "t2.micro"

key\_name = "project"

vpc\_security\_group\_ids = ["${aws\_security\_group.demosg.id}"]

subnet\_id = "${aws\_subnet.public-subnet-1.id}"

associate\_public\_ip\_address = true

user\_data = "${file("data.sh")}"

tags = {

Name = "My Public Instance"

}

}

# Creating 2nd EC2 instance in Public Subnet

resource "aws\_instance" "demoinstance1" {

ami = "ami-0e742cca61fb65051"

instance\_type = "t2.micro"

key\_name = "project"

vpc\_security\_group\_ids = ["${aws\_security\_group.demosg.id}"]

subnet\_id = "${aws\_subnet.public-subnet-2.id}"

associate\_public\_ip\_address = true

user\_data = "${file("data.sh")}"

tags = {

Name = "My Public Instance 2"

}

}

# Creating 3nd EC2 instance in Private Subnet

resource "aws\_instance" "datainstance" {

ami = "ami-0e742cca61fb65051"

instance\_type = "t2.micro"

key\_name = "project"

vpc\_security\_group\_ids = ["${aws\_security\_group.demosg.id}"]

subnet\_id = "${aws\_subnet.private-subnet-5.id}"

associate\_public\_ip\_address = false

tags = {

Name = "My Private Instance"

}

}

# Creating 4nd EC2 instance in Private Subnet

resource "aws\_instance" "datainstance1" {

ami = "ami-0e742cca61fb65051"

instance\_type = "t2.micro"

key\_name = "project"

vpc\_security\_group\_ids = ["${aws\_security\_group.demosg.id}"]

subnet\_id = "${aws\_subnet.private-subnet-6.id}"

associate\_public\_ip\_address = false

tags = {

Name = "My Private Instance"

}

}

Websg.tf

# Creating Security Group

resource "aws\_security\_group" "demosg" {

vpc\_id = "${aws\_vpc.demovpc.id}"

# Inbound Rules

# HTTP access from anywhere

ingress {

from\_port = 80

to\_port = 80

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

# HTTPS access from anywhere

ingress {

from\_port = 443

to\_port = 443

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

# SSH access from anywhere

ingress {

from\_port = 22

to\_port = 22

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

# Outbound Rules

# Internet access to anywhere

egress {

from\_port = 0

to\_port = 0

protocol = "-1"

cidr\_blocks = ["0.0.0.0/0"]

}

tags = {

Name = "Web SG"

}

}

Databasesg.tf

# Create Database Security Group

resource "aws\_security\_group" "database-sg" {

name = "Database SG"

description = "Allow inbound traffic from application layer"

vpc\_id = aws\_vpc.demovpc.id

ingress {

description = "Allow traffic from application layer"

from\_port = 3306

to\_port = 3306

protocol = "tcp"

security\_groups = [aws\_security\_group.demosg.id]

}

egress {

from\_port = 32768

to\_port = 65535

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

tags = {

Name = "Database SG"

}

}

Data.sh

#!/bin/bash

yum update -y

yum install -y httpd.x86\_64

systemctl start httpd.service

systemctl enable httpd.service

echo "Hello World from $(hostname -f)" > /var/www/html/index.html

alb.tf

#!/bin/bash

yum update -y

yum install -y httpd.x86\_64

systemctl start httpd.service

systemctl enable httpd.service

echo "Hello World from $(hostname -f)" > /var/www/html/index.html

[root@ip-172-31-43-39 ec2-user]# ls

alb.tf databasesg.tf data.sh ec2.tf igw.tf main.tf rdf.tf route.tf subnet.tf terraform.tfstate terraform.tfstate.backup vars.tf vpc.tf websg.tf

[root@ip-172-31-43-39 ec2-user]# cat alb.tf

resource "aws\_lb" "external-alb" {

name = "External-LB"

internal = false

load\_balancer\_type = "application"

security\_groups = [aws\_security\_group.demosg.id]

subnets = [aws\_subnet.public-subnet-1.id, aws\_subnet.public-subnet-2.id]

}

resource "aws\_lb\_target\_group" "target-elb" {

name = "ALB-TG"

port = 80

protocol = "HTTP"

vpc\_id = aws\_vpc.demovpc.id

}

resource "aws\_lb\_target\_group\_attachment" "attachment" {

target\_group\_arn = aws\_lb\_target\_group.target-elb.arn

target\_id = aws\_instance.demoinstance.id

port = 80

depends\_on = [

aws\_instance.demoinstance,

]

}

resource "aws\_lb\_target\_group\_attachment" "attachment1" {

target\_group\_arn = aws\_lb\_target\_group.target-elb.arn

target\_id = aws\_instance.demoinstance1.id

port = 80

depends\_on = [

aws\_instance.demoinstance1,

]

}

resource "aws\_lb\_listener" "external-elb" {

load\_balancer\_arn = aws\_lb.external-alb.arn

port = "80"

protocol = "HTTP"

default\_action {

type = "forward"

target\_group\_arn = aws\_lb\_target\_group.target-elb.arn

}

}