**Terraform Architecture Task**

***Mohammed Rafiyoodin – 04-06-2024***

1. A Virtual Private Cloud (VPC)
2. Subnets (public and private)
3. An Internet Gateway
4. Route Tables and Routes
5. Security Groups
6. EC2 Instances
7. RDS Instance & Security group

**Main.tf**

provider "aws" {

  region = "us-east-1"

}

# Creating VPC

resource "aws\_vpc" "demovpc" {

  cidr\_block       = var.vpc\_cidr

  instance\_tenancy = "default"

  tags = {

    Name = "Demo VPC"

  }

}

# 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       = "us-east-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       = "us-east-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       = "us-east-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       = "us-east-1b"

  tags = {

    Name = "Application Subnet 2"

  }

}

# Create Database Private Subnet

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

  vpc\_id            = aws\_vpc.demovpc.id

  cidr\_block        = var.subnet5\_cidr

  availability\_zone = "us-east-1a"

  tags = {

    Name = "Database Subnet 1"

  }

}

# Create Database Private Subnet

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

  vpc\_id            = aws\_vpc.demovpc.id

  cidr\_block        = var.subnet6\_cidr

  availability\_zone = "us-east-1b"

  tags = {

    Name = "Database Subnet 1"

  }

}

# Creating Internet Gateway

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

  vpc\_id = aws\_vpc.demovpc.id

}

# 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

}

# Creating Security Group

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

  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"

  }

}

# Creating 1st EC2 instance in Public Subnet

resource "aws\_instance" "demoinstance" {

  ami           = "ami-00beae93a2d981137"

  instance\_type = "t2.micro"

  // count                       = 1

  key\_name                    = "jenkins0509"

  vpc\_security\_group\_ids      = ["${aws\_security\_group.terra-sg.id}"]

  subnet\_id                   = aws\_subnet.public-subnet-1.id

  associate\_public\_ip\_address = true

  tags = {

    Name = "My Public Instance"

  }

  user\_data = <<-EOF

    #!/bin/bash

    yum update -y

    yum install -y httpd

    systemctl start httpd

    systemctl enable httpd

  EOF

}

# Creating 2nd EC2 instance in Public Subnet

resource "aws\_instance" "demoinstance1" {

  ami           = "ami-00beae93a2d981137"

  instance\_type = "t2.micro"

  // count                       = 1

  key\_name                    = "jenkins0509"

  vpc\_security\_group\_ids      = ["${aws\_security\_group.terra-sg.id}"]

  subnet\_id                   = aws\_subnet.public-subnet-2.id

  associate\_public\_ip\_address = true

  tags = {

    Name = "My Public Instance 2"

  }

  user\_data = <<-EOF

    #!/bin/bash

    yum update -y

    yum install -y httpd

    systemctl start httpd

    systemctl enable httpd

  EOF

}

# Creating External LoadBalancer

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

  name               = "external-lb"

  internal           = false

  load\_balancer\_type = "application"

  security\_groups    = [aws\_security\_group.terra-sg.id]

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

}

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

  name     = "alb-tg8"

  port     = 80

  protocol = "HTTP"

  vpc\_id   = aws\_vpc.demovpc.id

}

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

  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" "attachment2" {

  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

}

}

output "lb\_dns\_name" {

  description = "The DNS name of the load balancer"

  value      = "${aws\_lb.external-alb.dns\_name}"

}

**Variable.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 = "10.0.0.0/24"

}

# Defining CIDR Block for 2nd Subnet

variable "subnet2\_cidr" {

  default = "10.0.1.0/24"

}

# Defining CIDR Block for 3rd Subnet

variable "subnet3\_cidr" {

  default = "10.0.2.0/24"

}

# Defining CIDR Block for 3rd Subnet

variable "subnet4\_cidr" {

  default = "10.0.3.0/24"

}

# Defining CIDR Block for 3rd Subnet

variable "subnet5\_cidr" {

  default = "10.0.4.0/24"

}

# Defining CIDR Block for 3rd Subnet

variable "subnet6\_cidr" {

  default = "10.0.5.0/24"

}

# 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.terra-sg.id]

}

egress {

  from\_port   = 32768

  to\_port     = 65535

  protocol    = "tcp"

  cidr\_blocks = ["0.0.0.0/0"]

}

tags = {

  Name = "Database SG"

}

}

# Creating RDS Instance

resource "aws\_db\_subnet\_group" "default" {

  name       = "main"

  subnet\_ids = [aws\_subnet.database-subnet-1.id, aws\_subnet.database-subnet-2.id]

tags = {

  Name = "My DB subnet group"

}

}

resource "aws\_db\_instance" "default" {

  allocated\_storage      = 10

  db\_subnet\_group\_name   = aws\_db\_subnet\_group.default.id

  engine                 = "mysql"

  engine\_version         = "8.0.35"

  instance\_class         = "db.m5d.large"

  multi\_az               = true

  identifier             = "mydb"

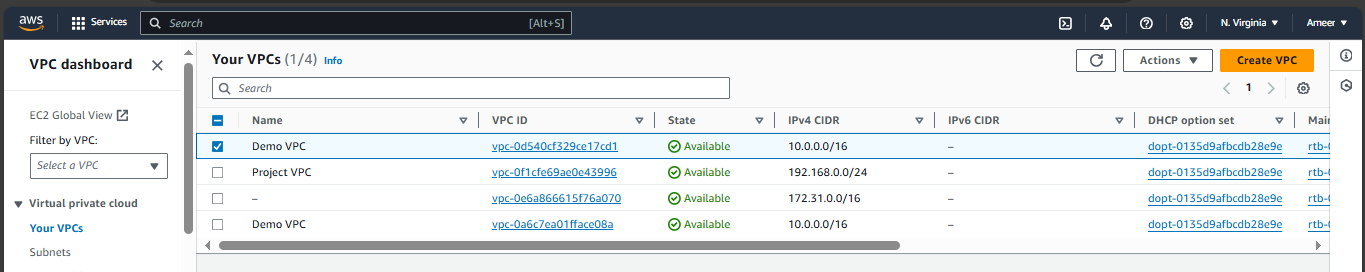
  username               = "admin"

  password               = "admin123456"

  skip\_final\_snapshot    = true

  vpc\_security\_group\_ids = [aws\_security\_group.database-sg.id]

}

****