01-VPC

Terraform VPC CLASS 7 CONFIGURATION!!!

```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
}
provider "aws" {
  region = "us-east-1"
}
resource "aws_vpc" "main_vpc" {
  cidr_block           = "10.10.0.0/16"
  enable_dns_hostnames = true
  enable_dns_support   = true
  tags = {
    Name = "Class7 VPC"
  }
}
resource "aws_security_group" "terraform_sg" {
  name_prefix = "terraform-sg-"
  description = "Terraform Security Group"

  # Allow inbound HTTP traffic
  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  # Allow inbound SSH traffic
  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"] # Restrict this in production!
  }

  # Allow all outbound traffic
  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
```

```
  }

  tags = {
    Name        = "terraform-security-group"
    Environment = "test"
  }
}
resource "aws_subnet" "public_subnet1" {
  vpc_id            = aws_vpc.main_vpc.id
  cidr_block        = "10.10.1.0/24"
  availability_zone = "us-east-1a"
}
resource "aws_subnet" "public_subnet2" {
  vpc_id            = aws_vpc.main_vpc.id
  cidr_block        = "10.10.2.0/24"
  availability_zone = "us-east-1b"
}
resource "aws_subnet" "public_subnet3" {
  vpc_id            = aws_vpc.main_vpc.id
  cidr_block        = "10.10.3.0/24"
  availability_zone = "us-east-1c"
}
resource "aws_subnet" "private_subnet1" {
  vpc_id            = aws_vpc.main_vpc.id
  cidr_block        = "10.10.11.0/24"
  availability_zone = "us-east-1a"
}
resource "aws_subnet" "private_subnet2" {
  vpc_id            = aws_vpc.main_vpc.id
  cidr_block        = "10.10.12.0/24"
  availability_zone = "us-east-1b"
}
resource "aws_subnet" "private_subnet3" {
  vpc_id            = aws_vpc.main_vpc.id
  cidr_block        = "10.10.13.0/24"
  availability_zone = "us-east-1c"
}
resource "aws_subnet" "private_subnet4" {
  vpc_id            = aws_vpc.main_vpc.id
  cidr_block        = "10.10.111.0/24"
  availability_zone = "us-east-1a"
}
resource "aws_subnet" "private_subnet5" {
  vpc_id            = aws_vpc.main_vpc.id
  cidr_block        = "10.10.112.0/24"
  availability_zone = "us-east-1b"
}
resource "aws_subnet" "private_subnet6" {
  vpc_id            = aws_vpc.main_vpc.id
  cidr_block        = "10.10.113.0/24"
```

```
  availability_zone = "us-east-1c"
}
resource "aws_internet_gateway" "igw" {
  vpc_id = aws_vpc.main_vpc.id
}
resource "aws_route_table" "public_rtb" {
  vpc_id = aws_vpc.main_vpc.id

  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.igw.id
  }
}
# Create an Elastic IP for the NAT Gateway
resource "aws_eip" "nat_eip" {
  domain = "vpc"
  tags = {
    Name = "nat-gateway-eip"
  }
}

# Create a NAT Gateway in public_subnet1
resource "aws_nat_gateway" "nat_gw" {
  allocation_id = aws_eip.nat_eip.id
  subnet_id     = aws_subnet.public_subnet1.id # Place NAT Gateway in a public subnet
  tags = {
    Name = "main-nat-gateway"
  }

  # Ensure the internet gateway is created before the NAT Gateway
  depends_on = [aws_internet_gateway.igw]
}
resource "aws_route_table_association" "public_subnet1" {
  subnet_id      = aws_subnet.public_subnet1.id
  route_table_id = aws_route_table.public_rtb.id
}
resource "aws_route_table_association" "public_subnet2" {
  subnet_id      = aws_subnet.public_subnet2.id
  route_table_id = aws_route_table.public_rtb.id
}
resource "aws_route_table_association" "public_subnet3" {
  subnet_id      = aws_subnet.public_subnet3.id
  route_table_id = aws_route_table.public_rtb.id
}
```