

Write Terraform script to create highly available infrastructure in AWS. The infra should have 1 vpc, 3 subnets setup in 3 different az and 2 instances setup in 2 different subnets

task12.tf

Provider Configuration

```
provider "aws" {  
  region    = "us-west-2"  
  access_key = "*****"  
  secret_key = "*****"  
}
```

VPC Creation

```
resource "aws_vpc" "my_vpc" {  
  cidr_block      = "10.0.0.0/16"  
  enable_dns_support = true  
  enable_dns_hostnames = true  
  
  tags = {  
    Name = "my_vpc"  
  }  
}
```

Subnet Creation across three different availability zones

```
resource "aws_subnet" "subnet1" {  
  vpc_id      = aws_vpc.my_vpc.id  
  cidr_block   = "10.0.1.0/24"  
  availability_zone = "us-west-2a"  
  
  tags = {  
    Name = "subnet1"  
  }  
}
```

```
resource "aws_subnet" "subnet2" {  
  vpc_id      = aws_vpc.my_vpc.id  
  cidr_block   = "10.0.2.0/24"  
  availability_zone = "us-west-2b"  
  
  tags = {  
    Name = "subnet2"  
  }  
}
```

```

resource "aws_subnet" "subnet3" {
  vpc_id      = aws_vpc.my_vpc.id
  cidr_block  = "10.0.3.0/24"
  availability_zone = "us-west-2c"

  tags = {
    Name = "subnet3"
  }
}

# Security Group for instance access control
resource "aws_security_group" "instance_sg" {
  vpc_id = aws_vpc.my_vpc.id

  ingress {
    from_port = 0
    to_port   = 0
    protocol  = "-1"
    cidr_blocks = ["10.0.0.0/16"]
  }

  egress {
    from_port = 0
    to_port   = 0
    protocol  = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }

  tags = {
    Name = "instance_sg"
  }
}

# EC2 Instances in different subnets
resource "aws_instance" "instance1" {
  ami          = "ami-08116b9957a259459"
  instance_type = "t2.micro"
  subnet_id    = aws_subnet.subnet1.id
  vpc_security_group_ids = [aws_security_group.instance_sg.id]

  tags = {
    Name = "instance1"
  }
}

```

```
resource "aws_instance" "instance2" {
  ami          = "ami-08116b9957a259459"
  instance_type = "t2.micro"
  subnet_id    = aws_subnet.subnet2.id
  vpc_security_group_ids = [aws_security_group.instance_sg.id]

  tags = {
    Name = "instance2"
  }
}
```

#terraform init

```
ubuntu@ip-172-31-23-205:~/tfproj01$ terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.44.0

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

#terraform plan

```

ubuntu@ip-172-31-23-205:~/tfproj01$ terraform plan
Terraform used the selected providers to generate the following execution plan. Resources to be created:
+ create

Terraform will perform the following actions:

# aws_instance.instance1 will be created
+ resource "aws_instance" "instance1" {
  + ami                               = "ami-08116b9957a259459"
  + associate_public_ip_address      = (known after apply)
  + availability_zone                 = (known after apply)
  + cpu_core_count                    = (known after apply)
  + cpu_threads_per_core              = (known after apply)
  + disable_api_stop                  = (known after apply)
  + disable_api_termination           = (known after apply)
  + ebs_optimized                     = false
  + get_password_data                 = (known after apply)
  + host_id                           = (known after apply)
  + host_resource_group_arn           = (known after apply)
  + iam_instance_profile              = (known after apply)
  + id                                = (known after apply)
  + instance_initiated_shutdown_behavior = (known after apply)
  + instance_lifecycle                 = (known after apply)
  + instance_state                     = (known after apply)
  + instance_type                     = "t2.micro"
  + ipv6_address_count                 = (known after apply)
  + ipv6_addresses                     = (known after apply)
  + key_name                           = (known after apply)
  + monitoring                         = (known after apply)
  + outpost_arn                       = (known after apply)
  + password_data                     = (known after apply)
  + placement_group                   = (known after apply)
  + placement_partition_number         = (known after apply)
  + primary_network_interface_id       = (known after apply)
  + private_dns                        = (known after apply)
  + private_ip                         = (known after apply)
  + public_dns                         = (known after apply)
  + public_ip                          = (known after apply)
  + secondary_private_ips              = (known after apply)
  + security_groups                    = (known after apply)
  + source_dest_check                  = true
  + spot_instance_request_id           = (known after apply)
  + subnet_id                          = (known after apply)
  + tags_all                           = (known after apply)
  + tenancy                            = (known after apply)
  + user_data                          = (known after apply)
  + user_data_base64                   = (known after apply)
  + user_data_replace_on_change        = false
  + vpc_security_group_ids             = (known after apply)
}

# aws_instance.instance2 will be created
+ resource "aws_instance" "instance2" {
  + ami                               = "ami-08116b9957a259459"
  + associate_public_ip_address      = (known after apply)
  + availability_zone                 = (known after apply)
  + cpu_core_count                    = (known after apply)
  + cpu_threads_per_core              = (known after apply)
  + disable_api_stop                  = (known after apply)
  + disable_api_termination           = (known after apply)
  + ebs_optimized                     = false
  + get_password_data                 = (known after apply)
  + host_id                           = (known after apply)
  + host_resource_group_arn           = (known after apply)
  + iam_instance_profile              = (known after apply)
  + id                                = (known after apply)
  + instance_initiated_shutdown_behavior = (known after apply)
  + instance_lifecycle                 = (known after apply)
  + instance_state                     = (known after apply)
  + instance_type                     = "t2.micro"
  + ipv6_address_count                 = (known after apply)
  + ipv6_addresses                     = (known after apply)
  + key_name                           = (known after apply)
  + monitoring                         = (known after apply)
  + outpost_arn                       = (known after apply)
  + password_data                     = (known after apply)
  + placement_group                   = (known after apply)
  + placement_partition_number         = (known after apply)
  + primary_network_interface_id       = (known after apply)
  + private_dns                        = (known after apply)
  + private_ip                         = (known after apply)
  + public_dns                         = (known after apply)
  + public_ip                          = (known after apply)
  + secondary_private_ips              = (known after apply)
  + security_groups                    = (known after apply)
  + source_dest_check                  = true
  + spot_instance_request_id           = (known after apply)
  + subnet_id                          = (known after apply)
  + tags_all                           = (known after apply)
  + tenancy                            = (known after apply)
  + user_data                          = (known after apply)
  + user_data_base64                   = (known after apply)
  + user_data_replace_on_change        = false
  + vpc_security_group_ids             = (known after apply)
}

# aws_security_group.instance_sg will be created
+ resource "aws_security_group" "instance_sg" {
  + arn                               = (known after apply)
  + description                       = "Managed by Terraform"
  + egress                             = [
    + {
      + cidr_blocks = [
        + "0.0.0.0/0",
      ]
      + description = ""
      + from_port   = 0
      + ipv6_cidr_blocks = []
      + prefix_list_ids = []
      + protocol     = "tcp"
      + security_groups = []
      + self         = false
      + to_port      = 0
    },
  ]
  + id                                = (known after apply)
  + ingress                             = [
    + {
      + cidr_blocks = [
        + "10.0.0.0/16",
      ]
      + description = ""
      + from_port   = 0
      + ipv6_cidr_blocks = []
      + prefix_list_ids = []
      + protocol     = "tcp"
      + security_groups = []
      + self         = false
      + to_port      = 0
    },
  ]
  + name                               = (known after apply)
  + name_prefix                       = (known after apply)
  + owner_id                           = (known after apply)
  + revoke_rules_on_delete             = false
  + tags_all                           = (known after apply)
  + vpc_id                             = (known after apply)
}

# aws_subnet.subnet1 will be created
+ resource "aws_subnet" "subnet1" {
  + arn                                = (known after apply)
  + assign_ipv6_address_on_creation    = false
  + availability_zone                   = "us-west-2a"
  + availability_zone_id                = (known after apply)
  + cidr_block                          = "10.0.1.0/24"
  + enable_dns64                       = false
  + enable_resource_name_dns_a_record_on_launch = false
  + enable_resource_name_dns_aaaa_record_on_launch = false
  + id                                  = (known after apply)
  + ipv6_cidr_block_association_id      = (known after apply)
  + ipv6_native                         = false
  + map_public_ip_on_launch             = false
  + owner_id                            = (known after apply)
  + private_dns_hostname_type_on_launch = (known after apply)
  + tags_all                            = (known after apply)
  + vpc_id                              = (known after apply)
}

# aws_subnet.subnet2 will be created
+ resource "aws_subnet" "subnet2" {
  + arn                                = (known after apply)
  + assign_ipv6_address_on_creation    = false
  + availability_zone                   = "us-west-2b"
  + availability_zone_id                = (known after apply)
  + cidr_block                          = "10.0.2.0/24"

```

```

# aws_subnet.subnet2 will be created
+ resource "aws_subnet" "subnet2" {
  + arn                                = (known after apply)
  + assign_ipv6_address_on_creation    = false
  + availability_zone                  = "us-west-2b"
  + availability_zone_id                = (known after apply)
  + cidr_block                         = "10.0.2.0/24"
  + enable_dns64                       = false
  + enable_resource_name_dns_a_record_on_launch = false
  + enable_resource_name_dns_aaaa_record_on_launch = false
  + id                                 = (known after apply)
  + ipv6_cidr_block_association_id      = (known after apply)
  + ipv6_native                         = false
  + map_public_ip_on_launch             = false
  + owner_id                           = (known after apply)
  + private_dns_hostname_type_on_launch = (known after apply)
  + tags_all                           = (known after apply)
  + vpc_id                             = (known after apply)
}

# aws_subnet.subnet3 will be created
+ resource "aws_subnet" "subnet3" {
  + arn                                = (known after apply)
  + assign_ipv6_address_on_creation    = false
  + availability_zone                  = "us-west-2c"
  + availability_zone_id                = (known after apply)
  + cidr_block                         = "10.0.3.0/24"
  + enable_dns64                       = false
  + enable_resource_name_dns_a_record_on_launch = false
  + enable_resource_name_dns_aaaa_record_on_launch = false
  + id                                 = (known after apply)
  + ipv6_cidr_block_association_id      = (known after apply)
  + ipv6_native                         = false
  + map_public_ip_on_launch             = false
  + owner_id                           = (known after apply)
  + private_dns_hostname_type_on_launch = (known after apply)
  + tags_all                           = (known after apply)
  + vpc_id                             = (known after apply)
}

# aws_vpc.my_vpc will be created
+ resource "aws_vpc" "my_vpc" {
  + arn                                = (known after apply)
  + cidr_block                         = "10.0.0.0/16"
  + default_network_acl_id             = (known after apply)
  + default_route_table_id             = (known after apply)
  + default_security_group_id          = (known after apply)
  + dhcp_options_id                    = (known after apply)
  + enable_dns_hostnames                = true
  + enable_dns_support                  = true
  + enable_network_address_usage_metrics = (known after apply)
  + id                                 = (known after apply)
  + instance_tenancy                    = "default"
  + ipv6_association_id                 = (known after apply)
  + ipv6_cidr_block                     = (known after apply)
  + ipv6_cidr_block_network_border_group = (known after apply)
  + main_route_table_id                 = (known after apply)
  + owner_id                           = (known after apply)
  + tags_all                           = (known after apply)
}

```

Plan: 7 to add, 0 to change, 0 to destroy.

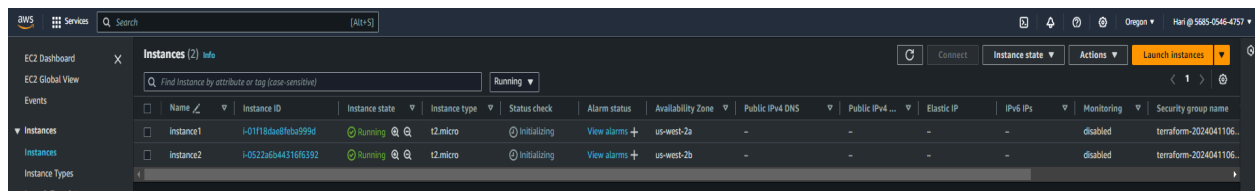
#terraform apply

```

aws_vpc.my_vpc: Creating...
aws_vpc.my_vpc: Still creating... [10s elapsed]
aws_vpc.my_vpc: Creation complete after 11s [id=vpc-050d6194a8ad4e48e]
aws_subnet.subnet1: Creating...
aws_subnet.subnet2: Creating...
aws_subnet.subnet3: Creating...
aws_security_group.instance_sg: Creating...
aws_subnet.subnet3: Creation complete after 1s [id=subnet-061c9b69920da6af0]
aws_subnet.subnet1: Creation complete after 1s [id=subnet-09fd05d775c2a5b56]
aws_subnet.subnet2: Creation complete after 1s [id=subnet-09eab0982b32624af]
aws_security_group.instance_sg: Creation complete after 2s [id=sg-0ecd2c5a0f1d8e088]
aws_instance.instance2: Creating...
aws_instance.instance1: Creating...
aws_instance.instance2: Still creating... [10s elapsed]
aws_instance.instance1: Still creating... [10s elapsed]
aws_instance.instance2: Still creating... [20s elapsed]
aws_instance.instance1: Still creating... [20s elapsed]
aws_instance.instance2: Still creating... [30s elapsed]
aws_instance.instance1: Still creating... [30s elapsed]
aws_instance.instance1: Creation complete after 32s [id=i-01f18dae8feba999d]
aws_instance.instance2: Still creating... [40s elapsed]
aws_instance.instance2: Creation complete after 42s [id=i-0522a6b44316f6392]

```

Result



The screenshot shows the AWS Management Console interface for the EC2 service. The 'Instances' page is active, displaying a list of two EC2 instances. The left sidebar shows the navigation menu with 'Instances' selected. The top navigation bar includes the AWS logo, 'Services' link, a search bar, and the user's profile information. The main content area shows a table of instances with columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, Public IPv4 DNS, Public IPv4, Elastic IP, IPv6 IPs, Monitoring, and Security group name. Both instances are in the 'Running' state.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4	Elastic IP	IPv6 IPs	Monitoring	Security group name
Instance1	i-01f18dae8feba999d	Running	t2.micro	Initializing	View alarms +	us-west-2a	-	-	-	-	disabled	terraform-2024041106...
Instance2	i-0522a6b44316f6392	Running	t2.micro	Initializing	View alarms +	us-west-2b	-	-	-	-	disabled	terraform-2024041106...