Table of Contents

Tools used 2

Architecture Diagrams 2

Server provisioning with Terraform and Ansible 3

3-Tier Application architecture on AWS 4

Project Resources 6

Thought Process 8

Setting up the environment 8

Prerequisites 8

Step 1: AWS account setup and login 8

Step 2: AWS User creation, policy assignment and credentials setup 8

Step 3 Install Terraform (Manual Process) 10

Defining SSH key-pair files 11

local-exec and remote-exec 11

remote-exec 11

local-exec 11

For Ansible, you can first run the Terraform, and output the IP addresses, then run ansible-playbook on those hosts 11

Description of various config files 11

Terraform 12

Define Terraform version: terraform.tf 12

Define AWS Provider: provider.tf 12

Define AWS Resources: resources.tf 12

Define Terraform Variables: variables.tf 12

Define Terraform Outputs: outputs.tf 12

Ansible 12

install_jenkins.yaml 12

install_java.yaml 12

install GitLab 12

Deploy Application 12

terraform init 13

terraform plan 13

terraform apply 34

terraform destroy.....65
My VPC Resources 74
VPC - main 74
Subnets 75
Route Table.....75
IG.....75
NAT.....75
NACL.....76

Tools used

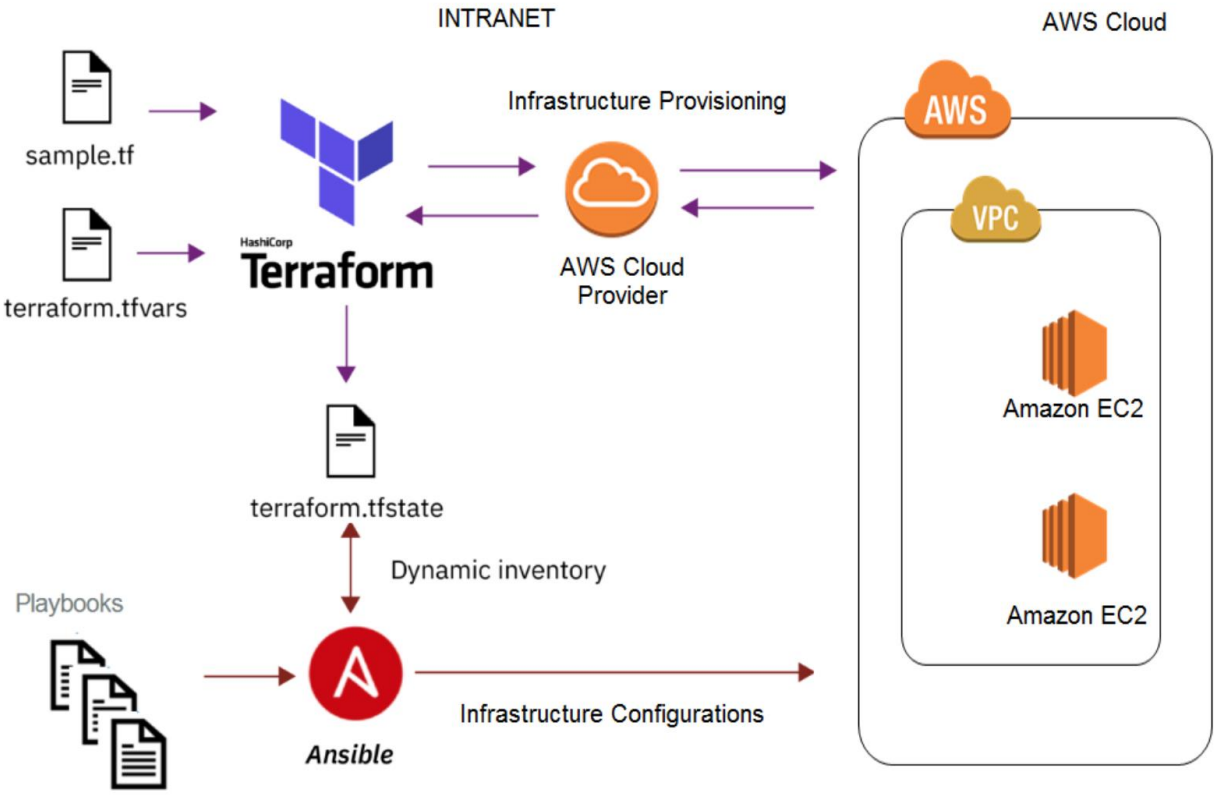
Ansible – To install and setup Jenkins master

Terraform – To form my VPC Resources

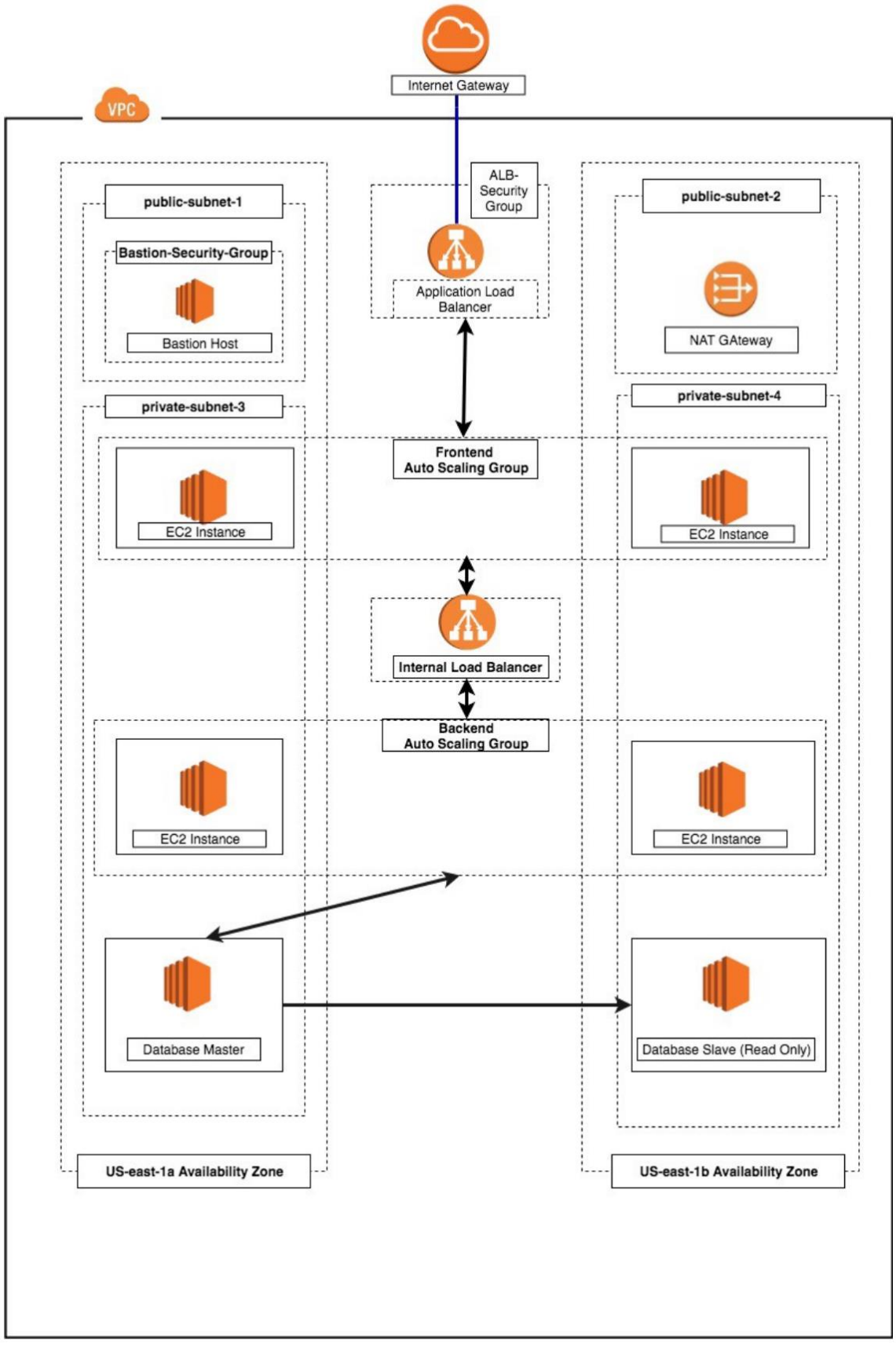
Jenkins – To implement CI/CD by accessing the VPC instances and deploying the node application

Architecture Diagrams

Server provisioning with Terraform and Ansible



3-Tier Application architecture on AWS


Internet Gateway
VPC
public-subnet-1
Bastion-Security-Group
Bastion Host
ALB-
Security
Group
Application Load
Balancer
public-subnet-2
NAT GAteway
private-subnet-3
Frontend
Auto Scaling Group
private-subnet-4
EC2 Instance
EC2 Instance
Internal Load Balancer
Backend
Auto Scaling Group
EC2 Instance
EC2 Instance
Database Master
Database Slave (Read Only)
US-east-1a Availability Zone
US-east-1b Availability Zone

Project Resources

Repository for this project: https://github.com/hani-hub/nodeApp-ansible-terraform-repo

Directory structure

```
.
├── artifacts
│   ├── config
│   │   └── config_multi-nodes.yaml
│   ├── playbooks
│   │   ├── install_gitlab.yaml
│   │   ├── install_java.yaml
│   │   └── install_jenkins.yaml
│   ├── scripts
│   │   ├── config_software.sh
│   │   ├── install_software.sh
│   │   └── ssh_pass.sh
│   ├── templates
│   │   ├── install_busybox.sh
│   │   ├── install_jenkins.sh
│   │   ├── install_nginx.sh
│   │   └── user_data.sh
│   └── terraform
│       ├── outputs.tf
│       ├── provider.tf
│       ├── resources.tf
│       ├── terraform.tf
│       └── variables.tf
├── images
│   ├── aws_configure.png
```

```
|   ├── aws_terraform_ans\v1.png
|   ├── aws_terraform_ans_v1.png
|   └── jenkins-ci.png
├── install.sh
├── README.md
├── screening
|   ├── api
|   |   ├── app.js
|   |   ├── bin
|   |   |   └── www
|   |   ├── package.json
|   |   ├── package-lock.json
|   |   └── README.md
|   ├── README.md
|   └── web
|       ├── app.js
|       ├── bin
|       |   └── www
|       ├── package.json
|       ├── package-lock.json
|       ├── public
|       |   └── stylesheets
|       |       └── style.css
|       ├── README.md
|       ├── routes
|       |   └── index.js
|       └── views
|           ├── error.jade
|           ├── index.jade
```

```
|       └── layout.jade
└── Vagrantfile
```

Thought Process

Use combination of IAC and CM

Terraform will provision infrastructure like EC2 instances, Security Goups, ELB and VPC into AWS IaC

Ansible will deploy/test application on EC2 instance as CM like Jenkins and GitLab

Setting up the environment

This guide assumes that you already have some understanding of AWS and have a working account.

The installation of Terraform and Ansible are straightforward, and the details are at this link.

Prerequisites

AWS CLI (Install AWS CLI)

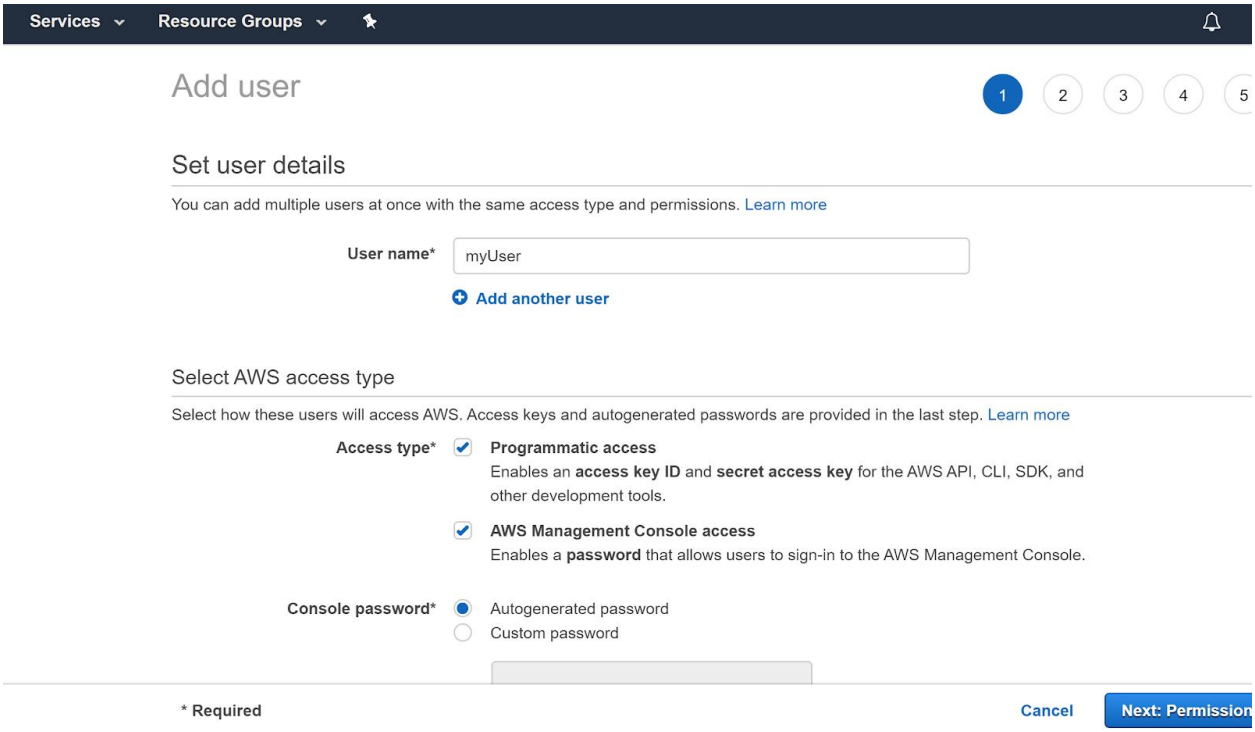
Terraform (Install Terraform)

Step 1: AWS account setup and login

Setup AWS account if not already done
Login to your aws account

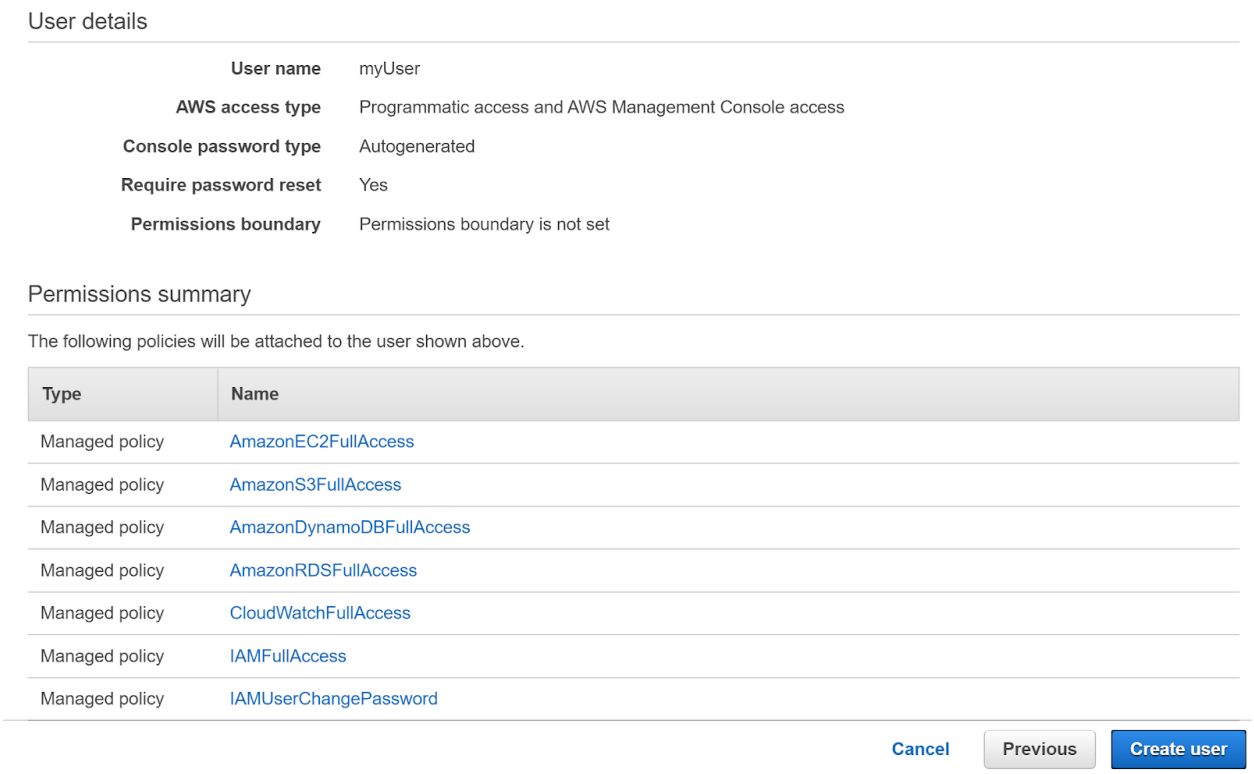
Step 2: AWS User creation, policy assignment and credentials setup

Go to services -> IAM -> Users -> Add user
Add user details

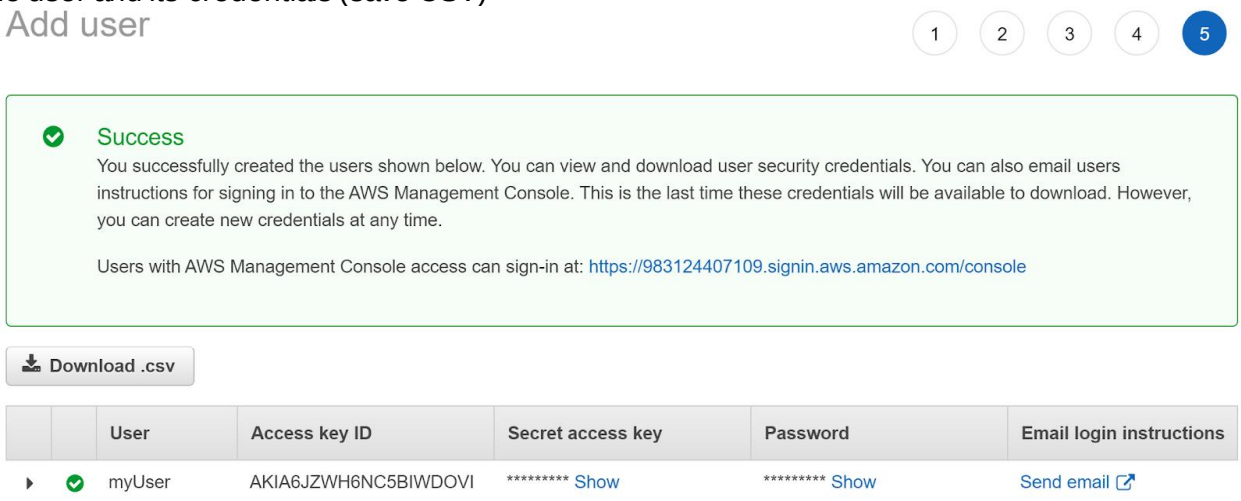


Attach policies to this user

```
AmazonEC2FullAccess
AmazonS3FullAccess
AmazonDynamoDBFullAccess
AmazonRDSFullAccess
IAMFullAccess
CloudWatchFullAccess
```



Save the user and its credentials (save CSV)



Step 3 Install Terraform (Manual Process)

Download the package in a location of your choice, from
https://releases.hashicorp.com/terraform/0.12.26/terraform_0.12.26_linux_amd64.zip
Unzip this package

unzip terraform_0.12.26_linux_amd64.zip

3. Add the binary terraform path to PATH variable

echo $PATH

vi ~/.bashrc
Add line *export PATH = $PATH:<PATH_TO_YOURTERRAFORM_BINARY>*
source ~/.bashrc

4 Verify installation
Terraform -help

Ansible (Install Ansible)

Defining SSH key-pair files

local-exec and remote-exec

These two built in provisioners local-exec and remote-exec are required for Ansible to work in Terraform, as Terraform lacks the necessary native plug-ins. This is the workaround to invoke Ansible within the local-exec provisioner. That requires to **configure** the connection with the host, user, and private_key, see resource.tf for more details.

remote-exec

Python is required for Ansible to work, by using the "**remote-exec**" it makes sure that Python is installed before it's possible to invoke "**local-exec**"

local-exec

For Ansible, you can first run the Terraform, and output the IP addresses, then run ansible-playbook on those hosts

Description of various config files

Terraform

Define Terraform version: **terraform.tf**

Define AWS Provider: **provider.tf**

Define AWS Resources: resources.tf

Define Terraform Variables: **variables.tf**

Define Terraform Outputs: **outputs.tf**

Ansible

install_jenkins.yaml

install_java.yaml

install GitLab

Deploy Application

terraform init

```
[ec2-user@ip-172-31-34-15 AWS-Terraform]$ terraform init

Initializing the backend...

Initializing provider plugins...

The following providers do not have any version constraints in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking
changes, it is recommended to add version = "..." constraints to the
corresponding provider blocks in configuration, with the constraint strings
suggested below.

* provider.aws: version = "~> 2.64"


Warning: Interpolation-only expressions are deprecated

  on aws.tf line 2, in provider "aws":
   2:     access_key = "${var.aws_access_key}"

Terraform 0.11 and earlier required all non-constant expressions to be
provided via interpolation syntax, but this pattern is now deprecated. To
silence this warning, remove the "${ sequence from the start and the }"
sequence from the end of this expression, leaving just the inner expression.

Template interpolation syntax is still used to construct strings from
expressions when the template includes multiple interpolation sequences or a
mixture of literal strings and interpolations. This deprecation applies only
to templates that consist entirely of a single interpolation sequence.

(and 42 more similar warnings elsewhere)

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

```
[ec2-user@ip-172-31-34-15 AWS-Terraform]$ terraform plan

Refreshing Terraform state in-memory prior to plan...
```

```
The refreshed state will be used to calculate this plan, but will not be

persisted to local or remote state storage.




------------------------------------------------------------------------



An execution plan has been generated and is shown below.

Resource actions are indicated with the following symbols:

  + create



Terraform will perform the following actions:



  # aws_eip.nat will be created

  + resource "aws_eip" "nat" {

      + allocation_id     = (known after apply)

      + association_id    = (known after apply)

      + customer_owned_ip = (known after apply)
```

```
    + domain           = (known after apply)
    + id               = (known after apply)
    + instance         = (known after apply)
    + network_interface = (known after apply)
    + private_dns      = (known after apply)
    + private_ip       = (known after apply)
    + public_dns       = (known after apply)
    + public_ip        = (known after apply)
    + public_ipv4_pool  = (known after apply)
    + vpc              = (known after apply)
  }


# aws_internet_gateway.main-igw will be created
+ resource "aws_internet_gateway" "main-igw" {
    + id       = (known after apply)
    + owner_id = (known after apply)
    + tags     = {
```

```
      + "Name" = "main-igw"
    }
  + vpc_id   = (known after apply)
}

# aws_nat_gateway.main-natgw will be created
+ resource "aws_nat_gateway" "main-natgw" {
  + allocation_id       = (known after apply)
  + id                  = (known after apply)
  + network_interface_id = (known after apply)
  + private_ip          = (known after apply)
  + public_ip           = (known after apply)
  + subnet_id           = (known after apply)
  + tags                = {
      + "Name" = "main-nat"
    }
}
```

```
# aws_route_table.main-private-rt will be created
+ resource "aws_route_table" "main-private-rt" {
    + id               = (known after apply)
    + owner_id         = (known after apply)
    + propagating_vgws = (known after apply)
    + route            = [
        + {
            + cidr_block               = "0.0.0.0/0"
            + egress_only_gateway_id   = ""
            + gateway_id               = (known after apply)
            + instance_id              = ""
            + ipv6_cidr_block          = ""
            + nat_gateway_id           = ""
            + network_interface_id     = ""
            + transit_gateway_id       = ""
            + vpc_peering_connection_id = ""
```

```
        },
      ]
    + tags           = {
        + "Name" = "main-private-rt"
      }
    + vpc_id         = (known after apply)
  }


# aws_route_table.main-public-rt will be created
+ resource "aws_route_table" "main-public-rt" {
    + id              = (known after apply)
    + owner_id        = (known after apply)
    + propagating_vgws = (known after apply)
    + route           = [
        + {
            + cidr_block              = "0.0.0.0/0"
            + egress_only_gateway_id  = ""
```

```
      + gateway_id             = (known after apply)
      + instance_id            = ""
      + ipv6_cidr_block        = ""
      + nat_gateway_id         = ""
      + network_interface_id   = ""
      + transit_gateway_id     = ""
      + vpc_peering_connection_id = ""
    },
  ]
  + tags           = {
    + "Name" = "main-public-rt"
  }
  + vpc_id         = (known after apply)
}

# aws_route_table_association.private-assoc-1 will be created
+ resource "aws_route_table_association" "private-assoc-1" {
```

```
    + id             = (known after apply)
    + route_table_id = (known after apply)
    + subnet_id      = (known after apply)
  }

# aws_route_table_association.private-assoc-2 will be created
+ resource "aws_route_table_association" "private-assoc-2" {
    + id             = (known after apply)
    + route_table_id = (known after apply)
    + subnet_id      = (known after apply)
  }

# aws_route_table_association.private-assoc-3 will be created
+ resource "aws_route_table_association" "private-assoc-3" {
    + id             = (known after apply)
    + route_table_id = (known after apply)
    + subnet_id      = (known after apply)
```

```
  }


# aws_route_table_association.private-assoc-4 will be created
+ resource "aws_route_table_association" "private-assoc-4" {
    + id             = (known after apply)
    + route_table_id = (known after apply)
    + subnet_id      = (known after apply)
  }


# aws_route_table_association.public-assoc-1 will be created
+ resource "aws_route_table_association" "public-assoc-1" {
    + id             = (known after apply)
    + route_table_id = (known after apply)
    + subnet_id      = (known after apply)
  }


# aws_route_table_association.public-assoc-2 will be created
```

```
+ resource "aws_route_table_association" "public-assoc-2" {
    + id             = (known after apply)
    + route_table_id = (known after apply)
    + subnet_id      = (known after apply)
  }

# aws_route_table_association.public-assoc-3 will be created
+ resource "aws_route_table_association" "public-assoc-3" {
    + id             = (known after apply)
    + route_table_id = (known after apply)
    + subnet_id      = (known after apply)
  }

# aws_route_table_association.public-assoc-4 will be created
+ resource "aws_route_table_association" "public-assoc-4" {
    + id             = (known after apply)
    + route_table_id = (known after apply)
```

```
    + subnet_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            = "ap-southeast-1a"

    + availability_zone_id         = (known after apply)

    + cidr_block                   = "10.0.1.0/24"

    + id                           = (known after apply)

    + ipv6_cidr_block              = (known after apply)

    + ipv6_cidr_block_association_id = (known after apply)

    + map_public_ip_on_launch      = false

    + owner_id                     = (known after apply)

    + tags                         = {

        + "Name" = "app-subnet-1"
```

```
    }

  + 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           = "ap-southeast-1b"

  + availability_zone_id        = (known after apply)

  + cidr_block                  = "10.0.2.0/24"

  + id                          = (known after apply)

  + ipv6_cidr_block             = (known after apply)

  + ipv6_cidr_block_association_id = (known after apply)

  + map_public_ip_on_launch     = false

  + owner_id                    = (known after apply)

  + tags                        = {
```

```
      + "Name" = "app-subnet-2"
    }
  + 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               = "ap-southeast-1a"
  + availability_zone_id            = (known after apply)
  + cidr_block                      = "10.0.3.0/24"
  + id                              = (known after apply)
  + ipv6_cidr_block                 = (known after apply)
  + ipv6_cidr_block_association_id  = (known after apply)
  + map_public_ip_on_launch         = false
  + owner_id                        = (known after apply)
```

```
    + tags                          = {
        + "Name" = "elb-subnet-1"
      }
    + vpc_id                        = (known after apply)
  }


# aws_subnet.subnet4 will be created
+ resource "aws_subnet" "subnet4" {
    + arn                           = (known after apply)
    + assign_ipv6_address_on_creation = false
    + availability_zone             = "ap-southeast-1b"
    + availability_zone_id          = (known after apply)
    + cidr_block                    = "10.0.4.0/24"
    + id                            = (known after apply)
    + ipv6_cidr_block               = (known after apply)
    + ipv6_cidr_block_association_id = (known after apply)
    + map_public_ip_on_launch       = false
```

```
    + owner_id                      = (known after apply)
    + tags                          = {
        + "Name" = "elb-subnet-2"
      }
    + vpc_id                        = (known after apply)
  }


# aws_subnet.subnet5 will be created
+ resource "aws_subnet" "subnet5" {
    + arn                           = (known after apply)
    + assign_ipv6_address_on_creation = false
    + availability_zone             = "ap-southeast-1a"
    + availability_zone_id          = (known after apply)
    + cidr_block                    = "10.0.5.0/24"
    + id                            = (known after apply)
    + ipv6_cidr_block               = (known after apply)
    + ipv6_cidr_block_association_id = (known after apply)
```

```
    + map_public_ip_on_launch        = false
    + owner_id                       = (known after apply)
    + tags                           = {
        + "Name" = "db-subnet-1"
      }
    + vpc_id                         = (known after apply)
  }


# aws_subnet.subnet6 will be created
+ resource "aws_subnet" "subnet6" {
    + arn                            = (known after apply)
    + assign_ipv6_address_on_creation = false
    + availability_zone              = "ap-southeast-1b"
    + availability_zone_id           = (known after apply)
    + cidr_block                     = "10.0.6.0/24"
    + id                             = (known after apply)
    + ipv6_cidr_block                = (known after apply)
```

```
    + ipv6_cidr_block_association_id = (known after apply)
    + map_public_ip_on_launch        = false
    + owner_id                       = (known after apply)
    + tags                           = {
        + "Name" = "db-subnet-2"
      }
    + vpc_id                         = (known after apply)
  }


# aws_subnet.subnet7 will be created
+ resource "aws_subnet" "subnet7" {
    + arn                            = (known after apply)
    + assign_ipv6_address_on_creation = false
    + availability_zone              = "ap-southeast-1a"
    + availability_zone_id           = (known after apply)
    + cidr_block                     = "10.0.7.0/24"
    + id                             = (known after apply)
```

```
    + ipv6_cidr_block               = (known after apply)
    + ipv6_cidr_block_association_id = (known after apply)
    + map_public_ip_on_launch       = false
    + owner_id                      = (known after apply)
    + tags                          = {
        + "Name" = "nat-subnet-1"
      }
    + vpc_id                        = (known after apply)
  }


# aws_subnet.subnet8 will be created
+ resource "aws_subnet" "subnet8" {
    + arn                           = (known after apply)
    + assign_ipv6_address_on_creation = false
    + availability_zone             = "ap-southeast-1b"
    + availability_zone_id          = (known after apply)
    + cidr_block                    = "10.0.8.0/24"
```

```
    + id                      = (known after apply)
    + ipv6_cidr_block         = (known after apply)
    + ipv6_cidr_block_association_id = (known after apply)
    + map_public_ip_on_launch = false
    + owner_id                = (known after apply)
    + tags                    = {
        + "Name" = "nat-subnet-2"
      }
    + vpc_id                  = (known after apply)
  }


# aws_vpc.main will be created
+ resource "aws_vpc" "main" {
    + arn                            = (known after apply)
    + assign_generated_ipv6_cidr_block = false
    + 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_classiclink             = (known after apply)
  + enable_classiclink_dns_support  = (known after apply)
  + enable_dns_hostnames           = true
  + enable_dns_support             = true
  + id                             = (known after apply)
  + instance_tenancy               = "default"
  + ipv6_association_id            = (known after apply)
  + ipv6_cidr_block                = (known after apply)
  + main_route_table_id            = (known after apply)
  + owner_id                       = (known after apply)
  + tags                           = {
      + "Name" = "main"
    }
}
```

```
Plan: 22 to add, 0 to change, 0 to destroy.

Warning: Interpolation-only expressions are deprecated

  on aws.tf line 2, in provider "aws":
   2:     access_key = "${var.aws_access_key}"

Terraform 0.11 and earlier required all non-constant expressions to be
provided via interpolation syntax, but this pattern is now deprecated. To
silence this warning, remove the "${ sequence from the start and the }"
sequence from the end of this expression, leaving just the inner expression.

Template interpolation syntax is still used to construct strings from
expressions when the template includes multiple interpolation sequences or a
mixture of literal strings and interpolations. This deprecation applies only
to templates that consist entirely of a single interpolation sequence.
```

```
(and 42 more similar warnings elsewhere)

------------------------------------------------------------------------

Note: You didn't specify an "-out" parameter to save this plan, so Terraform
can't guarantee that exactly these actions will be performed if
"terraform apply" is subsequently run.
```

terraform apply

```
An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create
```

```
Terraform will perform the following actions:



  # aws_eip.nat will be created

  + resource "aws_eip" "nat" {

      + allocation_id     = (known after apply)

      + association_id    = (known after apply)

      + customer_owned_ip = (known after apply)

      + domain            = (known after apply)

      + id                = (known after apply)

      + instance          = (known after apply)

      + network_interface = (known after apply)

      + private_dns       = (known after apply)

      + private_ip        = (known after apply)
```

```
    + public_dns        = (known after apply)
    + public_ip         = (known after apply)
    + public_ipv4_pool  = (known after apply)
    + vpc               = (known after apply)
  }


# aws_internet_gateway.main-igw will be created
+ resource "aws_internet_gateway" "main-igw" {
    + id       = (known after apply)
    + owner_id = (known after apply)
    + tags     = {
        + "Name" = "main-igw"
      }
```

```
    + vpc_id   = (known after apply)

  }



# aws_nat_gateway.main-natgw will be created

+ resource "aws_nat_gateway" "main-natgw" {

    + allocation_id       = (known after apply)

    + id                  = (known after apply)

    + network_interface_id = (known after apply)

    + private_ip          = (known after apply)

    + public_ip           = (known after apply)

    + subnet_id           = (known after apply)

    + tags                = {

        + "Name" = "main-nat"
```

```
    }

  }



# aws_route_table.main-private-rt will be created

+ resource "aws_route_table" "main-private-rt" {

    + id              = (known after apply)

    + owner_id        = (known after apply)

    + propagating_vgws = (known after apply)

    + route           = [

        + {

            + cidr_block              = "0.0.0.0/0"

            + egress_only_gateway_id  = ""

            + gateway_id              = (known after apply)
```

```
      + instance_id             = ""
      + ipv6_cidr_block         = ""
      + nat_gateway_id          = ""
      + network_interface_id    = ""
      + transit_gateway_id      = ""
      + vpc_peering_connection_id = ""
    },
  ]
  + tags          = {
    + "Name" = "main-private-rt"
  }
  + vpc_id        = (known after apply)
}
```

```
# aws_route_table.main-public-rt will be created

+ resource "aws_route_table" "main-public-rt" {

    + id              = (known after apply)

    + owner_id        = (known after apply)

    + propagating_vgws = (known after apply)

    + route           = [

        + {

            + cidr_block              = "0.0.0.0/0"

            + egress_only_gateway_id  = ""

            + gateway_id              = (known after apply)

            + instance_id             = ""

            + ipv6_cidr_block         = ""
```

```
      + nat_gateway_id         = ""
      + network_interface_id   = ""
      + transit_gateway_id     = ""
      + vpc_peering_connection_id = ""
    },
  ]
  + tags           = {
    + "Name" = "main-public-rt"
  }
  + vpc_id         = (known after apply)
}

# aws_route_table_association.private-assoc-1 will be created
```

```
+ resource "aws_route_table_association" "private-assoc-1" {

    + id             = (known after apply)

    + route_table_id = (known after apply)

    + subnet_id      = (known after apply)

  }




# aws_route_table_association.private-assoc-2 will be created

+ resource "aws_route_table_association" "private-assoc-2" {

    + id             = (known after apply)

    + route_table_id = (known after apply)

    + subnet_id      = (known after apply)

  }
```

```
# aws_route_table_association.private-assoc-3 will be created

+ resource "aws_route_table_association" "private-assoc-3" {

    + id             = (known after apply)

    + route_table_id = (known after apply)

    + subnet_id      = (known after apply)

  }




# aws_route_table_association.private-assoc-4 will be created

+ resource "aws_route_table_association" "private-assoc-4" {

    + id             = (known after apply)

    + route_table_id = (known after apply)

    + subnet_id      = (known after apply)

  }
```

```
# aws_route_table_association.public-assoc-1 will be created

+ resource "aws_route_table_association" "public-assoc-1" {

    + id             = (known after apply)

    + route_table_id = (known after apply)

    + subnet_id      = (known after apply)

  }



# aws_route_table_association.public-assoc-2 will be created

+ resource "aws_route_table_association" "public-assoc-2" {

    + id             = (known after apply)

    + route_table_id = (known after apply)

    + subnet_id      = (known after apply)
```

```
  }



# aws_route_table_association.public-assoc-3 will be created

+ resource "aws_route_table_association" "public-assoc-3" {

    + id             = (known after apply)

    + route_table_id = (known after apply)

    + subnet_id      = (known after apply)

  }



# aws_route_table_association.public-assoc-4 will be created

+ resource "aws_route_table_association" "public-assoc-4" {

    + id             = (known after apply)

    + route_table_id = (known after apply)
```

```
  + subnet_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               = "ap-southeast-1a"

  + availability_zone_id            = (known after apply)

  + cidr_block                      = "10.0.1.0/24"

  + id                              = (known after apply)

  + ipv6_cidr_block                 = (known after apply)

  + ipv6_cidr_block_association_id  = (known after apply)
```

```
    + map_public_ip_on_launch        = false
    + owner_id                       = (known after apply)
    + tags                           = {
        + "Name" = "app-subnet-1"
      }
    + 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              = "ap-southeast-1b"
```

```
  + availability_zone_id          = (known after apply)
  + cidr_block                    = "10.0.2.0/24"
  + id                            = (known after apply)
  + ipv6_cidr_block               = (known after apply)
  + ipv6_cidr_block_association_id = (known after apply)
  + map_public_ip_on_launch       = false
  + owner_id                      = (known after apply)
  + tags                          = {
      + "Name" = "app-subnet-2"
    }
  + 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               = "ap-southeast-1a"

    + availability_zone_id            = (known after apply)

    + cidr_block                      = "10.0.3.0/24"

    + id                              = (known after apply)

    + ipv6_cidr_block                 = (known after apply)

    + ipv6_cidr_block_association_id  = (known after apply)

    + map_public_ip_on_launch         = false

    + owner_id                        = (known after apply)

    + tags                            = {
```

```
      + "Name" = "elb-subnet-1"

    }

  + vpc_id                    = (known after apply)

 }



# aws_subnet.subnet4 will be created

+ resource "aws_subnet" "subnet4" {

  + arn                       = (known after apply)

  + assign_ipv6_address_on_creation = false

  + availability_zone         = "ap-southeast-1b"

  + availability_zone_id      = (known after apply)

  + cidr_block                = "10.0.4.0/24"

  + id                        = (known after apply)
```

```
    + ipv6_cidr_block               = (known after apply)
    + ipv6_cidr_block_association_id = (known after apply)
    + map_public_ip_on_launch       = false
    + owner_id                      = (known after apply)
    + tags                          = {
        + "Name" = "elb-subnet-2"
      }
    + vpc_id                        = (known after apply)
  }


# aws_subnet.subnet5 will be created
+ resource "aws_subnet" "subnet5" {
    + arn                           = (known after apply)
```

```
+ assign_ipv6_address_on_creation = false

+ availability_zone              = "ap-southeast-1a"

+ availability_zone_id           = (known after apply)

+ cidr_block                    = "10.0.5.0/24"

+ id                            = (known after apply)

+ ipv6_cidr_block               = (known after apply)

+ ipv6_cidr_block_association_id = (known after apply)

+ map_public_ip_on_launch        = false

+ owner_id                      = (known after apply)

+ tags                          = {

   + "Name" = "db-subnet-1"

 }

+ vpc_id                        = (known after apply)
```

```
  }



# aws_subnet.subnet6 will be created

+ resource "aws_subnet" "subnet6" {

    + arn                             = (known after apply)

    + assign_ipv6_address_on_creation = false

    + availability_zone               = "ap-southeast-1b"

    + availability_zone_id            = (known after apply)

    + cidr_block                      = "10.0.6.0/24"

    + id                              = (known after apply)

    + ipv6_cidr_block                 = (known after apply)

    + ipv6_cidr_block_association_id  = (known after apply)

    + map_public_ip_on_launch         = false
```

```
    + owner_id                      = (known after apply)
    + tags                          = {
        + "Name" = "db-subnet-2"
      }
    + vpc_id                        = (known after apply)
  }


# aws_subnet.subnet7 will be created
+ resource "aws_subnet" "subnet7" {
    + arn                           = (known after apply)
    + assign_ipv6_address_on_creation = false
    + availability_zone             = "ap-southeast-1a"
    + availability_zone_id          = (known after apply)
```

```
  + cidr_block                  = "10.0.7.0/24"
  + id                          = (known after apply)
  + ipv6_cidr_block             = (known after apply)
  + ipv6_cidr_block_association_id = (known after apply)
  + map_public_ip_on_launch     = false
  + owner_id                    = (known after apply)
  + tags                        = {
      + "Name" = "nat-subnet-1"
    }
  + vpc_id                      = (known after apply)
}

# aws_subnet.subnet8 will be created
```

```
+ resource "aws_subnet" "subnet8" {

    + arn                             = (known after apply)

    + assign_ipv6_address_on_creation = false

    + availability_zone               = "ap-southeast-1b"

    + availability_zone_id            = (known after apply)

    + cidr_block                      = "10.0.8.0/24"

    + id                              = (known after apply)

    + ipv6_cidr_block                 = (known after apply)

    + ipv6_cidr_block_association_id  = (known after apply)

    + map_public_ip_on_launch         = false

    + owner_id                        = (known after apply)

    + tags                            = {

        + "Name" = "nat-subnet-2"
```

```
    }

  + vpc_id                     = (known after apply)

 }



# aws_vpc.main will be created

+ resource "aws_vpc" "main" {

  + arn                          = (known after apply)

  + assign_generated_ipv6_cidr_block = false

  + 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_classiclink             = (known after apply)

+ enable_classiclink_dns_support  = (known after apply)

+ enable_dns_hostnames           = true

+ enable_dns_support             = true

+ id                             = (known after apply)

+ instance_tenancy               = "default"

+ ipv6_association_id            = (known after apply)

+ ipv6_cidr_block                = (known after apply)

+ main_route_table_id            = (known after apply)

+ owner_id                       = (known after apply)

+ tags                           = {

   + "Name" = "main"

 }
```

```
  }


Plan: 22 to add, 0 to change, 0 to destroy.


Warning: Interpolation-only expressions are deprecated

 on aws.tf line 2, in provider "aws":
  2:     access_key = "${var.aws_access_key}"

Terraform 0.11 and earlier required all non-constant expressions to be
provided via interpolation syntax, but this pattern is now deprecated. To
silence this warning, remove the "${ sequence from the start and the }"
```

sequence from the end of this expression, leaving just the inner expression.

Template interpolation syntax is still used to construct strings from

expressions when the template includes multiple interpolation sequences or a

mixture of literal strings and interpolations. This deprecation applies only

to templates that consist entirely of a single interpolation sequence.

(and 42 more similar warnings elsewhere)

Do you want to perform these actions?

Terraform will perform the actions described above.

Only 'yes' will be accepted to approve.

```
  Enter a value: yes



aws_vpc.main: Creating...

aws_eip.nat: Creating...

aws_eip.nat: Creation complete after 3s [id=eipalloc-08a2cad1a6bd39bd4]

aws_vpc.main: Still creating... [10s elapsed]

aws_vpc.main: Creation complete after 13s [id=vpc-0e388a3cde7b45bcb]

aws_subnet.subnet4: Creating...

aws_subnet.subnet3: Creating...

aws_subnet.subnet5: Creating...

aws_subnet.subnet8: Creating...

aws_subnet.subnet1: Creating...

aws_subnet.subnet2: Creating...
```

```
aws_subnet.subnet7: Creating...

aws_subnet.subnet6: Creating...

aws_internet_gateway.main-igw: Creating...

aws_subnet.subnet8: Creation complete after 5s [id=subnet-0b3713658a5024d07]

aws_nat_gateway.main-natgw: Creating...

aws_subnet.subnet7: Creation complete after 5s [id=subnet-0be0a74e7410aca4f]

aws_subnet.subnet1: Creation complete after 5s [id=subnet-047be506958ac9505]

aws_subnet.subnet3: Creation complete after 5s [id=subnet-00b11f9d1117c0027]

aws_subnet.subnet2: Creation complete after 5s [id=subnet-00d390a9c8dfdc977]

aws_subnet.subnet4: Creation complete after 5s [id=subnet-08e56a8a4365ac2ef]

aws_subnet.subnet5: Creation complete after 5s [id=subnet-0ab2e7c1b4f327350]

aws_subnet.subnet6: Creation complete after 5s [id=subnet-03438e4b834bcb577]

aws_internet_gateway.main-igw: Creation complete after 6s [id=igw-0cafcbb261928cffe]
```

aws_route_table.main-public-rt: Creating...

aws_route_table.main-public-rt: Creation complete after 5s [id=rtb-0aee36fab0e7b35eb]

aws_route_table_association.public-assoc-4: Creating...

aws_route_table_association.public-assoc-1: Creating...

aws_route_table_association.public-assoc-2: Creating...

aws_route_table_association.public-assoc-3: Creating...

aws_route_table_association.public-assoc-1: Creation complete after 1s [id=rtbassoc-046c1a6e64d873878]

aws_route_table_association.public-assoc-2: Creation complete after 1s [id=rtbassoc-0722125e07e237008]

aws_route_table_association.public-assoc-4: Creation complete after 1s [id=rtbassoc-076227cf24d5c3755]

aws_route_table_association.public-assoc-3: Creation complete after 1s [id=rtbassoc-074d0c83a1a3c90a3]

aws_nat_gateway.main-natgw: Still creating... [10s elapsed]

```
aws_nat_gateway.main-natgw: Still creating... [20s elapsed]

aws_nat_gateway.main-natgw: Still creating... [30s elapsed]

aws_nat_gateway.main-natgw: Still creating... [40s elapsed]

aws_nat_gateway.main-natgw: Still creating... [50s elapsed]

aws_nat_gateway.main-natgw: Still creating... [1m0s elapsed]

aws_nat_gateway.main-natgw: Still creating... [1m10s elapsed]

aws_nat_gateway.main-natgw: Still creating... [1m20s elapsed]

aws_nat_gateway.main-natgw: Still creating... [1m30s elapsed]

aws_nat_gateway.main-natgw: Creation complete after 1m38s [id=nat-0d67b00a7036d53c4]

aws_route_table.main-private-rt: Creating...

aws_route_table.main-private-rt: Creation complete after 5s [id=rtb-0542bf97862b5f10d]

aws_route_table_association.private-assoc-1: Creating...

aws_route_table_association.private-assoc-3: Creating...
```

```
aws_route_table_association.private-assoc-2: Creating...

aws_route_table_association.private-assoc-4: Creating...

aws_route_table_association.private-assoc-1: Creation complete after 1s [id=rtbassoc-
09c3a6aaefcb3e73c]

aws_route_table_association.private-assoc-2: Creation complete after 1s [id=rtbassoc-
000f93d37d0e129be]

aws_route_table_association.private-assoc-4: Creation complete after 1s [id=rtbassoc-
062c4783aaf0bfdbc]

aws_route_table_association.private-assoc-3: Creation complete after 1s [id=rtbassoc-
0df9bc5dc1735ec34]



Apply complete! Resources: 22 added, 0 changed, 0 destroyed.
```

terraform destroy

```
aws_eip.nat: Refreshing state... [id=eipalloc-08a2cad1a6bd39bd4]
aws_vpc.main: Refreshing state... [id=vpc-0e388a3cde7b45bcb]
aws_subnet.subnet3: Refreshing state... [id=subnet-00b11f9d1117c0027]
aws_subnet.subnet7: Refreshing state... [id=subnet-0be0a74e7410aca4f]
aws_subnet.subnet4: Refreshing state... [id=subnet-08e56a8a4365ac2ef]
aws_subnet.subnet6: Refreshing state... [id=subnet-03438e4b834bcb577]
```

```
aws_subnet.subnet8: Refreshing state... [id=subnet-0b3713658a5024d07]
aws_subnet.subnet1: Refreshing state... [id=subnet-047be506958ac9505]
aws_subnet.subnet5: Refreshing state... [id=subnet-0ab2e7c1b4f327350]
aws_subnet.subnet2: Refreshing state... [id=subnet-00d390a9c8dfdc977]
aws_internet_gateway.main-igw: Refreshing state... [id=igw-0cafcbb261928cffe]
aws_route_table.main-public-rt: Refreshing state... [id=rtb-0aee36fab0e7b35eb]
aws_nat_gateway.main-natgw: Refreshing state... [id=nat-0d67b00a7036d53c4]
aws_route_table_association.public-assoc-3: Refreshing state... [id=rtbassoc-074d0c83a1a3c90a3]
aws_route_table_association.public-assoc-4: Refreshing state... [id=rtbassoc-076227cf24d5c3755]
aws_route_table_association.public-assoc-1: Refreshing state... [id=rtbassoc-046c1a6e64d873878]
aws_route_table_association.public-assoc-2: Refreshing state... [id=rtbassoc-0722125e07e237008]
aws_route_table.main-private-rt: Refreshing state... [id=rtb-0542bf97862b5f10d]
aws_route_table_association.private-assoc-4: Refreshing state... [id=rtbassoc-062c4783aaf0bfdbc]
aws_route_table_association.private-assoc-1: Refreshing state... [id=rtbassoc-09c3a6aaefcb3e73c]
aws_route_table_association.private-assoc-3: Refreshing state... [id=rtbassoc-0df9bc5dc1735ec34]
aws_route_table_association.private-assoc-2: Refreshing state... [id=rtbassoc-000f93d37d0e129be]

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # aws_eip.nat will be destroyed
  - resource "aws_eip" "nat" {
      - association_id   = "eipassoc-37d35cfa" -> null
      - domain           = "vpc" -> null
      - id               = "eipalloc-08a2cad1a6bd39bd4" -> null
      - network_interface = "eni-0bab343104c00aca6" -> null
      - private_dns      = "ip-10-0-8-92.ap-southeast-1.compute.internal" -> null
      - private_ip       = "10.0.8.92" -> null
      - public_dns       = "ec2-18-140-146-63.ap-southeast-1.compute.amazonaws.com" -> null
      - public_ip        = "18.140.146.63" -> null
      - public_ipv4_pool  = "amazon" -> null
      - tags             = {} -> null
      - vpc              = true -> null
    }

  # aws_internet_gateway.main-igw will be destroyed
  - resource "aws_internet_gateway" "main-igw" {
      - id       = "igw-0cafcbb261928cffe" -> null
      - owner_id = "964305647536" -> null
      - tags     = {
          - "Name" = "main-igw"
        } -> null
      - vpc_id   = "vpc-0e388a3cde7b45bcb" -> null
    }
```

```
# aws_nat_gateway.main-natgw will be destroyed
- resource "aws_nat_gateway" "main-natgw" {
    - allocation_id       = "eipalloc-08a2cad1a6bd39bd4" -> null
    - id                  = "nat-0d67b00a7036d53c4" -> null
    - network_interface_id = "eni-0bab343104c00aca6" -> null
    - private_ip          = "10.0.8.92" -> null
    - public_ip           = "18.140.146.63" -> null
    - subnet_id           = "subnet-0b3713658a5024d07" -> null
    - tags                = {
        - "Name" = "main-nat"
      } -> null
  }

# aws_route_table.main-private-rt will be destroyed
- resource "aws_route_table" "main-private-rt" {
    - id             = "rtb-0542bf97862b5f10d" -> null
    - owner_id       = "964305647536" -> null
    - propagating_vgws = [] -> null
    - route          = [
        - {
            - cidr_block              = "0.0.0.0/0"
            - egress_only_gateway_id  = ""
            - gateway_id              = ""
            - instance_id             = ""
            - ipv6_cidr_block         = ""
            - nat_gateway_id          = "nat-0d67b00a7036d53c4"
            - network_interface_id    = ""
            - transit_gateway_id      = ""
            - vpc_peering_connection_id = ""
          },
      ] -> null
    - tags           = {
        - "Name" = "main-private-rt"
      } -> null
    - vpc_id         = "vpc-0e388a3cde7b45bcb" -> null
  }

# aws_route_table.main-public-rt will be destroyed
- resource "aws_route_table" "main-public-rt" {
    - id             = "rtb-0aee36fab0e7b35eb" -> null
    - owner_id       = "964305647536" -> null
    - propagating_vgws = [] -> null
    - route          = [
        - {
            - cidr_block              = "0.0.0.0/0"
            - egress_only_gateway_id  = ""
            - gateway_id              = "igw-0cafcbb261928cffe"
            - instance_id             = ""
```

```
          - ipv6_cidr_block          = ""
          - nat_gateway_id           = ""
          - network_interface_id     = ""
          - transit_gateway_id       = ""
          - vpc_peering_connection_id = ""
        },
      ] -> null
    - tags           = {
        - "Name" = "main-public-rt"
      } -> null
    - vpc_id         = "vpc-0e388a3cde7b45bcb" -> null
  }

# aws_route_table_association.private-assoc-1 will be destroyed
- resource "aws_route_table_association" "private-assoc-1" {
    - id           = "rtbassoc-09c3a6aaefcb3e73c" -> null
    - route_table_id = "rtb-0542bf97862b5f10d" -> null
    - subnet_id    = "subnet-047be506958ac9505" -> null
  }

# aws_route_table_association.private-assoc-2 will be destroyed
- resource "aws_route_table_association" "private-assoc-2" {
    - id           = "rtbassoc-000f93d37d0e129be" -> null
    - route_table_id = "rtb-0542bf97862b5f10d" -> null
    - subnet_id    = "subnet-00d390a9c8dfdc977" -> null
  }

# aws_route_table_association.private-assoc-3 will be destroyed
- resource "aws_route_table_association" "private-assoc-3" {
    - id           = "rtbassoc-0df9bc5dc1735ec34" -> null
    - route_table_id = "rtb-0542bf97862b5f10d" -> null
    - subnet_id    = "subnet-0ab2e7c1b4f327350" -> null
  }

# aws_route_table_association.private-assoc-4 will be destroyed
- resource "aws_route_table_association" "private-assoc-4" {
    - id           = "rtbassoc-062c4783aaf0bfdbc" -> null
    - route_table_id = "rtb-0542bf97862b5f10d" -> null
    - subnet_id    = "subnet-03438e4b834bcb577" -> null
  }

# aws_route_table_association.public-assoc-1 will be destroyed
- resource "aws_route_table_association" "public-assoc-1" {
    - id           = "rtbassoc-046c1a6e64d873878" -> null
    - route_table_id = "rtb-0aee36fab0e7b35eb" -> null
    - subnet_id    = "subnet-00b11f9d1117c0027" -> null
  }
```

```
  # aws_route_table_association.public-assoc-2 will be destroyed
  - resource "aws_route_table_association" "public-assoc-2" {
      - id           = "rtbassoc-0722125e07e237008" -> null
      - route_table_id = "rtb-0aee36fab0e7b35eb" -> null
      - subnet_id    = "subnet-08e56a8a4365ac2ef" -> null
    }

  # aws_route_table_association.public-assoc-3 will be destroyed
  - resource "aws_route_table_association" "public-assoc-3" {
      - id           = "rtbassoc-074d0c83a1a3c90a3" -> null
      - route_table_id = "rtb-0aee36fab0e7b35eb" -> null
      - subnet_id    = "subnet-0be0a74e7410aca4f" -> null
    }

  # aws_route_table_association.public-assoc-4 will be destroyed
  - resource "aws_route_table_association" "public-assoc-4" {
      - id           = "rtbassoc-076227cf24d5c3755" -> null
      - route_table_id = "rtb-0aee36fab0e7b35eb" -> null
      - subnet_id    = "subnet-0b3713658a5024d07" -> null
    }

  # aws_subnet.subnet1 will be destroyed
  - resource "aws_subnet" "subnet1" {
      - arn                             = "arn:aws:ec2:ap-southeast-1:964305647536:subnet/subnet-
047be506958ac9505" -> null
      - assign_ipv6_address_on_creation = false -> null
      - availability_zone               = "ap-southeast-1a" -> null
      - availability_zone_id            = "apse1-az1" -> null
      - cidr_block                      = "10.0.1.0/24" -> null
      - id                              = "subnet-047be506958ac9505" -> null
      - map_public_ip_on_launch         = false -> null
      - owner_id                        = "964305647536" -> null
      - tags                            = {
          - "Name" = "app-subnet-1"
        } -> null
      - vpc_id                          = "vpc-0e388a3cde7b45bcb" -> null
    }

  # aws_subnet.subnet2 will be destroyed
  - resource "aws_subnet" "subnet2" {
      - arn                             = "arn:aws:ec2:ap-southeast-1:964305647536:subnet/subnet-
00d390a9c8dfdc977" -> null
      - assign_ipv6_address_on_creation = false -> null
      - availability_zone               = "ap-southeast-1b" -> null
      - availability_zone_id            = "apse1-az2" -> null
      - cidr_block                      = "10.0.2.0/24" -> null
      - id                              = "subnet-00d390a9c8dfdc977" -> null
      - map_public_ip_on_launch         = false -> null
```

```
    - owner_id                      = "964305647536" -> null
    - tags                          = {
        - "Name" = "app-subnet-2"
      } -> null
    - vpc_id                        = "vpc-0e388a3cde7b45bcb" -> null
  }

  # aws_subnet.subnet3 will be destroyed
  - resource "aws_subnet" "subnet3" {
    - arn                           = "arn:aws:ec2:ap-southeast-1:964305647536:subnet/subnet-
00b11f9d1117c0027" -> null
    - assign_ipv6_address_on_creation = false -> null
    - availability_zone             = "ap-southeast-1a" -> null
    - availability_zone_id          = "apse1-az1" -> null
    - cidr_block                    = "10.0.3.0/24" -> null
    - id                            = "subnet-00b11f9d1117c0027" -> null
    - map_public_ip_on_launch       = false -> null
    - owner_id                      = "964305647536" -> null
    - tags                          = {
        - "Name" = "elb-subnet-1"
      } -> null
    - vpc_id                        = "vpc-0e388a3cde7b45bcb" -> null
  }

  # aws_subnet.subnet4 will be destroyed
  - resource "aws_subnet" "subnet4" {
    - arn                           = "arn:aws:ec2:ap-southeast-1:964305647536:subnet/subnet-
08e56a8a4365ac2ef" -> null
    - assign_ipv6_address_on_creation = false -> null
    - availability_zone             = "ap-southeast-1b" -> null
    - availability_zone_id          = "apse1-az2" -> null
    - cidr_block                    = "10.0.4.0/24" -> null
    - id                            = "subnet-08e56a8a4365ac2ef" -> null
    - map_public_ip_on_launch       = false -> null
    - owner_id                      = "964305647536" -> null
    - tags                          = {
        - "Name" = "elb-subnet-2"
      } -> null
    - vpc_id                        = "vpc-0e388a3cde7b45bcb" -> null
  }

  # aws_subnet.subnet5 will be destroyed
  - resource "aws_subnet" "subnet5" {
    - arn                           = "arn:aws:ec2:ap-southeast-1:964305647536:subnet/subnet-
0ab2e7c1b4f327350" -> null
    - assign_ipv6_address_on_creation = false -> null
    - availability_zone             = "ap-southeast-1a" -> null
    - availability_zone_id          = "apse1-az1" -> null
```

```
    - cidr_block                     = "10.0.5.0/24" -> null
    - id                             = "subnet-0ab2e7c1b4f327350" -> null
    - map_public_ip_on_launch        = false -> null
    - owner_id                       = "964305647536" -> null
    - tags                           = {
        - "Name" = "db-subnet-1"
      } -> null
    - vpc_id                         = "vpc-0e388a3cde7b45bcb" -> null
  }

  # aws_subnet.subnet6 will be destroyed
  - resource "aws_subnet" "subnet6" {
    - arn                            = "arn:aws:ec2:ap-southeast-1:964305647536:subnet/subnet-
03438e4b834bcb577" -> null
    - assign_ipv6_address_on_creation = false -> null
    - availability_zone              = "ap-southeast-1b" -> null
    - availability_zone_id           = "apse1-az2" -> null
    - cidr_block                     = "10.0.6.0/24" -> null
    - id                             = "subnet-03438e4b834bcb577" -> null
    - map_public_ip_on_launch        = false -> null
    - owner_id                       = "964305647536" -> null
    - tags                           = {
        - "Name" = "db-subnet-2"
      } -> null
    - vpc_id                         = "vpc-0e388a3cde7b45bcb" -> null
  }

  # aws_subnet.subnet7 will be destroyed
  - resource "aws_subnet" "subnet7" {
    - arn                            = "arn:aws:ec2:ap-southeast-1:964305647536:subnet/subnet-
0be0a74e7410aca4f" -> null
    - assign_ipv6_address_on_creation = false -> null
    - availability_zone              = "ap-southeast-1a" -> null
    - availability_zone_id           = "apse1-az1" -> null
    - cidr_block                     = "10.0.7.0/24" -> null
    - id                             = "subnet-0be0a74e7410aca4f" -> null
    - map_public_ip_on_launch        = false -> null
    - owner_id                       = "964305647536" -> null
    - tags                           = {
        - "Name" = "nat-subnet-1"
      } -> null
    - vpc_id                         = "vpc-0e388a3cde7b45bcb" -> null
  }

  # aws_subnet.subnet8 will be destroyed
  - resource "aws_subnet" "subnet8" {
    - arn                            = "arn:aws:ec2:ap-southeast-1:964305647536:subnet/subnet-
0b3713658a5024d07" -> null
```

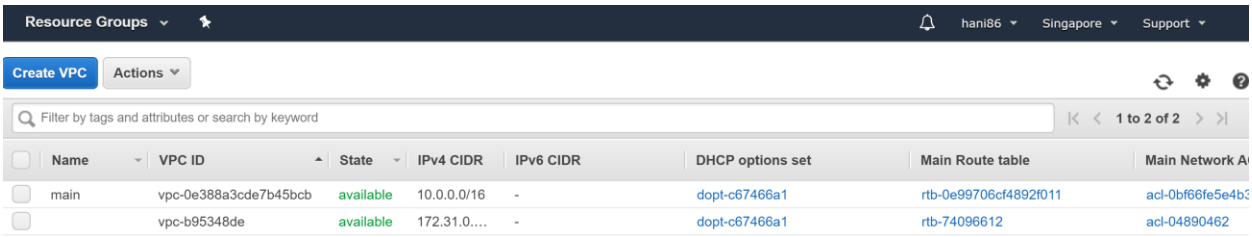
```
    - assign_ipv6_address_on_creation = false -> null
    - availability_zone             = "ap-southeast-1b" -> null
    - availability_zone_id          = "apse1-az2" -> null
    - cidr_block                    = "10.0.8.0/24" -> null
    - id                            = "subnet-0b3713658a5024d07" -> null
    - map_public_ip_on_launch       = false -> null
    - owner_id                      = "964305647536" -> null
    - tags                          = {
        - "Name" = "nat-subnet-2"
      } -> null
    - vpc_id                        = "vpc-0e388a3cde7b45bcb" -> null
  }

 # aws_vpc.main will be destroyed
 - resource "aws_vpc" "main" {
    - arn                            = "arn:aws:ec2:ap-southeast-1:964305647536:vpc/vpc-0e388a3cde7b45bcb" ->
null
    - assign_generated_ipv6_cidr_block = false -> null
    - cidr_block                     = "10.0.0.0/16" -> null
    - default_network_acl_id         = "acl-0bf66fe5e4b3bde3e" -> null
    - default_route_table_id         = "rtb-0e99706cf4892f011" -> null
    - default_security_group_id      = "sg-0094abe5361ecc0e5" -> null
    - dhcp_options_id                = "dopt-c67466a1" -> null
    - enable_classiclink             = false -> null
    - enable_classiclink_dns_support  = false -> null
    - enable_dns_hostnames           = true -> null
    - enable_dns_support             = true -> null
    - id                             = "vpc-0e388a3cde7b45bcb" -> null
    - instance_tenancy               = "default" -> null
    - main_route_table_id            = "rtb-0e99706cf4892f011" -> null
    - owner_id                       = "964305647536" -> null
    - tags                           = {
        - "Name" = "main"
      } -> null
  }

Plan: 0 to add, 0 to change, 22 to destroy.



Warning: Interpolation-only expressions are deprecated

 on aws.tf line 2, in provider "aws":
  2:     access_key = "${var.aws_access_key}"

Terraform 0.11 and earlier required all non-constant expressions to be
provided via interpolation syntax, but this pattern is now deprecated. To
silence this warning, remove the "${ sequence from the start and the }"
sequence from the end of this expression, leaving just the inner expression.
```

```
Template interpolation syntax is still used to construct strings from
expressions when the template includes multiple interpolation sequences or a
mixture of literal strings and interpolations. This deprecation applies only
to templates that consist entirely of a single interpolation sequence.

(and 42 more similar warnings elsewhere)

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_route_table_association.private-assoc-3: Destroying... [id=rtbassoc-0df9bc5dc1735ec34]
aws_route_table_association.private-assoc-2: Destroying... [id=rtbassoc-000f93d37d0e129be]
aws_route_table_association.public-assoc-3: Destroying... [id=rtbassoc-074d0c83a1a3c90a3]
aws_route_table_association.private-assoc-4: Destroying... [id=rtbassoc-062c4783aaf0bfdbc]
aws_route_table_association.public-assoc-2: Destroying... [id=rtbassoc-0722125e07e237008]
aws_route_table_association.public-assoc-1: Destroying... [id=rtbassoc-046c1a6e64d873878]
aws_route_table_association.public-assoc-4: Destroying... [id=rtbassoc-076227cf24d5c3755]
aws_route_table_association.private-assoc-1: Destroying... [id=rtbassoc-09c3a6aaefcb3e73c]
aws_route_table_association.public-assoc-2: Destruction complete after 1s
aws_subnet.subnet4: Destroying... [id=subnet-08e56a8a4365ac2ef]
aws_route_table_association.public-assoc-3: Destruction complete after 1s
aws_subnet.subnet7: Destroying... [id=subnet-0be0a74e7410aca4f]
aws_route_table_association.public-assoc-1: Destruction complete after 1s
aws_subnet.subnet3: Destroying... [id=subnet-00b11f9d1117c0027]
aws_route_table_association.private-assoc-3: Destruction complete after 1s
aws_subnet.subnet5: Destroying... [id=subnet-0ab2e7c1b4f327350]
aws_route_table_association.public-assoc-4: Destruction complete after 1s
aws_route_table_association.private-assoc-2: Destruction complete after 1s
aws_subnet.subnet2: Destroying... [id=subnet-00d390a9c8dfdc977]
aws_route_table.main-public-rt: Destroying... [id=rtb-0aee36fab0e7b35eb]
aws_route_table_association.private-assoc-1: Destruction complete after 1s
aws_subnet.subnet1: Destroying... [id=subnet-047be506958ac9505]
aws_route_table_association.private-assoc-4: Destruction complete after 1s
aws_subnet.subnet6: Destroying... [id=subnet-03438e4b834bcb577]
aws_route_table.main-private-rt: Destroying... [id=rtb-0542bf97862b5f10d]
aws_subnet.subnet3: Destruction complete after 3s
aws_subnet.subnet4: Destruction complete after 3s
aws_subnet.subnet5: Destruction complete after 3s
aws_subnet.subnet1: Destruction complete after 3s
aws_subnet.subnet7: Destruction complete after 3s
aws_subnet.subnet2: Destruction complete after 3s
aws_subnet.subnet6: Destruction complete after 3s
aws_route_table.main-public-rt: Destruction complete after 3s
aws_internet_gateway.main-igw: Destroying... [id=igw-0cafcbb261928cffe]
```

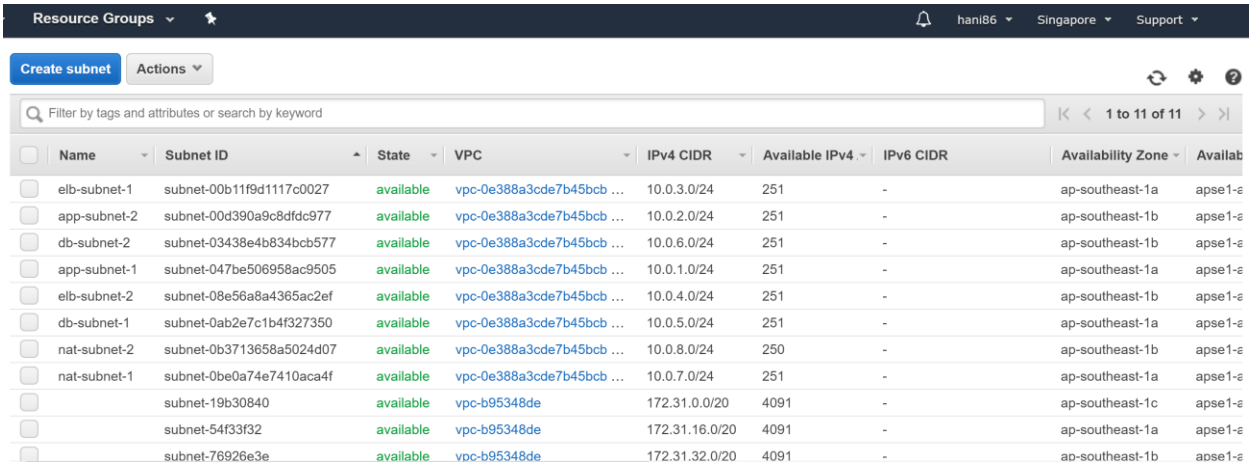
```
aws_route_table.main-private-rt: Destruction complete after 3s
aws_nat_gateway.main-natgw: Destroying... [id=nat-0d67b00a7036d53c4]
aws_internet_gateway.main-igw: Still destroying... [id=igw-0cafcbb261928cffe, 10s elapsed]
aws_nat_gateway.main-natgw: Still destroying... [id=nat-0d67b00a7036d53c4, 10s elapsed]
aws_internet_gateway.main-igw: Still destroying... [id=igw-0cafcbb261928cffe, 20s elapsed]
aws_nat_gateway.main-natgw: Still destroying... [id=nat-0d67b00a7036d53c4, 20s elapsed]
aws_internet_gateway.main-igw: Still destroying... [id=igw-0cafcbb261928cffe, 30s elapsed]
aws_nat_gateway.main-natgw: Still destroying... [id=nat-0d67b00a7036d53c4, 30s elapsed]
aws_internet_gateway.main-igw: Still destroying... [id=igw-0cafcbb261928cffe, 40s elapsed]
aws_nat_gateway.main-natgw: Still destroying... [id=nat-0d67b00a7036d53c4, 40s elapsed]
aws_internet_gateway.main-igw: Destruction complete after 49s
aws_nat_gateway.main-natgw: Still destroying... [id=nat-0d67b00a7036d53c4, 50s elapsed]
aws_nat_gateway.main-natgw: Destruction complete after 56s
aws_subnet.subnet8: Destroying... [id=subnet-0b3713658a5024d07]
aws_eip.nat: Destroying... [id=eipalloc-08a2cad1a6bd39bd4]
aws_eip.nat: Destruction complete after 3s
aws_subnet.subnet8: Destruction complete after 3s
aws_vpc.main: Destroying... [id=vpc-0e388a3cde7b45bcb]
aws_vpc.main: Destruction complete after 1s

Destroy complete! Resources: 22 destroyed.
```

My VPC Resources

VPC - main

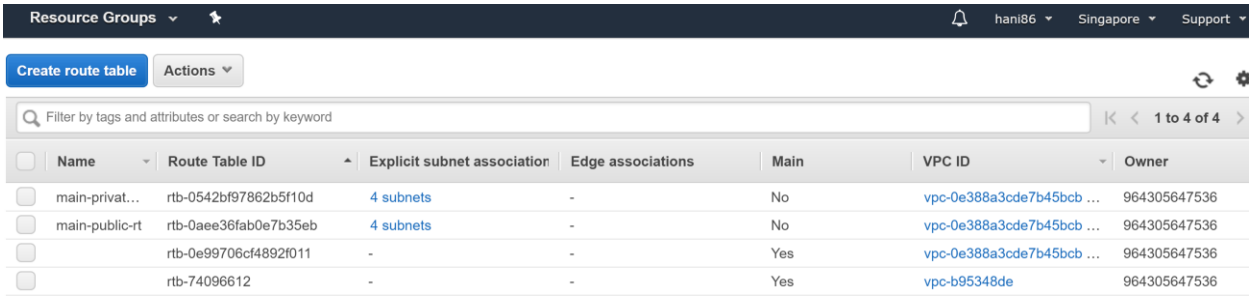


Subnets



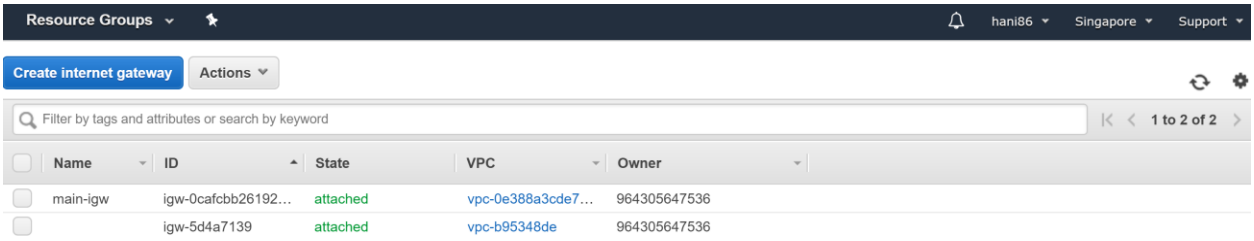
Name	Subnet ID	State	VPC	IPv4 CIDR	Available IPv4	IPv6 CIDR	Availability Zone	Availab
elb-subnet-1	subnet-00b11f9d1117c0027	available	vpc-0e388a3cde7b45bcb ...	10.0.3.0/24	251	-	ap-southeast-1a	apse1-a
app-subnet-2	subnet-00d390a9c8dfdc977	available	vpc-0e388a3cde7b45bcb ...	10.0.2.0/24	251	-	ap-southeast-1b	apse1-a
db-subnet-2	subnet-03438e4b834bcb577	available	vpc-0e388a3cde7b45bcb ...	10.0.6.0/24	251	-	ap-southeast-1b	apse1-a
app-subnet-1	subnet-047be506958ac9505	available	vpc-0e388a3cde7b45bcb ...	10.0.1.0/24	251	-	ap-southeast-1a	apse1-a
elb-subnet-2	subnet-08e56a8a4365ac2ef	available	vpc-0e388a3cde7b45bcb ...	10.0.4.0/24	251	-	ap-southeast-1b	apse1-a
db-subnet-1	subnet-0ab2e7c1b4f327350	available	vpc-0e388a3cde7b45bcb ...	10.0.5.0/24	251	-	ap-southeast-1a	apse1-a
nat-subnet-2	subnet-0b3713658a5024d07	available	vpc-0e388a3cde7b45bcb ...	10.0.8.0/24	250	-	ap-southeast-1b	apse1-a
nat-subnet-1	subnet-0be0a74e7410aca4f	available	vpc-0e388a3cde7b45bcb ...	10.0.7.0/24	251	-	ap-southeast-1a	apse1-a
	subnet-19b30840	available	vpc-b95348de	172.31.0.0/20	4091	-	ap-southeast-1c	apse1-a
	subnet-54f33f32	available	vpc-b95348de	172.31.16.0/20	4091	-	ap-southeast-1a	apse1-a
	subnet-76926e3e	available	vpc-b95348de	172.31.32.0/20	4091	-	ap-southeast-1b	apse1-a

Route Table



Name	Route Table ID	Explicit subnet association	Edge associations	Main	VPC ID	Owner
main-privat...	rtb-0542bf97862b5f10d	4 subnets	-	No	vpc-0e388a3cde7b45bcb ...	964305647536
main-public-rt	rtb-0aee36fab0e7b35eb	4 subnets	-	No	vpc-0e388a3cde7b45bcb ...	964305647536
	rtb-0e99706cf4892f011	-	-	Yes	vpc-0e388a3cde7b45bcb ...	964305647536
	rtb-74096612	-	-	Yes	vpc-b95348de	964305647536

IG

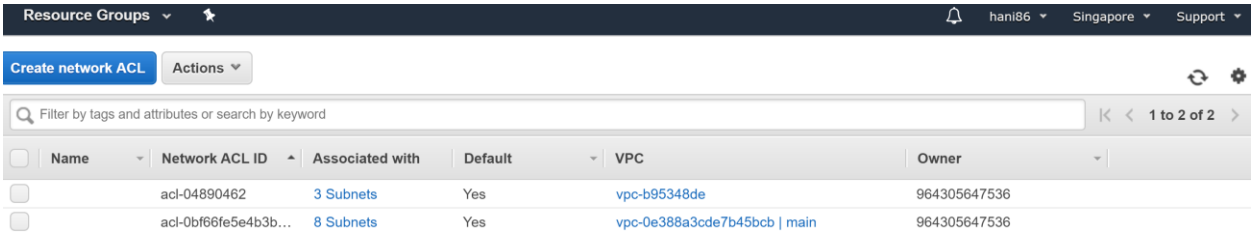


Name	ID	State	VPC	Owner
main-igw	igw-0cafcbb26192...	attached	vpc-0e388a3cde7...	964305647536
	igw-5d4a7139	attached	vpc-b95348de	964305647536

NAT

Name	NAT Gateway ID	Status	Status Message	Elastic IP Address	Private IP Address	Network Interface	VPC	Subnet
main-nat	nat-0d67b00a703...	available	-	18.140.146.63	10.0.8.92	eni-0bab343104c...	vpc-0e388a3cde7...	subnet-0b3

NACL



Security