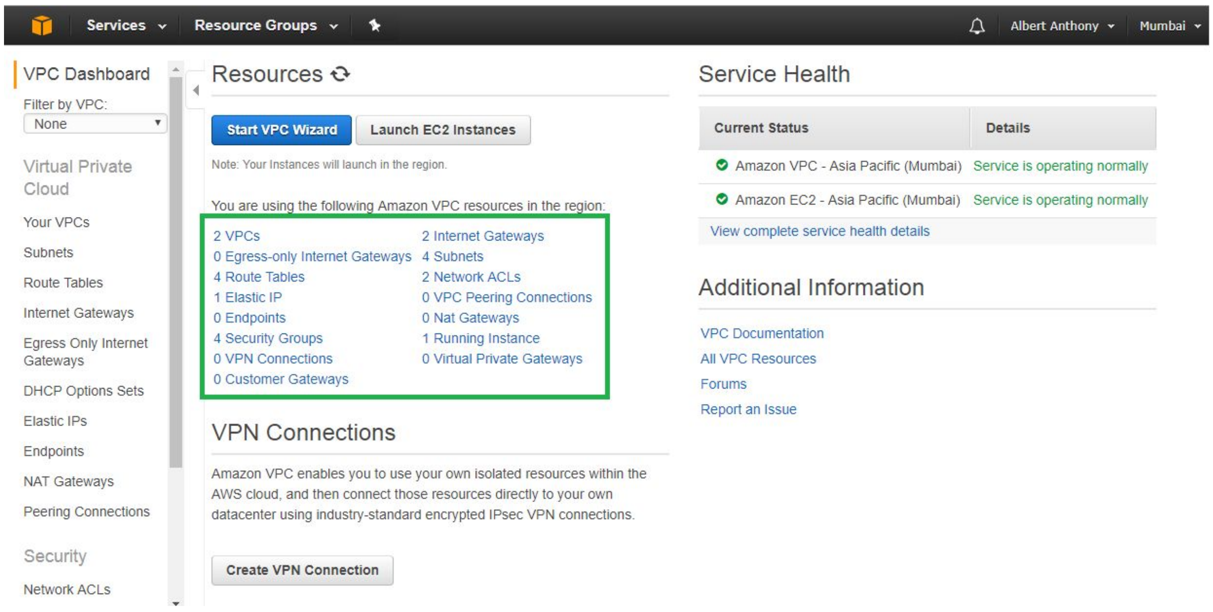
Understanding and Automating Your VPC with Terraform and Jenkins

What is a VPC?

AWS VPC is a logically separated network isolated from other networks. It lets you set your own IP address range and configure security settings and routing for all your traffic. AWS VPC is made up of several networking components.

To summarize, using Amazon VPC you have:

- An isolated environment inside AWS to launch your instances
- A virtual network where you can define rules and policies for your services
- A great solution to have a wall between the Internet and your instances, you can decide to expose only a part of your infrastructure, basically, the only one that needs to speak with the rest of the world
- The opportunity to set up subnets, IP ranges, and network configurations as you prefer.



Resources

Note: Your Instances will launch in the region.

You are using the following Amazon VPC resources in the region:

2 VPCs	2 Internet Gateways
0 Egress-only Internet Gateways	4 Subnets
4 Route Tables	2 Network ACLs
1 Elastic IP	0 VPC Peering Connections
0 Endpoints	0 Nat Gateways
4 Security Groups	1 Running Instance
0 VPN Connections	0 Virtual Private Gateways
0 Customer Gateways	

VPN Connections

Amazon VPC enables you to use your own isolated resources within the AWS cloud, and then connect those resources directly to your own datacenter using industry-standard encrypted IPsec VPN connections.

Service Health

Current Status	Details
Amazon VPC - Asia Pacific (Mumbai)	Service is operating normally
Amazon EC2 - Asia Pacific (Mumbai)	Service is operating normally

View complete service health details

Additional Information

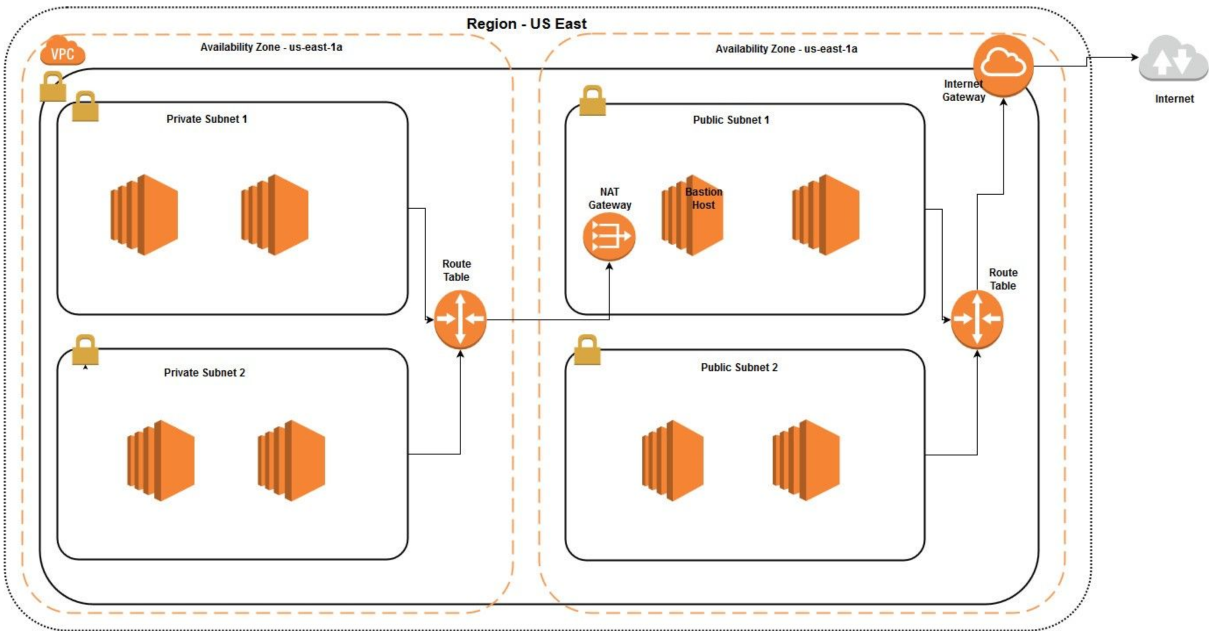
As shown in the following figure; some of them are as follows VPC components:

- **AWS regions:** An AWS Region is a geographical location with a collection of availability zones mapped to physical data centers in that region. Every region is physically isolated from and independent of every other region in terms of location, power, water supply, etc.
  This level of isolation is critical for workloads with compliance and data sovereignty requirements where guarantees must be made that user data does not leave a particular geographic region. The presence of AWS regions worldwide is also important for workloads that are latency-sensitive and need to be located near users in a particular geographic area.
  Inside each region, you will find two or more availability zones with each zone hosted in separate data centers from another zone. I'll explain more later on why having at least two zones in a region is important.

The largest AWS region, us-east-1, has five zones. Moving forward, new AWS regions will have three or more zones whenever possible. When you create certain resources in a region, you will be asked to choose a zone in which to host that resource.

- **VPC CIDR range:** When designing your Amazon VPC instance, you must consider the number of IP addresses required and the connectivity type with the data center before choosing the **CIDR block**. The permissible size of the block ranges between /16 netmask and a /28 netmask.
  As of now, you cannot alter or modify Amazon VPC, so it is better to pick the CIDR block that has more IP addresses. While designing your Amazon VPC architecture to communicate with the on-premises data center, it is required that the CIDR range used in Amazon VPC does not overlap or cause a conflict with the CIDR block in the on-premises data center.

- **Subnets:** A VPC spans an AWS region. A region contains two or more availability zones. A VPC contains subnets that are used to logically separate resources inside a region. A subnet cannot span across multiple availability zones. A subnet can either be a private subnet or a public subnet based on its accessibility from outside of VPC and if it can access resources outside of VPC. Subnets are used for separating resources, such as web servers and database servers.



- **Availability Zones:** An **availability** zone is a logical data center in a region available for use by any **AWS** customer. Each zone in a region has redundant and separate power, networking and connectivity to reduce the likelihood of two **zones** failing simultaneously. A common misconception is that a single zone equals a single data center.

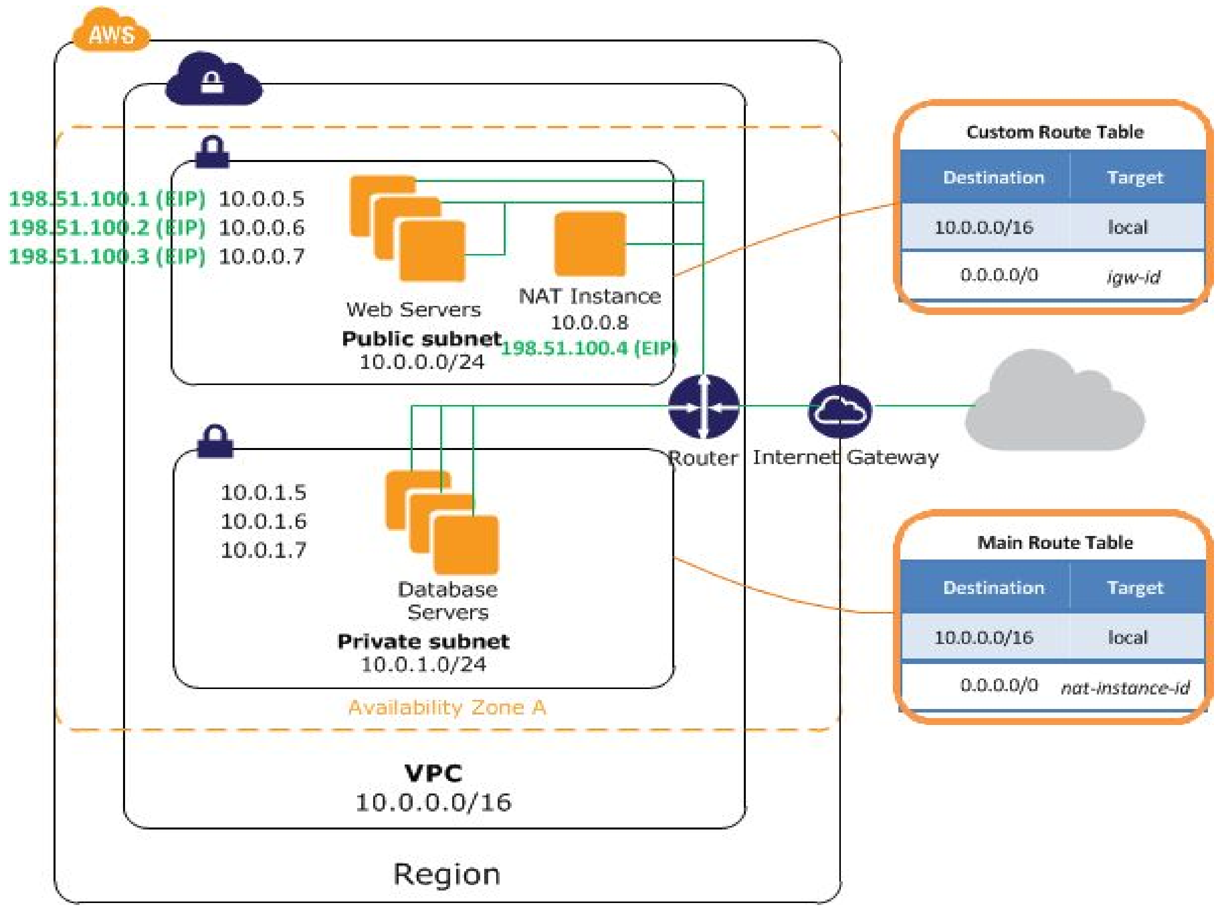
- **Internet gateways:** The internet gateway allows our bi-directional internet access to and from services within our VPC. Note that only resources in our "public" subnets will have access to an internet gateway.

- **NAT:** Network Address Translation (NAT) gateway enable services in our private subnets to connect to the internet or other AWS services, but prevent the internet from initiating a connection with those instances. This is needed for package and patch installs on our private EC2 instances.

- **Route Table:** These tables are a set of rules or ***routes*** that govern where traffic is routed. We attach a routing table to each subnet. Our public subnets will have a route to the **Internet Gateway** and our private subnets will have a route to the **NAT Gateway**.

- **Network ACLs:** A network access control list (ACL) is an optional layer of security for your VPC that acts as a firewall for controlling traffic in and out of one or more subnets. You might set up network ACLs with rules similar to your security groups in order to add an additional layer of security to your VPC.

- **Security Group: AWS security groups** (SGs) are associated with **EC2** instances and provide **security** at the protocol and port access level. Each **security group** — working much the same way as a firewall — contains a set of rules that filter traffic coming into and out of an **EC2** instance.



Custom Route Table

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	*igw-id*

Main Route Table

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	*nat-instance-id*

What are we going to build?
We have to decide the design for the VPC base on the requirement. Here we are going to create **VPC** with **three private subnets** and **three public subnets** as show in the above diagram. Let's do the implementation step by step.

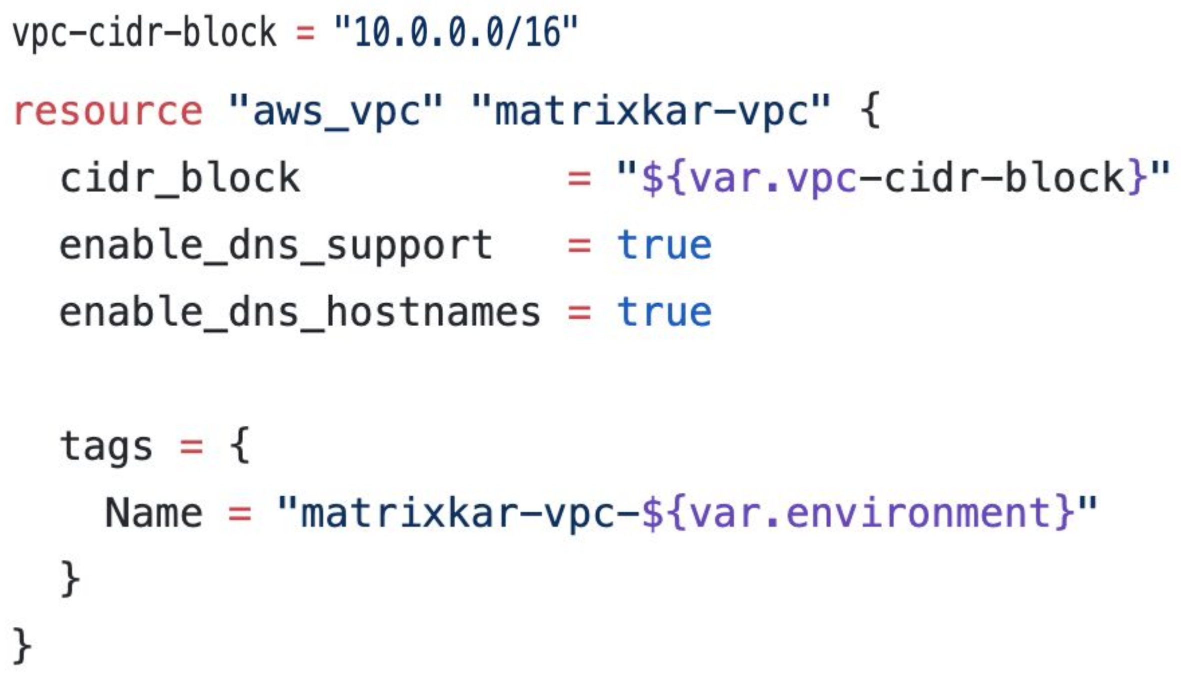
Design the VPC architecture
Decide the IP range for the **VPC** before creating the **VPC**. Here we are using **10.0.0.0/16** as the CIDR block.This CIDR block assigns 256 IP address for the VPC. Hence we are using 6 subnets for the VPC we have to divide the 256 IP addresses among 6 subnets.
CIDR block for the VPC -> 10.0.0.0/16
Number of IP address -> 2^(32–16) -> 65534
Here is the IP address distribution for the above 6 subnets

Public Subnet 1 -> 10.0.0.0/24 (IP range - 10.0.0.1 - 10.0.0.254)
Public Subnet 2 -> 10.0.2.0/24 (IP range - 10.0.2.1 - 10.0.2.254)
Public Subnet 3 -> 10.0.4.0/24 (IP range - 10.0.4.1 - 10.0.4.254)
Private Subnet 1 -> 10.0.1.0/24 (IP range - 10.0.1.1 - 10.0.1.254)
Private Subnet 2 -> 10.0.3.0/24 (IP range - 10.0.3.1 - 10.0.3.254)
Private Subnet 3 -> 10.0.5.0/24 (IP range - 10.0.5.1 - 10.0.5.254)

We are going to implement VPC with Terraform according to the above details
We can identify the **Terraform** script steps as below:

1. **Defining a VPC.**

Resource type: AWS::EC2::VPC
VPC CIDR range is hardcoded in here. Also, we may parameterize the CIRD range by using the **variables/development.tfvars**.



```
vpc-cidr-block = "10.0.0.0/16"

resource "aws_vpc" "matrixkar-vpc" {
  cidr_block           = "${var.vpc-cidr-block}"
  enable_dns_support   = true
  enable_dns_hostnames = true


  tags = {
    Name = "matrixkar-vpc-${var.environment}"
  }
}
```

2. **Defining public subnets.**

Resource type: AWS::EC2::Subnet

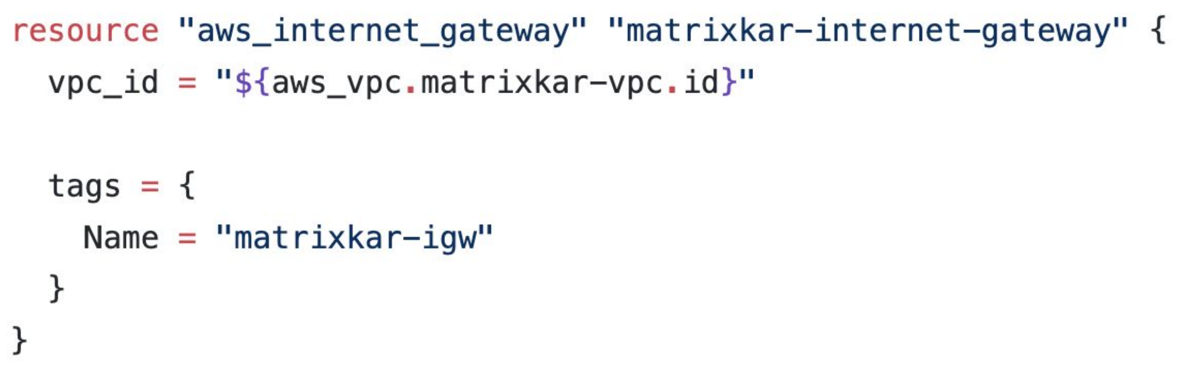
As per our infrastructure diagram, we do have three public subnets with three separate CIDR ranges. These three subnets will be provisioned into three separate availability zones us-east-1a,us-east-1b and us-east-1c.

```
public-subnets = ["10.0.0.0/24", "10.0.2.0/24", "10.0.4.0/24"]
```

```
resource "aws_subnet" "matrixkar-subnet-01" {
  availability_zone = "${var.region}a"
  vpc_id            = "${aws_vpc.matrixkar-vpc.id}"
  cidr_block        = "${element(var.public-subnets, 0)}"

  tags = {
    Name = "matrixkar-public-subnet-01"
  }
}

resource "aws_subnet" "matrixkar-subnet-02" {
  availability_zone = "${var.region}b"
  vpc_id            = "${aws_vpc.matrixkar-vpc.id}"
  cidr_block        = "${element(var.public-subnets, 1)}"

  tags = {
    Name = "matrixkar-public-subnet-02"
  }
}

resource "aws_subnet" "matrixkar-subnet-03" {
  availability_zone = "${var.region}c"
  vpc_id            = "${aws_vpc.matrixkar-vpc.id}"
  cidr_block        = "${element(var.public-subnets, 2)}"

  tags = {
    Name = "matrixkar-public-subnet-03"
  }
}
```

3. **Defining an internet gateway.**

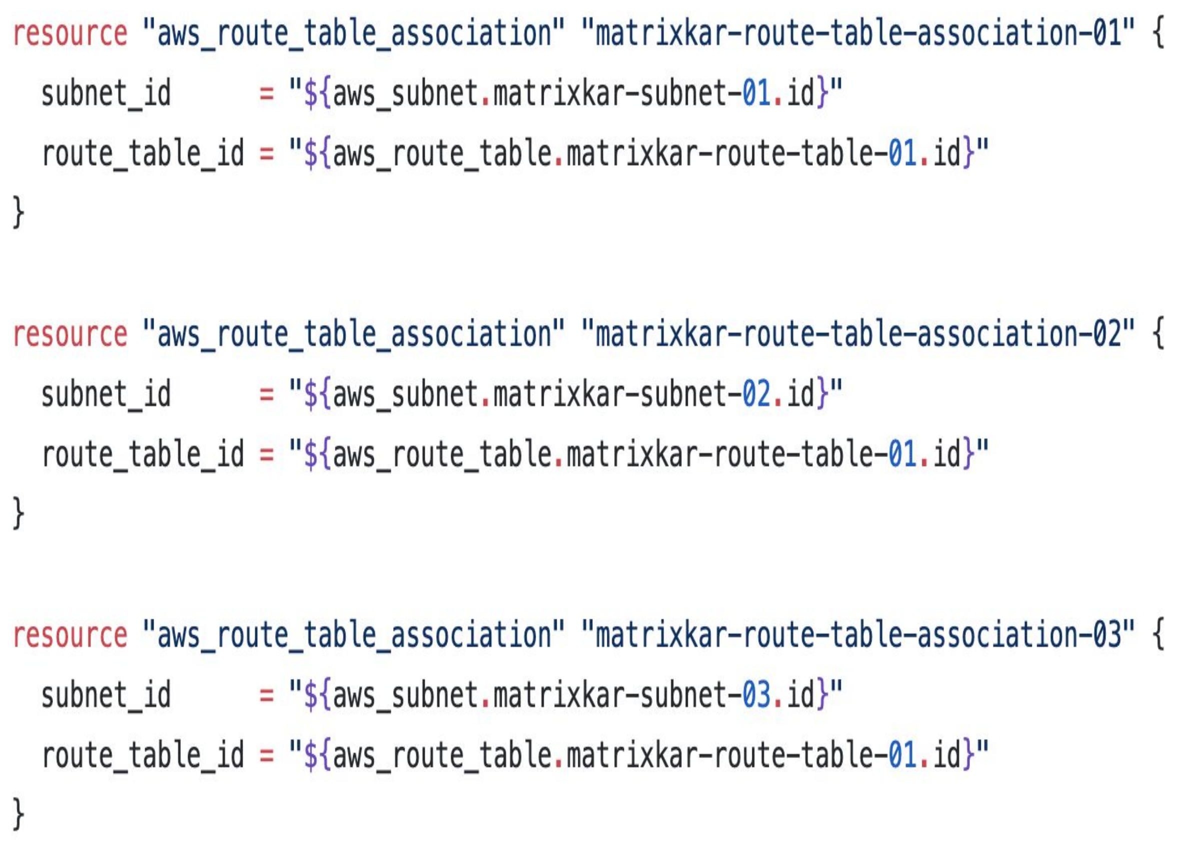
Resource types: AWS::EC2::InternetGateway, AWS::EC2::VPCGatewayAttachment
In here we are defining an internet gateway and attaching it to the VPC.



```
resource "aws_internet_gateway" "matrixkar-internet-gateway" {
  vpc_id = "${aws_vpc.matrixkar-vpc.id}"

  tags = {
    Name = "matrixkar-igw"
  }
}
```

4. **Defining public route table and routes, associate those with public subnets.**

Resource types: AWS::EC2::RouteTable, AWS::EC2::Route, AWS::EC2::SubnetRouteTableAssociation
In here we are defining a route table and route by assigning an above-created internet gateway. Then the subnet will be associated with this route table.



```
resource "aws_route_table_association" "matrixkar-route-table-association-01" {
  subnet_id      = "${aws_subnet.matrixkar-subnet-01.id}"
  route_table_id = "${aws_route_table.matrixkar-route-table-01.id}"
}

resource "aws_route_table_association" "matrixkar-route-table-association-02" {
  subnet_id      = "${aws_subnet.matrixkar-subnet-02.id}"
  route_table_id = "${aws_route_table.matrixkar-route-table-01.id}"
}

resource "aws_route_table_association" "matrixkar-route-table-association-03" {
  subnet_id      = "${aws_subnet.matrixkar-subnet-03.id}"
  route_table_id = "${aws_route_table.matrixkar-route-table-01.id}"
}
```

5. **Defining private subnets.**

Resource type: AWS::EC2::Subnet

```
private-subnets = ["10.0.1.0/24", "10.0.3.0/24", "10.0.5.0/24"]

resource "aws_subnet" "matrixkar-subnet-04" {
  availability_zone = "${var.region}a"
  vpc_id            = "${aws_vpc.matrixkar-vpc.id}"
  cidr_block        = "${element(var.private-subnets, 0)}"

  tags = {
    Name = "matrixkar-private-subnet-01"
  }
}

resource "aws_subnet" "matrixkar-subnet-05" {
  availability_zone = "${var.region}b"
  vpc_id            = "${aws_vpc.matrixkar-vpc.id}"
  cidr_block        = "${element(var.private-subnets, 1)}"

  tags = {
    Name = "matrixkar-private-subnet-02"
  }
}

resource "aws_subnet" "matrixkar-subnet-06" {
  availability_zone = "${var.region}c"
  vpc_id            = "${aws_vpc.matrixkar-vpc.id}"
  cidr_block        = "${element(var.private-subnets, 2)}"

  tags = {
    Name = "matrixkar-private-subnet-03"
  }
}
```

6. **Defining Elastic IP and NAT gateway.**

Resource types: AWS::EC2::EIP, AWS::EC2::NatGateway
In here we are defining an EIP and NAT gateway.

```
resource "aws_eip" "matrixkar-eip" {
  vpc = true
}

resource "aws_nat_gateway" "matrixkar-nat-gateway" {
  allocation_id = "${aws_eip.matrixkar-eip.id}"
  subnet_id     = "${aws_subnet.matrixkar-subnet-01.id}"

  tags = {
    Name = "matrixkar-nat-gateway-01"
  }

  depends_on = ["aws_internet_gateway.matrixkar-internet-gateway"]
}
```

7. **Defining private route table and routes, associate those with private subnets.**

Resource types: AWS::EC2::RouteTable, AWS::EC2::Route, AWS::EC2::SubnetRouteTableAssociation
In here we are defining a route table and route by assigning an above-created NAT gateway. Then subnet will be associated with this route table.

```
resource "aws_route_table_association" "matrixkar-route-table-association-04" {
  subnet_id      = "${aws_subnet.matrixkar-subnet-04.id}"
  route_table_id = "${aws_route_table.matrixkar-route-table-02.id}"
}

resource "aws_route_table_association" "matrixkar-route-table-association-05" {
  subnet_id      = "${aws_subnet.matrixkar-subnet-05.id}"
  route_table_id = "${aws_route_table.matrixkar-route-table-02.id}"
}

resource "aws_route_table_association" "matrixkar-route-table-association-06" {
  subnet_id      = "${aws_subnet.matrixkar-subnet-06.id}"
  route_table_id = "${aws_route_table.matrixkar-route-table-02.id}"
}
```

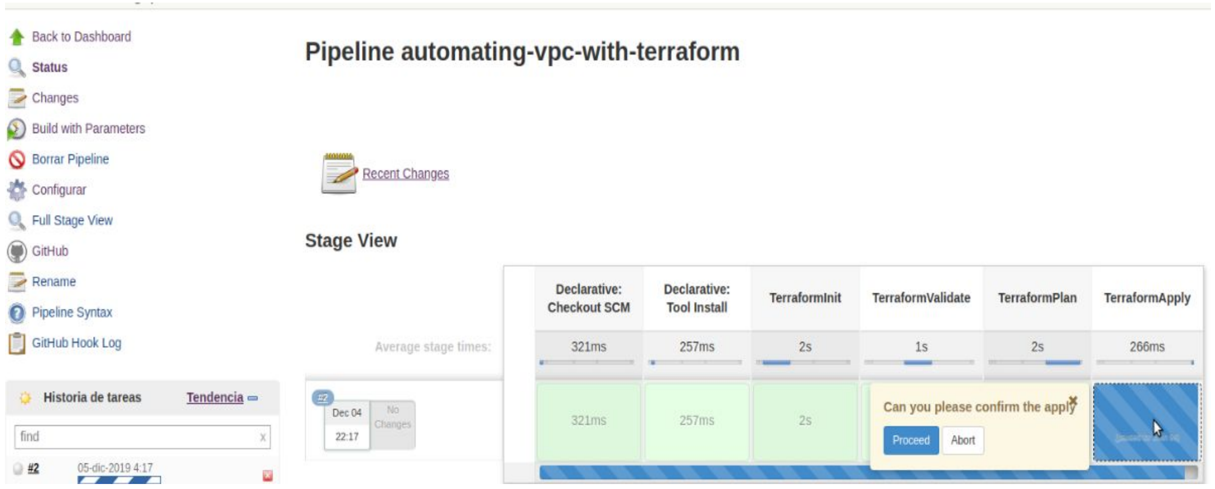
Notes:

Note 1: Create other 3 subnets also as mentioned above diagram. Insert the IPv4 CIDR block according to the IP ranges decided in the step 1. Keep matrixkar-public-subnet-01, matrixkar-public-subnet-02 and matrixkar-public-subnet-03 inside the differentes availability zones and matrixkar-private-subnet-01,matrixkar-private-subnet-02 and matrixkar-private-subnet-03 inside the differents availability zones to increase the availability. You can see all the subnets as follows once completed the process. 5 IP addresses out of all the IP addresses of the subnet are reserved for the internal usage.This is the reason to display number of **Available IPv4 as 251**.

Note 2: One route table can be associated with multiple subnets. But one subnet can be associated with only one route table.

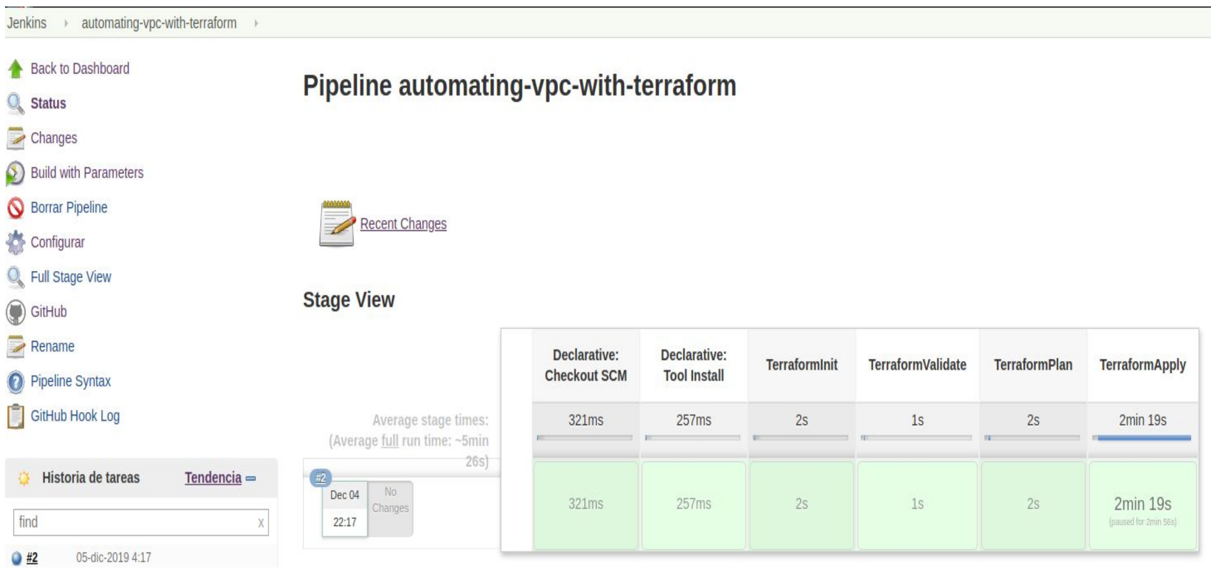
Note 3: Elastic IP address works as public address and it will visible to outside.
We are done with our VPC design. We have to create private instance inside private subnet and public instance inside public subnet.

Run and approve pipeline with **Jenkinsfile** to launch **VPC** resources.



Pipeline automating-vpc-with-terraform

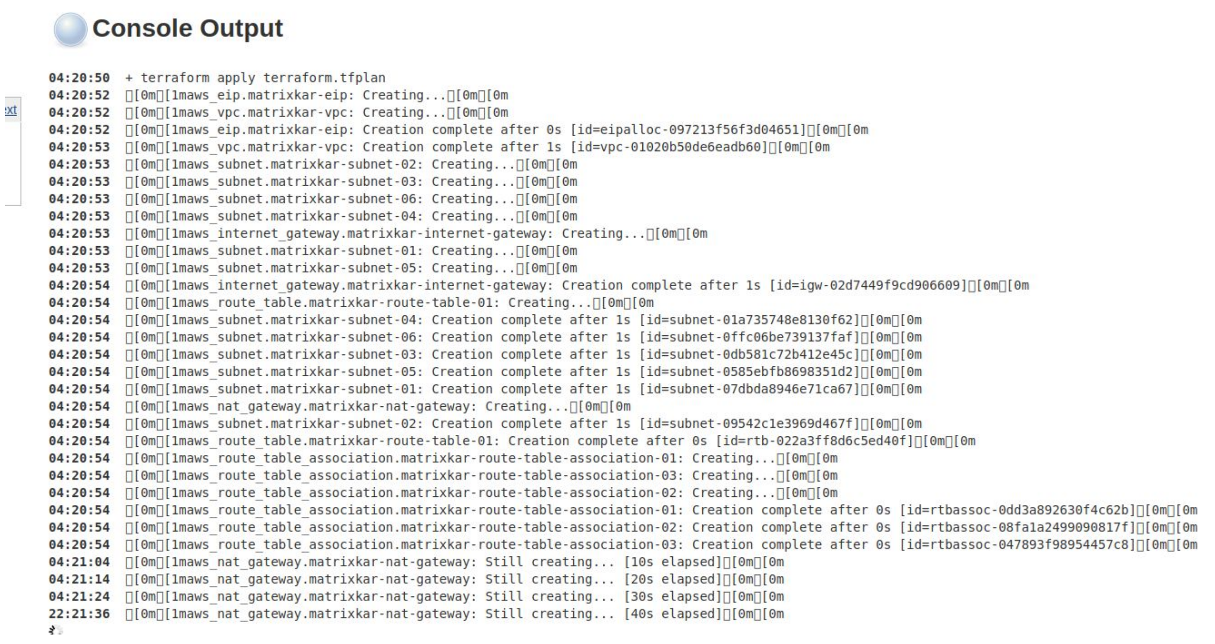
	Declarative: Checkout SCM	Declarative: Tool Install	TerraformInit	TerraformValidate	TerraformPlan	TerraformApply
Average stage times:	321ms	257ms	2s	1s	2s	266ms
#2 Dec 04 22:17 No Changes	321ms	257ms	2s	Can you please confirm the apply?		



Pipeline automating-vpc-with-terraform

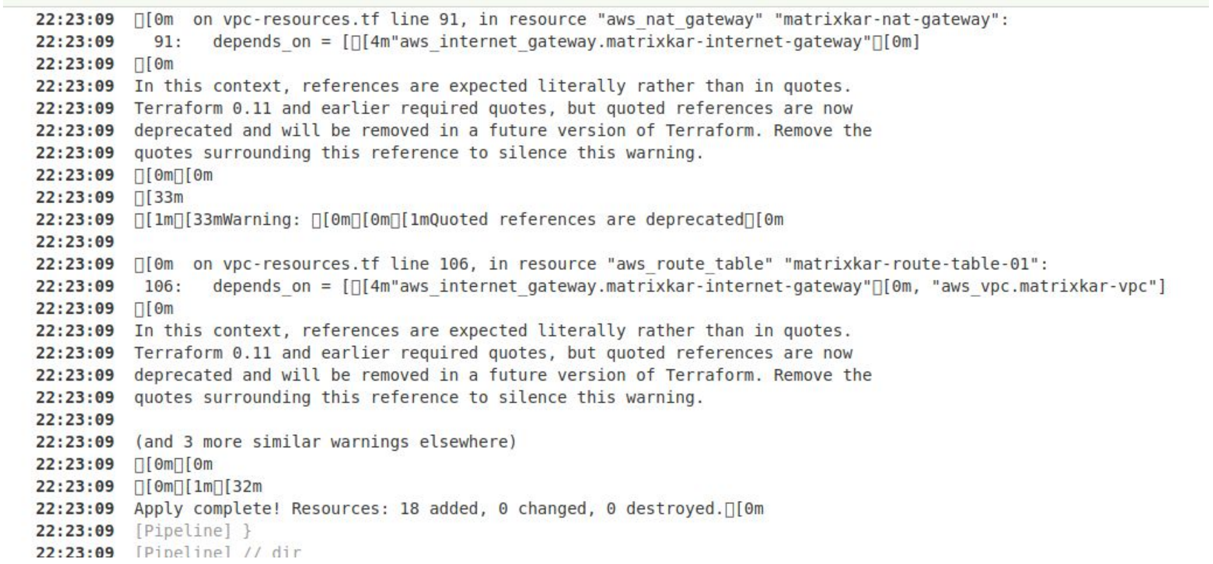
	Declarative: Checkout SCM	Declarative: Tool Install	TerraformInit	TerraformValidate	TerraformPlan	TerraformApply
Average stage times: (Average full run time: ~5min 26s)	321ms	257ms	2s	1s	2s	2min 19s
#2 Dec 04 22:17 No Changes	321ms	257ms	2s	1s	2s	2min 19s (paused for 2min 56s)

In the output console we can observe the records of resources that terraform is generating in the AWS account.



Console Output

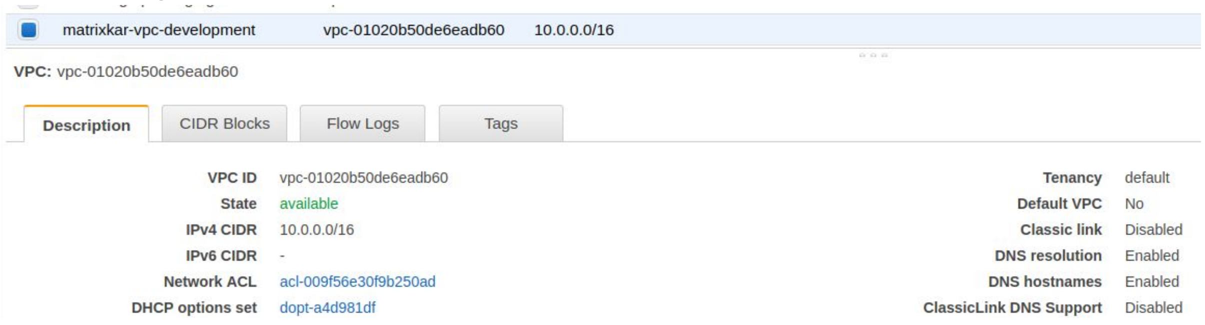
```
04:20:50  + terraform apply terraform.tfplan
04:20:52  [0m[1maws_eip.matrixkar-eip: Creating...[0m[0m
04:20:52  [0m[1maws_vpc.matrixkar-vpc: Creating...[0m[0m
04:20:52  [0m[1maws_eip.matrixkar-eip: Creation complete after 0s [id=eipalloc-097213f56f3d04651][0m[0m
04:20:53  [0m[1maws_vpc.matrixkar-vpc: Creation complete after 1s [id=vpc-01020b50de6eadb60][0m[0m
04:20:53  [0m[1maws_subnet.matrixkar-subnet-02: Creating...[0m[0m
04:20:53  [0m[1maws_subnet.matrixkar-subnet-03: Creating...[0m[0m
04:20:53  [0m[1maws_subnet.matrixkar-subnet-06: Creating...[0m[0m
04:20:53  [0m[1maws_subnet.matrixkar-subnet-04: Creating...[0m[0m
04:20:53  [0m[1maws_internet_gateway.matrixkar-internet-gateway: Creating...[0m[0m
04:20:53  [0m[1maws_subnet.matrixkar-subnet-01: Creating...[0m[0m
04:20:53  [0m[1maws_subnet.matrixkar-subnet-05: Creating...[0m[0m
04:20:54  [0m[1maws_internet_gateway.matrixkar-internet-gateway: Creation complete after 1s [id=igw-02d7449f9cd906609][0m[0m
04:20:54  [0m[1maws_route_table.matrixkar-route-table-01: Creating...[0m[0m
04:20:54  [0m[1maws_subnet.matrixkar-subnet-04: Creation complete after 1s [id=subnet-01a735748e8130f62][0m[0m
04:20:54  [0m[1maws_subnet.matrixkar-subnet-06: Creation complete after 1s [id=subnet-0ffc06be739137faf][0m[0m
04:20:54  [0m[1maws_subnet.matrixkar-subnet-03: Creation complete after 1s [id=subnet-0db581c72b412e45c][0m[0m
04:20:54  [0m[1maws_subnet.matrixkar-subnet-05: Creation complete after 1s [id=subnet-0585ebfb8698351d2][0m[0m
04:20:54  [0m[1maws_subnet.matrixkar-subnet-01: Creation complete after 1s [id=subnet-07dbda8946e71ca67][0m[0m
04:20:54  [0m[1maws_nat_gateway.matrixkar-nat-gateway: Creating...[0m[0m
04:20:54  [0m[1maws_subnet.matrixkar-subnet-02: Creation complete after 1s [id=subnet-09542c1e3969d467f][0m[0m
04:20:54  [0m[1maws_route_table.matrixkar-route-table-01: Creation complete after 0s [id=rtb-022a3ff8d6c5ed40f][0m[0m
04:20:54  [0m[1maws_route_table_association.matrixkar-route-table-association-01: Creating...[0m[0m
04:20:54  [0m[1maws_route_table_association.matrixkar-route-table-association-03: Creating...[0m[0m
04:20:54  [0m[1maws_route_table_association.matrixkar-route-table-association-02: Creating...[0m[0m
04:20:54  [0m[1maws_route_table_association.matrixkar-route-table-association-01: Creation complete after 0s [id=rtbassoc-0dd3a892630f4c62b][0m[0m
04:20:54  [0m[1maws_route_table_association.matrixkar-route-table-association-02: Creation complete after 0s [id=rtbassoc-08fa1a2499090817f][0m[0m
04:20:54  [0m[1maws_route_table_association.matrixkar-route-table-association-03: Creation complete after 0s [id=rtbassoc-047893f98954457c8][0m[0m
04:21:04  [0m[1maws_nat_gateway.matrixkar-nat-gateway: Still creating... [10s elapsed][0m[0m
04:21:14  [0m[1maws_nat_gateway.matrixkar-nat-gateway: Still creating... [20s elapsed][0m[0m
04:21:24  [0m[1maws_nat_gateway.matrixkar-nat-gateway: Still creating... [30s elapsed][0m[0m
22:21:36  [0m[1maws_nat_gateway.matrixkar-nat-gateway: Still creating... [40s elapsed][0m[0m
```



```
22:23:09  [0m  on vpc-resources.tf line 91, in resource "aws_nat_gateway" "matrixkar-nat-gateway":
22:23:09    91:   depends_on = [[4m"aws_internet_gateway.matrixkar-internet-gateway"[0m]
22:23:09  [0m
22:23:09  In this context, references are expected literally rather than in quotes.
22:23:09  Terraform 0.11 and earlier required quotes, but quoted references are now
22:23:09  deprecated and will be removed in a future version of Terraform. Remove the
22:23:09  quotes surrounding this reference to silence this warning.
22:23:09  [0m[0m
22:23:09  [33m
22:23:09  [1m[33mWarning: [0m[0m[1mQuoted references are deprecated[0m
22:23:09
22:23:09  [0m  on vpc-resources.tf line 106, in resource "aws_route_table" "matrixkar-route-table-01":
22:23:09   106:   depends_on = [[4m"aws_internet_gateway.matrixkar-internet-gateway"[0m, "aws_vpc.matrixkar-vpc"]
22:23:09  [0m
22:23:09  In this context, references are expected literally rather than in quotes.
22:23:09  Terraform 0.11 and earlier required quotes, but quoted references are now
22:23:09  deprecated and will be removed in a future version of Terraform. Remove the
22:23:09  quotes surrounding this reference to silence this warning.
22:23:09
22:23:09  (and 3 more similar warnings elsewhere)
22:23:09  [0m[0m
22:23:09  [0m[1m[32m
22:23:09  Apply complete! Resources: 18 added, 0 changed, 0 destroyed.[0m
22:23:09  [Pipeline] }
22:23:09  [Pipeline] // dir
```

At the end, we can observe in the AWS console the VPC resources defined with Terraform:

Resource type: AWS::EC2::VPC

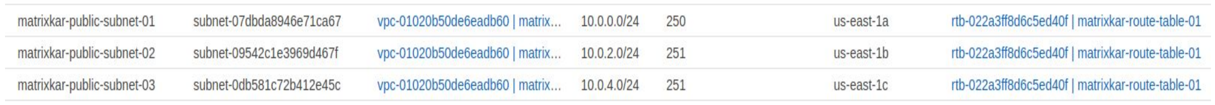


matrixkar-vpc-development   vpc-01020b50de6eadb60   10.0.0.0/16

VPC: vpc-01020b50de6eadb60

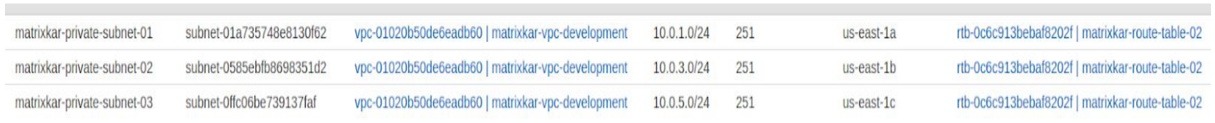
VPC ID	vpc-01020b50de6eadb60	Tenancy	default
State	available	Default VPC	No
IPv4 CIDR	10.0.0.0/16	Classic link	Disabled
IPv6 CIDR	-	DNS resolution	Enabled
Network ACL	acl-009f56e30f9b250ad	DNS hostnames	Enabled
DHCP options set	dopt-a4d981df	ClassicLink DNS Support	Disabled

Resource type: AWS::EC2::Subnet - Public



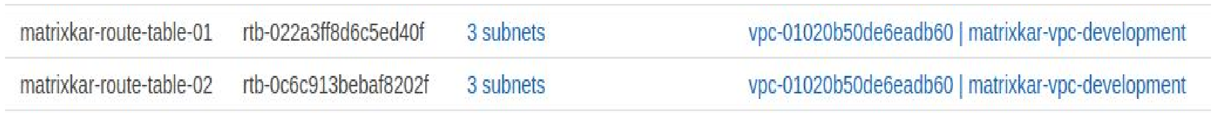
matrixkar-public-subnet-01	subnet-07dbda8946e71ca67	vpc-01020b50de6eadb60 | matrix...	10.0.0.0/24	250	us-east-1a	rtb-022a3ff8d6c5ed40f | matrixkar-route-table-01
matrixkar-public-subnet-02	subnet-09542c1e3969d467f	vpc-01020b50de6eadb60 | matrix...	10.0.2.0/24	251	us-east-1b	rtb-022a3ff8d6c5ed40f | matrixkar-route-table-01
matrixkar-public-subnet-03	subnet-0db581c72b412e45c	vpc-01020b50de6eadb60 | matrix...	10.0.4.0/24	251	us-east-1c	rtb-022a3ff8d6c5ed40f | matrixkar-route-table-01

Resource type: AWS::EC2::Subnet - Private

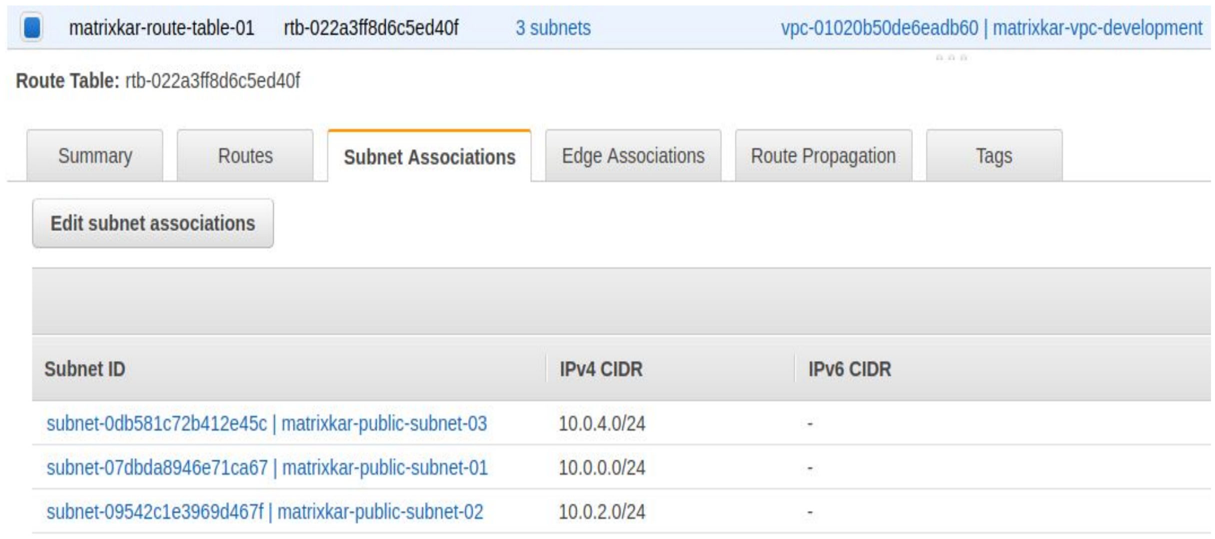


matrixkar-private-subnet-01	subnet-01a735748e8130f62	vpc-01020b50de6eadb60 | matrixkar-vpc-development	10.0.1.0/24	251	us-east-1a	rtb-0c6c913bebaf8202f | matrixkar-route-table-02
matrixkar-private-subnet-02	subnet-0585ebfb8698351d2	vpc-01020b50de6eadb60 | matrixkar-vpc-development	10.0.3.0/24	251	us-east-1b	rtb-0c6c913bebaf8202f | matrixkar-route-table-02
matrixkar-private-subnet-03	subnet-0ffc06be739137faf	vpc-01020b50de6eadb60 | matrixkar-vpc-development	10.0.5.0/24	251	us-east-1c	rtb-0c6c913bebaf8202f | matrixkar-route-table-02

Resource types: AWS::EC2::RouteTable, AWS::EC2::Route, AWS::EC2::SubnetRouteTableAssociation



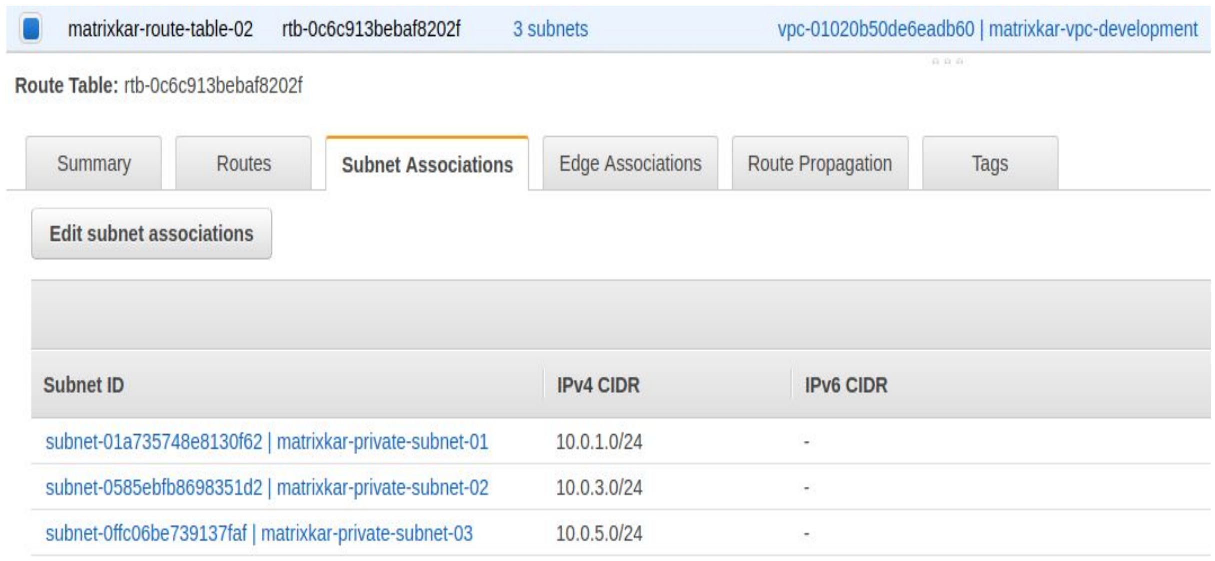
matrixkar-route-table-01	rtb-022a3ff8d6c5ed40f	3 subnets	vpc-01020b50de6eadb60 | matrixkar-vpc-development
matrixkar-route-table-02	rtb-0c6c913bebaf8202f	3 subnets	vpc-01020b50de6eadb60 | matrixkar-vpc-development



matrixkar-route-table-01	rtb-022a3ff8d6c5ed40f	3 subnets	vpc-01020b50de6eadb60 | matrixkar-vpc-development

Route Table: rtb-022a3ff8d6c5ed40f

Subnet ID	IPv4 CIDR	IPv6 CIDR
subnet-0db581c72b412e45c | matrixkar-public-subnet-03	10.0.4.0/24	-
subnet-07dbda8946e71ca67 | matrixkar-public-subnet-01	10.0.0.0/24	-
subnet-09542c1e3969d467f | matrixkar-public-subnet-02	10.0.2.0/24	-

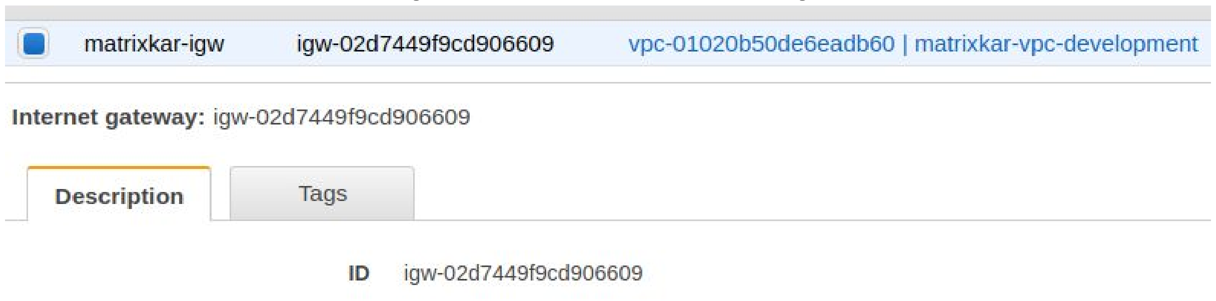


matrixkar-route-table-02	rtb-0c6c913bebaf8202f	3 subnets	vpc-01020b50de6eadb60 | matrixkar-vpc-development

Route Table: rtb-0c6c913bebaf8202f

Subnet ID	IPv4 CIDR	IPv6 CIDR
subnet-01a735748e8130f62 | matrixkar-private-subnet-01	10.0.1.0/24	-
subnet-0585ebfb8698351d2 | matrixkar-private-subnet-02	10.0.3.0/24	-
subnet-0ffc06be739137faf | matrixkar-private-subnet-03	10.0.5.0/24	-

Resource types: AWS::EC2::InternetGateway,AWS::EC2::VPCGatewayAttachment

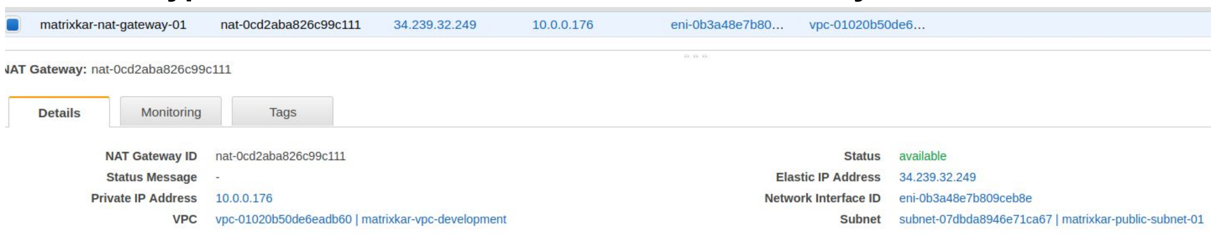


matrixkar-igw igw-02d7449f9cd906609 vpc-01020b50de6eadb60 | matrixkar-vpc-development

Internet gateway: igw-02d7449f9cd906609

ID igw-02d7449f9cd906609

Resource types: AWS::EC2::EIP, AWS::EC2::NatGateway



matrixkar-nat-gateway-01 nat-0cd2aba826c99c111 34.239.32.249 10.0.0.176 eni-0b3a48e7b80… vpc-01020b50de6…

NAT Gateway: nat-0cd2aba826c99c111

NAT Gateway ID	nat-0cd2aba826c99c111	**Status**	available
Status Message	-	**Elastic IP Address**	34.239.32.249
Private IP Address	10.0.0.176	**Network Interface ID**	eni-0b3a48e7b809ceb8e
VPC	vpc-01020b50de6eadb60 | matrixkar-vpc-development	**Subnet**	subnet-07dbda8946e71ca67 | matrixkar-public-subnet-01

Completed project can be found here:
https://github.com/matrixkar/automating-vpc-with-terraform-and-jenkins.git

References:

AWS Certified Advanced Networking Official Study Guide
Mastering AWS Security
Amazon VPC - Practical AWS Networking