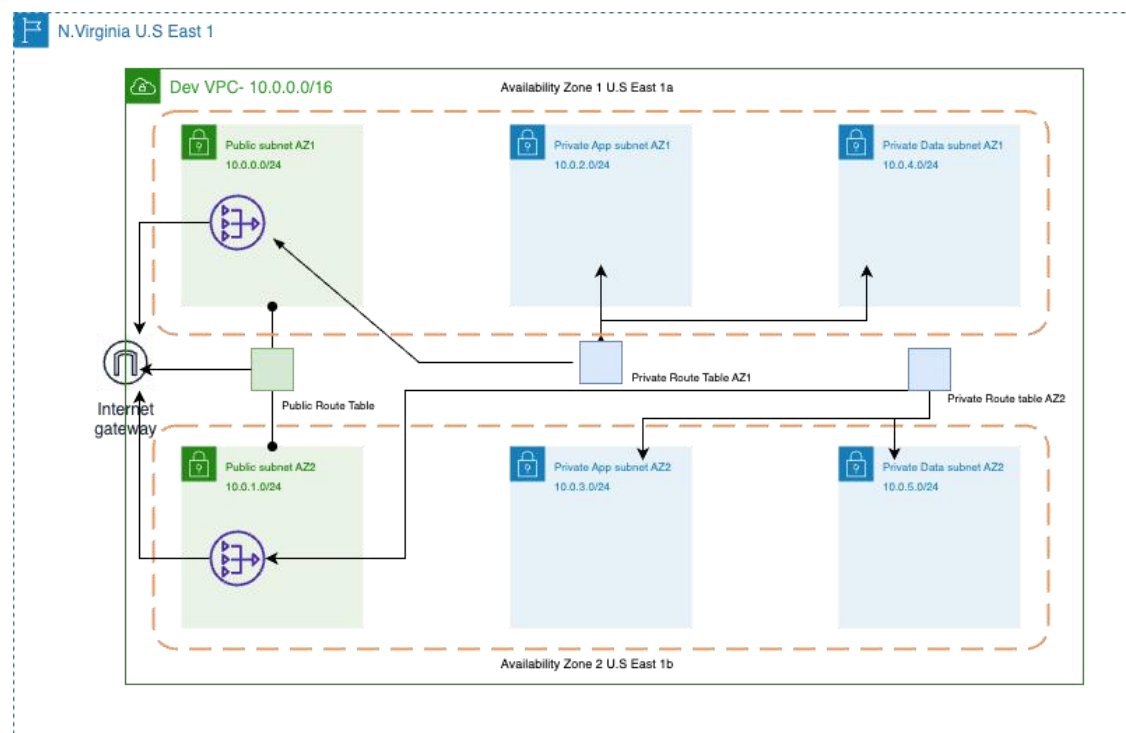


# How To Build a Three Tier AWS Network VPC

The objective of building a three-tier AWS network VPC is to create a scalable environment that can meet the specific requirements of the application or system being developed, while ensuring the highest levels of security, availability, and performance. The three tiers typically consist of the following:

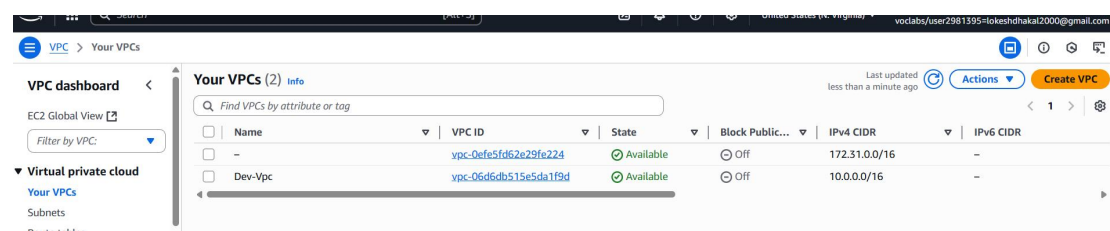
1. Presentation tier or web tier: This tier handles user interface functions and serves as the entry point for the application. It can include web servers, load balancers, and content delivery networks (CDNs).
2. Application tier or middle tier: This tier handles the application logic and can include application servers and databases.
3. Data tier or back-end tier: This tier manages data storage and processing and can include databases and data warehouses.



From the above architecture, the infrastructure is divided into three tiers. The first tier includes the public subnet which holds resources such as the Nat Gateway and Bastion Host. The second tier includes the private subnet which holds our web servers i.e our EC2 instances. Lastly, the third tier, which includes another private subnet which holds our databases.

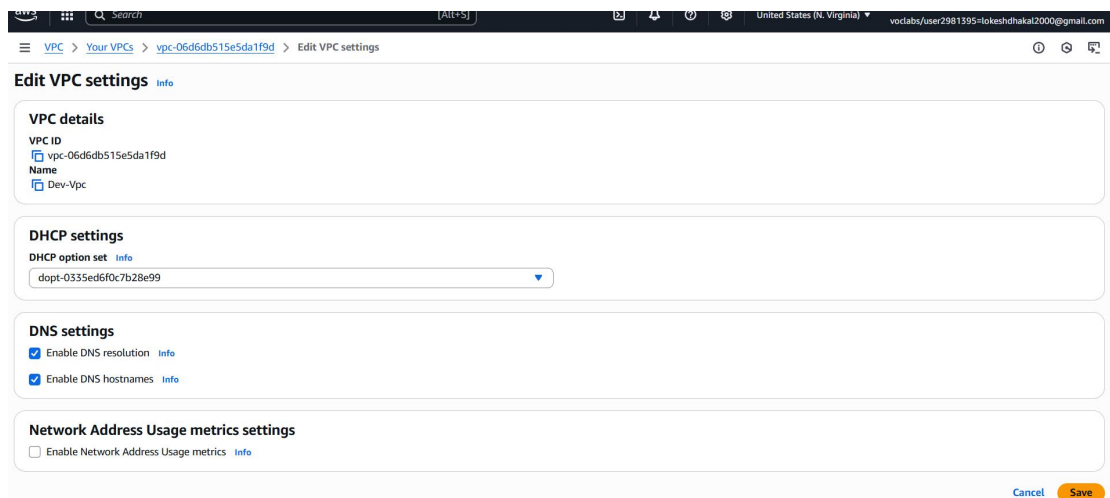
For the purpose of high availability and fault tolerance, we will duplicate the subnets across another availability zone. We will also create an internet gateway to allow the resources in the VPC to have access to the internet. To create the VPC, we'll move to the management console.

1. Select the region where the VPC will be created. Here we'll be using the N. Virginia (U.S East 1) region.
2. On the search box, type "VPC" and select VPC under the "services" drop down.
3. In your VPC dashboard, select VPC then click on "create VPC".
4. Give your VPC a name. Here i'll be using "Dev VPC" as my VPC name.
5. Under the IPv4 CIDR Block, leave the default setting "IPv4 CIDR manual input", then enter the CIDR block (10.0.0.0/16) referencing the architecture above.
6. Leave the other settings as they are (default), then scroll down and click on "create VPC".



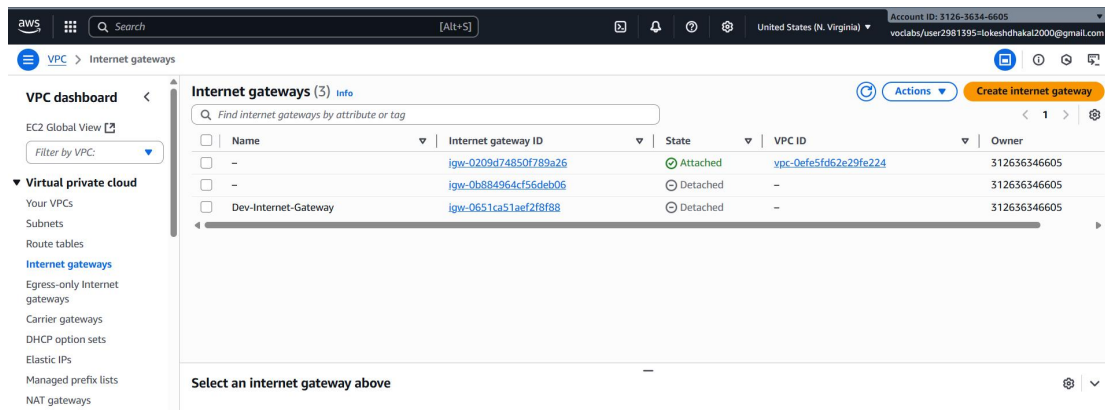
The next step is to enable DNS Hostname in the VPC we just created. To do that;

1. In the VPC we just created, select “Actions” on the top right corner
2. Select “Edit VPC settings”
3. Check the box which enables DNS Hostnames, then click on “Save Changes”



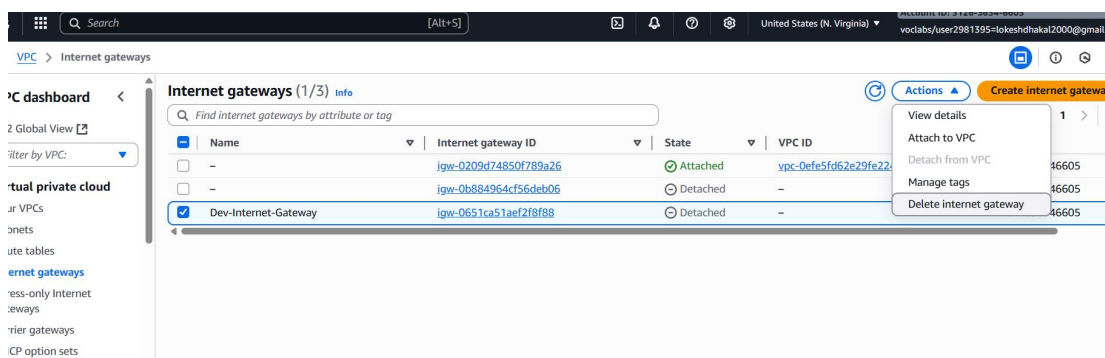
The next step is to create an Internet Gateway for our VPC and to do this;

1. On the left side where the you have the VPC dashboard drop down menu, select “Internet Gateway”
2. Then click on “Create Internet Gateway”
3. Give your Internet Gateway a name. In this case, the name is “Dev Internet Gateway”
4. Click on “create Internet Gateway”



The next step is to attach the Internet Gateway to the Dev VPC to allow the VPC to communicate with the internet. To do this, we could either click on the pop up on the Internet Gateway interface which appears after creating the Internet Gateway or we could click on “Actions”, then select “attach VPC” from the drop down menu.

Under “Available VPCs”, click on the search box and select the VPC you wish to attach the Internet Gateway to. In this case, select “Dev VPC”, then click on “Attach Internet Gateway”.



The Next step is to create our public subnets in the first and second availability zones according to our reference architecture. To do this;

1. Select “Subnets” on the left side where you have the VPC dashboard drop down menu.
2. On the top right corner, click on “Create Subnet”
3. Select the VPC where you want the subnet to create the subnet in and in this case, it’d be the Dev VPC
4. Give the subnet a name; Public Subnet AZ1
5. Under availability zones, select U.S East 1a referencing our architecture above
6. Under IPv4 CIDR Block, enter and select 10.0.0.0/24 according to our architecture.
7. Click on “Create Subnet”

**Subnet 1 of 1**

**Subnet name**

Create a tag with a key of 'Name' and a value that you specify.

public-subnet-AZ1

The name can be up to 256 characters long.

**Availability Zone** [Info](#)

Choose the zone in which your subnet will reside, or let Amazon choose one for you.

United States (N. Virginia) / use1-az6 (us-east-1a)

**IPv4 VPC CIDR block** [Info](#)

Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

10.0.0.0/16

**IPv4 subnet CIDR block**

10.0.0.0/24

256 IPs

< > ^ v

Knowing that we need two subnets but in different availability zones, we'll repeat the above steps but with some changes and they are;

Subnet name- Public Subnet AZ2

Availability Zone- U.S East 1b

CIDR Block- 10.0.1.0/24

**Subnet 1 of 1**

**Subnet name**  
Create a tag with a key of 'Name' and a value that you specify.  
  
The name can be up to 256 characters long.

**Availability Zone** [Info](#)  
Choose the zone in which your subnet will reside, or let Amazon choose one for you.

**IPv4 VPC CIDR block** [Info](#)  
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

**IPv4 subnet CIDR block**  
 256 IPs

To see both subnets, on the left side, click on “filter by VPC” then select “Dev VPC”. It'll show the subnets in Dev VPC i.e Public Subnet AZ1 and Public Subnet AZ2.

aws

Search [Alt+S]

United States (N. Virginia) voclabs/user2981395--itokeshdthakal2000@gmail.com

VPC Subnets

VPC dashboard

EC2 Global View

Filter by VPC: ▼

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only Internet gateways

Carrier gateways

DHCP option sets

You have successfully created 1 subnet: subnet-03b86d5dfafa271b4

Subnets (8) [Info](#)

Find subnets by attribute or tag

<input type="checkbox"/>	Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
<input type="checkbox"/>	-	subnet-0482c98a5f3c138fa	Available	vpc-0efe5fd62e29fe224	Off	172.31.48.0/2
<input type="checkbox"/>	-	subnet-0912cd1b7f9acfb2	Available	vpc-0efe5fd62e29fe224	Off	172.31.16.0/2
<input type="checkbox"/>	-	subnet-06c66b7963c51f525	Available	vpc-0efe5fd62e29fe224	Off	172.31.64.0/2
<input type="checkbox"/>	-	subnet-0af07fb15547065c2	Available	vpc-0efe5fd62e29fe224	Off	172.31.32.0/2
<input type="checkbox"/>	-	subnet-011874864fdc7aaa9	Available	vpc-0efe5fd62e29fe224	Off	172.31.0.0/20
<input type="checkbox"/>	public-subnet-AZ1	subnet-018b76c5f06a4bc64	Available	vpc-06d6db515e5da1f9d   Dev:...	Off	10.0.0.0/24
<input type="checkbox"/>	public-subnet-AZ2	subnet-03b86d5dfafa271b4	Available	vpc-06d6db515e5da1f9d   Dev:...	Off	10.0.1.0/24

The next step is to enable “Auto Assign” for the two public subnets. This is to ensure that EC2 instances launched in these public subnets will be assigned a public IPv4 address. To enable this feature;

1. Select the first subnet (Public Subnet AZ1)
2. Click on “Actions” and click on “Edit Subnet settings”
3. Enable auto assign public IPv4 address by checking the box
4. Scroll down and click on “save changes”

⌵ VPC > Subnets > subnet-018b76c5f06a4bc64 > Edit subnet settings ⓘ ⌵ ⌵ ⌵

### Edit subnet settings ⓘ

**Subnet**

Subnet ID  
subnet-018b76c5f06a4bc64

Name  
public-subnet-AZ1

**Auto-assign IP settings ⓘ**  
Enable AWS to automatically assign a public IPv4 or IPv6 address to a new primary network interface for an instance in this subnet.

☒ Enable auto-assign public IPv4 address ⓘ  
☐ Enable auto-assign customer-owned IPv4 address ⓘ  
Option disabled because no customer owned pools found.

**Resource-based name (RBN) settings ⓘ**  
Specify the hostname type for EC2 instances in this subnet and optional RBN DNS query settings.

☐ Enable resource name DNS A record on launch ⓘ  
☐ Enable resource name DNS AAAA record on launch ⓘ

**Hostname type ⓘ**  
☐ Resource name  
☒ IP name

Repeat this steps for Public Subnet AZ2.

⌵ VPC > Subnets > subnet-03b86d5dfafa271b4 > Edit subnet settings ⓘ ⌵ ⌵ ⌵

### Edit subnet settings ⓘ

**Subnet**

Subnet ID  
subnet-03b86d5dfafa271b4

Name  
public-subnet-AZ2

**Auto-assign IP settings ⓘ**  
Enable AWS to automatically assign a public IPv4 or IPv6 address to a new primary network interface for an instance in this subnet.

☒ Enable auto-assign public IPv4 address ⓘ  
☐ Enable auto-assign customer-owned IPv4 address ⓘ  
Option disabled because no customer owned pools found.

**Resource-based name (RBN) settings ⓘ**  
Specify the hostname type for EC2 instances in this subnet and optional RBN DNS query settings.

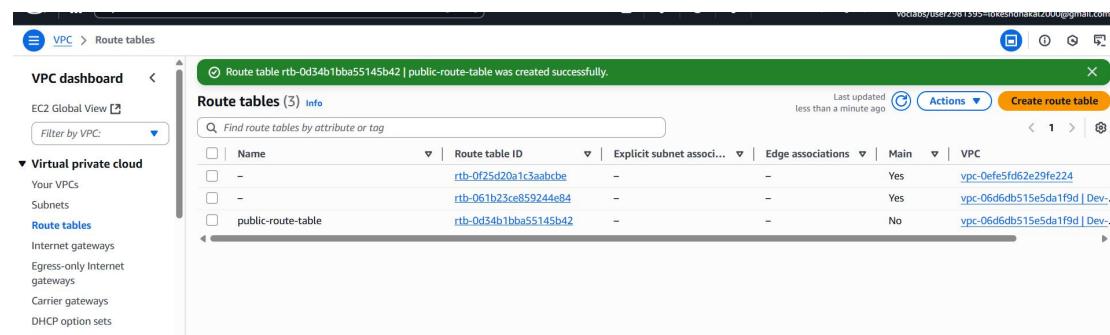
☐ Enable resource name DNS A record on launch ⓘ  
☐ Enable resource name DNS AAAA record on launch ⓘ

**Hostname type ⓘ**  
☐ Resource name  
☒ IP name

**DNS64 settings**  
Enable DNS64 to allow IPv6-only services in Amazon VPC to communicate with IPv4-only services and networks.

The next step is to create a Route Table and this would be called “Public Route Table” according to our reference architecture. To create a Route Table;

1. Select Route Tables from the left side. Here, you’d see an already existing route table and this was automatically created when the Dev VPC was created. This route table is called the Main Route Table and it is private by default.
2. Click on “create route table”
3. Give the route table a name. In this case, it would be “Public Route Table”
4. Select the VPC you want to create the route table in i.e the Dev VPC.
5. Click “create Route Table”



After creating the route table, the next step is to add a public route to the route table. This would allow the route table to route traffic to the internet. To add a route to the public route table;

1. On the Public Route Table interface, select the “route” tab, then click on “edit routes”



2. Click on “add route”
3. Under destination, type and select 0.0.0.0/0
4. Under target, select the Internet Gateway i.e Dev Internet Gateway
5. Click on “save changes”

The screenshot shows the 'Edit routes' interface in the AWS Management Console. The breadcrumb navigation at the top indicates the path: VPC > Route tables > rtb-0d34b1bba55145b42 > Edit routes. The main section is titled 'Edit routes' and contains a table with columns: Destination, Target, Status, Propagated, and Route Origin. There are two rows of routes. The first row has a Destination of 10.0.0.0/16, a Target of 'local', a Status of 'Active' (with a green checkmark), and a Route Origin of 'CreateRouteTable'. The second row has a Destination of 0.0.0.0/0, a Target of 'Internet Gateway', a Status of '-', and a Route Origin of 'CreateRoute'. Below the table, there is an 'Add route' button. At the bottom right, there are three buttons: 'Cancel', 'Preview', and 'Save changes'.

Destination	Target	Status	Propagated	Route Origin
10.0.0.0/16	local	Active	No	CreateRouteTable
0.0.0.0/0	Internet Gateway	-	No	CreateRoute

Buttons: Add route, Cancel, Preview, Save changes

The next step is to associate the two public subnets with this route table. To do this;

1. On the Route Table interface, click on “subnet associations” then click on “edit subnet associations”
2. On this page, you’ll see the two subnets we created (Public Subnet AZ1 and Public Subnet AZ2)
3. Select both subnets by checking the boxes then click on “save associations”

The screenshot shows the 'Edit subnet associations' interface in the AWS Management Console. The breadcrumb navigation at the top indicates the path: VPC > Route tables > rtb-0d34b1bba55145b42 > Edit subnet associations. The main section is titled 'Edit subnet associations' with a subtitle 'Change which subnets are associated with this route table.' Below this, there is a table titled 'Available subnets (2/2)' with columns: Name, Subnet ID, IPv4 CIDR, IPv6 CIDR, and Route table ID. There are two rows of subnets, both of which are selected (indicated by checked checkboxes). The first row is 'public-subnet-AZ1' with Subnet ID 'subnet-018b76c5f06a4bc64' and IPv4 CIDR '10.0.0.0/24'. The second row is 'public-subnet-AZ2' with Subnet ID 'subnet-03b86d5dfafa271b4' and IPv4 CIDR '10.0.1.0/24'. Both rows have a Route table ID of 'Main (rtb-061b23ce859244e84)'. Below the table, there is a section titled 'Selected subnets' which shows two tags: 'subnet-018b76c5f06a4bc64 / public-subnet-AZ1' and 'subnet-03b86d5dfafa271b4 / public-subnet-AZ2'. At the bottom right, there are two buttons: 'Cancel' and 'Save associations'.

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
public-subnet-AZ1	subnet-018b76c5f06a4bc64	10.0.0.0/24	-	Main (rtb-061b23ce859244e84)
public-subnet-AZ2	subnet-03b86d5dfafa271b4	10.0.1.0/24	-	Main (rtb-061b23ce859244e84)

Buttons: Cancel, Save associations

The next step is to create our four private subnets. To do this;

1. Select “subnets” on the left side
2. Click on “create subnets”
3. Select the VPC we want the subnet to be in, in the case; Dev VPC
4. Give the subnet a name. According to our reference architecture, it’d be Private App Subnet AZ1
5. Select the availability zone we want the subnet to be in and that is U.S East 1a according to our reference architecture.
6. Under IPv4 CIDR Block, type and select 10.0.2.0/24 according to our reference architecture.
7. Scroll down and Click on “create subnet”

Subnet 1 of 1

**Subnet name**  
Create a tag with a key of 'Name' and a value that you specify.  
Private-Subnet-AZ1  
The name can be up to 256 characters long.

**Availability Zone** [Info](#)  
Choose the zone in which your subnet will reside, or let Amazon choose one for you.  
United States (N. Virginia) / us-east-1a

**IPv4 VPC CIDR block** [Info](#)  
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.  
10.0.0.0/16

**IPv4 subnet CIDR block**  
10.0.2.0/24 256 IPs

▼ **Tags - optional**

Key	Value - optional
Name	Private-Subnet-AZ1

[Add new tag](#) [Remove](#)

Now we would repeat the steps for other private subnets bearing in mind that their names, availability zone and CIDR Block would be according to the reference architecture.

For the second private subnet, we'll have;

Name: Private App Subnet AZ2

CIDR Block: 10.0.3.0/24

Availability zone: U.S East 1b

### Subnet settings

Specify the CIDR blocks and Availability Zone for the subnet.

#### Subnet 1 of 1

##### Subnet name

Create a tag with a key of 'Name' and a value that you specify.

Private App Subnet AZ2

The name can be up to 256 characters long.

##### Availability Zone [Info](#)

Choose the zone in which your subnet will reside, or let Amazon choose one for you.

United States (N. Virginia) / use1-az1 (us-east-1b)

##### IPv4 VPC CIDR block [Info](#)

Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

10.0.0.0/16

##### IPv4 subnet CIDR block

10.0.3.0/24

256 IPs

For the third private subnet, we'll have;

Name: Private Data Subnet AZ1

CIDR Block: 10.0.4.0/24

Availability zone: U.S East 1a

### Subnet settings

Specify the CIDR blocks and Availability Zone for the subnet.

#### Subnet 1 of 1

##### Subnet name

Create a tag with a key of 'Name' and a value that you specify.

Private Data Subnet Az1

The name can be up to 256 characters long.

##### Availability Zone [Info](#)

Choose the zone in which your subnet will reside, or let Amazon choose one for you.

United States (N. Virginia) / use1-az6 (us-east-1a)

##### IPv4 VPC CIDR block [Info](#)

Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

10.0.0/16

##### IPv4 subnet CIDR block

10.0.4.0/24

256 IPs

For the fourth private subnet, we'll have

Name: Private Data Subnet AZ2

CIDR Block: 10.0.5.0/24

Availability zone: U.S East 1b

### Subnet settings

Specify the CIDR blocks and Availability Zone for the subnet.

#### Subnet 1 of 1

##### Subnet name

Create a tag with a key of 'Name' and a value that you specify.

Private Data Subnet AZ2

The name can be up to 256 characters long.

##### Availability Zone [Info](#)

Choose the zone in which your subnet will reside, or let Amazon choose one for you.

United States (N. Virginia) / use1-az1 (us-east-1b)

##### IPv4 VPC CIDR block [Info](#)

Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

10.0.0/16

##### IPv4 subnet CIDR block

10.0.5.0/24

256 IPs

One important thing to know is the difference between a public subnet and a private subnet. When you create a route table and add a route to the route table to route traffic to the internet, any subnet you associate with that route table becomes public. So basically, a public subnet is associated with a route table that is able to route traffic to the internet. On the other hand, when you have a route table that does not have a route to the internet, any subnet associated to that route table is private.

The next step is to create a NAT Gateway which allows the instances in the private subnets to access the internet. According to our reference architecture, we'll create a NAT Gateway in the public subnet AZ1, then we'll create a private route table i.e Private Route Table AZ1. Afterwards, we'll add a route to the route table to route traffic to the internet through the NAT Gateway, then we'll associate the private subnets in AZ1 to the private route table. We'll repeat the process in the second availability zone so as to ensure that the private subnets in AZ2 can access the internet.

To create a NAT Gateway, make sure you're in the region where you created your VPC, in this case N.Virginia for Dev VPC.

1. In the search box of the management console, type "VPC", then select VPC under services
2. In the VPC dashboard, on the left side, select NAT Gateways
3. Click on "create NAT Gateway"
4. According to our reference architecture, we'll create our first NAT Gateway in Public Subnet AZ1
5. Give the NAT Gateway a name i.e NAT Gateway AZ1
6. Select the subnet we want to create the NAT Gateway in i.e Public Subnet AZ1
7. Under Elastic IP Allocation ID, click on "allocate elastic IP" which will allocate an Elastic IP for you.
8. Scroll down and click on "create NAT Gateway"

**Create NAT gateway** [Info](#)  
A highly available, managed Network Address Translation (NAT) service that instances in private subnets can use to connect to services in other VPCs, on-premises networks, or the internet.

**NAT gateway settings**  
**Name - optional**  
Create a tag with a key of 'Name' and a value that you specify.  
  
The name can be up to 256 characters long.

**Subnet**  
Select a subnet in which to create the NAT gateway.

**Connectivity type**  
Select a connectivity type for the NAT gateway.  
☒ Public  
☐ Private

**Elastic IP allocation ID** [Info](#)  
Assign an Elastic IP address to the NAT gateway.  
 [Allocate Elastic IP](#)

The next step is to create a route table and we'll call it "Private Route Table AZ1". To do this;

1. On the left side, click on "Route Tables"
2. Give the route table a name i.e "Private Route Table AZ1"
3. Select the VPC you want the route table to be created in. In this case; Dev VPC
4. Then Click on "create Route Table"

The screenshot shows the 'Create route table' page in the AWS Management Console. At the top, there's a title 'Create route table' with an 'info' link. Below it, a descriptive sentence states: 'A route table specifies how packets are forwarded between the subnets within your VPC, the internet, and your VPN connection.' The main form is divided into two sections. The first section, 'Route table settings', contains a 'Name - optional' field with the placeholder text 'Create a tag with a key of 'Name' and a value that you specify.' and a text input field containing 'Private Route Table AZ1'. Below this is a 'VPC' dropdown menu with the text 'The VPC to use for this route table.' and a selected value 'vpc-06d6db515e5da1f9d (Dev-Vpc)'. The second section, 'Tags', includes a description: 'A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.' It features a table with two columns: 'Key' and 'Value - optional'. The 'Key' column has a search input with 'Name' and a blue 'X' icon. The 'Value' column has a search input with 'Private Route Table AZ1', a blue 'X' icon, and a 'Remove' button. There is an 'Add new tag' button and a note 'You can add 49 more tags.' At the bottom right of the form, there are two buttons: 'Cancel' and 'Create route table'.

Next, we'll add a route to the Private Route Table AZ1. To do this;

1. On the Private Route Table AZ1 interface, under routes, click on "edit routes"
2. Click on "add route"
3. Under destination, type and select 0.0.0.0/0 and under target, it'd be NAT Gateway AZ1
4. Then click on "save changes"

☰ VPC > Route tables > rtb-04cb00cb22620688d > Edit routes

### Edit routes

Destination	Target	Status	Propagated	Route Origin
10.0.0.0/16	local	Active	No	CreateRouteTable
0.0.0.0/0	NAT Gateway	-	No	CreateRoute

Buttons: Add route, Cancel, Preview, Save changes

The next step is to associate this route table to the private app subnet AZ1 and private data subnet AZ1. To do this;

1. Still on the Private Route Table AZ1 interface, Click on “subnet associations”, then click on “edit Subnet associations”
2. Under the available subnets, select only the Private App Subnet AZ1 and Private Data Subnet AZ1
3. Then click on “save Associations”

### Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (2/6)

Filter subnet associations

<input type="checkbox"/>	Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
<input checked="" type="checkbox"/>	Private App Subnet AZ1	subnet-038b38c68fa0a84f2	10.0.2.0/24	-	Main (rtb-061b23ce859244e84)
<input type="checkbox"/>	public-subnet-AZ1	subnet-018b76c5f06a4bc64	10.0.0.0/24	-	rtb-0d34b1bba55145b42 / public-rout...
<input type="checkbox"/>	public-subnet-AZ2	subnet-03b86d5dfafa271b4	10.0.1.0/24	-	rtb-0d34b1bba55145b42 / public-rout...
<input type="checkbox"/>	Private App subnet AZ2	subnet-025743cc40388367d	10.0.3.0/24	-	Main (rtb-061b23ce859244e84)
<input checked="" type="checkbox"/>	Private Data Subnet Az1	subnet-08a02a7e7ee3b726d	10.0.4.0/24	-	Main (rtb-061b23ce859244e84)
<input type="checkbox"/>	Private Data Subnet AZ2	subnet-01f2940f91c4dd761	10.0.5.0/24	-	Main (rtb-061b23ce859244e84)

The next step is to create another NAT Gateway in the Public Subnet AZ2, then repeat the process but for private subnets that are in the second availability zone referencing our architecture. To do this;

1. Click on NAT Gateways on the left side
2. Click on “create NAT Gateway”
3. Give the NAT Gateway a name. In this case, it’ll be NAT Gateway AZ2

4. Select the Subnet for this NAT Gateway which is Public Subnet AZ2
5. Under Elastic IP Allocation ID, click “Allocate Elastic IP”
6. Scroll down and click on “create NAT Gateway”

VPC > NAT gateways > Create NAT gateway

Elastic IP address 3.221.126.53 (eipalloc-0ae71fd0095a1220a) allocated.

### NAT gateway settings

**Name - optional**  
Create a tag with a key of 'Name' and a value that you specify.

Nat Gateway AZ2

The name can be up to 256 characters long.

**Subnet**  
Select a subnet in which to create the NAT gateway.

subnet-03b86d5dfafa271b4 (public-subnet-AZ2)

**Connectivity type**  
Select a connectivity type for the NAT gateway.

☒ Public  
☐ Private

**Elastic IP allocation ID** [Info](#)  
Assign an Elastic IP address to the NAT gateway.

eipalloc-0ae71fd0095a1220a

[Allocate Elastic IP](#)

The next step is to create another route table (Private Route Table AZ2) following the previous step for creating the first private route table.

VPC > Route tables > Create route table

### Create route table

A route table specifies how packets are forwarded between the subnets within your VPC, the internet, and your VPN connection.

#### Route table settings

**Name - optional**  
Create a tag with a key of 'Name' and a value that you specify.

Private Route Table AZ2

**VPC**  
The VPC to use for this route table.

vpc-06d6db515e5da1f9d (Dev-Vpc)

#### Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value - optional
Name	Private Route Table AZ2

[Add new tag](#)

You can add 49 more tags.

[Cancel](#) [Create route table](#)

The next step is to add a route to the Private Route Table AZ2. We'll also repeat the steps for this process but in this case, we'll select the NAT Gateway AZ2.



VPC > Route tables > rtb-0eb8a286d6e45b822 > Edit routes

### Edit routes

Destination	Target	Status	Propagated	Route Origin
10.0.0.0/16	local	Active	No	CreateRouteTable
0.0.0.0/0	NAT Gateway	-	No	CreateRoute

The next and final step is to associate this route table to the Private App Subnet AZ2 and Private Data Subnet AZ2. We'll also repeat the steps for this process but in this case, we'll only select the aforementioned Subnets.

VPC > Route tables > rtb-0eb8a286d6e45b822 > Edit subnet associations

### Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (2/6)

Filter subnet associations

	Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
<input type="checkbox"/>	Private App Subnet AZ1	subnet-038b38c68fa0a84f2	10.0.2.0/24	-	rtb-04cb00cb22620688d / Private Rou...
<input type="checkbox"/>	public-subnet-AZ1	subnet-018b76c5f06a4bc64	10.0.0.0/24	-	rtb-0d34b1bba55145b42 / public-rout...
<input type="checkbox"/>	public-subnet-AZ2	subnet-03b86d5dfafa271b4	10.0.1.0/24	-	rtb-0d34b1bba55145b42 / public-rout...
<input checked="" type="checkbox"/>	Private App subnet AZ2	subnet-025743cc40388367d	10.0.3.0/24	-	Main (rtb-061b23ce859244e84)
<input type="checkbox"/>	Private Data Subnet Az1	subnet-08a02a7e7ee3b726d	10.0.4.0/24	-	rtb-04cb00cb22620688d / Private Rou...
<input checked="" type="checkbox"/>	Private Data Subnet AZ2	subnet-01f2940f91c4dd761	10.0.5.0/24	-	Main (rtb-061b23ce859244e84)

Selected subnets

Finally, there it is, our Three Tier AWS Network VPC.