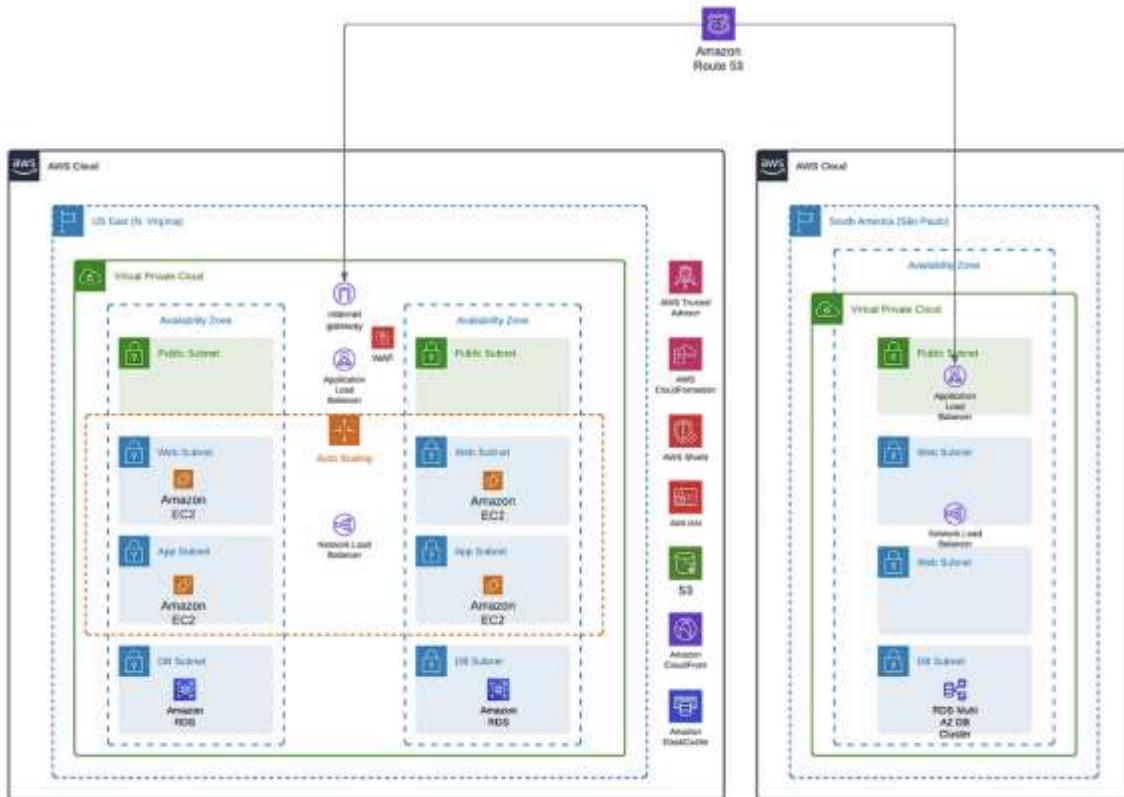


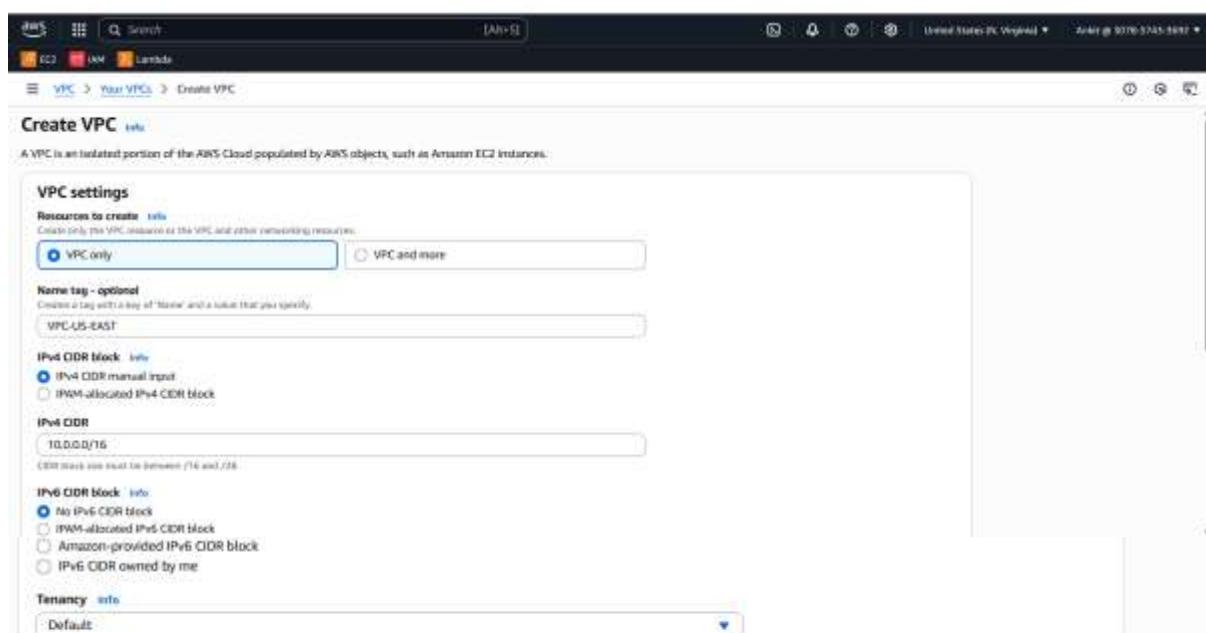
Building Scalable and Secure Web Applications with Three-Tier Architecture on AWS



Phase 1: Initial Setup and Networking

1. Create VPCs in Two Regions: [N. Virginiaus-east-1](#) & [Stockholmeu-north-1](#)

- Create Vpc in this Regoin : [N. Virginiaus-east-1](#)
- Go to the VPC dashboard in AWS Console.
- Enter a name : VPC-US-East
- Set CIDR block: US East: 10.0.0.0/16
- Click on "Create 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:

Name

Value - optional:

VPC-US-EAST

[Remove tag](#)

- Enable DNS resolution and hostname support.

The screenshot shows the 'Edit VPC settings' page for a VPC with ID 'vpc-06115a2ba1bd1eadbf'. The 'DNS settings' section is highlighted with a red box. It contains two checked checkboxes: 'Enable DNS resolution' and 'Enable DNS hostnames'. Other sections like 'DHCP settings' and 'Network Address Usage metrics settings' are also visible.

- Create Vpc in this Regoin : **Stockholmeu-north-1**
- Go to the VPC dashboard in AWS Console.
- Enter a name : VPC-EU-North
- Set CIDR block: US East: 10.1.0.0/16
- Click on "Create VPC"

The screenshot shows the 'Create VPC' page. The 'VPC settings' section is filled with the following details:

- Region to create: EU (Ireland)
- Name tag - optional: VPC-EU-NORTH
- IPv4 CIDR Block: 10.1.0.0/16
- IPv6 CIDR Block: None (selected)
- Tenancy: Default
- Tags: A tag named 'Name' with value 'VPC-EU-NORTH' is added.

 The 'Create VPC' button at the bottom right is highlighted with a red box.

- Enable DNS resolution and hostname support.

The screenshot shows the 'Edit VPC settings' page for a VPC with ID vpc-048d10c7a26c95d11. The page is divided into several sections:

- VPC details:** Shows the VPC ID (vpc-048d10c7a26c95d11) and Name (VPC-ELU-NORTH).
- DHCP settings:** Displays the DHCP option set (dopt-09e0c22862a709271).
- DNS settings:** Includes checkboxes for 'Enable DNS resolution' (checked) and 'Enable DNS hostnames' (checked).
- Network Address Usage metrics settings:** Includes a checkbox for 'Enable Network Address Usage metrics' (unchecked).

At the bottom right are 'Cancel' and 'Save' buttons.

2. Create Subnets in Each Region:

N. Virginiaus-east-1

Create at least two subnets per Availability Zone:

- Public Subnet: For Load Balancers, NAT Gateway.
- Web Subnet: For Web EC2 Instances.
- App Subnet: For App EC2 Instances.
- DB Subnet: For RDS instances.

Distribute subnets across two Availability Zones for HA.

Allocate CIDR ranges (e.g., /24) within the VPC CIDR.

Subnet CIDR Allocation per AZ

| Subnet Type | AZ-A (us-east-1a) | AZ-B (us-east-1b) |
|---------------|-------------------|-------------------|
| Public Subnet | 10.0.1.0/24 | 10.0.5.0/24 |
| Web Subnet | 10.0.2.0/24 | 10.0.6.0/24 |
| App Subnet | 10.0.3.0/24 | 10.0.7.0/24 |
| DB Subnet | 10.0.4.0/24 | 10.0.8.0/24 |

Find subnets by attribute or tag

| <input type="checkbox"/> | Name | ▼ Subnet ID | ▼ State | VPC | ▼ Block Public... ▼ | IPv4 CIDR | ▼ |
|--------------------------|-------------------|---|------------------------|--|---|-------------|---|
| <input type="checkbox"/> | web-subnet-az2 | subnet-0f249c35b356a83b3 | Available | vpc-06115a28a1b5eadbf VPC... | <input checked="" type="checkbox"/> Off | 10.0.6.0/24 | |
| <input type="checkbox"/> | web-subnet-az1 | subnet-03a88ddbb2562da285 | Available | vpc-06115a28a1b5eadbf VPC... | <input checked="" type="checkbox"/> Off | 10.0.2.0/24 | |
| <input type="checkbox"/> | public-subnet-az2 | subnet-0fc476455565cea9d | Available | vpc-06115a28a1b5eadbf VPC... | <input checked="" type="checkbox"/> Off | 10.0.5.0/24 | |
| <input type="checkbox"/> | public-subnet-az1 | subnet-01b8abe6554d79a63 | Available | vpc-06115a28a1b5eadbf VPC... | <input checked="" type="checkbox"/> Off | 10.0.1.0/24 | |
| <input type="checkbox"/> | db-subnet-az2 | subnet-020cb1e12ee543cdc | Available | vpc-06115a28a1b5eadbf VPC... | <input checked="" type="checkbox"/> Off | 10.0.8.0/24 | |
| <input type="checkbox"/> | db-subnet-az1 | subnet-0773cbf9c799fa49d | Available | vpc-06115a28a1b5eadbf VPC... | <input checked="" type="checkbox"/> Off | 10.0.4.0/24 | |
| <input type="checkbox"/> | app-subnet-az2 | subnet-0a08704598c353430 | Available | vpc-06115a28a1b5eadbf VPC... | <input checked="" type="checkbox"/> Off | 10.0.7.0/24 | |
| <input type="checkbox"/> | app-subnet-az1 | subnet-09b1a64e600da2e40 | Available | vpc-06115a28a1b5eadbf VPC... | <input checked="" type="checkbox"/> Off | 10.0.3.0/24 | |

Stockholmeu-north-1

Create at least two subnets per Availability Zone:

Public Subnet : for Application Load Balancer (ALB).

Web Subnet 1 : for Network Load Balancer (NLB).

Web Subnet 2 : Another web tier subnet (for redundancy or separation of layers).

DB Subnet : for RDS Multi-AZ DB Cluster.

| Subnet Type | Example CIDR Block |
|---------------|--------------------|
| Public Subnet | 10.1.1.0/24 |
| Web Subnet 1 | 10.1.2.0/24 |
| Web Subnet 2 | 10.1.3.0/24 |
| DB Subnet | 10.1.4.0/24 |

| <input type="checkbox"/> | Name | ▲ Subnet ID | ▼ State | VPC | ▼ Block Public... ▼ | IPv4 CIDR | ▼ |
|--------------------------|------------------|--|------------------------|--|---|-------------|---|
| <input type="checkbox"/> | DB-Subnet | subnet-008e9c7115b0f8e15 | Available | vpc-04b810c7e26e05d51 VPC... | <input checked="" type="checkbox"/> Off | 10.1.4.0/24 | |
| <input type="checkbox"/> | Public-Subnet-AZ | subnet-028a45c412a9a508d | Available | vpc-04b810c7e26e05d51 VPC... | <input checked="" type="checkbox"/> Off | 10.1.1.0/24 | |
| <input type="checkbox"/> | Web-Subnet-AZ1 | subnet-08ce80df98336131 | Available | vpc-04b810c7e26e05d51 VPC... | <input checked="" type="checkbox"/> Off | 10.1.2.0/24 | |
| <input type="checkbox"/> | Web-Subnet-AZ2 | subnet-0d51c457e5bd8030f | Available | vpc-04b810c7e26e05d51 VPC... | <input checked="" type="checkbox"/> Off | 10.1.3.0/24 | |

Select a subnet

3. Configure Route Tables and Internet Gateway:

N. Virginia-us-east-1

First Create Route Table :

- For Public Route Table Create : Public-RT
- And then Create Neternet Getway : Public-igw

Create internet gateway Info

An internet gateway is a virtual router that connects a VPC to the internet. To create a new internet gateway specify the name for the gateway below.

Internet gateway settings

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

My-Public-igw

Tags - optional
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**

Add new tag
You can add 49 more tags.

- Attach to Current Regoin VPC

The following internet gateway was created: igw-0cc62fc917a69a79d - My-Public-igw. You can now attach to a VPC to enable the VPC to communicate with the internet.

igw-0cc62fc917a69a79d / My-Public-igw

Details Info

| | | | |
|--|------------------------------------|--------------|------------------------|
| Internet gateway ID igw-0cc62fc917a69a79d | State: Detached | VPC ID: - | Owner: 307857433692 |
|--|------------------------------------|--------------|------------------------|

Tags

| Key | Value |
|-------|---------------|
| Names | My-Public-igw |

- Route the Internet Getway in Public Route Table

rtb-021b769d61112928b / Public-RT

Details Info

| | | | |
|--|--|---|-------------------|
| Route Table ID: rtb-021b769d61112928b | Main <input type="radio"/> <input checked="" type="radio"/> | Explicit subnet associations 4 subnets | Edge associations |
|--|--|---|-------------------|

Routers

| Destination | Target | Status | Propagated |
|-------------|---------------------------------------|---|------------|
| 0.0.0.0/0 | igw-0cc62fc917a69a79d | <input checked="" type="radio"/> Active | No |
| 10.0.0.0/16 | local | <input checked="" type="radio"/> Active | No |

- And Subnet Associations :

1. public-subnet-az1
2. public-subnet-az2
3. web-subnet-az1
4. web-subnet-az2

The screenshot shows the 'Subnet associations' tab for route table 'rtb-021b769d61112928b / Public-RT'. It lists four explicit subnet associations:

| Name | Subnet ID | IPv4 CIDR | IPv6 CIDR |
|-------------------|-------------------------|-------------|-----------|
| public-subnet-az1 | subnet-01bbab6554479a63 | 10.0.1.0/24 | - |
| web-subnet-az1 | subnet-01a88cc2562da285 | 10.0.2.0/24 | - |
| web-subnet-az2 | subnet-01249c55b356a895 | 10.0.6.0/24 | - |
| public-subnet-az2 | subnet-01a476455565ca9d | 10.0.5.0/24 | - |

- For Private Route Table : Private-RT
- And then Create NAT Getway for Private internet to private subnet : My-NAT-gateway
- Select Subnet in Nat Creation is Public Subnet and Allocate Elastic IP

The screenshot shows the 'Create NAT gateway' wizard step. It includes fields for:

- NAT gateway settings**: Name is 'My-NAT-gateway'.
- Subnet**: Selected subnet is 'subnet-01bbab6554479a63 (public-subnet-az1)'.
- Connectivity type**: Set to 'Public'.
- Basic IP allocation ID**: 'ipalloc-0aabbcccddeeff' is selected, with a 'Allocate Elastic IP' button.
- Tags**: A tag 'Name' is assigned with value 'My-NAT-gateway'.

At the bottom are 'Cancel' and 'Create NAT gateway' buttons.

- And then Route the NAT getway in Private Route Table

The screenshot shows the 'Routes' tab for route table 'rtb-0e547ceab464d6e39 / Private-RT'. It lists two routes:

| Destination | Target | Status | Propagated |
|-------------|-----------------------|--------|------------|
| 0.0.0.0/0 | nat-07f55644bbfb12a76 | Active | No |
| 10.0.0.0/16 | local | Active | No |

- And Subnet Associations :

1. App-subnet-az1
2. App-subnet-az2
3. DB-subnet-az1
4. DB-subnet-az2

rtb-0e547ceab464d6e39 / Private-RT

The screenshot shows the 'Subnet associations' tab for the route table 'rtb-0e547ceab464d6e39'. It lists four subnets under 'Explicit subnet associations': app-subnet-az1, app-subnet-az2, db-subnet-az1, and db-subnet-az2. Each entry includes the subnet ID and its IPv4 CIDR range.

| Name | Subnet ID | IPv4 CIDR | IPv6 CIDR |
|----------------|--------------------------|-------------|-----------|
| app-subnet-az1 | subnet-09b1a64e500da2e40 | 10.0.3.0/24 | - |
| app-subnet-az2 | subnet-0a08704598c355430 | 10.0.7.0/24 | - |
| db-subnet-az1 | subnet-0773dbf9d739fa49d | 10.0.4.0/24 | - |
| db-subnet-az2 | subnet-020cb1e12ee543cdc | 10.0.8.0/24 | - |

Configure Route Tables :

Stockholmeu-north-1

1. Create Public Route Table : Public-RT
2. Create Igw : igw-Public
3. And Attach to your Current VPC

igw-04667700afb21d4fb

The screenshot shows the creation of an Internet Gateway named 'igw-04667700afb21d4fb'. It displays the gateway ID, state (Detached), VPC ID (empty), and owner information. A note indicates it can now be attached to a VPC.

Associated Subnet(s):

- Public Subnet (where the Application Load Balancer is placed)

VPC > Route tables > rt-0546223c101e0774 > Edit subnet associations

The screenshot shows the 'Edit subnet associations' dialog for the route table 'rt-0546223c101e0774'. It lists available subnets and selected subnets. The selected subnet is 'Public-Subnet-AZ1'.

| Name | Subnet ID | IPv4 CIDR | IPv6 CIDR | Route table ID |
|-------------------|-------------------------|-------------|-----------|----------------------------------|
| Public-Subnet-AZ1 | subnet-020ca64a71a6a00d | 10.1.1.0/24 | - | rt-0546223c101e0774 / Public-RT |
| Web-Subnet-AZ1 | subnet-09c004878538183 | 10.1.2.0/24 | - | rt-0546223c101e0774 / Public-RT |
| DB-Subnet | subnet-0306c7115a09fe36 | 10.1.4.0/24 | - | rt-0546223c101e0774 / Private-RT |
| Web-Subnet-AZ2 | subnet-0451c457c5e6003d | 10.1.3.0/24 | - | rt-0546223c101e0774 / Public-RT |

Selected subnets:

subnet-020ca64a71a6a00d / Public-Subnet-AZ1

Cancel Save associations

4. Create Private Route Table : Private-RT

The screenshot shows the 'Edit subnet associations' page in the AWS VPC console. At the top, it says 'Available subnets (3/4)'. Below is a table with columns: Name, Subnet ID, IPv4 CIDR, IPv6 CIDR, and Router table ID. There are three subnets listed:

| Name | Subnet ID | IPv4 CIDR | IPv6 CIDR | Router table ID |
|--|-------------------------|-------------|-----------|--------------------------------------|
| Public-Subnet-AZ | subnet-025a4c471e45e008 | 10.1.1.0/24 | - | rta-07fe0203-1e3e2174 / Public-RT |
| <input checked="" type="checkbox"/> Web-Subnet-AZ1 | subnet-08a30d93e338613 | 10.1.2.0/24 | - | rtab-07f92044-05e2-1d |
| <input checked="" type="checkbox"/> DB-Subnet | subnet-008a67133049e79 | 10.1.4.0/24 | - | rta-050054-05e2-05e2-1d / Private-RT |
| <input checked="" type="checkbox"/> Web-Subnet-AZ2 | subnet-0d51c457e5bd8030 | 10.1.5.0/24 | - | rtab-07759204-05e2-05e2-1d |

Below the table is a section titled 'Selected subnets' containing three items: 'subnet-008a67133049e79 / DB-Subnet', 'subnet-0d51c457e5bd8030 / Web-Subnet-AZ2', and 'subnet-08a30d93e338613 / Web-Subnet-AZ1'. At the bottom right are 'Cancel' and 'Save associations' buttons.

4. Enable VPC Peering or Route53 Latency-Based Routing:

If inter-region communication is required, configure VPC peering between US East and South America.

Alternatively, configure Route 53 with latency-based routing to direct users to the nearest region.

1. Create VPC Peering Connection

- In the US East region: N. Virginiaus-east-1

Go to VPC Console → Peering Connections → Create Peering Connection

Requester VPC: e.g., vpc-06115a28a1b5eadbf (US East)

Acceptor VPC: Stockholmeu-north-1 VPC (you'll select the region and VPC ID)

Copy VPC ID from another Region and Paste in this Acceptor Vpc ID

The screenshot shows the 'Create peering connection' page in the AWS VPC console. At the top, it says 'Peering connection settings' and 'Name - optional' (with a note: 'Create a tag with a key of "Name" and a value that you specify'). The name is set to 'US-East-to-EU-North-peer'.

Below is the 'Select a local VPC to peer with' section:

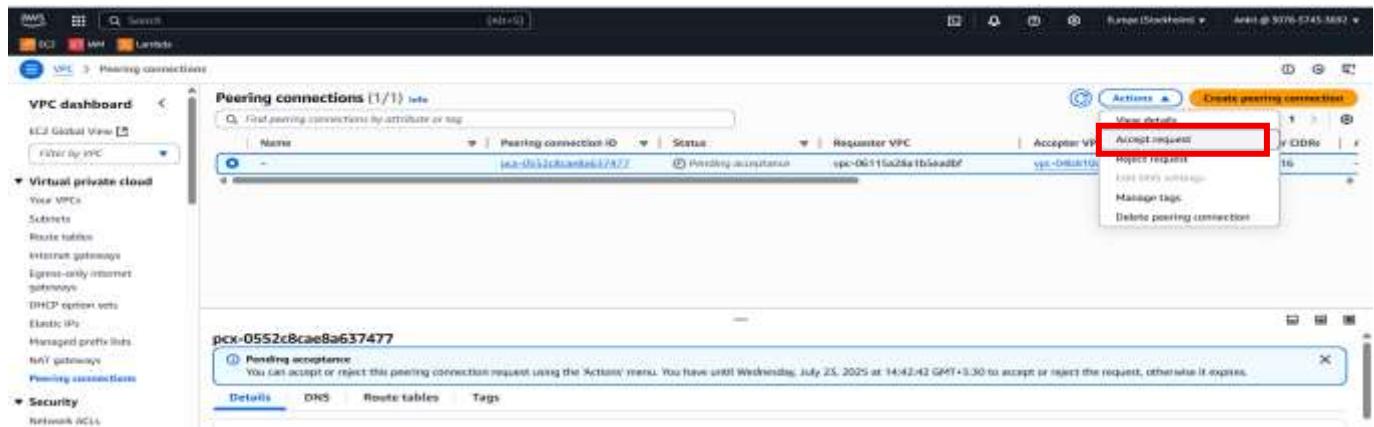
- VPC ID (Requester): vpc-06115a28a1b5eadbf (VPC-US-EAST)
- VPC CIDRs for vpc-06115a28a1b5eadbf (VPC-US-EAST):

| CIDR | Status | Status reason |
|-------------|------------|---------------|
| 10.0.0.0/16 | Associated | - |

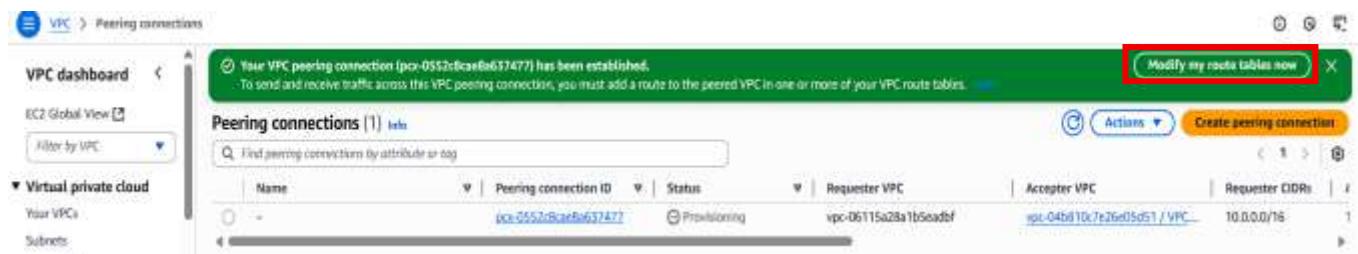
Below is the 'Select another VPC to peer with' section:

- Account: My account
- Region: Another Region (selected), Europe (Stockholm) (eu-north-1)
- VPC ID (Acceptor): vpc-04b810c7e26e05d5

And After Created VPC Peering then Go to another region - Stockholmeu-north-1 in that region in that vpc go to VPC peering connection in that after refresh that you will see one peering connection you can select that peering connection and click on Action Tab one icon will Pop-up in that Select the Accept request



After that you will see the Peering Connection is Accepted And After that you will see one green Icon will Pop-up for Modify the route table So you can modify the route table

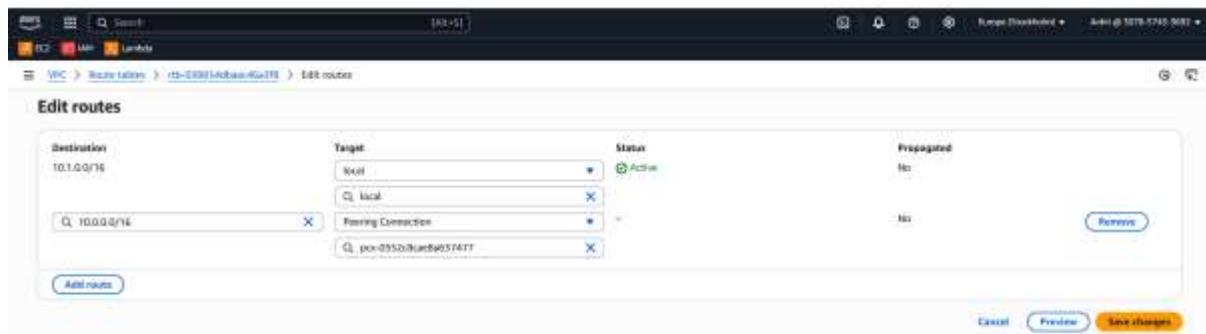


If you Clicking on Modify route table now should redirect you to the route table screen.

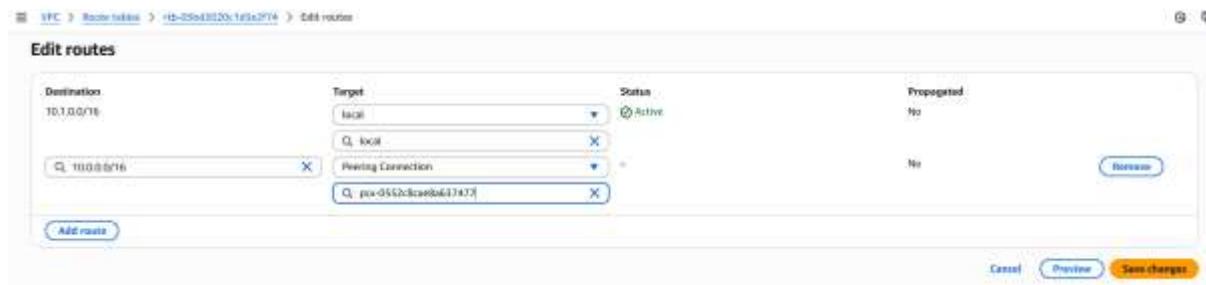
Then in that route table, Modify All Route Table those RT will use like Public and Private Route table

First modify the **Public Route** table for Peering connection route – in that click on route and click on Add route

The Destination(Like CIDR) will be the from another region (10.0.0.0/16)

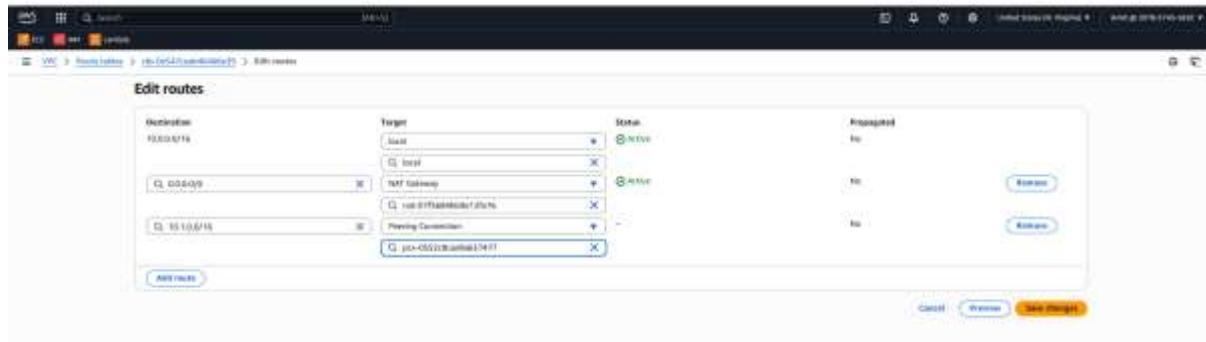


First modify the **Private Route** table

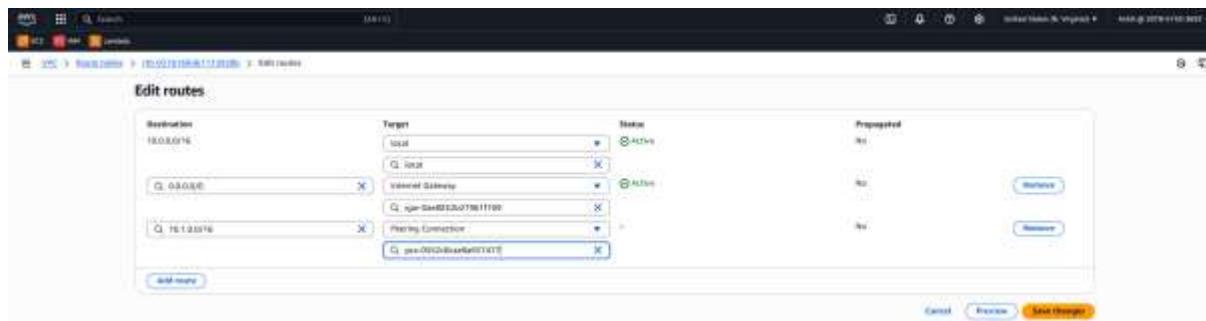


After that go to another region **N. Virginiaus-east-1** in that Region go to VPC and click on Route table, in that Route table same process will follow as like upper routing for peering for all Public and Private Route Table

Public Route Table :



Private Route Table :



- Security Group Both Region : [N. Virginiaus-east-1](#) & [Stockholmeu-north-1](#)

1. Create Security Groups: [N. Virginiaus-east-1](#)

ALB SG:

- Inbound: Allow HTTP (80) and HTTPS (443) from 0.0.0.0/0.
- Outbound: Allow all.

Web SG:

- Inbound: Allow traffic from ALB SG on port 80/443.
- Outbound: Allow to App SG and Internet (if needed).

App SG:

- Inbound: Allow traffic from Web SG on application ports (e.g., 3000, 8080).
- Outbound: Allow to DB SG.

DB SG:

- Inbound: Allow traffic from App SG on DB port (e.g., 3306 for MySQL).
- Outbound: Allow as required (typically restricted).

Security Groups : [Stockholmeu-north-1](#)

1. SG-ALB

- Inbound: Allow 80, 443 from 0.0.0.0/0
- Outbound: Allow to SG-Web on 80/443

2. SG-Web

- Inbound: Allow 80 from SG-ALB-SA
- Outbound: Allow to SG-DB on 3306

3. SG-DB

- Inbound: Allow 3306 from SG-Web
- Outbound: Allow all (default)

Network ACLs (Stateless, per subnet) :

Nacl for [N. Virginiaus-east-1](#) Regoin :

1. Public Subnet (ALB, NAT, IGW)

- Inbound:
 - Allow 80, 443 from 0.0.0.0/0
 - Allow 1024–65535 from everywhere (for return traffic)
- Outbound:
 - Allow 80, 443 to 0.0.0.0/0
 - Allow ephemeral ports 1024–65535 to 0.0.0.0/0

2. Web Subnet

- Inbound:
 - Allow 80 from ALB subnet IPs
 - Allow 22 from Bastion/your IP
 - Allow 1024–65535 from App subnet (for responses)
- Outbound:
 - Allow 8080 to App subnet
 - Allow 1024–65535 to everywhere

3. App Subnet

- Inbound:
 - Allow 8080 from Web subnet
 - Allow 1024–65535 from DB subnet
- Outbound:
 - Allow 3306 to DB subnet
 - Allow 1024–65535 to everywhere

4. DB Subnet

- Inbound:
 - Allow 3306 from App subnet
- Outbound:
 - Allow 1024–65535 to App subnet

Nacl for [Stockholmeu-north-1](#) Regoin :

1. Public Subnet

- Inbound:
 - Allow 80, 443 from 0.0.0.0/0
- Outbound:
 - Allow 80, 443 to 0.0.0.0/0

2. Web Subnet

- Inbound:
 - Allow 80 from ALB subnet
- Outbound:
 - Allow 3306 to DB subnet

3. DB Subnet

- Inbound:
 - Allow 3306 from Web subnet
- Outbound:
 - Allow ephemeral ports 1024–65535 to Web subnet

Create EC2 Role : In [N. Virginiaus-east-1](#) Regoin

Role Name: EC2-S3-CloudWatch-Role

Attach these AWS managed policies:

- AmazonS3ReadOnlyAccess
- CloudWatchAgentServerPolicy
- AmazonSSMManagedInstanceCore

The screenshot shows the 'EC2_S3_CloudWatch_Role' configuration page in the AWS IAM console. The role has been created on July 16, 2024, at 22:19 UTC+06:00. It has no activity. The ARN is arn:aws:iam::107057433692:role/EC2_S3_CloudWatch_Role. An instance profile ARN is also listed: arn:aws:iam::107057433692:instance-profile/EC2_S3_CloudWatch_Role. The 'Permissions' tab is selected, showing three attached AWS managed policies: AmazonS3ReadOnlyAccess, AmazonSSMManagedInstanceCore, and CloudWatchAgentServerPolicy. Each policy is listed with its name, type (AWS managed), and the number of entities it is attached to (1 for each).

| Policy name | Type | Attached entities |
|------------------------------|-------------|-------------------|
| AmazonS3ReadOnlyAccess | AWS managed | 1 |
| AmazonSSMManagedInstanceCore | AWS managed | 1 |
| CloudWatchAgentServerPolicy | AWS managed | 2 |

Web Tier ASG (with ALB)

Create a Launch Template (Recommended)

You need a Launch Template before creating an Auto Scaling Group.

Launch Template -1

- Name: New-launch-template
- AMI: Amazon Linux 2 or your custom AMI
- Instance Type: t3.micro or as required
- Key Pair: Select your key
- IAM Role: EC2_S3_CloudWatch_Role
- Security Group: SG-Web
- User Data



Create Auto Scaling Group (ASG) : : Web Tier ASG (with ALB) & App Tier ASG (with NLB)

Create Target Group for Web Tier

- Go to EC2 → Target Groups → Create target group
- Type: Instances
- Protocol: HTTP, Port: 80
- Name: web-tg
- Health checks: Path /
- Register no targets

Create Application Load Balancer

- Go to EC2 → Load Balancers → Create Load Balancer
- Choose Application Load Balancer
- Name: web-alb
- Scheme: Internet-facing
- Listeners: HTTP (port 80)
- Subnets: Public subnets – 1&2
- Security Group: Allow HTTP (port 80) from 0.0.0.0/0
- Target group: Attach web-tg
- Create

Create Auto Scaling Group

- Go to EC2 → Auto Scaling Groups → Create
- Choose web-launch-template
- Name: web-asg
- VPC: Select
- Subnets: Public subnets
- Attach to ALB: Yes → select web-alb → target group web-tg
- Desired/Min/Max capacity: 2 / 2 / 4
- Enable health checks (ALB + EC2)
- Scaling Policy:
 - Choose Target tracking
 - Metric: CPU utilization @ 50%
- Instance refresh: Enable
- Create ASG

App Tier ASG (with NLB)

Create Launch Template

- Go to Launch Templates → Create
- Name: app-launch-template
- AMI: Ubuntu 22.04
- Instance Type: t2.micro
- Key Pair: Use the same or new
- Security Group: APP-SG
 - Allow port 8080 from NLB SG only
- User Data

```
bash                                         ⌂ Copy code

#!/bin/bash
apt update -y
apt install nodejs npm -y
mkdir /opt/app
cd /opt/app
npm init -y
npm install express
cat <<EOF > index.js
const express = require('express');
const app = express();
app.get("/", (req, res) => res.send('Hello from App Tier'));
app.listen(8080, () => console.log('App running on port 8080'));
EOF
node index.js &
```

Create Target Group for App Tier

- Go to Target Groups → Create
- Type: Instances
- Protocol: TCP, Port: 8080
- Name: app-tg
- Health checks: TCP on port 8080
- Register no targets yet

Create Network Load Balancer

- Go to Load Balancers → Create Load Balancer
- Choose Network Load Balancer
- Name: app-nlb
- Scheme: Internal
- Listener: TCP on 8080
- Subnets: Private subnets in 2 AZs
- Target group: app-tg
- Create

Create Auto Scaling Group

- EC2 → Auto Scaling Groups → Create
- Launch Template: app-launch-template
- Name: app-asg
- VPC: Same as web tier
- Subnets: Private subnets – APP -1&2
- Attach to NLB: Yes → select app-nlb → target group app-tg
- Desired/Min/Max capacity: 2 / 2 / 4
- Scaling Policy: CPU @ 50%
- Health Checks: EC2 + NLB
- Create ASG

Install and Deploy Applications:

- Web Tier: i. Install Nginx/Apache, deploy frontend code.
- App Tier: i. Install backend dependencies (Node.js, Java, Python).
 - ii. Connect to RDS.

Create database :

- Choose a database creation method : Standard create
- Engine options-Engine type : MySQL
- Choose a Template : Free tier
- Settings : DB instance identifier - mydb-instance
 - Master username – admin
 - Master password : Click on Self Manage
 - Enable Auto generate password
- Connectivity : Your Virtual private cloud (VPC)
 - Existing VPC security groups : DB-SG
 - Availability Zone : us-east-1a or us-east-2a

ALB & NLB for Stockholmeu-north-1 Regoin

Create Application Load Balancer (ALB)

Go to EC2 > Load Balancers > Create Load Balancer > Application Load Balancer

Configure Load Balancer

- Name: my-alb
- Scheme: Internet-facing
- IP Address type: IPv4
- Listeners: HTTP (Port 80) or HTTPS (Port 443)
- Availability Zones: Select your VPC, then select 2 Public Subnets in EU-North (e.g., eu-north-1a, eu-north-1b)

Configure Security Groups

- Create or use an SG that allows:
 - Port 80 (HTTP) or 443 (HTTPS) from the internet (0.0.0.0/0)
 - Attach this to ALB

Configure Target Group

- Name: web-target-group
- Target type: Instance
- Protocol: HTTP (or HTTPS)
- Port: 80 (or your app port)
- Health checks: HTTP → / or your app's health check path

Create Network Load Balancer (NLB)

Go to EC2 > Load Balancers > Create Load Balancer > Network Load Balancer

Configure Load Balancer

- Name: my-nlb
- Scheme: Internal (for internal app communication)
- IP Address type: IPv4
- Listeners: TCP (e.g., port 3000 or 8080)
- Availability Zones: Select 2 Private Web Subnets (same region)

Configure Target Group

- Name: app-target-group
- Target type: Instance
- Protocol: TCP
- Port: 3000 (or your backend port)
- Health checks: TCP or custom HTTP path if needed

Create database :

- Choose a database creation method : Standard create
- Engine options-Engine type : MySQL
- Choose a Template : Free tier
- Settings : DB instance identifier - mydb-instance
 - Master username – admin
 - Master password : Click on Self Manage
 - Enable Auto generate password
- Connectivity : Your Virtual private cloud (VPC)
 - Existing VPC security groups : DB-SG
 - Availability Zone : eu-north-1

Create S3 Buckets (for static files / backups)

- Create Bucket
- Go to S3 → Create bucket
- Name: your-project-assets-bucket
- Region: same as app
- Enable Versioning : Under Bucket Versioning, select Enable
- Enable Encryption : Choose: Amazon S3-managed keys (SSE-S3) or KMS
- Create bucket

Create a Public Hosted Zone

- Go to Route 53 → Hosted Zones → Create Hosted Zone
 - Domain name: harshitraj8507.wixsite.com/my-site
 - Type: Public hosted zone
- Add Latency-Based Routing Records
 - In your hosted zone, click Create Record
 - Add 2 records pointing to your ALBs in both regions:

| Name | Type | Alias | Region | Routing Policy |
|--|-----------|--|------------|----------------|
| https://harshitraj8507.wixsite.com/my-site | A (alias) | <input checked="" type="checkbox"/> Alias to ALB | us-east-1 | Latency-based |
| https://harshitraj8507.wixsite.com/my-site | A (alias) | <input checked="" type="checkbox"/> Alias to ALB | Eu-north-1 | Latency-based |

Configure Health Checks

- Go to Route 53 → Health Checks → Create health check
 - Target: Public IP or domain of ALB
 - Protocol: HTTP
 - Path: /health or / (depends on your app)
- Associate health check with the Route 53 record
- If a region fails health check, Route 53 reroutes to healthy region

Enable AWS WAF for ALB

Create a Web ACL (WAF)

- Go to WAF → Web ACLs → Create Web ACL
 - Name: myapp-waf
 - Region: same as your ALB
 - Resource type: Application Load Balancer

- Associate with your ALB(s)

Add Rules to WAF

- Add Managed Rule Groups:
 - AWSManagedRulesCommonRuleSet (includes SQLi, XSS protection)
 - AWSManagedRulesKnownBadInputsRuleSet
- Optional: Add Custom rules
 - e.g., Block IPs, Rate-based rules

Add CloudWatch Alarms (Optional)

- Go to CloudWatch → Alarms → Create Alarm
- Set threshold for metrics (e.g., CPU > 70%)
- Set actions:
 - Trigger scale out/in
 - Send notification (SNS topic)

Enable CloudWatch Monitoring

Setup Dashboards (EC2, ALB, RDS)

- Go to CloudWatch → Dashboards → Create dashboard
- Name: prod-monitoring-dashboard
- Add widgets:
 - EC2: CPUUtilization, StatusCheckFailed
 - ALB: RequestCount, TargetResponseTime, HTTPCode_ELB_5XX_Count
 - RDS: FreeStorageSpace, CPUUtilization, DatabaseConnections