

## Problem Statement:

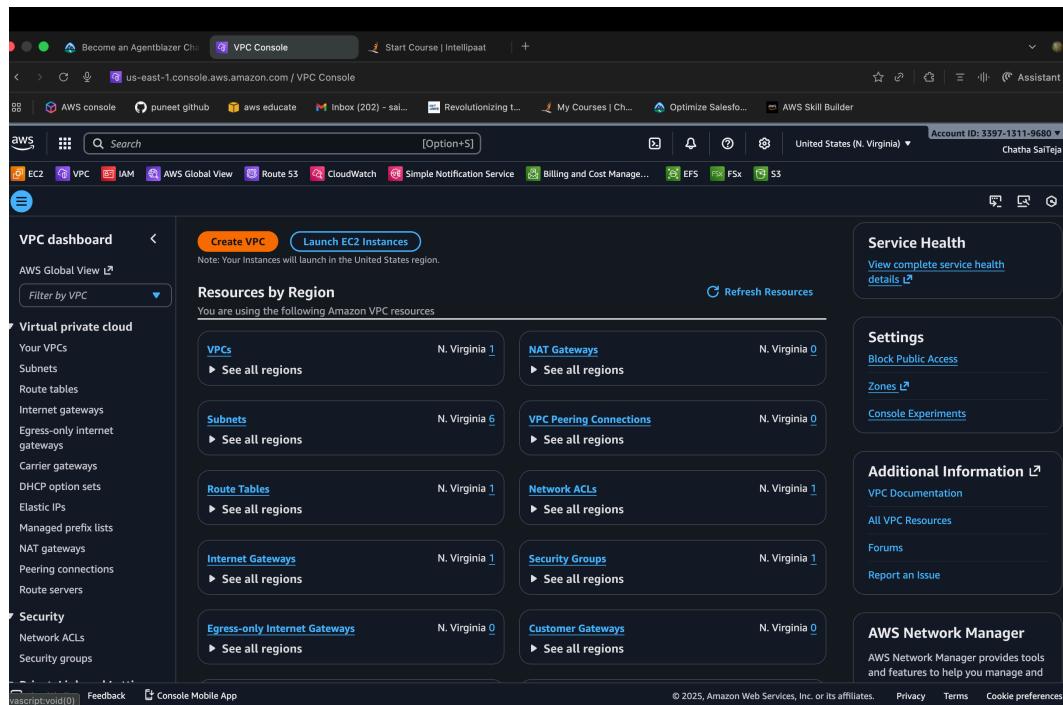
You work for XYZ Corporation and based on the expansion requirements of your corporation you have been asked to create and set up a distinct Amazon VPC for the production and development team. You are expected to perform the following tasks for the respective VPCs

### Production Network:

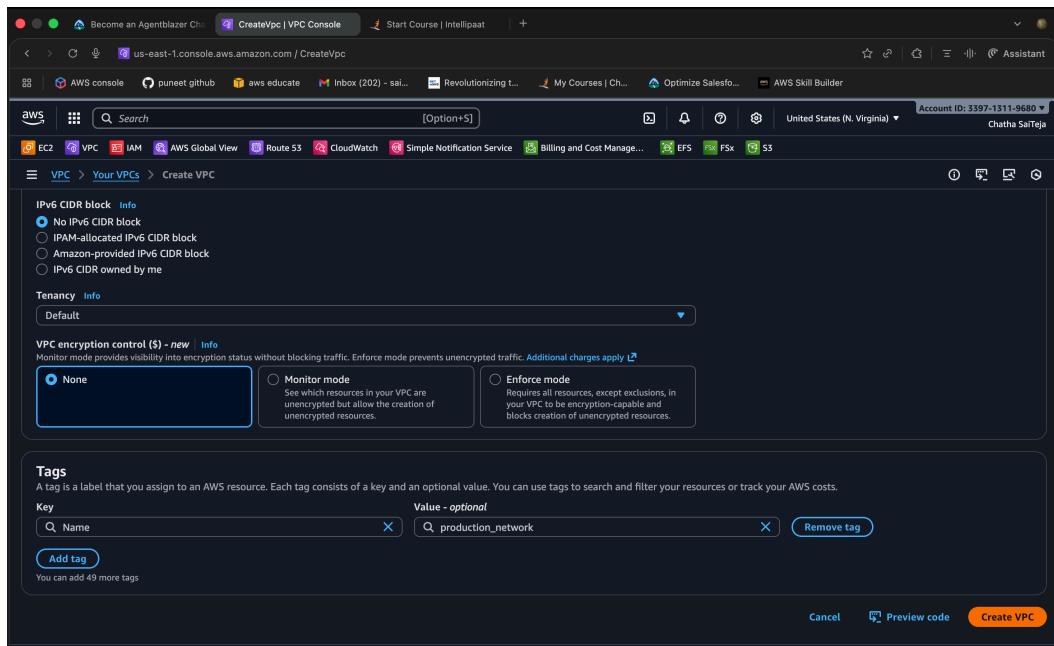
1. Design and build a 4-tier architecture.
2. Create 5 subnets out of which 4 should be private named app1, app2, dbcache and db and one should be public, named web.
3. Launch instances in all subnets and name them as per the subnet that they have been launched in.
4. Allow dbcache instance and app1 subnet to send internet requests.
5. Manage security groups and NACLs.

### Step -By- Step Procedure:-

Step 1:-In the AWS VPC console, click Create VPC, assign a CIDR block (e.g., 10.0.0.0/16) to define the network range, name it Production Network, and click Create to establish the VPC.



Click on create Vpc

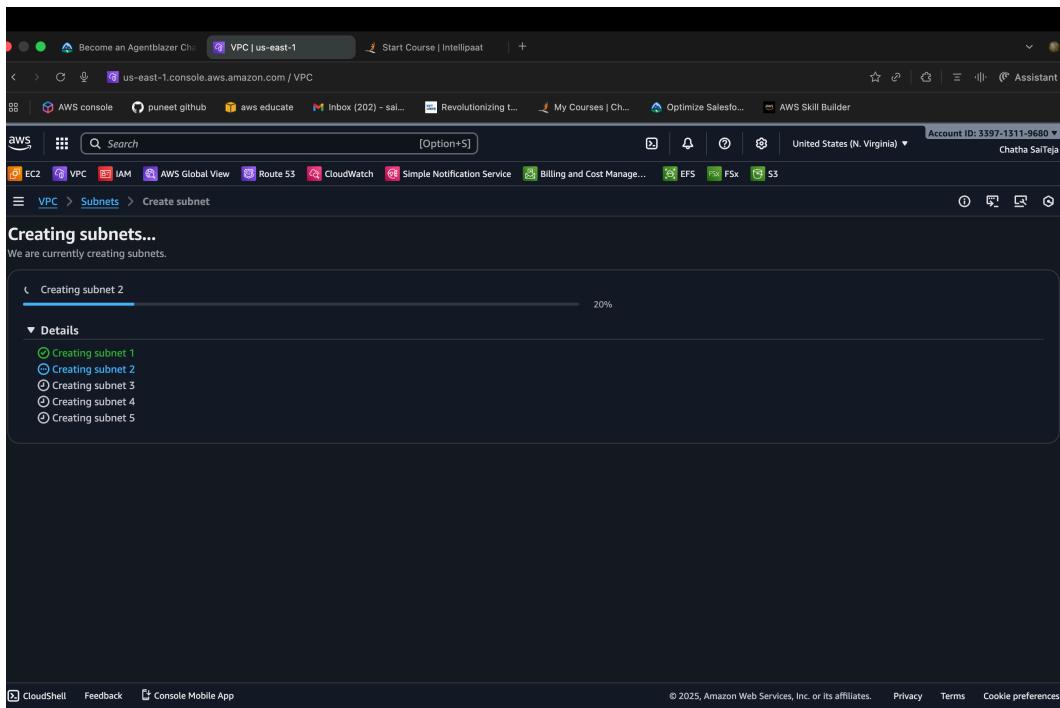


Give network range and specify the name of vpc and click on create

Step 2:- In production network we have to create 5 subnets out of which 4 should be private named app1, app2, dbcache and db and one should be public, named web , To create subnet,go to subnets then give name and network range of subnet which should be under production network range.

Subnets (6) Info							
Last updated 1 minute ago							
	Name	Subnet ID	State	VPC	Actions	Block Public...	IPv4
<input type="checkbox"/>	-	subnet-06435103899767477	<input checked="" type="radio"/> Available	vpc-07c56690086316e6f	<input checked="" type="radio"/>	Off	172.0.0.0/16
<input type="checkbox"/>	-	subnet-0f48341bf780d262d	<input checked="" type="radio"/> Available	vpc-07c56690086316e6f	<input checked="" type="radio"/>	Off	172.0.0.0/16
<input type="checkbox"/>	-	subnet-0a0cd7877470de6e	<input checked="" type="radio"/> Available	vpc-07c56690086316e6f	<input checked="" type="radio"/>	Off	172.0.0.0/16
<input type="checkbox"/>	-	subnet-0a1f17210dc4e010f	<input checked="" type="radio"/> Available	vpc-07c56690086316e6f	<input checked="" type="radio"/>	Off	172.0.0.0/16
<input type="checkbox"/>	-	subnet-0a0f5a0a46a1826e41	<input checked="" type="radio"/> Available	vpc-07c56690086316e6f	<input checked="" type="radio"/>	Off	172.0.0.0/16
<input type="checkbox"/>	-	subnet-02cd008ff2799e5a0	<input checked="" type="radio"/> Available	vpc-07c56690086316e6f	<input checked="" type="radio"/>	Off	172.0.0.0/16

Click on create



Five subnets are creating

Subnets (5) <span style="float: right;">Actions ▾ Create subnet</span>						
<span style="float: left;">Filter by VPC</span> <span style="float: right;">Last updated less than a minute ago</span>						
<span style="color: green;">✔ You have successfully created 5 subnets: subnet-00e5996c2f2bc0f4c, subnet-0d8ee9ec0f5c17c98, subnet-09b6320c5afa4462f, subnet-090603ad4fdce87d, subnet-0efd71a42e9155609</span>						
Name	Subnet ID	State	VPC	Block Public...	IPv4	
app2	subnet-0d8ee9ec0f5c17c98	Available	vpc-0125a255ea037003d   pro...	Off	10.10.10.0/24	
web	subnet-0efd71a42e9155609	Available	vpc-0125a255ea037003d   pro...	Off	10.10.11.0/24	
dbcache	subnet-09b6320c5afa4462f	Available	vpc-0125a255ea037003d   pro...	Off	10.10.12.0/24	
app1	subnet-00e5996c2f2bc0f4c	Available	vpc-0125a255ea037003d   pro...	Off	10.10.13.0/24	
db	subnet-090603ad4fdce87d	Available	vpc-0125a255ea037003d   pro...	Off	10.10.14.0/24	

Five subnets were created

Step 3:- Create an Internet Gateway by clicking Create, naming it appropriately, and clicking Create. Then, go to Actions → Attach to VPC and attach it to the Production Network VPC, enabling internet connectivity for the public subnet

The screenshot shows the AWS VPC Internet Gateways page. On the left, there's a navigation sidebar with sections like VPC dashboard, Virtual private cloud, Security, and others. The main area displays a table titled 'Internet gateways (1) Info'. The table has columns for Name, Internet gateway ID, State, VPC ID, and Owner. One row is listed with the values: Name is empty, Internet gateway ID is 'ipw-028f0719d61d6874c', State is 'Attached', VPC ID is 'vpc-07c56690086316e6f', and Owner is '33971311196'. Below the table, there's a message 'Select an internet gateway above'.

Click on create Internet Gateway

The screenshot shows the 'Create internet gateway' wizard. Step 1 is 'Internet gateway settings'. It has a 'Name tag' section where 'production\_network\_ig' is entered into a text input field. Below it is a 'Tags - optional' section where a single tag 'Name' with value 'production\_network\_ig' is added. At the bottom right of the form are 'Cancel' and 'Create internet gateway' buttons.

Specify the name (production\_network\_ig)

VPC dashboard < Internet gateways (1/2) Info

Name	Internet gateway ID	State	VPC ID
production_network_ig	igw-04ebacf0c8d903345	Attached	vpc-07c56...

Actions Create internet gateway

- View details
- Attach to VPC**
- Detach from VPC
- Manage tags
- Delete internet gateway

Owner: 3397131196

igw-04ebacf0c8d903345 / production\_network\_ig

Details Tags Details

Go to Actions -> click on attach and then attach to production network

Step 4:- When subnets are created, a route table with only local routes is available by default, but we still need to create separate public and private route tables; each new route table initially contains only local routes, so for the public route table we must add a default route pointing to the Internet Gateway to allow the public subnet to send traffic to the internet.

Create route table Info

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.  
public\_route\_table

VPC  
The VPC to use for this route table.  
vpc-0125a255ea037003d (production\_network)

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 public\_route\_table Remove

Add new tag  
You can add 49 more tags.

Create route table

Give name and select production(VPC)

The screenshot shows the 'Edit routes' section of the AWS VPC console. It lists two routes:

- A route to destination 10.10.0.0/18 with target 'local' and status 'Active'. Propagated: No, Route Origin: CreateRouteTable.
- A route to destination 0.0.0.0/0 with target 'Internet Gateway' and status '-'. Propagated: No, Route Origin: CreateRoute. A 'Remove' button is visible next to this entry.

Buttons at the bottom include 'Add route', 'Cancel', 'Preview', and 'Save changes'.

Add internet gateway route to public route table

The screenshot shows the 'RouteTableDetails' interface for route table rtb-0a33f6cfe190a0397. The 'Routes' tab is selected, showing one route:

Destination	Target	Status	Propagated	Route Origin
10.10.0.0/18	local	Active	No	CreateRouteTable

On the right, a context menu is open with options: Set main route table, Edit subnet associations, Edit edge associations, Edit route propagation, Edit routes, Manage tags, and Delete.

Step 5:-After creating the two route tables (public and private), we need to update the subnet associations so that four private subnets are attached to the private route table and the single public subnet is attached to the public route table.

The screenshot shows the AWS VPC Route Table Details page. The route table ID is rtb-0a33f6cfe190a0397. The 'Actions' menu is open, and the 'Edit subnet associations' option is highlighted. The 'Routes' tab is selected, showing one route entry: Destination 10.10.0.0/18, Target local, Status Active, Propagated No, and Route Origin Create Route Table.

Click on edit subnet associations

The screenshot shows the 'Edit subnet associations' dialog box. Under 'Available subnets (4/5)', there is a table with columns: Name, Subnet ID, IPv4 CIDR, IPv6 CIDR, and Route table ID. The subnets listed are: app2 (subnet-0d8ee9ec0f5c17c98, 10.10.1.0/24, Main (rtb-0ed2633b74c6b77ac)), web (subnet-0ef71a42e9155609, 10.10.13.0/24, Main (rtb-0ed2633b74c6b77ac)), dbcache (subnet-09b6320c5afa4462f, 10.10.10.0/24, Main (rtb-0ed2633b74c6b77ac)), app1 (subnet-00e5996c2f2bc0f4c, 10.10.0.0/24, Main (rtb-0ed2633b74c6b77ac)), and db (subnet-0c90603ad4fdce87d, 10.10.12.0/24, Main (rtb-0ed2633b74c6b77ac)). Under 'Selected subnets', the subnets app2, dbcache, app1, and db are listed. At the bottom right are 'Cancel' and 'Save associations' buttons.

Select 4 subnets(app1, app2, dbcache, db) and save associations

You have successfully updated subnet associations for rtb-0a33f6cfe190a0397 / private\_route\_table.

Name	Route table ID	Explicit subnet associations	Edge associations
-	rtb-0ed2633b74c6b77ac	-	-
-	rtb-0e7e8f5dbe66cae2	-	-
<input checked="" type="checkbox"/> public_route_table	rtb-0032c55da4adc634e	-	-
<input type="checkbox"/> private_route_table	rtb-0a33f6cfe190a0397	4 subnets	-

For public route table click on edit subnet

Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (1/5)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
app2	subnet-0d8ee9ec0f5c17c98	10.10.1.0/24	-	rtb-0a33f6cfe190a0397 / private_route_table
<input checked="" type="checkbox"/> web	subnet-0efd71a42e9155609	10.10.13.0/24	-	Main (rtb-0ed2633b74c6b77ac)
dbcache	subnet-09b6320c5af94462f	10.10.10.0/24	-	rtb-0a33f6cfe190a0397 / private_route_table
app1	subnet-00e5996c2f2bc0f4c	10.10.0.0/24	-	rtb-0a33f6cfe190a0397 / private_route_table
db	subnet-0c90603ad4fdce87d	10.10.12.0/24	-	rtb-0a33f6cfe190a0397 / private_route_table

Selected subnets

subnet-0efd71a42e9155609 / web

Cancel Save associations

Select public subnet(web) and click on save

Step 6:- "We have created a production VPC with five subnets (app1, app2, db, dbcache, and web), attached an Internet Gateway, and set up two route tables (public and private); now we need to launch EC2 instances in each subnet.

The screenshot shows the AWS EC2 Dashboard for the us-east-1 region. The left sidebar includes sections for Dashboard, Instances, Images, and Elastic Block Store. The main area displays EC2 resources: 0 instances (running), 0 auto scaling groups, 0 capacity reservations, 0 dedicated hosts, 0 elastic IPs, 1 instance, 2 key pairs, 0 load balancers, 0 placement groups, 3 security groups, 0 snapshots, and 0 volumes. Below this, there's a 'Launch instance' section with a 'Launch Instance' button and a note about launching in the United States (N. Virginia) Region. To the right, there's a 'Service health' section showing AWS Health Dashboard status and a 'Zones' table with Zone name 'us-east-1a' and Zone ID 'use1-az2'. On the far right, there are 'Account attributes' and 'Explore AWS' sections.

Click on Launch Instance

The screenshot shows the 'Launch an instance' configuration page. It includes sections for 'Add security group rule', 'Advanced network configuration', 'Configure storage' (with 1x 8 GiB gp3 volume), 'Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage', 'File system', 'Advanced details', and 'Summary' (which shows 1 instance). There are also sections for 'Software Image (AMI)', 'Virtual server type (instance type)', 'Firewall (security group)', 'Storage (volumes)', and 'Free tier' information. At the bottom, there are 'Cancel', 'Launch instance', and 'Preview code' buttons.

Give instance details and click on launch

The screenshot shows the AWS EC2 Instances page. The left sidebar is collapsed. The main area displays a table of instances with the following details:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Publ.
app2_instance	i-0b6e4bc8a53abe1bf	Running	t2.nano	2/2 checks passed	<a href="#">View alarms</a>	us-east-1b	-
db_instance	i-0ccf92b248db8250a	Running	t2.nano	Initializing	<a href="#">View alarms</a>	us-east-1b	-
app1_instance	i-0ba6fabed817dfc	Running	t2.nano	2/2 checks passed	<a href="#">View alarms</a>	us-east-1b	-
dbcache_insta...	i-0a1e7c0582cdf5b36	Running	t2.nano	Initializing	<a href="#">View alarms</a>	us-east-1b	-
web_instance	i-05164e6c9de9298ed	Running	t2.nano	Initializing	<a href="#">View alarms</a>	us-east-1b	-

We created five instances in five subnets

Step 7:-To establish SSH connectivity to private instances in the dbcache and app1 subnets through the web instance (which acts as a bastion/jump server), first connect to the web instance in the public subnet, then from within that session, connect to the private instances by executing the SSH command: `ssh -i <privateKey> ec2-user@<privateIp>`, where `<privateKey>` is the path to your EC2 key pair file and `<privateIp>` is the private IP address of the target instance. This multi-hop SSH connection allows instances in private subnets to establish outbound internet connectivity and communicate securely without direct internet exposure.

```

Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

Last login: Sun Nov 30 06:50:58 2025 from 18.206.107.29
[ec2-user@ip-10-10-13-99 ~]$ vi app.pem
[ec2-user@ip-10-10-13-99 ~]$ ls -lstr
total 4
4 -rw-r--r--. 1 ec2-user ec2-user 1675 Nov 30 06:56 app.pem
[ec2-user@ip-10-10-13-99 ~]$ chmod 400 app.pem
[ec2-user@ip-10-10-13-99 ~]$ ls -lstr
total 4
4 -----. 1 ec2-user ec2-user 1675 Nov 30 06:56 app.pem
[ec2-user@ip-10-10-13-99 ~]$ ssh -i app.pem ec2-user@10.10.10.76
The authenticity of host '10.10.10.76 (10.10.10.76)' can't be established.
ED25519 key fingerprint is SHA256:qQcNsZNG519Qx4bvKicWTasb/ItD21QC61b9wsxs.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.10.76' (ED25519) to the list of known hosts.

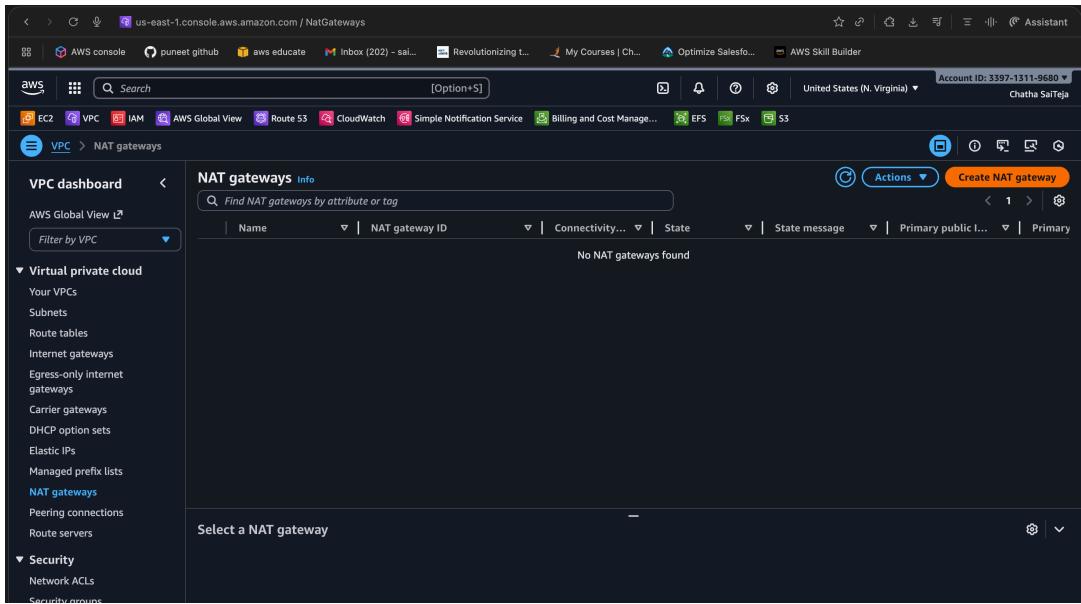
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

i-05164e6c9de9298ed (web_instance)

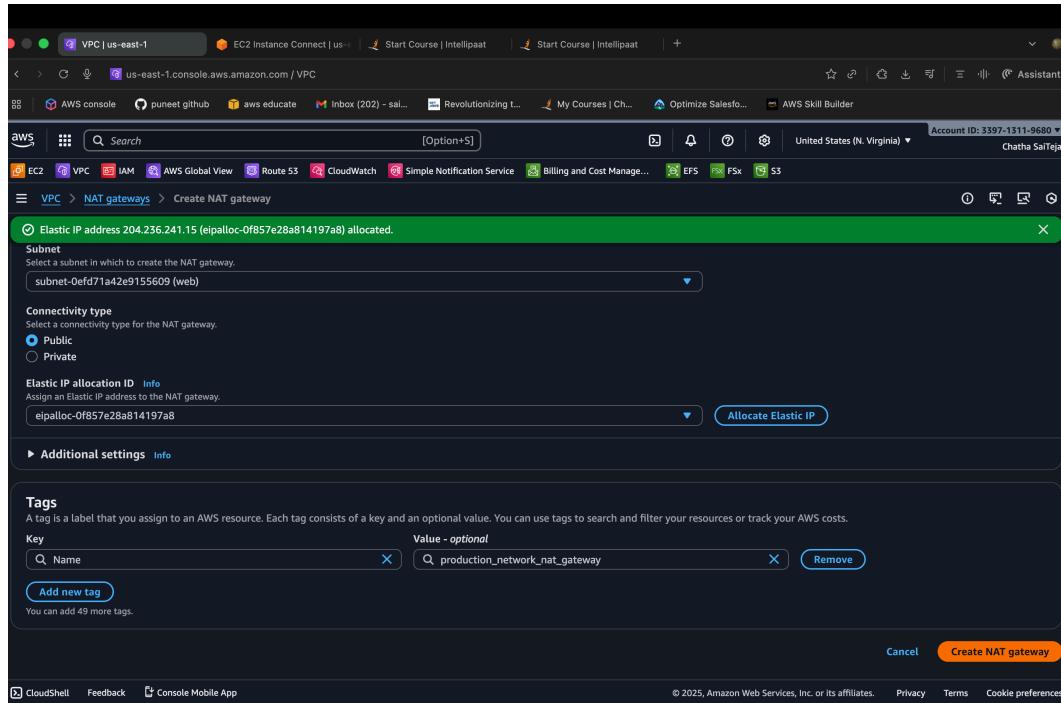
```

We connected db cache Instance through web by ssh connectivity

Step 8:- To enable outbound internet connectivity from instances in private subnets, a NAT (Network Address Translation) Gateway must be created and placed in the public subnet. During NAT Gateway creation, an Elastic IP address must be allocated and associated with the gateway, which serves as the static public IP address used for all outbound internet communications originating from the private subnet instances. This configuration allows private instances to initiate outbound requests to the internet while remaining isolated from inbound internet traffic



Click on create Nat Gateway



Give name and allocate elastic Ip then click on create

Step 9 :-Now update the private route table by adding a route that points internet-bound traffic to the NAT Gateway. After this, the private-subnet instances can send outbound internet requests. Connect to the instance and test with a command like **ping google.com**; you should see successful packet responses, confirming internet access.

The screenshot shows the AWS VPC RouteTables console. On the left, there's a sidebar with 'Route tables' selected under 'Virtual private cloud'. The main area displays a table titled 'Route tables (1/4) Info' with one entry: 'private\_route\_table' (rtb-0a33f6cfe190a0397). A context menu is open over this row, with 'Edit routes' highlighted. Below the table, a detailed view of 'rtb-0a33f6cfe190a0397 / private\_route\_table' is shown with tabs for Details, Routes, Subnet associations, Edge associations, Route propagation, and Tags. The 'Details' tab is selected.

Click on edit routes

The screenshot shows the 'Edit routes' dialog for the 'private\_route\_table'. It lists two routes:

Destination	Target	Status	Propagated	Route Origin
10.10.0.0/18	local	Active	No	CreateRouteTable
0.0.0.0/0	NAT Gateway	-	No	CreateRoute

At the bottom, there are buttons for 'Add route', 'Cancel', 'Preview', and 'Save changes'.

Add the nat route and click on save changes

```
aws [Subnets] | VPC Console EC2 Instance Connect | us-east-1.us-east-1.console.aws.amazon.com | EC2 Instance Connect Start Course | Intellipaat Start Course | Intellipaat +  
us-east-1.us-east-1.console.aws.amazon.com | EC2 Instance Connect AWS console puneetgithub aws educate Inbox (202) - saini... Revolutionizing ... My Courses | Ch... Optimize Salesfo... AWS Skill Builder Account ID: 3397-1311-9680 ▾ ChathaSaiTeja  
aws [Subnets] Search [Option+S] AWS Global View Route 53 CloudWatch Simple Notification Service Billing and Cost Manage... EFS FSx S3 United States (N. Virginia) ▾ ChathaSaiTeja  
  
-- google.com ping statistics --  
packets transmitted, 0 received, 100% packet loss, time 100844ms  
ec2-user@ip-10-10-10-76:~$ ping google.com  
PING google.com (142.250.31.138) 56(84) bytes of data.  
-- google.com ping statistics --  
packets transmitted, 0 received, 100% packet loss, time 20786ms  
  
ec2-user@ip-10-10-10-76:~$ ping google.com  
PING google.com (142.250.31.138) 56(84) bytes of data.  
...  
bytes from bj-in-f113.1e100.net (142.250.31.113): icmp_seq=1 ttl=104 time=3.04 ms  
bytes from bj-in-f113.1e100.net (142.250.31.113): icmp_seq=2 ttl=104 time=1.92 ms  
bytes from bj-in-f113.1e100.net (142.250.31.113): icmp_seq=3 ttl=104 time=2.71 ms  
bytes from bj-in-f113.1e100.net (142.250.31.113): icmp_seq=4 ttl=104 time=1.95 ms  
bytes from bj-in-f113.1e100.net (142.250.31.113): icmp_seq=5 ttl=104 time=2.38 ms  
bytes from bj-in-f113.1e100.net (142.250.31.113): icmp_seq=6 ttl=104 time=2.27 ms  
bytes from bj-in-f113.1e100.net (142.250.31.113): icmp_seq=7 ttl=104 time=2.01 ms  
bytes from bj-in-f113.1e100.net (142.250.31.113): icmp_seq=8 ttl=104 time=1.02 ms  
bytes from bj-in-f113.1e100.net (142.250.31.113): icmp_seq=9 ttl=104 time=1.96 ms  
bytes from bj-in-f113.1e100.net (142.250.31.113): icmp_seq=10 ttl=104 time=1.93 ms  
bytes from bj-in-f113.1e100.net (142.250.31.113): icmp_seq=11 ttl=104 time=1.93 ms  
bytes from bj-in-f113.1e100.net (142.250.31.113): icmp_seq=12 ttl=104 time=1.93 ms  
  
-- google.com ping statistics --  
packets transmitted, 12 received, 0% packet loss, time 11019ms  
rtt min/avg/max/mdev = 1.921/2.199/3.036/0.349 ms  
ec2-user@ip-10-10-10-76:~$  
  
i-05164ee9c9de9298ed (web_instance)  
PublicIPs: 54.209.212.249 PrivateIPs: 10.10.13.99  
CloudShell Feedback Console Mobile App © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
```

dbcache instance can send internet requests

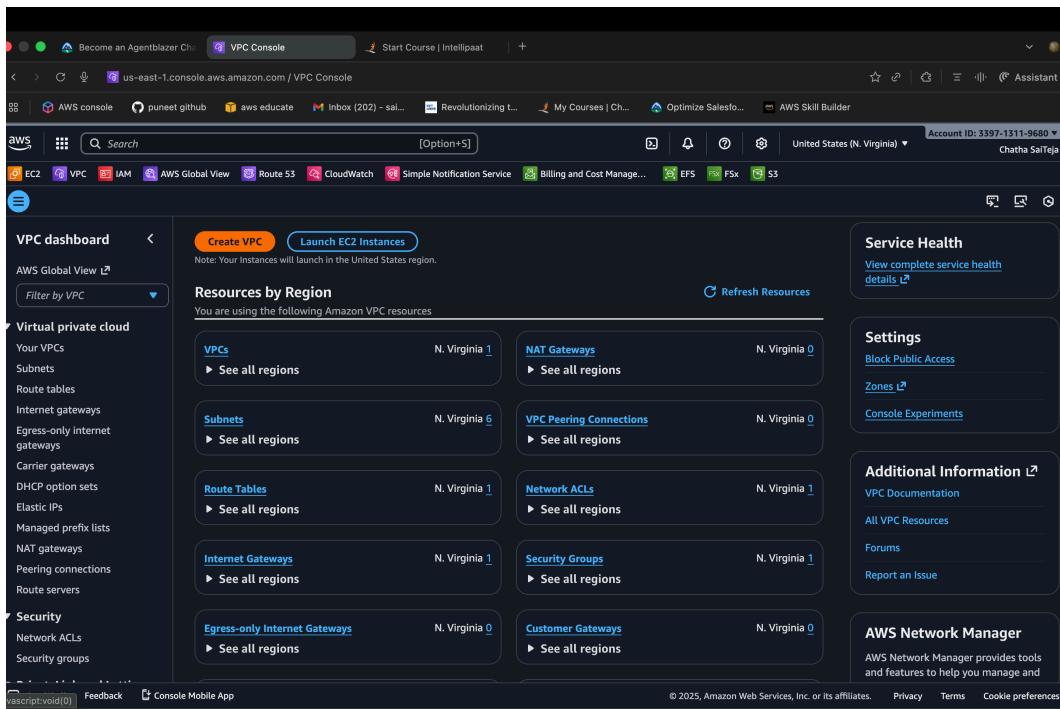
App1 instance can send internet requests

## Development Network:

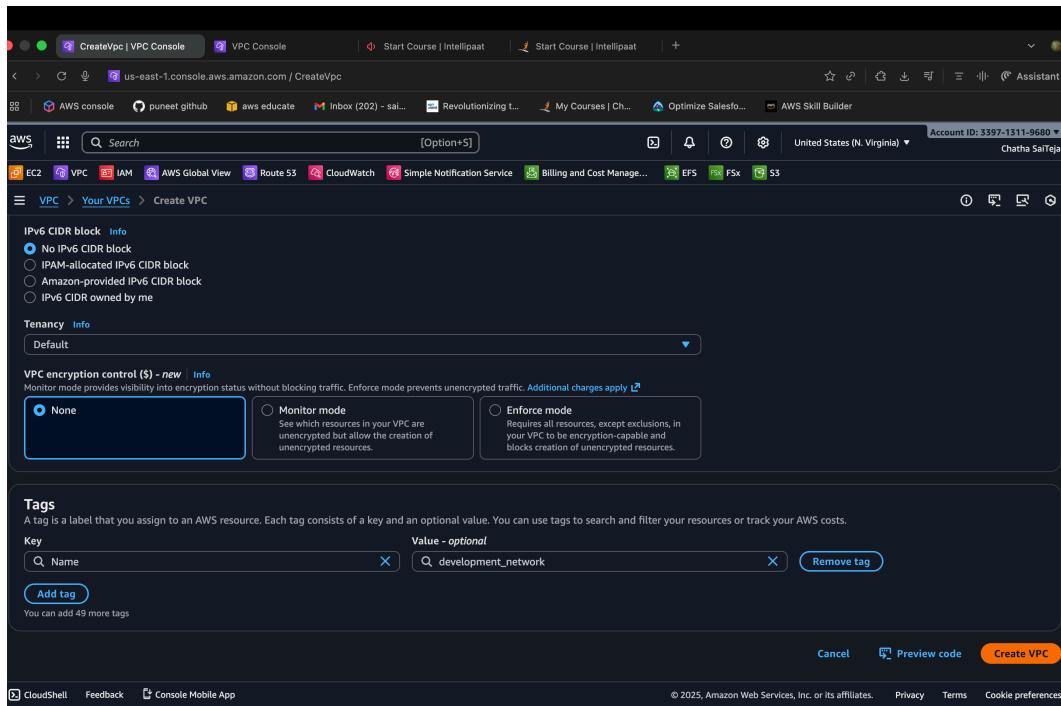
1. Design and build 2-tier architecture with two subnets named web and db and launch instances in both subnets and name them as per the subnet names.
2. Make sure only the web subnet can send internet requests.
3. Create peering connection between production network and development network.
4. Setup connection between db subnets of both production network and development network respectively.

### Step -By - Step Procedure:-

Step1 :- In the AWS VPC console, click Create VPC, assign a CIDR block (e.g., 10.0.0.0/16) to define the network range, name it Development Network, and click Create to establish the VPC.

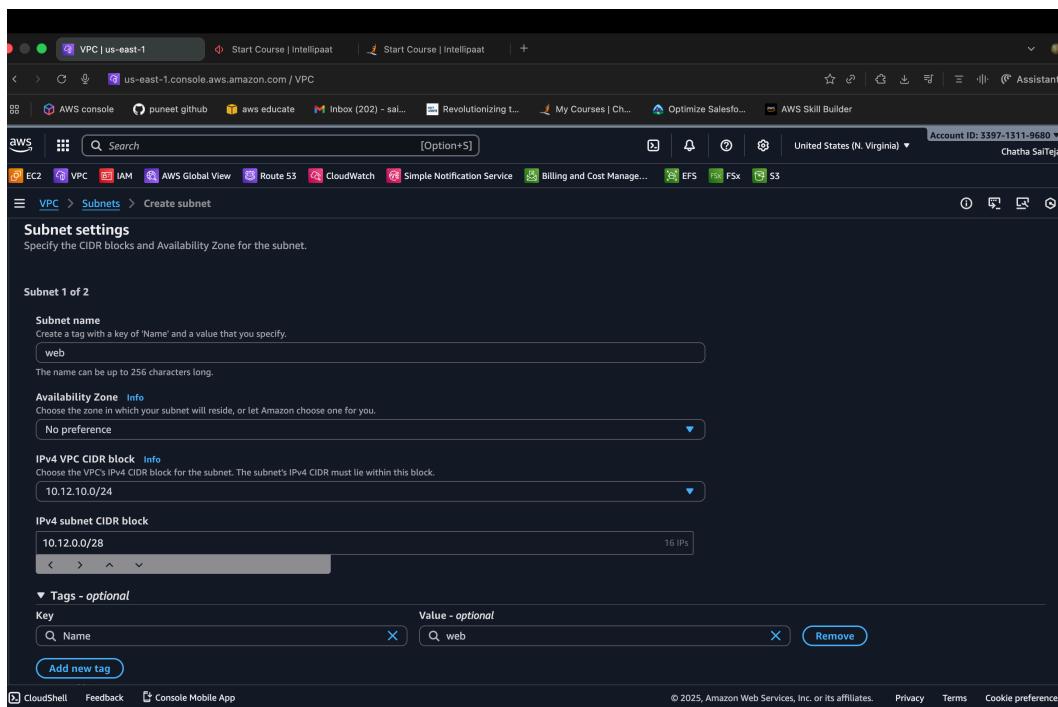


Caption



Caption

Step 2:-In Development Network we have to create two subnets named web and db and web is public subnet which can send internet requests and db is private.



Creating web subnet

Subnet 2 of 2

**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**  
 16 IPs

**Tags - optional**

Key	Value - optional
<input type="text" value="Name"/>	<input type="text" value="db"/>

[Add new tag](#)  
You can add 49 more tags.  
[Remove](#)

## Db subnet

You have successfully created 2 subnets: subnet-0786c0667f0dd4662, subnet-020a41aebe20a4746

**Subnets (2) info**

Name	Subnet ID	State	VPC	Block Public...	IPv4
db	subnet-020a41aebe20a4746	Available	vpc-0dfde86b0ab3e767a   dev...	Off	10.1...
web	subnet-0786c0667f0dd4662	Available	vpc-0dfde86b0ab3e767a   dev...	Off	10.1...

Select a subnet

Two subnets are created in development network

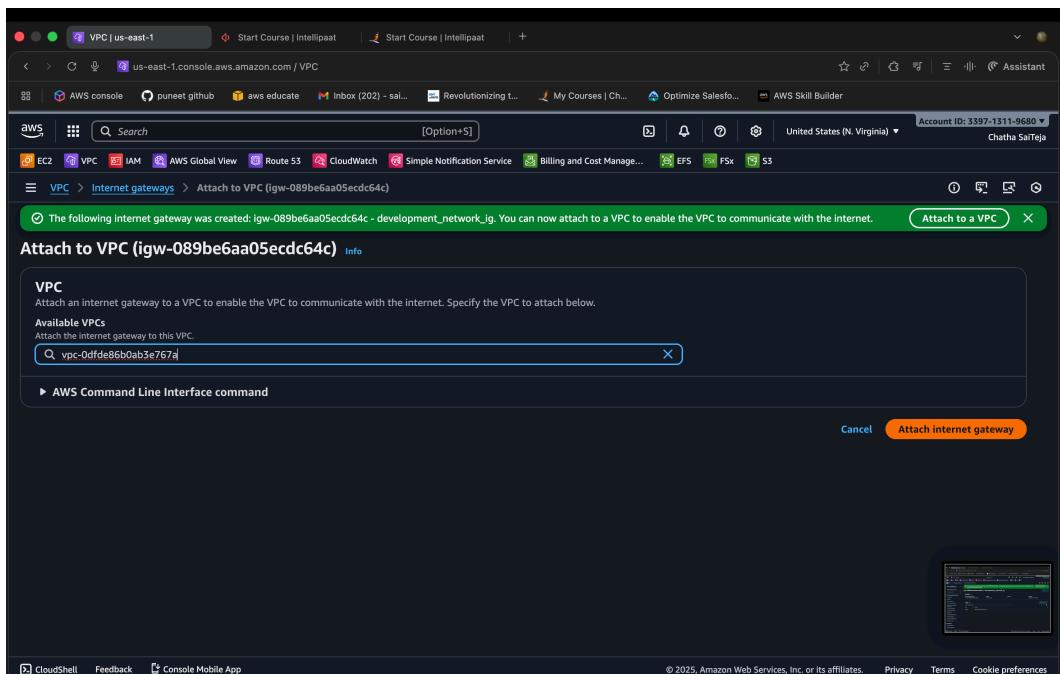
Step 3:- Create an Internet Gateway by clicking Create, naming it appropriately, and clicking Create. Then, go to Actions → Attach to VPC and attach it to the Development Network VPC, enabling internet connectivity for the public subnet

Name	Internet gateway ID	State	VPC ID	Owner
-	igw-028f0719d61d6874c	Attached	vpc-07c56690086316e6f	33971311196
production_network_ig	igw-04ebacf0c8d903345	Attached	vpc-0125a255ea037003d   production...	33971311196

Click on create

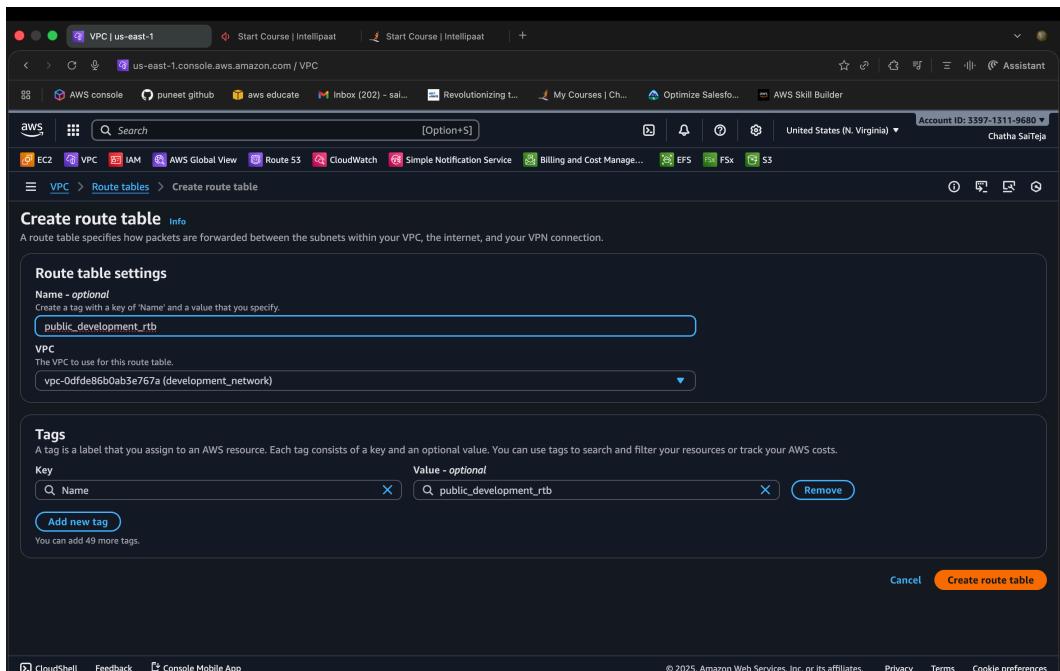
Internet gateway ID	State	VPC ID	Owner
igw-089be6aa05ecdc64c	Detached	-	33971311196

Click on attach to vpc



In Attach to Vpc mention -> development Network

Step 4 :- When subnets are created, a route table with only local routes is available by default, but we still need to create separate public route table; each new route table initially contains only local routes, so for the public route table we must add a default route pointing to the Internet Gateway to allow the public subnet to send traffic to the internet



Create public route table In development network

The screenshot shows the AWS VPC Route Tables console. On the left, there's a sidebar with 'VPC dashboard' and various VPC-related options like 'Your VPCs', 'Subnets', 'Route tables', etc. The main area shows a table of route tables with columns for Name, Route table ID, Explicit subnet associations, and Edge associations. One row is selected: 'public\_development\_rtb' (rtb-0e2e24bf2f16f6c96). A context menu is open over this row, with 'Edit routes' highlighted.

Click on edit routes

The screenshot shows the 'Edit routes' dialog for the 'rtb-0e2e24bf2f16f6c96' route table. The dialog has sections for 'Destination', 'Target', 'Status', 'Propagated', and 'Route Origin'. There are two entries: one for '10.12.10.0/24' with target 'local' and status 'Active', and another for '0.0.0.0/0' with target 'Internet Gateway' and status 'Active'. At the bottom are 'Add route', 'Cancel', 'Preview', and 'Save changes' buttons.

Add internet gateway route to public route table

Step 5:- After creating the route table (public), we need to update the subnet associations .

The screenshot shows the AWS VPC Route Tables interface. The URL is [us-east-1.console.aws.amazon.com/VPC/RouteTables/rtb-0e2e24bf2f16f6c96>Edit subnet associations](#). The 'Available subnets (1/2)' section lists two subnets: 'web' (subnet-0786c0667f0dd4662, 10.12.10.0/25, Main route table ID) and 'db' (subnet-020a41aebd20a4746, 10.12.10.128/27, Main route table ID). The 'Selected subnets' section contains 'subnet-0786c0667f0dd4662 / web'. At the bottom right are 'Cancel' and 'Save associations' buttons.

Select web and add save associations

Step 6:-Now we have to launch a instance in development network in db subnet which is a private subnet .

The screenshot shows the AWS EC2 Instances interface. The URL is [us-east-1.console.aws.amazon.com/EC2/Instances](#). The left sidebar shows 'Instances' under 'EC2'. The main table shows one instance: 'db\_instance(development\_network)' (Instance ID: i-063b02ee435e70124, Instance state: Running, Instance type: t2.nano, Status check: Initializing, Availability zone: us-east-1). The 'Actions' dropdown menu is open, showing options like 'Launch instances'.

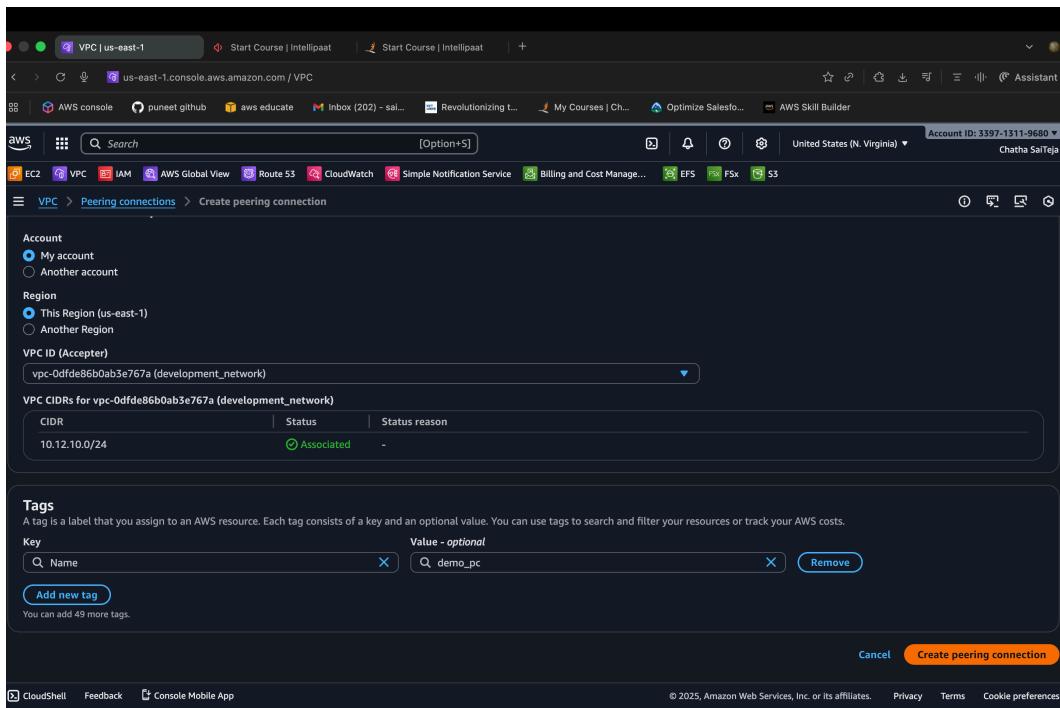
Step 7 :-To connect the Production and Development networks, create a VPC peering connection by specifying the requester and accepter VPCs, whether they're in the same region or another, and whether they belong to the same account or different accounts. After the peering connection is created, go to the accepter VPC and approve the request by selecting **Accept Request**.

The screenshot shows the AWS VPC Peering Connections page. On the left, there's a sidebar with options like 'Virtual private cloud' and 'Peering connections'. The main area is titled 'Peering connections' and shows a table with one row: 'No peering connection found'. Below the table, it says 'Select a peering connection above'.

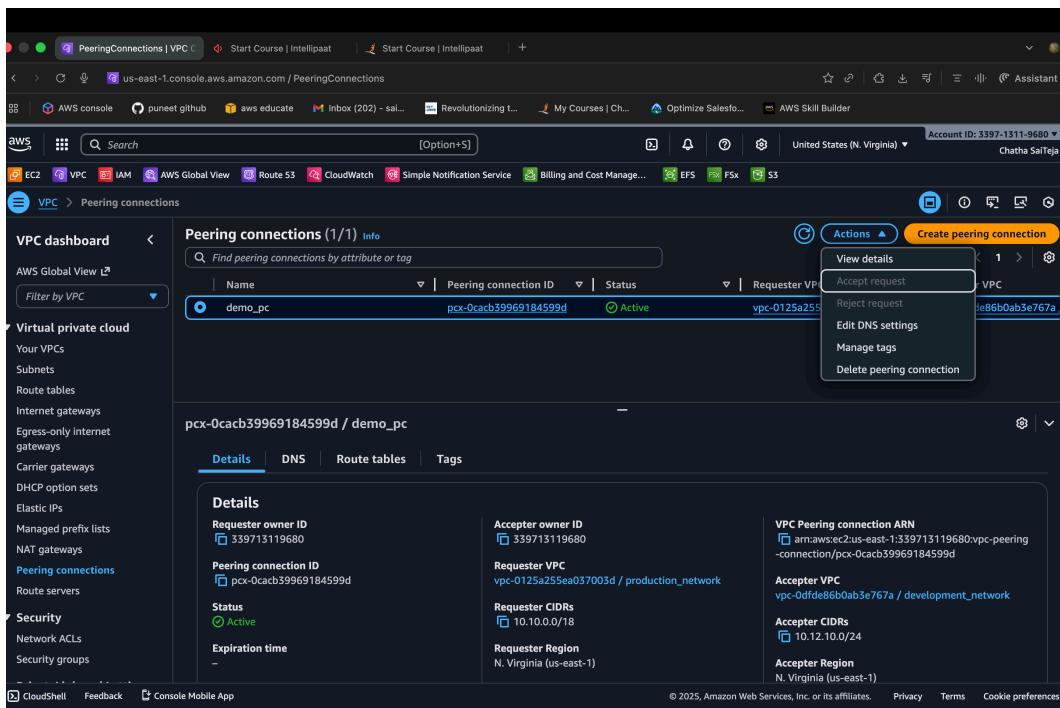
Click on create peering connection

The screenshot shows the 'Create peering connection' wizard. Step 1: 'Select a local VPC to peer with'. It shows a dropdown for 'VPC ID (Requester)' with 'vpc-0125a255ea037003d (production\_network)' selected. Below it is a table for 'VPC CIDRs for vpc-0125a255ea037003d (production\_network)'. Step 2: 'Select another VPC to peer with'. It shows 'Account' set to 'My account' and 'Region' set to 'This Region (us-east-1)'. Below it is a table for 'VPC CIDRs for vpc-0dfde86b0ab3e767a (development\_network)'. At the bottom, there are 'Next Step' and 'Cancel' buttons.

Specify the details



Click on create



Click on Accept request

Step 8:- Now we have to edit route tables ( private) of both production and Development network so that db subnet in production network can communicate with db subnet in development network.

The screenshot shows the AWS VPC Route Tables interface. The URL is [us-east-1.console.aws.amazon.com/VPC](#). The page title is "Route tables". The breadcrumb navigation shows "VPC > Route tables > rtb-0a33f6cfe190a0397 > Edit routes". The main content area is titled "Edit routes". It displays a table with columns: Destination, Target, Status, Propagated, and Route Origin. There are two rows:

Destination	Target	Status	Propagated	Route Origin
10.10.0.0/18	local	Active	No	CreateRouteTable
10.12.10.128/27	Peering Connection pcx-0cacb39969184599d	-	No	CreateRoute

At the bottom right are "Cancel", "Preview", and "Save changes" buttons. At the bottom left are links for "CloudShell", "Feedback", and "Console Mobile App". The footer includes copyright information: "© 2025, Amazon Web Services, Inc. or its affiliates." and links for "Privacy", "Terms", and "Cookie preferences".

Add pcx connection and give destination of production network subnet it could be done for development network subnet also.

Step 9:- Now it can connect with subnet in other network so that first we connect the web instance in production networks and perform ssh connectivity to connect a private instance (db instance ) from that instance to db instance in development network .

```

Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

st login: Sun Nov 30 07:52:25 2025 from 18.206.107.27
ec2-user@ip-10-10-13-99 ~]$ ls -lstr
total 4
drwxr-xr-x 2 ec2-user ec2-user 1675 Nov 30 06:56 app.pem
ec2-user@ip-10-10-13-99 ~]$ ssh -i app.pem ec2-user@10.10.12.120
Warning: Permanently added '10.10.12.120' (ED25519) to the list of known hosts.

Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

ec2-user@ip-10-10-12-120 ~]$ 

i-05164e6c9de9298ed (web_instance)
PublicIPs: 54.87.12.68 PrivateIPs: 10.10.13.99
CloudShell Feedback Console Mobile App

```

Connected db instance ( production network) through web Instance

```

Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

Warning: Permanently added '10.10.12.120' (ED25519) to the list of known hosts.

Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

ec2-user@ip-10-10-12-120 ~]$ vi ram.pem
ec2-user@ip-10-10-12-120 ~]$ chmod 400 ram.pem
ec2-user@ip-10-10-12-120 ~]$ ssh -i ram.pem ec2-user@10.12.10.151
Warning: Permanently added '10.12.10.151' (ED25519) to the list of known hosts.

Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

ec2-user@ip-10-10-12-10-151 ~]$ 

i-05164e6c9de9298ed (web_instance)
PublicIPs: 54.87.12.68 PrivateIPs: 10.10.13.99
CloudShell Feedback Console Mobile App

```

dbinstance (production network) connected to db instance in development network by peering connection between two vpc's