

Assignment - VPC

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.

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.

Assignment - VPC

Production Network

Design and build a 4-tier architecture

Step 1: Login to account, and Go to VPC, click on “Create VPC”

The screenshot shows the AWS VPC Dashboard. At the top, there are buttons for "Create VPC" and "Launch EC2 Instances". Below this, a section titled "Resources by Region" lists various Amazon VPC resources across the Tokyo region. The resources include:

- VPCs: 1 item
- Subnets: 3 items
- Route Tables: 1 item
- Internet Gateways: 1 item
- Egress-only Internet Gateways: 0 items
- DHCP option sets: 1 item
- Security Groups: 1 item
- Customer Gateways: 0 items
- Virtual Private Gateways: 0 items
- Site-to-Site VPN Connections: 0 items

On the left sidebar, there are sections for "Virtual private cloud", "Security", "PrivateLink and Lattice", and "AWS Network Manager". On the right side, there are sections for "Service Health", "Settings", "Additional Information", and "AWS Network Manager".

Step 2: Give a name and IPv4 CIDR block: 10.0.0.0/16. Tenancy: default.

The screenshot shows the "Create VPC" configuration page. The "VPC settings" section includes:

- Resources to create: VPC only
- Name tag: prod-vpc
- IPv4 CIDR block: 10.0.0.0/16
- IPv6 CIDR block: No IPv6 CIDR block selected
- Tenancy: Default

The "VPC encryption control" section shows:

- None selected
- Monitor mode: See which resources in your VPC are unencrypted but allow the creation of unencrypted resources.
- Enforce mode: Requires all resources, except exclusions, in your VPC to be encryption-capable and blocks creation of unencrypted resources.

Assignment - VPC

VPC only VPC and more

Name tag - optional
Creates a tag with a key of 'Name' and a value that you specify.
prod-vpc

IPv4 CIDR block - info
IPV4 CIDR manual input
IPAM-allocated IPv4 CIDR block

IPv4 CIDR
10.0.0.0/16
CIDR block size must be between /16 and /28.

IPv6 CIDR block - info
No IPv6 CIDR block
IPAM-allocated IPv6 CIDR block
Amazon-provided IPv6 CIDR block
IPv6 CIDR owned by me

Tenancy - info
Default

VPC encryption control (\$)- new - info
Monitor mode provides visibility into encryption status without blocking traffic. Enforce mode prevents unencrypted traffic. Additional charges apply.

None Monitor mode Enforce mode

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 prod-vpc

Add tag

Cancel Preview code Create VPC

You successfully created vpc-03f56fae719561657 / prod-vpc

vpc-03f56fae719561657 / prod-vpc

Details - info

VPC ID: vpc-03f56fae719561657
State: Available
Tenancy: default
Main network ACL: ad-01246668b58303252
IPv6 CIDR (Network border group): -
Encryption control ID: -

Block Public Access: Off
DHCP option set: dopt-06802192b467dfc75
IPv4 CIDR: 10.0.0.0/16
Network Address Usage metrics: Disabled
Encryption control mode: -

DNS hostnames: Disabled
Main route table: rtb-0a416431a011d5a34
IPv6 pool: -
Owner ID: 256716302785

Resource map - info

VPC: Your AWS virtual network
Subnets (0): Subnets within this VPC
Route tables (1): Route network traffic to resources
Network Connections (0): Connections to other networks

VPC has been created

Create 5 subnets out of which 4 should be private named app1, app2, dbcache and db and one should be public, named web.

Step 1: Go to subnets as shown in below picture

Your VPCs

VPCs VPC encryption controls - new

Your VPCs (1) - info

Name	VPC ID	State	Encryption c...	Encryption control ...	Block Public...	IPv4 CIDR	IPv6 CIDR	DHCP option set
prod-vpc	vpc-03f56fae719561657	Available	-	-	Off	10.0.0.0/16	-	dopt-06802192b467dfc75

Step 2: Click on “Create Subnet”

Assignment - VPC

The screenshot shows the AWS VPC Subnets page. On the left, there's a sidebar with 'VPC dashboard' and 'Virtual private cloud' sections. The main area displays a table titled 'Subnets (3) Info' with columns: Name, Subnet ID, State, VPC, Block Public..., and IPv4 CIDR. Three subnets are listed: 'subnet-0d7164499227e0cae' (Available, vpc-05f7c3c2c03230712, prod..., Off, 172.31.16.0/20), 'subnet-0668899f359161d7' (Available, vpc-05f7c3c2c03230712, prod..., Off, 172.31.32.0/20), and 'subnet-04bb3bf341fa23797' (Available, vpc-05f7c3c2c03230712, prod..., Off, 172.31.0.0/20). At the top right, there's a 'Create subnet' button highlighted with a red box.

Step 4: Create 5 subnets in prod-vpc each in a different AZ where possible:

- **web (public): 10.0.0.0/24 (AZ-a)**
- **app1 (private with internet): 10.0.1.0/24 (AZ-a or b)**
- **app2 (private): 10.0.2.0/24 (AZ-b)**
- **dbcache (private with internet access for *that* instance): 10.0.3.0/24 (AZ-c)**
- **db (private): 10.0.4.0/24 (AZ-c)**

Choose AZs to distribute availability.

The screenshot shows the 'Create subnet' wizard. Step 1: Set subnet settings. It asks for a subnet name ('web'), an availability zone ('Asia Pacific (Tokyo) / apnne1-az4 (ap-northeast-1a)'), and an IPv4 VPC CIDR block ('10.0.0.0/16'). The VPC dropdown shows 'vpc-03f56ae719561657 (prod-vpc)'.

Assignment - VPC

The screenshot shows the AWS VPC Subnets creation interface. A new subnet is being created with the following details:

- IPv4 CIDRs:** 10.0.0.0/16
- Subnet name:** web
- Availability Zone:** Asia Pacific (Tokyo) / ap-northeast-1a (ap-northeast-1a)
- IPv4 VPC CIDR block:** 10.0.0.0/16
- IPv4 subnet CIDR block:** 10.0.0.0/20 (4,096 IPs)
- Tags - optional:** A single tag 'Name' is added with value 'web'.

At the bottom right, there are 'Cancel' and 'Create subnet' buttons.

Step 5: Created 5 different subnets

- web (already) = 10.0.0.0/20 (covers 10.0.0.0 - 10.0.15.255)
- app1 = 10.0.16.0/24
- app2 = 10.0.17.0/24 ← use this one
- dbcache = 10.0.18.0/24
- db = 10.0.19.0/24

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR	IPv6 CIDR	IPv6 CIDR association ID	Available
web	subnet-05199ed4528456d045	Available	vpc-0ce7a172543cc0d47 prod..	Off	10.0.0.0/24	-	-	251
app1	subnet-029920d12716ed6d5	Available	vpc-0ce7a172543cc0d47 prod..	Off	10.0.1.0/24	-	-	251
app2	subnet-009f96754bc132053	Available	vpc-0ce7a172543cc0d47 prod..	Off	10.0.2.0/24	-	-	251
dbcache	subnet-0150aae1beab0fb7	Available	vpc-0ce7a172543cc0d47 prod..	Off	10.0.3.0/24	-	-	251
db	subnet-0b0e430a4de0920f4	Available	vpc-0ce7a172543cc0d47 prod..	Off	10.0.4.0/24	-	-	251

Create an Internet Gateway and attach it

Step 1: Click on Internet gateways

Name	Internet gateway ID	State	VPC ID	Owner
-	igw-0d5620ba2b4a100f	Attached	vpc-05f7c5c2c03230712	256716302785

Step 2: Give a name

Assignment - VPC

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
Creates a tag with a key of 'Name' and a value that you specify.
prod-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** [Remove](#)

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

[Cancel](#) [Create internet gateway](#)

Step 3: Attach prod-igw to prod-vpc.

The following internet gateway was created: igw-0e6cb1cbedaab9f60 - prod-igw. You can now attach to a VPC to enable the VPC to communicate with the internet.

Actions [Create internet gateway](#)

[View details](#) [Attach to VPC](#) [Detach from VPC](#) [Manage tags](#) [Delete internet gateway](#)

Internet gateways (1/2) [Info](#)

Name	Internet gateway ID	State	VPC ID	Owner
igw-0d5620ba2b4a100ff	Attached	ycp-05f7c3c2c05230712	2567165	
prod-igw	Detached	-	2567165	
igw-0e6cb1cbedaab9f60				

The following internet gateway was created: igw-0e6cb1cbedaab9f60 - prod-igw. You can now attach to a VPC to enable the VPC to communicate with the internet.

[Attach to a VPC](#)

Attach to VPC (igw-0e6cb1cbedaab9f60) [Info](#)

VPC
Attach an internet gateway to a VPC to enable the VPC to communicate with the internet. Specify the VPC to attach below.

Available VPCs
Attach the internet gateway to this VPC.

[AWS Command Line Interface command](#)

[Cancel](#) [Attach internet gateway](#)

Route tables for Prod

Step 1: Click on Route Tables

Last updated 52 minutes ago

Actions [Create route table](#)

Route tables (6) [Info](#)

Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC	Owner ID
-	rth-076e318836563afdf	-	-	Yes	ycp-05f7c3c2c05230712	256716302785
-	rth-0025aaaf1hf7508ae	-	-	Yes	ycp-0b99d4694a8134393	256716302785
prod-vpc-rtb-private2-ap-northeast-1c	rth-005870ab5ec2c3ffd	-	-	No	ycp-0b99d4694a8134393	256716302785
prod-vpc-rtb-public	rth-0296d997385a0bc14	-	-	No	ycp-0b99d4694a8134393	256716302785
prod-vpc-rtb-private1-ap-northeast-1a	rth-0526920b94e3e22fe	-	-	No	ycp-0b99d4694a8134393	256716302785
-	rth-0a16431a011rd3a34	-	-	Yes	ycp-03f56fae719561657 prod...	256716302785

Step 2: Create route table

Assignment - VPC

Name	Route table ID	Explicit subnet assoc...	Main	VPC	Owner ID
rtb-07f50a18e44618c53	-	-	Yes	vpc-008eb801be510f98	256716302785
rt-public	rtb-03aceb1adb44007c	subnet-05199e4528456d...	No	vpc-0ce7a172543cc0d47 prod...	256716302785
-	rtb-07032da1f107b8730	-	Yes	vpc-0ce7a172543cc0d47 prod...	256716302785

Step 3: Associate the web subnet to new route table

Name	Route table ID	Explicit subnet assoc...	Main	VPC	Owner ID
-	rtb-07f50a18e44618c53	-	Yes	vpc-008eb801be510f98	256716302785
rt-public	rtb-03aceb1adb44007c	subnet-05199e4528456d...	No	vpc-0ce7a172543cc0d47 prod...	256716302785
-	rtb-07032da1f107b8730	-	Yes	vpc-0ce7a172543cc0d47 prod...	256716302785

Step 4: Add route 0.0.0.0/0 → prod-igw

Destination	Target	Status	Propagated	Route Origin
10.0.0.0/16	local	Active	No	CreateRouteTable
0.0.0.0/0	igw-0e6cb1cbedaab9f60	Active	No	CreateRoute

Assignment - VPC

The screenshot shows the AWS VPC Route Tables page. A green banner at the top indicates that routes have been updated successfully. The main section displays a route table named "rtb-0c12cf46ff2aea0da / prod-public-rt". It shows the following details:

- Details Info:** Route table ID: rtb-0c12cf46ff2aea0da, Main: No, Owner ID: vpc-03f56fae719561657 | prod-vpc.
- Explicit subnet associations:** subnet-0cf1e3c70ff02e1c9 / web.
- Edge associations:** None.

The "Routes" tab is selected, showing two entries:

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	igw-0e6cb1cbedaab9f60	Active	No	Create Route
10.0.0.0/16	local	Active	No	Create Route Table

Step 5: Create Nat gateway

The screenshot shows the AWS NAT Gateways page. The "Actions" dropdown menu has a "Create NAT gateway" option highlighted. The main table shows no results found.

The screenshot shows the "Select a NAT gateway" configuration page. It includes fields for:

- NAT gateway settings:** Name (optional) is set to "prod-nat-gw".
- Availability mode:** Zonal is selected.
- Subnet:** subnet-05199e4528456d043 (web) is selected.
- Connectivity type:** Public is selected.
- Elastic IP allocation ID:** eipalloc-0bec80884a5199ea9 is assigned.
- Tags:** A tag "prod-nat-gw" is added.

A green banner at the top indicates that an elastic IP address is allocated.

Edit routes

Assignment - VPC

The screenshot shows the 'Edit routes' page for a specific route table. A route is defined from the destination `10.0.0.16` to the target `local`. The status is `Active`, propagation is `No`, and the route origin is `CreateRouteTable`. Another route is shown from `0.0.0.0/0` to a `NAT Gateway` with the ID `nat-05f8851c766bb1d4f`. The status is `Active`, propagation is `No`, and the route origin is `CreateRoute`. There are buttons for `Add route`, `Remove`, `Cancel`, `Preview`, and `Save changes`.

Associate app1 and dbcache to

The screenshot shows the 'Route tables' page with one entry: `prod-private-rt`. This table has two explicit subnet associations: `app1` (CIDR `10.0.1.0/24`) and `dbcache` (CIDR `10.0.3.0/24`). The table is associated with the VPC `vpc-0ce7a172543cc0d47` and has an owner ID of `256716302785`. The table is marked as `Main` and has no edge associations.

Isolated route table is associated with app2 and db

The screenshot shows the 'Route tables' page with one entry: `rt-isolated`. This table has two explicit subnet associations: `app2` (CIDR `10.0.2.0/24`) and `db` (CIDR `10.0.4.0/24`). The table is associated with the VPC `vpc-0ce7a172543cc0d47` and has an owner ID of `256716302785`. The table is marked as `Main` and has no edge associations.

Assignment - VPC

Launch instances in all subnets and name them as per the subnet that they have been launched in

Instance	Subnet	Public IP?	Notes
web-instance	Public	Yes	Internet-facing
app1-instance	Private	No	NAT gateway gives outbound internet
dbcache-instance	Private	No	NAT gateway gives outbound internet
app2-instance	Private	No	Must stay internal
db-instance	Private	No	Never expose DB

Step 1: Create Security groups

The screenshot shows the AWS EC2 Security Groups page. A success message at the top states "Security group (sg-04cc3be2f3d21b8f0 | sgdb) was created successfully". Below this, a table lists nine security groups:

Name	Description	Owner	Inbound rules count
default	default VPC security group	256716302785	1 Permission entry
launch-wizard-1	launch-wizard-1 created 2025-10-07T1...	256716302785	2 Permission entries
launch-wizard-2	launch-wizard-2 created 2025-10-07T1...	256716302785	1 Permission entry
sgapp2	app2 subnet sg	256716302785	3 Permission entries
sgweb	web subnet sg	256716302785	5 Permission entries
sgapp1	app1 subnet sg	256716302785	3 Permission entries
sgdb	db subnet sg	256716302785	0 Permission entries
default	default VPC security group	256716302785	1 Permission entry
sgdbcache	dbcache subnet sg	256716302785	0 Permission entries

Step 2: Create keypairs and instances for all subnets

Web instance

The screenshot shows the "Launch an instance" wizard. The first step, "Name and tags", has "web-instance" entered. The second step, "Application and OS Images (Amazon Machine Image)", shows the "Amazon Linux 2023 kernel-6.1 AMI" selected. The third step, "Summary", shows one instance being launched with the AMI, security group sgweb, and 1 volume (8 GiB). The "Launch instance" button is highlighted.

Assignment - VPC

The screenshot shows the AWS EC2 'Launch an instance' wizard. The first step, 'Instance type', is displayed. It shows the selected AMI (Amazon Linux 2023 kernel-6.1 AMI) and the chosen instance type (t3.micro). The second step, 'Network settings', is shown below. It includes fields for VPC (prod-vpc), subnet (subnet-05199e4528456d043), and security groups (sgweb). The 'Auto-assign public IP' section is also visible.

App1 Instance

The screenshot shows the AWS EC2 'Launch an instance' wizard. The 'Name and tags' step is displayed, where the instance is named 'app1-instance'. The 'Application and OS Images' step follows, showing the selected AMI (Amazon Linux 2023 kernel-6.1 AMI) and the 'Quick Start' tab selected. The 'Recent' section lists various operating systems including Amazon Linux, macOS, Ubuntu, Windows, Red Hat, SUSE Linux, and Debian.

Assignment - VPC

The screenshot shows the 'Launch an instance' wizard in the AWS EC2 console. The 'Network settings' section includes a VPC dropdown set to 'prod-vpc', a subnet dropdown set to 'subnet-02930d1d1716e6cd3', and a security group dropdown set to 'sgapp1'. The 'Configure storage' section shows a 1x 8 GiB gp3 volume. The 'Summary' panel indicates 1 instance will be launched using the 'Amazon Linux 2023 AMI 2023.9.2...' AMI, with a t3.micro virtual server type, sgapp1 firewall, and 1 volume(s) of 8 GiB.

Dbcache instance

This screenshot shows the 'Launch an instance' wizard for a 'dbcache' instance. The 'Network settings' section has a subnet dropdown set to 'subnet-01500aa1b1eaeab0ff'. The 'Configure storage' section shows a 1x 8 GiB gp3 volume. The 'Summary' panel indicates 1 instance will be launched using the 'Amazon Linux 2023 AMI 2023.9.2...' AMI, with a t3.micro virtual server type, sgdbcache firewall, and 1 volume(s) of 8 GiB.

All instances are created

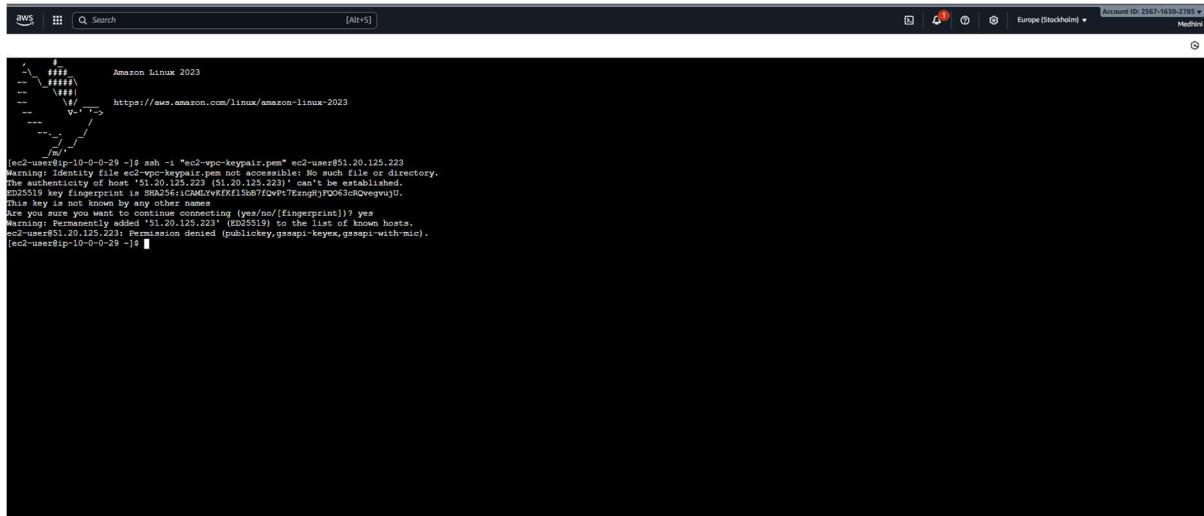
The screenshot shows the EC2 Instances page with a list of five running instances: 'app1-instance', 'web-instance', 'dbcache-instance', 'app2-instance', and 'db-instance'. Each instance is listed with its instance ID, state (Running), type (t3.micro), status check results, availability zone (eu-north-1a or eu-north-1b), and public IP address (e.g., 51.20.125.22).

Result

Instance	Subnet	Public IP	Internet?
web-instance	web	YES	Direct
app1	app1	NO	Through NAT
dbcache	dbcache	NO	Through NAT
app2	app2	NO	X (isolated)
db	db	NO	X (isolated)

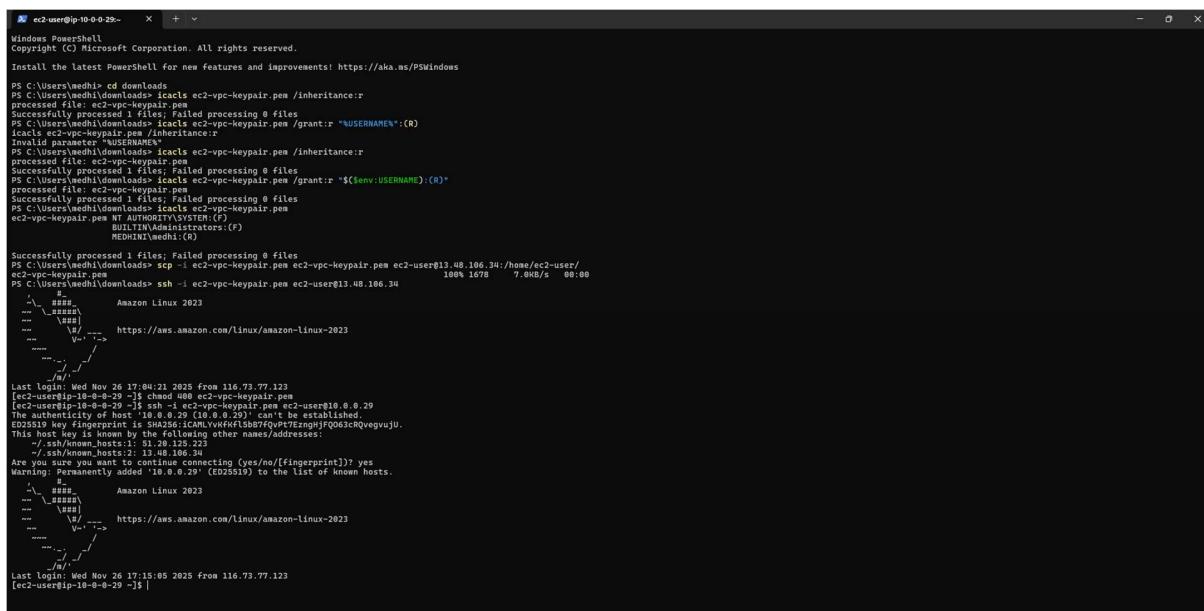
Assignment - VPC

Web-instance



The screenshot shows the AWS Lambda function configuration page. At the top, it displays the function name 'app1' and the runtime 'Python 3.8'. Under the 'Environment' section, there is one environment variable named 'AWS_LAMBDA_FUNCTION_NAME' with the value 'app1'. In the 'Triggers' section, there is one trigger named 'app1' which is a CloudWatch Logs trigger. The log group is '/aws/lambda/app1' and the log stream is 'Log Stream: app1'. The 'Last triggered' time is listed as '2023-11-26T17:04:21Z'.

App1 verification is completed as shown in below picture (*app1 should connect from your laptop using keypair*)



The screenshot shows a Windows PowerShell session. The user runs the command `ssh -i "ec2-vpc-keypair.pem" ec2-user@51.20.125.223`. The terminal output shows the host key fingerprint and asks if the user wants to continue connecting. The user responds with 'yes'. The session then connects to the EC2 instance.

Assignment - VPC

Dbcache has been verified

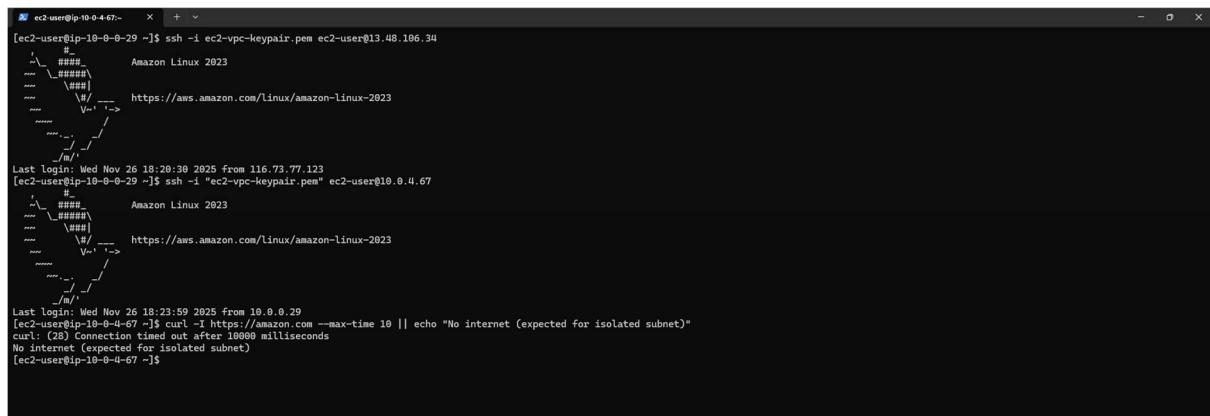
App2 verification completed

```
[ec2-user@ip-10-0-2-130- ~]$ ssh -i ec2-vpc-keypair.pem ec2-user@13.48.106.34
[ec2-user@ip-10-0-0-29- ~]$ ssh -i ec2-vpc-keypair.pem ec2-user@13.48.106.34
[ec2-user@ip-10-0-0-29- ~]$ curl -I https://aws.amazon.com/linux/amazon-linux-2023
Last login: Wed Nov 26 18:10:26 2025 from 13.48.106.34
[ec2-user@ip-10-0-0-29- ~]$ ssh -i ec2-vpc-keypair.pem ec2-user@10.0.2.130
The authenticity of host '10.0.2.130 (10.0.2.130)' can't be established.
ED25519 key fingerprint is SHA256:nL03J6wtJZxOeQ1bPH8pZMk/ekSVLoNk78i9o2Rpw.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.0.2.130' (ED25519) to the List of known hosts.

[ec2-user@ip-10-0-0-29- ~]$ curl -I https://aws.amazon.com/linux/amazon-linux-2023
[ec2-user@ip-10-0-2-130- ~]$ curl -I https://amazon.com --max-time 10 || echo "No internet (expected for isolated subnet)"
^[[Acurl: (28) Connection timed out after 10001 milliseconds
No internet (expected for isolated subnet)
[ec2-user@ip-10-0-2-130- ~]$ curl -I https://amazon.com --max-time 10 || echo "No internet (expected for isolated subnet)"
^[[Acurl: (28) Connection timed out after 10000 milliseconds
No internet (expected for isolated subnet)
[ec2-user@ip-10-0-2-130- ~]$
```

Assignment - VPC

db verification has been completed

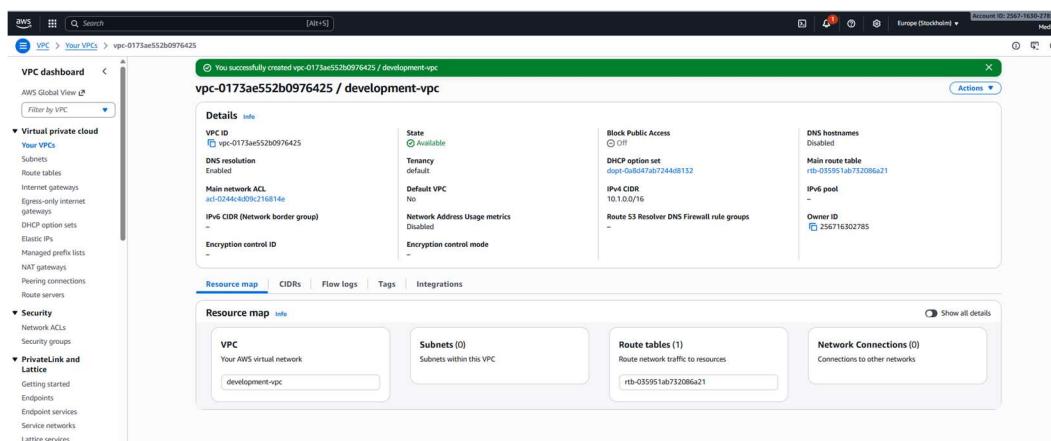


```
[ec2-user@ip-10-0-4-67- ~]$ ssh -i ec2-vpc-keypair.pem ec2-user@13.48.106.34
Last login: Wed Nov 26 18:20:38 2025 from 116.73.77.123
[ec2-user@ip-10-0-0-29- ~]$ ssh -i "ec2-vpc-keypair.pem" ec2-user@10.0.4.67
Last login: Wed Nov 26 18:23:59 2025 from 10.0.0.29
[ec2-user@ip-10-0-4-67- ~]$ curl -I https://amazon.com --max-time 10 || echo "No internet (expected for isolated subnet)"
curl: (28) Connection timed out after 10000 milliseconds
No internet (expected for isolated subnet)
[ec2-user@ip-10-0-4-67- ~]$
```

Development Network

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.

Step 1: Create vpc



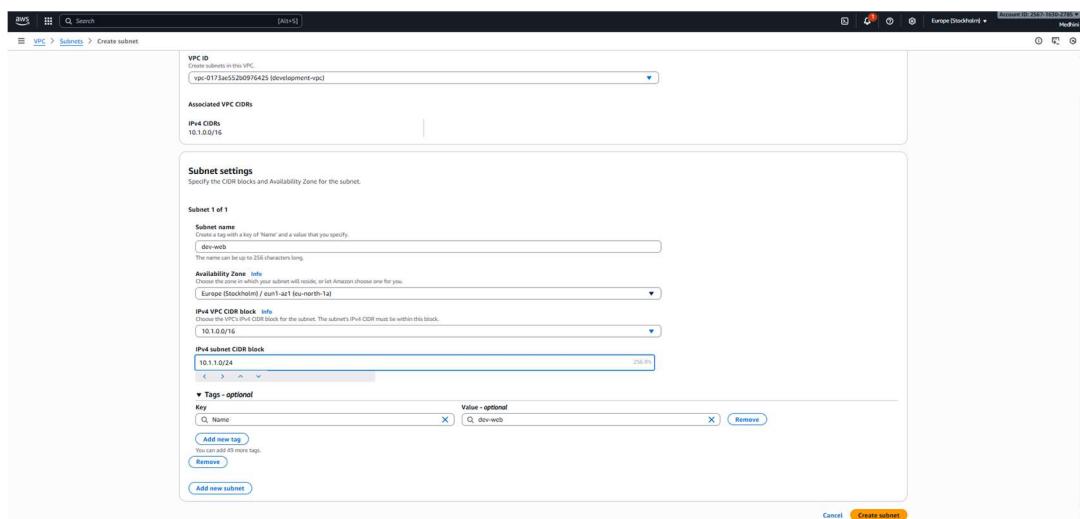
The screenshot shows the AWS VPC dashboard with a success message: "You successfully created vpc-0173ae552b0976425 / development-vpc". The VPC details include:

- VPC ID: vpc-0173ae552b0976425
- State: Available
- Tenancy: default
- Main network ACL: ad-2244c4d9c92716814e
- DNS resolution: Enabled
- Default VPC: No
- IPv6 CDR (Network border group): -
- Network Address Usage metrics: Disabled
- Encryption control ID: -
- Block Public Access: Off
- DHCP option set: dopt-0ab47ab7244db132
- IPv4 CIDR: 10.1.0.0/16
- Route 53 Resolver DNS Firewall rule groups: -
- DNS hostnames: Disabled
- Main route table: rtb-035951ab732086a21
- IPv6 pool: -
- Owner ID: 256716302785

The Resource map section shows:

- VPC: Your AWS virtual network (development-vpc)
- Subnets (0): Subnets within this VPC
- Route tables (1): Route network traffic to resources (rtb-035951ab732086a21)
- Network Connections (0): Connections to other networks

Step 2: Create public subnet and private subnet



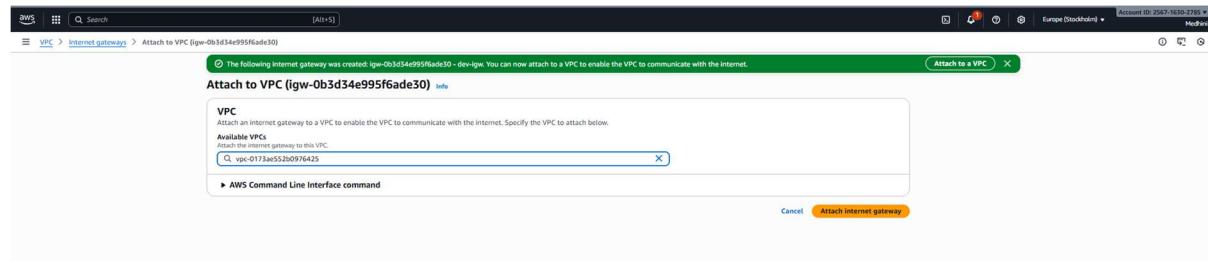
The screenshot shows the "Create subnet" wizard for VPC ID vpc-0173ae552b0976425 (development-vpc). The steps are:

- Associated VPC CDRs: IPv4 CIDR: 10.1.0.0/16
- Subnet settings:
 - Subnet 1 of 1:
 - Subnet name: dev-web (Name up to 255 characters long)
 - Availability Zone: Europe (Stockholm) / eu-north-1a (eu-north-1a)
 - IPv4 VPC CIDR block: 10.1.0.0/16
 - IPv4 subnet CIDR block: 10.1.1.0/24 (255 IPs)
- Tags - optional:
 - Key: Name Value: dev-web

Buttons at the bottom: Cancel, Create subnet.

Assignment - VPC

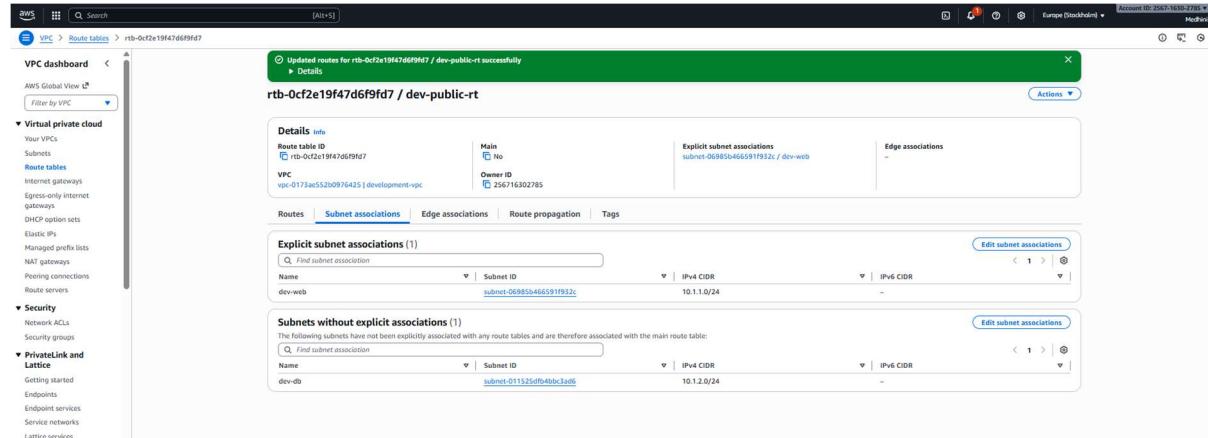
Step 3: Create Internet gateway and Attach it to dev vpc



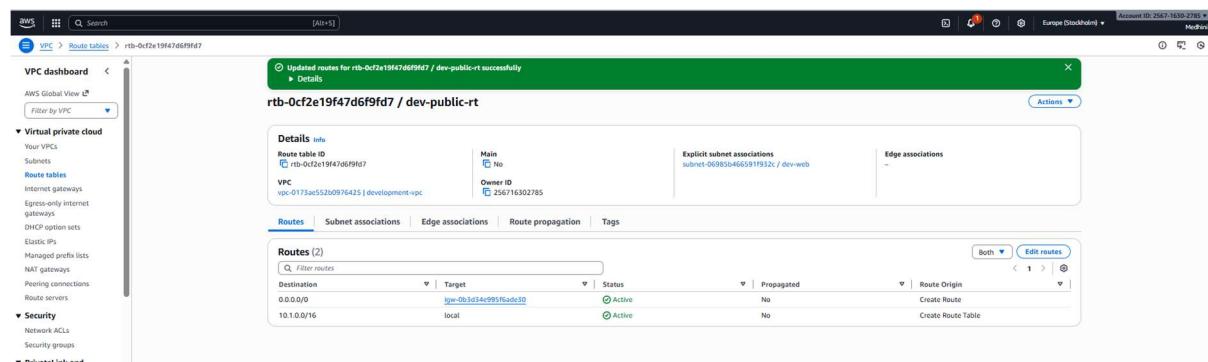
Step 4: Create route table

Public Route Table (for web subnet)

- Name: dev-public-rt
- Associate subnets:
 - dev-web
- Add route:
 - 0.0.0.0/0 → Internet Gateway (dev-igw)



Add inbound rule



Assignment - VPC

Private Route Table (for db subnet)

- Name: dev-private-rt
- Associate subnet:
 - dev-db
- No internet route
 - Only local VPC traffic

The screenshot shows the AWS VPC Route Tables page. The route table ID is rtb-0ef7915151ce23aa1 and it is associated with the VPC vpc-0173aae552b0976425. It has one explicit subnet association to subnet-011525dfb4bc3ad6 (dev-db) with an IPv4 CIDR of 10.1.2.0/24. There are no routes or edge associations.

The screenshot shows the routes section of the route table. It contains one route entry: Destination 10.1.0.0/16, Target local, Status Active, Propagated No, and Route Origin Create Route Table.

Step 4: Launch EC2 Instances

For web subnet

- Subnet: dev-web
- Security Group: allow SSH + HTTP
- Enable Auto-assign Public IP
- Name: dev-web-instance

Assignment - VPC

The screenshot shows three sequential steps in the AWS EC2 "Launch an instance" wizard:

Step 1: Launch an instance

This step allows you to create a new instance. It includes fields for "Name and tags", "Application and OS Images (Amazon Machine Image)", and "Amazon Machine Image (AMI)". A summary panel on the right shows the configuration: 1 instance of t3.micro (Amazon Linux 2023 AMI), 1 volume (8 GiB), and a New security group.

Step 2: Instance type

This step specifies the instance type as t3.micro. It also asks for a key pair (ec2-vpc-keypair) and sets up network settings, including a VPC (development-vpc) and a subnet (subnet-06985b466591f932c). The summary panel remains the same.

Step 3: Inbound Security Group Rules

This step configures security group rules. It adds two rules: one for SSH (TCP, port 22) from Anywhere and one for HTTP (TCP, port 80) from Anywhere. A warning message notes that rules with 0.0.0.0/0 allow all IP addresses. The summary panel remains the same.

Assignment - VPC

For db subnet

- Subnet: dev-db
- No Public IP
- Security Group:
 - Allow MySQL/Postgres from Production-DB subnet only
- Name: dev-db-instance

The screenshot shows the AWS EC2 Instances page. The instance summary for 'i-02b9f9bbef24d0cb1 (dev-db-instance)' is displayed. Key details include:

- Instance ID: i-02b9f9bbef24d0cb1
- Private IPv4 address: 10.1.2.117
- Instance state: Running
- Instance type: t3.micro
- VPC ID: vpc-0173ae552b0976425 (development-vpc)
- Subnet ID: subnet-011525dfb4bbc3ad6 (dev-db)
- Instance ARN: arn:aws:ec2:eu-north-1:256716302785:instance/i-02b9f9bbef24d0cb1

STEP 6 : VPC Peering (Prod ↔ Dev)

On the AWS console → VPC Peering

1. Choose Production VPC as requester
→ production-vpc
2. Choose Development VPC as accepter
→ development-vpc
3. Accept the request in the Development VPC

Click on VPC Peering connections

The screenshot shows the AWS VPC dashboard. The 'VPC Peering Connections' section is highlighted with a red box. It lists 1 connection:

Requester VPC	Accepter VPC	Status
production-vpc	development-vpc	Pending acceptance

Assignment - VPC

The screenshot shows the AWS VPC Peering connections dashboard. At the top right, there is an orange button labeled "Create peering connection". Below it, a modal window titled "Create peering connection" is open. In the "Peering connection settings" section, the "Name" field contains "production-vpc". Under "Select a local VPC to peer with", the "VPC ID (Requester)" dropdown is set to "vpc-0ce7a172543cc0d47 (prod-vpc)". The "VPC CIDRs for vpc-0ce7a172543cc0d47 (prod-vpc)" table shows one entry: "CIDR: 10.0.0.0/16, Status: Associated, Status reason: -". In the "Select another VPC to peer with" section, the "Region" dropdown is set to "This Region (eu-north-1)". The "VPC ID (Accepter)" dropdown is set to "vpc-0173ae552b0976425 (development-vpc)". The "VPC CIDRs for vpc-0173ae552b0976425 (development-vpc)" table shows one entry: "CIDR: 10.1.0.0/16, Status: Associated, Status reason: -". The "Tags" section contains a single tag: "Key: Name, Value: production-vpc". At the bottom right of the modal is a blue "Create peering connection" button.

Accept the Request

The screenshot shows the AWS VPC Peering connections details page for a specific peering connection. A green banner at the top states: "A VPC peering connection pxc-0b85f2a15f11d073d / production-vpc has been requested." Below this, a "Pending acceptance" message indicates that acceptance or rejection must be done by December 4, 2025. The "Actions" menu on the right includes options like "Accept request", "Reject request", "Edit times settings", and "Manage tags". The "Details" section provides information about the requester and accepter, including their VPC IDs, CIDRs, and regions. The "VPC Peering connection ARN" is also listed.

Connect has been established once you accept the request

Assignment - VPC

Your VPC peering connection (pcx-0b85f2a15f11d073d | production-vpc) has been established.

To send and receive traffic across this VPC peering connection, you must add a route to the peered VPC in one or more of your VPC route tables. [View routes](#)

Actions

Details

Requester owner ID	256716302785	Acceptor owner ID	256716302785
Peering connection ID	pcx-0b85f2a15f11d073d	Requester VPC	vpc-0ce7a172543cc0d47 / prod-vpc
Status	Active	Requester CIDR	10.0.0.0/16
Expiration time	-	Requester Region	Stockholm (eu-north-1)
		VPC Peering connection ARN	arn:aws:vpc:eu-north-1:256716302785:vpc-peering-connection/pcx-0b85f2a15f11d073d
		Acceptor VPC	vpc-0173aae552b0976425 / development-vpc
		Requester CIDR	10.1.0.0/16
		Acceptor Region	Stockholm (eu-north-1)

DNS **Route tables** **Tags**

DNS settings

Requester VPC ([vpc-0ce7a172543cc0d47 / prod-vpc](#)) [Info](#)

Allow accepter VPC to resolve DNS of hosts in requester VPC to private IP addresses

Disabled

Acceptor VPC ([vpc-0173aae552b0976425 / development-vpc](#)) [Info](#)

Allow requester VPC to resolve DNS of hosts in accepter VPC to private IP addresses

Disabled

[Edit DNS settings](#)

Set the route table

Edit routes

Destination	Target	Status	Propagated	Route Origin	
10.1.0.0/16	local	Active	No	CreateRouteTable	
Q 10.0.0.0/16	Peering Connection	Active	No	CreateRoute	Remove
Q pcx-0b85f2a15f11d073d					
Q 0.0.0.0/0	Internet Gateway	Active	No	CreateRoute	Remove
Q igw-0b3d34e995f6ade30					

[Add route](#)

[Cancel](#) [Preview](#) [Save changes](#)

Route tables (1/8) Info

Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC
rt-public	rtb-03aaceb1adb44007c	subnet-05199e4528456d043 / web	-	No	vpc-0ce7a172543cc0d47 prod-vpc
-	rtb-07f50a1be4618c63	-	-	Yes	vpc-008eb801be510f98
prod-private-rt	rtb-065000171500ac422	2 subnets	-	No	vpc-0ce7a172543cc0d47 prod-vpc
rt-isolated	rtb-0c7b1fb284a932b76	2 subnets	-	No	vpc-0ce7a172543cc0d47 prod-vpc
dev-private-rt	rtb-0ef7915151ce23aa1	subnet-011525dfb4bbc3ad6 / dev-db	-	No	vpc-0173aae552b0976425 development-vpc
-	rtb-055951ab732086g21	-	-	Yes	vpc-0173aae552b0976425 development-vpc
-	rtb-07032da1f107b8730	-	-	Yes	vpc-0ce7a172543cc0d47 prod-vpc
dev-public-rt	rtb-0cf2e19f47d6f9fd7	subnet-06985b466591f932c / dev-web	-	No	vpc-0173aae552b0976425 development-vpc

rtb-0ef7915151ce23aa1 / dev-private-rt

Routes (2)

Destination	Target	Status	Propagated
10.0.0.0/16	pcx-0b85f2a15f11d073d	Active	No
10.1.0.0/16	local	Active	No

Assignment - VPC

VPC dashboard

Route tables (1/8) info

Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC
<input checked="" type="checkbox"/> rt-public	rtb-03aaceb1adb44007c	subnet-05199e4528456d045 / web	-	No	vpc-0ce7a172543cc0d47 prod-vpc
<input type="checkbox"/> -	rtb-07f50a1be44618c63	-	-	Yes	vpc-008eb8b01be510f98
<input type="checkbox"/> prod-private-rt	rtb-065000171500ac422	2 subnets	-	No	vpc-0ce7a172543cc0d47 prod-vpc
<input type="checkbox"/> rt-isolated	rtb-0c7b1fb284a932b76	2 subnets	-	No	vpc-0ce7a172543cc0d47 prod-vpc
<input type="checkbox"/> dev-private-rt	rtb-0ef7915151ce23aa1	subnet-011525dfb4bbc3ad6 / dev-db	-	No	vpc-0173ae552b0976425 development-vpc
<input type="checkbox"/> -	rtb-035951ab732086a21	-	-	Yes	vpc-0173ae552b0976425 development-vpc
<input type="checkbox"/> -	rtb-07032da1f10708730	-	-	Yes	vpc-0ce7a172543cc0d47 prod-vpc
<input type="checkbox"/> dev-public-rt	rtb-0cf2e19f47d6f9fd7	subnet-06985b466591f932c / dev-web	-	No	vpc-0173ae552b0976425 development-vpc

Last updated 1 minute ago Actions Create route table

rtb-03aaceb1adb44007c / rt-public

Routes (3)

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	igw-de7037cb2f9748c5d	Active	No	Create Route
10.0.0.0/16	local	Active	No	Create Route Table
10.1.0.0/16	pcx-0b85f2a15f11d073d	Active	No	Create Route

VPC dashboard

Route tables (1/8) info

Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC
<input type="checkbox"/> rt-public	rtb-03aaceb1adb44007c	subnet-05199e4528456d045 / web	-	No	vpc-0ce7a172543cc0d47 prod-vpc
<input type="checkbox"/> -	rtb-07f50a1be44618c63	-	-	Yes	vpc-008eb8b01be510f98
<input checked="" type="checkbox"/> prod-private-rt	rtb-065000171500ac422	2 subnets	-	No	vpc-0ce7a172543cc0d47 prod-vpc
<input type="checkbox"/> rt-isolated	rtb-0c7b1fb284a932b76	2 subnets	-	No	vpc-0ce7a172543cc0d47 prod-vpc
<input type="checkbox"/> dev-private-rt	rtb-0ef7915151ce23aa1	subnet-011525dfb4bbc3ad6 / dev-db	-	No	vpc-0173ae552b0976425 development-vpc
<input type="checkbox"/> -	rtb-035951ab732086a21	-	-	Yes	vpc-0173ae552b0976425 development-vpc
<input type="checkbox"/> -	rtb-07032da1f10708730	-	-	Yes	vpc-0ce7a172543cc0d47 prod-vpc
<input type="checkbox"/> dev-public-rt	rtb-0cf2e19f47d6f9fd7	subnet-06985b466591f932c / dev-web	-	No	vpc-0173ae552b0976425 development-vpc

Last updated 2 minutes ago Actions Create route table

rtb-065000171500ac422 / prod-private-rt

Routes (3)

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	nat-05f8851c766bb1d4f	Active	No	Create Route
10.0.0.0/16	local	Active	No	Create Route Table
10.1.0.0/16	pcx-0b85f2a15f11d073d	Active	No	Create Route

VPC dashboard

Route tables (1/8) info

Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC
<input type="checkbox"/> rt-public	rtb-03aaceb1adb44007c	subnet-05199e4528456d045 / web	-	No	vpc-0ce7a172543cc0d47 prod-vpc
<input type="checkbox"/> -	rtb-07f50a1be44618c63	-	-	Yes	vpc-008eb8b01be510f98
<input type="checkbox"/> prod-private-rt	rtb-065000171500ac422	2 subnets	-	No	vpc-0ce7a172543cc0d47 prod-vpc
<input checked="" type="checkbox"/> rt-isolated	rtb-0c7b1fb284a932b76	2 subnets	-	No	vpc-0ce7a172543cc0d47 prod-vpc
<input type="checkbox"/> dev-private-rt	rtb-0ef7915151ce23aa1	subnet-011525dfb4bbc3ad6 / dev-db	-	No	vpc-0173ae552b0976425 development-vpc
<input type="checkbox"/> -	rtb-035951ab732086a21	-	-	Yes	vpc-0173ae552b0976425 development-vpc
<input type="checkbox"/> -	rtb-07032da1f10708730	-	-	Yes	vpc-0ce7a172543cc0d47 prod-vpc
<input type="checkbox"/> dev-public-rt	rtb-0cf2e19f47d6f9fd7	subnet-06985b466591f932c / dev-web	-	No	vpc-0173ae552b0976425 development-vpc

Last updated 3 minutes ago Actions Create route table

rtb-0c7b1fb284a932b76 / rt-isolated

Routes (2)

Destination	Target	Status	Propagated	Route Origin
10.0.0.0/16	local	Active	No	Create Route Table
10.1.0.0/16	pcx-0b85f2a15f11d073d	Active	No	Create Route

Assignment - VPC

Edit inbound rules for both production sg and development sg

Inbound security group rules successfully modified on security group (sg-04cc3be2f3d21b8f0) sgdb

Details

Security Groups (1/12) Info

Find security groups by attribute or tag

Name	Security group ID	Security group name	VPC ID	Description
sg-04cc3be2f3d21b8f0	sgdb	default	vpc-0ce7a172545cc0047	db subnet sg
sg-0459a2f610b82c15ca		default	vpc-0173aae552b0976425	default VPC security group
sg-0286da42fb6540dec4		launch-wizard-3	vpc-0173aae552b0976425	launch-wizard-3 created 2025-11-26T1...
sg-0cc4d1454c430af7	sgweb	default	vpc-0ce7a172545cc0047	web subnet sg
sg-0996aa0a9b15d6a61		default	vpc-008eb8b016e510f98	default VPC security group
sg-0cb3a199df5a2c79	sgapp1	default	vpc-0ce7a172545cc0047	app1 subnet sg
sg-0dfdf26dad41d89b4	launch-wizard-4	default	vpc-0173aae552b0976425	launch-wizard-4 created 2025-11-26T1...

Actions Export security groups to CSV Create security group

sg-04cc3be2f3d21b8f0 - sgdb

Inbound rules (2)

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source
sgr-04348154ada6cd03e	IPv4	SSH	TCP	22	10.0.0.29/32	
sgr-0b45a9d2249bae5e	-	MySQL/Aurora	TCP	3306	sg-04cc3be2f3d21b8f0	

Run Instances

The screenshot shows the AWS CloudWatch Metrics Insights interface. A search bar at the top contains the query: "aws.ec2.instances[*]". Below the search bar, a table titled "Instances (1/7) info" lists seven EC2 instances. The columns include Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, and Public IPv4 DNS. One instance, "dev-web-instance", is selected, highlighted with a blue border. A red box highlights the "Connect" button in the top right corner of the table header. The bottom section shows detailed information for the selected instance, including its name, state, and metrics.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
app1-test-instance	i-004224730245af65	Running	t3.micro	3/3 checks pass	View alarms +	eu-north-1a	-
web-instance	i-08a74f825e05fead	Running	t3.micro	3/3 checks pass	View alarms +	eu-north-1a	-
dev-web-instance	i-09ff447735c97536	Running	t3.micro	3/3 checks pass	View alarms +	eu-north-1a	-
dbcache-instance	i-0c1121h02366d11	Running	t3.micro	3/3 checks pass	View alarms +	eu-north-1b	-
app2-instance	i-005d36523317e3fd	Running	t3.micro	3/3 checks pass	View alarms +	eu-north-1b	-
db-instance	i-087348cc4e3dbad5d	Running	t3.micro	3/3 checks pass	View alarms +	eu-north-1b	-
dev-db-instance	i-02b9f9bbef240c0b1	Running	t3.micro	3/3 checks pass	View alarms +	eu-north-1b	-

Logged into **dev-web** EC2 (Amazon Linux)

Verified it has **Internet access** (by pinging google.com)

Assignment - VPC

Verified the dev and prod instances

```
Last login: Thu Nov 27 05:01:59 2025 from 116.73.77.123
[ec2-user@ip-10-1-1-221 ~]$ chmod 400 ~/keys/ec2-vpc-keypair.pem
[ec2-user@ip-10-1-1-221 ~]$ ssh -i ~/keys/ec2-vpc-keypair.pem ec2-
user@10.1.2.117
ssh -v -i ~/keys/ec2-vpc-keypair.pem ec2-user@10.1.2.117^Z
[1]+  Stopped                  ssh -i ~/keys/ec2-vpc-keypair.pem ec2-
user@10.1.2.117
[ec2-user@ip-10-1-1-221 ~]$ ping 10.1.2.117
PING 10.1.2.117 (10.1.2.117) 56(84) bytes of data.
^C
--- 10.1.2.117 ping statistics ---
68 packets transmitted, 0 received, 100% packet loss, time 69705ms

[ec2-user@ip-10-1-1-221 ~]$ ssh -i ~/keys/ec2-vpc-keypair.pem ec2-
user@10.1.2.117
, #_
~\_\ ####_     Amazon Linux 2023
~~ \#####\
~~ \|##|
~~  \#/ __ https://aws.amazon.com/linux/amazon-linux-2023
~~   \V~'`->
~~~   /
~~_. _/
  _/
 _/m/'
```

Last login: Wed Nov 26 21:06:42 2025 from 10.1.1.221
[ec2-user@ip-10-1-2-117 ~]\$

This confirms:

- SSH connection from bastion - private EC2 works.
- The network (SGs, ACLs, routing) between bastion and private EC2 is correctly configured.