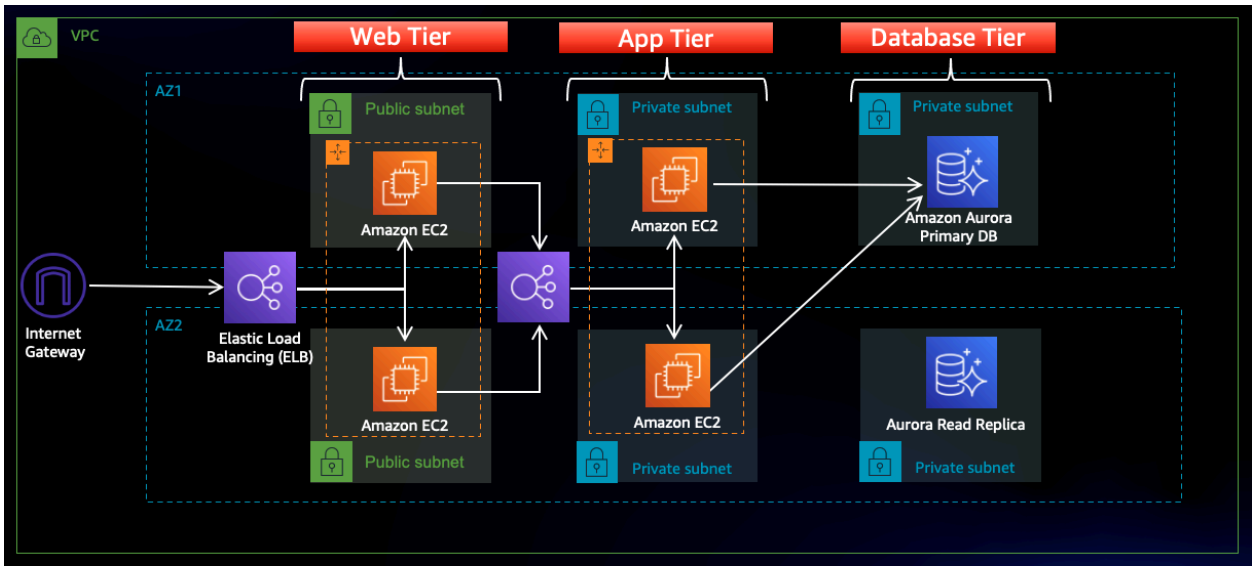
Design and implement a three-tier web application architecture on AWS

Architecture Diagram for AWS :



1. Presentation Tier

>>> Application Load Balancer (ALB) in public subnets

>>> Routes internet traffic to web tier

2. Application Tier

>>> EC2 Instances in private subnets

>>> Auto Scaling Group for scaling

>>> Instances serve backend logic

3. Data Tier

>>> RDS (could be MySQL, Aurora) in isolated/private subnets

>>> Multi-AZ support for high availability

STEP 1 : Cloning GitHub Repo

The application source code was cloned from GitHub to understand the backend and frontend structure used in the deployment.

Git repo Link :

https://github.com/aws-samples/aws-three-tier-web-architecture-workshop.git

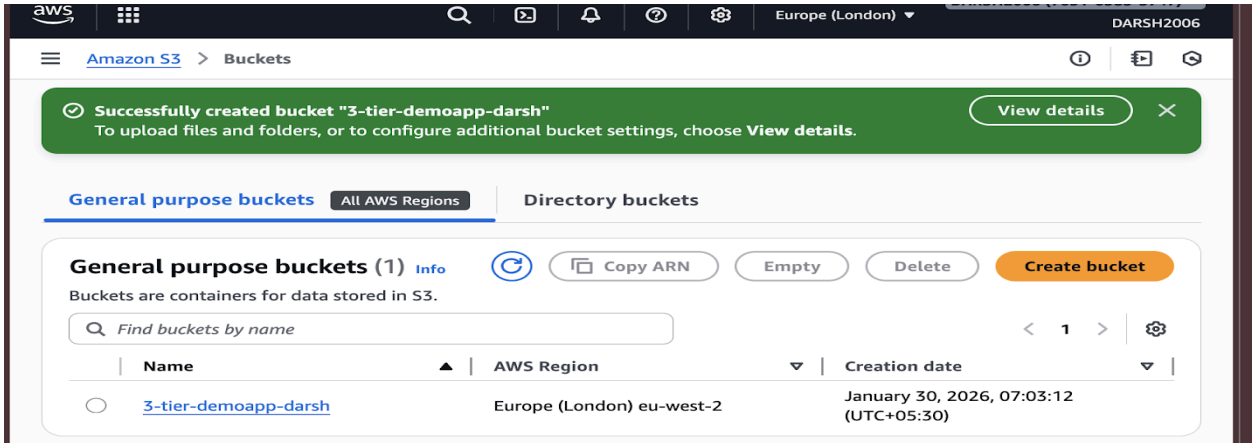
Commands :

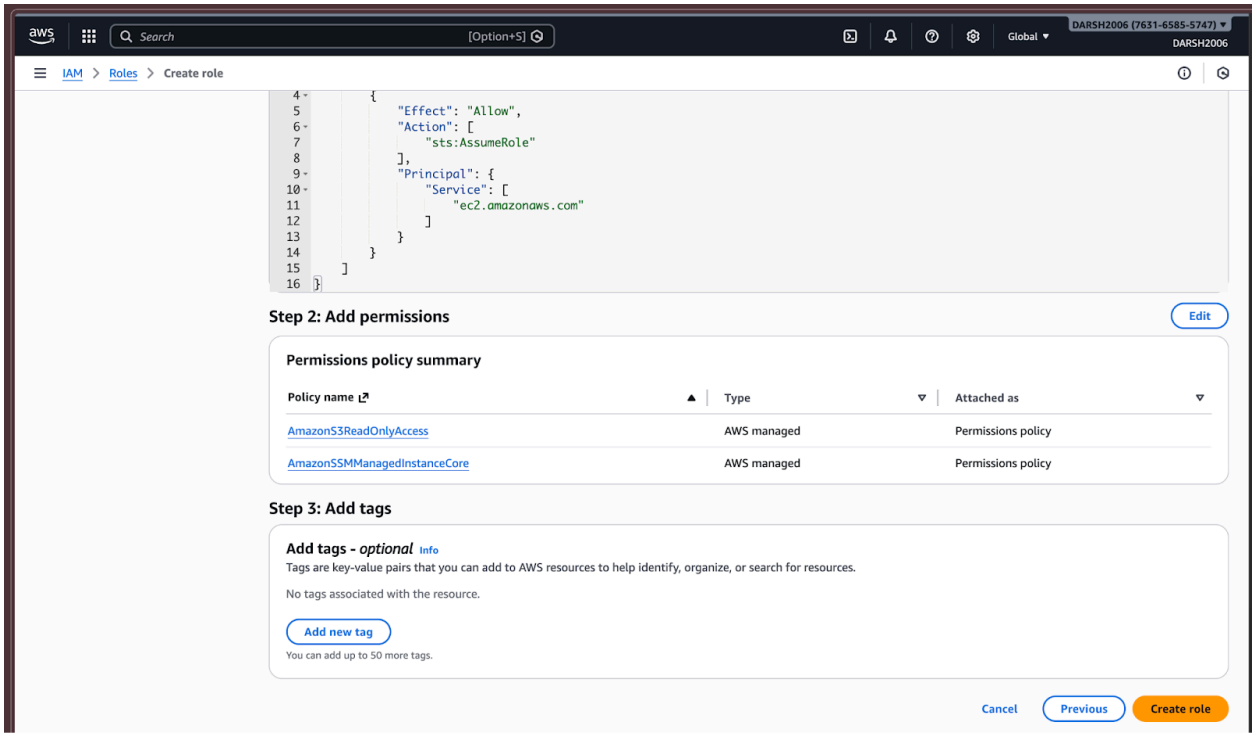
>> mkdir c1
>> cd c1
>> git clone
https://github.com/aws-samples/aws-three-tier-web-architecture-workshop.git
>> ls

```
Last login: Fri Jan 30 06:56:45 on ttys000
darshana@Darshanas-MacBook-Air ~ % mkdir c1
darshana@Darshanas-MacBook-Air ~ % cd c1
darshana@Darshanas-MacBook-Air c1 % git clone https://github.com/aws-samples/aws-three-tier-web-architecture-workshop.git
Cloning into 'aws-three-tier-web-architecture-workshop'...
remote: Enumerating objects: 133, done.
remote: Counting objects: 100% (60/60), done.
remote: Compressing objects: 100% (34/34), done.
remote: Total 133 (delta 36), reused 26 (delta 26), pack-reused 73 (from 1)
Receiving objects: 100% (133/133), 226.99 KiB | 1.06 MiB/s, done.
Resolving deltas: 100% (53/53), done.
darshana@Darshanas-MacBook-Air c1 % ls
aws-three-tier-web-architecture-workshop
darshana@Darshanas-MacBook-Air c1 %
```

STEP 2 : Creating S3 Bucket and IAM Role

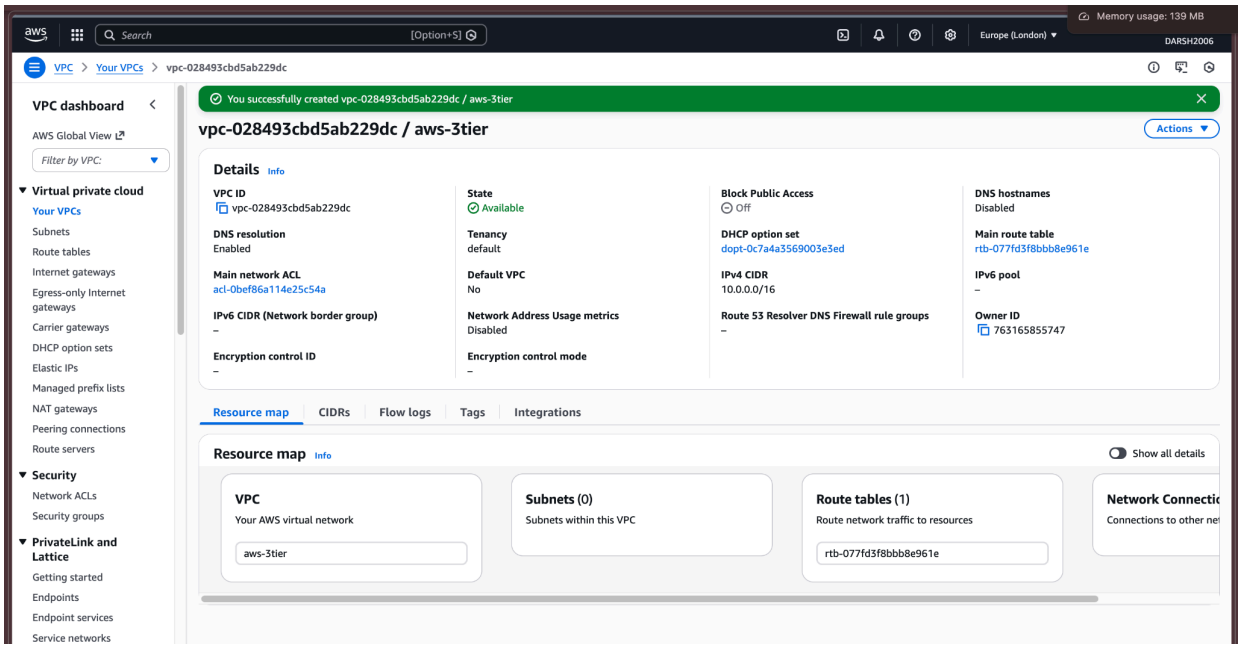
An S3 bucket was created to store application files, and an IAM role was configured to allow EC2 instances secure access to S3.

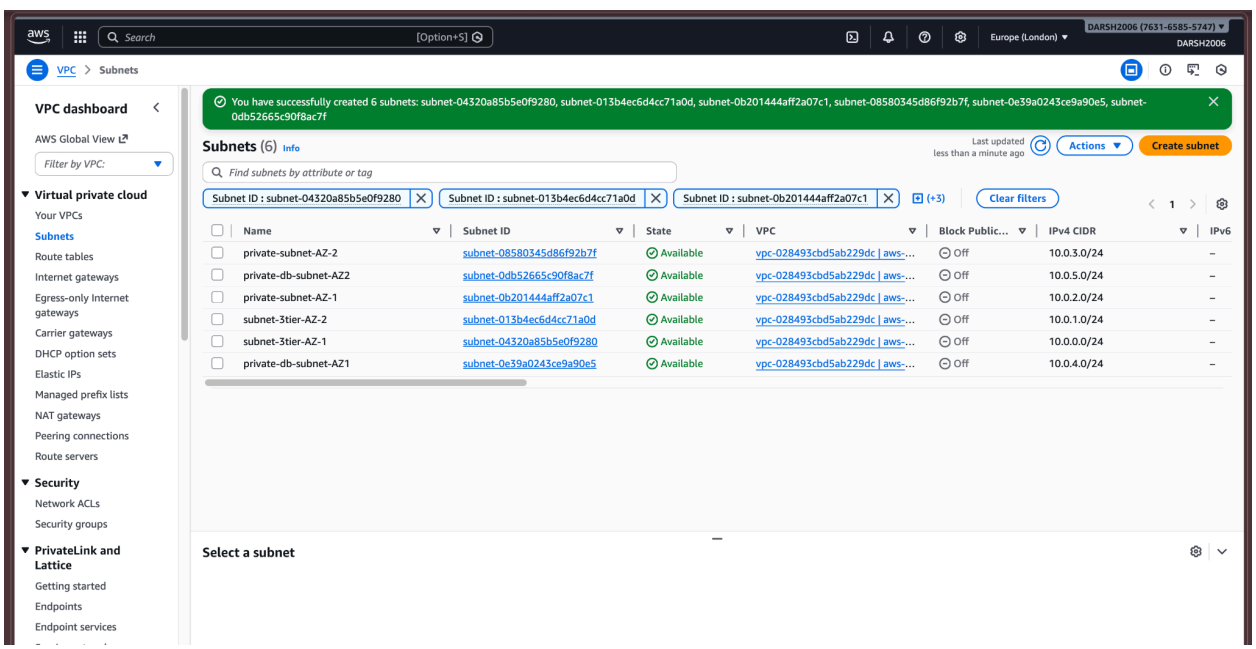


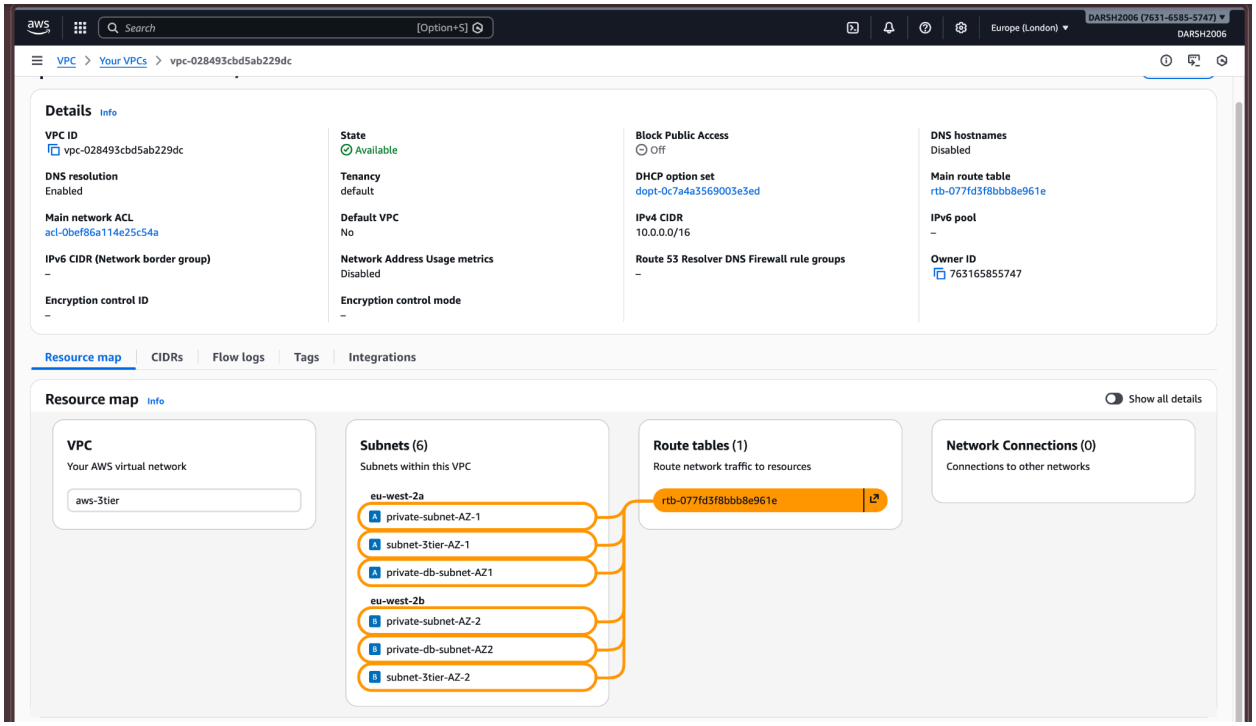


STEP 3 : VPC Setup and Subnet Creation

A custom VPC was created with public and private subnets to isolate web, application, and database tiers.

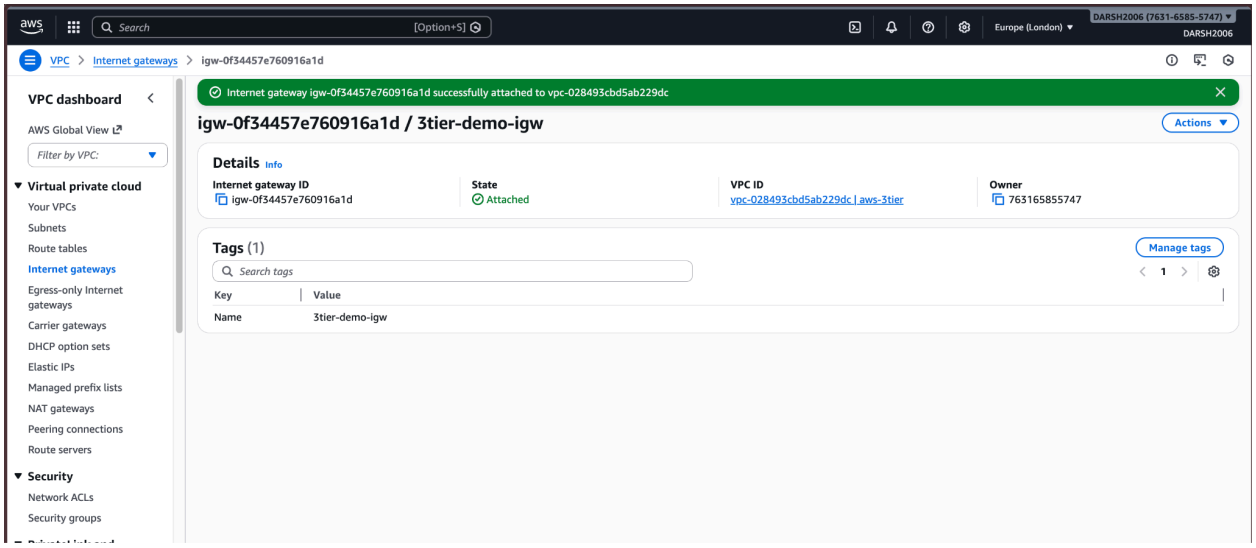


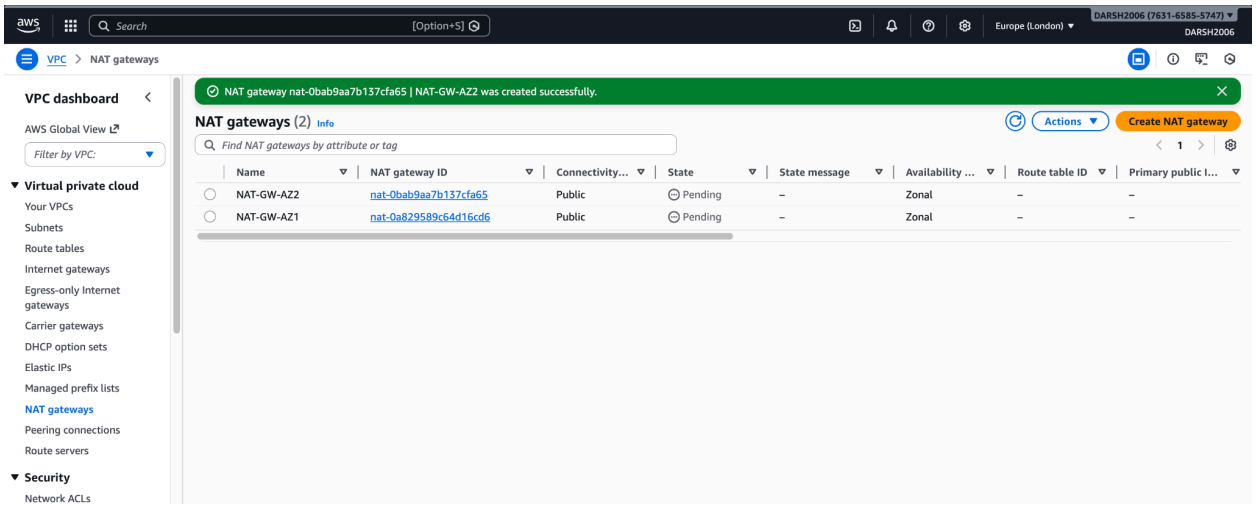



aws
Search
[Option+S]
Europe (London)
DARSH2006 (7631-6585-5747)
DARSH2006
VPC > Your VPCs > vpc-028493cbd5ab229dc
Details Info
VPC ID
vpc-028493cbd5ab229dc
State
Available
Block Public Access
Off
DNS hostnames
Disabled
DNS resolution
Enabled
Tenancy
default
DHCP option set
dopt-0c7a4a3569003e3ed
Main route table
rtb-077fd3f8bbb8e961e
Main network ACL
acl-0bef86a114e25c54a
Default VPC
No
IPv4 CIDR
10.0.0.0/16
IPv6 pool
–
IPv6 CIDR (Network border group)
–
Network Address Usage metrics
Disabled
Route 53 Resolver DNS Firewall rule groups
–
Owner ID
763165855747
Encryption control ID
–
Encryption control mode
–
Resource map
CIDRs
Flow logs
Tags
Integrations
Resource map Info
Show all details
VPC
Your AWS virtual network
aws-3tier
Subnets (6)
Subnets within this VPC
eu-west-2a
private-subnet-AZ-1
subnet-3tier-AZ-1
private-db-subnet-AZ1
eu-west-2b
private-subnet-AZ-2
private-db-subnet-AZ2
subnet-3tier-AZ-2
Route tables (1)
Route network traffic to resources
rtb-077fd3f8bbb8e961e
Network Connections (0)
Connections to other networks

STEP 4 : Internet Gateway & NAT Gateway Setup

The Internet Gateway allows public access, while the NAT Gateway enables private instances to access the internet securely.

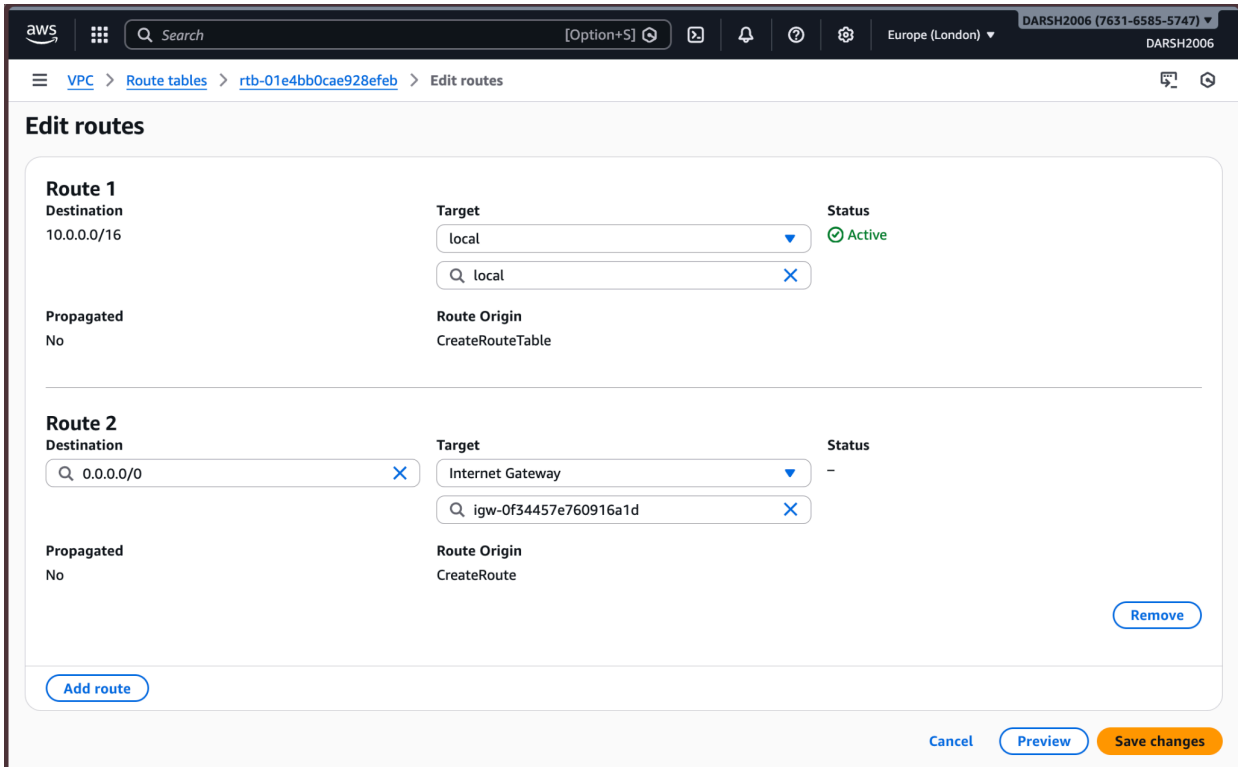


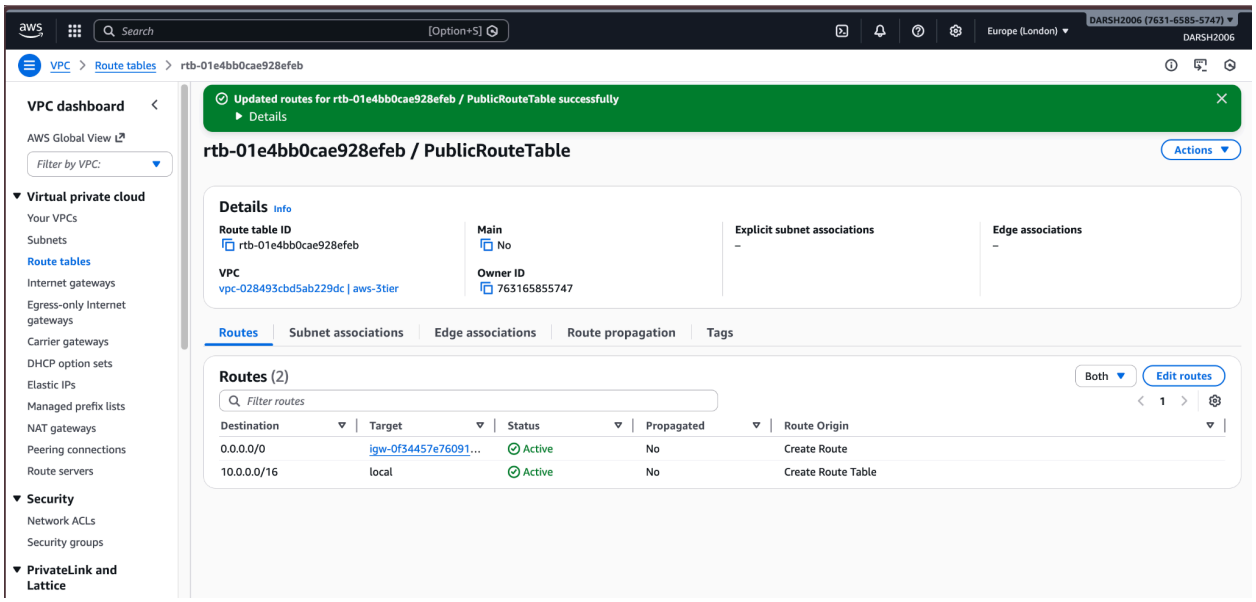


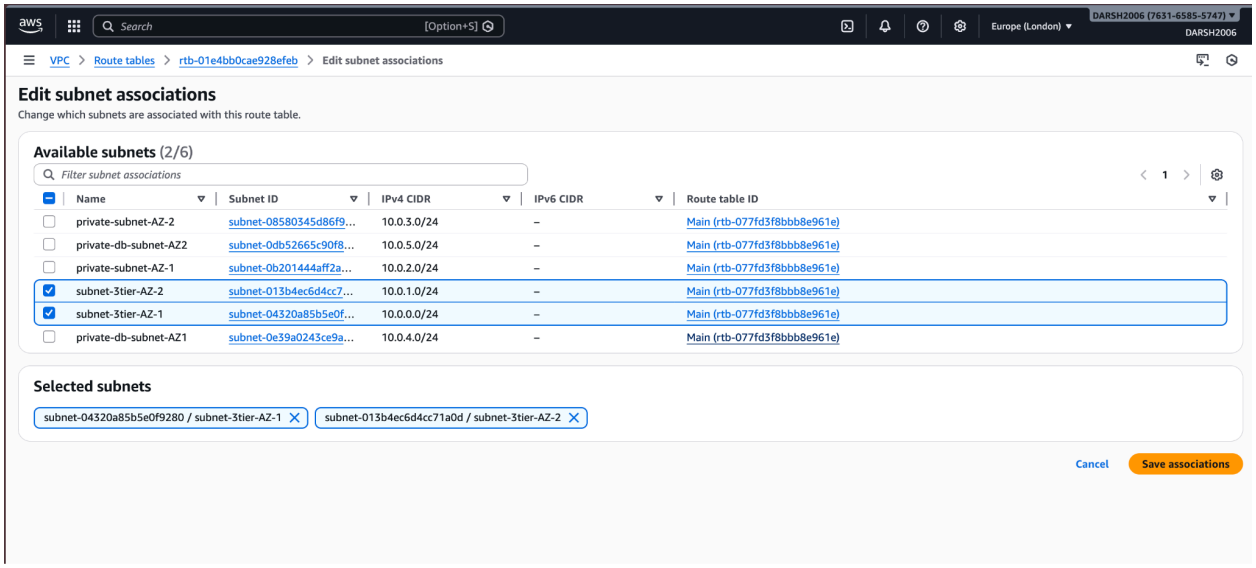
STEP 5 : Route Tables and Subnet Mapping

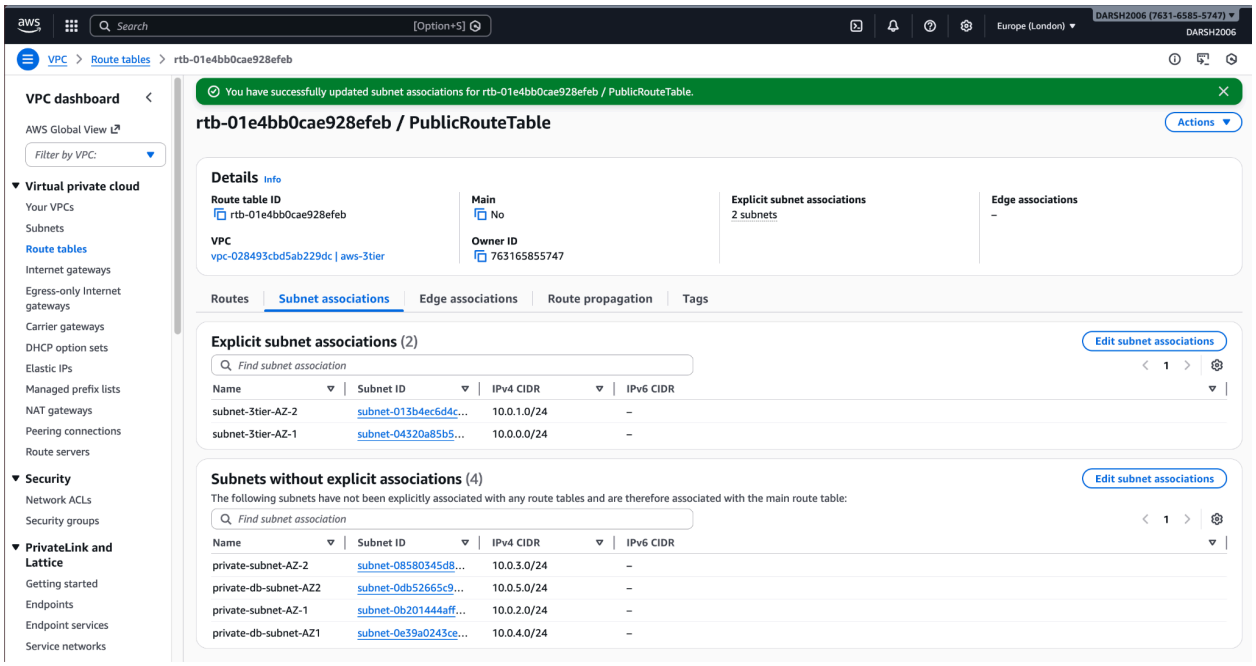
Route tables were configured to control traffic flow between public and private subnets.

ROUTE TABLE 1 : PublicRouteTable

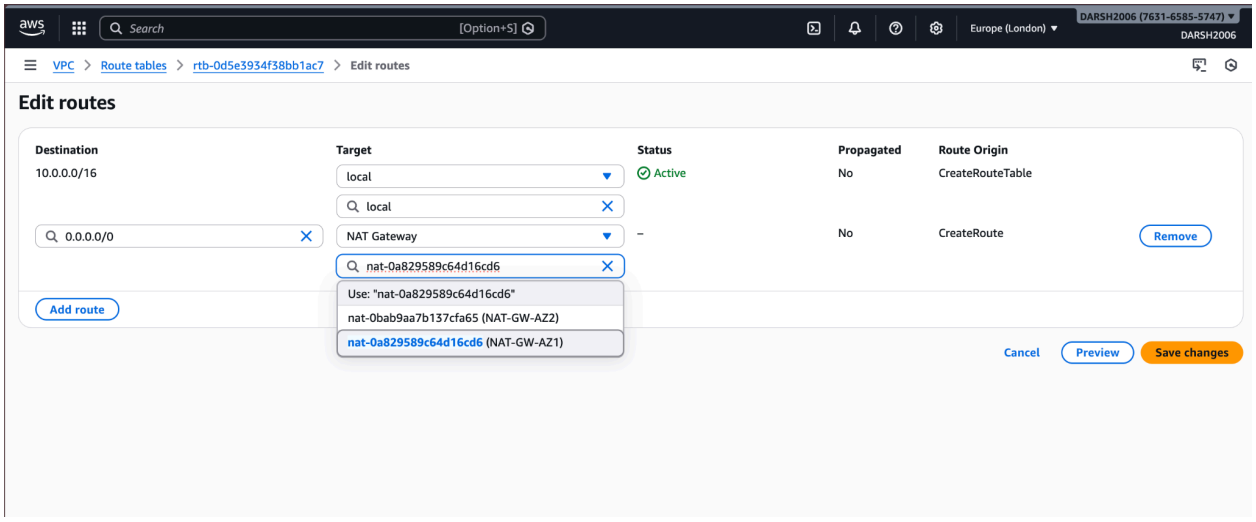


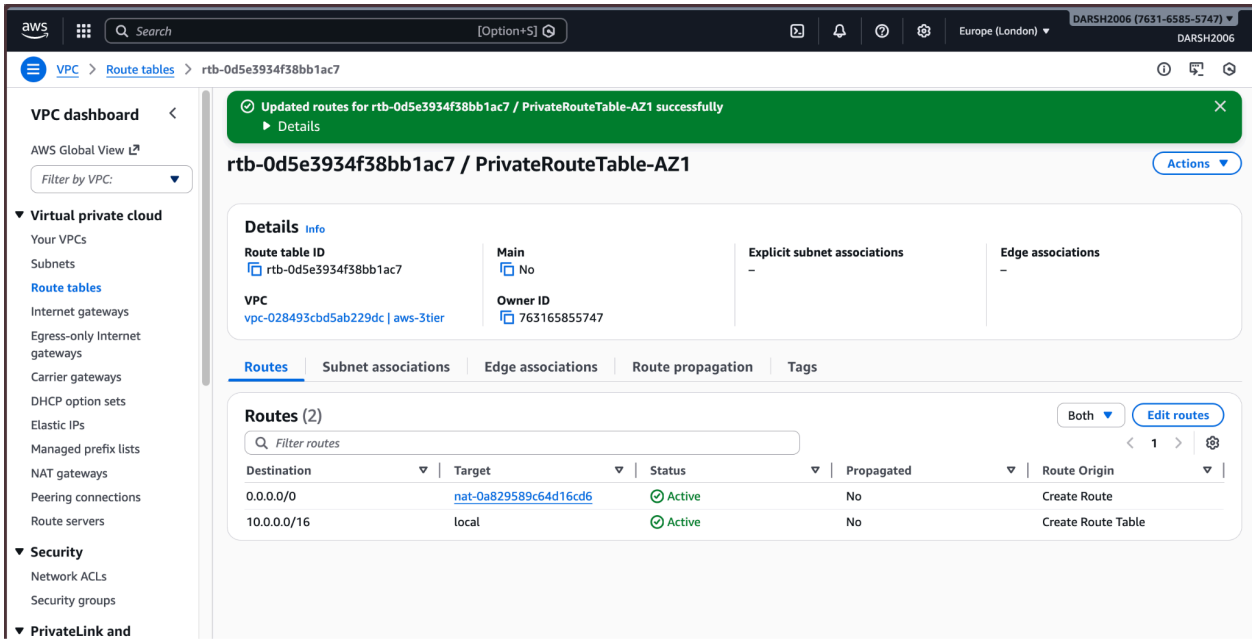



aws
Search
[Option+S]
Europe (London)
DARSH2006 (7631-6585-5747)
DARSH2006
VPC > Route tables > rtb-01e4bb0cae928efeb > Edit subnet associations
Edit subnet associations
Change which subnets are associated with this route table.
Available subnets (2/6)
Filter subnet associations
Name | Subnet ID | IPv4 CIDR | IPv6 CIDR | Route table ID
private-subnet-AZ-2 | subnet-08580345d86f9... | 10.0.3.0/24 | – | Main (rtb-077fd3f8bbb8e961e)
private-db-subnet-AZ2 | subnet-0db52665c90f8... | 10.0.5.0/24 | – | Main (rtb-077fd3f8bbb8e961e)
private-subnet-AZ-1 | subnet-0b201444aff2a... | 10.0.2.0/24 | – | Main (rtb-077fd3f8bbb8e961e)
subnet-3tier-AZ-2 | subnet-013b4ec6d4cc7... | 10.0.1.0/24 | – | Main (rtb-077fd3f8bbb8e961e)
subnet-3tier-AZ-1 | subnet-04320a85b5e0f... | 10.0.0.0/24 | – | Main (rtb-077fd3f8bbb8e961e)
private-db-subnet-AZ1 | subnet-0e39a0243ce9a... | 10.0.4.0/24 | – | Main (rtb-077fd3f8bbb8e961e)
Selected subnets
subnet-04320a85b5e0f9280 / subnet-3tier-AZ-1
subnet-013b4ec6d4cc71a0d / subnet-3tier-AZ-2
Cancel
Save associations


aws
Search
[Option+S]
Europe (London)
DARSH2006 (7631-6585-5747)
DARSH2006
VPC > Route tables > rtb-01e4bb0cae928efeb
VPC dashboard
AWS Global View
Filter by VPC:
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only Internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
Peering connections
Route servers
Security
Network ACLs
Security groups
PrivateLink and Lattice
Getting started
Endpoints
Endpoint services
Service networks
You have successfully updated subnet associations for rtb-01e4bb0cae928efeb / PublicRouteTable.
rtb-01e4bb0cae928efeb / PublicRouteTable
Actions
Details Info
Route table ID
rtb-01e4bb0cae928efeb
Main
No
Explicit subnet associations
2 subnets
Edge associations
–
VPC
vpc-028493cbd5ab229dc | aws-3tier
Owner ID
763165855747
Routes
Subnet associations
Edge associations
Route propagation
Tags
Explicit subnet associations (2)
Edit subnet associations
Find subnet association
Name | Subnet ID | IPv4 CIDR | IPv6 CIDR
subnet-3tier-AZ-2 | subnet-013b4ec6d4c... | 10.0.1.0/24 | –
subnet-3tier-AZ-1 | subnet-04320a85b5... | 10.0.0.0/24 | –
Subnets without explicit associations (4)
Edit subnet associations
The following subnets have not been explicitly associated with any route tables and are therefore associated with the main route table:
Find subnet association
Name | Subnet ID | IPv4 CIDR | IPv6 CIDR
private-subnet-AZ-2 | subnet-08580345d8... | 10.0.3.0/24 | –
private-db-subnet-AZ2 | subnet-0db52665c9... | 10.0.5.0/24 | –
private-subnet-AZ-1 | subnet-0b201444aff... | 10.0.2.0/24 | –
private-db-subnet-AZ1 | subnet-0e39a0243ce... | 10.0.4.0/24 | –

ROUTE TABLE 2 : PrivateRouteTable-AZ1





aws | Search [Option+S] | Europe (London) | DARSH2006 (7631-6585-5747) | DARSH2006

VPC > Route tables > rtb-0d5e3934f38bb1ac7 > Edit subnet associations

Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (1/6)

Filter subnet associations

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
private-subnet-AZ-2	subnet-08580345d86f92b7f	10.0.3.0/24	–	Main (rtb-077fd3f8bbb8e961e)
private-db-subnet-AZ2	subnet-0db52665c90f8ac7f	10.0.5.0/24	–	Main (rtb-077fd3f8bbb8e961e)
private-subnet-AZ-1	subnet-0b201444aff2a07c1	10.0.2.0/24	–	rtb-0d5e3934f38bb1ac7 / PrivateRouteTable-AZ1
subnet-3tier-AZ-2	subnet-013b4ec6d4cc71a0d	10.0.1.0/24	–	rtb-01e4bb0cae928efeb / PublicRouteTable
subnet-3tier-AZ-1	subnet-04320a85b5e0f9280	10.0.0.0/24	–	rtb-01e4bb0cae928efeb / PublicRouteTable
private-db-subnet-AZ1	subnet-0e39a0243ce9a90e5	10.0.4.0/24	–	Main (rtb-077fd3f8bbb8e961e)

Selected subnets

subnet-0b201444aff2a07c1 / private-subnet-AZ-1

Cancel | Save associations

aws | Search [Option+S] | Europe (London) | DARSH2006 (7631-6585-5747) | DARSH2006

VPC > Route tables > rtb-0d5e3934f38bb1ac7

rtb-0d5e3934f38bb1ac7 / PrivateRouteTable-AZ1

Actions

Details Info

Route table ID	Main	Explicit subnet associations	Edge associations
rtb-0d5e3934f38bb1ac7	No	subnet-0b201444aff2a07c1 / private-subnet-AZ-1	–
VPC	**Owner ID**		
vpc-028493cbd5ab229dc | aws-3tier	763165855747		

Routes | Subnet associations | Edge associations | Route propagation | Tags

Explicit subnet associations (1)

Edit subnet associations

Find subnet association

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
private-subnet-AZ-1	subnet-0b201444aff2a07c1	10.0.2.0/24	–

Subnets without explicit associations (3)

Edit subnet associations

The following subnets have not been explicitly associated with any route tables and are therefore associated with the main route table:

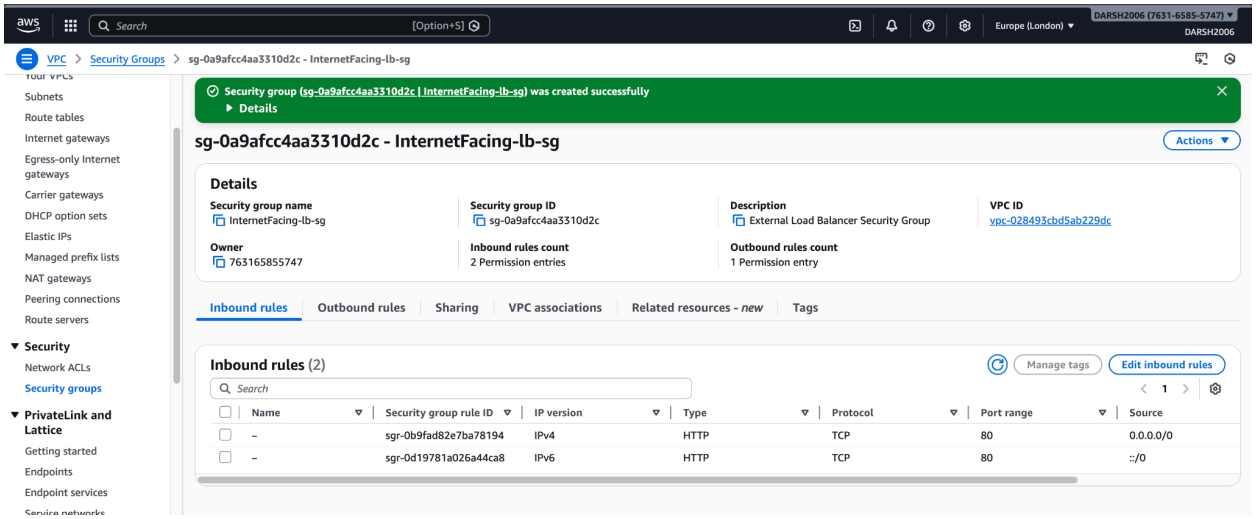
Find subnet association

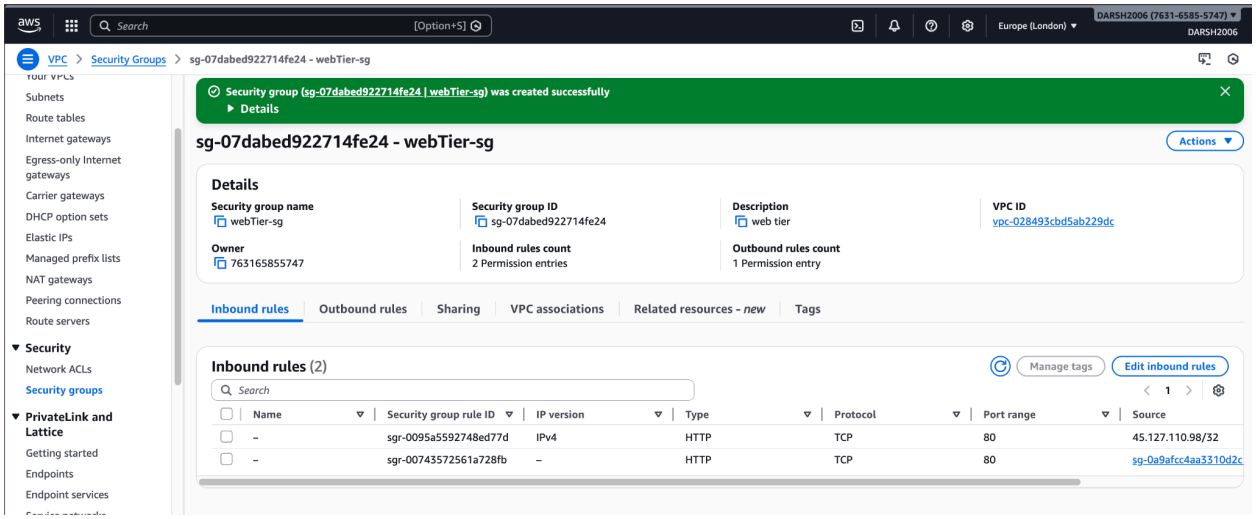
Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
private-subnet-AZ-2	subnet-08580345d86f92b7f	10.0.3.0/24	–
private-db-subnet-AZ2	subnet-0db52665c90f8ac7f	10.0.5.0/24	–
private-db-subnet-AZ1	subnet-0e39a0243ce9a90e5	10.0.4.0/24	–

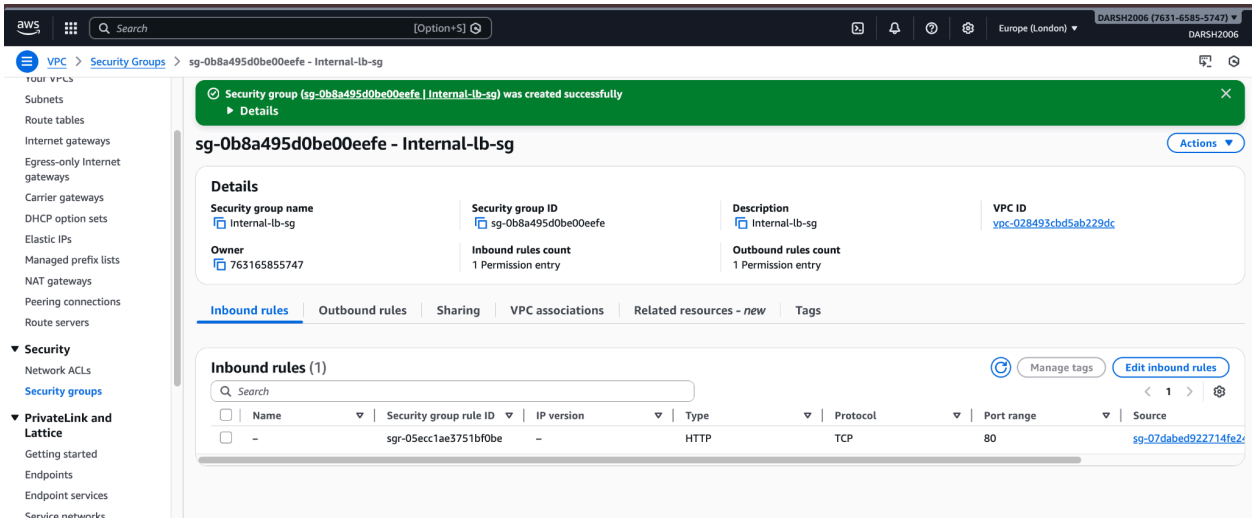
STEP 6 : Security Group Configuration (Web, App, DB)

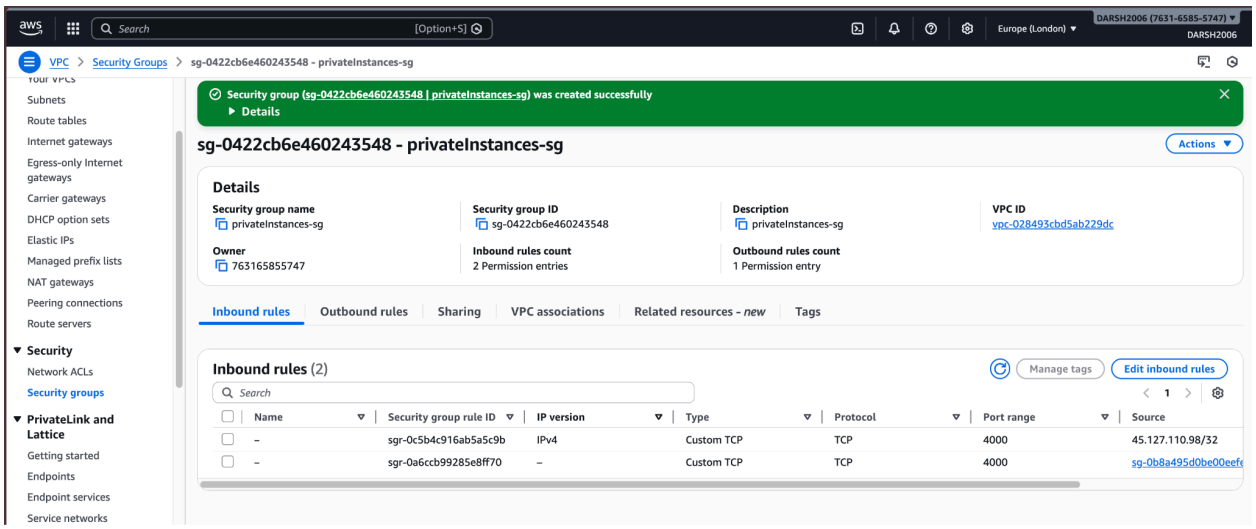
Separate security groups were created to restrict access between the web, application, and database tiers.

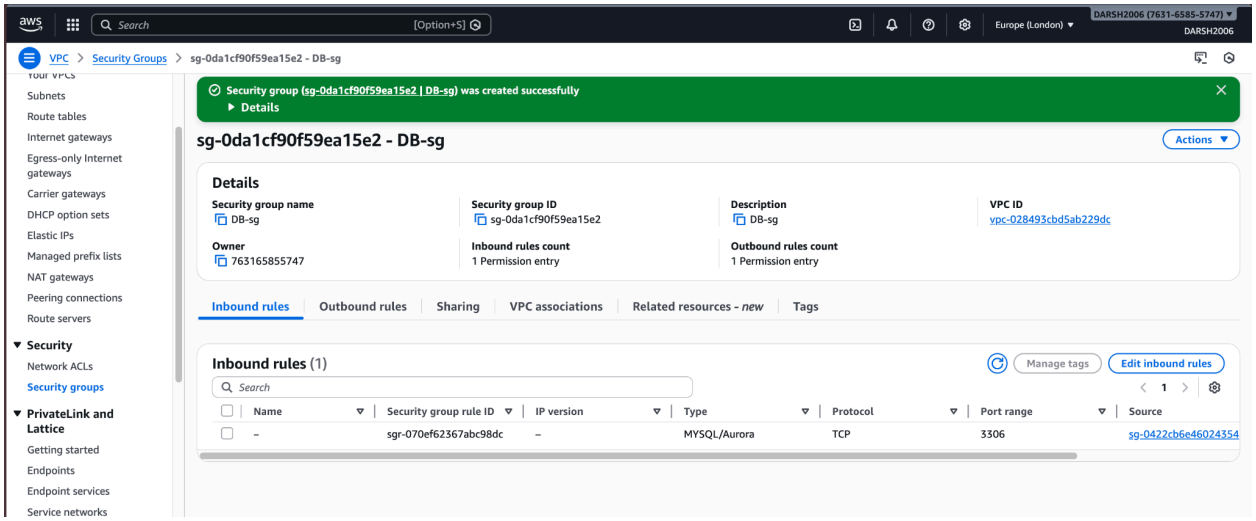
Create 5 Security Groups :





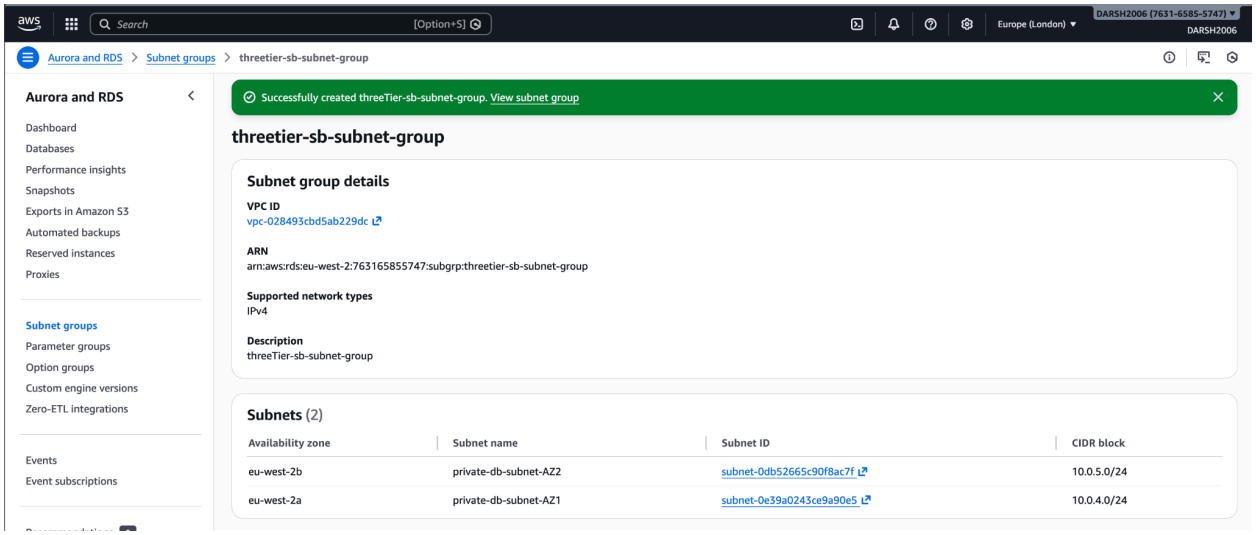

aws
Search
[Option+S]
Europe (London)
DARSH2006 (7631-6585-5747)
DARSH2006
VPC > Security Groups > sg-0b8a495d0be00eefe - Internal-lb-sg
Subnets
Route tables
Internet gateways
Egress-only Internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
Peering connections
Route servers
Security
Network ACLs
Security groups
PrivateLink and Lattice
Getting started
Endpoints
Endpoint services
Security group (sg-0b8a495d0be00eefe | Internal-lb-sg) was created successfully
Details
sg-0b8a495d0be00eefe - Internal-lb-sg
Actions
Details
Security group name
Internal-lb-sg
Security group ID
sg-0b8a495d0be00eefe
Description
Internal-lb-sg
VPC ID
vpc-028493cbd5ab229dc
Owner
763165855747
Inbound rules count
1 Permission entry
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Sharing
VPC associations
Related resources - new
Tags
Inbound rules (1)
Manage tags
Edit inbound rules
Search
Name
Security group rule ID
IP version
Type
Protocol
Port range
Source
–
sgr-05ecc1ae3751bf0be
–
HTTP
TCP
80
sg-07dabed922714fe2

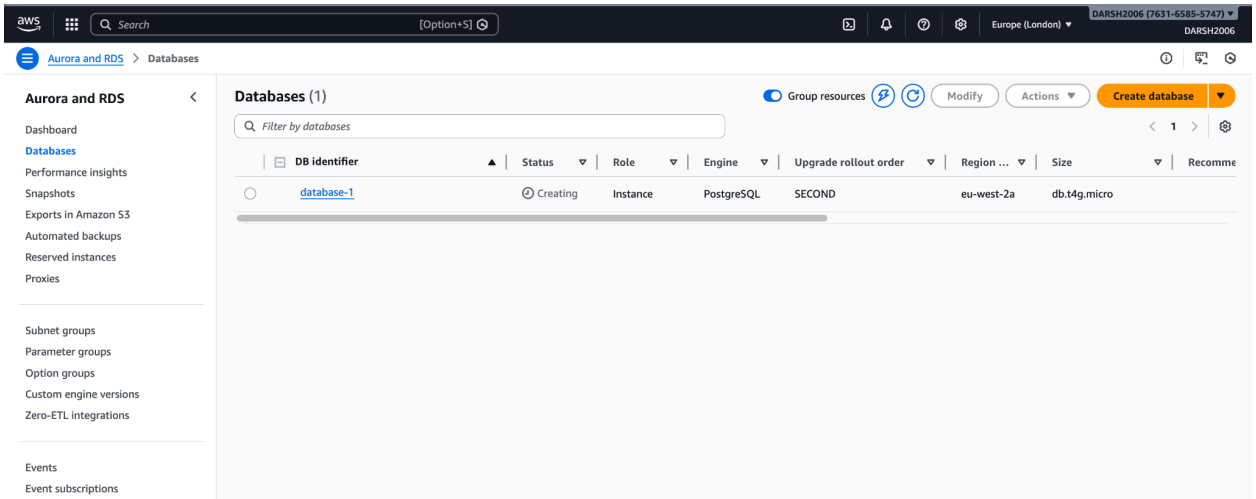

aws
Search
[Option+S]
Europe (London)
DARSH2006 (7631-6585-5747)
DARSH2006
VPC > Security Groups > sg-0422cb6e460243548 - privateInstances-sg
Subnets
Route tables
Internet gateways
Egress-only Internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
Peering connections
Route servers
Security
Network ACLs
Security groups
PrivateLink and Lattice
Getting started
Endpoints
Endpoint services
Service networks
Security group (sg-0422cb6e460243548 | privateInstances-sg) was created successfully
Details
sg-0422cb6e460243548 - privateInstances-sg
Actions
Details
Security group name
privateInstances-sg
Security group ID
sg-0422cb6e460243548
Description
privateInstances-sg
VPC ID
vpc-028493cbd5ab229dc
Owner
763165855747
Inbound rules count
2 Permission entries
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Sharing
VPC associations
Related resources - new
Tags
Inbound rules (2)
Manage tags
Edit inbound rules
Search
Name
Security group rule ID
IP version
Type
Protocol
Port range
Source
–
sgr-0c5b4c916ab5a5c9b
IPv4
Custom TCP
TCP
4000
45.127.110.98/32
–
sgr-0a6ccb99285e8ff70
–
Custom TCP
TCP
4000
sg-0b8a495d0be00eefe


aws
Search
[Option+S]
Europe (London)
DARSH2006 (7631-6585-5747)
DARSH2006
VPC > Security Groups > sg-0da1cf90f59ea15e2 - DB-sg
Subnets
Route tables
Internet gateways
Egress-only Internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
Peering connections
Route servers
Security
Network ACLs
Security groups
PrivateLink and Lattice
Getting started
Endpoints
Endpoint services
Service networks
Security group (sg-0da1cf90f59ea15e2 | DB-sg) was created successfully
Details
sg-0da1cf90f59ea15e2 - DB-sg
Actions
Details
Security group name
DB-sg
Security group ID
sg-0da1cf90f59ea15e2
Description
DB-sg
VPC ID
vpc-028493cbd5ab229dc
Owner
763165855747
Inbound rules count
1 Permission entry
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Sharing
VPC associations
Related resources - new
Tags
Inbound rules (1)
Manage tags
Edit inbound rules
Search
Name
Security group rule ID
IP version
Type
Protocol
Port range
Source
–
sgr-070ef62367abc98dc
–
MYSQL/Aurora
TCP
3306
sg-0422cb6e46024354

STEP 7 : RDS Subnet Group and MySQL DB Setup

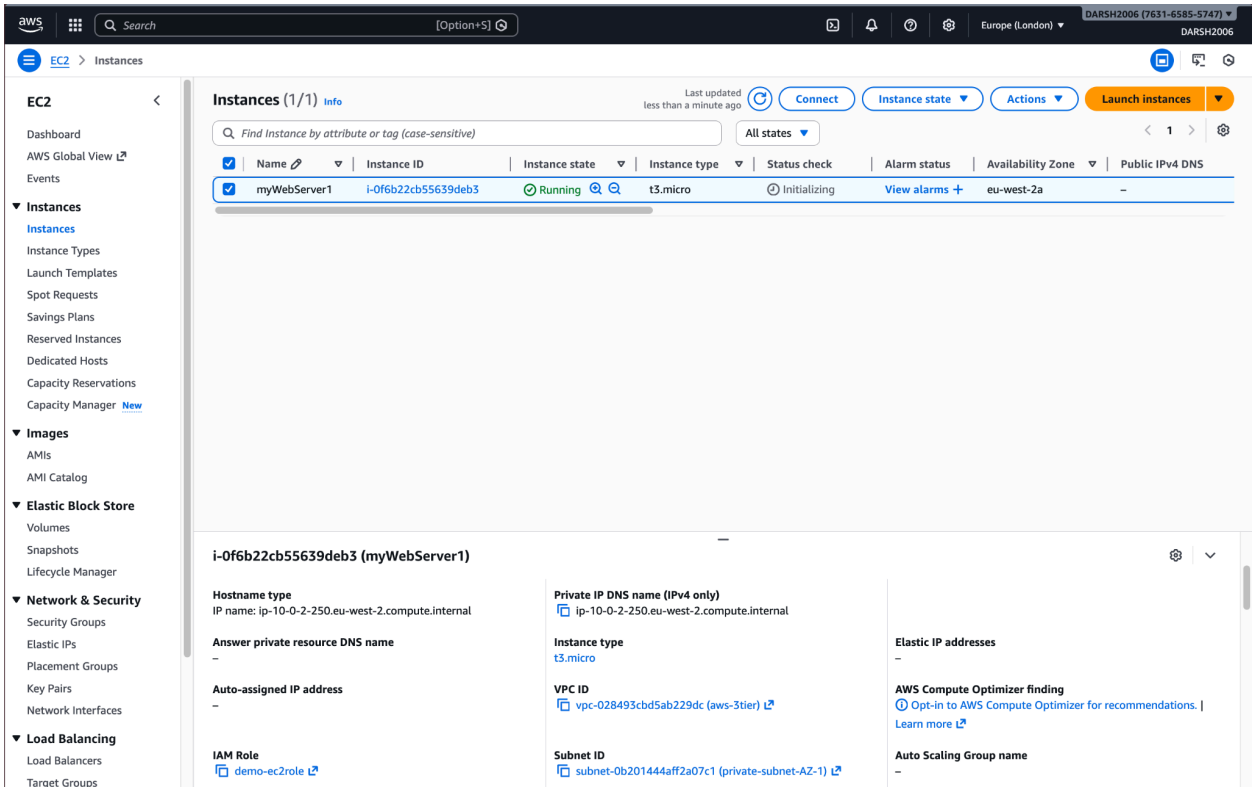
An Aurora database was deployed in private subnets to securely store application data.





STEP 8 : EC2 App Instance + Install MySQL Client

An EC2 instance was launched for the application tier and configured with the MySQL client to connect to the database.



Connect to the session manager

Commands :

>>> sudo -su ec2-user

>>> sudo wget https://dev.mysql.com/get/mysql80-community-release-el9-1.noarch.rpm

>>> sudo dnf install mysql80-community-release-el9-1.noarch.rpm -y

>>> sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2023

>>> sudo dnf install mysql-community-client -y (or) sudo yum install mysql

STEP 9 : Connect EC2 to DB + Create Tables

The EC2 instance was connected to the Aurora database, and required tables were created.

Mysql commands(to test) :

>>> Create database webappdb;

>>> Use webappdb;

>>> CREATE TABLE transactions (

id INT AUTO_INCREMENT PRIMARY KEY,

amount DECIMAL(10, 2) NOT NULL,

description VARCHAR(255) NOT NULL,

);

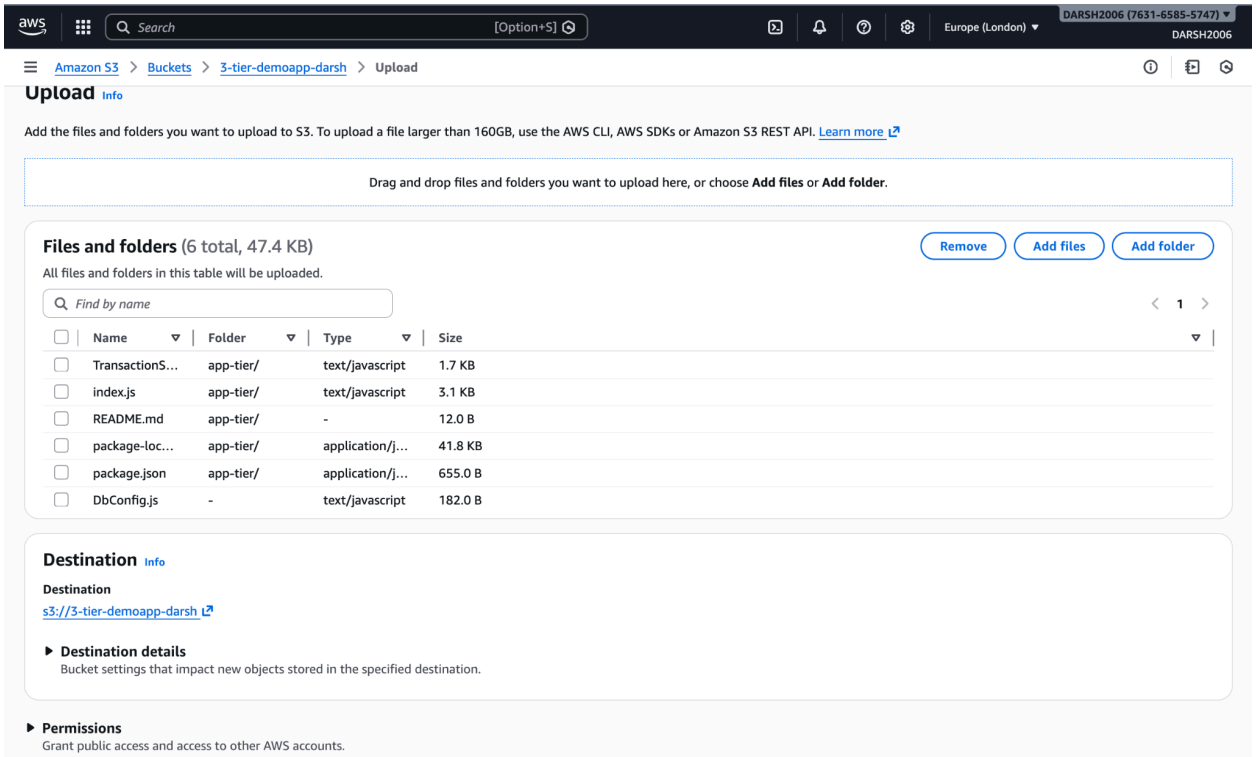
>>> INSERT INTO transactions (amount, description) VALUES (45.75, 'Groceries purchase');

>>> SELECT * FROM transactions;

```
mysql> create DATABASE webappdb;
Query OK, 1 row affected (0.04 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| webappdb           |
+--------------------+
5 rows in set (0.01 sec)

mysql> use webappdb;
Database changed
mysql> show tables;
Empty set (0.00 sec)

mysql> create table transaction;
ERROR 4028 (HY000): A table must have at least one visible column.
mysql> CREATE TABLE transactions (
    ->     id INT AUTO_INCREMENT PRIMARY KEY,
    ->     amount DECIMAL(10, 2) NOT NULL,
    ->     description VARCHAR(255) NOT NULL,
    ->     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> select * from transactions;
Empty set (0.01 sec)

mysql> INSERT INTO transactions (amount, description) VALUES (45.75, 'Groceries purchase');
Query OK, 1 row affected (0.00 sec)

mysql> select * from transactions;
+----+--------+--------------------+---------------------+
| id | amount | description        | created_at          |
+----+--------+--------------------+---------------------+
|  1 |  45.75 | Groceries purchase | 2026-01-30 08:26:16 |
+----+--------+--------------------+---------------------+
1 row in set (0.00 sec)
```

STEP 10 : Upload App Code & Install [Node.js]

Node.js was installed and backend application code was uploaded to the EC2 instance.



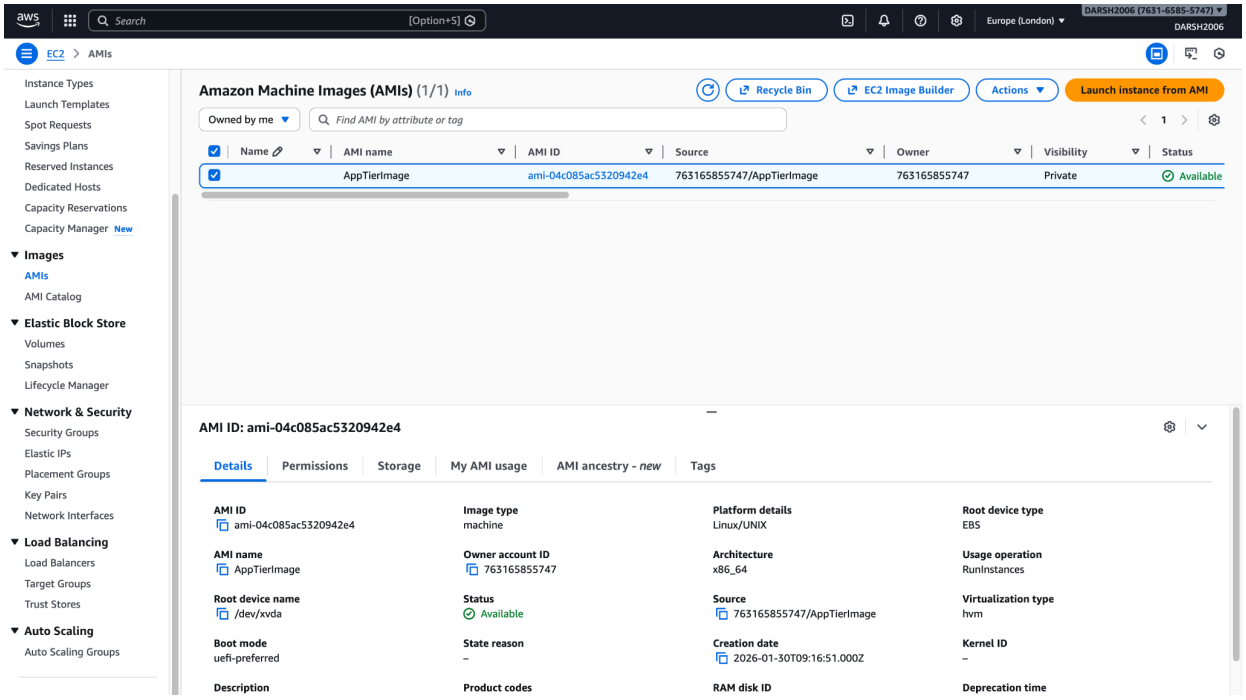
STEP 11 : Run App with PM2 and Test Backend

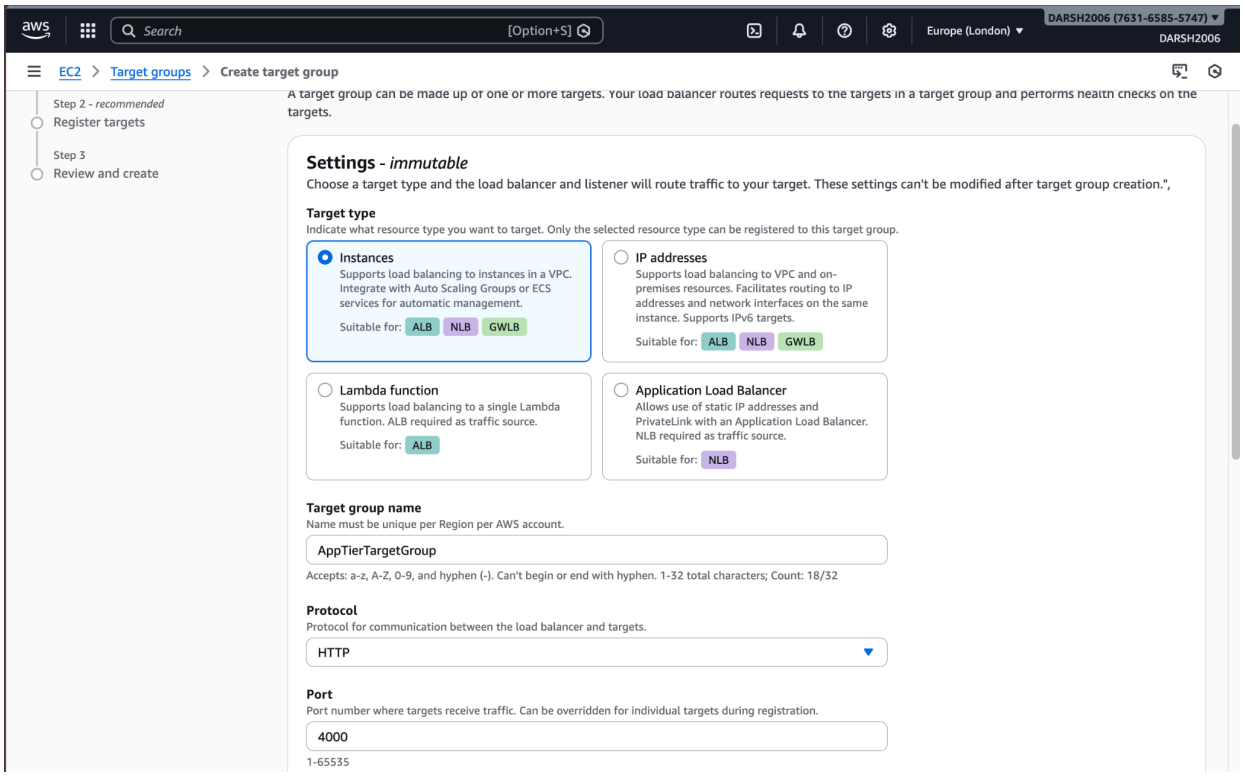
PM2 was used to run the Node.js application in the background and verify backend functionality.

```
[ec2-user@ip-10-0-2-183 app-tier]$ pm2 logs
[TAILING] Tailing last 15 lines for [all] processes (change the value with --lines option)
/home/ec2-user/.pm2/pm2.log last 15 lines:
PM2        | 2026-01-30T09:10:38: PM2 log: Node.js version      : 16.20.2
PM2        | 2026-01-30T09:10:38: PM2 log: Current arch         : x64
PM2        | 2026-01-30T09:10:38: PM2 log: PM2 home             : /home/ec2-user/.pm2
PM2        | 2026-01-30T09:10:38: PM2 log: PM2 PID file         : /home/ec2-user/.pm2/pm2.pid
PM2        | 2026-01-30T09:10:38: PM2 log: RPC socket file      : /home/ec2-user/.pm2/rpc.sock
PM2        | 2026-01-30T09:10:38: PM2 log: BUS socket file      : /home/ec2-user/.pm2/pub.sock
PM2        | 2026-01-30T09:10:38: PM2 log: Application log path : /home/ec2-user/.pm2/logs
PM2        | 2026-01-30T09:10:38: PM2 log: Worker Interval      : 30000
PM2        | 2026-01-30T09:10:38: PM2 log: Process dump file    : /home/ec2-user/.pm2/dump.pm2
PM2        | 2026-01-30T09:10:38: PM2 log: Concurrent actions   : 2
PM2        | 2026-01-30T09:10:38: PM2 log: SIGTERM timeout      : 1600
PM2        | 2026-01-30T09:10:38: PM2 log: Runtime Binary       : /home/ec2-user/.nvm/versions/node/v1
6.20.2/bin/node
PM2        | 2026-01-30T09:10:38: PM2 log: ========================================================
====================
PM2        | 2026-01-30T09:10:38: PM2 log: App [index:0] starting in -fork mode-
PM2        | 2026-01-30T09:10:38: PM2 log: App [index:0] online

/home/ec2-user/.pm2/logs/index-error.log last 15 lines:
/home/ec2-user/.pm2/logs/index-out.log last 15 lines:
0|index    | AB3 backend app listening at http://localhost:4000

^C
[ec2-user@ip-10-0-2-183 app-tier]$ history
    1  ping 8.8.8.8
    2  sudo wget https://dev.mysql.com/get/mysql80-community-release-el9-1.noarch.rpm
    3  sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2023
    4  sudo yum install mysql
    5  sudo dnf install mysql80-community-release-el9-1.noarch.rpm -y
    6  sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2023
    7  sudo yum install mysql
    8  mysql
    9  mysql -h database-1.cnygq0ua6iua.eu-west-2.rds.amazonaws.com -P 3306 -u admin -p'Darshana456' -
-ssl-verify-server-cert  --ssl-ca=/certs/global-bundle.pem mysql
   10  mysql -h database-1.cnygq0ua6iua.eu-west-2.rds.amazonaws.com -u admin -p
   11  curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh | bash
   12  source ~/.bashrc
   13  nvm install 16
   14  nvm use 16
   15  npm install -g pm2
   16  cd ~/
   17  aws s3 cp s3://3-tier-demoapp-darsh/app-tier/ app-tier --recursive
   18  cd ~/app-tier
   19  npm install
   20  pm2 start index.js
   21  pm2 list
   22  pm2 logs
```

STEP 12 : Create AMI, Target Group, Internal Load Balancer

An AMI was created from the app instance and used with a target group and internal load balancer for high availability.





aws | Search [Option+S] | Europe (London) | DARSH2006 (7631-6585-5747) | DARSH2006

EC2 > Load balancers

- Instance Types
- Launch Templates
- Spot Requests
- Savings Plans
- Reserved Instances
- Dedicated Hosts
- Capacity Reservations
- Capacity Manager New

Images
- AMIs
- AMI Catalog

Elastic Block Store
- Volumes
- Snapshots
- Lifecycle Manager

Network & Security
- Security Groups
- Elastic IPs
- Placement Groups
- Key Pairs
- Network Interfaces

Load Balancing
- Load Balancers
- Target Groups
- Trust Stores

Auto Scaling
- Auto Scaling Groups

Settings

Load balancers (1/1) What's new?

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Actions | Create load balancer

Filter load balancers

Name	State	Type	Scheme	IP address type	VPC ID	Availability Zones	Se
app-tier-internal-lb	Provisioning..	application	Internal	IPv4	vpc-028493cbd5ab229dc	2 Availability Zones	sg-

Load balancer: app-tier-internal-lb

Details | Listeners and rules | Network mapping | Resource map | Security | Monitoring | Integrations | Attributes | Capacity | Tags

Details

Load balancer type
Application

Status
Provisioning

VPC
vpc-028493cbd5ab229dc

Load balancer IP address type
IPv4

Scheme
Internal

Hosted zone
ZHURV8PSTC4K8

Availability Zones
subnet-08580345d86f92b7f eu-west-2b (euw2-az3)
subnet-0b201444aff2a07c1 eu-west-2a (euw2-az2)

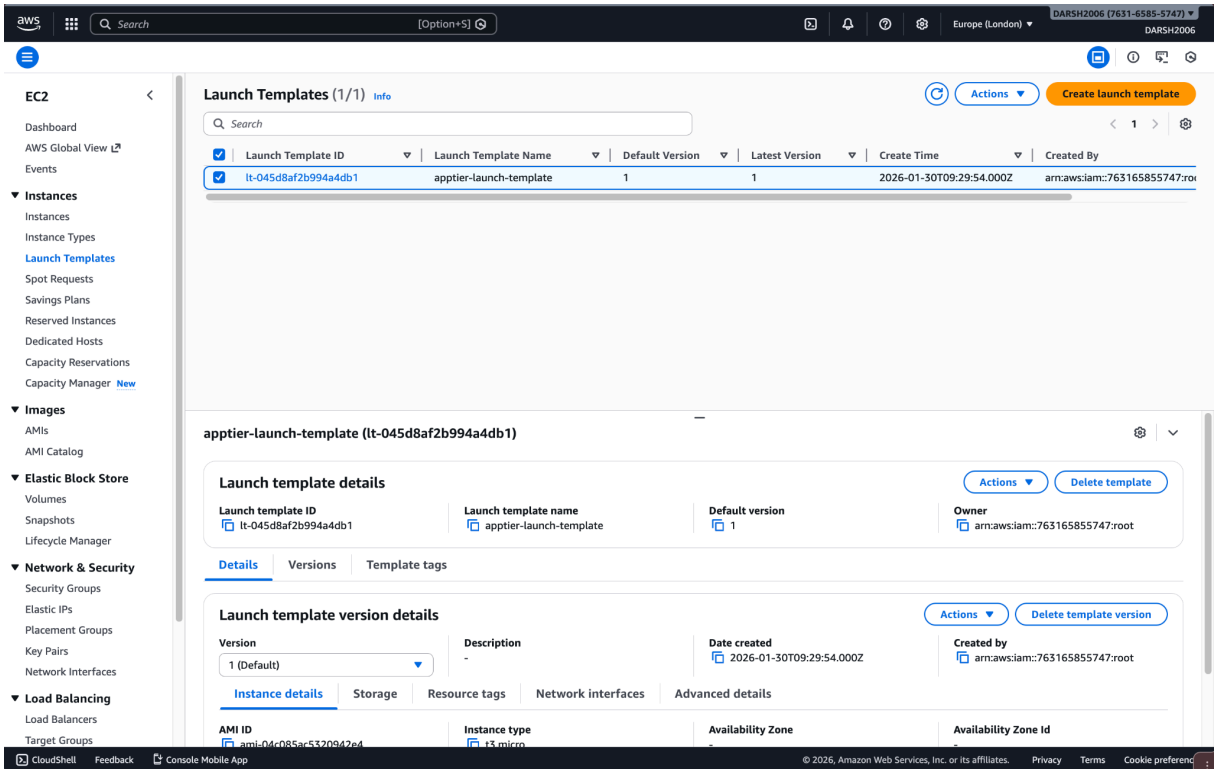
Date created
January 30, 2026, 14:54 (UTC+05:30)

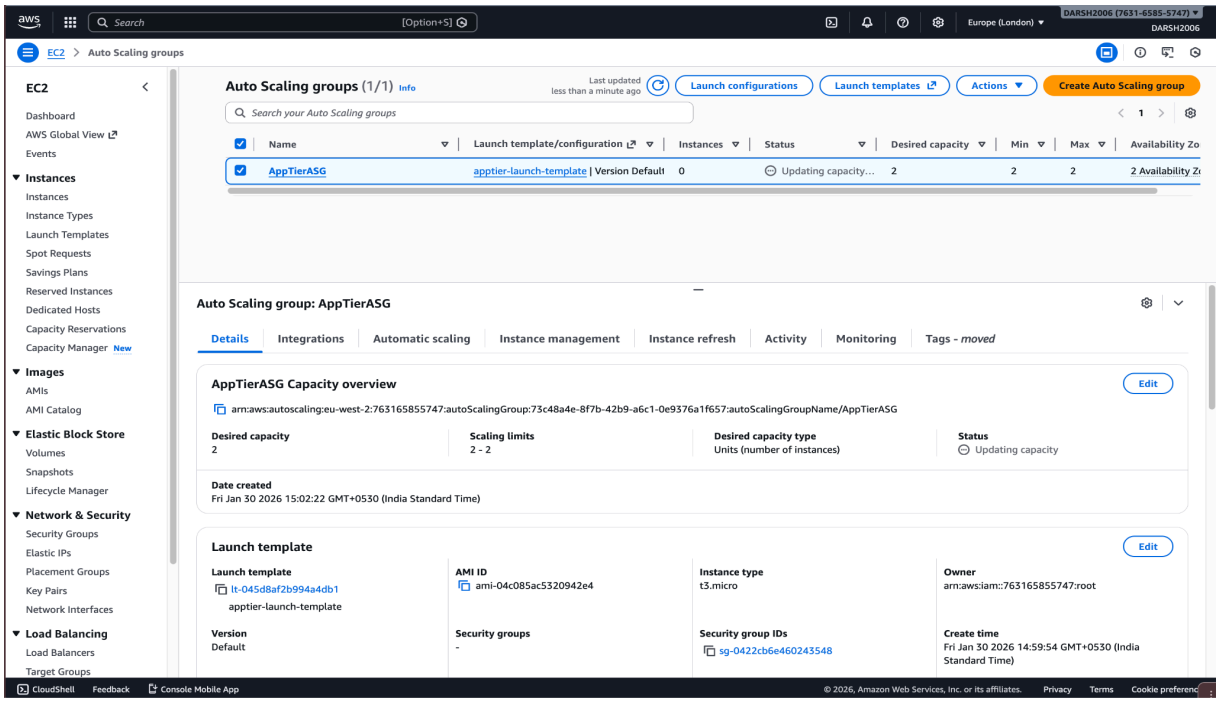
Load balancer ARN
arn:aws:elasticloadbalancing:eu-west-2:763165855747:loadbalancer/app/app-tier-internal-lb/09135472e68f54e6

DNS name Info
internal-app-tier-internal-lb-684668622.eu-west-2.elb.amazonaws.com (A Record)

STEP 13 : Launch Template + Auto Scaling for App Tier

A launch template and Auto Scaling Group were configured to automatically manage application instances.

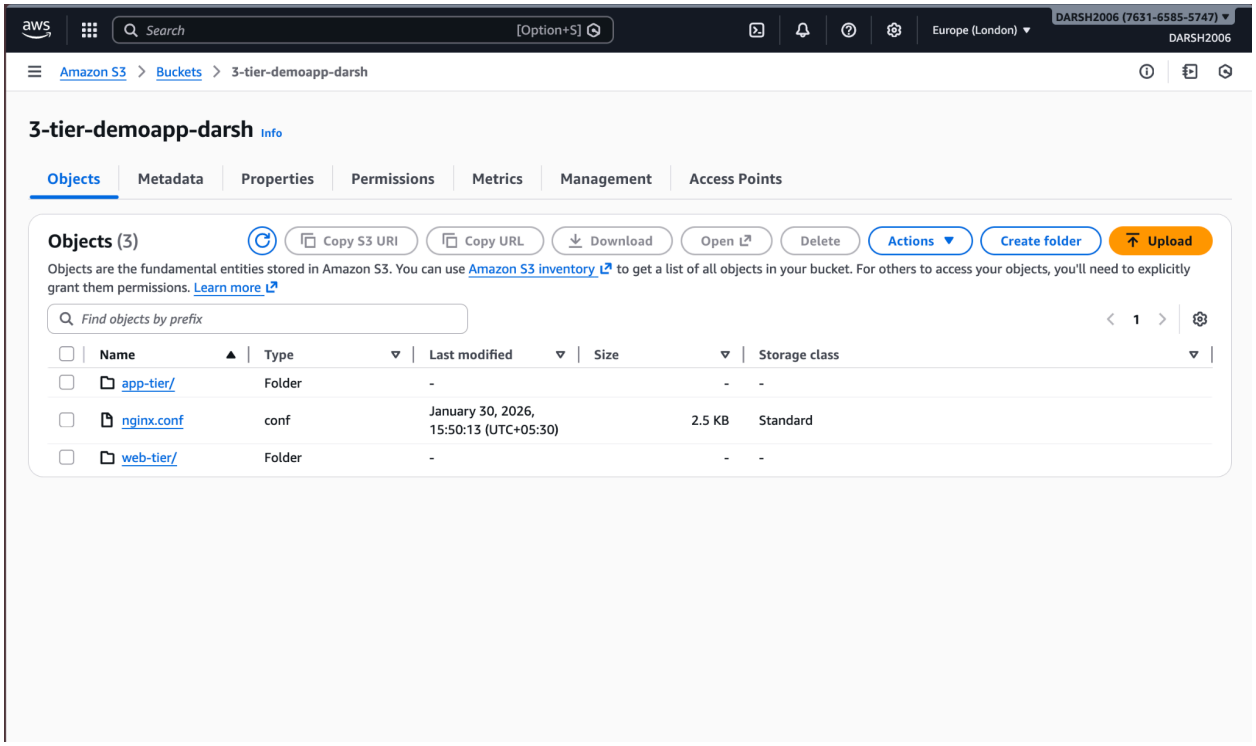




STEP 14 : Public EC2 Web Server + Install NGINX

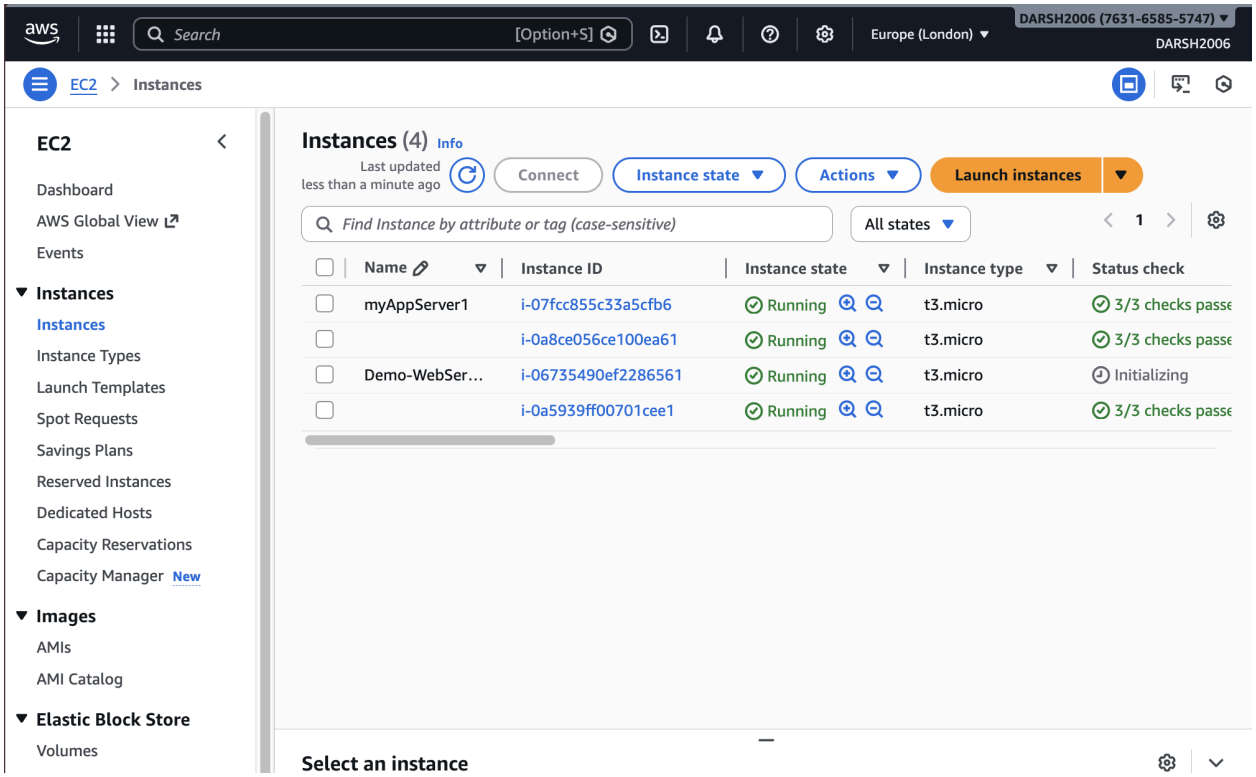
A public EC2 instance was launched and configured with NGINX to serve frontend content.

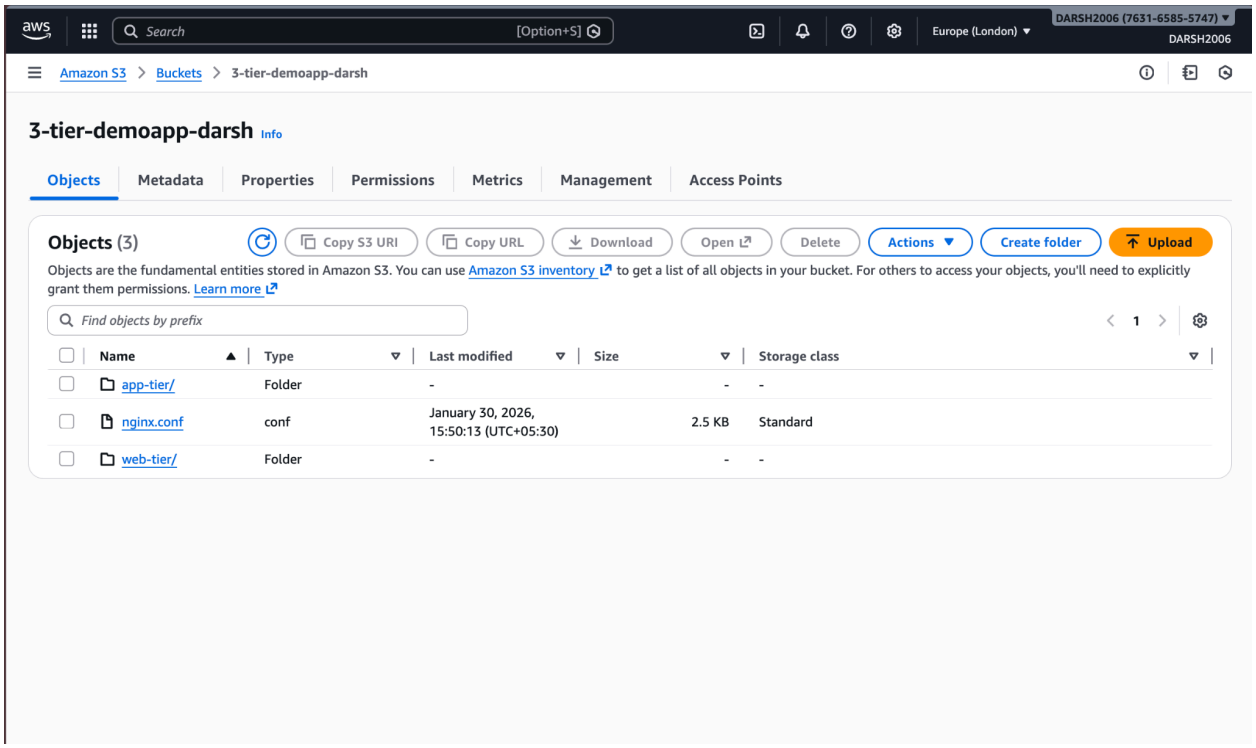
```
    include             /etc/nginx/mime.types;
    default_type        application/octet-stream;

    # Load modular configuration files from the /etc/nginx/conf.d directory.
    # See http://nginx.org/en/docs/ngx_core_module.html#include
    # for more information.
    include /etc/nginx/conf.d/*.conf;

    server {
        listen       80;
        listen       [::]:80;
        server_name  _;

        #health check
        location /health {
        default_type text/html;
        return 200 "<!DOCTYPE html><p>Web Tier Health Check</p>\n";
        }

        #react app and front end files
        location / {
        root    /home/ec2-user/web-tier/build;
        index index.html index.htm
        try_files $uri /index.html;        }

        #proxy for internal lb
        location /api/{
                proxy_pass http://internal-app-tier-internal-lb-684668622.eu-west-2.elb.amazonaws.com:80/;
        }


    }

# Settings for a TLS enabled server.
#
#    server {
#        listen       443 ssl http2;
#        listen       [::]:443 ssl http2;
#        server_name  _;
#        root         /usr/share/nginx/html;
#
#        ssl_certificate "/etc/pki/nginx/server.crt";
#        ssl_certificate_key "/etc/pki/nginx/private/server.key";
#        ssl_session_cache shared:SSL:1m;
#        ssl_session_timeout  10m;
#        ssl_ciphers PROFILE=SYSTEM;
#        ssl_prefer_server_ciphers on;
#
#        # Load configuration files for the default server block.
#        include /etc/nginx/default.d/*.conf;
#
#        error_page 404 /404.html;
#            location = /40x.html {
#        }
#
#        error_page 500 502 503 504 /50x.html;
#            location = /50x.html {
#        }
#    }

}
~
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --
```


aws
Search
[Option+S]
Europe (London)
DARSH2006 (7631-6585-5747)
DARSH2006
Amazon S3 > Buckets > 3-tier-demoapp-darsh
3-tier-demoapp-darsh Info
Objects
Metadata
Properties
Permissions
Metrics
Management
Access Points
Objects (3)
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Find objects by prefix
1
Name
Type
Last modified
Size
Storage class
app-tier/
Folder
-
-
-
nginx.conf
conf
January 30, 2026, 15:50:13 (UTC+05:30)
2.5 KB
Standard
web-tier/
Folder
-
-
-

STEP 15 : Upload Web Files + Permission Fixes

Frontend files were uploaded to the web server and required permissions were fixed.





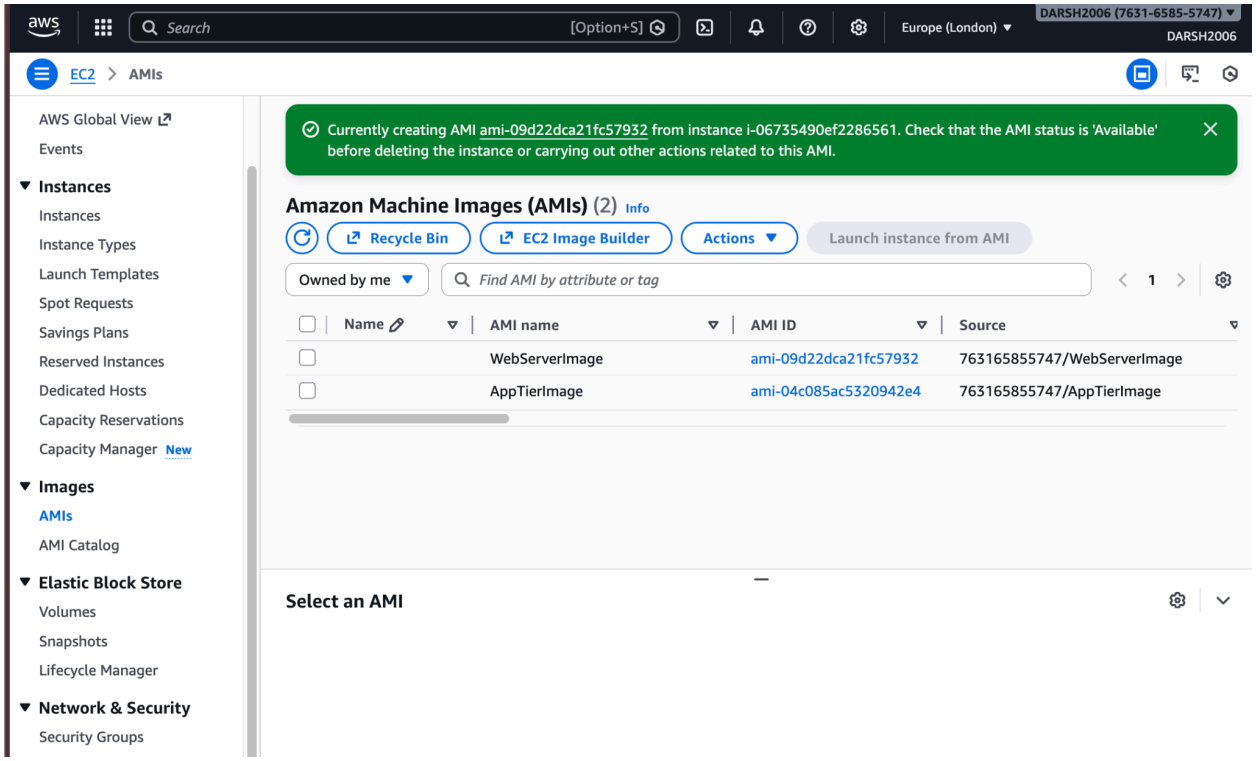
STEP 16 : Test NGINX + Internal LB Verification

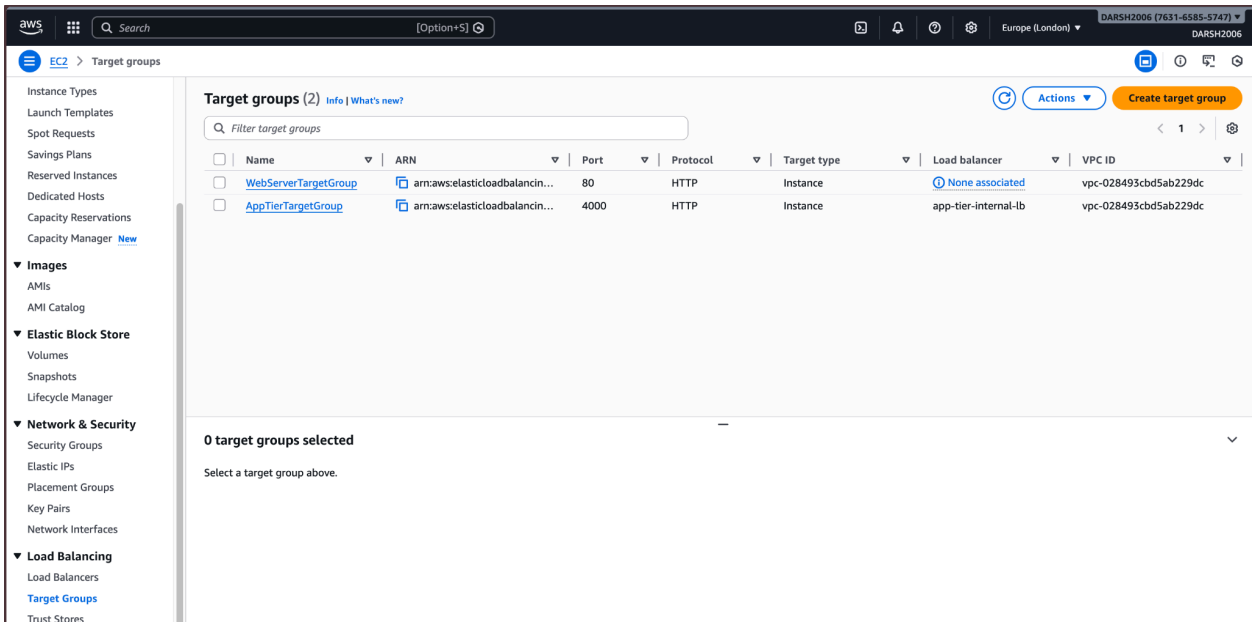
NGINX was tested to confirm correct routing to the internal load balancer.

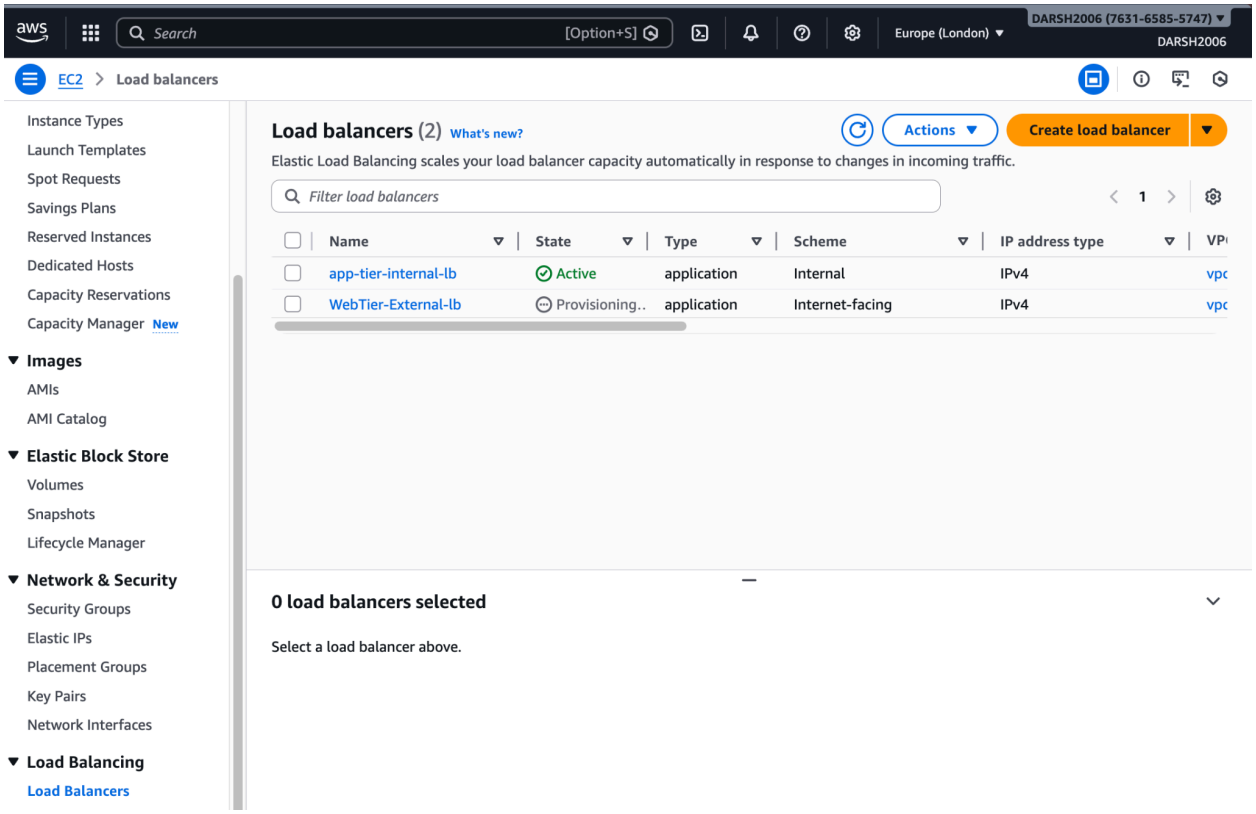
```
Session ID: root-657ohsaivrkv3jo9kbaa3v9o7e   Shortcuts   Instance ID: i-06735490ef2286561                                   Terminate

     Active: active (running) since Fri 2026-01-30 10:39:51 UTC; 1min 42s ago                                  /
   Main PID: 26512 (nginx)
● nginx.service - The nginx HTTP and reverse proxy server                                                       n
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
     Active: active (running) since Fri 2026-01-30 10:39:51 UTC; 1min 42s ago
   Main PID: 26512 (nginx)
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
     Active: active (running) since Fri 2026-01-30 10:39:51 UTC; 1min 42s ago
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled
)
     Active: active (running) since Fri 2026-01-30 10:39:51 UTC; 1min 42s ago
   Main PID: 26512 (nginx)
      Tasks: 3 (limit: 1067)
     Memory: 3.2M
        CPU: 59ms
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabledrvice - The nginx HTTP >>
)                                                                  iguration file /etc/ngi>>
     Active: active (running) since Fri 2026-01-30 10:39:51 UTC; 1min 42s ago        ation file /etc/nginx/n>>
   Main PID: 26512 (nginx)                                                           vice - The nginx HTTP a>>
      Tasks: 3 (limit: 1067)
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled
)
     Active: active (running) since Fri 2026-01-30 10:39:51 UTC; 1min 42s ago
   Main PID: 26512 (nginx)
      Tasks: 3 (limit: 1067)
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled
)
     Active: active (running) since Fri 2026-01-30 10:39:51 UTC; 1min 42s ago
   Main PID: 26512 (nginx)
      Tasks: 3 (limit: 1067)
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
     Active: active (running) since Fri 2026-01-30 10:39:51 UTC; 1min 42s ago
   Main PID: 26512 (nginx)
      Tasks: 3 (limit: 1067)
     Memory: 3.2M
        CPU: 59ms
     CGroup: /system.slice/nginx.service
             ├─26512 "nginx: master process /usr/sbin/nginx"
             ├─26513 "nginx: worker process"
             └─26514 "nginx: worker process"

Jan 30 10:39:51 ip-10-0-0-79.eu-west-2.compute.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Jan 30 10:39:51 ip-10-0-0-79.eu-west-2.compute.internal nginx[26510]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Jan 30 10:39:51 ip-10-0-0-79.eu-west-2.compute.internal nginx[26510]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Jan 30 10:39:51 ip-10-0-0-79.eu-west-2.compute.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
                                                                                                   ...sole.aws.amazon.com
```

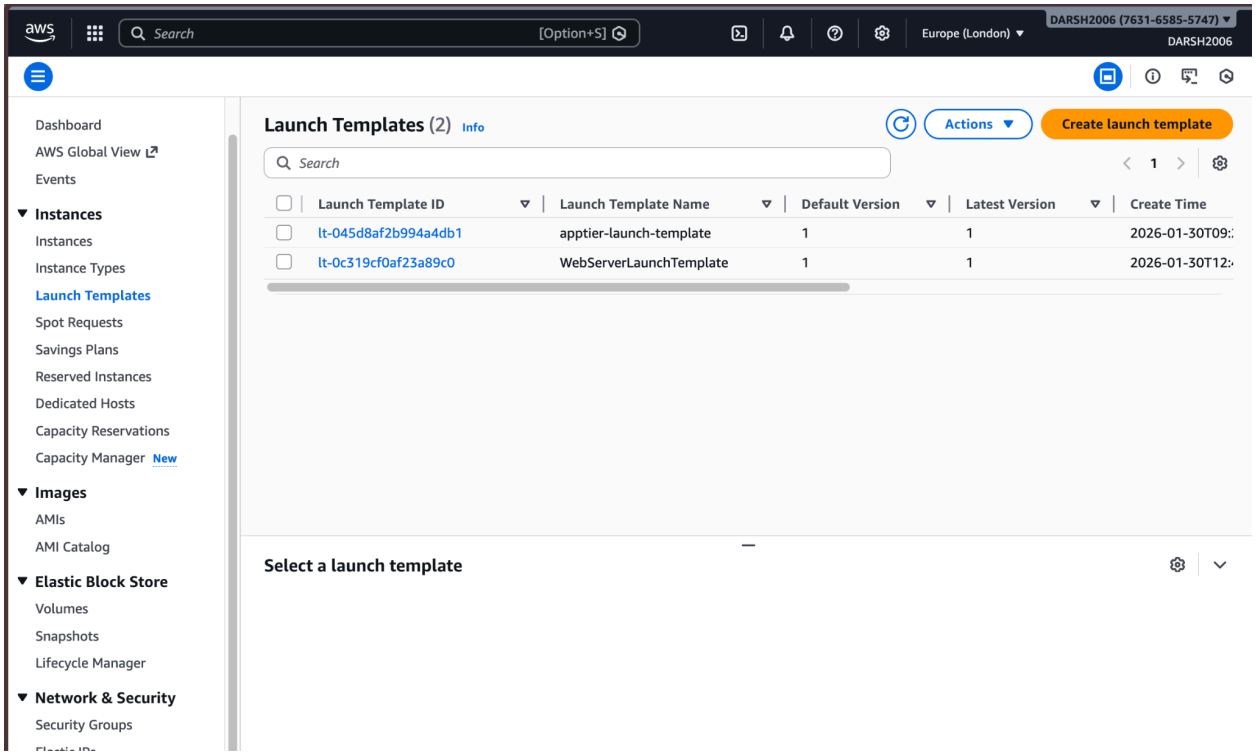
STEP 17 : Public Load Balancer + Auto Scaling

A public Application Load Balancer and Auto Scaling Group were configured for the web tier.



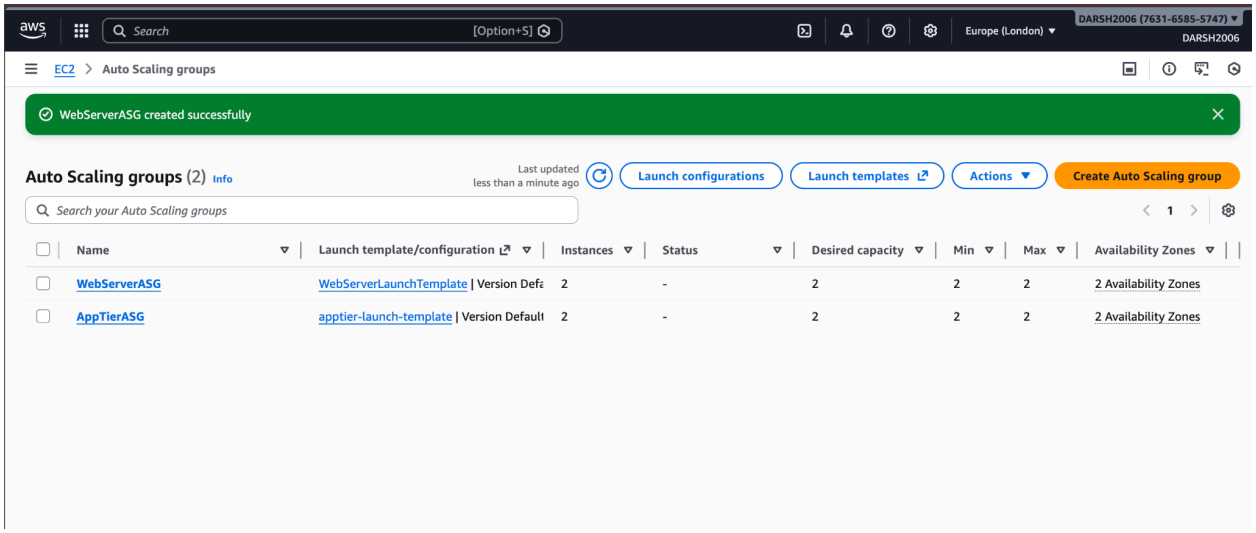



aws
Search
[Option+S]
Europe (London)
DARSH2006 (7631-6585-5747)
DARSH2006
EC2 > Load balancers
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Capacity Manager New
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Load Balancing
Load Balancers
Load balancers (2) What's new?
Actions
Create load balancer
Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.
Filter load balancers
Name
State
Type
Scheme
IP address type
app-tier-internal-lb
Active
application
Internal
IPv4
WebTier-External-lb
Provisioning..
application
Internet-facing
IPv4
0 load balancers selected
Select a load balancer above.

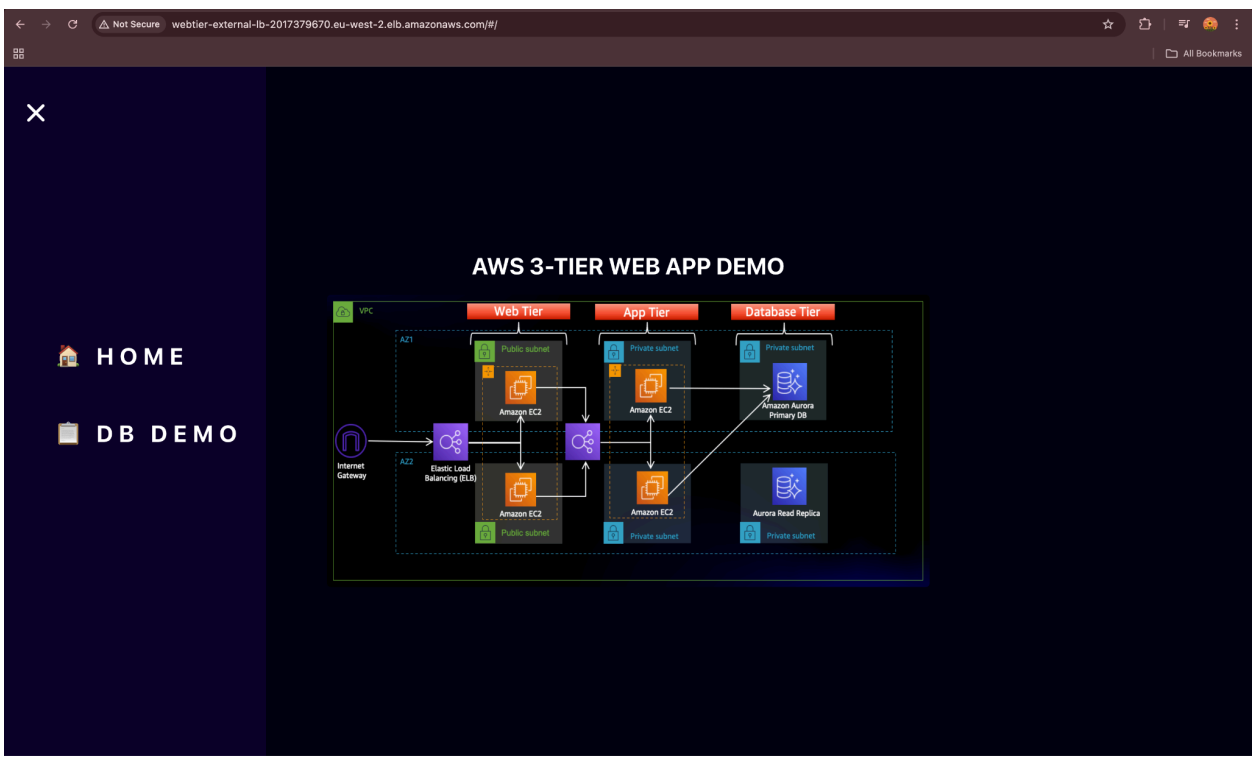

aws
Search
[Option+S]
Europe (London)
DARSH2006 (7631-6585-5747)
DARSH2006
Dashboard
AWS Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Capacity Manager New
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Launch Templates (2) Info
Actions
Create launch template
Search
Launch Template ID
Launch Template Name
Default Version
Latest Version
Create Time
lt-045d8af2b994a4db1
apptier-launch-template
1
1
lt-0c319cf0af23a89c0
WebServerLaunchTemplate
1
1
Select a launch template

STEP 18 : Final Testing: DNS + Live App

The application was accessed using the public load balancer DNS, confirming successful deployment.



FINAL OUTPUT



Project: Three-Tier Web Application on AWS
Name: Darshana S
GitHub:
https://github.com/Darshanasundar/three-tier-web-application.git
Date: 2026