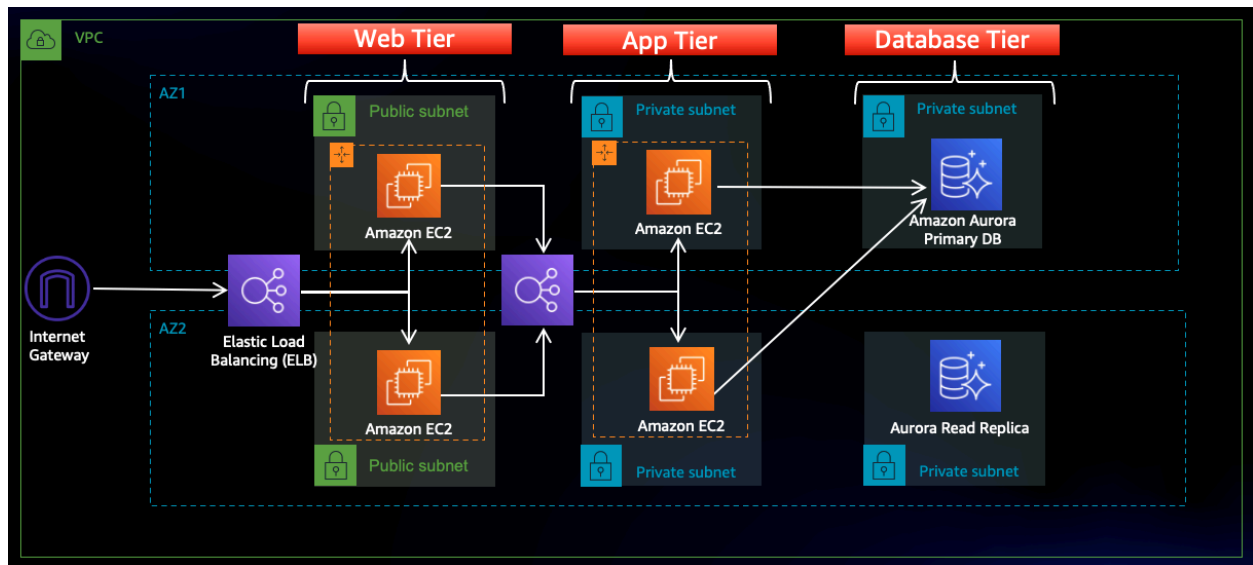


AWS Three-Tier Web Architecture Project

In this architecture, a public-facing Application Load Balancer forwards client traffic to our web tier EC2 instances. The web tier is running Nginx web servers that are configured to serve a React.js website and redirects our API calls to the application tier's internal facing load balancer. The internal facing load balancer then forwards that traffic to the application tier, which is written in Node.js. The application tier manipulates data in an Aurora MySQL multi-AZ database and returns it to our web tier. Load balancing, health checks and autoscaling groups are created at each layer to maintain the availability of this architecture.



Part 1 Setup

1.1 - S3 Bucket

The screenshot shows the Amazon S3 console interface. The left sidebar contains navigation options: Buckets, Access management and security, and Storage management and insights. The main content area displays the 'aqib-three-tier-project-bucket' with tabs for Objects, Metadata, Properties, Permissions, Metrics, Management, and Access Points. The 'Objects' tab is active, showing a table of objects. The table has columns for Name, Type, Last modified, Size, and Storage class. The objects listed are 'app-tier/' (Folder), 'nginx.conf' (conf), and 'web-tier/' (Folder). The 'nginx.conf' object is highlighted, showing its last modified date as February 2, 2026, 06:45:02 (UTC+05:30) and its size as 2.6 KB.

Name	Type	Last modified	Size	Storage class
app-tier/	Folder	-	-	-
nginx.conf	conf	February 2, 2026, 06:45:02 (UTC+05:30)	2.6 KB	Standard
web-tier/	Folder	-	-	-

1.2- IAM

The screenshot shows the AWS IAM console interface. The left sidebar contains navigation options: Identity and Access Management (IAM), Access Management, and Access reports. The main content area displays the 'aqib-three-tier-role' with tabs for Summary, Trust relationships, Tags, Last Accessed, and Revoke sessions. The 'Summary' tab is active, showing the role's creation date, last activity, and permissions policies. The 'Permissions policies' section shows two attached policies: 'AmazonS3ReadOnlyAccess' and 'AmazonSSMManagedInstanceCore'. The 'Permissions boundary' section shows that no boundary is set. The 'Generate policy based on CloudTrail events' section provides a link to generate a policy based on access activity.

Policy name	Type	Attached entities
AmazonS3ReadOnlyAccess	AWS managed	1
AmazonSSMManagedInstanceCore	AWS managed	1

Part 2 - Networking and Security

1.1- VPC & Subnets

The screenshot displays the AWS VPC console interface. On the left, there is a navigation sidebar with sections for 'Virtual private cloud' (containing VPCs, Subnets, Route tables, Internet gateways, Egress-only Internet gateways, Carrier gateways, DHCP option sets, Elastic IPs, Managed prefix lists, NAT gateways, Peering connections, and Route servers) and 'Security' (containing Network ACLs and Security groups). The main content area shows the 'Details' tab for the VPC 'vpc-074399afdb0c194f7 / aqib-three-tier-vpc'. The details are organized into four columns: VPC ID (vpc-074399afdb0c194f7), State (Available), Block Public Access (Off), and DNS hostnames (Disabled). Other details include DNS resolution (Enabled), Main network ACL (acl-01c39447ff67ff8), IPv6 CIDR (10.0.0.0/16), DHCP option set (dopt-0f04b304ef5170ae0), Route 53 Resolver DNS Firewall rule groups, and Owner ID (961852293555). Below the details, there is a 'Resource map' section showing a diagram of the VPC resources: a central VPC box connected to a box of 6 subnets (us-east-1a and us-east-1b), which are then connected to a box of 4 route tables (Private-Route-AZ1, Private-Route-AZ2, rtb-0a4ed73c28853f19d, and aqib-route-public), which finally connect to a box of 5 network connections (aqib-IGW, NAT-GW-AZ1, NAT-GW-AZ2, and two connections to other networks).

1.2 - Subnets

The screenshot displays the AWS VPC console interface for the 'Subnets' section. The top part shows a list of 12 subnets with columns for Name, Subnet ID, State, VPC, Block Public Access, IPv4 CIDR, IPv6 CIDR, and IPv6 CIDR association ID. The subnets are organized by VPC and availability zone. The 'aqib-public-web-AZ2' subnet is selected. Below the list, the 'Details' tab for 'subnet-09c0b56436a43c9fd / aqib-public-web-AZ2' is shown. The details are organized into four columns: Subnet ID (subnet-09c0b56436a43c9fd), State (Available), Block Public Access (Off), and IPv6 CIDR association ID (None). Other details include Availability Zone (us-east-1b), Network ACL (acl-01c39447ff67ff8), Auto-assign customer-owned IPv4 address (No), Subnet ARN (arn:aws:ec2:us-east-1:961852293555:subnet/subnet-09c0b56436a43c9fd), Available IPv4 addresses (248), Network border group (us-east-1), Default subnet (No), and Outpost ID (None).

2.1 - IGW & NAT Gateway

Internet gateways (2) Info					
<input type="text" value="Find Internet gateways by attribute or tag"/>					
<input type="checkbox"/>	Name	Internet gateway ID	State	VPC ID	Owner
<input type="checkbox"/>	aqib-IGW	igw-00f9c85037d7848c2	Attached	vpc-074399afdb0c194f7 aqib-three-ti...	961852293555
<input type="checkbox"/>	-	igw-0652c2f1be3c891d0	Attached	vpc-01efa776e27409e66	961852293555

2.2 - NAT

NAT gateways (2) Info										
<input type="text" value="Find NAT gateways by attribute or tag"/>										
<input type="radio"/>	Name	NAT gateway ID	Connectivity...	State	State message	Availability ...	Route table ID	Primary public I...	Primary private I...	Prim
<input type="radio"/>	NAT-GW-AZ1	nat-09ba6e37f7d18d0b9	Public	Available	-	Zonal	-	35.172.122.170	10.0.0.161	eni-C
<input type="radio"/>	NAT-GW-AZ2	nat-0bd99de3f0fa9fd4d	Public	Available	-	Zonal	-	52.205.64.115	10.0.1.186	eni-C

2.3 - Route Tables

Route tables (1/5) Info							
<input type="text" value="Find route tables by attribute or tag"/>							
<input type="checkbox"/>	Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC	Owner ID
<input type="checkbox"/>	Private-Route-AZ1	rtb-00cc8964fcc9e7c3c	subnet-041b60d9dfd8d0d85 / ...	-	No	vpc-074399afdb0c194f7 aqib-...	961852293555
<input type="checkbox"/>	Private-Route-AZ2	rtb-0d392913e203871d8	subnet-03260225338c32bda / ...	-	No	vpc-074399afdb0c194f7 aqib-...	961852293555
<input type="checkbox"/>	-	rtb-0a856fbbaa47885168	-	-	Yes	vpc-01efa776e27409e66	961852293555
<input type="checkbox"/>	-	rtb-0a4ed73c28853f19d	-	-	Yes	vpc-074399afdb0c194f7 aqib-...	961852293555
<input checked="" type="checkbox"/>	aqib-route-public	rtb-0ef38ff847f2cb234	2 subnets	-	No	vpc-074399afdb0c194f7 aqib-...	961852293555

rtb-0ef38ff847f2cb234 / aqib-route-public

Details

Route table ID

rtb-0ef38ff847f2cb234

VPC

vpc-074399afdb0c194f7 | aqib-three-tier-vpc

Main

No

Owner ID

961852293555

Explicit subnet associations

2 subnets

Edge associations

-

rtb-Oef38ff847f2cb234 / aqib-route-public

Find subnet association

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
aqib-public-web-AZ1	subnet-00d5f9348b7420d8a	10.0.0.0/24	-
aqib-public-web-AZ2	subnet-09cdb56436a43c9fd	10.0.1.0/24	-

Subnets without explicit associations (2)

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
aqib-private-DB-AZ2	subnet-0bc36266cf63932bc	10.0.5.0/24	-
aqib-private-DB-AZ1	subnet-0ea12616d67af703b	10.0.4.0/24	-

Edit subnet associations

2.4- Security groups

VPC > Security Groups

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

Security Groups (1/7)

Find security groups by attribute or tag

Actions

Export security groups to CSV

Create security group

	Name	Security group ID	Security group name	VPC ID	Description	Owner
<input type="checkbox"/>	-	sg-05f3a962dd8b56a5c	default	vpc-01efa776e27409e66	default VPC security group	961852293555
<input checked="" type="checkbox"/>	-	sg-0c674dc1baf10b024	aqib-SG-internetfacing	vpc-074399afdb0c194f7	External LB Security Group	961852293555
<input type="checkbox"/>	-	sg-0f7871c47747b57de	web-tier-SG	vpc-074399afdb0c194f7	web tier security group	961852293555
<input type="checkbox"/>	-	sg-02b57c34ebd47e7a4	private-instance-SG	vpc-074399afdb0c194f7	private-instance-SG	961852293555
<input type="checkbox"/>	-	sg-00177c84923a5287e	default	vpc-074399afdb0c194f7	default VPC security group	961852293555
<input type="checkbox"/>	-	sg-0171f62907a5b7fe2	DB-SG	vpc-074399afdb0c194f7	DB-SG	961852293555
<input type="checkbox"/>	-	sg-0488ccb3e3e336929	Internal_LB-SG	vpc-074399afdb0c194f7	Internal_LB-SG	961852293555

sg-0c674dc1baf10b024 - aqib-SG-internetfacing

Details

Inbound rules

Outbound rules

Sharing

VPC associations

Related resources - new

Tags

Details

Security group name

aqib-SG-internetfacing

Owner

961852293555

Security group ID

sg-0c674dc1baf10b024

Inbound rules count

2 Permission entries

Description

External LB Security Group

Outbound rules count

1 Permission entry

VPC ID

vpc-074399afdb0c194f7

Part 3 - DB Deployment

3.1 -Subnet groups

Aurora and RDS > **Subnet groups** > three-tier-sb-subnet-group

Subnet group details

VPC ID
vpc-074399afdb0c194f7 [i](#)

ARN
arn:aws:rds:us-east-1:961852293555:subgrp:three-tier-sb-subnet-group

Supported network types
IPv4

Description
three-tier-sb-subnet-group

Subnets (2)

Availability zone	Subnet name	Subnet ID	CIDR block
us-east-1b	aqib-private-DB-AZ2	subnet-0bc36266cf65932bc i	10.0.5.0/24
us-east-1a	aqib-private-DB-AZ1	subnet-0ea12616d67af703b i	10.0.4.0/24

Tags (0)

[Manage tags](#)

Tags

▲ | Value ▼

3.2 -DB

Aurora and RDS > **Databases** > database-1

database-1

Related

☐ DB identifier ▲ | Status ▼ | Role ▼ | Engine ▼ | Region ... ▼ | Size ▼ | Recommendations ▼ | CPU ▼ | Curren... ▼ | Mainte... ▼

database-1	Available	Regional c...	Aurora My...	us-east-1	2 instances	2 Informational and 2 others	-	-	none
database-1-instance-1	Available	Writer ins...	Aurora My...	us-east-1a	db.r7g.large	1 Informational	<div><div></div></div> 6.25%	<div><div></div></div> 2 Selec	none
database-1-instance-1-us-east-1b	Available	Reader ins...	Aurora My...	us-east-1b	db.r7g.large	1 Informational	<div><div></div></div> 6.33%	<div><div></div></div> 2 Selec	none

Connectivity & security | Monitoring | Logs & events | Configuration | Zero-ETL integrations | Maintenance & backups | Data migrations | Tags | Recommendations

Connect using [info](#)

☒ **Code snippets**
Use when connecting through SDK, APIs, or third-party tools including agents.

☐ **CloudShell**
Use for a quick access to AWS CLI that launches directly from the AWS Management Console.

☐ **Endpoints**
Use when connecting through any IDE interface.

Internet access gateway
☐ Disabled

IAM Authentication
☐ Disabled

Programming language
MySQL (macOS) ▼

Endpoint type
Cluster endpoint ▼

Connect to
Writer ▼

Connection steps
Follow the steps below to paste the code of each step in your tool and run the commands. The snippets dynamically reflect the authentication configuration.

```
1 mysql -h database-1.cluster-coxkg4iei70.us-east-1.rds.amazonaws.com -P 3306 -u admin -p'<Enter_DB_Password>' --ssl-verify-server-cert --ssl-ca=/certs/global-bundle.pem mysql
```

Part 4 - App tier EC2 deployment

4.1 - EC2

The screenshot shows the AWS Management Console for EC2 instances. The left sidebar contains navigation links for EC2, Dashboard, EC2 Global View, Events, Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Capacity Manager, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, Security Groups, Elastic IPs, Placement Groups, Key Pairs, and Network Interfaces.

The main content area displays a list of instances. The instance 'myAppServer1' (ID: i-011d88e1f27a4b55c) is selected. Below the list, the details for this instance are shown, including its state (Running), type (t2.micro), and various addresses.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...
	i-09cc9d8c8a0949cfe	Running	t2.micro	2/2 checks passed	View alarms	us-east-1b	-	-
	i-045c1ce9f360540cd	Running	t2.micro	2/2 checks passed	View alarms	us-east-1b	-	-
	i-0849b09cf8f352ef	Running	t2.micro	2/2 checks passed	View alarms	us-east-1a	-	-
myAppServer1	i-011d88e1f27a4b55c	Running	t2.micro	2/2 checks passed	View alarms	us-east-1a	-	-
	i-0c0a1b9f8048be761	Running	t2.micro	2/2 checks passed	View alarms	us-east-1a	-	-
demoWebServer	i-01927eb6164510e71	Running	t2.micro	2/2 checks passed	View alarms	us-east-1a	-	34.236.249.25...

Details for i-011d88e1f27a4b55c (myAppServer1):

- Instance ID: i-011d88e1f27a4b55c
- IPv6 address: -
- Hostname type: IP name: ip-10-0-2-198.ec2.internal
- Answer private resource DNS name: -
- Auto-assigned IP address: -
- IAM Role: -
- Public IPv4 address: -
- Instance state: Running
- Private IP DNS name (IPv4 only): ip-10-0-2-198.ec2.internal
- Instance type: t2.micro
- VPC ID: vpc-074399afdb0c194f7 (aqib-three-tier-vpc)
- Subnet ID: -
- Private IPv4 addresses: 10.0.2.198
- Public DNS: -
- Elastic IP addresses: -
- AWS Compute Optimizer finding: Opt-in to AWS Compute Optimizer for recommendations. | Learn more
- Auto Scaling Group name: -

Connect to instance

The screenshot shows the 'Connect' page in the AWS Management Console. The 'Session Manager' tab is selected, showing instructions on how to connect to the instance using the Systems Manager just-in-time node access.

Introducing Systems Manager just-in-time node access
Move towards zero standing privileges by requiring operators to request access before remotely connecting to instances. [Learn more](#)

Session Manager usage:

- Connect to your instance without SSH keys, a bastion host, or opening any inbound ports.
- Sessions are secured using an AWS Key Management Service key.
- You can log session commands and details in an Amazon S3 bucket or CloudWatch Logs log group.
- Configure sessions on the Session Manager [Preferences](#) page.

[Cancel](#) [Connect](#)

4.2 - Configure DB

1. Start by downloading the MySQL CLI:

Run

```
sudo wget
```

```
https://dev.mysql.com/get/mysql57-community-release-el7-11.noarch.rpm
```

```
sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2022
```

```
sudo yum install
```

```
https://dev.mysql.com/get/mysql57-community-release-el7-11.noarch.rpm
```

```
sudo yum install mysql -y
```

2. Initiate your DB connection with your Aurora RDS writer endpoint. In the following command, replace the RDS writer endpoint and the username, and then execute it in the browser terminal:

```
mysql -h CHANGE-TO-YOUR-RDS-ENDPOINT -u CHANGE-TO-USER-NAME -p
```

You will then be prompted to type in your password. Once you input the password and hit enter, you should now be connected to your database.

NOTE: If you cannot reach your database, check your credentials and security groups.

3. Create a database called webappdb with the following command using the MySQL CLI:

```
CREATE DATABASE webappdb;
```

You can verify that it was created correctly with the following command:

```
SHOW DATABASES;
```

4. Create a data table by first navigating to the database we just created:

```
USE webappdb;
```


Then, create the following transactions table by executing this create table command:

```
CREATE TABLE IF NOT EXISTS transactions(id INT NOT NULL
AUTO_INCREMENT, amount DECIMAL(10,2), description
VARCHAR(100), PRIMARY KEY(id));
```

Verify the table was created:

```
SHOW TABLES;
```

5. Insert data into table for use/testing later:

```
INSERT INTO transactions (amount,description) VALUES ('400','groceries');
```

Verify that your data was added by executing the following command:

```
SELECT * FROM transactions;
```

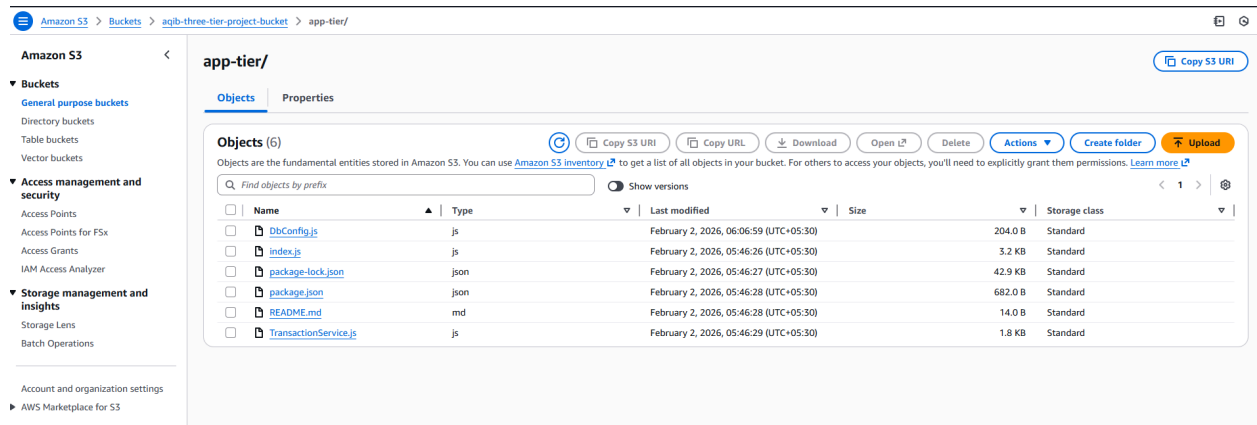
6. When finished, just type exit and hit enter to exit the MySQL client.

4.3 - Configure app Instance

Open the application-code/app-tier/DbConfig.js file from the github repo in your favorite text editor on your computer. You'll see empty strings for the hostname, user, password and database. Fill this in with the credentials you configured for your database, the writer endpoint of your database as the hostname, and webappdb for the database. Save the file.

NOTE: This is NOT considered a best practice, and is done for the simplicity of the lab.

Moving these credentials to a more suitable place like Secrets Manager is left as an extension for this workshop.



Go back to your SSM session. Now we need to install all of the necessary components to run our backend application. Start by installing NVM (node version manager).

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh |
bash
```

```
source ~/.bashrc
```

4. Next, install a compatible version of Node.js and make sure it's being used

```
nvm install 16
```

```
nvm use 16
```

5. PM2 is a daemon process manager that will keep our node.js app running when we exit the instance or if it is rebooted. Install that as well.

```
npm install -g pm2
```

6. Now we need to download our code from our s3 buckets onto our instance. In the command below, replace `BUCKET_NAME` with the name of the bucket you uploaded the `app-tier` folder to:

```
cd ~/
```

```
aws s3 cp s3://BUCKET_NAME/app-tier/ app-tier --recursive
```

7. *Navigate to the app directory, install dependencies, and start the app with pm2.*

```
cd ~/app-tier
```

```
npm install
```

```
pm2 start index.js
```

To make sure the app is running correctly run the following:

```
pm2 list
```

If you see a status of online, the app is running. If you see errored, then you need to do some troubleshooting. To look at the latest errors, use this command:

```
pm2 logs
```

NOTE: If you're having issues, check your configuration file for any typos, and double check that you have followed all installation commands till now.

8. *Right now, pm2 is just making sure our app stays running when we leave the SSM session. However, if the server is interrupted for some reason, we still want the app to start and keep running. This is also important for the AMI we will create:*

```
pm2 startup
```

After running this you will see a message similar to this.

```
[PM2] To setup the Startup Script, copy/paste the following command: sudo env  
PATH=$PATH:/home/ec2-user/.nvm/versions/node/v16.0.0/bin  
/home/ec2-user/.nvm/versions/node/v16.0.0/lib/node_modules/pm2/bin/pm2 startup  
systemd -u ec2-user -hp /home/ec2-user
```

DO NOT run the above command, rather you should copy and past the command in the output you see in your own terminal. After you run it, save the current list of node processes with the following command:

```
pm2 save
```

4.4- Test App Tier

Now let's run a couple tests to see if our app is configured correctly and can retrieve data from the database.

To hit out health check endpoint, copy this command into your SSM terminal. This is our simple health check endpoint that tells us if the app is simply running.

```
curl http://localhost:4000/health
```

The response should looks like the following:

```
"This is the health check"
```

Next, test your database connection. You can do that by hitting the following endpoint locally:

```
curl http://localhost:4000/transaction
```

You should see a response containing the test data we added earlier:

```
{"result":[{"id":1,"amount":400,"description":"groceries"}, {"id":2,"amount":100,"description":"class"}, {"id":3,"amount":200,"description":"other groceries"}, {"id":4,"amount":10,"description":"brownies"}]}
```

If you see both of these responses, then your networking, security, database and app configurations are correct.

Part 5 - Internal Load Balancing and Auto Scaling

5.1 - App Tier AMI

The screenshot shows the Amazon Machine Images (AMIs) console. The left sidebar contains navigation links for EC2, Instances, Images, Elastic Block Store, and Network & Security. The main content area displays a list of AMIs. The 'AppTierImage' is selected, and its details are shown below the list.

Name	AMI name	AMI ID	Source	Owner	Visibility	Status	Creation date
web-server-image	ami-0c2d6f32524fcb2a4	ami-0c2d6f32524fcb2a4	961852293555/web-server-image	961852293555	Private	Available	2026/02/02 07:03 GMT+5:3
AppTierImage	ami-0d16eebe69ebe8219	ami-0d16eebe69ebe8219	961852293555/AppTierImage	961852293555	Private	Available	2026/02/02 06:22 GMT+5:3

AMI ID: ami-0d16eebe69ebe8219

Details | Permissions | Storage | My AMI usage | AMI ancestry - new | Tags

Property	Value
AMI ID	ami-0d16eebe69ebe8219
Image type	machine
Platform details	Linux/UNIX
Root device type	EB5
AMI name	AppTierImage
Owner account ID	961852293555
Architecture	x86_64
Usage operation	RunInstances
Root device name	/dev/xvda
Status	Available
Source	961852293555/AppTierImage
Virtualization type	hvm
Boot mode	uefi-preferred
State reason	-
Creation date	2026-02-02T00:52:23.000Z
Kernel ID	-
Description	App Tier
Product codes	-
RAM disk ID	-
Deprecation time	-
Last launched time	-
Block devices	/dev/xvda=snap-0541aff18a0ad44b:8:true:gp3
Deregistration protection	Disabled
Allowed image	-
Source AMI ID	ami-0532be01f26a3de55
Source AMI Region	us-east-1

5.2 - Target Group

The screenshot shows the Amazon Target Groups console. The left sidebar contains navigation links for Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Capacity Manager, Images, Elastic Block Store, Network & Security, Load Balancing, and Auto Scaling. The main content area displays a list of target groups. The 'App-Tier-TargetGroup' is selected, and its details are shown below the list.

Name	ARN	Port	Protocol	Target type	Load balancer	VPC ID
App-Tier-TargetGroup	arn:aws:elasticloadbalancing:us-east-1:961852293555:targetgroup/App-Tier-TargetGroup/9ab1a01cecb6b6a0	4000	HTTP	Instance	app-tier-internal-LB	vpc-074399afdb0c194f7
web-server-target-group	arn:aws:elasticloadbalancing:us-east-1:961852293555:targetgroup/web-server-target-group/9ab1a01cecb6b6a0	80	HTTP	Instance	web-tier-external-LB	vpc-074399afdb0c194f7

Target group: App-Tier-TargetGroup

Details | Targets | Monitoring | Health checks | Attributes | Tags

Details

arn:aws:elasticloadbalancing:us-east-1:961852293555:targetgroup/App-Tier-TargetGroup/9ab1a01cecb6b6a0

Property	Value
Target type	Instance
Protocol : Port	HTTP: 4000
Protocol version	HTTP1
VPC	vpc-074399afdb0c194f7
IP address type	IPv4
Load balancer	app-tier-internal-LB

Property	Value
Total targets	2
Healthy	2
Unhealthy	0
Unused	0
Initial	0
Draining	0

Distribution of targets by Availability Zone (AZ)

Select values in this table to see corresponding filters applied to the Registered targets table below.

5.3 - Internal LB

EC2 > Load balancers > app-tier-internal-LB

Introducing ALB target optimizer
Target optimizer lets you enforce a maximum number of requests per target using an ALB-provided agent, improving success rates, latency, and efficiency. [Learn more](#)

app-tier-internal-LB

Details

Load balancer type Application	Status Active	VPC vpc-074399a6db0c194f7	Load balancer IP address type IPv4
Scheme Internal	Hosted zone Z355XD0TRQ7X7K	Availability Zones subnet-041b60d9df8d0d85 us-east-1a (use1-az1) subnet-03260225338c32bda us-east-1b (use1-az2)	Date created February 2, 2026, 06:27 (UTC+05:30)
Load balancer ARN arn:aws:elasticloadbalancing:us-east-1:961852293555:loadbalancer/app/app-tier-internal-LB/2d8e362ce6789791		DNS name internal-app-tier-internal-LB-1545315534.us-east-1.elb.amazonaws.com (A Record)	

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

Listeners and rules (1)

A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate	mTLS	Trust store
HTTP:80	Forward to target group App-Tier-TargetGroup, 1 (100%) Target group stickiness: Off	1 rule	ARN	Not applicable	Not applicable	Not applicable	Not applicable

5.4- Launch Template

EC2 > Launch templates > app-tier-launch-temp

app-tier-launch-temp (lt-0b046669b32a937df)

Launch template details

Launch template ID lt-0b046669b32a937df	Launch template name app-tier-launch-temp	Default version 1	Owner arn:aws:iam::961852293555:root
---	---	-----------------------------	--

Details | Versions | Template tags

Launch template version details

Version: 1 (Default)

Description App-Tier-launch-temp	Date created 2026-02-02T00:59:23.000Z	Created by arn:aws:iam::961852293555:root
--	---	---

Instance details | Storage | Resource tags | Network interfaces | Advanced details

AMI ID ami-0d16eeb69ebe8219	Instance type t2.micro	Availability Zone -	Availability Zone Id -
Key pair name -	Security groups -	Security group IDs sg-02b57c34ebd47e7a4	

5.5 - ASG App-Tier

Auto Scaling groups (1/2) Info

Last updated less than a minute ago

Launch configurations

Launch templates

Actions

Create Auto Scaling group

Search your Auto Scaling groups

< 1 >

	Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availability Zones	Creation time
<input type="checkbox"/>	web-sever-ASG	web-server-launch-temp Version Default	2	-	2	2	2	2 Availability Zones	Mon Feb 02 2026...
<input checked="" type="checkbox"/>	app-tier-ASG	app-tier-launch-temp Version Default	2	-	2	2	2	2 Availability Zones	Mon Feb 02 2026...

Auto Scaling group: app-tier-ASG

Details | Integrations | Automatic scaling | Instance management | Instance refresh | Activity | Monitoring | Tags - moved

app-tier-ASG Capacity overview

Edit

arn:aws:autoscaling:us-east-1:961852293555:autoScalingGroup:7b54abd7-9060-46ac-9e66-a9cd7771ca56:autoScalingGroupName/app-tier-ASG

Desired capacity	Scaling limits	Desired capacity type	Status
2	2 - 2	Units (number of instances)	-

Date created
Mon Feb 02 2026 06:33:01 GMT+0530 (India Standard Time)

Launch template

Edit

Launch template	AMI ID	Instance type	Owner
lt-0b046669b32a937df app-tier-launch-temp	ami-0d16eebe69ebe8219	t2.micro	arn:aws:iam::961852293555:root
Version	Security groups	Security group IDs	Create time
Default	-	sg-02b57c34ebd47e7a4	Mon Feb 02 2026 06:29:23 GMT+0530 (India Standard Time)



Part 6: Web Tier Instance Deployment

6.1 - Update Config File

Before we create and configure the web instances, open up the application-code/nginx.conf file from the repo we downloaded. Scroll down to line 58 and replace [INTERNAL-LOADBALANCER-DNS] with your internal load balancer's DNS entry. You can find this by navigating to your internal load balancer's details page.

```
38 server {
39     listen      80;
40     listen      [::]:80;
41     server_name _;
42
43     #health check
44     location /health {
45         default_type text/html;
46         return 200 "<!DOCTYPE html><p>Web Tier Health Check</p>\n";
47     }
48
49     #react app and front end files
50     location / {
51         root    /home/ec2-user/web-tier/build;
52         index index.html index.htm
53         try_files $uri /index.html;
54     }
55
56     #proxy for internal lb
57     location /api/{
58         proxy_pass http://[REPLACE-WITH-INTERNAL-LB-DNS]:80/; 1
59     }
60
61
62 }
63
64 # Settings for a TLS enabled server.
65 #
66 # server {
67 #     listen      443 ssl http2;
68 #     listen      [::]:443 ssl http2;
69 #     server_name _;
70 #     root        /usr/share/nginx/html;
71 #
```

Then, upload this file and the application-code/web-tier folder to the s3 bucket you created for this lab.

6.2 - Web Instance Deployment & connect to Instance

The screenshot displays the AWS Management Console interface for EC2 instances. At the top, there's a header for 'Instances (1/6)' with a search bar and filters. Below this is a table listing several instances, including 'demoWebServer' which is highlighted. The table columns include Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, Public IPv4 DNS, and Public IPv4 ...

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...
	i-09cc9d8c8a0949cfe	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	-	-
	i-045c1ce9f360540cd	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	-	-
	i-0849b09cf8f352ef	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	-	-
myAppServer1	i-011d88e1f27a4b55c	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	-	-
demoWebServer	i-01927eb6164510e71	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	-	34.236.249.254

Below the table, the details for the selected instance 'i-01927eb6164510e71 (demoWebServer)' are shown. The 'Details' tab is active, displaying various attributes:

- Instance ID:** i-01927eb6164510e71
- IPv6 address:** -
- Hostname type:** IP name: ip-10-0-0-217.ec2.internal
- Answer private resource DNS name:** -
- Auto-assigned IP address:** 34.236.249.254 (Public IP)
- IAM Role:** aqib-three-tier-role
- Public IPv4 address:** 34.236.249.254 | open address
- Instance state:** Running
- Private IP DNS name (IPv4 only):** ip-10-0-0-217.ec2.internal
- Instance type:** t2.micro
- VPC ID:** vpc-074399afdb0c194f7 (aqib-three-tier-vpc)
- Subnet ID:** subnet-00d5f9348b7420d8a (aqib-public-web-AZ1)
- Private IPv4 addresses:** 10.0.0.217
- Public DNS:** -
- Elastic IP addresses:** -
- AWS Compute Optimizer finding:** Opt-in to AWS Compute Optimizer for recommendations. | Learn more
- Auto Scaling Group name:** -

6.3 - Configure Web Instance

1. We now need to install all of the necessary components needed to run our front-end application. Again, start by installing NVM and node :

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh | bash
source ~/.bashrc
nvm install 16
nvm use 16
```

2. Now we need to download our web tier code from our s3 bucket:

```
cd ~/
aws s3 cp s3://BUCKET_NAME/web-tier/ web-tier --recursive
```

Navigate to the web-layer folder and create the build folder for the react app so we can serve our code:

```
cd ~/web-tier
npm install
npm run build
```

3. NGINX can be used for different use cases like load balancing, content caching etc, but we will be using it as a web server that we will configure to serve our application on port 80, as well as help direct our API calls to the internal load balancer.

```
sudo amazon-linux-extras install nginx1 -y
```

4. We will now have to configure NGINX. Navigate to the Nginx configuration file with the following commands and list the files in the directory:

```
cd /etc/nginx
ls
```

You should see an nginx.conf file. We're going to delete this file and use the one we uploaded to s3. Replace the bucket name in the command below with the one you created for this workshop:

```
sudo rm nginx.conf
sudo aws s3 cp s3://BUCKET_NAME/nginx.conf .
```

Then, restart Nginx with the following command:

```
sudo service nginx restart
```

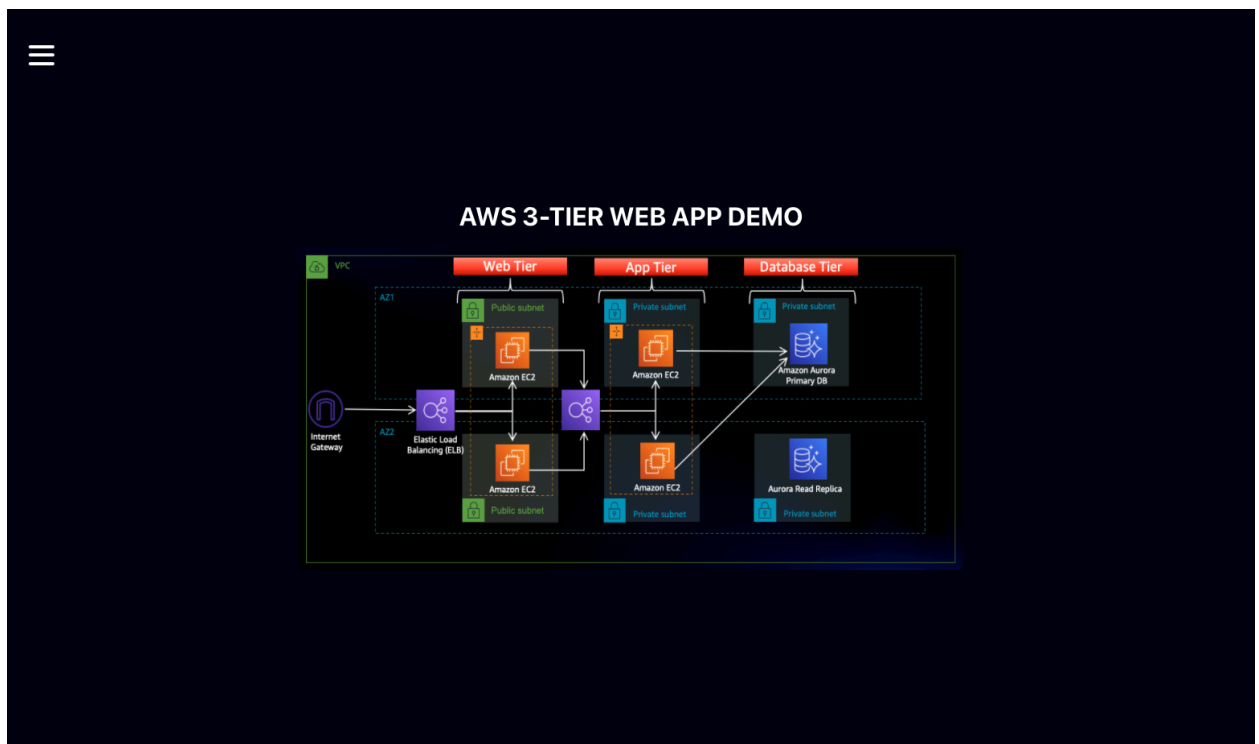
To make sure Nginx has permission to access our files execute this command:

```
chmod -R 755 /home/ec2-user
```

And then to make sure the service starts on boot, run this command:

```
sudo chkconfig nginx on
```

5. Now when you plug in the public IP of your web tier instance, you should see your website, which you can find on the Instance details page on the EC2 dashboard. If you have the database connected and working correctly, then you will also see the database working. You'll be able to add data. Careful with the delete button, that will clear all the entries in your database.



Part 7: External Load Balancer and Auto Scaling

7.1 - Web Tier AMI

Amazon Machine Images (AMIs) (1/2) [Info](#)

Owned by me

[Recycle Bin](#) [EC2 Image Builder](#) [Actions](#) [Launch instance from AMI](#)

<input checked="" type="checkbox"/>	Name	AMI name	AMI ID	Source	Owner	Visibility	Status	Creation date
<input checked="" type="checkbox"/>	web-server-image	ami-0c2d6f32524fcb2a4	ami-0c2d6f32524fcb2a4	961852293555/web-server-image	961852293555	Private	Available	2026/02/02 07:03 GMT+5:30
<input type="checkbox"/>	AppTierImage	ami-0d16eebe69ebe8219	ami-0d16eebe69ebe8219	961852293555/AppTierImage	961852293555	Private	Available	2026/02/02 06:22 GMT+5:30

AMI ID: ami-0c2d6f32524fcb2a4

[Details](#) [Permissions](#) [Storage](#) [My AMI usage](#) [AMI ancestry - new](#) [Tags](#)

AMI ID ami-0c2d6f32524fcb2a4	Image type machine	Platform details Linux/UNIX	Root device type EBS
AMI name web-server-image	Owner account ID 961852293555	Architecture x86_64	Usage operation RunInstances
Root device name /dev/xvda	Status Available	Source 961852293555/web-server-image	Virtualization type hvm
Boot mode uefi-preferred	State reason -	Creation date 2026-02-02T01:33:55.000Z	Kernel ID -
Description web-server-image	Product codes -	RAM disk ID -	Deprecation time -
Last launched time -	Block devices /dev/xvda=snap-0d8121b5241330733:8:true:gp3	Deregistration protection Disabled	Allowed image -
Source AMI ID ami-0532be01f26a3de55	Source AMI Region us-east-1		

7.2 - TG

Target groups (1/2) [Info](#) [What's new?](#)

[Actions](#) [Create target group](#)

<input type="checkbox"/>	Name	ARN	Port	Protocol	Target type	Load balancer	VPC ID
<input type="checkbox"/>	App-Tier-TargetGroup	arn:aws:elasticloadbalancing:us-east-1:961852293555:targetgroup/web-server-target-group/2354d4f9fc4833b4	4000	HTTP	Instance	app-tier-internal-LB	vpc-074399afdb0c194f7
<input checked="" type="checkbox"/>	web-server-target-group	arn:aws:elasticloadbalancing:us-east-1:961852293555:targetgroup/web-server-target-group/2354d4f9fc4833b4	80	HTTP	Instance	web-tier-external-LB	vpc-074399afdb0c194f7

Target group: web-server-target-group

[Details](#) [Targets](#) [Monitoring](#) [Health checks](#) [Attributes](#) [Tags](#)

Details
arn:aws:elasticloadbalancing:us-east-1:961852293555:targetgroup/web-server-target-group/2354d4f9fc4833b4

Target type Instance	Protocol : Port HTTP: 80	Protocol version HTTP1	VPC vpc-074399afdb0c194f7
IP address type IPv4	Load balancer web-tier-external-LB		

2 Total targets	2 Healthy 0 Anomalous	0 Unhealthy	0 Unused	0 Initial	0 Draining
--------------------	-----------------------------	----------------	-------------	--------------	---------------

► **Distribution of targets by Availability Zone (AZ)**
Select values in this table to see corresponding filters applied to the Registered targets table below.

7.3 - Internet Facing LB

Load balancers (1/2) [What's new?](#)

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

<input type="checkbox"/>	Name	State	Type	Scheme	IP address type	VPC ID	Availability Zones	Security groups	DNS name
<input type="checkbox"/>	app-tier-internal-LB	Active	application	Internal	IPv4	vpc-074399afdb0c194f7	2 Availability Zones	sg-0488ccb3e3e336939	internal-app-tie
<input checked="" type="checkbox"/>	web-tier-external-LB	Active	application	Internet-facing	IPv4	vpc-074399afdb0c194f7	2 Availability Zones	sg-0c674dc1ba1f0b024	web-tier-extern

Load balancer: web-tier-external-LB

Details

Listeners and rules

Network mapping

Resource map

Security

Monitoring

Integrations

Attributes

Capacity

Tags

Details

Load balancer type

Application

Scheme

Internet-facing

Load balancer ARN

arn:aws:elasticloadbalancing:us-east-1:961852293555:loadbalancer/app/web-tier-external-LB/8cbcd9f0dd842966

Status

Active

Hosted zone

Z355XDOTRQ7X7K

VPC

vpc-074399afdb0c194f7

Availability Zones

subnet-00d5f9348b7420d8a us-east-1a (use1-az1)
subnet-09cdb56436a43c9fd us-east-1b (use1-az2)

Load balancer IP address type

IPv4

Date created

February 2, 2026, 07:06 (UTC+05:30)

DNS name

web-tier-external-LB-1122120879.us-east-1.elb.amazonaws.com (A Record)

7.4 - Launch Template

Launch Templates (1/2) [info](#)

<input type="checkbox"/>	Launch Template ID	Launch Template Name	Default Version	Latest Version	Create Time	Created By	Managed	Operator
<input checked="" type="checkbox"/>	lt-0c9fedcf7b4392db4	web-server-launch-temp	1	1	2026-02-02T01:39:40.000Z	arn:aws:iam::961852293555:root	false	-
<input type="checkbox"/>	lt-0b046669b32a937df	app-tier-launch-temp	1	1	2026-02-02T00:59:23.000Z	arn:aws:iam::961852293555:root	false	-

web-server-launch-temp (lt-0c9fedcf7b4392db4)

Launch template details

Launch template ID

lt-0c9fedcf7b4392db4

Launch template name

web-server-launch-temp

Default version

1

Owner

arn:aws:iam::961852293555:root

Details

Versions

Template tags

Launch template version details

Version

1 (Default)

Description

web-server-launch-temp

Date created

2026-02-02T01:39:40.000Z

Created by

arn:aws:iam::961852293555:root

Instance details

Storage

Resource tags

Network interfaces

Advanced details

AMI ID

ami-0c2d6f32524fcb2a4

Instance type

t2.micro

Availability Zone

-

Availability Zone Id

-

Key pair name

-

Security groups

-

Security group IDs

sg-0f7871c47747b57de

7.5 - ASG

Auto Scaling groups (1/2) Info

Last updated 8 minutes ago

Launch configurations

Launch templates

Actions

Create Auto Scaling group

Search your Auto Scaling groups

	Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availability Zones	Creation time
<input checked="" type="checkbox"/>	web-sever-ASG	web-server-launch-temp Version Default	2	-	2	2	2	2 Availability Zones	Mon Feb 02 2026...
<input type="checkbox"/>	app-tier-ASG	app-tier-launch-temp Version Default	2	-	2	2	2	2 Availability Zones	Mon Feb 02 2026...

Auto Scaling group: web-sever-ASG

Details

Integrations

Automatic scaling

Instance management

Instance refresh

Activity

Monitoring

Tags - moved

web-sever-ASG Capacity overview

Edit

arn:aws:autoscaling:us-east-1:961852293555:autoScalingGroup:f6535de1-1e1d-4284-af6b-dafd33d0d8b1:autoScalingGroupName/web-sever-ASG

Desired capacity	Scaling limits	Desired capacity type	Status
2	2 - 2	Units (number of instances)	-

Date created
Mon Feb 02 2026 07:11:37 GMT+0530 (India Standard Time)

Launch template

Edit

Launch template

lt-0c9fedcf7b4392db4
web-server-launch-temp

AMI ID

ami-0c2d6f32524fcb2a4

Instance type

t2.micro

Owner

arn:aws:iam::961852293555:root

Version

Default

Security groups

-

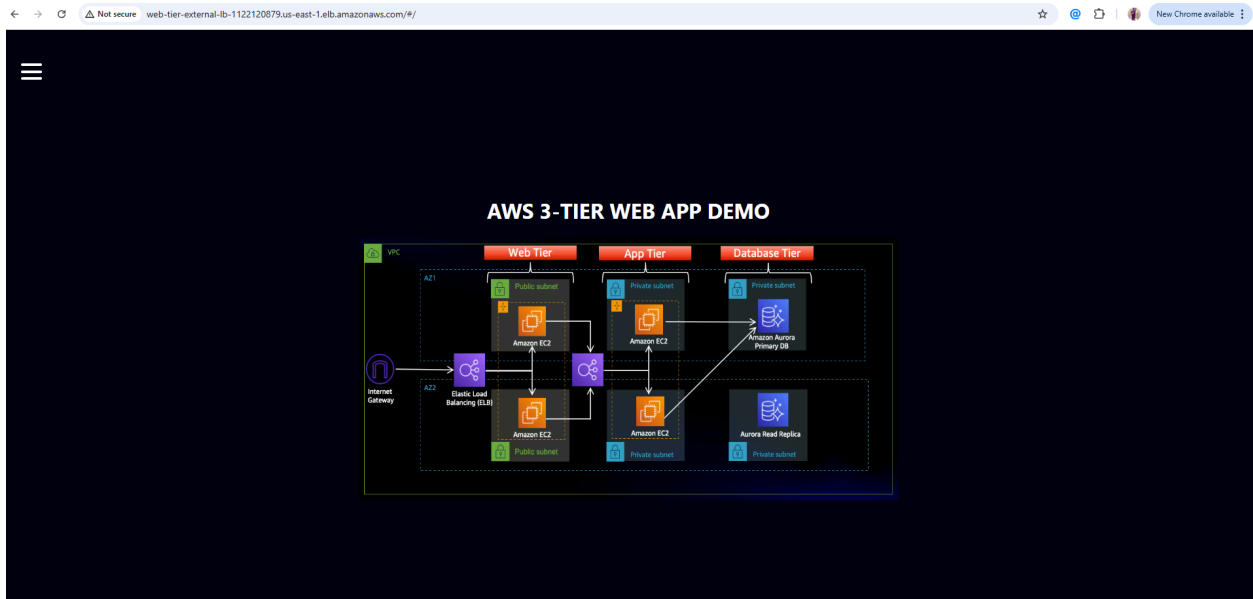
Security group IDs

sg-0f7871c47747b57de

Create time

Mon Feb 02 2026 07:09:40 GMT+0530 (India Standard Time)

Load Balancer URL working fine the website is up and running



← → Not secure web-tier-external-lb-1122120879.us-east-1.elb.amazonaws.com/#/db

✕

AURORA DATABASE DEMO PAGE

DEL

ID	AMOUNT	DESC
ADD	<input type="text"/>	<input type="text"/>
1	400	groceries
2	200	test

HOME

DB DEMO

