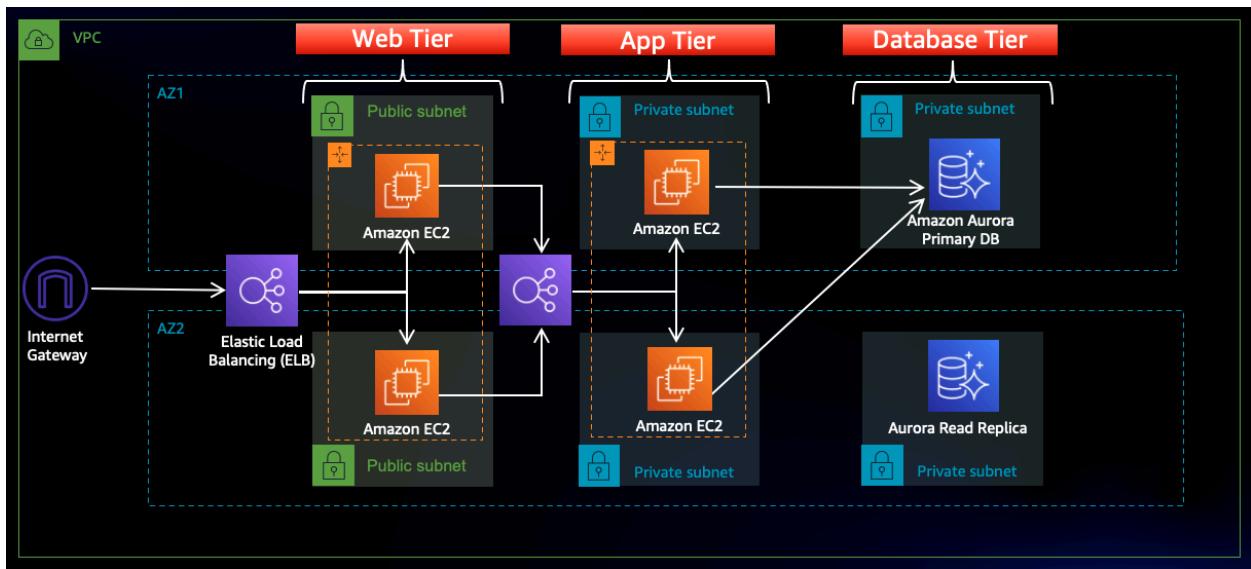


AWS Three-Tier Web Architecture Project

In this architecture, a public-facing Application Load Balancer forwards client traffic to our web tier EC2 instances. The web tier is running Nginx webservers 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 AWS S3 console with the path: Amazon S3 > Buckets > aqib-three-tier-project-bucket. The left sidebar includes sections for Buckets, Access management and security, Storage management and insights, and Account and organization settings. The main content area displays the 'aqib-three-tier-project-bucket' details, including its ARN (arn:aws:s3:::aqib-three-tier-project-bucket) and a table of objects. The table has columns for Name, Type, Last modified, Size, and Storage class. It lists three objects: 'app-tier/' (Folder), 'nginx.conf' (conf), and 'web-tier/' (Folder). The 'nginx.conf' file was last modified on February 2, 2026, at 06:45:02 (UTC+05:30) and is 2.6 KB in size, stored in the Standard storage class.

1.2- IAM

The screenshot shows the AWS IAM console with the path: IAM > Roles > aqib-three-tier-role. The left sidebar includes sections for Identity and Access Management (IAM), Access Management, Access reports, and Credential report. The main content area displays the 'aqib-three-tier-role' details, including its ARN (arn:aws:iam::961852293555:role/aqib-three-tier-role), creation date (February 1, 2026, 05:03 (UTC+05:30)), and last activity (9 minutes ago). It also shows the instance profile ARN (arn:aws:iam::961852293555:instance-profile/aqib-three-tier-role). The 'Permissions' tab is selected, showing the 'Permissions policies' section with two managed policies attached: 'AmazonS3ReadOnlyAccess' and 'AmazonSSMManagedInstanceCore'. The 'Permissions boundary (not set)' and 'Generate policy based on CloudTrail events' sections are also visible.

Part 2 - Networking and Security

1.1- VPC & Subnets

The screenshot shows the AWS VPC dashboard for a specific VPC named "vpc-074399afdb0c194f7 / aqib-three-tier-vpc".

Details:

- VPC ID: vpc-074399afdb0c194f7
- State: Available
- Tenancy: default
- Main network ACL: acl-01c59447ff67ff8
- IPv6 CIDR: 10.0.0.16
- Route 53 Resolver DNS Firewall rule groups: -
- Owner ID: 961852293555

Resource map:

- VPC:** Your AWS virtual network (aqib-three-tier-vpc)
- Subnets (6):** Subnets within this VPC
 - us-east-1a
 - aqib-public-web-AZ1
 - aqib-private-AZ1
 - us-east-1b
 - aqib-public-web-AZ2
 - aqib-private-DB-AZ2
 - aqib-private-AZ2
- Route tables (4):** Route network traffic to resources
 - Private-Route-AZ1
 - Private-Route-AZ2
 - rtb-0a4ef75c28853f19d
 - aqib-route-public
- Network Connections (3):** Connections to other networks
 - aqib-GW
 - NAT-GW-AZ1
 - NAT-GW-AZ2

1.2 - Subnets

The screenshot shows the AWS Subnets page for a VPC.

Subnets (1/12) Info:

Name	Subnet ID	State	VPC	Block Public Access	IPv4 CIDR	IPv6 CIDR	IPv4 CIDR association ID	Aval
-	subnet-03a9012e0cfdf4c6a	Available	vpc-01efa776e27409e66	Off	172.31.48.0/20	-	-	409'
-	subnet-06ff0249da875bf0f	Available	vpc-01efa776e27409e66	Off	172.31.64.0/20	-	-	409'
aqib-public-web-AZ1	subnet-0045f934ab7420d8a	Available	vpc-074399afdb0c194f7 aqib...	Off	10.0.0.0/24	-	-	247
aqib-private-AZ1	subnet-041be0d0ff80c8d85	Available	vpc-074399afdb0c194f7 aqib...	Off	10.0.2.0/24	-	-	248
aqib-private-DB-AZ2	subnet-0b3626d6f63932b6	Available	vpc-074399afdb0c194f7 aqib...	Off	10.0.5.0/24	-	-	250
-	subnet-0b077b11c1fec7d6c	Available	vpc-01efa776e27409e66	Off	172.31.0.0/20	-	-	409'
-	subnet-03260225339c372bd0	Available	vpc-074399afdb0c194f7 aqib...	Off	10.0.3.0/24	-	-	249
aqib-private-AZ2	subnet-043db1fc98fb2ad6b	Available	vpc-01efa776e27409e66	Off	172.31.32.0/20	-	-	409'
-	subnet-0b7841cf2ab779d	Available	vpc-01efa776e27409e66	Off	172.31.80.0/20	-	-	409'
aqib-public-web-AZ2	subnet-09cd56436a43c9fd	Available	vpc-074399afdb0c194f7 aqib...	Off	10.0.1.0/24	-	-	248
-	subnet-00bb0b0c28920009	Available	vpc-01efa776e27409e66	Off	172.31.16.0/20	-	-	409'
aqib-private-DB-AZ1	subnet-0ea1261d67af703b	Available	vpc-074399afdb0c194f7 aqib...	Off	10.0.4.0/24	-	-	250

subnet-09cd56436a43c9fd / aqib-public-web-AZ2:

Details:

Subnet ID: subnet-09cd56436a43c9fd	Subnet ARN: arn:aws:ec2:us-east-1:961852293555:subnet/subnet-09cd56436a43c9fd	State: Available
IPv4 CIDR: 10.0.1.0/24	Available IPv4 addresses: 248	IPv6 CIDR: -
Availability Zone: us-east-1a (us-east-1b)	Network border group: us-east-1	VPC: vpc-074399afdb0c194f7 aqib-three-tier-vpc
Network ACL: acl-01c59447ff67ff8	Default subnet: No	Auto-assign public IPv4 address: No
Auto-assign customer-owned IPv4 address: -	Outpost ID: -	Block Public Access: Off
IPv6 CIDR association ID: -	Route table: rtb-0ef38ff847f2cb234 aqib-route-public	IPv4 CIDR reservations: -
Auto-assign IPv6 address: No	Auto-assign IPv6 address: No	IPv4 CIDR reservations: -

2.1 - IGW & NAT Gateway

Internet gateways (2) Info						
Name		Internet gateway ID	State	VPC ID	Owner	
<input type="checkbox"/>	aqib-IGW	igw-00f9c85037d7848c2	Attached	vpc-074399afdb0c194f7 aqib-three-tier-vpc	961852293555	
<input type="checkbox"/>	-	igw-0652c2f1be3c891d0	Attached	vpc-01efa776e27409e66	961852293555	

2.2 - NAT

NAT gateways (2) Info											
Find NAT gateways by attribute or tag											
Name	NAT gateway ID	Connectivity...	State	State message	Availability...	Route table ID	Primary public I...	Primary private I...	Prim...	Public IP	Private IP
NAT-GW-AZ1	nat-09ba6e37f7d18d0b9	Public	Available	-	Zonal	-	35.172.122.170	10.0.0.161	eni-C	172.17.0.1	en-i-C
NAT-GW-AZ2	nat-0bd99de3f0fa9fd4d	Public	Available	-	Zonal	-	52.205.64.115	10.0.1.186	eni-C	172.17.0.2	en-i-C

2.3 - Route Tables

Route tables (1/5) Info									
Find route tables by attribute or tag									
Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC	Owner ID	Last updated	Actions	Create route table
Private-Route-AZ1	rtb-00c8964fc9e7c3c	subnet-041b60d9fdb0d085 / ...	-	No	vpc-074399afdb0c194f7 aqib-three-tier-vpc	961852293555	5 minutes ago		
Private-Route-AZ2	rtb-0d392913c203871d8	subnet-03260225338c32bda / ...	-	No	vpc-074399afdb0c194f7 aqib-three-tier-vpc	961852293555			
-	rtb-0a856fbfaa47885168	-	-	Yes	vpc-01efa776e27409e66	961852293555			
-	rtb-0a4ed73c28853f19d	-	-	Yes	vpc-074399afdb0c194f7 aqib-three-tier-vpc	961852293555			
aqib-route-public	rtb-0ef38ff847f2cb234	2 subnets	-	No	vpc-074399afdb0c194f7 aqib-three-tier-vpc	961852293555			

rtb-0ef38ff847f2cb234 / aqib-route-public

[Details](#) | [Routes](#) | [Subnet associations](#) | [Edge associations](#) | [Route propagation](#) | [Tags](#)

Details

Route table ID rtb-0ef38ff847f2cb234	Main No	Explicit subnet associations 2 subnets	Edge associations -
VPC vpc-074399afdb0c194f7 aqib-three-tier-vpc	Owner ID 961852293555		

rtb-0ef38ff847f2cb234 / aqib-route-public				
<input type="text" value="Find subnet association"/> < 1 > <input checked="" type="checkbox"/> <input type="button" value="Edit subnet associations"/>				
Name	▼	Subnet ID	▼	IPv4 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)				
<input type="text" value="Find subnet association"/> < 1 > <input checked="" type="checkbox"/> <input type="button" value="Edit subnet associations"/>				
Name	▼	Subnet ID	▼	IPv4 CIDR
aqib-private-DB-AZ2		subnet-0bc36266cf63932bc		10.0.5.0/24
aqib-private-DB-AZ1		subnet-0ea12616d67af703b		10.0.4.0/24

2.4- Security groups

VPC dashboard < AWS Global View < Security Groups

Security Groups (1/7) Info

<input type="text" value="Find security groups by attribute or tag"/> Actions <input type="button" value="Export security groups to CSV"/> <input type="button" value="Create security group"/>						
Name	Security group ID	Security group name	VPC ID	Description	Owner	
-	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	
-	sg-0f7871c47747b57de	web-tier-SG	vpc-074399afdb0c194f7	web tier security group	961852293555	
-	sg-02b57c34eb4d47e7a4	private-instance-SG	vpc-074399afdb0c194f7	private-instance-SG	961852293555	
-	sg-00177c84923a5287e	default	vpc-074399afdb0c194f7	default VPC security group	961852293555	
-	sg-0171f62907a5b7fe2	DB-SG	vpc-074399afdb0c194f7	DB-SG	961852293555	
-	sg-0488ccb3e5e336939	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 <input type="checkbox"/> aqib-SG-internetfacing	Security group ID <input type="checkbox"/> sg-0c674dc1baf10b024	Description <input type="checkbox"/> External LB Security Group	VPC ID <input type="checkbox"/> vpc-074399afdb0c194f7			
Owner <input type="checkbox"/> 961852293555	Inbound rules count <input type="checkbox"/> 2 Permission entries	Outbound rules count <input type="checkbox"/> 1 Permission entry				

Part 3 - DB Deployment

3.1 -Subnet groups

The screenshot shows the AWS Aurora and RDS Subnet groups page. The left sidebar includes options like Dashboard, Databases, Query editor, Performance insights, Snapshots, Exports in Amazon S3, Automated backups, Reserved instances, Proxies, Subnet groups, Parameter groups, Option groups, Custom engine versions, Zero-ETL integrations, Events, Event subscriptions, Recommendations, and Certificate update.

The main content area displays the details of a subnet group named "three-tier-sb-subnet-group". It shows the VPC ID (vpc-074399afdb0c194f7), ARN (arn:aws:rds:us-east-1:961852293555:subgrp:three-tier-sb-subnet-group), Supported network types (IPv4), and a Description (three-tier-sb-subnet-group).

A table lists two subnets:

Availability zone	Subnet name	Subnet ID	CIDR block
us-east-1b	aqib-private-DB-AZ2	subnet-0bc36266cf63932bc	10.0.5.0/24
us-east-1a	aqib-private-DB-AZ1	subnet-0ea12616d67af703b	10.0.4.0/24

Below the table is a section for Tags (0) with a search bar and a "Manage tags" button.

3.2 -DB

The screenshot shows the AWS Aurora and RDS Databases page. The left sidebar includes options like Dashboard, Databases, Query editor, Performance insights, Snapshots, Exports in Amazon S3, Automated backups, Reserved instances, Proxies, Subnet groups, Parameter groups, Option groups, Custom engine versions, Zero-ETL integrations, Events, Event subscriptions, Recommendations, and Certificate update.

The main content area displays the details of a database named "database-1". It shows the DB identifier (database-1), Status (Available), Role (Regional), Engine (Aurora MySQL), Region (us-east-1), Size (2 instances), Recommendations (2 informational and 2 others), CPU usage (6.25%), Current (2 Select), and Maintenance (none).

Below the database details, there are tabs for Connectivity & security, Monitoring, Logs & events, Configuration, Zero-ETL integrations, Maintenance & backups, Data migrations, Tags, and Recommendations.

The Connectivity & security section includes sections for Connect using (Code snippets, CloudShell, Endpoints), Internet access gateway (disabled), IAM Authentication (disabled), Programming language (MySQL (macOS)), Endpoint type (Cluster endpoint), and Connect to (Writer).

At the bottom, there is a Connection steps section with a code snippet:

```
1 mysql -h database-1.cluster-cokxkg4lei170.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 EC2 Instances page. On the left, there's a sidebar with navigation links for EC2 services like Dashboard, 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 area displays a table of instances with columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, Public IPv4 DNS, and Public IPv4 ... (partial). One instance, 'myAppServer1' (instance ID i-011d88e1f27a4b55c), is selected and highlighted in blue. Below the table, a detailed view for 'myAppServer1' is shown with tabs for Details, Status and alarms, Monitoring, Security, Networking, Storage, and Tags. The Details tab is active, showing fields like Instance ID (i-011d88e1f27a4b55c), Instance state (Running), Public IPv4 address (10.0.2.198), Private IP DNS name (ip-10-0-2-198.ec2.internal), Instance type (t2.micro), VPC ID (vpc-074399afdb0c194f7), and Subnet ID.

Connect to instance

The screenshot shows the 'Connect' page for an EC2 instance. At the top, there are tabs for EC2 Instance Connect, Session Manager, SSH client, and EC2 serial console. A note below the tabs states: '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](#)' with a 'Try for free' button. Below this, a section titled 'Session Manager usage:' lists the following steps: '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.', and 'Configure sessions on the Session Manager [Preferences](#) page.' At the bottom right are 'Cancel' and 'Connect' buttons.

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.

Name	Type	Last modified	Size	Storage class
DbConfig.js	js	February 2, 2026, 06:06:59 (UTC+05:30)	204.0 B	Standard
index.js	js	February 2, 2026, 05:46:26 (UTC+05:30)	3.2 KB	Standard
package-lock.json	json	February 2, 2026, 05:46:27 (UTC+05:30)	42.9 KB	Standard
package.json	json	February 2, 2026, 05:46:28 (UTC+05:30)	682.0 B	Standard
README.md	md	February 2, 2026, 05:46:28 (UTC+05:30)	14.0 B	Standard
TransactionService.js	js	February 2, 2026, 05:46:29 (UTC+05:30)	1.8 KB	Standard

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 AWS EC2 AMIs page. On the left, there's a navigation sidebar with links like Dashboard, EC2 Global View, Events, Instances, Images, Elastic Block Store, Network & Security, and Auto Scaling. The main area displays a table titled "Amazon Machine Images (AMIs) (1/2) Info". It lists two AMIs:

Name	AMI ID	Source	Owner	Visibility	Status	Creation date
web-server-image	ami-02d6f32524fc2b24	961852293555/web-server-image	961852293555	Private	Available	2026/02/02 07:03 GMT+5:30
AppTierImage	ami-0d16eebe69ebe8219	961852293555/AppTierImage	961852293555	Private	Available	2026/02/02 06:22 GMT+5:30

Below the table, a specific AMI is selected: "AMI ID: ami-0d16eebe69ebe8219". The "Details" tab is active, showing various configuration details:

AMI ID	Image type	Platform details	Root device type
ami-0d16eebe69ebe8219	machine	Linux/UNIX	EBS
AMI name	Owner account ID	Architecture	Usage operation
AppTierImage	961852293555	x86_64	RunInstances
Root device name	Status	Source	Virtualization type
/dev/xvda	Available	961852293555/AppTierImage	hvm
Boot mode	State reason	Creation date	Kernel ID
uefi-preferred	-	2026-02-02T00:52:23.000Z	-
Description	Product codes	RAM disk ID	Deprecation time
App Tier	-	-	-
Last launched time	Block devices	Deregistration protection	Allowed image
-	/dev/xvda:/snap-0541afff18a0ad44b8:truegp3	Disabled	-
Source AMI ID	Source AMI Region		
ami-0532be01f26a3de55	us-east-1		

5.2 - Target Group

The screenshot shows the AWS EC2 Target groups page. The left sidebar includes links for Images, Elastic Block Store, Network & Security, and Auto Scaling. The main area displays a table titled "Target groups (1/2) Info | What's new?". It lists two target groups:

Name	ARN	Port	Protocol	Target type	Load balancer	VPC ID
App-Tier-TargetGroup	arnaws:elasticloadbalancing:us-east-1:961852293555:targetgroup/App-Tier-TargetGroup/9ab1a01ceb6b6a0	4000	HTTP	Instance	app-tier-internal-LB	vpc-074399afdb0c194f7
web-server-target-group	arnaws:elasticloadbalancing:us-east-1:961852293555:targetgroup/web-server-target-group/12345678901234567890	80	HTTP	Instance	web-tier-external-LB	vpc-074399afdb0c194f7

Below the table, a specific target group is selected: "Target group: App-Tier-TargetGroup". The "Details" tab is active, showing its configuration:

Target type	Protocol: Port	Protocol version	VPC
Instance	HTTP: 4000	HTTP/1	vpc-074399afdb0c194f7
IP address type	Load balancer app-tier-internal-LB		
IPv4			

At the bottom, a summary table shows target status across availability zones:

Total targets	Healthy	Unhealthy	Unused	Initial	Draining
2	2	0	0	0	0
	Healthy	Unhealthy	Unused	Initial	Draining
	0 anomalous				

A note at the bottom says: "► 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

The screenshot shows the AWS CloudFormation console with the following details:

Details Tab:

- Load balancer type: Application
- Status: Active
- VPC: vpc-074399afdb0c194f7
- Hosted zone: Z355XDOTRQ7X7K
- Availability Zones:
 - subnet-041b60d9df8d00d85 (us-east-1a)
 - subnet-03260225338c32bda (us-east-1b)
- Load balancer IP address type: IPv4
- Date created: February 2, 2026, 06:27 (UTC+05:30)

Listeners and rules Tab:

- Protocol: HTTP-80
- Default action: Forward to target group (App-Tier-TargetGroup) (100%)
- Target group stickiness: Off

5.4- Launch Template

The screenshot shows the AWS CloudFormation console with the following details:

Launch template details:

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

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:

- AMI ID: ami-0d16eebe69ebe8219
- Instance type: t2.micro
- Availability Zone: -
- 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](#)

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availability Zones	Creation time
web-server-ASG	web-server-launch-temp Version Default	2	-	2	2	2	2 Availability Zones	Mon Feb 02 2026...
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

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

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 06:29:25 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 shows the AWS EC2 Instances page. At the top, there's a search bar and a filter dropdown set to 'All states'. Below the header are several buttons: 'Last updated less than a minute ago', 'Connect', 'Instance state', 'Actions', and 'Launch instances'. A navigation bar with arrows and a refresh icon is at the bottom.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...
i-09cc9d8c8a0949cf	i-09cc9d8c8a0949cf	Running	t2.micro	2/2 checks passed	View alarms	us-east-1b	-	-
i-045c1ce9f560540cd	i-045c1ce9f560540cd	Running	t2.micro	2/2 checks passed	View alarms	us-east-1b	-	-
i-0849b09fcf8f352ef	i-0849b09fcf8f352ef	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	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	-	34.236.249.254

i-01927eb6164510e71 (demoWebServer)

Details | Status and alarms | Monitoring | Security | Networking | Storage | Tags

Instance summary

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

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
```

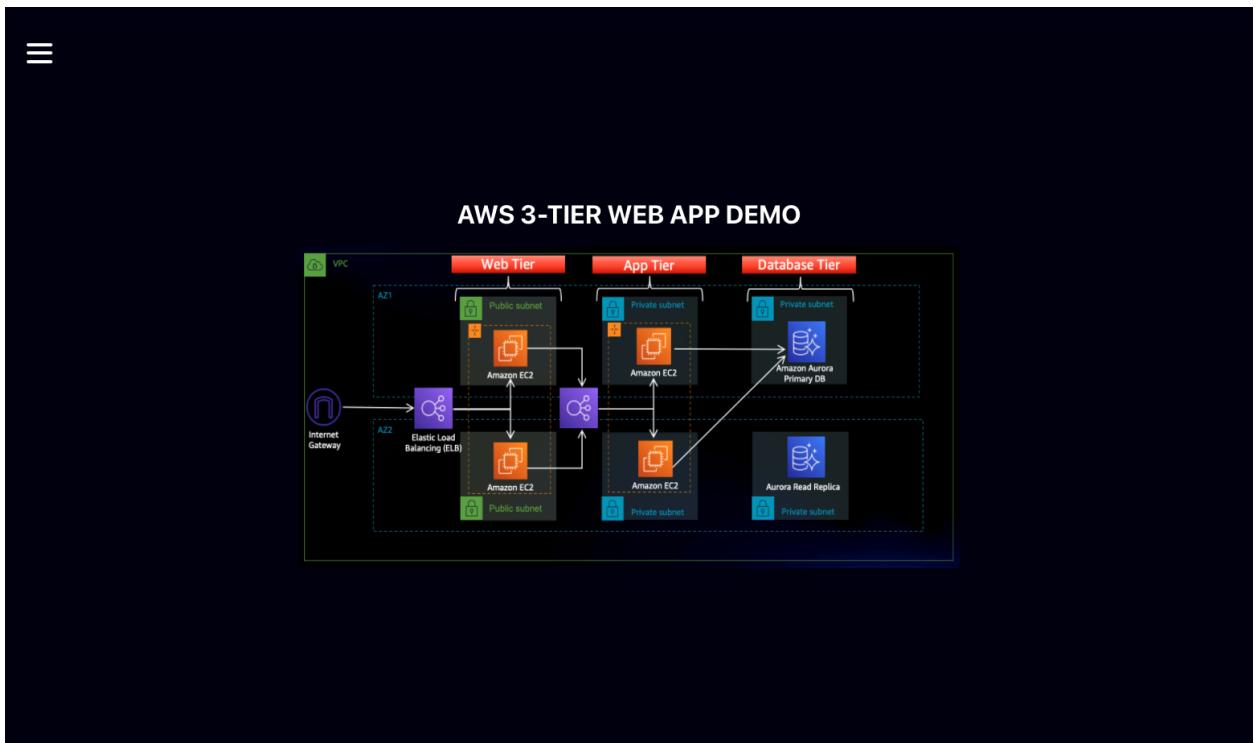
To make sure Nginx has permission to access our files execute this command:

```
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

The screenshot shows the AWS Lambda console with the search bar set to "Lambda function". Below the search bar, there are two Lambda functions listed:

Name	Description	Last modified	Region
lambda-function-1	My first Lambda function	2023-09-15 10:00:00 UTC	us-east-1
lambda-function-2	My second Lambda function	2023-09-15 10:00:00 UTC	us-east-1

7.2 - TG

The screenshot shows the AWS Lambda console with the search bar set to "Lambda function". Below the search bar, there are two Lambda functions listed:

Name	Description	Last modified	Region
lambda-function-1	My first Lambda function	2023-09-15 10:00:00 UTC	us-east-1
lambda-function-2	My second Lambda function	2023-09-15 10:00:00 UTC	us-east-1

7.3 - Internet Facing LB

The screenshot shows the AWS Elastic Load Balancing console. At the top, there is a search bar labeled "Filter load balancers". Below it is a table with columns: Name, State, Type, Scheme, IP address type, VPC ID, Availability Zones, Security groups, and DNS name. Two load balancers are listed:

Name	State	Type	Scheme	IP address type	VPC ID	Availability Zones	Security groups	DNS name
app-tier-internal-LB	Active	application	Internal	IPv4	vpc-074399afdb0c194f7	2 Availability Zones	sg-0488ccb5e5e336939	internal-app-tie
web-tier-external-LB	Active	application	Internet-facing	IPv4	vpc-074399afdb0c194f7	2 Availability Zones	sg-0c674dc1ba1f10b024	web-tier-extern

Below the table, a section titled "Load balancer: web-tier-external-LB" is expanded. It contains tabs for Details, Listeners and rules, Network mapping, Resource map, Security, Monitoring, Integrations, Attributes, Capacity, and Tags. The Details tab is selected. The "Details" section includes fields for Load balancer type (Application), Status (Active), VPC (vpc-074399afdb0c194f7), Load balancer IP address type (IPv4), Scheme (Internet-facing), Hosted zone (Z35SXDOTRQ7X7K), Availability Zones (subnet-00d5f9348b7420d8a us-east-1a, subnet-09cd5b56436a43c9fd us-east-1b), Date created (February 2, 2026, 07:06 (UTC+05:30)), Load balancer ARN (arn:aws:elasticloadbalancing:us-east-1:961852293555:loadbalancer/app/web-tier-external-LB/8cbed9f0dd842966), and DNS name (Info, web-tier-external-LB-1122120879.us-east-1.elb.amazonaws.com (A Record)).

7.4 - Launch Template

The screenshot shows the AWS Lambda console. At the top, there is a search bar labeled "Search". Below it is a table with columns: Launch Template ID, Launch Template Name, Default Version, Latest Version, Create Time, Created By, Managed, and Operator. Two launch templates are listed:

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

Below the table, a section titled "web-server-launch-temp (lt-0c9fedcf7b4392db4)" is expanded. It contains tabs for Details, Versions, and Template tags. The Details tab is selected. The "Launch template details" section includes fields for Launch template ID (lt-0c9fedcf7b4392db4), Launch template name (web-server-launch-temp), Default version (1), and Owner (arn:aws:iam::961852293555:root). The "Launch template version details" section includes fields for Version (1 (Default)), Description (web-server-launch-temp), Date created (2026-02-02T01:39:40.000Z), and Created by (arn:aws:iam::961852293555:root). The "Instance details" section includes fields for AMI ID (ami-0c2d6f32524fcba4), Instance type (t2.micro), Availability Zone (-), and Availability Zone Id (-). The "Storage" section includes fields for Key pair name (-) and Security groups (-). The "Resource tags" section includes fields for 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](#)

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availability Zones	Creation time
<input checked="" type="checkbox"/> web-server-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-server-ASG

[Details](#) [Integrations](#) [Automatic scaling](#) [Instance management](#) [Instance refresh](#) [Activity](#) [Monitoring](#) [Tags - moved](#)

Capacity overview

[Edit](#)

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

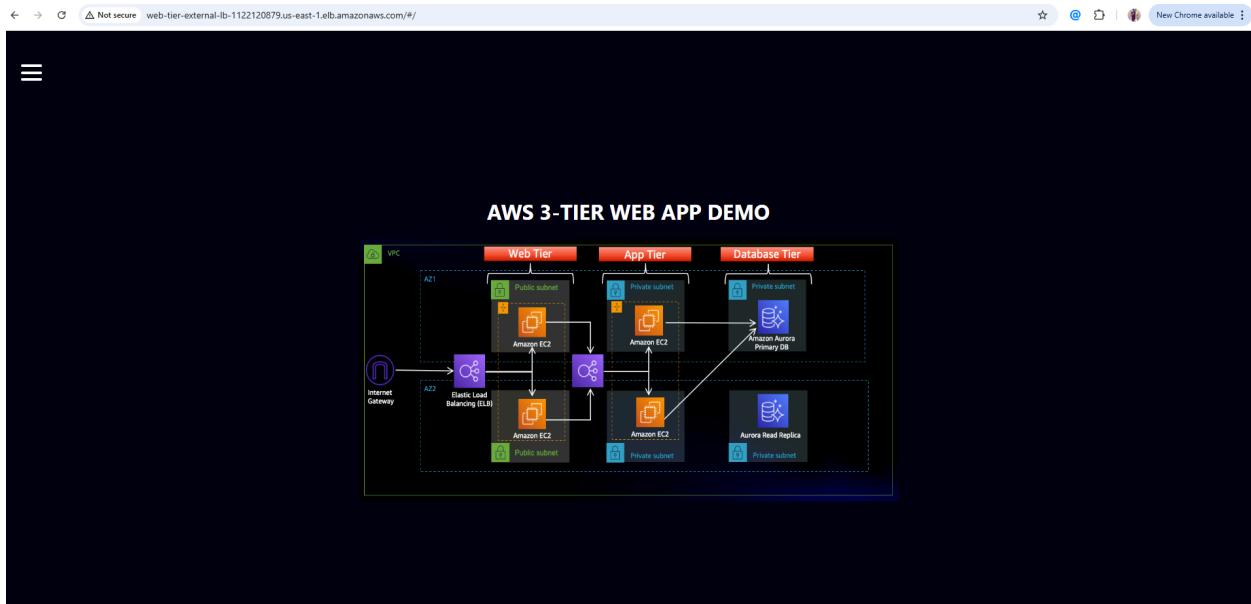
Date created
Mon Feb 02 2026 07:11:37 GMT+0530 (India Standard Time)

Launch template

[Edit](#)

Launch template lt-0c9fedcf7b4392db4 web-server-launch-temp	AMI ID ami-0c2d6f32524fcba4	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



The screenshot displays a database demo page titled "AURORA DATABASE DEMO PAGE". On the left, there is a navigation menu with "HOME" and "DB DEMO" options. The main content area shows a table with the following data:

ID	AMOUNT	DESC
1	400	groceries
2	200	test
		DEL

