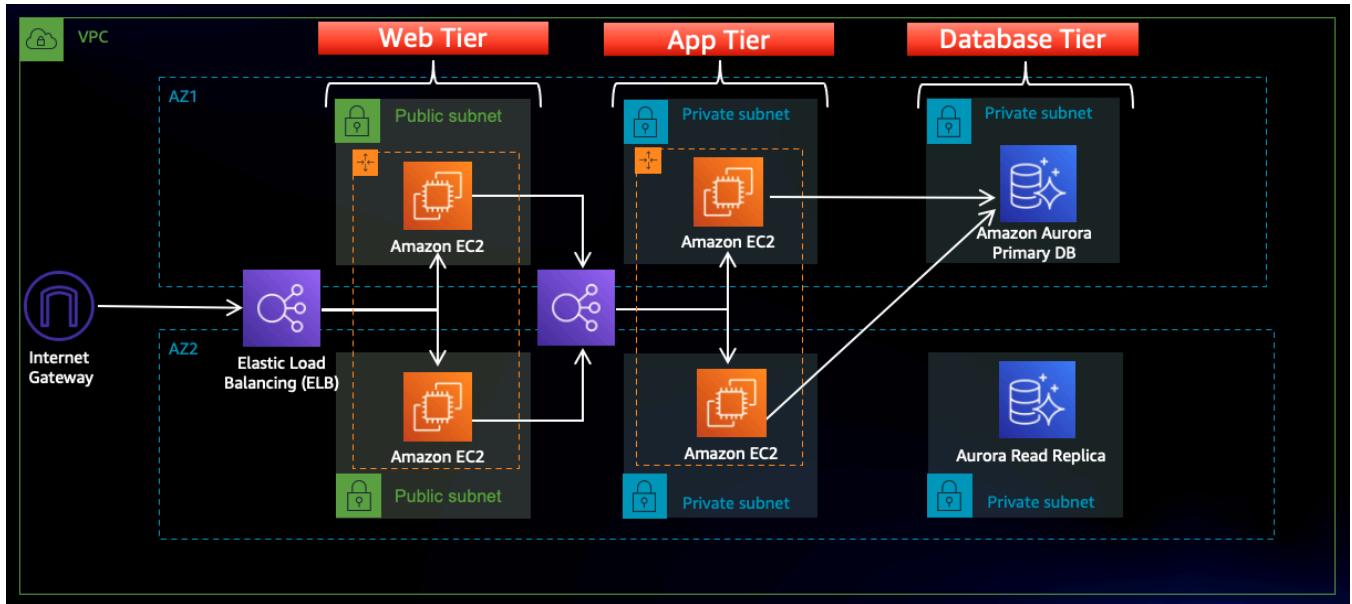


Build a Three Tier Architecture



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.

What is a three-tier architecture?

A three-tier application is a type of software architecture that divides an application into three logical layers: presentation, application (or logic), and data. Each layer serves a specific purpose and communicates with the other layers to perform various functions within the application.

Parts of a three-tier architecture

Presentation Tier (Front-end)

This layer is responsible for handling user interactions and displaying information to users. It includes user interfaces, such as web pages, mobile apps, or desktop interfaces. The presentation tier communicates with the application tier to request and receive data, but it doesn't directly interact with the data storage.

Application Tier (Middle-tier or Logic Tier)

This layer contains the core logic and functionality of the application. It processes user requests from the presentation tier, executes business logic, and interacts with the data tier to retrieve or update data.

The application tier acts as an intermediary between the presentation tier and the data tier, ensuring separation of concerns and modularity.

Data Tier (Back-end)

This layer manages the storage and retrieval of data used by the application. It typically consists of databases or other data storage systems. The data tier stores and organizes data, handles data retrieval and manipulation requests from the application tier, and ensures data integrity and security.

Navigate to the S3 service in the AWS console and create a new S3 bucket.

Give it a name → Create Bucket Give the Bucket Name → Select Region → Enabled ACLs → Enable Bucket Versioning → Click on Create

The screenshot shows the AWS S3 Buckets page. At the top, there are tabs for 'General purpose buckets' and 'All AWS Regions'. Below that, a table lists one bucket: 'mybucket-himanshu97' located in 'US East (N. Virginia) us-east-1'. The 'Create bucket' button at the top right is highlighted with a red box.

Navigate to the IAM dashboard in the AWS console and create an EC2 role.

Click on IAM → Roles → Create Roles → Selected Trusted Entity → AWS service → use case (EC2) → Add Permission → we need two policies AmazonSSMManagedInstanceCore AmazonS3ReadOnlyAccess → Click on Create Role

The screenshot shows the IAM Create Role wizard. It consists of three main sections: Step 2: Add permissions (listing two policies), Step 3: Add tags (with an 'Add new tag' button), and a final review step. In the review step, the role name 'Todays-ec2-s3-role' is highlighted with a red box. The 'Create role' button at the bottom right is also highlighted with a red box.

In this section we will be building out the VPC networking components as well as security groups that will add a layer of protection around our EC2 instances, Aurora databases, and Elastic Load Balancers.

Create VPC → Define IPv4 CIDR → and click on Create VPC

Create VPC Info
A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances.

VPC settings
Resources to create: **Info**
Creates only the VPC resource or the VPC and other networking resources.
 VPC only VPC and more

Name tag (optional)
Creates a tag with a key of 'Name' and a value that you specify.
 3-Tier VPC 1-Tier VPC

IPv4 CIDR block **Info**
 IPv4 CIDR manual input
Use the allocated IPv4 CIDR block
 IPv4 CIDR block assigned by me

IPv4 CIDR
10.0.0.16 IPv4 CIDR must be between /16 and /28.

IPv6 CIDR block **Info**
 No IPv6 CIDR block
 IPv6-assigned IPv6 CIDR block
 Auto-assigned IPv6 CIDR block
 IPv6 CIDR assigned by me

Tenancy **Info**
Default

Tags
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value (optional)
<input type="text"/> Name	<input type="text"/> 3-Tier VPC

Add tag You can add 49 more tags

Cancel Preview code **Create VPC**

We will need six subnets across two availability zones. That means that three subnets will be in one availability zone, and three subnets will be in another zone. Each subnet in one availability zone will correspond to one layer of our three tier architecture. Create each of the 6 subnets by specifying the VPC we created in part 1 and then choose a name, availability zone, and appropriate CIDR range for each of the subnets.

Public-Web-Subnet-AZ-1 -10.0.0.0/24

Private-App-Subnet-AZ-1 -10.0.10.0/24

Private-DB-Subnet-AZ-1 -10.0.20.0/24

Public-Web-Subnet-AZ-2 -10.0.30.0/24

Private-App-Subnet-AZ-2 -10.0.40.0/24

Private-DB-Subnet-AZ-2 -10.0.50.0/24

Subnets (6) Info
Last updated 1 minute ago

Name	Subnet ID	State	VPC	Block Public	IPv4 CIDR	IPv6 CIDR	IPv6 CIDR association...
Public-Web-Subnet-AZ-1	subnet-04f0075b251391c29	Available	ypc-0377c4b02fb8c2fcfa 3-T...	<input type="radio"/> Off	10.0.0.0/24	-	-
Private-App-Subnet-AZ-2	subnet-0696efb67feaf0ff	Available	ypc-0377c4b02fb8c2fcfa 3-T...	<input type="radio"/> Off	10.0.40.0/24	-	-
Private-App-Subnet-AZ-1	subnet-0a2c09eb8b5d80de	Available	ypc-0377c4b02fb8c2fcfa 3-T...	<input type="radio"/> Off	10.0.10.0/24	-	-
Public-Web-Subnet-AZ-2	subnet-0e4952f33bf4ce25	Available	ypc-0377c4b02fb8c2fcfa 3-T...	<input type="radio"/> Off	10.0.30.0/24	-	-
Private-DB-Subnet-AZ-2	subnet-06419472962b0e6e5	Available	ypc-0377c4b02fb8c2fcfa 3-T...	<input type="radio"/> Off	10.0.50.0/24	-	-
Private-DB-Subnet-AZ-1	subnet-03e5c146b1fc7be18	Available	ypc-0377c4b02fb8c2fcfa 3-T...	<input type="radio"/> Off	10.0.20.0/24	-	-

Create Internet Gateway

Attach to VPC (igw-025aa0a6f42ca6a2e) [Info](#)

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

Available VPCs
Attach the internet gateway to this VPC.
 [X](#)

▶ AWS Command Line Interface command

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

And Attach to Custom VPC that we made

⌚ The following internet gateway was created: igw-025aa0a6f42ca6a2e - 3-Tier-IGW. You can now attach to a VPC to enable the VPC to communicate with the internet. [Attach to a VPC](#) [X](#)

igw-025aa0a6f42ca6a2e / 3-Tier-IGW [Actions](#)

Details Info	Internet gateway ID igw-025aa0a6f42ca6a2e	State Detached	VPC ID -	Owner 148761672643
Tags	Manage tags			
<input type="text" value="Search tags"/>	Key <input type="text" value="Name"/>	Value <input type="text" value="3-Tier-IGW"/>	< 1 > X	

In order for our instances in the app layer private subnet to be able to access the internet they will need to go through a NAT Gateway. For high availability, you'll deploy one NAT gateway in each of your public subnets. Navigate to NAT Gateways on the left side of the current dashboard and click Create NAT Gateway.

Create NAT gateway [Info](#)

A highly available, managed Network Address Translation (NAT) service that instances in private subnets can use to connect to services in other VPCs, on-premises networks, or the internet.

NAT gateway settings
Name - optional
Create a tag with a key of 'Name' and a value that you specify.
NAT-GW-A1
The name can be up to 256 characters long.

Select a subnet in which to create the NAT gateway.
subnet-0400075251391c29 (Public-Web-Subnet-AZ-1)

Connectivity type
Select a connectivity type for the NAT gateway.
 Public
 Private

Elastic IP allocation ID [Info](#)
Assign an Elastic IP address to the NAT gateway.
eipalloc-0e0321a201e75e77 [Allocate Elastic IP](#)

▶ Additional settings [Info](#)

Tags
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key <input type="text" value="Name"/>	Value - optional <input type="text" value="NAT-GW-A1"/> X Remove
Add new tag You can add 49 more tags.	

[Cancel](#) **Create NAT gateway**

Fill in the Name, choose one of the public subnets you created, and then allocate an Elastic IP. Click Create NAT gateway.

Create NAT gateway [Info](#)

A highly available, managed Network Address Translation (NAT) service that instances in private subnets can use to connect to services in other VPCs, on-premises networks, or the internet.

NAT gateway settings
Name - optional
Create a tag with a key of 'Name' and a value that you specify.
NAT-GW-A2
The name can be up to 256 characters long.

Select a subnet in which to create the NAT gateway.
subnet-0400075251391c29 (Public-Web-Subnet-AZ-2)

Connectivity type
Select a connectivity type for the NAT gateway.
 Public
 Private

Elastic IP allocation ID [Info](#)
Assign an Elastic IP address to the NAT gateway.
eipalloc-056462f1006992015 [Allocate Elastic IP](#)

▶ Additional settings [Info](#)

Tags
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key <input type="text" value="Name"/>	Value - optional <input type="text" value="NAT-GW-A2"/> X Remove
Add new tag You can add 49 more tags.	

[Cancel](#) **Create NAT gateway**

Here we need 3 Route Table

One is public Route Table

And Two PrivateRoute Table for AZ1 and AZ2

Click on create route → give it name (Public Route Table) → select custom VPC → after create route table → got to edit routes → add routes 0.0.0.0 → target → Internet gateway after edit routes → click on subnet association → select both → public Subnet AZ1 and AZ2 → and Click on Save Changes

rtb-0f53c8a802f394fee / PUBLIC-ROUTE-TABLE

Details Info

Route table ID: rtb-0f53c8a802f394fee
VPC: vpc-03774b02fb8c2fcf | 3-Tier-VPC

Main
Owner ID: 148761672643

Explicit subnet associations: -
Edge associations: -

Routes (1)

Destination	Target	Status	Propagated
10.0.0.0/16	local	Active	No

Edit routes

Destination	Target	Status	Propagated
10.0.0.0/16	local	Active	No
0.0.0.0/0	Internet Gateway	-	-
0.0.0.0/0	igw-025aa0a6f42ca6a2	-	-

Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (2/6)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
Public-Web-Subnet-AZ-1	subnet-04f0075b251391c29	10.0.0.0/24	-	Main (rtb-0d04b7a7319ff412f)
Private-App-Subnet-AZ-2	subnet-0696efb7e7fa0f03	10.0.40.0/24	-	Main (rtb-0d04b7a7319ff412f)
Private-App-Subnet-AZ-1	subnet-0a2c09eb8be5d80de	10.0.10.0/24	-	Main (rtb-0d04b7a7319ff412f)
Public-Web-Subnet-AZ-2	subnet-0ed49b2f3b0f4ce25	10.0.30.0/24	-	Main (rtb-0d04b7a7319ff412f)
Private-DB-Subnet-AZ-2	subnet-06419472962bae6e5	10.0.50.0/24	-	Main (rtb-0d04b7a7319ff412f)
Private-DB-Subnet-AZ-1	subnet-03e5e146bfdfc7be18	10.0.20.0/24	-	Main (rtb-0d04b7a7319ff412f)

Selected subnets

subnet-04f0075b251391c29 / Public-Web-Subnet-AZ-1	subnet-0ed49b2f3b0f4ce25 / Public-Web-Subnet-AZ-2
---------------------------------------------------	---------------------------------------------------

Save associations

Create Private route table for AZ1 → select private app subnet AZ1

After creating private route table 1 → edit route → edit route 0.0.0.0 → Source Target is → NAT gateway AZ1 (same step for private route table 2)

Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (1/6)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
Public-Web-Subnet-AZ-1	subnet-04f0075b251391c29	10.0.0.0/24	-	rtb-0f53c8a802f394fee / PUBLIC-...
Private-App-Subnet-AZ-2	subnet-0696efb7e7fa0f03	10.0.40.0/24	-	Main (rtb-0d04b7a7319ff412f)
Private-App-Subnet-AZ-1	subnet-0a2c09eb8be5d80de	10.0.10.0/24	-	Main (rtb-0d04b7a7319ff412f)
Public-Web-Subnet-AZ-2	subnet-0ed49b2f3b0f4ce25	10.0.30.0/24	-	rtb-0f53c8a802f394fee / PUBLIC-...
Private-DB-Subnet-AZ-2	subnet-06419472962bae6e5	10.0.50.0/24	-	Main (rtb-0d04b7a7319ff412f)
Private-DB-Subnet-AZ-1	subnet-03e5e146bfdfc7be18	10.0.20.0/24	-	Main (rtb-0d04b7a7319ff412f)

Selected subnets

subnet-0696efb7e7fa0f03 / Private-App-Subnet-AZ-2

Save associations

The first screenshot shows the 'Edit routes' interface for a NAT Gateway target. A new route is being added with a destination of 0.0.0.0/0 and a target of 'nat-07b6b013067318443'. The second screenshot shows a similar process for a specific IP address target. The third screenshot shows the 'Route tables (5) Info' page with five route tables listed.

Name	Route table ID	Explicit subnet associations	Main	VPC	Owner ID
rtb-0f9a10bd02ab3f032	-	-	Yes	vpc-09xe4835ac45db0a6	148761672643
PRIVATE-ROUTE-TABLE-AZ2	rtb-0ef0553a315841	subnet-0d04e5b67e7fac03 / Private-App-Subnet-AZ2	No	vpc-0377c4b02fb8c2fc 3-T...	148761672643
PRIVATE-ROUTE-~TABLE-AZ1	rtb-0c0f6ed36ade5b230	subnet-0a2c09e0bb0e5d80de / Private-App-Subnet-AZ1	No	vpc-0377c4b02fb8c2fc 3-T...	148761672643
rtb-06067a7319f124	-	-	Yes	vpc-0377c4b02fb8c2fc 3-T...	148761672643
PUBLIC-ROUTE-TABLE	rtb-0f53c8802f39dfcc	2 subnets	No	vpc-0377c4b02fb8c2fc 3-T...	148761672643

Security groups will tighten the rules around which traffic will be allowed to our Elastic Load Balancers and EC2 instances.

The first security group you'll create is for the public, internet facing load balancer. After typing a name and description, add an inbound rule to allow HTTP type traffic for your IP.

The screenshot shows the 'Create security group' interface. In the 'Basic details' section, the security group name is 'internet facing lb sg' and it is associated with the VPC 'vpc-0377c4b02fb8c2fc (3-Tier VPC)'. In the 'Inbound rules' section, there are two rules for 'HTTP' traffic on port 80, both allowing traffic from '0.0.0.0/0'.

The second security group you'll create is for the public instances in the web tier. After typing a name and description, add an inbound rule that allows HTTP type traffic from your internet facing load balancer security group you created in the previous step. This will allow traffic from your public facing load balancer to hit your instances. Then, add an additional rule that will allow HTTP type traffic for your IP. This will allow you to access your instance when we test.



Create security group [Info](#)

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name [Info](#)
Web-Tier-SG
Name cannot be edited after creation.

Description [Info](#)
sg for the web-tier

VPC [Info](#)
vpc-0377c4b02fb8c2fca (3-Tier-VPC)

Inbound rules [Info](#)

Type	Protocol	Port range	Source	Description - optional	Delete
HTTP	TCP	80	Anywhere... Info <input type="text" value="0.0.0.0"/> Delete	<input type="text" value="0.0.0.0/0"/> Delete	Delete
HTTP	TCP	80	My IP Info Delete	<input type="text" value="27.57.255.157/32"/> Delete	Delete

[Add rule](#)

The third security group will be for our internal load balancer. Create this new security group and add an inbound rule that allows HTTP type traffic from your public instance security group. This will allow traffic from your web tier instances to hit your internal load balancer.

Create security group [Info](#)

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name [Info](#)
internet-facing-lb-sg
Name cannot be edited after creation.

Description [Info](#)
internet facing lb sg

VPC [Info](#)
vpc-0377c4b02fb8c2fca (3-Tier-VPC)

Inbound rules [Info](#)

Type	Protocol	Port range	Source	Description - optional	Delete
HTTP	TCP	80	My IP Info Delete	<input type="text" value="27.57.255.157/32"/> Delete	Delete

[Add rule](#)

Outbound rules [Info](#)

Type	Protocol	Port range	Destination	Description - optional	Delete
All traffic	All	All	Custom Info <input type="text" value="0.0.0.0"/> Delete	<input type="text" value="0.0.0.0/0"/> Delete	Delete

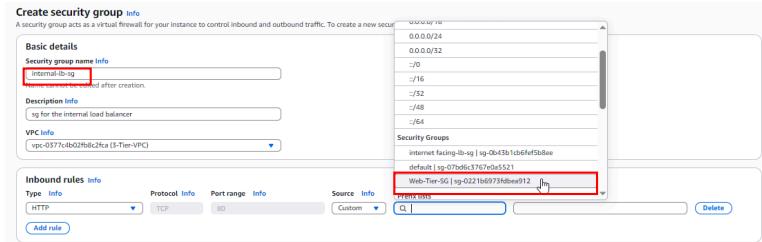
[Add rule](#)

⚠ Rules with destination of 0.0.0.0/0 or ::/0 allow your instances to send traffic to any IPv4 or IPv6 address. We recommend setting security group rules to be more restrictive and to only allow traffic to specific known IP addresses.

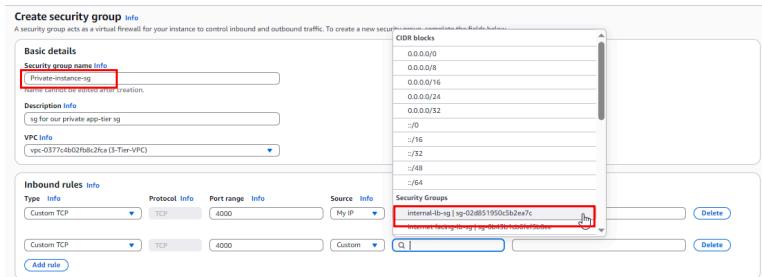
Tags - optional
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.
No tags associated with the resource.

[Add new tag](#)
You can add up to 50 more tags

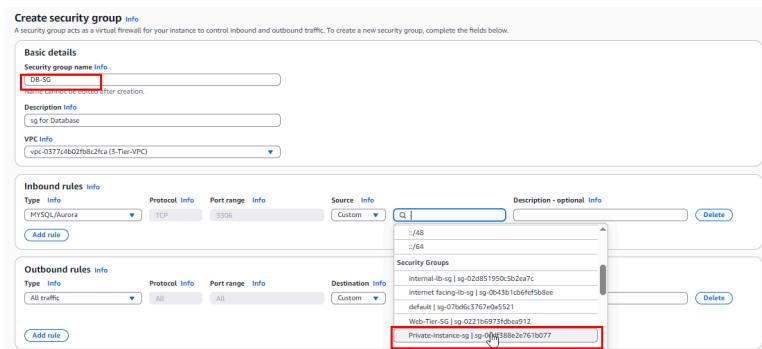
[Cancel](#) [Create security group](#)



The fourth security group we'll configure is for our private instances. After typing a name and description, add an inbound rule that will allow TCP type traffic on port 4000 from the internal load balancer security group you created in the previous step. This is the port our app tier application is running on and allows our internal load balancer to forward traffic on this port to our private instances. You should also add another route for port 4000 that allows your IP for testing.



The fifth security group we'll configure protects our private database instances. For this security group, add an inbound rule that will allow traffic from the private instance security group to the MySQL/Aurora port (3306).



Security Groups (6) <small>Info</small>						
<input type="text"/> VPC ID + vpc-03377c4b02fb8c2fc4		<input type="text"/> Find security groups by attribute or tag		<input type="button"/> Actions		<input type="button"/> Export security groups to CSV
<input type="button"/> Create security group						
Name	Security group ID	Security group name	VPC ID	Description	Owner	Inbound rules count
sg-02d851950592ea7c	vpc-03377c4b02fb8c2fc4	internal-lb-sg	vpc-03377c4b02fb8c2fc4	sg for the internal load balancer	148761672643	1 Permission entry
sg-0b3b193ef4f5fb9e9	vpc-03377c4b02fb8c2fc4	internet-facing-lb-sg	vpc-03377c4b02fb8c2fc4	External lb sg	148761672643	2 Permission entries
sg-07bd6c3767v0a5321	vpc-03377c4b02fb8c2fc4	default	vpc-03377c4b02fb8c2fc4	default VPC security group	148761672643	1 Permission entry
sg-02216073fb0be012	vpc-03377c4b02fb8c2fc4	Web-Tier-SG	vpc-03377c4b02fb8c2fc4	sg for the web-tier	148761672643	2 Permission entries
sg-05027ed8c195ec0fa	vpc-03377c4b02fb8c2fc4	DB-SG	vpc-03377c4b02fb8c2fc4	sg for Database	148761672643	1 Permission entry
sg-0dd4338e2e27616077	vpc-03377c4b02fb8c2fc4	Private-instance-sg	vpc-03377c4b02fb8c2fc4	sg for our private app-tier sg	148761672643	2 Permission entries

Database Deployment

First we create a subnet group → give it a name → give the description → select custom vpc → select AZ → Add subnet → click on create subnet

Create DB subnet group

Please create a new subnet group, give it a name and a description, and choose an existing VPC. You will then be able to add subnets related to that VPC.

Subnet group details
Name You won't be able to modify the name after your subnet group has been created. <input type="text" value="Three-Tier-DB-subnet-group"/>
Description Must contain from 1 to 253 characters. Alphanumeric characters, spaces, hyphens, underscores, and periods are allowed. <input type="text" value="subnet group for the database layer of the architecture"/>
VPC Choose a VPC identifier that corresponds to the subnets you want to use for your DB subnet group. You won't be able to choose a different VPC identifier after your subnet group has been created. <input type="text" value="3-Tier-VPC (vpc-03377c4b02fb8c2fc4)
6 Subnets, 2 Availability Zones"/>

Add subnets

Availability Zones Choose the Availability Zones that include the subnets you want to add. <input type="button" value="Choose an availability zone"/> <input type="button" value="us-east-1a"/> <input type="button" value="us-east-1c"/>
Subnets Choose the subnets that you want to add. The list includes the subnets in the selected Availability Zones. <input type="button" value="Select subnets"/> <input type="button" value="Private-DB-Subnet-AZ-1"/> <input type="button" value="Private-DB-Subnet-AZ-2"/>
For Multi-AZ DB clusters, you must select 3 subnets in 3 different Availability Zones.
Subnets selected (2)
Availability zone Subnet name Subnet ID CIDR block
us-east-1a Private-DB-Subnet-AZ-1 subnet-03e5e146bfcd7be18 10.0.20.0/24
us-east-1c Private-DB-Subnet-AZ-2 subnet-06419472962bae65 10.0.50.0/24

Successfully created Three-Tier-DB-subnet-group. View subnet group

Subnet groups (1)
<input type="text"/> Filter by subnet group
Name Description Status VPC
three-tier-db-subnet-group subnet group for the database layer of the architecture Complete vpc-03377c4b02fb8c2fc4

After creating subnet group → we create Database → click on Database → select creating method (Standard) → select engine (mysql) → free tier → Master user name → admin → give it a password → instance configuration (default) → select custom VPC → select Db subnet group → select security group (DB-SG) → Click on Create Database

Create database info

Choose a database creation method

Standard create
You set all of the configuration options, including ones for availability, security, backups, and maintenance.

Easy create
Use recommended best-practice configurations. Some configuration options can be changed after the database is created.

Engine options

Engine type [Info](#)

Aurora (MySQL Compatible) 

Aurora (PostgreSQL Compatible) 

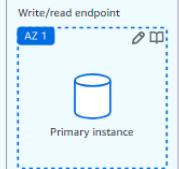
MySQL 

Free tier
Use RDS Free Tier to develop new applications, test existing applications, or gain hands-on experience with Amazon RDS. [Info](#)

Depending on the deployment option you choose. Learn more in the [Amazon RDS](#)

Single-AZ DB instance deployment (1 instance)
Creates a single DB instance without standby instances. This setup provides:

- 99.9% uptime
- No data redundancy



Settings

DB instance identifier [Info](#)
Type a name for your DB instance. The name must be unique across all DB instances owned by your AWS account in the current AWS Region.

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 63 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

Credentials Settings

Master username [Info](#)
Type a login ID for the master user of your DB instance.

1 to 16 alphanumeric characters. The first character must be a letter.

Credentials management
You can use AWS Secrets Manager or manage your master user credentials.

Managed in AWS Secrets Manager - most secure
RDS generates a password for you and manages it throughout its lifecycle using AWS Secrets Manager.

Self managed
Create your own password or have RDS create a password that you manage.

Auto generate password
Amazon RDS can generate a password for you, or you can specify your own password.

Master password [Info](#)

Password strength **Very weak**
Minimum constraints: At least 8 printable ASCII characters. Can't contain any of the following symbols: / \ * @

Confirm master password [Info](#)

Instance configuration

The DB instance configuration options below are limited to those supported by the engine that you selected above.

[DB instance class](#) | [Info](#)

[▼ Hide filters](#)

Show instance classes that support Amazon RDS Optimized Writes [Info](#)
Amazon RDS Optimized Writes improves write throughput by up to 2x at no additional cost.

Include previous generation classes

Standard classes (includes m classes)

Memory optimized classes (includes r and x classes)

Burstable classes (includes t classes)

db.t4g.micro
2 vCPUs 1 GiB RAM Network: Up to 2,085 Mbps

Virtual private cloud (VPC) Info
Choose the VPC. The VPC defines the virtual networking environment for this DB instance.
3-Tier-VPC (vpc-0377c4ab02fb8c2fc)
6 Subnets, 2 Availability Zones

Only VPCs with a corresponding DB subnet group are listed.

(i) After a database is created, you can't change its VPC.

DB subnet group Info
Choose the DB subnet group. The DB subnet group defines which subnets and IP ranges the DB instance can use in the VPC that you selected.
three-tier-db-subnet-group
2 Subnets, 2 Availability Zones

Virtual private cloud (VPC) Info
Choose the VPC. The VPC defines the virtual networking environment for this DB instance.
3-Tier-VPC (vpc-0377c4ab02fb8c2fc)
6 Subnets, 2 Availability Zones

Only VPCs with a corresponding DB subnet group are listed.

(i) After a database is created, you can't change its VPC.

DB subnet group Info
Choose the DB subnet group. The DB subnet group defines which subnets and IP ranges the DB instance can use in the VPC that you selected.
three-tier-db-subnet-group
2 Subnets, 2 Availability Zones

VPC security group (firewall) Info
Choose one or more VPC security groups to allow access to your database. Make sure that the security group rules allow the appropriate incoming traffic.

Choose existing
Choose existing VPC security groups

Create new
Create new VPC security group

Existing VPC security groups
Choose one or more options
DB-SG X

Availability Zone Info
us-east-1c

AWS Search [Alt+S] United States (N. Virginia) Himanshu Nimje

Aurora and RDS > Databases

Creating database database-1
Your database might take a few minutes to launch. You can use settings from database-1 to simplify configuration of suggested database add-ons while we finish creating your DB for you.

View connection details Create database Actions

Databases (1)

DB identifier	Status	Role	Engine	Region...	Size	Recommendations	CPU	Current activity	Maintenance	VPC
database-1	Creating	Instance	MySQL C...	us-east-1c	db.t4g.micro	-	-	-	none	vpc-0377...

AWS Search [Alt+S] United States (N. Virginia) Himanshu Nimje

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

View connection details Actions

Successfully created database database-1
You can use settings from database-1 to simplify configuration of suggested database add-ons while we finish creating your DB for you.

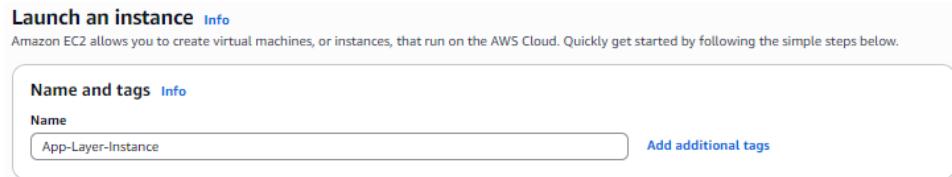
database-1

Summary

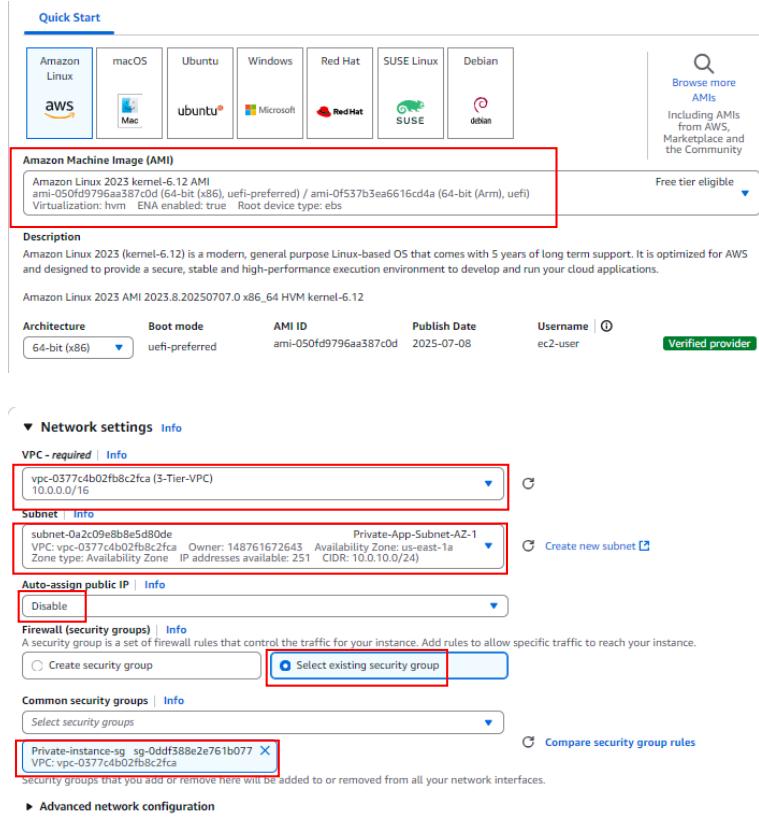
DB Identifier database-1	Status Available	Role Instance	Engine MySQL Community	Recommendations
CPU	Class db.t4g.micro	Current activity	Region & AZ us-east-1c	

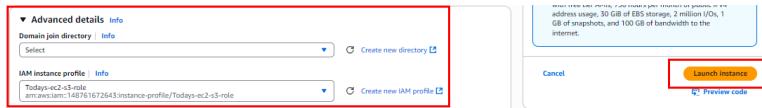
App Tier Instance Deployment

In this section of our project we will create an EC2 instance for our app layer and make all necessary software configurations so that the app can run. The app layer consists of a Node.js application that will run on port 4000. We will also configure our database with some data and tables.



Click on launch instance → give it name → select Amazon machine image → instance type (t2-micro) → process without key pair → select vpc → subnet (select private App subnet AZ1) → public ip disable → in security group (select private instance sg) → click on Advanced Details → IAM profile (select ec2 and s3 role that we created) → click on launch instance





Connect to instances

`sudo -su ec2-user`

Let's take this moment to make sure that we are able to reach the internet via our NAT gateways.

`ping 8.8.8.8`

```
sh-5.2$ sudo -su ec2-user
[ec2-user@ip-10-0-10-133 bin]$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=116 time=2.01 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=116 time=1.33 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=116 time=1.39 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=116 time=1.31 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=116 time=1.32 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=116 time=1.26 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=116 time=1.29 ms
^C
--- 8.8.8.8 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6008ms
rtt min/avg/max/mdev = 1.264/1.415/2.011/0.245 ms
[ec2-user@ip-10-0-10-133 bin]$
```

```
[ec2-user@ip-10-0-10-133 bin]$ sudo yum install mysql -y
Amazon Linux 2023 Kernel Livepatch repository
No match for argument: mysql
Error: Unable to find a match: mysql
[ec2-user@ip-10-0-10-133 bin]$
```

`sudo wget`

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

```
[ec2-user@ip-10-0-10-133 bin]$ wget https://dev.mysql.com/get/mysql80-community-release-el9-1.noarch.rpm
```

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

```
[ec2-user@ip-10-0-10-133 bin]$ sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2022
```

sudo yum install mysql80-community-release-el9-1.noarch.rpm

```
[ec2-user@ip-10-0-10-133 bin]$ sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2022
[ec2-user@ip-10-0-10-133 bin]$ sudo yum install mysql80-community-release-el9-1.noarch.rpm
```

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

```
[ec2-user@ip-10-0-10-133 bin]$ sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2022
[ec2-user@ip-10-0-10-133 bin]$ sudo yum install mysql
```

```
[ec2-user@ip-10-0-10-133 bin]$ sudo rpm -e gpg-pubkey-3a79bd29-*
```

```
[ec2-user@ip-10-0-10-133 bin]$ sudo rpm -e gpg-pubkey-3a79bd29-*
[ec2-user@ip-10-0-10-133 bin]$ sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2022
```

YOU FOUND THIS ERROR MESSAGE → error: GPG check FAILED
SIMPLY TYPE THIS COMMAND ↓
sudo dnf install mysql-server --nogpgcheck -y

```
[ec2-user@ip-10-0-10-133 bin]$ sudo rpm -e gpg-pubkey-3a79bd29-*
[ec2-user@ip-10-0-10-133 bin]$ sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2022
[ec2-user@ip-10-0-10-133 bin]$ sudo dnf install mysql-server --nogpgcheck -y
```

```
Error: GPG check FAILED
[ec2-user@ip-10-0-10-133 bin]$ sudo rpm -e gpg-pubkey-3a79bd29-*
[ec2-user@ip-10-0-10-133 bin]$ sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2022
[ec2-user@ip-10-0-10-133 bin]$ sudo dnf install mysql-server --nogpgcheck -y
Last metadata expiration check: 0:03:23 ago on Mon Jul 21 15:42:50 2025.
Dependencies resolved.

=====
| Package           | Architecture | Version      | Repository | Size   |
|:-----|:-----|:-----|:-----|:-----|
| Installing:      |             |             |            |        |
|   mysql-community-server | x86_64      | 8.0.42-1.el9 | mysql80-community | 50 M  |
| Installing dependencies: |             |             |            |        |
|   mysql-community-client | x86_64      | 8.0.42-1.el9 | mysql80-community | 3.4 M |
|   mysql-community-client-plugins | x86_64      | 8.0.42-1.el9 | mysql80-community | 1.4 M |
|   mysql-community-common | x86_64      | 8.0.42-1.el9 | mysql80-community | 555 k |
|   mysql-community-icu-data-files | x86_64      | 8.0.42-1.el9 | mysql80-community | 2.3 M |
|   mysql-community-libs | x86_64      | 8.0.42-1.el9 | mysql80-community | 1.5 M |

Transaction Summary
=====
Install 6 Packages

Total size: 59 M
Downloaded size: 52 M
Installed size: 337 M
Downloading Packages:
[SKIPPED] mysql-community-client-8.0.42-1.el9.x86_64.rpm: Already downloaded
[SKIPPED] mysql-community-client-plugins-8.0.42-1.el9.x86_64.rpm: Already downloaded
[SKIPPED] mysql-community-common-8.0.42-1.el9.x86_64.rpm: Already downloaded
[SKIPPED] mysql-community-icu-data-files-8.0.42-1.el9.x86_64.rpm: Already downloaded
(5/6): mysql-community-server-8.0.42-1.el9.x86_64.rpm: 45 MB/s | 2.3 MB 00:00
(6/6): mysql-community-server-8.0.42-1.el9.x86_64.rpm: 84 MB/s | 50 MB 00:00
Total                                         88 MB/s | 52 MB 00:00

Running transaction check
```

```
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction upgrade
Preparing :
Installing : mysql-community-common-8.0.42-1.el9.x86_64
1/1
Installing : mysql-community-client-plugins-8.0.42-1.el9.x86_64
2/6
Installing : mysql-community-client-8.0.42-1.el9.x86_64
3/6
Running scriptlet: mysql-community-client-8.0.42-1.el9.x86_64
3/6
Installing : mysql-community-client-plugins-8.0.42-1.el9.x86_64
4/6
Installing : mysql-community-common-8.0.42-1.el9.x86_64
5/6
Running scriptlet: mysql-community-common-8.0.42-1.el9.x86_64
5/6
Installing : mysql-community-icu-data-files-8.0.42-1.el9.x86_64
6/6
Running scriptlet: mysql-community-icu-data-files-8.0.42-1.el9.x86_64
6/6
Installing : mysql-community-server-8.0.42-1.el9.x86_64
6/6
Running scriptlet: mysql-community-server-8.0.42-1.el9.x86_64
6/6
Verifying : mysql-community-client-8.0.42-1.el9.x86_64
1/6
Verifying : mysql-community-client-plugins-8.0.42-1.el9.x86_64
2/6
Verifying : mysql-community-common-8.0.42-1.el9.x86_64
3/6
Verifying : mysql-community-icu-data-files-8.0.42-1.el9.x86_64
4/6
Verifying : mysql-community-libs-8.0.42-1.el9.x86_64
5/6
Verifying : mysql-community-server-8.0.42-1.el9.x86_64
6/6

WARNING: A newer release of "Amazon Linux" is available.
```

```
A newer release of "Amazon Linux" is available.
Available Versions:
Version 2023.8.20250715:
Run the following command to upgrade to 2023.8.20250715:
dnf upgrade --releasever=2023.8.20250715

Release notes:
https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes-2023.8.20250715.html

=====
Installed:
mysql-community-client-8.0.42-1.el9.x86_64      mysql-community-client-plugins-8.0.42-1.el9.x86_64      mysql-community-common-8.0.42-1.el9.x86_64
mysql-community-icu-data-files-8.0.42-1.el9.x86_64  mysql-community-lib-8.0.42-1.el9.x86_64          mysql-community-server-8.0.42-1.el9.x86_64

Complete!
[ec2-user@ip-10-0-10-133 bin]$
```

`mysql --version`

```
sh-5.2$ sudo -su ec2-user
[ec2-user@ip-10-0-10-133 bin]$ mysql --version
mysql Ver 8.0.42 for Linux on x86_64 (MySQL Community Server - GPL)
[ec2-user@ip-10-0-10-133 bin]$ mysql -h database-1.cmn2keomqcz.us-east-1.rds.amazonaws.com -u admin -p
```

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`

```
sh-5.2$ sudo -su ec2-user
[ec2-user@ip-10-0-10-133 bin]$ mysql --version
mysql Ver 8.0.42 for Linux on x86_64 (MySQL Community Server - GPL)
[ec2-user@ip-10-0-10-133 bin]$ mysql -h database-1.cmn2keomqcz.us-east-1.rds.amazonaws.com -u admin -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 44
Server version: 8.0.35 Source distribution

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

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

`CREATE DATABASE webappdb;`

```
sh-5.2$ sudo -su ec2-user
[ec2-user@ip-10-0-10-133 bin]$ mysql --version
mysql Ver 8.0.42 for Linux on x86_64 (MySQL Community Server - GPL)
[ec2-user@ip-10-0-10-133 bin]$ mysql -h database-1.cmn2keomqcz.us-east-1.rds.amazonaws.com -u admin -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 44
Server version: 8.0.35 Source distribution

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> CREATE DATABASE webappdb;
Query OK, 1 row affected (0.01 sec)

mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| webappdb |
+-----+
5 rows in set (0.00 sec)

mysql> USE webappdb;
Database changed
mysql>
```

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

```
mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| webappdb |
+-----+
5 rows in set (0.00 sec)

mysql> USE webappdb;
Database changed
mysql> CREATE TABLE IF NOT EXISTS transactions(id INT NOT NULL
    -> AUTO_INCREMENT, amount DECIMAL(10,2), description
    -> VARCHAR(100), PRIMARY KEY(id));
Query OK, 0 rows affected (0.05 sec)

mysql> INSERT INTO transactions (amount,description) VALUES ('400','groceries');
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM transactions;
+---+-----+
| id | amount | description |
+---+-----+
| 1 | 400.00 | groceries |
+---+-----+
1 row in set (0.00 sec)

mysql>
```

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;

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;

```
mysql> SHOW DATABASES;
+-----+
| Database      |
+-----+
| information_schema |
| mysql          |
| performance_schema |
| sys            |
| webappdb       |
+-----+
5 rows in set (0.00 sec)

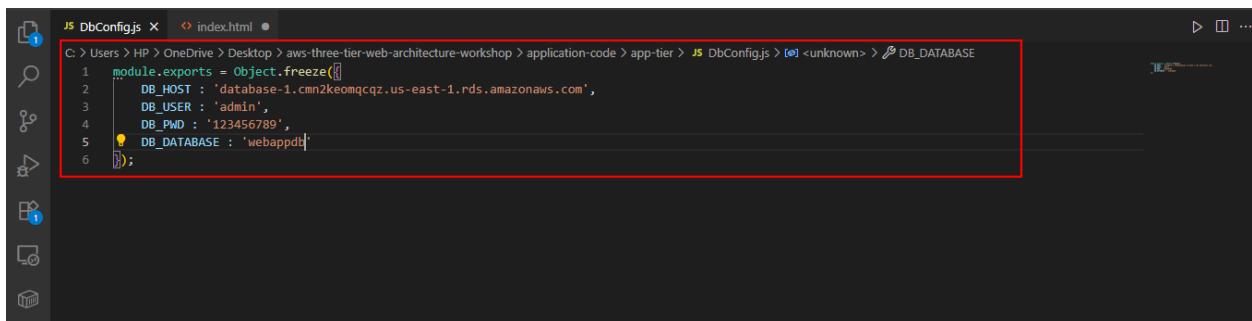
mysql> USE webappdb;
Database changed
mysql> CREATE TABLE IF NOT EXISTS transactions(id INT NOT NULL
    -> AUTO_INCREMENT, amount DECIMAL(10,2), description
    -> VARCHAR(100), PRIMARY KEY(id));
Query OK, 0 rows affected (0.05 sec)

mysql> INSERT INTO transactions (amount,description) VALUES ('400','groceries');
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM transactions;
+---+---+---+
| id | amount | description |
+---+---+---+
| 1  | 400.00 | groceries   |
+---+---+---+
1 row in set (0.00 sec)

mysql> exit
```

The first thing we will do is update our database credentials for the app tier. To do this, 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.



Upload the app-tier folder to the S3 bucket

The screenshot shows the AWS S3 console interface. At the top, there's a navigation bar with tabs for Objects, Metadata, Properties, Permissions, Metrics, Management, and Access Points. Below the navigation bar, the 'Objects (1)' section shows a single folder named 'application-code/'. A red box highlights the 'Upload' button in the top right corner of this section. In the bottom section, the 'application-code/' bucket is selected, showing three objects: 'app-tier/' (a folder), 'nginx.conf' (a configuration file), and 'web-tier/' (another folder). A red box highlights the contents of this bucket.

Now go back to our 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
```

```
sh-5.2$ sudo -su ec2-user
[ec2-user@ip-10-0-10-133 bin]$ curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh | bash
```

install a compatible version of Node.js and make sure it's being used
nvm install 16

nvm use 16

```
sh-5.2$ sudo -su ec2-user
[ec2-user@ip-10-0-10-133 bin]$ curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh | bash
  % Total    % Received % Xferd  Average Speed   Time     Time  Current
          Dload  Upload   Total   Spent    Left  Speed
100 14926  100 14926    0     0  690k      0 --:--:--  --:--:--  728k
=> Downloading nvm as script to '/home/ec2-user/.nvm'

=> Appending nvm source string to '/home/ec2-user/.bashrc'
=> Appending bash_completion source string to '/home/ec2-user/.bashrc'
=> Close and reopen your terminal to start using nvm or run the following to use it now:

export NVM_DIR="$HOME/.nvm"
[ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh" # This loads nvm
[ -s "$NVM_DIR/bash_completion" ] && \. "$NVM_DIR/bash_completion" # This loads nvm bash_completion
[ec2-user@ip-10-0-10-133 bin]$ source ~/.bashrc
[ec2-user@ip-10-0-10-133 bin]$ nvm install 16
Downloading and installing node v16.20.2...
Downloaded https://nodejs.org/dist/v16.20.2/node-v16.20.2-linux-x64.tar.xz...
#####
Computing checksums for package /home/ec2-user/.nvm/versions/node/v16.20.2/... 100.0%
Checksums matched for node@v16.20.2 (npm@v8.19.4)
Creating default alias: default → v16.20.2
[ec2-user@ip-10-0-10-133 bin]$ nvm use 16
Now using node v16.20.2 (npm v8.19.4)
[ec2-user@ip-10-0-10-133 bin]$
```

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

```
[ec2-user@ip-10-0-10-133 bin]$ npm install -g pm2
added 135 packages, and audited 136 packages in 7s

13 packages are looking for funding
  run `npm fund` for details

1 low severity vulnerability

Some issues need review, and may require choosing
a different dependency.

Run `npm audit` for details.
npm notice New major version of npm available! 8.19.4 => 11.4.2
npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.4.2
npm notice Run npm install -g npm@11.4.2 to update!
npm notice
[ec2-user@ip-10-0-10-133 bin]$
```

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

```
[ec2-user@ip-10-0-10-133 bin]$ source ./bashrc
[ec2-user@ip-10-0-10-133 bin]$ npm install 16
Downloading and installing node v16.20.2...
Downloaded https://nodejs.org/dist/v16.20.2/node-v16.20.2-linux-x64.tar.xz...
Computing checksum with sha256sum
Checksum matched.
Creating node v16.20.2 (npm v8.19.4)
Creating default alias: default -v16.20.2 (> v16.20.2)
[ec2-user@ip-10-0-10-133 bin]$ npm use 16
Now using node v16.20.2 (npm v8.19.4)
[ec2-user@ip-10-0-10-133 bin]$ npm install -g pm2
added 135 packages, and audited 136 packages in 7s

13 packages are looking for funding
  run `npm fund` for details

1 low severity vulnerability

Some issues need review, and may require choosing
a different dependency.

Run `npm audit` for details.
npm notice New major version of npm available! 8.19.4 => 11.4.2
npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.4.2
npm notice Run npm install -g npm@11.4.2 to update!
npm notice
[ec2-user@ip-10-0-10-133 bin]$ cd ~/
[ec2-user@ip-10-0-10-133 ~]$ aws s3 cp s3://mybucket-himanshu97/application-code/app-tier/ app-tier --recursive
```

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

cd ~/app-tier

npm install

pm2 start index.js

```
[ec2-user@ip-10-0-10-133 ~]$ cd ~/app-tier
[ec2-user@ip-10-0-10-133 app-tier]$ npm install
added 68 packages, and audited 69 packages in 2s

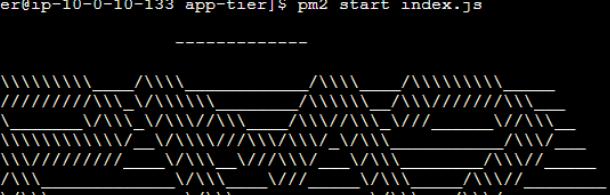
2 packages are looking for funding
  run `npm fund` for details

7 vulnerabilities (3 low, 4 high)

To address all issues, run:
  npm audit fix

Run `npm audit` for details.
[ec2-user@ip-10-0-10-133 app-tier]$ pm2 start index.js
-----

```



Runtime Edition

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

```
[ec2-user@ip-10-0-10-133 app-tier]$ pm2 list
    id  name   namespace  version  mode   pid   uptime  ⚡  status   cpu    mem   user   watching
  0  index  default    1.0.0     fork  31768  2m      0  online  0%  51.1mb  ec2-user  disabled

[ec2-user@ip-10-0-10-133 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 | 2025-07-21T17:39:21: PM2 log: Node.js version : 16.20.2
PM2 | 2025-07-21T17:39:21: PM2 log: Current arch : x64
PM2 | 2025-07-21T17:39:21: PM2 log: PM2 home : /home/ec2-user/.pm2
PM2 | 2025-07-21T17:39:21: PM2 log: PM2 PID file : /home/ec2-user/.pm2/pm2.pid
PM2 | 2025-07-21T17:39:21: PM2 log: RPC socket file : /home/ec2-user/.pm2/rpc.sock
PM2 | 2025-07-21T17:39:21: PM2 log: HUS socket file : /home/ec2-user/.pm2/pub.sock
PM2 | 2025-07-21T17:39:21: PM2 log: Application log path : /home/ec2-user/.pm2/logs
PM2 | 2025-07-21T17:39:21: PM2 log: Worker Interval : 30000
PM2 | 2025-07-21T17:39:21: PM2 log: Process dump file : /home/ec2-user/.pm2/dump.pm2
PM2 | 2025-07-21T17:39:21: PM2 log: Concurrent actions : 2
PM2 | 2025-07-21T17:39:21: PM2 log: SIGTERM timeout : 1600
PM2 | 2025-07-21T17:39:21: PM2 log: Runtime Binary : /home/ec2-user/.nvm/versions/node/v16.20.2/bin/node
PM2 | 2025-07-21T17:39:21: PM2 log:
PM2 | 2025-07-21T17:39:21: PM2 log: App [index:0] starting in -fork mode-
PM2 | 2025-07-21T17:39:21: 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

⚡
[ec2-user@ip-10-0-10-133 app-tier]$
```

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

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.

1

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

The response should look like the following:

2

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

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

3

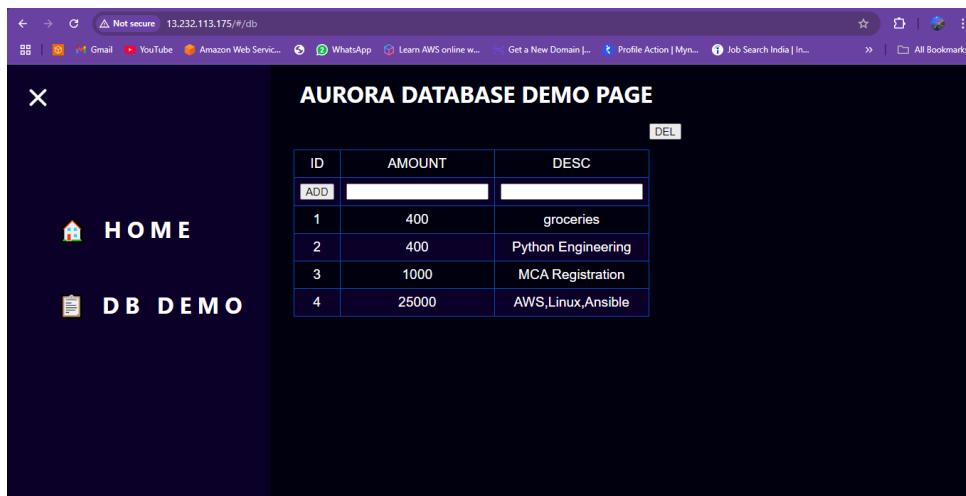
```
curl http://localhost:4000/transaction
```

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

4

```
{"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"}]}
```

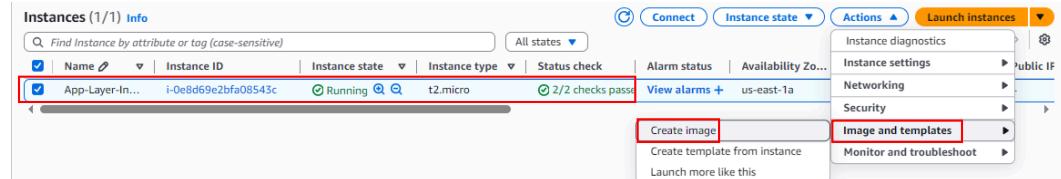
If you see both of these responses, then your networking, security, database and app configurations are correct.



Internal Load Balancing and Auto Scaling

Navigate to Instances on the left hand side of the EC2 dashboard. Select the app tier instance we created and under Actions select Image and templates. Click Create Image.

Give it a name App-Tier-image → description (Apptier) → click on create image



After create image we need Target group

Create image Info

An Image (also referred to as an AMI) defines the programs and settings that are applied when you launch an EC2 instance. You can create an image from the configuration of an existing instance.

Instance ID: i-0ed00cf5209570a99 (AppLayer)

Image name: AppTierImage ①

Image description - optional: App Tier ②

No reboot: Enable

Instance volumes:

Storage type	Device	Snapshot	Size	Volume type	IOPS	Throughput	Delete on termination	Encrypted
EBS	/dev/...	Create new snapshot fr...	8	EBS General Purpose S...	100		<input checked="" type="checkbox"/> Enable	<input type="checkbox"/> Enable

Add volume

During the image creation process, Amazon EC2 creates a snapshot of each of the above volumes.

Tags - optional

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Tag image and snapshots together
Tag the image and the snapshots with the same tag.

Tag image and snapshots separately
Tag the image and the snapshots with different tags.

No tags associated with the resource.

Now we create Target Group → click on create TG → choose to target type → instance → give it a TG name (App Tier target group) → http (4000) → select custom VPC → in the health check path → /health → click on next

Target group name

App-Tier-Target-Group

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Protocol

Protocol for load balancer-to-target communication. Can't be modified after creation.

HTTP

Port

Port number where targets receive traffic. Can be overridden for individual targets during registration.

4000

1-65535

VPC

Select the VPC with the instances that you want to include in the target group. Only VPCs that support the IP address type selected above are available in this list.

3-Tier-VPC
vpc-0377c4b02fb8c2fca
IPv4 VPC CIDR: 10.0.0.0/16

Health checks
The associated load balancer periodically sends requests, per the settings below, to the registered targets to test their status.

Health check protocol
HTTP ▾

Health check path
Use the default path of "/" to perform health checks on the root, or specify a custom path if preferred.
/health
Up to 1024 characters allowed.

Now register targets → enter the port value is 4000Check mark to target group and include as pending below → click on target group

Register targets
This is an optional step to create a target group. However, to ensure that your load balancer routes traffic to this target group you must register your targets.

Available instances (1)

Instance ID	Name	State	Security groups	Zone	Private IPv4 address	Subnet ID
i-0e8d69e2bfa08543c	App-Layer-Instance	Running	Private-instance-sg	us-east-1a	10.0.10.153	subnet-0a2c09e8b8e5d80c

0 selected

Ports for the selected instances
Ports for routing traffic to the selected instances.
4000
1-65535 (separate multiple ports with commas)
Include as pending below

1 selection is now pending below. Include more or register targets when ready.

Review targets

Targets (1)

Instance ID	Name	Port	State	Security groups	Zone	Private IPv4 address	Subnet ID	Launch time
i-0e8d69e2bfa08543c	App-Layer-Instance	4000	Running	Private-instance-sg	us-east-1a	10.0.10.153	subnet-0a2c09e8b8e5d80c	July 21, 2025, 20:45 (UTC+05:30)

1 pending

Cancel Previous Create target group

Now Target Group is Created

Successfully created the target group App-Tier-Target-Group. Anomaly detection is automatically applied to all registered targets. Results can be viewed in the Targets tab.

App-Tier-Target-Group

Details

aws.amazon.com/elasticloadbalancing/us-east-1:148761672642/targetgroups/App-Tier-Target-Group/5342070fe706ba	Protocol : Port HTTP:4000	Protocol version HTTP/1
Target type Instance	Load balancer None associated	VPC vpc-0377c4bd02b8c2fca
IP address type IPv4		

Total targets: 1
Healthy: 0
Unhealthy: 0
Anomalous: 0

Unused: 1
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.

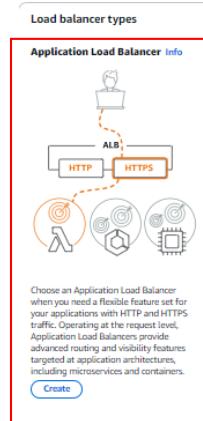
Targets **Monitoring** **Health checks** **Attributes** **Tags**

Registered targets (1) Info

Target groups route requests to individual registered targets using the protocol and port number specified. Health checks are performed on all registered targets according to the target group's health check settings. Anomaly detection is automatically applied to HTTP/HTTPS target groups with at least 3 healthy targets.

Instance ID	Name	Port	Zone	Health status	Health status details	Administrative override	Override details
i-0e8d69e2bfa08543c	App-Layer-Instance	4000	us-east-1a	Unused	Target group is not c...	-	-

After creating target group → we create Application Load Balancer



Click on create Load Balancer → Select (ALB) → Give the name (App-tier internal LB)
→ scheme select (instance) → select custom VPC

Network mapping Info
The load balancer routes traffic to targets in the selected subnets, and in accordance with your IP address settings.

VPC | Info
The load balancer will exist and scale within the selected VPC. The selected VPC is also where the load balancer targets must be hosted unless routing to Lambda or on-premises targets, or if using VPC peering. To confirm the VPC for your targets, view [target groups](#). For a new VPC, [create a VPC](#).

3-Tier-VPC
vpc-0377c4b02fb8c2fca
IPv4 VPC CIDR: 10.0.0.0/16

Select subnet and AZ (select Private App subnet AZ1 and AZ2) →

Availability Zones and subnets | Info
Select at least two Availability Zones and a subnet for each zone. A load balancer node will be placed in each selected zone and will automatically scale in response to traffic. The load balancer routes traffic to targets in the selected Availability Zones only.

<input checked="" type="checkbox"/> us-east-1a (use1-az1)	Subnet Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses are required for your load balancer to scale efficiently. subnet-032c09e8b8e5d80de IPv4 subnet CIDR: 10.0.10.0/24	Private-App-Subnet-AZ-1
<input checked="" type="checkbox"/> us-east-1c (use1-az1)	Subnet Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses are required for your load balancer to scale efficiently. subnet-0696fb7e77fc0f3 IPv4 subnet CIDR: 10.0.40.0/24	Private-App-Subnet-AZ-2

→ In security Group (internal-lb-sg)

Security groups Info
A security group is a set of firewall rules that control the traffic to your load balancer. Select an existing security group, or you can [create a new security group](#).

Security groups

Select up to 5 security groups

internal-lb-sg
sg-02d851950c5b2ea7c VPC: vpc-0377c4b02fb8c2fca

Here Listener and Routing → select protocol http 80 → add (App-Tier-TG)

Listeners and routing Info
A listener is a process that checks for connection requests using the port and protocol you configure. The rules that you define for a listener determine how the load balancer routes requests to its registered targets.

Listener HTTP:80

Protocol: HTTP	Port: 80
----------------	----------

Default action | Info

Forward to: App-Tier-Target-Group	HTTP
-----------------------------------	------

Review and Create Load Balancer ↴

Review

Review the load balancer configurations and make changes if needed. After you finish reviewing the configurations, choose **Create load balancer**.

Summary

Review and confirm your configurations. [Estimate cost](#)

Basic configuration [Edit](#)
Name: App-Tier-Internal Load Balancer
Scheme: Internal
IP address type: IPv4

Network mapping [Edit](#)
VPC: [vpc-0377c4b02fb8c2fc](#)
Availability Zones and subnets:

- us-east-1c [subnet-02c09a8b8e5d90de](#) Private-App-Subnet-AZ-1
- us-east-1c [subnet-0696effb67c7fae0f3](#) Private-App-Subnet-AZ-2

Security groups [Edit](#)
internal-lb-sg [sg-02d851950c5b2ea7c](#)

Listeners and routing [Edit](#)
HTTP:80 | Target group: [App-Tier-Target-Group](#)

Service integrations [Edit](#)
AWS WAF: -
AWS Global Accelerator: -

Tags [Edit](#)
-

Attributes

(*) Certain default attributes will be applied to your load balancer. You can view and edit them after creating the load balancer.

Creation workflow and status

► **Server-side tasks and status**
After completing and submitting the above steps, all server-side tasks and their statuses become available for monitoring.

Create load balancer

Successfully created load balancer: AppTier-Internal-Load-Balancer
It might take a few minutes for your load balancer to fully set up and route traffic. Targets will also take a few minutes to complete the registration process and pass initial health checks.

AppTier-Internal-Load-Balancer

Details

Load balancer type: Application
Status: Provisioning
Scheme: Internal
Hosted zone: Z35SXDOTRQ7X7K

VPC: [vpc-0377c4b02fb8c2fc](#)
Availability Zones:

- us-east-1c [subnet-0696effb67c7fae0f3](#) (us-east-1c (use1-az1))
- us-east-1a [subnet-0a2c09eb8b8e5d90de](#) (us-east-1a (use1-az4))

Load balancer IP address type: IPv4
Date created: July 21, 2025, 23:49 (UTC+05:30)

DNS name info: [internal-AppTier-Internal-Load-Balancer-2045026502.us-east-1.elb.amazonaws.com \(A Record\)](#)

Listeners and rules (1) [Info](#)

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-Target-Group (100%) • Target group stickiness: Off	1 rule	ARN	Not applicable	Not applicable	Not applicable	Not applicable

Before we configure Auto Scaling, we need to create a Launch template with the AMI we created earlier. On the left side of the EC2 dashboard navigate to Launch Template under Instances and click Create Launch Template.

Click on create launch → give the name → (app tier launch template) →

Launch template name and description

Launch template name - *required*

Must be unique to this account. Max 128 chars. No spaces or special characters like '&', '*', '@'.

Template version description

Select Custom AMI (click on my ami) → select (App-tier Image)



▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

Search our full catalog including 1000s of application and OS images

[AMI from catalog](#) [Recents](#) [My AMIs](#) 

[Quick Start](#)

Name	APP-Tier-Image				
Description	APP-Tier				
Image ID	ami-0f19570f41ba44e757				
Username	root (Check with the AMI provider.)				
Catalog	Published	Architecture	Virtualization	Root device type	ENI Enabled
My AMIs	2025-07-21T17:56:11.000Z	x86_64	hvm	ebs	Yes
Boot mode uefi-preferred					

Select instance type (t2micro)

Processed without keypair

Don't include subnet

▼ Instance type [Info](#) | [Get advice](#)

[Advanced](#)

Instance type

t2.micro 
 Family: t2 1 vCPU 1 GiB Memory Current generation: true On-Demand Windows base pricing: 0.0162 USD per Hour
 On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour On-Demand SUSE base pricing: 0.0116 USD per Hour
 On-Demand RHEL base pricing: 0.026 USD per Hour On-Demand Linux base pricing: 0.0116 USD per Hour

All generations

[Compare instance types](#)

Additional costs apply for AMIs with pre-installed software

Select Security Group (Private-instance-SG)

Firewall (security groups) | [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Select existing security group Create security group

Security groups | [Info](#)

[Select security groups](#)

Private-instance-sg sg-0ddf388e2e761b077 
 VPC: vpc-0377c4b02fb8c2fca

[Compare security group rules](#)

Click on Advanced Details

Click on IAM instance profile (select ec2-s3 role we created in the beginning)

Click on launch template

▼ Advanced details [Info](#)

IAM instance profile | [Info](#)

Todays-ec2-s3-role
 arn:aws:iam::148761672643:instance-profile/Todays-ec2-s3-role

[Create new IAM profile](#)

We will now create the Auto Scaling Group for our app instances.

Click on create auto scaling group → give the name (App-tier-ASG)

Select template (App-tier-template)

Choose launch template Info
Specify a launch template that contains settings common to all EC2 instances that are launched by this Auto Scaling group.

Name
Auto Scaling group name
Enter a name to identify the group.
App Tier ASG

You can't change the name of an Auto Scaling group in the current Region and no more than 255 characters.

Launch template Info

For accounts created after May 31, 2023, the EC2 console only supports creating Auto Scaling groups with launch templates. Creating Auto Scaling groups with launch configurations is not recommended but still available via the CLI and API until December 31, 2023.

Launch template
Choose a launch template that contains the instance-level settings, such as the Amazon Machine Image (AMI), instance type, key pair, and security groups.
AppTierLaunchTemplate

Create a launch template

Version
Default (1)

Description
Launch template for app tier

AMI ID
ami-0f19570f41ba4e75

Key pair name

Security groups

Request Spot Instances
No

Security group IDs
sg-0d9ff388e2e761b077

Additional details

Storage (volumes)

Date created
Tue Jul 22 2025 00:06:55 GMT+0530 (India Standard Time)

Next

Select custom vpc → select AZ and Subnet (Private app subnet AZ1 and AZ2)

Choose instance launch options Info
Choose the VPC network environment that your instances are launched into, and customize the instance types and purchase options.

Instance type requirements Info
You can keep the same instance attributes or instance type from your launch template, or you can choose to override the launch template by specifying different instance attributes or manually adding instance types.

Launch template
AppTierLaunchTemplate

Version
Default

Description
Launch template for app tier

Instance type
t2.micro

Override launch template

Network Info
For most applications, you can use multiple Availability Zones and let EC2 Auto Scaling balance your instances across the zones. The default VPC and default subnets are suitable for getting started quickly.

VPC
Choose the VPC that defines the virtual network for your Auto Scaling group.
vpc-0377c4b2fb8c2fc4 (3-Tier-VPC)

Availability Zones and subnets
Define which Availability Zones and subnets your Auto Scaling group can use in the chosen VPC.

Select Availability Zones and subnets

use1-az1 (us-east-1c) | subnet-0696efb67e7fac0f5 (Private-App-Subnet-AZ-2)

use1-az4 (us-east-1a) | subnet-0a2c09e8b8e5d80de (Private-App-Subnet-AZ-1)

Create a subnet

Availability Zone distribution - new
Auto Scaling automatically balances instances across Availability Zones. If launch failures occur in a zone, select a strategy.

Balanced best effort
If launches fail in one Availability Zone, Auto Scaling will attempt to launch in another healthy Availability Zone.

Balanced only
If launches fail in one Availability Zone, Auto Scaling will continue to attempt to launch in the unhealthy Availability Zone to preserve balanced distribution.

Next

Click on load balancing → selected (Attach to an existing load balancer)
Select (App-Tier-Target Group)

Step 5 - optional

- Integrate with other services**
- Step 4 - optional**
- Configure group size and scaling**
- Step 5 - optional**
- Add notifications**
- Step 6 - optional**
- Add tags**
- Step 7**
- Review**

Load balancing Info
Use the options below to attach your Auto Scaling group to an existing load balancer, or to a new load balancer that you define.

No load balancer
Traffic to your Auto Scaling group will not be fronted by a load balancer.

Attach to an existing load balancer
Choose from your existing load balancers.

Attach to a new load balancer
Quickly create a basic load balancer to attach to your Auto Scaling group.

Attach to an existing load balancer
Select the load balancers that you want to attach to your Auto Scaling group.

Choose from your load balancer target groups
This option allows you to attach Application, Network, or Gateway Load Balancers.

Choose from Classic Load Balancers

Existing load balancer target groups
Only instance target groups that belong to the same VPC as your Auto Scaling group are available for selection.

Select target group

App-Tier-Target-Group [1]
Application Load Balancer: AppTier-internal-Load-Balancer

To improve networking capabilities and consistency, integrate your Auto Scaling group with VPC Lattice. VPC Lattice facilitates communications between AWS services and helps you connect and manage your applications across compute services in AWS.

Health checks
Health checks increase availability by replacing unhealthy instances. When you use multiple health checks, all are evaluated, and if at least one fails, instance replacement occurs.

EC2 health checks
 Always enabled

Additional health check types - optional | Info

- Turn on Elastic Load Balancing health checks | Recommended
EC2 Auto Scaling will start to detect and act on health checks performed by Elastic Load Balancing. To avoid unexpected terminations, first verify the settings of these health checks in the [Load Balancer console](#).
- Turn on VPC Lattice health checks
VPC Lattice can monitor whether instances are available to handle requests. If it considers a target as failed a health check, EC2 Auto Scaling replaces it after its next periodic check.
- Turn on Amazon EBS health checks
EBS monitors whether an instance's root volume or attached volume stalls. When it reports an unhealthy volume, EC2 Auto Scaling can replace the instance on its next periodic health check.

Health check grace period | Info
This time period delays the first health check until your instances finish initializing. It doesn't prevent an instance from terminating when placed into a non-running state.
120 seconds

Cancel | Skip to review | Previous | **Next**

Select the group size and scaling

Desired Capacity 2

Minimum capacity 2

Maximum Capacity 4

Select (no scaling policies)

Configure group size and scaling - optional | Info

Define your group's desired capacity and scaling limits. You can optionally add automatic scaling to adjust the size of your group.

Group size | Info

Set the initial size of the Auto Scaling group. After creating the group, you can change its size to meet demand, either manually or by using automatic scaling.

Desired capacity type
Choose the unit of measurement for the desired capacity value. vCPUs and Memory(GiB) are only supported for mixed instances groups configured with a set of instance attributes.

Units (number of instances) ▾

Desired capacity
Specify your group size.

Scaling | Info

You can resize your Auto Scaling group manually or automatically to meet changes in demand.

Scaling limits
Set limits on how much your desired capacity can be increased or decreased.

Min desired capacity <input type="text" value="2"/>	Max desired capacity <input type="text" value="4"/>
Equal or less than desired capacity	Equal or greater than desired capacity

Automatic scaling - optional

Choose whether to use a target tracking policy | Info
You can set up other metric-based scaling policies and scheduled scaling after creating your Auto Scaling group.

No scaling policies
Your Auto Scaling group will remain at its initial size and will not dynamically resize to meet demand.

Target tracking scaling policy
Choose a CloudWatch metric and target value and let the scaling policy adjust the desired capacity in proportion to the metric's value.

Cancel | Skip to review | Previous | **Next**

In additional setting

Enable Monitoring

Additional settings

Instance scale-in protection
If protect from scale in is enabled, newly launched instances will be protected from scale in by default.
 Enable instance scale-in protection

Monitoring | Info
 Enable group metrics collection within CloudWatch

Default instance warmup | Info
The amount of time that CloudWatch metrics for new instances do not contribute to the group's aggregated instance metrics, as their usage data is not reliable yet.
 Enable default instance warmup

Cancel | Skip to review | Previous | **Next**

Add Tags

Review and Create Auto Scaling Group

Step 1
Choose launch template
Step 2
Choose instance launch options
Step 3 - optional
Step 4 - optional
Configure group size and scaling
Step 5 - optional
Add notifications
Step 6 - optional
Add tags
Step 7
Review

Add tags - optional Info

Add tags to help you search, filter, and track your Auto Scaling group across AWS. You can also choose to automatically add these tags to instances when they are launched.

You can optionally choose to add tags to instances (and their attached EBS volumes) by specifying tags in your launch template. We recommend caution, however, because the tag values for instances from your launch template will be overridden if there are any duplicate keys specified for the Auto Scaling group.

Tags (1)

Key	Value - optional	Tag new instances
Name	Web-Tier-VMs	<input checked="" type="checkbox"/>

Add tag 49 remaining

Cancel **Previous** **Next**

Auto Scaling groups (1) Info

Last updated less than a minute 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
App-Tier-ASG	AppTierLaunch-Template Version Def. 2	-	-	2	2	4	2 Availability Zones	Tue Jul 22 2025 00:27:23 GMT+0530 (India Standard Time)

Here are 2 instances automatically created

Instances (3) Info

Find Instance by attribute or tag (case-sensitive)

All status

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zon...	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6 IPs	Monitoring	Security
Web-Tier-VMs	i-008ebbb750880fe33	Running	t2.micro	Initializing	View alarms +	us-east-1c	-	-	-	-	disabled	Private
Web-Tier-VMs	i-05f570acef827127c	Running	t2.micro	Initializing	View alarms +	us-east-1a	-	-	-	-	disabled	Private
App-Layer-In...	i-0e8d69w2fa08543c	Running	t2.micro	2/2 checks pass	View alarms +	us-east-1a	-	-	-	-	disabled	Private

Web Tier Instance Deployment

We will deploy an EC2 instance for the web tier and make all necessary software configurations for the NGINX web server and React.js website.

Update NGINX Configuration Files

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.

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

```
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 #   }
```

Launch New Instance the Name is Web-App-Layer

Launch an instance [Info](#)

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags [Info](#)

Name

WEB-App-Layer

[Add additional tags](#)

Select Amazon Machine

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

[Recent](#) | [My AMIs](#) | **Quick Start**

[Amazon Linux](#)  [macOS](#)  [Ubuntu](#)  [Windows](#)  [Red Hat](#)  [SUSE Linux](#)  [Debian](#) 

[Browse more AMIs](#)
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2023 kernel-6.12 AMI
ami-050fd9796aa387c0d (64-bit (x86), uefi-preferred) / ami-0f537b3ea6616cd4a (64-bit (Arm), uefi)
Virtualization: hvm ENA enabled: true Root device type: ebs Free tier eligible ▾

Description
Amazon Linux 2023 (kernel-6.12) is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.

Amazon Linux 2023 AMI 2023.8.20250707.0 x86_64 HVM kernel-6.12

Architecture	Boot mode	AMI ID	Publish Date	Username
64-bit (x86) ▾	uefi-preferred	ami-050fd9796aa387c0d	2025-07-08	ec2-user Verified provider

Select custom VPC

Select subnet (Private-App-Subnet-AZ1)

Select Security Groups (Private-Instance SG)

▼ Network settings [Info](#)

VPC - required | [Info](#)

vpc-0377c4b02fb8c2fca (3-Tier-VPC)
10.0.0.0/16

Subnet | [Info](#)

subnet-0a2c09e8b8e5d80de Private-App-Subnet-AZ-1
VPC: vpc-0377c4b02fb8c2fca Owner: 148761672643 Availability Zone: us-east-1a
Zone type: Availability Zone IP addresses available: 248 CIDR: 10.0.10.0/24

Create new subnet

Auto-assign public IP | [Info](#)

Disable

Firewall (security groups) | [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group Select existing security group

Common security groups | [Info](#)

Select security groups

Private-instance-sg sg-0dddf388e2e761b077 X
VPC: vpc-0377c4b02fb8c2fca

Compare security group rules

Security groups that you add or remove here will be added to or removed from all your network interfaces.

► Advanced network configuration

Click on Advanced Details (in the IAM Profile Select ec2-s3-role)

Launch Instance

After Launch instance connect to Session Manager

Instances (1/4) Info											
Find Instance by attribute or tag (case-sensitive) All states											
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zon...	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6 IPs	Monitoring
Web-Tier-VMs	i-008ebbb250880fe33	Running Q Q	t2.micro	2/2 checks pass	View alarms +	us-east-1c	-	-	-	-	disabled Private
Web-Tier-VMs	i-05f570aceef827127c	Running Q Q	t2.micro	2/2 checks pass	View alarms +	us-east-1a	-	-	-	-	disabled Private
App-Layer-In...	i-0e8d69w2fa08543c	Running Q Q	t2.micro	2/2 checks pass	View alarms +	us-east-1a	-	-	-	-	disabled Private
WEB-App-Layer	i-00843e7c15cef7e7	Running Q Q	t2.micro	2/2 checks pass	View alarms +	us-east-1a	-	-	-	-	disabled Private

Enter command

Sudo -su ec2-user

Configure Web Instance

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

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

```
2 source ~/.bashrc
```

```
3 nvm install 16
```

```
4 nvm use 16
```

- 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:

```
1 cd ~/web-tier
```

```
2 npm install
```

```
3 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:

```
1 cd /etc/nginx
```

```
2 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:

```
1 sudo rm nginx.conf
```

```
2 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
```

- 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.

```
sh-5.2$ sudo -su ec2-user
[ec2-user@ip-10-0-10-184 bin]$ cd ~/web-tier
[ec2-user@ip-10-0-10-184 web-tier]$ npm run build

> aws-3tier-web-layer@0.1.0 build
> react-scripts build

Creating an optimized production build...
Compiled successfully.

File sizes after gzip:

 74.86 kB  build/static/js/main.9f58f4bc.js
  1.78 kB  build/static/js/453.5846ab7c.chunk.js
   493 B    build/static/css/main.b20b6ac4.css

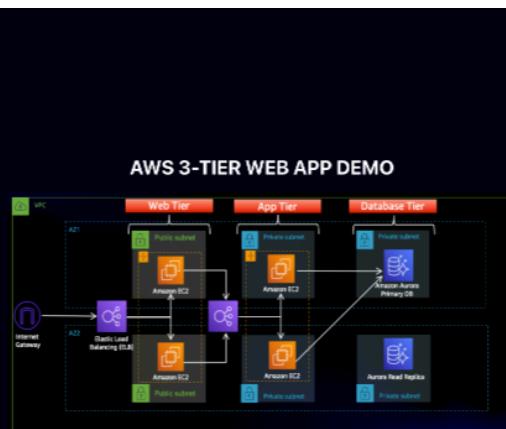
The project was built assuming it is hosted at ./.
You can control this with the homepage field in your package.json.

The build folder is ready to be deployed.

Find out more about deployment here:

  https://cra.link/deployment

[ec2-user@ip-10-0-10-184 web-tier]$
```



External Load Balancer and Auto Scaling

We will create an Amazon Machine Image (AMI) of the web tier instance we just created, and use that to set up autoscaling with an external facing load balancer in order to make this tier highly available.

Create Webserver-Image

Create image Info

An image (also referred to as an AMI) defines the programs and settings that are applied when you launch an EC2 instance. You can create an image from the configuration of an existing instance.

Image details

Instance ID
i-00843e7cc15cef7e (WEB-App-Layer)

Image name
WebServer-image Maximum 127 characters. Can't be modified after creation.

Image description - optional
Maximum 255 characters

Reboot instance
When selected, Amazon EC2 reboots the instance so that data is at rest when snapshots of the attached volumes are taken. This ensures data consistency.

Instance volumes

Storage type	Device	Snapshot	Size	Volume type	IOPS	Throughput	Delete on termination	Encrypted
EBS	/dev/xv...	Create new snapshot fro...	8	EBS General Purpose SSD ...	3000		<input checked="" type="checkbox"/> Enable	<input type="checkbox"/> Enable

Add volume

During the image creation process, Amazon EC2 creates a snapshot of each of the above volumes.

Tags - optional
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Tag image and snapshots together
Tag the image and the snapshots with the same tag.

Tag image and snapshots separately
Tag the image and the snapshots with different tags.

No tags associated with the resource.

Add new tag
You can add up to 50 more tags.

Create image

After create Image we need Target Group

Define Ports

Select custom vpc

Click on create Target Group

Target group name
Web-Server-TG Maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Protocol
Protocol for load balancer-to-target communication. Can't be modified after creation.

Port
Port number where targets receive traffic. Can be overridden for individual targets during registration.

IP address type
Only targets with the indicated IP address type can be registered to this target group.

IPv4
Each instance has a default network interface (eth0) that is assigned the primary private IPv4 address. The instance's primary private IPv4 address is the one that will be applied to the target.

IPv6
Each instance you register must have an assigned primary IPv6 address. This is configured on the instance's default network interface (eth0). [Learn more](#)

VPC
Select the VPC with the instances that you want to include in the target group. Only VPCs that support the IP address type selected above are available in this list.

3-Tier-VPC
vpc-03774b02fb8c2fcfa
IPv4 VPC CIDR: 10.0.0.0/16

Create Application Load Balancer (Give it a name) Web-Tier-External-lb

Select scheme (Now at this time we select internet-facing)

Create Application Load Balancer [Info](#)
The Application Load Balancer distributes incoming HTTP and HTTPS traffic across multiple targets such as Amazon EC2 instances, microservices, and containers, request, it evaluates the listener rules in priority order to determine which rule to apply, and if applicable, it selects a target from the target group for the rule act

► How Application Load Balancers work

Basic configuration

Load balancer name
Name must be unique within your AWS account and can't be changed after the load balancer is created.

Scheme [Info](#)
Scheme can't be changed after the load balancer is created.

Internet-facing

- Serves internet-facing traffic.
- Has public IP addresses.
- DNS name resolves to public IPs.
- Requires a public subnet.

Internal

- Serves internal traffic.
- Has private IP addresses.
- DNS name resolves to private IPs.
- Compatible with the **IPv4** and **Dualstack** IP address types.

Load balancer IP address type [Info](#)
Select the front-end IP address type to assign to the load balancer. The VPC and subnets mapped to this load balancer must include the selected IP address type:

IPv4
Includes only IPv4 addresses.

Dualstack
Includes IPv4 and IPv6 addresses.

Dualstack without public IPv4
Includes a public IPv6 address, and private IPv4 and IPv6 addresses. Compatible with **internet-facing** load balancers only.

Select custom vpc

Select subnet (public web subnet AZ1 and AZ2)

Network mapping [Info](#)
The load balancer routes traffic to targets in the selected subnets, and in accordance with your IP address settings.

VPC [Info](#)
This load balancer will exist and scale within the selected VPC. The selected VPC is also where the load balancer targets must be hosted unless routing to Lambda or on-premises targets, or if using VPC peering. To confirm the VPC for your targets, view [target source](#) [Edit](#) For a new VPC, [create a VPC](#) [Edit](#)

T-Tier-VPC
T-Tier-VPC ID: 0ed49b2f9b2fcfa
IPv4 VPC CIDR: 10.0.0.0/16

IP pools - new [Info](#)
You can optionally choose to configure an IPAM pool as the preferred source for your load balancer's IP addresses. Create or view [Pools](#) in [Amazon VPC IP Address Manager console](#) [Edit](#)

Use IPAM pool for public IPv4 addresses

Availability Zones and subnets [Info](#)
Select at least two Availability Zones and a subnet for each zone. A load balancer node will be placed in each selected zone and will automatically scale in response to traffic. The load balancer routes traffic to targets in the selected Availability Zones only.

us-east-1a (use1-a2z) <input checked="" type="checkbox"/> Only CIDR blocks corresponding to the load balancer's address type are used. At least 8 available IP addresses are required for your load balancer to scale efficiently. subnet-04f00075b251391c29 (IPv4 subnet CIDR: 10.0.0.0/24)	Public-Subnet-AZ-1 <input checked="" type="checkbox"/>
us-east-1c (use1-a2z) <input checked="" type="checkbox"/> Only CIDR blocks corresponding to the load balancer's address type are used. At least 8 available IP addresses are required for your load balancer to scale efficiently. subnet-0ed49b2f9b2fcfa25 (IPv4 subnet CIDR: 10.0.30.0/24)	Public-Subnet-AZ-2 <input checked="" type="checkbox"/>

Security Group (internet facing lb-sg)

Security groups [Info](#)
A security group is a set of firewall rules that control the traffic to your load balancer. Select an existing security group, or you can [create a new security group](#) [Edit](#)

Security groups
Select up to 5 security groups

Internet-facing-lb-sg
sg-0b43b1cb5feef5b8ee VPC: vpc-0377c4b02fb8c2fcfa

Review and Create load balancer

Review
Review the load balancer configurations and make changes if needed. After you finish reviewing the configurations, choose [Create load balancer](#).

Summary
Review and confirm your configurations. [Estimate cost](#) [Edit](#)

Basic configuration Edit Name: Web-Tier-External-lb Scheme: Internet-facing IP address type: IPv4	Network mapping Edit VPC: vpc-0377c4b02fb8c2fcfa Public IPv4 IPAM pool: - Availability Zones and subnets: <ul style="list-style-type: none"> us-east-1a subnet-04f00075b251391c29 Public-Subnet-AZ-1 us-east-1c subnet-0ed49b2f9b2fcfa25 Public-Subnet-AZ-2 	Security groups Edit internet-facing-lb-sg sg-0b43b1cb5feef5b8ee	Listeners and routing Edit HTTP:80 Target group: Web-Server-TG
Service integrations Edit Amazon CloudFront + AWS Web Application Firewall (WAF): - AWS WAF: - AWS Global Accelerator: -	Tags Edit -		
Attributes <input checked="" type="checkbox"/> Certain default attributes will be applied to your load balancer. You can view and edit them after creating the load balancer.			

Creation workflow and status

► Server-side tasks and status
After completing and submitting the above steps, all server-side tasks and their statuses become available for monitoring.

[Cancel](#) [Create load balancer](#)

Create template for Auto-scaling Group

Create launch template

Creating a launch template allows you to create a saved instance configuration that can be reused, shared and launched at a later time. Templates can have multiple versions.

Launch template name and description

Launch template name - required

Must be unique to this account. Max 128 chars. No spaces or special characters like '&', '*', '@'.

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

Recent | My AMIs | Quick Start

Don't include in launch template

Owned by me

Shared with me

Indicates AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

WebServer-Image

ami-0e480ea264fb7e28
2025-07-21T21:05:32.000Z Virtualization: hvm ENA enabled: true Root device type: ebs Boot mode: uefi-preferred

Description

Architecture

x86_64

AMI ID

ami-0e480ea264fb7e28

▼ Network settings [Info](#)

Subnet | [Info](#)

Don't include in launch template

Create new subnet

When you specify a subnet, a network interface is automatically added to your template.

Availability Zone [Info](#)

Don't include in launch template

Enable additional zones

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Select existing security group

Create security group

Security groups [Info](#)

Select security groups

X

AURORA DATABASE DEMO PAGE

HOME

DB DEMO ←

ID	AMOUNT	DESC
ADD		
1	400	groceries

Advanced details [Info](#)

IAM instance profile: [Info](#)
Todays-ec2-s3-role
arn:aws:iam::148761672643:instance-profile/Todays-ec2-s3-role

[Create new IAM profile](#)

Create launch template

Launch Templates (2) [Info](#)

Launch Template ID	Launch Template Name	Default Version	Latest Version	Create Time	Created By	Managed	Operator
lt-06aea19bf088d4261	WebServerLaunch-Template	1	1	2025-07-21T21:19:44.000Z	arn:aws:iam::148761672643:root	false	-
lt-0fa1c1623cde5a5dc	AppTierLaunch-Template	1	1	2025-07-21T18:36:55.000Z	arn:aws:iam::148761672643:root	false	-

Create Auto-scaling group → give the name → webserver -ASG → select Template (WebserverLaunch-Template)

Name

Auto Scaling group name
Enter a name to identify the group.
WebServer-ASG

Must be unique to this account in the current Region and no more than 255 characters.

Launch template [Info](#)

For accounts created after May 31, 2023, the EC2 console only supports creating Auto Scaling groups with launch configurations. Creating Auto Scaling groups with launch configurations is not recommended but still available via the CLI and API until December 31, 2023.

Launch template
Choose a launch template that contains the instance-level settings, such as the Amazon Machine Image (AMI), instance type, key pair, and security groups.
WebServerLaunch-Template

[Create a launch template](#)

Version
Default (1) [Create a launch template version](#)

Description
-

Launch template
WebServerLaunch-Template (lt-06aea19bf088d4261)

Instance type
t2.micro

AMI ID
ami-0e4a80ea264fb7e28

Security groups
-

Request Spot Instances
No

Key pair name
-

Security group IDs
[sg-0221b6973fdbear912](#)

Additional details

Storage (volumes)
-

Date created
Tue Jul 22 2025 02:49:44 GMT+0530 (India Standard Time)

[Cancel](#) **Next**

Select VPC

Select AZ and Subnet (public-web-subnet-AZ1 and AZ2)

Network [Info](#)

For most applications, you can use multiple Availability Zones and let EC2 Auto Scaling balance your instances across the zones. The default VPC and default subnets are suitable for getting started quickly.

VPC
Choose the VPC that defines the virtual network for your Auto Scaling group.
vpc-0377c4b02fb8c2fca (3-Tier-VPC)

[Create a VPC](#)

Availability Zones and subnets
Define which Availability Zones and subnets your Auto Scaling group can use in the chosen VPC.
Select Availability Zones and subnets

use1-az1 (us-east-1a) | subnet-0ed49b2f3b0f4ce25 (Public-Web-Subnet-AZ-2)
10.0.0.0/24

use1-az4 (us-east-1a) | subnet-04f0075b251391c29 (Public-Web-Subnet-AZ-1)
10.0.0.0/24

[Create a subnet](#)

Availability Zone distribution - new
Auto Scaling automatically balances instances across Availability Zones. If launch failures occur in a zone, select a strategy.

Balanced best effort
If launches fail in one Availability Zone, Auto Scaling will attempt to launch in another healthy Availability Zone.

Balanced only
If launches fail in one Availability Zone, Auto Scaling will continue to attempt to launch in the unhealthy Availability Zone to preserve balanced distribution.

[Cancel](#) [Skip to review](#) [Previous](#) **Next**

In the load balancing → select existing load balancer that we created (Web-server-TG)

Load balancing Info

Use the options below to attach your Auto Scaling group to an existing load balancer, or to a new load balancer that you define.

No load balancer
Traffic to your Auto Scaling group will not be fronted by a load balancer.

Attach to an existing load balancer
Choose from your existing load balancers.

Attach to a new load balancer
Quickly create a basic load balancer to attach to your Auto Scaling group.

Attach to an existing load balancer

Select the load balancers that you want to attach to your Auto Scaling group.

Choose from your load balancer target groups
This option allows you to attach Application, Network, or Gateway Load Balancers.

Choose from Classic Load Balancers

Existing load balancer target groups

Only instance target groups that belong to the same VPC as your Auto Scaling group are available for selection.

Select target groups Web-Server-TG | HTTP Application Load Balancer: Web-Tier-External-lb C

Configure group and scaling size

Desired capacity 2

Minimum Capacity 2

And maximum capacity 4

Configure group size and scaling - optional Info

Define your group's desired capacity and scaling limits. You can optionally add automatic scaling to adjust the size of your group.

Group size Info

Set the initial size of the Auto Scaling group. After creating the group, you can change its size to meet demand, either manually or by using automatic scaling.

Desired capacity type

Choose the unit of measurement for the desired capacity value. vCPUs and Memory(GiB) are only supported for mixed instances groups configured with a set of instance attributes.

Units (number of instances)

Desired capacity

Specify your group size:

Scaling Info

You can resize your Auto Scaling group manually or automatically to meet changes in demand.

Scaling limits

Set limits on how much your desired capacity can be increased or decreased.

Min desired capacity Equal or less than desired capacity

Max desired capacity Equal or greater than desired capacity

Automatic scaling - optional

Choose whether to use a target tracking policy | Info

You can set up other metric-based scaling policies and scheduled scaling after creating your Auto Scaling group.

No scaling policies
Your Auto Scaling group will remain at its initial size and will not dynamically resize to meet demand.

Target tracking scaling policy
Choose a CloudWatch metric and target value and let the scaling policy adjust the desired capacity in proportion to the metric's value.

Give the Tag

And create Auto scaling

Add tags - optional Info

Add tags to help you search, filter, and track your Auto Scaling group across AWS. You can also choose to automatically add these tags to instances when they are launched.

(i) You can optionally choose to add tags to instances (and their attached EBS volumes) by specifying tags in your launch template. We recommend caution, however, because the tag values for instances from your launch template will be overridden if there are any duplicate keys specified for the Auto Scaling group.

Tags (1)

Key	Value - optional	Tag new instances
Name	Web-Server-VM1	<input checked="" type="checkbox"/>

Add tag
49 remaining

Cancel Previous Next

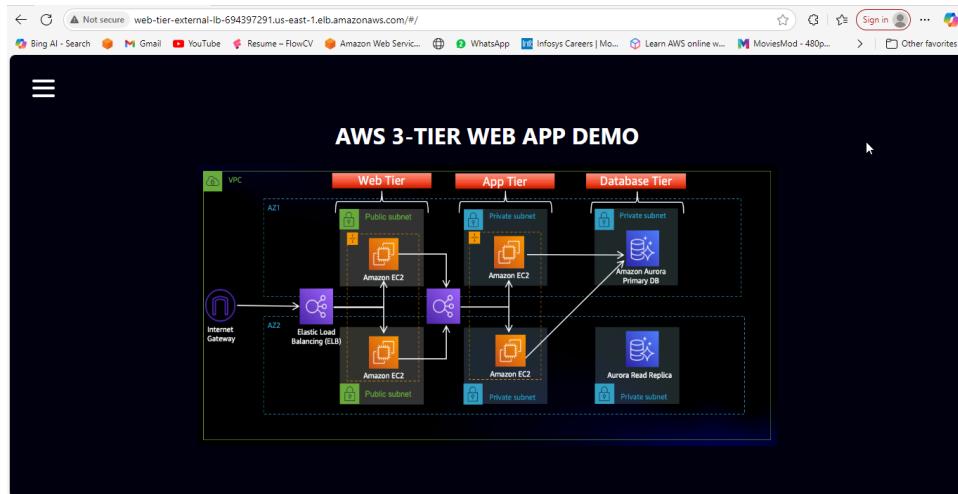
After creating Auto-scaling group

Go to load balancer → there is a Web-server-external-LB → copy the DNS Name and paste it into new browser

The screenshot shows the AWS EC2 Auto Scaling Groups page. At the top, a green notification bar says "WebServer-ASG created successfully". Below it, the table lists two Auto Scaling groups:

Name	Launch template/configuratio...	Instan...	Status	Desired capa...	M...	M...	Availability Zo...	Creation time
WebServer-ASG	WebServerLaunch-Templat...	Version 1	Updating capacity.	2	2	4	2 Availability Zones	Tue Jul 22 2025 02:56:18 GMT+0530 (India Standard Time)
App-Tier-ASG	AppTierLaunch-Templat...	Version Def	-	2	2	4	2 Availability Zones	Tue Jul 22 2025 00:27:23 GMT+0530 (India Standard Time)

You should now have your external load balancer and autoscaling group configured correctly. You should see the autoscaling group spinning up 2 new web tier instances. If you want to test if this is working correctly, you can delete one of your new instances manually and wait to see if a new instance is booted up to replace it. To test if your entire architecture is working, navigate to your external facing loadbalancer, and plug in the DNS name into your browser.



The screenshot shows a web application titled "AURORA DATABASE DEMO PAGE". The interface includes a navigation menu on the left with options "HOME" and "DB DEMO". The main content area displays a table with the following data:

ID	AMOUNT	DESC
ADD		
6	149	Mobile
7	199	Basic
8	499	Standard
9	649	Premium

Successfully I've Implemented a 3 Tier Web Architecture!