

# AWS 3 Tier Web Architecture Practical

## **Step-1: Create S3 bucket**

- Navigate to the S3 service in the AWS console and create a new S3 bucket.
- Give it a unique name, and then leave all the defaults as in. Make sure to select the region that you intend to run this whole lab in. This bucket is where we will upload our code later.

The screenshot shows the AWS S3 Buckets page. On the left, there's a sidebar with options like General purpose buckets, Directory buckets, Table buckets, Vector buckets, Access Grants, Access Points (General Purpose Buckets, FSx file systems), Access Points (Directory Buckets), Object Lambda Access Points, Multi-Region Access Points, and Batch Operations. The main area is titled "General purpose buckets" and shows one bucket named "mytestbucket-demo12". The bucket details are: Name: mytestbucket-demo12, AWS Region: US East (N. Virginia) us-east-1, Creation date: October 4, 2025, 18:04:16 (UTC+05:30). There are buttons for Copy ARN, Empty, Delete, and Create bucket.

## **Step-2: IAM EC2 Instance Role Creation**

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

The screenshot shows the AWS IAM Roles page. On the left, there's a sidebar with Identity and Access Management (IAM) options like Dashboard, Access management (User groups, Users, Roles, Policies, Identity providers, Account settings, Root access management), and Access reports (Access Analyzer, Resource analysis, Unused access). The main area is titled "Roles (7)" and lists seven roles: AWSReservedSSO\_SystemAdministrator\_dc86c1d4c0f04bcc, AWSServiceRoleForECS, AWSServiceRoleForOrganizations, AWSServiceRoleForSSO, AWSServiceRoleForSupport, AWSServiceRoleForTrustedAdvisor, and MyFirstFunction-role-ewvm7bqv. Each role has a "Trusted entities" column showing its provider and last activity. There are buttons for Delete and Create role.

- Select EC2 as the trusted entity and click on next.

Step 1 **Select trusted entity**

Step 2 Add permissions

Step 3 Name, review, and create

## Select trusted entity Info

### Trusted entity type

- AWS service**  
Allow AWS services like EC2, Lambda, or others to perform actions in this account.
- AWS account**  
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
- Web identity**  
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.
- SAML 2.0 federation**  
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
- Custom trust policy**  
Create a custom trust policy to enable others to perform actions in this account.

### Use case

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

#### Service or use case

EC2

Choose a use case for the specified service.

**Use case**

- EC2**  
Allows EC2 instances to call AWS services on your behalf.
- EC2 Role for AWS Systems Manager**  
Allows EC2 instances to call AWS services like CloudWatch and Systems Manager on your behalf.
- EC2 Spot Fleet Role**  
Allows EC2 Spot Fleet to request and terminate Spot Instances on your behalf.

- When adding permissions, include the following AWS managed policies. You can search for them and select them. These policies will allow our instances to download our code from S3 and use Systems Manager Session Manager to securely connect to our instances without SSH keys through the AWS console.
  - AmazonSSMManagedInstanceCore**
  - AmazonS3ReadOnlyAccess**

☰ IAM > Roles > Create role

Step 1 Select trusted entity

Step 2 **Add permissions**

Step 3 Name, review, and create

## Add permissions Info

### Permissions policies (1/1076) Info

Choose one or more policies to attach to your new role.

Filter by Type			
<input type="text" value="AmazonSSM"/>	<input type="button" value="X"/>	All types	9 matches
<input type="checkbox"/> Policy name <input type="button" value="C"/>	Type	Description	
<input type="checkbox"/> <input type="checkbox"/> AmazonSSMAutomationApproverAccess	AWS managed	Provides access to view automation ex...	
<input type="checkbox"/> <input type="checkbox"/> AmazonSSMAutomationRole	AWS managed	Provides permissions for EC2 Automati...	
<input type="checkbox"/> <input type="checkbox"/> AmazonSSMDirectoryServiceAccess	AWS managed	This policy allows SSM Agent to access...	
<input type="checkbox"/> <input type="checkbox"/> AmazonSSMFullAccess	AWS managed	Provides full access to Amazon SSM.	
<input type="checkbox"/> <input type="checkbox"/> AmazonSSMMaintenanceWindowRole	AWS managed	Service Role to be used for EC2 Mainte...	
<input type="checkbox"/> <input type="checkbox"/> AmazonSSMManagedEC2InstanceDefaultPolicy	AWS managed	This policy enables AWS Systems Man...	
<input checked="" type="checkbox"/> <input type="checkbox"/> AmazonSSMManagedInstanceCore	AWS managed	The policy for Amazon EC2 Role to ena...	
<input type="checkbox"/> <input type="checkbox"/> AmazonSSMPatchAssociation	AWS managed	Provide access to child instances for pa...	
<input type="checkbox"/> <input type="checkbox"/> AmazonSSMReadOnlyAccess	AWS managed	Provides read only access to Amazon S...	

- Step 1 Select trusted entity
- Step 2 Add permissions
- Step 3 Name, review, and create

## Add permissions Info

### Permissions policies (2/1076) Info

Choose one or more policies to attach to your new role.

Filter by Type

<input type="checkbox"/> Policy name	Type	Description
<input type="checkbox"/> <a href="#">AmazonS3FullAccess</a>	AWS managed	Provides full access to all buckets via t...
<input type="checkbox"/> <a href="#">AmazonS3ObjectLambdaExecutionRolePolicy</a>	AWS managed	Provides AWS Lambda functions permi...
<input type="checkbox"/> <a href="#">AmazonS3OutpostsFullAccess</a>	AWS managed	Provides full access to Amazon S3 on ...
<input type="checkbox"/> <a href="#">AmazonS3OutpostsReadOnlyAccess</a>	AWS managed	Provides read only access to Amazon S...
<input checked="" type="checkbox"/> <a href="#">AmazonS3ReadOnlyAccess</a>	AWS managed	Provides read only access to all bucket...
<input type="checkbox"/> <a href="#">AmazonS3TablesFullAccess</a>	AWS managed	Provides full access to all S3 table buc...
<input type="checkbox"/> <a href="#">AmazonS3TablesLakeFormationServiceRole</a>	AWS managed	This managed policy grants AWS Lake ...
<input type="checkbox"/> <a href="#">AmazonS3TablesReadOnlyAccess</a>	AWS managed	Provides read only access to all S3 tabl...

- Step 1 Select trusted entity
- Step 2 Add permissions
- Step 3 Name, review, and create

## Name, review, and create

### Role details

#### Role name

Enter a meaningful name to identify this role.

Myec2demo-role

Maximum 64 characters. Use alphanumeric and '+=\_,@-\_` characters.

#### Description

Add a short explanation for this role.

Allows EC2 instances to call AWS services on your behalf.

Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tabs, new lines, or any of the following characters: \_+=., @~\[\]!#\$%^&\*();`^`

## Step 1: Select trusted entities

### Trust policy

```

1  [
2    "Version": "2012-10-17",
3    "Statement": [
4      {
5        "Effect": "Allow",
6        "Action": [
7          "sts:AssumeRole"
8        ],
9        "Principal": {
10          "Service": [
11            "ec2.amazonaws.com"
12          ]
13        }
14      }
15    ]
16  ]

```

## Step 2: Add permissions

Edit

### Permissions policy summary

#### Policy name

[AmazonS3ReadOnlyAccess](#)

[AmazonSSMManagedInstanceCore](#)

Type

AWS managed

Attached as

Permissions policy

Permissions policy

## Step 3: Add tags

### Add tags - optional Info

Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

Add new tag

You can add up to 50 more tags.

Cancel

Previous

Create role

### Step-3: Create VPC & Subnets

- Navigate to the VPC dashboard in the AWS console and navigate to Your VPCs on the left side.

The screenshot shows the AWS VPC dashboard with the 'Your VPCs' section. It lists two VPCs: 'vpc-0bee02294f4bf63d6' (Available, Off, 172.31.0.0/16) and 'MydemoVPC' (Available, Off, 10.0.0.0/16). The 'MydemoVPC' row is selected. The left sidebar shows 'Virtual private cloud' with 'Your VPCs' and 'Subnets' options.

- Next, create your subnets by navigating to Subnets on the left side of the dashboard and clicking Create subnet. 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.

The screenshot shows the AWS VPC dashboard with the 'Subnets' section. It lists six subnets across two availability zones (us-east-1a and us-east-1b). The subnets are: Public-Web-Subnet-AZ-2 (10.0.2.0/24), Public-Web-Subnet-AZ-1 (10.0.1.0/24), Private-DB-Subnet-AZ-2 (10.0.6.0/24), Private-DB-Subnet-AZ-1 (10.0.5.0/24), Private-App-Subnet-AZ-2 (10.0.4.0/24), and Private-App-Subnet-AZ-1 (10.0.3.0/24). The left sidebar shows 'Virtual private cloud' with 'Subnets' option selected.

This screenshot shows a detailed view of the subnets table from the previous screenshot. It includes columns for IPv4 CIDR, IPv6 CIDR, IPv6 CIDR association ID, Available IPv4 addresses, Availability Zone, and Network border group. The data remains the same as the first screenshot.

### Step-4: Create Internet Gateway and configure routing

- To give the public subnets in our VPC internet access we will have to create and attach an Internet Gateway. On the left-hand side of the VPC dashboard, select Internet Gateways. Create your internet gateway by simply giving it a name and clicking Create internet gateway. After creating the internet gateway, attach it to your VPC.

The screenshot shows the AWS VPC dashboard with the 'Internet gateways' section. It lists one internet gateway: 'igw-0826ed05636bb4bdc / mydemo-igw'. The status is 'Detached'. The left sidebar shows 'Virtual private cloud' with 'Internet gateways' option selected.

Internet gateways (1/2) <a href="#">Info</a>				
<input type="text"/> Find internet gateways by attribute or tag				
Name	Internet gateway ID	State	VPC ID	Owner
-	igw-013b404533a8ad404	Attached	ypc-0bee02294f4bf63d6	076526621834
<input checked="" type="checkbox"/> mydemo-igw	igw-0826ed05636bb4bdc	Attached	ypc-030cb26268efc5870   MydemoVPC	076526621834

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

[VPC](#) > [NAT gateways](#) > [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.

Mydemo-ngw

The name can be up to 256 characters long.

#### Subnet

Select a subnet in which to create the NAT gateway.

subnet-0f2886d3284a9068b (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-08f8e80fb3f9d98d6

[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

Name

#### Value - optional

Mydemo-ngw

[Remove](#)

[VPC](#) > [NAT gateways](#)

NAT gateways (2) <a href="#">Info</a>								
<input type="text"/> Find NAT gateways by attribute or tag								
Name	NAT gateway ID	Connectivity...	State	State message	Primary public I...	Primary private I...	Primary netw...	
NAT-GW-AZ2	nat-0078a9ea747ada5c5	Public	Available	-	13.223.113.138	10.0.2.92	eni-0e218bd2	
NAT-GW-AZ1	nat-0973bc45f2d7a9039	Public	Available	-	3.228.243.203	10.0.1.93	eni-04ec4f3cd	

- Navigate to Route Tables on the left side of the VPC dashboard and click Create route table First, let's create one route table for the web layer public subnets and name it accordingly.

**Create route table** Info

A route table specifies how packets are forwarded between the subnets within your VPC, the internet, and your VPN connection.

**Route table settings**

Create a tag with a key of 'Name' and a value that you specify.

PublicRouteTable

**VPC**

The VPC to use for this route table.

vpc-030cb26268efc5870 (MydemoVPC)

**Tags**

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

**Key**

Name

**Value - optional**

PublicRouteTable

X

Remove

[Add new tag](#)

You can add 49 more tags.

Cancel

[Create route table](#)

- After creating the route table, you'll automatically be taken to the details page. Scroll down and click on the Routes tab and Edit routes. Add a route that directs traffic from the VPC to the internet gateway. In other words, for all traffic destined for IPs outside the VPC CIDR range, add an entry that directs it to the internet gateway as a target. Save the changes.

**Edit routes****Destination**

10.0.0.0/16

**Target**

local

**Status**

Active

**Propagated**

No

**Route Origin**

CreateRouteTable

[Add route](#)

Q 0.0.0.0/0 X

Internet Gateway

Q igw-0826ed05636bb4bdc X

Use: "igw-0826ed05636bb4bdc"

igw-0826ed05636bb4bdc (mydemo-igw)

Cancel

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

- Edit the Explicit Subnet Associations of the route table by navigating to the route table details again. Select Subnet Associations and click Edit subnet associations. Select the two-web layer public subnets you created earlier and click Save associations.

**VPC dashboard****AWS Global View**

Filter by VPC: ▾

**Virtual private cloud**

Your VPCs

Subnets

**Route tables**

Internet gateways

Egress-only Internet gateways

Carrier gateways

DHCP option sets

Elastic IPs

Managed prefix lists

NAT gateways

**rtb-0b4eb1ca3d3243b65 / PublicRouteTable**

Actions ▾

**Details** Info**Route table ID**

rtb-0b4eb1ca3d3243b65

**Main**

No

**Explicit subnet associations**

2 subnets

**Edge associations**

--

[Routes](#)[Subnet associations](#)[Edge associations](#)[Route propagation](#)[Tags](#)**Explicit subnet associations (2)**[Edit subnet associations](#)

Find subnet association

Name

Subnet ID

IPv4 CIDR

IPv6 CIDR

Public-Web-Subnet-AZ-1

subnet-0f2886d3284a9068b

10.0.1.0/24

-

Public-Web-Subnet-AZ-2

subnet-0a4e0fa6239091e93

10.0.2.0/24

-

- Now create 2 more route tables, one for each app layer private subnet in each availability zone. These route tables will route app layer traffic destined for outside the VPC to the NAT gateway in the respective availability zone, so add the appropriate routes for that.

VPC > Route tables

VPC dashboard <

AWS Global View

Virtual private cloud

- Your VPCs
- Subnets
- Route tables

Route tables (1/4) Info

Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC	Last updated	Actions	Create route table
-	<a href="#">rtb-0ac8cd34ae006a6eb</a>	-	-	Yes	<a href="#">vpc-0bee02294f4bf63d6</a>	1 minute ago		
-	<a href="#">rtb-0905735037471bb9d</a>	-	-	Yes	<a href="#">vpc-030cb26268efc5870   Myd...</a>			
PublicRouteTable	<a href="#">rtb-0b4eb1ca3d3243b65</a>	2 subnets	-	No	<a href="#">vpc-030cb26268efc5870   Myd...</a>			
<input checked="" type="checkbox"/> Private-RT-AZ1	<a href="#">rtb-04f87a18af897fb4</a>	-	-	No	<a href="#">vpc-030cb26268efc5870   Myd...</a>			

VPC > Route tables > rtb-04f87a18af897fb4 > Edit routes

## Edit routes

Destination: 10.0.0.0/16

Target: local (Active)

NAT Gateway: nat- (Use: "nat-")

Add route

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	local	Active	No	CreateRouteTable
0.0.0.0/0	NAT Gateway	-	No	CreateRoute
0.0.0.0/0	nat-	-	-	-

Cancel Preview Save changes

VPC > Route tables

VPC dashboard <

Route tables (1/5) Info

Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC	Last updated	Actions	Create route table
-	<a href="#">rtb-0ac8cd34ae006a6eb</a>	-	-	Yes	<a href="#">vpc-0bee02294f4bf63d6</a>	less than a minute ago		
-	<a href="#">rtb-0905735037471bb9d</a>	-	-	Yes	<a href="#">vpc-030cb26268efc5870   Myd...</a>			
PublicRouteTable	<a href="#">rtb-0b4eb1ca3d3243b65</a>	2 subnets	-	No	<a href="#">vpc-030cb26268efc5870   Myd...</a>			
<input checked="" type="checkbox"/> Private-RT-AZ1	<a href="#">rtb-04f87a18af897fb4</a>	subnet-03220334593e61...	-	No	<a href="#">vpc-030cb26268efc5870   Myd...</a>			
<input checked="" type="checkbox"/> Private-RT-AZ2	<a href="#">rtb-0bcb017869a7d2cff</a>	-	-	No	<a href="#">vpc-030cb26268efc5870   Myd...</a>			

VPC > Route tables > rtb-0bcb017869a7d2cff > Edit routes

## Edit routes

Destination: 10.0.0.0/16

Target: local (Active)

NAT Gateway: nat- (Use: "nat-")

Add route

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	local	Active	No	CreateRouteTable
0.0.0.0/0	NAT Gateway	-	No	CreateRoute
0.0.0.0/0	nat-	-	-	-

Cancel Preview Save changes

- Once the route tables are created and routes added, add the appropriate subnet associations for each of the app layer private subnets.

VPC > Route tables > rtb-04f87a18af897fb4

VPC dashboard <

rtb-04f87a18af897fb4 / Private-RT-AZ1

Actions

Details Info

Route table ID: <a href="#">rtb-04f87a18af897fb4</a>	Main: <input type="checkbox"/> No	Explicit subnet associations: subnet-03220334593e6148e / Private-App-Subnet-AZ-1	Edge associations: -
VPC: <a href="#">vpc-030cb26268efc5870   MydemoVPC</a>	Owner ID: <a href="#">076526621834</a>		

Routes Subnet associations Edge associations Route propagation Tags

Explicit subnet associations (1)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
Private-App-Subnet-AZ-1	<a href="#">subnet-03220334593e6148e</a>	10.0.3.0/24	-

Edit subnet associations

The screenshot shows the AWS VPC Route Tables page. The top navigation bar includes 'VPC' > 'Route tables' > 'rtb-0bcb017869a7d2cff'. On the left sidebar, under 'Virtual private cloud', 'Route tables' is selected. The main content area displays the details of the route table 'rtb-0bcb017869a7d2cff / Private-RT-AZ2'. The 'Subnet associations' tab is active, showing one explicit subnet association: 'Private-App-Subnet-AZ-2' associated with 'subnet-01604a8aab303c194'. There are tabs for 'Routes', 'Edge associations', 'Route propagation', and 'Tags'.

## Step-5: Create Security Groups

- Security groups will tighten the rules around which traffic will be allowed to our Elastic Load Balancers and EC2 instances. Navigate to Security Groups on the left side of the VPC dashboard, under Security. 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 AWS Security Groups page. The top navigation bar includes 'VPC' > 'Security Groups' > 'sg-0912fb5302f362b05 - internet-facing-lb-sg'. On the left sidebar, under 'Virtual private cloud', 'Route tables' is selected. The main content area displays the details of the security group 'sg-0912fb5302f362b05 - internet-facing-lb-sg'. The 'Inbound rules' tab is active, showing one inbound rule: 'sgr-0c489307256771af1' allowing IPv4 traffic on port 80 from source IP 10.0.4.0/24. There are tabs for 'Outbound rules', 'Sharing - new', 'VPC associations - new', and 'Tags'.

- 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-030cb26268efc5870 (MydemoVPC)

**Inbound rules** Info

Type	Protocol	Port range	Source	Description - optional	
HTTP	TCP	80	Custom	sg-0912fb5302f362b05	<a href="#">Delete</a>
HTTP	TCP	80	My IP	sg-0912fb5302f362b05	<a href="#">Delete</a>

[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

internal-lb-sg

Name cannot be edited after creation.

**Description** Info

SG for the internal load balancer

**VPC Info**

vpc-030cb26268efc5870 (MydemoVPC)

**Inbound rules** Info

Type	Protocol	Port range	Source	Description - optional	
HTTP	TCP	80	Custom	sg-03047ab4462a46643	<a href="#">Delete</a>

[Add rule](#)

- 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.

**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

PrivateInstanceSG

Name cannot be edited after creation.

**Description** Info

SG for our private app tier sg

**VPC** Info

vpc-030cb26268efc5870 (MydemoVPC)

**Inbound rules** Info

Type	Protocol	Port range	Source	Description - optional	
HTTP	TCP	80	Custom	sg-0785805455f94196e	<a href="#">Delete</a>
HTTP	TCP	80	My IP	sg-0785805455f94196e	<a href="#">Delete</a>

[Add rule](#)

- 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).

**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

DBSG

Name cannot be edited after creation.

**Description** Info

SG for our databases

**VPC** Info

vpc-030cb26268efc5870 (MydemoVPC)

**Inbound rules** Info

Type	Protocol	Port range	Source	Description - optional	
MYSQL/Aurora	TCP	3306	Custom	sg-0878d95cb5c2a7ae7	<a href="#">Delete</a>

[Add rule](#)**VPC dashboard****AWS Global View**[Filter by VPC](#)**Virtual private cloud**

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only Internet gateways

Carrier gateways

**Security Groups (6)** Info[Find security groups by attribute or tag](#)

VPC ID = vpc-030cb26268efc5870

[Clear filters](#)[Actions](#)[Export security groups to CSV](#)[Create security group](#)

Security group name	VPC ID	Description	Owner	Inbound rules count
PrivateInstanceSG	vpc-030cb26268efc5870	SG for our private app tier sg	076526621834	2 Permission entries
DBSG	vpc-030cb26268efc5870	SG for our databases	076526621834	1 Permission entry
internet-facing-lb-sg	vpc-030cb26268efc5870	External load balancer security group	076526621834	1 Permission entry
default	vpc-030cb26268efc5870	default VPC security group	076526621834	1 Permission entry
Web-tier-sg	vpc-030cb26268efc5870	SG for the web tier	076526621834	2 Permission entries
internal-lb-sg	vpc-030cb26268efc5870	SG for the internal load balancer	076526621834	1 Permission entry

**Step-6: Create Database Subnet Groups and deploy database**

- Navigate to the RDS dashboard in the AWS console and click on Subnet groups on the left hand side. Click Create DB subnet group. Give your subnet group a name, description, and choose the VPC we created.

When adding subnets, make sure to add the subnets we created in each availability zone specifically for our database layer, make sure you're selecting the correct subnet IDs.

Aura and RDS > Subnet groups > Create DB subnet group

### Create DB subnet group

To 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.

Must contain from 1 to 255 characters. Alphanumeric characters, spaces, hyphens, underscores, and periods are allowed.

**Description**

**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.  
▼  
6 Subnets, 2 Availability Zones

#### Add subnets

**Availability Zones**  
Choose the Availability Zones that include the subnets you want to add.  
▼

**Subnets**  
Choose the subnets that you want to add. The list includes the subnets in the selected Availability Zones.  
▼  
   
Subnet ID: subnet-0724b92f04c74de04 CIDR: 10.0.5.0/24  
Subnet ID: subnet-0a5abf34391bbe11f CIDR: 10.0.6.0/24

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-0724b92f04c74de04	10.0.5.0/24
us-east-1b	Private-DB-Subnet-AZ-2	subnet-0a5abf34391bbe11f	10.0.6.0/24

Aura and RDS > Subnet groups

Successfully created db-subnet-group. View subnet group

### Subnet groups (1)

Name	Description	Status	VPC
db-subnet-group	Subnet group for the database layer	Complete	vpc-030cb26268efc5870

- Navigate to Databases on the left hand side of the RDS dashboard and click Create database. Start with a Standard create for this MySQL-Compatible Amazon Aurora database. Leave the rest of the defaults in the Engine options as default.

Aura and RDS > Databases > Create database

### Create database

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
<input checked="" type="radio"/> Aurora (MySQL Compatible)	
<input type="radio"/> PostgreSQL	
<input type="radio"/> Microsoft SQL Server	
<input type="radio"/> Aurora (PostgreSQL Compatible)	
<input type="radio"/> MariaDB	
<input type="radio"/> IBM Db2	
<input type="radio"/> MySQL	
<input type="radio"/> Oracle	

- Under the Templates section choose Dev/Test since this isn't being used for production now. Under Settings set a username and password of your choice and note them down since we'll be using password authentication to access our database.

## Templates

Choose a sample template to meet your use case.

<input type="radio"/> Production Use defaults for high availability and fast, consistent performance.	<input checked="" type="radio"/> Dev/Test This instance is intended for development use outside of a production environment.
--	---

## Settings

### DB cluster identifier [Info](#)

Type a name for your DB cluster. The name must be unique across all DB clusters owned by your AWS account in the current AWS Region.

The DB cluster identifier is case-insensitive, but is stored as all lowercase (as in "mydbcluster"). 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 32 alphanumeric characters. The first character must be a letter.

#### Credentials management

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

<input type="radio"/> Managed in AWS Secrets Manager - most secure RDS generates a password for you and manages it throughout its lifecycle using AWS Secrets Manager.	<input checked="" type="radio"/> 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](#)

- Next, under Availability and durability change the option to create an Aurora Replica or reader node in a different availability zone. Under Connectivity, set the VPC, choose the subnet group we created earlier, and select no for public access.

## Cluster storage configuration [Info](#)

Choose the storage configuration for the Aurora DB cluster that best fits your application's price predictability and price performance needs.

### Configuration options

Database instance, storage, and I/O charges vary depending on the configuration. [Learn more](#)

<input checked="" type="radio"/> Aurora I/O-Optimized <ul style="list-style-type: none"> <li>Predictable pricing for all applications. Improved price performance for I/O-intensive applications (I/O costs &gt;25% of total database costs).</li> <li>No additional charges for read/write I/O operations. DB instance and storage prices include I/O usage.</li> </ul>	<input type="radio"/> Aurora Standard <ul style="list-style-type: none"> <li>Cost-effective pricing for many applications with moderate I/O usage (I/O costs &lt;25% of total database costs).</li> <li>Pay-per-request I/O charges apply. DB instance and storage prices don't include I/O usage.</li> </ul>
--	---

## 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

Include previous generation classes

Serverless v2

Memory optimized classes (includes r classes)

Burstable classes (includes t classes)

db.r5.large 2 vCPUs 16 GiB RAM Network: Up to 4,750 Mbps	▼
---	---

## Availability & durability

### Multi-AZ deployment [Info](#)

Create an Aurora Replica or Reader node in a different AZ (recommended for scaled availability)

Creates an Aurora Replica for fast failover and high availability.

Don't create an Aurora Replica

## Connectivity [Info](#)

### Compute resource

Choose whether to set up a connection to a compute resource for this database. Setting up a connection will automatically change connectivity settings so that the compute resource can connect to this database.

<input checked="" type="radio"/> Don't connect to an EC2 compute resource Don't set up a connection to a compute resource for this database. You can manually set up a connection to a compute resource later.	<input type="radio"/> Connect to an EC2 compute resource Set up a connection to an EC2 compute resource for this database.
---	---

### Network type [Info](#)

To use dual-stack mode, make sure that you associate an IPv6 CIDR block with a subnet in the VPC you specify.

<input checked="" type="radio"/> IPv4 Your resources can communicate only over the IPv4 addressing protocol.	<input type="radio"/> Dual-stack mode Your resources can communicate over IPv4, IPv6, or both.
---	---

### Virtual private cloud (VPC) [Info](#)

Choose the VPC. The VPC defines the virtual networking environment for this DB cluster.

MydemoVPC (vpc-030cb26268efc5870) 6 Subnets, 2 Availability Zones	▼
--	---

Only VPCs with a corresponding DB subnet group are listed.

<input type="checkbox"/> 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 cluster can use in the VPC that you selected.

**db-subnet-group**  
 2 Subnets, 2 Availability Zones

**Public access** [Info](#)

Yes  
 RDS assigns a public IP address to the cluster. Amazon EC2 instances and other resources outside of the VPC can connect to your cluster. Resources inside the VPC can also connect to the cluster. Choose one or more VPC security groups that specify which resources can connect to the cluster.

No  
 RDS doesn't assign a public IP address to the cluster. Only Amazon EC2 instances and other resources inside the VPC can connect to your cluster. Choose one or more VPC security groups that specify which resources can connect to the cluster.

**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

**DBSG X**

**RDS Proxy**  
 RDS Proxy is a fully managed, highly available database proxy that improves application scalability, resiliency, and security.

Create an RDS Proxy [Info](#)  
 RDS automatically creates an IAM role and a Secrets Manager secret for the proxy. RDS Proxy has additional costs. For more information, see [Amazon RDS Proxy pricing](#).

**Certificate authority - optional** [Info](#)  
 Using a server certificate provides an extra layer of security by validating that the connection is being made to an Amazon database. It does so by checking the server certificate that is automatically installed on all databases that you provision.

rds-ca-rsa2048-g1 (default)  
 Expiry: May 26, 2061

**Database authentication** [Info](#)  
 Password authentication is always active for your database engine. You can also turn on additional authentication methods for your database below.

IAM database authentication  
 Authenticates using IAM database authentication.

Kerberos authentication  
 Authenticates using Kerberos authentication through an AWS Directory Service for Microsoft Active Directory.

**Monitoring** [Info](#)  
 Choose monitoring tools for this database. Database Insights provides a combined view of Performance Insights and Enhanced Monitoring for your fleet of databases. **Database Insights** pricing is separate from RDS monthly estimates. See [Amazon CloudWatch pricing](#).

Database Insights - Advanced
 

- Retains 15 months of performance history
- Fleet-level monitoring
- Integration with CloudWatch Application Signals

Database Insights - Standard
 

- Retains 7 days of performance history, with the option to pay for the retention of up to 24 months of performance history

**Performance Insights**

Enable Performance insights  
 With Performance Insights dashboard, you can visualize the database load on your Amazon RDS DB instance load and filter the load by waits, SQL statements, hosts, or users.

**Retention period**  
 7 days

**AWS KMS key** [Info](#)  
 (default) aws/rds

- When your database is provisioned, you should see a reader and writer instance in the database subnets of each availability zone. Note down the writer endpoint for your database for later use.

## Step-7: App Tier Instance Deployment

- Navigate to the EC2 service dashboard and click on Instances on the left hand side. Then, click Launch Instances.
- Set the Name and Tag. Select Amazon Linux for the OS and Amazon Linux 2 AMI (HVM) - Kernel 5.10 for the AMI. For Instance Type, select t2.micro. (Select Proceed without a key pair.) For Network settings, click Edit.
- Select a VPC and a subnet. For Subnet, select Private-App-Subnet. For Security Group, select PrivateInstanceSG, which you created in the previous step. For Auto-assign public IP, select Disable. click Advanced details.
- In the IAM instance profile field, select the IAM Role you created in the previous step. (In our example, we created the EC2andS3Role.) Finally, click Launch instance to create the instance.

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with 'EC2' selected under 'Instances'. The main area is titled 'Instances (1/1) Info' and shows one instance. The instance details are as follows:

- Name: AppLayer
- Instance ID: i-09d439ac17f8499f9
- Instance state: Running
- Instance type: t2.micro
- Status check: Initializing
- Alarm status: View alarms +
- Availability Zone: us-east-1a
- Public IPv4 DNS: -
- Public IP: -

- When the instance state is running, connect to your instance. Switch to ec2-user by executing the following command in the browser terminal:
  - `sudo -su ec2-user`

Let's take this moment to make sure that we are able to reach the internet via our NAT gateways. If your network is configured correctly up till this point, you should be able to ping the google DNS servers:

- `ping 8.8.8.8`

You should see a transmission of packets. Stop it by pressing cntrl c.

Session ID: root-i2yngjb882nqouqqd5ye3suju8
Shortcuts
Instance ID: i-09d439ac17f8499f9

```
sh-5.2$ sudo -su ec2-user
[ec2-user@ip-10-0-3-112 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=112 time=2.17 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=112 time=1.80 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=112 time=1.81 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=112 time=1.77 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=112 time=1.85 ms
^C
--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 1.770/1.878/2.168/0.147 ms
[ec2-user@ip-10-0-3-112 bin]$
```

- Now configure database. Start by downloading the MySQL CLI.
  - `sudo dnf update -y`
  - `sudo wget https://dev.mysql.com/get/mysql84-community-release-el9-1.noarch.rpm`
  - `sudo dnf install mysql84-community-release-el9-1.noarch.rpm -y`
  - `sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql`
  - `sudo dnf install mysql-community-server -y`
  - `sudo systemctl enable --now mysqld`
- Initiate your DB connection with your Aurora RDS writer endpoint. In the following command, replace the RDS writer endpoint and the username, and then execute it in the browser terminal:
  - `mysql -h CHANGE-TO-YOUR-RDS-ENDPOINT -u CHANGE-TO-USER-NAME -p`
- You will then be prompted to type in your password. Once you input the password and hit enter, you should now be connected to your database.
- Create a database called webappdb with the following command using the MySQL CLI:
  - `CREATE DATABASE webappdb;`

- You can verify that it was created correctly with the following command:
  - SHOW DATABASES;
- 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;
- When finished, just type exit and hit enter to exit the MySQL client.

```
[ec2-user@ip-10-0-3-112 bin]$ sudo wget https://dev.mysql.com/get/mysql84-community-release-el9-1.noarch.rpm
--2025-10-04 15:50:06-- https://dev.mysql.com/get/mysql84-community-release-el9-1.noarch.rpm
Resolving dev.mysql.com (dev.mysql.com)... 23.1.48.99, 2600:1408:c400:188c::2e31, 2600:1408:c400:1881::2e31
Connecting to dev.mysql.com (dev.mysql.com)|23.1.48.99|:443... connected.
HTTP request sent, awaiting response... 302 Moved Temporarily
Location: https://repo.mysql.com//mysql84-community-release-el9-1.noarch.rpm [following]
--2025-10-04 15:50:06-- https://repo.mysql.com//mysql84-community-release-el9-1.noarch.rpm
Resolving repo.mysql.com (repo.mysql.com)... 23.66.209.41, 2600:1408:c400:1a9f::1d68
Connecting to repo.mysql.com (repo.mysql.com)|23.66.209.41|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 13139 (13K) [application/x-redhat-package-manager]
Saving to: 'mysql84-community-release-el9-1.noarch.rpm'

mysql84-community-release-el9-1.noarch.rpm    100%[=====] 12.83K --.-KB/s   in 0s

2025-10-04 15:50:06 (247 MB/s) - 'mysql84-community-release-el9-1.noarch.rpm' saved [13139/13139]

[ec2-user@ip-10-0-3-112 bin]$ sudo dnf install mysql84-community-release-el9-1.noarch.rpm -y
Last metadata expiration check: 0:01:42 ago on Sat Oct  4 15:48:35 2025.
Dependencies resolved.

=====
 Package           Architecture     Version       Repository   Size
=====
Installing:
 mysql84-community-release      noarch        el9-1        @commandline 13 k

Transaction Summary
=====
Install 1 Package

Total size: 13 k
Installed size: 14 k
Downloading Packages:
Running transaction check
transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing : 1/1
  Installing : mysql84-community-release-el9-1.noarch 1/1
  Verifying  : mysql84-community-release-el9-1.noarch 1/1

Installed:
  mysql84-community-release-el9-1.noarch

Complete!
```

```
[ec2-user@ip-10-0-3-112 bin]$ sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql
[ec2-user@ip-10-0-3-112 bin]$ sudo dnf install mysql-community-server -y
MySQL 8.4 LTS Community Server
MySQL Connectors Community
MySQL Tools 8.4 LTS Community
Dependencies resolved.

=====
Package           Architecture      Version       Repository   Size
=====
Installing:
mysql-community-server      x86_64        8.4.6-1.el9    mysql-8.4-lts-community 50 M
Installing dependencies:
mysql-community-client      x86_64        8.4.6-1.el9    mysql-8.4-lts-community 3.1 M
mysql-community-client-plugins x86_64        8.4.6-1.el9    mysql-8.4-lts-community 1.5 M
mysql-community-common      x86_64        8.4.6-1.el9    mysql-8.4-lts-community 578 K
mysql-community-icu-data-files x86_64        8.4.6-1.el9    mysql-8.4-lts-community 2.3 M
mysql-community-libs         x86_64        8.4.6-1.el9    mysql-8.4-lts-community 1.5 M

Transaction Summary
Install 6 Packages

Total download size: 59 M
Installed size: 331 M
Downloading Packages:
(1/6): mysql-community-common-8.4.6-1.el9.x86_64.rpm          6.5 MB/s | 578 kB  00:00
(2/6): mysql-community-client-8.4.6-1.el9.x86_64.rpm        28 MB/s | 3.1 MB  00:00
(3/6): mysql-community-libs-8.4.6-1.el9.x86_64.rpm          55 MB/s | 1.5 MB  00:00
(4/6): mysql-community-icu-data-files-8.4.6-1.el9.x86_64.rpm 32 MB/s | 2.3 MB  00:00
(5/6): mysql-community-client-plugins-8.4.6-1.el9.x86_64.rpm 6.4 MB/s | 1.5 MB  00:00
(6/6): mysql-community-server-8.4.6-1.el9.x86_64.rpm        79 MB/s | 50 MB   00:00

Total                                         76 MB/s | 59 MB  00:00
MySQL 8.4 LTS Community Server
Importing GPG key 0xA8D3785C:
[ec2-user@ip-10-0-3-112 bin]$ sudo systemctl enable --now mysqld
[ec2-user@ip-10-0-3-112 bin]$ mysql -h database-1.cluster-cclm2aewgne.us-east-1.rds.amazonaws.com -u admin -p takdew-3qegwe-cikwak
Enter password:
ERROR 1049 (42000): Unknown database 'takdew-3qegwe-cikwak'
[ec2-user@ip-10-0-3-112 bin]$ mysql -h database-1.cluster-cclm2aewgne.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 155
Server version: 8.0.39 8bc99e28

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 webapppdb;
Query OK, 1 row affected (0.00 sec)

mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| webapppdb |
+-----+
5 rows in set (0.01 sec)

mysql> USE webapppdb;
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 (0.02 sec)

mysql> SHOW TABLES;
+-----+
| Tables_in_webapppdb |
+-----+
| transactions |
+-----+
1 row in set (0.00 sec)

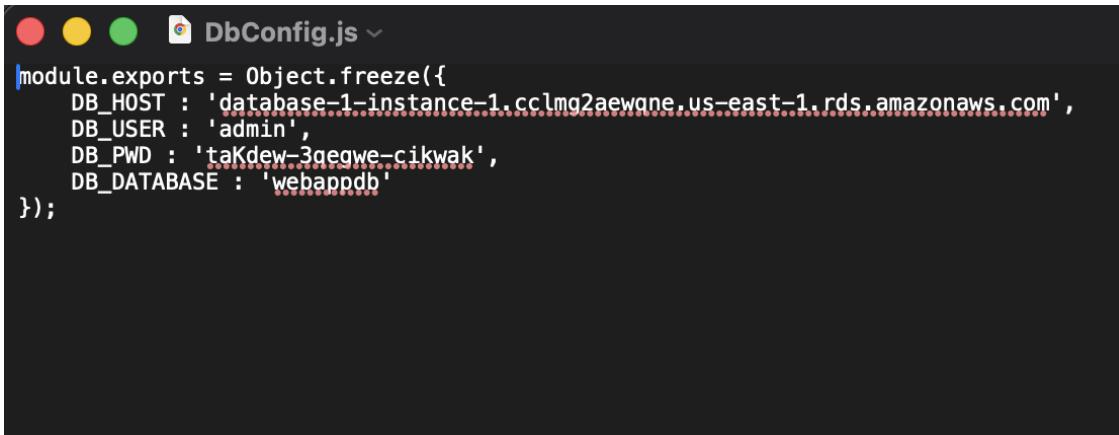
mysql> INSERT INTO transactions (amount,description) VALUES ('400','groceries');
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM transactions;
+----+----+-----+
| id | amount | description |
+----+----+-----+
| 1  | 400.00 | groceries  |
+----+----+-----+
1 row in set (0.01 sec)

mysql> exit
```

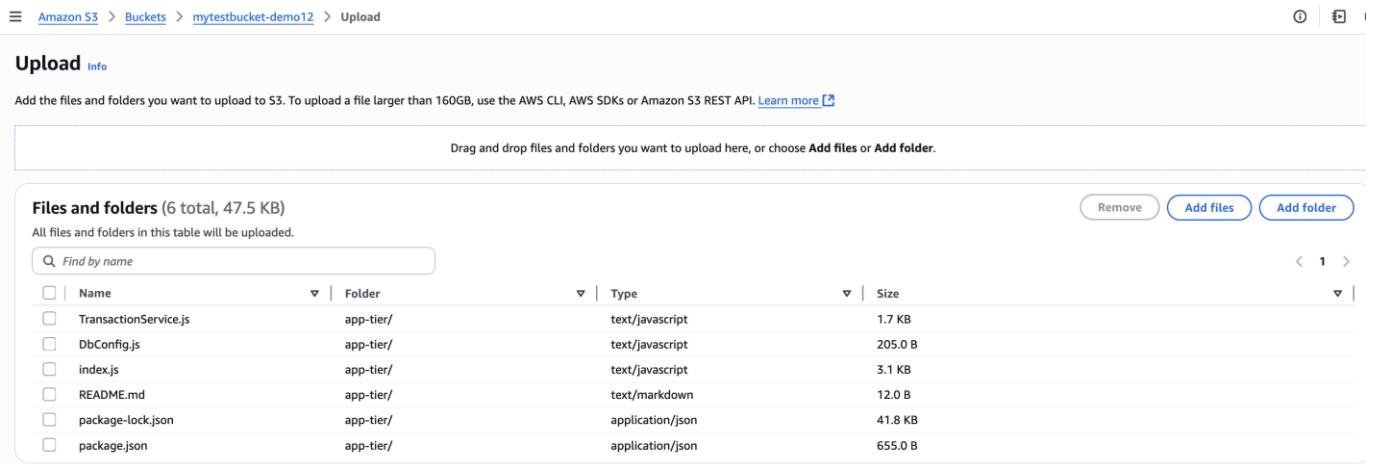
## Step-8: Configure App Instance

- 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 webapppdb for the database. Save the file.
- NOTE: This is NOT considered a best practice, and is done for the simplicity of the lab. Moving these credentials to a more suitable place like Secrets Manager is left as an extension for this workshop.



```
module.exports = Object.freeze({
  DB_HOST : 'database-1-instance-1.cclmg2aewgane.us-east-1.rds.amazonaws.com',
  DB_USER : 'admin',
  DB_PWD : 'taKdew-3qegwe-cikwak',
  DB_DATABASE : 'webappdb'
});
```

- Upload the app-tier folder to the S3 bucket that you created in part 0.



Amazon S3 > Buckets > mytestbucket-demo12 > Upload

**Upload** [Info](#)

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDKs or Amazon S3 REST API. [Learn more](#)

Drag and drop files and folders you want to upload here, or choose **Add files** or **Add folder**.

**Files and folders** (6 total, 47.5 KB)

All files and folders in this table will be uploaded.

<input type="checkbox"/>	Name	Folder	Type	Size
<input type="checkbox"/>	TransactionService.js	app-tier/	text/javascript	1.7 KB
<input type="checkbox"/>	DbConfig.js	app-tier/	text/javascript	205.0 B
<input type="checkbox"/>	index.js	app-tier/	text/javascript	3.1 KB
<input type="checkbox"/>	README.md	app-tier/	text/markdown	12.0 B
<input type="checkbox"/>	package-lock.json	app-tier/	application/json	41.8 KB
<input type="checkbox"/>	package.json	app-tier/	application/json	655.0 B

- 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
- Next, install a compatible version of Node.js and make sure it's being used
  - nvm install 16
  - nvm use 16
- 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
- Now we need to download our code from our s3 buckets onto our instance. In the command below, replace BUCKET\_NAME with the name of the bucket you uploaded the app-tier folder to:
  - Cd ~/
  - aws s3 cp s3:// mytestbucket-demo12/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
- To make sure the app is running correctly run the following:
  - pm2 list

- If you see a status of online, the app is running. If you see errored, then you need to do some troubleshooting. To look at the latest errors, use this command:
  - pm2 logs

NOTE: If you're having issues, check your configuration file for any typos, and double check that you have followed all installation commands till now.
- 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.
  - To setup the Startup Script, copy/paste the following command:  
`sudo env PATH=$PATH:/home/ec2-user/.nvm/versions/node/v16.20.2/bin /home/ec2-user/.nvm/versions/node/v16.20.2/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

```
[ec2-user@ip-10-0-3-112 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  444k   0  --::-- --::-- --::--  455k
-> 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" ] && \. "$NVM_DIR/bash_completion" # This loads nvm bash_completion
[ec2-user@ip-10-0-3-112 bin]$ source ~/.bashrc
[ec2-user@ip-10-0-3-112 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 checksum with sha256sum
Checksums matched!
Now using node v16.20.2 (npm v8.19.4)
Creating default alias: default -> 16 (-> v16.20.2)
[ec2-user@ip-10-0-3-112 bin]$ nvm use 16
Now using node v16.20.2 (npm v8.19.4)
```

```
[ec2-user@ip-10-0-3-112 bin]$ nvm use 16
Now using node v16.20.2 (npm v8.19.4)
[ec2-user@ip-10-0-3-112 bin]$ npm install -g pm2
added 133 packages, and audited 134 packages in 7s

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

found 0 vulnerabilities
npm notice
npm notice New major version of npm available! 8.19.4 -> 11.6.1
npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.6.1
npm notice Run npm install -g npm@11.6.1 to update!
npm notice
[ec2-user@ip-10-0-3-112 bin]$ cd ~
[ec2-user@ip-10-0-3-112 ~]$ aws s3 cp s3://mytestbucket-demo12/app-tier/ app-tier --recursive
download: s3://mytestbucket-demo12/app-tier/package.json to app-tier/package.json
download: s3://mytestbucket-demo12/app-tier/dbConfig.js to app-tier/dbConfig.js
download: s3://mytestbucket-demo12/app-tier/transactionService.js to app-tier/TransactionService.js
download: s3://mytestbucket-demo12/app-tier/package-lock.json to app-tier/package-lock.json
download: s3://mytestbucket-demo12/app-tier/index.js to app-tier/index.js
download: s3://mytestbucket-demo12/app-tier/README.md to app-tier/README.md
[ec2-user@ip-10-0-3-112 ~]$ cd ~/app-tier
[ec2-user@ip-10-0-3-112 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-10-0-3-112 app-tier]$ pm2 start index.js
-----
[PM2] Spawning PM2 daemon with pm2_home=/home/ec2-user/.pm2
[PM2] PM2 Successfully daemonized
[PM2] Starting /home/ec2-user/app-tier/index.js in fork_mode (1 instance)

[PM2] Spawning PM2 daemon with pm2_home=/home/ec2-user/.pm2
[PM2] PM2 Successfully daemonized
[PM2] Starting /home/ec2-user/app-tier/index.js in fork_mode (1 instance)
[PM2] Done.

[ec2-user@ip-10-0-3-112 app-tier]$ pm2 list
[ec2-user@ip-10-0-3-112 app-tier]$ pm2 startup
[PM2] Init System found: systemd
[PM2] To setup the Startup Script, copy/paste the following command:
sudo env PATH=$PATH:/home/ec2-user/.nvm/versions/node/v16.20.2/bin /home/ec2-user/.nvm/versions/node/v16.20.2/lib/node_modules/pm2/bin/pm2 startup systemd -u ec2-user --hp /home/ec2-user
[ec2-user@ip-10-0-3-112 app-tier]$ 
[ec2-user@ip-10-0-3-112 app-tier]$ pm2 save
[PM2] Saving current process list...
[PM2] Successfully saved in /home/ec2-user/.pm2/dump.pm2
[ec2-user@ip-10-0-3-112 app-tier]$ 
```

## Step-9: Test App Tier

- Now let's run a couple tests to see if our app is configured correctly and can retrieve data from the database. To hit out health check endpoint, copy this command into your SSM terminal. This is our simple health check endpoint that tells us if the app is simply running.
    - curl <http://localhost:4000/health>
  - The response should looks like the following:
    - "This is the health check"
  - Next, test your database connection. You can do that by hitting the following endpoint locally:
    - curl <http://localhost:4000/transaction>
  - You should see a response containing the test data we added earlier:
    - {"result": [{"id": 1, "amount": 400, "description": "groceries"}]}If you see both of these responses, then your networking, security, database and app configurations are correct. Congrats! Your app layer is fully configured and ready to go.

## **Step-10: 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.

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation pane with 'EC2' selected under 'Instances'. The main area shows one instance: 'AppLayer' (i-09d439ac17f8499f9), which is 'Running'. A context menu is open over this instance, with 'Create image' highlighted. Other options in the menu include 'Create template from instance' and 'Launch more like this'.

- Give the image a name and description and then click Create image. This will take a few minutes, but if you want to monitor the status of image creation you can see it by clicking AMIs under Images on the left hand navigation panel of the EC2 dashboard.

The screenshot shows the 'Create image' configuration page. It starts with 'Image details' for instance 'i-09d439ac17f8499f9 (AppLayer)'. The 'Image name' field is set to 'AppTierImage'. The 'Image description - optional' field contains 'App Tier'. There's a checkbox for 'Reboot instance'. Below these are volume settings: a table with columns 'Storage type', 'Device', 'Snapshot', 'Size', 'Volume type', 'IOPS', 'Throughput', 'Delete on termination', and 'Encrypted'. One EBS volume is listed with size 8, IOPS 3000, and throughput 100. The 'Delete on termination' checkbox is checked. An 'Add volume' button is available. A note says 'During the image creation process, Amazon EC2 creates a snapshot of each of the above volumes.' The 'Tags - optional' section has two radio button options: 'Tag image and snapshots together' (selected) and 'Tag image and snapshots separately'. A note below says 'No tags associated with the resource.' and an 'Add new tag' button is present. At the bottom are 'Cancel' and 'Create image' buttons.

- While the AMI is being created, we can go ahead and create our target group to use with the load balancer. On the EC2 dashboard navigate to Target Groups under Load Balancing on the left hand side. Click on Create Target Group.  
The purpose of forming this target group is to use with our load balancer so it may balance traffic across our private app tier instances. Select Instances as the target type and give it a name.
- Then, set the protocol to HTTP and the port to 4000. Remember that this is the port our Node.js app is running on. Select the VPC we've been using thus far, and then change the health check path to be /health. This is the health check endpoint of our app. Click Next.

- Step 1  
 [Create target group](#)  
 Step 2  
 Register targets

## Create target group

Your load balancer routes requests to the targets in a target group and performs health checks on the targets.

### Basic configuration

Settings in this section can't be changed after the target group is created.

#### Choose a target type

Instances

- Supports load balancing to instances within a specific VPC.
- Facilitates the use of [Amazon EC2 Auto Scaling](#) to manage and scale your EC2 capacity.

IP addresses

- Supports load balancing to VPC and on-premises resources.
- Facilitates routing to multiple IP addresses and network interfaces on the same instance.
- Offers flexibility with microservice based architectures, simplifying inter-application communication.
- Supports IPv6 targets, enabling end-to-end IPv6 communication, and IPv4-to-IPv6 NAT.

Lambda function

- Facilitates routing to a single Lambda function.
- Accessible to Application Load Balancers only.

Application Load Balancer

#### Target group name

AppTierTargetGroup

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.

80

1-65535

#### 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.

vpc-030cb26268efc5870 (MydemoVPC)  
10.0.0.0/16



[Create VPC](#)

#### Protocol version

HTTP1

Send requests to targets using HTTP/1.1. Supported when the request protocol is HTTP/1.1 or HTTP/2.

HTTP2

Send requests to targets using HTTP/2. Supported when the request protocol is HTTP/2 or gRPC, but gRPC-specific features are not available.

gRPC

Send requests to targets using gRPC. Supported when the request protocol is gRPC.

### 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.

[Advanced health check settings](#)

- We are NOT going to register any targets for now, so just skip that step and create the target group.

Step 1  
Create target group

Step 2

Register targets

## 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)

 Filter instances

<input type="checkbox"/> Instance ID	Name	State	Security groups	Zone
<input type="checkbox"/> i-09d439ac17f8499f9	AppLayer	Running	PrivateInstanceSG	us-east-1a

0 selected

### Ports for the selected instances

Ports for routing traffic to the selected instances.

80

1-65535 (separate multiple ports with commas)

 Include as pending below

## Review targets

### Targets (0)

 Filter targets

 Remove all pending

 < | 1 | > | 

Instance ID	Name	Port	State	Security groups	Zone	Private IPv4 address	Subnet ID	Launch time
-------------	------	------	-------	-----------------	------	----------------------	-----------	-------------

No instances added yet

Specify instances above, or leave the group empty if you prefer to add targets later.

0 pending

 Cancel

 Previous

 Create target group

- On the left hand side of the EC2 dashboard select Load Balancers under Load Balancing and click Create Load Balancer. We'll be using an Application Load Balancer for our HTTP traffic so click the create button for that option.
- After giving the load balancer a name, be sure to select internal since this one will not be public facing, but rather it will route traffic from our web tier to the app tier.
- Select the correct network configuration for VPC and private subnets.
- Select the security group we created for this internal ALB. Now, this ALB will be listening for HTTP traffic on port 80. It will be forwarding the traffic to our target group that we just created, so select it from the dropdown, and create the load balancer.

## 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, based on request attributes. When the load balancer receives a connection 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 action.

### ► 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.

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

##### 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 types. Public IPv4 addresses have an additional cost.

 IPv4

Includes only IPv4 addresses.

#### 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](#).

10.0.0.1/6

[Create VPC](#)

##### IP pools 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 the [Amazon VPC IP Address Manager console](#).

 Use IPAM pool for public IPv4 addresses

The IPAM pool you choose will be the preferred source of public IPv4 addresses. If the pool is depleted IPv4 addresses will be assigned by AWS.

##### 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-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.

  
IPv4 subnet CIDR: 10.0.3.0/24

Private-App-Subnet-AZ-1

 us-east-1b (use1-az2)

##### 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.

  
IPv4 subnet CIDR: 10.0.4.0/24

Private-App-Subnet-AZ-2

⚠ The selected subnet does not have a route to an internet gateway. This means that your load balancer will not receive internet traffic.

You can proceed with this selection; however, for internet traffic to reach your load balancer, you must update the subnet's route table in the [VPC console](#).

#### 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-0785805455f94196e VPC: vpc-030cb26268efc5870

## 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

[Remove](#)

#### Protocol

HTTP

#### Port

80

1-65535

#### Default action Info

The default action is used if no other rules apply. Choose the default action for traffic on this listener.

#### Routing action

Forward to target groups

Redirect to URL

Return fixed response

#### Forward to target group Info

Choose a target group and specify routing weight or [create target group](#).

##### Target group

AppTierTargetGroup

Target type: Instance, IPv4 | Target stickiness: Off

HTTP



##### Weight

1

##### Percent

100%

#### + Add target group

You can add up to 4 more target groups.

## 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-lb

Scheme: Internet-facing

IP address type: IPv4

#### Network mapping [Edit](#)

VPC: [vpc-030cb26268efc5870](#)

Public IPv4 IPAM pool: -

Availability Zones and subnets:

- us-east-1a  
[subnet-032203534593e6148e](#)  
Private-App-Subnet-AZ-1
- us-east-1b  
[subnet-01604a8aab303c194](#)  
Private-App-Subnet-AZ-2

#### Security groups [Edit](#)

internal-lb-sg

[sg-078580545f94196e](#)

#### Listeners and routing [Edit](#)

HTTP:80 | Forward to 1 target group

#### Service integrations [Edit](#)

Amazon CloudFront + AWS Web Application Firewall (WAF): -

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.

- 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.
- Name the Launch Template, and then under Application and OS Images include the app tier AMI you created.
- Under Instance Type select t2.micro. For Key pair and Network Settings don't include it in the template. We don't need a key pair to access our instances and we'll be setting the network information in the autoscaling group.
- Set the correct security group for our app tier, and then under Advanced details use the same IAM instance profile we have been using for our EC2 instances.

## 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 '&', '\*', '@'.

#### Template version description

Max 255 chars

#### Auto Scaling guidance | [Info](#)

Select this if you intend to use this template with EC2 Auto Scaling

Provide guidance to help me set up a template that I can use with EC2 Auto Scaling

#### ► Template tags

#### ► Source template

### ▼ Summary

#### Software Image (AMI)

App Tier

ami-0b5b575c7c2ba2404

#### Virtual server type (instance type)

t2.micro

#### Firewall (security group)

PrivateInstanceSG

#### Storage (volumes)

1 volume(s) - 8 GiB

**Free tier:** In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included

### Launch template contents

Specify the details of your launch template below. Leaving a field blank will result in the field not being included in the launch template.

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

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose [Browse more AMIs](#).

Recents    [My AMIs](#)    Quick Start

Don't include in launch template

Owned by me

Shared with me



Browse more AMIs  
Including AMIs from AWS, Marketplace and the Community

#### Amazon Machine Image (AMI)

AppTierImage  
ami-0b5b575c7c2ba2404  
2025-10-04T16:37:08.000Z Virtualization: hvm ENA enabled: true Root device type: ebs Boot mode: uefi-preferred

#### Description

App Tier

Architecture  
x86\_64

AMI ID  
ami-0b5b575c7c2ba2404

**Free tier:** In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included

### ▼ 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

Additional costs apply for AMIs with pre-installed software

All generations

[Compare instance types](#)

**Free tier:** In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included

### ▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

#### Key pair name

[Create new key pair](#)

**Free tier:** In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included

Cancel

[Create launch template](#)

**▼ Network settings** [Info](#)

**Subnet** | [Info](#)

Don't include in launch template

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

**Availability Zone** | [Info](#)

Don't include in launch template

[Create new subnet](#)

**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

[Compare security group rules](#)

**Security groups** | [Info](#)

Select security groups

PrivateInstanceSG sg-0878d95cb5c2a7ae7 VPC: vpc-030cb26268efc5870

**► Advanced network configuration**

**▼ Summary**

**Software Image (AMI)**  
App Tier  
ami-0b5b575c7c2ba2404

**Virtual server type (instance type)**  
t2.micro

**Firewall (security group)**  
PrivateInstanceSG

**Storage (volumes)**  
1 volume(s) - 8 GiB

**Free tier:** In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included

**▼ Advanced details** [Info](#)

**IAM instance profile** | [Info](#)

Myec2demo-role arn:aws:iam::076526621834:instance-profile/Myec2demo-role

[Create new IAM profile](#)

**Hostname type** | [Info](#)

Don't include in launch template

**DNS Hostname** | [Info](#)

Enable resource-based IPv4 (A record) DNS requests

Enable resource-based IPv6 (AAAA record) DNS requests

**Instance auto-recovery** | [Info](#)

Don't include in launch template

**Shutdown behavior** | [Info](#)

Don't include in launch template

**Stop - Hibernate behavior** | [Info](#)

Don't include in launch template

**Termination protection** | [Info](#)

Don't include in launch template

**▼ Summary**

**Software Image (AMI)**  
App Tier  
ami-0b5b575c7c2ba2404

**Virtual server type (instance type)**  
t2.micro

**Firewall (security group)**  
PrivateInstanceSG

**Storage (volumes)**  
1 volume(s) - 8 GiB

**Free tier:** In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included

[Cancel](#) [Create launch template](#)

- We will now create the Auto Scaling Group for our app instances. On the left side of the EC2 dashboard navigate to Auto Scaling Groups under Auto Scaling and click Create Auto Scaling group.
- Give your Auto Scaling group a name, and then select the Launch Template we just created and click next.
- On the Choose instance launch options page set your VPC, and the private instance subnets for the app tier and continue to step 3.
- For this next step, attach this Auto Scaling Group to the Load Balancer we just created by selecting the existing load balancer's target group from the dropdown. Then, click next.
- For Configure group size and scaling policies, set desired, minimum and maximum capacity to 2. Click skip to review and then Create Auto Scaling Group.
- You should now have your internal load balancer and autoscaling group configured correctly. You should see the autoscaling group spinning up 2 new app tier instances. If you wanted 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.

NOTE: Your original app tier instance is excluded from the ASG so you will see 3 instances in the EC2 dashboard. You can delete your original instance that you used to generate the app tier AMI but it's recommended to keep it around for troubleshooting purposes.

**Step 1**

Choose launch template or configuration

Step 2

Choose instance launch options

Step 3 - 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

**Choose launch template or configuration Info**

Specify a launch template that contains settings common to all EC2 instances that are launched by this Auto Scaling group. If you currently use launch configurations, you might consider migrating to launch templates.

**Name**

**Auto Scaling group name**  
Enter a name to identify the group.

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

**Launch template Info**

**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.

Switch to launch configuration 

[Create a launch template !\[\]\(6ddc11a8bdc1305d7079f78d83b61962\_img.jpg\)](#)

**Version**



[Create a launch template version !\[\]\(db137d2ec4030f45be45839359b8d40e\_img.jpg\)](#)

**Description**  
Launch template for app tier

**Launch template** [AppTierLaunchTemplate !\[\]\(ad7549d770e799b43f589a7b59b24cdf\_img.jpg\)](#)  
lt-07c36c540719a646f

**Instance type** t2.micro

**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 !\[\]\(afce20130bd45e3bfad2416121919c03\_img.jpg\)](#)  
lt-07c36c540719a646f

**Version** Default

**Description** Launch template for app tier

**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.

  
10.0.0.0/16 

[Create a VPC !\[\]\(88ca4906a7937afe1d8cbd5121f73d1e\_img.jpg\)](#)

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



**use1-az1 (us-east-1a)** | subnet-03220334593e6148e (Private-App-Subnet-AZ-1)  
10.0.3.0/24 

**use1-az2 (us-east-1b)** | subnet-01604a8aab303c194 (Private-App-Subnet-AZ-2)  
10.0.4.0/24 

[Create a subnet !\[\]\(59ab518646c8271a4befe39265ad48d6\_img.jpg\)](#)

**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.

- Choose launch template or configuration
- Step 2
- Choose instance launch options
- Step 3 - 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

## Integrate with other services - *optional* Info

Use a load balancer to distribute network traffic across multiple servers. Enable service-to-service communications with VPC Lattice. Shift resources away from impaired Availability Zones with zonal shift. You can also customize health check replacements and monitoring.

### 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.

#### Select Load balancing options

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 to attach

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

AppTierTargetGroup | HTTP

Application Load Balancer: app-tier-internal-lb



- Step 1
- Choose launch template or configuration
- Step 2
- Choose instance launch options
- Step 3 - 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

## 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.

2

### 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

2

Max desired capacity

2

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.

### Instance maintenance policy Info

Control your Auto Scaling group's availability during instance replacement events. This includes health checks, instance refreshes, maximum instance lifetime features and events that happen automatically to keep your group balanced, called rebalancing events.

#### Choose a replacement behavior depending on your availability requirements

Mixed behavior

No policy

For rebalancing events, new instances will launch before terminating others. For all other events, instances terminate and launch at the same time.

Prioritize availability

Launch before terminating

Launch new instances and wait for them to be ready before terminating others. This allows you to go above your desired capacity by a given percentage and may temporarily increase costs.

Control costs

Terminate and launch

Terminate and launch instances at the same time. This allows you to go below your desired capacity by a given percentage and may temporarily reduce availability.

Flexible

Custom behavior

Set custom values for the minimum and maximum amount of available capacity. This gives you greater flexibility in setting how far below and over your desired capacity EC2 Auto Scaling goes when replacing instances.

### Additional capacity settings

Capacity Reservation preference Info

Select whether you want Auto Scaling to launch instances into an existing Capacity Reservation or Capacity Reservation resource group.

Default

## Step-11: Web Tier Instance Deployment

- 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.

```
nginx.conf — Edited
listen      [::]:80;
server_name _;

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

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

#proxy for internal lb
location /api/{
    proxy_pass http://app-tier-internal-lb-52025663.us-east-1.elb.amazonaws.com:80/;
}

}

# Settings for a TLS enabled server.
#
#     server {
#         listen      443 ssl http2;
#         listen      [::]:443 ssl http2;
#         server_name _;
#         root       /usr/share/nginx/html;
#
#         ssl_certificate "/etc/pki/nginx/server.crt";
#         ssl_certificate_key "/etc/pki/nginx/private/server.key";
#         ssl_session_cache shared:SSL:1m;
#     }

```

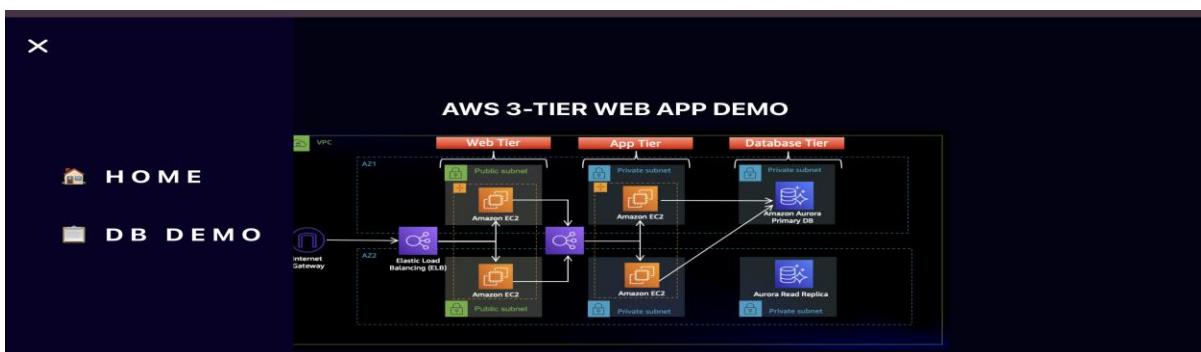
- Then, upload this file and the application-code/web-tier folder to the s3 bucket you created for this lab.

The screenshot shows the AWS S3 console with the path [Amazon S3](#) > [Buckets](#) > [mytestbucket-demo12](#). The bucket details page is displayed, showing 3 objects:

Name	Type	Last modified	Size	Storage class
<a href="#">app-tier/</a>	Folder	-	-	-
<a href="#">nginx.conf</a>	conf	October 4, 2025, 22:44:18 (UTC+05:30)	2.5 KB	Standard
<a href="#">web-tier/</a>	Folder	-	-	-

- Set the Name and Tag. Select Amazon Linux for the OS and Amazon Linux 2 AMI (HVM) - Kernel 5.10 for the AMI. For Instance Type, select t2.micro. (Select Proceed without a key pair.) For Network settings, click Edit.
- Select a VPC and a subnet. For Subnet, select Public-Web-Subnet. For Security Group, select WebTierSG, which you created in the previous step. For Auto-assign public IP, select Enable. click Advanced details.
- In the IAM instance profile field, select the IAM Role you created in the previous step. (In our example, we created the EC2andS3Role.) Finally, click Launch instance to create the instance.
- Follow the same steps you used to connect to the app instance and change the user to ec2-user. Test connectivity here via ping as well since this instance should have internet connectivity:

- sudo -su ec2-user
  - ping 8.8.8.8
- We now need to install all of the necessary components needed to run our front-end application. Again, start by installing NVM and node :
  - curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh | bash
  - source ~/.bashrc
  - nvm install 16
  - nvm use 16
- Now we need to download our web tier code from our s3 bucket:
  - Cd ~/
  - aws s3 cp s3://mytestbucket-demo12/web-tier/ web-tier –recursive
- Navigate to the web-layer folder and create the build folder for the react app so we can serve our code:
  - cd ~/web-tier
  - npm install
  - npm run build
- 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 dnf install nginx -y
- We will now have to configure NGINX. Navigate to the Nginx configuration file with the following commands and list the files in the directory:
  - cd /etc/nginx
  - ls
- You should see an nginx.conf file. We're going to delete this file and use the one we uploaded to s3. Replace the bucket name in the command below with the one you created for this workshop:
  - sudo rm nginx.conf
  - sudo aws s3 cp s3://BUCKET\_NAME/nginx.conf /etc/nginx/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.



## **Step-12: External Load Balancer and Auto Scaling**

- Navigate to Instances on the left hand side of the EC2 dashboard. Select the web tier instance we created and under Actions select Image and templates. Click Create Image.
- Give the image a name and description and then click Create image. This will take a few minutes, but if you want to monitor the status of image creation you can see it by clicking AMIs under Images on the left hand navigation panel of the EC2 dashboard.
- While the AMI is being created, we can go ahead and create our target group to use with the load balancer. On the EC2 dashboard navigate to Target Groups under Load Balancing on the left hand side. Click on Create Target Group.
- The purpose of forming this target group is to use with our load balancer so it may balance traffic across our public web tier instances. Select Instances as the target type and give it a name.
- Then, set the protocol to HTTP and the port to 80. Remember this is the port NGINX is listening on. Select the VPC we've been using thus far, and then change the health check path to be /health. Click Next.
- We are NOT going to register any targets for now, so just skip that step and create the target group.
- On the left hand side of the EC2 dashboard select Load Balancers under Load Balancing and click Create Load Balancer.
- We'll be using an Application Load Balancer for our HTTP traffic so click the create button for that option.
- After giving the load balancer a name, be sure to select internet facing since this one will not be public facing, but rather it will route traffic from our web tier to the app tier.
- Select the correct network configuration for VPC and public subnets.
- Select the security group we created for this internal ALB. Now, this ALB will be listening for HTTP traffic on port 80. It will be forwarding the traffic to our target group that we just created, so select it from the dropdown, and create the load balancer.
- 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.
- Name the Launch Template, and then under Application and OS Images include the app tier AMI you created.
- Under Instance Type select t2.micro. For Key pair and Network Settings don't include it in the template. We don't need a key pair to access our instances and we'll be setting the network information in the autoscaling group.
- Set the correct security group for our web tier, and then under Advanced details use the same IAM instance profile we have been using for our EC2 instances.

## 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

WebTierLaunchTemplate

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

Template version description

Launch template for web tier

Max 255 chars

Auto Scaling guidance | [Info](#)

Select this if you intend to use this template with EC2 Auto Scaling

Provide guidance to help me set up a template that I can use with EC2 Auto Scaling

► [Template tags](#)

► [Source template](#)

### Launch template contents

Specify the details of your launch template below. Leaving a field blank will result in the field not being included in the launch template.

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

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose [Browse more AMIs](#).

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

Recents | [My AMIs](#) | Quick Start

Don't include in launch template

Owned by me

Shared with me



Browse more AMIs  
Including AMIs from AWS, Marketplace and the Community

#### Amazon Machine Image (AMI)

WebTierImage

ami-08abd5b75a525f90c  
2025-10-04T18:24:38.000Z Virtualization: hvm ENA enabled: true Root device type: ebs Boot mode: uefi-preferred

#### Description

Image of our web tier instance

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

Advanced

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

Additional costs apply for AMIs with pre-installed software

All generations

[Compare instance types](#)

#### ▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

#### Key pair name

Don't include in launch template

[Create new key pair](#)

#### ▼ Network settings [Info](#)

##### Subnet

Don't include in launch template

[Create new subnet](#)

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

### ▼ Summary

#### Software Image (AMI)

Image of our web tier instance...[read more](#)  
ami-08abd5b75a525f90c

#### Virtual server type (instance type)

t2.micro

#### Firewall (security group)

Web-tier-sg

#### Storage (volumes)

1 volume(s) - 8 GiB

**Free tier:** In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included

### ▼ Summary

#### Software Image (AMI)

Image of our web tier instance...[read more](#)  
ami-08abd5b75a525f90c

#### Virtual server type (instance type)

t2.micro

#### Firewall (security group)

Web-tier-sg

#### Storage (volumes)

1 volume(s) - 8 GiB

**Free tier:** In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included

[Cancel](#)

[Create launch template](#)

### ▼ Summary

#### Software Image (AMI)

Image of our web tier instance...[read more](#)  
ami-08abd5b75a525f90c

#### Virtual server type (instance type)

t2.micro

#### Firewall (security group)

Web-tier-sg

#### Storage (volumes)

1 volume(s) - 8 GiB

**Free tier:** In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included

[Cancel](#)

[Create launch template](#)

The screenshot shows the configuration of a new AWS Lambda instance. It includes sections for Availability Zone, Firewall (security groups), Security groups, Storage (volumes), and a detailed description of the Free tier.

**Availability Zone**: Info  
Don't include in launch template ▾

**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 (radio button selected) Create security group

**Security groups**: Info  
Select security groups ▾  
Web-tier-sg sg-03047ab4462a46643 X  
VPC: vpc-030cb26268efc5870

**Storage (volumes)**: Info  
EBS Volumes Hide details

**Launch Template**: Info  
ami-0a61c7c8c/caca09e6c  
Virtual server type (instance type)  
t2.micro  
Firewall (security group)  
Web-tier-sg  
Storage (volumes)  
1 volume(s) - 8 GiB

**Free tier**: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included

Cancel Create launch template

- We will now create the Auto Scaling Group for our web instances. On the left side of the EC2 dashboard navigate to Auto Scaling Groups under Auto Scaling and click Create Auto Scaling group.
- Give your Auto Scaling group a name, and then select the Launch Template we just created and click next.
- On the Choose instance launch options page set your VPC, and the public subnets for the web tier and continue to step 3.
- For this next step, attach this Auto Scaling Group to the Load Balancer we just created by selecting the existing web tier load balancer's target group from the dropdown. Then, click next.
- For Configure group size and scaling policies, set desired, minimum and maximum capacity to 2. Click skip to review and then Create Auto Scaling Group.

**Instances (6) Info**

Last updated 1 minute ago	Connect	Instance state	Actions	Launch instances			
<input type="text"/> Find Instance by attribute or tag (case-sensitive)	All states ▾						
Instance state = running	X	Clear filters					
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv
i-0c7f0aeb4ce64965e	i-0c7f0aeb4ce64965e	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	-
i-02e34cccd821d214af	i-02e34cccd821d214af	Running	t2.micro	Initializing	View alarms +	us-east-1b	-
i-0cbdf10da0c0a8690	i-0cbdf10da0c0a8690	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	-
i-0b2880b39d26499a5	i-0b2880b39d26499a5	Running	t2.micro	Initializing	View alarms +	us-east-1a	-
AppLayer	i-09d439ac17f8499f9	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	-
AppLayer-web	i-027aac3b32a922c8d	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	-

**Launch Templates (2) Info**

Launch Template ID	Launch Template Name	Default Version	Latest Version	Create Time	Created By
lt-0c5d9a8aa0cf8a9e4	WebTierLaunchTemplate	1	1	2025-10-04T18:42:30.000Z	arn:aws:iam::076
lt-07c36c540719a646f	AppTierLaunchTemplate	1	1	2025-10-04T16:56:17.000Z	arn:aws:iam::076

**Security Groups (6) [Info](#)**

[Actions](#) | [Export security groups to CSV](#) | [Create security group](#)

VPC ID = vpc-030cb26268efc5870	X	Clear filters	< 1 >	⚙️				
Security group ID	▼	Security group name	▼	VPC ID	▼	Description	▼	Owner
<a href="#">sg-0878d95cb5c2a7ae7</a>		PrivateInstanceSG		<a href="#">vpc-030cb26268efc5870</a>		SG for our private app tier sg		0765:
<a href="#">sg-0f1b11cfafbd4f484</a>		DBSG		<a href="#">vpc-030cb26268efc5870</a>		SG for our databases		0765:
<a href="#">sg-0912fb5302f362b05</a>		internet-facing-lb-sg		<a href="#">vpc-030cb26268efc5870</a>		External load balancer security group		0765:
<a href="#">sg-01a680e902eb0db8e</a>		default		<a href="#">vpc-030cb26268efc5870</a>		default VPC security group		0765:
<a href="#">sg-03047ab4462a46643</a>		Web-tier-sg		<a href="#">vpc-030cb26268efc5870</a>		SG for the web tier		0765:
<a href="#">sg-0785805455f94196e</a>		internal-lb-sg		<a href="#">vpc-030cb26268efc5870</a>		SG for the internal load balancer		0765:

**Load balancers (2)**

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

[Actions](#) | [Create load balancer](#)

Name	State	Type	Scheme	IP address type	VPC ID	Availability Zones
<a href="#">web-tier-external-lb</a>	Active	application	Internet-facing	IPv4	<a href="#">vpc-030cb26268efc5870</a>	2 Availability Zone
<a href="#">app-tier-internal-lb</a>	Active	application	Internal	IPv4	<a href="#">vpc-030cb26268efc5870</a>	2 Availability Zone

**Target groups (2) [Info](#)**

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

Name	ARN	Port	Protocol	Target type	Load balancer	VPC ID
<a href="#">AppTierTargetGroup</a>	<a href="#">arn:aws:elasticloadbalancing:us-east-1:030cb26268efc5870:targetgroup/app-tier-internal-lb/54895455f94196e</a>	80	HTTP	Instance	app-tier-internal-lb	<a href="#">vpc-030cb26268efc5870</a>
<a href="#">WebTierTargetGroup</a>	<a href="#">arn:aws:elasticloadbalancing:us-east-1:030cb26268efc5870:targetgroup/web-tier-external-lb/0f1b11cfafbd4f484</a>	80	HTTP	Instance	web-tier-external-lb	<a href="#">vpc-030cb26268efc5870</a>

**Auto Scaling groups (2) [Info](#)**

Last updated less than a minute ago

[Launch configurations](#) | [Launch templates](#) | [Actions](#) | [Create Auto Scaling group](#)

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max
<a href="#">WebTierASG</a>	<a href="#">WebTierLaunchTemplate</a>   Version Default	2	-	2	2	2
<a href="#">AppTierASG</a>	<a href="#">AppTierLaunchTemplate</a>   Version Default	2	-	2	2	2

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 wanted 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.

NOTE: Again, your original web tier instance is excluded from the ASG so you will see 3 instances in the EC2 dashboard. You can delete your original instance that you used to generate the web tier AMI but it's recommended to keep it around for troubleshooting purposes.

Congrats! You've Implemented a 3 Tier Web Architecture!