

AWS Three-Tier Architecture Project

- AWS Three-tier architecture is used to separate the application into three distinct layers namely Web-tier, App-tier and Database-Tier.

Web-Tier:

This Layer is responsible for displaying the user-interface. It handles user input, displays information, and manages the user experience.

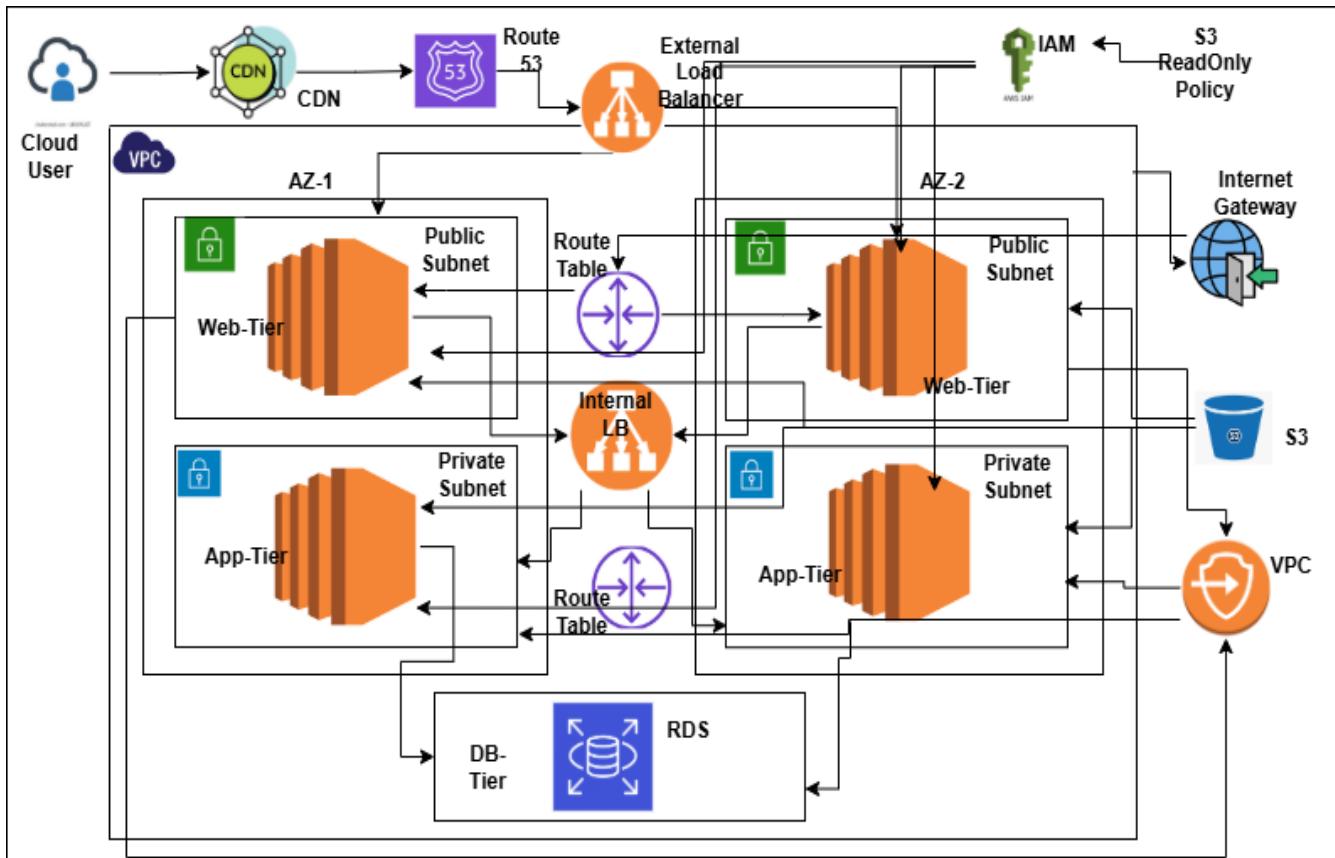
App-Tier:

This Layer contains the business logic of the application, it implements business rules, and interact with database tier to manage and retrieve data.

Database-Tier:

This Layer handles database operation, ensuring data integrity and security.

3-Tier Architecture Diagram



AWS Three-Tier Architecture Project	1
3-Tier Architecture Diagram.....	1
Step1: S3 bucket Creation	3
Step2: IAM Role Creation	4
Step3: VPC Creation.....	5
VPC Creation	5
Subnets:	6
Creating Internet Gateway	6
Creating NAT Gateway.....	6
Route Tables	7
Step4: Security Group Creation	9
Step5: Database-Tier Configuration	10
Step6: Procedure to Create the EC2 instance	11
Step7: App-Tier Configuration.....	12
Step8: Creating Internal Application Load Balancer for App-tier configuration	17
Step9: Creating Auto Scaling for Internal Application Load Balancer	22
Create AMI Template.....	22
Add AMI Template inside the Auto Scaling group.....	24
Step10: Web-Tier Configuration:.....	27
Step 11: Creating External Application Load Balancer for Web-tier configuration	30
Procedure to Create External ALB:.....	30
Step12: Creating Auto Scaling for External Application Load Balancer	33
Create AMI Template	34
Add AMI Template inside the Auto Scaling group.....	35
Step13: Testing Application Locally.....	38
Step14: Enabling CloudFront Distribution (CDN)	38
Creating Hosted Zone	40
Enabling SSL Certificate (ACM)	40
Adding Domain in CloudFront.....	41

Project Flow:

Step1: S3 bucket Creation

An Amazon S3 bucket is a fundamental storage resource within AWS S3. It functions as a container for storing objects, which are essentially files (like images, documents, or application data) along with any associated metadata.

- Download the code from [Github Repository](#) into your local machine.
- Create the Unique S3 Bucket.
- Node JS Application contains Web-tier and App-tier folder.
- Uploading the web-tier and app-tier folder inside the S3 Bucket.

The screenshot shows the 'Create bucket' wizard in the AWS S3 console. The 'General configuration' section is selected. The 'AWS Region' is set to 'US East (N. Virginia) us-east-1'. The 'Bucket type' section shows two options: 'General purpose' (selected) and 'Directory'. The 'Bucket name' field contains 's3projdemoapp'. Below it, there's a note about bucket naming rules and a 'Choose bucket' button for copying settings from another bucket. The URL format is shown as 'Format: s3://bucket/prefix'.

- By default, S3 bucket is Private, Bucket versioning is disabled.
- If we want we can enabling the bucket versioning to recover the previous files.

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

 Block all public access

Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

 Block public access to buckets and objects granted through new access control lists (ACLs)

S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.

 Block public access to buckets and objects granted through any access control lists (ACLs)

S3 will ignore all ACLs that grant public access to buckets and objects.

 Block public access to buckets and objects granted through new public bucket or access point policies

S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

 Block public and cross-account access to buckets and objects through any public bucket or access point policies

S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Bucket Versioning

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. [Learn more](#)

Bucket Versioning

- Disable
- Enable

s3projdemoapp [Info](#)

[Objects](#) [Metadata](#) [Properties](#) [Permissions](#) [Metrics](#) [Management](#) [Access Points](#)

Objects (2)

[C](#) [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions](#) [Create folder](#) [Upload](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Find objects by prefix

< 1 > [⚙️](#)

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	app-tier/	Folder	-	-	-
<input type="checkbox"/>	web-tier/	Folder	-	-	-

→ S3 Bucket created and uploaded the folders.

Step2: IAM Role Creation

IAM is an AWS security service that provides secure control over access to AWS resources. It enables administrators to manage users, groups, roles, and permissions, dictating who can be authenticated (signed in) and authorized (granted permissions) to use AWS services and resources.

Create the **IAM role for AmazonS3ReadOnlyAccess policy** to access the S3 bucket from EC2 instance.

AmazonS3ReadOnlyAccess

Provides read only access to all buckets via the AWS Management Console.

```
1  [{}  
2      "Version": "2012-10-17",  
3      "Statement": [  
4          {  
5              "Effect": "Allow",  
6              "Action": [  
7                  "s3:Get*",  
8                  "s3>List*",  
9                  "s3:Describe*",  
10                 "s3-object-lambda:Get*",  
11                 "s3-object-lambda>List*"  
12             ],  
13             "Resource": "*"  
14         }  
15     ]  
16 ]
```

Step3: VPC Creation

VPC is a logically isolated section of the AWS cloud that allows you to launch AWS resources in a virtual network that you define.

VPC Creation

Creating the **VPC** to customize the network configuration within AWS.

vpc-0fc171b4597b75040 / Project-vpc				Actions ▾
Details		Info		
VPC ID	vpc-0fc171b4597b75040	State	Available	Block Public Access
DNS resolution	Enabled	Tenancy	default	Off
Main network ACL	acl-06c5389df1083dc72	Default VPC	No	DNS hostnames
IPv6 CIDR (Network border group)	-	Network Address Usage metrics	Disabled	Disabled
				Main route table
				rtb-06ca0ad79c87088f4
				IPv6 pool
				-
				Owner ID
				054728709811

- Creating the **public** and **private Subnets** with multiple Available Zone within the VPC.

Subnets:

Subnets (9) Info						Last updated 1 minute ago	Actions	Create subnet
	Name	Subnet ID	State	VPC	Block Public...		IPv4 CIDR	
<input type="checkbox"/>	public-sub-01	subnet-02a0a295fe1eddb92	Available	vpc-0fc171b4597b75040 Proj...	<input type="checkbox"/> Off		10.0.1.0/24	
<input type="checkbox"/>	private-sub-01	subnet-0ffed5b597bb76d4b	Available	vpc-0fc171b4597b75040 Proj...	<input type="checkbox"/> Off		10.0.2.0/24	
<input type="checkbox"/>	DB-sub-01	subnet-0762cc0dbd31ecff	Available	vpc-0fc171b4597b75040 Proj...	<input type="checkbox"/> Off		10.0.3.0/24	
<input type="checkbox"/>	-	subnet-0e636ec522eba67c3	Available	vpc-0305879d26328841e vpc...	<input type="checkbox"/> Off		172.31.16.0	

Creating Internet Gateway

- An AWS Internet Gateway (IGW) is a horizontally scaled, redundant, and highly available VPC component that provides a connection between your Virtual Private Cloud (VPC) and the internet.
- Internet gateway supports both inbound and outbound rules.

→ Creating Internet gateway and attach with VPC.

igw-0d9d748b9b84ce0b8 / project-igw

Details Info		Actions	
Internet gateway ID igw-0d9d748b9b84ce0b8	State Attached	VPC ID vpc-0fc171b4597b75040 Project-vpc	Owner 054728709811
Tags			
Manage tags			
Search tags			
Key	Value	Manage tags	
Name	project-igw	Search tags	

Creating NAT Gateway

- NAT Gateway in AWS is a managed Network Address Translation (NAT) service that allows instances in a private subnet to access the internet or other VPCs without being directly exposed to the public internet.
- It acts as a bridge, enabling outbound traffic from private instances while preventing unsolicited inbound connections.

→ Creating NAT gateway and attach with VPC.

- Make the Connectivity type as Public to allows instances within a private subnet to access the internet.

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.

The name can be up to 256 characters long.

Subnet
Select a subnet in which to create the NAT gateway.

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.

- Allocate Elastic IP Address, allow to enable instances in a private subnet to connect to the internet while preventing inbound connections from the internet. The EIP acts as a static, public IPv4 address that is associated with the NAT Gateway, allowing it to initiate outbound connections and translate private IP addresses to public ones.

Route Tables

- Create public and private **Route tables** and associate subnets respectively and attach **Internet Gateway** in public route table and attach the **NAT gateway** in public subnet as well as private route table to provide internet access for private subnets. NAT gateway is used to allow only outbound rules.

Public Route Tables:

- Edit Routes – Adding Internet Gateway in public route table.

rtb-0c95ed0235e8b8d14 / public-routetbl

Actions ▾

Details [Info](#)

Route table ID	Main	Explicit subnet associations	Edge associations
rtb-0c95ed0235e8b8d14	<input type="checkbox"/> No	subnet-02a0a295fe1eddb92 / public-sub-01	-
VPC	Owner ID		
vpc-0fc171b4597b75040 Project-vpc	054728709811		

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

Routes (2)

Filter routes					
Destination	Target	Status	Propagated	Route Origin	
0.0.0.0/0	igw-0d9d748b9b84ce0b8	Active	No	Create Route	Edit routes
10.0.0.0/16	local	Active	No	Create Route Table	Edit routes

→ Subnet associations – Adding Public subnets in public route table.

rtb-0c95ed0235e8b8d14 / public-routetbl

Actions ▾

Details [Info](#)

Route table ID	Main	Explicit subnet associations	Edge associations
rtb-0c95ed0235e8b8d14	<input type="checkbox"/> No	subnet-02a0a295fe1eddb92 / public-sub-01	-
VPC	Owner ID		
vpc-0fc171b4597b75040 Project-vpc	054728709811		

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

Explicit subnet associations (1)

Find subnet association					
Name	Subnet ID	IPv4 CIDR	IPv6 CIDR		
public-sub-01	subnet-02a0a295fe1eddb92	10.0.1.0/24	-	Edit subnet associations	Edit subnet associations

Private Route Tables:

→ Edit Routes – Adding NAT Gateway in private route table.

rtb-059ce683c6c607cd4 / private-routetbl

Details [Info](#)

Route table ID rtb-059ce683c6c607cd4	Main <input checked="" type="checkbox"/> No	Explicit subnet associations 2 subnets	Edge associations -
VPC vpc-0fc171b4597b75040 Project-vpc	Owner ID 054728709811		

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

Routes (2)

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	nat-0d7022c54f1de13b8	Active	No	Create Route
10.0.0.0/16	local	Active	No	Create Route Table

→ Subnet associations – Adding Private subnets in private route table.

rtb-059ce683c6c607cd4 / private-routetbl

Details [Info](#)

Route table ID rtb-059ce683c6c607cd4	Main <input checked="" type="checkbox"/> No	Explicit subnet associations 2 subnets	Edge associations -
VPC vpc-0fc171b4597b75040 Project-vpc	Owner ID 054728709811		

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

Explicit subnet associations (2)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
DB-sub-01	subnet-0762cc0dbd31eccff	10.0.3.0/24	-
private-sub-01	subnet-0fed5b597bb76d4b	10.0.2.0/24	-

Step4: Security Group Creation

Creating **Security group**, it acts as a firewall that controls the inbound and outbound traffics.

We need to create six Security group each for different purposes.

External Load Balancer: Allow Http (port 80) to anyone.

Jump Server: Allow SSH (port 22) to anyone.

Web Server: Http (port 80) to External Load Balancer,

Allow SSH to Jump Server.

Internal Load Balancer: Allow Http (port 80) to Web Server.

App Server: Allow TCP port 4000 to Internal Load Balancer,

SSH (port 22) to Jump Server.

Database Server: Allow MySQL/aurora port 3306 to App Server.

Security group ID	Security group name	VPC ID	Description
sg-0d9dcf19241100275	Internal-LB	vpc-0fc171b4597b75040	Internal-LB
sg-0832c71b2686856a0	Webserver	vpc-0fc171b4597b75040	Webserver
sg-010137275c5aa8165	Jump-Server	vpc-0fc171b4597b75040	Jump-Server
sg-08ddfc80763defd	App-Server	vpc-0fc171b4597b75040	App-Server
sg-0fd3eca3eade3706	DB-secgrp	vpc-0fc171b4597b75040	DB-secgrp
sg-0c18959cb40d2fe74	Public-LB	vpc-0fc171b4597b75040	Public-LB

Step5: Database-Tier Configuration

- Goto RDS → Create DB instances → Create Database
- Select Engine Option: MySQL → Edition: MySQL Community
- Deployment Option: Select Availability Zone type.
- Settings → Credentials management : Self-Managed, set username and password
- Select custom VPC you have created
- Select Security group → choose existing : Database security group
- Select Availability Zone
- Additional settings like Log Reports (optional)

Aurora and RDS > Databases > database-1

The screenshot shows the AWS RDS 'Databases' section for a database named 'database-1'. The 'Connectivity & security' tab is active. Key details include:

- Endpoint:** database-1.ckjqk0osi3pp.us-east-1.rds.amazonaws.com
- Port:** 3306
- Networking:**
 - Availability Zone:** us-east-1c
 - VPC:** Project-vpc (vpc-0fc171b4597b75040)
 - Subnet group:** project-rds
 - Subnets:** subnet-0ffed5b597bb76d4b, subnet-0762cc0dbd31eccff
 - Network type:** IPv4
- Security:**
 - VPC security groups:** DB-secgrp (sg-0fd3eca3eade3706) (Active)
 - Publicly accessible:** No
 - Certificate authority:** rds-ca-rsa2048-g1
 - Certificate authority date:** May 26, 2061, 05:04 (UTC+05:30)
 - DB instance certificate expiration date:** August 11, 2026, 17:56 (UTC+05:30)

Database created successfully.

Step6: Procedure to Create the EC2 instance

EC2 (Elastic Compute Cloud) instance is a virtual server in the cloud, providing resizable compute capacity that allows user to run applications.

→ Launch Instance → Name → AMI (Application Machine Image) browse AMI → Provides pre-configured template to launch Ec2 instance.

→ Select Instance Type : t2.micro (which is free)

→ Create Key pair → Key pair name, key pair type - RSA (which is encrypted private and public key pair) → Select Private key file format → .pem Private Key file format types:

- .pem Private Enhanced Mail (for openssh)
- .ppk PuTTy Private key (for putty)

→ Network Settings: choose VPC, Subnet, Available zone, Auto-assign public ip, Security group, Configure Inbound security rules.

→ Advanced Networking configurations : We can assign Primary Ip , Secondary Ip.

→ User data (optional): we can run bash scripting which helps to run the script during Ec2 server creation.

- Create Ec2 instance for Jump Server → select VPC, select **Public** subnet with multiple AZ, auto assign public Ip: **enable**, Select Security group: Jump server security group, set primary Ip and secondary Ip (optional) Eg. 10.0.1.5.
- Create Ec2 instance for Web Server → select VPC, select **Public** subnet with multiple AZ, auto assign public Ip: **enable**, Select Security group: Web server security group, set primary Ip and secondary Ip (optional) Eg. 10.0.1.15.
- Create Ec2 instance for App Server → select VPC, select **Private** subnet with multiple AZ, auto assign public Ip: **disable**, Select Security group: App server security group, set primary Ip and secondary Ip (optional) Eg. 10.0.2.5.
- Select Ec2 instance → Goto Action → Security → Modify IAM and Attach the IAM role for Jump Server, Web Server, App Server.
- Jump Server: It is responsible for safely bypassing firewalls that isolate the public and private networks. It acts as a bridge between different security zones. Every cloud user should communicate with Application only through the Jump server.

Instances (3) Info							Last updated less than a minute ago	Connect	Instance state ▾	Actions ▾	Launch instances	▼
	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4				
<input type="checkbox"/>	WebServer	i-03f3242e5987c9e71	Running	t2.micro	2/2 checks passed View alarms	2/2 checks passed View alarms	us-east-1a	—				
<input type="checkbox"/>	JumpServer	i-036a970ab16f30120	Running	t2.micro	2/2 checks passed View alarms	2/2 checks passed View alarms	us-east-1a	—				
<input type="checkbox"/>	AppServer	i-0c61895b78a18b76e	Running	t2.micro	2/2 checks passed View alarms	2/2 checks passed View alarms	us-east-1b	—				

Step7: App-Tier Configuration

7a. Goto app server

→ Open jump server → Create the pem key and provide read permission
 sudo chmod 400 key.pem

→ Goto app server through SSH credentials

ssh -i <keypair> <username>@ip i.e., ssh -i key.pem ec2-user@10.0.2.5

7b. Install MySQL Client and Connect

Commands to Install MySQL Client:

```
yum list mariadb*
sudo yum install mariadb105-server.x86_64
sudo systemctl enable mariadb
sudo systemctl start mariadb
sudo systemctl status mariadb.service
sudo mysql --version
```

Master Username: admin

password: Admin123Pwd

endpoint (hostname): database-1.ckjqk0osi3pp.us-east-1.rds.amazonaws.com

Command to Login MySQL DB Engine:

mysql -h <Host_Name> -P 3306 -u <User_Name> -p

mysql -h database-1.ckjqk0osi3pp.us-east-1.rds.amazonaws.com -P 3306 -u admin -p

Enter the password:

```
[root@ip-10-0-2-6 ec2-user]# mysql -h database-1.ckjgk0osi3pp.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 36
Server version: 8.0.39 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

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

MySQL [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| webappdb           |
+--------------------+
5 rows in set (0.002 sec)

MySQL [(none)]> use webappdb;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
Database changed
```

→ Your database is now connected...If not connected, then check your credentials and Security group.

7c. Create Database, table and Insert the record in db.

SQL commands:

Create the database	create database <database_name>; Create database webappdb;
Show database list	Show database;
Create table in database	Use webappdb; CREATE TABLE IF NOT EXISTS transactions (id INT NOT NULL AUTO_INCREMENT, amount DECIMAL(10,2), description VARCHAR(100), PRIMARY KEY (id));
Show table list	SHOW TABLES;
Insert record	INSERT INTO transactions (amount, description) VALUES ('400', 'groceries');
Select record	SELECT * FROM transactions;

```

MySQL [(none)]> create database webappdb;
Query OK, 1 row affected (0.021 sec)

MySQL [(none)]> use webappdb;
Database changed
MySQL [webappdb]> create table if not exist transactions (
    -> id int not null auto_increment,
    -> amount decimal(10,2),
    -> description varchar(100),
    -> primary key (id)
    ->
    -> )
    ->
MySQL 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'exist transactions (
    id int not null auto_increment,
    amount decimal(10,2),
    descr' at line 1
MySQL [webappdb]> 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 (0.054 sec)

MySQL [webappdb]> show tables
-> ;
+-----+
| Tables_in_webappdb |
+-----+
| transactions      |
+-----+
1 row in set (0.002 sec)

MySQL [webappdb]> insert into transactions (amount,description) values (1000,'diarys');
Query OK, 1 row affected (0.011 sec)

MySQL [webappdb]> select * from transactions
-> ;
+-----+-----+
| id | amount | description |
+-----+-----+
| 1  | 1000.00 | diarys   |
+-----+-----+
1 row in set (0.001 sec)

```

```

| Tables_in_webappdb |
+-----+
| transactions      |
+-----+
1 row in set (0.002 sec)

MySQL [webappdb]> select * from transactions;
+-----+-----+
| id | amount | description |
+-----+-----+
| 18 | 100.00 | fruits   |
| 19 | 200.00 | Almond  |
| 20 | 7868.00 | uynyi   |
+-----+-----+
3 rows in set (0.001 sec)

MySQL [webappdb]> insert into transactions (amount,description) values (500,'chee');
Query OK, 1 row affected (0.005 sec)

MySQL [webappdb]> select * from transactions;
+-----+-----+
| id | amount | description |
+-----+-----+
| 18 | 100.00 | fruits   |
| 19 | 200.00 | Almond  |
| 20 | 7868.00 | uynyi   |
| 21 | 500.00 | Ghee    |
+-----+-----+

```

→ after inserting the records, then Type Exit

7d. Copy the S3 bucket App-tier folder into App server

In App server, make directory for App-tier folder content,

`sudo mkdir app-tier`

`cd app-tier`

Command to copy s3 bucket folder into app server:

`aws s3 cp s3://s3projdemoapp/app-tier/ . --recursive`

`aws s3 ls` → to list all content inside s3 bucket

Update the hostname, username, password, database in DbConfig.js file.

Open → app-tier → DbConfig.js

```
module.exports = Object.freeze({  
  DB_HOST : 'database-1.ckjqk0osi3pp.us-east-1.rds.amazonaws.com',  
  DB_USER : 'admin',  
  DB_PWD : 'Admin123Pwd',  
  DB_DATABASE : 'webappdb'  
});
```

```
module.exports = Object.freeze({  
  DB_HOST : 'database-1.ckjqk0osi3pp.us-east-1.rds.amazonaws.com',  
  DB_USER : 'admin',  
  DB_PWD : 'Admin123Pwd',  
  DB_DATABASE : 'webappdb'  
});  
~
```

7e: Installing NVM (node version manager).

Run the command:

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

7f: Next, install a compatible version of Node.js and make sure it's being used

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.

Run the below commands to install Pm2 process manager

```
nvm install 16  
nvm use 16  
npm install -g pm2  
npm install  
pm2 start index.js
```

```

[npm] npm start for details.
[root@ip-10-0-2-5 app-tier]# pm2 start index.js
=====
```
 /
 / \ _
 \ / _
 \ _
 _
```
Runtime Edition

PM2 is a Production Process Manager for Node.js applications
with a built-in Load Balancer.

Start and Daemonize any application:
$ pm2 start app.js

Load Balance 4 instances of api.js:
$ pm2 start api.js -i 4

Monitor in production:
$ pm2 monitor

Make pm2 auto-boot at server restart:
$ pm2 startup

To go further checkout:
http://pm2.io/
```
[root@ip-10-0-2-5 app-tier]# pm2 start index.js
[PM2] Spawning PM2 daemon with pm2_home=/root/.pm2
[PM2] PM2 Successfully daemonized
[PM2] Starting /home/ec2-user/app-tier/index.js in fork_mode (1 instance)
[PM2] Done.

id	name	namespace	version	mode	pid	uptime	↴	status	cpu	mem	user	watching
 0 | index | default | 1.0.0 | fork | 31080 | 0s | 0 | online | 0% | 25.7mb | root | disabled
```
[root@ip-10-0-2-5 app-tier]#

```

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

pm2 list

→ you will find status as online; the app is running. If you find any error occurred, then you need to do some troubleshooting. To look at the latest errors, use this command:

pm2 logs

pm2 startup

After running this pm2 startup command, you will get the following script to run the command to save the process done till now, if you don't run the script given below, then node process will not be saved.

The command to save the process is starts with ...

Sudo env PATH=\$PATH:/home/ec2-user/.nvm/versions/node/v16.20.....

Hint: after running pm2 startup, then you will get the below instructions,

To setup the Startup Script, copy/paste the following command:

Sudo env PATH=\$PATH:/home/ec2-user/.nvm/versions/node/v16.20.....

pm2 save

Testing the App-Tier configurations:

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, curl <http://localhost:4000/health>

To test App to connected with database, curl <http://localhost:4000/transaction>

```
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

Last login: Sun Aug 17 12:43:41 2025 from 18.206.107.27
[ec2-user@ip-10-0-1-5 ~]$ ssh -i mykey.pem ec2-user@10.0.2.6
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

Last login: Sun Aug 17 12:43:59 2025 from 10.0.1.5
[ec2-user@ip-10-0-2-6 ~]$ curl http://localhost:4000/health
"This is the health check"
[ec2-user@ip-10-0-2-6 ~]$ curl http://localhost:4000/transaction
{"result":[{"id":18,"amount":100,"description":"fruits"}, {"id":19,"amount":200,"description":"Almond"}, {"id":20,"amount":7868,"description":"uynyi"}]}[ec2-user@ip-10-0-2-6 ~]$ 
```



Before Creating AMI, make sure everything is working fine even after the server is restarted.

Now, everything is fine, then take AMI for Auto Scaling process for Internal Load Balancer

Step8: Creating Internal Application Load Balancer for App-tier configuration

- Elastic Load Balancer (ELB) service, is a mechanism that automatically distributes incoming application traffic across multiple targets, such as Amazon EC2 instances, within one or more Availability Zones.
- Application Load Balancer works based on Path based routing policy, it operates in L7 of OSI Model. It handles http, https traffics.

Create Internal Load Balancer without creating instance inside the target group.

Procedure to Create Internal ALB:

→ Create the Target group

Target group

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
 - Offers the flexibility for a Network Load Balancer to accept and route TCP requests within a specific VPC.
 - Facilitates using static IP addresses and PrivateLink with an Application Load Balancer.

Target group name

→ Select port number 4000 which is configured in backend Application and select the VPC we created.

Target group

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

Port
Port for load balancer-to-target communication. Can't be modified after creation. Individual targets during registration.

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.

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

→ Health Check Path is '**health**', which is configured in backend process.

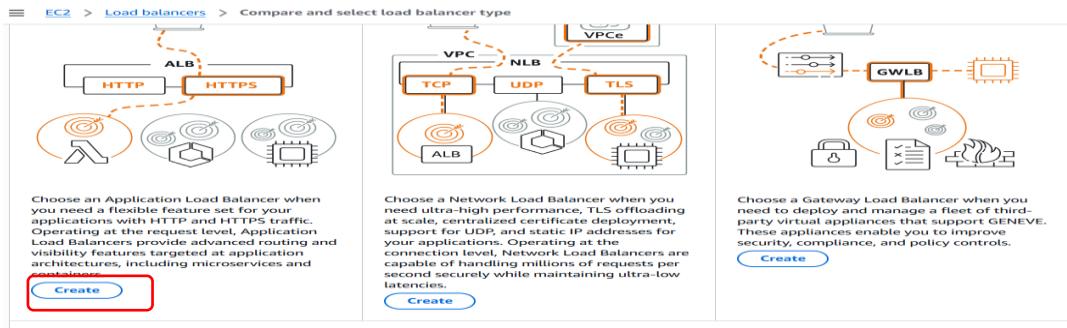
The screenshot shows the 'Health checks' configuration section of a CloudFormation stack. It includes fields for 'Health check protocol' (set to 'HTTP'), 'Health check path' (containing '/health'), 'Advanced health check settings' (with a 'Restore defaults' button), 'Health check port' (set to 'Traffic port'), and 'Healthy threshold' (set to 2).

→ Creating LB without registering the EC2 instance inside the target group.

The screenshot shows the 'Register targets' step of a CloudFormation stack. It displays a table of 'Available instances' with three entries: 'WebServer', 'JumpServer', and 'AppServer'. A red box highlights the checkbox column for selecting instances. Below the table, a red box highlights the 'Ports for the selected instances' field, which contains the value '4000'.

Instance ID	Name	State	Security groups	Zone
i-03f3242e5987c9e71	WebServer	Running	Webserver	us-east-1a
i-036a970ab16f30120	JumpServer	Running	Jump-Server	us-east-1a
i-0c61895b78a18b76e	AppServer	Running	App-Server	us-east-1b

→ Create Application Load Balancer and add the Target group inside the ALB.



EC2 > Load balancers > Create Application Load Balancer

▶ 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.

InternalLB

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.

Dualstack

Includes IPv4 and IPv6 addresses.

Dualstack without public IPv4

Includes a public IPv6 address, and private IPv4 and IPv6 addresses. Compatible with **internet-facing** load balancers only.

→ Select VPC and Private subnets for multiple Available Zones.

EC2 > Load balancers > Create Application Load Balancer

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.

vpc-0fc171b4597b75040 (Project-vpc)
10.0.0.0/16

Create VPC

IP pools - new | Info

You can optionally choose to configure an IPAM pool as the preferred source for your load balancer's IP addresses. Create or view Pools in 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-az6)

us-east-1b (use1-az1)

Subnet

Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses are required for your load balancer to scale efficiently.

subnet-0ffed5b597bb76d4b
IPv4 subnet CIDR: 10.0.2.0/24

private-sub-01

us-east-1c (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.

subnet-0762cc0dbd31ecff
IPv4 subnet CIDR: 10.0.3.0/24

DB-sub-01

- Select the App Server Security group
- Select the Internal Target group inside the Internal Load Balancer.

EC2 > Load balancers > Create Application Load Balancer

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

App-Server
sg-08ddfc80763deffd VPC: vpc-0fc171b4597b75040

Listeners and routing | Info

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

▼ Listener HTTP:80

Protocol: HTTP Port: 80

Default action | Info

Forward to: InternalTG1 Target type: Instance, IPv4

Create target group

Listener tags - optional

Consider adding tags to your listener. Tags enable you to categorize your AWS resources so you can more easily manage them.

Load Balancer is created successfully.

Step9: Creating Auto Scaling for Internal Application Load Balancer

AWS Auto Scaling allows you to automatically adjust the number of EC2 instances capacity based on the real-time demand.

- Generate Auto Scaling for Internal ALB and Select the (Private) App-tier configuration Ec2 instance as the Template.

Procedure to Create Auto Scaling:

Create AMI Template.

Amazon Machine Image (AMI) is a pre-configured template for launching EC2 instance.

The screenshot shows the AWS EC2 Instances page. There are three instances listed: WebServer (i-03f3242e5987c9e71), JumpServer (i-036a970ab16f30120), and AppServer (i-0c61895b78a18b76e). The AppServer instance is selected. A context menu is open over the AppServer row, with the 'Create image' option highlighted. Below, the 'Create image' wizard is shown, with the 'Image name' field set to 'Appami'. Other fields include 'Image description - optional' (empty) and a checkbox for 'Reboot instance' (unchecked).

Instances (1/3) [Info](#)

Last updated 1 minute ago

Actions ▲ [Launch instances](#) ▾

Instance diagnostics [View details](#)

Instance settings [Edit](#) [View details](#)

Networking [View details](#)

Security [View details](#)

Image and templates [Edit](#) [View details](#)

Monitor and troubleshoot [View details](#)

Find Instance by attribute or tag (case-sensitive)

Running ▾

Name	Instance ID	Instance state	Instance type	Status check	Alarm status
WebServer	i-03f3242e5987c9e71	Running View details	t2.micro	2/2 checks passed View alarms	
JumpServer	i-036a970ab16f30120	Running View details	t2.micro	2/2 checks passed View alarms	
<input checked="" type="checkbox"/> AppServer	i-0c61895b78a18b76e	Running View details	t2.micro	2/2 checks passed View alarms	

EC2 > Instances > i-0c61895b78a18b76e > Create image

Image details

Instance ID [i-0c61895b78a18b76e](#) (AppServer)

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

Image description - optional Maximum 255 characters.

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

Instance volumes

Storage type	Device	Snapshot	Size	Volume type	IOPS	Throughput	Delete on termination	Encrypted
EBS	/dev...	Create new snapshot fro...	8	EBS General Purpose SS...	5000		<input checked="" type="checkbox"/> Enable	<input type="checkbox"/> Enable

[Add volume](#)

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

AppTemplate

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

Template version description

A prod webserver for MyApp

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

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)

Appami

ami-05b5cade8e7275e13 Virtualization: hvm ENA enabled: true Root device type: ebs Boot mode: uefi-preferred

Description

EC2 > Launch templates > Create launch template

Key pair name
Don't include in launch template

Network settings Info

Subnet Info
Don't include in launch template

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 Create security group

Common security groups Info

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

Advanced network configuration

Summary

Software Image (AMI)
Appami
ami-05b5cade8e7275e13

Virtual server type (instance type)
t2.micro

Firewall (security group)

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

Add AMI Template inside the Auto Scaling group.

- Choose the App Server AMI Template inside the Auto Scaling group.
- EC2 instance will be automatically assigned inside the target group.

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1

Choose launch template

Step 2

Step 3 - optional

Step 4 - optional

Step 5 - optional

Step 6 - optional

Step 7

Choose launch template Info

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

Name

Auto Scaling group name
Enter a name to identify the group Must be unique to this account in the current Region and no more than 255 characters.

Launch template Info

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

Launch template

Apptemplate

Version

- Select VPC and Private subnets for multiple Available Zones.
- Attach to an Existing Load Balancer.
- Select the Internal Load Balancer.

o Scaling group

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

vpc-0fc171b4597b75040 (Project-vpc)
10.0.0.0/16

Create a VPC

Availability Zones and subnets

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

Select Availability Zones and subnets

use1-az1 (us-east-1b) | subnet-0ffed5b597bb76d4b (private-sub-01) 10.0.2.0/24
use1-az2 (us-east-1c) | subnet-0762cc0dbd31eccff (DB-sub-01) 10.0.3.0/24

Create a subnet

Availability Zone distribution - new

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

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

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

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1
Choose launch template
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

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

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

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

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

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

Attach to an existing load balancer

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

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

Choose from Classic Load Balancers

Existing load balancer target groups

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

Select target groups

InternalTG1 | HTTP
Application Load Balancer: InternalLB1

Fixing the desired capacity based on the real-time demand.

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1
Step 2
Step 3 - optional
Step 4 - optional
Configure group size and scaling
Step 5 - optional
Step 6 - optional
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.

1

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 1

Max desired capacity 2

Equal or less than desired capacity

Equal or greater than desired capacity

Here I'm fixing the Target Value as 1 for testing purpose.

Scaling group

Min desired capacity 1

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 automatically 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.

Scaling policy name Target Tracking Policy

Metric type Info

Monitored metric that determines if resource utilization is too low or high. If using EC2 metrics, consider enabling detailed monitoring for better scaling performance.

Average CPU utilization

Target value 1

Instance warmup Info

300 seconds

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1
Step 2
Step 3 - optional
Step 4 - optional
Step 5 - optional
Step 6 - optional
Add tags
Step 7
Review

Add tags - optional Info

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

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

Tags (1)

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

Add tag

49 remaining

Cancel Previous Next

Auto Scaling for External Load Balancer is created successfully.

Step10: Web-Tier Configuration:

10a. Goto web server

- Open jump server
- Goto app server through SSH credentials

```
ssh -i <keypair> <username>@ip i.e., ssh -i key.pem ec2-user@10.0.1.15
```

10b. Copy the S3 bucket web-tier folder into web server

In web server, make directory for web-tier folder content,
sudo mkdir web-tier
cd web-tier

Command to copy s3 bucket folder into web server:

```
aws s3 cp s3://s3projdemoapp/web-tier/ . --recursive  
aws s3 ls → to list all content inside s3 bucket
```

10c. Installing NVM (Node Version Manager)

NVM is a command-line tool that facilitates the installation, management, and switching between multiple versions of Node.js on a single system.

Run the following command:

```
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
```

Install NPM (Node Package Manager)

NPM is the key tool in JavaScript development, allowing developers to install, share, and manage dependencies in their projects.

```
npm install → it will create the node_module folder
```

npm run build → it will create the build folder, which contains index.html file

10d. Connecting Frontend with backend application by Nginx server

Nginx is the powerful web server that can do reverse proxy, load balancer, mail proxy, and http cache.

Add Internal Load balancer URL in nginx.config file by installing nginx

sudo yum install nginx -y

- You will have nginx.conf file in application-code folder, which will be provided by the developer.
- Copy the internal Load Balancer url and paste inside the nginx file. (in proxy for internal load balancer).
- And copy that nginx.config file content, replace it with the /etc/nginx/nginx.conf file.

cd /etc/nginx

ls → you will find nginx.conf

sudo vi nginx.conf

```

      #_
~\_ #####_      Amazon Linux 2023
~~ \###|
~~   \#/ ____ https://aws.amazon.com/linux/amazon-linux-2023
~~     V~' .->
~~_
~~.-
~~/_/ _/
~/m/'

Last login: Mon Aug 18 11:25:02 2025 from 18.206.107.29
[ec2-user@ip-10-0-1-5 ~]$ ssh -i mykey.pem ec2-user@10.0.1.15
      #
~\_ #####_      Amazon Linux 2023
~~ \###|
~~   \#/ ____ https://aws.amazon.com/linux/amazon-linux-2023
~~     V~' .->
~~_
~~.-
~~/_/ _/
~/m'

Last login: Mon Aug 18 11:25:29 2025 from 10.0.1.5
[ec2-user@ip-10-0-1-15 ~]$ cd web-tier
[ec2-user@ip-10-0-1-15 web-tier]$ ls
README.md  build  node_modules  package-lock.json  package.json  public  src
[ec2-user@ip-10-0-1-15 web-tier]$ 

```

```

[ec2-user@ip-10-0-1-15 ~]$ cd /etc/nginx
[ec2-user@ip-10-0-1-15 nginx]$ ls
conf.d  fastcgi.conf  fastcgi_params  koi-utf  mime.types  nginx.conf  scgi_params  uwsgi_params  win-utf
default.conf.default  fastcgi_params.default  koi-win  mime.types.default  nginx.conf.default  scgi_params.default  uwsgi_params.default
[ec2-user@ip-10-0-1-15 nginx]$ sudo vi nginx.conf
[ec2-user@ip-10-0-1-15 nginx]$ 

```

```

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

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

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

    #proxy for internal lb
    location /api/ {
        proxy_pass http://InternalLB-160122618.us-east-1.elb.amazonaws.com:80/; ←
    }

}
-- INSERT --

```

providing permission for home directory

sudo chmod -R 755 /home/ec2-user → providing permission for nginx to access the files
 sudo service nginx restart

`sudo chkconfig nginx on` → this command is used to start the service even after the system is rebooted. Make sure that both frontend and backend database is inserting the records correctly

`curl http://localhost` → Checking Web server Locally,

```
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

Last login: Mon Aug 18 11:42:05 2025 from 18.206.107.28
[ec2-user@ip-10-0-1-5 ~]$ ssh -i mykey.pem ec2-user@10.0.1.15
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

Last login: Mon Aug 18 11:42:25 2025 from 10.0.1.5
[ec2-user@ip-10-0-1-15 ~]$ curl http://localhost
<!DOCTYPE html><html lang="en"><head><meta charset="utf-8"/><meta name="viewport" content="width=device-width,initial-scale=1"/><meta name="theme-color" content="#000000"/><meta name="description" content="Web site created using create-react-app"/><title>React App</title><script defer="defer" src="/static/js/main.2466d2aa.js"></script><link href="/static/css/main.b20bfae4.css" rel="stylesheet"></head><body><noscript>You need to enable JavaScript to run this app.</noscript><div id="root"></div></body></html>[ec2-user@ip-10-0-1-15 ~]$ 
```

Now, everything is fine, then take AMI for Auto Scaling process for External Load Balancer.

Step 11: Creating External Application Load Balancer for Web-tier configuration

- Create External Load Balancer without creating instance inside the target group.

Procedure to Create External ALB:

→ Create the Target group

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

- Offers the flexibility for a Network Load Balancer to accept and route TCP requests within a specific VPC.
- Facilitates using static IP addresses and PrivateLink with an Application Load Balancer.

Target group name

ExternalTG1

→ Select port number 80 and select the VPC we created.

target group

modified after creation.

individual targets during registration.

HTTP 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-Ofc171b4597b75040 (Project-vpc)
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 Check Path is 'health', which is configured in backend process.

target group

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

Health check port
The port the load balancer uses when performing health checks on targets. By default, the health check port is the same as the target group's traffic port. However, you can specify a different port as an override.

Traffic port

Override

Healthy threshold
The number of consecutive health checks successes required before considering an unhealthy target healthy.

2

Restore defaults

→ Creating LB without registering the EC2 instance inside the target group.

- Step 1
Specify group details
- 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 (5)

<input type="checkbox"/>	Instance ID	Name	State	Security groups	Zone
<input type="checkbox"/>	i-0b4346e83d116b3a5	App_as	Running	App-Server	us-east-1b
<input type="checkbox"/>	i-053aaa446aee8c944	App_as	Running	App-Server	us-east-1c
<input type="checkbox"/>	i-03f3242e5987c9e71	WebServer	Running	Webserver	us-east-1a
<input type="checkbox"/>	i-036a970ab16f30120	JumpServer	Running	Jump-Server	us-east-1a
<input type="checkbox"/>	i-0c61895b78a18b76e	AppServer	Running	App-Server	us-east-1b

0 selected

Ports for the selected instances

Ports for routing traffic to the selected instances.

jet group

0 selected

Ports for the selected instances

Ports for routing traffic to the selected instances.

80
1-65535 (separate multiple ports with commas)

Review targets

Targets (0)

No instances added yet

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

0 pending

→ Create Application Load Balancer and add the Target group inside the ALB.

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.

 Dualstack

Includes IPv4 and IPv6 addresses.

 Dualstack without public IPv4

Includes a public IPv6 address, and private IPv4 and IPv6 addresses. Compatible with **internet-facing** load balancers only.

→ Select VPC and Public subnets for multiple Available Zones.

EC2 > Load balancers > Create Application Load Balancer

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 endpoints. To confirm the VPC for your targets, view target groups.

vpc-0fc171b4597b75040 (Project-vpc)
10.0.0.0/16

Create VPC [Create VPC](#)

IP pools - new [Info](#)

You can optionally choose to configure an IPAM pool as the preferred source for your load balancer's IP addresses. Create or view Pools in 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-az6)

Subnets

Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses are required for your load balancer to scale efficiently.

subnet-02a0a295fe1eddb92
IPv4 subnet CIDR: 10.0.1.0/24

us-east-1b (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.

public-sub-01

→ Select the Web Server Security group

→ Select the External Target group inside the External Load Balancer.

EC2 > Load balancers > Create Application Load Balancer

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

Webserver
sg-0832c71b2686856a0 VPC: vpc-0fc171b4597b75040

Listeners and routing [Info](#)

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

▼ Listener HTTP:80

Protocol	Port
HTTP	80

Default action [Info](#)

Forward to [ExternalTG1](#) Target type: Instance, IPv4

HTTP

[Create target group](#)

Listener tags - optional

Consider adding tags to your listener. Tags enable you to categorize your AWS resources so you can more easily manage them.

[Add listener tag](#)

External Load Balancer is created successfully.

Step12: Creating Auto Scaling for External Application Load Balancer

- Generate Auto Scaling for External ALB and Select the (Public) Web-tier configuration Ec2 instance as the Template.

Procedure to Create Auto Scaling:

Create AMI Template

EC2 > Instances > i-03f3242e5987c9e71 > Create image



Create image Info

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

Image details

Instance ID

i-03f3242e5987c9e71 (WebServer)

Image name

WebAmi

Maximum 127 characters. Can't be modified after creation.

Image description - optional

Image description

Maximum 255 characters

Reboot instance

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

Instance volumes

Storage type	Device	Snapshot	Size	Volume type	IOPS	Throughput	Delete on termination	Encrypted
--------------	--------	----------	------	-------------	------	------------	-----------------------	-----------

EC2 > Launch templates > Create launch template

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

WebTemplate

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

Template version description

A prod webserver for MyApp

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

EC2 > Launch templates > Create launch template

Don't include in launch template Owned by me Shared with me

Amazon Machine Image (AMI)

webAmi
ami-0929c3e9cce2279a7
2025-08-15T00:16:36.000Z Virtualization: hvm ENA enabled: true Root device type: ebs Boot mode: uefi-preferred

Description

Architecture x86_64 AMI ID ami-0929c3e9cce2279a7

▼ 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

Free tier eligible All generations

Cancel Create launch template

Add AMI Template inside the Auto Scaling group.

- Choose the Web Server AMI Template inside the Auto Scaling group.
- EC2 instance will be automatically assigned inside the target group.

EC2 > Auto Scaling groups > Create Auto Scaling group

Choose launch template

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 [Info](#)

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

Name

Auto Scaling group name

Enter a name to identify the group.

ExternalASG1

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

Launch template [Info](#)

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

Launch template

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

Webtemplate

Create a launch template

Version

Default (1)

Create a launch template version

- Select VPC and Public subnets for multiple Available Zones.
- Attach to an Existing Load Balancer.
- Select the External Load Balancer.

Scaling group

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

VPC
Choose the VPC that defines the virtual network for your Auto Scaling group.

vpc-0fc171b4597b75040 (Project-vpc)
10.0.0.0/16

Create a VPC

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

Select Availability Zones and subnets

use1-az6 (us-east-1a) | subnet-02a0a295fe1eddb92 (public-sub-01) X
10.0.1.0/24

Create a subnet

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

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

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

Cancel Skip to review Previous Next

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1 Choose launch template
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.

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
Create a basic load balancer to attach to your Auto Scaling group.

Attach to an existing load balancer

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

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

← 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

ExternalTG1 | HTTP X ←
Application Load Balancer: ExternalLB1

Fixing the desired capacity based on the real-time demand.

EC2 > Auto Scaling groups > Create Auto Scaling group

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.

1

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	Max desired capacity
1	2

Equal or less than desired capacity Equal or greater than desired capacity



Here I'm fixing the Target Value as 1 for testing purpose.

Scaling group

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 scale 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.

Scaling policy name

Target Tracking Policy

Metric type Info

Monitored metric that determines if resource utilization is too low or high. If using EC2 metrics, consider enabling detailed monitoring for better scaling performance.

Average CPU utilization

Target value

1



Instance warmup Info

300 seconds

Disable scale in to create only a scale-out policy

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1
Step 2
Step 3 - optional
Step 4 - optional
Step 5 - optional
Step 6 - optional
Add tags
Step 7
Review

Add tags - optional Info

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

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

Tags (1)

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

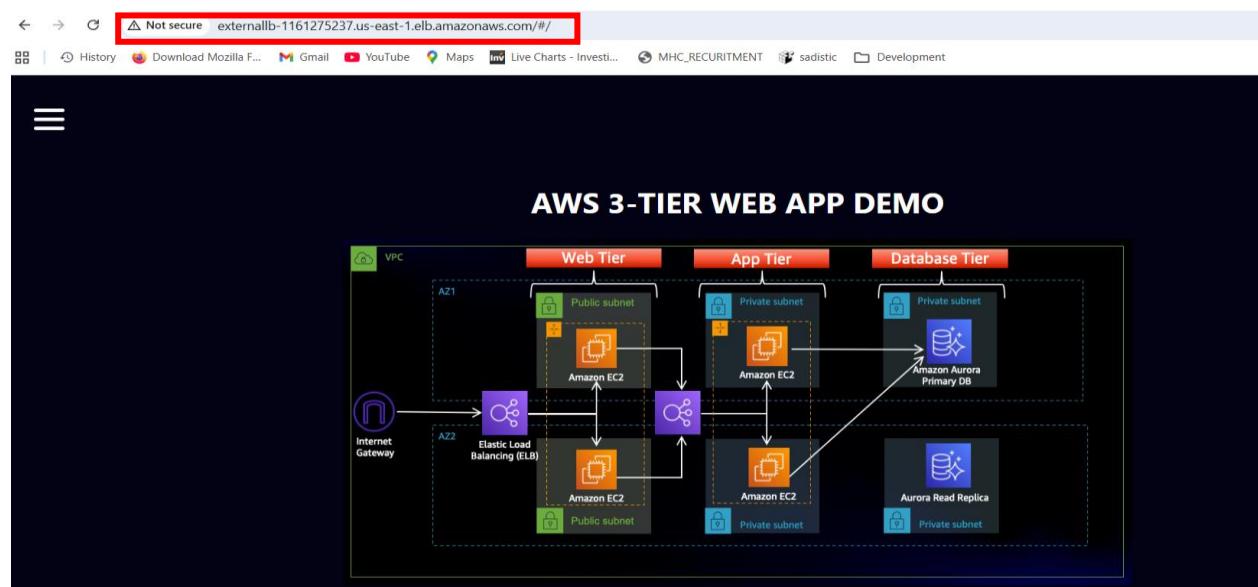
Add tag 49 remaining

Cancel Previous Next

Auto Scaling for External Load Balancer is created successfully.

Step13: Testing Application Locally

Copy the External Load Balancer URL and run in browser



Our Application is working fine.

Step14: Enabling CloudFront Distribution (CDN)

- Amazon CloudFront is a content delivery network service that accelerates the delivery of your static and dynamic web content to users globally.

- It achieves this by caching content at edge locations (data centers) around the world, bringing the content closer to your users and reducing latency.

→ Attach the External Load Balancer inside the CloudFront.

[CloudFront](#) > [Distributions](#) > Create distribution

We've streamlined the process of creating a CloudFront distribution. Continue here and let us know what you think. Or go to the previous [Create Distribution page](#).

Step 1

Get started

Step 2
Specify origin

Step 3
Enable security

Step 4
Get TLS certificate

Step 5
Review and create

Get started

Connect your websites, apps, files, video streams, and other content to CloudFront. We optimize the performance, reliability, and security for your web traffic.

Distribution options Info

Distribution name
Name will be stored as a tag on the resource. You can add a name, or more tags, later.

Description - optional

Distribution type

Single website or app
Choose if each website or application will have a unique configuration.

Multi-tenant architecture - *New*
Choose when you have multiple domains that need to share configurations. This is a common architecture for SaaS providers.



[CloudFront](#) > [Distributions](#) > Create distribution

We've streamlined the process of creating a CloudFront distribution. Continue here and let us know what you think. Or go to the previous [Create Distribution page](#).

Specify origin

Step 3
Enable security

Step 4
Review and create

Origin type

Your origin is where your content (such as a website or app) lives. CloudFront works with AWS-based origins and origins hosted on other cloud providers.

Amazon S3
Deliver static assets like files and images, statically generated websites or single page applications (SPA).

Elastic Load Balancer
Deliver applications hosted behind ELB such as dynamic websites, web services, and APIs.

Elemental MediaPackage
Deliver end-to-end live events or video on demand (VOD).

API Gateway
Deliver API endpoints for REST APIs hosted on API Gateway.

VPC origin
Deliver applications and content hosted within private VPCs, such as EC2 instances and Application Load Balancers.

Other
Refer to any AWS or non-AWS origin through its publicly resolvable URL.

Origin

Elastic Load Balancing origin
Choose an AWS origin, or enter your origin's domain name. [Learn more](#)

[Browse load balancers](#)

Origin path - optional
The directory path within your origin where your content is stored. [Learn more](#)

[CloudFront](#) > [Distributions](#) > Create distribution

We've streamlined the process of creating a CloudFront distribution. Continue here and let us know what you think. Or go to the previous [Create Distribution page](#).

Step 1

Get started

Specify origin

Enable security

Step 4
Review and create

Enable security

Web Application Firewall (WAF) Info

Enable security protections
Keep your application secure from the most common web threats and security vulnerabilities using AWS WAF. Blocked requests are stopped before they reach your web servers.

Do not enable security protections
Select this option if your application does not need security protections from AWS WAF.

[Cancel](#) [Previous](#) [Next](#)

The screenshot shows the AWS CloudFront console for a distribution named 'cf1' (Standard). The 'General' tab is selected. Key details include:

- Name:** cf1
- Distribution domain name:** d3m4srxbogztub.cloudfront.net
- ARN:** arn:aws:cloudfront::054728709811:distribution/E3ONZM14OBW3AP
- Last modified:** Deploying

The 'Settings' section includes:

- Description:** -
- Price class:** Use all edge locations (best performance)
- Supported HTTP versions:** HTTP/2, HTTP/1.1, HTTP/1.0
- Alternate domain names:** Add domain
- Logging:** Standard logging Off, Cookie logging Off, Default root object -

Run CDN through Http. Copy the domain and run in browser,

Creating Hosted Zone

- Purchase the Domain in GoDaddy website and Create the Hosted Zone with the same Domain name in Route 53 .
- Automatically Namespaces will be created inside the Hosted Zone.
- Copy the namespaces and paste inside the GoDaddy Domain website.

Enabling SSL Certificate (ACM)

- Request a Public Certificate

The screenshot shows the 'Request certificate' page in AWS Certificate Manager. The 'Certificate type' section is highlighted:

- Certificate type:** Request a public certificate (selected)
- Request a private certificate:** No private CAs available for issuance.

A note states: "Requesting a private certificate requires the creation of a private certificate authority (CA). To create a private CA, visit [AWS Private Certificate Authority](#)".

At the bottom right are 'Cancel' and 'Next' buttons.

- Provide the fully qualified Domain Name (contains sub domain and main domain)

Request public certificate

Domain names

Provide one or more domain names for your certificate.

Fully qualified domain name | [Info](#)

myproj.mycloudtech.shop

[Add another name to this certificate](#)

You can add additional names to this certificate. For example, if you're requesting a certificate for "www.example.com", you might want to add the name "example.com" so that customers can reach your site by either name.

Allow export [Info](#)

Disable export

Use this certificate only with integrated AWS services. The private key for this certificate will be disallowed for exporting from AWS.

Enable export

Export this certificate and private key for use with any TLS workflow. ACM will charge your account based on the requested domains when the certificate is issued for the first time and for each renewal.

Validation method [Info](#)

Select a method for validating domain ownership.

DNS validation - recommended

Choose this option if you are authorized to modify the DNS configuration for the domains in your certificate request.

→ Create the DNS record in Route 53.

AWS Certificate Manager (ACM)

List certificates
Request certificate
Import certificate
AWS Private CA

380dad48-477a-47f6-adb1-03623b942453

[Delete](#)

Certificate status

Identifier

380dad48-477a-47f6-adb1-03623b942453

Status

Pending validation [Info](#)

ARN

arn:aws:acm:us-east-1:054728709811:certificate/380dad48-477a-47f6-adb1-03623b942453

Type

Amazon Issued

Domains (1)

[Create records in Route 53](#)

[Export to CSV](#)

< 1 >

Domain	Status	Renewal status	Type	CNAME name
myproj.mycloudtech.shop	Pending validation	-	CNAME	_94efc1aad97ca6a599910a517c0d7e81.myproj.shop.

<input type="checkbox"/> Certificate ID	Domain name	Type	Status
7ca5195f-db3d-43dd-a160-cf30fdf3b120	myproj.mycloudtech.shop	Amazon Issued	Issued

DNS Record is Created in Route 53 and Request for public certificate (SSL) is Issued.

Adding Domain in CloudFront

→ Enter Fully qualified Domain Name which we have created for SSL.

The screenshot shows the 'Configure domains' step in the AWS CloudFront distribution configuration wizard. The left sidebar shows steps 1 through 3: Step 1 (Configure domains) is selected, Step 2 (Get TLS certificate) is the current step, and Step 3 (Review changes) is the next step. The main area is titled 'Configure domains' with the sub-instruction 'Choose the domains that will be served by this distribution.' Below this is a 'Domains' section with a 'Domains to serve' list containing 'myproj.mycloudtech.shop'. A red box highlights this list. At the bottom right of the main area are 'Cancel' and 'Next' buttons.

Now, Certificate is selected, Add the Domain.

The screenshot shows the 'Get TLS certificate' step in the AWS CloudFront distribution configuration wizard. The left sidebar shows steps 1 through 3: Step 1 (Configure domains) is completed, Step 2 (Get TLS certificate) is selected, and Step 3 (Review changes) is the next step. The main area is titled 'Get TLS certificate' with a 'TLS certificate' section. It shows an available certificate for 'myproj.mycloudtech.shop' with ARN 'arn:aws:acm:us-east-1:054728709811:certificate/7ca5195f-db3d-43dd-a160-cf30fdf3b120'. A red arrow points to this certificate entry. Below it is a 'Create a new certificate' option. At the bottom right are 'View in AWS Certificate Manager' and 'Cancel' buttons. The 'Next' button is highlighted in orange.

CloudFront > Distributions > E3ONZM14OBW3AP > Add domain

Step 1
Configure domains

Step 2
Get TLS certificate

Step 3
Review changes

Review changes

Domain Domains to serve myproj.mycloudtech.shop [Edit](#)

TLS certificate ARN arn:aws:acm:us-east-1:054728709811:certificate/7ca5195fdb3d43dd-a160-cf30fdf3b120 [Edit](#)

Covered domains myproj.mycloudtech.shop

Source Amazon

[Cancel](#) [Previous](#) [Add domains](#)

CloudFront > Distributions > E2EXJM7NFC3W8T

CloudFront cf Standard [View metrics](#)

Distributions

- Policies
- Functions
- Static IPs
- VPC origins
- What's new

SaaS

- Multi-tenant distributions
- Distribution tenants

Telemetry

- Monitoring
- Alarms
- Logs

Reports & analytics

- Cache statistics
- Popular objects
- Top referrers

General Security Origins Behaviors Error pages Invalidations Tags Logging

Details

Name	Distribution domain name cf d2m2u8uemk1y3x.cloudfront.net	ARN arn:aws:cloudfront::054728709811:distribution/E2EXJM7NFC3W8T	Last modified August 18, 2025 at 10:54:00 AM UTC
------	--	---	---

Settings

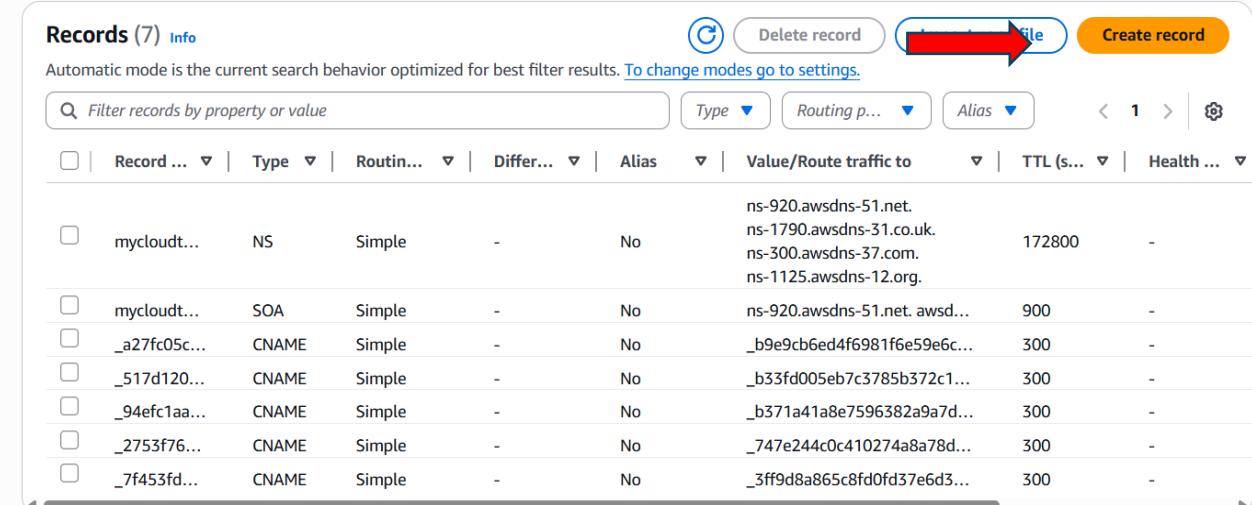
Description	Alternate domain names myproj.mycloudtech.shop Edit	Standard logging Off
Price class	Route domains to CloudFront	Cookie logging Off
Supported HTTP versions	Custom SSL certificate myproj.mycloudtech.shop Edit	Default root object
	Security policy TLSv1.2_2021	

→ SSL Certificate is enabled for CloudFront.

→ Now, Route the CloudFront Distribution in Route 53.

Step15: Hosting Application in Route53

- Amazon Route 53 is to act as a highly available and scalable Domain Name System (DNS) web service for routing internet traffic to applications and resources
- It translates human-readable domain names (like "www.example.com") into the numerical IP addresses that computers use to connect. Additionally, Route 53 offers domain registration and health checking capabilities.



Records (7) Info							
Automatic mode is the current search behavior optimized for best filter results. To change modes go to settings.							
<input type="text"/> Filter records by property or value Type Routing p... Alias TTL (s...) Health ...							
	Record ...	Type	Routin...	Differ...	Alias	Value/Route traffic to	
<input type="checkbox"/>	mycloudt...	NS	Simple	-	No	ns-920.awsdns-51.net. ns-1790.awsdns-31.co.uk. ns-300.awsdns-37.com. ns-1125.awsdns-12.org.	172800
<input type="checkbox"/>	mycloudt...	SOA	Simple	-	No	ns-920.awsdns-51.net. awsd...	900
<input type="checkbox"/>	_a27fc05c...	CNAME	Simple	-	No	_b9e9cb6ed4f6981f6e59e6c...	300
<input type="checkbox"/>	_517d120...	CNAME	Simple	-	No	_b33fd005eb7c3785b372c1...	300
<input type="checkbox"/>	_94efc1aa...	CNAME	Simple	-	No	_b371a41a8e7596382a9a7d...	300
<input type="checkbox"/>	_2753f76...	CNAME	Simple	-	No	_747e244c0c410274a8a78d...	300
<input type="checkbox"/>	_7f453fd...	CNAME	Simple	-	No	_3ff9d8a865c8fd0fd37e6d3...	300

Create the Record in Hosted Zone:

- Select the Record type
- Enable Alias
- Route the traffic : Alias to CloudFront distribution
- Selecting Simple routing policy.

Route 53 > Hosted zones > mycloudtech.shop > Create record

Record name Info myproj	.mycloudtech.shop	Record type Info A – Routes traffic to an IPv4 address and some AWS resources
Keep blank to create a record for the root domain.		
<input checked="" type="radio"/> Alias		
Route traffic to Info Alias to CloudFront distribution	US East (N. Virginia)	An alias to a CloudFront distribution and another record in the same hosted zone are global and available only in US East (N. Virginia).
<input type="text"/> d2m2u8uemk1y3x.cloudfront.net		<input type="button"/>
Routing policy Info Simple routing	Evaluate target health	<input checked="" type="radio"/> No
<input type="button"/> Add another record		
		<input type="button"/> Create records

Route 53 > Hosted zones > mycloudtech.shop

Records (1/8) Info									
Automatic mode is the current search behavior optimized for best filter results. To change modes go to settings.									
<input type="text"/> Filter records by property or value <input type="button"/> Delete record <input type="button"/> Import zone file <input type="button"/> Create record									
	Record ...	Type	Routing ...	Differ...	Alias	Value/Route traffic to	TTL (s...)	Health ...	
<input type="checkbox"/>	mycloudt...	NS	Simple	-	No	ns-920.awsdns-51.net. ns-1790.awsdns-31.co.uk. ns-300.awsdns-37.com. ns-1125.awsdns-12.org.	172800	-	
<input type="checkbox"/>	mycloudt...	SOA	Simple	-	No	ns-920.awsdns-51.net.awsd...	900	-	
<input type="checkbox"/>	_a27fc05c...	CNAME	Simple	-	No	_b9e9cb6ed4f6981f6e59e6c...	300	-	
<input type="checkbox"/>	S17d120...	CNAME	Simple	-	No	b33frf005eb7c3785h372c1...	300	-	
<input checked="" type="checkbox"/>	myproj.m...	A	Simple	-	Yes	d2m2u8uemk1y3x.cloudfron...	-	-	
<input type="checkbox"/>	_94efc1aa...	CNAME	Simple	-	No	_b371a41a8e7596382a9a7d...	300	-	
<input type="checkbox"/>	_2753f76...	CNAME	Simple	-	No	_747e244c0c410274a8a78d...	300	-	
<input type="checkbox"/>	_7f453fd...	CNAME	Simple	-	No	_3ff9d8a865c8fd0fd37e6d3...	300	-	

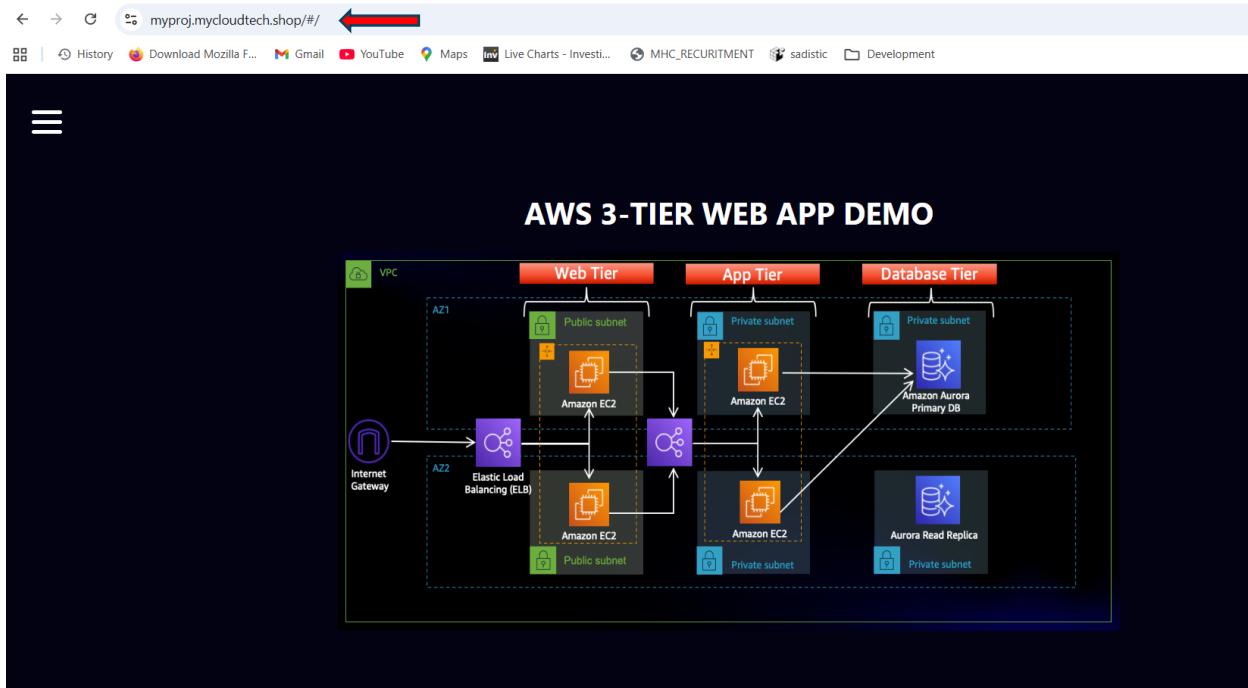
Record details

 Edit record

Record name	<input type="text"/> myproj.mycloudtech.shop
Record type	A
Value	<input type="text"/> d2m2u8uemk1y3x.cloudfront.net.
Alias	Yes
TTL (seconds)	-
Routing policy	Simple

Now our application is hosted in Route 53, let's run the application.

Run: myproj.mycloudtech.shop



ID	AMOUNT	DESC
ADD		
18	100	fruits
19	200	Almond

AWS Three-Tier Architecture Project has deployed Successfully.