

AWS Cloud Infrastructure Deployment Project

VPC settings

Resources to create [Info](#)

Create only the VPC resource or the VPC and other networking resources.

 VPC only VPC and more

Name tag auto-generation [Info](#)

Enter a value for the Name tag. This value will be used to auto-generate Name tags for all resources in the VPC.

 Auto-generate

Agri

IPv4 CIDR block [Info](#)

Determine the starting IP and the size of your VPC using CIDR notation.

10.0.0.0/16

65,536 IPs

CIDR block size must be between /16 and /28.

IPv6 CIDR block [Info](#)

 No IPv6 CIDR block Amazon-provided IPv6 CIDR block

Tenancy [Info](#)

Default



Number of Availability Zones (AZs) [Info](#)

Choose the number of AZs in which to provision subnets. We recommend at least two AZs for high availability.

1 | **2** | 3

► Customize AZs

Number of public subnets [Info](#)

The number of public subnets to add to your VPC. Use public subnets for web applications that need to be publicly accessible over the internet.

0 | **2**

Number of private subnets [Info](#)

The number of private subnets to add to your VPC. Use private subnets to secure backend resources that don't need public access.

0 | **2** | 4

► Customize subnets CIDR blocks

NAT gateways (\$) [Info](#)

Choose the number of Availability Zones (AZs) in which to create NAT gateways. Note that there is a charge for each NAT gateway.

None | In 1 AZ | 1 per AZ

VPC endpoints [Info](#)

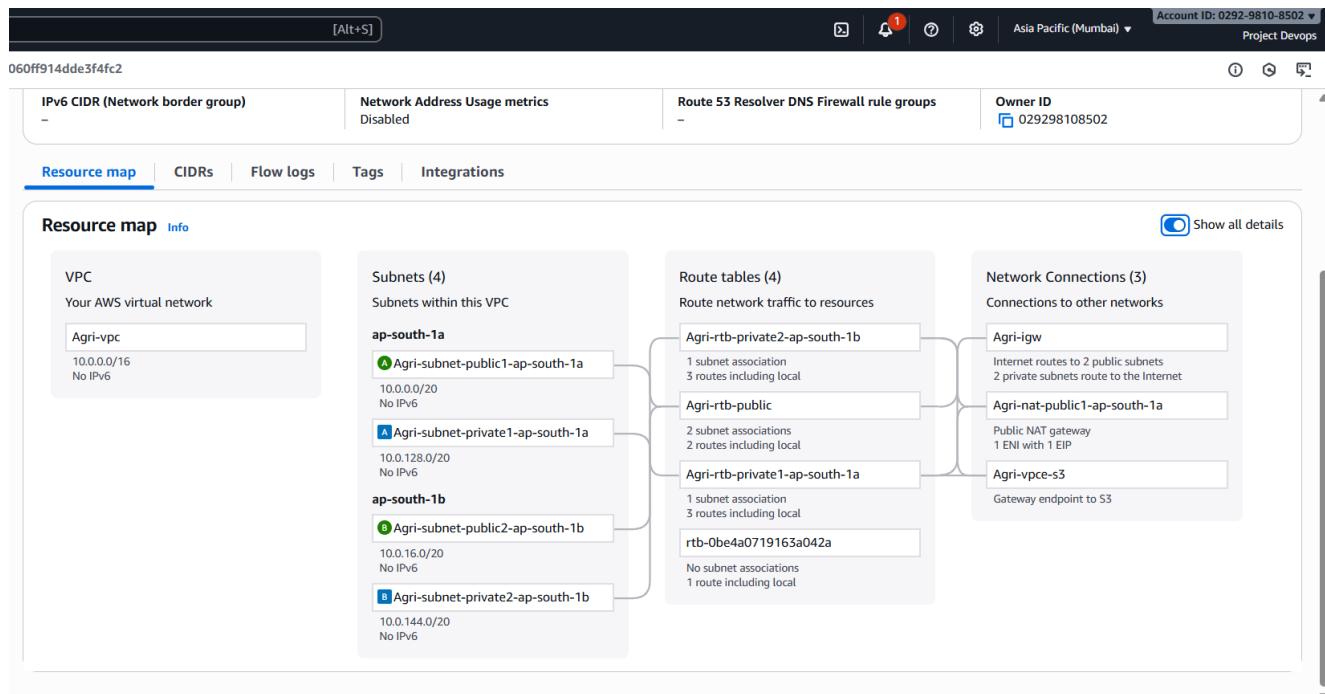
Endpoints can help reduce NAT gateway charges and improve security by accessing S3 directly from the VPC. By default, full access policy is used. You can customize this policy at any time.

None | **S3 Gateway**

DNS options [Info](#)

- Enable DNS hostnames
- Enable DNS resolution

► Additional tags



Database:

Account ID: 0292-9810-8502 ▾ Project Devops

aws Search [Alt+S]

Aurora and RDS > Databases > Create database

Create database [Info](#)

Choose a database creation method

- Standard create
You set all of the configuration options, including ones for availability, security, backups, and maintenance.
- Easy create
Use recommended best-practice configurations. Some configuration options can be changed after the database is created.

Engine options

Engine type [Info](#)

- Aurora (MySQL Compatible)
- Aurora (PostgreSQL Compatible)
- MySQL
- PostgreSQL
- MariaDB
- Oracle
- Microsoft SQL Server
- IBM Db2

Account ID: 0292-9810-8502 ▾ Project Devops

aws Search [Alt+S]

Aurora and RDS > Databases > Create database

Templates
Choose a sample template to meet your use case.

- Production
Use defaults for high availability and fast, consistent performance.
- Dev/Test
This instance is intended for development use outside of a production environment.
- Sandbox
To develop new applications, test existing applications, or gain hands-on experience with Amazon RDS.

Availability and durability

Deployment options [Info](#)
Choose the deployment option that provides the availability and durability needed for your use case. AWS is committed to a certain level of uptime depending on the deployment option you choose. Learn more in the Amazon RDS service level agreement (SLA) [\[SLA\]](#).

- Multi-AZ DB cluster deployment (3 instances)
Creates a primary DB instance with two readable standbys in separate Availability Zones. This setup provides:
 - 99.95% uptime
 - Redundancy across Availability Zones
 - Increased read capacity
 - Reduced write latency
- Multi-AZ DB instance deployment (2 instances)
Creates a primary DB instance with a non-readable standby instance in a separate Availability Zone. This setup provides:
 - 99.95% uptime
 - Redundancy across Availability Zones
- Single-AZ DB instance deployment (1 instance)
Creates a single DB instance without standby instances. This setup provides:
 - 99.5% uptime
 - No data redundancy

Write/read endpoint AZ 1 Primary instance + SSD

Reader endpoints AZ 2 Readable Standby + SSD AZ 3 Readable Standby + SSD

Write/read endpoint AZ 1 Primary instance

Standby (no endpoint) AZ 2 Standby

Write/read endpoint AZ 1 Primary instance

Aurora and RDS > Databases > Create database

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.

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.

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.

IPv4
Your resources can communicate only over the IPv4 addressing protocol.

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 instance.

Agri-vpc (vpc-060ff914dde3f4fc2)
4 Subnets, 2 Availability Zones

Only VPCs with a corresponding DB subnet group are listed.

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

DB subnet group Info
Choose the DB subnet group. The DB subnet group defines which subnets and IP ranges the DB instance can use in the VPC that you selected.

default-vpc-060ff914dde3f4fc2
4 Subnets, 2 Availability Zones

Public access Info

Aurora and RDS > Databases > Create database

Public access Info

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

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

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

AgriDatabase X

Availability Zone Info
No preference

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.

Aurora and RDS > Databases > Create database

▼ Additional configuration

Database options, encryption turned on, backup turned off, backtrack turned off, maintenance, CloudWatch Logs, delete protection turned off.

Database options

Initial database name Info
agri

If you do not specify a database name, Amazon RDS does not create a database.

DB parameter group Info
default.postgres17

Option group Info
default:postgres-17

Backup

Enable automated backup
Creates a point-in-time snapshot of your database

Enable encryption
Choose to encrypt the given instance. Master key IDs and aliases appear in the list after they have been created using the AWS Key Management Service console. [Info](#)

AWS KMS key Info
(default) aws/rds

Account

aws Search [Alt+S] Account ID: 0292-9810-8502 Project Devops

Amazon S3 > Buckets > Create bucket

General configuration

AWS Region: Asia Pacific (Mumbai) ap-south-1

Bucket type: General purpose Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

Directory Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name: **my-agriloan-app**

Bucket names must be 3 to 63 characters and unique within the global namespace. Bucket names must also begin and end with a letter or number. Valid characters are a-z, 0-9, periods (.), and hyphens (-). [Learn More](#)

Copy settings from existing bucket - optional Only the bucket settings in the following configuration are copied.

Choose bucket

Format: s3://bucket/prefix

Object Ownership [Info](#)

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

Object Ownership:

- ACLs disabled (recommended) All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.
- ACLs enabled Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership: Bucket owner enforced

aws Search [Alt+S] Account ID: 0292-9810-8502 Project Devops

Amazon S3 > Buckets > Create bucket

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

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

aws Search [Alt+S] Account ID: 0292-9810-8502 Project Devops

Amazon S3 > Buckets > my-agriloan-app > Create folder

If your bucket policy prevents uploading objects without specific tags, metadata, or access control list (ACL) grantees, you will not be able to create a folder using this configuration. Instead, you can use the [upload configuration](#) to upload an empty folder and specify the appropriate settings.

Folder

Folder name: **images**

Folder names can't contain "/". [See rules for naming](#)

Server-side encryption [Info](#)

Server-side encryption protects data at rest.

The following encryption settings apply only to the folder object and not to sub-folder objects.

Server-side encryption:

- Don't specify an encryption key The bucket settings for default encryption are used to encrypt the folder object when storing it in Amazon S3.
- Specify an encryption key The specified encryption key is used to encrypt the folder object before storing it in Amazon S3.

If your bucket policy requires objects to be encrypted with a specific encryption key, you must specify the same encryption key when you create a folder. Otherwise, folder creation will fail.

Create folder

IAM

Screenshot of the AWS IAM 'Create user' wizard Step 1: Specify user details.

The page shows a navigation bar with 'Search' and 'Account ID: 0292-9810-8502'. The breadcrumb path is 'IAM > Users > Create user'. A sidebar on the left lists steps: Step 1 (Specify user details, highlighted), Step 2 (Set permissions), and Step 3 (Review and create).

User details

User name: S3User

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = . @ _ - (hyphen).

Provide user access to the AWS Management Console - optional

If you're providing console access to a person, it's a [best practice](#) to manage their access in IAM Identity Center.

Info: If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. [Learn more](#)

Buttons at the bottom: 'Cancel' and 'Next'.

Screenshot of the AWS IAM 'Create user' wizard Step 2: Set permissions.

The breadcrumb path is 'IAM > Users > Create user'. The sidebar shows Step 1 (Specify user details) completed, Step 2 (Set permissions, highlighted), and Step 3 (Review and create).

Set permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

Permissions options

Add user to group: Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.

Copy permissions: Copy all group memberships, attached managed policies, and inline policies from an existing user.

Attach policies directly: Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

Permissions policies (1/1399)

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

Filter by Type: All types, 1 match.

Policy name	Type	Attached entities
AmazonS3FullAccess	AWS managed	1

Set permissions boundary - optional

Buttons at the bottom: 'Cancel', 'Previous', and 'Next'.

Screenshot of the AWS IAM 'Create user' wizard Step 3: Create access key.

The breadcrumb path is 'IAM > Users > S3User > Create access key'. The sidebar shows Step 2 (optional) completed, Step 3 (Retrieve access keys, highlighted).

Use case

Command Line Interface (CLI): You plan to use this access key to enable the AWS CLI to access your AWS account.

Local code: You plan to use this access key to enable application code in a local development environment to access your AWS account.

Application running on an AWS compute service: You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

Third-party service: You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.

Application running outside AWS: You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.

Other: Your use case is not listed here.

Alternatives recommended

- Use [AWS CloudShell](#), a browser-based CLI, to run commands. [Learn more](#)
- Use the [AWS CLI V2](#) and enable authentication through a user in IAM Identity Center. [Learn more](#)

Confirmation

I understand the above recommendation and want to proceed to create an access key.

Buttons at the bottom: 'Cancel' and 'Next'.

Frontend Instance

The screenshot shows the AWS EC2 Security Groups page. On the left, there's a sidebar with links for Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Capacity Manager (New), Images (AMIs, AMI Catalog), Elastic Block Store (Volumes, Snapshots, Lifecycle Manager), Network & Security (Security Groups, Elastic IPs, Placement Groups, Key Pairs), and a general Project Devops section.

The main content area displays the details for the security group "sg-02271cda0fbcd4da7 - AgriFrontend". It includes fields for Security group name (AgriFrontend), Security group ID (sg-02271cda0fbcd4da7), Description (80, 22), Owner (029298108502), Inbound rules count (4 Permission entries), and Outbound rules count (1 Permission entry). Below this, the "Inbound rules" tab is selected, showing four rules:

Name	Security group rule ID	IP version	Type	Protocol	Port range
-	sgr-09aca045f5c5311076	IPv4	HTTPS	TCP	443
-	sgr-0fab65dc0ff87df98	IPv4	Custom TCP	TCP	5173
-	sgr-08b9f0be6148aa38e	IPv4	HTTP	TCP	80
-	sgr-03f2c2dce47f4aae	IPv4	SSH	TCP	22

The screenshot shows the AWS EC2 Launch an instance page. On the left, the "Network settings" section includes fields for VPC (selected VPC: vpc-060ff914dde3f4fc2 (Agri-vpc)), Subnet (selected subnet-06127bae57751a58), Auto-assign public IP (Enable), Firewall (security groups) (Select existing security group), and Common security groups (AgriFrontend sg-02271cda0fbcd4da7). On the right, the "Summary" section shows the following details:

- Number of instances: 1
- Software Image (AMI): Canonical, Ubuntu, 24.04, amd6... (Read more)
- Virtual server type (instance type): t2.micro
- Firewall (security group): AgriFrontend
- Storage (volumes): 1 volume(s) - 8 GiB

At the bottom, there are "Cancel", "Launch instance", and "Preview code" buttons.

```
#!/bin/bash

set -e

sudo apt update -y

sudo apt upgrade -y

sudo apt install -y curl

curl -fsSL https://deb.nodesource.com/setup_lts.x | sudo -E bash -

sudo apt install -y nodejs

sudo apt install npm -y
```

Backend Instance

The screenshot shows the AWS EC2 Security Groups page. The left sidebar includes links for Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Capacity Manager, Images, AMIs, AMI Catalog, Elastic Block Store (Volumes, Snapshots, Lifecycle Manager), Network & Security (Security Groups, Elastic IPs, Placement Groups, Key Pairs), and CloudShell/Feedback.

The main content displays the details for the security group **sg-086dd831e503ed7d9 - AgriBackend**. It lists the security group name (AgriBackend), security group ID (sg-086dd831e503ed7d9), owner (029298108502), description (8080, 22), and VPC ID (vpc-060ff914dde3f4fc2). The **Inbound rules** tab is selected, showing four entries:

Name	Security group rule ID	IP version	Type	Protocol	Port range
-	sgr-0ddc179894620eb89a	IPv4	SSH	TCP	22
-	sgr-080842a8d68a36b46	IPv4	HTTP	TCP	80
-	sgr-0ce159adf45c78a37	IPv4	HTTPS	TCP	443
-	sgr-0fd64b0ef2d346f49	IPv4	Custom TCP	TCP	8080

Other tabs include Outbound rules, Sharing - new, VPC associations - new, and Tags. The bottom right corner shows copyright information: © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences.

The screenshot shows the AWS EC2 Launch an instance page. The left sidebar includes links for Instances, Network settings, Auto-assign public IP, Firewall (security groups), Common security groups, Advanced network configuration, and CloudShell/Feedback.

The main content shows the **Network settings** section, which includes a VPC dropdown set to "vpc-060ff914dde3f4fc2 (Agri-vpc) 10.0.0.0/16", a Subnet dropdown set to "subnet-069640c78fcb2bba4 Agri-subnet-private1-ap-south-1a", and a "Create new subnet" button. It also includes sections for Auto-assign public IP (disabled), Firewall (security groups) (with "Select existing security group" selected), and Common security groups (listing "AgriBackend sg-086dd831e503ed7d9").

The right side shows the **Summary** section, which includes the number of instances (1), Software Image (AMI) (Canonical, Ubuntu, 24.04, amd64), Virtual server type (instance type) (t2.micro), Firewall (security group) (AgriBackend), Storage (volumes) (1 volume(s) - 8 GiB), and buttons for "Cancel", "Launch instance", and "Preview code".

At the bottom right, there is copyright information: © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences.

```
#!/bin/bash

set -e

sudo apt update -y

sudo apt upgrade -y

sudo apt install openjdk-17-jdk -y

sudo apt install maven -y
```

Send the pem key to public instance(Frontend) for access the private instance(Backend)

C:\Users\Anbarasu\Downloads>scp -i agri.pem agri.pem ubuntu@13.235.254.132:/home/ubuntu/

```
C:\Users\Anbarasu\Downloads>scp -i agri.pem agri.pem ubuntu@13.235.254.132:/home/ubuntu/
The authenticity of host '13.235.254.132 (13.235.254.132)' can't be established.
ED25519 key fingerprint is SHA256:S2lLUYAhvUpnqsq/bQR+afGBgIkNE4ifYptBAmBbo.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
Warning: Permanently added '13.235.254.132' (ED25519) to the list of known hosts.
agri.pem
```

100% 1674 3.6KB/s 00:00

C:\Users\Anbarasu\Downloads>ssh -i agri.pem [ubuntu@13.235.254.132](https://13.235.254.132)

```
ubuntu@ip-10-0-5-128:~$ node -v
v18.19.1
ubuntu@ip-10-0-5-128:~$ npm -v
9.2.0
ubuntu@ip-10-0-5-128:~$ ls
agri.pem
ubuntu@ip-10-0-5-128:~$ |
```

ubuntu@ip-10-0-4-190:~\$ sudo apt install -y curl nginx

ubuntu@ip-10-0-4-190:/\$ sudo mkdir -p /var/www/frontend

ubuntu@ip-10-0-4-190:/\$ sudo chown ubuntu:ubuntu /var/www/frontend

ubuntu@ip-10-0-4-190:/\$ cd /var/www

ubuntu@ip-10-0-4-190:/var/www\$ git clone https://github.com/Anbarasu-AN/Agriculture_Frontend.git frontend

ubuntu@ip-10-0-4-190:/var/www\$ cd frontend

ubuntu@ip-10-0-4-190:/var/www/frontend\$ sudo npm ci --force

ubuntu@ip-10-0-5-128:/var/www/frontend\$ cd src

ubuntu@ip-10-0-5-128:/var/www/frontend/src\$ sudo nano Config.jsx

// Paste your frontend public IP or domain

export const BASE_URL = 'http://13.232.140.236/api';

ubuntu@ip-10-0-4-190:/var/www/frontend\$ cd .. && npm run build

ubuntu@ip-10-0-5-128:/var/www/frontend\$ ls /var/www/html/

index.nginx-debian.html

ubuntu@ip-10-0-5-128:/var/www/frontend\$ sudo rm -rf /var/www/html/*

ubuntu@ip-10-0-5-128:/var/www/frontend\$ ls /var/www/html/

ubuntu@ip-10-0-4-190:/var/www/frontend\$ sudo cp -r dist/* /var/www/html/

ubuntu@ip-10-0-5-128:/var/www/frontend\$ ls /var/www/html/

Allsmart.png assets index.html

ubuntu@ip-10-0-4-190:/var/www/frontend\$ sudo ls /etc/nginx/sites-enabled/

default

ubuntu@ip-10-0-4-190:/var/www/frontend\$ sudo nano /etc/nginx/sites-available/default

```

# Paste this below para

server {
    listen 80;
    server_name _;

    root /var/www/html;
    index index.html;

    # serve static files
    location / {
        try_files $uri /index.html;
    }

    # proxy API requests to backend private IP
    location /api/ {
        proxy_http_version 1.1;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_pass http://10.0.131.108:8080/;
        proxy_read_timeout 90;
    }

    add_header X-Frame-Options "SAMEORIGIN";
    add_header X-Content-Type-Options "nosniff";
}

```

```

ubuntu@ip-10-0-5-128:/var/www/frontend$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
ubuntu@ip-10-0-5-128:/var/www/frontend$ sudo systemctl reload nginx
ubuntu@ip-10-0-5-128:/var/www/frontend$ sudo systemctl enable nginx
Synchronizing state of nginx.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable nginx
ubuntu@ip-10-0-5-128:/var/www/frontend$ sudo systemctl restart nginx
ubuntu@ip-10-0-5-128:/var/www/frontend$ cd ~
ubuntu@ip-10-0-5-128:~$ ls
agri.pem

```

```
ubuntu@ip-10-0-5-128:~$ chmod 700 agri.pem
```

Login into Backend using private IP

```
ubuntu@ip-10-0-5-128:~$ ssh -i agri.pem ubuntu@10.0.131.108
```

```
ubuntu@ip-10-0-131-108:~$ java --version
```

```
ubuntu@ip-10-0-131-108:~$ mvn -v
```

```
ubuntu@ip-10-0-131-108:~$ cd /opt
```

```
ubuntu@ip-10-0-131-108:/opt$ sudo mkdir -p agriculture-backend
```

```
ubuntu@ip-10-0-131-108:/opt$ sudo chown ubuntu:ubuntu /opt/agriculture-backend
```

```
ubuntu@ip-10-0-131-108:/opt$ cd /opt/agriculture-backend
```

```
ubuntu@ip-10-0-131-108:/opt/agriculture-backend$ git clone https://github.com/Anbarasu-A-N/Agriculture\_Backend.git .
```

```
ubuntu@ip-10-0-131-108:/opt/agriculture-backend$ sudo nano  
src/main/resources/application.properties
```

```
# Change this Configuration
```

```
# RDS Endpoint, Database name, username, password
```

```
spring.datasource.url=jdbc:postgresql://project.cnk8gomow5fx.ap-south-1.rds.amazonaws.com/agri
```

```
spring.datasource.username=$ MENTION_YOUR_USERNAME_HERE
```

```
spring.datasource.password=MENTION_YOUR_PASSWORD_HERE
```

```
spring.datasource.driver-class-name=org.postgresql.Driver
```

```
# SMTP Server Configuration
```

```
# SMTP username and password
```

```
spring.mail.host=smtp.gmail.com
```

```
spring.mail.port=587
```

```
spring.mail.properties.mail.smtp.auth=true
```

```
spring.mail.properties.mail.smtp.starttls.enable=true
```

```
spring.mail.username= MENTION_YOUR_USERNAME_HERE
```

```
spring.mail.password=MENTION_YOUR_PASSWORD_HERE
```

```
# Frontend Public IP, Backend Private IP  
allowed.origins=http://13.235.254.132:80,http://13.235.254.132:80/*  
jwt.server.url=http://10.0.131.108:8080
```

```
# S3 Bucket, Folder name, Access key, Secret Key  
# AWS S3 Configuration  
aws.s3.bucket=MENTION_YOUR_S3_NAME_HERE  
aws.s3.folder=MENTION_YOUR_S3_FOLDER_NAME_HERE  
aws.access.key.id=MENTION_YOUR_SECRET_ACCESS_KEY_HERE  
aws.secret.access.key=MENTION_YOUR_SECRET_ACCESS_KEY_HERE  
aws.region=us-east-1
```

Now Test

```
ubuntu@ip-10-0-131-108:/opt/agriculture-backend$ mvn clean install package  
ubuntu@ip-10-0-131-108:/opt/agriculture-backend$ java -jar target/app.jar
```

Find the port and kill

```
sudo ss -tulnp | grep 8080  
kill -9 <pid>
```

Simple Background Run (temporary)

If you just want it to keep running after logout (not on reboot):

```
ubuntu@ip-10-0-131-108:/opt/agriculture-backend$ nohup java -jar target/app.jar > app.log 2>&1 &
```

Above steps for communicate the frontend and backend without Load Balancer and ASG

Create Target Groups

Backend Target Group

1. Go to EC2 → Target Groups → Create target group
2. Target type → Instances
3. Protocol → HTTP, Port → 8080
4. Name → agri-backend-tg
5. VPC → same as your backend instance
6. Health check path → /actuator/health # Default lightweight health endpoint in Spring Boot

Screenshot of the AWS EC2 Target Groups 'Create target group' wizard.

Step 1: Create target group

Basic configuration

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
agri-backend-tg
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.
 8080
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-0817b5b38de9de355 (Agri-vpc)
10.0.0.0/16

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.
 /actuator/health
Up to 1024 characters allowed.

Ports for the selected instances
Ports for routing traffic to the selected instances.
8080
1-65535 (separate multiple ports with commas)

Include as pending below

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 target group

Frontend Target Group

1. Target type → Instances
2. Protocol → HTTP, Port → 80
3. Name → agri-frontend-tg
4. Health check path → /

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
agri-frontend-tg
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-0817b5b38de9de355 (Agri-vpc)
10.0.0.0/16

Create Application Load Balancers

Backend ALB (Internal)

1. EC2 → Load Balancers → Create Load Balancer
2. Type → Application Load Balancer
3. Name → agri-backend-alb
4. Scheme → Internal
5. Network → same VPC

6. Select two private subnets

7. Security Group → allow **HTTP (8080)** only from frontend subnet or frontend ALB SG

8. Listener → HTTP:80 → forward to agri-backend-tg

The screenshot shows the 'Create Application Load Balancer' wizard. In the 'Basic configuration' step, the load balancer name is set to 'agri-backend-alb'. The 'Scheme' dropdown is set to 'Internal', which is highlighted with a blue border. Below it, the 'Load balancer IP address type' section shows 'IPv4' selected. A note indicates that IPv4 includes only IPv4 addresses, while Dualstack includes both IPv4 and IPv6.

The screenshot shows the 'Network mapping' step. Under 'Availability Zones and subnets', two zones are selected: 'ap-south-1a (aps1-az1)' and 'ap-south-1b (aps1-az3)'. Each zone has a corresponding subnet listed: 'subnet-065bf401251703a0' for ap-south-1a and 'subnet-07bf04f5b0042120f' for ap-south-1b. Both subnets are IPv4 subnets with CIDR ranges 10.0.128.0/20 and 10.0.144.0/20 respectively. The 'Agri-subnet-private1-ap-south-1a' and 'Agri-subnet-private2-ap-south-1b' dropdown menus are also visible.

The screenshot shows the 'Listeners and routing' step. A new listener rule is being created for 'HTTP:80'. The 'Protocol' is set to 'HTTP' and the 'Port' is set to '80'. The 'Default action' is set to 'Forward to target groups'. A single target group, 'agri-backend-tg', is selected. The 'Weight' for this target group is set to '1' and the 'Percent' is set to '100%'. The 'Forward to target group' dropdown shows the target group selected.

Frontend ALB (Public)

1. Name → agri-frontend-alb
2. Scheme → **Internet-facing**
3. Network → same VPC
4. Select **two public subnets**
5. Security Group → allow **HTTP (80)** from anywhere (0.0.0.0/0)
6. Listener → HTTP:80 → forward to agri-frontend-tg

The screenshot shows the 'Basic configuration' step of the Application Load Balancer creation wizard. It includes fields for the load balancer name ('agri-frontend-alb'), scheme ('Internet-facing'), and load balancer IP address type ('IPv4'). The 'IPv4' option is selected, showing its details: it includes only IPv4 addresses.

Load balancer name:
Name must be unique within your AWS account and can't be changed after the load balancer is created.
agri-frontend-alb
A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Scheme: Internet-facing

Load balancer IP address type: IPv4

The screenshot shows the 'Network mapping' step of the Application Load Balancer creation wizard. It includes fields for the VPC and subnets.

VPC: vpc-0817b5b38de9de355 (Agri-vpc)
10.0.0.0/16

Create VPC: Create VPC

IP pools: 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: 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.

ap-south-1a (aps1-az1)
Subnet: subnet-0225977d5682418a0
IPv4 subnet CIDR: 10.0.0.0/20

ap-south-1b (aps1-az3)
Subnet: subnet-05714e5246f351c5
IPv4 subnet CIDR: 10.0.16.0/20

Agri-subnet-public1-ap-south-1a
Agri-subnet-public2-ap-south-1b

The screenshot shows the AWS CloudFormation interface for creating a new Application Load Balancer. In the 'Security groups' section, 'AgriFrontend' is selected. Under 'Listeners and routing', a new listener for 'HTTP:80' is configured with port 80. The 'Default action' is set to 'Forward to target groups'. A single target group, 'agri-frontend-tg', is listed with a weight of 1 and a percent of 100%. Other options like 'Redirect to URL' and 'Return fixed response' are also available.

Backend Setting!!!

Update the application.properties file

```
sudo nano /opt/agriculture-backend/src/main/resources/application.properties
```

```
# Frontend Public IP, Backend Private IP
```

```
allowed.origins=http://agri-frontend-alb-1705890550.ap-south-1.elb.amazonaws.com
```

```
jwt.server.url=http://internal-agri-backend-alb-886281513.ap-south-1.elb.amazonaws.com
```

```
cd /opt/agriculture-backend
```

```
ubuntu@ip-10-0-139-42:/opt/agriculture-backend$ mvn clean package -DskipTests
```

```
ubuntu@ip-10-0-139-42:/opt/agriculture-backend$ sudo chmod 777 run.sh
```

```
ubuntu@ip-10-0-139-42:/opt/agriculture-backend$ ./run.sh
```

Setting up agriculture-backend service...

Created symlink /etc/systemd/system/multi-user.target.wants/agriculture-backend.service → /etc/systemd/system/agriculture-backend.service.

agriculture-backend service has been started successfully!

Status command: sudo systemctl status agriculture-backend

Logs command: sudo journalctl -u agriculture-backend -f

Frontend Setting!!!

Update Backend IP to Backend Endpoint /etc/nginx/sites-available/default

sudo nano /etc/nginx/sites-available/default

```
server {  
    listen 80;  
    server_name _;  
  
    root /var/www/html;  
    index index.html;  
  
    # Serve React app  
    location / {  
        try_files $uri /index.html;  
    }  
  
    # Proxy API to Backend ALB  
    location /api/ {  
        proxy_http_version 1.1;  
        proxy_set_header Host $host;  
        proxy_set_header X-Real-IP $remote_addr;  
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;  
        proxy_set_header X-Forwarded-Proto $http_x_forwarded_proto;  
  
        # PASTE YOUR BACKEND ALB DNS HERE  
        proxy_pass http://internal-agri-backend-alb-886281513.ap-south-1.elb.amazonaws.com/;  
        proxy_read_timeout 90;  
    }  
  
    add_header X-Frame-Options "SAMEORIGIN";  
    add_header X-Content-Type-Options "nosniff";  
}
```

ubuntu@ip-10-0-12-31:~\$ **cd /var/www/frontend/src/**

ubuntu@ip-10-0-12-31:/var/www/frontend/src\$ **sudo nano Config.jsx**

// config.jsx

```
export const BASE_URL = 'http://agri-frontend-alb-1705890550.ap-south-1.elb.amazonaws.com/api';  
ubuntu@ip-10-0-12-31:/var/www/frontend/src$ cd ..
```

ubuntu@ip-10-0-12-31:/var/www/frontend\$ **npm run build**

ubuntu@ip-10-0-12-31:/var/www/frontend\$ **sudo rm -rf /var/www/html/***

ubuntu@ip-10-0-12-31:/var/www/frontend\$ **sudo cp -r dist/* /var/www/html/**

ubuntu@ip-10-0-12-31:/var/www/frontend\$ **sudo systemctl reload nginx**

ubuntu@ip-10-0-12-31:/var/www/frontend\$ **sudo nginx -t**

nginx: the configuration file /etc/nginx/nginx.conf syntax is ok

nginx: configuration file /etc/nginx/nginx.conf test is successful

ubuntu@ip-10-0-12-31:/var/www/frontend\$ sudo systemctl restart nginx

EC2 > Target groups > agri-backend-tg > Register targets

Register targets

Select instances, specify ports, and add the instances to the list of pending targets. Repeat to add additional combinations of instances and ports to the list of pending targets. Once you are satisfied with your selections, click Register pending targets.

Available instances (1/2)

Instance ID	Name	State	Security groups	Zone	Private IPv4 address	Subnet ID
i-0907c7bcb3dfeea6	AgriBackend	Running	AgriBackend	ap-south-1a	10.0.139.42	subnet-065bff40
i-060fa6a267b17e162	AgriFrontend	Running	AgriFrontend	ap-south-1a	10.0.12.31	subnet-0225977c

1 selected

Ports for the selected instances
Ports for routing traffic to the selected instances.
8080
1-65535 (separate multiple ports with commas)

[Include as pending below](#)

EC2 > Target groups > agri-frontend-tg > Register targets

Register targets

Select instances, specify ports, and add the instances to the list of pending targets. Repeat to add additional combinations of instances and ports to the list of pending targets. Once you are satisfied with your selections, click Register pending targets.

Available instances (1/2)

Instance ID	Name	State	Security groups	Zone	Private IPv4 address	Subnet ID
i-0907c7bcb3dfeea6	AgriBackend	Running	AgriBackend	ap-south-1a	10.0.139.42	subnet-065bff40
i-060fa6a267b17e162	AgriFrontend	Running	AgriFrontend	ap-south-1a	10.0.12.31	subnet-0225977c

1 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](#)

EC2 > Target groups > agri-backend-tg

agri-backend-tg

Details
arn:aws:elasticloadbalancing:ap-south-1:029298108502:targetgroup/agri-backend-tg/8261a61ae280ebb

Target type	Protocol	Protocol version
Instance	HTTP: 8080	HTTP1
IP address type	Load balancer	VPC
IPv4	agri-backend-alb	vpc-0817b5b38de9de355

Total targets	Healthy	Unhealthy	Unused	Initial	Draining
1	1	0	0	0	0
0 Anomalous					

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

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

Registered targets (1)

Instance ID	Name	Port	Zone	Health status	Health status details	Administrative over...	Override details	Launch time	Anomalous
i-0907c7bcb3dfeea6	AgriBackend	8080	ap-south-1a (...	Healthy	-	No override	No override is currentl...	October 2...	Green

Auto Scaling Group

AMI for Frontend

The screenshot shows the AWS EC2 Instances page. There are two instances listed:

- AgriFrontend (i-060fa6a267b17e162) - Running, t2.micro, 2/2 checks passed
- AgriBackend (i-0907c7bcb3deffea6) - Running, t2.micro

A context menu is open over the AgriBackend instance, with the "Create image" option highlighted.

Below the instances, the details for the AgriBackend instance are shown:

i-0907c7bcb3deffea6 (AgriBackend)	
Details	
Instance ID	i-0907c7bcb3deffea6
Public IPv4 address	-
Private IPv4 addresses	10.0.139.42

The "Create image" dialog is open, showing the following configuration:

Image details

Instance ID: i-0907c7bcb3deffea6 (AgriBackend)

Image name: AgriBackendImage

Image description - optional: V1

Reboot instance

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

Tags - optional

Tag image and snapshots together

Tag image and snapshots separately

AMI for Backend

Instances (1/2) [Info](#)

Last updated 2 minutes ago

Connect Instance state Actions ▲ Launch instances ▾

Find Instance by attribute or tag (case-sensitive)

All states ▾

Name	Instance ID	Instance state	Instance type	Status check	Alarm state
<input checked="" type="checkbox"/> AgriFrontend	i-060fa6a267b17e162	Running	t2.micro	2/2 checks passed	View alarm
<input type="checkbox"/> AgriBackend	i-0907c7bcb3deffea6	Running	t2.micro		

Create image Create template from instance Launch more like this

i-060fa6a267b17e162 (AgriFrontend)

Details Status and alarms Monitoring Security Networking Storage Tags

Instance summary [Info](#)

Instance ID [i-060fa6a267b17e162](#)

Public IPv4 address [13.201.84.124](#) | [open address](#)

Private IPv4 addresses [10.0.12.31](#)

Image details

Instance ID [i-060fa6a267b17e162 \(AgriFrontend\)](#)

Image name

Image description - optional

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/sd...	Create new snapshot from v...	8	EBS General Purpose SSD - ...	3000		<input checked="" type="checkbox"/> Enable	<input type="checkbox"/> Enable

[Add volume](#)

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

Tags - optional

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

Tag image and snapshots together

Tag the image and the snapshots with the same tag.

Tag image and snapshots separately

Tag the image and the snapshots with different tags.

Backend Launch Template

Screenshot of the AWS EC2 'Create launch template' wizard.

Launch template name and description

Launch template name - required
AgriBackend

Template version description
V1

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.

Summary

Software Image (AMI)
AgriBackendImage
ami-058f45f3abf8c673d

Virtual server type (instance type)
t2.micro

Firewall (security group)
AgriBackend

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 as part of the free tier.

Cancel Create launch template

Screenshot of the AWS EC2 'Create launch template' wizard.

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)
AgriBackendImage
ami-058f45f3abf8c673d
2025-10-25T12:44:48.000Z Virtualization: hvm ENA enabled: true Root device type: ebs Boot mode: uefi-preferred

Description
-

Architecture x86_64 **AMI ID** ami-058f45f3abf8c673d

Summary

Software Image (AMI)
AgriBackendImage
ami-058f45f3abf8c673d

Virtual server type (instance type)
t2.micro

Firewall (security group)
AgriBackend

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 as part of the free tier.

Cancel Create launch template

Screenshot of the AWS EC2 'Create launch template' wizard.

Instance type
t2.micro
Family: t2 1 vCPU 1 GiB Memory Current generation: true On-Demand Windows base pricing: 0.017 USD per Hour
On-Demand RHEL base pricing: 0.0268 USD per Hour On-Demand Linux base pricing: 0.0124 USD per Hour
On-Demand Ubuntu Pro base pricing: 0.0142 USD per Hour On-Demand SUSE base pricing: 0.0124 USD per Hour

Free tier eligible | All generations

Compare instance types

Additional costs apply for AMIs with pre-installed software

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
agri | Create new key pair

Network settings | Info

Subnet | Info
Don't include in launch template | Create new subnet

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

Availability Zone | Info
Don't include in launch template | Enable additional zones

Firewall (security groups) | Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Select existing security group | Create security group

Security groups | Info
Select security groups

AgriBackend sg-06ddfc141beedf265 | Create security group rules

Compare security group rules

Summary

Software Image (AMI)
AgriBackendImage
ami-058f45f3abf8c673d

Virtual server type (instance type)
t2.micro

Firewall (security group)
AgriBackend

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 as part of the free tier allowance.

Cancel Create launch template

Frontend Launch Template

Screenshot of the AWS EC2 'Create launch template' wizard.

Summary:

- Software Image (AMI):** AgriFrontendImage, ami-0639958955f0029af
- Virtual server type (instance type):** t2.micro
- Firewall (security group):** AgriFrontend
- Storage (volumes):** 1 volume(s) - 8 GiB

Launch template name and description:

- Launch template name - required: AgriFrontend
- Must be unique to this account. Max 128 chars. No spaces or special characters like '&', '<', '@'.

Template version description:

- V1
- Max 255 chars

Auto Scaling guidance: 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.

Create launch template

Screenshot of the AWS EC2 'Create launch template' wizard.

Summary:

- Software Image (AMI):** AgriFrontendImage, ami-0639958955f0029af
- Virtual server type (instance type):** t2.micro
- Firewall (security group):** AgriFrontend
- Storage (volumes):** 1 volume(s) - 8 GiB

Application and OS Images (Amazon Machine Image) - required:

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

Recent AMIs: Owned by me, Shared with me

Amazon Machine Image (AMI):

AgriFrontendImage ami-0639958955f0029af 2025-10-25T12:46:51.000Z	Virtualization: hvm	ENAv enabled: true	Root device type: ebs	Boot mode: uefi-preferred
--	---------------------	--------------------	-----------------------	---------------------------

Description:

-

Architecture: x86_64 **AMI ID:** ami-0639958955f0029af

Create launch template

Screenshot of the AWS EC2 'Create launch template' wizard.

Summary:

- Software Image (AMI):** AgriFrontendImage, ami-0639958955f0029af
- Virtual server type (instance type):** t2.micro
- Firewall (security group):** AgriFrontend
- Storage (volumes):** 1 volume(s) - 8 GiB

Instance type:

t2.micro
Family: t2
1 vCPU
1.7 GiB Memory
Current generation: true
On-Demand Windows base pricing: 0.017 USD per Hour
On-Demand RHEL base pricing: 0.0268 USD per Hour
On-Demand Linux base pricing: 0.0124 USD per Hour
On-Demand Ubuntu Pro base pricing: 0.0142 USD per Hour
On-Demand SUSE base pricing: 0.0124 USD per Hour

Key pair (login):

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: agri

Network settings:

Subnet: Don't include in launch template

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

Availability Zone: Don't include in launch template

Not applicable for EC2 Auto Scaling

Firewall (security groups):

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: AgriFrontend

Common security groups:

Select security groups: AgriFrontend sg-0ebdf7db9f851026b
VPC: vpc-0817b5b8d89d9355

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

Create launch template

Frontend ASG

aws Search [Alt+S] Account ID: 0292-9810-8502 ▾ Asia Pacific (Mumbai) ▾ Project Devops

EC2 > Auto Scaling groups > Create Auto Scaling group

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

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

Version
 [C](#)

Description
V1

AMI ID
ami-0639958955f0029af

Launch template
AgriFrontend [C](#)
lt-06bd53853be5947df

Instance type
t2.micro

Security groups
-

Request Spot Instances
No

aws Search [Alt+S] Account ID: 0292-9810-8502 ▾ Asia Pacific (Mumbai) ▾ Project Devops

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 7
Review

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.
 [Create a VPC](#) [C](#)

Availability Zones and subnets
Define which Availability Zones and subnets your Auto Scaling group can use in the chosen VPC.
 [C](#)

aps1-az1 (ap-south-1a) | subnet-0225977d5682418a0 (Agri-subnet-public1-ap-south-1a)
10.0.0.0/20

aps1-az3 (ap-south-1b) | subnet-05714e5246f351cc5 (Agri-subnet-public2-ap-south-1b)
10.0.16.0/20

Create a subnet [C](#)

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.

aws Search [Alt+S] Account ID: 0292-9810-8502 ▾ Asia Pacific (Mumbai) ▾ Project Devops

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.

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 [C](#)

agri-frontend-tg | HTTP
Application Load Balancer: agri-frontend-alb

VPC Lattice integration options [Info](#)

aws Search [Alt+S] Account ID: 0292-9810-8502 ▾ Project Devops

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
Configure group size and scaling (selected)
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.

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 Equal or less than desired capacity

Max desired capacity 5 Equal or greater than desired capacity

aws Search [Alt+S] Account ID: 0292-9810-8502 ▾ Project Devops

EC2 > Auto Scaling groups > Create Auto Scaling group

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.

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 50

Instance warmup Info

60 seconds

Disable scale in to create only a scale-out policy

aws Search [Alt+S] Account ID: 0292-9810-8502 ▾ Project Devops

EC2 > Auto Scaling groups > Create Auto Scaling group

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 Auto Scaling uses the Capacity Reservation preference from your launch template.

None Instances will not be launched into a Capacity Reservation.

Capacity Reservations only Instances will only be launched into a Capacity Reservation. If capacity isn't available, the instances fail to launch.

Capacity Reservations first Instances will attempt to launch into a Capacity Reservation first. If capacity isn't available, instances will run in On-Demand capacity.

Backend ASG

aws Search [Alt+S] Account ID: 0292-9810-8502 ▾ Project Devops

EC2 > Auto Scaling groups > Create Auto Scaling group

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

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

Version
 [Create a launch template version](#) [C](#)

Description	Launch template	Instance type
V1	AgriBackend lt-0208157690f4d0690	t2.micro
AMI ID	ami-058f45f5abf8c673d	Request Spot Instances No
Security groups	-	

aws Search [Alt+S] Account ID: 0292-9810-8502 ▾ Project Devops

EC2 > Auto Scaling groups > Create Auto Scaling group

Add tags Step 7 Review

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.
 [Create a VPC](#) [C](#)

Availability Zones and subnets
Define which Availability Zones and subnets your Auto Scaling group can use in the chosen VPC.
 [C](#)

[aps1-az3 \(ap-south-1b\) | subnet-07bf04f5b0042120f \(Agri-subnet-private2-ap-south-1b\) 10.0.144.0/20](#)

[aps1-az1 \(ap-south-1a\) | subnet-065bff401251703a0 \(Agri-subnet-private1-ap-south-1a\) 10.0.128.0/20](#)

[Create a subnet](#) [C](#)

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.

aws Search [Alt+S] Account ID: 0292-9810-8502 ▾ Project Devops

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.

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.

[C](#)

[agri-backend-tg | HTTP agri-backend-alb](#)

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

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	5

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.

aws Search [Alt+S] Account ID: 0292-9810-8502 Asia Pacific (Mumbai) Project Devops

EC2 > Auto Scaling groups > Create Auto Scaling group

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.

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

50

Instance warmup Info

60 seconds

Disable scale in to create only a scale-out policy

aws Search [Alt+S] Account ID: 0292-9810-8502 Asia Pacific (Mumbai) Project Devops

EC2 > Auto Scaling groups > Create Auto Scaling group

Control your Auto Scaling group's availability during instance replacement events. This includes health checks, instance terminations, 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 <input checked="" type="radio"/> 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 <input type="radio"/> 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 <input type="radio"/> 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 <input type="radio"/> 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
Auto Scaling uses the Capacity Reservation preference from your launch template.

None
Instances will not be launched into a Capacity Reservation.

Capacity Reservations only
Instances will only be launched into a Capacity Reservation. If capacity isn't available, the instances fail to launch.

Capacity Reservations first
Instances will attempt to launch into a Capacity Reservation first. If capacity isn't available, instances will run in On-Demand capacity.

Auto Scaling groups (2) Info		Last updated less than a minute ago	 Launch configurations	 Launch templates	 Actions	 Create Auto Scaling group					
<input type="text" value="Search your Auto Scaling groups"/> < 1 > ⚙️											
<input type="checkbox"/>	Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availability Zones	Creation time		
<input type="checkbox"/>	AgriBackendASG	AgriBackend Version Default	1	-	1	1	5	2 Availability Zones	Sat Oct 25 2025 18:55:5...		
<input type="checkbox"/>	AgriFrontendASG	AgriFrontend Version Default	1	-	1	1	5	2 Availability Zones	Sat Oct 25 2025 18:52:5...		

Instances (4) Info		Last updated less than a minute ago		C	Connect	Instance state	Actions	Launch instances	▼	
				All states	▼	<	1	>	☰	
		Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public
Images	AMIs									
AMIs			i-0650e92af5c7b7a55	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1b	-	-
Snapshots			i-0ee59fd52c536606	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1b	-	-
Lifecycle Manager		AgriFrontend	i-060fa6a267b17e162	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1a	ec2-13-201-84-124.ap...	13.201
Network & Security		AgriBackend	i-0907c7bcb5dfeffa6	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1a	-	-

EC2 < Target groups > agri-backend-tg

agri-backend-tg

Details

Target type: Instance
Protocol: Port
HTTP: 8080
IP address type: IPv4
Load balancer: agri-backend-alb

Protocol version: HTTP1
VPC: vpc-0817b5b38de9de355

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

0 Anomalous

Distribution of targets by Availability Zone (AZ)

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

Targets | Monitoring | Health checks | Attributes | Tags

Registered targets (2)

Anomaly mitigation: Not applicable

Deregister | Register targets

Instance ID	Name	Port	Zone	Health status	Health status details	Admini...	Overri...	Launch...	Anomaly detectio...
i-0650e92af5c7b7a55		8080	ap-south-1b (a...)	Healthy	-	No override.	No overri...	October 2...	Normal
i-0907c7bc3deffe6a	AgriBackend	8080	ap-south-1a (a...)	Healthy	-	No override.	No overri...	October 2...	Normal

The screenshot shows the AWS EC2 Target groups page. The left sidebar has sections for EC2, Dashboard, AWS Global View, Events, Instances (selected), Images, Elastic Block Store, Network & Security, and Project DevOps. The main content area shows the details for target group 'agri-frontend-tg'. It includes fields for Target type (Instance), IP address type (IPv4), Protocol (HTTP: 80), Protocol version (HTTP1), and VPC (vpc-0817b5b38de9de355). Below this, a table shows target counts: 2 Total targets, 2 Healthy, 0 Unhealthy, 0 Anomalous, 0 Unused, 0 Initial, and 0 Draining. A section titled 'Distribution of targets by Availability Zone (AZ)' indicates 2 healthy targets across two zones. The 'Targets' tab is selected in the navigation bar. The 'Registered targets (2)' section lists two targets: 'i-0ee59fdc52c536606' and 'i-060fa6267b17e162', both with port 80 and healthy status. The 'Actions' button is at the top right.