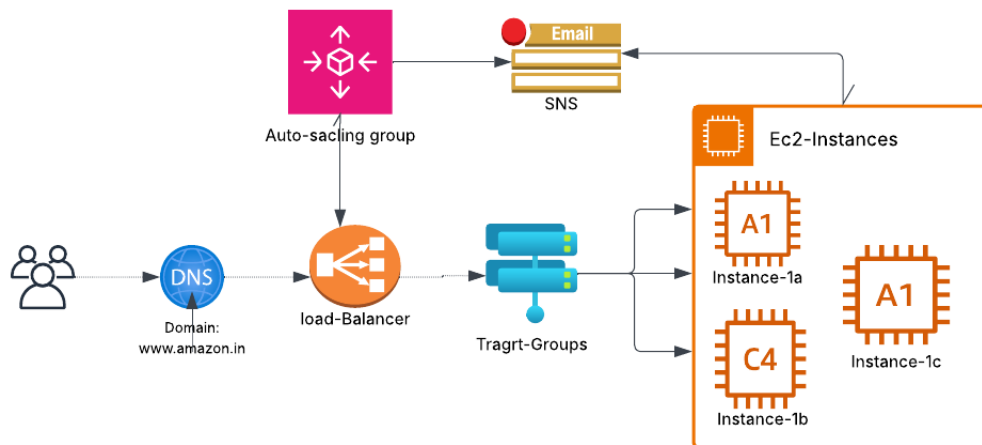


Deploying a Scalable Food Delivery App on AWS

Overview :

This guide walks through deploying a Food Delivery Web Application (similar to Swiggy or Zomato) on AWS using EC2, **Application Load Balancer** (ALB), **Auto Scaling Group** (ASG), and **SNS** for alerts. This architecture ensures the application is always available, highly responsive under heavy load, and can automatically scale to handle customer traffic. ideal for real-time food ordering platforms.

Architecture Overview:



- Users access the app using a domain name (e.g., www.amazon.in).
- Requests go to the Application Load Balancer (ALB), which evenly distributes traffic.
- The Target Group monitors and routes traffic only to healthy EC2 instances.
- Multiple EC2 Instances (e.g., Instance-1a, 1b, 1c) run in different Availability Zones for high availability.
- The Auto Scaling Group (ASG) adds or removes EC2 instances automatically based on CPU load.
- SNS (Simple Notification Service) sends email alerts during scale-in/scale-out events.

Steps to Deploy the Food App:

Step 1: Launch EC2 Instance & Deploy the App

- Navigate to EC2 > Instance > Launch Instances

www.linkedin.com/in/arun-kumar-akula

- Create instance named: Instance-1(in us-east-1a).
- AMI: Amazon Linux 2 (or) ubuntu
- Type: t2.micro
- Key pair: create (or) select existing key-pair
- Security Group: SSH & HTTP

Instances (1/1) Info		Last updated less than a minute ago		Connect	Instance state ▼	Actions ▼	Launch instances ▼
Find Instance by attribute or tag (case-sensitive)		All states ▼		< 1 > ⚙			
✓ Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	
✓ Instance-1	i-064eacee1b58088a7	Running 🔍 🔍	t2.micro	2/2 checks passed	View alarms +	us-east-1d	

SSH into Each Instance and Deploy the App :

➤ Commands to Deploy the App

```
sudo yum update -y
sudo yum install httpd -y
nginx -v
sudo systemctl start httpd
sudo systemctl enable httpd
sudo git clone https://github.com/Arunkumarakula/Food-app.git /var/www/html/
```

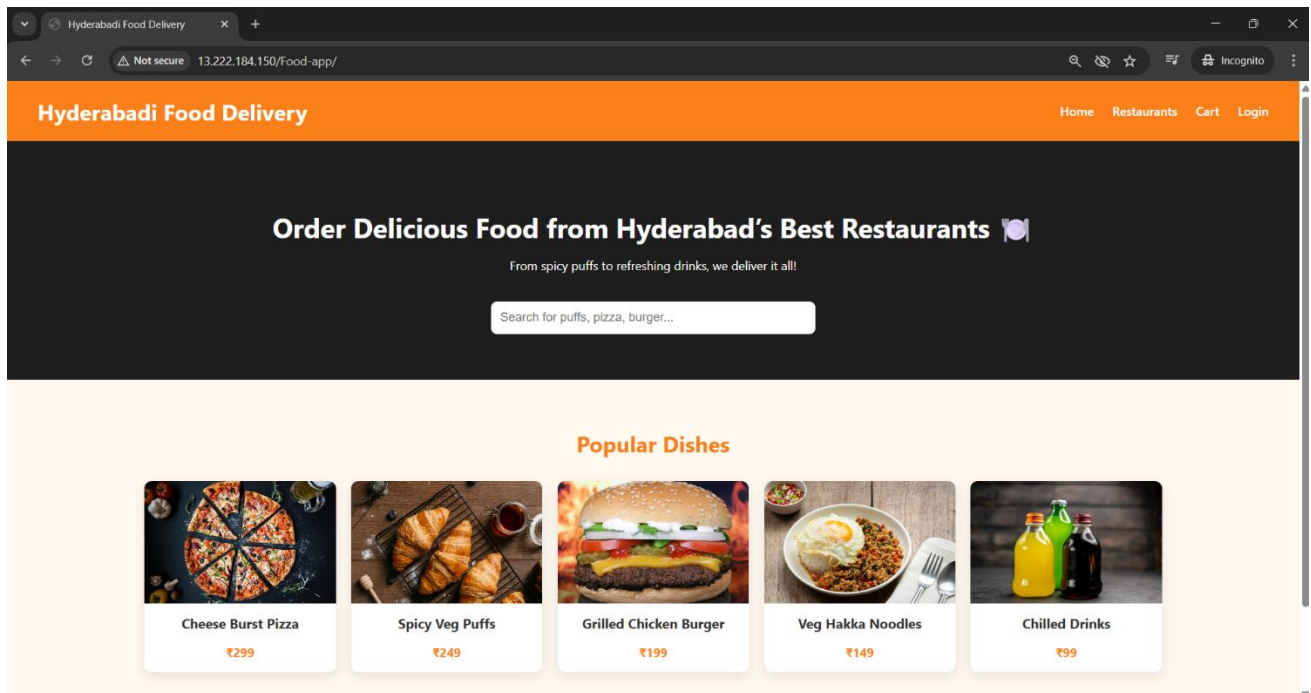
```
root@ip-172-31-22-126:~#
root@ip-172-31-22-126:~# nginx -v
nginx version: nginx/1.24.0 (Ubuntu)
root@ip-172-31-22-126:~#
```

```
root@ip-172-31-22-126:~#
root@ip-172-31-22-126:~# cd /var/www/html/
root@ip-172-31-22-126:/var/www/html# ls
index.nginx-debian.html
root@ip-172-31-22-126:/var/www/html# rm -rf index.nginx-debian.html
root@ip-172-31-22-126:/var/www/html# ls
root@ip-172-31-22-126:/var/www/html#
root@ip-172-31-22-126:/var/www/html# git clone https://github.com/Arunkumarakula/Food-app.git
Cloning into 'Food-app'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 4 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (4/4), done.
root@ip-172-31-22-126:/var/www/html# ls
Food-app
root@ip-172-31-22-126:/var/www/html# cd Food-app/
root@ip-172-31-22-126:/var/www/html/Food-app# ls
index.html styles.css
root@ip-172-31-22-126:/var/www/html/Food-app#
```

www.linkedin.com/in/arun-kumar-akula

- Test the application using the public IP address in your browser:

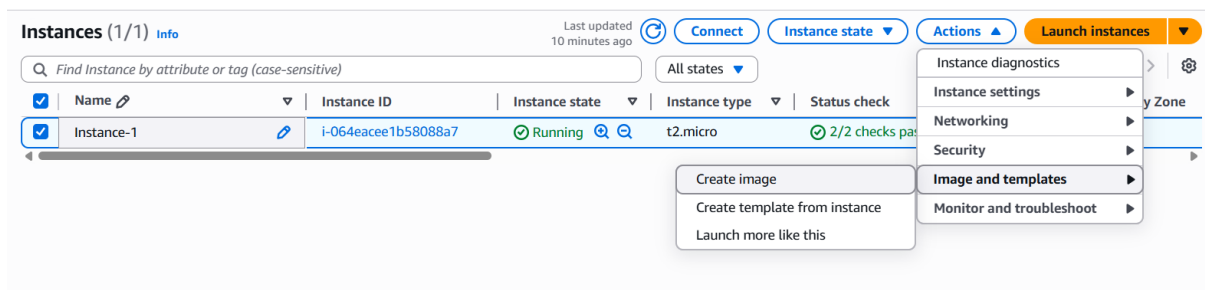
http://<your-ec2-public-ip>



Step 2: Create an EC2 Instance Using AMI For Load Balancer High Availability.

Steps to Create a New AMI from Your Existing EC2 Server

- Go to EC2 > Instances
- Select your running app instance (e.g: Server-1)
- Click Actions > Image and templates > Create image



Fill in:

- Image name: FoodApp-AMI
- Click Create Image

Wait a few minutes , go to **EC2 > AMIs** and make sure the AMI status shows **available**

Amazon Machine Images (AMIs) (1/1) Info

Owned by me

Find AMI by attribute or tag

Recycle Bin

EC2 Image Builder

Actions

Launch instance from AMI

Name	AMI name	AMI ID	Source	Owner	Visibility	Status	
Food-app-ami	ami-09f5ac8e10eead3ce	869546918261/Food-app-ami	869546918261	Private	Available		

Launch a New EC2 Instance from the New AMI

- Select your **FoodApp-AMI** → Click Launch instance from image
- Configure:
 - **Name:** FoodApp-Instance
 - **Instance type:** t2.micro
 - **Key pair:** my-keypair
 - **Network:** Default VPC
 - **Subnet:** Choose a different AZ (e.g., us-east-1a)
 - **Security Group:** SSH & HTTP
 - Click **Launch Instance**
- Test the application using the public IP address in your browser.(The output will be Food-app)

Step 3: Create a Target group to Load Balancer

- Navigate to EC2 > Target Groups
- Click Create target group
- Configure basic settings:
 - Target type: Instances
 - Name: e.g., FoodApp-TG
 - Protocol: HTTP
 - Port: 80
 - VPC: Choose your default or specific VPC
 - Click Next.

➤ Register Targets (EC2 Instances):

- Under Available instances, select one or more EC2 instances to attach
- Click **Include as pending below**

EC2 > Target groups > Create target group

Available instances (2)

<input type="checkbox"/>	Instance ID	Name	State	Security groups	Zone	Private IPv4 address	Subnet ID
<input type="checkbox"/>	i-085ef2d55e0847539	Server-2	Running	launch-wizard-35	us-east-1d	172.31.24.195	subnet-00fba79f2bacd83d
<input type="checkbox"/>	i-064eacee1b58088a7	Instance-1	Running	launch-wizard-34	us-east-1d	172.31.22.126	subnet-00fba79f2bacd83d

0 selected

Ports for the selected instances

Ports for routing traffic to the selected instances.

1-65535 (separate multiple ports with commas)

Include as pending below

2 selections are now pending below. Include more or register targets when ready.

Review targets

Targets (2)

Show only pending

Remove all pending

Instance ID	Name	Port	State	Security groups	Zone	Private IPv4 address	Subnet ID	Launch time
i-085ef2d55e0847539	Server-2	80	Running	launch-wizard-35	us-east-1d	172.31.24.195	subnet-00fba79f2bacd83d	July 3, 2025, 10:18 (UTC+05:30)
i-064eacee1b58088a7	Instance-1	80	Running	launch-wizard-34	us-east-1d	172.31.22.126	subnet-00fba79f2bacd83d	July 3, 2025, 09:53 (UTC+05:30)

2 pending

Cancel

Previous

Create target group

- Click **Create target group**

Your target group is now ready (This group will be used when you create or modify a **Load Balancer** to forward traffic to these registered EC2 instances.)

Step 4: Ceate a Load Balancer

- Go to AWS Console → **EC2 > Load Balancers**
- Click **Create Load Balancer**
- Choose **Application Load Balancer**
- Configure Load Balancer :
 - Name:** FoodApp-ALB
 - Scheme:** Internet-facing

Create Application Load Balancer [Info](#)

The Application Load Balancer distributes incoming HTTP and HTTPS traffic across multiple targets such as Amazon EC2 instances, microservices, and containers, based on request attributes. When the load balancer receives a connection request, it evaluates the listener rules in priority order to determine which rule to apply, and if applicable, it selects a target from the target group for the rule action.

► How Application Load Balancers work

Basic configuration

Load balancer name

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

My-Load-Balancer

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.

- **IP address type:** IPv4
- **Network mapping :** Default & Select at least two availability zones the actual app server's are running.

Network mapping [Info](#)

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

VPC [Info](#)

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

-

vpc-0b9c82995f074d71b
IPv4 VPC CIDR: 172.31.0.0/16

Ⓢ

IP pools - new [Info](#)

You can optionally choose to configure an IPAM pool as the preferred source for your load balancers IP addresses. Create or view Pools in [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)**

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-035f758bd581f5430
IPv4 subnet CIDR: 172.31.32.0/20

▼

☐ **us-east-1b (use1-az1)**

☐ **us-east-1c (use1-az2)**

☒ **us-east-1d (use1-az4)**

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-00fba79f2bacd836d
IPv4 subnet CIDR: 172.31.16.0/20

▼

☐ **us-east-1e (use1-az3)**

- **Listeners:** Add listener for HTTP on port 80
- Click **Next**
- **Configure Security Group :** Choose an existing Security Group or default (*Ensure it allows HTTP - port 80*).

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

Ⓢ

default
sg-0b0a8070f290a58d8 VPC: vpc-0b9c82995f074d71b

✕

Listeners and routing [Info](#)

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

▼ Listener HTTP:80

Remove

Protocol	Port	Default action Info
HTTP ▼	80 1-65535	<div>Forward to</div> <div>My-Target-Group Target type: Instance, IPv4</div> <div>HTTP ▼</div> <div>Ⓢ</div>

[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

You can add up to 50 more tags.

Add listener

Review and Create

- Review settings
- Click Create Load Balance

www.linkedin.com/in/arun-kumar-akula

- Test Your Load Balancer
- Once status becomes **active**
- Open in your browser

`http://<your-load-balancer-DNS-name>`

My-Load-Balancer

▼ Details

Load balancer type
Application

Scheme
Internet-facing

Status
Active

Hosted zone
Z35SXD0TRQ7X7K

VPC
vpc-0b9c82995f074d71b

Availability Zones
 subnet-035f758bd581f5430 us-east-1a (use1-az6)
 subnet-00fba79f2bacd836d us-east-1d

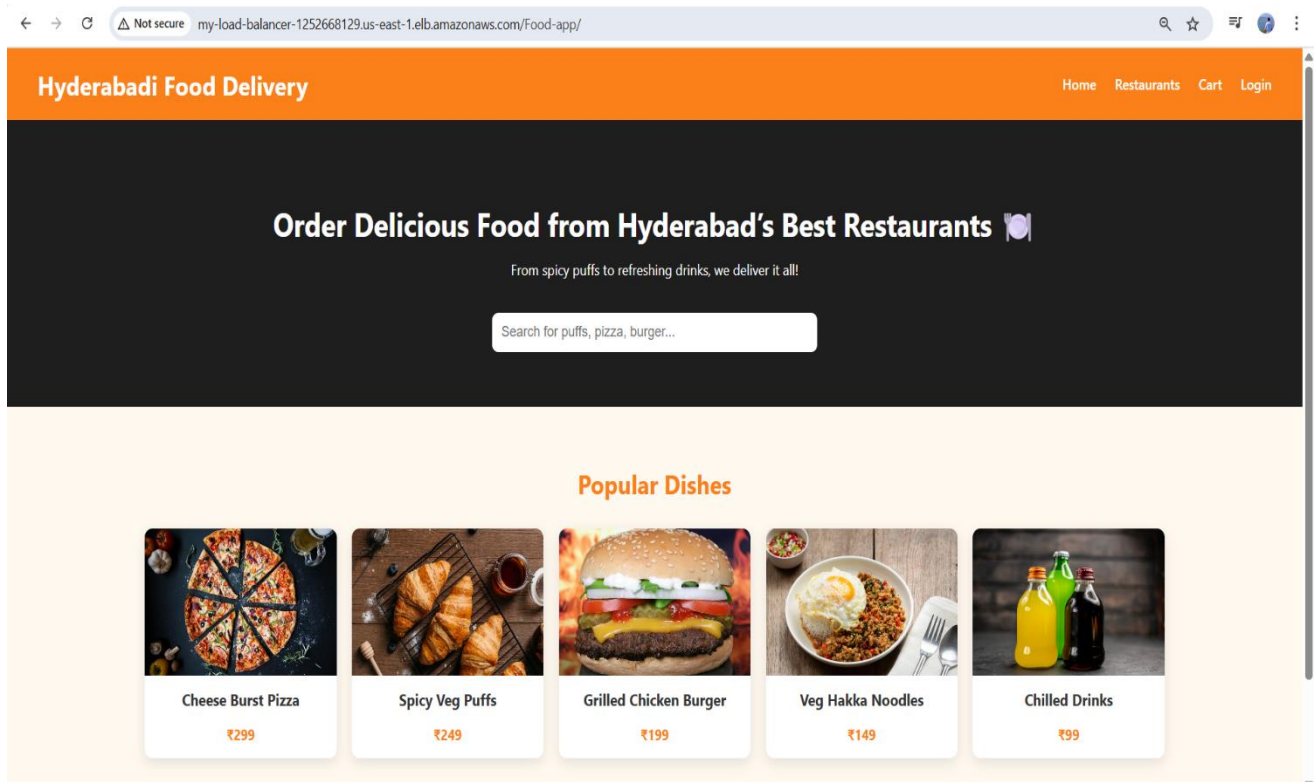
Load balancer IP address type
IPv4

Date created
July 3, 2025, 10:35 (UTC+05:30)

Load balancer ARN
 arn:aws:elasticloadbalancing:us-east-1:869546918261:loadbalancer/app/My-Load-Balancer/d0e3d232afa38626

DNS name copied
 My-Load-Balancer-1252668129.us-east-1.elb.amazonaws.com (A Record)

- You should see your food app served from any healthy instance in the target group.



Step 5: Create Auto Scaling Group with SNS Notifications

Prerequisites :

- A Launch Template or AMI ready (e.g., FoodApp-LTEMP)

www.linkedin.com/in/arun-kumar-akula

- A Target Group (e.g., FoodApp-TG)
- A Load Balancer created (e.g., FoodApp-ALB)
- A Security Group – default or existing
- A Key Pair

Create Auto Scaling Group:

- Go to EC2 > Auto Scaling Groups
- Click Create Auto Scaling group

Fill in:

Name: FoodApp-ASG

Launch Template: Select FoodApp-LTEMP or your AMI-based launch template

Click Next

- Configure Network
 - **VPC:** Choose default or custom VPC
 - **Availability Zones:** Select at least 2 (e.g., us-east-1a, 1b)
 - Click **Next**

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

Instance type requirements [Info](#) [Override launch template](#)

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

Launch template	Version	Description
My-Template ↗ lt-083f7bad0c7325349	Default	-

Instance type
t2.micro

Network [Info](#)

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

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

vpc-0b9c82995f074d71b
172.31.0.0/16 Default [↻](#)

[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-az2 (us-east-1c) | subnet-024f5f040b92bbf49 [✕](#)
172.31.80.0/20 Default
- use1-az3 (us-east-1e) | subnet-0bf055ee047ada9cd [✕](#)
172.31.48.0/20 Default
- use1-az6 (us-east-1a) | subnet-035f758bd581f5450 [✕](#)
172.31.32.0/20 Default
- use1-az4 (us-east-1d) | subnet-00fba79f2bacd836d [✕](#)
172.31.16.0/20 Default

[Create a subnet](#) [↗](#)

[Availability Zone distribution - new](#)

- Attach Load Balancer
 - Choose **Attach to an existing load balancer**
 - **Attach to existing load balancer** : choose from your existing load balancer target groups

www.linkedin.com/in/arun-kumar-akula

- **Target Group:** FoodApp-TG
- Click **Next**

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
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 ▼ 

My-Target-Group | HTTP ✕
Application Load Balancer: My-Load-Balancer

VPC Lattice integration options [Info](#)

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

Select VPC Lattice service to attach

- ☒ No VPC Lattice service
VPC Lattice will not manage your Auto Scaling group's network access and connectivity with other services.
- ☐ Attach to VPC Lattice service
Incoming requests associated with specified VPC Lattice target groups will be routed to your Auto Scaling group.

• Set Desired Capacity :

Group size:

- Minimum: 2
- Desired: 2
- Maximum: 4
- Click **Next**

Configure group size and scaling - *optional* [Info](#)

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

Group size [Info](#)

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

Desired capacity type

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

Units (number of instances) ▼

Desired capacity

Specify your group size.

2

Scaling [Info](#)

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

Scaling limits

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

Min desired capacity

1

Equal or less than desired capacity

Max desired capacity

2

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.

- **Configure Scaling Policies**
 - Choose **No scaling policies**
 - Metric type: **Average CPU utilization**
 - Target value: 70%
 - Click **Next**
- **Set-up SNS Topic**
 - Click **Create topic**
 - Name: FoodApp-SNS
 - With these recipients : youemail@gmail.com
 - Click **Next**

Add notifications - optional info

Send notifications to SNS topics whenever Amazon EC2 Auto Scaling launches or terminates the EC2 instances in your Auto Scaling group.

Notification 1 Remove

Send a notification to
ny-sns-topic

With these recipients
arunkumarakula58@gmail.com

[Use existing topic](#)

Event types
Notify subscribers whenever instances

- ☒ Launch
- ☒ Terminate
- ☒ Fail to launch
- ☒ Fail to terminate

[Add notification](#)

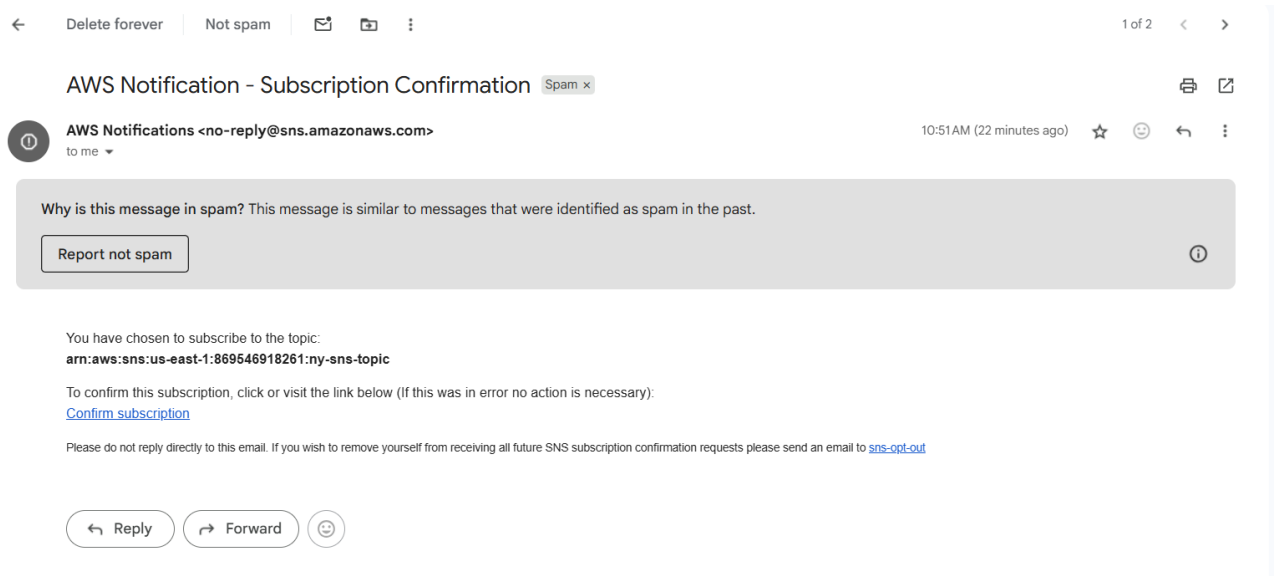
[Cancel](#) [Skip to review](#) [Previous](#) [Next](#)

- **Review and Create**
 - Review all configurations
 - Click **Create Auto Scaling Group**

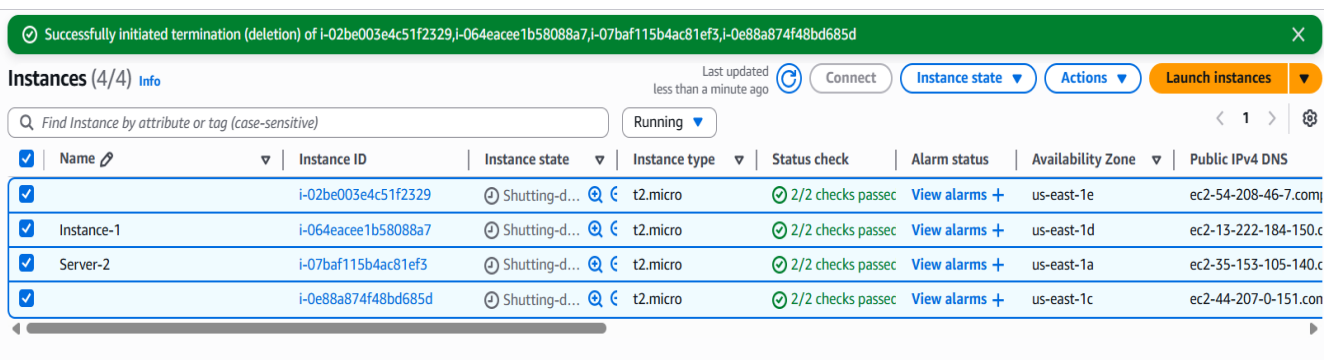
Auto Scaling groups (1) Info Last updated less than a minute ago [Launch configurations](#) [Launch templates](#) [Actions](#) [Create Auto Scaling group](#)

<input type="checkbox"/>	Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availability ...
<input type="checkbox"/>	My-ASG	My-Template Version Default	2	-	2	1	2	4 Availability ...

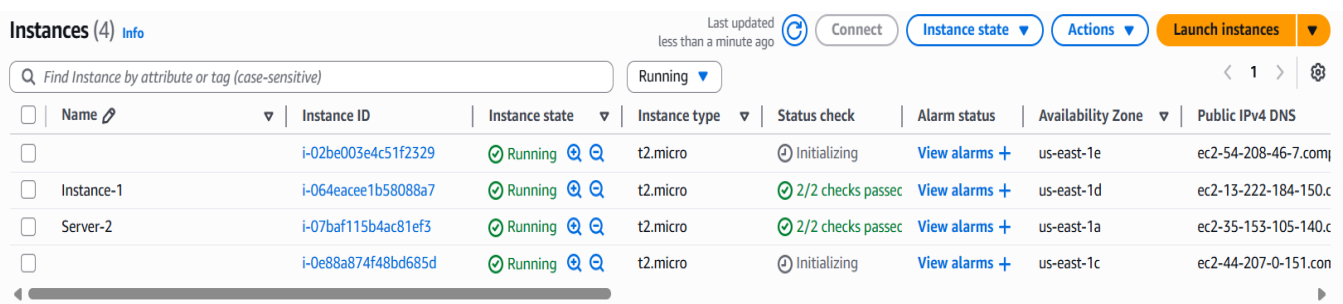
- **Subscribe SNS – Topic via Email:**
- Go to the email (check the email which is shared by aws)
- **Confirm** the subscription from your inbox



- Test the application using the load balancer DNS-ip or Public-IP of your server.(you will see your food-app)
- Now Terminate or install stress on your linux machine.



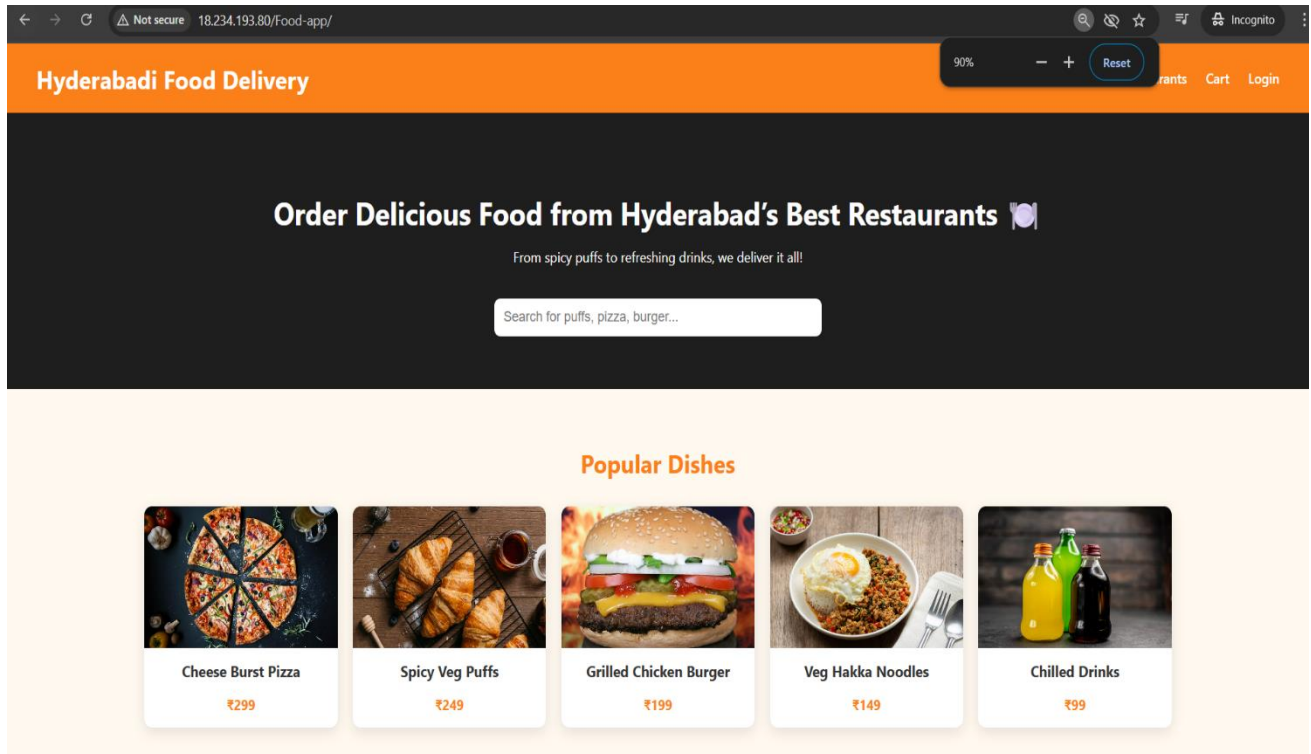
- Now the instances are shutting down, and when our application goes down, the Auto Scaling Group automatically detects the issue and replaces them with new instances.



- As shown in the above image, the Auto Scaling Group launches new instances to ensure high availability.

www.linkedin.com/in/arun-kumar-akula

- Access the application via the Load Balancer DNS or public IP. Even if some EC2 instances are terminated, the Auto Scaling Group ensures the service remains available.



- Whenever our servers go down or get terminated, SNS will send a notification via email.

