

Project

#01

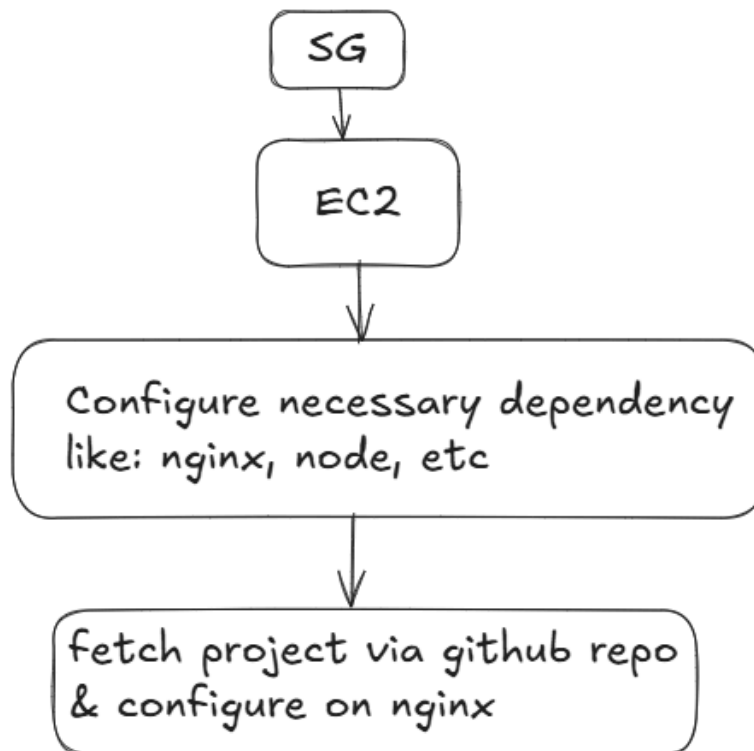
AWS

**Deploy a website on
AWS**

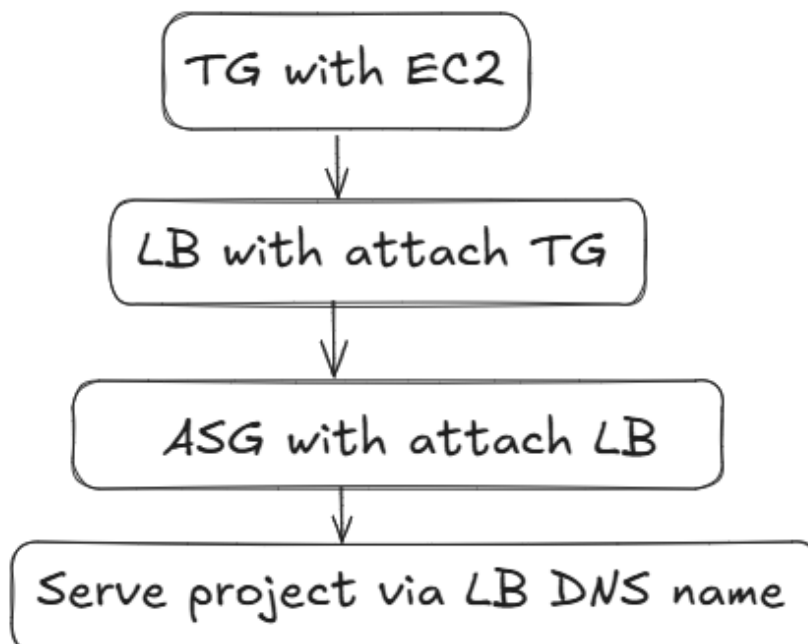


Project Overview:

Step 1: Deploy a Vue js project on AWS EC2.



Step 2: Add load balancer and auto scaling on this project.



Project Overview:

Cover AWS service name lists:

1. SG → Security Groups
2. EC2 → Elastic Compute Cloud
3. TG → Target Groups
4. LB → Load Balancer (use ALB)
5. ASG → Auto Scaling Groups

Configuration Dependency:

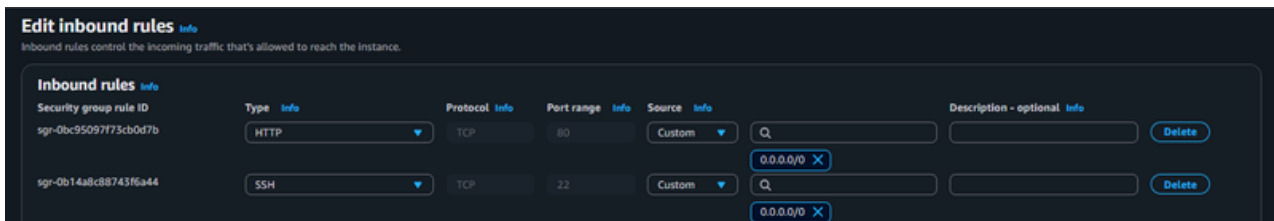
1. Nginx
2. Node.js
3. npm → package manager

Step 2 is not mandatory for deploying a website on AWS EC2.

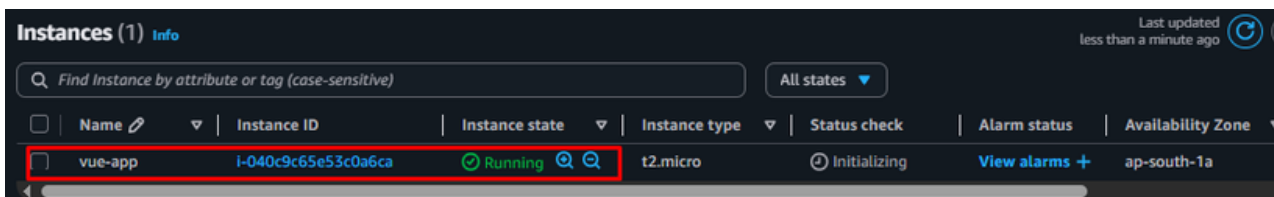
I have previously learned the AWS services mentioned in Step 2. I included Step 2 for my practice purposes. You don't need to follow this if configuring a Load Balancer is not required for your project.

Step 1: Deploy on AWS EC2

First, create a Security Groups (SG) that allows SSH and HTTP access. If you plan to use SSL, allow HTTPS as well. In my case, I didn't configure SSL, so I ignored HTTPS.



Now, launch an EC2 instance using this SG. If you want access through SSH, don't forget to generate a key pair key.



I used the following repository for this project.

<https://github.com/alamgirweb11/meals-app>

Step 1: Deploy on AWS EC2

These are the configuration file I set up.

```
# update packages
```

```
sudo apt update && sudo apt upgrade -y
```

```
# install nginx & check status
```

```
sudo apt install nginx -y
```

```
systemctl status nginx
```

```
# Install Node.js & npm (latest LTS)
```

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

```
# this command installs Node.js & npm both
```

```
sudo apt install -y nodejs build-essential
```

```
# check version
```

```
node -v
```

```
npm -v
```

```
# clone GitHub repo
```

```
git clone https://github.com/alamgirweb11/meals-app.git
```

```
# rename file, for meaningful based on EC2 name
```

```
mv meals-app vue-app
```

```
# locate vue app
```

```
cd vue-app
```

```
# create Environment file for api setup
```

```
nano .env
```

```
# attach this end point and save .env file
```

```
VITE_API_BASE_URL =
```

```
'https://www.themealdb.com/api/json/v1/1/'
```

Step 1: Deploy on AWS EC2

These are the configuration file I set up.

```
# now install npm and build
# --legacy-peer-deps use for ignoring old version package installations
npm install --legacy-peer-deps
npm run build

# configure app in nginx
sudo nano /etc/nginx/sites-available/vue-app

# add config file & save
server {
    listen 80;
    server_name _;

    root /var/www/vue-app/dist;
    index index.html;

    location / {
        try_files $uri $uri/ /index.html;
    }
}

# remove default config file
sudo rm /etc/nginx/sites-enabled/default

# create folder & copy build files
sudo mkdir -p /var/www/vue-app
sudo cp -r ~/vue-app/dist /var/www/vue-app/
sudo cp -r ~/vue-app/dist /var/www/vue-app/

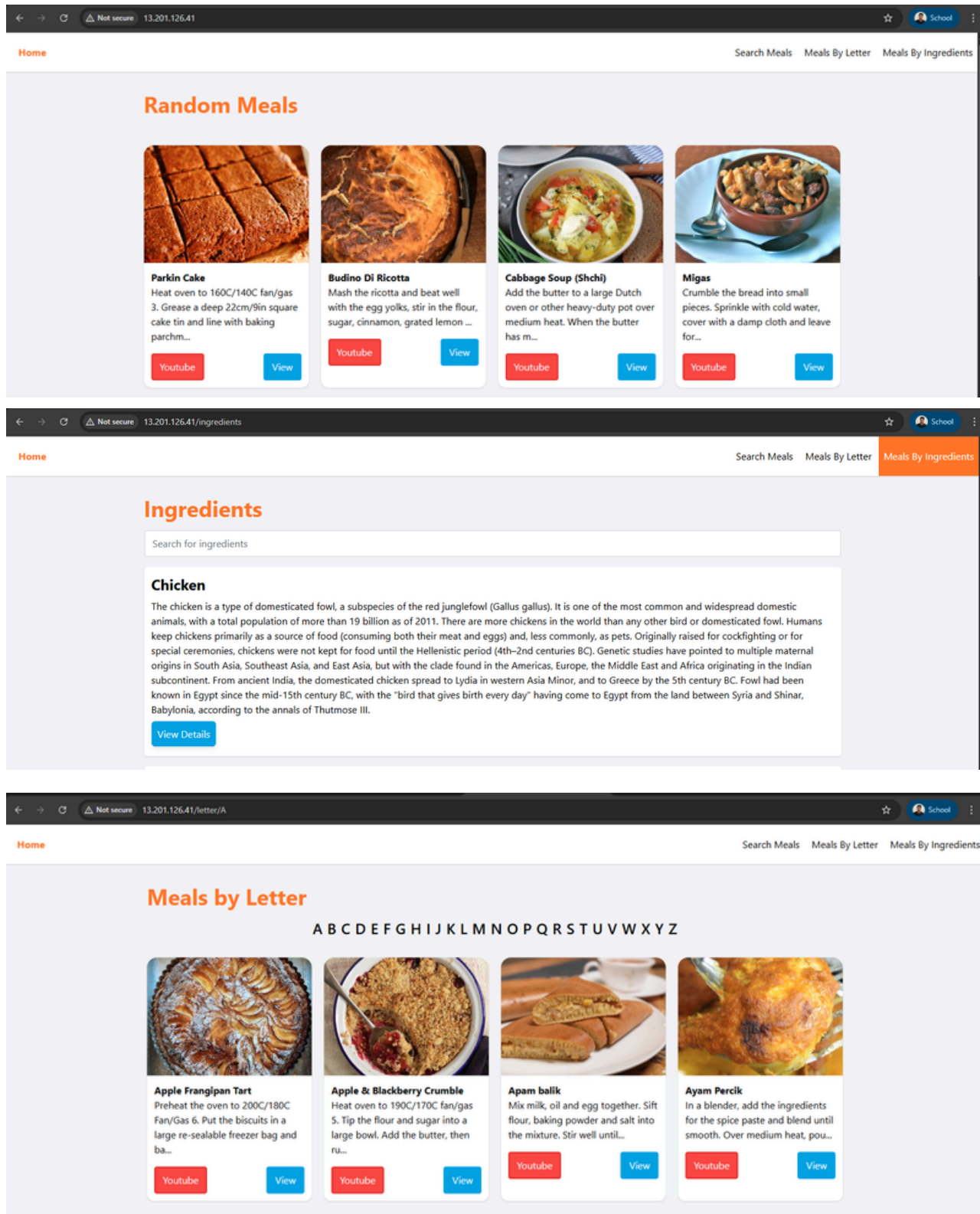
# enable config
sudo ln -s /etc/nginx/sites-available/vue-app /etc/nginx/sites-enabled/

# test nginx config
sudo nginx -t

# restart nginx
sudo systemctl restart nginx
```

Step 1: Deploy on AWS EC2

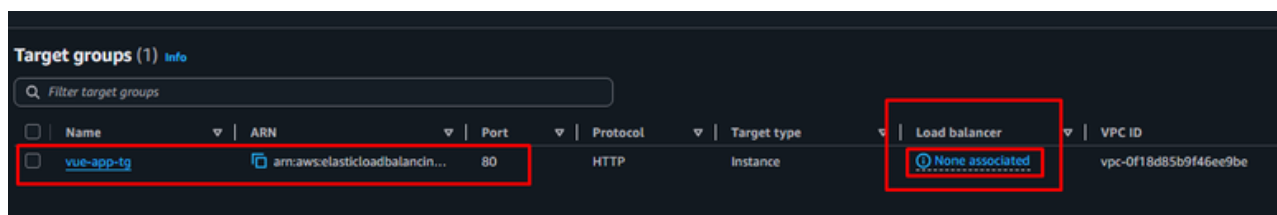
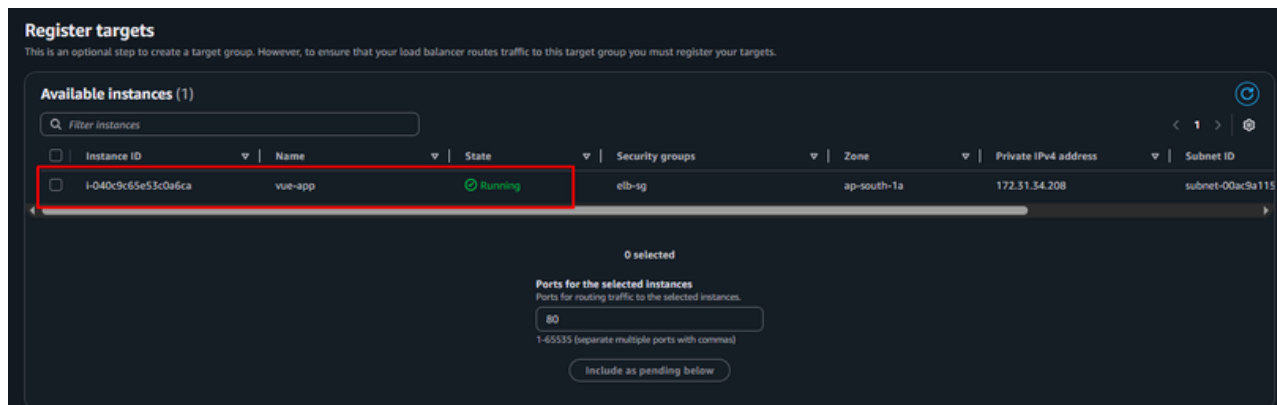
Successfully deploy our site on AWS EC2.



Step 1 Completed

Step 2: Load Blancer & Auto Scaling

For load balance & Auto scaling, first you need to create a Target Groups. Register targets here i select my EC2 instance. Here it is:

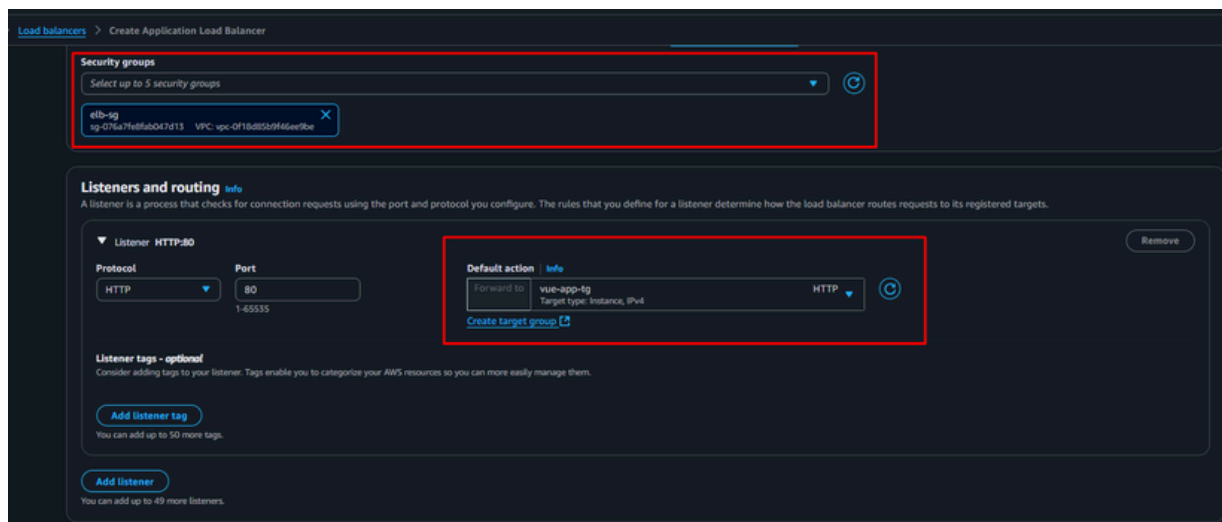


The target groups created successfully, but the load balancer is not yet configured. That's why it shows as not associated with a load balancer.

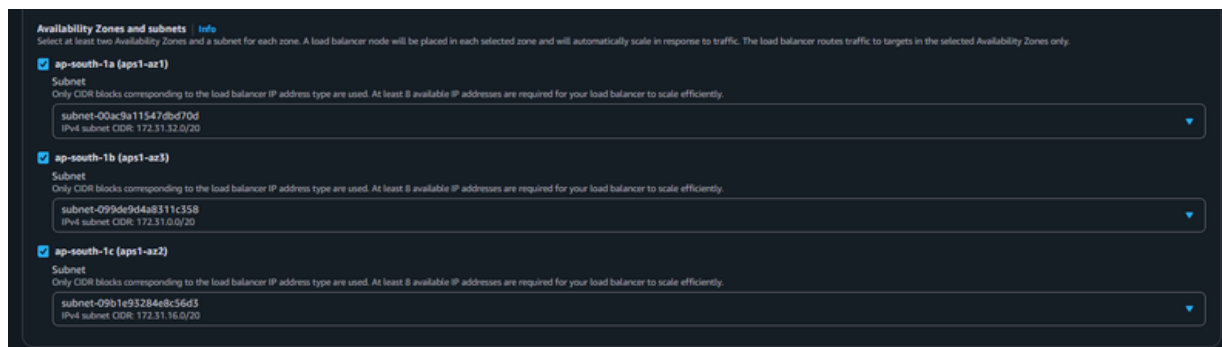
Step 2: Load Blancer & Auto Scaling

Now, created a load balancer with this TG.

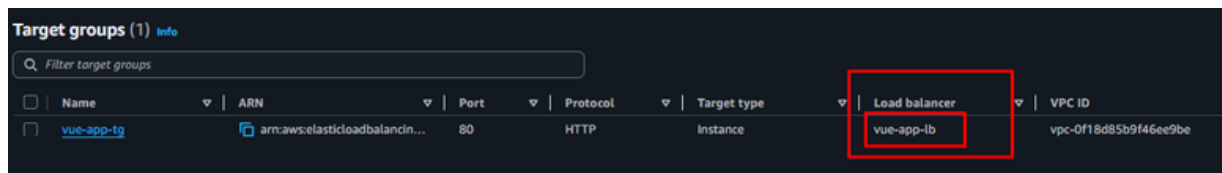
When create a LB please ensure the SG are same as EC2 and select TG which you attach with TG.



Make sure to choose the Availability Zone (AZ).



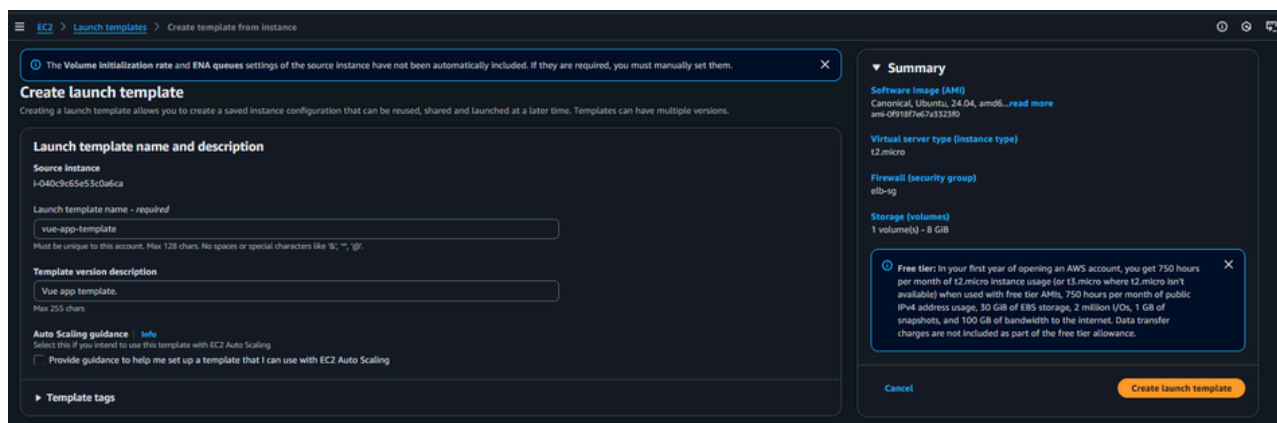
Now go to TG section and see the LB is located.



Step 2: Load Blancer & Auto Scaling

Now it's time to create an Auto Scaling Group (ASG). To do this, go to the Auto Scaling Groups section and configure it based on your needs.

Before creating an ASG, you first need to create a launch template. In my case, I used my EC2 instance as the launch template.



The screenshot shows the 'Create launch template' page in the AWS Management Console. The page is titled 'Create launch template' and includes a warning message at the top: 'The Volume Initialization rate and ENA queues settings of the source instance have not been automatically included. If they are required, you must manually set them.' Below the warning, there is a section for 'Launch template name and description' with a 'Source instance' dropdown set to 'i-040c9e5e53c0a6ca'. The 'Launch template name' field is required and contains 'vue-app-template'. The 'Template version description' field contains 'Vue app template'. There is also an 'Auto Scaling guidance' section with a checkbox to 'Provide guidance to help me set up a template that I can use with EC2 Auto Scaling'. On the right side, there is a 'Summary' section showing details like 'Software image (AMI)', 'Virtual server type (instance type)', 'Firewall (security group)', and 'Storage (volumes)'. At the bottom right, there is a 'Free tier' warning and a 'Create launch template' button.

If you want to use a previous launch template, just choose it from the dropdown. Or, you can create a new one from here, as this option is also provided.



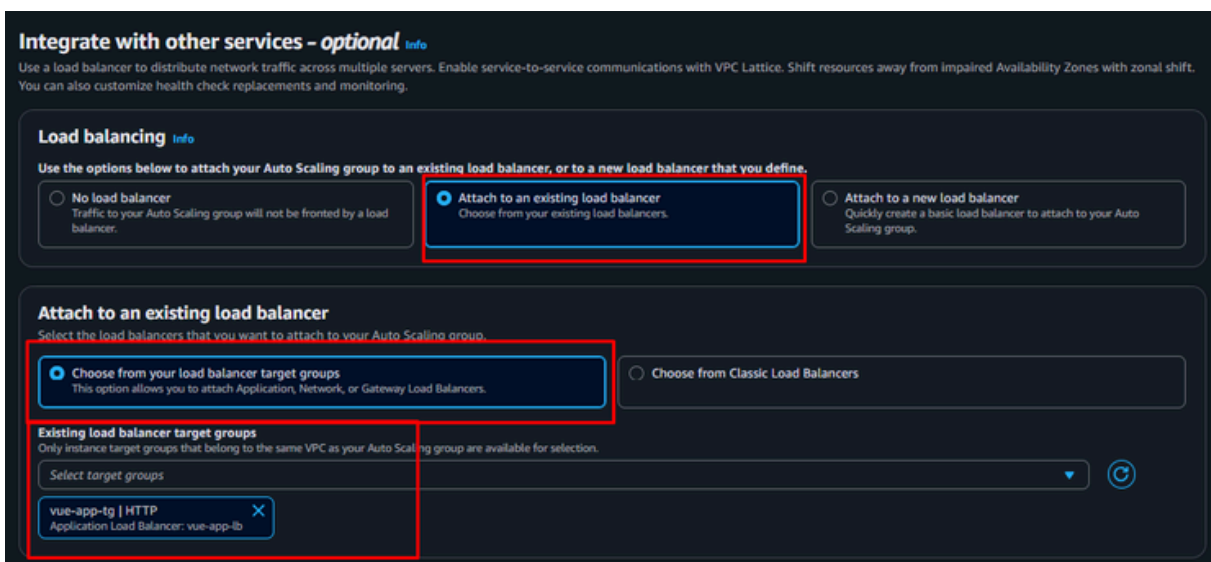
The screenshot shows the 'Choose launch template' page in the AWS Management Console. The page is titled 'Choose launch template' and includes a note: 'Specify a launch template that contains settings common to all EC2 instances that are launched by this Auto Scaling group.' Below the note, there is a 'Name' section with a text input field for 'Auto Scaling group name' containing 'vue-app-asg'. The 'Launch template' section has a dropdown menu labeled 'Select a launch template' and a 'Create a launch template' link. At the bottom right, there are 'Cancel' and 'Next' buttons.

Step 2: Load Blancer & Auto Scaling

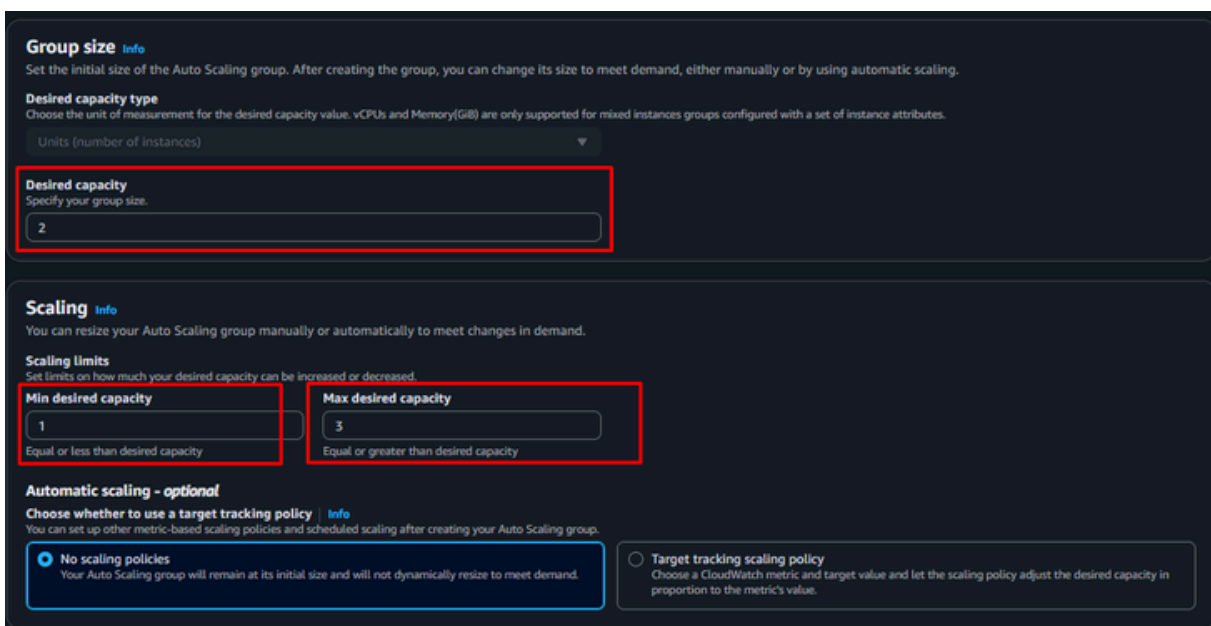
Don't forget to select AZ when creating an ASG.



In attaching load balancer section, don't forget to attach your created LB.

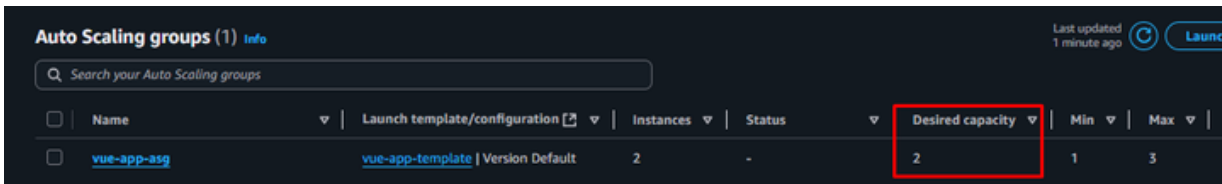


Select desired, min, max capacity it's important.



Step 2: Load Blancer & Auto Scaling

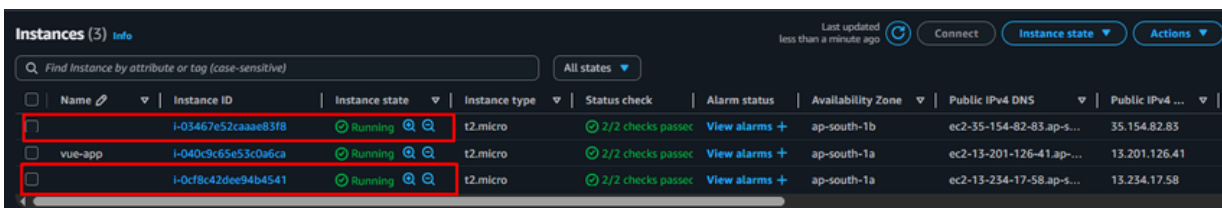
Our ASG successfully created, and instances were launched based on the desired capacity. Since I set it to 2, it launched 2 instances.



The screenshot shows the AWS Auto Scaling groups console. A table lists the 'vue-app-asg' group. The 'Desired capacity' column is highlighted with a red box and shows the value '2'.

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max
vue-app-asg	vue-app-template Version Default	2	-	2	1	3

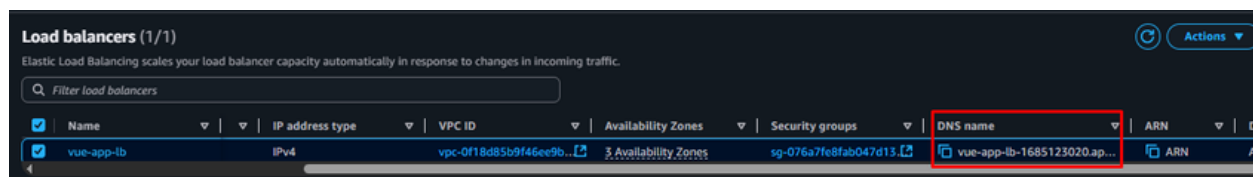
EC2 instances:



The screenshot shows the AWS EC2 instances console. A table lists three running instances, all of which are highlighted with red boxes. The instances are 'vue-app' and two unnamed instances.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...
	i-03467e52caaa83f8	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1b	ec2-35-154-82-83.ap-s...	35.154.82.83
vue-app	i-04dc9c65e53c0a6ca	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1a	ec2-13-201-126-41.ap-...	13.201.126.41
	i-0cf8c42dee94b4541	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1a	ec2-13-234-17-58.ap-s...	13.234.17.58

Here the LB now copy the DNS name and browse.



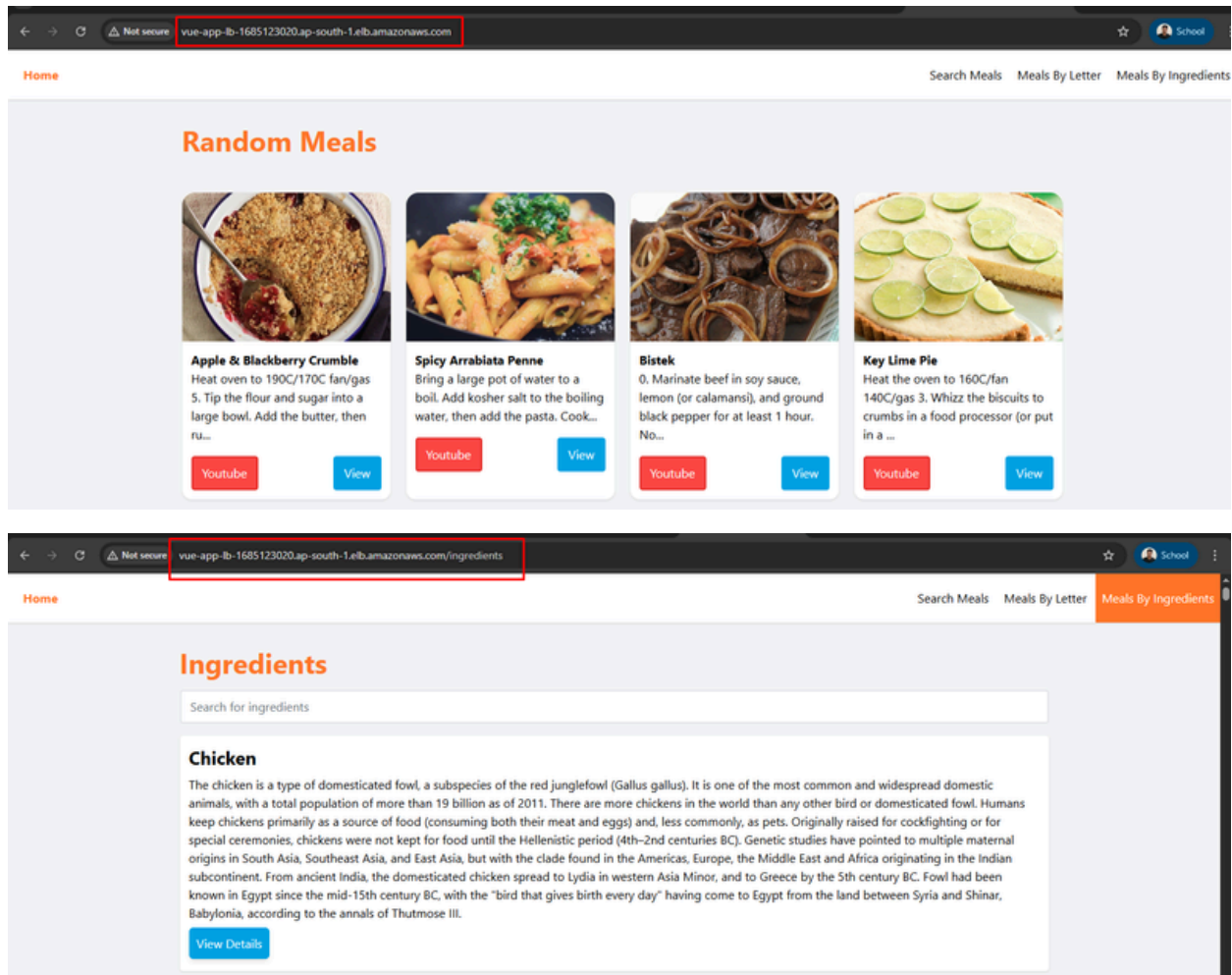
The screenshot shows the AWS Load Balancers console. A table lists the 'vue-app-lb' load balancer. The 'DNS name' column is highlighted with a red box and shows the value 'vue-app-lb-1685123020.ap-...'. The 'ARN' column is also highlighted with a red box.

Name	IP address type	VPC ID	Availability Zones	Security groups	DNS name	ARN
vue-app-lb	IPv4	vpc-0f18d85b9f46ee9b	3 Availability Zones	sg-076a7fe8fab047d13	vue-app-lb-1685123020.ap-...	

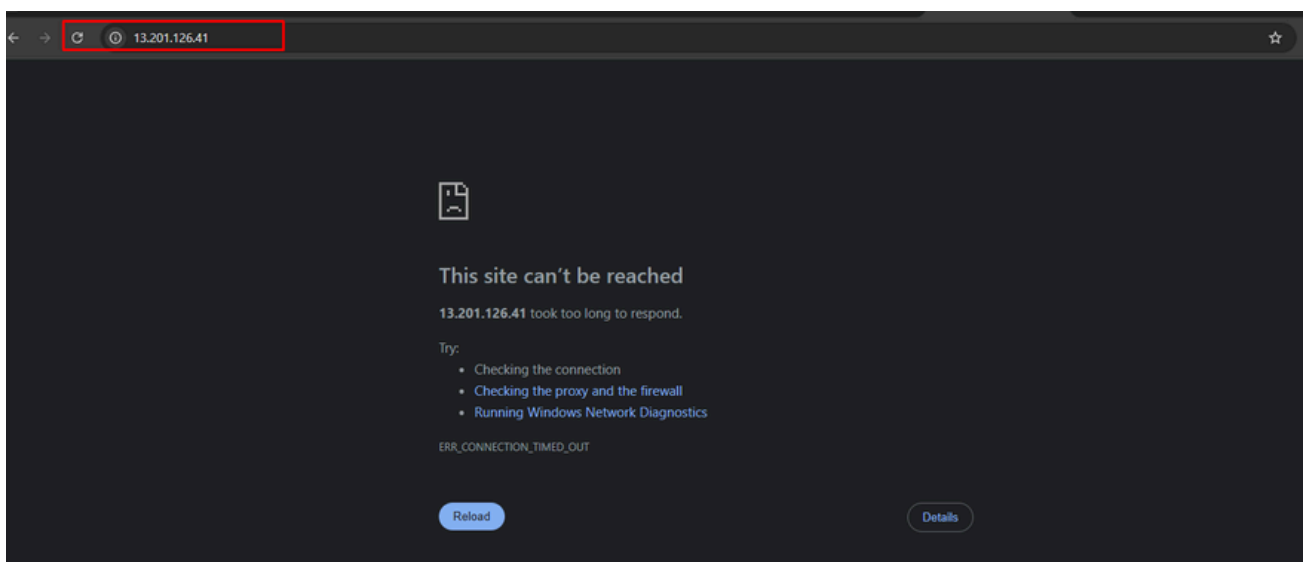
Step 2 Completed

Step 2: Load Blancer & Auto Scaling

Browse the site using load balancer DNS name.



This is the EC2 public IP, but it is not accessible from the browser.



Thank You

Stay Connect:

/in/alamgirweb11

/alamgirweb11