

Deployed a Scalable Web Application on AWS using EC2 Auto Scaling, Load Balancer & CloudWatch Monitoring

STEP 1: Prepare EC2 Instances

You can skip this step if your 3 instances are already set up.

1. Launch 1 EC2 instance

2. Install a web server:

```
sudo apt install apache2 -y
```

```
sudo systemctl enable apache2
```

```
echo "<h1>Server Details</h1>
```

```
<p><strong>Hostname:</strong> $(hostname)</p>
```

```
<p><strong>IP Address:</strong> $(hostname -I | awk '{print $1}')</p>" | sudo  
tee /var/www/html/index.html
```

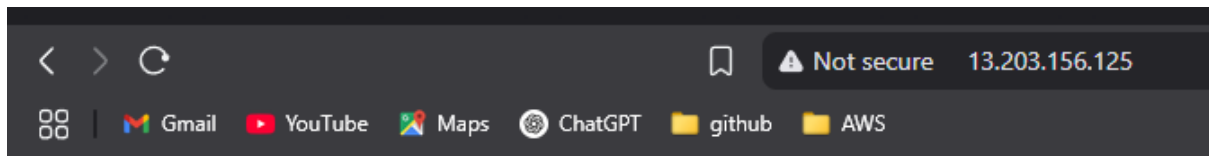
```
sudo systemctl restart apache2
```

3. Create an AMI from this instance (used for Auto Scaling):

- EC2 Dashboard → Select instance → **Actions > Image > Create image**

GNU nano 7.2

```
<h1>Server Details</h1>  
<p><strong>Hostname:</strong> ip-172-31-0-140</p>  
<p><strong>IP Address:</strong> 172.31.0.140</p>  
<h1> EC2 WORKER 03 </h1>
```



Server Details

Hostname: ip-172-31-0-140

IP Address: 172.31.0.140

EC2 WORKER 03

STEP 2: Create a Launch Template

This defines how new EC2 instances should launch.

1. Go to **EC2 > Launch Templates > Create Launch Template**
2. Name: web-server-template
3. AMI: Select the one you just created
4. Instance Type: t2.micro (Free Tier)
5. Key pair: Choose existing or create new
6. Security Group: Allow HTTP (80) + SSH (22)
7. Save

> web-server-template ⓘ ⌵ 🖨

web-server-template (lt-0c689879c7029c014) Actions ▾ Delete template

Launch template details

| | | | |
|---|--|-----------------------------|--|
| Launch template ID lt-0c689879c7029c014 | Launch template name web-server-template | Default version 1 | Owner arn:aws:iam::317471784525:root |
|---|--|-----------------------------|--|

Details Versions Template tags

Launch template version details Actions ▾ Delete template version

Version
1 (Default) ▾

Description
web-server-template

Date created
2025-06-20T15:19:16.000Z

Created by
arn:aws:iam::317471784525:root

Instance details Storage Resource tags Network interfaces Advanced details

| | | | |
|--|---|-------------------------------|-------------------------------------|
| AMI ID ami-0f918f7e67a3323f0 | Instance type t2.micro | Availability Zone - | Key pair name bisw-test01 |
| Security groups - | Security group IDs sg-072f254b4b6456048 | | |

STEP 3: Create Target Group

This allows Load Balancer to route traffic.

1. Go to **EC2 > Target Groups > Create Target Group**
2. Type: Instances
3. Protocol: HTTP, Port: 80
4. Name: web-target-group
5. Register one existing instance for test
6. Save

web-target-group

web-target-group Actions

Details

arn:aws:elasticloadbalancing:ap-south-1:317471784525:targetgroup/web-target-group/151fa45593fba73e

| | | | |
|--------------------------------|---|----------------------------------|---|
| Target type Instance | Protocol : Port HTTP: 80 | Protocol version HTTP1 | VPC vpc-08abfeff183ad7b9f |
| IP address type IPv4 | Load balancer None associated | | |

| | | | | | |
|--------------------|--------------|----------------|-------------|--------------|---------------|
| 3 Total targets | 0 Healthy | 0 Unhealthy | 3 Unused | 0 Initial | 0 Draining |
|--------------------|--------------|----------------|-------------|--------------|---------------|

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 (3) Info [Anomaly mitigation: Not applicable](#) [Deregister](#) [Register targets](#)

Target groups route requests to individual registered targets using the protocol and port number specified. Health checks are performed on all registered targets according to the target group's health check settings. Anomaly detection is automatically applied to HTTP/HTTPS target groups with at least 3 healthy targets.

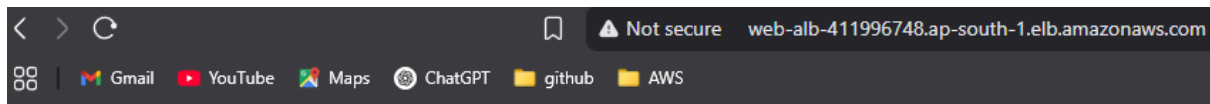
Filter targets

| <input type="checkbox"/> | Instance ID | Name | Port | Zone | Health status | Health status details | Admini... | Overrid... | Launch... | Anomaly detection... |
|--------------------------|-------------------------------------|--------|------|-------------------|---------------|---------------------------|-----------|------------|---------------|----------------------|
| <input type="checkbox"/> | i-002db845ba8400675 | EC2-03 | 80 | ap-south-1b (a... | Unused | Target group is not co... | - | - | June 20, 2... | Normal |
| <input type="checkbox"/> | i-0d18284cebb3dc893 | EC2-02 | 80 | ap-south-1b (a... | Unused | Target group is not co... | - | - | June 20, 2... | Normal |
| <input type="checkbox"/> | i-045ebd1820f45b715 | EC2-01 | 80 | ap-south-1b (a... | Unused | Target group is not co... | - | - | June 20, 2... | Normal |

STEP 4: Create Application Load Balancer

1. Go to **EC2 > Load Balancers > Create Load Balancer**
2. Choose **Application Load Balancer**
3. Name: web-alb
4. Scheme: Internet-facing
5. Listeners: HTTP (80)
6. Availability Zones: Choose 2+ subnets
7. Target Group: Use web-target-group

After creation, test the **ALB DNS** to see your webpage.

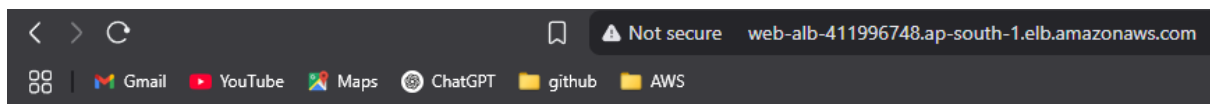


Server Details

Hostname: ip-172-31-13-161

IP Address: 172.31.13.161

EC2 WORKER 02

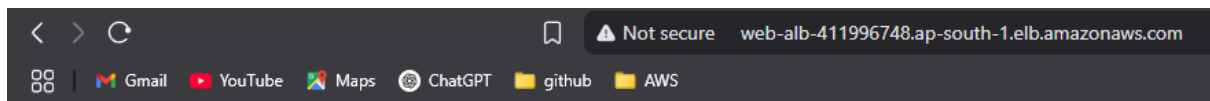


Server Details

Hostname: ip-172-31-0-140

IP Address: 172.31.0.140

EC2 WORKER 03



Server Details

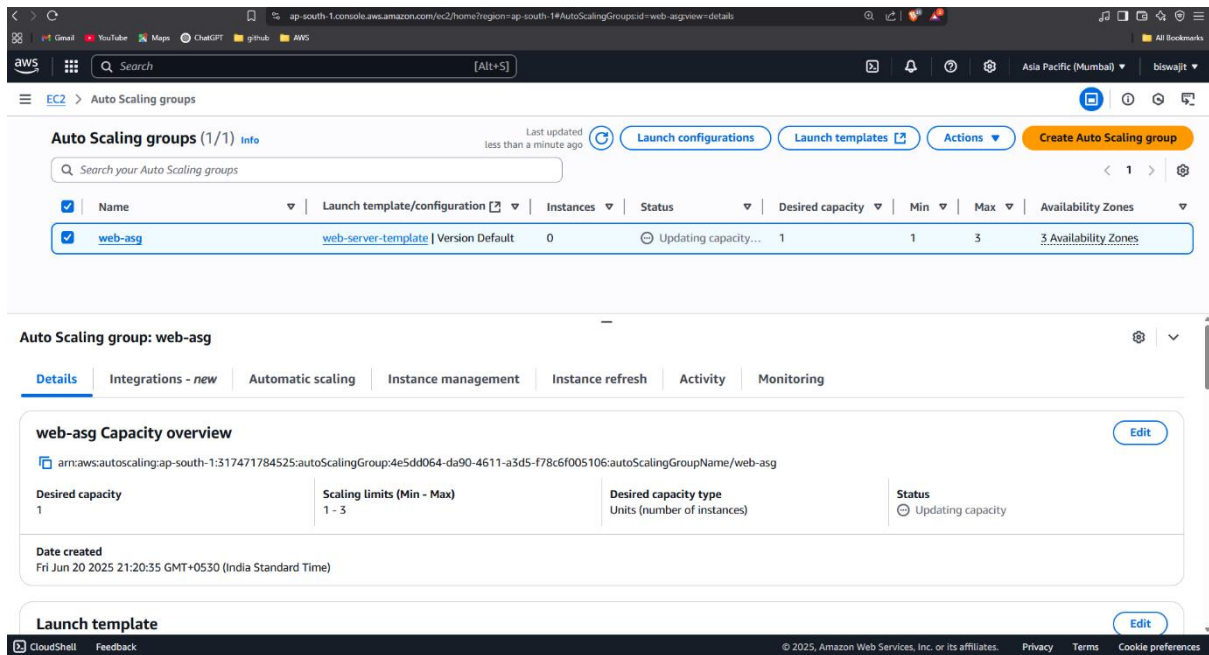
Hostname: ip-172-31-0-39

IP Address: 172.31.0.39

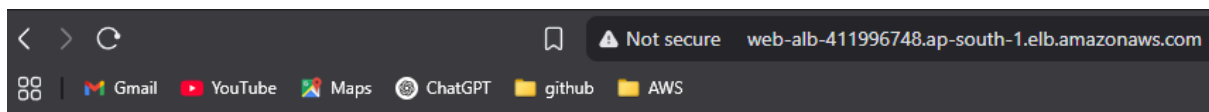
EC2 WORKER 01

STEP 5: Create Auto Scaling Group (ASG)

1. Go to **EC2 > Auto Scaling Groups > Create**
2. Name: web-asg
3. Launch Template: web-server-template
4. VPC & Subnets: Select existing
5. Attach to Load Balancer: Yes → Choose your target group
6. Group Size: Desired=2, Min=1, Max=3
7. Scaling Policy: Target tracking
 - Metric type: Average CPU Utilization
 - Target: 50%



After generating some load, Auto Scaling automatically added one instance



Server Details

Hostname: ip-172-31-9-196

IP Address: 172.31.9.196

STEP 6: Set Up CloudWatch Alarms

1. Go to **CloudWatch > Alarms > Create Alarm**
2. Metric: EC2 → Per-Instance Metrics → CPUUtilization
3. Condition: CPU > 70% for 2 periods (1 min each)
4. Action: Trigger Auto Scaling (optional)
5. Set Notification (optional): Use SNS to get email alerts

Alarms (1)

☐ Hide Auto Scaling alarms

Alarm state: In alarm

Alarm type: Any

Actions status: Any

< 1 >

| <input type="checkbox"/> | Name | State | Last state update (UTC) | Conditions | Actions |
|--------------------------|--|----------|-------------------------|---|-----------------|
| <input type="checkbox"/> | TargetTracking-web-asg-AlarmLow-82bcc34d-c3a6-477f-b977-d7a2d968c793 | In alarm | 2025-06-20 16:05:01 | CPUUtilization < 35 for 15 datapoints within 15 minutes | Actions enabled |

Then wait a few minutes → the alarm will go **"In Alarm"**, and you'll get an email notification.

Search mail

1 of 1,325

ALARM: "Alarm ec2-70%" in Asia Pacific (Mumbai)

Inbox x

AWS Notifications

<no-reply@sns.amazonaws.com>

to me

21:52 (0 minutes ago)

You are receiving this email because your Amazon CloudWatch Alarm "Alarm ec2-70%" in the Asia Pacific (Mumbai) region has entered the ALARM state, because "Threshold Crossed: 1 out of the last 1 datapoints [98.72582277689395 (20/06/25 16:20:00)] was greater than the threshold (70.0) (minimum 1 datapoint for OK -> ALARM transition)," at "Friday 20 June, 2025 16:22:13 UTC".

View this alarm in the AWS Management Console:
<https://ap-south-1.console.aws.amazon.com/cloudwatch/deeplink.js?region=ap-south-1#alarmsV2:alarm/Alarm%20ec2-70%25>

Alarm Details:

- Name: Alarm ec2-70%
- Description:
- State Change: OK -> ALARM
- Reason for State Change: Threshold Crossed: 1 out of the last 1 datapoints [98.72582277689395 (20/06/25 16:20:00)] was greater than the threshold (70.0) (minimum 1 datapoint for OK -> ALARM transition).
- Timestamp: Friday 20 June, 2025 16:22:13 UTC
- AWS Account: 317471784525
- Alarm Arn: arn:aws:cloudwatch:ap-south-1:317471784525:alarm:Alarm ec2-70%

Threshold:

- The alarm is in the ALARM state when the metric is GreaterThanThreshold 70.0 for at least 1 of the last 1 period(s) of 60 seconds.

Monitored Metric:

- MetricNamespace: AWS/EC2
- MetricName: CPUUtilization
- Dimensions: [InstanceId = i-002db845ba8400675]
- Period: 60 seconds
- Statistic: Average
- Unit: not specified
- TreatMissingData: missing