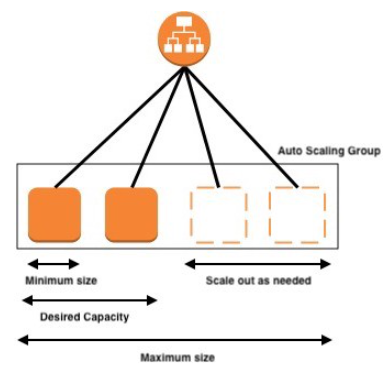
**Auto Scaling**

• Amazon EC2 Auto Scaling helps you maintain application availability and allows you to automatically add or remove EC2 instances according to conditions you define.



• Steps we are following in Auto Scaling

• Create Launch configuration / Launch Template

• Create a Topic in SNS (Simple Notification Service)

• Create Autoscaling group

• Create Application Load Balancer

• Create Target Group

• Create an Alarm in Cloud Watch

• Add Policy in Auto Scaling

**Create Launch Configuration**

• A launch configuration is an instance configuration template that an Auto Scaling group uses to launch EC2 instances.



Click on Create Launch Configuration

• Enter the name.

• Select Amazon Machine Image (AMI) (ami-08e0ca9924195beba)

**• Select Instance type**

• Click on Choose Instance type

• Search instance type (t2.micro)

• Select our instance & Click on Choose

• Click on Advanced Option

• Add Bootstrap script Code:

#!/bin/bash

sudo su

yum update -y

yum install httpd -y

cd /var/www/html

echo "AutoScallingCode" > index.html

service httpd start

chkconfig httpd on

• Select the existing Security Group or create a new security group

• Select the existing Key Pair or create a new key pair

• Click on Create Launch Configuration

**• Create a Topic in SNS (Simple Notification Service)**

• Search Simple Notification Service

• Enter the Name

• Click on Next Step

• Select type as Standard & Enter Display name

• Click on Create Topic

**Click on Create Subscription**

• Select protocol as Email

• Enter endpoint (Contact Email Address)

• Click on Create Subscription

• Now check our email address & Confirm

• Now check the status of SNS

**• Create Auto Scaling Group**

• An AWS Auto Scaling group (ASG) is a fleet of EC2 instances that can scale up or down depending on application demand.

• Go to Launch Configuration



• Select the Launch Configuration



• Enter the name of Auto Scaling

• Click on Next

• Select subnet

• Click on Next

• Select attach to a new load balancer (Create Application Load Balancer)

• Select load balancer schema as internet-facing

• Select default routing as create a target group

• Click on Next

• Enter the Group Size

• Click on Next

• Click on Add Notification

• Select SNS Topic

• Click on Next

• Click on Add Tag

• Enter the Key & Value

• Click on Next

• Click on Create Auto Scaling Group

**• Create an Alarm in Cloud Watch**

• Amazon CloudWatch is a monitoring and management service that provides data and actionable insights for AWS, hybrid, and on-premises applications and infrastructure resources

• Select CloudWatch

• Go to Alarms

• Click on In alarm

• Click on Create Alarm

• Click on Select metric

• Select EC2

• Select By Auto Scaling Group

• Select CPUUtilization as Metric Name.

• Click on Select metric

• Select the Condition & Enter the value. (Greater than 80)

• Click on Next

• Select the Topic

• Click on Next

• Enter Alarm Name & Description

• Click on Next

• Review & Click on Create Alarm

• Create one More alarm

• Select the Condition & Enter the value. (Lower than 30)

**• Add Policy in Auto Scaling**

• Go to Auto Scaling Group

• Click on the Name of our Auto Scaling Group

• Go to the Automatic Scaling tab

• Click on Create Dynamic Scaling Policy

• Change Policy Type to Simple Scaling

• Enter the name of the Policy

• Select the CloudWatch Alarm

• Enter the Capacity Units. (How Many Machine Get Added While Reaching our Alarm Limit)

• Click on Create

• Like the same we need to create one more policy when the Load will be less on our machines then the system will automatically delete those EC2 Machines

**Auto Scaling - Deleting Process**

• Delete Autoscaling

• Delete launch configuration (Instances will be terminated automatically)

• Delete Load balancer

• Delete Target Group

• Delete Topic

• Delete Alarm

**Amazon Simple Queue Service (SQS)**

Fully managed message queuing for microservices, distributed systems, and serverless applications.

First-in-first-out (FIFO) queues help make sure the messages you send to systems are published in the correct order.

There are only two types of queues in SQS:

1. First-in-first-out (FIFO): First-in-first-out (FIFO) queue the order of messages is preserved.

2. Standard queue: Standard queue messages are delivered randomly.

**AWS SQS VS SNS**

| **AWS SQS** | **AWS SNS** |
| --- | --- |
| SQS is an message queuing service. | [SNS](https://www.geeksforgeeks.org/simple-notification-service-sns-in-aws/) is a publish/subscribe service. |
| It will transmit the data between distributed components. | It will distribute messages to multiple subscribers with the help of email, HTTP, and [AWS Lambda](https://www.geeksforgeeks.org/introduction-to-aws-lambda/). |
| Used when you need to decouple the components. | Used to broadcast the data to all the subscribers. |

**Amazon Simple Email Service (SES)**

Cloud-based email platform that provides an easy, cost-effective way for you to send and receive email using your email addresses and domains.

SES allows integration with SMTP (Simple Mail Transfer Protocol), enhancing email delivery capabilities and ensuring reliable communication.

**Lambda (Serverless Compute on Amazon Web Services)**

Run code without thinking about servers or clusters

AWS Lambda is an event-driven, serverless Function as a Service provided by Amazon as a part of Amazon Web Services.

It enables developers to run code without provisioning or managing servers.

It executes code in response to events and automatically manages the computing resources required by that code.

**AWS Elastic Beanstalk (Orchestration service by AWS)**

Elastic Beanstalk is a Platform as A Service (PAAS)

For deploying applications orchestrating various AWS services, including [EC2](https://en.wikipedia.org/wiki/Amazon_EC2), [S3](https://en.wikipedia.org/wiki/Amazon_S3), SNS, [CloudWatch](https://en.wikipedia.org/wiki/Amazon_CloudWatch), [autoscaling](https://en.wikipedia.org/wiki/Autoscaling), and [Elastic Load Balancers](https://en.wikipedia.org/wiki/Elastic_Load_Balancing).

You upload your application, and Elastic Beanstalk automatically handles the details of capacity provisioning, load balancing, scaling, and application health monitoring.

Elastic Beanstalk supports applications developed in Go, Java, .NET, Node.js, PHP, Python, and Ruby.

Elastic Beanstalk also supports Docker platforms.