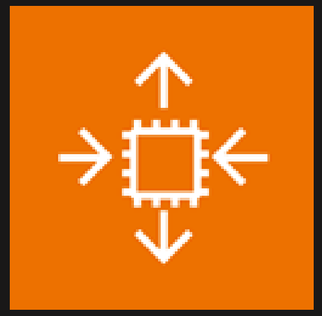
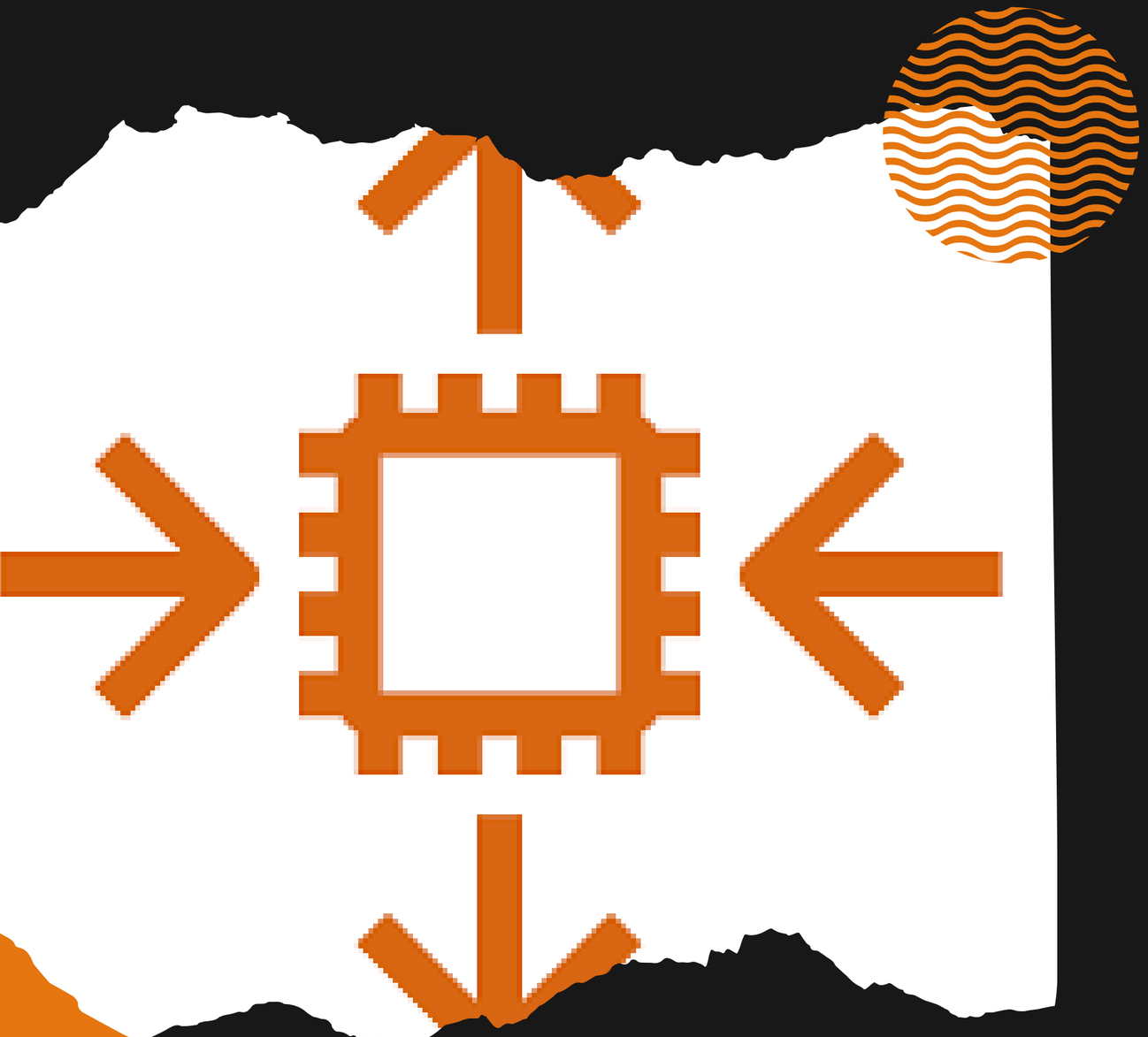


AWS

AUTO SCALING

Content Table

01 ~ Introduction

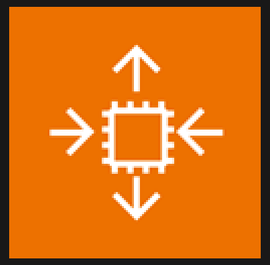


Intro To Auto Scaling Groups

ASG helps you maintain application availability and allows you to automatically add or remove EC2 instances according to conditions you define. You can use Amazon EC2 Auto Scaling to ensure that you have the right number of Amazon EC2 instances available to handle the load for your application.

LEARN MORE

02 ~ Key Concepts



Key Concepts

- Auto Scaling Group (ASG)
- Launch Configuration/Launch Template
- Scaling Policies
- Lifecycle Hooks
- Health Checks

Key Concepts

Auto Scaling Group (ASG)

Contains a collection of EC2 instances that are treated as a logical grouping for the purposes of automatic scaling and management. You can specify the minimum, maximum, and desired number of instances in your ASG.

Launch Configuration/Launch Template

an instance configuration template that an Auto Scaling group uses to launch EC2 instances. It includes information such as the AMI ID, instance type, key pair, security groups, and block device mapping.

Key Concepts

Scaling Policies

Scaling policies define how the ASG should scale in response to changes in demand. These can be simple scaling policies, step scaling policies, or target tracking scaling policies.

Lifecycle Hooks

Lifecycle hooks enable you to perform custom actions as the Auto Scaling group launches or terminates instances. These actions can include downloading software updates, installing applications, or gracefully shutting down services.

Key Concepts

Health Checks

Health checks ensure that the instances within your ASG are running properly. Amazon EC2 Auto Scaling integrates with Elastic Load Balancing (ELB) health checks, EC2 instance status checks, and custom health checks.

03 ~ Use Cases

Scenario 01

An e-commerce website experiences fluctuating traffic patterns, with spikes during sales events and holidays.

Solution

Use ASG to automatically scale out additional instances during high traffic periods to maintain performance and scale in during low traffic periods to reduce costs.

Benefits

Ensures high availability and optimal performance while controlling costs

Scenario 02

A company processes large volumes of data in batch jobs that run periodically.

Solution

Configure ASG to launch instances only when batch processing jobs are scheduled and terminate them once the jobs are completed.

Benefits

Efficient resource utilization and cost savings by only running instances when needed.

Scenario 03

A mission-critical application needs to be highly available and resilient to failures.

Solution

Deploy instances across multiple Availability Zones (AZs) using ASG to ensure redundancy and failover capabilities.

Benefits

Enhances fault tolerance and ensures continuous availability even in case of an AZ failure

Scenario 04

An online multiplayer game experiences varying player loads throughout the day.

Solution

Implement ASG to automatically scale gaming servers up or down based on the number of active players.

Benefits

Ensures smooth gameplay experience during peak times and reduces costs during off-peak hours.

Scenario 05

Development and testing environments need to replicate production workloads for accurate testing.

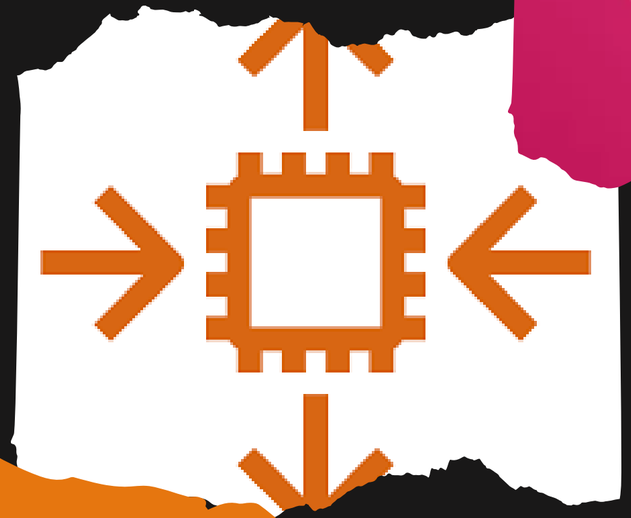
Solution

Use ASG to automatically provision instances in dev/test environments based on the testing schedule and terminate them after tests are completed.

Benefits

Provides a cost-effective way to run development and testing environments by scaling resources up and down as needed.

04 ~ Monitoring & Management



Monitoring in ASG

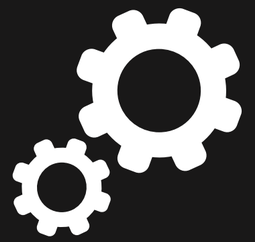
01 CloudWatch Metrics

Monitor ASG metrics such as group size, desired capacity, in-service instances, and pending instances.

02 CloudWatch Alarms

Set up alarms to trigger notifications or scaling actions based on specific metrics.

Management in ASG



Instance Management:

- View and manage instances within the ASG.
- Perform instance refreshes to update instances with the latest configuration.

Lifecycle Management:

- Use lifecycle hooks to perform custom actions during instance launch and termination.

05 ~ Best Practices

Best Practices

Use Multiple AZs:

1. Distribute instances across multiple Availability Zones to improve availability.

Implement Health Checks:

2. Use ELB health checks for better integration and reliability.

Optimize Scaling Policies:

3. Regularly review and adjust scaling policies based on application performance and usage patterns.

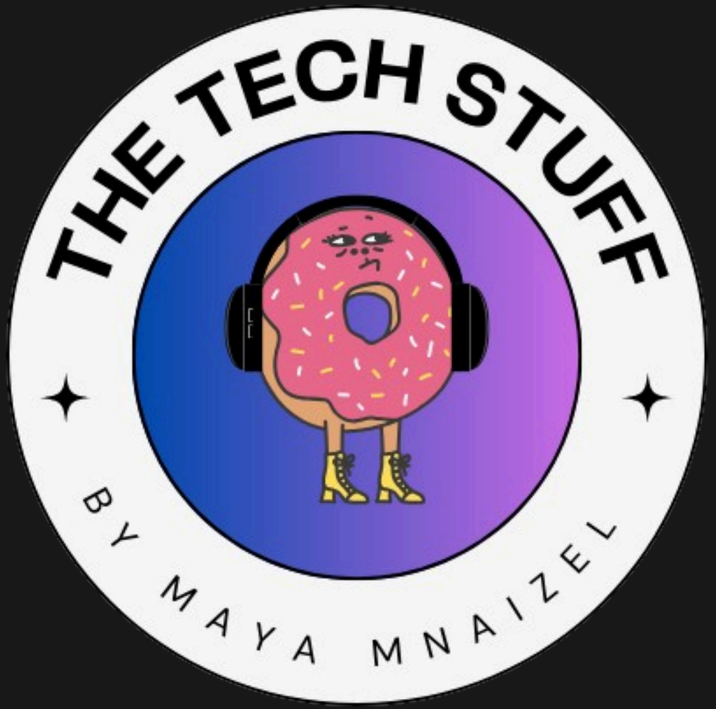
Monitor Costs:

4. Keep an eye on costs associated with scaling and adjust instance types and purchasing options accordingly.

Use Auto Scaling with ELB:

5. Combine ASG with Elastic Load Balancing to distribute traffic evenly across instances.

Thank you!
Any Questions?



Youtube: The Tech Stuff

Email : mayamnaizel2013@gmail.com