NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Principles in Cloud Architecture Design - Reliability

Cloud Infrastructure Engineering

Nanyang Technological University
& Skills Union - 2022/2023

Course Content

- Quick Check-In
- Dive into what Reliability means and the best practices in terms of Reliability
- Build architectures that have strong foundations, resilient architecture, consistent change management, and proven failure recovery processes

Time	What	How or Why
7:15pm - 7:40pm	Part 1 - Presentation	Reliability Design Principles
7:40pm - 8:00pm	Part 2 - Activity	
8:00pm - 8:10pm	Break	
8:10pm - 8:30pm	Part 3 - Presentation	Best Practices for Reliability
8:30pm - 8:50pm	Part 4 - Activity	
8:50pm - 10:00pm	Summary & Assignments	

Recap

- Security Design Principles
  - Strong Identity Foundation, Enable Traceability, Security At All Layers, Automate Security Best Practices, Protect Data In-Transit & At-Rest, Keep People Away From Data & Prepare For Security Events
- Best Practices
  - Security, IAM, Detection, Infrastructure Protection, Data Protection & Incident Response

Self Study Check-In

Q1) How would you enable logging for your application?

Q2) What would you do in order to enable better application reliability?

Overview

Overview

This module focuses on the **reliability** pillar and how to apply it to your solutions.

Achieving reliability can be **challenging in traditional on-premises** environments due to single points of failure, lack of automation, and lack of elasticity.

By adopting the practices in this module you will **build architectures that have strong foundations, resilient architecture, consistent change management, and proven failure recovery processes.**

Overview

This module is intended for those in technology roles, such as chief technology officers (CTOs), architects, developers, and operations team members.

After this, you will understand **AWS best practices and strategies to use when designing cloud architectures for reliability**.

This includes high-level implementation details and architectural patterns, as well as references to additional resources.

Reliability Principles

Reliability Principles Summary

- Automatically recover from failure
- Test recovery procedures
- Scale horizontally to increase aggregate workload availability
- Stop guessing capacity
- Manage change in automation

Automatically Recover From Failures

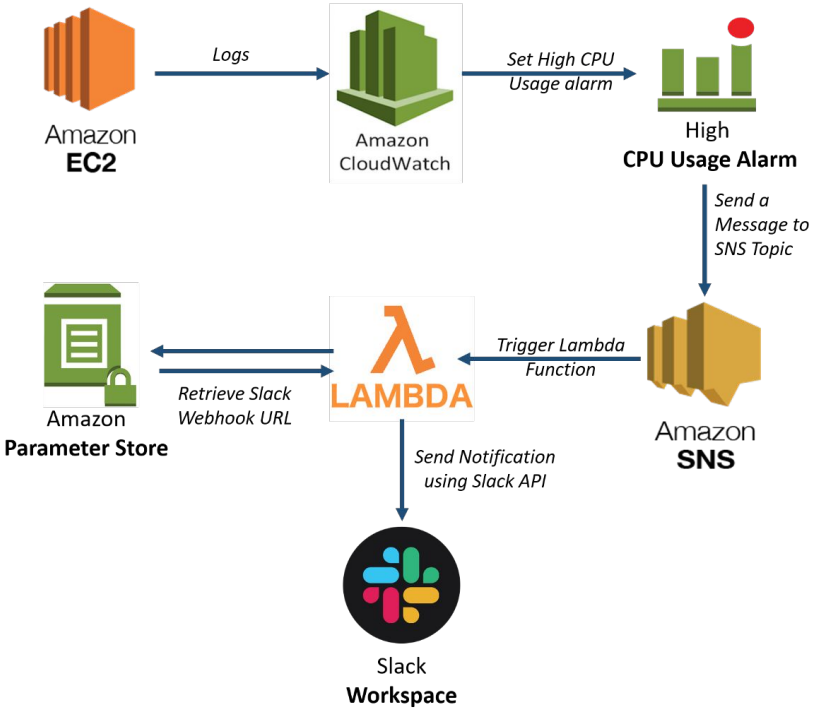
By monitoring a workload for key performance indicators (KPIs), you can trigger automation when a **threshold** is breached.

These KPIs should be a **measure of business value**, not of the technical aspects of the operation of the service.

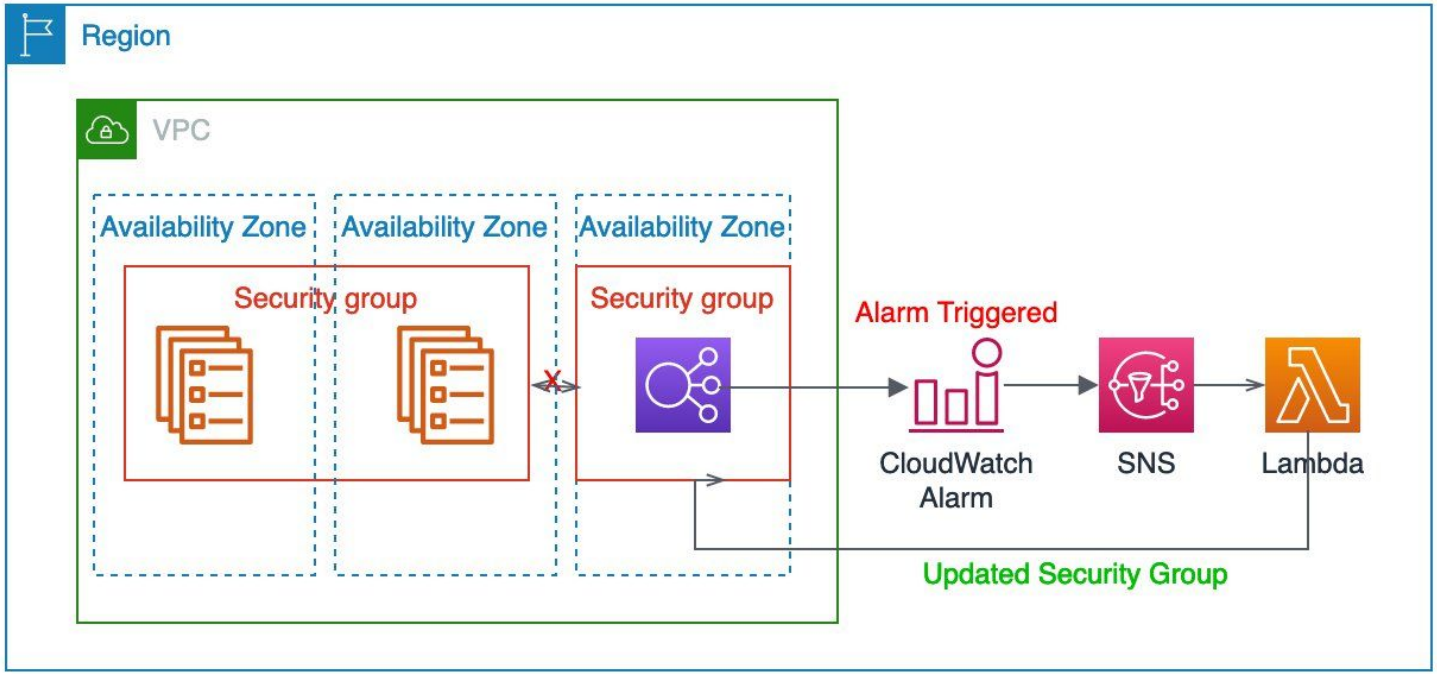
This allows for **automatic notification and tracking of failures**, and for automated recovery processes that work around or repair the failure.

With more sophisticated automation, it's possible to **anticipate and remediate failures** before they occur.

Automatically Recover From Failures



Automatically Recover From Failures



Test Recovery Procedures

In an on-premises environment, testing is often conducted to prove that the workload works in a particular scenario.

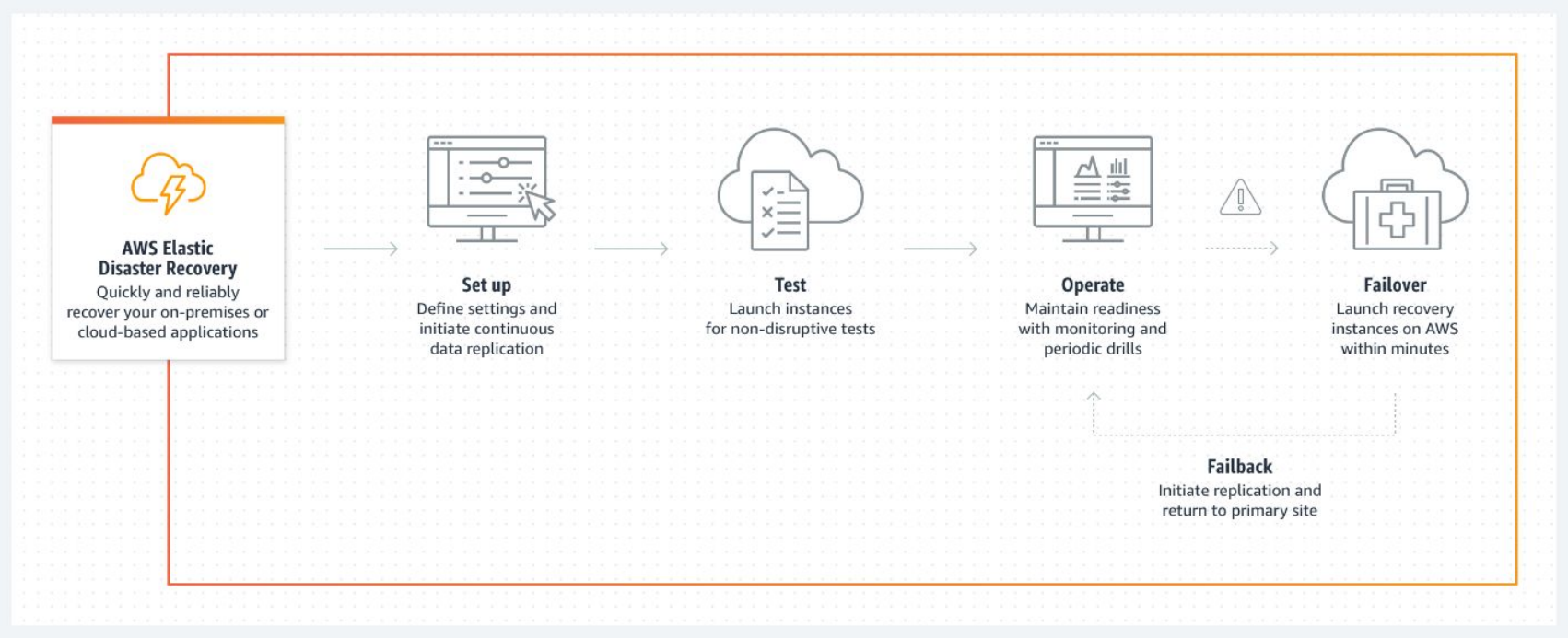
Testing is not typically used to validate recovery strategies.

In the cloud, you can **test how your workload fails, and you can validate your recovery procedures.**

You can use **automation to simulate different failures** or to r**ecreate scenarios that led to failures before.**

This approach exposes failure pathways that you can test and fix before a real failure scenario occurs, thus reducing risk.

Test Recovery Procedures

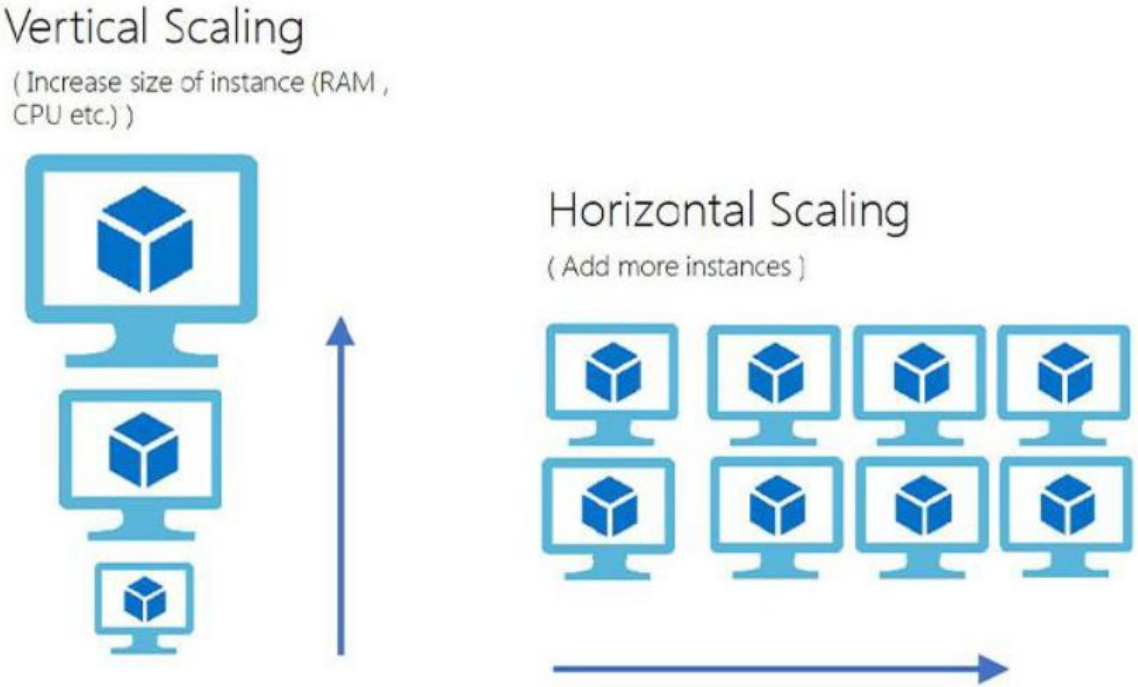


Scale Horizontally to Increase Aggregate Workload Availability

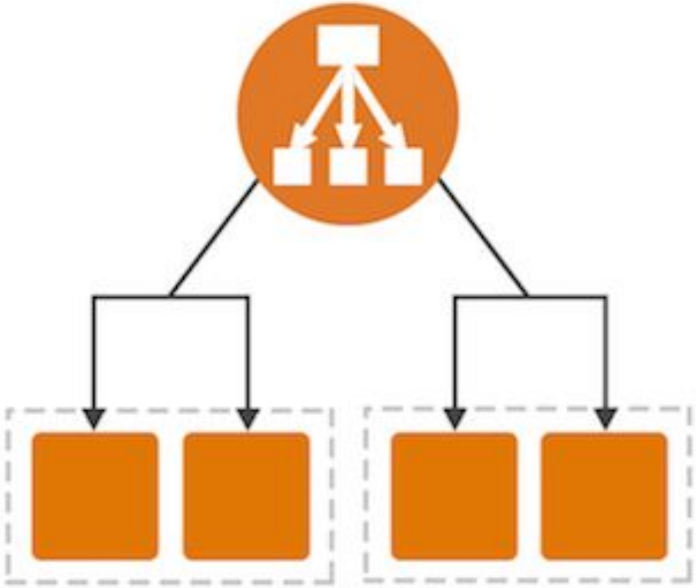
Replace one large resource with multiple small resources to reduce the impact of a single failure on the overall workload.

Distribute requests across multiple, smaller resources to ensure that they don't share a common point of failure.

Scale Horizontally to Increase Aggregate Workload Availability



Scale Horizontally to Increase Aggregate Workload Availability



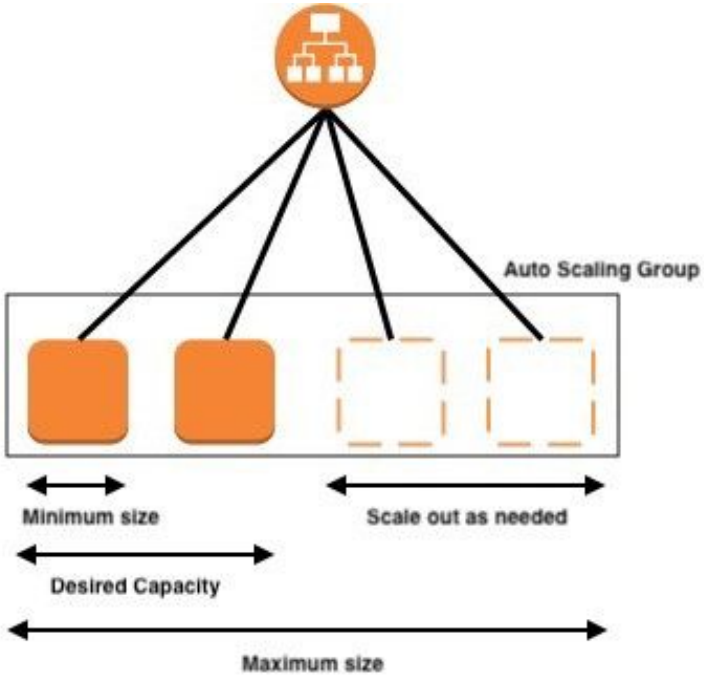
Stop Guessing Capacity

A common cause of failure in on-premises workloads is resource saturation.

In the cloud, you can **monitor demand and workload utilization**, and **automate the addition or removal of resources** to maintain the optimal level to satisfy demand without over- or under-provisioning.

There are still limits, but some quotas can be controlled and others can be managed (see Manage Service Quotas and Constraints).

Stop Guessing Capacity

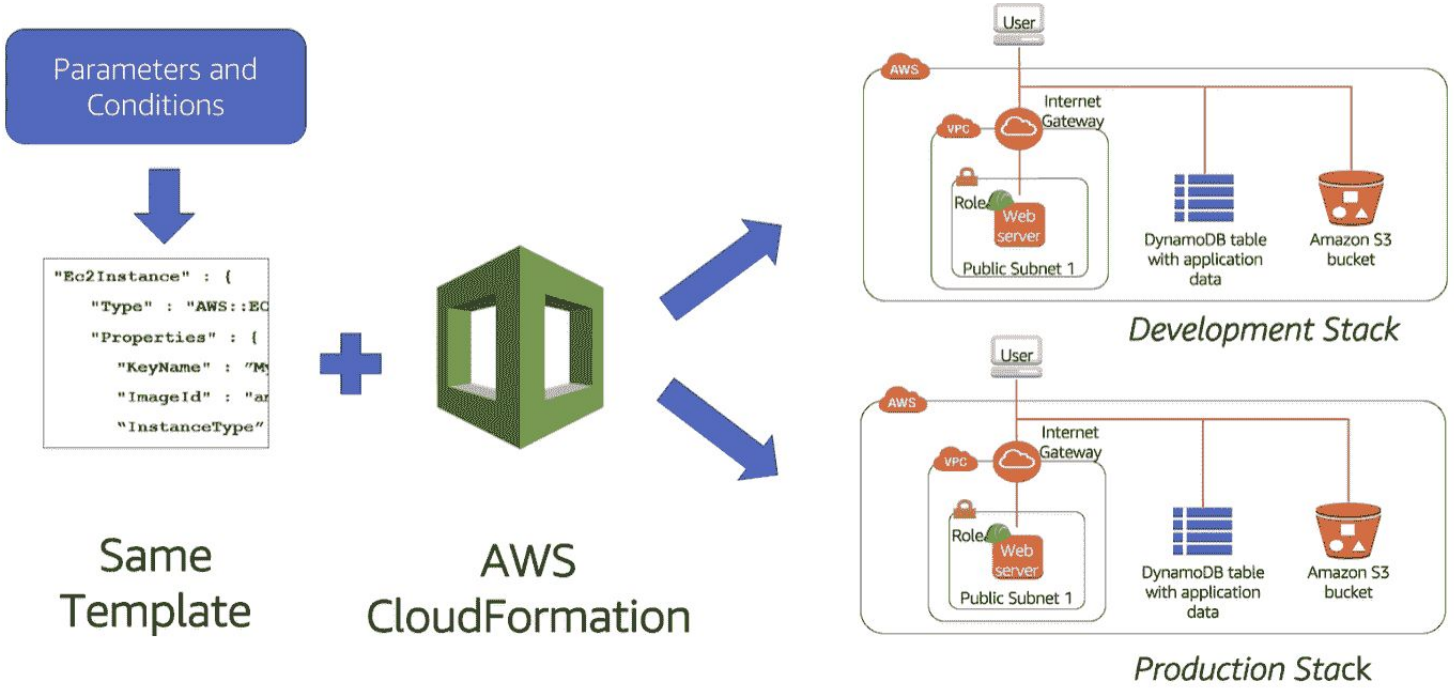


Manage Change In Automation

Manage change in automation: **Changes to your infrastructure should be made using automation.**

The changes that need to be managed include changes to the automation, which then can be tracked and reviewed.

Manage Change In Automation



Reliability Principles Summary

- Automatically recover from failure
- Test recovery procedures
- Scale horizontally to increase aggregate workload availability
- Stop guessing capacity
- Manage change in automation

Activity - Understanding Reliability

In this activity, gather into your own group and each group should take on one or two research problem.

Ensure all research problems are taken and presented by the end of this section.

Activity - Understanding Reliability

- Automatically recover from failure
- Test recovery procedures
- Scale horizontally to increase aggregate workload availability
- Stop guessing capacity
- Manage change in automation

Best Practices For Reliability

Summary

There are four best practice areas for reliability in the cloud:

- Foundations
- Workload Architecture
- Change Management
- Failure Management

Foundations

Foundational requirements are those whose scope extends beyond a single workload or project.

Before architecting any system, **foundational requirements that influence reliability should be in place**.

For example, you must have **sufficient network bandwidth to your data center**.

Foundations

With AWS, most of these foundational requirements are already incorporated or can be addressed as needed.

The cloud is designed to be nearly limitless, so it’s the responsibility of AWS to satisfy the requirement for sufficient networking and compute capacity, leaving you free to change resource size and allocations on demand.

Workload Architecture

A reliable workload **starts with upfront design decisions for both software and infrastructure**.

Your architecture choices will impact your workload behavior across all five Well-Architected pillars.

For reliability, there are specific patterns you must follow.

Workload Architecture

With AWS, workload developers have their **choice of languages and technologies to use**.

AWS SDKs take the complexity out of coding by providing language-specific APIs for AWS services. These SDKs, plus the choice of languages, **allow developers to implement the reliability best practices** listed here.

Developers can also read about and learn from how Amazon builds and operates software in The Amazon Builders' Library.

Change Management

Changes to your workload or its environment **must be anticipated and accommodated to achieve reliable operation** of the workload.

Changes include those imposed on your workload, such as **spikes in demand**, as well as those from within, such as feature deployments and security patches.

Change Management

Using AWS, you can **monitor the behavior of a workload and automate the response to KPIs**.

For example, your workload can add additional servers as a workload gains more users.

You can **control who has permission to make workload changes** and **audit the history of these changes**.

Failure Management

In any system of reasonable complexity, it is **expected that failures will occur.**

Reliability requires that your workload be **aware of failures as they occur** and **take action to avoid impact on availability**.

Workloads **must be able to both withstand failures and automatically repair issues**.

Failure Management

With AWS, you can take advantage of automation to react to monitoring data.

For example, when a **particular metric crosses a threshold, you can trigger an automated action to remedy the problem.**

Also, rather than trying to diagnose and fix a failed resource that is part of your production environment, you can **replace it with a new one** and **carry out the analysis on the failed resource out of band**.

Since the cloud enables you to stand up temporary versions of a whole system at low cost, you can use automated testing to verify full recovery processes.

Failure Management - Chaos

Netflix - Chaos Engineering

Chaos Engineering involves running thoughtful, planned experiments that teach us how our systems behave in the face of failure.

These experiments follow three steps:

Failure Management - Chaos

I. Plan an Experiment

Create a hypothesis. What could go wrong?

II. Contain the Blast Radius

Execute the smallest test that will teach you something.

III. Scale or squash

Find an issue? Job well done. Otherwise increase the blast radius until you're at full scale.

Summary

There are four best practice areas for reliability in the cloud:

- Foundations
- Workload Architecture
- Change Management
- Failure Management

Activity - Understanding Reliability

In this activity, gather into your own group and each group should take on one or two research problem.

Ensure all research problems are taken and presented by the end of this section.

Activity - Understanding Reliability

- How do you manage service quotas and constraints?
- How do you monitor workload resources?
- How do you design your workload service architecture?
- How do you back-up data?

Activity

Learner:

- Clean up AWS.
- Remove/delete/terminate all service/ resources that created.

Instructor

- Clean up AWS.
- Remove/delete/terminate all service/ resources that created.
- Check the AWS account after learner clean up.

What’s Next?