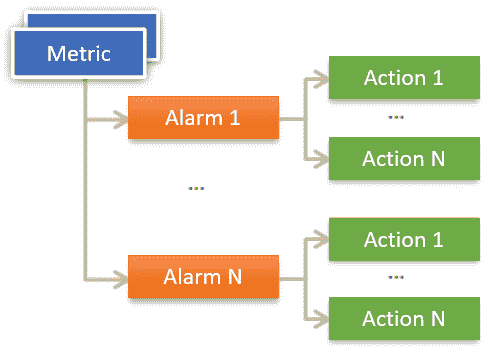
**CLOUDWATCH:**

Q: What is AWS CloudWatch?  
Ans:

*CloudWatch gathers monitoring and operational data in the form of logs, metrics, and events and visualises it by using automated dashboards to provide you with a unified view of your AWS resources, applications, and services which it runs both in AWS and on-premises.*

Q: What is a CloudWatch Alarm?  
Ans:

*The new CloudWatch Alarms feature helps in monitoring CloudWatch metrics and receive notifications when they fall outside of the levels (high or low thresholds) you set. Each metric can have multiple Alarms, each with its own set of actions.*

[](https://www.techgeeknext.com/img/aws/cloudwatch-alarm.PNG)

The status of a CloudWatch Alarm is always one of three: OK, ALARM, or INSUFFICIENT DATA. The Monitor is in the OK state whenever the metric is within the acceptable range which you have defined. When it reaches a certain threshold, it enters the ALARM state. The monitor enters the INSUFFICIENT DATA state when the data required to make the decision is missing or incomplete.

Q: What is the difference between CloudTrail and CloudWatch?  
Ans:

*CloudWatch monitors and reports on the health and performance of AWS services and resources. CloudTrail, on the other hand, is a log of all actions that have occurred within your AWS environment.*

#### Q: What is the difference between CloudTrail and CloudWatch? Ans:

*CloudWatch monitors and reports on the health and performance of AWS services and resources. CloudTrail, on the other hand, is a log of all actions that have occurred within your AWS environment.*

#### Q: How to delete alarms in AWS CloudWatch Console? Ans:

Use the following command to delete the alarms in AWS CloudWatch Console:

aws cloudwatch delete-alarms --region us-east-2 --alarm-names ...

#### Q: What is CloudWatch Metric Streams? Ans:

*CloudWatch Metric Streams is a feature that allows you to stream CloudWatch metrics to a destination of your choice indefinitely with minimal setup and configuration. It is a fully managed solution that eliminates the need for you to write code or maintain infrastructure. Users can configure a metric stream to destinations such as Amazon Simple Storage Service with a few clicks (S3). To maintain your operational dashboards up to date, users could also send the metrics to a number of third-party service providers.*

#### Q: How do I send Grafana from CloudWatch metrics? Ans:

1. Install Grafana : **Follow the steps to**[**Install Grafana**](https://www.techgeeknext.com/tools/docker/install-grafana-using-docker)**.**
2. Go to AWS -> IAM -> Policies.
3. Add below JSON in policy -> Create Policy:
4. {
5. "Version": "2024-10-23", -- current Date
6. "Statement": [
7. {
8. "Sid": "AllowReadingMetricsFromCloudWatch",
9. "Effect": "Allow",
10. "Action": [
11. "cloudwatch:ListMetrics",
12. "cloudwatch:GetMetricStatistics",
13. "cloudwatch:GetMetricData"
14. ],
15. "Resource": "\*"
16. },
17. {
18. "Sid": "AllowReadingTagsInstancesRegionsFromEC2",
19. "Effect": "Allow",
20. "Action": [
21. "ec2:DescribeTags",
22. "ec2:DescribeInstances",
23. "ec2:DescribeRegions"
24. ],
25. "Resource": "\*"
26. }
27. ]
28. }
29. IAM -> Roles -> Create Role -> Select AWS Service / EC2
30. Attach Permission policies
31. IAM -> Users and click Add User ->Attach existing policies -> copy Access Key ID, your Secret Key
32. EC2 -> Instances-> Select Grafana Server and click on Actions -> Instance Settings -> Attach/Replace IAM Role -> Attach your Grafana IAM Role to the instance.
33. Log in to your Grafana Server using Terminal as root user and provide Access Key ID, your Secret Key:

# vim /usr/share/grafana/.credentials

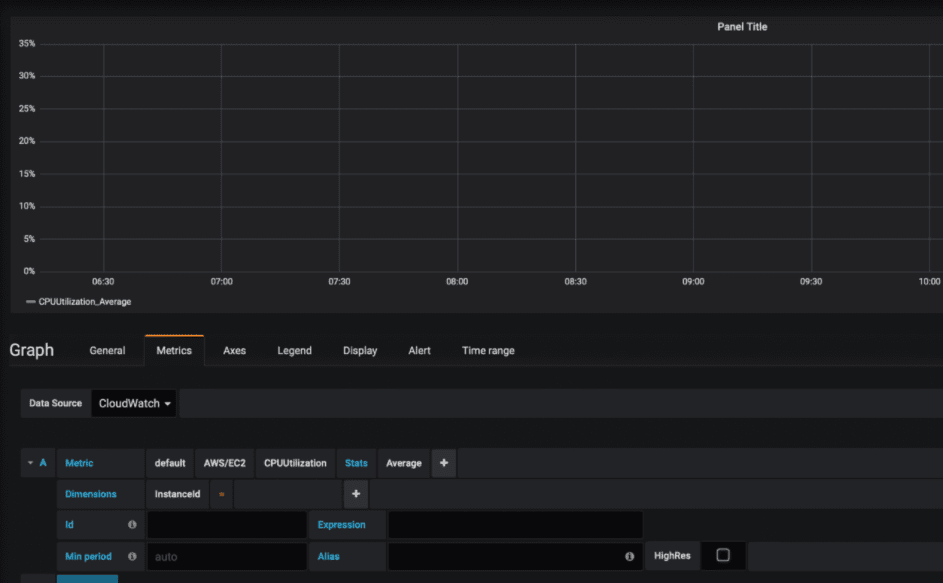
aws\_access\_key\_id = 000000000000

aws\_secret\_access\_key = 0000000000

region = us-west-2

# chmod 0644 .credentials

1. Grafana -> Navigate to Data Sources -> Select CloudWatch Type
2. Create Dashboard -> Select Graph -> Select Panel Title -> edit and provide namespace.

[](https://www.techgeeknext.com/img/aws/grafana-cloudwatch.PNG)**Here's an**[**example of a Grafana dashboard**](https://www.techgeeknext.com/spring-boot/spring-boot-actuator-prometheus-grafana)**that's being used to keep track and monitor the data.**

#### Q: What is Amazon CloudWatch ServiceLens? Ans:

*Amazon CloudWatch ServiceLens is a new feature that lets you visualise and analyse the health, performance, and availability of your applications in one place. Amazon CloudWatch ServiceLens is accessible in all public AWS Regions that offer AWS-X-Ray.*

#### Q: What is Amazon CloudWatch Synthetics? Ans:

*To monitor your endpoints and APIs, you can use Amazon CloudWatch Synthetics to create****canaries****, which are configurable scripts that run on a schedule. Canaries take the same routes and perform the same actions as customers, allowing you to continuously validate your customer experience even if there is no customer traffic on your applications. You can detect problems before your customers do by using canaries.*

Synthetic Monitoring is an effective way of testing a website or web service by simulating visitor requests to test for availability, performance, and functionality.

#### Q: What are Canaries in Amazon CloudWatch Synthetics? Ans:

*Scripts written in Node.js or Python are known as****Canaries****. In your account, users create Lambda functions that use Node.js or Python as a framework. Canaries support both the HTTP and HTTPS protocols.*

**SIMPLE QUEUE SERVICES:**

# **AWS Simple Queue Service – SQS**

# **AWS Simple Queue Service – SQS**

* Simple Queue Service – SQS is a highly available distributed queue system
* A queue is a temporary repository for messages awaiting processing and acts as a buffer between the component producer and the consumer
* is a message queue service used by distributed applications to exchange messages through a **polling** model, and can be used to decouple sending and receiving components.
* is fully managed and requires no administrative overhead and little configuration
* offers a reliable, highly-scalable, hosted queue for storing messages in transit between applications.
* provides fault-tolerant, loosely coupled, flexibility of distributed components of applications to send & receive without requiring each component to be concurrently available
* helps build distributed applications with decoupled components
* supports encryption at rest and encryption in transit using the HTTP over SSL (HTTPS) and Transport Layer Security (TLS) protocols for security.
* provides two types of Queues
  + [Standard Queue](https://jayendrapatil.com/aws-sqs-standard-queue/)
  + [FIFO Queue](https://jayendrapatil.com/aws-sqs-fifo-queue/)

## [**SQS Standard Queue**](https://jayendrapatil.com/aws-sqs-standard-queue/)

* Standard queues are the default queue type.
* Standard queues support at-least-once message delivery. However, occasionally (because of the highly distributed architecture that allows nearly unlimited throughput), more than one copy of a message might be delivered out of order.
* Standard queues support a nearly unlimited number of API calls per second, per API action (SendMessage, ReceiveMessage, or DeleteMessage).
* Standard queues provide best-effort ordering which ensures that messages are generally delivered in the same order as they’re sent.

Refer [SQS Standard Queue](https://jayendrapatil.com/aws-sqs-standard-queue/) for detailed information

## [**SQS FIFO Queue**](https://jayendrapatil.com/aws-sqs-fifo-queue/)

* FIFO (First-In-First-Out) queues provide messages in order and exactly once delivery.
* FIFO queues have all the capabilities of the standard queues but are designed to enhance messaging between applications when the order of operations and events is critical, or where duplicates can’t be tolerated.
* **Work Queues**
  + Decouple components of a distributed application that may not all process the same amount of work simultaneously.
* **Buffer and Batch Operations**
  + Add scalability and reliability to the architecture and smooth out temporary volume spikes without losing messages or increasing latency
* **Request Offloading**
  + Move slow operations off of interactive request paths by enqueueing the request.
* **Fan-out**
  + Combine SQS with SNS to send identical copies of a message to multiple queues in parallel for simultaneous processing.
* **Auto Scaling**
  + SQS queues can be used to determine the load on an application, and combined with Auto Scaling, the EC2 instances can be scaled in or out, depending on the volume of traffic

## **How SQS Queues Works**

* SQS allows queues to be created, deleted and messages can be sent and received from it
* SQS queue retains messages for four days, by default.

Ezoic

* Queues can be configured to retain messages for 1 minute to 14 days after the message has been sent.
* SQS can delete a queue without notification if any action hasn’t been performed on it for 30 consecutive days.
* SQS allows the deletion of the queue with messages in it

## [**SQS Features & Capabilities**](https://jayendrapatil.com/amazon-sqs-features/)

* [Visibility timeout](https://jayendrapatil.com/amazon-sqs-features/#Visibility_timeout) defines the period where SQS blocks the visibility of the message and prevents other consuming components from receiving and processing that message.
* [SQS Dead-letter queues – DLQ](https://jayendrapatil.com/amazon-sqs-features/#SQS_Dead_Letter_Queues_-_DLQ) helps source queues ([Standard](https://jayendrapatil.com/aws-sqs-standard-queue/) and [FIFO](https://jayendrapatil.com/aws-sqs-fifo-queue/)) target messages that can’t be processed (consumed) successfully.
* DLQ Redrive policy specifies the source queue, the dead-letter queue, and the conditions under which SQS moves messages from the former to the latter if the consumer of the source queue fails to process a message a specified number of times.
* [SQS Short and Long polling](https://jayendrapatil.com/amazon-sqs-features/#Short_and_Long_polling) control how the queues would be polled and Long polling help reduce empty responses.

## SQS Buffered Asynchronous Client

* Amazon SQS Buffered Async Client for Java provides an implementation of the AmazonSQSAsyncClient interface and adds several important features:
  + **Automatic batching** of multiple SendMessage, DeleteMessage, or ChangeMessageVisibility requests without any required changes to the application
  + **Prefetching** of messages into a local buffer that allows the application to immediately process messages from SQS without waiting for the messages to be retrieved
* Working together, automatic batching and prefetching increase the throughput and reduce the latency of the application while reducing the costs by making fewer SQS requests.

## **SQS Security and reliability**

* SQS stores all message queues and messages within a single, highly-available AWS region with multiple redundant Availability Zones (AZs)
* SQS supports HTTP over SSL (HTTPS) and Transport Layer Security (TLS) protocols.
* SQS supports Encryption at Rest. SSE encrypts messages as soon as SQS receives them and decrypts messages only when they are sent to an authorized consumer.
* SQS also supports resource-based permissions