**AWS SQS**

Amazon *Simple Queue Service (SQS)* is a fully managed message queuing service provided by Amazon Web Services (AWS). It allows you to ***decouple*** and scale microservices, distributed systems, and serverless applications by providing a reliable way to transmit messages between different components or systems.

SQS follows a "producer-consumer" model, where one component (producer) sends messages to a queue, and another component (consumer) processes those messages from the queue.

**Key features of Amazon SQS:**

**1. Fully Managed:** AWS manages the infrastructure for you, including message durability, availability, and scalability.

**2. Decoupling:** SQS enables loose coupling between components, allowing them to work independently without being directly connected.

**3. Reliability:** Messages are stored redundantly across multiple availability zones to ensure high availability.

**4. Scalability:** SQS can handle a virtually unlimited number of messages, and you can scale your applications without worrying about managing the underlying infrastructure.

**5. Delay Queues:** You can configure a delay for messages, which allows you to control when messages become visible in the queue for consumers.

**6. Message Visibility Timeout:** When a consumer receives a message from the queue, SQS hides it from other consumers for a configurable period. This prevents multiple consumers from processing the same message simultaneously.

**7. Long Polling:** Consumers can request messages from the queue with a longer polling time, reducing the number of empty responses and providing better efficiency.

**8. Dead-Letter Queues:** If a message fails to be processed a certain number of times, you can move it to a separate queue (dead-letter queue) for analysis and troubleshooting.

**9. Message Attributes:** Messages can have custom attributes that provide additional metadata.

**10. FIFO Queues:** First-In-First-Out (FIFO) queues ensure that the order in which messages are sent is the same order in which they are received.

**11. Event-Driven Architecture:** SQS is often used in event-driven architectures, such as processing notifications, triggering Lambda functions, and more.

**To use SQS, you typically:**

1. Create an SQS Queue: You create a queue, which acts as a buffer between your components.

2. Send Messages: Producers send messages to the queue.

3. Receive and Process Messages: Consumers retrieve messages from the queue and process them.

4. Delete Messages: Once a message is successfully processed, the consumer deletes it from the queue.

**SQS supports two types of queues:** *Standard* Queues and *FIFO* Queues. Standard Queues provide best-effort ordering and high throughput, while FIFO Queues provide strict message ordering and deduplication to ensure that a message is processed only once.

In your application, you use the AWS SDK (version 1 or version 2) to interact with SQS, as shown in the examples provided earlier. Make sure to set up your AWS credentials, configure your queue, and handle exceptions appropriately to build a robust and reliable messaging system using SQS.

**AWS CLI COMMANDS**

AWS Command Line Interface (AWS CLI) commands to interact with Amazon SQS (Simple Queue Service) and manage queues using the AWS CLI on an AWS EC2 instance.

These commands assume that you have the AWS CLI configured on your EC2 instance with appropriate IAM permissions.

**1. List SQS Queues:**

aws sqs list-queues

**2. Create an SQS Queue:**

aws sqs create-queue --queue-name my-queue-name

**3. Send a Message to an SQS Queue:**

aws sqs send-message --queue-url QUEUE\_URL --message-body "Hello, SQS!"

**4. Receive Messages from an SQS Queue:**

aws sqs receive-message --queue-url QUEUE\_URL --max-number-of-messages 5

Replace `QUEUE\_URL` with the actual URL of your SQS queue.

**5. Delete a Message from an SQS Queue:**

aws sqs delete-message --queue-url QUEUE\_URL --receipt-handle RECEIPT\_HANDLE

Replace `QUEUE\_URL` with the actual URL of your SQS queue and `RECEIPT\_HANDLE` with the receipt handle of the message you want to delete.

**6. Delete an SQS Queue:**

aws sqs delete-queue --queue-url QUEUE\_URL

Replace `QUEUE\_URL` with the actual URL of the SQS queue you want to delete.

**7. Purge All Messages from an SQS Queue:**

aws sqs purge-queue --queue-url QUEUE\_URL

Replace `QUEUE\_URL` with the actual URL of the SQS queue you want to purge.

**AWS SDK version 2 (AWS SDK v2) with Java and Maven to interact with AWS SQS:**

**1. Create a Maven Project:**

Create a new Maven project in IDE or using the command line.

**2. Add Dependencies to pom.xml:**

Add the AWS SDK v2 dependencies to your `pom.xml` file:

<dependencies>

<dependency>

<groupId>software.amazon.awssdk</groupId>

<artifactId>sqs</artifactId>

<version>2.17.36</version>

</dependency>

<dependency>

<groupId>software.amazon.awssdk</groupId>

<artifactId>auth</artifactId>

<version>2.17.36</version>

</dependency>

</dependencies>

**3. Java Code:**

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;

import software.amazon.awssdk.regions.Region;

import software.amazon.awssdk.services.sqs.SqsClient;

import software.amazon.awssdk.services.sqs.model.\*;

public class SQSExample {

public static void main(String[] args) {

String queueUrl = "https://sqs.ap-south-1.amazonaws.com/654283987332/TestQueue";

SqsClient sqsClient = SqsClient.builder()

.region(Region.US\_EAST\_1) **// Change to your desired region**

.credentialsProvider(DefaultCredentialsProvider.create())

.build();

***// Send a message to the queue***

SendMessageRequest sendMessageRequest = SendMessageRequest.builder()

.queueUrl(queueUrl)

.messageBody("Hello, SQS!")

.build();

sqsClient.sendMessage(sendMessageRequest);

***// Receive messages from the queue***

ReceiveMessageRequest receiveMessageRequest = ReceiveMessageRequest.builder()

.queueUrl(queueUrl)

.maxNumberOfMessages(5)

.build();

for (Message message : sqsClient.receiveMessage(receiveMessageRequest).messages()) {

System.out.println("Received: " + message.body());

}

// Close the SQS client

sqsClient.close();

}

}

**Replace `QUEUE\_URL` with your actual SQS queue URL.**

4. Run the Application:

- Compile and run the Java application.

- You should see messages being sent and received from the SQS queue.