**What is Amazon SQS**

Fully managed **messaging queuing service** you can use decouple applications so they can run independently.

A queue is a temporary storage for messages waiting to be processed. Messages are stored until they are processed. Messages can stay from 1 to 14 days with default being 4 days. Messages can contain 256 KB text of any format.

SQS is capable of processing and managing messages at massive scale. It can process billions of messages per day.

Messages are stored in single region within multiple redundant available availability zones and therefore any single availability zone failure or network failure cannot make the messages inaccessible.

Messages can be sent or read simultaneously.

Messages can be shared anonymously, or you can share them with other AWS accounts. Message sharing can be restricted by IP address or time of day.

Messages in SQS are encrypted with server-side-encryption using KMS and are decrypted only when they are sent to authorized consumers.

SQS support multiple producer and consumer working with the same queue.

**SQS and Loose Coupling**

* Amazon SQS supports asynchronous processing and therefore the response time is faster.
* Since the processing is asynchronous you can increase your number of jobs processing SQS and performance is improved.
* Since messages are remained in queue you can recover your application from failure and continue processing messages form queue.

**SQS general use cases**

**Work Queue** – You can use work queue to decouple components of your enterprise application.

**Buffering Batch operations** – You can utilize SQS to buffer batch operations and to add scalability and reliability to your architecture. For temporary spikes messages are stored in queue and can be processed by the batch jobs with increasing latency. Since it’s a durable storage it will ensure you do not loose any messages in between.

**Request Offloading** – It can be used for slow operations (request processing).

**Auto Scaling** – Depending on the volume of traffic you can use Autoscaling to scale your ec2 instances automatically.

**Type of queues in SQS**

**Standard Queues**

* Standard queues guarantee at-least-once delivery where a message is delivered at least once. Although for some rare occasions more than one copy of the message is delivered.
* Ensure the best-effort ordering of the messages. Although some occasions messages might not be delivered in the order, they are sent in.
* Standard que supports unlimited number of transactions per second per API transaction.

**FIFO Queues**

* Guarantees exactly once processing and processes in the exact order they are sent and received.
* Provides high throughput (300 per second i.e. 300 send, receive or delete message per second)

If you batch 10 messages per operation FIFO queue can support 3000 messages per second.

**Features of SQS**

**DEAD Letter Queues**

* Contains messages that cannot be processed. It receives messages after the maximum number of processing attempts are reached.
* Same as SQS and it can receive and send messaged.
* A dead letter queue can be created from AQS API/console.

**Visibility Timeout**

* This is the time SQS prevents other consumers can read/process same message.
* Assures a message is not processed multiple times.
* During visibility timeout component receives, process the message, and deletes it.
* Default visibility timeout is 30 seconds and max 5 hours.
* If the message is not processed before visibility timeout it becomes visible to other consumers and can be processed again. Typically set the visibility timeout maximum so the message must be processed during that time.

**Short/long Polling**

Amazon SQS supports both short and long polling.

By default, it supports short polling where it queries a subset f the servers based on random distribution and find messages that can be included in the response. SQS send the response right away even if it does not find any messages.

Long poling queries all the servers for messages and sends the response after it collects maximum number of messages for the response or if the polling wait time expires.

Long polling reduces SQS costs as it eliminates the empty response.

SQS does not automatically delete the message after the message is processed. It is consumer responsibility to delete the message. There is a chance of duplicate message processing if the message is not deleted by the consumer (After the visibility timeout is complete another consumer can process the message again).

**SQS Use Cases**

1. Service-to-service communication
2. Asynchronous message communication
3. State change notifications

**SQS When not to be used**

1. Selecting specific messages
2. Message size is large – best case is storing the large object in a storage system e.g. S3 and send the reference to SQS.