## What is Amazon Simple Queue Service?

Amazon Simple Queue Service (Amazon SQS) offers a reliable, highly-scalable hosted queue for storing messages as they travel between applications. It moves data between distributed application components and helps us decouple these components. Amazon SQS provides familiar middleware constructs such as dead-letter queues. It also provides a generic web services API and can be accessed by any programming language that the AWS SDK supports. Amazon SQS supports both standard and FIFO queues.

## How we can use amazon SQS?

We use Amazon SQS when we need each unique message to be consumed only once and for cases such as the following:

**Decoupling the components of an application** – We have a queue of work items and want to track the successful completion of each item independently. Amazon SQS tracks the ACK/FAIL results, so the application does not have to maintain a persistent checkpoint. After a configured visibility timeout, Amazon SQS deletes acknowledged messages and redelivers failed messages.

**Configuring individual message delay** – We have a job queue and we need to schedule individual jobs with a delay. With standard queues, we can configure individual messages to have a delay of up to 15 minutes.

**Dynamically increasing concurrency or throughput at read time** – We have a work queue and want to add more consumers until the backlog is cleared. Amazon SQS requires no pre-provisioning.

**Scaling transparently** – We buffer requests and the load changes as a result of occasional load spikes or the natural growth of our business. Because Amazon SQS can process each buffered request independently, Amazon SQS can scale transparently to handle the load without any provisioning instructions from us.

## Standard Queue

- Available in all regions.
- High Throughput – Standard queues have nearly-unlimited transactions per second (TPS).
- At-Least-Once Delivery – A message is delivered at least once, but occasionally more than one copy of a message is delivered.

- Best-Effort Ordering – Occasionally, messages might be delivered in an order different from which they were

Send data between applications when the throughput is important, for example:

- Decouple live user requests from intensive background work: let users upload media while resizing or encoding it.
- Allocate tasks to multiple worker nodes: process a high number of credit card validation requests.
- Batch messages for future processing: schedule multiple entries to be added to a database.

**FIFO Queue**

- Available in limited AWS regions. AS of now in the US East (N. Virginia), US East (Ohio), US West (Oregon), and EU (Ireland) regions.
- First-In-First-Out Delivery – The order in which messages are sent and received is strictly preserved.
- Exactly-Once Processing – A message is delivered once and remains available until a consumer processes and deletes it. Duplicates are not introduced into the queue
- Limited Throughput – 300 transactions per second (TPS).
  Send data between applications when the order of events is important, for example:
- Ensure that messages are executed in the right order.
- Display the correct product price by sending price modifications in the right order.

**What Are the Main Features of Amazon SQS?**

Amazon SQS provides the following major features:

**Redundant infrastructure** – Standard queues support at-least-once message delivery, while FIFO queues support exactly-once message processing. Amazon SQS provides highly-concurrent access to messages and high availability for producing and consuming messages.

**Multiple producers and consumers** – Multiple parts of our system can send or receive messages at the same time. Amazon SQS locks the message during processing, keeping other parts of our system from processing the message simultaneously.

**Configurable settings per queue** – All of our queues don't have to be exactly alike. For example, we can optimize one queue for messages that require a longer processing time than others.

**Variable message size** – Our messages can be up to 262,144 bytes (256 KB) in size. We can store the contents of larger messages using the Amazon Simple Storage Service (Amazon S3) or Amazon DynamoDB, with Amazon SQS holding a pointer to the Amazon S3 object. For more information, see Managing Amazon SQS Messages with Amazon S3. We can also split a large message into smaller ones.

**Access control** – We control who can send messages to a queue, and who can receive messages from a queue.

**Delay queues** – We can set a default delay on a queue, so that delivery of all enqueued messages is postponed for the specified duration. We can set the delay value when we create a queue and we can update the value with SetQueueAttributes. If we update the value, the new value affects only messages enqueued after the update.

**PCI compliance** – Amazon SQS supports the processing, storage and transmission of credit card data by a merchant or service provider and has been validated as compliant with Payment Card Industry (PCI) Data Security Standard (DSS).

**HIPAA compliance** – AWS has expanded its HIPAA compliance program to include Amazon SQS as a HIPAA Eligible Service. If we have an executed Business Associate Agreement (BAA) with AWS, we can use Amazon SQS to build our HIPAA-compliant applications, store messages in transit and transmit messages—including messages containing protected health information (PHI).

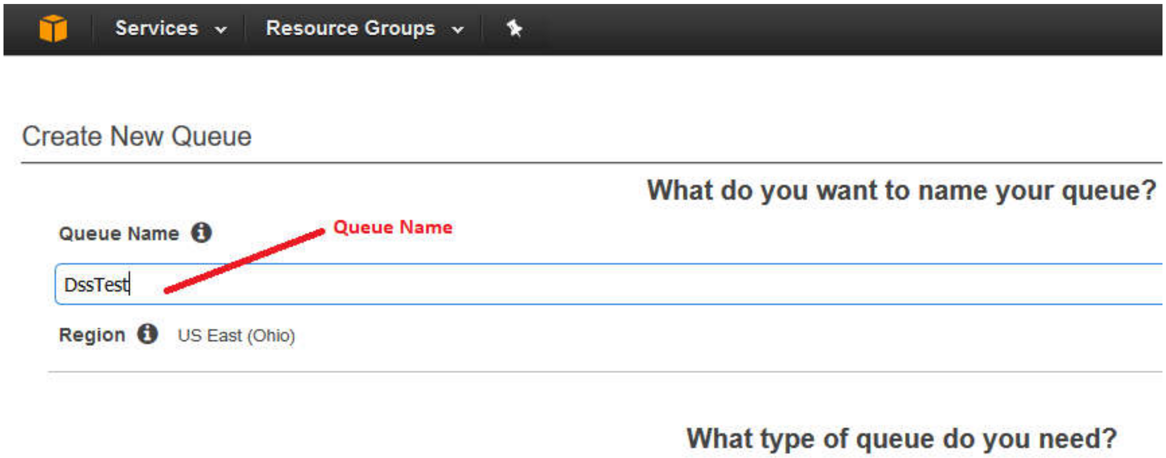
**What is the Basic Architecture of Amazon SQS?**

There are three main actors in the overall system:

- The components of your distributed system
- Queues
- Messages in the queues

**To create Amazon Simple Queue Service we need to follow the particular steps.**

**Step: 1**



Create New Queue

What do you want to name your queue?

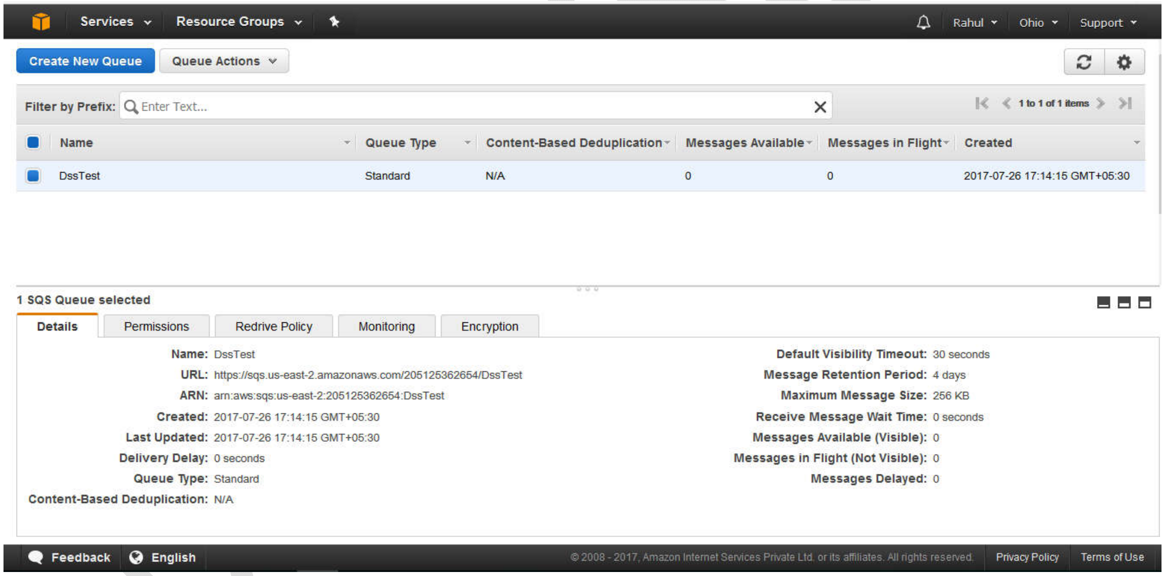
Queue Name: DssTest

Queue Name

Region: US East (Ohio)

What type of queue do you need?

**Step 2: --In the next step we can find the details of the SQE**



| Name | Queue Type | Content-Based Deduplication | Messages Available | Messages in Flight | Created |
|---|---|---|---|---|---|
| DssTest | Standard | N/A | 0 | 0 | 2017-07-26 17:14:15 GMT+05:30 |

1 SQS Queue selected

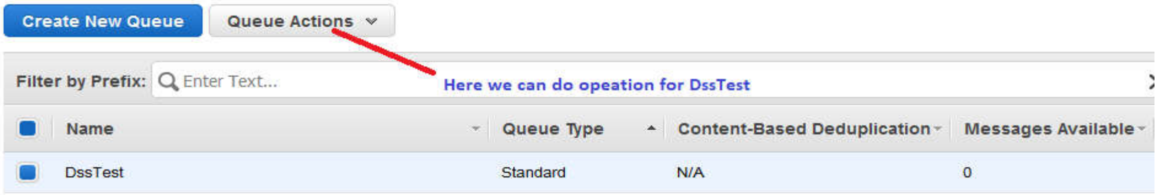
Details | Permissions | Redrive Policy | Monitoring | Encryption

Name: DssTest
URL: https://sqs.us-east-2.amazonaws.com/205125362654/DssTest
ARN: arn:aws:sqs:us-east-2:205125362654:DssTest
Created: 2017-07-26 17:14:15 GMT+05:30
Last Updated: 2017-07-26 17:14:15 GMT+05:30
Delivery Delay: 0 seconds
Queue Type: Standard
Content-Based Deduplication: N/A

Default Visibility Timeout: 30 seconds
Message Retention Period: 4 days
Maximum Message Size: 256 KB
Receive Message Wait Time: 0 seconds
Messages Available (Visible): 0
Messages in Flight (Not Visible): 0
Messages Delayed: 0

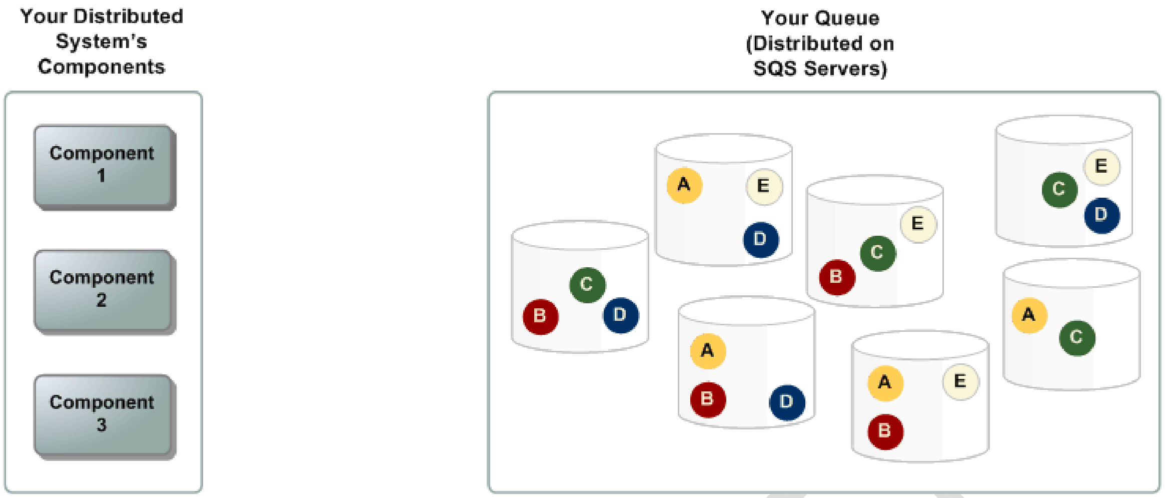
© 2008 - 2017, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.

**Step 3: --**



Here we can do opeation for DssTest

| Name | Queue Type | Content-Based Deduplication | Messages Available |
|---|---|---|---|
| DssTest | Standard | N/A | 0 |


Your Distributed System's Components
Component 1
Component 2
Component 3
Your Queue (Distributed on SQS Servers)
A
E
D
E
C
D
C
B
D
E
C
B
A
B
D
A
E
B
A
C