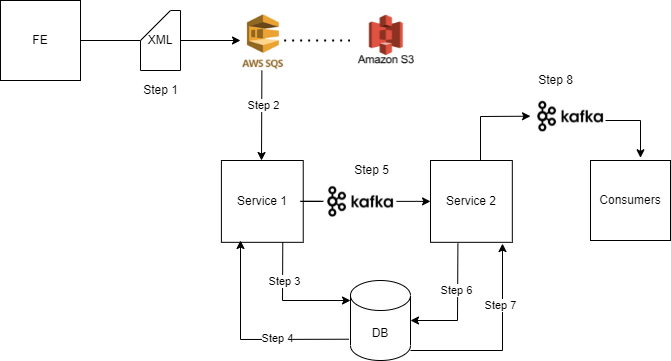
**Event Driven Microservice Via Kafka**



**Introduction: -**

The above diagram describes services interacting with each other via Kafka (kind of event driven approach) with each service fulfilling its own business needs and publishing the end result to Consumer’s topic

**What is Event-Driven Approach?**

An event-driven architecture uses events to trigger and communicate between decoupled services and is common in modern applications built with microservices.

An event is a change in state, or an update, like an item being placed in a shopping cart on an e-commerce website.

Events can either carry the state (the item purchased, its price, and a delivery address) or events can be identifiers (a notification that an order was shipped).

**Components Defined in Above Architecture**

1. **FE (Front End) -** FE is any service/application that sends data to downstream system for processing. In this case the data is XML Files.—unprocessed data
2. **AWS SQS (Amazon Simple Queue Service) –**

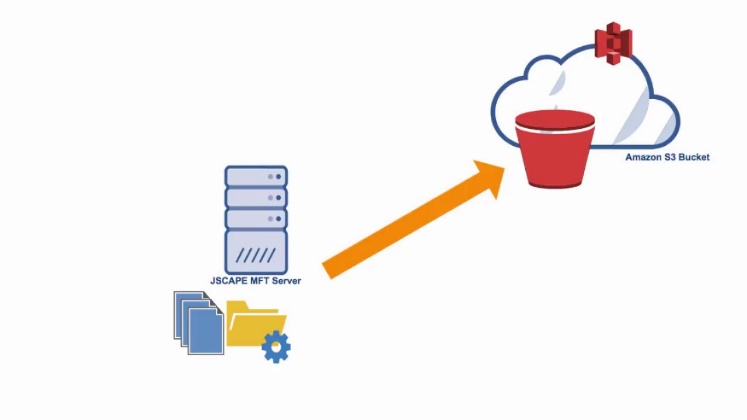


Queue

**Queue** is the data structure that is similar to the queue in the real world. A queue is a data structure in which whatever comes first will go out first, and it follows the FIFO (First-In-First-Out) policy.

Amazon SQS is a distributed queue system that enables web service applications to quickly and reliably queue messages that one component in the application generates to be consumed by another component where a queue is a temporary repository for messages that are awaiting processing.

1. **Amazon S3 –**



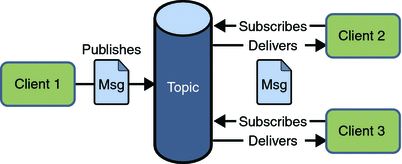
Amazon S3 is a resource that’s built to store, protect, and retrieve data from “buckets” at anytime from anywhere on any device.

Organizing, storing and retrieving data in Amazon S3 focuses on two key components: **buckets** and **objects** that work together to create the storage system.

 In [Amazon S3](https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html) terms, **objects** are data files, including documents, photos, and videos. Each object is identified by a unique key within the S3 environment that differentiates it from other stored objects.

In an S3 environment, objects need somewhere to go, which is why buckets exist, serving as fundamental storage containers for objects.

1. **Kafka –**



Apache Kafka is a software platform which is based on a distributed streaming process.

It is a publish-subscribe messaging system which let exchanging of data between applications, servers etc

A real-life example of Publish-Subscribe is Dish TV, which publishes different channels like sports, movies, music, etc., and anyone can subscribe to their own set of channels and get them whenever their subscribed channels are available.

1. **DB –** DB here is any Database used for storage.
2. **Service 1 & Service 2 –** These are any two microservices fulfilling their respective business needs
3. **Consumers –** Consumers here refers to any end users/clients/vendors or any other services who needs the final processed output data.

**Data Flow: -**

**Step 1: -** The FE System pushes the data to Amazon SQS queue in order to maintain the sequence of data processing. [The data is pushed to S3 bucket in case its size is more than threshold defined in SQS configurations]

**Step 2: -** Service 1 takes data from SQS Queue, applies business logic to it and calculates the hash code of the data to be stored in the DB.

**Step 3: -** It then queries the hash code from DB (of existing data) from the respected entities.

**Step 4: -** As soon as the hash is retrieved from DB, It is matched with the calculated one. If the hash code matches that means DB is already updated and no write operation is required, if an un-match is found DB is updated.

**Step 5: -** ***This is the step in which the Service 1 actually communicates with Service 2 via Kafka.***

The Service 1 then publishes an internal message to Kafka to be consumed by Service 2.

Service 2 consumes the message from Kafka topic and checks the timestamp of the message.

**Step 6: -** Service 2 checks the timestamp stored in DB.

**Step 7: -** The timestamp is retrieved from DB.

If it matches that means DB is recently updated and Consumers need to be informed about this real time update. It then constructs JSON/Avro Schema of the data that is needed by Consumer.

**Step 8: -** The final message is published to Kafka so that End Consumers can consume it.