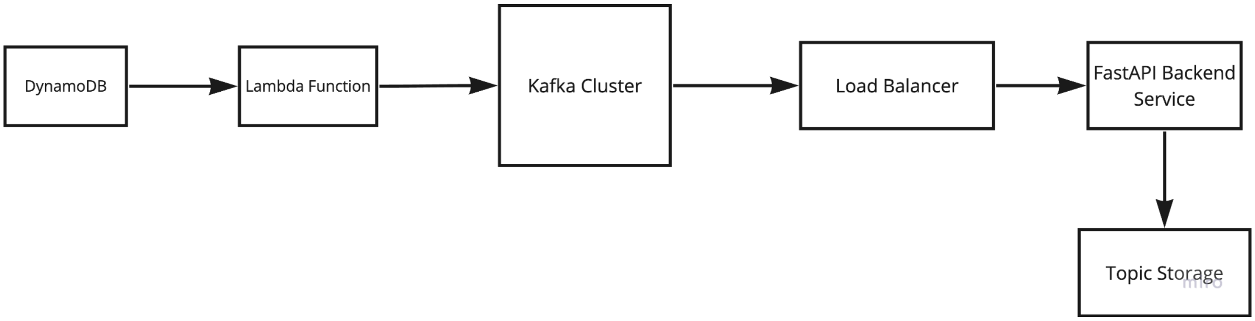
Topic Modeller Service Design Document

Description

This document proposes the design to operationalise the topic modeller service in order to scale for millions of URLs.



High-Level Design for Topic Modeller Service

Components

1. DynamoDB- We will use NoSQL storage to manage the storage of URLs.
2. Lambda Function - The Lambda function can be used to trigger on the basis of data updation events in dynamoDB.It will be publishing the new URLs to our Kafka Cluster.
3. Kafka Cluster - The Kafka Cluster will act as a messaging broker and will cater to the scale of millions of URLs.
4. FastAPI Backend - Our FastAPI-based backend service will consume the data from Kafka and extract the relevant topics after scraping the URLs.
5. Topic Storage - We can use any NoSQL-based storage to store the relevant topics extracted from the URL.

Scaling

1. AWS Lambda is designed to automatically scale based on the number of incoming triggers or events. When the number of triggers from DynamoDB increases to a substantial level, AWS Lambda can handle the increased workload by scaling

horizontally, which means it creates more instances (containers) of our Lambda function to process the events concurrently.

2. FastAPI backend service can be scaled horizontally by running a number of stateless replicas of the service in front of a load balancer to balance the load between each of them.
3. Kafka cluster scales by adding more Kafka broker nodes to the cluster to distribute the message processing and storage load.

Efficiency

1. We can increase the efficiency of our backend service by refining the process of parsing the HTML page.
2. The efficiency of the LDA model can be increased by following steps
    a. Data Preprocessing - Ensure that our text data is properly preprocessed. Remove noise, such as HTML tags, special characters, and punctuation. Normalize text by converting to lowercase and stemming/lemmatizing words
    b. Feature Selection - Limit the vocabulary size by considering only the most frequent words or words that occur above a certain threshold. Reducing the vocabulary size can improve efficiency.
    c. Optimized Model Parameters -
        i. Number of Topics: Choosing an appropriate number of topics for our dataset.
        ii. Alpha and Beta Parameters: Experimenting with the hyperparameters alpha and beta to influence the topic distributions.
    d. Parallelization - LDA can be parallelized for more efficient training. We can use multi-core processors or distributed computing frameworks like Dask or Spark for parallel LDA training.