1. **High Level Design:** The ppt has high level design which can be explained below:
2. Customer adds item to cart or places the order on website
3. This invokes a JAVA web service at the backend.
4. This JAVA web service acts as a producer and publishes messages to Kafka topic. There can be separate topics for adding the item to cart and successful order.
5. Kafka acts as the primary data ingestion tool here. Now the data from Kafka needs to flow to HDFS.
6. This is done by using Spark as the consumer for Kafka topics. Spark subscribes to Kafka topics and pulls the data real time as and when it is published to Kafka.
7. This data is then stored in no sql database like hbase.
8. Now whenever a duplicate item is added in cart, the information flows to Spark via web service and Kafka. The Kafka topic in this case will have the information about customer ID, Product ID and the timestamp.
9. Spark will validate this information against hbase which will have active cart information and order history information in different tables. Customer ID, Product ID and timestamp will be the primary fields used for validation.
10. This validation can be done using Scala or python API over Spark.
11. In case there is a duplicate item in hbase active cart and order history tables according to the required criteria, Scala/Python API over Spark directly connects to JAVA web service and sends the message indicating duplicate item.
12. This message in turn is populated on website by JAVA web service.
13. **Choice of Technologies and tools:** Below are the reasons behind choosing various technologies
14. JAVA Web service: Offers easiest solution to interact between web applications and other interfaces.
15. Kafka: It is chosen as data ingestion tool due to various factors. Some of them are reliability, ability to process huge volumes of data and ability to process this data at real time.
16. Spark: It is chosen because it does in memory computing thus increasing the processing speed of application.
17. Scala/Python API: Either of the two can be chosen. Both are perfectly suitable for the problem in hand and we can decide on the choice of API based on Scala/Python developers availability.
18. Hbase: Reason behind chosing no sql database is its ability to process huge volumes of data at a fast processing speed. Hbase is chosen as it can be easily integrated with HDFS.
19. **Data Modelling:** There will be two tables in no sql database for the problem in hand, Active cart table and Order History table. Below is how the tables would look like:

**ActiveCart**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Column Family: Product** | | |
| **Cust ID** | **Product ID** | **Quantity** | **Timestamp** |
| 123 | ABC | 1 | 201809230601011 |
| 124 | XYZ | 1 | 201809230601011 |
| 125 | ABC | 1 | 201809230601011 |

**OrderHistory**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Cust ID** | **Order ID** | **Product ID** | **Quantity** | **Timestamp** |
| 123 | 999 | ABC | 1 | 201809230601011 |
| 123 | 998 | XYZ | 1 | 201809220601011 |
| 123 | 997 | ABC | 1 | 201809210601011 |

1. **Code to consume the events from Kafka:**
2. **Code to validate current order details against active cart or order history:**