**Real-Time Analytics Dashboard with Stream Processing**

HLD:

A screenshot of a computer

Description automatically generated

**Functional Requirement:**

1. Produce the Data and send it to the Kafka Topic.
2. Consume the data to process and aggregate it.
3. Send the Data to elastic search.
4. Kibana reads the data and displays data for analysis.

**Non-Functional Requirement:**

**Scalability & Fault Tolerance**

* **Kafka:** Configure partitions, replication, and consumer group balancing.
* **Kafka Streams:** Stateless operations for scalability, checkpointing for fault tolerance.
* **Elasticsearch:** Auto-scaling and replica shards for high availability.
* **AWS Lambda:** Scalable for event bursts, set appropriate limits for concurrent executions.

**Security:**

* **Elasticsearch:** Secure indexes using authentication and access control.
* **Kibana:** Use role-based access for dashboards.

**Application:** Secure REST APIs with OAuth2 or token-based authentication

**Major Component:**

1. Dynamodb
2. Spring Boot Application to deploy in AWS Lambda
3. AWS Lambda
4. Kafka
5. Spring Boot Application (Kafka Consumer for Data Processing and aggregation)
6. Logstash
7. Elastic Search
8. Kibana

**Integration Points:**

1. DynamoDB → AWS Lambda
2. AWS Lambda → Kafka
3. Kafka → Spring Boot Kafka Streams.
4. Spring Boot Kafka Streams → Logstash
5. Logstash → Elasticsearch.
6. Elasticsearch → Kibana.

**AWS I am:**

**DynamoDB:**

Created Table: - **UserEventLogs**

Created Items:

|  |  |
| --- | --- |
| eventId | logevent |
| 103 | {  "eventId": "103",  "eventType": "search",  "timestamp": "2024-12-12T10:03:30Z",  "userId": "amit kumar ",  "sessionId": "abcd1234",  "query": "wireless headphones",  "device": {  "type": "mobile",  "os": "iOS",  "browser": "Safari"  },  "location": {  "country": "USA",  "city": "San Francisco"  }  } |

**AWS Lambda:**

Lambda Function – created new Lambda Function with name **AnalyticDynamoToKafka**

Trigger: Use DynamoDB as trigger configuration and use Table name **UserEventLogs**.

Application: Created Spring Boot Application and deployed to AWS Lambda. Aws lambda will get triggered once new item added to the table or modify existing item.

**Kafka:**

EC2: Created new EC2 instance **Analytic\_Kafka\_Server**

Basic setup:

* 1. Install java
  2. Download and extract Kafka
  3. Configure Zookeeper
  4. Configure Kafka Broker
  5. Created Kafka Topic:
  6. Create a consumer group:

**Spring Boot Application:**

Build an application to which will receive the data from kafka to process the data and sent it to the event specific topic and consumed by specific consumer

EC2: Created new EC2 instance **Analytics\_DashBoard\_App**

**Logstash:**

**Elastic Search:**

**Kibana:**