Implementation Notes

1. Introduction

This document provides an overview of the implementation of the real-time data pipeline. The system consists of various components such as Kafka, Flink, Flask, PostgreSQL, and AWS services that work together to ingest, process, store, and retrieve data efficiently.

2. Kafka Producer

The Kafka Producer is responsible for simulating real-time data and publishing it to the Kafka topic. It generates random sensor data (device ID, temperature, humidity, timestamp) and continuously sends it to Kafka.

3. Kafka Broker

The Kafka Broker acts as a message queue that stores and distributes real-time data between producers and consumers. It ensures high availability and fault tolerance.

4. Stream Processing with Apache Flink

Flink processes incoming Kafka messages in real time. It performs operations such as filtering, transformation, window-based aggregation, and anomaly detection before forwarding the processed data.

5. Event Processor (Flask API)

The Flask API acts as an intermediary that processes incoming Kafka messages. It runs a background Kafka consumer to fetch messages and provides a health check endpoint for monitoring.

6. Database Connector

The Database Connector listens for processed data from Kafka and stores it in a PostgreSQL database. It ensures persistence and allows querying of historical data.

7. AWS Services

AWS services such as S3, Lambda, and API Gateway facilitate data storage, processing, and external API access. Data can be archived in S3, and Lambda functions can process events for

downstream applications.

8. Conclusion

This real-time data pipeline leverages Kafka for messaging, Flink for stream processing, Flask for event handling, and PostgreSQL for storage. The modular architecture ensures scalability, fault tolerance, and efficient processing.