Objective: Building a real-time data streaming pipeline to analyse huge data using Hadoop technologies like Kafka, Spark, Zookeeper, Apache Airflow, Cassandra,Docker and PostgreSQL.

Scope of This Project?

Real-time analysis is crucial for various businesses and industries for several reasons. When it comes to data processing projects, real-time analysis can provide significant advantages, and here are some key reasons why it's essential:

Information of customer:

Real-time data processing allows businesses to collect, analyze, and respond to customer data as it happens. This enables them to personalize customer interactions, make product recommendations, and provide better customer service based on immediate insights. This, in turn, improves the overall customer experience, leading to increased customer satisfaction and loyalty.

Personalization:

Real-time data processing enables businesses to understand individual customer in real-time. This information can be used to create personalized marketing campaigns, product recommendations, and content, increasing the likelihood of customer engagement and conversion.

Fraud Detection and Prevention:

In industries like finance and e-commerce, real-time customer analysis is crucial for detecting and preventing fraud. By continuously monitoring transactions and customer behavior patterns, businesses can quickly identify and respond to suspicious activities, protecting themselves and their customers from potential fraud.

Competitive Advantage:

Give businesses a competitive edge by allowing them to adapt to changing market conditions, customer preferences, and emerging trends in real-time. This agility can help them stay ahead of competitors and capitalize on opportunities faster.

Real-time Testing:

In some industries, real-time data analysis is necessary for compliance with regulations and security standards. It ensures that sensitive customer information is handled and protected appropriately.

**Usage of tools:**

**1. Apache Airflow :** Apache Airflow provides a way to programmatically author, schedule, and monitor workflows, which are defined as Directed Acyclic Graphs (DAGs). used for orchestrating and scheduling complex data workflows and data pipeline automation

**2. Kafka :** Kafka is designed to handle real-time data feeds and stream processing in a highly scalable and fault-tolerant manner. It is commonly used in big data and real-time data processing applications and process and monitor data in distributed systems.

3.**Zookeeper :** Zookeeper Server for highly reliable distributed coordination of cloud applications. This is a service for distributed systems offering a hierarchical key-value store, which is used to provide a distributed configuration service, synchronization service and naming registry for large distributed systems

**4**. **Docker :** Docker enables you to package and distribute applications and their dependencies as containers, making it easier to develop, deploy, and manage software across different computing environments, such as development, testing, and production.

**5. Apache Spark :** Apache Spark is an open-source, distributed data processing framework that is designed for big data processing and analytics. It provides a fast and general-purpose cluster computing system for large-scale data processing tasks.

**6. Cassandra :** highly scalable, distributed NoSQL database management system designed for handling massive amounts of data across multiple commodity servers while ensuring high availability and fault tolerance.

**7. PostgreSQL :** a powerful, open-source relational database management system (RDBMS).

**8. Pyspark :** PySpark enables Python developers and data scientists to leverage the capabilities of Apache Spark for big data processing, analytics, and machine learning while using the Python programming language.

**Project output:**

* Setting up a data pipeline with Apache Airflow
* Streaming data with Kafka and Kafka Connect
* Using Zookeeper for distributed synchronization
* Data processing with Apache Spark
* Data storage solutions with Cassandra and PostgreSQL
* Containerizing your data engineering environment with Docker
* How to handle real time user data

**Steps involved in the project:**

* Setup Apache Airflow
* Install and set up IDE and Kafka
* Install and setup Spark and Pyspark
* Configure server using Zookeeper
* Gather data using API
* Monitor streaming data using Kafka
* Store the incoming data into Cassandra
* Dockerize the developed application and deploy at client location in cloud

**Project Execution Steps:**

Step 1: Set-up Zookeeper, Airflow, Kafka and Cassandra

* Use windows/Ubuntu OS
* Install Java and Pyspark and setup paths
* Set password for root user
* Install Kafka,spark and Install cassandra

Step 2: Data fetching

* Get data using API
* Setup data pipeline using Airflow
* Monitor data streaming using Kafka
* Get meta data information and print

Step3: Dockerize the application

* Install docker
* Create docket container
* Deploy at client location
* Clone the docker repository into client system

Step5: Project report

* Present installation and system set-up process
* Present data collection, loading and EDA
* Detailed steps for Dockerization
* Detailed steps to deploy at client location