**Project Name:** Real-Time Streaming Data Pipeline with Kafka and Docker

**Purpose:**

The goal of this project is to construct a real-time streaming data pipeline that uses Docker and Kafka to ingest, process, and store streaming data. The pipeline consists of parts for reading messages from a Kafka topic, analyzing the information, and creating messages and insights for new topics in Kafka.

**Components:**

* **Kafka:** Message broker for streaming data.
* **Zookeeper:** Coordination service for Kafka.
* **Python Kafka Consumer and Producer:** Processes the data and generates insights.
* **Docker Compose:** Manages the deployment of Kafka, Zookeeper, and the data generator.

**Instructions on How to Set Up and Run the Docker Environment**

**Prerequisites:**

* Docker and Docker Compose installed on your machine.

**Steps:**

1. **Clone the Repository:**

Git clone <https://github.com/VidyaranyaRJ/Real-time-Streaming-Data-Pipeline-with-Kafka.git>

1. **Start the Docker Containers:**

docker-compose up -d

1. **to check the containers:**

docker ps

1. **Run** python .\producer\_consumer.py
2. **Verify Kafka Topics**:

docker exec -it fetch\_take\_home-kafka-1 kafka-topics --list --bootstrap-server localhost:9092

Note:

To check the kafka topics and print the contents of the kafka topics

docker exec -it fetch\_take\_home-kafka-1 kafka-topics --list --bootstrap-server localhost:9092

**processed-user-login Topic**-

docker exec -it fetch\_take\_home-kafka-1 kafka-console-consumer --topic processed-user-login --bootstrap-server localhost:9092 --from-beginning

**user-login Topic -**

docker exec -it fetch\_take\_home-kafka-1 kafka-console-consumer --topic user-login --bootstrap-server localhost:9092 --from-beginning

**user-login-insights Topic –**

docker exec -it fetch\_take\_home-kafka-1 kafka-console-consumer --topic user-login-insights --bootstrap-server localhost:9092 --from-beginning

**Explanation of Kafka Consumer Logic**

**process\_data Function**:

Count Logins by Device Type and App Version: Increases the count for every app version and device type found in the incoming data.

* Analyze Login Frequency by Locale: Increments the count for each locale.
* Identify Unusual Login Patterns
* Converts the timestamp to a datetime object.
* Appends the current login to the user\_logins dictionary.
* Removes login records older than the defined time window.
* Checks for unusual login patterns (multiple IPs within the time window).
* Add Processed Information to Data: Flags the data as processed and includes the unusual login status.
* Generate Insights: Creates a dictionary of insights to be sent to the insights Kafka topic.

**Consumer and Producer Logic:**

* Kafka Consumer: Listens to the user-login topic and deserializes incoming messages.
* Kafka Producer: Produces messages to the processed-user-login and user-login-insights topics.
* Processing Loop: Processes each message, generates insights, and sends the processed data and insights to their respective topics.
* Error Handling: Catches and logs any exceptions that occur during processing.