Assignment Overview

The aim of this assignment is to create a **real-time** data pipeline for processing e-commerce data using Apache Kafka and Apache Cassandra. You will ingest data from a CSV file using a Kafka producer, transform the data using a Kafka consumer, and finally store the processed data in a Cassandra table.

Problem Statement

Imagine you're part of the data engineering team at a large e-commerce company. Your company receives real-time data related to customer orders. This data is crucial for various business operations, including inventory management, customer service, and business analytics.

The data is received in CSV format and needs to be processed in real-time to extract valuable insights.

Assignment Steps

1. **Dataset**
   Load the '**olist_orders_dataset.csv**' into a pandas dataframe and examine its structure and contents.

2. **Apache Kafka Setup**
   Install and configure Apache Kafka on your system. Create a Kafka topic, named '**ecommerce-orders**', to hold the e-commerce data.

3. **Kafka Producer**
   Develop a Kafka producer in Python that reads the data from the pandas dataframe and publishes it to the **'ecommerce-orders'** Kafka topic. The key for each message should be a combination of the **'customer_id'** and '**order_id**' fields from the dataset.

4. **Apache Cassandra Setup**
   Install and set up Apache Cassandra. Create a keyspace, named **'ecommerce'**, for storing the e-commerce data.

5. **Cassandra Data Model**
   Design a table, named **'orders'**, within the **'ecommerce'** keyspace. This table should reflect the schema of the incoming data and include additional columns for the derived features: **'OrderHour'** and **'OrderDayOfWeek'**. The data model should have **'customer_id'** as the partition key and **'order_id'** and **'order_purchase_timestamp'** as clustering keys.

```
CREATE TABLE ecommerce.orders (
    order_id uuid,
    customer_id uuid,
    order_status text,
    order_purchase_timestamp timestamp,
    order_approved_at timestamp,
    order_delivered_carrier_date timestamp,
    order_delivered_customer_date timestamp,
    order_estimated_delivery_date timestamp,
    OrderHour int,
    OrderDayOfWeek text,
    PRIMARY KEY ((customer_id), order_id,
order_purchase_timestamp));
```

6. **Kafka Consumer and Data Transformation**
   Develop a Kafka consumer (make sure to have a consumer group) in Python that subscribes to the **'ecommerce-orders'** topic. The consumer should derive two new columns **'PurchaseHour'** and **'PurchaseDayOfWeek'**, then ingest transformed data into the **'orders'** table in Cassandra.

7. **Quorum Consistency**
   While inserting data into the Cassandra 'orders' table, ensure that the write operations maintain quorum consistency.

8. **Testing**
   Test your data pipeline end-to-end. Run your Kafka producer to ingest the data, then execute the Kafka consumer to process the

data and insert it into the Cassandra table. Verify the data in the Cassandra table matches the processed data and that all transformations have been executed correctly.

9. **Assignment Submission**
   Submit your Python scripts for the Kafka producer and consumer, the CQL commands used to create the keyspace and table in Cassandra, and a detailed report explaining the pipeline, any challenges faced, and how they were addressed.