Code Logic - Retail Data Analysis

The purpose of the project was to compute various **Key Performance Indicators (KPIs)** for an e-commerce company, Retail Corp Inc.

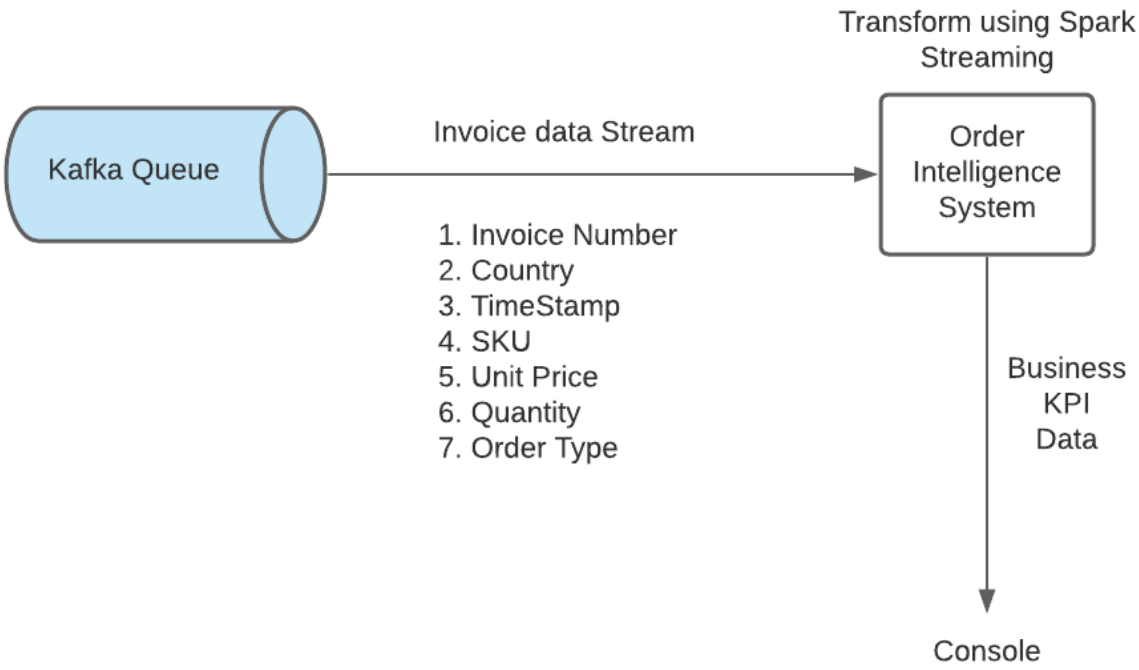

Fig1. **Architecture of project**

The data is based on an Online Retail Data Set in the UCI Machine Learning Repository. The data contains the following information in JSON Format:

1. **Invoice number**: Identifier of the invoice
2. **Country**: Country where the order is placed
3. **Timestamp**: Time at which the order is placed
4. **Type**: Whether this is a new order or a return order
5. **SKU (Stock Keeping Unit)**: Identifier of the product being ordered
6. **Title**: Name of the product is ordered
7. **Unit price**: Price of a single unit of the product
8. **Quantity**: Quantity of the product being ordered

Steps followed during project :

1. Reading the sales data from the Kafka server hosted centrally
   a. Command to consume raw Stream data from Kafka server

```
bin/zookeeper-server-start.sh config/zookeeper.properties

bin/kafka-server-start.sh config/server.properties

bin/kafka-console-consumer.sh --bootstrap-server \
18.211.252.152:9092 --topic real-time-project
```

   b. As the Cloudera Distribution was used for spark streaming following command was executed:

```
export SPARK_KAFKA_VERSION=0.10
```

2. Pre-processing the data to calculate additional derived columns
   a.The following attributes from the raw Stream data have to be taken into account for the project:
   - **invoice_no**: Identifier of the invoice
   - **country**: Country where the order is placed
   - **timestamp**: Time at which the order is placed

   In addition to these attributes, the following **UDFs** were calculated and added to the table:
   - **total_cost**: Total cost of an order arrived at by summing up the cost of all products in that invoice (The return cost is treated as a loss. Thus, for return orders, this value would be negative.)
   - **total_items**: Total number of items present in an order
   - **is_order**: This flag denotes whether an order is a new order or not. If this invoice is for a return order, the value should be 0.
   - **is_return**: This flag denotes whether an order is a return order or not. If this invoice is for a new sales order, the value should be 0.

   b.The input table be generated for each **one-minute window**, hence the trigger was set to “1 minute”.

   c. Schema of a single order (JSON format) was defined

   d. Code to define the aforementioned UDFs and any utility functions were written to calculate them

3. Calculating the time-based KPIs and time and country-based KPIs
   a. **Total volume of sales**:
      Total transaction value of a specific order=
      `∑(quantity * unitprice)`

      Total volume of sales =
      `∑Order(quantity * unitprice) – ∑Return(quantity * unitprice)`

   b. **OPM (orders per minute)**:
      Total number of orders received in a minute= `count(invoice_no)`

   c. **Rate of return**:
      `∑Returns/ (∑Returns+∑Orders)`

   d. **Average transaction size**:
      `Total Sales Volume/(∑Returns+∑Orders)`

4. Stored the KPIs (both time-based and time- and country-based) for a 10-minute interval into separate JSON files in HDFS for further analysis

5. Spark streaming command for execution of python script:

```
spark2-submit --jars spark-sql-kafka-0-10_2.11-2.3.0.jar spark-
streaming.py 18.211.252.152 9092 real-time-project
```