1. I fetch streaming data using Kafka and the sample code provided for the same.

2. I parse these incoming values and then pass them to the POJO class TransactionData.

3. Then I pass the object of TransactionData class to HbaseDAO, which is used to fetch the data from Hbase tables.

4. Inside HbaseDAO we first establish the connection to HBase using the class HbaseConnection.

5. Once the connection is established, I fetch the data of my hbase lookup table tmp_hive_lookup_table. I pass the values of ucl, score, zipcode and txn_dt for the incoming card_id from the stream.

6. These values are passed as an object to my main class.

7. Next I use the Distance utility to calculate the distance between last zipcode value and current value. After calculating distance, I calculate time difference in seconds between the last time of transaction and the current one. Then I calculate the speed in km/sec and use an imaginable speed of 0.25km/sec as a criteria to classify a transaction as GENUINE or FRAUD.

8. Now that I have the 3 values- speed, ucl and score I use below line of code to make the comparisions:

```
        if(current_speed >= 0.25 || amount > lookup_ucl || lookup_score < 200) {

                flag = false;

                System.out.println("FRAUD");

        }

        else {

                 flag = true;

                 System.out.println("GENUINE");


                 //if transaction is genuine, update lookup table


                  hbaseDao.updateLookupData(transactionData);
        }
```

If the transaction is GENUINE, only then the lookup table is updated. updateLookupData is another method in HbaseDAO class.

9. Next I use the same flag variable to update the card_transactions table data with GENUINE or FRAUD status.

```
        if(flag) {
                    CardTxnsPojo cardtxnsdata = new CardTxnsPojo(card_id, member_id, amount, postcode, pos_id, transaction_dt, "GENUINE");
                    cardTxns.updateCardTxns(cardtxnsdata);
                }
                else {
                      CardTxnsPojo cardtxnsdata = new CardTxnsPojo(card_id, member_id, amount, postcode, pos_id, transaction_dt, "FRAUD");
                      cardTxns.updateCardTxns(cardtxnsdata);
                  }
```

CardTxnsPojo is another POJO class to instantiate the values of latest transaction.

CardTransactions is the class used to finally update the Hbase table card_transactions_hive_new.

Command used execute the logic in ec2:

```
spark2-submit --class streamAnalysis.KafkaSparkStreaming --deploy-mode client --name KafkaSparkStreamingDemo /home/ec2-user/FinalSubmissionCapstone-0.0.1-SNAPSHOT.jar 34.205.77.177 &> /tmp/output.txt
```