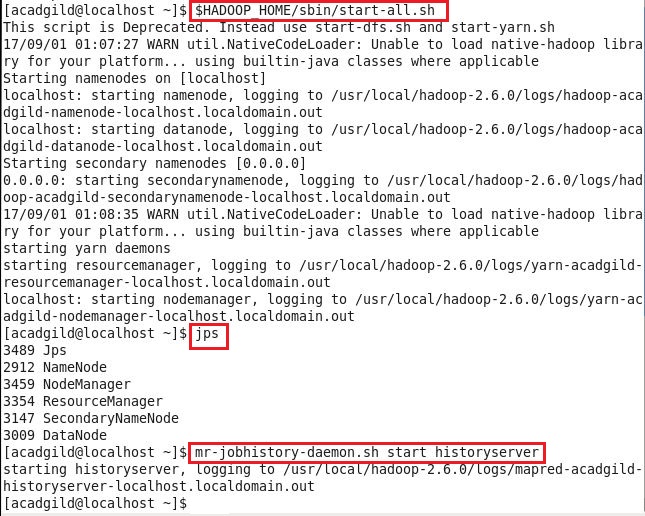
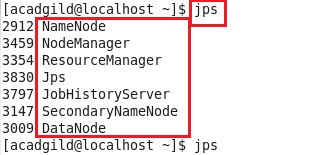
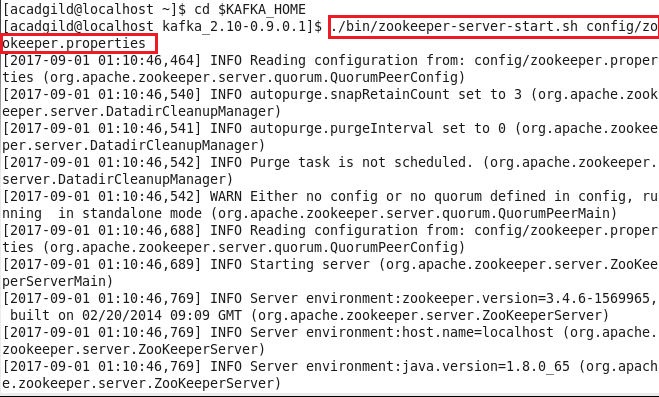
**Stateful Streaming in Apache Spark**

**Step 1: Started Hadoop Daemons and job history server**

**Step 2: Using “jps” command, checked whether required processes started or not**

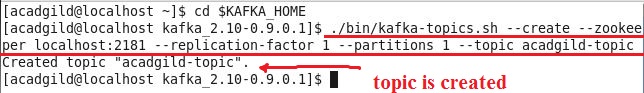
****

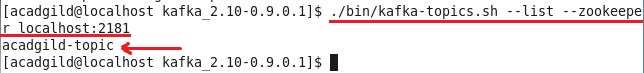
**Step 3: Started zookeeper server in Kafka by navigating into $KAFKA\_HOME with the command given below:**

****

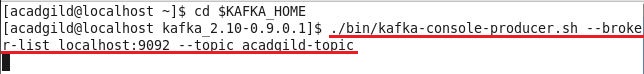
**Step 4: In new terminal, started the Kafka broker using the following command:**

****

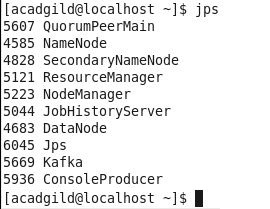
**Step 5:** **After starting, kept both the terminals running, opened a new terminal, and created a Kafka topic with the following command:**

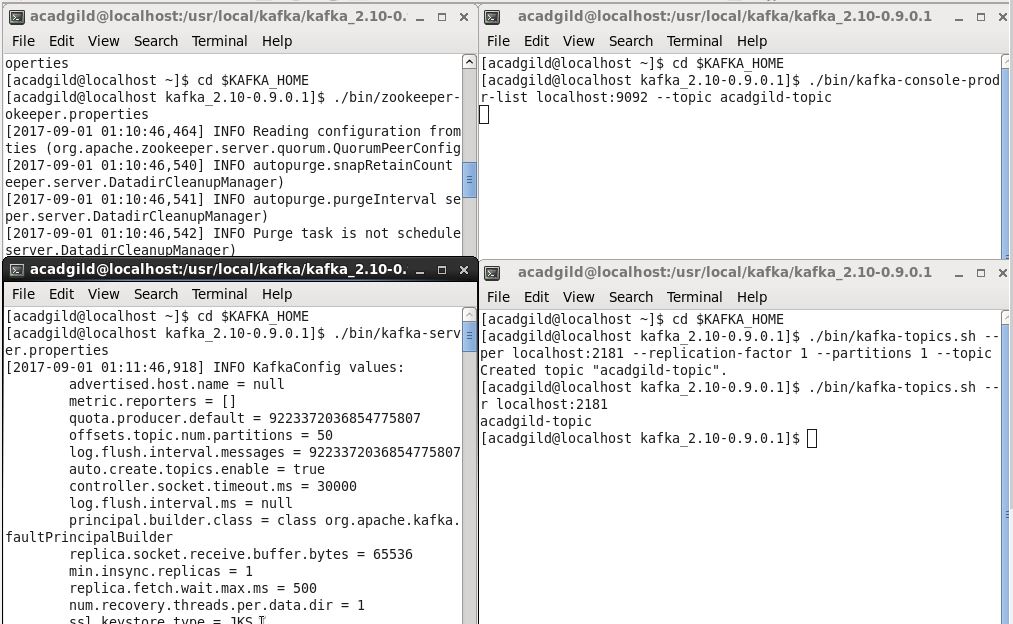
****

**Using below command, checked the topic list:**

**Step 6: Now for sending messages to this topic, used the following command to start the console producer:**

**Step 7: Using jps command, checked status of all processes, whether started or not:**

****

**Step 8: All the 4 consoles are shown below:**

**Step 9: Created below code in IntelliJ Idea IDE:**

import org.apache.spark.{ SparkConf, SparkContext }

import org.apache.spark.streaming.StreamingContext

import org.apache.spark.streaming.Seconds

import org.apache.spark.streaming.dstream.DStream

import org.apache.spark.rdd.RDD

import org.apache.spark.streaming.{ State, StateSpec }

import org.apache.spark.streaming.kafka010.KafkaUtils

import org.apache.kafka.common.serialization.StringDeserializer

import org.apache.kafka.clients.consumer.ConsumerRecord

import org.apache.spark.streaming.kafka010.LocationStrategies.PreferConsistent

import org.apache.spark.streaming.kafka010.ConsumerStrategies.Subscribe

object stateFulWordCount {

  def main(args: Array[String]) {

    val conf = new SparkConf().setMaster("local[\*]").setAppName("KafkaReceiver")

    val ssc = new StreamingContext(conf, Seconds(10))

    /\*

     \* Defingin the Kafka server parameters

     \*/

    val kafkaParams = Map[String, Object](

      "bootstrap.servers" -> "localhost:9092,localhost:9092",

      "key.deserializer" -> classOf[StringDeserializer],

      "value.deserializer" -> classOf[StringDeserializer],

      "group.id" -> "use\_a\_separate\_group\_id\_for\_each\_stream",

      "auto.offset.reset" -> "latest",

      "enable.auto.commit" -> (false: java.lang.Boolean))

    val topics = Array("acadgild-topic") //topics list

    val kafkaStream = KafkaUtils.createDirectStream[String, String](

      ssc,

      PreferConsistent,

      Subscribe[String, String](topics, kafkaParams))

    val splits = kafkaStream.map(record => (record.key(), record.value.toString)).flatMap(x => x.\_2.split(" "))

    val updateFunc = (values: Seq[Int], state: Option[Int]) => {

      val currentCount = values.foldLeft(0)(\_ + \_)

      val previousCount = state.getOrElse(0)

      Some(currentCount + previousCount)

    }

    //Defining a check point directory for performing stateful operations

    ssc.checkpoint("hdfs://localhost:9000/WordCount\_checkpoint")

    val wordCounts = splits.map(x => (x, 1)).reduceByKey(\_+\_).updateStateByKey(updateFunc)

    kafkaStream.print() //prints the stream of data received

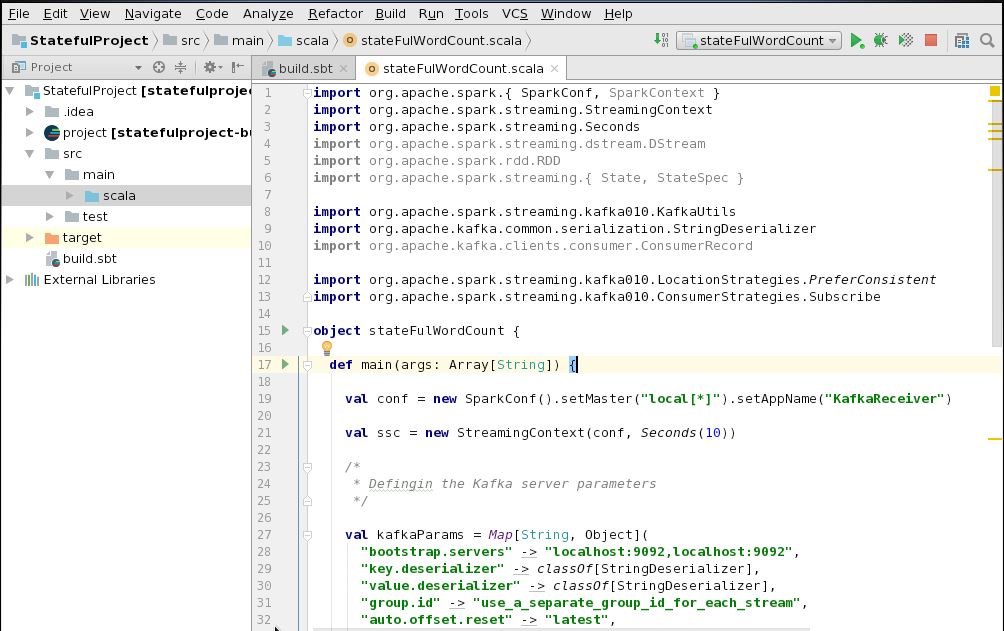
    wordCounts.print() //prints the wordcount result of the stream

    ssc.start()

    ssc.awaitTermination()

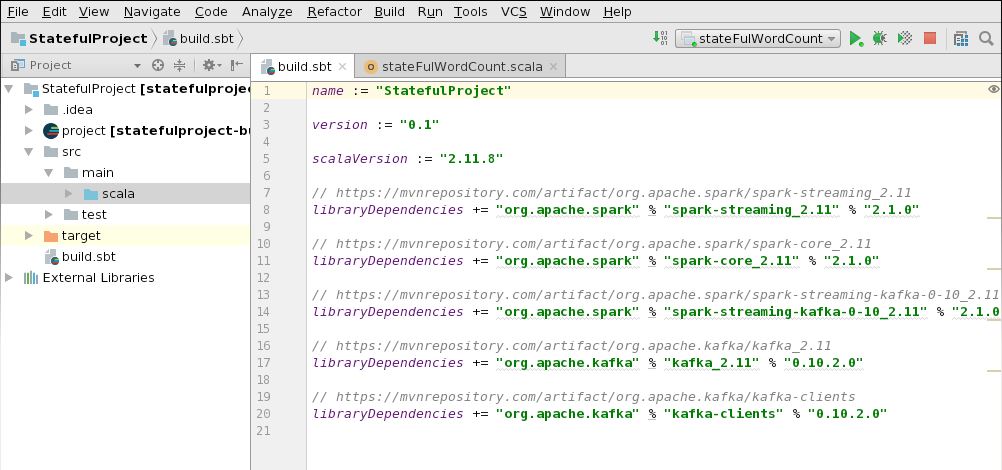
  }

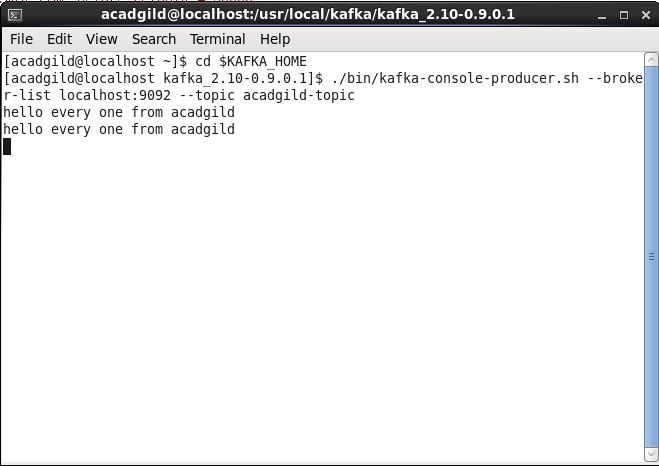
}



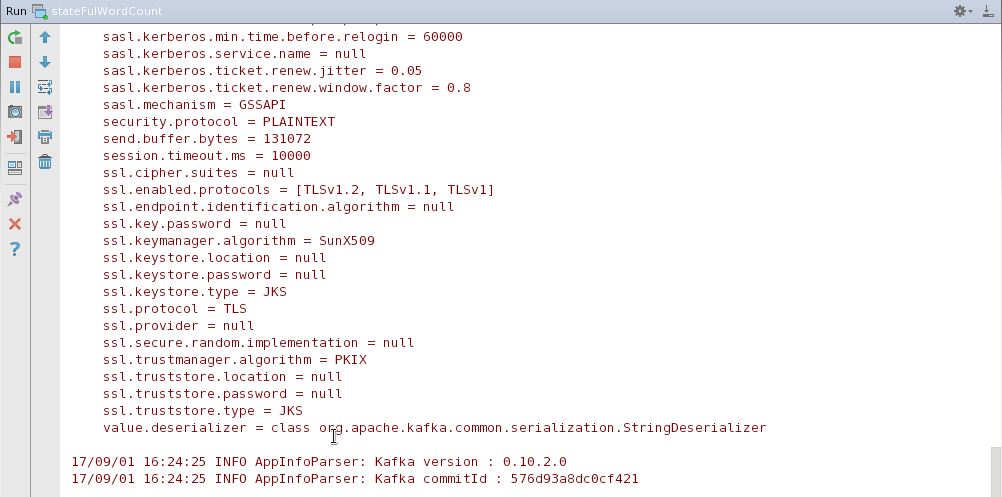
**Below is the screenshot for the code created inside intellij idea IDE:**

**Step 10: Below is the build.sbt file:**

****

**Step 11: In console producer window, typed below lines but intellij idea console didn’t show count for the words:**

**Step 12: No count of words is shown in intellij idea console:**

****