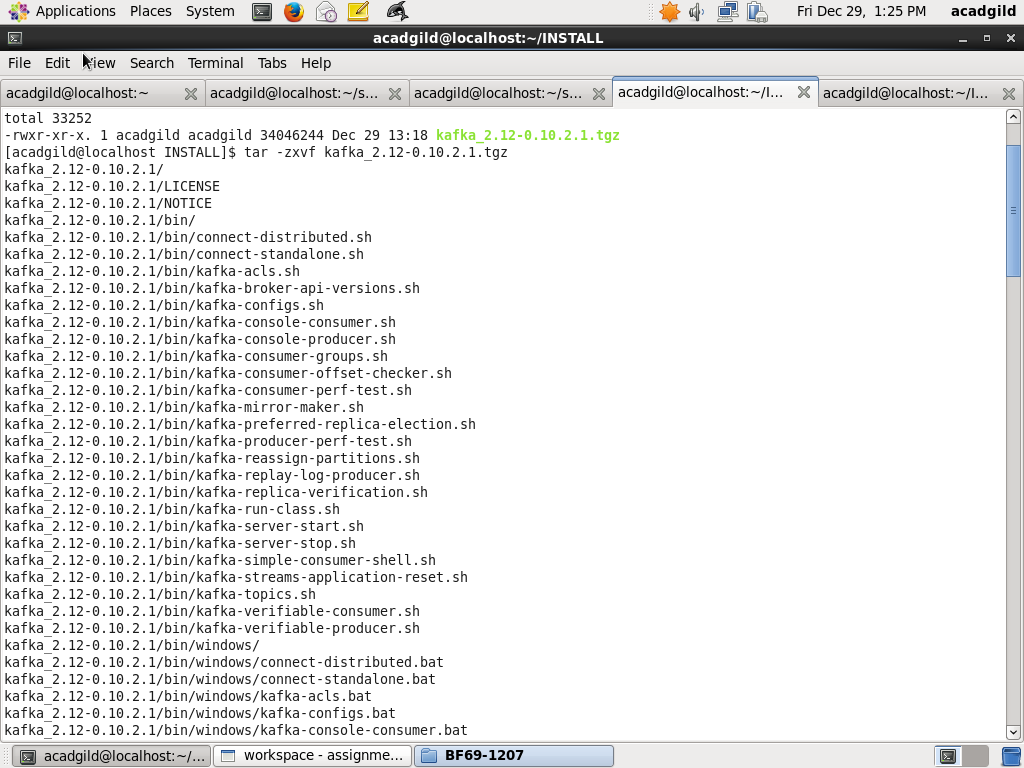
Assignment 22.2

Step1: Install Kafka

Download kafka\_2.12-0.10.2.1.tgz fie using

tar –zxvf kafka\_2.12-0.10.2.1.tgz

Screenshot is as below:

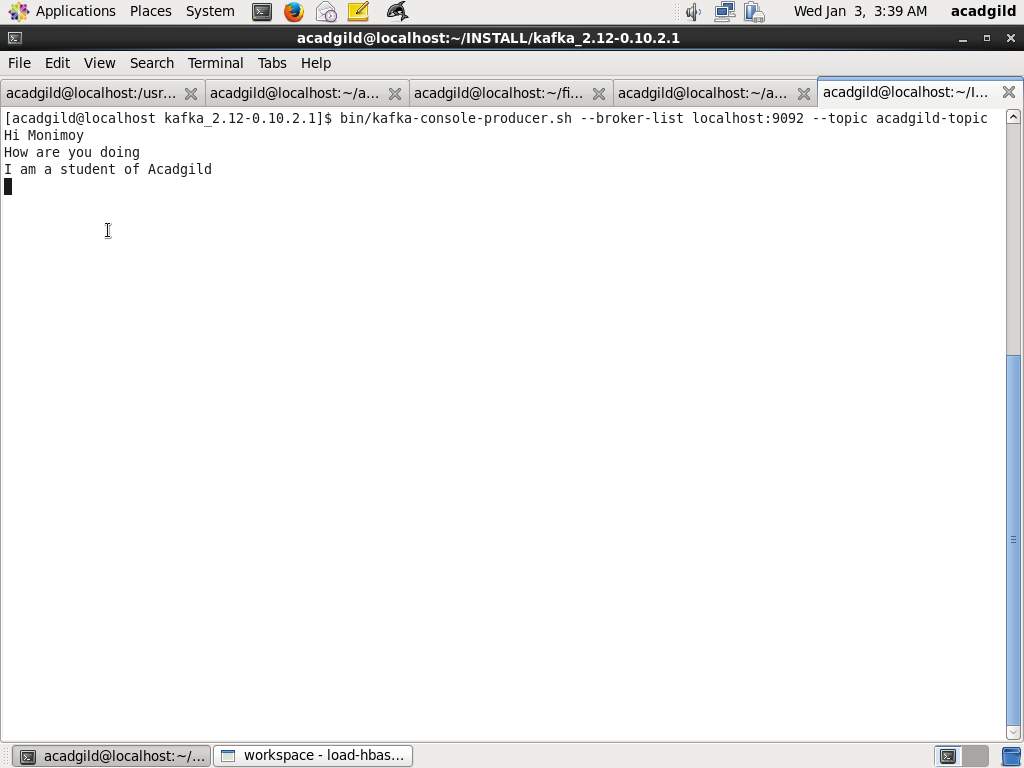


Step2: Start Kafka

Using kafka from the directory where it was installed using command below:

bin/kafka-console-producer.sh -broker-list localhost:9092 –topic acadgild-topic

Screenshot is as below:



Step3: Write code for kafka spark streaming

Step3a: Define package as WordCount and all the imports

package WordCount

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

Step3b: Define object stateFulWordCount, main method and Streaming context

object stateFulWordCount {

def main(args: Array[String]) {

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

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

Step3c: Define Kafka parameters

* Server as localhost, port as 9092
* Serializer, Deserializer, group.id

/\*

\* 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))

Step3d: Define topic as acadgild-topic

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

Step3e: Create kafka steam

* Create kafka streaming context with parameters StreamingContext , PreferConsistent, Subscribe

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

ssc,

PreferConsistent,

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

Step 3f: Split the records to a list of words

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

Step 3f: Define updateFunc

Define updateFunc which add the word count by adding to count stored earlier

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

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

val previousCount = state.getOrElse(0)

Some(currentCount + previousCount)

}

Step 3f:Define checkpoint directory for performing stateful operations

//Defining a check point directory for performing stateful operations

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

Step 3f:Map and reduceByKey to calculate word count per word

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

Step 3f:Map and reduceByKey to calculate word count per word

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

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

Step 3g: Start the streaming context and wait for its termination

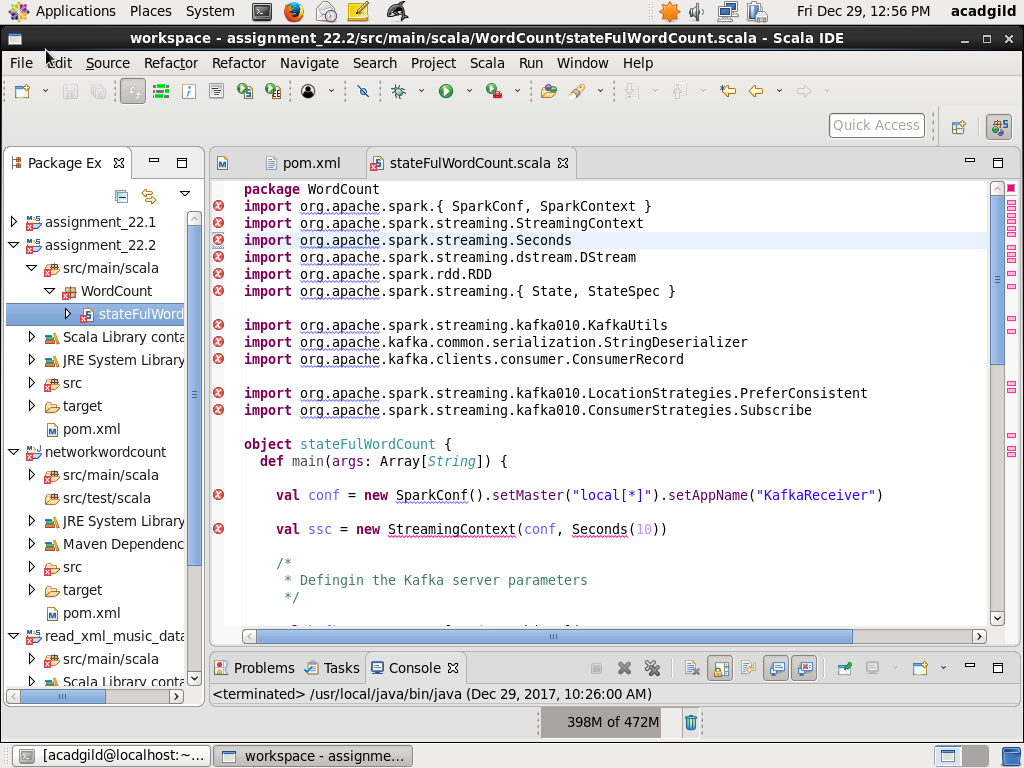
ssc.start()

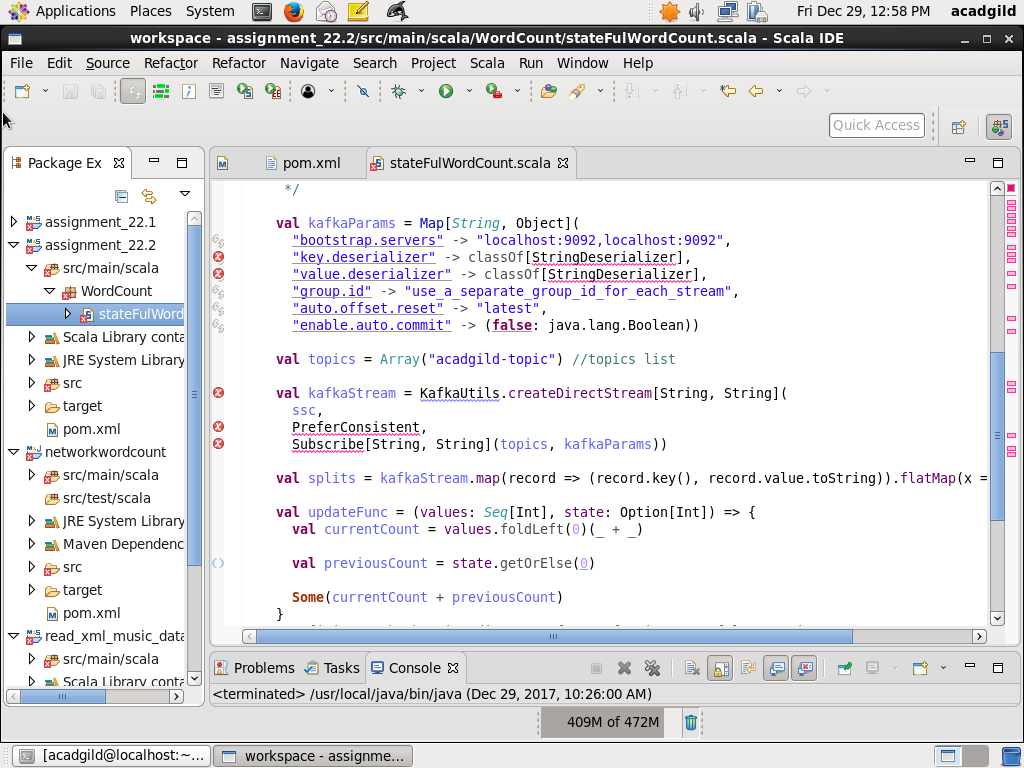
ssc.awaitTermination()

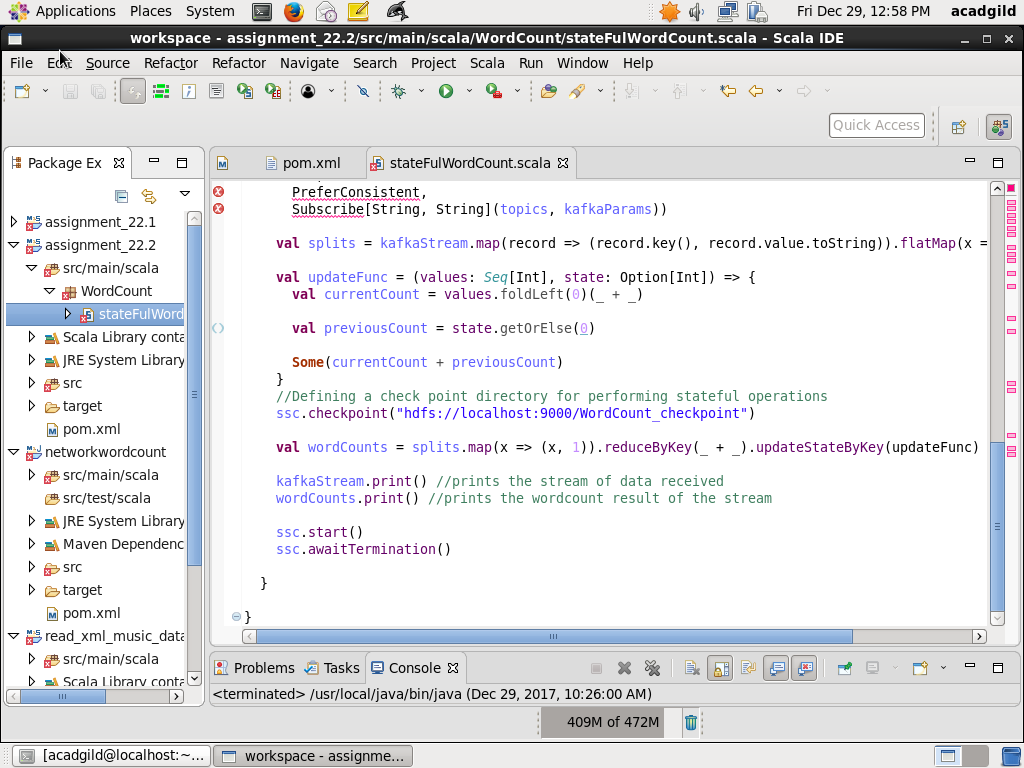
}

}

Screenshot is as below:







Step4: Define pom.xml file to define dependencies

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>assignment\_22.2</groupId>

<artifactId>network\_stateful\_wordcount</artifactId>

<version>0.0.1-SNAPSHOT</version>

<properties>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

<encoding>UTF-8</encoding>

<scala.version>2.10.5</scala.version>

</properties>

<build>

<sourceDirectory>src/main/scala</sourceDirectory>

<resources>

<resource>

<directory>src/main/scala</directory>

<excludes>

<exclude>\*\*/\*.java</exclude>

</excludes>

</resource>

</resources>

<plugins>

<plugin>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.5.1</version>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

<plugin>

<groupId>org.scala-tools</groupId>

<artifactId>maven-scala-plugin</artifactId>

<version>2.15.2</version>

<executions>

<execution>

<goals>

<goal>compile</goal>

</goals>

</execution>

</executions>

</plugin>

</plugins>

</build>

<dependencies>

<dependency>

<groupId>org.apache.spark</groupId>

<artifactId>spark-streaming\_2.11</artifactId>

<version>2.1.0</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.apache.spark/spark-core\_2.11 -->

<dependency>

<groupId>org.apache.spark</groupId>

<artifactId>spark-core\_2.11</artifactId>

<version>2.1.0</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.apache.spark/spark-streaming-kafka-0-10\_2.11 -->

<dependency>

<groupId>org.apache.spark</groupId>

<artifactId>spark-streaming-kafka-0-10\_2.11</artifactId>

<version>2.1.0</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.apache.kafka/kafka\_2.11 -->

<dependency>

<groupId>org.apache.kafka</groupId>

<artifactId>kafka\_2.11</artifactId>

<version>0.10.2.0</version>

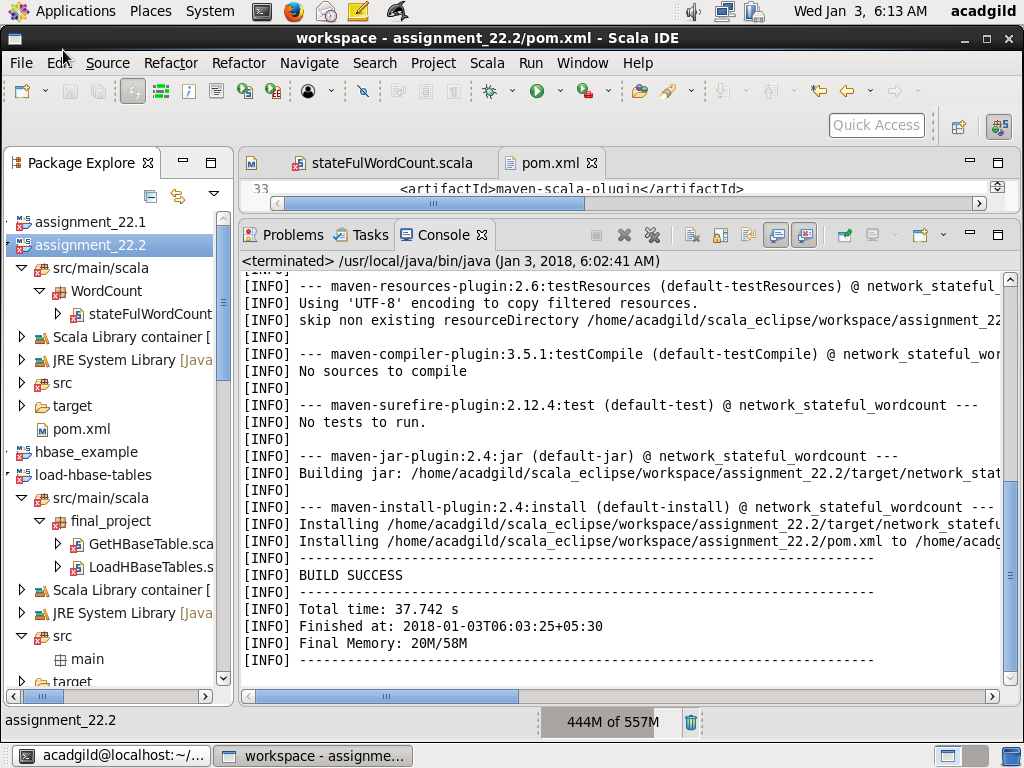
</dependency>

</dependencies>

</project>

Step5: Perform Maven install

Perform maven install to build the packages



Step6: Start a spark job with spark-submit

Start a spark job and submit it using the following command:

bin/spark-submit --class WordCount.stateFulWordCount --master local[2] --packages org.apache.spark:spark-streaming-kafka-0-10\_2.11:2.1.0,org.apache.kafka:kafka\_2.11:0.10.2.0,org.slf4j:slf4j-log4j12:1.7.21,org.slf4j:slf4j-api:1.7.21 /home/acadgild/scala\_eclipse/workspace/assignment\_22.2/target/network\_stateful\_wordcount-0.0.1-SNAPSHOT.jar

The screenshots are as below

