Assignment 25:

Give your own input in output-screenshot of report.
Task 1 As discussed in class integrate Spark Hive Task
Steps to integrate Spark hive:

---

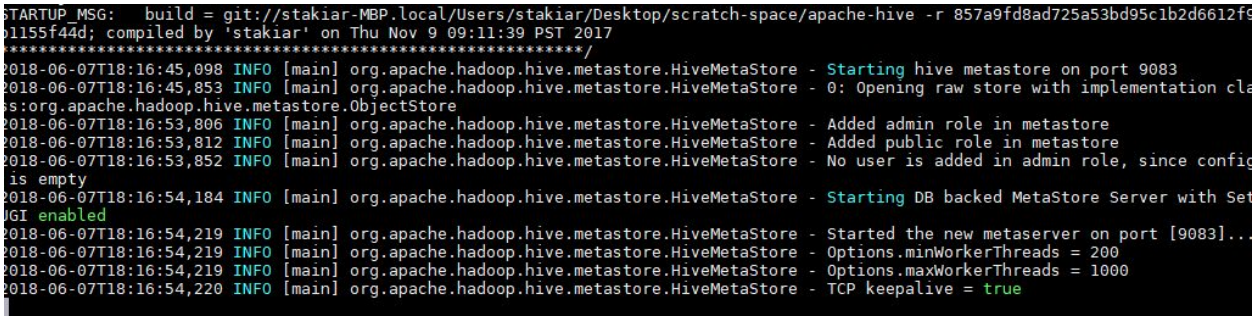
2. Copy the hive-site.xml file from $HIVE_HOME/conf to $SPARK_HOME/conf
3. Add the following properties to hive-site.xml on spark side :

- <property>
- <name>hive.metastore.uris</name>
- <value>thrift://localhost:9083</value>
- <description>password for connecting to mysql server</description>
- </property>

4. Write the code in Scala IDE to list the Databases in the hive. Source code is uploaded separately.
5. Make sure that your hadoop is started
6. Start hive metastore by executing hive —service metastore command.
7. Run the code from the IDE. You should be able to see all the hive databases.

Output Screen Shot:

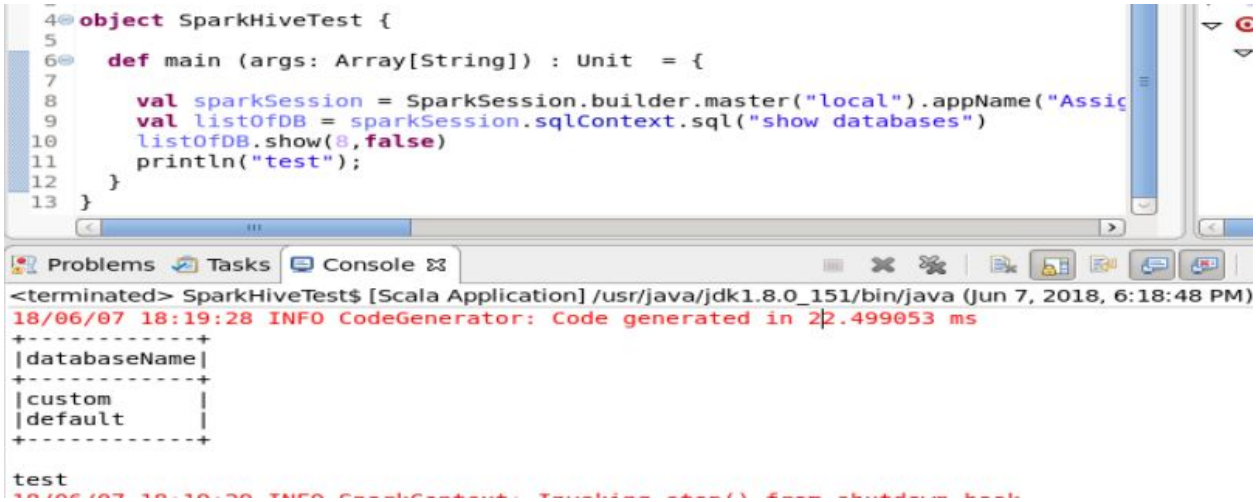
Starting metastore:



Hive database in the terminal:

```
hive> show databases;
OK
custom
default
Time taken: 0.04 seconds, Fetched: 2 row(s)
hive>
```

Output after running the code in Scala Ide



2. As discussed in class integrate Spark Hbase Task

Steps:

1.Write an API code in scala ide to create a new table in hbase. Source code is uploaded separately.

2. Run the code in scala ide and check the hbase for the newly created table.

3. Make sure to start the hbase shell using the below commands

Start-hbase.sh

Hbase shell

Output Screenshots:

List of tables before running the code:

```
hbase(main):001:0> list
TABLE
SparkHBasesTable
TRANSACTIONS
bulktable
clicks
4 row(s) in 0.3620 seconds

=> ["SparkHBasesTable", "TRANSACTIONS", "bulktable", "clicks"]
```

List of tables after running the code:

Newly created table highlighted in red

```
hbase(main):001:0> list
TABLE
SparkHBasesTable
TRANSACTIONS
bulktable
clicks
4 row(s) in 0.3620 seconds

=> ["SparkHBasesTable", "TRANSACTIONS", "bulktable", "clicks"]
hbase(main):002:0> list
TABLE
SparkHBasesTable
SparkHBasesTable1
TRANSACTIONS
bulktable
clicks
5 row(s) in 0.0160 seconds

=> ["SparkHBasesTable", "SparkHBasesTable1", "TRANSACTIONS", "bulktable", "clicks"]
```

Console output:

```
creating table:SparkHBasesTable1        18/06/07 18:38:57 INFO HBaseAdmin: Created SparkHBasesTable1
Data Entered In TableData Entered In TableData Entered In TableData Entered In TableData Entered In TableDat
```

Newly created table with column family, column and value

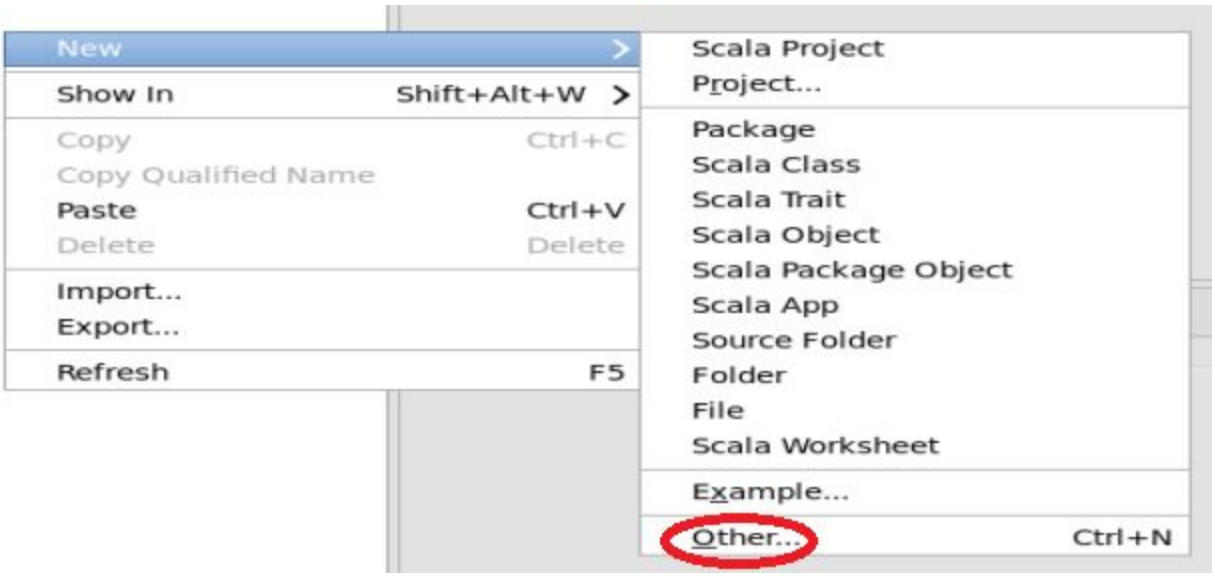
```
hbase(main):004:0> scan 'SparkHBasesTable1'
ROW                                   COLUMN+CELL
 row1                                 column=cf:column, timestamp=1528376937645, value=value1
 row10                                column=cf:column, timestamp=1528376937716, value=value10
 row2                                 column=cf:column, timestamp=1528376937673, value=value2
 row3                                 column=cf:column, timestamp=1528376937679, value=value3
 row4                                 column=cf:column, timestamp=1528376937683, value=value4
 row5                                 column=cf:column, timestamp=1528376937688, value=value5
 row6                                 column=cf:column, timestamp=1528376937694, value=value6
 row7                                 column=cf:column, timestamp=1528376937701, value=value7
 row8                                 column=cf:column, timestamp=1528376937705, value=value8
 row9                                 column=cf:column, timestamp=1528376937711, value=value9
10 row(s) in 0.4000 seconds
```

3 As discussed in class integrate Spark Kafka

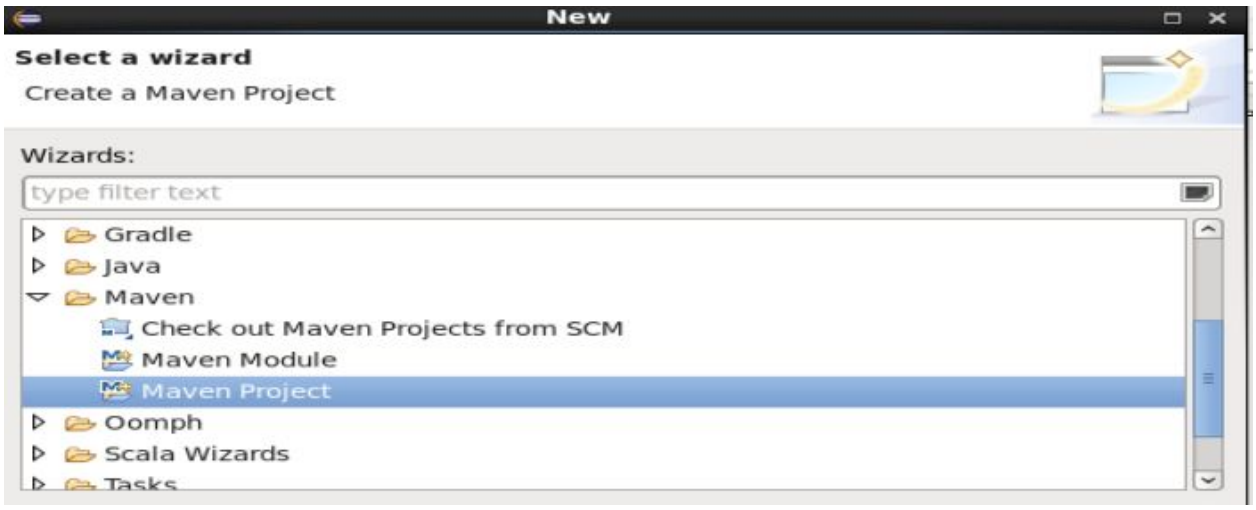
Steps:

Create a Maven project as given in the below screenshots:

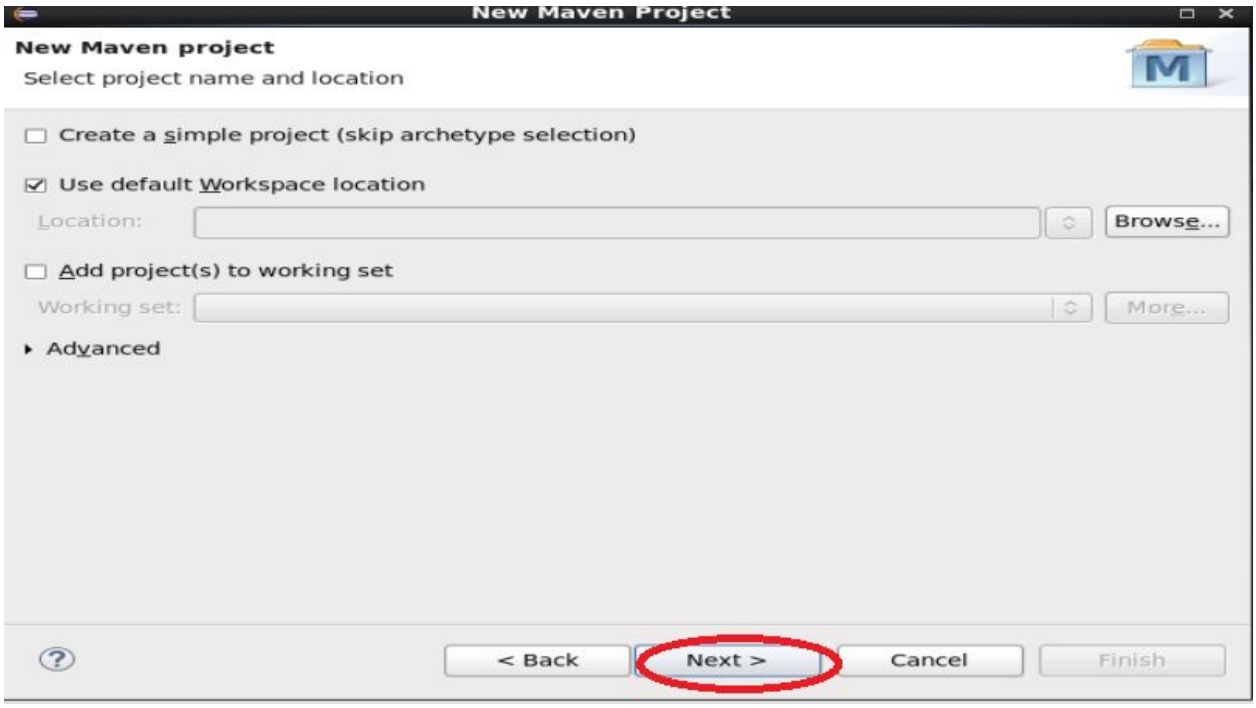
Right click on the package explorer and select others as below:



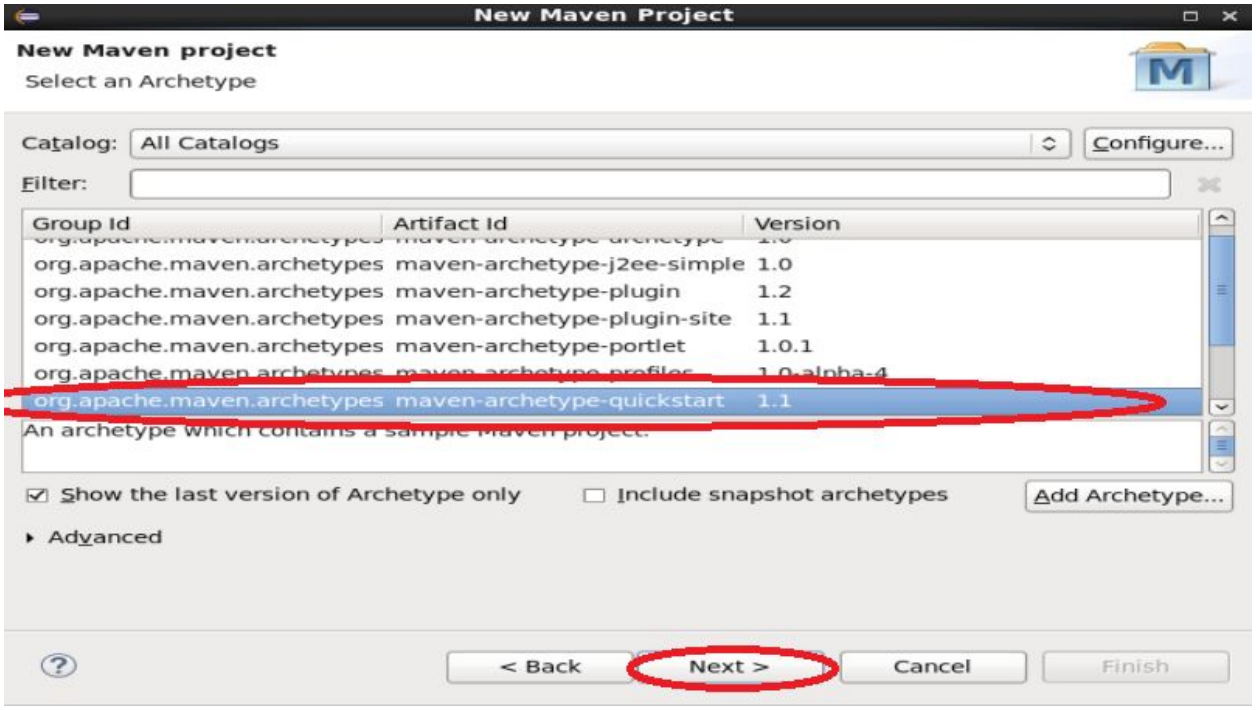
Select maven project as shown below:



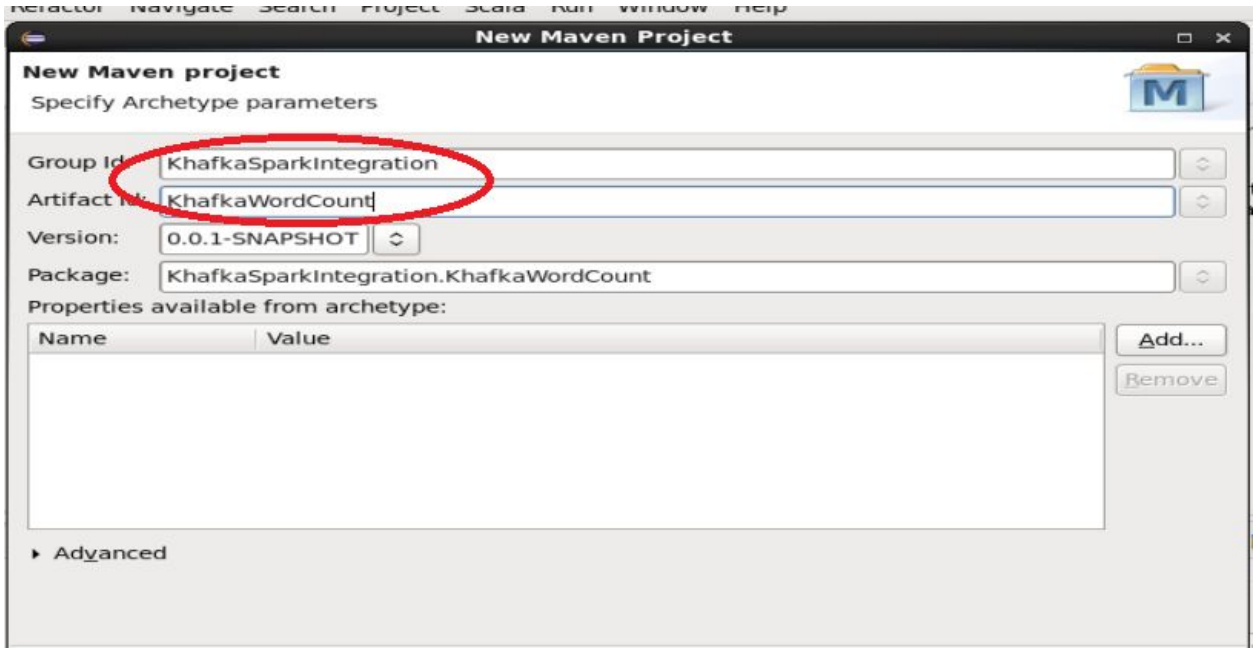
In the new Maven project wizard select next as below



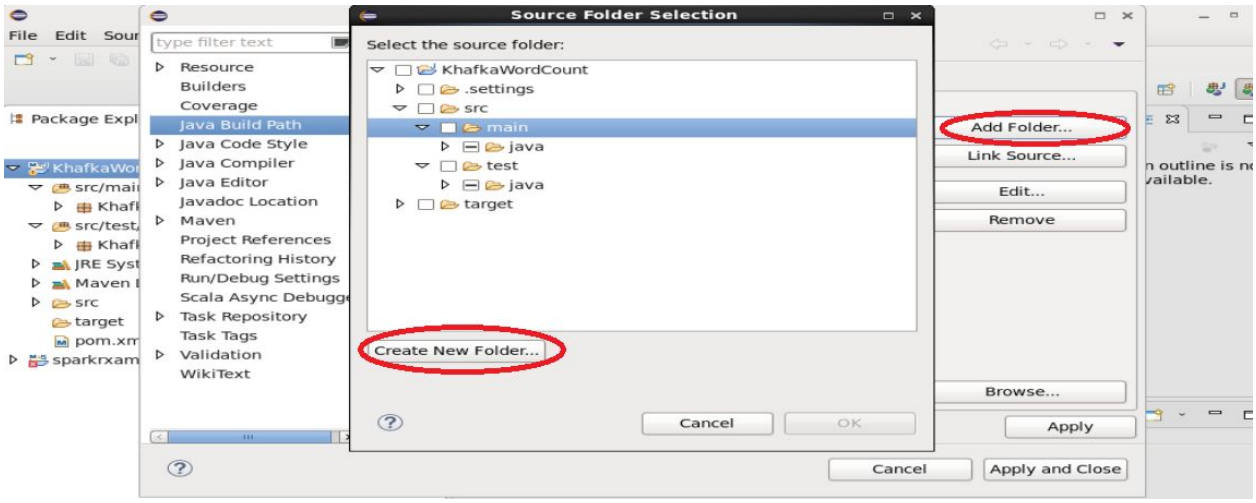
Select the below option highlighted in red and click next:



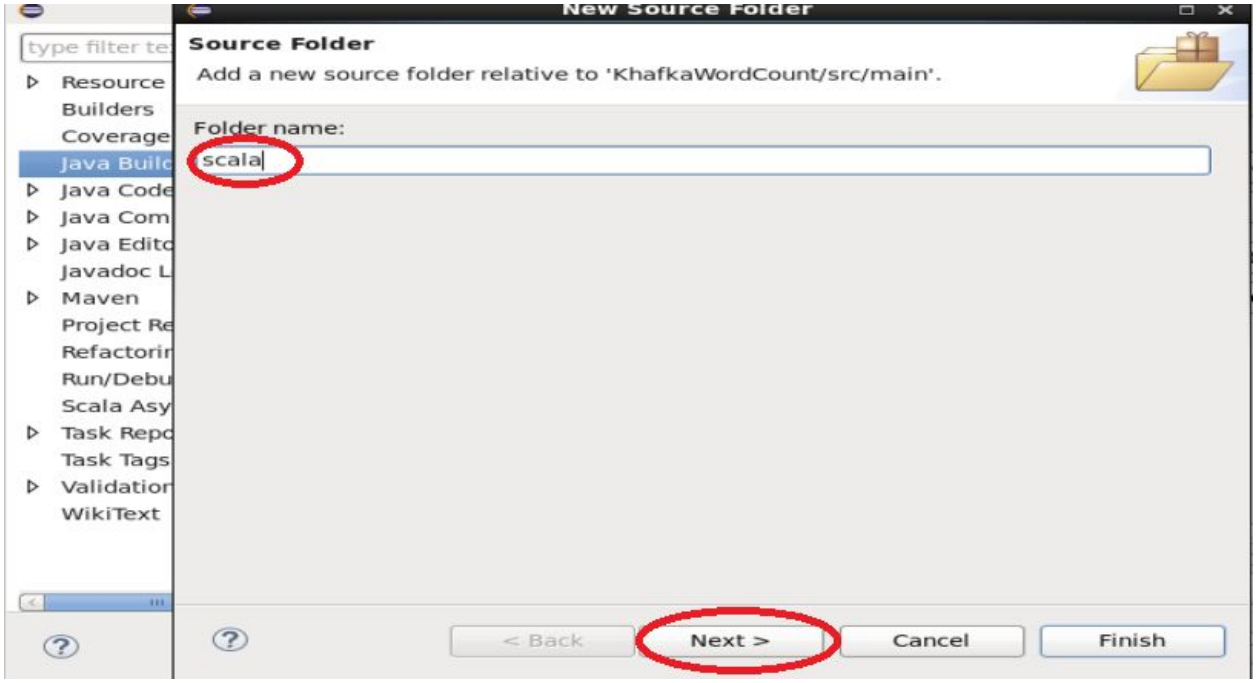
Provide the GroupId and Artifact Id as given below



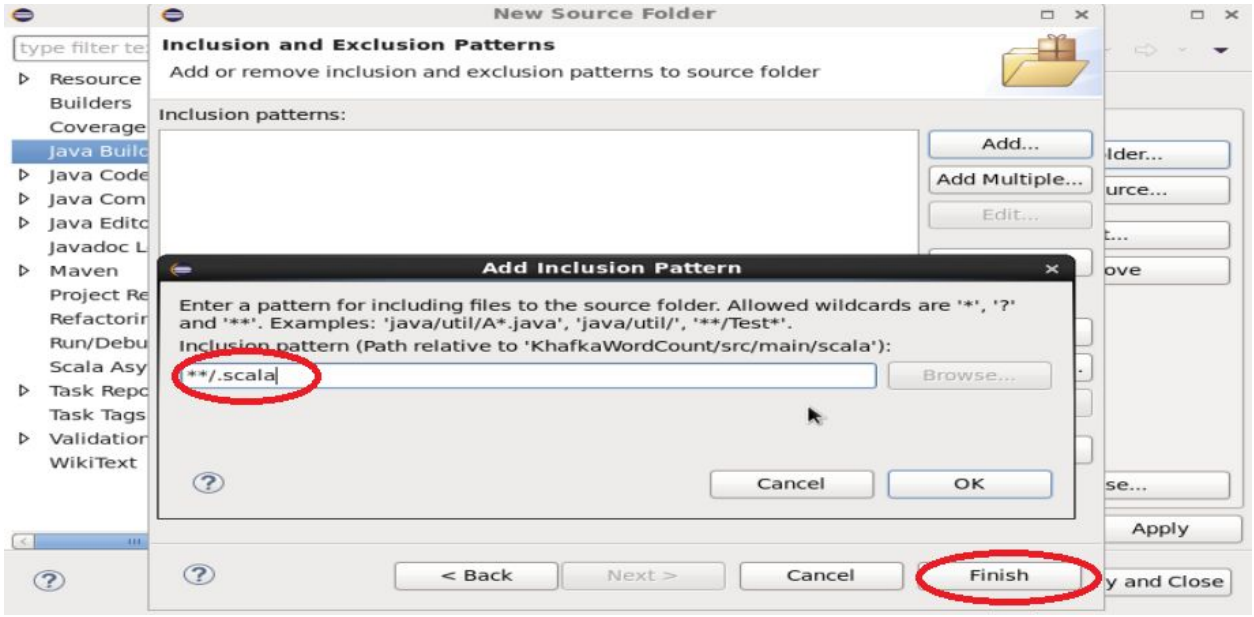
After creating Maven project create a new folder as given in the below screenshot



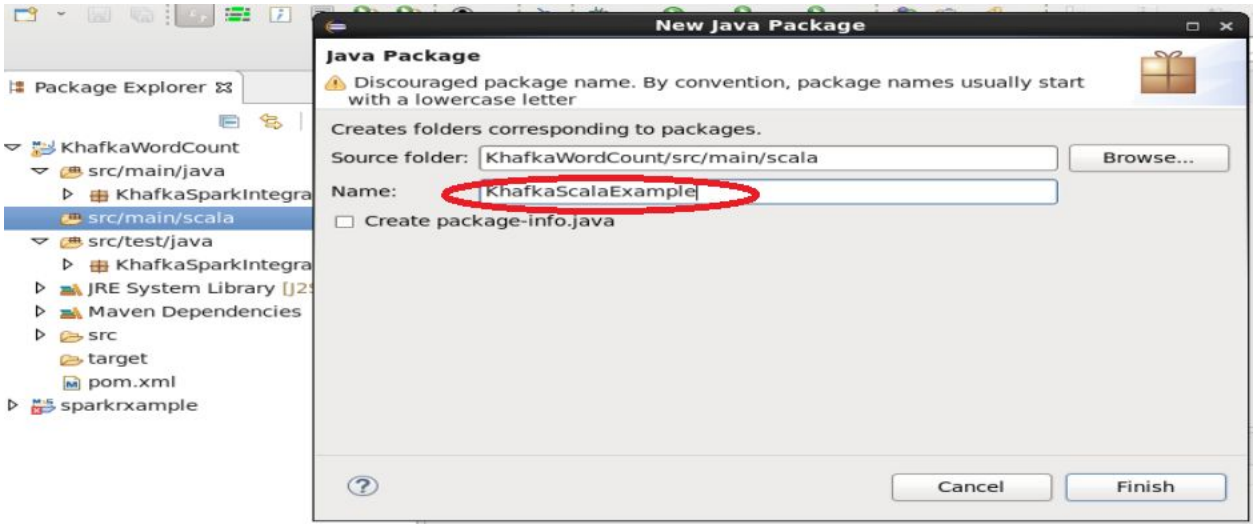
Create a folder named scala as below:



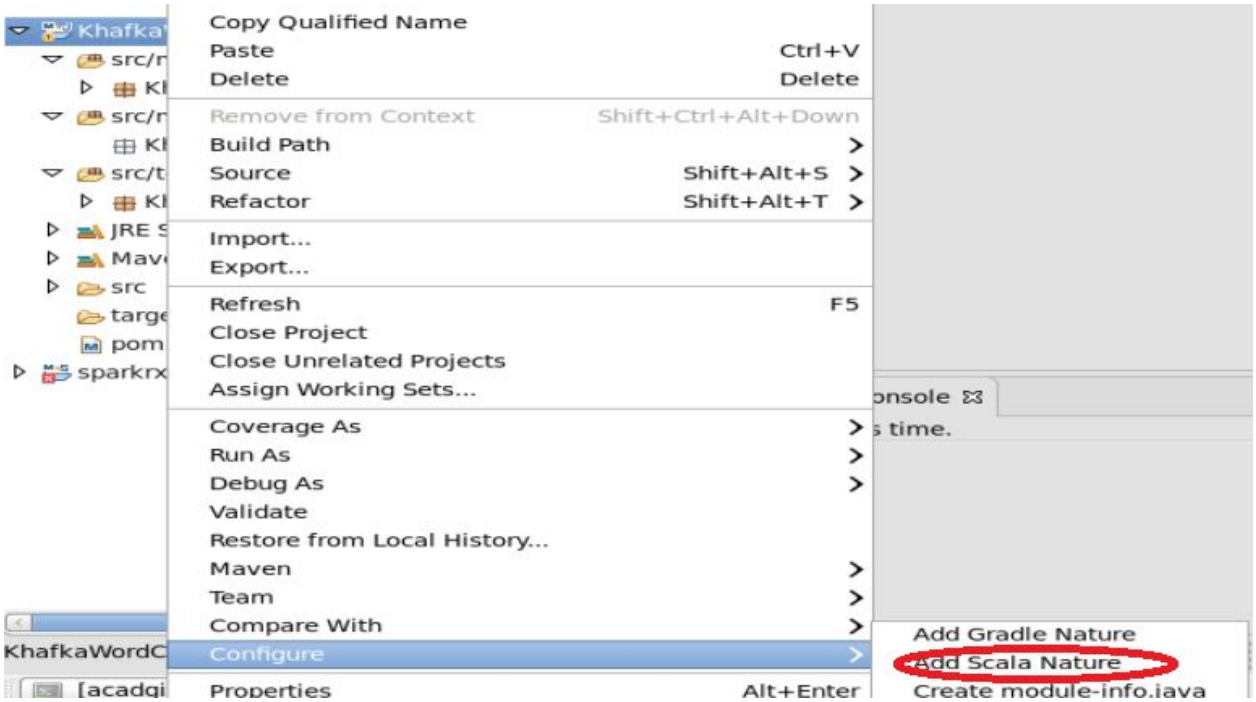
Give the inclusion and pattern as given below:



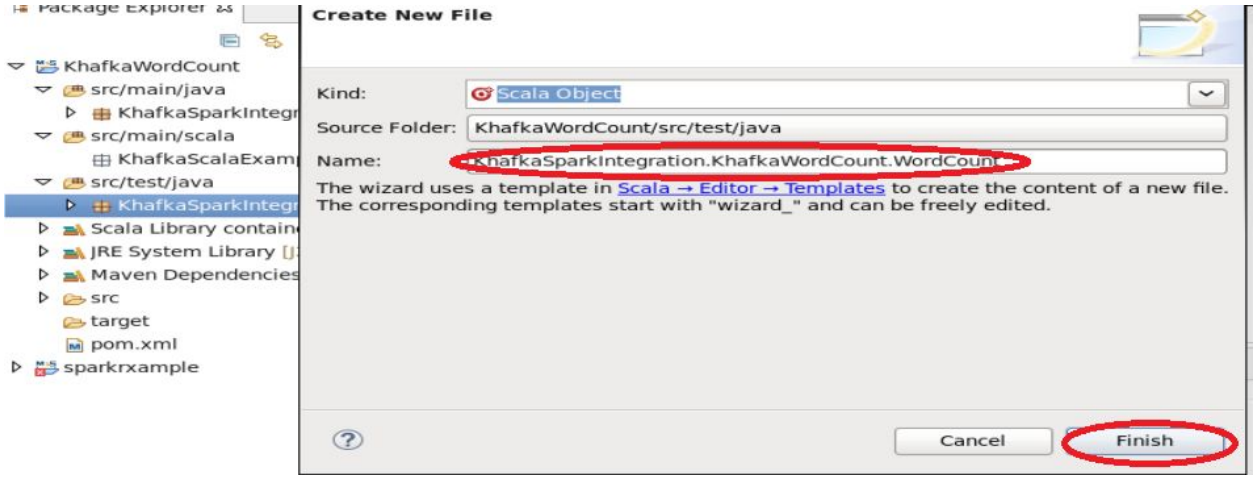
Create a package under the newly created folder as given below:



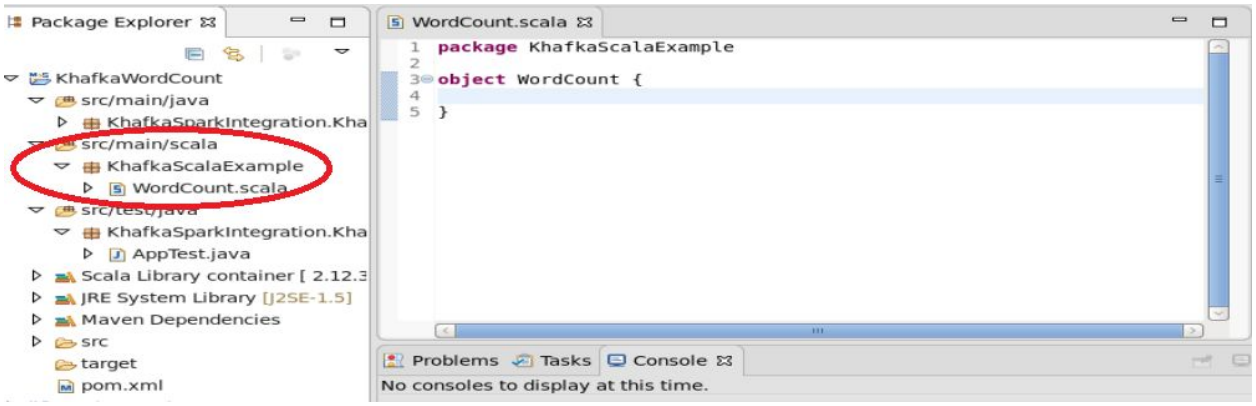
Select the project and configure as Scala nature as given below:



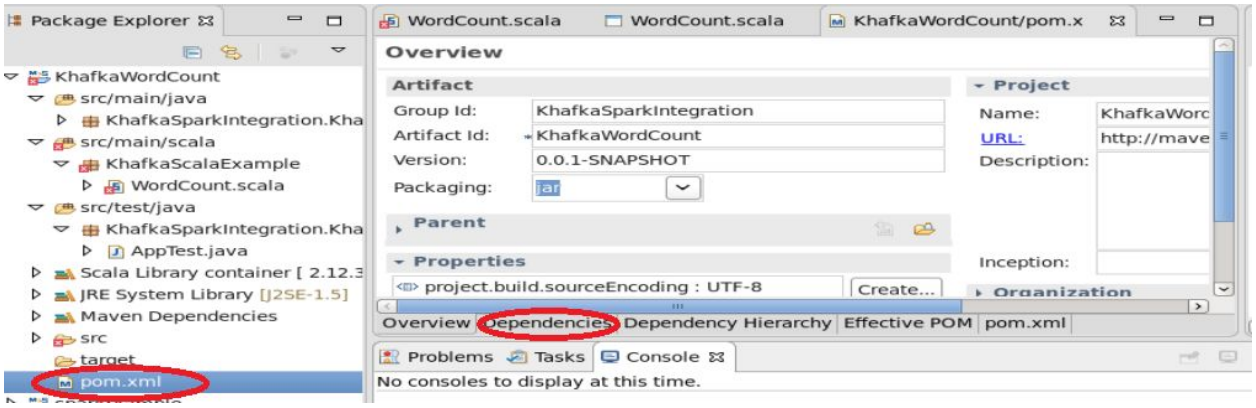
Create a scala object under the newly created package as below:



Find the project/folder/package structure below:

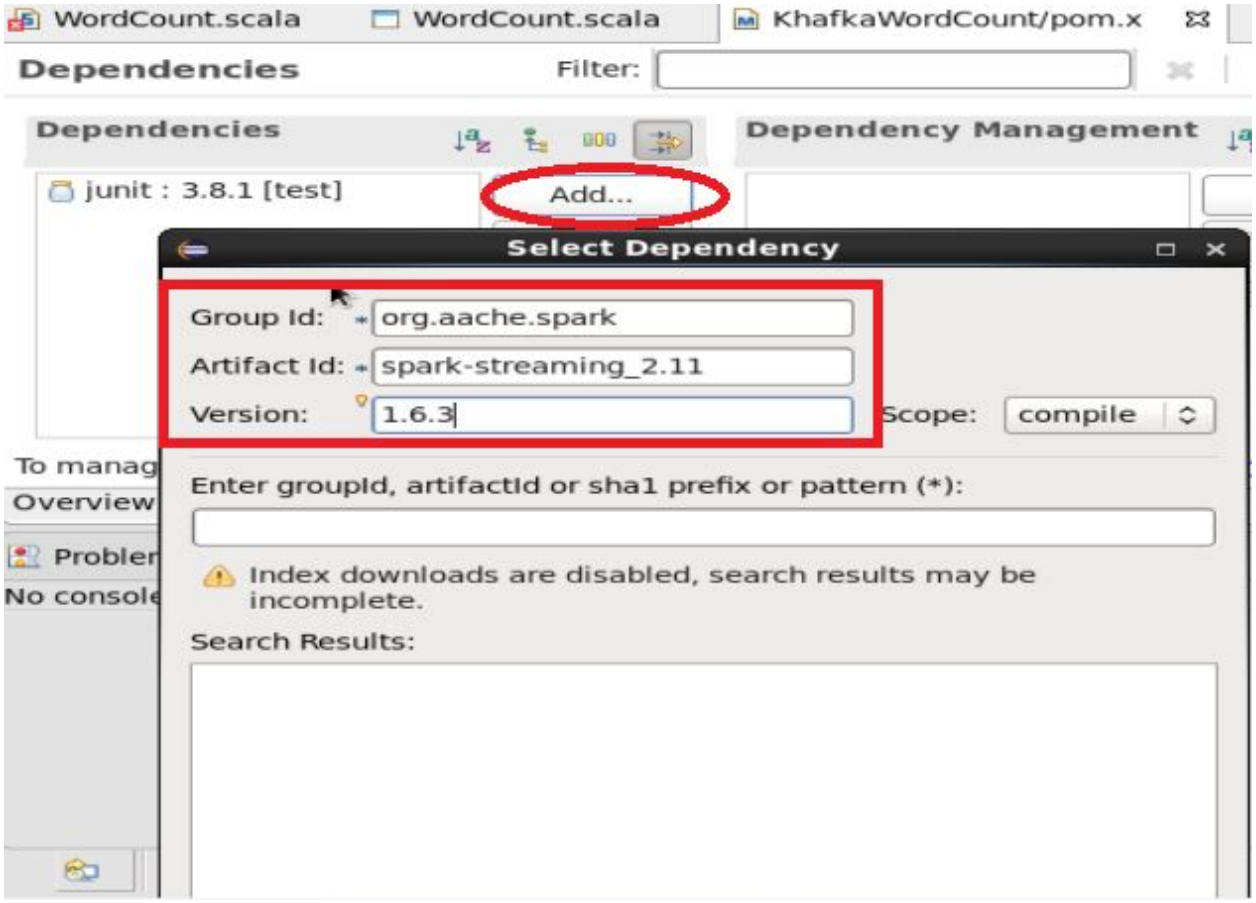


While creating a maven project an xml file named pom is created. Add the dependencies as given below:

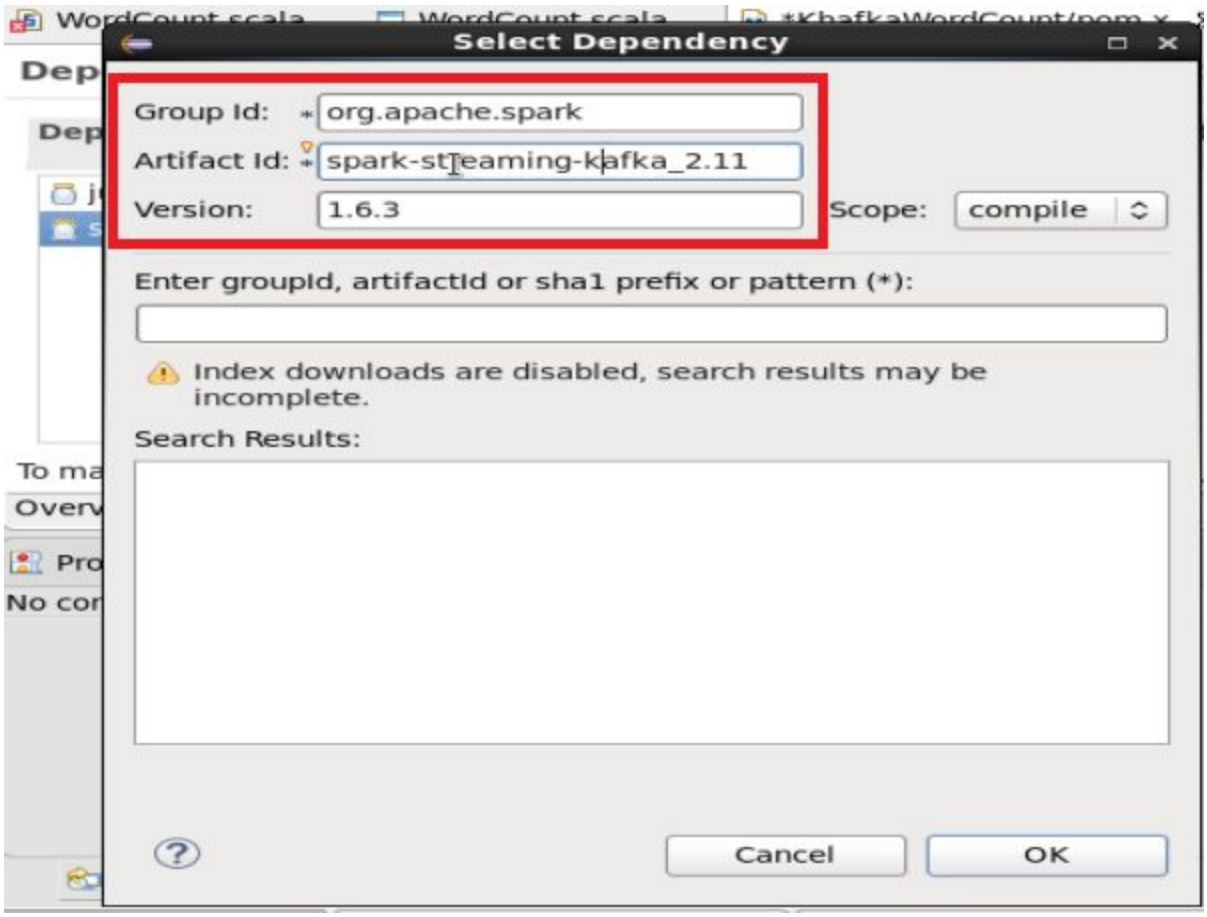


Add the below 2 dependencies highlighted in red:

1st dependency:



2nd dependency:



After creating the scala object write the consumer code for counting the words entered in the producer and run as scala application
Before running the consumer code, start the zookeeper server , khafka server, create the topic and start the producer to enter the message that needs to sent to the consumer using below commands:

Starting zookeeper
cd $KAFKA_HOME
#./bin/zookeeper-server-start ./etc/kafka/zookeeper.properties
./bin/zookeeper-server-start.sh ./config/zookeeper.properties

Starting broker

cd $KAFKA_HOME

#./bin/kafka-server-start./etc/kafka/server.properties

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

Creating topic

cd $KAFKA_HOME

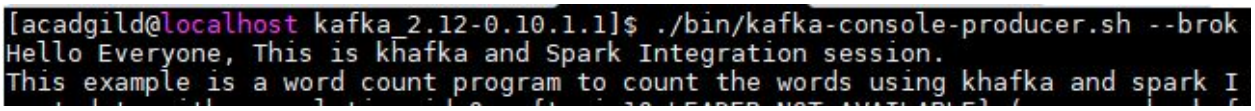
./bin/kafka-topics.sh --create --topic Mytopic10 --zookeeper localhost:2181 --partitions 1 --replication-factor 1

##Starting the producer:

cd $KAFKA_HOME

./bin/kafka-console-producer.sh --broker-list localhost:9092 --topic Mytopic10

Input:



Output :

Wordcount screenshot: