Driver Class

```
package sparkAssignmentMR;

import java.io.*;

import org.apache.hadoop.mapred.TextOutputFormat;
import org.apache.hadoop.mapred.lib.MultipleOutputs;
import org.apache.hadoop.fs.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.*;
import org.apache.hadoop.conf.*;



public class fetchDriver extends Configured implements Tool{


        public static void main(String[] args) throws Exception {
              int returnStatus = ToolRunner.run(new Configuration(), new fetchDriver(), args);
              System.exit(returnStatus);
           }


        public int run(String[] args) throws IOException{

        Job job = new Job(getConf());
         job.setJobName("Fetch Record");
         job.setJarByClass(fetchDriver.class);
         //job.setOutputKeyClass(Text.class);
         //job.setOutputValueClass(Text.class);

         job.setOutputKeyClass(Text.class);
         job.setOutputValueClass(Text.class);
         //job.setOutputFormatClass(TextOutputFormat.class);

         job.setMapperClass(fetchDriverMapper.class);
         //job.setCombinerClass(fetchDriverReducer.class);
         //job.setReducerClass(fetchDriverReducer.class);

        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job,new Path(args[1]));


        try {
                        return job.waitForCompletion(true) ? 0 : 1;
                } catch (ClassNotFoundException e) {
                        // TODO Auto-generated catch block
```

```
				e.printStackTrace();
			} catch (InterruptedException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			}
			return 0;



   }

}
```

Mapper Class

```
package sparkAssignmentMR;


import java.io.BufferedReader;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.HashMap;
import java.util.Iterator;
import org.apache.hadoop.filecache.DistributedCache;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;


public class fetchDriverMapper extends
Mapper<Object, Text, Text, Text> {

	@SuppressWarnings("unused")
	@Override

	public void map(Object key, Text value, Context context) throws IOException,
InterruptedException
	{


		String st [] = value.toString().trim().split(",");
```

```
			//checks for data in the dataset
			if (st.length == 17)

			{
				String VendorID = st[0];
				String tpep_pickup_datetime = st[1];
				String tpep_dropoff_datetime = st[2];
				String passenger_count= st[3];
				String trip_distance= st[4];
				String RatecodeID= st[5];
				String store_and_fwd_flag= st[6];
				String PULocationID= st[7];
				String DOLocationID= st[8];
				String payment_type= st[9];
				String fare_amount= st[10];
				String extra= st[11];
				String mta_tax= st[12];
				String tip_amount= st[13];
				String tolls_amount= st[14];
				String improvement_surcharge= st[15];
				String total_amount= st[16];

				//int VendorIDInt = Integer.parseInt(VendorID);

				if(VendorID.equals("2")
						&& passenger_count.equals("1")
						&& tpep_pickup_datetime.equals("2017-10-01 
00:15:30")
						&& tpep_dropoff_datetime.equals("2017-10-01 
00:25:11")
						&& trip_distance.equals("2.17")
						)
				{

					context.write(new Text(VendorID), new 
Text(tpep_pickup_datetime + " " + tpep_dropoff_datetime + " "
					+ passenger_count + " "+ trip_distance + " "+RatecodeID + " 
"+store_and_fwd_flag + " "+PULocationID + " "+DOLocationID + " "
							+ payment_type + " "+fare_amount +" "+ extra
							+ " "+mta_tax + " "+tip_amount + " 
"+tolls_amount + " "+improvement_surcharge + " "+total_amount));

				}
			}
		}
}
```