LogProcessorMapper.java

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class LogProcessorMapper extends Mapper<LongWritable, Text, Text, IntWritable> {
    private final static IntWritable one = new IntWritable(1);
    private Text ip = new Text();

    public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
        String line = value.toString();
        String[] parts = line.split(" ");
        if (parts.length > 0) {
            ip.set(parts[0]);  // The IP address
            context.write(ip, one);
        }
    }
}
```

LogProcessorReducer.java

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class LogProcessorReducer extends Reducer<Text, IntWritable, Text, IntWritable> {
    public void reduce(Text key, Iterable<IntWritable> values, Context context)
            throws IOException, InterruptedException {
        int sum = 0;
        for (IntWritable value : values) {
            sum += value.get();
        }
        context.write(key, new IntWritable(sum));
    }
}
```

LogProcessorDriver.java

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class LogProcessorDriver {
    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Log File IP Count");

        job.setJarByClass(LogProcessorDriver.class);
        job.setMapperClass(LogProcessorMapper.class);
        job.setReducerClass(LogProcessorReducer.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

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

        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
}
```

Input file (log.txt)

```
192.168.1.1 - - [08/Apr/2024:12:05:23] "GET /index.html"
192.168.1.2 - - [08/Apr/2024:12:06:12] "POST /login"
192.168.1.1 - - [08/Apr/2024:12:07:01] "GET /about.html"
192.168.1.3 - - [08/Apr/2024:12:08:00] "GET /contact"
```

1. Create input directory:

bash

CopyEdit

```
mkdir -p ~/hadoop/input
cp your_log_file.txt ~/hadoop/input/log.txt
```

2. Run Hadoop in local mode:

bash

CopyEdit

```
hadoop jar logprocessor.jar LogProcessorDriver ~/hadoop/input ~/hadoop/output
```

3. Check the output:

bash

CopyEdit

```
cat ~/hadoop/output/part-r-00000
```