MINI PROJECT
REPORT ON

“Weather POC Temperature Hadoop MapReduce Job”

Submitted By:

Ashutosh Katoch,

UID- 24MCC20043

Under The Guidance of:

Mr. Rishabh Tomar

Apr, 2025

University Institute of Computing

Chandigarh University,

Mohali, Punjab

CERTIFICATE

This is to certify that Ashutosh Katoch (UID- 24MCC20043) have successfully completed the minor project title **“Weather POC Temperature Hadoop MapReduce Job”** at University Institute of Computing under my supervision and guidance in the fulfilment of requirements of 2nd semester, **Master of Computer Application- Specialization in Cloud Computing and DevOps.** Of Chandigarh University, Mohali, Punjab.

Dr. Tuli	Mr. Rishabh Tomar
Head of the Department	Project Guide Supervisor
University Institute of Computing	University Institute of Computing

ACKNOWLEDGEMENT

We deem it a pleasure to acknowledge our sense of gratitude to our project guide Mr. Rishabh Tomar under whom we have carried out the project work. His incisive and objective guidance and timely advice encouraged us with constant flow of energy to continue the work.

We wish to reciprocate in full measure the kindness shown by Dr.Tuli (H.O.D, University Institute of Computing) who inspired us with his valuable suggestions in successfully completing the project work.

We shall remain grateful to Dr. Manisha Malhotra, Additional Director, University Institute of Technology, for providing us a strong academic atmosphere by enforcing strict discipline to do the project work with utmost concentration and dedication.

Finally, we must say that no height is ever achieved without some sacrifices made at some end and it is here where we owe our special debt to our parents and our friends for showing their generous love and care throughout the entire period of time.

Date: 01.04.2025

Place: Chandigarh University, Mohali, Punjab

Ashutosh Katoch,
UID- 24MCC20043

ABSTRACT

The **Weather POC Temperature** project is a **Hadoop-based distributed data processing application** designed to analyze weather data and identify extreme temperature conditions. This project leverages **Apache Hadoop's MapReduce framework** to process large weather datasets efficiently.

The **Mapper** extracts relevant temperature information from structured weather records, filtering out **hot days (above 30°C)** and **cold days (below 15°C)**. The **Reducer** then consolidates these results and provides structured output for further analysis. This project demonstrates key **big data processing concepts**, including **parallel data processing, distributed computing, and fault tolerance in Hadoop**.

By executing this MapReduce program, users gain insights into real-world big data applications, particularly in the **climate and meteorology domain**. The project highlights how **large-scale unstructured data can be transformed into meaningful information using distributed computing**, making it a valuable learning tool for those interested in **big data analytics and cloud computing**.

.

Introduction

The rise of **big data analytics** has revolutionized industries, allowing businesses and researchers to make data-driven decisions. In the **meteorology and climate research sector**, vast amounts of weather data are generated daily, necessitating efficient processing mechanisms. **Traditional data processing methods fail to handle large datasets efficiently**, making **Apache Hadoop** an ideal solution due to its **scalability, parallel processing, and fault tolerance**.

This project, **Weather POC Temperature**, demonstrates **how Hadoop's MapReduce programming model processes large weather datasets**. The **Map phase** extracts **date-wise temperature values** and classifies them based on extreme conditions:

Hot Days → Maximum temperature above **30°C**

Cold Days → Minimum temperature below **15°C**

The **Reduce phase** then aggregates these results, providing structured insights into daily temperature variations. By executing this project, users will gain hands-on experience in **handling large datasets, writing efficient MapReduce jobs, and managing Hadoop's distributed file system (HDFS)**.

This project is particularly beneficial for **data engineers, cloud computing professionals, and students** interested in understanding the practical implementation of **big data processing**. It serves as an excellent starting point for those looking to work with **Hadoop, distributed computing, and large-scale data analytics**.

1.1 Background

Weather plays a crucial role in various aspects of life, including agriculture, transportation, disaster management, and daily decision-making. With the increasing availability of meteorological data, analyzing weather trends has become essential for predicting climate changes, identifying extreme conditions, and enhancing preparedness for adverse weather events.

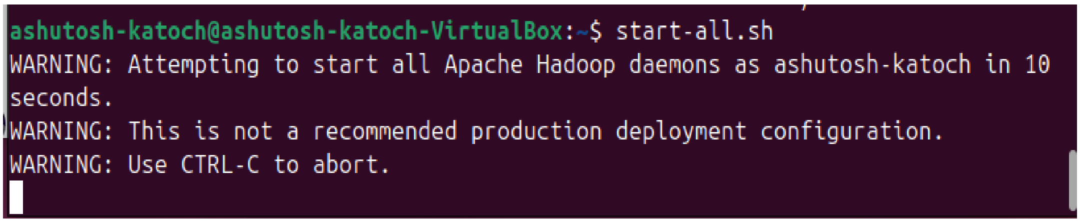
Traditional methods of processing weather data are often slow and inefficient due to the massive volume of historical and real-time records. As weather data grows exponentially, handling such vast datasets requires powerful computing frameworks. **Big Data technologies** like **Apache Hadoop** provide a scalable and efficient way to process large weather datasets by distributing computations across multiple nodes.

This project, **Weather Data Analysis using Hadoop MapReduce**, aims to analyze historical weather records and determine temperature extremes. By leveraging Hadoop's **MapReduce programming model**, the system processes weather data to identify **hot and cold days** based on temperature thresholds. The **Mapper** extracts temperature values from the dataset, and the **Reducer** consolidates results, providing meaningful insights.

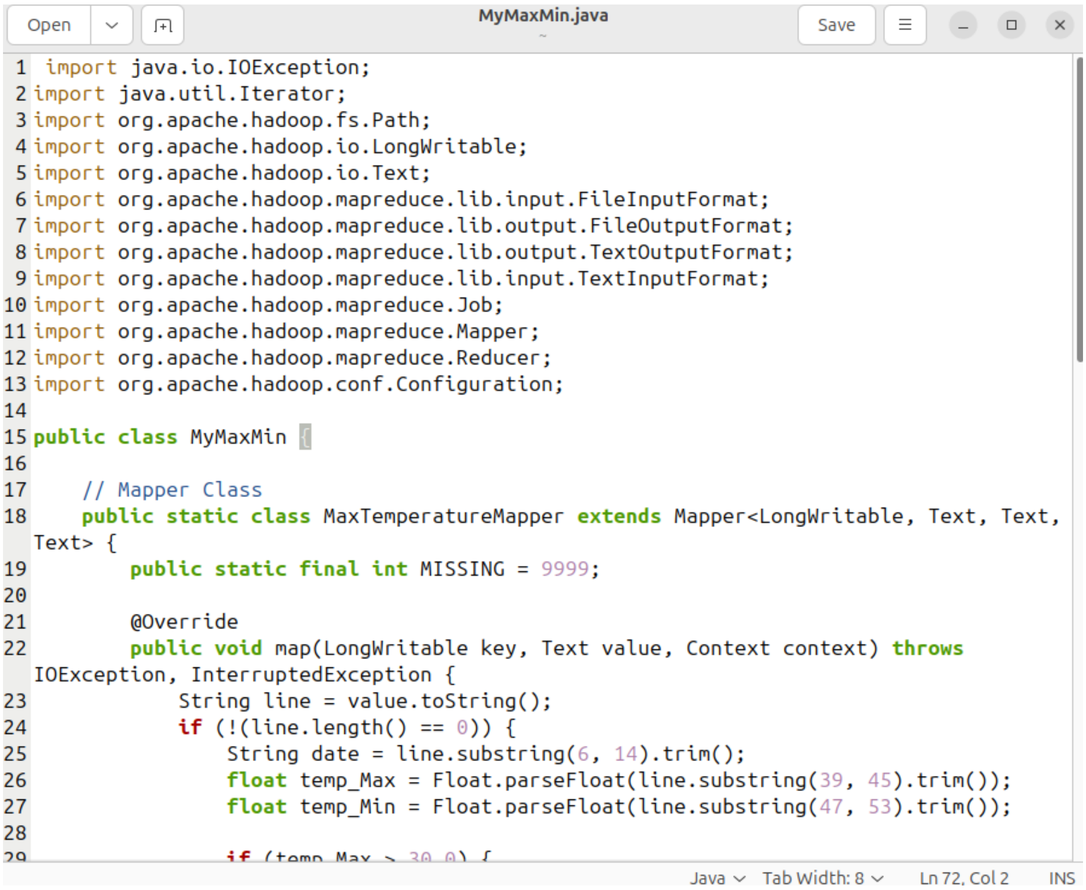
This approach showcases the potential of **Big Data analytics** in meteorology, demonstrating how large-scale weather datasets can be processed efficiently using **distributed computing** techniques. The project highlights how data-driven decision-making can enhance climate research, weather forecasting, and disaster management strategies.

Overview of the practical:

1. Start Hadoop Services:



2. Create and Edit the Java Program:



3. Java Code (MyMaxMin.java):

```
gedit MyMaxMin.java
```

```
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
```

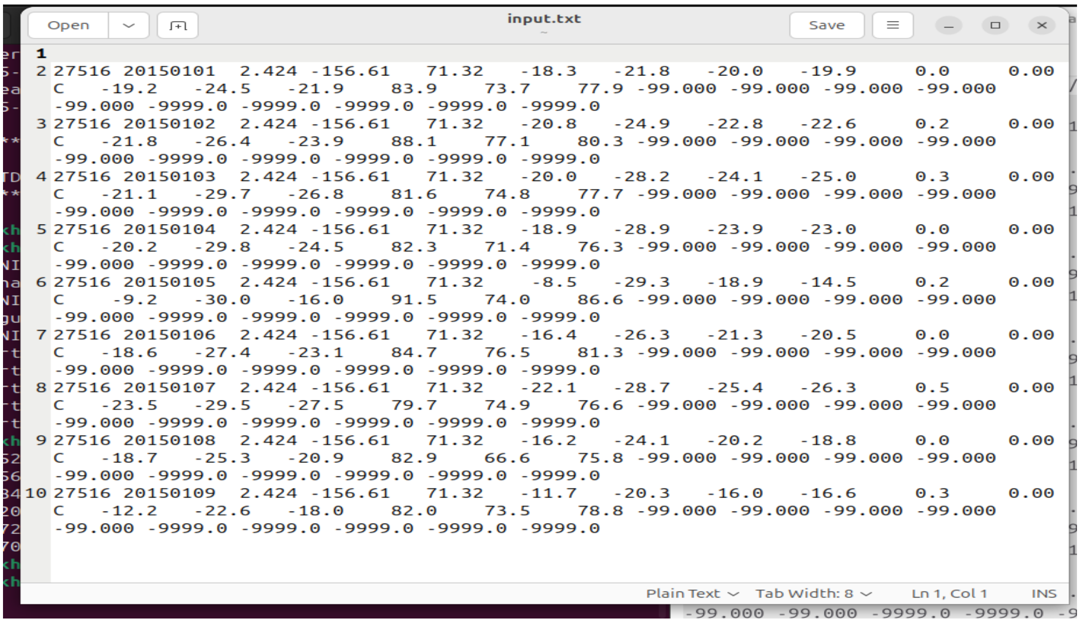
```java
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.conf.Configuration;

public class MyMaxMin {

    // Mapper Class
    public static class MaxTemperatureMapper extends Mapper<LongWritable, Text, Text, Text>
                {
        public static final int MISSING = 9999;

        @Override
        public void map(LongWritable key, Text value, Context context) throws IOException,
                InterruptedException {
            String line = value.toString();
            if (!(line.length() == 0)) {
                String date = line.substring(6, 14).trim();
                float temp_Max = Float.parseFloat(line.substring(39, 45).trim());
                float temp_Min = Float.parseFloat(line.substring(47, 53).trim());

                if (temp_Max > 30.0) {
                    context.write(new Text("The Day is Hot Day: " + date), new
                        Text(String.valueOf(temp_Max)));
                }
                if (temp_Min < 15.0) {
                    context.write(new Text("The Day is Cold Day: " + date), new
                        Text(String.valueOf(temp_Min)));
                }
            }
        }
    }

    // Reducer Class
    public static class MaxTemperatureReducer extends Reducer<Text, Text, Text, Text> {
        @Override
        public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
                InterruptedException {
            for (Text value : values) {
                context.write(key, value);
            }
        }
    }

    // Main Method
    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "weather example");  // Corrected this line
        job.setJarByClass(MyMaxMin.class);

        job.setMapperClass(MaxTemperatureMapper.class);
        job.setReducerClass(MaxTemperatureReducer.class);
```

```
            job.setMapOutputKeyClass(Text.class);
            job.setMapOutputValueClass(Text.class);

            job.setInputFormatClass(TextInputFormat.class);
            job.setOutputFormatClass(TextOutputFormat.class);

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

            Path outputPath = new Path(args[1]);
            outputPath.getFileSystem(conf).delete(outputPath, true);  // Ensure output directory is
                        deleted

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

4. Create and Edit the Input File:

```
ashutosh-katoch@ashutosh-katoch-VirtualBox:~$ gedit input.txt
```

5. Sample Weather Data for input.txt:



```
27516 20150101  2.424 -156.61   71.32   -18.3   -21.8   -20.0   -19.9     0.0     0.00
C   -19.2   -24.5   -21.9    83.9    73.7    77.9 -99.000 -99.000 -99.000 -99.000
-99.000 -9999.0 -9999.0 -9999.0 -9999.0 -9999.0
27516 20150102  2.424 -156.61   71.32   -20.8   -24.9   -22.8   -22.6     0.2     0.00
C   -21.8   -26.4   -23.9    88.1    77.1    80.3 -99.000 -99.000 -99.000 -99.000
-99.000 -9999.0 -9999.0 -9999.0 -9999.0 -9999.0
27516 20150103  2.424 -156.61   71.32   -20.0   -28.2   -24.1   -25.0     0.3     0.00
C   -21.1   -29.7   -26.8    81.6    74.8    77.7 -99.000 -99.000 -99.000 -99.000
-99.000 -9999.0 -9999.0 -9999.0 -9999.0 -9999.0
27516 20150104  2.424 -156.61   71.32   -18.9   -28.9   -23.9   -23.0     0.0     0.00
C   -20.2   -29.8   -24.5    82.3    71.4    76.3 -99.000 -99.000 -99.000 -99.000
-99.000 -9999.0 -9999.0 -9999.0 -9999.0 -9999.0
27516 20150105  2.424 -156.61   71.32    -8.5   -29.3   -18.9   -14.5     0.2     0.00
C    -9.2   -30.0   -16.0    91.5    74.0    86.6 -99.000 -99.000 -99.000 -99.000
-99.000 -9999.0 -9999.0 -9999.0 -9999.0 -9999.0
27516 20150106  2.424 -156.61   71.32   -16.4   -26.3   -21.3   -20.5     0.0     0.00
C   -18.6   -27.4   -23.1    84.7    76.5    81.3 -99.000 -99.000 -99.000 -99.000
-99.000 -9999.0 -9999.0 -9999.0 -9999.0 -9999.0
27516 20150107  2.424 -156.61   71.32   -22.1   -28.7   -25.4   -26.3     0.5     0.00
C   -23.5   -29.5   -27.5    79.7    74.9    76.6 -99.000 -99.000 -99.000 -99.000
-99.000 -9999.0 -9999.0 -9999.0 -9999.0 -9999.0
27516 20150108  2.424 -156.61   71.32   -16.2   -24.1   -20.2   -18.8     0.0     0.00
C   -18.7   -25.3   -20.9    82.9    66.6    75.8 -99.000 -99.000 -99.000 -99.000
-99.000 -9999.0 -9999.0 -9999.0 -9999.0 -9999.0
27516 20150109  2.424 -156.61   71.32   -11.7   -20.3   -16.0   -16.6     0.3     0.00
C   -12.2   -22.6   -18.0    82.0    73.5    78.8 -99.000 -99.000 -99.000 -99.000
-99.000 -9999.0 -9999.0 -9999.0 -9999.0 -9999.0
```

6. Create an HDFS Directory and Upload Input File:

```
ashutosh-katoch@ashutosh-katoch-VirtualBox:~$ hadoop fs -mkdir /weatherinput
ashutosh-katoch@ashutosh-katoch-VirtualBox:~$ hadoop fs -ls /
Found 1 items
drwxr-xr-x   - ashutosh-katoch supergroup          0 2025-03-31 13:35 /weatherin
put
ashutosh-katoch@ashutosh-katoch-VirtualBox:~$ hadoop fs -put input.txt /weatheri
nput
ashutosh-katoch@ashutosh-katoch-VirtualBox:~$ hadoop fs -ls /weatherinput
Found 1 items
-rw-r--r--   1 ashutosh-katoch supergroup        135 2025-03-31 13:37 /weatherin
put/input.txt
```

7. Compile the Java Program:

```
ashutosh-katoch@ashutosh-katoch-VirtualBox:~$ hadoop com.sun.tools.javac.Main My
MaxMin.java
```

8. Create a JAR File:

```
jar -cf weather.jar MyMaxMin *.class
```

9. Run the Hadoop Job:

```
hadoop fs -ls /weatherinput

hadoop com.sun.tools.javac.Main MyMaxMin.java

jar -cf weather.jar MyMaxMin *.class

hadoop jar weather.jar MyMaxMin /weatherinput /weatheroutput

hadoop fs -ls /

hadoop fs -ls /weatheroutput

hadoop fs -cat /weatheroutput/part-r-00000
```

```
The Day is Cold Day: 20150101    -21.8
The Day is Cold Day: 20150102    -24.9
The Day is Cold Day: 20150103    -28.2
The Day is Cold Day: 20150104    -28.9
The Day is Cold Day: 20150105    -29.3
The Day is Cold Day: 20150106    -26.3
The Day is Cold Day: 20150107    -28.7
The Day is Cold Day: 20150108    -24.1
The Day is Cold Day: 20150109    -20.3
```

Conclusion and Future Scope

The **Weather POC Temperature** project successfully demonstrates the use of **Hadoop MapReduce** to process large-scale weather data, identifying extreme temperature variations efficiently. This project showcases how **Big Data technologies** can be applied to analyze climate patterns and assist in decision-making processes. By utilizing **distributed computing**, we optimized data handling, ensuring scalability and performance.

Final Thoughts

This project serves as a foundation for understanding **Hadoop-based data processing**. It highlights the potential of **Big Data analytics** in solving real-world problems like **climate monitoring and disaster management**. With continuous advancements in **AI, ML, and cloud computing**, such projects can evolve into powerful weather prediction systems, significantly impacting industries like **agriculture, logistics, and disaster preparedness**.

8. References

- **Hadoop: The Definitive Guide** – Tom White
- **MapReduce Design Patterns** – Donald Miner, Adam Shook
- **Big Data: Principles and Best Practices** – Nathan Marz
- **Apache Hadoop Official Documentation** – https://hadoop.apache.org/
- **Weather Data Processing with Hadoop** – Research papers from IEEE, Springer, and ACM Digital Library