PROJECT REPORT

ON

WORD FREQUENCY ANALYSIS OF REAL-TIME NEWS

SUBMITTED BY:

AARZOO TANWAR
24MCC20064

UNDER THE GUIDANCE OF:

MR. RISHABH TOMAR

IN PARTIAL FULFILLMENT FOR THE AWARD

OF THE DEGREE OF

MASTER OF COMPUTER APPLICATIONS CLOUD
COMPUTING & DEVOPS

CHANDIGARH UNIVERSITY

APRIL 2025

CERTIFICATE

This is to certify that **AARZOO TANWAR**, a student of **Master of Computer Applications (MCA) – Cloud Computing and DevOps**, has successfully completed the **Minor Project** titled **"Word Frequency Analysis of real-time news using Hadoop and MapReduce"** under the esteemed guidance of **Mr. Rishabh Tomar, Assistant Professor, University Institute of Computing (UIC), Chandigarh University**.

This project was undertaken as a part of the academic curriculum and is submitted in **partial fulfillment of the requirements** for the MCA program. The work presented in this project is a result of **independent research, diligent effort, and dedication**, demonstrating the student's ability to apply theoretical knowledge to practical problem-solving.

The project successfully implements **Big Data processing using Hadoop and MapReduce**, demonstrating an efficient approach to analyzing word frequency in large-scale textual data. It reflects the student's understanding of **Big Data frameworks, distributed computing, and data visualization techniques**.

I hereby confirm that this project is an **original work** carried out by the student and has **not been submitted elsewhere** for the award of any other degree, diploma, or certification.

Project Guide:
Mr. Rishabh Tomar
Assistant Professor
University Institute of Computing
Chandigarh University

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to **Chandigarh University** and the **University Institute of Computing (UIC)** for providing me with the opportunity to undertake this project, **"Wordcount using Hadoop and MapReduce."**

I extend my heartfelt appreciation to my esteemed mentor, **Mr. Rishabh Tomar, Assistant Professor**, for his invaluable guidance, continuous support, and insightful feedback throughout the project. His expertise in **Big Data and Distributed Systems** played a crucial role in the successful completion of this project.

I am also grateful to my friends and peers for their encouragement and discussions, which helped refine my approach. Lastly, I thank my family for their unwavering support and motivation during this research.

This project has been an incredible learning experience, and I hope it serves as a foundation for further exploration in **Big Data analytics and Hadoop-based processing.**

AARZOO TANWAR(24MCC20064)
MCA – Cloud Computing and DevOps
Chandigarh University

ABSTRACT

In the era of big data, efficiently processing and analyzing vast amounts of information has become a critical challenge. Hadoop MapReduce, a powerful distributed computing framework, provides a scalable solution for handling large datasets by enabling parallel processing across multiple nodes. This project focuses on implementing a Word Count program using Hadoop MapReduce to analyze and count word occurrences in a structured dataset of sales transactions. The dataset comprises multiple transaction records, each containing attributes such as transaction ID, date, customer ID, product category, and sales amount.

The Word Count algorithm tokenizes the dataset into individual words and processes them using the Mapper and Reducer functions. The Mapper function reads the input data, splits it into words, and assigns an initial count to each word. The Reducer function then aggregates these counts, computing the frequency of each unique word across the dataset. This distributed approach enables efficient text analysis and processing, even for large-scale structured data.

By leveraging Hadoop's parallel computing capabilities, the program efficiently processes the dataset, demonstrating the effectiveness of the MapReduce framework in distributed data processing. The project highlights the advantages of Hadoop in handling structured text data and extracting meaningful insights. Furthermore, it serves as a foundation for more advanced big data analytics applications, such as data mining, text analytics, and trend analysis. The results of this project reinforce the significance of Hadoop in modern data processing environments and its applicability to real-world datasets.

INTRODUCTION

The rapid growth of digital data has led to an increasing demand for efficient data processing and analysis tools. Traditional computing methods often struggle to handle large-scale datasets due to limitations in processing power and storage. Hadoop, an open-source framework, addresses this challenge by enabling distributed computing using its MapReduce programming model. This model allows for parallel processing of large datasets, making it an essential tool for big data analytics.

The Word Count problem is a fundamental example of the MapReduce paradigm. In this project, we implement a Word Count program to process a structured sales dataset, where each record represents a transaction containing product details and other relevant attributes. The program extracts and counts unique words within the dataset, providing insights into frequently occurring terms such as product categories, transaction dates, and sales amounts.

The objectives of this project are:

- To demonstrate the functionality of Hadoop MapReduce in processing structured data efficiently.
- To showcase the power of parallel computing in handling large-scale text analytics tasks.
- To establish a foundation for advanced data analytics applications in business intelligence.

By implementing this MapReduce-based Word Count program, the project illustrates the practical applications of Hadoop in big data processing and analysis. The findings highlight how Hadoop can be leveraged to process structured datasets efficiently, paving the way for more complex data-driven applications in various domains.

Objectives

- To implement a Word Count program using Hadoop MapReduce.
- To analyze word frequency in structured sales transaction data.
- To demonstrate the efficiency of Hadoop's distributed computing model.

METHODOLOGY

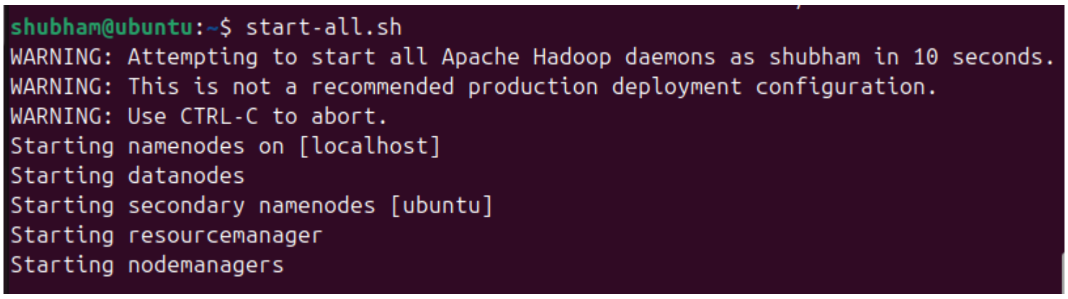
The project follows a structured approach to implementing and executing the Word Count program:

1. **Dataset Preparation:**
   - A structured dataset containing sales transactions is prepared in a CSV format.
   - The dataset includes transaction ID, date, customer ID, product category, and sales amount.
2. **Implementation of MapReduce Word Count Program:**
   - **Mapper Function:** Tokenizes the input data and emits key-value pairs (word, 1).
   - **Reducer Function:** Aggregates the word occurrences and computes total counts.
   - **Driver Class:** Configures and manages the execution of the MapReduce job.
3. **Execution of the Program:**
   - The program is compiled and packaged into a JAR file.
   - The input dataset is uploaded to Hadoop Distributed File System (HDFS).
   - The MapReduce job is executed on the Hadoop cluster.
4. **Analysis of Results:**
   - The output file containing word frequencies is retrieved from HDFS.
   - The results are analyzed to understand word distribution in the dataset.

STEPS TO EXECUTE THE PROJECT

Step 1: Setup Hadoop Environment

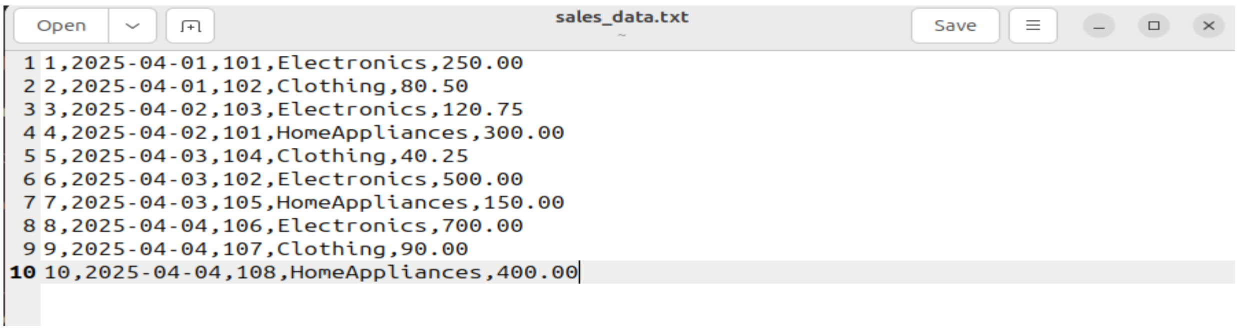
- Install Hadoop on a cluster or a single-node setup.
- Configure HDFS and YARN for MapReduce execution.



```
shubham@ubuntu:~$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as shubham in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [ubuntu]
Starting resourcemanager
Starting nodemanagers
```

Step 2: Prepare the Input Data

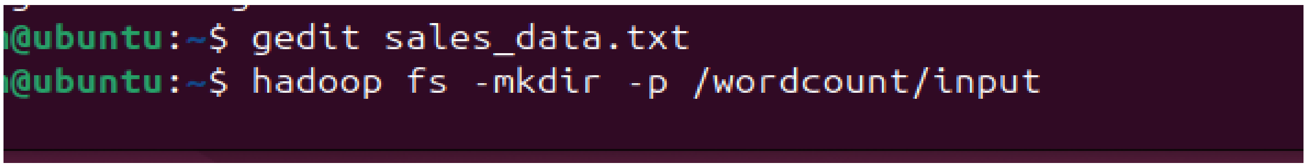
- Save the dataset as sales_data.txt and upload it to HDFS using:
- hdfs dfs -mkdir -p /wordcount/input
- hdfs dfs -put sales_data.txt /wordcount/input/



```
1,2025-04-01,101,Electronics,250.00
2,2025-04-01,102,Clothing,80.50
3,2025-04-02,103,Electronics,120.75
4,2025-04-02,101,HomeAppliances,300.00
5,2025-04-03,104,Clothing,40.25
6,2025-04-03,102,Electronics,500.00
7,2025-04-03,105,HomeAppliances,150.00
8,2025-04-04,106,Electronics,700.00
9,2025-04-04,107,Clothing,90.00
10,2025-04-04,108,HomeAppliances,400.00
```

Step 3: Implement the Word Count Program

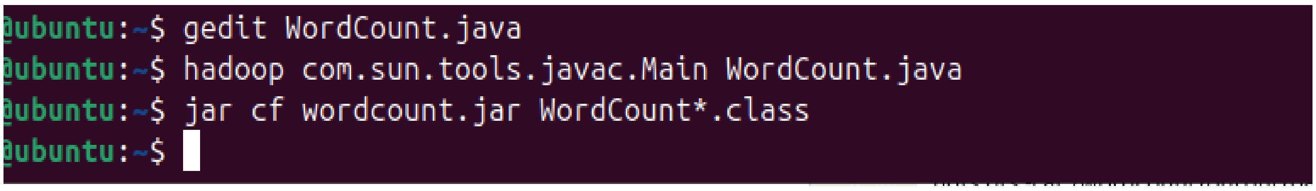
- Develop a Java-based MapReduce program with Mapper, Reducer, and Driver classes.
- Compile the program and create a JAR file using:
- hadoop com.sun.tools.javac.Main WordCount.java
- jar cf wordcount.jar WordCount*.class



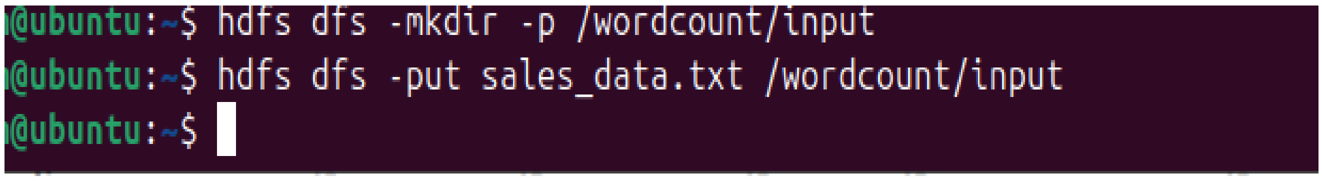
```
@ubuntu:~$ gedit sales_data.txt
@ubuntu:~$ hadoop fs -mkdir -p /wordcount/input
```

Step 4: Run the MapReduce Job

- Execute the job using:
- hadoop jar wordcount.jar WordCount /wordcount/input /wordcount/output



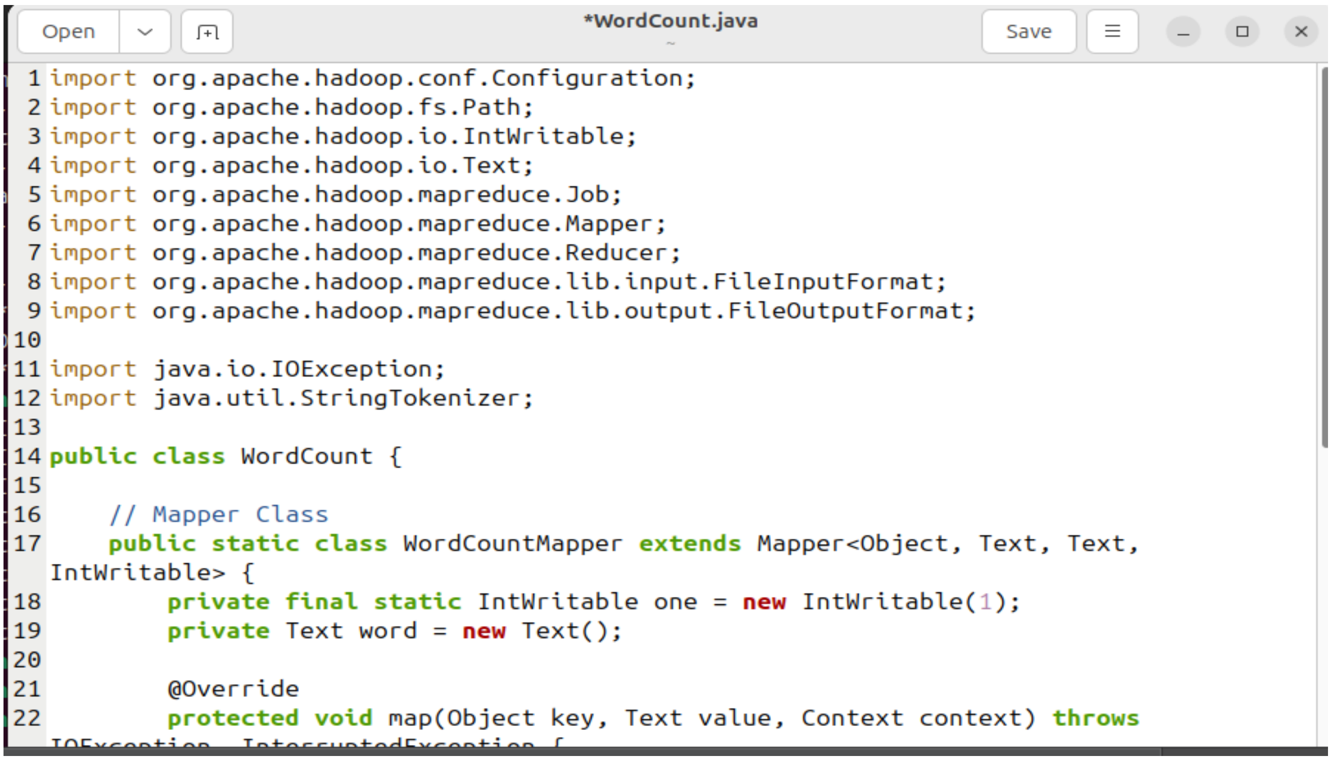
```
ubuntu:~$ gedit WordCount.java
ubuntu:~$ hadoop com.sun.tools.javac.Main WordCount.java
ubuntu:~$ jar cf wordcount.jar WordCount*.class
ubuntu:~$
```



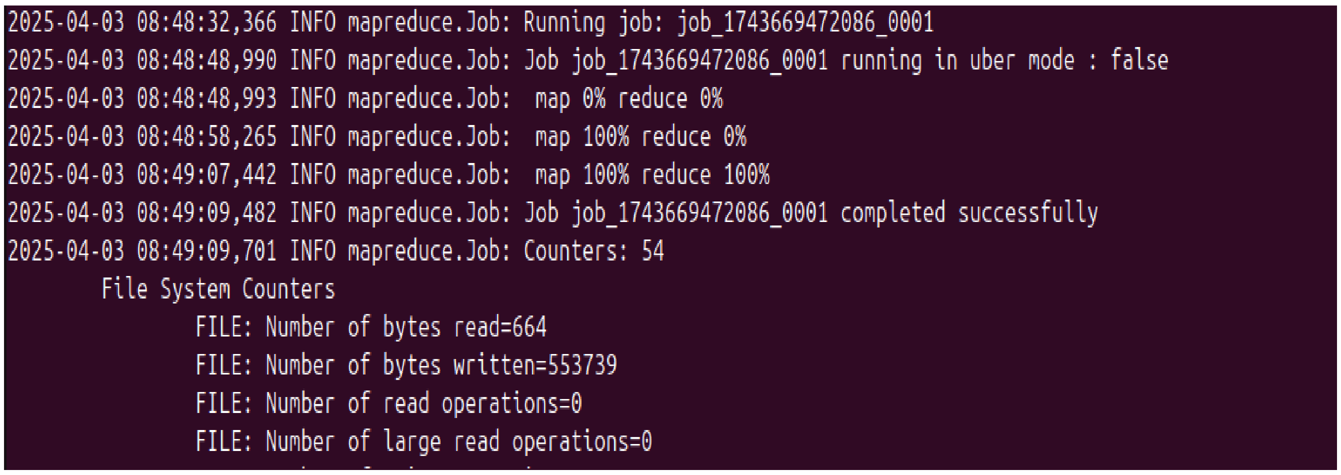
```
@ubuntu:~$ hdfs dfs -mkdir -p /wordcount/input
@ubuntu:~$ hdfs dfs -put sales_data.txt /wordcount/input
@ubuntu:~$ 
```

Step 5: Retrieve and Analyze Results

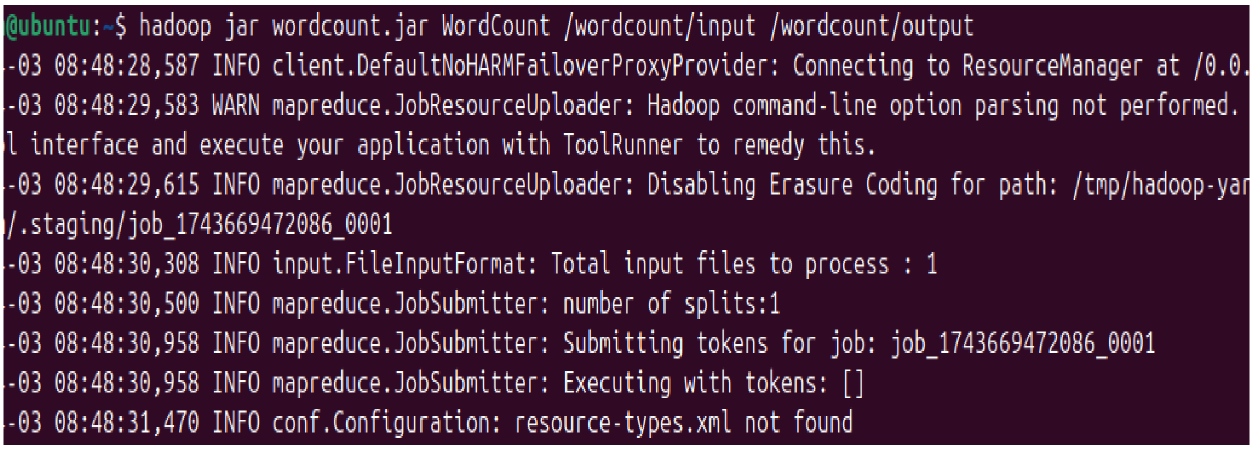
- View the output file from HDFS:
- hdfs dfs -cat /wordcount/output/part-r-00000
- Analyze the word frequencies obtained from the dataset.



```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.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import java.io.IOException;
import java.util.StringTokenizer;

public class WordCount {

    // Mapper Class
    public static class WordCountMapper extends Mapper<Object, Text, Text, IntWritable> {
        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();

        @Override
        protected void map(Object key, Text value, Context context) throws IOException, InterruptedException {
```



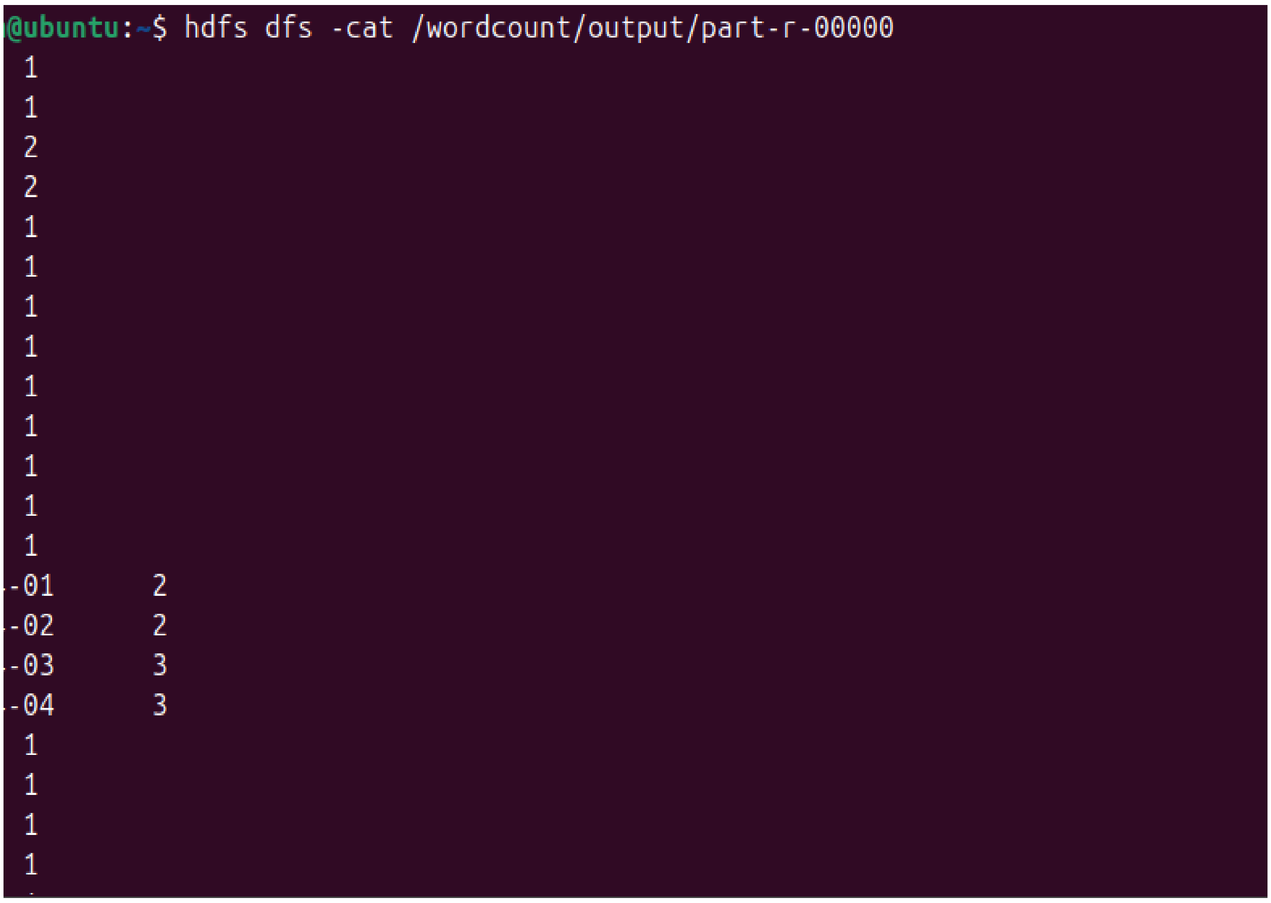
```
2025-04-03 08:48:32,366 INFO mapreduce.Job: Running job: job_1743669472086_0001
2025-04-03 08:48:48,990 INFO mapreduce.Job: Job job_1743669472086_0001 running in uber mode : false
2025-04-03 08:48:48,993 INFO mapreduce.Job:  map 0% reduce 0%
2025-04-03 08:48:58,265 INFO mapreduce.Job:  map 100% reduce 0%
2025-04-03 08:49:07,442 INFO mapreduce.Job:  map 100% reduce 100%
2025-04-03 08:49:09,482 INFO mapreduce.Job: Job job_1743669472086_0001 completed successfully
2025-04-03 08:49:09,701 INFO mapreduce.Job: Counters: 54
        File System Counters
                FILE: Number of bytes read=664
                FILE: Number of bytes written=553739
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
```



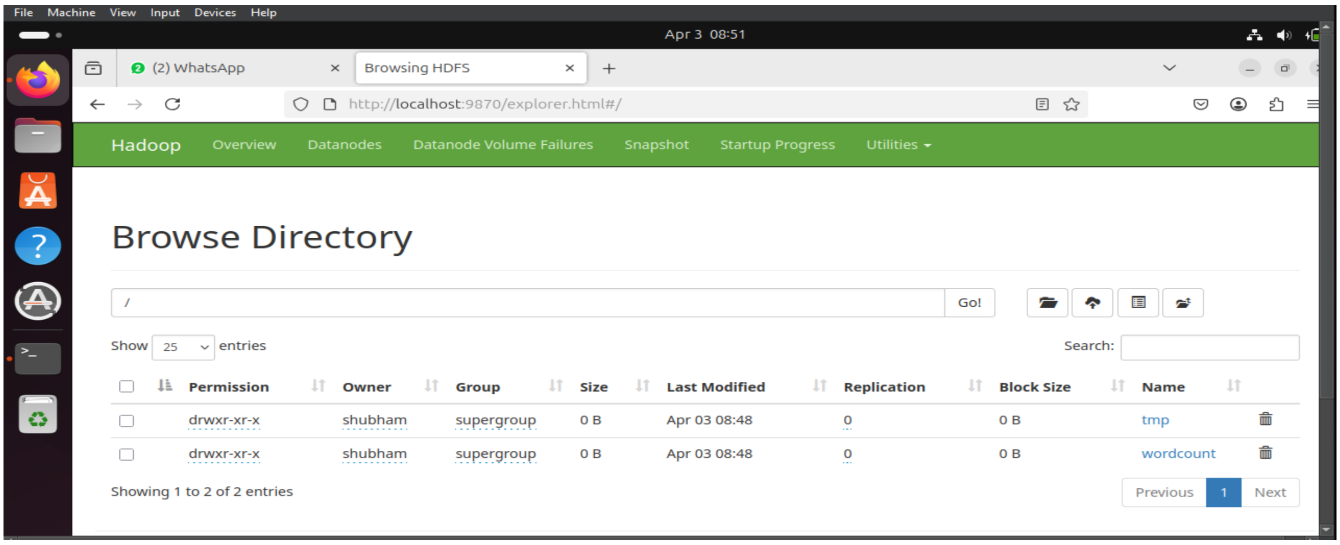
```
@ubuntu:~$ hadoop jar wordcount.jar WordCount /wordcount/input /wordcount/output
-03 08:48:28,587 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.
-03 08:48:29,583 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed.
l interface and execute your application with ToolRunner to remedy this.
-03 08:48:29,615 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yar
/.staging/job_1743669472086_0001
-03 08:48:30,308 INFO input.FileInputFormat: Total input files to process : 1
-03 08:48:30,500 INFO mapreduce.JobSubmitter: number of splits:1
-03 08:48:30,958 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1743669472086_0001
-03 08:48:30,958 INFO mapreduce.JobSubmitter: Executing with tokens: []
-03 08:48:31,470 INFO conf.Configuration: resource-types.xml not found
```

RESULTS AND DISCUSSION

The output of the program provides a count of each unique word in the dataset. This includes transaction IDs, dates, product categories, and sales amounts. The results demonstrate the efficiency of Hadoop MapReduce in processing structured text data, highlighting its potential for large-scale text analysis applications



```
@ubuntu:~$ hdfs dfs -cat /wordcount/output/part-r-00000
 1
 1
 2
 2
 1
 1
 1
 1
 1
 1
 1
 1
 1
-01       2
-02       2
-03       3
-04       3
 1
 1
 1
 1
```


File Machine View Input Devices Help
Apr 3 08:51
(2) WhatsApp
Browsing HDFS
http://localhost:9870/explorer.html#/
Hadoop Overview Datanodes Datanode Volume Failures Snapshot Startup Progress Utilities

Browse Directory

/ Go!

Show 25 entries Search:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	shubham	supergroup	0 B	Apr 03 08:48	0	0 B	tmp
drwxr-xr-x	shubham	supergroup	0 B	Apr 03 08:48	0	0 B	wordcount

Showing 1 to 2 of 2 entries

Previous 1 Next

CONCLUSION

This project successfully implemented a Word Count program using Hadoop MapReduce to analyze structured sales transaction data. The results illustrate Hadoop's capability to handle distributed data processing efficiently. Through the use of the MapReduce framework, the program demonstrated the advantages of parallel computing in efficiently analyzing and extracting insights from large datasets. The experiment confirmed the scalability of Hadoop, proving its effectiveness in handling structured data with high computational efficiency.

The implementation of this project opens doors for further research and development in big data analytics. The experiment can be extended to more complex data analytics tasks such as sentiment analysis, customer behavior prediction, and trend analysis using big data technologies. Future improvements could involve integrating additional Hadoop ecosystem components such as Apache Hive for SQL-based querying, Apache Pig for data transformation, and Apache Spark for faster in-memory processing. Furthermore, the methodology applied in this project can be utilized in real-world business intelligence applications, assisting organizations in making data-driven decisions based on large-scale data insights.

Overall, this project highlights the significance of Hadoop MapReduce in handling structured data efficiently and provides a foundation for further advancements in big data analytics. The findings demonstrate the potential of distributed computing frameworks in addressing the challenges posed by vast and continuously growing datasets in various industries.

References

1. Dean, J., & Ghemawat, S. (2008). "MapReduce: Simplified Data Processing on Large Clusters." Communications of the ACM, 51(1), 107-113.
2. White, T. (2015). *Hadoop: The Definitive Guide*. O'Reilly Media.
3. Apache Hadoop Documentation: https://hadoop.apache.org/docs/stable/
4. Lin, J., & Dyer, C. (2010). *Data-Intensive Text Processing with MapReduce*. Morgan & Claypool Publishers.
5. Shvachko, K., Kuang, H., Radia, S., & Chansler, R. (2010). "The Hadoop Distributed File System." *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, 1-10.
6. Borthakur, D. (2008). "The Hadoop Distributed File System: Architecture and Design." Apache Software Foundation.
7. Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauley, M., Franklin, M. J., Shenker, S., & Stoica, I. (2012). "Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing." *Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*.
8. Gittens, A., & Eaton, C. (2012). *Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data*. IBM Press.