Course	MSc in Data Science (CW_KCDAR_MY5)	Lecturer	Michael Gleeson
Module	Infrastructure for BigData (DATAC5201)	Student	Gokul Thillainathan
Assignment	: Big Data Solution – CA3		

HADOOP MULTINODE CLUSTER WITH SPARK

1. PROBLEM STATEMENT:

The Motto of this is to develop and implement a scalable and fault-tolerant infrastructure for data processing and analysis. As a result, it will improve

- The performance by supporting parallel and distributed computing for higher performance while managing larger datasets.
- Ensure the availability to continuously access the data even during node failures through multiple nodes.
- Improved response time for both batch processing and real-time data analytics.

The solution addresses the limitation of old single server design, providing improved scalability, reliability and processing efficiently the datasets.

2. SOLUTION ARCHITECTURE:

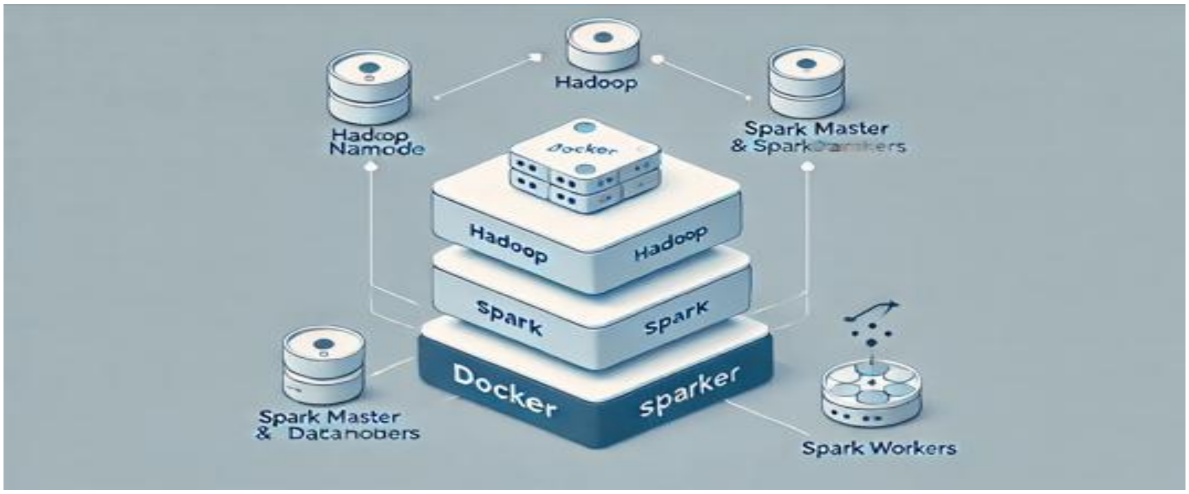
To address the issue, I have used Hadoop Ecosystem (HDFS, SPARK, ZEPPELIN) on a Docker-based setup with following components:

Hadoop Cluster: A multinode HDFS cluster to handle data.

Spark: For processing the data.

Zeppelin: for data visualization and executing query.

Docker: Containers to deploy Hadoop, Spark, and Zepplin services.



3. INFRASTRUCTURE DETAILS:

Docker Configuration:

The Hadoop cluster was deployed using docker compose with the following services:

- Namenode: Master node managing metadata
- Datanode: Two nodes for data storage
- Spark Master: Controller for spark jobs
- Spark Workers: for processing (2 nodes)
- Zeppelin: for running spark queries and visualizations

4. PREREQUISITES:

--> Docker and Docker compose installed on the host machine.

--> Dataset: quakes-cleaned.csv (Earthquake data) stored locally before ingestion.

5. STEP-BY-STEP IMPLEMENTATION:

Step 1:

- Deploy the Hadoop, spark, and zeppelin services using Docker compose:
  docker-compose up –d
- Verify that all the containers are running:
  docker ps

Step 2:

- Copy dataset to Namenode container:

  docker cp /path/quakes-cleaned.csv namenode:/quakes-cleaned.csv
- Exec the namenode container:

  docker exec –it namenode bash
- Upload the dataset to HDFS:

  hdfs dfs -put /quakes-cleaned.csv /iot_data/
- Verify the upload:

  hdfs dfs -ls /iot_data/

Step 3:

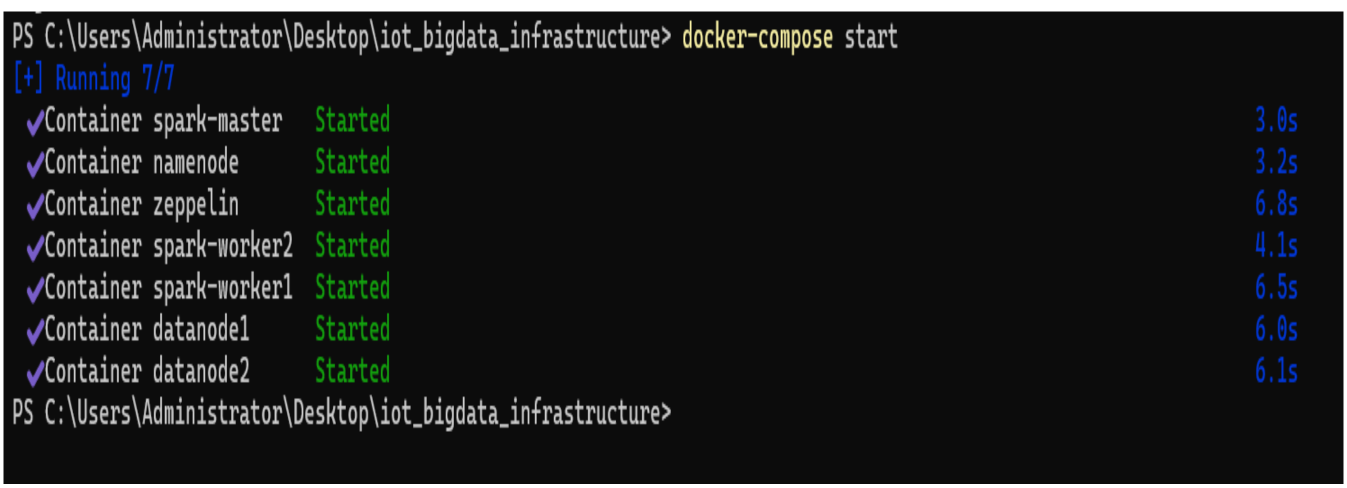
- Open the Zeppelin web interface (http://localhost:8085) and create a new notebook.
- Load the dataset into spark
- Perform aggregation (average magnitude by location)

Step 4:

- Save the aggregated data.
- Verify the data

6. RESULTS:

Starting container in the docker:

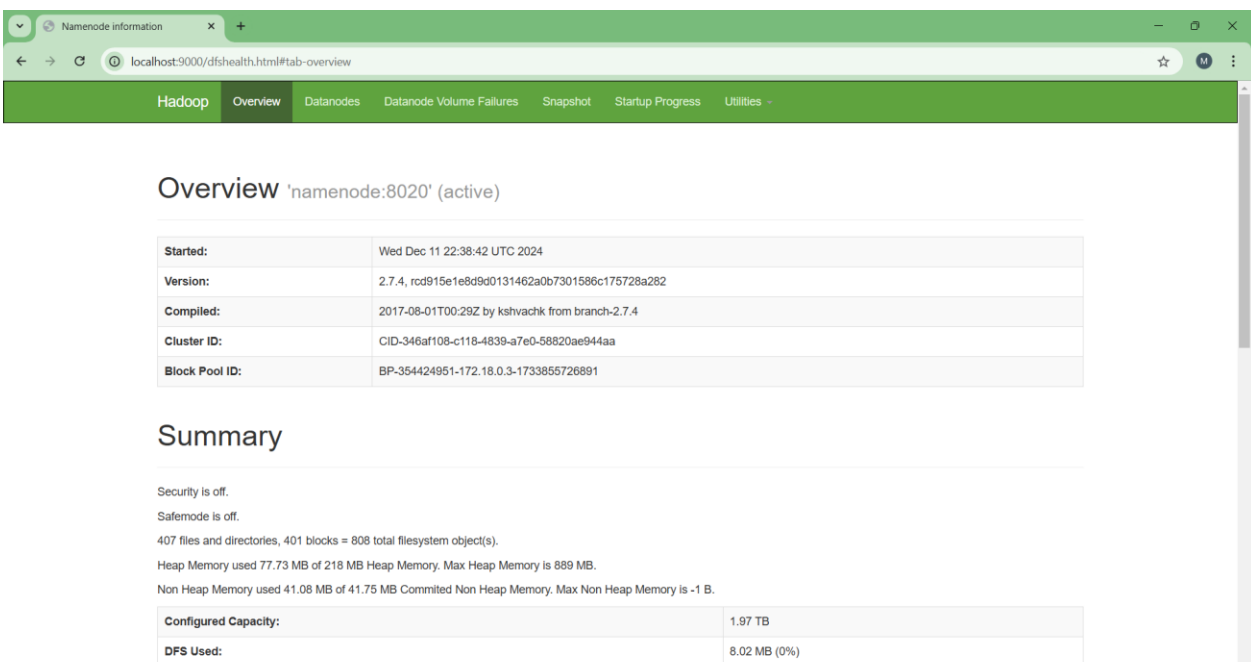


Container status:

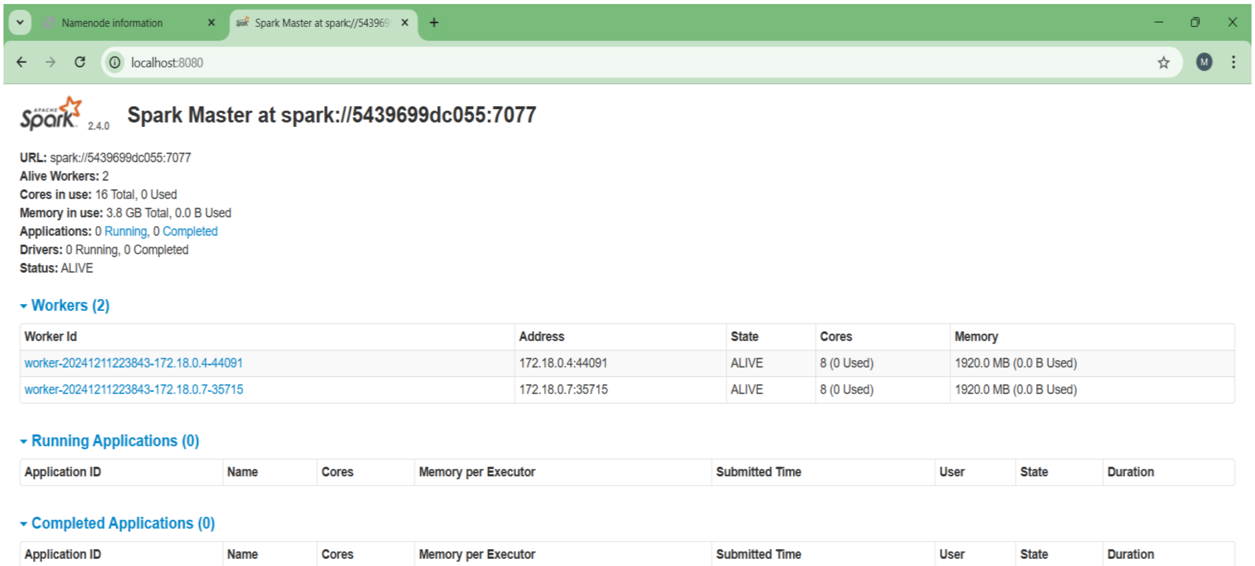
```
PS C:\Users\Administrator\Desktop\iot_bigdata_infrastructure> docker ps
CONTAINER ID   IMAGE                                                COMMAND                  CREATED        STATUS                   PORTS
                       NAMES
eda69365c700   apache/zeppelin:0.9.0                                "/usr/bin/tini -- bi…"   26 hours ago   Up 2 minutes             0.0.0.0:8085->8080/tcp
                       zeppelin
178dc3c41a12   bde2020/spark-worker:2.4.0-hadoop2.7                 "/bin/bash /worker.sh"   26 hours ago   Up 2 minutes             0.0.0.0:8081->8081/tcp
                       spark-worker1
fcef26726af6   bde2020/spark-worker:2.4.0-hadoop2.7                 "/bin/bash /worker.sh"   26 hours ago   Up 2 minutes             0.0.0.0:8082->8081/tcp
                       spark-worker2
73b9766c050b   bde2020/hadoop-datanode:2.0.0-hadoop2.7.4-java8      "/entrypoint.sh /run…"   26 hours ago   Up 2 minutes (healthy)   0.0.0.0:9001->50075/tcp
                       datanode1
b4d2073957e6   bde2020/hadoop-datanode:2.0.0-hadoop2.7.4-java8      "/entrypoint.sh /run…"   26 hours ago   Up 2 minutes (healthy)   0.0.0.0:9002->50075/tcp
                       datanode2
5439699dc055   bde2020/spark-master:2.4.0-hadoop2.7                 "/bin/bash /master.sh"   26 hours ago   Up 2 minutes             0.0.0.0:7077->7077/tcp, 60
66/tcp, 0.0.0.0:8080->8080/tcp   spark-master
aeb389f6997b   bde2020/hadoop-namenode:2.0.0-hadoop2.7.4-java8      "/entrypoint.sh /run…"   26 hours ago   Up 2 minutes (healthy)   0.0.0.0:8020->8020/tcp, 0.
0.0.0:9000->50070/tcp          namenode
```

Accessing the webpage:

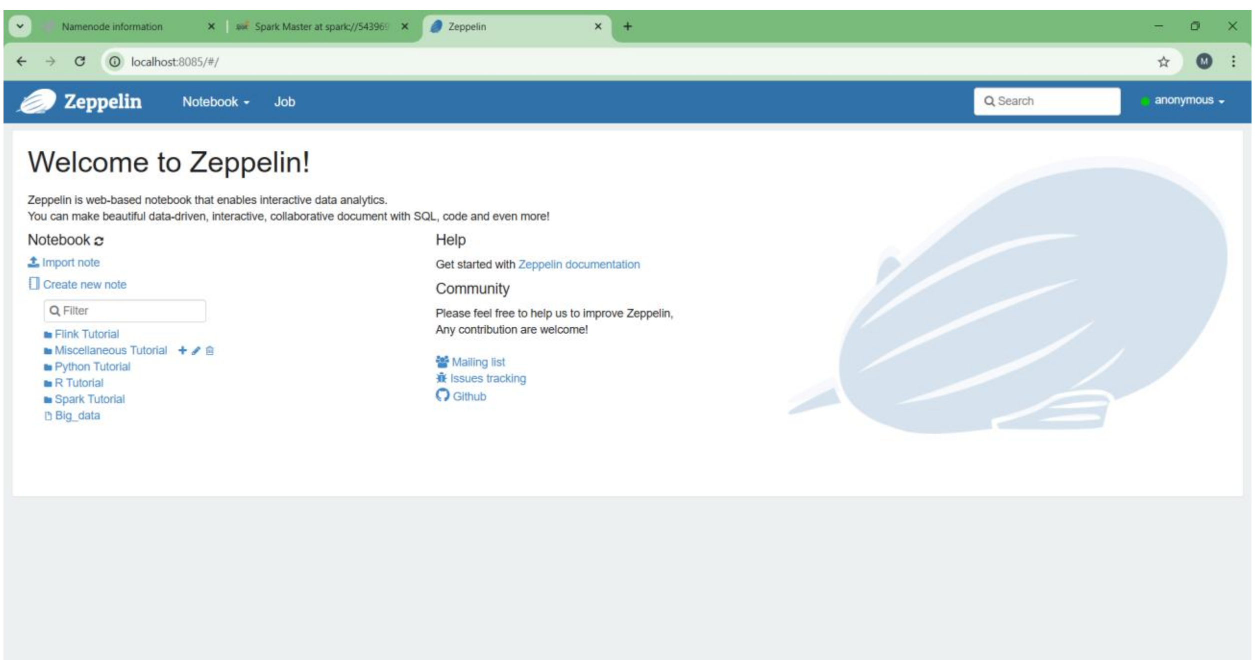
Hadoop Namenode: http://localhost:9000/



Spark: http://localhost:8088/



Zeppelin: http://localhost:8085/



Verify Dataset in the container:

```
PS C:\Users\Administrator\Desktop\iot_bigdata_infrastructure> docker exec -it namenode bash
root@aeb389f6997b:/# hdfs dfs -ls /iot_data/
Found 3 items
drwxr-xr-x   - zeppelin zeppelin          0 2024-12-10 19:12 /iot_data/aggregated_quakes_data
drwxr-xr-x   - zeppelin zeppelin          0 2024-12-10 21:15 /iot_data/filtered_quakes_data
-rwxrwxr-x   3 zeppelin zeppelin     845738 2024-12-10 18:56 /iot_data/quakes-cleaned.csv
root@aeb389f6997b:/# 
```

Code to load dataset to Spark:

```
%spark

val csvDF = spark.read.option("header", "true")

                      .option("inferSchema", "true")

                      .csv("hdfs://namenode:8020/iot_data/quakes-cleaned.csv")

csvDF.show(truncate = false)
```

Code to perform filter:

```
%spark

import org.apache.spark.sql.functions.avg


val aggDF = csvDF.groupBy("place")

  .agg(avg("mag").alias("avg_mag"))

aggDF.show(truncate = false)
```

Save the processed data back to HDFS:

%spark

aggDF.write.option("header", "true").csv("hdfs://namenode:8020/iot_data/aggregated_quakes_data")

Verify the dataset:

hdfs dfs -ls /iot_data/aggregated_quakes_data

```
root@aeb389f6997b:/# hdfs dfs -ls /iot_data/aggregated_quakes_data
Found 201 items
-rw-r--r--   3 zeppelin zeppelin          0 2024-12-10 19:12 /iot_data/aggregated_quakes_data/_SUCCESS
-rw-r--r--   3 zeppelin zeppelin       1013 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00000-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin       1142 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00001-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        654 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00002-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        768 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00003-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin       1100 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00004-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin       1083 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00005-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        702 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00006-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin       1115 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00007-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        415 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00008-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        788 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00009-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        782 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00010-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        938 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00011-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin       1139 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00012-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        768 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00013-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        902 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00014-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        591 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00015-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin       1024 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00016-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin       1298 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00017-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        803 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00018-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        890 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00019-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin       1197 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00020-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        874 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00021-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        479 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00022-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin       1171 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00023-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        919 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00024-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        701 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00025-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        521 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00026-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        750 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00027-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin        903 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00028-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
-rw-r--r--   3 zeppelin zeppelin       1197 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00029-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
```

7. CHALLENGES AND SOLUTIONS:

--> Permission issues: Resolved by setting appropriate permissions on HDFS directories.

--> Replication Factor: Ensured data replication across both datanodes.

8. FUTURE ENHANCEMENTS:

* Automating the data Ingestion.

* Real-time processing using spark streaming.

* Integrating jupyter Notebook for advanced workflows.

9. CONCLUSION:

In this project successfully implemented a scalable and fault-tolerant Big Data infrastructure using Hadoop, Spark, and Zeppelin. The infrastructure also lays the base for the future enhancements which includes real-time data processing and advanced data analytics.

10. REFERENCES:

https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html

https://www.tutorialkart.com/apache-spark/how-to-setup-an-apache-spark-cluster/

https://princetonits.com/blog/technology/how-to-configure-replication-factor-and-block-size-for-hdfs/

https://spark.apache.org/docs/latest/spark-standalone.html

https://medium.com/@ARishi/optimizing-apache-spark-executors-for-improved-performance-067eea2349e2