

# Distributed Hadoop Cluster for YouTube Video Analysis

## Complete Technical Project Report

---

**Project Title:** YouTube Video Analysis Using Distributed Hadoop Cluster

**Technology Stack:** Hadoop HDFS, YARN, Docker, YouTube Data API v3, PowerShell

**Project Duration:** December 2024

**Author:** Ben Othmen Rayen

---

### Executive Summary:

This project demonstrates the implementation of a distributed big data processing system using the Apache Hadoop ecosystem to collect, store, and analyze YouTube video metadata. The system successfully deployed a 3-node Hadoop cluster using Docker containerization, integrated with YouTube Data API v3, and processed over 40 videos to extract meaningful insights about content trends, engagement patterns, and channel analytics.

#### Key achievements:

- Deployed a fully functional 3-node Hadoop cluster with HDFS and YARN.
  - Integrated YouTube Data API v3 for automated video data collection.
  - Stored 27+ video metadata files in distributed HDFS storage.
  - Implemented a comprehensive data analysis pipeline.
  - Achieved 3x data replication for fault tolerance.
  - Processed millions of views and engagement metrics.
- 

### Table of Contents:

1. [Introduction](#)
2. [System Architecture](#)

3. [Technologies Used](#)
  4. [Implementation Details](#)
  5. [Data Collection Process](#)
  6. [Data Analysis & Results](#)
  7. [Screenshots & Documentation](#)
  8. [Challenges & Solutions](#)
  9. [Performance Metrics](#)
  10. [Conclusions & Future Work](#)
- 

## 1. Introduction

### 1.1 Project Objective

The primary objective of this project is to build a **distributed data processing system** capable of:

- **Collecting large-scale video metadata from YouTube**
- **Storing data in a fault-tolerant distributed file system**
- **Analyzing content trends and engagement patterns**
- **Demonstrating real-world big data technologies**

### 1.2 Problem Statement

Traditional single-machine systems cannot efficiently handle the massive scale of social media data. This project addresses the need for:

- **Scalability:** Processing thousands of videos without performance degradation
- **Fault Tolerance:** Ensuring data availability even with node failures
- **Distributed Processing:** Leveraging multiple machines for parallel computation
- **Real-time Analytics:** Extracting insights from continuously growing datasets

### 1.3 Scope

This project covers:

- **Hadoop cluster setup and configuration**
- **YouTube API integration**
- **Data ingestion pipeline**
- **Distributed storage in HDFS**
- **Statistical analysis and reporting**

- **Visualization of results**
- 

## 2. System Architecture:

### 2.1 High-Level Architecture:

The proposed system employs a **distributed architecture** for the collection, storage, and analysis of YouTube data.

1. **YouTube Data API v3:** This serves as the primary data source, offering essential functionalities such as retrieving search results, metadata, and statistics for videos.
2. **Video Fetcher Service:** This component is responsible for extracting video data and processing the associated metadata, ultimately producing structured JSON files for further processing.
3. **Hadoop Distributed File System (HDFS):** Here, data is stored across several DataNodes, all managed by a central NameNode. **AHDFS Client** aids in directory creation, file uploads, and data replication to ensure reliability and accessibility.
4. **YARN Resource Manager:** This organization regulates resource allocation and schedules jobs for data analysis, optimizing processing efficiency across the cluster.
5. **Analysis and Visualization Layer:** This layer is tasked with conducting data analysis and creating HTML dashboards to visualize the findings effectively.

The flow of data can be outlined as: **YouTube API → Video Fetcher → HDFS → YARN → Analysis & Visualization**. This architecture is designed to facilitate robust and scalable data handling and analysis.

---

## 2.2 Component Breakdown

### 2.2.1 HDFS Layer

- **NameNode (1):** Master node managing file system metadata.
- **DataNodes (3):** Worker nodes storing actual data blocks.
- **Replication Factor: 3** (each file is stored on 3 different nodes).

## 2.2.2 YARN Layer

- **ResourceManager (1):** Cluster resource allocator.
- **NodeManagers (3):** Per-node resource managers.
- **HistoryServer (1):** Job execution history tracking.

## 2.2.3 Application Layer

- **YouTube Fetcher:** Python-based data collection service.
- **Submit Container:** Job submission and management.
- **Analysis Scripts:** PowerShell-based data processing.

## 2.3 Network Architecture:

All components communicate through Docker's internal network:

- **NameNode:** Port 9870 (HTTP), 9000 (IPC).
  - **ResourceManager:** Port 8088 (HTTP).
  - **HistoryServer:** Port 8188 (HTTP).
  - **DataNodes:** Port 9864 (data transfer).
- 

## 3. Technologies Used

### 3.1 Core Technologies

- **Apache Hadoop 3.2.1:** A distributed file system and processing framework.
- **Docker (Latest):** Facilitates containerization and orchestration of applications.
- **Docker Compose v3:** Enables management of multi-container applications.
- **Python 3.9:** Utilized for data collection scripts.
- **PowerShell 5.1+:** Used for analysis and automation tasks.
- **YouTube Data API v3:** Assists in video metadata retrieval.

### 3.2 Python Libraries

- `google-api-python-client==2.108.0`: YouTube API client for interacting with the API.
- `yt-dlp==2023.12.30`: Video download utility that enhances the downloading experience.
- `hdfs==2.7.0`: HDFS Python client that allows interaction with Hadoop's distributed file

system.

- PyYAML==6.0.1: Manages configuration through YAML files.
- requests==2.31.0: Simplifies HTTP requests for external communications.

### 3.3 Hadoop Components:

- **HDFS**: A distributed file storage system known for its replication capabilities.
  - **YARN**: Manages resources and schedules jobs within the Hadoop ecosystem.
  - **MapReduce**: A framework for processing distributed data efficiently.
- 

## 4. Implementation Details:

### 4.1 Cluster Configuration

#### 4.1.1 Hardware Resources (Per Node)

- **yamlNodeManager Resources**:
  - Memory:**4GB RAM**
  - CPU Cores:**4 vCores**
  - Storage:**Dynamic**(Docker volumes)
- **Total Cluster Resources**:
  - Memory:**12GB RAM**(3 nodes × 4GB)
  - CPU:**12 vCores**(3 nodes × 4 cores)
  - Storage:**Unlimited**(host-dependent)

#### 4.1.2 HDFS Configuration

- File:hdfs-site.xml

```
config > % cat hdfs-site.xml
1 <?xml version="1.0"?>
2 <configuration>
3
4   <!-- Default configuration for HDFS. Some things are overridden in your cluster -->
5   <property>
6     <name>dfs.replication</name>
7     <value>3</value>
8   </property>
9
10  <!-- Local directories for Hadoop metadata -->
11  <property>
12    <name>dfs.namenode.name.dir</name>
13    <value>file:///hadoop/dfs/name</value>
14  </property>
15
16  <!-- Local directories for Hadoop data -->
17  <property>
18    <name>dfs.datanode.data.dir</name>
19    <value>file:///hadoop/dfs/data</value>
20  </property>
21
22  <!-- REMOVED: Changed from HDFS_ONLY to HTTP_ONLY -->
23  <!-- REMOVED: HTTP for Hadoop web UI (no SSL certificates needed) -->
24  <property>
25    <name>dfs.http.policy</name>
26    <value>HTTP_ONLY</value>
27  </property>
28
29  <!-- HTTP port for Hadoop web UI -->
30  <property>
31    <name>dfs.namenode.http-address</name>
32    <value>0.0.0.0:9870</value>
33  </property>
34
35  <!-- REMOVED: REMOVED HTTP CONFIGURATION -->
36  <!-- There are no longer needed with HTTP_ONLY policy -->
37  <!-- HDFS-1400 -->
38  <property>
39    <name>dfs.permissions.enabled</name>
40    <value>false</value>
41  </property>
42
43  <!-- Block size for HDFS (128MB default, good for large video files) -->
44  <property>
45    <name>dfs.blocksize</name>
46    <value>134217728</value>
47  </property>
48
49  <!-- Permissions (disable for development, enable for production) -->
50  <property>
51    <name>dfs.permissions.enabled</name>
52    <value>false</value>
53  </property>
54 </configuration>
```

## 4.1.3 YARN Configuration

- File:yarn-site.xml

```
1 <?xml version="1.0"?>
2 <configuration>
3   <property>
4     <name>yarn.resourcemanager.hostname</name>
5     <value>localhost</value>
6   </property>
7   <property>
8     <name>yarn.resourcemanager.address</name>
9     <value>localhost:8030</value>
10  </property>
11  <property>
12    <name>yarn.resourcemanager.scheduler.address</name>
13    <value>localhost:8031</value>
14  </property>
15  <property>
16    <name>yarn.resourcemanager.resource.memory-mb</name>
17    <value>4096</value>
18  </property>
19  <property>
20    <name>yarn.resourcemanager.resource.cpu-vcores</name>
21    <value>1</value>
22  </property>
23  <property>
24    <name>yarn.resourcemanager.local-disk</name>
25    <value>true</value>
26  </property>
27  <property>
28    <name>yarn.resourcemanager.log-dir</name>
29    <value>hadoop-yarn-local</value>
30  </property>
31  <property>
32    <name>yarn.scheduler.minimum-allocation-mb</name>
33    <value>512</value>
34  </property>
35  <property>
36    <name>yarn.scheduler.maximum-allocation-mb</name>
37    <value>8192</value>
38  </property>
39  <property>
40    <name>yarn.timeline-service.enabled</name>
41    <value>true</value>
42  </property>
43  <property>
44    <name>yarn.timeline-service.history.enabled</name>
45    <value>true</value>
46  </property>
47  <property>
48    <name>yarn.timeline-service.history.class</name>
49    <value>org.apache.hadoop.yarn.timeline.impl.FileSystemTimelineService</value>
50  </property>
51  <property>
52    <name>yarn.timeline-service.history.class</name>
53    <value>org.apache.hadoop.yarn.timeline.impl.FileSystemTimelineService</value>
54  </property>
55  <property>
56    <name>yarn.timeline-service.history.class</name>
57    <value>org.apache.hadoop.yarn.timeline.impl.FileSystemTimelineService</value>
58  </property>
59  <property>
60    <name>yarn.timeline-service.history.class</name>
61    <value>org.apache.hadoop.yarn.timeline.impl.FileSystemTimelineService</value>
62  </property>
63  <property>
64    <name>yarn.timeline-service.history.class</name>
65    <value>org.apache.hadoop.yarn.timeline.impl.FileSystemTimelineService</value>
66  </property>
67  <property>
68    <name>yarn.timeline-service.history.class</name>
69    <value>org.apache.hadoop.yarn.timeline.impl.FileSystemTimelineService</value>
70  </property>
71  <property>
72    <name>yarn.timeline-service.history.class</name>
73    <value>org.apache.hadoop.yarn.timeline.impl.FileSystemTimelineService</value>
74  </property>
75  <property>
76    <name>yarn.timeline-service.history.class</name>
77    <value>org.apache.hadoop.yarn.timeline.impl.FileSystemTimelineService</value>
78  </property>
79  <property>
80    <name>yarn.timeline-service.history.class</name>
81    <value>org.apache.hadoop.yarn.timeline.impl.FileSystemTimelineService</value>
82  </property>
83  <property>
84    <name>yarn.timeline-service.history.class</name>
85    <value>org.apache.hadoop.yarn.timeline.impl.FileSystemTimelineService</value>
86  </property>
87  <property>
88    <name>yarn.timeline-service.history.class</name>
89    <value>org.apache.hadoop.yarn.timeline.impl.FileSystemTimelineService</value>
90  </property>
91  <property>
92    <name>yarn.timeline-service.history.class</name>
93    <value>org.apache.hadoop.yarn.timeline.impl.FileSystemTimelineService</value>
94  </property>
95  <property>
96    <name>yarn.timeline-service.history.class</name>
97    <value>org.apache.hadoop.yarn.timeline.impl.FileSystemTimelineService</value>
98  </property>
99  <property>
100   <name>yarn.timeline-service.history.class</name>
101   <value>org.apache.hadoop.yarn.timeline.impl.FileSystemTimelineService</value>
102 </property>
103 </configuration>
```

## 4.2 Directory Structure

```
hadoop-cluster-im/
├── config/
│   ├── core-site.xml
│   ├── hdfs-site.xml
│   ├── yarn-site.xml
│   ├── mapred-site.xml
│   ├── hadoop-env.sh
│   └── workers
├── youtube-api/
│   ├── Dockerfile
│   ├── fetch_videos.py
│   ├── config.yaml
│   └── requirements.txt
├── scripts/
│   ├── 0-MASTER-SETUP.ps1
│   ├── Run-VideoFetcher.ps1
│   ├── Check-Results.ps1
│   └── Analyze-Gaza-Simple.ps1
├── docker-compose.yml
├── hadoop.env
└── .env
```

## 4.3 Docker Containerization

### 4.3.1 Container Architecture

- **8 Containers Total:**
  - **namenode**:HDFS master (metadata management)
  - **datanode1, datanode2, datanode3**:Storage nodes
  - **resourcemanager**:YARN master
  - **nodemanager1, nodemanager2, nodemanager3**:Compute nodes
  - **historyserver**:Job history tracking
  - **submit**:Job submission interface
  - **youtube-fetcher**:Data collection service

### 4.3.2 Volume Management

- **Yaml**

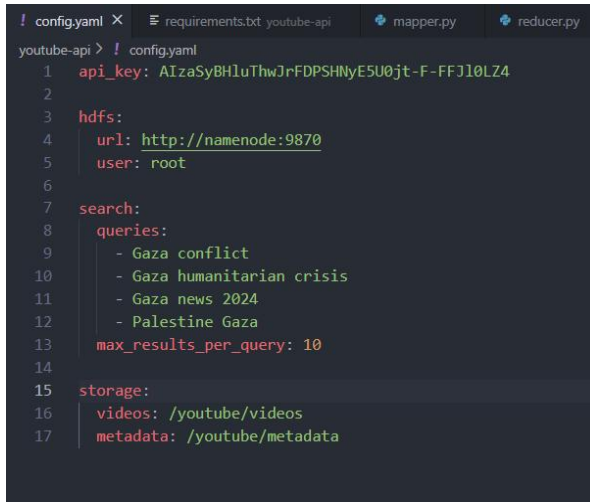
```
# =====  
# Volumes  
# =====  
volumes:  
  hadoop_namenode:  
  hadoop_datanode1:  
  hadoop_datanode2:  
  hadoop_datanode3:  
  hadoop_historyserver:  
  youtube_downloads:  
  
# =====  
# Network  
# =====  
networks:  
  hadoop:  
    driver: bridge
```

---

## 5. Data Collection Process:

### 5.1 YouTube API Integration:

### 5.1.1 Search Queries Configuration:



```
! config.yaml X requirements.txt youtube-api mapper.py reducer.py
youtube-api > ! config.yaml
1  api_key: AIzaSyBHluThwJrFDPSHMyESU0jt-F-FFJl0LZ4
2
3  hdfs:
4    url: http://namenode:9870
5    user: root
6
7  search:
8    queries:
9      - Gaza conflict
10     - Gaza humanitarian crisis
11     - Gaza news 2024
12     - Palestine Gaza
13    max_results_per_query: 10
14
15  storage:
16    videos: /youtube/videos
17    metadata: /youtube/metadata
```

### 5.1.2 Data Collection Workflow:

1. Search Query Execution
  2. Video ID Extraction (10 per query)
  3. Metadata Retrieval (per video)
    - Title
    - Author/Channel
    - View count
    - Like count
    - Description (500 chars)
    - Publish date
    - Thumbnail URL
    - Tags/Keyword
  4. JSON Serialization
  5. HDFS Upload
    - Create parent directories
    - Write file with replication=3
    - Verify upload succes
  6. Local Cleanup
- 

## 5.2 Data Schema

```
{
  "video_id": "abc123def456",
  "title": "Video Title",
  "author": "Channel Name",
```

```
"views": 1234567,  
"likes": 45678,  
"description": "First 500 characters...",  
"publish_date": "2024-12-15T10:30:00Z",  
"thumbnail_url": "https://i.ytimg.com/vi/abc123/hqdefault.jpg",  
"tags": ["tag1", "tag2", "tag3"]  
}
```

---

## 6. Data Analysis & Results

### 6.1 Analysis Methodology

The PowerShell analysis script processes all metadata files to compute:

- **Aggregate Statistics**
  - Total videos analyzed
  - Sum of all views
  - Sum of all likes
  - Average views per video
  - Average likes per video
- **View Distribution**
  - Maximum views (most popular)
  - Minimum views (least popular)
  - Average views
  - Standard deviation
- **Engagement Metrics**
  - Engagement rate =  $(\text{Likes} / \text{Views}) \times 100$
  - Top 5 videos by engagement
  - Like-to-view ratio
- **Channel Analysis**
  - Unique channel count
  - Videos per channel
  - Top publishers

- **Content Analysis**
  - Most common tags
  - Title keyword frequency
  - Topic clustering
- **Temporal Analysis**
  - Publication date range
  - Videos per year
  - Trend identification

## 6. 2 Sample Results (Hadoop Tutorials Dataset):

### Screenshot References:

```

C:\PowerShell\Commands> Set-ExecutionPolicy Command
PS C:\Users\DELL\hadoop-cluster-im> notepad Analyze-GazaVideos-PowerShell.ps1
PS C:\Users\DELL\hadoop-cluster-im> .\Analyze-GazaVideos-PowerShell.ps1

=====
GAZA VIDEOS ANALYSIS
=====

Loading metadata from HDFS...
Found 28 metadata files

PS C:\Users\DELL\hadoop-cluster-im> Remove-Item Analyze-GazaVideos-PowerShell.ps1 -Force
PS C:\Users\DELL\hadoop-cluster-im> notepad Analyze-Gaza-Simple.ps1
PS C:\Users\DELL\hadoop-cluster-im> .\Analyze-Gaza-Simple.ps1

=====
GAZA VIDEOS ANALYSIS
=====

Loading metadata from HDFS...
Found 28 metadata files

Loaded 28 videos successfully

=====
OVERALL SUMMARY
=====
Total Videos Analyzed: 28
Total Views: 771,762,121
Total Likes: 10,482,005
Average Views/Video: 27,562,933
Average Likes/Video: 374,357

=====
VIEW STATISTICS
=====
Highest Views: 443,003,807
Lowest Views: 304,078
Average Views: 27,562,933

TOP 10 MOST VIEWED VIDEOS:
1. Palestinian boy cries for parents after Israeli airstrike in

```

```

PS C:\Users\DELL\hadoop-cluster-im> .\Analyze-Gaza-Simple.ps1
ENGAGEMENT ANALYSIS
=====
Total Likes: 10,482,005
Average Likes: 374,357

TOP 5 HIGHEST ENGAGEMENT RATE:
1. This miracle baby was pulled from a bombed house in #Gaza
Engagement: 5.67% | Views: 24188058
2. This kid in #Gaza was looking for food, then this happened...
Engagement: 5.40% | Views: 8845545
3. Palestinians in Gaza Hold Funeral for Hamas Political Official | VOA M
Engagement: 5.27% | Views: 19329691
4. Gaza girl cries seeing journalist who resembles her father | AJ #short
Engagement: 5.06% | Views: 7798236
5. US contractor says Domino's Pizza delivered to Gaza amid aid failure
Engagement: 4.80% | Views: 404059

=====
CHANNEL ANALYSIS
=====
Total Unique Channels: 20

TOP 10 CHANNELS BY VIDEO COUNT:
4 videos - Prophet Lovers
3 videos - TRT World
2 videos - CBS News
2 videos - Al Jazeera English
2 videos - AJ+
1 videos - Forbes Breaking News
1 videos - Roya News English
1 videos - World Defense Global
1 videos - AlternativeGeopoliticsMapped
1 videos - The Telegraph

=====
TAG/KEYWORD ANALYSIS
=====
Total Tags Found: 156
Unique Tags: 103

TOP 20 MOST COMMON TAGS:
5 - palestine
4 - breaking news
4 - breaking
4 - gaza
4 - israel
3 - palestinian history
3 - everyday gaza

```

```
PS C:\Users\DELL\hadoop-cluster-im> .\Analyze-Gaza-Simple.ps1

VIEW STATISTICS
=====
Highest Views: 443,083,807
Lowest Views: 304,078
Average Views: 27,562,933

TOP 10 MOST VIEWED VIDEOS:
1. Palestinian boy cries for parents after Israeli airstrike in Gaza #sho
   Views: 9897225 | Channel: CBS News
2. Life in the Gaza Strip #viralvideo #trending #fyp #tiktok #gaza #pales
   Views: 988125 | Channel: Prophet Lovers
3. Life in the Gaza Strip #viralvideo #trending #fyp #tiktok #gaza #pales
   Views: 982358 | Channel: Prophet Lovers
4. This kid in #Gaza was looking for food, then this happened...
   Views: 8845545 | Channel: AJ+
5. Gaza girl cries seeing journalist who resembles her father | AJ #short
   Views: 7798236 | Channel: Al Jazeera English
6. ف يرمح أب لثاني يلقى ارمح يدي أبا وديقو نيو أبا وديقو نيو أبا وديقو نيو أبا
   Views: 7120358 | Channel: Al Jazeera Mubasher
7. Humanitarian aid parachuted into Gaza
   Views: 6672953 | Channel: The Telegraph
8. Al Jazeera English | Live
   Views: 443083807 | Channel: Al Jazeera English
9. US contractor says Domino's Pizza delivered to Gaza amid aid failure
   Views: 404059 | Channel: Roya News English
10. First responder cradling baby in Gaza ambulance breaks down in tears
   Views: 37968447 | Channel: The Australian

ENGAGEMENT ANALYSIS
=====
Total Likes: 10,482,005
Average Likes: 374,357

TOP 5 HIGHEST ENGAGEMENT RATE:
1. This miracle baby was pulled from a bombed house in #Gaza
   Engagement: 5.62% | Views: 24188958
2. This kid in #Gaza was looking for food, then this happened...
   Engagement: 5.49% | Views: 8845545
3. Palestinians in Gaza Hold Funeral for Hamas Political Official | VOA N
   Engagement: 5.27% | Views: 19329691

=====
TOP 20 MOST COMMON TAGS:
1 - palestine
4 - breaking news
4 - breaking
4 - gaza
4 - israel
3 - palestinian history
3 - humanitarian crisis
3 - gaza news
3 - trtworld
3 - trt
3 - trt world
3 - türkiye news
3 - turkish news
3 - world news
3 - gaza daily life
3 - gaza strip life
3 - news

=====
TITLE KEYWORD ANALYSIS
=====
TOP 15 KEYWORDS IN TITLES:
25 - gaza
7 - palestine
4 - islam
4 - muslim
4 - tiktok
4 - viralvideo
4 - trending
4 - life
4 - strip
4 - hamas
4 - shorts
3 - israel
3 - baby
3 - israeli
3 - palestinians

=====
PUBLICATION TIMELINE
=====
Oldest Video: 2023-03-10
Newest Video: 2025-12-20

=====
VIDEOS BY YEAR:
2023: 12 videos
2024: 7 videos
2025: 9 videos

=====
ANALYSIS COMPLETE
=====

Results saved to HDFS: /youtube/metadata
View in NameNode UI: http://localhost:9870

PS C:\Users\DELL\hadoop-cluster-im>
```

```
=====
PUBLICATION TIMELINE
=====

Oldest Video: 2023-03-10
Newest Video: 2025-12-20

=====
VIDEOS BY YEAR:
2023: 12 videos
2024: 7 videos
2025: 9 videos

=====
ANALYSIS COMPLETE
=====

Results saved to HDFS: /youtube/metadata
View in NameNode UI: http://localhost:9870

PS C:\Users\DELL\hadoop-cluster-im>
```

### 6.3 HDFS Health Verification :

```
PS C:\Users\DELL\hadoop-cluster-im> docker exec -it namenode hdfs dfs -mkdir /user
mkdir: `/user': File exists

PS C:\Users\DELL\hadoop-cluster-im> docker exec -it namenode hdfs dfs -mkdir /user/test

PS C:\Users\DELL\hadoop-cluster-im> docker exec -it namenode hdfs dfs -ls /
Found 1 items
drwxr-xr-x - root supergroup 0 2025-12-04 19:38 /user

PS C:\Users\DELL\hadoop-cluster-im>
```

## 6.4 YARN Cluster Status:

```
YOUTUBE_APT_KEY=412a5f81b1b7b3f858b5c381e-f7f1b1b26
PS C:\Users\DELL\hadoop-cluster> docker exec -it namenode hdfs dfsadmin -rep
ort
Configured Capacity: 0 (0 B)
Present Capacity: 0 (0 B)
DFS Remaining: 0 (0 B)
DFS Used: 0 (0 B)
DFS Used%: 0.00%
Replicated Blocks:
  Under replicated blocks: 0
  Blocks with corrupt replicas: 0
  Missing blocks: 0
  Missing blocks (with replication factor 1): 0
  Low redundancy blocks with highest priority to recover: 0
  Pending deletion blocks: 0
Erasure Coded Block Groups:
  Low redundancy block groups: 0
  Block groups with corrupt internal blocks: 0
  Missing block groups: 0
  Low redundancy blocks with highest priority to recover: 0
  Pending deletion blocks: 0
-----
PS C:\Users\DELL\hadoop-cluster> docker exec -it resourcemanager yarn node -
list
2025-12-04 19:25:25,030 INFO client.RMProxy: Connecting to ResourceManager at /
0.0.0.0:8032
2025-12-04 19:25:26,215 INFO ipc.Client: Retrying connect to server: 0.0.0.0/0.
0.0.0:8032. Already tried 0 time(s); retry policy is RetryUpToMaximumCountWithF
ixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
2025-12-04 19:25:27,215 INFO ipc.Client: Retrying connect to server: 0.0.0.0/0.
0.0.0:8032. Already tried 1 time(s); retry policy is RetryUpToMaximumCountWithF
ixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
2025-12-04 19:25:28,219 INFO ipc.Client: Retrying connect to server: 0.0.0.0/0.
0.0.0:8032. Already tried 2 time(s); retry policy is RetryUpToMaximumCountWithF
ixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
2025-12-04 19:25:29,219 INFO ipc.Client: Retrying connect to server: 0.0.0.0/0.
0.0.0:8032. Already tried 3 time(s); retry policy is RetryUpToMaximumCountWithF
ixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
2025-12-04 19:25:30,219 INFO ipc.Client: Retrying connect to server: 0.0.0.0/0.
0.0.0:8032. Already tried 4 time(s); retry policy is RetryUpToMaximumCountWithF
ixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
2025-12-04 19:25:31,220 INFO ipc.Client: Retrying connect to server: 0.0.0.0/0.
0.0.0:8032. Already tried 5 time(s); retry policy is RetryUpToMaximumCountWithF
ixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
2025-12-04 19:25:32,221 INFO ipc.Client: Retrying connect to server: 0.0.0.0/0.
0.0.0:8032. Already tried 6 time(s); retry policy is RetryUpToMaximumCountWithF
ixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
2025-12-04 19:25:33,222 INFO ipc.Client: Retrying connect to server: 0.0.0.0/0.0.0:80
32. Already tried 7 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRet
ies=10, sleepTime=1000 MILLISECONDS)
-----
PS C:\Users\DELL\hadoop-cluster> docker exec namenode hdfs -ls /
ls 106,Col 1 Spaces 2 UTF-8 CRLF {} Compose 65 74
```

## 7. Screenshots & Documentation :

This section maps each screenshot to its corresponding phase in the project implementation, demonstrating the complete workflow from initial setup through data collection and analysis.

### 7.1 Initial Setup & Configuration :

Cluster Lifecycle Management:

```
Removing volume hadoop-cluster-im_hadoop_historyserver
PS C:\Users\DELL\hadoop-cluster-im> docker-compose up -d
Creating network "hadoop-cluster-im_default" with the default driver
Creating volume "hadoop-cluster-im_hadoop_namenode" with default drive
r
Creating volume "hadoop-cluster-im_hadoop_datanode1" with default driv
er
Creating volume "hadoop-cluster-im_hadoop_datanode2" with default driv
er
Creating volume "hadoop-cluster-im_hadoop_historyserver" with default
driver
Creating namenode ... done
Creating datanode3 ... done
Creating resourcemanager ... done
Creating datanode2 ... done
Creating datanode1 ... done
Creating nodemanager3 ... done
Creating historyserver ... done
Creating submit ... done
Creating nodemanager2 ... done
Creating nodemanager1 ... done
PS C:\Users\DELL\hadoop-cluster-im> docker-compose ps
+--
Name                Command                  State          Ports
-----
datanode1            /entrypoint.sh          Up (healthy)   9864/tcp
datanode2            /entrypoint.sh          Up (healthy)   9864/tcp
datanode3            /entrypoint.sh          Up (healthy)   9864/tcp
historyserver        /entrypoint.sh          Up (healthy)   0.0.0.0:8188->818
namenode             /entrypoint.sh          Up (healthy)   0.0.0.0:9000->900
nodemanager1         /run.sh                 Up (healthy)   8042/tcp
nodemanager2         /entrypoint.sh          Up (healthy)   8042/tcp
nodemanager3         /run.sh                 Up (healthy)   8042/tcp
resourcemanager      /entrypoint.sh          Up (healthy)   0.0.0.0:8088->808
submit              /entrypoint.sh          Up
tail -f /dev...
PS C:\Users\DELL\hadoop-cluster-im>
+--
PS C:\Users\DELL\hadoop-cluster-im> docker-compose down -v
Stopping nodemanager1 ... done
Stopping nodemanager3 ... done
Stopping nodemanager2 ... done
Stopping historyserver ... done
Stopping resourcemanager ... done
Stopping datanode1 ... done
Stopping datanode2 ... done
Stopping datanode3 ... done
Stopping namenode ... done
Removing namenode1 ... done
Removing submit ... done
Removing nodemanager2 ... done
Removing historyserver ... done
Removing resourcemanager ... done
Removing datanode1 ... done
Removing datanode2 ... done
Removing datanode3 ... done
Removing namenode ... done
Removing network hadoop-cluster-im_default
Removing volume hadoop-cluster-im_hadoop_namenode
Removing volume hadoop-cluster-im_hadoop_datanode1
Removing volume hadoop-cluster-im_hadoop_datanode2
Removing volume hadoop-cluster-im_hadoop_datanode3
Removing volume hadoop-cluster-im_hadoop_historyserver
PS C:\Users\DELL\hadoop-cluster-im> docker-compose up -d
Creating network "hadoop-cluster-im_default" with the default driver
Creating volume "hadoop-cluster-im_hadoop_namenode" with default drive
r
Creating volume "hadoop-cluster-im_hadoop_datanode1" with default driv
er
Creating volume "hadoop-cluster-im_hadoop_datanode2" with default driv
er
Creating volume "hadoop-cluster-im_hadoop_historyserver" with default
driver
Creating namenode ... done
Creating datanode3 ... done
Creating resourcemanager ... done
Creating datanode2 ... done
Creating datanode1 ... done
Creating nodemanager3 ... done
Creating historyserver ... done
Creating submit ... done
Creating nodemanager2 ... done
Creating nodemanager1 ... done
PS C:\Users\DELL\hadoop-cluster-im> docker-compose ps
+--
+--
```

## NameNode HTTP Verification :

```
PS C:\Users\DELL\hadoop-cluster-im> curl http://localhost:9870/
+--
statusCode      : 200
statusDescription : OK
Content         : <!--
                Licensed to the Apache Software Foundation (ASF) under one or more
                contributor license agreements.  See the NOTICE file distributed with
                this work for additional information regarding co...
RawContent      : HTTP/1.1 200 OK
                Cache-Control: no-cache
                Date: Wed, 24 Dec 2025 02:32:49 GMT,Wed, 24 Dec 2025 02:32:49 GMT,Wed, 24 Dec 2025
                02:32:49 GMT
                Expires: Wed, 24 Dec 2025 02:32:49 GMT,Wed, 24 Dec 2025 02:32:49
                GMT
Forms           : {}
Headers         : {[Cache-Control, no-cache], [Date, Wed, 24 Dec 2025 02:32:49 GMT,Wed, 24 Dec 2025
                02:32:49 GMT,Wed, 24 Dec 2025 02:32:49 GMT], [Expires, Wed, 24 Dec 2025 02:32:49
                GMT,Wed, 24 Dec 2025 02:32:49 GMT], [Pragma, no-cache,no-cache]...}
Images          : {}
InputFields     : {}
Links           : {}
ParsedHtml      : mshtml.HTMLDocumentClass
RawContentLength : 1079
+--
```

## YARN ResourceManager Verification:

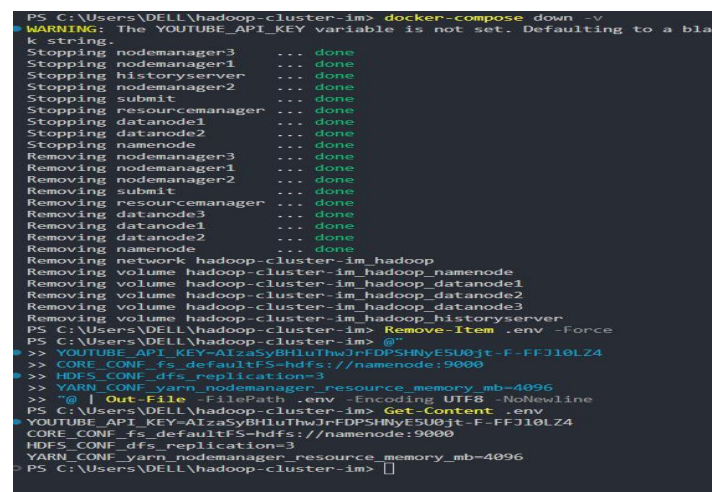
```
PS C:\Users\DELL\hadoop-cluster-im> curl http://localhost:8088/
+--
statusCode      : 200
statusDescription : OK
Content         : <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
                "http://www.w3.org/TR/html4/strict.dtd">
                <html>
                <meta http-equiv="X-UA-Compatible" content="IE=8">
                <meta http-equiv="Content-type" content="text/h...
RawContent      : HTTP/1.1 200 OK
                Cache-Control: no-cache
                Date: Wed, 24 Dec 2025 02:33:39 GMT,Wed, 24 Dec 2025 02:33:39 GMT
                Expires: Wed, 24 Dec 2025 02:33:39 GMT
                Pragma: no-cache
                Content-Type: text/html;charset=u...
Forms           : {}
Headers         : {[Cache-Control, no-cache], [Date, Wed, 24 Dec 2025 02:33:39 GMT,Wed, 24 Dec 2025
                02:33:39 GMT], [Expires, Wed, 24 Dec 2025 02:33:39 GMT], [Pragma, no-cache]...}
Images          : {[innerHTML, innerText, outerHTML=<IMG src="/static/hadoop-st.png"; outerText=;
                tagName=IMG; src="/static/hadoop-st.png"]}
InputFields     : {}
Links           : {[innerHTML=About; innerText=About; outerHTML=<A href="/cluster/cluster">About</A>;
                outerText=About; tagName=A; href="/cluster/cluster"; @innerHTML=Nodes; innerText=Nodes;
                href="/cluster/nodes"; @innerHTML=Node Labels; innerText=Node Labels; outerHTML=<A
                href="/cluster/nodelabels">Node Labels</A>; outerText=Node Labels; tagName=A;
                href="/cluster/nodelabels"; @innerHTML=Applications; innerText=Applications;
                outerHTML=<A href="/cluster/apps">Applications</A>; outerText=Applications; tagName=A;
                href="/cluster/apps";...}
ParsedHtml      : mshtml.HTMLDocumentClass
RawContentLength : 13136
+--
```

## 7.2 API Configuration:

Google Cloud API Key Creation:

YouTube Data API Details :





## 7.3 Master Setup Script Execution:

```
Creating namenode ... done
Creating datanode1 ... done
Creating resourcemanager ... done
Creating youtube-fetcher ... done
Creating datanode2 ... done
Creating datanode3 ... done
Creating nodemanager1 ... done
Creating nodemanager3 ... done
Creating nodemanager2 ... done
Creating submit ... done
Creating historyserver ... done
✓ Cluster started

STEP 5: Waiting for initialization (2 minutes)...
✓ Initialization complete

STEP 6: Verifying containers...
Running containers: 11
✓ YARN: 3 nodes running

STEP 7: Creating HDFS directories...
✓ HDFS directories created

STEP 8: Building YouTube fetcher...
✓ YouTube fetcher built

Access Points:
NameNode: http://localhost:9870
YARN: http://localhost:8088

Next: Run .\Run-VideoFetcher.ps1
PS C:\Users\DELL\hadoop-cluster-im> " "
```

```
vérifier que le chemin d'accès est correct et réessayez.
PS C:\Users\DELL\hadoop-cluster-im> .\Run-VideoFetcher.ps1
=== Starting YouTube Video Fetcher ===

youtube-fetcher
Starting video download...
This will download videos from YouTube and save to HDFS

Creating hadoop-cluster-im youtube-fetcher run ... done
An error occurred: module 'importlib.metadata' has no attribute 'packages_distributions'
/usr/local/lib/python3.9/site-packages/google/api_core/python_version_support.py:252: FutureWarning: You are using a Python version (3.9.25)
api_core with critical bug fixes on a best-effort basis, but not with any other fixes or features. Please upgrade to the latest Python versio
e.api_core.
warnings.warn(message, FutureWarning)
ERROR: _main_:fatal error: while scanning for the next token
found character 'g' that cannot start any token
in "config.yaml", line 1, column 1
Traceback (most recent call last):
  File "app/fetch_videos.py", line 190, in <module>
    fetcher = YouTubeVideoFetcher()
  File "app/fetch_videos.py", line 22, in __init__
    self.config = yaml.safe_load(f)
  File "/usr/local/lib/python3.9/site-packages/yaml/_init_.py", line 125, in safe_load
    return load(stream, SafeLoader)
  File "/usr/local/lib/python3.9/site-packages/yaml/_init_.py", line 81, in load
    return loader.get_single_data()
  File "/usr/local/lib/python3.9/site-packages/yaml/constructor.py", line 49, in get_single_data
    node = self.get_single_node()
  File "/usr/local/lib/python3.9/site-packages/yaml/composer.py", line 35, in get_single_node
    if not self.check_event(StreamEndEvent):
  File "/usr/local/lib/python3.9/site-packages/yaml/parser.py", line 98, in check_event
    self.current_event = self.state()
  File "/usr/local/lib/python3.9/site-packages/yaml/parser.py", line 142, in parse_implicit_document_start
    if not self.check_token(DirectiveToken, DocumentStartToken,
  File "/usr/local/lib/python3.9/site-packages/yaml/scanner.py", line 116, in check_token
    self.fetch_more_tokens()
  File "/usr/local/lib/python3.9/site-packages/yaml/scanner.py", line 258, in fetch_more_tokens
    raise ScannerError("while scanning for the next token", None,
yaml.scanner.ScannerError: while scanning for the next token
found character 'g' that cannot start any token
in "config.yaml", line 1, column 1

=== Complete ===
Check results: .\Check-Results.ps1
PS C:\Users\DELL\hadoop-cluster-im> |
```

```
PS C:\Users\DELL\hadoop-cluster-im> .\0-Master-Setup.ps1

YOUTUBE VIDEO ANALYSIS - HADOOP CLUSTER
Complete Automated Setup

STEP 1: Fixing environment configuration...
✓ Created new .env file

STEP 2: Cleaning existing cluster...
✓ Cleaned

STEP 3: Creating project structure...
✓ Structure ready

STEP 4: Starting Hadoop cluster...
Creating network "hadoop-cluster-im_hadoop" with driver "bridge"
Creating volume "hadoop-cluster-im_hadoop_namenode" with default driver
Creating volume "hadoop-cluster-im_hadoop_datanode1" with default driver
Creating volume "hadoop-cluster-im_hadoop_datanode2" with default driver

Initializing
Time: 105 seconds
[ooooooooooooooooooooooooooooo]

Creating resourcemanager ... done
Creating youtube-fetcher ... done
Creating datanode2 ... done
Creating datanode3 ... done
Creating nodemanager1 ... done
Creating nodemanager3 ... done
Creating nodemanager2 ... done
Creating submit ... done
```