ADROID Conference Series: Engineering and Technology
Volume 2, Issue 3, pp.1-8, 2026

Analyzing YouTube Content With AWS: Data Pipeline, AI And Visualization

Anuradha Chinta[1], Manoj Pothuraju[2], Niveditha Manam[3], Chandana Chigurupati[4]

Siddhartha Academy of Higher Education, Andhra Pradesh, India

anuradha.chinta4@gmail.com, 228w1a5454@vrsec.ac.in, 228w1a5436@vrsec.ac.in, 228w1a5413@vrsec.ac.in

Abstract. YouTube generates a large volume of semi-structured data that provides valuable insights into audience engagement, publishing behavior, and emerging content trends. Efficient analysis of this data has become increasingly important due to the rapid growth of digital media consumption. This study presents a cloud-based data engineering pipeline developed using Amazon Web Services (AWS) to process and analyze large-scale YouTube datasets. The pipeline begins with data ingestion either directly from Amazon S3 or via the YouTube Data API. Automated preprocessing is performed using AWS Lambda functions, including data cleaning, transformation, and feature extraction. To improve storage efficiency and query performance, the processed data is converted to the Apache Parquet format and stored in partitioned structures on Amazon S3. Amazon Athena enables serverless querying of optimized datasets, allowing scalable analysis without the need for infrastructure management. AWS Glue is utilized for ETL automation, metadata cataloging, and schema management, while AWS Step Functions coordinate workflow execution and enhance reliability through error-handling mechanisms. The proposed architecture is designed to be scalable and extensible, enabling future integration with visualization platforms and machine learning workflows. Overall, the system provides a practical and cost-effective approach for transforming raw YouTube data into meaningful insights that support data-driven decision-making for content creators, analysts, and digital media organizations.

Keywords: YouTube Data Analysis; AWS Cloud Services; Data Engineering Pipeline; Amazon S3; AWS Lambda; Amazon Athena; ETL Processing; Cloud Analytics

1 INTRODUCTION

In contemporary digital ecosystems, YouTube is one of many platforms that generate substantial amounts of raw data, including viewer interactions, engagement metrics, and detailed video metadata. The information is significant in the study of audience behaviour, the determination of new content trends, and the strategic decisions of content creators and organisations. Nonetheless, this data is extremely large and unstructured, which complicates processing using conventional data management techniques. Traditional systems are usually unable to scale efficiently or respond to queries, leading to greater resource and processing time consumption.

Cloud computing is an ideal solution to these problems, as it offers elastic storage, scaling processing systems, and automated data processing. The data processing of YouTube datasets is designed and implemented using Amazon Web Services (AWS) in this work. The pipeline consists of the data ingestion, transformation, and querying stages. Raw and processed data are stored in Amazon S3, and preprocessing is automated with AWS Lambda. Amazon Athena allows serverless querying, and it is possible to combine it with Amazon Redshift where a structured storage and analytics can be performed. The overall aim of the project is to create a scalable and economical pipeline, which processes raw You Tube data into query-ready format, allowing an efficient analysis without necessarily using the conventional processing mechanisms.

2 PROBLEM STATEMENT

Digital content such as YouTube is generating massive content. quantity of raw and unstructured data as viewing statistics, metadata and user engagement be-haviour. Bringing out valuable conclusions. based on this information is critical in the enhancement of content performance, knowledge. audience interests, and advocating evidence-based decision-making. Nevertheless, the current data processing systems can be restricted in the capability. to process the amount, speed, and diversity of such data. Traditional tools are not scalable, manual and in-volve intensive. computational resources, which translates to slow analysis and less. responsiveness.

Such weaknesses limit prompt trend identification, prevent individualization. content strategies and have a negative impact on engagement with the user. To solve these problems, the project will be cloud-native based on. Amazon Web Services (AWS). Through the combination of services like Amazon S3, The pro-posed

pipeline is AWS Lambda, AWS Glue and Amazon Redshift. facilitates the ingestion, transformation and analysis of massive quantities of data automatically. YouTube data. This architecture enables quicker processing, cuts back. unit overhead, and enables stakeholders to see insights using effective querying and reporting processes, hence, enhancing decision-making. content and making optimization strategies.

3 LITERATURE SURVEY

Sangeetha and Anu [8] discuss the methods of sentiment analysis that are used on. YouTube comments are based on machine learning techniques to categorize viewers. opinions. Although their strategy is successful in capturing emotional trends since it has successfully done so. user feedback, it mostly aims at sentiment classification and does not. meet the requirements of a unified end-to-end pipeline of data. consumption, metamorphosis and analytics. The proposed work, in contrast, is based on the idea that a person has to be born into a society where norms and values are apparent and evident, and that the individual possesses a higher capacity than anyone else in the society to adhere to them, which is the difficult aspect of the term. is not limited to sentiment analysis but includes such analytical. capabilities in a scalable data pipeline based on AWS that supports. automated processing.

Kready et al. [1] have put forward a parallel processing model to enhance the. efficiency of YouTube data collection. Their efforts show a lot of importance. optimized faster data ingestion, but lacks downstream. step(s) in the processing, i.e. analytics, querying, or visualization. Building based on this notion, the suggested system combines ingestion with serverless. querying mechanisms preprocessing, providing a single pipeline between data capture to knowledge creation.

In [7], there is a YouTube analytics dashboard that has to be used to visualize its engagement. metrics and content performance. The dashboard gives convenient information but still, the dashboard fails to offer important information. insights, the backend is not that much automated, and there are limitations when. handling large datasets. These challenges are dealt with in the present project by implementing AWS Glue and Amazon Athena to automate ETL. and champion scalable analytics.

Sarker et al. [5] examine video streaming behaviour on the cloud environment. through latency and throughput change analysis. While their study is useful in providing insights on performance of streaming infrastructure, it. is not concerned with content-level analytics. In comparison, this work focuses on content-driven analysis such as engagement patterns and behaviour of the viewers whilst using cloud-native services to get. scalability and cost effectiveness.

A. Research Gap

The majority of solutions available that address YouTube data analytics are aimed at. retrieving the measures of engagement or ad-hoc analysis but also giving. little care regarding the management issues of the cost and resources. that relates to big data cloud processing. Optimizing cloud-based data pipelines to reduce computational cost without tradeoffs. performance is an under-developed field, especially when it comes to high-volume. data created by such platforms as YouTube.

The preprocessing and transformation of data phases are the most resource consuming. intensive elements of cloud pipelines. To enhance their productivity is Thus, very important in developing high-performance systems that are cost-effective. Moreover, ETL processes on clouds are scaled out. was not given much thought particularly in situations that involved continuously. modifying streams of data received using YouTube APIs. Existing approaches are primarily in service to simple analytics or isolated querying tasks and are not yet offering a complete, cloud-native pipeline that can support. massive ingestion and transformation of data.

Modern cloud computing systems like Amazon Athena and redshift spectrum. have features that can serve to greatly improve real-time analytics but are not. not fully used in the modern studies. Besides, predictive analytics with. There is historical You Tube data which is not adequately researched. Most studies emphasise real-time engagement analysis, and few efforts are targeted at it. predicting patterns or performance of content in the future based on ma-chine learning. techniques. Lastly, no detailed analysis of the cost has been done. benefit trade-offs and ROI of YouTube on the clouds. data pipelines. Knowledge of economic viability of such architectures is necessary in case one has to work with large-scale multimedia data. These gaps point out the necessity to have systematic research to measure performance, scalability, and financial feasibility of cloud-based YouTube analytics.

Subsection Cloud Architecture and Service Selection

The suggested cloud system is to be built to convert semi structured Normalize YouTube data into a queriable and organized format by merging various AWS services, which facilitate scalability, automation and efficiency processing. The pipeline starts with raw JSON data obtained with the help of the YouTube Data API which is stored in an Amazon S3 raw data bucket. Amazon S3 is a unified data pool, a durable and cost-efficient storage

whose service semi-structured data in large amounts. After the raw data has been stored, S3 event notifications invoke an AWS Lambda function to start automated processing. The Lambda function uses AWS A Python library called Data Wrangler and based on Pandas and AWS services, to convert JSON files to Apache parquet. Parquet is chosen because of its columnar storage structure that lowers the cost of storage and enhances query performance. The processed information is then saved to a different S3 cleaned bucket, arranged in logical parts in order to facilitate effective data recovery and downstream processing.

In order to facilitate metadata management and structured querying, an AWS Glue Cleaned Data Data Catalog crawler goes through the data, deduces schema data, and records the metadata within a central database. This metadata layer allows native support with Amazon Athena, a serverless SQL query engine that is an S3-based Parquet storage querying library. Athena and AWS Scholarly collaborate to assist schema development and query optimization execution. AWS Step is used in workflow orchestration and error handling. Interactions between Lambda, Glue can be coordinated using functions and S3.

Even though the data used today does not involve persistent data, the architecture, which is extensible, can be combined with warehousing Amazon Red-shift or Amazon RDS in case of advanced analytics and dashboard-based reporting in subsequent stages. Operationally, the monitors and logs are supported by AWS CloudWatch with architecture enabling performance, error rates, and execution times to be monitored with the system in real time. This allows processing failure to be detected early on and is helpful unrelenting pipeline optimization. In general, the suggested architecture is a proven, scalable, and secure platform of huge scale YouTube data analytics are flexible so that they can accommodate the future advancement in the form of real-time streaming analytics, machine learning-interactive visualization systems, based insight generation and interactive visualization systems.

4 METHODOLOGY

The suite of YouTube data processing pipelines suggested is constructed with the help of a set of serverless AWS services, which were chosen to serve a particular step in the data life cycle, such as ingestion, transformation, storage, cataloging, and querying. Amazon S3 is the main storage layer because of its scales, reliability, and can support high volumes of semi-structured data. Two distinct S3 buckets are kept to enable clarity separation of concerns: one bucket is filled with raw JSON data obtained out of the YouTube API, while the other stores optimized and changed Parquet files.

Event notifications when new raw data is uploaded on the S3 bucket trigger an AWS Lambda automatically, making it run a wholly automated transformation process. The Python implementation of the Lambda function grounds its use on libraries like AWS Data Wrangler to communicate with AWS and Pandas to manipulate data. Environment variables are used to set parameters such as the S3 location of output, AWS Glue database name, target table, and write mode, which enables flexibility across deployment environments.

When executed, the Lambda function parses the raw data in the form of a JSON piece with wr.s3.read json(). YouTube API responses are usually nested as they are hierarchical. The hierarchical structure is flattened with the items array to create a tabular form using computed JSON normalize operations. Other pre-processing procedures, such as the elimination of redundant fields and treatment of missing values, are utilized to provide consistency in data. The cleaned S3 bucket is written with the processed data in Apache Parquet format using wr.s3.to parquet(). The registering of this operation also creates a dataset on the AWS Glue database catalog, making it possible to partition and efficient query access.

After the metadata has been registered in the AWS Glue Data Catalog, Amazon Athena is a query engine to execute SQL-based analytical queries over the Parquet datasets stored in Amazon S3. Athena makes use of the schema definitions simplifying complex queries and allowing analysts to run them with-out operating specific database infrastructure. This serverless querying model enables the system to scale itself based on demand of the query and yet cost-effective.

To have a credible implementation and synchronized processing among various operation stages, AWS Step Functions is used to coordinate relations be-tween AWS Lambda, AWS Glue, and Amazon S3 components. Step Functions coordinate execution, task monitoring and retry when there are short-lived failures, enhancing the robustness of the pipeline. The pipeline is enabled to work 24/7 and reduces manual intervention.

As a performance optimization, data splitting policies are set in S3 storage layer according to the characteristics of upload date or channel identifier. The process of partitioning data is very important to minimize the data scanned on query execution, which causes faster response time and lower query costs. Besides, the storage of information in Parquet form reduces I/O overhead and enhances analytical performance of large scale workloads.

```
import awswrangler as wr
import pandas as pd
import urllib.parse
import os
os_input_s3_cleansed_layer = os.environ['s3_cleansed_layer']
os_input_glue_catalog_db_name = os.environ['glue_catalog_db_name']
os_input_glue_catalog_table_name = os.environ['glue_catalog_table_name']
os_input_write_data_operation = os.environ['write_data_operation']
def lambda_handler(event, context):
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = urllib.parse.unquote_plus(
        event['Records'][0]['s3']['object']['key'], encoding='utf-8'
    )
    try:
        df_raw = wr.s3.read_json(f's3://{bucket}/{key}')
        df_step_1 = pd.json_normalize(df_raw['items'])
        wr_response = wr.s3.to_parquet(
            df=df_step_1,
            path=os_input_s3_cleansed_layer,
            dataset=True,
            database=os_input_glue_catalog_db_name,
            table=os_input_glue_catalog_table_name,
            mode=os_input_write_data_operation
        )
        return wr_response
    except exception as e:
        print(e)
        print(
            f"Error getting object {key} from bucket {bucket}. "
            "Ensure the object exists and the bucket is in the correct region."
        )
        raise e
```

Fig. 1: Code of Aws lambda function for our dataset

It is also a centralized metadata that can be used with AWS Glue Data Catalog. store that can be looked into by downstream analytic services. and without the need to further configure it. This centralized catalog facilitates the integration of various components of the analytics. pipeline. It is a serverless SQL query engine, which is called Amazon Athena. The parquet files that are stored in the purged S3 layer. Athena relies on the schema data that can be found in the Glue Data Catalog, and can then be accessed effectively through SQL. without having to provision or run compute-based analysis. infrastructure. With this integration, analysts and data engineers can. good at performing exploratory and analytical queries and helps more. interconnection with reporting systems, business intelligence systems, and prolonged analytics processes.

4.1 Architecture

The provided architecture is a data that is fully automated and scaled. processing pipeline Automated using AWS services to process raw. JSONify YouTube data into analytics readable form. The workflow begins via the consumption of unprocessed JSON files into an Amazon S3 raw bucket. Upon S3 event notifications trigger an AWS Lambda function which in turn triggers data arrival. drives the process of transformation. Semi- converted by the Lambda function. converts structured JSON to Apache Parquet using AWS Data. Wrangler. Parquet is chosen because it is columnar in nature and is therefore stored. optimizes query response and storage and scan costs.

The processed data is saved in another S3 cleansed bucket to guarantee. dis-tinct difference between unprocessed and processed data. Metadata related to the purged Parquet files are entered in the AWS Glue Data Catalog, allowing systematic retrieval of the information. This metadata combination permits. Amazon Athena to query the datasets in S3 directly, and without a server. manner. In general, the architecture offers a cost-effective, scalable and analytical querying that is supported by performance-oriented solution, reporting and subsequent downstream processing.

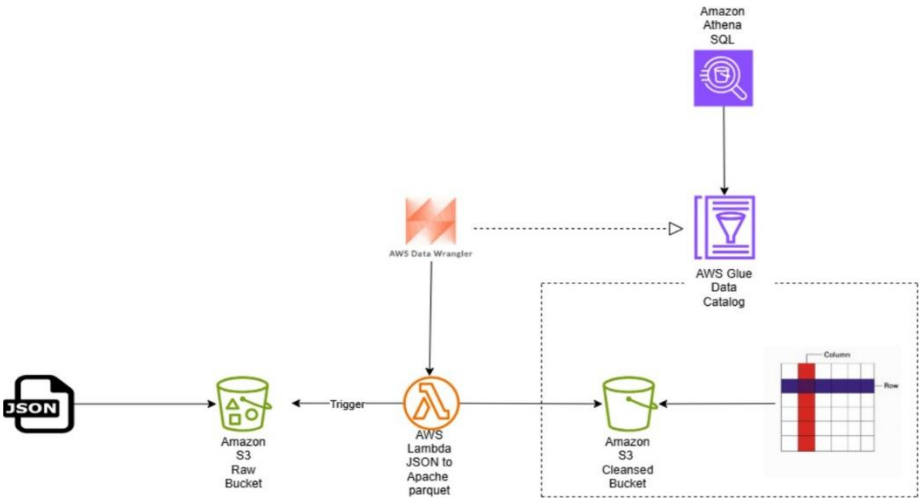

Fig. 2: Code of Aws lambda function for our dataset

5 RESULTS

The execution of the suggested YouTube data analysis pipeline has proven that AWS services are efficient to manage massive data ingesting, transformation, querying, and visualizing operations. The YouTube Data API was used to ex-tract data, which included important metadata like video identifiers, video titles, video descriptions, date of publication, number of views, number of likes and number of comments. This extraction was automated with a Python-based AWS Lambda function, which allowed it to have a serverless, infinitely scalable and low-maintenance ingestion layer.

Fig. 3: Raw YouTube data in JSON format

Figure 3 is a representation of the raw YouTube category reference and metadata in the form of a JSON. The data set has nested and semi-structured variables like items, snippet and engagement-related variables that cannot be di-rectly queried analytically. In order to overcome this shortcoming, the retrieved information was saved in Amazon S3 and later cleaned and transformed through AWS Glue. At this phase, the handling of missing values, schema inconsistencies and field standardization to aid downstream analytics were addressed.

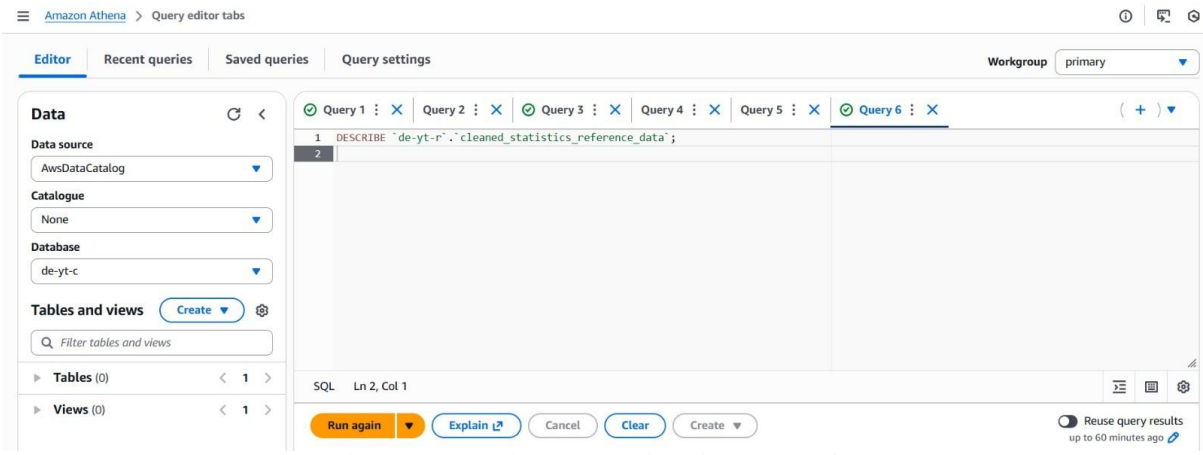

Fig. 4: Query execution on Parquet data using Amazon Athena

Figure 4 demonstrates how SQL queries are run on transformed Parquet data using Amazon Athena. The effective implementation of the queries proves the absence of friction between the AWS Glue Data Catalog and the schema of the inferred structure, which is represented in Figure 5,This structured repre-sentation greatly simplifies the handling of analytics and increases the speed of the queries on large amounts of data, making it serverless.

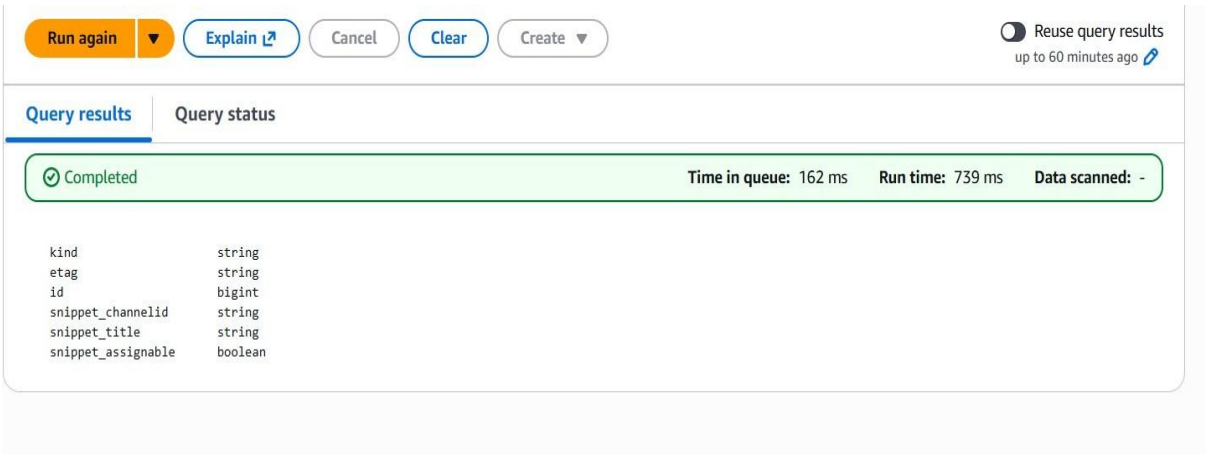

Fig. 5: Structured schema after querying Parquet data

The inferred structured schema obtained after querying is shown in Figure 5, This structured representation significantly simplifies analytical processing and enhances query performance.

In addition to querying, visualization plays a crucial role in extracting action-able insights from the processed data. Amazon QuickSight was used to create interactive dashboards on top of the Athena datasets. Figure 6 shows a KPI visual displaying the total count of likes across all analyzed videos, providing a high-level view of overall audience engagement.

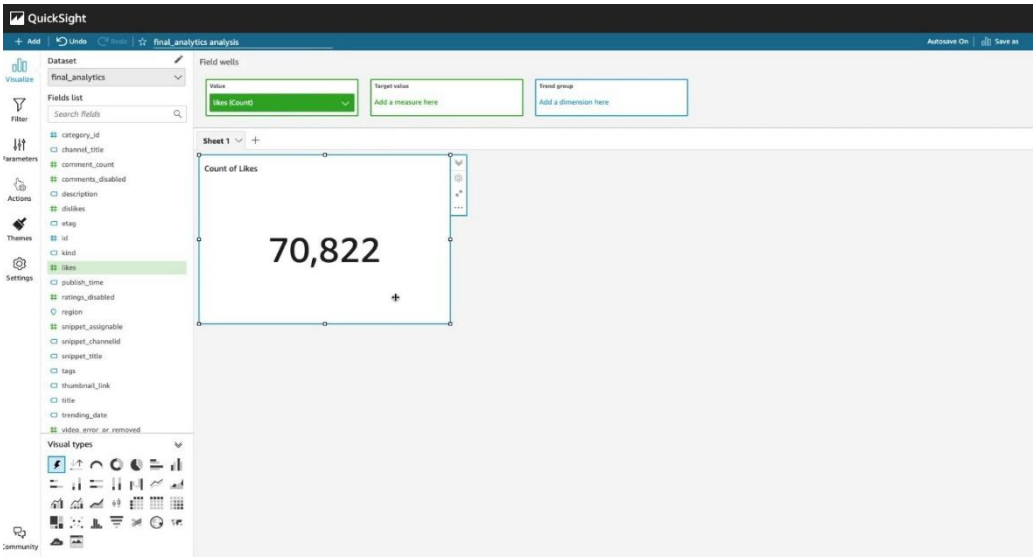

Fig. 6: Total likes across YouTube videos using Amazon QuickSight

In Figure 6, a KPI visual representation of the total number of likes of all the videos analyzed will be displayed, which will give an overall picture of the audience. This visualization assists in determining the categories of content like Music, Entertainment and People & Blogs which have more audience interaction than others

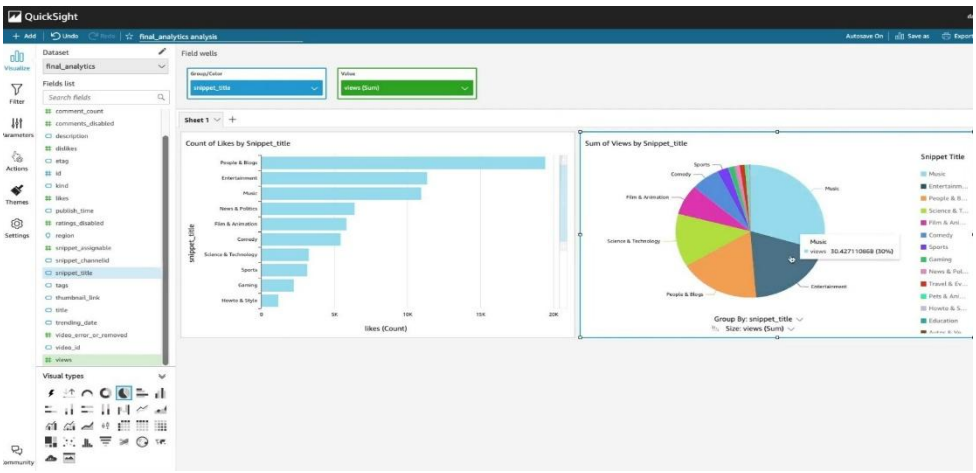

Fig. 7: Likes and views comparison by content category in Amazon QuickSight

Figure 7 shows a synthesis analysis perspective comprising of a bar chart (horizontally) and a pie chart created in the Amazon QuickSight. The bar chart will compare the number of likes per category of content whereas the pie chart will be in form of a percentage of total views contributed by the category. This two-way visualization allows comparing the popularity and reach, and it turns out that those categories that have the highest engagement do not necessarily have the highest number of views.

In general, the findings assure that the suggested AWS-based pipeline is resilient, can be expanded, and completely automated. The combination of Amazon Athena and Amazon QuickSight can provide end-to-end analytics, that is, the ingestion of raw data to interactive visualization, without any infrastruc-ture to manage. The system can be used in large-scale applications of YouTube content intelligence, as the modular architecture will enable future additions of real-time dashboards, sentiment analysis powered by AI, and predictive analyt-ics.

6 CONCLUSIONS

This project offers a start to finish analytics pipeline based on the cloud. AWS services analysis of YouTube content. Amazon S3, AWS Lambda, to construct an automated and scaled ingestion AWS Glue and AWS Glue are used. and pre-processing system of YouTube metadata processing and handling. engagement statistics. Amazon Athena is a tool that facilitates querying the. processed data sets, allowing the analysis of the audience

interaction patterns. and performance trends - content.
The suggested pipeline illustrates the way cloud-native and serverless ser-vices. can be used successfully to process large amounts of semi-structured. data having low operational overhead. The architecture will be structured to be. expandable, so that it can be integrated with visualization tools in the future like

Amazon QuickSight or web-based custom dashboards. Real-time data The processing can be facilitated by such services as AWS Kinesis or. Apache Kafka to facilitate unending tracking of the behavior of the audience.
More sophisticated machine-based analytics can be developed in the future. knowledge on how to perform sentiment analysis, content classification, and trend prediction. Further Natural language processing models can be further. improved to enhance the extraction of comments and metadata as insight. Other extensions can include cross platform analytics, monetization. creator personalized dashboards and analytics, anomaly detection, and analysis. organizations. Future efforts will aim at incorporation of real time streams of data and ma-chine. further interactive dashboards to further learning-based sentiment analysis. develop analytical abilities. Besides, cost performance. benchmarking may be performed in order to measure query efficiency, storage. optimization, and large scale deployments in terms of ROI. It may also be further designed to cater to cross-platform. analytics with integration of other social media platforms, which makes it possible. content performance comparative analysis. These enhancements would enhance the system to be able to comply with advanced, scalable, and data-supportive systems. spurred decision-making of changing digital media settings.

Acknowledgements
The authors acknowledge Siddhartha Academy of Higher Education for aca-demic support.

Abbreviations
AWS – Amazon Web Services S3 – Simple Storage Service ETL – Extract Transform Load

Conflict of Interest
The authors declare no conflicts of interest regarding the current research.

Author Contribution
First author: research design.
Second author: system implementation.
Third author: data analysis and documentation. Fourth author: validation and review.

References
[1] Kready, J., Shimray, S. A., Hussain, M. N., Agarwal, N. YouTube data collec-tion using parallel processing. In IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). 2020. pp. 1119–1122.
[2] Krishnappa, D. K., Bhat, D., Zink, M. DASHing YouTube: An analysis of using DASH in YouTube video service. In 38th Annual IEEE Conference on Local Computer Networks. 2013. pp. 407–415.
[3] Khosla, C. YouTube data analysis using Hadoop. Doctoral Dissertation, California State University, Sacramento. 2016.
[4] Sowmiya, K., Supriya, S., Subhashini, R. Scraping and analysing YouTube trending videos for BI. In Smart Intelligent Computing and Communication Technology. 2021. pp. 542–547.
[5] Sarker, S. A., Rahman, M., Muslim, N., Islam, S. Performance analysis of video streaming at the edge and core cloud. In International Conference on Networking, Systems and Security (NSysS). 2018. pp. 1–7.
[6] Dhulavvagol, P. M., Totad, S. G., Sourabh, S. Performance analysis of job scheduling algorithms on Hadoop multi-cluster environment. In Emerging Research in Electronics, Computer Science and Technology: ICEREC T 2018. 2019. pp. 457–470.
[7] Singh, R. P., Kumar, A., Niharika, N., Jain, G., Rastogi, R., Jain, M. Development of a comprehensive YouTube analytics dashboard for enhanced performance insights. In International Conference on Artificial Intelligence and Emerging Technology (Global AI Summit). 2024. pp. 1089–1094.
[8] Sangeetha, J., Anu, V. M. Current trends on sentimental analysis on YouTube videos. In International Conference on Trends in Electronics and Informatics (ICOEI). 2022. pp. 327–335.
[9] Ranjani, U., Gokulnath, R., Rajasathish, R., Suriyaprakash, A. YouTube Data Analysis Using Python.
[10] Eagar, G. Data Engineering with AWS: Learn how to design and build cloud-based data transformation pipelines using AWS. Packt Publishing Ltd. 2021.