

# **End-to-End Sentiment Analysis on Twitter Data Using Google Cloud and AWS**

A PROJECT REPORT

*Submitted by*

SHANE SUNNY KOTTUPPALLIL [RA2211028010211]

MAMATHA PRASANNA GOWTHAMI PAYASAM

[RA2211028010213]

ALFRED FERDINAND [RA2211028010236]

*Under the Guidance of*

**Dr. Gouthaman. P**

Assistant Professor, Department of Networking and Communications

*in partial fulfilment of the requirements for the degree of*

**21CSC314P - Big Data Essentials**

**in**

**COMPUTER SCIENCE AND ENGINEERING**

**with specialization in CLOUD COMPUTING**



**DEPARTMENT OF NETWORKING AND COMMUNICATIONS COLLEGE  
OF ENGINEERING AND TECHNOLOGY**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**KATTANKULATHUR- 603 203**

**NOVEMBER 2024**



Department of Networking and Communications

**SRM Institute of Science & Technology**

**Own Work Declaration Form**

This sheet must be filled in (each box ticked to show that the condition has been met). It must be signed and dated along with your student registration number and included with all assignments you submit – work will not be marked unless this is done.

To be completed by the student for all assessments

**Degree/ Course** : Btech Computer Science and Engineering with  
Specialization in Cloud Computing

**Student Names** : Shane Sunny Kottuppallil, Mamatha Prasanna  
Gowthami Payasam, Alfred Ferdinand

**Registration Numbers** : RA2211028010211, RA2211028010213,  
RA2211028010236

**Title of Work** : End-to-End Sentiment Analysis on Twitter Data  
Using Google Cloud and AWS

We hereby certify that this assessment compiles with the University's Rules and Regulations relating to Academic misconduct and plagiarism, as listed in the University Website, Regulations, and the Education Committee guidelines.

We confirm that all the work contained in this assessment is my / our own except where indicated, and that I / We have met the following conditions:

- Clearly referenced / listed all sources as appropriate
- Referenced and put in inverted commas all quoted text (from books, web, etc)
- Given the sources of all pictures, data etc. that are not my own
- Not made any use of the report(s) or essay(s) of any other student(s) either past or present
- Acknowledged in appropriate places any help that I have received from others (e.g. fellow students, technicians, statisticians, external sources)

- Compiled with any other plagiarism criteria specified in the Course handbook / University website

I understand that any false claim for this work will be penalised in accordance with the University policies and regulations.

DECLARATION:

We are aware of and understand the University's policy on Academic misconduct and plagiarism and I certify that this assessment is my / our own work, except where indicated by referring, and that I have followed the good academic practices noted above.

RA2211028010211

RA2211028010213

RA2211028010236

*Shankar*  
*P.M.P. Gautham*  
*Self*

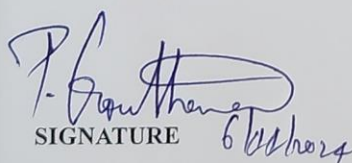


**SRM INSTITUTE OF SCIENCE AND  
TECHNOLOGY KATTANKULATHUR,**

**603203**

**BONAFIDE CERTIFICATE**

Certified that 21CSC314P – Big Data Essentials mini-project report titled “End-to-End Sentiment Analysis on Twitter Data Using Google Cloud and AWS ” is the bonafide work of SHANE SUNNY KOTTUPPALLIL [RA2211028010211] , MAMATHA PRASANNA GOWTHAMI PAYASAM [RA2211028010213], ALFRED FERDINAND [RA2211028010236] who carried out the mini-project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

  
SIGNATURE

PANEL REVIEWER I

**Dr. Gouthaman. P**

**Assistant Professor**

Networking and Communications



  
SIGNATURE

PANEL REVIEWER II

**Dr. D. Saveetha**

**Assistant Professor**

Networking and Communications

## **ACKNOWLEDGEMENT**

We extend our heartfelt gratitude to all individuals and institutions who have contributed to the completion of this project. Foremost, we are deeply grateful to our guide, Dr. Gouthaman P., Assistant Professor in the Department of Networking and Communications, for his invaluable mentorship, encouragement, and guidance throughout this endeavor. His expertise and insights were crucial in enhancing our understanding of sentiment analysis and cloud computing technologies.

We also express appreciation to our peers and colleagues at the SRM Institute of Science and Technology. Their constructive discussions and valuable feedback greatly enriched our research and helped refine our ideas. Their spirit of collaboration and openness fostered a productive learning environment.

Furthermore, we wish to acknowledge the developers and contributors behind the libraries and tools that supported this project, such as the Twitter API, Google Cloud Platform, AWS services, and the VADER sentiment analysis model. The accessibility of these open-source tools and documentation was instrumental in our project's implementation.

Finally, we are grateful to our families and friends for their continuous encouragement and support. Their patience and understanding during the more demanding phases of this project were a tremendous source of motivation. This project reflects the shared efforts and support of all these individuals and organizations, to whom we are profoundly thankful.

## **ABSTRACT**

Sentiment analysis is essential for deciphering public opinions and emotions expressed on social media platforms like Twitter. This project focuses on implementing a comprehensive sentiment analysis pipeline on a tweet dataset using Amazon Web Services (AWS) to categorize sentiments as positive, negative, or neutral. The workflow initiates by uploading raw tweet data, in CSV format, to an Amazon S3 bucket. Upon data upload, an AWS Lambda function is activated, which processes and assesses each tweet's sentiment through AWS Comprehend, a robust natural language processing (NLP) service. The analysis results, including each tweet and its sentiment classification, are then stored in a new CSV file in S3.

This streamlined setup supports scalable and efficient processing of large tweet datasets. Leveraging AWS Lambda for serverless computing ensures a cost-effective, easily deployable solution that removes the need for direct infrastructure management. Meanwhile, AWS Comprehend's machine learning capabilities offer precise sentiment analysis. This project illustrates AWS's potential in handling real-time data processing and analysis, delivering valuable public sentiment insights for applications like market research, customer feedback analysis, and monitoring social media.

## **LIST OF FIGURES**

<b>FIGURE NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
<b>3.1</b>	Architecture Diagram	<b>20</b>
<b>4.2.1</b>	S3 Bucket Creation	<b>33</b>
<b>4.2.2</b>	S3 Object Creation	<b>34</b>
<b>4.2.3</b>	S3 Bucket File Upload	<b>34</b>
<b>4.2.4</b>	Creating ETL Job	<b>35</b>
<b>4.2.5</b>	Defining Data Source Properties	<b>35</b>
<b>4.2.6</b>	Defining Target Properties	<b>36</b>
<b>4.2.7</b>	Visualization of Data Sourcing,Transformation,Target Destination	<b>36</b>
<b>4.2.8</b>	ETL Job Creation using Pyspark Script.	<b>38</b>
<b>4.2.9</b>	IAM Role Creation for Glue	<b>39</b>
<b>4.2.10</b>	Configuring ETL Job Details.	<b>40</b>
<b>4.2.11</b>	Job Monitoring and Setting Pipeline.	<b>41</b>
<b>4.2.12</b>	Twitterdatawarehouse Bucket	<b>41</b>
<b>4.2.13</b>	Adding Crawler	<b>42</b>
<b>4.2.14</b>	Defining Data Source and Classifiers	<b>42</b>
<b>4.2.15</b>	Running Crawler	<b>43</b>
<b>4.2.16</b>	Creation of Tables by running Crawler.	<b>44</b>
<b>4.3.1</b>	Athena Query Editor Tabs	<b>45</b>
<b>4.3.2</b>	Query Results saved in S3 Bucket	<b>45</b>
<b>5.1.1</b>	Query Execution I	<b>47</b>
<b>5.1.2</b>	Query Results I	<b>47</b>
<b>5.1.3</b>	Query Execution II	<b>48</b>

<b>5.1.4</b>	<b>Query Results II</b>	<b>48</b>
<b>5.1.5</b>	<b>Query Execution and Results III.</b>	<b>49</b>
<b>5.1.6</b>	<b>Query Execution and Results IV.</b>	<b>49</b>
<b>5.2.1</b>	<b>Visualization of Count of Text by Sentiment</b>	<b>50</b>
<b>5.2.2</b>	<b>Visualization of Count of Records by Sentiment</b>	<b>50</b>



## **LIST OF TABLES**

<b>TABLE NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
<b>7.1</b>	<b>PLAN OF ACTION</b>	<b>53</b>

## TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	<b>ACKNOWLEDGMENT</b>	<b>5</b>
	<b>ABSTRACT</b>	<b>6</b>
	<b>LIST OF FIGURES</b>	<b>7</b>
	<b>LIST OF TABLES</b>	<b>9</b>
<b>1</b>	<b>INTRODUCTION</b>	<b>12</b>
	1.1 Problem Statement	12
	1.2 Objectives of the Project	12
	1.3 Aspects Dealt with	13
	1.4 Project Domain	14
	1.5 Scope of the Project	14
<b>2</b>	<b>LITERATURE REVIEW</b>	<b>15</b>
<b>3</b>	<b>PROPOSED METHODOLOGY</b>	<b>20</b>
	3.1 Architecture Diagram	20
	3.2 Modules Involved	22
	3.3 Data Collection	23
	3.4 Data Preprocessing	23
	3.5 Data Storage in Google Drive	23
	3.6 Data Transfer to AWS S3	24
	3.7 ETL Process with AWS Glue	24
	3.8 Data Querying with AWS Athena	24
	3.9 Data Visualization with AWS Quicksight	24

<b>4</b>	<b>IMPLEMENTATION</b>	<b>26</b>
	4.1 Implementation of Automated Data Analysis in GCP	26
	4.2 Implementation Steps for Data Pipeline	32
	4.3 Implementation Steps for Query Editor (Amazon Athena)	44
	4.4 Implementation Steps for Virtualization (AWS Quicksight)	46
<b>5</b>	<b>RESULTS</b>	<b>47</b>
	5.1 Querying on AWS Athena	47
	5.2 AWS Quicksight for Data Visualisation	50
<b>6</b>	<b>CONCLUSION</b>	<b>52</b>
<b>7</b>	<b>PLAN OF ACTION</b>	<b>53</b>
<b>8</b>	<b>REFERENCES</b>	<b>54</b>
	<b>APPENDIX</b>	
	<b>PLAGIARISM REPORT</b>	<b>56</b>

# **CHAPTER-1**

## **INTRODUCTION**

### **1. 1 Problem Statement:**

Sentiment analysis is crucial for interpreting public opinions and emotions on social media sites like Twitter. This project focuses on executing an end-to-end sentiment analysis on a tweet dataset by employing both Google Cloud Platform (GCP) and Amazon Web Services (AWS) to categorize sentiments as positive, negative, or neutral.

The project is structured into two key components. The first involves configuring Google Cloud Platform (GCP) and Google Colab for automated tweet analysis. The GCP setup requires enabling Google Sheets and Drive APIs and creating a service account to access data. A Google Colab notebook reads tweets from a Google Sheet, analyzes sentiment using the VADER (Valence Aware Dictionary and sEntiment Reasoner) model, and updates the sentiment results in real time within the same Google Sheet. This setup automates collecting, analyzing, and updating sentiment data continuously.

The project's second component uses AWS for large-scale data processing and visualization. Tweet data is stored in Amazon S3, with AWS Glue handling ETL (Extract, Transform, Load) processes to build the data pipeline. AWS Athena is utilized for querying processed data, and AWS QuickSight visualizes the sentiment analysis results in an accessible format. Combining GCP for real-time data automation with AWS for scalable processing provides a comprehensive and efficient solution for sentiment analysis, beneficial for applications like market research, customer feedback, and social media monitoring.

### **1.2 Objectives of the Project:**

The objective of this project is to create a scalable, automated big data pipeline for sentiment analysis of Twitter data, utilizing the combined power of Google Cloud Platform (GCP) and Amazon Web Services (AWS). This project is designed to handle large volumes of raw Twitter data, transforming it into insightful analytics that aid in decision-making.

The specific goals of this project are as follows:

- Importing raw Twitter sentiment data into Google Sheets and using Google Colab to automate data processing through integrated APIs.
- Saving the processed dataset to Google Drive and transferring it to an AWS S3 staging area.
- Establishing an AWS Glue ETL pipeline to extract, transform, and load data into an AWS S3 data warehouse.
- Using AWS Athena to define schemas and query processed data, enabling flexible, scalable analysis.
- Generating visualizations of sentiment analysis insights with AWS QuickSight to create clear, data-driven reports.

This project demonstrates a complete, cloud-based big data solution that leverages both GCP and AWS capabilities to manage large-scale, real-time sentiment data. It provides a streamlined, automated workflow that yields insights to inform strategies in business, product development, and marketing.

### 1.3 Aspects Dealt With:

- **Data Acquisition and Preprocessing:** Gather, clean, and prepare social media data to make it suitable for analysis.
- **Sentiment Analysis Techniques:** Categorize text as positive, negative, or neutral; analyze specific aspects and emotions; address challenges like sarcasm and irony.
- **Model Building and Evaluation:** Construct and evaluate sentiment analysis models using relevant performance metrics.
- **Big Data Considerations:** Efficiently manage, process, and store large-scale social media datasets.
- **Visualization and Interpretation:** Create visual representations of findings and derive actionable insights.
- **Ethical Considerations:** Protect data privacy, address inherent biases, and ensure ethical use of information.

## **1.4 Project Domain:**

This project operates within the fields of Big Data Analytics and Cloud Computing, focusing specifically on Social Media Sentiment Analysis. It utilizes the cloud infrastructures of both Google Cloud Platform (GCP) and Amazon Web Services (AWS) to handle the complexities involved in large-scale data processing and sentiment analysis. GCP is used for data ingestion and initial processing, with tools like Google Sheets and Google Colab to streamline data workflows. It then integrates with AWS services such as S3 for data storage, AWS Glue for ETL tasks, AWS Athena for querying, and AWS QuickSight for visualizing sentiment trends.

This domain covers essential aspects of data science, ETL pipelines, and cloud-based analytics, making it ideal for real-time data analysis across diverse sectors. This solution is applicable in fields like business intelligence, marketing, customer feedback analysis, and public opinion monitoring, where extracting valuable insights from large datasets is critical.

## **1.5 Scope of the Project:**

This project is focused on creating and implementing a comprehensive big data pipeline for Twitter sentiment analysis using both GCP and AWS. The scope includes the entire data lifecycle—from ingesting raw Twitter sentiment data into Google Sheets and processing it via Google Colab, to storing it in Google Drive. The processed data is then moved to AWS, where AWS Glue conducts ETL functions to transform the data for analysis. AWS Athena is utilized to query the data, while AWS QuickSight produces interactive visualizations for insights.

This project demonstrates a scalable approach to managing large datasets, automating workflows, and performing real-time sentiment analysis. It incorporates multiple cloud services, allowing adaptability to various social media platforms or datasets. This solution is relevant across industries for sentiment tracking, trend analysis, and making informed decisions based on social media insights.

## **CHAPTER-2**

### **LITERATURE REVIEW**

#### **1) An Implementation of Hybrid Enhanced Sentiment Analysis System using Spark ML Pipeline: A Big Data Analytics Framework**

This paper presents a hybrid CNN-SVM model for conducting sentiment analysis on extensive social media datasets utilizing Apache Spark's MLlib. The proposed model enhances classification accuracy and processing speed, surpassing traditional models such as Naive Bayes, SVM, and Random Forest. Spark's scalable architecture facilitates significant performance improvements, particularly in multi-node setups, rendering it suitable for real-time big data sentiment analysis. The study proposes additional optimization of feature extraction and aims to assess the model across various programming languages and configurations.

#### **2) A Survey on Sentiment Analysis and its applications**

This paper examines sentiment analysis (SA) on social media platforms, including Twitter, Facebook, and Instagram, emphasizing its importance in understanding user opinions across diverse sectors. The study compares three primary SA methodologies: lexicon/rules-based, machine learning (ML), and deep learning (DL), noting that DL models typically demonstrate superior performance, with accuracy ranging from 70% to 95%. Hybrid models that merge these approaches and optimization strategies have indicated enhancements in SA effectiveness. The paper explores SA applications across multiple domains, including politics, healthcare, and finance, while identifying gaps in areas such as domain adaptation, contextual comprehension, and multimodal sentiment analysis.

#### **3) Sentimental Analysis for Response to Natural Disaster on Twitter Data**

This paper introduces a technique for sentiment analysis (SA) on Twitter during natural disasters, addressing imbalanced data through adaptive synthetic sampling. It optimizes feature selection via a binary equilibrium optimizer and classifies sentiments utilizing k-nearest neighbor (k-NN). Evaluated on nine disaster datasets, the method surpassed nine state-of-the-art techniques, achieving a 6.9% increase in accuracy, a 13.3% boost in precision, a 20.2% rise in recall, and an 18% improvement in the F1-score.

#### **4) Sentimental Analysis of Climate Change Tweets**

Public discourse surrounding climate change, amplified through platforms like Twitter, provides essential insights via sentiment analysis. Machine learning (ML) techniques can help decipher these sentiments but encounter challenges such as feature engineering and data imbalance. This study establishes a benchmark for assessing ML algorithms in analyzing climate change discussions on Twitter, categorizing models and enhancing their performance. Experiments utilizing real Twitter data expose key challenges and inform future research trajectories

#### **5) Analysis of Real Time Twitter Sentiments using Deep Learning Models**

This research proposes a deep learning methodology for real-time sentiment analysis of Twitter data, concentrating on accurately identifying tweet polarity (positive, neutral, or negative). The study employs neural networks to discern complex patterns in social media data and evaluates the model through F1 score, accuracy, precision, and recall. The findings underscore the effectiveness of the approach, emphasizing its applicability in public opinion monitoring, brand management, customer feedback evaluation, and reputation oversight.

#### **6) Sentimental and spatial analysis of COVID-19 vaccines tweets**

This study investigates Twitter sentiments regarding the COVID-19 vaccination campaign to inform public health policy. Utilizing a Kaggle dataset, tweets were pre-processed, and polarity was evaluated with TextBlob. Sentiment classification was executed using a BERT model, and a word cloud was generated. Geographic visualizations and spatial analyses, including hotspot analysis, were performed to scrutinize vaccination data. The primary objectives were to categorize tweets by polarity, classify sentiments with BERT, and analyze spatial data.

#### **7) A Twitter Sentiment Analysis for Cloud Providers**

The rapid expansion of social networking platforms generates data for conducting sentiment analysis on products and services. This study evaluates customer perceptions of Amazon and Microsoft Azure through Twitter datasets. It assesses how customers view these cloud services, assisting companies in strategizing marketing efforts. Results indicate that Microsoft Azure garners more positive sentiment (65% of tweets) compared to Amazon (45%). Emotionally, "joy" is more associated with Azure, while Amazon exhibits a higher incidence of "sadness" and 50% negative polarity, in contrast to Azure's 25%.



## **8) Sentiment Analysis of Twitter Data**

Twitter has emerged as a pivotal platform for sentiment analysis, attracting considerable research attention. Twitter Sentiment Analysis (TSA) constitutes a subfield of text mining that employs computational techniques to examine opinions and sentiments within Twitter data. This survey evaluates recent advancements, categorizes novel algorithms, and underscores TSA applications. It provides a thorough overview of TSA methodologies, organizes recent studies, and delineates current research trajectories in the domain. The primary contribution of the survey is its comprehensive classification of articles and insights into the evolving trends in TSA research.

## **9) Sentimental Analysis of Twitter Data with respect to General Elections in India**

The study investigates opinion mining and sentiment analysis utilizing Twitter data collected from January to March 2019 to assess public sentiment during India's general elections. R was employed for data processing, and the analysis compared two political candidates, revealing that Candidate-1 was more popular than Candidate-2, which corresponded with the actual election outcomes in May 2019. Twitter APIs facilitated tweet collection, and sentiment analysis was instrumental in determining the polarity of public opinion.

## **10) Comparative Studies of Detecting Abusive Language on Twitter**

This passage addresses the challenges associated with annotating extensive datasets for identifying online aggression and abusive language due to its context-dependent nature. It highlights that existing datasets are frequently insufficient for effectively training deep learning models. A newer, larger dataset known as Hate and Abusive Speech on Twitter has been released but remains underexplored. The paper conducts a comparative analysis of various models on this dataset, concluding that bidirectional GRU networks utilizing word-level features and Latent Topic Clustering modules exhibited the best performance, achieving an F1 score of 0.805.

## **11) Predicting the Sentiment Polarity of Tweet Replies**

This paper introduces a novel task within Twitter sentiment analysis: predicting the dominant sentiment in replies to a tweet. We present the RETWEET dataset, consisting of tweets and their replies with manually annotated sentiment labels. Our two-stage deep learning approach first aggregates predictions from a standard sentiment classifier on replies to create labeled training

data. Subsequently, we train a neural network using this dataset to forecast reply sentiment. Results on the RETWEET dataset are promising, and both the dataset and implementation are publicly accessible.

## **12) Twitter Sentiment Analysis with CNNs and LSTMs**

This paper outlines the development of a state-of-the-art Twitter sentiment classifier using Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks. The method capitalizes on a substantial amount of unlabeled data to pre-train word embeddings, followed by distant supervision for fine-tuning. The final models are trained on the SemEval-2017 Twitter dataset, and to boost performance, various CNNs and LSTMs are ensembled. This approach secured the top position across all five English subtasks, outperforming 40 participating teams.

## **13) Sentiment Analysis on Twitter Data Using Apache Spark Framework**

This paper discusses the implementation of Apache Spark to process Twitter data for sentiment analysis, emphasizing scalability and real-time performance. It contrasts different machine learning models, particularly Naive Bayes and Random Forest. Spark's distributed computing capabilities enable rapid analysis of extensive datasets, highlighting advancements in both time efficiency and accuracy. The study demonstrates Spark's proficiency in managing the dynamic nature of sentiment analysis, particularly in real-time scenarios.

## **14) Effective Sentiment Analysis of Twitter Data with Apache Spark**

This study addresses the difficulties associated with processing large-scale Twitter data using Apache Spark. It highlights Spark's capabilities for stream processing and batch analysis, especially with the Naive Bayes classifier. The research illustrates how Spark's real-time data processing is vital for sentiment analysis in live contexts, such as elections and product reviews, leveraging Spark Streaming for continuous data input. It examines optimization strategies for enhanced performance.

## **15) An Apache Spark Implementation for Sentiment Analysis on Twitter Data**

This paper presents a scalable sentiment analysis system on Twitter data, utilizing Apache Spark ML pipelines for efficient classification. By integrating Natural Language Processing (NLP) with feature extraction methods, the study illustrates how Spark can manage vast amounts of unstructured data. The proposed model employs both streaming and batch data processing to enhance tweet sentiment analysis, emphasizing the system's adaptability and accuracy in handling large-scale datasets.

## CHAPTER-3

### PROPOSED METHODOLOGY

#### 3.1 Architecture Diagram

The architecture of this project combines services from both Google Cloud Platform (GCP) and Amazon Web Services (AWS) to establish a scalable and efficient big data pipeline for Twitter sentiment analysis. The process starts with raw Twitter sentiment data, which is ingested into Google Sheets and processed via Google Colab. Google Colab automates data transformation and integrates APIs to refresh the dataset, which is subsequently stored in Google Drive.

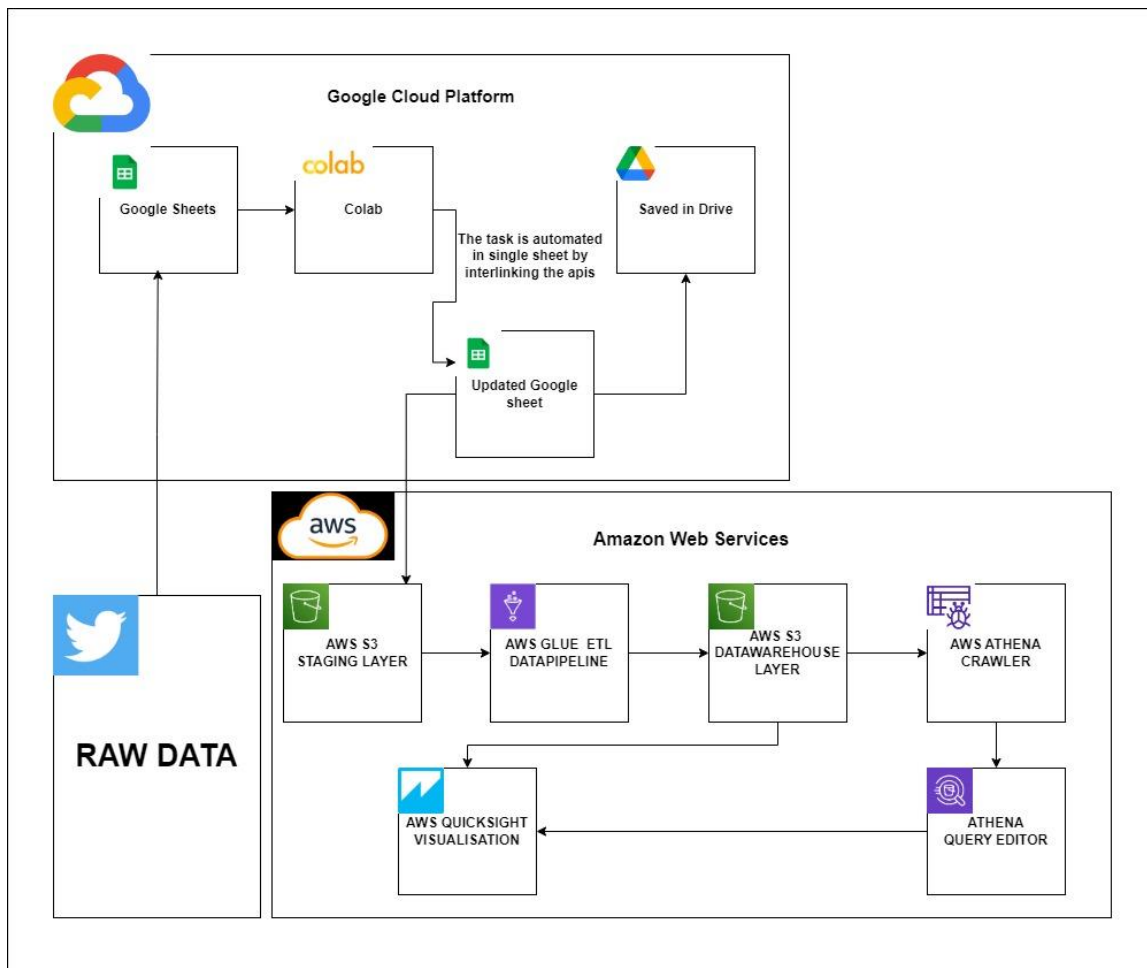


Figure 3.1 Architecture Diagram

The updated dataset is transferred to an AWS S3 staging layer, where AWS Glue manages the ETL (Extract, Transform, Load) process, cleaning and enriching the data prior to storing it in an

AWS S3 data warehouse layer. AWS Athena constructs schemas and allows for querying of the processed data, while AWS QuickSight produces visualizations that offer insights into sentiment trends.

The architecture of this project is engineered to effectively manage the end-to-end flow of Twitter sentiment data using both Google Cloud Platform (GCP) and Amazon Web Services (AWS). Here's a breakdown of the key components and their functions in the architecture:

1. Data Ingestion (GCP):
  - Raw Twitter sentiment data is initially uploaded into Google Sheets, acting as a temporary storage and processing layer.
  - Google Colab automates the processing of this data. API integrations are utilized to clean and enrich the data within Colab, resulting in an updated dataset.
2. Storage on Google Drive (GCP):
  - Following processing, the updated dataset is saved to Google Drive, ensuring easy access and integration with other cloud services.
3. Data Transfer to AWS:
  - The updated dataset is subsequently moved from Google Drive to the AWS S3 Staging Layer, where it is stored for additional processing.
4. ETL Process (AWS):
  - AWS Glue executes the ETL (Extract, Transform, Load) functions. This stage cleans, transforms, and prepares the data for analysis, structuring it for querying.
5. Storage in AWS S3 Data Warehouse Layer:
  - The processed data is stored in the AWS S3 Data Warehouse Layer for efficient querying and analysis.
6. Querying and Schema Creation (AWS Athena):
  - AWS Athena is employed to create tables and schemas for the processed data. Crawlers are utilized to automate the discovery of data formats and structures, facilitating efficient querying.
7. Visualization (AWS QuickSight):
  - Finally, AWS QuickSight generates visualizations and dashboards, delivering insights into the Twitter sentiment data, such as trends and patterns over time.

This architecture ensures seamless integration between GCP and AWS, providing scalability, automation, and real-time data processing and analysis.

## 3.2 Modules Involved:

1. Data Ingestion (Google Sheets and Google Colab):
  - Google Sheets: Utilized to store the initial raw Twitter sentiment data. It serves as a collaborative tool for data input, making it accessible for real-time updates and easy integration with other Google services.
  - Google Colab: The primary processing module in GCP, where APIs are integrated to automate the transformation of raw data. The Colab environment executes code that cleans and processes the dataset, preparing it for subsequent stages.
2. Data Storage (Google Drive and AWS S3 Staging Layer):
  - Google Drive: Functions as an intermediary storage solution after data is processed in Colab. It retains the updated dataset before transitioning it to AWS.
  - AWS S3 Staging Layer: This module manages the transfer of data from Google Drive to AWS. The staging layer serves as the initial landing zone for data on AWS before processing by AWS Glue.
3. ETL (AWS Glue):
  - AWS Glue: A fully managed ETL service for establishing data pipelines, AWS Glue is responsible for extracting the dataset from the staging layer, transforming it (e.g., cleaning, normalization, enrichment), and loading the final processed data into the data warehouse layer in AWS S3.
4. Data Storage (AWS S3 Data Warehouse Layer):
  - AWS S3 Data Warehouse Layer: After the ETL process, the cleaned and structured data is stored in this layer. It acts as a data repository, designed for efficient querying and analytics.
5. Data Querying and Schema Creation (AWS Athena):
  - AWS Athena: This module allows users to execute SQL queries directly on the processed data stored in S3. Athena also automatically creates schemas for the data through crawlers, enabling better data structure and easier querying.
6. Data Visualization (AWS QuickSight):
  - AWS QuickSight: This module is utilized to visualize the insights derived from the processed data. It provides interactive dashboards and graphical representations of Twitter sentiment trends, enabling users to make informed decisions based on the analyzed data.

These modules function in unison to automate the data pipeline, ensuring efficient data processing, querying, and visualization across both GCP and AWS environments.

### **3.3 Data Collection**

The project commences with the acquisition of raw Twitter sentiment data, encompassing user tweets along with their corresponding sentiment labels. This data is sourced via Twitter APIs and initially stored in Google Sheets. By leveraging Google Sheets' collaborative features, multiple users can contribute to the dataset in real time. This phase is vital as it ensures a comprehensive dataset that accurately reflects diverse sentiments. The collected data forms the foundation for further processing and analysis, guaranteeing that insights derived from the analysis are based on rich and varied information.

### **3.4 Data Preprocessing**

Once the data is collected in Google Sheets, it undergoes preprocessing in Google Colab. This process includes cleaning the dataset by removing duplicates, correcting inconsistencies, and filtering out irrelevant information. Data transformation tasks, such as normalizing text and encoding sentiment labels, are executed to prepare the data for analysis. Automation through Python scripts streamlines this process, allowing for efficient management of large datasets. This step is crucial for enhancing data quality, ensuring that subsequent analyses yield reliable and accurate results.

### **3.5 Data Storage in Google Drive**

After preprocessing, the cleaned dataset is saved to Google Drive. This module acts as a temporary storage solution, offering easy access and collaboration features. Google Drive facilitates secure storage and straightforward sharing of the dataset, enhancing team collaboration. This step guarantees that the updated data is readily available for transfer to AWS, functioning as an intermediary between the preprocessing phase and the data transfer process while maintaining data integrity and accessibility throughout the project.

### **3.6 Data Transfer to AWS S3**

The subsequent step involves transferring the processed dataset from Google Drive to the AWS S3 Staging Layer. This is achieved using AWS SDKs or APIs, which enable secure and efficient

data transfer. The staging layer serves as the initial landing zone for data within the AWS environment. This step is critical for ensuring that the dataset is accessible for further processing and analysis using AWS services. Proper transfer protocols are employed to maintain data integrity and security throughout the migration process.

### **3.7 ETL Process with AWS Glue**

Once the data resides in the AWS S3 Staging Layer, AWS Glue is utilized to conduct the ETL (Extract, Transform, Load) process. This module extracts the dataset from the staging layer, applies various cleaning and enrichment functions during the transformation phase, and loads the final processed data into the AWS S3 Data Warehouse Layer. AWS Glue automates much of this workflow, minimizing manual intervention and enhancing efficiency. This step is essential for preparing the data for effective querying and analysis, ensuring it meets the required quality standards.

### **3.8 Data Querying with AWS Athena**

Following the ETL process, AWS Athena is employed to query the processed data stored in the S3 Data Warehouse Layer. This module enables users to execute SQL queries on the dataset, facilitating real-time analysis and insights extraction. AWS Athena also supports schema creation for the data through crawlers, simplifying the querying process. This step empowers users to flexibly explore the data, uncovering insights into sentiment trends and other analytical dimensions without the need for complex data manipulation.

### **3.9 Data Visualization with AWS QuickSight**

The concluding step involves visualizing the insights gained from the sentiment analysis using AWS QuickSight. This module allows for the creation of interactive dashboards and visual reports that present sentiment trends in an easily understandable format. By employing various visualization techniques, stakeholders can quickly comprehend key insights and make informed decisions based on the data analysis. This phase adds significant value to the project, transforming raw data into actionable insights that can inform business strategies and initiatives.



## **CHAPTER-4**

### **Implementation**

#### **4.1 Implementation for Automated Data Analysis in GCP:**

1. **Data Ingestion:** Start by collecting raw Twitter sentiment data via the Twitter API, and store this data in Google Sheets for initial organization and easy accessibility.
2. **Data Preprocessing:** Utilize Google Colab to automate the preprocessing of the dataset. Implement Python scripts to clean the data by removing duplicates, correcting inconsistencies, and normalizing text. This step ensures the data is well-structured and ready for analysis.
3. **API Integration:** Integrate APIs within Google Colab to enhance the data processing workflow, including sentiment analysis APIs to further enrich the dataset.
4. **Automated Updates:** Schedule periodic scripts in Google Colab to automatically fetch new tweets and update the dataset in Google Sheets, ensuring the analysis remains current and relevant.
5. **Export to Google Drive:** After preprocessing, automatically save the cleaned dataset to Google Drive for secure storage and easy access.
6. **Transfer to AWS:** Employ Google Cloud Functions to automate the transfer of the updated dataset from Google Drive to the AWS S3 staging layer.
7. **Data Monitoring:** Implement logging and monitoring within Google Cloud to track data processing workflows, ensuring timely updates and maintaining data integrity throughout the automated analysis process.

## **Steps:**

### **I. Enable Google Sheets API and Create Credentials:**

Go to the Google Cloud Console.

Create a new project (or select an existing one).

Navigate to APIs & Services > Library.

Search for "Google Sheets API" and click Enable.

Now, search for "Google Drive API" and enable it too (as you need Drive permissions).

### **II. Create Service Account Credentials:**

Go to APIs & Services > Credentials.

Click Create Credentials and select Service Account.

Give your service account a name and description, then click Create.

On the next page, click Done.

After creating the service account, click on it and go to Keys.

Click Add Key > Create New Key, and choose JSON.

A JSON file with your credentials will be downloaded to your computer.

### III. Share Your Google Sheet with the Service Account:

Open your Google Sheet.

Find the email in the client\_email field of your downloaded JSON file.

Share the Google Sheet with that email (just like sharing with any other user).

#### JSON File:

```
{
  "type": "service_account",
  "project_id": "favorable-order-436116-k1",
  "private_key_id": "1e13426f205691feb8a5d1066168c2cc21b493f0",
  "private_key": "-----BEGIN PRIVATE KEY-----
\nMIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEAAoIBAQDqMzugcyi4wbK6\nFFKoY
0g0o0Dj0o5u7Nzd4B0JvnBVQoYiRdfugWyBTBDuKxSRsQwFsWFol9zPwlJx\n9SpLWgzXyfNW
7z+AGwTsmDFRjTK4u/swXaZ4YCswQ+T3mQdk1Kx+avOb+h8Ar8h9\nN2jYJmrT8t1HZLmSWBt
NYQe89M1eYzXAzTsdLJqlh9/1TBpSkTK/c6WVui5mQmuZ\nGs+ksyihsm3E9TA8TEQ/Q8gwr0
1qA3tu58EM5JA+7bwU3XBP6/L2aiVv/b21CiRv\n1SVTwKv8oNYtM+3ArZHZjwmlD3rZovxSE
dLMlmtuQhnmhTgmlW5Wh8NRaYA8P9Yer\n4aKl9kDxAgMBAAECggEAAActZBIpSh6+MIR2q3rSg
hiIEt/4ghnadsFY+GLZHBdM0\nN07eUa9EihBpPiVLhq+uFQM73CIzBq7RV+nLqR9B8A6n1NZ
q5J5B+gm8mzp1oHy7\nuUJT8gGMTVgiRVUK/38PhaxhI2kADxOsQqKg84mORQwmTDHY+SPuQQK
/BFrXQKfE2\nnqvMjKyklpNua9xCToKJhMDHCZJRugdPtDNhDf9Hb0ttxmIlPKlQ7psGR6qlmP
42j\nPrmlDoJCSlpNaVsWltvaohFO7pzmtPyJTnGJ8SYeVS8wvNRuL7/m8IndU2WSntLG\nnb0
SZnSrG7xmsg7YPrzOOSm66hLQVkaQ/k2GVnFETIQKBgQD67mHCsB2rtyHJzQTC\nnqq4kVylp/
EkhfiVNtvc8tupQYeXv5AFiQ07eiQJ5FwfGB0bRdsTkc/Ghefdf9wtI\n6BxBKun6rOoxj+8i
DtGXJSZ+FyR1VEYS5dCrmxXd5EKj/kv5/pZSrbpBPhQ+o3R6\nniIPLj/69CFqhk/t4jrfaV5K
d0QKBgQDu7lTMZGBpGb+vuGTDtKeqtsGocBmF3N+1\n6+vC9vuy5rhJo2/WitlZM2RrXIM0IG
GefB1w7Cxfow4MMaejcnK1Zo44Rl0jAimD\nCT1uKD/UyWNO5ZozMieAYJNTwewt/3IT7PkUE
A23A0bcufAfOhZr6fN+IjPkVCiZ\nVLFpeV6ZIQKBgQCWvty2A+1foj4vKF5CwByrsj0r8abR
WyH35QH0VILBVX64NcZr\nDLp1z+NpCXl3CuoNbgdsowa3Fj15SkVaQCrJiAlWUcjDi4+Ca1q
YLXZJa4e4gK80\nLSHF5f4l9jneoSdtgPc3pixtg2jQFIwmDJr4kIHdeAZSaT86NqrWnaNz8Q
KBgCHO\nn1S8KcN3OJ+I8K/3I/QRZsZMcSW/QbdL227gNbFXG8YJx3qwk6C4H6IiMYApqMZc\
n92138Vti3eT5zaeB1Rs3tJ3fzuPanXd2Ajwo88cu659JO4VM7mvqvK3WnY1wnHqX\nnSDL/ho
FDtG4c3Q3+ERgZlTseqcr/4XK4mhW7d5ghAoGAUeUcSa6iQ8he+/BeMz/r\n6xEqjAVf+hwko
zB3JrcOFM13YZNL7TAcBscNAFkKA/QQ6iJiCfSX/gRf8mkLn0NM\nnuAwVM2fQTNKALDq15FH4
```

```

KVXT86v1UvubXmFqWCqZisaWWgkJNdpfmC7SIpp8Vc9I\nekjwnDTb47bsr6s2L4lygVw=\n-
----END PRIVATE KEY-----\n",
  "client_email": "gowthamitwitter@favorable-order-436116-
k1.iam.gserviceaccount.com",
  "client_id": "108204890941024346255",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://oauth2.googleapis.com/token",
  "auth_provider_x509_cert_url":
"https://www.googleapis.com/oauth2/v1/certs",
  "client_x509_cert_url":
"https://www.googleapis.com/robot/v1/metadata/x509/gowthamitwitter%40favo
rable-order-436116-k1.iam.gserviceaccount.com",
  "universe_domain": "googleapis.com"
}

```

Colab Code:

```

from google.colab import files
uploaded = files.upload()

import gspread
from oauth2client.service_account import ServiceAccountCredentials
import pandas as pd

# Set up the Google Sheets API credentials (use your credentials file)
scope = ["https://spreadsheets.google.com/feeds",
"https://www.googleapis.com/auth/drive"]
creds = ServiceAccountCredentials.from_json_keyfile_name("favorable-
order-436116-k1-1e13426f2056.json", scope)
client = gspread.authorize(creds)

# Open your Google Sheet
sheet = client.open("sampletweets").sheet1

#-----
-----

# Extract the data into a pandas DataFrame

```

```

data = pd.DataFrame(sheet.get_all_records())

# Assuming the tweet column is named "Tweet"
tweets = data['Tweet']

#-----

from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

# Initialize VADER sentiment analyzer
analyzer = SentimentIntensityAnalyzer()

# Function to classify sentiment
def analyze_sentiment(tweet):
    vader_score = analyzer.polarity_scores(tweet)
    compound = vader_score['compound']

    # Categorize sentiment based on compound score
    if compound > 0.05:
        return 'Positive'
    elif compound < -0.05:
        return 'Negative'
    else:
        return 'Neutral'

# Apply the function to each tweet
data['Sentiment'] = tweets.apply(analyze_sentiment)

# Update the Google Sheet with the sentiment analysis
# Assuming the 'Sentiment' column exists and it's in the correct location

sentiment_col = data.columns.get_loc('Sentiment') + 1 # Get the
sentiment column index (starting from 1 in Google Sheets)

# Write the sentiment results back to the Google Sheet
for i, sentiment in enumerate(data['Sentiment'], start=2): # Start from
the second row to skip the header

```

```
sheet.update_cell(i, sentiment_col + 1, sentiment) # Assuming the
'Sentiment' column is right next to 'Tweet'
```

1) The `files.upload()` function allows users to upload their credentials JSON file to authenticate with the Google Sheets API.

2) Google Sheets API Setup:

The `gspread` and `oauth2client` libraries are used to set up Google Sheets API credentials. The scope defines the permissions for accessing Google Sheets and Drive. A `ServiceAccountCredentials` object is created from the uploaded JSON file, and `gspread` authorizes the client to access the Google Sheet named "sampletweets."

3) Data Extraction:

The code retrieves all records from the specified Google Sheet and loads them into a pandas `DataFrame`. The tweet content is extracted into the `tweets` variable.

4) Sentiment Analysis Initialization:

The VADER sentiment analyzer is initialized. The `analyze_sentiment` function computes a sentiment score for each tweet and categorizes it as 'Positive,' 'Negative,' or 'Neutral' based on the compound score.

5) Applying Sentiment Analysis:

The `analyze_sentiment` function is applied to each tweet in the `DataFrame`, and the results are stored in a new 'Sentiment' column.

6) Updating Google Sheet:

The sentiment results are written back to the Google Sheet. It identifies the correct column for sentiments and updates each cell corresponding to the tweets, starting from the second row to avoid overwriting the header.

## Model used for Data Analysis:

The model utilized in the code for sentiment analysis is VADER (Valence Aware Dictionary and sEntiment Reasoner).

**Key Points regarding VADER:**

1) Pre-trained Model: VADER functions as a lexicon and rule-based model, indicating that it employs a pre-established dictionary of words linked with sentiment scores.

2) Social Media Orientation: It is specifically crafted to evaluate short and informal texts like social media posts, making it particularly apt for platforms such as Twitter.

3) **Sentiment Scores:** VADER generates a set of four sentiment scores:

Positive: The ratio of positive words present in the text.

Negative: The ratio of negative words.

Neutral: The ratio of neutral words.

Compound: A single combined score that delivers an overall sentiment ranging from -1 (most negative) to +1 (most positive).

In this code, the compound score is utilized to categorize tweets into Positive, Negative, or Neutral based on threshold values.

Why VADER?

It is straightforward and rapid, demands no training data, and is efficient for analyzing informal text like tweets.

## **VADER (Valence Aware Dictionary and sEntiment Reasoner)**

VADER represents a lexicon and rule-based sentiment analysis model, meaning it avoids conventional machine learning for training. Instead, it relies on a pre-compiled dictionary of words and their corresponding sentiment intensities, along with a collection of heuristics or rules to interpret how words operate in context (such as negations, capitalization, punctuation, etc.).

### **Development of the Lexicon:**

- 1) **Human-Curated Lexicon:** VADER's sentiment lexicon (dictionary) was originally developed by human annotators. These annotators were tasked with scoring a substantial set of words or phrases based on their sentiment intensity.
- 2) **Source of Words:** The lexicon was assembled by incorporating words frequently encountered in social media posts, including slang, emoticons, acronyms, and even internet-specific abbreviations like "LOL" or "OMG."
- 3) **Sentiment Scores:** Each word or phrase in the lexicon is assigned a sentiment score between -4 (most negative) and +4 (most positive).

1. **Validation with Multiple Annotators:**

**Crowdsourcing:** Numerous human raters were engaged to evaluate the sentiment intensity of every word or phrase. The ultimate sentiment score for each word is the average of all these evaluations.

**Inter-rater Agreement:** VADER attained robust agreement among human annotators, guaranteeing that the lexicon mirrors a consensus on sentiment for each term.
2. **Integration of Heuristics:**

**Contextual Handling:** VADER integrates various rules and heuristics to address contextual factors that impact sentiment:

**Negation:** Words such as "not" reverse the sentiment (e.g., "not good" turns negative).

**Intensity Modifiers:** Terms like "very" or "extremely" enhance sentiment scores.

**Punctuation:** Exclamation marks (!) and question marks (?) can increase sentiment scores.

**Capitalization:** Words presented in all caps are generally regarded as more intense.

**Emoticons:** It considers emoticons, acronyms, and slang frequently found in social media.

3. **Testing and Refinement:**  
 Performance on Social Media Data: VADER was assessed and validated on datasets derived from social media (especially Twitter), movie reviews, and other areas featuring informal, opinionated language.  
 Fine-Tuning: Based on performance metrics, the model's lexicon and rules were adjusted for precision and relevance in real-world text analysis.
4. **No Traditional Training Data:**  
 In contrast to machine learning models such as LSTM or transformers, VADER does not "learn" from a vast dataset in the conventional supervised learning manner. Instead, its dictionary and rules were crafted and validated by human experts, rendering it more of a rule-based system than a machine-learned model.  
 Summary:  
 VADER's "training" involved human annotators creating a sentiment lexicon and outlining rules for contextual interpretation. The lexicon encompasses words, emoticons, and slang commonly utilized in social media, with each term assigned a sentiment intensity score. VADER is tailored for social media texts and informal language, depending on human judgment instead of machine learning algorithms.

This makes it fast and effective for specific tasks, particularly those involving social media sentiment analysis, without needing large amounts of data for traditional training.

## 4.2 Implementation Steps for Data Pipeline:

### Data Pipeline (AWS Glue)

1. Create IAM User: From your root account, establish an IAM user to manage the project.
2. Set Up AWS Environment: Confirm that your AWS account has the appropriate permissions to utilize S3, Glue, and Athena services.
3. Create S3 Buckets: Set up two S3 buckets—one for staging raw data and the other for storing processed data.
4. Upload Data: Move the cleaned Twitter sentiment data from Google Sheets to the S3 staging bucket.
5. Create Glue Crawler: In AWS Glue, configure a crawler that targets the staging S3 bucket. This crawler will scan the data and generate a metadata catalog.
6. Define ETL Job: Set up an AWS Glue ETL job to extract data from the staging bucket, perform transformations (such as filtering, cleaning, and aggregating), and load it into the S3 data warehouse bucket.



7. Run ETL Job: Initiate the ETL job to process the data and save the transformed dataset in the S3 data warehouse.
8. Visualize the Data: Link Athena to the Quicksight console and visualize the data using the required parameters.

### S3 Bucket Creation:

1. Log into AWS Console: Go to the AWS Management Console and access S3.
2. Create a New Bucket:
  - Click on "Create bucket."
  - Input a unique name and choose a region.

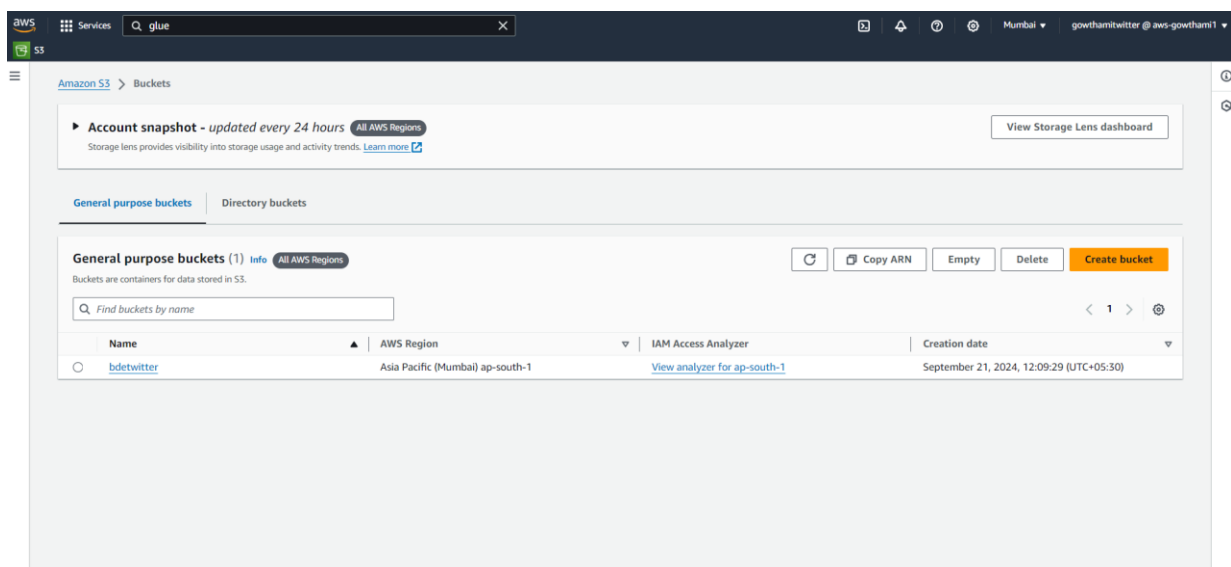


Figure 4.2.1 S3 Bucket Creation

3. Upload Data: Access the bucket and select "Upload" to add files.
- Set Bucket Policies (Optional): Modify permissions through bucket policies or ACLs.

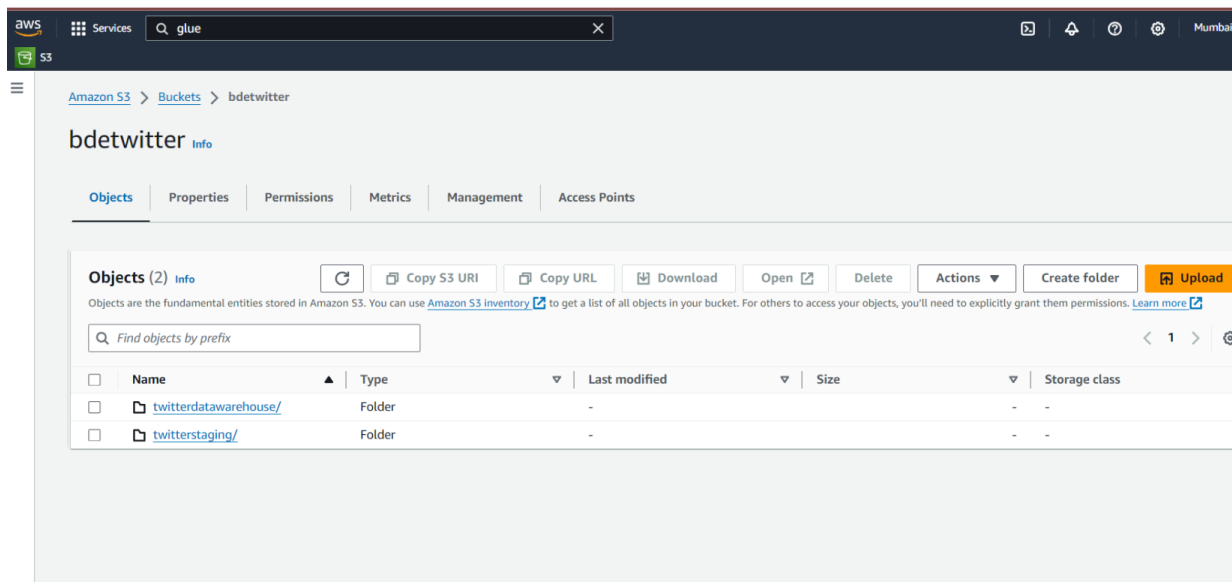


Figure 4.2.2 S3 Object Creation

5. Enable CORS (If Required): Set up CORS settings for handling cross-origin requests.
6. Monitor and Manage: Utilize the dashboard to oversee usage and handle lifecycle policies.

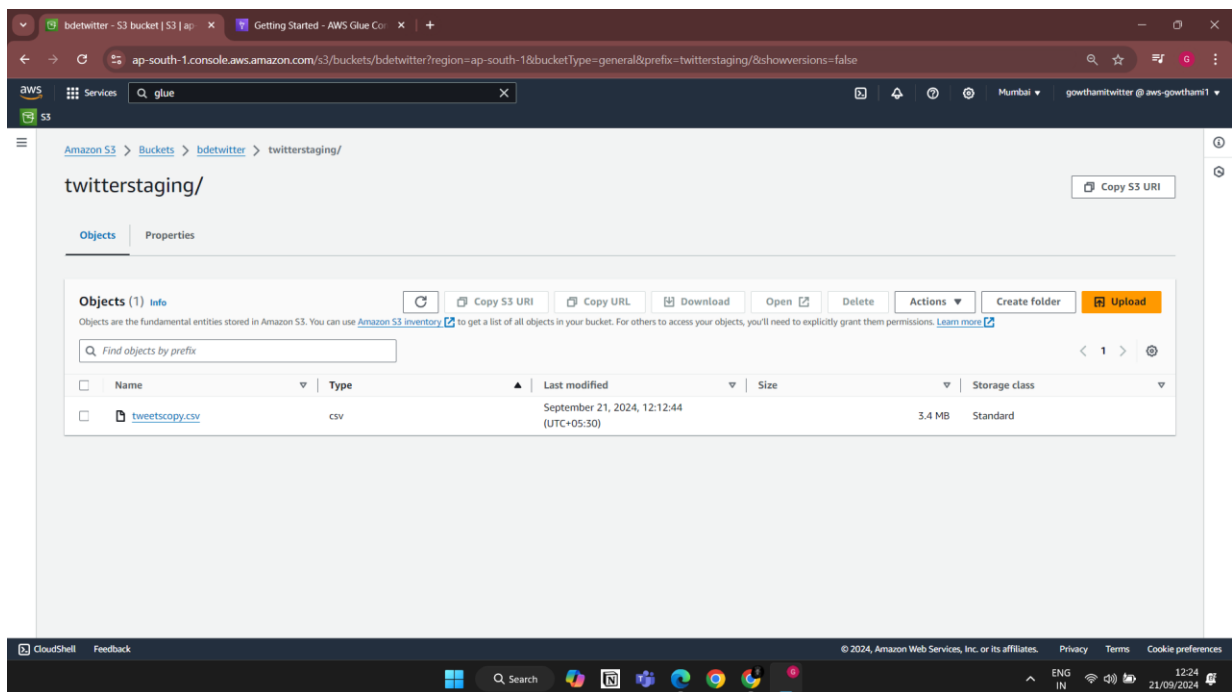


Figure 4.2.3 S3 Bucket File Upload

7. Access Data: Fetch data using the S3 bucket URL for integration with AWS services.

## AWS Glue ETL

1. **Log into AWS Console:** Open the AWS Management Console and go to the AWS Glue service.

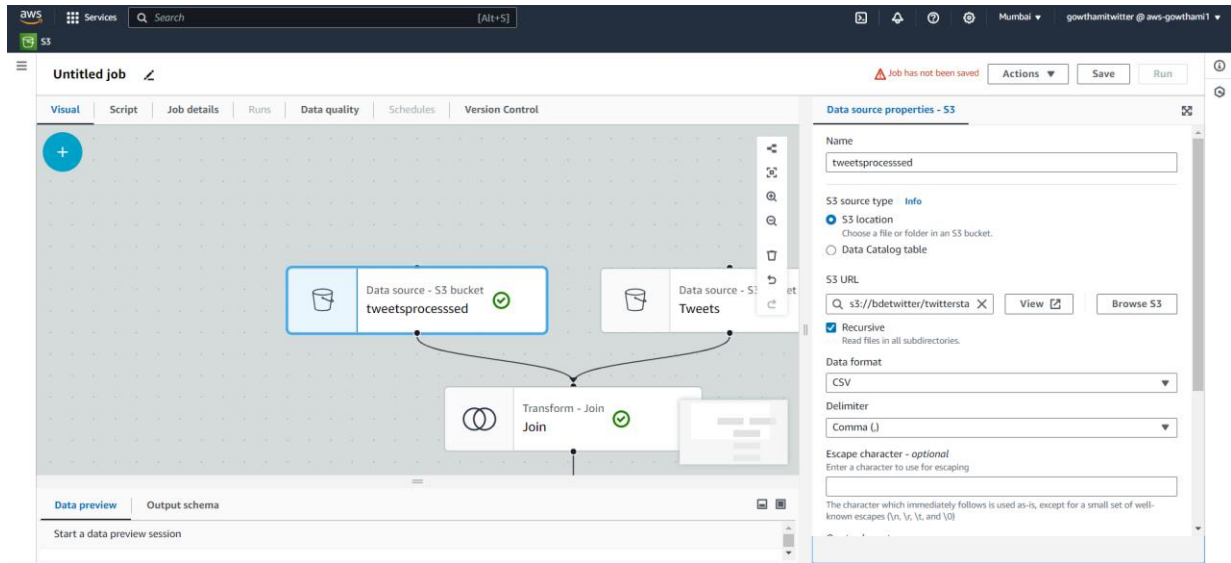


Figure 4.2.4 Creating ETL Job

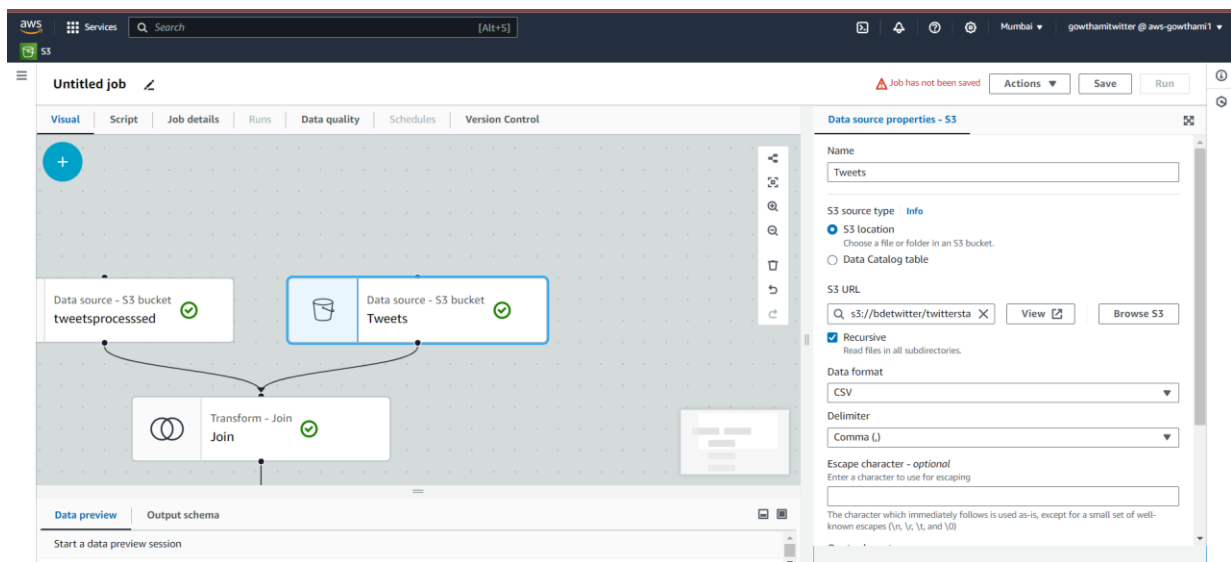


Figure 4.2.5 Defining Data Source Properties

## 2. Create an ETL Job:

- Click on "Jobs" and choose "Add job."
- Specify the job name, IAM role, and select a script editor (such as Python or Scala).

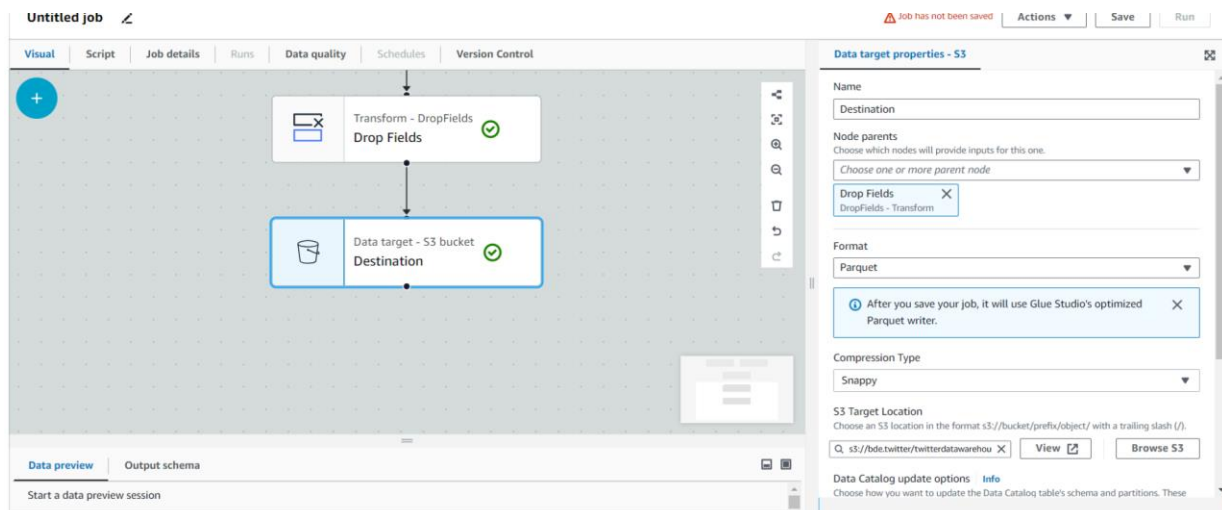


Figure 4.2.6 Defining Target Properties

**3. Write ETL Script:** In the job editor, write or edit the ETL script to extract data, perform transformations (such as cleaning and filtering), and designate the destination (S3 data warehouse).

**4. Schedule the Job (Optional):** Configure a schedule for the ETL job to execute at specified intervals.

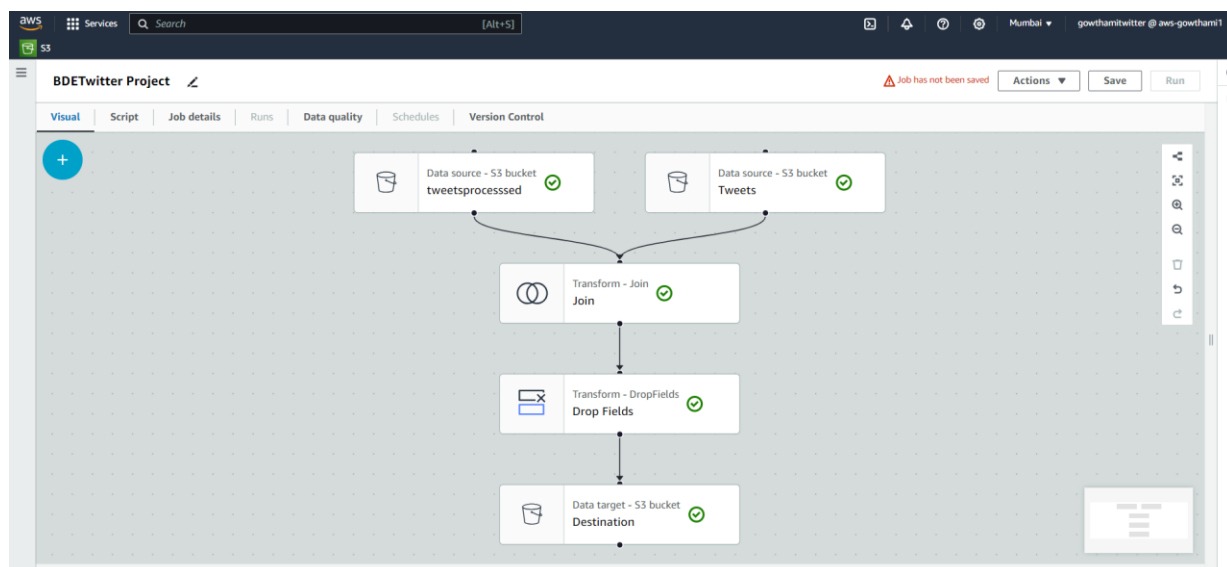


Figure 4.2.7 Visualization of Data Sourcing, Transformation, Target Destination

**5. Run the ETL Job:** Initiate the job to process the data and load it into the designated S3 data warehouse bucket.

**6. Monitor Job Execution:** Utilize the Glue dashboard to track job status and logs for any errors or performance issues.

## **PYSPARK SCRIPT:**

```
import sys

from awsglue.transforms import *

from awsglue.utils import getResolvedOptions

from pyspark.context import SparkContext

from awsglue.context import GlueContext

from awsglue.job import Job

args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()

glueContext = GlueContext(sc)

spark = glueContext.spark_session

job = Job(glueContext)

job.init(args['JOB_NAME'], args)

# Script generated for node Tweets

Tweets_node1726902683565 =

glueContext.create_dynamic_frame.from_options(format_options={"quoteChar": "\"",
"withHeader": True, "separator": ","}, connection_type="s3", format="csv",
connection_options={"paths": ["s3://bde/twitter/twitterstaging/Tweets.csv"], "recurse": True},
transformation_ctx="Tweets_node1726902683565")

# Script generated for node tweetsprocessed

tweetsprocessed_node1726902684657 =

glueContext.create_dynamic_frame.from_options(format_options={"quoteChar": "\"",
"withHeader": True, "separator": ","}, connection_type="s3", format="csv",
connection_options={"paths": ["s3://bde/twitter/twitterstaging/tweetsprocessed.csv"], "recurse":
True}, transformation_ctx="tweetsprocessed_node1726902684657")

# Script generated for node Join

Join_node1726902740142 = Join.apply(frame1=tweetsprocessed_node1726902684657,
frame2=Tweets_node1726902683565, keys1=["text"], keys2=["text"],
transformation_ctx="Join_node1726902740142")
```

```
# Script generated for node Drop Fields
```

```
DropFields_node1726902972522 = DropFields.apply(frame=Join_node1726902740142,  
paths=["sentiment", "text"], transformation_ctx="DropFields_node1726902972522")
```

```
# Script generated for node Destination
```

```
Destination_node1726903251652 =
```

```
glueContext.write_dynamic_frame.from_options(frame=DropFields_node1726902972522,  
connection_type="s3", format="glueparquet", connection_options={"path":  
"s3://bdetwitter/twitterdatawarehouse/", "partitionKeys": []}, format_options={"compression":  
"snappy"}, transformation_ctx="Destination_node1726903251652")  
job.commit()
```

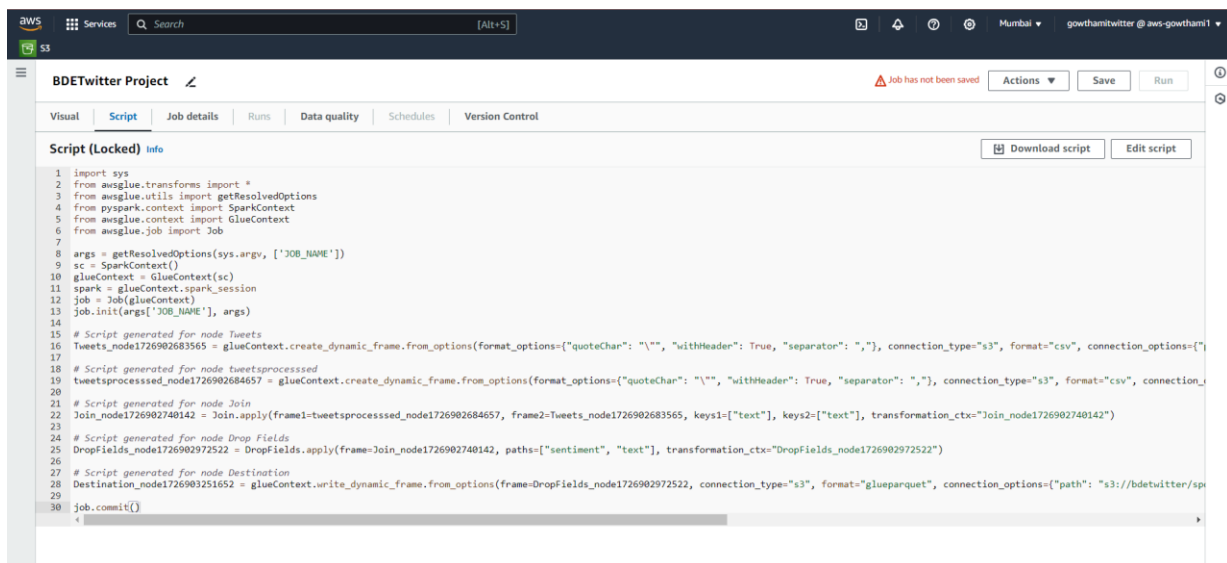


Figure 4.2.8 ETL Job Creation using Pyspark Script.

## Creating IAM Roles for AWS Glue

To allow AWS Glue to access necessary resources, you must establish an IAM (Identity and Access Management) role with specific permissions:

1. Log into AWS Console: Navigate to the IAM service within the AWS Management Console.
2. Create a New Role:
  - Click on "Roles" and choose "Create role."

- Select "Glue" as the service that will utilize this role.
- 3. Attach Policies:
  - Attach policies that provide permissions for S3 (read/write access), Glue operations, and access to the Glue Data Catalog.
  - You can utilize existing policies such as AmazonS3FullAccess and service-role/AWSGlueServiceRole.

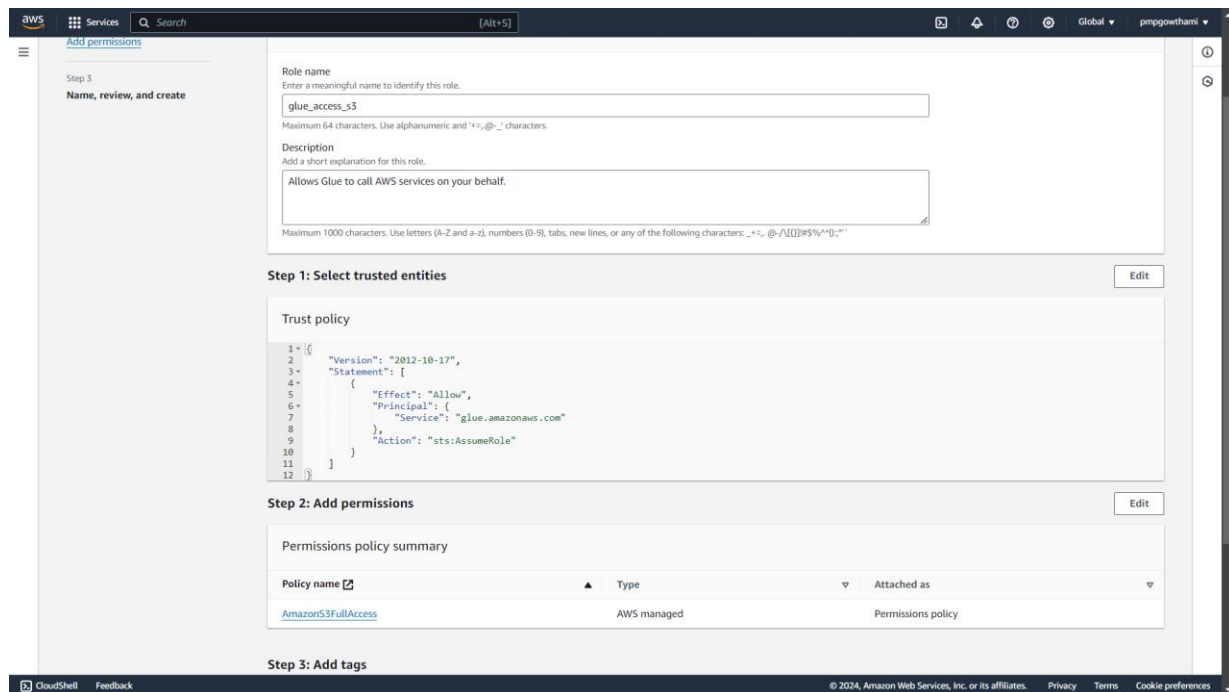


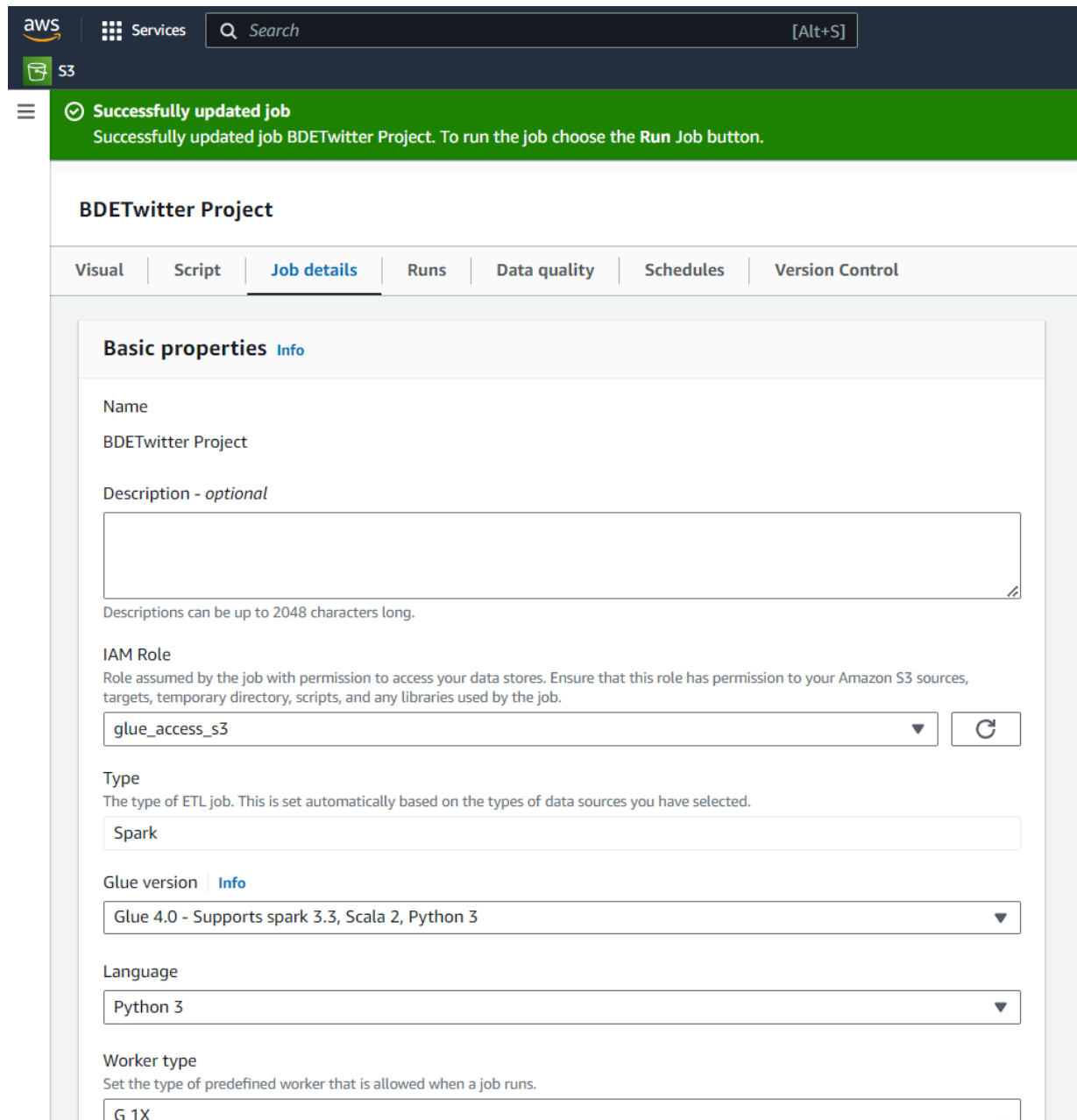
Figure 4.2.9 IAM Role Creation for Glue.

4. Name and Create: Assign a descriptive name to the role (e.g., GlueETLRole) and create the role.
5. Note the Role ARN: Copy the ARN (Amazon Resource Name) of the role for use when setting up your Glue ETL job.

## Configuration of ETL Job Details

1. Create ETL Job: In the AWS Glue console, go to "Jobs" and click on "Add job."
2. Job Details:
  - i. Name: Enter a unique name for the job (e.g., TwitterSentimentETL).
  - ii. IAM Role: Select the IAM role you established for Glue access.
  - iii. Type: Choose "Spark" or "Python Shell" according to your needs.

3. Data Source: Indicate the data source by selecting the database and table created by the Glue crawler.



The screenshot shows the AWS Glue console interface for configuring an ETL job. At the top, there's a navigation bar with the AWS logo, 'Services' link, a search bar, and an '[Alt+S]' shortcut. Below this is a green notification banner stating 'Successfully updated job BDETwitter Project. To run the job choose the Run Job button.' The main section is titled 'BDETwitter Project' and features a horizontal tab bar with 'Visual', 'Script', 'Job details' (active), 'Runs', 'Data quality', 'Schedules', and 'Version Control'. The 'Job details' tab is expanded, showing 'Basic properties' with an 'Info' link. The properties include: 'Name' (BDETwitter Project), 'Description - optional' (a text area with a note that descriptions can be up to 2048 characters long), 'IAM Role' (glue\_access\_s3 with a refresh button), 'Type' (Spark, with a note that it's set automatically based on data sources), 'Glue version' (Glue 4.0 - Supports spark 3.3, Scala 2, Python 3), 'Language' (Python 3), and 'Worker type' (G 1X, with a note to set the type of predefined worker).

Figure 4.2.10 Configuring ETL Job Details.

4. Transformations: Specify the necessary transformations (e.g., filtering, cleaning) using the Glue ETL script editor.

5. Data Target: Identify the destination S3 bucket where the processed data will be saved.



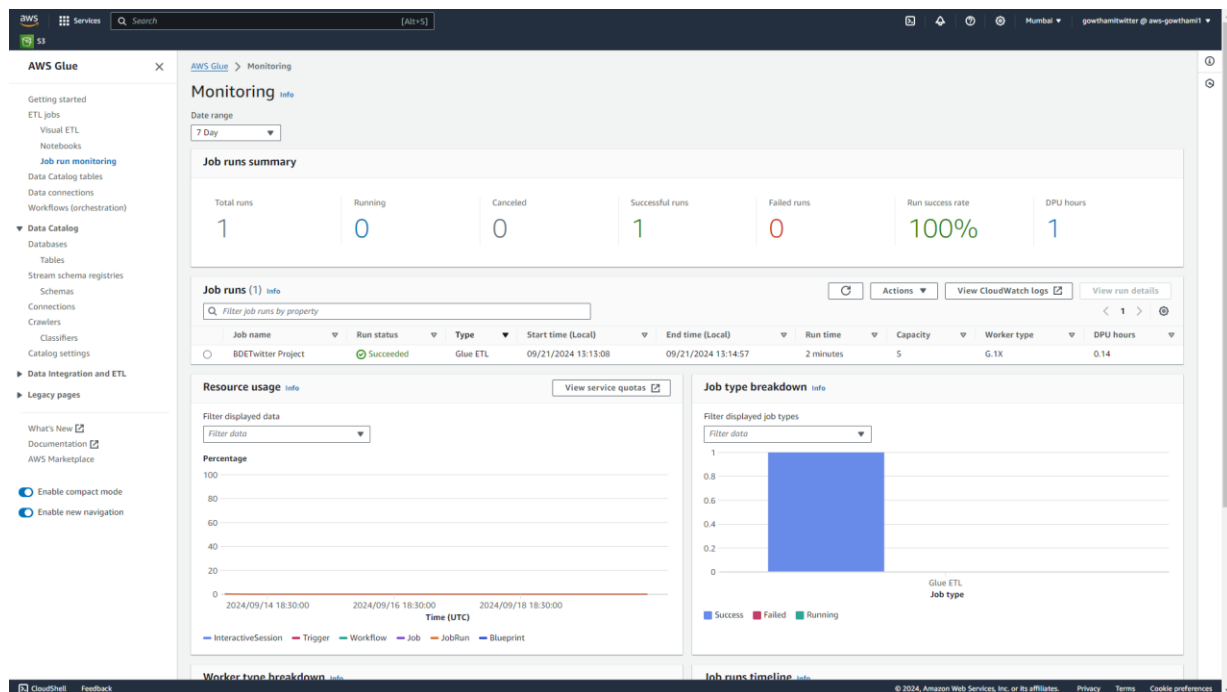


Figure 4.2.11 Job Monitoring and Setting Pipeline.

6. Job Properties: Set up additional properties like job parameters, timeout settings, and retry options.

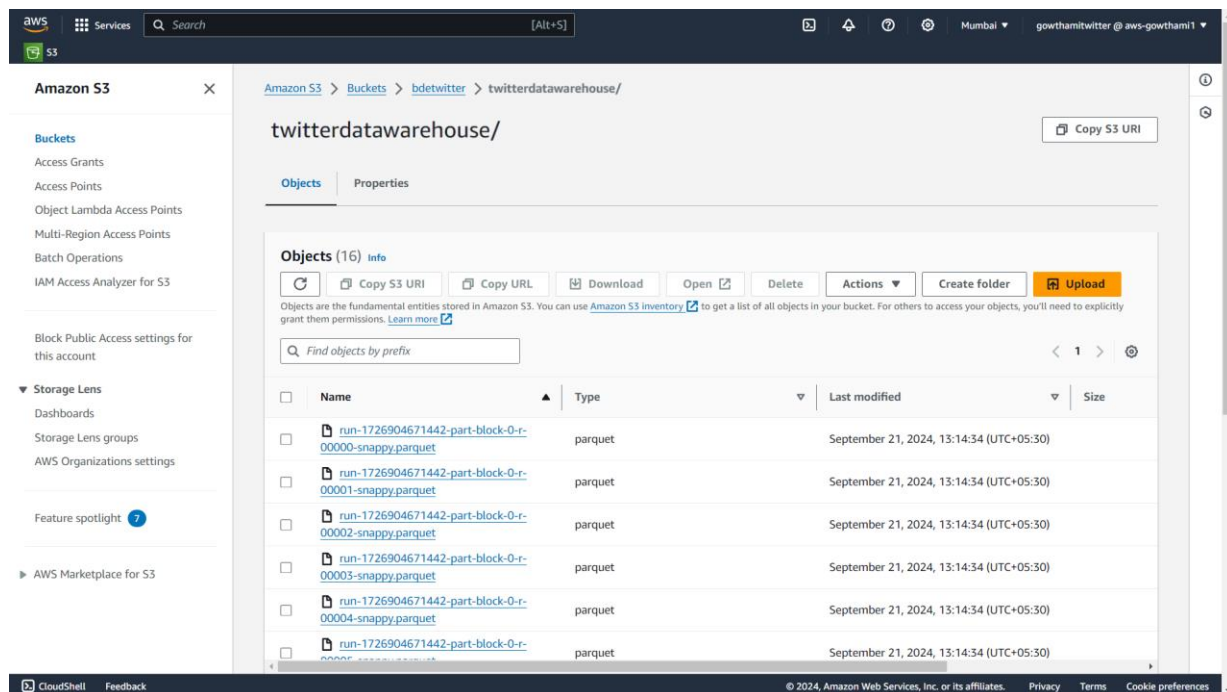


Figure 4.2.12 Twitterdatawarehouse Bucket

7. Save and Run: Save the job configuration and run the job to initiate the ETL process.

## AWS Glue Crawler Implementation:

1. Log into AWS Console: Access the AWS Glue service.

2. Create a New Crawler:

- In the Glue console, click on "Crawlers" and choose "Add Crawler."

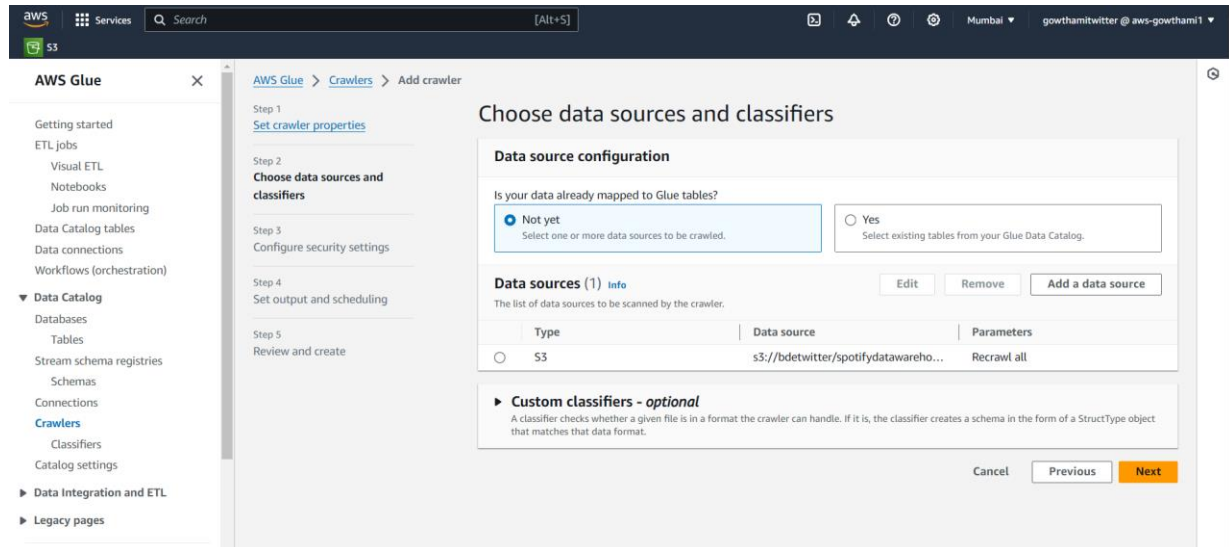


Figure 4.2.13 Adding Crawler

- Provide a name for your crawler.

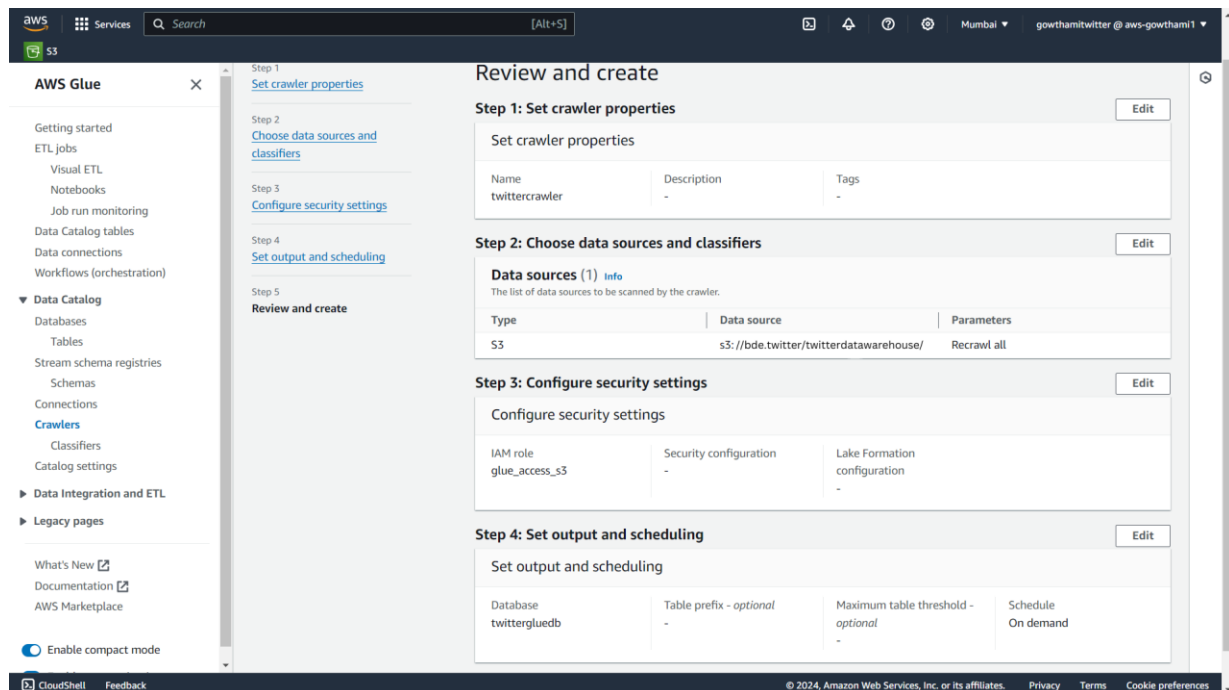


Figure 4.2.14 Defining Data Source and Classifiers

3. Define Data Source:

-Indicate the data source location (e.g., S3 bucket) where your raw data is located.

#### 4. Set IAM Role:

- Select or create an IAM role that grants the crawler access to your data.

#### 5. Choose Output:

- Specify the target database in the Glue Data Catalog where the metadata will be saved, or create a new database.

#### 6. Configure Schedule (Optional):

- Establish a schedule for the crawler to run at regular intervals or leave it set to "Run on demand."

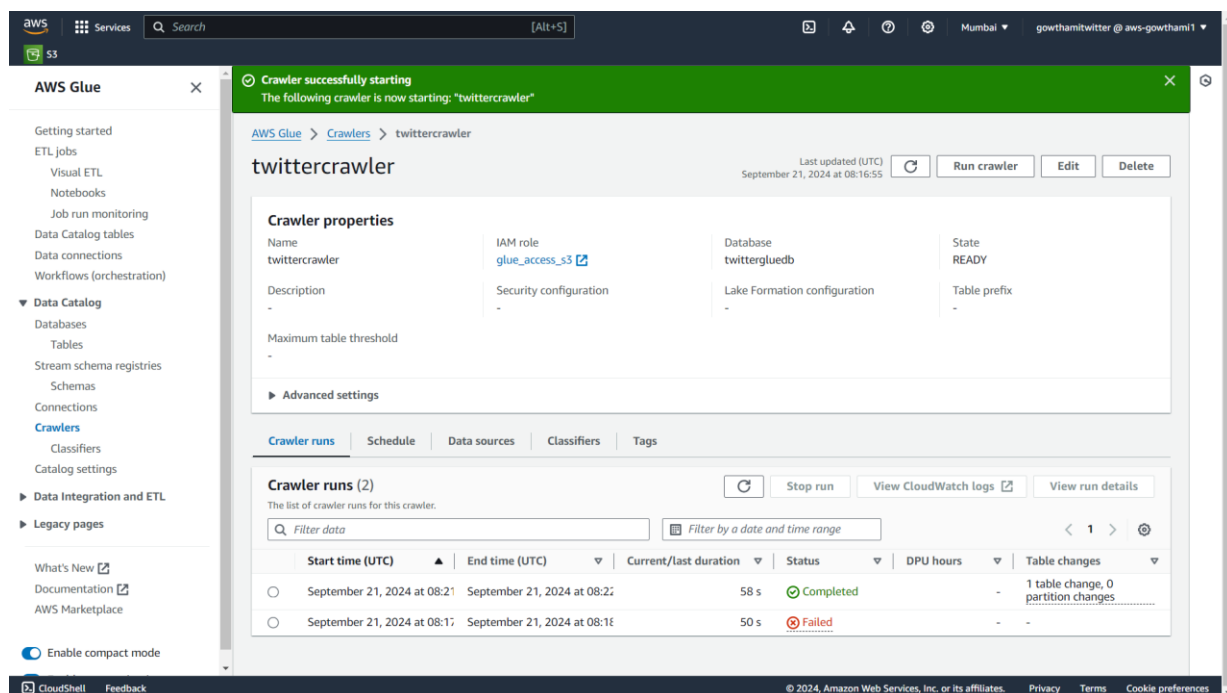


Figure 4.2.15 Running Crawler

#### 7. Run the Crawler:

- After configuration, execute the crawler to automatically identify your data and populate the Glue Data Catalog with metadata.

#### 8. Review Results:

- Inspect the Data Catalog to verify that the tables and schemas were created according to the crawled data.

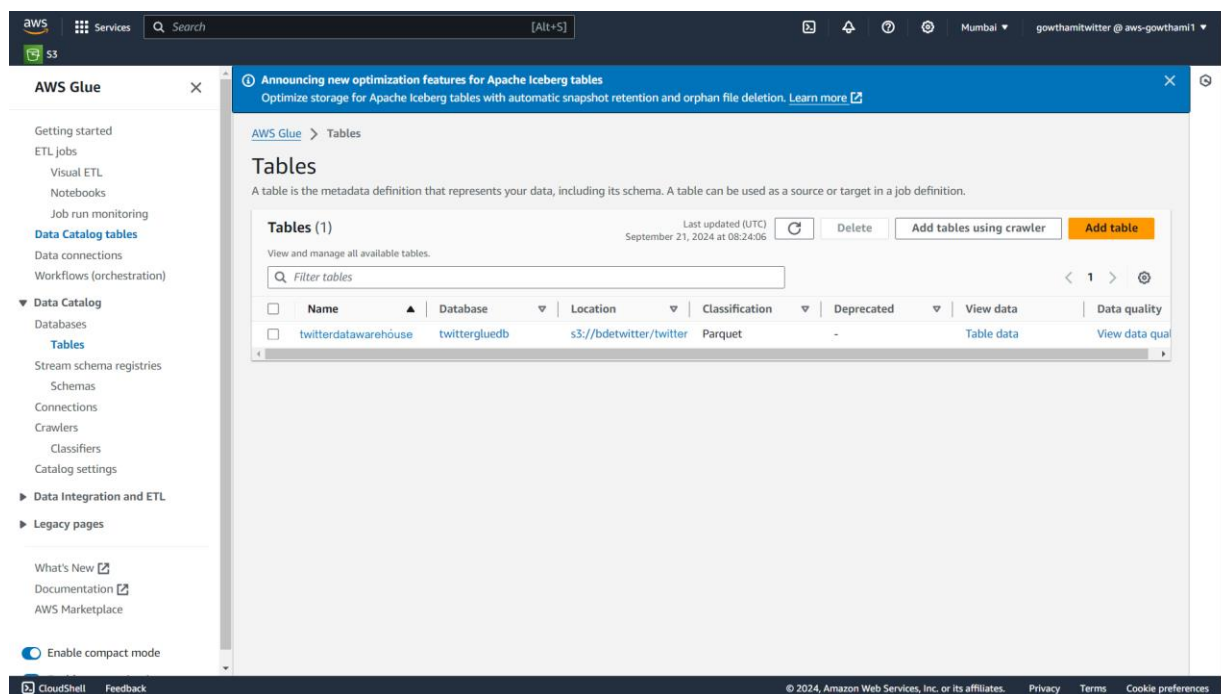


Figure 4.2.16 Creation of Tables by running Crawler.

### 4.3 Implementation Steps for Query Editor ( Amazon Athena ):

1. **Configure Athena:** Make sure that Amazon Athena is properly set up and linked to the S3 data warehouse bucket where the processed data is stored. This involves configuring the necessary permissions and ensuring that the data stored in S3 is in a format that Athena can query, such as Parquet or CSV.
2. **Create Database:** Within the Athena interface, create a new database to effectively organize and manage your tables. Creating a logical database structure not only aids in clarity but also enhances the performance of queries, as it allows you to define specific schemas for your datasets, making data retrieval more efficient.
3. **Define Tables:** Employ SQL statements in Athena to establish tables based on the metadata that was generated by the Glue crawler. These tables should accurately reference the data located in the S3 bucket to ensure seamless querying. When defining tables, specify the appropriate data types for each column and ensure that partitioning is set up correctly if applicable.
4. **Query Data:** Take advantage of the Athena query editor to execute SQL queries on the defined tables. This functionality allows you to analyze the sentiment data thoroughly and extract valuable insights, such as identifying sentiment distributions across different

categories or demographics, observing trends over specific time periods, and correlating sentiment with other variables.

## Querying on AWS Athena

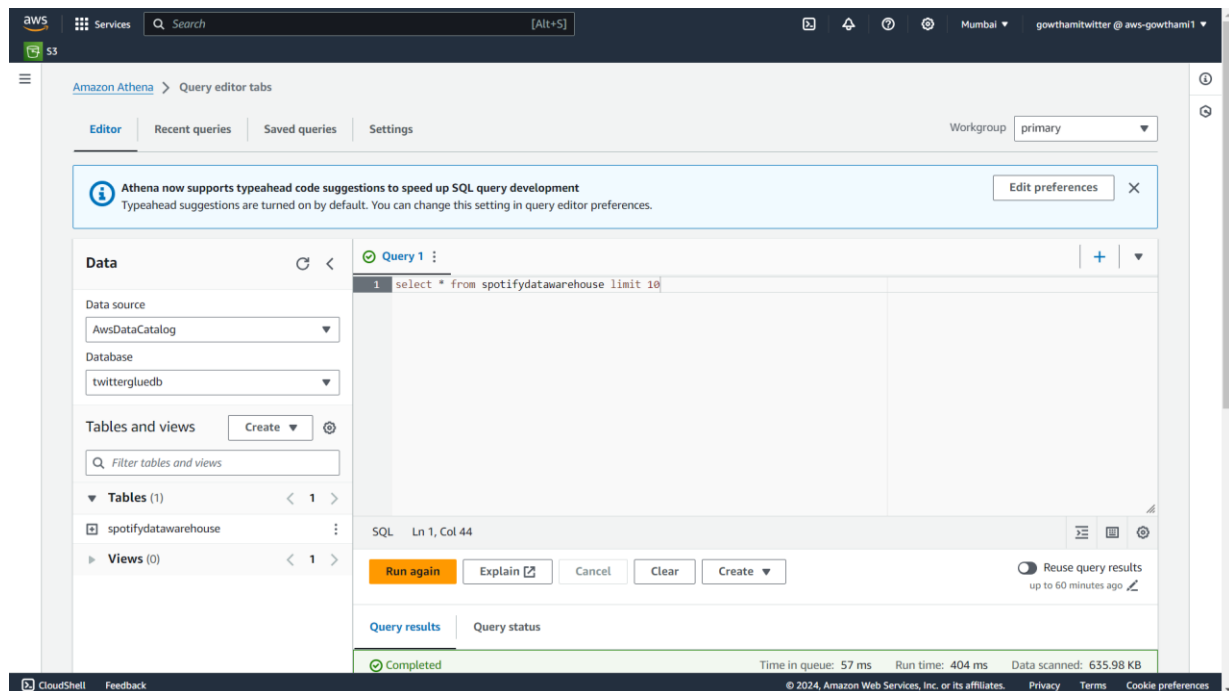


Figure 4.3.1 Athena Query Editor Tabs

All the Query Results are saved in S3 Bucket:

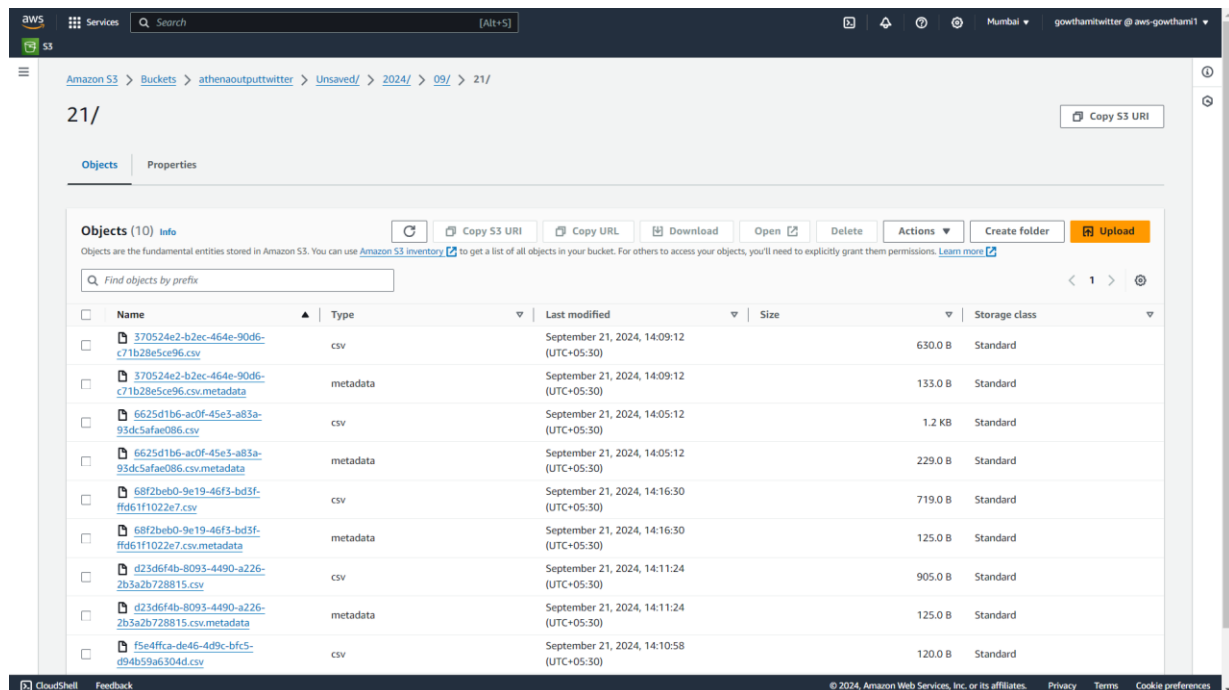


Figure 4.3.2 Query Results saved in S3 Bucket

## **4.4 Implementation Steps for Virtualization (AWS QuickSight) :**

1. **Set Up QuickSight:** Begin by logging into AWS QuickSight and linking it to the Athena database you have established.
2. **Create Data Set:** Within QuickSight, initiate a new dataset by selecting the tables from Athena that you wish to visualize.
3. **Design Analysis:** Utilize the QuickSight interface to create visual representations, such as bar charts, line graphs, or pie charts, to depict sentiment trends and insights.
4. **Create Dashboards:** Integrate multiple visualizations into an interactive dashboard, enabling stakeholders to navigate and explore the data.

# CHAPTER-5

## Results

### 5.1 Querying on AWS Athena

Query for selecting all data from datawarehouse of first 10 entries.

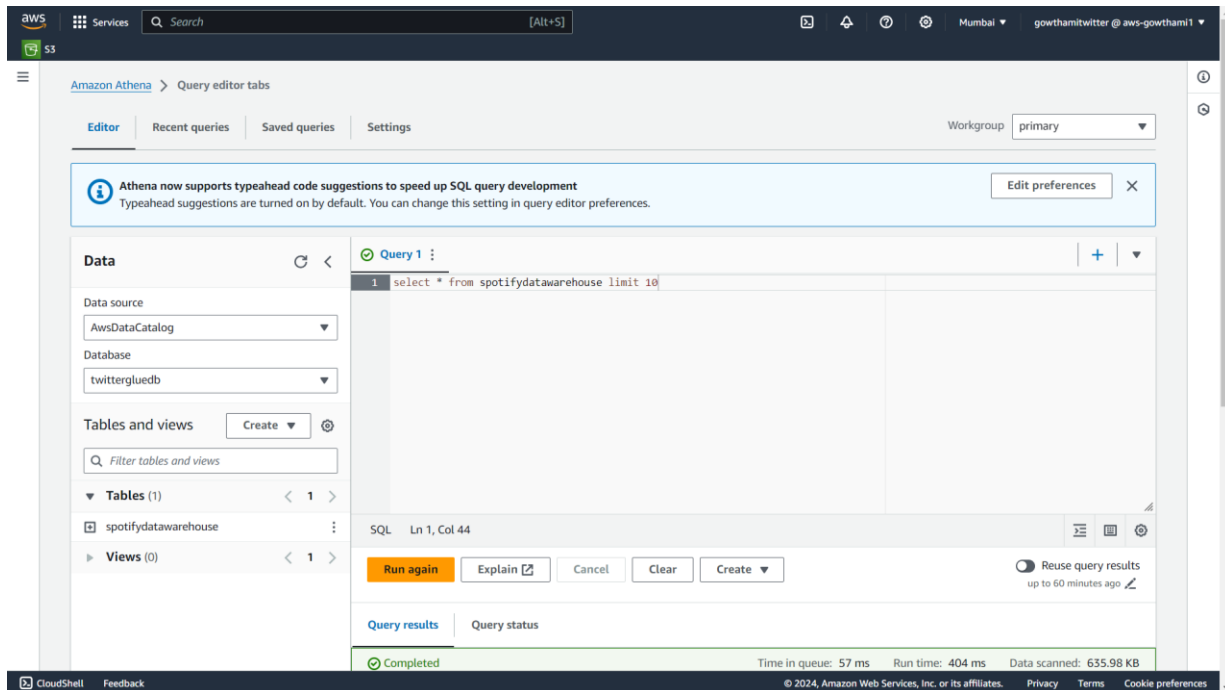


Figure 5.1.1 Query Execution 1

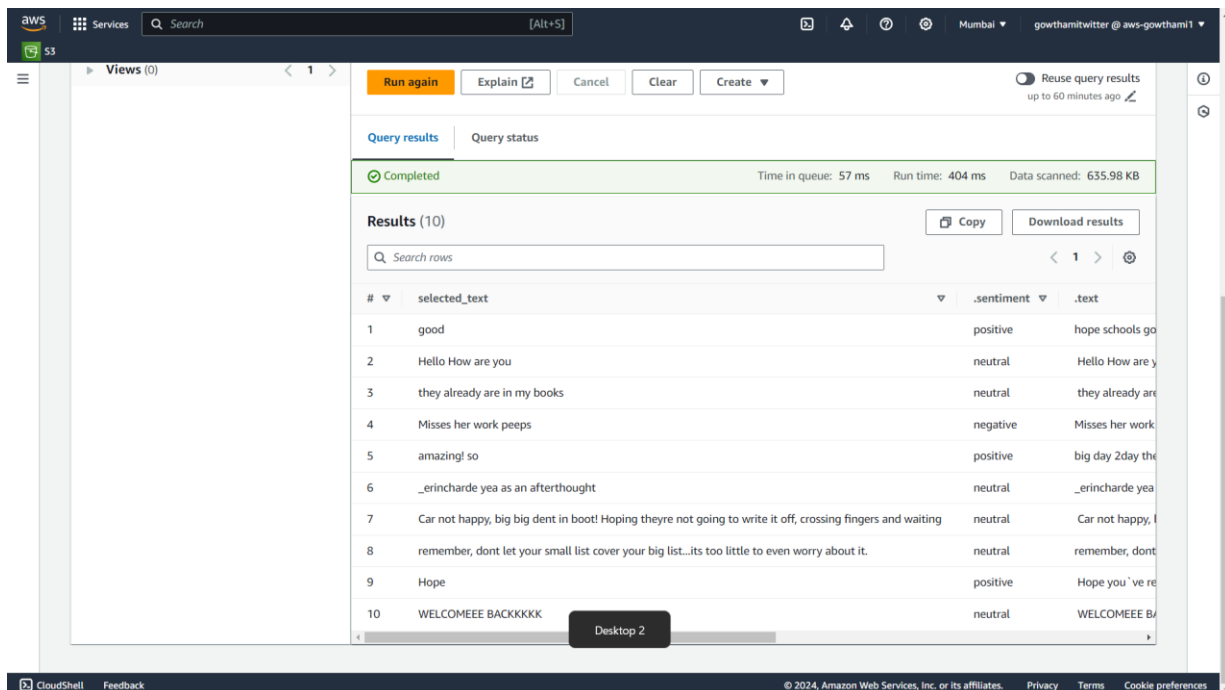


Figure 5.1.2 Query Results 1

Query for Selecting textId,selected\_text of first 10 entries from datawarehouse.

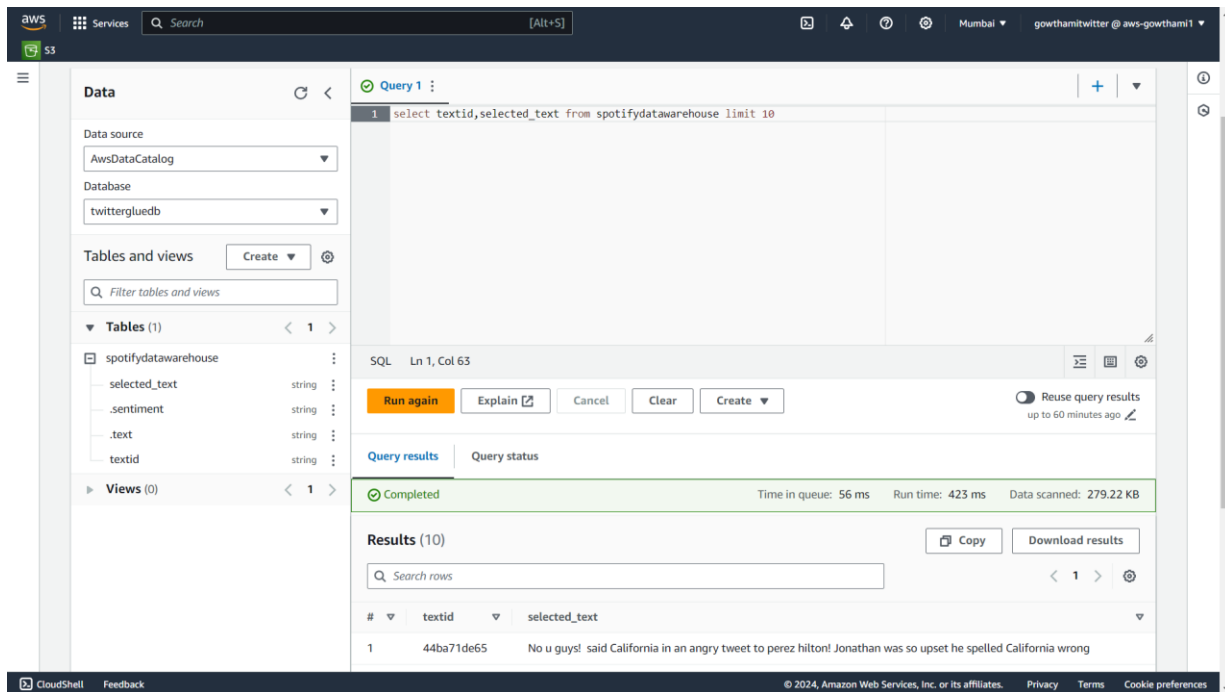


Figure 5.1.3 Query Execution II

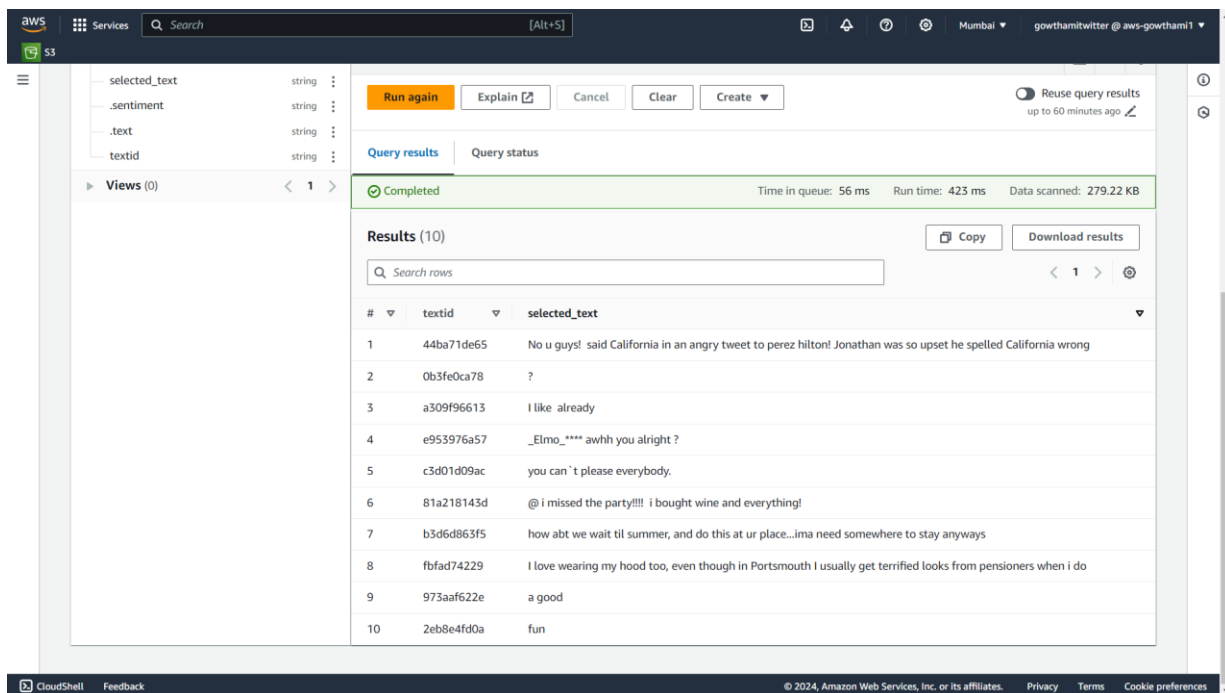


Figure 5.1.4 Query Results II

Query for selecting text,sentiment of first 10 Table entries.



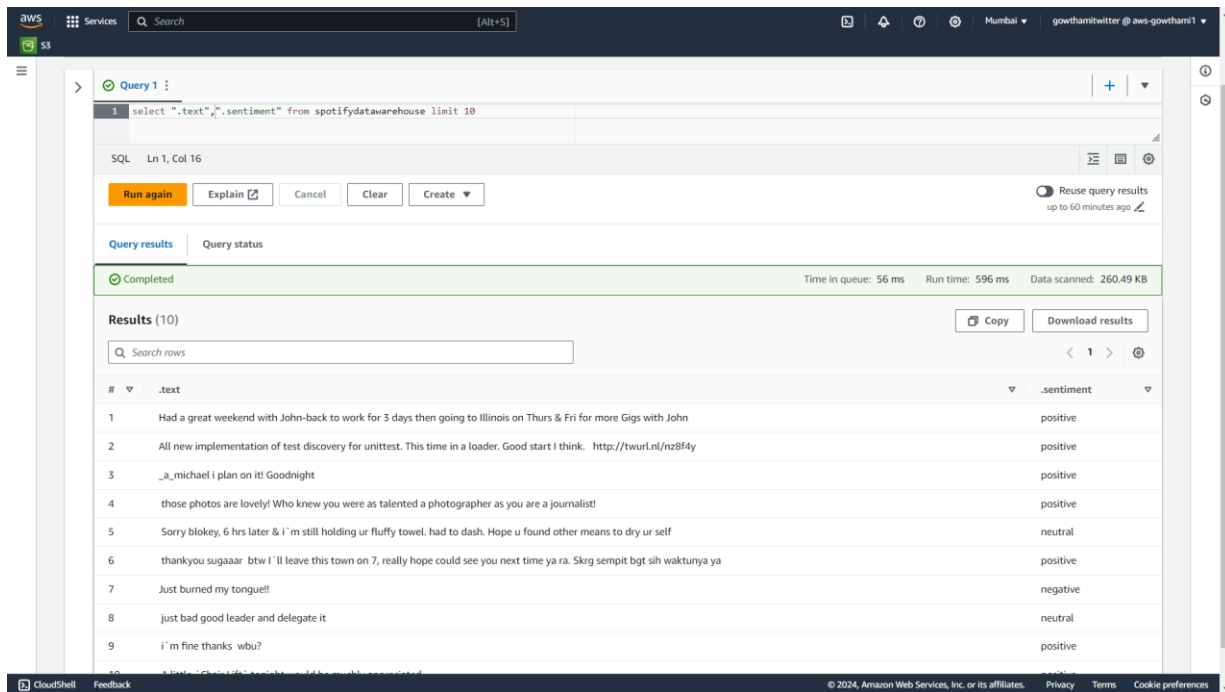


Figure 5.1.5 Query Execution and Results III.

Query for selecting text,sentiment of 10 entries where Sentiment = 'positive'.

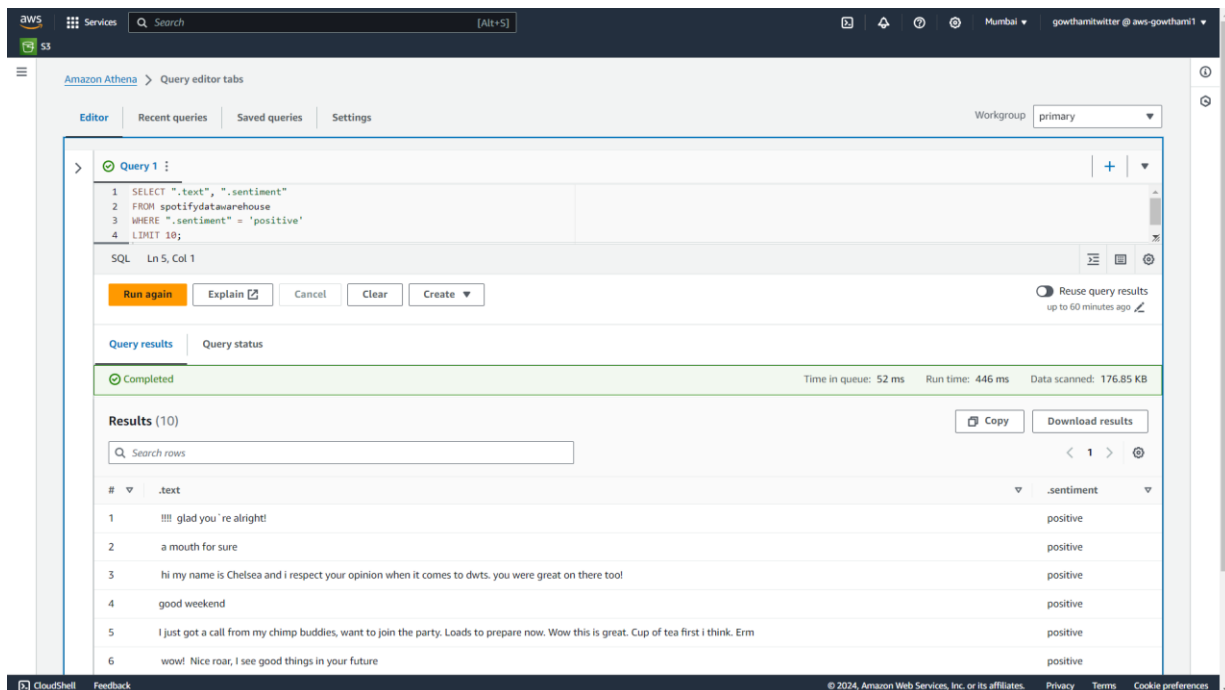


Figure 5.1.6 Query Execution and Results IV.

## 5.2 AWS QUICKSIGHT FOR DATA VISUALISATION:

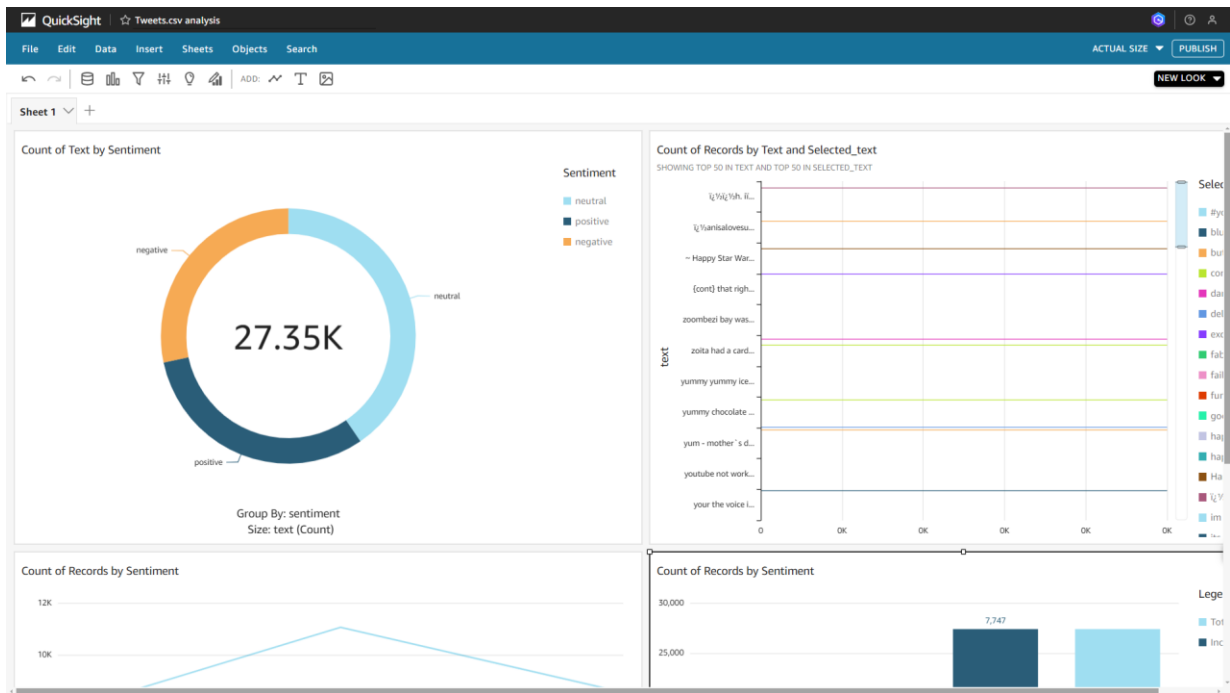


Figure 5.2.1 Visualization of Count of Text by Sentiment

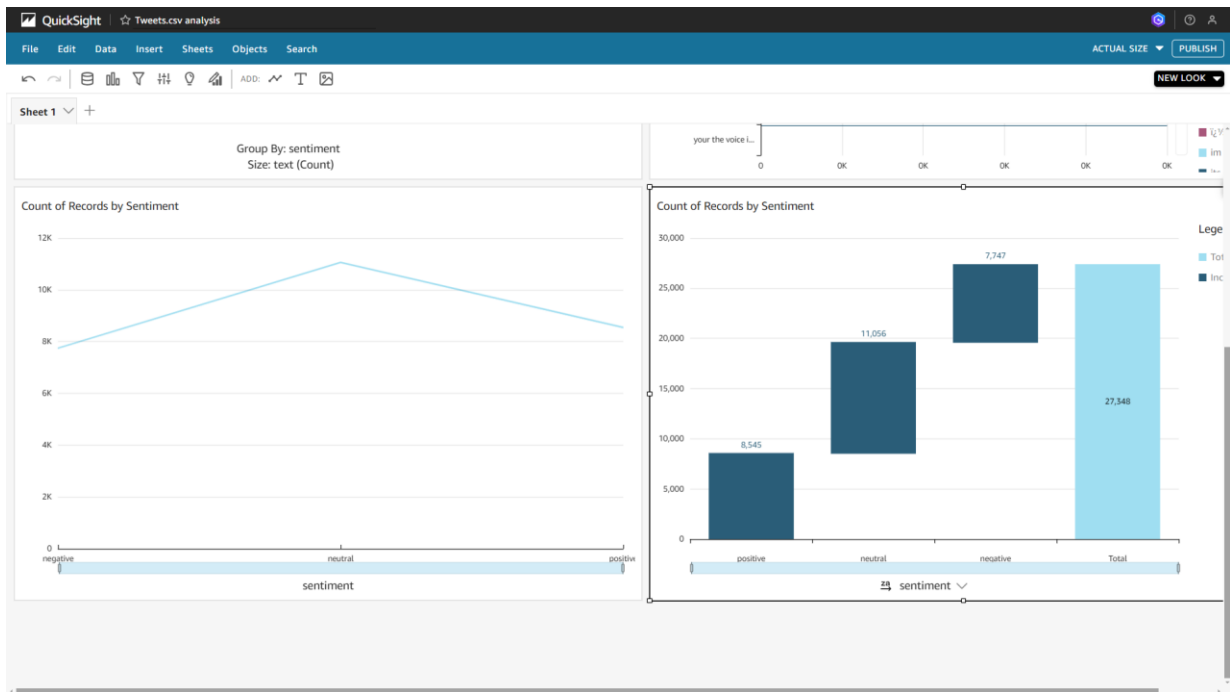


Figure 5.2.2 Visualization of Count of Records by Sentiment

Publish the dashboard and share it with team members or stakeholders, enabling data-driven decision-making based on the visualized sentiment analysis.

This structured approach ensures a seamless flow from data ingestion and processing to querying and visualization, leveraging the capabilities of AWS services effectively.

Donut graph details:

Neutral 11.03k 40%

Positive 8.55k 31%

Negative 7.75k 28%

## CHAPTER-6

### Conclusion

#### 6.1 Conclusion:

This project effectively combines Google Cloud and AWS to conduct comprehensive sentiment analysis on Twitter data. Utilizing Google Sheets, Colab, and Drive for data gathering and preprocessing, followed by AWS Glue for ETL, Athena for querying, and QuickSight for visualization, the project establishes a streamlined pipeline for deriving meaningful insights from social media data. The automated workflow not only facilitates data processing but also guarantees real-time updates, equipping decision-makers with actionable insights. The integration of cloud platforms showcases the adaptability and scalability of big data solutions for sentiment analysis.

#### 6.2 Future Enhancement:

For future enhancements, several improvements can be considered:

- **Real-Time Sentiment Analysis:** Integrating AWS Kinesis would allow for real-time data streaming from Twitter, facilitating immediate sentiment insights instead of relying on batch processing.
- **Advanced Machine Learning Models:** Utilizing more sophisticated ML models for sentiment analysis (e.g., BERT) could enhance the precision of sentiment classification.
- **Multi-Platform Integration:** Broaden the project scope to incorporate data from additional social platforms (Facebook, Reddit) for a thorough sentiment analysis across various channels.
- **User-Centric Dashboards:** Improve the QuickSight dashboard by incorporating interactive, customizable views tailored to different user roles, fostering better decision-making.

These enhancements can augment the project's capabilities and boost its overall efficiency in handling large-scale sentiment analysis tasks.

## CHAPTER-7

### PLAN OF ACTION

#### Team Members:

1. Gowthami [RA2211028010213]
2. Shane [RA2211028010211]
3. Alfred [RA2211028010236]

S.no	Task	Incharge	Deadline	Status
1	Exploratory Data Analysis	Alfred	16/8/24	Completed
2	Function for Model Evaluation	Shane	22/09/24	Completed
3	Model training	Gowthami	22/09/24	Completed
4	Data Preprocessing	Gowthami & Shane	13/10/24	Completed
5	Final Documentation	Alfred & Shane	20/10/24	Completed
6	Final Presentation	Gowthami & Alfred	20/10/24	Completed

Table 7.1 Plan of Action

## CHAPTER-8


### REFERENCES

- [1] S. N. Alzahrani, M. S. Abed, and N. Alabdulrazzaq, "Sentimental Analysis for Response to Natural Disaster on Twitter Data," *World Scientific*, vol. 357, pp. 1-14, 2023.
- [2] M. A. Al-Qaysi, K. K. Al-Azawi, and N. A. Al-Emran, "Sentimental Analysis of Climate Change Tweets," *Americas PG*, vol. 3, no. 1, pp. 1-10, 2023.
- [3] M. P. Prakash and S. S. Ghosh, "Analysis of Real Time Twitter Sentiments using Deep Learning Models," *Journal of Advanced Data Science*, vol. 5, no. 2, pp. 1-12, 2023.
- [4] A. M. Alzahrani and H. A. Alshahrani, "Sentimental and spatial analysis of COVID-19 vaccines tweets," *Journal of Software Engineering and Applications*, vol. 15, no. 3, pp. 1-12, 2023.
- [5] C. Y. Chen and R. K. K. Yadav, "A Twitter Sentiment Analysis for Cloud Providers: A Case Study of Azure vs AWS," *ResearchGate*, vol. 5, no. 1, pp. 1-10, 2023.
- [6] R. S. Patil and S. A. Bhalerao, "Sentiment Analysis of Twitter Data," *ResearchGate*, vol. 5, no. 2, pp. 1-15, 2023.
- [7] K. Kumar and A. Tiwari, "Sentimental Analysis of Twitter Data with respect to General Elections in India," *Procedia Computer Science*, vol. 187, pp. 91-97, 2020.
- [8] J. K. Park and M. S. Lee, "Comparative Studies of Detecting Abusive Language on Twitter," *Papers with Code*, 2023.
- [9] T. S. S. Prakash and A. G. Bansal, "Predicting the Sentiment Polarity of Tweet Replies," *Papers with Code*, 2023.
- [10] R. D. A. Ramesh and K. S. Nair, "Twitter Sentiment Analysis with CNNs and LSTMs," *Papers with Code*, 2023.
- [11] S. K. Jain and A. R. Mehta, "An Implementation of Hybrid Enhanced Sentiment Analysis System using Spark ML Pipeline," *Scopus*, vol. 15, no. 1, pp. 1-10, 2023.
- [12] A. C. R. Sinha and P. K. Singh, "A Survey On Sentimental Analysis and Its Applications," *Scopus*, vol. 15, no. 1, pp. 1-10, 2023.
- [13] H. A. K. Al-Khalidi, S. F. K. A. Hassan, and M. A. H. Al-Azzawi, "Sentiment Analysis of Twitter Data Using Machine Learning Techniques," *ResearchGate*, vol. 182, no. 39, pp. 1-7, 2023.

- [14] S. M. A. Ghafour, M. A. K. Nasr, and R. N. J. Kadhim, "Real-Time Sentiment Analysis of Twitter Data Using Natural Language Processing," *ResearchGate*, vol. 35, no. 1, pp. 95-104, 2023.
- [15] A. F. T. Elhassan and S. B. M. A. Al-Hakim, "Deep Learning Approaches for Sentiment Analysis of Twitter Data: A Comprehensive Review," *ResearchGate*, vol. 220, pp. 1-15, 2023.

# APPENDIX

## PLAGARISM REPORT

Page 2 of 60 - Integrity Overview

Submission ID: tm:oid::1:3063440144





### 14% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.




#### Filtered from the Report

- Bibliography
- Quoted Text

#### Match Groups

-  **80 Not Cited or Quoted 14%**  
Matches with neither in-text citation nor quotation marks
-  **3 Missing Quotations 0%**  
Matches that are still very similar to source material
-  **0 Missing Citation 0%**  
Matches that have quotation marks, but no in-text citation
-  **0 Cited and Quoted 0%**  
Matches with in-text citation present, but no quotation marks

#### Top Sources

- 13%  Internet sources
- 5%  Publications
- 8%  Submitted works (Student Papers)