**Enhancing Train Maintenance with NLP Analysis of Social Media Tweets**

**The Fine Tuners**

Akshay Rajesh

M. Cem Akuras

Erjon Buka

Ecem Günhar Akuras

Shinu Joseph

**Table of Contents**

[1. EXECUTIVE SUMMARY 5](#_Toc155638452)

[1.1. Introduction 5](#_Toc155638453)

[1.2. Data Insights 5](#_Toc155638454)

[1.3. Modeling Overview 5](#_Toc155638455)

[1.4. Evaluation Methodology 6](#_Toc155638456)

[1.5. Deployment Concept 6](#_Toc155638457)

[1.6. Results, Discussion, and Outlook 6](#_Toc155638458)

[1.7. Process and Collaboration 7](#_Toc155638459)

[1.8. Key Takeaways and Recommendations 7](#_Toc155638460)

[2. BUSINESS UNDERSTANDING 8](#_Toc155638461)

[2.1. Introduction 8](#_Toc155638462)

[2.1.1. Project Objective 8](#_Toc155638463)

[2.1.2. Expected Outcomes 8](#_Toc155638464)

[2.2. Stakeholder Analysis 9](#_Toc155638465)

[2.3. Risk Analysis 10](#_Toc155638466)

[2.4. Product vision MVP 11](#_Toc155638467)

[2.5. Architecture 11](#_Toc155638468)

[2.6. Product Alignment with the Business 12](#_Toc155638469)

[2.7. Performance Metrics 12](#_Toc155638470)

[3. DATA UNDERSTANDING AND PREPARATION 14](#_Toc155638471)

[3.1. DATA UNDERSTANDING 14](#_Toc155638472)

[3.1.1. Data Structure 14](#_Toc155638473)

[3.1.2. Column Analysis 14](#_Toc155638474)

[3.1.3. Time Analysis 15](#_Toc155638475)

[3.1.4. Sentiment Analysis 16](#_Toc155638476)

[3.1.5. Topic Classification and Analysis 18](#_Toc155638477)

[3.1.6. Text Analysis: Word Frequency 19](#_Toc155638478)

[3.1.7. Analysis of Author ID 19](#_Toc155638479)

[3.1.8. Missing Values Analysis 20](#_Toc155638480)

[3.1.9. Duplicates and Outliers 21](#_Toc155638481)

[3.2. DATA PREPARATION 21](#_Toc155638482)

[3.2.1. Launching Data Preparation 21](#_Toc155638483)

[3.2.2. ETL Process Enhancements 21](#_Toc155638484)

[3.2.3. Before and After ETL Comparison 22](#_Toc155638485)

[3.2.4. Impact of ETL on Topic Distribution 23](#_Toc155638486)

[3.3. An Overview of Data Understanding and Preparation 24](#_Toc155638487)

[3.3.1. Data Structure and Annotations 24](#_Toc155638488)

[3.3.2. Data Quantity, Distributions, and Representation 25](#_Toc155638489)

[3.3.3. Comprehensive Dataset Usage 25](#_Toc155638490)

[3.3.4. Handling Missing Values and Outliers 25](#_Toc155638491)

[3.3.5. Decision Clarification 25](#_Toc155638492)

[4. MODELLING 26](#_Toc155638493)

[4.1 Pre-Processing 26](#_Toc155638494)

[4.2 Irony Detection 26](#_Toc155638495)

[4.3 Sentiment Analysis 26](#_Toc155638496)

[4.4 ChatGPT suggestions 26](#_Toc155638497)

[5. EVALUATION 28](#_Toc155638498)

[5.1.1. Purpose of Evaluation 28](#_Toc155638499)

[5.1.2. The Importance of Evaluation 28](#_Toc155638500)

[5.1.3. Evaluation Process 28](#_Toc155638501)

[5.1.4. The Process of Modelling and Evaluation 28](#_Toc155638502)

[5.2. Evaluation of Irony 30](#_Toc155638503)

[5.2.1. Defining Irony 30](#_Toc155638504)

[5.2.2. Significance of Identifying Irony 30](#_Toc155638505)

[5.2.3. Importance of Irony Detection 30](#_Toc155638506)

[5.2.4. Evaluation of Irony Detection 30](#_Toc155638507)

[5.2.5. Process of Irony Evaluation 32](#_Toc155638508)

[5.2.6. Evaluation Scores of Irony 32](#_Toc155638509)

[5.3. Evaluation of Sentiment Analysis 33](#_Toc155638510)

[5.3.1. Defining Negative Sentiment 33](#_Toc155638511)

[5.3.3. Evaluation of Sentiment Analysis 33](#_Toc155638512)

[5.3.4. Process of Sentiment Analysis 34](#_Toc155638513)

[5.3.5. Evaluation Scores of Sentiment Analysis 35](#_Toc155638514)

[5.4. Evaluation of LLM Analysis 36](#_Toc155638515)

[5.4.1. Purpose of LLM Analysis 36](#_Toc155638516)

[5.4.2. Evaluation of LLM Analysis 36](#_Toc155638517)

[5.4.3. LLM Analysis Result Evaluation 36](#_Toc155638518)

[5.4.4. Necessary Future Actions 37](#_Toc155638519)

[5.5. Evaluation of KPIs 37](#_Toc155638520)

[6. DEPLOYMENT 39](#_Toc155638521)

[6.1.1. What is deployment? 39](#_Toc155638522)

[6.1.2. Objective of our project deployment 39](#_Toc155638523)

[6.1.3. Pre-requisites and Interfaces 39](#_Toc155638524)

[6.2. Deployment Architecture 40](#_Toc155638525)

[6.2.1. Extraction Phase 41](#_Toc155638526)

[6.2.2. Transformation Phase 42](#_Toc155638527)

[6.2.3. Loading Phase 44](#_Toc155638528)

[6.3. Challenges 45](#_Toc155638529)

[6.4. Best Practices for Sustained High Performance of Deployed Models 45](#_Toc155638530)

[7. RESULTS 47](#_Toc155638531)

[8. PROCESS 50](#_Toc155638532)

[8.1. Project Inception and Team Dynamics 50](#_Toc155638533)

[8.2. Agile Methodology and Communication 50](#_Toc155638534)

[8.3. Utilization of Development Tools 51](#_Toc155638535)

[8.4. Utilization of Project Management Tools 52](#_Toc155638536)

[8.5. Role Evolution and Adaptation 53](#_Toc155638537)

[8.6. Continuous Learning and Feedback Integration 53](#_Toc155638538)

[9. WORK BREAKDOWN 54](#_Toc155638539)

# 1. EXECUTIVE SUMMARY

# 1.1. Introduction

In this project, we have employed advanced NLP techniques to analyse social media tweets, targeting Thameslink's service users. This approach aimed to unearth valuable insights for improving train maintenance. The initiative aligned with Thameslink's commitment to operational excellence, leveraging data-driven decision-making and a customer-focused approach.

Additionally, this project is pioneered in its use of social media as a direct feedback mechanism to inform and enhance maintenance strategies. The real-time nature of social media allowed for immediate identification of issues, which is important in the context of train maintenance where safety and reliability are paramount. This approach not only aimed to improve the quality of maintenance services but also to demonstrate Thameslink's proactive stance in leveraging technology for customer service excellence.

# 1.2. Data Insights

The data exploration phase involved a thorough analysis of 16,949 tweets, with a focus on sentiments and topics specifically related to Thameslink services. This analysis revealed significant trends and patterns, such as a notable correlation between negative sentiments and longer text posts. Our data preparation process was marked by substantial ETL enhancements, ensuring data clarity and consistency. Special attention was given to removing redundant columns and rows to zero in on customer-generated content for the most relevant insights.

# 1.3. Modeling Overview

In the modeling phase of our project, we employed advanced Natural Language Processing (NLP) and Language Model Learning (LLM) techniques to analyze a dataset of Thameslink-related tweets. Our process began with data extraction and pre-processing, where irrelevant data such as tweets from the service provider's account or those mentioning delays were filtered out. The core of the modeling involved irony detection and sentiment analysis using the TweetNLP library. This approach ensured accurate sentiment capture, crucial for identifying maintenance-related issues from user tweets. The final step involved categorizing these tweets into actionable insights for the Thameslink maintenance team, effectively bridging the gap between social media feedback and practical maintenance strategies.

# 1.4. Evaluation Methodology

Our evaluation strategy employed a systematic approach, beginning with Irony and Sentiment Analysis and progressing to LLM Analysis. We concluded with an assessment of the impact on Key Performance Indicators (KPIs). This comprehensive process effectively demonstrated the accuracy of our models in detecting irony and sentiment and their efficacy in generating actionable recommendations for train maintenance improvements.

Furthermore, the evaluation process included a detailed analysis of the model's performance metrics, such as precision, recall, and F1 score. These metrics provided a quantitative assessment of the model's ability to accurately identify and categorize tweets, ensuring that our recommendations for maintenance improvements were based on reliable and validated data.

# 1.5. Deployment Concept

The deployment of our solution included an integrated end-to-end pipeline, which involved acquiring data from Twitter, using Azure serverless functions, and visualizing data with Power BI. Our system was designed to be scalable, secure, and capable of analyzing sentiments in real-time. It used data processing modules to analyse tweets and categorize their sentiments on the spot.

Moreover, during the deployment phase, we focused on making the system flexible and responsive. This was important to keep up with the constantly changing nature of social media data. We added features to continuously monitor and update the system, allowing it to stay effective and relevant by recognizing new trends and patterns in social media conversations.

# 1.6. Results, Discussion, and Outlook

Our project will help solve problems faster by successfully detecting basic maintenance-related problems from user tweets. We recognize the importance of integrating user feedback for continuous improvement. It is important to include additional data sources, such as operational data, to further improve the model's accuracy and suitability.

# 1.7. Process and Collaboration

The development of 'Tweets2GPT' was led by 'The Fine Tuners,' a team skilled in adapting roles from Project Owners to Scrum Masters, and Developers, ensuring a dynamic and comprehensive project approach. Rooted in Agile methodology, the project thrived on iterative development, with daily standups central to communication and alignment. Key tools like GitHub and Bash streamlined workflow and collaboration, while Visual Studio Code and Jupyter Notebook were essential for coding and model development. The use of Twitter and ChatGPT APIs was needed in data harvesting and response generation. ClickUp played a pivotal role in project management, supporting systematic Agile processes and transparent task tracking.

Furthermore, the project's success was significantly attributed to the diverse expertise and collaborative effort of the team members. Each member brought unique skills and perspectives, contributing to a rich, multidisciplinary approach to problem-solving. This diversity not only fostered innovation but also ensured comprehensive coverage of all aspects of the project, from technical development to stakeholder engagement and user experience design.

# 1.8. Key Takeaways and Recommendations

The project highlighted the significant potential of NLP and LLM in extracting valuable insights from social media for practical applications in train maintenance. We recommend ongoing monitoring, managing data drift, and establishing feedback loops. Considering additional data types is crucial for ensuring sustained high performance and relevance of the deployed models. By refining our approach and adapting to new data, we can maintain the effectiveness of our solution in improving train maintenance strategies.

# 2. BUSINESS UNDERSTANDING

# 2.1. Introduction

The advent of social media platforms, particularly X (formerly Twitter), has revolutionized the way information is circulated and consumed. Today, X (formerly Twitter), serves as a real-time information network where users share their experiences, opinions, and feedback about various services, including public transportation. Our project aims to leverage this wealth of data to enhance train maintenance by analyzing tweets from users of the train service.

Effective train maintenance is crucial for ensuring the safety, reliability, and efficiency of train services. Traditionally, train maintenance schedules have been determined based on predefined intervals or distance travelled. However, this approach does not consider the real-time condition of the trains, which can lead to unnecessary maintenance activities or unexpected breakdowns. By analyzing tweets from train service users, we can gain insights into the real-time performance and condition of the trains. Users often tweet about their experiences, such as delays, breakdowns, or other issues, which can serve as early indicators of potential maintenance needs.

# 2.1.1. Project Objective

The primary objective of this project is to utilize Natural Language Processing (NLP) and Language Model Learning (LLM) to automatically derive actionable recommendations from tweets. By analyzing the tweets from Thameslink’s service users, we aim to extract valuable insights that can lead to the improvement of the maintenance process, quicker issue resolution, and enhanced user satisfaction.

# 2.1.2. Expected Outcomes

The expected outcomes of this project are:

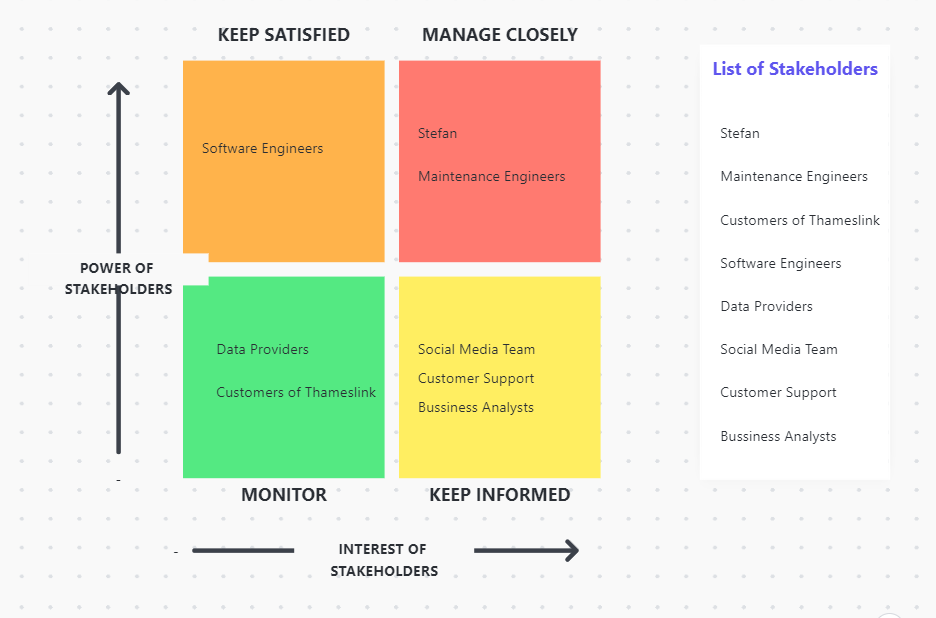
***Improved Maintenance Process****:* By identifying common issues reported by users in their tweets, we can pinpoint areas in the maintenance process that need attention. This data-driven approach allows for more targeted and effective maintenance efforts.

***Quick Issue Resolution****:* Real-time analysis of tweets can help identify issues as they arise, enabling quicker response times from the maintenance team. This can minimize downtime and disruption to services.

***Enhanced User Satisfaction****:* By addressing issues promptly and improving the overall quality of the train services, we anticipate an increase in user satisfaction. Furthermore, this project demonstrates Thameslink’s commitment to listening to its users and continuously improving its services based on their feedback.

# 2.2. Stakeholder Analysis

We have identified several key stakeholders and used a Power and Interest matrix to categorize them. This matrix is a useful tool for managing stakeholders based on their level of authority (‘power’) and their level of concern (‘interest’) regarding the project outcomes.



***Illustration 1:*** *Power and Interest Matrix*

***Software Engineers***

Software engineers are crucial to the development and implementation of the NLP and LLM models. However, as they are primarily concerned with the technical aspects, their interest in the broader business outcomes might be limited. Therefore, it’s important to ***keep them satisfied*** by ensuring they have the resources and support needed to do their job effectively.

***Clients and Maintenance Engineers***

Clients and maintenance engineers have a high degree of interest and power in this project. Clients are interested in the benefits that the project can bring to their business, while maintenance engineers will be directly using the insights generated by the project to enhance their work. Therefore, these stakeholders need to be **managed closely** to ensure their needs and expectations are met.

***Data Providers and Customers***

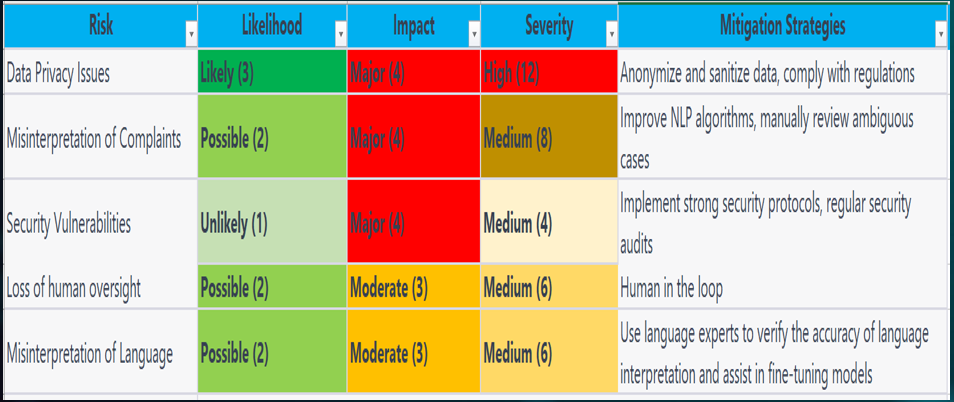
Data providers and customers have a significant interest in the project. Data providers enable the project by supplying the necessary data. However, their power to influence the project is relatively low. Therefore, it’s important to **monitor** these stakeholders to understand their needs and concerns and address any issues that may arise.

***Social Media Team, Customer Support, Business Analysts***

The social media team, customer support, and business analysts play a supportive role in the project. They have a moderate level of interest and power. The social media team and customer support can provide valuable feedback and insights, while business analysts can help interpret the results and guide decision-making. Therefore, these stakeholders should be **kept informed** about the project’s progress and outcomes.

# 2.3. Risk Analysis

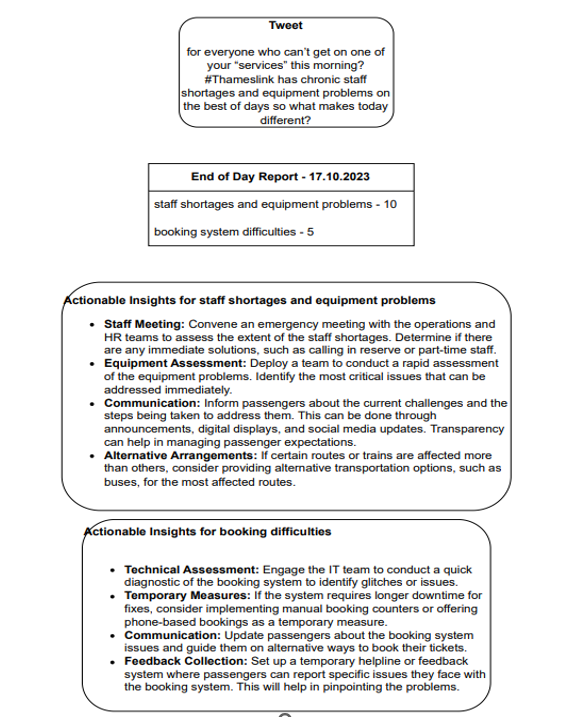
Risk Analysis is captured through a matrix that considers both the likelihood of occurrence and potential impact, offering a more nuanced understanding than focusing on a single factor. This approach facilitates the prioritization of risks, enabling easy identification of those requiring immediate attention, monitoring, or acceptance based on their likelihood and impact. It also acts as a structured framework for decision-making, aiding in the assessment and comparison of risks and guiding the development of mitigation strategies and contingency plans.



***Illustration 2:*** *Risk Analysis*

# 2.4. Product vision MVP

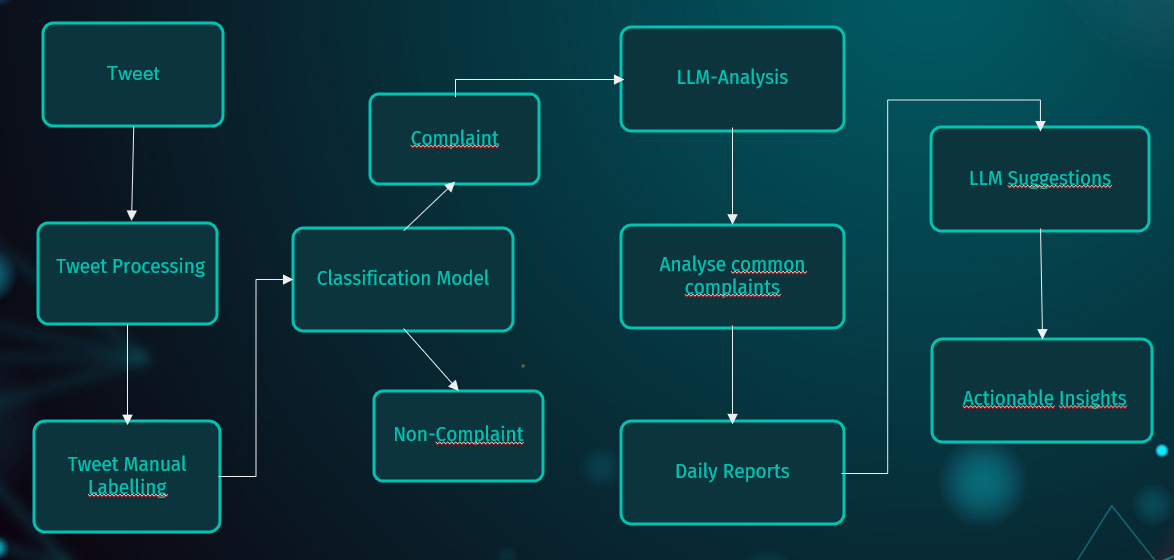
The product vision is to analyse the tweets and do the necessary pre-processing and finally provide an end of the day report to the maintenance team just like below. This report will include the issue topics and steps on how these issues can be resolved.



***Illustration 3:***

# 2.5. Architecture

Since we followed the CRISP DM process, we were able to come back to the initial architecture and make some changes to suit our project needs. The initial architecture was as below:

 ***Illustration 4:*** *Architecture*

We decided to make some changes because we observed:

* Many of the tweets are from Thameslink updates which is not useful for the project.
* We also decided to remove tweets about delays because a delay is not really a maintenance problem. There are other external factors such as weather, operational issues, or unforeseen events that can contribute to delays.
* We observed there were some ironic tweets which needs to be captured before we push the data to our sentiment analysis model, so a model to detect irony is used before passing the output to sentiment analysis model.
* Finally, we passed ironic and negative tweets to the LLM model to get the actionable recommendations.

# 2.6. Product Alignment with the Business

The proposed product aligns seamlessly with Thameslink's business, enhancing maintenance operations, promoting data-driven decision-making, improving operational efficiency, and fostering a customer-centric approach.

***Enhanced Maintenance Operations:*** The application of NLP and LLM to analyze tweets enables real-time identification and resolution of issues, improving the safety and reliability of train services.

***Data-Driven Decision Making:*** The product derives valuable insights from tweet analysis, empowering Thameslink to refine services based on real-time user feedback, aligning strategies with user experiences and needs.

***Operational Efficiency and Proactive Issue Resolution:*** Early issue identification leads to proactive resolution, reducing downtime, preventing major breakdowns, and saving costs associated with extensive repairs.

***Customer-Centric Approach:*** Focusing on user feedback and enhancing the overall customer experience reflects Thameslink's dedication to prioritizing customers at the core of their operations.

# 2.7. Performance Metrics

The success of this project can be evaluated using the following performance metrics:

***Maintenance Issue Resolution Time:*** This metric measures the time taken to resolve maintenance issues identified through the analysis of tweets. A decrease in this metric over time would indicate that the project is helping to identify and address maintenance issues more quickly.

***Maintenance Cost Savings:*** This metric quantifies the cost savings achieved through more efficient and targeted maintenance activities. This could be measured by comparing the maintenance costs before and after the implementation of the project.

***Customer Satisfaction:*** Customer satisfaction can be assessed through surveys or by analyzing sentiment in tweets. An increase in customer satisfaction would indicate that the improvements in maintenance are positively impacting the user experience.

# 3. DATA UNDERSTANDING AND PREPARATION

The Tweets2GPT project aims to leverage Natural Language Processing (NLP) to improve train maintenance strategies, with a particular focus on analysis of Thameslink-related tweets. In this report, the Data Understanding and Data Preparation stages of the project, which is a joint work of 'The Fine Tuners' team, are examined.

Our main goal in this section of the report is to examine the processes involved in converting raw, unstructured Twitter data into a refined format suitable for NLP techniques. This includes preparatory steps to examine the data, assess its quality, and ensure it is ready for analytical procedures. Future sections will outline the methodologies and strategies used in these stages and provide a clear roadmap of our approach, from data collection to transformation.

# 3.1. DATA UNDERSTANDING

# 3.1.1. Data Structure

The critical step in understanding any data set is understanding its structure. The Tweets2GPT project's dataset is collected from two different platforms: Brandwatch and Sprinklr.

The dataset consists of 16,949 rows and 20 columns, representing a significant amount of data that needs to be worked on. This data is in JSON (JavaScript Object Notation) format, which is typically used to store and transport data.

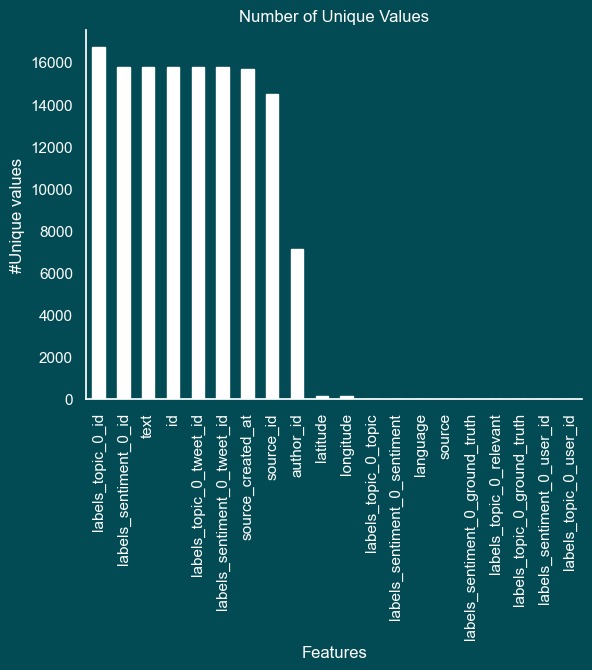
The dataset is diverse in terms of data types. It includes a Date/Time column, which is crucial for analysing tweets chronologically. The dataset also contains four categorical, eight numeric, and one string data type, along with six columns with constant values. This mix of data types provides a well-rounded view of information.

But the dataset is not without its challenges. One notable issue is data quality, with 2,219 instances of duplicate entries. Additionally, there are three columns with missing values.

Another aspect of data structure is its organisation which is a flat file.

# 3.1.2. Column Analysis

In-depth analysis of individual columns within the dataset is a necessary step in understanding the data.



***Illustration 5:*** *Analysis of Key Columns – Number of Unique Values*

Six of the 20 columns shown in Illustration 6 were chosen to extract relevant and understandable information. The columns include 'text', 'author\_id', 'labels\_sentiment\_0\_topic', 'source\_created\_at', 'labels\_topic\_0\_topic', and 'source' .

Text: This column contains the actual tweet content and is the primary data source for NLP analysis.

***Author\_id:*** Identifier of the author of the tweet and is important for understanding the demographics of the dataset.

***Labels\_sentiment\_0\_topic and Labels\_topic\_0\_topic:*** These columns provide pre-categorised sentiment and topic labels.

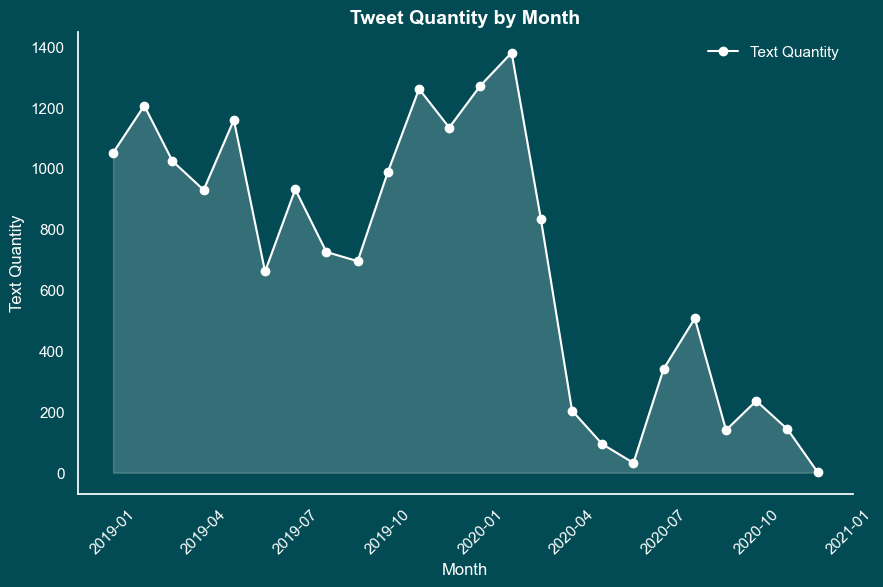
***Source\_created\_at:*** Timestamp of when the tweet was published. This temporal data is essential for trend analysis and understanding feedback timing.

***Source:*** The source of the tweet, whether it was sourced from Brandwatch or Sprinklr.

The text column, containing around 17,000 tweets, is particularly important for extracting meaningful patterns and themes from the public's conversations about Thameslink.

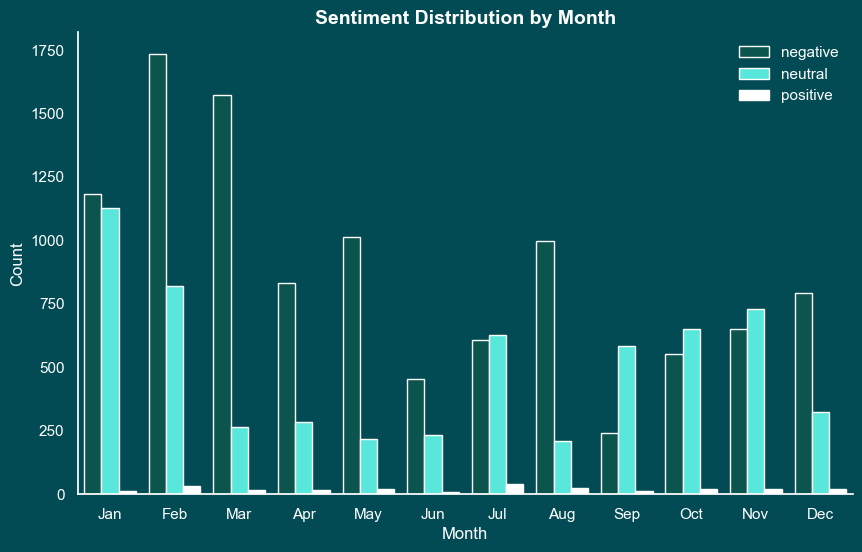
# 3.1.3. Time Analysis

The Thameslink tweet dataset covers the period from the beginning of 2019 to the end of 2020.



***Illustration 6:***  *Tweet Distribution Over Time – Tweet Quantity by Month*

This line Graph, called Illustration 7, describes the monthly frequency of tweets. This chart helps identify significant spikes or decreases in Tweet volume.



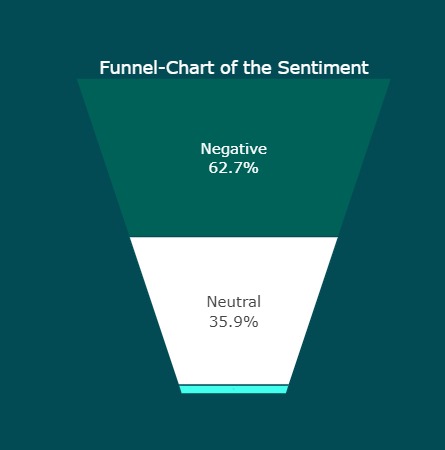
***Illustration 7:*** *Sentiment Analysis Over Time – Sentiment Analysis by Month*

By examining the sentiment expressed in tweets month by month, sentiment analysis over this two-year period can reveal how the public's perception of Thameslink services has changed.

An important observation from this analysis is the consistency of data collection across the twelve months of the year. This coverage provides a more accurate understanding of customer sentiment at different times of the year, minimising the potential for seasonal bias.

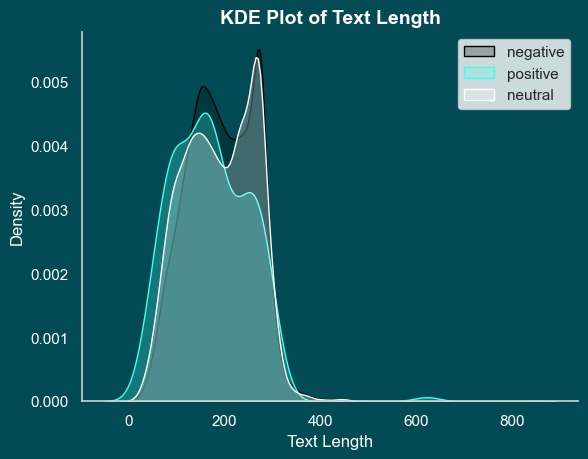
# 3.1.4. Sentiment Analysis

A key component of the dataset is the 'Sentiment' column, which divides tweets into three categories: negative, positive, or neutral. This classification is important for understanding the public's perception and response to Thameslink services.



***Illustration 8:***  *Sentiment Distribution – Funnel-Chart of the Sentiment*

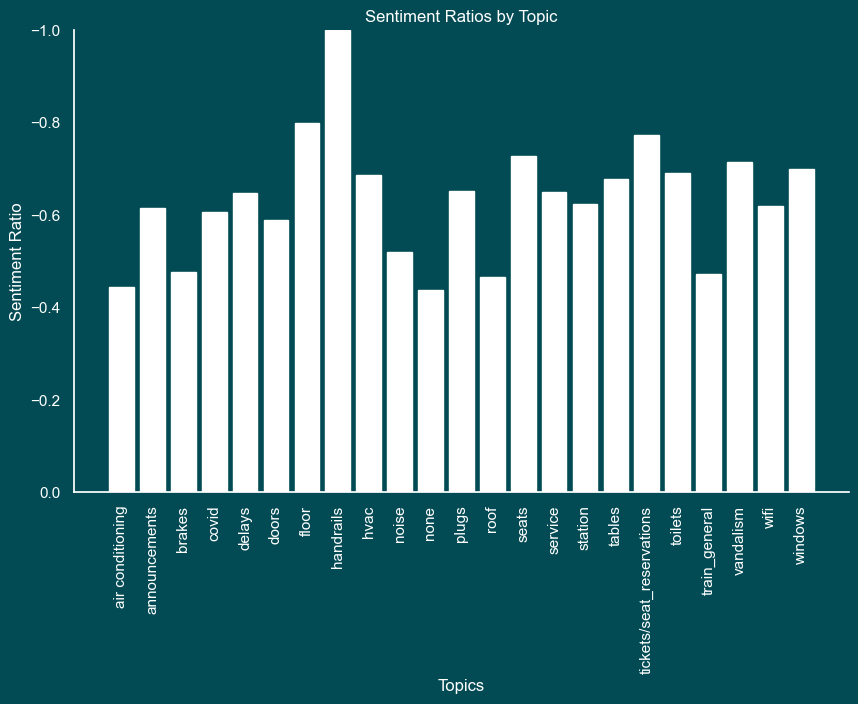
With this funnel chart, we gain insight into general sentiment trends among Thameslink customers by analysing the proportions of negative, positive and neutral sentiments in the dataset.



***Illustration 9:***  *Sentiment by Text Length – KDE Plot of Text Length*

A notable trend observed in the dataset is the correlation between negative sentiments and longer text posts. KDE Plot visualises this trend and highlights how negative emotions such as anger or frustration often show up in longer, detailed tweets.

The analysis of sentiments in relation to different topics mentioned in the tweets provides a more granular understanding of customer feedback. This approach enables the identification of specific areas of service that are sources of satisfaction or dissatisfaction among customers.

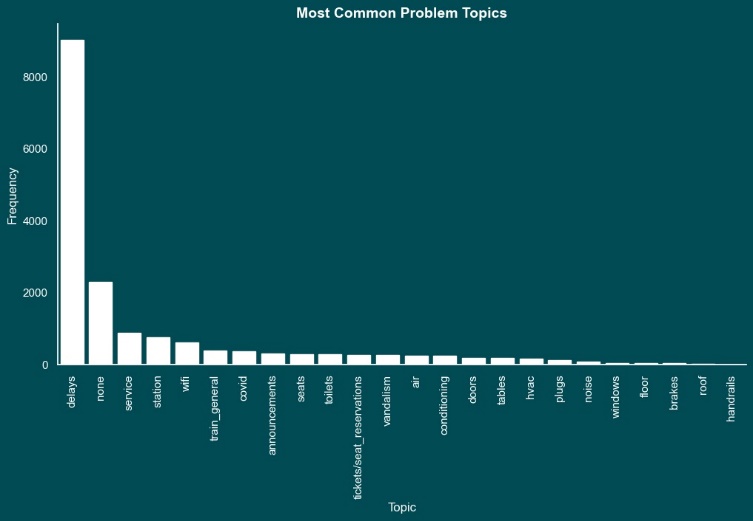


***Illustration 10:*** *Average Sentiment Across Topics – Sentiment Ratios by Topics*

This bar chart visualises how sentiment averages vary based on different topics discussed in tweets. The Illustration also indicates aspects of Thameslink services that cause positive or negative reactions.

# 3.1.5. Topic Classification and Analysis

An important aspect of the dataset is the inclusion of categorical annotations for tweet topics. This classification provides a more structured and focused analysis of customer feedback.

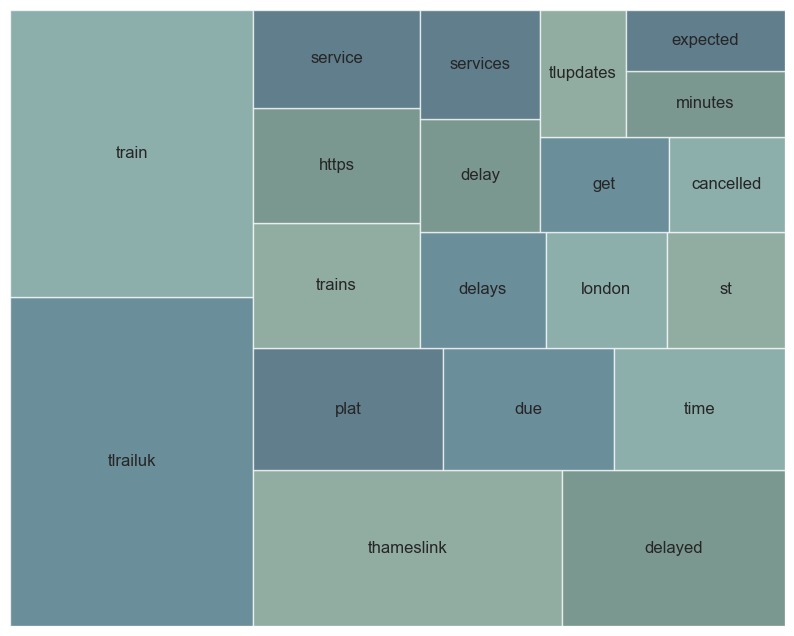
. 

***Illustration 11:*** *Common Topics in Tweets – Most Common Problem Topics*

This bar chart visualises the frequency of topics mentioned in the dataset. By identifying the most common topics we will not only be able to more systematically analyse Thameslink customers' tweets, but also understand the nuances of customer feedback.

# 3.1.6. Text Analysis: Word Frequency

A key component of understanding the textual content of the dataset involves analysing the frequency of word usage. This analysis helps in identifying the dominant terms and themes in the tweets, providing insights into the main issues or topics of interest for Thameslink customers.



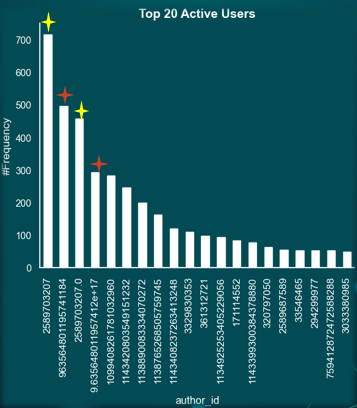
***Illustration 12:*** *Visualisation of Frequently Used Words*

The prominence of certain words reveals customers' primary concerns and topics of interest. For example, the chart features several variations of the term “delay,” including “delay,” “delays,” and “delayed,” demonstrating the importance of this issue in customer feedback.

# 3.1.7. Analysis of Author ID

A significant aspect of the dataset is the identification of the most active users based on the Author ID. Understanding the activity levels of different users is important for distinguishing between regular customer feedback and official communications from the company.

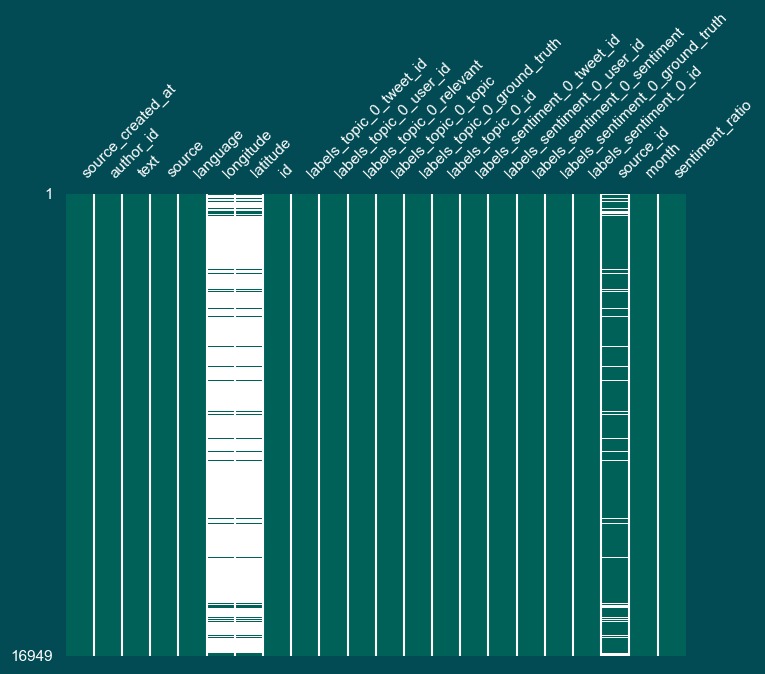
Illustration 13 provides a clear visualisation of the users who are most active in terms of tweeting about Thameslink. More importantly, it highlights those accounts that serve as official channels for Thameslink communications. Users marked with yellow stars typically use the hashtag #TLUpdates, whereas those marked with red stars frequently use #ThameslinkUpdate. These accounts help in segregating the dataset into tweets that are official communications from Thameslink and those that are customer feedback.



***Illustration 13:***  *Top 20 Active Users*

# 3.1.8. Missing Values Analysis

One of the challenges in data analysis is handling missing values. Identifying and addressing missing data is important for maintaining the quality and reliability of the analysis.



***Illustration 14:*** *Missing Values in Each Column*

Illustration 15 provides a visual representation of missing values across different columns, highlighting three columns in particular: Longitude, Latitude, and Source ID. The missing values are notably high in the Longitude and Latitude columns. Understanding the implications of these missing values is important for the subsequent stages of data preparation.

# 3.1.9. Duplicates and Outliers

Another crucial aspect of data preparation is the identification and handling of duplicates and outliers. These can significantly affect the conclusions drawn from the analysis if not addressed properly.

The dataset contains 2,219 duplicate entries and 37 retweets. These duplicates need to be addressed to prevent redundancy and potential biases in the analysis. Additionally, the box plot reveals 11 outliers in text length, with some texts having a length of over 800 characters. Outliers in text length can indicate unusual posting behaviour or data entry errors, and their handling is crucial for maintaining the dataset's consistency.

The identification of these duplicates and outliers is a key step in ensuring the accuracy and reliability of the analysis.

# 3.2. DATA PREPARATION

# 3.2.1. Launching Data Preparation

Data Preparation is a vital step in ensuring the quality and relevance of data for analysis. In the Tweet2GPT project, it is decided to use entire dataset for the analysis without restricting it to a specific time period or subset. This approach was chosen to ensure a comprehensive examination of all available data, providing a complete overview of customer feedback and sentiments across different timeframes and topics.

# 3.2.2. ETL Process Enhancements

The Extract, Transform, Load (ETL) process underwent several enhancements to improve the dataset's quality and relevance for analysis.

|  |  |  |
| --- | --- | --- |
| **Findings/Problems** | **Decision** | **Impact** |
| Unreadable Column Names | Shorten and Standardise | Improved Data Clarity |
| Data Types | Fixed Data Types | Enhanced Data Consistency |
| Redundant Time Information | Remove Second and Millisecond Information | Reduced Data Volume and Improved Data Quality |
| Singleton and Junk Columns | Removed 14 Columns | Simplified Data Structure and Reduced Data Redundancy |
| Duplicate Rows Based on Text Column | Removed 2,219 Rows | Enhanced Data Relevance for Maintenance Improvements |
| Tweets Sent by Thameslink Accounts | Removed 9 Authors | Enhanced Data Purity and Content Relevance |
| Delay Related Topics | Removed 7,645 Rows | Reduced Noise and Enhanced Data Relevance |

***Illustration 15:*** *Summary of ETL Process Enhancements*

***Unreadable Column Names:*** These column names are shortened and standardised to improve data clarity.

***Data Types:*** Corrections were made to ensure enhanced data consistency.

***Time Information:*** Seconds and millisecond information were removed to improve data quality.

***Singleton and Junk Columns:*** A total of 14 singleton or junk columns were removed to simplify the data structure.

***Duplicate Rows:*** 2,219 duplicated rows were removed to reduce redundancy and enhance data relevance.

***Tweets from Thameslink Accounts:*** As part of the project architecture, tweets from Thameslink accounts were removed to focus on customer-generated content.

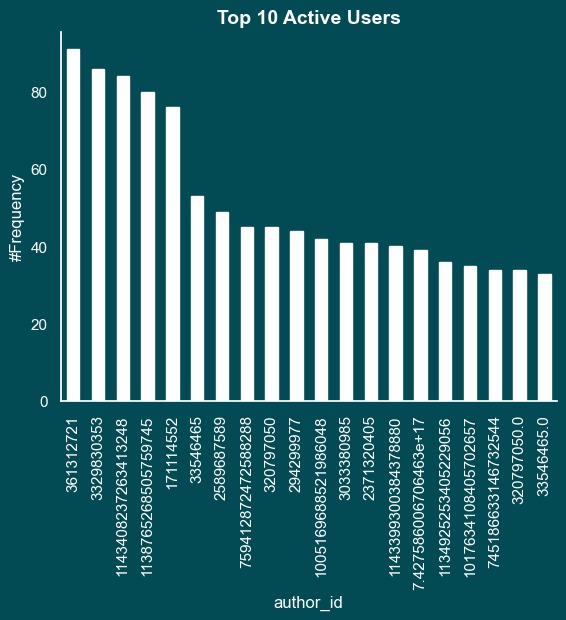
***Delay and None Topics:*** A total of 7,645 rows related to these topics were removed to reduce noise and enhance the relevance of data for maintenance improvements.

Each of these steps in the ETL process was critical to improve the quality and relevance of the dataset for the NLP analysis. By addressing issues such as unreadable column names, redundant data, and duplicates, the process ensured that the dataset was optimised for deriving meaningful and accurate insights.

# 3.2.3. Before and After ETL Comparison

An important aspect of data preparation is to visually demonstrate the impact of the ETL process on the dataset. This comparison provides an understanding of how data cleaning and transformation efforts have refined the dataset for analysis.

Refer to Illustration 14, as previously discussed in the "Analysis of Author ID" section, originally showed the top 20 active users, including Thameslink-related accounts.



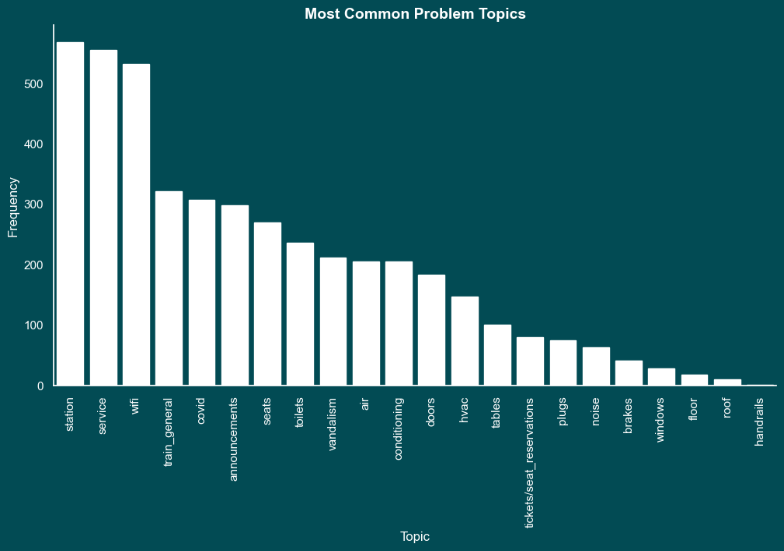
***Illustration 16:***  *Distribution of Top 20 Active Users (After ETL) – Top 10 Active Users*

Illustration 16 illustrates the dataset after the ETL process, specifically showing the impact of removing Thameslink-related accounts. By excluding these official accounts, the focus shifts entirely to customer-generated content, offering a more authentic view of public sentiment and feedback. This illustration demonstrates how the ETL process has refined the dataset, removing potential biases introduced by official communications and ensuring that the analysis is centered on genuine customer experiences and opinions.

# 3.2.4. Impact of ETL on Topic Distribution

The decision to remove topics related to latency in the ETL process significantly changed the topic distribution within the dataset. This change is important to focus the analysis on areas that can provide actionable information for care improvements.

Before the ETL process, the dataset included a significant number of tweets related to delays refer to Illustration 12, as previously discussed in the "Topic Classification and Analysis" section. While this is a critical area of concern for passengers, such tweets often do not offer the type of detailed feedback necessary for informing maintenance work.



***Illustration 17:*** *Most Common Problem Topics (After ETL)*

Illustration 17 shows how the removal of delay-related topics reshapes the focus of the dataset. By excluding these tweets, the dataset becomes more aligned with the project's objective of extracting actionable insights for maintenance improvements. This chart reveals a new distribution of issues, highlighting other areas of concern more directly related to maintenance issues.

Removing delay-related tweets is a strategic decision during the ETL process aimed at improving the suitability of the dataset for the project's goals. Analysis focusing on topics that provide more direct feedback on maintenance-related issues, provides more targeted and actionable information to improve Thameslink services.

# 3.3. An Overview of Data Understanding and Preparation

The stages of Data Understanding and Data Preparation in the Tweets2GPT project involved a series of methodical and strategic steps to ensure the dataset was primed for effective analysis.

# 3.3.1. Data Structure and Annotations

Initially, we described the data structure in detail, understanding the different types of data present and their organisation. Annotations, such as sentiment labels and topic categories, were carefully checked to ensure they were accurately reflected in the dataset. This step was crucial to set the stage for a more in-depth analysis.

# 3.3.2. Data Quantity, Distributions, and Representation

A thorough evaluation of data quantity, distributions, and representation was undertaken. This process involved assessing the volume of data, understanding how it was distributed across various categories, and ensuring that it was representative of the wider customer sentiment.

# 3.3.3. Comprehensive Dataset Usage

Our approach involved using the entire dataset for analysis without limiting it to specific timeframes or subsets. This decision was made to ensure a holistic view of customer feedback.

# 3.3.4. Handling Missing Values and Outliers

We identified and addressed missing values in the dataset, which is an important step in maintaining the integrity of the analysis. Additionally, outliers were identified and addressed. These actions were important to ensure that the data accurately reflected the reality of customer feedback without being distorted by anomalies.

# 3.3.5. Decision Clarification

Finally, we clarified the logic behind each decision made during the Data Preparation stage. This transparency is key in ensuring that the analysis is replicable and grounded in a solid understanding of the dataset's characteristics and limitations.

In conclusion, the Data Understanding and Data Preparation stages of the Tweet2GPT project involved a series of careful considerations and actions. From analysing the structure and content of the data to preparing it for analysis by addressing missing values, outliers, and making informed decisions, each step was crucial in ensuring the dataset was optimally prepared for meaningful and accurate analysis. These preparations are the basis on which the suggestions prepared for the maintenance team will be built.

# 4. MODELLING

This chapter delves into the models used and the general workflow which starts with the extraction of daily tweets using Python and the Twitter API and continues by checking the content of the messages to gauge customer sentiment and identify common complaints.

# 4.1 Pre-Processing

The first step in the process involves pre-processing the tweets to filter out irrelevant data. Tweets that come directly from the train service provider’s account, @TLRailUK, are excluded, as the team is interested in customer-generated content. @TLRailUK posts announcements, PR related tweets, schedule changes, none of which is relevant to the final product the end user needs

Additionally, tweets that mention delays are removed from the dataset because the maintenance team already has delay data from operational reports. The tweets about delays have subjective and often incorrect information does not lead to actionable feedback and service improvements.

# 4.2 Irony Detection

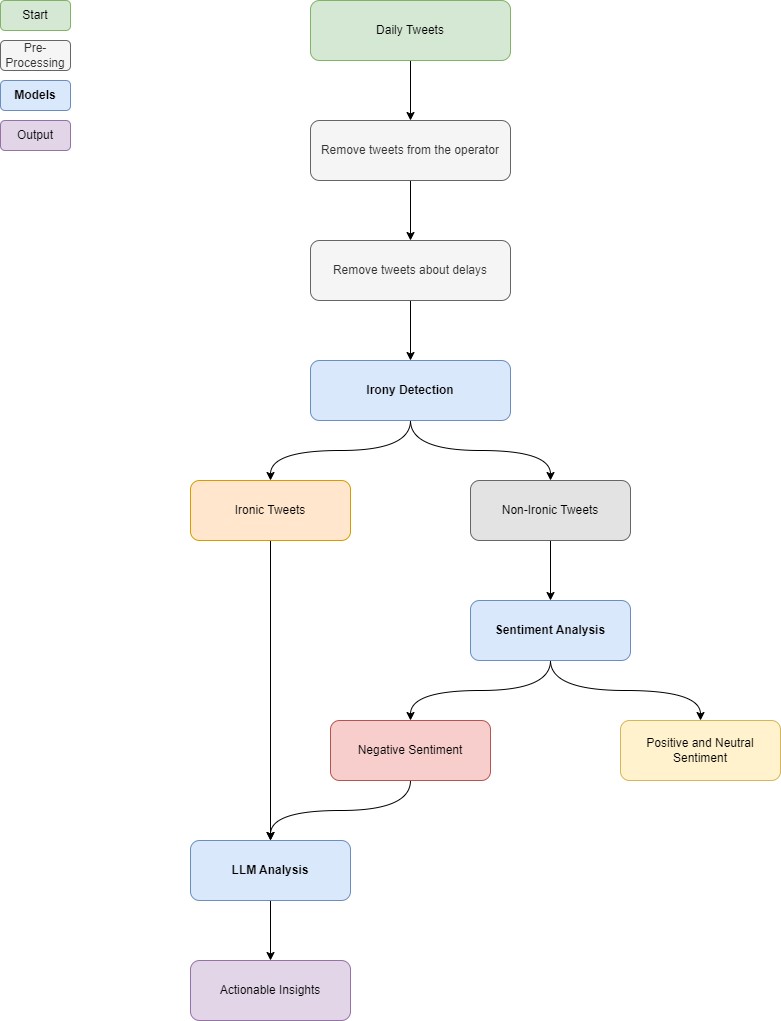
For irony detection, the team uses the TweetNLP library. This tool is crucial for identifying irony in tweets—where customers may express dissatisfaction through seemingly positive language, which could be misconstrued by standard sentiment analysis algorithms. By employing this specialized NLP tool, the team ensures they accurately capture the true sentiment behind customer feedback. Tweets marked with 'irony' are given special attention in the categorization process to ensure that the customer’s true sentiment is understood, avoiding the misinterpretation that can occur with literal analysis. The non-ironic tweets will go to the sentiment analysis model.

# 4.3 Sentiment Analysis

TweetNLP library is used also for the sentiment analysis. This library has been trained specifically on tweets and it has resulted to be quite good in determining the sentiment of the text. The positive and neutral tweets are removed while the negative tweets are kept to be merged with the ironic tweets and analysed further.

# 4.4 ChatGPT suggestions

The resulting filtered data, which includes the text, date, and irony flags, of the negative and ironic tweets, is formatted into a DataFrame 'day\_tweets'. This DataFrame is then used to query the ChatGPT API with a detailed prompt that instructs the model to categorize tweets into complaints and non-complaints and to further sub-categorize the complaints into specific topics like Train Conditions, Staff Conduct and Service, Station Facilities, and so on. This categorization allows the maintenance team to prioritize issues based on their frequency and impact on the customer experience. The results are made tabular and pushed to the next step, which is Dashboard.



***Illustration 18:*** *Model Diagram*

# 5. EVALUATION

This chapter focuses on evaluating the Tweets2GPT project, which is designed to improve Thameslink train maintenance through Natural Language Processing (NLP) Analysis. The chapter provides an evaluation of the methods used to evaluate the project's effectiveness.

# 5.1.1. Purpose of Evaluation

Evaluating the Tweets2GPT project is important to understand its impact on improving Thameslink train maintenance through NLP Analysis. During the evaluation phase, we aim to identify the strengths, weaknesses, and areas of improvement of the project.

# 5.1.2. The Importance of Evaluation

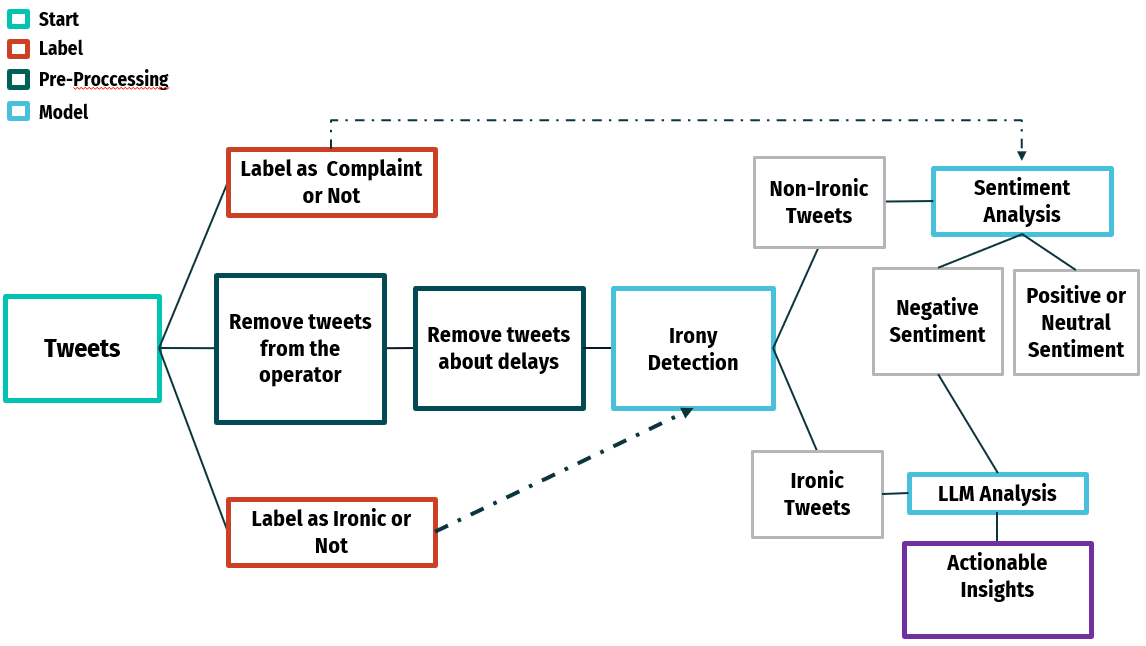
Evaluation acts as a compass and draws a road map for The Fine Tuners team to identify and eliminate shortcomings. By both filling gaps and ensuring sustainability, evaluation helps the project achieve its goals by shaping next steps.

# 5.1.3. Evaluation Process

A step-by-step approach was followed during the evaluation phase. he evaluation phase began by examining irony, and then moved on to Sentiment Analysis and LLM Analysis. Finally, the evaluation was concluded by controlling how the project affected the company's Key Performance Indicators (KPIs). This sequence helped make the impact of the project manageable.

# 5.1.4. The Process of Modelling and Evaluation

The diagram below shows how both the model and the evaluation process work collaboratively. The visual presentation highlights key steps in the methodology and outlines the step-by-step workflow. This visual gives an idea of the functioning of the model and subsequent evaluation procedures.



***Illustration 19:*** *Process of Modelling and Evaluation*

The evaluation started with two separate surveys dedicated to the analysis of Irony and Sentiment. All team members responded to surveys categorized as Ironic or Non-Ironic and Complaint or Non-Complaint. The answered questions consisted of 100 randomly selected tweets. The results of these surveys were set aside for further evaluation steps.

The process was simplified by initially filtering tweets to select only those relevant to the maintenance team. The system eliminated tweets from Thameslink and those including delays, focusing instead on tweets that pointed out specific issues. Following this, irony detection was applied to these tweets. The model then identified ironic tweets and directed them to LLM Analysis. The non-ironic tweets underwent Sentiment Analysis; those classified as negative were also sent to LLM Analysis. Positive and neutral tweets, deemed irrelevant for the maintenance team, were disregarded. Tweets that exhibited both irony and negative sentiments were particularly valuable during the LLM analysis, providing significant insights for the maintenance team's consideration.

For the Irony and Sentiment Evaluation, a similar filtering process was applied to the 100 random tweets as done in the model. The results were then compared to evaluate the accuracy of the model’s classifications against the team's assessments from the surveys. In the LLM Evaluation, the focus was on assessing the consistency of the LLM Analysis by comparing its outputs for identical inputs. A more detailed explanation of these processes will be presented in the following sections of the report.

# 5.2. Evaluation of Irony

# 5.2.1. Defining Irony

Irony, as per the Oxford Dictionary, is the use of words that convey the opposite of their literal meaning, often in a humorous or sarcastic manner.

# 5.2.2. Significance of Identifying Irony

Understanding and identifying ironic tweets is crucial for:

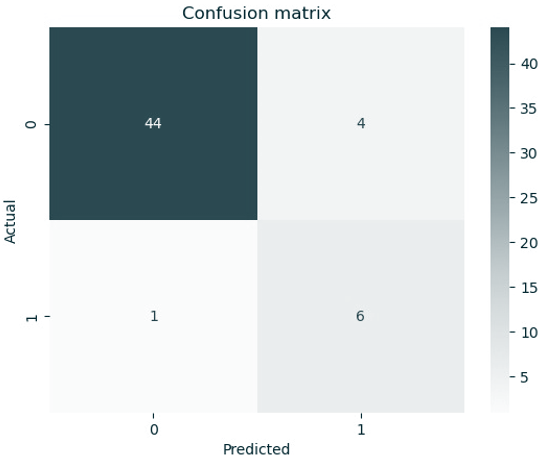
* ***Accurate Sentiment Analysis:*** It is crucial for understanding the true sentiment in passenger messages, avoiding oversight of complaints or problems.
* ***Preventing Misinterpretation:*** It prevents misinterpretation during the analysis of passenger opinions, contributing to a clearer understanding and more effective responses.
* ***Enabling Effective Responses:*** It enables a clear understanding for effective responses, ensuring that actions are based on an accurate interpretation of the underlying sentiment.

# 5.2.3. Importance of Irony Detection

The use of 'irony detection' in the process is crucial for mitigating misleading analysis. The primary purpose is to mark ironic tweets and alert the LLM Analysis at the end of the pipeline. This precaution ensures that ironic tweets do not lead to misleading conclusions during the analysis.

# 5.2.4. Evaluation of Irony Detection

To assess how well the "irony detection" works, a two-step method is used. Firstly, the outcomes of the model are compared with insights from a survey where all group members give their opinions, bringing different viewpoints to the table. Then, the survey results are treated as the Ground Truth, setting a standard to measure how accurate the model is. The Ground Truth is then compared with the model's results using a confusion matrix, assisting in the thorough evaluation of the 'irony detection' process.



***Illustration 20:*** *Irony Confusion Matrix*

To evaluate the accuracy of "irony detection," a confusion matrix is utilized. This tool provides insight into the model's ability to predict irony. The matrix is divided into four sections:

***True Positive (6/7):*** This section shows where the model accurately identified 6 out of 7 ironic tweets, in line with the survey results. This indicates the model's effectiveness in recognizing irony.

***False Positives (4):*** Representing instances where the model incorrectly identified non-ironic tweets as ironic, this section comprises 4 instances. However, this is not a significant concern, as these are likely to be corrected by subsequent analysis by the LLM.

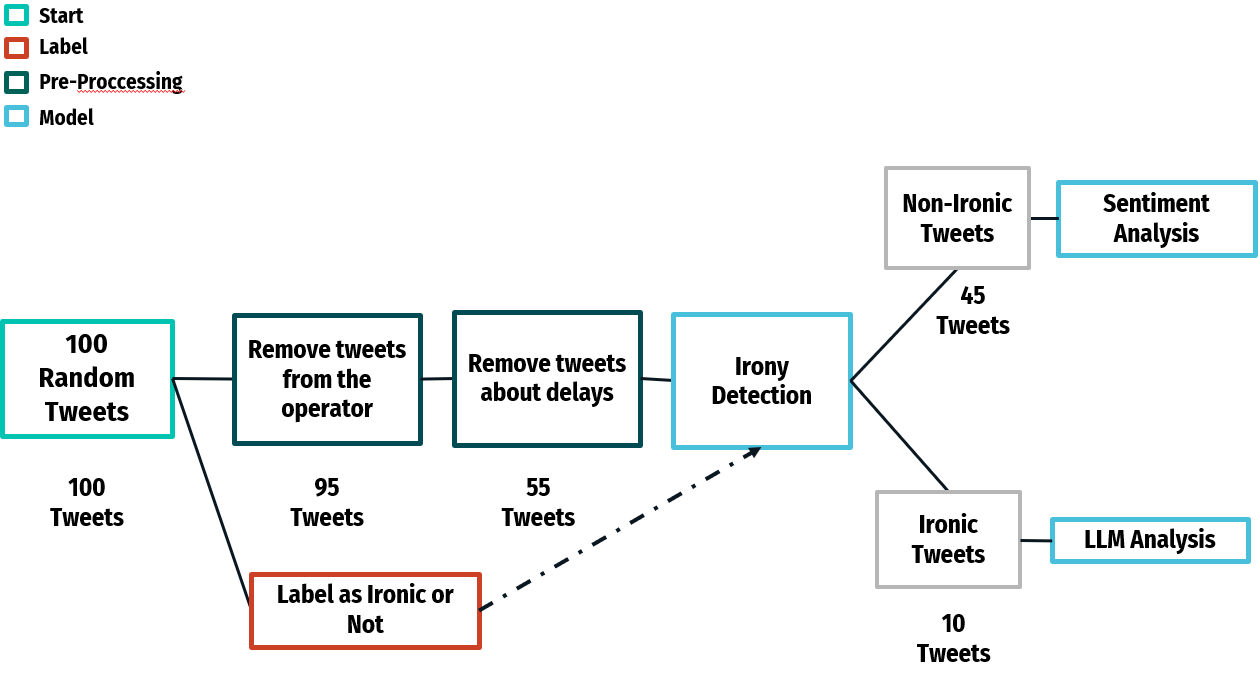
***True Negative (44):*** The True Negative section highlights instances (44) where the model correctly identified non-ironic tweets. This signifies a robust performance in discerning non-ironic content.

***False Negatives (1/7):*** In this section, the model missed identifying 1 ironic tweet out of 7. While a small error, it provides valuable insights for potential improvements in the detection process.

This breakdown within the confusion matrix offers a comprehensive understanding of the model's strengths and areas for enhancement in "irony detection".

# 

# 5.2.5. Process of Irony Evaluation



***Illustration 21:*** *Process of Irony Evaluation*

For the irony evaluation, 100 random tweets were selected. From these, 5 tweets related to Thameslink operators and 40 concerning delays were excluded, leaving 55 relevant tweets. These tweets were then put through the irony detection process. The outcomes of this process were compared with insights from a survey, which served as the ground truth for the evaluation.

# 5.2.6. Evaluation Scores of Irony

In the summary of the irony evaluation, 5 evaluation scores were derived, with a particular emphasis on the key metric of Recall. Recall measures the ratio of true positives to the total of true positives and false negatives, assessing the model's capacity to identify relevant instances in the dataset. A high recall score, such as the achieved 85.7%, indicates the model's effectiveness in minimizing false negatives and accurately identifying most positive instances. In this case, with a focus on True Positives, the recall percentage suggests a low likelihood of the model missing ironic tweets.

***Precision (60%)****:* The model is correct in predicting positive instances 60% of the time.

***Recall (85.7%)****:* The model captures approximately 85.7% of all actual positive instances.

***Specificity (91.7%)****:* The model accurately predicts negative instances about 91.7% of the time.

***Accuracy (90.9%)****:* The model's predictions are correct around 90.9% of the time.

***F1 score (70.6%)****:* Provides a balanced measure of both precision and recall, giving an overall indication of the model's performance.

# 5.3. Evaluation of Sentiment Analysis

# 5.3.1. Defining Negative Sentiment

Negative sentiment refers to a generally pessimistic, critical, and dissatisfied viewpoint.

**5.3.2. Significance of Identifying Negative Sentiment**

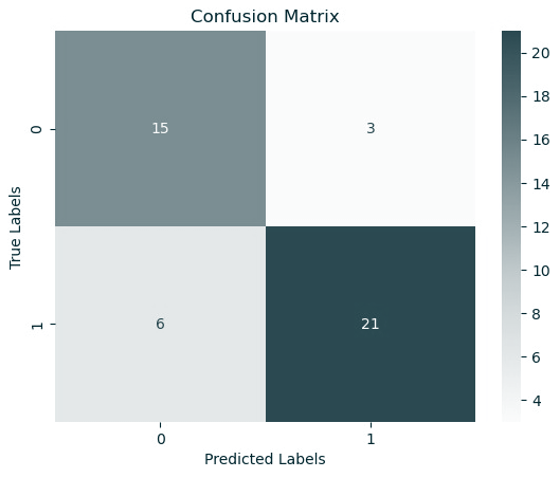
Understanding and identifying negative sentiment tweets is crucial for:

* ***Enables Early Problem Detection:*** It is crucial for quickly identifying potential issues. Early detection can lead to prompt intervention, which helps prevent small concerns from becoming larger problems.
* ***Increases Customer Satisfaction:*** It is essential in showing customers that their feedback is valued. By addressing negative comments effectively, it demonstrates a commitment to customer care, which can significantly enhance customer satisfaction.
* ***Maintains a Positive Situation:*** It is important for preventing negative experiences from worsening.

# 5.3.3. Evaluation of Sentiment Analysis

A two-stage method such as Irony Evaluation is used to evaluate how well "Sentiment Analysis Evaluation" works. The only difference is that the survey questions are Complaint & Non-Complaint instead of Ironic & Non-Ironic.

To evaluate the accuracy of "Sentiment Analysis" a confusion matrix is utilized. This tool provides insight into the model's ability to predict negative sentiment. The matrix is divided into four sections as shown in the figure below and explained:



***Illustration 22:*** *Confusion Matrix of Sentiment Analysis*

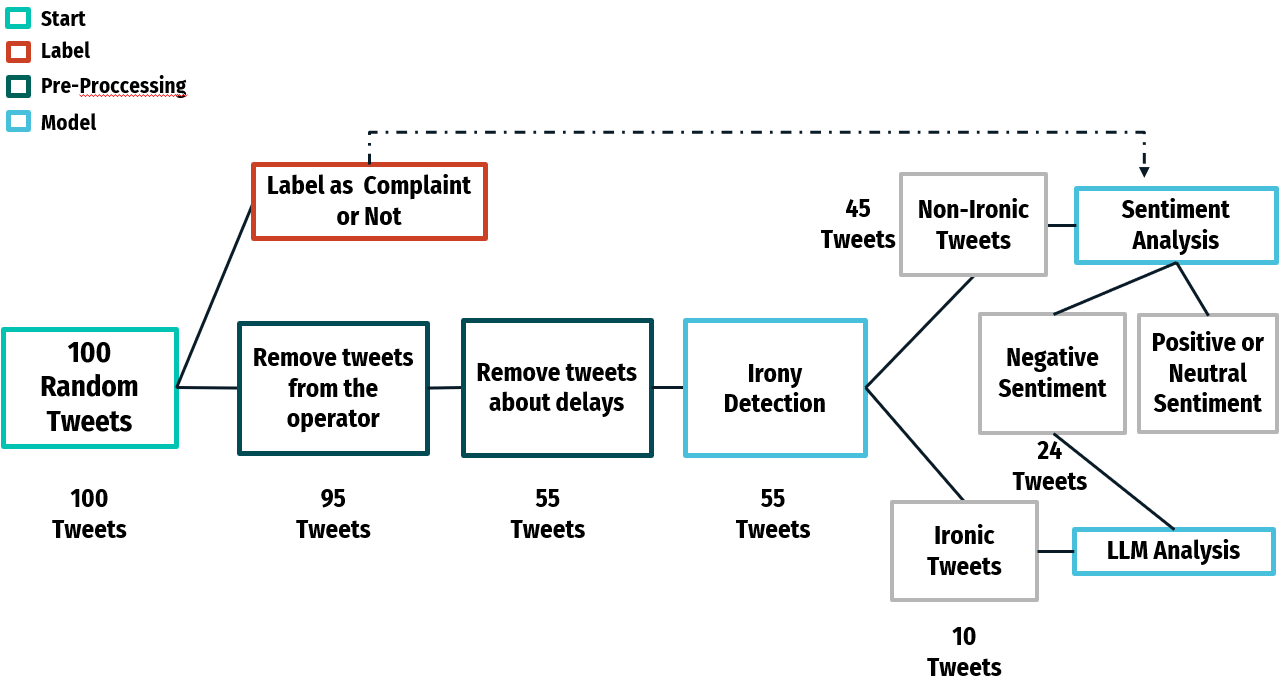
***True Positive (21/30):*** This part indicates the instances where the model accurately identified 21 out of 30 tweets as having negative sentiment, aligning with the survey findings. It highlights the model's capacity to correctly pinpoint negative sentiments.

***False Positives (3):*** In this category, there are 3 cases where the model incorrectly labeled tweets as negative. While these are errors, they are generally not a major concern, as subsequent LLM analysis is likely to reclassify them as non-complaints.

***True Negative (15):*** The True Negative section shows 15 instances where the model rightly identified tweets as positive or neutral. This reflects the model's strength in discerning tweets that do not carry negative sentiment.

***False Negatives (6):*** The model missed 6 instances of negative sentiment. These omissions, while not numerous, are informative for potential enhancements to the model's detection capabilities.

# 5.3.4. Process of Sentiment Analysis



***Illustration 23:*** *Process of Sentiment Analysis*

In the evaluation process, 100 random tweets were initially gathered. Out of these, 5 tweets concerning Thameslink operators and 40 regarding delays were excluded, resulting in a total of 55 tweets deemed relevant for analysis. These selected tweets were then processed through irony detection. The output of this stage was 45 tweets that were classified as non-ironic and were subsequently analyzed for sentiment, being evaluated as negative, positive, or neutral. Tweets identified with negative sentiment were then directed into LLM Analysis.

# 5.3.5. Evaluation Scores of Sentiment Analysis

In the sentiment analysis evaluation, several scores were calculated, with each offering insights into the model's performance, particularly in detecting negative sentiment.

***Precision (71.4%)****:* The model demonstrates reliability in predicting positive sentiment, with 71.4% of its positive predictions being accurate.

***Recall (83.3%)****:* The model successfully identified 83.3% of the actual negative tweets, as indicated in the survey, though it missed some positive tweets.

***Specificity (77.8%)****:* The model's ability to recognize non-negative sentiments is reasonably good, with an accuracy of 77.8% in these predictions.

***Accuracy (80.0%)****:* Overall, the model demonstrates an accuracy of 80.0%, taking into account both positive and negative predictions.

***F1 Score (76.9%)****:* The F1 Score, which balances precision and recall, stands at 76.9%, indicating a solid overall performance of the model.

The results of the Sentiment Analysis Evaluation offer valuable insights. The Recall score, at 83.3%, indicates the model's efficiency in identifying negative tweets, based on the survey. This high percentage demonstrates the model's effectiveness in aligning with the ground truth in detecting negative sentiments.

# 5.4. Evaluation of LLM Analysis

# 5.4.1. Purpose of LLM Analysis

LLM Analysis is employed to thoroughly analyze passenger tweets, with a focus on efficiently categorizing and resolving train maintenance issues. A key part of this process involves interpreting irony within the tweets to ensure an accurate assessment of sentiment. This approach is vital for understanding the true nature of passenger feedback and addressing maintenance-related concerns.

# 5.4.2. Evaluation of LLM Analysis

The evaluation of LLM Analysis centers on testing its reliability. This is done by checking if the model provides consistent outputs when presented with identical inputs. Consistency in response is crucial as it indicates the model's stability and reliability in processing and interpreting information. By ensuring that similar inputs yield similar results, the reliability of the LLM Analysis in handling and categorizing tweets for maintenance issues can be confidently assessed.

# 5.4.3. LLM Analysis Result Evaluation

The evaluation indicates that LLM Analysis functions effectively, as demonstrated by:

***Consistent Formatting Results:*** The analysis consistently delivers well-formatted outputs, ensuring reliable data interpretation.

***Effective Topic Categorization:*** It adeptly categorizes topics, which is crucial for providing meaningful suggestions to the maintenance team. This categorization allows for the efficient distribution of tasks among relevant departments.

***Detailed Maintenance Suggestions:*** Detailed and practical suggestions are provided, tailored to the needs of the maintenance teams, facilitating informed decision-making.

***Handling Irony:*** The ability of the analysis to interpret and manage irony in tweets is a noteworthy feature, contributing to accurate sentiment analysis.

# 5.4.4. Necessary Future Actions

Looking ahead, certain actions are identified as essential for the continued success and improvement of the LLM Analysis:

***User Feedback Gathering:*** Collecting feedback from users is critical for evaluating the real-world impact and effectiveness of the project outputs.

***Continuous Improvement:*** There is a need for ongoing refinement of the model, incorporating new datasets, feedback, and corrections. This approach is vital to ensure the stability and enhancement of the model's capabilities.

# 5.5. Evaluation of KPIs

We examined KPIs in two sections: those directly affected and those indirectly affected. Operational Efficiency, a directly affected KPI, is used to measure the success of the model. We also monitor other KPIs to assess if there are improvement.

***Operational Efficiency:*** This KPI measures how the project streamlines the maintenance process. The resulting effect observed is faster problem resolution in trains, leading to an increase in overall operational efficiency.

***Cost of Maintenance:*** By facilitating efficient issue resolution, the project helps in cost-saving. This efficiency translates into the company benefiting from reduced maintenance expenses.

***Equipment Reliability:*** The project's swift problem-solving approach enhances the reliability of train equipment. As a result, trains become more reliable, providing safer transportation services.

***Safety Performance:*** Timely maintenance ensures optimal safety. The resulting effect is a decrease in operational risks and an improvement in safety performance.

***Downtime Reduction:*** The project's quick identification and resolution of issues positively affect train downtime. Consequently, train downtime is reduced, leading to increased customer satisfaction.

***Customer Satisfaction:*** Improved service quality and prompt issue resolution indirectly influence customer satisfaction. This leads to passengers experiencing better service, which in turn results in higher customer satisfaction.

In summary, the Tweets2GPT project has been effective in detecting irony and negative sentiment within passenger tweets, contributing significantly to the maintenance process of Thameslink trains. The LLM analysis plays a pivotal role in this process, assisting the maintenance team by organizing similar tweets and offering detailed recommendations for action.

***Effectiveness in Detection:*** The project has demonstrated its capability in accurately identifying irony and negative sentiment in tweets, which is crucial for understanding and addressing passenger concerns related to train maintenance.

***Role of LLM Analysis:*** LLM analysis aided the maintenance team by categorizing tweets with similar content and providing comprehensive suggestions, thus it streamlined the maintenance workflow.

***Need for Continuous Improvement:*** To maintain and enhance the success of the project, there is a critical need for ongoing user feedback and continuous improvement. Incorporating new data and adapting to feedback are essential steps for the project's sustained effectiveness and relevance.

# 6. DEPLOYMENT

# 6.1.1. What is deployment?

Deployment is the process of making a software application or system available for use in a production environment. It involves activities such as code compilation, configuration management, and testing to ensure the application operates as intended. Deployable artifacts are released to the target environment, and infrastructure may be provisioned as needed. Post-deployment verification and monitoring are essential to ensure the application's correct functionality, and rollback plans are prepared for contingency.

# 6.1.2. Objective of our project deployment

Orchestrating an end-to-end pipeline for the development and deployment of a sentiment detection model involves creating a seamless and automated workflow that spans the entire lifecycle of the model. The designed solution aims to streamline the monitoring and assessment of maintenance activities within Thameslink to quickly identify areas of concern thereby leading to prompt response and continuous improvement while being scalable, secure and highly available for data storage.

# 6.1.3. Pre-requisites and Interfaces

Before proceeding with the deployment, the system must be configured with the pre-requisites mentioned below:

* ***Twitter account:*** Required to connect to Twitter Developer account and to generate authentication keys and tokens to connect to Twitter API.
* ***Azure account:*** To integrate with VSCode for building, deploying and managing the entire orchestration on cloud premises. Create an Azure account for students to get the Azure Student Subscription, which comes with a free 100$ credit to use the necessary Azure resources.
* ***VSCode:*** Required to build, integrate and deploy the Python codes.

The following interfaces have been identified for the end-to-end deployment process:

* The first interface is between Twitter (X) and VSCode , to pull the tweets with specific keywords by connecting to Twitter API , using Python as the underlying programming language.
* The second interface is between VSCode and Azure Cloud Ecosystem (specifically Azure serverless function) to build and deploy the python code in VSCode to Azure.
* The final interface is between Azure Cloud (specifically Azure SQL Database) and Power BI (for data visualization).

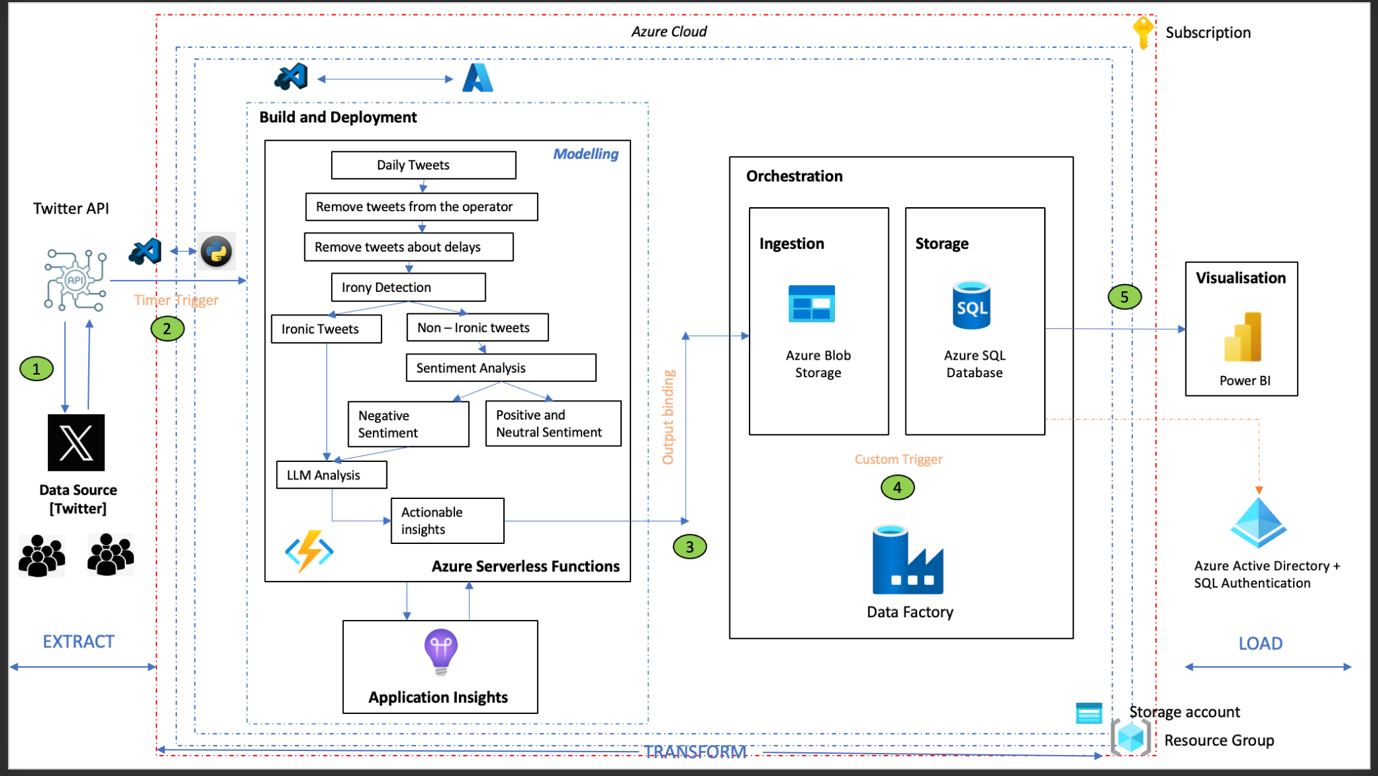


1 2 3

***Illustration 24:*** *Identified Interfaces*

# 6.2. Deployment Architecture

In the deployment architecture, we embark on a journey that begins with Twitter data acquisition through the Twitter API. Visual Studio Code (VS Code) serves as our development and integration hub, orchestrating the entire process. Leveraging Azure's serverless function capabilities, we deploy sentiment analysis functions with output bindings, seamlessly storing the analyzed data into an Azure Blob Storage. The synchronized data is then efficiently processed and orchestrated into Azure SQL Database using Azure Data Factory, forming a vital link in our data pipeline. Finally, the enriched data is visualized in Power BI, providing insightful analytics to empower the Thameslink maintenance team with real-time sentiment insights derived from the tweets. This architecture combines the flexibility of serverless computing, the power of Azure services, and the visual storytelling capabilities of Power BI for a comprehensive and impactful sentiment analysis solution.



***Illustration 25:*** *End-to-end deployment architecture*

# 6.2.1. Extraction Phase

***Step 1: Accessing Twitter Developer Platform***

To initiate the process, navigate to the Twitter Developer website and log in using the Twitter account credentials. Once logged in, proceed to create a new project on the platform and add an application to it. This step also involves generating essential authentication keys and tokens required to connect to the Twitter API programmatically. Specifically, we will need the API key, API secret key, Access token, and Access token secret.

***Step 2: Connecting to Twitter API with Tweepy and generated API credentials***

With the obtained API credentials, the next step involves connecting to the Twitter API using the Tweepy library in Python. Tweepy is a widely used Python library for interacting with the Twitter API, offering convenient functionalities for accessing and extracting tweets. Tweepy acts as a wrapper around the Twitter API, providing a convenient interface for sending requests and receiving responses.

***Step 3: Tweet Extraction and Storage***

Transitioning to the extraction phase, our primary goal is to retrieve tweets that specifically contain predefined keywords. This task is efficiently tackled by leveraging the robust search functionalities offered by the Twitter API through the Tweepy library. The extracted tweets (in JSON format) are flattened before storing them in a Dataframe. To enhance the organization and maintainability of our codebase, we decided to adopt a modular code strategy by organizing our code into main modules and corresponding submodules and the submodules being called by the main module during execution.

# 6.2.2. Transformation Phase

***Transformation Phase 1 [Build and Deploy]***

The first phase of transformation involves integrating the python code (end-to-end model) present in VSCode with Azure.

***Step 1: Creating a resource group and storage account***

In Azure, a Resource Group is a logical container for resources deployed in a region. It helps you manage and organize related Azure resources, such as virtual machines, storage accounts, and databases, as a single administrative unit. Resource groups enable you to manage and monitor resources collectively, apply policies, and control access and permissions.

A Storage Account, on the other hand, is a fundamental Azure resource that provides scalable and secure cloud-based storage. It supports various types of data storage services, including blobs (for unstructured data), tables (for NoSQL data), queues (for message communication), and files (for file storage).

***Step 2: Creating an Azure serverless function app***

After creating a resource group and storage account, the next step is to create an Azure serverless function app. Azure Functions allow us to run event-triggered code without managing the underlying infrastructure. They automatically scale based on demand and are an excellent choice for serverless computing scenarios. VSCode can be integrated with Azure functions by installing the necessary Azure extensions. After writing/ modifying function code in VSCode, we can use the debugging features in VS Code to test your function locally. Once satisfied, deploy the function to Azure using the Azure Functions extension.

***Step 3: Creating a timer trigger***

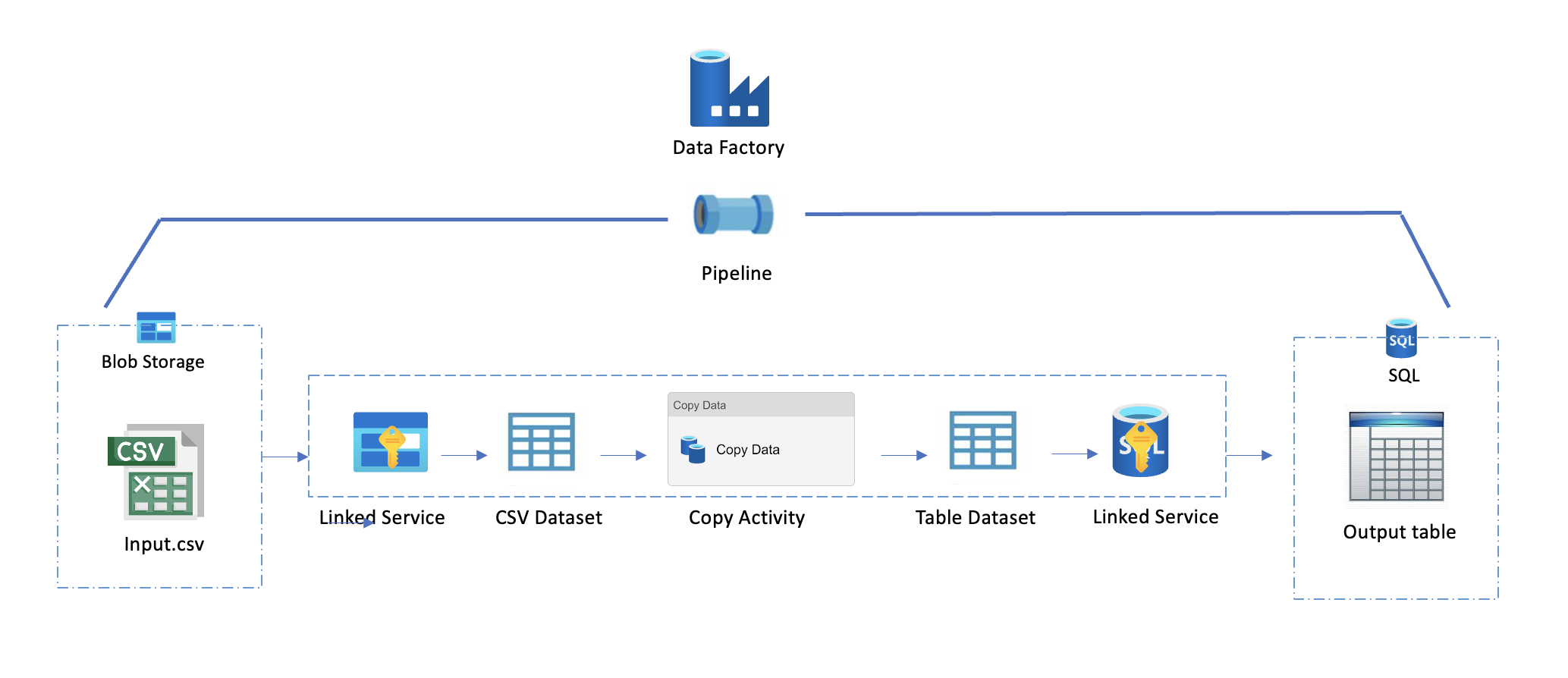
Once function app is created, create a new function in Azure portal with timer trigger. A timer trigger is a type of trigger that helps to run a function on a specified schedule , similar to a cron job. Eg : "0 0 0 \* \* \*" [ runs everyday at midnight ]. In our scenario the trigger was scheduled to run once per day to accumulate all the tweets, run the model on the data collected and retrieve suggestions and recommendations to improve the services of Thameslink.

***Step 4: Creating an output binding storage***

An output binding storage defines the target destination or resource in a declarative manner within the function's configuration function.json file. Here the output binding is Azure Blob Storage which indicates output of Azure functions will be stored in a blob container to then move it to Azure SQL database in the final transformation phase.

***Transformation Phase II [Orchestration]***

Orchestration involves defining and managing pipelines and activities within the data integration and transformation processes. Here we have used, Azure Data Factory to copy the data from Azure Blob Storage (source) to Azure SQL Database (sink) using copy as the data movement activity with custom trigger.



***Illustration 26:*** *Azure Data Factory Internal pipeline*

The following steps were undertaken to perform the orchestration after creating an Azure Data Factory instance and connecting it to the Azure storage account:

***Step 1: Define source and sink***

Azure Blob Storage acts as the source, holding a versatile range of structured, semi-structured, and unstructured data, specifically in the form of CSV files. The destination, or sink, is an Azure SQL Database designed to handle relational data. The orchestrated flow involves extracting, transforming, and loading (ETL) data from the diverse CSV files stored in Azure Blob Storage into a structured format suitable for relational storage in Azure SQL Database.

***Step 2: Create linked services for source and sink***

Linked services store the connection information and credentials needed to connect to the data sources. For the source, these linked services encapsulate the information required to establish a connection with Azure Blob Storage, where the CSV files reside. Meanwhile, the linked service for the sink encapsulates the connection details for the Azure SQL Database, the relational storage destination.

***Step 3: Create datasets for source and sink***

Datasets define structure of the data and to also indicate how it should be consumed by activities in the pipeline.

***Step 4: Pipeline creation in Azure Data Factory***

Creating a pipeline and adding a "Copy Data Activity" is a fundamental step in orchestrating data movement between different data sources. Once pipeline is created and the activity is added, configure the copy data activity by configuring the source and sink datasets to point to source and sink providing necessary details. Map the source and sink schemas correctly. Also, add a custom trigger to facilitate movement of data from Blob Storage to SQL Database.

***Step 5: Execute the pipeline***

Run the pipeline to execute the data copy operation. Monitor the progress, success, or any errors in the Azure Data Factory Monitoring interface.

# 6.2.3. Loading Phase

The interface for the loading phase involves utilizing an Azure SQL database as the final storage repository, providing a secure and scalable solution for data storage. To enhance data analysis and presentation, Power BI is employed as the visualization tool for creating interactive and insightful dashboards. This interface architecture ensures a seamless flow of data from storage to visualization, empowering users to derive meaningful insights and make informed decisions based on the stored information in the Azure SQL database.

To establish connection between the interfaces, we require the below from Azure:

* Azure Server name (eg : <server\_name>.database.windows.net)
* Database name , username and password (required if using SQL server authentication).
* Update Azure server firewall settings to include the IP address of the machine trying to connect to the SQL database.

# 6.3. Challenges

* The major challenge encountered was an ODBC Driver compatibility issue thereby failing to establish a direct connection between the Azure Function App and SQL database. Leveraging the capabilities of Azure Data Factory proved to be a strategic resolution. By orchestrating data movement between the source and sink using custom triggers, compatibility challenges were bypassed, ensuring seamless and reliable data transfer. This not only resolved the immediate connection issue but also introduced a more robust and scalable solution for managing data flow within the Azure ecosystem.
* The Tweetnlp library (a custom library), designed for irony and sentiment analysis, was facing recognition issues within the Azure environment, which led to a build failure. The library was checked with the Azure Function version and Python version and no incompatibilities found. The possible option to resolve the conflict was to import the source code of the library into Azure environment, which was quite challenging. As an easy fix, we decided to demonstrate the end-to-end data pipeline and orchestration using the output csv file obtained from the model.

# 6.4. Best Practices for Sustained High Performance of Deployed Models

* ***Continuous Monitoring and Resource Scaling:***

1. Use monitoring tools like Azure Monitor and Azure Application Insights to track the deployment process in real-time. Both Azure Monitor and Application Insights seamlessly integrate with various Azure services, making it easier to monitor the entire application stack.
2. Adjust the compute resources allocated to the deployed model dynamically based on changes in demand or data volume. Also define auto-scaling rules based on specific metrics, such as CPU usage or request rates, to trigger automatic adjustments to resource allocation.

By combining continuous monitoring with dynamic resource scaling, you can ensure that your deployed models are both performant and cost-effective, adapting to changing demands in real-time.

* ***Data Drift Monitoring:*** Data drift occurs when the statistical properties of the input data used for training a machine learning model change in the production environment. This shift in data distribution can impact model performance, leading to a degradation in predictions.

1. Azure Machine Learning Data Drift allows you to monitor the distribution of features in the incoming data and compare it with the distribution of the training data.
2. Integrate data drift monitoring with automated model retraining pipelines. When significant drift is detected, trigger a retraining process to update the model with the most recent data.

* ***Documentation:*** Maintain comprehensive documentation about the model, its deployment, and any updates or changes made over time.
* ***Feedback Loops:*** Establish feedback loops to collect information about model predictions and their real-world outcomes.
* ***Logging and Auditing:*** Ensure that the modular code has logging mechanisms to capture information for debugging and troubleshooting.
* ***Model Retraining:*** Schedule periodic retraining of the model using new and updated data to ensure model remains accurate and relevant to changes in data.

# 7. RESULTS

The Tweets2GPT project's dataset was analysed. Pre-processing was carried out to exclude tweets from the service provider’s account and tweets about delays, focusing on customer-generated content and actionable feedback. The TweetNLP library was then employed for irony detection and sentiment analysis. This library helps in detecting irony in tweets and performing sentiment analysis, ensuring that only negative and ironic tweets are retained for further analysis. These filtered tweets were then used to query the ChatGPT API, which categorizes the tweets into complaints and non-complaints, and further categorizes the complaints into specific topics. This categorization assists the maintenance team in prioritizing issues based on their frequency and impact on customer experience. The results were tabulated and used in the next step, which involves the Dashboard.

Prior to the deployment, an evaluation process was conducted to check the model performance. For the evaluation, 100 random tweets were selected, with tweets related to Thameslink operators and those concerning delays excluded, leaving 55 relevant tweets. These tweets underwent the irony detection and sentiment analysis process, and the outcomes were compared with insights from a survey, which served as the ground truth. The evaluation begins with two surveys on Irony and Sentiment, with team members categorizing tweets as Ironic or Non-Ironic and Complaint or Non-Complaint. The tweets are then filtered to select only those relevant to the maintenance team, excluding tweets from Thameslink and those mentioning delays. Irony detection is applied to these tweets, with ironic tweets directed to LLM Analysis. Non-ironic tweets undergo Sentiment Analysis, with negative ones also sent to LLM Analysis. Positive and neutral tweets are disregarded.

*For the irony evaluation :*

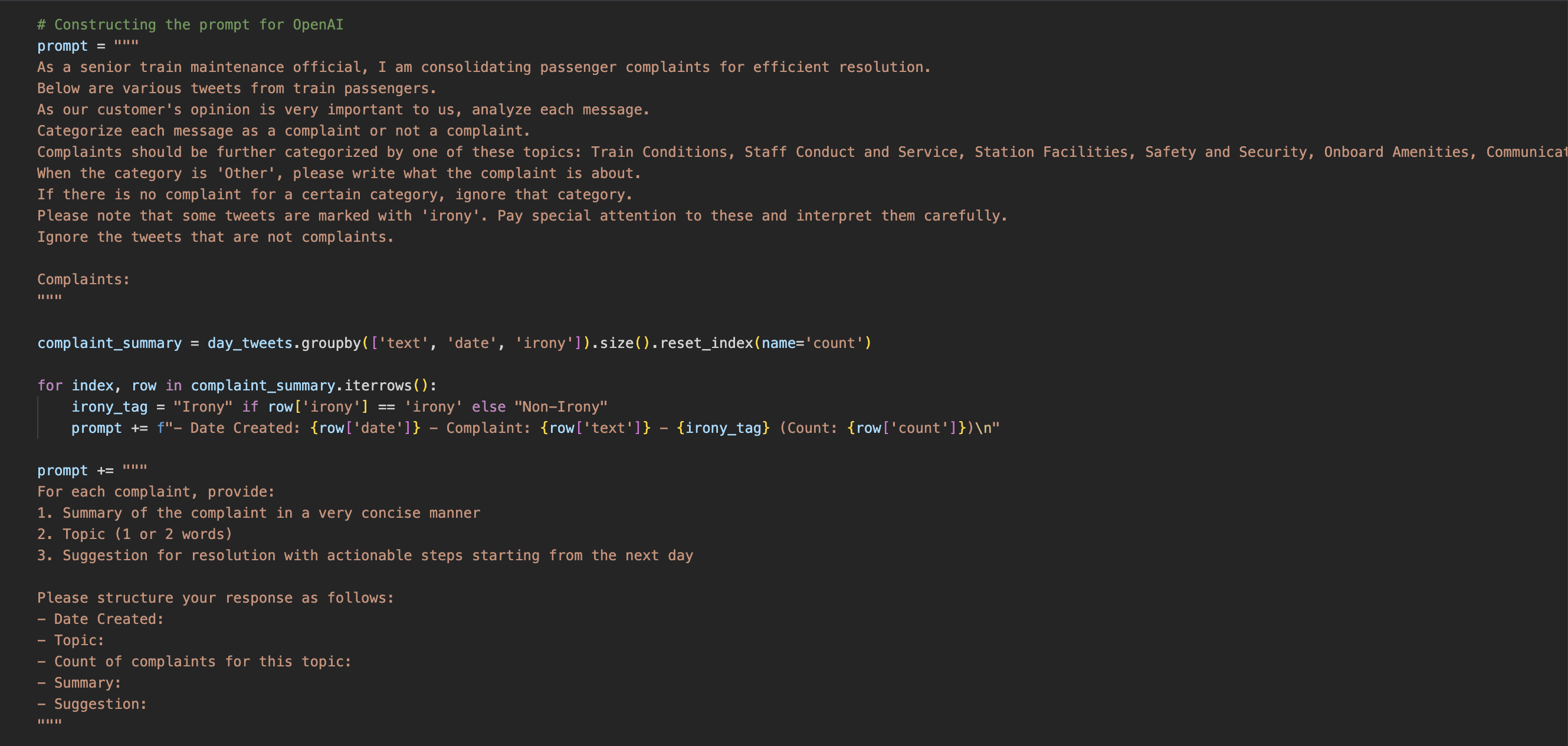
The ground truth (insights from survey) was then compared with the model’s results using a confusion matrix. The model accurately identified 6 out of 7 ironic tweets (True Positives) and correctly recognized 44 non-ironic tweets (True Negatives). However, it incorrectly labelled 4 non-ironic tweets as ironic (False Positives) and missed 1 out of 7 ironic tweets (False Negatives). Based on performance metrics,  it correctly predicted positive instances 60% of the time. It had a recall of 85.7%, capturing most of the actual positive instances. The model’s specificity and accuracy were 91.7% and 90.9% respectively, showing its effectiveness in predicting negative instances and overall correctness. The F1 score, a balance of precision and recall, was 70.6%, indicating the model’s overall performance.

*For the irony evaluation :*

The “Sentiment Analysis” model was evaluated using a confusion matrix and performance metrics. The model correctly identified 21 out of 30 negative tweets (True Positives), and accurately recognized 15 non-negative tweets (True Negatives). However, it incorrectly labelled 3 tweets as negative (False Positives) and missed 6 negative tweets (False Negatives). The model’s precision was 71.4%, indicating its reliability in predicting negative sentiment. It had a recall of 83.3%, showing its efficiency in identifying negative tweets. The model’s specificity and accuracy were 77.8% and 80.0% respectively, reflecting its ability to recognize non-negative sentiments and overall correctness. The F1 Score, which balances precision and recall, was 76.9%, indicating a solid overall performance. These results provide valuable insights into the model’s effectiveness in detecting negative sentiments.

*For LLM evaluation :*

The evaluation of the LLM Analysis focuses on its reliability, which is tested by checking the consistency of the model’s outputs when given identical inputs. The evaluation results indicate that the LLM Analysis is effective, as it consistently delivers well-formatted outputs, adeptly categorizes topics, provides detailed and practical maintenance suggestions, and has the ability to interpret and manage irony in tweets. These features contribute to reliable data interpretation, efficient task distribution, informed decision-making, and accurate sentiment analysis.



***Illustration 27:*** *AI Prompt*

Post evaluation, the model was then deployed. The described deployment architecture is a sentiment analysis solution that starts with acquiring Twitter data via the Twitter API. Visual Studio Code (VS Code) is used as the central hub for development and integration. Azure’s serverless function capabilities are utilized to deploy sentiment analysis functions, with the analysed data being stored in Azure Blob Storage. This data is then processed and moved into an Azure SQL Database using Azure Data Factory, which forms a crucial part of the data pipeline. The final step involves visualizing the enriched data in Power BI to provide insightful analytics. These real-time sentiment insights derived from the tweets are used by the Thameslink maintenance team. This architecture effectively combines the flexibility of serverless computing, the robustness of Azure services, and the visual storytelling capabilities of Power BI to deliver a comprehensive sentiment analysis solution.

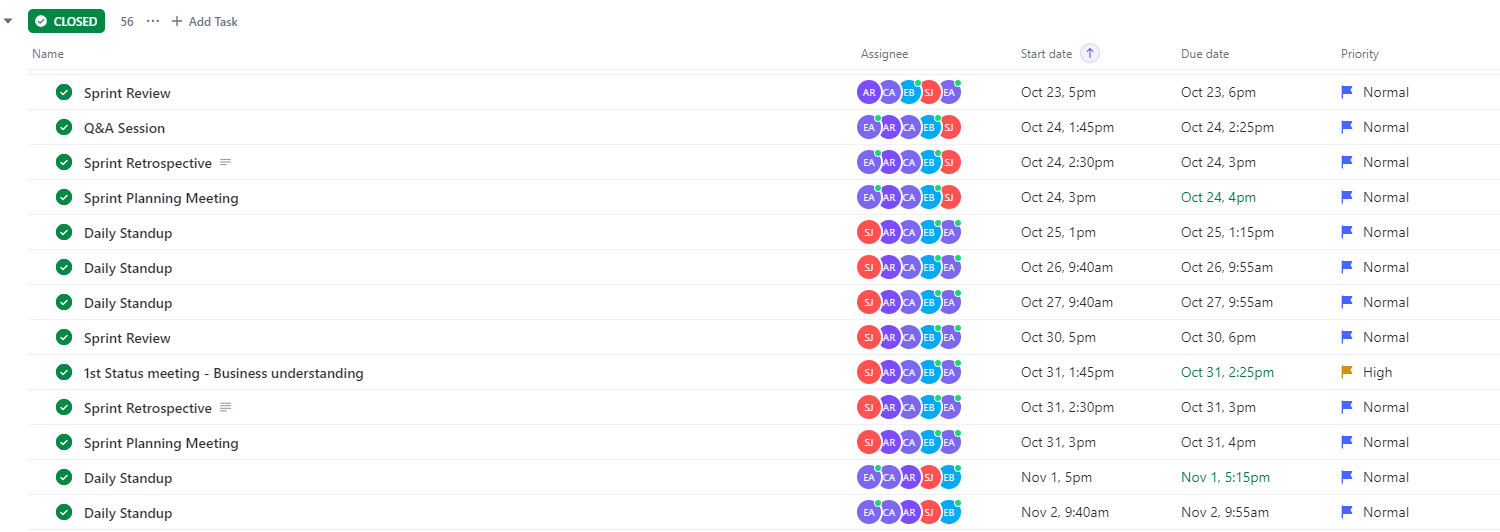
# 8. PROCESS

# 8.1. Project Inception and Team Dynamics

The journey of 'Tweets2GPT' began with the formation of The Fine Tuners group, a team structured to evolve and adapt throughout the project. The team members, Erjon Buka, Ecem Günhar Akuras, Shinu Joseph, Akshay Rajesh, and M. Cem Akuras, were designated diverse roles ranging from Project Owners to Scrum Masters and Developers across different sprints. This fluidity in roles allowed for a comprehensive understanding of the project from various perspectives, fostering a collaborative and multifaceted approach to problem-solving.

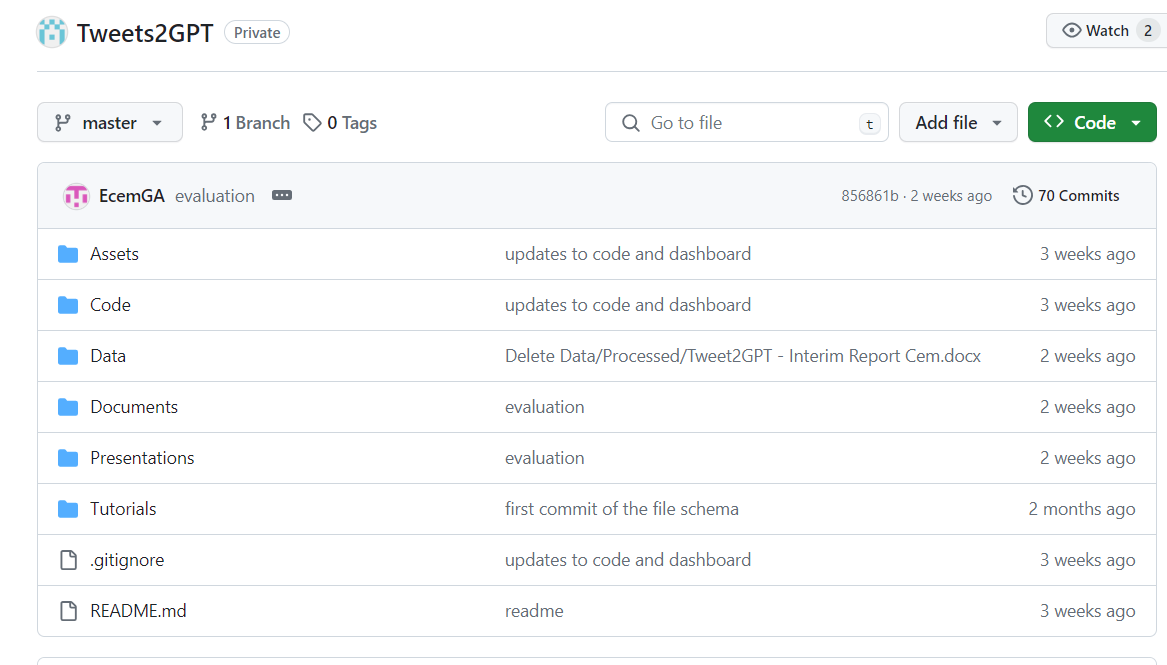
# 8.2. Agile Methodology and Communication

Our approach was grounded in the Agile framework, emphasizing iterative development and continuous evaluation. The project's heartbeat was the daily stand-ups, where team members shared their previous day's achievements, current day's objectives, and any obstacles they faced. These meetings were not just routine checks but pivotal in maintaining project momentum and ensuring team alignment.



***Illustration 28:*** *Agile Ceremonies - ClickUp Snapshot*

Weekly sprint reviews and retrospectives formed the cornerstone of our reflective practice. In these sessions, we showcased our completed work, gathered feedback, and critically evaluated our methods and outcomes. This iterative assessment, coupled with the planning sessions at each sprint's onset, ensured that our project was always aligned with its goals and could dynamically adapt to new insights or challenges.



***Illustration 29:*** *GitHub Repository Overview*

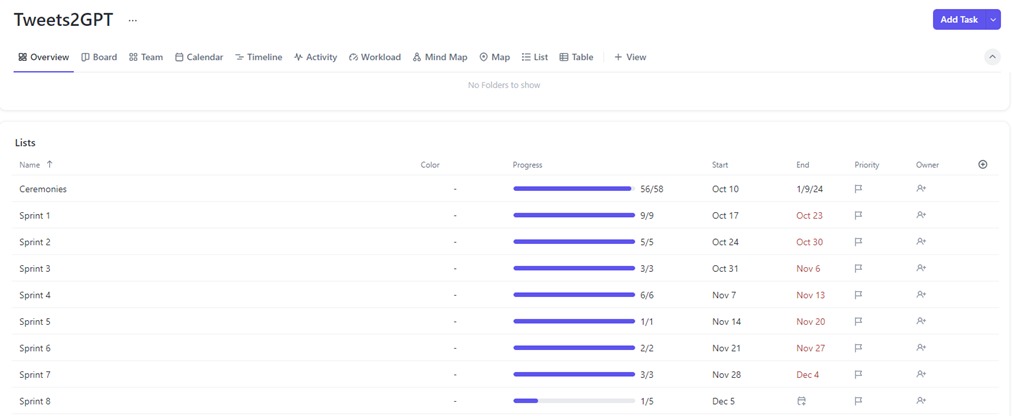
During this process, GitHub served as our central platform for version control and collaboration. The command-line interface, Bash, was an integral part of our workflow, allowing us to efficiently push and pull changes to the repository. This tool facilitated a smooth interaction with GitHub, making it accessible for team members to synchronize their work effortlessly. Using Bash scripts, we automated repetitive tasks, enhancing our productivity and allowing us to focus on the more creative aspects of the project. Its versatility and power made it an invaluable asset in our development toolkit, contributing positively to our project's progression.

# 8.3. Utilization of Development Tools

In the realm of development tools, Visual Studio Code and Jupyter Notebook were our chosen editor for its robust features. Visual Studio Code's user-friendly interface and extension marketplace enhanced our productivity and coding standards. Jupyter Notebook's interactive environment allowed for a seamless transition from data exploration to model development, providing an excellent platform for documenting our findings.

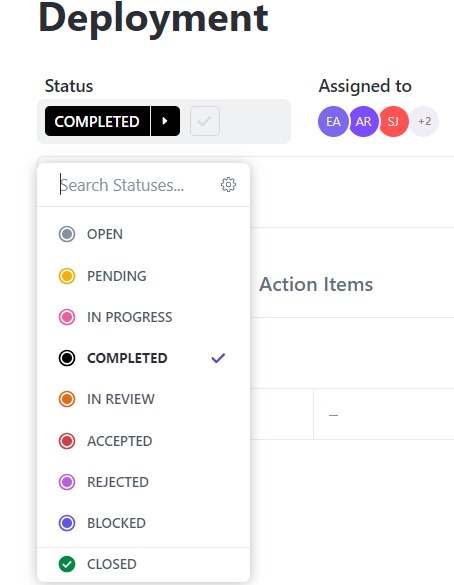
Additionally, our project leveraged key APIs to drive our data-centric solutions. The ChatGPT API was utilized to generate intelligent maintenance responses, training our models to provide automated, context-aware solutions for Thameslink's operational needs.

# 8.4. Utilization of Project Management Tools



***Illustration 30:*** *ClickUp Overview*

The project's organizational backbone was ClickUp, which provided a visual and interactive platform to manage tasks and monitor progress. Within ClickUp, we meticulously followed Agile ceremonies under the Ceremonies list, ensuring that processes such as Daily Stand-ups, Sprint Planning, Sprint Reviews and Sprint Retrospectives were systematically conducted. Our tasks for each sprint were organized and tracked, with each task being diligently completed by at least one group member and reviewed by another, fostering accountability and quality in our deliverables.



***Illustration 31:*** *ClickUp Task`s Status*

Upon creation, tasks were initially marked as 'Open,' signaling readiness for commencement. As team members began working on their respective assignments, the status of each task was updated to 'In Progress,' reflecting active development and effort. This transparent tracking of activity allowed for real-time monitoring and support, where necessary.

Once the tasks reached a satisfactory level of completion, their status was changed to 'Completed,' and subsequently moved to 'In Review,' where another layer of scrutiny was applied by different team members. This peer review process was critical, providing an opportunity for quality assurance and collaborative improvement.

Finally, tasks that met all our rigorous criteria were marked as 'Closed,' signifying their successful resolution and the achievement of their objectives. This structured approach to task status, as visualized in the ClickUp interface, ensured a clear pathway from initiation to closure, fostering a disciplined and efficient workflow that propelled the 'Tweets2GPT' project towards its milestones.

# 8.5. Role Evolution and Adaptation

A special aspect of our process was the role evolution within the team. Members transitioned between various responsibilities, which was not just a logistical necessity but a strategic decision to infuse the project with fresh perspectives and diverse skill sets. This adaptation was instrumental in overcoming unforeseen challenges and exploring novel solutions.

In the first half of the project, Erjon Buka served as the Product Owner, while Ecem Günhar Akuras took on the role of Scrum Master. The development team consisted of Shinu Joseph, Akshay Rajesh, and M. Cem Akuras. In the subsequent half, Shinu Joseph shifted to the position of Product Owner, and Akshay Rajesh assumed the role of Scrum Master. Concurrently, Erjon Buka and Ecem Günhar Akuras shifted to the roles of developers.

# 8.6. Continuous Learning and Feedback Integration

Our project was a learning journey as much as it was a development one. Feedback from both internal and external reviews was a vital source of learning and growth. It informed our decisions and strategies, ensuring that our project was not only technically sound but also aligned with stakeholder expectations and market realities.

In conclusion, the 'Tweets2GPT' project was a blend of agility, technical proficiency, and collaboration. Our team dynamics, supported by Agile methodologies and robust project management tools fulfilled the necessities of development. The synergy between the technologies we chose not only enabled the team to be successful in terms of outputs, but also provided new knowledge to each team member.

# 9. WORK BREAKDOWN

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Topic** | **Akshay Rajesh** | **M. Cem Akuras** | **Erjon Buka** | **Ecem Günhar Akuras** | **Shinu Joseph** |
| **Business Understanding** | 20% | 20% | 20% | 20% | 20% |
| **Data Understanding & Preparation** | 20% | 20% | 20% | 20% | 20% |
| **Modelling** | 20% | 20% | 20% | 20% | 20% |
| **Evaluation** | 20% | 20% | 20% | 20% | 20% |
| **Deployment** | 20% | 20% | 20% | 20% | 20% |
| **Result** | 20% | 20% | 20% | 20% | 20% |
| **Process** | 20% | 20% | 20% | 20% | 20% |
| **Executive Summary** | 20% | 20% | 20% | 20% | 20% |
| **Report Writing** | 20% | 20% | 20% | 20% | 20% |