FACULTY OF SCIENCE & TECHNOLOGY

BSc (Hons) Software Engineering
May 2025

Sentiment Analysis of Political Bias in Online Discourse

by

Reece Slade

Faculty of Science & Technology

Department of Computing and Informatics

Final Year Project

Abstract

In an era where online content significantly shapes political perspectives, understanding bias in digital discourse is increasingly important. This dissertation explores the use of sentiment analysis through large language models (LLMs) to detect potential political bias in news articles and YouTube videos, aiming to provide users with a transparent and interactive tool for evaluating sentiment across various sources. The project addresses growing concerns over the influence of algorithm-driven content and the reliability of sentiment classification, particularly in politically charged topics.

A full-stack web application was developed to enable users to search political content, apply LLMs to assign sentiment to YouTube video titles or news headlines via APIs, select from a range of LLMs, and view sentiment analysis results in real time. Users can agree or disagree with the sentiment assigned by the models, provide optional feedback, and view model performance and organisational statistics related to how different organisations use sentiment on a given search query. The system integrates external APIs to fetch live content and leverages LLMs such as BERT, GPT-3, and locally hosted models via Ollama for sentiment classification. User feedback on sentiment accuracy is collected and stored in a database to support future refinement.

Development was carried out iteratively, guided by a structured requirements framework using MoSCoW prioritisation and mapped through user stories and data flow diagrams. The final iteration focused on usability, reliability, and responsiveness across devices, with extensive testing conducted on different browsers and screen sizes. Ethical considerations, including model and source bias, were addressed through careful model selection and balanced content sourcing.

The result is a responsive, modular platform that offers users interpretability in navigating political sentiment online. By enabling comparative sentiment analysis across different models and organisations, the system contributes to ongoing discussions about bias in digital information ecosystems. This project demonstrates the practical application of sentiment analysis in political contexts and highlights the importance of transparent, user-involved systems in mitigating media bias.

Dissertation Declaration

I agree that, should the University wish to retain it for reference purposes, a copy of my dissertation may be held by Bournemouth University normally for a period of 3 academic years. I understand that once the retention period has expired my dissertation will be destroyed.

Confidentiality

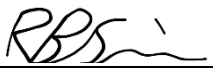
I confirm that this dissertation does not contain information of a commercial or confidential nature or include personal information other than that which would normally be in the public domain unless the relevant permissions have been obtained. In particular any information which identifies a particular individual's religious or political beliefs, information relating to their health, ethnicity, criminal history or sex life has been anonymised unless permission has been granted for its publication from the person to whom it relates.

Copyright

The copyright for this dissertation remains with me.

Requests for Information

I agree that this dissertation may be made available as the result of a request for information under the Freedom of Information Act.

Signed: 

Name: Reece Slade

Date: 20/05/2025

Programme: BSc (Hons) Software Engineering

Original Work Declaration

This dissertation and the project that it is based on are my own work, except where stated, in accordance with University regulations.

Signed:  .

Name: Reece Slade

Date: 20/05/2025

Acknowledgments

I would like to express my sincere gratitude to my parents for their unwavering support and encouragement throughout my degree. Their belief in me has been a constant source of motivation.

I am also deeply grateful to my project supervisor, Marcin Budka, for his guidance, feedback, and support throughout this project.

I would like to thank my church community for their prayers and encouragement during my studies. Above all, I thank God for the strength, wisdom, and perseverance to complete this work.

TABLE OF CONTENTS

Contents

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 Problem Definition

In today's digital landscape, individuals are inundated with information from news platforms, social media, and many other online sources. While this abundance of content exposes users to diverse viewpoints, much of it is shaped by subtle political or organizational biases, embedded in nuanced language, selective framing, and contextual tone. These biases are not always overt, making them difficult to detect and analyse at scale.

Despite recent advances in natural language processing (NLP), particularly with large language models (LLMs) such as ChatGPT, key computational challenges remain. Many widely used methods, including sentiment analysis and keyword matching, still reduce complex texts to overly simplistic categories, often overlooking nuances like intent, ideological framing, and implicit bias. Even more advanced systems struggle to account for how meaning is shaped through rhetorical strategies, selective phrasing, or omission. While sentiment analysis has improved significantly, existing approaches often fail to capture these deeper contextual elements, leaving a notable gap in the field *(Hussein, 2018)*. Bias detection still relies heavily on domain-specific heuristics or labelled datasets, limiting scalability and adaptability to new or evolving narratives. In politically sensitive contexts, where meaning is subjective and contested, the absence of a definitive ground truth further complicates analysis and data curation.

1.1.1 Problem Statement:

Current NLP tools and sentiment analysis methods lack the nuance and contextual awareness needed to detect subtle political bias in online discourse, leaving users unable to critically evaluate the impartiality of the information they consume. This contributes to wider issues such as misinformation, political polarisation, and declining public trust in digital media platforms.

1.2 Aims and objectives

The overarching aim of this project is to explore and evaluate the effectiveness of LLMs in analysing sentiment within online political discourse. This will be achieved by developing a reliable, accurate, and user-focused web application, built on robust principles of software engineering, system design, and data science. Table 1 outlines the specific objectives and their corresponding success criteria that guide the project from initial development through to final evaluation.

Objective	Success Criterion
1. Conduct comprehensive research into the capabilities of LLMs and apply them to analyse sentiment in online political discourse.	Relevant literature and model research are incorporated throughout the project; one or more LLMs are successfully applied to analyse both news headlines and YouTube video titles.
2. Implement and document best practices to ensure reliable and accurate sentiment assignment using LLMs.	A range of LLMs are tested; sentiment outputs and explanation prompts are evaluated for clarity, coherence, and logical consistency.
3. Employ software engineering best practices, including modular architecture, version control, and testing strategies.	The application demonstrates modular design, clean code practices, and systematic testing.
4. Integrate real-time political discourse data using third-party APIs.	APIs are correctly authenticated, queried, and parsed to retrieve and populate the system with relevant real-world data.
5. Identify and prioritise functional and non-functional requirements, refining scope iteratively through corresponding user stories and acceptance criteria.	A documented and evolving list of prioritised requirements is maintained, with corresponding user stories and acceptance criteria aligned to each development iteration.
6. Deliver the project according to the agreed timeline and university submission deadlines.	All planned milestones are met in accordance with the project schedule.

7. Develop a responsive, intuitive web interface that visualises LLM-driven sentiment and statistical insights.	A fully functional web application is deployed, compatible across devices and browsers, with visualisations of sentiment output.

Table 1: Objectives and Success Criteria

1.3 Project Risks and Mitigations

Table 2 outlines key risks associated with the project, providing an estimate of their likelihood, potential impact, overall risk level, and the corresponding mitigation strategies designed to minimise disruption or failure.

Risk	Category	Likelihood	Impact	Risk Level	Mitigation Strategy
Running out of time	Project Management	**Medium**	**High**	**High**	Prioritise critical requirements, define clear milestones, and use agile practices to maintain steady progress.
Project complexity	Technical	**Medium**	**High**	**High**	Break down the project into smaller tasks, adopt modular design, and aim for an MVP early in development.
API rate limits	Technical	**High**	**Low**	**Medium**	Implement request throttling, caching and inform users of any request limits.
Data or progress loss	Data Integrity	**Medium**	**High**	**High**	Use Git for version control with frequent commits and regularly back up key files to external storage.
Computational limitations with LLMs	Technical	**Medium**	**High**	**Medium**	Select a diverse set of models with varying parameters to ensure compatibility with devices having different levels of processing power.
User tampering via browser tools	Security	**Medium**	**High**	**High**	Apply strict input validation and sanitisation on both frontend and backend components.
High traffic or system slowdown	Technical	**Low**	**Medium**	**Medium**	Optimise backend performance, use caching, and monitor system load to ensure responsiveness.
Ethical concerns in sentiment classification	Ethical	**High**	**High**	**High**	Clearly communicate to users that all sentiment outputs and explanations are AI-generated and may not reflect objective truth.
Bias in LLM outputs	Ethical	**High**	**High**	**High**	Use diverse models from different organisations and include disclaimers about AI generated results.

Limited diversity in news sources	Ethical / Data Integrity	**Medium**	**Medium**	**Medium**	Select a broad range of reputable news sources with varying political leaning.

Table 2: Risks and Mitigations

2 BACKGROUND RESEARCH

2.1 Overview

This chapter provides a more in-depth understanding of the project's problem domain, considering the software engineering, system design, and data science considerations that influence the development of the artefact. The research was conducted in the following areas:

Machine Learning (ML) and Model Development:
Research into ML and model development was conducted to identify the most suitable models for analysing sentiment in news headlines and video titles. This included different language models, fine-tuning, and selection criteria to ensure accurate and reliable sentiment classification.

System Architecture and Implementation:
Research into software engineering principles such as separation of concerns, error handling and testing were undertaken to guide the development of a high-quality application. By applying best practices in architecture design the project aimed to minimize errors and ensure robustness, efficiency, and ease of future improvements.

Ethical Considerations and Bias:
To mitigate bias, research was conducted to identify and address potential sources of skew in sentiment analysis. Ensuring the selection of reliable and diverse news sources and LLMs allowed users to access a broad range of perspectives. This approach promotes a more balanced understanding of political discourse and reduces the risk of model-induced bias.

2.2 Machine Learning (ML) and Model Development

2.2.1 Sentiment Analysis and NLP

Sentiment Analysis (SA) is a field within Natural Language Processing (NLP) that involves text analysis and computational linguistics to identify and extract subjective information from textual data. It plays a crucial role in understanding public opinion, perception, and social dynamics. The rise of social media and digital news platforms has led to an explosion of textual data, providing a vast resource for sentiment analysis *(Joseph, 2024).*

2.2.2 Model Selection in Sentiment Analysis

Selecting an appropriate model for sentiment analysis is crucial, as different approaches perform better depending on the context and dataset characteristics. The choice involves balancing trade-offs between performance, computational efficiency, and interpretability. Deep learning models, particularly Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks, and transformers, have gained popularity due to their ability to capture complex linguistic patterns *(Zhang, Wang, and Liu, 2018)*. Among these, transformer-based models have proven highly effective in sentiment analysis due to their contextual understanding capabilities.

2.2.3 Transformers and Large Language Models in Sentiment Analysis

State-of-the-art transformer-based models such as BERT and GPT-3 have outperformed traditional machine learning approaches like Naïve Bayes and Support Vector Machines in sentiment analysis tasks. These models leverage deep contextual embeddings, allowing for a more nuanced understanding of sentiment within text *(Devlin, Chang, Lee and Toutanova, 2018).*

These models have also demonstrated strong performance when fine-tuned on domain-specific datasets. Their strength lies in generating coherent, contextually accurate outputs and understanding language intricacies due to training on large-scale corpora *(Vaniukov, 2024).* However, one limitation is their input size constraint, typically capped at 512 tokens, restricting their effective context window to around 350 words. This can be a drawback when analysing longer documents or complex arguments that span multiple paragraphs *(Gao et al., 2021)*. Nevertheless, for shorter texts such as social media posts or video titles, these models remain highly effective.

In parallel, Large Language Models (LLMs) offer a significant evolution in NLP capabilities. These models not only perform sentiment classification but can also generate coherent text aligned with the input context *(Vaniukov, 2024).* They reduce dependency on large, labelled datasets through few-shot and zero-shot learning, enabling efficient deployment in domains where data is sparse. Moreover, their ability to return explanatory context with outputs enhances transparency in sentiment classification, a key feature in politically sensitive applications. Recent models have begun to address token limitations by supporting longer context windows, though such features are currently limited to specific platforms.

2.2.4 Comparing Different Model Approaches

Table 3 highlights a comparison of different model types and illustrates the trade-offs between various sentiment analysis approaches:

Model Type	Strengths	Weaknesses
Pre-trained Transformer Models (e.g., BERT, RoBERTa)	Strong contextual understanding; effective with limited data via transfer learning	Computationally intensive; requires fine-tuning for specific domains
Large Language Models (e.g., GPT-3, GPT-4)	Versatile; handles multiple NLP tasks; few-shot learning	High computational cost; risk of bias or hallucinated outputs
Training from Scratch	Fully customizable; avoids pre-trained model biases	Requires vast labelled datasets; time and resource heavy
Traditional Models (e.g., Naïve Bayes, SVM)	Fast and interpretable; suitable for small, clean datasets	Poor performance on nuanced or context-heavy text

Table 3: Trade-Offs Between Sentiment Analysis Approaches

2.2.5 Introduction to Ollama

Ollama is a platform offering streamlined access to powerful LLMs via simple API integration. It provides a library of pre-trained models for a range of NLP tasks, including sentiment analysis, without requiring significant computational resources or fine-tuning.

Unlike external APIs such as OpenAI or Hugging Face, which require internet access and cloud-based tokens or credits, Ollama's local-first model access aligns better with the project's goals of portability and cost-efficiency.

A key strength of Ollama lies in its scalability. Users can choose models with varying parameter sizes, allowing for a balance between performance and computational demand depending on the use case. Moreover, Ollama abstracts away much of the traditional NLP preprocessing (e.g., tokenisation, stop-word removal), as its models are designed to handle these internally. This greatly reduces development overhead and simplifies deployment *(Ollama, 2025).*

Final Thoughts:
The project carefully considered the trade-offs between performance, efficiency, and accessibility when selecting models for sentiment analysis. Due to time and resource constraints, training a model from scratch or extensively fine-tuning pre-trained models was not feasible, as these typically require cloud-based environments such as Google Colab or Kaggle. Ollama was identified as a practical and scalable alternative. Its models delivered high accuracy with minimal setup and enabled contextual sentiment analysis without extensive preprocessing. Furthermore, its flexibility in model selection allowed the application to run across various hardware configurations. This approach helped streamline development while maintaining interpretability and transparency - an essential consideration in politically sensitive domains such as news and online discourse analysis.

2.3 System Architecture and Implementation

This section explores established software and system engineering practices applied to ensure the artefact's quality and maintainability.

2.3.1 Separation of Concerns

The project followed the key software engineering principle of Separation of Concerns (SoC). This principle involves breaking down the system into smaller, well-defined components, each handling a specific responsibility. By isolating concerns, the system becomes more modular, maintainable, and scalable. SoC minimized the risk of failures by promoting loose coupling between components and encapsulating logic, which simplifies development and troubleshooting *(GeeksforGeeks, 2022).* Ultimately, this design approach enhances the overall robustness and adaptability of the system, ensuring long-term ease of maintenance and evolution.

2.3.2 Version Control

A version control system (VCS) is a vital tool for the project in mitigating the potential risk of data or progress loss during development (Table 2). In the event of hardware failure, the most recent version stored in the VCS remains accessible. This can also protect the codebase from detrimental changes as it can be rolled back to a safe version from the commit history, ensuring stable recovery. GitHub was selected for the VCS of this project as it is widely adopted and well-documented. GitHub enables the codebase to be securely stored on a cloud platform to mitigate the risk of data loss.

2.3.3 Error handling and Logging

Error handling and logging were implemented because when thoughtfully designed, error messages can enhance the user experience and assist users during unexpected behaviour, promoting trust and transparency. Effective error handling enhances the user experience by offering clear, actionable feedback, while comprehensive logging enables better debugging and system maintenance *(Vadhadia, 2023).* Additionally, these practices could improve system reliability and ensure smoother operation over time.

2.3.4 Testing

The project adopted a combination of exploratory and ad hoc testing methods to minimize defects and improve software reliability. These approaches are particularly effective for identifying edge cases that structured testing may overlook. They promote real-time learning, test design, and execution, allowing testers to adapt dynamically as they uncover issues *(Atlassian, 2025).* Both methods are flexible and tester-driven, prioritizing adaptability over rigid test scripts. While their lack of structure makes them unsuitable for high-risk or critical systems *(Wikipedia, 2024),* they are well-suited to a project with limited scope and lower associated risks. This strategy enables the system to be evaluated under unexpected conditions, helping to assess its stability, security, and overall resilience beyond standard scenarios.

2.4 Ethical Considerations and Bias

Eliminating all ethical concerns and inherent biases remains a significant challenge, especially within data sources and selected LLMs. These biases are not always fully understood, and research was undertaken to minimize these biases as much as possible, ensuring a fair and diverse representation of news sources and LLMs.

2.4.1 Selection of Organizations and Media

One of the primary challenges is ensuring that the selection of news sources reflects a broad spectrum of political perspectives. This is essential as audiences often form opinions based solely on headlines depending on the political figure or topic being addressed by different organizations.

Country	Top 5 Most-Subscribed News and Politics YouTube Channels	Top 5 Most Visited News Websites
UK	1. Sky News 2. Daily Mail 3. Guardian News 4. Channel 4 News 5. Piers Morgan Uncensored	1. BBC 2. Mail Online 3. The Guardian 4. The Sun 5. The Mirror
US	1. ABC News 2. CNN 3. BBC News 4. Al Jazeera 5. English Fox News	1. The New York Times 2. CNN 3. Fox News 4. MSN 5. People.com

Table 4: Most Popular News Content Organisations (Appendix A & B), (Press Gazette, 2024; HypeAuditor, 2024).

The decision to utilize YouTube and Online News as primary data sources was driven by their wide reach and significant influence. These platforms often employ attention-grabbing language, shaping public perception as many form opinions based on headlines or video titles without fully engaging with the content. A study by Columbia University and the French National Institute found that 59% of shared links on social media go unopened, illustrating how news is often retweeted without being read *(Dewey, 2016).*

To ensure a balanced representation and prevent users from being exposed to content from only one political perspective, additional research was conducted into the political leanings of various news sources. This was done to reflect the differing ways media outlets present public figures. Source selection was guided by the AllSides Media Bias Chart (Figure 1), which categorizes outlets across the political spectrum. News content was retrieved using the NewsAPI, though its free tier limits the range of available sources compared to YouTube's API, which offers broader access to news channels *(NewsAPI, n.d.; Google Developers, n.d.)*. Individual media figures, such as Piers Morgan (Table 4), were excluded due to their unavailability on NewsAPI. To maintain consistency and ensure fair comparisons, the same sources will be used for both news and video content across the application.

Figure 1: Organisational Political Leaning (AllSides Media Bias Chart™. AllSides, 2024.)

2.4.1 Selection of LLMs

LLMs have revolutionized industries by handling complex tasks such as logical reasoning. However, they are not without limitations, especially when it comes to biases introduced by developers or governing entities. For instance, DeepSeek, a Chinese-developed AI chatbot, avoids

sensitive topics like the Tiananmen Square protests or Taiwan's political status, aligning with the Chinese government's official stance *(Lu, 2025).*

Similarly, Google's Gemini AI chatbot made controversial decisions, prioritizing gender identification over averting a nuclear catastrophe in a hypothetical scenario, highlighting potential ethical conflicts in AI decision-making *(LiveMint, 2024).*

These examples demonstrate that AI models can reflect biases, leading to unfair or unjust outcomes, particularly in tasks like sentiment analysis. To address this, the project integrates multiple open-source LLMs via Ollama, allowing users to select their preferred model for sentiment analysis. This ensures greater transparency and accountability, as users can explore the reasoning behind each model's sentiment and understand potential biases, fostering trust in AI-driven analysis.

2.5 Summary of Background Research

The research in this chapter has provided essential insights into best practices in software engineering, system design, machine learning, and data science, alongside the ethical concerns of bias. These principles will guide the design, development, and evaluation of the artefact. By employing clean code practices, modular design, rigorous testing, and leveraging diverse data sources and open-source models, the project ensures both technical accuracy and ethical integrity. The integration of data science methods ensures the reliability and precision of the results, fostering transparency, fairness, and trust in the final product.

3 METHODOLOGY AND IMPLEMENTATION

An effective development methodology is essential to deliver core functionality on time and within scope. Given the project's limited timeframe and evolving requirements, an Agile approach has been adopted. Agile supports iterative development, continuous refinement, and flexibility in response to change *(Vijayasarathy & Turk, 2008; Balaji et al., 2012).*

To manage evolving priorities, the MoSCoW method is used throughout. It categorizes requirements into:

- **Must Have** - Critical for release
- **Should Have** - Important but not vital
- **Could Have** - Desirable if time permits
- **Won't Have** - Out of scope for this release

This ensures that essential features are prioritized while maintaining the flexibility to accommodate additional enhancements.

Development progress is managed using the Kanban framework, a visual task management system that tracks tasks across stages such as *To Do*, *Doing*, and *Done (Anderson, 2010).* Kanban provides real-time visibility, limits work-in-progress (WIP) and ensures focused effort on current priorities. Figure 2 shows a simplified Kanban board; a project-specific version is provided in Appendix E.

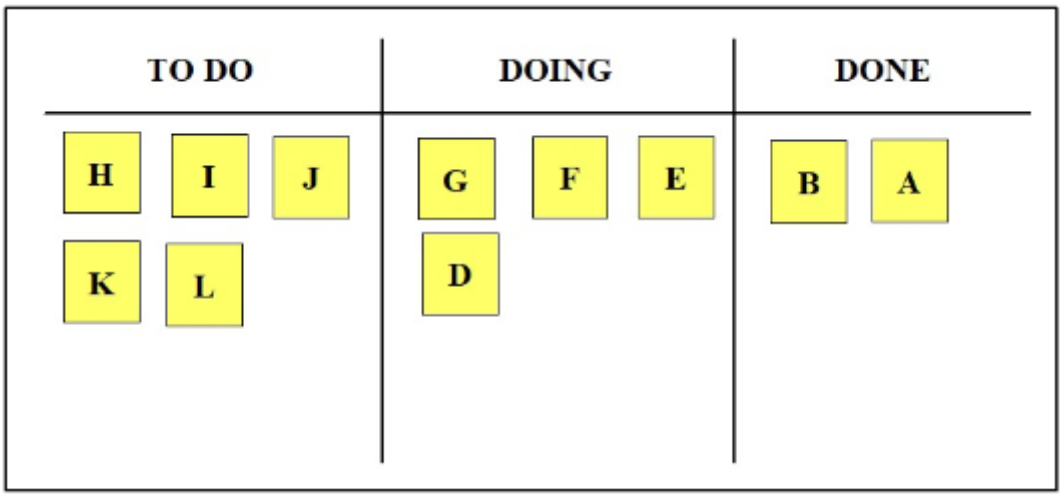


Figure 2: A Simple Kanban Board (Alaidaros, Omar and Romli, 2021)

Kanban's strength lies in its ability to maintain visibility and control over flow, making it ideal for solo or small-team development. Compared to frameworks like Scrum, which rely on fixed-length sprints, Kanban supports continuous delivery, helping to reduce lead times and adapt to change more efficiently *(Flora & Chande, 2014).* Kanban led to a 50% reduction in lead time and an 11% drop in defects - evidence of its practical benefits for managing complexity in software projects (*Sjøberg et al., 2012)*.

3.1 Execution of Agile Methodology: Iteration Planning and Execution

3.1.1 Iteration Planning

Development is structured into iterations lasting 2-3 weeks. Each begins with defining current requirements and ends with a review to identify outstanding issues and guide the next cycle. This supports adaptive planning and gradual integration of new features.

High-level goals are broken into actionable tasks, prioritized using MoSCoW, and tracked on the Kanban board. This ensures visibility, aligns contributors, and supports incremental delivery based on evolving project needs.

3.1.2 System Design: User Stories and Data Flow Diagrams (DFDs)

System design is driven by Data Flow Diagrams (DFDs), which illustrate how data moves through the application. These evolve with the system, reflecting architectural and functional changes.

In parallel, user stories define functionality from the user's perspective. Each story aligns with functional or non-functional requirements and includes clear acceptance criteria to ensure completeness and relevance. MoSCoW prioritization is applied to these stories to maintain focus and alignment during development.

3.1.3 Iteration Execution

Tasks are executed and updated on the Kanban board, offering a clear view of ongoing progress. Whenever possible, issues are addressed within the same iteration to avoid carryover. The board helps identify blockers early and maintain project momentum.

3.1.4 Iteration Review and Adjustment

Each iteration concludes with a review assessing feature completion and adherence to requirements. A structured issue table is generated to document challenges and propose resolutions. This supports continuous improvement and risk management. A full issue table is provided in Appendix F.

3.2 Requirements

To ensure core functionality is delivered within the timeframe, requirements are prioritized using the MoSCoW method *(Moisiadis, 2002),* as outlined earlier. This structured prioritization enables systematic refinement as the project evolves. All changes are documented and version-controlled to maintain traceability and transparency.

4 DEVELOPMENT OF THE ARTEFACT

4.1 Development Iteration 1

This iteration focused on delivering a functional Minimum Viable Product (MVP) that forms the foundation of the system. Users can input politically focused queries, apply filters through dropdown menus, and view sentiment analysis results rendered dynamically. This version satisfies the core *Must Have* requirements outlined in the table 5, providing a working baseline for further development and refinement.

4.2 Requirements and User Stories

Table 5 maps the initial requirements to their corresponding user stories, with each story featuring clear acceptance criteria to ensure both functional and non-functional requirements are met. As development progresses, new requirements will be introduced based on iterative reviews and system growth, with prioritization to be determined using the MoSCoW method

Priority	**ID**	**Requirement Description**	**User Story**	**Acceptance Criteria**
Must Have	**F1**	Enable users to input politically focused search queries	As a user, I want to search for politically focused content so that I can find relevant information	User can enter search terms related to political topics or figures
Must Have	**F2**	Allow users to select from a range of dropdown options	As a user, I want to filter my search using dropdown options so that I can narrow my results	User can select various filter options from dropdowns
Must Have	**F3**	Execute search based on specified filters	As a user, I want my search to use the filters I've selected so that results match my criteria	System executes search with all specified filter parameters
Must Have	**F4**	Dynamically render relevant search results	As a user, I want to see relevant results from my search so that I can browse the content	System displays results that match the search query and filters
Must Have	**N1**	Responsive interface	As a user, I want the interface to work on various devices and browsers so that I can use it anywhere	Interface adapts to different screen sizes and browsers
Must Have	**N2**	User-friendly interface	As a user, I want an intuitive interface so that I can use the system efficiently	Interface follows UX best practices and is easy to navigate
Must Have	**N3**	Clean UI design	As a user, I want a clean interface so that I can focus on the content	UI has minimal clutter and clear visual hierarchy
Must Have	**N4**	Accessibility features	As a user with accessibility needs, I want accessible features so that I can use the system	Interface follows accessibility standards (WCAG)
Must Have	**N5**	Error handling	As a user, I want clear error messages so that I know when something goes wrong	System provides user-friendly error messages

Must Have	**N6**	Security features	As a user, I want my data to be secure so that I feel safe using the system	Sensitive data is properly secured
Must Have	**N7**	Maintainable code	As a developer, I want modular code so that I can maintain it efficiently	Code follows best practices for maintainability

Table 5: Iteration 1 – Requirements and User Stories

4.3 Design

Before development began, careful planning was undertaken to ensure the system's core objectives were effectively met (Table 1). Both visual and logical design steps were employed to create a development roadmap, ensuring a smooth and coordinated process.

4.3.1 Design Overview

A Data Flow Diagram (DFD) was created to visualise user interaction and system flow (Figure 3). The interface design drew inspiration from Google's Advanced Search (Figure 4), with responsiveness and accessibility achieved via Bootstrap (N1-N4). A modular base template was introduced to ensure layout consistency across views (N7).

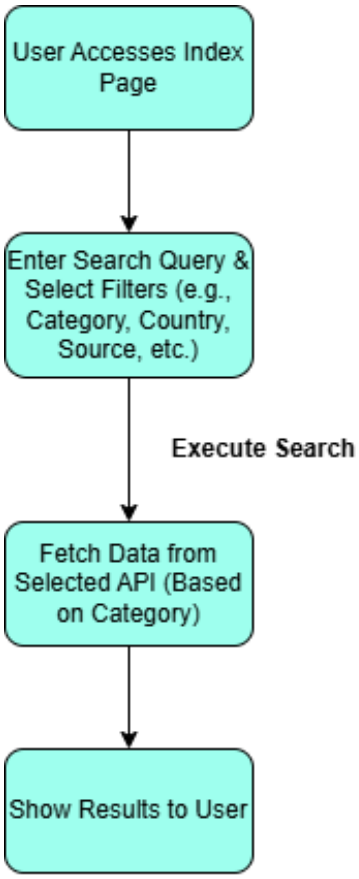


Figure 3: Iteration 1 – Data Flow Diagram

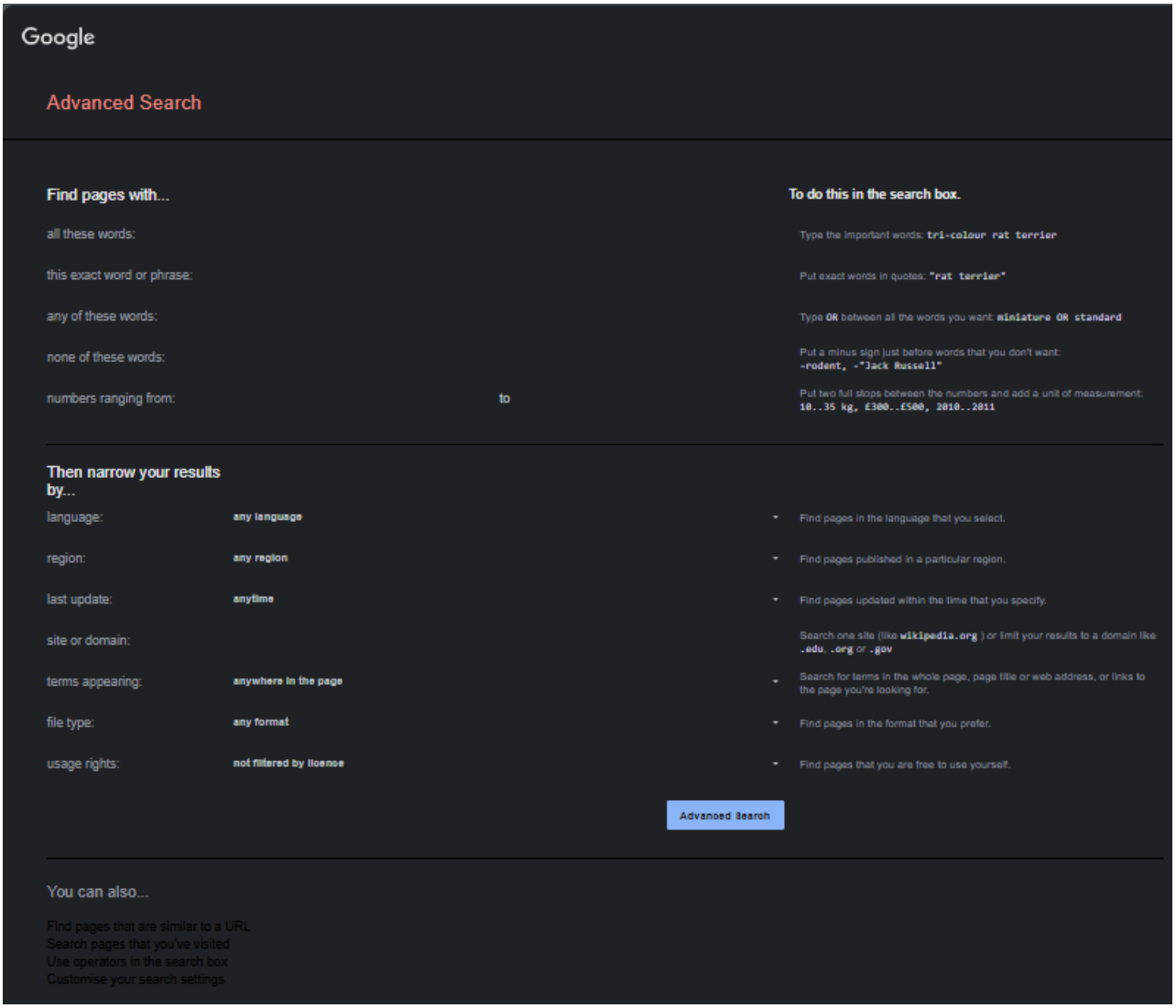


Figure 4: Iteration 1 – Data Flow Diagram

4.3.2 Architecture and Project Structure

The system follows a modular structure, separating routing, configuration, and data handling for maintainability (N7), (Figure 5). External API logic was encapsulated within isolated modules, while credentials were securely managed through environment variables and external config files (N6).

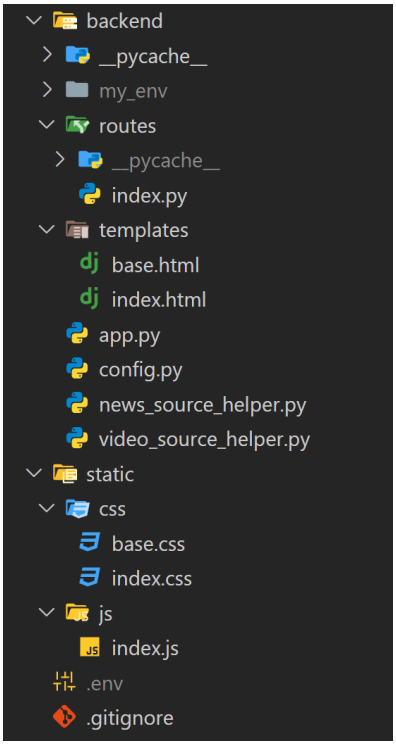


Figure 5: Iteration 1 – Initial Project Structure

4.4 Build

4.4.1 Core Functionality Implementation

Initial development focused on fetching and displaying results from the News and YouTube APIs based on user input (F1). API keys were secured using Python's os library (N6), and a lightweight interface was built to render results (F2-F4, N1-N5), (Figures 6 & 7). This established the groundwork for future improvements. All tasks were managed via a Kanban board (Appendix E).

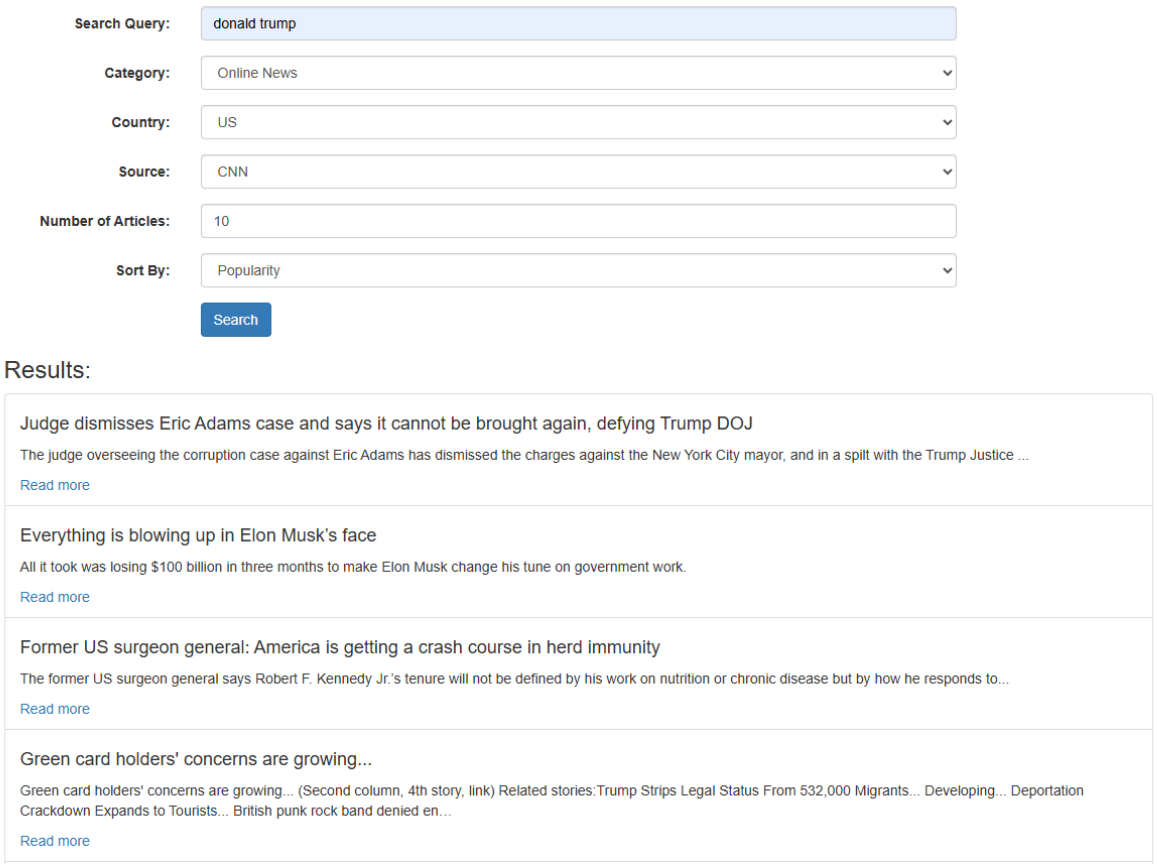


Figure 6: Iteration 1 – Index Search Page with Online News Results

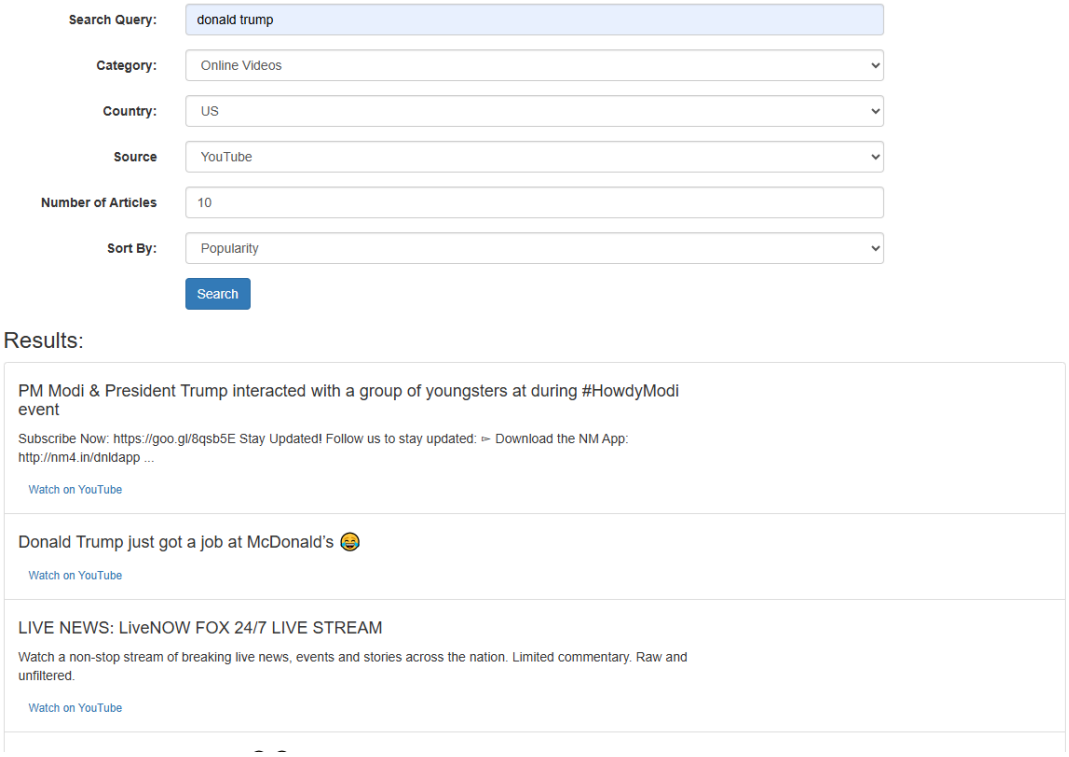


Figure 7: Iteration 1 – Index Search Page with Online Videos Results

4.4.2 Testing

Initial testing was limited to exploratory checks of the implemented core features, with basic error handling introduced to support user feedback (N5). Testing focused on a single country and news source to validate API integration, following the documentation provided by NewsAPI and Google Developers *(NewsAPI, n.d.; Google Developers, n.d.).* Broader coverage and additional sources are planned for future iterations.

4.4.3 Identified Issues

Although testing was limited, table 6 identifies several issues found during testing in this iteration:

Issue ID	Issue Description	Planned Solution
I	The current implementation retrieves all API content at once, causing inefficiency and delays, which degrades the user experience.	Introduce sequential loading to load content incrementally, reducing waiting times, and showcase a progress bar for feedback.
II	The interface does not dynamically update based on selected categories, allowing incompatible dropdown selections, leading to incorrect parameter submission.	Implement JavaScript to dynamically update the UI based on category selections, preventing incorrect parameter submission.
III	Some retrieved content does not explicitly mention the search query in the title or headline, which reduces the accuracy of sentiment analysis.	Apply stricter searches to filter out irrelevant content and ensure more accurate sentiment analysis.

Table 6: Iteration 1 – Identified Issues & Planned Solutions

4.5 Iteration Retrospect

The first iteration was successful in building the core search functionality, implementing key functional & non-functional requirements, all whilst maintaining a clean codebase. The design decisions and implementation strategies have laid a strong foundation for future iterations.

In the next iteration, the focus will shift towards addressing the issues that arose, improving the UI and integrating sentiment analysis using Ollama's API. Users will also be able to select a large language model to process and analyse results, with output that includes sentiment tags and optional explanatory summaries. These planned upgrades will enrich the user experience and bring the application closer to its vision of analysing and validating politically charged content through AI.

4.6 Development Iteration 2

The second iteration tackled issues outlined in Table 6, with a focus on performance improvements and enhancing the user experience (N1-N5 & N8). Work also began on integrating Ollama's API for assigning and justifying sentiment classification (F5-F7). Progress was managed using a Kanban board (Appendix E), ensuring the newly outlined functional and non-functional requirements were met (Appendix C).

4.7 Requirements and User Stories

Table 7 outlines the newly defined requirements developed to extend the application's functionality.

Priority	ID	Requirement Description	User Story	Acceptance Criteria
Must Have	**F5**	Integrate Ollama API for sentiment analysis	As a user, I want content to be analysed for sentiment so that I can understand the organisations emotional tone	System connects to Ollama API, works with prompts and processes content
Must Have	**F6**	Enable sentiment classification on fetched content	As a user, I want to see sentiment labels for content so that I can understand potential organisational bias	System assigns sentiment labels (positive/negative/neutral) to content
Must Have	**F7**	Provide justifications for sentiment classifications	As a user, I want explanations for sentiment labels so that I understand the reasoning behind assigned labels, and to determine if the selected LLM has any inherent biases towards any political topics or figures	LLM generates explanations for assigned sentiment labels
Must Have	**N8**	Performance optimization	As a user, I want fast response times so that I don't have to wait	System responds within acceptable time limits

Table 7: Iteration 2 – Requirements and User Stories

4.8 Design

This section refines the designs from the first iteration and incorporates new functionality to meet the updated requirements. The designs will outline both the logical and visual structure of the artifact.

4.8.1 Revised Data Flow Diagram

The revised data flow diagram (Figure 8) incorporates the new functionality introduced in this iteration, building on the version created during the first iteration (Figure 3). This diagram was essential in visualizing the interactions between the new and existing features, guiding the development of this phase.

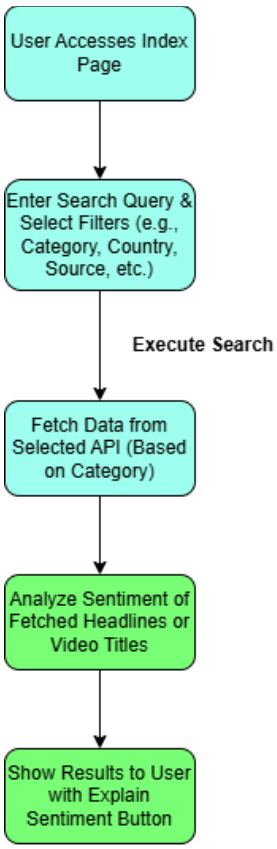


Figure 8: Iteration 2 – Revised Data Flow Diagram

4.8.2 Architecture and Project Structure

A base class, sentiment_analyser.py, was introduced to handle sentiment analysis for both news headlines and video titles (F6), maintaining the principle of separation of concerns (section 2.3.1). This abstraction reduced redundancy by providing a shared interface for different content types. Using polymorphism, the same method processes varying inputs (headlines or video titles) while ensuring consistent LLM interaction and response handling. This design simplifies the architecture, improves maintainability, and facilitates future system extensions (N7).

4.9 Build

4.9.1 Implementation of Ollama API

The first stage involved downloading an Ollama model and integrating its API for sentiment analysis (N6, F5, F6). The gemma:3b model was chosen for its small size and fast response, ideal for initial testing (N8), (Figure 9). Using the sentiment_analyser.py base class, two prompts were created: one to analyse sentiment of news headlines or video titles, and another to explain the reasoning behind the sentiment (F7), (Figures 10 & 11). This ensures consistent analysis and laid the groundwork for greater interpretability in future iterations.

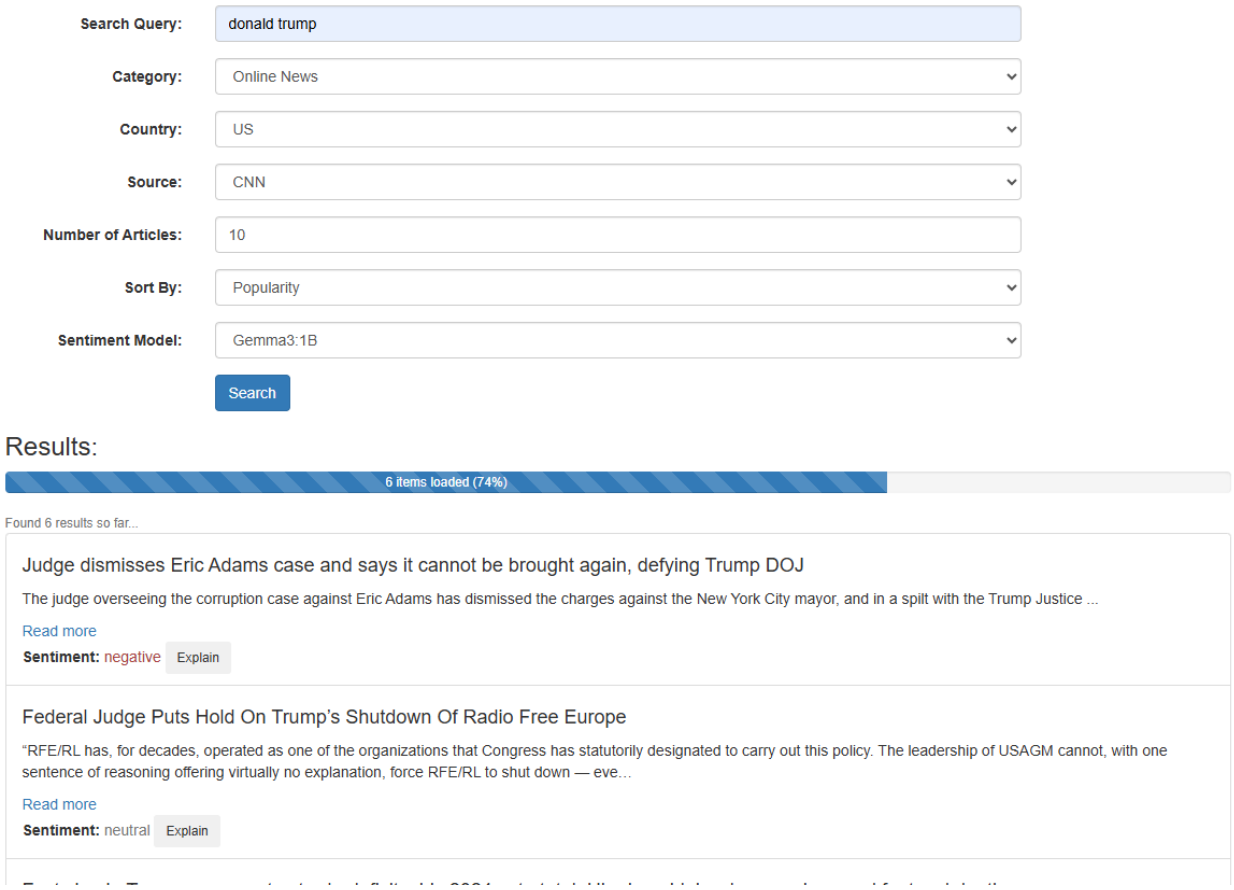


Figure 9: Iteration 2 – Search Results with Assigned Sentiment & Explain Button

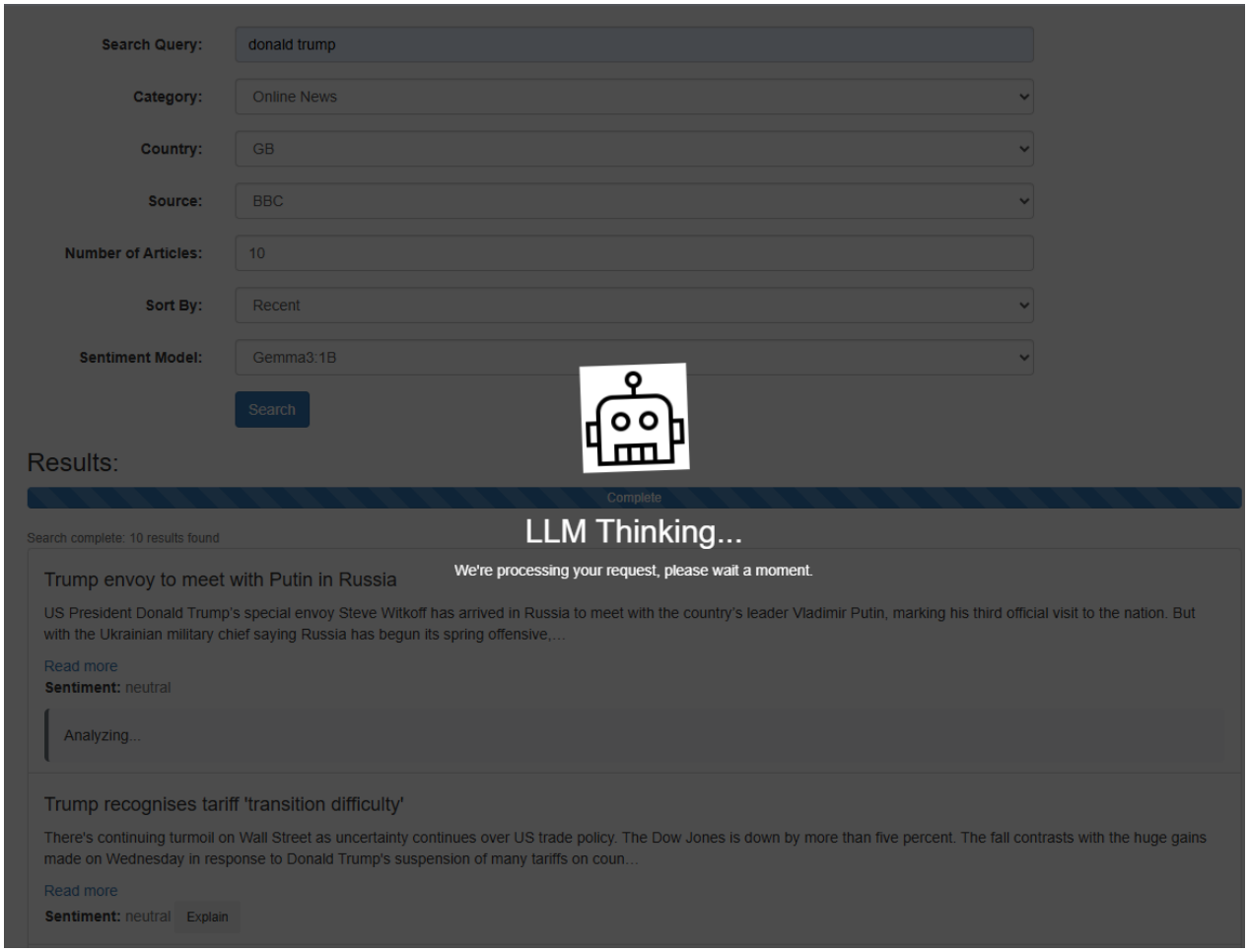


Figure 10: Iteration 2 – LLM Explanation Loading Screen

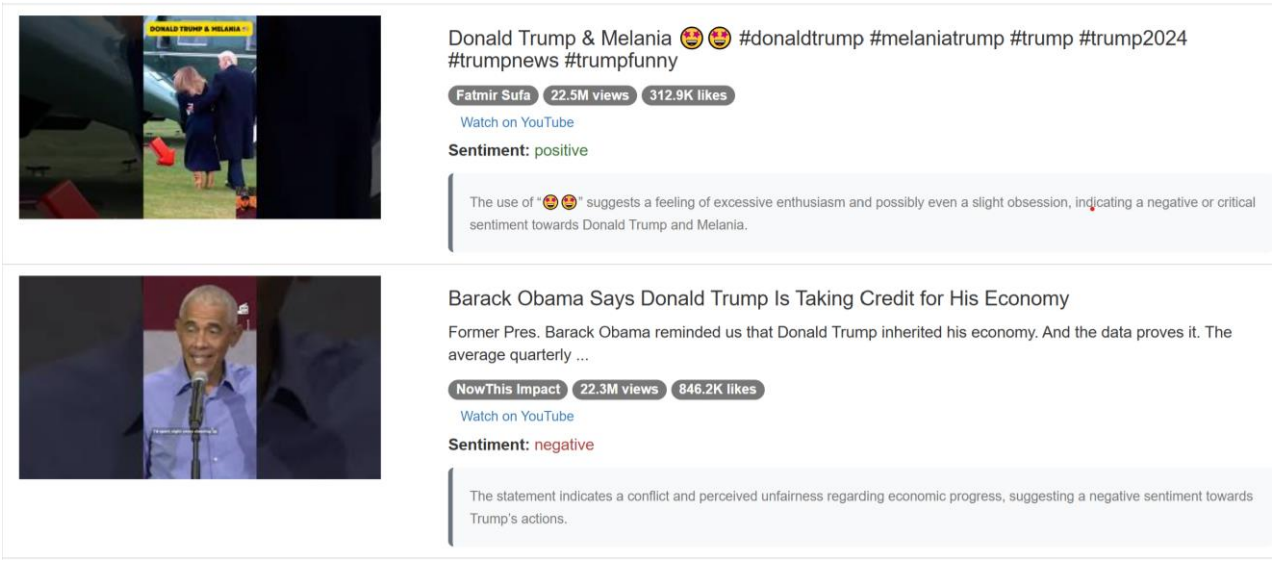


Figure 11: Iteration 2 – LLM Sentiment Explanation

4.10 Addressing Issues

Table 8 highlights the issues from iteration one and the solutions implemented to improve functionality and user experience.

Issue ID	Solution Implementation
I	Sequential content loading was implemented using EventSource to send items as separate messages, triggering the onmessage event. This parses the item, updates the progress counter, adds it to the cache, and renders it in the UI. A progress bar visually tracks the loading process. Backend methods in the news and video source helper files yield results one by one as server-sent events. Sentiment results for common queries are cached to improve performance and reduce unnecessary API calls.
II	An onchange event listener was added to dynamically update the frontend form, ensuring only valid combinations of category and country are displayed. This prevents errors and guides users through correct input choices.
III	The NewsAPI's "everything" endpoint was refined to search strictly within headlines, filtering out irrelevant results. This ensures only relevant content is processed by the SentimentAnalyser for sentiment analysis and statistical evaluation.

Table 8: Iteration 2 – Identified Issues & Implemented Solutions

4.11 Testing

Exploratory testing, alongside ad-hoc testing, was carried out during this iteration to observe how the application responds to various edge cases. This process uncovered several issues and defects, which have been documented and addressed in the upcoming iteration.

4.12 Identified Issues

Table 9 identifies the issues found through testing during this iteration:

Issue ID	Issue Description	Planned Solution
IV	Users can repeatedly click the "Search" button, which results in uncontrolled API calls. This can quickly exhaust the API quota and impact system performance.	Implement JavaScript to disable the search button until all content loaded.
V	The “Explain” button can be triggered multiple times in quick succession, prompting the LLM to return different justifications for the same sentiment. This leads to inconsistencies and undermines the credibility of the model’s reasoning.	The system will remove the explain button once clicked to prevent users from triggering multiple LLM (Large Language Model) requests in quick succession and will enforce a one-explanation-per-result policy
VI	In some cases, the LLM assigns a sentiment label (e.g., positive) but provides an explanation that supports a different sentiment (e.g., negative), creating confusion and reducing trust in the system.	Refine prompt to align better between assigning sentiment and the generating corresponding explanation.
VII	Users can initiate new searches or switch platforms without cancelling ongoing searches. This leads to overlapping requests being sent to the backend, which can overload the server and result in unexpected behaviour.	Implement JavaScript to disable the search button until all content loaded so the backend isn’t overwhelmed.

Table 9: Iteration 2 – Identified Issues & Planned Solutions

4.13 Iteration Retrospect

Iteration two successfully integrated the newly defined core requirements (Table 7), resolved issues from the first iteration (Table 6), all while maintaining a clean and modular codebase (N7). The revised design and strategies have laid a solid foundation for future development (Figure 8).

In the next iteration, new requirements will be introduced, with focus on addressing the identified issues (Table 9), refining the UI (N1-N5), and adding a new web page to display sentiment statistics for each organization (F8). Originally intended to be stored in a file, sentiment statistics will now be stored in a database (F9) to ensure better data integrity, scalability, and long-term maintainability. Along with this integration and still adhering to best practices and future development, the database should also include automatic daily backups to prevent any data corruption or loss.

This database integration also enables a new model feedback system (F10), allowing users to provide input on the LLM's sentiment explanations, enhancing transparency and user engagement (section 1.2). This feedback, along with sentiment results, will be stored for long-term analysis and evaluation.

The feedback system sets the stage for future enhancements, such as using feedback to fine-tune alternative LLMs like those on Hugging Face (F11). While not a core feature, this potential upgrade could add significant value in future iterations.

4.14 Development iteration 3

The focus of this iteration was to resolve issues identified in iteration two (Table 9) and implement newly defined requirements (Table 10). The iteration introduced several functional enhancements aimed at improving the user experience (N1-N5), ensuring data reliability (F9 & F12), and providing more insightful feedback on sentiment analysis (Issue V).

4.15 Requirements and User Stories

Table 10 consolidates the newly defined iteration focusing on both technical and user-centric features to improve overall system functionality.

Priority	ID	Requirement Description	User Story	Acceptance Criteria
Must Have	**F8**	Display organization-specific sentiment statistics	As a user, I want to see sentiment statistics by organization so I can compare sources	System displays sentiment statistics per organization
Must Have	**F9**	Store sentiment data in a database	As a user, I want reliable data storage so that statistics are accurate over time	System stores sentiment data in PostgreSQL database
Must Have	**F10**	Implement model feedback system	As a user, I want to provide feedback on sentiment analysis so that I can indicate if I agree or disagree	Users can submit agreement/disagreement with sentiment labels
Could Have	**F11**	Support for fine-tuning alternative LLMs	As a user, I want access to different LLMs so that I can compare analysis approaches	System supports integration with Hugging Face models and finetunes

Should Have	**F12**	Automatic database backups	As an administrator, I want automatic backups so that data is protected	System performs regular database backups
Must Have	**F13**	Display user feedback	As a user, I want to see feedback from other users so that I can understand different perspectives	System displays submitted feedback on sentiment analysis

Table 10: Iteration 3 – Requirements and User Stories

4.16 Design

The data flow diagram (Figure 12) is an updated version from iteration two (Figure 8), reflecting the new features in this iteration. This update helps visualize how the newly introduced features integrate into the system's architecture.

4.16.1 Revised Data Flow Diagram

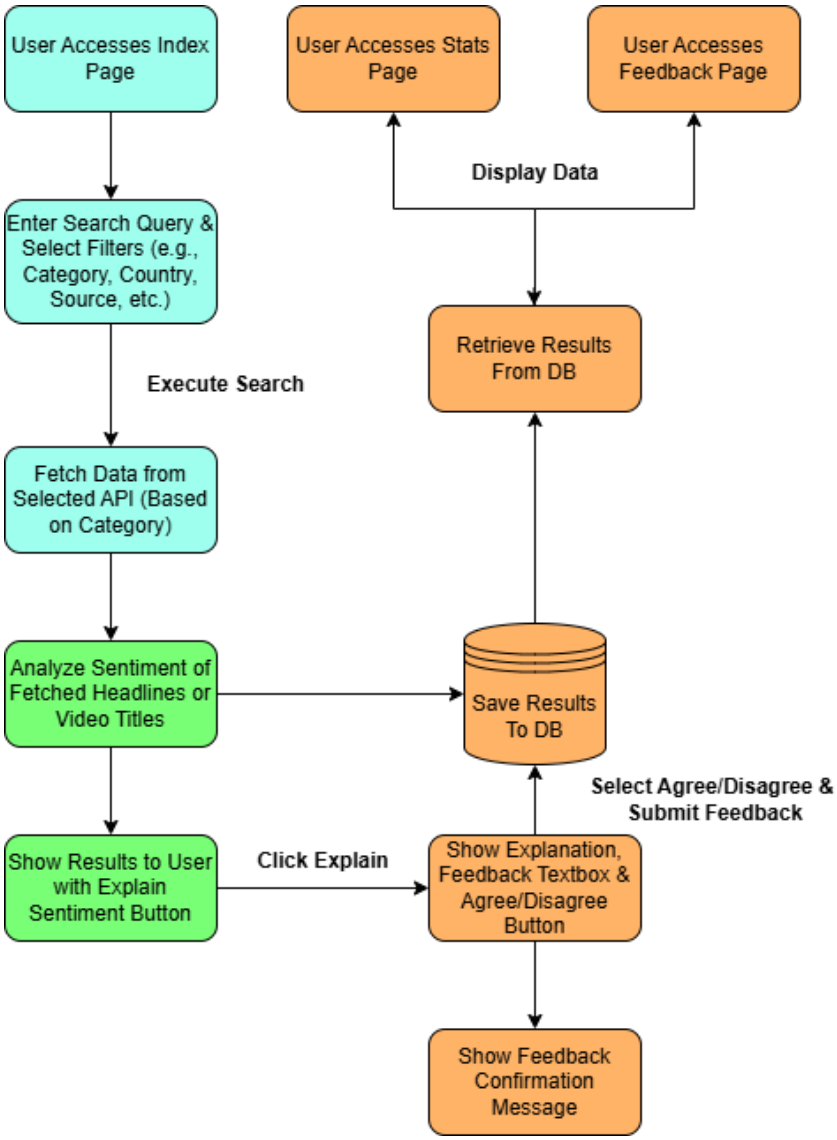


Figure 12: Revised Data Flow Diagram

4.16.2 Architecture and Project Structure

This iteration introduced several structural improvements, including a new models.py file to define database models for storing user search queries, the AI model used, the analysed headlines/video titles, and sentiment data (F9). The database approach enhances reliability by replacing in-memory storage with persistent data handling. A model_feedback.py file was also added to manage user interactions with sentiment analysis results, allowing users to submit and view feedback (F10 & F13), (Figures 13 & 14). This feature supports transparency and potential future model fine-tuning (F11). A statistics page was added (Figure 15) to dynamically display sentiment data based on organizations and selected LLMs (F8 & F9). Using polymorphism, the system handles both video and news content uniformly, ensuring maintainability and scalability (N8).

4.17 Build

4.17.1 Database Selection and Setup

PostgreSQL was chosen for its scalability and complex query support, making it ideal for storing and displaying sentiment data and user feedback (F9 & F13). The database was connected to the Flask backend via SQLAlchemy, with connection settings securely managed (N6). Additionally, PostgreSQL was set up to perform automatic database backups (F12) to ensure data protection and continuity. Backup processes are scheduled regularly, safeguarding the data and allowing for recovery in case of system failures (Figure 15).

4.17.2 Saving Data to Database

The search functionality was updated to store sentiment data in the database while avoiding duplicated records. Logic was added to detect repeated combinations of titles or headlines and their sources, ensuring each sentiment is recorded only once. This prevents inflated results and improves the reliability of sentiment statistics (F8 & F9).

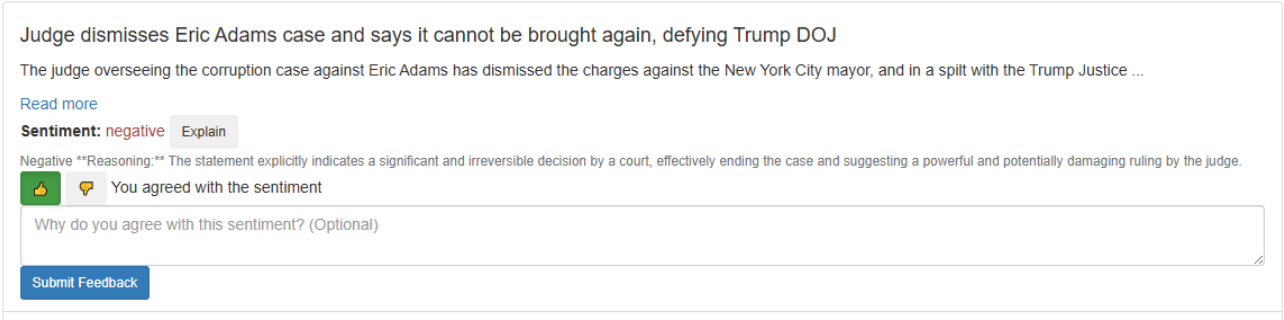


Figure 13: Figure 13: Iteration 3 – Submit Feedback UI

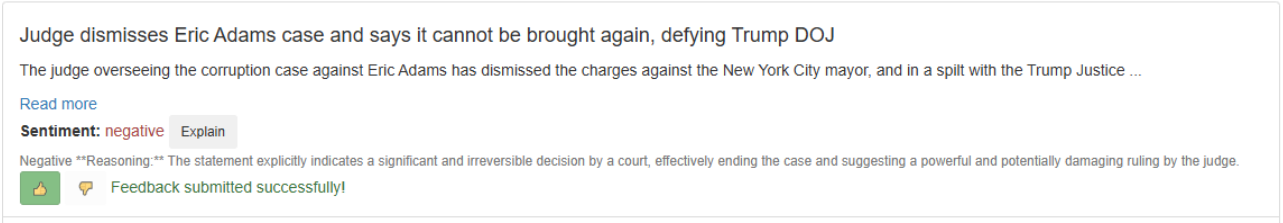


Figure 14: Iteration 3 – Submit Feedback UI

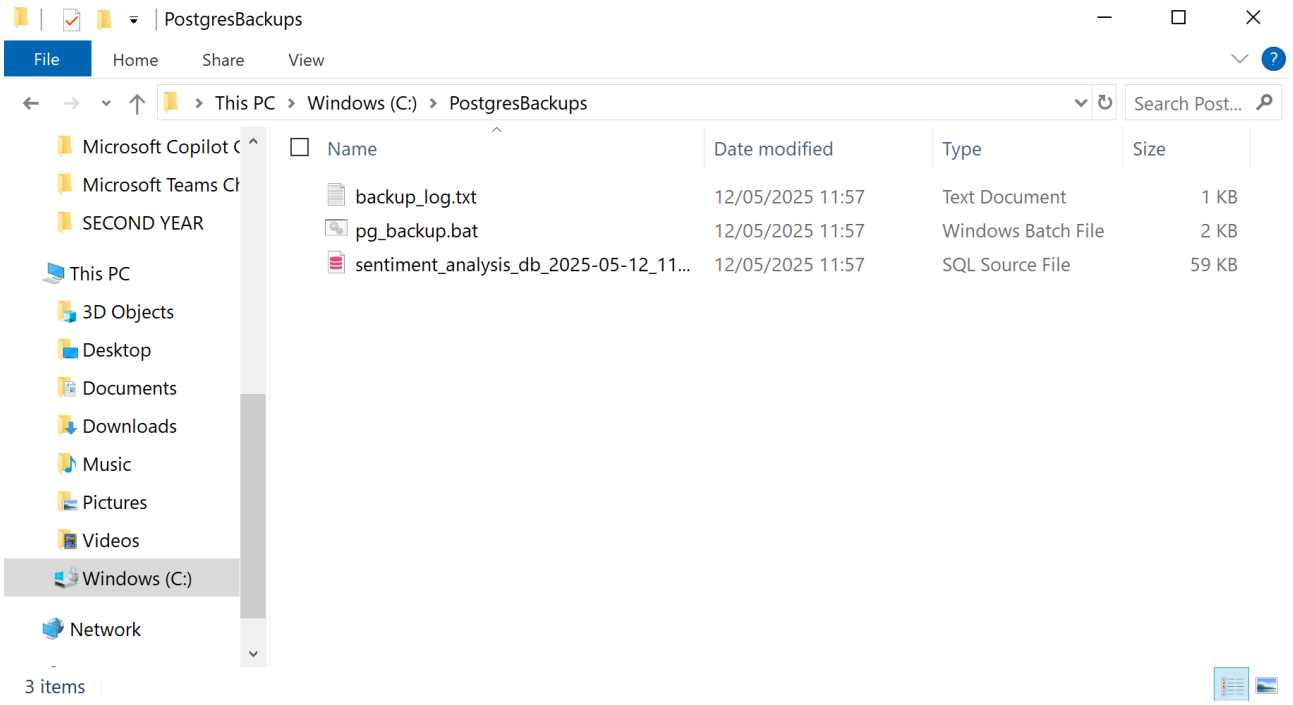


Figure 15: Iteration 3 – Automatic Daily Backups

4.17.3 Displaying Data

A basic HTML table was implemented on the /stats page to render sentiment results, while the /model_feedback page was used to display a table of user feedback (Figures 16 & 17). Though minimalistic, this setup provides a foundation for future enhancements, such as advanced analytics and visual trend representation.

Sentiment Results

Query	Title	Sentiment	Source	Category
trump	Fear and resignation after 'world's most powerful company' pays Trump a $100 billion 'protection fee'	negative	CNN	online news
trump	Judge dismisses Eric Adams case and says it cannot be brought again, defying Trump DOJ	negative	CNN	online news
trump	Trump said Russia had 'all the cards,' but this ceasefire proposal just called Putin's bluff	negative	CNN	online news
trump	Federal Judge Puts Hold On Trump's Shutdown Of Radio Free Europe	neutral	CNN	online news
trump	Fact check: Trump exaggerates trade deficits, his 2024 vote total, Ukraine aid, border crossings and fentanyl deaths	neutral	CNN	online news
trump	TRUMP GAMBLES WITH AMERICANS' FINANCES	negative	CNN	online news
trump	Zambia removes US-based players from women's squad due to 'additional travel measures' under Trump administration	neutral	CNN	online news
trump	Wyatt Hendrickson stuns Olympic champion Gable Steveson to win wrestling national title, celebrates with President Trump	positive	CNN	online news
trump	George Foreman remembered as a 'great fighter' by President Trump and a 'knockout artist' by Magic Johnson	neutral	CNN	online news
trump	Voice of America channels fall silent as Trump administration guts agency and cancels contracts | CNN Business	negative	CNN	online news
starmer	Don't let Putin 'play games' over Ukraine ceasefire, says UK's Starmer before key summit | CNN	neutral	CNN	online news

Figure 16: Iteration 3 – Sentiment Results UI (/Stats Page)

Raw Feedback Data

Article ID	Title	Sentiment	Feedback Type	Feedback Text	Timestamp
-7978435640334629000	'Truly a moron': Musk escalates spat with Trump trade adviser as he divides with White House over tariffs	negative	thumbs_up		2025-04-10 11:17:03

Figure 17: Iteration 3 – Feedback Data UI

4.18 Addressing issues

Table 11 highlights the issues discovered during iteration two with the corresponding solution implementations:

Issue ID	Solution Implementation
IV	Sequential content loading was implemented using JavaScript to disable the search button until all content is fully loaded Figure 18). This prevents users from repeatedly clicking the button, thereby reducing uncontrolled API calls and improving system performance.
V	The "Explain" button was removed from the HTML once clicked, preventing multiple triggers of the LLM request in quick succession (Figure 19). This enforces a one-explanation-per-result policy, ensuring consistency in sentiment analysis.
VI	The LLM prompt was refined to better align the sentiment label and the generated explanation (Figure 19). This ensures that the explanation supports the sentiment, providing clearer and more reliable feedback for users.
VII	A cancellation feature was introduced, allowing users to halt an ongoing search (Figure 18). This ensures that overlapping requests are prevented, reducing system load and avoiding unexpected behaviour.

Table 11: Iteration 3 – Identified Issues & Implemented Solutions

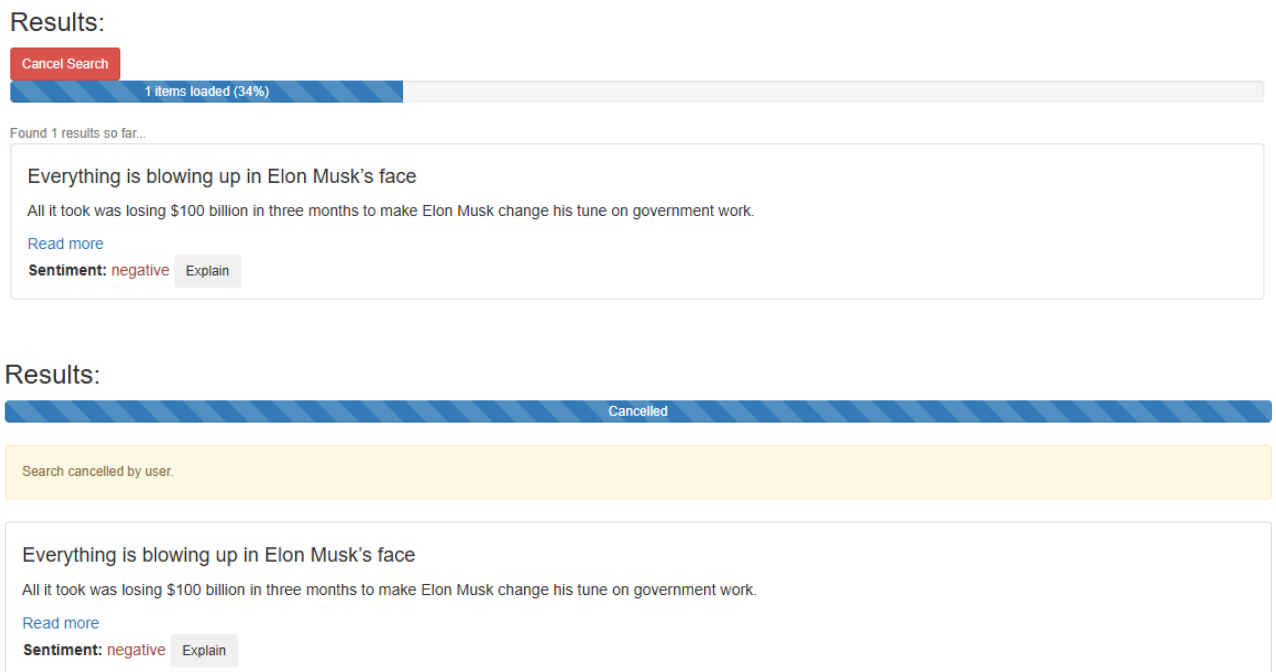


Figure 18: Iteration 3 – UI Improvements

Figure 19: Iteration 3 – Improved Explanation response

4.19 Testing

The testing strategy was carried on during this iteration to test how the application responds to various edge cases. Again, this process uncovered several issues and defects, which have been documented in table 12 and will be addressed in the upcoming iteration.

4.20 Identified issues

Issue ID	Issue Description	Planned Solution
VIII	Multiple users are unable to provide feedback on the same title/headline. Once feedback is submitted for a title, no additional feedback can be added by other users.	New User table cross referenced with the Feedback table to allow multiple users to provide feedback on titles with the complements of Googles login
IX	The current UI is overly simplistic and lacks visual consistency.	Focus will be on creating a consistent and aesthetically pleasing UI across the application
X	In addition to Issue IX, statistical data is neither calculated accurately nor displayed in a visually engaging way.	Calculate sentiment metrics and integrate visualisation tools such as Three.js to present statistics more effectively.

Table 12: Iteration 3 – Issue Table

4.21 Iteration Retrospect

This iteration successfully addressed several key usability and functionality issues from the previous phase while implementing important new high priority requirements (Table 10 & 12). The updated flow enhances the overall user experience and introduces new functionality that keeps users informed and engaged. Iteration four will focus on resolving the issues identified in this phase by introducing additional features based on newly defined requirements to meet the system's evolving needs.

4.22 Development iteration 4

Iteration 4 will focus on implementing new key requirements, including a user login system and feedback management. Google authentication (F14) will streamline and secure the login process, while the User table (F15), linked to the Feedback table (F16), will enable proper association of feedback with individual users. Privacy and Terms of Service pages will also be created to ensure GDPR compliance (F17). The iteration will prioritize security and transparency by adhering to industry standards and ensuring sensitive credentials are not stored (N6, N9 & N10). Additionally, non-functional improvements, such as a responsive layout and a more intuitive interface (N1-N5), will set the foundation for a significant UI overhaul in future iterations.

4.23 Requirements and User Stories

Table 13 consolidates the newly defined requirements, focusing on enhancing the user experience and ensuring the successful delivery of technical and usability improvements.

Priority	ID	Requirement Description	User Story	Acceptance Criteria
Must Have	**F14**	Implement Google authentication	As a user, I want to login with Google so that I don't need to create new credentials	Users can sign in using their Google accounts
Must Have	**F15**	Create User database table	As a developer, I want to store user information so that I can link users to their feedback	System stores user information in a database table
Must Have	**F16**	Link User table to Feedback table	As a developer, I want to associate feedback with users so that multiple users can provide feedback on the same content	System maintains relationships between users and their feedback
Must Have	**F17**	GDPR compliance	As a user, I want my data to be handled according to GDPR so that my privacy is protected	System includes privacy policy and terms of service
Should Have	**F18**	Admin interface for system monitoring	As an administrator, I want to monitor system usage so that I can troubleshoot issues	System provides admin dashboard with usage statistics
Must Have	**N9**	Secure credential handling	As a user, I want my credentials to be secure so that my information is protected	System avoids storing sensitive credentials
Must Have	**N10**	Industry standard security practices	As a user, I want industry standard security so that I feel confident using the system	System follows security best practices

Table 13: Iteration 4 – Requirements and User Stories

4.24 Design

Figure 20 presents a revised data flow diagram, building upon iteration three (Figure 12), updated to show how the new functionality integrates with the existing architecture. Developing this diagram has been crucial for visualizing the system's new flow.

4.24.1 Revised Data Flow Diagram

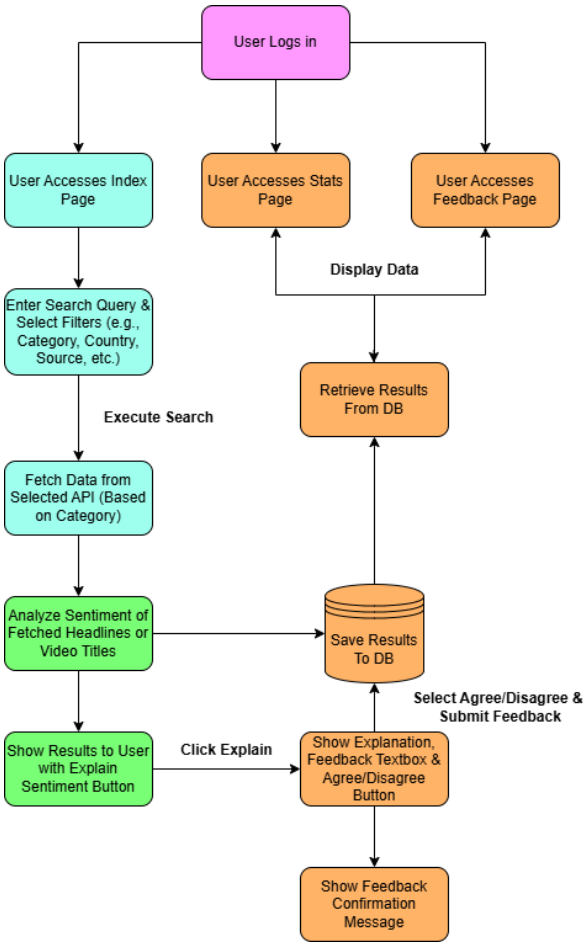


Figure 20: Revised Data Flow Diagram

4.24.2 Database Architecture & Project Structure

To align with the project structure, a User model was defined in models.py, alongside existing models. A one-to-many relationship was established between the User and Feedback models, allowing users to submit multiple feedback entries for different sentiment analysis results (F15 & F16). This addresses *Issue VIII* (Table 12), where the full solution implementation can be found in Table 14.

4.25 Build

4.25.1 Implementing New Flow with Google login

The integration of Google Login (F14) introduced a streamlined authentication flow, starting with the creation of an auth.py file to manage user login and logout using OAuth 2.0 (Figure 21-23). This setup eliminated the need for password handling or storage, enhancing user privacy and data protection (F17, N6, N9 & N10), (Figures 24 & 25).

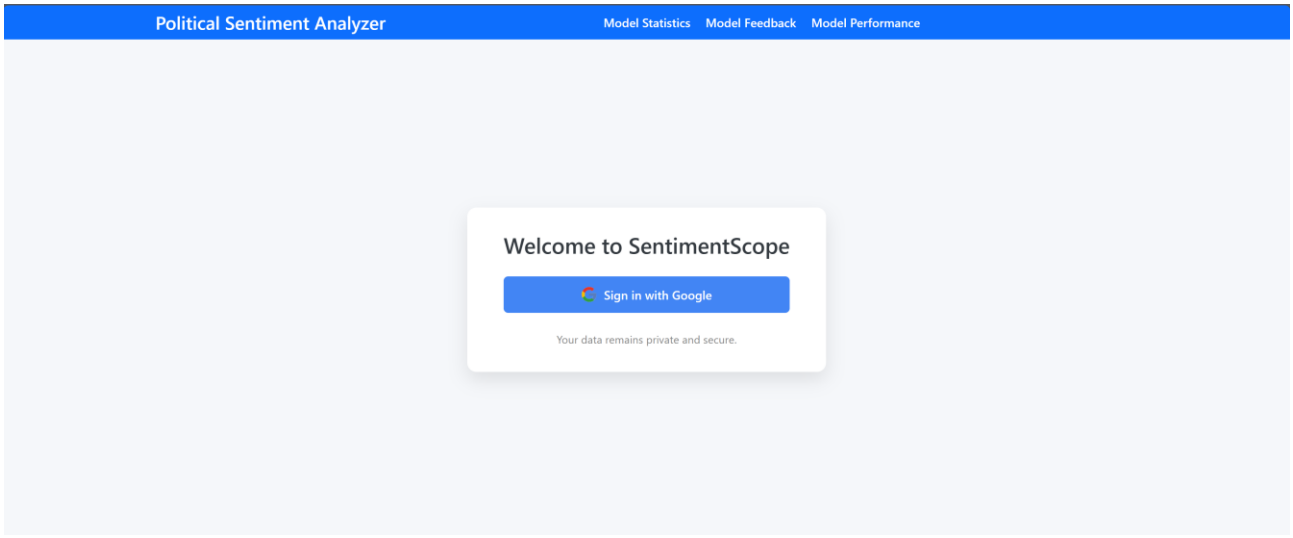


Figure 21: Iteration 4 – New Login Index

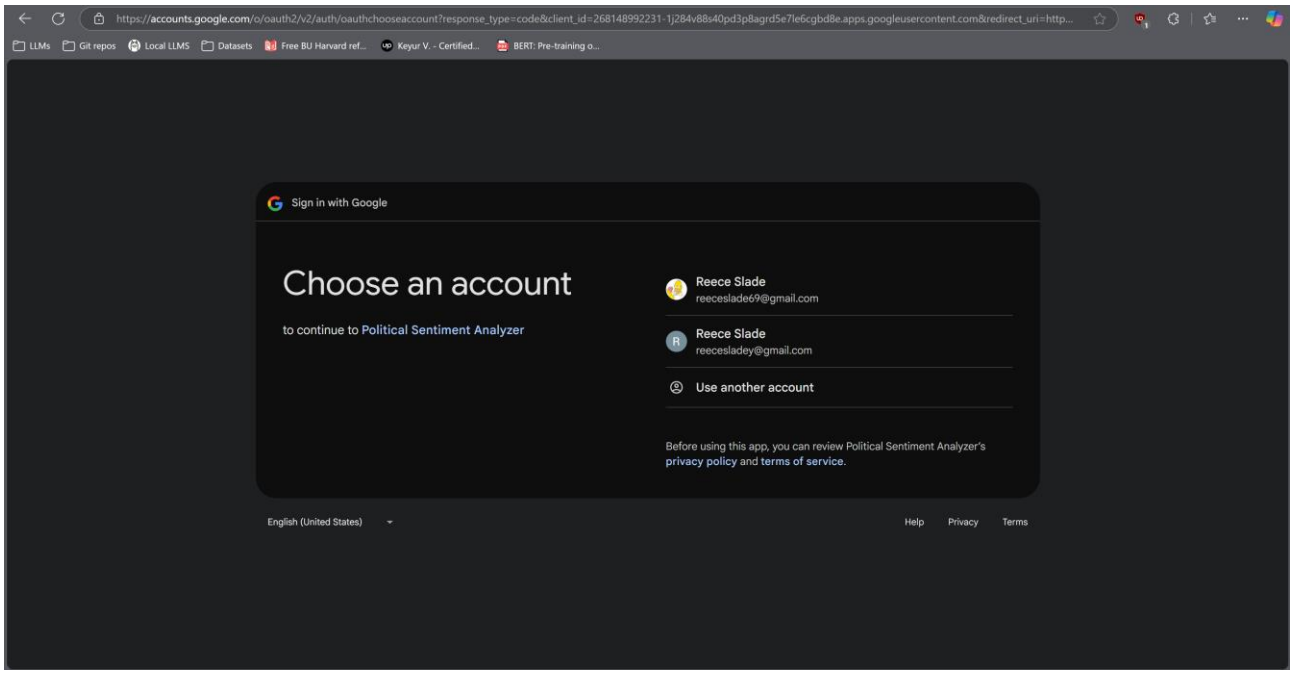


Figure 22: Iteration 4 – Choose an Account Screen

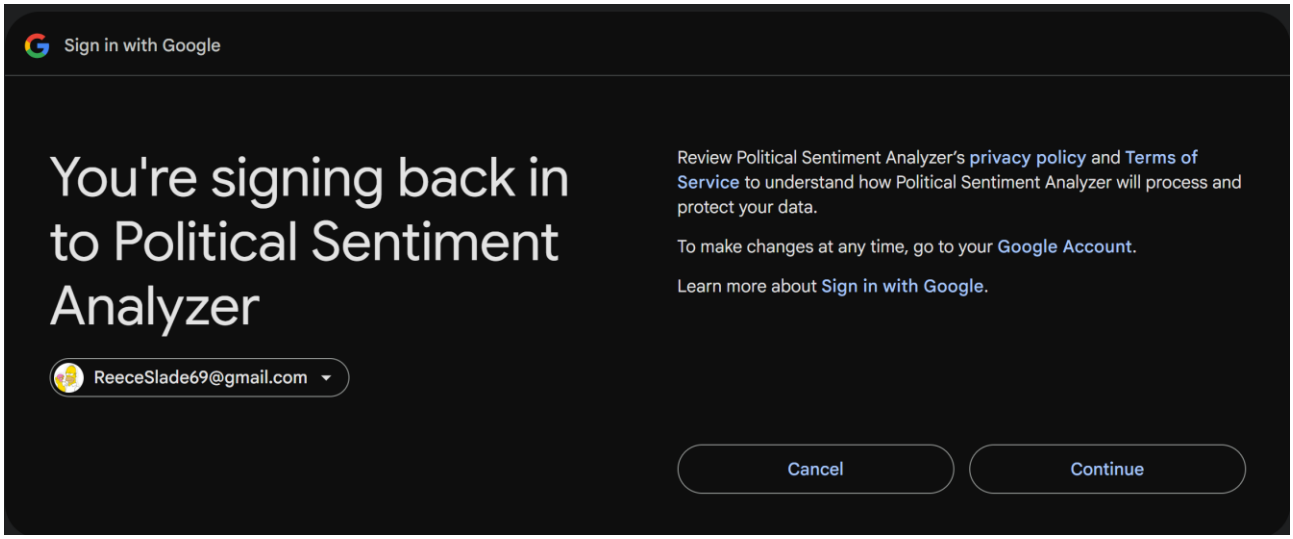


Figure 23: Iteration 4 – Consent Screen

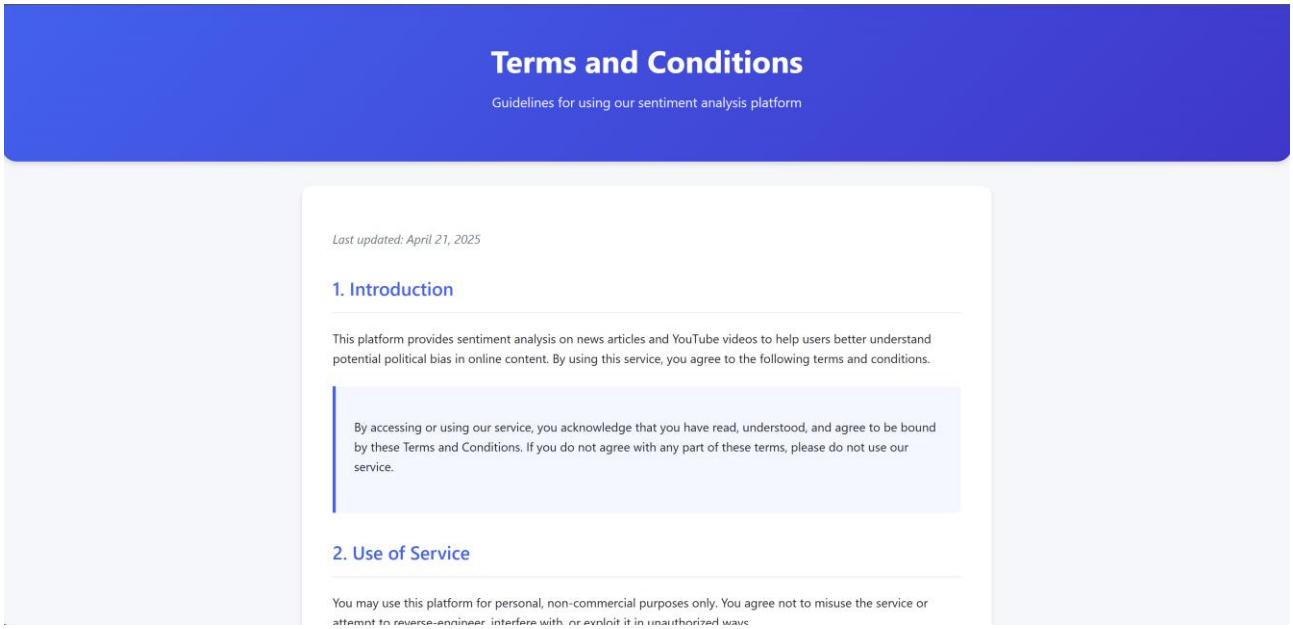


Figure 24: Iteration 4 – Terms and Conditions Page

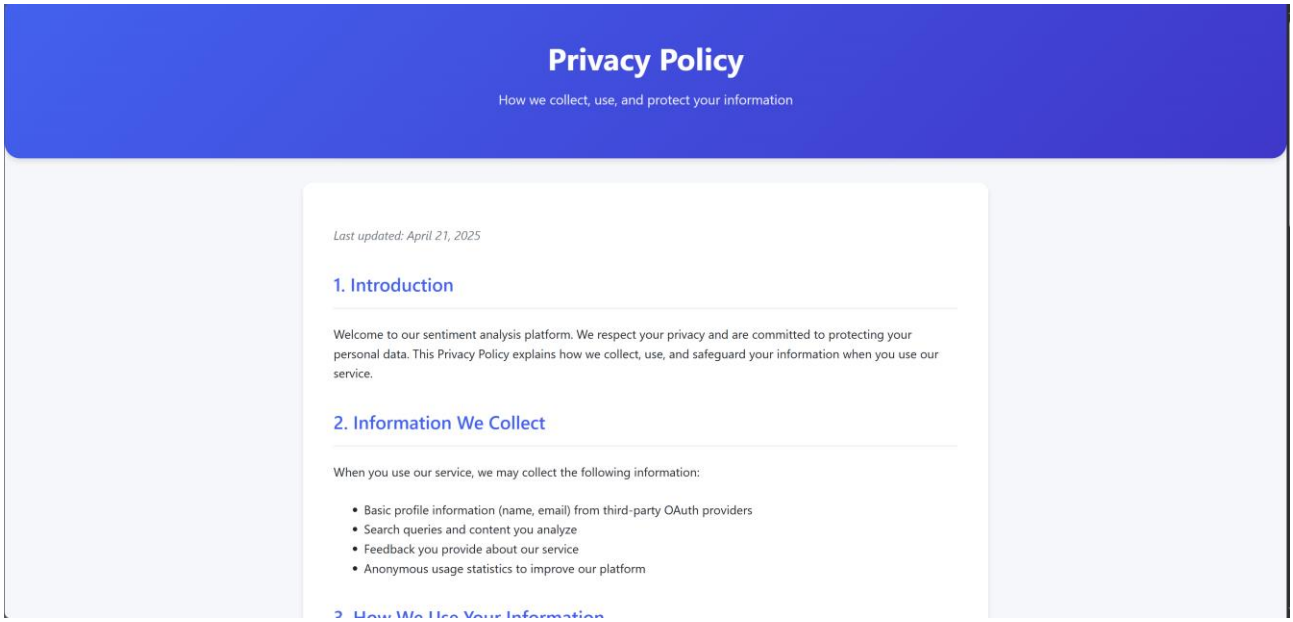


Figure 25: Iteration 4 – Privacy Policy Page

To enhance system monitoring and maintainability, Flask-Admin was integrated (F18), providing privileged users with access to key database models for oversight and troubleshooting. It also supports secure routing by restricting access to specific pages, ensuring only authenticated users can view certain areas of the application (Figure 26).

Sentiment Analysis Admin | Home | Feedback | Sentiment Results | Return to Site

List (11) | Create | Add Filter | With selected | Search: item_title, feedback_text, source_type, model_used

		Id	User Email	Item Id	Item Title	Source Type	Predicted Sentiment	Feedback Type	Feedback Text	Timestamp	Model Used
☐	✎ 🗑	14	reeceslade y@gmail.com	-7978435640334629000	'Truly a moron': Musk escalates spat with Trump trade adviser as he divides with White House over tariffs	news	negative	thumbs_up		2025-04-10 11:17:03.155890	gemma3:1b
☐	✎ 🗑	15	reeceslade69@gmail.com	k4VfaYi_vJY	Fareed's Take: Trump's tariffs upended a booming economy	video	neutral	thumbs_up		2025-04-21 14:34:49.164748	gemma3:1b
☐	✎ 🗑	16	reeceslade69@gmail.com	3653408169392692700	Nigel Farage says men make sacrifices many women don't for jobs	news	negative	thumbs_up	because i do	2025-04-24 23:19:14.566422	gemma3:1b

Figure 26: Iteration 4 – Flask-Admin Page

With Google Login, users' email addresses are now stored and linked to their feedback entries, visible on the newly developed /model_feedback page (F13), (Figure 27). This feature enhances transparency by showcasing public contributions and fostering community engagement allowing

users to explore varying perspectives on sentiment classifications from different language models (section 1.2).

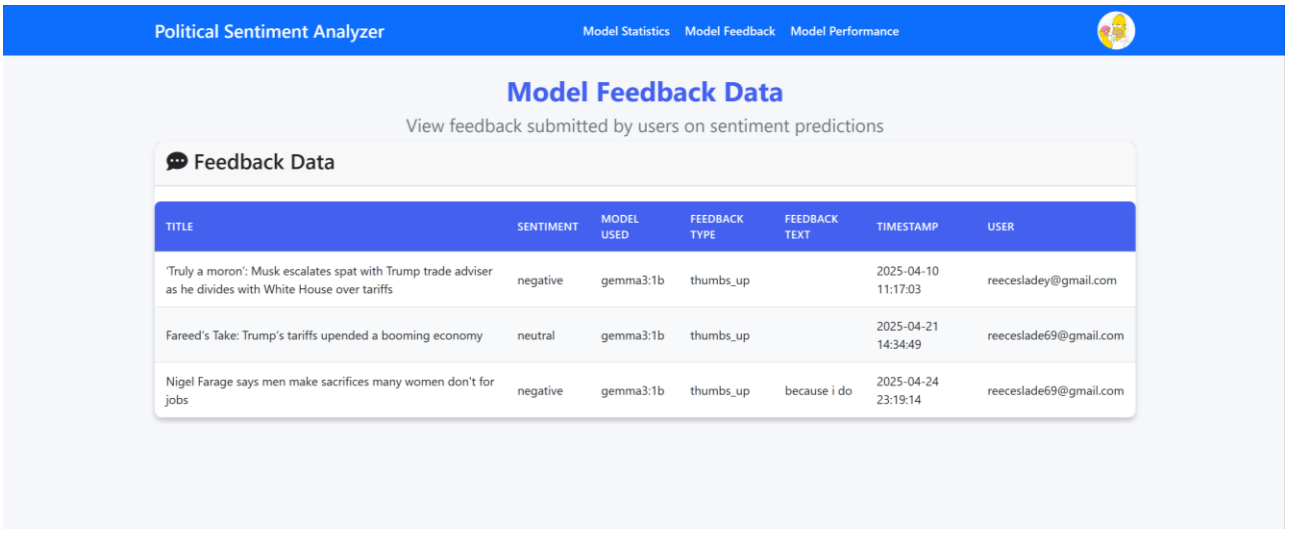


Figure 27: Iteration 4 – New Model Feedback Data Page

4.25.2 New Statistics Page

In addition to the existing /stats page, a new page titled /model_performance was implemented (Figure 28). This page was designed to provide a more granular view of user feedback by splitting sentiment evaluation into three distinct metrics: thumbs up, thumbs down, and total feedback. Users can select a specific LLM model to view its performance based on user agreement levels. This feature supports the project's broader aims and objectives by helping users understand potential limitations or inconsistencies in different language models' sentiment analysis.

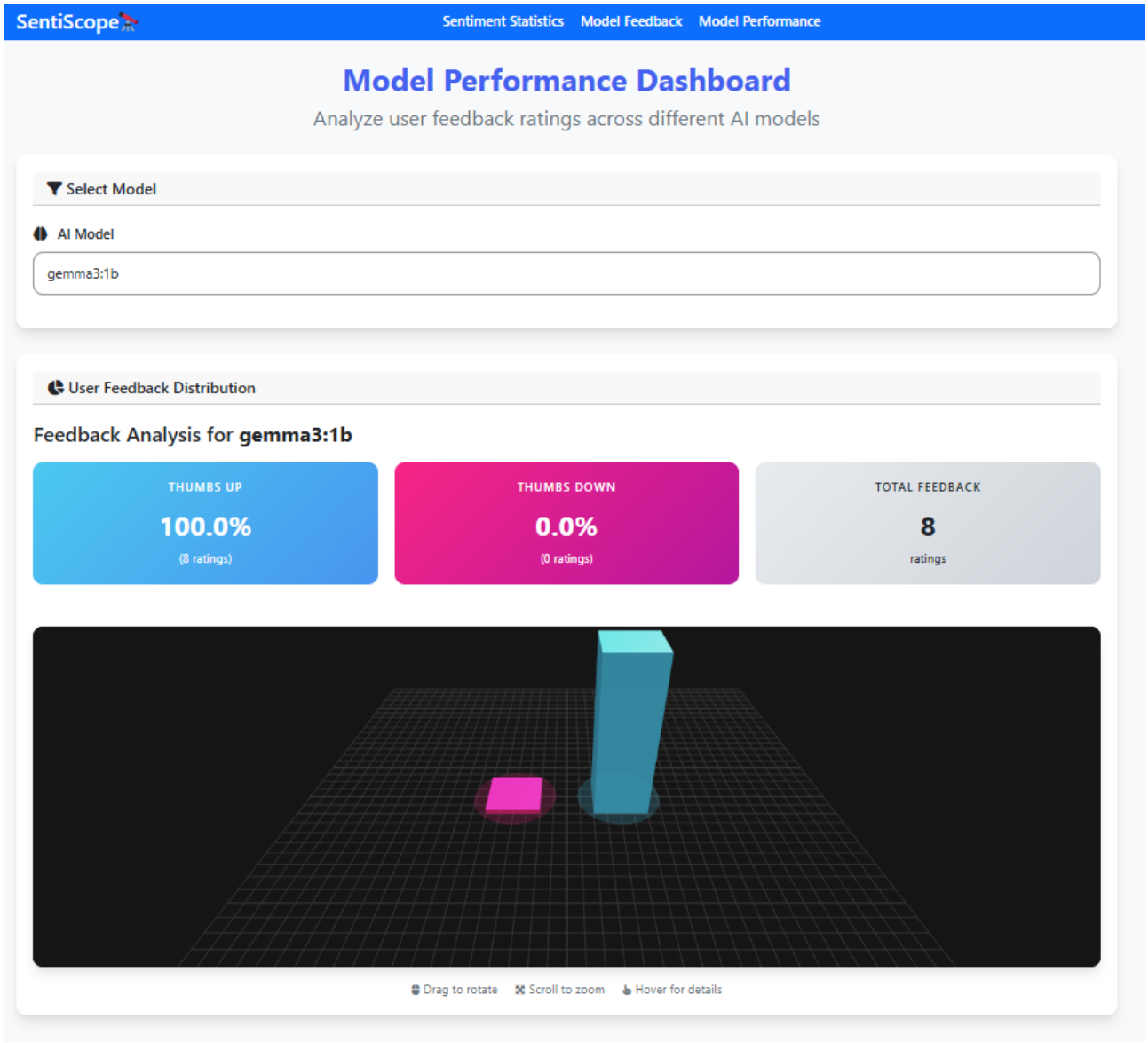


Figure 28: Iteration 4 – Model Performance Page

4.26 Addressing issues

In Iteration 4, key issues from the previous phase were addressed, focusing on improving usability, UI consistency, and system security. Table 14 demonstrates a summary of the implemented solutions for each issue.

Issue ID	Solution Implementation
VIII	A new User table was created and cross-referenced with the Feedback table to enable multiple users to provide feedback on the same title/headline. The combination of user_id and item_title was constrained to ensure users can only provide feedback once per unique result. Google login (OAuth 2.0) ensures secure user identification and linkage to feedback entries (F14-F17). This maintains data integrity and allows participation across multiple headlines or items. Each feedback entry is associated with its user via a user_id foreign key, ensuring all submissions are trackable and attributable.
IX	A consistent UI layout was introduced using a base.html template with shared headers and footers. Page-specific content is injected dynamically, with styling handled by both individual and global CSS files. This ensured a cohesive visual identity and seamless user experience across the application (N1-N5). All figures presented this iteration showcase the improved UI, addressing the issue of the overly simplistic UI (Figures 21-32).
X	Three.js was integrated to visualize statistical data from the SentimentResults table, replacing static visualization tools like matplotlib. This allowed for dynamic, interactive 3D visualizations of sentiment data, improving the user experience and making statistical data more engaging. Additionally, statistical calculations were refined to display accurate percentages, ensuring the information presented is both visually compelling and reliable (N1-N5, F8). Figures 28 & 31 demonstrate the updated statistic pages, showing how the 3D visualization of the data enhances the presentation of sentiment analysis results and model performance.

Table 14: Iteration 4 – Identified Issues & Implemented Solutions

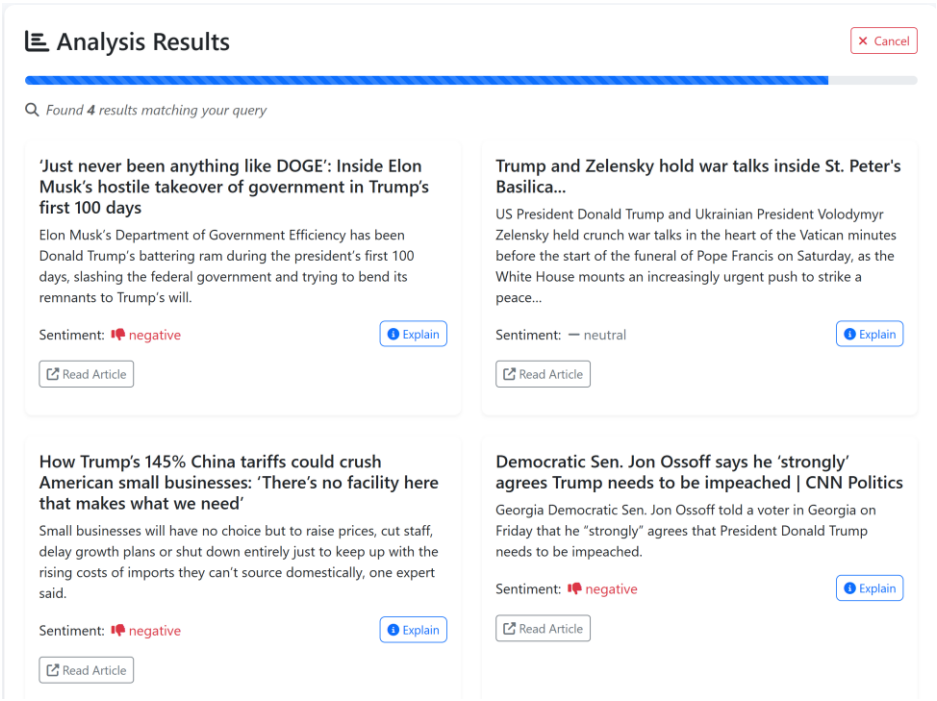


Figure 29: Iteration 4 – News Article Results

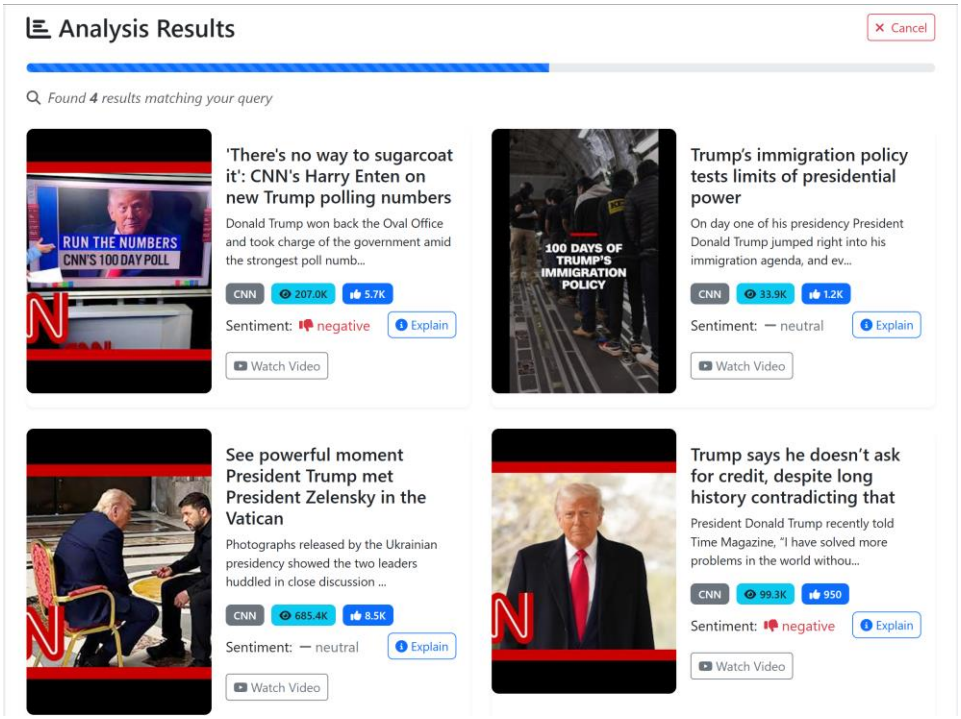


Figure 30: Iteration 4 – New Video Results

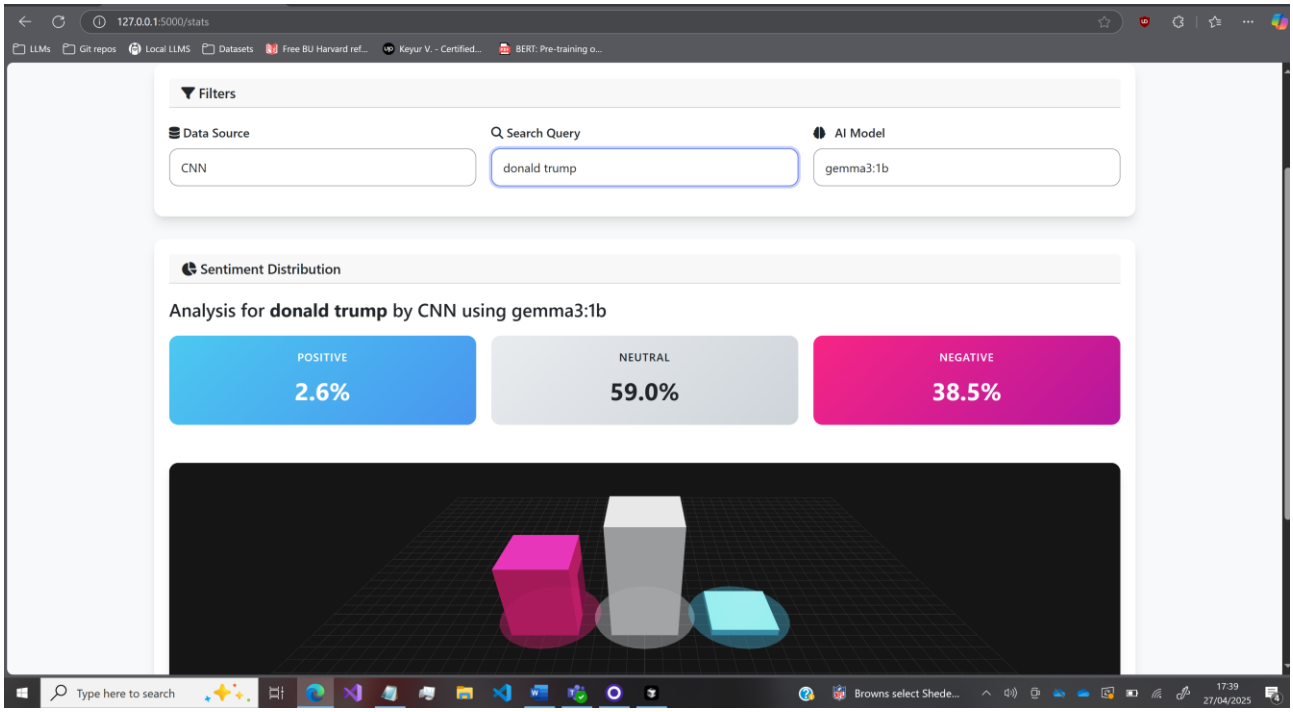


Figure 31: Iteration 4 – Organisational Statistics Page

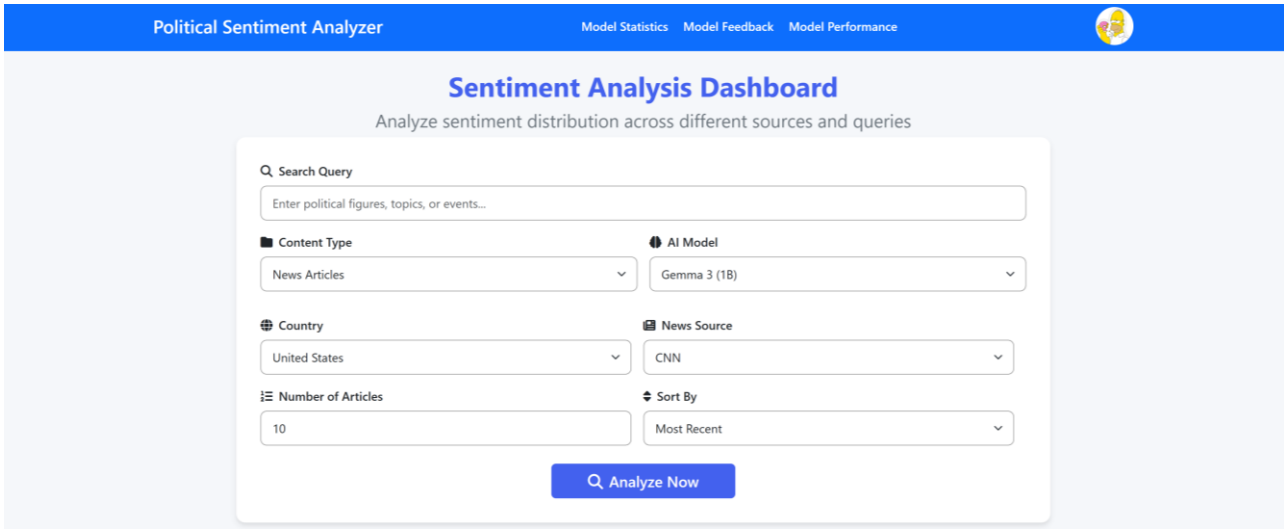


Figure 32: Iteration 4 – New Index Page

4.27 Iteration Retrospect

No new bugs or defects were identified in this iteration, as no formal testing was conducted. This was due to most core functional and non-functional requirements already being implemented and previously reported issues resolved. The next and likely final iteration will focus on code refactoring and comprehensive testing to minimize any remaining defects before submission. If time permits, lower-priority or out-of-scope requirements may also be considered for inclusion.

4.28 Development Iteration 5

The fifth and final iteration focused on the comprehensive evaluation, testing, and refinement of the artefact. With all core features implemented, this stage prioritised usability (N1-N5), functional robustness, and overall reliability, ensuring the system was free from major defects and supported by maintainable, modular code (N7). No new features or DFDs were introduced; instead, attention turned to resolving identified issues and simulating real-world usage through exploratory and ad hoc testing.

Testing scenarios included logging in and out via Google, registering new users, searching political content, and reviewing sentiment classifications. The sentiment statistics interface and feedback submission system were also rigorously evaluated. These tests aimed to ensure all critical workflows functioned reliably from an end-user perspective and under varied conditions.

Tests were performed across multiple screen sizes and modern browsers to ensure cross-platform consistency and responsiveness. Key outcomes are summarised in tables 15 & 16:

Screen Size	Requirements	Notes
Small (360x800, 390x844, 414x896)	N1-N5	The application works fully as intended. All inputs, dropdowns and buttons are easy to read in the correct format. One limitation could be the dropdown options stretching across the whole screen when selected.
Medium (768x1024, 810x1080, 1200x800)	N1-N5	The application works fully as intended. All inputs, dropdowns and buttons are easy to read in the correct format. When less than 768px in width, the links in the header are stored in a hamburger.
Large (1920x1080, 1366x768, 1536x864)	N1-N5	The application works fully as intended. All text, inputs and buttons are easy to read in the correct format.

Table 15: Display Size Testing Results

Browser	Requirements	Notes
Chrome	N1-N5	The application works fully as intended and is responsive.
Edge	N1-N5	The application works fully as intended and is responsive.
Firefox	N1-N5	The application works fully as intended and is responsive.

Table 16: Browser Compatibility Testing Results

Following this evaluation, several issues were identified and addressed, as detailed in Sections 4.29 & 4.30.

4.29 Identified issues

Table 17 outlines the issues discovered during this testing-based iteration, alongside their corresponding planned solutions:

Issue ID	Issue Description	Planned Solution
XI	Uninformative and inconsistent error messages	Implement JavaScript to display clear and context-specific error messages for different conditions.
XII	Application shows analysed results but fewer than the number of articles or videos requested	Inform users about the strictness of search filters and possible reasons for limited results.
XIII	Inconsistent page titles and navigation labels	Rename and refactor web pages to use appropriate, informative, and consistent naming conventions.
XIV	Insufficient explanation about project purpose and functionality	Create a web page to provide detailed information about the project's goals, scope, and functionality.
XV	Limited number of news sources and LLMs integrated	Expand the system by introducing additional news sources and large language models (LLMs) as planned.
XVI	Content visible even when not logged in	Implement proper authentication checks to ensure that restricted content is only visible to logged-in users.
XVII	Selected model DeepSeek 1.5b does not work whilst all other DeepSeek models do	Exclude DeepSeek 1.5b from model selection.
XVIII	No error message when API quota limits are exceeded.	Inform the users that they have exceeded the maximum number of API requests and to try again in X number of hours.
XIX	When searching for a query, the search is re-initialised even if the query is repeated, leading to longer waiting times.	Retrieve results from cache when submitting a previously searched query instead of initialising a new request.

Table 17: Iteration 5 – Issue Table

4.30 Addressing Issues

Table 18 highlights the issues discovered during this iteration with the corresponding solution implementation:

Issue ID	Solution Implementation
XI & XII	A help button ("?") was implemented to guide users by explaining why strict search filters may result in fewer retrieved articles or videos. Additionally, error messages were enhanced to provide more context, improving overall clarity and user experience (Figures 33 & 34).

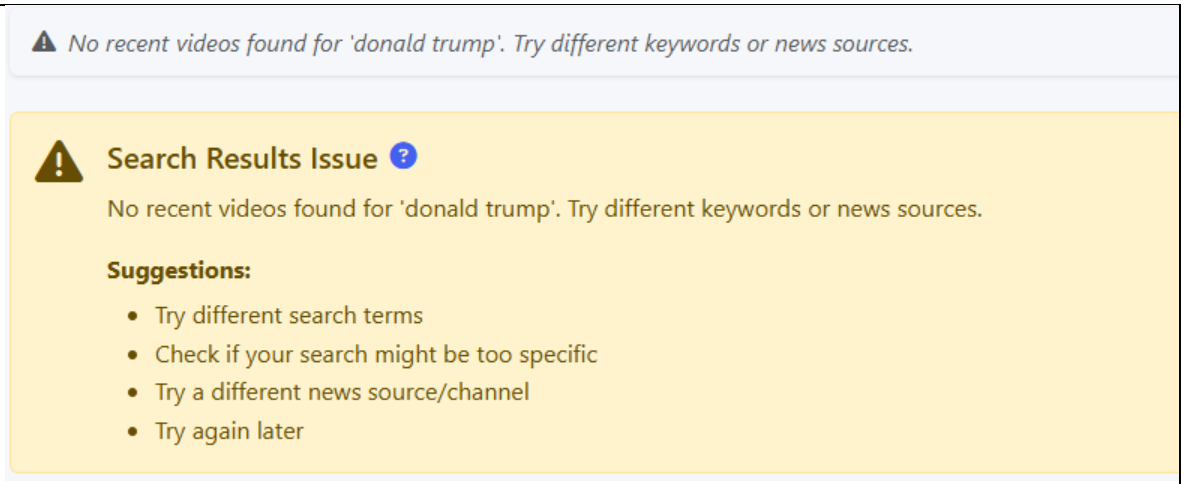


Figure 33: Iteration 5 – Solution for Issue XI

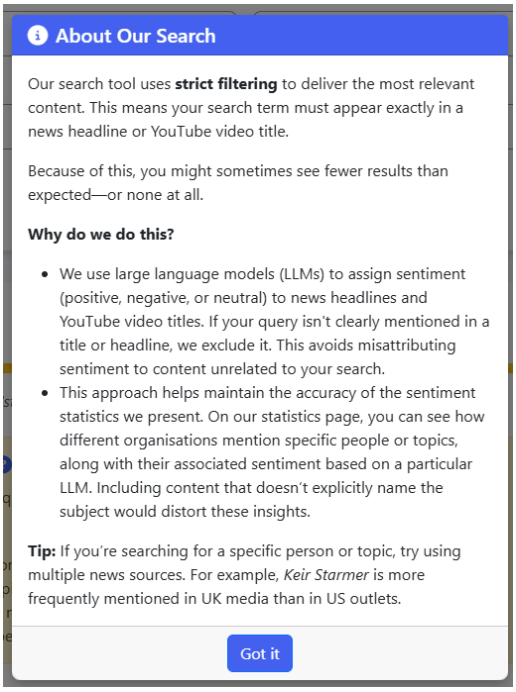


Figure 34: Iteration 5 – Solution for Issue XI & Issue XII

XIII	A code refactor was done to standardize web pages and navigation labels, improving user navigation and UI consistency.
XIV	An About page was created to communicate the project's aims, scope, functionality, and significance, enhancing the onboarding experience for new users (Figure 35).

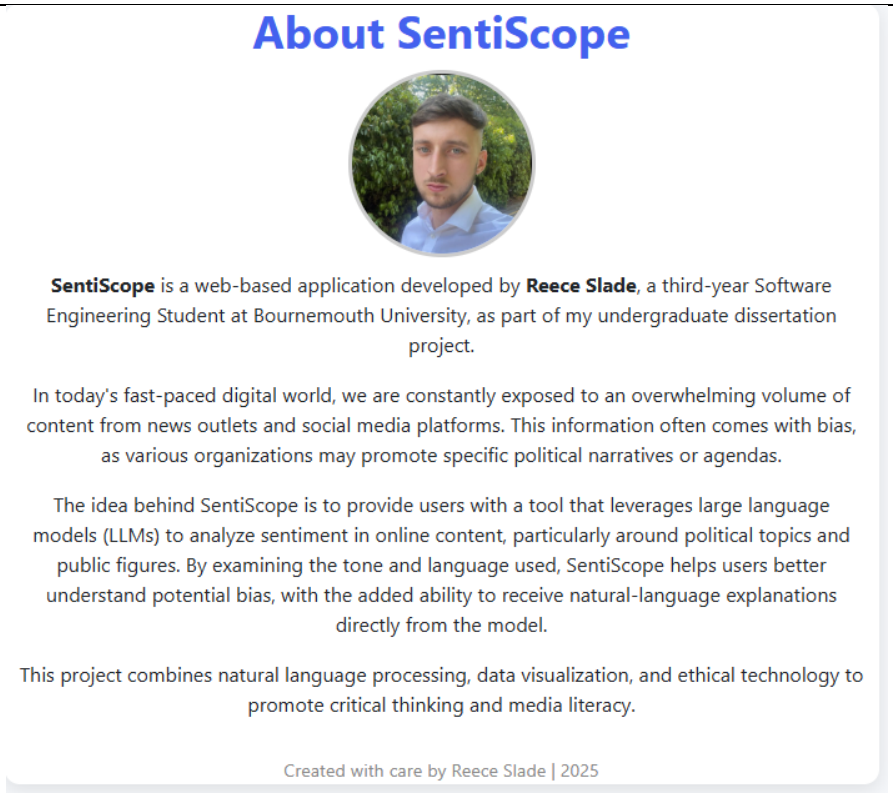


Figure 35: Iteration 5 – New About Page

XV

Based on the insights from Table 4, the following news sources were selected:

News Source	Country
• MSBC (Left) • NewsWeek (Centre) • Fox News (Right)	**US**
• BBC News (Centre) • Daily Mail (Right)	**UK**

These sources were chosen to provide a balanced representation of political leanings, ensuring a diverse set of perspectives in the data collection and sentiment classification process. However, a key limitation is the lack of UK-based left-leaning media sources. This is due to restrictions imposed by NewsAPI, which, under the free tier, does not provide access to such outlets. As a result, only two UK sources were selected based on their availability and relevance to the project. For future work, developers may consider expanding the list of sources and supported countries to increase the system's scope. Additionally, a diverse set of models with varying parameters was integrated to mitigate inherent biases and improve compatibility with a range of devices.

XVI

Authentication checks were reinforced across the app by applying the is_authenticated method to headers and page routes, restricting content to authenticated users only. This ensures that only logged-in users can submit and view model feedback, maintaining accountability and protecting the system from spam or anonymous misuse. By tying feedback to authenticated users, the integrity and reliability of the data collected is significantly improved.

XVII

DeepSeek 1.5b was excluded from model selection due to prompt compatibility issues. While the prompt performed successfully across other models, it consistently failed with DeepSeek 1.5b. Given its relatively low parameter count, excluding the model was deemed more efficient than restructuring the prompts.

XVIII

To handle rate limits for both the News and Youtube APIs, an alert box was introduced to notify users when the limit was reached.

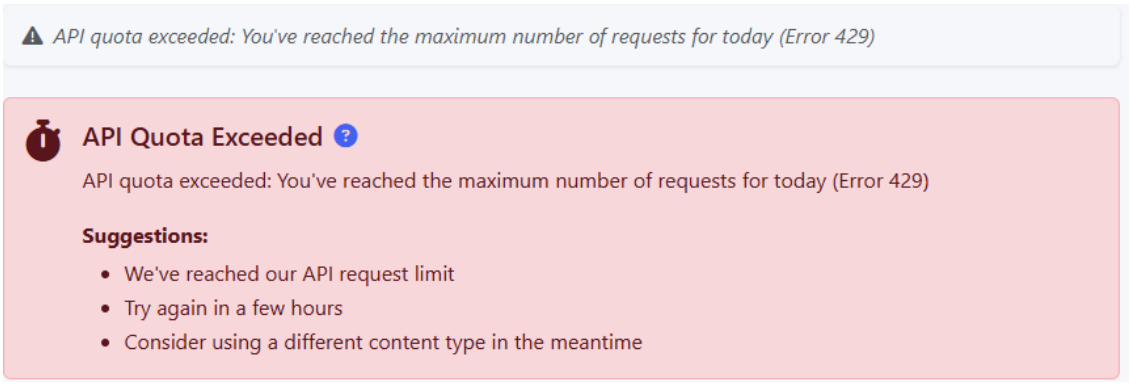
	 **Figure 36: Iteration 5 – Solution for Issue XVIII**
XIX	Caching was added to optimize performance. If a user submits a previously searched query, the system retrieves the result from cache instead of initiating a new request.

Table 18: Iteration 5 – Identified Issues & Implemented Solutions

4.31 Iteration retrospect

Iteration five successfully conducted comprehensive evaluation, testing, and refinement of the artefact. All identified issues, as outlined in Table 17, were successfully resolved through exploratory and ad hoc testing from an end user's perspective (Table 18). This iteration played a vital role in improving the system's performance, stability, and maintainability. The focus on thorough evaluation also ensured the artefact met usability standards and was aligned with the project's goals, contributing to a cleaner, functional, more robust and user-friendly application.

5 EVALUTATION OF ARTEFACT

5.1.1 Usability Evaluation

The artefact's usability was shaped by established principles, primarily Nielsen's heuristics *(Nielsen, 2012)* and the three-click navigation convention *(Jiménez Iglesias et al., 2018).* These frameworks guided interface evaluation to ensure intuitive interactions during political sentiment analysis. Further evaluation included structured heuristic analysis and developer-led exploratory testing. The artefact demonstrated strong alignment with usability best practices, including consistency, error prevention, and aesthetic minimalist design.

Simulated first-time user interactions proved that all core functions were accessible within three clicks, adhering to the three-click rule. Cognitive load was minimized through consistent visual hierarchy, logical grouping of elements, and uniform layout patterns. Persistent navigation and clearly labelled elements enhanced discoverability and user orientation throughout the application.

However, the absence of user testing limits the generalisability of these findings, as usability issues unnoticed by the developer may only emerge in real-world use. Additionally, the lack of quantitative metrics (e.g., task completion times, error rates) weakens the empirical basis of the analysis.

To improve evaluation rigour, future iterations should incorporate controlled usability studies with target users, such as political and computer science students. The integration of interaction analytics and beta testing would further uncover user behaviour trends and inform refinements to navigation and layout.

5.1.2 Evaluation of Sources and LLMs

The artefact successfully fulfils the project's functional objectives by integrating a diverse array of news sources and LLMs for sentiment analysis. This implementation allows users to compare sentiment across multiple perspectives, fostering more critical and balanced political understanding (section 1.2).

The inclusion of sentiment accuracy and model performance statistics, as well as user-driven feedback mechanisms enables transparent evaluation of both the media sources and the LLM-generated outputs. This dual-layered assessment supports the identification of potential bias in news sources and LLMs, strengthening trust in the sentiment classification process.

5.1.3 Satisfaction

The artefact effectively supports informed and independent political discourse through its core features. By allowing users to compare politically relevant content from various sources and review real-time sentiment classifications from various LLMs, the system enhances user awareness of media/model bias and promotes critical thinking.

Its transparent design, where users can evaluate the sentiment analysis process and its sources, reflects the project's broader aim of contributing to media literacy and responsible political engagement. While qualitative satisfaction has been inferred from internal testing, future work should include structured user feedback to validate perceived value and usability in real-world contexts.

5.1.4 Requirement Evaluation

All requirements classified as 'Must Have' and 'Should Have' under the MoSCoW prioritisation method were successfully implemented in the final artefact. These categories represent the minimum criteria necessary for the project to be deemed successful, indicating that development proceeded in alignment with the original plan. A complete evaluation of requirements is provided in Appendix D.

6 CONCLUSION

6.1 Objective Analysis

Objective	Evaluation
1. Conduct comprehensive research into the capabilities of LLMs and apply them to analyse sentiment in online political discourse.	The capabilities of several LLMs were explored through literature review and applied directly to political sentiment analysis. Models were tested on real-world headlines and YouTube titles, providing generally accurate sentiment predictions.
2. Implement and document best practices to ensure reliable and accurate sentiment assignment using LLMs.	Best practices were researched and implemented such as prompt engineering techniques. These ensured sentiment assignment remained consistent, coherent, and accurate across models and sources.
3. Employ software engineering best practices, including modular architecture, version control, and testing strategies.	The project employed modular architecture via Flask, Git-based version control, and testing strategies. These practices supported clean, maintainable code and a scalable foundation for future development.
4. Integrate real-time political discourse data using third-party APIs.	YouTube and News APIs were successfully integrated, with dynamic querying based on user input. This enabled real-time retrieval of political discourse and enhanced the relevance of sentiment outputs.
5. Identify and prioritise functional and non-functional requirements, refining scope iteratively through corresponding user stories and acceptance criteria.	All major milestones were achieved on schedule, supported by progress tracking tools (Appendix E). Some low-priority, out-of-scope requirements were not implemented. Minor adjustments were made without affecting final delivery, demonstrating strong project planning and execution.
6. Deliver the project according to the agreed timeline and university submission deadlines.	All major milestones were achieved within the scheduled timeframe, supported by progress tracking tools (Appendix E), with the exclusion of some low priority, out of scope requirements not being implemented. Minor adjustments were made without impacting final delivery, demonstrating strong project planning and execution.
7. Develop a responsive, intuitive web interface that visualises LLM-driven sentiment and statistical insights.	The web app was built using Bootstrap and tested across browsers and devices. It features intuitive navigation and clear sentiment visualisations, though further UI enhancements could be explored.

6.2 Appreciation of Objectives

All objectives were successfully met, resulting in a functional, insightful, and user-focused application. The integration of LLMs for sentiment analysis and the presentation of statistics based on the output was a standout achievement, highlighting both the potential and limitations of AI in analysing real-time political discourse.

6.3 Reflection and Lessons Learned

The iterative, agile approach was essential in adapting to evolving requirements. Each cycle introduced new features or refinements that aligned the system with functional and non-functional goals, all while keeping within time constraints. This flexibility helped maintain consistent project momentum and quality.

Exploratory and ad-hoc testing from a user's end perspective offered actionable insights, particularly regarding interface clarity and feature expectations. However, the lack of structured beta testing limited the depth of feedback. A key lesson is to conduct beta testing at the end of the development process to further improve the design and user experience.

On the technical side, working with LLMs revealed both their strengths and limitations. While they performed well in general sentiment classification, future development should focus on fine-tuning models based on real-world feedback. Additionally, integrating alternative LLMs from platforms like Hugging Face will enhance robustness, enabling the creation of a more balanced model driven by diverse user perspectives.

Overall, the process demonstrated the importance of balancing flexibility with structure, and the artefact revealed the potential, and current boundaries, of applying LLMs to politically sensitive data.

6.4 Concluding Remarks

This project demonstrated that large language models can be effectively assign sentiment to real-time of political content. Despite the challenges associated with inherent biases, the system performed reliably and met the intended goals of accuracy, responsiveness, and usability.

While Ollama offered fast local processing and ease of use, future development could explore fine-tuned models on Hugging Face to improve interpretability and robustness (F11). Additionally, expanding the system to support multilingual sentiment analysis, additional countries, or topic filters could increase its scope and usefulness.

Best practices in software development including modular design, versioning, and testing ensured a solid codebase. Legal and ethical considerations were also addressed through transparent policies and data reliance, establishing trust and compliance.

Ultimately, this work lays a strong foundation for further development, demonstrating the viability of combining real-time data with advanced AI to aid understanding of online political discourse.

6.5 Acknowledgement of Generative AI Use

Generative AI tools were utilised during the development of this dissertation to support clarity, conciseness, and grammatical accuracy. Large Language Models (LLMs) such as ChatGPT, Claude, and DeepSeek were employed to assist with refining written content, generating boilerplate code, and debugging implementation issues. These tools were used as supplementary aids, with all critical thinking, analysis, and final decision-making.

6.6 Future Work

To facilitate future development, the artefact offers flexibility by supporting use within a virtual environment or through a Dockerized setup, enabling rapid environment configuration and simplified deployment. Version control is managed with Git, allowing contributors to efficiently propose and review changes via pull requests, which enhances collaborative workflows. Future enhancements may focus on broadening the geographical and topical scope of political content by integrating additional countries, news sources, and language models in a seamless manner. Furthermore, implementing structured user feedback loops could support continuous fine-tuning of sentiment models, improving their accuracy and relevance. Additional improvements might include advanced filtering and visualization tools to enrich user experience, alongside strategies to address API rate limits, such as adopting paid plans, thereby ensuring greater scalability and system robustness. To run the project on a machine or device, refer to Appendix K.

Word count (main body of the report): 6,045

REFERENCES

Hussein, D.M.E.-D.M., 2018. A survey on sentiment analysis challenges. *Journal of King Saud University - Engineering Sciences*, 30(4), pp.330–338. Available at: https://doi.org/10.1016/j.jksues.2016.04.002 [Accessed 14 May 2025].

Joseph, T., 2024. Natural language processing (NLP) for sentiment analysis in social media. *International Journal of Computing and Engineering*, 6(2), pp.35–48. Available at: https://doi.org/10.47941/ijce.2135 [Accessed 14 May 2025].

Zhang, L., Wang, S. and Liu, B., 2018. Deep learning for sentiment analysis: A survey. *arXiv*. Available at: https://arxiv.org/abs/1801.07883 [Accessed 14 Apr. 2025].

Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K., 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint*. Available at: https://arxiv.org/abs/1810.04805 [Accessed 14 Apr. 2025].

Vaniukov, S., 2024. NLP vs LLM: A comprehensive guide to understanding key differences. *Medium*. Available at: https://medium.com/@vaniukov.s/nlp-vs-llm-a-comprehensive-guide-to-understanding-key-differences-0358f6571910 [Accessed 23 Apr. 2025].

Gao, S., Young, M.T., Gounley, J., Tinn, R., Stahlberg, E., Puri, R., Mohan, S., Zhang, Y., & Peng, J., 2021. Limitations of Transformers on Clinical Text Classification. *IEEE Journal of Biomedical and Health Informatics*, 25(9), pp. 3596–3607. Available at: https://doi.org/10.1109/JBHI.2021.3062322 [Accessed 14 May 2025].

Ollama, 2025. Ollama API documentation. Available at: https://github.com/ollama/ollama/blob/main/docs/api.md [Accessed 7 May 2025].

GeeksforGeeks, 2022. Separation of concerns (SoC). Available at: https://www.geeksforgeeks.org/separation-of-concerns-soc/ [Accessed 24 Apr. 2025].

Vadhadia, K., 2023. Designing error messages that enhance user experience. *Medium*. Available at: https://medium.com/design-bootcamp/designing-error-messages-that-enhance-user-experience-1e0cc8d8f5f8 [Accessed 14 Apr. 2025].

Atlassian, 2025. Exploratory testing. Available at: https://www.atlassian.com/continuous-delivery/software-testing/exploratory-testing [Accessed 24 Apr. 2025].

Wikipedia, 2024. Ad hoc testing. Available at: https://en.wikipedia.org/wiki/Ad_hoc_testing [Accessed 24 Apr. 2025].

HypeAuditor, 2024. Top News & Politics YouTube Channels in the United Kingdom. Available at: https://hypeauditor.com/top-youtube-channels/news-politics-category-gb/ [Accessed 14 Apr. 2025].

HypeAuditor, 2024. Top News & Politics YouTube Channels in the United States. Available at: https://hypeauditor.com/top-youtube-channels/news-politics-category-us/ [Accessed 14 Apr. 2025].

Press Gazette, 2024. Most popular websites for news in the UK: Monthly top 50 listing. Available at: https://pressgazette.co.uk/media-audience-and-business-data/most-popular-uk-news-websites/ [Accessed 14 Apr. 2025].

Press Gazette, 2024. Biggest websites for news US: Top 50 updated each month. Available at: https://pressgazette.co.uk/media-audience-and-business-data/largest-news-websites-us/ [Accessed 14 Apr. 2025].

Dewey, C., 2016. 6 in 10 of you will share this link without reading it, a new, depressing study says. *The Washington Post*. Available at: https://www.washingtonpost.com/news/the-intersect/wp/2016/06/16/six-in-10-of-you-will-share-this-link-without-reading-it-according-to-a-new-and-depressing-study/ [Accessed 14 Apr. 2025].

NewsAPI, n.d. NewsAPI documentation. Available at: https://newsapi.org/docs [Accessed 18 Apr. 2025].

Google Developers, n.d. YouTube Data API v3 documentation. Available at: https://developers.google.com/youtube/v3/docs [Accessed 18 Apr. 2025].

AllSides, 2024. Media Bias Chart. Available at: https://www.allsides.com/media-bias/media-bias-chart [Accessed 14 Apr. 2025].

Lu, D., 2025. We tried out DeepSeek. It worked well, until we asked it about Tiananmen Square and Taiwan. *The Guardian*, 28 Jan. Available at: https://www.theguardian.com/technology/2025/jan/28/we-tried-out-deepseek-it-works-well-until-we-asked-it-about-tiananmen-square-and-taiwan [Accessed 14 Apr. 2025].

LiveMint, 2024. Elon Musk shocked by Gemini AI's answer on misgendering Caitlyn Jenner: 'AI mirrors mistakes of its creators'. *LiveMint*. Available at: https://www.livemint.com/technology/tech-news/elon-musk-shocked-by-google-gemini-ais-answer-on-misgendering-caitlyn-jenner-ai-mirrors-mistakes-of-its-creators-11709612302665.html [Accessed 14 Apr. 2025].

Vijayasarathy, L.E.O.R. and Turk, D., 2008. Agile software development: A survey of early adopters. *Journal of Information Technology Management*, 19(2), pp.1–8. Available at: https://www.researchgate.net/publication/251745421_Agile_software_development_A_survey_of_early_adopters [Accessed 14 May 2025].

Balaji, S. and Murugaiyan, M.S., 2012. Waterfall vs. V-Model vs Agile: A comparative study on SDLC. *International Journal of Information Technology and Business Management*, 2(1), pp.26–30. Available at: https://mediaweb.saintleo.edu/Courses/COM430/M2Readings/WATEERFALLVs%20V-MODEL%20Vs%20AGILE%20A%20COMPARATIVE%20STUDY%20ON%20SDLC.pdf [Accessed 14 May 2025].

Anderson, D.J., 2010. *Kanban: Successful evolutionary change for technology organizations*. Sequim, Washington, USA: Blue Hole Press.

Flora, H.K. and Chande, S.V., 2014. A systematic study on Agile software development methodologies and practices. *International Journal of Computer Science and Information Technologies*, 5(3), pp.3626–3637. Available at: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=5d2ad96c2dce23e6875dfeeae70a4cf122372457 [Accessed 14 May 2025].

Sjøberg, D.I., Johnsen, A. and Solberg, J., 2012. Quantifying the effect of using Kanban versus Scrum: A case study. *IEEE Software*, 29(5), pp.47–53. Available at: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6231615 [Accessed 14 May 2025].

Moisiadis, F., 2002. The fundamentals of prioritising requirements. In: J. Andersen, ed., *Proceedings of the Systems Engineering, Test and Evaluation Conference: The Five Layers of*

System Engineering and Test and Evaluation. Sydney: Systems Engineering Society of Australia, pp.59–70. Available at: https://www.researchgate.net/publication/228548912_The_fundamentals_of_prioritising_requirements [Accessed 14 Apr. 2025].

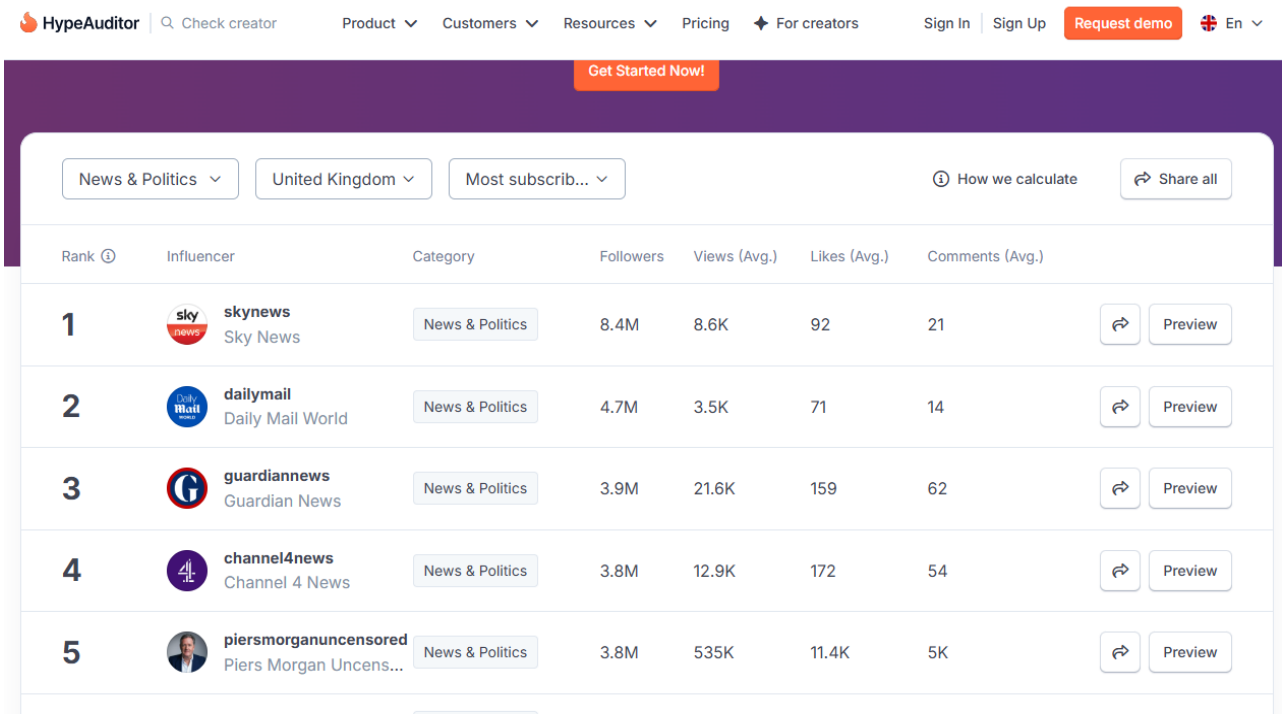
Google, n.d. Advanced Search. Available at: https://www.google.co.uk/advanced_search [Accessed 18 Apr. 2025].

Alaidaros, H., Omar, M. and Romli, R., 2021. The state of the art of agile kanban method: challenges and opportunities. *Independent Journal of Management & Production*, 12(8). Available at: https://doi.org/10.14807/ijmp.v12i8.1482 [Accessed 18 Apr. 2025].

Nielsen, J., 2012. *Usability 101: Introduction to Usability*. Nielsen Norman Group. Available at: https://www.nngroup.com/articles/usability-101-introduction-to-usability/ [Accessed 20th February 2023].

Jiménez Iglesias, L., Aguilar Paredes, C., Sánchez Gómez, L. and Pérez-Montoro Gutiérrez, M., 2018. User experience and media. The three click rule in newspapers' webs for smartphones. *Revista Latina de Comunicación Social*, 73, pp.595–613. Available at: http://www.revistalatinacs.org/073paper/1271/30en.html [Accessed 8 May 2025]. https://doi.org/10.4185/RLCS-2018-1271en.

APPENDIX A – MOST POPULAR UK & US CHANNELS

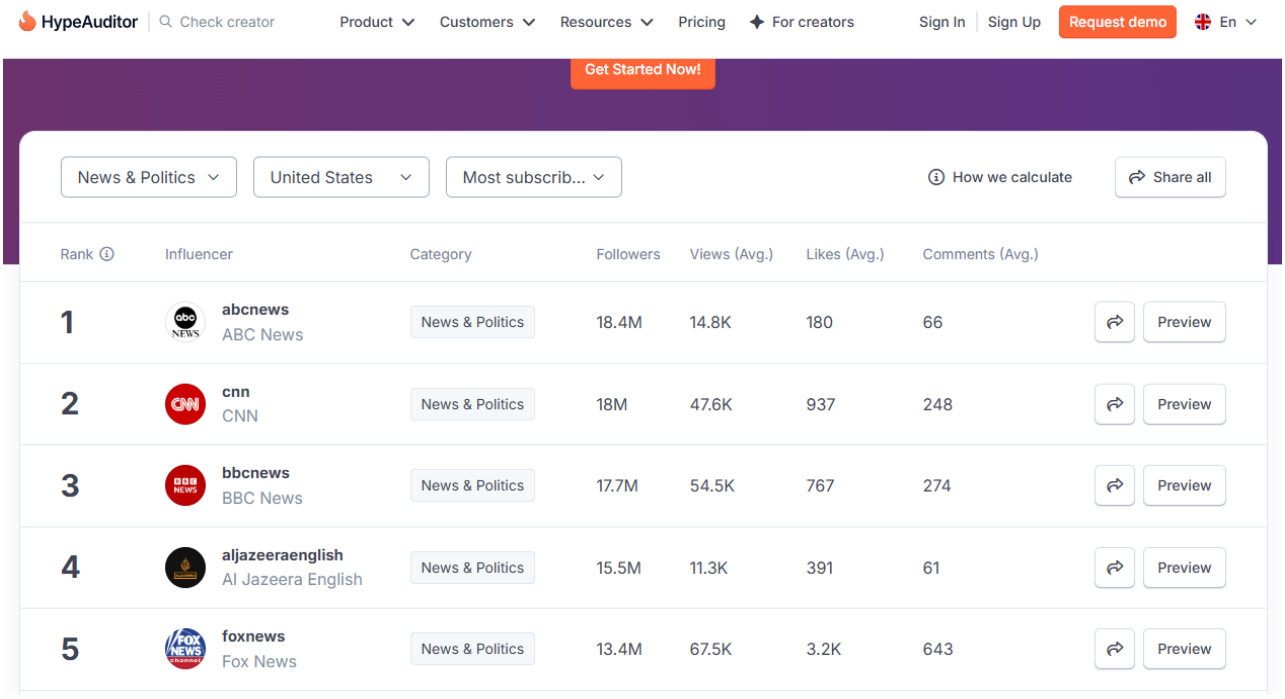


HypeAuditor | Check creator | Product | Customers | Resources | Pricing | For creators | Sign In | Sign Up | Request demo | En

Get Started Now!

News & Politics | United Kingdom | Most subscrib... | How we calculate | Share all

Rank	Influencer	Category	Followers	Views (Avg.)	Likes (Avg.)	Comments (Avg.)	
1	skynews Sky News	News & Politics	8.4M	8.6K	92	21	Preview
2	dailymail Daily Mail World	News & Politics	4.7M	3.5K	71	14	Preview
3	guardiannews Guardian News	News & Politics	3.9M	21.6K	159	62	Preview
4	channel4news Channel 4 News	News & Politics	3.8M	12.9K	172	54	Preview
5	piersmorganuncensored Piers Morgan Uncens...	News & Politics	3.8M	535K	11.4K	5K	Preview



HypeAuditor | Check creator | Product | Customers | Resources | Pricing | For creators | Sign In | Sign Up | Request demo | En

Get Started Now!

News & Politics | United States | Most subscrib... | How we calculate | Share all

Rank	Influencer	Category	Followers	Views (Avg.)	Likes (Avg.)	Comments (Avg.)	
1	abcnews ABC News	News & Politics	18.4M	14.8K	180	66	Preview
2	cnn CNN	News & Politics	18M	47.6K	937	248	Preview
3	bbcnews BBC News	News & Politics	17.7M	54.5K	767	274	Preview
4	aljazeeraenglish Al Jazeera English	News & Politics	15.5M	11.3K	391	61	Preview
5	foxnews Fox News	News & Politics	13.4M	67.5K	3.2K	643	Preview

APPENDIX B – MOST POPULAR UK & US NEWS SOURCES

Top 50 newsbrands in the UK by audience

February 2025

Search in table

Page 1 of 2 >

	Newsbrand	Audience	MoM % change	YoY % change	Audience reach (%)
1	BBC	38.6m	-1.5	2.7	77%
2	Mail Online	20.8m	-3.2	-2.9	41%
3	The Sun	20.1m	0.8	-14.2	40%
4	The Guardian	19.9m	-6.2	-3	40%
5	Mirror	18.2m	-7.3	-14.5	36%
6	The Independent	18.2m	-5.4	-6.2	36%
7	Yahoo!	16.9m	-5.6	11.6	34%
8	Sky News	16.4m	-8	-7.2	33%
9	Daily Express	15.7m	-8.4	34.5	31%
10	ITV	14.6m	-2.8	1.3	29%
11	The Telegraph	14.4m	-7.4	0.1	29%
12	Metro	14m	-7.5	3.1	28%
13	Money Saving Expert	13.9m	-2.6	2.5	28%
14	Manchester Evening News	12.1m	-3.8	24.4	24%
15	Good Food	11.6m	-8.6	-20.1	23%
16	Times & Sunday Times	10.3m	2.7	-14.7	21%
17	The Evening Standard	9.9m	-1.6	-8.8	20%
18	Birmingham Live	8.4m	0.4	-23.8	17%
19	GB News	8.3m	-10.9	-7.5	17%
20	Radio Times	7.8m	-12.9	1.2	16%
21	New York Times	7.6m	-6.2	6	15%
22	Hello! Magazine	7.5m	-3.7	10.9	15%
23	Daily Record	7.3m	-1.7	-9.9	14%
24	Healthline Media	7m	-18.1	-26.7	14%
25	Liverpool Echo	6.6m	4.3	-4.7	13%

Top 50 websites in the US

Visits, March 2025

Search in table

	Website	Mar 2025	Month-on-month change	Year-on-year change
1	nytimes.com	492.5m	8.9%	6.0%
2	cnn.com	351.5m	1.9%	-9.1%
3	foxnews.com	272.8m	4.7%	-0.2%
4	msn.com	201.3m	2.5%	-9.0%
5	people.com	158.3m	5.1%	14.1%
6	finance.yahoo.com	155.9m	8.6%	3.0%
7	usatoday.com	134.7m	4.6%	-17.2%
8	apnews.com	128m	4.3%	65.5%
9	news.google.com	121.1m	9.3%	8.9%
10	nypost.com	120.4m	-0.3%	-15.8%
11	bbc.com	119.6m	6.5%	28.6%
12	cnbc.com	113.2m	29.9%	20.6%
13	dailymail.co.uk	101.1m	6.6%	-17.6%
14	forbes.com	97.9m	4.2%	-2.7%
15	newsweek.com	96.6m	24.9%	-1.2%
16	nbcnews.com	86.4m	7.8%	6.3%
17	washingtonpost.com	81.8m	-6.0%	-18.7%
18	theguardian.com	80m	12.0%	18.9%
19	news.yahoo.com	78.4m	7.0%	-13.4%
20	wsj.com	76.3m	4.3%	12.1%
21	substack.com	73.1m	1.3%	59.5%
22	cbsnews.com	71.1m	-9.3%	-7.4%
23	reuters.com	61.7m	-4.9%	37.9%
24	businessinsider.com	59m	23.0%	-20.3%
25	abcnews.go.com	54.9m	3.8%	24.5%
26	drudgereport.com	50.6m	9.4%	5.3%
27	buzzfeed.com	50.6m	5.6%	1.9%
28	thehill.com	46.1m	7.5%	22.1%
29	huffpost.com	44m	2.5%	-23.3%
30	politico.com	43.4m	10.6%	22.5%
31	breitbart.com	38.6m	7.9%	29.0%
32	athlonsports.com	37.1m	-10.5%	693.4%

APPENDIX C – FULLY COMPLETE REQUIREMENTS AND USER STORIES

Priority	ID	Requirement Description	User Story	Acceptance Criteria
Must Have	**F1**	Enable users to input politically focused search queries	As a user, I want to search for politically focused content so that I can find relevant information	User can enter search terms related to political topics or figures
Must Have	**F2**	Allow users to select from a range of dropdown options	As a user, I want to filter my search using dropdown options so that I can narrow my results	User can select various filter options from dropdowns
Must Have	**F3**	Execute search based on specified filters	As a user, I want my search to use the filters I've selected so that results match my criteria	System executes search with all specified filter parameters
Must Have	**F4**	Dynamically render relevant search results	As a user, I want to see relevant results from my search so that I can browse the content	System displays results that match the search query and filters
Must Have	**N1**	Responsive interface	As a user, I want the interface to work on various devices and browsers so that I can use it anywhere	Interface adapts to different screen sizes and browsers
Must Have	**N2**	User-friendly interface	As a user, I want an intuitive interface so that I can use the system efficiently	Interface follows UX best practices and is easy to navigate
Must Have	**N3**	Clean UI design	As a user, I want a clean interface so that I can focus on the content	UI has minimal clutter and clear visual hierarchy
Must Have	**N4**	Accessibility features	As a user with accessibility needs, I want accessible features so that I can use the system	Interface follows accessibility standards (WCAG)
Must Have	**N5**	Error handling	As a user, I want clear error messages so that I know when something goes wrong	System provides user-friendly error messages
Must Have	**N6**	Security features	As a user, I want my data to be secure so that I feel safe using the system	Sensitive data is properly secured
Must Have	**N7**	Maintainable code	As a developer, I want modular code so that I can maintain it efficiently	Code follows best practices for maintainability
Must Have	**F5**	Integrate Ollama API for sentiment analysis	As a user, I want content to be analysed for sentiment so that I can understand the	System connects to Ollama API, works with prompts and processes content

			organisations emotional tone	
Must Have	**F6**	Enable sentiment classification on fetched content	As a user, I want to see sentiment labels for content so that I can understand potential organisational bias	System assigns sentiment labels (positive/negative/neutral) to content
Must Have	**F7**	Provide justifications for sentiment classifications	As a user, I want explanations for sentiment labels so that I understand the reasoning behind assigned labels, and to determine if the selected LLM has any inherent biases towards any political topics or figures	LLM generates explanations for assigned sentiment labels
Must Have	**N8**	Performance optimization	As a user, I want fast response times so that I don't have to wait	System responds within acceptable time limits
Must Have	**F8**	Display organization-specific sentiment statistics	As a user, I want to see sentiment statistics by organization so I can compare sources	System displays sentiment statistics per organization
Must Have	**F9**	Store sentiment data in a database	As a user, I want reliable data storage so that statistics are accurate over time	System stores sentiment data in PostgreSQL database
Must Have	**F10**	Implement model feedback system	As a user, I want to provide feedback on sentiment analysis so that I can indicate if I agree or disagree	Users can submit agreement/disagreement with sentiment labels
Could Have	**F11**	Support for fine-tuning alternative LLMs	As a user, I want access to different LLMs so that I can compare analysis approaches	System supports integration with Hugging Face models and finetunes
Should Have	**F12**	Automatic database backups	As an administrator, I want automatic backups so that data is protected	System performs regular database backups
Must Have	**F13**	Display user feedback	As a user, I want to see feedback from other users so that I can understand different perspectives	System displays submitted feedback on sentiment analysis
Must Have	**F14**	Implement Google authentication	As a user, I want to login with Google so that I don't need to create new credentials	Users can sign in using their Google accounts
Must Have	**F15**	Create User database table	As a developer, I want to store user information so that I can link users to their feedback	System stores user information in a database table
Must Have	**F16**	Link User table to Feedback table	As a developer, I want to associate feedback with users so that multiple	System maintains relationships between users and their feedback

			users can provide feedback on the same content	
Must Have	**F17**	GDPR compliance	As a user, I want my data to be handled according to GDPR so that my privacy is protected	System includes privacy policy and terms of service
Should Have	**F18**	Admin interface for system monitoring	As an administrator, I want to monitor system usage so that I can troubleshoot issues	System provides admin dashboard with usage statistics
Must Have	**N9**	Secure credential handling	As a user, I want my credentials to be secure so that my information is protected	System avoids storing sensitive credentials
Must Have	**N10**	Industry standard security practices	As a user, I want industry standard security so that I feel confident using the system	System follows security best practices

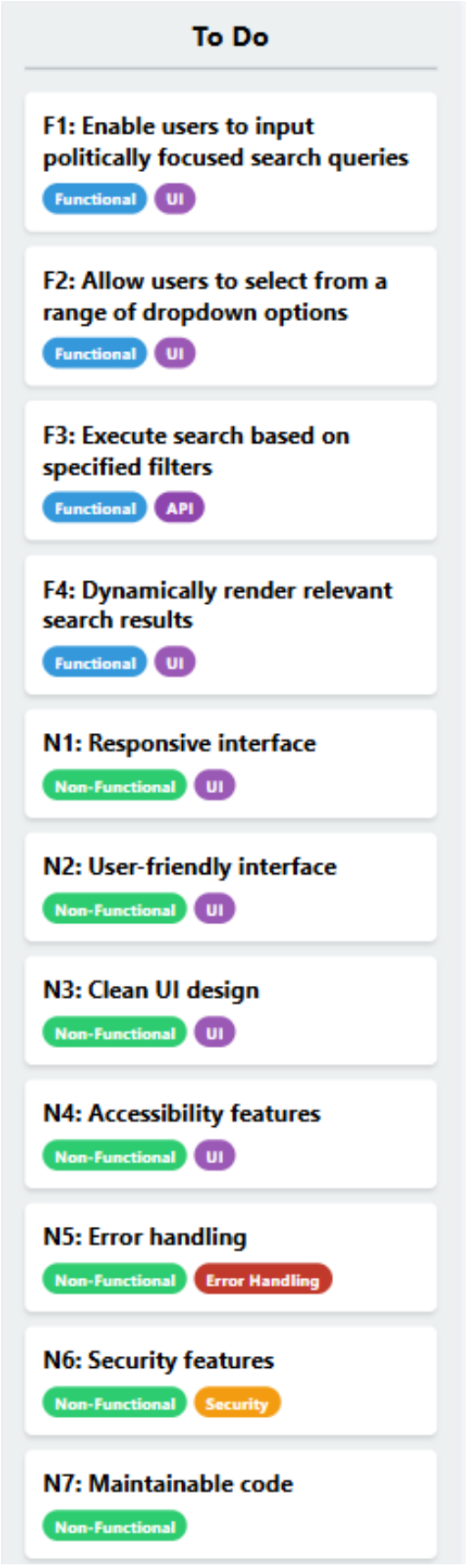
APPENDIX D – EVALUATION OF REQUIREMENTS

Iteration	Requirement ID	Description	Status
Iteration 1	F1	Enable users to input politically focused search queries	Implemented
Iteration 1	F2	Allow users to select from a range of dropdown options	Implemented
Iteration 1	F3	Execute search based on specified filters	Implemented
Iteration 1	F4	Dynamically render relevant search results	Implemented
Iteration 1	N1	Responsive interface	Implemented
Iteration 1	N2	User-friendly interface	Implemented
Iteration 1	N3	Clean UI design	Implemented
Iteration 1	N4	Accessibility features	Implemented
Iteration 1	N5	Error handling	Implemented
Iteration 1	N6	Security features	Implemented
Iteration 1	N7	Maintainable code	Implemented
Iteration 2	F5	Integrate Ollama API for sentiment analysis	Implemented
Iteration 2	F6	Enable sentiment classification on fetched content	Implemented
Iteration 2	F7	Provide justifications for sentiment classifications	Implemented
Iteration 2	N8	Performance optimization	Implemented
Iteration 3	F8	Display organization-specific sentiment statistics	Implemented
Iteration 3	F9	Store sentiment data in a database	Implemented
Iteration 3	F10	Implement model feedback system	Implemented
Iteration 3	F12	Automatic database backups	Implemented
Iteration 3	F13	Display user feedback	Implemented
Iteration 3	F11	Support for fine-tuning alternative LLMs	Not Implemented
Iteration 4	F14	Implement Google authentication	Implemented
Iteration 4	F15	Create User database table	Implemented
Iteration 4	F16	Link User table to Feedback table	Implemented
Iteration 4	F17	GDPR compliance	Implemented
Iteration 4	F18	Admin interface for system monitoring	Implemented

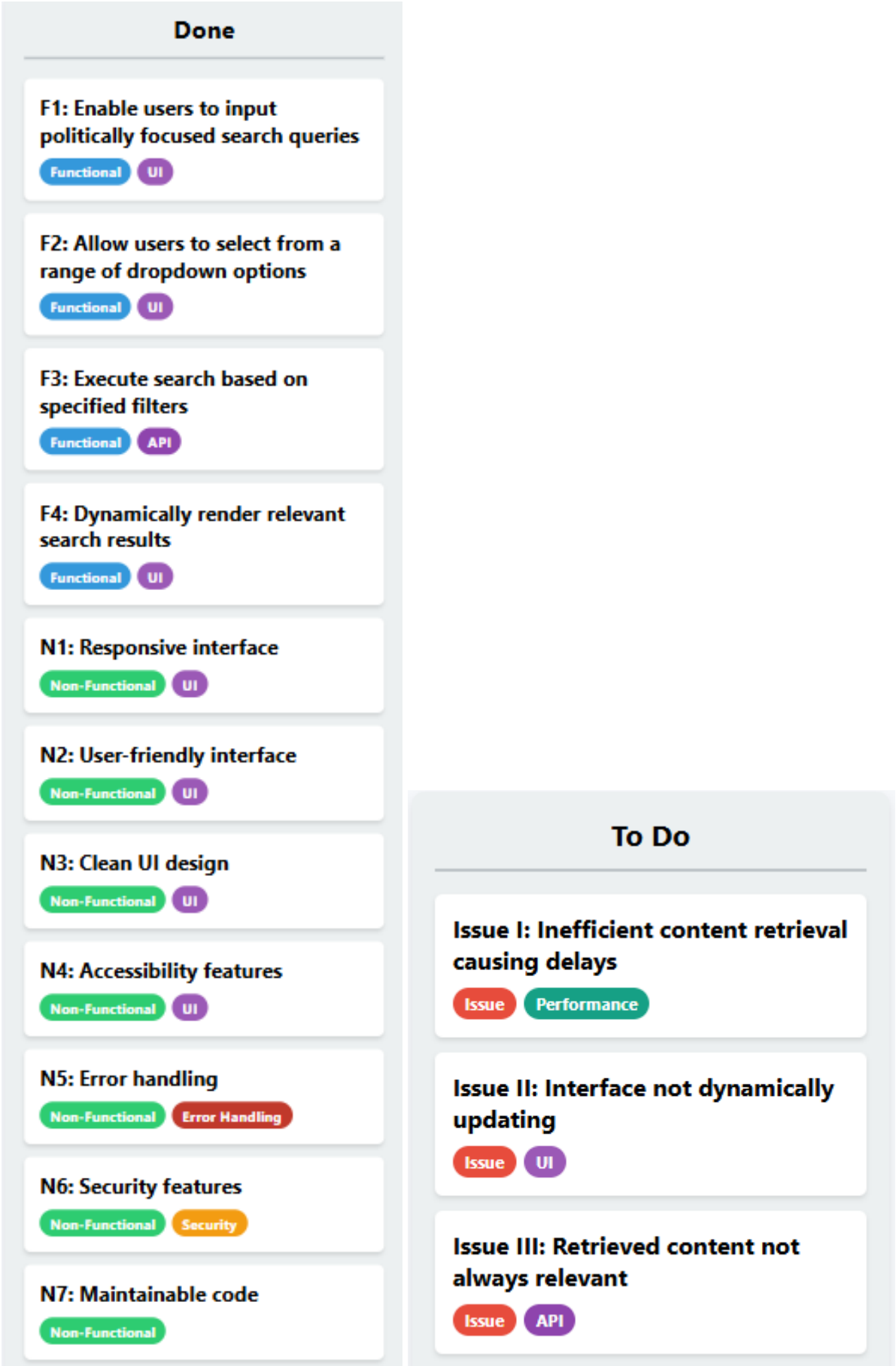
Iteration 4	N9	Secure credential handling	Implemented
Iteration 4	N10	Industry standard security practices	Implemented

APPENDIX E – PROJECT KANBAN BOARDS

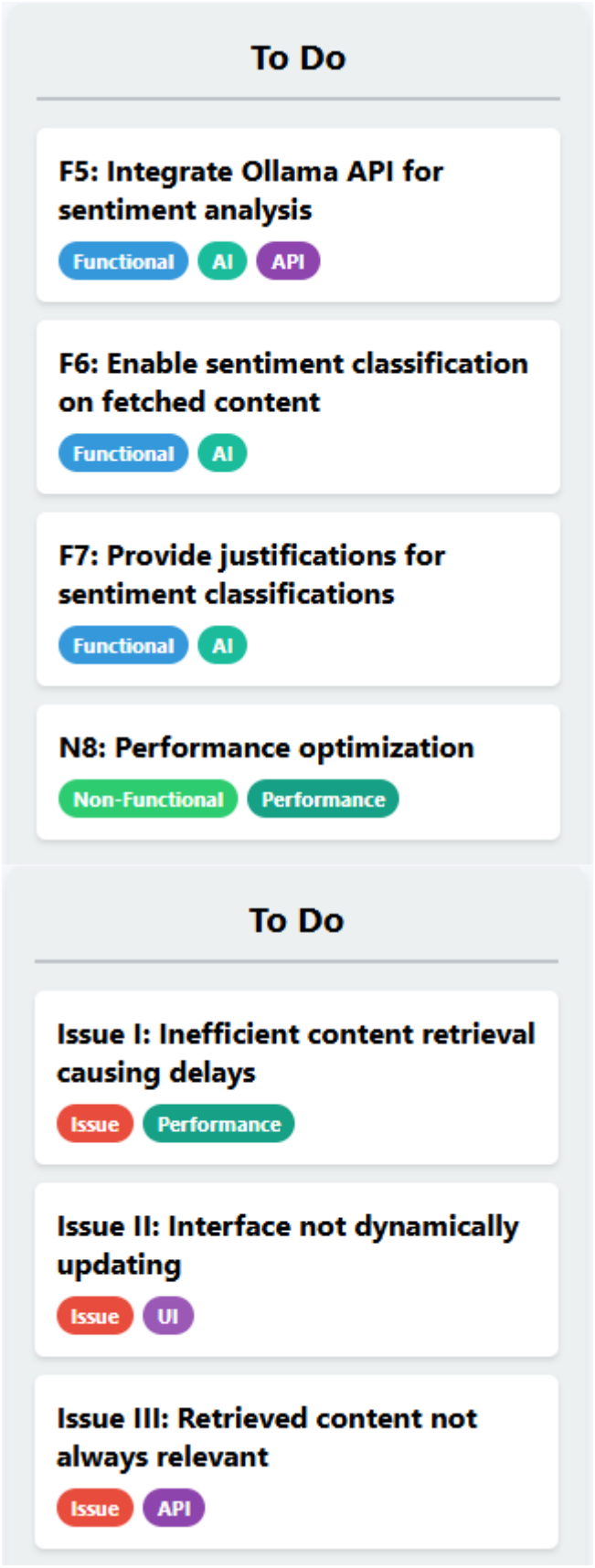
Development Iteration 1 – Start of Iteration Kanban Board



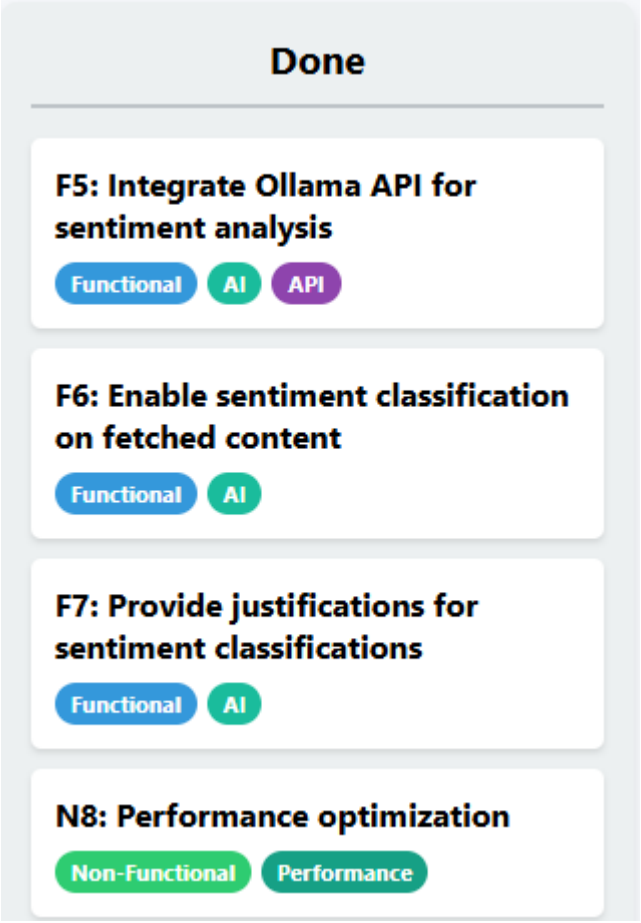
Development Iteration 1 – End of Iteration Kanban Board

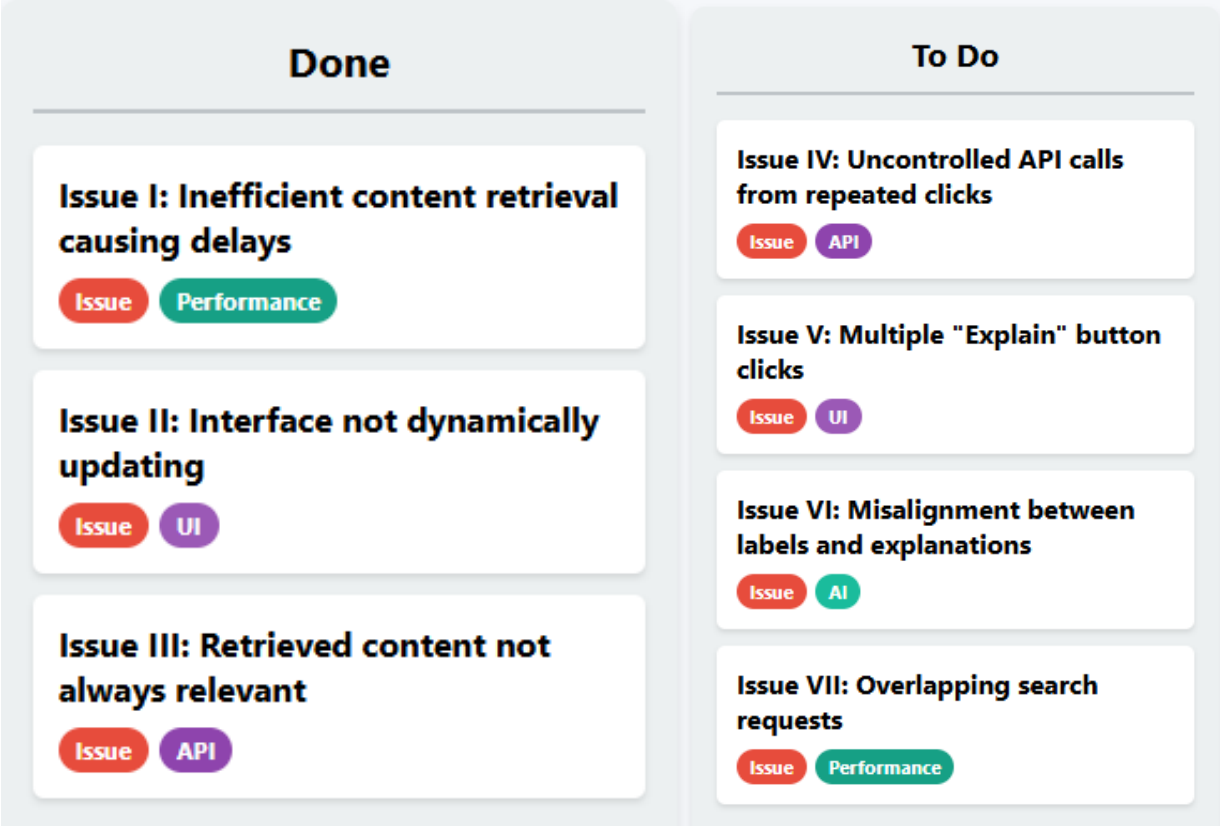


Development Iteration 2 – Start of Iteration Kanban Board

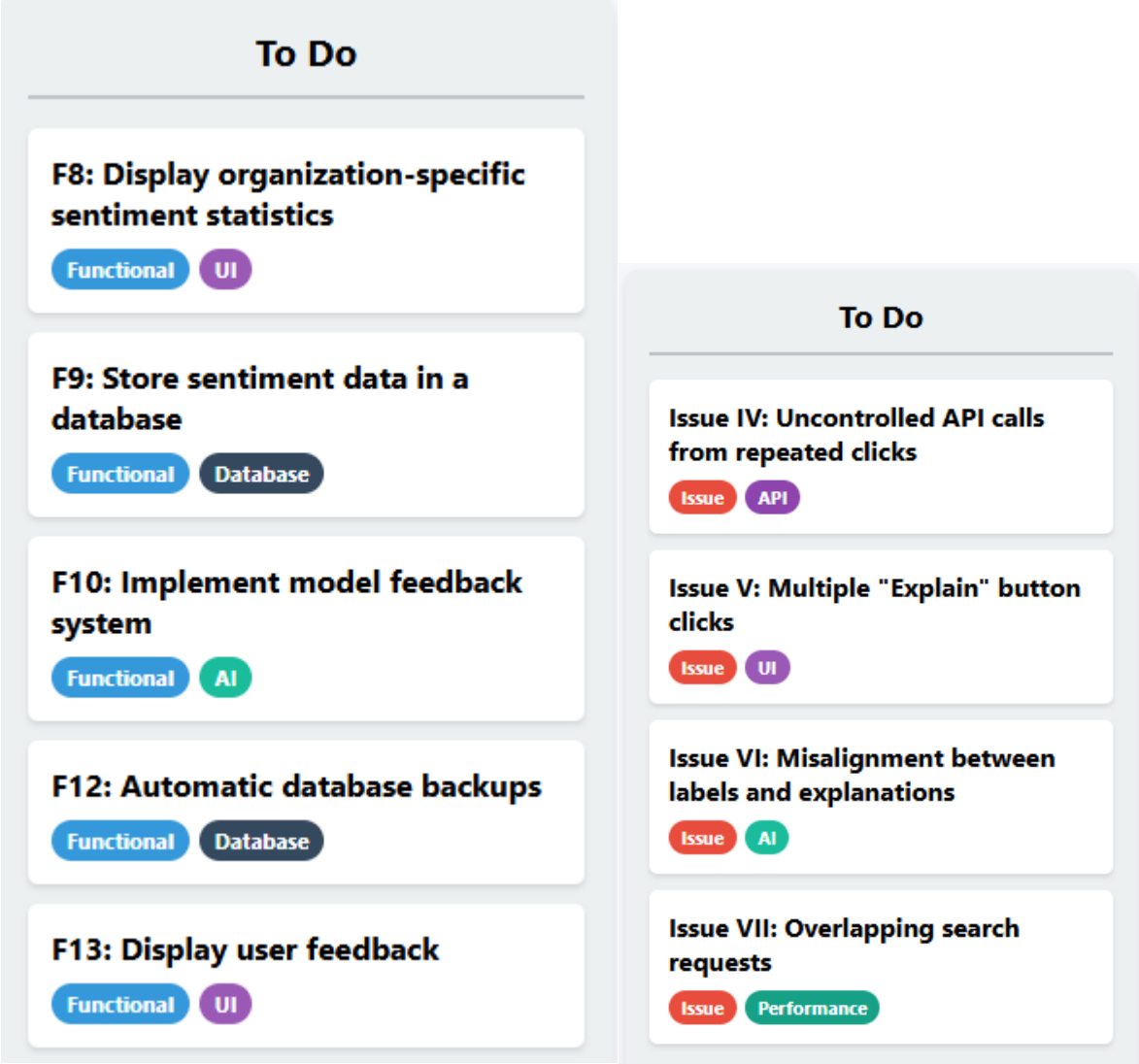


Development Iteration 2 – End of Iteration Kanban Board

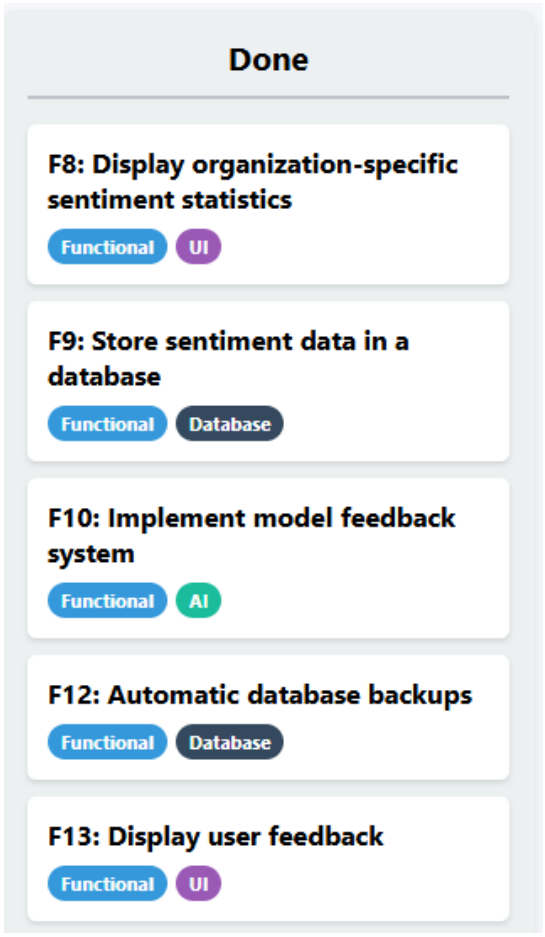


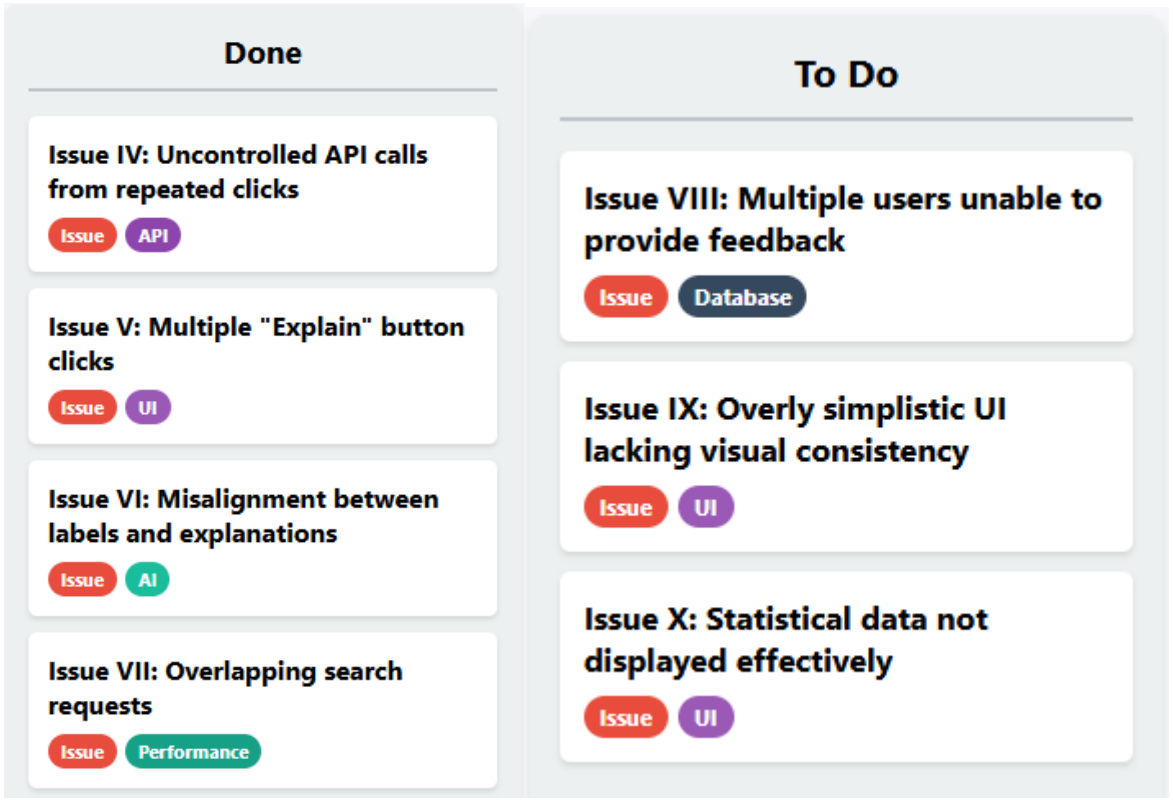


Development Iteration 3 – Start of Iteration Kanban Board

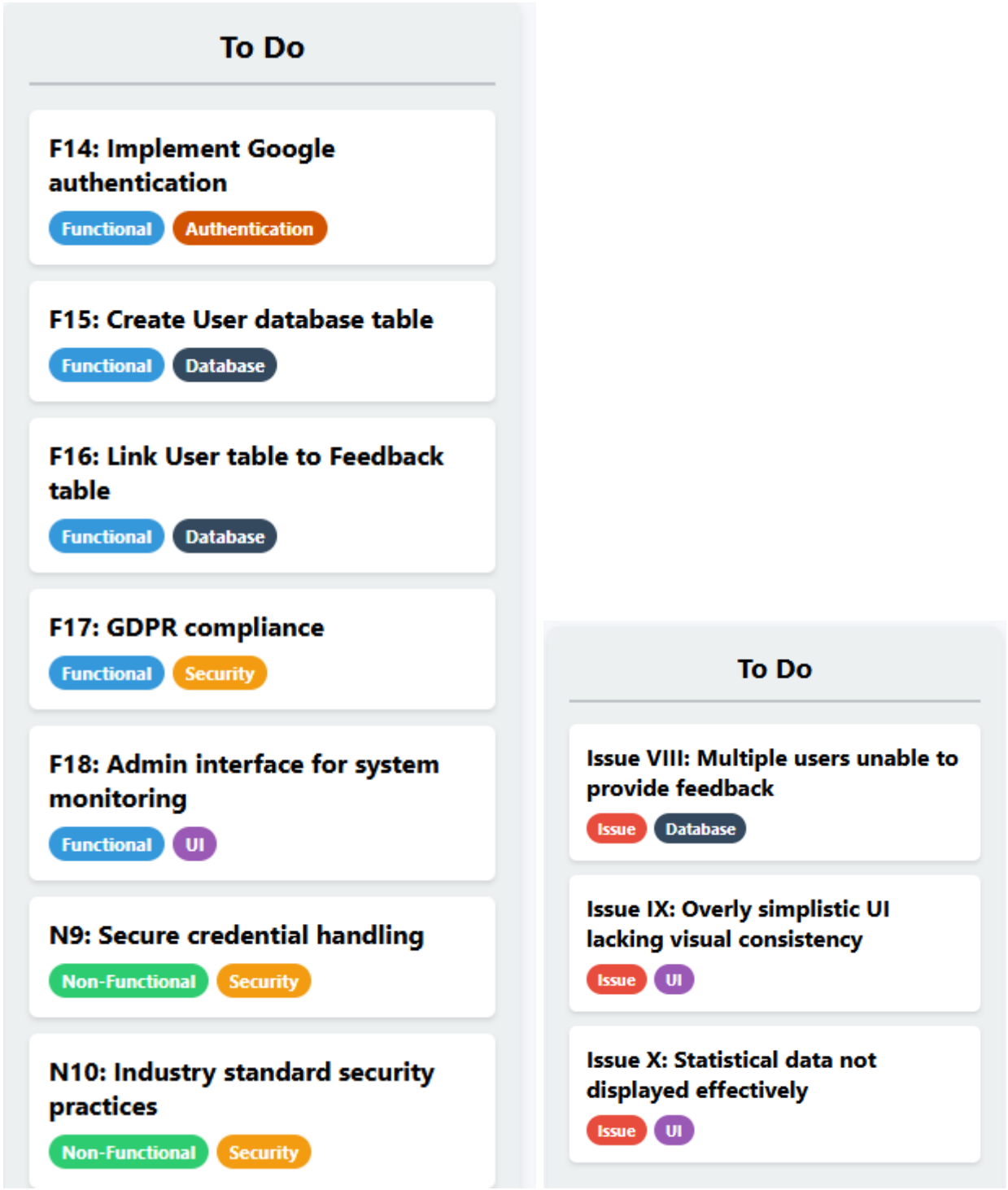


Development Iteration 3 – End of Iteration Kanban Board

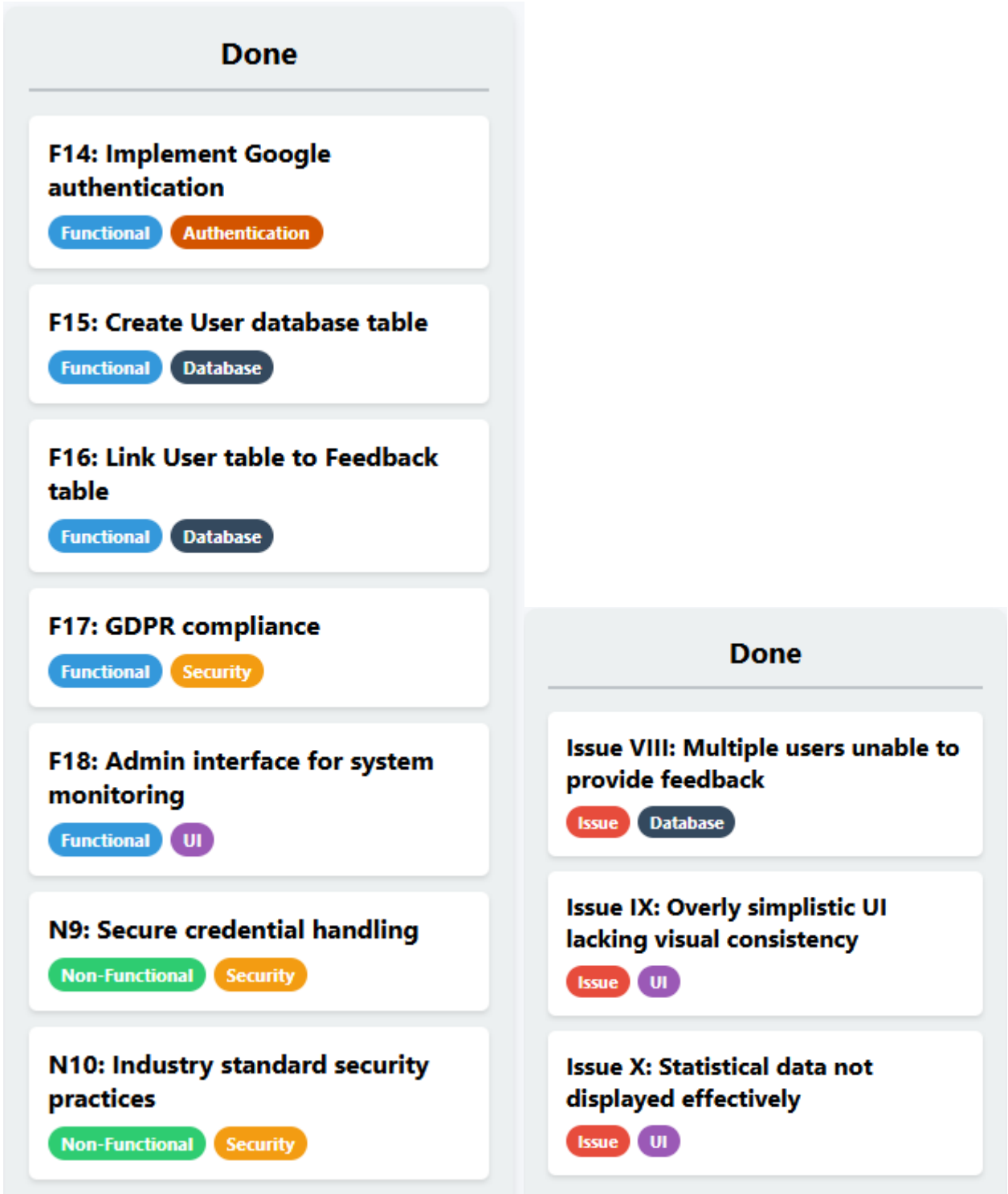




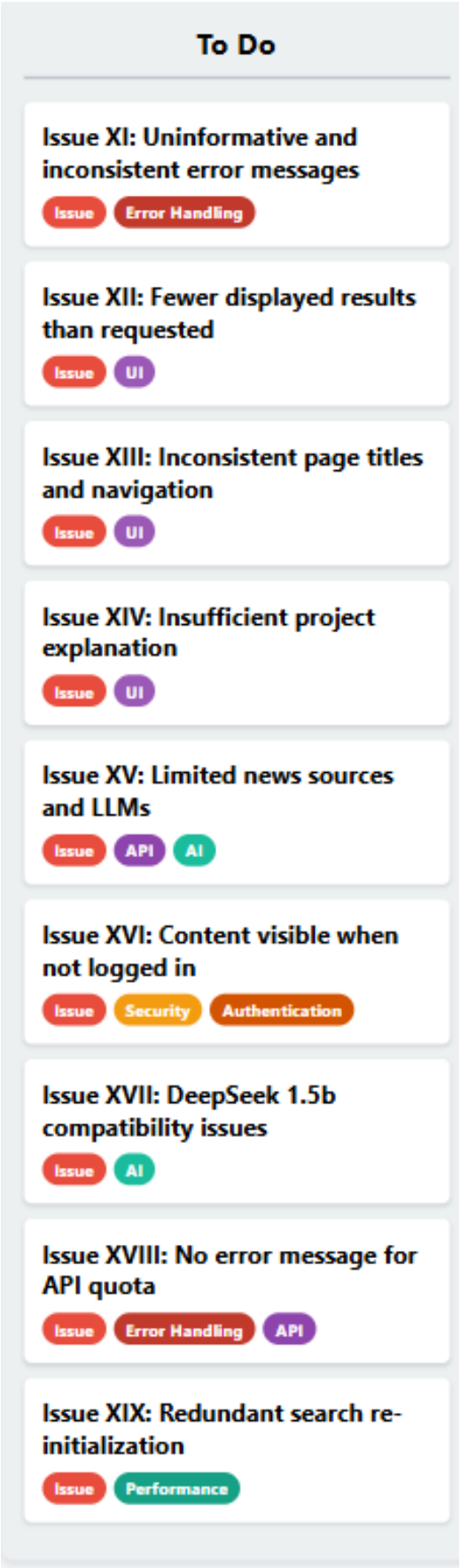
Development Iteration 4 – Start of Iteration Kanban Board



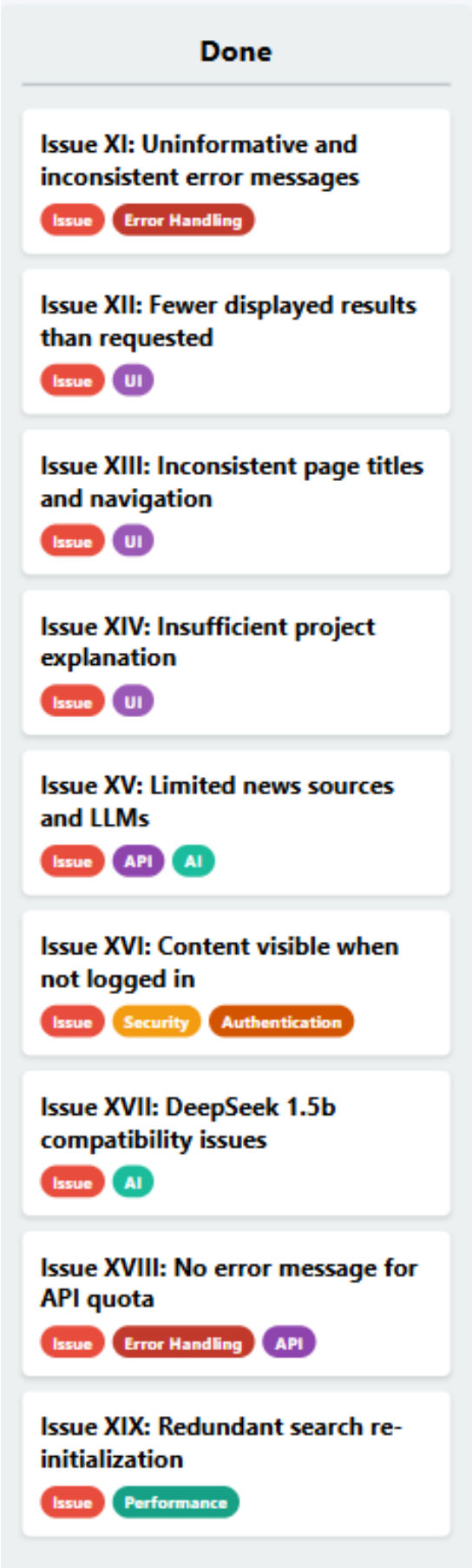
Development Iteration 4 – End of Iteration Kanban Board



Development Iteration 5 – Start of Iteration Kanban Board



Development Iteration 5 – End of Iteration Kanban Board



APPENDIX F – FULL ISSUE TABLE

Issue ID	Issue Description	Planned Solution
I	The current implementation retrieves all API content at once, causing inefficiency and delays, which degrades the user experience.	Introduce sequential loading to load content incrementally, reducing waiting times, and showcase a progress bar for feedback.
II	The interface does not dynamically update based on selected categories, allowing incompatible dropdown selections, leading to incorrect parameter submission.	Implement JavaScript to dynamically update the UI based on category selections, preventing incorrect parameter submission.
III	Some retrieved content does not explicitly mention the search query in the title or headline, which reduces the accuracy of sentiment analysis.	Apply stricter searches to filter out irrelevant content and ensure more accurate sentiment analysis.
IV	Users can repeatedly click the "Search" button, which results in uncontrolled API calls. This can quickly exhaust the API quota and impact system performance.	Implement JavaScript to disable the search button until all content loaded.
V	The “Explain” button can be triggered multiple times in quick succession, prompting the LLM to return different justifications for the same sentiment. This leads to inconsistencies and undermines the credibility of the model’s reasoning.	The system will remove the explain button once clicked to prevent users from triggering multiple LLM (Large Language Model) requests in quick succession and will enforce a one-explanation-per-result policy
VI	In some cases, the LLM assigns a sentiment label (e.g., positive) but provides an explanation that supports a different sentiment (e.g., negative), creating confusion and reducing trust in the system.	Refine prompt to align better between assigning sentiment and the generating corresponding explanation.
VII	Users can initiate new searches or switch platforms without cancelling ongoing searches. This leads to overlapping requests being sent to the backend, which can overload the server and result in unexpected behaviour.	Implement JavaScript to disable the search button until all content loaded so the backend isn’t overwhelmed.
VIII	Multiple users are unable to provide feedback on the same title/headline. Once feedback is submitted for a title, no additional feedback can be added by other users.	New User table cross referenced with the Feedback table to allow multiple users to provide feedback on titles with the complements of Googles login
IX	The current UI is overly simplistic and lacks visual consistency.	Focus will be on creating a consistent and aesthetically pleasing UI across the application
X	In addition to Issue IX, statistical data is neither calculated accurately nor displayed in a visually engaging way.	Calculate sentiment metrics and integrate visualisation tools such as Three.js to present statistics more effectively.
XI	Uninformative and inconsistent error messages	Implement JavaScript to display clear and context-specific error messages for different conditions.
XII	Application shows analysed results but fewer than the number of articles or videos requested	Inform users about the strictness of search filters and possible reasons for limited results.

XIII	Inconsistent page titles and navigation labels	Rename and refactor web pages to use appropriate, informative, and consistent naming conventions.
XIV	Insufficient explanation about project purpose and functionality	Create a web page to provide detailed information about the project's goals, scope, and functionality.
XV	Limited number of news sources and LLMs integrated	Expand the system by introducing additional news sources and large language models (LLMs) as planned.
XVI	Content visible even when not logged in	Implement proper authentication checks to ensure that restricted content is only visible to logged-in users.
XVII	Selected model DeepSeek 1.5b does not work whilst all other DeepSeek models do	Exclude DeepSeek 1.5b from model selection.
XVIII	No error message when API quota limits are exceeded.	Inform the users that they have exceeded the maximum number of API requests and to try again in X number of hours.
XIX	When searching for a query, the search is re-initialised even if the query is repeated, leading to longer waiting times.	Retrieve results from cache when submitting a previously searched query instead of initialising a new request.

APPENDIX G – FULL SOLUTION TABLE

Issue ID	Solution Implementation
I	Sequential content loading was implemented using EventSource to send items as separate messages, triggering the onmessage event. This parses the item, updates the progress counter, adds it to the cache, and renders it in the UI. A progress bar visually tracks the loading process. Backend methods in the news and video source helper files yield results one by one as server-sent events. Sentiment results for common queries are cached to improve performance and reduce unnecessary API calls.
II	An onchange event listener was added to dynamically update the frontend form, ensuring only valid combinations of category and country are displayed. This prevents errors and guides users through correct input choices.
III	The NewsAPI's "everything" endpoint was refined to search strictly within headlines, filtering out irrelevant results. This ensures only relevant content is processed by the SentimentAnalyser for sentiment analysis and statistical evaluation.
IV	Sequential content loading was implemented using JavaScript to disable the search button until all content is fully loaded Figure 18). This prevents users from repeatedly clicking the button, thereby reducing uncontrolled API calls and improving system performance.
V	The "Explain" button was removed from the HTML once clicked, preventing multiple triggers of the LLM request in quick succession (Figure 19). This enforces a one-explanation-per-result policy, ensuring consistency in sentiment analysis.
VI	The LLM prompt was refined to better align the sentiment label and the generated explanation (Figure 19). This ensures that the explanation supports the sentiment, providing clearer and more reliable feedback for users.
VII	A cancellation feature was introduced, allowing users to halt an ongoing search (Figure 18). This ensures that overlapping requests are prevented, reducing system load and avoiding unexpected behaviour.
VIII	A new User table was created and cross-referenced with the Feedback table to enable multiple users to provide feedback on the same title/headline. The combination of user_id and item_title was constrained to ensure users can only provide feedback once per unique result. Google login (OAuth 2.0) ensures secure user identification and linkage to feedback entries (F14-F17). This maintains data integrity and allows participation across multiple headlines or items. Each feedback entry is associated with its user via a user_id foreign key, ensuring all submissions are trackable and attributable.
IX	A consistent UI layout was introduced using a base.html template with shared headers and footers. Page-specific content is injected dynamically, with styling handled by both individual and global CSS files. This ensured a cohesive visual identity and seamless user experience across the application (N1-N5). All figures presented this iteration showcase the improved UI, addressing the issue of the overly simplistic UI (Figures 21-32).
X	Three.js was integrated to visualize statistical data from the SentimentResults table, replacing static visualization tools like matplotlib. This allowed for dynamic, interactive 3D visualizations of sentiment data, improving the user experience and making statistical data more engaging. Additionally, statistical calculations were refined to display accurate percentages, ensuring the information presented is both visually compelling and reliable (N1-N5, F8). Figures 28 & 31 demonstrate the updated statistic pages, showing how the 3D visualization of the data enhances the presentation of sentiment analysis results and model performance.
Issue ID	**Solution Implementation**
XI & XII	A help button ("?") was implemented to guide users by explaining why strict search filters may result in fewer retrieved articles or videos. Additionally, error messages were

<table>
<tr><td></td><td>enhanced to provide more context, improving overall clarity and user experience (Figures 33 & 34).</td></tr>
<tr><td>XIII</td><td>A code refactor was done to standardize web pages and navigation labels, improving user navigation and UI consistency.</td></tr>
<tr><td>XIV</td><td>An About page was created to communicate the project’s aims, scope, functionality, and significance, enhancing the onboarding experience for new users (Figure 35).</td></tr>
<tr><td>XV</td><td>Based on the insights from Table 4, the following news sources were selected:

<table>
<tr><th>News Source</th><th>Country</th></tr>
<tr><td>• MSBC (Left)
• NewsWeek (Centre)
• Fox News (Right)</td><td>US</td></tr>
<tr><td>• BBC News (Centre)
• Daily Mail (Right)</td><td>UK</td></tr>
</table>

These sources were chosen to provide a balanced representation of political leanings, ensuring a diverse set of perspectives in the data collection and sentiment classification process. However, a key limitation is the lack of UK-based left-leaning media sources. This is due to restrictions imposed by NewsAPI, which, under the free tier, does not provide access to such outlets. As a result, only two UK sources were selected based on their availability and relevance to the project. For future work, developers may consider expanding the list of sources and supported countries to increase the system's scope. Additionally, a diverse set of models with varying parameters was integrated to mitigate inherent biases and improve compatibility with a range of devices.</td></tr>
<tr><td>XVI</td><td>Authentication checks were reinforced across the app by applying the is_authenticated method to headers and page routes, restricting content to authenticated users only. This ensures that only logged-in users can submit and view model feedback, maintaining accountability and protecting the system from spam or anonymous misuse. By tying feedback to authenticated users, the integrity and reliability of the data collected is significantly improved.</td></tr>
<tr><td>XVII</td><td>DeepSeek 1.5b was excluded from model selection due to prompt compatibility issues. While the prompt performed successfully across other models, it consistently failed with DeepSeek 1.5b. Given its relatively low parameter count, excluding the model was deemed more efficient than restructuring the prompts.</td></tr>
<tr><td>XVIII</td><td>To handle rate limits for both the News and Youtube APIs, an alert box was introduced to notify users when the limit was reached.</td></tr>
<tr><td>XIX</td><td>Caching was added to optimize performance. If a user submits a previously searched query, the system retrieves the result from cache instead of initiating a new request.</td></tr>
</table>

APPENDIX H – PROJECT PROPOSAL

Bournemouth University

Department of Computing and Informatics

2024-25 Academic Year Undergraduate Final Year Project

Project Proposal Form

Degree Title: Software Engineering	**Student's Name:** Reece Slade
	Supervisor's Name: Marcin Budka
	Project Title/Area: Sentiment Analysis of Political Bias in Online Discourse

Section 1: Project Overview

1.1 Problem definition - use one sentence to summarise the problem: Online platforms often present varying sentiments—both positive and negative—toward political figures or topics, which can influence public perception and may indicate political bias across different sources of information.

1.2 Project description - briefly explain your project: This project will leverage APIs and web scraping techniques to collect data based on a user's inputted search query. For example, when a user searches for a specific topic, such as "**Donald Trump**," the language across selected online platform(s) will be analysed using sentiment analysis. The results will be visual analytics, providing insights into the overall sentiment—whether positive or negative—across different platforms, helping users determine if those platforms exhibit political bias.

Edited by Dr Nan Jiang, Dr Deniz Cetinkaya, and Dr Andrew M'manga based on Project Handbook Section 4

1

Bournemouth University

Department of Computing and Informatics

2024-25 Academic Year Undergraduate Final Year Project

1.3 Background - please provide brief background information, e.g., client, problem domain:

Digital platforms have dramatically changed the way people access news and information. Social media, online news sites, search engines, and AI-powered tools have become the go-to sources for millions of people seeking updates and insights. However, the content shared on these platforms often contains varied sentiments—positive and negative—towards political figures or topics, which can influence public opinion.

This sentiment, whether intentional or unintentional, may reveal underlying political biases in how different platforms report or discuss political topics. It's important to understand these biases because they significantly shape political conversations, influence how people vote, and affect society's views on political issues.

This project uses sentiment analysis to explore potential political biases in online conversations, shedding light on how different platforms might fuel the polarization of political opinions.

1.4 Aims and objectives – what are the aims and objectives of your project?

This project aims to offer statistical insights into online discussions, helping users understand how political language is used and how sentiment shifts across different platforms. This analysis will help users identify potential political biases in online content and understand how language shapes public perception in digital media.

My objectives of this project include:

- **To collect data** from a range of online platforms—such as social media, AI language models, search engines, and digital news sources—based on user-provided search queries related to political figures or topics.
- **To perform sentiment analysis** on the collected data to determine the tone of the language (positive, negative, or neutral) used in online discussions about these political subjects.
- **To develop a tool** that provides users with visualizations and statistics summarizing sentiment trends, helping them gain insights into platform-specific biases.
- **To evaluate the effectiveness** of sentiment analysis in detecting political bias, considering the accuracy and reliability of different sentiment analysis methods across various platforms.

Section 2: Artefact

2.1 What is the artefact that you intend to produce? The artefact I intend to produce is a web application that functions similarly to a search engine. The user will input a search query, and before

executing the search, they must specify which platform they wish to search across (e.g., Google, CNN, etc.). Once the user enters the query and selects a platform, the system will execute the search and perform sentiment analysis on the results.

2.2 How is your artefact actionable (i.e., routes to exploitation in the technology domain)?

The web app offers a practical solution for identifying political bias and sentiment trends across various online platforms, making it a valuable tool in multiple fields. Researchers, journalists, and content analysts can use it to track and analyse sentiment around political figures, events, or issues across diverse media sources. Beyond politics, the tool can be adapted to uncover biases in other areas, such as marketing, public relations, or brand sentiment analysis, providing broader applications and insights.

Additionally, the app can be integrated with AI-driven tools and platforms to provide real-time insights, enabling automated bias detection in content aggregation systems, news feeds, or social media monitoring tools.

The exploitation of this artefact is possible in both academic and commercial contexts, including the development of enterprise-level bias detection systems, media watchdog services, and tools aimed at improving transparency in digital media.

Section 3: Evaluation

3.1 How are you going to evaluate your work? I will evaluate my work through the following methods:

Accuracy of Sentiment Analysis:

- I will compare the sentiment analysis results generated by the web app against human-labelled data or pre-existing sentiment analysis benchmarks to measure the accuracy, precision, and recall of the sentiment detection model.

User Testing:

- I will carry out user testing with a sample group to evaluate the web app's usability, gather feedback on the user interface (UI), and determine how effectively it presents sentiment analysis results.
- Key metrics for evaluation will include user satisfaction, ease of use, and clarity of the sentiment output.

Performance Evaluation:

- I will assess the performance of the web app, focusing on the speed and efficiency of data scraping, sentiment analysis, and display of results. This will include load times, system responsiveness, and the ability to handle multiple queries concurrently.

Effectiveness of Insights:

- I will evaluate whether the tool effectively helps users identify potential political biases in the platforms they are analysing. This will be done by comparing the sentiment analysis results with external sources or expert opinions about the platforms' political leanings.

3.2 Why is this project honours worthy?

This project is honours-worthy because it takes an interdisciplinary approach, merging advanced sentiment analysis, web scraping, and AI language models to tackle a pressing societal issue: political bias in online discourse.

By creating a tool that helps users identify bias in the content they consume, the project promotes more informed and critical engagement with digital media. It demonstrates a high level of technical expertise in building a web application that integrates diverse data sources, conducts real-time sentiment analysis, and delivers complex insights through an intuitive interface.

The potential addition of a user-driven discussion feature enhances interactivity, encouraging reflection and debate on the biases encountered, which could have significant implications for improving media literacy.

This blend of innovative technology, user-centric design, and contribution to understanding political discourse in the digital era highlights the project's value, making it a strong candidate for honours in Software Engineering while addressing an important societal challenge.

3.3 How does this project relate to your degree title outcomes? This project directly relates to the outcomes of my Software Engineering degree as it involves the application of fundamental software engineering principles and techniques. These include:

System Design and Development:

Bournemouth University

Department of Computing and Informatics

2024-25 Academic Year Undergraduate Final Year Project

The project involves creating a fully functional web application, requiring both front-end and back-end development. This includes integrating external APIs and employing web scraping techniques to gather data from multiple online platforms.

Problem-Solving and Algorithm Design:
To tackle the challenge of detecting political bias, the project incorporates natural language processing (NLP) techniques, machine learning models, and custom algorithms. These components power the sentiment analysis, offering nuanced insights into online discourse.

User-Centred Design:
The project prioritizes usability and user experience, featuring an intuitive interface and data visualizations. These elements help users easily interpret sentiment trends and recognize potential biases across different platforms.

Software Testing and Evaluation:
Comprehensive testing ensures the system performs reliably, with evaluations focused on sentiment accuracy, system functionality, and user engagement. This process guarantees that the application meets its intended goals and provides meaningful insights.

Real-World Application:
By offering a tool to detect political bias in digital media, the project serves a practical purpose. It empowers users to engage more critically with online information, potentially influencing how they form opinions and interact with digital content.

3.4 How does your project meet the BCS Undergraduate Project Requirements?

This project satisfies the BCS Undergraduate Project Requirements by showcasing originality, technical complexity, and practical application. It combines sentiment analysis, web scraping, and AI language models to create a tool that helps users identify political bias across various online platforms.

The project demonstrates technical complexity through real-time data collection, processing large datasets, and implementing advanced machine learning algorithms for accurate sentiment analysis. It also adheres to professional standards by prioritizing user experience, offering an intuitive web interface, and including a user discussion feature to enhance engagement and usability.

By tackling a real-world issue with innovative and technically advanced methods, the project aligns with the BCS criteria, making it a strong candidate for recognition.

3.5 What are the risks in this project and how are you going to manage them? There are several potential risks in this project, and I plan to manage them as follows:

Risk: No data to scrape
Management: I will ensure a reliable data source by using APIs or web scraping tools from platforms like Google, Twitter, or news websites. If necessary, I will utilize services like DataForSEO to access high-quality data.

Risk: Inconsistent sentiment analysis results
Management: I will fine-tune the sentiment analysis models to improve consistency. This will involve testing and validating the models on different types of content and applying techniques like text normalization and context-aware sentiment scoring.

Risk: CAPTCHA and anti-scraping measures
Management: To address CAPTCHA and other anti-scraping mechanisms, I will explore proxy-based solutions or web scraping services that bypass such restrictions. In cases where scraping is not feasible, I will rely on APIs provided by the platforms/data aggregators or choose alternative media services.

Risk: Running out of time
Management: I will prioritize features based on their importance and implement an MVP (Minimum Viable Product) that focuses on core functionalities, such as data collection and basic sentiment analysis. This will allow for iterative testing and development, reducing the risk of incomplete features.

Risk: Project complexity
Management: I will break the project down into smaller, manageable tasks with clear milestones and deadlines. Regular progress checks will help ensure that each component is completed on time, and I will seek guidance from my supervisor when facing particularly challenging technical aspects.

Section 4: References

Bournemouth University

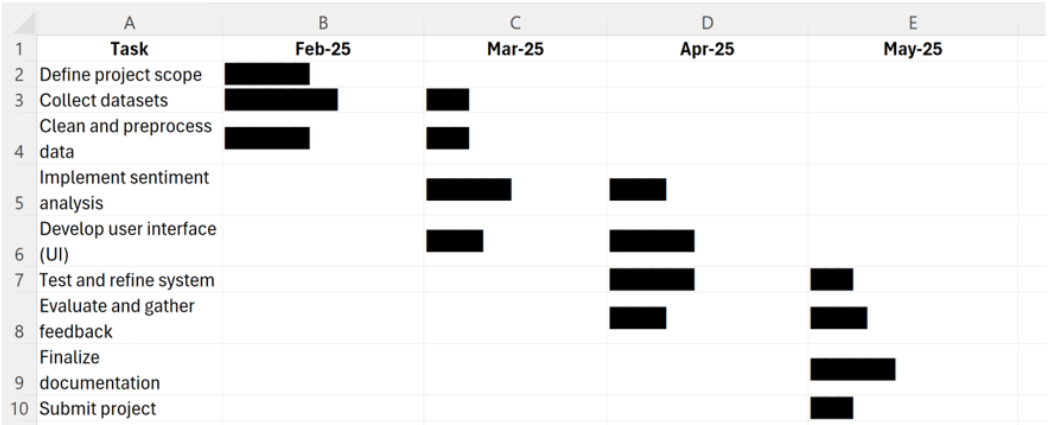
Department of Computing and Informatics

2024-25 Academic Year Undergraduate Final Year Project

4.1 Please provide references.

Kaggle:
Kaggle. (n.d.). *Kaggle*. Retrieved from https://www.kaggle.com
Beautiful Soup:
Richardson, L. (2021). *Beautiful Soup Documentation*. Retrieved from https://www.crummy.com/software/BeautifulSoup/bs4/doc/
Twitter API:
Twitter, Inc. (n.d.). *Twitter Developer Documentation*. Retrieved from https://developer.twitter.com/en/docs
OpenAI API:
OpenAI. (n.d.). *OpenAI API Documentation*. Retrieved from https://platform.openai.com/docs
Ground News:
Ground News. (n.d.). *Ground News*. Retrieved from https://ground.news
DataForSEO:
DataForSEO. (n.d.). DataForSEO. Retrieved from https://dataforseo.com

Section 5: Proposed Plan (please attach your Gantt chart below)



	A	B	C	D	E
1	**Task**	**Feb-25**	**Mar-25**	**Apr-25**	**May-25**
2	Define project scope				
3	Collect datasets				
4	Clean and preprocess data				
5	Implement sentiment analysis				
6	Develop user interface (UI)				
7	Test and refine system				
8	Evaluate and gather feedback				
9	Finalize documentation				
10	Submit project				

APPEDIX I - ETHICS CHECKLIST

Bournemouth University

Research Ethics Checklist

About Your Checklist	
Ethics ID	61806
Date Created	10/01/2025 10:36:02
Status	Approved
Date Approved	13/01/2025 08:56:20
Risk	Low

Researcher Details	
Name	Reece Slade
Faculty	Faculty of Science & Technology
Status	Undergraduate (BA, BSc)
Course	BSc (Hons) Software Engineering

Project Details	
Title	Sentiment Analysis of Political Bias in Online Discourse
Start Date of Project	03/02/2025
End Date of Project	16/05/2025
Proposed Start Date of Data Collection	25/01/2025
Supervisor	Marcin Budka
Approver	Marcin Budka

Summary - no more than 600 words (including detail on background methodology, sample, outcomes, etc.)

Research Summary

This research investigates potential political bias in online information by analysing the language used by companies and organizations when presenting search results. The primary focus is on determining whether the tone of the content associated with specific search queries, such as "Donald Trump," is positive, negative, or neutral. The study will examine how various platforms, including Google, Yahoo, BBC, and CNN, portray these topics.

The research aims to provide clear, statistical evidence to users, illustrating the sentiment of the language used for specific search topics. For instance, if someone searches for "Donald Trump," they will be able to see whether the results on a platform are generally favourable, critical, or neutral. This will help users identify potential bias in how information is presented across different platforms.

The research is motivated by the growing concern over the influence of political bias in shaping public opinion. By providing an objective analysis of sentiment, this project seeks to promote transparency and empower users to make more informed decisions about the information they consume.

The methods involve collecting data through web scraping and APIs, which allow the extraction of search result content from the selected platforms. This data will then be analysed using sentiment analysis tools to classify the tone of the language. The findings will be presented in a user-friendly format, highlighting the sentiment statistics for each platform.

Research Plan

Aims: The primary aim is to analyse political bias in online discourse by examining the sentiment of language in search results across platforms.

Objectives:

- Developing a method to extract and analyse search result data.
- Classifying sentiment as positive, negative, or neutral.
- Presenting results in a clear and accessible format for users.

Reasons for Undertaking the Research: The research addresses the increasing public concern about media and platform bias. By highlighting how companies present information, the project aims to foster awareness and encourage critical thinking about online content.

Methods: Data will be gathered using web scraping and APIs to collect search result content. Sentiment analysis techniques, such as natural language processing, will classify the tone of the language. Statistical tools will summarize the findings, which will be visualized for clarity.

Ethical Considerations

While this research does not involve human participants, several ethical considerations must be addressed:

Data Privacy and Legal Compliance: Web scraping involves extracting data from online platforms, which may raise concerns about violating terms of service or intellectual property rights. To mitigate this, the research will adhere to the platforms' terms and ensure that only publicly available data is collected. Additionally, no personal or sensitive information will be accessed or stored.

Bias and Misuse of Results: The analysis aims to identify bias in a neutral and objective manner. However, there is a risk that the findings could be misinterpreted or misused to support specific political agendas. To address this, the results will be presented with clear disclaimers emphasizing their context and limitations.

Transparency and Reproducibility: The methods and tools used in the analysis will be documented to ensure transparency and allow others to replicate the study. This helps maintain the integrity of the research and prevents misuse.

Minimizing Harm: The research will avoid any actions that could harm the reputation of the companies or platforms analysed. The study's goal is not to accuse but to inform and promote awareness.

Filter Question: Is your study solely literature based?

Additional Details	
Will you have access to personal data that allows you to identify individuals which is not already in the public domain?	No
Will you have access to confidential corporate or company data (that is not covered by confidentiality terms within an agreement or separate confidentiality agreement)?	No

Storage, Access and Disposal of Research Data
Where will your research data be stored and who will have access during and after the study has finished.
Data Storage: During the study, all research data will be stored locally on secure, password-protected devices to ensure privacy and security. **Access During the Study:** Access to the data will be limited to myself, the researcher. If anyone wishes to access the data during or after the study to contribute further work to the project, they will be able to do so with my permission. **Access After the Study:** After the study is finished, the data will remain stored locally. If anyone is interested in continuing or building on

the research, they can access the data with my consent.	
Once your project completes, will your dataset be added to an appropriate research data repository such as BORDaR, BU's Data Repository?	Yes

Filter Question: Will your research study take place outside the UK and/or specifically target a country outside the UK?

Additional Details	
List the European and/or Overseas country where the research will take place	
United States, United Kingdom	
Are you currently a resident of the country named above?	No
Do you intend to remain in/visit the country named above to undertake the research?	No
Do you intend to remain in the UK but recruit participants from the country named above?	No
Explain what you are doing	
This project will involve analyzing publicly available data from platforms such as Google, Yahoo, and CNN, which are based in the United States, and the BBC, which operates in the United Kingdom. The research will target these two countries as they are the primary sources of the data being studied, specifically in relation to political discourse and sentiment analysis.	
By participating in this research, are there any potential risks to participants?	No
Does the country in which you are conducting research require that you obtain internal ethical approval (other than BU ethical approval)?	No

Final Review	
Are there any other ethical considerations relating to your project which have not been covered above?	No

Risk Assessment	
Have you undertaken an appropriate Risk Assessment?	Yes

APPENDIX J – FIRST PROJECT PROGRESS REVIEW

Undergraduate Project First Progress Review

Activity: First Progress Review (FEEDBACK ONLY)

Course: Individual Project 24/25

Name: Reece Slade

Criteria	Yes	To some extent	No
Definition of the problem - Has the problem been defined, has the artefact been identified and have objectives been set?	Yes ✓	To some extent	No
Review of literature and related work - Is there evidence of appropriate research?	Yes	To some extent ✓	No
Criterion Feedback Good progress is being made but work in progress.			
Methodology and Artefact - Is there evidence of appropriate analysis of the problem and design of a solution and appropriate evaluation?	Yes	To some extent ✓	No
Dissertation - Have sections of the dissertation been written and has the Supervisor seen these?	Yes ✓	To some extent	No
Planning & Progress - Is there an acceptable plan for this project and is it being followed?	Yes ✓	To some extent	No
Proposal & Online Ethics Checklist - Are proposal and ethic checklist submitted? Are they approved?	Yes ✓	To some extent	No
Form completed with student	Yes ✓		No

Overall Score

Satisfactory	Requires minor improvement ✓	Requires major improvement (at some risk of failing the project)	Unsatisfactory (at high risk of failing the project)	Invalid - Proposal not approved	Invalid - Proposal not submitted	Invalid - Ethics checklist not approved	Invalid - Ethics checklist not submitted

APPENDIX K – ADDITIONAL FILES

README.md - Contains the instructions to run the application locally, either through a virtual environment or docker setup.

The **Backend** folder contains all components necessary for the server-side operation of the artefact. The **routes** directory holds the Python files responsible for defining application routes and handling HTTP requests. These routes render HTML files stored in the **templates** folder, which use the Jinja2 templating engine to inject dynamic content. Supporting assets such as JavaScript and CSS are organized within the **static** folder, enabling interactivity and styling for each page. Environment-specific variables, including API keys and configuration settings, are stored securely in a **.env** file, which is loaded at runtime to protect sensitive information and maintain modularity across environments. A **.env.example** file is provided to serve as a reference for setting up the virtual environment, while a **.env.example.docker** file is available for Docker-based setups. These example files can be copied and populated with the appropriate values to create the actual .env files required for each environment.