

Anonfeedback.io — Enhancing Student Feedback Collection and Analysis with AI & HCI Techniques



CM4105 Honours Project

Keywords: Real-time feedback, AI in Education, Student Engagement, Feedback Analysis, Educational Data Mining

By

Francesco Gruosso

Student Number: 1806450

Degree Name: Computer Science (Hons)

Department Name: School of Computing

Supervisor: Dr. Roger McDermott

A report submitted as part of the requirements for the degree of Computer Science (Hons) at the
Robert Gordon University

Aberdeen, Scotland — 25th April 2024

Contents

Abstract	3
Introduction	4
Declaration	5
1 Project Scope	6
1.1 Literature Review	6
1.1.1 Participation Rates in Feedback Surveys	7
1.1.2 Anonymity and Feedback Quality in a Live System	7
1.1.3 Sentiment Analysis for Educational Feedback	8
1.2 Requirements Analysis	9
1.2.1 Problem Analysis and Artefact Definition	9
1.2.2 Functional Requirements	10
1.2.3 Non-Functional Requirements	10
1.2.4 Boundaries	10
1.2.5 Adaptability for Project Handover	10
1.3 Conclusions	11
2 System Design & Methodology	12
2.1 System Design	12
2.2 Methodology	14
2.2.1 Human-Computer Interaction (HCI) Considerations	14
2.2.2 Designing a Prompt for the Sentiment Analysis Classification	16
3 Implementation	21
3.1 Technical Specifications & Technologies	21
3.2 Legal & Ethical Considerations	22
3.3 Securing Anonymity	22
3.4 Insight Tools	22
3.4.1 Sentiment Analysis	23
3.4.2 Moderation Layer	25
3.4.3 Wordcloud Visualisation	27
3.5 Instructor Settings	28
3.5.1 Dynamic Event Links	28
4 Testing & Evaluation	30

5	Handover & Future	31
	Glossary	32
	References	33
A	Project Logs	35
A.1	Github Commits	35
B	Source Code	39

Abstract

This thesis presents the development of a real-time feedback system for educational settings, taking advantage of the modern capabilities of Artificial Intelligence (AI), Large Language Models (LLMs) and Sentiment Analysis. The system aims to address the limitations of traditional feedback mechanisms, which often suffer from low participation rates and delayed responses, by introducing an anonymous, immediate, and easy-to-use platform for students to express their sentiments regarding their educational experiences. Utilising GPT models, the system analyses and provides a sentiment classification and reasoning for the student feedback, providing educators with actionable insights in real time. This project explores the implementation, functionality, and potential impacts of integrating advanced AI technologies in educational feedback systems, offering a detailed analysis of its effectiveness in enhancing communication between students and educational institutions. The ultimate goal is to create a continuous improvement cycle in educational settings by enabling timely and accurate responses to student feedback.

Introduction

In early 2020, the COVID-19 pandemic forced universities globally to quickly transition to online or hybrid learning models. This sudden change highlighted the challenges in traditional educational setups in a rapidly digitalised learning environment, including the ineffectiveness of conventional feedback mechanisms, the lack of adequate support for quality online feedback interactions, and a need for better strategies to adapt conventional feedback practices to the online space (Yang, Mak, and Yuan 2021).

Towards the end of the pandemic, in early 2023, a significant shift occurred in the field of Natural Language Processing (NLP) with the rapid ascent of Generative Pre-trained Transformers (GPT). This development had a ripple effect, extending its influence into the educational domain as well, with the rapidly expanding popularity of OpenAI's ChatGPT, a Large Language Model (LLM) built with a transformer architecture capable of replicating human language. According to an article from the Guardian Newspaper reporting Similarweb's analysis of internet traffic (cited in Loh 2023), this model reached over 100 million users by January 2023, establishing itself as the fastest-growing consumer application up to that point.

Considering the shifts in educational practices and advancements in NLP technologies, particularly the rise of GPT models, this project aims to introduce a real-time feedback system for educational settings. The system allows students to provide anonymous, immediate feedback easily. Utilising Artificial Intelligence (AI), specifically Large Language Models (LLMs) and Sentiment Analysis, the data will be analysed to offer timely, actionable insights to educators. This approach aims to enable educational institutions to act promptly on the feedback. This action, often referred to as "closing the loop," ensures a continuous cycle of improvement based on real-time insights.

In a fast-paced environment like higher education institutions, it is easier for students to focus on their classes than the issues that arise during the academic year and report them when they are approached for feedback provision. The lack of immediacy affects the effectiveness of pedagogical adjustments. Additionally, traditional feedback mechanisms usually suffer from low participation rates, reducing the diversity and accuracy of the collected data.

The following sections in the first chapter will delve into the literature concerning existing feedback mechanisms, discuss the role of anonymity in feedback quality, and examine the potential of real-time feedback systems. Finally, the project will explore how Sentiment Analysis can offer valuable insights in an educational context. The ultimate goal is to clarify the requirements for developing such a system, providing a comparative analysis of existing applications and outlining both functional and non-functional requirements.

Declaration

I confirm that the work contained in this Honours project report has been composed solely by myself and has not been accepted in any previous application for a degree. All sources of information have been specifically acknowledged and all verbatim extracts are distinguished by quotation marks.

Francesco Gruosso

Chapter 1

Project Scope

1.1 Literature Review

This literature review, inspired by the structured approach from Tony Clear's paper ([2012](#)) on *Systematic Literature Reviews in undergraduate research*, aims to explore the differences between real-time feedback systems and traditional online feedback forms, particularly in learning settings. Following Clear's method, this review goes through a series of steps including identifying the need for this review, creating research questions, searching for and evaluating relevant studies, gathering and summarising data, and finally, drawing conclusions and documenting the findings. This structured process provides a detailed and well-understood exploration of the chosen research area.

The main research questions for the exploration of the literature are as follows:

RQ1: Does the immediacy of a live feedback system lead to more accurate and actionable insights compared to traditional online forms?

RQ2: How can AI and sentiment analysis enhance the efficacy of feedback in a learning environment?

To explore these questions, a set of keywords to guide the exploration were utilised. The following table outlines these keywords:

Keyword	Scope of Application
Student engagement	Level of student involvement
Real-time feedback	Immediate, interactive responses
Anonymous feedback	Unidentifiable responses
Educational technology	Digital tools in pedagogy
Educational data mining	Data analysis in education
AI in education	AI applications in learning
Sentiment analysis	Evaluating emotional tone
Feedback analysis	Interpretation of feedback
Performance monitoring	Tracking academic progress
Course satisfaction	Student contentment levels

Table 1.1: Keywords Utilised in the Literature Review

The selected papers were considered relevant based on different metrics, which are outlined below. These were not strict criteria, but rather guidelines to help in the selection process:

1. The paper originates from a reputable or credible academic publisher, such as peer-reviewed journals or books.
2. The number of citations is taken into account.
3. The paper has been published within the last decade, ensuring its relevance to the current discussion; this is particularly crucial for technical papers, where advancements occur rapidly.
4. The study conducted experiments on a statistically significant sample size, ensuring the reliability and validity of the findings.

1.1.1 Participation Rates in Feedback Surveys

A paper by Marcus et al. (2007) on factors influencing web survey response rates was selected as one of the first papers to discuss because it provides important insights that can inform the design decisions of the feedback system this project aims to create.

In particular, understanding what drives participation and response rates in web-based surveys is crucial for maximising engagement with a feedback system. The study findings indicate that topic salience, survey length, incentives and personalised feedback (providing the study results to respondents) impact response rates, these lessons can be applied to the configuration of the live feedback system.

1.1.2 Anonymity and Feedback Quality in a Live System

To elicit a higher participation rate, the artefact that will be output from this research aims to anonymise the students who engage. Various papers explored during the literature review indicate that anonymising feedback can result in better student performance and more critical, honest feedback. A study by Lu and Bol (2007) investigated the impact of anonymity in e-peer review processes on student writing performance and the quality of feedback.

The study revealed that students were more critical in providing feedback when the review was conducted anonymously. The anonymous group provided negative comments and lower ratings more frequently, confirming that students working in anonymous conditions were more critical than those in identifiable conditions.

Students in the anonymous group found it easier to give and receive honest and critical feedback. They associated critical feedback with constructive feedback, as it helped the authors make the necessary improvements to their paper. In contrast, students in the identifiable group were less likely to share critical feedback and were concerned about their peers' reactions, indicating a tendency towards being less critical and objective.

A different study by Bergstrom et al. (2011) examined an anonymous classroom feedback live system called the Fragmented Social Mirror (FSM) and found benefits to anonymity in

eliciting more critical and honest student feedback. The FSM allowed students to provide real-time anonymous feedback during lectures via icons representing questions, information, and agreement/disagreement.

Students reported feeling more comfortable providing critical feedback under anonymous conditions. However, the study also found completely anonymous feedback led to some disruptive or irrelevant comments. The researchers suggested retaining anonymity in the public display but enabling instructors to identify students privately to maintain accountability. While this suggestion may not be directly feasible to apply to the current project as it would defeat its point, it does emphasise the importance of integrating a mechanism to screen out off-topic or insincere feedback.

The methodology chosen for this project to tackle this issue follows OpenAI's (2023) article about how utilising their latest and most accurate model to date GPT-4 to handle content moderation, could produce more consistent labelling for harmful or misused text user inputs through an iterative that considers carefully defined policies fed to the model.

1.1.3 Sentiment Analysis for Educational Feedback

The analysis of student feedback with Generative Pre-trained Transformers (GPT) is a relatively unexplored territory. A recent study from Kheiri and Karimi (2023) indicates that utilising GPT models for sentiment-analysis tasks via a prompt-tuning-based approach substantially outperforms more traditional machine learning classification techniques.

Their investigation of the literature showed that previous sentiment classification techniques included assigning positive and negative scores to words to determine overall sentiment. Next came machine learning techniques like Naive Bayes and Support Vector Machines (SVM), which were data-driven and better captured context. However, they still had limitations. Most recently, powerful transformer models like BERT have pushed state-of-the-art by learning contextual representations.

This research shows that fine-tuning GPT models and crafting prompts to guide the models substantially improves performance on sentiment analysis of tweets over traditional classifiers. Their GPT approach achieves much higher accuracy than the aforementioned methods in identifying sentiment polarity. This indicates the potential of leveraging large pre-trained language models like GPT for gaining more nuanced insights from unstructured student comments and surveys.

As educational institutions increasingly collect open-ended feedback, employing more recent natural language processing techniques could enrich the insights extracted and better inform instructional decisions and policy.

1.2 Requirements Analysis

Considering the findings of the literature, this section defines the requirements and boundaries for creating a piece of software that aims to tackle the issue at hand. These requirements have been defined using the MoSCoW approach which defines must, should, could and will not have. These have then been prioritised and explained.

1.2.1 Problem Analysis and Artefact Definition

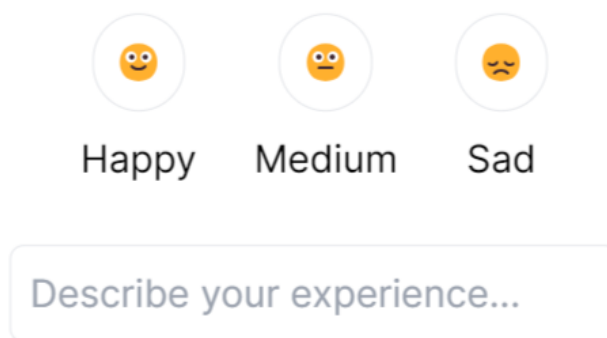
As discussed in Section 1.1.1 ([Participation Rates in Feedback Surveys](#)), often users find feedback processes tedious and time-consuming, unless they are naturally interested in the topic, or if the survey is short.

To address the problem at hand, creating a user-friendly and easy-to-use piece of software is pivotal in encouraging students to provide feedback. Several factors can deter students from engaging with the feedback system, and addressing these through a well-designed user interface can significantly enhance participation rates.

A visually engaging feedback submission method will be utilised to encourage users to provide feedback. An initial medium of a simple yet effective triad of emoticons has been selected to establish the user's mood for providing feedback, ensuring an intuitive and user-friendly experience. Emoticons can facilitate easier communication of emotions and add emphasis to the feedback context (Manganari [2021](#)), and could also help with its analysis.

To further enhance usability, selecting an emoticon will unveil associated tags reflecting that emotion. For instance, clicking on the smiley face could reveal multi-selectable tags such as "Engaging", "Interesting", etc., with an option for additional textual feedback.

We value your feedback



Happy Medium Sad

Describe your experience...

Your feedback is anonymous and will be analysed with sentiment analysis.

Figure 1.1: A simple wireframe of a possible User Interface created on Figma (subject to change).

1.2.2 Functional Requirements

- The system must allow real-time submission of feedback through a web interface.
- The landing page must display a triad of emoticons (Happy, Neutral, Unhappy) as the primary feedback mechanism to capture immediate sentiment.
- The system must provide a way for students to input optional text-based comments.
- The system must employ a prompt-based classification model, based on GPT-3.5 or GPT-4, to perform sentiment analysis on optional text-based feedback.
- The system should store the feedback given by the students, and provide a way to visualise such feedback as a word cloud.
- The system should provide the reasoning from the language model for the classification, to offer instructors insights about the sentiment trends.
- The system could provide a content moderation mechanism to filter out hateful or non-relevant feedback.

1.2.3 Non-Functional Requirements

- The system will not store personally identifiable information to maintain anonymity.
- The system should be accessible from various devices, including laptops and smartphones.
- The system should offer a simple, intuitive user interface, requiring no more than three clicks to submit feedback.

1.2.4 Boundaries

- The system will not offer manual sentiment tagging for comments, relying solely on the language model for this task.
- The system will not host an open API for data retrieval or modification.
- The system will not allow students to edit or retract the feedback once submitted to ensure the integrity and immediacy of the sentiment capture.

1.2.5 Adaptability for Project Handover

Considering that this project could be handed over to another team or individual, it is essential to build the system with adaptability and maintainability in mind. This entails:

- Well-documented code and architecture.
- A modular design that allows for easy replacement or enhancement of components.
- Usage of well-established libraries and frameworks if possible, to ensure a broad pool of developers can work on it.

1.3 Conclusions

From the literature review, several key insights emerge regarding the potential benefits of a real-time anonymous feedback system over traditional survey methods.

Immediacy and ease of access are crucial for maximising participation rates. Offering easy emoji-based feedback submissions could boost engagement versus cumbersome surveys. Anonymity helps elicit more critical, objective feedback from students without concerns over being "exposed" to the instructors' authority. This can provide more accurate insights.

Live feedback enables rapid detection of issues and timely pedagogical adjustments, creating a feedback loop for continuous improvement. Delayed surveys fail to capture the pulse of student sentiment. Large language models like GPT-3.5 show great promise in sentiment analysis of unstructured textual comments and have little or no need for pre-processing techniques such as spell-checking, and extracting nuanced insights versus rules-based classifiers.

Chapter 2

System Design & Methodology

2.1 System Design

The first design for the system, illustrated in Figure 2.1, ended up being a solid base to build upon in the implementation phase, as it was carefully planned and thought out to satisfy the Scope's requirements.

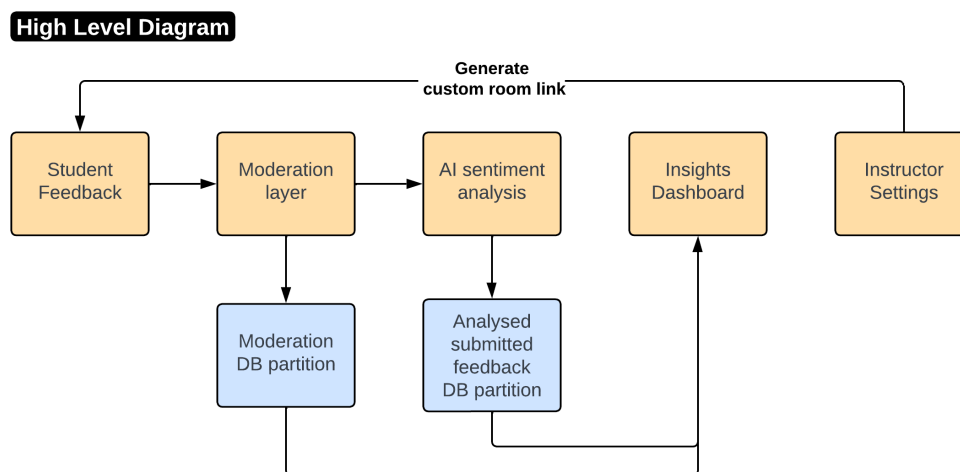


Figure 2.1: High-Level System Flow Diagram

The idea is that there are two entry points for the application:

- The Student Feedback view, which students can access without the need for authentication. This, in practice, would be the landing page students would interact with, and provide feedback from.
- The Instructor view, which would require the instructors to authenticate, in order to safeguard the feedback, which is meant to be confidential.

This diagram naturally evolved into a more technical one, as seen in Figure 2.2, which outlines some of the technologies' flow, described more in depth in the [3.1 Technical Specifications & Technologies](#) section.

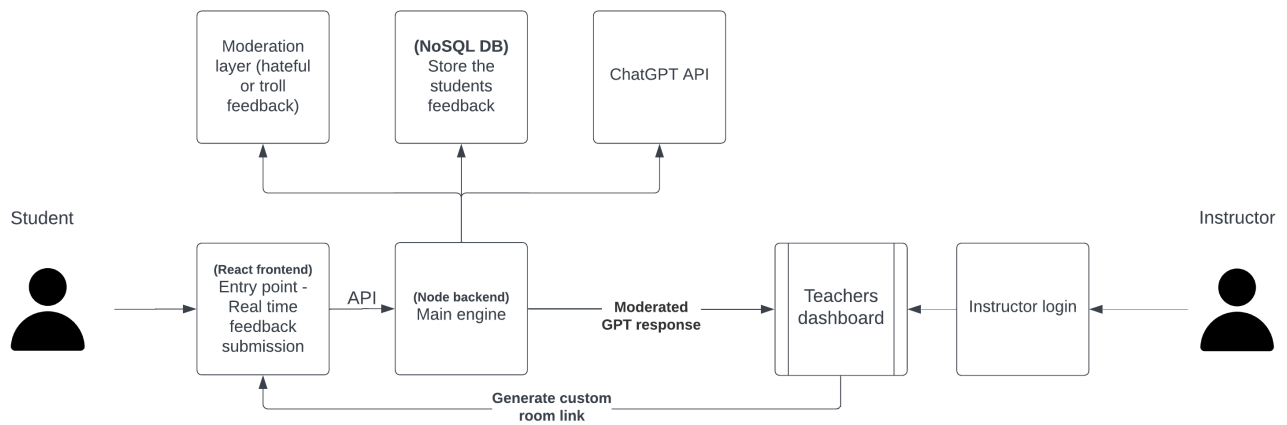


Figure 2.2: Anonfeedback Technical Diagram

To see the full system model from a different perspective, I designed the Venn Diagram shown below in Figure 2.3, which is meant to highlight how the different components of the system need to work together to close the feedback loop effectively.

Closing the feedback loop effectively

Grusso Francesco 2024



Aberdeen, Scotland

Figure 2.3: A Venn Diagram showing the system model

The overlapping areas represent the interdependencies among these components. For example, acting on feedback relies on AI GPT classification and Insight Tools, while clarifying the anonymity of the system delivery ties into Good HCI Considerations.

2.2 Methodology

2.2.1 Human-Computer Interaction (HCI) Considerations

One of the most important parts of this project is to tackle the low participation rate for this type of form, as there is no immediate reward for a feedback submitter. Therefore anything that helps reducing the user's cognitive load is paramount.

For this project, the use of a minimal user interface with as few clicks and interactions as possible is supported by the principles discussed in a Nielsen Norman Group article "Minimize

Cognitive Load to Maximize Usability” (Whitenton 2024).

The article highlights the importance of minimising cognitive load, which refers to the mental resources required to operate a system. It emphasises that human brains have limited processing power, and when the amount of information exceeds this capacity, users may miss important details or abandon the task altogether.

To address this, the project incorporates design elements such as the use of emojis to represent sentiment options, as shown in Figure 2.4. By utilising familiar visual elements like emojis, the interface builds on existing mental models, reducing the cognitive load required for users to understand and interact with the system.

I also chose “Your Anonymous Voice Matters” as the main header for the app, since it makes it very obvious that the feedback cannot be tied up to them, which as previously discussed, tends to elicit more honest feedback.



Figure 2.4: The view students see when first interacting with the website

As seen later in Figure 2.5, the use of guided sentences and pre-defined tags to help users provide feedback. This approach aligns with the article’s recommendation to offload tasks from users by minimising the amount of decisions they need to make.

Further considerations on cognitive load

According to the highly regarded paper *“Integrating cognitive load theory and concepts of HCI”* by Hollender et al. (2010), which is published in a journal with an impressive impact score of 9.9 and has been cited over 300 times, one practical application to reduce cognitive load in user interfaces is to employ techniques like progressive disclosure to simplify the initial presentation and only introduce more advanced functionality when needed.

In my application, I have used guided sentences and tags to allow users to express their feedback in a structured and progressive manner. By providing pre-defined options for common feedback points, I aim to reduce the extraneous cognitive load imposed on users when

they need to formulate and articulate their thoughts from scratch. This approach aligns with the split-attention principle discussed in the paper, as users do not have to mentally integrate information from multiple sources to provide their feedback.

Additionally, the use of tags and guided sentences prevents redundancy, another principle highlighted in the paper for minimising extraneous load. By progressively disclosing additional input fields only when needed (interacting with the landing page's emojis), my application avoids overwhelming users with too many options upfront, mitigating the intrinsic cognitive load associated with the feedback task.

Figure 2.5: The view students enter when interacting with the happy emoticon

Overall, this user interface design approach, informed by cognitive load theory and human-computer interaction principles, aims to optimise the cognitive resources users can dedicate to the core task of reflecting on and providing meaningful feedback.

2.2.2 Designing a Prompt for the Sentiment Analysis Classification

In the SentimentGPT paper by Kheiri and Karimi (2023), they utilised a prompt [0] for sentiment analysis of Twitter posts, assigning scores from 0 to 2 for negative, neutral, and positive sentiments, respectively:

[0] *“As a social scientist, Your task is to analyze the sentiment of a series of user tweets extracted from Twitter. Please assign a sentiment score from 0 to 2 for each tweet, where 0 signifies negative sentiment, 1 indicates neutral sentiment, and 2 corresponds to positive sentiment. In situations where the sentiment is difficult to definitively classify, please provide your best estimation of the sentiment score.”*

However, their strategy did not cover the generation of a structured JSON output, which is needed in this project in order to process the sentiment and reasoning in the easiest way possible. To address this, a custom prompt was developed, instructing the GPT-4 model to analyse the sentiment of a given text and provide the output in a JSON format. This JSON output should include the overall sentiment, main sentiment, key phrases, explanations for those phrases, and the main reasoning behind the sentiment classification.

In order to come up with a tangible prompt for the task, I looked into some prompt engineering strategies mentioned on Microsoft Learn (2024), OpenAI Documentation (2023), and IBM's article on tips for writing foundation model prompts (2024) to improve the quality of the sentiment classification.

As mentioned on IBM's tips (2024): *"Including one example in your prompt is called one-shot prompting, including two or more examples in your prompt is called few-shot prompting, and when your prompt has no examples, that's called zero-shot prompting."*

For this task, I ran a number of zero-shot to k-shot prompting experiments on some of the tweets outlined in Kheiri and Karimi's paper (2023), and some selected ones from a student feedback dataset (Wang et al. 2023). The following Table 2.1 presents the results of experiments conducted to evaluate the performance of a sentiment analysis model. The columns in the table provide the following information:

- **Iteration:** This column indicates the iteration of the experiment cycle.
- **Prompt:** The prompts used in each iteration are represented by numbers in square brackets ([1]).
- **Methodology:** This column specifies the methodology used for each iteration. In this case, the methodologies are "Zero-shot" and "One-shot," which are common techniques in natural language processing tasks.
- **Model:** The column "Model" indicates the specific model used in the experiments, which is GPT-4 in all iterations.
- **Output is JSON (y/n):** This column shows whether the output of the GPT model was in a valid, non-broken JSON format.
- **Correct Classification (y/n):** This column indicates whether or not the feedback submitted was classified with the same value of its true label.

Iteration	Prompt	Methodology	Model	Output is JSON (y/n)	Correct Classification (y/n)
1	[1]	Zero-shot	GPT-4	35/40 (87.5%)	37/40 (92.5%)
2	[2]	One-shot	GPT-4	40/40 (100%)	39/40 (97.5%)
3	[3]	Few-shot	GPT-4	38/40 (95%)	38/40 (95%)

Table 2.1: Prompt-Engineering Iterations Results

My first attempt to output a sentiment classification and JSON output was a simple zero-shot prompt.

[1]

“Analyze the sentiment of the following text (positive/neutral/negative) and provide the output in a JSON format: Feedback: \$feedbackText”

In the first iteration, the model produced valid JSON output 87.5% of the time (35 out of 40 instances). While this is quite remarkable, considering the simplicity of the prompt, I decided to add an example of the structure I wanted from the model’s output.

Because this iteration’s classification results were correct 92.5% of the time, and particularly, the incorrect predictions were related to the model getting confused with sarcasm in the feedback text, I also added the following sentence to the second iteration’s prompt:

...if there is sarcasm, consider marking the main sentiment as the opposite and provide reasoning

Find below the full prompt for the second iteration:

[2]

```
1 Analyze the sentiment of the following text. Provide the output in a JSON format:
2
3 {
4   "sentiment": "positive", (positive/negative/neutral/sarcasm/mixed, etc.), if more
   separate with commas
5   "mainSentiment": "(positive/negative) only", (main sentiment of the feedback)
6   "phrases": ["really enjoyed", "great experience"],
7   "explanations": {
8     "really enjoyed": "indicates strong positive feeling",
9     "great experience": "expresses overall satisfaction"
10  },
11  "mainReasoning": "The feedback contains phrases that indicate overall satisfaction
   and positive feelings. The main sentiment is positive."
12
13   if there is sarcasm, consider marking the main sentiment as the opposite and provide
   reasoning
14 }
15 Feedback: \${feedbackText}
16 }
```

The the second iteration of experiments surprisingly produced valid JSON output 100% of the time (40 out of 40 tests). The Correct Classification (y/n) results were also really good, achieving an amazing 39/40 (97.5%) number of correct predicitions.

I still wanted to see whether adding more examples, and adding emotional stimuli into the prompt would somewhat improve the performance or the clarity of the reasoning. In a recent paper, *“Large Language Models Understand and Can Be Enhanced by Emotional Stimuli”* (Li et al. [2023](#)), the authors propose a novel approach called “EmotionPrompt” to incorporate emotional stimuli into prompts for large language models (LLMs).

[3]

```
1 Analyze the sentiment of the following text. Provide the output in a JSON format:
2
3 {
4   "sentiment": "positive", (positive/negative/neutral/sarcasm/mixed, etc.), if more
   separate with commas
```

```

5    "mainSentiment": "(positive/neutral/negative) only", (main sentiment of the feedback)
6    "phrases": ["really enjoyed", "great experience"],
7    "explanations": {
8        "really enjoyed": "indicates strong positive feeling",
9        "great experience": "expresses overall satisfaction"
10    },
11    "mainReasoning": "The feedback contains phrases that indicate overall satisfaction
and positive feelings. The main sentiment is positive."
12
13    if there is sarcasm, consider marking the main sentiment as the opposite.
14 }
15
16 Full sarcasm example:
17 {
18     "sentimentAnalysisResult": {
19         "sentiment": "sarcasm, negative",
20         "mainSentiment": "positive",
21         "phrases": [
22             "I hate",
23             "#NOT"
24         ],
25         "explanations": {
26             "I hate": "indicates strong negative feeling",
27             "#NOT": "expresses sarcasm"
28         },
29         "mainReasoning": "The feedback contains phrases that indicate strong negative
feelings and sarcasm. Because of the sarcasm, the sentiment is positive."
30     }
31 }
32
33 ensure you only return ONE JSON object. IT WILL BREAK and the WORLD WILL END if you
return more than one.
34
35 Feedback: "${textFeedback}"';

```

Unfortunately, in my specific case, the results were actually worse when using Emotion-Prompt, in both JSON output and correct classification, making the one-shot prompt [\[2\]](#) the best performing and final prompt for this task.

Logo Design

When designing the logo, I wanted to avoid anything too elaborate or complex. My goal was to create a visual representation that would stand out clearly on mobile devices and computer browsers (where it is currently used as a Favicon, which I believe stands out compared to other tabs). I settled on a minimalist smiley face design created on Adobe Illustrator, which captures the essence of the emoji-based feedback system at the core of the application. The smiley face conveys a friendly and approachable vibe, aligning with the

overall nature of the platform.



Figure 2.6: Anonfeedback Logo

The eyes of the smiley face are thought to represent two faces looking at each other, symbolising the feedback loop between students and instructors — also a key aspect of the app’s functionality. This subtle yet meaningful detail was thought to add some depth, and sort of an Easter egg, to the otherwise straightforward logo.

The end result is a seemingly silly design, but purposefully made to look friendly, positive, and be a representation of the app’s core purpose.

Chapter 3

Implementation

3.1 Technical Specifications & Technologies

I have carefully selected the technologies for this project to ensure a smooth implementation of the set objectives. I have chosen the MERN stack as the base, using MongoDB for its scalable and flexible document-based data storage, Express for its minimalistic and efficient web application framework, React for its reactive and component-based approach to building user interfaces, and Node.js for its event-driven, non-blocking I/O model that allows for efficient handling of GPT API calls.

The authentication for instructors uses the Passport library in the backend, and is handled with JSON Web Tokens, meaning that every request that comes from any private route sends a JSON Web Token (JWT) in the authorization header of the HTTP request to the backend API.

For the visual design itself, I decided to create everything with vanilla CSS, as it allowed me to create custom designs from scratch, and make the app more unique.

To facilitate deployment, I have opted for hosting the Node backend on Google Cloud through a Docker container, which provides a consistent and isolated environment for the application, ensuring scalability and ease of maintenance. For the frontend, I have chosen Netlify for its easy deployment process and cost-effectiveness, as it offers a free tier that aligns with my needs that has not given me any issues for the past four years that I have been using it.

Additionally, I have incorporated Express.js for its simplicity and flexibility in building web applications, TypeScript for its static typing, Vite for bundling and optimising web assets, and Visx for creating high-quality data visualisation components within the React ecosystem.

During the implementation, I have maintained the system architecture as modular as possible, ensuring that the system is as scalable as possible.

The domain name "anonfeedback.io" has been carefully chosen for its catchy name and self-explanatory nature, making it easily accessible and inviting for students looking to provide anonymous feedback.

3.2 Legal & Ethical Considerations

As with any system handling personal data, legal and ethical considerations were carefully evaluated during the implementation process. To ensure compliance with the General Data Protection Regulation (GDPR) and protect user privacy, strict data minimisation principles were followed. Only essential information required for the system's functionality is collected and stored, minimising the risk of potential data breaches or misuse.

Furthermore, the anonymous nature of the feedback system eliminates the need to store personally identifiable information, further safeguarding user privacy. Any feedback data collected is securely stored and processed solely for the intended purpose of providing instructors with valuable insights, without compromising the anonymity of the feedback providers.

It is worth noting that the integration of language models, such as GPT, introduces additional privacy and ethical concerns. While these models offer powerful natural language processing capabilities, their training data and inner workings are often opaque, raising questions about potential biases and the handling of sensitive information. Ongoing research and refinement of these models are necessary to address these concerns and ensure responsible use in real-world applications.

3.3 Securing Anonymity

In order to ensure anonymity, the payload sent from the frontend to the server purposefully does not include any identifiable information.

```
1 {  
2   "feedback": {  
3     "selectedEmoji": "neutral",  
4     "selectedTags": [  
5       "The class content was as expected"  
6     ],  
7     "feedbackText": "This is feedback"  
8   }  
9 }
```

3.4 Insight Tools

The core elements for the instructors' view currently contain three main "Insight Tools", these were created to help instructors better analyse and consider the submitted feedback. The main view contains two sidebars: The "Filters", and the "Insight Tools".

3.4.1 Sentiment Analysis

The Sentiment Analysis tool is the heart of the Instructors Dashboard, this section contains all the students' feedback, already classified and coming sorted by latest from the backend.

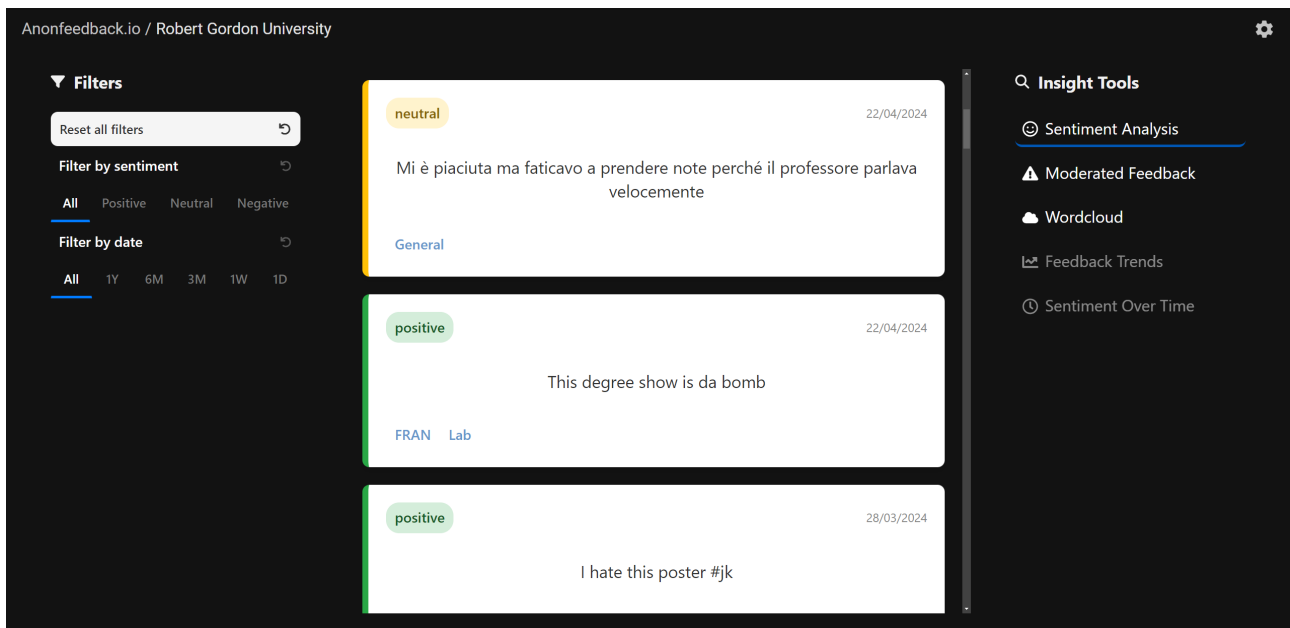


Figure 3.1: The Sentiment Analysis Tool View

One of the benefits of using the GPT-4 model, as shown in the Figure 3.5 above and talked about in the previous sections, is that the model already captures linguistic nuances, different languages, and sarcasm, therefore removing the need to pre-process the text like in traditional machine-learning text processing.

Hovering on any card in this view, reveals a magic wand icon, that once interacted with, shows the reasoning for the classification, the selected tags, and the custom event (if not classified as General feedback) details.

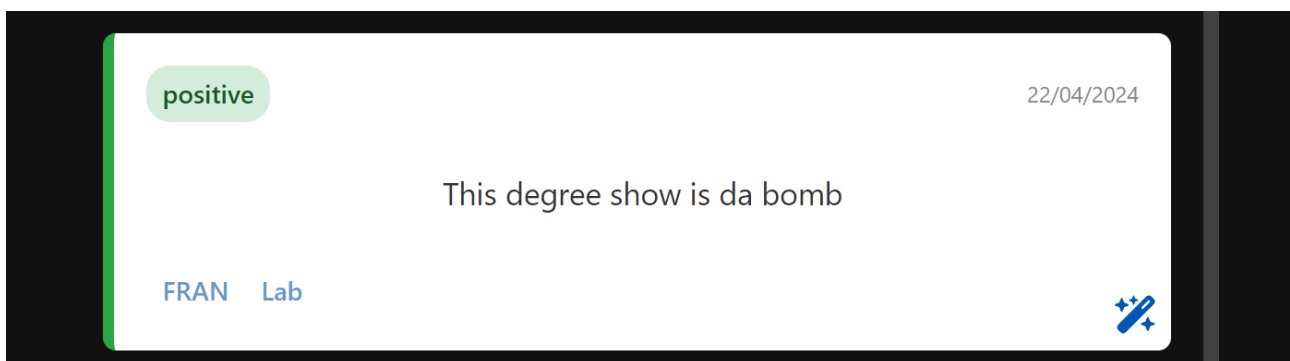


Figure 3.2: The Magic Wand Icon appearing on Card Hover



Figure 3.3: The Client Requesting the Classification Reasoning from the Backend

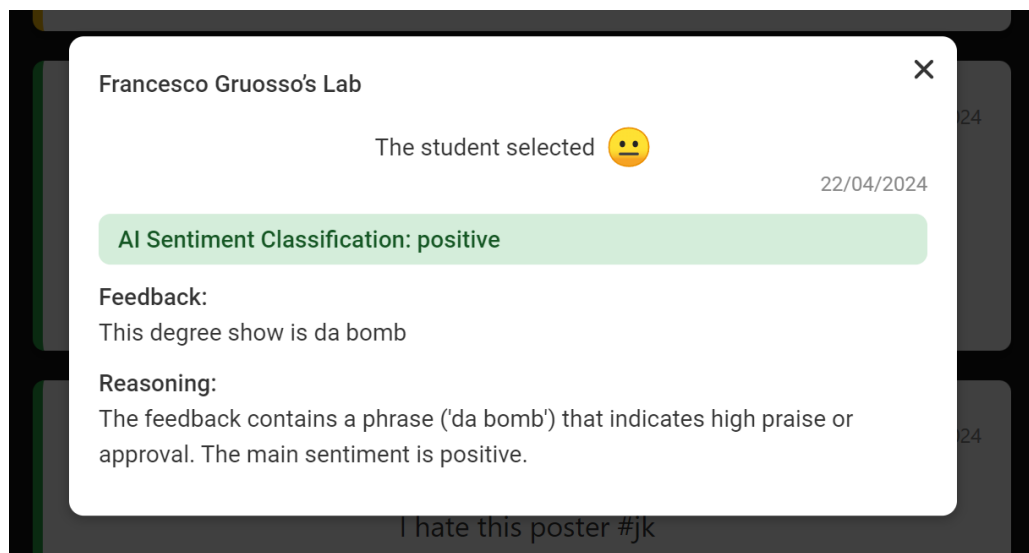


Figure 3.4: Feedback Insights

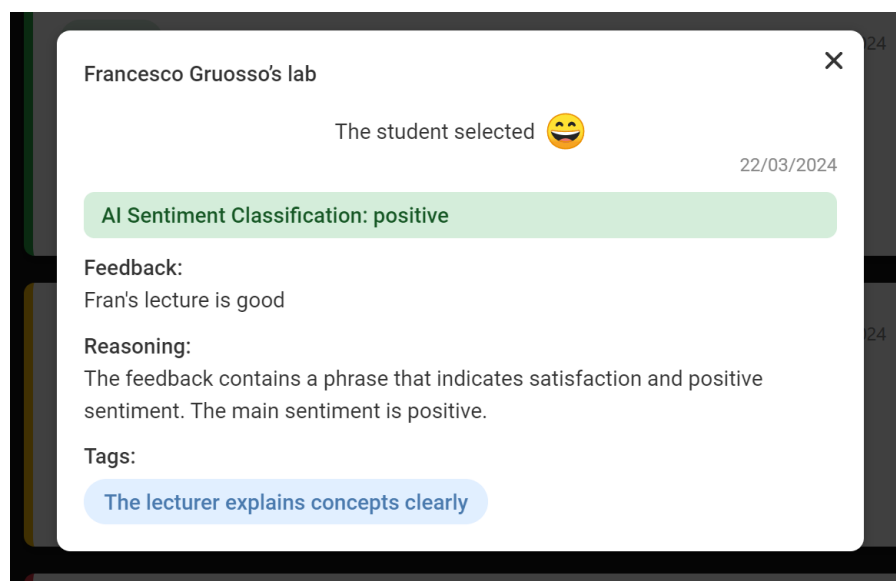


Figure 3.5: Feedback Insights with Tags

3.4.2 Moderation Layer

Initially, as defined in the [1 Project Scope](#) Chapter, my intention was to follow the approach outlined in OpenAI's article on employing their state-of-the-art GPT-4 language model for content moderation in order to minimise harmful or non-sincere feedback polluting the useful data.

The idea involved an iterative prompt-based strategy, where carefully defined policies would be fed to the model, enabling more consistent labeling of harmful or misused text inputs. However, as the project progressed into the implementation phase, OpenAI released a dedicated model available through API ([2024](#)) specifically designed for moderation tasks, prompting me to adapt this new tool for the project's needs.

Below I outline the relevant output from OpenAI's moderation API documentation ([2024](#)), to which I pass the student feedback before running the sentiment analysis function.

```
1      {
2          "id": "modr-XXXXX",
3          "model": "text-moderation-007",
4          "results": [
5              {
6                  "flagged": true,
7                  "categories": {
8                      "sexual": false,
9                      "hate": false,
10                     "harassment": false,
11                     "self-harm": false,
12                     "sexual/minors": false,
13                     "hate/threatening": false,
14                     "violence/graphic": false,
15                     "self-harm/intent": false,
16                     "self-harm/instructions": false,
17                     "harassment/threatening": true,
18                     "violence": true
19                 },
20                 ...other properties that are not relevant for this project...
21             }
22         ]
23     }
```

The implemented tool resulted in a view containing the moderated feedback. Although submitting harmful feedback returns a 403 status to the user (forbidden), with a message to submit non-harmful feedback, this feedback is saved in a separate database partition that never reaches the sentiment analysis stage, and then sent to the view seen in [Figure 3.6](#), and is saved to monitor this sort of behaviour, and also make sure no false positives have been flagged.

```

1 if (result.results[0].flagged === true) {
2     // Find which categories have been flagged as inappropriate
3     const flaggedCategories = Object.keys(
4         result.results[0].categories
5     ).filter((key) => result.results[0].categories[key]);
6
7     const message = 'Your feedback cannot be posted as it contains content that may be
8     considered ${flaggedCategories.join(
9         ", "
10    )}. Please review our community guidelines and ensure your comments are respectful
11    and appropriate.';
12
13    // 403 Forbidden
14    res.status(403).json({
15        message: message,
16        flaggedCategories: flaggedCategories,
17    });
18
19    // Save the flagged feedback to the database.
20    saveFlaggedFeedback(feedback.feedbackText, flaggedCategories);
21
22    return; // Ensure no further processing is done
23 }
24
25 // If the feedback is not flagged, proceed with sentiment analysis
26 const sentimentAnalysisResult = await sentimentAnalysis(
27     feedback.feedbackText
28 );

```

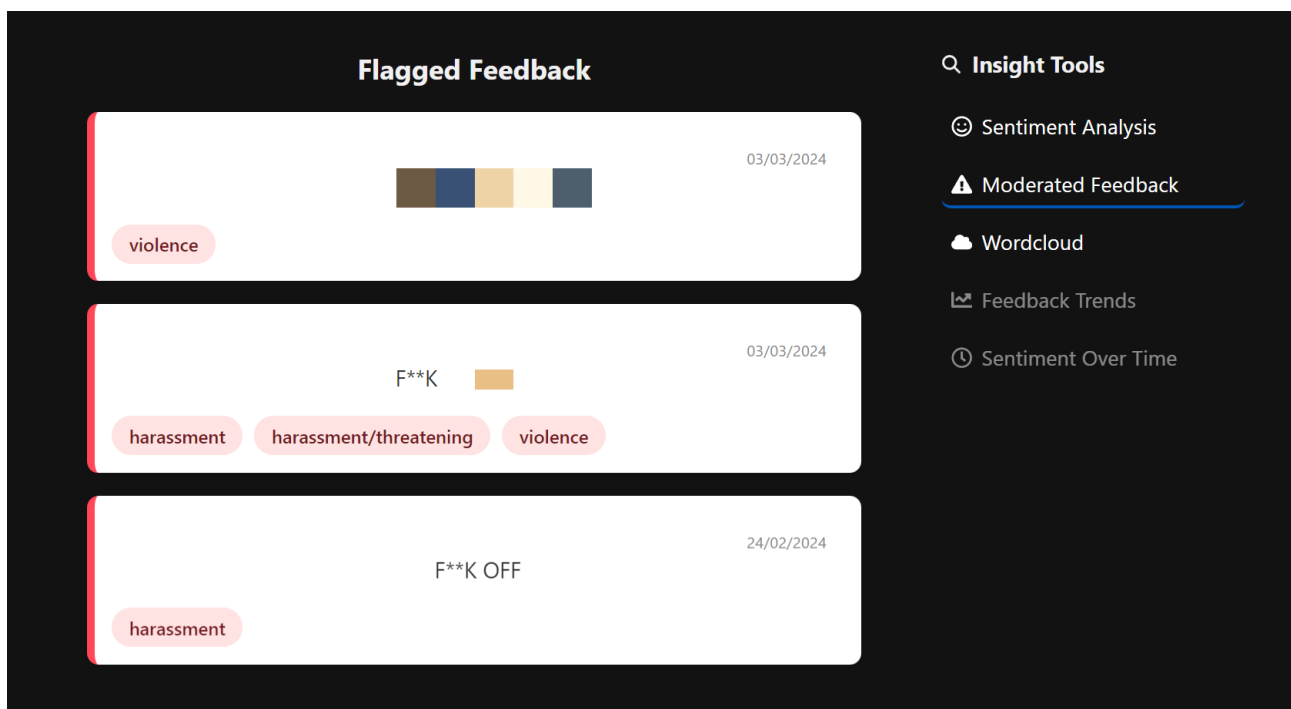


Figure 3.6: Anonfeedback Flagged Feedback Tool

3.4.3 Wordcloud Visualisation

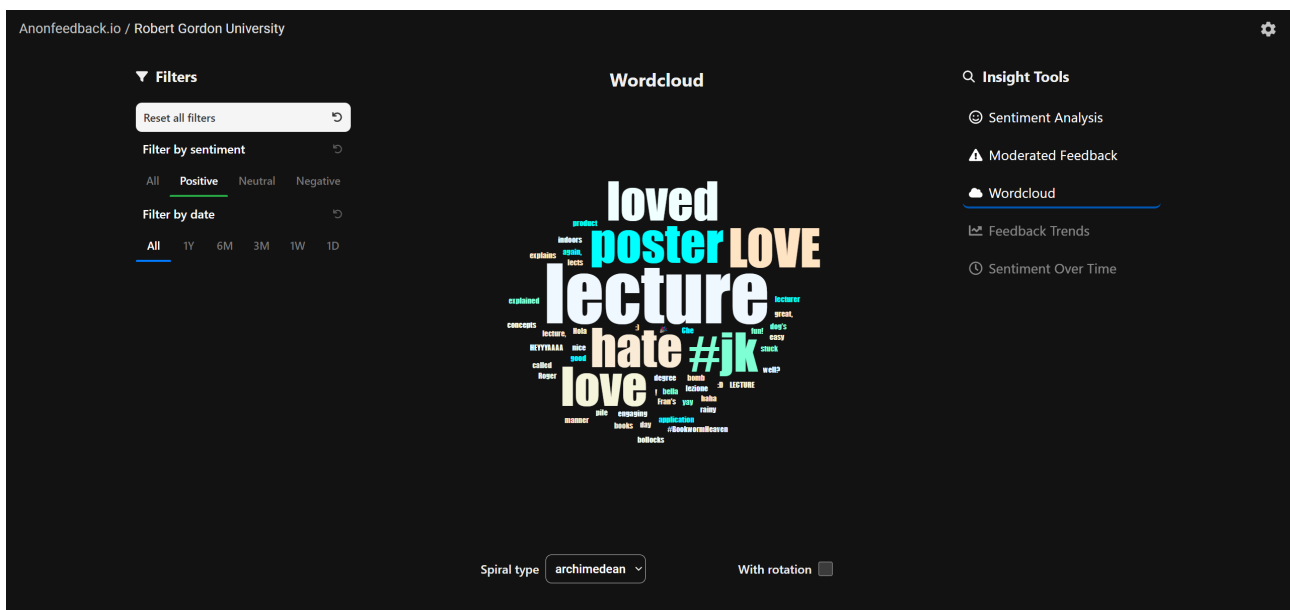


Figure 3.7: Wordcloud filtered by Positive Sentiment

The paper *"Get Your Head into the Clouds: Using Word Clouds for Analyzing Qualitative Assessment Data"* by Concetta A. DePaolo and Kelly Wilkinson (2014) provides a compelling case for the use of word clouds as a tool for analysing qualitative data, such as student feedback. The authors argue that word clouds can serve as effective graphical representations of text data, allowing instructors to quickly identify patterns, common themes, and key concepts expressed by students.

I find the paper's findings highly relevant to my project, where I aim to display word clouds of student feedback to instructors. By presenting the most frequently used words in a filterable and visually appealing format, word clouds can provide instructors with an immediate overview of the sentiments expressed by students, whether they are positive, neutral, or negative.

The ability to dynamically display word clouds over different time periods in the implemented filters, can offer instructors valuable insights into the evolving perceptions and experiences of their students. This longitudinal perspective resonates with the paper's recommendation to use word clouds for *"programmatic assessment using longitudinal data"*, enabling instructors to track changes in student sentiment and address emerging concerns or challenges promptly.

The visualisation is implemented with the Visx library from Airbnb ([2024](#)), which I found the most beautiful and easy to adapt library for a word cloud in React. All of the filtered feedback text is pre-processed, removing stopwords of the English language to avoid noise using an NLTK dictionary I found on GitHub (sebleier [2010](#)). The word frequencies are then calculated in order to show the size of the words in the visualization.

3.5 Instructor Settings

Similarly to the Insight Tools View, the Instructor Settings also contain two sidebars. The one of the left is just a navigation handler, and allows the instructor to change their unique ID, which is used as part of the generated URL for the custom events rooms.

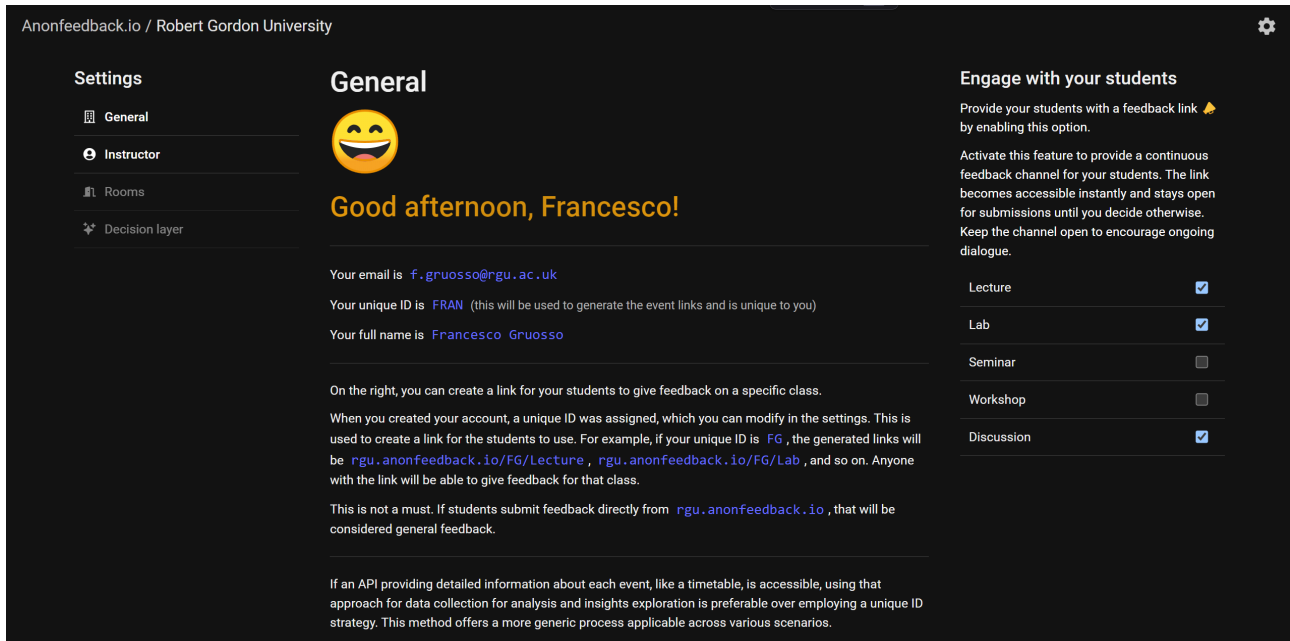


Figure 3.8: Instructor Settings — Generating Custom Links

3.5.1 Dynamic Event Links

The software contains this feature, accessible from the private settings, that allows instructors to generate custom feedback links for different types of events like lectures, labs, workshops, and discussions. These links serve as continuous feedback channels, enabling students to provide their input and engage in an ongoing dialogue.

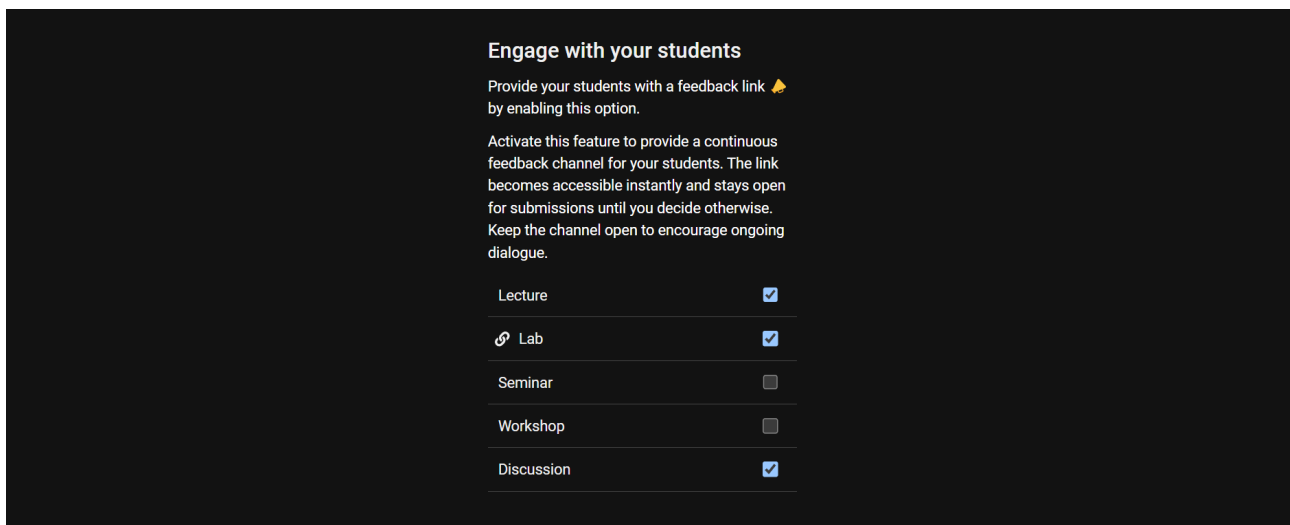


Figure 3.9: Sidebar to create the event links

When the instructor creates an account, a unique ID is assigned based on their initials, which can be modified in the settings. This is used to create a link for the students to use. For example, if my unique ID is FG, the generated links will be rgu.anonfeedback.io/FG/Lecture, rgu.anonfeedback.io/FG/Lab, and so on. Anyone with the link will be able to give feedback for that class.

Providing a feedback link is not a must. If students submit feedback directly from rgu.anonfeedback.io, it would be considered as general feedback.

The instructors have complete control over which event rooms they want to activate the feedback link for. As illustrated in Figure 3.9, they can choose to enable it for lectures, labs, discussions, or any other relevant event type. Once activated, the corresponding link becomes instantly accessible to students, and it remains open for submissions until the instructor decides to close it.

Keeping these feedback channels open is crucial as it encourages consistent engagement and an environment of open communication between instructors and students. This way, instructors can gather real-time feedback, address concerns, and make necessary adjustments to enhance the learning experience.

The ability to customise and control these feedback channels is a key aspect of my software. It allows instructors to tailor the feedback process according to the specific needs of each event room or course component. This approach aligns with the goals of my honors project, facilitating effective communication and collaboration within the academic setting.

Chapter 4

Testing & Evaluation

To ensure the robustness and security of the application, rigorous quality assurance measures were implemented throughout the development process. Continuous testing was conducted to verify the functionality of individual components and identify potential bugs or edge cases. The use of TypeScript for both the frontend and backend helped catch type-related errors during development. Cross-Origin Resource Sharing (CORS) configurations were also tested to prevent unauthorised cross-origin requests and mitigate security vulnerabilities.

ESLint paired with the Prettier extension and Airbnb guidelines were also employed to maintain consistent code formatting and style, and make sure that best practices were followed, and the risk of potential flaws in the codebase was minimised.

The deployment process was thoroughly evaluated using Docker and GitHub Actions. Docker containers were used to package the application and its dependencies, ensuring consistent and reproducible deployments across different environments. Thanks to GitHub Actions, I also implemented continuous integration and continuous deployment (CI/CD) to Google Cloud, automating the build, testing, and deployment pipeline. This approach made the deployments automatic and reliable, minimising manual intervention. The testing and deployment strategies adopted in this project have contributed to the overall quality, security, and maintainability of the codebase.

Due to time constraints, I was not able to run user testing for this application in an actual university class setting, and I am hoping to do so in the near future. However, the expected results anticipate increased student engagement, as the system aims to limit the amount of effort required for students to submit feedback. The anonymous nature is expected to promote more honest and critical feedback from students. The GPT-4's sentiment analysis capabilities and the provided insight tools should provide instructors with a more nuanced understanding of the students' perspectives on the course material and delivery.

Chapter 5

Handover & Future

I am happy with the outcome of this work, but it is worth noticing that due to time constraints and other deliverables, this is not yet a complete product, and can be considerably improved.

Some additions I would like to work on in the future are:

- Additional filters. Currently the only available filters are enough to prove the usefulness of the system, but adding the ability to filter by lecturer, tags, and selected emoji, would make it more useful and complete.
- More visualisations. I would like to see how many total feedback submissions have been sent over time, this should be quite easy to implement. The idea is to create some sort of heatmap like the one on GitHub commits (green squares).
- Creating custom room with dedicated tags. Currently all of the rooms have the same pre-defined tags, but ideally this software should give to the instructor the ability to create a specific class, and define their own custom tabs (e.g. for a lab you would rather use "These exercises were difficult" than "The lecture was difficult), to enhance the benefits of reducing cognitive load as previously discussed.

Overall, the application has a solid foundation for it to be tested on a wide audience, and gather more data to further improve it. The chosen stack and set of technologies were also chosen because of their popularity, making handover for this project easier.

Glossary

- **Few-Shot Learning:** “Providing general instructions that apply to all examples is generally more efficient than demonstrating all permutations of a task by example, but in some cases providing examples may be easier. For example, if you intend for the model to copy a particular style of responding to user queries which is difficult to describe explicitly. This is known as “few-shot” prompting.” (OpenAI [2024](#)).
- **Generative Pre-trained Transformers (GPT):** “Generative Pre-trained Transformers, commonly known as GPT, are a family of neural network models that uses the transformer architecture and is a key advancement in artificial intelligence (AI) powering generative AI applications such as ChatGPT. GPT models give applications the ability to create human-like text and content (images, music, and more), and answer questions in a conversational manner.” (AWS [2023b](#))
- **Large Language Model (LLM):** “Large language models (LLM) are very large deep learning models that are pre-trained on vast amounts of data. The underlying transformer is a set of neural networks that consist of an encoder and a decoder with self-attention capabilities. The encoder and decoder extract meanings from a sequence of text and understand the relationships between words and phrases in it.” (AWS [2023a](#))
- **Natural Language Processing (NLP):** “Natural language processing (NLP) refers to the branch of computer science—and more specifically, the branch of artificial intelligence or AI—concerned with giving computers the ability to understand text and spoken words in much the same way human beings can.” (IBM [2023](#))
- **Sentiment Analysis:** “Sentiment analysis or opinion mining is the computational study of people’s opinions, sentiments, evaluations, attitudes, moods, and emotions.” (Cambria et al. [2017](#)).

References

- AWS (2023a). *What are Large Language Models? - LLM AI Explained - AWS*. URL: <https://aws.amazon.com/what-is/large-language-model/> (visited on 10/26/2023).
- (2023b). *What is GPT AI? - Generative Pre-Trained Transformers Explained - AWS*. URL: <https://aws.amazon.com/what-is/gpt/> (visited on 10/26/2023).
- Bergstrom, Tony, Andrew Harris, and Karrie Karahalios (2011). “Encouraging Initiative in the Classroom with Anonymous Feedback”. en. In: *Human-Computer Interaction – INTERACT 2011*. Ed. by Pedro Campos et al. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 627–642. ISBN: 978-3-642-23774-4. DOI: [10.1007/978-3-642-23774-4_49](https://doi.org/10.1007/978-3-642-23774-4_49).
- Cambria, Erik et al., eds. (2017). *A Practical Guide to Sentiment Analysis*. en. Vol. 5. Socio-Affective Computing. Cham: Springer International Publishing. ISBN: 978-3-319-55392-4 978-3-319-55394-8. DOI: [10.1007/978-3-319-55394-8](https://doi.org/10.1007/978-3-319-55394-8). URL: <http://link.springer.com/10.1007/978-3-319-55394-8> (visited on 10/26/2023).
- Clear, Tony (2012). “Systematic literature reviews and undergraduate research”. In: *ACM Inroads* 3.4, pp. 10–11. ISSN: 2153-2184. DOI: [10.1145/2381083.2381087](https://doi.org/10.1145/2381083.2381087). URL: <https://dl.acm.org/doi/10.1145/2381083.2381087>.
- DePaolo, Concetta A. and Kelly Wilkinson (May 2014). “Get Your Head into the Clouds: Using Word Clouds for Analyzing Qualitative Assessment Data”. en. In: *TechTrends* 58.3, pp. 38–44. ISSN: 1559-7075. DOI: [10.1007/s11528-014-0750-9](https://doi.org/10.1007/s11528-014-0750-9). URL: <https://doi.org/10.1007/s11528-014-0750-9> (visited on 04/25/2024).
- Documentation, IBM (Apr. 2024). *Tips for writing foundation model prompts: prompt engineering*. en-US. URL: <https://www.ibm.com/docs/en/watsonx/saas?topic=lab-prompt-tips> (visited on 04/25/2024).
- Hollender, Nina et al. (Nov. 2010). “Integrating cognitive load theory and concepts of human–computer interaction”. In: *Computers in Human Behavior*. Online Interactivity: Role of Technology in Behavior Change 26.6, pp. 1278–1288. ISSN: 0747-5632. DOI: [10.1016/j.chb.2010.05.031](https://doi.org/10.1016/j.chb.2010.05.031). URL: <https://www.sciencedirect.com/science/article/pii/S0747563210001718> (visited on 04/23/2024).
- IBM (2023). *What is Natural Language Processing? — IBM*. URL: <https://www.ibm.com/topics/natural-language-processing> (visited on 10/25/2023).
- Kheiri, Kiana and Hamid Karimi (2023). *SentimentGPT: Exploiting GPT for Advanced Sentiment Analysis and its Departure from Current Machine Learning*. arXiv:2307.10234 [cs]. URL: <http://arxiv.org/abs/2307.10234>.
- Learn, Microsoft (Feb. 2024). *Prompt engineering techniques with Azure OpenAI - Azure OpenAI Service*. en-us. URL: <https://learn.microsoft.com/en-us/azure/ai-services/openai/concepts/advanced-prompt-engineering> (visited on 04/23/2024).

- Li, Cheng et al. (Nov. 2023). *Large Language Models Understand and Can be Enhanced by Emotional Stimuli*. en. arXiv:2307.11760 [cs]. URL: <http://arxiv.org/abs/2307.11760> (visited on 04/25/2024).
- Loh, Erwin (May 2023). “ChatGPT and generative AI chatbots: challenges and opportunities for science, medicine and medical leaders”. en. In: *BMJ Leader*. Publisher: BMJ Specialist Journals Section: Commentary, leader. ISSN: 2398-631X. DOI: [10.1136/leader-2023-000797](https://doi.org/10.1136/leader-2023-000797). URL: <https://bmjleader.bmj.com/content/early/2023/05/02/leader-2023-000797>.
- Lu, Ruiling and Linda Bol (2007). “A Comparison of Anonymous Versus Identifiable E-Peer Review On College Student Writing Performance and the Extent of Critical Feedback”. en. In: *The International Technology Management Review* 10.1. Publisher: Atlantis Press, pp. 1–11. ISSN: 1835-5269. DOI: [10.2991/itm.r.k.210105.001](https://doi.org/10.2991/itm.r.k.210105.001). URL: <https://www.atlantispress.com/journals/itm.r/125951268> (visited on 10/26/2023).
- Marcus, Bernd et al. (2007). “Compensating for Low Topic Interest and Long Surveys: A Field Experiment on Nonresponse in Web Surveys”. In: *Social Science Computer Review* 25.3, pp. 372–383. DOI: <https://doi.org/10.1177/0894439307297606>.
- Moderation API (Apr. 2024). en. URL: <https://platform.openai.com/docs/guides/moderation/moderation> (visited on 04/25/2024).
- New embedding models and API updates (Jan. 2024). en-US. URL: <https://openai.com/blog/new-embedding-models-and-api-updates> (visited on 04/25/2024).
- OpenAI (2023). *Using GPT-4 for content moderation*. URL: <https://openai.com/blog/using-gpt-4-for-content-moderation> (visited on 10/25/2023).
- (2024). *Prompt Engineering Tactic: Provide examples*. en. URL: <https://platform.openai.com/docs/guides/prompt-engineering/tactic-provide-examples> (visited on 04/25/2024).
- Platform, OpenAI (Feb. 2023). *Prompt Engineering Documentation*. en. URL: <https://platform.openai.com/docs/guides/prompt-engineering> (visited on 04/23/2024).
- sebleier (Aug. 2010). *NLTK’s list of english stopwords*. en. URL: <https://gist.github.com/sebleier/554280> (visited on 04/25/2024).
- visx — @visx/wordcloud documentation (2024). en. URL: <https://airbnb.io/visx/> (visited on 04/25/2024).
- Wang, Rose E. et al. (June 2023). *SIGHT: A Large Annotated Dataset on Student Insights Gathered from Higher Education Transcripts*. arXiv:2306.09343 [cs]. DOI: [10.48550/arXiv.2306.09343](https://doi.org/10.48550/arXiv.2306.09343). URL: <http://arxiv.org/abs/2306.09343> (visited on 04/25/2024).
- Whitenton, Kathryn (2024). *Minimize Cognitive Load to Maximize Usability*. en. URL: <https://www.nngroup.com/articles/minimize-cognitive-load/> (visited on 04/25/2024).
- Yang, Min, Pauline Mak, and Rui Yuan (2021). “Feedback Experience of Online Learning During the COVID-19 Pandemic: Voices from Pre-service English Language Teachers”. In: *The Asia-Pacific Education Researcher* 30.6, pp. 611–620. ISSN: 0119-5646. DOI: [10.1007/s40299-021-00618-1](https://doi.org/10.1007/s40299-021-00618-1). URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8547125/>.

Appendix A

Project Logs

A.1 Github Commits

```
1 1b7e09f - Francesco Gruosso, 3 days ago : Merge pull request #9 from
    ↪ FrancescoCoding/dev
2 d0c2576 - Francesco Gruosso, 3 days ago : Display appropriate moderation message on
    ↪ 403 (hateful or harmful feedback)
3 d75b86d - Francesco Gruosso, 5 days ago : Merge pull request #8 from FrancescoCoding/
    ↪ dev
4 80e0a2a - Francesco Gruosso, 5 days ago : Refactored Feedback component to handle
    ↪ submission state and display appropriate messages
5 986f789 - Francesco Gruosso, 4 weeks ago : Merge pull request #7 from FrancescoCoding
    ↪ /dev
6 1a78df4 - Francesco Gruosso, 4 weeks ago : Added Netlify redirects
7 1a2b756 - Francesco Gruosso, 4 weeks ago : Merge pull request #6 from FrancescoCoding
    ↪ /dev
8 b9e503c - Francesco Gruosso, 4 weeks ago : Handled case where student submits with a
    ↪ non-uppercase instructorId
9 986140a - Francesco Gruosso, 5 weeks ago : Merge pull request #5 from FrancescoCoding
    ↪ /dev
10 312ed13 - Francesco Gruosso, 5 weeks ago : Added event and instructor to insights
    ↪ from the student feedback
11 a586f85 - Francesco Gruosso, 5 weeks ago : Merge branch 'main' into dev
12 f7fb1aa - Francesco Gruosso, 5 weeks ago : Update GitHub Actions to trigger only on
    ↪ changes to backend directory
13 95067ce - Francesco Gruosso, 5 weeks ago : Merge pull request #4 from FrancescoCoding
    ↪ /dev
14 e92f651 - Francesco Gruosso, 5 weeks ago : Merge branch 'main' into dev
15 c46d6fe - Francesco Gruosso, 5 weeks ago : UX & Prompt improvements
16 522151b - Francesco Gruosso, 5 weeks ago : Merge pull request #3 from FrancescoCoding
    ↪ /dev
17 d986d8d - Francesco Gruosso, 5 weeks ago : Added Modal component, magic AI reasoning,
    ↪ and secret ASCII art
18 58c77f8 - Francesco Gruosso, 5 weeks ago : Remove Azure Web Apps workflow and update
    ↪ API base URL
19 8c71b75 - Francesco Gruosso, 5 weeks ago : Update CORS options for production and
    ↪ development
20 e562fc3 - Francesco Gruosso, 5 weeks ago : Add Docker Buildx cache setup action
21 d68b7bd - Francesco Gruosso, 5 weeks ago : Build docker container and push to Google
    ↪ Cloud
```

22 6e32cf9 - Francesco Gruosso, 5 weeks ago : [draft] Sending selected tags and selected
 emoji to the db

23 179edcf - Francesco Gruosso, 6 weeks ago : Code refactoring frontend

24 2d26e5b - Francesco Gruosso, 6 weeks ago : UI enhancements

25 415c295 - Francesco Gruosso, 6 weeks ago : Moved room name to breadcrumb

26 db55c4e - Francesco Gruosso, 6 weeks ago : Added CustomSkeleton component

27 f75d75b - Francesco Gruosso, 6 weeks ago : Added dockerfile & Minor UI changes

28 78d7934 - Francesco Gruosso, 6 weeks ago : Changed breadcrumb production vs local
 functionality

29 01a1908 - Francesco Gruosso, 6 weeks ago : Added Navbar, Favicons & Enhanced some UI

30 8f55541 - Francesco Gruosso, 6 weeks ago : Update build script in package.json

31 f3f8376 - Francesco Gruosso, 6 weeks ago : Update build script in package.json

32 8678018 - Francesco Gruosso, 6 weeks ago : Update TypeScript version to 5.4.2

33 c4ae250 - Francesco Gruosso, 6 weeks ago : [API Deployment Test]

34 99dd0ac - Francesco Gruosso, 6 weeks ago : Merge pull request #2 from FrancescoCoding
 /Generate-event-links

35 4620bc7 - Francesco Gruosso, 6 weeks ago : Added dynamic event links (rooms)
 generation, organization schema, some lazy loading

36 9bf2d04 - Francesco Gruosso, 7 weeks ago : [draft] Generate links

37 2e711d0 - Francesco Gruosso, 7 weeks ago : Small UI and microcopy changes

38 a11f31e - Francesco Gruosso, 7 weeks ago : Login page redirect on 401 from the
 backend & small UI changes in Settings

39 a9b08e7 - Francesco Gruosso, 7 weeks ago : Small refactoring UI styles and document
 titles

40 cb134e1 - Francesco Gruosso, 7 weeks ago : Add support for last three months date
 range & UI enhancements

41 303e4bf - Francesco Gruosso, 7 weeks ago : Added toasts & Event shortcuts in
 settings

42 5b4b83b - Francesco Gruosso, 7 weeks ago : Added user services, settings draft

43 01ed6bc - Francesco Gruosso, 7 weeks ago : Validation for user initials on register
 to avoid duplicate key problems

44 118bc85 - Francesco Gruosso, 7 weeks ago : Backend routes and controllers refactor

45 40c1fd5 - Francesco Gruosso, 7 weeks ago : Added user system database, routes, views,
 middleware, context

46 e3b0be5 - Francesco Gruosso, 8 weeks ago : Merge pull request #1 from FrancescoCoding
 /local-filters-to-context

47 91e1a54 - Francesco Gruosso, 8 weeks ago : Refactor CSS styles for flagged feedback
 items

48 5a17a42 - Francesco Gruosso, 8 weeks ago : Fully refactored and outsourced filters to
 useFilters Hook

49 c31191c - Francesco Gruosso, 8 weeks ago : Using filter context for active date
 ranges, from here filters will be much more flexible

50 814ecce - Francesco Gruosso, 8 weeks ago : Filters to have pre-set ranges & created
 context to better handle them

51 37ad1a6 - Francesco Gruosso, 8 weeks ago : Connected WordCloud to filters

52 75c7128 - Francesco Gruosso, 8 weeks ago : Added date ranges to calendar for easier
 navigation

53 f82dad9 - Francesco Gruosso, 8 weeks ago : [draft] Add Wordcloud component

54 ded2ce7 - Francesco Gruosso, 9 weeks ago : Refactor date filtering logic in
 FlaggedFeedback and SentimentAnalysis components

55 fab9d65 - Francesco Gruosso, 9 weeks ago : Refactor sentiment filter logic in
 SentimentAnalysis.tsx

56 72089c1 - Francesco Gruosso, 9 weeks ago : Added filters resets & insourced feedback
 data fetch to sentiment analysis component

6788e01 - Francesco Gruosso, 9 weeks ago : Filter active state now more obvious

b5cc812 - Francesco Gruosso, 9 weeks ago : Added targeted filters and no flagged
 ↳ feedback UI

2771d66 - Francesco Gruosso, 9 weeks ago : Small refactor and basic calendar styling

343a55a - Francesco Gruosso, 9 weeks ago : Filters improvements and added
 ↳ calendar to filter by time

7eb5874 - Francesco Gruosso, 9 weeks ago : Added full moderation pipeline

689b893 - Francesco Gruosso, 9 weeks ago : Outsourced SentimentAnalysis component to
 ↳ its own file

6359e0f - Francesco Gruosso, 9 weeks ago : Saving flagged feedback to DB & analysis
 ↳ reasoning

0974833 - Francesco Gruosso, 9 weeks ago : Added feedback insights dashboard for
 ↳ analysis

397a7fa - Francesco Gruosso, 9 weeks ago : Remove unnecessary import statement

30b28d0 - Francesco Gruosso, 9 weeks ago : Add main navigation wrapper to have the
 ↳ options available globally

eab68e9 - Francesco Gruosso, 9 weeks ago : Routes pipeline implementation [public/
 ↳ private] & architectural nuances changes

8b50895 - Francesco Gruosso, 9 weeks ago : MongoDB feedback and analysis now is
 ↳ properly saving to the database

58050b5 - Francesco Gruosso, 9 weeks ago : Refactor for better readability and UX

e92a857 - Francesco Gruosso, 9 weeks ago : Refactor sentiment analysis prompt for
 ↳ more reliable JSON output

1d65cf5 - Francesco Gruosso, 9 weeks ago : UI/UX small improvements [Options
 ↳ component & media queries]

8bfbde2 - Francesco Gruosso, 10 weeks ago : HCI UI/UX improvements, [draft] MongoDB
 ↳ Schema, emotes render strategy rework

9da9b98 - Francesco Gruosso, 10 weeks ago : Added MongoDB pipeline

7958293 - Francesco Gruosso, 2 months ago : Return 403 status code on moderated
 ↳ feedback

e51eda0 - Francesco Gruosso, 2 months ago : Tags LS removal logic, fixed theme looks

fa510d1 - Francesco Gruosso, 2 months ago : Frontend rework, added theme, re-
 ↳ architected components rendering strategy

32f1787 - Francesco Gruosso, 2 months ago : Small refactor - removed ESLint issue

87d4999 - Francesco Gruosso, 2 months ago : Add memoization to EmojiSelector
 ↳ component and export it as MemoizedEmojiSelector***

379f32d - Francesco Gruosso, 3 months ago : Remove manual emoji cache service worker
 ↳ as it's been substituted by vitepwa/workbox

0c3a321 - Francesco Gruosso, 3 months ago : Add Vite PWA plugin to cache images
 ↳ effectively with workbox

034c095 - Francesco Gruosso, 3 months ago : [test] Add service worker for emoji
 ↳ caching

d8d9326 - Francesco Gruosso, 3 months ago : Refactor feedback component and set-up
 ↳ first index.ts re-import.

59fd7af - Francesco Gruosso, 3 months ago : Structured feedback JSON from prompt
 ↳ engineering

b85315b - Francesco Gruosso, 3 months ago : Migrated moderation service to backend &
 ↳ created submitFeedback logic with draft sentiment analysis pipeline

d87b8ee - Francesco Gruosso, 3 months ago : Fix import case in moderation service

473bd07 - Francesco Gruosso, 3 months ago : Add response interceptor to handle Axios
 ↳ errors

89fca9a - Francesco Gruosso, 3 months ago : Add @types/axios package

68f205f - Francesco Gruosso, 3 months ago : Added moderation API client test

9519738 - Francesco Gruosso, 3 months ago : Add axios lib and its config/err handling

```

90 31abb03 - Francesco Gruosso, 3 months ago : Commented out dns import and function
    ↳ call in vite.config.ts
91 11c6848 - Francesco Gruosso, 3 months ago : Update page title and add TypeScript type
    ↳ definitions
92 0874d37 - Francesco Gruosso, 3 months ago : CD/CI test
93 9b957b1 - Francesco Gruosso, 3 months ago : Azure workflow config
94 60fb48a - Francesco Gruosso, 3 months ago : Merge branches 'main' and 'main' of https
    ↳ ://github.com/FrancescoCoding/Educational-Feedback
95 7eb906e - Francesco Gruosso, 3 months ago : Update GPT controller response message
96 6bc4105 - Francesco Gruosso, 3 months ago : Create azure-webapps-node.yml
97 5989186 - Francesco Gruosso, 3 months ago : Added endpoints and routes draft
98 8f96758 - Francesco Gruosso, 3 months ago : Update tsconfig path in .eslintrc.cjs
99 60023f7 - Francesco Gruosso, 3 months ago : Update nodemon.json and index.ts
100 d4cb24a - Francesco Gruosso, 3 months ago : Add .gitignore and remove dist/index.js
101 9761841 - Francesco Gruosso, 3 months ago : Added backend base
102 bbbcb34 - Francesco Gruosso, 3 months ago : Remove unused folder
103 7f4f32b - Francesco Gruosso, 3 months ago : Add EventDetails component [draft] and
    ↳ import it in Feedback component
104 2212981 - Francesco Gruosso, 3 months ago : Refactor feedback component imports and
    ↳ types
105 0afadf4 - Francesco Gruosso, 3 months ago : Implemented additional tags functionality
    ↳ , conditional rendering, local storage tags handling
106 55d40dd - Francesco Gruosso, 3 months ago : AI Sentiment Analysis first commit

```

Appendix B

Source Code

The Source Code is currently in a private repository, I might make it available at some point in the future, but for the time being I would rather keep it this way.