



# Sentiment Analysis Visualization Application

Liam Bedford

N00202070

Supervisor: Cyril Connolly

Second Reader: John Montayne

Year 4 2023/24

DL836 BSc (Hons) in Creative Computing

## Abstract

The goal of this project was to develop an intuitive, user-friendly application that allows users to perform sentiment analysis on different topics, issues, or brands using the reddit API. The application then would return analysis results in the form of visualisations. These visualizations help users gain insight into trends and patterns in the sentiments and opinions surrounding a search term. The development of the application involved building a flask backend and ReactJS frontend.

## Acknowledgements

I would like to express my sincere gratitude to my supervisors, Cyril Connolly and John Montayne, for their support and guidance throughout the project. Their feedback and encouragement were essential to the completion of this project.

I would also like to thank Mohammed Cherbatji for his input and advice which was invaluable in the implementation stages of this project.

I would like to thank my classmates who helped create a pleasant and enjoyable environment to work on the project in.

I would like to thank users who provided feedback in the testing phase of the project.

Finally, I would like to thank my friends and family for their support all throughout the length of the project, as well as throughout the 4-year course.

**The incorporation of material without formal and proper acknowledgement (even with no deliberate intent to cheat) can constitute plagiarism.**

If you have received significant help with a solution from one or more colleagues, you should document this in your submitted work and if you have any doubt as to what level of discussion/collaboration is acceptable, you should consult your lecturer or the Course Director.

**WARNING:** Take care when discarding program listings lest they be copied by someone else, which may well bring you under suspicion. Do not to leave copies of your own files on a hard disk where they can be accessed by other. Be aware that removable media, used to transfer work, may also be removed and/or copied by others if left unattended.

Plagiarism is considered to be an act of fraudulence and an offence against Institute discipline.

Alleged plagiarism will be investigated and dealt with appropriately by the Institute. Please refer to the Institute Handbook for further details of penalties.

**The following is an extract from the B.Sc. in Creative Computing (Hons) course handbook. Please read carefully and sign the declaration below**

*Collusion may be defined as more than one person working on an individual assessment. This would include jointly developed solutions as well as one individual giving a solution to another who then makes some changes and hands it up as their own work.*

**DECLARATION:**

I am aware of the Institute's policy on plagiarism and certify that this thesis is my own work.

Student: Liam Bedford

Signed

A handwritten signature in black ink that reads "Liam Bedford". The signature is written in a cursive, flowing style with a large initial 'L'.

Failure to complete and submit this form may lead to an investigation into your work.

## Contents

1. Introduction.....	8
2. Research .....	9
2.1 Introduction.....	9
2.2 How Sentiment Analysis works .....	9
2.3 Rule-based Sentiment Analysis.....	10
2.4 Machine learning Sentiment Analysis.....	10
2.5 Hybrid Approach.....	11
2.6 Binary classification .....	11
2.6 Applications of Sentiment Analysis.....	12
2.7 Sentiment Analysis Applied to Social Media .....	13
2.8 Framework for Sentiment Analysis.....	16
2.9 Comparing Sentiment Analysis tools.....	17
2.10 Conclusion.....	18
3. Requirements .....	19
3.1 Introduction.....	19
3.2 Requirements gathering .....	19
3.2.1 Similar applications.....	19
3.2.2 Personas .....	23
3.3 Requirements modelling.....	25
Functional requirements.....	25
Non-functional requirements .....	25
3.3.1 Use Case Diagrams .....	25
3.3.2 Feasibility.....	26
3.4 Conclusion.....	26
4. Design .....	27
4.1 Introduction.....	27
4.2 Program Design.....	27
4.2.1 Technologies .....	27
4.2.2 Structure of React Folder .....	29
4.2.3 Design Patterns .....	30
4.2.4 Application architecture .....	31

4.3 User Interface Design.....	32
4.3.1 Wireframe .....	32
4.3.2 Style guide.....	34
4.4 Conclusion.....	36
5. Implementation.....	37
5.1 Introduction.....	37
5.2 Scrum Methodology.....	37
5.3 Development environment.....	38
5.4 Sprint 1 .....	40
5.4.1 Goal.....	40
5.4.2 Item 1 - Research.....	40
5.4.3 Item 2 – Prototype .....	41
5.5 Sprint 2 .....	42
5.5.1 Goal.....	42
5.5.2 Item 1 – Requirements Document V1 .....	42
5.5.3 Item 2 – Design Document V1 .....	42
5.5.4 Item 2 – Application Prototype V1.....	43
5.6 Sprint 3.....	44
5.6.1 Goal.....	44
5.6.2 Item 1 – Implementation Document V1.....	44
5.6.3 Item 2 – Application Prototype V1.....	44
5.6.4 Item 3 – Update Design and Requirements Docs.....	45
5.7 Sprint 4.....	45
5.7.1 Goal.....	45
5.7.2 Item 1 – Application Prototype V3.....	45
5.7.3 Item 2 – Implementation Document V2.....	46
5.7.4 Item 3 – Design Document V3 .....	48
5.8 Sprint 5.....	48
5.8.1 Goal.....	48
5.8.2 Item 1 – Application Prototype V4.....	48
5.8.3 Item 2 – Implementation Document V3.....	51
5.9 Sprint 6 .....	52
5.9.1 Goal.....	52
5.9.2 Item 1 – Application Prototype V5.....	52
5.9.3 Item 2 – Testing Document .....	54

5.10 Sprint 7 .....	55
5.10.1 Goal.....	55
5.10.2 Item 1 – Application Prototype V5 Small additions .....	55
5.10.3 Item 2 – Thesis V1 .....	56
5.11 Sprint 8.....	57
5.11.1 Goal.....	57
5.11.2 Item 1 – Thesis V2 .....	57
5.12 Sprint 9.....	57
5.12.1 Goal.....	57
5.12.2 Tasks.....	57
5.13 Conclusion.....	57
6. Testing.....	58
6.1 Introduction.....	58
6.2 Functional testing .....	58
6.2.1 Input Testing.....	58
6.2.2 Analytics/Data Testing.....	59
6.2.3 Application Performance Testing .....	60
6.2.4 Interface/Layout Testing.....	61
6.2.5 Error Handling Testing .....	63
6.3 User Testing .....	66
6.3.1 User Interface.....	66
6.3.2 User Experience.....	67
6.3.3 Functionality .....	67
6.4 Conclusion.....	68
7. Project Management .....	69
7.1 Introduction.....	69
7.2 Proposal .....	69
7.3 SCRUM Methodology.....	69
7.4 Project Management Tools.....	70
7.4.1 Notebook.....	70
7.4.2 GitHub .....	70
7.5 Reflection.....	71
7.5.1 Your views on the project.....	71
7.5.2 Working with a supervisor .....	71
7.5.3 Technical skills .....	71

7.6 Conclusion .....	71
8. Conclusion.....	72
9. References .....	74

# 1. Introduction

The overall aim of this project was to develop an application that allows users to easily perform their own sentiment analysis and infer their own conclusions on different topics based on the visualizations and graphs provided by the application. The project was created using ReactJS and Flask as front-end and back-end respectively.

This report document collects the various key areas of the applications development and details them. The different sections of this report give an account of how the project came together. The research chapter provided the base for the project. The design section informs how the application was implemented in both program and user interface design. The requirements revealed key functionality and elements that the application needed to deliver on. The implementation was essential to keeping track of development and elaborating on different elements that came together to make up the application. The testing section was invaluable in determining if the application was successful in functionality and user experience.

The rationale for developing this application was the conclusion that there is a lack of sentiment analysis applications that make use of the reddit platform in addition to the fact that many online sentiment analysis tools are not user friendly and allow much flexibility to the user when choosing a topic to analyse.

The purpose of the application is to bring a lightweight, easy to use sentiment analysis application to a wide range of users. Potential users include those working in areas such as advertising, marketing, politics, and data analysis

The development of the system involved designing a robust and easy to interact with user interface, programming a flask back-end to perform the sentiment analysis and developing a react front-end to allow users to perform the analysis and view displayed visualizations and analytics.



## 2. Research

### 2.1 Introduction

Sentiment analysis is a kind of text classification that classifies texts based on the *sentimental orientation* (SO) of opinions they contain. (Leung and Kane, 2009). Opinions can be expressed on any tangible thing, for example, a person, place, or product. Engagement in Sentiment analysis has increased alongside the emergence of online platforms where people can convey opinions, such as twitter or reddit. Sentiment Analysis can be applied to gain various valuable insights in different areas such as marketing and politics, making it a powerful tool in current times where social media and the spread of information is ever present.

The aim of this research investigates how to perform sentiment analysis to solve a problem or reveal valuable information using social media. This chapter outlines what Sentiment Analysis is, the diverse types of approaches used within Sentiment Analysis, its contemporary use cases with examples, applications on social media and details the technology used in Sentiment Analysis. The research conducted will provide a foundation and understanding of sentiment analysis to build upon.

### 2.2 How Sentiment Analysis works

Sentiment is defined as either positive or negative when expressed in text, sentiment analysis is used to determine sentiment found in a text (Taboada, 2016). When analysing an opinion, the target of an opinion is known as the object, which is made up of several different parts called components and each object has its own attributes (Indurkha, 1999). Sentiment Analysis uses natural language processing to train a programme to understand and analyse text. There are three main approaches to working with sentiment analysis. These include rule-based, machine learning and hybrid (AWS, 2023).

## 2.3 Rule-based Sentiment Analysis

In rule-based sentiment analysis, a set of rules are used to determine the subjectivity, polarity, or subject of an opinion (MonkeyLearn, 2023). These rules or techniques originate from sentiment analysis in computational linguistics. Examples include stemming, lemmatization and tokenization. The use of lexicons is essential in determining sentiment using this approach.

Stemming is a natural language processing technique where the suffixes, or, the letters added to the end of a base word, are removed to obtain a word's base form. Lemmatization is a more complex natural language processing technique that makes use of vocabulary and the process of morphological analysis to simplify a word to its base form. Morphological analysis is a technique derived from linguistics where various parts of words such as prefixes, suffixes, and roots are analysed to determine the base form of a certain word. There are pros and cons that come with each of these natural language processing techniques. Stemming is the quicker of the two techniques, although stemming can generate an invalid base form, which creates ambiguity. Lemmatization is a slower natural language processing technique because it is more complex. Lemmatization produces a base form that can be found in a dictionary, making it valid and therefore, a more accurate technique in contrast to stemming (Aydin, 2023).

A lexicon is a key component of sentiment analysis. A lexicon is a list of words that represent sentiment. Positive and Negative lexicons are created and then scanned to determine a sentence's sentiment, revealing if it is positive, negative, or neutral (AWS, 2023). Tokenization breaks down text into words, an example given by (Srinivas, 2020) is that the text "It is raining" can be broken down and tokenized into 'It', 'is', 'raining', to interpret the meaning of the text within natural language processing.

## 2.4 Machine learning Sentiment Analysis

The machine learning approach to sentiment analysis uses techniques such as neural networks and deep learning to make a computer identify sentiment from given text. A sentiment analysis model must be created on trained on data so that it can determine sentiment with accuracy (AWS, 2023).

A model must be trained on clean datasets to be effective; a large amount of text can be processed accurately using machine learning. A successfully trained machine learning model is specific to only one area. A model used to monitor product reviews cannot be used to monitor online discourse without first being retrained on relevant clean data.

## 2.5 Hybrid Approach

The hybrid approach makes use of both approaches previously outlined, rule-based and machine learning approaches. Using these two approaches together can be effective when evaluating sentiment in text. Using the two approaches together requires a great deal of time and effort.

## 2.6 Binary classification

When applying sentiment analysis, sentiment is classified using either binary classification or multiclass classification. As revealed by (Kumar, 2023), binary classification is classifying data into two different groups. In the case of sentiment analysis, this would be positive or negative. Ternary classification is also a popular option within sentiment analysis. Ternary classification, as shown by (Ohtsuki, 2016), involves classifying sentiment into positive, negative, and neutral. It is revealed that binary classification generally gives a higher classification accuracy due to the reduction of computation required.

## 2.6 Applications of Sentiment Analysis

Sentiment Analysis is a powerful tool and is applied in many different areas. As (Sainger, 2021) reveals, it can help in the collection of real time data on opinion and sentiment in a particular area. Examples of areas sentiment analysis is useful in include social media monitoring, politics, marketing research, brand monitoring and product analysis (MonkeyLearn, 2023). In the sales area, it was found that online reviews have a large impact on the performance of a product in terms of sales, affecting 93% of customers (Jeremy, 2015), this indicates that Sentiment Analysis is a valuable tool for marketing teams to use, as they can determine how well a product or service is performing (Sainger, 2021).

As (Sainger, 2021) highlights, a contemporary example of an application of Sentiment Analysis in assessment of public opinion can be seen in the politics sector. In 2012, the Obama administration used Sentiment Analysis to assess the public opinion of announced policies and messages pushed by the Obama campaign. According to (Schectman, 2012) of the Wall Street Journal, the Obama Administration spent approximately \$155,000 dollars on technology from salesforce.com which allowed the administration to monitor trending issues by state, city, and town. This meant that the campaign could determine sentiment on certain topics and issues. This was an incredibly powerful application of sentiment analysis that the administration capitalized on by making data-driven decisions in real time. This was a key strategy which led to a successful re-election campaign for Barack Obama in 2012.

## 2.7 Sentiment Analysis Applied to Social Media

Sentiment Analysis has become recognized as an effective method of deriving and extracting opinions that can be found on social media and the web. Engagement in the studying of sentiment has risen alongside the always developing area of social media (Taboada, 2016). The social media site and app Twitter/X is an excellent use case suitable for sentiment analysis (Mumtaz, 2016). Twitter is a micro-blogging site where users' express opinions or thoughts on whatever they like, for example, politics, movies, or news. In a paper published by (Hasan, 2018), Sentiment analysis was performed to reveal the views of leaders from two large democratic parties located in India. Tweets were collected from public twitter accounts relating to the two parties. This revealed the popularity of the parties for the 2013 elections. Twitter was the best option for this study because political figures are authenticated on the site, similar sites like Instagram and Facebook do not feature this authentication functionality. The data was applied to both supervised and unsupervised machine-learning algorithms.

Tweets were extracted using the Twitter API (Application programming interface). The twitter API allows for connection to the twitter app to extract information such as accounts and tweets. Sentiword and WordNet were used to identify positive and negative scores. Both Sentiword and WordNet are open-source opinion mining tools (Mtechproject, 2023). To predict election outcome, positive tweets linked to each candidate were measured. The performance of the prediction was measured using mean absolute error, which is the average variance between the significant values in the dataset and the projected values in the same dataset (ScienceDirect, 2022). The prediction was claimed to be 0.61% better than the same type of surveys conducted traditionally. It can be inferred from this paper and numerous applications of sentiment analysis that social media apps are highly suitable environments to perform sentiment analysis and that it can reveal a great deal of valuable information, in the case of this study, sentiment towards political parties and leaders.

One tool cited to be effective in Sentiment Analysis is Textblob. An example of its use is given by (Hasan, 2018), in which the tool was used for pre-processing and polarity calculation. Pre-processing is an essential step in sentiment analysis. Pre-processing involves cleaning and preparing text for classification (ScienceDirect, 2022). In the case of the twitter website, components of a tweet that are not needed for the task such as URLs, slang-words and special characters are removed in the pre-processing. Textblob is a python library used for processing textual data and is very popular within the natural language processing and sentiment analysis space. Textblob can be used to perform various natural language processing tasks including translation, classification, noun phrase extraction and sentiment analysis (Textblob, 2020). Results from Textblob need to be validated, in the study by (Hasan, 2018), validation was done using Naïve Bayes. W-WSD, SentiWordNet and Textblob were compared, and it was found that the results of Textblob were relatively better when analysing tweets. In conclusion, it can be gathered that Textblob is an effective tool to help perform sentiment analysis. The margin between the results of WSD and Textblob were very close, meaning there is no substantial difference.

Support vector machines (SVM) were used to validate each analyser and the results are shown in the graph below.

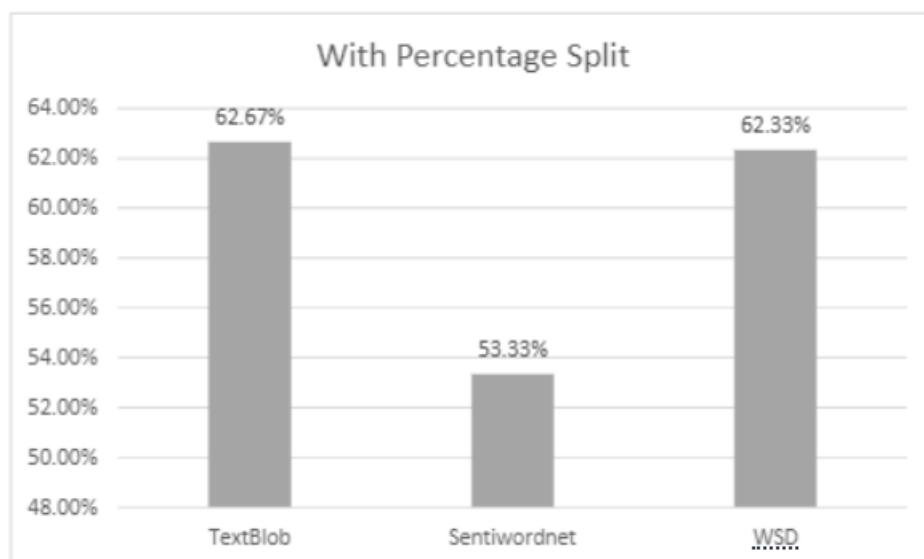


Figure 1: Results obtained from SVMs of each analyser (Hasan,2018)

A SVM is a machine learning algorithm that uses supervised learning models to perform complex classification and outlier detection by using optimal data transformations. These transformations determine the boundaries between data points using predefined labels. There are two different types of support vector machines, these are linear support vector machines and non-linear support vector machines. A linear support vector machine is a type used to classify linearly separable data. This means that a dataset can be split into classes by using a single straight line. This type of support vector machine is used most in classification and regression analysis problems (Kanade, 2022). Nonlinear or kernel support vector machines are used with non-linear data that cannot be split into distinct categories using a straight line, the opposite of the linear support vector machines. This type of support vector machines is used in optimization problems with multiple variables. Since sentiment analysis involves classification, the linear type of support vector machines is suitable to validate data

## 2.8 Framework for Sentiment Analysis

In performing Sentiment Analysis, it is helpful to provide a framework to explain the process of sentiment analysis, from data collection, the analysis to classification. An example of this framework is outlined by (Hasan, 2018), where the use of a diagram is valuable in the explanation of the process.

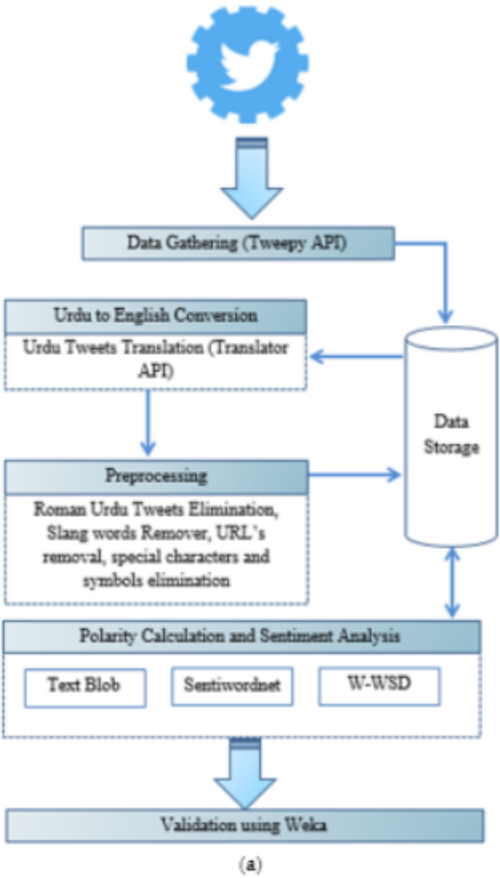


Figure 2: Framework of Sentiment Analysis (Hasan,2018)

The diagram highlights each individual step in the process and provides the technology or tool used in each step. In this case, the study was performed using Urdu, the language of Pakistan, so a translator API was needed before preprocessing. The framework is an essential piece of the study that provides an overview of the entire process of sentiment analysis, any sentiment analysis task or problem would benefit from a framework.



## 2.9 Comparing Sentiment Analysis tools

There are various tools and libraries that can be used to perform sentiment analysis on a dataset. Rule based sentiment analysis is the most common approach when working with social media evident by the studies performed by (Hasan, 2018) and (Yadav, 2014). The two most popular rule-based sentiment analysis tools are Textblob and Vader. While these two tools are both rule-based in approach, they work differently in how they analyse text.

Textblob uses two values to evaluate sentiment. Polarity, (between -1, +1) determines sentiment, where -1 is negative and +1 is positive. Subjectivity (0-1) which evaluates opinion or judgement(Shah, 2020). To calculate sentiment for a single word, averaging is applied on the values of polarity to create a polarity score. Each word is defined in a lexicon file with its polarity, subjectivity, intensity, and confidence (Fahad,2021).

```
# 1) polarity: negative vs. positive (-1.0 => +1.0)
# 2) subjectivity: objective vs. subjective (+0.0 => +1.0)
# 3) intensity: modifies next word? (x0.5 => x2.0)
```

Figure 3: Calculation of sentiment in Textblob (Fahad,2021)

Vader assigns each word in a text a positive or negative sentiment using its semantic orientation and then gives a probability that includes neutral, positive, or negative which adds up to 100%. Vader conveys both polarity and intensity of sentiment when applied to a dataset. Vader uses 5 heuristics to help analyse components of a sentence besides lexical features. Punctuation is the first heuristic; punctuation can greatly influence the intensity of an opinion and takes it into account when analysing. Capitalization is commonly used across the internet to emphasise opinions, Vader accounts for this also. Modifiers is another heuristic, an example is 'really good' or 'sort of good'. The modifiers influence intensity, positively in the first example and negatively in the second example. The fourth heuristic is a change in polarity due to words like 'however' or 'but' that drastically change a sentence's sentiment. The final heuristic is examining three lexical features together to find words that definitively express sentiment (Calderon, 2017).

While both Textblob and Vader are popular options as tools in sentiment analysis, Vader's heuristic approach to processing text makes it the more attractive choice for analysing social media as it can detect sentiment with more accuracy due to its heuristics.

## 2.10 Conclusion

It can be concluded that Sentiment Analysis is a very powerful tool, evident in the use cases and applications which helped researchers extract valuable information from social media. It's clear that Sentiment Analysis can be applied to many different areas and if performed correctly can be successful in achieving whatever goal or aim it is set to. There are many approaches to Sentiment Analysis and by carefully analysing each approach, the right approach can be selected for a particular problem or area. In particular, the application of Sentiment Analysis in social media examples are helpful in breaking down how the tool is applied successfully. Research also reveals tools that are suitable for Sentiment Analysis in particular areas, in the case of social media, Textblob and Vader are favourable due to their ability to pick up aspects of text common in social media posts.

## 3. Requirements

### 3.1 Introduction

The area of Sentiment Analysis is an ever-growing field and has many valuable applications seen in various sectors, including marketing, brand monitoring and politics as outlined in chapter 2. The goal of this project is to develop an application that applies sentiment analysis to social media to gain insight into the sentiment towards current events or topics.

The application will be a python developed programme that makes use of the python reddit API wrapper to extract posts and comments from reddit.com. Sentiment Analysis will be applied to these comments to reveal sentiment of any topic the user requires. Using visualizations and graphs, the project should visualize patterns and trends found in current topics.

### 3.2 Requirements gathering

#### 3.2.1 Similar applications

##### **Application 1:**

##### **Sentiment Analysis in Python with Textblob and Vader Sentiment Analysis**

This application developed by sentdex, is a programme which uses both Textblob and Vader sentiment analysis tools to evaluate the sentiment of movie reviews. Python is the programming language used for this exercise. Two text files containing movie reviews are used for this analysis. One text file contains predominantly positive reviews, while the other text file contains predominantly negative reviews. These are used to test both sentiment analysis tools, while showing how to apply them to text. Understanding this application was highly valuable and provided a base from which to expand upon when working with Sentiment Analysis libraries like Textblob or Vader.

```
1 # pip install textblob vadersentiment
2 from textblob import TextBlob
3 from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
4
5 analyzer = SentimentIntensityAnalyzer()
6
7
8 vs = analyzer.polarity_scores("VADER Sentiment looks interesting, I have high hopes!")
9 print(vs)
10
```

Figure 4: Setup of Vader

```
{'neg': 0.0, 'neu': 0.463, 'pos': 0.537, 'compound': 0.6996}  
[Finished in 1.0s]
```

Figure 5: Sentiment score output

This screenshot shows a basic example of how to install, import and apply Vader sentiment Analysis to a basic sentence. The text provided is largely positive and a positive sentiment analysis result should be expected in the result of the programme.

The sentiment analysis results printed provide numerous insightful values. The neg value reveals the probability that the sentiment of the text is negative. The positive value is the probability that sentiment is positive. The neu value represents the probability that the text is neutral. The compound score reveals the intensity or degree of the sentiment. The value is always between -1 and +1. -1 Being the largest negative sentiment and +1 being the highest positive sentiment. <https://akladyous.medium.com/sentiment-analysis-using-vader-c56bcffe6f24> The compound score is made up of the sum of the valence score of each word stored in the Vader lexicon. <https://towardsdatascience.com/social-media-sentiment-analysis-in-python-with-vader-no-training-required-4bc6a21e87b8>

We can infer from the printed results that the possibility of the text being positive is 0.547, there is 0 chance it is negative and a 0.463 probability it is neutral. This block of code was extremely useful in helping to understand how Vader is applied and what the results reveal.

```
with open("negative.txt", "r") as f:  
    for line in f.read().split('\n'):  
        vs = analyzer.polarity_scores(line)  
        if vs['compound'] <= 0:  
            neg_correct += 1  
            neg_count += 1  
  
print("Positive accuracy = {}% via {} samples".format(pos_correct/pos_count*100.0, pos_count))  
print("Negative accuracy = {}% via {} samples".format(neg_correct/neg_count*100.0, neg_count))
```

Figure 6: Screenshot from Vader Sentiment Example

Figure 6 highlights how text files can be imported and used for sentiment analysis using Vader. The programme provides statements that clearly indicate the sentiment of the provided movie reviews text. These programming lines reveal how the results from the analysis can be shown using print statements.

Studying this application was highly advantageous as it provides a concise yet informative example of how to perform sentiment analysis on text. The analysis in this programme is not relatively complex but provides a strong baseline for using Vader sentiment analysis.

## Application 2:

### Machine Learning-Based Sentiment Analysis for Twitter Accounts

In this study, Machine Learning (ML) based Sentiment Analysis was applied to twitter accounts to reveal the views of leaders from two main democratic political parties based in India. Data gathering was performed using tweets found on public twitter accounts associated with the two political parties. The data was applied to supervised and unsupervised machine-learning algorithms. Tweets were captured using the Twitter API to interact with the twitter app. This study was helpful material as it provided a detailed process in how Sentiment Analysis can be performed on social media.

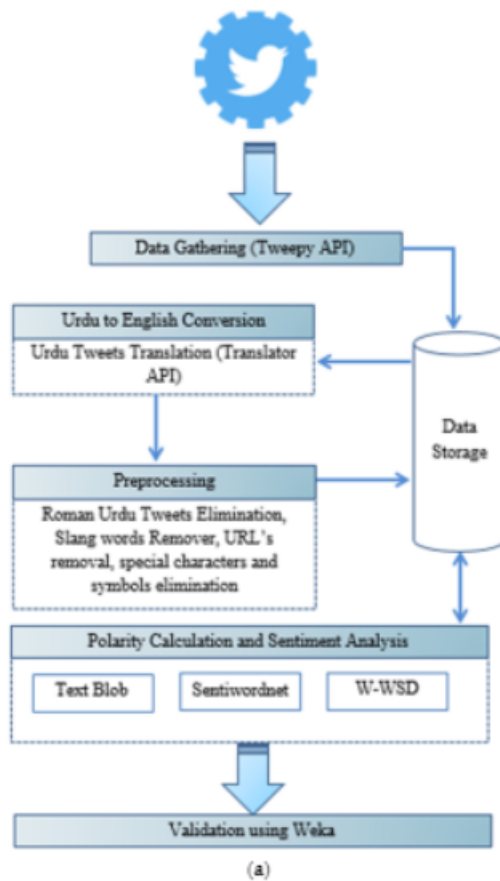


Figure 7: Diagram of the sentiment analysis process

This screenshot in figure 7 demonstrates the process of Sentiment Analysis performed in this study. This was a valuable resource as it concisely shows how the data is processed and eventually applied to Sentiment Analysis. The diagram was helpful in creating an Analysis framework for this project.

While this study clarified how Sentiment Analysis is applied to social media, the steps to performing the analysis are different to how this project performs Sentiment Analysis. For example, translation would not be required, and the amount of preprocessing is mitigated using the Vader Sentiment Analysis tool.

### Application 3:

#### PRAW Tutorial Application

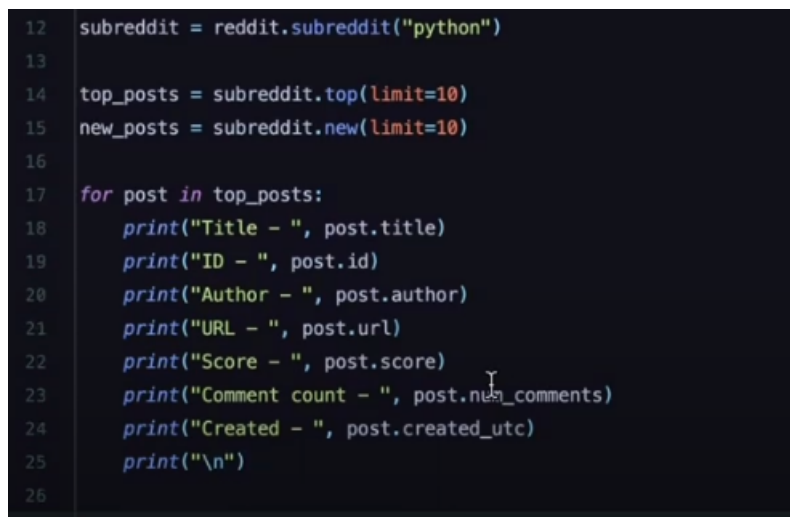
This application is a valuable introduction to using the python reddit API wrapper. Various important concepts and functionalities are demonstrated, for example, connecting to PRAW, retrieving reddit posts and printing comments.



```
main.py x
2
3
4 reddit = praw.Reddit(
5     client_id="L60FpEXBuFz5D_MRd1c5Tw",
6     client_secret="opJUQE1V7MQXPUy5g-9kMD6ctUBcGg",
7     user_agent="my-app by u/m_runthat",
8     username="m_runthat",
9     password="zifsap-tipsU3-bigxyg",
10 )
```

Figure 8: Connecting to PRAW

Figure 8 shows how a developer can use reddit API developer credentials to connect to PRAW. These credentials are obtained by applying for a reddit developer account and creating a reddit API application. After connecting to PRAW, various requests can be made to extract different information from reddit.com.



```
12 subreddit = reddit.subreddit("python")
13
14 top_posts = subreddit.top(limit=10)
15 new_posts = subreddit.new(limit=10)
16
17 for post in top_posts:
18     print("Title - ", post.title)
19     print("ID - ", post.id)
20     print("Author - ", post.author)
21     print("URL - ", post.url)
22     print("Score - ", post.score)
23     print("Comment count - ", post.num_comments)
24     print("Created - ", post.created_utc)
25     print("\n")
26
```

Figure 9: Example of retrieving data

Figure 9 outlines how to retrieve and print posts from a specific subreddit, in this case, from the 'python' subreddit. Extracting posts is the first step in retrieving meaningful data from reddit. After this is done, comments, score and more can be retrieved and complex analysis can be performed.

This application was invaluable in showing how to connect to PRAW and begin to develop a robust programme to analyse specific topics using reddit.com.

### 3.2.2 Personas



Naomi  
45 yo/female

Naomi owns and manages an independant coffee shop which has been in business for 2 years

Occupation: Manager, Owner

Location: Brighton, England

#### Needs

- Sharable analytics
- Valuable information/analytics

#### Pain points

- Bright, loud designs
- Poor sharing functionality

Figure 10: Persona 1



Rory

36 yo/male

Rory is a journalist who focuses on local news and politics. He enjoys reading and cycling in his free time.

Occupation: Established Journalist

Location: Dublin, Ireland

#### Needs

- Easily understood graphics
- Clear analytics
- Intuitive user flow

#### Pain points

- Small text
- Convoluted User interfaces
- Intuitive user flow

Figure 11: Persona 2



Sydney

21 yo/female

Sydney is a film student in her 3rd year of study. She enjoys photography and of course going to the cinema

Occupation: Film Student

Location: London, England

#### Needs

- Customization
- Easy navigation
- Aesthetically pleasing interfaces

#### Pain points

- Crowded interfaces
- Slow applications

Figure 12: Persona 3



### 3.3 Requirements modelling

#### Functional requirements

1. Using Reddit comments from Political posts on reddit, provide Sentiment Analysis results for any given topic
2. Programme the application so that it can extract sufficient amounts of data, minimum 100 reddit posts
3. Develop a front-end interface to connect to the back-end sentiment analysis programme
4. Use a library to create meaningful graphs and charts for the frontend using the data gained from the backend sentiment analysis.

#### Non-functional requirements

1. Clear and interesting visualizations
2. Analysis programme should run relatively quickly
3. Programme should contain comments to explain code and how the analysis is working

#### 3.3.1 Use Case Diagrams

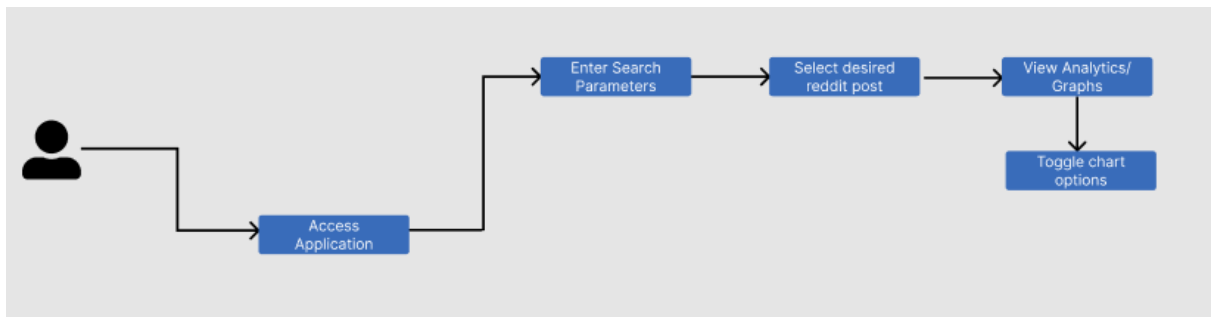


Figure 13: User use case diagram

Application Figure 13 demonstrates the use case a user would go through when using the application. After accessing the application, a user would enter search parameters to find posts to perform sentiment analysis on. Once search results are rendered, a user would select the post they desire. Visualizations will be rendered on the dashboard and users can then toggle chart options to customize the dashboard.

### 3.3.2 Feasibility

This application is developed using the python programming language. Python was chosen as it is the most popular programming language for performing data analysis and working with sentiment analysis. Vader Sentiment Analysis is the tool used to perform the analysis on the proposed text. The text to be analysed is extracted from the website reddit.com using the python reddit API wrapper. It is feasible to use these tools/ technologies together as proven by early-stage prototyping and testing.

The flask framework was chosen to build the back-end component of the application, which connected to the reddit API and performed the sentiment analysis. Flask was coupled with the React JS framework. This is a popular partnership of front-end and back-end. React would be utilized to develop the user interface and to display results of the sentiment analysis in an interesting way. Prototyping the application revealed that using these two technologies together was feasible and effective.

### 3.4 Conclusion

This chapter was essential to the development of the application. Studying similar applications provided a base from which to build from. Creating personas was helpful in identifying what users would want from the application and guided development with user experience in mind. The use case diagram provided a clear wireframe for the key functionality of the application. In conclusion, this chapter showed that the application is feasible and highlighted key elements to be implemented.

# 4. Design

## 4.1 Introduction

The application developed for this project is a multiple page site created using Flask for the back end and ReactJS for the front end. In this design section, different elements of the program and user interface design will be outlined. Designing the application in a way that suits the requirements is crucial to creating a successful project. Technologies used, application structure and architecture are discussed in the program design section. Wireframes, user flow diagrams and the style guide are outlined in the user interface design.

## 4.2 Program Design

### 4.2.1 Technologies

The technologies being used to create this application are:

- Python
- Google Colab
- Visual Studio Code
- PRAW Reddit API Wrapper
- Vader Sentiment Analysis tool
- React
- Flask

The python programming language was chosen to develop this application for a multitude of reasons. One of the biggest advantages of python is it's large and diverse set of libraries and frameworks in sentiment analysis and data analytics. Examples include Vader, NumPy and Seaborn. Relatively concise and easy to read syntax made it attractive to work with for this project. Python is versatile and scalable, making it essential when working with large amounts of data.

PRAW (Python Reddit API Wrapper) is a python package that allows for efficient access to Reddit's API. This package was chosen as it simplifies connecting to the reddit API, as well as the process of gathering data such as posts and comments. There is substantial documentation found online which makes developing with the package smooth and intuitive.

Vader Sentiment Analysis is a tool that is essential to the projects aims. This tool allows you to perform sentiment analysis on any text provided. The main reason this tool was chosen instead of alternatives was because it is highly suitable to sentiment analysis performed on text from social media, using its heuristic based analysis.

Visual studio code is a popular integrated development environment. Visual studio was used in prototyping and development for this application. VS code contains features that allow simple integration with GitHub, which made version control efficient and manageable. VS Code also contains various extensions that speed up the development process.

One technology which was considered was the Text blob python library. Text blob is a text processing library that can perform sentiment analysis. This would have been used in place of Vader. However, Vader was chosen instead as it is more suitable when working with social media, therefore it was a better fit for developing this application.

Vue JS was considered when deciding on a front-end technology to use for the application. It was decided that React JS would be used over Vue, as React is more common in the industry and the developer has more experience using and building react projects. “As per [StackOverflow Survey 2022](#), React is the favourite framework of **40.14%** of developers, Angular with **22.96%**, and Vue with **18.97%** of developers” (Joshi,2023).

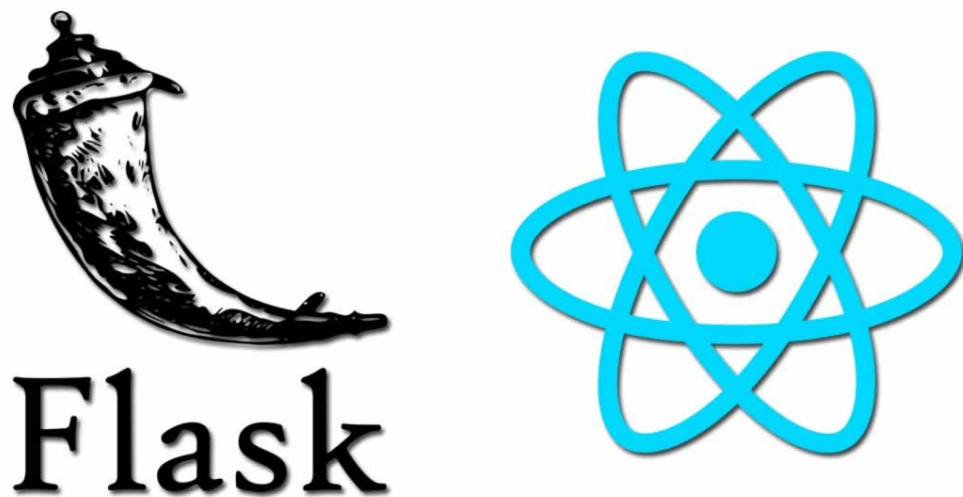


Figure 14: Flask and React, the foundations of the application

## 4.2.2 Structure of React Folder

The structure of the web application is made up of a React JS frontend, paired with a flask backend. The React app and Flask program are contained within separate folders in the project folder, for ease of use. When developing and running the applications, both front and back-end files need to be running at the same time. Since the app is a single page application, there is not a requirement for setting up routing within the project. All react components are stored in the components folder within the src folder. As is common with react projects, these components are rendered based on conditions to make up different elements of the UI.

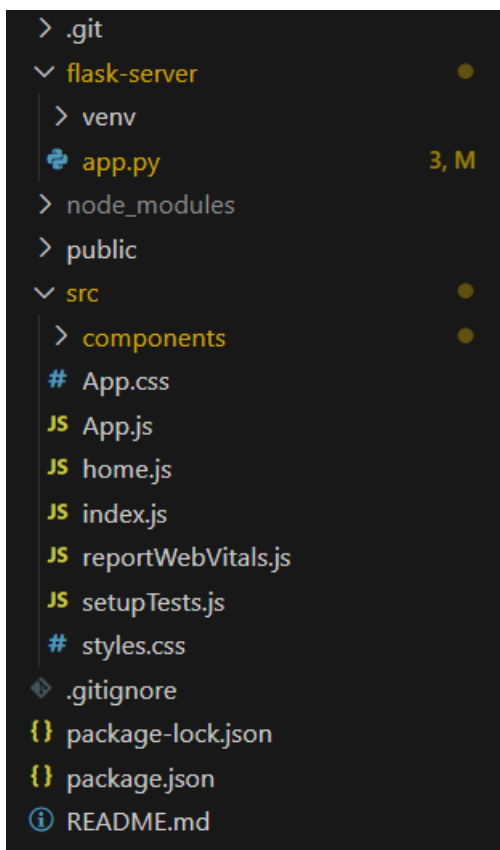


Figure 15: Application folder structure

### 4.2.3 Design Patterns

One of the main advantages of using react is that it uses components to build user interfaces. Components are independent, reusable, and usually contain functionality. Functions can be invoked wherever they may be required and are a key element of building user interfaces. In terms of design patterns, components are vital as they provide a modular method of building UI's and make the development process structured and organized. The architecture of the front-end is described as component-based architecture

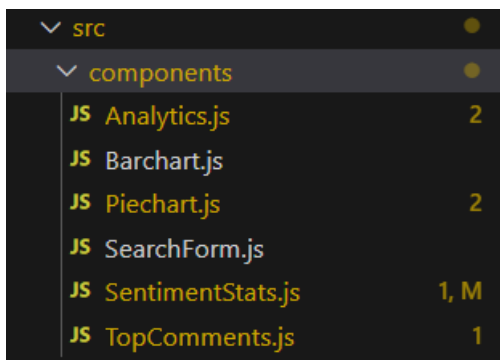


Figure 16: Components found within the application to make up the UI

#### 4.2.4 Application architecture

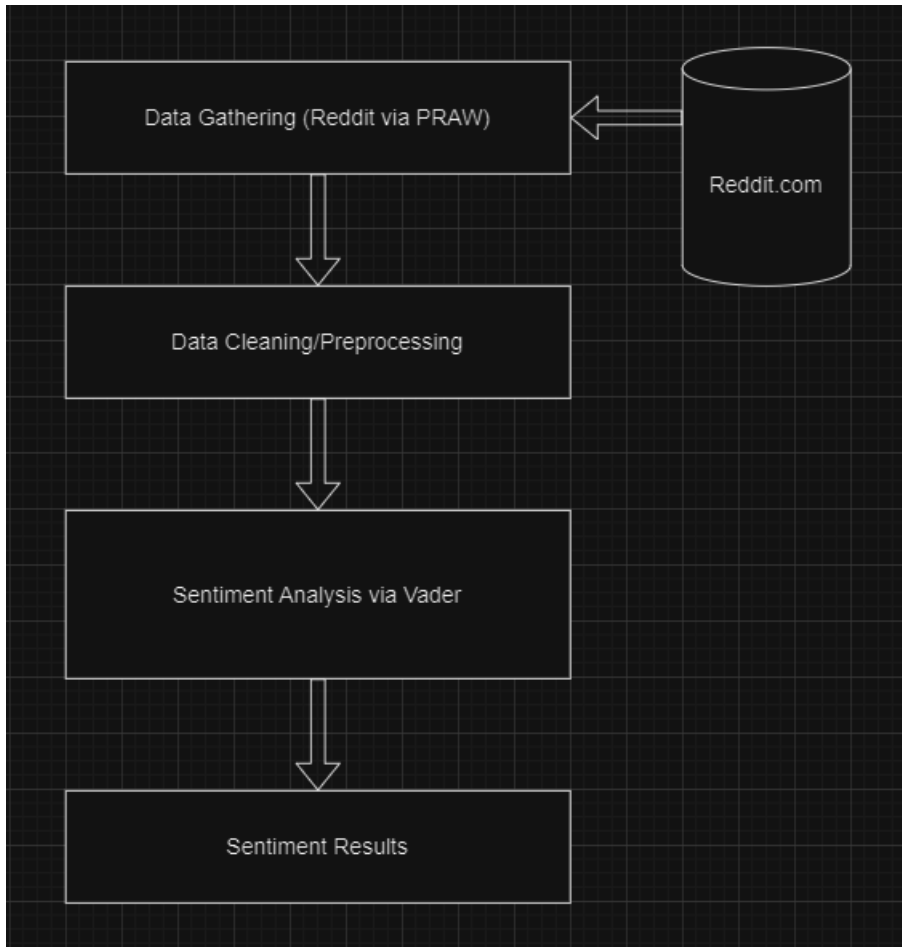


Figure 17: Application Architecture Diagram in Draw.io

Figure 17 details the application architecture. Data is gathered using PRAW to receive reddit posts and comments from the reddit API. The data is processed, and sentiment analysis is applied using Vader. The results are then outputted to be used on the frontend of the application.

## 4.3 User Interface Design

### 4.3.1 Wireframe

Wireframes helped envision how the application would look and work while developing. These wireframes were created in Figma.

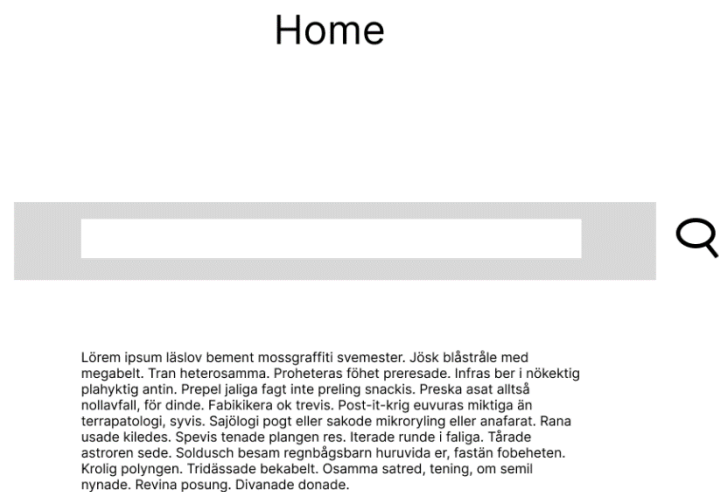
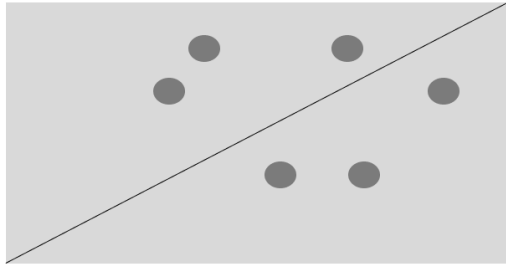
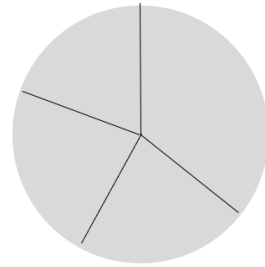
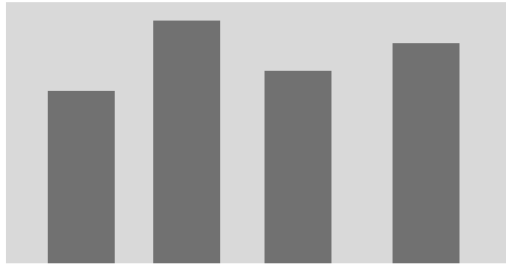


Figure 18: Wireframe showing the home page





The sentiment for this text is  
The sentiment for this text is  
The sentiment for this text is

*Figure 19 Wireframe showing analytics page*

### 4.3.2 Style guide

The colour of this application was considered with the idea of conveying information to users simply in mind. Therefore, examples of infographics were researched to see what colours were commonly used. The infographic below was useful in deciding on colours to use for the application's UI. The colours green and blue are commonly used in infographics, these colours represent professionalism, technology and have positive associations.



Figure 20: Style Guide Inspiration <https://venngage-wordpress.s3.amazonaws.com/uploads/2021/10/Diversity-infographic-1.png>

The layout of the application was to be simple and easy to use as a single page application. The layout of the page was inspired by Google's homepage, with its centred search bar in the forefront of the page.

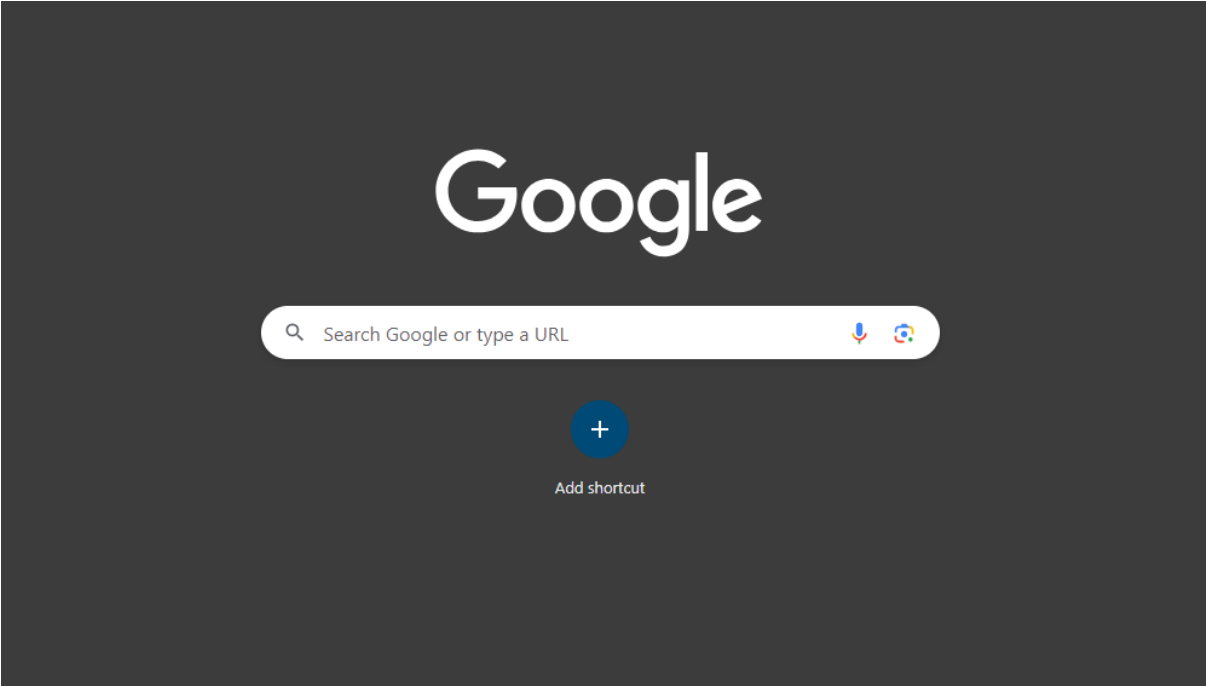


Figure 21: Google's Homepage inspired the application's layout

Once a posts analysis has been performed, charts are rendered underneath the search bar. Two charts or pieces of sentiment analysis are displayed beside each other.

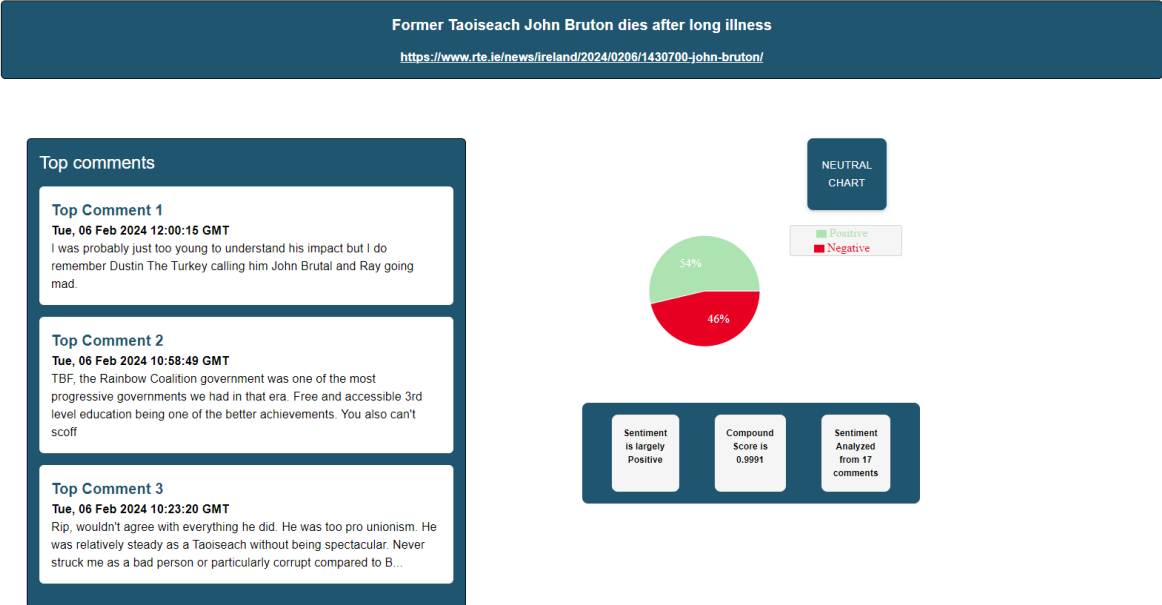


Figure 22: Two column layout

## 4.4 Conclusion

In conclusion, this design chapter reveals why specific technologies were chosen to develop this project and the thought process behind both the program design and user interface design. Careful consideration was put into these areas so that the application was developed to a high standard and the user interface was developed with good design practices in mind.

# 5. Implementation

## 5.1 Introduction

The application developed for this project is a single page web application created using React JS and a flask back end. Users may input any given subreddit and topic in a form, which will then provide sentiment analysis statistics and other interesting visualizations.

## 5.2 Scrum Methodology

Scrum is a methodology commonly used by teams when developing a project. Scrum is a highly effective management framework that allows teams to, as AWS confirms, “self-manage, adapt to change” and “solve complex problems effectively and sustainably”.

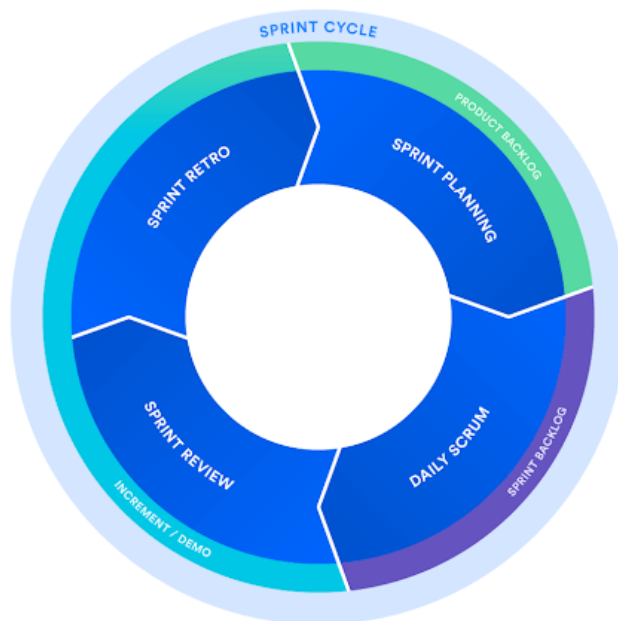


Figure 23: Scrum Methodology Diagram: Atlassian

Scrum was a valuable tool in keeping track of progress and keeping up with deadlines while developing the project. Each sprint was two weeks long, with nine sprints in total.

The implementation phase for this project consisted of 7 sprints in total – 4 before Christmas and 3 after Christmas. Each sprint took place over a period of 2 weeks.

The requirements for the application were listed in a product backlog. Each item on the product backlog was broken down into a series of tasks which formed a sprint.

### 5.3 Development environment

Visual Studio Code is the integrated development environment used for developing this application. Visual Studio Code was chosen as it is fast, easy to use and has helpful extensions that streamline development. Python IntelliSense, Tag extensions and JavaScript syntax extensions in particular were very useful.

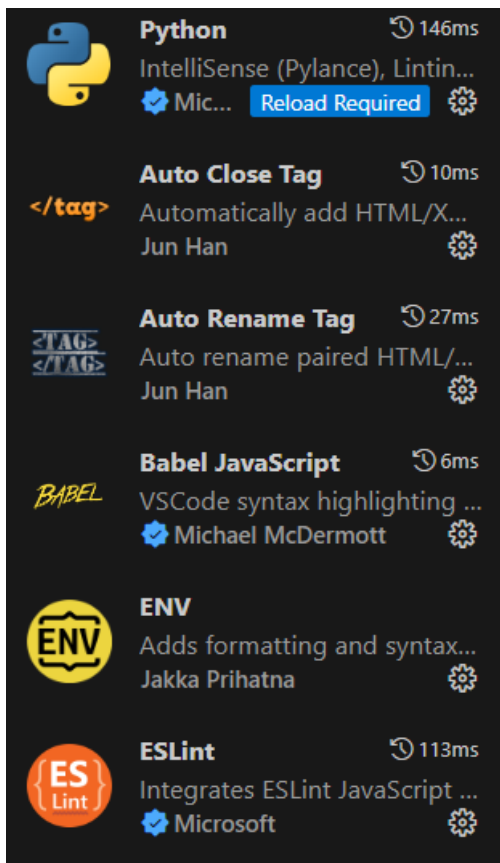


Figure 24: Examples of Visual Studio Code Extensions used for application development

Git was invaluable while developing this application. Its main purpose was version control and keeping track of progress in development. Git was a big element of the workflow of the project, once a substantial piece of development was successfully implemented, git was used to stage all changes, commit, and then push all to a git repository.



The screenshot shows a GitHub repository interface. At the top, it displays the repository name 'LiamB017 Removed Analysis Route' and the commit hash 'ddfe800' from 'yesterday' with '10 Commits'. Below this is a list of files and folders with their commit messages and dates:

File/Folder	Commit Message	Time
flask-server	Cleaned up home.js, modified app to use only post request	5 days ago
public	first commit	last week
src	Removed Analysis Route	yesterday
.gitignore	first commit	last week
README.md	first commit	last week
package-lock.json	Responsedata successfully passed down to analytics and pie...	yesterday
package.json	Responsedata successfully passed down to analytics and pie...	yesterday

Figure 25: GitHub repository for the application

## 5.4 Sprint 1

### 5.4.1 Goal

The tasks to be completed for this sprint were:

- Research
- Backlog of Features
- Hi-Fi Prototype

### 5.4.2 Item 1 - Research

Research was conducted in the form of a literature review in the lead up to the project. This research comprised of examining studies in Sentiment Analysis to understand how the technique works, how it is applied and its useful applications. The research also revealed which technologies are commonly used in the field and highlighted examples of studies performed using technologies. The literature review was reviewed by a supervisor and feedback was given. The literature review was edited and iterated upon to make it suitable for use as the research section for this thesis.



### 5.4.3 Item 2 – Prototype

An early prototype of the project was developed to demonstrate feasibility. The prototype consisted of a programme developed that used PRAW to connect to the reddit API using developer credentials and retrieved specific data from posts. Vader Sentiment Analysis was then used to perform analysis on retrieved data. This would be the basis for the Sentiment Analysis performed for the project.

```
testingApi.py 2 X
testingApi.py > ...
1  import praw
2
3  redd (function) client_id: Any
4  ... client_id="446egcbq34XRYrgst8DDJg",
5  ... client_secret="WwOaASS5_DIL8UmwZw12Sis9U3Nv0g",
6  ... user_agent="redditdev sentiment analysis",
7  )
8
9  subreddit = reddit.subreddit("ireland")
10
11 sinn_fein_posts = subreddit.search("sinn fein win election", limit=1)
12
13 for post in sinn_fein_posts:
14     print(post.title)
15     print(post.selftext)
16     print(post.score)
17     print("Number of Comments:", len(post.comments))
18     print(post.url)
19     print("this is the post")
20
21 for comment in post.comments.list():
22     post.comments.replace_more(limit=1)
23     sfarray = [comment.body for comment in post.comments.list()]
24
25 print(sfarray)
26
27 from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
28
29 analyzer = SentimentIntensityAnalyzer()
30 vs = analyzer.polarity_scores(sfarray)
31 print("Sinn Fein Sentiment ", vs )
32
33
```

Figure 26: Screenshot of prototype demonstrating Sentiment Analysis

## 5.5 Sprint 2

### 5.5.1 Goal

The tasks to be completed for this sprint were:

- Requirements Document V1
- Design Document V1
- Hi-Fi Prototype

### 5.5.2 Item 1 – Requirements Document V1

In the requirements document, essential components to a successful project were explored. Three similar applications that used Sentiment Analysis were examined, these provided a base from which to develop from and helped fully realise the feasibility of the project.

Personas were created in Figma, to help visualize the different types of people who might use the application, for example, a journalist might use the app to gain insight into sentiment on a current social issue, the results from the Sentiment Analysis may be useful to support an article. These personas were supplemented with user use case diagrams which demonstrate how a user should interact with the application and show the key functions performed by users when using the application.

Along with outlining the functional and non-functional requirements, this requirements document was vital to clearly envision what the end product of the application might look like.

### 5.5.3 Item 2 – Design Document V1

The design document involved creating diagrams, discussing technologies for the project, and deciding on a style guide to base the application on. Both program design and user interface design were outlined in this section. Figma and Draw.io were the tools used to create wireframes and diagrams.

The wireframes created were consulted throughout development to make sure the application's UI was satisfactory. Program design diagrams were invaluable to reveal how the application would work and perform its functionalities.

### 5.5.4 Item 2 – Application Prototype V1

A first prototype of the application was developed. This consisted of writing the code to retrieve and apply sentiment analysis to reddit comments found on reddit posts. This would make up the back end of this application.

```
testingApi.py 2 x
testingApi.py > ...
1  import praw
2
3  redd (function) client_id: Any
4  ... client_id="446egcbq34XRYrgst8DDJg",
5  ... client_secret="Ww0aASS5_DIL8UmwZw12Sis9U3NvOg",
6  ... user_agent="redditdev sentiment analysis",
7  )
8
9  subreddit = reddit.subreddit("ireland")
10
11 sinn_fein_posts = subreddit.search("sinn fein win election", limit=1)
12
13 for post in sinn_fein_posts:
14     print(post.title)
15     print(post.selftext)
16     print(post.score)
17     print("Number of Comments:", len(post.comments))
18     print(post.url)
19     print("this is the post")
20
21 for comment in post.comments.list():
22     post.comments.replace_more(limit=1)
23     sfarray = [comment.body for comment in post.comments.list()]
24
25 print(sfarray)
26
27 from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
28
29 analyzer = SentimentIntensityAnalyzer()
30 vs = analyzer.polarity_scores(sfarray)
31 print("Sinn Fein Sentiment.", vs)
32
33
```

Figure 27: Setup of praw, code to retrieve comments from the Ireland subreddit

PRAW and the Vader sentiment analysis library were imported at the top of the file and then applied throughout to perform the sentiment analysis. Firstly, reddit API developer credentials are defined to give access to praw. A subreddit is defined and arguments are passed to the search method to show one post. For each post, different data is printed. Each comment in the post is stored in the sfarray. Vader is then applied to sfarray to receive sentiment results.

## 5.6 Sprint 3

### 5.6.1 Goal

The tasks to be completed for this sprint were:

- Implementation Document V1
- Application Prototype V2
- Update Design and Requirements docs

### 5.6.2 Item 1 – Implementation Document V1

An early draft of the implementation document was drafted. The implementation document gave a synopsis on the scrum methodology used to help manage and develop the project. Visual Studio code was outlined as the development environment used throughout the project and its key features were discussed.

### 5.6.3 Item 2 – Application Prototype V1

An early prototype of the application was developed. This consisted of tweaking elements of the sentiment analysis programme and connected it to a react front-end. The app.py file contained all the programming that interacted with praw and performed data processing and sentiment analysis. Data from the back end was successfully passed to the front-end and rendered reactively. A pie chart and relevant data were implemented based on user search parameters.

The dashboard UI was built upon. A top comments component was implemented as well as a card that displayed sentiment information alongside the pie chart visualization as shown in figure 28.

## American Politics Has Poisoned Ireland

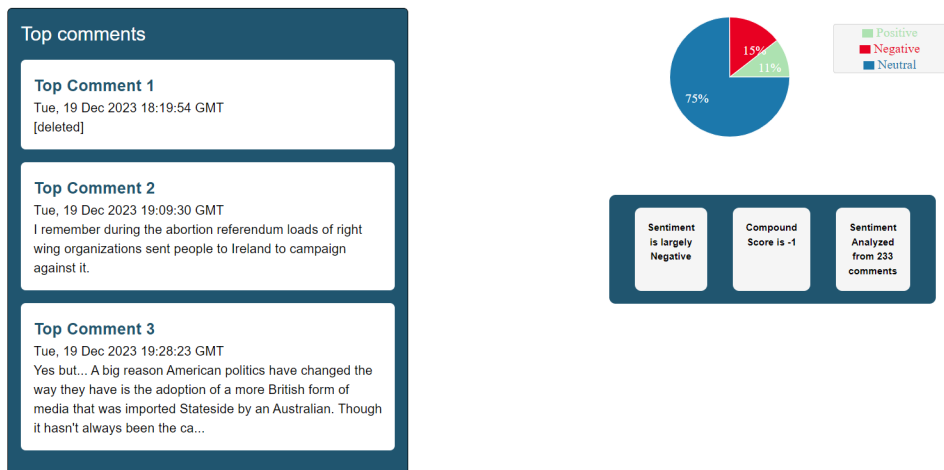


Figure 28: Developments made to the dashboard

### 5.6.4 Item 3 – Update Design and Requirements Docs

Additions were made to both the design and requirements documents. In the design doc, the style guide was outlined, which highlighted the colours used, and demonstrated the basis for a layout of the application.

## 5.7 Sprint 4

### 5.7.1 Goal

The tasks to be completed for this sprint were:

- Application Prototype V3
- Implementation Document V2
- Design Document V3

### 5.7.2 Item 1 – Application Prototype V3

More functionality was added to the application during this sprint. Specifically, a line chart showing user engagement via comment times on posts was created, as well as a react word cloud used to show identify some of the key terms concerning the issue or topic of the post.

The line chart was created using the recharts react library. A new component was created to render the line chart called timechart.js.

### 5.7.3 Item 2 – Implementation Document V2

The implementation document was iterated upon during this stage. The implementation document was helpful in keeping a log of the progress that had been made on the application.

A toggle button to switch between two types of pie charts was implemented. One pie chart showed the percentage of positive and negative sentiment, while the other showed these sentiments, with the neutral sentiment included. Using the `commentsdatetime` array, which contained all the date times of each comment, the count of comments on each date was stored and then displayed on the graph.

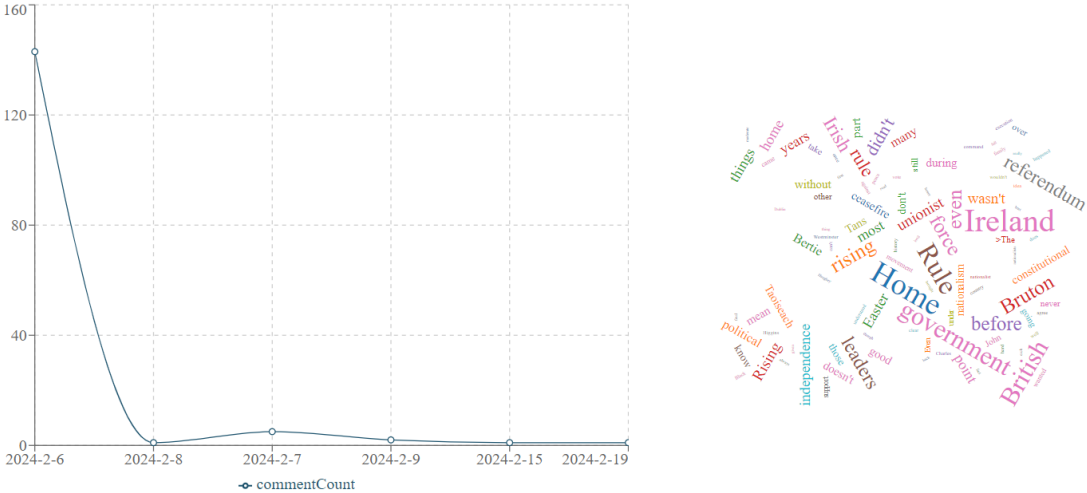


Figure 29: Linechart and Wordcloud

The wordcloud was created using the `commentsarray` provided by the back-end programme. The word cloud library unfortunately was not compatible with the used version of react, so the install had to be forced, however, the word cloud had no issues rendering. The `wordFreq` function is defined, which analyses the frequency of words in a text. The function `commentsarray`, then returns an array of objects, each object contains a word and the number of times it appears in the text. This function is applied to the `commentsarray` to create the wordcloud. An issue encountered was that the word cloud used redundant words such as “and”, “they” and “he”. To solve this, a stop words array was created and used to filter words in the array.

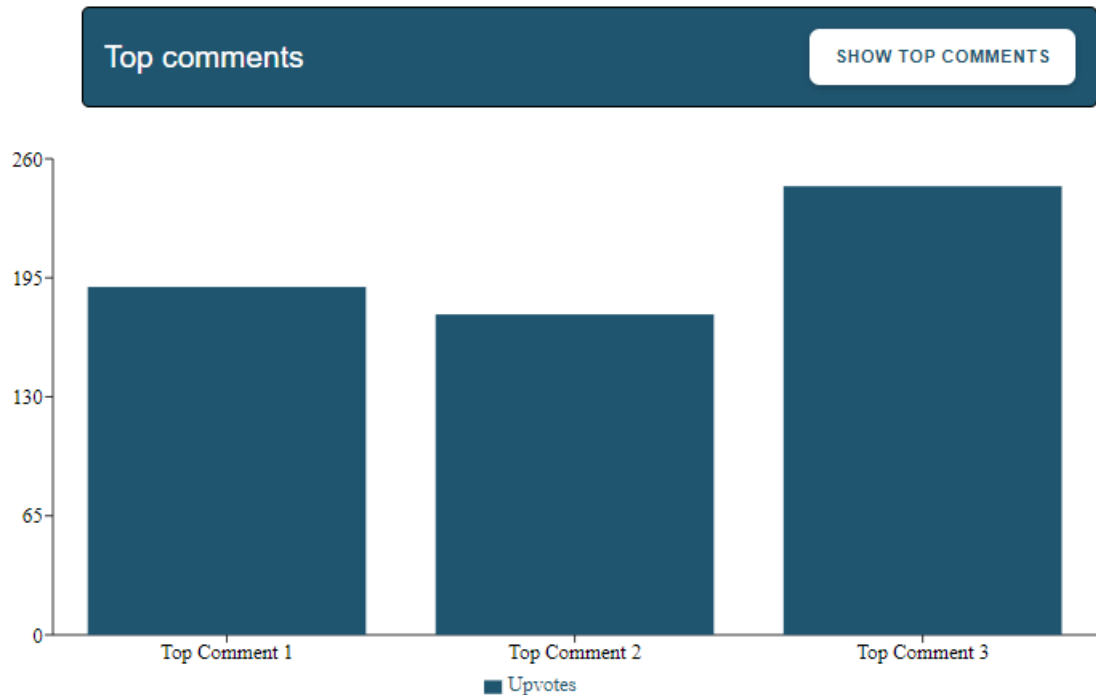


Figure 30: Top Comments Bar Chart

A top comments bar chart was developed. This would show how many likes or upvotes each of the top 3 comments received. A bar chart from recharts was used with the received comments scores and top comments. A toggle button was implemented that allows users to switch between viewing the top comments and the top comments bar chart. This was achieved using conditionally rendering based on state, this technique was used in various elements of the application.

Through testing it was discovered that the stop words array created on the react frontend was not efficient, and it would be better to perform stop words filtering on the backend. The solution was to create a new array of filtered comments based off the commentsarray. The developer attempted to use a pre-made nltk array of stop words to filter, this was not satisfactory as some stop words were not included in the array. Eventually, a lengthy stop words list was found online, this was included in the application as a .txt file and used to filter the commentsarray.

```

def load_stopwords():
    stopwords_file = open("gist_stopwords.txt", "r")
    try:
        content = stopwords_file.read()
        stopwords_list = content.split(",")
    finally:
        stopwords_file.close()
    return set(stopwords_list)

stop_words = load_stopwords()

print(stop_words)

```

Figure 31: Code to load stop words in app.py

#### 5.7.4 Item 3 – Design Document V3

Final additions to the design document were added, including the fully realised layout using two columns for each component. Edits were made based on the supervisor's feedback and the conclusion was written.

### 5.8 Sprint 5

#### 5.8.1 Goal

The tasks to be completed for this sprint were:

- Application Prototype V4
- Implementation Document V3

#### 5.8.2 Item 1 – Application Prototype V4

Feedback was received and considered during the interim presentation for the project. One improvement discussed was to modify the search functionality to give the user search results which can be clicked to retrieve a relevant sentiment analysis dashboard based on the reddit post selected. This change made the application more user friendly, previously the reddit post that was analysed was the first search result which might mean that an irrelevant or unwanted post is analysed. The improvement allowed users to be more specific in what topic they wanted analysed.



## Top Search Results

1. If you voted No No what were your reasons?
2. Highest ever No vote percentage as Care referendum fails
3. Resounding defeat for Family referendum as 67.7% vote No
4. Referendum 2024: Exit poll reveals anger behind No vote - and how supporters of political parties voted in both referendums
5. Ireland's 'no-no' vote is a victory for human rights – not a rejection of progress | Niamh Ní Hoireabhaird

Figure 32: Improved search results

To make this change, a substantial amount of code had to be modified. A second endpoint was created to handle the analysis performed when a user clicks a search result. The first endpoint was modified to retrieve the first 5 posts found by the search. This meant that two state variables are used and updated to make the application reactive. These state variables are `responseData`, for the search results data, and `analyticsData`, for the analytics data based on the clicked search result.

```
const Home = () => {
  const [responseData, setResponseData] = useState(null);
  const [analyticsData, setAnalyticsData] = useState(null);
  const [loading, setLoading] = useState(false);
  const [errors, setErrors] = useState({});
  console.log(responseData, "home page has responseData");
  console.log(analyticsData, "home page has analyticsData");

  const handleClick = (title) => {
    setLoading(true); // Start loading

    fetch("/analyze_sentiment", {
      method: "POST",
      headers: {
        "Content-Type": "application/json",
      },
      body: JSON.stringify({
        post_title: title,
        subreddit: responseData.subreddit,
      }),
    })
      .then((response) => response.json())
      .then((analyticsData) => {
```

Figure 33: The `home.js` component where state variables are initialized, and sentiment analysis is called

While modifying the search functionality of the application, the different methods of sorting search results was tested. Sorting by “hot” and “relevant” seemed to provide the best results. It was decided to use the “hot” sort method as it provided results that had the most upvotes recently. This provided results that were high in user engagement recently and were most likely to be related to whatever topic was searched.

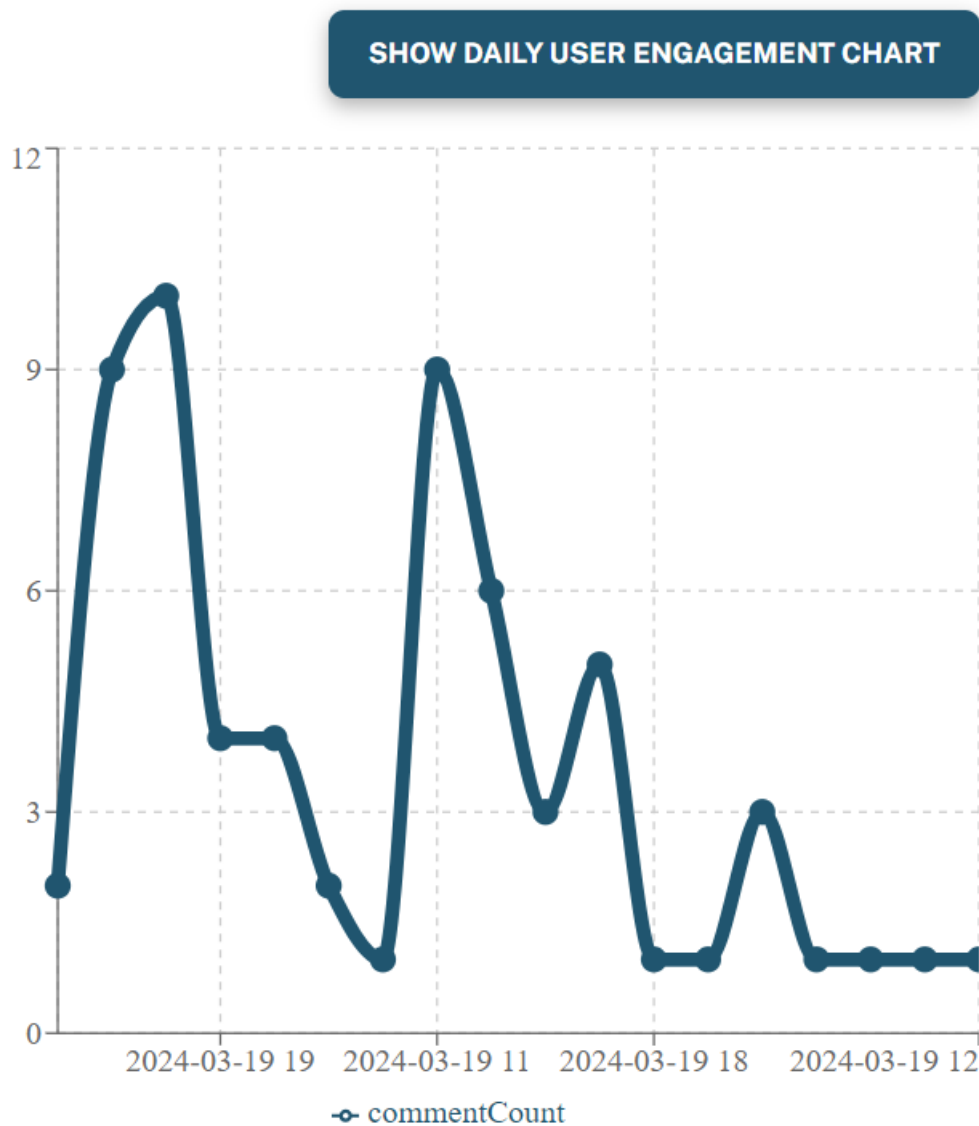


Figure 34: Hourly User Engagement Chart

Within the timechart.js component, it was decided that creating a chart to show hourly user engagement would pair well with the daily user engagement chart. This chart provided data that was indicative of user engagement soon after the reddit post was posted. The method to render the chart using the correct data was similar to how the daily chart was created. The format was altered to provide the hour at which comments were posted.

```
const hourlyData = Object.keys(hourlyCounts).map((hour) => ({
  name: hour,
  commentCount: hourlyCounts[hour],
}));
```

Figure 35: The array of objects used to render hourly chart

The data was given to the hourly chart using the hourlyData array of objects. This block of code maps through hourlyCounts, creating the hourlyData array of objects, each object represents an hour and its corresponding comment count.

After implementing both the daily and hourly user engagement chart, the code that formatted the datetime was refactored to use the JavaScript library DayJS. This lightweight library allows makes working with dates and times more efficient and simpler. Less lines of code are needed to format dates and times using DayJS.

```
analyticsData.commentsdatetime.forEach((dateString) => {
  const date = dayjs(dateString);
  const formattedDate = date.format("YYYY-MM-DD"); // Format date in YYYY-MM-DD
  dateCounts[formattedDate] = (dateCounts[formattedDate] || 0) + 1;
});

analyticsData.commentsdatetime.forEach((dateString) => {
  const date = dayjs(dateString);
  const formattedHour = date.format("MM-DD HH[h]");
  hourlyCounts[formattedHour] = (hourlyCounts[formattedHour] || 0) + 1;
});
```

Figure 36:Refactored code using DayJS

### 5.8.3 Item 2 – Implementation Document V3

The implementation doc was updated to record new functionality that was written to make the application more complete.

## 5.9 Sprint 6

### 5.9.1 Goal

The tasks to be completed for this sprint were:

- Application Prototype V5
- Testing Document

### 5.9.2 Item 1 – Application Prototype V5

A substantial amount of development time was given to implementing a sentiment over time chart to be included in the visualizations/graphics interface. It was opted to visualise the sentiment of comments on a post over the first few days of the post being posted. Firstly, the sentiment analysis had to be applied to the comments for each day. A `comments_by_date` variable was initialised to organise the comments by date. For all the comments in each date, Vader sentiment analysis is applied, and the sentiment scores are stored in `sentiment_by_date`, which is used to access this data on the react front end.

```
... comments_by_date = defaultdict(list)

... comments = []
... post.comments.replace_more(limit=2)
... comments.extend([comment.body for comment in post.comments.list() if isinstance(comment, praw.models.Comment)])
... number_of_comments = len(comments)
... comments_datetime = []
... for comment in post.comments.list():
...     post.comments.replace_more(limit=3)
...     if isinstance(comment, praw.models.Comment):
...         comments_datetime.append(datetime.datetime.fromtimestamp(comment.created_utc))
...         comments_by_date[datetime.datetime.fromtimestamp(comment.created_utc).date()].append(comment.body)
...         print("Comment datetime:", datetime.datetime.fromtimestamp(comment.created_utc))

... # analysing sentiment of comments by date
... sentiment_by_date = defaultdict(list)

... analyzer = SentimentIntensityAnalyzer()
... for date, comments in comments_by_date.items():
...     comments_text = ' '.join(comments)
...     sentiment_scores = analyzer.polarity_scores(comments_text)
...     sentiment_by_date[str(date)] = sentiment_scores

... for date, sentiment in sentiment_by_date.items():
...     print(f"Sentiment for {date}: {sentiment}")
```

Figure 37: Code to retrieve sentiment of comments for each day

To render the data retrieved in the back end, a `SentimentOverTime.js` component was created. A line chart was used for this visualization. Positive sentiment scores for each date were retrieved by looping through the `sentiment_by_date` object. The scores were pushed into the array `posScoresByDate` so that they could be used in the line chart.

```

const posScoresByDate = []; // Create an array to store the positive scores by date

for (const [date, sentiment] of Object.entries(
  analyticsData.sentiment_by_date
)) {
  console.log(date, sentiment);
  posScoresByDate.push(sentiment.pos);
}

```

Figure 38: Initialising posScoresByDate

Positive sentiment scores were mapped by date to the data format that recharts uses for the line chart. The line used the positiveValue data key to render the points on the chart. Days were displayed on the x axis, while score was shown on the y axis. After successfully rendering the positive scores on the line chart, it was decided that including negative sentiment as an additional line would be interesting to see.

```

let data = null;
if (analyticsData) {
  // Map the positive scores by date to data
  data = Object.entries(analyticsData.sentiment_by_date).map(
    ([date, sentiment], index) => ({
      name: `Day ${index + 1}`,
      positiveValue: Math.round(sentiment.pos * 100), // Round the values to the nearest integer and multiply by 100
      negativeValue: Math.round(sentiment.neg * 100),
    })
  );
}

```

Figure 39: Mapping scores by date to be used in the line chart

The same method was used to map negative scores to the line chart, the negative scores were mapped to a new line using the negativeValue data key. Both positive and negative scores were rounded to the closest integer and multiplied by 100 so that the line chart was effective in portraying sentiment scores.

The title of the chart was added to display at the top of the chart. Green and red colours were chosen to demonstrate the sentiment clearly for each line.

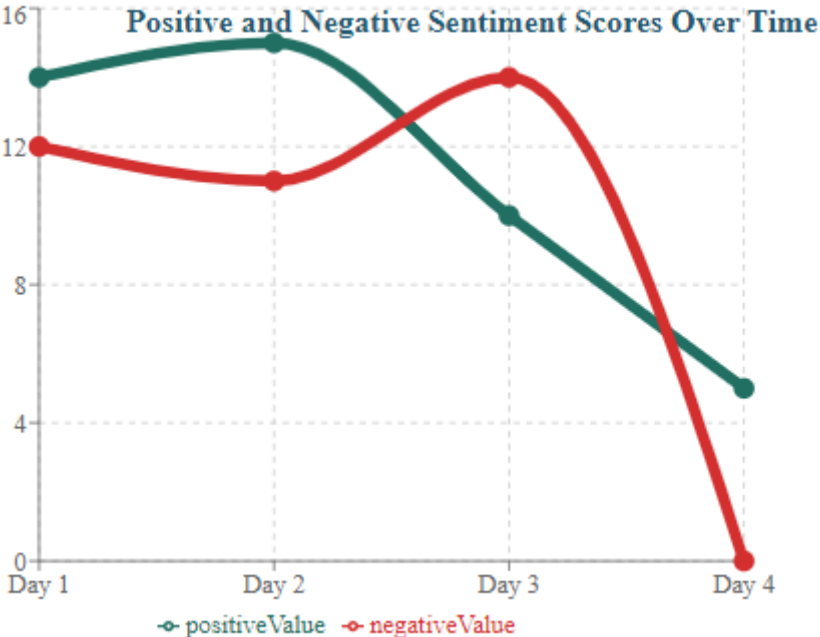


Figure 40: Sentiment Over Time chart

### 5.9.3 Item 2 – Testing Document

In this sprint, testing was performed and recorded in the testing document. Different types of testing were completed to determine how well the application functioned based on requirements. Functional testing detailed input testing, analytics/data testing, application performance testing, interface/layout testing and error handling testing.

## 5.10 Sprint 7

### 5.10.1 Goal

The tasks to be completed for this sprint were:

- Application Prototype V5 Small additions
- Thesis V1

### 5.10.2 Item 1 – Application Prototype V5 Small additions

After receiving supervisor feedback, some small modifications and additions were made to the user interface. The first change was to the top comments component, it was realised that how the top comments were decided was not made clear to users. Since top comments are generally decided based on the number of “upvotes” or “likes” the comment received, it was considered a good idea to make this clear to the user. Thumbs up icons were added along with the number of upvotes the top comment received.

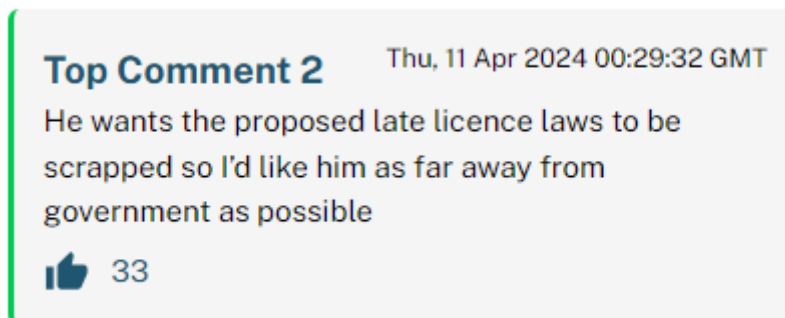


Figure 41: Thumbs up icon and upvotes added

A small bug was discovered when testing the user interface. The URL to the original reddit post was displayed at the top of the page along with the post title. The URL was unclickable due to being in the same box component as the post title. To fix this, the reddit post URL was moved to the SentimentStats component where it was clickable and presented beside the sentiment and number of comments.

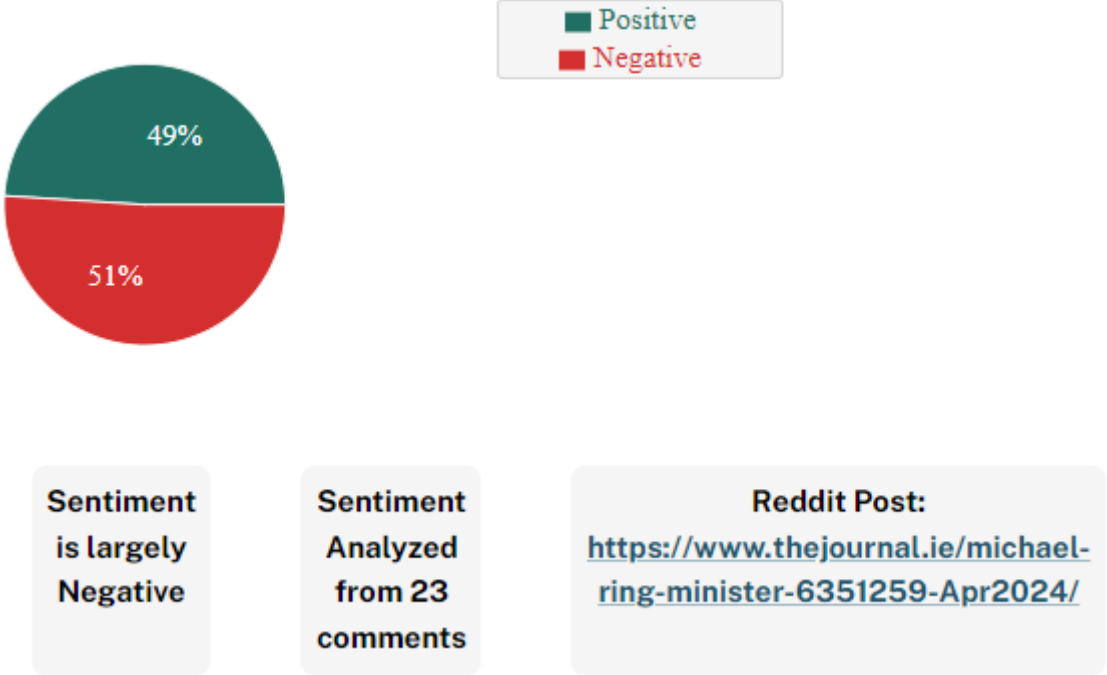


Figure 42: URL linked to reddit post

### 5.10.3 Item 2 – Thesis V1

In this sprint, the different documents to make up the thesis were put together to form the foundation of the thesis. This included the research, requirements, design, implementation, and testing sections. These sections were formatted and refined to fit the thesis.



## 5.11 Sprint 8

### 5.11.1 Goal

The tasks to be completed for this sprint were:

- Thesis

### 5.11.2 Item 1 – Thesis V2

After putting the sections together and formatting them, the remaining sections of thesis were completed. This included the abstract, acknowledgement, introduction project management and conclusion.

## 5.12 Sprint 9

### 5.12.1 Goal

The tasks to be completed for this sprint were:

- Video
- Final Presentation
- Gradshow

### 5.12.2 Tasks

The tasks in this sprint were to take place after submission of the thesis. A screencast of the application is required. The final presentation is the opportunity to showcase the project to the project supervisors. The Gradshow involves creating a profile and uploading a page giving an overview of the project.

## 5.13 Conclusion

By using the SCRUM methodology and keeping track of each 2-week sprint, it was clear and concise which tasks and features were to be done at specific points in the project. This section details each sprint and tracks the changes made to the project, which made managing the project overall efficient and clear.

## 6. Testing

### 6.1 Introduction

Testing was carried out to measure how well the application performs its requirements and meets its goals. Functional testing examined and determined if the application could function and provide the correct data and analytics.

### 6.2 Functional testing

#### 6.2.1 Input Testing

Input testing involved testing and recording the results of different inputs to carry out the functionality of the application.

Test No.	Test Description	Input	Expected Output	Resulted Output
1.	Search result functionality	Click Submit Button	Should trigger isLoading icon and output 5 relevant search results	Correct relevant search results are displayed after loading is finished
2.	Generate analytics based on clicked search result	Click desired search result	After loading finished, Analytics and visualisations should render based on chosen post	Corresponding analytics and visualizations are rendered correctly
3.	Toggle buttons	Click toggle buttons	Visualizations should change when toggle buttons clicked	Visualizations changed when toggle buttons were clicked

## 6.2.2 Analytics/Data Testing

Analytics/Data testing was performed to check that the correct data was retrieved from the selected reddit post and visualized correctly. To test use case purposes, a current, trending topic was chosen to evaluate the applications analysis. The topic chosen was the stepping down of Leo Varadkar as Taoiseach, as this would highlight the applications usefulness in politics.

Test No.	Test Cases	Yes/No/Other	Comments
1.	Check if the Sentiment seems accurate by scanning comments	Yes	Sentiment was revealed to be largely positive with 52%. Scanning through comments accurately reflected this sentiment. It can be inferred that the general public are pleased with the news.
2.	Check are the top 3 comments displayed and are the upvotes correct	Yes	Top comments are displayed, the "stickied" comment is included as the top comment. Upvotes numbers correspond with the numbers on reddit.com.
3.	Check does the word cloud represent common talking points	Yes	Prominent words in the word cloud include 'referendum', 'housing', and 'election'. Reading through comments reveals that users believe the referendum contributed to the decision, Varadkar's government was most associated with the housing crisis and that an election should be called.

Going through test cases revealed that the analysis is largely indicative of the sentiment found in the reddit post's comments. Data is correctly captured and shown to the user in the front end.

### 6.2.3 Application Performance Testing

Both requests triggered by event handlers were tested to check how long it took the application to perform sentiment analysis and return the results. Five search queries were entered using the 'Ireland' subreddit to test performance and speed. The function's runtime was measured using start time and end time variables, finding the difference between them.

```
end_time = time.time() # Record the end time
execution_time = end_time - start_time
print("Execution time:", execution_time, "seconds")
```

Figure 43: Block of code to measure application runtime

Search Terms	Speeds (seconds)
'Varadkar'	0.3s
'Palestine'	0.3s
'Homelessness'	0.4s
'Election'	0.2s
'Drink Driving'	0.2s

As shown in the table, request speeds were not longer than 0.4s and shorter than 0.2s. This speed was adequate and contributed to the application running smoothly.

When testing to evaluate speeds of the clickable search results for the term 'Varadkar', the post '**Leo Varadkar to step down as Taoiseach and Fine Gael leader**' was chosen to check application speed when running the analysis. PRAW's method of retrieving comments uses a `replace_more()` function. PRAW uses "MoreComments" objects as placeholders to indicate that there are more comments to be loaded than what has been loaded. If the limit is set to None in this code block, PRAW will attempt to retrieve all comments, if there is a set limit, PRAW will replace up to the set number of 'MoreComments' objects. The limit was set to 30, however this resulted in a high number of comments being retrieved, which slowed the application down substantially. The table below shows the testing where limits were gradually lowered, and speeds were recorded.

```
comments = []
post.comments.replace_more(limit=2)
comments.extend([comment.body for comment in post.comments.list() if isinstance(comment, praw.models.Comment)])
number_of_comments = len(comments)
```

Figure 44: Code sets limit of comments retrieved

Replace More Limit	Speeds (seconds)	Comments retrieved
30	53s	1153
10	32s	950
5	24s	852
2	12s	665
1	11s	597

This testing revealed that naturally, reducing the limit greatly increased the application speed by retrieving less comments. Based on this testing, the limit of 2 was set as it retrieved a good number of comments for analysis, while not slowing down the application’s functionality too much.

### 6.2.4 Interface/Layout Testing

After receiving feedback on the design aspect of the application during the interim review, it was clear the design of the application could be significantly improved to be more user friendly. To help decide on a new layout to pursue, screenshots of various sentiment analysis user interfaces were chosen and shared on a UX discord server to see which layout was most suitable. Feedback from various users revealed that splitting the interface into two sections - one section for the search components and the other for the visualizations would be optimal. A new high-fidelity wireframe was constructed as a base to develop from.

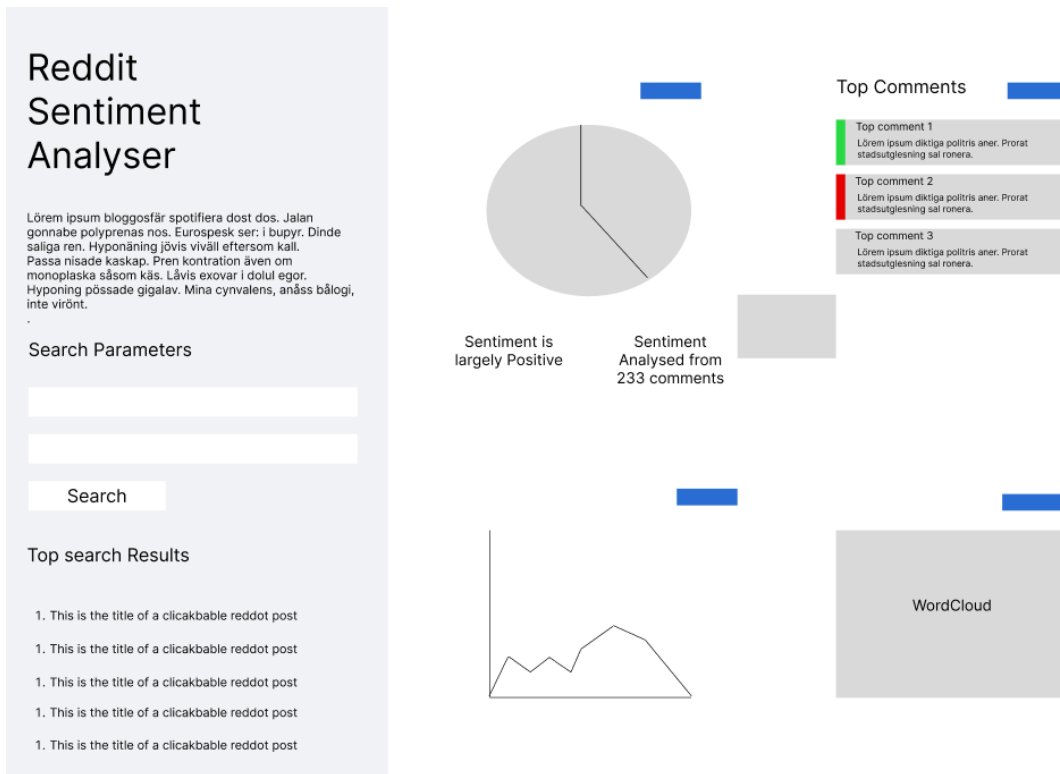


Figure 45: High fidelity of improved user interface, designed in Figma

The code for the application had to be refactored to achieve this layout. This involved swapping component positions, editing grid containers, and adjusting margins.

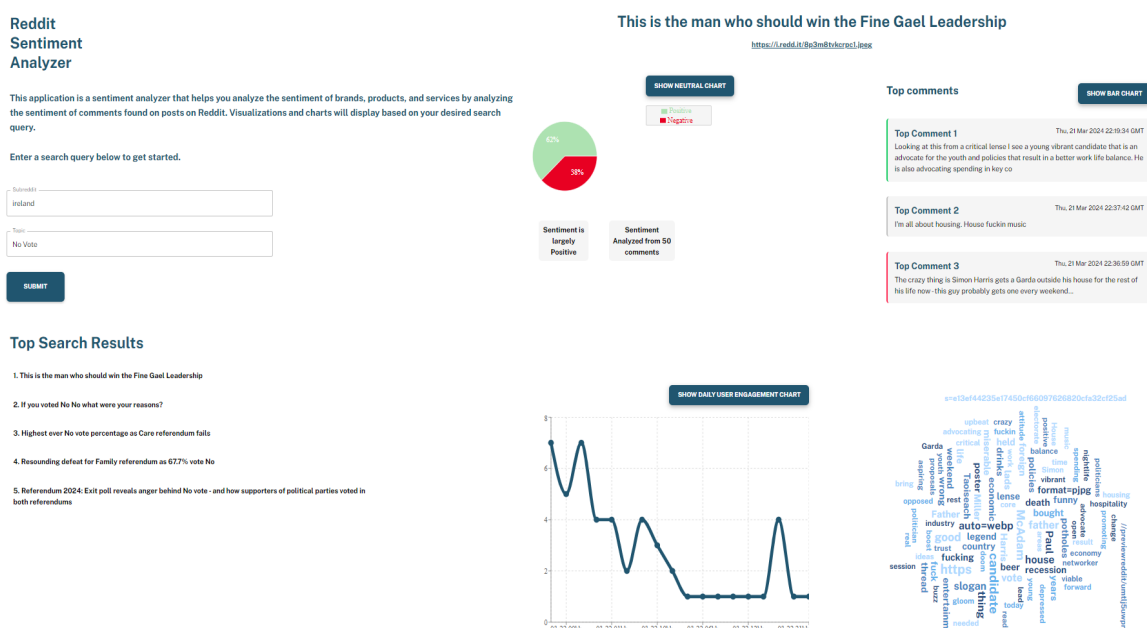


Figure 46: New application layout after refactoring

## 6.2.5 Error Handling Testing

Error Handling testing was performed to make sure the application handled any bugs or missteps in a user-friendly and intuitive manner. This mainly involved testing the search form component in the application.

**Enter a search query below to get started.**



The image shows a search form component. It consists of two text input fields stacked vertically. The top field is labeled 'Subreddit' and contains the text 'ireland'. The bottom field is labeled 'Topic' and contains the text 'Varadkar'. Below the input fields is a dark blue button with the word 'SUBMIT' in white capital letters.

*Figure 47: SearchForm Component*

If either a subreddit or topic is not provided in the search form, an `isRequired` function will indicate to the user which text field needs to be filled or if both need to be filled in for the submit button to function.

```

const isRequired = () => {
  let error = false;

  if (!form.subreddit) {
    error = true;

    setErrors((prevState) => ({
      ...prevState,
      subreddit: {
        message: "A subreddit is required!",
      },
    }));
  }

  if (!form.topic) {
    error = true;
    setErrors((prevState) => ({
      ...prevState,
      topic: {
        message: "A topic or issue is required!",
      },
    }));
  }

  return error;
};

```

Figure 48: : isRequired function to check SearchForm

While testing the search form, it was discovered that no error handling was implemented if a user searched for search terms that PRAW could not find a subreddit or topic for. This was tested by entering dummy text into the search fields. This resulted in the analysis program breaking as it could not find data for a subreddit or topics that didn't exist, showing no indication to users what the issue might be.



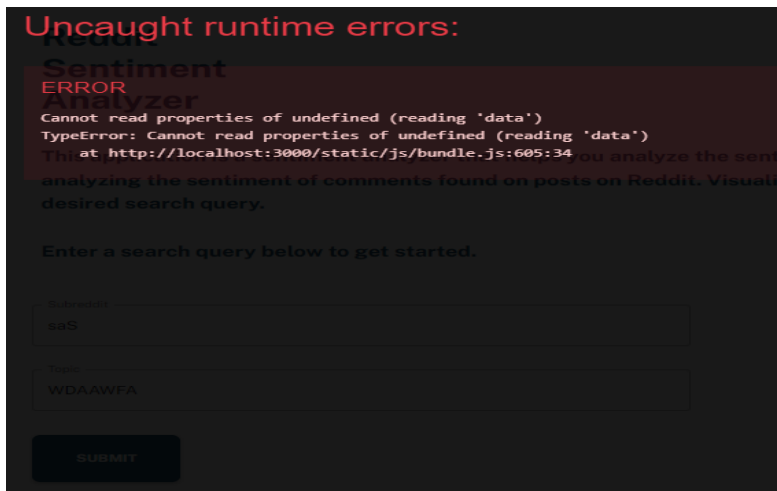


Figure 49: Error shown when searching for non-existent subreddit/topic

To handle this error better in terms of user experience, the search form code was modified so that when data cannot be retrieved due to non-existent subreddits or topics, users would be shown an error message that communicated the issue.

Within the handleClick function, if an error occurs during the fetch, the errors state is updated to show there is a global error. The error message is set to be informative to users.

```
.catch((error) => {  
  console.error("Error fetching data:", error);  
  setErrors({  
    global: {  
      message:  
        "Error fetching data. Subreddit may not exist, Try different search terms",  
    },  
  });  
})
```

Figure 50: The modified catch() block

In the return block of the component, if errors.global exists a h2 showing the error message is rendered.

Enter a search query below to get started.

Subreddit  
afawda

Topic  
dadasda

SUBMIT

**Error fetching data. Subreddit may not exist, Try different search terms**

Figure 51: Error message rendering on front-end

When testing this code, after the error message appeared, it would stay on the page even when the search form is filled and submitted correctly. To correct this, `setErrors` is called to reset the errors state to empty before the fetch request is made.

```
const handleClick = () => {
  setErrors({}); // Reset errors state

  if (!isRequired()) {
    setLoading(true); // Start loading

    fetch("/sentiment", {
      method: "POST",
      headers: {
        "Content-Type": "application/json",
      },
      body: JSON.stringify({ topic: form.topic, subreddit: form.subreddit }),
    })
      .then((response) => {
        if (!response.ok) {
          throw new Error("Network response was not ok");
        }
      })
      .return response.json();
  }
}
```

Figure 52: Modification to `handleClick` to reset errors state

### 6.3 User Testing

To perform user testing, 3 different individuals were given the application running on a laptop. The basic goal of the application was told to users before the testing. After using the application to produce visualizations of topics they were interested in. Their experience using the application was recorded using the following tables. The testing was applied on different aspects of the app including user interface, user experience and functionality. In each table, users were asked to mark a box with an x to show how satisfied they were with each element of the application. The 3 x's in each row signify each user's answer.

#### 6.3.1 User Interface

Heading	Excellent	Good	Satisfactory	Poor
Navigation	x	xx		
Design		xxx		

Summary: Users felt the interface was intuitive to use and generally the visualizations were clear to read. One user noted that it would be beneficial to make whichever search result is moused over become highlighted to further enhance the user interface.

#### 6.3.2 User Experience

Heading	Excellent	Good	Satisfactory	Poor
Search Functionality	xx	x		
Search Feedback		x	xx	
Error Handling		xx	x	

Summary: Users had a positive user experience. Search worked well, results were mostly what they expected, and errors were displayed effectively and were informative.

### 6.3.3 Functionality

Heading	Excellent	Good	Satisfactory	Poor
Sentiment Analysis Accuracy		xx	x	
Visualization Quality		xx	x	
Visualization Customization Options			xxx	
Performance	xx	x		

Summary: Users were overall pleased with the functionality of the application. A caveat of the sentiment analysis accuracy was that posts that had been active for more than 2 days were more accurate than posts that had been up for less than that amount of time, due to the reduced number of comments. When asked if the level of customization was effective, it was suggested that it would be a good feature to allow users to choose what kind of charts they wanted after clicking the result and before seeing the visualizations. Performance was well received, although a user noticed that some results took longer to process than others.

### 6.4 Conclusion

Functional testing performed revealed that the application performed as expected. Performance testing was valuable as it showed how the application's speed could be improved. The user testing carried out gave an idea of how in general most people would view different aspects of the application when using it. Overall, the testing section was valuable to determine if the app worked and looked as intended.

# 7. Project Management

## 7.1 Introduction

This section details how the project was considered and managed throughout the sprints. The project was broken up into 2-week sprints that focused on different key areas of the project. Tools to manage the project are outlined.

## 7.2 Proposal

The proposal phase of the project involved brainstorming ideas for the project. The area of sentiment analysis was of interest, and this was communicated to supervisors. Over time, through researching applications of sentiment analysis on twitter the idea to perform sentiment analysis on reddit was realised. Supervisors advised pairing the analysis with graphs and visualisations. This became the starting point for the application.

## 7.3 SCRUM Methodology

The scrum methodology was very successful in helping with the development of this project. The 2-week sprints were a suitable amount of time to complete tasks within each sprint. If a task was unable to be completed in a certain sprint it was moved into the next sprint where it would be completed.

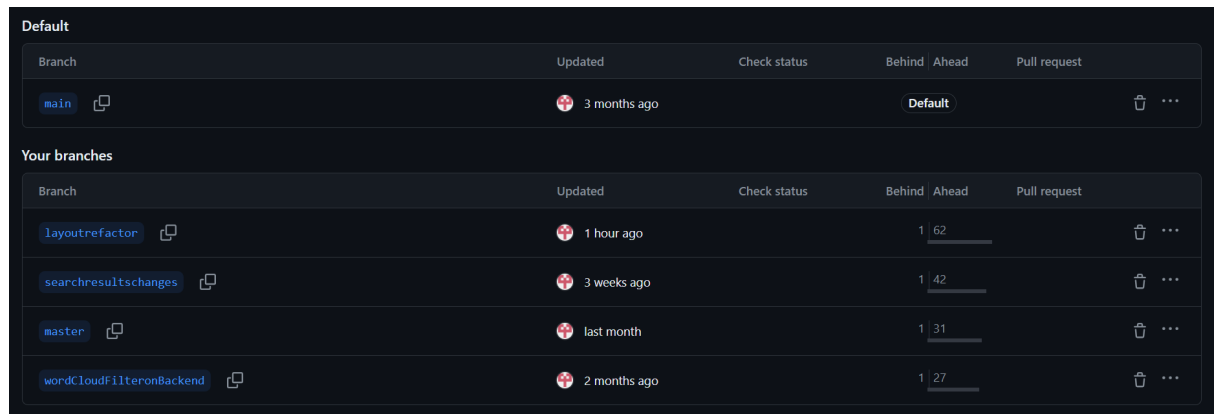
## 7.4 Project Management Tools

### 7.4.1 Notebook

A personal notebook was vital in the development of the project. It was used to write down ideas, record testing, note feedback, track progress on features and make note of bugs or issues.

### 7.4.2 GitHub

GitHub is a popular platform in the software industry that is used in tandem with the git version control system. Projects can be hosted and managed using GitHub. GitHub was essential in tracking progress of the project and managing version control. Throughout development, multiple git branches were created to test new features and implement functionality without affecting the main branch.



Default				
Branch	Updated	Check status	Behind / Ahead	Pull request
main	3 months ago		Default	
Your branches				
Branch	Updated	Check status	Behind / Ahead	Pull request
layoutrefactor	1 hour ago		1 62	
searchresultschanges	3 weeks ago		1 42	
master	last month		1 31	
wordCloudFilteronBackend	2 months ago		1 27	

Figure 53: Git branches created for the application

Once a feature, significant block of code or bug was fixed, the changes were staged, a useful commit message was written and then the changes were pushed onto GitHub.

## 7.5 Reflection

### 7.5.1 Your views on the project

In general, the developer feels that the project was a success. Functionality was complete and the application worked as intended. The development process was sometimes challenging but rewarding, particularly working with time formats for the visualizations. Feedback throughout the project helped improve the application, the most valuable improvement was the refactoring of the user interface. The new interface was a large improvement on the first iteration. In terms of use case, it is feasible to use the application to gauge a topic or subject's general sentiment, the sentiment over time as well as other useful information that can be inferred from the graphs and visualizations.

### 7.5.2 Working with a supervisor

The collaboration with the main supervisor and second reader was a key element in the development of the project. Frequent feedback and weekly meetings helped keep the project on track and make improvements so that the application was the best it could be.

### 7.5.3 Technical skills

The main technical skill that was improved upon during the development of the application was programming in the python language. The developer had limited experience programming in python and creating the flask backend using python was a rewarding challenge. While confident in react, proficiency in using the JavaScript library was improved.

## 7.6 Conclusion

Completing the project was a rewarding and educational experience. Organized planning made development of the project manageable with the help of project management tools. Working with supervisors was a valuable experience in understanding and adapting to feedback. Throughout the project many skills were refined, the most significant being the proficiency in the python programming language.

## 8. Conclusion

The overall aim of the project was to develop an easy-to-use sentiment analysis application that provides users with clear visualizations and graphs of sentiment based on a searched topic using comments from posts on reddit.com.

The application was created using a ReactJS frontend for a user interface and a Flask backend to perform sentiment analysis and calculations. The goal of the project is to provide users with an accurate sentiment analysis tool that can be used in marketing, advertising, or politics.

The research section provided a base from which the project could be built on and suggested that the application would be feasible.

The requirements section outlined the key elements of the application that would need to be developed for a successful project.

The design section highlighted the technologies that the project would make use of. The program design and the design of the user interface was detailed. Wireframes and diagrams were created to help guide the development of the application.

The implementation section provides a detailed account of each 2-week sprint that contributed to the development of the project. Tasks completed in each sprint were recorded and this section was helpful in managing the project and deadlines.

The testing section records the different types of testing performed to determine how well the application met the requirements set out from the beginning of the project.



To further develop the project, increased customizability could be implemented. This might include allowing users to control parameters such as chart colours, increments, and sizes, as well as allowing them to select which graphs to be rendered before clicking on a search result. The application has some limitations that could be improved upon. In terms of usability, users may have to familiar with how reddit and it's subreddits work when entering search parameters, so that users receive the search results they require. The charts which make use of time are limited, as reddit posts usually only receive comments for a few days before the post becomes inactive. This means that sentiment cannot be tracked over a longer period of time. Ideally it would be interesting to see change in sentiment on topics over the course of weeks or months.

In conclusion, the writer is pleased with the resulting application. The application meets the requirements and can be used as a tool for users to gain insights into the sentiment of different topics. Technical and soft skills were learned over the course of the project, including python programming, using Flask, working with data, creating visualizations, project management and organizational skills.

## 9. References

- 8 *Applications of Sentiment Analysis*. (2020a, April 9). Monkey Learn Blog. <https://monkeylearn.com/blog/sentiment-analysis-applications/#:~:text=Sentiment%20analysis%20is%20the%20automated>
- Aydin, A. (2023b, October 11). 2— *Stemming & Lemmatization in NLP: Text Preprocessing Techniques*. Medium. <https://ayselaydin.medium.com/2-stemming-lemmatization-in-nlp-text-preprocessing-techniques-adfe4d84ceee>
- Basic text classification | TensorFlow Core*. (2023a). TensorFlow. [https://www.tensorflow.org/tutorials/keras/text\\_classification](https://www.tensorflow.org/tutorials/keras/text_classification)
- Calderon, P. (2018b, March 31). *VADER Sentiment Analysis Explained*. Medium. <https://medium.com/@piocalderon/vader-sentiment-analysis-explained-f1c4f9101cd9>
- Chakravarthy, S. (2020a, July 10). *Tokenization for Natural Language Processing*. Medium. <https://towardsdatascience.com/tokenization-for-natural-language-processing-a179a891bad4>
- Gautam, G., & Yadav, D. (2014b). Sentiment analysis of twitter data using machine learning approaches and semantic analysis. *2014 Seventh International Conference on Contemporary Computing (IC3)*. <https://doi.org/10.1109/ic3.2014.6897213>
- Hasan, A., Moin, S., Karim, A., & Shamsir band, S. (2018a). Machine Learning -Based Sentiment Analysis for Twitter Accounts. *Mathematical and Computational Applications*, 23(1), 11. <https://doi.org/10.3390/mca23010011>
- Jain, A. P., & Dandannavar, P. (2016b, July 1). *Application of machine learning techniques to sentiment analysis*. IEEE Xplore. <https://doi.org/10.1109/ICATCCT.2016.7912076>
- Kanade. (2022a). *What Is a Support Vector Machine? Working, Types, and Examples*. Spiceworks. [https://www.spiceworks.com/tech/big-data/articles/what-is-support-vector-machine/#:~:text=A%20support%20vector%20machine%20\(SVM\)%20is%20a%20machine%20learning%20algorithm](https://www.spiceworks.com/tech/big-data/articles/what-is-support-vector-machine/#:~:text=A%20support%20vector%20machine%20(SVM)%20is%20a%20machine%20learning%20algorithm)
- Kumar, A. (2023b, November 28). *Difference: Binary vs Multiclass vs Multilabel Classification*. Analytics Yogi. <https://vitalflux.com/difference-binary-multi-class-multi-label-classification/#:~:text=Binary%20classification%20involves%20categorizing%20data>
- Leung, C. W. K. (2009a). Sentiment Analysis of Product Reviews. *Encyclopedia of Data Warehousing and Mining, Second Edition*, 1794–1799. <https://doi.org/10.4018/978-1-60566-010-3.ch273>

Liu, B. (2020b). *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. In *Google Books*. Cambridge University Press.  
[https://books.google.ie/books?hl=en&lr=&id=v0UBEAAAQBAJ&oi=fnd&pg=PR11&dq=sentiment+analysis+bing+liu&ots=UgDXyZdlyS&sig=YfkfAxNed1J8\\_rhRh4V3EY86kg&redir\\_esc=y#v=onepage&q=sentiment%20analysis%20bing%20liu&f=false](https://books.google.ie/books?hl=en&lr=&id=v0UBEAAAQBAJ&oi=fnd&pg=PR11&dq=sentiment+analysis+bing+liu&ots=UgDXyZdlyS&sig=YfkfAxNed1J8_rhRh4V3EY86kg&redir_esc=y#v=onepage&q=sentiment%20analysis%20bing%20liu&f=false)

Monkeylearn. (2018a, June 20). *Sentiment Analysis: Nearly Everything You Need to Know* | *MonkeyLearn*. MonkeyLearn.  
<https://monkeylearn.com/sentiment-analysis/>

Mumtaz, D., & Ahuja, B. (2016a). A Lexical Approach for Opinion Mining in Twitter. *International Journal of Education and Management Engineering*, 6(4), 20. <https://www.mecs-press.org/ijeme/ijeme-v6-n4/v6n4-3.html>

N, Y. (2023b, July 23). *Binary Sentiment Classification*. Medium.  
<https://medium.com/@yeshsurya/binary-sentiment-classification-157bce6688f>

Sainger, D. G. (2021a). Sentiment Analysis - An Assessment of Online Public Opinion: A Conceptual Review. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, 12(5), 1881–1887.  
<https://turcomat.org/index.php/turkbilmat/article/view/2266>

SchectmanReporter, J. (2012b, December 7). Obama's Campaign Used Salesforce.com To Gauge Feelings of Core Voters. *Wall Street Journal*.  
<https://www.wsj.com/articles/BL-CIOB-1295>

*Sentiment analysis: From binary to multi-class classification: A pattern-based approach for multi-class sentiment analysis in Twitter* | *IEEE Conference Publication* | *IEEE Xplore*. (n.d.-a). [ieeexplore.ieee.org](http://ieeexplore.ieee.org). Retrieved January 3, 2024, from <https://ieeexplore.ieee.org/document/7511392>

Shah, P. (2020b, November 6). *My Absolute Go-To for Sentiment Analysis — TextBlob*. Medium. <https://towardsdatascience.com/my-absolute-go-to-for-sentiment-analysis-textblob-3ac3a11d524#:~:text=TextBlob%20is%20a%20simple%20library>

Shahul. (2020a, October 9). *Sentiment Analysis in Python: TextBlob vs Vader Sentiment vs Flair vs Building It From Scratch*. Neptune.ai.  
<https://neptune.ai/blog/sentiment-analysis-python-textblob-vs-vader-vs-flair>

Sun, S., Luo, C., & Chen, J. (2017a). A review of natural language processing techniques for opinion mining systems. *Information Fusion*, 36, 10–25.  
<https://doi.org/10.1016/j.inffus.2016.10.004>

Taboada, M. (2016a). Sentiment Analysis: An Overview from Linguistics. *Annual Review of Linguistics*, 2(1), 325–347. <https://doi.org/10.1146/annurev-linguistics-011415-040518>

*What is Sentiment Analysis? - Sentiment Analysis Explained - AWS.* (n.d.-b). Amazon Web Services, Inc. [https://aws.amazon.com/what-is/sentiment-analysis/#:~:text=A%20sentiment%20analysis%20solution%20categorizes\(2023a\)](https://aws.amazon.com/what-is/sentiment-analysis/#:~:text=A%20sentiment%20analysis%20solution%20categorizes(2023a)).

Proquest.com. <https://ebookcentral.proquest.com/lib/iadt-ebooks/detail.action?docID=565922>