**Argumentation and Analysis Based Infographics concerning the 2020 Presidential Debates.**

| **Afzal Miah**  **AC40001 Honours Project**  **BSc (Hons) Applied Computing**  **University of Dundee, 2021**  **Supervisor: Prof. Jacky Visser** |
| --- |

***Abstract* –** *The problem the project aimed to solve was to use argumentation and analysis to create infographics about the 2020 Presidential Debates taking note of the issues modern coverage of politics often have. The Author worked as planned through the requirements stage to implementation to deployment to create both a backend application to generate the visualisations using analysis and a frontend application to act as the interface and to present said visualisations as infographics. A suitable solution was presented, with a backend application that executes sentiment analysis, word level analysis and sentence level analysis to create visualisations. Accompanied with that, is a frontend application which presents the visualisations to readers with accompanying commentary. The author worked diligently throughout the process of the project from background research, requirements gathering and design into development, testing, deployment and evaluation and is presenting in this report, how those processes went.*

# Introduction

This project is to use argumentation and sentiment analysis on the 2020 US Presidential Debates in order to create data visualisations which can be further used to analyse the debates from an argumentation perspective. The problem this project solves is to create an application which can, using natural language processing (NLP) and argumentation techniques, read in the Presidential Debate data and use those techniques in order to generate data visualisations to be used for further unbiased analysis. Considering the current political climate with the emergence of fake news and as narrative has become a prevalent force over objective commentary and facts in media, it has become of increasingly importance to generate unbiased analysis and to present said analysis to the masses; to provide a counter with objectivity and facts to the impact subjective analysis and fake news can have on the general misinformed public. Especially when considering the topic at hand “The 2020 US Presidential Election” and specifically with this project focusing on the “The 2020 Presidential Debates”, the biggest political event in the world since the last US Presidential Election cycle, therefore placing an even higher importance on making sure objective analysis and facts are presented to the readers since considering the impact of such an event, it is imperative that the campaigns and election isn’t rife with disinformation and rather uses unbiased information, an issue this project would help address.

There are clear objectives which the project aims to achieve, other than the issues brought up in the above. The project aims to create a backend Python application, which will be responsible for dealing with the argumentation and analysis aspect that the project aims to achieve, ultimately being responsible for creating the visualisations for the infographics. In tandem with the aforementioned, the project also aims to create a frontend application to act as the interface for users to actually see the generated visualisations and analysis presented in a readable fashion.

Whilst the main objective of this project is to generate argument-based analysis and to present them to readers, this would include other sub objectives which would tie into the 2 main objectives. Sub objectives on the backend would be to use sentiment, word level and sentence level analysis upon the debate dataset and from that produce a suitable data visualisation. The application should generate easy to read data visualisation from which conclusions can be drawn by the readers. The application should be easy to use and navigate for users making sure there is good ease of access to the analysis. A key objective is to make sure that all analysis and visualisations are presented to users in a readable and easy to understand fashion, making sure not to clutter or overwhelm users with a lot of information all at once. A key objective is that the application should be available and accessible on PCs namely laptops, desktops and larger tablets as those devices would have screens large enough to provide a good experience to users.

In terms of end users for the end product, this project aims to reach general users who are interested in US politics but also supporters from both sides liberal and conservative in order to provide them analysis backed by the numbers without any bias or ulterior motives.

Contextually it would be remiss to not consider the current world situation with Covid 19 and all the complications that was brought with the rules and regulations, the key one being the lockdown in Scotland which affected some aspects of development for this project. Due to the societal consequences of Covid, this saw the project have a dramatic shift in its delivery from its original title of being “…infographics for a large multi touch display”, specifically a monolith display to act as the interface for the infographics. Considering the lockdown, the project had to quickly pivot to an alternative way of presenting the infographics in the project now titled just “…infographics…”, it was decided that an application would be used instead as the interface and delivery method for the infographics.

# Background

## The Political Background

The 2020 US Presidential election(which saw Biden win with 306 electoral votes to Trump’s 232 [1]) was the most important political event of the year and arguably in recent memory not only when considering its conventional draw with it being the election of the US, the major superpower of the world, but particularly when considering its polarizing candidates with Donald Trump looking to secure his 2nd term but with this time Joe Biden as his adversary as opposed to Hilary Clinton. Not only are the candidates for this cycle polarizing but the circumstances of the world and the ongoing events contributed to the controversy of this election cycle and its height and coverage around the world.

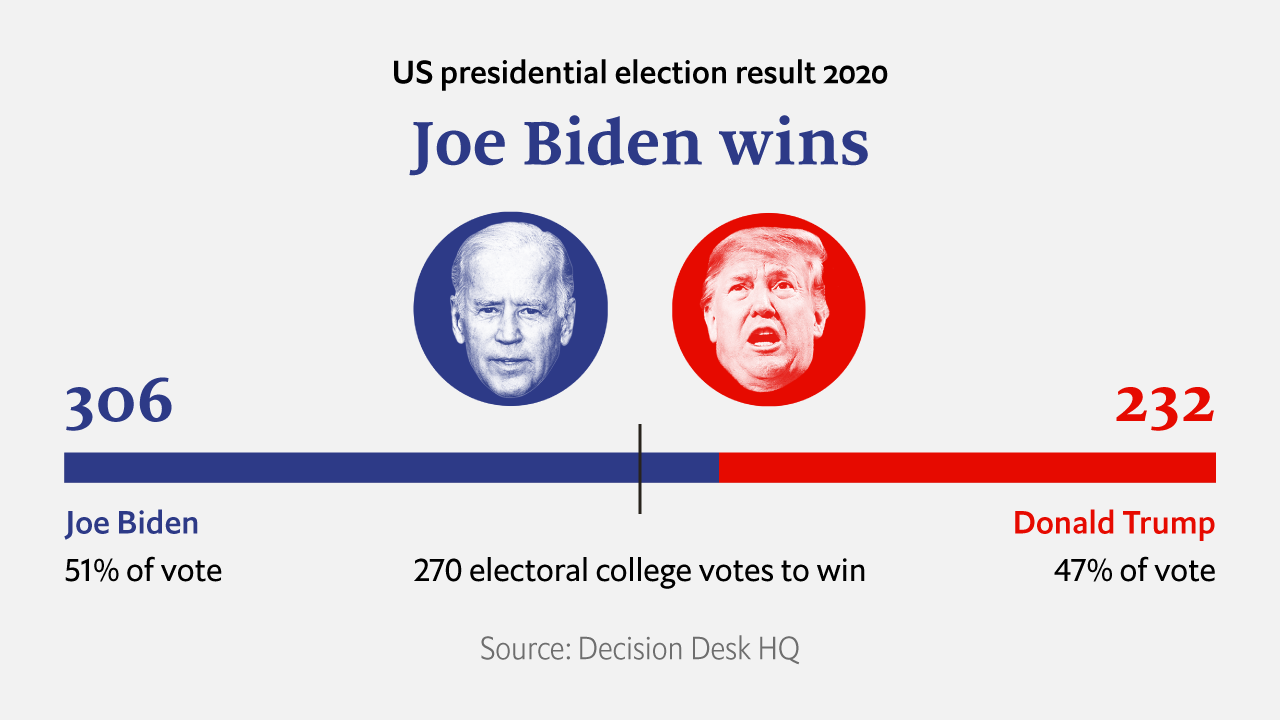


Figure 1 – Election Results (source: [The Economist](https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.economist.com%2Fgraphic-detail%2F2020%2F11%2F03%2Fthe-us-2020-election-results&psig=AOvVaw0Tb-uxr0E0AzMVKvRc_OHT&ust=1619111865047000&source=images&cd=vfe&ved=0CAIQjRxqFwoTCICS6_Drj_ACFQAAAAAdAAAAABAE))

There were many external factors which contributed to the animosity surrounding this election cycle on top of the republican and conservative campaigns. At the height of the election cycle, both candidates had to address the issue of civil unrest amidst the peak of Covid 19 and the worldwide pandemic. This affected Trump’s campaign as it is estimated that Covid-19 cases negatively affected Trump’s vote share particularly in urban counties, in states without stay-at-home orders, in swing states, and in states that Trump won in 2016. This is important to note, as it is suggested by counterfactual analysis that Trump likely would have been re-elected if the case numbers were just 5% lower. Contrary on the other side it is suggested that Covid-19 had a positive effect on voter mobilisation, particularly for those in Biden’s base helping him win the Presidency. [2]

The US economy took a big hit due to Covid 19 which was destructive enough by itself but it was only amplified striking during the election year, shifting the political narrative and Trump’s re-election campaign; Trump’s pandemic response was criticized by epidemiologists and scientists, in stark contrast to many of the others responses in the west. [2]

Compounding to the pandemic, the US saw civil unrest due to the story of George Floyd where on May 25th, Minneapolis police officers arrested George Floyd, a 46 year old black man [3]. Where video was caught of police officer Derek Chauvin restraining George, kneeling on his neck restraining his breathing as George pleaded for his life before becoming non-responsive and was later pronounced dead approximately an hour later [4]. Following the release of video from bystanders and cameras, the US erupted into civil unrest with protests and riots which continued throughout the election year, being a major point of contention amongst voters. Naturally per due process, there was conflict simply due to opposing views on political affairs and policy which are always relevant in election races, which was conflated by Trump’s aggressive campaigning with his claims of election fraud and efforts to pressure election officials which culminated in the post-election Capitol Riots, where voters stormed the Capitol in protest of Biden winning the election resulting in deaths and following procedures viz. one article of impeachment against Trump for the incitement of insurrection. [5]

## The Presidential Debates

The US Presidential debates are considered to be the marquee political event for the election and this year’s saw what many around the world saw as unfitting of a World Leader [6]. In such a debate, the one who would emerge the winner would be the one least tainted by what was the political equivalent of a food fight, which going by instant polls and betting markets was Joe Biden; Only off the back of being able to prove that he could hold up under pressure and had not lost a step in his advancing age. This was a debate however wrought by constant interruption, where Trump rarely gave the former vice president Biden a chance to say something damaging to his campaign. With Trump needing to use the debate as an opportunity to shake up the race, which was by most metrics tilting against him, Trump turned the Debate into a chaotic free for all, resulting in a debate which likely have little to no likeliness to actually alter the dynamics of the race or the minds of undecided voters. However, this reversed in the second debate which saw the candidates settle in to more thoughtful and calculated responses and debate.



Figure 2 – Trump and Biden pictured during the first presidential debate (source: [BBC](https://ichef.bbci.co.uk/news/640/cpsprodpb/E505/production/_114692685_uspresidentialdebate2020timedonaldtrumpandjoebiden.jpg))

The first debate in particular was memorable where it was clear from the onset the type of debate it would be, one where Trump would try to rattle his opponent with constant interruptions and chaotic exchanges. The debate format saw race issues and urban violence lumped together leading to exchanges where it was clear who was more comfortable about addressing such topics with Biden being more comfortable addressing the former and Trump the latter, with Trump refusing to renounce right wing violence though he said he would and instead telling one group when asked about them, the Proud Boys, to “stand back and stand by”. Other key points of interest where the discussion of racial division, criminal legislation, covid policy, foreign policy, the economy, the green new deal alluding to conflict in the Democrat party and taxes. [7]

In stark contrast to the first, the second debate saw a more controlled debate with the candidates being more restrained, with the addition of a mute button or the threat of it leading to the debate being more civilised and less chaotic. Both candidates were respectful in tone, allowed each other to speak and in the cases when one went on the offensive, they did so in a calm and deliberate manner. Whilst the first debate would be one more remembered for the presentation of the debate, the second would likely to be more for its content. Covid was the key point of interest for this debate, the topic the public was most interested in, with Biden going on the offensive challenging Trump on his response as President. Other key topics were Trump and Bidens exchange about Biden’s son Hunter Biden, an argument surrounding immigration and its policies and the issue of criminal justice with context to race relations. [8]

## Argument Technology

Argumentation and its analysis covers a range of different aspects of argument analysis however for this project the focus will be on Natural Language Processing (NLP) and analysis at the sentiment, sentence and word level. NLP is considered to be one aspect of argument technology and is the area of research and the application of technology which explores how computers can be used in the understanding and manipulation of language text or speech in order to further execute useful operations. The main goal of NLP research is to use technology in order to create a greater understanding of how people use language so that appropriate tools and techniques can be further developed in order to provide computer systems with the capacity to understand and manipulate natural language in order to carry out the desired tasks.

The foundations of NLP are prevalent in a range of different disciplines, as its impact can be seen in a large number of applications in: computer and information sciences, linguistics,

mathematics, electrical and electronic engineering, artificial intelligence and robotics and psychology to mention a few. In terms of the type of applications NLP is used in, that also covers a range of topics, viz. machine

translation, natural language text processing and summarization, user interfaces, multilingual and

cross language information retrieval (CLIR), speech recognition, artificial intelligence and expert

systems being a number of the more ubiquitous applications.

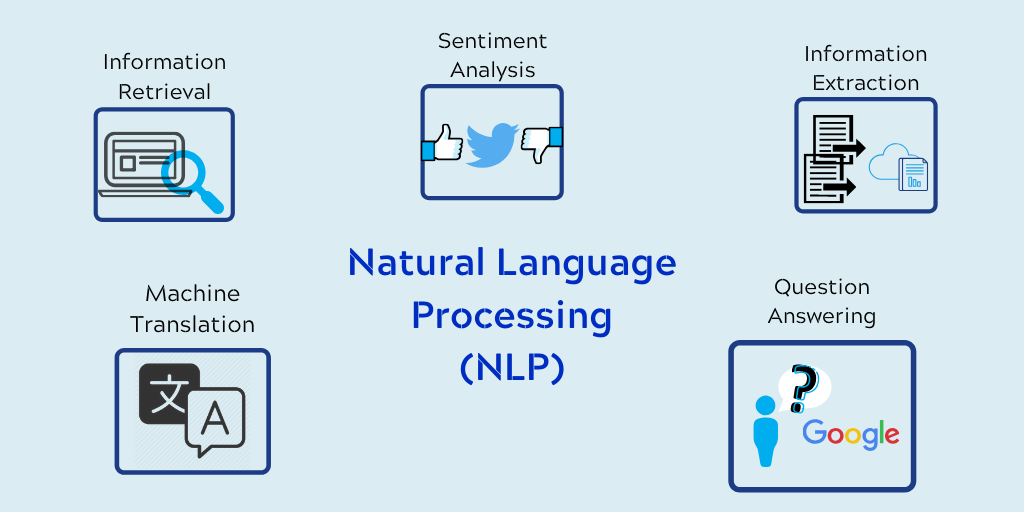


Figure 3 – The most widespread uses if NLP (source: [Cybiant](https://www.cybiant.com/resources/natural-language-processing/?v=79cba1185463))

The emergence of NLP in its encroaching prominence in research and industry can be tied to the evolution in technology, specifically the proliferation of the internet and the world wide web and digital libraries. With the internet being host to all the languages in the world, researchers have placed an importance on there being appropriate research in NLP to help the breaking of the language barriers across the internets to facilitate the use of cross-lingual research and information retrieval allowing of full use of the internet on a worldwide scale by removing the limitations of language and linguistics.

Every instance of NLP research concerns with the issue of natural language understanding, such that the building of any NLP applications leads into three issues: the thought process of the application, the representation and meaning of the linguistic input and finally to consider the wider general knowledge on NLP. Leading in to how NLP systems work differently at different levels for example at the word level NLP can be used to determine specific things such as the nature of the word or its morphological structure however when moving onto the sentence level those same things may not need be determined and instead other aspects of processing such as word order, meaning or grammar may need determining with this trend of different processes and different determinations continuing as you follow through to the document level and so on. Therefore in the pursuit of natural language processing, the ability to distinguish among the seven levels is important: [9]

* Phonetic or phonological level
* Morphological level
* Lexical level
* Syntactic level
* Semantic level
* Discourse level
* Pragmatic level

Sentiment analysis (or opinion mining) is a NLP technique and is a form of shallow semantic analysis of texts which is concerned with the extraction of opinions, emotions or attitudes regarding entities of interest. As discussed earlier with how the proliferation of the world wide web has placed an importance on NLP, one specific technique where this is true is sentiment analysis. Considering the rise of social media and general consumerism, sentiment has emerged as a key field of NLP and general research as independent parties strive to mine opinions in order to use the results to further pursue their targets. Sentiment analysis is an intriguing method of NLP due to the nature of how powerful the data mined from sentiment can be and the range of sentiments which can be mined; Sentiment allows for companies to analyse and identify public sentiment towards them allowing them to use that information to respond accordingly, this is one example of its use and extrapolates to how it is so important in many fields. Considering sentiment analysis also covers a range of industries, entities strive to mine sentiment for data for use in retail to politics to finance, further compounding its importance on research. [10].

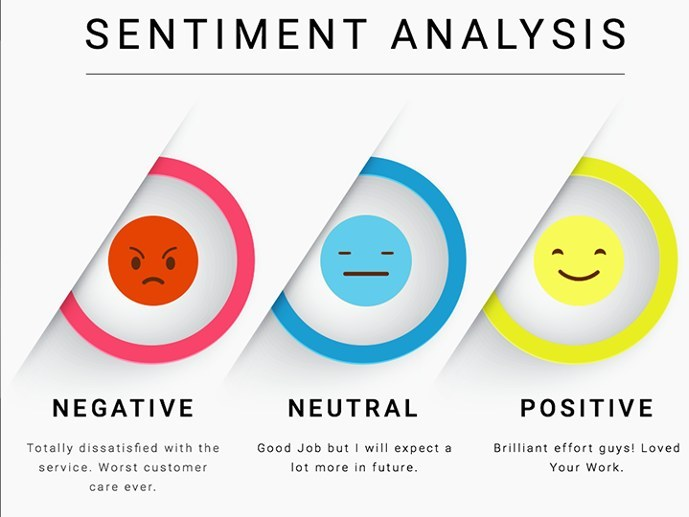


Figure 4 – Example of 3 Classes of sentiment (source: [KDnuggets](https://www.kdnuggets.com/2018/03/5-things-sentiment-analysis-classification.html))

## Infographics

An infographic as defined in the Oxford English dictionary is a visual representation of information or data but in practice the meaning of an infographic can be more specific; An Infographic is a collection of imagery, charts and text that gives an overview over a topic. They are considered to be a valuable tool for visual communication and their goal is not to just attract attention but also to help readers understand the content of it. They are used in many instances for providing an overview of something, explaining a process, display research findings or data, creating summaries, running comparisons and raising awareness for a cause. For the context of this project the relevant cause is to present research findings and data.

## Existing Solutions

In terms of existing solutions, there are some but in a more obscure sense. Most of the existing solution which follow the same general target of creating argument technology based analysis and visualisations surrounding the presidential debates are done in a non-mainstream sphere in various data science blogs and journals away from the eyes of the general uninformed user. In terms of existing solutions in modern and typical media, there isn’t really much argument technology analysis to be found or indeed much data-based analysis period other than the typical analysis of polling numbers, projections and voting statistics.

## Commercial Viability

There is also the commercial viability of the application of the application to assess as well; Whether the application can be monetized or not and what avenues can be taken if choosing to commercialise the application for profit. There are a number of things to consider when it comes to the monetisation of the application, whether the application provides enough value to justify charging readers for access. A key issue this project aims to assess is the lack of access, or simply lack of unbiased analysis so monetizing the application in some sense defeats the whole purpose for creating it. However, there are ways of monetisation which are not invasive to the access of the product. Typically, there are a couple of ways to monetize a project such as this. One way is to lock the content behind a paywall, however again this defeats the purpose of easy access. Then that leaves on other route, where as often is the case with web applications, the best method of monetisation is the use of ad platforms to convert site traffic into revenue. Another obscure method of monetisation could be to monetise one part of the project such as licencing out the backend application which carries out the analysis and visualisation generation to a willing buyer.

## Legal and Ethical Implications

It is important to ensure that the application also takes into account any legal or ethical implications in development and deployment. There are many standard laws which the application much follow, one example being the Data Protection Act which the application must adhere to. The Data protection act is the UK’s implementation of the General Data Protection Regulation (GDPR) and the basis of the act is everyone responsible for using any personal data must follow strict rules called the ‘data protection principles’; The data used must be used fairly, for explicit purposes, only for what’s necessary, be accurate and up to date, handled with good security and for no longer than necessary. One thing to note is that there is a stronger legal protection for the more sensitive data [11]. But this is only relevant for instances where user data is needed, in the case of applications which don’t store or use any sensitive data whatsoever, by default it would meet the standards of the act. Choosing not to use any user data creates a compounding effect to the development of the application as though adopting to avoid any sensitive data may lead to the loss of some potential features; Alternatively, this leads into there being more time in implementation for other important features as due to there being no sensitive data being stored, more emphasis in development can be placed on the implementation of the application rather than the security of any sensitive data.

The other legal consideration would be the Copyright Patents and Designs Act; There is an importance in ensuring that any intellectual property used is either the property of the developer or rather in the case that the intellectual property isn’t then ensuring it is used with the permission of the Copyright holder. The Copyright, Designs and Patents Act gives the owner the exclusive right to copy, adapt, communicate, lend or sell copies of the work, although this right can be sold or transferred [12]. Relatedly, The Intellectual Property Act similarly should be considered; it aims to modernise the copyright law where its most significant changes relate to patents and design rights and is law specific to the expression of idea (given Copyright is law specific to the expression of ideas in form). [13]

As far as ethical concerns, there is a responsibility to ensure that the application is accurate and unbiased considering the subject matter at hand of politics. Spreading misinformation and “fake news” is damaging to society due to the nature how it is easier to spread disinformation than it is to renounce said disinformation with information therefore care must be taken ensuring that the information and analysis shared by the application doesn’t contribute to the above sentiment.

## Deployment

The Deployment and the Framework to be used for the project is another key thing to review in terms of the scientific and technical aspect of the project. There are a variety of different frameworks which can be used to build the application and likewise there are many ways to deploy the application their own concerns and benefits therefore it is important to ensure that sufficient research is done to review them in order to ultimately decide on a combination which would best fit the approach of this project. In terms of the frontend side of the project which in this case is concerned with presenting and displaying the analysis and visualisations, it is a key decision on deciding on the framework to use for development. The main frameworks in consideration for use were Vue, Angular, React and native HTML; frameworks widely used in industry and of which there is an abundance of documentation. React and Vue are JavaScript frameworks, Angular a typescript framework and of course then you have just native HTML with CSS styling. React and Vue can be considered to be very similar to each other as frameworks. React in its docs interestingly although many consider it a framework, defines itself as more of a library however since the difference between a framework and a library is often subtle, this is often open to interpretation.

## Frameworks

In terms of the backend development web development frameworks, the frameworks in consideration were Node.js, Flask and Django. Node.js is a cross platform, open source, backend, JavaScript runtime environment which is responsible for the execution of JavaScript code and does so outside of a web browser. Node.js in practice is an event driven framework and a key feature it has is that is uses a non-blocking I/O model which allows for Node.js to be a lightweight and efficient model which is why it is considered to be the premiere environment for creating data-intensive real time applications particularly in the instances where applications run across distributed devices. Another key feature of Node.js is that it allows access to a large library of JavaScript modules allowing the development process of application to be simplified to a certain manner. The other key features of Node.js are that it is an Asynchronous framework and Event Driven, executes code fast, is single threaded and highly scalable and doesn’t require the use data buffering. The only negative consideration to make for Node.js is that it isn’t suitable for use in CPU intensive applications [15]. Django on the other hand is a high-level Python based framework which encourages the use of rapid development and pragmatic design. It is a fast framework, with the philosophy of allowing developers to help take applications from concept to completion whilst also placing an emphasis on being secure and highly scalable [16]. Flask is a lightweight Web Server Gateway Interface application framework designed with good scalability with the features of offering suggestions but not enforcing any dependencies or project layout, leaving it wholly up to the developer to choose what tools and libraries they’d like to use. Another benefit of flask is that there are many available extensions provided by the community that can make adding new functionality easy. [17]

## Databases

Since there may be a need to store data in this project, another key thing to research and review would be suitable database servers/tools to use for the storage. The two main ones to be reviewed for this are mongoDB and MySQL for use in a database server. MongoDB is a no-SQL database which has a focus on three key aspects when it comes to development: Availability, Scalability and Faster Development. Something important to remember about mongoDB is that is uses JSON-like objects to store any data. In terms of failure detection, mongoDB has fast failover detection giving it the ability to natively detect failures whilst also having the feature of tuneable consistency guarantees. For the development process, having the data as flexible JSON documents instead of rows and columns like traditional databases allows for the developers to rapidly implement the database, due to how JSON documents are more flexible, natural and allow the applications to be faster and more responsive. In terms of scalability, mongoDB is cheap to scale applications with the added benefit of not requiring the need to make changes for the scaling process [18]. Contrary to mongoDB databases, MySQL databases are relational and based upon the uses of queries and tables; MySQL is secure, as it follows the working of a Client/Server architecture and is also scalable with high speeds and flexibility, with also good memory efficiency, and good compatibility. [19]

## Source Control

A key but often overlooked aspect to research for the project would be source control for the implementation and development cycle of the project. In terms of source control, although there are a number of solutions, GitHub (a Git repository hosting service with the addition of many of its own features) dominates amongst general users. Git is a command line-based source control interface, GitHub expands on that by integrating with Git to provide a web-based interface which is easy to understand and use but also adds access control features and ease of installation/use which would be key for a project when working with multiple devices. [20]

## Programming Environment

Finally, an important consideration to make is the framework to use for the development of the backend argument technology analysis and visualisation. Python is considered to be the premiere language for data analysis on the industry for a number of reasons: it is open-source with there being widely available documentation, it is easy to use and understand having been in use since the early 90s and also it is a general purpose and dynamic language which is inherently object oriented which supports the use of multiple paradigms, in function, structures or procedural programming. With regards to data science, due to the packages available in Python, Natural data processing and data learning becomes intuitive making it easy to extract, analyse and visualise data in Python. Scala is a more recent solution for data science and is a modern and considered to be elegant programming language. Scala is scalable and most effective for use in handling big data problems. Scala also supports object oriented and functional programming in concurrent and synchronized processing. R is an open-source language and software specifically built by statisticians for the use of statistical computing and graphics. Although it has many available libraries for use in data science and is good for conducting ad hoc analysis, it is a more complex language to learn than the others for the same use case. [21]

# Specification

In terms of the specification of the project, there wasn’t a specific brief provided by a client and rather the specification was to be extrapolated from the given title of the project “Argumentation and Analysis Based Infographics concerning the 2020 Presidential Debates.” This is quite a packed title as when it comes to the specification to be derived from this title, there are many different considerations to made. The project can be broken down into two parts, the “Argumentation and analysis…” and the “…infographics…”. Taking into account both parts and further breaking them down to form a specification, the key aim of the project is to apply argument technology to the presidential debates in order to generate analysis and visualisations and to then present it in some form of infographics to readers.

The title can be broken down into the following brief:

* There should be no need for any accounts or feature which requires the use of protected data due to the issue of data privacy.
* The application should be able to read in the debate data and extract the necessary information.
* The development should be made with laptops and desktops as the target device. This specification in particular rose due to the complications of Covid on the project, requiring to swiftly change the end device from a large touch monolith display to an alternative. Ideally the pivot on the end device would have been any equipped web device however due to how suddenly the change was needed (which came in the middle of the development cycle), the timing meant that this could not be satisfied and thus a focus was placed on larger pc devices although it would still work in theory for smaller devices it would not be at all tested or optimised for mobile devices due to the time constraints as mentioned in the above, this would not have been an issue had it not been for covid nor if this change were to be made at the beginning of the development cycle.
* The application should be able to read in the presidential debate dataset and extract the text and any other important information.
* The application should be able to generate analysis and consequent visualisations at the word level on the dataset.
* The application should be able to generate analysis and consequent visualisations at the sentence level on the dataset.
* The application should be able to generate analysis and consequent visualisations regarding the sentiment of the dataset.
* The application should be available to access via the internet.
* The generated analysis and visualisations should be presented in an application for readers.
* There should be accompanying commentary explaining the analysis and visualisations.
* The analysis and commentary should be free of any bias and not misleading with regards to journalistic integrity.
* The visualisations and application should be readable to any demographic and suitable for its intended use case of providing analysis.

The project was first started by breaking the brief into specific parts; The backend analysis and visualisation part and the front end application and interface part. Following this, the projects different points on the brief were all given a priority knowing some points were more important than others and with it being unknown at the start of development whether everything would be achievable or not. It was decided that it was of highest priority to first create an application to take in the debate and perform argument technology techniques on it in order to create the desired analysis and visualisation, specifically having there be analysis at the word, sentence and sentiment level. After which the priority would then shift to having a front-end application to act as the interface for readers. There would be a lower priority given to making the front-end interface aesthetically pleasing, having the visualisations be interactive for each analysis case and having commentary on the front end interface. These priorities remained consistent throughout the project, with the higher priorities being expected to be achieved and the lesser ones to be achieved to some extent but potentially not fully.

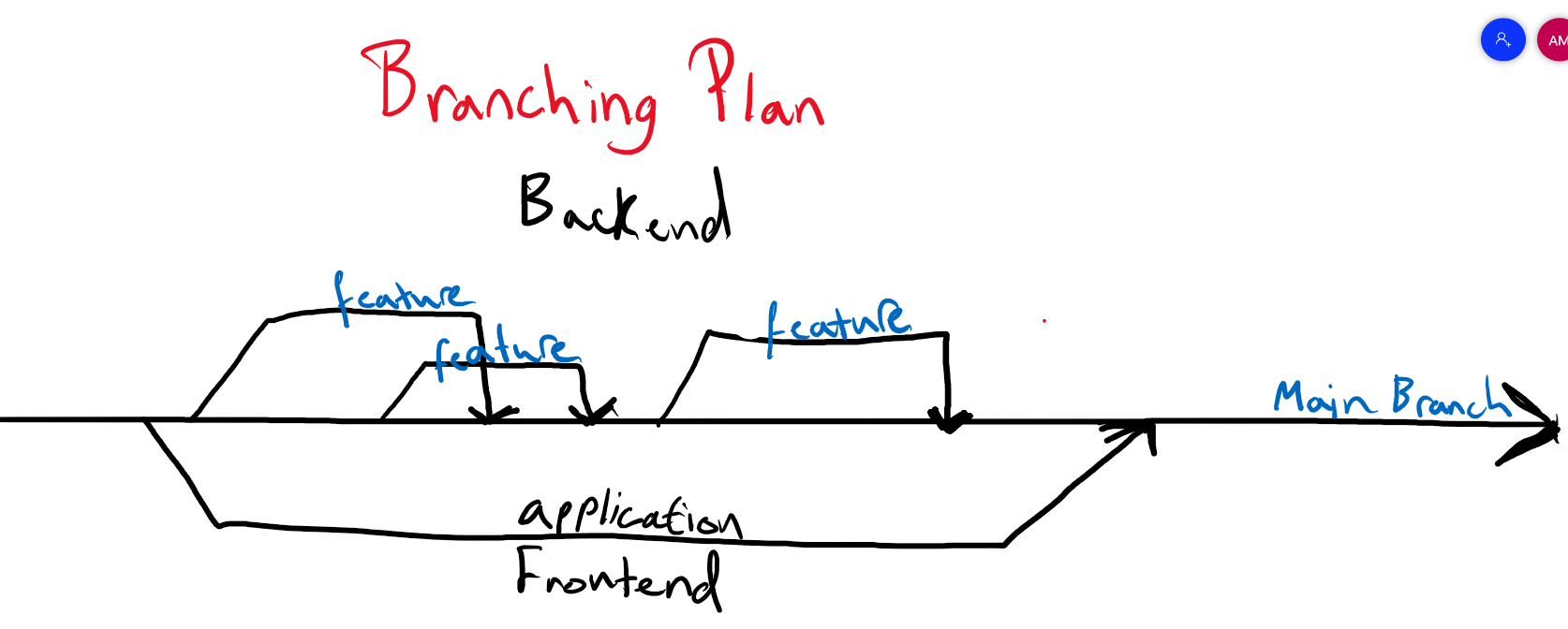
It was then researched, and contemplation was made on how the requirements would be fulfilled. The first requirements to be fulfilled were ones to do with the back argument technology. In order to carry out argument technology techniques, there was first a need to have a suitable dataset to do so on. Therefore, it is of high importance to find or generate a dataset on the presidential debate in order to complete the project in any capacity. The overall project was to be split up into two separate applications which would carry different things and have different goals. One application would be responsible for the analysis and generation of the visualisation, the other application would be responsible for the presentation of the analysis and visualisations. In terms of the analysis, for sentiment the key analysis to be done to the dataset would be polarity and subjectivity analysis. At the word level, analysis would be done in the form of word counts and word bubbles. At the sentence level analysis would be done in terms of statistical analysis and analysing the heat of the debate. Charts, histograms, and graphs would be the visualisations of choice for presenting the analysis done to the dataset. These would then be saved in image form to allow them to be presented and used on the front-end application side. In terms of being available on the web, this was the default choice of development as creating standalone applications for each of the major platforms would not have been attainable in the given time period. However it is important to note that having it be a web application has the disadvantage of there being a need to have internet in order to access the information, however the advantage is that there is ease of access in that there aren’t any hardware of software limitations to access the application and also for content such as political analysis it is of standard practice anyway to have it be accessible online so the disadvantage of requiring an internet connection can be considered to be redundant especially in a seemingly connected world where there is internet access almost everywhere. Considering the above, there would be a need to have the content hosted somewhere to allow for this access.

During the initial planning stage, a decision was made to adapt a waterfall approach of development for this project, meaning following the sequential and linear process of project management starting with requirements gathering to design to implementation to testing to deployment to maintenance. This meant that no phase would begin until a prior phase was complete, this meant ensuring that there was sufficient documentation for planning especially concerning the project’s requirements. To track the progress of the project throughout its life in all stages, a GANTT chart with the key points of development was used to be referenced throughout the development of the project to ensure when internal deadlines were so they could be met. Alongside this a document of all the functional requirements was created to be used as the baseline for all development for the project. [22]

During the initial requirement gathering and documentation stage, first the functional requirements were set. Once this was set more general research was carried out regarding argument technology with regards to analysis and also data visualisation. The setup of the project environment was also done in this stage, this required the installation of the chosen source control software, and relevant IDEs for both the front and backend in with the installation of GitHub and Jetbrains developer tools and required extensions.

Leading into the System design phase, this was split up into 2 sections for the backend and frontend. Using the established requirements, designs were made for both parts of the project. Starting with the backend designs first, planning was done first for the file system layout of the backend. Following into the design and planning out functions and the general code with pseudocode. In terms of the frontend design phase there were a number of things to design. First the UX design for the application was done, after which the design of the routes required for routing between pages was done. Finally, there was some thought given to the design of the repository structure and a branching plan was created.

Figure 5 – Branching Plan



The implementation phase saw the implementation of the entire project with the backend and frontend with close regards to the designs and requirements. Starting initially with creating a boilerplate using the create react app tutorial for the front end before then moving onto to fully focusing on the backend development of the argument technology and analysis capping off the implementation phase with the development of the front end interface.

Testing was then done to ensure that the code ran as expected, to ensure that the application on the front end ran smoothly and to ensure that the backend managed to carry out and generate the required analysis and visualisations. Finally, all that was left was to move on to the deployment phase of the project by which the front end was deployed onto Firebase and the file system was prepped and managed for submission. Maintenance is not a required phase as there is no client per say to deliver the project to not is it being actively used, however it can be said that maintenance was done via various tweaks and fixes which were during and after the write-up of this project.

# Design

In terms of the design for the front-end, the decision was made to create design prototypes in Adobe XD. The first part of this process was to undergo research on existing solutions concerning argument technology analysis, data visualisation and also infographics in general in order to see any design considerations existing solution use and whether there are any notable design trends which could be applied to the design of this application. After taking into consideration the research, the candidate moved onto the wireframing phase of design for the front-end application.

In terms of the wireframing phase of the design prototyping, first a decision had to be made concerning the network of pages required for the application. To determine this, the candidate designed out the general flow of the application in its navigation starting from the home page following onto the rest of the application. This led to knowing what and how many pages needed designing, in the homepage, analysis landing page, sentiment analysis, word analysis, sentence analysis and heatmap page. Having these as the main pages to design, these were wire-framed as low-fidelity prototypes. One of the big design decisions made were to have a homebar with a menu and home button which would allow for users to go to every page on the application from any one place on the application. With the pages to be designed decided on, it was important to ensure that the design was suitable for its intended platform of laptops and desktops. Following the completion of the wireframes, a quick review of the designs was done to ensure there was design consistency.

Following the completion of the low fidelity prototypes, the next step of the prototyping and design process was to create higher fidelity artifacts, with the intention of having the implementation phase of the project follow as closely as possible. A major design decision which was made concerning the front-end designs was that the prototypes and by extension the implementation was to be done with material design in mind. Material design defines the qualities that is expressed by the UI, surfaces, and components of design. There is a number of material design principles and methodologies that were adopted in the design of the application. Layouts in the design should be consistent and predictable with the use of intuitive and predictable layouts, care was taken in choosing the navigation methods to be used with the 3 types of navigation of lateral, forward, and reverse navigation. Ultimately material design was intended and used to create hierarchy, meaning and focus to the design where material design components were as building blocks for creating the user interface, of which are standard components made by contributors to the project designed with the intention of creating an immersive experience. Theming was also a key thing considered in the design of the application with there being a conscious decision being made on the theme of the design with regards to the colour, typography, and component shape of the designs.[23]

The author also deliberated on how to design the presentation of the visualisation in the project. There are 2 avenues to design how the visualisation would be presented, either as static images or interactive graphs. After much deliberation the design decision to make use of images instead of dynamic graphs was made as although having dynamic graphs would provide better interactivity to the design, the trade-off of requiring the need to process data and execute the analysis and visualisation on the application wasn’t worth the resource requirements for diminishing returns when an image is fine for displaying a visualisation and aesthetically is no difference.

There was also design on what approach of sentiment analysis was going to be used for this project, either machine learning approach or a lexicon-based approach. The design decision was to use the lexicon-based approach is its corpus based approach and the libraries which adopt that were sufficient for the level of analysis this project hoped to generate. (see figure 6)

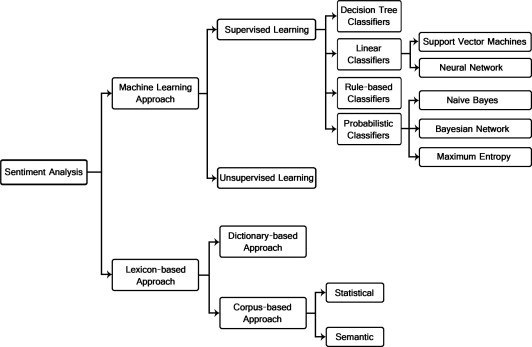


Figure 6 – Sentiment analysis approaches

There was design consideration to be made concerning the project management; in terms of the development process and in terms of the development tools to be used. When it came to the development process, as discussed previously, a waterfall approach was used to development considering the term and size and scope of the project with the aim to do development with close regard to documentation and the requirements. Github was used as the repository and Tom’s Planner was used to create the GANTT chart which worked as the greater project plan and overview as both were simple and easy to use, whilst also having tools which would make the development process of the project easier whilst also making it simpler to track and monitor the progress of the progress against the project plan.

In terms of the design considerations for the front-end application, the environment to be used as the service for the application was Node.js with Javascript as the language but mainly in the use of React.js for the interfacing. This decision was made due to how the candidate was well versed in Javascript (albeit not react and node specifically) as well as Javescript with node.js and react.js being the industry standard language and frameworks. Compounding onto the experience benefit of working with Javascript with node.js and react.js, they both host a suite of features which would be helpful in the implementation of this project, such as the available packages and libraries (previously discussed in background). The alternatives to this were Flask and Django and although those were also viable options, in the end Node.js was chosen due the edge of how lightweight and easy to pickup it is.

In terms of the specifics surrounding the choice of Node.js, Node.js is a cross-platform runtime environment which is built on an open source and high-performance engine of Javascript. Node.js further ensures that this is the case by employing the feature of an event-driven, non-blocking I/O paradigm. It is considered an easy to learn framework requiring only some base knowledge of Javascript and object-oriented programming to get an understanding of it. In terms of the difficulties surrounding it, it can be considered more difficult to grasp its concept of its Client-server model and also to understand its asynchronous flow. Node.js has a fast time to marker due to how its so lightweight and scalable, which is achieved by load balancing. Node.js supports scaling in both a vertical or horizontal approach with also having the benefit of being a framework with great community support with widely available and useful tools to help aid development, the key one being NPM which is a node package manager with a range of libraries and templates which can be used to help speed up the development process. NPM was key for the development of this project as it was an essential tool for creating development environments and production builds. [39]

The back-end Node server was setup using Webstorms default automated settings simplifying the running of the webserver in development and creation of routes for the development team.

To start with backend design, a document laying out the design of the project file hierarchy was created. After which a UML diagram was created to layout the design of the application. A flowchart was made to design out the overall flow of the application with also there being pseudocode created to plan out specific parts of the code such as the read in data function.

For the UX design, Adobe XD was used to create the low and high-fidelity prototypes. This was chosen due to the researcher previously having experience with interface design in XD. During the research process preceding the actual making of the prototypes, inspiration of the types of visualisations, layout and design were pulled from existing applications of argument technology and notebooks.

With the requirement of ensuring that the application is readable for a range of demographics, a conscious design decision was made to not use heavy styling methods such as animations, videos, gifs and large images opting to instead go for a more minimalistic design as to not over complicate the application with unnecessary filler and to ensure that the reader’s focus remains solely on the data visualisations and analysis. In terms of the colour palette, typography and layout, making sure to adhere to material design principles, the standard default material ui theme was chosen for the applications design. Originally the plan was to use a bootstraps library for the components and design with a custom theme, however due to limitations in development and with how much more intuitive material ui was to implement, the shift to material ui was made.

Leading onto the actual style implementation of the application, there was minimal use of css with the bulk of the style implementation done using material ui libraries and attributes. Syntactically using material ui is just using Javascript but specifically material ui’s packages and libraries therefore there was no issues occurred in the programming of the front-end in terms of compiling as everything was done in Javascript.

In reflection the design process of the application and its implementation was rewarding to the researcher in learning new skills and broadening their scope of development. It wasn’t a perfectly smooth process, often requiring some sort of trial and error in some cases nevertheless the final product works and looks as initially planned.

# Implementation and Testing

During the initial prepping phase of implementation, the researcher took special care on making judgments regarding what tools, programming languages and libraries would be used for the implementation process of the project based on the factors of previous experience, industry standards and useful features to the brief. Kite was a tool used across the development of both applications. Kite is an AI-powered programming assistant which helps with the writing of code, with AI powered code completion more advanced than the typical IDE’s. Kite was able to integrate with the Jetbrains suite being used for development with Webstorm for the front-end application and Pycharm for the backend application. [25]

## Front End Application

For the front-end application, React was chosen which is a Javascript library for building user interfaces. There was a number of reasons as to why React was chosen: React has readily available and extensive documentation and a large amount of training and learning resources lending it to be fast and efficient to learn. A key concept of React is its use of components, and with the use of components objects, reusable code which can be used anywhere on the interface can be made. The benefit being that having reusable components makes the application easier to maintain and develop and makes the task of future updates easier to handle. For React, an important thing to consider is the virtual DOM; the virtual DOM is one key reason as to why the researcher chose React for implementation as it allows for any changes to the code to be first performed on the virtual DOM which lives in memory. JSX is used to write code, JSX being a syntax extension of Javascript, which in essence is a mixture of Javascript and HTML which makes the code easier to understand and simpler not only to read but also to implement.

In terms of the chosen IDE, Webstorm was chosen due to its host of features. Webstorm js a modern Javascript ecosystem with the main features of intelligent code, error detection and support for the popular frameworks, the relevant ones being for this project, React and Node.js. It comes with a built-in debugger which allowed for ease of debugging with the use of breakpoints. It had good tool integration, with linters, build tools, test runners, REST client and other tools being available and integrated with it. A big feature was also its integration with VCS as it allows for the use of a simplified unified UI to work with GitHub, complete with built in tools for committing files, reviewing changes, and resolving conflicts. [26]

The initial development environment setup was done by using Webstorm’s CRA (create-react-act) which generates a boiler plated application with included browser refresh on code change, a feature which greatly help lower the development time and made the process more enjoyable as it removed the need to manually do so one every code change. Webstorm’s CRA also handled the production build of the application as it allowed for the project to be built from the command line with a single command with the use of NPM. In order to endure that the versions of code on the developers different working devices were up to data, each device had to have bash installed which allowed the use of NPM for package installation, version management and dependency management, however this also came native in WebStorm as well so in the end there was some redundancy there.

There were some preceding procedures done in the process of development. Prior to beginning implementation, the author would research and watch related tutorials on YouTube followed by implementing a demo based on the watched tutorial to allow the researcher to have a greater grasp on key features, this was then followed by the author following the React demo and implementing it themselves to invoke a greater understanding of React and how React renders an application. In the actual implementation of the application, the programmer would have a rough checklist of the sorts of things which were needed to be done which would then be marked off as they were implemented.

A number of components and pages were created for the application with the work being done according to the priority given to the requirements. The author had used React sparingly before but the learning curve of getting familiar with it again was rather smooth with the added benefit of in-depth research time contributing greatly.

One thing which helped greatly in the implementation was having good and clear interface prototypes laid out which removed the potential for any guesswork or improvisation. The structure of the application, of each page and the functionality of everything was planned during the design and prototyping stage which meant all the focus was purely on ensuring the implementation was as close to the prototypes as possible instead of there being wasted time on tweaking any design issues during the development process.

At the beginning of the implementation process for the front-end application, deadlines were set for specific parts of the program to be completed such to monitor progress and stay within reach of the final deadline, leaving enough time for the write-up. A key to meeting deadlines in this project was to be sincere within themselves acknowledging any issues and facing them head on as opposed to delaying the inevitable as is this sets the standard for the developer to follow throughout the process.

During weekly progress reports with the attending advisor and other independent researchers, any concerns or challenges encountered were brought up and discussed and solutions were brought up where possible.

Due to how React is vastly documented, when issues or challenges were encountered, there was a abundance of tutorials and articles available for reading to help solve them.

Concerning the components and views of the application, they each has their own css file although they were very minimal in use and code was done following the design decision to use material ui (see figure 8). This was one instance were having a high-fidelity prototype as a reference greatly helped and was useful for ease and speed of development. Throughout the development process, careful consideration to maintain design consistency across the application was made. With React as the building block and foundation for the project, attention was made to setup the React router to ensure the application navigated as intended page to page. (see figure 7)

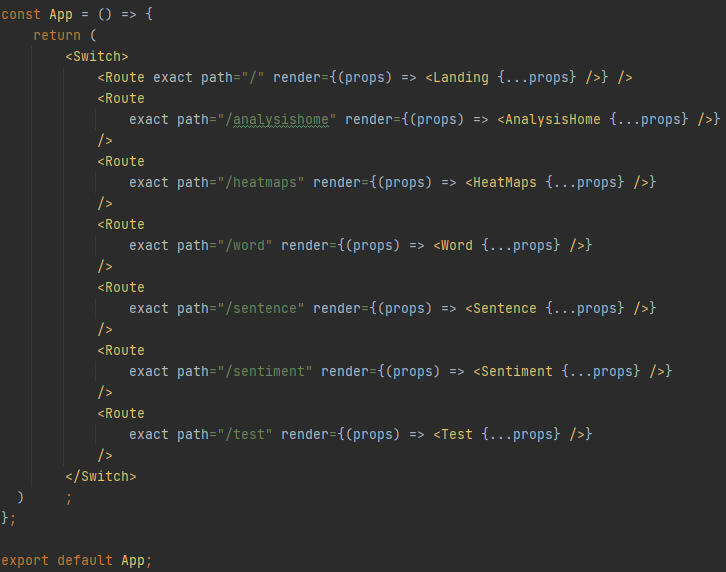


Figure 7 – Router setup

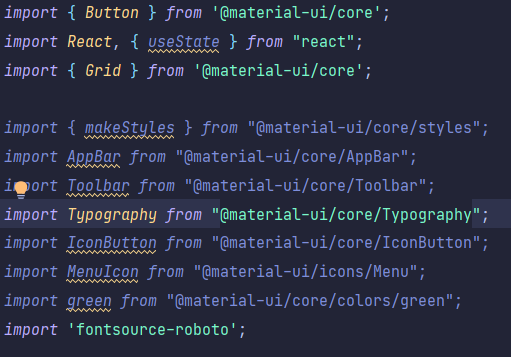


Figure 8 – Material UI components used

After the initial setup and preparation for the implementation was completed, the developer aimed to program the application on a page-by-page basis filling in the content with placeholders.

Every page on the application contains a homebar at the top of the page, which acts as the way for users to navigate across the application from any one page. This was created as its own component to be used on every page of the application. The homebar would have two key components built into it, a menu, and a home button. The menu would have each of the analysis pages as options to navigate to, allowing the user to go to any analysis page from anywhere on the application. Similarly, the home button allows the user to go to the homepage from anywhere in the application.

Starting with the homepage, no major issue crept up in development. First the layout was implemented then filled in with dummy data before filling in with the actual data for presentation. There would be two wordclouds, one for Trump and one for Biden flanking the sides of the page sandwiching a blurb of text introducing the application and project.

The landing page for the analysis contains the most elements of any page in the application with the homebar and four cards with information for each type of analysis done which on-click would take users to the chosen analysis to see. The aim of this page is to act as the landing page continuing from the home page which allows users to see what analysis is available and what they can choose to see.

Each of the pages for the analysis follows the same layout, the only difference between each being the number of graphs and analysis to present. There would be the homebar, then the graphs on the left and information and commentary concerning that graph on the right of it. The only analysis page with a different layout is the word analysis page where it puts graphs side to side but with text talking about the graphs above them.

It was decided that there was no need to program a database for dataset, the graphs, images or content as since the capacity is minimal, they could be hosted directly with the source code.

In terms of the backend infrastructure, Node.js was installed and used Webstorms key feature of auto-recognition which automatically set up and integrated Node.js into the application. The routing was simple and would simply re-route to the application to the appropriate page whenever the appropriate linked was clicked.

For all the pages with commentary and spoken analysis on the plots, the conclusions came directly from the generated graphs.

## Backend Application

In terms of the dataset to use, rather than generate a dataset on the database, it made more sense to use an already tested and tried dataset of the presidential debates. The dataset was found on Kaggle [27] which provided the debate data in terms of the speakers, what was spoken and the timestamp in a csv format providing a csv for each of the 2 debates.

For the implementation of the backend application, this was split into a rough to-do checklist, where in the development process once an item would be chosen to developed and the necessary functions would be programmed, on completion it would be checked off the list.

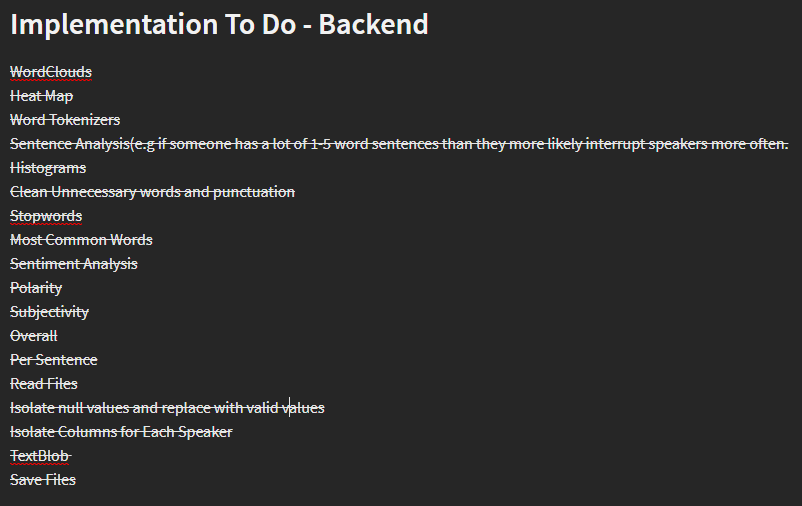


Figure 9 – Checklist for Backend Application

The structure that the application had was a functional programming paradigm to create the application. This helped make the application more modular and break the code down into separate functions make the development cycle easier to manage. The program had a main function which would be used to run the application by which it would call relevant functions in the code to carry out the necessitated operations.



Figure 10 – main() function runs application

For the visualisations that the application generated, the Plotly library was used. Plotly was chosen as it is a free and open-source software so also has a large number of documentation and a good community for any help and guidance for use. Another benefit was that Plotly can be used offline to generate graphics, making it an appealing library to develop with when travelling as was the case during the project often. Matplotlib was also used to create the more static plots as it had graphical options suitable for this project such as its WordCloud visualisation. In terms of the visualisations to use, WordClouds, Histograms and bar or graphs were to be used.



Figure 11 – Example WordCloud visualisation (source:[TowardsDataScience](https://towardsdatascience.com/word-clouds-without-context-3a71b6dc3e2d))

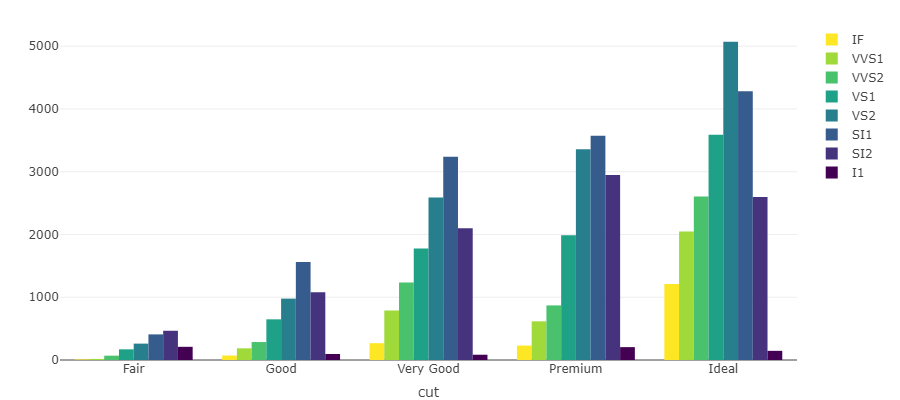


Figure 12 – Example Histogram visualisation (source: [Plotly](https://plotly-r.com/bars-histograms.html))

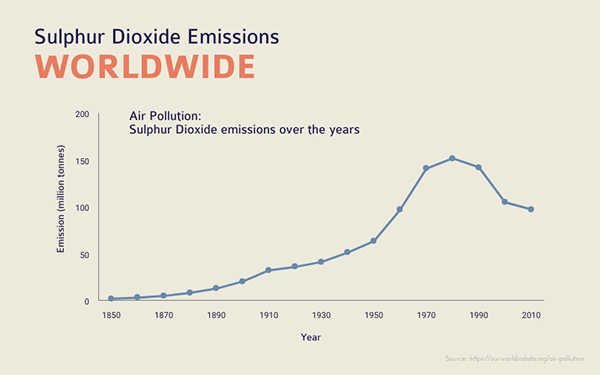


Figure 13 – Example Line Graph visualisation (source: [Visme](https://www.visme.co/line-graph-maker/))

The backend was programmed on a function-by-function process, meaning programming it one function at a time and then applying that function to the main whenever possible.

In terms of generating the graphs, they were broken down into functions. One function which is responsible for taking in the necessary data as parameters and then generating the visualisation for it, that function is then called in a further master function which is responsible calling and creating all the needed instances for that particular visualisation. An example of this is the Heatmaps() and doHeatmaps() function, where Heatmap() is responsible for generating the heatmaps and doHeatmaps() calls Heatmap() for however many times is needed to generate the required instances (see figure 14). All of the code is called and executed at the highest level in a main function which is responsible for everything.

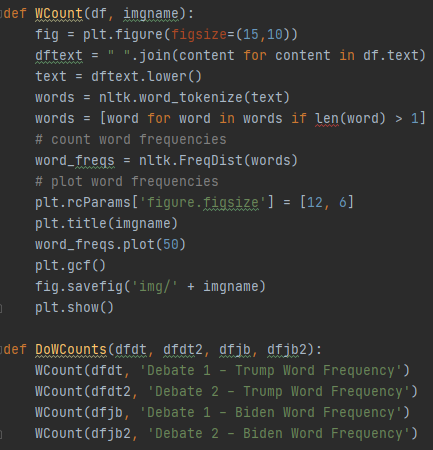


Figure 14 – Code extract of two connected functions

NLTK (or Natural Language Toolkit) was chosen as the platform to be used for working with the language data to build the application. NLTK was used as it provides a host of features useful for development. One of these features is that NLTK provides the use of easy-to-use interfaces to a range of corpora and lexical resources and comes with a whole suite of text processing libraries to use for classification, tokenization, stemming, tagging, parsing, and semantic reasoning with also the benefit of having good community support. NLTK also has the benefit of having useful guides, comprehensive API documentation making it specifically useful for the researcher in learning to use it as they had never practiced with natural language data analysis before [28]. Shown in (figure 15) is an example of how the application used NLTK’s word tokenizer to prep the dataframes for analysis.

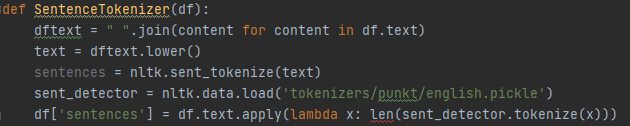


Figure 15 – NLTK code sample

One big problem encountered with the development of the programming was that the datasets timestamp field was not working well with the code for analysis as it was inconsistent with there being entries where the timestamp reset to zero again or null, presumably representing the part two of the debate, which is not useful for the sake of this application and caused issues in generating working analysis and visualisations. This was fixed by first locating the null values which represented this in the code and changing them to the starting point of 00:00. To fix the issue of using the default time format of “00:00” in analysis, that was converted into seconds and minutes which were then added as their own fields to the dataframe. (see figure 16)

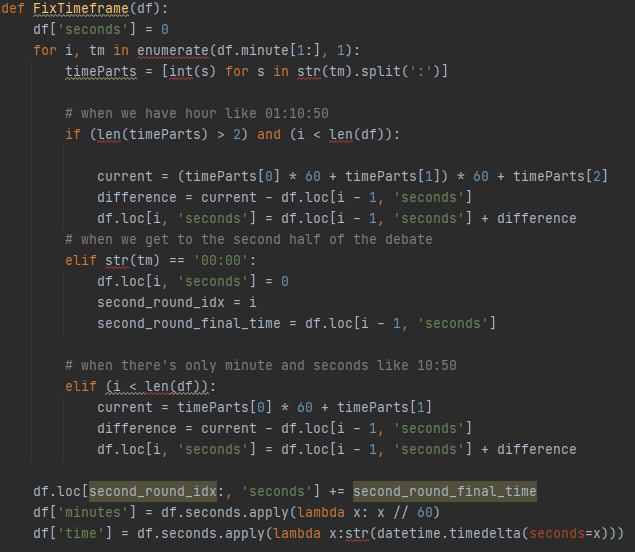


Figure 16 – Code samples for fixing timeframe issue

For sentiment analysis TextBlob’s sentiment analysis library was used. Textblob library provides a simple API for NLP task (see figure 16). [29]

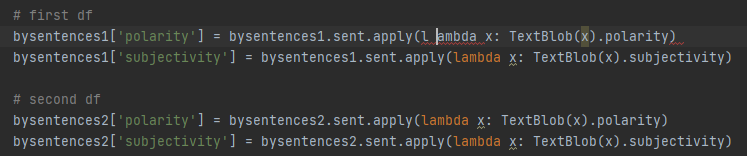


Figure 17 – Sentiment analysis code sample

## 5.3 Version Control and Testing

To aid and to allow writing code across different devices (especially while considering remote working), a version control system was used for the project. GitHub and Git was the version control system of choice, as this is what the author had been using throughout their studies not to mention how it is one of the most widely used version control systems in industry around the world. They also integrate well into the IDE and deployment choices and support good features for continuous integration. The alternative which was considered was Subversion (SVN), as the author had never used it before and the idea of getting more familiar and learning it was appealing, however they chose to stick with Git due to comfortability.

Originally a GitHub branching plan was written., so to simulate how the project would work in a group environment and how most of industry operate with the master branch being the main trunk with feature branches coming off it and merging back into the master branch when completed. This was the plan made, as since there is only one developer in this project, there is no need to over complicate things and following the standard practices is suitable enough. This idea came about from research into development models where this method seemed most suitable for this project where the goal is to be structured similar to industry implementations of source control. [24].

The following is a comparison of the planned branch structure against the actual branch structure. Overall, the plan was followed well albeit with a low number of branches but with one major issue. The issue was found in the deployment stage which saw the frontend branch being unable to merge into the main branch (figure 19). The reason wasn’t clear but after research the most likely reason was due to there being conflicting Heads on the tree. To try to solve this problem. The project was split up into two repositories, one for the backend application and the other for the frontend application.

A key thing to note that since this was a solo project some things which would be ideal in your typical group project were not put into practice. A protected branch was not setup for this project but would be in typical projects with more than one developer which would protect the master branch by ensuring that reviews were done by others prior to any branch being merged.

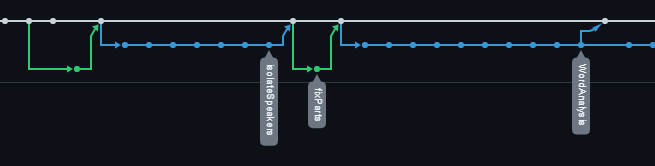


Figure 18 – Branching in action

For testing the Python application, manual or exploratory testing was used without a specific test plan, testing with each iteration whether all the features worked or not. For the web application, there was no deeper level testing required other than testing to make sure the application renders as expected, and all the links and content is or works as expected according to the design. This was done with each iteration manually. Although manual testing had the downside of requiring more time than automated testing, the impact of this was lessened due to the longer timescale available for the project.

## Deployment

The deployment saw some issues arise. Originally, the application was to be hosted on Github pages however some issued arose where the application build would not compile or build for the host location even though the local development host worked. This seemed to be connected to the issue where the developer was unable to merge the front end branch into the main for some reason.

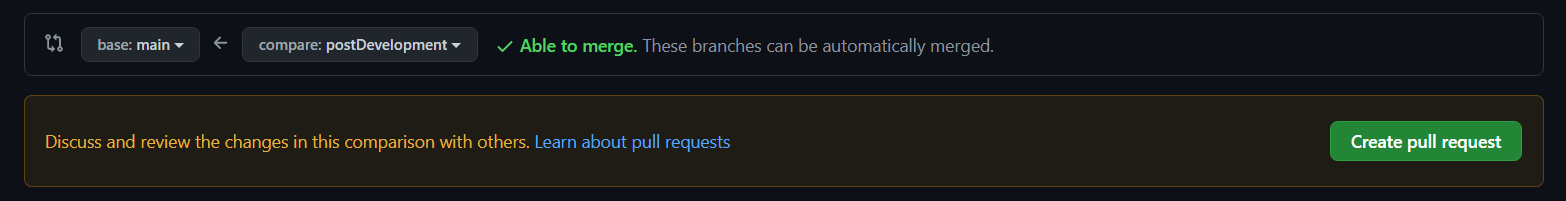


Figure 19 – Pictured top is what should be shown when creating pull requests. Pictured bottom is what was shown when trying to merge frontend branch.

Considering this problem which was unable to be solved even after the attempted solutions to fix this, the switch was made to use Firebase as deployment. Firebase was actually simpler to use and required less setup than Github pages to prepare and host. Using the firebase command line interface was simple an easy to use especially when taking into account the vast number of guides and documentation available for consultation. One issue did prop up here when although the host was successful, the wrong page would be displayed on the application. This was solved as an error was made in the setup process which overwrote the folder to use for rendering where it used the sample folder public instead of the build folder for the application to render. This was fixed by changing the host path to build in the host settings in the firebase,json file.

<https://honours-project-fa3b1.web.app/>

*.*

# Evaluation

Before testing for recruitment could take place, certain rules and regulations had to be met. The researcher read the Universities Code of Practice for Non-Clinical Research Ethics on Human Participants and also had read all the documentation regarding ethics provide by the project supervisor. The first thing was to ensure that the Ethical approval for User Involvement was read, making sure to adhere to the guidance implicated in it where applicable. Typically, an ethics application is required to be prepared and submitted to the ethics committee, with the application describing the relevant procedures intended to be adopted which should then be reviewed and approved. However, for this project, a group application was submitted to the ethics committee on behalf of all the researchers running a project, which is intended to cover the user-centric needs for the majority of the projects. This was the case for this project, at this was a low-risk project and only straightforward user engagement activities were used. For the recruitment of participants, external recruitment wasn’t required as the researcher was able to ask their family and friends to take part in the evaluation of the project. This decision was made due to the restraints of Covid, where it allowed for testing to be done in person with people in the researcher’s quarantine bubble.

For the user evaluation, the researcher was able to get their whole family to participate so four participants. All of which have no experience with testing, so this helped with the testing of the usability as they held no technical bias or understanding on what good usability is, all evaluation would be authentic to their views. This helped in getting feedback of the usability of the application for an outsider’s perspective.

The evaluation was done as a focus group, where each of the four people took part. This was done with one moderator and the 4 participants. The goal of this focus group was to get a good grasp and evaluation on how the UX/UI of the front facing application would be received by users. This type of evaluation was beneficial as since the participants were in the researcher’s bubble, the added benefit of in person testing in being able to respond to body language and communication tone was possible. The focus group kicked off by introducing the project, what the application was, what was working at the time of testing and what the aim of the focus group was. The participants were sent to a set up desktop pc which had the front-end application running, and they were to set and explore the application, trying it themselves. The intention of the researcher of this was to see how the participants reacted to the use of the application and whether it was intuitive to by themselves without any guidance. After around a five-minute testing process, a conversation was had between the tester and the researcher with the researcher taking note of what was said and any key points.

The feedback in general was pretty positive. All participants agreed that the UI was intuitive and simple to navigate and use. Some negative feedback was brought up in the feedback as well, most of which were concerning the overall aesthetics of the application. A point was made that the spacing on the analysis pages of the application weren’t particularly pleasing and they would like to have seen more spacing or margins around the graphs or visualisation. They also gave the feedback that they would like it if the visualisations on the application were more interactive and not static.

In terms of evaluation concerning the backend application, due to time constraints sufficient evaluation of the application could not be done in terms of evaluating the results in terms of accuracy. Ideally the backend application would be evaluated, by doing the analysis with a different model other than NLTK for example using a BERT model and cross refencing the results to gauge the accuracy of the analysis used in the application. This would require programming and running another model to run the same analysis and create the same visualisations to run the comparisons. With priority being placed on creating a front end application to present the results over evaluating the results, not enough time was left in the project to do so.

# Description of the final product

The final resulting application matched very accurately to the prototypes generated during the design phase of the project.

When the user launches into the application, they are greeted by a homepage which has two word clouds of Biden and Trump with a written entry about the application front and centre of the page and a button centred below it which when clicked takes the user to the analysis home page, with also a homebar containing the title of the project, a home button and the menu button which drops down a small menu with buttons that link directly to the listed analysis option. This menu is prevalent on every page of the application.

The analysis homepage contains a title detailing what page it is followed by 4 cards in a grid. Each card in the grid represents each of the analysis types available for viewing so there is a card each for sentiment analysis, sentence analysis, word analysis and heatmaps. Present on the cards are a background image, a title titling the type of analysis that card is for and a brief summary for the analysis of that card. When a card is clicked, that takes the user to the corresponding analysis page.

Excluding the word analysis page; the sentiment analysis, heatmaps and sentence analysis pages follow the same structure. Below the homebar there is a title, titling the type of analysis that page is focused on. Below that is a grid where the visualisation is situated on the left, with accompanying content and commentary concerning that visualisation on the right. This structure is the same for however many graphs each analysis page has so if a page has two visualisations to present, each is given its own row. The word analysis page while similar in content to the other pages, follows a different structure. It has the title under the homebar like the others, instead of having a piece of commentary for each visualisation, there is one big piece of commentary below the title which applies to all the graphs. The graphs are also visualised in a different layout so side by side where each row represents a candidate, so row one is Biden and row 2 is Trump.

However, to achieve the above, first the backend application had to be implemented. The backend application is responsible for doing all the leg work of the analysis and visualisations. The program first reads in the dataset from the saved file and extracts the needed information into a dataframe. It then uses that dataframe to perform the needed analysis and to create the required data visualisations. It executes the analysis and creates data visualisations for sentiment, sentence level and word level analysis and saves them as image file, those images are then used in the front-end application.

# Summary and Conclusions

## What the Author Achieved

The researcher was able to produce 2 fully functional applications, one on the backend for generating analysis and visualizations, with the other as the front facing application for presenting the findings of the first application. The first application successfully can read in the data of the Presidential Debates to then apply argument technology techniques to perform analysis on them. For sentiment analysis, the application analyses the polarity score of the debate for the candidates and the subjectivity score regarding the content of what was said in the debate by each speaker. In terms of word analysis, the application is able to generate word clouds and word counts analyzing the popular language which was used in the debate. In terms of sentence analysis, the application is able to analyze how many speakers speak in each minute in order to generate heatmaps which reflect the heat of the debate to allow readers to visualize how chaotic the debate really was. The length of sentences and sentence count was also analysed to see whether any conclusions surrounding the sentences could be found. On the front-end application, an interface to allow users to see the generated graphs and analysis was created. The application allows for users to easily navigate through the application to see the different types of analysis done. Next to each analysis artifact, there is also the addition of some commentary to provide some insight as to what the analysis and graphs are suggesting.

## Lessons Leaned

Throughout the project, the author learned a number of skills and technologies which they had not previously held or had used before the start of the project. More knowledge was learnt about Javascript, the React framework, HTML/CSS and the use of external libraries such as material ui. There was also to opportunity to expand on the authors Python skills with regards to data science and data visualization techniques using Python. Finally, a lot was learned on how to manage a longer form project over an extended period of time whilst also juggling other responsibilities and the added difficulty of doing so during a worldwide pandemic causing unforeseen circumstances.

## Did all of the Decisions Work Out?

In terms of the decisions for the project, for the most part they stayed intact from the start of the project to the end with very minimal changes. The only noticeable change was the change of the target device for presenting the data which came about due to Covid, affecting access to the intended monolith display causing the project to change the end device to mainly Laptops or Desktops.

## What would be recommended to others?

An important thing the author would recommend is to take time at the beginning of the project to plan things out and ensure that there is a path of success to follow and clear outcomes set to achieve. It is also important to regularly self-reflect on the project, its progress and on any difficulties or positives found over the duration of the project.

## Future Work

Future iterations of the project and applications would include improvements and features which were not included in this iteration of the project. One key thing would be to have more forms of analysis done to the data for example with sentiment analysis, another aspect to potentially expose could be to analyze the sentiment in terms of intensity for the spoken words. To give an example for word analysis, bigrams could be a field to be analyzed. Improvements could also be made to the front facing application responsible for presenting the analysis, more attention could be made to the aesthetics of the applications to it look more infographic like. Another improvement to be made here could be to also have a related reading section where links could be provided to similar or related analysis elsewhere where if the reader is interested could investigate.

## Acknowledgments

The author would like to give thanks and gratitude to their parents for providing an excellent support system and help with helping them manage the pressures of this project during these difficult times of the pandemic and its subsequent impact on society and myself.

The author would like to thank their supervisor Jacky Visser for providing tremendous help and support throughout the project, offering great insight, advice, and commentary over the year. This thanks is also extended to the author’s peers who participated with them in the weekly advisor meetings with Jacky, in Mo Nadat and Christian Hegarty for providing good discussion and also general re-assurance amongst peers concerning the project and studies in general.

The author would like to thank their friends, they know who they are, for providing support and needed breaks and pauses, which allowed the development of this project to be a lot less stressful than it could have been.

# References

[1] Nytimes.com. 2021. *Presidential Election Results: Biden Wins*. [online] Available at: <https://www.nytimes.com/interactive/2020/11/03/us/elections/results-president.html> [Accessed 17 April 2021].

[2] Baccini, L., Brodeur, A. and Weymouth, S., 2021. The COVID-19 pandemic and the 2020 US presidential election. *Journal of Population Economics*, 34(2), pp.739-767.

[3] BBC News. 2021. George Floyd: What happened in the final moments of his life. [online] Available at: <https://www.bbc.co.uk/news/world-us-canada-52861726> [Accessed 17 April 2021].

[4] Hill, E., Tiefenthäler, A., Triebert, C., Jordan, D., Willis, H. and Stein, R., 2021. How George Floyd Was Killed in Police Custody. [online] Nytimes.com. Available at: <https://www.nytimes.com/2020/05/31/us/george-floyd-investigation.html> [Accessed 17 April 2021].

[5] BBC News. 2021. Capitol riots timeline: The evidence presented against Trump. [online] Available at: <https://www.bbc.co.uk/news/world-us-canada-56004916> [Accessed 17 April 2021].

[6] BBC News. 2021. Presidential debate: How the world's media reacted. [online] Available at: <https://www.bbc.co.uk/news/election-us-2020-54354405> [Accessed 17 April 2021].

[7] BBC News. 2021. Presidential debate: Who won the Trump-Biden clash?. [online] Available at: <https://www.bbc.co.uk/news/election-us-2020-54345192> [Accessed 17 April 2021].

[8] Zurcher, A., 2021. Presidential debate: Key takeaways from the Trump-Biden showdown. [online] BBC News. Available at: <https://www.bbc.co.uk/news/election-us-2020-54650681> [Accessed 17 April 2021].

[9] Chowdhury, G. (2003) Natural language processing. Annual Review of Information Science and Technology, 37. pp. 51-89.

[10]Doaa Mohey El-Din Mohamed Hussein., 2018. A survey on sentiment analysis challenges, Journal of King Saud University - Engineering Sciences, Pages 330-338.

[11] GOV.UK. 2021. Data protection. [online] Available at: <https://www.gov.uk/data-protection> [Accessed 18 April 2021].

[12] Boyd, K., 2021. Copyright Law Basics For UK Software Developers — Smashing Magazine. [online] Smashing Magazine. Available at: <https://www.smashingmagazine.com/2018/03/copyright-law-basics-for-uk-software-developers/> [Accessed 18 April 2021].

[13] Herbert, P., 2021. Intellectual Property Act 2014: A Summary of the Key Provisions - Goodman Derrick LLP. [online] Goodman Derrick LLP. Available at: <https://www.gdlaw.co.uk/site/blog/sectors-blog/media-blog/intellectual-property-act-2014-a-summary-of-the-key-provisions> [Accessed 18 April 2021].

[14] Vuejs.org. 2020. Comparison With Other Frameworks — Vue.Js. [online] Available at: <https://vuejs.org/v2/guide/comparison.html> [Accessed 26 October 2020].

[15] Tutorialspoint.com. 2020. Node.Js - Introduction - Tutorialspoint. [online] Available at: <https://www.tutorialspoint.com/nodejs/nodejs\_introduction.htm> [Accessed 18 April 2021].

[16] Djangoproject.com. 2021. Django overview | Django. [online] Available at: <https://www.djangoproject.com/start/overview/> [Accessed 18 April 2021].

[17] Pallets. 2021. Flask. [online] Available at: <https://palletsprojects.com/p/flask/> [Accessed 18 April 2021].

[18] Tutorialspoint.com. 2020. Mongodb - Advantages - Tutorialspoint. [online] Available at: <https://www.tutorialspoint.com/mongodb/mongodb\_advantages.htm> [Accessed 26 October 2020].

[19] Knight, S., 2020. Mongodb Vs SQL. [online] Knowi. Available at: <https://www.knowi.com/blog/mongodb-vs-sql/> [Accessed 26 October 2020].

[20] GitHub. 2020. Uberspot/Opentriviaqa. [online] Available at: <https://github.com/uberspot/OpenTriviaQA> [Accessed 26 October 2020].

[21] upGrad blog. 2021. Top 6 Data Science Programming Languages 2021 [Hand-Picked] | upGrad blog. [online] Available at: <https://www.upgrad.com/blog/data-science-programming-languages/#:~:text=1.-,Python,language%20in%20the%20world%20today.&text=Natural%20data%20processing%20and%20data,by%20creating%20a%20CSV%20output.> [Accessed 19 April 2021].

[22] Team, L., 2021. What the Waterfall Project Management Methodology Can (and Can’t) Do for You. [online] Lucidchart.com. Available at: <https://www.lucidchart.com/blog/waterfall-project-management-methodology> [Accessed 19 April 2021].

[23] Material Design. 2021. Material Design. [online] Available at: <https://material.io/design> [Accessed 19 April 2021].

[24] Trunkbaseddevelopment.com. 2021. Trunk Based Development. [online] Available at: <https://trunkbaseddevelopment.com/> [Accessed 20 April 2021].

[25] Code Faster with Kite. 2021. Kite - Free AI Coding Assistant and Code Auto-Complete Plugin. [online] Available at: <https://www.kite.com/> [Accessed 22 April 2021].

[26] JetBrains. 2021. WebStorm: The Smartest JavaScript IDE by JetBrains. [online] Available at: <https://www.jetbrains.com/webstorm/> [Accessed 22 April 2021].

[27] Kaggle.com. 2021. US Election 2020 - Presidential Debates. [online] Available at: <https://www.kaggle.com/headsortails/us-election-2020-presidential-debates> [Accessed 20 April 2021].

[28] Nltk.org. 2021. Natural Language Toolkit — NLTK 3.6.2 documentation. [online] Available at: <https://www.nltk.org/> [Accessed 22 April 2021].

[29] Textblob.readthedocs.io. 2021. TextBlob: Simplified Text Processing — TextBlob 0.16.0 documentation. [online] Available at: <https://textblob.readthedocs.io/en/dev/> [Accessed 23 April 2021].

[30] Vu, D., 2021. Generating Word Clouds in Python. [online] Datacamp. Available at: <https://www.datacamp.com/community/tutorials/wordcloud-python> [Accessed 25 April 2021].

[31] Luvsandorj, Z., 2021. Simple wordcloud in Python. [online] Medium. Available at: <https://towardsdatascience.com/simple-wordcloud-in-python-2ae54a9f58e5> [Accessed 25 April 2021].

[32] Krasner, E., 2021. How to choose between rule-based AI and machine learning. [online] TechTalks. Available at: <https://bdtechtalks.com/2020/06/05/rule-based-ai-vs-machine-learning/> [Accessed 25 April 2021].

[33] Create-react-app.dev. 2021. Deployment | Create React App. [online] Available at: <https://create-react-app.dev/docs/deployment/> [Accessed 25 April 2021].

[34] Solomon, B., 2021. Python Plotting With Matplotlib (Guide) – Real Python. [online] Realpython.com. Available at: <https://realpython.com/python-matplotlib-guide/> [Accessed 25 April 2021].

[35] Guru99.com. 2021. NLTK Tokenize: Words and Sentences Tokenizer with Example. [online] Available at: <https://www.guru99.com/tokenize-words-sentences-nltk.html> [Accessed 25 April 2021].

[36] Selvaraj, N., 2021. A Beginner’s Guide to Sentiment Analysis with Python. [online] Medium. Available at: <https://towardsdatascience.com/a-beginners-guide-to-sentiment-analysis-in-python-95e354ea84f6> [Accessed 25 April 2021].

[37] Kuzminykh, N., 2021. Sentiment Analysis in Python With TextBlob. [online] Stack Abuse. Available at: <https://stackabuse.com/sentiment-analysis-in-python-with-textblob/> [Accessed 25 April 2021].

[38] Montoyo, A., Martínez-Barco, P. and Balahur, A., 2012. Subjectivity and sentiment analysis: An overview of the current state of the area and envisaged developments. Decision Support Systems, 53(4), pp.675-679.

[39] Feoktistov, I., 2021. 7 Advantages of Node.js for startups. [online] Relevant Software. Available at: <https://relevant.software/blog/7-benefits-of-node-js-for-startups/> [Accessed 27 April 2021].

# Appendices

* Appendix A – Mid Term Report
* Appendix B – Researcher Declaration Form
* Appendix C – UX Wireframes
* Appendix D – Focus Group
* Appendix E – UX Prototypes
* Appendix F – Frontend Application Flowchart
* Appendix G – Source Code
* Appendix H – Functional Requirements
* Appendix I – Frontend Application Screenshots
* Appendix J – GANNT Chart
* Appendix K – Web application Link
* Appendix L – User Guide